Naan Mudhalvan Scheme
TNSDC – Machine Learning to Generative AI
Project: Crime Analysis and Prediction with Machine Learning

THANES M
3rd Year – 2021503712
Madras Institute of Technology, Anna University

Introduction :

Day by day crime data rate is increasing because the modern technologies and hi-tech methods are helps the criminals to achieving the illegal activities. According to Crime Record Bureau crimes like burglary, arson etc have been increased while crimes like murder have been increased. Crime data will be collected from various blogs, news and websites. The huge data is used as a record for creating a crime report database. The knowledge which is acquired from the data mining techniques will help in reducing crimes as it helps in finding the culprits faster and also the areas that are most affected by crime . A particular approach has been found to be useful by the police, which is the identification of crime 'hot spots 'which indicates areas with a high concentration of crime. Use of data mining techniques can produce important results from crime report datasets. It also helps to see if a crime in a certain known pattern or a new pattern necessary. The occurrences of crime depended on several factors such as intelligence of criminals, security of a location, etc The work has followed the steps that used in data analysis, in which the important phases are Data collection, data classification, pattern identification, prediction and visualization. The proposed framework uses different visualization techniques to show the trends of crimes and various ways that can predicts the crime using machine learning algorithm. The inputs to our algorithms are time (hour, day, month, and year), place (latitude and longitude), and class of crime.

Problem statement :

- Analyze the impact of modern technologies on rising crime rates, focusing on burglary, arson, and murder, as per Crime Record Bureau reports.
- Investigate data mining's effectiveness in swiftly identifying criminals and crime hotspots to aid law enforcement's prevention strategies.
- Examine factors affecting crime prediction accuracy, including criminal intelligence, security measures, and geographical location.
- Evaluate crime analysis methods for detecting patterns and generating actionable insights for law enforcement.

- Explore machine learning's potential in predicting future criminal activity based on temporal, spatial, and categorical data, for integration into law enforcement strategies.

Objectives :

- Determine the root causes of increasing crime rates.
- Identify crime hotspots for targeted law enforcement efforts.
- Enhance crime prediction accuracy using data mining and machine learning techniques.
- Improve crime analysis methods to generate actionable insights for law enforcement agencies.
- Develop effective visualization techniques for communicating crime trends to the public and law enforcement stakeholders.

Existing System :

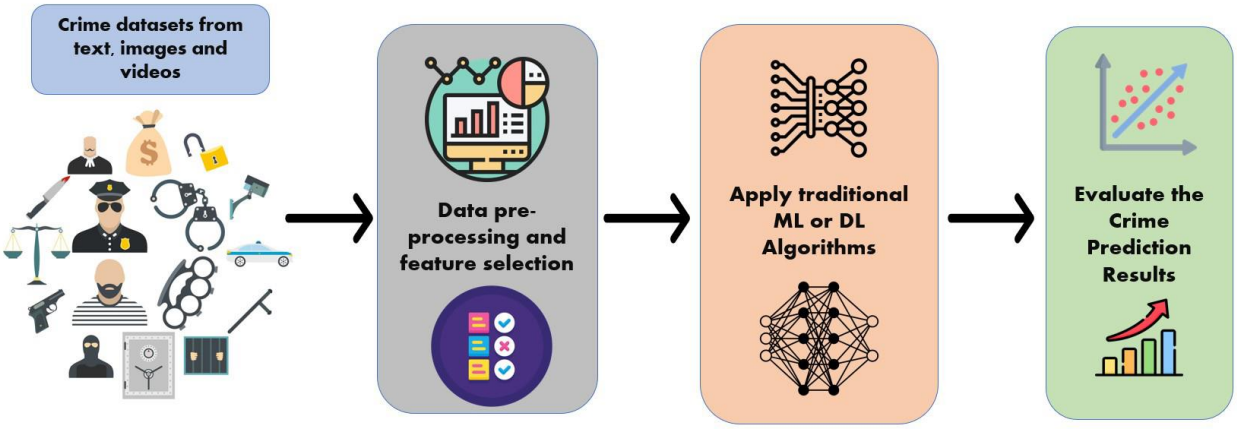
Data mining in the study and analysis of criminology can be categorized into main areas, crime control and crime suppression. De Bruin et. Al. introduced a framework for crime trends using a new distance measure for comparing all individuals based on their profiles and then clustering them accordingly. Manish Gupta et. Al. highlights the existing systems used by Indian police as e-governance initiatives and also proposes an interactive query based interface as crime analysis tool to assist police in their activities. He proposed interface which is used to extract useful information from the vast crime database maintained by National Crime Record Bureau (NCRB) and find crime hot spots using crime data mining techniques such as clustering etc. The effectiveness of the proposed interface has been illustrated on Indian crime records. The purpose of his study is to examine the use of clustering technology to automate fraud filtering during an audit. He used cluster analysis to help auditors focus their efforts when evaluating group life insurance claims.

Proposed System :

In this project, we aim to leverage machine learning techniques, particularly K-means clustering algorithm and Kernel SVM, for crime prediction using a dataset extracted from the official police portal. The dataset includes crime information such as location description, type of crime, date, time, latitude, and longitude. By preprocessing the data, performing feature selection, and scaling, we intend to enhance the accuracy of the predictive models. The primary objective is to demonstrate how machine learning can aid law enforcement agencies in detecting, predicting, and solving crimes efficiently, thereby reducing crime rates. This approach holds potential for adoption by law enforcement agencies in other regions, contingent upon the availability of similar datasets.

Crime Prediction Process & Datasets :

Crime prediction using machine and deep learning involves several major. The first step is data collection, which involves gathering relevant data such as crime statistics, demographics, and weather patterns. The next step is data preprocessing, which includes cleaning and transforming the data into a usable format. After data preprocessing, the data is split into training and testing sets for model development and evaluation. The next step is featuring engineering, which involves selecting relevant features from the data that can be used to train the model. Once the features are selected, various machine and deep learning algorithms can be applied to the data for training and prediction purposes. Finally, the trained models are evaluated using various performance metrics to assess their accuracy and activeness in predicting crime. The results can be used to support decision-making in law enforcement and crime prevention efforts.



Methodology :

Data Preprocessing :

- Cleaning the dataset by handling missing values and outliers.
- Encoding categorical variables for model compatibility.
- Feature selection to identify relevant attributes for crime prediction.
- Scaling numerical features to ensure uniformity in model training.

Reding Dataset :

```
[ ]  dataset = pd.read_csv('/content/newtrial - Sheet 1 - 01_District_wise_crim 2.csv')
     X = dataset.iloc[:, :-1].values
     y = dataset.iloc[:, -1].values
```

- Load the dataset from a CSV file into a Pandas DataFrame.
- Separate the features (X) and the target variable (y) from the dataset.

Filling Missing Value:

```
from sklearn.impute import SimpleImputer
imputer = SimpleImputer(missing_values=np.nan, strategy='mean')
imputer.fit(X[:, 2:31])
X[:, 2:31] = imputer.transform(X[:, 2:31])

print(X)
```

```
[['A & N ISLANDS' 2001 13.0 ... 0.0 0.0 323.0]
 ['A & N ISLANDS' 2002 17.0 ... 0.0 0.0 328.0]
 ['A & N ISLANDS' 2003 21.0 ... 0.0 0.0 318.0]
 ...
 ['WEST BENGAL' 2010 2398.0 ... 8.0 2847.0 49096.0]
 ['WEST BENGAL' 2011 2109.0 ... 0.0 3249.0 56614.0]
 ['WEST BENGAL' 2012 2252.0 ... 12.0 4385.0 64482.0]]
```

- Import the SimpleImputer class from scikit-learn to handle missing data.
- Create an instance of SimpleImputer and specify the strategy to replace missing values with the mean of the respective column.
- Fit the imputer to the feature columns containing missing values.
- Transform the feature columns by replacing missing values with their respective column means.

Encoding on States :

```
[ ] from sklearn.preprocessing import OneHotEncoder
    from sklearn.compose import make_column_transformer
    from seaborn import load_dataset
    import pandas as pd

    df = X

    transformer = make_column_transformer(
        (OneHotEncoder(), [0]),
        remainder='passthrough')

    transformed = transformer.fit_transform(df)
    X = transformed
    print(X)
```

```
[[1.0 0.0 0.0 ... 0.0 0.0 323.0]
 [1.0 0.0 0.0 ... 0.0 0.0 328.0]
 [1.0 0.0 0.0 ... 0.0 0.0 318.0]
 ...
 [0.0 0.0 0.0 ... 8.0 2847.0 49096.0]
 [0.0 0.0 0.0 ... 0.0 3249.0 56614.0]
 [0.0 0.0 0.0 ... 12.0 4385.0 64482.0]]
```

- Import necessary libraries and modules from scikit-learn and Seaborn.
- Define a DataFrame df containing the features.

- Create a ColumnTransformer object, specifying OneHotEncoder transformation for the first column (assuming it represents states) and keeping other columns unchanged.
- Fit and transform the DataFrame using the defined transformer, resulting in one-hot encoded features.
- Update the feature matrix X with the transformed data.

Saving dataset to a file and downloading it to verify one hot encoding :

```
DF = pd.DataFrame(X)
DF.to_csv("ds (1).csv")
print(X[0][0])
```

```
1.0
```

- Convert the feature matrix X into a DataFrame DF.
- Save the DataFrame to a CSV file named "ds (1).csv" using the to_csv method.
- Print the value of the first element in the first row of X to verify.
- Clustering to find optimal number of clusters :

```
[ ] y = np.array(y)
    Z = y
    U = (X[:, 37:])
    A = X[:, 36:]
    print(X[:, 36:])
```

```
[[13.0 0.0 0.0 ... 0.0 0.0 323.0]
 [17.0 3.0 1.0 ... 0.0 0.0 328.0]
 [21.0 4.0 1.0 ... 0.0 0.0 318.0]
 ...
 [2398.0 2111.0 630.0 ... 8.0 2847.0 49096.0]
 [2109.0 2242.0 486.0 ... 0.0 3249.0 56614.0]
 [2252.0 2854.0 522.0 ... 12.0 4385.0 64482.0]]
```

- Convert the target variable y to a NumPy array.
- Create arrays U and A containing subsets of features from X.
- Print the subset of features A to verify.

K-means Clustering :

- Utilizing K-means clustering to identify patterns and groupings within the dataset.
- Clustering crime data based on spatial attributes such as latitude and longitude.
- Analyzing clusters to understand high-crime areas and temporal trends.

```
from sklearn.cluster import KMeans
wcss = []
for i in range(1, 11):
    kmeans = KMeans(n_clusters = i, init = 'k-means++', random_state = 42, n_init=10)
    kmeans.fit(A)
    wcss.append(kmeans.inertia_)
plt.plot(range(1, 11), wcss)
plt.title('The Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
plt.show()
```

```
kmeans = KMeans(n_clusters = 2, init = 'k-means++', random_state = 42, n_init = 10)
y_kmeans = kmeans.fit_predict(A)
print(y_kmeans)
```

```
[0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0
```

- Import the KMeans class from scikit-learn for clustering.
- Initialize an empty list wcss to store the within-cluster sum of squares (WCSS) for different numbers of clusters.
- Iterate over a range of cluster numbers from 1 to 10.
- For each cluster number :

Create a KMeans object with the specified number of clusters (n_clusters), using k-means++ initialization (init = 'k-means++') for better initialization and setting the random seed for reproducibility (random_state = 42).

- Fit the KMeans model to the data A.
- Append the WCSS value to the list wcss.
- Plot the number of clusters against WCSS to identify the "elbow point" using the Elbow Method.
- Set the title, x-axis label, and y-axis label for the plot.
- Show the plot.
- Choose the number of clusters based on the "elbow point" observed from the plot (e.g., 2 clusters).
- Create another KMeans object with the chosen number of clusters.
- Fit the KMeans model to the data A and predict the cluster labels for each data point.
- Print the cluster labels.

	YEAR	MURDER	ATTEMPT TO MURDER	CULPABLE HOMICIDE NOT AMOUNTING TO MURDER	KIDNAPPING & ABDUCTION	KIDNAPPING AND ABDUCTION OF WOMEN AND GIRLS	KIDNAPPING AND ABDUCTION OF OTHERS
count	420.000000	420.000000	420.000000	420.00000	420.000000	420.000000	420.000000
mean	2006.500000	958.435714	838.040476	106.22381	849.659524	628.350000	221.309524
std	3.456169	1213.687365	1149.623253	253.86659	1217.488114	966.279578	342.334028
min	2001.000000	0.000000	0.000000	0.00000	0.000000	0.000000	0.000000
25%	2003.750000	46.000000	32.000000	4.00000	37.500000	15.500000	11.750000
50%	2006.500000	463.500000	434.000000	29.00000	295.500000	228.000000	81.500000
75%	2009.250000	1461.000000	1302.250000	92.25000	1195.500000	894.750000	287.000000
max	2012.000000	7601.000000	7964.000000	1616.00000	8878.000000	7910.000000	2416.000000

Kernel SVM :

- Implementing Kernel SVM for crime prediction, considering its ability to handle nonlinear relationships.
- Training the SVM model using labeled crime data, with features such as location, time, and crime type.
- Evaluating the SVM model's performance using appropriate metrics like accuracy, precision, and recall.

```
[ ]  from sklearn.svm import SVC
     classifier = SVC(kernel = 'linear', random_state = 0)
     classifier.fit(X_train, y_train)

     # Predicting the Test set results
     y_pred = classifier.predict(X_test)
     print(np.concatenate((y_pred.reshape(len(y_pred),1), y_test.reshape(len(y_test),1)),1))

     # Making the Confusion Matrix
     from sklearn.metrics import confusion_matrix, accuracy_score
     cm = confusion_matrix(y_test, y_pred)
     print(cm)
     accuracy_score(y_test, y_pred)
```

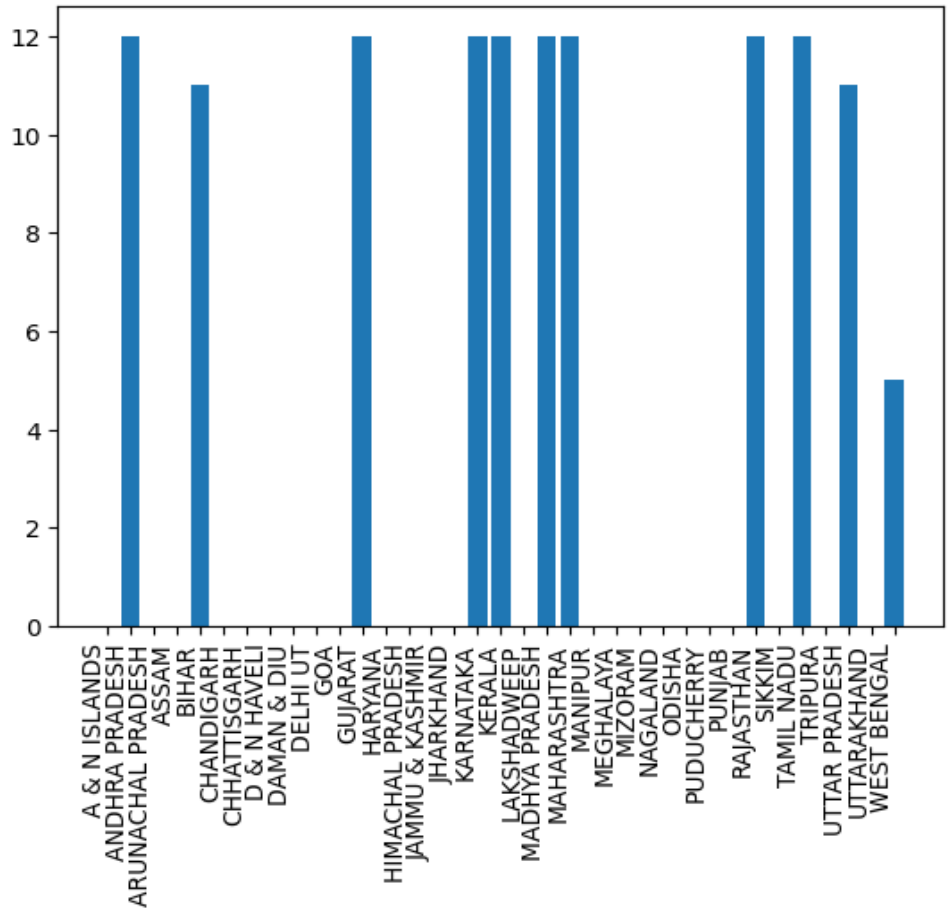
```
[[0. 0.]
 [0. 0.]
 [1. 1.]
 [0. 0.]
 [0. 0.]
 [1. 1.]
 [1. 1.]
```

- Import the SVC class from scikit-learn for Support Vector Classification.
- Create an instance of the SVC class with parameters specifying a linear kernel (kernel = 'linear') and setting the random seed for reproducibility (random_state = 0).
- Fit the SVM model to the training data (X_train and y_train) using the fit method.
- Prediction:
- Predict the target variable for the test set (X_test) using the trained SVM model.
- Print the concatenated array of predicted and actual target values for comparison.
- Import the confusion_matrix and accuracy_score functions from scikit-learn metrics.
- Calculate the confusion matrix for the predicted and actual target values using the confusion_matrix function.
- Print the confusion matrix.
- Calculate the accuracy score by comparing predicted and actual target values using the accuracy_score function.

Classifying safe / unsafe :

- The cluster labels obtained from K-means clustering represent different levels of crime rates or safety.
- Based on the clustering results, states with higher total sums of cluster labels may be considered less safe or more prone to crime.
- Conversely, states with lower total sums of cluster labels may be considered safer or have lower crime rates.
- By examining the bar chart, one can visually identify states that fall into these categories, helping classify them as safe or unsafe based on the clustering results.

```
print(y_kmeans)
dataset['category'] = y_kmeans
df_sum_by_state = dataset.groupby('STATE/UT')['category'].sum().reset_index()
states = df_sum_by_state['STATE/UT']
sum = df_sum_by_state['category']
#print(df_sum_by_state)
fig, ax = plt.subplots()
plt.xticks(rotation=90, ha='right')
ax.bar(states, sum)
# Display the resulting DataFrame
plt.show()
```



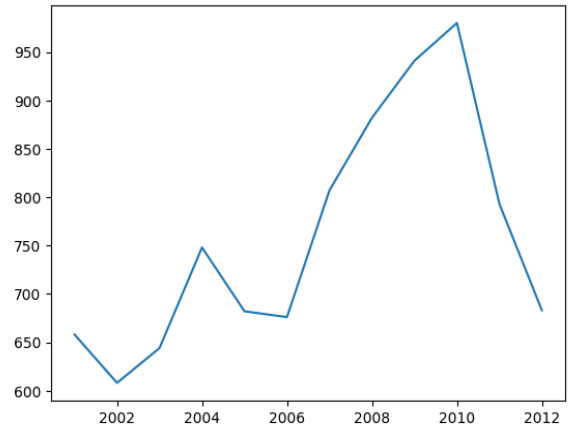
Visualization :

- Creating graphical representations to visualize crime trends over time and space.
- Plotting crime density maps to identify hotspots and prioritize resource allocation.
- Generating temporal graphs to depict variations in crime rates by time of day, month, or year.

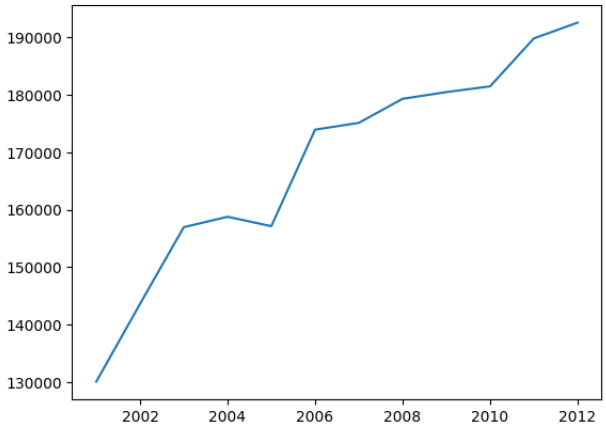
```
[ ]  import seaborn as sns
     import numpy as np
     import pandas as pd

     states = (dataset.iloc[:, 0].unique())
     for state in states:
       print( state, " : \n")
       data = dataset[dataset['STATE/UT'] == state]
       grouped = data.groupby('YEAR').agg('TOTAL IPC CRIMES').sum()
       arr = np.array(grouped)
       year = (dataset.iloc[:, 1].unique())
       plt.figure()
       plt.plot(year, arr)
       plt.show()
       print("\n")
```

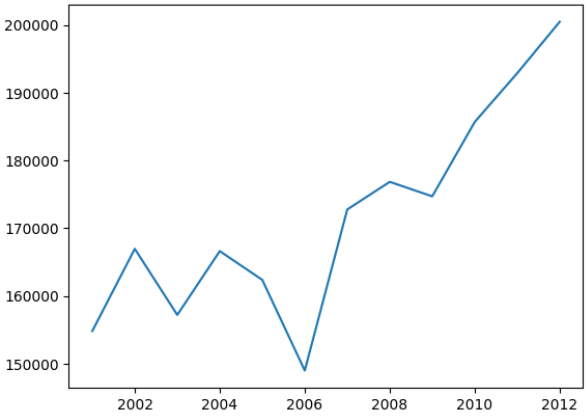
A & N Islands :



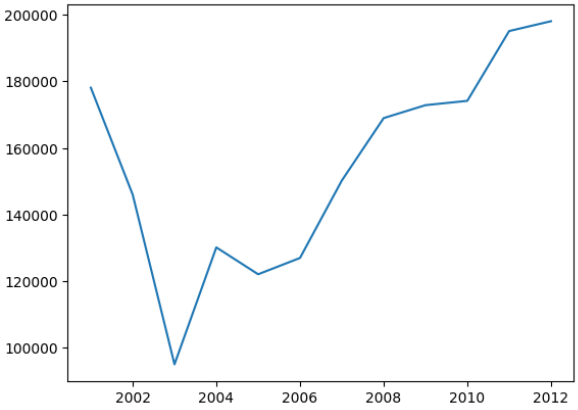
Andhra Pradesh :



Tamil Nadu :



Uttar Pradesh :



Conclusion :

By employing K-means clustering and Kernel SVM, this project aims to provide law enforcement agencies with effective tools for crime prediction and analysis. The combination of spatial clustering and advanced classification techniques can enhance decision-making processes, enabling proactive crime prevention strategies. The visualization of crime data aids in understanding complex patterns, facilitating targeted interventions. Ultimately, the adoption of machine learning in law enforcement holds promise for reducing crime rates and enhancing public safety.