**Project Title:** **Identifying Tools for Bias Mitigation in AI Training Data**

**NAME:** Rohan kini M  
**CAN\_ID:** CAN\_33763153  
**MY CONTRIBUTION:** visualization of plots

**Assignment****:Results and visualization**

### **1 1. Results and Visualization**

Bias in AI training data can be observed through various performance metrics and fairness evaluations. This section presents key metrics and visualizations to assess bias and the effectiveness of mitigation techniques.

* 1. **Performance Metrics**

| **Model** | **Accuracy** | **Demographic Parity** | **Equalized Odds** | **Disparate Impact** |
| --- | --- | --- | --- | --- |
| Baseline Model | 85% | 0.65 | 0.70 | 0.55 |
| Bias-mitigated Model | 82% | 0.90 | 0.88 | 0.92 |

* **Key Insights**: Bias mitigation techniques improved fairness metrics with minimal accuracy trade-offs.

A screenshot of a graph

AI-generated content may be incorrect.

**1.2 Visualizations**

* **Confusion Matrix Analysis**: Highlights true positives, false positives, true negatives, and false negatives for different demographic groups.
* **Feature Distribution Comparisons**: Demonstrates disparities in feature distributions between groups.
* **ROC Curves per Demographic Group**: Evaluates classification performance across different subsets.

**Explanation of the Visualizations:**

1. **Statistical Parity Difference Before and After Mitigation (Left Chart)**
   * This bar chart compares the **Statistical Parity Difference (SPD)** before and after applying different bias mitigation techniques.
   * **Before mitigation** (red bars), the SPD is higher, indicating significant bias.
   * **After mitigation** (orange bars), the SPD is lower, showing that techniques like **Reweighing, Disparate Impact Remover, and Adversarial Debiasing** effectively reduced bias.
2. **Bias Score Trend Over Time (Right Chart)**
   * This line chart tracks the **Bias Score (SPD)** reduction over four weeks.
   * The decreasing trend indicates that continuous bias mitigation strategies lead to fairer AI models over time.

These visualizations demonstrate the effectiveness of bias mitigation techniques in improving fairness in AI models.

A graph and diagram of a graph

AI-generated content may be incorrect.

**ROC Curves per Demographic Group:**

The **Receiver Operating Characteristic (ROC) curve** is a graphical representation of a classification model's performance across different threshold values. It plots the **True Positive Rate (TPR)** against the **False Positive Rate (FPR)**, showing the trade-off between sensitivity and specificity. The **Area Under the Curve (AUC)** quantifies the model's ability to distinguish between classes, where a higher AUC indicates better discrimination. In this case, the **Gender Model AUC (0.7625) and Racist Model AUC (0.7765)** suggest moderate predictive performance.

## A graph of a model AI-generated content may be incorrect.

## ****4. Fairness Metrics and Evaluation****

**Key Insights:** Bias mitigation techniques improved fairness metrics with minimal accuracy trade-offs.

A blue and green squares

AI-generated content may be incorrect.

## ****5. Code****

## A screen shot of a computer AI-generated content may be incorrect.A computer screen shot of a program AI-generated content may be incorrect.A screen shot of a computer program AI-generated content may be incorrect.

## ****5. Analysis of IBM Cloud Resources****

## ****5.1 Resource Units Utilized****

## **The deployment and operation of the AutoAI model in Watson Studio involved the following**

## **resource usage:**

## **● **ComputeHours(CUH):** Approximately 14 CUH were utilized across various tasks,**

## **including:**

## **○ Preprocessing data using Jupyter Notebook.**

## **○ Modelcreation and training through AutoAI.**

## **○ Testing and deployment for evaluation.**

## **● **APIRequests:** Watson Machine Learning on the Lite plan supports up to 50**

## **deployment requests per month. These requests were used during deployment**

## **testing and user interactions.**

## ****● Storage:** 1 GB allocated for storing the dataset and trained model.**

## **This setup operates within the free tier limits of IBM services, which supports low to**

## **moderate workloads without incurring additional costs.**

## ****5.2 Model Performance Metrics****

## **● Average Response Time: ~200 ms for typical API calls under standard conditions.**

## **● PeakResponseTime: ~500 ms during periods of high traffic or resource contention.**

## **These response times indicate that the model is efficient in handling predictions, even during**

## **peak demand, making it suitable for tasks like fraud detection.**

## ****5.3 IBM CLOUD****

## **The free tier of IBM Cloud services offers limited resources, which may restrict scalability for**

## **larger-scale deployments:**

## **● ComputeHours: The free tier provides 20 CUH/month, leaving limited overhead for**

## **additional experimentation or scaling.**

## **● APIRequests: Limited to 50 deployment requests/month, suitable for small-scale**

## **testing but insufficient for high-traffic applications.**

## **● Storage: 1 GB storage is sufficient for small datasets and models but may require**

## **upgrades for larger or more complex projects.**

## **To scale this setup:**

## **● Transition to a paid tier to unlock higher CUH and request limits.**

## **● Optimize preprocessing and testing workflows to conserve compute resources.**

## **● Implement caching strategies to reduce API call frequency for repetitive tasks.**

## 

## **This analysis highlights the feasibility of deploying lightweight models within the free tier**

## **while underscoring the need for resource optimization or upgrades for expanded use cases.**

## **Steps to Upload a Project to GitHub**

## **1. Initialize Git Repository:**

## **bash**

## **git init**

## **git add .**

## **git commit-m "Initial commit"**

## **2. Create a Repository on GitHub:**

## **○ Loginto GitHub.**

## **○ Click "New Repository".**

## **○ Nametherepository and click Create.**

## **3. Push Code to GitHub:**

## **git remote add origin**

## **https://github.com/yourusername/yourrepository.git**

## **git branch-M main**

## **git push-u origin main**

## <https://github.com/Rohankinim/AI-DATA-QUALITY->

## **Note: Avoid committing sensitive information (e.g., API keys, passwords). Use a**

## **.gitignore file to exclude such files and manage secrets with environment variables**

## A screenshot of a computer AI-generated content may be incorrect.

## ****6. Future Directions****

* **Explainable AI (XAI)**: Enhancing transparency in decision-making.
* **Federated Learning**: Reducing bias by training on diverse decentralized datasets.
* **Continuous Monitoring**: Regular audits to detect emerging biases over time.

## ****7. Conclusion****

Identifying and mitigating bias in AI models is essential for building ethical, fair, and reliable AI systems. By applying rigorous fairness evaluation and mitigation strategies, AI models can make unbiased and equitable decisions, fostering trust in AI-driven applications.YOUTUBELINK <https://youtu.be/eTwvgREfesg>