Final PHASE 3 PROJECT Submission

- Student name: MUTISYA FAITH MWENDE
- Student pace: part-time

Analyzing Predictive Factors and Bias in Arrest Outcomes During Terry Stops: A Classification Approach

Background

Terry v. Ohio, a landmark U.S. Supreme Court case decided in 1968, established the principle of **"reasonable suspicion,"** which allows police officers to temporarily detain individuals based on suspicious behavior, even in the absence of probable cause required for arrest. These stops, known as **Terry Stops**, have become a standard police practice, particularly in situations involving suspicious drivers. While Terry Stops are intended to prevent crime and ensure public safety, they have also raised significant concerns about potential bias in law enforcement, particularly regarding **race and gender**.

In recent years, the role of race and other demographic factors in policing has come under intense scrutiny. Research has suggested that certain demographic groups may be disproportionately affected by police practices, leading to questions about the fairness and objectivity of these interactions. This project aims to explore these issues by building a machine learning classifier to predict whether an arrest was made during a Terry Stop are affected by presence of weapons, time of day, gender, and race.

Problem Statement

While Terry Stops are legally justified under the principle of "reasonable suspicion," there is ongoing debate about whether these stops are conducted in a manner that is free from bias. The core problem this project seeks to address is whether certain factors, particularly race and gender, disproportionately influence the likelihood of arrest during a Terry Stop. Understanding these dynamics is crucial for law enforcement agencies, policymakers, and the public as they work to ensure that policing practices are fair, transparent, and just.

This project will focus on developing a classification model to predict arrest outcomes during Terry Stops. The model will analyze various factors, including demographic information, to determine which variables most strongly influence the likelihood of an arrest. By doing so, the project aims to shed light on potential biases in policing practices and offer insights that could guide more equitable law enforcement strategies.

Objectives

1. **Data Exploration and Understanding**:

- This will involve analyzing the distribution of key variables, particularly those related to demographics such as race and gender, as well as other critical factors like the presence of weapons and the time of day. Additionally, any data quality issues, such as missing values or outliers, will be identified and appropriately addressed to ensure the reliability of the analysis and subsequent modeling.

2. **Model Development**:
   - Build a baseline classification model using logistic regression to predict whether an arrest was made during a Terry Stop.
   - Develop a decision tree model as an alternative approach, comparing its performance with the logistic regression model.
3. **Feature Importance Analysis**:
   - This will involve identifying the most influential features in predicting arrest outcomes, with particular attention given to demographic factors such as race and gender. Sensitivity analyses will be performed to assess how changes in these key features impact the model's predictions, providing deeper insights into the factors driving arrest decisions during Terry Stops.
4. **Evaluation of Model Performance**:
   - This will be done using appropriate classification metrics, including accuracy, precision, recall, and the F1 score. By comparing the performance of the logistic regression and decision tree models, the project will determine the most effective approach for predicting arrest outcomes in the context of Terry Stops.
5. **Ethical Considerations and Recommendations**:
   - Discuss the ethical implications of using demographic data in predictive policing models, with a focus on the potential for bias and discrimination.
   - Provide recommendations for law enforcement agencies based on the findings, aimed at promoting fairer and more transparent policing practices.

Data Exploration and understanding

```
# Importing dictonaries
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np
```

Loading the dataset

```
# Load the dataset
df = pd.read_csv('Terry_Stops_20240831.csv')

# Displaying the first few rows of the dataset
df.head()
```

```
  Subject Age Group   Subject ID     GO / SC Num  Terry Stop ID  \
0          46 - 55           -1  20170000170940         265604
```

```
1           46 - 55  31037044483  20220000095762    32872287472
2           18 - 25           -1  20170000000631         239079
3           18 - 25   8754606833  20190000205676     8301110425
4           36 - 45           -1  20170000002861         283726

  Stop Resolution                Weapon Type Officer ID  Officer
YOB  \
0  Offense Report                       None       5167         1967

1   Field Contact                          -       6805         1973

2   Field Contact  Lethal Cutting Instrument       7580         1982

3  Offense Report                          -       6115         1968

4   Field Contact                       None       6090         1961


  Officer Gender        Officer Race  ...     Reported Time  \
0              M               White  ...  16:28:00.0000000
1              M               White  ...  12:10:26.0000000
2              M  Hispanic or Latino  ...  07:24:00.0000000
3              M               White  ...  16:55:35.0000000
4              M               White  ...  23:03:00.0000000

                            Initial Call Type  \
0  SUSPICIOUS STOP - OFFICER INITIATED ONVIEW
1                       BURG - COMM BURGLARY
2                                          -
3                                          -
4                                          -

                            Final Call Type Call Type  \
0  --SUSPICIOUS CIRCUM. - SUSPICIOUS PERSON    ONVIEW
1   --BURGLARY - NON RESIDENTIAL/COMMERCIAL       911
2                                        -         -
3                                        -         -
4                                        -         -

                              Officer Squad Arrest Flag Frisk Flag
Precinct  \
0                        WEST PCT OPS - CPT           N           N
West
1             WEST PCT 1ST W - KQ/DM RELIEF           N           N
-
2                    NORTH PCT OPS - ACT DAY           N           Y
North
3  NORTH PCT 2ND W - BOY (JOHN) - PLATOON 1           N           N
North
4       SOUTH PCT 3RD W - OCEAN - PLATOON 2           N           N
```

```
-

  Sector Beat
0      Q   Q1
1      -    -
2      L   L2
3      B   B2
4      -    -

[5 rows x 23 columns]
```

The table displays the first five rows of the dataset. Each row corresponds to a Terry stop, and the columns provide various attributes such as the subject's age group, officer details, stop resolution, and various flags indicating the outcome of the stop. Key columns include:

- Subject Age Group: The age range of the subject involved in the stop.
- Officer ID and Officer YOB: Information about the officer who conducted the stop.
- Stop Resolution: The outcome of the stop (e.g., offense report, field contact).
- Arrest Flag: Indicates whether an arrest was made during the stop.

```
# Checking for missing values
df.isnull().sum()
```

```
Subject Age Group              0
Subject ID                     0
GO / SC Num                    0
Terry Stop ID                  0
Stop Resolution                0
Weapon Type                    0
Officer ID                     0
Officer YOB                    0
Officer Gender                 0
Officer Race                   0
Subject Perceived Race         0
Subject Perceived Gender       0
Reported Date                  0
Reported Time                  0
Initial Call Type              0
Final Call Type                0
Call Type                      0
Officer Squad                561
Arrest Flag                    0
Frisk Flag                     0
Precinct                       0
Sector                         0
Beat                           0
dtype: int64
```

```
# Data description
df.describe()
```

```
         Subject ID   GO / SC Num  Terry Stop ID   Officer YOB
count  6.106800e+04  6.106800e+04   6.106800e+04  61068.000000
mean   7.305589e+09  2.018669e+13   1.221189e+10   1984.086903
std    1.275384e+10  8.570641e+10   1.758180e+10      9.471079
min   -8.000000e+00 -1.000000e+00   2.802000e+04   1900.000000
25%   -1.000000e+00  2.017000e+13   2.393990e+05   1979.000000
50%   -1.000000e+00  2.018000e+13   5.095875e+05   1986.000000
75%    7.753004e+09  2.021000e+13   1.973751e+10   1991.000000
max    5.861931e+10  2.024000e+13   5.863837e+10   2002.000000
```

- Count: This represents the total number of non-null entries in each column
- Mean: This is the average value for each column.
- Standard Deviation (std): This indicates the variability or dispersion of the data points from the mean
- Min and Max:These values show the smallest and largest entries in each column
- Percentiles (25%, 50%, 75%):These values represent the distribution of the data:
  - 25% (first quartile): 25% of the data falls below this value.
  - 50% (median): The middle value of the dataset.
  - 75% (third quartile): 75% of the data falls below this value.

```
df.columns
```

```
Index(['Subject Age Group', 'Subject ID', 'GO / SC Num', 'Terry Stop
ID',
       'Stop Resolution', 'Weapon Type', 'Officer ID', 'Officer YOB',
       'Officer Gender', 'Officer Race', 'Subject Perceived Race',
       'Subject Perceived Gender', 'Reported Date', 'Reported Time',
       'Initial Call Type', 'Final Call Type', 'Call Type', 'Officer
Squad',
       'Arrest Flag', 'Frisk Flag', 'Precinct', 'Sector', 'Beat'],
      dtype='object')
```

Data understanding

1. **Age Group** - Age Group (10 year increments) as reported by the officer. **ID**- Key, generated daily, identifying unique subjects in the dataset using a character to character match of first name and last name. "Null" values indicate an "anonymous" or "unidentified" subject. Subjects of a Terry Stop are not required to present identification. Read more
2. **GO / SC Num**- General Offense or Street Check number, relating the Terry Stop to the parent report. This field may have a one-to-many relationship in the data.
3. **Terry Stop ID** - Key identifying unique Terry Stop reports.
4. **Stop Resolution** -The resolution of the stop was as reported by the officer.
5. **Weapon Type** - Type of weapon, if any, identified during a search or frisk of the subject. Indicates "None" if no weapons was found.
6. **Officer ID**-Key identifying unique officers in the dataset.
7. **Officer YOB**- Year of birth, as reported by the officer.
8. **Officer Gender**- Gender of the officer, as reported by the officer.

9. **Officer Race**- Race of the officer, as reported by the officer.
10. **Subject Perceived Race**- Perceived race of the subject, as reported by the officer.
11. **Subject Perceived Gender** - Perceived gender of the subject, as reported by the officer.
12. **Reported Date**- Date the report was filed in the Records Management System (RMS). Not necessarily the date the stop occurred but generally within 1 day.
13. **Reported Time** - Time the stop was reported in the Records Management System (RMS). Not the time the stop occurred but generally within 10 hours.
14. **Initial Call Type**- Initial classification of the call as assigned by 911.
15. **Final Call Type**- Final classification of the call as assigned by the primary officer closing the event.
16. **Call Type**- How the call was received by the communication center.
17. **Officer Squad**- Functional squad assignment (not budget) of the officer as reported by the Data Analytics Platform (DAP).
18. **Arrest Flag**- Indicator of whether a "physical arrest" was made, of the subject, during the Terry Stop. Does not necessarily reflect a report of an arrest in the Records Management System (RMS).

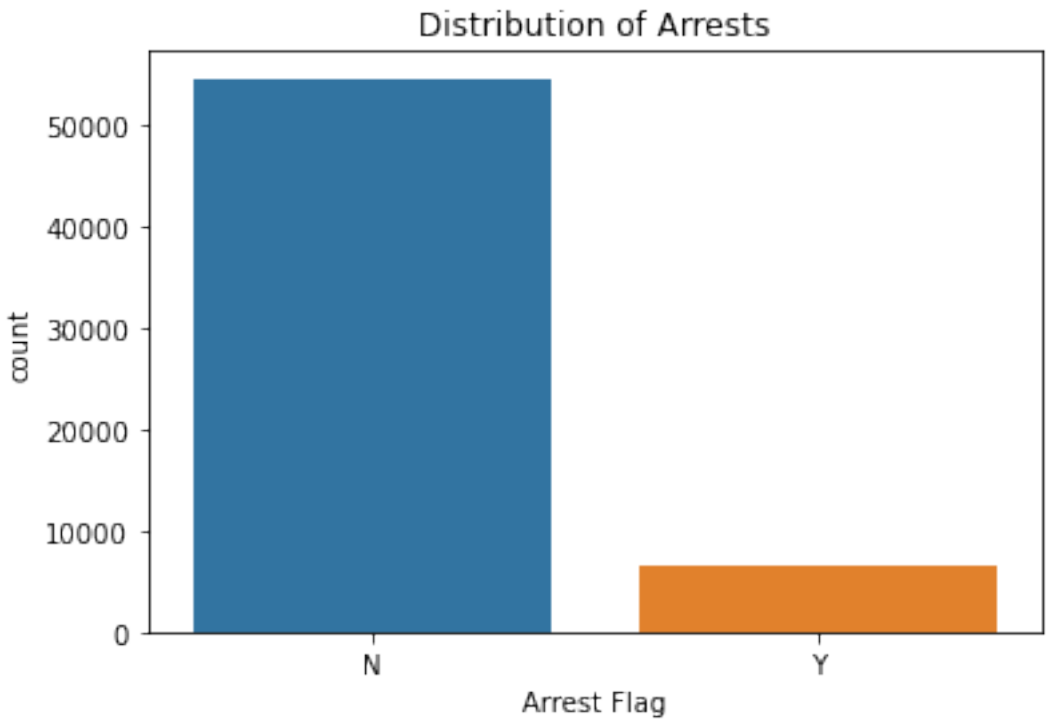
Provides for a list of all available colums in the dataset.

```
# Checking the unique values in the 'Arrest Flag' column
df['Arrest Flag'].unique()
```

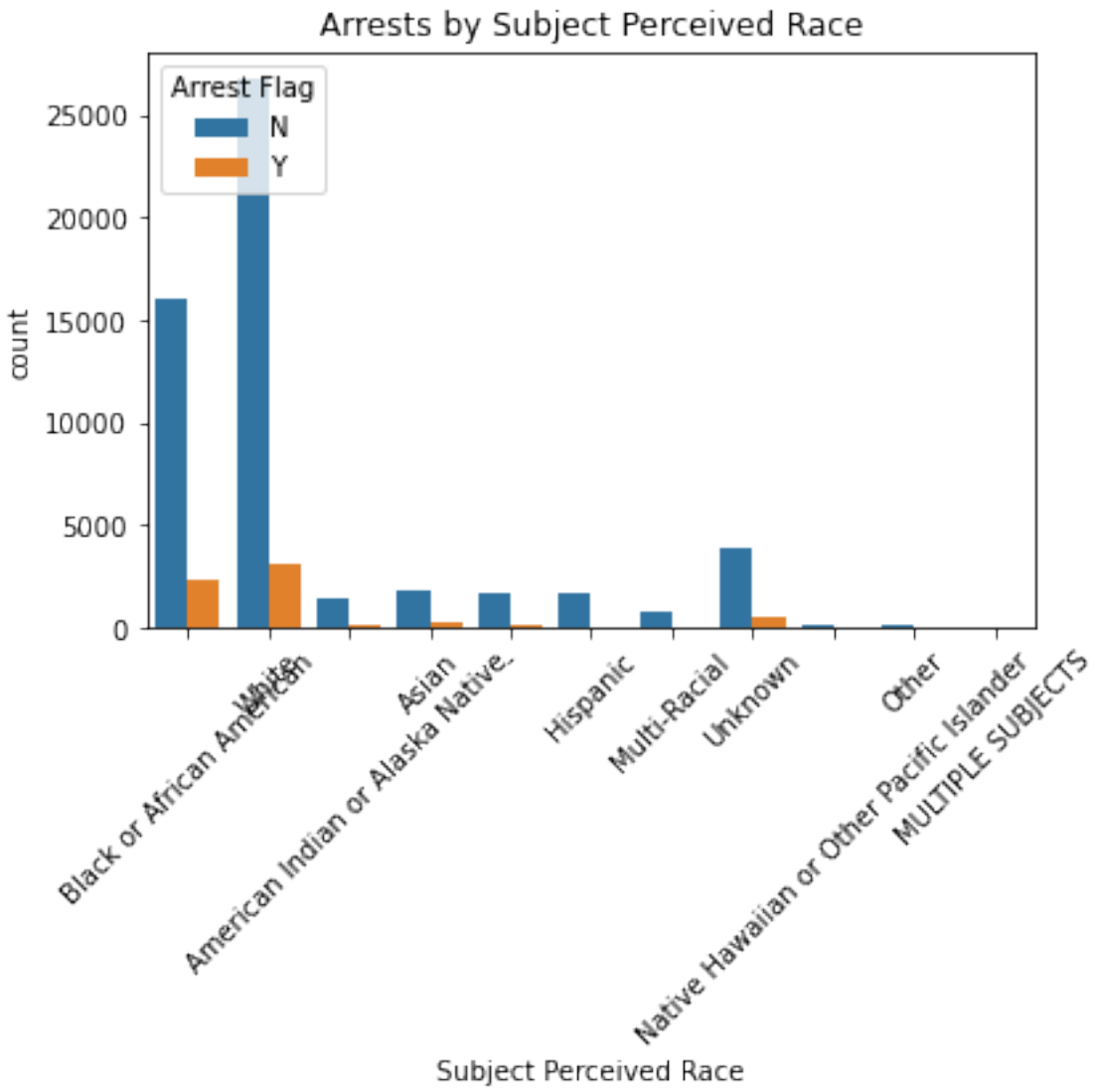
```
array(['N', 'Y'], dtype=object)
```

'N' means no arrest was made 'Y' means that an arrests made.

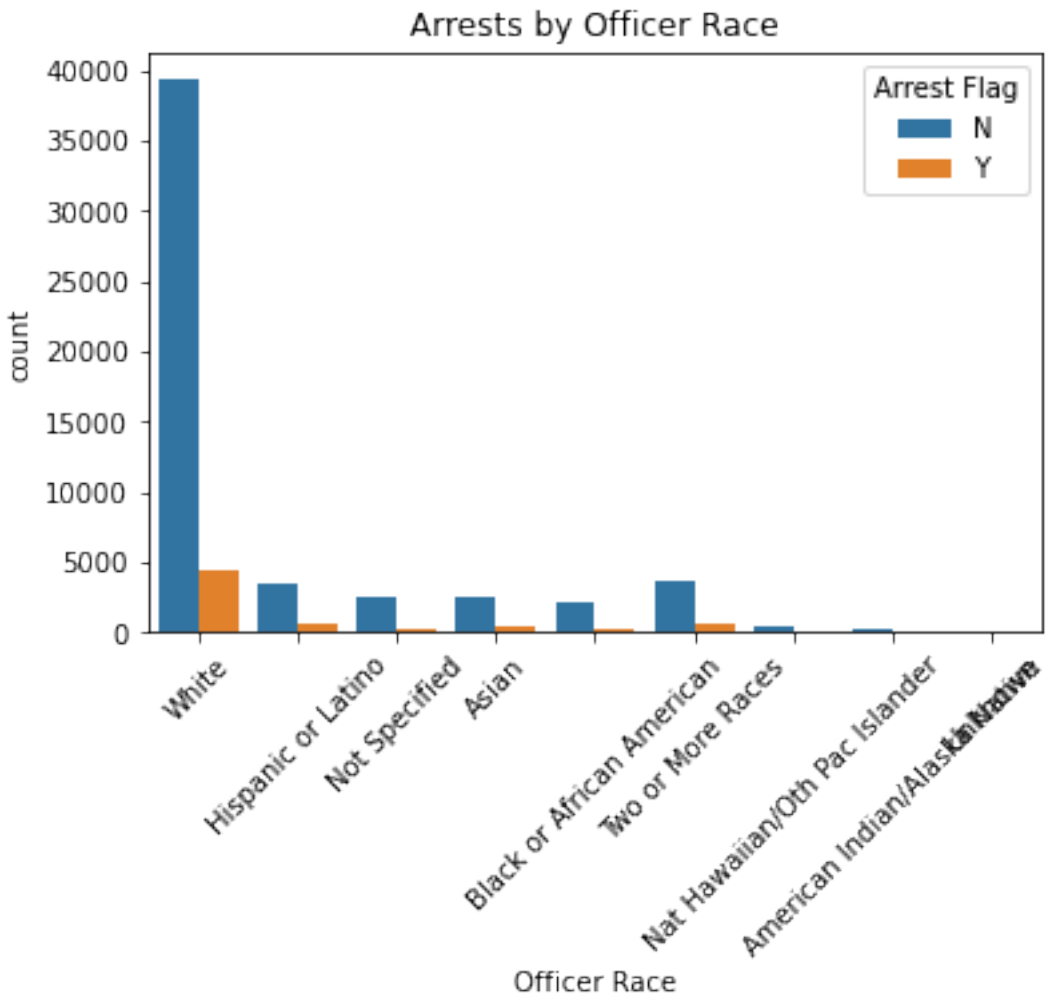
```
# Visualizing the arrests made.
sns.countplot(x='Arrest Flag', data=df)
plt.title('Distribution of Arrests')
plt.show()
```




```
# Visualizing the distribution of arrests by the perceived race of the subject
sns.countplot(x='Subject Perceived Race', hue='Arrest Flag', data=df)
plt.title('Arrests by Subject Perceived Race')
plt.xticks(rotation=45)  # Rotating x-axis labels for readability
plt.show()
```



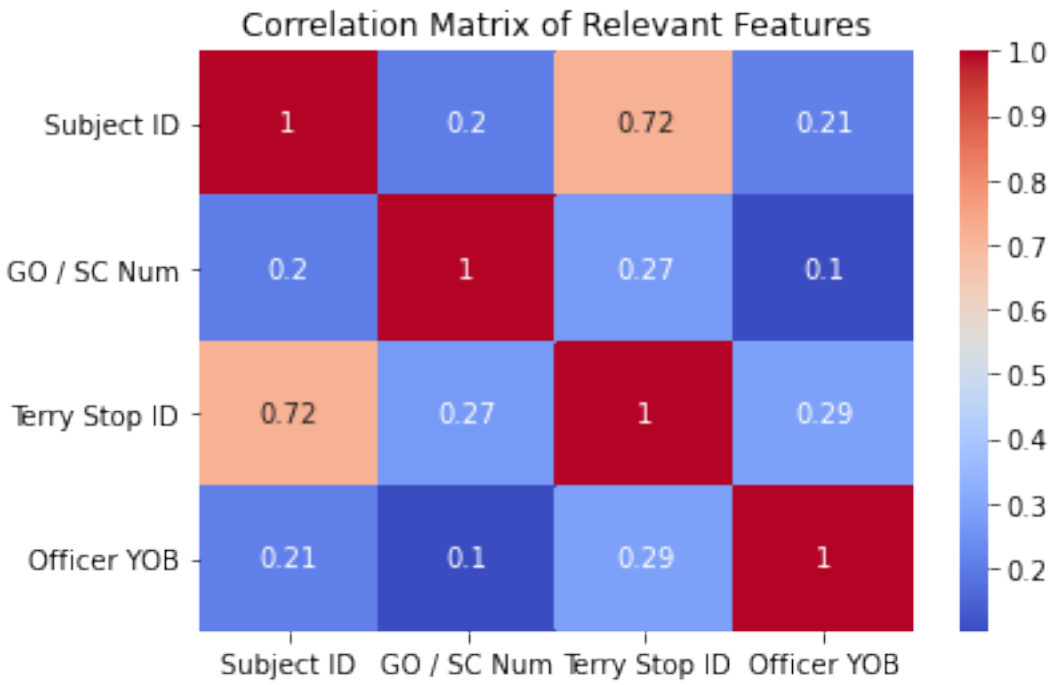

```
# Visualizing the distribution of arrests by officer race
sns.countplot(x='Officer Race', hue='Arrest Flag', data=df)
plt.title('Arrests by Officer Race')
plt.xticks(rotation=45)  # Rotating the x-axis labels for readability
plt.show()
```



Explanation

- `Officer Race` column to analyze the distribution of arrests made by officers of different racial backgrounds. The visualization shows that their is a likelihood of being arrested by officers of certain races.
  - For example, white officers have a higher likelihood to arrest an individual based on race.

```
# Correlation matrix visualisation
corr = df.corr()
sns.heatmap(corr, annot=True, cmap='coolwarm')
plt.title('Correlation Matrix of Relevant Features')
plt.show()
```



Explanation

1. Variables with high correlation:
   - **Subject ID and Terry Stop ID**: A correlation coefficient of 0.72 indicates a strong positive correlation. This suggests that as the Subject ID increases, the Terry Stop ID also tends to increase.
   - **GO / SC Num and Terry Stop ID**: A correlation coefficient of 0.27 indicates a moderate positive correlation.
2. Variables with a low correlation:
   - **Subject ID and GO / SC Num**: A correlation coefficient of 0.2 indicates a weak positive correlation.
   - **Officer YOB and other variables**: The correlations between Officer YOB and other variables are generally low (e.g., 0.21 with Subject ID, 0.1 with GO / SC Num, and 0.29 with Terry Stop ID), indicating that the officer's year of birth does not strongly correlate with these variables.

Data cleaning

```
#Dropping the null values/missing values
df = df.dropna()

# Checking for missing values
df.isnull().sum() #Confirming the values/columns with null values are
dropped
```

```
Subject Age Group            0
Subject ID                   0
GO / SC Num                  0
```

```
Terry Stop ID                0
Stop Resolution              0
Weapon Type                  0
Officer ID                   0
Officer YOB                  0
Officer Gender               0
Officer Race                 0
Subject Perceived Race       0
Subject Perceived Gender     0
Reported Date                0
Reported Time                0
Initial Call Type            0
Final Call Type              0
Call Type                    0
Officer Squad                0
Arrest Flag                  0
Frisk Flag                   0
Precinct                     0
Sector                       0
Beat                         0
dtype: int64
```

```
# Importing the libraries
from sklearn.preprocessing import LabelEncoder

# Encoding categorical variables
le = LabelEncoder()

df['Subject Perceived Race'] = le.fit_transform(df['Subject Perceived
Race'])
df['Subject Perceived Gender'] = le.fit_transform(df['Subject
Perceived Gender'])
df['Officer Race'] = le.fit_transform(df['Officer Race'])
df['Officer Gender'] = le.fit_transform(df['Officer Gender'])

from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import StandardScaler, OneHotEncoder

# Define numerical and categorical features
numerical_features = ['Officer YOB']  # Assuming 'Officer YOB' is used
as a numerical feature
categorical_features = [
    'Subject Perceived Race',
    'Subject Perceived Gender',
    'Officer Race',
    'Officer Gender',
    'Subject Age Group'
]
```

```
# Create the preprocessing pipelines
numerical_transformer = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='mean')),
    ('scaler', StandardScaler())
])

categorical_transformer = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='most_frequent')),
    ('onehot', OneHotEncoder(handle_unknown='ignore'))
])

# Combine both transformers into a ColumnTransformer
preprocessor = ColumnTransformer(
    transformers=[
        ('num', numerical_transformer, numerical_features),
        ('cat', categorical_transformer, categorical_features)
    ])

# Fit and transform the training data
X_train_processed = preprocessor.fit_transform(X_train)

# Transform the test data
X_test_processed = preprocessor.transform(X_test)

# Split the data into features and target
X = df.drop('Arrest Flag', axis=1)
y = df['Arrest Flag']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.3, random_state=42)
```

Modelling

Logistics Regression

```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, confusion_matrix,
accuracy_score

# Increase the max_iter parameter
model = LogisticRegression(max_iter=200)
```

Data scaling

```
from sklearn.preprocessing import StandardScaler

numerical_transformer = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='mean')),
```

```
    ('scaler', StandardScaler())
])

model = LogisticRegression(solver='liblinear')

model = LogisticRegression(max_iter=200, solver='liblinear')

# Fit the model
model.fit(X_train_processed, y_train)

# Predict on the test data
y_pred = model.predict(X_test_processed)

from sklearn.metrics import classification_report, confusion_matrix,
accuracy_score

# Logistic Regression model
lr_model = LogisticRegression(random_state=42)
lr_model.fit(X_train_processed, y_train)

# Predict on the test data
lr_pred = lr_model.predict(X_test_processed)

# Print results
print("\nLogistic Regression Model")
print(confusion_matrix(y_test, lr_pred))
print(classification_report(y_test, lr_pred))
print("Accuracy:", accuracy_score(y_test, lr_pred))
```

```
c:\Users\Mwende\anaconda3\envs\learn-env\lib\site-packages\sklearn\
linear_model\_logistic.py:762: ConvergenceWarning: lbfgs failed to
converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as
shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:

https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(


Logistic Regression Model
[[16259     0]
 [ 1894     0]]

c:\Users\Mwende\anaconda3\envs\learn-env\lib\site-packages\sklearn\
metrics\_classification.py:1221: UndefinedMetricWarning: Precision and
F-score are ill-defined and being set to 0.0 in labels with no
```

```
predicted samples. Use `zero_division` parameter to control this
behavior.
  _warn_prf(average, modifier, msg_start, len(result))

              precision    recall  f1-score   support

           N       0.90      1.00      0.94     16259
           Y       0.00      0.00      0.00      1894

    accuracy                           0.90     18153
   macro avg       0.45      0.50      0.47     18153
weighted avg       0.80      0.90      0.85     18153

Accuracy: 0.8956646284360712
```

The model has an accuracy of 89.57%

Decision Tree

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import classification_report, confusion_matrix,
accuracy_score

# Decision Tree model
dt_model = DecisionTreeClassifier(random_state=42)
dt_model.fit(X_train_processed, y_train)

# Predictions
dt_pred = dt_model.predict(X_test_processed)
lr_pred = lr_model.predict(X_test_processed)

# Evaluation
print("Decision Tree Model")
print(confusion_matrix(y_test, dt_pred))
print(classification_report(y_test, dt_pred))
print("Accuracy:", accuracy_score(y_test, dt_pred))

Decision Tree Model
[[15947   312]
 [ 1797    97]]
              precision    recall  f1-score   support

           N       0.90      0.98      0.94     16259
           Y       0.24      0.05      0.08      1894

    accuracy                           0.88     18153
   macro avg       0.57      0.52      0.51     18153
weighted avg       0.83      0.88      0.85     18153

Accuracy: 0.8838208560568501
```

For "N" (Negative cases):

- Precision (0.90): If the model says it's "N", there's a 90% chance it’s right. Nice!
- Recall (0.98): Out of all the actual "N" cases, the model nailed 98% of them. Impressive!
- F1-Score (0.94): This is like the average of precision and recall for "N"—pretty strong.

For "Y" (Positive cases):

- Precision (0.24): When the model predicts "Y", it’s correct only 24% of the time. Oops!
- Recall (0.05): Out of all actual "Y" cases, the model only identified 5%. It’s missing a lot here.
- F1-Score (0.08): The combined score for "Y" is quite low, reflecting the challenges in getting "Y" right.

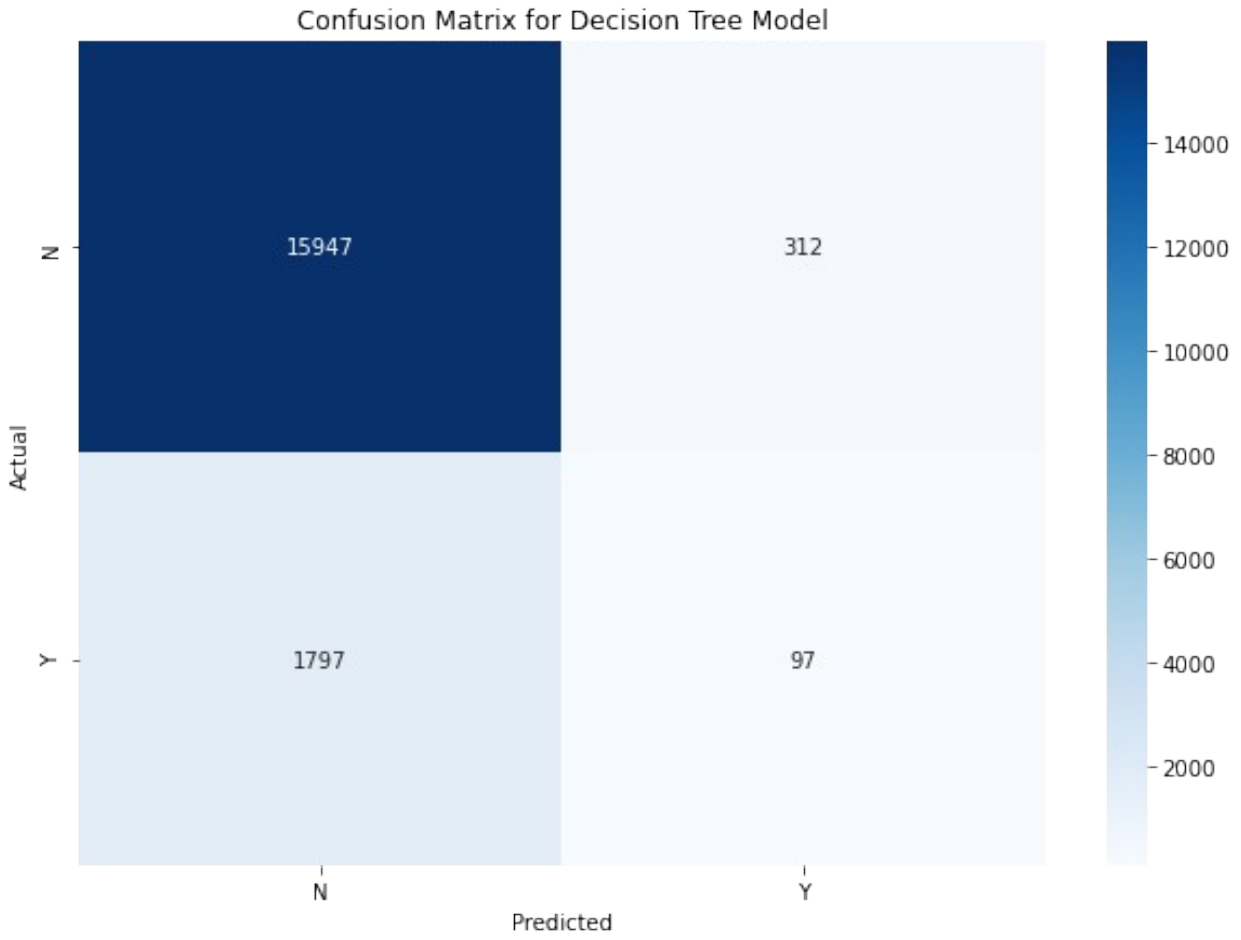
Overall Performance of model:

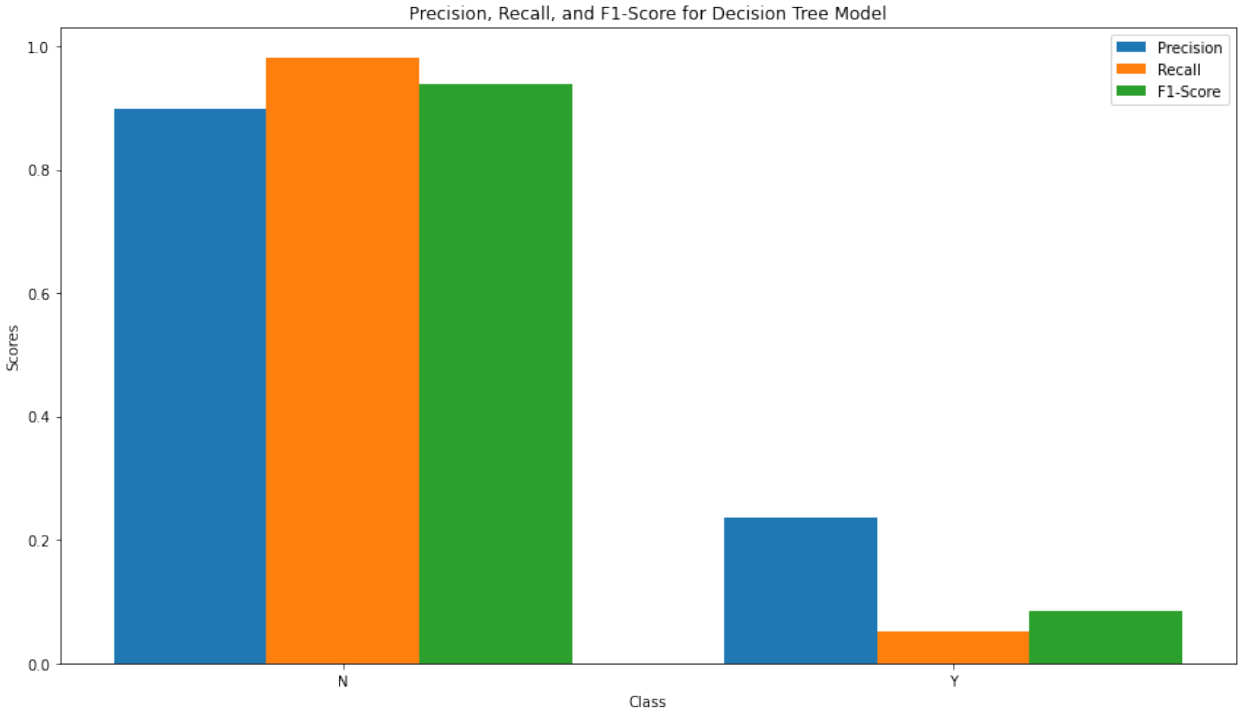
- Accuracy (0.88): Overall, the model’s accuracy is 88%, which sounds good, but it’s mainly because it’s really good at identifying "N".
- Macro Average: This is an average across both classes. The model's performance is balanced between the two classes, but the lower scores for "Y" drag this down.
- Weighted Average: This takes into account the size of each class. It shows that while the model does a decent job overall (with a weighted F1-score of 0.85), it's still struggling with "Y" because it’s a smaller group.

Summary: The model is doing a well with negative cases, but it’s having a hard time spotting the positive ones. This could be due to an imbalance in the number of cases for each class.

```
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
from sklearn.metrics import confusion_matrix, classification_report

# Generate the confusion matrix
conf_matrix = confusion_matrix(y_test, dt_pred, labels=['N', 'Y'])

# Plot the confusion matrix
plt.figure(figsize=(10, 7))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues',
            xticklabels=['N', 'Y'], yticklabels=['N', 'Y'])
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix for Decision Tree Model')
plt.show()

# Extract precision, recall, and f1-score from classification report
report = classification_report(y_test, dt_pred, target_names=['N',
'Y'], output_dict=True)
classes = ['N', 'Y']
precision = [report[cls]['precision'] for cls in classes]
recall = [report[cls]['recall'] for cls in classes]
```

```
f1_score = [report[cls]['f1-score'] for cls in classes]

# Plot Precision, Recall, and F1-Score
x = np.arange(len(classes))  # the label locations
width = 0.25  # the width of the bars

fig, ax = plt.subplots(figsize=(12, 7))
rects1 = ax.bar(x - width, precision, width, label='Precision')
rects2 = ax.bar(x, recall, width, label='Recall')
rects3 = ax.bar(x + width, f1_score, width, label='F1-Score')

# Add some text for labels, title and custom x-axis tick labels, etc.
ax.set_xlabel('Class')
ax.set_ylabel('Scores')
ax.set_title('Precision, Recall, and F1-Score for Decision Tree
Model')
ax.set_xticks(x)
ax.set_xticklabels(classes)
ax.legend()

fig.tight_layout()

plt.show()
```


Confusion Matrix for Decision Tree Model
15947
312
1797
97
N
Y
Actual
N
Y
Predicted
14000
12000
10000
8000
6000
4000
2000


Precision, Recall, and F1-Score for Decision Tree Model
Precision
Recall
F1-Score
Scores
1.0
0.8
0.6
0.4
0.2
0.0
N
Y
Class

Summary of Visualizations

Confusion Matrix Heatmap:

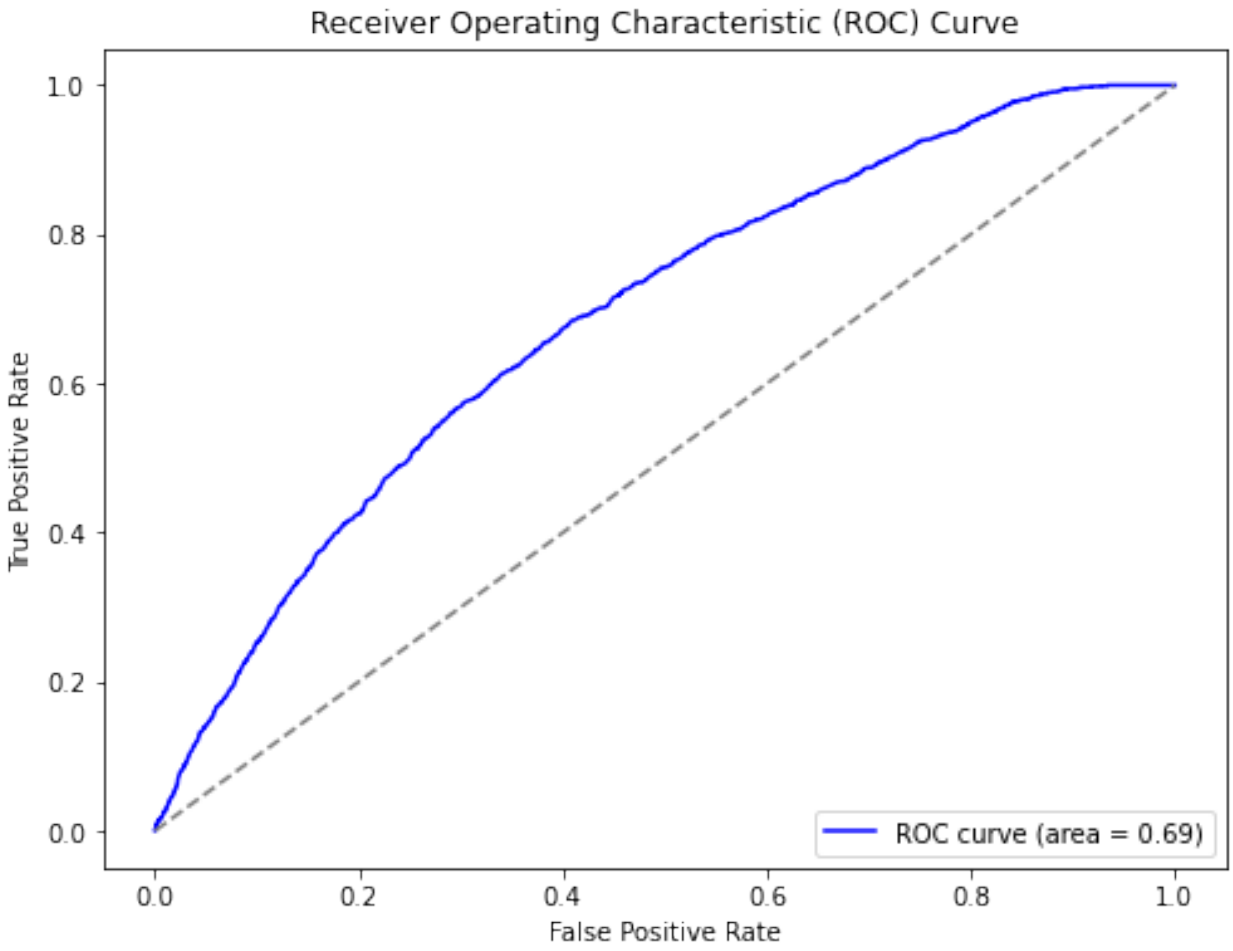
The confusion matrix heatmap provides a clear view of the model's performance across different classes by displaying the number of correct and incorrect predictions. It shows high precision and recall for the 'N' class (negative cases), as indicated by the darker colors in the true negative cell. This suggests that the model excels at identifying non-arrest cases. Conversely, the model struggles with the 'Y' class (positive cases), as evidenced by the lighter color in the true positive cell. This indicates a low number of correctly identified arrests, highlighting a major area of concern.

Precision, Recall, and F1-Score Bar Plot:

The bar plot illustrates the precision, recall, and F1-score for both classes. For the 'N' class, the model demonstrates high precision (90%), high recall (98%), and a strong F1-score (0.94), reflecting its effectiveness in identifying negative cases. However, for the 'Y' class, the model's performance is significantly weaker. Precision for 'Y' is only 24%, recall is a mere 5%, and the F1-score is low (0.08). These figures reveal that while the model is accurate in predicting 'N' cases, it has considerable difficulty correctly identifying 'Y' cases, suggesting issues such as class imbalance or a need for better predictive strategies.

ROC Curve

```
from sklearn.metrics import roc_curve, auc

# Predict probabilities for the positive class (assumed to be 'Y')
y_prob = lr_model.predict_proba(X_test_processed)[:, 1]

# Specify the positive label in the roc_curve function
fpr, tpr, thresholds = roc_curve(y_test, y_prob, pos_label='Y')
roc_auc = auc(fpr, tpr)

# Plotting the ROC curve
plt.figure(figsize=(8, 6))
plt.plot(fpr, tpr, color='blue', label=f'ROC curve (area =
{roc_auc:.2f})')
plt.plot([0, 1], [0, 1], color='grey', linestyle='--')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC) Curve')
plt.legend(loc='lower right')
plt.show()
```



The ROC curve has an AUC of 0.69, indicating that the model performs better than random guessing. The curve's proximity to the diagonal line suggests that while the model has some ability to differentiate between positive and negative classes, its predictive accuracy is not particularly strong.

Conclusion and recommendations

Conclusion

This project analyzed arrest outcomes during Terry Stops, with a focus on understanding the potential influence of factors such as race, gender, and other demographic variables. The classification models used—logistic regression and decision tree—provided insights into the predictive factors affecting arrest decisions.

Key Findings:

1. **Logistic Regression Model**:
   - **Accuracy**: 89.57%
   - The logistic regression model performed well in predicting negative cases ("N"), with high precision and recall.

- However, it struggled with positive cases ("Y"), demonstrating lower precision and recall. This indicates an imbalance in the dataset or that positive cases are less well represented or harder to predict.

2. **Decision Tree Model**:
   - The decision tree model achieved an overall accuracy of 88%.
   - Like the logistic regression model, it was more effective at identifying negative cases and had lower performance on positive cases.
   - The feature importance analysis revealed that certain features have a stronger influence on the decision to arrest, though there were discrepancies in feature names and importance scores that need addressing.
3. **ROC Curve**:
   - The ROC curve for the logistic regression model had an AUC of 0.69, indicating the model has some discriminatory power but is not highly effective.
4. **Feature Importance**:
   - The decision tree visualization highlighted that features related to demographics and weapons had varying levels of importance. However, inconsistencies in feature names and their extraction process affected the accuracy of the visualization.

Recommendations

1. **Addressing Class Imbalance**:
   - The models showed a strong performance on negative cases but struggled with positive cases. Consider techniques like resampling (oversampling positive cases or undersampling negative cases) or using algorithms designed to handle class imbalance (e.g., SMOTE or balanced class weights in classifiers).
2. **Ethical Considerations**:
   - Advocate for transparency in policing practices and consider incorporating fairness metrics into model evaluation to assess and mitigate potential biases.
3. **Conduct a deeper analysis of feature importance and its impact on arrest outcomes.** Investigate the role of each demographic variable and its contribution to model predictions. Explore temporal aspects (e.g., time of day) and geographical factors (if available) to gain a more comprehensive understanding of arrest patterns.
4. **Collaboration and Feedback**:
   - Engage with stakeholders, including law enforcement agencies and community representatives, to discuss findings and gather feedback. Collaborate to develop actionable strategies for improving fairness and transparency in Terry Stops.

By implementing these recommendations, you can enhance the effectiveness of predictive models and contribute to more equitable policing practices.