**GROUP 03- Ziyan Liu, Melissa Dennis, Seung Hyeon Ban, Harriet Odhiambo, Sona Sheena Biju**

**Executive Summary**

*OUR GOAL:* After deploying our model and running the front end, we can predict the status of a bicycle (whether it will be 'Stolen' or 'Recovered') based on certain input features. These features include 'PRIMARY\_OFFENCE', 'BIKE\_COST', 'REPORT\_DOY', 'BIKE\_MAKE', and 'BIKE\_MODEL'. When a user inputs these details into our form, the Flask API will use the deployed model to predict and return the bicycle's status.

*PROBLEM*: The imbalanced data distribution greatly affected the model’s ability to predict

*SOLUTION:* We used the SMOTE technique to balance the dataset. We could have used cross-validation to better improve the performance of our model.

*KEY FINDINGS:* By using Random Forest Model, the overall accuracy of the model is 97%.

**Data exploration and findings**

A screenshot of a computer

Description automatically generated

A graph of a bike making

Description automatically generatedA graph of blue bars

Description automatically generatedA map with a red and yellow dot

Description automatically generated with medium confidence

**Feature selection**

Feature selection plays a critical role in refining the dataset for effective model training and analysis. This process involves various techniques and tools to determine the most relevant features that significantly impact the model's performance. Initially, a thorough correlation analysis is performed on the numerical data, excluding the target variable 'BIKE\_COST'. This analysis aims to identify features with a strong correlation to the target variable, thereby indicating their importance in predictive modeling. Features with a correlation higher than a set threshold (0.3 in this case) are selected for further processing.

Additionally, categorical variables like 'PRIMARY\_OFFENCE', 'LOCATION\_TYPE', and 'PREMISES\_TYPE' are encoded using one-hot encoding. This transformation is crucial as it converts categorical data into a numerical format, enabling the logistic regression model to process and learn from these features effectively. The final selection combines both highly correlated numerical features and encoded categorical variables, providing a comprehensive dataset tailored for the predictive model.

The combination of correlation analysis and categorical encoding ensures a balanced and insightful selection of features. By focusing on variables with significant correlation and properly handling categorical data, the model is equipped with a robust and relevant dataset. This selection process aims to enhance the model's accuracy and reliability, demonstrating the importance of thoughtful feature selection in data science and machine learning projects.

**Data modeling**

**Missing Values Handling:**

* Filled missing numerical values with their respective column means.
* Imputed missing categorical values with the mode (most frequent value) of each column.

**Data Transformation:**

* Encoded categorical columns using ***Label Encoding***.
* Standardized numerical columns using ***StandardScaler*** to maintain consistency and facilitate model training.

**Data Wrangling Techniques:**

* Utilized SimpleImputer to handle missing values based on mean and mode for numerical and categorical columns, respectively.

***RESULT AFTER DATA CLEANING***

A screenshot of a computer

Description automatically generated

**Assumptions:**

* Imputation: Assumes imputing missing values with mean/mode maintains data integrity without significantly skewing distributions.

**Model building**

**Train-Test Split and Sampling:**

* Splitting Data: Divided the dataset into training and testing sets using train\_test\_split() with a test size of 20%.
* Handling Class Imbalance: Checked for class imbalance in the target variable (y\_train).
* Balanced the training set using SMOTE to address class imbalance.

A computer screen shot of numbers and letters

Description automatically generated

**Model Selection and Training:**

* Logistic Regression

A screenshot of a computer program

Description automatically generated

* Decision Tree.

A screenshot of a computer screen

Description automatically generated