Datacentric Insights for Enhancing Law Enforcement: Analysing Burglary Patterns in Los Angeles

Abstract: This project will examine trends for crime in Los Angeles crime dataset using data visualisation techniques and machine learning models. The analysis showed the fluctuation of crime patterns during temporal intervals, highlighting peaks and pinpointed crime hotspots in the city offering valuable insights. One of the results indicated that burglaries are more common late at night while theft and assault show higher rates during the afternoon. By utilising machine learning methods, specifically logistic regression, and random forest classifier we were able to predict resolved and unresolved burglary's rates based on previous data with over high accuracy for the majority cases. These findings aim to give stakeholders data-driven insights that can be implemented to combat crime and enhance safety in Los Angeles effectively, thereby improving policing tactics, optimising resource allocation, or implementing targeted strategies in high-crime areas, effectively fighting crime, and improving safety in Los Angeles.

Keywords: Public Safety, Crime Data Analysis, Visualisation Tools, Exploratory Data Analysis, Machine Learning

Contents

1 Introduction

When implementing laws or simply understanding the characteristics of the crime, it is essential to consider the factors by which the crime is most influenced; these can range from the most targeted victim group to the location most impacted by a given crime. By looking at trends and visualising the data, this project will aim to provide policymakers insights.

This project examines the patterns of activity, in Los Angeles from 2020 to the present using a dataset to uncover practical insights that can improve law enforcement tactics allocate resources more effectively and enhance public safety. The research focuses on regions within Los Angeles to pinpoint trends in crime and high-risk areas assisting in precise and efficient law enforcement actions. It aids in policing and strategic resource distribution by revealing when and where crimes are most probable. The analysis was carried out using Python 3.6+ in a Jupyter Notebook setting utilizing tools like pandas for data handling for visualisation and scikit learn, for machine learning (McKinney, 2017; Hunter, 2007; Pedregosa et al., 2011). Current services generally offer crime mapping tools and statistical reporting systems. This initiative seeks to elevate these services by incorporating analytics and comprehensive visual representations.

2 Dataset Description

2.1 Overview of the Dataset

The dataset comprises over 897,106 instances and 29 features, detailing crime incidents reported in Los Angeles. It provides a comprehensive view of various aspects of crime occurrences, including temporal, spatial, and demographic dimensions (Los Angeles Open Data, 2024). This extensive dataset allows for in-depth analysis and the derivation of actionable insights to enhance public safety and law enforcement strategies.

The initial analysis revealed that the dataset was rich in temporal and demographic details (Appendix 1) which made it highly especially for identifying patterns and trends over time and also across different areas. For example, certain crimes might peak at specific times of the day, year or be more prevalent in particular areas. This in turn provides valuable insights for targeted interventions.

2.2 Dataset's key fields and their definitions

Field in dataset	Definition	Data Type
AREA NAME	Cardinal direction on where the crime occurred	object
AREA	Area code for where the crime was committed	Int64
Crm Cd	Crime code for the specific crime committed	Int64
Crm Cd Desc	Type of crime committed e.g. burglary	object
Vict Age	Age of the victim	Int64
Vict Sex	Gender of the victim, this includes Male, Female, Hermaphrodite or Other	object
Vict Descent	Ethnicity of the victim such as Black, White, Hispanic etc.	object
Time OCC	The time of the day when the time occurred	Int64
Premise Cd	Code of the location	Float64
Premis Desc	Location of the crime e.g. street	object
Date Rptd	Date on which the crime was reported	object
DATE OCC	Date on which the crime occurred	object

Status	Status of the case, IC being the default	object
Status Desc	If the case is still ongoing	object
LON	Longitude of where the crime occurred	Float64
LAT	Latitude of where the crime occurred	Float64
Weapon Desc	Description of the weapon used	object

2.3 Descriptive Statistics and Visualisations

Visualisations were created in order to derive ideas and insights from the dataset. One of them included the demographic breakdown of victims in descent and sex (Appendix 5), this allowed us to see if there are any patterns which could be derived from this information. Two other visualisations involved looking at the burglary frequency by the days of the week and by distribution by the hour of the day (Appendix 10). Finally, we looked at the four burglary crimes and visualised them on a boxplot in order to compare them (Appendix 13). This gave us an insight into the distribution of those crimes as we noted that successful burglaries had almost the same count and heavily outweighed the attempted burglaries and attempted burglaries from vehicles.

2.4 Cleaning the data

Before exploratory data analysis the data was first cleaned, and the following steps were undertaken:

Trailing spaces found in names were removed and a function was created to change any names to a correct format in order to avoid errors while processing data (Appendix 2.1).

71 duplicate variables were identified in the specified rows relating to burglary and later removed, the same was done with columns unrelated to burglary as they were irrelevant to the study (Appendix 2.4, 2.5 and 2.6).

We have categorised the day into 4 stages, these include late night, morning, afternoon, and evening in order to identify the pattern of the crime and the frequency of crime across this segment of the day (Appendix 18). We did the same thing with months, categorising them into seasons, in in turn makes it more clear for the stakeholder.

Columns such as Weapon Desc (weapon descriptions) were dropped (Appendix 2.7). This was due to the lack of data in relation to burglary related crimes, 101788 out of 112342 cases they were not documented, which translates to roughly 90% of the cases lacking data. Continuing could potentially create inaccuracies or biases in analyses. Similarly, Weapon Used Cd (type of weapon used) was also dropped as by result it made the majority of the variables invalid, with 8233 out of 10554 instances being unknown. (Müller & Guido, 2016).

These steps ensured the dataset was accurate, ready for an in-depth analysis, and able to facilitate meaningful insights.

3 Exploratory Data Analysis

3.1 Purpose

EDA was conducted to understand the basic structure and patterns within the dataset, utilising summary statistics and visualisations. The primary goal was to uncover any underlying trends, identify anomalies and find possible relationships between variables for model development.

3.2 Steps and Techniques Used

One of the techniques that was used involved filtering the dataset for 4 type of burglaries, these being: burglary, attempted burglary, vehicle burglary and attempted burglary (Appendix 2.7).

After the visualisation of the victim age, rows where a number was missing or less than 0 were filtered out after plotting a histogram (Appendix 3).

Filtering aggregating was also performed on visualising burglaries using the folium library (Appendix 9) in order to create a map, this map allowed for a visualisation of the density or counts of burglaries (from high crime count being counted as red and green as low).

The analysis of box plots reveals that the typical age range, for individuals engaged in burglary related offenses tends to fall between 30 and 40 years. Specifically, the median age for "Burglary From Vehicle" is lower at 30 to 35 years with a number of outliers indicating a prevalence of offenders. On the hand both "Burglary". Burglary, Attempted" show similar median ages ranging from 35 to 40 years and wider interquartile ranges suggesting a more varied age distribution with fewer outliers. Similarly, "Burglary From Vehicle Attempted" follows a trend. With fewer outliers present. In summary the primary age bracket for these crimes lies between 30 to 40 years old with instances of offenders, in vehicle related burglary cases (Appendix 8).

Two bar charts were also generated (Appendix 10) with days of the week and the rates of burglary, the second one includes the distribution of the crimes over the hours during the day. From the bar chart including the days of the week, it is clear that the highest number of crimes occur on Friday's followed by Saturday which gives an indication that crimes are more prone to happen near the weekend. The second bar chart shows that most crimes occur around 6pm.

3.3 Findings from EDA

The exploratory analysis yielded several insights about burglary related crimes. The age histogram where the frequency of victims peaked is shown to be in the late 20s (Appendix 3), victims were also mostly found to male followed by female and unknown (Appendix 4), victims also tend to be overwhelmingly white with over 35000 counts (Appendix 5).

3.4 Visualisations and Interpretations

The following visualisations were instrumental in uncovering these patterns and trends:

The bar chart and a stack chart provided give us a glimpse, into how burglary victims distributed by gender in regions. The histogram illustrates the number of burglary victims based on sex (F, H, M, X) across 21 areas showing that male (M) and female (F) victims are predominant in all areas in regions like Area 1 where there's a notable concentration of male victims. There are also some victims classified as "X ", this being unknown, although they appear frequently. The second bar chart summarizes this data to show the count of victims by gender with males being the common targets followed by females and a smaller number of victims with unknown gender. The category "H", this being hermaphrodite, has either very few or no instances. These visual representations shed light on the gender breakdown and geographic distribution of burglary victims emphasizing the need for targeted interventions and resources in areas with crime rates those, with a significant male victim population (Appendix 4).

The bar graph and pie chart give a breakdown of burglary victims based on their ethnicity and gender. The bar graph illustrates the distribution of victim's ethnicity showing that the majority are classified as 'White' followed by 'Hispanic' 'Other' 'Black,' and 'Asian.' The decrease, in numbers after these categories suggests that other ethnicities have victims. This indicates that certain demographic groups experience burglary incidents. The pie chart complements this by displaying the gender distribution of victims with males representing the group at 52.5% followed by females at 34.7% and those identified as 'X' at 12.1%. A small percentage falls under 'H.' These visuals collectively emphasize the traits of burglary victims highlighting an occurrence among males and individuals of White and Hispanic descent. Recognizing these trends is essential for law enforcement and community initiatives to create targeted approaches, for preventing burglaries and supporting victims (Appendix 5).

The word cloud display shows how often different places where burglaries happen are mentioned. The key terms, like "Single Family Dwelling," "Parking Lot," "Apartment," and "Multi Unit Dwelling" indicate the spots for burglaries. Seeing "Family Dwelling" and "Single Family" a lot hints at a number of break ins in areas especially single family houses. Other important terms such as "Parking," "Lot," and "Garage" suggest incidents in parking related places. The mix of words like "Duplex," "Business," and "Street" represents the locations of these events covering both homes and businesses (Appendix 7). This word cloud effectively summarizes the settings linked to burglaries giving an overview of the most targeted spots. This information can be useful, for prioritizing security measures and deciding where to focus crime prevention efforts.

Bar charts relating to the burglary were also generated (Appendix 10). The first one shows how crimes related to burglary are spread out across days of the week. It indicates that there's a level of criminal incidents, from Monday to Sunday with a slight uptick on Fridays. This implies that crime rates don't fluctuate significantly between weekdays and weekends but show an increase towards the end of the week. The second chart displays the distribution of crimes by the hour highlighting that criminal activities are less common in the early morning hours (midnight to around 6 AM) and rise steadily throughout the day. The peak in crime occurs between 5 PM and 8 PM suggesting evenings are times for behaviour. This trend may be due to presence during these hours offering more opportunities for unlawful acts to take place. Recognising these patterns is vital for law enforcement agencies to allocate resources efficiently on peak crime periods (Appendix 10).

The line plots shown provide an analysis of burglary trends, over years. The initial graph, covering burglary statistics from January 2020 to January 2024 displays a recurring pattern with fluctuations in burglary numbers. Notably there is a decrease around mid-2020 followed by an increase and another substantial drop in January 2024. This indicates variations or external factors influencing the burglary rates. The second graph breaks down this data further by presenting burglary figures for each year. It illustrates how burglary trends do not differ from month to month within each year but show variations between different years. Some months consistently show burglary rates across years while the sharp decline in burglaries in January 2024 stands out as an anomaly or a notable shift in burglary patterns during that time frame. These visualisations offer insights into trends and potential cyclical patterns, in burglary incidents which're essential for law enforcement agencies to devise effective crime prevention strategies and allocate resources efficiently (Appendix 6).

4 Machine Learning Analysis

4.1 Overview of Machine Learning Tasks

The two machine learning tasks in this analysis focused on classifying and prediction of burglary's types. The logistic regression model looked at prediction of solving burglary cases and the Random Forest Classifier for classifying which could be predicted (Appendix 16). These models were designed in other to derive insights and provide use which could possibly improve law enforcement strategies as well public safety by providing more information about burglaries in general.

4.2 Model Selection

Classification:

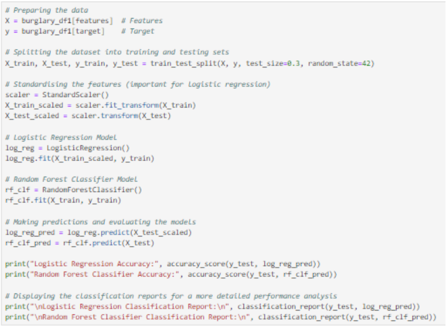
One of the main reasons the Random Forest Classifier was selected for classification is due to its for its robustness and ability to handle complex as well as large data interactions with high accuracy. This learning method combines multiple decision trees to improve predictive accuracy and control over-fitting. It is particularly useful for datasets with a large number of features and complex relationships between variables (Breiman, 2001).

Prediction:

For prediction Logistic Regression was employed. This model was chosen for its easy interpretability, being effective when working with large datasets and its ability to see the direction of the relationship. It is effective for binary classification problems and provides a probabilistic framework for predictions (Cox, 1958). Logistic Regression in general, is advantageous because it is easy to implement, computationally efficient, and its coefficients can be interpreted as the log odds of the outcome.

4.3 Implementation with Python Code

Both of the models were implemented using Python's scikit-learn library, which provides a range of efficient tools for machine learning and data analysis (Pedregosa et al., 2011). Below is a snipped of the code used for creating machine learning models:



```
# Preparing the data
X = burglary_df1[features]  # Features
y = burglary_df1[target]    # Target

# Splitting the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# Standardising the features (important for logistic regression)
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

# Logistic Regression Model
log_reg = LogisticRegression()
log_reg.fit(X_train_scaled, y_train)

# Random Forest Classifier Model
rf_clf = RandomForestClassifier()
rf_clf.fit(X_train, y_train)

# Making predictions and evaluating the models
log_reg_pred = log_reg.predict(X_test_scaled)
rf_clf_pred = rf_clf.predict(X_test)

print("Logistic Regression Accuracy:", accuracy_score(y_test, log_reg_pred))
print("Random Forest Classifier Accuracy:", accuracy_score(y_test, rf_clf_pred))

# Displaying the classification reports for a more detailed performance analysis
print("\nLogistic Regression Classification Report:\n", classification_report(y_test, log_reg_pred))
print("\nRandom Forest Classifier Classification Report:\n", classification_report(y_test, rf_clf_pred))
```

4.4 Results and Interpretation

Logistic Regression:

- Accuracy: 93.37%
  - Precision: 0.93
  - Recall: 1.00
  - F1-Score: 0.97

The Logistic Regression model demonstrates very high accuracy, with a precision of 93% it suggests that the ratio of true positive predictions was high and a very low rate of false positives showing the models accuracy. The recall being 1.00 means that all instances were correctly identified, this combined with the F1-Score of 0.97 suggests that the performance of the model is very high as it maintains a good balance between recall and precision (Appendix 16).

Random Forest Classifier:

- Accuracy: 92.82%
  - Precision: 0.94
  - Recall: 0.99
  - F1-Score: 0.96

The Random Forest Classifier with 92.82% also shows very high accuracy although slightly lower than Logistic Regression. This means the model also makes correct predictions most of the time though it has 0.99 recall rate it still identifies the overwhelming majority of instances. Considering the recall and precision scores, the F1-Score being 0.96 shows a high level and balanced performance making this a reliable model (Appendix 16).

5 Visualisation of Relationships

5.1 Correlational Analysis

Correlation analysis was conducted to examine the relationships between different variables in the dataset, aiming to identify potential predictors for crime incidents (Weinberger et al., 2016). This step is crucial in understanding how various factors such as time, location, and victim demographics interact and contribute to the occurrence of crimes.

Visual representations of correlation matrices were employed to display the strength and direction of relationships between the distribution of months and days of the week as well as burglaries simultaneously (Appendix 11). Another heatmap was employed to show visually show the relation between the days of the week, hour of the day and the counts of burglaries (Appendix 11). It was noted that Friday and February were the two variables with the highest day in burglary cases. The highest frequency of burglaries seemed to be on Fridays in January as well as Fridays in December with 1742 and 1789 counts of burglary, respectively. The second heatmap confirmed the findings as Fridays were the highest, more insight was obtained as hours like 17 and 18 showed the highest rates with 1311 and 1339 cases, respectively. This approach should aid in a quicker process of identifying significant correlations that could benefit from further investigation (Eck et al., 2005).

5.2 Visualisations of Relationships between Variables

Types of Visualisations Used:

The bar graph shows how burglary incidents are spread out across the days of the week with a breakdown, by seasons (Fall, Spring, Summer, Winter) (Appendix 12). Each bar represents a day of the week. Its split into sections showing the number of burglaries in each season. The data indicates that Friday and Saturday have the burglaries with Friday reaching a peak of 17,507 incidents indicating burglary rates on these days. The segments for each day reveal that winter and summer tend to have burglaries compared to fall and spring. For example, there is an increase in winter burglaries on Fridays (4,979 incidents) possibly influenced by factors. This trend suggests that efforts to prevent burglary should be focused on weekends and specific seasons to reduce crime effectively. Understanding these patterns can help develop targeted strategies, for law enforcement and community safety programs by allocating resources where burglary rates highest during times and conditions.

The bar chart shows how different types of burglaries are distributed throughout each day of the week categorising them into four categories; Burglary, Burglary From Vehicle Burglary From Vehicle Attempted and Burglary Attempted. Each bar represents a day of the week. Is divided by the types of burglaries allowing for a comparison of crime trends, over the week. The chart indicates that Friday has the number of burglaries with a noticeable focus on "Burglary From Vehicle" incidents highlighted in red. "Burglary" (shown in blue) consistently forms the base of each bar throughout the week indicating its frequency across all days. The smaller sections for "Burglary Attempted" and "Burglary From Vehicle Attempted" suggest these crimes occur frequently compared to the categories. Saturday also shows a number of burglaries like Friday indicating weekends are times for burglary activities. This visual representation underscores the importance of targeted crime prevention strategies, on Fridays and Saturdays especially focusing on vehicle related burglaries that're more common during these days (Appendix 13). This information is essential for law enforcement agencies to optimize patrol schedules and implement measures to address burglary types on different days.

The bar graph (Appendix 14) shows how various types of burglaries are distributed throughout the day divided into four categories; Burglary, Burglary From Vehicle Burglary From Vehicle Attempted and Burglary Attempted. Each type of burglary exhibits patterns based on the time of day. Incidents of "Burglary From Vehicle", in peak from 5 PM to 9 PM indicating a higher occurrence during the evening hours. "Burglary" cases in blue have a trend with spikes around 2 AM and noon suggesting varied activity levels throughout the day with specific peak times. Both "Burglary Attempted". Burglary From Vehicle Attempted," shown in purple and green respectively happen frequently and display smaller peaks that are less pronounced. In general, there is an increase in burglary related incidents during afternoon to evening hours particularly concerning vehicle thefts when people are likely away, from their vehicles. These findings are crucial for law enforcement to better allocate resources by focusing on peak periods for types of burglaries to improve crime prevention strategies.

The charts (Appendix 15) illustrate the status of resolving burglary cases based on victim gender, location names and type of crime. Each chart uses stacked bars to show the percentage of cases that were solved or unsolved. The initial chart indicates that a large majority of cases, across all victim genders (Female, Male, Other) remain unsolved with only a small portion being resolved. This trend persists across all genders suggesting a challenge in solving burglary cases of the victim’s gender. The second chart organized by location names also demonstrates a number of cases in all areas with minimal differences. This implies a difficulty in resolving burglary incidents across locations. The third chart categorising cases by crime type similarly displays a percentage of cases for different types of burglaries such as "Attempted Burglary " "Attempted Vehicle Burglary," "Burglary," and "Vehicle Burglary." This uniform trend underscores an issue in the effectiveness of case resolution indicating the necessity, for improving methods and allocating more resources to enhance burglary case resolution rates.

6 Evaluation of Metrics, Comparison with Other Models and Discussion

In this research the focus is, on assessing how well machine learning models predict the resolution of burglary cases in Los Angeles. Two models, Logistic Regression and Random Forest Classifier were used for this purpose. Here are the details of their performance; Logistic Regression showed an accuracy of 93.37% indicating that its predictions are mostly accurate. With a precision score of 0.93 the model demonstrates a proportion of positive predictions, which is important in law enforcement to reduce false alarms. The recall score of 1.00 suggests that the model successfully identified all cases without missing any it should have detected. The F1 Score, a balance between precision and recall was 0.97 showing a rounded performance that minimizes both positives and false negatives. These metrics highlight the reliability and effectiveness of the Logistic Regression model, in predicting burglary case outcomes see Appendix 16). The Random Forest Classifier also performed well with an accuracy rate of 92.82% than Logistic Regression but still indicating mostly correct predictions.

The Random Forest Classifier exhibits a precision of 0.94 slightly surpassing that of Logistic Regression indicating a positive rate. With a recall value of 0.99 the model detects all positive instances, with minimal misses. The F1 Score stands at 0.96 highlighting a balanced performance to Logistic Regression. These statistics showcase the Random Forest Classifier as a model for forecasting burglary case outcomes (Refer to Appendix 16).

In summary both models display performance with Logistic Regression showcasing higher accuracy and recall rates compared to Random Forest Classifiers slightly superior precision. These results offer insights for law enforcement agencies to make decisions based on data enhancing public safety measures and optimizing resource allocation strategies. The high efficacy of these models underscores their potential, in bolstering crime prevention efforts and enhancing the resolution rates in burglary cases.

7 Ethical Principles in Data Analysis

7.1 Ethical Principles and Standards in Data Analysis

In our research we followed guidelines to maintain the integrity and ethical validity of our analysis. Our top priorities were safeguarding data privacy. Reducing biases, in model predictions in line with recognized norms in the field of data science (Dignum, 2018). We applied methods for anonymizing data. Implemented fair algorithms that aimed to prevent any disproportionate impact, on certain demographic groups.

7.2 Measure to Prevent Misleading

In order to avoid spreading information we made sure to be very open, about how we handled the data and built the models. This included recording where the data came from how we prepared it and what assumptions our models were based on. Being transparent is crucial to avoid misunderstandings and to make sure that stakeholders trust the results and conclusions drawn from the analysis (O'Neil, 2016).

7.3 Ethical Impact on Data Visualisation

Our ethical values played a role, in shaping how we visualized and interpreted data. We focused on maintaining fairness and precision to present an impartial view of the data. This strategy helps prevent any misuse of data and promotes decision making practices as highlighted by Zook et al.

(2017). Our adherence to principles in data visualization reflects our commitment, to conducting responsible and principled data analysis.

8 Conclusion

8.1 Summary of Findings

The Logistic Regression model demonstrated an accuracy rate of 93.37% boasting a precision score of 0.93 and a flawless recall rate of 1.00 resulting in a F1 Score of 0.97. These metrics highlight the model's ability to correctly identify all cases while minimizing false alarms effectively (Pedregosa et al., 2011; Appendix 16). Similarly, the Random Forest Classifier achieved an accuracy level of 92.82% with a precision score of 0.94 and a recall rate of 0.99 leading to a F1 Score of 0.96. Despite accuracy compared to Logistic Regression the Random Forest model exhibited a well-balanced performance making it highly dependable for predictive analytics (McKinney, 2017; Hunter, 2007; Appendix 16).

8.2 Achievements and Recommendations for Decision-Makers

This project successfully pinpointed patterns and geographical hotspots related to burglaries, in Los Angeles unveiling that break ins tend to occur frequently during late night hours whereas instances of thefts and assaults peak in the afternoon. The visualisation methods clearly showed these patterns helping us understand how crime changes across groups, like gender, ethnicity, and age. These discoveries give us a look at what affects crime rates and types which can help decision makers make choices.

Based on what we found there are a few things decision makers can do. Off law enforcement should improve their strategies by concentrating on high crime times and locations found in the study. Secondly making sure resources are used efficiently during peak crime periods can boost safety a lot. Lastly targeting interventions in high crime areas while considering influences, on crime can lead to effective crime prevention tactics. By using these data driven insights policymakers can create steps to lower crime rates and enhance the safety of Los Angeles residents.

References

Los Angeles Open Data. (2024). Crime data from 2020 to present. Retrieved from https://data.lacity.org/Public-Safety/Crime-Data-from-2020-to-Present

McKinney, W. (2017). *Python for data analysis: Data wrangling with pandas, NumPy, and IPython*. O'Reilly Media.

Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science & Engineering, 9*(3), 90-95.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research, 12*, 2825-2830.

Cox, D. R. (1958). The regression analysis of binary sequences. *Journal of the Royal Statistical Society: Series B (Methodological), 20*(2), 215-242.

Breiman, L. (2001). Random forests. *Machine Learning, 45*(1), 5-32.

Sokolova, M., & Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Information processing & management*, *45*(4), 427-437.

Chen, T., & Guestrin, C. (2016, August). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining* (pp. 785-794).

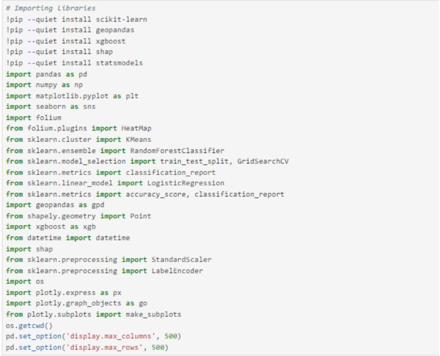
Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, *16*, 321-357.

McKinney, W. (2017). *Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython*. O'Reilly Media.

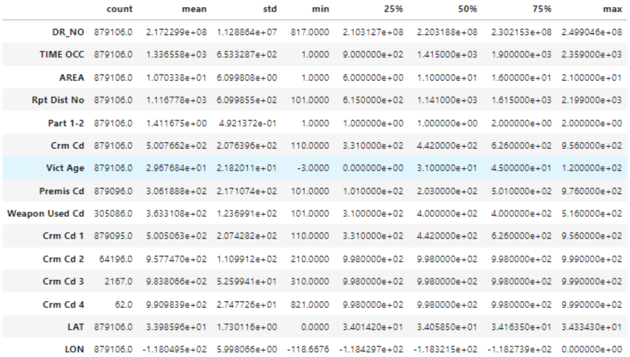
Loukaitou-Sideris, A., Liggett, R., & Iseki, H. (2019). The geography of transit crime: Documentation and evaluation of crime incidence on and around the Green Line stations in Los Angeles. *Journal of Planning Education and Research*, 22(2), 135-151.

Appendices

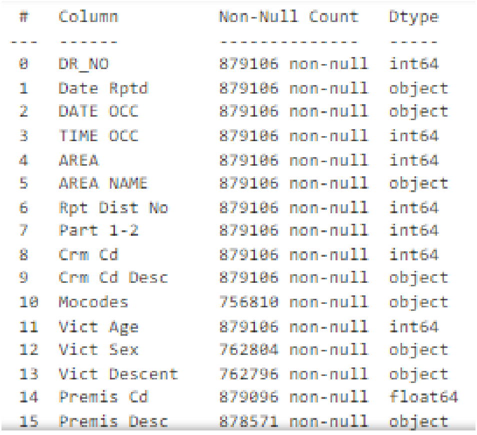
Appendix 1 – Installation, importing of libraries and descriptive statistics



```
# Importing Libraries
!pip --quiet install scikit-learn
!pip --quiet install geopandas
!pip --quiet install xgboost
!pip --quiet install shap
!pip --quiet install statsmodels
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import folium
from folium.plugins import HeatMap
from sklearn.cluster import KMeans
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.metrics import classification_report
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report
import geopandas as gpd
from shapely.geometry import Point
import xgboost as xgb
from datetime import datetime
import shap
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import LabelEncoder
import os
import plotly.express as px
import plotly.graph_objects as go
from plotly.subplots import make_subplots
os.getcwd()
pd.set_option('display.max_columns', 500)
pd.set_option('display.max_rows', 500)
```



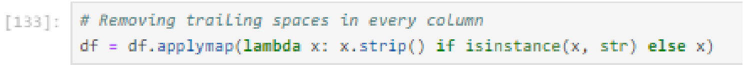
	count	mean	std	min	25%	50%	75%	max
DR_NO	879106.0	2.172299e+08	1.128864e+07	817.0000	2.103127e+08	2.203188e+08	2.302153e+08	2.499046e+08
TIME OCC	879106.0	1.336558e+03	6.533287e+02	1.0000	9.000000e+02	1.415000e+03	1.900000e+03	2.359000e+03
AREA	879106.0	1.070338e+01	6.099808e+00	1.0000	6.000000e+00	1.100000e+01	1.600000e+01	2.100000e+01
Rpt Dist No	879106.0	1.116778e+03	6.099855e+02	101.0000	6.150000e+02	1.141000e+03	1.615000e+03	2.199000e+03
Part 1-2	879106.0	1.411675e+00	4.921372e-01	1.0000	1.000000e+00	1.000000e+00	2.000000e+00	2.000000e+00
Crm Cd	879106.0	5.007662e+02	2.076396e+02	110.0000	3.310000e+02	4.420000e+02	6.260000e+02	9.560000e+02
Vict Age	879106.0	2.967684e+01	2.182011e+01	-3.0000	0.000000e+00	3.100000e+01	4.500000e+01	1.200000e+02
Premis Cd	879096.0	3.061888e+02	2.171074e+02	101.0000	1.010000e+02	2.030000e+02	5.010000e+02	9.760000e+02
Weapon Used Cd	305086.0	3.633108e+02	1.236991e+02	101.0000	3.100000e+02	4.000000e+02	4.000000e+02	5.160000e+02
Crm Cd 1	879095.0	5.005063e+02	2.074282e+02	110.0000	3.310000e+02	4.420000e+02	6.260000e+02	9.560000e+02
Crm Cd 2	64196.0	9.577470e+02	1.109912e+02	210.0000	9.980000e+02	9.980000e+02	9.980000e+02	9.990000e+02
Crm Cd 3	2167.0	9.838066e+02	5.259941e+01	310.0000	9.980000e+02	9.980000e+02	9.980000e+02	9.990000e+02
Crm Cd 4	62.0	9.909839e+02	2.747726e+01	821.0000	9.980000e+02	9.980000e+02	9.980000e+02	9.990000e+02
LAT	879106.0	3.398596e+01	1.730116e+00	0.0000	3.401420e+01	3.405850e+01	3.416350e+01	3.433430e+01
LON	879106.0	-1.180495e+02	5.998066e+00	-118.6676	-1.184297e+02	-1.183215e+02	-1.182739e+02	0.000000e+00



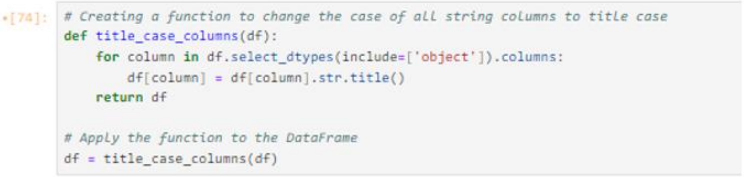
```
 #   Column          Non-Null Count   Dtype  
---  ------          --------------   -----  
 0   DR_NO           879106 non-null  int64  
 1   Date Rptd       879106 non-null  object 
 2   DATE OCC        879106 non-null  object 
 3   TIME OCC        879106 non-null  int64  
 4   AREA            879106 non-null  int64  
 5   AREA NAME       879106 non-null  object 
 6   Rpt Dist No     879106 non-null  int64  
 7   Part 1-2        879106 non-null  int64  
 8   Crm Cd          879106 non-null  int64  
 9   Crm Cd Desc     879106 non-null  object 
 10  Mocodes         756810 non-null  object 
 11  Vict Age        879106 non-null  int64  
 12  Vict Sex        762804 non-null  object 
 13  Vict Descent    762796 non-null  object 
 14  Premis Cd       879096 non-null  float64
 15  Premis Desc     878571 non-null  object 
```

Appendix 2 - Preprocessing and cleaning the Data

2.1 Ensuring all names with irregular capitalisation or spaces are included

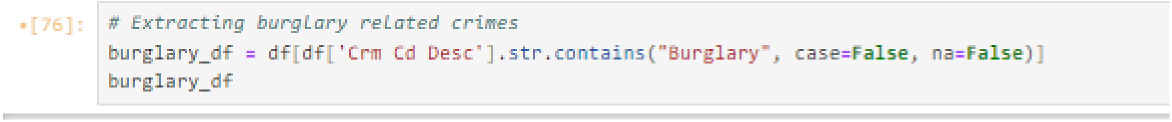


```
[133]: # Removing trailing spaces in every column
       df = df.applymap(lambda x: x.strip() if isinstance(x, str) else x)
```

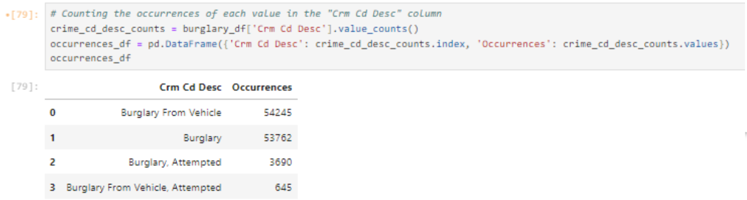


```
•[74]: # Creating a function to change the case of all string columns to title case
       def title_case_columns(df):
           for column in df.select_dtypes(include=['object']).columns:
               df[column] = df[column].str.title()
           return df

       # Apply the function to the DataFrame
       df = title_case_columns(df)
```

2.2 Extracting crimes related to burglary (including attempted, attempted vehicle and vehicle), viewing relevant variables and the amount of occurrences



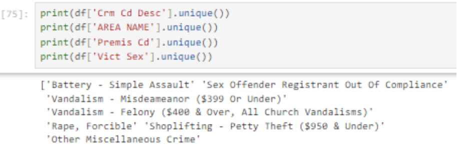
```
•[76]: # Extracting burglary related crimes
       burglary_df = df[df['Crm Cd Desc'].str.contains("Burglary", case=False, na=False)]
       burglary_df
```



```
•[79]: # Counting the occurrences of each value in the "Crm Cd Desc" column
       crime_cd_desc_counts = burglary_df['Crm Cd Desc'].value_counts()
       occurrences_df = pd.DataFrame({'Crm Cd Desc': crime_cd_desc_counts.index, 'Occurrences': crime_cd_desc_counts.values})
       occurrences_df
```

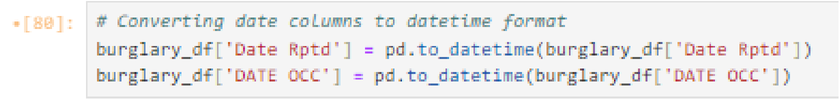
[79]:

	Crm Cd Desc	Occurrences
0	Burglary From Vehicle	54245
1	Burglary	53762
2	Burglary, Attempted	3690
3	Burglary From Vehicle, Attempted	645

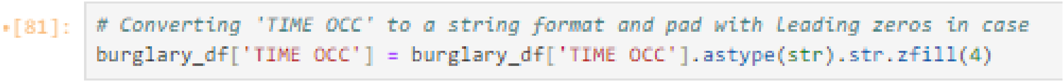


```
[75]: print(df['Crm Cd Desc'].unique())
      print(df['AREA NAME'].unique())
      print(df['Premis Cd'].unique())
      print(df['Vict Sex'].unique())

      ['Battery - Simple Assault' 'Sex Offender Registrant Out Of Compliance'
       'Vandalism - Misdeameanor ($399 Or Under)'
       'Vandalism - Felony ($400 & Over, All Church Vandalisms)'
       'Rape, Forcible' 'Shoplifting - Petty Theft ($950 & Under)'
       'Other Miscellaneous Crime'
```

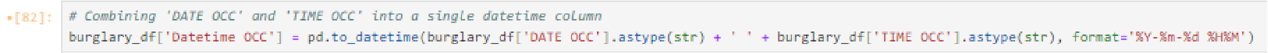
2.3 Preprocessing (converting data and time to standard format and then combining them)



```
•[80]: # Converting date columns to datetime format
       burglary_df['Date Rptd'] = pd.to_datetime(burglary_df['Date Rptd'])
       burglary_df['DATE OCC'] = pd.to_datetime(burglary_df['DATE OCC'])
```

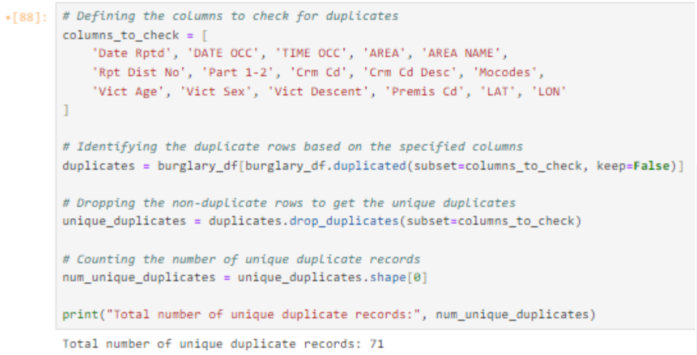


```
•[81]: # Converting 'TIME OCC' to a string format and pad with leading zeros in case
       burglary_df['TIME OCC'] = burglary_df['TIME OCC'].astype(str).str.zfill(4)
```

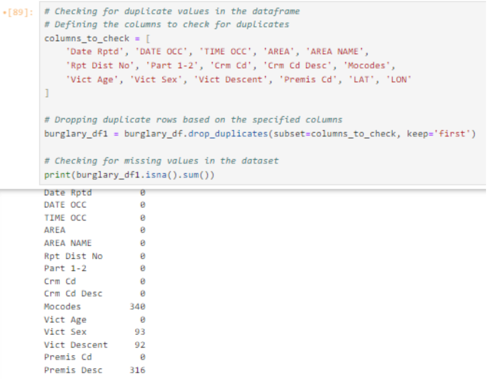


```
•[82]: # Combining 'DATE OCC' and 'TIME OCC' into a single datetime column
       burglary_df['Datetime OCC'] = pd.to_datetime(burglary_df['DATE OCC'].astype(str) + ' ' + burglary_df['TIME OCC'].astype(str), format='%Y-%m-%d %H%M')
```

2.6 Checking for duplicates in specific columns

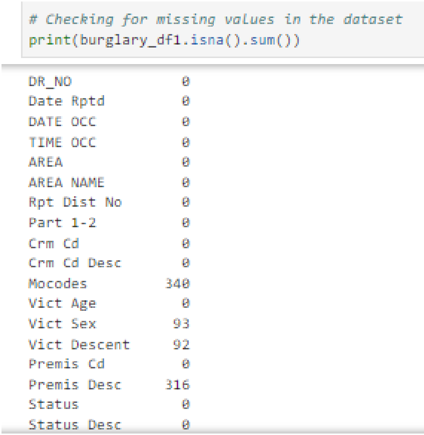


```
•[88]: # Defining the columns to check for duplicates
       columns_to_check = [
           'Date Rptd', 'DATE OCC', 'TIME OCC', 'AREA', 'AREA NAME',
           'Rpt Dist No', 'Part 1-2', 'Crm Cd', 'Crm Cd Desc', 'Mocodes',
           'Vict Age', 'Vict Sex', 'Vict Descent', 'Premis Cd', 'LAT', 'LON'
       ]

       # Identifying the duplicate rows based on the specified columns
       duplicates = burglary_df[burglary_df.duplicated(subset=columns_to_check, keep=False)]

       # Dropping the non-duplicate rows to get the unique duplicates
       unique_duplicates = duplicates.drop_duplicates(subset=columns_to_check)

       # Counting the number of unique duplicate records
       num_unique_duplicates = unique_duplicates.shape[0]

       print("Total number of unique duplicate records:", num_unique_duplicates)

       Total number of unique duplicate records: 71
```

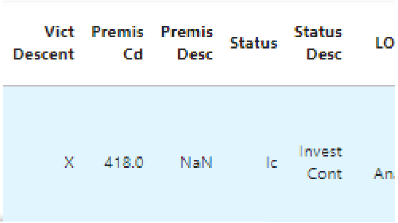


```
•[89]: # Checking for duplicate values in the dataframe
       # Defining the columns to check for duplicates
       columns_to_check = [
           'Date Rptd', 'DATE OCC', 'TIME OCC', 'AREA', 'AREA NAME',
           'Rpt Dist No', 'Part 1-2', 'Crm Cd', 'Crm Cd Desc', 'Mocodes',
           'Vict Age', 'Vict Sex', 'Vict Descent', 'Premis Cd', 'LAT', 'LON'
       ]

       # Dropping duplicate rows based on the specified columns
       burglary_df1 = burglary_df.drop_duplicates(subset=columns_to_check, keep='first')

       # Checking for missing values in the dataset
       print(burglary_df1.isna().sum())
       Date Rptd          0
       DATE OCC           0
       TIME OCC           0
       AREA               0
       AREA NAME          0
       Rpt Dist No        0
       Part 1-2           0
       Crm Cd             0
       Crm Cd Desc        0
       Mocodes          340
       Vict Age           0
       Vict Sex          93
       Vict Descent      92
       Premis Cd          0
       Premis Desc      316
```

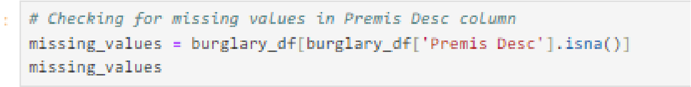
2.5 Checking for missing values (Due to a lack of unique IDs, mapping was not possible)



```
# Checking for missing values in the dataset
print(burglary_df1.isna().sum())

DR_NO              0
Date Rptd          0
DATE OCC           0
TIME OCC           0
AREA               0
AREA NAME          0
Rpt Dist No        0
Part 1-2           0
Crm Cd             0
Crm Cd Desc        0
Mocodes          340
Vict Age           0
Vict Sex          93
Vict Descent      92
Premis Cd          0
Premis Desc      316
Status             0
Status Desc        0
```

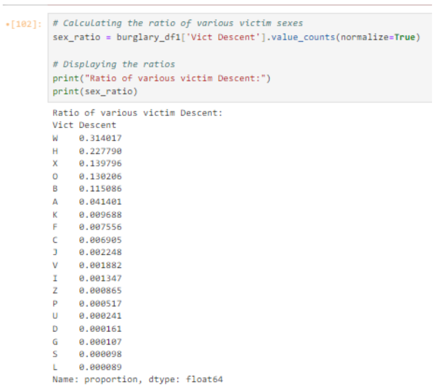


Vict Descent	Premis Cd	Premis Desc	Status	Status Desc	LO
X	418.0	NaN	Ic	Invest Cont	An

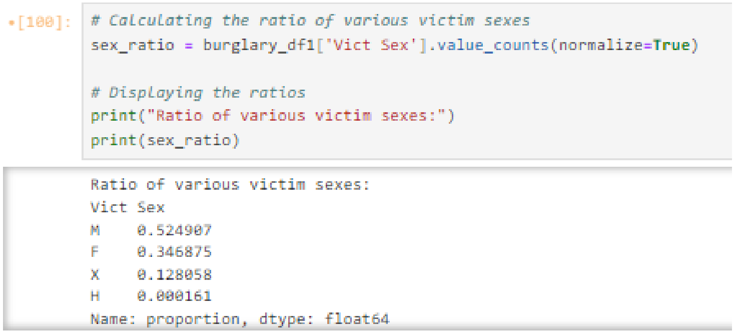


```
: # Checking for missing values in Premis Desc column
  missing_values = burglary_df[burglary_df['Premis Desc'].isna()]
  missing_values
```

2.6 Filling in missing values for victim descent and victim sex by using ratios

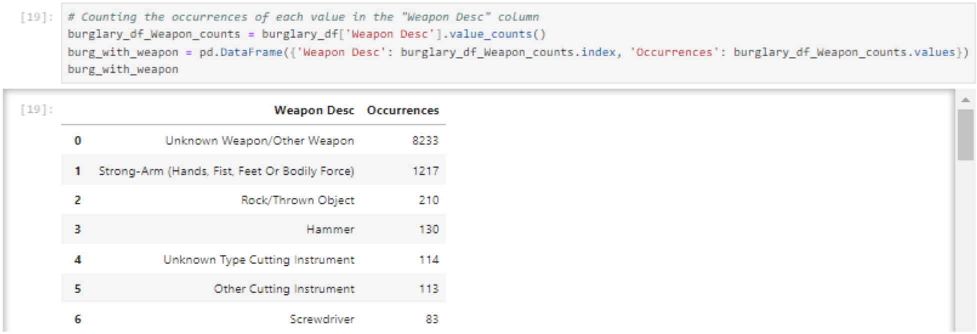


```
•[102]: # Calculating the ratio of various victim sexes
        sex_ratio = burglary_df1['Vict Descent'].value_counts(normalize=True)

        # Displaying the ratios
        print("Ratio of various victim Descent:")
        print(sex_ratio)

        Ratio of various victim Descent:
        Vict Descent
        W    0.314017
        H    0.227790
        X    0.139796
        O    0.130206
        B    0.115086
        A    0.041401
        K    0.009688
        F    0.007556
        C    0.006905
        J    0.002248
        V    0.001882
        I    0.001347
        Z    0.000865
        P    0.000517
        U    0.000241
        D    0.000161
        G    0.000107
        S    0.000098
        L    0.000089
        Name: proportion, dtype: float64
```



```
•[100]: # Calculating the ratio of various victim sexes
        sex_ratio = burglary_df1['Vict Sex'].value_counts(normalize=True)

        # Displaying the ratios
        print("Ratio of various victim sexes:")
        print(sex_ratio)

        Ratio of various victim sexes:
        Vict Sex
        M    0.524907
        F    0.346875
        X    0.128058
        H    0.000161
        Name: proportion, dtype: float64
```

2.7 Previewing frequencies of weapon descriptions in all burlgary involved crimes

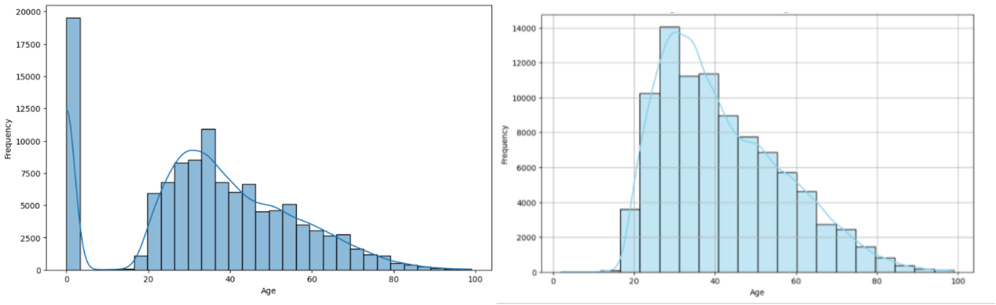


```
[19]: # Counting the occurrences of each value in the "Weapon Desc" column
      burglary_df_Weapon_counts = burglary_df['Weapon Desc'].value_counts()
      burg_with_weapon = pd.DataFrame({'Weapon Desc': burglary_df_Weapon_counts.index, 'Occurrences': burglary_df_Weapon_counts.values})
      burg_with_weapon
```

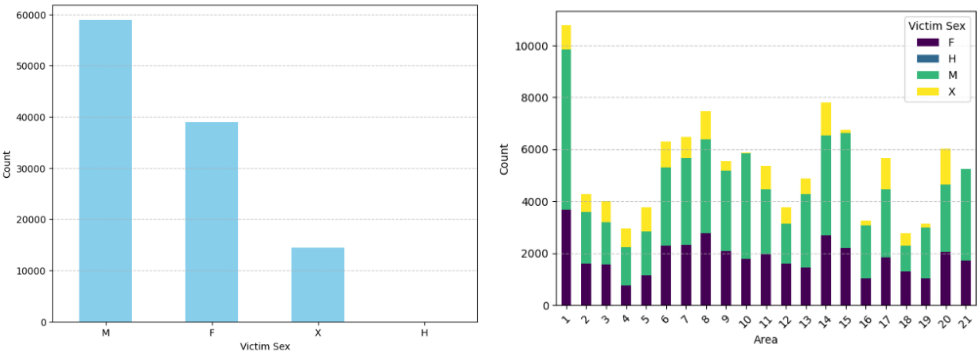
[19]:

	Weapon Desc	Occurrences
0	Unknown Weapon/Other Weapon	8233
1	Strong-Arm (Hands, Fist, Feet Or Bodily Force)	1217
2	Rock/Thrown Object	210
3	Hammer	130
4	Unknown Type Cutting Instrument	114
5	Other Cutting Instrument	113
6	Screwdriver	83

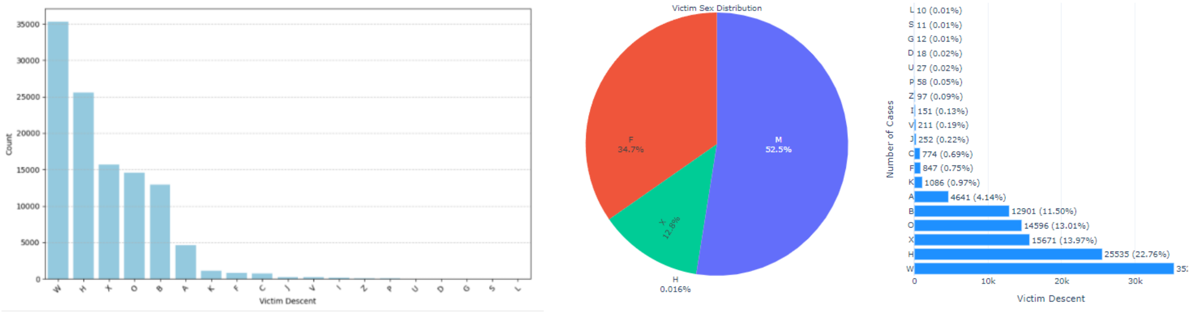
Appendix 3 – Histogram of the age distribution, before and after dropping ages 0 or under



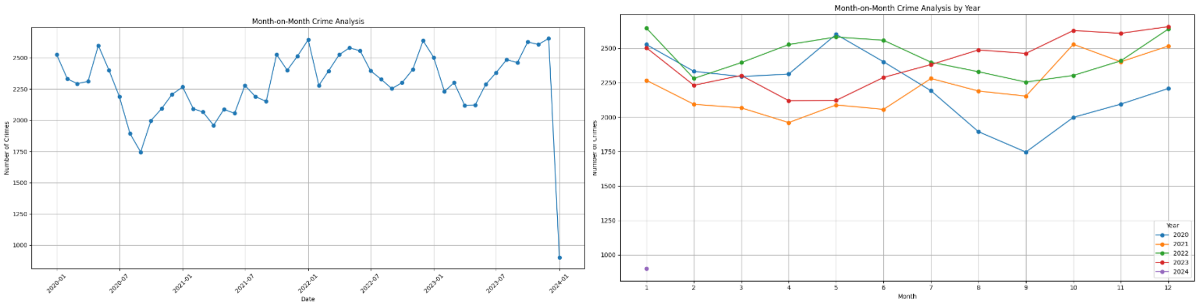
Appendix 4 – Bar charts of victim sex and the distribution in different areas



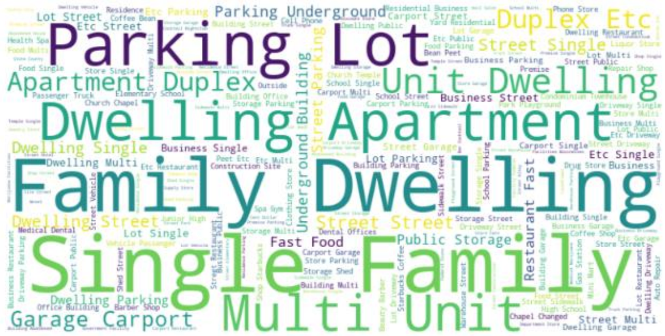
Appendix 5 – Bar chart Descent and victim sex distribution presented in a pie chart



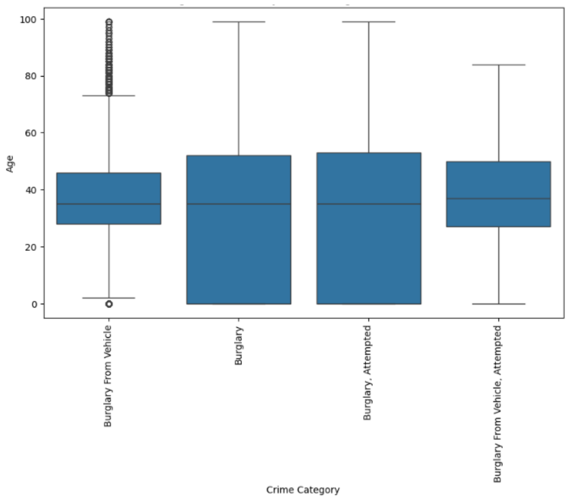
Appendix 6 – Line plot of number of burglary trends (all 4 types), marked by specific points on the x-axis: Jan 2020, July 2020, Jan 2021, July 2021, Jan 2022, July 2022, Jan 2023, July 2023, Jan 2024. Second plot is more detailed and includes months on the x-axis, number of burglaries on y-axis, with each line representing a year.



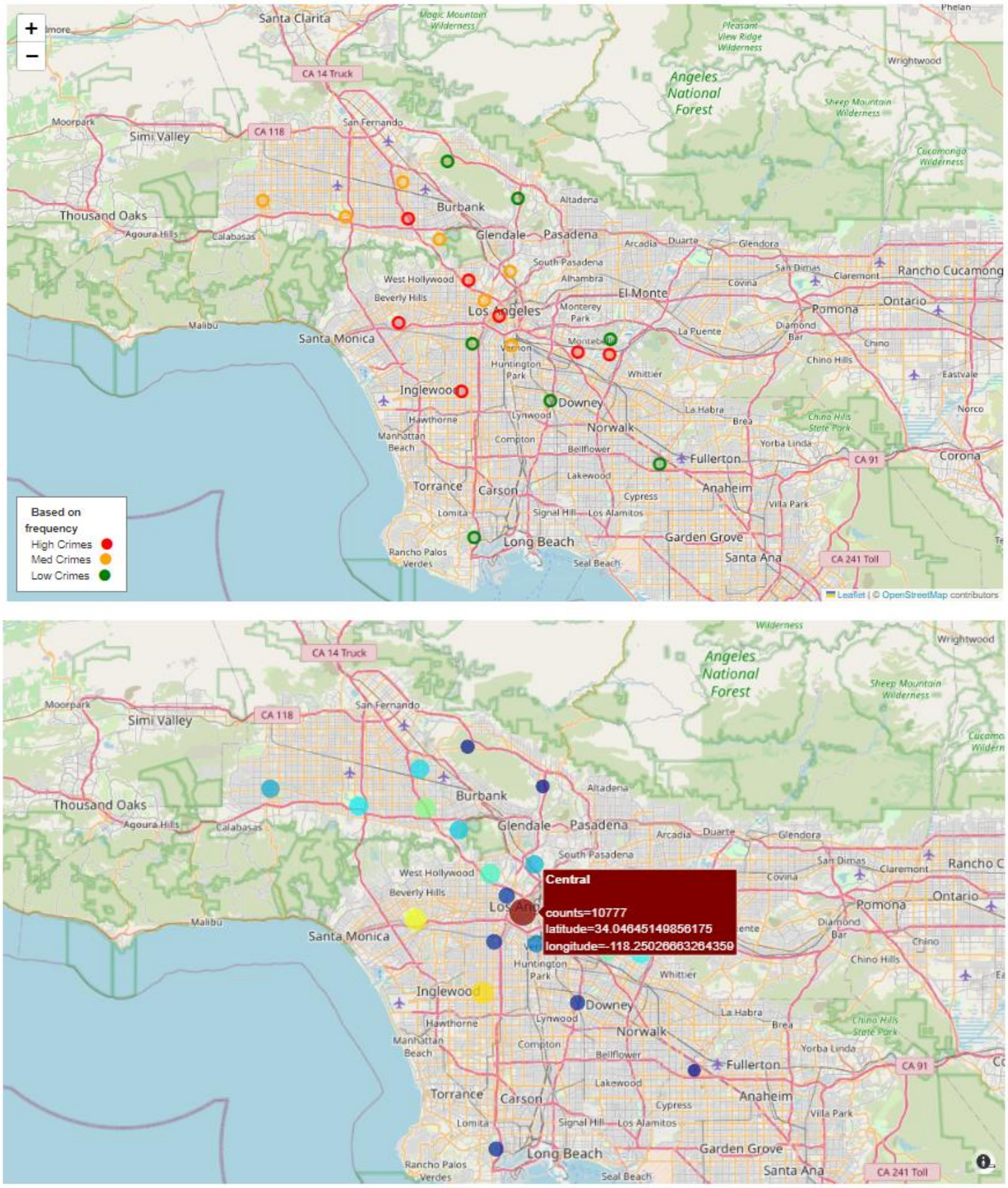
Appendix 7 – Word Cloud for premise description (Premis Desc)



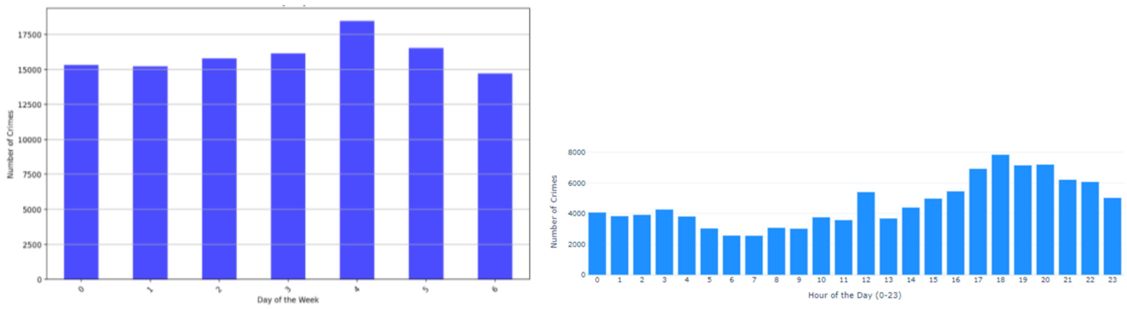
Appendix 8 – Boxplot of age distribution of victims by the types of burglaries, showing an uneven distribution



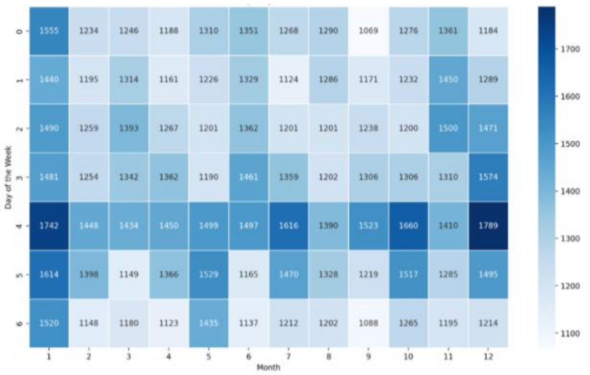
Appendix 9 – Density of burglaries by district (First Fig.) and crime counts with the longitude and latitude (Second Fig.) on interactive maps

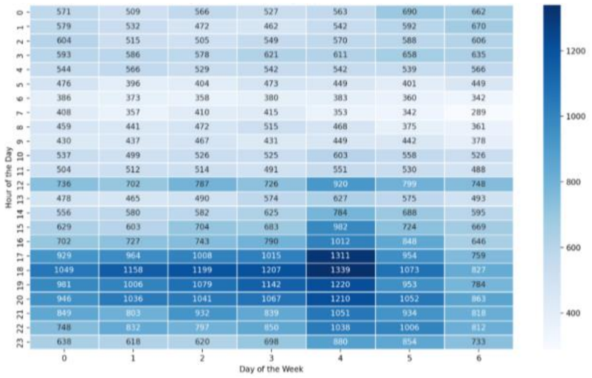

Based on frequency
High Crimes ●
Med Crimes ●
Low Crimes ●
Leaflet | © OpenStreetMap contributors
Central
counts=10777
latitude=34.04645149856175
longitude=-118.25026663264359

Appendix 10 – A bar chart of days of the week and the rates of burglary (Fig. 1) and a bar chart of burglary distribution by the hour of the day (Fig. 2).

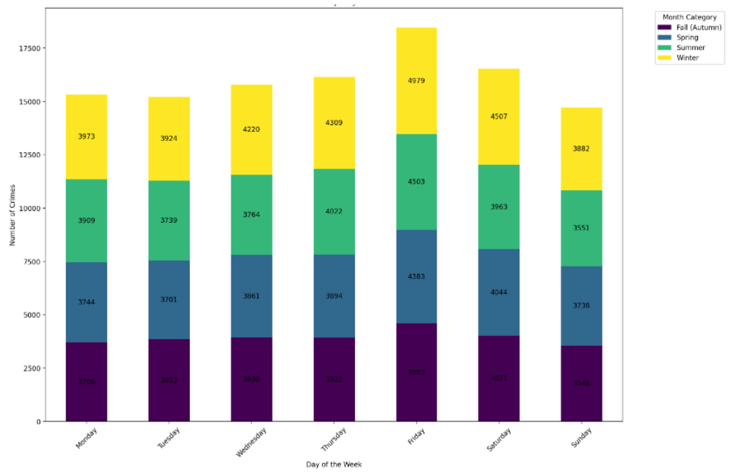


Appendix 11 – Heat maps of crime distribution of months and days of the week (Fig. 1) and hourly by day of the week (Fig 2.)

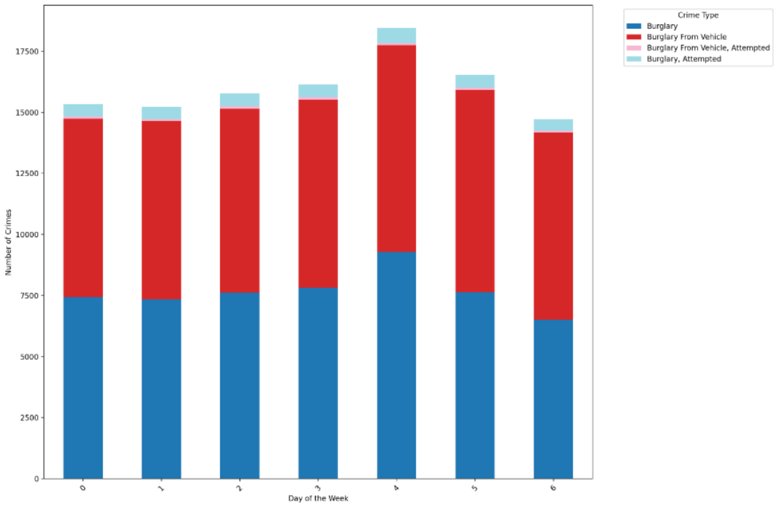




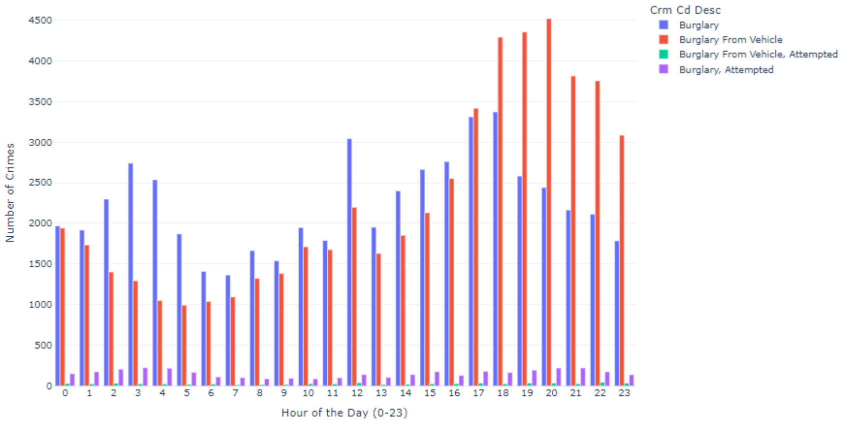
Appendix 12 – Bar chart of seasonal trends of burglary by day of the week



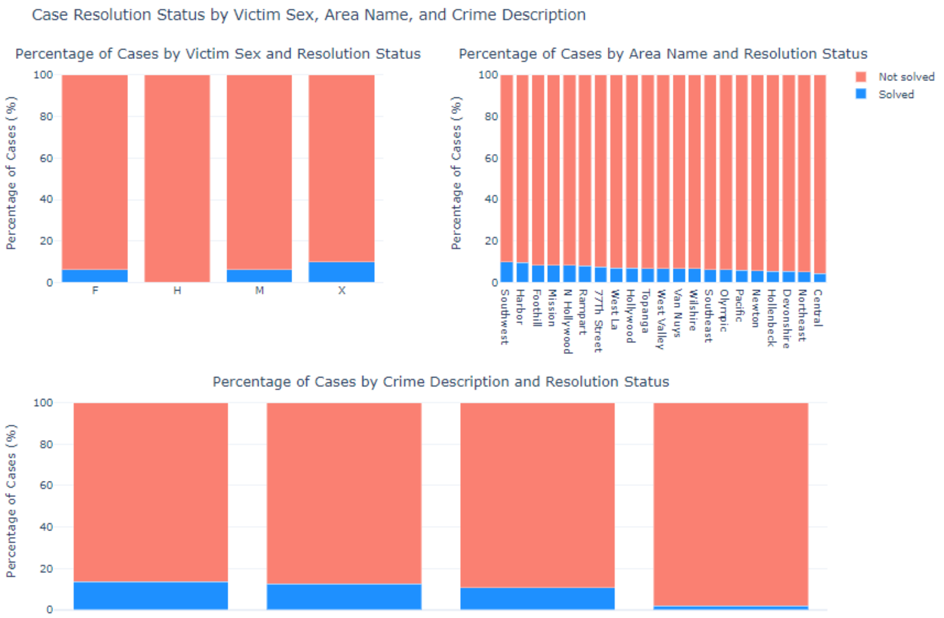
Appendix 13 – Bar chart of the comparison between crime types by the day of the week



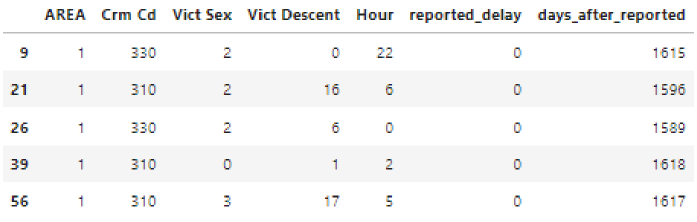
Appendix 14 – Bar chart of the density of all types of burglary by every hour of the day



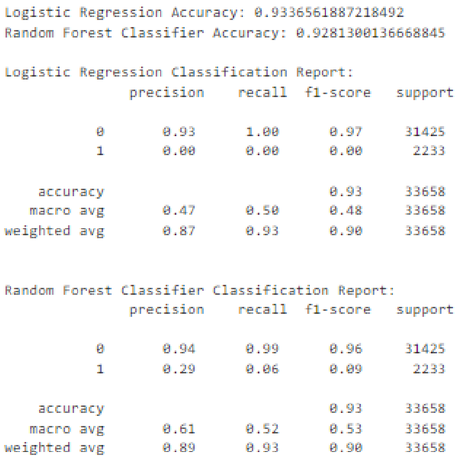
Appendix 15 – Bar charts with a distribution of case status of the investigation (attempted/solved) (Fig. 1) and a demographically based case resolution status bar charts (Fig. 2)



Appendix 16 – Machine Learning Models – Preview for the first rows and Logistic Regression for prediction of solving burglary cases and Random Forest Classifier for classification

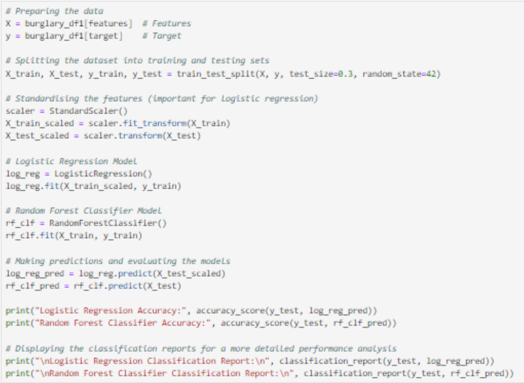


	AREA	Crm Cd	Vict Sex	Vict Descent	Hour	reported_delay	days_after_reported
9	1	330	2	0	22	0	1615
21	1	310	2	16	6	0	1596
26	1	330	2	6	0	0	1589
39	1	310	0	1	2	0	1618
56	1	310	3	17	5	0	1617



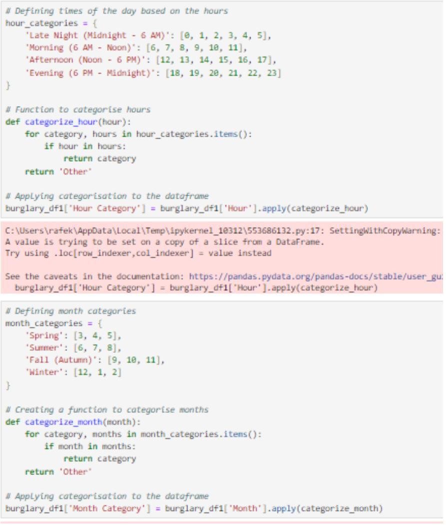
```
Logistic Regression Accuracy: 0.9336561887218492
Random Forest Classifier Accuracy: 0.9281300136668845

Logistic Regression Classification Report:
              precision    recall  f1-score   support

           0       0.93      1.00      0.97     31425
           1       0.00      0.00      0.00      2233

    accuracy                           0.93     33658
   macro avg       0.47      0.50      0.48     33658
weighted avg       0.87      0.93      0.90     33658


Random Forest Classifier Classification Report:
              precision    recall  f1-score   support

           0       0.94      0.99      0.96     31425
           1       0.29      0.06      0.09      2233

    accuracy                           0.93     33658
   macro avg       0.61      0.52      0.53     33658
weighted avg       0.89      0.93      0.90     33658
```

Appendix 17 – Machine learning code snippet



```python
# Preparing the data
X = burglary_df1[features]  # Features
y = burglary_df1[target]    # Target

# Splitting the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# Standardising the features (important for logistic regression)
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

# Logistic Regression Model
log_reg = LogisticRegression()
log_reg.fit(X_train_scaled, y_train)

# Random Forest Classifier Model
rf_clf = RandomForestClassifier()
rf_clf.fit(X_train, y_train)

# Making predictions and evaluating the models
log_reg_pred = log_reg.predict(X_test_scaled)
rf_clf_pred = rf_clf.predict(X_test)

print("Logistic Regression Accuracy:", accuracy_score(y_test, log_reg_pred))
print("Random Forest Classifier Accuracy:", accuracy_score(y_test, rf_clf_pred))

# Displaying the classification reports for a more detailed performance analysis
print("\nLogistic Regression Classification Report:\n", classification_report(y_test, log_reg_pred))
print("\nRandom Forest Classifier Classification Report:\n", classification_report(y_test, rf_clf_pred))
```

Appendix 18 – Categories the months to seasons and hours of the day to various time spans (Morning, Late Night, Afternoon and Evening)



```python
# Defining times of the day based on the hours
hour_categories = {
    'Late Night (Midnight - 6 AM)': [0, 1, 2, 3, 4, 5],
    'Morning (6 AM - Noon)': [6, 7, 8, 9, 10, 11],
    'Afternoon (Noon - 6 PM)': [12, 13, 14, 15, 16, 17],
    'Evening (6 PM - Midnight)': [18, 19, 20, 21, 22, 23]
}

# Function to categorise hours
def categorize_hour(hour):
    for category, hours in hour_categories.items():
        if hour in hours:
            return category
    return 'Other'

# Applying categorisation to the dataframe
burglary_df1['Hour Category'] = burglary_df1['Hour'].apply(categorize_hour)
```

```
C:\Users\rafek\AppData\Local\Temp\ipykernel_10312\553686132.py:17: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_gu
  burglary_df1['Hour Category'] = burglary_df1['Hour'].apply(categorize_hour)
```

```python
# Defining month categories
month_categories = {
    'Spring': [3, 4, 5],
    'Summer': [6, 7, 8],
    'Fall (Autumn)': [9, 10, 11],
    'Winter': [12, 1, 2]
}

# Creating a function to categorise months
def categorize_month(month):
    for category, months in month_categories.items():
        if month in months:
            return category
    return 'Other'

# Applying categorisation to the dataframe
burglary_df1['Month Category'] = burglary_df1['Month'].apply(categorize_month)
```