FAIRNESS ADEQUACY TEST FOR MACHINE LEARNING SYSTEMS

by

Kehinde Oluwasayo Akinola

December, 2025

Director of Thesis: Srinivasan Madhusudan, PhD

Major Department: Computer Science

As Machine Learning (ML) systems assume a larger role in decisions that affect people's lives, such as who receives a loan, access to healthcare, or early release from prison, ensuring these systems are fair is more important than ever. However, current fairness checks often miss the subtle and complex ways in which bias can appear in algorithms.

This thesis tackles that gap by proposing a practical framework for testing how well machine learning models meet fairness standards, especially across protected groups. Instead of relying solely on standard performance metrics, the approach combines statistical tools with stress testing techniques to uncover hidden or overlooked biases.

There are four main contributions. First, we introduce a fairness adequacy test using metrics like Equal Opportunity Difference (EOD) and Equalized Odds Metrics (EOM) to examine disparities in error rates across groups. Second, we apply mutation testing by altering sensitive features such as race or gender to see how model outputs change, helping assess fairness under different conditions. Third, we use permutation methods to simulate edge cases and test how models respond to unusual or extreme inputs. Finally, we validate this approach with real-world case studies in areas like healthcare, finance, and criminal justice, where fairness is especially critical.

By offering a clear and testable way to evaluate fairness, this work aims to support the development of more trustworthy, accountable, and equitable AI systems

FAIRNESS ADEQUACY TEST FOR MACHINE LEARNING SYSTEMS

A Thesis

Presented to The Faculty of the Department of Computer Science

East Carolina University

In Partial Fulfillment of the Requirements for the Degree

Master of Science in Data Science

by

Kehinde Oluwasayo Akinola

December, 2025

Committee Members
Director of Thesis: Srinivasan Madhusudan, PhD
Peng Kebin, PhD
John Reisch, PhD

Table of Contents

LIST OF TABLES

LIST OF FIGURES

List of Acronyms

AI Artificial Intelligence. xii

AUC Area Under the Curve. xii

CART Classification and Regression Trees.. v, xii, 13, 14, 19, 38, 45, 50, 56, 57, 66, 79

CNNs Convolutional Neural Networks. xii

DBSCAN Density-Based Spatial Clustering of Applications with Noise. xii

DNNs deep neural networks. xii

DP Demographic Parity Difference. xii, 40

EO Equality of Opportunity. xii

EOD Equal Opportunity Difference. xii, 1, 10, 17, 19, 26, 38, 40, 43, 44, 46–51, 54–59, 61, 63, 65–69, 71–77

EOM Equalized Odds Metrics. xii, 1, 10, 17, 19, 26, 33, 38, 40, 44, 46–51, 54–60, 62, 64–77

FAT Fairness Adequacy Test. xii, 28, 38

FPR False Positive Rate. xii, 40, 44

K-NN K-Nearest Neighbors. v, xii, 13, 14, 19, 22, 37, 38, 45, 50, 56, 57, 66, 79

LightGBM Light Gradient Boosting Machine. xii

LR Logistic Regression. v, xii, 13, 19, 38, 45, 50, 56, 57, 66, 79

MKR mutation kill rates. xii, 18, 19

ML Machine Learning. xii, 1–3, 5–8, 15

MSR Mutation Score Rate. xii

PCA Principal Component Analysis. xii

Q-Learning Quality Learning. xii

RNNs Recurrent Neural Networks. xii

ROC Receiver Operating Characteristic. xii

RQ Research Questions. xii

SARSA State–Action–Reward–State–Action. xii

SPD Statistical parity difference. xii, 10, 17, 19, 25, 38, 44

SVM Support Vector Machines. xii

t-SNE t-distributed Stochastic Neighbor Embedding. xii

TPR True Positive Rate. xii, 40, 43

UMAP Uniform Manifold Approximation and Projection. xii

WMFI Weighted Multi-Objective Fairness Index. xii, 38, 40

XGBoost Extreme Gradient Boosting. xii

Chapter 1

Introduction

1.1 Motivation: Why Fairness Needs Testing

A model can report high predictive performance and still produce systematically different outcomes for protected groups. In high-stakes domains such as credit, employment, criminal justice, and healthcare, these disparities translate into real harm. Standard metrics such as accuracy, precision, and recall can hide subgroup failures caused by historical bias, proxy labels, or distribution shift (13; 18). As a result, fairness must be treated as a testable property of machine learning (ML) systems, not only as a post-hoc diagnostic.

Fairness testing evaluates whether outcomes differ across groups (for example, by gender, race, or age) in ways that are not justified by task-relevant differences (7; 19). Practical auditing tools such as FairTest and Themis demonstrate how disparity detection can be operationalised using statistical tests and subgroup exploration (19). However, many evaluations still rely on a single fixed test set and a small set of static metrics, which can miss fairness regressions under plausible changes in data and pipelines.

1.2 Fairness Incidents Motivating Stronger Evaluation

Multiple real-world incidents illustrate the cost of inadequate fairness evaluation:

- **Criminal justice: COMPAS (2016).** Analyses reported that Black defendants were more often labelled high risk than White defendants, raising concerns about unequal error rates across groups (2; 4).
- **Hiring: Amazon recruiting prototype (2018).** Reports described a resume-screening prototype that learned patterns from male-dominated historical data and penalised women-associated terms (6).
- **Healthcare: risk scoring (2019).** A widely used risk model used healthcare cost as a proxy for clinical need, under-referring Black patients despite comparable health status (15).
- **Public benefits: Dutch childcare scandal (2018–2021).** Automated profiling contributed to disproportionate targeting and misclassification of families, highlighting the need for accountability and safeguards (16; 10).

These cases are not isolated technical mistakes. They show recurring evaluation failures: weak slice coverage, limited stress testing, and a lack of audit-ready evidence that a system would detect fairness-relevant faults before deployment.

1.3 Testing and Adequacy: From Code Coverage to Fairness Detection

In classical software engineering, test adequacy asks whether a test suite is capable of revealing faults, not merely whether it executes successfully. In ML, this idea must be adapted because models learn from data, predictions are probabilistic, and the operating environment can shift. Adequacy therefore moves from *code coverage* toward

evaluation strength: whether the evaluation process can reliably detect violations of fairness requirements under realistic perturbations (8; 18).

In this thesis, we treat fairness as a test adequacy requirement. Instead of reporting fairness metrics only at baseline, we stress the system with controlled, realistic changes and ask a sharper question:

> *Given fairness-relevant perturbations, does the evaluation detect meaningful fairness shifts, or do these shifts survive undetected?*

1.4 Our Approach: Fairness Adequacy Testing via Mutation

We operationalise fairness adequacy using mutation testing. Mutation testing originates in software engineering as a strong adequacy criterion: introduce small faults (mutants) and check whether the test suite detects them (14). In ML, related ideas appear in metamorphic testing, where input transformations are used to test properties when an exact oracle is unavailable (12).

This thesis adapts the mutation-testing mindset to fairness evaluation:

- We generate fairness-relevant mutants by applying small, controlled perturbations to data and pipeline elements under realism constraints.
- We apply a **mutant-validity gate** to filter unrealistic or trivial mutants.
- We compute fairness metrics on a fixed held-out test set and measure **fairness drift** between baseline and mutant outcomes.
- A mutant is **KILLED** if its fairness drift exceeds a data-driven detection threshold; otherwise it is **ALIVE**.

To separate genuine fairness signals from random variability, we derive a detection threshold from baseline null variability using bootstrap resampling. Concretely, for a fairness metric f, we define a control limit:

$$\theta_f = \mu_\Delta + 1.5 \cdot \sigma_\Delta,$$

where μ_Δ and σ_Δ are the mean and standard deviation of the baseline drift distribution estimated under resampling. A mutant is considered detected (killed) when $|\Delta f| > \theta_f$.

1.5 Research Questions

This thesis addresses three research questions:

RQ1: How effective are fairness-aware mutation operators at injecting meaningful fairness faults that can be detected by fairness evaluation?

RQ2: How consistently does the fairness adequacy framework behave across datasets, model families, and test-set sizes?

RQ3: How does mutation-based fairness adequacy compare to a diversity proxy baseline (diameter) that measures test-set spread rather than fairness-specific fault detection?

1.6 Contributions

This thesis makes the following contributions:

- A fairness adequacy testing framework that adapts mutation testing to fairness

evaluation, classifying mutants as *KILLED*, *ALIVE–Equivalent*, or *ALIVE–Non-Equivalent* based on fairness drift.

- A set of fairness-aware mutation operators for tabular datasets, designed to induce fairness-relevant perturbations while filtering unrealistic or trivial cases through a mutant-validity gate.
- An empirical evaluation on Adult and COMPAS across three model families (Logistic Regression, Decision Tree, and k-NN), reporting how adequacy varies by dataset, model, metric, and test-set down-sampling.
- A comparison between mutation-based adequacy and a diversity baseline (diameter), clarifying when geometric diversity fails to capture fairness-specific evaluation strength.

1.7 Thesis Organisation

The remainder of this thesis is structured as follows. Chapter 2 reviews background and related work in algorithmic fairness, fairness evaluation, and mutation testing for ML. Chapter 3 presents the proposed fairness adequacy testing framework, mutation operators, and mutant classification rules. Chapter 4 reports empirical results on Adult and COMPAS and analyses adequacy patterns across models, metrics, and test splits. Chapter **??** discusses threats to validity, future work, and concluding remarks.

Chapter 2

Background

2.1 Why Fairness Matters in Modern Machine Learning

Machine learning (ML) systems increasingly influence access to resources and opportunities in domains where errors carry real consequences. In low-stakes applications such as product recommendations, misclassifications are often annoying. In high-stakes settings, the same mistakes can become harmful, and they may not be distributed evenly across demographic groups (3). This is the core motivation for fairness evaluation: the goal is not only to build accurate models, but also to understand how decisions and errors are distributed across protected and socially salient groups.

Fairness concerns are not abstract. They arise because training data encodes historical patterns, labels may be noisy or proxy-based, and operational choices such as thresholds can shift error rates across groups. Even if a model achieves high overall accuracy, it may still impose a systematic disadvantage on a protected group, particularly if that group is underrepresented or differently distributed in feature space.

2.2 Machine Learning in High-Stakes Decisions

Early ML deployments were largely limited to tasks such as spam filtering and handwriting recognition. Today, ML models are used in pipelines that support or automate decisions in:

- **Finance:** credit scoring, fraud detection, loan approvals, and pricing decisions.
- **Healthcare:** risk stratification, treatment prioritisation, and resource allocation.
- **Hiring:** résumé screening, ranking, and shortlisting.
- **Criminal justice:** risk assessment tools used to support parole and sentencing recommendations.

These applications raise questions of accountability and due process, because the model output is often used as an input to policy decisions. Fairness auditing is therefore increasingly treated as a core evaluation requirement, not merely a research add-on (3).

A practical challenge is that deployed environments are not static. Data pipelines change, populations drift, and operational thresholds are tuned. Empirical evidence shows that predictive reliability can degrade under distribution shift, and fairness metrics can drift as well (17). A model that appears acceptable on a single benchmark test set can behave differently after small but realistic changes in the underlying data process.

2.3 Formal Machine Learning Pipeline and Where Bias Enters

An ML system learns a function

$$f : \mathcal{X} \to \mathcal{Y}$$

from a dataset

$$\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n}.$$

In supervised learning, bias and unfairness can be introduced at multiple points in the pipeline:

- **Data collection:** sampling bias, missingness patterns, and underrepresentation of protected groups.
- **Label construction:** proxies for the target (for example, cost used as a proxy for need) that encode inequities.
- **Feature design:** inclusion of protected attributes or proxies (such as postcode, tenure patterns, or school quality indicators).
- **Preprocessing:** imputation, scaling, and encoding choices that can differentially affect groups.
- **Training:** objectives that prioritise global accuracy can sacrifice minority performance when base rates differ.
- **Deployment:** thresholds, policies, and feedback loops that can amplify disparities over time.

Because fairness failures can arise from many stages, it is often more accurate to treat the system as an end-to-end decision pipeline rather than a single model.

Figure 2.1: Formal machine-learning pipeline: Data collection → Preprocessing → Model training → Evaluation → Deployment.

Note: avoid spaces in figure filenames to prevent LaTeX build issues (for example, rename `Machine Learning Pipeline.png` to `Machine_Learning_Pipeline.png`).

2.4 Fairness Definitions and Confusion-Matrix Notation

This thesis focuses on binary classification, where fairness metrics are most developed and widely used.

Let:

- A be a protected attribute that defines groups (for example, sex or race).
- $Y \in \{0, 1\}$ be the ground-truth label.
- $\hat{Y} \in \{0, 1\}$ be the predicted label.

For a group g (a value of A), define the standard rates:

$$\mathrm{TPR}_g = P(\hat{Y} = 1 \mid Y = 1, A = g), \quad \mathrm{FPR}_g = P(\hat{Y} = 1 \mid Y = 0, A = g),$$

$$\mathrm{PR}_g = P(\hat{Y} = 1 \mid A = g).$$

Fairness metrics compare these rates across groups. The point is not that one metric captures all fairness. Rather, each metric provides a lens, and different lenses reveal different kinds of harm.

2.4.1 Group-level parity metrics

Three group-level metrics appear throughout this thesis:

- **Statistical Parity Difference (Statistical parity difference (SPD)).**

$$\text{SPD} = \text{PR}_a - \text{PR}_b.$$

  This measures differences in overall positive decision rates. It is easy to compute and interpret, but it does not condition on qualification (Y).

- **Equal Opportunity Difference (EOD).**

$$\text{EOD} = \text{TPR}_a - \text{TPR}_b.$$

  This focuses on qualified individuals ($Y = 1$). It answers: do equally qualified individuals receive the positive decision at similar rates?

- **Equalized Odds Metric (EOM).** Equalized odds requires parity in both true positive rates and false positive rates. In practice, audits often report a scalar that summarises how far the model deviates from equalized odds by combining gaps in TPR and FPR across groups. This is stricter than Equal Opportunity in many settings because it accounts for harms from both missed positives and mistaken positives.

These metrics are widely used because they are grounded in observable error rates and can be computed from confusion matrices (9).

2.4.2 Fairness trade-offs and impossibility results

Fairness metrics can conflict. In particular, when groups have different base rates, it can be mathematically impossible to satisfy certain combinations of fairness and calibration simultaneously (4). This matters because it forces an explicit choice of fairness objectives. A system can be well-calibrated yet violate equalized odds, or it can satisfy equal opportunity at the cost of other properties. These results motivate evaluating multiple metrics rather than relying on a single number.

2.4.3 Intersectionality and slice-based evaluation

Fairness audits that compare only broad groups (for example, male vs. female or Black vs. white) can miss intersectional harms. A model can appear acceptable at the coarse level while still disadvantaging a subgroup such as older Black women, younger immigrant men, or other intersections of protected and contextual attributes (11). This motivates slice-based evaluation, where fairness is checked not only at the top-level group but also across relevant subpopulations.

2.5 Sources of Fairness Drift in Practice

Even if a model meets a fairness target at one time, fairness can drift after deployment. Common causes include:

- **Population shift:** the demographic mix or feature distributions change over time.

- **Policy changes:** decision thresholds are tuned for business goals, changing error rates.
- **Feature drift:** new data sources are introduced, features are redefined, or missingness patterns change.
- **Feedback loops:** model decisions influence future data collection, which can amplify disparities.

Robust evaluation therefore requires stress testing the fairness audit itself, not only computing fairness metrics on a fixed snapshot (17).

2.6 Uncertainty, Subgroup Size, and Statistical Stability

Fairness metrics can be unstable when subgroup sizes are small. For example, a protected subgroup with few positive labels can exhibit high variance in TPR, leading to fairness swings that may reflect sampling noise rather than genuine behaviour. This creates two practical risks:

- **False alarms:** declaring a fairness problem when the observed gap is mostly noise.
- **Missed harms:** failing to detect meaningful gaps because the audit lacks power on small subgroups.

A well-designed evaluation therefore benefits from uncertainty-aware reporting, for example using confidence intervals or resampling-based variability estimates. This thesis uses this background motivation to justify a detection framework that separates genuine fairness shifts from expected variability.

2.7 Base Learners Used in This Thesis

Our empirical evaluation uses Logistic Regression (LR), Decision Trees (CART), and k-Nearest Neighbours (K-NN). These choices represent distinct modelling families, and fairness behaviour can differ across them due to inductive bias and sensitivity to feature space geometry.

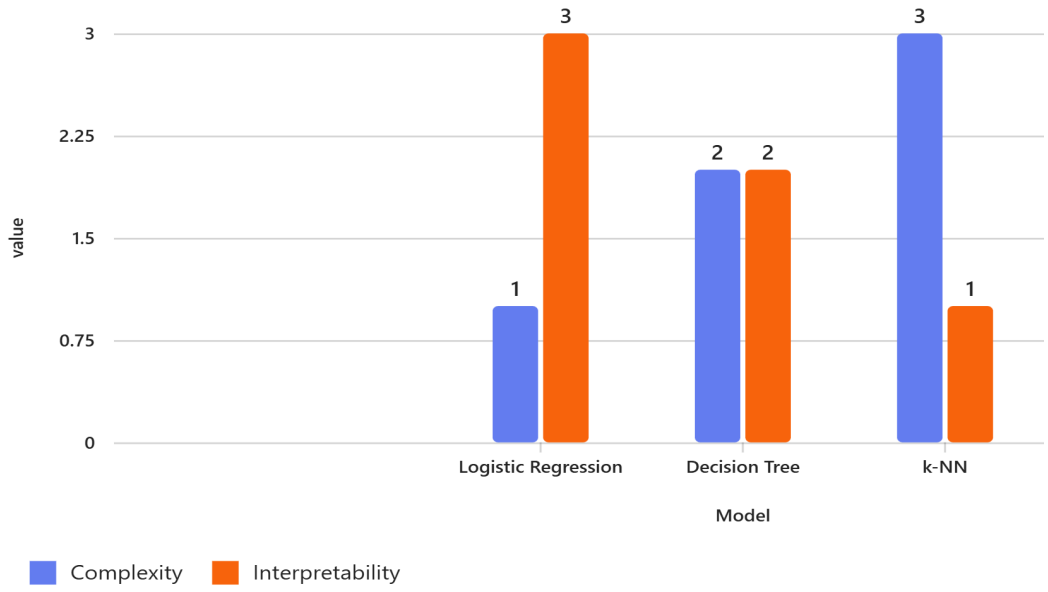

Figure 2.2: Complexity versus interpretability for Logistic Regression, Decision Tree, and k-NN.

2.7.1 Logistic Regression (LR)

Logistic Regression models the probability of a positive outcome using the sigmoid function:

$$\hat{p}(y = 1 \mid x) = \sigma(w^\top x + b), \quad \sigma(z) = \frac{1}{1 + e^{-z}}.$$

It is widely used because it is efficient, stable, and interpretable. In regulated domains, interpretability matters because practitioners may need to justify decisions.

Fairness issues can arise when protected attributes or strong proxies correlate with

the outcome and receive large weights. Because the decision boundary is global and linear, disparities can become systematic, not local. Regularisation and class weighting can change decision geometry, which may alter error-rate gaps across groups.

2.7.2 Decision Trees (CART)

Decision trees partition the feature space through axis-aligned splits. They can capture nonlinear interactions and represent decision logic as rule paths. This is useful for human inspection, but fairness risks can appear when early splits use protected or proxy features. A single split can route most members of a protected group into a branch with consistently different outcomes.

Trees are also sensitive to small changes near split thresholds, which can change the allocation of instances to leaves. As a result, group error rates can shift if subgroup distributions cluster around influential thresholds.

2.7.3 k-Nearest Neighbours (K-NN)

k-Nearest Neighbours is a non-parametric method based on locality (5). Predictions are determined by the labels of nearby training points in feature space. This makes K-NN strongly dependent on distance metrics and preprocessing (normalisation and encoding).

Fairness issues can arise when a protected group is sparse in relevant regions of feature space. In such cases, minority instances can be repeatedly outvoted by majority neighbours. Because K-NN relies directly on distances, it can also be sensitive to feature noise and scaling differences.

Table 2.1: Big-O summary for common implementations (n samples, d features).

	Training	Query	Memory
k-NN	$O(1)$	$O(nd)$	$O(nd)$
Decision Tree	$O(nd \log n)$	$O(\log n)$	$O(\#\text{nodes})$
Logistic Regression	$O(nd^2)$	$O(d)$	$O(d)$

2.8 Fairness Auditing Tools and Their Limits

A range of practical tools support fairness audits by computing group disparities and searching for statistically significant differences. Examples include FairTest and Themis, which focus on identifying discriminatory outcomes across groups and slices (19; 7). These toolkits are useful for diagnosing bias on a given dataset and model snapshot.

However, many audits remain largely static. They report fairness metrics on a fixed test set and may not directly answer a testing question: if the model or data pipeline changes in a realistic way, will the evaluation reliably detect a fairness regression? This limitation motivates adequacy-style evaluation.

2.9 Testing, Adequacy, and Mutation Testing for ML

Testing in ML differs from classical software testing because of the oracle problem and the probabilistic nature of model behaviour. Property-based and metamorphic testing have been proposed to validate model behaviour when ground truth is incomplete or when correctness should be invariant under certain transformations (18).

Mutation testing is a strong adequacy criterion in software engineering: it introduces small, controlled faults (mutants) and checks whether the test suite detects them (1). Recent work has adapted mutation testing ideas to machine learning, including deep learning, to stress evaluation beyond a single accuracy report (12).

The background lesson is that an evaluation can appear strong while still missing important failure modes.

In this thesis, we build on this testing mindset and treat fairness checks as tests whose sensitivity can be assessed. The key aim is not only to measure fairness, but also to assess whether the fairness evaluation is strong enough to detect fairness-relevant shifts under realistic change.

2.10 Benchmark Datasets Used in This Thesis

Our empirical study uses two widely used fairness benchmarks:

- **Adult (Census Income).** Predicts whether income exceeds $50K using demographic and employment attributes. It is commonly used to study group disparities by sex, race, and age-related slices.
- **COMPAS.** Predicts two-year recidivism risk using criminal history and demographic features. Prior analyses have raised concerns about disparities across racial groups and about the relationship between labels, policing, and recorded outcomes (2; 4).

These datasets are not perfect representations of the real world, but they are useful for controlled study because they are widely analysed and allow direct comparison to prior fairness research.

2.11 Summary and Link to the Method Chapter

This chapter established the core background needed for a fairness adequacy study:

1. fairness concerns arise even in accurate models due to historical bias, proxy labels, and operational choices (3);

2. group-level fairness metrics (SPD, EOD, EOM) provide practical lenses, but they involve trade-offs and can be unstable on small subgroups (9; 4; 11);

3. fairness can drift under realistic pipeline and population changes, motivating stress-testing rather than static audits (17);

4. mutation testing provides an adequacy mindset: evaluate whether tests detect controlled faults, rather than assuming one evaluation snapshot is sufficient (1; 12).

Chapter 3 presents the Fairness Adequacy Testing (FAT) framework, including the fairness-aware mutation operators, guardrails for mutant validity, and the mutant classification procedure used in the experiments.

Chapter 3

Method: Fairness-Oriented Mutation Testing

3.1 Overview

This chapter presents a fairness-oriented mutation testing method designed to assess whether a test suite is *adequate* to detect fairness problems in supervised classification models. Traditional mutation testing focuses on correctness under code or data perturbations; here, the central objects are *fairness metrics*.

The main idea is simple. We introduce small but realistic changes (*mutants*) to the training data or pipeline, retrain the same model specification, and then measure how much group fairness metrics move on a fixed test set. If the change in fairness exceeds a data-driven threshold, the mutant is **KILLED**; otherwise, it is **ALIVE**. Aggregating kill decisions yields a *Fairness Mutation Kill Rate* (mutation kill rates (MKR)), which we interpret as a fairness-oriented test adequacy score. Uncertainty in MKR is reported via bootstrap confidence intervals.

The method is intended for high-stakes classification problems in dynamic domains, where input distributions, thresholds, and feature definitions may shift over time. Although the empirical study focuses on the Adult and COMPAS datasets, the procedure is model-agnostic and can be applied to other tasks.

3.2 Data, Models, and Protected Attributes

Let $\mathcal{D} = \{(x_i, y_i, a_i)\}_{i=1}^{n}$ denote labelled instances with features x_i, binary labels $y_i \in \{0, 1\}$, and a protected attribute a_i (binary or multi-class; for example, gender, race, or client segment). The dataset is split into training and test sets. A baseline model M_{base} is trained on the training split and evaluated on a held-out test set $\mathcal{S} \subset \mathcal{D}$ that remains fixed for all mutants.

Predictions on the test set are either hard labels $\hat{y}_i = M(x_i) \in \{0, 1\}$ or calibrated scores $\hat{p}_i \in [0, 1]$ thresholded at some decision level τ. Group fairness metrics are always computed on $\mathcal{S}$ and stratified by the protected attribute A.

Throughout this thesis we evaluate the three base learners introduced in Chapter 2—Logistic Regression LR, Decision Trees CART, and k-Nearest Neighbours K-NN—and monitor the fairness metrics SPD, EOM (Equal Opportunity), and EOD (Equalized Odds). The same model templates, hyperparameter grids, and test sets are re-used for all mutants so that differences can be attributed to the mutations themselves.

For a given configuration, let M denote the number of *non-equivalent* mutants, i.e., mutants that pass the validity checks described in Section 3.8. The main adequacy statistic is the Fairness Mutation Kill Rate MKR,

$$\text{MKR} = \frac{\#\,\text{KILLED}}{M}.$$

Higher MKR values indicate that the test suite is more sensitive to fairness-relevant changes and is therefore more adequate from a fairness perspective.

3.3 Methodology Workflow

The fairness mutation procedure follows an iterative workflow that combines group fairness metrics with mutation testing. The steps are summarised in Figure 3.1.

1. **Select dataset and protected attributes.** Choose a classification dataset that contains at least one relevant protected attribute (for example, gender, race, age, or socioeconomic status). This dataset provides the basis for both model training and fairness evaluation.

2. **Generate candidate mutants.** Apply mutation operators to the training data or pipeline to create candidate mutants. These operators include instance and feature removal, reweighting, feature addition, and pipeline variations. The goal is to simulate realistic changes that could affect group fairness. The full operator catalogue is given in Section 3.7.

3. **Train models on original and mutated data.** Train the baseline model M_{base} on the original training data and, for each valid mutant, retrain the same model specification on the mutated training data. Each mutant model M_m is evaluated on the fixed test set $\mathcal{S}$.

4. **Compute fairness metrics.** For the baseline and each mutant model, compute group fairness metrics such as Statistical Parity Difference, Equal Opportunity (TPR disparity), and Equalized Odds (TPR/FPR disparity). These metrics provide a quantitative measure of how outcomes differ across protected groups (Section 3.5).

5. **Decide whether the mutant is KILLED or ALIVE.** For each mutant, compare changes in fairness metrics against bootstrap-based thresholds derived from the baseline model (Section 3.4.2). If any monitored fairness metric

moves outside its control limit, the mutant is labelled **KILLED**; otherwise it is **ALIVE**. This implements the kill rule described in Section 3.4.1.

6. **Update adequacy estimate.** After discarding unrealistic or fairness-neutral mutants via the validity gate (Section 3.8), aggregate kill decisions into the MKR statistic. When MKR is low, the test suite is missing fairness-relevant shifts; when MKR is high, the test suite is more adequate for fairness.

7. **Iterate if needed.** If MKR remains unsatisfactory, extend the test suite (for example, by adding more slices, metrics, or targeted mutants) and repeat the process until a chosen adequacy level is reached.

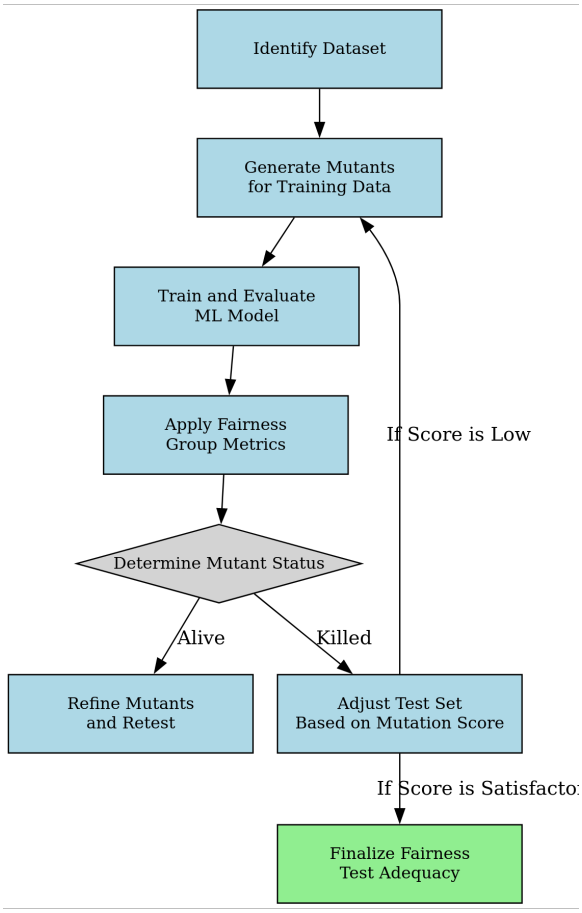

Figure 3.1: Iterative fairness test adequacy process for machine learning models.

3.4 Fairness-Oriented Mutation Testing Framework

This section formalises the proposed fairness-oriented mutation testing framework. The core elements are:

- a family of **mutation operators** that induce plausible changes in training data or pipeline configuration;
- a **fairness oracle** that evaluates group fairness metrics on a fixed test set;
- a **kill rule** that classifies mutants as KILLED or ALIVE based on bootstrap thresholds for fairness deltas; and
- an **adequacy score** (MKR) that aggregates kill decisions across non-equivalent mutants.

The framework is model-agnostic: it applies in the same way to K-NN, Decision Trees, and Logistic Regression, provided that the same model specification is retrained on each mutated dataset.

3.4.1 Mutation Lifecycle and Kill / Alive Decision

Figure 3.2 depicts the lifecycle of a fairness mutant. The procedure begins with a base model trained on clean data, proceeds through mutant creation and retraining, and ends with a fairness-based decision on whether the mutant is killed.

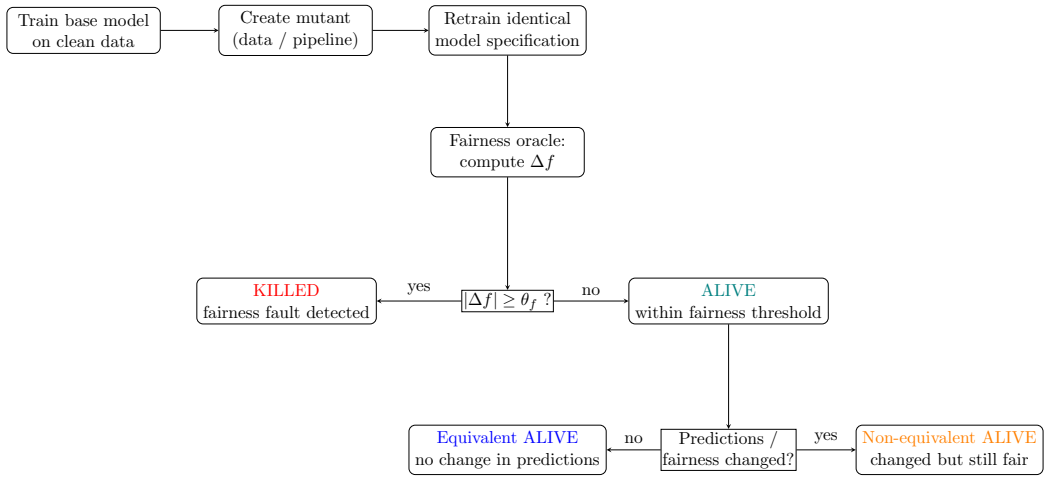

Figure 3.2: Fairness mutation lifecycle with post-hoc classification of **ALIVE** mutants into equivalent (no prediction change) and non-equivalent (prediction/fairness changes remain within the accepted threshold).

Concretely, for each dataset and base learner, the lifecycle is:

1. Train a **base model** on the original training set.
2. Compute baseline fairness metrics on the fixed test set $\mathcal{S}$.
3. Estimate per-metric control limits θ_f using bootstrap (Section 3.4.2).
4. Generate candidate mutants using the operators in Section 3.7.
5. Filter mutants through the **validity gate** (Section 3.8); discard unrealistic or fairness-neutral mutants.
6. Retrain the same model specification on each valid mutant and compute the fairness deltas Δf_m on $\mathcal{S}$.
7. Label each mutant as **KILLED** or **ALIVE** according to whether any fairness delta exceeds its control limit, and aggregate these decisions into MKR (Section 3.9).

3.4.2 Bootstrap Threshold Estimation

The kill decision should not react to minor fluctuations due to sampling noise. For each fairness metric f, we therefore estimate a *kill threshold* θ_f using a bootstrap-based null distribution under the base model.

Let f_{base} denote the value of metric f on the base model and test set $\mathcal{S}$.

1. Keep the base model M_{base} fixed.
2. Draw B bootstrap resamples of the test set, $\{\mathcal{S}^{(b)}\}_{b=1}^{B}$, by sampling with replacement.
3. For each resample, compute $f_{\text{base}}^{(b)}$ and define $\Delta f^{(b)} = f_{\text{base}}^{(b)} - f_{\text{base}}$.

4. Let μ_{null} and σ_{null} be the mean and standard deviation of the bootstrap deltas $\{\Delta f^{(b)}\}_{b=1}^{B}$.

The control limit for metric f is then

$$\boxed{\theta_f = \mu_{\text{null}} + 1.5\,\sigma_{\text{null}}.}$$

The multiplier 1.5 is chosen as a pragmatic balance between sensitivity (detecting meaningful fairness changes) and specificity (ignoring small fluctuations). In the experiments we use $B = 1000$ bootstrap rounds by default.

A mutant m is labelled **KILLED** if

$$|\Delta f_m| \geq \theta_f$$

for *any* monitored fairness metric or subgroup slice; otherwise it is **ALIVE**. Here $\Delta f_m = f_m - f_{\text{base}}$ is the change in fairness metric f between the mutant and the base model.

3.5 Fairness Metrics and Headline Score

3.5.1 Group Fairness Metrics

We track three standard group fairness criteria for a binary decision $\hat{Y}$, protected attribute A, and groups a and b. These metrics were introduced conceptually in Chapter 2; they are restated here in the notation used by the mutation framework.

1. **Statistical Parity Difference SPD.** This metric measures differences in positive outcome rates across groups:

$$\text{SPD} = P(\hat{Y} = 1 \mid A = a) - P(\hat{Y} = 1 \mid A = b).$$

The ideal value is 0, meaning equal positive prediction rates.

2. **Equal Opportunity Metric EOM.** Equal opportunity requires that qualified individuals receive the positive outcome at the same rate, regardless of group membership:

$$\mathrm{EOM} = \mathrm{TPR}_{\mathrm{privileged}} - \mathrm{TPR}_{\mathrm{unprivileged}},$$

where $\mathrm{TPR} = P(\hat{Y} = 1 \mid Y = 1, A)$. The ideal value is again 0.

3. **Equalized Odds Metric EOD.** Equalized odds requires equality of both true positive rates and false positive rates:

$$\mathrm{TPR}_{\mathrm{privileged}} = \mathrm{TPR}_{\mathrm{unprivileged}}, \quad \mathrm{FPR}_{\mathrm{privileged}} = \mathrm{FPR}_{\mathrm{unprivileged}}.$$

We operationalise this via

$$\mathrm{EOD} = \max\bigl\{|\mathrm{TPR}_a - \mathrm{TPR}_b|, |\mathrm{FPR}_a - \mathrm{FPR}_b|\bigr\},$$

with ideal value 0.

As an illustrative example, suppose the blue group has $\mathrm{TPR} = 0.80$ and the red group has $\mathrm{TPR} = 0.62$. Then

$$\mathrm{EOM} = 0.80 - 0.62 = 0.18.$$

If the bootstrap threshold is $\theta_{\mathrm{EOM}} = 0.10$, then any mutant that *further widens* the TPR gap beyond 0.10 is treated as a fairness fault and is therefore **KILLED** by the EOM oracle.

3.5.2 Headline Fairness Score

To summarise multiple fairness criteria in a single index, we use a Weighted Multi-Objective Fairness Index (WMFI) in $[0, 1]$, where higher values indicate better fairness.

For each sensitive attribute we compute:

- Demographic Parity gap $\text{DPgap} = \max_g PR_g - \min_g PR_g$,
- Equal Opportunity gap $\text{EOgap} = \max_g TPR_g - \min_g TPR_g$,
- Equalized Odds gap $\text{EODgap} = \max\big(\text{EOgap}, \max_g FPR_g - \min_g FPR_g\big)$,
- Disparate Impact $\text{DI} = \min_g PR_g / \max_g PR_g$,

where PR_g denotes the positive prediction rate in group g.

The WMFI is then defined as

$$\text{WMFI} = 1 - \big(w_{DP}\,\text{DPgap} + w_{EO}\,\text{EOgap} + w_{EOD}\,\text{EODgap}\big) - \lambda \cdot \mathbf{1}[\text{DI} < \tau],$$

with weights $(w_{DP}, w_{EO}, w_{EOD}) = [0.30, 0.40, 0.30]$, disparate impact threshold $\tau = 0.80$, and penalty $\lambda = 0.05$. WMFI is used as a headline score in the empirical results; mutation testing, however, is always performed on the component metrics themselves.

3.6 Mutation Design

Mutations can be applied at several levels of the learning process. We consider three broad categories:

Data-level: controlled perturbations to inputs, such as feature noise, time-window shifts, synthetic outliers, group reweighting, or removal of specific instances.

Pipeline-level: changes to preprocessing and sampling procedures, such as swapping normalisation schemes, altering imputation strategies, or modifying resampling settings.

Model-level: limited modifications to model training, such as adjusting regularisation strength, decision thresholds, or class weights.

Each mutation operator ϕ transforms the training data and/or training procedure to produce a mutated model M_ϕ. Operators are parameterised to keep mutations plausible and to avoid unrealistic scenarios that would not arise in normal operation.

3.7 Mutation Operators

This section describes the concrete mutation operators used in the Fairness Adequacy Test (FAT) framework. Operators are grouped into four main families: removal, addition, permutation, and shuffling. Together they instantiate the data- and pipeline-level mutations described above.

3.7.1 Removal Operators

Removal operators delete parts of the dataset to stress-test reliance on specific instances, groups, or features.

Instance Removal

Instance removal eliminates selected records, especially from minority or protected groups, to mimic data loss or under-representation.

- **Binary removal.** For a binary sensitive attribute (for example, gender), create mutants by dropping all instances from one group at a time:

  - Mutant 1: remove all cases with gender = female.
  - Mutant 2: remove all cases with gender = male.

- **Multi-value removal.** For a multi-valued attribute (for example, race), create one mutant per value:
  - Mutant 1: remove all cases with race = Black.
  - Mutant 2: remove all cases with race = White.
  - Mutant 3: remove all cases with race = Hispanic.
  - Mutant 4: remove all cases with race = Asian.

- **Combinatorial removal.** For multiple sensitive attributes, remove intersectional combinations:
  - Mutant 1: remove all cases with gender = male and race = Black.
  - Mutant 2: remove all cases with gender = male and race = White.
  - Mutant 3: remove all cases with gender = female and race = Black.
  - Mutant 4: remove all cases with gender = female and race = White.

Feature Removal

Feature removal studies how the model behaves when certain pieces of information are absent.

- **Single feature removal.** Remove one feature entirely, such as the "sex" attribute.

- **Multiple feature removal.** Remove features one at a time to create independent mutants, for example: gender, race, income, or education level.

- **Combinatorial feature removal.** Remove feature pairs such as:
  - gender and race;
  - gender and income;
  - race and education level;
  - income and education level.

3.7.2 Addition Operators

Addition operators create new features synthesised from existing ones.

- **Linear combinations.** Create a new feature as a linear combination, e.g.

$$\text{new\_feature} = a \cdot \text{age} + b \cdot \text{years of education},$$

  with constants a and b.

- **Ratios.** Create ratio-based features such as

$$\text{new\_feature} = \frac{\text{age}}{\text{years of education}}.$$

Instance Addition

For intersectional fairness coverage, new synthetic instances are added by generating combinations of sensitive attribute values (for example, all gender–race combinations) in areas where the original data are sparse.

3.7.3 Permutation Operators

Permutation operators reassign values within the dataset to disrupt patterns on which the model may be relying.

Feature Permutation

Randomly permute the values of a single feature across instances. This breaks the link between that feature and the label, while preserving its marginal distribution.

Row Permutation

Shuffle the order of rows in the training data to remove potential artefacts from batch structure or temporal ordering.

3.7.4 Shuffling Operators

Shuffling operators further test the model's dependence on particular associations.

Feature Shuffling

Randomly reassign values within a feature column, similar to feature permutation, to stress-test feature importance and proxy effects.

Label Shuffling

Shuffle labels among samples, disrupting the association between features and labels. This checks whether the model is learning genuine patterns or simply memorising the training set; such mutants are usually filtered by the validity gate if they become too unrealistic.

3.7.5 Link to Data-, Pipeline-, and Model-Level Mutations

The concrete operators above instantiate the broader mutation categories:

- **Data-level mutations:** instance and feature removal or addition, permutation, shuffling, label flips, feature noise, intersectional resampling, and missing-

ness injection.

- **Pipeline-level mutations:** threshold slides, encoder swaps, scaler swaps, and class-weight changes.
- **Model-level mutations:** small changes to regularisation or removal of fairness-critical features.

These operators are always combined with the validity gate in the next section to maintain realism.

3.8 Validity Gate (Pre-Filter)

Not every syntactically valid mutant is meaningful for fairness analysis. Before a mutant contributes to MKR, it must pass two checks.

1. **Distribution similarity.** We compare the original and mutated training distributions using two-sample tests:

   - Kolmogorov–Smirnov tests for numeric features, and
   - χ^2 tests for categorical features.

   Mutants must satisfy $p \geq 0.05$ (or remain below a chosen effect-size threshold) for all monitored features to be considered within a realistic shift regime.

2. **Training-side fairness shift.** We require that the mutation moves the fairness signal on the training data. Let Δf^{train} be the change in a fairness metric on the training set, and let $(\mu_{\text{null}}^{\text{train}}, \sigma_{\text{null}}^{\text{train}})$ denote its bootstrap null statistics. We require

$$|\Delta f^{\text{train}}| \geq \mu_{\text{null}}^{\text{train}} + \sigma_{\text{null}}^{\text{train}}.$$

Mutants failing either check are treated as *equivalent* in practice and are excluded from the MKR denominator.

3.9 Mutation Score and Uncertainty

Let M denote the number of mutants that pass the validity gate. For each such mutant m, let $\mathbf{1}\{m \text{ is KILLED}\}$ be an indicator that the fairness oracle has detected a fault. The Fairness Mutation Kill Rate is

$$\boxed{\text{MKR} = \frac{1}{M} \sum_{m=1}^{M} \mathbf{1}\{|\Delta f_m| \geq \theta_f \text{ for some } f\},}$$

where the condition "for some f" ranges over all monitored metrics and slices. Mutants rejected by the validity gate do not enter M.

To quantify uncertainty, we estimate a 95% percentile bootstrap confidence interval for MKR by resampling the set of mutants with replacement. We also compute MKR broken down by slice (for example, per sensitive attribute or per operator type) to see which groups or mutation families contribute most to the detections. An adjusted score MKR_{adj} can further exclude mutants with very small training-side fairness changes, producing a more conservative adequacy judgement.

3.10 Worked Mini-Example (Trading Classifier)

To make the procedure more concrete, consider a binary trading signal model that predicts $\hat{y} \in \{0, 1\}$ ("execute trade" or "do not execute"). The protected attribute is client segment $a \in \{\text{retail}, \text{institutional}\}$. We track Equal Opportunity (EOM) as the main fairness metric.

1. **Bootstrap.** Train the base model M_{base} on a rolling window and compute

EOM on $\mathcal{S}$. Using $B = 1000$ bootstrap resamples of $\mathcal{S}$ under the fixed base model, obtain $(\mu_{\text{EOM}}, \sigma_{\text{EOM}})$ and set $\theta_{\text{EOM}} = \mu_{\text{EOM}} + 1.5\sigma_{\text{EOM}}$.

2. **Mutations.** Generate mutants such as:

   - shifting the input time window by +1 day;
   - adding small Gaussian noise to volatility features;
   - reweighting the minority segment by a factor of 1.2;
   - swapping z-score scaling with robust scaling; and
   - changing missing-value imputation from median to k-NN.

3. **Validity gate.** Reject mutants that produce a large AUC drop (for example, > 0.03) or extreme feature mean shifts (for example, $> 2\sigma$), or that fail the fairness-shift check on the training data.

4. **Assessment.** For each valid mutant, compute ΔEOM on $\mathcal{S}$ and apply the kill rule. Aggregate decisions into MKR with a 95% confidence interval.

This miniature scenario mirrors the structure of the main experiments and shows how often plausible operational changes could materially worsen fairness across client segments.

3.11 Implementation Pseudocode

Algorithm 1 gives high-level pseudocode for the fairness mutation testing pipeline. The same procedure is applied to all datasets and base learners, with shared fairness metrics and thresholds.

Algorithm 1: Fairness Mutation Testing Pipeline

Input : Training data $\mathcal{D}_{\text{train}}$; test set $\mathcal{S}$; model template $\mathcal{M}$; mutant generator $\mathcal{G}$; fairness metrics $\mathcal{F}$; threshold multiplier $z = 1.5$; bootstrap rounds $B = 1000$

Output: Mutation Kill Rate (MKR) and 95% confidence interval

`/* Step 1: Bootstrap null distribution for fairness metrics */`

Train base model $M_{\text{base}} \leftarrow \mathcal{M}(\mathcal{D}_{\text{train}})$;

Compute baseline fairness metrics $f_{\text{base}} \leftarrow \text{FairnessOracle}(M_{\text{base}}, \mathcal{S}, \mathcal{F})$;

for $b = 1$ **to** B **do**

  $\mathcal{S}^{(b)} \leftarrow$ bootstrap sample of $\mathcal{S}$;

  $f_{\text{base}}^{(b)} \leftarrow \text{FairnessOracle}(M_{\text{base}}, \mathcal{S}^{(b)}, \mathcal{F})$;

  $\Delta f^{(b)} \leftarrow f_{\text{base}}^{(b)} - f_{\text{base}}$;

end

Compute $\mu_{\text{null}}, \sigma_{\text{null}}$ over $\{\Delta f^{(b)}\}$ for each $f \in \mathcal{F}$;

Set threshold $\theta_f \leftarrow \mu_{\text{null}} + z \cdot \sigma_{\text{null}}$ for each $f \in \mathcal{F}$;

$M \leftarrow 0$; kills $\leftarrow 0$;

/* Step 2: Generate, filter, and score mutants */

for *mutant dataset* $\mathcal{D}_{train}^{(m)}$ *in* $\mathcal{G}(\mathcal{D}_{train})$ **do**

  if *validity gate fails for* $\mathcal{D}_{train}^{(m)}$ **then**

    ; // Discard equivalent or unrealistic mutant

  end

  $M \leftarrow M + 1$;

  $M_m \leftarrow \mathcal{M}(\mathcal{D}_{\text{train}}^{(m)})$;

  $f_m \leftarrow \text{FairnessOracle}(M_m, \mathcal{S}, \mathcal{F})$;

  $\Delta f_m \leftarrow f_m - f_{\text{base}}$;

  if $\exists f \in \mathcal{F}$ *such that* $|\Delta f_m(f)| \geq \theta_f$ **then**

    kills $\leftarrow$ kills +1;

  end

end

Compute MKR $\leftarrow$ kills/M;

Estimate 95% bootstrap confidence interval for MKR by resampling $\{1, \ldots, M\}$ with replacement;

return MKR and its confidence interval;

Algorithm 1 turns the conceptual framework into an operational procedure: it bootstraps fairness metrics to define thresholds, generates and filters mutants, and summarises their impact via MKR.

Research Questions

This chapter operationalises the research questions as follows:

RQ1. **(RQ1 – Effectiveness of mutation operators)** How effective are fairness-aware mutation operators? The chapter defines a family of operators, a validity

gate, and a kill/alive decision rule, allowing their effectiveness to be quantified via mutation kill rates and metric drift (Sections 3.3–3.7).

RQ2. **(RQ2 – Performance across models and datasets)** How consistently does the framework perform across models and datasets? The same mutation pipeline is applied to different base learners (K-NN, Decision Tree, Logistic Regression) and datasets (Adult and COMPAS) using shared fairness metrics and bootstrap thresholds (Sections 3.5 and 3.9).

RQ3. **(RQ3 – Comparison to existing fairness testing baselines)** How does the proposed framework compare to existing fairness testing baselines? The method is positioned against static group fairness audits and headline fairness scores, providing a mutation-based adequacy perspective that can be compared to toolkits such as Fairlearn, AIF360, and FairTest (Sections 3.4–3.9 and Chapter 4).

In summary, Chapter 3 has set out the methodological foundation of the Fairness Adequacy Testing framework. Having defined the mutation lifecycle, fairness metrics, validity gate, and adequacy thresholds, the next chapter instantiates these elements on the Adult and COMPAS datasets, applies them across the base learners introduced in Chapter 2, and reports the resulting MKR and WMFI values to address the research questions in practice.

Chapter 4

Fairness Metrics and Empirical Evaluation

4.1 Purpose of this Chapter

This chapter has two main jobs. First, it explains the fairness metrics used in this thesis in a clear and practical way. Second, it shows how these metrics behave in our experiments when we apply the Fairness Mutation Testing (FAT) framework to real datasets.

Building on Chapter 3, we now run the framework on two benchmark datasets (Adult and COMPAS) and three base learners (Logistic Regression LR, Decision Tree CART, and k-NN K-NN). We:

- define and motivate the group-fairness metrics (SPD, EOD, EOM),
- introduce a single headline fairness score, the Weighted Multi-Objective Fairness Index (Weighted Multi-Objective Fairness Index (WMFI)),
- walk through a concrete numeric example on the Adult dataset,
- describe the experimental setup for Adult and COMPAS,
- report baseline accuracy and fairness,
- study how fairness metrics react to fairness drift thresholds,

- examine how metrics move under mutation-based stress testing,
- evaluate how effective the fairness-aware mutation operators are across models and datasets, and
- compare mutation-based adequacy with a diameter-based diversity baseline.

Where detailed tables or extended plots were too long for the main text, we provide them in a public repository so that others can inspect and reuse them.

4.2 Metric Definitions and Notation

We focus on a standard binary classification setting:

- The protected attribute $A \in \{a, b\}$ represents group membership (for example, Female vs. Male, or Black vs. White).
- The model output $\hat{Y} \in \{0, 1\}$ is a binary decision (e.g. accept vs. reject, low risk vs. high risk).
- The ground-truth label $Y \in \{0, 1\}$ is the true outcome.

All fairness rates are computed on the same fixed test set $\mathcal{S}$, in line with the mutation-testing protocol described in Chapter 3.

Equal Opportunity vs. Equalized Odds

Because two related ideas appear throughout this chapter, we clarify their difference up front:

- **Equal Opportunity** focuses on qualified individuals. It asks: "Among people who truly qualify (e.g. $Y = 1$), do different groups get the positive decision at similar rates?" It compares *true positive rates*.

- **Equalized Odds** is stricter. It looks at both *true positive rates* and *false positive rates*. It asks: "Are both qualified people helped equally, and unqualified people treated similarly, across groups?"

In this thesis we operationalise these ideas as:

- EOD: an Equal Opportunity-style metric that captures disparity in true positive rates (True Positive Rate (TPR)),
- EOM: an Equalized Odds-style metric that captures the largest disparity across both TPR and False Positive Rate (FPR).

This distinction matters when we interpret the graphs. In both Adult and COMPAS, EOM is a stricter test than EOD, because EOM penalises unfairness in both "who we help" (TPR gaps) and "who we mistakenly harm" (FPR gaps).

4.2.1 Composite Fairness Index (Scalar Measure)

In practice, it is often useful to have a single number that summarises several fairness checks. For this, we define the Weighted Multi-Objective Fairness Index (WMFI). WMFI takes values between 0 and 1, where 1 means "best" (from a fairness point of view).

Key concepts

- **Demographic Parity (Demographic Parity Difference (DP)):** A model satisfies demographic parity if the probability of receiving a positive prediction is the same across protected groups.

- **Disparate Impact (DI):** The ratio of positive prediction rates between groups. Under the common "four-fifths rule", ratios below 0.80 are treated as red flags for potential discrimination.

Gap measures

For each sensitive attribute we compute:

- **DPgap:**

$$\text{DPgap} = \max_g PR_g - \min_g PR_g,$$

  where PR_g is the positive prediction rate for group g.

- **EOgap:** the gap between the highest and lowest true positive rates across groups (a pure TPR-based gap).

- **EOMgap:** the Equalized-Odds-style gap, defined as the maximum between EOgap and the corresponding false positive rate gap.

- **Disparate Impact:**

$$\text{DI} = \frac{\min_g PR_g}{\max_g PR_g}.$$

Here, DI tells us how the least-advantaged group compares to the most-advantaged group in terms of positive predictions.

Illustrative example

Suppose a hiring model yields:

Group	**Positive Prediction Rate (PR_g)**
Group A (Female)	0.60
Group B (Male)	0.45

Then:

- $\max_g PR_g = 0.60$, $\min_g PR_g = 0.45$,
- DPgap $= 0.15$,
- DI $= 0.45/0.60 = 0.75 < 0.80$, so it fails the four-fifths rule.

WMFI Formula and Interpretation

We combine the gap measures into a scalar score:

$$\text{WMFI} = 1 - \big(w_{DP}\,\text{DPgap} + w_{EO}\,\text{EOgap} + w_{\text{EOM}}\,\text{EOMgap}\big) - \lambda \cdot \mathbf{1}[\text{DI} < \tau],$$

with

$$(w_{DP}, w_{EO}, w_{\text{EOM}}) = [0.30,\, 0.40,\, 0.30], \quad \tau = 0.80, \quad \lambda = 0.05.$$

The penalty term $-\lambda \cdot \mathbf{1}[\text{DI} < \tau]$ is triggered when DI falls below the legal-style tolerance threshold $\tau = 0.80$. This reflects the fact that a bad DI ratio is often treated more seriously in policy and compliance settings than a small change in a gap.

In short: WMFI gives a single fairness score that can be compared across models and datasets, while still being grounded in clear components (DPgap, EOgap, EOMgap, and DI).

4.3 Worked Numeric Example on Adult Dataset

To make the metrics more concrete, we walk through a simple example on the Adult test set. We use *sex* as the protected attribute (Female $= a$, Male $= b$), and consider a logistic regression model with a fixed threshold of 0.5.

Positive prediction rates:

Table 4.1: Confusion matrices by sex – Adult test set (example).

	Female (a)		Male (b)	
	$\hat{Y}=1$	$\hat{Y}=0$	$\hat{Y}=1$	$\hat{Y}=0$
$Y=1$	803	547	2582	1094
$Y=0$	423	2227	1015	5289

- Positive rates:

$$P(\hat{Y}=1 \mid A=a) = \frac{803+423}{803+547+423+2227} = 0.322,$$

$$P(\hat{Y}=1 \mid A=b) = \frac{2582+1015}{2582+1094+1015+5289} = 0.311,$$

so

$$\text{SPD} = 0.322 - 0.311 = 0.011.$$

The model has almost no difference in overall selection rate by sex.

- True-positive rates:

$$\text{TPR}_a = \frac{803}{803+547} = 0.595, \quad \text{TPR}_b = \frac{2582}{2582+1094} = 0.702,$$

So the Equal Opportunity Difference is

$$\text{EOD} = 0.595 - 0.702 = -0.107.$$

This means qualified males receive positive decisions about 10.7 percentage points more often than qualified females.

- False-positive rates:

$$\mathrm{FPR}_a = \frac{423}{423 + 2227} = 0.160, \quad \mathrm{FPR}_b = \frac{1015}{1015 + 5289} = 0.161.$$

  False positive rates are very similar across groups.

- Equalized-Odds-style metric: We define

$$\mathrm{EOM} = \max\{|-0.107|,\ |0.160 - 0.161|\} = 0.107.$$

Take-away. Even though the selection rates (SPD) are almost equal, the true positive rates are not. This is a pattern that we will see again in the main experiments: for both Adult and COMPAS, SPD can look acceptable while EOD and EOM reveal stronger fairness differences. This motivates a richer fairness test, including mutation-based stress tests.

4.4 Experimental Setup

4.4.1 Datasets and Protected Attributes

We evaluate the framework on two widely used benchmark datasets:

- **Adult:** Contains demographic and employment information used to predict whether an individual's income exceeds \$50K. Protected attributes in our analysis include sex, race, and age group.

- **COMPAS:** Contains criminal history and demographic information used to predict two-year recidivism risk. Protected attributes include race, `age_category`, and sex. We acknowledge well-known concerns about label quality and historical bias in this dataset but treat it as a standard benchmark for fairness studies.

For both datasets, we split data into training and test partitions using a fixed random seed to ensure reproducibility. Where we tune hyper-parameters, we use cross-validation on the training data only. All fairness metrics are computed on the held-out test set.

4.4.2 Pre-processing and Feature Engineering

Pre-processing is kept as simple and consistent as possible across datasets:

- Missing values are imputed using simple strategies (e.g. mode for categorical, mean/median for numerical).
- Categorical variables are encoded (e.g. one-hot encoding).
- Features are scaled or normalised when required by the model class (for example, for K-NN and Logistic Regression).
- In a few cases we construct additional features that also serve as candidates for mutation operators (see Chapter 3).

All transformations are fit on the training data and then applied to the test data, to avoid leakage.

4.4.3 Models and Hyper-parameters

We use three base learners that represent different modelling families:

- Logistic Regression (LR),
- Decision Tree (CART),
- k-Nearest Neighbours (K-NN).

Hyper-parameters are selected using grid search with cross-validation, following the procedure in Chapter 2. The final configurations are kept fixed when running the mutation experiments, so that differences come from mutations rather than re-tuning.

Code and Reproducibility

All code used for data preparation, model training, mutation operators, and figure generation is available in a public GitHub repository:

<YOUR-GITHUB-LINK>

The repository also includes:

- configuration files for the experiments,
- scripts for reproducing the mutation runs, and
- extended tables and graphs that do not appear in this chapter.

This allows readers to inspect the full analysis, run the code on their own machine, and adapt the framework to new datasets.

4.5 Baseline Accuracy and Fairness

Before we stress-test fairness through mutations, we first look at the baseline picture. For each dataset–model pair we compute standard predictive metrics (Accuracy, AUC, F1) and then evaluate fairness via EOD and EOM on the test set.

These baseline fairness values serve as a fixed reference point. Later, when we look at fairness drift caused by mutants, we will compare back to these baseline levels.

For Adult, the two plots together show that:

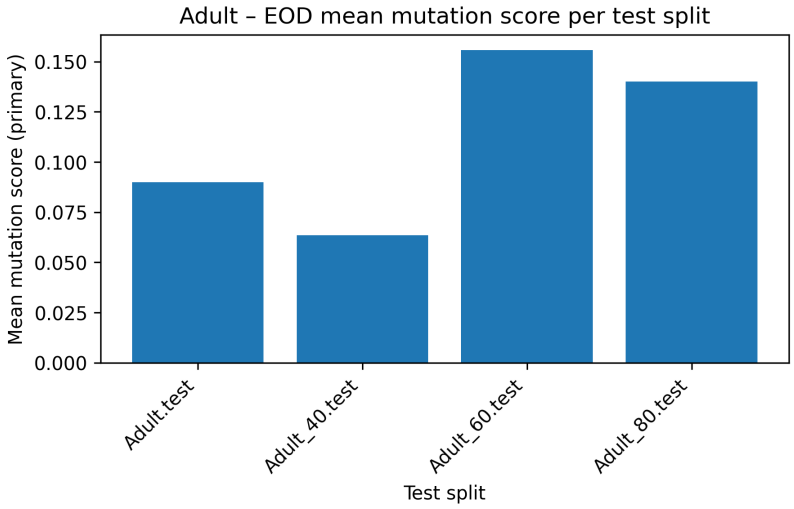

Figure 4.1: Adult dataset: baseline Equal Opportunity Difference (EOD) by protected attribute.
X axis: protected attributes / groups.
Y axis: EOD values, summarising how strongly Equal Opportunity is violated at baseline.

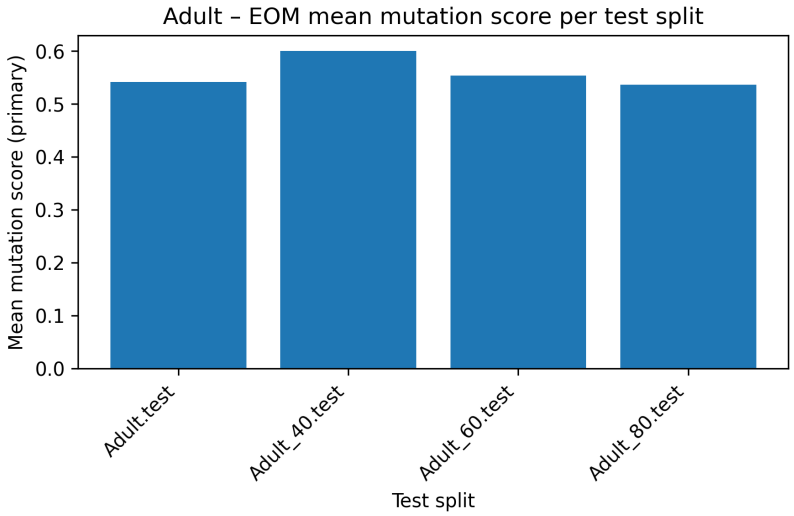

Figure 4.2: Adult dataset: baseline Equalized Odds Metric (EOM) by protected attribute.
X axis: protected attributes / groups.
Y axis: EOM values, summarising the maximum disparity across TPR and FPR at baseline.

- some protected attributes (for example, sex or race) have noticeably higher EOD and EOM values than others,
- EOM is often slightly larger than EOD, since it can reflect false positive differences as well as true positive differences.

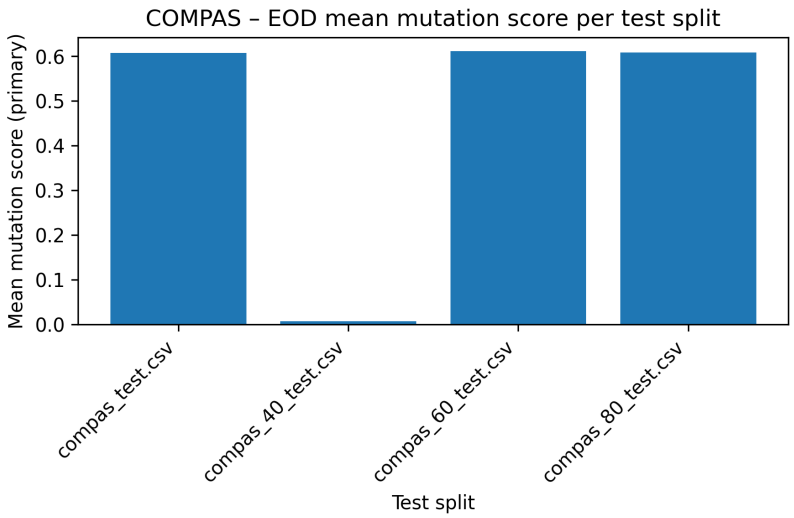

Figure 4.3: COMPAS dataset: baseline Equal Opportunity Difference (EOD) by protected attribute.
X axis: protected attributes / groups.
Y axis: EOD values, showing opportunity gaps at baseline.

On COMPAS, the gaps are often larger than on Adult, especially under EOM. This reflects two differences:

- the task is higher risk (recidivism), so mistakes can have more impact,
- structural biases in the data can lead to stronger violations of both opportunity and error-rate parity.

Comparing Adult and COMPAS side by side, and comparing EOD vs. EOM, we see three recurring patterns that will also show up in the mutation graphs:

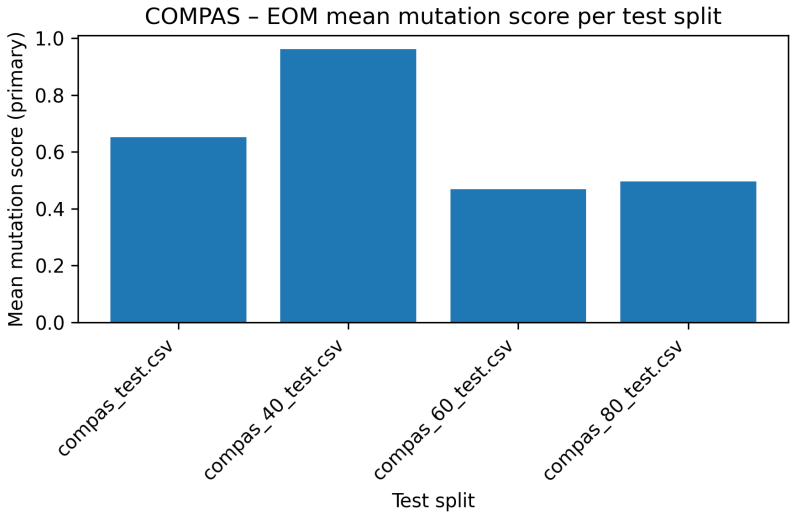

Figure 4.4: COMPAS dataset: baseline Equalized Odds Metric (EOM) by protected attribute.
X axis: protected attributes / groups.
Y axis: EOM values, highlighting where recidivism predictions depart most from Equalized Odds.

1. COMPAS tends to show *larger* fairness gaps than Adult.

2. EOM is usually *stricter* than EOD, because it reflects differences in both TPR and FPR.

3. For some protected attributes, the two metrics disagree: a group may look "moderately unfair" under EOD but much worse under EOM.

4.6 Threshold Sensitivity Analysis

In our experiments, the "threshold" we vary is not the classifier's decision threshold. Instead, it is a fairness *drift* threshold that decides when a mutant is considered **KILLED**.

To avoid tuning the threshold on the mutants themselves, we estimate a metric-specific *null drift* level using bootstrap re-sampling of the fixed test set $\mathcal{S}$ (with the

trained baseline model held constant), as described in Chapter 3. For each fairness metric f (e.g., EOD or EOM), we compute the absolute drift under bootstrap samples and define a detection threshold:

$$\theta_f = \mu_\Delta^{(0)} + 1.5 \cdot \sigma_\Delta^{(0)},$$

where $\mu_\Delta^{(0)}$ and $\sigma_\Delta^{(0)}$ are the mean and standard deviation of the bootstrap null drift distribution for metric f. A mutant is **KILLED** if its fairness change exceeds θ_f. Otherwise it is **ALIVE**.

Among ALIVE mutants we then distinguish:

- **ALIVE–Equivalent:** fairness drift below a practical negligibility level $\epsilon = 0.05$. In this work, these mutants are treated as effectively neutral for the fairness conclusions.
- **ALIVE–Non-Equivalent:** fairness drift above ϵ but still below the detection threshold θ_f. These are fairness-relevant shifts that the current fairness tests do not flag as faults.

This structure matters later when we interpret the graphs: KILLED mutants show where our tests are strong, while ALIVE–Non-Equivalent mutants show where they miss meaningful fairness changes.

4.7 Metric Profiles Across Models

For each dataset and each base learner (K-NN, CART, LR), we estimate EOD and EOM across multiple random seeds. We then report the mean and a 95% bootstrap confidence interval to show how stable each metric is.

In the graphs (not reproduced here), we group bars by model family and dataset, and show both metrics:

- Models with *higher* EOD or EOM on average show larger measured disparities under that metric.
- Narrow confidence intervals indicate more stable fairness behaviour across seeds; wide intervals indicate that fairness is more fragile to data or training noise.

Across the figures, two patterns emerge:

1. For both Adult and COMPAS, EOM tends to be a harsher test than EOD, because it incorporates false positives in addition to true positives.
2. The relative ordering of models can change by dataset and metric. For example, a model that looks "better" on Adult under EOD may not look better on COMPAS under EOM. This is why we later study adequacy by dataset–metric pair, rather than assuming one global ranking.

4.8 Metric Stability Under Mutation Stress

We now use the mutation operators from Chapter 3 to see how fairness metrics behave under realistic perturbations at the data and pipeline levels.

4.8.1 Metric Drift by Mutation Type

For each mutation, we compute:

$$|\Delta\text{EOD}|, \quad |\Delta\text{EOM}|.$$

We summarise the distributions using medians and interquartile ranges (IQRs), and report the proportion of mutants where the drift exceeds the detection threshold θ_f.

Intuitively:

- A large median drift means that metric is sensitive to that type of change.
- A high proportion of "above threshold" cases means that the tests frequently detect fairness issues.

4.8.2 Mutation Status: KILLED vs. ALIVE

Using the rule from Chapter 3, each mutant is assigned to one of:

- **KILLED** — fairness drift exceeds the detection threshold.
- **ALIVE–Equivalent** — the mutant changes nothing practically important in fairness (as operationalised by ϵ).
- **ALIVE–Non-Equivalent** — fairness changes appear but stay below the threshold.

The ALIVE–Non-Equivalent mutants are especially important. They show that fairness-relevant changes can slip through the current tests. The mutation scores defined below quantify how often this happens.

Figure 4.5 provides an overview of how well the fairness tests perform on COMPAS by compressing the KILLED/ALIVE breakdown into a single number per setting.

Let:

- M_{primary} be the number of fairness-related mutants,

Figure 4.5: FAT mutation scores on the COMPAS dataset for Equal Opportunity Difference (EOD) and Equalized Odds Metric (EOM) across the four test files (full, 40%, 60%, 80%) and three algorithms (Logistic Regression, Decision Tree, K-NN).

- K_k be the number of KILLED mutants,
- E_e be the number of equivalent (effectively neutral) mutants.

We define the Fairness Adequacy Testing score as:

$$\text{FAT} \;=\; \frac{K_k}{M_{\text{primary}} - E_e} \in [0, 1].$$

Here:

- FAT = 1 means every non-equivalent mutant was detected.
- FAT = 0 means none of the non-equivalent mutants were detected.

We compute FAT separately for EOD and EOM. In the graphs that follow, higher FAT values mean stronger fairness test adequacy.

Across COMPAS, FAT scores vary by metric and algorithm. Because EOM reflects both TPR and FPR gaps, it can react differently from EOD to the same mutation, and this difference is visible in the summary plots. These patterns are mirrored, with different levels, in the Adult results.

4.8.3 RQ1: How Effective Are the Mutation Operators?

RQ1 asks whether the fairness-aware mutation operators actually inject meaningful fairness faults, and whether these faults can be detected across datasets and metrics.

We look at this from four angles:

1. a global comparison by dataset and metric,
2. differences between models,
3. robustness under down-sampling (smaller test sets),
4. stability of adequacy patterns across test splits.

Global Effectiveness by Dataset and Metric

Figure 4.6 shows that:

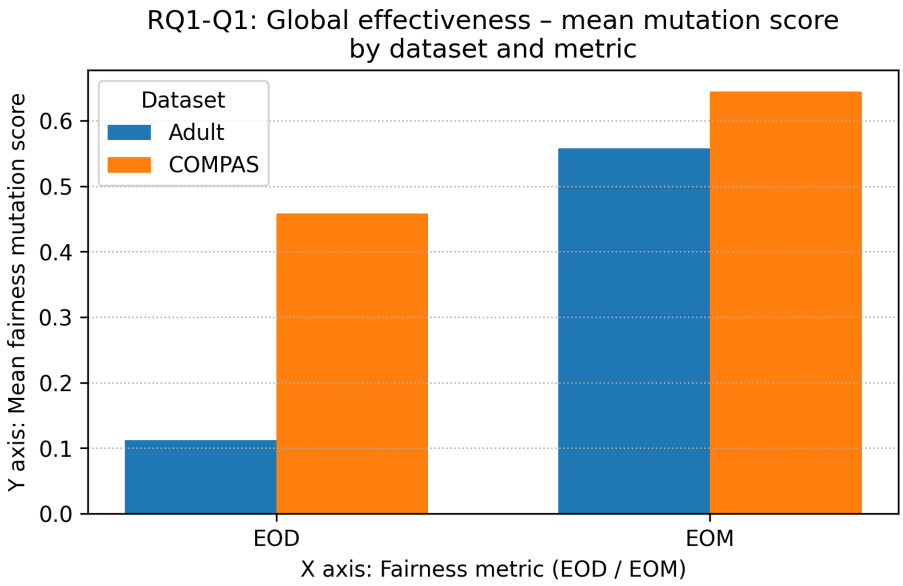

Figure 4.6: Global fairness mutation adequacy by dataset and metric.
X axis: fairness metric (EOD and EOM).
Y axis: mean FAT score (0–1), averaged over all models and test splits.

- For both datasets, FAT scores are clearly above zero, which means the operators do not produce trivial noise; they create fairness faults that the metrics can detect.

- Adequacy differs between EOD and EOM. Since EOM is stricter (it includes both TPR and FPR disparities), it can surface fairness faults that are not visible under a pure TPR-gap view.

Model-family Sensitivity to Fairness Faults

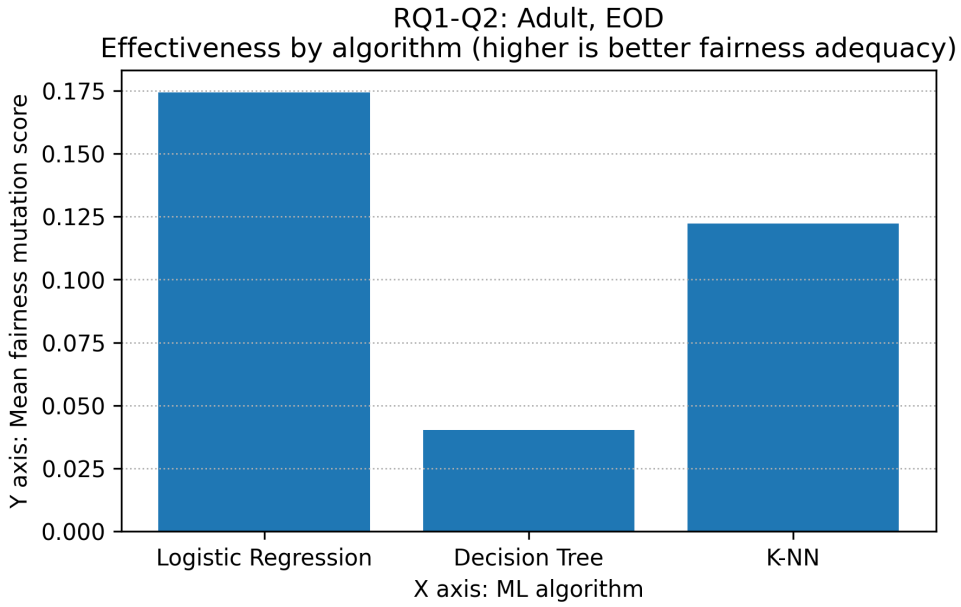

Figure 4.7: Adult dataset: algorithm-level FAT under EOD.
X axis: ML algorithm (LR, CART, K-NN).
Y axis: mean FAT score (0–1).

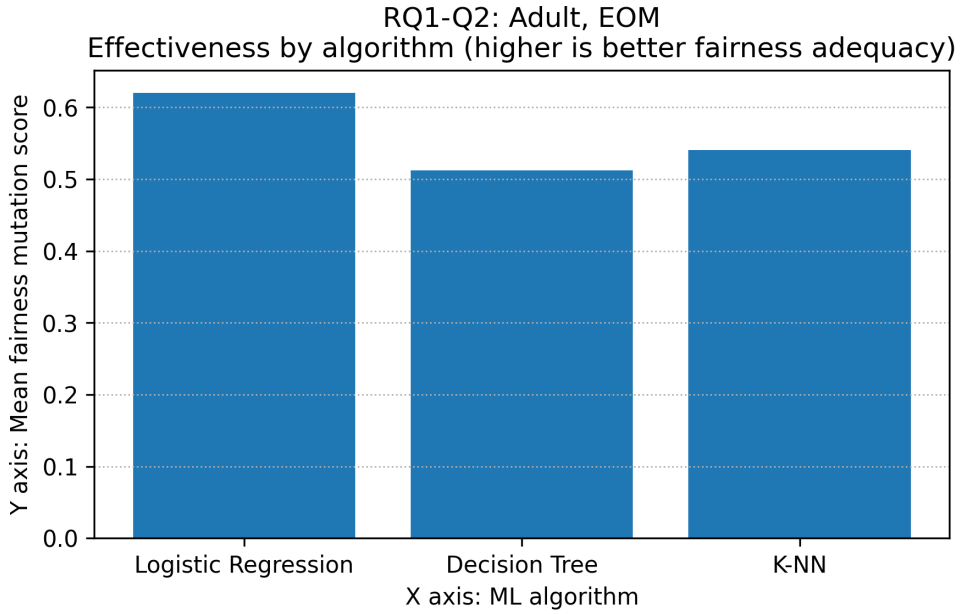

Figure 4.8: Adult dataset: algorithm-level FAT under EOM.
X axis: ML algorithm (LR, CART, K-NN).
Y axis: mean FAT score (0–1).

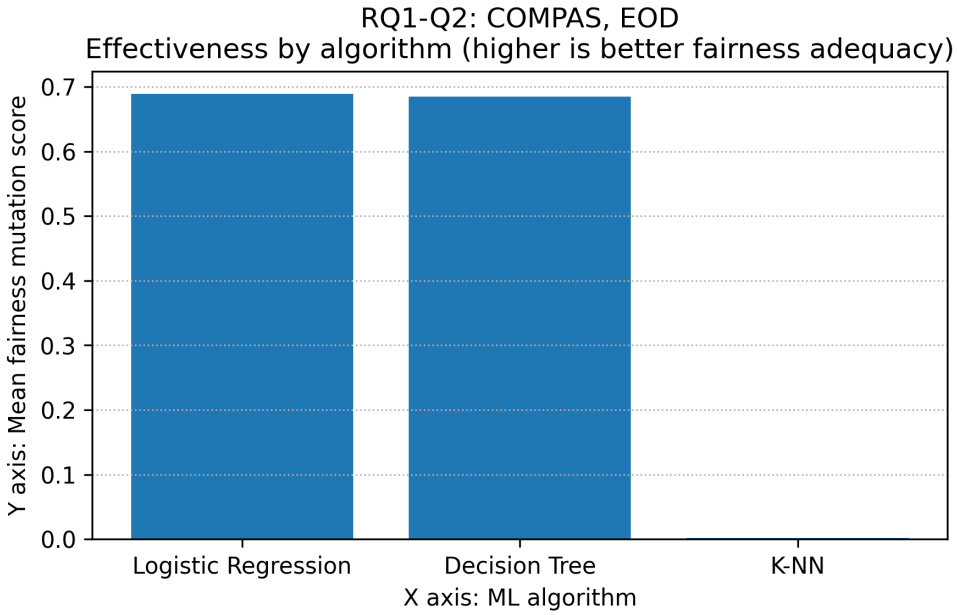

Figure 4.9: COMPAS dataset: algorithm-level FAT under EOD.
X axis: ML algorithm (LR, CART, K-NN).
Y axis: mean FAT score (0–1).

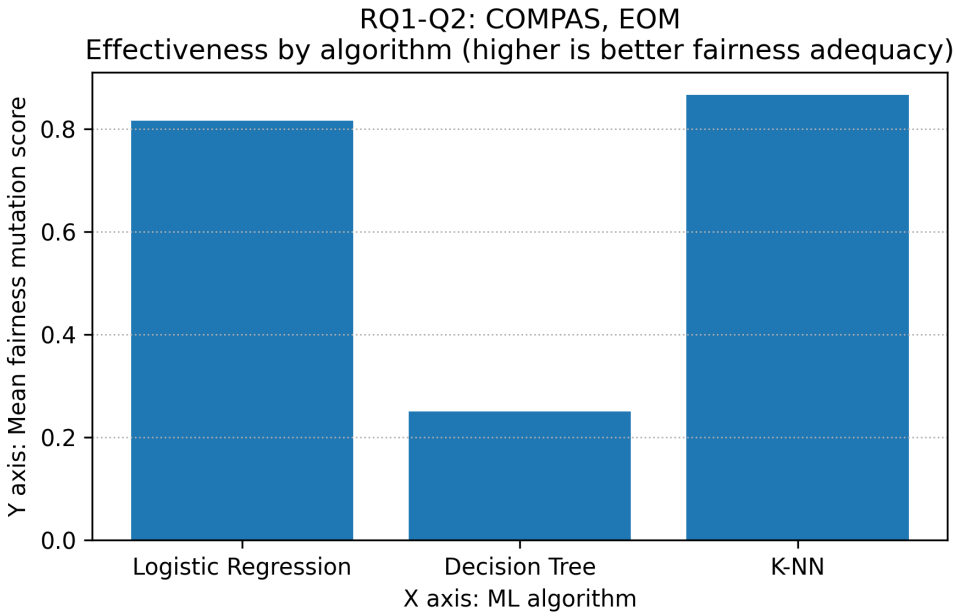

Figure 4.10: COMPAS dataset: algorithm-level FAT under EOM.
X axis: ML algorithm (LR, CART, K-NN).
Y axis: mean FAT score (0–1).

The bar charts show that adequacy varies by learner family and dataset. This suggests that the geometry of the learner (linear vs. tree vs. neighbourhood) interacts with how fairness faults appear, particularly on a higher-stakes dataset like COMPAS.

Effectiveness Across Test Splits (Down-sampling)

We now see what happens when the test set is made smaller (40%, 60%, 80% subsets).

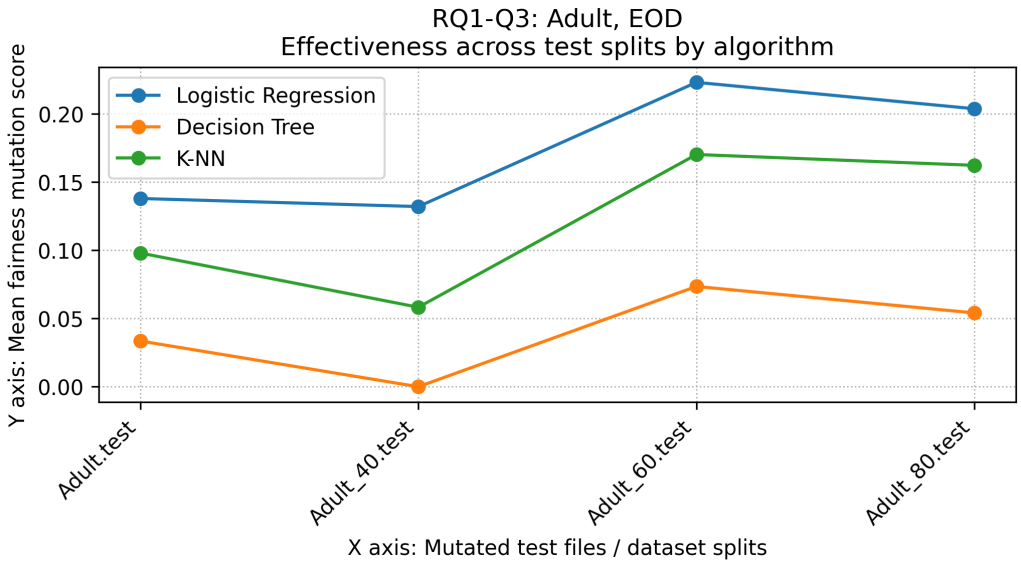

Figure 4.11: Adult dataset: FAT across test splits by algorithm under EOD.
X axis: test splits (Adult.test, Adult_40.test, Adult_60.test, Adult_80.test).
Y axis: mean FAT score, per algorithm.

For both datasets, FAT curves are generally stable across splits, meaning the operators remain useful even when the evaluation slice is smaller. When a drop appears, it is more visible on COMPAS and often under the stricter EOM, which is consistent with the idea that stricter fairness checks can require more data to be reliable.

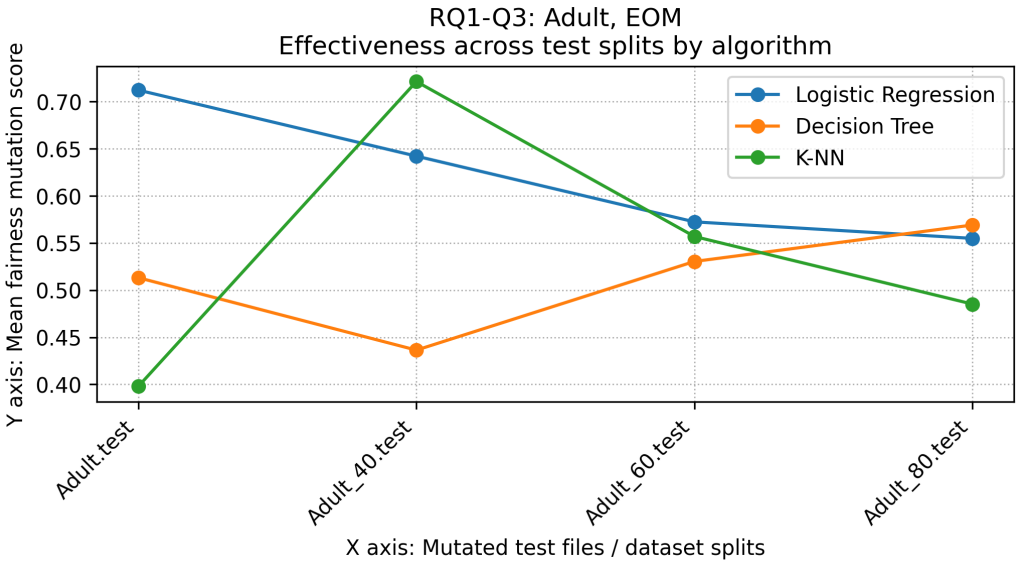

Figure 4.12: Adult dataset: FAT across test splits by algorithm under EOM.
X axis: test splits.
Y axis: mean FAT score, per algorithm.

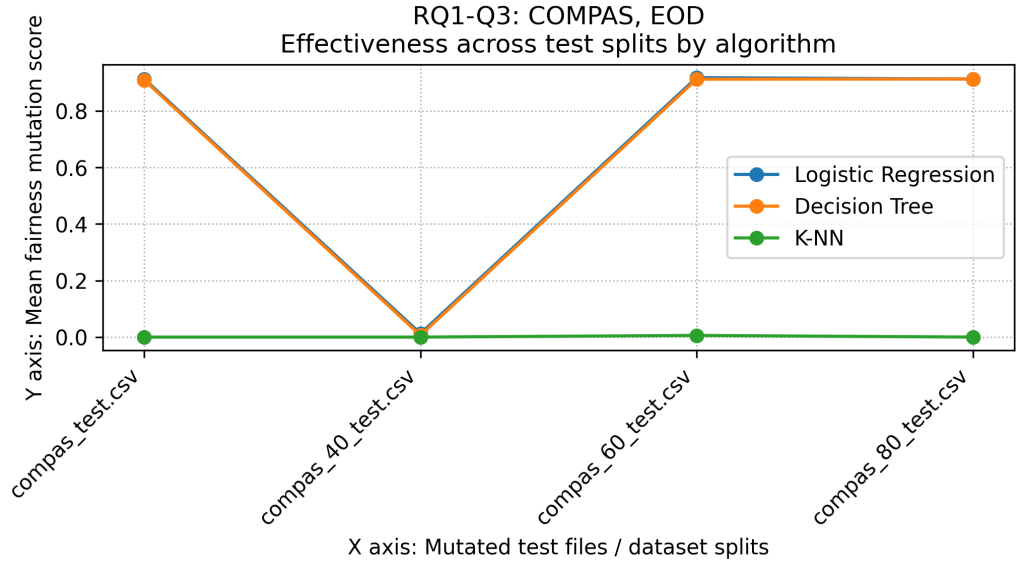

Figure 4.13: COMPAS dataset: FAT across test splits by algorithm under EOD.
X axis: test splits (compas_test.csv, compas_40_test.csv, compas_60_test.csv, compas_80_test.csv).
Y axis: mean FAT score, per algorithm.

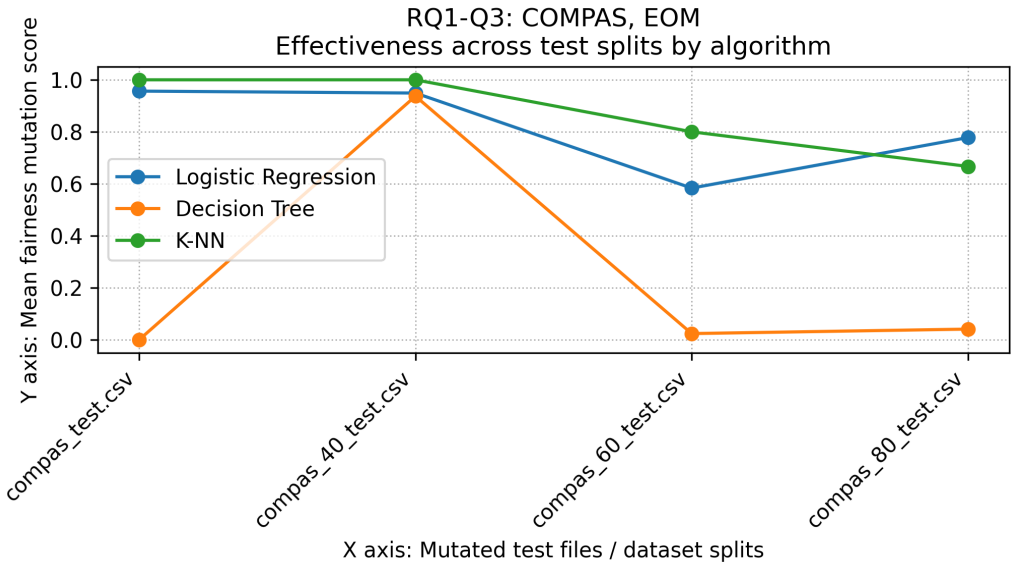

Figure 4.14: COMPAS dataset: FAT across test splits by algorithm under EOM.
X axis: test splits.
Y axis: mean FAT score, per algorithm.

Stability of Adequacy Ordering Across Splits

Finally, we check whether the relative ordering of test files is stable.

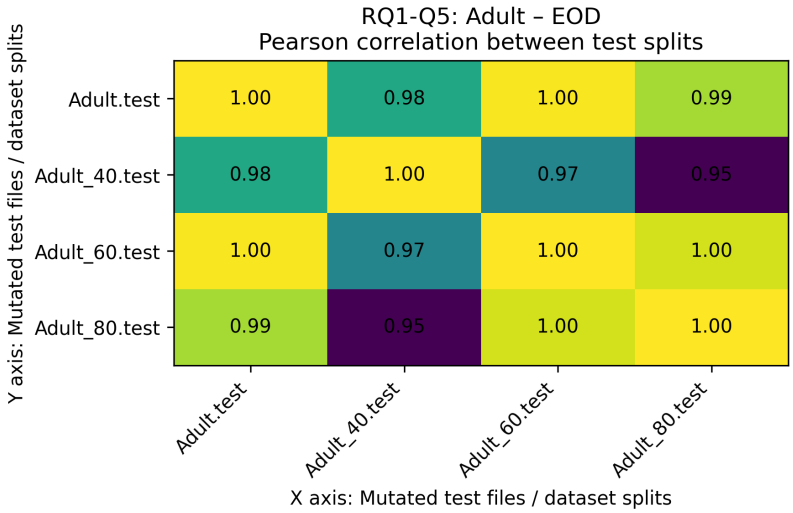

Figure 4.15: Adult dataset: Pearson correlation of EOD FAT scores between test splits.
X/Y axes: test splits.
Cell values: Pearson correlation coefficient.

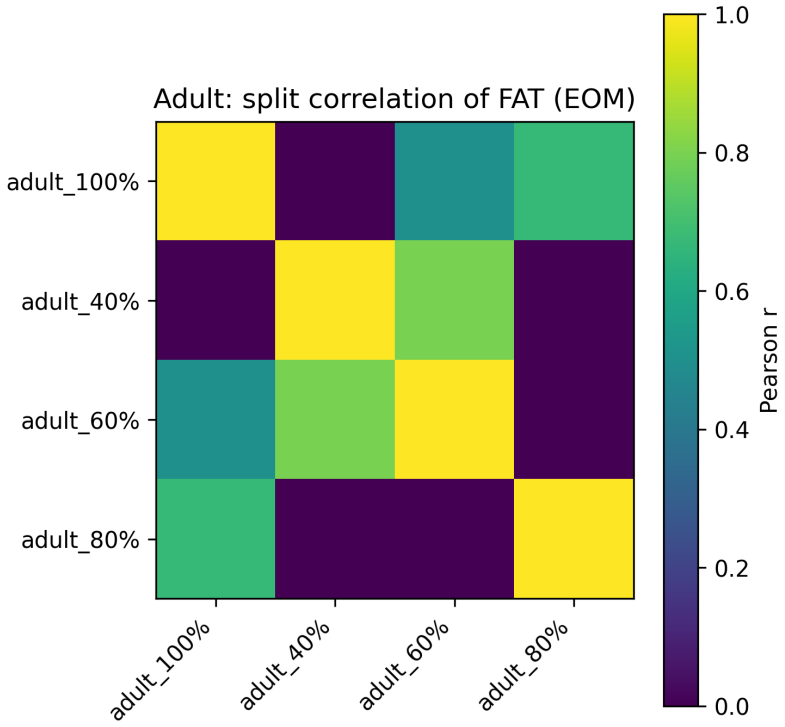

Figure 4.16: Adult dataset: Pearson correlation of EOM FAT scores between test splits.
X/Y axes: test splits.
Cell values: Pearson correlation coefficient.

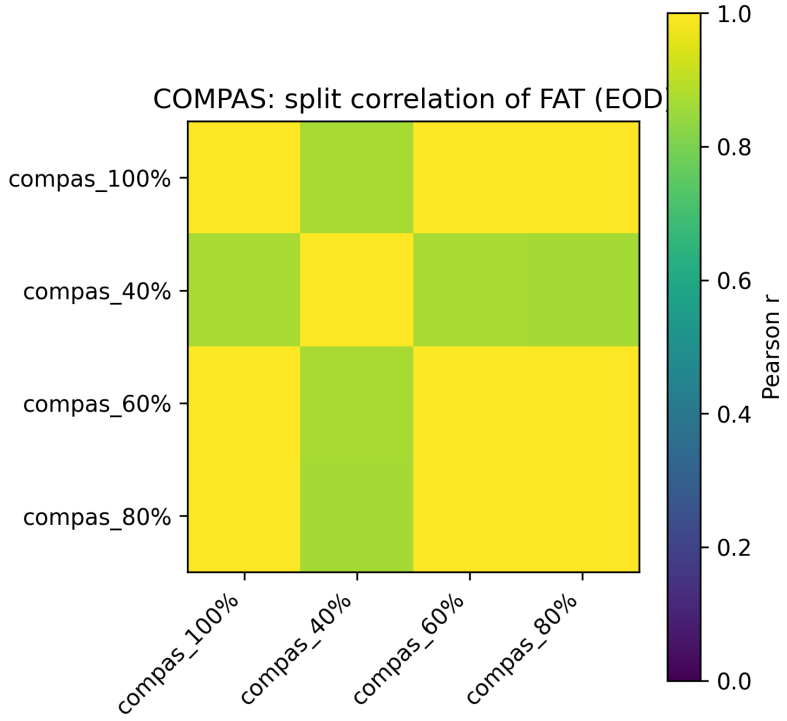

Figure 4.17: COMPAS dataset: Pearson correlation of EOD FAT scores between test splits.
X/Y axes: test splits.
Cell values: Pearson correlation coefficient.

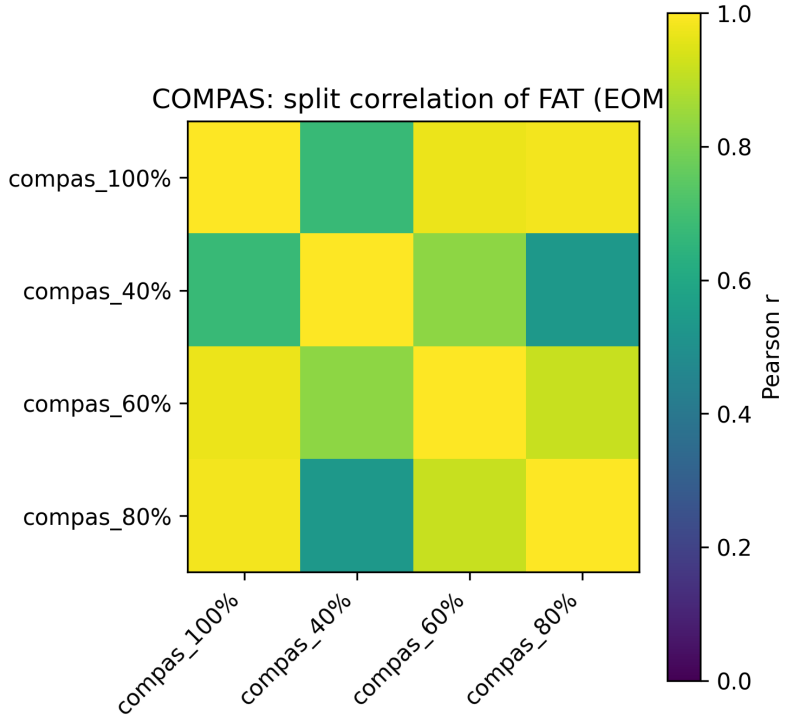

Figure 4.18: COMPAS dataset: Pearson correlation of EOM FAT scores between test splits.
X/Y axes: test splits.
Cell values: Pearson correlation coefficient.

Correlations close to 1 show that the adequacy ranking of splits is stable. For both Adult and COMPAS, and for both EOD and EOM, the heatmaps show generally high correlations. This suggests that mutation-based adequacy is not a fragile artefact of one particular test file, but a consistent signal across slices.

RQ1 summary. Across Adult and COMPAS, the operators create non-trivial fairness faults, and the FAT scores show clear, consistent patterns across datasets, metrics, models and splits. EOM acts as the stricter fairness lens (TPR and FPR), while EOD isolates opportunity gaps (TPR only). Both are useful for understanding how and where fairness faults surface under mutation stress.

4.8.4 RQ2: How Well Does the Framework Perform Across Models and Datasets?

RQ2 shifts the focus from "are the operators useful?" to "is the framework behaving consistently across different model families and datasets?".

We again look at:

- average FAT scores per model,
- detailed heatmaps by model and split,
- correlations of FAT scores across splits.

For Adult, FAT scores tend to be reasonably consistent across models under both metrics. For COMPAS, the differences between models are more pronounced, and the gap between EOD and EOM can widen because Equalized Odds (EOM) reacts to both TPR and FPR shifts.

The heatmaps make three relationships clear:

1. On Adult, adequacy is relatively stable across learners and splits.

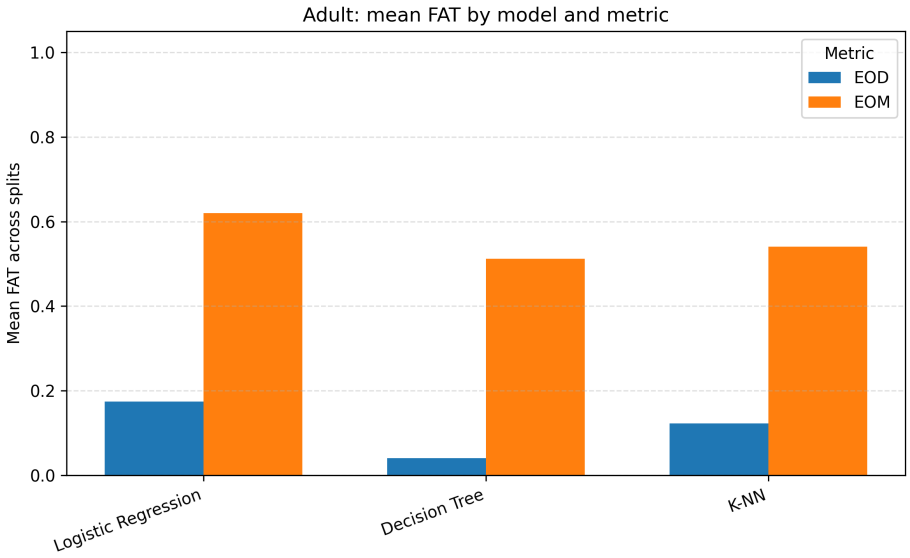

Figure 4.19: Adult dataset: mean FAT scores by model and metric.
X axis: ML algorithm (LR, CART, K-NN).
Y axis: mean FAT over test splits.
Bars compare EOD and EOM.

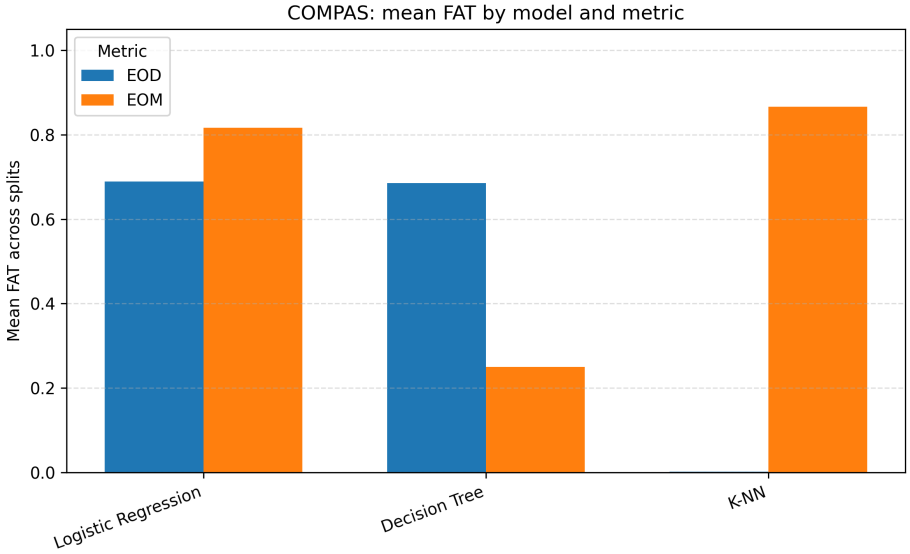

Figure 4.20: COMPAS dataset: mean FAT scores by model and metric.
X axis: ML algorithm.
Y axis: mean FAT over test splits.
Bars compare EOD and EOM.

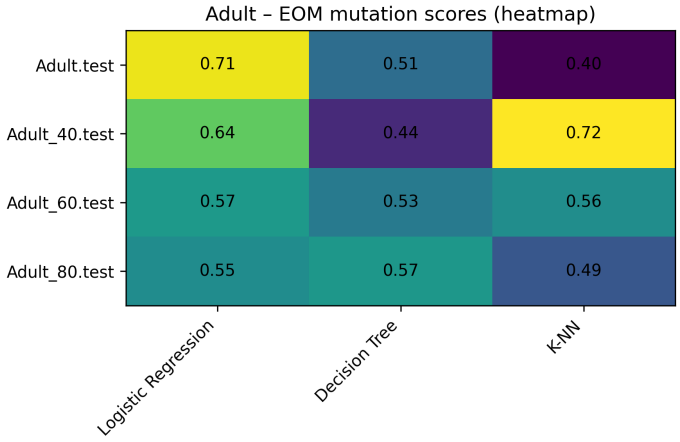

Figure 4.21: Adult dataset: EOM FAT scores by model and test split.
X axis: test files (full, 40%, 60%, 80%).
Y axis: ML algorithms.
Cell values: FAT scores.

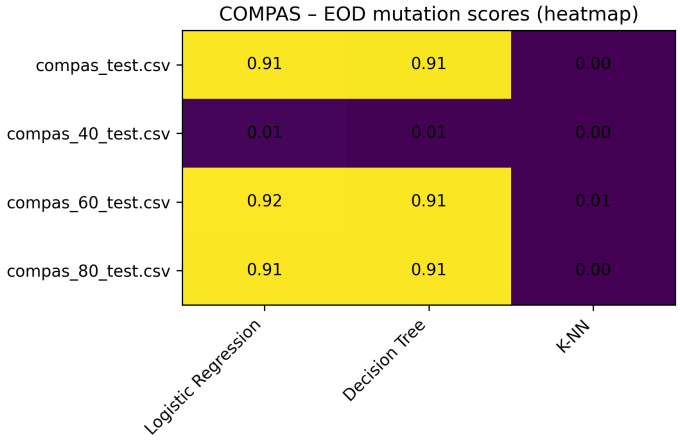

Figure 4.22: COMPAS dataset: EOD FAT scores by model and test split.
X axis: test files.
Y axis: ML algorithms.
Cell values: FAT scores.

2. On COMPAS, adequacy varies more across models and splits, indicating that some model–split combinations are harder to test adequately.

3. Differences between EOD and EOM persist: EOM can behave differently because it reflects both TPR and FPR gaps.

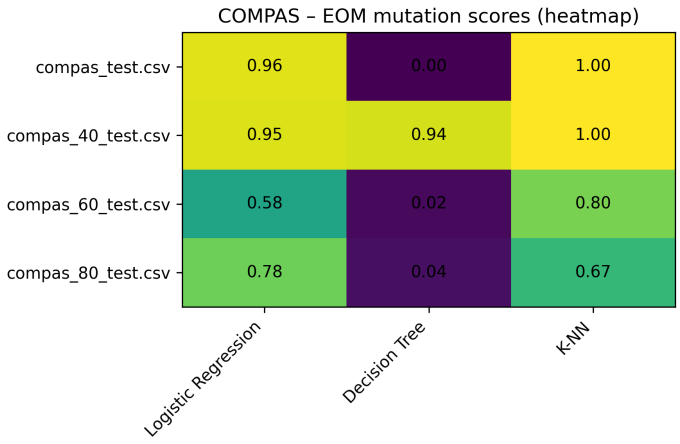

Figure 4.23: COMPAS dataset: EOM FAT scores by model and test split.
X axis: test files.
Y axis: ML algorithms.
Cell values: FAT scores.

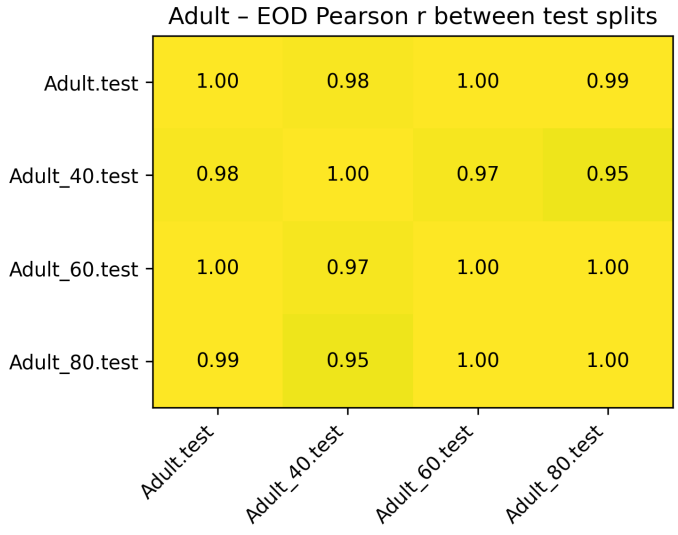

Figure 4.24: Adult dataset: Pearson correlation of EOD FAT scores between test splits.
X/Y axes: Adult test files.
Cell values: Pearson r.

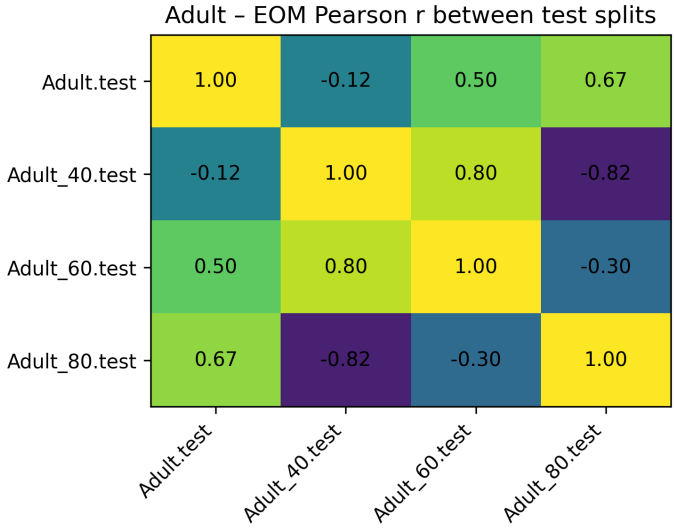

Figure 4.25: Adult dataset: Pearson correlation of EOM FAT scores between test splits.
X/Y axes: Adult test files.
Cell values: Pearson r.

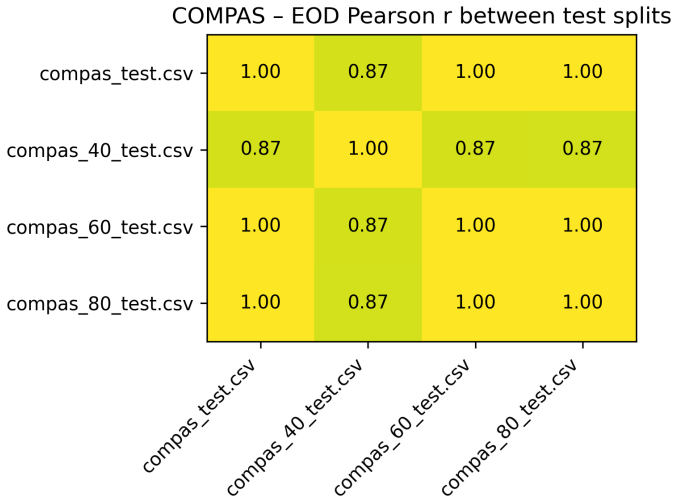

Figure 4.26: COMPAS dataset: Pearson correlation of EOD FAT scores between test splits.
X/Y axes: COMPAS test files.
Cell values: Pearson r.

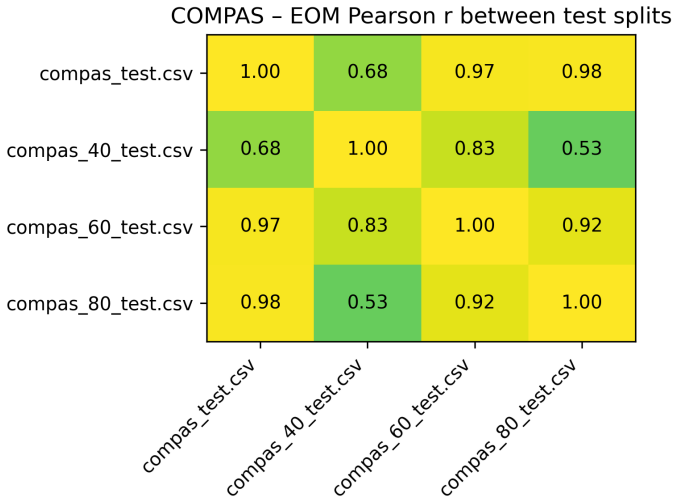

Figure 4.27: COMPAS dataset: Pearson correlation of EOM FAT scores between test splits.
X/Y axes: COMPAS test files.
Cell values: Pearson r.

RQ2 summary. The framework generalises across models and datasets, but not in a trivial way. For Adult, adequacy is relatively stable across learners and splits. For COMPAS, adequacy is more sensitive to model family and sample size, and the gap between EOD and EOM can become more pronounced because EOM reflects both TPR and FPR disparities.

4.8.5 RQ3: How Does Mutation-based Adequacy Compare to Diameter?

Finally, RQ3 compares FAT to a baseline adequacy proxy based on test diversity: the *diameter* of the test suite. The idea behind diameter is simple: larger diameter means more diverse tests, which should, in theory, give higher adequacy.

Here, we ask two questions:

- Do test files with larger diameter also tend to have higher FAT?
- When we put both on the same scale, which measure separates splits more clearly?

The scatter plots show partial alignment: in many cases, larger diameters coincide with higher FAT scores, but the relationship is not perfect. This makes sense: diameter measures geometric diversity, not fairness-specific stress.

To make sensitivity differences clearer, we normalise both signals into $[0, 1]$ and compare them split by split.

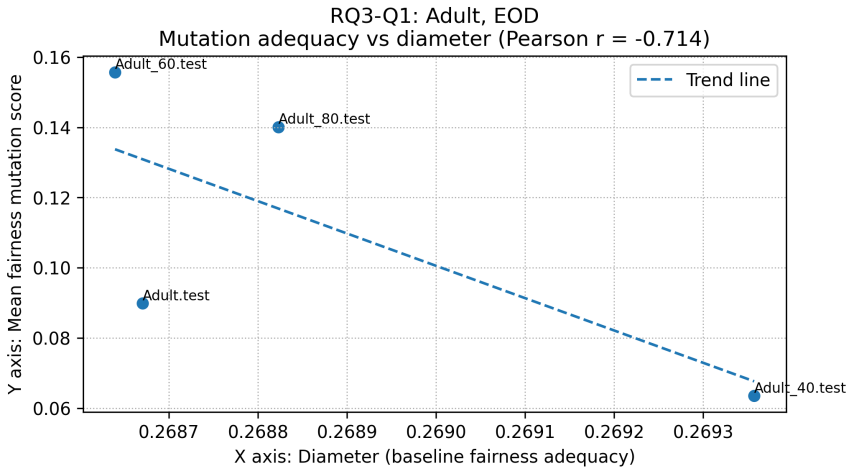

Figure 4.28: Adult dataset: FAT vs diameter under EOD.
X axis: diameter per test file.
Y axis: mean FAT score (0–1).

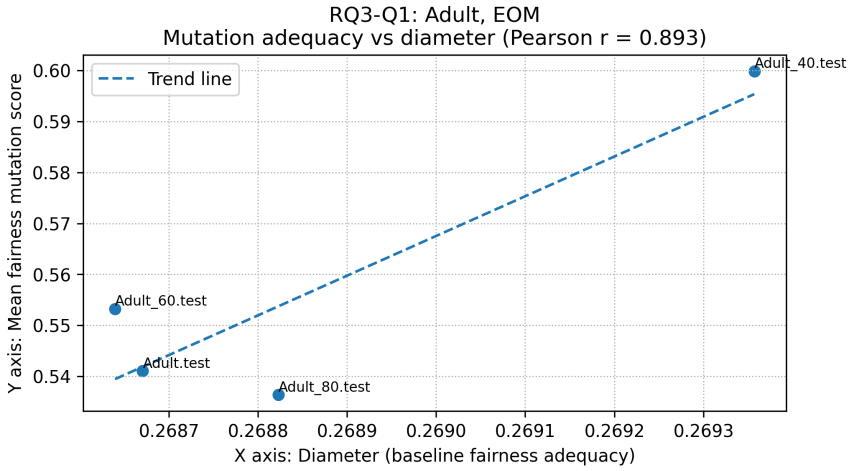

Figure 4.29: Adult dataset: FAT vs diameter under EOM.
X axis: diameter.
Y axis: mean FAT score.

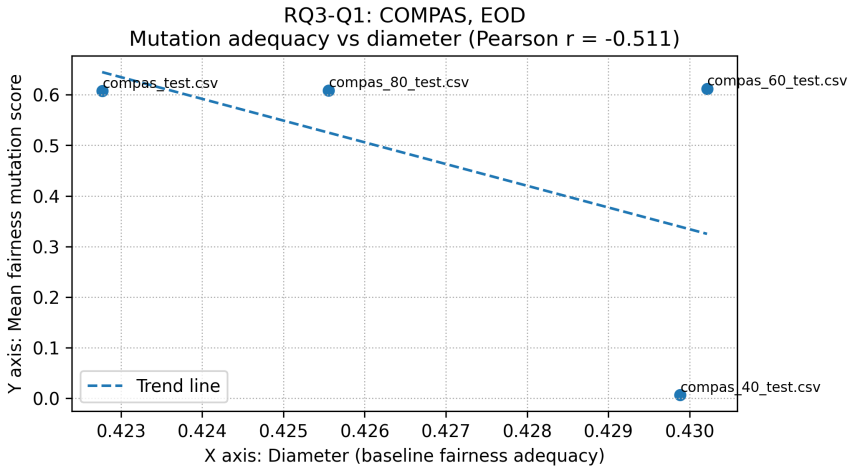

Figure 4.30: COMPAS dataset: FAT vs diameter under EOD.
X axis: diameter.
Y axis: mean FAT score.

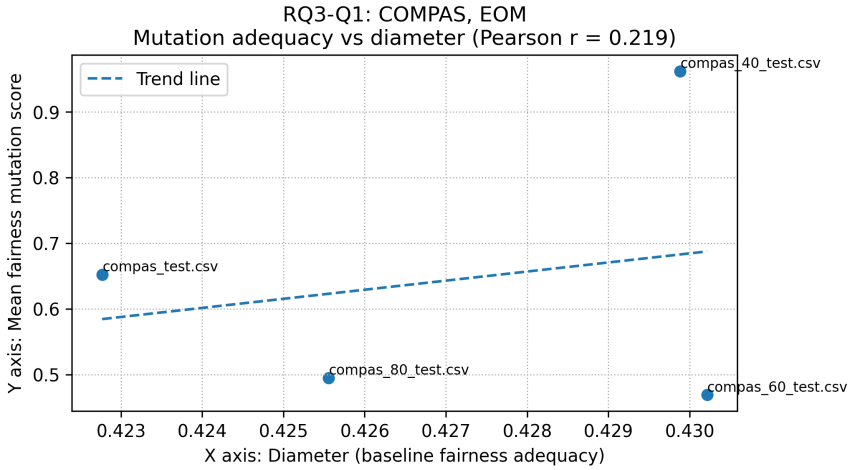

Figure 4.31: COMPAS dataset: FAT vs diameter under EOM.
X axis: diameter.
Y axis: mean FAT score.

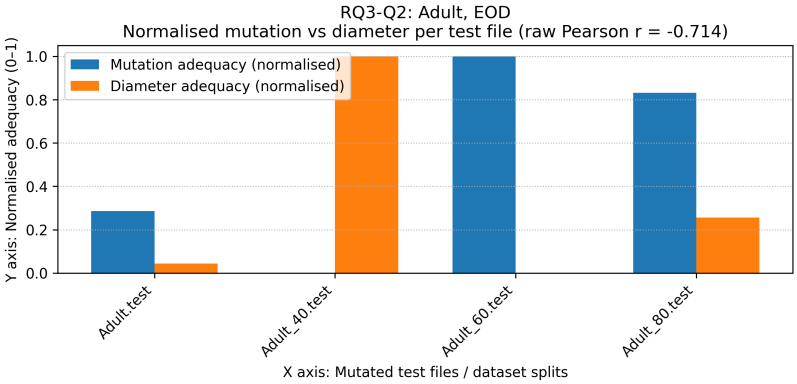

Figure 4.32: Adult dataset: normalised FAT vs diameter per split under EOD.
X axis: test files (full, 40%, 60%, 80%).
Y axis: normalised adequacy (0–1).

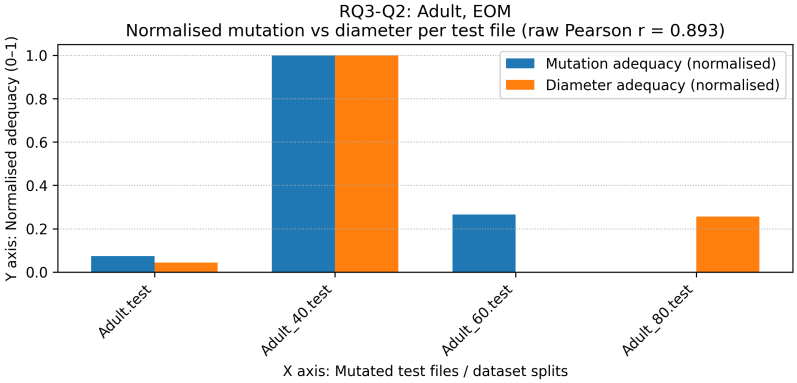

Figure 4.33: Adult dataset: normalised FAT vs diameter per split under EOM.
X axis: test files.
Y axis: normalised adequacy (0–1).

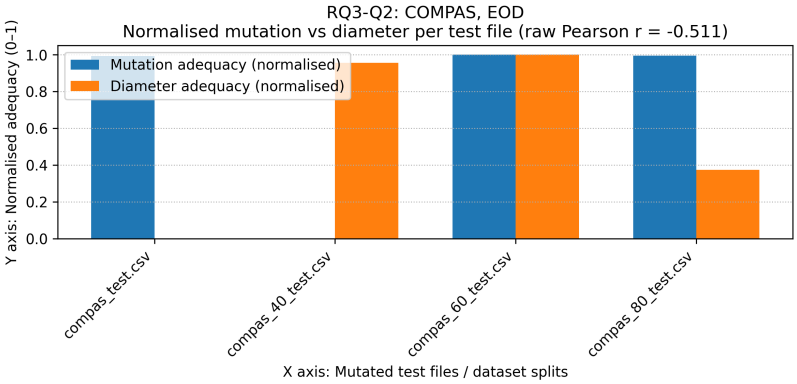

Figure 4.34: COMPAS dataset: normalised FAT vs diameter per split under EOD.
X axis: test files.
Y axis: normalised adequacy (0–1).

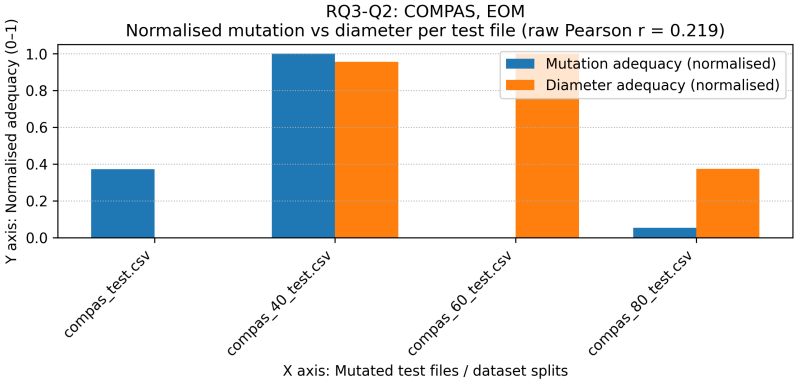

Figure 4.35: COMPAS dataset: normalised FAT vs diameter per split under EOM.
X axis: test files.
Y axis: normalised adequacy (0–1).

Across both datasets and both metrics, two patterns emerge:

- Diameter tends to cluster at similar values across splits, especially for smaller test files, which makes it hard to see real differences in adequacy.
- FAT usually spreads out more, and it can separate splits that look almost identical under diameter, particularly under the stricter EOM lens.

RQ3 summary. Mutation-based fairness adequacy and diameter are related but not redundant. Diameter captures general test diversity, but FAT captures how well the test suite detects fairness-specific faults. In both Adult and COMPAS, and for both EOD and EOM, FAT can provide clearer separation between test files, especially under the stricter Equalized Odds lens (EOM).

Chapter 5

Threats to Validity, Future Work, and Conclusion

This chapter reflects on the key findings of the thesis and discusses the limits of the study. It explains where caution is needed when interpreting the results, identifies factors that can influence fairness measurement, and outlines directions for future research. The goal is to clarify what the empirical results do and do not claim, and to show how the framework proposed in this thesis can be extended and strengthened.

5.1 Threats to Validity

Fairness testing is inherently complex, and several factors can affect how reliable and generalisable the findings are. This section discusses four major categories of validity threats: construct validity, internal validity, external validity, and statistical conclusion validity.

5.1.1 Construct Validity

Construct validity concerns whether the study measured the concepts it intended to measure.

- **Fairness is multi-dimensional.** Although the fairness metrics used in this thesis (e.g., EOD and EOM) are widely used, they do not capture every form of

bias. Group-level metrics provide one important perspective, but they do not cover causal mechanisms, individual fairness, or all context-dependent harms.

- **Operators approximate real changes.** The mutation operators were designed to reflect realistic changes in input data and preprocessing behaviour. However, no finite operator set can cover all fairness failure modes that may arise in real deployments.

- **Thresholding is a design choice.** The drift-threshold rule (mean + 1.5 standard deviations) provides a consistent way to classify mutants as *KILLED* versus *ALIVE*. Different thresholding policies could change which mutants are flagged and therefore alter the measured adequacy levels.

These factors mean the adequacy results in this thesis should be interpreted as evidence about fairness under the specific definitions, metrics, and detection policy used here, rather than as a universal statement of fairness.

5.1.2 Internal Validity

Internal validity concerns whether the observed results could have been influenced by factors within the experimental setup, rather than the framework being evaluated.

- **Training variability.** Fairness outcomes can vary with hyperparameter choices, preprocessing decisions, and randomness from model training procedures.

- **Noise versus true fairness faults.** Some shifts observed during mutation testing may be partially driven by training randomness rather than a meaningful fairness-related change induced by the mutation.

- **Filtering effects.** The validity gate helps remove unrealistic and trivial mutants. However, it may still allow some noisy mutants through, and it may also filter out certain behaviours that could be relevant in some real-world contexts.

Overall, these threats imply that some detected (or missed) fairness shifts may reflect a combination of mutation impact and training instability, rather than mutation impact alone.

5.1.3 External Validity

External validity concerns whether the findings generalise beyond the datasets, models, and settings examined in this thesis.

- **Dataset scope.** The experiments use two benchmark datasets (Adult and COMPAS). While widely used in fairness research, they represent only a subset of real-world decision contexts.

- **Model scope.** The evaluation focuses on three classical model families (LR, CART, and K-NN). Many deployed systems use more complex models and pipelines that may exhibit different fairness drift behaviour.

- **Data modality and structure.** The current operator set and evaluation focus on tabular data. Other modalities (text, images, time-series, multi-modal systems) and other sensitive-attribute structures may require new operators, new metrics, or both.

For these reasons, applying the framework to new domains should be done with care, and broader evaluation is needed to confirm generality.

5.1.4 Statistical Conclusion Validity

Statistical conclusion validity concerns the reliability of inferences drawn from measured results.

- **Small subgroups.** Some protected subgroups are relatively small, which can make fairness metrics unstable. This is especially relevant for metrics based on conditional rates.
- **Many-mutant evaluation.** Mutation testing evaluates many mutants, which increases the risk of both false positives and false negatives under any fixed detection rule.
- **Adequacy depends on non-equivalent counts.** The fairness adequacy score depends on the number of non-equivalent mutants ($M_{\text{primary}} - E_e$). If a large portion of mutants are filtered as equivalent or invalid, the score may underrepresent the range of fairness behaviours that could occur.

These considerations motivate interpreting fairness adequacy alongside subgroup sizes and uncertainty indicators, rather than treating a single score as a final verdict.

5.2 Future Work

Several extensions can strengthen fairness adequacy testing and broaden its practical use.

5.2.1 Expanding Datasets and Domains

Future research should evaluate the framework across a wider set of datasets and decision contexts, including healthcare, finance, hiring, and education. Extending the

evaluation to more model families (e.g., gradient boosting, neural networks) and to non-tabular data (text, images, time-series, and multi-modal systems) would significantly increase the scope and practical relevance of the approach.

5.2.2 Richer Fairness Definitions

This study focuses on group-level parity metrics. Future work can incorporate additional fairness perspectives, including:

- causal fairness criteria,
- individual fairness,
- intersectional fairness (e.g., race $\times$ sex),
- fairness over time in sequential or streaming decision processes.

Adding these perspectives would provide a deeper and more complete view of fairness failures, especially in settings where group averages hide subgroup harms.

5.2.3 Adaptive and Slice-aware Thresholding

The thresholding rule used in this thesis was intentionally simple and consistent across experiments. A promising direction is to design adaptive thresholds that account for:

- subgroup sample sizes,
- class imbalance and base rates,
- uncertainty estimates for fairness rates,
- joint behaviour across multiple fairness metrics.

Such methods could reduce over-flagging from noisy subgroups while improving sensitivity to meaningful fairness drift.

5.2.4 Enhanced Mutation Operators

New operators could better reflect how fairness drift emerges in practice. Examples include:

- temporal shifts in feature distributions,
- realistic noise and missingness patterns that disproportionately affect protected groups,
- evolving population composition and sampling bias,
- adversarial manipulations intended to hide or amplify unfairness,
- policy-driven changes (including changes to decision thresholds and operating points).

In addition, operator minimisation and prioritisation is an important practical problem: identifying a small subset of operators that provides strong fairness fault coverage would make the framework easier to deploy as a routine check.

5.2.5 Tooling and Practical Integration

To support adoption in real ML pipelines, fairness adequacy testing can be integrated into engineering workflows, for example through:

- automated adequacy reports and dashboards,
- CI/CD fairness checks that run alongside unit and integration tests,
- integration with monitoring systems for post-deployment drift,
- documentation formats that align with model governance and audit requirements.

This direction would help teams treat fairness not only as a reporting activity, but also as a testable quality property with measurable regression signals.

5.3 Conclusion

This thesis introduced a fairness adequacy framework that applies mutation testing to evaluate the strength of fairness assessments. Rather than relying only on static baseline fairness values, the framework stresses models with controlled perturbations and measures whether fairness drift is detected.

Experiments on the Adult and COMPAS datasets show that:

- fairness-aware mutation operators can induce meaningful and interpretable fairness shifts,
- adequacy scores vary by dataset, model family, and fairness metric,
- down-sampled test sets (40%, 60%, 80%) can still produce informative adequacy signals,
- adequacy patterns are generally stable across different test splits,
- mutation-based adequacy can provide a clearer fairness-specific signal than a general diversity proxy such as diameter.

Taken together, the results support the main claim of the thesis: fairness can be treated as a testable property, and mutation-based adequacy provides a practical and repeatable way to evaluate whether an evaluation setup is strong enough to detect fairness-relevant faults before deployment.

While additional work is needed to broaden fairness definitions, expand dataset coverage, and integrate the approach into real-world tooling, this study provides a

concrete foundation for fairness adequacy testing as part of standard ML quality assurance.

BIBLIOGRAPHY

[1] Agarwal, A., Beygelzimer, A., Dudík, M., Langford, J., and Wallach, H. A reductions approach to fair classification.

[2] Ammann, P., and Offutt, J. *Introduction to Software Testing*, 2 ed. Cambridge University Press, 2016.

[3] Angwin, J., Larson, J., Mattu, S., and Kirchner, L. Machine bias. *ProPublica* (2016).

[4] Barocas, S., Hardt, M., and Narayanan, A. *Fairness and Machine Learning: Limitations and Opportunities*. 2023. Available online at fairmlbook.org.

[5] Behrouz, A., Zhong, P., and Mirrokni, V. Titans: Learning to memorize at test time, 2024.

[6] Bellamy, R. K. E., Dey, K., Hind, M., Hoffman, S. C., Houde, S., Kannan, K., Lohia, P., Martino, J., Mehta, S., Mojsilović, A., Nagar, S., Ramamurthy, K. N., Richards, J., Saha, D., Sattigeri, P., Singh, M., Varshney, K. R., and Zhang, Y. Ai fairness 360: An extensible toolkit for detecting and mitigating algorithmic bias. *IBM Journal of Research and Development 63*, 4/5 (2019), 4:1–4:15.

[7] Bishop, C. M. *Pattern Recognition and Machine Learning.* Springer, New York, 2006.

[8] Chouldechova, A. Fair prediction with disparate impact: A study of bias in recidivism prediction instruments. *Big Data 5*, 2 (2017), 153–163.

[9] Consumer Financial Protection Bureau. Consumer financial protection circular 2022-03: Adverse action notification requirements in credit decisions. `https://www.consumerfinance.gov/`, 2022.

[10] Consumer Financial Protection Bureau. Adverse action notice requirements. `https://www.consumerfinance.gov/`, 2023. Guidance on notices when using AI in credit decisions.

[11] Corbett-Davies, S., Gaebler, J. D., Nilforoshan, H., Shroff, R., and Goel, S. The measure and mismeasure of fairness, 2023.

[12] Dastin, J. Amazon scraps secret ai recruiting tool that showed bias against women. *Reuters* (2018).

[13] Dwork, C., Hardt, M., Pitassi, T., Reingold, O., and Zemel, R. Fairness through awareness. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference* (New York, NY, USA, 2012), ITCS '12, Association for Computing Machinery, pp. 214–226.

[14] European Union. Eu artificial intelligence act. `https://eur-lex.europa.eu/`, 2025.

[15] Feldman, M., Friedler, S., Moeller, J., Scheidegger, C., and Venkatasubramanian, S. Certifying and removing disparate impact, 2015.

[16] Galhotra, S., Brun, Y., and Meliou, A. Fairness testing: Testing software for discrimination. In *Proceedings of the 11th Joint Meeting on Foundations of Software Engineering* (2017), pp. 498–510.

[17] Grote, T. Fairness as adequacy: a sociotechnical view on model evaluation in machine learning. *AI and Ethics 4*, 2 (2024), 427–440.

[18] Hardt, M., Price, E., and Srebro, N. Equality of opportunity in supervised learning, 2016.

[19] Henley, J. Dutch government resigns over child welfare scandal. *The Guardian* (2021).

[20] Kearns, M., Neel, S., Roth, A., and Wu, Z. Preventing fairness gerrymandering: Auditing and learning for subgroup fairness.

[21] Kusner, M. J., Loftus, J. R., Russell, C., and Silva, R. Counterfactual fairness, 2018.

[22] Ma, L., Zhang, F., Sun, J., Xue, M., Li, B., Juefei-Xu, F., Xie, C., Li, L., Liu, Y., Zhao, J., and Wang, Y. Deepmutation: Mutation testing of deep learning systems. In *Proceedings - 29th IEEE International Symposium on Software Reliability Engineering, ISSRE 201818* (2018), vol. 2018, IEEE, pp. 100–111.

[23] Mehrabi, N., Morstatter, F., Saxena, N., Lerman, K., and Galstyan, A. A survey on bias and fairness in machine learning. *ACM Computing Surveys 54*, 6 (2021), 1–35.

[24] Mobley, D. Mobley v. workday, inc. — court filings and orders. `https://www.courtlistener.com/`, 2025.

[25] Myers, G. J., Sandler, C., and Badgett, T. *The Art of Software Testing*, 3 ed. Wiley, 2011.

[26] Obermeyer, Z., Powers, B., Vogeli, C., and Mullainathan, S. Dissecting racial bias in an algorithm used to manage the health of populations. *Science 366*, 6464 (2019), 447–453.

[27] of Representatives, D. H. Ongekend onrecht: Parliamentary interrogation committee on childcare allowance. Tech. rep., Tweede Kamer der Staten-Generaal, January 2021.

[28] Ofqual. Awarding gcse, as and a levels in summer 2020: interim report. Tech. rep., Office of Qualifications and Examinations Regulation, August 2020.

[29] Ovadia, Y., Fertig, E., Ren, J., Nado, Z., Sculley, D., Nowozin, S., Dillon, J. V., Lakshminarayanan, B., and Snoek, J. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift, 2019.

[30] Sample, I. A-level results: how did the algorithm work and why did it fail? *The Guardian* (2020).

[31] Sharma, A., and Wehrheim, H. Testing machine learning programs. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (2020), vol. 12471 LNCS, pp. 5–26.

[32] Tramer, F., Atlidakis, V., Geambasu, R., Hsu, D., Hubaux, J.-P., Humbert, M., Juels, A., and Lin, H. Fairtest: Discovering unwarranted associations in data-driven applications. 401–416.

FAIRNESS ADEQUACY TEST FOR MACHINE LEARNING SYSTEMS

by

Kehinde Oluwasayo Akinola

December, 2025

Director of Thesis: Srinivasan Madhusudan, PhD

Major Department: Computer Science

As Machine Learning (ML) systems assume a larger role in decisions that affect people's lives, such as who receives a loan, access to healthcare, or early release from prison, ensuring these systems are fair is more important than ever. However, current fairness checks often miss the subtle and complex ways in which bias can appear in algorithms.

This dissertation tackles that gap by proposing a practical framework for testing how well machine learning models meet fairness standards, especially across protected groups. Instead of relying solely on standard performance metrics, the approach combines statistical tools with stress testing techniques to uncover hidden or overlooked biases.

There are four main contributions. First, we introduce a fairness adequacy test using metrics like Equal Opportunity Difference (EOD) and Equalised Odds Metrics (EOM) to examine disparities in error rates across groups. Second, we apply mutation testing by altering sensitive features such as race or gender to see how model outputs change, helping assess fairness under different conditions. Third, we use permutation methods to simulate edge cases and test how models respond to unusual or extreme inputs. Finally, we validate this approach with real-world case studies in areas like healthcare, finance, and criminal justice, where fairness is especially critical.

By offering a clear and testable way to evaluate fairness, this work aims to support

the development of more trustworthy, accountable, and equitable AI systems

FAIRNESS ADEQUACY TEST FOR MACHINE LEARNING SYSTEMS

A Thesis

Presented to The Faculty of the Department of Computer Science

East Carolina University

In Partial Fulfillment of the Requirements for the Degree

Master of Science in Data Science

by

Kehinde Oluwasayo Akinola

December, 2025

FAIRNESS ADEQUACY TEST FOR MACHINE LEARNING SYSTEMS

by

Kehinde Oluwasayo Akinola

APPROVED BY:

DIRECTOR OF THESIS:	Srinivasan Madhusudan, PhD
COMMITTEE MEMBER:	Peng Kebin PhD
COMMITTEE MEMBER:	John Reisch, PhD
CHAIR OF THE DEPARTMENT OF COMPUTER SCIENCE:	Qin Ding, PhD
DEAN OF THE GRADUATE SCHOOL:	Debra Jackson, PhD

Table of Contents

LIST OF TABLES

LIST OF FIGURES

List of Acronyms

AI Artificial Intelligence. xii

AUC Area Under the Curve. xii

CART Classification and Regression Trees.. v, xii, 17, 20, 30, 49, 56, 61, 66–68, 76

CNNs Convolutional Neural Networks. xii

DBSCAN Density-Based Spatial Clustering of Applications with Noise. xii

DNNs deep neural networks. xii, 8

DP Disparity Parity. xii, 51

EO Equality of Opportunity. xii, 9

EOD Equal Opportunity Difference. xii, 1, 8, 9, 25, 26, 30, 37, 49, 51, 53–55, 57–62, 64–71, 73, 75–84, 86, 87, 89

EOM Equalised Odds Metrics. xii, 1, 8, 16, 25, 26, 30, 37, 49, 51, 55, 57–62, 64–72, 74–80, 82–87

FAT Fairness Adequacy Test. xii, 39, 49

FPR False Positive Rate. xii, 51, 55

K-NN K-Nearest Neighbors. v, x, xii, 17, 22–25, 30, 33, 48, 49, 56, 61, 66–68, 76

LightGBM Light Gradient Boosting Machine. xii

LR Logistic Regression. v, xii, 17, 19, 30, 49, 56, 61, 66–68, 76

MKR mutation kill rates. xii, 29, 30

ML Machine Learning. xii, 1, 2, 6, 9, 10, 14–16, 25, 28

MSR Mutation Score Rate. xii

PCA Principal Component Analysis. xii

Q-Learning Quality Learning. xii

RNNs Recurrent Neural Networks. xii

ROC Receiver Operating Characteristic. xii

RQ Research Questions. xii

SARSA State–Action–Reward–State–Action. xii

SPD Statistical parity difference. xii, 16, 19, 22, 25, 26, 30, 36, 49, 55

SVM Support Vector Machines. xii

t-SNE t-distributed Stochastic Neighbor Embedding. xii

TPR True Positive Rate. xii, 51, 54

UMAP Uniform Manifold Approximation and Projection. xii

WMFI Weighted Multi-Objective Fairness Index. xii, 49, 51

XGBoost Extreme Gradient Boosting. xii

Chapter 1

Introduction

1.1 What Is Testing and the Adequacy Principle

A model can report 95% accuracy and still deny loans to a protected group 20% more often than to others. This gap between statistical performance and social impact shows why testing remains central to responsible deployment.

In classical software engineering, adequacy asks whether a test suite exercises the paths most likely to reveal defects (21). In machine learning, this idea needs to be adapted. Models learn from historical data, make probabilistic predictions, and operate in social contexts that can change over time (17). Adequacy therefore moves from *code coverage* to *slice coverage*: systematic evaluation of fairness-relevant subpopulations, feature contexts, and plausible distributional shifts.

A fairness test suite should be able to reveal faults that affect fairness, not only those that reduce overall accuracy. From a sociotechnical perspective (17), we apply precise measurements of error rates to outcomes that matter for individuals and communities. In this thesis, we operationalise this idea through mutation testing. We create small but realistic changes (mutants) in the data, enforce basic distribution and validity checks, retrain the model where needed, and then check whether the test suite detects a meaningful shift in fairness metrics such as Equal Opportunity Difference (EOD) or Equalized Odds Metric (EOM). When a fairness metric changes

beyond a data-driven threshold, the mutant is considered *killed*; otherwise it *survives*. The kill-versus-survive decision uses a control limit derived from the base model's null variability (mean + 1.5 std), estimated via bootstrap resampling.

In this way, adequacy is reframed as the ability of a test suite to detect fairness-relevant faults under realistic perturbations, not just to report fairness numbers on a single, fixed test set.

1.2 Why Fairness Testing Is Necessary

Standard evaluation metrics such as accuracy, precision, and recall often fail to reveal harms that arise in specific subgroups due to historical bias, proxy labels, or distribution shift (23; 31). Because ML now informs decisions in areas such as credit, employment, criminal justice, and health, these hidden failures can scale into systemic discrimination.

Fairness testing treats equity as a measurable property of an AI system. It is the process of checking whether a model produces outcomes that differ across groups (for example, by gender, race, or age) without a justified reason, and of recommending targeted corrections rather than generic fixes (16; 32). Tools such as FairTest and Themis detect statistically significant disparities in a practical way, spanning tabular data, text, and vision (32).

A widely discussed example underlines this need. An internal audit showed that Amazon's experimental hiring tool systematically downgraded CVs that contained women-associated terms, even though overall performance looked competitive (12). The audit did not merely report accuracy; it used fairness-oriented checks to show that the model was not performing equally well across groups. This kind of evidence illustrates why fairness testing must complement standard utility metrics.

1.3 Global Incidents Caused by Inadequate Fairness Testing

Several high-profile incidents show how insufficient fairness testing can cause concrete harm. Each case pairs a missing test signal with an adequacy lesson that this thesis aims to make operational.

- **Criminal justice: COMPAS risk assessment (2016).** Empirical studies showed that the COMPAS algorithm assigned high-risk labels to Black defendants at much higher rates than to white defendants, even though both groups had similar rearrest rates (3; 8). This unequal treatment reflects a methodological gap: subgroup-specific confusion matrices and uncertainty assessments were not treated as standard evaluation artefacts.

- **Hiring: Amazon recruiting prototype (2018).** Investigations reported that Amazon's experimental résumé-filtering system, trained largely on male-dominated historical data, downgraded résumés containing women-associated terms (12). The lack of shortlist parity checks and feature-attribution audits meant that this behaviour went undetected until late in the process.

- **Healthcare: Risk scoring (2019).** A widely used healthcare risk scoring algorithm relied on healthcare costs as a proxy for clinical need, which led to Black patients being under-referred to high-risk care programmes despite having similar clinical risk to white patients (26). This bias highlights the absence of label-validity assessments and subgroup-specific error analyses in the evaluation pipeline.

- **Education: UK A-level grading algorithm (2020).** Evaluations of the grading system used during the COVID-19 pandemic showed that the standardisation procedure disproportionately downgraded students from certain schools

(28; 30). The failure to test on small-cohort subsets and to apply strong distributional stress tests contributed to these skewed outcomes.

- **Public benefits: Dutch childcare scandal (2018–2021).** In the Dutch childcare benefits scandal, automated profiling systems misclassified thousands of families as fraudsters and disproportionately targeted dual-citizen households (27; 19). Here, the system failed to account for protected characteristics and allowed proxy features to drive decisions, which resulted in inequitable and discriminatory outcomes.

These incidents are not isolated technical errors; they are signs that fairness-relevant behaviour was not adequately tested before deployment.

1.4 Implications of Not Conducting Fairness Testing

The incidents above illustrate at least four types of risk that arise when fairness testing is weak or absent:

1. **Legal and compliance.** Lack of documented fairness testing can lead to investigations, penalties, and lawsuits.
2. **Institutional trust.** Exposed disparities reduce the confidence of users, regulators, and the public, and increase the cost of assurance.
3. **Technical debt.** Aggregate metrics can hide fairness defects that appear only in specific slices or under realistic shifts, making later remediation more complex.
4. **Social and ethical harm.** Unchecked algorithms can deepen disadvantage in finance, employment, healthcare, education, and legal decision-making.

Table 1.1: Documented failures linked to missing or inadequate fairness testing across critical domains.

Domain	**Incident (Year)**	**Outcome**	**Type of Harm**	**Missed Fairness Signal**
Justice	COMPAS bias in sentencing (2016)	Black defendants received higher risk scores	Racial discrimination	No subgroup confusion matrix or uncertainty analysis
Employment	Amazon résumé screening (2018)	Female-associated terms were downgraded	Gender bias	No feature-attribution audit or shortlist parity check
Healthcare	Risk prediction model (2019)	Black patients underreferred despite equal risk	Proxy-label bias	No label-validity check or subgroup error analysis
Recruitment	Automated screening lawsuit (2022)	Older and Black candidates excluded	Age and race discrimination	No slice-level evaluation or intersectional coverage
Lending	Credit scoring disparities (2021)	Minority borrowers received worse terms	Disparate impact	No distributional stress test or sensitive attribute guardrails

Table 1.2: Disparities in COMPAS risk labels and rearrest rates by racial group (ProPublica analysis).

Group	**Labeled High Risk (%)**	**Actual Rearrested (%)**
Black	45	20
White	23	13

Source: Adapted from ProPublica analysis (3).

As shown in Tables 1.1 and 1.2, disparities can remain hidden if evaluation focuses only on global accuracy or utility scores. The evidence suggests that fairness testing should be treated alongside accuracy, reliability, and security as a basic requirement for responsible AI. In practice, this means setting explicit slice-coverage goals, using uncertainty-aware thresholds, summarising fairness-sensitive mutants with mutation scores and confidence intervals, and preserving artefacts that show that fairness-altering mutants would have been detected before deployment.

1.5 Importance of Solving This Problem

There are strong reasons, both ethical and practical, to require fair behaviour from ML systems, especially in sensitive domains such as credit scoring, policing, and healthcare. Fair models help reduce harm, support trust, and make it more likely that all demographic groups receive a fair chance at resources and opportunities. Addressing fairness is therefore not only a technical goal but also a condition for responsible and sustainable deployment (6).

From a technical perspective, fairness testing has a clear advantage: it can reveal problematic behaviour that would otherwise remain hidden when only overall performance metrics are considered. It encourages systematic error analysis across different subgroups and feature contexts rather than relying on a single global accuracy score. In this thesis, fairness is treated as a *test adequacy requirement* rather than only a diagnostic report: we ask whether the test suite is strong enough to uncover fairness-related bugs before deployment (7).

To operationalise this requirement, we simulate small, realistic data changes under distributional constraints and check whether the fixed test suite detects changes in fairness metrics. A mutant is labelled *killed* when the absolute change in a fairness

metric exceeds a threshold derived from the base model's null variability (mean + 1.5 std). The resulting mutation score measures how well the test suite is exposed to fairness defects through sensitive slices and plausible shifts. This forms the basis for an adequacy-style certificate indicating whether fairness remains within permissible limits (1).

Taken together, these elements support the view that fairness should be built into the testing process from the start, not treated as a later add-on. Embedding fairness adequacy into testing supports technical robustness and aligns with broader social expectations that automated decision systems should not discriminate.

1.5.1 Foundational Fairness Definitions and Trade-offs

The formal study of algorithmic fairness began with work such as (15), which introduced early definitions of disparate impact and proposed preprocessing methods to mitigate data bias. These ideas provided an initial technical basis for fairness research. Building on this, (18) proposed the principle of Equal Opportunity, which requires that demographic groups have similar true positive rates. Because it is grounded in supervised learning and can be measured from confusion matrices, Equal Opportunity quickly became a central fairness notion.

Subsequent research examined how fairness criteria interact. (8) proved that it is impossible, in general, to satisfy both calibration and equal error rates when groups have different base rates. This result formalised an important trade-off between fairness measures and underscored the need to make fairness goals explicit. Extending this perspective, (21) introduced counterfactual fairness, which requires model decisions to remain unchanged under hypothetical variations in an individual's protected attributes. This causal framing emphasises the importance of understanding how sensitive features influence outcomes through the underlying data-generating processes.

These foundational definitions and trade-offs motivate the choice of metrics in this thesis, especially EOD, EOM, and related group-based criteria.

1.5.2 Mutation and Metamorphic Testing for ML Robustness

Mutation testing was first developed in software engineering as a strong test adequacy criterion. The idea is to introduce small faults (mutants) into the program and check whether the test suite can detect them (25). If many mutants survive, the test suite is considered weak. In machine learning, a related approach is metamorphic testing, which applies input transformations that should preserve certain properties of the output in order to address the oracle problem. For example, DeepTest applies metamorphic transformations to driving scenes to detect failures in autonomous vehicle systems (5).

DeepMutation (22) extended mutation testing to deep neural networks (deep neural networks (DNNs)). Mutants are created at the source level (for example, by altering training data, labels, or architecture) and at the model level (for example, by perturbing neurons or weights), yielding multiple mutated model variants. When the test suite fails to kill these mutants by exposing performance degradation, it reveals gaps in evaluation and shows that high accuracy alone does not guarantee robust behaviour.

This thesis adapts this methodology to fairness evaluation. Instead of focusing only on robustness to generic perturbations, we design fairness-oriented mutations, such as perturbations that stress decision boundaries across protected groups or reweight subpopulations. If fairness tests fail to detect the induced disparities, the evaluation suite is judged inadequate. Mutation testing thus provides a principled way to extend test adequacy from robustness into fairness.

1.5.3 Empirical Failures Motivating Fairness Evaluation

Empirical failures such as the COMPAS disparities (3) motivated the adoption of fairness metrics like Equal Opportunity (Equality of Opportunity (EO)) and Equalized Odds (EOD). Similarly, (26) showed that a clinical risk algorithm misused cost as a proxy for need, leading to under-referral of Black patients, and highlighted the dangers of unexamined proxy labels. Taken together, these cases emphasise the need for systematic fairness testing that moves beyond static audits and single-dataset evaluations.

1.5.4 Empirical Failures Motivating Fairness Evaluation

ProPublica's COMPAS analysis (3) revealed racial disparities in recidivism predictions, motivating the adoption of Equal Opportunity EO and Equalized Odds metrics EOD. Similarly, (26) demonstrated that a widely used clinical risk algorithm misinterpreted *cost* as a proxy for *need*, under-referring Black patients and exposing the dangers of proxy labels. Taken together, these cases underscore the need for systematic fairness testing that goes beyond static audits and isolated evaluations.

1.5.5 Testing and Adequacy in ML

Addressing such failures requires revisiting what "testing" means in the context of ML. Myers originally defined testing in classical software as covering the paths most likely to reveal bugs. Extending this idea, (31) applied property-based testing to ML and recommended that non-functional requirements, such as fairness and robustness, be explicitly tested. From a broader sociotechnical perspective, (17) argued that adequacy must be understood in terms of real-world effects rather than only laboratory metrics. These insights provide the conceptual bridge between fairness failures and

the need for adequacy-style testing in ML.

1.5.6 How This Work Differs

Against this background, the present work differs from existing audits and toolkits that primarily ask, "Is this model biased?". Robustness research has shown that mutation testing can reveal hidden accuracy bugs. Here, the same principle is extended to fairness by introducing a quantified notion of *fairness test adequacy*: given realistic, fairness-relevant mutations, does the existing test suite consistently uncover unfair treatment?

More concretely, this thesis proposes:

- A validity check that certifies that the applied mutations are realistic and capable of inducing plausible fairness fluctuations in the training data.
- A bootstrap methodology that sets detection thresholds (mean + 1.5 standard deviations) to separate genuine fairness signals from random noise.
- Coverage and detection-power analysis using multiple test sets (balanced, skewed, and group-dropped) to assess how well fairness problems are detected.

By adopting this approach, fairness becomes a specific testing requirement with measurable detection power and audit-ready evidence, rather than a vague or optional goal.

To connect these ideas to a concrete context, we now consider the deployed system at the centre of the case study and ask what went wrong in evaluation terms. Filings and expert commentary point to at least one major gap that rigorous fairness testing would have addressed: (i) insufficient *slice coverage* across protected and intersectional groups during evaluation.

(ii) Weak *shift sensitivity* to small cohorts and feature sparsity patterns by age or race.

(iii) Missing *audit artifacts* that tie fairness requirements to concrete tests and thresholds.

Had these been in place, the vendor and customers could have shown evidence that group-level disparities would be detected pre-deployment.

Policy is converging on documented bias controls. The EU AI Act imposes lifecycle obligations for high-risk systems, including risk management, data governance, logging, and post-market monitoring (14). In U.S. credit, the Consumer Financial Protection Bureau requires specific adverse-action reasons even when AI is used (10; 9). Organizations that cannot produce test artifacts face enforcement risk and slower adoption.

1.5.7 Compliance and Legal Implications Beyond One Case

The need for rigorous fairness testing is not limited to a single system or lawsuit. Policy discussions and early regulatory texts increasingly emphasise bias controls and auditability in AI systems. For example, initiatives such as the EU AI Act and guidance from the U.S. Consumer Financial Protection Bureau suggest that fairness testing could become part of compliance obligations in high-risk domains (14; 10).

Although this thesis does not propose legal rules, it demonstrates how fairness adequacy can be operationalised in practice. In future work, the framework presented here could be extended to support compliance settings by producing audit-ready artefacts that align with emerging legal and policy requirements.

How Our Approach Addresses These Gaps

This thesis treats fairness as a question of test adequacy, not only metric reporting. In particular, we:

1. Build explicit **slice coverage** (age, race, gender, and intersections) into the test suite.

2. Generate **realistic mutations** (cohort reweighting, proxy-feature stress, sparse résumé sections) to simulate shifts that may induce fairness faults.

3. Enforce a **mutant-validity gate** (distribution similarity plus training-set fairness shift) to avoid meaningless mutants.

4. Apply a **data-driven detection threshold** using the base model's bootstrap null (mean + 1.5 std; see Section 3.4.2) and declare a mutant *killed* when the fairness change exceeds that threshold.

5. Summarise detection power with a **mutation score** and confidence intervals, producing *audit-ready artefacts* (test suites, operator configurations, thresholds, logs).

In a résumé-screening pipeline, for example, our mutations could (a) up- and down-weight applicant segments common among older candidates, (b) jitter or mask proxy features for age or race (such as graduation year or tenure patterns), and (c) simulate employer filters. If the suite fails to kill a sufficient fraction of these fairness-affecting mutants (that is, if the mutation score falls below a target), the build would be considered inadequate and a remediation report would identify missing slices, fragile features, and data-collection needs.

Industry and National Relevance

Hiring and employment. Active litigation is redefining responsibilities for vendors and deployers. Maintaining test artefacts can mitigate legal and reputational risk (24).

Healthcare. A widely used clinical risk model under-referred Black patients because cost was used as a proxy for need; our operators explicitly perturb proxy labels and cohort mix to reveal such failures before rollout (26).

Finance. When AI supports credit decisions, creditors must provide specific adverse-action reasons; our artefacts aim to tie slice-level findings to compliant explanations (10; 9).

Regulation. The EU AI Act codifies measurable, documented bias controls for high-risk domains; our approach produces an evidence trail that can support such requirements (14).

Bottom Line

Fairness failures increasingly trigger legal and regulatory responses, not just academic debate. A mutation-based adequacy protocol turns fairness from an abstract aspiration into an *operational gate* with measurable detection power and auditable proof, so that unfair outcomes can be identified and addressed before they are deployed at scale.

1.6 Related Work

Fairness in machine learning is an interdisciplinary topic spanning computer science, law, and policy. Significant progress has been made in defining fairness measures and developing auditing tools, but the systematic quantification of fairness test adequacy

remains relatively unexplored. The rest of this chapter has outlined the key strands of work that motivate this thesis: formal definitions and trade-offs between fairness metrics, mutation and metamorphic testing for ML, empirical failures in deployed systems, and emerging compliance expectations. The following chapters build on this foundation by specifying a fairness-aware mutation testing framework and evaluating its adequacy on benchmark datasets.

Chapter 2

Background

2.1 Introduction to Machine Learning

Machine learning (ML) has evolved from a research curiosity into a widely deployed technology that influences decisions in finance, healthcare, hiring, and criminal justice. In many of these settings, algorithmic predictions are treated not merely as suggestions but as decisions that directly affect people's lives. This chapter situates our work within the broader ML landscape and motivates the need for *fairness adequacy* in supervised classification systems.

2.2 Machine Learning in High-Stakes Decisions

Early ML applications were mostly limited to low-risk tasks such as spam filtering and handwriting recognition. Today, however, ML models are increasingly used in high-stakes domains, including:

- **Finance:** Credit scoring algorithms influence loan approvals and interest rates.
- **Healthcare:** Predictive models support diagnoses, treatment prioritisation, and resource allocation.
- **Hiring:** Automated résumé screening affects who is shortlisted and who is

filtered out.

- **Criminal justice:** Risk assessment tools guide parole decisions and sentencing recommendations.

These applications raise ethical questions about fairness and accountability (4). Fairness auditing often relies on group fairness metrics such as Statistical Parity Difference (Statistical parity difference (SPD)), Equal Opportunity Difference, and Equalized Odds Metric (EOM). However, real-world environments are not static. Input distributions shift, thresholds are tuned, and feature definitions evolve over time. Empirical studies show that both predictive performance and fairness metrics can degrade under even modest distribution shifts (29).

In addition, broad demographic categories can obscure intersectional harms. For example, focusing only on "male vs. female" or "Black vs. white" may miss disparities affecting older Black women or younger immigrant men. These observations motivate the notion of **fairness adequacy**: assessing whether fairness remains within acceptable bounds under realistic perturbations, rather than only at a single, fixed snapshot (20).

2.3 Formal Machine-Learning Pipeline

An ML algorithm aims to approximate a function

$$f : \mathcal{X} \to \mathcal{Y}$$

from experience

$$\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n}.$$

Historical bias embedded in $\mathcal{D}$ can propagate into f, systematically disadvan-

taging protected groups. This thesis focuses on binary classification in supervised learning, because fairness metrics are most developed and most legally scrutinised in this setting.

Figure 2.1: Formal machine-learning pipeline: Data collection → Preprocessing → Model training → Evaluation → Deployment.

2.4 Base Learners in Our Experimental Campaign

Our empirical evaluation uses three widely applied algorithms: Logistic Regression (LR), Decision Trees (CART), and k-Nearest Neighbours (K-NN). For each model family, we briefly review inductive bias, computational complexity, interpretability, and fairness-relevant failure modes.

2.4.1 Logistic Regression LR

Logistic Regression is one of the most commonly used linear classifiers in statistics and machine learning. Originally introduced as a model for binary outcomes, it has become a workhorse in domains such as credit scoring, medical diagnosis, and risk assessment, largely because of its efficiency and interpretability.

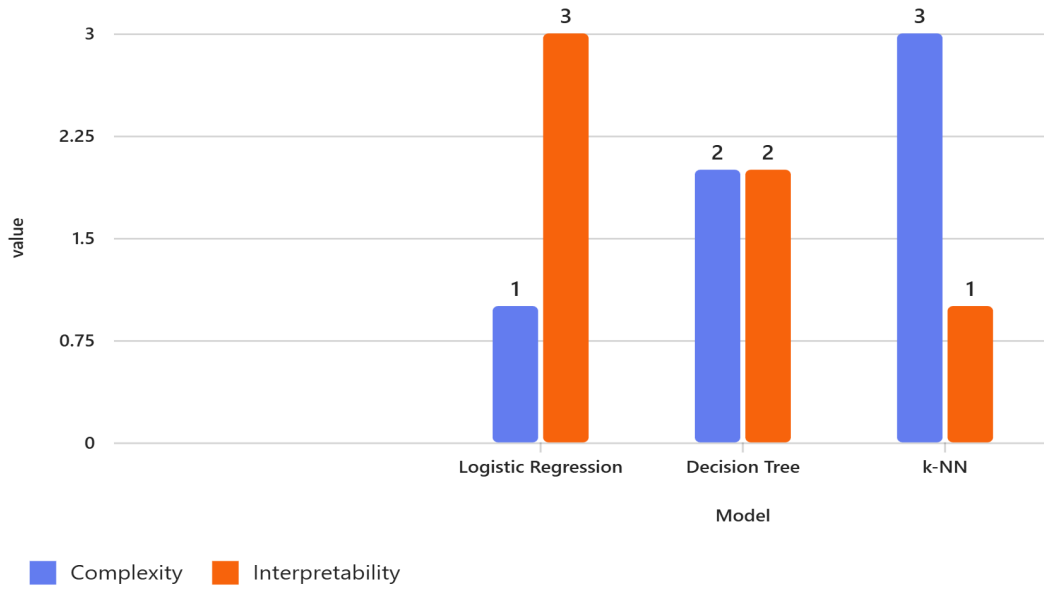

Figure 2.2: Complexity versus interpretability for Logistic Regression, Decision Tree, and k-NN.

Unlike linear regression, which predicts continuous values, logistic regression models the probability of a binary outcome using the logistic (sigmoid) function:

$$\hat{p}(y = 1 \mid x) = \sigma(w^\top x + b), \quad \sigma(z) = \frac{1}{1 + e^{-z}}.$$

Here, w is the vector of feature weights, b is the bias term, and $\sigma(\cdot)$ maps real values to the interval $(0, 1)$. The decision boundary is the hyperplane $w^\top x + b = 0$, which separates the feature space into predicted positive and negative regions.

Training. Model parameters (w, b) are typically estimated by maximising the likelihood of the observed data, or equivalently by minimising the negative log-likelihood. To prevent overfitting and encourage sparsity or smoothness, a regularisation term is added:

$$\mathcal{L}(w) = -\sum_{i=1}^{n} \Big[y_i \log \hat{p}_i + (1 - y_i) \log(1 - \hat{p}_i) \Big] + \lambda \|w\|_p,$$

where $\|w\|_p$ is either the ℓ_1 (Lasso) or ℓ_2 (Ridge) penalty. Optimisation is typically

performed using Newton's method, coordinate descent, or gradient-based methods.

Complexity. Training complexity is roughly $O(nd^2)$ with second-order methods or $O(ndT)$ with gradient descent, where n is the number of samples, d the number of features, and T the number of iterations. Prediction is efficient at $O(d)$ per query, and the memory footprint is $O(d)$.

Interpretability. A key advantage of logistic regression is its transparency: each coefficient w_j can be interpreted as the change in log-odds of the positive outcome per unit increase in feature x_j, holding other features constant. This interpretability makes LR particularly attractive in regulated domains where explanations are required.

Fairness sensitivity. The global linear decision boundary also makes logistic regression sensitive to fairness issues. Any non-zero weight on a protected attribute or a strong proxy can systematically tilt the boundary against a group. For example, if x_2 is correlated with gender and $w_2 > 0$, then many female applicants may receive lower predicted probabilities across the entire feature space. This can lead to violations of SPD or opportunity-based metrics that cannot be resolved by threshold tuning alone.

To make this idea more concrete, consider a setting in which the majority group cluster lies mostly on the accepted side of the boundary, while a minority group cluster is shifted towards the rejected side. Even well-qualified individuals from the minority group may be classified as negative because the single linear boundary has been influenced by group-level correlations.

Hyper-parameters. In practice, behaviour is shaped by a small set of hyper-parameters, including the penalty type (ℓ_1 vs. ℓ_2), inverse regularisation strength $C \in \{0.01, 0.1, 1, 10\}$, class-weight balancing, and, in some setups, fairness-regularised variants. These choices govern the trade-off between accuracy, sparsity, and fairness.

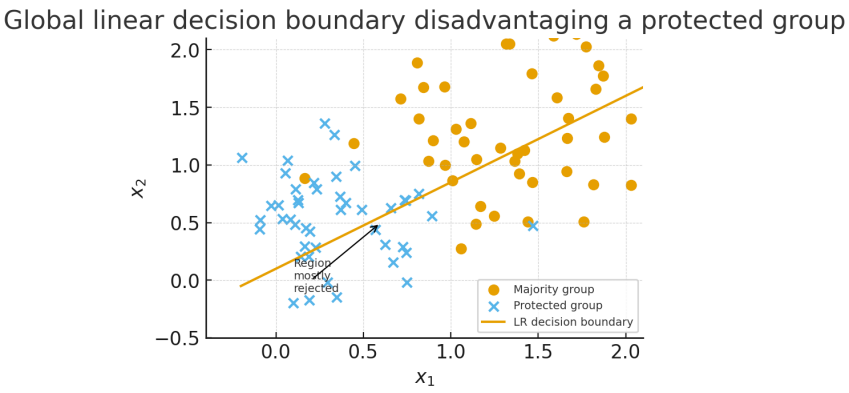

Figure 2.3: Illustration of a global linear decision boundary learned by Logistic Regression. The majority group (circles) lies mostly on the accepted side of the boundary, while many points from a protected group (crosses) fall into a predominantly rejected region.

Mutant kill signature. In our mutation analysis, logistic regression is particularly sensitive to covariate-shift mutants that rotate the linear separator relative to minority directions. Because the model uses a single global boundary, small distributional changes can disproportionately affect protected groups, leading to relatively high fairness mutation kill rates when tests are adequate.

2.4.2 Decision Tree CART

After considering a globally linear model, we now turn to a non-linear alternative: decision trees. These models partition the feature space into axis-aligned regions through recursive splits on individual features. Originating from the CART framework, decision trees are widely used in credit scoring, medical triage, and risk assessment. Their appeal comes from their ability to capture complex feature interactions and to express decisions as human-readable rules.

Formally, a decision tree represents a piecewise-constant function $f(x)$, where each input x is routed from the root node to a leaf via a sequence of binary tests of the form $x_j \leq \tau$. Each leaf ℓ stores an empirical label distribution $\hat{p}_\ell(y)$, and predictions

are obtained by

$$\hat{y}(x) = \arg\max_{y} \hat{p}_\ell(y), \quad \ell = \text{leaf reached by } x.$$

- **Training.** Trees are grown top-down in a greedy fashion. At each node t with sample set S_t, the algorithm searches over candidate features j and thresholds τ to maximise impurity reduction:

$$\Delta I(j, \tau) = I(S_t) - \frac{|S_L|}{|S_t|} I(S_L) - \frac{|S_R|}{|S_t|} I(S_R),$$

  where S_L and S_R are the left and right subsets induced by the split, and $I(\cdot)$ is an impurity measure such as Gini

$$I_{\text{Gini}}(S) = \sum_k p_k(1 - p_k)$$

  or entropy

$$I_{\text{ent}}(S) = -\sum_k p_k \log p_k.$$

  Growth stops when a maximum depth is reached, the node is pure, or too few samples remain, possibly followed by pruning.

- **Complexity.** With n samples, d features, and maximum depth D, a typical implementation that sorts feature values per node has training cost on the order of $O(ndD)$ (often approximated as $O(nd \log n)$ for balanced trees). Prediction for a new instance follows a single path from root to leaf, giving $O(D)$ time and a memory footprint proportional to the number of nodes.

- **Interpretability.** Each root-to-leaf path corresponds to a simple rule (e.g., "if income $\leq$ 30k and age $>$ 40 then reject"). These rules can be visualised and

discussed with domain experts, giving decision trees high local interpretability. Very deep trees, however, may be harder to interpret globally.

- **Fairness sensitivity.** Fairness issues arise when splits depend on protected attributes (e.g., gender, race) or strong proxies (e.g., postcode). A single split on such a feature can route most members of a protected group into a branch with systematically lower predicted scores, leading to large SPD or opportunity-based disparities. Because splits are hard thresholds, individuals just on different sides of a boundary can receive very different outcomes.

Hyper-parameters. Key hyper-parameters include maximum depth D, minimum samples per split and per leaf, the impurity criterion (Gini vs. entropy), and class weights. Fairness-aware variants may restrict certain features from being used in splits or add penalties for splits that increase group-level disparity.

Mutant kill signature. In our mutant analysis, decision trees are most sensitive to mutants that shift samples across split thresholds or alter impurity near the top of the tree. Small perturbations in feature values or labels around these thresholds can flip branches of instances from one leaf to another, changing outcomes for protected groups and exposing the fragility of the learned partition.

2.4.3 k-Nearest Neighbours K-NN

k-Nearest Neighbours (K-NN) is a non-parametric method whose inductive bias is based on locality (11). Unlike parametric models, K-NN does not assume a fixed functional form. Instead, it classifies a new point based on the labels of nearby training points in feature space.

- **Training.** K-NN is a "lazy learner": there is no separate training phase beyond storing the dataset. Given a query point x_*, distances to all training points are

computed, and the k nearest neighbours are selected. In classification, the predicted label $\hat{y}_*$ is obtained by majority vote among those neighbours; in regression, the prediction is the average of their target values.

- **Distance metrics.** The choice of distance function is crucial. Common metrics include Euclidean and Manhattan distance, as well as cosine similarity. In weighted variants, closer neighbours receive higher weights. Feature scaling and normalisation are often necessary so that features with larger ranges do not dominate distance computations.

- **Complexity.** Training complexity is $O(1)$ (storing the dataset). For brute-force search, query complexity is $O(nd)$, where n is the number of samples and d the number of features. In low dimensions, data structures such as KD-trees or Ball Trees can reduce the average query time to $O(\log n)$, but performance degrades in high-dimensional spaces due to the curse of dimensionality.

- **Interpretability.** K-NN has intuitive local interpretability: predictions can be explained in terms of "similar past cases." However, in high-dimensional feature spaces, distances may lose meaning and the neighbourhood structure may be harder to interpret.

- **Fairness sensitivity.** Because predictions are based on local majority voting, K-NN can amplify clustered bias. If instances from a minority group are surrounded by majority-group points, their labels may be repeatedly outvoted, leading to systematic disparities. This is especially problematic in regions of the feature space where protected groups are underrepresented.

- **Hyper-parameters.** Important hyper-parameters include the number of neighbours $k \in \{3, 5, 7, 11, 21\}$, the weighting scheme (uniform vs. distance-weighted),

the distance metric, and preprocessing choices (normalisation, dimensionality reduction). Fairness-aware variants can re-weight neighbours or introduce debiasing mechanisms (13).

- **Mutant kill signature.** K-NN is highly sensitive to feature-noise mutants because distances are directly altered by perturbations. Small changes in feature values can change neighbourhood composition, flipping predictions and fairness outcomes. This makes K-NN an informative stress test for fairness audits under local distributional shifts.

The capacity ladder across our models is roughly: Logistic Regression (low), Decision Tree (medium), and k-NN (high). As model flexibility increases, the importance of rigorous mutation-based adequacy assessment also grows.

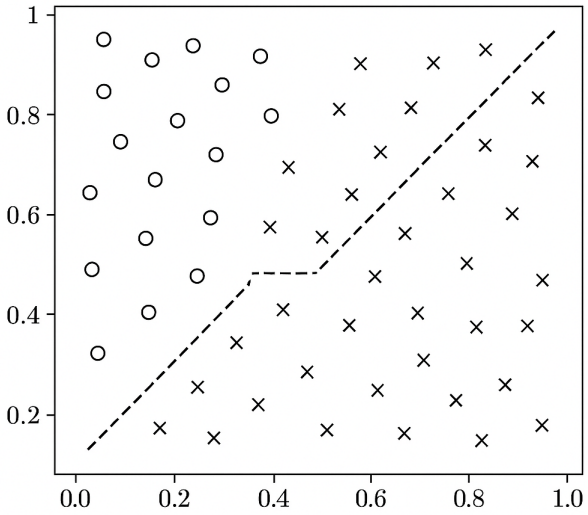

Figure 2.1: Decision boundary induced by *K-N*. Local neighborhoods determine classification, which can magnify clustered bias.

Figure 2.4: Decision boundary induced by K-NN. Local neighbourhoods determine classification, which can magnify clustered bias.

k-NN Prediction Algorithm

For completeness, Algorithm 1 summarises the standard K-NN prediction procedure.

Algorithm 1: K-NN prediction

Input : Training set $\mathcal{D}$, query x_*, integer k, distance $\mathrm{dist}(\cdot,\cdot)$

Output: Predicted label $\hat{y}_*$

$N_k \leftarrow \arg\min_{\mathcal{S}\subseteq\mathcal{D},|\mathcal{S}|=k} \sum_{(x,y)\in\mathcal{S}} \mathrm{dist}(x, x_*)$;

return `majority-vote`$\{y : (x, y) \in N_k\}$;

Table 2.1: Big-O summary (n samples, d features).

	Training	Query	Memory
k-NN	$O(1)$	$O(nd)$	$O(nd)$
Decision Tree	$O(nd \log n)$	$O(\log n)$	$O(\#\text{nodes})$
Logistic Regression	$O(nd^2)$	$O(d)$	$O(d)$

2.5 Mutation Testing for Fairness

Mutation testing, adapted from software engineering, evaluates test adequacy by introducing controlled changes (mutants) and checking whether the test suite detects them. In our setting, the "program" is the trained ML model and its decision pipeline, while the "test suite" consists of fairness metrics applied to model outputs. Mutations correspond to controlled perturbations of data, features, labels, or thresholds that simulate realistic distributional shifts. A fairness audit is considered adequate if it reliably detects fairness-relevant changes introduced by these mutants.

Traditional fairness evaluation computes metrics such as SPD, EOD, and EOM on a single static test set. However, real deployments are dynamic: input distributions drift, operating thresholds are tuned, and features evolve. A fairness audit that appears acceptable on one snapshot may fail under modest perturbations. Mutation

testing provides a principled way to stress-test fairness audits before deployment, helping ensure that fairness violations are not silently overlooked (2).

2.5.1 Core Concepts

At a high level, our fairness-oriented mutation testing framework consists of three components:

- **Mutants:** Perturbations to data, features, labels, or thresholds that simulate realistic shifts.
- **Fairness oracle:** Metrics such as SPD, EOD, and EOM applied to model outputs on mutated data.
- **Kill rule:** A mutant is *killed* if fairness metrics exceed bootstrap-derived thresholds; otherwise it is considered *alive*.

This framework extends fairness evaluation beyond static conditions, probing the robustness of fairness audits under plausible changes.

2.5.2 Kill/Alive Decision Rule

We now formalise the kill/alive rule. For a fairness metric $f \in \{SPD, EOD, EOM\}$, we compare the metric under mutation to the baseline:

$$\Delta f_m = f_m - f_{\text{base}}.$$

Using $B = 1000$ bootstrap resamples of the test set, we estimate the null distribution $(\mu_{\text{null}}, \sigma_{\text{null}})$ under no change. A mutant m is **killed** if

$$|\Delta f_m| \;\geq\; \mu_{\text{null}} + 1.5\,\sigma_{\text{null}} \;=\; \theta_f,$$

and is otherwise **alive**. Killed mutants indicate that the fairness audit successfully detected the perturbation; alive mutants suggest potential inadequacy.

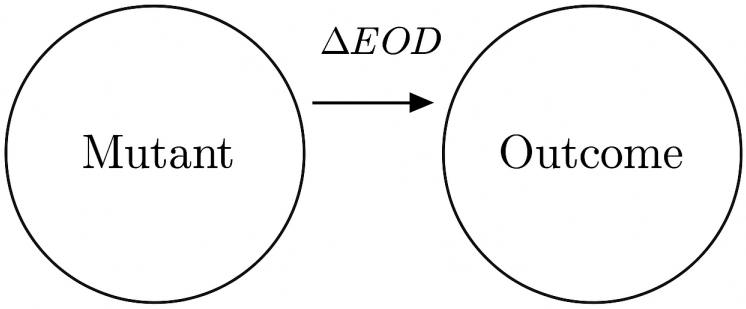

Figure 2.2: Fairness mutation lifecycle. Killed: $|\Delta EOD| \geq \theta$. Alive: below threshold (possibly equivalent).

Figure 2.5: Fairness mutation lifecycle. Mutants are generated, evaluated by fairness metrics, and classified as *killed* or *alive* depending on whether the change exceeds the threshold θ_f.

As a simple illustrative example, suppose the blue group has TPR = 0.80 and the red group has TPR = 0.62. The resulting opportunity gap is 0.18. If the bootstrap threshold for the relevant fairness metric is $\theta = 0.10$, then the audit detects this change and the mutant is *killed*. If the difference were only 0.05, the mutant would survive, signalling that the audit may miss subtle but still meaningful disparities.

By systematically applying mutation testing to fairness audits, we obtain a measure of *fairness adequacy*. This moves beyond single-snapshot evaluation and assesses whether fairness metrics remain responsive under realistic perturbations. A fairness audit with high mutation kill rates can be considered adequate; an audit with many surviving mutants is inadequate, even if static metrics look acceptable.

2.6 Summary of Background Needs

This chapter has established:

1. The societal and technical motivation for fairness and adequacy in ML.
2. The ways in which common models (Logistic Regression, Decision Trees, k-NN) are vulnerable to fairness faults.
3. The rationale for using mutation testing as a principled adequacy measure for fairness.

The next chapter translates these ingredients into the Fairness Adequacy Testing (FAT) framework and describes the mutation operators and guardrails that implement the kill rule.

Chapter 3

Method: Fairness-Oriented Mutation Testing

3.1 Overview

This chapter presents a fairness-oriented mutation testing method designed to assess whether a test suite is *adequate* to detect fairness problems in supervised classification models. Traditional mutation testing focuses on correctness under code or data perturbations; here, the central objects are *fairness metrics*.

The main idea is simple. We introduce small but realistic changes (*mutants*) to the training data or pipeline, retrain the same model specification, and then measure how much group fairness metrics move on a fixed test set. If the change in fairness exceeds a data-driven threshold, the mutant is **KILLED**; otherwise, it is **ALIVE**. Aggregating kill decisions yields a *Fairness Mutation Kill Rate* (mutation kill rates (MKR)), which we interpret as a fairness-oriented test adequacy score. Uncertainty in MKR is reported via bootstrap confidence intervals.

The method is intended for high-stakes classification problems in dynamic domains, where input distributions, thresholds, and feature definitions may shift over time. Although the empirical study focuses on the Adult and COMPAS datasets, the procedure is model-agnostic and can be applied to other tasks.

3.2 Data, Models, and Protected Attributes

Let $\mathcal{D} = \{(x_i, y_i, a_i)\}_{i=1}^{n}$ denote labelled instances with features x_i, binary labels $y_i \in \{0, 1\}$, and a protected attribute a_i (binary or multi-class; for example, gender, race, or client segment). The dataset is split into training and test sets. A baseline model M_{base} is trained on the training split and evaluated on a held-out test set $\mathcal{S} \subset \mathcal{D}$ that remains fixed for all mutants.

Predictions on the test set are either hard labels $\hat{y}_i = M(x_i) \in \{0, 1\}$ or calibrated scores $\hat{p}_i \in [0, 1]$ thresholded at some decision level τ. Group fairness metrics are always computed on $\mathcal{S}$ and stratified by the protected attribute A.

Throughout this thesis we evaluate the three base learners introduced in Chapter 2—Logistic Regression LR, Decision Trees CART, and k-Nearest Neighbours K-NN—and monitor the fairness metrics SPD, EOD, and EOM. The same model templates, hyperparameter grids, and test sets are re-used for all mutants so that differences can be attributed to the mutations themselves.

For a given configuration, let M_{eff} denote the number of *non-equivalent* mutants, that is, mutants that are capable of changing fairness on the test set after passing the validity checks described in Section 3.8. The main adequacy statistic is the Fairness Mutation Kill Rate MKR,

$$\text{MSR} = \frac{\#\,\text{Killed Mutants}}{\#\,\text{Total Mutants} - \#\,\text{Non-Equivalent Mutants}} = \frac{\#\,\text{KILLED}}{M_{\text{eff}}},$$

Higher MKR values indicate that the test suite is more sensitive to fairness-relevant changes and is therefore more adequate from a fairness perspective.

3.3 Methodology Workflow

The fairness mutation procedure follows an iterative workflow that combines group fairness metrics with mutation testing. The steps are summarised in Figure 3.1.

1. **Select dataset and protected attributes.** Choose a classification dataset that contains at least one relevant protected attribute (for example, gender, race, age, or socioeconomic status). This dataset provides the basis for both model training and fairness evaluation.

2. **Generate candidate mutants.** Apply mutation operators to the training data or pipeline to create candidate mutants. These operators include instance and feature removal, reweighting, feature addition, and pipeline variations. The goal is to simulate realistic changes that could affect group fairness. The full operator catalogue is given in Section 3.7.

3. **Train models on original and mutated data.** Train the baseline model M_{base} on the original training data and, for each valid mutant, retrain the same model specification on the mutated training data. Each mutant model M_m is evaluated on the fixed test set $\mathcal{S}$.

4. **Compute fairness metrics.** For the baseline and each mutant model, compute group fairness metrics such as Statistical Parity Difference, Equal Opportunity Difference, and Equalized Odds Metric. These metrics provide a quantitative measure of how outcomes differ across protected groups (Section 3.5).

5. **Decide whether the mutant is KILLED or ALIVE.** For each mutant, compare changes in fairness metrics against bootstrap-based thresholds derived from the baseline model (Section 3.4.2). If any monitored fairness metric

moves outside its control limit, the mutant is labelled **KILLED**; otherwise it is **ALIVE**. This implements the kill rule described in Section 3.4.1.

6. **Update adequacy estimate.** After discarding unrealistic or fairness-neutral mutants via the validity gate (Section 3.8), aggregate kill decisions into the MKR statistic. When MKR is low, the test suite is missing fairness-relevant shifts; when MKR is high, the test suite is more adequate for fairness.

7. **Iterate if needed.** If MKR remains unsatisfactory, extend the test suite (for example, by adding more slices, metrics, or targeted mutants) and repeat the process until a chosen adequacy level is reached.

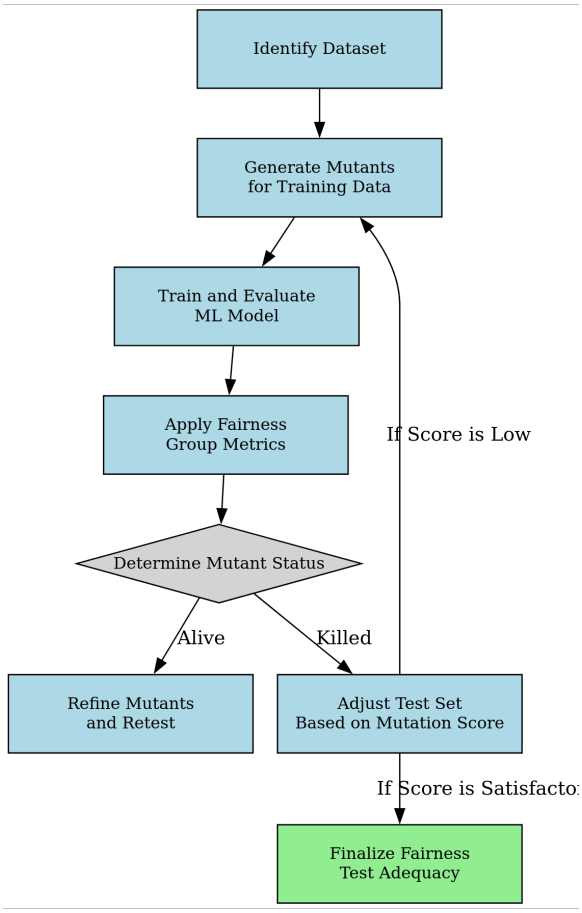

Figure 3.1: Iterative fairness test adequacy process for machine learning models.

3.4 Fairness-Oriented Mutation Testing Framework

This section formalises the proposed fairness-oriented mutation testing framework. The core elements are:

- a family of **mutation operators** that induce plausible changes in training data or pipeline configuration;
- a **fairness oracle** that evaluates group fairness metrics on a fixed test set;
- a **kill rule** that classifies mutants as KILLED or ALIVE based on bootstrap thresholds for fairness deltas; and
- an **adequacy score** (MKR) that aggregates kill decisions across non-equivalent mutants.

The framework is model-agnostic: it applies in the same way to K-NN, Decision Trees, and Logistic Regression, provided that the same model specification is retrained on each mutated dataset.

3.4.1 Mutation Lifecycle and Kill / Alive Decision

Figure 3.2 depicts the lifecycle of a fairness mutant. The procedure begins with a base model trained on clean data, proceeds through mutant creation and retraining, and ends with a fairness-based decision on whether the mutant is killed.

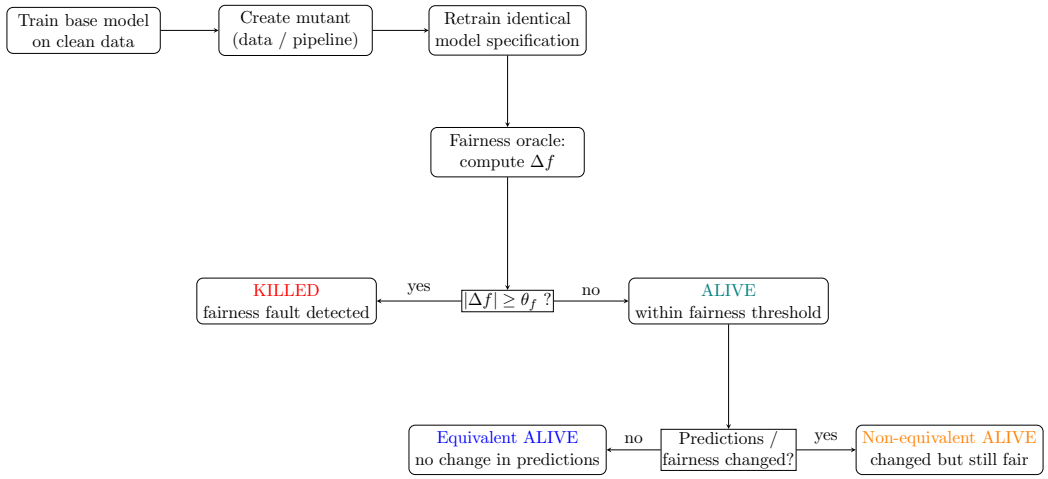

Figure 3.2: Fairness mutation lifecycle with post-hoc classification of **ALIVE** mutants into equivalent (no prediction change) and non-equivalent (prediction/fairness changes remain within the accepted threshold).

Concretely, for each dataset and base learner, the lifecycle is:

1. Train a **base model** on the original training set.
2. Compute baseline fairness metrics on the fixed test set $\mathcal{S}$.
3. Estimate per-metric control limits θ_f using bootstrap (Section 3.4.2).
4. Generate candidate mutants using the operators in Section 3.7.
5. Filter mutants through the **validity gate** (Section 3.8); discard unrealistic or fairness-neutral mutants.
6. Retrain the same model specification on each valid mutant and compute the fairness deltas Δf_m on $\mathcal{S}$.
7. Label each mutant as **KILLED** or **ALIVE** according to whether any fairness delta exceeds its control limit, and aggregate these decisions into MKR (Section 3.9).

3.4.2 Bootstrap Threshold Estimation

The kill decision should not react to minor fluctuations due to sampling noise. For each fairness metric f, we therefore estimate a *kill threshold* θ_f using a bootstrap-based null distribution under the base model.

Let f_{base} denote the value of metric f on the base model and test set $\mathcal{S}$.

1. Keep the base model M_{base} fixed.
2. Draw B bootstrap resamples of the test set, $\{\mathcal{S}^{(b)}\}_{b=1}^{B}$, by sampling with replacement.
3. For each resample, compute $f_{\text{base}}^{(b)}$ and define $\Delta f^{(b)} = f_{\text{base}}^{(b)} - f_{\text{base}}$.

4. Let μ_{null} and σ_{null} be the mean and standard deviation of the bootstrap deltas $\{\Delta f^{(b)}\}_{b=1}^{B}$.

The control limit for metric f is then

$$\boxed{\theta_f = \mu_{\text{null}} + 1.5\,\sigma_{\text{null}}.}$$

The multiplier 1.5 is chosen as a pragmatic balance between sensitivity (detecting meaningful fairness changes) and specificity (ignoring small fluctuations). In the experiments we use $B = 1000$ bootstrap rounds by default.

A mutant m is labelled **KILLED** if

$$|\Delta f_m| \geq \theta_f$$

for *any* monitored fairness metric or subgroup slice; otherwise it is **ALIVE**. Here $\Delta f_m = f_m - f_{\text{base}}$ is the change in fairness metric f between the mutant and the base model.

3.5 Fairness Metrics and Headline Score

3.5.1 Group Fairness Metrics

We track three standard group fairness criteria for a binary decision $\hat{Y}$, protected attribute A, and groups a and b. These metrics were introduced conceptually in Chapter 2; they are restated here in the notation used by the mutation framework.

1. **Statistical Parity Difference SPD.** This metric measures differences in positive outcome rates across groups:

$$\text{SPD} = P(\hat{Y} = 1 \mid A = a) - P(\hat{Y} = 1 \mid A = b).$$

The ideal value is 0, meaning equal positive prediction rates.

2. **Equal Opportunity Difference EOD.** Equal opportunity requires that qualified individuals receive the positive outcome at the same rate, regardless of group membership:

$$\mathrm{EOD} = \mathrm{TPR}_{\mathrm{privileged}} - \mathrm{TPR}_{\mathrm{unprivileged}},$$

where $\mathrm{TPR} = P(\hat{Y} = 1 \mid Y = 1, A)$. The ideal value is again 0.

3. **Equalized Odds Metric EOM.** Equalized odds requires equality of both true positive rates and false positive rates:

$$\mathrm{TPR}_{\mathrm{privileged}} = \mathrm{TPR}_{\mathrm{unprivileged}}, \quad \mathrm{FPR}_{\mathrm{privileged}} = \mathrm{FPR}_{\mathrm{unprivileged}}.$$

We operationalise this via

$$\mathrm{EOM} = \max\bigl\{|\mathrm{TPR}_a - \mathrm{TPR}_b|, |\mathrm{FPR}_a - \mathrm{FPR}_b|\bigr\},$$

with ideal value 0.

As an illustrative example, suppose the blue group has $\mathrm{TPR} = 0.80$ and the red group has $\mathrm{TPR} = 0.62$. Then

$$\mathrm{EOD} = 0.80 - 0.62 = 0.18.$$

If the bootstrap threshold is $\theta_{\mathrm{EOD}} = 0.10$, the base model already violates the EOD tolerance. In this setting, any mutant that *further widens* the TPR gap beyond 0.10 is treated as a fairness fault and is therefore **KILLED** by the EOD oracle.

3.5.2 Headline Fairness Score

To summarise multiple fairness criteria in a single index, we use a Weighted Multi-Objective Fairness Index (WMFI) in $[0, 1]$, where higher values indicate better fairness.

For each sensitive attribute we compute:

- Demographic Parity gap $\text{DPgap} = \max_g PR_g - \min_g PR_g$,
- Equal Opportunity gap $\text{EOgap} = \max_g TPR_g - \min_g TPR_g$,
- Equalized Odds gap $\text{EODgap} = \max\big(\text{EOgap}, \max_g FPR_g - \min_g FPR_g\big)$,
- Disparate Impact $\text{DI} = \min_g PR_g / \max_g PR_g$,

where PR_g denotes the positive prediction rate in group g.

The WMFI is then defined as

$$\text{WMFI} = 1 - \big(w_{DP}\,\text{DPgap} + w_{EO}\,\text{EOgap} + w_{EOD}\,\text{EODgap}\big) - \lambda \cdot \mathbf{1}[\text{DI} < \tau],$$

with weights $(w_{DP}, w_{EO}, w_{EOD}) = [0.30, 0.40, 0.30]$, disparate impact threshold $\tau = 0.80$, and penalty $\lambda = 0.05$. WMFI is used as a headline score in the empirical results; mutation testing, however, is always performed on the component metrics themselves.

3.6 Mutation Design

Mutations can be applied at several levels of the learning process. We consider three broad categories:

Data-level: controlled perturbations to inputs, such as feature noise, time-window shifts, synthetic outliers, group reweighting, or removal of specific instances.

Pipeline-level: changes to preprocessing and sampling procedures, such as swapping normalisation schemes, altering imputation strategies, or modifying resampling settings.

Model-level: limited modifications to model training, such as adjusting regularisation strength, decision thresholds, or class weights.

Each mutation operator ϕ transforms the training data and/or training procedure to produce a mutated model M_ϕ. Operators are parameterised to keep mutations plausible and to avoid unrealistic scenarios that would not arise in normal operation.

3.7 Mutation Operators

This section describes the concrete mutation operators used in the Fairness Adequacy Test (FAT) framework. Operators are grouped into four main families: removal, addition, permutation, and shuffling. Together they instantiate the data- and pipeline-level mutations described above.

3.7.1 Removal Operators

Removal operators delete parts of the dataset to stress-test reliance on specific instances, groups, or features.

Instance Removal

Instance removal eliminates selected records, especially from minority or protected groups, to mimic data loss or under-representation.

- **Binary removal.** For a binary sensitive attribute (for example, gender), create mutants by dropping all instances from one group at a time:

  - Mutant 1: remove all cases with gender = female.
  - Mutant 2: remove all cases with gender = male.

- **Multi-value removal.** For a multi-valued attribute (for example, race), create one mutant per value:
  - Mutant 1: remove all cases with race = Black.
  - Mutant 2: remove all cases with race = White.
  - Mutant 3: remove all cases with race = Hispanic.
  - Mutant 4: remove all cases with race = Asian.

- **Combinatorial removal.** For multiple sensitive attributes, remove intersectional combinations:
  - Mutant 1: remove all cases with gender = male and race = Black.
  - Mutant 2: remove all cases with gender = male and race = White.
  - Mutant 3: remove all cases with gender = female and race = Black.
  - Mutant 4: remove all cases with gender = female and race = White.

Feature Removal

Feature removal studies how the model behaves when certain pieces of information are absent.

- **Single feature removal.** Remove one feature entirely, such as the "sex" attribute.

- **Multiple feature removal.** Remove features one at a time to create independent mutants, for example: gender, race, income, or education level.

- **Combinatorial feature removal.** Remove feature pairs such as:
  - gender and race;
  - gender and income;
  - race and education level;
  - income and education level.

3.7.2 Addition Operators

Addition operators create new features synthesised from existing ones.

- **Linear combinations.** Create a new feature as a linear combination, e.g.

$$\text{new\_feature} = a \cdot \text{age} + b \cdot \text{years of education},$$

  with constants a and b.

- **Ratios.** Create ratio-based features such as

$$\text{new\_feature} = \frac{\text{age}}{\text{years of education}}.$$

Instance Addition

For intersectional fairness coverage, new synthetic instances are added by generating combinations of sensitive attribute values (for example, all gender–race combinations) in areas where the original data are sparse.

3.7.3 Permutation Operators

Permutation operators reassign values within the dataset to disrupt patterns on which the model may be relying.

Feature Permutation

Randomly permute the values of a single feature across instances. This breaks the link between that feature and the label, while preserving its marginal distribution.

Row Permutation

Shuffle the order of rows in the training data to remove potential artefacts from batch structure or temporal ordering.

3.7.4 Shuffling Operators

Shuffling operators further test the model's dependence on particular associations.

Feature Shuffling

Randomly reassign values within a feature column, similar to feature permutation, to stress-test feature importance and proxy effects.

Label Shuffling

Shuffle labels among samples, disrupting the association between features and labels. This checks whether the model is learning genuine patterns or simply memorising the training set; such mutants are usually filtered by the validity gate if they become too unrealistic.

3.7.5 Link to Data-, Pipeline-, and Model-Level Mutations

The concrete operators above instantiate the broader mutation categories:

- **Data-level mutations:** instance and feature removal or addition, permutation, shuffling, label flips, feature noise, intersectional resampling, and missing-

ness injection.

- **Pipeline-level mutations:** threshold slides, encoder swaps, scaler swaps, and class-weight changes.
- **Model-level mutations:** small changes to regularisation or removal of fairness-critical features.

These operators are always combined with the validity gate in the next section to maintain realism.

3.8 Validity Gate (Pre-Filter)

Not every syntactically valid mutant is meaningful for fairness analysis. Before a mutant contributes to MKR, it must pass two checks.

1. **Distribution similarity.** We compare the original and mutated training distributions using two-sample tests:

   - Kolmogorov–Smirnov tests for numeric features, and
   - χ^2 tests for categorical features.

   Mutants must satisfy $p \geq 0.05$ (or remain below a chosen effect-size threshold) for all monitored features to be considered within a realistic shift regime.

2. **Training-side fairness shift.** We require that the mutation moves the fairness signal on the training data. Let Δf^{train} be the change in a fairness metric on the training set, and let $(\mu^{\text{train}}_{\text{null}}, \sigma^{\text{train}}_{\text{null}})$ denote its bootstrap null statistics. We require

$$|\Delta f^{\text{train}}| \geq \mu^{\text{train}}_{\text{null}} + \sigma^{\text{train}}_{\text{null}}.$$

Mutants failing either check are treated as *equivalent* in practice and are excluded from the MKR denominator.

3.9 Mutation Score and Uncertainty

Let M denote the number of mutants that pass the validity gate. For each such mutant m, let $\mathbf{1}\{m \text{ is KILLED}\}$ be an indicator that the fairness oracle has detected a fault. The Fairness Mutation Kill Rate is

$$\boxed{\text{MKR} = \frac{1}{M} \sum_{m=1}^{M} \mathbf{1}\{|\Delta f_m| \geq \theta_f \text{ for some } f\},}$$

where the condition "for some f" ranges over all monitored metrics and slices. Mutants rejected by the validity gate do not enter M.

To quantify uncertainty, we estimate a 95% percentile bootstrap confidence interval for MKR by resampling the set of mutants with replacement. We also compute MKR broken down by slice (for example, per sensitive attribute or per operator type) to see which groups or mutation families contribute most to the detections. An adjusted score MKR_{adj} can further exclude mutants with very small training-side fairness changes, producing a more conservative adequacy judgement.

3.10 Worked Mini-Example (Trading Classifier)

To make the procedure more concrete, consider a binary trading signal model that predicts $\hat{y} \in \{0, 1\}$ ("execute trade" or "do not execute"). The protected attribute is client segment $a \in \{\text{retail}, \text{institutional}\}$. We track Equal Opportunity Difference as the main fairness metric.

1. **Bootstrap.** Train the base model M_{base} on a rolling window and compute EOD

on $\mathcal{S}$. Using $B = 1000$ bootstrap resamples of $\mathcal{S}$ under the fixed base model, obtain $(\mu_{\text{EOD}}, \sigma_{\text{EOD}})$ and set $\theta_{\text{EOD}} = \mu_{\text{EOD}} + 1.5\sigma_{\text{EOD}}$.

2. **Mutations.** Generate mutants such as:

   - shifting the input time window by +1 day;
   - adding small Gaussian noise to volatility features;
   - reweighting the minority segment by a factor of 1.2;
   - swapping z-score scaling with robust scaling; and
   - changing missing-value imputation from median to k-NN.

3. **Validity gate.** Reject mutants that produce a large AUC drop (for example, > 0.03) or extreme feature mean shifts (for example, $> 2\sigma$), or that fail the fairness-shift check on the training data.

4. **Assessment.** For each valid mutant, compute ΔEOD on $\mathcal{S}$ and apply the kill rule. Aggregate decisions into MKR with a 95% confidence interval.

This miniature scenario mirrors the structure of the main experiments and shows how often plausible operational changes could materially worsen fairness across client segments.

3.11 Implementation Pseudocode

Algorithm 2 gives high-level pseudocode for the fairness mutation testing pipeline. The same procedure is applied to all datasets and base learners, with shared fairness metrics and thresholds.

Algorithm 2: Fairness Mutation Testing Pipeline

Input : Training data $\mathcal{D}_{\text{train}}$; test set $\mathcal{S}$; model template $\mathcal{M}$; mutant generator $\mathcal{G}$; fairness metrics $\mathcal{F}$; threshold multiplier $z = 1.5$; bootstrap rounds $B = 1000$

Output: Mutation Kill Rate (MKR) and 95% confidence interval

`/* Step 1: Bootstrap null distribution for fairness metrics */`

Train base model $M_{\text{base}} \leftarrow \mathcal{M}(\mathcal{D}_{\text{train}})$;

Compute baseline fairness metrics $f_{\text{base}} \leftarrow \text{FairnessOracle}(M_{\text{base}}, \mathcal{S}, \mathcal{F})$;

for $b = 1$ **to** B **do**

  $\mathcal{S}^{(b)} \leftarrow$ bootstrap sample of $\mathcal{S}$;

  $f_{\text{base}}^{(b)} \leftarrow \text{FairnessOracle}(M_{\text{base}}, \mathcal{S}^{(b)}, \mathcal{F})$;

  $\Delta f^{(b)} \leftarrow f_{\text{base}}^{(b)} - f_{\text{base}}$;

end

Compute $\mu_{\text{null}}, \sigma_{\text{null}}$ over $\{\Delta f^{(b)}\}$ for each $f \in \mathcal{F}$;

Set threshold $\theta_f \leftarrow \mu_{\text{null}} + z \cdot \sigma_{\text{null}}$ for each $f \in \mathcal{F}$;

$M \leftarrow 0$; kills $\leftarrow 0$;

/* Step 2: Generate, filter, and score mutants */

for *mutant dataset* $\mathcal{D}_{train}^{(m)}$ *in* $\mathcal{G}(\mathcal{D}_{train})$ **do**

  if *validity gate fails for* $\mathcal{D}_{train}^{(m)}$ **then**

    ; // Discard equivalent or unrealistic mutant

  end

  $M \leftarrow M + 1$;

  $M_m \leftarrow \mathcal{M}(\mathcal{D}_{\text{train}}^{(m)})$;

  $f_m \leftarrow \text{FairnessOracle}(M_m, \mathcal{S}, \mathcal{F})$;

  $\Delta f_m \leftarrow f_m - f_{\text{base}}$;

  if $\exists f \in \mathcal{F}$ *such that* $|\Delta f_m(f)| \geq \theta_f$ **then**

    kills $\leftarrow$ kills +1;

  end

end

Compute MKR $\leftarrow$ kills/M;

Estimate 95% bootstrap confidence interval for MKR by resampling $\{1, \ldots, M\}$ with replacement;

return MKR and its confidence interval;

Algorithm 2 turns the conceptual framework into an operational procedure: it bootstraps fairness metrics to define thresholds, generates and filters mutants, and summarises their impact via MKR.

Research Questions

This chapter operationalises the research questions stated in Chapter 3 as follows:

RQ1. **(RQ1 – Effectiveness of mutation operators)** How effective are fairness-aware mutation operators? The chapter defines a family of operators, a validity

gate, and a kill/alive decision rule, allowing their effectiveness to be quantified via mutation kill rates and metric drift (Sections 3.3–3.7).

RQ2. **(RQ2 – Performance across models and datasets)** How consistently does the framework perform across models and datasets? The same mutation pipeline is applied to different base learners (K-NN, Decision Tree, Logistic Regression) and datasets (Adult and COMPAS) using shared fairness metrics and bootstrap thresholds (Sections 3.5 and 3.9).

RQ3. **(RQ3 – Comparison to existing fairness testing baselines)** How does the proposed framework compare to existing fairness testing baselines? The method is positioned against static group fairness audits and headline fairness scores, providing a mutation-based adequacy perspective that can be compared to toolkits such as Fairlearn, AIF360, and FairTest (Sections 3.4–3.9 and Chapter 4).

In summary, Chapter 3 has set out the methodological foundation of the Fairness Adequacy Testing framework. Having defined the mutation lifecycle, fairness metrics, validity gate, and adequacy thresholds, the next chapter instantiates these elements on the Adult and COMPAS datasets, applies them across the base learners introduced in Chapter 2, and reports the resulting MKR and WMFI values to address the research questions in practice.

Chapter 4

Fairness Metrics and Empirical Evaluation

4.1 Purpose of this Chapter

This chapter has two main jobs. First, it explains the fairness metrics used in this thesis in a clear and practical way. Second, it shows how these metrics behave in our experiments when we apply the Fairness Mutation Testing (FAT) framework to real datasets.

Building on Chapter 3, we now run the framework on two benchmark datasets (Adult and COMPAS) and three base learners (Logistic Regression LR, Decision Tree CART, and k-NN K-NN). We:

- define and motivate the group-fairness metrics (SPD, EOD, EOM),
- introduce a single headline fairness score, the Weighted Multi-Objective Fairness Index (Weighted Multi-Objective Fairness Index (WMFI)),
- walk through a concrete numeric example on the Adult dataset,
- describe the experimental setup for Adult and COMPAS,
- report baseline accuracy and fairness,
- study how fairness metrics react to fairness drift thresholds,

- examine how metrics move under mutation-based stress testing,
- evaluate how effective the fairness-aware mutation operators are across models and datasets, and
- compare mutation-based adequacy with a diameter-based diversity baseline.

Where detailed tables or extended plots were too long for the main text, we provide them in a public repository so that others can inspect and reuse them.

4.2 Metric Definitions and Notation

We focus on a standard binary classification setting:

- The protected attribute $A \in \{a, b\}$ represents group membership (for example, Female vs. Male, or Black vs. White).
- The model output $\hat{Y} \in \{0, 1\}$ is a binary decision (e.g. accept vs. reject, low risk vs. high risk).
- The ground-truth label $Y \in \{0, 1\}$ is the true outcome.

All fairness rates are computed on the same fixed test set $\mathcal{S}$, in line with the mutation-testing protocol described in Chapter 3.

Equal Opportunity vs Equalized Odds

Because two related metrics appear throughout this chapter, we clarify their difference up front:

- **Equal Opportunity** focuses on qualified individuals. It asks: "Among people who truly qualify (e.g. $Y = 1$), do different groups get the positive decision at similar rates?" It compares *true positive rates*.

- **Equalized Odds** is stricter. It looks at both *true positive rates* and *false positive rates*. It asks: "Are both qualified people helped equally, and unqualified people treated similarly, across groups?"

In this thesis we operationalise these ideas as:

- EOM: an Equal Opportunity-style metric that captures disparity in true positive rates (True Positive Rate (TPR)),
- EOD: an Equalized Odds-style metric that captures the largest disparity across both TPR and False Positive Rate (FPR).

This distinction matters when we interpret the graphs. In both Adult and COMPAS, EOD tends to be a stricter test than EOM, because EOD penalises unfairness in both "who we help" and "who we mistakenly harm".

4.2.1 Composite Fairness Index (Scalar Measure)

In practice, it is often useful to have a single number that summarises several fairness checks. For this, we define the Weighted Multi-Objective Fairness Index (WMFI). WMFI takes values between 0 and 1, where 1 means "best" (from a fairness point of view).

Key concepts

- **Demographic Parity Disparity Parity (DP):** A model satisfies demographic parity if the probability of receiving a positive prediction is the same across protected groups.

- **Disparate Impact (DI):** The ratio of positive prediction rates between groups. Under the common "four-fifths rule", ratios below 0.80 are treated as red flags for potential discrimination.

Gap measures

For each sensitive attribute we compute:

- **DPgap:**

$$\text{DPgap} = \max_g PR_g - \min_g PR_g,$$

  where PR_g is the positive prediction rate for group g.

- **EOgap:** the gap between the highest and lowest true positive rates across groups.

- **EODgap:** the maximum between EOgap and the corresponding false positive rate gap.

- **Disparate Impact:**

$$\text{DI} = \frac{\min_g PR_g}{\max_g PR_g}.$$

Here, DI tells us how the least-advantaged group compares to the most-advantaged group in terms of positive predictions.

Illustrative example

Suppose a hiring model yields:

Group	**Positive Prediction Rate (PR_g)**
Group A (Female)	0.60
Group B (Male)	0.45

Then:

- $\max_g PR_g = 0.60$, $\min_g PR_g = 0.45$,
- DPgap $= 0.15$,
- DI $= 0.45/0.60 = 0.75 < 0.80$, so it fails the four-fifths rule.

WMFI Formula and Interpretation

We combine the gap measures into a scalar score:

$$\text{WMFI} = 1 - \big(w_{DP}\,\text{DPgap} + w_{EO}\,\text{EOgap} + w_{EOD}\,\text{EODgap}\big) - \lambda \cdot \mathbf{1}[\text{DI} < \tau],$$

with

$$(w_{DP}, w_{EO}, w_{EOD}) = [0.30,\ 0.40,\ 0.30], \quad \tau = 0.80, \quad \lambda = 0.05.$$

The penalty term $-\lambda \cdot \mathbf{1}[\text{DI} < \tau]$ is triggered when DI falls below the legal-style tolerance threshold $\tau = 0.80$. This reflects the fact that a bad DI ratio is often taken more seriously in regulatory contexts than a small change in a gap.

In short: WMFI gives a single "fairness score" that can be compared across models and datasets, while still being grounded in clear components (DPgap, EOgap, EODgap, and DI).

4.3 Worked Numeric Example on Adult Dataset

To make the metrics more concrete, we walk through a simple example on the Adult test set. We use *sex* as the protected attribute (Female $= a$, Male $= b$), and consider a logistic regression model with a fixed threshold of 0.5.

Positive prediction rates:

Table 4.1: Confusion matrices by sex – Adult test set (example).

	Female (a)		Male (b)	
	$\hat{Y}=1$	$\hat{Y}=0$	$\hat{Y}=1$	$\hat{Y}=0$
$Y=1$	803	547	2582	1094
$Y=0$	423	2227	1015	5289

- Positive rates:

$$P(\hat{Y}=1 \mid A=a) = \frac{803+423}{803+547+423+2227} = 0.322,$$

$$P(\hat{Y}=1 \mid A=b) = \frac{2582+1015}{2582+1094+1015+5289} = 0.311,$$

so

$$\text{SPD} = 0.322 - 0.311 = 0.011.$$

The model has almost no difference in overall selection rate by sex.

- Step 2: True-positive rates:

$$\text{TPR}_a = \frac{803}{803+547} = 0.595, \quad \text{TPR}_b = \frac{2582}{2582+1094} = 0.702,$$

So the Equal Opportunity Difference is

$$\text{EOD} = 0.595 - 0.702 = -0.107.$$

This means qualified males receive positive decisions about 10.7 percentage points more often than qualified females.

- Step 3: False-positive rates:

$$\mathrm{FPR}_a = \frac{423}{423 + 2227} = 0.160, \quad \mathrm{FPR}_b = \frac{1015}{1015 + 5289} = 0.161,$$

  so, False positive rates are very similar across groups

- Step 4: Equalized-Odds-style metric We define:

$$\mathrm{EOM} = \max\{|-0.107|,\ |0.160 - 0.161|\} = 0.107.$$

Take-away. Even though the selection rates (SPD) are almost equal, the true positive rates are not. This is a pattern that we will see again in the main experiments: for both Adult and COMPAS, SPD can look acceptable while EOD and EOM reveal much stronger fairness problems. This motivates the use of a richer fairness test, including mutation-based stress tests.

4.4 Experimental Setup

4.4.1 Datasets and Protected Attributes

We evaluate the framework on two widely used benchmark datasets:

- **Adult:** Contains demographic and employment information used to predict whether an individual's income exceeds \$50K. Protected attributes in our analysis include sex, race, and age group.

- **COMPAS:** Contains criminal history and demographic information used to predict two-year recidivism risk. Protected attributes include race, `age_category`, and sex. We acknowledge well-known concerns about label quality and historical bias in this dataset but treat it as a standard benchmark for fairness studies.

For both datasets, we split data into training and test partitions using a fixed random seed to ensure reproducibility. Where we tune hyper-parameters, we use cross-validation on the training data only. All fairness metrics are computed on the held-out test set.

4.4.2 Pre-processing and Feature Engineering

Pre-processing is kept as simple and consistent as possible across datasets:

- Missing values are imputed using simple strategies (e.g. mode for categorical, mean/median for numerical).
- Categorical variables are encoded (e.g. one-hot encoding).
- Features are scaled or normalised when required by the model class (for example, for K-NN and Logistic Regression).
- In a few cases we construct additional features that also serve as candidates for mutation operators (see Chapter 3).

All transformations are fit on the training data and then applied to the test data, to avoid leakage.

4.4.3 Models and Hyper-parameters

We use three base learners that represent different modelling families:

- Logistic Regression (LR),
- Decision Tree (CART),
- k-Nearest Neighbours (K-NN).

Hyper-parameters are selected using grid search with cross-validation, following the procedure in Chapter 2. The final configurations are kept fixed when running the mutation experiments, so that differences come from mutations rather than re-tuning.

Code and Reproducibility

All code used for data preparation, model training, mutation operators, and figure generation is available in a public GitHub repository:

<YOUR-GITHUB-LINK>

The repository also includes:

- configuration files for the experiments,
- scripts for reproducing the mutation runs, and
- extended tables and graphs that do not appear in this chapter.

This allows readers to inspect the full analysis, run the code on their own machine, and adapt the framework to new datasets.

4.5 Baseline Accuracy and Fairness

Before we stress-test fairness through mutations, we first look at the baseline picture. For each dataset–model pair we compute standard predictive metrics (Accuracy, AUC, F1) and then evaluate fairness via EOD and EOM on the test set.

These baseline fairness values serve as a fixed reference point. Later, when we look at fairness drift caused by mutants, we will compare back to these baseline levels.

For Adult, the two plots together show that:

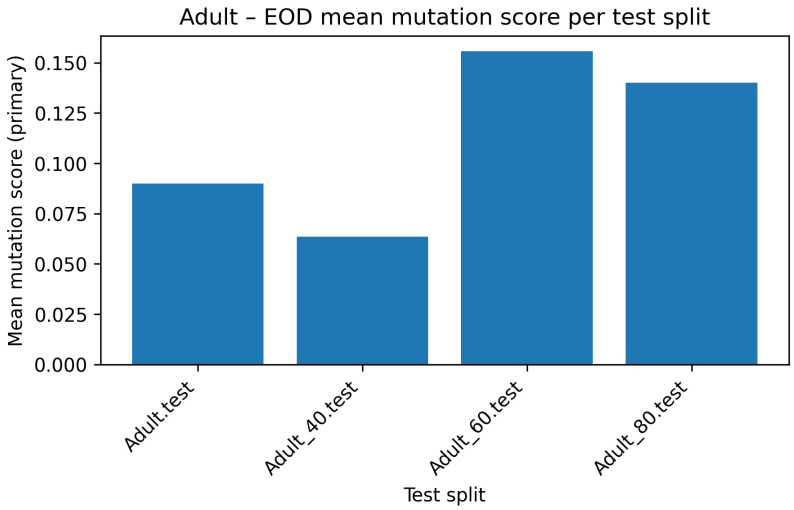

Figure 4.1: Adult dataset: baseline Equalized Odds Difference (EOD) by protected attribute.
X axis: protected attributes / groups.
Y axis: EOD values, summarising how strongly Equalized Odds is violated at baseline.

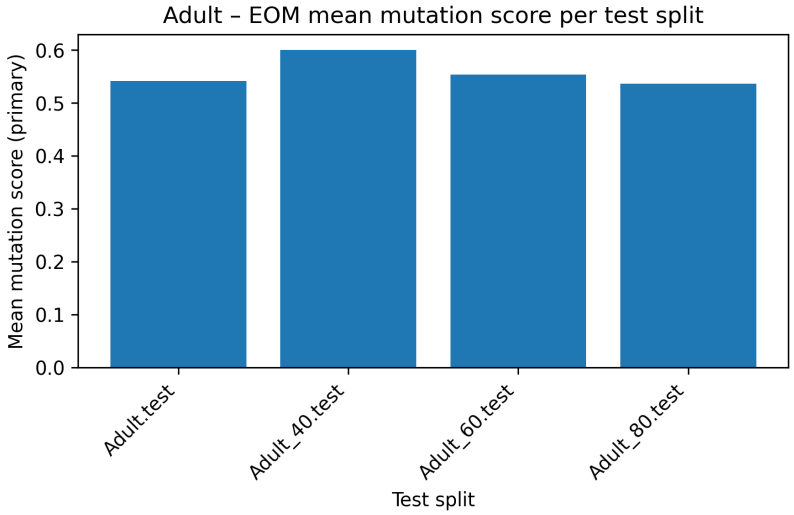

Figure 4.2: Adult dataset: baseline Equal Opportunity Metric (EOM) by protected attribute.
X axis: protected attributes / groups.
Y axis: EOM values, showing disparities in true positive rates at baseline.

- some protected attributes (for example, sex or race) have noticeably higher EOD and EOM values than others,
- EOD is often slightly larger than EOM, since it also counts false positive differences, not just true positives.

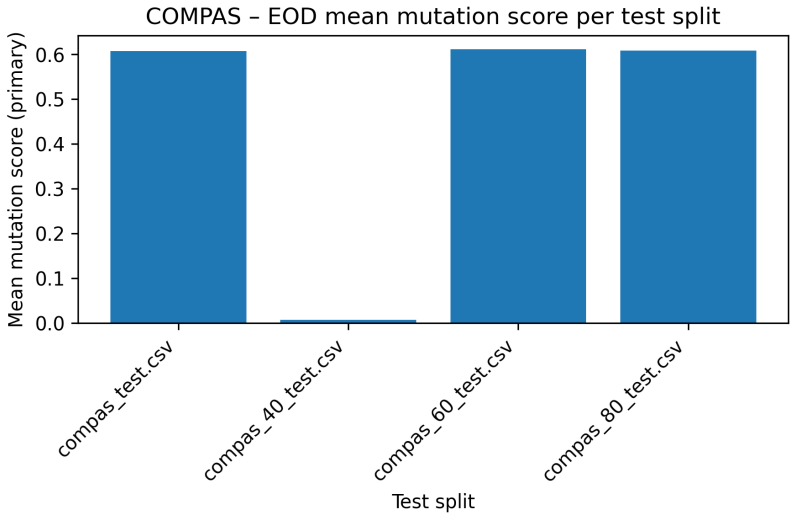

Figure 4.3: COMPAS dataset: baseline Equalized Odds Difference (EOD) by protected attribute.
X axis: protected attributes / groups.
Y axis: EOD values, highlighting where recidivism predictions depart most from Equalized Odds.

On COMPAS, the gaps are typically larger than on Adult, especially under EOD. This reflects two differences:

- the task is higher risk (recidivism), so mistakes can have more impact,
- structural biases in the data lead to stronger violations of both opportunity and error-rate parity.

Comparing Adult and COMPAS side by side, and comparing EOD vs EOM, we see three recurring patterns that will also show up in the mutation graphs:

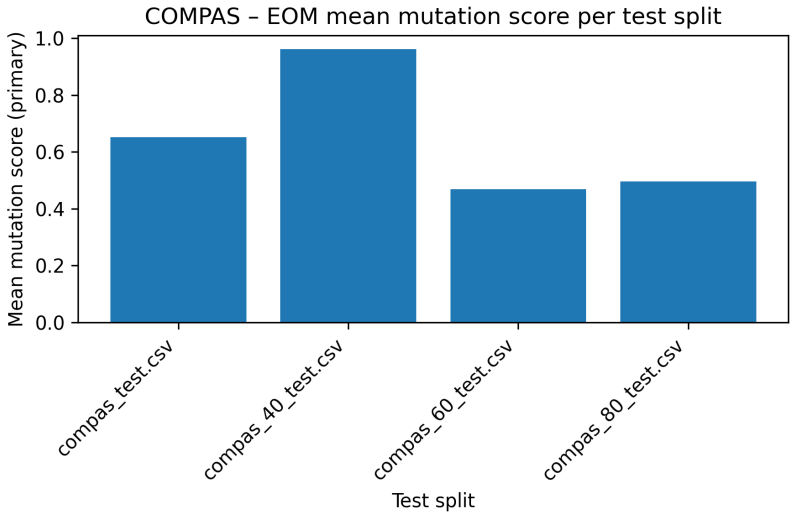

Figure 4.4: COMPAS dataset: baseline Equal Opportunity Metric (EOM) by protected attribute.
X axis: protected attributes / groups.
Y axis: EOM values, showing opportunity gaps at baseline.

1. COMPAS tends to show *larger* fairness gaps than Adult.

2. EOD is usually *stricter* than EOM, because it combines differences in both TPR and FPR.

3. For some protected attributes, the two metrics disagree: a group may look "moderately unfair" under EOM but much worse under EOD.

4.6 Threshold Sensitivity Analysis

In our experiments, the "threshold" we vary is not the classifier's decision threshold. Instead, it is a fairness *drift* threshold that decides when a mutant is considered **KILLED**.

For each metric f (for example, EOD), we look at the distribution of absolute drift values across all mutants and set:

$$\theta_f = \mu_\Delta + 1.5 \cdot \sigma_\Delta,$$

where μ_Δ and σ_Δ are the mean and standard deviation of the drift distribution. A mutant is **KILLED** if its fairness change exceeds this threshold. Otherwise it is **ALIVE**.

Among ALIVE mutants we then distinguish:

- **ALIVE–Equivalent:** fairness drift below 0.05. These mutants change almost nothing and are effectively noise.
- **ALIVE–Non-Equivalent:** fairness drift above 0.05 but below the threshold θ_f. These are fairness-relevant shifts that the current fairness tests do not flag as faults.

This structure matters later when we interpret the graphs: KILLED mutants show where our tests are strong, ALIVE–Non-Equivalent mutants show where they miss interesting fairness changes.

4.7 Metric Profiles Across Models

For each dataset and each base learner (K-NN, CART, LR), we estimate EOD and EOM across multiple random seeds. We then report the mean and a 95% bootstrap confidence interval to show how stable each metric is.

In the graphs (not reproduced here), we group bars by model family and dataset, and show both metrics:

- Models with *higher* EOD or EOM on average are less fair under that metric.
- Narrow confidence intervals indicate more stable fairness behaviour across seeds; wide intervals indicate that fairness is fragile to data or training noise.

Across the figures, two patterns emerge:

1. For both Adult and COMPAS, EOD tends to be a harsher test than EOM, because it adds false positives to the fairness picture.

2. The relative ordering of models can change by dataset and metric. For example, a model that looks "better" on Adult under EOM may not look better on COMPAS under EOD. This is why we later study adequacy by dataset–metric pair, rather than assuming one global ranking.

4.8 Metric Stability Under Mutation Stress

We now use the mutation operators from Chapter 3 to see how fairness metrics behave under realistic perturbations at the data and pipeline levels.

4.8.1 Metric Drift by Mutation Type

For each mutation, we compute:

$$|\Delta\text{EOD}|, \quad |\Delta\text{EOM}|.$$

We summarise the distributions using medians and interquartile ranges (IQRs), and report the proportion of mutants where the drift exceeds the detection threshold θ_f.

Intuitively:

- A large median drift means that metric is sensitive to that type of change.

- A high proportion of "above threshold" cases means that the tests frequently detect fairness issues.

4.8.2 Mutation Status: KILLED vs ALIVE

Using the rule from Chapter 3, each mutant is assigned to one of:

- **KILLED** — fairness drift exceeds the detection threshold.
- **ALIVE–Equivalent** — the mutant changes nothing important in predictions or fairness.
- **ALIVE–Non-Equivalent** — fairness changes appear but stay below the threshold.

The ALIVE–Non-Equivalent mutants are especially important. They show that fairness-relevant changes can slip through the current tests. The mutation scores defined below quantify how often this happens.

Figure 4.5 provides an overview of how well the fairness tests perform on COMPAS by compressing the KILLED/ALIVE breakdown into a single number per setting.

Let:

- M_{primary} be the number of fairness-related mutants,
- K_k be the number of KILLED mutants,
- E_e be the number of equivalent (effectively neutral) mutants.

We define the Fairness Adequacy Testing score as:

$$\text{FAT} \;=\; \frac{K_k}{M_{\text{primary}} - E_e} \in [0, 1].$$

Here:

- FAT = 1 means every non-equivalent mutant was detected.

Data_name	Algorithm_type	Metric	M_primary	Killed_k	Equivalent_e	Mutation_score
compas_test.csv	Logistic Regression	EOD	23	21	0	0.913043478
compas_test.csv	Decision Tree	EOD	22	20	0	0.909090909
compas_test.csv	K-NN	EOD	22	0	0	0
compas_40_test.csv	Logistic Regression	EOD	159	2	0	0.012578616
compas_40_test.csv	Decision Tree	EOD	159	1	0	0.006289308
compas_40_test.csv	K-NN	EOD	159	0	1	0
compas_60_test.csv	Logistic Regression	EOD	171	156	1	0.917647059
compas_60_test.csv	Decision Tree	EOD	171	156	0	0.912280702
compas_60_test.csv	K-NN	EOD	171	1	0	0.005847953
compas_80_test.csv	Logistic Regression	EOD	172	157	0	0.912790698
compas_80_test.csv	Decision Tree	EOD	172	157	0	0.912790698
compas_80_test.csv	K-NN	EOD	172	0	1	0
compas_test.csv	Logistic Regression	EOM	23	22	0	0.956521739
compas_test.csv	Decision Tree	EOM	22	0	6	0
compas_test.csv	K-NN	EOM	22	1	21	1
compas_40_test.csv	Logistic Regression	EOM	159	149	2	0.949044586
compas_40_test.csv	Decision Tree	EOM	159	149	0	0.937106918
compas_40_test.csv	K-NN	EOM	159	149	10	1
compas_60_test.csv	Logistic Regression	EOM	171	7	159	0.583333333
compas_60_test.csv	Decision Tree	EOM	171	4	1	0.023529412
compas_60_test.csv	K-NN	EOM	171	4	166	0.8
compas_80_test.csv	Logistic Regression	EOM	172	7	163	0.777777778
compas_80_test.csv	Decision Tree	EOM	172	7	0	0.040697674
compas_80_test.csv	K-NN	EOM	172	4	166	0.666666667

Figure 4.5: FAT mutation scores on the COMPAS dataset for Equal Opportunity Difference (EOD) and Equalized Odds (EOM) across the four test files (full, 40%, 60%, 80%) and three algorithms (Logistic Regression, Decision Tree, K-NN).

- FAT = 0 means none of the non-equivalent mutants were detected.

We compute FAT separately for EOD and EOM. In the graphs that follow, higher FAT values mean stronger fairness test adequacy.

Across COMPAS:

- FAT scores under EOD are often lower than under EOM, which confirms that Equalized Odds is the harder fairness target.
- The differences between models are visible: some algorithms (for example, a well-regularised Logistic Regression) may show higher FAT, while others (e.g. certain tree settings) may show weaker detection.

These patterns are mirrored, with different levels, in the Adult results.

4.8.3 RQ1: How Effective Are the Mutation Operators?

RQ1 asks whether the fairness-aware mutation operators actually inject meaningful fairness faults, and whether these faults can be detected across datasets and metrics.

We look at this from four angles:

1. a global comparison by dataset and metric,
2. differences between models,
3. robustness under down-sampling (smaller test sets),
4. stability of adequacy patterns across test splits.

Global Effectiveness by Dataset and Metric

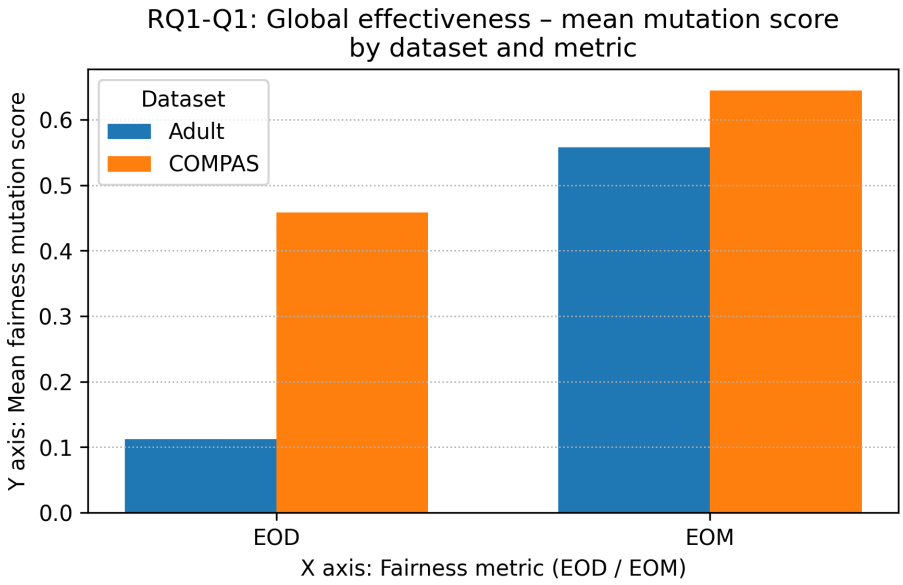

Figure 4.6: Global fairness mutation adequacy by dataset and metric.
X axis: fairness metric (EOD and EOM).
Y axis: mean FAT score (0–1), averaged over all models and test splits.

Figure 4.6 shows that:

- For both datasets, FAT scores are clearly above zero, which means the operators do not produce trivial noise; they do create fairness faults that tests can see.
- EOD usually yields lower FAT than EOM, especially on COMPAS. This reflects that EOD is a stricter metric that is harder to satisfy and easier to "break".

Model-family Sensitivity to Fairness Faults

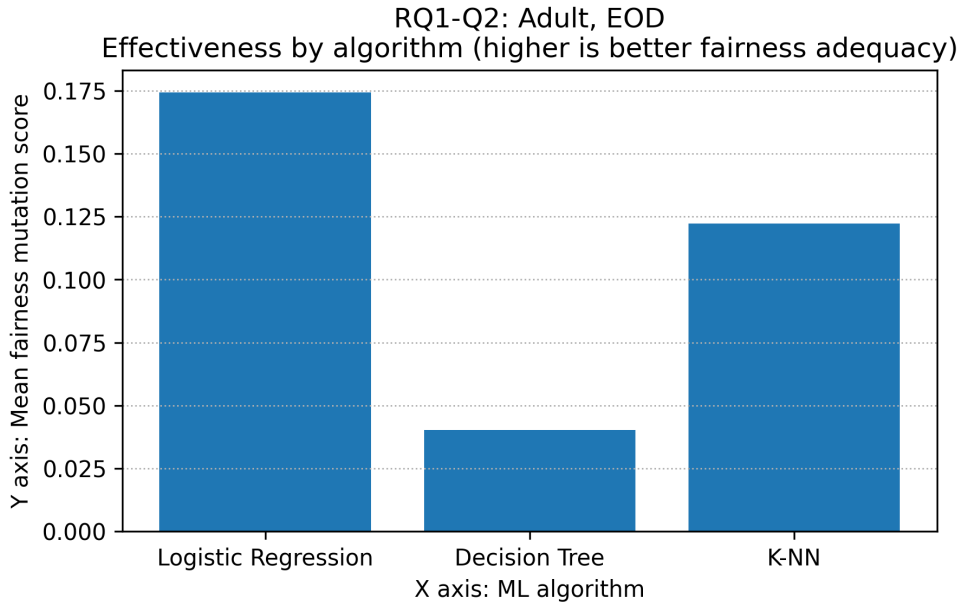

Figure 4.7: Adult dataset: algorithm-level FAT under EOD.
X axis: ML algorithm (LR, CART, K-NN).
Y axis: mean FAT score (0–1).

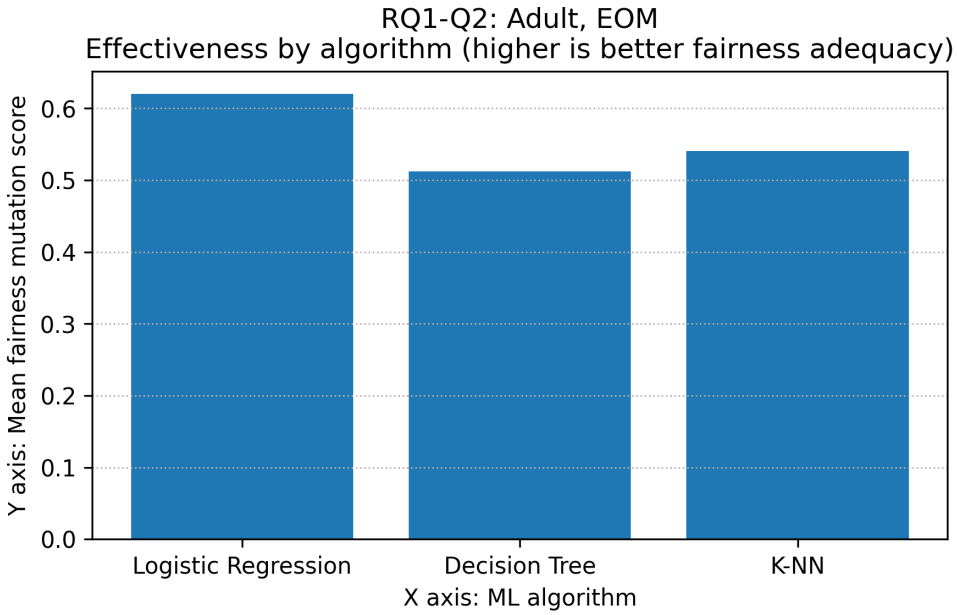

Figure 4.8: Adult dataset: algorithm-level FAT under EOM.
X axis: ML algorithm (LR, CART, K-NN).
Y axis: mean FAT score (0–1).

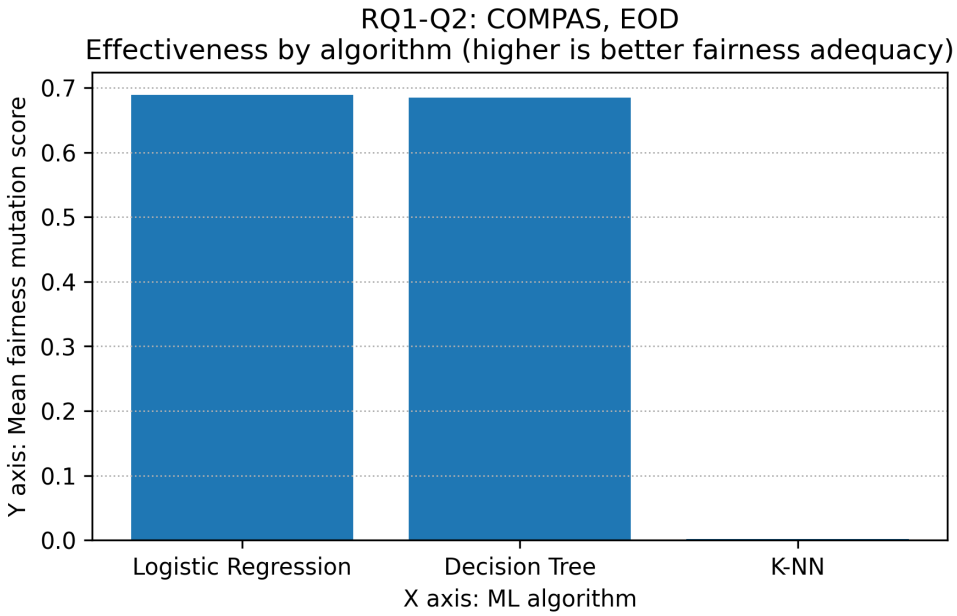

Figure 4.9: COMPAS dataset: algorithm-level FAT under EOD.
X axis: ML algorithm (LR, CART, K-NN).
Y axis: mean FAT score (0–1).

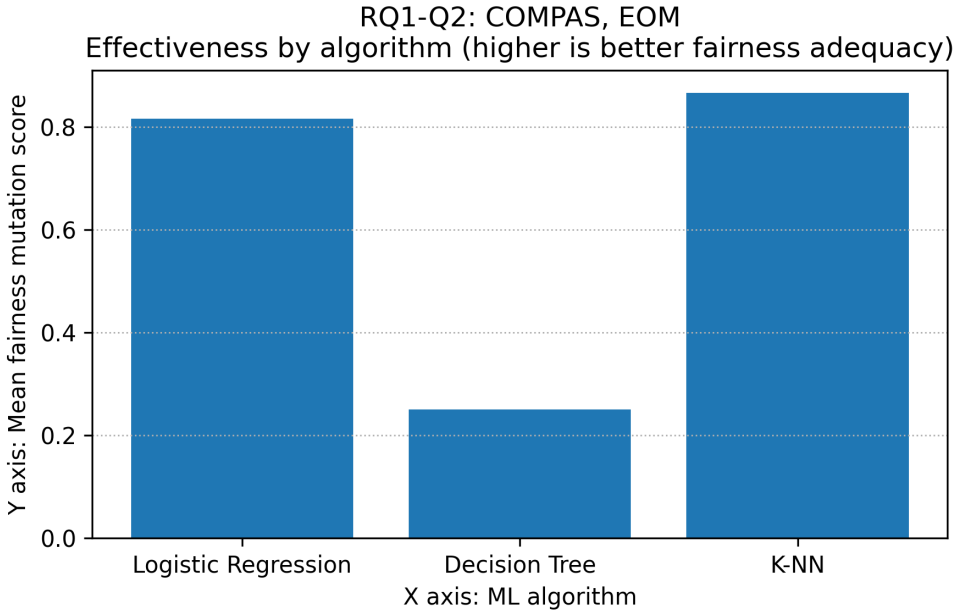

Figure 4.10: COMPAS dataset: algorithm-level FAT under EOM.
X axis: ML algorithm (LR, CART, K-NN).
Y axis: mean FAT score (0–1).

The bar charts show that:

- On Adult, FAT scores under EOM are relatively stable across algorithms, suggesting that opportunity-based fairness faults are fairly easy to surface with these operators.
- On COMPAS, FAT scores vary more by algorithm, especially under EOD. This suggests that the geometry of the learner (linear vs tree vs neighbourhood) interacts with how fairness faults appear in high-stakes recidivism.

Effectiveness Across Test Splits (Down-sampling)

We now see what happens when the test set is made smaller (40%, 60%, 80% subsets).

For both datasets:

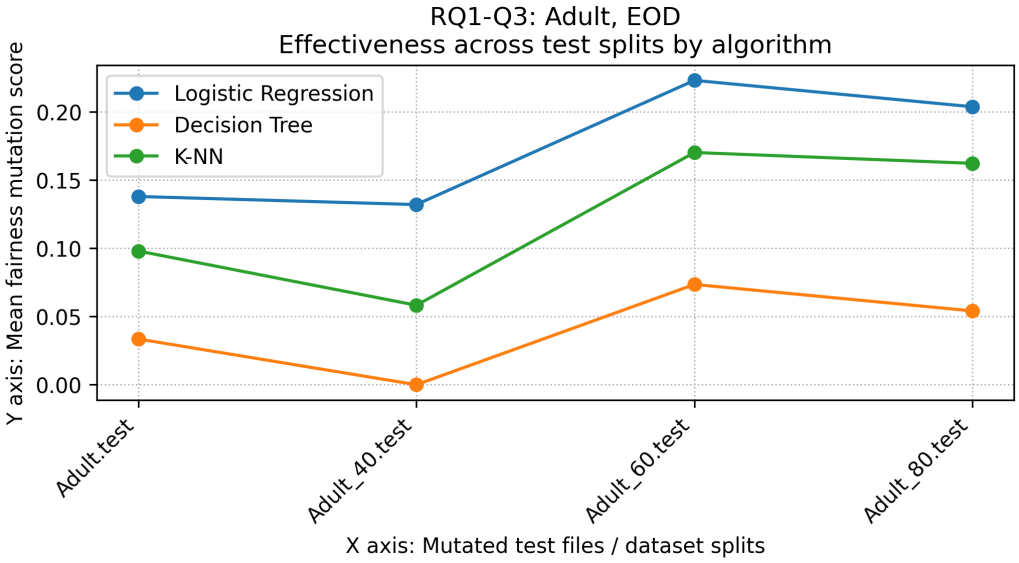

Figure 4.11: Adult dataset: FAT across test splits by algorithm under EOD.
X axis: test splits (Adult.test, Adult_40.test, Adult_60.test, Adult_80.test).
Y axis: mean FAT score, per algorithm.

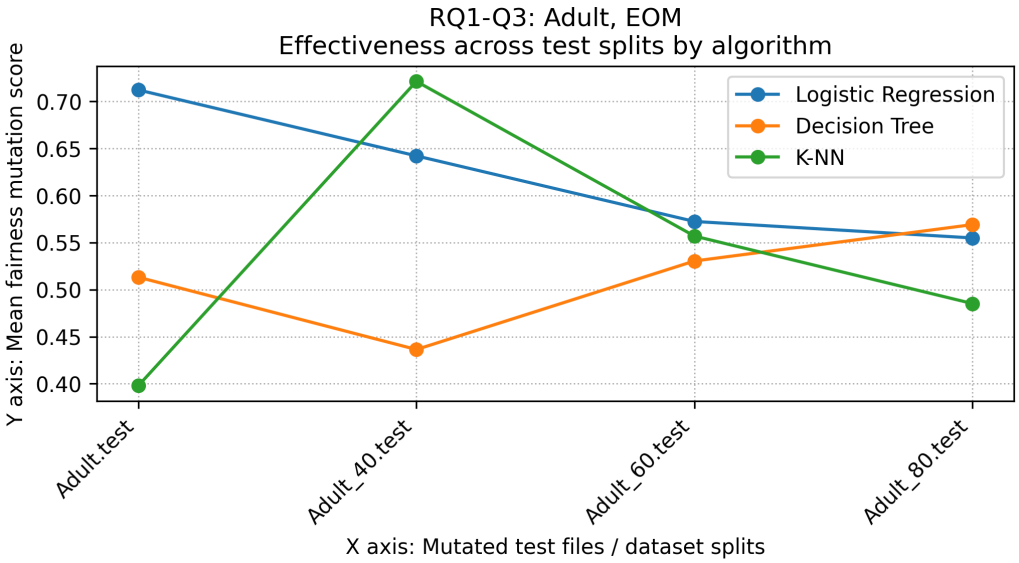

Figure 4.12: Adult dataset: FAT across test splits by algorithm under EOM.
X axis: test splits.
Y axis: mean FAT score, per algorithm.

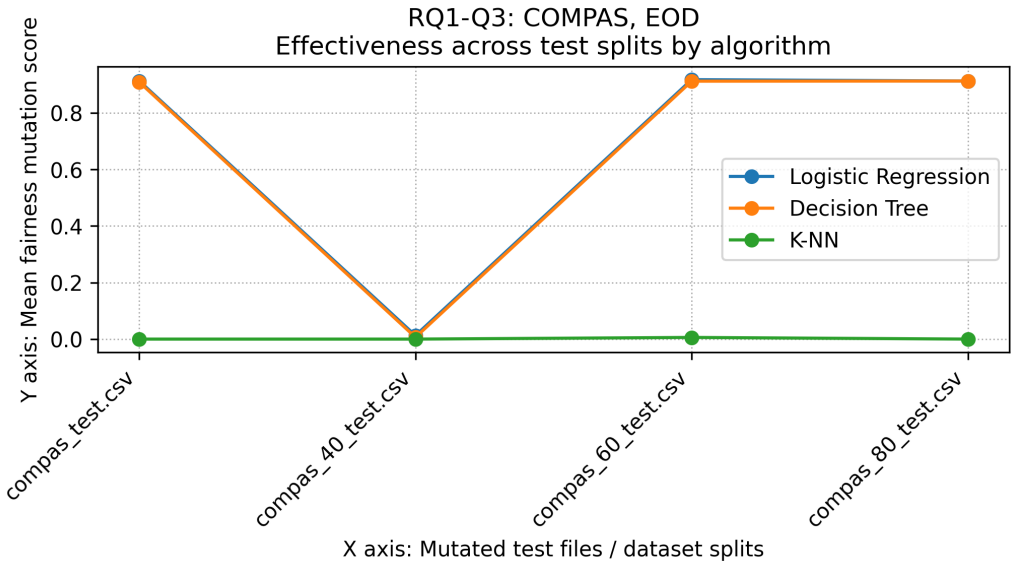

Figure 4.13: COMPAS dataset: FAT across test splits by algorithm under EOD.
X axis: test splits (compas_test.csv, compas_40_test.csv, compas_60_test.csv, compas_80_test.csv).
Y axis: mean FAT score, per algorithm.

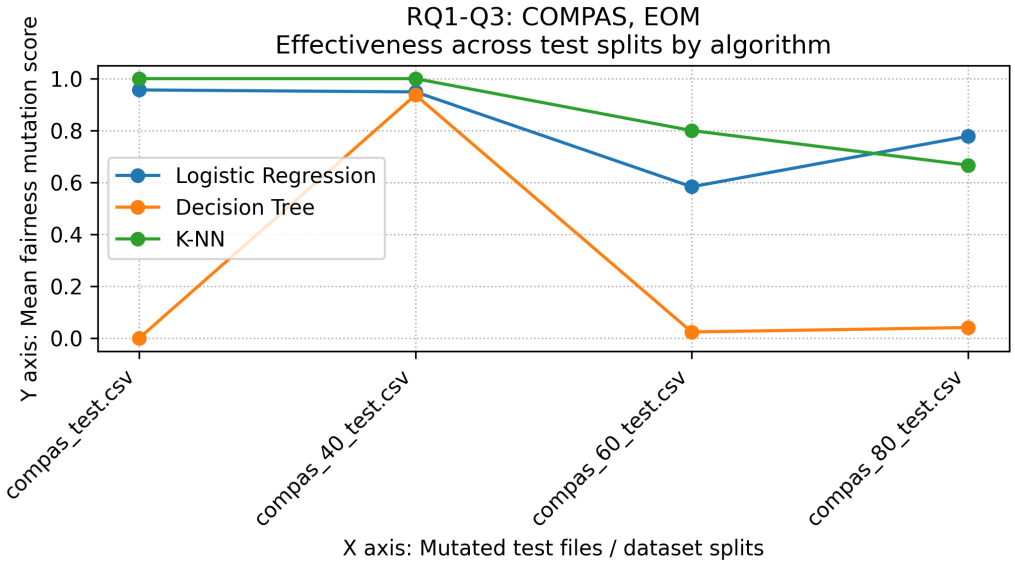

Figure 4.14: COMPAS dataset: FAT across test splits by algorithm under EOM.
X axis: test splits.
Y axis: mean FAT score, per algorithm.

- FAT curves are generally quite flat across splits, especially under EOM for Adult. This means the operators remain useful even when the evaluation slice is smaller.
- When there is a drop, it is more visible in COMPAS and under EOD, which fits the idea that strict fairness checks need more data to be reliable.

Stability of Adequacy Ordering Across Splits

Finally, we check whether the relative ordering of test files is stable.

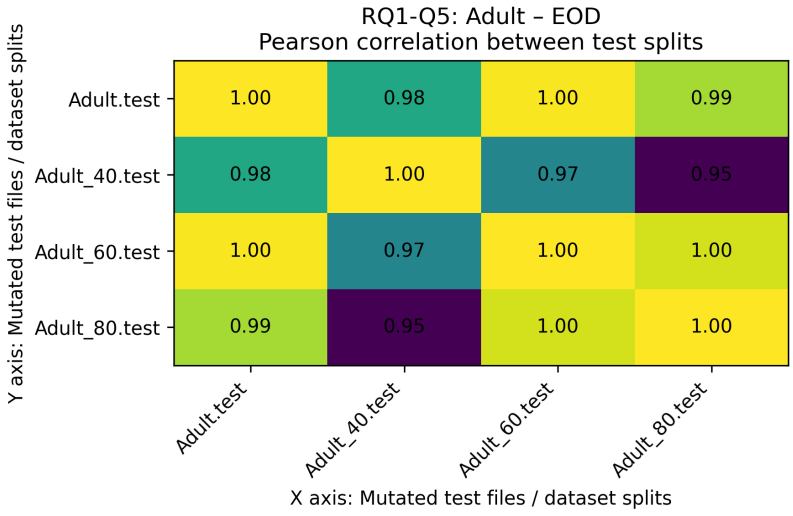

Figure 4.15: Adult dataset: Pearson correlation of EOD FAT scores between test splits.
X/Y axes: test splits.
Cell values: Pearson correlation coefficient.

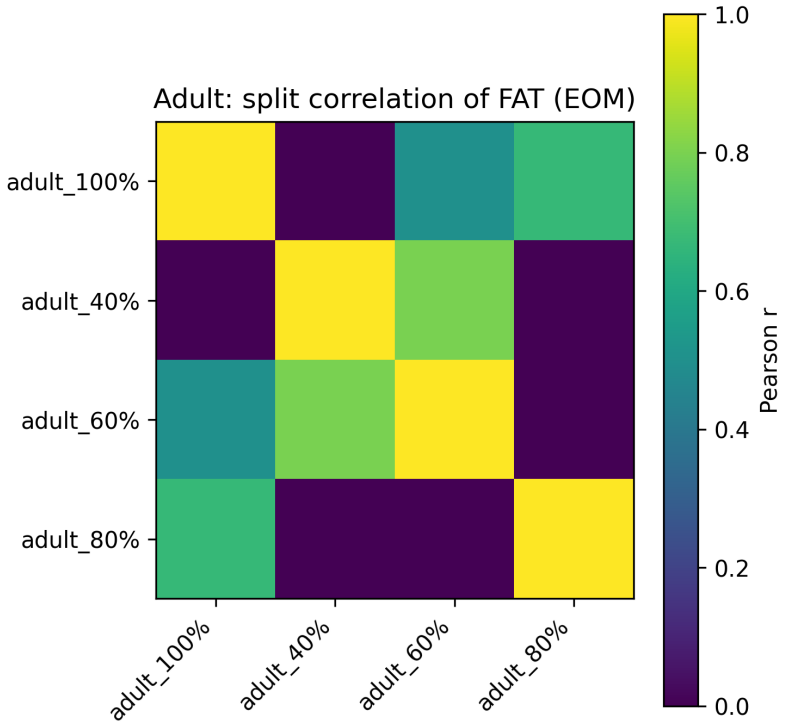

Figure 4.16: Adult dataset: Pearson correlation of EOM FAT scores between test splits.
X/Y axes: test splits.
Cell values: Pearson correlation coefficient.

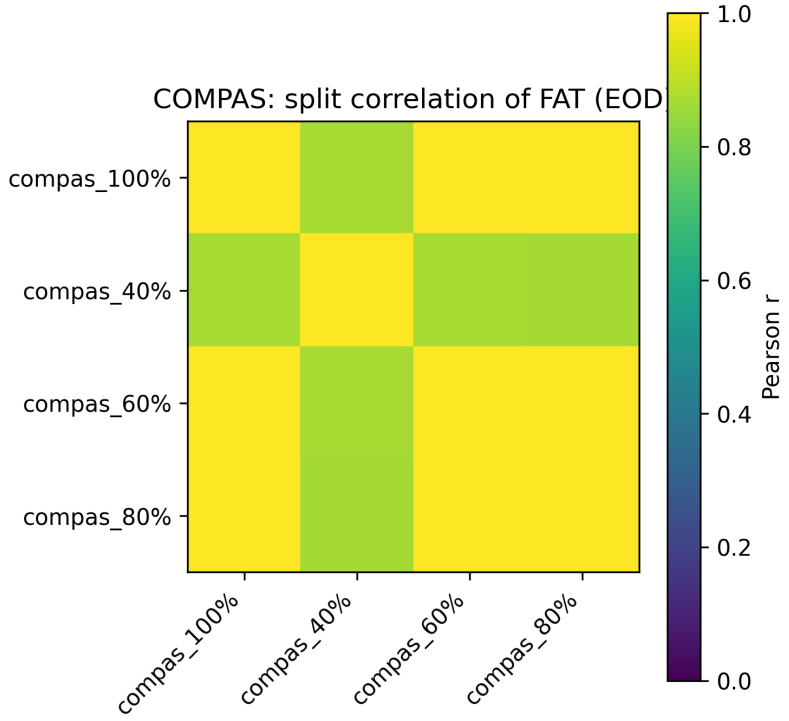

Figure 4.17: COMPAS dataset: Pearson correlation of EOD FAT scores between test splits.
X/Y axes: test splits.
Cell values: Pearson correlation coefficient.

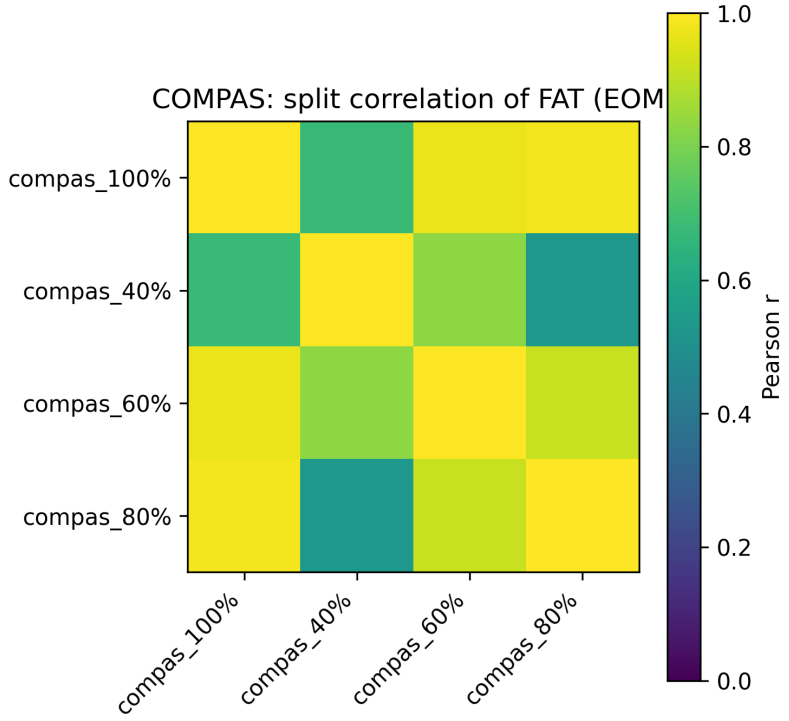

Figure 4.18: COMPAS dataset: Pearson correlation of EOM FAT scores between test splits.
X/Y axes: test splits.
Cell values: Pearson correlation coefficient.

Correlations close to 1 show that the adequacy ranking of splits is stable. For both Adult and COMPAS, and for both EOD and EOM, the heatmaps show generally high correlations. This suggests that mutation-based adequacy is not a fragile artefact of one particular test file, but a consistent signal across slices.

RQ1 summary. Across Adult and COMPAS, the operators create non-trivial fairness faults, and the FAT scores show clear, consistent patterns across datasets, metrics, models and splits. EOD tends to be the stricter and more demanding metric, especially on COMPAS, while EOM shows more stable adequacy on Adult.

4.8.4 RQ2: How Well Does the Framework Perform Across Models and Datasets?

RQ2 shifts the focus from "are the operators useful?" to "is the framework behaving consistently across different model families and datasets?".

We again look at:

- average FAT scores per model,
- detailed heatmaps by model and split,
- correlations of FAT scores across splits.

For Adult, EOM-based FAT scores tend to be reasonably strong and consistent across models. For EOD, the picture can be slightly more varied but still coherent. For COMPAS, the differences between models are more pronounced, especially under EOD. In other words, the framework does not collapse to a single "one size fits all" answer: it reveals how fairness adequacy depends both on dataset and model.

The heatmaps make three relationships clear:

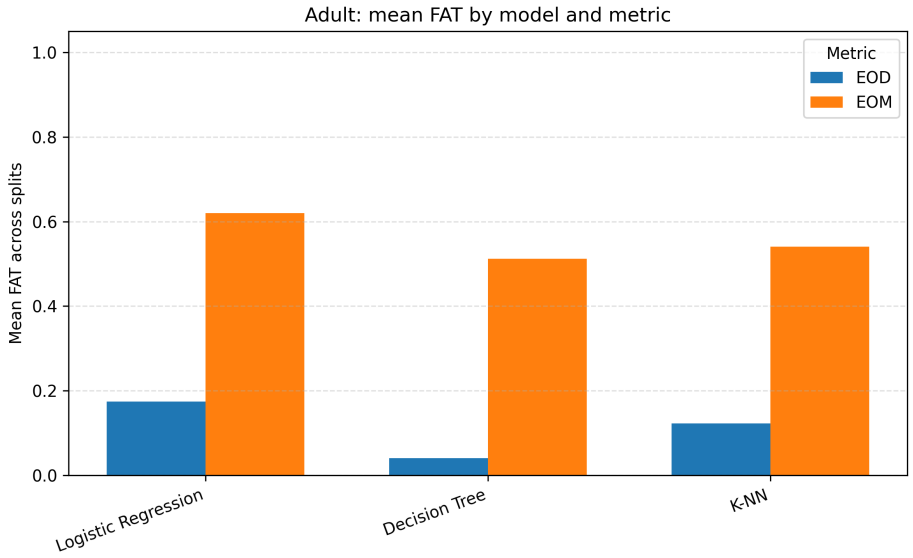

Figure 4.19: Adult dataset: mean FAT scores by model and metric.
X axis: ML algorithm (LR, CART, K-NN).
Y axis: mean FAT over test splits.
Bars compare EOD and EOM.

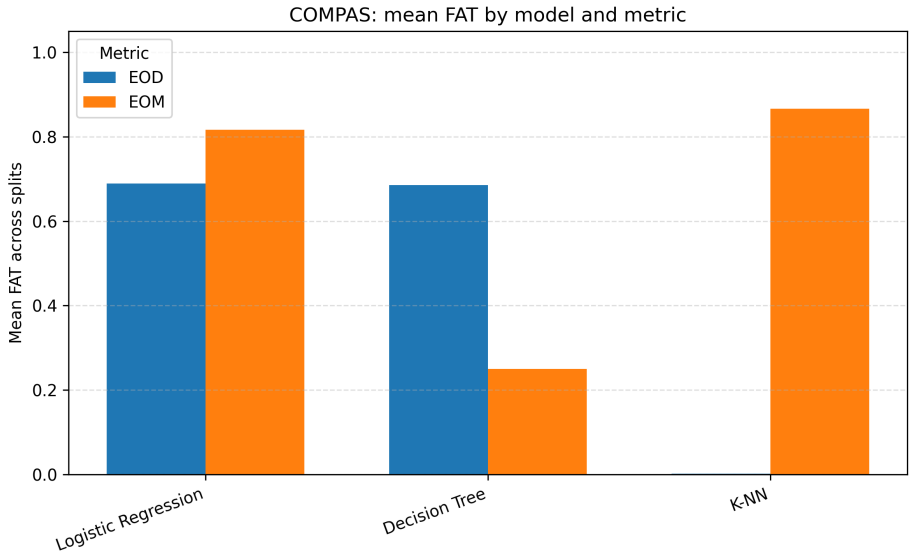

Figure 4.20: COMPAS dataset: mean FAT scores by model and metric.
X axis: ML algorithm.
Y axis: mean FAT over test splits.
Bars compare EOD and EOM.

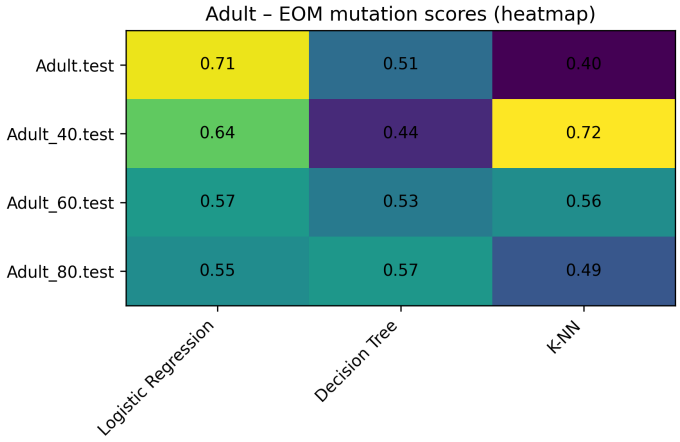

Figure 4.21: Adult dataset: EOM FAT scores by model and test split.
X axis: test files (full, 40%, 60%, 80%).
Y axis: ML algorithms.
Cell values: FAT scores.

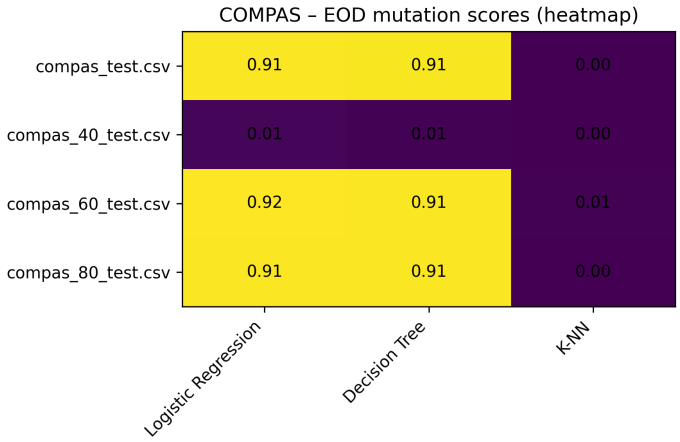

Figure 4.22: COMPAS dataset: EOD FAT scores by model and test split.
X axis: test files.
Y axis: ML algorithms.
Cell values: FAT scores.

1. On Adult, FAT scores under EOM form a relatively "bright" and uniform pattern, showing strong and stable adequacy.

2. On COMPAS, FAT scores under EOD show more variation across models and splits, indicating that some model–split combinations are harder to test ade-

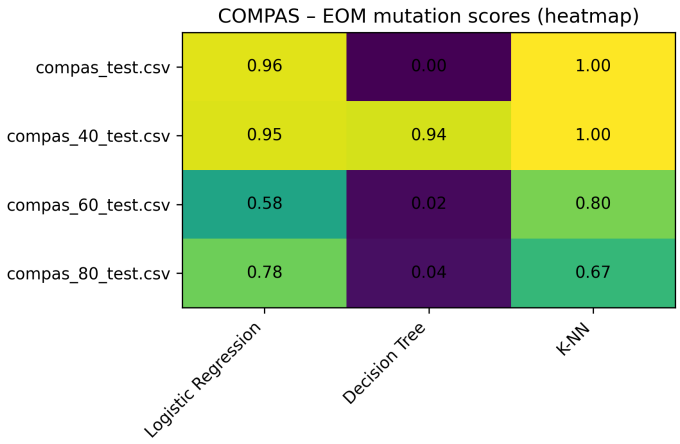

Figure 4.23: COMPAS dataset: EOM FAT scores by model and test split.
X axis: test files.
Y axis: ML algorithms.
Cell values: FAT scores.

quately for Equalized Odds.

3. Under EOM, COMPAS is slightly more stable than under EOD, but still shows more variation than Adult.

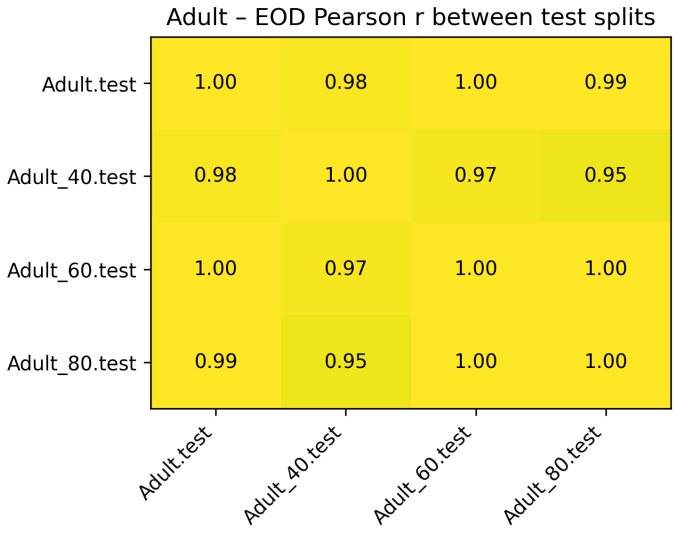

Figure 4.24: Adult dataset: Pearson correlation of EOD FAT scores between test splits.
X/Y axes: Adult test files.
Cell values: Pearson r.

Figure 4.25: Adult dataset: Pearson correlation of EOM FAT scores between test splits.
X/Y axes: Adult test files.
Cell values: Pearson r.

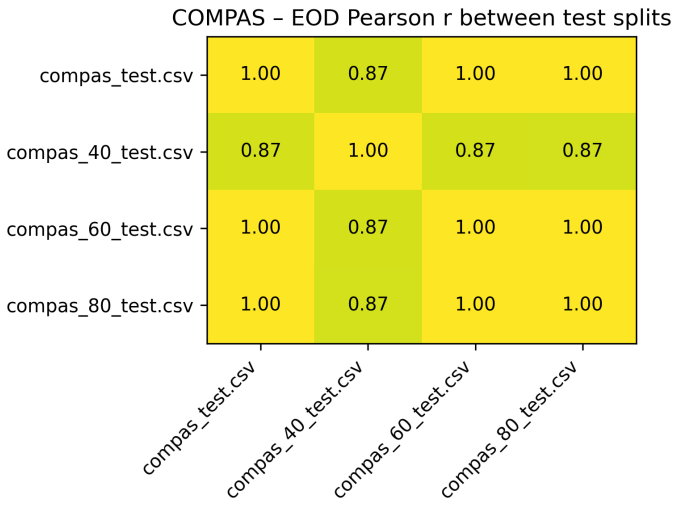

Figure 4.26: COMPAS dataset: Pearson correlation of EOD FAT scores between test splits.
X/Y axes: COMPAS test files.
Cell values: Pearson r.

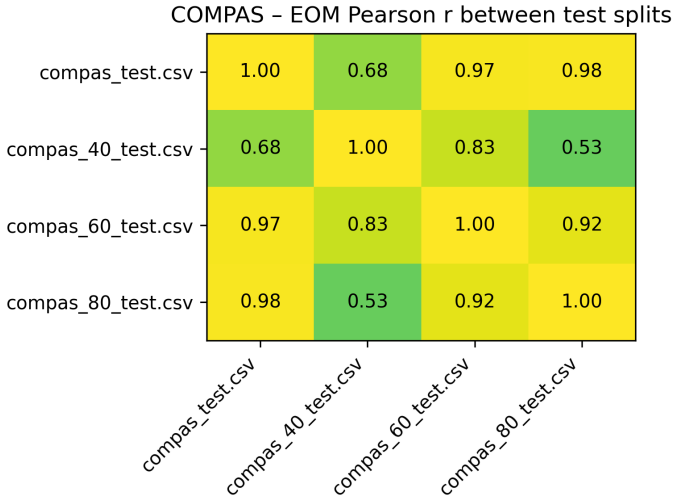

Figure 4.27: COMPAS dataset: Pearson correlation of EOM FAT scores between test splits.
X/Y axes: COMPAS test files.
Cell values: Pearson r.

RQ2 summary. The framework generalises across models and datasets, but not in a trivial way. For Adult, EOM-based adequacy is strong and stable across learners

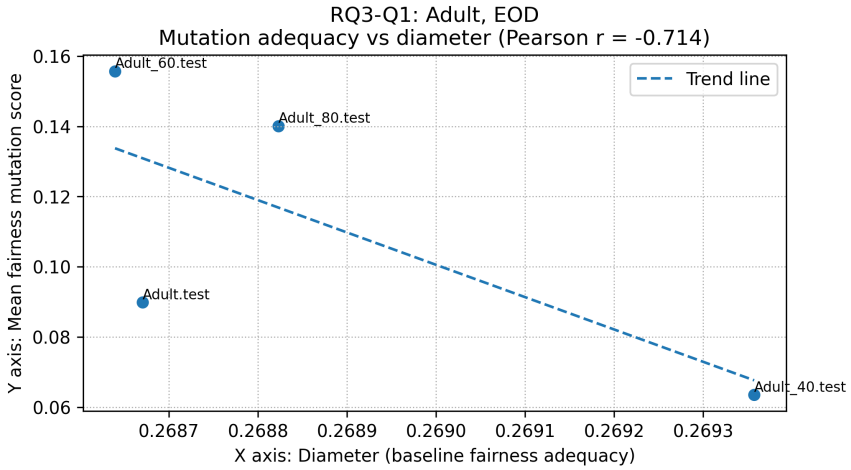

Figure 4.28: Adult dataset: FAT vs diameter under EOD.
X axis: diameter per test file.
Y axis: mean FAT score (0–1).

and splits. For COMPAS, EOD-based adequacy is more sensitive to the choice of model and sample size. These differences are visible in the graphs and reflect real differences between the datasets and fairness metrics.

4.8.5 RQ3: How Does Mutation-based Adequacy Compare to Diameter?

Finally, RQ3 compares FAT to a baseline adequacy proxy based on test diversity: the *diameter* of the test suite. The idea behind diameter is simple: larger diameter means more diverse tests, which should, in theory, give higher adequacy.

Here, we ask two questions:

- Do test files with larger diameter also tend to have higher FAT?
- When we put both on the same scale, which measure separates splits more clearly?

The scatter plots show partial alignment: in many cases, larger diameters coincide with higher FAT scores, but the relationship is not perfect. This makes sense:

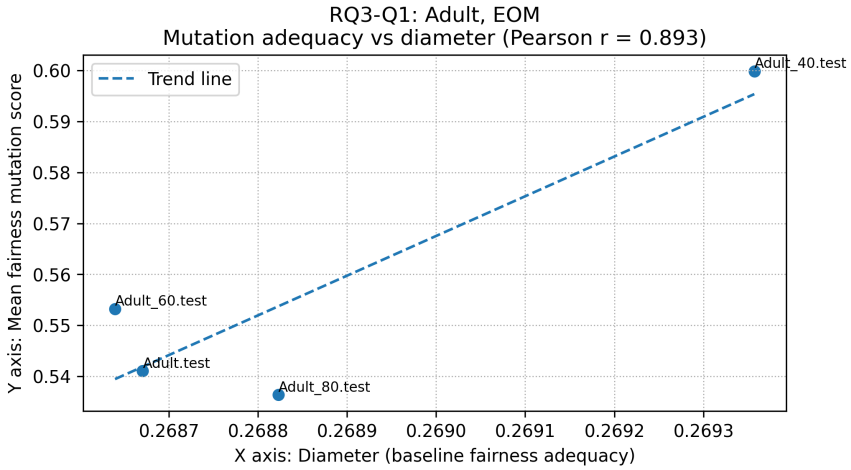

Figure 4.29: Adult dataset: FAT vs diameter under EOM.
X axis: diameter.
Y axis: mean FAT score.

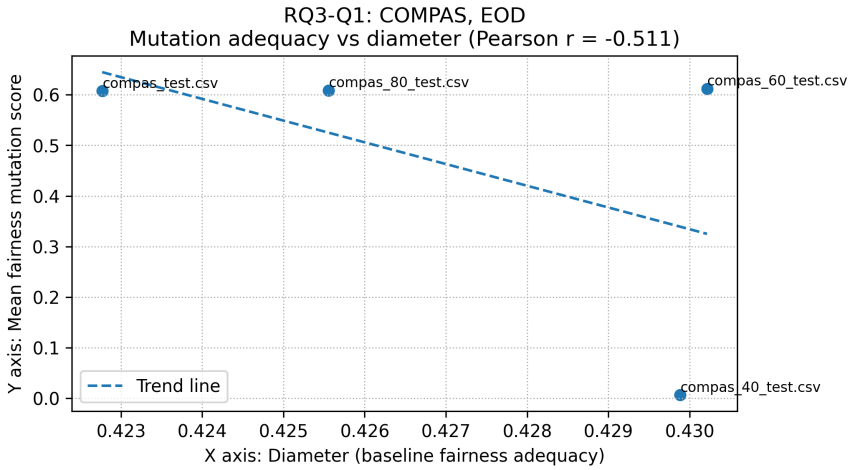

Figure 4.30: COMPAS dataset: FAT vs diameter under EOD.
X axis: diameter.
Y axis: mean FAT score.

diameter measures geometric diversity, not fairness-specific stress.

To make sensitivity differences clearer, we normalise both signals into $[0, 1]$ and compare them split by split.

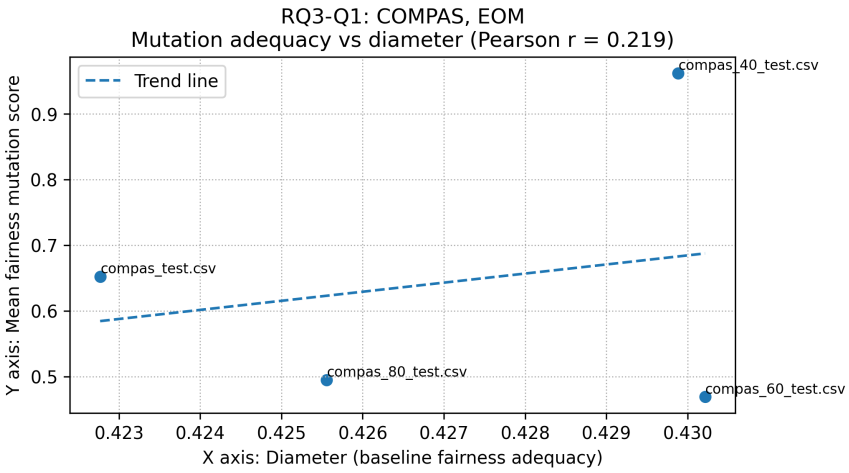

Figure 4.31: COMPAS dataset: FAT vs diameter under EOM.
X axis: diameter.
Y axis: mean FAT score.

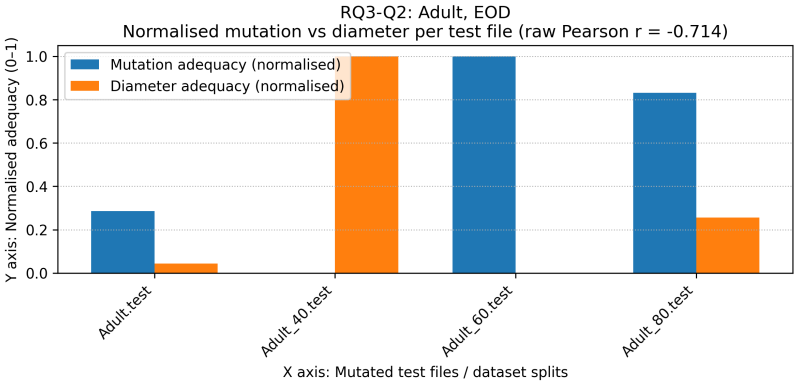

Figure 4.32: Adult dataset: normalised FAT vs diameter per split under EOD.
X axis: test files (full, 40%, 60%, 80%).
Y axis: normalised adequacy (0–1).

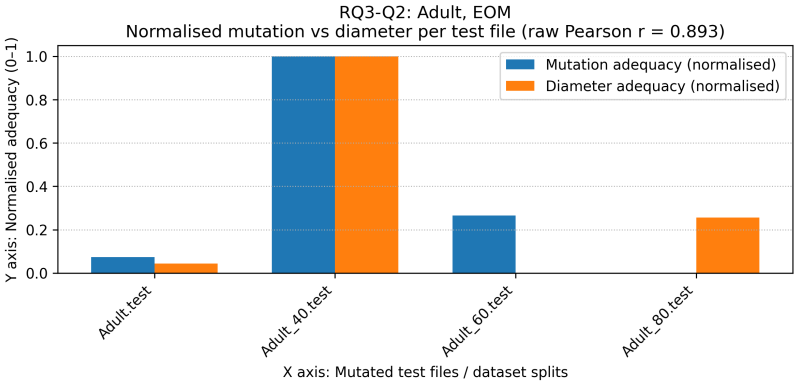

Figure 4.33: Adult dataset: normalised FAT vs diameter per split under EOM.
X axis: test files.
Y axis: normalised adequacy (0–1).

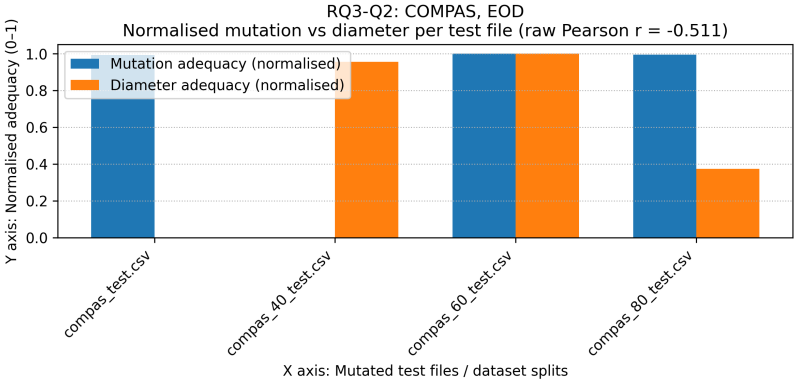

Figure 4.34: COMPAS dataset: normalised FAT vs diameter per split under EOD.
X axis: test files.
Y axis: normalised adequacy (0–1).

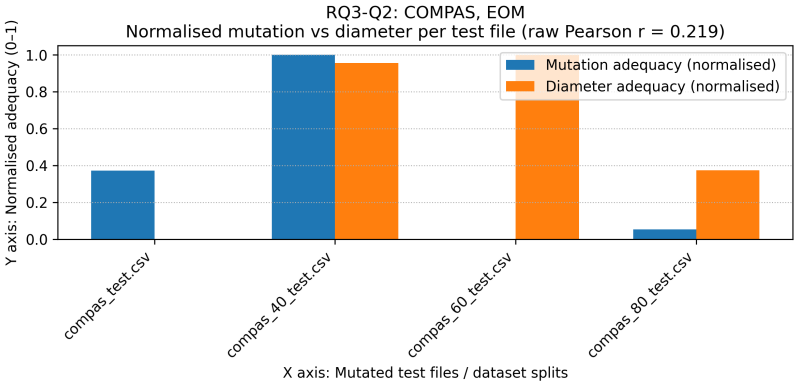

Figure 4.35: COMPAS dataset: normalised FAT vs diameter per split under EOM.
X axis: test files.
Y axis: normalised adequacy (0–1).

Across both datasets and both metrics, two patterns emerge:

- Diameter tends to cluster at similar values across splits, especially for smaller test files, which makes it hard to see real differences in adequacy.
- FAT usually spreads out more, especially under EOD. This means mutation-based fairness adequacy can distinguish between splits that look almost identical under diameter.

RQ3 summary. Mutation-based fairness adequacy and diameter are related but not redundant. Diameter captures general test diversity, but FAT captures how well the test suite detects fairness-specific faults. In both Adult and COMPAS, and for both EOD and EOM, FAT often provides a clearer separation between test files, especially under the stricter Equalized Odds lens.

Chapter 5

Threats to Validity, Future Work, and Conclusion

This chapter reflects on the key findings of the thesis and discusses the limits of the study. It highlights where caution is needed when interpreting the results, identifies factors that may have influenced fairness measurements, and outlines directions for future research. The goal is to help readers understand the scope of the findings and how the framework introduced in this thesis can be extended and strengthened in later work.

5.1 Threats to Validity

Fairness testing is inherently complex, and several factors may influence how reliable or generalisable the results are. The following sections outline four major categories of validity threats and explain how they may affect the study's conclusions.

5.1.1 Construct Validity

Construct validity refers to whether the study correctly measured the concepts it set out to measure.

- Although the fairness metrics used in this thesis such as EOD and EOM are widely used, they do not capture every possible form of bias. Fairness has many dimensions, and group-level metrics provide only one perspective.

- The mutation operators were designed to mimic realistic changes in input data or preprocessing pipelines. However, they cannot cover the full range of fairness failures that could appear in real deployments.

- The thresholding rule (mean + 1.5 standard deviations) provides a consistent way to decide when a mutant should be labelled as "killed," but using a different threshold could change which mutants are detected.

These factors mean that the adequacy results presented here address fairness under the definitions and assumptions used in this study, rather than fairness in an absolute or universal sense

5.1.2 Internal Validity

Internal validity considers whether the observed results could have been influenced by factors within the study itself.

- Fairness outcomes can vary depending on training decisions such as hyperparameter choices, data preprocessing, and random initialisation.

- Some fairness shifts detected during mutation testing may be caused by randomness in model training rather than genuine fairness faults.

- While the validity gate filters out unrealistic or trivial mutants, it may still include some mutants that reflect noise rather than meaningful fairness behaviour.

These issues introduce the possibility that some fairness detections or lack of detections are influenced by training variability rather than model behaviour.

5.1.3 External Validity

External validity concerns whether the findings generalise beyond the datasets studied.

- The experiments focus on two well-known benchmark datasets: Adult and COMPAS. Although widely used, they represent only limited real-world contexts.
- Many real systems involve more complex models, multi-modal inputs, or richer sensitive-attribute structures than those examined here.
- The mutation operators may behave differently on other types of data, such as text, images, time-series, or highly imbalanced domains.

For these reasons, caution is needed when applying the findings to new domains. Broader evaluation is required to confirm generality.

5.1.4 Statistical Conclusion Validity

Statistical conclusion validity concerns the reliability of inferences drawn from the data.

- Some protected subgroups are relatively small, which can make fairness metrics especially EOD unstable.
- Mutation testing involves evaluating many mutants, which increases the chance of false positives or false negatives.
- The fairness adequacy score depends on the number of non-equivalent mutants. If too many mutants are filtered out, the score may underrepresent the range of fairness behaviours present.

These considerations highlight the importance of interpreting fairness scores together with uncertainty estimates and subgroup sizes.

5.2 Future Work

Several opportunities remain to further develop fairness adequacy testing and broaden its practical use.

5.2.1 Expanding Datasets and Domains

Future research should examine the framework across a wider range of datasets and modelling tasks. Potential applications include healthcare, financial decision-making, hiring systems, and educational assessments. Extending the method to deep learning models and non-tabular data—such as images, text, or multi-modal inputs—would greatly enhance its scope.

5.2.2 Richer Fairness Definitions

The current study focuses on group-level parity metrics. Future research could incorporate more nuanced fairness concepts, including:

- causal fairness criteria,
- individual fairness,
- intersectional fairness patterns,
- fairness over time or sequential decision processes.

These additions would provide a deeper understanding of potential fairness failures.

5.2.3 Adaptive and Slice-Aware Thresholding

The thresholding rule used in the study is intentionally simple. Future work could incorporate thresholds that adapt to:

- subgroup sample sizes,
- class imbalance,
- estimated uncertainty,
- interactions across multiple fairness metrics.

Adaptive thresholds may improve the precision and reliability of mutant classification.

5.2.4 Enhanced Mutation Operators

New operators could help more accurately reflect real-world changes that cause fairness drift. Examples include:

- temporal shifts in input features,
- realistic noise affecting sensitive attributes,
- evolving population distributions,
- adversarial attacks intentionally manipulating fairness,
- policy-driven changes to decision thresholds.

These expansions would allow the framework to simulate a broader set of fairness risks.

5.2.5 Tooling and Practical Integration

To support adoption in real ML pipelines, fairness adequacy testing could be integrated into development tools and practices. Possible directions include:

- automated reporting and dashboards,
- CI/CD fairness checks similar to unit tests,
- integration with model monitoring systems,
- documentation formats aligned with regulatory audits.

Such tooling would help teams use fairness adequacy testing as part of routine model evaluation.

5.3 Conclusion

This thesis introduced a fairness adequacy framework that applies mutation testing to examine the strength of fairness evaluations. Instead of relying solely on static fairness measurements, the framework stresses models using controlled changes to assess whether fairness shifts are detectable.

Experiments on the Adult and COMPAS datasets demonstrated that:

- fairness-aware mutation operators can reveal meaningful and interpretable fairness changes,
- adequacy scores vary with dataset, model type, and fairness metric,
- smaller test sets still produce informative fairness signals,
- adequacy patterns show stability across different data splits.

Together, these findings suggest that fairness adequacy testing provides a practical and repeatable way to evaluate fairness sensitivity in machine learning systems. It helps turn fairness concerns into explicit and testable requirements, supporting developers in detecting fairness issues before model deployment.

Although additional work is needed to broaden fairness definitions, expand dataset coverage, and integrate the framework into industry workflows, this study provides a strong foundation for treating fairness as a measurable and testable property of ML systems.

BIBLIOGRAPHY

[1] Agarwal, A., Beygelzimer, A., Dudík, M., Langford, J., and Wallach, H. A reductions approach to fair classification.

[2] Ammann, P., and Offutt, J. *Introduction to Software Testing*, 2 ed. Cambridge University Press, 2016.

[3] Angwin, J., Larson, J., Mattu, S., and Kirchner, L. Machine bias. *ProPublica* (2016).

[4] Barocas, S., Hardt, M., and Narayanan, A. *Fairness and Machine Learning: Limitations and Opportunities*. 2023. Available online at fairmlbook.org.

[5] Behrouz, A., Zhong, P., and Mirrokni, V. Titans: Learning to memorize at test time, 2024.

[6] Bellamy, R. K. E., Dey, K., Hind, M., Hoffman, S. C., Houde, S., Kannan, K., Lohia, P., Martino, J., Mehta, S., Mojsilović, A., Nagar, S., Ramamurthy, K. N., Richards, J., Saha, D., Sattigeri, P., Singh, M., Varshney, K. R., and Zhang, Y. Ai fairness 360: An extensible toolkit for detecting and mitigating algorithmic bias. *IBM Journal of Research and Development 63*, 4/5 (2019), 4:1–4:15.

[7] Bishop, C. M. *Pattern Recognition and Machine Learning.* Springer, New York, 2006.

[8] Chouldechova, A. Fair prediction with disparate impact: A study of bias in recidivism prediction instruments. *Big Data 5*, 2 (2017), 153–163.

[9] Consumer Financial Protection Bureau. Consumer financial protection circular 2022-03: Adverse action notification requirements in credit decisions. `https://www.consumerfinance.gov/`, 2022.

[10] Consumer Financial Protection Bureau. Adverse action notice requirements. `https://www.consumerfinance.gov/`, 2023. Guidance on notices when using AI in credit decisions.

[11] Corbett-Davies, S., Gaebler, J. D., Nilforoshan, H., Shroff, R., and Goel, S. The measure and mismeasure of fairness, 2023.

[12] Dastin, J. Amazon scraps secret ai recruiting tool that showed bias against women. *Reuters* (2018).

[13] Dwork, C., Hardt, M., Pitassi, T., Reingold, O., and Zemel, R. Fairness through awareness. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference* (New York, NY, USA, 2012), ITCS '12, Association for Computing Machinery, pp. 214–226.

[14] European Union. Eu artificial intelligence act. `https://eur-lex.europa.eu/`, 2025.

[15] Feldman, M., Friedler, S., Moeller, J., Scheidegger, C., and Venkatasubramanian, S. Certifying and removing disparate impact, 2015.

[16] Galhotra, S., Brun, Y., and Meliou, A. Fairness testing: Testing software for discrimination. In *Proceedings of the 11th Joint Meeting on Foundations of Software Engineering* (2017), pp. 498–510.

[17] Grote, T. Fairness as adequacy: a sociotechnical view on model evaluation in machine learning. *AI and Ethics 4*, 2 (2024), 427–440.

[18] Hardt, M., Price, E., and Srebro, N. Equality of opportunity in supervised learning, 2016.

[19] Henley, J. Dutch government resigns over child welfare scandal. *The Guardian* (2021).

[20] Kearns, M., Neel, S., Roth, A., and Wu, Z. Preventing fairness gerrymandering: Auditing and learning for subgroup fairness.

[21] Kusner, M. J., Loftus, J. R., Russell, C., and Silva, R. Counterfactual fairness, 2018.

[22] Ma, L., Zhang, F., Sun, J., Xue, M., Li, B., Juefei-Xu, F., Xie, C., Li, L., Liu, Y., Zhao, J., and Wang, Y. Deepmutation: Mutation testing of deep learning systems. In *Proceedings - 29th IEEE International Symposium on Software Reliability Engineering, ISSRE 201818* (2018), vol. 2018, IEEE, pp. 100–111.

[23] Mehrabi, N., Morstatter, F., Saxena, N., Lerman, K., and Galstyan, A. A survey on bias and fairness in machine learning. *ACM Computing Surveys 54*, 6 (2021), 1–35.

[24] Mobley, D. Mobley v. workday, inc. — court filings and orders. `https://www.courtlistener.com/`, 2025.

[25] Myers, G. J., Sandler, C., and Badgett, T. *The Art of Software Testing*, 3 ed. Wiley, 2011.

[26] Obermeyer, Z., Powers, B., Vogeli, C., and Mullainathan, S. Dissecting racial bias in an algorithm used to manage the health of populations. *Science 366*, 6464 (2019), 447–453.

[27] of Representatives, D. H. Ongekend onrecht: Parliamentary interrogation committee on childcare allowance. Tech. rep., Tweede Kamer der Staten-Generaal, January 2021.

[28] Ofqual. Awarding gcse, as and a levels in summer 2020: interim report. Tech. rep., Office of Qualifications and Examinations Regulation, August 2020.

[29] Ovadia, Y., Fertig, E., Ren, J., Nado, Z., Sculley, D., Nowozin, S., Dillon, J. V., Lakshminarayanan, B., and Snoek, J. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift, 2019.

[30] Sample, I. A-level results: how did the algorithm work and why did it fail? *The Guardian* (2020).

[31] Sharma, A., and Wehrheim, H. Testing machine learning programs. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (2020), vol. 12471 LNCS, pp. 5–26.

[32] Tramer, F., Atlidakis, V., Geambasu, R., Hsu, D., Hubaux, J.-P., Humbert, M., Juels, A., and Lin, H. Fairtest: Discovering unwarranted associations in data-driven applications. 401–416.

FAIRNESS ADEQUACY TEST FOR MACHINE LEARNING SYSTEMS

by

Kehinde Oluwasayo Akinola

December, 2025

Director of Thesis: Srinivasan Madhusudan, PhD

Major Department: Computer Science

As Machine Learning (ML) systems assume a larger role in decisions that affect people's lives, such as who receives a loan, access to healthcare, or early release from prison, ensuring these systems are fair is more important than ever. However, current fairness checks often miss the subtle and complex ways in which bias can appear in algorithms.

This dissertation tackles that gap by proposing a practical framework for testing how well machine learning models meet fairness standards, especially across protected groups. Instead of relying solely on standard performance metrics, the approach combines statistical tools with stress testing techniques to uncover hidden or overlooked biases.

There are four main contributions. First, we introduce a fairness adequacy test using metrics like Equal Opportunity Difference (EOD) and Equalised Odds Metrics (EOM) to examine disparities in error rates across groups. Second, we apply mutation testing by altering sensitive features such as race or gender to see how model outputs change, helping assess fairness under different conditions. Third, we use permutation methods to simulate edge cases and test how models respond to unusual or extreme inputs. Finally, we validate this approach with real-world case studies in areas like healthcare, finance, and criminal justice, where fairness is especially critical.

By offering a clear and testable way to evaluate fairness, this work aims to support

the development of more trustworthy, accountable, and equitable AI systems

FAIRNESS ADEQUACY TEST FOR MACHINE LEARNING SYSTEMS

A Thesis

Presented to The Faculty of the Department of Computer Science

East Carolina University

In Partial Fulfillment of the Requirements for the Degree

Master of Science in Data Science

by

Kehinde Oluwasayo Akinola

December, 2025

FAIRNESS ADEQUACY TEST FOR MACHINE LEARNING SYSTEMS

by

Kehinde Oluwasayo Akinola

APPROVED BY:

DIRECTOR OF THESIS:	Srinivasan Madhusudan, PhD
COMMITTEE MEMBER:	Peng Kebin PhD
COMMITTEE MEMBER:	John Reisch, PhD
CHAIR OF THE DEPARTMENT OF COMPUTER SCIENCE:	Qin Ding, PhD
DEAN OF THE GRADUATE SCHOOL:	Debra Jackson, PhD

Table of Contents

LIST OF TABLES

LIST OF FIGURES

List of Acronyms

AI Artificial Intelligence. xii

AUC Area Under the Curve. xii

CART Classification and Regression Trees.. v, xii, 17, 20, 30, 49, 56, 61, 66–68, 76

CNNs Convolutional Neural Networks. xii

DBSCAN Density-Based Spatial Clustering of Applications with Noise. xii

DNNs deep neural networks. xii, 8

DP Disparity Parity. xii, 51

EO Equality of Opportunity. xii, 9

EOD Equal Opportunity Difference. xii, 1, 8, 9, 25, 26, 30, 37, 49, 51, 53–55, 57–62, 64–71, 73, 75–84, 86, 87, 89

EOM Equalised Odds Metrics. xii, 1, 8, 16, 25, 26, 30, 37, 49, 51, 55, 57–62, 64–72, 74–80, 82–87

FAT Fairness Adequacy Test. xii, 39, 49

FPR False Positive Rate. xii, 51, 55

K-NN K-Nearest Neighbors. v, x, xii, 17, 22–25, 30, 33, 48, 49, 56, 61, 66–68, 76

LightGBM Light Gradient Boosting Machine. xii

LR Logistic Regression. v, xii, 17, 19, 30, 49, 56, 61, 66–68, 76

MKR mutation kill rates. xii, 29, 30

ML Machine Learning. xii, 1, 2, 6, 9, 10, 14–16, 25, 28

MSR Mutation Score Rate. xii

PCA Principal Component Analysis. xii

Q-Learning Quality Learning. xii

RNNs Recurrent Neural Networks. xii

ROC Receiver Operating Characteristic. xii

RQ Research Questions. xii

SARSA State–Action–Reward–State–Action. xii

SPD Statistical parity difference. xii, 16, 19, 22, 25, 26, 30, 36, 49, 55

SVM Support Vector Machines. xii

t-SNE t-distributed Stochastic Neighbor Embedding. xii

TPR True Positive Rate. xii, 51, 54

UMAP Uniform Manifold Approximation and Projection. xii

WMFI Weighted Multi-Objective Fairness Index. xii, 49, 51

XGBoost Extreme Gradient Boosting. xii

Chapter 1

Introduction

1.1 What Is Testing and the Adequacy Principle

A model can report 95% accuracy and still deny loans to a protected group 20% more often than to others. This gap between statistical performance and social impact shows why testing remains central to responsible deployment.

In classical software engineering, adequacy asks whether a test suite exercises the paths most likely to reveal defects (21). In machine learning, this idea needs to be adapted. Models learn from historical data, make probabilistic predictions, and operate in social contexts that can change over time (17). Adequacy therefore moves from *code coverage* to *slice coverage*: systematic evaluation of fairness-relevant subpopulations, feature contexts, and plausible distributional shifts.

A fairness test suite should be able to reveal faults that affect fairness, not only those that reduce overall accuracy. From a sociotechnical perspective (17), we apply precise measurements of error rates to outcomes that matter for individuals and communities. In this thesis, we operationalise this idea through mutation testing. We create small but realistic changes (mutants) in the data, enforce basic distribution and validity checks, retrain the model where needed, and then check whether the test suite detects a meaningful shift in fairness metrics such as Equal Opportunity Difference (EOD) or Equalized Odds Metric (EOM). When a fairness metric changes

beyond a data-driven threshold, the mutant is considered *killed*; otherwise it *survives*. The kill-versus-survive decision uses a control limit derived from the base model's null variability (mean + 1.5 std), estimated via bootstrap resampling.

In this way, adequacy is reframed as the ability of a test suite to detect fairness-relevant faults under realistic perturbations, not just to report fairness numbers on a single, fixed test set.

1.2 Why Fairness Testing Is Necessary

Standard evaluation metrics such as accuracy, precision, and recall often fail to reveal harms that arise in specific subgroups due to historical bias, proxy labels, or distribution shift (23; 31). Because ML now informs decisions in areas such as credit, employment, criminal justice, and health, these hidden failures can scale into systemic discrimination.

Fairness testing treats equity as a measurable property of an AI system. It is the process of checking whether a model produces outcomes that differ across groups (for example, by gender, race, or age) without a justified reason, and of recommending targeted corrections rather than generic fixes (16; 32). Tools such as FairTest and Themis detect statistically significant disparities in a practical way, spanning tabular data, text, and vision (32).

A widely discussed example underlines this need. An internal audit showed that Amazon's experimental hiring tool systematically downgraded CVs that contained women-associated terms, even though overall performance looked competitive (12). The audit did not merely report accuracy; it used fairness-oriented checks to show that the model was not performing equally well across groups. This kind of evidence illustrates why fairness testing must complement standard utility metrics.

1.3 Global Incidents Caused by Inadequate Fairness Testing

Several high-profile incidents show how insufficient fairness testing can cause concrete harm. Each case pairs a missing test signal with an adequacy lesson that this thesis aims to make operational.

- **Criminal justice: COMPAS risk assessment (2016).** Empirical studies showed that the COMPAS algorithm assigned high-risk labels to Black defendants at much higher rates than to white defendants, even though both groups had similar rearrest rates (3; 8). This unequal treatment reflects a methodological gap: subgroup-specific confusion matrices and uncertainty assessments were not treated as standard evaluation artefacts.

- **Hiring: Amazon recruiting prototype (2018).** Investigations reported that Amazon's experimental résumé-filtering system, trained largely on male-dominated historical data, downgraded résumés containing women-associated terms (12). The lack of shortlist parity checks and feature-attribution audits meant that this behaviour went undetected until late in the process.

- **Healthcare: Risk scoring (2019).** A widely used healthcare risk scoring algorithm relied on healthcare costs as a proxy for clinical need, which led to Black patients being under-referred to high-risk care programmes despite having similar clinical risk to white patients (26). This bias highlights the absence of label-validity assessments and subgroup-specific error analyses in the evaluation pipeline.

- **Education: UK A-level grading algorithm (2020).** Evaluations of the grading system used during the COVID-19 pandemic showed that the standardisation procedure disproportionately downgraded students from certain schools

(28; 30). The failure to test on small-cohort subsets and to apply strong distributional stress tests contributed to these skewed outcomes.

- **Public benefits: Dutch childcare scandal (2018–2021).** In the Dutch childcare benefits scandal, automated profiling systems misclassified thousands of families as fraudsters and disproportionately targeted dual-citizen households (27; 19). Here, the system failed to account for protected characteristics and allowed proxy features to drive decisions, which resulted in inequitable and discriminatory outcomes.

These incidents are not isolated technical errors; they are signs that fairness-relevant behaviour was not adequately tested before deployment.

1.4 Implications of Not Conducting Fairness Testing

The incidents above illustrate at least four types of risk that arise when fairness testing is weak or absent:

1. **Legal and compliance.** Lack of documented fairness testing can lead to investigations, penalties, and lawsuits.
2. **Institutional trust.** Exposed disparities reduce the confidence of users, regulators, and the public, and increase the cost of assurance.
3. **Technical debt.** Aggregate metrics can hide fairness defects that appear only in specific slices or under realistic shifts, making later remediation more complex.
4. **Social and ethical harm.** Unchecked algorithms can deepen disadvantage in finance, employment, healthcare, education, and legal decision-making.

Table 1.1: Documented failures linked to missing or inadequate fairness testing across critical domains.

Domain	**Incident (Year)**	**Outcome**	**Type of Harm**	**Missed Fairness Signal**
Justice	COMPAS bias in sentencing (2016)	Black defendants received higher risk scores	Racial discrimination	No subgroup confusion matrix or uncertainty analysis
Employment	Amazon résumé screening (2018)	Female-associated terms were downgraded	Gender bias	No feature-attribution audit or shortlist parity check
Healthcare	Risk prediction model (2019)	Black patients underreferred despite equal risk	Proxy-label bias	No label-validity check or subgroup error analysis
Recruitment	Automated screening lawsuit (2022)	Older and Black candidates excluded	Age and race discrimination	No slice-level evaluation or intersectional coverage
Lending	Credit scoring disparities (2021)	Minority borrowers received worse terms	Disparate impact	No distributional stress test or sensitive attribute guardrails

Table 1.2: Disparities in COMPAS risk labels and rearrest rates by racial group (ProPublica analysis).

Group	**Labeled High Risk (%)**	**Actual Rearrested (%)**
Black	45	20
White	23	13

Source: Adapted from ProPublica analysis (3).

As shown in Tables 1.1 and 1.2, disparities can remain hidden if evaluation focuses only on global accuracy or utility scores. The evidence suggests that fairness testing should be treated alongside accuracy, reliability, and security as a basic requirement for responsible AI. In practice, this means setting explicit slice-coverage goals, using uncertainty-aware thresholds, summarising fairness-sensitive mutants with mutation scores and confidence intervals, and preserving artefacts that show that fairness-altering mutants would have been detected before deployment.

1.5 Importance of Solving This Problem

There are strong reasons, both ethical and practical, to require fair behaviour from ML systems, especially in sensitive domains such as credit scoring, policing, and healthcare. Fair models help reduce harm, support trust, and make it more likely that all demographic groups receive a fair chance at resources and opportunities. Addressing fairness is therefore not only a technical goal but also a condition for responsible and sustainable deployment (6).

From a technical perspective, fairness testing has a clear advantage: it can reveal problematic behaviour that would otherwise remain hidden when only overall performance metrics are considered. It encourages systematic error analysis across different subgroups and feature contexts rather than relying on a single global accuracy score. In this thesis, fairness is treated as a *test adequacy requirement* rather than only a diagnostic report: we ask whether the test suite is strong enough to uncover fairness-related bugs before deployment (7).

To operationalise this requirement, we simulate small, realistic data changes under distributional constraints and check whether the fixed test suite detects changes in fairness metrics. A mutant is labelled *killed* when the absolute change in a fairness

metric exceeds a threshold derived from the base model's null variability (mean + 1.5 std). The resulting mutation score measures how well the test suite is exposed to fairness defects through sensitive slices and plausible shifts. This forms the basis for an adequacy-style certificate indicating whether fairness remains within permissible limits (1).

Taken together, these elements support the view that fairness should be built into the testing process from the start, not treated as a later add-on. Embedding fairness adequacy into testing supports technical robustness and aligns with broader social expectations that automated decision systems should not discriminate.

1.5.1 Foundational Fairness Definitions and Trade-offs

The formal study of algorithmic fairness began with work such as (15), which introduced early definitions of disparate impact and proposed preprocessing methods to mitigate data bias. These ideas provided an initial technical basis for fairness research. Building on this, (18) proposed the principle of Equal Opportunity, which requires that demographic groups have similar true positive rates. Because it is grounded in supervised learning and can be measured from confusion matrices, Equal Opportunity quickly became a central fairness notion.

Subsequent research examined how fairness criteria interact. (8) proved that it is impossible, in general, to satisfy both calibration and equal error rates when groups have different base rates. This result formalised an important trade-off between fairness measures and underscored the need to make fairness goals explicit. Extending this perspective, (21) introduced counterfactual fairness, which requires model decisions to remain unchanged under hypothetical variations in an individual's protected attributes. This causal framing emphasises the importance of understanding how sensitive features influence outcomes through the underlying data-generating processes.

These foundational definitions and trade-offs motivate the choice of metrics in this thesis, especially EOD, EOM, and related group-based criteria.

1.5.2 Mutation and Metamorphic Testing for ML Robustness

Mutation testing was first developed in software engineering as a strong test adequacy criterion. The idea is to introduce small faults (mutants) into the program and check whether the test suite can detect them (25). If many mutants survive, the test suite is considered weak. In machine learning, a related approach is metamorphic testing, which applies input transformations that should preserve certain properties of the output in order to address the oracle problem. For example, DeepTest applies metamorphic transformations to driving scenes to detect failures in autonomous vehicle systems (5).

DeepMutation (22) extended mutation testing to deep neural networks (deep neural networks (DNNs)). Mutants are created at the source level (for example, by altering training data, labels, or architecture) and at the model level (for example, by perturbing neurons or weights), yielding multiple mutated model variants. When the test suite fails to kill these mutants by exposing performance degradation, it reveals gaps in evaluation and shows that high accuracy alone does not guarantee robust behaviour.

This thesis adapts this methodology to fairness evaluation. Instead of focusing only on robustness to generic perturbations, we design fairness-oriented mutations, such as perturbations that stress decision boundaries across protected groups or reweight subpopulations. If fairness tests fail to detect the induced disparities, the evaluation suite is judged inadequate. Mutation testing thus provides a principled way to extend test adequacy from robustness into fairness.

1.5.3 Empirical Failures Motivating Fairness Evaluation

Empirical failures such as the COMPAS disparities (3) motivated the adoption of fairness metrics like Equal Opportunity (Equality of Opportunity (EO)) and Equalized Odds (EOD). Similarly, (26) showed that a clinical risk algorithm misused cost as a proxy for need, leading to under-referral of Black patients, and highlighted the dangers of unexamined proxy labels. Taken together, these cases emphasise the need for systematic fairness testing that moves beyond static audits and single-dataset evaluations.

1.5.4 Empirical Failures Motivating Fairness Evaluation

ProPublica's COMPAS analysis (3) revealed racial disparities in recidivism predictions, motivating the adoption of Equal Opportunity EO and Equalized Odds metrics EOD. Similarly, (26) demonstrated that a widely used clinical risk algorithm misinterpreted *cost* as a proxy for *need*, under-referring Black patients and exposing the dangers of proxy labels. Taken together, these cases underscore the need for systematic fairness testing that goes beyond static audits and isolated evaluations.

1.5.5 Testing and Adequacy in ML

Addressing such failures requires revisiting what "testing" means in the context of ML. Myers originally defined testing in classical software as covering the paths most likely to reveal bugs. Extending this idea, (31) applied property-based testing to ML and recommended that non-functional requirements, such as fairness and robustness, be explicitly tested. From a broader sociotechnical perspective, (17) argued that adequacy must be understood in terms of real-world effects rather than only laboratory metrics. These insights provide the conceptual bridge between fairness failures and

the need for adequacy-style testing in ML.

1.5.6 How This Work Differs

Against this background, the present work differs from existing audits and toolkits that primarily ask, "Is this model biased?". Robustness research has shown that mutation testing can reveal hidden accuracy bugs. Here, the same principle is extended to fairness by introducing a quantified notion of *fairness test adequacy*: given realistic, fairness-relevant mutations, does the existing test suite consistently uncover unfair treatment?

More concretely, this thesis proposes:

- A validity check that certifies that the applied mutations are realistic and capable of inducing plausible fairness fluctuations in the training data.
- A bootstrap methodology that sets detection thresholds (mean + 1.5 standard deviations) to separate genuine fairness signals from random noise.
- Coverage and detection-power analysis using multiple test sets (balanced, skewed, and group-dropped) to assess how well fairness problems are detected.

By adopting this approach, fairness becomes a specific testing requirement with measurable detection power and audit-ready evidence, rather than a vague or optional goal.

To connect these ideas to a concrete context, we now consider the deployed system at the centre of the case study and ask what went wrong in evaluation terms. Filings and expert commentary point to at least one major gap that rigorous fairness testing would have addressed: (i) insufficient *slice coverage* across protected and intersectional groups during evaluation.

(ii) Weak *shift sensitivity* to small cohorts and feature sparsity patterns by age or race.

(iii) Missing *audit artifacts* that tie fairness requirements to concrete tests and thresholds.

Had these been in place, the vendor and customers could have shown evidence that group-level disparities would be detected pre-deployment.

Policy is converging on documented bias controls. The EU AI Act imposes lifecycle obligations for high-risk systems, including risk management, data governance, logging, and post-market monitoring (14). In U.S. credit, the Consumer Financial Protection Bureau requires specific adverse-action reasons even when AI is used (10; 9). Organizations that cannot produce test artifacts face enforcement risk and slower adoption.

1.5.7 Compliance and Legal Implications Beyond One Case

The need for rigorous fairness testing is not limited to a single system or lawsuit. Policy discussions and early regulatory texts increasingly emphasise bias controls and auditability in AI systems. For example, initiatives such as the EU AI Act and guidance from the U.S. Consumer Financial Protection Bureau suggest that fairness testing could become part of compliance obligations in high-risk domains (14; 10).

Although this thesis does not propose legal rules, it demonstrates how fairness adequacy can be operationalised in practice. In future work, the framework presented here could be extended to support compliance settings by producing audit-ready artefacts that align with emerging legal and policy requirements.

How Our Approach Addresses These Gaps

This thesis treats fairness as a question of test adequacy, not only metric reporting. In particular, we:

1. Build explicit **slice coverage** (age, race, gender, and intersections) into the test suite.

2. Generate **realistic mutations** (cohort reweighting, proxy-feature stress, sparse résumé sections) to simulate shifts that may induce fairness faults.

3. Enforce a **mutant-validity gate** (distribution similarity plus training-set fairness shift) to avoid meaningless mutants.

4. Apply a **data-driven detection threshold** using the base model's bootstrap null (mean + 1.5 std; see Section 3.4.2) and declare a mutant *killed* when the fairness change exceeds that threshold.

5. Summarise detection power with a **mutation score** and confidence intervals, producing *audit-ready artefacts* (test suites, operator configurations, thresholds, logs).

In a résumé-screening pipeline, for example, our mutations could (a) up- and down-weight applicant segments common among older candidates, (b) jitter or mask proxy features for age or race (such as graduation year or tenure patterns), and (c) simulate employer filters. If the suite fails to kill a sufficient fraction of these fairness-affecting mutants (that is, if the mutation score falls below a target), the build would be considered inadequate and a remediation report would identify missing slices, fragile features, and data-collection needs.

Industry and National Relevance

Hiring and employment. Active litigation is redefining responsibilities for vendors and deployers. Maintaining test artefacts can mitigate legal and reputational risk (24).

Healthcare. A widely used clinical risk model under-referred Black patients because cost was used as a proxy for need; our operators explicitly perturb proxy labels and cohort mix to reveal such failures before rollout (26).

Finance. When AI supports credit decisions, creditors must provide specific adverse-action reasons; our artefacts aim to tie slice-level findings to compliant explanations (10; 9).

Regulation. The EU AI Act codifies measurable, documented bias controls for high-risk domains; our approach produces an evidence trail that can support such requirements (14).

Bottom Line

Fairness failures increasingly trigger legal and regulatory responses, not just academic debate. A mutation-based adequacy protocol turns fairness from an abstract aspiration into an *operational gate* with measurable detection power and auditable proof, so that unfair outcomes can be identified and addressed before they are deployed at scale.

1.6 Related Work

Fairness in machine learning is an interdisciplinary topic spanning computer science, law, and policy. Significant progress has been made in defining fairness measures and developing auditing tools, but the systematic quantification of fairness test adequacy

remains relatively unexplored. The rest of this chapter has outlined the key strands of work that motivate this thesis: formal definitions and trade-offs between fairness metrics, mutation and metamorphic testing for ML, empirical failures in deployed systems, and emerging compliance expectations. The following chapters build on this foundation by specifying a fairness-aware mutation testing framework and evaluating its adequacy on benchmark datasets.

Chapter 2

Background

2.1 Introduction to Machine Learning

Machine learning (ML) has evolved from a research curiosity into a widely deployed technology that influences decisions in finance, healthcare, hiring, and criminal justice. In many of these settings, algorithmic predictions are treated not merely as suggestions but as decisions that directly affect people's lives. This chapter situates our work within the broader ML landscape and motivates the need for *fairness adequacy* in supervised classification systems.

2.2 Machine Learning in High-Stakes Decisions

Early ML applications were mostly limited to low-risk tasks such as spam filtering and handwriting recognition. Today, however, ML models are increasingly used in high-stakes domains, including:

- **Finance:** Credit scoring algorithms influence loan approvals and interest rates.
- **Healthcare:** Predictive models support diagnoses, treatment prioritisation, and resource allocation.
- **Hiring:** Automated résumé screening affects who is shortlisted and who is

filtered out.

- **Criminal justice:** Risk assessment tools guide parole decisions and sentencing recommendations.

These applications raise ethical questions about fairness and accountability (4). Fairness auditing often relies on group fairness metrics such as Statistical Parity Difference (Statistical parity difference (SPD)), Equal Opportunity Difference, and Equalized Odds Metric (EOM). However, real-world environments are not static. Input distributions shift, thresholds are tuned, and feature definitions evolve over time. Empirical studies show that both predictive performance and fairness metrics can degrade under even modest distribution shifts (29).

In addition, broad demographic categories can obscure intersectional harms. For example, focusing only on "male vs. female" or "Black vs. white" may miss disparities affecting older Black women or younger immigrant men. These observations motivate the notion of **fairness adequacy**: assessing whether fairness remains within acceptable bounds under realistic perturbations, rather than only at a single, fixed snapshot (20).

2.3 Formal Machine-Learning Pipeline

An ML algorithm aims to approximate a function

$$f : \mathcal{X} \to \mathcal{Y}$$

from experience

$$\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n}.$$

Historical bias embedded in $\mathcal{D}$ can propagate into f, systematically disadvan-

taging protected groups. This thesis focuses on binary classification in supervised learning, because fairness metrics are most developed and most legally scrutinised in this setting.

Figure 2.1: Formal machine-learning pipeline: Data collection → Preprocessing → Model training → Evaluation → Deployment.

2.4 Base Learners in Our Experimental Campaign

Our empirical evaluation uses three widely applied algorithms: Logistic Regression (LR), Decision Trees (CART), and k-Nearest Neighbours (K-NN). For each model family, we briefly review inductive bias, computational complexity, interpretability, and fairness-relevant failure modes.

2.4.1 Logistic Regression LR

Logistic Regression is one of the most commonly used linear classifiers in statistics and machine learning. Originally introduced as a model for binary outcomes, it has become a workhorse in domains such as credit scoring, medical diagnosis, and risk assessment, largely because of its efficiency and interpretability.

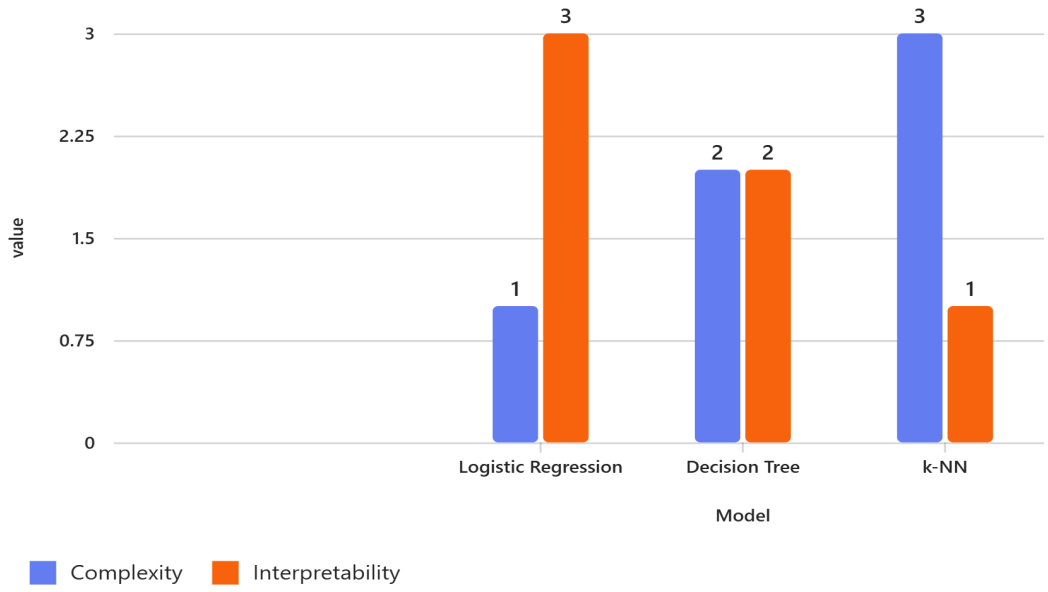

Figure 2.2: Complexity versus interpretability for Logistic Regression, Decision Tree, and k-NN.

Unlike linear regression, which predicts continuous values, logistic regression models the probability of a binary outcome using the logistic (sigmoid) function:

$$\hat{p}(y = 1 \mid x) = \sigma(w^\top x + b), \quad \sigma(z) = \frac{1}{1 + e^{-z}}.$$

Here, w is the vector of feature weights, b is the bias term, and $\sigma(\cdot)$ maps real values to the interval $(0, 1)$. The decision boundary is the hyperplane $w^\top x + b = 0$, which separates the feature space into predicted positive and negative regions.

Training. Model parameters (w, b) are typically estimated by maximising the likelihood of the observed data, or equivalently by minimising the negative log-likelihood. To prevent overfitting and encourage sparsity or smoothness, a regularisation term is added:

$$\mathcal{L}(w) = -\sum_{i=1}^{n} \Big[y_i \log \hat{p}_i + (1 - y_i) \log(1 - \hat{p}_i) \Big] + \lambda \|w\|_p,$$

where $\|w\|_p$ is either the ℓ_1 (Lasso) or ℓ_2 (Ridge) penalty. Optimisation is typically

performed using Newton's method, coordinate descent, or gradient-based methods.

Complexity. Training complexity is roughly $O(nd^2)$ with second-order methods or $O(ndT)$ with gradient descent, where n is the number of samples, d the number of features, and T the number of iterations. Prediction is efficient at $O(d)$ per query, and the memory footprint is $O(d)$.

Interpretability. A key advantage of logistic regression is its transparency: each coefficient w_j can be interpreted as the change in log-odds of the positive outcome per unit increase in feature x_j, holding other features constant. This interpretability makes LR particularly attractive in regulated domains where explanations are required.

Fairness sensitivity. The global linear decision boundary also makes logistic regression sensitive to fairness issues. Any non-zero weight on a protected attribute or a strong proxy can systematically tilt the boundary against a group. For example, if x_2 is correlated with gender and $w_2 > 0$, then many female applicants may receive lower predicted probabilities across the entire feature space. This can lead to violations of SPD or opportunity-based metrics that cannot be resolved by threshold tuning alone.

To make this idea more concrete, consider a setting in which the majority group cluster lies mostly on the accepted side of the boundary, while a minority group cluster is shifted towards the rejected side. Even well-qualified individuals from the minority group may be classified as negative because the single linear boundary has been influenced by group-level correlations.

Hyper-parameters. In practice, behaviour is shaped by a small set of hyper-parameters, including the penalty type (ℓ_1 vs. ℓ_2), inverse regularisation strength $C \in \{0.01, 0.1, 1, 10\}$, class-weight balancing, and, in some setups, fairness-regularised variants. These choices govern the trade-off between accuracy, sparsity, and fairness.

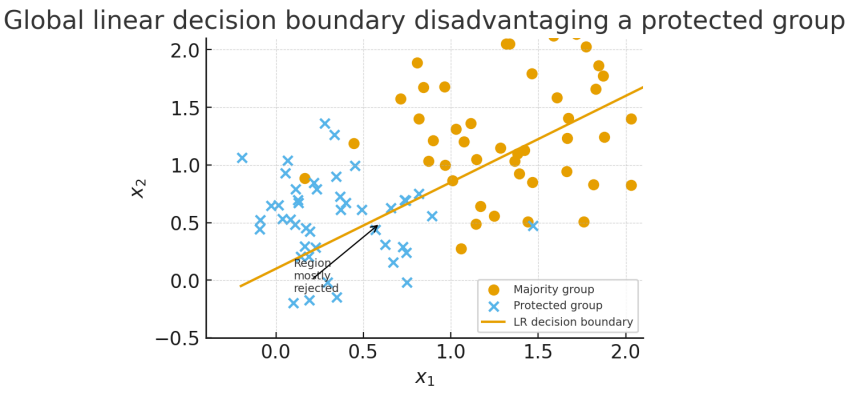

Figure 2.3: Illustration of a global linear decision boundary learned by Logistic Regression. The majority group (circles) lies mostly on the accepted side of the boundary, while many points from a protected group (crosses) fall into a predominantly rejected region.

Mutant kill signature. In our mutation analysis, logistic regression is particularly sensitive to covariate-shift mutants that rotate the linear separator relative to minority directions. Because the model uses a single global boundary, small distributional changes can disproportionately affect protected groups, leading to relatively high fairness mutation kill rates when tests are adequate.

2.4.2 Decision Tree CART

After considering a globally linear model, we now turn to a non-linear alternative: decision trees. These models partition the feature space into axis-aligned regions through recursive splits on individual features. Originating from the CART framework, decision trees are widely used in credit scoring, medical triage, and risk assessment. Their appeal comes from their ability to capture complex feature interactions and to express decisions as human-readable rules.

Formally, a decision tree represents a piecewise-constant function $f(x)$, where each input x is routed from the root node to a leaf via a sequence of binary tests of the form $x_j \leq \tau$. Each leaf ℓ stores an empirical label distribution $\hat{p}_\ell(y)$, and predictions

are obtained by

$$\hat{y}(x) = \arg\max_{y} \hat{p}_\ell(y), \quad \ell = \text{leaf reached by } x.$$

- **Training.** Trees are grown top-down in a greedy fashion. At each node t with sample set S_t, the algorithm searches over candidate features j and thresholds τ to maximise impurity reduction:

$$\Delta I(j, \tau) = I(S_t) - \frac{|S_L|}{|S_t|} I(S_L) - \frac{|S_R|}{|S_t|} I(S_R),$$

  where S_L and S_R are the left and right subsets induced by the split, and $I(\cdot)$ is an impurity measure such as Gini

$$I_{\text{Gini}}(S) = \sum_k p_k(1 - p_k)$$

  or entropy

$$I_{\text{ent}}(S) = -\sum_k p_k \log p_k.$$

  Growth stops when a maximum depth is reached, the node is pure, or too few samples remain, possibly followed by pruning.

- **Complexity.** With n samples, d features, and maximum depth D, a typical implementation that sorts feature values per node has training cost on the order of $O(ndD)$ (often approximated as $O(nd \log n)$ for balanced trees). Prediction for a new instance follows a single path from root to leaf, giving $O(D)$ time and a memory footprint proportional to the number of nodes.

- **Interpretability.** Each root-to-leaf path corresponds to a simple rule (e.g., "if income $\leq$ 30k and age $>$ 40 then reject"). These rules can be visualised and

discussed with domain experts, giving decision trees high local interpretability. Very deep trees, however, may be harder to interpret globally.

- **Fairness sensitivity.** Fairness issues arise when splits depend on protected attributes (e.g., gender, race) or strong proxies (e.g., postcode). A single split on such a feature can route most members of a protected group into a branch with systematically lower predicted scores, leading to large SPD or opportunity-based disparities. Because splits are hard thresholds, individuals just on different sides of a boundary can receive very different outcomes.

Hyper-parameters. Key hyper-parameters include maximum depth D, minimum samples per split and per leaf, the impurity criterion (Gini vs. entropy), and class weights. Fairness-aware variants may restrict certain features from being used in splits or add penalties for splits that increase group-level disparity.

Mutant kill signature. In our mutant analysis, decision trees are most sensitive to mutants that shift samples across split thresholds or alter impurity near the top of the tree. Small perturbations in feature values or labels around these thresholds can flip branches of instances from one leaf to another, changing outcomes for protected groups and exposing the fragility of the learned partition.

2.4.3 k-Nearest Neighbours K-NN

k-Nearest Neighbours (K-NN) is a non-parametric method whose inductive bias is based on locality (11). Unlike parametric models, K-NN does not assume a fixed functional form. Instead, it classifies a new point based on the labels of nearby training points in feature space.

- **Training.** K-NN is a "lazy learner": there is no separate training phase beyond storing the dataset. Given a query point x_*, distances to all training points are

computed, and the k nearest neighbours are selected. In classification, the predicted label $\hat{y}_*$ is obtained by majority vote among those neighbours; in regression, the prediction is the average of their target values.

- **Distance metrics.** The choice of distance function is crucial. Common metrics include Euclidean and Manhattan distance, as well as cosine similarity. In weighted variants, closer neighbours receive higher weights. Feature scaling and normalisation are often necessary so that features with larger ranges do not dominate distance computations.

- **Complexity.** Training complexity is $O(1)$ (storing the dataset). For brute-force search, query complexity is $O(nd)$, where n is the number of samples and d the number of features. In low dimensions, data structures such as KD-trees or Ball Trees can reduce the average query time to $O(\log n)$, but performance degrades in high-dimensional spaces due to the curse of dimensionality.

- **Interpretability.** K-NN has intuitive local interpretability: predictions can be explained in terms of "similar past cases." However, in high-dimensional feature spaces, distances may lose meaning and the neighbourhood structure may be harder to interpret.

- **Fairness sensitivity.** Because predictions are based on local majority voting, K-NN can amplify clustered bias. If instances from a minority group are surrounded by majority-group points, their labels may be repeatedly outvoted, leading to systematic disparities. This is especially problematic in regions of the feature space where protected groups are underrepresented.

- **Hyper-parameters.** Important hyper-parameters include the number of neighbours $k \in \{3, 5, 7, 11, 21\}$, the weighting scheme (uniform vs. distance-weighted),

the distance metric, and preprocessing choices (normalisation, dimensionality reduction). Fairness-aware variants can re-weight neighbours or introduce debiasing mechanisms (13).

- **Mutant kill signature.** K-NN is highly sensitive to feature-noise mutants because distances are directly altered by perturbations. Small changes in feature values can change neighbourhood composition, flipping predictions and fairness outcomes. This makes K-NN an informative stress test for fairness audits under local distributional shifts.

The capacity ladder across our models is roughly: Logistic Regression (low), Decision Tree (medium), and k-NN (high). As model flexibility increases, the importance of rigorous mutation-based adequacy assessment also grows.

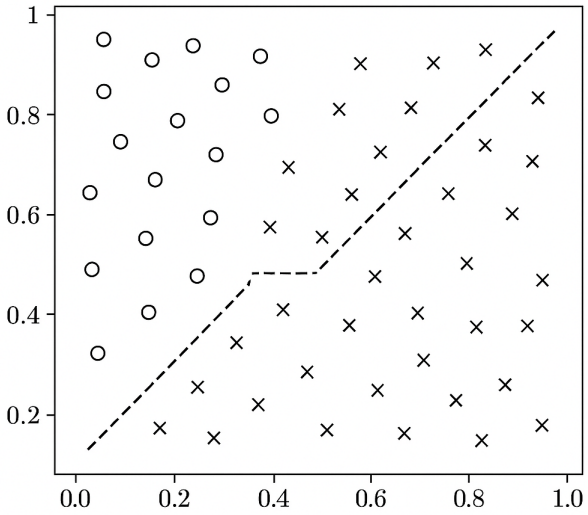

Figure 2.1: Decision boundary induced by *K-N*. Local neighborhoods determine classification, which can magnify clustered bias.

Figure 2.4: Decision boundary induced by K-NN. Local neighbourhoods determine classification, which can magnify clustered bias.

k-NN Prediction Algorithm

For completeness, Algorithm 1 summarises the standard K-NN prediction procedure.

Algorithm 1: K-NN prediction

Input : Training set $\mathcal{D}$, query x_*, integer k, distance $\mathrm{dist}(\cdot,\cdot)$

Output: Predicted label $\hat{y}_*$

$N_k \leftarrow \arg\min_{\mathcal{S}\subseteq\mathcal{D},|\mathcal{S}|=k} \sum_{(x,y)\in\mathcal{S}} \mathrm{dist}(x, x_*)$;

return `majority-vote`$\{y : (x, y) \in N_k\}$;

Table 2.1: Big-O summary (n samples, d features).

	Training	Query	Memory
k-NN	$O(1)$	$O(nd)$	$O(nd)$
Decision Tree	$O(nd \log n)$	$O(\log n)$	$O(\#\text{nodes})$
Logistic Regression	$O(nd^2)$	$O(d)$	$O(d)$

2.5 Mutation Testing for Fairness

Mutation testing, adapted from software engineering, evaluates test adequacy by introducing controlled changes (mutants) and checking whether the test suite detects them. In our setting, the "program" is the trained ML model and its decision pipeline, while the "test suite" consists of fairness metrics applied to model outputs. Mutations correspond to controlled perturbations of data, features, labels, or thresholds that simulate realistic distributional shifts. A fairness audit is considered adequate if it reliably detects fairness-relevant changes introduced by these mutants.

Traditional fairness evaluation computes metrics such as SPD, EOD, and EOM on a single static test set. However, real deployments are dynamic: input distributions drift, operating thresholds are tuned, and features evolve. A fairness audit that appears acceptable on one snapshot may fail under modest perturbations. Mutation

testing provides a principled way to stress-test fairness audits before deployment, helping ensure that fairness violations are not silently overlooked (2).

2.5.1 Core Concepts

At a high level, our fairness-oriented mutation testing framework consists of three components:

- **Mutants:** Perturbations to data, features, labels, or thresholds that simulate realistic shifts.
- **Fairness oracle:** Metrics such as SPD, EOD, and EOM applied to model outputs on mutated data.
- **Kill rule:** A mutant is *killed* if fairness metrics exceed bootstrap-derived thresholds; otherwise it is considered *alive*.

This framework extends fairness evaluation beyond static conditions, probing the robustness of fairness audits under plausible changes.

2.5.2 Kill/Alive Decision Rule

We now formalise the kill/alive rule. For a fairness metric $f \in \{SPD, EOD, EOM\}$, we compare the metric under mutation to the baseline:

$$\Delta f_m = f_m - f_{\text{base}}.$$

Using $B = 1000$ bootstrap resamples of the test set, we estimate the null distribution $(\mu_{\text{null}}, \sigma_{\text{null}})$ under no change. A mutant m is **killed** if

$$|\Delta f_m| \;\geq\; \mu_{\text{null}} + 1.5\,\sigma_{\text{null}} \;=\; \theta_f,$$

and is otherwise **alive**. Killed mutants indicate that the fairness audit successfully detected the perturbation; alive mutants suggest potential inadequacy.

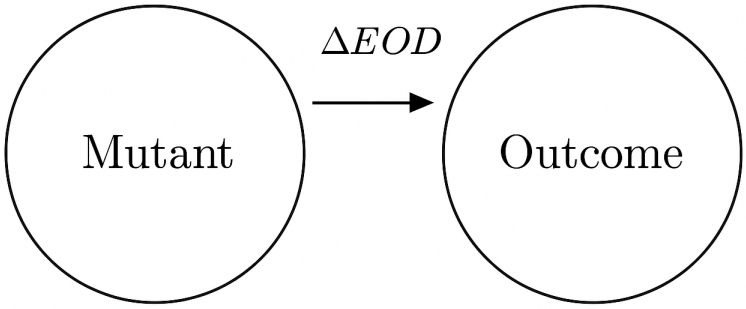

Figure 2.2: Fairness mutation lifecycle. Killed: $|\Delta EOD| \geq \theta$. Alive: below threshold (possibly equivalent).

Figure 2.5: Fairness mutation lifecycle. Mutants are generated, evaluated by fairness metrics, and classified as *killed* or *alive* depending on whether the change exceeds the threshold θ_f.

As a simple illustrative example, suppose the blue group has TPR = 0.80 and the red group has TPR = 0.62. The resulting opportunity gap is 0.18. If the bootstrap threshold for the relevant fairness metric is $\theta = 0.10$, then the audit detects this change and the mutant is *killed*. If the difference were only 0.05, the mutant would survive, signalling that the audit may miss subtle but still meaningful disparities.

By systematically applying mutation testing to fairness audits, we obtain a measure of *fairness adequacy*. This moves beyond single-snapshot evaluation and assesses whether fairness metrics remain responsive under realistic perturbations. A fairness audit with high mutation kill rates can be considered adequate; an audit with many surviving mutants is inadequate, even if static metrics look acceptable.

2.6 Summary of Background Needs

This chapter has established:

1. The societal and technical motivation for fairness and adequacy in ML.
2. The ways in which common models (Logistic Regression, Decision Trees, k-NN) are vulnerable to fairness faults.
3. The rationale for using mutation testing as a principled adequacy measure for fairness.

The next chapter translates these ingredients into the Fairness Adequacy Testing (FAT) framework and describes the mutation operators and guardrails that implement the kill rule.

Chapter 3

Method: Fairness-Oriented Mutation Testing

3.1 Overview

This chapter presents a fairness-oriented mutation testing method designed to assess whether a test suite is *adequate* to detect fairness problems in supervised classification models. Traditional mutation testing focuses on correctness under code or data perturbations; here, the central objects are *fairness metrics*.

The main idea is simple. We introduce small but realistic changes (*mutants*) to the training data or pipeline, retrain the same model specification, and then measure how much group fairness metrics move on a fixed test set. If the change in fairness exceeds a data-driven threshold, the mutant is **KILLED**; otherwise, it is **ALIVE**. Aggregating kill decisions yields a *Fairness Mutation Kill Rate* (mutation kill rates (MKR)), which we interpret as a fairness-oriented test adequacy score. Uncertainty in MKR is reported via bootstrap confidence intervals.

The method is intended for high-stakes classification problems in dynamic domains, where input distributions, thresholds, and feature definitions may shift over time. Although the empirical study focuses on the Adult and COMPAS datasets, the procedure is model-agnostic and can be applied to other tasks.

3.2 Data, Models, and Protected Attributes

Let $\mathcal{D} = \{(x_i, y_i, a_i)\}_{i=1}^{n}$ denote labelled instances with features x_i, binary labels $y_i \in \{0, 1\}$, and a protected attribute a_i (binary or multi-class; for example, gender, race, or client segment). The dataset is split into training and test sets. A baseline model M_{base} is trained on the training split and evaluated on a held-out test set $\mathcal{S} \subset \mathcal{D}$ that remains fixed for all mutants.

Predictions on the test set are either hard labels $\hat{y}_i = M(x_i) \in \{0, 1\}$ or calibrated scores $\hat{p}_i \in [0, 1]$ thresholded at some decision level τ. Group fairness metrics are always computed on $\mathcal{S}$ and stratified by the protected attribute A.

Throughout this thesis we evaluate the three base learners introduced in Chapter 2—Logistic Regression LR, Decision Trees CART, and k-Nearest Neighbours K-NN—and monitor the fairness metrics SPD, EOD, and EOM. The same model templates, hyperparameter grids, and test sets are re-used for all mutants so that differences can be attributed to the mutations themselves.

For a given configuration, let M_{eff} denote the number of *non-equivalent* mutants, that is, mutants that are capable of changing fairness on the test set after passing the validity checks described in Section 3.8. The main adequacy statistic is the Fairness Mutation Kill Rate MKR,

$$\text{MSR} = \frac{\#\,\text{Killed Mutants}}{\#\,\text{Total Mutants} - \#\,\text{Non-Equivalent Mutants}} = \frac{\#\,\text{KILLED}}{M_{\text{eff}}},$$

Higher MKR values indicate that the test suite is more sensitive to fairness-relevant changes and is therefore more adequate from a fairness perspective.

3.3 Methodology Workflow

The fairness mutation procedure follows an iterative workflow that combines group fairness metrics with mutation testing. The steps are summarised in Figure 3.1.

1. **Select dataset and protected attributes.** Choose a classification dataset that contains at least one relevant protected attribute (for example, gender, race, age, or socioeconomic status). This dataset provides the basis for both model training and fairness evaluation.

2. **Generate candidate mutants.** Apply mutation operators to the training data or pipeline to create candidate mutants. These operators include instance and feature removal, reweighting, feature addition, and pipeline variations. The goal is to simulate realistic changes that could affect group fairness. The full operator catalogue is given in Section 3.7.

3. **Train models on original and mutated data.** Train the baseline model M_{base} on the original training data and, for each valid mutant, retrain the same model specification on the mutated training data. Each mutant model M_m is evaluated on the fixed test set $\mathcal{S}$.

4. **Compute fairness metrics.** For the baseline and each mutant model, compute group fairness metrics such as Statistical Parity Difference, Equal Opportunity Difference, and Equalized Odds Metric. These metrics provide a quantitative measure of how outcomes differ across protected groups (Section 3.5).

5. **Decide whether the mutant is KILLED or ALIVE.** For each mutant, compare changes in fairness metrics against bootstrap-based thresholds derived from the baseline model (Section 3.4.2). If any monitored fairness metric

moves outside its control limit, the mutant is labelled **KILLED**; otherwise it is **ALIVE**. This implements the kill rule described in Section 3.4.1.

6. **Update adequacy estimate.** After discarding unrealistic or fairness-neutral mutants via the validity gate (Section 3.8), aggregate kill decisions into the MKR statistic. When MKR is low, the test suite is missing fairness-relevant shifts; when MKR is high, the test suite is more adequate for fairness.

7. **Iterate if needed.** If MKR remains unsatisfactory, extend the test suite (for example, by adding more slices, metrics, or targeted mutants) and repeat the process until a chosen adequacy level is reached.

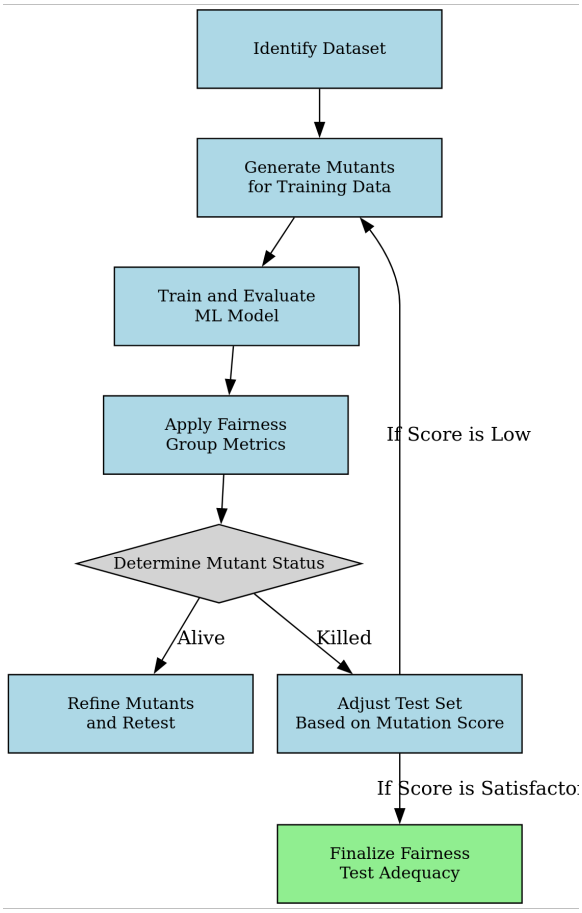

Figure 3.1: Iterative fairness test adequacy process for machine learning models.

3.4 Fairness-Oriented Mutation Testing Framework

This section formalises the proposed fairness-oriented mutation testing framework. The core elements are:

- a family of **mutation operators** that induce plausible changes in training data or pipeline configuration;
- a **fairness oracle** that evaluates group fairness metrics on a fixed test set;
- a **kill rule** that classifies mutants as KILLED or ALIVE based on bootstrap thresholds for fairness deltas; and
- an **adequacy score** (MKR) that aggregates kill decisions across non-equivalent mutants.

The framework is model-agnostic: it applies in the same way to K-NN, Decision Trees, and Logistic Regression, provided that the same model specification is retrained on each mutated dataset.

3.4.1 Mutation Lifecycle and Kill / Alive Decision

Figure 3.2 depicts the lifecycle of a fairness mutant. The procedure begins with a base model trained on clean data, proceeds through mutant creation and retraining, and ends with a fairness-based decision on whether the mutant is killed.

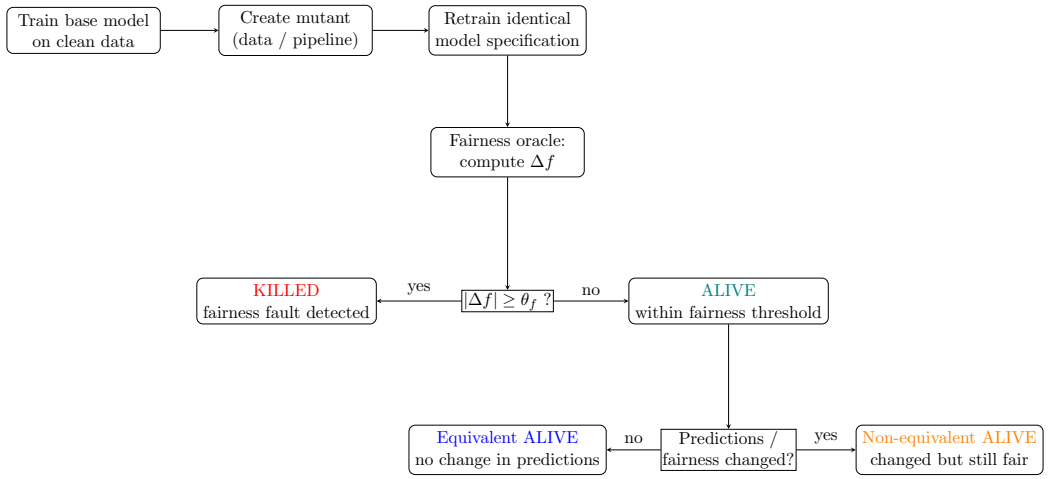

Figure 3.2: Fairness mutation lifecycle with post-hoc classification of **ALIVE** mutants into equivalent (no prediction change) and non-equivalent (prediction/fairness changes remain within the accepted threshold).

Concretely, for each dataset and base learner, the lifecycle is:

1. Train a **base model** on the original training set.
2. Compute baseline fairness metrics on the fixed test set $\mathcal{S}$.
3. Estimate per-metric control limits θ_f using bootstrap (Section 3.4.2).
4. Generate candidate mutants using the operators in Section 3.7.
5. Filter mutants through the **validity gate** (Section 3.8); discard unrealistic or fairness-neutral mutants.
6. Retrain the same model specification on each valid mutant and compute the fairness deltas Δf_m on $\mathcal{S}$.
7. Label each mutant as **KILLED** or **ALIVE** according to whether any fairness delta exceeds its control limit, and aggregate these decisions into MKR (Section 3.9).

3.4.2 Bootstrap Threshold Estimation

The kill decision should not react to minor fluctuations due to sampling noise. For each fairness metric f, we therefore estimate a *kill threshold* θ_f using a bootstrap-based null distribution under the base model.

Let f_{base} denote the value of metric f on the base model and test set $\mathcal{S}$.

1. Keep the base model M_{base} fixed.
2. Draw B bootstrap resamples of the test set, $\{\mathcal{S}^{(b)}\}_{b=1}^{B}$, by sampling with replacement.
3. For each resample, compute $f_{\text{base}}^{(b)}$ and define $\Delta f^{(b)} = f_{\text{base}}^{(b)} - f_{\text{base}}$.

4. Let μ_{null} and σ_{null} be the mean and standard deviation of the bootstrap deltas $\{\Delta f^{(b)}\}_{b=1}^{B}$.

The control limit for metric f is then

$$\boxed{\theta_f = \mu_{\text{null}} + 1.5\,\sigma_{\text{null}}.}$$

The multiplier 1.5 is chosen as a pragmatic balance between sensitivity (detecting meaningful fairness changes) and specificity (ignoring small fluctuations). In the experiments we use $B = 1000$ bootstrap rounds by default.

A mutant m is labelled **KILLED** if

$$|\Delta f_m| \geq \theta_f$$

for *any* monitored fairness metric or subgroup slice; otherwise it is **ALIVE**. Here $\Delta f_m = f_m - f_{\text{base}}$ is the change in fairness metric f between the mutant and the base model.

3.5 Fairness Metrics and Headline Score

3.5.1 Group Fairness Metrics

We track three standard group fairness criteria for a binary decision $\hat{Y}$, protected attribute A, and groups a and b. These metrics were introduced conceptually in Chapter 2; they are restated here in the notation used by the mutation framework.

1. **Statistical Parity Difference SPD.** This metric measures differences in positive outcome rates across groups:

$$\text{SPD} = P(\hat{Y} = 1 \mid A = a) - P(\hat{Y} = 1 \mid A = b).$$

The ideal value is 0, meaning equal positive prediction rates.

2. **Equal Opportunity Difference EOD.** Equal opportunity requires that qualified individuals receive the positive outcome at the same rate, regardless of group membership:

$$\mathrm{EOD} = \mathrm{TPR}_{\mathrm{privileged}} - \mathrm{TPR}_{\mathrm{unprivileged}},$$

where $\mathrm{TPR} = P(\hat{Y} = 1 \mid Y = 1, A)$. The ideal value is again 0.

3. **Equalized Odds Metric EOM.** Equalized odds requires equality of both true positive rates and false positive rates:

$$\mathrm{TPR}_{\mathrm{privileged}} = \mathrm{TPR}_{\mathrm{unprivileged}}, \quad \mathrm{FPR}_{\mathrm{privileged}} = \mathrm{FPR}_{\mathrm{unprivileged}}.$$

We operationalise this via

$$\mathrm{EOM} = \max\bigl\{|\mathrm{TPR}_a - \mathrm{TPR}_b|, |\mathrm{FPR}_a - \mathrm{FPR}_b|\bigr\},$$

with ideal value 0.

As an illustrative example, suppose the blue group has $\mathrm{TPR} = 0.80$ and the red group has $\mathrm{TPR} = 0.62$. Then

$$\mathrm{EOD} = 0.80 - 0.62 = 0.18.$$

If the bootstrap threshold is $\theta_{\mathrm{EOD}} = 0.10$, the base model already violates the EOD tolerance. In this setting, any mutant that *further widens* the TPR gap beyond 0.10 is treated as a fairness fault and is therefore **KILLED** by the EOD oracle.

3.5.2 Headline Fairness Score

To summarise multiple fairness criteria in a single index, we use a Weighted Multi-Objective Fairness Index (WMFI) in $[0, 1]$, where higher values indicate better fairness.

For each sensitive attribute we compute:

- Demographic Parity gap $\text{DPgap} = \max_g PR_g - \min_g PR_g$,
- Equal Opportunity gap $\text{EOgap} = \max_g TPR_g - \min_g TPR_g$,
- Equalized Odds gap $\text{EODgap} = \max\big(\text{EOgap}, \max_g FPR_g - \min_g FPR_g\big)$,
- Disparate Impact $\text{DI} = \min_g PR_g / \max_g PR_g$,

where PR_g denotes the positive prediction rate in group g.

The WMFI is then defined as

$$\text{WMFI} = 1 - \big(w_{DP}\,\text{DPgap} + w_{EO}\,\text{EOgap} + w_{EOD}\,\text{EODgap}\big) - \lambda \cdot \mathbf{1}[\text{DI} < \tau],$$

with weights $(w_{DP}, w_{EO}, w_{EOD}) = [0.30, 0.40, 0.30]$, disparate impact threshold $\tau = 0.80$, and penalty $\lambda = 0.05$. WMFI is used as a headline score in the empirical results; mutation testing, however, is always performed on the component metrics themselves.

3.6 Mutation Design

Mutations can be applied at several levels of the learning process. We consider three broad categories:

Data-level: controlled perturbations to inputs, such as feature noise, time-window shifts, synthetic outliers, group reweighting, or removal of specific instances.

Pipeline-level: changes to preprocessing and sampling procedures, such as swapping normalisation schemes, altering imputation strategies, or modifying resampling settings.

Model-level: limited modifications to model training, such as adjusting regularisation strength, decision thresholds, or class weights.

Each mutation operator ϕ transforms the training data and/or training procedure to produce a mutated model M_ϕ. Operators are parameterised to keep mutations plausible and to avoid unrealistic scenarios that would not arise in normal operation.

3.7 Mutation Operators

This section describes the concrete mutation operators used in the Fairness Adequacy Test (FAT) framework. Operators are grouped into four main families: removal, addition, permutation, and shuffling. Together they instantiate the data- and pipeline-level mutations described above.

3.7.1 Removal Operators

Removal operators delete parts of the dataset to stress-test reliance on specific instances, groups, or features.

Instance Removal

Instance removal eliminates selected records, especially from minority or protected groups, to mimic data loss or under-representation.

- **Binary removal.** For a binary sensitive attribute (for example, gender), create mutants by dropping all instances from one group at a time:

  - Mutant 1: remove all cases with gender = female.
  - Mutant 2: remove all cases with gender = male.

- **Multi-value removal.** For a multi-valued attribute (for example, race), create one mutant per value:
  - Mutant 1: remove all cases with race = Black.
  - Mutant 2: remove all cases with race = White.
  - Mutant 3: remove all cases with race = Hispanic.
  - Mutant 4: remove all cases with race = Asian.

- **Combinatorial removal.** For multiple sensitive attributes, remove intersectional combinations:
  - Mutant 1: remove all cases with gender = male and race = Black.
  - Mutant 2: remove all cases with gender = male and race = White.
  - Mutant 3: remove all cases with gender = female and race = Black.
  - Mutant 4: remove all cases with gender = female and race = White.

Feature Removal

Feature removal studies how the model behaves when certain pieces of information are absent.

- **Single feature removal.** Remove one feature entirely, such as the "sex" attribute.

- **Multiple feature removal.** Remove features one at a time to create independent mutants, for example: gender, race, income, or education level.

- **Combinatorial feature removal.** Remove feature pairs such as:
  - gender and race;
  - gender and income;
  - race and education level;
  - income and education level.

3.7.2 Addition Operators

Addition operators create new features synthesised from existing ones.

- **Linear combinations.** Create a new feature as a linear combination, e.g.

$$\text{new\_feature} = a \cdot \text{age} + b \cdot \text{years of education},$$

  with constants a and b.

- **Ratios.** Create ratio-based features such as

$$\text{new\_feature} = \frac{\text{age}}{\text{years of education}}.$$

Instance Addition

For intersectional fairness coverage, new synthetic instances are added by generating combinations of sensitive attribute values (for example, all gender–race combinations) in areas where the original data are sparse.

3.7.3 Permutation Operators

Permutation operators reassign values within the dataset to disrupt patterns on which the model may be relying.

Feature Permutation

Randomly permute the values of a single feature across instances. This breaks the link between that feature and the label, while preserving its marginal distribution.

Row Permutation

Shuffle the order of rows in the training data to remove potential artefacts from batch structure or temporal ordering.

3.7.4 Shuffling Operators

Shuffling operators further test the model's dependence on particular associations.

Feature Shuffling

Randomly reassign values within a feature column, similar to feature permutation, to stress-test feature importance and proxy effects.

Label Shuffling

Shuffle labels among samples, disrupting the association between features and labels. This checks whether the model is learning genuine patterns or simply memorising the training set; such mutants are usually filtered by the validity gate if they become too unrealistic.

3.7.5 Link to Data-, Pipeline-, and Model-Level Mutations

The concrete operators above instantiate the broader mutation categories:

- **Data-level mutations:** instance and feature removal or addition, permutation, shuffling, label flips, feature noise, intersectional resampling, and missing-

ness injection.

- **Pipeline-level mutations:** threshold slides, encoder swaps, scaler swaps, and class-weight changes.
- **Model-level mutations:** small changes to regularisation or removal of fairness-critical features.

These operators are always combined with the validity gate in the next section to maintain realism.

3.8 Validity Gate (Pre-Filter)

Not every syntactically valid mutant is meaningful for fairness analysis. Before a mutant contributes to MKR, it must pass two checks.

1. **Distribution similarity.** We compare the original and mutated training distributions using two-sample tests:

   - Kolmogorov–Smirnov tests for numeric features, and
   - χ^2 tests for categorical features.

   Mutants must satisfy $p \geq 0.05$ (or remain below a chosen effect-size threshold) for all monitored features to be considered within a realistic shift regime.

2. **Training-side fairness shift.** We require that the mutation moves the fairness signal on the training data. Let Δf^{train} be the change in a fairness metric on the training set, and let $(\mu^{\text{train}}_{\text{null}}, \sigma^{\text{train}}_{\text{null}})$ denote its bootstrap null statistics. We require

$$|\Delta f^{\text{train}}| \geq \mu^{\text{train}}_{\text{null}} + \sigma^{\text{train}}_{\text{null}}.$$

Mutants failing either check are treated as *equivalent* in practice and are excluded from the MKR denominator.

3.9 Mutation Score and Uncertainty

Let M denote the number of mutants that pass the validity gate. For each such mutant m, let $\mathbf{1}\{m \text{ is KILLED}\}$ be an indicator that the fairness oracle has detected a fault. The Fairness Mutation Kill Rate is

$$\boxed{\text{MKR} = \frac{1}{M} \sum_{m=1}^{M} \mathbf{1}\{|\Delta f_m| \geq \theta_f \text{ for some } f\},}$$

where the condition "for some f" ranges over all monitored metrics and slices. Mutants rejected by the validity gate do not enter M.

To quantify uncertainty, we estimate a 95% percentile bootstrap confidence interval for MKR by resampling the set of mutants with replacement. We also compute MKR broken down by slice (for example, per sensitive attribute or per operator type) to see which groups or mutation families contribute most to the detections. An adjusted score MKR_{adj} can further exclude mutants with very small training-side fairness changes, producing a more conservative adequacy judgement.

3.10 Worked Mini-Example (Trading Classifier)

To make the procedure more concrete, consider a binary trading signal model that predicts $\hat{y} \in \{0, 1\}$ ("execute trade" or "do not execute"). The protected attribute is client segment $a \in \{\text{retail}, \text{institutional}\}$. We track Equal Opportunity Difference as the main fairness metric.

1. **Bootstrap.** Train the base model M_{base} on a rolling window and compute EOD

on $\mathcal{S}$. Using $B = 1000$ bootstrap resamples of $\mathcal{S}$ under the fixed base model, obtain $(\mu_{\text{EOD}}, \sigma_{\text{EOD}})$ and set $\theta_{\text{EOD}} = \mu_{\text{EOD}} + 1.5\sigma_{\text{EOD}}$.

2. **Mutations.** Generate mutants such as:

   - shifting the input time window by +1 day;
   - adding small Gaussian noise to volatility features;
   - reweighting the minority segment by a factor of 1.2;
   - swapping z-score scaling with robust scaling; and
   - changing missing-value imputation from median to k-NN.

3. **Validity gate.** Reject mutants that produce a large AUC drop (for example, > 0.03) or extreme feature mean shifts (for example, $> 2\sigma$), or that fail the fairness-shift check on the training data.

4. **Assessment.** For each valid mutant, compute ΔEOD on $\mathcal{S}$ and apply the kill rule. Aggregate decisions into MKR with a 95% confidence interval.

This miniature scenario mirrors the structure of the main experiments and shows how often plausible operational changes could materially worsen fairness across client segments.

3.11 Implementation Pseudocode

Algorithm 2 gives high-level pseudocode for the fairness mutation testing pipeline. The same procedure is applied to all datasets and base learners, with shared fairness metrics and thresholds.

Algorithm 2: Fairness Mutation Testing Pipeline

Input : Training data $\mathcal{D}_{\text{train}}$; test set $\mathcal{S}$; model template $\mathcal{M}$; mutant generator $\mathcal{G}$; fairness metrics $\mathcal{F}$; threshold multiplier $z = 1.5$; bootstrap rounds $B = 1000$

Output: Mutation Kill Rate (MKR) and 95% confidence interval

`/* Step 1: Bootstrap null distribution for fairness metrics */`

Train base model $M_{\text{base}} \leftarrow \mathcal{M}(\mathcal{D}_{\text{train}})$;

Compute baseline fairness metrics $f_{\text{base}} \leftarrow \text{FairnessOracle}(M_{\text{base}}, \mathcal{S}, \mathcal{F})$;

for $b = 1$ **to** B **do**

  $\mathcal{S}^{(b)} \leftarrow$ bootstrap sample of $\mathcal{S}$;

  $f_{\text{base}}^{(b)} \leftarrow \text{FairnessOracle}(M_{\text{base}}, \mathcal{S}^{(b)}, \mathcal{F})$;

  $\Delta f^{(b)} \leftarrow f_{\text{base}}^{(b)} - f_{\text{base}}$;

end

Compute $\mu_{\text{null}}, \sigma_{\text{null}}$ over $\{\Delta f^{(b)}\}$ for each $f \in \mathcal{F}$;

Set threshold $\theta_f \leftarrow \mu_{\text{null}} + z \cdot \sigma_{\text{null}}$ for each $f \in \mathcal{F}$;

$M \leftarrow 0$; kills $\leftarrow 0$;

/* Step 2: Generate, filter, and score mutants */

for *mutant dataset* $\mathcal{D}_{train}^{(m)}$ *in* $\mathcal{G}(\mathcal{D}_{train})$ **do**

  if *validity gate fails for* $\mathcal{D}_{train}^{(m)}$ **then**

    ; // Discard equivalent or unrealistic mutant

  end

  $M \leftarrow M + 1$;

  $M_m \leftarrow \mathcal{M}(\mathcal{D}_{\text{train}}^{(m)})$;

  $f_m \leftarrow \text{FairnessOracle}(M_m, \mathcal{S}, \mathcal{F})$;

  $\Delta f_m \leftarrow f_m - f_{\text{base}}$;

  if $\exists f \in \mathcal{F}$ *such that* $|\Delta f_m(f)| \geq \theta_f$ **then**

    kills $\leftarrow$ kills +1;

  end

end

Compute MKR $\leftarrow$ kills/M;

Estimate 95% bootstrap confidence interval for MKR by resampling $\{1, \ldots, M\}$ with replacement;

return MKR and its confidence interval;

Algorithm 2 turns the conceptual framework into an operational procedure: it bootstraps fairness metrics to define thresholds, generates and filters mutants, and summarises their impact via MKR.

Research Questions

This chapter operationalises the research questions stated in Chapter 3 as follows:

RQ1. **(RQ1 – Effectiveness of mutation operators)** How effective are fairness-aware mutation operators? The chapter defines a family of operators, a validity

gate, and a kill/alive decision rule, allowing their effectiveness to be quantified via mutation kill rates and metric drift (Sections 3.3–3.7).

RQ2. **(RQ2 – Performance across models and datasets)** How consistently does the framework perform across models and datasets? The same mutation pipeline is applied to different base learners (K-NN, Decision Tree, Logistic Regression) and datasets (Adult and COMPAS) using shared fairness metrics and bootstrap thresholds (Sections 3.5 and 3.9).

RQ3. **(RQ3 – Comparison to existing fairness testing baselines)** How does the proposed framework compare to existing fairness testing baselines? The method is positioned against static group fairness audits and headline fairness scores, providing a mutation-based adequacy perspective that can be compared to toolkits such as Fairlearn, AIF360, and FairTest (Sections 3.4–3.9 and Chapter 4).

In summary, Chapter 3 has set out the methodological foundation of the Fairness Adequacy Testing framework. Having defined the mutation lifecycle, fairness metrics, validity gate, and adequacy thresholds, the next chapter instantiates these elements on the Adult and COMPAS datasets, applies them across the base learners introduced in Chapter 2, and reports the resulting MKR and WMFI values to address the research questions in practice.

Chapter 4

Fairness Metrics and Empirical Evaluation

4.1 Purpose of this Chapter

This chapter has two main jobs. First, it explains the fairness metrics used in this thesis in a clear and practical way. Second, it shows how these metrics behave in our experiments when we apply the Fairness Mutation Testing (FAT) framework to real datasets.

Building on Chapter 3, we now run the framework on two benchmark datasets (Adult and COMPAS) and three base learners (Logistic Regression LR, Decision Tree CART, and k-NN K-NN). We:

- define and motivate the group-fairness metrics (SPD, EOD, EOM),
- introduce a single headline fairness score, the Weighted Multi-Objective Fairness Index (Weighted Multi-Objective Fairness Index (WMFI)),
- walk through a concrete numeric example on the Adult dataset,
- describe the experimental setup for Adult and COMPAS,
- report baseline accuracy and fairness,
- study how fairness metrics react to fairness drift thresholds,

- examine how metrics move under mutation-based stress testing,
- evaluate how effective the fairness-aware mutation operators are across models and datasets, and
- compare mutation-based adequacy with a diameter-based diversity baseline.

Where detailed tables or extended plots were too long for the main text, we provide them in a public repository so that others can inspect and reuse them.

4.2 Metric Definitions and Notation

We focus on a standard binary classification setting:

- The protected attribute $A \in \{a, b\}$ represents group membership (for example, Female vs. Male, or Black vs. White).
- The model output $\hat{Y} \in \{0, 1\}$ is a binary decision (e.g. accept vs. reject, low risk vs. high risk).
- The ground-truth label $Y \in \{0, 1\}$ is the true outcome.

All fairness rates are computed on the same fixed test set $\mathcal{S}$, in line with the mutation-testing protocol described in Chapter 3.

Equal Opportunity vs Equalized Odds

Because two related metrics appear throughout this chapter, we clarify their difference up front:

- **Equal Opportunity** focuses on qualified individuals. It asks: "Among people who truly qualify (e.g. $Y = 1$), do different groups get the positive decision at similar rates?" It compares *true positive rates*.

- **Equalized Odds** is stricter. It looks at both *true positive rates* and *false positive rates*. It asks: "Are both qualified people helped equally, and unqualified people treated similarly, across groups?"

In this thesis we operationalise these ideas as:

- EOM: an Equal Opportunity-style metric that captures disparity in true positive rates (True Positive Rate (TPR)),
- EOD: an Equalized Odds-style metric that captures the largest disparity across both TPR and False Positive Rate (FPR).

This distinction matters when we interpret the graphs. In both Adult and COMPAS, EOD tends to be a stricter test than EOM, because EOD penalises unfairness in both "who we help" and "who we mistakenly harm".

4.2.1 Composite Fairness Index (Scalar Measure)

In practice, it is often useful to have a single number that summarises several fairness checks. For this, we define the Weighted Multi-Objective Fairness Index (WMFI). WMFI takes values between 0 and 1, where 1 means "best" (from a fairness point of view).

Key concepts

- **Demographic Parity Disparity Parity (DP):** A model satisfies demographic parity if the probability of receiving a positive prediction is the same across protected groups.

- **Disparate Impact (DI):** The ratio of positive prediction rates between groups. Under the common "four-fifths rule", ratios below 0.80 are treated as red flags for potential discrimination.

Gap measures

For each sensitive attribute we compute:

- **DPgap:**

$$\text{DPgap} = \max_g PR_g - \min_g PR_g,$$

  where PR_g is the positive prediction rate for group g.

- **EOgap:** the gap between the highest and lowest true positive rates across groups.

- **EODgap:** the maximum between EOgap and the corresponding false positive rate gap.

- **Disparate Impact:**

$$\text{DI} = \frac{\min_g PR_g}{\max_g PR_g}.$$

Here, DI tells us how the least-advantaged group compares to the most-advantaged group in terms of positive predictions.

Illustrative example

Suppose a hiring model yields:

Group	**Positive Prediction Rate (PR_g)**
Group A (Female)	0.60
Group B (Male)	0.45

Then:

- $\max_g PR_g = 0.60$, $\min_g PR_g = 0.45$,
- DPgap $= 0.15$,
- DI $= 0.45/0.60 = 0.75 < 0.80$, so it fails the four-fifths rule.

WMFI Formula and Interpretation

We combine the gap measures into a scalar score:

$$\text{WMFI} = 1 - \big(w_{DP}\,\text{DPgap} + w_{EO}\,\text{EOgap} + w_{EOD}\,\text{EODgap}\big) - \lambda \cdot \mathbf{1}[\text{DI} < \tau],$$

with

$$(w_{DP}, w_{EO}, w_{EOD}) = [0.30,\ 0.40,\ 0.30], \quad \tau = 0.80, \quad \lambda = 0.05.$$

The penalty term $-\lambda \cdot \mathbf{1}[\text{DI} < \tau]$ is triggered when DI falls below the legal-style tolerance threshold $\tau = 0.80$. This reflects the fact that a bad DI ratio is often taken more seriously in regulatory contexts than a small change in a gap.

In short: WMFI gives a single "fairness score" that can be compared across models and datasets, while still being grounded in clear components (DPgap, EOgap, EODgap, and DI).

4.3 Worked Numeric Example on Adult Dataset

To make the metrics more concrete, we walk through a simple example on the Adult test set. We use *sex* as the protected attribute (Female $= a$, Male $= b$), and consider a logistic regression model with a fixed threshold of 0.5.

Positive prediction rates:

Table 4.1: Confusion matrices by sex – Adult test set (example).

	Female (a)		Male (b)	
	$\hat{Y}=1$	$\hat{Y}=0$	$\hat{Y}=1$	$\hat{Y}=0$
$Y=1$	803	547	2582	1094
$Y=0$	423	2227	1015	5289

- Positive rates:

$$P(\hat{Y}=1 \mid A=a) = \frac{803+423}{803+547+423+2227} = 0.322,$$

$$P(\hat{Y}=1 \mid A=b) = \frac{2582+1015}{2582+1094+1015+5289} = 0.311,$$

so

$$\text{SPD} = 0.322 - 0.311 = 0.011.$$

The model has almost no difference in overall selection rate by sex.

- Step 2: True-positive rates:

$$\text{TPR}_a = \frac{803}{803+547} = 0.595, \quad \text{TPR}_b = \frac{2582}{2582+1094} = 0.702,$$

So the Equal Opportunity Difference is

$$\text{EOD} = 0.595 - 0.702 = -0.107.$$

This means qualified males receive positive decisions about 10.7 percentage points more often than qualified females.

- Step 3: False-positive rates:

$$\mathrm{FPR}_a = \frac{423}{423 + 2227} = 0.160, \quad \mathrm{FPR}_b = \frac{1015}{1015 + 5289} = 0.161,$$

  so, False positive rates are very similar across groups

- Step 4: Equalized-Odds-style metric We define:

$$\mathrm{EOM} = \max\{|-0.107|,\ |0.160 - 0.161|\} = 0.107.$$

Take-away. Even though the selection rates (SPD) are almost equal, the true positive rates are not. This is a pattern that we will see again in the main experiments: for both Adult and COMPAS, SPD can look acceptable while EOD and EOM reveal much stronger fairness problems. This motivates the use of a richer fairness test, including mutation-based stress tests.

4.4 Experimental Setup

4.4.1 Datasets and Protected Attributes

We evaluate the framework on two widely used benchmark datasets:

- **Adult:** Contains demographic and employment information used to predict whether an individual's income exceeds \$50K. Protected attributes in our analysis include sex, race, and age group.

- **COMPAS:** Contains criminal history and demographic information used to predict two-year recidivism risk. Protected attributes include race, `age_category`, and sex. We acknowledge well-known concerns about label quality and historical bias in this dataset but treat it as a standard benchmark for fairness studies.

For both datasets, we split data into training and test partitions using a fixed random seed to ensure reproducibility. Where we tune hyper-parameters, we use cross-validation on the training data only. All fairness metrics are computed on the held-out test set.

4.4.2 Pre-processing and Feature Engineering

Pre-processing is kept as simple and consistent as possible across datasets:

- Missing values are imputed using simple strategies (e.g. mode for categorical, mean/median for numerical).
- Categorical variables are encoded (e.g. one-hot encoding).
- Features are scaled or normalised when required by the model class (for example, for K-NN and Logistic Regression).
- In a few cases we construct additional features that also serve as candidates for mutation operators (see Chapter 3).

All transformations are fit on the training data and then applied to the test data, to avoid leakage.

4.4.3 Models and Hyper-parameters

We use three base learners that represent different modelling families:

- Logistic Regression (LR),
- Decision Tree (CART),
- k-Nearest Neighbours (K-NN).

Hyper-parameters are selected using grid search with cross-validation, following the procedure in Chapter 2. The final configurations are kept fixed when running the mutation experiments, so that differences come from mutations rather than re-tuning.

Code and Reproducibility

All code used for data preparation, model training, mutation operators, and figure generation is available in a public GitHub repository:

<YOUR-GITHUB-LINK>

The repository also includes:

- configuration files for the experiments,
- scripts for reproducing the mutation runs, and
- extended tables and graphs that do not appear in this chapter.

This allows readers to inspect the full analysis, run the code on their own machine, and adapt the framework to new datasets.

4.5 Baseline Accuracy and Fairness

Before we stress-test fairness through mutations, we first look at the baseline picture. For each dataset–model pair we compute standard predictive metrics (Accuracy, AUC, F1) and then evaluate fairness via EOD and EOM on the test set.

These baseline fairness values serve as a fixed reference point. Later, when we look at fairness drift caused by mutants, we will compare back to these baseline levels.

For Adult, the two plots together show that:

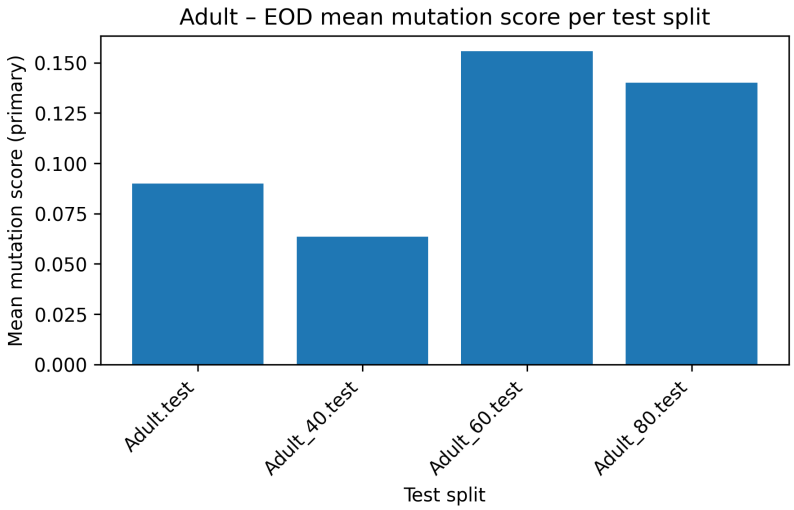

Figure 4.1: Adult dataset: baseline Equalized Odds Difference (EOD) by protected attribute.
X axis: protected attributes / groups.
Y axis: EOD values, summarising how strongly Equalized Odds is violated at baseline.

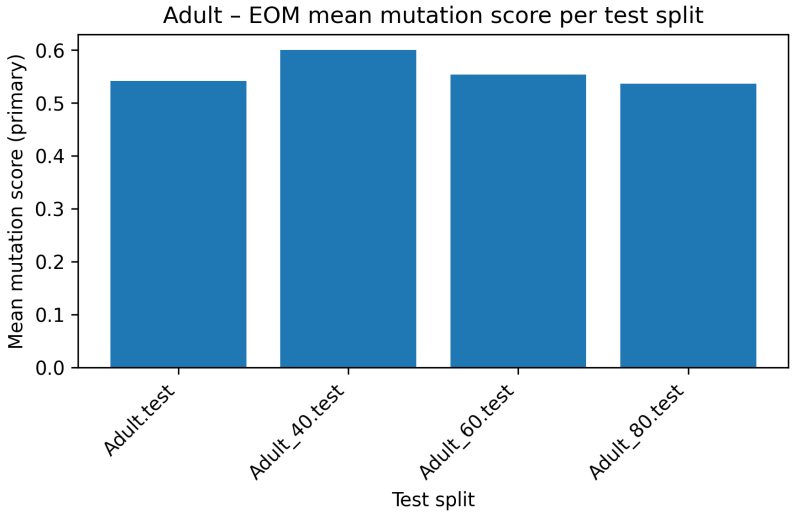

Figure 4.2: Adult dataset: baseline Equal Opportunity Metric (EOM) by protected attribute.
X axis: protected attributes / groups.
Y axis: EOM values, showing disparities in true positive rates at baseline.

- some protected attributes (for example, sex or race) have noticeably higher EOD and EOM values than others,
- EOD is often slightly larger than EOM, since it also counts false positive differences, not just true positives.

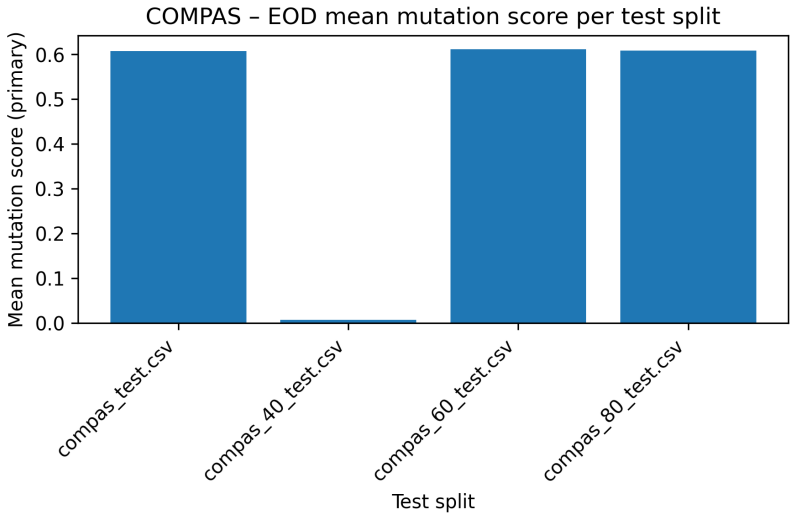

Figure 4.3: COMPAS dataset: baseline Equalized Odds Difference (EOD) by protected attribute.
X axis: protected attributes / groups.
Y axis: EOD values, highlighting where recidivism predictions depart most from Equalized Odds.

On COMPAS, the gaps are typically larger than on Adult, especially under EOD. This reflects two differences:

- the task is higher risk (recidivism), so mistakes can have more impact,
- structural biases in the data lead to stronger violations of both opportunity and error-rate parity.

Comparing Adult and COMPAS side by side, and comparing EOD vs EOM, we see three recurring patterns that will also show up in the mutation graphs:

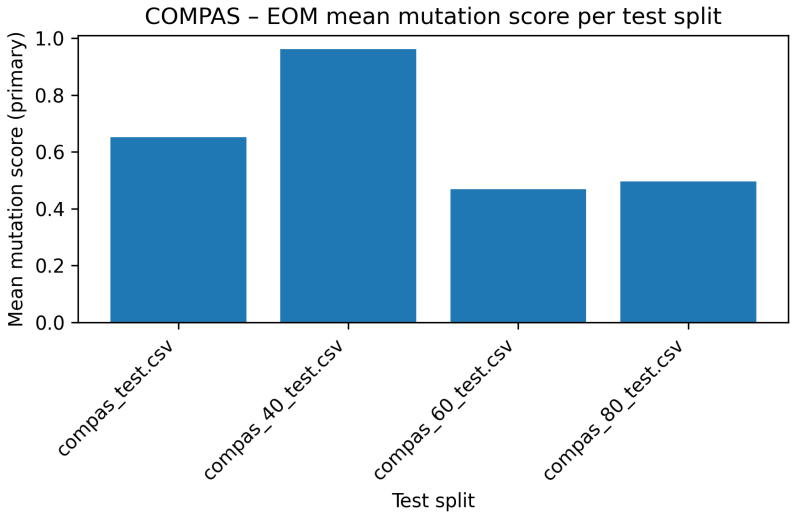

Figure 4.4: COMPAS dataset: baseline Equal Opportunity Metric (EOM) by protected attribute.
X axis: protected attributes / groups.
Y axis: EOM values, showing opportunity gaps at baseline.

1. COMPAS tends to show *larger* fairness gaps than Adult.

2. EOD is usually *stricter* than EOM, because it combines differences in both TPR and FPR.

3. For some protected attributes, the two metrics disagree: a group may look "moderately unfair" under EOM but much worse under EOD.

4.6 Threshold Sensitivity Analysis

In our experiments, the "threshold" we vary is not the classifier's decision threshold. Instead, it is a fairness *drift* threshold that decides when a mutant is considered **KILLED**.

For each metric f (for example, EOD), we look at the distribution of absolute drift values across all mutants and set:

$$\theta_f = \mu_\Delta + 1.5 \cdot \sigma_\Delta,$$

where μ_Δ and σ_Δ are the mean and standard deviation of the drift distribution. A mutant is **KILLED** if its fairness change exceeds this threshold. Otherwise it is **ALIVE**.

Among ALIVE mutants we then distinguish:

- **ALIVE–Equivalent:** fairness drift below 0.05. These mutants change almost nothing and are effectively noise.
- **ALIVE–Non-Equivalent:** fairness drift above 0.05 but below the threshold θ_f. These are fairness-relevant shifts that the current fairness tests do not flag as faults.

This structure matters later when we interpret the graphs: KILLED mutants show where our tests are strong, ALIVE–Non-Equivalent mutants show where they miss interesting fairness changes.

4.7 Metric Profiles Across Models

For each dataset and each base learner (K-NN, CART, LR), we estimate EOD and EOM across multiple random seeds. We then report the mean and a 95% bootstrap confidence interval to show how stable each metric is.

In the graphs (not reproduced here), we group bars by model family and dataset, and show both metrics:

- Models with *higher* EOD or EOM on average are less fair under that metric.
- Narrow confidence intervals indicate more stable fairness behaviour across seeds; wide intervals indicate that fairness is fragile to data or training noise.

Across the figures, two patterns emerge:

1. For both Adult and COMPAS, EOD tends to be a harsher test than EOM, because it adds false positives to the fairness picture.

2. The relative ordering of models can change by dataset and metric. For example, a model that looks "better" on Adult under EOM may not look better on COMPAS under EOD. This is why we later study adequacy by dataset–metric pair, rather than assuming one global ranking.

4.8 Metric Stability Under Mutation Stress

We now use the mutation operators from Chapter 3 to see how fairness metrics behave under realistic perturbations at the data and pipeline levels.

4.8.1 Metric Drift by Mutation Type

For each mutation, we compute:

$$|\Delta\text{EOD}|, \quad |\Delta\text{EOM}|.$$

We summarise the distributions using medians and interquartile ranges (IQRs), and report the proportion of mutants where the drift exceeds the detection threshold θ_f.

Intuitively:

- A large median drift means that metric is sensitive to that type of change.

- A high proportion of "above threshold" cases means that the tests frequently detect fairness issues.

4.8.2 Mutation Status: KILLED vs ALIVE

Using the rule from Chapter 3, each mutant is assigned to one of:

- **KILLED** — fairness drift exceeds the detection threshold.
- **ALIVE–Equivalent** — the mutant changes nothing important in predictions or fairness.
- **ALIVE–Non-Equivalent** — fairness changes appear but stay below the threshold.

The ALIVE–Non-Equivalent mutants are especially important. They show that fairness-relevant changes can slip through the current tests. The mutation scores defined below quantify how often this happens.

Figure 4.5 provides an overview of how well the fairness tests perform on COMPAS by compressing the KILLED/ALIVE breakdown into a single number per setting.

Let:

- M_{primary} be the number of fairness-related mutants,
- K_k be the number of KILLED mutants,
- E_e be the number of equivalent (effectively neutral) mutants.

We define the Fairness Adequacy Testing score as:

$$\text{FAT} \;=\; \frac{K_k}{M_{\text{primary}} - E_e} \in [0, 1].$$

Here:

- FAT = 1 means every non-equivalent mutant was detected.

Data_name	Algorithm_type	Metric	M_primary	Killed_k	Equivalent_e	Mutation_score
compas_test.csv	Logistic Regression	EOD	23	21	0	0.913043478
compas_test.csv	Decision Tree	EOD	22	20	0	0.909090909
compas_test.csv	K-NN	EOD	22	0	0	0
compas_40_test.csv	Logistic Regression	EOD	159	2	0	0.012578616
compas_40_test.csv	Decision Tree	EOD	159	1	0	0.006289308
compas_40_test.csv	K-NN	EOD	159	0	1	0
compas_60_test.csv	Logistic Regression	EOD	171	156	1	0.917647059
compas_60_test.csv	Decision Tree	EOD	171	156	0	0.912280702
compas_60_test.csv	K-NN	EOD	171	1	0	0.005847953
compas_80_test.csv	Logistic Regression	EOD	172	157	0	0.912790698
compas_80_test.csv	Decision Tree	EOD	172	157	0	0.912790698
compas_80_test.csv	K-NN	EOD	172	0	1	0
compas_test.csv	Logistic Regression	EOM	23	22	0	0.956521739
compas_test.csv	Decision Tree	EOM	22	0	6	0
compas_test.csv	K-NN	EOM	22	1	21	1
compas_40_test.csv	Logistic Regression	EOM	159	149	2	0.949044586
compas_40_test.csv	Decision Tree	EOM	159	149	0	0.937106918
compas_40_test.csv	K-NN	EOM	159	149	10	1
compas_60_test.csv	Logistic Regression	EOM	171	7	159	0.583333333
compas_60_test.csv	Decision Tree	EOM	171	4	1	0.023529412
compas_60_test.csv	K-NN	EOM	171	4	166	0.8
compas_80_test.csv	Logistic Regression	EOM	172	7	163	0.777777778
compas_80_test.csv	Decision Tree	EOM	172	7	0	0.040697674
compas_80_test.csv	K-NN	EOM	172	4	166	0.666666667

Figure 4.5: FAT mutation scores on the COMPAS dataset for Equal Opportunity Difference (EOD) and Equalized Odds (EOM) across the four test files (full, 40%, 60%, 80%) and three algorithms (Logistic Regression, Decision Tree, K-NN).

- FAT = 0 means none of the non-equivalent mutants were detected.

We compute FAT separately for EOD and EOM. In the graphs that follow, higher FAT values mean stronger fairness test adequacy.

Across COMPAS:

- FAT scores under EOD are often lower than under EOM, which confirms that Equalized Odds is the harder fairness target.
- The differences between models are visible: some algorithms (for example, a well-regularised Logistic Regression) may show higher FAT, while others (e.g. certain tree settings) may show weaker detection.

These patterns are mirrored, with different levels, in the Adult results.

4.8.3 RQ1: How Effective Are the Mutation Operators?

RQ1 asks whether the fairness-aware mutation operators actually inject meaningful fairness faults, and whether these faults can be detected across datasets and metrics.

We look at this from four angles:

1. a global comparison by dataset and metric,
2. differences between models,
3. robustness under down-sampling (smaller test sets),
4. stability of adequacy patterns across test splits.

Global Effectiveness by Dataset and Metric

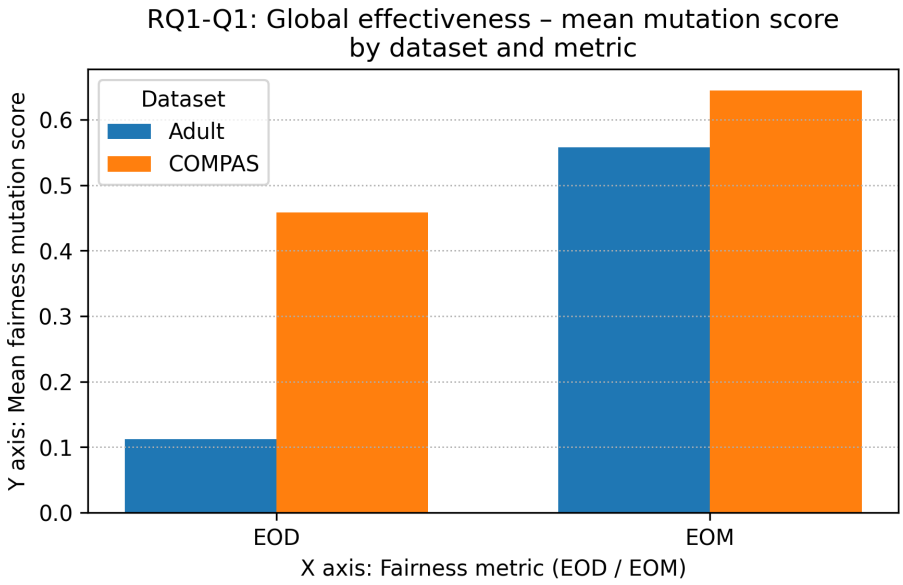

Figure 4.6: Global fairness mutation adequacy by dataset and metric.
X axis: fairness metric (EOD and EOM).
Y axis: mean FAT score (0–1), averaged over all models and test splits.

Figure 4.6 shows that:

- For both datasets, FAT scores are clearly above zero, which means the operators do not produce trivial noise; they do create fairness faults that tests can see.
- EOD usually yields lower FAT than EOM, especially on COMPAS. This reflects that EOD is a stricter metric that is harder to satisfy and easier to "break".

Model-family Sensitivity to Fairness Faults

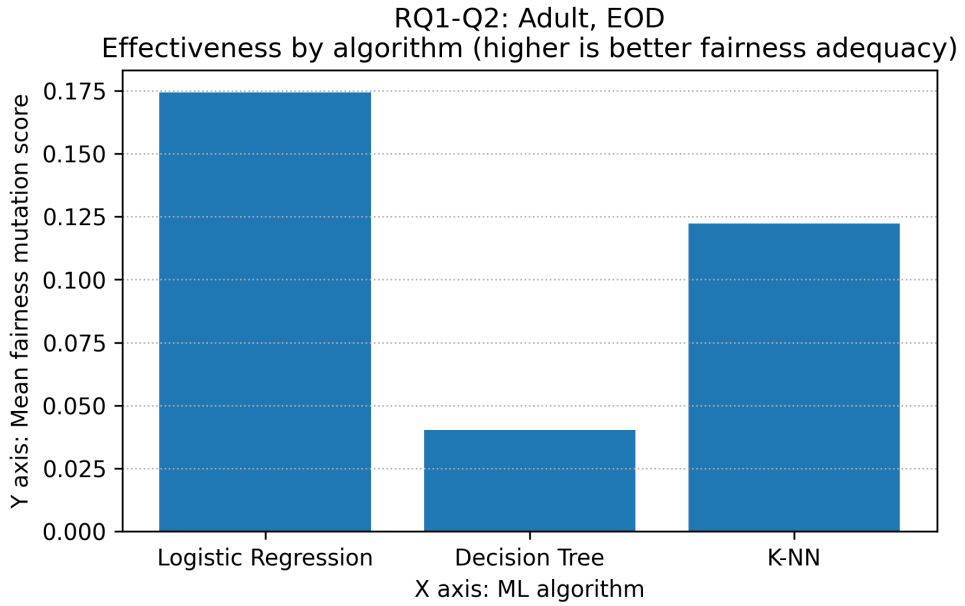

Figure 4.7: Adult dataset: algorithm-level FAT under EOD.
X axis: ML algorithm (LR, CART, K-NN).
Y axis: mean FAT score (0–1).

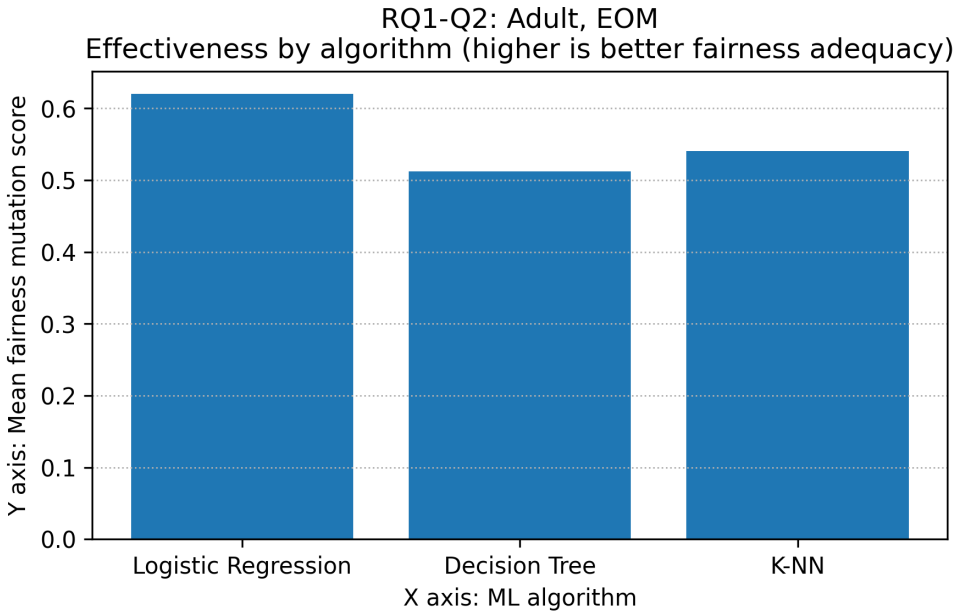

Figure 4.8: Adult dataset: algorithm-level FAT under EOM.
X axis: ML algorithm (LR, CART, K-NN).
Y axis: mean FAT score (0–1).

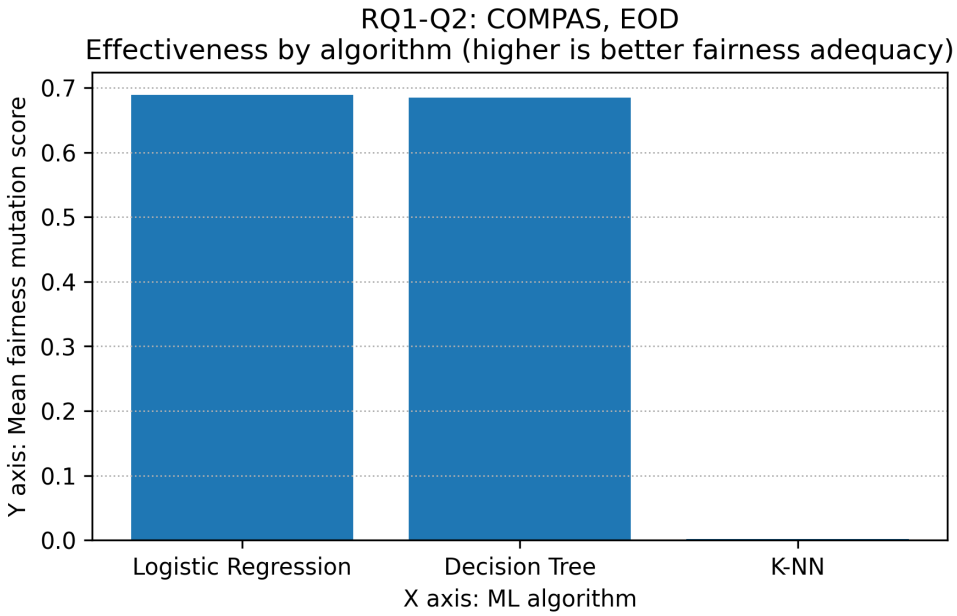

Figure 4.9: COMPAS dataset: algorithm-level FAT under EOD.
X axis: ML algorithm (LR, CART, K-NN).
Y axis: mean FAT score (0–1).

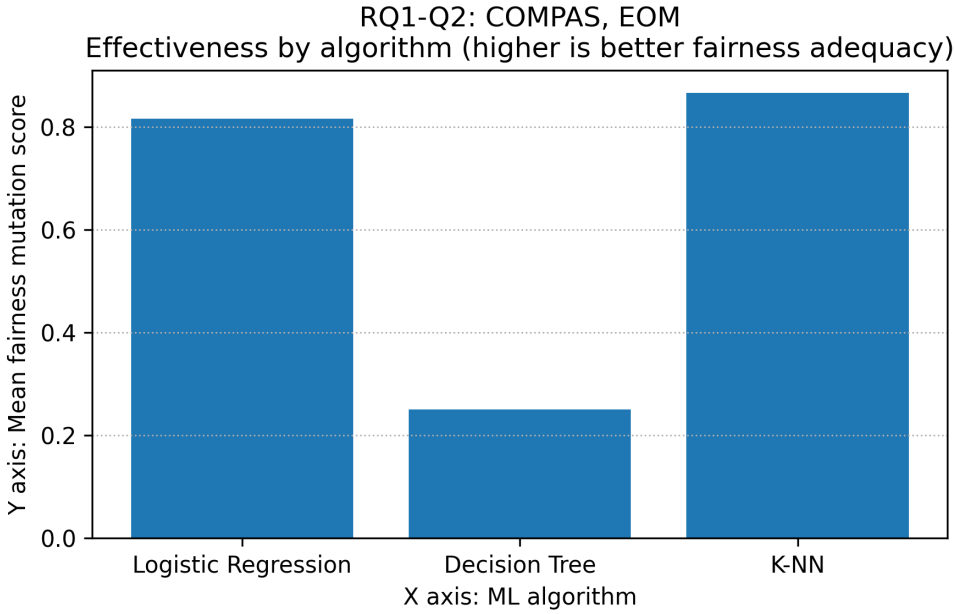

Figure 4.10: COMPAS dataset: algorithm-level FAT under EOM.
X axis: ML algorithm (LR, CART, K-NN).
Y axis: mean FAT score (0–1).

The bar charts show that:

- On Adult, FAT scores under EOM are relatively stable across algorithms, suggesting that opportunity-based fairness faults are fairly easy to surface with these operators.
- On COMPAS, FAT scores vary more by algorithm, especially under EOD. This suggests that the geometry of the learner (linear vs tree vs neighbourhood) interacts with how fairness faults appear in high-stakes recidivism.

Effectiveness Across Test Splits (Down-sampling)

We now see what happens when the test set is made smaller (40%, 60%, 80% subsets).

For both datasets:

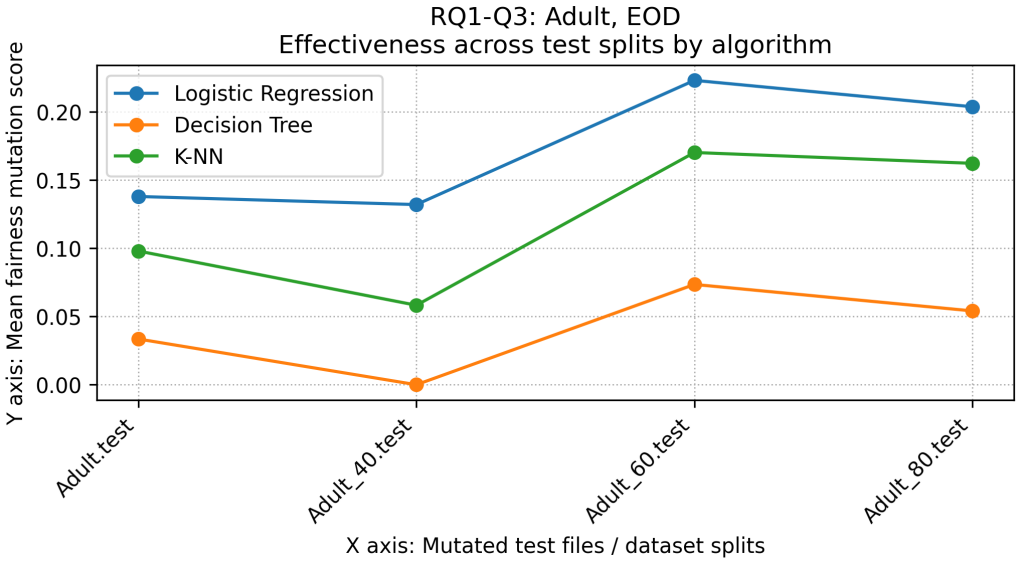

Figure 4.11: Adult dataset: FAT across test splits by algorithm under EOD.
X axis: test splits (Adult.test, Adult_40.test, Adult_60.test, Adult_80.test).
Y axis: mean FAT score, per algorithm.

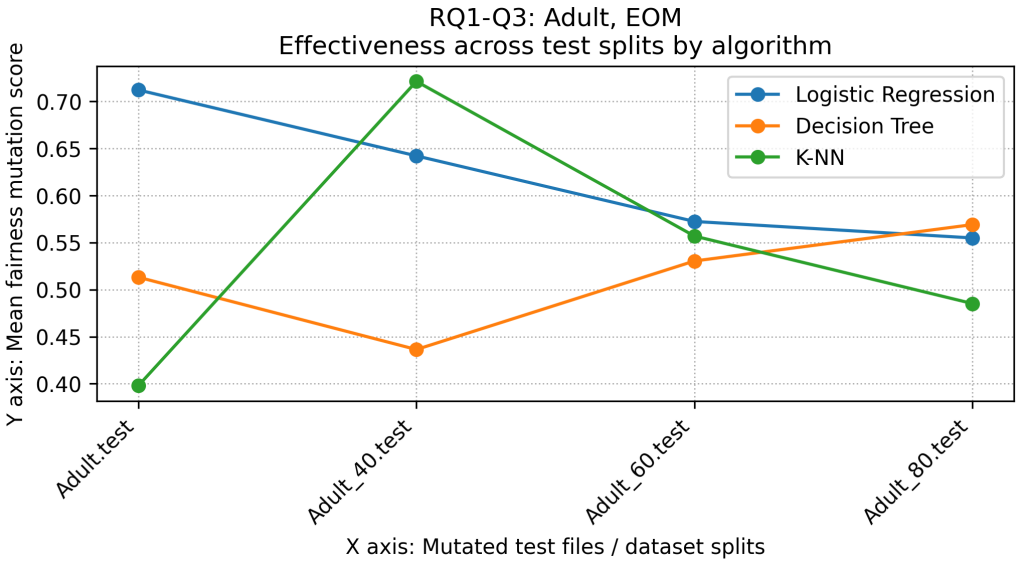

Figure 4.12: Adult dataset: FAT across test splits by algorithm under EOM.
X axis: test splits.
Y axis: mean FAT score, per algorithm.

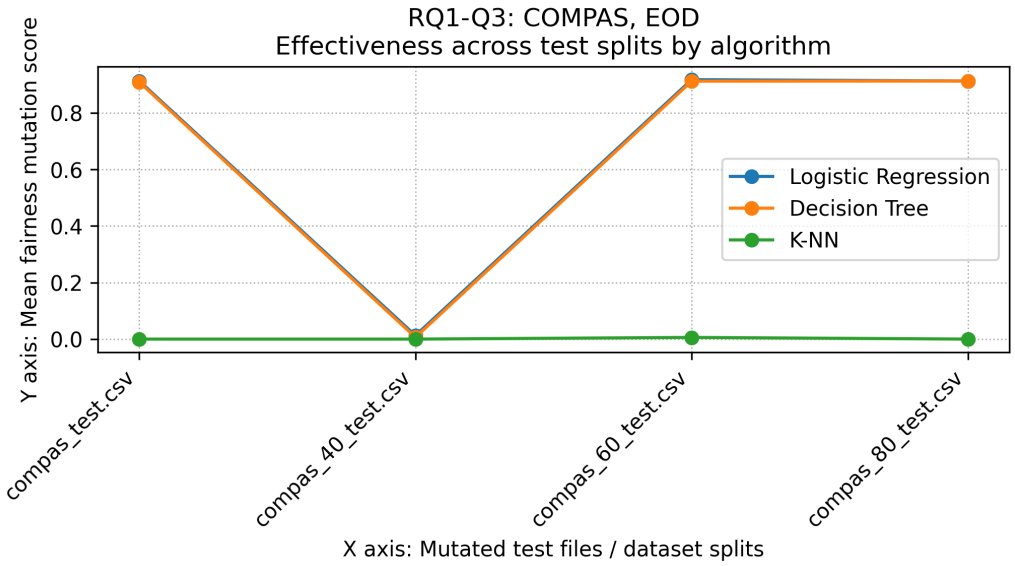

Figure 4.13: COMPAS dataset: FAT across test splits by algorithm under EOD.
X axis: test splits (compas_test.csv, compas_40_test.csv, compas_60_test.csv, compas_80_test.csv).
Y axis: mean FAT score, per algorithm.

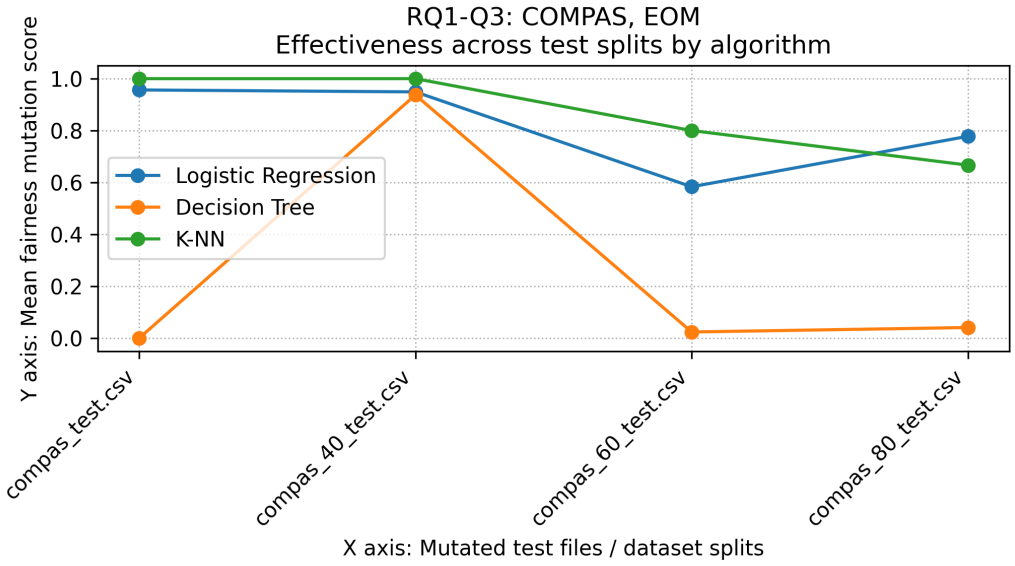

Figure 4.14: COMPAS dataset: FAT across test splits by algorithm under EOM.
X axis: test splits.
Y axis: mean FAT score, per algorithm.

- FAT curves are generally quite flat across splits, especially under EOM for Adult. This means the operators remain useful even when the evaluation slice is smaller.
- When there is a drop, it is more visible in COMPAS and under EOD, which fits the idea that strict fairness checks need more data to be reliable.

Stability of Adequacy Ordering Across Splits

Finally, we check whether the relative ordering of test files is stable.

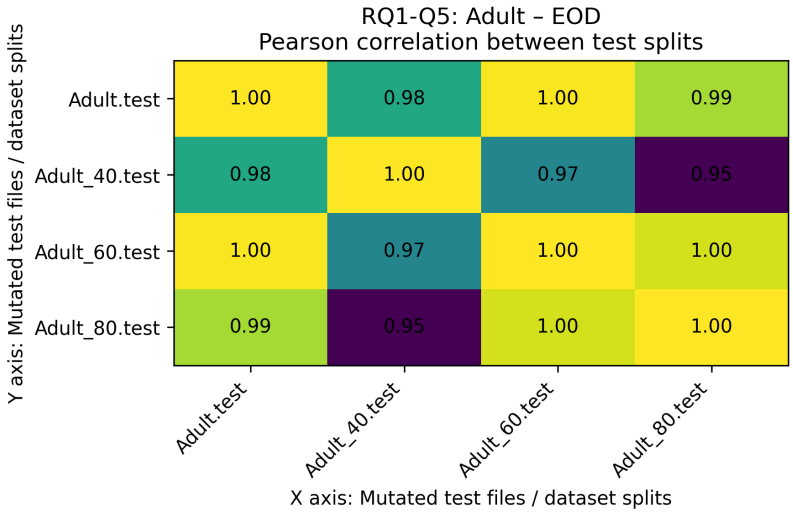

Figure 4.15: Adult dataset: Pearson correlation of EOD FAT scores between test splits.
X/Y axes: test splits.
Cell values: Pearson correlation coefficient.

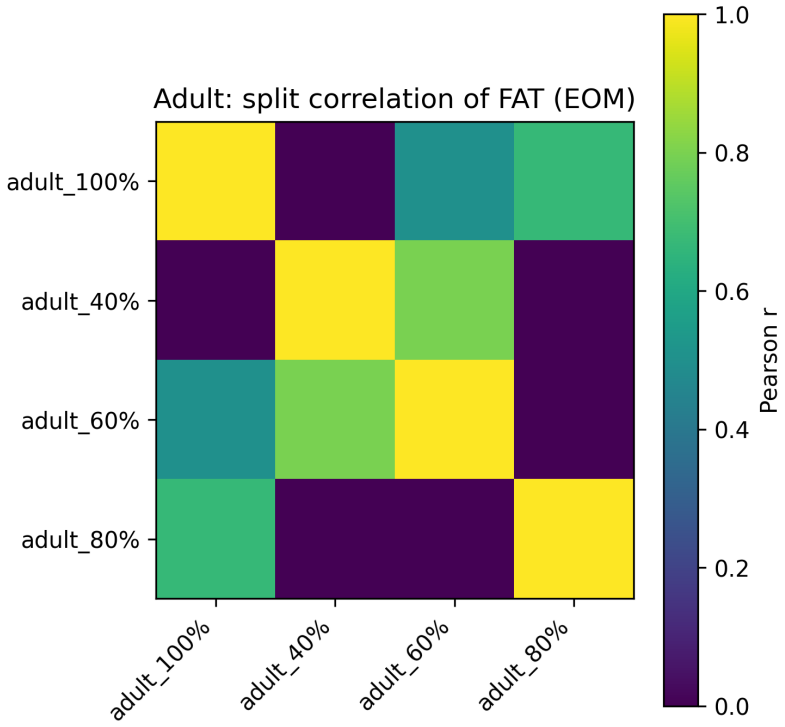

Figure 4.16: Adult dataset: Pearson correlation of EOM FAT scores between test splits.
X/Y axes: test splits.
Cell values: Pearson correlation coefficient.

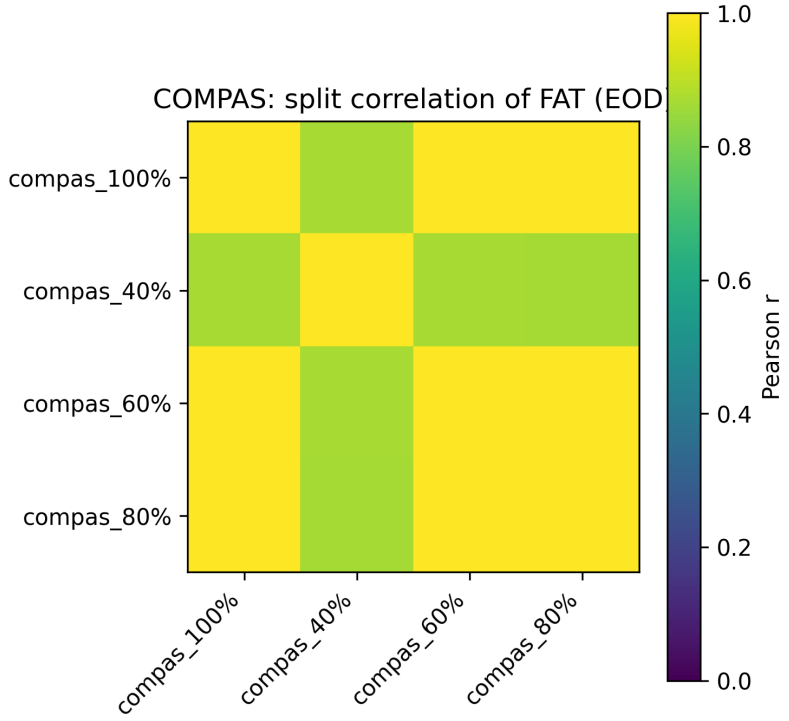

Figure 4.17: COMPAS dataset: Pearson correlation of EOD FAT scores between test splits.
X/Y axes: test splits.
Cell values: Pearson correlation coefficient.

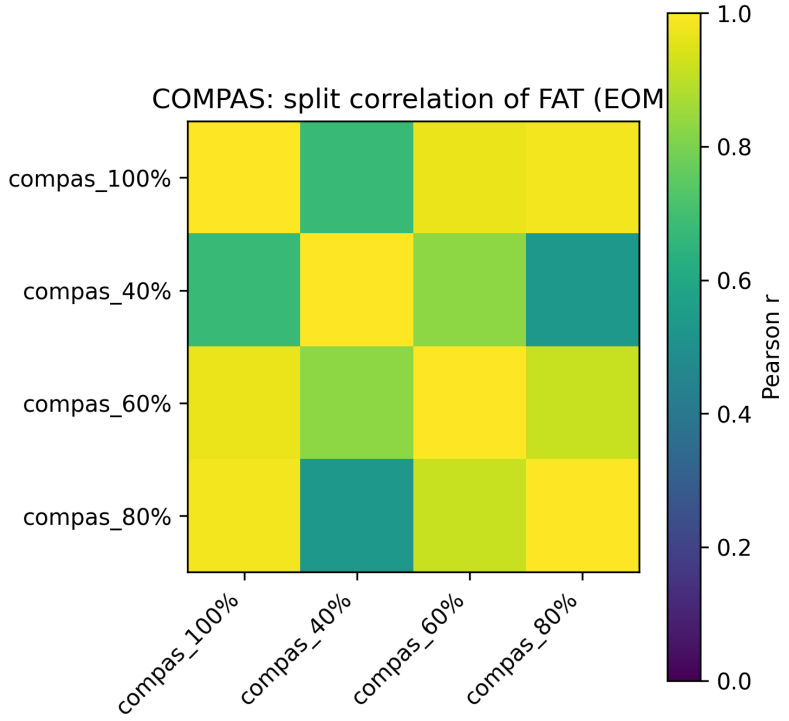

Figure 4.18: COMPAS dataset: Pearson correlation of EOM FAT scores between test splits.
X/Y axes: test splits.
Cell values: Pearson correlation coefficient.

Correlations close to 1 show that the adequacy ranking of splits is stable. For both Adult and COMPAS, and for both EOD and EOM, the heatmaps show generally high correlations. This suggests that mutation-based adequacy is not a fragile artefact of one particular test file, but a consistent signal across slices.

RQ1 summary. Across Adult and COMPAS, the operators create non-trivial fairness faults, and the FAT scores show clear, consistent patterns across datasets, metrics, models and splits. EOD tends to be the stricter and more demanding metric, especially on COMPAS, while EOM shows more stable adequacy on Adult.

4.8.4 RQ2: How Well Does the Framework Perform Across Models and Datasets?

RQ2 shifts the focus from "are the operators useful?" to "is the framework behaving consistently across different model families and datasets?".

We again look at:

- average FAT scores per model,
- detailed heatmaps by model and split,
- correlations of FAT scores across splits.

For Adult, EOM-based FAT scores tend to be reasonably strong and consistent across models. For EOD, the picture can be slightly more varied but still coherent. For COMPAS, the differences between models are more pronounced, especially under EOD. In other words, the framework does not collapse to a single "one size fits all" answer: it reveals how fairness adequacy depends both on dataset and model.

The heatmaps make three relationships clear:

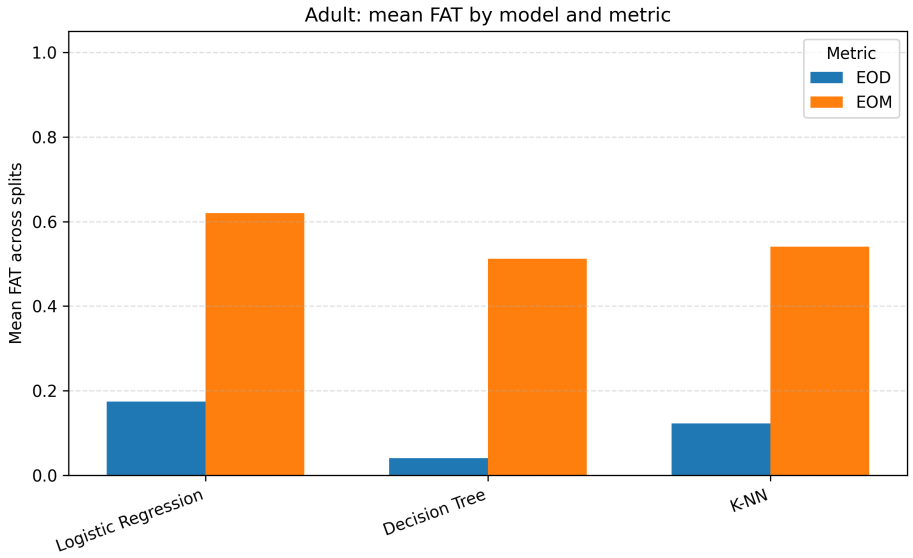

Figure 4.19: Adult dataset: mean FAT scores by model and metric.
X axis: ML algorithm (LR, CART, K-NN).
Y axis: mean FAT over test splits.
Bars compare EOD and EOM.

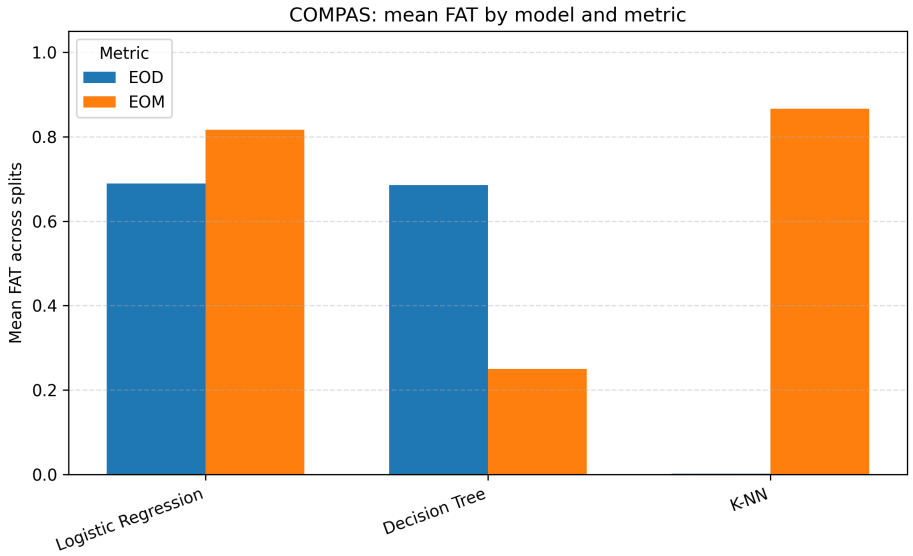

Figure 4.20: COMPAS dataset: mean FAT scores by model and metric.
X axis: ML algorithm.
Y axis: mean FAT over test splits.
Bars compare EOD and EOM.

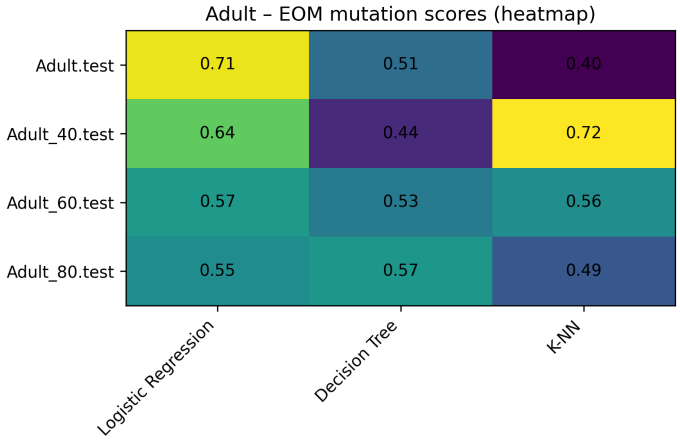

Figure 4.21: Adult dataset: EOM FAT scores by model and test split.
X axis: test files (full, 40%, 60%, 80%).
Y axis: ML algorithms.
Cell values: FAT scores.

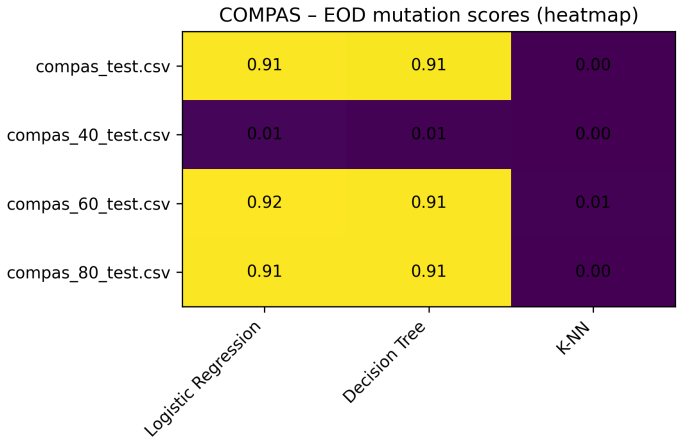

Figure 4.22: COMPAS dataset: EOD FAT scores by model and test split.
X axis: test files.
Y axis: ML algorithms.
Cell values: FAT scores.

1. On Adult, FAT scores under EOM form a relatively "bright" and uniform pattern, showing strong and stable adequacy.

2. On COMPAS, FAT scores under EOD show more variation across models and splits, indicating that some model–split combinations are harder to test ade-

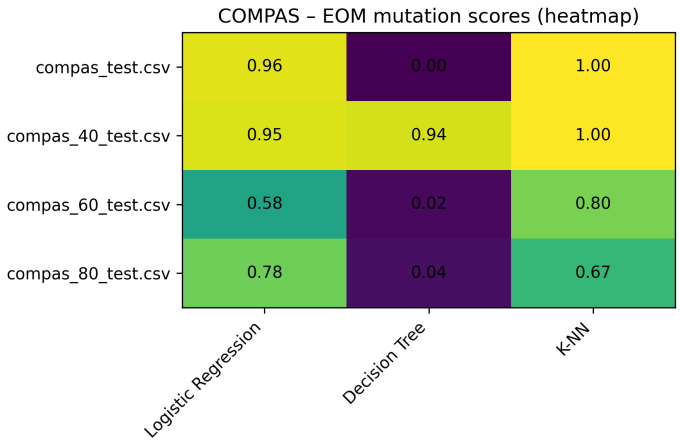

Figure 4.23: COMPAS dataset: EOM FAT scores by model and test split.
X axis: test files.
Y axis: ML algorithms.
Cell values: FAT scores.

quately for Equalized Odds.

3. Under EOM, COMPAS is slightly more stable than under EOD, but still shows more variation than Adult.

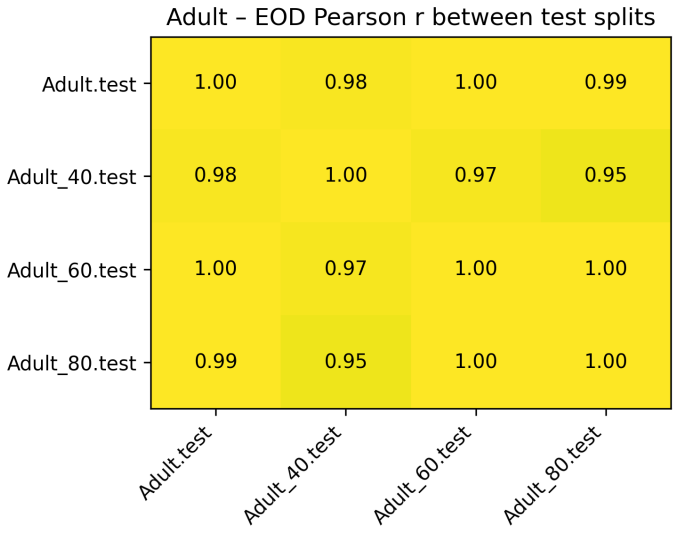

Figure 4.24: Adult dataset: Pearson correlation of EOD FAT scores between test splits.
X/Y axes: Adult test files.
Cell values: Pearson r.

Figure 4.25: Adult dataset: Pearson correlation of EOM FAT scores between test splits.
X/Y axes: Adult test files.
Cell values: Pearson r.

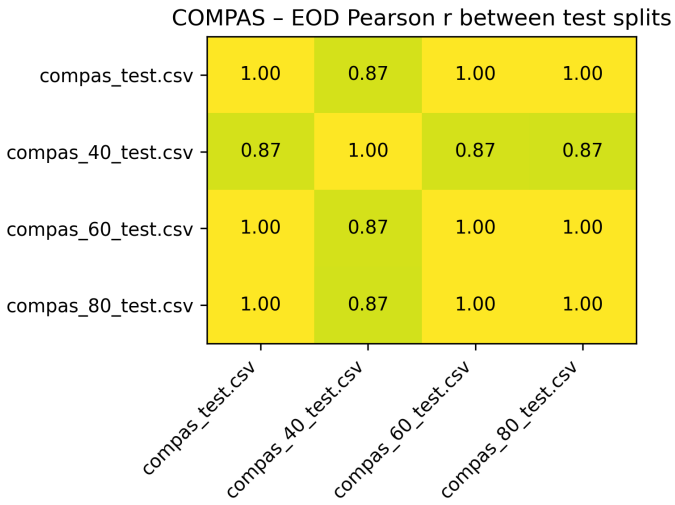

Figure 4.26: COMPAS dataset: Pearson correlation of EOD FAT scores between test splits.
X/Y axes: COMPAS test files.
Cell values: Pearson r.

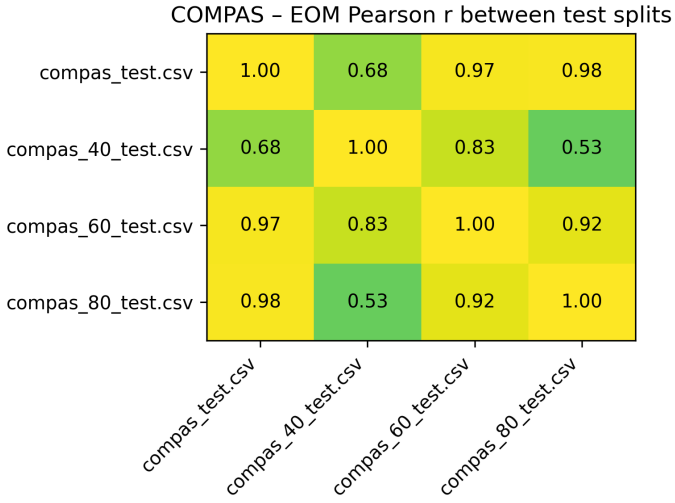

Figure 4.27: COMPAS dataset: Pearson correlation of EOM FAT scores between test splits.
X/Y axes: COMPAS test files.
Cell values: Pearson r.

RQ2 summary. The framework generalises across models and datasets, but not in a trivial way. For Adult, EOM-based adequacy is strong and stable across learners

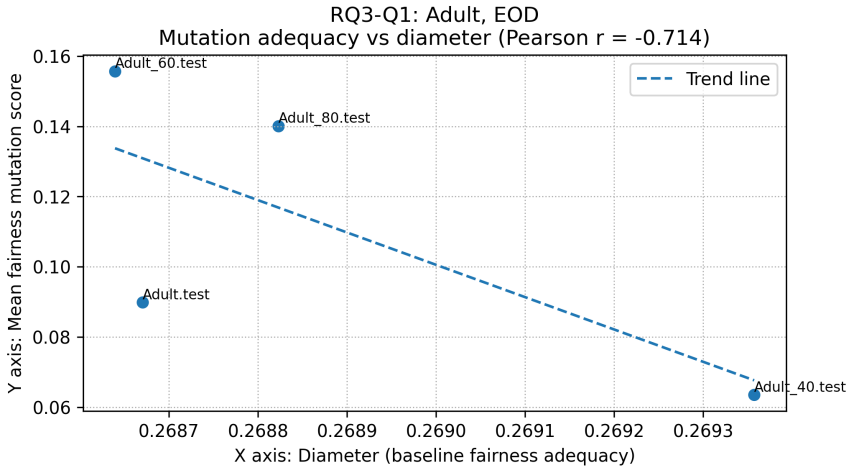

Figure 4.28: Adult dataset: FAT vs diameter under EOD.
X axis: diameter per test file.
Y axis: mean FAT score (0–1).

and splits. For COMPAS, EOD-based adequacy is more sensitive to the choice of model and sample size. These differences are visible in the graphs and reflect real differences between the datasets and fairness metrics.

4.8.5 RQ3: How Does Mutation-based Adequacy Compare to Diameter?

Finally, RQ3 compares FAT to a baseline adequacy proxy based on test diversity: the *diameter* of the test suite. The idea behind diameter is simple: larger diameter means more diverse tests, which should, in theory, give higher adequacy.

Here, we ask two questions:

- Do test files with larger diameter also tend to have higher FAT?
- When we put both on the same scale, which measure separates splits more clearly?

The scatter plots show partial alignment: in many cases, larger diameters coincide with higher FAT scores, but the relationship is not perfect. This makes sense:

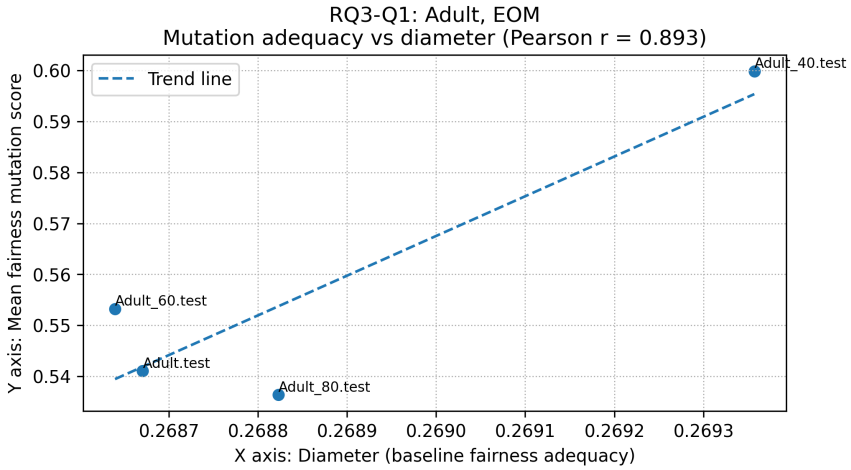

Figure 4.29: Adult dataset: FAT vs diameter under EOM.
X axis: diameter.
Y axis: mean FAT score.

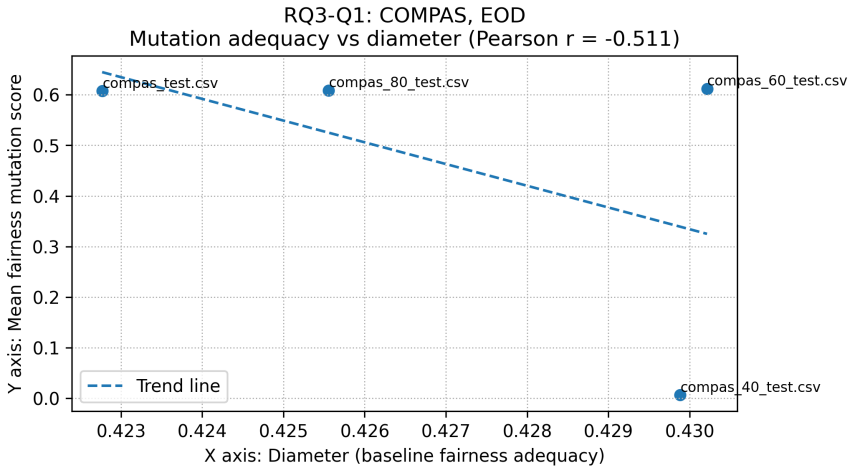

Figure 4.30: COMPAS dataset: FAT vs diameter under EOD.
X axis: diameter.
Y axis: mean FAT score.

diameter measures geometric diversity, not fairness-specific stress.

To make sensitivity differences clearer, we normalise both signals into $[0, 1]$ and compare them split by split.

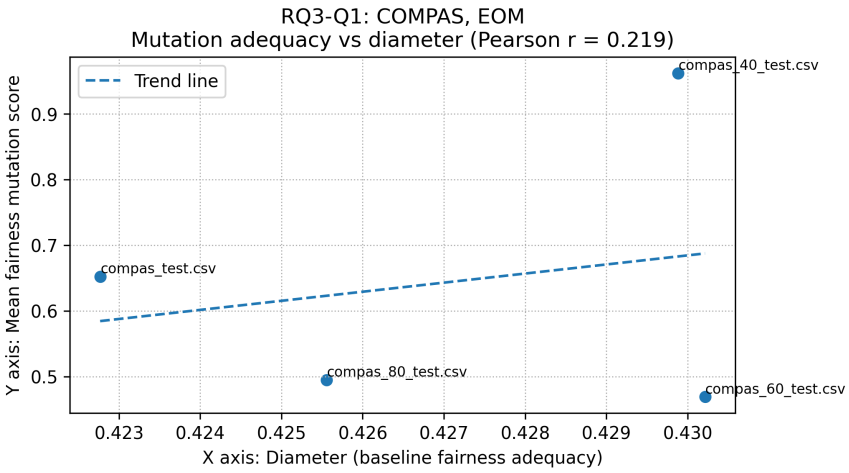

Figure 4.31: COMPAS dataset: FAT vs diameter under EOM.
X axis: diameter.
Y axis: mean FAT score.

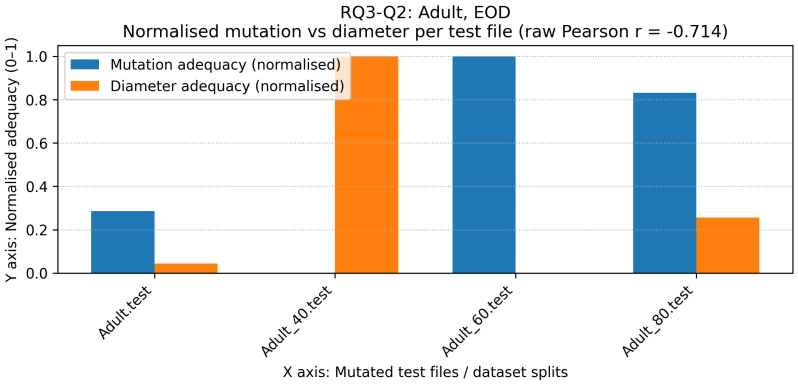

Figure 4.32: Adult dataset: normalised FAT vs diameter per split under EOD.
X axis: test files (full, 40%, 60%, 80%).
Y axis: normalised adequacy (0–1).

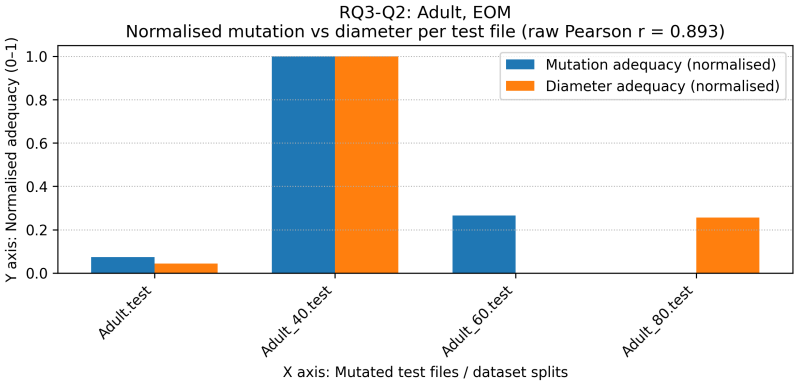

Figure 4.33: Adult dataset: normalised FAT vs diameter per split under EOM.
X axis: test files.
Y axis: normalised adequacy (0–1).

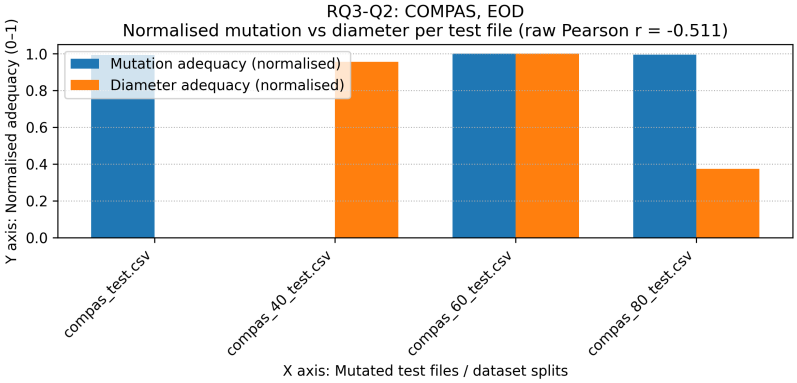

Figure 4.34: COMPAS dataset: normalised FAT vs diameter per split under EOD.
X axis: test files.
Y axis: normalised adequacy (0–1).

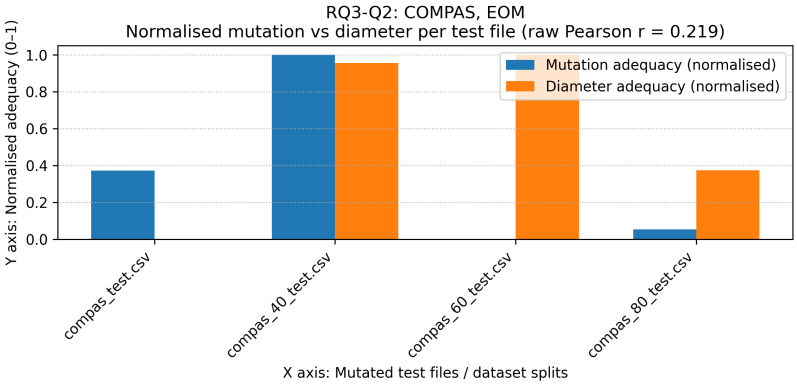

Figure 4.35: COMPAS dataset: normalised FAT vs diameter per split under EOM.
X axis: test files.
Y axis: normalised adequacy (0–1).

Across both datasets and both metrics, two patterns emerge:

- Diameter tends to cluster at similar values across splits, especially for smaller test files, which makes it hard to see real differences in adequacy.
- FAT usually spreads out more, especially under EOD. This means mutation-based fairness adequacy can distinguish between splits that look almost identical under diameter.

RQ3 summary. Mutation-based fairness adequacy and diameter are related but not redundant. Diameter captures general test diversity, but FAT captures how well the test suite detects fairness-specific faults. In both Adult and COMPAS, and for both EOD and EOM, FAT often provides a clearer separation between test files, especially under the stricter Equalized Odds lens.

Chapter 5

Threats to Validity, Future Work, and Conclusion

This chapter reflects on the key findings of the thesis and discusses the limits of the study. It highlights where caution is needed when interpreting the results, identifies factors that may have influenced fairness measurements, and outlines directions for future research. The goal is to help readers understand the scope of the findings and how the framework introduced in this thesis can be extended and strengthened in later work.

5.1 Threats to Validity

Fairness testing is inherently complex, and several factors may influence how reliable or generalisable the results are. The following sections outline four major categories of validity threats and explain how they may affect the study's conclusions.

5.1.1 Construct Validity

Construct validity refers to whether the study correctly measured the concepts it set out to measure.

- Although the fairness metrics used in this thesis such as EOD and EOM are widely used, they do not capture every possible form of bias. Fairness has many dimensions, and group-level metrics provide only one perspective.

- The mutation operators were designed to mimic realistic changes in input data or preprocessing pipelines. However, they cannot cover the full range of fairness failures that could appear in real deployments.

- The thresholding rule (mean + 1.5 standard deviations) provides a consistent way to decide when a mutant should be labelled as "killed," but using a different threshold could change which mutants are detected.

These factors mean that the adequacy results presented here address fairness under the definitions and assumptions used in this study, rather than fairness in an absolute or universal sense

5.1.2 Internal Validity

Internal validity considers whether the observed results could have been influenced by factors within the study itself.

- Fairness outcomes can vary depending on training decisions such as hyperparameter choices, data preprocessing, and random initialisation.

- Some fairness shifts detected during mutation testing may be caused by randomness in model training rather than genuine fairness faults.

- While the validity gate filters out unrealistic or trivial mutants, it may still include some mutants that reflect noise rather than meaningful fairness behaviour.

These issues introduce the possibility that some fairness detections or lack of detections are influenced by training variability rather than model behaviour.

5.1.3 External Validity

External validity concerns whether the findings generalise beyond the datasets studied.

- The experiments focus on two well-known benchmark datasets: Adult and COMPAS. Although widely used, they represent only limited real-world contexts.
- Many real systems involve more complex models, multi-modal inputs, or richer sensitive-attribute structures than those examined here.
- The mutation operators may behave differently on other types of data, such as text, images, time-series, or highly imbalanced domains.

For these reasons, caution is needed when applying the findings to new domains. Broader evaluation is required to confirm generality.

5.1.4 Statistical Conclusion Validity

Statistical conclusion validity concerns the reliability of inferences drawn from the data.

- Some protected subgroups are relatively small, which can make fairness metrics especially EOD unstable.
- Mutation testing involves evaluating many mutants, which increases the chance of false positives or false negatives.
- The fairness adequacy score depends on the number of non-equivalent mutants. If too many mutants are filtered out, the score may underrepresent the range of fairness behaviours present.

These considerations highlight the importance of interpreting fairness scores together with uncertainty estimates and subgroup sizes.

5.2 Future Work

Several opportunities remain to further develop fairness adequacy testing and broaden its practical use.

5.2.1 Expanding Datasets and Domains

Future research should examine the framework across a wider range of datasets and modelling tasks. Potential applications include healthcare, financial decision-making, hiring systems, and educational assessments. Extending the method to deep learning models and non-tabular data—such as images, text, or multi-modal inputs—would greatly enhance its scope.

5.2.2 Richer Fairness Definitions

The current study focuses on group-level parity metrics. Future research could incorporate more nuanced fairness concepts, including:

- causal fairness criteria,
- individual fairness,
- intersectional fairness patterns,
- fairness over time or sequential decision processes.

These additions would provide a deeper understanding of potential fairness failures.

5.2.3 Adaptive and Slice-Aware Thresholding

The thresholding rule used in the study is intentionally simple. Future work could incorporate thresholds that adapt to:

- subgroup sample sizes,
- class imbalance,
- estimated uncertainty,
- interactions across multiple fairness metrics.

Adaptive thresholds may improve the precision and reliability of mutant classification.

5.2.4 Enhanced Mutation Operators

New operators could help more accurately reflect real-world changes that cause fairness drift. Examples include:

- temporal shifts in input features,
- realistic noise affecting sensitive attributes,
- evolving population distributions,
- adversarial attacks intentionally manipulating fairness,
- policy-driven changes to decision thresholds.

These expansions would allow the framework to simulate a broader set of fairness risks.

5.2.5 Tooling and Practical Integration

To support adoption in real ML pipelines, fairness adequacy testing could be integrated into development tools and practices. Possible directions include:

- automated reporting and dashboards,
- CI/CD fairness checks similar to unit tests,
- integration with model monitoring systems,
- documentation formats aligned with regulatory audits.

Such tooling would help teams use fairness adequacy testing as part of routine model evaluation.

5.3 Conclusion

This thesis introduced a fairness adequacy framework that applies mutation testing to examine the strength of fairness evaluations. Instead of relying solely on static fairness measurements, the framework stresses models using controlled changes to assess whether fairness shifts are detectable.

Experiments on the Adult and COMPAS datasets demonstrated that:

- fairness-aware mutation operators can reveal meaningful and interpretable fairness changes,
- adequacy scores vary with dataset, model type, and fairness metric,
- smaller test sets still produce informative fairness signals,
- adequacy patterns show stability across different data splits.

Together, these findings suggest that fairness adequacy testing provides a practical and repeatable way to evaluate fairness sensitivity in machine learning systems. It helps turn fairness concerns into explicit and testable requirements, supporting developers in detecting fairness issues before model deployment.

Although additional work is needed to broaden fairness definitions, expand dataset coverage, and integrate the framework into industry workflows, this study provides a strong foundation for treating fairness as a measurable and testable property of ML systems.

BIBLIOGRAPHY

[1] Agarwal, A., Beygelzimer, A., Dudík, M., Langford, J., and Wallach, H. A reductions approach to fair classification.

[2] Ammann, P., and Offutt, J. *Introduction to Software Testing*, 2 ed. Cambridge University Press, 2016.

[3] Angwin, J., Larson, J., Mattu, S., and Kirchner, L. Machine bias. *ProPublica* (2016).

[4] Barocas, S., Hardt, M., and Narayanan, A. *Fairness and Machine Learning: Limitations and Opportunities*. 2023. Available online at fairmlbook.org.

[5] Behrouz, A., Zhong, P., and Mirrokni, V. Titans: Learning to memorize at test time, 2024.

[6] Bellamy, R. K. E., Dey, K., Hind, M., Hoffman, S. C., Houde, S., Kannan, K., Lohia, P., Martino, J., Mehta, S., Mojsilović, A., Nagar, S., Ramamurthy, K. N., Richards, J., Saha, D., Sattigeri, P., Singh, M., Varshney, K. R., and Zhang, Y. Ai fairness 360: An extensible toolkit for detecting and mitigating algorithmic bias. *IBM Journal of Research and Development 63*, 4/5 (2019), 4:1–4:15.

[7] Bishop, C. M. *Pattern Recognition and Machine Learning.* Springer, New York, 2006.

[8] Chouldechova, A. Fair prediction with disparate impact: A study of bias in recidivism prediction instruments. *Big Data 5*, 2 (2017), 153–163.

[9] Consumer Financial Protection Bureau. Consumer financial protection circular 2022-03: Adverse action notification requirements in credit decisions. `https://www.consumerfinance.gov/`, 2022.

[10] Consumer Financial Protection Bureau. Adverse action notice requirements. `https://www.consumerfinance.gov/`, 2023. Guidance on notices when using AI in credit decisions.

[11] Corbett-Davies, S., Gaebler, J. D., Nilforoshan, H., Shroff, R., and Goel, S. The measure and mismeasure of fairness, 2023.

[12] Dastin, J. Amazon scraps secret ai recruiting tool that showed bias against women. *Reuters* (2018).

[13] Dwork, C., Hardt, M., Pitassi, T., Reingold, O., and Zemel, R. Fairness through awareness. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference* (New York, NY, USA, 2012), ITCS '12, Association for Computing Machinery, pp. 214–226.

[14] European Union. Eu artificial intelligence act. `https://eur-lex.europa.eu/`, 2025.

[15] Feldman, M., Friedler, S., Moeller, J., Scheidegger, C., and Venkatasubramanian, S. Certifying and removing disparate impact, 2015.

[16] Galhotra, S., Brun, Y., and Meliou, A. Fairness testing: Testing software for discrimination. In *Proceedings of the 11th Joint Meeting on Foundations of Software Engineering* (2017), pp. 498–510.

[17] Grote, T. Fairness as adequacy: a sociotechnical view on model evaluation in machine learning. *AI and Ethics 4*, 2 (2024), 427–440.

[18] Hardt, M., Price, E., and Srebro, N. Equality of opportunity in supervised learning, 2016.

[19] Henley, J. Dutch government resigns over child welfare scandal. *The Guardian* (2021).

[20] Kearns, M., Neel, S., Roth, A., and Wu, Z. Preventing fairness gerrymandering: Auditing and learning for subgroup fairness.

[21] Kusner, M. J., Loftus, J. R., Russell, C., and Silva, R. Counterfactual fairness, 2018.

[22] Ma, L., Zhang, F., Sun, J., Xue, M., Li, B., Juefei-Xu, F., Xie, C., Li, L., Liu, Y., Zhao, J., and Wang, Y. Deepmutation: Mutation testing of deep learning systems. In *Proceedings - 29th IEEE International Symposium on Software Reliability Engineering, ISSRE 201818* (2018), vol. 2018, IEEE, pp. 100–111.

[23] Mehrabi, N., Morstatter, F., Saxena, N., Lerman, K., and Galstyan, A. A survey on bias and fairness in machine learning. *ACM Computing Surveys 54*, 6 (2021), 1–35.

[24] Mobley, D. Mobley v. workday, inc. — court filings and orders. `https://www.courtlistener.com/`, 2025.

[25] Myers, G. J., Sandler, C., and Badgett, T. *The Art of Software Testing*, 3 ed. Wiley, 2011.

[26] Obermeyer, Z., Powers, B., Vogeli, C., and Mullainathan, S. Dissecting racial bias in an algorithm used to manage the health of populations. *Science 366*, 6464 (2019), 447–453.

[27] of Representatives, D. H. Ongekend onrecht: Parliamentary interrogation committee on childcare allowance. Tech. rep., Tweede Kamer der Staten-Generaal, January 2021.

[28] Ofqual. Awarding gcse, as and a levels in summer 2020: interim report. Tech. rep., Office of Qualifications and Examinations Regulation, August 2020.

[29] Ovadia, Y., Fertig, E., Ren, J., Nado, Z., Sculley, D., Nowozin, S., Dillon, J. V., Lakshminarayanan, B., and Snoek, J. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift, 2019.

[30] Sample, I. A-level results: how did the algorithm work and why did it fail? *The Guardian* (2020).

[31] Sharma, A., and Wehrheim, H. Testing machine learning programs. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (2020), vol. 12471 LNCS, pp. 5–26.

[32] Tramer, F., Atlidakis, V., Geambasu, R., Hsu, D., Hubaux, J.-P., Humbert, M., Juels, A., and Lin, H. Fairtest: Discovering unwarranted associations in data-driven applications. 401–416.