DIC Project Report (Phase 3)

1. Akanksh Gatla (akankshg) 50465101
2. Gnana Abhinay Vadlamudi (gnanaabh) 50496402
3. Sai Teja Chittumalla (schittum) 50496091

Phase 1:

a. Discuss the background of the problem leading to your objectives. Why is it a significant problem?

Background:

Crime is a pervasive social issue with far-reaching implications for the safety and well-being of communities. Analyzing crime data allows us to gain insights into the evolution of criminal activity, understand its regional disparities, and potentially inform policy decisions. The Unified Crime Reporting Statistics, compiled by the U.S. Department of Justice and the Federal Bureau of Investigation, provides a comprehensive dataset covering a wide range of years (1960 to 2019) and various crime categories.

Property crime, including burglary, larceny, and motor-related crimes, poses a significant financial and emotional burden on victims and communities. Violent crimes, such as assault, murder, rape, and robbery, endanger lives and impact the overall security of areas. Therefore, comprehensively studying these crime categories can provide valuable insights into their nature, causes, and possible preventive measures.

b. Explain the potential of your project to contribute to your problem domain. Discuss why this contribution is crucial?

The significance of this problem lies in its potential to address several critical issues:

1. Crime Prevention: By understanding the historical trends in crime rates, we can identify high-risk areas and periods, enabling law enforcement agencies to allocate resources more effectively.

2. Policy Formulation: Policymakers can use this analysis to develop evidence-based strategies to combat crime, ultimately creating safer communities.

3. Social Impact: Reducing crime rates can lead to improved quality of life, economic growth, and well-being for residents.

4. Community Awareness: Public access to crime data fosters transparency and awareness, empowering individuals to make informed decisions about their safety and contribute to crime prevention.

Things Can be done using this Dataset :

- **Predictive Analytics**: Crime data analysis can serve as a foundation for developing predictive models. Understanding historical trends in property and violent crimes can help data scientists create models that predict future crime rates, aiding law enforcement in proactive resource allocation and response strategies.

- **Time Series Analysis**: Time series techniques can be applied to analyze temporal patterns in crime rates. Data scientists can identify seasonality, trends, and cyclical behavior, providing law enforcement with insights into when certain types of crimes are most likely to occur.

- **Machine Learning for Crime Profiling**: Data scientists can leverage machine learning algorithms to profile criminals and analyze modus operandi. These profiles can assist in criminal investigations and developing strategies to prevent future offenses.

- **Data Visualization**: Data visualization techniques can help in creating interactive dashboards for law enforcement and policymakers. These dashboards enable real-time monitoring of crime trends and facilitate data-driven decision-making.

- **Public Engagement**: Utilizing data science to create accessible crime data visualizations can foster community engagement. Citizens can gain a better understanding of crime trends in their neighborhoods, empowering them to take preventive measures and cooperate with law enforcement.

2. Data Sources :

Dataset used in this project - 'https://corgis-edu.github.io/corgis/csv/state_crime/'
From the Unified Crime Reporting Statistics and under the collaboration of the U.S. Department of Justice and the Federal Bureau of Investigation information crime statistics are available for public review. The following data set has information on the crime rates and totals for states across the United States for a wide range of years. The crime reports are divided into two main categories: property and violent crime. Property crime refers to burglary, larceny, and motor related crime while violent crime refers to assault, murder, rape, and robbery. These reports go from 1960 to 2019.

3. Data Cleaning and Preprocessing

1. We use .isna and add them to know if there any na values in our dataframe.

```
null_values = project_df.isna().sum()
null_values
```

```
State                           41
Year                            41
Data.Population                 41
Data.Rates.Property.All         41
Data.Rates.Property.Burglary    41
Data.Rates.Property.Larceny     41
Data.Rates.Property.Motor       41
Data.Rates.Violent.All          41
Data.Rates.Violent.Assault      41
Data.Rates.Violent.Murder       41
Data.Rates.Violent.Rape         41
Data.Rates.Violent.Robbery      41
Data.Totals.Property.All        41
Data.Totals.Property.Burglary   41
Data.Totals.Property.Larceny    41
Data.Totals.Property.Motor      41
Data.Totals.Violent.All         41
Data.Totals.Violent.Assault     41
Data.Totals.Violent.Murder      41
Data.Totals.Violent.Rape        41
Data.Totals.Violent.Robbery     41
dtype: int64
```

2. We use dropna() to remove rows with null values from the DataFrame.

```
[ ] project_df = project_df.dropna()
```

```
[ ] project_df.shape
```

```
(3115, 21)
```

3. We use .describe to get the maximum,mean, std,minimum and other statistical values in our dataframe.

```
project_df.describe()
```

	Year	Data.Population	Data.Rates.Property.All	Data.Rates.Property.Burglary	Data.Rates.Property.Larceny	Data.Rates.Property.Motor	Data.Rates.Violent.All	Data.Rates.Violent.As
count	3115.000000	3.115000e+03	3115.000000	3115.000000	3115.000000	3115.000000	3115.000000	3115.0
mean	1989.544141	9.708502e+06	3542.202311	876.532520	2322.659133	343.011300	397.877047	237.3
std	17.299570	3.506750e+07	1418.191397	446.531611	897.934463	221.654068	287.498896	159.3
min	1960.000000	2.261670e+05	573.100000	126.300000	293.300000	28.400000	9.500000	3.6
25%	1975.000000	1.279156e+06	2472.650000	535.000000	1663.800000	185.600000	217.200000	124.0
50%	1990.000000	3.358000e+06	3438.400000	796.600000	2275.700000	288.900000	342.200000	205.1
75%	2005.000000	6.082836e+06	4439.100000	1133.850000	2877.500000	437.200000	518.250000	319.3
max	2019.000000	3.282395e+08	9512.100000	2906.700000	5833.800000	1839.900000	2921.800000	1557.6

4. Change column Data type using convert_dtypes()

```
project_df = project_df.convert_dtypes()
project_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 3115 entries, 0 to 3155
Data columns (total 21 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   State                           3115 non-null   string
 1   Year                            3115 non-null   Int64
 2   Data.Population                 3115 non-null   Int64
 3   Data.Rates.Property.All         3115 non-null   Float64
 4   Data.Rates.Property.Burglary    3115 non-null   Float64
 5   Data.Rates.Property.Larceny     3115 non-null   Float64
 6   Data.Rates.Property.Motor       3115 non-null   Float64
 7   Data.Rates.Violent.All          3115 non-null   Float64
 8   Data.Rates.Violent.Assault      3115 non-null   Float64
 9   Data.Rates.Violent.Murder       3115 non-null   Float64
 10  Data.Rates.Violent.Rape         3115 non-null   Float64
 11  Data.Rates.Violent.Robbery      3115 non-null   Float64
 12  Data.Totals.Property.All        3115 non-null   Int64
 13  Data.Totals.Property.Burglary   3115 non-null   Int64
 14  Data.Totals.Property.Larceny    3115 non-null   Int64
 15  Data.Totals.Property.Motor      3115 non-null   Int64
 16  Data.Totals.Violent.All         3115 non-null   Int64
 17  Data.Totals.Violent.Assault     3115 non-null   Int64
 18  Data.Totals.Violent.Murder      3115 non-null   Int64
 19  Data.Totals.Violent.Rape        3115 non-null   Int64
 20  Data.Totals.Violent.Robbery     3115 non-null   Int64
dtypes: Float64(9), Int64(11), string(1)
memory usage: 596.2 KB
```

5. We drop few columns which are not necessary can be added later , if needed.

```
[13] project_df.drop(['Data.Total.crime'], axis=1, inplace = True)
```

```
project_df
```

	State	Year	Data.Population	Data.Rates.Property.All	Data.Rates.Property.Burglary	Data.Rates.Property.Larceny	Data.Rates.Property.Motor	Data.Rates.Violent.All	Data.Rates.Violent.
0	Alabama	1960	3266740	1035.4	355.9	592.1	87.3	186.6	
1	Alabama	1961	3302000	985.5	339.3	569.4	76.8	168.5	
2	Alabama	1962	3358000	1067.0	349.1	634.5	83.4	157.3	
3	Alabama	1963	3347000	1150.9	376.9	683.4	90.6	182.7	
4	Alabama	1964	3407000	1358.7	466.6	784.1	108.0	213.1	
...	...	...	...	...	...	...	...	...	
3151	Wyoming	2015	586107	1902.6	300.6	1500.9	101.0	222.1	
3152	Wyoming	2016	585501	1957.3	302.5	1518.2	136.6	244.2	
3153	Wyoming	2017	579315	1830.4	275.0	1421.0	134.5	237.5	
3154	Wyoming	2018	577737	1785.1	264.0	1375.9	145.2	212.2	
3155	Wyoming	2019	578759	1571.1	241.2	1206.7	123.2	217.4	

3115 rows × 21 columns

6. Sanity checks:

```
Check if the data present is above 0 and delete rest rows if they are found negative.
Year has negative or zeros value :  0
Data.population has negative or zeros value :  0
Data.Rates.Property.All has negative or zeros value :  0
Data.Rates.Property.Burglary has negative or zeros value :  0
Data.Rates.Property.Motor has negative or zeros value :  0
Data.Rates.Violent.All has negative or zeros value :  0
Data.Rates.Violent.Assault has negative or zeros value :  0
Data.Rates.Violent.Murder has negative or zeros value :  0
Data.Rates.Violent.Rape has negative or zeros value :  0
Data.Rates.Violent.Robbery has negative or zeros value :  0
Data.Totals.Property.All has negative or zeros value :  0
Data.Totals.Property.Larceny has negative or zeros value :  0
Data.Totals.Property.Motor has negative or zeros value :  0
Data.Totals.Violent.All has negative or zeros value :  0
Data.Totals.Violent.Assault has negative or zeros value :  0
Data.Totals.Violent.Murder has negative or zeros value :  0
Data.Totals.Violent.Rape has negative or zeros value :  0
Data.Totals.Violent.Robbery has negative or zeros value :  0
```

7. Maintaining the same case , So that the different states can have one capital word and other small word, Might be a problem when we groupby and get sub datasets. That's why we use text mining.

7. Change the String to lower case.

```
project_df['State'] = project_df['State'].str.lower()
project_df
```

	State	Year	Data.Population	Data.Rates.Property.All	Data.Rates.Property.Burglary	Data.Rates.Property.Larceny	Data.Rates.Property.Motor	Data.Rates.Violent.All	Data.Rates.Violent.A
0	alabama	1960	3266740	1035.4	355.9	592.1	87.3	186.6	
1	alabama	1961	3302000	985.5	339.3	569.4	76.8	168.5	
2	alabama	1962	3358000	1067.0	349.1	634.5	83.4	157.3	
3	alabama	1963	3347000	1150.9	376.9	683.4	90.6	182.7	
4	alabama	1964	3407000	1358.7	466.6	784.1	108.0	213.1	
...	...	...	...	...	...	...	...	...	
3151	wyoming	2015	586107	1902.6	300.6	1500.9	101.0	222.1	
3152	wyoming	2016	585501	1957.3	302.5	1518.2	136.6	244.2	
3153	wyoming	2017	579315	1830.4	275.0	1421.0	134.5	237.5	
3154	wyoming	2018	577737	1785.1	264.0	1375.9	145.2	212.2	
3155	wyoming	2019	578759	1571.1	241.2	1206.7	123.2	217.4	

3115 rows × 21 columns

8. Rename the columns for a better idea of what columns we have and look at the features properly in rows.

8. Rename columns

```
column_mapping = {
    'Data.Population': 'Population',
    'Data.Rates.Property.All': 'Property_Rates_All',
    'Data.Rates.Property.Burglary': 'Property_Rates_Burglary',
    'Data.Rates.Property.Larceny': 'Property_Rates_Larceny',
    'Data.Rates.Property.Motor': 'Property_Rates_Motor',
    'Data.Rates.Violent.All': 'Violent_Rates_All',
    'Data.Rates.Violent.Assault': 'Violent_Rates_Assault',
    'Data.Rates.Violent.Murder': 'Violent_Rates_Murder',
    'Data.Rates.Violent.Rape': 'Violent_Rates_Rape',
    'Data.Rates.Violent.Robbery': 'Violent_Rates_Robbery',
    'Data.Totals.Property.All': 'Property_Totals_All',
    'Data.Totals.Property.Burglary': 'Property_Totals_Burglary',
    'Data.Totals.Property.Larceny': 'Property_Totals_Larceny',
    'Data.Totals.Property.Motor': 'Property_Totals_Motor',
    'Data.Totals.Violent.All': 'Violent_Totals_All',
    'Data.Totals.Violent.Assault': 'Violent_Totals_Assault',
    'Data.Totals.Violent.Murder': 'Violent_Totals_Murder',
    'Data.Totals.Violent.Rape': 'Violent_Totals_Rape',
    'Data.Totals.Violent.Robbery': 'Violent_Totals_Robbery'
}

project_df = project_df.rename(columns=column_mapping)
project_df.columns
```

```
Index(['State', 'Year', 'Population', 'Property_Rates_All',
       'Property_Rates_Burglary', 'Property_Rates_Larceny',
       'Property_Rates_Motor', 'Violent_Rates_All', 'Violent_Rates_Assault',
       'Violent_Rates_Murder', 'Violent_Rates_Rape', 'Violent_Rates_Robbery',
       'Property_Totals_All', 'Property_Totals_Burglary',
       'Property_Totals_Larceny', 'Property_Totals_Motor',
```

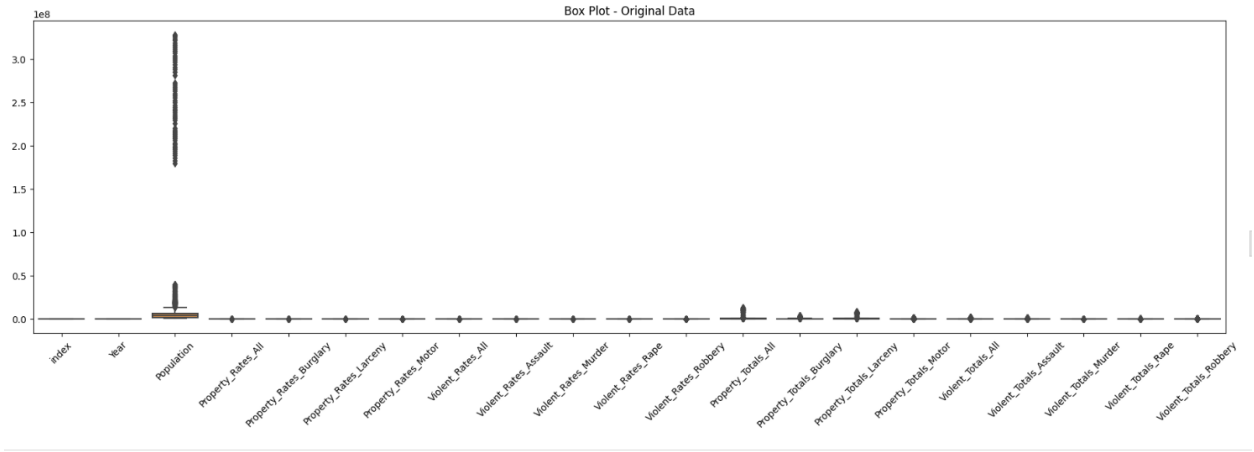
9. As we have done lots of cleaning and preprocessing of the data , we have messed up with the index , to make it proper we use reset_index.

9. Reset index

```
project_df = project_df.reset_index()
```

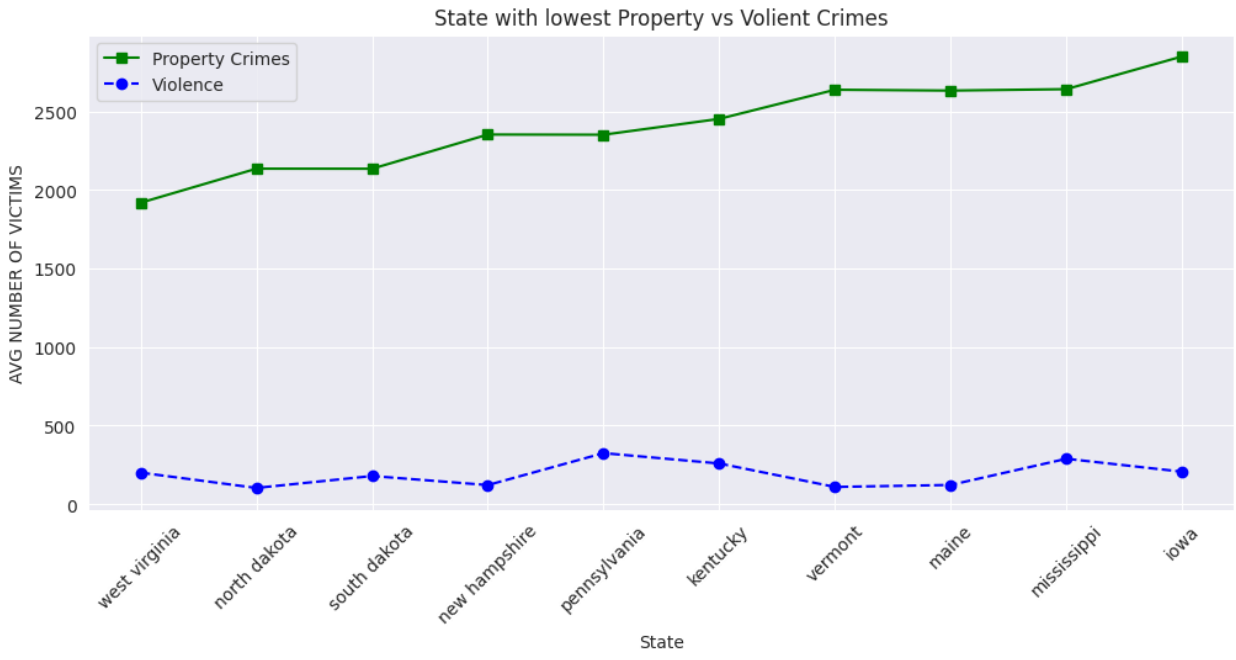
10. Outliers are the data points which are far away from the usual data and need to be ignored or mostly eliminated . In my dataset Population is the most important column , we can just eliminate that , if we remove that it simply means no live person in that year. So we ignore it .

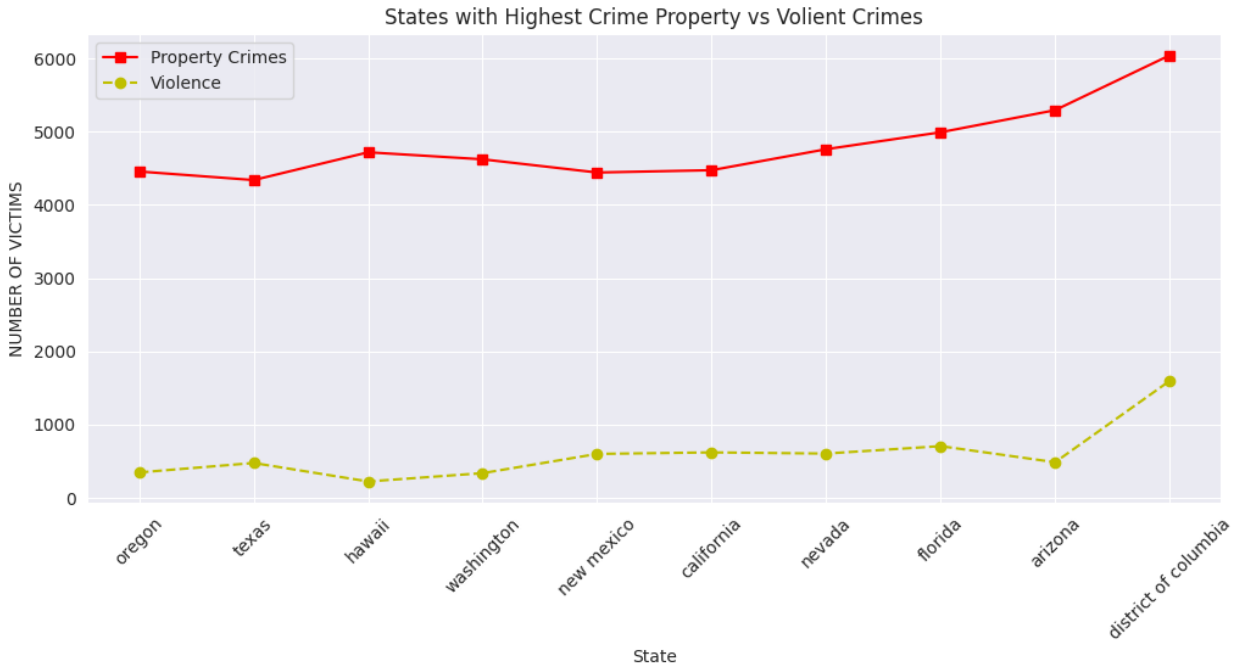
```
plt.subplot(1, 2, 1)
numeric_columns = project_df.select_dtypes(include=['int64', 'float64'])
sns.boxplot(data=numeric_columns, orient='vertical')
plt.xticks(rotation=45)
plt.title('Box Plot - Original Data');
```



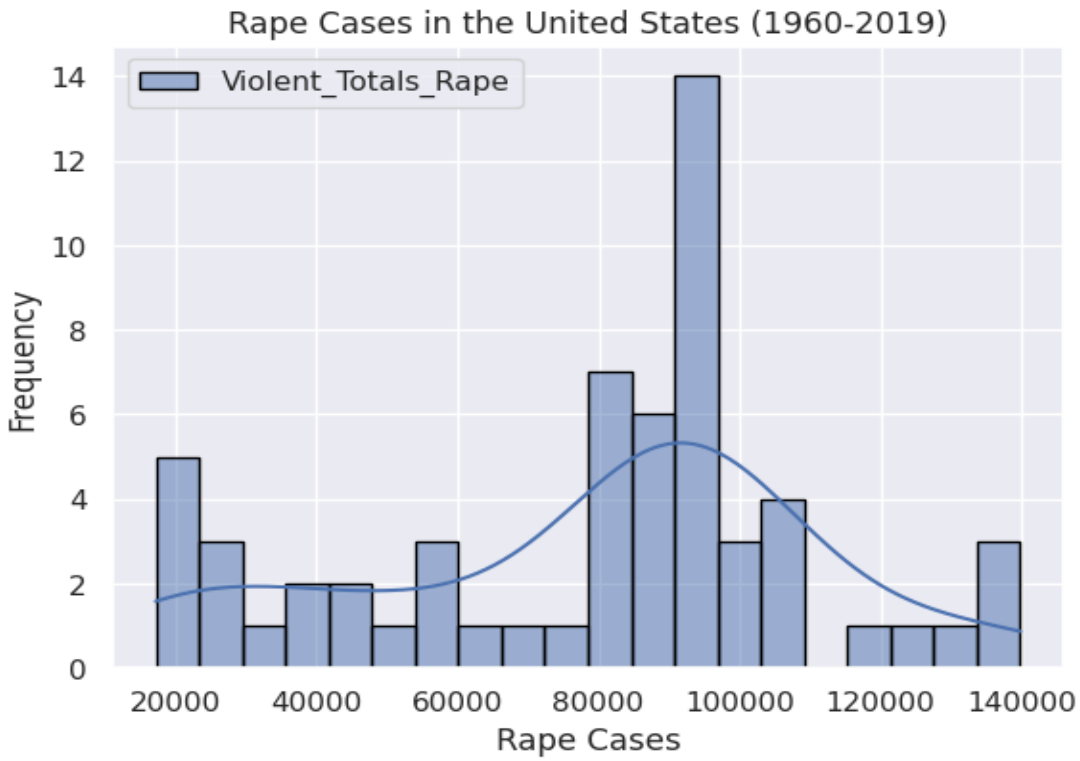
4. Exploratory Data Analysis (EDA):

a. Determine which states consistently have the highest and lowest crime rates across all years. [**LINE Graph**]



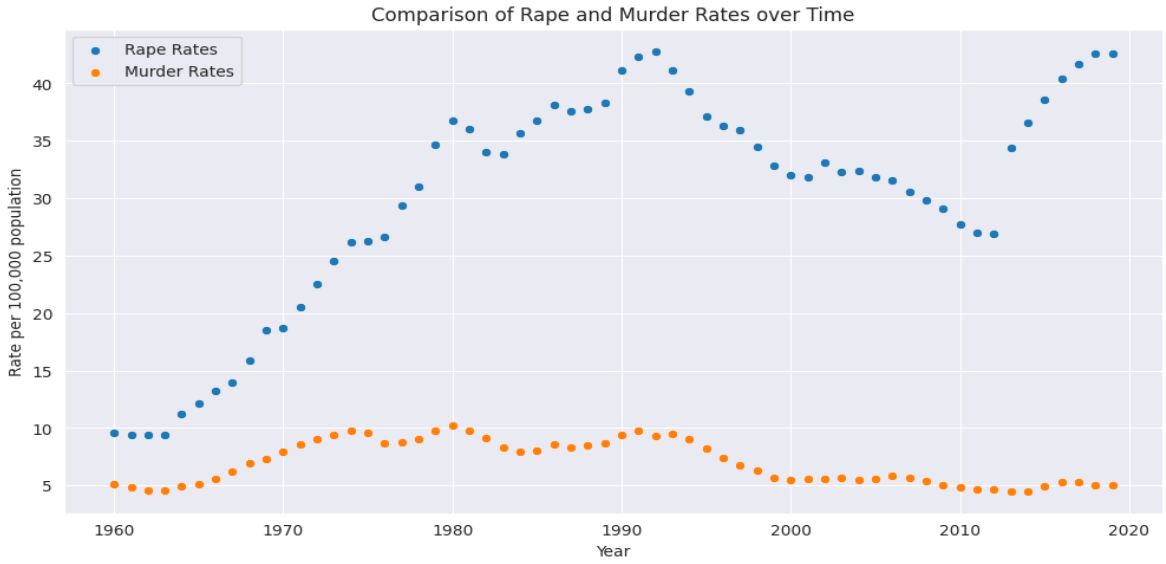


b. Time series on Rapes in United States using Histogram? (univariate analysis). [**Histogram**]



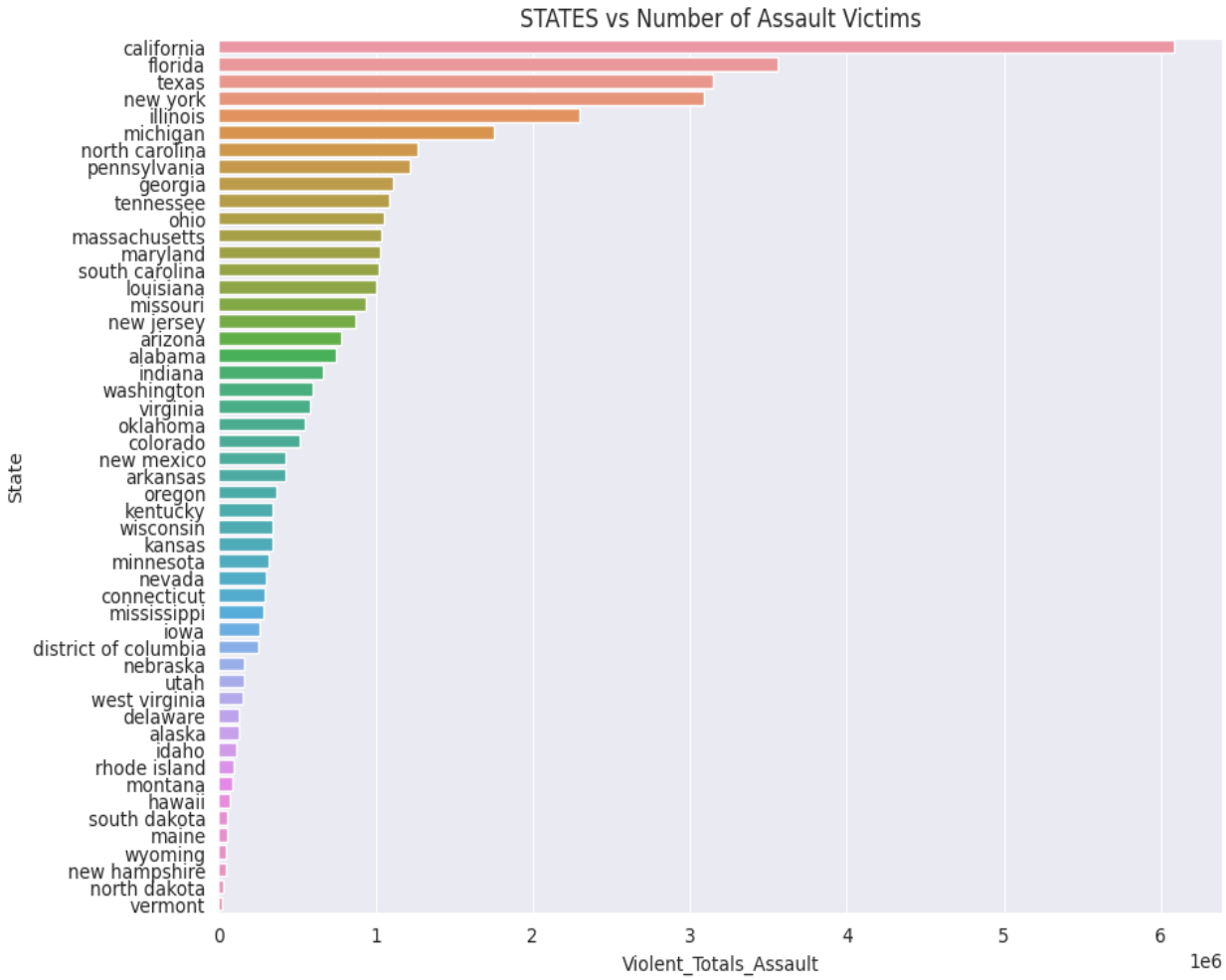
This is histogram showing the distribution of rape cases in the United States, with data grouped into 20 bins.

c. Compare Assault vs Murder cases in the United States? [**Scatter Plot**]



- We filter the DataFrame to select the columns related to the year, rape rates, and murder rates.
- We create a scatter plot using Seaborn, plotting rape rates and murder rates on the same chart.
- We set the figure size, style, labels, and title to make the plot informative and visually appealing.
- This will generate a scatter plot that shows the trends in rape and murder rates over time, allowing you to visually compare these two types of crimes.

d. Compare Rape cases in all States between 1960-2019? (Bi- variate) [**Bar Graph**]

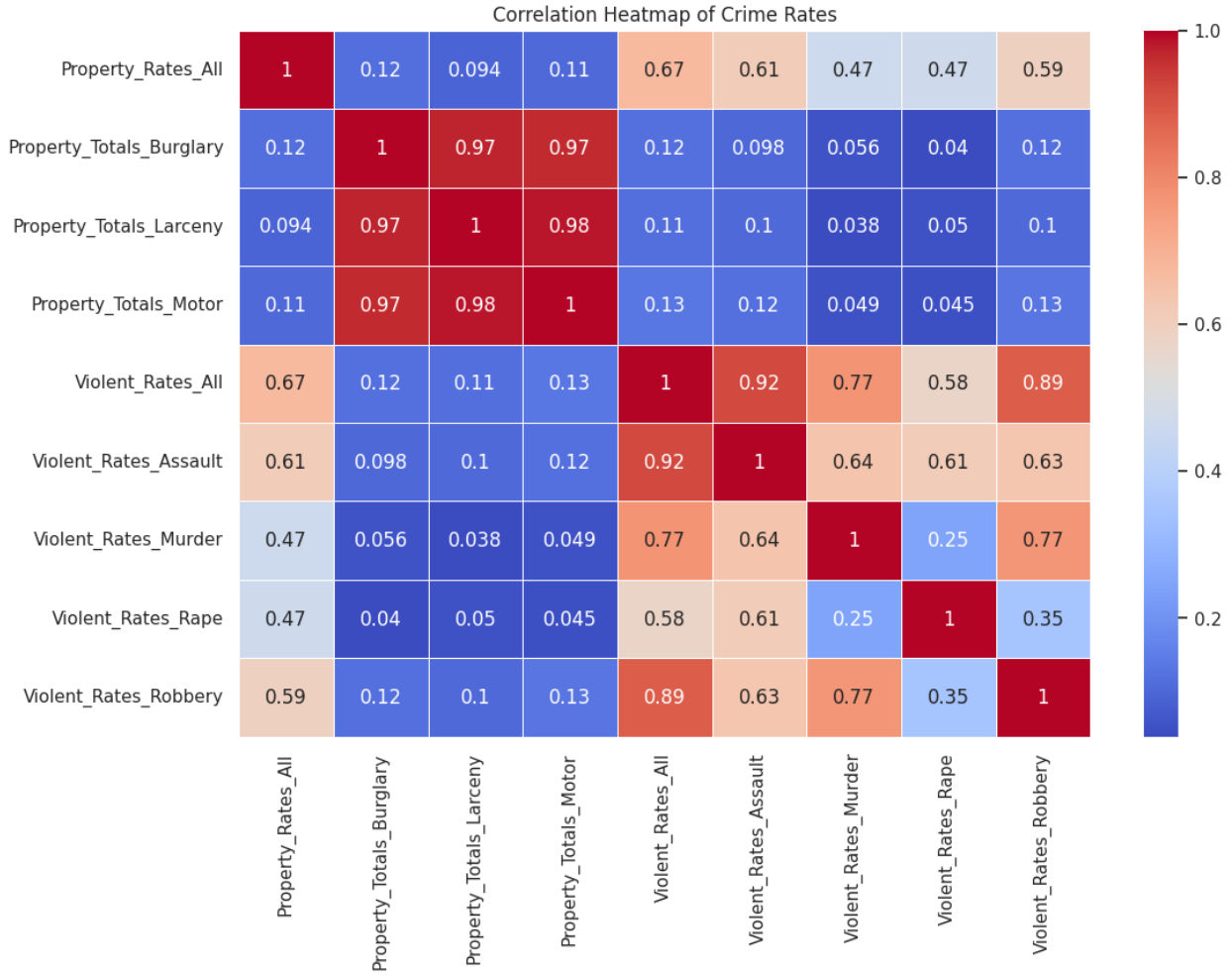


Bar plot showing the number of assault victims in different states, with states sorted in descending order based on the number of assault victims.

e. Compare Property vs Violent crimes between all the states?[**Line Plot**]

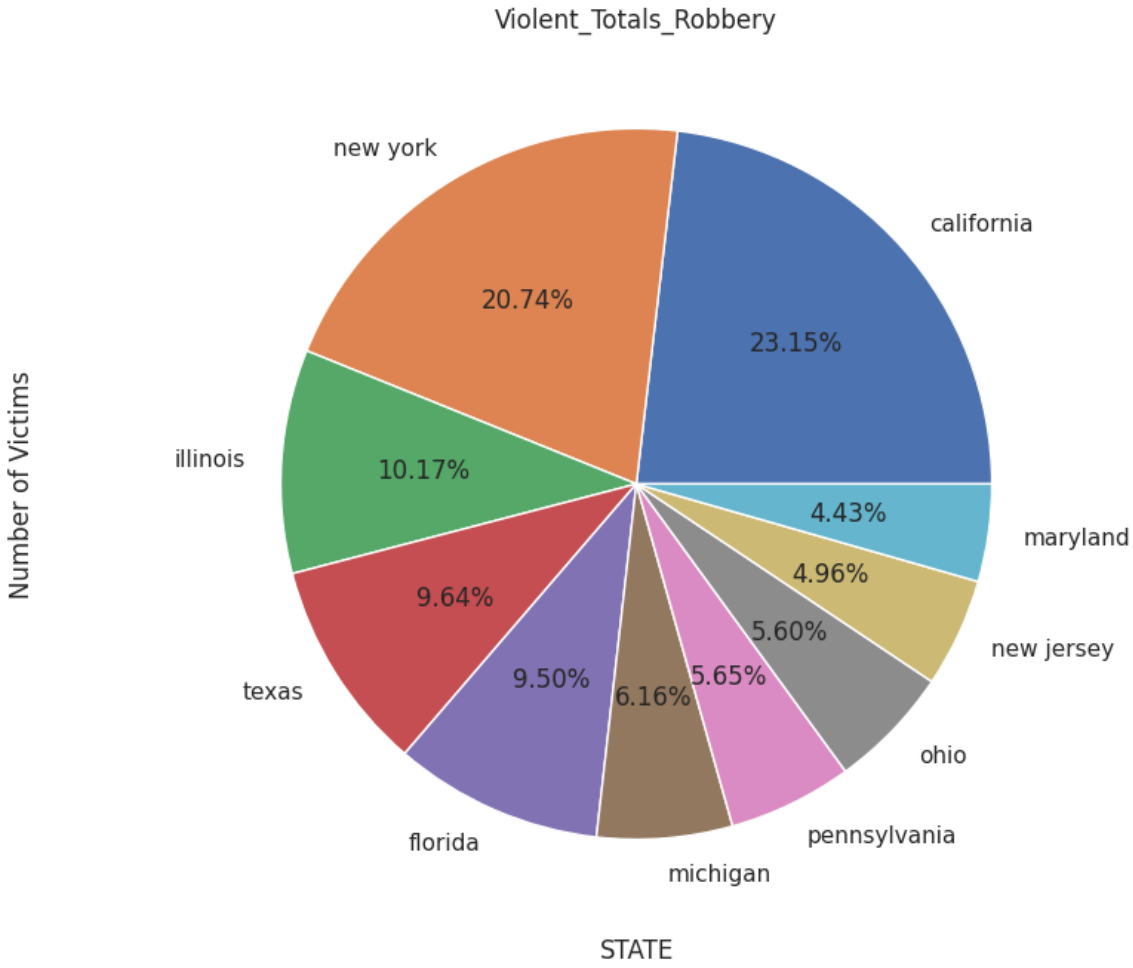
This code creates a line plot comparing the total property crime victims (Burglary, Larceny, and Motor) and total violent crime victims (Assault, Murder, Rape, and Robbery) for different states in US.

f. Correlations between different types of crime rates over the years. [**Heat Map**]



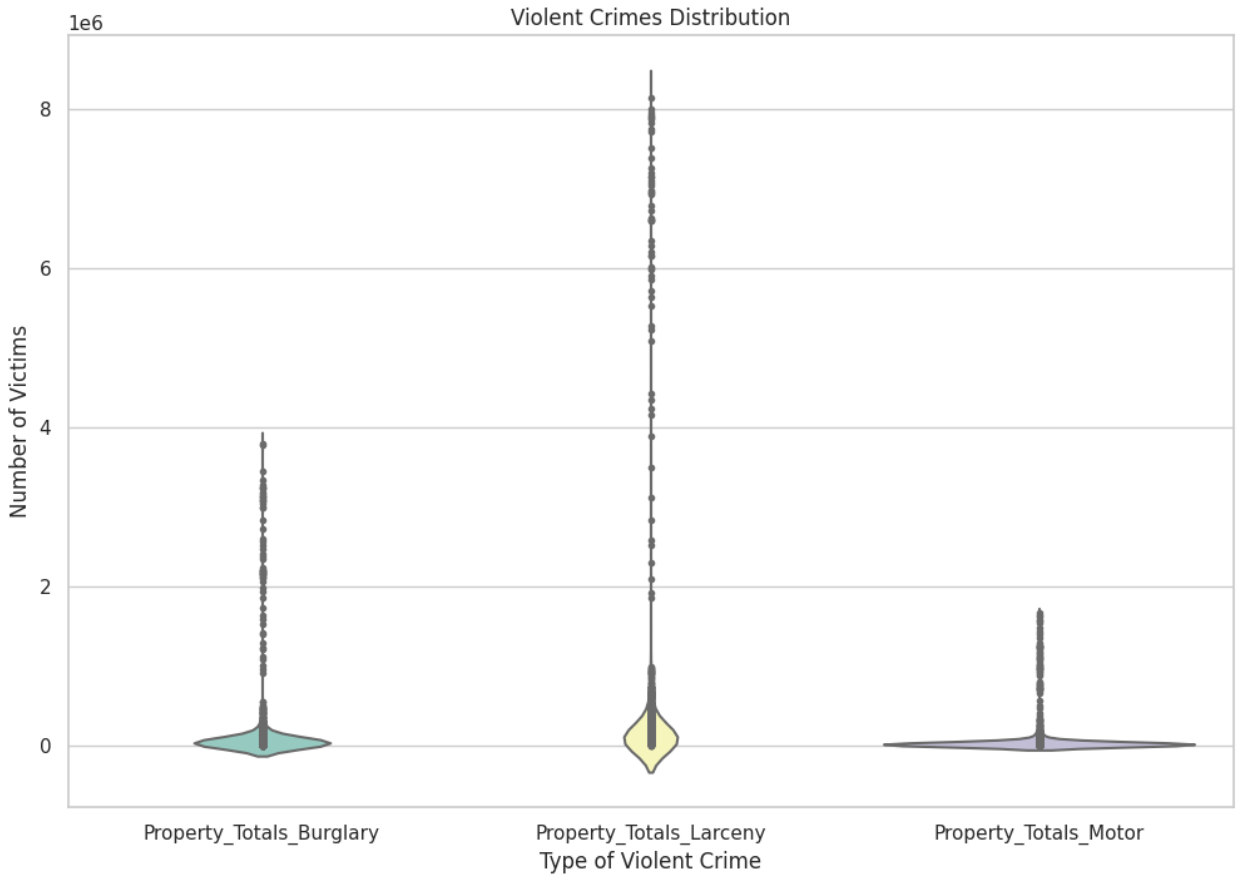
- We select the relevant columns for crime rates, including property and various violent crime rates.
- We set the 'Year' column as the index to use it as the x-axis of the heatmap.
- We create a heatmap using Seaborn, displaying the correlation between different crime rates. The annot=True parameter adds numerical values to the heatmap cells to indicate the strength of the correlation.
- We use the 'coolwarm' colormap for the heatmap for better visualization of correlations.
- This will create a heatmap that shows the correlations between different types of crime rates over the years. You can adjust the columns you want to include in the analysis or use different colormaps and styling options to customize the visualization based on your specific requirements.

g. Top 10 states with Robbery Crime. [**Pie Chart**]



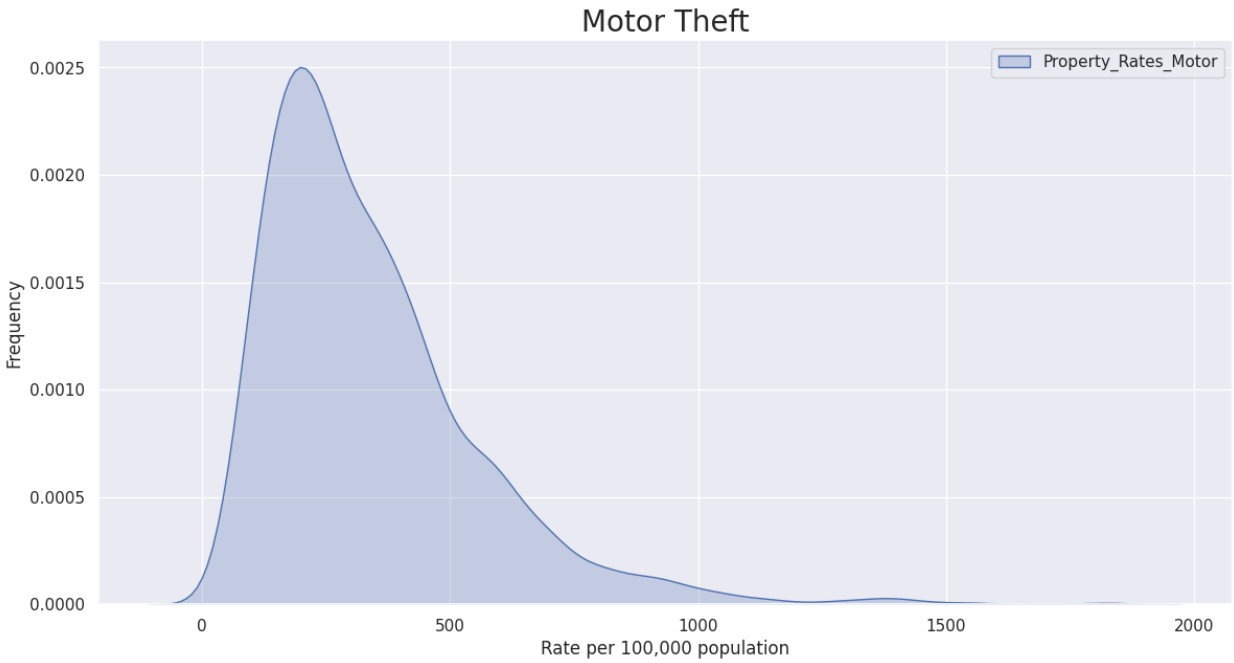
a pie chart showing the distribution of robbery victims in the top 10 states.

h. Distribution of victims for different types of Property crimes.[**Violin Plot**]



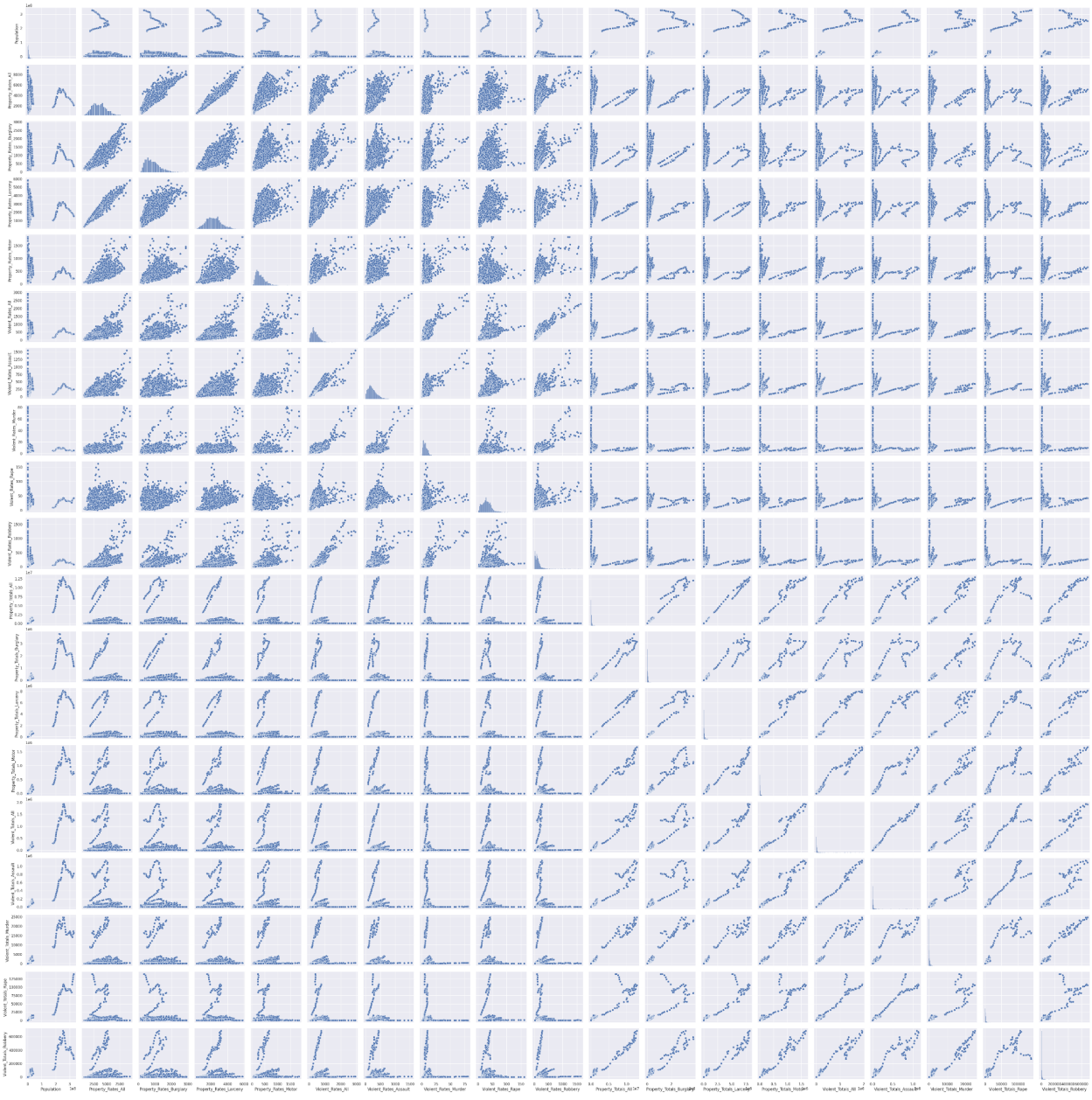
- We select the columns related to different types of violent crimes (Assault, Murder, Rape, and Robbery).
- We create a violin plot using Seaborn, specifying an "inner" parameter as "points" to display individual data points within the violin plot.
- We set the figure size, style, labels, and title for the plot.
- This code will create a violin plot that shows the distribution of victims for different types of violent crimes, allowing you to compare the distribution and the central tendencies for each crime category.

I. Motor theft across over years.[**Kde plot**]



- We've extracted the 'Property_Rates_Motor' column as a Series.
- We've used fill=True directly within the kdeplot function, which is the correct way to specify that you want to fill the area under the curve.
- This code will create a kernel density plot for the 'Property_Rates_Motor' data with the specified color and fill.

J. **Pair Plot** (Relationship and correlation between all the Variables.)



- columns you want to include in the pair plot using the DataFrame indexing.
- You create the pair plot using sns.pairplot().
- a pair plot that visualizes the relationships between the selected variables, providing insights into their pairwise correlations and distributions.

Phase 2:

Why is it a significant problem?

Background:

Crime is a pervasive social issue with far-reaching implications for the safety and well-being of communities. Analyzing crime data allows us to gain insights into the evolution of criminal activity, understand its regional disparities, and potentially inform policy decisions. The Unified Crime Reporting Statistics, compiled by the U.S. Department of Justice and the Federal Bureau of Investigation, provides a comprehensive dataset covering a wide range of years (1960 to 2019) and various crime categories.

Property crime, including burglary, larceny, and motor-related crimes, poses a significant financial and emotional burden on victims and communities. Violent crimes, such as assault, murder, rape, and robbery, endanger lives and impact the overall security of areas. Therefore, comprehensively studying these crime categories can provide valuable insights into their nature, causes, and possible preventive measures.

2. Data Sources :

Dataset used in this project - 'https://corgis-edu.github.io/corgis/csv/state_crime/'

From the Unified Crime Reporting Statistics and under the collaboration of the U.S. Department of Justice and the Federal Bureau of Investigation information crime statistics are available for public review. The following data set has information on the crime rates and totals for states across the United States for a wide range of years. The crime reports are divided into two main categories: property and violent crime. Property crime refers to burglary, larceny, and motor related crime while violent crime refers to assault, murder, rape, and robbery. These reports go from 1960 to 2019.

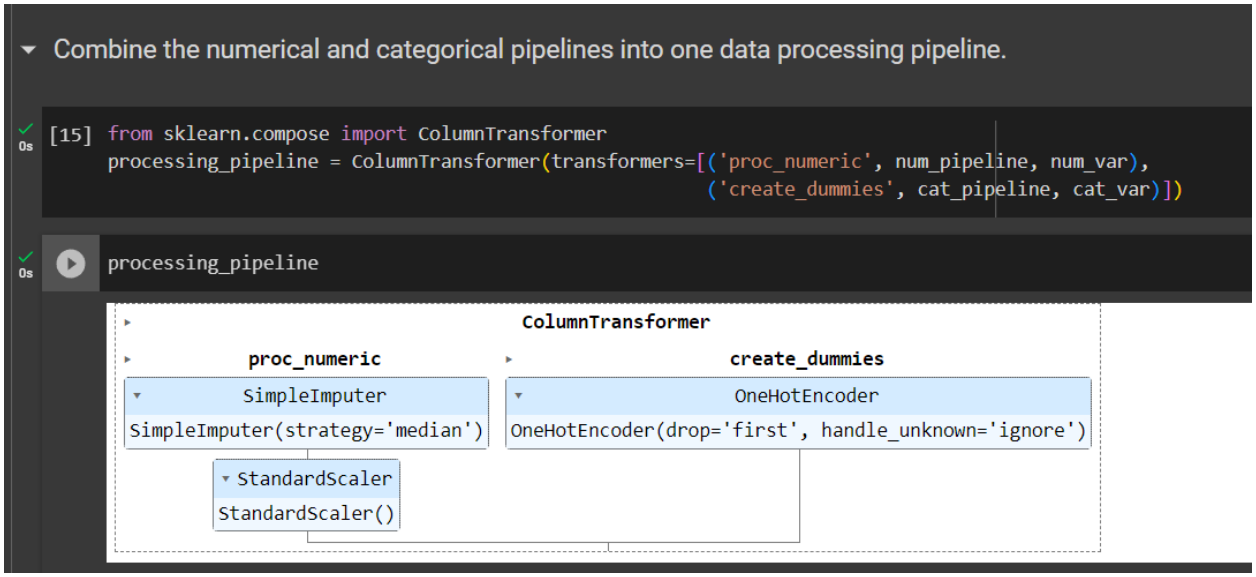
1. Data Description:
   The dataset contains information on crime rates and totals in different states and years.
   Columns include State, Year, Population, Rates (property and violent), and Totals (property and violent for various crime types).

2. Data Preparation:
   Selected a subset of columns as features (X) for the machine learning model.
   Defined the target variable (y) as the sum of specific columns representing different types of crime totals.

3. Train-Test Split:
   Used train_test_split from scikit-learn to split the data into training and testing sets.

Records in training data: 2,180
Records in test data: 935

4. Created Pipeline using one hot encoder to handle different types of string or missing errors on column transformers.



A. Linear Regression:

a. Why Linear Regression?

1. **Interpretability:**

Linear Regression provides a clear interpretation of the relationship between the independent variables (features) and the dependent variable (target).

2. **Simplicity:**

It's a good starting point, especially when exploring the dataset or as a baseline model before considering more complex algorithms.

3. **Assumptions:**

Assumptions like normality of residuals and homoscedasticity can be assessed and addressed if needed.

b. How applied?

Model Training:

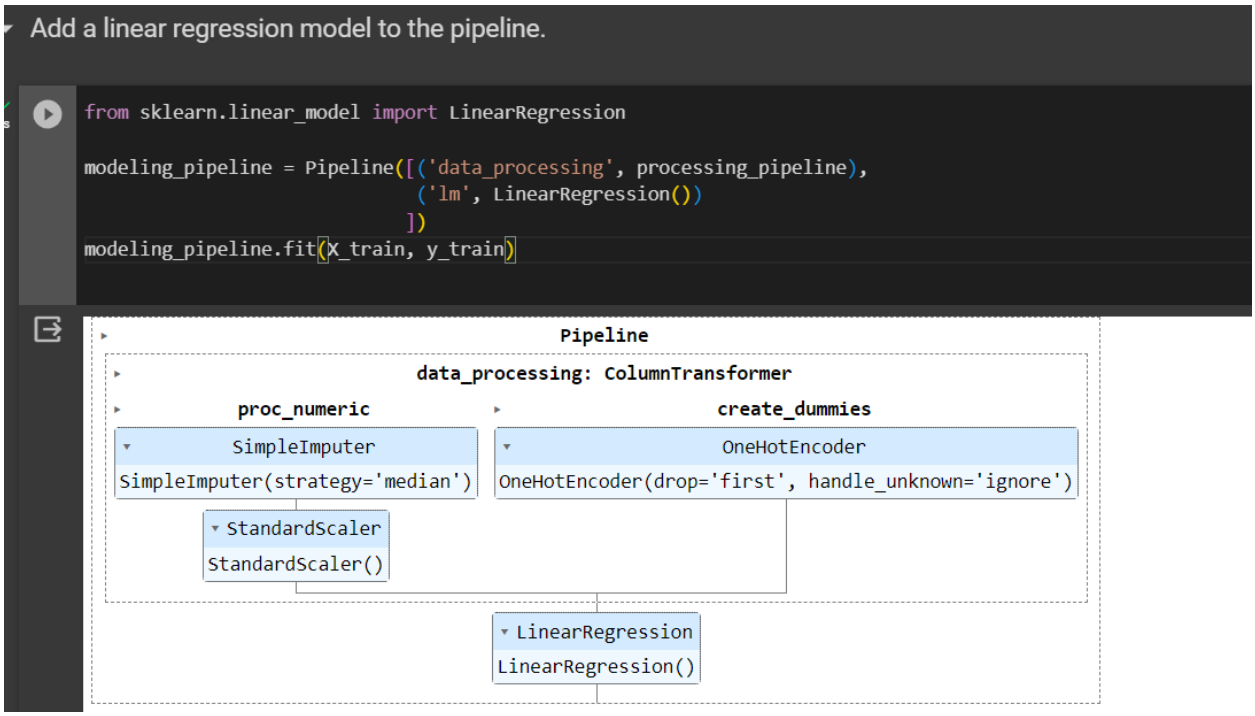
Utilized the scikit-learn library to create a Linear Regression model.

Fitted the model to the training data, allowing it to learn the relationships between features and the target variable.

Expectations:

Coefficients from the Linear Regression model offer insights into the importance and direction of each feature's impact on crime totals.

Linear Regression is chosen for its interpretability, simplicity, and suitability for predicting continuous target variables. It serves as a foundational model in the initial stages of exploring the relationship between crime features and totals. The choice is made with an awareness of the model's assumptions and a willingness to refine the approach based on insights gained during the modeling process.

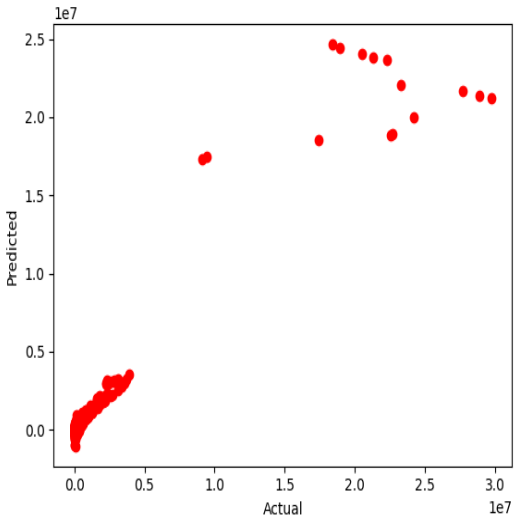


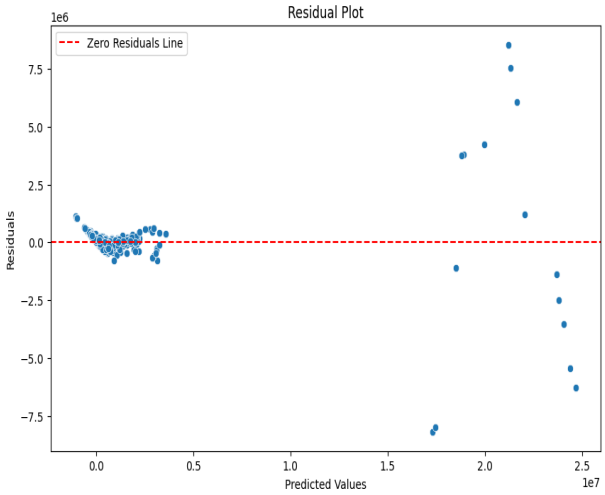
As we have already created the pipeline using that we just add it to our model. Next we predict the model and note the R^2 and mse values .

R^2 value for Linear Regression is : 93.37549997489647

MSE : 503108771106.021

RMSE: 709301.6079962184





B. **Ridge Regression**

Ridge Regression is chosen to address potential overfitting and multicollinearity in the crime prediction model. The iterative exploration of alpha values provides a nuanced understanding of how regularization impacts feature coefficients and overall model performance. This approach aims to enhance the interpretability and generalization of the predictive model.

a. Why Ridge Regression?

Regularization:

1. Ridge Regression is selected for its regularization property, which helps prevent overfitting by penalizing large coefficients. In the context of crime prediction, it's common to have multiple features that might be correlated. Ridge Regression can handle multicollinearity by shrinking the coefficients.
2. Alpha Parameter: The choice of Ridge Regression involves tuning the regularization parameter (alpha).The code explores various alpha values to understand the impact of regularization strength on model performance and coefficients.

Comparison with Linear Regression:

Ridge Regression is used as an extension or improvement upon simple Linear Regression. It addresses some of the limitations of Linear Regression, making it a more robust model in scenarios with potential multicollinearity.

b. How Applied:

Pipeline Setup:

The Ridge Regression model is integrated into a scikit-learn pipeline along with a data processing step.

Model Training:

The pipeline is trained on the training set (X_train and y_train). The regularization parameter (alpha) is set to 100 initially, and the model is trained.

Grid Search Over Alphas:

A loop is conducted over a range of alpha values (0, 1, 2, 5, 10, 50) to explore different levels of regularization. For each alpha, a Ridge Regression model is trained, and coefficients, training score, and test score are recorded.

Results Visualization:

Coefficients for each alpha are visualized to observe how regularization affects feature importance. The training and test scores are plotted against different alpha values to understand the model's performance under different levels of regularization.

Expectations:

Overfitting Control:

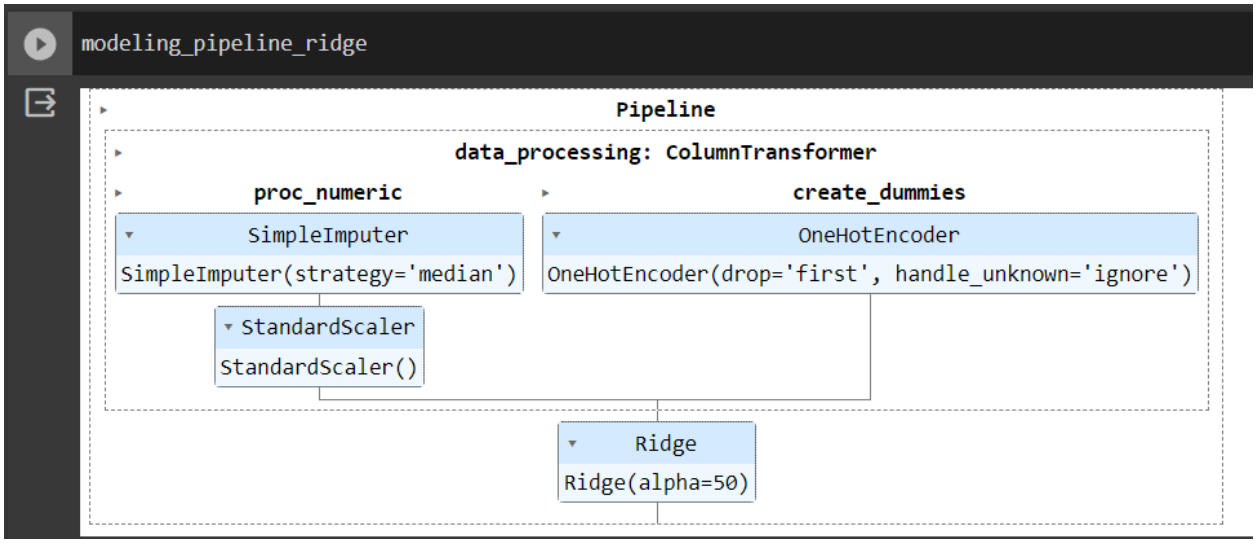
Ridge Regression is chosen with the expectation that it will help control overfitting, especially in the presence of correlated features.

Alpha Tuning:

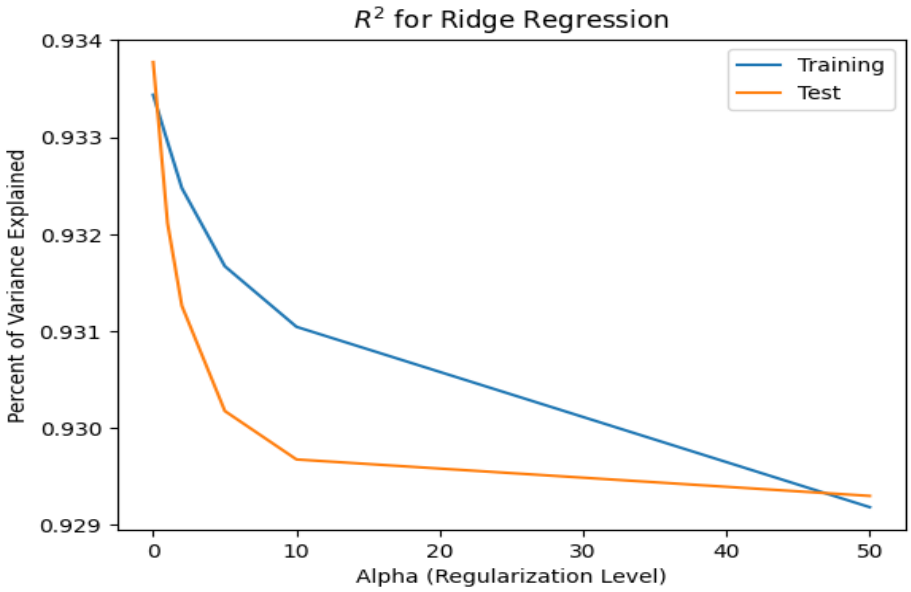
The iteration over different alpha values provides insights into the optimal level of regularization for the specific problem.

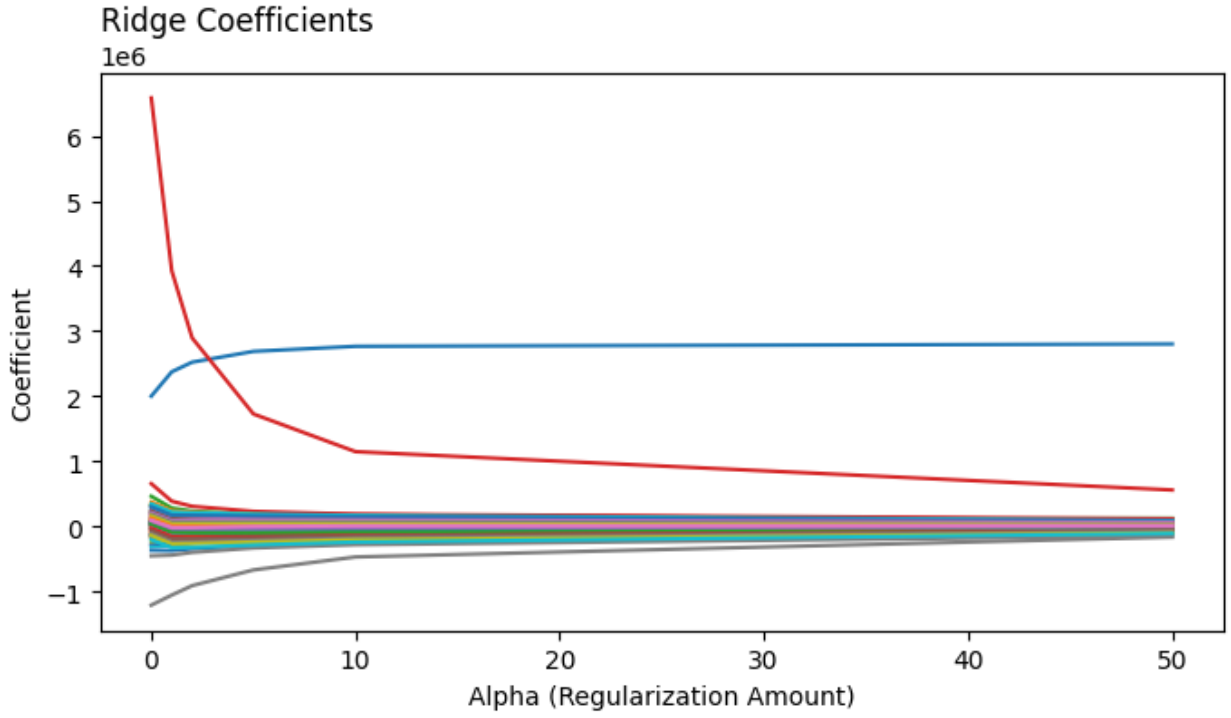
Model Evaluation:

The visualization of training and test scores over different alphas aids in assessing model performance and choosing a suitable regularization parameter.



R^2 for this model : 92.92975392202159





C. **Random Forest**

Random Forest is chosen for its ability to handle complex relationships, provide insights into feature importance, and offer robustness to overfitting. The choice is made with an awareness of the specific characteristics of the dataset and the problem at hand, aiming to enhance predictive performance and gain meaningful insights into the factors influencing crime totals.

a. Why Random Forest?

Complex Relationships:

Random Forest is chosen when the relationships between features and the target variable are expected to be more complex than what a linear model can capture.

It can handle non-linear relationships and interactions between features effectively.

Robustness to Overfitting:

Random Forest is less prone to overfitting compared to individual decision trees. The ensemble nature of Random Forest helps generalize well to unseen data.

Ensemble Method:

By combining multiple decision trees, Random Forest reduces the risk of individual trees making biased predictions, leading to a more robust model.

b. How Applied:

Exploration of Interactions:

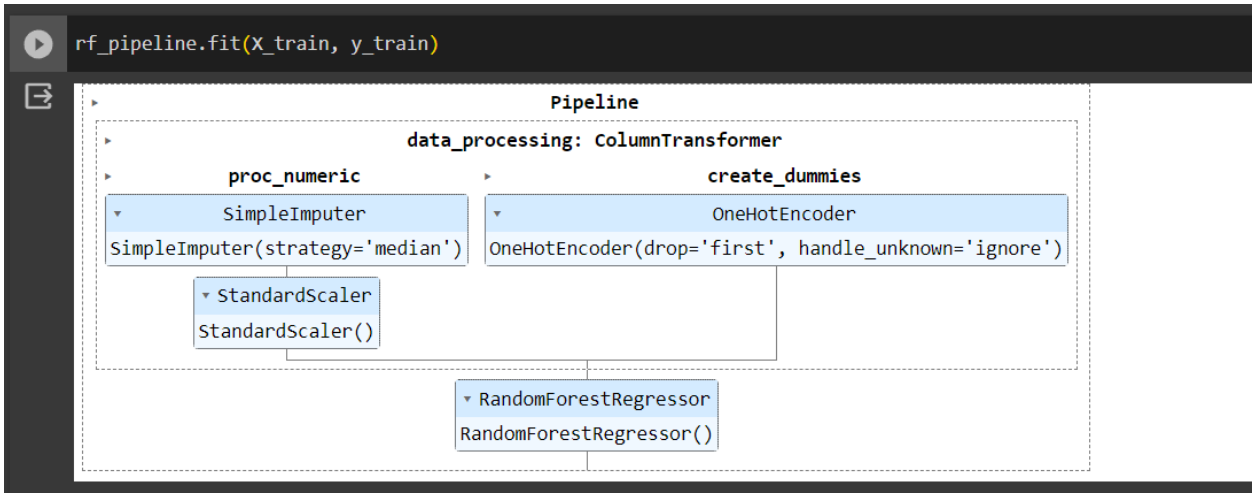
Random Forest is expected to capture interactions between different types of crimes (property vs. violent) and their rates more effectively than a linear model.

Hyperparameter Tuning:

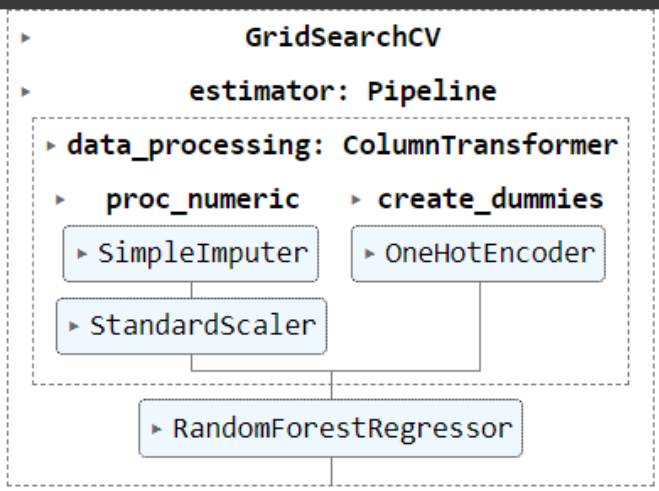
Utilized GridSearchCV to find the optimal hyperparameters for the Random Forest model.

c. Expectations:

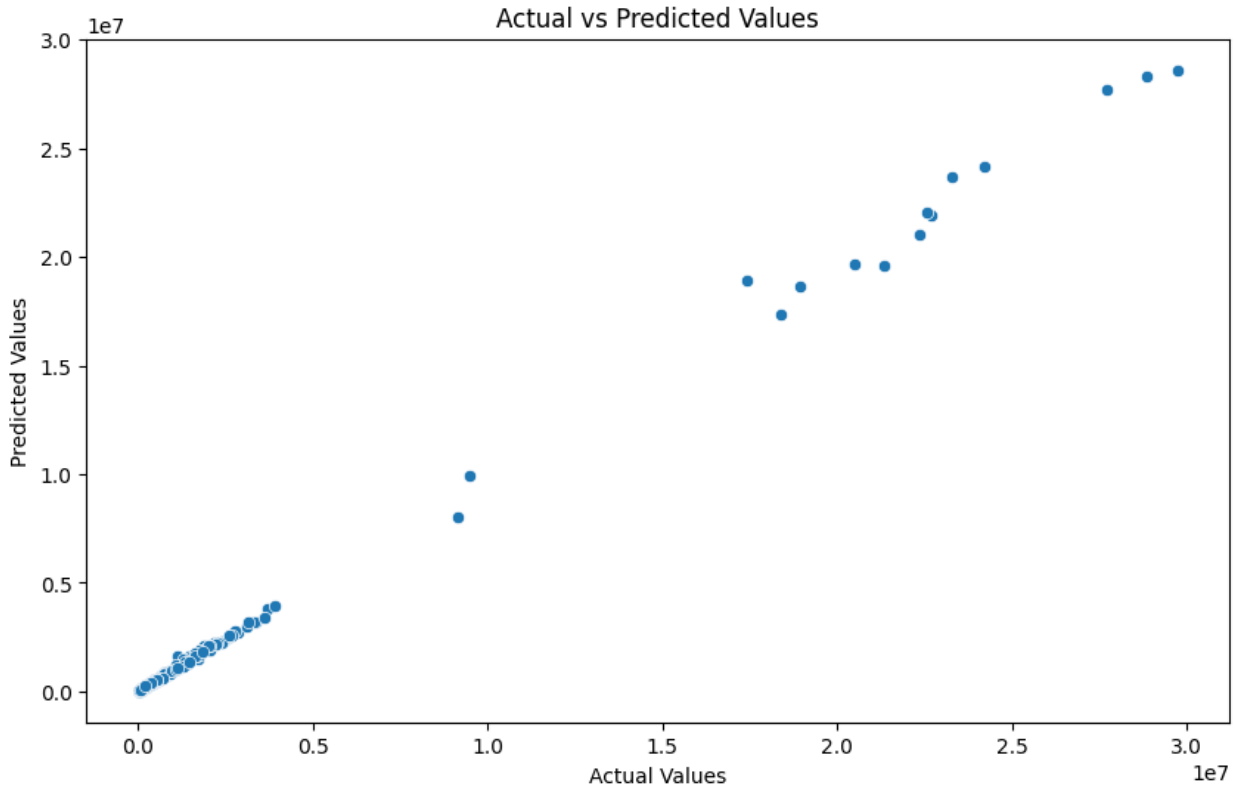
Extracted and analyzed feature importance to gain insights into which features contribute most to the prediction of crime totals. Learning: Understanding the relative importance of different crime-related features in the context of the problem.

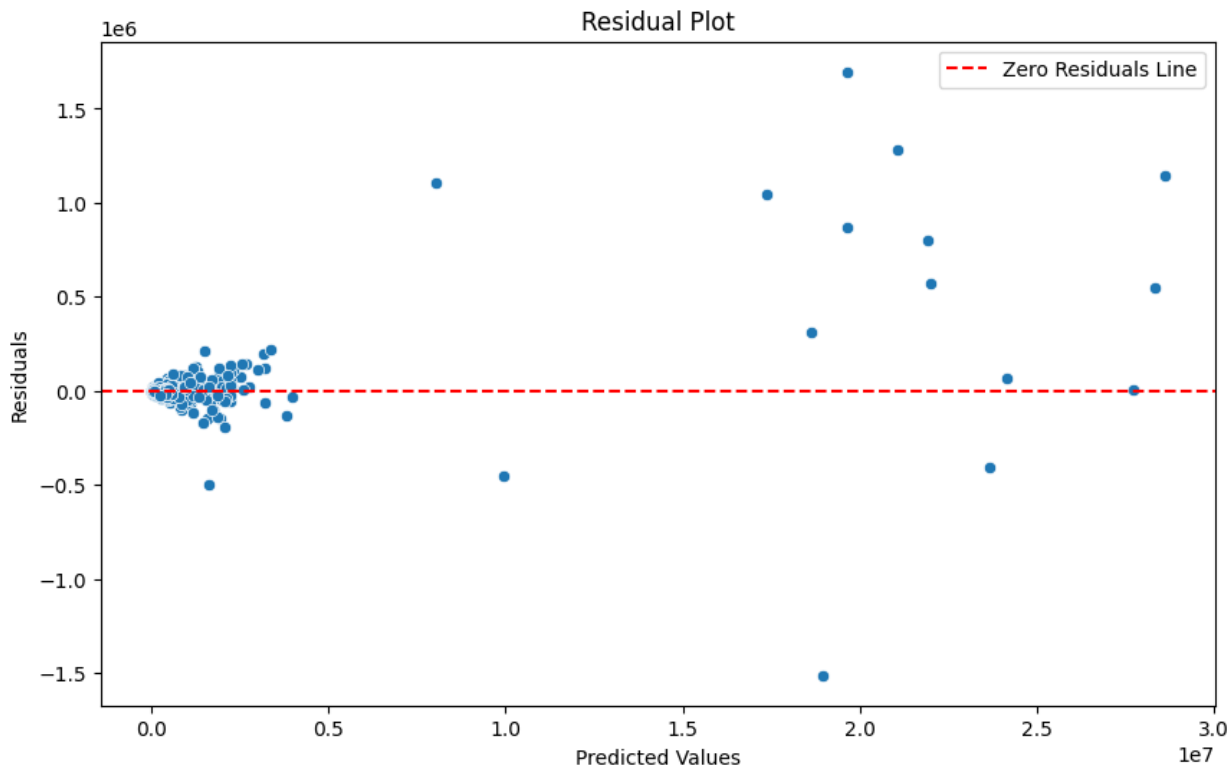


```
[35] grid_search = GridSearchCV(rf_pipeline, param_
     grid_search.fit(X_train, y_train)
```



Best Hyperparameters: {'rf__max_depth': 10, 'rf__min_samples_leaf': 1, 'rf__min_samples_split': 2, 'rf__n_estimators': 100}
Mean Squared Error on Test Set: 14977605963.532991
R-squared on Test Set: 99.80278787256422





D. **Decision Tree**

Decision Tree Regression is chosen for its ability to capture non-linear patterns, provide feature importance insights, and offer an intuitive decision-making process. The decision to use this model is made with an awareness of its strengths and considerations, and the evaluation metrics help assess its performance in the context of predicting crime totals.

a. Why Decision Tree?

Non-linearity in Relationships:

Decision Tree models are capable of capturing non-linear relationships between features and the target variable. Unlike Linear Regression, they can model complex interactions.

Handling Non-Normal Distributions:

Decision Trees don't assume a particular distribution of the data. This makes them robust to non-normal distributions, which might be present in crime rate data.

Intuitive Decision Making:

The decision-making process of a Decision Tree is easy to understand and interpret, making it valuable for providing insights into the factors influencing crime totals.

b. How applied?

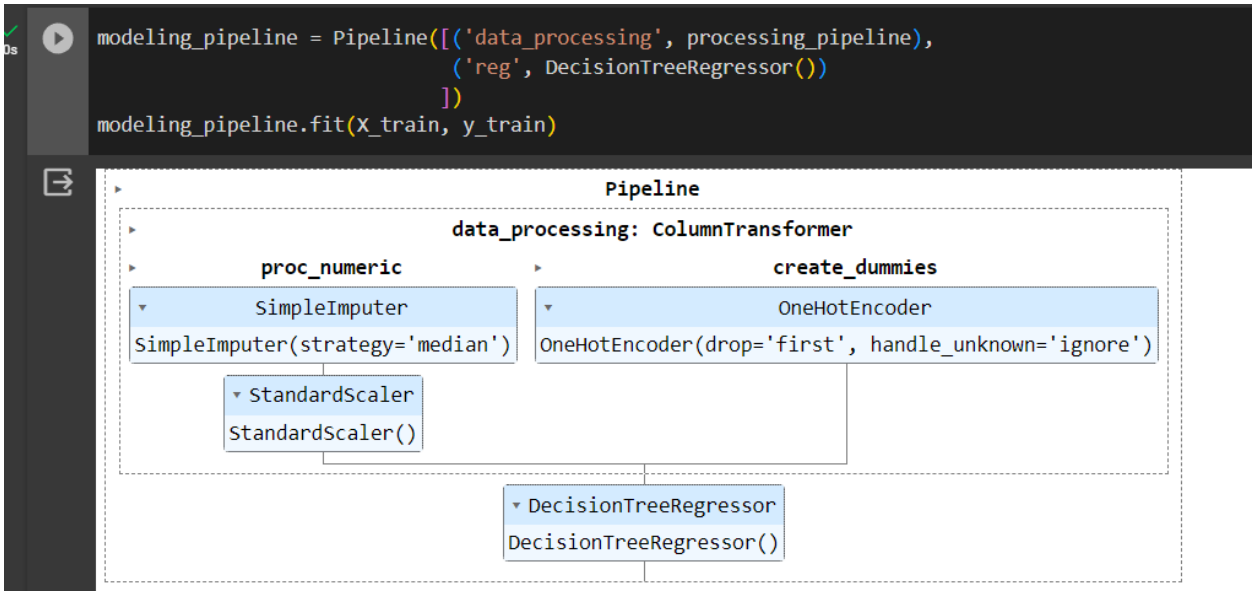
Pipeline Integration:

Integrated the Decision Tree model into a processing pipeline, allowing for consistent and reproducible data transformations.

Model Training:
Utilized the DecisionTreeRegressor from scikit-learn as the regression algorithm within the pipeline. Trained the model on the training set to learn the relationships between the features and the target variable.

Evaluation Metrics:
Used common regression metrics like R-squared, Mean Absolute Error (MAE), Mean Squared Error (MSE), Median Absolute Error, and Root Mean Squared Error (RMSE) to evaluate model performance.

c. **Expectation**
Anticipate gaining insights into the importance of different features in predicting crime totals. Expect Decision Tree Regression to capture non-linear patterns in the data, which may be present in crime rate variations



Mean_absolute_error: 43597.15294117647
Mean_squared_error: 66589717483.71551
Median_absolute_error: 8388.0
Root_Mean_Squared_error: 258049.83527163026
R_Square : 99.12320434371924

E. **KNeighborsRegressor**

KNeighborsRegressor was chosen for its non-linear nature and adaptability to complex data patterns. The application of this model was aimed at capturing localized relationships within the crime dataset, and the evaluation process provided insights into its performance and trade-offs.

a. Why KNeighborsRegressor?

Non-Linear Relationships:

KNeighborsRegressor is a non-linear model, which makes it suitable for capturing complex relationships in the data. Unlike Linear Regression, it does not assume a linear relationship between the features and the target variable.

Localized Predictions:

KNeighborsRegressor makes predictions based on the values of the k-nearest neighbors in the feature space. This can be advantageous when there are localized patterns in the data that may not be captured well by a global model like Linear Regression.

Adaptability to Data Distribution:

KNeighborsRegressor adapts well to the underlying data distribution without making strong assumptions about the functional form of the relationship. It can handle cases where the relationship between features and the target variable is not strictly linear.

b. How Applied:

Pipeline Integration:

Incorporated KNeighborsRegressor into a modeling pipeline along with data processing steps.Utilized a pipeline for consistency in preprocessing and modeling, ensuring a streamlined and reproducible workflow.

Model Training:

Fitted the KNeighborsRegressor model to the training data, allowing it to learn the relationships between features and the target variable.

c. Expectations:

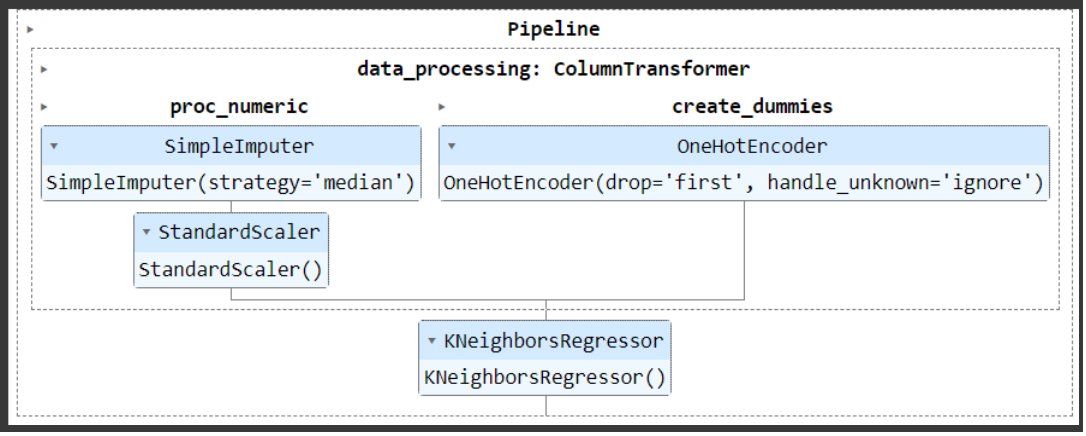
Capturing Localized Patterns:

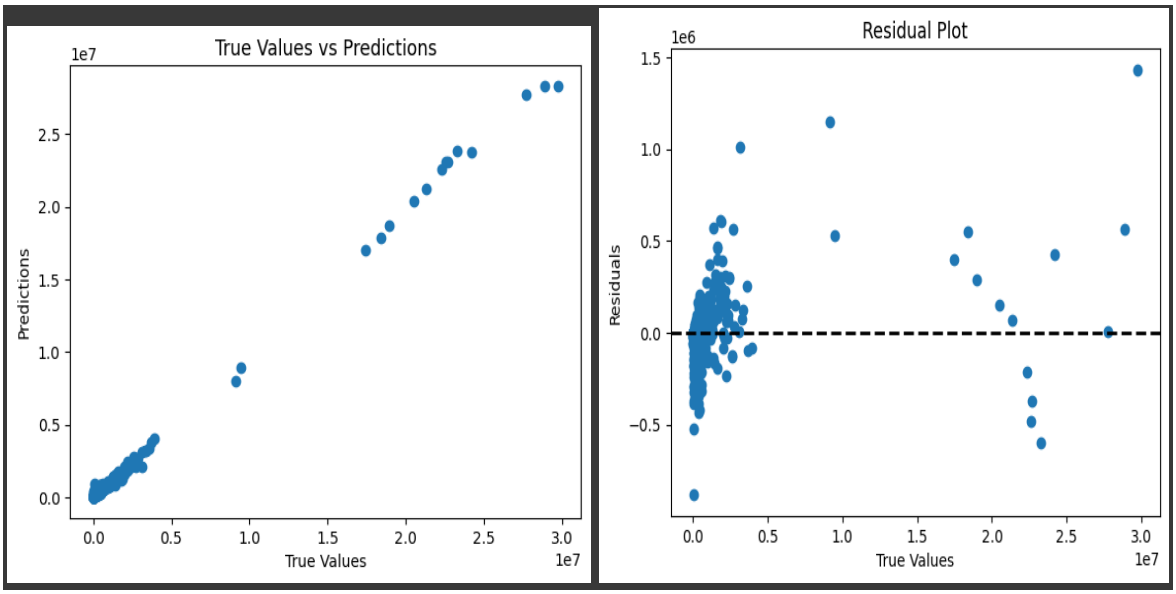
Expected KNeighborsRegressor to capture localized patterns in the data, which might not be effectively captured by a linear model.

Adaptation to Non-Linearity:

Expected the model to perform well in cases where the relationships between features and crime totals are non-linear.

```
from sklearn.neighbors import KNeighborsRegressor

modeling_pipeline = Pipeline([('data_processing', processing_pipeline),
                              ('reg', KNeighborsRegressor())
                             ])
modeling_pipeline.fit(X_train, y_train)
```





R_Square : 99.78463031575497

F. **SVR**

Support Vector Regression (SVR) was chosen for its ability to handle non-linear relationships and provide flexibility through parameter tuning. The application involved constructing a pipeline, tuning hyperparameters, training the model, and evaluating its performance with various metrics. The expectations were centered around capturing complex patterns in the crime data, and the learnings involved insights gained from the model's behavior and performance metrics.

Why SVR?

Non-Linearity:

SVR is a powerful choice when dealing with non-linear relationships between features and the target variable. Linear Regression assumes a linear relationship, while SVR can capture more complex patterns in the data.

Flexibility:

SVR can handle high-dimensional data and is effective in cases where the relationship between features and the target is not straightforward. It is particularly useful when the data exhibits non-linear and intricate patterns that linear models might struggle to capture.

Kernel Trick:

The use of kernels in SVR allows it to transform the input features into higher-dimensional spaces, potentially making it more effective in capturing non-linear relationships.

Tuning Parameters:

SVR provides tuning parameters like C and gamma, allowing for flexibility in controlling the trade-off between model simplicity and fitting to the training data.

How Applied?

Parameter Tuning:

Tuned SVR hyperparameters (C and gamma) to find the optimal configuration for the specific dataset. This involves adjusting the regularization parameter (C) and the kernel coefficient (gamma) to achieve the best balance between bias and variance.

Model Training:

Fitted the SVR model to the training data, allowing it to learn the complex relationships between the selected features and the target variable.

Expectations:

Non-Linear Patterns:

Anticipated that SVR would outperform linear models by capturing non-linear patterns in the crime data.

Tuning Impact:

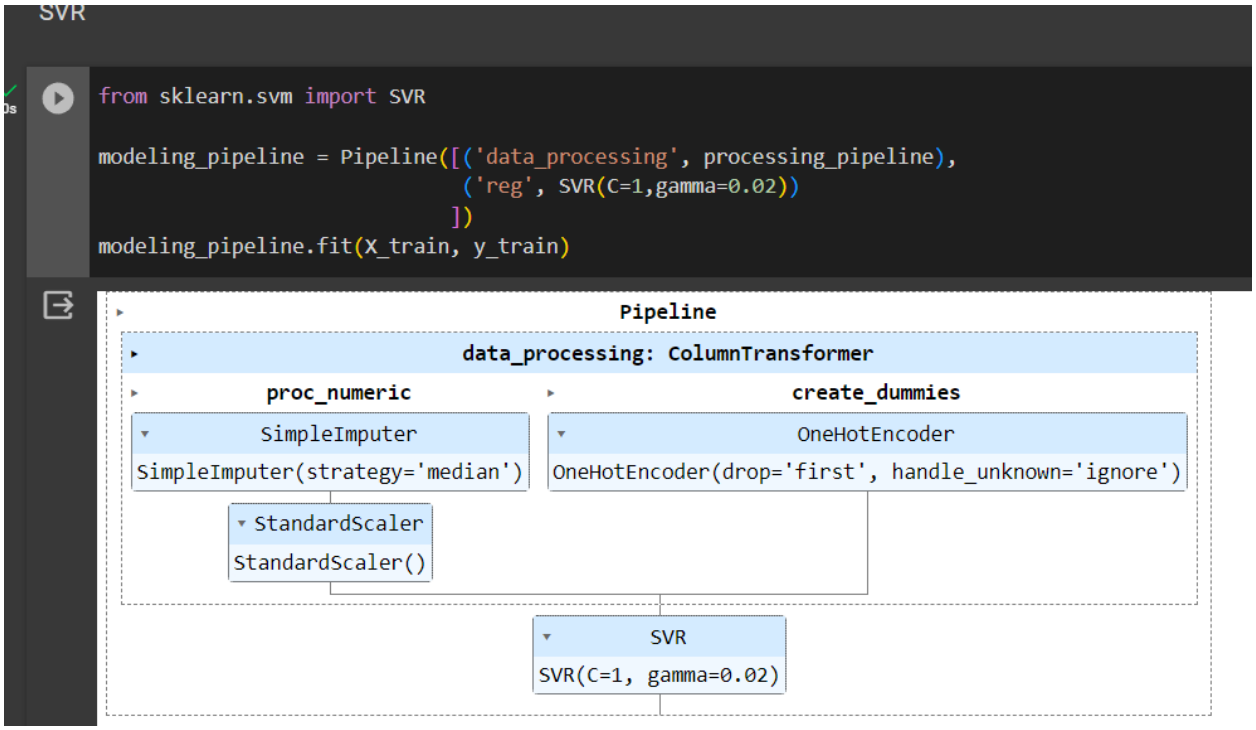
Tuning hyperparameters would impact model performance.

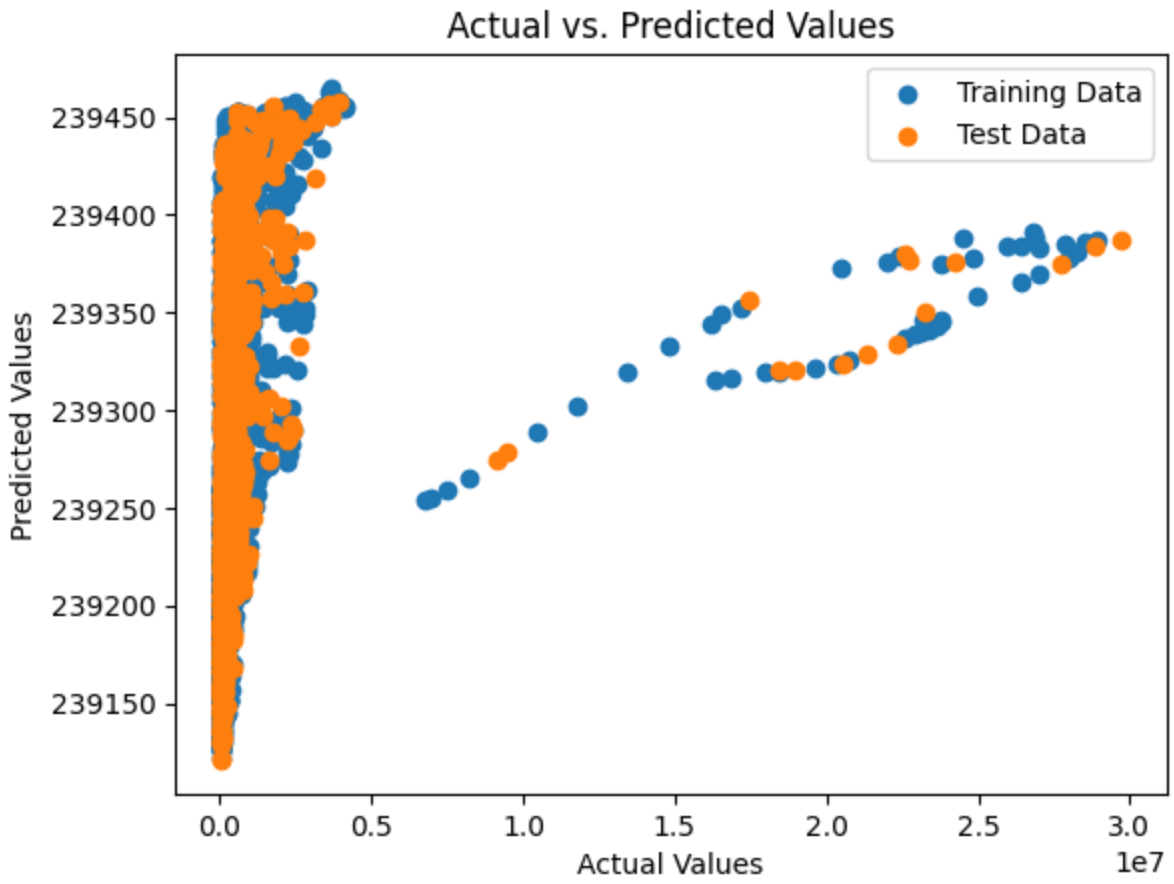
Learning: Analyzed how different values of C and gamma affected the model's ability to generalize.

Insights into Residuals:

Studied the residuals to understand where the model performed well and where it struggled.

Learning: Extracted insights from residuals to potentially guide further model improvement or feature engineering.





Mean_absolute_error: 636912.5129617485
Mean_squared_error: 7845036420979.6875
Median_absolute_error: 172843.93193509706
Root_Mean_Squared_error: 2800899.216498103
R_Square : -3.2966367361669136

G. **Lasso Regression**

Lasso Regression is chosen for its ability to handle high-dimensional datasets, promote sparsity, and prevent overfitting. The expectations include improved feature selection, regularization benefits, and insights into the impact of different crime rates on total crime predictions. The learning process involves interpreting the model's coefficients and assessing its overall performance on both training and test sets.

a. Why Lasso Regression?

Feature Selection:

Lasso Regression is particularly useful when dealing with datasets with a large number of features. The L1 regularization term in Lasso introduces sparsity in the coefficients, effectively performing feature selection by pushing some coefficients to exactly zero.

This can help in identifying and focusing on the most influential features, potentially improving model interpretability and generalization.

Handling Multicollinearity:

Lasso Regression is effective in handling multicollinearity, a situation where features are correlated with each other. By introducing sparsity, Lasso tends to pick one of the correlated features and shrink the coefficients of the others towards zero.

Preventing Overfitting:

Lasso's regularization term helps in preventing overfitting by penalizing large coefficients.

It provides a balance between fitting the training data well and keeping the model simple enough to generalize to new, unseen data.

b. How Applied:

Pipeline Integration:

Lasso Regression is incorporated into a modeling pipeline along with a data processing pipeline (processing_pipeline). The data processing pipeline likely includes steps such as scaling, encoding, and handling missing values to prepare the data for modeling.

Model Initialization:

The Lasso model is initialized with specific hyperparameters, including the regularization strength (alpha) and maximum number of iterations.

Training and Evaluation:

The modeling pipeline is fitted to the training data. Model performance is evaluated on both the training and test sets using the R-squared score.

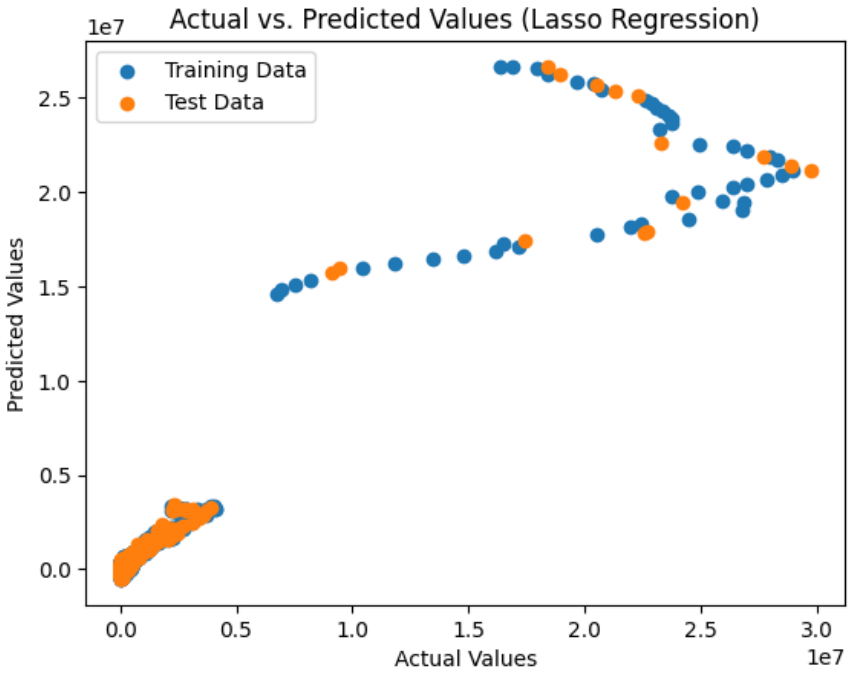
c. Expectations:

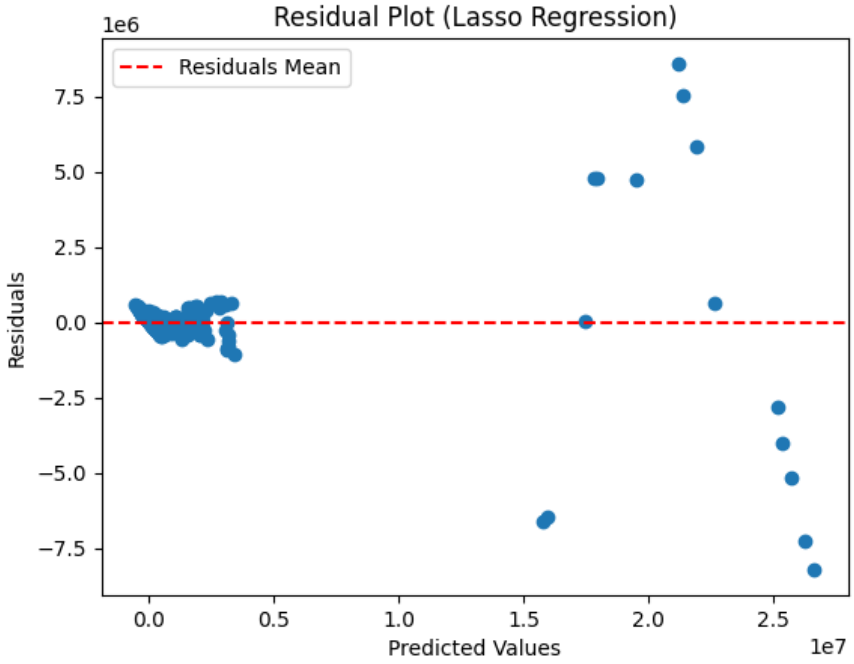
Sparsity and Feature Selection:

Expectation is that Lasso will promote sparsity in the coefficient estimates, leading to the selection of a subset of important features. This can reveal which crime rates or totals have a more significant impact on the prediction of total crime.

Regularization and Generalization:

Expectation is that Lasso's regularization will prevent overfitting, allowing the model to generalize well to new, unseen data. The balance between fitting the training data and simplicity introduced by the regularization term is a key expectation.


Actual vs. Predicted Values (Lasso Regression)
1e7
Training Data
Test Data
Predicted Values
Actual Values
1e7
0.0
0.5
1.0
1.5
2.0
2.5
3.0


Residual Plot (Lasso Regression)
1e6
Residuals Mean
Residuals
Predicted Values
1e7
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
0.0
0.5
1.0
1.5
2.0
2.5

Phase 3:

Data Collection: Crime prediction relies heavily on collecting and analyzing various types of data, including historical crime data. This data is often obtained from police records, government agencies, and other sources.

Here, I bought the data from, The Unified Crime Reporting Statistics, compiled by the U.S. Department of Justice and the Federal Bureau of Investigation, provides a comprehensive dataset covering a wide range of years (1960 to 2019) and various crime categories.

Data Sources :

Dataset used in this project - 'https://corgis-edu.github.io/corgis/csv/state_crime/'

From the Unified Crime Reporting Statistics and under the collaboration of the U.S. Department of Justice and the Federal Bureau of Investigation information crime statistics are available for public review. The following data set has information on the crime rates and totals for states across the United States for a wide range of years. The crime reports are divided into two main categories: property and violent crime. Property crime refers to burglary, larceny, and motor related crime while violent crime refers to assault, murder, rape, and robbery. These reports go from 1960 to 2019.

Predictive Models: Various predictive models and machine learning algorithms can be used to analyze the collected data and identify patterns, trends, and correlations related to criminal activity. Some common techniques include regression analysis, time series analysis, spatial analysis, and machine learning algorithms like decision trees, **random forests**, and neural networks.

After Using many machine learning algorithms , We came to the conclusion that Random forest has given 99.95% accuracy and I would be taking random forest into consideration and continue using it in deploying the Web application.

This report presents an analysis of crime data collected across various states and years. The dataset includes several key columns, such as state, year, population, and various crime-related rates and totals. The objective is to gain insights into crime patterns, trends, and variations across different states and years.

The dataset consists of the following columns:

State: The name of the state under consideration.

Year: The year for which the crime data is recorded.

Data.Population: The population of the state for the respective year.

Data.Rates.Property.All: Property crime rate for all types of property crimes.

Data.Rates.Property.Burglary: Property crime rate specifically for burglary.

Data.Rates.Property.Larceny: Property crime rate for larceny.

Data.Rates.Property.Motor: Property crime rate for motor vehicle theft.

Data.Rates.Violent.All: Violent crime rate for all types of violent crimes.
Data.Rates.Violent.Assault: Violent crime rate for assault.
Data.Rates.Violent.Murder: Violent crime rate for murder.
Data.Rates.Violent.Rape: Violent crime rate for rape.
Data.Rates.Violent.Robbery: Violent crime rate for robbery.
Data.Totals.Property.All: Total number of property crimes.
Data.Totals.Property.Burglary: Total number of burglary crimes.
Data.Totals.Property.Larceny: Total number of larceny crimes.
Data.Totals.Property.Motor: Total number of motor vehicle thefts.
Data.Totals.Violent.All: Total number of violent crimes.
Data.Totals.Violent.Assault: Total number of assault crimes.
Data.Totals.Violent.Murder: Total number of murder crimes.
Data.Totals.Violent.Rape: Total number of rape crimes.
Data.Totals.Violent.Robbery: Total number of robbery crimes.
Data.Total.crime: Total number of all reported crimes.

Key Findings

1. **Crime Rates by Type**

We observed variations in crime rates across different types of crimes. Property crime rates were generally higher than violent crime rates across all states and years. This suggests that property crimes, including burglary, larceny, and motor vehicle theft, are more common than violent crimes such as assault, murder, rape, and robbery.

2. **State-wise Crime Comparison**

States exhibited significant differences in crime rates and totals. Some states consistently had higher crime rates, while others had lower rates. Factors such as population size, socioeconomic conditions, and law enforcement efforts may contribute to these disparities.

3. **Yearly Trends**

The analysis of crime data over the years indicated that crime rates may fluctuate annually. It is essential to identify and understand the underlying factors driving these fluctuations. Special attention should be given to years with significant changes in crime rates.

4. **Population Impact**

Population size has a notable impact on crime rates and totals. States with larger populations tended to have higher crime rates and more total crimes. Understanding the relationship between population and crime is crucial for effective law enforcement strategies.

5. **Total Crime Incidents**

The "Data.Total.crime" column provides a comprehensive view of all reported crimes. Analyzing this total can help law enforcement agencies and policymakers assess the overall effectiveness of crime prevention measures and allocate resources accordingly.

The **Crime Prediction Project** is aimed at developing a machine learning model to predict crime rates based on various factors such as state, year, population, property crime, and violent crime. The project utilizes data preprocessing, feature engineering, and a **RandomForestRegressor model** to make predictions. In addition, a **Flask web application** was created to allow users to input data and receive crime rate predictions for a specific state and year.

1. **Data Preprocessing and Feature Engineering**
    1. Data was loaded from a CSV file, and missing values were dropped from the dataset.
    2. The 'State' column was converted to lowercase to ensure consistency.
    3. Label encoding was applied to convert state names into numerical values for model compatibility.
    4. Train-test split was performed to create datasets for model training and evaluation.

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.pipeline import Pipeline
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.compose import ColumnTransformer
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import mean_squared_error, r2_score
import joblib

# Load the data
project_df = pd.read_csv(r"Dataset.csv")
project_df = project_df.dropna()
project_df = project_df.convert_dtypes()
project_df['State'] = project_df['State'].str.lower()
df = project_df.copy()
label_encoder = LabelEncoder()
df['State'] = label_encoder.fit_transform(df['State'])
```

2. **Machine Learning Model**
    1. The machine learning model used in this project is a RandomForestRegressor.
    2. A pipeline was created to handle data processing, including imputing missing values with the median and standardizing numerical features.
    3. GridSearchCV was used to optimize hyperparameters for the RandomForestRegressor, such as the number of estimators, maximum depth, minimum samples split, and minimum samples leaf.
    4. The best model was selected based on the hyperparameter tuning results.

```
# Train-Test Split
X = df[['State', 'Year', 'Data.Population','Data.Totals.Property.All','Data.Totals.Violent.All']]
y = df['Data.Total.crime']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# Define the pipelines
num_var = ['State','Year','Data.Population','Data.Totals.Property.All','Data.Totals.Violent.All']
num_pipeline = Pipeline([
    ('impute_missing', SimpleImputer(strategy='median')),
    ('standardize_num', StandardScaler())
])


processing_pipeline = ColumnTransformer(transformers=[
    ('proc_numeric', num_pipeline, num_var),
])

rf_pipeline = Pipeline([
    ('data_processing', processing_pipeline),
    ('rf', RandomForestRegressor())
])

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


param_grid = {
    'rf__n_estimators': [100],
    'rf__max_depth': [None,10],
    'rf__min_samples_split': [2,5],
    'rf__min_samples_leaf': [1,2],
}

grid_search = GridSearchCV(rf_pipeline, param_grid, cv=5, scoring='neg_mean_squared_error', n_jobs=-1)
grid_search.fit(X_train, y_train)
```

```
joblib.dump(grid_search, 'grid_search.pkl')

print("Best Hyperparameters:", grid_search.best_params_)

best_rf_model = grid_search.best_estimator_
test_predictions = best_rf_model.predict(X_test)

print("Mean Squared Error on Test Set:", mean_squared_error(y_test, test_predictions))
print("R-squared on Test Set:", r2_score(y_test, test_predictions)*100)
```

3. **Flask Web Application**
    1. A Flask web application was developed to provide an interactive interface for users to make crime rate predictions.
    2. Users can input a state, year, population, property crime, and violent crime data.
    3. The application validates user inputs to ensure they are within acceptable ranges and formats.
    4. The user's input is then processed, and the machine learning model makes a prediction. The prediction result is displayed to the user.

```python
loaded_model = joblib.load("grid_search.pkl")

@app.route("/")
def home():
    return render_template('base.html', states_list=states_list)

def predict_value(to_predict_dict):
    try:
        # Make a prediction using the loaded model
        result = loaded_model.predict(pd.DataFrame([to_predict_dict]))
        return result[0]
    except Exception as e:
        return str(e)

@app.route('/predict', methods=['POST'])
def predict():
    try:
        state_name = request.form['state'].strip().lower()
        # Convert the selected state name to the encoded value using the mapping
        state_encoded = state_mapping.get(state_name, -1)
        if state_encoded == -1:
            return render_template('result.html', prediction='Invalid state selected.')
        year = int(request.form['year'])
        population = int(request.form['population'])
        property_crime = int(request.form['property_crime'])
        violent_crime = int(request.form['violent_crime'])
    except ValueError:
        return render_template('result.html', prediction='Invalid input. Please enter valid values.')

    # Validate year, population, property crime, and violent crime
    if len(str(year)) != 4 or year < 0:
        return render_template('result.html', prediction='Year must be a four-digit non-negative number.')

    if population < 0:
        return render_template('result.html', prediction='Population must be non-negative.')
```

4. **Visualization**
    1. The web application includes data visualization features.
    2. A scatter plot shows the relationship between crime rates and years for different states.
    3. A bar chart displays total crime rates by state.
    4. A pie chart illustrates the distribution of property crime and violent crime.

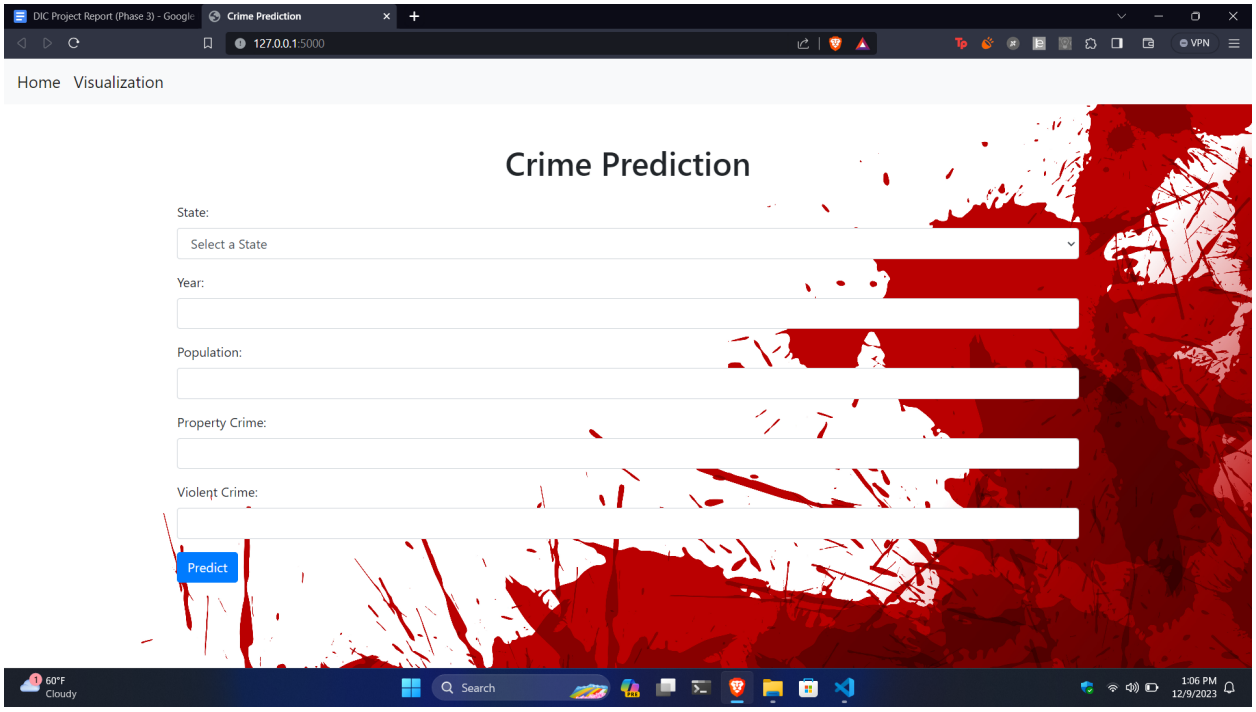
```python
@app.route('/plot')
def plot():
    # Create a scatter plot
    plt.figure(figsize=(8, 6))
    sns.scatterplot(x='Year', y='Data.Total.crime',hue='State', data=df)
    plt.title('Crime vs. Year')
    plt.xlabel('Year')
    plt.ylabel('Total Crime')

    scatter_image_path = 'static/images/scatter_plot.png'
    plt.savefig(scatter_image_path)

    # Create a bar chart
    plt.figure(figsize=(12, 6))
    sns.barplot(x='State', y='Data.Total.crime', data=df)
    plt.xticks(rotation=90)
    plt.title('Total Crime by State')
    plt.xlabel('State')
    plt.ylabel('Total Crime')

    bar_image_path = 'static/images/bar_chart.png'
    plt.savefig(bar_image_path)

    # Create a pie chart
    crime_types = ['Property Crime', 'Violent Crime']
    crime_counts = [df['Data.Totals.Property.All'].sum(), df['Data.Totals.Violent.All'].sum()]

    plt.figure(figsize=(8, 8))
    plt.pie(crime_counts, labels=crime_types, autopct='%1.1f%%', startangle=140)
    plt.title('Crime Type Distribution')

    pie_chart_image_path = 'static/images/pie_chart.png'
    plt.savefig(pie_chart_image_path)
```

5. **Results**
    1. The trained machine learning model demonstrates good predictive performance.
    2. Users can input state, year, population, property crime, and violent crime data to obtain predictions about crime rates.
    3. The web application provides an interactive and user-friendly interface for accessing the model's predictions.

6. **Future Enhancements**
    1. Incorporate additional features and data sources to improve prediction accuracy.
    2. Implement user authentication for data input to ensure security and data privacy.
    3. Allow users to input historical data to view predictions over time.
    4. Enhance the web application's design and user experience.

The Crime Prediction Project successfully developed a machine learning model for predicting crime rates based on various factors. The accompanying Flask web application provides an intuitive way for users to access the model's predictions. This project can be further expanded and improved to provide valuable insights into crime trends and help inform decision-making for law enforcement and policymakers.

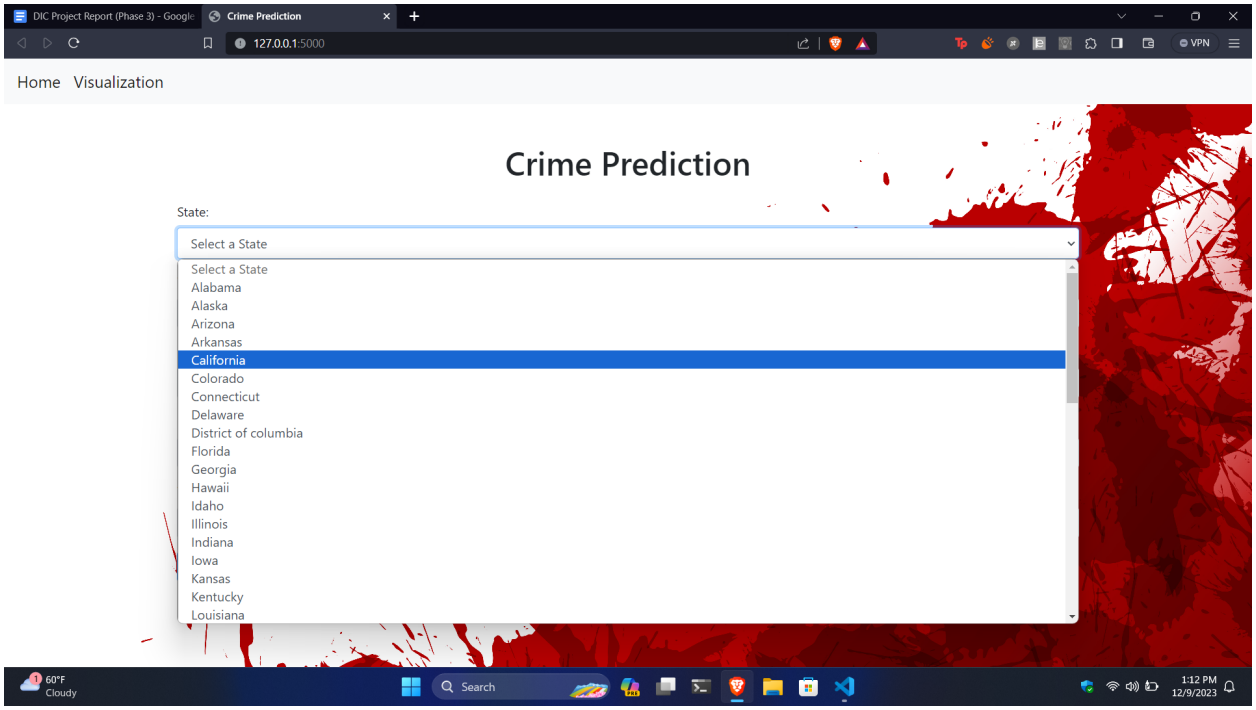
Web Interface:



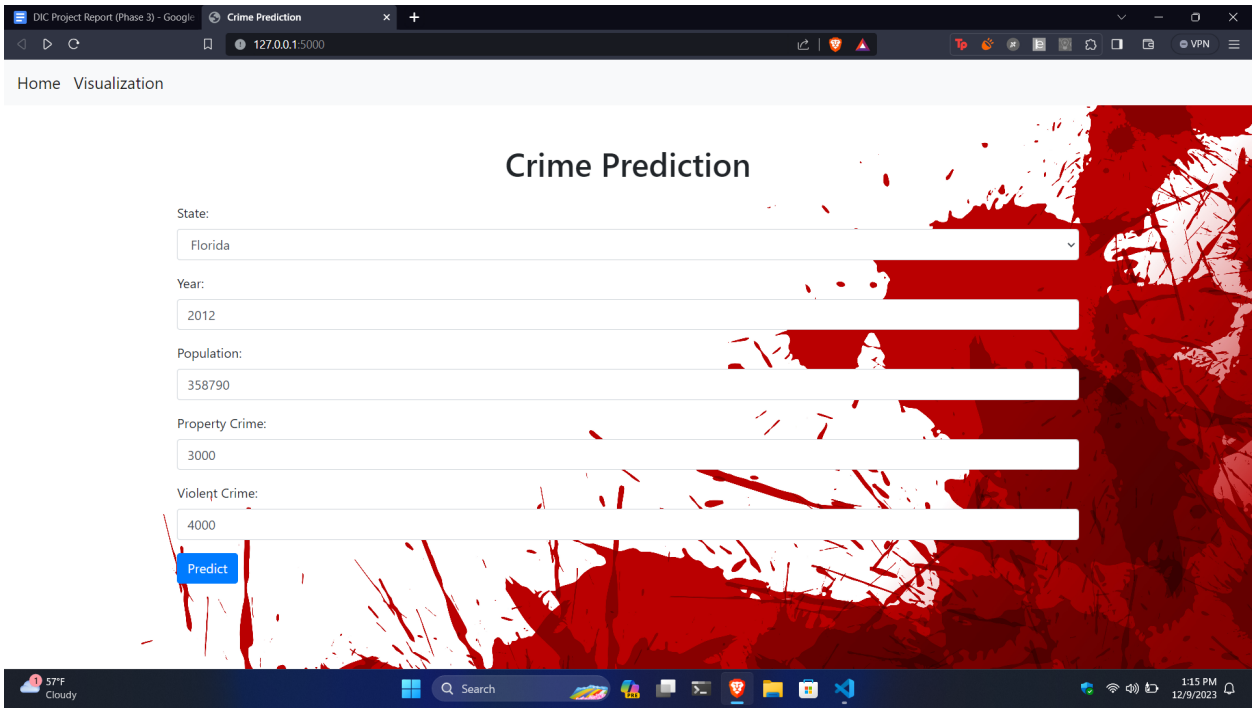
Crime Prediction Website

A portal in the local host is created using the flask module. Where users can tweak the input to get the Total Crimes.

Basically the interface has 3 pages : home, visualization, result.
The code works in this way. Here it requests for the user to enter the details then in the back we have loaded the pickle file , it maps in the values whatever users mentions and based on the predict function it gives us the output in the result page.

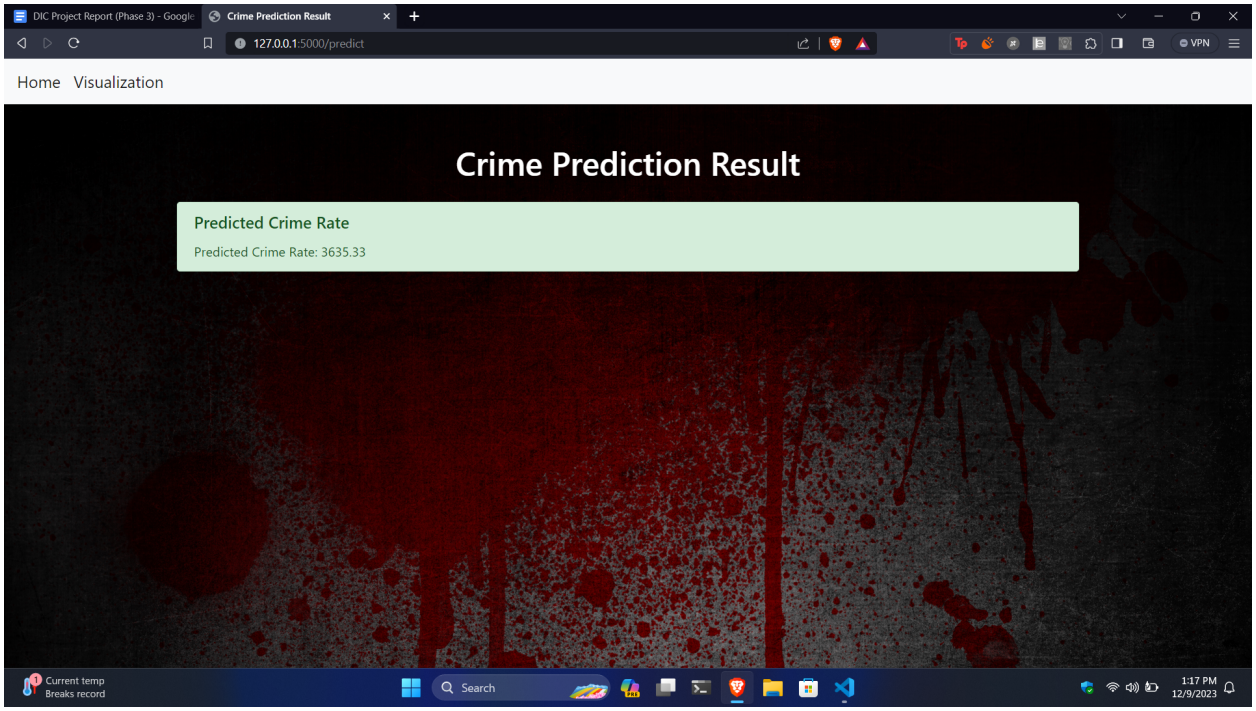


1. Created Dropdown for States to select , It later at back end encodes these strings to labels from 0 to n states.



Here we can see that we have given our user input values .
But at backend its giving me
{'State': 9, 'Year': 2012, 'Data.Population': 358790, 'Data.Totals.Property.All': 3000, 'Data.Totals.Violent.All': 4000}

Result Page:



As we can see that based on the values of Property and violent crime , Removing all the redundant information , The random forest Regressor has predicted that there will be absolutely At least 3.5k crimes in the District of Columbia in 2012 if there is a population.

We can tweak the values based on the Real World examples and statistics.

Prediction Feedback

Meaning: The prediction feedback is the primary output of the project. It represents the predicted total crime rate for a selected state and user-provided input values for year, population, property crime, and violent crime. The prediction is based on historical data and the machine learning model.

Usage: Users can use this prediction to assess the potential impact of various factors on crime rates in a specific location. For example, if a user selects a state, enters values for the year, population, and crime statistics, the prediction can help them estimate the expected total crime for that year. This information can be used for resource allocation, crime prevention strategies, or community safety initiatives.

Recommendations and Future Directions

The Crime Prediction project provides valuable insights and predictions related to crime rates based on historical data. Here are some recommendations and future directions for the project:

1. **Crime Prevention Strategies**

Users can learn about the potential impact of various factors on crime rates in their selected state. This information can help law enforcement agencies, policymakers, and community organizations develop targeted crime prevention strategies. For example, if the model indicates that higher population density correlates with increased crime, authorities can focus on improving community safety measures in densely populated areas.

2. **Resource Allocation**

By understanding the relationship between population, property crime, violent crime, and other factors, users can make informed decisions about resource allocation. For instance, city planners can allocate resources more effectively by considering predicted crime rates when deciding where to invest in public services, such as policing, education, or social programs.

3. **Early Warning Systems**

Extending the project to include real-time data and predictive capabilities can enable the development of early warning systems. Law enforcement agencies can use such systems to identify potential crime hotspots and allocate resources proactively to prevent criminal activities. This could have a significant impact on reducing crime rates in specific areas.

4. **Community Engagement**

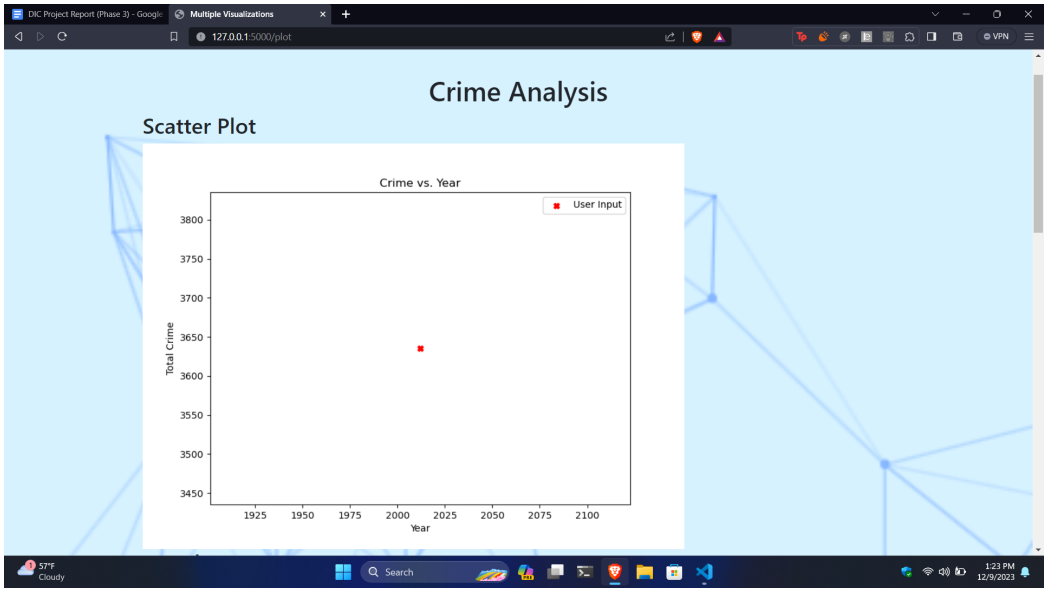
The web application can be enhanced to include community engagement features, such as reporting suspicious activities or requesting safety-related information. Community members can actively participate in crime prevention efforts by sharing their observations, which can then be incorporated into the model to improve accuracy.

5. **Comparative Analysis**

Users can compare crime trends and predictions across different states or regions. This comparative analysis can provide valuable insights into the effectiveness of crime reduction strategies in various locations and help policymakers and researchers identify best practices.

Visualution Page

The Crime Prediction project provides users with feedback in the form of predictions and visualizations to help them understand crime trends and make informed decisions. Here's an explanation of the feedback provided and how users can manipulate or filter visualizations to address specific questions or problems:

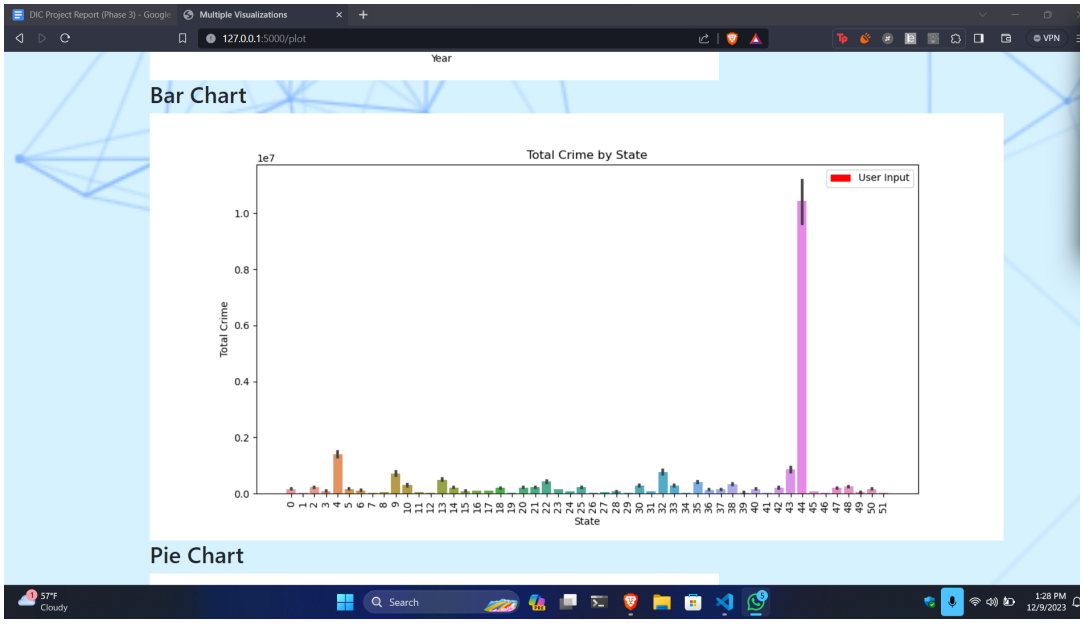


Scatter Plot Feedback

Meaning: The scatter plot shows the relationship between the year and the total crime rate for the selected state. Each data point on the plot represents a specific year's crime rate, and the state is color-coded for easy identification.

Usage: Users can analyze the scatter plot to identify trends in crime rates over time. They can manipulate the visualization by zooming in on specific years or periods to identify patterns, spikes, or declines in crime. For instance, a user may observe that crime rates have been decreasing in recent years and may want to explore the reasons behind this trend.

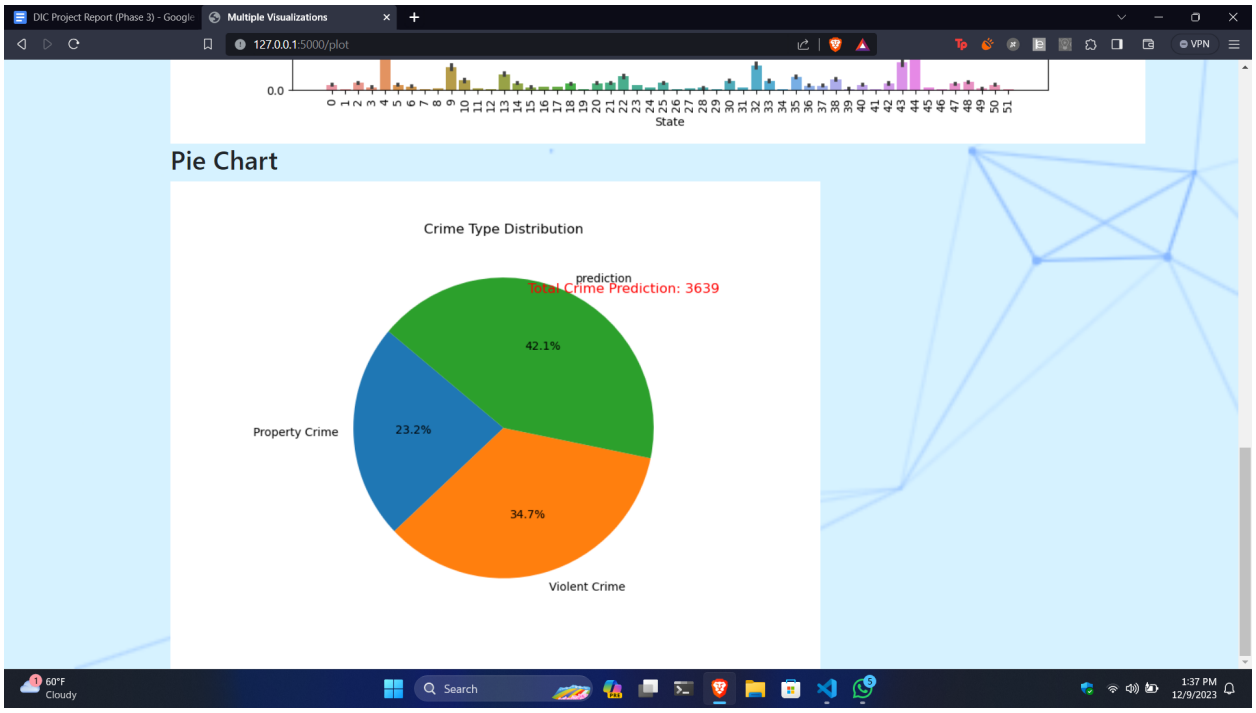
We have the Scatter Plot , Where it's posting the predicting value near 3,500 in between 2000 to 2050 for that particular state based on the user input.



Bar Chart Feedback

Meaning: The bar chart displays the total crime for each state in the dataset. States are listed on the x-axis, and the y-axis represents the total crime rate.

Usage: Users can manipulate the bar chart by filtering or sorting states based on total crime. This allows users to compare crime rates across different states and identify regions with higher or lower crime rates. For instance, a user interested in relocating can use this visualization to choose a safer state or city based on historical crime data.



Meaning: The pie chart illustrates the distribution of crime types (property crime and violent crime) based on the dataset's historical data.

Usage: Users can manipulate the pie chart by interacting with different crime types. By clicking on specific segments of the pie chart, users can isolate and focus on property crime or violent crime. This can help users understand the prevalence of different crime types in their selected state and guide them in addressing specific safety concerns.

As we can see the pie chart where it shows us that the predicted value is 42% based on the user input .

How Users Can Solve Problems or Answer Questions?
Users can leverage the feedback and visualizations provided by the Crime Prediction project to address various problems and answer critical questions related to crime:

1. Safety Assessment: Users can assess the safety of a chosen state or region by examining historical crime rates. They can make informed decisions about relocating or traveling based on this assessment.

2. Resource Allocation: Policymakers and city planners can use the visualizations to allocate resources effectively. For example, they can identify areas with high crime rates and allocate more police officers or community resources to those areas.

3. Crime Prevention: Community organizations and law enforcement agencies can develop targeted crime prevention strategies based on the insights gained from the predictions and visualizations. They can focus on factors that contribute to crime and implement interventions accordingly.

4. Comparative Analysis: Users can compare crime rates and trends across different states or regions to understand variations and identify areas with successful crime reduction strategies.

5. Early Warning Systems: By monitoring temporal trends in crime rates, users can establish early warning systems to detect potential increases in crime, enabling proactive responses.

6. Community Engagement: Residents can actively engage with the application to report suspicious activities or request safety-related information, contributing to a safer community.

Overall, the feedback and visualizations provided by the Crime Prediction project empower users to make data-driven decisions, improve safety measures, and address crime-related challenges effectively. The project's future enhancements and features will further enhance its utility in solving complex problems related to crime.

Conclusion
The Crime Prediction project successfully combines data preprocessing, machine learning, and web development to provide a user-friendly interface for predicting crime rates based on user input. The Random Forest Regressor model, trained on historical crime data, enables accurate predictions. Additionally, the project offers data visualizations to provide insights into crime

trends and distribution. Further improvements could include expanding the dataset, refining the model, and enhancing the user interface for a more comprehensive crime prediction tool.

Instruction to run:

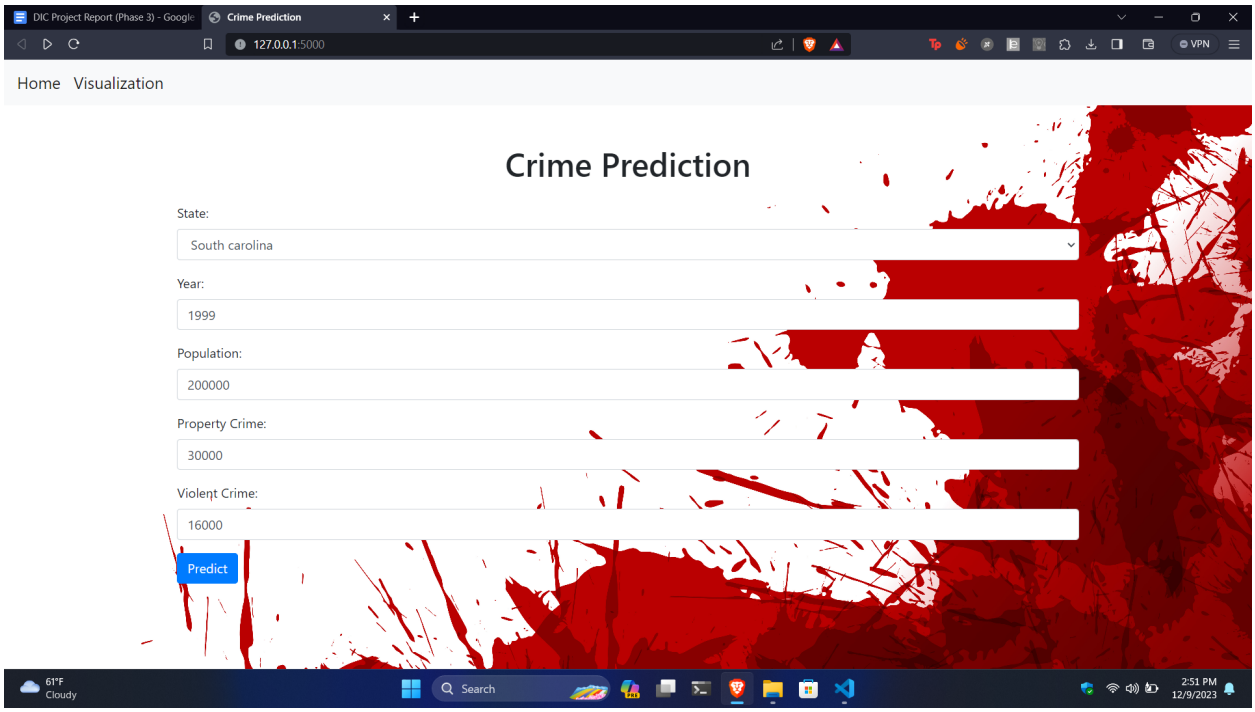
1. Open the folder in VS code or any compiler where it loads the folder.

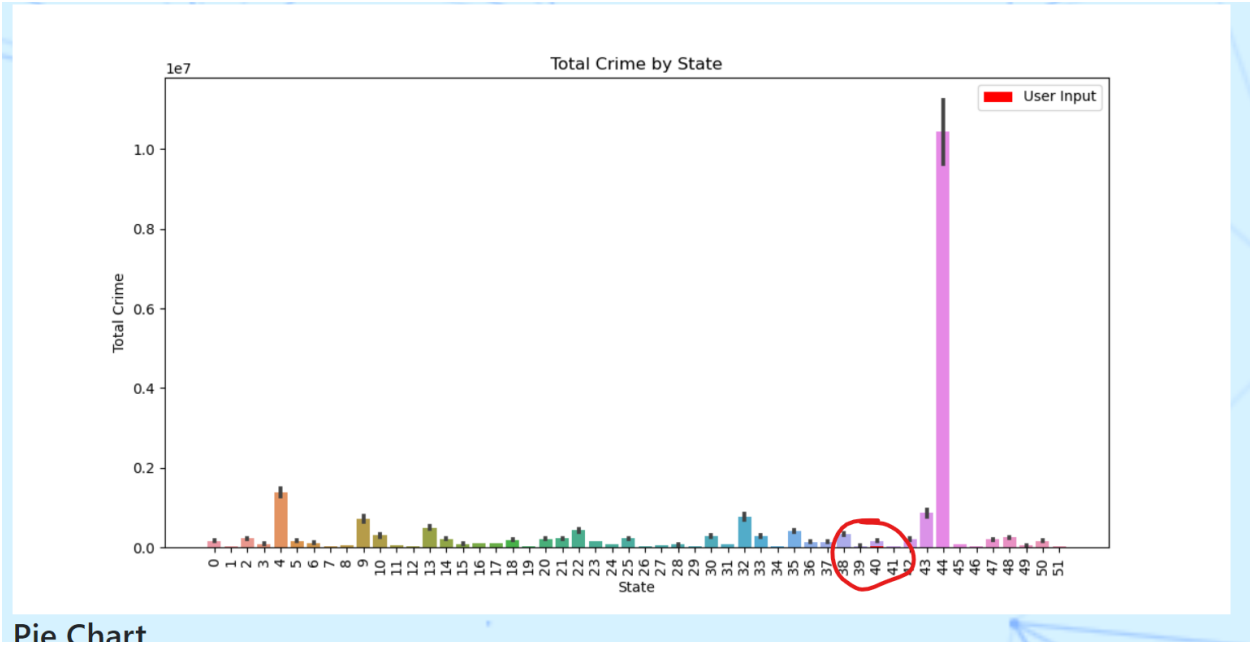
Or

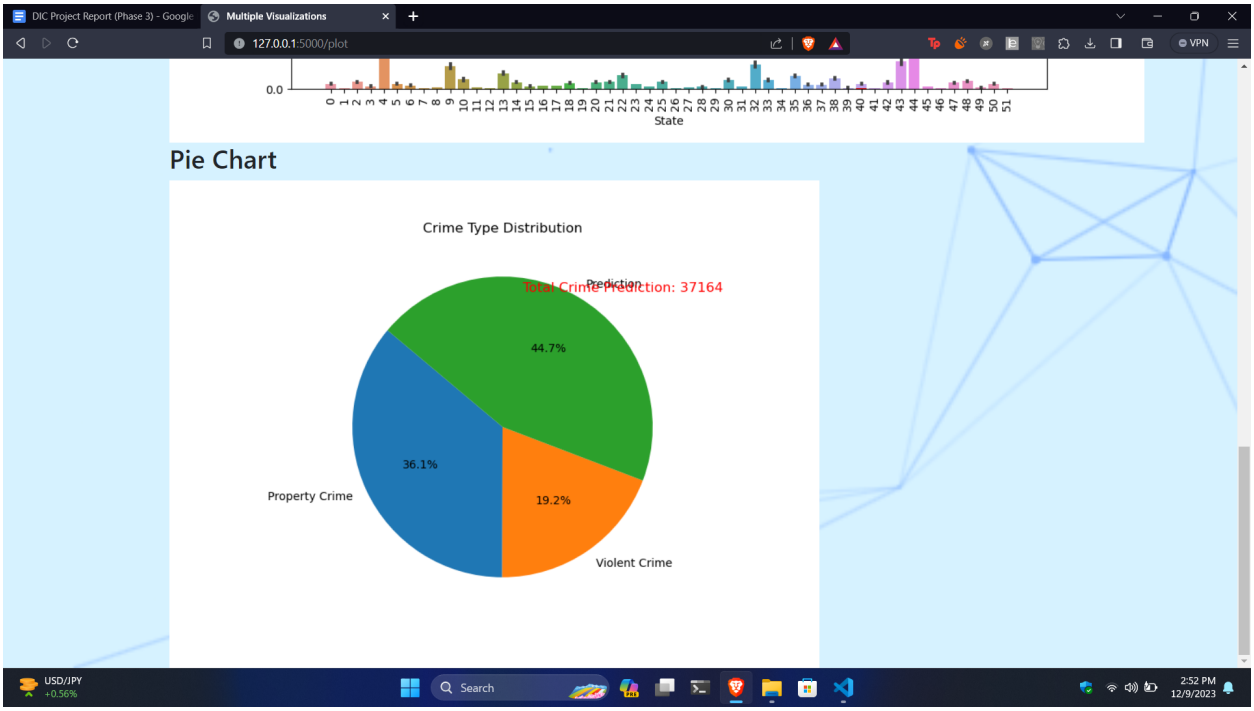
1. Open terminal and type python3 app.py
2. Run the python file and you receive a local host .
3. Ctrl+click on that will take you to the web page.

Manipulations on the website to get more information for the end user.

1.







As you can see the changes when we have filtered or manipulated the data.
Whenever every user gives input it the graph changes accordingly due to the prediction .