DC Metro Crime Data Analysis

feyintola lasekan

July 8, 2024

1 Introduction

This report analyzes crime data from the DC Metro area, specifically focusing on the dataset `dc_crime_add_vars.csv`. The dataset contains information on various crime incidents, including details such as report date, shift, offense type, location, and additional variables.

2 Data Description

The dataset contains 32 columns and multiple rows, each representing a crime incident. Key columns include:

- **REPORT_DAT**: Date and time the crime was reported.
- **SHIFT**: The shift during which the crime occurred.
- **OFFENSE**: Type of offense.
- **METHOD**: Method of offense.
- **BLOCK**: Location of the crime.
- **DISTRICT**, **PSA**, **WARD**: Geographic identifiers.
- **year**, **month**, **day**, **hour**, **minute**, **second**: Temporal details of the crime.
- **EW**, **NS**, **quad**: Directional and quadrant information.
- **crimetype**: Categorization into violent and non-violent crimes.

3 Data Quality

The dataset has several columns with missing values:

- **DISTRICT**: 200 missing values.

- **PSA**: 251 missing values.
- **NEIGHBORHOOD_CLUSTER**: 4705 missing values.
- **BLOCK_GROUP**, **CENSUS_TRACT**: 1091 missing values each.
- **VOTING_PRECINCT**: 84 missing values.
- **START_DATE**, **END_DATE**: 13 and 11651 missing values respectively.

4 Python Code for Analysis

`This Python 3 environment comes with many helpful analytics libraries installed`

It is defined by the kaggle/python Docker image: `https://github.com/kaggle/docker-python`. For example, here's several helpful packages to load:

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

# Input data files are available in the read-only "../input/" directory
# For example, running this (by clicking run or pressing Shift+Enter) will list

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))

# You can write up to 20GB to the current directory (/kaggle/working/) that gets
# You can also write temporary files to /kaggle/temp/, but they won't be saved o

import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report
from sklearn.linear_model import LinearRegression
from sklearn.impute import SimpleImputer
from sklearn.metrics import mean_squared_error

# Load the dataset
df = pd.read_csv('/kaggle/input/dc-metro-crime-data/dc_crime_add_vars.csv')

# Display the first few rows of the dataset
print(df.head())

# Check for any missing values
```

```
print(df.isnull().sum())

# Select numerical columns only
numerical_cols = df.select_dtypes(include=['float64', 'int64']).columns
df_numerical = df[numerical_cols]

# Define your target variable and features
target = 'X'  # Replace with your actual target variable name
features = df_imputed.drop(columns=[target])

# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(features, df_imputed[target]

# Instantiate the linear regression model
model = LinearRegression()

# Fit the model
model.fit(X_train, y_train)

# Predict on the testing data
y_pred = model.predict(X_test)

# Optionally, evaluate the model
mse = mean_squared_error(y_test, y_pred)
print(f"Mean Squared Error: {mse}")

# Print coefficients and intercept if needed
print("Coefficients:", model.coef_)
print("Intercept:", model.intercept_)
```

5 Explanation of Python Code

The Python code performs the following steps:

1. Load necessary libraries for data processing, model building, and evaluation.
2. List all files in the input directory to confirm the dataset's presence.
3. Load the dataset using `pandas.read_csv()`.
4. Display the first few rows of the dataset to understand its structure.
5. Check for missing values in the dataset.
6. Select only numerical columns for analysis.

7. Define the target variable and features, then split the data into training and testing sets.
8. Instantiate and fit a Linear Regression model on the training data.
9. Predict on the testing data and calculate the Mean Squared Error (MSE) to evaluate the model.
10. Print the model's coefficients and intercept.

6 Analysis and Findings

Initial analysis shows that the majority of crimes are non-violent. The data distribution across various shifts and geographic areas provides insights into crime patterns.

7 Conclusion

The dataset provides a comprehensive view of crime in the DC Metro area, though some columns have missing data which may need addressing for detailed analysis.

8 Future Work

Future analysis could focus on filling missing values, deeper temporal analysis, and spatial distribution to identify crime hotspots.

9 Your Presentation

Included in this report is the PowerPoint presentation that details the analysis performed on the DC Metro crime data. The presentation covers the data description, data quality assessment, initial findings, and the Python code used for the analysis, along with explanations and conclusions.

10 Source Code(s)

The source code used for the analysis is provided in the Python code section above. This code was executed in a Kaggle Python environment, leveraging libraries such as `pandas` for data manipulation, `numpy` for numerical operations, and `sklearn` for machine learning tasks including data splitting and model fitting.

11 Source Files for Figures

Any figures or visualizations generated from the analysis are saved as image files and included in the `figures` directory of the project. These figures illustrate key findings from the data analysis, such as the distribution of crime types and temporal patterns.

12 Excel Sheets for Datasets

The dataset used in this analysis is provided in CSV format, which can be easily opened in Excel for further exploration. The file `dc_crime_add_vars.csv` contains all the crime data analyzed in this report.