Supervised Machine Learning Model Integration Using Flask (Web Framework)

Project # 4

By: Mubashira Qari

Data Source:

- The datasets from the NYPD Stop, Question, and Frisk database are used in this project and are available for download from the links provided below. Data is made available in CSV format.
- Publications, Reports - NYPD

Problem Worth Solving, Analyzing, and Visualizing:

- ‘Stop, Question, and Frisk’ database is used in this machine learning project.
- The aim is to predict, whether the suspect will be arrested or not.
- And to predict, whether the summons will be issued or not for the suspect.

Implementation:

The project implementation is done using Scikit-learn library in machine learning, along with the following.

- Python Pandas
- Flask Web Framework
- Python Matplotlib
- HTML/CSS/Bootstrap
- JavaScript D3.js
- PostgreSQL Database
- Tableau

Host application using Heroku

Exploratory Data Analysis (EDA)

- The 2017, 2018 and 2019 ‘Stop, Question and Frisk’ datasets are merged together to perform the preprocessing steps and prepare for the training dataset.
- The 2020 ‘Stop, Question and Frisk’ dataset is used to perform the preprocessing steps and prepare for the testing dataset.

Data Preprocessing (ETL):

- Merging 2017,18, &19 for training data and keeping 2020 for testing.
- Removing unwanted text from columns.
- Finding null values and checking for missing values.
- Convert time into seconds to have integer values for machine learning.
- Replacing the text strings with zeros in the integer columns.
- Fixing two different names for single category.
- Removing special characters from the data values.

Data Preprocessing (ETL):

- Rename the columns.
- Dropping columns which are not impacting the outcome.
- Converting columns to correct data type.
- Binning the categorical values in almost 10 out of 27 columns .
- Converting values to numeric using Label Encoder for two category values columns.
- Converting values to numeric using ‘get_dummies’ function for hierarchical category columns.
- Normalizing the data using StandardScaler() function.

Random Forest Classifier for Arrest Prediction:

Random Forest Classifier:

```
# Fit a model, and then print a classification report
clf = RandomForestClassifier(random_state=1).fit(X_train_scaled, y_train)
y_pred = clf.predict(X_test_scaled)
print(classification_report(y_test, y_pred))
print(f'Training Score: {clf.score(X_train_scaled, y_train)}')
print(f'Testing Score: {clf.score(X_test_scaled, y_test)}')
```

```
              precision    recall  f1-score   support

           0       0.83      0.90      0.86      5987
           1       0.80      0.69      0.74      3557

    accuracy                           0.82      9544
   macro avg       0.82      0.79      0.80      9544
weighted avg       0.82      0.82      0.82      9544

Training Score: 0.9999445921985816
Testing Score: 0.8217728415758592
```

Arrest Issued -NO
Arrest Issued -YES

	Actual	Predicted
Positives(1's)	10884	- 1189(FN) - 2368(TP)
Negatives (0's)	25212	- 472(FP) - 5515(TN)

Total - 36,096

Random Forest Classifier for Summons Prediction:

		precision	recall	f1-score	support
Summons Issued -NO	0	0.97	1.00	0.99	9281
Summons Issued -YES	1	0.00	0.00	0.00	263
accuracy				0.97	9544
macro avg		0.49	0.50	0.49	9544
weighted avg		0.95	0.97	0.96	9544

Training Score: 0.9999445921985816
Testing Score: 0.9724434199497066

```
cm = confusion_matrix(y_true, y_pred)
tn, fp, fn,  tp = cm.ravel()
tn, fp, fn,  tp
```

```
(9281, 0, 263, 0)
```

	Actual	Predicted
Positives(1's)	1040 -----	263 (FN) ----- 0(TP)
Negatives (0's)	35056 ----	0 (FP) - 9281(TN)

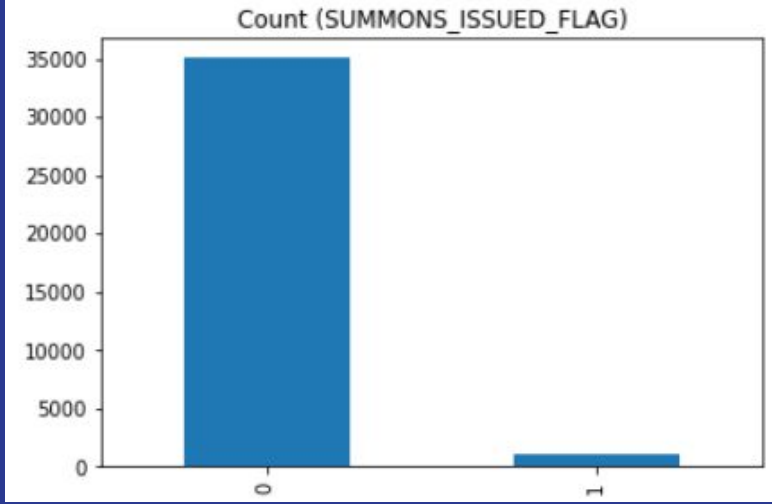
Total - 36,096

Techniques & Challenges for Handling Imbalance Data:

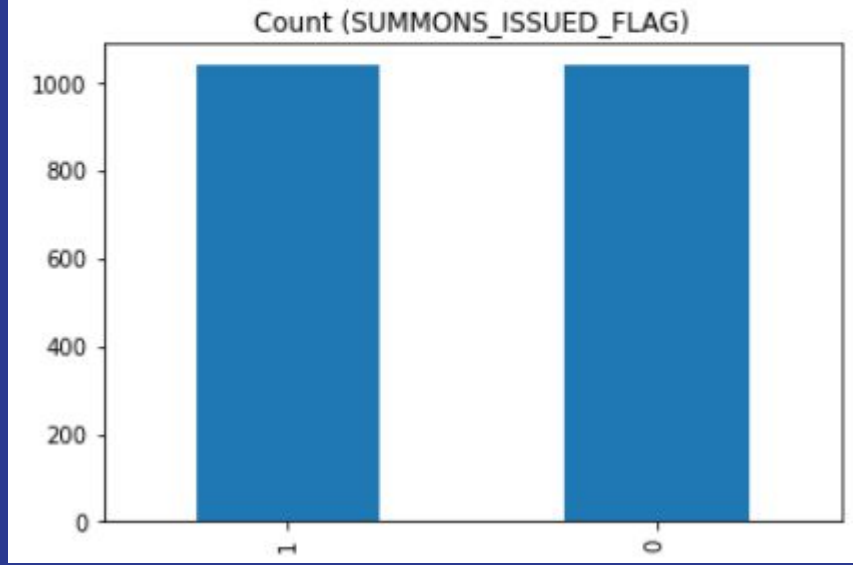
- Undersampling : Sampling from the majority class in order to keep only a part of these points.
- Oversampling: Replicating some points from the minority class in order to increase its cardinality.
- SMOTE: generating synthetic data or creating new synthetic points from the minority class to increase cardinality.
- When using the resampling techniques, we are training the model on wrong proportions of two classes.Thus classifier learned this way will have lower accuracy on the future real test data than the classifier trained on unchanged data. So the true proportion of classes is important to know for classifying a new point, which gets lost due to resampling.
- So, when applying these methods, one needs to be careful about the goal.

Analyzing the Outcome Data:

```
Before Random under-sampling:
0    35056
1     1040
Name: SUMMONS_ISSUED_FLAG, dtype: int64
```




```
Random under-sampling:
1    1040
0    1040
Name: SUMMONS_ISSUED_FLAG, dtype: int64
```



Refitting Random Forest Classifier on Balance Data for Summons Prediction:

```
# Import a Random Forests classifier
from sklearn.ensemble import RandomForestClassifier
```

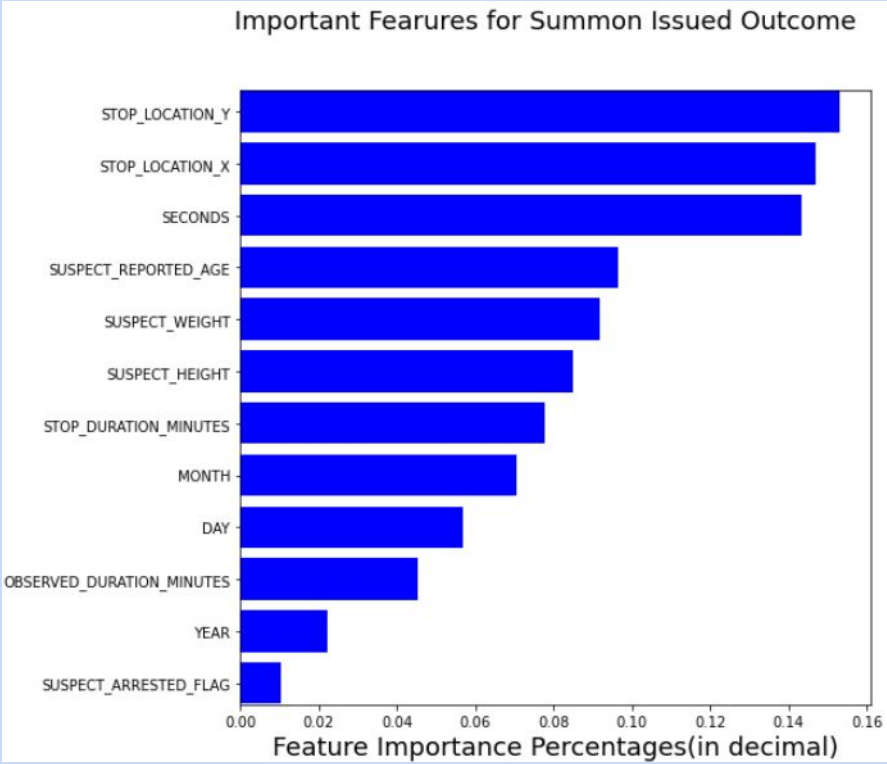
```
clf = ExtraTreesClassifier(random_state=1).fit(X_sm, y_sm)
y_pred = clf.predict(X_test_scaled)
print(classification_report(y_test, y_pred, target_names=target_names))
print(f'Training Score: {clf.score(X_sm, y_sm)}')
print(f'Testing Score: {clf.score(X_test_scaled, y_test)}')
```

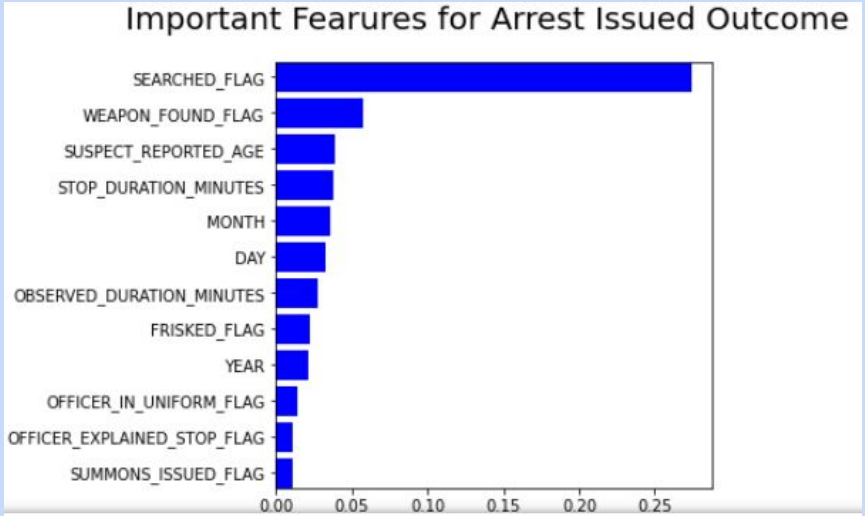
```
              precision    recall  f1-score   support

          NO       0.97      0.90      0.93      9281
         YES       0.05      0.18      0.08       263

    accuracy                           0.88      9544
   macro avg       0.51      0.54      0.51      9544
weighted avg       0.95      0.88      0.91      9544

Training Score: 0.9999857371063441
Testing Score: 0.8780385582564962
```

- Lower accuracy on the future real test data than the classifier trained on unchanged data.

Finding the Important Features

Twelve most important features are selected for prediction purposes and models are saved using joblib tools.



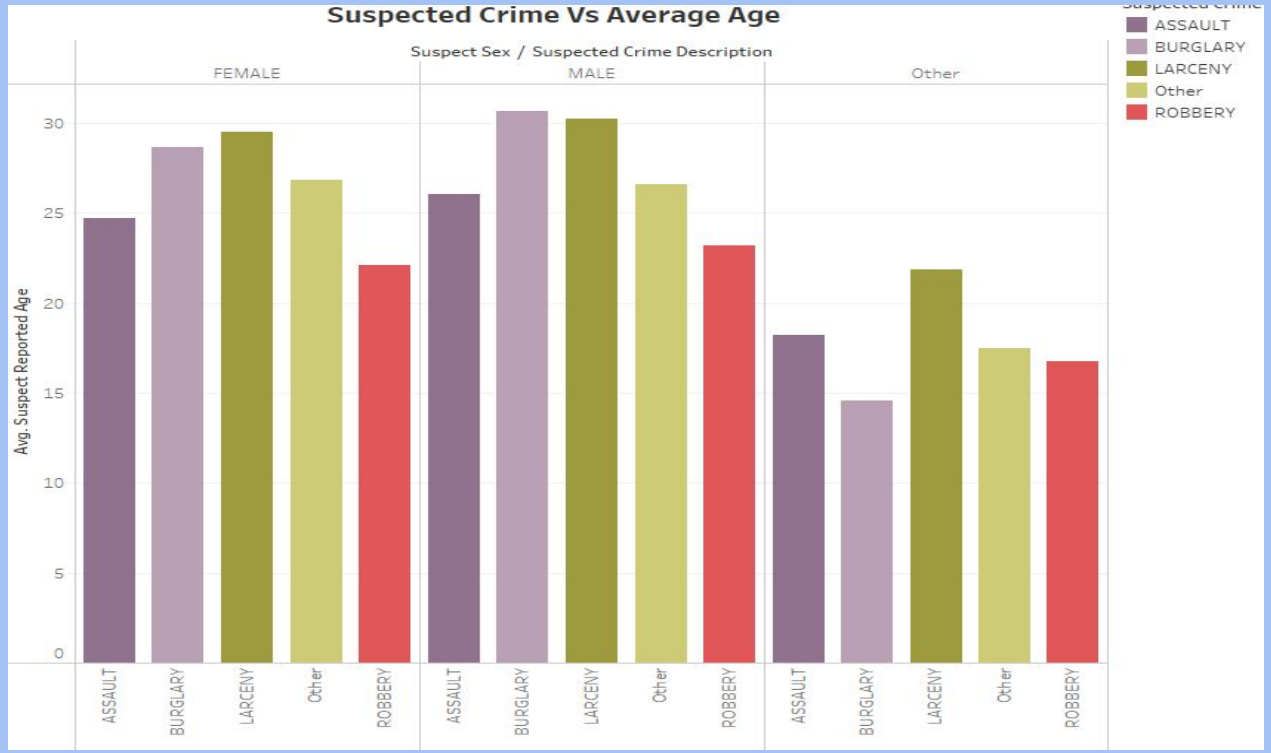


Flask Application:

Model is fetched and prediction is made in file main.py

```
@app.route('/api/generate_arrest_prediction', methods=['POST'])
def generate_prediction_arrest():
    user_inputs=request.json
    predict_df=pd.DataFrame({
    'SEARCHED_FLAG':[int(user_inputs['SEARCHED_FLAG'])],
    'WEAPON_FOUND_FLAG':[int(user_inputs['WEAPON_FOUND_FLAG'])],
    'SUSPECT_REPORTED_AGE':[int(user_inputs['SUSPECT_REPORTED_AGE'])],
    'STOP_DURATION_MINUTES':[int(user_inputs['STOP_DURATION_MINUTES'])],
    'MONTH':[int(user_inputs['MONTH'])],
    'DAY':[int(user_inputs['DAY'])],
    'OBSERVED_DURATION_MINUTES':[int(user_inputs['OBSERVED_DURATION_MINUTES'])],
    'FRISKED_FLAG':[int(user_inputs['FRISKED_FLAG'])],
    'YEAR':[int(user_inputs['YEAR'])],
    'OFFICER_EXPLAINED_STOP_FLAG':[int(user_inputs['OFFICER_EXPLAINED_STOP_FLAG'])],
    'OFFICER_IN_UNIFORM_FLAG':[float(user_inputs['OFFICER_IN_UNIFORM_FLAG'])],
    'SUMMONS_ISSUED_FLAG':[float(user_inputs['SUMMONS_ISSUED_FLAG'])]
    })
    prediction_arrest=str(trained_machine_learning_model_1.predict(predict_df)[0])
    return jsonify([prediction_arrest])
```

Tableau Visual Analytics



Heroku Deployment: Heroku Deployment

Lets see all this is action now!!!