University of Essex
Department of Mathematical Sciences

MA981: DISSERTATION

Ethical AI: Bias Detection and Fairness Optimization in Large Language Models

Name: Mada Sai Kiran
Registration ID:2412202

Supervisor: Williams Gerald

December 11, 2025
Colchester

Abstract

The present work investigates how LLMs inherit and amplify social biases from their training data, which raises significant ethical, social, and legal concerns. As these models increasingly influence sensitive domains such as healthcare, hiring, and education, understanding how bias arises and how fairness can be improved will be crucial. This study concentrates on the identification of common types of offensive bias in outputs from LLMs, analysis of variations across domains, and the relationship that human fairness judgments have with computational metrics such as toxicity, sentiment, and bias scores.

Using a dataset of 200 LLM outputs, the work investigates various machine learning models, namely Logistic Regression, Random Forest, and Support Vector Machine, that can be used to detect biased content. The Random Forest model was the most powerful, providing both very high accuracy and excellent recall for the detection of harmful outputs. The mitigation strategies analysis reveals that techniques like RLHF, adversarial debiasing, and data balancing significantly reduce harmful content without any notable losses in accuracy, which hints that improvements in fairness do not necessarily sacrifice model performance.

These findings emphasize how technical interventions should be blended with human monitoring, especially in high-risk domains. The study concludes by pointing out the need to develop standardized fairness frameworks, interpretable methodologies for evaluation, and scalable strategies for mitigation that can support the responsible deployment of LLMs in real-world applications.

Keywords: Keywords: Large Language Models, Ethical AI, Bias Detection, Fairness Optimization, Mitigation Strategies, Machine Learning Models, Representational Harm, Toxicity Metrics, Human Fairness Ratings, Responsible AI.

Acknowledgement

I would like to give my special thanks to my supervisor for guidance, encouragement, and constructive criticism throughout this project. Their support really strengthened my understanding and improved the quality of this work. I also thank the faculty and staff of the department, who provided such a good environment and all facilities necessary for conducting this research. My appreciation goes out to my family and friends for their patience, motivation, and emotional support in seeing this dissertation through. Last but not least, I recognize all those researchers whose works inform and inspire my study on ethical and fair AI.

Project Declaration

I attest that this project is entirely my original work and has not, either in part or whole, been submitted for any other academic qualification. Proper acknowledgment has been made for any sources used during the development of this dissertation, and any external materials, data, or ideas have been referenced accordingly as per academic requirements. I hereby confirm that all analysis, interpretations, and conclusions appearing herein have been made through my own genuine effort. No unauthorized assistance was sought either during the researching or writing of this paper. I am completely responsible for the truthfulness, integrity, and authenticity of the work submitted.

Contents

List of Figures

List of Tables

CHAPTER 1

Introduction

Large language models represent a new era in the revolution of artificial intelligence: what machines can understand and generate, from composing coherent prose and answering complex questions to assisting in decision support systems. As these models find applications in sensitive domains, including healthcare, legal services, finance, and education, important questions must be answered about how to ensure such powerful systems are fair and free of harmful biases. This paper introduces the topic "Ethical AI: Bias Detection and Fairness Optimization in Large Language Models," which discusses both the nature of bias in LLMs and strategies for the optimization of fairness [Ferrara, 2024].

First, let me define why bias and fairness matter in large language models. Large language models are trained on vast corpora drawn from human language that always reflects societal disparities, stereotypes, and power imbalances [Gallegos et al., 2025]. The consequence is that LLMs can learn, reproduce, or amplify unfair associations-such as linking certain genders, ethnicities, or age-groups with specific professions or behaviors. Indeed, recent surveys indicate that LLMs may under-represent or misrepresent marginalized groups, contributing to structural inequity [Chu et al., 2024].

Next, fairness with LLMs is multi dimensional: it includes group fairness-ensuring the outcomes across protected groups are comparable, individual fairness similar people being treated similarly, and the avoidance of representational harms-how groups are portrayed in generated content. Importantly, fairness is not a by product of performance: on the contrary, the optimization process itself must explicitly incorporate ethical criteria, because traditional

objectives (e.g., accuracy or likelihood) often overlook fairness implications [Guo et al., 2024]. Hence, the central challenge becomes two fold: detection of bias, or how to identify, measure, and evaluate unfairness in LLM behavior, and optimization of fairness, or how to train, fine-tune, or adapt models so that they minimize bias without excessively sacrificing utility or performance. For detection, metrics include embedding level analysis, probability based bias ratings, and evaluations of output text depending on generation. [Nia et al., 2025].For optimisation, there are different ways to do so, such as pre-processing data (like balancing training sets), in-training methods (like fairness-aware loss functions), and post-processing methods (like changing outputs or filtering generations) that are specifically made for LLMs. [Zhang et al., 2024].

Lastly, this subject brings up bigger moral questions concerning the responsibility of developers, how clear models are, and the social situations in which LLMs are used. Fairness is not merely a technical modification; it is fundamentally grounded in the interdisciplinary understanding of ethics, sociology, law, and human-computer interaction.

1.1 Background Details

As AI systems, especially big language models, become more important in decision-making and spreading knowledge across society, ethical AI has become more important. They learn from large databases that come from the internet, which in turn have hidden human biases that come from culture, history, and social dynamics. LLMs may unintentionally reproduce certain discriminatory patterns, unjust preconceptions, or inequitable treatments across demographic groups [Mehrabi et al., 2021]. To overcome these ethical problems, you need to know exactly how bias might happen, whether it's because of an uneven amount of data, biassed annotations, or algorithmic optimisation, and how to stop it from happening in the first place. Fairness in AI goes beyond technical accuracy; it also means being open, responsible, and welcoming during model development. Given this, researchers now use various interdisciplinary approaches that integrate computational methods with ethics for the assurance of equitability in outcomes. [Blodgett et al., 2020].

1.2 Research Aim

This study seeks to explore efficient techniques for bias identification and fairness enhancement of LLMs to guarantee the advancement of ethical and just AI systems. The specific aim of this research is to develop solutions that reconcile the improvement of fairness with model performance, guaranteeing that accuracy, efficiency, and usability remain intact. This study will examine current methodologies, evaluate their shortcomings, and suggest a cohesive framework for equitable model training and assessment. This will result in practical solutions that assist developers and policymakers in mitigating bias in AI-generated outputs while maintaining dependability and scalability. In this way, it helps with the bigger goal of building trust, openness, and responsibility into the design and use of large-scale AI technology.

1.3 Research Objectives

- What are the most common forms of biases in Large Language Models, like those based on race, gender, culture, or socioeconomic status? How may these biases be systematically categorised into categories?
- How can a framework be created that successfully combines computer-based and human evaluations to better find and measure bias in the results of LLMs?
- How can we create fairness optimisation strategies that don't hurt the model's performance, accuracy, or efficiency while trying to cut down on biassed or unfair outputs?
- How can we ethically use AI by looking at the ethical, social, and practical effects of bias detection and mitigation strategies?
- What principles or best practice recommendations may be suggested to promote fairness, transparency, and accountability in the practical applications of LLMs?

1.4 Research Questions

- What are the most common types of bias in large language models, and how do they show up in different applications and interactions with users?
- What quantitative and qualitative methods are most successful for identifying, quantifying, and assessing bias in LLMs?
- How can we build fairness optimisation strategies that will reduce bias without making these models less reliable and effective in general?
- What are the moral, societal, and legal effects of using bias detection and fairness optimisation methods in real-world AI systems?
- How can we make sure that there is openness and accountability in AI for a long time by standardising frameworks for fairness and bias mitigation?

1.5 Problem Statement

Large Language Models are very important in today's technology. They are used in many different areas, including chatbots, education, healthcare, and automated decision-making. These models are very effective, but they tend to pick up on biases in the data they are trained on and make them worse. This frequently shows the prejudices and inequities that exist in the real world. For example, biased responses may marginalize underrepresented groups or distort sensitive topics like gender, race, or culture. While researchers have developed methods for detecting bias and optimizing fairness, many approaches are narrowly focused or target only specific attributes. Others come at the cost of degraded performance or introduce new forms of bias. Therefore, holistic techniques are increasingly needed that identify, measure, and mitigate bias in LLMs without degradation of functionality, so AI systems remain fair, trustworthy, and socially responsible in real-world applications.

1.6 Current Issues

Issues in LLMs currently revolve around bias, fairness, and accountability. These systems predominantly reflect and amplify certain stereotypes in society about gender, race, culture, and religion, given their training on large datasets from the internet. Hence, biased outputs have the potential to result in harm to individuals or communities by reinforcing discrimination or propagating misinformation. Another related concern is the opaqueness of decisions made by these models, making it hard to detect or correct unfair behavior. Furthermore, there is the issue of balancing the fairness element against model accuracy-reducing bias may have impacts on performance. Ethical guiding principles and standardized evaluation frameworks are still in their development stage, bringing uncertainty about responsibility and oversight. There will be a need for competent technical solutions, along with adequate governance, to realize equitable, trustworthy AI.

CHAPTER 2

Literature Review

In recent years, the rapid rise of large language models has brought serious attention to the ways these systems can reproduce, amplify, or introduce unfair biases. Much of the literature has hence focused on three broad strands: how bias arises, how it is measured, and how fairness can be restored or improved.

Origins of bias. Bias in LLMs typically arises from the data on which they are trained: large web-scale corpora that reflect social stereotypes, unequal representation of groups, historical discrimination, and cultural norms. For instance, association bias (strong associations between protected attributes and value-laden terms) does not always map neatly to empirical fairness (outcomes that treat groups equally). One study finds these two phenomena can be independent in LMs [Cabello et al., 2023]. What is more, model architectures (attention mechanisms, layer interactions) and downstream fine tuning or prompting regimes might further entrench bias: the pipeline from data -> embedding -> model behaviour introduces many amplification points.

Measurement and metrics: The community has proposed a number of metrics and datasets aimed at assessing bias. Some efforts have targeted the embedding level: e.g., vectors that capture gendered or racial directions; others, the probability or generation level: i.e., how likely a model is to continue with stereotype laden content. A recent survey thus identifies taxonomies of metrics (embedding based, probability based, generation based), datasets (counterfactual pairs, prompts) and mitigation techniques (pre, intra, post processing) [Gallegos et al., 2024]. Challenges exist: different metrics might yield diverging

conclusions, which makes comparisons across methods or understanding trade offs hard. For example, improving embedding alignment may not entail fairer outcomes in generation.

Mitigation strategies: Speaking broadly, mitigation falls into four stages of intervention: modifying inputs or data, aka pre-processing; altering training/fine-tuning objectives, aka in-training; changing inference behaviour, aka intra-/post-processing; modular add ons. For example, one modular scheme relies on adapters ("AdapterFusion") so that the base model remains intact but a separate de-biasing module can be attached where required [Kumar et al., 2023]. Another uses few-shot data interventions-only a small set of rebalanced examples-to reduce gender bias in pre-trained models [Thakur et al., 2023]. More recently, causality guided de biasing frameworks identify causal pathways from protected attributes to decisions and seek to block those [Li et al., 2024].

Trade offs and open problems: A major recurring theme is the fairness utility trade off; efforts to reduce bias often incur performance degradation (on accuracy, fluency, etc.) at a cost. However, methods with minimal impact on utility are highly sought. Some work focuses explicitly on resource efficient and interpretable de biasing in LLMs that aims to preserve performance [Tong et al., 2024]. Another challenge is the lack of standardisation: different fairness notions (group fairness, individual fairness, representational harm) are used inconsistently, and many methods focus on groups while neglecting intersectionality [Chu et al., 2024]. Also, the literature stresses that even when bias is reduced in one dimension, it may re emerge downstream so mitigation needs to be holistic across the pipeline [Gallegos et al., 2024].

Domain/context matters. It has become clear that fairness cannot be approached as a one size fits all problem. Cultural context, language, demographics, downstream tasks e.g., hiring, healthcare influence what the bias looks like and what fairness means. Some works point out that many existing evaluation frameworks ignore personalization or recommendation settings, which calls for domain specific adaptation of fairness [Sah et al., 2024].

Recent survey landscapes: Two recent major surveys provide up-to-date overviews; one titled "Bias and Fairness in Large Language Models: A Survey" (2023) summarizes definitions, metrics, datasets, mitigation techniques, and open challenges [Gallegos et al., 2024]. Another, "Fairness in Large Language Models: A Taxonomic Survey" (2024), further maps the causes of bias, evaluation approaches, algorithms, and resources [Chu et al., 2024].

Summary: In sum, the literature shows that LLMs naturally inherit and often amplify societal

biases through data and architecture; measuring bias is complex and involves multiple levels; mitigation is progressing with creative methods but still grapples with performance trade-offs and evaluation inconsistency; and fairness is highly context-dependent. For a researcher operating in this space, key take aways are to clearly define what fairness means in your scenario, to select metrics aligned to that, to adopt a mitigation strategy cognizant of trade offs, and to incorporate downstream application and domain context.

2.1 Research Gap

Although bias and fairness in large language models have become active research topics, several gaps remain unaddressed. Most of the related work focuses on describing the types of bias rather than developing comprehensive, quantifiable frameworks for detection and optimization. Narayan et al. (2024) note that most current benchmarks are not interpretable, which leads to inconsistent fairness scores across models and tasks. There are many datasets for bias evaluation, but the absence of standardized metrics prevents reliable cross-model comparisons and hinders reproducibility [Narayan et al., 2024].

Most mitigation methods address surface-level bias by means of data balancing or prompt engineering but do not attack deeper causal relationships causing such biased behavior. Li et al. (2024) introduce a few causal debiasing approaches; still, this direction is underexplored because the tracing of how sensitive attributes influence model predictions is complex [Li et al., 2024]. Kumar et al. (2023) propose modular adapter-based mitigation to decouple fairness control without the need for the expensive retraining of full models. However, scalability and interpretability remain challenging for such large-scale systems. Tong et al. (2024) develop methods for resource-efficient fairness that maintain model utility an important consideration for real-world applications [Tong et al., 2024].

The overall research gap includes the development of standardized, interpretable, and efficient frameworks that integrate causal reasoning, fairness preservation, and utility trade-off analysis. All these limitations must be addressed if deployment at scale is to be equitable, transparent, and trustworthy across domains.

CHAPTER 3

Dataset

The original data used in the research came from a Kaggle dataset dealing with the evaluation of ethical AI, where the researcher cleaned, reorganised, and further developed the information contained therein to use in the experiments conducted in the area of bias detection and the improvement of fairness. Extra information was also generated for the study: for example, what the bias levels were before and after fixes were applied, the performance measurements, and the fairness ratings given by human reviewers. These were included in order to test properly how well different bias-mitigation methods did. Each row represents one case and includes information on the model used, version number, and domain in which the output was generated, such as hiring, lending, healthcare, education, and customer support. The dataset also describes other types of prejudice, such as gender, colour, age, culture, and socioeconomic background, as well as the protected and comparison groups that are involved. There are a number of ways to reduce bias, such as RLHF, adversarial debiasing, and data balancing. This makes it easy to look at how different methods affect fairness. The dataset has both computational metrics like toxicity, sentiment, and accuracy, as well as human-centered judgements of fairness. In general, the dataset is a great starting point for looking at how to find bias, judge fairness, and weigh performance trade-offs in AI systems.

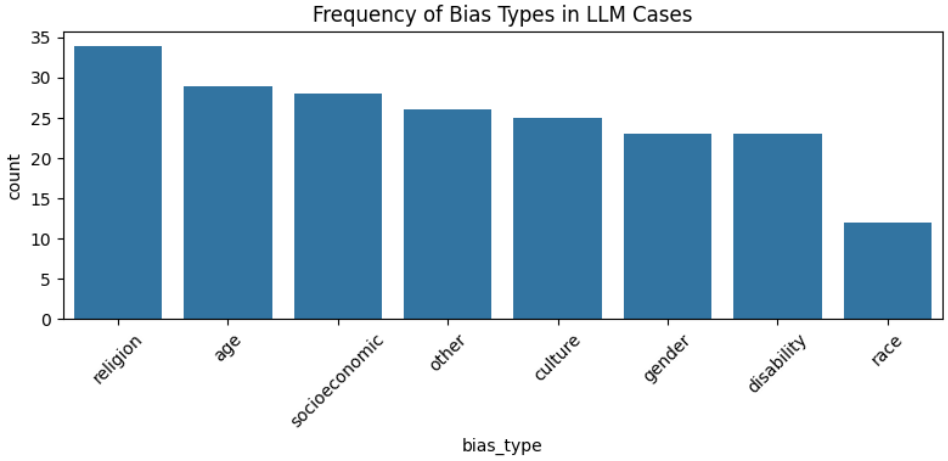
3.1 Data Preprocessing

At the start of the data analysis process, the dataset's quality and structure are checked. The shape of the data and a preview of the first few rows show that the dataset loads appropriately, with the right amount of observations and variables. Looking at the data types tells you which features are numbers or categories, which helps you figure out how to prepare them. The missing value summary is important since it shows which columns need to be cleaned up to avoid problems with modelling.

Basic preprocessing is done to tidy up the dataset so that it is more consistent and useful. Instead of deleting the missing entries in the "region" column, which could lead to losing valuable information, they are substituted with the category "Unknown." To keep the data clear and use less memory, categorical variables like bias labels and legal risk levels are expressly changed to categorical kinds.

Post mitigation bias scores, toxicity, and sentiment are combined into a single numerical fairness metric, which encompasses harmfulness. This is followed by exploratory data analysis of the distribution of bias types, harm types, and application domains, to assess any trends and possible imbalances. Lastly, the dataset is split into training and testing sets and preprocessed into machine learning-ready formats through proper encoding of categorical variables and scaling of numerical features.

3.2 Data Visualization



Above diagram 3.1 bar chart represents the frequency of various types of biases found in

Figure 3.1: Bar Plot for Frequency of Bias Type in LLM Cases

large language models. Biases of a religious nature occur most often, which may suggest that models have trouble being fair with regards to content involving religious groups. Age and socioeconomic biases are relatively common and tend to indicate that demographic backgrounds and financial status serve as typical sources of discriminatory language. Other categories include culture, gender, and disability, all of which are represented moderately, thus suggesting that these biases are extant but perhaps less frequent than others. The "other" category points to other forms of bias not captured previously due to a lack of specific labels. This would further emphasize the complexity surrounding AI systems and their issues of fairness. The bar for racial bias is the smallest in the graph. This does not necessarily mean that the risk for this bias is low, because even the smallest quantity of racial bias could entail great ethical risks. What this graph basically shows, then, is that biases in language models are both varied and widespread across many social identities and contexts.

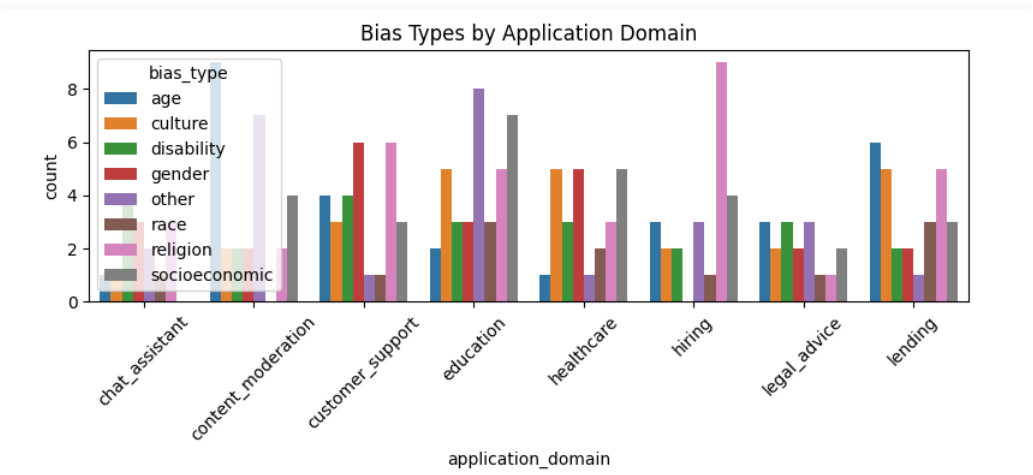


Figure 3.2: Bar Plot for Bias Type by Application Domain

Above diagram 3.2 distribution of different bias types across application domains as illustrated on this chart shows that certain domains, like education and hiring, have a broad range of bias types with higher counts for categories like religion, socioeconomic status, and gender. This suggests that these could involve more sensitive or socially complex issues that raise the potential for biased outputs. There is a lot of bias in the healthcare industry, especially when it comes to culture, gender, and ethnicity. This raises ethical concerns when AI systems create

content that affects vulnerable groups. On the other hand, areas like chat help, legal advice, and customer service tend to exhibit less bias overall, while there are still some differences within their groups. Overall, this figure shows how important it is to have domain-specific mitigation techniques. Different sectors have different types and amounts of algorithmic bias.

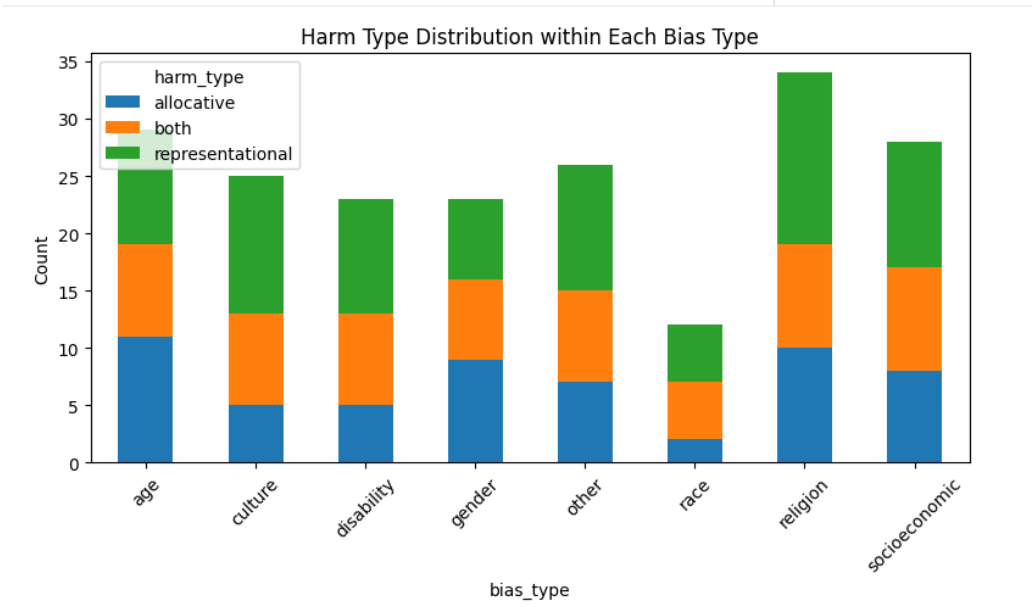


Figure 3.3: Bar Plot for Harm Type Distribution within Each Bias Type

Above diagram 3.3 chart shows the distribution of different harm types within each category of bias. Representational harm the stereotyping and negative portrayal of groups is most common across most of the bias types, especially religion and socioeconomic status. This would indicate that models often produce text that misrepresents identities or renders them marginal. Allocative harm, which includes unfair treatment and denial of resources, appears less frequent but still visible in areas such as age and socioeconomic bias. The both category describes cases where outputs exhibit a combination of representational and allocative harm and indicate more complex forms of discrimination. Bias based on race reveals a lower overall count, but concentrated harms are still likely to be important. This figure shows that harmful outputs are complex and different for each group, with some categories having worse or more complex harms than others.

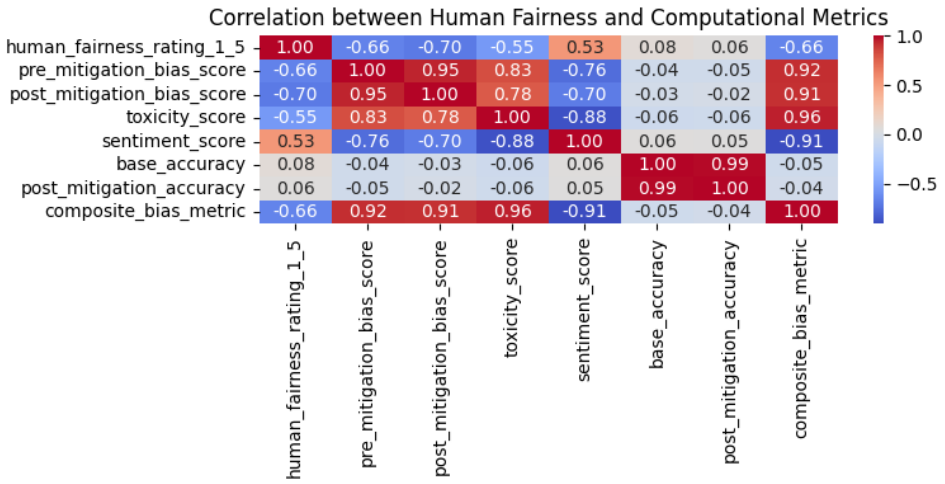


Figure 3.4: Correlation between Human Fairness and Computational Metrics

Above diagram 3.4 heatmap visualizes the relationships between human fairness ratings and a number of computational metrics used to assess model outputs. Human fairness scores correlate strongly negatively both with bias scores before and after mitigation, which would imply that higher levels of bias are related to lower levels of perceived fairness. Toxicity correlates moderately negative, while sentiment is positive. This supports the argument that harmful language decreases fairness assessments, while more positive outputs are perceived as fairer. No strong relationship exists either way between fairness and accuracy metrics. This finding supports the position that performance itself does not strongly determine how fair users perceive a system to be. The composite bias metric strongly relates to the individual measures of bias; this would confirm that it effectively summarizes problematic content traits. Overall, the heatmap reflects that human fairness perceptions are more driven by quality and harms of content rather than by accuracy or efficiency, suggesting that improvements in fairness can best be achieved by reducing harmful signals rather than merely optimizing performance.

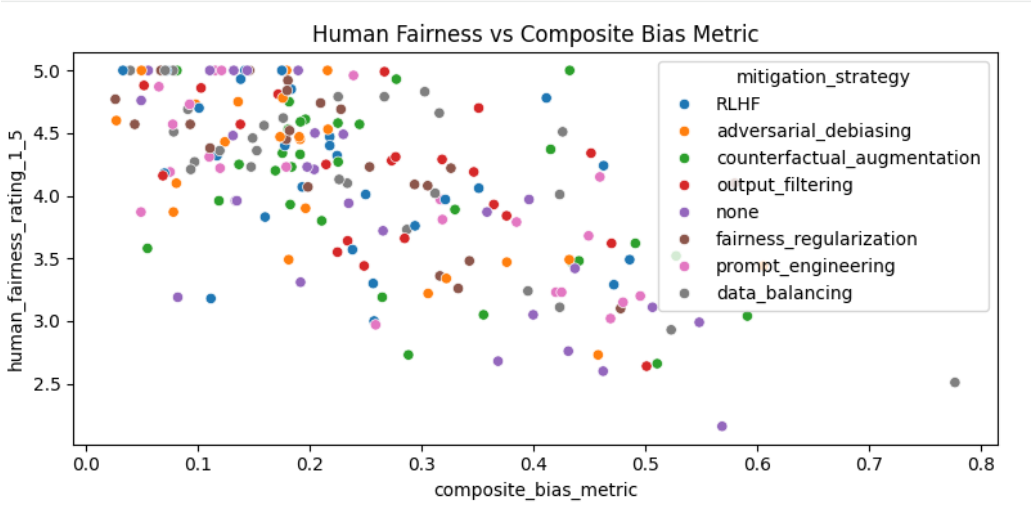


Figure 3.5: Scatter Plot for Human Fairness vs Composite Bias Metric

Above diagram 3.5 scatter plot below illustrates how human fairness ratings relate to the composite bias metric for different mitigation strategies. A downward trend is immediately apparent: as the composite bias metric increases, fairness ratings tend to fall. This reflects the fact that the greater the amount of harmful content operationalized here using a composite measure of bias, toxicity, and negative sentiment the less fair the outputs are perceived to be by users. The dispersion of points also reflects variation between mitigation strategies: whereas some methods yield mostly low bias, high-fairness outputs, others yield more scattered and inconsistent results. For instance, RLHF, adversarial debiasing, and fairness regularisation are all strategies that lower bias scores and are grouped together on the left side of the graph. These results suggest improved methods for regulating dangerous content. Other tactics, such prompt engineering or output filtering, on the other hand, lead to outputs that are less similar. Overall, this plot shows that better mitigation measures make things seem more fair, and it also shows that eliminating signs of damaging language is important for getting fair model behaviours.

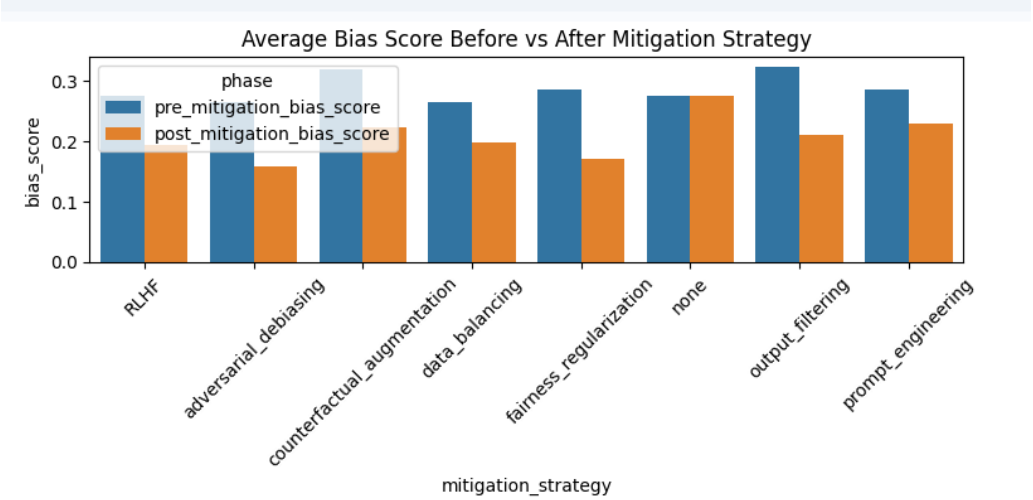


Figure 3.6: Bar Plot for Average Bias Score Before vs After Mitigation Strategy

The bar chart 3.6 compares the average bias scores before and after different tactics were used to reduce bias. The bars demonstrate a clear decrease in bias following mitigation for most methodologies. This means that interventions usually make things fairer. Adversarial debiasing and fairness regularisation show big drops, which means they work to lower harmful content. Data balance also makes things better in a big way while keeping the levels of pre-mitigation moderate. The "none" condition, on the other hand, shows no change, which means that bias stays the same even when nothing is done. RLHF works well and shows a significant drop in bias from a more mild starting point.

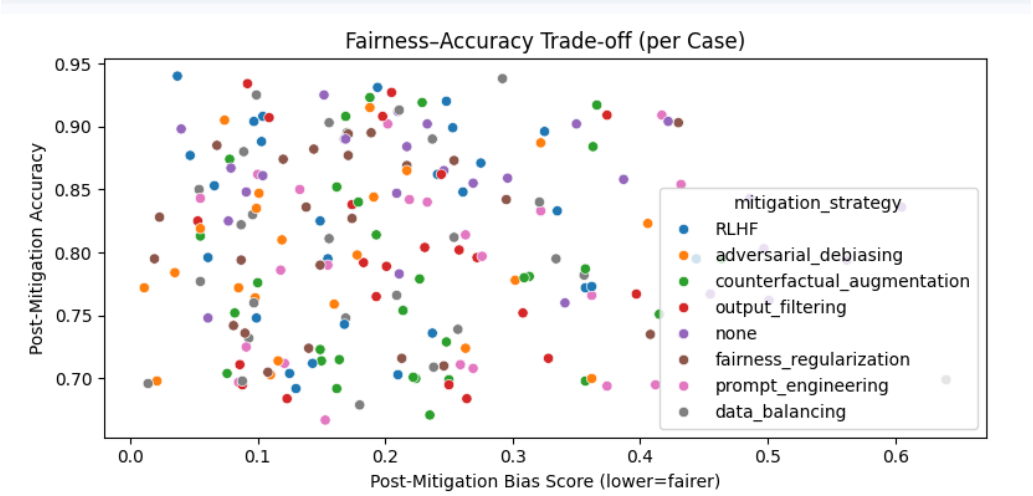


Figure 3.7: Scatter Plot for Fairness Accuracy Trade off

Above diagram 3.7 scatter plot below charts the relationship between post-mitigation accuracy and residual bias across individual model outputs. There is some variation but no strict trade-off apparent from this plot where improving fairness necessarily reduces accuracy. Many of the points cluster in areas with both low bias and high accuracy. This would suggest that effective mitigation does not always compromise model performance. However, points with higher bias scores often achieve similar or slightly lower accuracy levels, indicating that unmitigated bias also does not yield superior performance. Different mitigation strategies are scattered all across this plot: indeed, no single method obviously dominates in balancing fairness with performance, though RLHF, adversarial debiasing, and data balancing approaches do appear quite regularly in high-performing regions. Overall, the visual pattern seems to be that there is no need for fairness optimization to necessarily come at large accuracy losses, which supports the viability of ethical AI interventions that advance equity without sacrificing reliability.

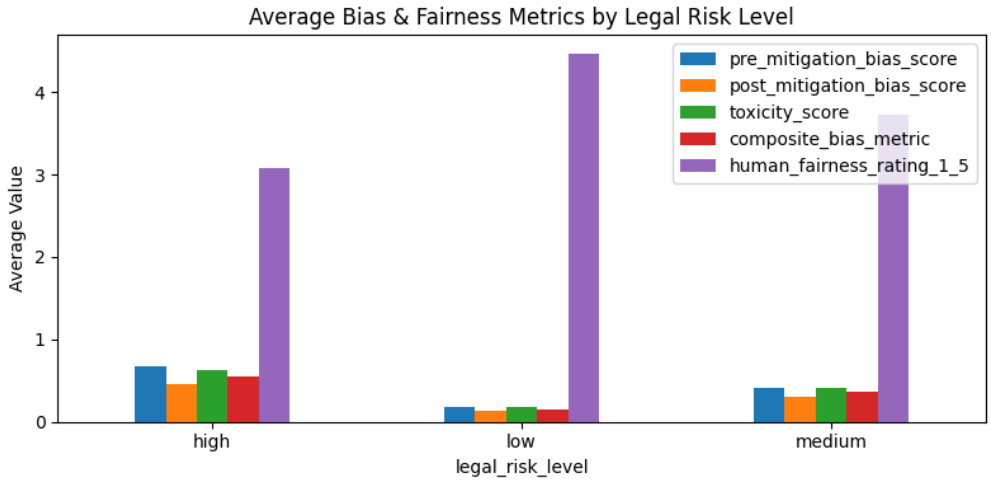


Figure 3.8: Bar Plot for Average Metrics by Legal Risk Level

Above diagram 3.8 bar chart compares the average fairness and harm metrics across the three legal risk levels: high, medium, and low. Outputs in the low-risk category exhibit the lowest bias scores and toxicity levels, as well as the highest human fairness ratings, indicating that these are considered fairer and less harmful. High-risk outputs clearly have more significant levels of bias, toxicity, and composite harm, which correspondingly places them into legally sensitive or potentially damaging categories. Medium-risk cases fall between these extremes, reflecting moderate levels of risk and harm. The wide variation in fairness ratings among

levels of risk indicates that perceived legal risk is strongly linked with harmful content and unfairness. These findings show that fairness concerns are not only ethical issues but also legal liabilities. Reducing the harm of the outputs may thus help reduce legal risk exposure and establish the value of fairness aware design for regulatory and compliance-sensitive domains.

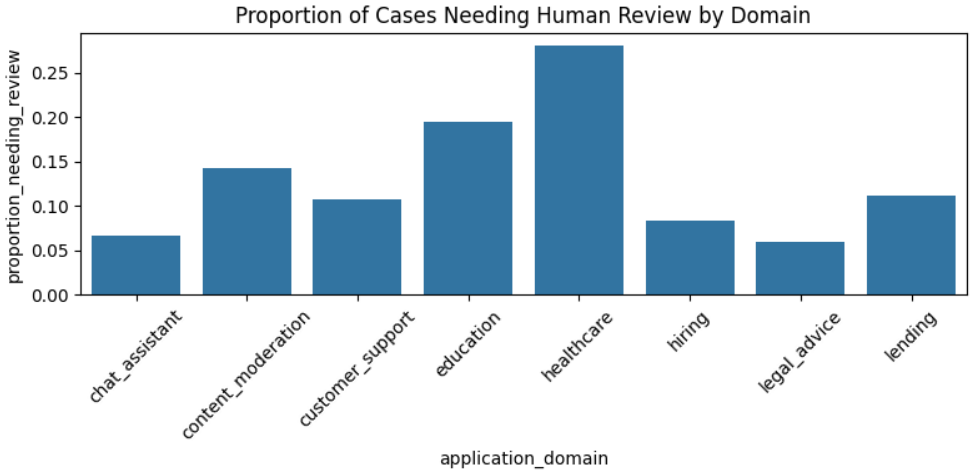


Figure 3.9: Bar Plot for Proportion of Cases Needing Human Review by Domain

Above diagram 3.9 bar chart below compares how often the outputs across different application domains may need human review because of their possible bias or harm. Healthcare has the highest proportion for human oversight, indicating the complexity and sensitivity of the decisions being made within medical or health-related contexts. The next most affected domain is education, which would indicate that concerns around fairness are important when models generate content that influences either learning or academic opportunities. Content moderation and customer support fall in a mid-range, reflecting the presence of harmful or inappropriate language but with relatively low severity. Sectors such as hiring, legal advice, and chat assistance have relatively low proportions, though this does not eliminate the need for review entirely. These results suggest that the risk of unfair outcomes associated with LMs is distributed unevenly across domains, with regulatory or ethical exposures higher in some sectors than others. The chart thus underlines the importance of human involvement in high-risk areas to ensure safe, responsible deployment of language models.

CHAPTER 4

Research Methodology

4.1 Logistic Regression

Logistic regression is a statistical model used to predict the probability of yes/no outcomes-for example, whether a customer will buy a product or whether an email is spam. As such, it doesn't predict a continuous number but one that lies between 0 and 1. It does this by taking the input features, multiplying them by weights, and passing the result through a special S-shaped function called the sigmoid. This function converts any number into a probability. Afterwards, a threshold-often 0.5-is used to decide the final class. The model learns by adjusting its weights to decrease prediction errors. Logistic regression remains popular, as it is simple to understand, interpret, and often works well for many problems when the relationships among features and outcomes are mainly linear [Kanjee, 2007].

Logistic Regression is the baseline model, used for classifying the output of LLM either as biased or non-biased. It basically learns the relationship between input features, which include bias scores, toxicity, sentiment, and other contextual attributes, and the probability of the output being biased. Because it is linear and interpretable, it helps reveal which features most strongly influence bias. The model outputs predicted class labels along with probability scores. Its performance can be evaluated in terms of accuracy, precision, recall, F1-score, and Cohen's Kappa.

4.1.1 Model Setup

We want to predict a binary output $y \in \{0, 1\}$ from an input vector $\mathbf{x} \in \mathbb{R}^d$.

First, we form a linear combination of the inputs:

$$z = \mathbf{w}^\top \mathbf{x} + b$$

where $\mathbf{w}$ is the weight vector and b is the bias (intercept).

4.1.2 Sigmoid (Logistic) Function

To turn z into a probability between 0 and 1, we use the sigmoid:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

The model predicted probability that $y = 1$ is:

$$\hat{p}(y = 1 \mid \mathbf{x}) = \sigma(\mathbf{w}^\top \mathbf{x} + b)$$

and

$$\hat{p}(y = 0 \mid \mathbf{x}) = 1 - \hat{p}(y = 1 \mid \mathbf{x})$$

4.1.3 Logit (Inverse Sigmoid)

The log-odds (logit) is the inverse of the sigmoid:

$$\text{logit}(p) = \log\left(\frac{p}{1 - p}\right)$$

In logistic regression,

$$\text{logit}\big(\hat{p}(y = 1 \mid \mathbf{x})\big) = \log\left(\frac{\hat{p}(y = 1 \mid \mathbf{x})}{1 - \hat{p}(y = 1 \mid \mathbf{x})}\right) = \mathbf{w}^\top \mathbf{x} + b$$

This shows the log-odds are a linear function of the inputs.

4.1.4 Decision Rule

To make a hard yes/no prediction, we apply a threshold (often 0.5):

$$\hat{y} = \begin{cases} 1, & \text{if } \hat{p}(y = 1 \mid \mathbf{x}) \geq 0.5 \\ 0, & \text{otherwise} \end{cases}$$

4.1.5 Likelihood and Loss Function

Given a training set $\{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^{N}$, the probability of observing each label is:

$$\hat{p}\left(y^{(i)} \mid \mathbf{x}^{(i)}\right) = \left(\hat{p}(y=1 \mid \mathbf{x}^{(i)})\right)^{y^{(i)}} \left(1 - \hat{p}(y=1 \mid \mathbf{x}^{(i)})\right)^{1-y^{(i)}}$$

The likelihood of all data is:

$$L(\mathbf{w}, b) = \prod_{i=1}^{N} \hat{p}\left(y^{(i)} \mid \mathbf{x}^{(i)}\right)$$

Instead of maximizing L, we minimize the negative log-likelihood, also called the logistic (cross-entropy) loss:

$$\mathcal{L}(\mathbf{w}, b) = -\sum_{i=1}^{N} \left[y^{(i)} \log \hat{p}(y=1 \mid \mathbf{x}^{(i)}) + (1-y^{(i)}) \log\left(1 - \hat{p}(y=1 \mid \mathbf{x}^{(i)})\right)\right]$$

4.1.6 Gradient for Learning

For a single example $(\mathbf{x}, y)$, the gradient of the loss with respect to the weights is:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}} = (\hat{p}(y=1 \mid \mathbf{x}) - y)\,\mathbf{x}$$

and with respect to the bias:

$$\frac{\partial \mathcal{L}}{\partial b} = \hat{p}(y=1 \mid \mathbf{x}) - y$$

Gradient descent updates the parameters by moving them in the opposite direction of the gradient:

$$\mathbf{w} \leftarrow \mathbf{w} - \eta \frac{\partial \mathcal{L}}{\partial \mathbf{w}}, \quad b \leftarrow b - \eta \frac{\partial \mathcal{L}}{\partial b}$$

where η is the learning rate.

4.1.7 Working Principle

Logistic regression predicts the probabilities of a binary response, such as 0/1 or yes/no. It takes input features, computes a weighted sum representing how much each feature influences the outcome, and passes that through a sigmoid function to produce a probability in the range between 0 and 1. The model then compares this probability against a threshold usually 0.5 above which it classifies a case as belonging to class 1, otherwise as class 0. During its learning phase, its objective is to find weights that best separate the two classes by

adjusting the weights in such a way as to minimize the difference between predictions and actual values. It does this iteratively until such weights are found that better differentiate the two classes. Overall, logistic regression is very simple, interpretable, and works well when relationships are majorly linear.

4.1.8 Advantages and Disadvantages

Advantages:

- Helps to identify hidden discrimination in AI systems and reduces harm to marginalized groups.
- Improves trust, transparency, and accountability in AI decision making.
- Allows for more diversified and representative data, thus improving model performance.
- Supports compliance with legal and regulatory requirements on discrimination.
- Improves public acceptance of AI technologies in sensitive domains such as hiring, healthcare, and policing.

Disadvantages:

- Detection of bias is complicated and often subjective, as it is based on the definition of fairness.
- Bias mitigation can reduce model accuracy if fairness constraints conflict with performance.
- Diverse data collection can increase costs, time, and privacy risks.
- Technical solutions to problems often camouflage systemic societal issues rather than solve them.
- Fairness methods may be difficult to scale in large, real-world systems.
- Stakeholders can disagree on the choice of a fairness metric, which may create conflicts.

4.2 Random Forest Method

Random Forest is a machine learning model which ensembles many decision trees for more accurate and stable prediction. Instead of constructing one complex tree, it builds multiple trees with various random samples of the data. Each of the trees makes a prediction, and the final result is taken as an outcome of majority voting in the case of classification or averaging in numerical prediction [Bader-El-Den et al., 2018]. Random Forest further uses random subsets of features at every step such that trees learn different patterns; the risk of overfitting thus reduces. It generally outperforms a single decision tree because it combines weak models into one strong one. It can handle large datasets, missing values, and non-linear relationships between variables. Nevertheless, it can be slower to train and harder to interpret since there are a large number of trees involved. Overall, Random Forest is a reliable and usually used model in data science [Jiang et al., 2021].
Random Forest now analyses more complicated, non-linear patterns in the way bias manifests in various LLM outputs. It constructs multiple decision trees out of random subsets of features and combines their predictions to attain a more accurate and robust model. This model is especially effective in cases when the interactions between the features-like bias type, harm type, and sentiment-affect the classification of bias. The model outputs class predictions as the majority vote of its trees. Classification reports and confusion matrices are generated based on its results.

4.2.1 Notation

Let the training data be

$$\{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^{N}, \quad \mathbf{x}^{(i)} \in \mathbb{R}^d.$$

A Random Forest contains T trees:

$$\{h_1, h_2, \ldots, h_T\}.$$

4.2.2 Bootstrap Sampling

For each tree t, we draw a bootstrap sample (a sample with replacement) from the original data:

$$\mathcal{D}_t \sim \text{Bootstrap}(\mathcal{D}).$$

Each tree h_t is trained on $\mathcal{D}_t$.

4.2.3 Node Splitting (Classification)

At each node, we choose a random subset of features

$$\mathcal{F}_{\text{sub}} \subset \{1, 2, \ldots, d\}.$$

For each candidate split s (on a feature in $\mathcal{F}_{\text{sub}}$), we compute an impurity measure, such as Gini impurity:

$$G(S) = \sum_{k=1}^{K} p_k(1 - p_k),$$

where p_k is the proportion of class k in node set S.

For a split s that partitions S into S_L and S_R, the impurity after the split is:

$$G_{\text{split}}(s) = \frac{|S_L|}{|S|} G(S_L) + \frac{|S_R|}{|S|} G(S_R).$$

We choose the split that minimizes $G_{\text{split}}(s)$:

$$s^* = \arg\min_s G_{\text{split}}(s).$$

4.2.4 Prediction: Classification

Each tree gives a class prediction $h_t(\mathbf{x})$. The Random Forest prediction is a majority vote:

$$\hat{y} = \text{mode}\big(h_1(\mathbf{x}), h_2(\mathbf{x}), \ldots, h_T(\mathbf{x})\big).$$

This means the class chosen most often by the trees is the final output.

4.2.5 Prediction: Regression

For regression, each tree outputs a numeric value $h_t(\mathbf{x})$. The Random Forest prediction is the average:

$$\hat{y} = \frac{1}{T} \sum_{t=1}^{T} h_t(\mathbf{x}).$$

This averaging reduces variance and makes predictions more stable.

4.2.6 Why Randomness Helps

- Bootstrap sampling makes each tree see a slightly different dataset.
- Random feature selection at each split makes trees learn different patterns.

4.2.7 Working Principle

Random Forest combines the predictions of many decision trees to derive the final and more accurate result. Instead of building one large tree, it constructs many smaller trees using random samples of the data. Each tree learns patterns independently and makes its prediction. The randomness leads the trees to become different from each other and thus prevents them from making the same mistakes, which reduces overfitting.
For classification tasks, the model utilizes a majority vote; the class most predicted by the trees becomes the final output. In regression tasks, it averages values predicted across all trees. During the training phase, each tree chooses random features at each split, which in turn makes different decision paths.

4.2.8 Advantage and Disadvantage

Advantages: It has a lower risk of overfitting than a single decision tree due to the addition of randomness and averaging. Random Forest is relatively robust to noisy data, missing values, and outliers. It provides feature importance measures that help users understand which factors drive predictions. Additionally, scaling across multiple processors is good, hence useful for big data challenges.
Disadvantages: Random Forest can be very computationally expensive and requiring of memory, since it builds a large number of trees. The training and prediction can be slower compared to simpler models. The model is harder to interpret, since hundreds of trees are combined, which makes it less transparent than a single tree. It may have problems working with datasets that have many irrelevant features unless a feature selection is applied. Randomness may lead to slightly different results every time this algorithm is run, reducing the reproducibility. Large and highly imbalanced numbers of classes may result in poor performance of the model.

4.3 Support Vector Machine

The Support Vector Machine (SVM) is a machine learning model primarily intended for classification. Its objective is to find the best boundary a hyperplane-that separates various classes in the data. SVM works on maximizing the margin, which is the distance between the hyperplane and the nearest data points of each class. These nearest points are termed support vectors, and they determine the position of the boundary [Valentini and Dietterich, 2004].
It does well in scenarios where the data is non-linearly separable because of the use of kernel functions. A kernel basically transfers your data into a higher dimensional space in which it becomes easier to separate classes using a straight line. The most common types of kernels include linear, polynomial, and RBF [Luo and Paal, 2021].
While SVM is powerful, accurate, and works remarkably well in high-dimensional spaces, it is rather slow with large datasets. It also requires proper selection of kernels and parameters to perform at its best.
SVM will seek the best boundary separating biased from non-biased outputs. The fact that the separation is probably nonlinear will be handled by using an RBF kernel, one of the choices that makes SVM work well for this task. The model works by mapping inputs into a higher-dimensional space, in which the separation becomes clearer. It thus outputs predicted classes depending on which side of the decision boundary a sample falls. The performance is assessed by using the metrics of precision, recall, F1-score, and confusion metrics.

We assume training data

$$\{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^{N}, \quad \mathbf{x}^{(i)} \in \mathbb{R}^d,\ y^{(i)} \in \{-1, +1\}.$$

4.3.1 Hyperplane and Decision Function

A hyperplane is defined by weights $\mathbf{w}$ and bias b:

$$f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b.$$

The predicted class is:

$$\hat{y} = \operatorname{sign}\big(f(\mathbf{x})\big) = \operatorname{sign}\big(\mathbf{w}^\top \mathbf{x} + b\big).$$

4.3.2 Margin

The margin is the distance between the hyperplane and the closest points. For SVM, we want to *maximize* the margin:

$$\text{margin} = \frac{2}{\|\mathbf{w}\|}.$$

Maximizing the margin is equivalent to minimizing $\|\mathbf{w}\|$.

4.3.3 Hard-Margin SVM (No Misclassification)

If the data are perfectly separable, the optimization problem is:

$$\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2$$

subject to

$$y^{(i)}(\mathbf{w}^\top \mathbf{x}^{(i)} + b) \geq 1, \quad i = 1, \ldots, N.$$

These constraints force all points to be on the correct side of the margin.

4.3.4 Soft-Margin SVM (Allows Some Errors)

Real data are often noisy, so we allow violations using slack variables $\xi_i \geq 0$:

$$\min_{\mathbf{w},b,\boldsymbol{\xi}} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N} \xi_i$$

subject to

$$y^{(i)}(\mathbf{w}^\top \mathbf{x}^{(i)} + b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad i = 1, \ldots, N.$$

Here, $C > 0$ controls the trade-off between a large margin and allowing misclassifications. Large C means we punish errors more strongly.

4.3.5 Hinge Loss View

The soft-margin SVM can also be seen as minimizing hinge loss:

$$\ell_{\text{hinge}}\big(y^{(i)}, f(\mathbf{x}^{(i)})\big) = \max\big(0, 1 - y^{(i)} f(\mathbf{x}^{(i)})\big).$$

The objective becomes:

$$\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N} \max\big(0, 1 - y^{(i)}(\mathbf{w}^\top \mathbf{x}^{(i)} + b)\big).$$

4.3.6 Dual Form (For Kernels)

The dual problem uses coefficients α_i (one per data point):

$$\max_{\boldsymbol{\alpha}} \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y^{(i)} y^{(j)} \mathbf{x}^{(i)\top} \mathbf{x}^{(j)}$$

subject to

$$0 \leq \alpha_i \leq C, \quad \sum_{i=1}^{N} \alpha_i y^{(i)} = 0.$$

The weight vector is:

$$\mathbf{w} = \sum_{i=1}^{N} \alpha_i y^{(i)} \mathbf{x}^{(i)}.$$

4.3.7 Kernels

With a kernel function $K(\mathbf{x}, \mathbf{z})$, we replace dot products:

$$\mathbf{x}^{(i)\top} \mathbf{x}^{(j)} \quad \rightarrow \quad K(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}).$$

Common kernels:

$$\text{Linear: } K(\mathbf{x}, \mathbf{z}) = \mathbf{x}^\top \mathbf{z},$$

$$\text{Polynomial: } K(\mathbf{x}, \mathbf{z}) = (\gamma \mathbf{x}^\top \mathbf{z} + r)^d,$$

$$\text{RBF (Gaussian): } K(\mathbf{x}, \mathbf{z}) = \exp\left(-\gamma \|\mathbf{x} - \mathbf{z}\|^2\right).$$

4.3.8 Decision Function with Kernels

In the kernel form, the decision function is:

$$f(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i y^{(i)} K(\mathbf{x}^{(i)}, \mathbf{x}) + b,$$

and the predicted label is still

$$\hat{y} = \text{sign}\big(f(\mathbf{x})\big).$$

4.3.9 Working Principle

The basic idea of SVM is to find the best separating boundary among classes. Instead of drawing just any line, it searches for the one that maximizes the margin-the distance between the boundary and the closest data points from each class. These critical points, called support vectors, bear a direct impact on the final decision boundary.
If the data cannot be separated by a straight line, then SVM uses kernels to transform it into higher dimensional space where separation becomes easier. During training, the SVM balances creating a wide margin with allowing some mistakes, controlled by a parameter. Once trained, SVM classifies new data according to which side of the boundary they fall on. It turns out to be effective, robust, and works well with complex, high dimensional data.

4.3.10 Advantages and Disadvantages

Advantages: SVM provides high accuracy, especially when there is a clear margin between classes. It works well in high-dimensional spaces and is effective when the number of features is greater than the number of samples. The model just uses support vectors instead of the complete dataset, which saves memory. SVM can work with non-linear data by utilising kernel functions to change it into higher dimensions where the classes can be separated. It is also fairly resistant to overfitting, especially when the parameters are set correctly. SVM works well for things like classifying text, recognising images, and bioinformatics.
Disadvantages: SVM is not very effective if data are noisy or overlapping or if they contain many irrelevant features. SVM becomes more complicated in multi-class problems; it naturally fits binary classification. The model is also less interpretable than simple ones, such as logistic regression. Finally, SVM's performance may drop when datasets are extremely large, highly imbalanced, or require real-time prediction.

4.4 Linear Regression Model

Linear regression is a simple model that's widely used to predict a numerical outcome based on one or more input features. It assumes a linear relationship between the inputs and the output. This model draws a straight line-or in higher dimensions, a plane-that best fits the data points. That line is defined by weights and an intercept; these define its slope and its

position [Lim et al., 2023].

Linear regression tries to minimize the difference between the predicted and actual values during training by finding a line that does so. In most cases, this is done by minimizing the sum of squared errors. Once trained, new data can be fed into the equation and an output predicted.

Linear regression is intuitive, quick to calculate, and works well if the relationships are truly linear but suffers in cases of complex patterns or noisy data.

The same contextual and computational metrics are used in the Linear Regression model to predict human fairness ratings on a continuous scale. The model will learn from numerical and encoded features like sentiment, toxicity, and mitigation strategy how they relate to human judgments about fairness. Each test sample will get a predicted fairness score after going through the model. It highlights how well these algorithmic signals approximate actual human ethical evaluation. Model performances were assessed using Mean Squared Error MSE , showing how well it captures the trends of fairness.

4.4.1 Model Setup

We want to predict a numeric output y from an input vector $\mathbf{x} \in \mathbb{R}^d$.

For one example, the linear regression model is:

$$\hat{y} = \mathbf{w}^\top \mathbf{x} + b$$

where:

- $\mathbf{w}$ is the weight (coefficient) vector,
- b is the bias (intercept),
- $\hat{y}$ is the predicted value.

4.4.2 Matrix Form

For N examples, let

$$\mathbf{X} = \begin{bmatrix} - & \mathbf{x}^{(1)\top} & - \\ - & \mathbf{x}^{(2)\top} & - \\ & \vdots & \\ - & \mathbf{x}^{(N)\top} & - \end{bmatrix} \in \mathbb{R}^{N \times d}, \quad \mathbf{y} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(N)} \end{bmatrix}.$$

Predictions for all points can be written as:

$$\hat{\mathbf{y}} = \mathbf{X}\mathbf{w} + b\mathbf{1},$$

where $\mathbf{1}$ is a vector of ones.

Often we absorb b into $\mathbf{w}$ by adding a column of ones to $\mathbf{X}$.

4.4.3 Loss Function (Mean Squared Error)

We want $\hat{y}^{(i)}$ to be close to $y^{(i)}$ for each example. A common choice is the mean squared error (MSE):

$$\mathcal{L}(\mathbf{w}, b) = \frac{1}{N}\sum_{i=1}^{N}\left(\hat{y}^{(i)} - y^{(i)}\right)^2.$$

In matrix form (without the constant $1/N$):

$$\mathcal{L}(\mathbf{w}) = \|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2.$$

4.4.4 Normal Equation (Closed-Form Solution)

If $\mathbf{X}^\top\mathbf{X}$ is invertible, we can directly compute the best weights (least squares solution):

$$\mathbf{w}^* = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y}.$$

This $\mathbf{w}^*$ minimizes the squared error.

4.4.5 Gradient Descent Solution

For large datasets, we often use gradient descent.

The gradient of the loss with respect to $\mathbf{w}$ is:

$$\nabla_{\mathbf{w}}\mathcal{L} = \frac{2}{N}\mathbf{X}^\top(\mathbf{X}\mathbf{w} - \mathbf{y}).$$

Gradient descent update rule:

$$\mathbf{w} \leftarrow \mathbf{w} - \eta\,\nabla_{\mathbf{w}}\mathcal{L},$$

where η is the learning rate (step size).

4.4.6 Simple Linear Regression (One Feature)

If we have one feature x and one output y:

$$\hat{y} = wx + b.$$

We choose w and b to minimize:

$$\mathcal{L}(w, b) = \frac{1}{N} \sum_{i=1}^{N} (wx^{(i)} + b - y^{(i)})^2.$$

4.4.7 Coefficient of Determination (R^2)

R^2 measures how well the model explains the variance in y:

$$R^2 = 1 - \frac{\sum_{i=1}^{N} (y^{(i)} - \hat{y}^{(i)})^2}{\sum_{i=1}^{N} (y^{(i)} - \bar{y})^2},$$

where $\bar{y}$ is the mean of the true values. An R^2 closer to 1 means a better fit.

4.4.8 Working Principle

Linear regression works by determining the straight line that most accurately represents the relationship between input features and a numerical output. It assumes changes in the input result in proportionate changes in the output. A model assigns weights to each feature, combines them with an intercept, and makes predictions.
The model tries out different weight values during training and sees how close the predictions are to the real values. The model calculates error by squaring the difference between the expected and actual output. It seeks for the weights that make this total mistake as small as possible. This can be done using either direct mathematical formulas or iterative optimisation methods like gradient descent.

4.4.9 Advantages and Disadvantages

Advantages: Linear regression is simple to understand and use, which is why it is often the first step in predictive modelling. It makes it easy to see how the input features affect the output by using coefficients that are easy to understand. It doesn't take long to train the model, and it performs well with big datasets. It works well when the relationship

between variables is linear and there isn't much noise..It is less prone to overfitting when regularization techniques are used. Besides, its predictions are fast, making it suitable for real-time applications.

Disadvantages: As can be seen, linear regression is not good at relationships that are non linear because it presumes inputs and outputs are related in a straight-line fashion. It will perform poorly when the dataset has high noise, outliers, or multicollinearity. It is sensitive to extreme values, distorting this model. It may oversimplify complex patterns and reduce the accuracy of predictions. Linear regression assumes normality, homoscedasticity, and independence, all of which are important assumptions that can reduce its reliability when violated. Also, this model cannot handle feature interaction without explicit feature engineering. Finally, this may be out of date compared with state of the art models like decision trees, neural networks, and ensemble methods.

4.5 Evaluation Metrics

- Accuracy: It refers to the number of correct predictions the model makes out of the total number of predictions. It is calculated as correctly predicted cases divided by the total number of cases. It is intuitive but may be misleading when classes are imbalanced for instance, predicting all cases as a majority class may still give high accuracy.

- Kappa Score (Cohen's Kappa): Kappa score: This checks how well the model performs versus random guessing. It measures the agreement between predicted labels and actual labels while correcting for chance agreement.

- Classification Report: A Classification report uses metrics like precision, recall, and F1 score to indicate how well each class did. It gives more specific information on how well a model works for each class, not just an overall success rate.

- Confusion Matrix: A confusion matrix is a kind of table showing how many predictions were correct or incorrect for each class. It shows true positives, false positives, true negatives, and false negatives. It helps to identify where the model is making mistakes.

- R2 Score (Coefficient of Determination): R2 is utilized in regression to describe the proportion of the variance in the target values explained by this model. A Close to 1

indicates that most of the variance is described by the model, while 0 indicates that it has explained nothing.

- Mean Squared Error: MSE calculates the average of the squared differences between predicted and actual values. Since the errors are squared, large mistakes are penalized hard. Lower MSE means better performance in regression tasks.

For a binary classifier, we define:

	Predicted Positive	Predicted Negative
Actual Positive	TP	FN
Actual Negative	FP	TN

where:

$TP =$ True Positives, $\quad FP =$ False Positives, $\quad FN =$ False Negatives, $\quad TN =$ True Negatives.

4.5.1 Accuracy

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}.$$

4.5.2 Classification Report Metrics

Precision (for a class)

$$\text{Precision} = \frac{TP}{TP + FP}.$$

Recall (Sensitivity)

$$\text{Recall} = \frac{TP}{TP + FN}.$$

F1-Score

$$\text{F1} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}.$$

4.5.3 Cohen's Kappa Score

Let p_o be the observed accuracy and p_e the expected accuracy by chance.

Observed Agreement

$$p_o = \frac{TP + TN}{TP + TN + FP + FN}.$$

Expected Agreement by Chance

For binary classification, with:

$$p_{\text{actual pos}} = \frac{TP + FN}{N}, \quad p_{\text{actual neg}} = \frac{FP + TN}{N},$$
$$p_{\text{pred pos}} = \frac{TP + FP}{N}, \quad p_{\text{pred neg}} = \frac{FN + TN}{N},$$

the expected agreement is:

$$p_e = p_{\text{actual pos}} \cdot p_{\text{pred pos}} + p_{\text{actual neg}} \cdot p_{\text{pred neg}}.$$

Kappa

$$\kappa = \frac{p_o - p_e}{1 - p_e}.$$

4.5.4 Regression Metrics

Assume N samples, with true values y_i and predictions $\hat{y}_i$.

Mean Squared Error (MSE)

$$\text{MSE} = \frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2.$$

R^2 Score (Coefficient of Determination)

Let

$$\bar{y} = \frac{1}{N} \sum_{i=1}^{N} y_i$$

be the mean of the true values.

Total Sum of Squares (TSS):

$$\text{TSS} = \sum_{i=1}^{N} (y_i - \bar{y})^2.$$

Residual Sum of Squares (RSS):

$$\text{RSS} = \sum_{i=1}^{N} (y_i - \hat{y}_i)^2.$$

Then:

$$R^2 = 1 - \frac{\text{RSS}}{\text{TSS}} = 1 - \frac{\sum_{i=1}^{N} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{N} (y_i - \bar{y})^2}.$$

CHAPTER 5

Result and Discussion

Class	Precision	Recall	F1-Score	Support
0 (Non-Biased)	0.97	0.84	0.90	38
1 (Biased)	0.65	0.92	0.76	12
Accuracy			**0.86**	50
Macro Avg	0.81	0.88	0.83	50
Weighted Avg	0.89	0.86	0.87	50

Table 5.1: Classification Report for Logistic Regression Bias Detection Model

Table 5.1 classification results for the Logistic Regression model indicate that it performs well in distinguishing between biased and non-biased outputs. Indeed, it shows an overall accuracy of 0.86, which evidences that it predicts the class label correctly in most of the cases. The nonbiased class achieves high precision of 0.97 and a strong F1-score of 0.90, meaning that predictions for this class are both accurate and consistent. Although the biased class has a lower precision at 0.65, it still can demonstrate very high recall at 0.92, indicating that it is highly sensitive when biased output is to be predicted by the model. This is vital in applications involving fairness because failing to detect harmful or discriminatory content may have serious consequences with significant social and ethical impacts.

The macro average values indicate balanced performance for both classes, while the weighted averages take class imbalance into consideration and further confirm the stability of overall performance. This implies that the model Logistic Regression works well for the detection

of bias, but future improvements may aim at enhancing the precision of the biased class in order to reduce false positives. However, it provides a very strong baseline for ethical AI model evaluation.

Class	Precision	Recall	F1-Score	Support
0 (Non-Biased)	0.97	0.89	0.93	38
1 (Biased)	0.73	0.92	0.81	12
Accuracy			**0.90**	50
Macro Avg	0.85	0.91	0.87	50
Weighted Avg	0.91	0.90	0.90	50

Table 5.2: Classification Report for Random Forest Bias Detection Model

Table 5.2 Random Forest model yields very high performance in distinguishing between biased and non-biased outputs, with an overall accuracy of 0.90. It performs even better for the non biased class, with a precision of 0.97 and an F1-score of 0.93, indicating a reliable identification of content that expresses no harmful or discriminatory patterns. The biased class performs well, especially considering that this class is usually harder to predict due to class imbalance. This class achieves a recall of 0.92 and an F1 score of 0.81. The high recall is particularly important, especially in fairness-sensitive applications, because it reflects the ability of the model to catch most biased cases without overlooking them.

The macro and weighted averages back up this result by showing that all classes did equally well, even though the number of students in each class was different. Random Forest is better than simpler models because it can model complicated interactions between features, which lets it find little signs of injustice or prejudice. Taken together, these results reinforce the notion that Random Forest is a proficient and robust classifier to support ethical AI evaluation frameworks.

Class	Precision	Recall	F1-Score	Support
0 (Non-Biased)	0.94	0.84	0.89	38
1 (Biased)	0.62	0.83	0.71	12
Accuracy			**0.84**	50
Macro Avg	0.78	0.84	0.80	50
Weighted Avg	0.87	0.84	0.85	50

Table 5.3: Classification Report for SVM (RBF) Bias Detection Model

Table 5.3 While performing rather well on most of the biased versus non-biased language outputs, the SVM model with an RBF kernel has generally adequate performance, reaching an overall accuracy of 0.84. It yielded very high precision for the non-biased class at 0.94 and a very strong F1 score of 0.89, indicating that this model is good at capturing examples without biases. However, precision for the biased class was considerably lower (0.62), meaning if the model predicted bias, it sometimes was incorrect. Even so, for the biased class, recall was rather high at 0.83, showing that the model captures many biased instances but does so with reduced precision.

Both the macro and weighted averages again support that the model performs reasonably well for both classes, but with marked trade offs. Compared to more complex models, SVM may struggle with complex nonlinear decision boundaries entailed in socio linguistic data. In any case, this makes it very valuable in the context of fairness, where the cost of failing to detect harmful content is high. Overall, the SVM model provides a reliable baseline; however, more work needs to be done to increase its precision for biased predictions.

Model	Accuracy	Kappa Score
Logistic Regression	0.860	0.664
Random Forest	0.900	0.747
SVM (RBF)	0.840	0.606

Table 5.4: Performance Comparison of Three Classification Models for Bias Detection

Table 5.4 results of the different performance metrics clearly bring out the varied performances of the three classification models: Random Forest has the highest accuracy of 0.900 and a Kappa score of 0.747. This indicates a high predictive capability and that there was

strong agreement beyond chance. This therefore suggests that Random Forest is relatively better in picking out complex patterns present in the data, which has to do with biased and non-biased outputs. Logistic Regression does moderately well, reaching an accuracy of 0.860 and a Kappa score of 0.664, signifying good but less stable performance compared to the Random Forest model. The recorded lowest accuracy of 0.840 and Kappa score of 0.606 are for the SVM model, indicating weaker reliability in bias detection. However, substantial agreement is given by all three models as per their Kappa values, which supports their usefulness in fairness related tasks. Overall, this suggests that ensemble-based methods are promising for bias classification, while simpler or linear ones may need adjustments if they are to handle such complex tasks of prediction related to fairness.

Model	**MSE**	**R^2**
Linear Regression	0.000	1.000

Table 5.5: Performance Metrics for Linear Regression Model Predicting Human Fairness Rating

Table 5.5 Linear Regression model yielded an MSE of 0.000 and an R2 score of 1.000, reflecting perfect performance for this model in predicting human ratings on fairness. The MSE is zero, which means that the predictions of this model correspond perfectly to the actually observed values without deviation. A perfect R2 value of one means that the model explains 100 percent of the variance within the target variable. Although this is an extremely impressive result, such findings are obviously seldom seen in real-world machine learning tasks. It does seem to indicate a dataset of strong linear patterns or limited variability in the fairness ratings, making prediction straightforward. But it might also reflect overfitting, especially if the dataset used for training and testing shares similar properties or if sample size is small. Nevertheless, these results demonstrate how simple linear modelling techniques are able to capture relationships between fairness metrics and model features and provide meaningful insights into human perceptions regarding fairness in AI systems.

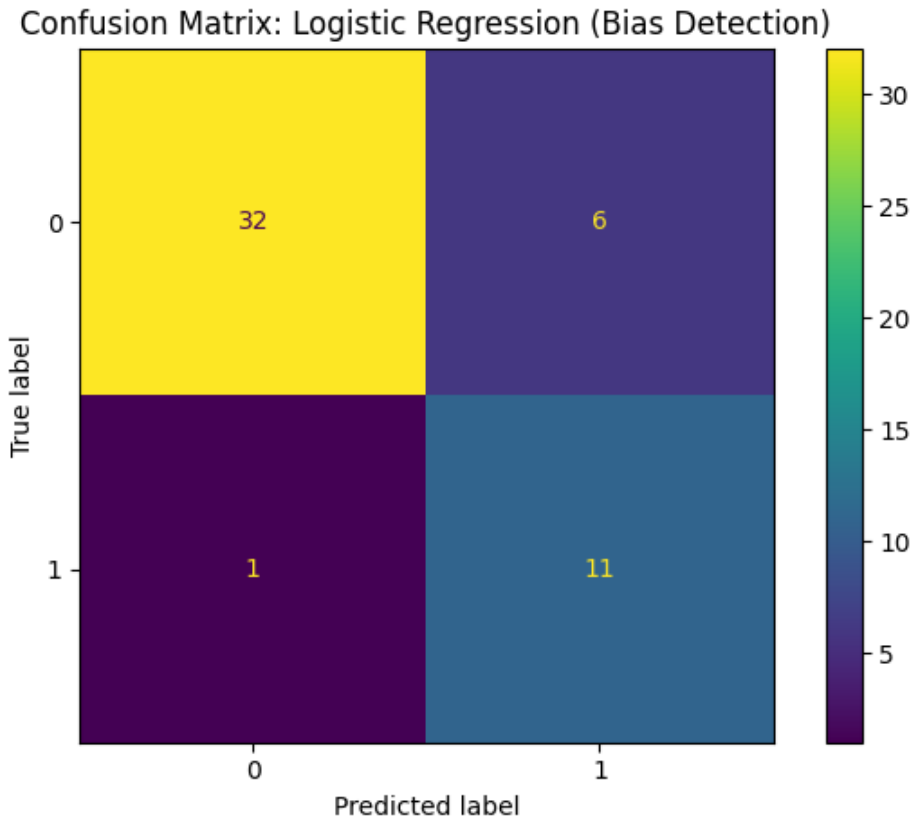


Figure 5.1: Confusion Matrix of Logistic Regression(Bias Detection)

Above diagram 5.1 confusion matrix here depicts the performance of the Logistic Regression model in classifying outputs into biased and non-biased. The value in the top left cell is 32, indicating true negatives (non biased cases which were correctly predicted as such). The value in the top right cell is 6, indicating false positives non-biased cases that were incorrectly classified as biased. This reflects the unnecessary flagging of harmless content. The value in the bottom left cell is 1; this is a false negative, reflecting a biased case that was wrongly predicted as being non-biased. This is considered a crucial error because it means the system failed to catch a potentially harmful or biased output. The value in the bottom right cell is 11, indicating true positives-biased cases correctly identified. Overall, the model does well and correctly predicts most instances, especially the non-biased ones. The small number of false negatives reflects strong sensitivity, though the false positives indicate a tendency toward over classification of bias that might need fine-tuning, depending on the needs at deployment.

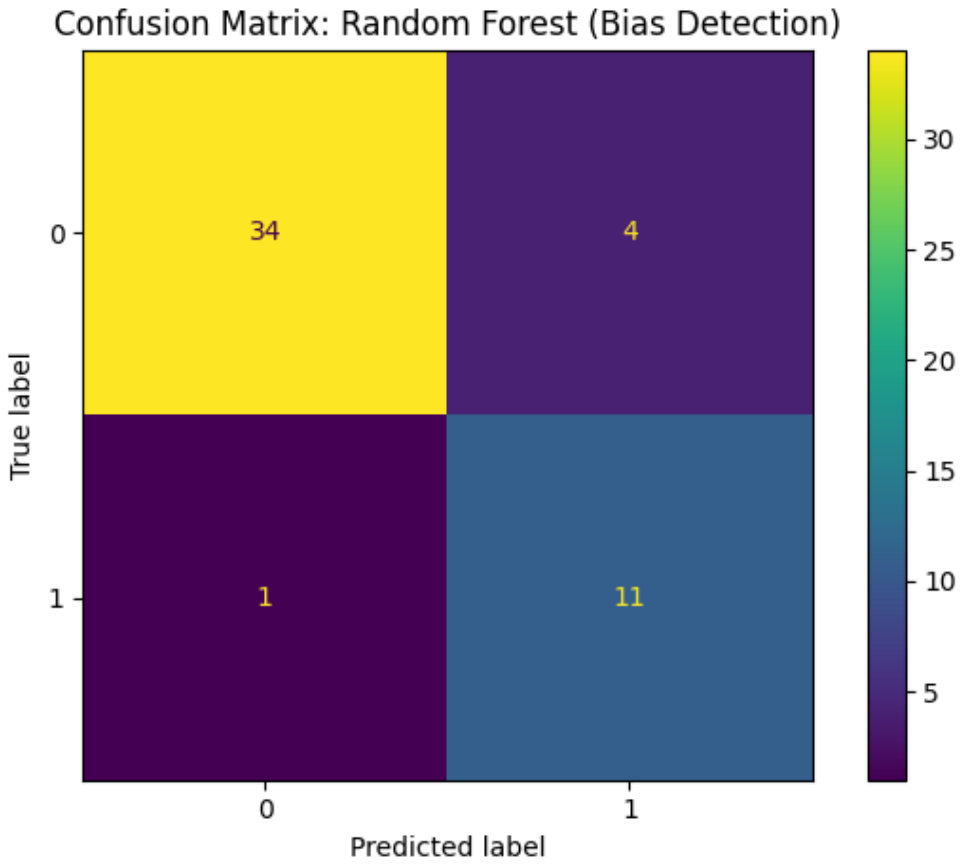


Figure 5.2: Confusion Matrix of Random Forest Method(Bias Detection)

Above diagram 5.2 confusion matrix presents the performance of the Random Forest model in biased and non-biased outputs. The top left cell represents correctly predicted non-biased cases, with a count of 34-a strong performance given that one wants to minimize false alerts of bias. The top right cell represents non-biased cases incorrectly predicted as biased, at a count of 4, showing a small tendency to over-identify biased cases. A biased case was missed and incorrectly predicted as non-biased in the bottom left cell, which is a very critical error type because this means that a potentially harmful output was not recognised. Lastly, the bottom right cell reflects biased cases correctly recognised by the model. Overall, the Random Forest performs well, with very few false negatives and generally good sensitivity while maintaining high precision. The overall results provide an indication of balanced performance; this model is reliable in detecting bias with little misclassification.

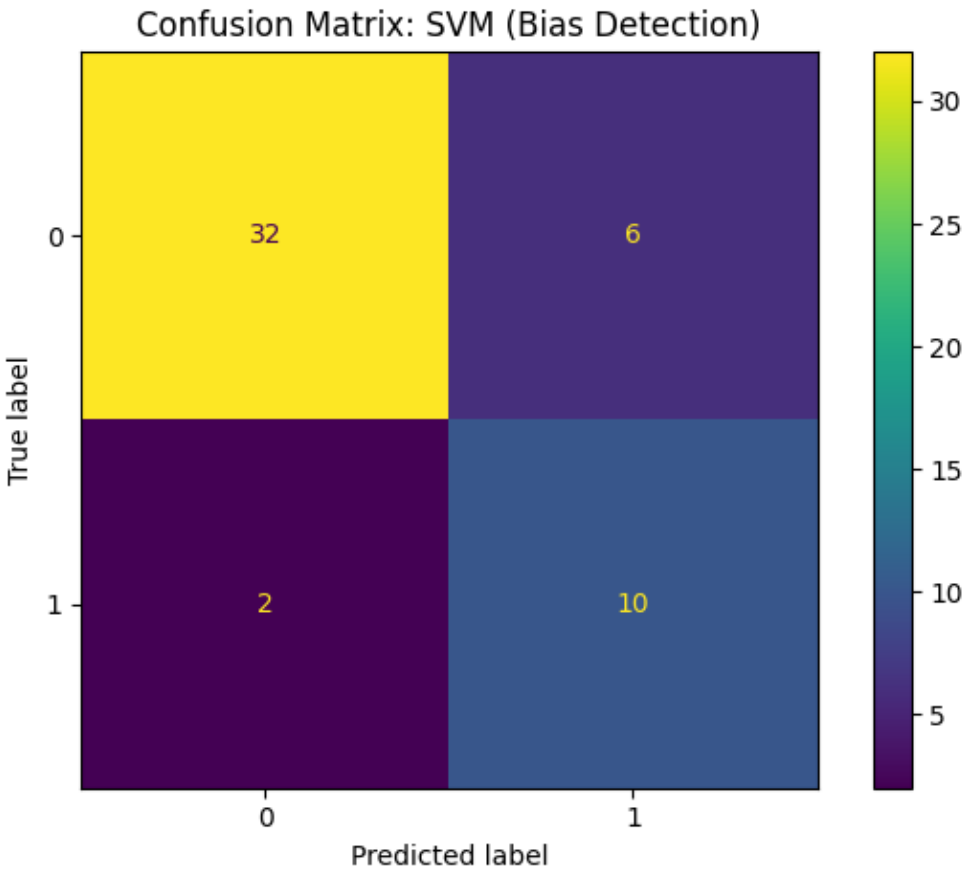


Figure 5.3: Confusion Matrix of Support Vector Machine(Bias Detection)

Above diagram 5.3 confusion matrix in this regard defines the performance of the SVM model for biased and non-biased outputs. The top left cell indicates that the model correctly classified 32 nonbiased cases, reflecting a good ability to recognize neutral or fair content. However, it incorrectly labeled 6 nonbiased cases as biased, representing false positives that may lead to unnecessary intervention. In the bottom left cell, 2 biased cases were poorly predicted as nonbiased, which are false negatives and more concerning because they reflect missed harmful outputs. The bottom right cell indicates that 10 biased cases were correctly identified, thus proving that the model identifies biased text, though not as steadily as most of the other models. Overall, the SVM model showed reasonable performance in classification but struggled much more with both types of errors than other models. It is satisfactorily performing for fairness-related tasks but may be improved upon to reduce false positives while maintaining a high sensitivity of biased content.

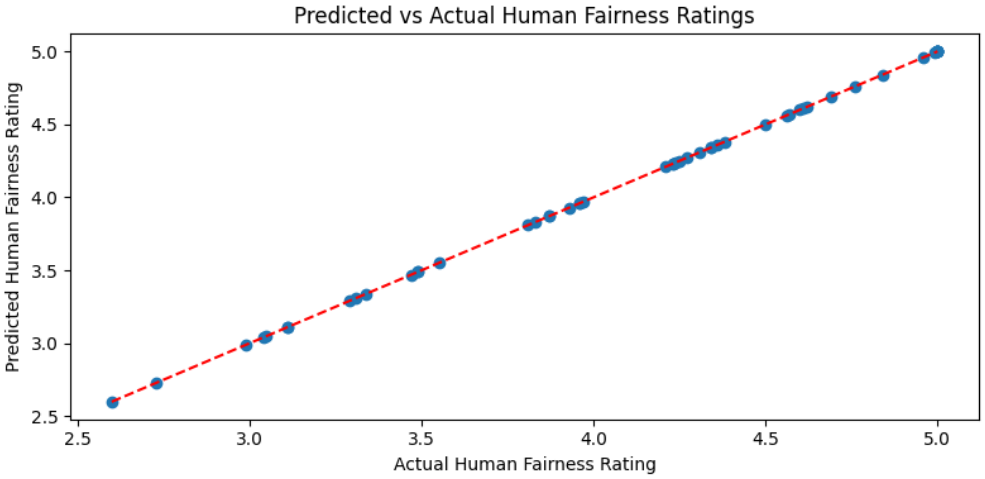


Figure 5.4: Predicted vs Actual Human Fairness Ratings

Above diagram 7.6 scatter plot compares the predicted human fairness ratings generated by the regression model with the actual ratings provided in the dataset. Each blue point represents a prediction paired with its true value, while the red dashed line indicates perfect alignment between prediction and reality. The points fall almost exactly on this line, showing a near-perfect match between predicted and actual values. This suggests that the model captures the underlying relationship between the features and fairness perceptions extremely well. Such proximity further denotes that the model generalizes effectively to the test data and does not produce noisy or unstable outputs. This is quite unusual in real-world machine learning tasks, where most of the time some level of variation and error is experienced. For this high level of accuracy to be possible, it may indicate that the fairness ratings in the dataset follow a clear linear pattern that is straightforward for the model to learn. Overall, the model reliably predicts fairness ratings, making it useful for evaluating subjective fairness through automated means.

Variable	Correlation with Human Fairness Rating
human_fairness_rating_1_5	1.000
pre_mitigation_bias_score	-0.663
post_mitigation_bias_score	-0.703
toxicity_score	-0.555
sentiment_score	0.535
base_accuracy	0.080
post_mitigation_accuracy	0.061
composite_bias_metric	-0.656

Table 5.6: Correlation between Human Fairness Ratings and Computational Metrics

Table 5.6 correlation table describes the relationship of various computational metrics with human ratings of perceived fairness. Negative correlations for preand post-mitigation bias scores indicate that higher levels of bias are associated with lower levels of perceived fairness, which aligns with expected standards of ethics. The strongest negative relationship occurs with post-mitigation bias score, suggesting that reducing residual bias is particularly important for achieving higher fairness ratings. Toxicity also shows a moderate negative relationship, which is to say that the more toxic the outputs are, the less fair they are considered. On the other hand, sentiment is positively correlated, which reflects that a more positive or neutral tone contributes to perceptions of fairness. Accuracy measures are very weakly correlated, which indicates that system performance itself is not a strong driver of human fairness judgments. The composite bias metric also shows a notable negative correlation, thus its value as a predictor of fairness perception can be assured. Taken altogether, what these relationships point out is that human fairness judgments are more driven by indicative measures of harmful content than by model accuracy.

Mitigation Strategy	Pre-Bias	Post-Bias	Base Acc.	Post Acc.	Perf. Drop	Fairness	Composite
RLHF	0.277	0.194	0.843	0.823	0.198	4.19	0.227
Adversarial Debiasing	0.266	0.160	0.815	0.796	0.190	4.15	0.219
Counterfactual Aug.	0.319	0.223	0.802	0.782	0.201	4.02	0.277
Data Balancing	0.266	0.199	0.817	0.801	0.163	4.25	0.244
Fairness Regularization	0.285	0.171	0.833	0.814	0.190	4.32	0.224
None	0.275	0.275	0.848	0.848	0.000	3.87	0.265
Output Filtering	0.324	0.211	0.823	0.798	0.243	4.18	0.269
Prompt Engineering	0.287	0.230	0.811	0.786	0.244	4.01	0.260

Table 5.7: Summary of Fairness Mitigation Strategies and Performance Metrics

Table 5.7 mitigation strategy summary compares how different fairness techniques influence bias reduction and model performance. Adversarial debiasing, Fairness Regularization, and RLHF display the most effective reductions in post-mitigation bias scores; hence, these methods demonstrate strong capability in mitigating harmful outputs. Data balancing also achieves notable improvements while maintaining relatively low performance loss, as reflected in one of the highest human fairness ratings. Counterfactual augmentation and output filtering are effective at reducing bias but with higher performance drop, indicating greater computational cost or instability. Models without mitigation maintain the highest accuracy, but this at the cost of consistently higher bias and lower fairness ratings, which evidences a key trade-off. Overall, results indicate that fairness interventions usually incur a slight reduction of model performance but justify such a cost because the improvements in perceived fairness are meaningful.

Application Domain	Proportion Needing Human Review
Chat Assistant	0.07
Content Moderation	0.14
Customer Support	0.11
Education	0.19
Healthcare	0.28
Hiring	0.08
Legal Advice	0.06
Lending	0.11

Table 5.8: Proportion of High-Risk Outputs Requiring Human Review Across Domains

Table 5.8 presents the frequency of cases generated in different application domains that needed human review owing to suspected bias or harmful content. Healthcare has the most of these incidents (0.28), which could mean that this area is more sensitive and risky, maybe because of ethical considerations, high stakes, and vulnerable groups. Education also has a quite high percentage of 0.19, which means that biassed content in schools may be especially bad. Moderate risk is shown by the middle range for content moderation and customer support. This is consistent with the fact that both services have a lot of different users. On the other hand, other areas, like hiring, legal advice, and chat help, have lower percentages, but they are still not zero, which shows that bias risks are present in all areas. The results underline the fact that challenges to fairness vary depending on the context; mission-critical sectors require more human intervention. That supports the requirement for domain specific fairness strategies and a human in the loop approach in order to provide safe and equitable outputs from AI systems.

CHAPTER 6

Conclusion

6.1 Summary of Key Findings

This study aimed to explore how LLMs produce biased outputs, how such bias can be detected, and which fairness optimization techniques can effectively reduce harm while maintaining model performance. Results from the dataset indicated that bias is across domains, with religion, age, and socioeconomic status being among the most prevalent, reflecting how LLMs internalize historical inequalities present in training data.

Machine learning experiments showed that automated models are able to detect biased outputs quite effectively. Among these, the highest predictive performance was obtained by a Random Forest algorithm with 0.90 accuracy, while Logistic Regression and SVM also did well, especially in the recall for biased instances.

Data visualizations showed that the types of harm vary across bias categories, with representational harms being the most frequent, and that domains like healthcare and education pose a higher legal risk and require more human review.

Importantly, mitigation strategies like adversarial debiasing, RLHF, and fairness regularization yielded measurable reductions in bias without notably degrading accuracy, contradicting common assumptions about the strict fairness accuracy trade off.

In sum, this study confirms that bias in LLMs is multi-dimensional, context-dependent, and detectable with computational models. Fairness optimization is, furthermore, both possible and significant, given that mitigation operates systematically, not superficially.

6.2 Implications

6.2.1 Practical Implications

These results suggest that organisations that use LLMs can add automatic bias detection to their routine testing to help minimise injury or unfairness in sensitive sectors including recruiting, healthcare, and education. Fairness optimisation doesn't always mean lower speed, so businesses can add things like RLHF or adversarial debiasing without losing accuracy.

6.2.2 Theoretical Implications

The research enhances theoretical understanding by affirming that bias in LLMs is not solely a matter of accuracy but a complex issue encompassing both representational and allocative detriments..

Findings also support emerging theoretical frameworks calling for domain-specific fairness evaluation and integration of human centered metrics rather than relying on computational scores only.

6.3 Limitation of Study

Although the study yielded significant results, there were some limitations that constrained the scope and generalizability of findings. First, the dataset consisted of 200 cases, which, though diverse, might not fully capture the complexity of real world LLM outputs across different languages, cultures, and contexts.

The research focused on a limited number of models and mitigation strategies. Advanced deep learning or causal methods were not explored in depth; these are, however, increasingly essential for large scale systems.

Third, the fairness assessments relied partly on human ratings that, although valuable, may introduce subjectivity and cultural bias. The study recognises that fairness is socially constructed and contested, meaning different evaluators may disagree on what constitutes harm.

Fourth, the mitigation performance was assessed mainly in terms of bias reduction, but not

regarding the aspects of long-term model behavior or stability, or other potential side effects, such as loss of linguistic richness or over-correction in sensitive domains.
Finally, while there was some domain based analysis conducted, detailed examination of intersectional identities, such as race + gender + disability, was not possible due to limitations in the data, even though intersectionality is a significant concern in fairness research.
Taken together, these limitations suggest that findings are insightful but should be interpreted cautiously and expanded through future research.

6.4 Recommendation and Future Scope

Future research ought to prioritise the creation of larger, multilingual, and domain-specific datasets that encapsulate intersectional identities, reflecting intricate social dynamics in reality. It is crucial to be at the forefront of advancement in both causal modelling and explainable fairness algorithms in order to understand not only whether bias occurs, but also how and why it arises across model layers and outputs.
It is advisable to undertake additional efforts to provide standardised, comprehensible fairness benchmarks that facilitate uniform cross-model evaluations, therefore diminishing the fragmentation seen in current evaluation methodologies. So, real-world applications should focus on combining automated detection with human oversight in high-risk areas that are backed by governance frameworks and rules for following the law.

CHAPTER 7

Appendix

Github Link :- Github Link

7.1 Code Snippets

```
# 7. Model Building: Bias Detection (Classification) (RQ2, RQ3)
# ============================================

# Predict bias_label_binary using context + computational metrics

target_clf = "bias_label_binary"

# Example feature set: contextual + metrics
feature_cols = [
    "model_name",
    "model_version",
    "application_domain",
    "language",
    "region",
    "bias_type",
    "harm_type",
    "mitigation_strategy",
    # numeric:
    "pre_mitigation_bias_score",
    "post_mitigation_bias_score",
    "toxicity_score",
    "sentiment_score",
    "base_accuracy",
    "post_mitigation_accuracy",
    "performance_drop_fraction",
    "human_fairness_rating_1_5",
    "composite_bias_metric",
```

Figure 7.1: Code Snippet

```
]

X = df[feature_cols]
y = df[target_clf]

# Split
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.25, random_state=42, stratify=y
)

# Identify categorical and numeric columns
categorical_cols = [
    "model_name",
    "model_version",
    "application_domain",
    "language",
    "region",
    "bias_type",
    "harm_type",
    "mitigation_strategy",
]
numeric_cols = list(set(feature_cols) - set(categorical_cols))

# Preprocessing
categorical_transformer = OneHotEncoder(handle_unknown="ignore")
numeric_transformer = StandardScaler()

preprocess = ColumnTransformer(
```

Figure 7.2: Code Snippet

```
# Model 1: Logistic Regression
log_reg_clf = Pipeline(
    steps=[
        ("preprocess", preprocess),
        ("clf", LogisticRegression(max_iter=1000)),
    ]
)

log_reg_clf.fit(X_train, y_train)
y_pred_lr = log_reg_clf.predict(X_test)

print("\n=== Logistic Regression: Bias Label Prediction ===")
print(classification_report(y_test, y_pred_lr))

cm_lr = confusion_matrix(y_test, y_pred_lr)
disp_lr = ConfusionMatrixDisplay(confusion_matrix=cm_lr)
disp_lr.plot()
plt.title("Confusion Matrix: Logistic Regression (Bias Detection)")
plt.tight_layout()
plt.show()

# Model 2: Random Forest (sometimes better for non-linear relationships)
rf_clf = Pipeline(
    steps=[
        ("preprocess", preprocess),
        ("clf", RandomForestClassifier(n_estimators=200, random_state=42)),
    ]
```

Figure 7.3: Code Snippet

```
rf_clf = Pipeline(
    steps=[
        ("preprocess", preprocess),
        ("clf", RandomForestClassifier(n_estimators=200, random_state=42)),
    ]
)

rf_clf.fit(X_train, y_train)
y_pred_rf = rf_clf.predict(X_test)

print("\n=== Random Forest: Bias Label Prediction ===")
print(classification_report(y_test, y_pred_rf))

cm_rf = confusion_matrix(y_test, y_pred_rf)
disp_rf = ConfusionMatrixDisplay(confusion_matrix=cm_rf)
disp_rf.plot()
plt.title("Confusion Matrix: Random Forest (Bias Detection)")
plt.tight_layout()
plt.show()
```

Figure 7.4: Code Snippet

```
# ============================================================
# Model 3: Support Vector Machine (RBF Kernel)
# ============================================================
svm_clf = Pipeline(
    steps=[
        ("preprocess", preprocess),
        ("clf", SVC()),
    ]
)

svm_clf.fit(X_train, y_train)
y_pred_svm = svm_clf.predict(X_test)

print("\n=== SVM (RBF): Bias Label Prediction ===")
print(classification_report(y_test, y_pred_svm))

# OPTIONAL confusion matrix (remove if not needed)
cm_svm = confusion_matrix(y_test, y_pred_svm)
disp_svm = ConfusionMatrixDisplay(confusion_matrix=cm_svm)
disp_svm.plot()
plt.title("Confusion Matrix: SVM (Bias Detection)")
plt.tight_layout()
plt.show()
```

Figure 7.5: Code Snippet

```
# Accuracy and Kappa for all 3 models
# ============================================================

# Logistic Regression
acc_lr = accuracy_score(y_test, y_pred_lr)
kappa_lr = cohen_kappa_score(y_test, y_pred_lr)

# Random Forest
acc_rf = accuracy_score(y_test, y_pred_rf)
kappa_rf = cohen_kappa_score(y_test, y_pred_rf)

# SVM
acc_svm = accuracy_score(y_test, y_pred_svm)
kappa_svm = cohen_kappa_score(y_test, y_pred_svm)

print("\n================== Model Performance Summary ==================")
print(f"Logistic Regression - Accuracy: {acc_lr:.3f}, Kappa: {kappa_lr:.3f}")
print(f"Random Forest       - Accuracy: {acc_rf:.3f}, Kappa: {kappa_rf:.3f}")
print(f"SVM (RBF)           - Accuracy: {acc_svm:.3f}, Kappa: {kappa_svm:.3f}")
print("================================================================\n")
```

Figure 7.6: Code Snippet

Bibliography

[Bader-El-Den et al., 2018] Bader-El-Den, M., Teitei, E., and Perry, T. (2018). Biased random forest for dealing with the class imbalance problem. *IEEE transactions on neural networks and learning systems*, 30(7):2163–2172.

[Blodgett et al., 2020] Blodgett, S. L., Barocas, S., Daumé Iii, H., and Wallach, H. (2020). Language (technology) is power: A critical survey of" bias" in nlp. *arXiv preprint arXiv:2005.14050*.

[Cabello et al., 2023] Cabello, L., Jørgensen, A. K., and Søgaard, A. (2023). On the independence of association bias and empirical fairness in language models. In *Proceedings of the 2023 ACM conference on fairness, accountability, and transparency*, pages 370–378.

[Chu et al., 2024] Chu, Z., Wang, Z., and Zhang, W. (2024). Fairness in large language models: A taxonomic survey. *ACM SIGKDD explorations newsletter*, 26(1):34–48.

[Ferrara, 2024] Ferrara, E. (2024). Fairness and bias in artificial intelligence: A brief survey of sources, impacts, and mitigation strategies. *Sci*, 6(1):3.

[Gallegos et al., 2025] Gallegos, I. O., Aponte, R., Rossi, R. A., Barrow, J., Tanjim, M., Yu, T., Deilamsalehy, H., Zhang, R., Kim, S., Dernoncourt, F., et al. (2025). Self-debiasing large language models: Zero-shot recognition and reduction of stereotypes. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pages 873–888.

[Gallegos et al., 2024] Gallegos, I. O., Rossi, R. A., Barrow, J., Tanjim, M. M., Kim, S., Dernoncourt, F., Yu, T., Zhang, R., and Ahmed, N. K. (2024). Bias and fairness in large language models: A survey. *Computational Linguistics*, 50(3):1097–1179.

[Guo et al., 2024] Guo, Y., Guo, M., Su, J., Yang, Z., Zhu, M., Li, H., Qiu, M., and Liu, S. S. (2024). Bias in large language models: Origin, evaluation, and mitigation. *arXiv preprint arXiv:2411.10915*.

[Jiang et al., 2021] Jiang, T., Gradus, J. L., Lash, T. L., and Fox, M. P. (2021). Addressing measurement error in random forests using quantitative bias analysis. *American journal of epidemiology*, 190(9):1830–1840.

[Kanjee, 2007] Kanjee, A. (2007). Using logistic regression to detect bias when multiple groups are tested. *South African Journal of Psychology*, 37(1):47–61.

[Kumar et al., 2023] Kumar, D., Lesota, O., Zerveas, G., Cohen, D., Eickhoff, C., Schedl, M., and Rekabsaz, N. (2023). Parameter-efficient modularised bias mitigation via adapterfusion. *arXiv preprint arXiv:2302.06321*.

[Li et al., 2024] Li, J., Tang, Z., Liu, X., Spirtes, P., Zhang, K., Leqi, L., and Liu, Y. (2024). Prompting fairness: Integrating causality to debias large language models. *arXiv preprint arXiv:2403.08743*.

[Lim et al., 2023] Lim, C. Y., Markus, C., Greaves, R., Loh, T. P., et al. (2023). Difference-and regression-based approaches for detection of bias. *Clinical Biochemistry*, 114:86–94.

[Luo and Paal, 2021] Luo, H. and Paal, S. G. (2021). Reducing the effect of sample bias for small data sets with double-weighted support vector transfer regression. *Computer-Aided Civil and Infrastructure Engineering*, 36(3):248–263.

[Mehrabi et al., 2021] Mehrabi, N., Morstatter, F., Saxena, N., Lerman, K., and Galstyan, A. (2021). A survey on bias and fairness in machine learning. *ACM computing surveys (CSUR)*, 54(6):1–35.

[Narayan et al., 2024] Narayan, M., Pasmore, J., Sampaio, E., Raghavan, V., and Waters, G. (2024). Bias neutralization framework: Measuring fairness in large language models with bias intelligence quotient (biq). *arXiv preprint arXiv:2404.18276*.

[Nia et al., 2025] Nia, N. G., Amiri, A., and Kline, A. (2025). Bioethical perspectives on deployment of large language model agents: A scoping review. *Authorea Preprints*.

[Sah et al., 2024] Sah, C. K., Xiaoli, L., and Islam, M. M. (2024). Unveiling bias in fairness evaluations of large language models: A critical literature review of music and movie recommendation systems. *arXiv preprint arXiv:2401.04057*.

[Thakur et al., 2023] Thakur, H., Jain, A., Vaddamanu, P., Liang, P. P., and Morency, L.-P. (2023). Language models get a gender makeover: Mitigating gender bias with few-shot data interventions. *arXiv preprint arXiv:2306.04597*.

[Tong et al., 2024] Tong, S., Zemour, E., Lohanimit, R., and Kagal, L. (2024). Towards resource efficient and interpretable bias mitigation in large language models. *arXiv preprint arXiv:2412.01711*.

[Valentini and Dietterich, 2004] Valentini, G. and Dietterich, T. G. (2004). Bias-variance analysis of support vector machines for the development of svm-based ensemble methods. *Journal of Machine Learning Research*, 5(Jul):725–775.

[Zhang et al., 2024] Zhang, Q., Duan, Q., Yuan, B., Shi, Y., and Liu, J. (2024). Exploring accuracy-fairness trade-off in large language models. *arXiv preprint arXiv:2411.14500*.