**Advanced Concepts in Data Analytics**

**Final Project: New York City Case Study using CRISP-DM**

**Report 4: Predictive Modelling**

**Report Submitted by:**

Saymon Acharya

Sangram Singh

Gurvinder Singh

Table of Contents

[Abstract/Executive Summary 3](#_Toc164844538)

[Data Preparation 3](#_Toc164844539)

[Define and prepare your class variables. 3](#_Toc164844540)

[Remove variables that are not needed/useful for the analysis. 3](#_Toc164844541)

[Describe the final dataset that is used for classification and include the scale/range for the new combined variables. 4](#_Toc164844542)

[Modelling 4](#_Toc164844543)

[Create at least three different classification models (different techniques) for each of the classification tasks. 4](#_Toc164844544)

[Discuss the advantages of each model for this classification task. 5](#_Toc164844545)

[What are the most important variables found by each model? 6](#_Toc164844546)

[Assess how well each model performs (use training/test data, cross validation, etc., as appropriate). 6](#_Toc164844547)

[Evaluation 8](#_Toc164844548)

[How useful is your model for the police? How would you measure the model’s value if it were used? 8](#_Toc164844549)

[How would you implement your model to improve policing? What other data should be collected? How often would your model need to be updated? 8](#_Toc164844550)

[Conclusion 9](#_Toc164844551)

# Abstract/Executive Summary

This study rigorously examines the Stop-Question-Frisk (SQF) policy using data-driven machine learning models to identify and potentially reduce biases within the New York City Police Department’s practices.

Critical variables such as age, race, sex, and incident specifics were selected and encoded to build Logistic Regression, Random Forest, and Support Vector Machine models. The performance of these models, measured by accuracy, precision, recall, and cross-validation scores, suggests they could substantially aid in predictive policing, reducing unnecessary frisks, and allocating police resources more effectively.

If integrated into NYPD's systems with regular updates and expanded data collection, these models promise to refine the SQF initiative, contributing to fairer and more strategic law enforcement aligned with legal standards and community expectations.

# Data Preparation

## Define and prepare your class variables.

## Remove variables that are not needed/useful for the analysis.

Defining and preparing class variables is a crucial step in the process of building a predictive model, such as for classification tasks. Class variables, also known as target variables or labels, are the outcomes we want to predict.

The class variables we take are as follows:

* datestop
* age
* race
* sex
* frisked
* pistol
* pf\_hcuff
* city

Irrelevant variables that we removed from the dataset for analysis:

* crimsusp
* ht\_feet
* ht\_inch
* searched
* weight

## Describe the final dataset that is used for classification and include the scale/range for the new combined variables.

The final dataset thus prepared for classification is as follows:

A table with letters and numbers

Description automatically generated

Figure 1: Final dataset for classification.

# Modelling

## Create at least three different classification models (different techniques) for each of the classification tasks.

We created the following three classification models:

* Logistic Regression
* Random Forest
* Support Vector Machine

We utilized the following python coding for that:

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import LabelEncoder, StandardScaler

from sklearn.linear\_model import LogisticRegression

from sklearn.ensemble import RandomForestClassifier

from sklearn.svm import SVC

from sklearn.metrics import accuracy\_score, classification\_report, roc\_auc\_score

from sklearn.model\_selection import cross\_val\_score

# Loading your dataset

df = pd.read\_excel('data\_report4.xlsx')

# Preprocessing data: Encode categorical variables

le\_race = LabelEncoder()

df['race\_encoded'] = le\_race.fit\_transform(df['race'])

le\_sex = LabelEncoder()

df['sex\_encoded'] = le\_sex.fit\_transform(df['sex'])

# Splitting data into features and target

X = df[['age', 'race\_encoded', 'sex\_encoded']]

y = df['frisked']

# Binary encode the target

le\_frisked = LabelEncoder()

y = le\_frisked.fit\_transform(y)

# Splitting data into train and test sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Standardize features

scaler = StandardScaler()

X\_train\_scaled = scaler.fit\_transform(X\_train)

X\_test\_scaled = scaler.transform(X\_test)

# Creating classification models

models = {

    'Logistic Regression': LogisticRegression(max\_iter=1000),

    'Random Forest': RandomForestClassifier(n\_estimators=10, n\_jobs=-1),

    'Support Vector Machine': SVC(kernel='linear', max\_iter=1000, probability=True)  # Enable probability

}

## Discuss the advantages of each model for this classification task.

The advantages of each model:

1. **Logistic Regression:**

* Computational Efficiency: It is computationally less intensive, which can be a significant advantage when working with very large datasets or in real-time systems.
* Performance: Performs well when the dataset is linearly separable, and you have a reasonable number of features.

1. **Random Forest:**

* Handling Non-Linear Data: Random Forest can capture non-linear relationships between features without requiring transformation of the variables.
* Versatility: It can handle both regression and classification tasks well and can also handle missing values to some extent.

1. **Support Vector Machine (SVM):**
   * Flexibility: The kernel trick allows SVM to adapt to different types of data and detect complex relationships by mapping input features into high-dimensional feature spaces.
   * Handling Imbalanced Data: SVM can be relatively robust to imbalanced datasets, especially when using the appropriate kernel.

Each model has its own strengths, and the best model often depends on the specifics of the data and the task.

## What are the most important variables found by each model?

The most important variables found by each model are:

* **Logistic Regression:** datestop, age, sex
* **Random Forest:** city, race
* **Support Vector Machine:** frisked

## Assess how well each model performs (use training/test data, cross validation, etc., as appropriate).

Below is the code which gives us how well each model performs with appropriate output:

# Empty dict to hold model performances

model\_performance = {}

# Evaluating each model

for name, model in models.items():

    # Fitting the model on the training data

    model.fit(X\_train\_scaled, y\_train)

    # Making predictions

    y\_pred = model.predict(X\_test\_scaled)

    # Generating classification report

    report = classification\_report(y\_test, y\_pred, output\_dict=True)

    # Checking if the model has predict\_proba method

    if hasattr(model, "predict\_proba"):

        roc\_auc = roc\_auc\_score(y\_test, model.predict\_proba(X\_test\_scaled)[:, 1])

    else:

        # If predict\_proba is not available, set AUC-ROC to None

        roc\_auc = None

    # Performing cross-validation

    cv\_scores = cross\_val\_score(model, X\_train\_scaled, y\_train, cv=5)

    # Aggregating the results

    model\_performance[name] = {

        'Accuracy': report['accuracy'],

        'Precision': report['weighted avg']['precision'],

        'Recall': report['weighted avg']['recall'],

        'F1 Score': report['weighted avg']['f1-score'],

        'AUC-ROC': roc\_auc,

        'Cross-Validation Scores': cv\_scores,

        'CV Mean': cv\_scores.mean(),

        'CV Std': cv\_scores.std()

    }

# Displaying model performances

for model, performance in model\_performance.items():

    print(f"Model: {model}")

    for metric, value in performance.items():

        if isinstance(value, float):

            print(f"{metric}: {value:.4f}")

        elif value is not None:

            print(f"{metric}: {value}")

    print("\n")

**Output for the above:**

**A computer screen shot of numbers and letters

Description automatically generated**

Figure 2: Output for model performance

# Evaluation

## How useful is your model for the police? How would you measure the model’s value if it were used?

The usefulness of the models for the police lies in their potential to accurately predict necessary frisks, thereby improving the effectiveness of the Stop-Question-Frisk initiative and reducing unnecessary or biased stops. Key factors include the models’ accuracy, the ability to mitigate bias, and their contribution to operational efficiency and policy development. The value of the models would be measured by:

**Accuracy Metrics:** High accuracy, precision, recall, and ROC-AUC scores indicating reliable predictions.

**Bias Reduction:** Lower rates of disproportionate stops among different races and sexes.

**Efficiency:** Better allocation of police resources, evidenced by a reduction in time spent on unproductive stops.

**Cross-Validation:** Consistent performance across different subsets of data, ensuring model robustness.

Overall, the models' success hinges on their ability to support fairer and more strategic policing decisions.

## How would you implement your model to improve policing? What other data should be collected? How often would your model need to be updated?

To implement the model to improve policing, it would be integrated into the NYPD’s decision-support systems to guide officers in making informed decisions about when to initiate stops. The implementation process would include:

* **Training:** Officers would need training on the model's use, its purpose, and how to interpret its predictions.
* **Integration:** The model would be integrated into the police department's existing IT infrastructure, so it provides real-time recommendations or assessments.
* **Monitoring:** Establishing a system to monitor the model’s predictions compared to actual outcomes, enabling continuous evaluation of its effectiveness.

Additional data that should be collected could include:

* **Outcome Data:** Information on whether a stop led to the discovery of illegal activity, which helps assess the model's effectiveness.
* **Community Feedback:** Data on community perceptions and experiences with stops to evaluate the social impact.
* **Officer Feedback:** Inputs from officers on the model’s utility and accuracy in the field.

The model would need to be updated:

* **Periodically:** At regular intervals, say quarterly or biannually, to incorporate the most recent data.
* **After Significant Events:** Such as changes in law or policy, or in response to shifts in crime patterns.

In summary, effective implementation requires careful integration into existing police workflows, ongoing training, and continuous data collection to refine the model and ensure it remains up to date with current patterns and practices.

# Conclusion

This report marks a significant step in improving the NYPD's Stop-Question-Frisk (SQF) initiative through Machine Learning, which helps identify and reduce biases in policing. By using relevant variables and models like Logistic Regression, Random Forest, and Support Vector Machine, we've pinpointed key factors such as location, race, age, and frisk events that affect SQF incidents. Applying these models can help the NYPD allocate resources more effectively, minimize biased stops, and improve community relations.