Comprehensive Crime Analysis and Prediction System: A Multi-Model Approach

December 8, 2024

Abstract

This project focuses on the development of a comprehensive crime analysis and prediction system, aimed at enhancing urban safety and law enforcement efforts. By leveraging advanced machine learning algorithms and data visualizations, the system analyzes historical crime data to assess safety levels across different neighborhoods. The key features of the project include neighborhood crime analysis, incident forecasting, and the creation of an interactive 3D crime map. Predictive models, such as Support Vector Machine (SVM) classifiers and XGBoost regressors, are utilized to predict the likelihood of specific crime types based on temporal and spatial features. Furthermore, the dashboard incorporates various visualization tools like heatmaps, pie charts, and bar charts to provide a clear view of crime distribution and peak hours, enabling stakeholders to make informed decisions. A web application built using Streamlit offers real-time interaction with the model, while integrating Plotly for dynamic visualizations. The project highlights the importance of using data-driven approaches for effective crime prevention and resource allocation. Future work focuses on incorporating real-time data, demographic overlays, and more advanced machine learning models to further enhance predictive accuracy.

1 Introduction

Urban crime analysis and prediction have become essential tools for law enforcement and policymakers. The increasing availability of open data and advances in machine learning have made it possible to develop sophisticated systems for analyzing and predicting criminal activity. This project integrates multiple machine learning techniques to address various aspects of crime analysis and prediction, including:

- **Crime type prediction**: Utilizing Random Forest Classification to categorize crimes based on historical data.
- **Threat score assessment**: Implementing XGBoost Regression to assign a threat score to specific incidents or regions.
- **Time-series forecasting**: Employing Facebook Prophet to model and predict crime trends over time.
- **Crime likelihood prediction**: Using Support Vector Machines (SVM) to estimate the probability of different crime types in specific areas.

The system is designed to assist stakeholders by providing actionable insights through user-friendly visualizations and predictive capabilities.

2 Dataset and Preprocessing

2.1 Dataset Overview

The dataset used in this project is sourced from Buffalo's open data portal, containing crime incident reports from 2009 onward. The dataset includes various features that provide comprehensive details about each incident. The key features are as follows:

2.2 Incident Details

The dataset includes information such as the type of crime, and the date and time of occurrence. These attributes are essential for understanding criminal activity patterns and are used for classification and regression tasks.

2.3 Location

Geographic data, including coordinates (latitude and longitude) and neighborhood identifiers, enables spatial analysis of crime distribution. This information is crucial for generating 3D crime maps and analyzing the geographic spread of incidents.

2.4 Temporal Features

Temporal attributes such as the hour of the day, day of the week, and month of the year help identify trends and patterns that are time-dependent. These features are particularly valuable for time-series analysis.

2.5 Preprocessing Steps

Several preprocessing steps were undertaken to prepare the data for analysis and modeling:

2.5.1 Temporal Feature Extraction

Date and time data were processed to derive additional features such as the year, month, day, weekday, and hour. These derived features enhance the predictive power of time-dependent models.

2.5.2 Coordinate Normalization

Raw geographic coordinates (latitude and longitude) were converted to floating-point numbers to ensure consistency and accuracy in spatial analysis. This step is vital for generating reliable spatial visualizations and location-based predictions.

2.5.3 Categorical Encoding

Categorical variables, such as crime types and neighborhood identifiers, were converted into numerical representations using techniques like string conversion to lowercase. This transformation allows these features to be effectively used in machine learning models.

2.5.4 Data Cleaning and Sorting

Incident descriptions were cleaned by replacing long descriptions with simplified text, such as replacing "Buffalo Police are investigating this report of a crime..." with "under investigation." Additionally, occurrences of 'UNKNOWN' were replaced with NaN values. The dataset was sorted by the incident date and time to maintain chronological order.

2.5.5 Drop Unnecessary Columns

The column 'created_at' was dropped if present, as it was not necessary for the analysis.

2.5.6 Crime Categorization

Incident types were categorized into broader crime categories such as sexual crimes, assault crimes, vehicle crimes, theft crimes, and murder crimes. The specific crime types were mapped to their respective broader categories, making it easier to analyze crime trends.

2.5.7 Handling Missing Values

Rows containing any NaN values were removed to ensure the integrity of the dataset for modeling purposes.

These preprocessing steps ensure that the data is cleaned, structured, and ready for use in various machine learning models.

3 Real-Time Crime Risk Prediction: Identifying the Highest Risk Day and Hour Using Streamlit

3.1 Introduction

This project involves the implementation of a machine learning model to predict crime risks in Buffalo, focusing on identifying the highest risk day of the week and hour. The model is integrated into an interactive web application built using Streamlit, allowing real-time predictions and visualizations based on user inputs. By analyzing crime data, the system identifies patterns of crime incidents over time and space, helping to understand and mitigate crime risks effectively.

3.2 Data Collection and Preprocessing

The dataset used in this project was sourced from Buffalo's open data portal, containing crime incident reports dating back to 2009. The dataset includes several key features such as incident type, date and time, location (neighborhood), and geographic coordinates. Preprocessing steps included:

- **Temporal Feature Extraction:** The date and time of each incident were extracted into separate features such as day of the week and hour of the day.
- **Categorical Encoding:** Categorical variables, such as neighborhoods and crime types, were encoded using `LabelEncoder` to convert them into numerical values suitable for machine learning models.
- **Data Cleaning:** Missing values and irrelevant columns were handled, and the dataset was normalized for consistency across geographic coordinates.

3.3 Machine Learning Model

A **Random Forest Classifier** was chosen for this task to predict the type of crime based on temporal and spatial features. The features used to train the model include the day of the week, the hour of the day, and the neighborhood of the incident. The model was trained with an 80-20 split for training and testing data, respectively, and was configured with 100 estimators for better accuracy and stability.

3.4 Model Saving and Integration

After training, the model and the label encoders were serialized using `joblib` and stored in a directory for later use in the Streamlit application. The saved files include:

- `crime_risk_model.pkl`: The trained Random Forest model.
- `le_neighborhood.pkl`: The encoder for neighborhood data.
- `le_incident_type.pkl`: The encoder for crime types.

These files were loaded into the Streamlit application to make real-time predictions based on user inputs.

3.5 Streamlit Integration and Real-Time Predictions

The Streamlit application allows users to interact with the crime risk prediction model in real-time. The workflow is as follows:

- **User Input:** Users select a neighborhood, day of the week, and hour of the day.
- **Preprocessing:** The selected neighborhood is encoded, and the day and hour inputs are transformed into a format suitable for the model.
- **Prediction:** The model predicts the probability of various crime types for each combination of day and hour, identifying the time with the highest risk.
- **Visualization:** A heatmap is generated, visualizing crime risk levels for each combination of day and hour.

3.6 Results

The Streamlit application provides users with the day and hour of the week that pose the highest crime risk for the selected neighborhood. For example, users may discover that crimes are more likely to occur on Friday evenings in a particular area. A heatmap also visually displays crime risks across the week and day, allowing users to easily identify patterns and trends in crime activity.

3.7 Conclusion

The integration of machine learning with Streamlit enables real-time crime risk assessment and effective visualization of high-risk times. By identifying the highest risk day and hour for crimes, this system can assist law enforcement agencies and local authorities in making data-driven decisions for crime prevention and resource allocation.

4 Crime Forecasting Using Facebook Prophet Model

4.1 Introduction

This section describes the process of selecting and training a forecasting model to predict crime trends in Buffalo, NY, using historical crime data. The Facebook Prophet model was chosen due to its effectiveness in handling time series data with strong seasonal effects, its ability to handle missing data, and its capacity to manage outliers. Given that crime data often exhibits patterns related to time, such as weekly and yearly cycles, Prophet is particularly well-suited for this task. Additionally, the model's flexibility in capturing trends, holidays, and seasonal variations allows for a comprehensive understanding of crime behavior over time.

4.2 Data Collection and Preparation

The data for this project was sourced from a SQLite database containing incident reports from 2009 onwards. These records were preprocessed to create daily crime counts for each neighborhood in Buffalo. The preprocessing involved several steps: first, the incident dates were converted into datetime objects to enable time-based analysis. Afterward, the data was aggregated by date and neighborhood to calculate the daily crime count for each area of Buffalo. The dataset was further filtered to include only crime incidents after January 1, 2009, ensuring that the analysis would cover relevant and up-to-date information.

The resulting dataset was transformed into a format suitable for input into the Prophet model, where the date column was renamed to `ds` and the crime count column to `y`. This transformation was essential for the model to properly interpret the time series data and generate meaningful forecasts.

4.3 Model Training

The training process involved fitting separate Prophet models for each neighborhood in Buffalo, as well as an additional model for the entire city. This approach allowed for both localized predictions specific to each neighborhood and global forecasts for the entire city. For each neighborhood, data was filtered from the aggregated crime dataset, and a Prophet model was trained on the neighborhood-specific data. This enabled the model to capture localized crime patterns and trends that may differ across different areas of the city.

Once the model was trained, a forecast for the next 365 days was generated, providing a year-long prediction of crime trends for each neighborhood. In addition to the neighborhood models, a city-wide forecast was generated by training a Prophet model on the aggregated data from all neighborhoods. This model provided an overall view of crime trends across Buffalo, allowing for the identification of broader city-wide patterns.

4.4 Model and Forecast Storage

After training the models, they were serialized and saved using the `joblib` library, which allows for efficient storage and retrieval of Python objects. The forecasts for each neighborhood and the city-wide forecast were saved as CSV files, making it easy to load and access them for use in the Streamlit application. The naming convention for the saved files followed a clear structure: the city-wide model was saved as `buffalo_crime_prophet_model.pkl`, the city-wide forecast as `buffalo_crime_forecast.csv`, and the neighborhood-specific models and forecasts were named `prophet_model_neighborhood.pkl` and `forecast_neighborhood.csv`, respectively.

This method of storage ensures that the models and forecasts are easily accessible and can be quickly loaded into the application for real-time use. The efficient retrieval of these files plays a crucial role in maintaining the performance and responsiveness of the Streamlit application.

4.5 Streamlit Application for Crime Forecast Visualization

The Streamlit application was developed to provide an interactive interface for users to explore crime forecasts for different neighborhoods and the city as a whole. Users can select a neighborhood from a dropdown menu, which triggers the application to load the corresponding pre-trained model and forecast data. Once the data is loaded, the application generates a line chart that displays the forecasted number of crimes over the next year, along with a shaded area representing the confidence interval.

The application also offers various interactive features, such as the ability to zoom in and out on specific time periods. Users can view data over the past month, six months, or the entire forecasted period, with the ability to adjust the time range as needed. This flexibility allows users to explore the data dynamically and gain deeper insights into potential crime trends in Buffalo.

4.6 Forecasting Results and Insights

The forecasts generated by the Prophet models provide valuable insights into crime trends at both the neighborhood level and city-wide level. For example, users can observe how crime rates are expected to fluctuate throughout the year, with visual indicators of confidence in the predictions. The inclusion of upper and lower bounds in the forecast helps to highlight the level of uncertainty in the predictions, offering users a clearer understanding of potential variations in crime rates.

This approach not only helps in predicting future crime trends but also allows local authorities and law enforcement agencies to identify high-risk periods and neighborhoods. By analyzing these forecasts, decision-makers can better allocate resources and plan strategies to prevent crime.

4.7 Benefits of the Approach

The use of Facebook Prophet for crime forecasting offers several key advantages. First, the scalability of the approach allows for the creation of separate models for each neighborhood, enabling localized forecasts while maintaining a city-wide model for broader trends. This ensures that both individual neighborhoods and the city as a whole can benefit from accurate crime predictions. Second, the ease of updating the models ensures that forecasts remain current as new data becomes available. As the model is trained on historical data, it can be retrained periodically to incorporate the most recent crime data, ensuring that the forecasts stay relevant.

Finally, the Streamlit application provides a user-friendly, interactive platform for exploring and visualizing the forecasts. The dynamic nature of the application allows users to investigate different time periods and compare forecasts across neighborhoods, making it an invaluable tool for crime analysis and planning. By combining these advantages, the forecasting system provides a deeper understanding of crime patterns, which can be leveraged to develop more effective crime prevention strategies in Buffalo.

5 Threat Score Assessment

5.1 Introduction to Threat Score Assessment

Crime and safety concerns in urban neighborhoods have become significant issues for residents and authorities. This project focuses on developing a threat assessment system that predicts safety levels based on historical incident data by leveraging machine learning and advanced analytics. The system evaluates threat scores using parameters like neighborhood, time, and day of the week. This can assist law enforcement agencies, city planners, and residents in understanding and mitigating safety risks.

5.1.1 Dataset

The data is collected from historical incident reports stored in an SQLite database. Each record includes the following key attributes:

- **Case Details**: Descriptive information about the reported incident, such as type and severity.
- **Location**: Identified by neighborhood and corresponding geographic coordinates.
- **Time**: Date and time of occurrence, offering temporal patterns in criminal activity.

The dataset spans multiple years, providing a comprehensive view of crime trends and allowing for both temporal and spatial threat assessments. By analyzing these records, the system calculates a composite threat score tailored to specific conditions, enabling meaningful and actionable predictions.

5.2 Methods

5.2.1 Modeling

The system employs an XGBoost Regressor (`XGBRegressor`) due to its efficiency and accuracy in handling structured datasets. The modeling process involved the following steps:

- **Data Splitting**: The dataset was divided into training and test sets using the `train_test_split` function, ensuring a balanced distribution of the features.
- **Model Training**: The model was trained with optimized hyperparameters:
  - `n_estimators`=100
  - `learning_rate`=0.1
  - `max_depth`=6
- **Feature Importance**: The model evaluated the contribution of features such as neighborhood, day of the week, and hour, ranking them by their impact on predictions.
- **Model Deployment**: The trained model and supporting encoders were serialized for seamless integration with the web application.

5.2.2 Web Application

An intuitive web application was developed using Streamlit, providing users with real-time interaction to predict threat levels. Key features of the application include:

- **Input Interface**: Users specify parameters like neighborhood, day of the week, and time of day.
- **Visualization**: Threat levels are displayed on an interactive gauge chart powered by Plotly, offering clear and immediate insights.
- **Threat Categories**: Predictions are categorized into low, moderate, and high threat levels, with contextual guidance for safety precautions.

5.2.3 Results

The model demonstrated strong predictive performance on the test data. Evaluation metrics include:

- **Mean Squared Error (MSE)**: 86.60
- **Mean Absolute Error (MAE)**: 6.70
- **R^2 Score**: 0.90

The threat scores are normalized to a scale of 0-100 and categorized as follows:

- **Low Threat (0-33)**: Indicates a safe environment suitable for travel or activities.
- **Moderate Threat (33-66)**: Suggests exercising caution, particularly during less populated hours.
- **High Threat (66-100)**: Warns of significant risk, advising individuals to avoid the area if possible.

5.3 Conclusion

This project successfully demonstrates the application of machine learning for predicting safety levels in urban neighborhoods. The system provides actionable insights for stakeholders by quantifying and categorizing threats, ultimately improving safety awareness and risk mitigation.

6 Analyzing Neighborhood Crime Trends and Forecasting

The Buffalo Crime Analysis Dashboard provides a comprehensive suite of tools and visualizations designed to analyze and forecast crime patterns within neighborhoods. The application leverages advanced predictive modeling and interactive data representations to empower users with actionable insights. Below, the primary features of the dashboard are elaborated upon in detail:

6.1 Neighborhood Crime Analysis and Incident Forecasting

The dashboard incorporates a predictive modeling feature to analyze neighborhood crime trends and forecast potential incidents. A Support Vector Machine (SVM) classifier forms the backbone of the prediction system. This model predicts the likelihood of specific crime types by analyzing various temporal features such as the hour of the day, day of the week, and month. The data preprocessing pipeline ensures high accuracy, utilizing a StandardScaler to normalize feature values and improve the model's performance.

Key capabilities of this feature include visualizing crime hotspots within neighborhoods using Pydeck's Hexagon Layer, which represents crime density in a clear and intuitive format. The dashboard also presents a pie chart to illustrate the distribution of different crime types in the selected neighborhood, offering users a snapshot of local crime dynamics. To enhance interactivity, sliders and dropdowns allow users to input specific time parameters and forecast incident probabilities accordingly. The likelihood of various crime types is presented with the top three categories ranked by their predicted probabilities. Terminology within this feature is carefully designed for clarity, replacing generic terms like "crime risk" with more precise labels such as "Incident Probability Analysis" and "Crime Pattern Analysis."

6.2 3D Crime Map

The 3D crime map is a standout feature of the dashboard, utilizing Pydeck to present a visually immersive and informative representation of crime data. The Hexagon Layer provides a bird's-eye view of crime density, where color-coded columns signify areas of varying crime intensity. Dark red columns represent the highest crime density, while light yellow columns denote the lowest. In addition to the Hexagon Layer, the Scatterplot Layer highlights individual crime incidents, complete with interactive tooltips that provide detailed information about each event. To aid interpretation, a comprehensive map legend is included, ensuring users can easily understand the color codes and their corresponding crime densities.

6.3 Crime Statistics

Crime statistics are presented through two primary visualizations that help users understand the broader patterns and trends:

- The first visualization is a pie chart depicting the distribution of various crime types. This chart allows users to quickly grasp the proportional representation of each type of crime in the dataset.
- The second visualization is a bar chart that highlights the distribution of crimes across different days of the week. This analysis reveals temporal patterns, such as the prevalence of certain crimes on specific days, aiding in strategic decision-making.

6.4 Peak Hours Analysis

To delve deeper into the temporal dimensions of crime, the dashboard features a heatmap that visualizes crime intensity by hour and day of the week. This tool helps identify peak crime hours and days, enabling law enforcement agencies to optimize patrol schedules and allocate resources more effectively. By providing a granular view of when crimes are most likely to occur, the heatmap serves as a critical component for strategic planning.

6.5 Conclusion

The Buffalo Crime Analysis Dashboard integrates data-driven visualizations and predictive modeling to offer valuable insights into urban crime patterns. By enabling users to explore historical crime data, identify high-crime areas and peak hours, and forecast potential incidents, the application serves as a powerful tool for both public safety officials and community members. Its intuitive design and advanced analytical features make it a versatile solution for crime analysis and prevention.

7 References

- Buffalo Open Data Portal `https://data.buffalony.gov/`
- Pydeck Documentation `https://deckgl.readthedocs.io/en/latest/`
- Plotly Documentation `https://plotly.com/python/`
- `xgboost` Documentation: `https://xgboost.readthedocs.io/`
- Streamlit Documentation: `https://docs.streamlit.io/`
- Plotly Visualization: `https://plotly.com/`
- Python 3.11 Documentation
- Streamlit Documentation: `https://docs.streamlit.io/`
- Pandas Documentation: `https://pandas.pydata.org/pandas-docs/stable/`
- Joblib Documentation: `https://joblib.readthedocs.io/en/latest/`
- Plotly Documentation: `https://plotly.com/python/`
- Scikit-learn Documentation: `https://scikit-learn.org/stable/`
- Prophet Documentation: `https://facebook.github.io/prophet/docs/`
- XGBoost Documentation: `https://xgboost.readthedocs.io/`
- Numpy Documentation: `https://numpy.org/doc/stable/`
- Requests Documentation: `https://docs.python-requests.org/en/latest/`
- Jupyter Documentation: `https://jupyter.org/documentation`