**Advanced Concepts in Data Analytics**

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

**Report 3: Cluster Analysis**

**Report Submitted by:**

Saymon Acharya

Sangram Singh

Gurvinder Singh

Table of Contents

[Abstract/Executive Summary 3](#_Toc164837953)

[Data Preparation 3](#_Toc164837954)

[Define and prepare your class variables. 3](#_Toc164837955)

[Remove variables that are not needed/useful for the analysis. [5 marks] 3](#_Toc164837956)

[Describe the final dataset that is used for classification and include the scale/range for the new combined variables. 4](#_Toc164837957)

[Modelling 4](#_Toc164837958)

[Perform cluster analysis. 4](#_Toc164837959)

[Cluster the location for a crime of your choice. 4](#_Toc164837960)

[Cluster stopped people by reasons for stop. 8](#_Toc164837961)

[What else can you use cluster analysis for in the data set? 10](#_Toc164837962)

[How did you determine a suitable number of clusters for each method? 11](#_Toc164837963)

[Use internal validation measures to describe and compare the clusters. 12](#_Toc164837964)

[Evaluation 13](#_Toc164837965)

[Describe your results. What findings are the most interesting? How can these findings be used? 13](#_Toc164837966)

[Conclusion 14](#_Toc164837967)

# Abstract/Executive Summary

This report presents a detailed analysis of the Stop-Question-Frisk (SQF) initiative using advanced data preparation and cluster analysis methods to identify patterns of enforcement across New York City. By focusing on key class variables such as age, race, sex, and city, and removing irrelevant variables, we refined the dataset for more precise modeling. The study employed techniques like KMeans clustering to explore the geographical distribution and demographic specifics of police stops, revealing notable findings in terms of geographic variation and demographic targeting.

Key results indicate significant geographic disparities in the execution of SQF policies, suggesting that the intensity and frequency of stops vary considerably across different boroughs. This has implications for policy adjustments aimed at ensuring equitable law enforcement throughout the city. Additionally, the cluster analysis pointed to uneven application of frisking practices, with certain demographics disproportionately affected. This underscores an urgent need for policy reviews and reforms to address potential biases in stop-and-frisk operations.

These insights provide a robust basis for data-driven decision-making to enhance policing strategies. The findings can influence policy making, resource allocation, and police training programs, emphasizing the need for fairness and effectiveness in law enforcement practices. The report advocates for the continuous use of analytical models to refine SQF policies, ensuring they align with legal standards and community expectations. Overall, this analysis highlights the critical role of data analytics in identifying and rectifying disparities within law enforcement practices, aiming to foster a fairer and more balanced approach to policing in New York City.

# Data Preparation

## Define and prepare your class variables.

## Remove variables that are not needed/useful for the analysis. [5 marks]

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
* city

Irrelevant variables that we removed from the dataset for analysis:

* crimsusp
* ht\_feet
* ht\_inch
* searched
* weight
* pistol
* pf\_hcuff

## 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 screenshot of a table

Description automatically generated**

Figure 1: Final dataset for classification

# Modelling

## Perform cluster analysis.

## Cluster the location for a crime of your choice.

Performing cluster analysis on the given dataset. We used the below code to analyze the data and got the following outputs:

import pandas as pd

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

from sklearn.preprocessing import LabelEncoder, StandardScaler

# Loading the dataset

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

# Defining class variables

class\_variable = 'frisked'  # Example class variable

# Checking if the class variable exists in the dataframe

if class\_variable in data.columns:

    # Encoding categorical class variable using Label Encoding

    label\_encoder = LabelEncoder()

    data[class\_variable] = label\_encoder.fit\_transform(data[class\_variable])

else:

    raise KeyError(f"{class\_variable} is not a column in the dataframe")

# Removing variables that are not needed

columns\_to\_drop = ['datestop']  # Add other columns as necessary

data = data.drop(columns=columns\_to\_drop)

# Describing the final dataset

# For classification:

#   - Encoded categorical variables ('race', 'sex', 'city')

#   - Scaled numerical variable ('age')

# Scaling and Normalizing the numerical variable

numerical\_columns = ['age']  # Add other numerical columns as necessary

scaler = StandardScaler()

data[numerical\_columns] = scaler.fit\_transform(data[numerical\_columns])

# Splitting the dataset into training and testing sets (not mentioned in the tasks, but added for completeness)

X = data.drop(columns=[class\_variable])

y = data[class\_variable]

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

import pandas as pd

from sklearn.cluster import KMeans

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.preprocessing import StandardScaler, OneHotEncoder

# Loading the dataset

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

# One-hot encode the categorical variables, excluding 'city'.

categorical\_variables = ['race', 'sex', 'frisked', 'pistol', 'pf\_hcuff']

one\_hot\_encoder = OneHotEncoder()

encoded\_categorical = one\_hot\_encoder.fit\_transform(data[categorical\_variables]).toarray()

encoded\_feature\_names = one\_hot\_encoder.get\_feature\_names\_out(categorical\_variables)

encoded\_data = pd.DataFrame(encoded\_categorical, columns=encoded\_feature\_names)

# Normalizing 'age' since it's a numerical feature

scaler = StandardScaler()

data['scaled\_age'] = scaler.fit\_transform(data[['age']])

# Combining the one-hot encoded columns with the 'scaled\_age' column

combined\_data = pd.concat([data[['scaled\_age']], encoded\_data], axis=1)

# One-hot encode the 'city' column for clustering

city\_one\_hot = pd.get\_dummies(data['city'])

combined\_data\_with\_city = pd.concat([combined\_data, city\_one\_hot], axis=1)

# Performing clustering using only the city one-hot encoded features

kmeans = KMeans(n\_clusters=5, random\_state=42)

combined\_data\_with\_city['cluster'] = kmeans.fit\_predict(city\_one\_hot)

# Adding the cluster back to the original data to make sense of it

data['cluster'] = combined\_data\_with\_city['cluster']

# If you just want to see the distribution of clusters across different boroughs:

borough\_counts = data.groupby(['city', 'cluster']).size().unstack(fill\_value=0)

borough\_counts.plot(kind='bar', stacked=True)

plt.title('Cluster Distribution by Borough')

plt.xlabel('Borough')

plt.ylabel('Count')

plt.show()

**Outputs:**

**A graph with different colored bars

Description automatically generated**

Figure 2: Cluster Distribution by city "Borough"

**Second Code:**

# Performing one-hot encoding for categorical variables

data\_encoded = pd.get\_dummies(data)

# Clustering the location for a crime of your choice

crime\_location\_data = data\_encoded[['city\_BRONX', 'city\_MANHATTAN', 'city\_BROOKLYN']]

kmeans = KMeans(n\_clusters=3, random\_state=42)

crime\_location\_data['location\_cluster'] = kmeans.fit\_predict(crime\_location\_data)

# Visualizing the clusters

plt.figure(figsize=(10, 6))

sns.scatterplot(data=crime\_location\_data, x='city\_BRONX', y='city\_MANHATTAN', hue='location\_cluster', palette='viridis')

plt.title('Clustered Crime Locations')

plt.xlabel('BRONX')

plt.ylabel('MANHATTAN')

plt.show()

# Clustering stopped people by reasons for stop

stopped\_reason\_data = data\_encoded.drop(columns=['city\_BRONX', 'city\_MANHATTAN', 'city\_BROOKLYN'])  # Drop city columns for clustering

kmeans = KMeans(n\_clusters=3, random\_state=42)

stopped\_reason\_data['reason\_cluster'] = kmeans.fit\_predict(stopped\_reason\_data)

**Output:**

**A graph with numbers and lines

Description automatically generated**

Figure 4: Clustered Crime Locations

## Cluster stopped people by reasons for stop.

Most people were “Frisked” by the officers, so we created a code to visualize around that. The following shows the code used and its output:

import pandas as pd

from sklearn.cluster import KMeans

from sklearn.preprocessing import StandardScaler, OneHotEncoder

import matplotlib.pyplot as plt

import seaborn as sns

# Loading the dataset

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

# Encode 'frisked' as a binary variable for aggregation purposes

data['frisked'] = data['frisked'].map({'Y': 1, 'N': 0})

# One-hot encode the 'city' column for clustering

one\_hot\_encoder = OneHotEncoder(sparse=False)

city\_encoded = one\_hot\_encoder.fit\_transform(data[['city']])

city\_feature\_names = one\_hot\_encoder.get\_feature\_names\_out(['city'])

city\_encoded\_df = pd.DataFrame(city\_encoded, columns=city\_feature\_names)

# Perform clustering using only the city one-hot encoded features

kmeans = KMeans(n\_clusters=5, random\_state=42)

data['cluster'] = kmeans.fit\_predict(city\_encoded)

# Calculate mean age and frisked proportion by cluster and city

cluster\_stats = data.groupby(['city', 'cluster']).agg({

    'age': 'mean',

    'frisked': 'mean'

}).reset\_index()

# Pivot this data to have cities as columns, for easy plotting

cluster\_stats\_pivot = cluster\_stats.pivot(index='cluster', columns='city', values=['age', 'frisked'])

# Plotting the average age by cluster and city

plt.figure(figsize=(12, 6))

cluster\_stats\_pivot['age'].plot(kind='bar')

plt.title('Average Age by Cluster and City')

plt.xlabel('Cluster')

plt.ylabel('Average Age')

plt.legend(title='City')

plt.show()

# Plotting the proportion of frisked by cluster and city

plt.figure(figsize=(12, 6))

cluster\_stats\_pivot['frisked'].plot(kind='bar')

plt.title('Proportion Frisked by Cluster and City')

plt.xlabel('Cluster')

plt.ylabel('Proportion Frisked')

plt.legend(title='City')

plt.show()

**Output:**

A graph with different colored lines

Description automatically generated

Figure 3: Proportion Frisked by Cluster and City

## What else can you use cluster analysis for in the data set?

One potential use could be clustering individuals based on demographic and behavioral characteristics.

From the above code we also generated an additional cluster analysis for the given data. We made analysis on average age of the suspects by cluster and city and got the following output for the same:  
A graph of different colored lines

Description automatically generated

Figure 5: Average Age by Cluster and City

## How did you determine a suitable number of clusters for each method?

We utilized the python code below to determine number of clusters along with visualization:

# Using the elbow method to determine the optimal number of clusters

inertia = []

for n\_clusters in range(1, 11):

    kmeans = KMeans(n\_clusters=n\_clusters, random\_state=42)

    kmeans.fit(stopped\_reason\_data)

    inertia.append(kmeans.inertia\_)

# Visualize the elbow curve

plt.figure(figsize=(10, 6))

plt.plot(range(1, 11), inertia, marker='o', linestyle='--')

plt.xlabel('Number of Clusters')

plt.ylabel('Inertia')

plt.title('Elbow Method for Optimal Number of Clusters')

plt.show()

# Use internal validation measures to describe and compare the clusters

# Silhouette Score

from sklearn.metrics import silhouette\_score

silhouette\_avg = silhouette\_score(stopped\_reason\_data, stopped\_reason\_data['reason\_cluster'])

print(f"Silhouette Score: {silhouette\_avg}")

# Adjusted Rand Index

from sklearn.metrics import adjusted\_rand\_score

adjusted\_rand\_idx = adjusted\_rand\_score(stopped\_reason\_data[class\_variable], stopped\_reason\_data['reason\_cluster'])

print(f"Adjusted Rand Index: {adjusted\_rand\_idx}")

**Output:**

**A graph of a number of clusters

Description automatically generated**

**Silhouette Score:** 0.3445526960026389

**Adjusted Rand Index:** 0.0018496679629946603

## Use internal validation measures to describe and compare the clusters.

In the SQF dataset cluster analysis, the Silhouette Score and the Adjusted Rand Index were used as internal validation measures. A Silhouette Score of 0.3445 suggests a moderate distinction between clusters, while an Adjusted Rand Index of 0.0018 shows minimal correspondence with the actual class labels, indicating that the clustering does not strongly align with predefined categories.

The number of clusters was determined using the elbow method, which did not show a clear 'elbow,' reflecting the complex nature of the data. These validation measures offer insights into the clusters' quality, aiding in the evaluation of clustering as a tool for informing NYPD policy decisions.

# 

# Evaluation

Describe your results. What findings are the most interesting? How can these findings be used?  
The results from the analysis of the Stop-Question-Frisk (SQF) data provide valuable insights into the dynamics of police stops across New York City. The cluster analysis, particularly on crime location and reasons for stops, reveals two main findings:

1. **Geographic Variation**: The clustering of crime locations by borough shows significant variation, which suggests that policing strategies and the frequency of stops vary considerably across different areas. This geographic disparity can inform targeted interventions and policy adjustments to ensure equitable law enforcement across the city.
2. **Demographics and Stop Reasons**: Clustering based on demographic variables and reasons for stops highlights that frisking practices are not uniformly applied. The analysis of 'Frisked' incidents across clusters indicates a disproportionate focus on certain demographics, underscoring the need for continued scrutiny and reform in stop-and-frisk practices to avoid biases.

These findings can be leveraged in several ways:

* **Policy Making**: By understanding the patterns and locations where stops are more prevalent or where discriminatory practices might be occurring, policymakers can tailor reforms to address these specific issues.
* **Resource Allocation**: Insights from the cluster analysis can help allocate resources more effectively by identifying high-need areas that require more policing or different types of community engagement strategies.
* **Training and Education**: Data on how stops are conducted can inform training programs for police officers, emphasizing fair practices and the importance of reducing bias in stop-and-frisk operations.

Overall, these results underscore the complexity of policing in a diverse city and the critical need for data-driven approaches to enhance transparency, fairness, and effectiveness in law enforcement.

# Conclusion

This analysis of New York City's Stop-Question-Frisk (SQF) initiative highlights important differences in stop patterns across various boroughs and a noticeable focus on specific demographics. Our clustering techniques reveal a moderate distinction between groups, indicated by a Silhouette Score of 0.3445, suggesting that while frisking practices vary, they aren’t sharply separated. Additionally, a very low Adjusted Rand Index suggests these groups don't closely match traditional categories, pointing to a chance for the NYPD to improve their strategies based on these insights.