TERRY-STOPS PROJECT

Data Understanding

In [1]:
```
import pandas as pd
df = pd.read_csv('Terry_Stops.csv')
df.head()
```

Out[1]:

	Subject Age Group	Subject ID	GO / SC Num	Terry Stop ID	Stop Resolution	Weapon Type	Officer ID	Officer YOB
0	36 - 45	8597903457	20190000243853	8597867579	Field Contact	-	5469	1967
1	26 - 35	-1	20160000003391	181276	Field Contact	None	7591	1985
2	18 - 25	7774286580	20210000118915	24056783769	Field Contact	-	7459	1973
3	26 - 35	-1	20180000047173	400437	Offense Report	None	6680	1972
4	18 - 25	-1	20160000085711	136669	Offense Report	None	7560	1986

5 rows × 23 columns

In [2]:
```
df.shape
```

Out[2]:
```
(58157, 23)
```

In [3]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 58157 entries, 0 to 58156
Data columns (total 23 columns):
 #   Column                    Non-Null Count  Dtype 
---  ------                    --------------  ----- 
 0   Subject Age Group         58157 non-null  object
 1   Subject ID                58157 non-null  int64 
 2   GO / SC Num               58157 non-null  int64 
 3   Terry Stop ID             58157 non-null  int64 
 4   Stop Resolution           58157 non-null  object
 5   Weapon Type               58157 non-null  object
 6   Officer ID                58157 non-null  object
 7   Officer YOB               58157 non-null  int64 
 8   Officer Gender            58157 non-null  object
 9   Officer Race              58157 non-null  object
 10  Subject Perceived Race    58157 non-null  object
 11  Subject Perceived Gender  58157 non-null  object
 12  Reported Date             58157 non-null  object
 13  Reported Time             58157 non-null  object
 14  Initial Call Type         58157 non-null  object
 15  Final Call Type           58157 non-null  object
 16  Call Type                 58157 non-null  object
 17  Officer Squad             57613 non-null  object
 18  Arrest Flag               58157 non-null  object
 19  Frisk Flag                58157 non-null  object
 20  Precinct                  58157 non-null  object
 21  Sector                    58157 non-null  object
 22  Beat                      58157 non-null  object
dtypes: int64(4), object(19)
memory usage: 10.2+ MB
```

In [4]:
```
df.isna().sum()
```

Out[4]:
```
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              544
Arrest Flag                  0
Frisk Flag                   0
Precinct                     0
Sector                       0
Beat                         0
dtype: int64
```

Data Cleaning

In [5]:
```
df.dropna(inplace=True)
df.isna().sum()
```

Out[5]:
```
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
```

In [6]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 57613 entries, 1 to 58156
Data columns (total 23 columns):
 #   Column                    Non-Null Count  Dtype 
---  ------                    --------------  ----- 
 0   Subject Age Group         57613 non-null  object
 1   Subject ID                57613 non-null  int64 
 2   GO / SC Num               57613 non-null  int64 
 3   Terry Stop ID             57613 non-null  int64 
 4   Stop Resolution           57613 non-null  object
 5   Weapon Type               57613 non-null  object
 6   Officer ID                57613 non-null  object
 7   Officer YOB               57613 non-null  int64 
 8   Officer Gender            57613 non-null  object
 9   Officer Race              57613 non-null  object
 10  Subject Perceived Race    57613 non-null  object
 11  Subject Perceived Gender  57613 non-null  object
 12  Reported Date             57613 non-null  object
 13  Reported Time             57613 non-null  object
 14  Initial Call Type         57613 non-null  object
 15  Final Call Type           57613 non-null  object
 16  Call Type                 57613 non-null  object
 17  Officer Squad             57613 non-null  object
 18  Arrest Flag               57613 non-null  object
 19  Frisk Flag                57613 non-null  object
 20  Precinct                  57613 non-null  object
 21  Sector                    57613 non-null  object
 22  Beat                      57613 non-null  object
dtypes: int64(4), object(19)
memory usage: 10.5+ MB
```

In [7]:
```
df.drop_duplicates()
```

Out[7]:

	Subject Age Group	Subject ID	GO / SC Num	Terry Stop ID	Stop Resolution	Weapon Type	Officer ID	C
1	26 - 35	-1	20160000003391	181276	Field Contact	None	7591	
2	18 - 25	7774286580	20210000118915	24056783769	Field Contact	-	7459	
3	26 - 35	-1	20180000047173	400437	Offense Report	None	6680	
4	18 - 25	-1	20160000085711	136669	Offense Report	None	7560	
5	26 - 35	7726413042	20200000137557	13081774163	Arrest	-	7662	
...	...	...	...	...	...	...	...	...
58152	18 - 25	-1	20160000130978	146528	Arrest	None	7463	
58153	18 - 25	-1	20160000426909	215171	Arrest	Lethal Cutting Instrument	8409	
58154	36 - 45	7727655354	20190000388430	11143903619	Field Contact	-	8624	
58155	36 - 45	7742492032	20220000335270	38989212363	Field Contact	-	8877	
58156	36 - 45	-1	20150000355899	91064	Offense Report	None	6810	

57613 rows × 23 columns

Exploratory Data Analysis

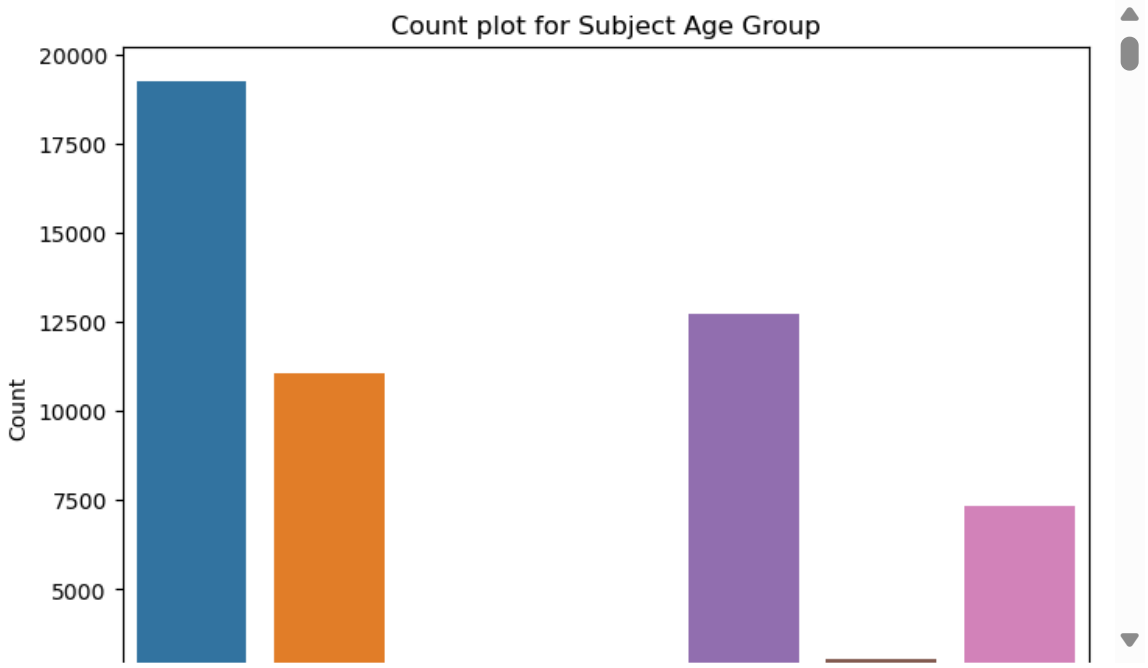
In [24]:
```
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
%matplotlib inline
```

Univariate Analysis

In [25]:
```
categorical_columns =df.select_dtypes(include=['object', 'category']).colum
print(categorical_columns)
```

```
['Subject Age Group', 'Stop Resolution', 'Weapon Type', 'Officer ID', 'Off
icer Gender', 'Officer Race', 'Subject Perceived Race', 'Subject Perceived
Gender', 'Reported Date', 'Reported Time', 'Initial Call Type', 'Final Cal
l Type', 'Call Type', 'Officer Squad', 'Arrest Flag', 'Frisk Flag', 'Preci
nct', 'Sector', 'Beat']
```

In [10]:
```
for col in categorical_columns :
    plt.figure(figsize=(8, 6))
    sns.countplot(x=col , data=df)
    plt.title(f'Count plot for {col}')
    plt.ylabel('Count')
    plt.xticks(rotation=45)
    plt.show()
```



Inference

- Subject age group is mostly 26-35 and 36-45
- Stop resolution shows more field contact which could potentially show how the police suspect people
- The male gender is doing more of the arresting and being arrested

Bivariate Analysis

In [11]:
```
df["Arrest Flag"].value_counts().to_frame()
```

Out[11]:

	Arrest Flag
N	51615
Y	5998

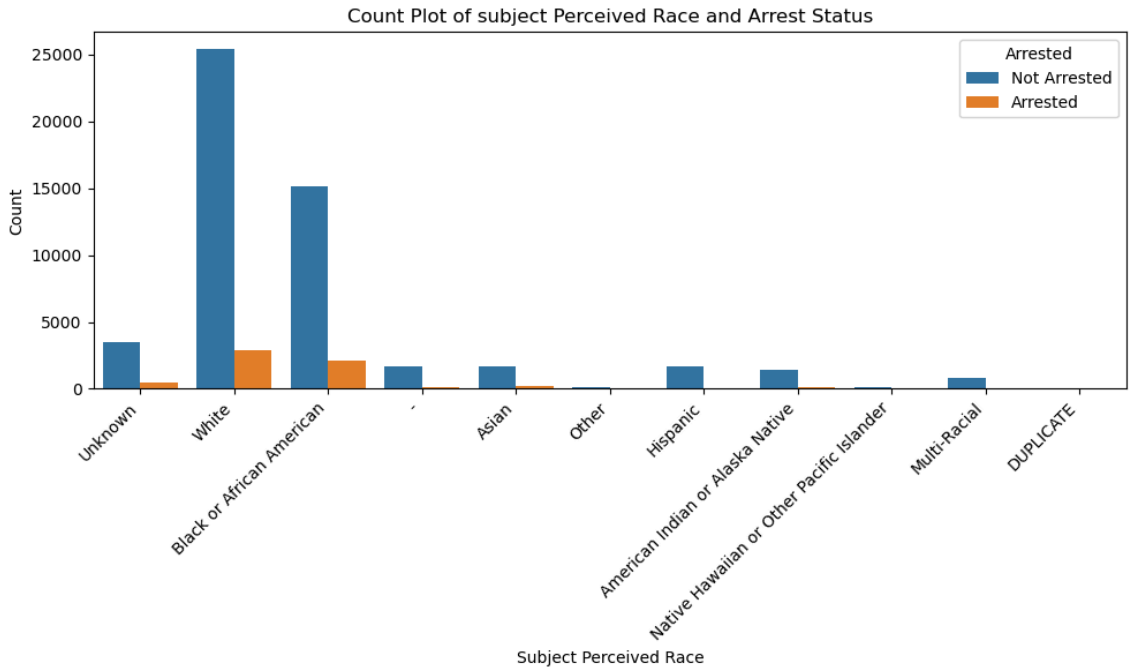
In [12]:
```
df_encoded = pd.get_dummies(df["Arrest Flag"])
print(df_encoded)
```

```
       N  Y
1      1  0
2      1  0
3      1  0
4      1  0
5      0  1
...   .. ..
58152  1  0
58153  1  0
58154  1  0
58155  1  0
58156  1  0

[57613 rows x 2 columns]
```

In [13]:
```
df['Arrested'] = df_encoded['Y']
```

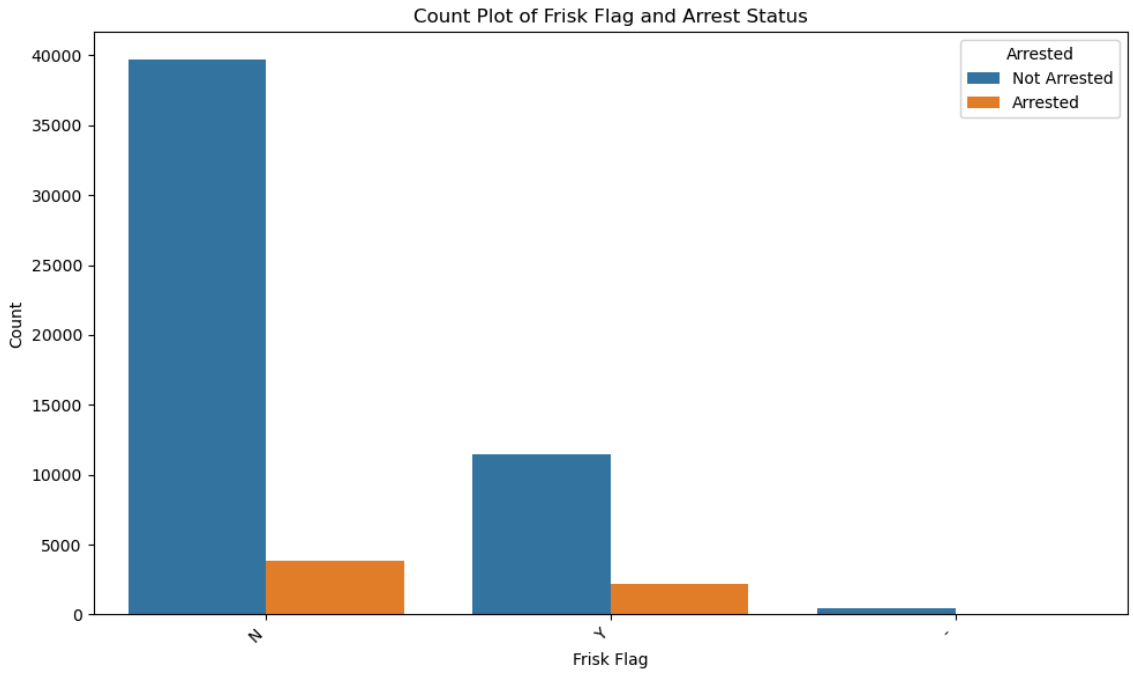
In [14]:
```
plt.figure(figsize=(10, 6))
sns.countplot(x='Subject Perceived Race', hue='Arrested', data=df)
plt.title('Count Plot of subject Perceived Race and Arrest Status')
plt.xlabel('Subject Perceived Race')
plt.ylabel('Count')
plt.legend(title='Arrested',loc='upper right', labels=['Not Arrested', 'Arr
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show
```

Out[14]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```



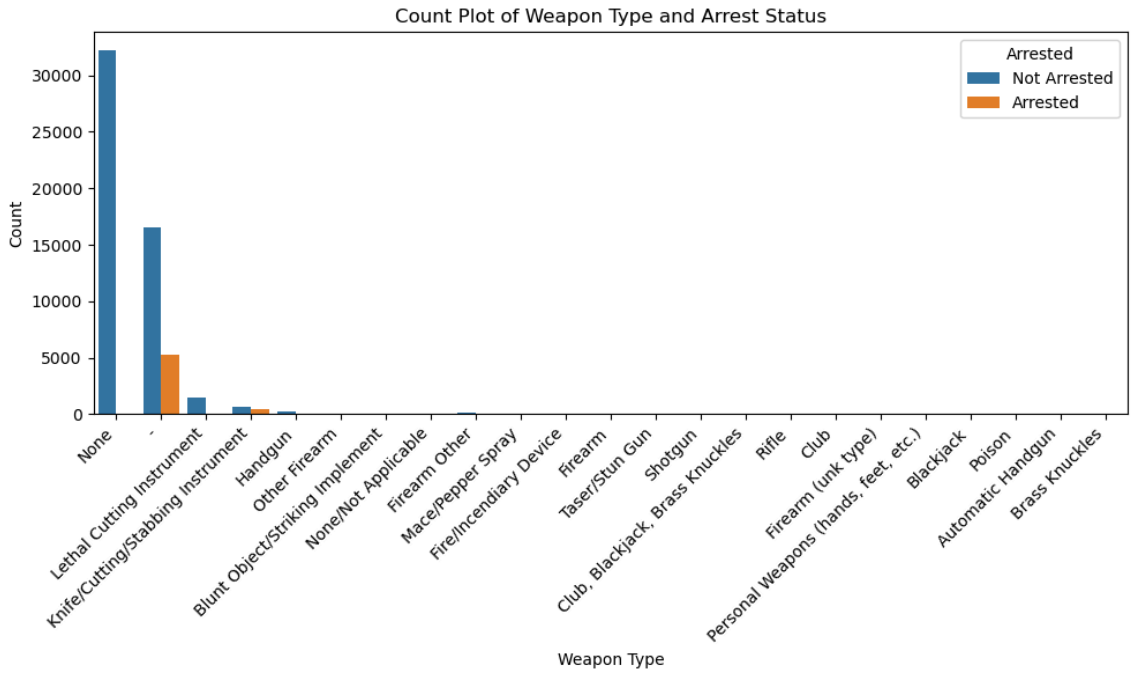
In [15]:
```
plt.figure(figsize=(10, 6))
sns.countplot(x='Frisk Flag', hue='Arrested', data=df)
plt.title('Count Plot of Frisk Flag and Arrest Status')
plt.xlabel('Frisk Flag')
plt.ylabel('Count')
plt.legend(title='Arrested',loc='upper right', labels=['Not Arrested', 'Arr
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show
```

Out[15]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```



In [16]:
```
plt.figure(figsize=(10, 6))
sns.countplot(x='Weapon Type', hue='Arrested', data=df)
plt.title('Count Plot of Weapon Type and Arrest Status')
plt.xlabel('Weapon Type')
plt.ylabel('Count')
plt.legend(title='Arrested',loc='upper right', labels=['Not Arrested', 'Arr
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show
```

Out[16]: <function matplotlib.pyplot.show(close=None, block=None)>



In [ ]:
```
```

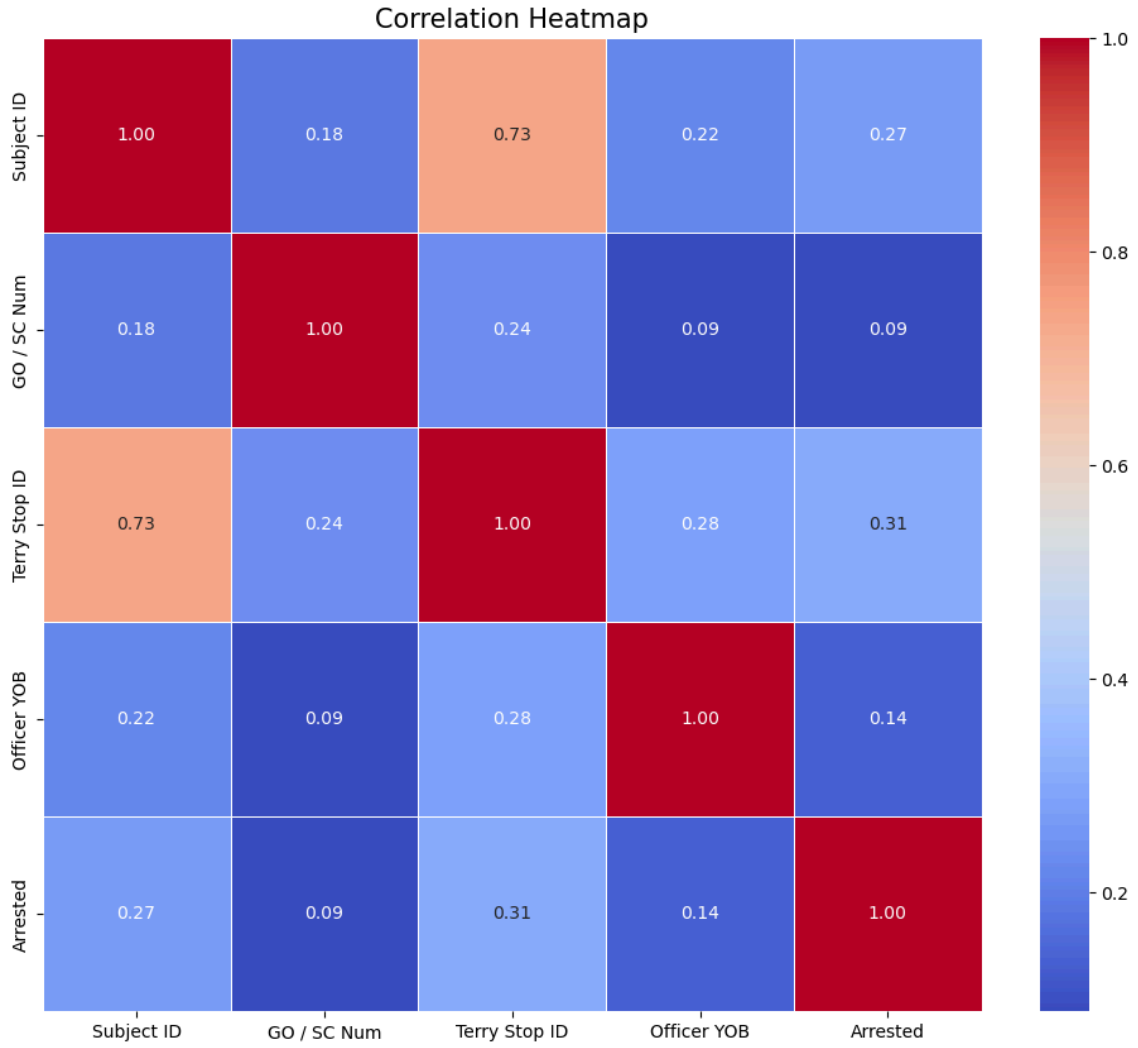
Multivariate Analysis

In [17]:
```
correlation_matrix = df.corr()
corr_arrested = correlation_matrix['Arrested']
corr_arrested

print(corr_arrested)
```

```
Subject ID       0.272734
GO / SC Num      0.088556
Terry Stop ID    0.305715
Officer YOB      0.137744
Arrested         1.000000
Name: Arrested, dtype: float64

C:\Users\admin\AppData\Local\Temp\ipykernel_26472\2152097093.py:1: FutureW
arning: The default value of numeric_only in DataFrame.corr is deprecated.
In a future version, it will default to False. Select only valid columns o
r specify the value of numeric_only to silence this warning.
  correlation_matrix = df.corr()
```

In [18]:

```
plt.figure(figsize=(12, 10))
sns.heatmap(correlation_matrix, annot=True, cmap="coolwarm", fmt=".2f", lin
plt.title('Correlation Heatmap', size=15)
plt.show()
```



In [ ]:

Multivariate Analysis

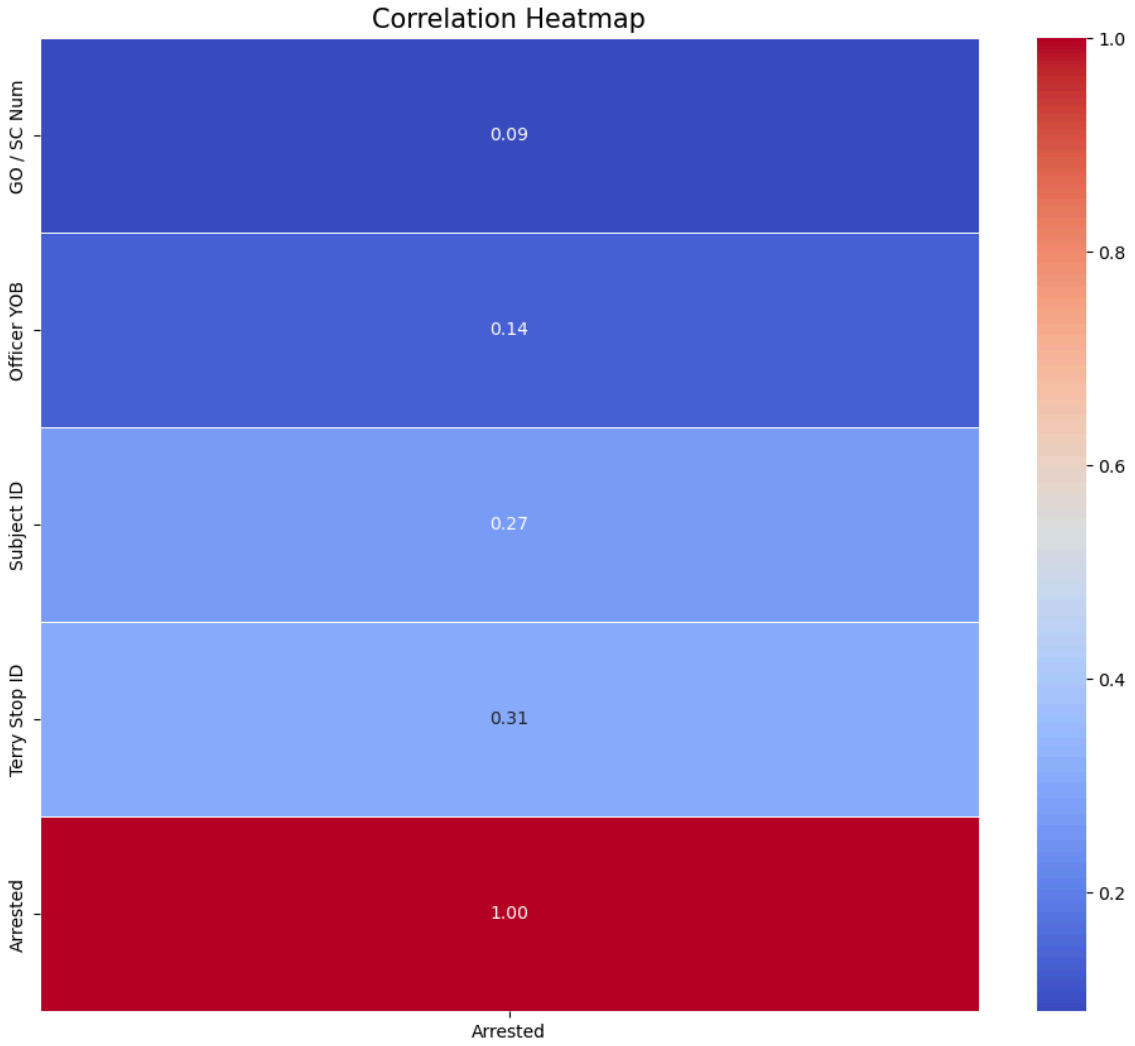
In [19]:
```
correlation_with_arrested = df.corr()['Arrested'].sort_values().to_frame()
correlation_with_arrested
```

```
C:\Users\admin\AppData\Local\Temp\ipykernel_26472\1640500724.py:1: FutureW
arning: The default value of numeric_only in DataFrame.corr is deprecated.
In a future version, it will default to False. Select only valid columns o
r specify the value of numeric_only to silence this warning.
  correlation_with_arrested = df.corr()['Arrested'].sort_values().to_frame
()
```

Out[19]:

	Arrested
GO / SC Num	0.088556
Officer YOB	0.137744
Subject ID	0.272734
Terry Stop ID	0.305715
Arrested	1.000000

In [20]:
```
plt.figure(figsize=(12, 10))
sns.heatmap(correlation_with_arrested, annot=True, cmap="coolwarm", fmt=".2
plt.title('Correlation Heatmap', size=15)
plt.show()
```



In [ ]:
```
```

Modelling

Logistic Regression

In [21]:
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import OneHotEncoder
from sklearn.metrics import accuracy_score, f1_score
from sklearn.metrics import roc_curve, auc
```

In [22]:

```
X = df[['Subject ID', 'Officer Race', 'Subject Age Group', 'Subject Perceiv
y = df['Arrested']

encoder = OneHotEncoder()
X_encoded = encoder.fit_transform(X)

X_train, X_test, y_train, y_test = train_test_split(X_encoded, y, test_size

model = LogisticRegression()
model.fit(X_train, y_train)

y_pred = model.predict(X_test)

print(f'Accuracy : {accuracy_score(y_test, y_pred)}')
print(f'F1 score : {f1_score(y_test, y_pred)}')
```

```
Accuracy : 0.9238045647834765
F1 score : 0.613556338028169
```

In [ ]:

The Class Imbalance Problem

In [23]:

```
from imblearn.over_sampling import SMOTE

smote = SMOTE(sampling_strategy='auto')
X_train_resampled, y_train_resampled = smote.fit_resample(X_train, y_train)


model_log = model.fit(X_train_resampled, y_train_resampled)

y_pred = model_log.predict(X_test)

print(f'Accuracy : {accuracy_score(y_test, y_pred)}')
```

```
---------------------------------------------------------------------------
ImportError                               Traceback (most recent call last)
Cell In[23], line 1
----> 1 from imblearn.over_sampling import SMOTE
      3 smote = SMOTE(sampling_strategy='auto')
      4 X_train_resampled, y_train_resampled = smote.fit_resample(X_train, y_train)

File ~\anaconda3\Lib\site-packages\imblearn\__init__.py:52
     48     sys.stderr.write("Partial import of imblearn during the build process.\n")
     49     # We are not importing the rest of scikit-learn during the build
     50     # process, as it may not be compiled yet
     51 else:
---> 52     from . import (
     53         combine,
     54         ensemble,
     55         exceptions,
     56         metrics,
     57         over_sampling,
     58         pipeline,
     59         tensorflow,
     60         under_sampling,
     61         utils,
     62     )
     63     from ._version import __version__
     64     from .base import FunctionSampler

File ~\anaconda3\Lib\site-packages\imblearn\combine\__init__.py:5
      1 """The :mod:`imblearn.combine` provides methods which combine
      2 over-sampling and under-sampling.
      3 """
----> 5 from ._smote_enn import SMOTEENN
      6 from ._smote_tomek import SMOTETomek
      8 __all__ = ["SMOTEENN", "SMOTETomek"]

File ~\anaconda3\Lib\site-packages\imblearn\combine\_smote_enn.py:13
     10 from sklearn.utils import check_X_y
     12 from ..base import BaseSampler
---> 13 from ..over_sampling import SMOTE
     14 from ..over_sampling.base import BaseOverSampler
     15 from ..under_sampling import EditedNearestNeighbours

File ~\anaconda3\Lib\site-packages\imblearn\over_sampling\__init__.py:8
      6 from ._adasyn import ADASYN
      7 from ._random_over_sampler import RandomOverSampler
----> 8 from ._smote import SMOTE, SMOTEN, SMOTENC, SVMSMOTE, BorderlineSMOTE, KMeansSMOTE
     10 __all__ = [
     11     "ADASYN",
     12     "RandomOverSampler",
   (...)
     18     "SMOTEN",
     19 ]
```

```
File ~\anaconda3\Lib\site-packages\imblearn\over_sampling\_smote\__init__.
py:1
----> 1 from .base import SMOTE, SMOTEN, SMOTENC
      2 from .cluster import KMeansSMOTE
      3 from .filter import SVMSMOTE, BorderlineSMOTE

File ~\anaconda3\Lib\site-packages\imblearn\over_sampling\_smote\base.py:1
8
     16 from sklearn.exceptions import DataConversionWarning
     17 from sklearn.preprocessing import OneHotEncoder, OrdinalEncoder
---> 18 from sklearn.utils import (
     19     _get_column_indices,
     20     _safe_indexing,
     21     check_array,
     22     check_random_state,
     23 )
     24 from sklearn.utils.sparsefuncs_fast import (
     25     csr_mean_variance_axis0,
     26 )
     27 from sklearn.utils.validation import _num_features

ImportError: cannot import name '_get_column_indices' from 'sklearn.utils'
(C:\Users\admin\anaconda3\Lib\site-packages\sklearn\utils\__init__.py)
```

Because of the import error we will continue with the classes as they were making sure they will not overfit

In [ ]:

In [26]:
```
from sklearn.tree import DecisionTreeClassifier

dt = DecisionTreeClassifier()

dt.fit(X_train, y_train)

dt_train_score = dt.score(X_train, y_train)

dt_train_score
```

Out[26]: 0.9988934692991972

The Decision tree overfits the data giving us 99% score

In [27]:
```
from sklearn.model_selection import cross_val_score, GridSearchCV

dt_cv = cross_val_score(dt, X_train, y_train, cv=5)
dt_cv
```

Out[27]: array([0.91418963, 0.9222174 , 0.91592536, 0.91711868, 0.92004773])

.

In [28]:
```
from sklearn.ensemble import RandomForestClassifier

random_forest = RandomForestClassifier(random_state=42)

random_forest.fit(X_train, y_train)
y_pred = random_forest.predict(X_test)

print(f'Accuracy : {accuracy_score(y_test, y_pred)}')
print(f'F1 score : {f1_score(y_test, y_pred)}')
```

```
Accuracy : 0.9136509589516619
F1 score : 0.4687666844634277
```

In [29]:
```
forest_cv_scores = cross_val_score(random_forest, X_train, y_train, cv=5)
forest_cv_scores
```

Out[29]:
```
array([0.91017574, 0.91516598, 0.91288783, 0.91559991, 0.91429811])
```

In [34]:
```
param_grid = {
    'n_estimators': [5, 10, 15],
    'max_depth': [None, 5, 10],
    'min_samples_split': [2, 5],
    'min_samples_leaf': [1, 2]
}

grid_search = GridSearchCV(estimator=random_forest, param_grid=param_grid,
grid_search.fit(X_train, y_train)

print(f'Best parameters: {grid_search.best_params_}')
print(f'Best parameters: {grid_search.best_score_}')
```

```
Best parameters: {'max_depth': None, 'min_samples_leaf': 1, 'min_samples_s
plit': 2, 'n_estimators': 15}
Best parameters: 0.9121501410284226
```

The Grid search takes very long to come up with results thus going with Randomized search.

In [37]:
```
from sklearn.model_selection import RandomizedSearchCV

random_search = RandomizedSearchCV(estimator=random_forest, param_distribut
random_search.fit(X_train, y_train)

print(f'Best parameters: {random_search.best_params_}')
print(f'Best score: {random_search.best_score_}')
```

```
Best parameters: {'n_estimators': 10, 'min_samples_split': 5, 'min_samples
_leaf': 1, 'max_depth': None}
Best score: 0.9110870036884358
```

In [32]:
```
rf_best = grid_search.best_estimator_
rf_best.fit(X_train, y_train)

best_y_pred = rf_best.predict(X_test)

best_model = accuracy_score(y_test, best_y_pred)
print(f'Accuracy : {accuracy_score(y_test, best_y_pred)}')
print(f'F1 score : {f1_score(y_test, best_y_pred)}')
```

```
Accuracy : 0.9140848737307993
F1 score : 0.48544698544698545
```

.

In [36]:
```
from xgboost import XGBClassifier
xgb_model = XGBClassifier()

xgb_model.fit(X_train, y_train)

xgb_y_pred = xgb_model.predict(X_test)

print(f'Accuracy : {accuracy_score(y_test, xgb_y_pred)}')
print(f'F1 score : {f1_score(y_test, xgb_y_pred)}')
```

```
Accuracy : 0.9209407272411698
F1 score : 0.5842081241442264
```

In [38]:
```
xgb_cv = cross_val_score(xgb_model, X_train, y_train, cv=5)
xgb_cv
```

Out[38]:
```
array([0.9144066 , 0.92373617, 0.92048167, 0.91928835, 0.92243437])
```

In [41]:
```
from sklearn.model_selection import RandomizedSearchCV

random_search = RandomizedSearchCV(estimator=xgb_model, param_distributions
random_search.fit(X_train, y_train)

print(f'Best parameters: {random_search.best_params_}')
print(f'Best score: {random_search.best_score_}')
```

```
Best parameters: {'subsample': 0.8, 'n_estimators': 15, 'max_depth': 7, 'l
earning_rate': 0.2, 'colsample_bytree': 1.0}
Best score: 0.9141462356259492
```