Department of Computer Science and Engineering (Data Science)
Final Year. B.Tech. Sem: VIII Subject: Data Ethics
Experiment 10

Bhuvi Ghosh
60009210191

Date:	**Experiment Title: Disparate Impact Analysis**
Aim	To study Disparate Impact Analysis.
Software	Google Colab, Visual Studio Code, Jupyter Notebook
Theory To Be Written	Describe Equal opportunity, Equalized odds and Disparate Impact as the fairness metrics.
Implementation	Import pandas, numpy, confusion matrix from sklearn. Load the data. Write compute_confusion_matrix ( ) function, taking inputs as actual (y_true) and predicted (y_pred) values. It computes the confusion matrix and extracts True Negatives (TN), False Positives (FP), False Negatives (FN), and True Positives (TP). It returns the values as a dictionary. Write compute_metrics ( ) function, taking confusion matrix as the input. It returns Accuracy, True Positive Rate, False Positive Rate and Disparate Impact. Write fairness_metrics ( ) function , taking dataframe as the input. It should split data into Group A and Group B based on "Group (A/B)" column.It computes the confusion matrix for each group. It computes accuracy, TPR, FPR, and disparate impact for both groups.It prints all the above fairness-related metrics. Write threshold_adjustment ( ) function, taking dataframe and new threshold value as the input. It creates a copy of the dataset. It selects the **"Model's Hiring Decision (Y hat)"** column **only for Group B**, so we are modifying values in this column for Group B members. It extracts the "True Qualification (Y)" values **only for Group B** and Compares each "True Qualification (Y)" value to new_threshold. It returns **True** for values that **meet or exceed** the threshold. It returns **False** for values that **are below** the threshold. It calls fairness_metrics() again to evaluate fairness after the threshold adjustment. Main Execution Block It loads the dataset. It computes initial fairness metrics. It adjusts the decision threshold for Group B and recomputes fairness metrics.

Conclusion	Hence, we have studied **Disparate Impact Analysis**

THEORY :

Fairness Metrics in Machine Learning

Fairness in machine learning is crucial to ensuring that models do not unintentionally discriminate against certain groups. Various fairness metrics are used to evaluate whether a model's predictions are biased. Some key fairness metrics include:

1. Equal Opportunity

Equal Opportunity focuses on ensuring fairness for individuals who actually qualify for a favorable outcome. It states that the **True Positive Rate (TPR)** should be the same across all demographic groups. In other words, if two individuals from different groups have the same qualification, they should have an equal chance of being correctly classified as positive.

🔻 **Example:** In a hiring model, if two candidates (one from Group A and one from Group B) are both truly qualified, they should have the same likelihood of being selected by the model.

🔻 **Purpose:** Ensures fairness for qualified individuals and reduces bias against certain groups.

2. Equalized Odds

Equalized Odds extends Equal Opportunity by requiring that both the **True Positive Rate (TPR)** and the **False Positive Rate (FPR)** be equal across different groups. This means that the model should not only give qualified individuals an equal chance of being selected, but it should also make incorrect positive predictions (false positives) at the same rate for all groups.

🔻 **Example:** In a loan approval model, if a model wrongly grants loans to unqualified individuals at a higher rate for one group compared to another, it violates equalized odds.

🔻 **Purpose:** Ensures fairness in both positive and negative outcomes, preventing favoritism or discrimination.

3. Disparate Impact

Disparate Impact measures whether a model's decision disproportionately favors one group over another. It compares the **selection rate** (the proportion of individuals receiving a favorable outcome) across different groups. A common rule used is the **80% rule**, which states that the selection rate for any group should be at least 80% of the selection rate for the most favored group.

🔻 **Example:** If a hiring model selects 50% of applicants from Group A but only 30% from Group B, the disparate impact ratio is 30/50 = 0.6, which is below the acceptable threshold of 0.8, indicating potential bias.

🔻 **Purpose:** Detects indirect discrimination and helps ensure fair representation of different groups in outcomes.

```python
import pandas as pd
import numpy as np
from sklearn.metrics import confusion_matrix

def compute_confusion_matrix(y_true, y_pred):
    tn, fp, fn, tp = confusion_matrix(y_true, y_pred).ravel()
    return {"TN": tn, "FP": fp, "FN": fn, "TP": tp}

def compute_metrics(conf_matrix, tpr_a, tpr_b):
    tn, fp, fn, tp = conf_matrix["TN"], conf_matrix["FP"], conf_matrix["FN"], conf_matrix["TP"]
    accuracy = (tp + tn) / (tp + tn + fp + fn)
    tpr = tp / (tp + fn) if (tp + fn) > 0 else 0  # True Positive Rate (Sensitivity)
    fpr = fp / (fp + tn) if (fp + tn) > 0 else 0  # False Positive Rate
    disparate_impact = tpr_b / tpr_a if tpr_a > 0 else 0  # Ratio of underprivileged to privileged group TPR
    return {"Accuracy": accuracy, "TPR": tpr, "FPR": fpr, "Disparate Impact": disparate_impact}

def fairness_metrics(df):
    group_a = df[df["Group (A/B)"] == "A"]
    group_b = df[df["Group (A/B)"] == "B"]

    conf_matrix_a = compute_confusion_matrix(group_a["True Qualification (Y)"], group_a["Model's Hiring Decision (Y hat)"])
    conf_matrix_b = compute_confusion_matrix(group_b["True Qualification (Y)"], group_b["Model's Hiring Decision (Y hat)"])

    tpr_a = conf_matrix_a["TP"] / (conf_matrix_a["TP"] + conf_matrix_a["FN"])
    tpr_b = conf_matrix_b["TP"] / (conf_matrix_b["TP"] + conf_matrix_b["FN"])

    metrics_a = compute_metrics(conf_matrix_a, tpr_a, tpr_b)
    metrics_b = compute_metrics(conf_matrix_b, tpr_a, tpr_b)
```

```python
    print("Confusion Matrix for Group A:", conf_matrix_a)
    print("Fairness Metrics for Group A:", metrics_a)
    print("Confusion Matrix for Group B:", conf_matrix_b)
    print("Fairness Metrics for Group B:", metrics_b)

def threshold_adjustment(df, new_threshold):
    df_adjusted = df.copy()
    group_b = df_adjusted[df_adjusted["Group (A/B)"] == "B"].copy()

    group_b["Model's Hiring Decision (Y hat)"] = (group_b["True Qualification (Y)"] >= new_threshold).astype(int)
    df_adjusted.update(group_b)

    print("Fairness Metrics After Threshold Adjustment:")
    fairness_metrics(df_adjusted)
    return df_adjusted

# Main Execution Block
if __name__ == "__main__":
    # Load dataset from Excel file
    df = pd.read_excel("/content/dataset.xlsx")

    print("Initial Fairness Metrics:")
    fairness_metrics(df)

    # Adjusting threshold for Group B
    new_threshold = 0.5
    df_adjusted = threshold_adjustment(df, new_threshold)
```

```
Initial Fairness Metrics:
Confusion Matrix for Group A: {'TN': 2, 'FP': 0, 'FN': 1, 'TP': 2}
Fairness Metrics for Group A: {'Accuracy': 0.8, 'TPR': 0.6666666666666666, 'FPR': 0.0, 'Disparate Impact': 0.5}
Confusion Matrix for Group B: {'TN': 2, 'FP': 0, 'FN': 2, 'TP': 1}
Fairness Metrics for Group B: {'Accuracy': 0.6, 'TPR': 0.3333333333333333, 'FPR': 0.0, 'Disparate Impact': 0.5}
Fairness Metrics After Threshold Adjustment:
Confusion Matrix for Group A: {'TN': 2, 'FP': 0, 'FN': 1, 'TP': 2}
Fairness Metrics for Group A: {'Accuracy': 0.8, 'TPR': 0.6666666666666666, 'FPR': 0.0, 'Disparate Impact': 1.5}
Confusion Matrix for Group B: {'TN': 2, 'FP': 0, 'FN': 0, 'TP': 3}
Fairness Metrics for Group B: {'Accuracy': 1.0, 'TPR': 1.0, 'FPR': 0.0, 'Disparate Impact': 1.5}
```

CONCLUSION:

Fairness metrics help identify and mitigate bias in machine learning models. While Equal Opportunity ensures fairness for qualified individuals, Equalized Odds considers both correct and incorrect predictions, and Disparate Impact focuses on overall selection rates. Using these metrics together allows for a more comprehensive fairness evaluation in decision-making systems.

Signature of Faculty