ENHANCING CRIME PREDICTION WITH KNORA DYNAMIC ENSEMBLE SELECTION AND CROSS-VALIDATION

S.R. Divyasri[1], R. Saranya[2]*, P.Kathiravan[3]

Abstract
Crimes are a social issue that affects not only an individual but also humanity. Crime classification techniques for crime forecasting are an emerging research area. generally, Crime data are centrally organized with regular maintenance of the criminal registers that can aid officers in sharing observations and improve early alert approaches to keep the citizens secure within their towns. Hence, the aim of this study is to compare the performance of the state-of-the-art Dynamic Ensemble Selection of Classifier algorithms for predicting crime. We used five different benchmark crime datasets (Chicago, San Francisco, Pheonix, Boston, and Vancouver) for this experimental research work. The performance of the state-of-the-art dynamic ensemble selection of classifiers algorithms was evaluated and compared using various performance evaluation metrics such as accuracy, F1-score, precision, and recall. The KNORA Dynamic ensemble algorithms, which select the subset of ensemble members before the forecasting, outperformed the typical machine learning algorithms, and also the traditional ensemble algorithm techniques in terms of accuracy showed that the dynamic ensemble algorithms are more powerful. This ability to predict crimes within urban societies can help citizens, and law enforcement makes precise informed conclusions and preserves the neighborhoods more unassailably to improve the quality of life for humans.

Keywords: Crime prediction, Dynamic ensemble algorithms, Machine Learning, Supervised Learning , KNORA.

[1]Ultramatics India private limited, Chennai, India
[2*,3]Department of Computer Science, Central University of Tamil Nadu, India. [2]Email: saranya@cutn.ac.in

***Corresponding Author:** R. Saranya
*Department of Computer Science, Central University of Tamil Nadu, India. Email: saranya@cutn.ac.in

DOI: - 10.48047/ecb/2023.12.si5a.0156

1.INTRODUCTION

Crime tremendously impacts people's minds, concerns, and spirits, not just its actual effects on society. As a result, law enforcement agencies continue to monitor controlled areas to notice suspicious activity, become more vigilant, and improve their ability to prevent potential criminal activity [1], [2].

Machine learning in this era of Artificial Intelligence (AI) is a gravitating topic for effectively conducting analysis and prediction [3], [4]. Lately, many research works have been carried out on Crime analysis and forecast using various prediction models and methodologies to peek at the trends and patterns of past crimes, which could further help to indicate and control the expected upcoming crime that may happen in advance [5]. This research paper organizes some of the major topics to be investigated in the crime detection and prediction techniques in machine learning and summarizes the superior methods like dynamic selection and dynamic ensemble selection algorithms for better accuracy. We will also present the future challenges and research gaps that will help scholars redefine the problems in crime analysis and prediction with various machine learning algorithms that are universally suitable for all datasets.

A few years ago, AI algorithms were limited to just the field of work for which they were processed. Nevertheless, computers could move beyond doing what they were programmed for and developing with every iteration of machine learning.

Machine learning also has an adequate flow of mixed and organized data required for a powerful ML solution. Many companies in the modern online world have access to a tremendous amount of data regarding their clients, usually millions of data. This data, which is immense in the number of data points and fields, is comprehended as big data because of its sheer amount of information, which is time consuming and challenging to process by mortal means [6], [7].

The more pure, usable, and machine-readable data in an expansive dataset, the better practical the training of the ML algorithm will be. Currently, ML algorithms are trained by employing three main ways Supervised learning, Unsupervised learning, and Reinforcement learning.

The objective of this experimental research work is to address the following research question:

RQ 1: What is the state-of-the-art dynamic ensemble of selection classifier algorithm for crime prediction?

RQ 2: Can we use a common ML classifier for different datasets (crime dataset)?

Our proposed work uses supervised learning, which is acquainted with labeled data. In supervised learning, the ML algorithm is presented with a small training dataset that is part of the larger dataset to operate with and helps provide the algorithm with a fundamental concept of the problem, the solution to that problem, and which data points to be encountered in the future. The trained dataset is significantly similar to the final dataset with its features and delivers the algorithm with the labeled parameters needed for the problem. Supervised algorithms find the relationships between the given parameters, effectively demonstrating a reason and outcome association between the variables in the dataset, thus explaining how the data functions and the affinity between the intake and the outcome of the algorithm.

Further, we moved one step forward to find the solution for lesser accuracy produced by the linear machine learning model that paved the way to experiment with several other algorithms. We came across many improved data pre-processing methods, such as feature engineering, which helps increase the model's efficiency. Different cross-validation techniques were applied to the datasets to check up on the algorithms working [8].

Then we explored Ensemble supervised machine learning classifiers [9]–[12]. The basic algorithms that return only a single hypothesis tend to suffer from three main problems. The problems include statistical problems due to high variance, computational friction, and representational issues that are highly biased, some of which can be overcome by using the ensemble method [13].

We have then introduced the dynamic ensemble algorithms to the pre-processed data, which dynamically selects one of the multiple trained models to make the forecasting based on specific input criteria. This field of dynamic selection has met with great success in many problems. These algorithms typically divide the input feature space and assign particular models to predict each partition. There are many different DCS algorithms, and our research focuses on ameliorated efficiency for accurately classifying crimes compared to previously achieved algorithms [14].

Results from each part of the research are compared to choose the best-performing algorithm that gives a better outcome. The rest of our study is arranged in the following paper. Section 2 has the sum up of the related works. Section 3 has the methodology used, i.e., mainly the dynamic ensemble learning algorithms. Section 4 presents the result and discussion, while section 6 contains the conclusion of our study.

2. RELATED WORKS

In this paper, we have proposed dynamic selection and dynamic ensemble selection architectures for predicting crime test data. We have used the Dynamic Algorithms OLA, KNORA-E, and KNORA-U. Apart from this, we have explored various methods and techniques to improve the accuracy of basic machine-learning algorithms through Ensembles, Cross-validation, and data preprocessing. Dynamic Ensemble selection classifiers modeling is one of the recently buzzed research fields. Using DES methods for crime prediction can help society and law enforcement to be more accurate and avoids confusion about which model to select and apply to the recently updated dataset. But unfortunately, it hasn't been considered for crime prediction and analysis. Our paper has implemented a few research gaps, such as the possibilities of recognizing different algorithms, using multiple algorithms on different datasets to ensure efficient working, directions of topics to increase accuracy, and whether the result obtained is consistent on all the databases.

In order to address the problem of locating adequate human trafficking data to permit machine learning solutions to analyze human trafficking data, the authors have provided a dataset and generalized dataset creation framework. For the state of Kentucky, this solution aggregates crime datasets from many sources to enable researchers to find patterns and information that would not be visible otherwise [1].
Authors have [2] presented machine learning data mining techniques in Crime prediction. They evaluated that the Naïve Bayes algorithm performs better than the KNN, which is considered the best with respect to the base paper in terms of accuracy. In [3], authors used the comparative analysis using accuracy with algorithms like Random Forest, Decision Tree, and ensemble algorithms such as Extra Trees, Bagging, and AdaBoost on the Chicago dataset, where the bagging algorithm shows higher accuracy.
Authors [4], compared the crime data prediction CC, accuracy, precision, recall and ROC of Chicago data with Naïve bayes and Decision tree algorithms. The Decision tree algorithm proved to work better on forecasting the selected features in test data.

Authors use text mining to extract logical relationships from unstructured crime data. In order to uncover multi-level linkages across crime entities, they specifically provide an associative questioning based searching approach. They used this approach with partition clustering to create a collaborative, human-assisted data mining and knowledge discovery process [5].

In [6], the authors proved that Naïve bayes shows highest accuracy among k-NN (all optimal value of k), Naive Bayes, and Decision Tree for the dataset of Sleman Regency. In [7], researchers proposed naïve Bayes with feature selection methods FAMD has shown more accuracy than the PCA method on Saudi Arabia crime data. [8] did FBI crime analysis with the Chicago dataset using Decision Tree, Gaussian Naïve Bayes, k-NN, and Logistic

Regression by predicting crimes classifying, pattern detection, and visualization.
[9]suggested the crime prediction model based on SVM and random forest algorithm can forecast the incidence of crime, and the trend of its forecasted data is consistent with the direction of actual data; this model that is established can effectively predict criminal behaviors that endanger public health and provide reliable data for prevention.

[10]applied various machine learning techniques to predict more than 35 crime types in Chicago and Los Angeles, such as logistic regression, SVM, Naïve Bayes, KNN, decision tree, MLP, random forest, and XG Boost, and time series analysis evaluated with RMSE and MAE by LSTM and ARIMA model to fit the crime data better. [11] revealed that the assemble-stacking-based crime prediction method (SBCPM) based on the SVM algorithm achieves domain-specific configurations compared with another machine learning model, J48, SMO Naïve byes bagging, and the Random Forest. They also proved that any empirical data on crime is compatible with criminological theories and suggested that the prediction accuracy of the stacking ensemble model is higher than that of the individual.
[12] gave an idea of how crime investigation agencies can utilize data mining to discover relevant precautionary measures from prediction rates using some supervised classification algorithms, namely decision trees, KNN, and random forest algorithms. It focused on forecasting the crime for frequently occurring crimes like robbery, assault, and theft, and test data showed random forest gives the highest accuracy. [13]Succeeded in analyzing the performance of the KNN method with the k-Fold of fold-3 Cross Validation algorithm as an evaluation model and the Analytic Hierarchy Process method as feature

selection for the data classification process to obtain the best level of accuracy and machine learning model.

[14] explored machine learning models like the Random Forest, KNN, AdaBoost, and Neural Network on the Chicago Police Department's CLEAR (Citizen Law Enforcement Analysis and Reporting) system. Among all four models, the Neural Network has the best outcome. [15] presented a crime prediction model by analyzing and comparing Naive Bayes, Random Forest, and Gradient Boosting Decision Tree algorithms. Exploratory data analysis is also performed for identified the patterns and understand the trends of crimes using a crime dataset. The Gradient Boosting Decision Tree prediction model is better than the other two techniques for predicting criminality, based on historical data from San Francisco city.

[16] alleviated the issues of Chronic kidney disease (CKD) and the necessity of early prediction. They used data from medical records of Brazilians with or without a diagnosis of CKD and presented an oversampling approach based on manual and automated augmentation. They experimented with the SMOTE, Borderline-SMOTE, and Borderline SMOTE SVM and implemented models based on the algorithms: decision tree, random forest, and multi-class Ada Boosted Decision Trees. They applied the overall local accuracy and local class accuracy methods for dynamic classifier selection; and the KNORA U, KNORA E, and META-DES for dynamic ensemble selection. They also analyzed the models' performances using hold-out validation, multiple stratified cross-validations (CV) and nested CV.

From all these substantial related works research, we came to know that the dynamic classifiers and dynamic ensemble selection of classifier algorithms are not researched for crime analysis and prediction exhaustively. Hence, we explored algorithms like OLA, KNORA U, and KNORA E in dynamic algorithms. Further, this implementation covered many related studies, which will be briefly explained in the following topics.

3.METHODOLOGY

In relation to this research, our primary goal is to produce a model that could be useful to the law enforcement unit and to our civilians [17]. Our objective is to train our model to accurately classify and forecast the crime category using the test data by using a dynamic ensemble classification algorithm [11], [18].This, in turn, could help in planning the deployment of the police force in the area with a high probability of crime occurrences so that it can be prevented prior. The block diagram of our proposed framework is shown in the figure Block diagram [19].

Fig 1 gives a brief view of the methodology that we have used. The Dataset is analyzed and visualized to understand the data. Then, we pre-processed the data with methods for handling the missing values by data imputation, data type conversions using various encoders, and removal of unclean data. The preprocessed data is used to detect and eradicate the outliers. Then feature engineering properties are applied to select the efficient features for model building. The dataset is split into train and test data. The train data is used for model building using a few classification algorithms first and then ensemble learning techniques. To attain better results, we have used cross-validation to achieve the best parameters and apply them to the same classification algorithm [20]. The dynamic ensemble classifier models are then applied to see the results. A detailed description of the algorithms is explained later in the forthcoming paper.

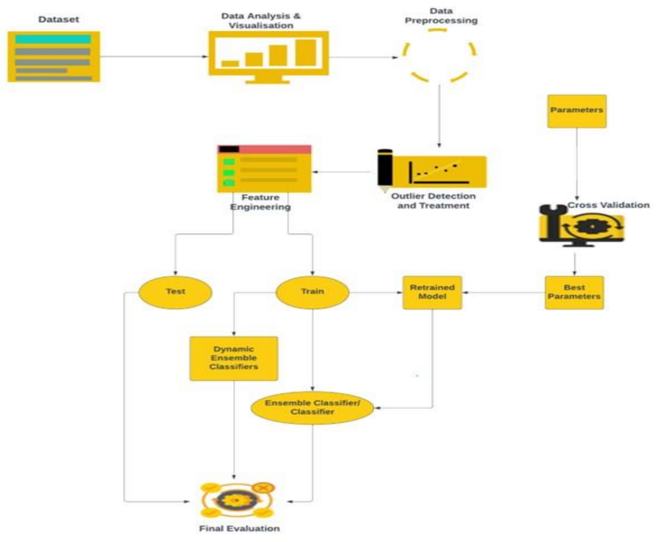


Fig 1 : Block Diagram of the proposed methodology

3.1. DATA COLLECTION AND STUDY AREA

Data collection is the approach that involves collecting and estimating data from numerous distinct sources. Gathering data permits us to grasp a record of past occurrences to analyze that data to discover systematic patterns. With the help of those patterns and machine learning algorithms, we can build predictive models that peek at tendencies and predict future changes. Proper data collection techniques are essential to design high-performing models. The data should be without errors and include pertinent information for the assignment [7], [19].

The data used for this work include the following datasets.

1) The open public San Francisco crime dataset (2003-2015) with 878049 rows in the training set and 884262 rows in the testing set, which has long registered a consistently high rate of crimes from Kaggle, comes in handy as a testing and training dataset [21]–[23].
2) The Kaggle dataset of Crime in Vancouver (2003-2017) contains 624,038 instances of violent crimes.
3) The Chicago crime dataset from Chicago Data Portal (2021-present) has 215969 records (dated June 13, 2022) [4], [8], [10].
4) The Kaggle dataset of Crime in Boston, from June 14, 2015, to September 3, 2018, contains 319073 instances of violent crimes.
5) The Pheonix Crime Dataset is a criminal record file in CSV format given by the City of Pheonix Ope n Data that is updated daily and contains crime data from November 2015 until the year 2021 with 4278 43.

Table 1: Benchmark Dataset description

Crime Dataset	Crime Record Period	# Instances	# Attributes y	#Crime Types/Categor
San Francisco	2003-2015	878049	9	38
Chicago	2021-2022	215969	17	31
Vancouver	2003-2017	624038	10	11
Boston	2015-2018	319073	17	34
Pheonix	2015-2021	427843	8	9

Tables 1 illustrate the descriptions of the benchmar k datasets.

3.2. DATA ANALYSIS

Data analysis is transforming, cleaning, and processing primary raw data and pulling out valid, pertinent data that enables the model to make informed decisions. The data analysis approach helps reduce the hazards inherent in decision making by delivering valuable understandings and statistical figures like charts, images, tables, and graphs [24].

We have used the classification technique for this work as we have decided to work on a particular target label (Category/Type of crime).

We have uncovered some information from the data analysis step, such as the total number of records or the rows. The San Francisco dataset has 878049 rows and 9 features, Vancouver dataset has 624038 rows and 10 features and Chicago dataset is a live dataset which gets updated every week with 17 features. Pheonix city crime data consists of 427843 rows and 8 features, and Boston dataset contains 319073 rows and 17 features to it. Analysation of the distribution of different types of crime gives a clear picture that Larceny is the crime with the highest frequency in San Francisco. The highest crime rate in Vancouver is theft from vehicles, and the Battery (a kind of theft) seems to be high in Chicago. Motor Vehicle accident response is shown as the highest occurred crime in Boston, whereas the Pheonix recorded Larceny-Theft as the city's frequently occurred crime.

To carry on with more data analysing, we need to pre-process the data, i.e., attribute splitting on the Date feature to split the timestamp into the date, month, year, hour, and minute. The frequency of crime seems to be high during 2013, and as of the month, October has the most elevated rate in San Francisco. In case of time of crime occurrences, at San Francisco most of the offenses likely occurred around six in the evening, and the crime rate touched its peak on Fridays. The highest rate of these crimes happened in the southern police department district of San Francisco.

In Vancouver, the month of August and the year 2003 got recorded as the peak of crime. By midnight, many crimes had occurred in Vancouver, Pheonix and Chicago. Around evening five there are frequent crime in Boston. In Chicago and Boston, the crime frequently happened in September and October, whereas in Pheonix the crime rate has peaked in January and mostly in 2019.

With Python's built-in data analytics mechanisms, we have easily penetrated patterns, correlated information in extensive sets, and got better insights into complementing other critical matrices in estimating performance.

3.3. DATA PRE-PROCESSING

Data pre-processing is the method of converting plain data into a discernible format. This step is crucial in machine learning because we cannot work with raw data. The data quality should be maintained before applying machine learning algorithms to the data [25]. Fig 2 depicts the various preprocessing activites, which are done in this experimental research on different crime dataset.

In Python, the libraries are predefined to perform specific tasks. Importing all the essential libraries is one of the mandatory steps in data pre-processing in machine learning. Some of the core Python libraries used for this project are NumPy, Pandas, Matplotlib, Plotly, Folium, Seaborn, SK learn, Statsmodels, and DESlib [26]. All the datasets in csv file format were imported using Python by the read csv() function inside the code.

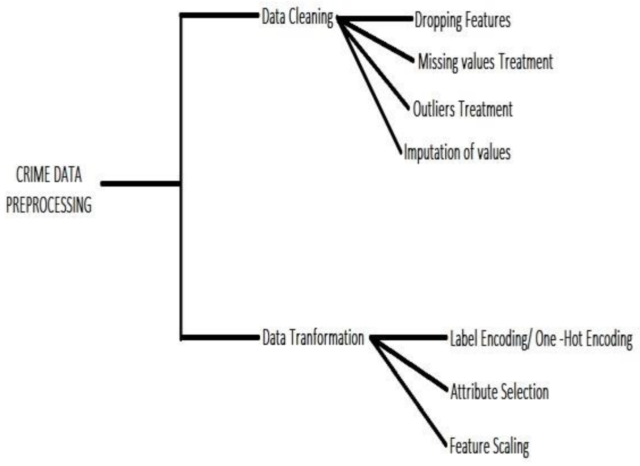


Fig 2: Various data pre-processsing activities

Data Cleaning:

Data cleaning is improving or extracting wrong, deteriorated, poorly formatted, replicas, or insufficient data in the dataset**.** Data cleansing is a vital component of the prevalent data management method and one of the essential parts of data preparation work that trains data sets for benefit in machine learning. When integrating numerous data origins, there are multiple possibilities for data to be replicated or mislabeled [27]–[29].

Handling the missing values in the dataset:

It is mandatory to conduct detailed analysis steps with good data visualizations besides data preprocessing to understand the data in a better way [30]. There are no null or missing values found in the San Francisco dataset. Chicago, Vancouver, Pheonix, and Bosten datasets had null values in features.

We used both data imputations using the standard deviation and deleted the unwanted columns and rows with missing data for this dataset. But removing missing values rows gave better accuracy for all the datasets [31]–[33].

Encoding the categorical data:

The Machine is trained chiefly with numerical values. Hence it is essential to convert the character data types to numerical data that the device can understand. So, we have used label encoding[30] and one-hot encoding with dummy variables to convert the char data typed feature to categorical variables and nominal variables—especially the encoding of target labels to categorical variables and then to numerical values by factorizing it [20], [34], [35]. The problem that arises during the conversion of categorical value is that the variable may show the multicollinear property, that is, a robust correlation of independent variables to each other.

Multicollinearity is a notion in statistics where multiple variables in a model are associated with each other, i.e., correlation. When the correlation coefficient is negative or positive, the variable is collinear in nature [36].This led to the consequence of less dependable statistical hypotheses.

R^2 represents VIF. The higher the R^2 value, the higher variable is correlated with the other variables. VIF is statistically denoted as:

$$VIF = \frac{1}{1-R^2} \quad (2)$$

Outliers Detection and Removal:
Outlier is a data entity that varies enormously from the remnant of the data entities and acts differently**.** This won't fail the model we build [37]. We tried plotting a crime awareness street map but found outliers in the latitude and longitude in some of our datasets. To confirm whether the data has outliers, we have applied various visualization techniques like Boxplot, violin plot, and scatter plot on the features.
Outliers are unusual data points that are distant from the other observation. Thus, it needs to be detected and treated before using the Machine learning algorithms as they are sensitive to the distributive range of the values of the features [37], [38].

Feature scaling:
Feature scaling is a step of Data Pre-Processing applied to independent variables or features of data. This is a part of data transformation which is utilized to transform the primary raw data into an appropriate format that efficiently relieves data mining and regains strategic data [39]. The columns' scaling helps standardize the data within a specific range. Sometimes, it also helps in racing up the estimations in an algorithm [40]. Feature scaling is applied for the dataset to remove points outside the bounding box, points with wrong coordinates, and drop duplicate rows.

Feature selection:
Feature selection is the method of decreasing the number of infusion variables when creating a predictive model [41], [42] essential because, with fewer features, the models that are yet to be built become additionally interpretable, and training of the model becomes faster, which in turn reduces the space required by the model [42]. To improve the predictive accuracy of our test data, we used feature selection algorithms in our dataset by choosing more relevant features and leaving the irrelevant and replicated features [43]. Fig 3 Feature Selection shows how the selection happens in the filter and wrapper method with a clear understanding.

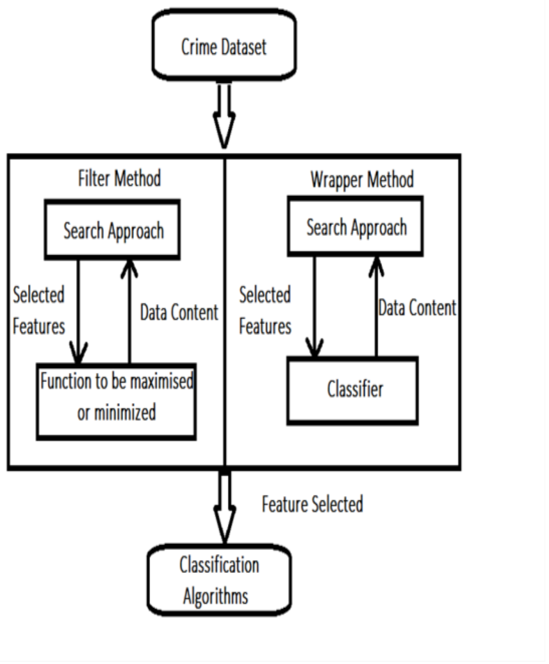


Fig 3: Various Feature Selection techniques

Cross-Validation Methods
Cross-validation is a process for estimating the machine learning Classifiers by training many different classifiers on the batch of the available data given as the input and assessing those on the complementary data set. For our work, we used cross-validation to find and see if data is overfitting [44].

K-Fold:
K-Fold cross-validation is a method employed to evaluate the mastery of the model on new data. The approach has one parameter (k), which refers to

splitting a given data sample into various groups [45]. Since K-Fold ensures that every observation from the initial dataset can occur in the training and test set, it works best when we have limited input data. The value for k is selected so that every train/test batch of data samples is adequately enormous to be statistically suggestive of the more comprehensive dataset. For our implementation, the k value for the k-fold methodology is taken as five, and the whole sample dataset was randomly split into five equally sized disjoint folds, every time giving a varied folding of the whole sample. For every i value, four of the folds were utilized for validating the model, and the rest one-fold was utilized for testing.

Stratified K-Fold:
Stratified K-Fold is a variation of k-fold, which produces layered folds. In Stratified K-Fold CV, every Individual set holds the same ratio of samples of separate target classes as the whole set. From our observation of the results, we can choose Stratified K-Fold over K-Fold while working with classifiers with highly variated class distributions [46].For our implementation of the Stratified k-fold, the k value is chosen as five, and the full sample dataset with the same ratio of samples of separate target classes was split into five equally sized disjoint folds, every time giving a varied folding of the whole sample. For every i value, four of the folds were utilized for validating the model, and the rest one-fold was used for testing.

Repeated K-Fold:
Repeated K-Fold cross-validation delivers a method to enhance the calculated implementation of a machine learning model. This procedure implies merely reiterating the cross-validation process numerous times and noting the mean development across all folds from all execution. So, this takes a high computation cost to execute this technique, and it fits smaller-sized datasets. Repeated K-Fold is an efficient approach to estimating the forecast fallacy and the precision of a model [47]. For our work of the Repeated k-fold, the k value is chosen as five, the whole sample dataset was split into five equally sized disjoint folds, every time giving a varied folding of the whole sample. For every value of i, four of the folds were utilized for validating the model, and the rest one-fold is utilized for testing. This process is repeated 3 times.

Shuffle Split/Monte Carlo:
Shuffle Split/Monte Carlo cross-validation uses the Reprised arbitrary subsampling validation mechanism that divides the dataset haphazardly into training and testing sets [48]. The traditional k-fold cross-validation splits the dataset into groups or folds, but shuffle split cross-validation uses the random split method. For every dataset that we have taken, we chose five as the number of splits, and the dataset is split into ten equal parts. Random fifty percent of data is used as a train set; thirty percent is used as a test set, and the rest is left unused. This procedure follows a shuffled pattern.

Ensemble methods

An ensemble method is obtained by blending diverse models to get a more optimal predictive model [49]. Instead of just counting on one model and expecting we earned the right decision at each split, ensemble methods permit us to take a sampling of various models into account, compute which features to utilize or queries to ask at each partition, and make a final predictor based on the aggregated results of the sampled models [49]. As of why the ensemble works better than the primary machine learning model due to its performance because it can predict better than the linear model does. The ensemble model's reliability is higher as there is a reduction in the distribution of the model performance and prediction.

There are different types of ensemble classifiers that we have used, including

- **Boosting:** Boosting is a technique that tries to build a robust classifier from the numerous feeble classifiers in a series [50]. A model is built at first using the training data, and then the next model is built that attempts to fix the errors present in the model created earlier [51]. Until the complete dataset is predicted correctly, this procedure is repeated with a maximum number of models [49]. The fig 12 gives a clear view of how the boosting technique works.
- **Voting Classifiers:** A voting classifier is an ensemble model that trains various models and forecasts based on aggregating the conclusions of each model are taken. In classification, the outcomes are produced by the preponderance vote of contributing models. The aggregating standards are an integrated conclusion of voting for every model outcome [52]. Either of the two methods performs the voting methods. Hard Voting: Voting on the expected output class and Soft Voting on the expected probability of the output class. We have used hard voting since it is purely based on each classifier's class labels and weights [53].
- **Stacking:** Stacking or Stack Generalization is one of the best ways of improving the accuracy of the predictive model. In fig 13 Stacking, the training dataset is taken and given to the

classifiers (Model 1, Model 2, Model 3) parallelly to get the predicted outcome of new stacked dataset. The newly built dataset is now fed into Level 1 in which the same models are applied once again to get the better trained dataset. This process of training is repeated till we get a better prediction with lesser loss. At the final level a single model is applied to the dataset that was built in the previous set. The final prediction is then made out of it [11].

```
Algorithm 1: STACKING
INPUT:
Cleansed Data set df;
Base Algorithms $X_r$(r = 1;2;3);
Meta Algorithm X; Number of Instances n;

PROCESS:
Step 1: Train all the base algorithms $X_r$ for learning with df.
for(r=1 to 3):
    Lr = X($df_r$)
Step 2: End
Step 3: Take a new dataset df $^0$ and classify it.
for(N=1 to n): for(r=1 to 3):
    $C_{ir}$ = $L_r$(Xi)
Step 4: New Data obtained is,
df 0 = df 0 U((zir; zir; :::zir); yi)
Step 5: End
Step 6: Use Meta Classifier to train the new data. L0 = L(df 0 )

OUTCOME : return
$L^0$ (L1(x);L2(x);L3(x))
```

We have taken three weakly correlated models every time in our work and performed the stacking with four-folds. So the best of all, worst of all, and combination of linear and ensemble learning models were used to understand and compare how the stacking prediction works [11], [53], [54].

DYNAMIC CLASSIFICATION METHODS

The dynamic ensemble is an ensemble learning approach that automatically selects a subset of ensemble algorithms during the classification period. Many machine learning prototypes were fitted to the training dataset in this process. Then the best model to predict a unique new instance is selected based on the expected components of the sample [55], [56].

The DES method can be performed using the knearest neighbor model to find the instance in the training dataset farthest from the expected new sample. Evaluating all the models in that particular neighborhood pool and using the model with the best performance in the neighborhood predicts the current criteria (classification classifies a collection of data into categories or classes) [6].

For our dataset we have applied K-Nearest Oracle Union (KNORA-U), k-Nearest Oracle Eliminate (KNORA-E), and DCS using OLA models.

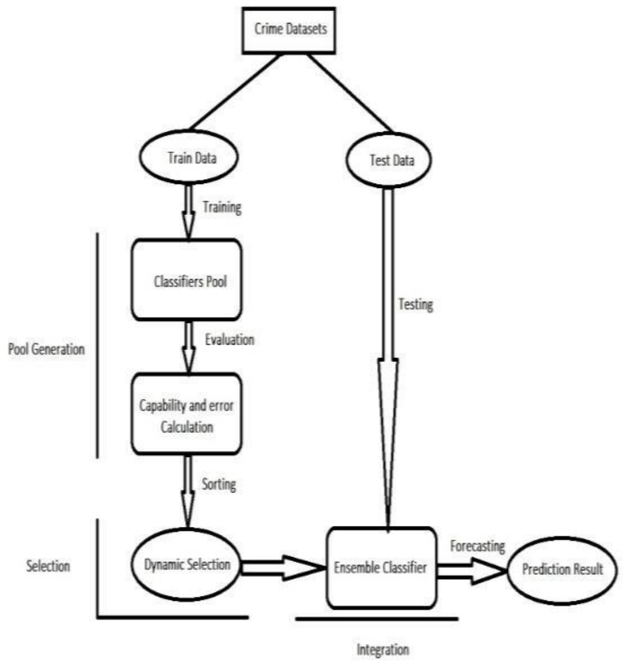


Fig 4: Dynamic Ensemble Classifiers

The fig 4 Dynamic ensemble classifiers represent how Dynamic Ensemble selection for classification works for Machine Learning algorithms [55]. Preprocessed training data is given for training to various models i.e., the pool of classifiers. The test data is used to make the classification. The classified label is then compared with the original label using KNN methodology and its accuracy is noted. This process is repeated foe each classifier in the pool and the model with high accuracy is selected. At the end the final prediction of the test data is made.

DCS With Overall Local Accuracy (OLA)
DCS-LA model is estimated using overall local accuracy on the artificial dataset. For our work, we will use default model hyperparameters, including bagged decision trees, as the collection of classifier models and the neighbor value is seven to select the local neighborhood during classification forecasting [48]. For the evaluation of the model, we did repeat stratified k-fold cross-validation with ten folds and three times repetitions. The results were taken using metrics like the mean and standard deviation of the accurateness of the model across all repeats and folds.

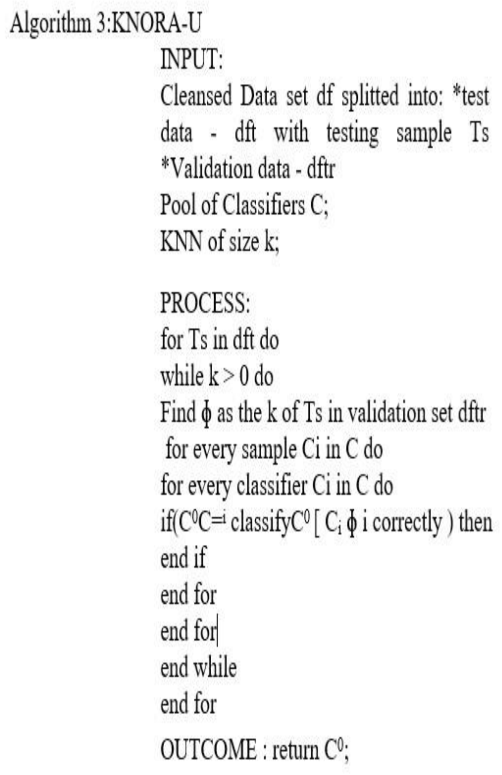
Algorithm 2: DYNAMIC CLASSIFIER(OLA)
INPUT:
Cleansed Data sets df1 and df2;
Base Algorithms X; KNN of size k;

PROCESS:
for t testing samples in df2 do train t with all Xi if(predicted label l == original label in all algorithms) return l; else
for every Xi do
Calculate OLA end for select
X is used to classify the data end if end for

OUTCOME : return Xi

Rather than discovering the most appropriate classifier, we pick the most appropriate ensemble for the individual sample. The idea of the K-nearestoracles (KNORA) is identical to the concepts of OLA, LCA, and the A Priori & A Posteriori techniques, considering the neighborhood of test patterns, while it can be differentiated from the rest by the immediate use of its possessions of holding training samples in the region with which to locate the most suitable ensemble for a given sample. For a given test data point, KNORA just discovers its closest K neighbors in the validation set, figures out which classifiers accurately classify those neighbors in that particular set, and utilizes them as the ensemble for classifying the provided pattern in that test set [57].

KNORA-U
K-Nearest Oracle Union (KNORA-U), the process determines all classifiers that perfectly categorize at most small one sample belonging to the region of competence of the query sample. Every classifier chosen has several votes equal to the number of samples in the region of competence that predicts the accurate label. The votes acquired by all ground classifiers are aggregated to obtain the last ensemble decision [58].



```
Algorithm 3:KNORA-U
    INPUT:
    Cleansed Data set df splitted into: *test data - dft with testing sample Ts
    *Validation data - dftr
    Pool of Classifiers C;
    KNN of size k;

    PROCESS:
    for Ts in dft do
    while k > 0 do
    Find $\phi$ as the k of Ts in validation set dftr
     for every sample Ci in C do
    for every classifier Ci in C do
    if($C^0C=^i$ classify$C^0$ [ $C_i$ $\phi$ i correctly ) then
    end if
    end for
    end for
    end while
    end for
    OUTCOME : return $C^0$;
```

[illegible]

X = argmaxi(OLAi)

KNORA-E
The KNORA-E process probes for a provincial Oracle, a ground classifier that accurately classifies all samplings belonging to the area of competence of the test data. All classifiers with ideal performance in the region of competence are selected (local Oracles). Suppose no classifier performs to perfect accuracy. In that case, the size of the competence region is lowered (by dragging the most distant neighbor), and the performance of the classifiers is re-evaluated. The outcomes of the selected ensemble of classifiers are integrated using the majority voting procedure. The entire pool is employed for classification if no base classifier is selected [59].

```
Algorithm 4: KNORA-E
INPUT:
Cleansed Data set df splitted into:
*test data - dft with testing sample Ts
*Validation data - dftr
Pool of Classifiers C;
KNN of size k;

PROCESS:
for Ts in dft do
while k > 0 do
Find $\phi$ as the k of $T_s$ in validation in set $df_{tr}$
For each classifier $C_i$ in C do
If ($C_i$ identifies all sample correctly in $\phi$) then
C' = C' U $C_i$;
End if
End for
If (C' == $\phi$) then
Reduce k by 1
Else
Break;
End if
End while
If(C' == $\phi$) then
Ci with correct classification:
Choose the Ci to construct the ensemble C':
End if
End for
Outcome: return C'
```

The fig 5, Difference between KNORA-E and KNORA-U, below shows the difference between KNORA-E and KNORA-U. The KNORA-E on the left only employs classifiers that accurately classify all the K-nearest patterns whereas the KNORA-U employs classifiers that precisely classify any of the K-nearest patterns. The test pattern is shown as a pentagon on the feature side, validation data points are shown as circles and the five nearest validation points are darkened. The used classifiers i.e., the intersection of accurate classifiers is shaded on the right side. The fig 15 Difference between KNORAE and KNORA-U, clearly depicts in KNORA-E, the result is obtained by intersection of the pool of classifiers from the features that are selected whereas in KNORA U, union of the pool of classifiers serves as the result.

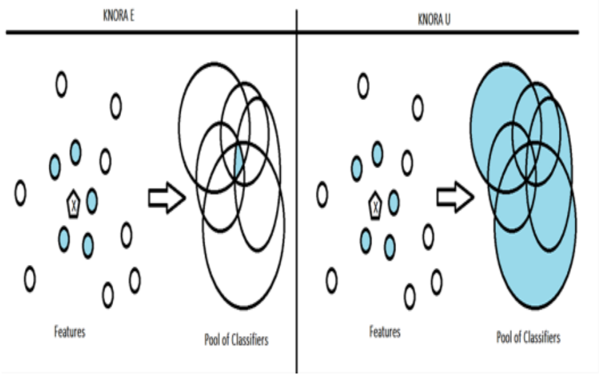


Fig 5: Difference between KNORA-E and KNORA-U

4. RESULT

Evaluating the performance of an algorithm using metrics is part of all ML pipelines. These indicators indicate whether or not progress is indicated by displaying them in a numerical format. All machine learning models require a measured value to estimate their performance, from basic linear models to complex models [60].

Metrics are utilized to observe and estimate the performance of the model (both in the training dataset and test dataset). Every machine learning task is either a Regression or Classification task. Many metrics are available for both problems.

Metrics and loss function are two different terms. Loss functions deliver a skeletal measure of the model's performance, mainly employed to train a machine learning model (with the help of Optimization algorithms. For example, Gradient Descent). The metrics for each algorithm are usually differentiable in the model's parameters [95].

Since we went with classification models in this work, we will focus on classification metrics. Classification models will likely have a discrete outcome; thus, we need a metric that compares discrete classes. Classification Metrics assess the model's performance and give a result of how adequate or inadequate the classification is, but each assumes it in a distinguishable way. The primary metrics lie in the confusion matrix to measure performance. The outcome is two or more classes in the tabular format with expected and actual value combinations [6].

The confusion matrix has:

True Positive states that the prediction is optimistic and it's true.

True Negative is for the prediction that is negative but true.

False Positive is predicted positive, but it is false.

False Negative is a prediction that is both negative and false.

Some of the important metrics used in our work include [61]:

Accuracy: Accuracy entirely calculates how frequently the classifier's predictions are correct. Accuracy, in short, is defined as the percentage of the number of accurate predictions and the total number of predictions. When a particular model shows an accuracy of a higher rate, like 99 or 100 percent, we might think that prototype we created is functioning very well. But this is not consistently correct and can be deceiving in a few situations, so it is better to check with other metrics.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (1)$$

[90]

Recall: The ratio of the number of Favorable samples correctly classified as Favorable to the total number of Favorable samples is called the Recall. The Recall estimates the ML model's capability to catch the positive samples. The more elevated the Recall, the better positive samples detected [69].

Table 3 (a): brief summary about the result of all bench mark datasets 1

$$Recall = \frac{TP}{TP+FN} \quad (2)$$

It is autonomous of the number of negative sample types. Additionally, if the model classifies all positive samples exactly as positive, the value of Recall will be 1.

Precision is the ratio of predicted favorable observations to the total predicted positive observations. Precision denotes the percentage of your relevant outcomes. In different words, it can be expressed as the ratio of precisely classified complimentary samples (which is True Positive) to the total number of classified positive samples (the samples can be either correct or incorrect). Precision permits visualizing the dependability of the machine learning model in classifying the model as positive [69].

$$Precision = \frac{TP}{TP+FP} \quad (3)$$

The precision of the machine learning model will be high when the Value of True positive in the numerator is greater than the denominator with the sum of true positive and False Positive.

F1 Score: The F1 score incorporates precision and Recall relative to a typical positive class. The F1 score is also known as F-measure, which shows the balance between precision and Recall. The F1 score analyzes the harmonic mean of precision and Recall. F1 score gets its best value at one and worst at zero.

$$F1-Score = \frac{TP}{(TP+(1/2)*(FP+FN)}$$

$$= 2 * [\frac{precision * recall}{precidion + recall}] \quad (4)$$

Throughout the result section we have mentioned the datasets Chicago, San Francisco, Vancouver, Pheonix and Boston as Dataset 1, Dataset 2, Dataset 3, Dataset 4, and Dataset 5 respectively.

Result of Basic Classification algorithms:

As stated in the methodology section, five basic machine learning algorithms were used and the results were compared using the above metrics. Table 3 below briefly summarizes and fig 16 visualization of the result of the 5 machine learning algorithms used on our five different crime datasets.

Table 2: Experimental algorithms with various CVs and their accuracy

DATASET	ALGORITHM	WITHOUT CV (ACCURACY)	K-FOLD	No of splits
Dataset 1	Decision Tree	99.7669 %	99.7426 %	5
	Logistic Regression	26.0838 %	36.0868 %	5
	KNN(1)	67.5889 %	66.2719 %	5
	Naïve Bayes	37.0481 %	36.9000 %	5
	Random Forest	90.0209 %	88.7833 %	5
Dataset 2	Decision Tree	99.3317 %	90.1749 %	5
	Logistic Regression	81.3462 %	90.9824 %	5
	KNN(25)	83.1342 %	84.2829 %	5
	Naïve Bayes	90.2841 %	90.2393 %	5
	Random Forest	84.9633 %	83.0355 %	5
Dataset 3	Decision Tree	38.6556 %	38.2793 %	5
	Logistic Regression	37.2604 %	37.1099 %	5
	KNN(1)	48.8449 %	43.9997 %	5
	Random Forest	37.2604 %	46.6102 %	5
Dataset 4	Decision Tree	46.1417%	45.9404%	5
	Logistic Regression	51.7334%	51.8499%	5
	KNN(25)	50.9153%	51.0480%	5
	Naïve Bayes	52.8070%	52.9335%	5
	Random Forest	55.5888%	55.2171%	5
Dataset 5	Decision Tree	49.2648%	49.2351%	5
	Logistic Regression	45.1613%	44.6889%	5
	KNN(25)	53.7710%	52.5882%	5
	Naïve Bayes	16.3760%	16.2652%	5
	Random Forest	77.5409%	76.8182%	5

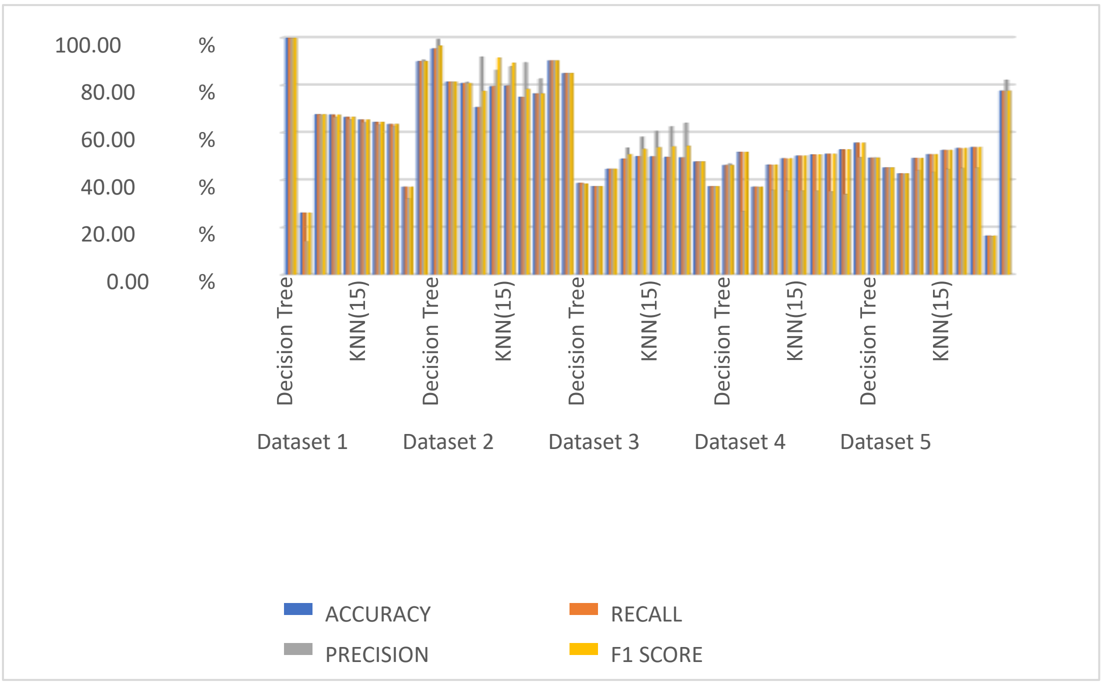


Fig 6: Visualization of the result of basic classification algorithms

Performance results post Cross-validation:
Further, the hyper parameter adjustments were made with different cross-validation methods. The following results are obtained, which are shown in table 2, table 3, table 4, and table 5.

Performance results with K-Fold CV:
K-fold Cross-validation helps in reducing overfitting. Fig 7 depicts the visualization of various ML algorithm's accuracy with k-fold CV. As for the accuracy improvement, a minimal increase was observed after K-fold Cross-validation.

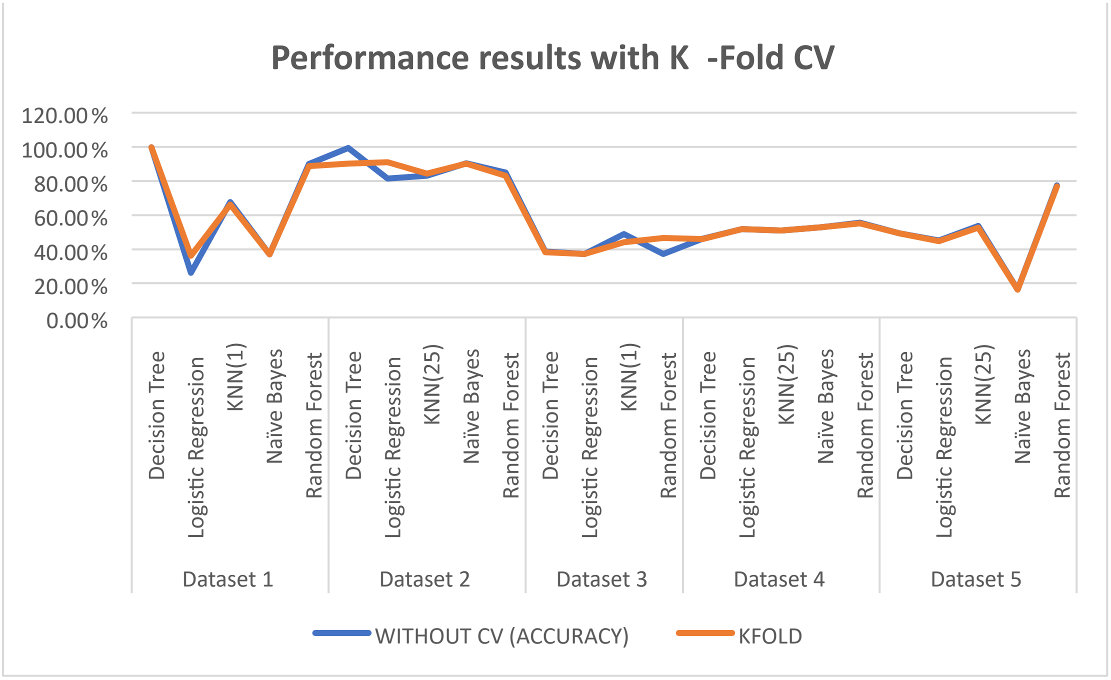


Fig 7: Visualization of the result of different classification algorithms with K-fold CV

Performance results with Stratified K-Fold CV:
Stratified K-Fold Cross Validation returns the stratified sampling folds and is the variant of K-fold illustrated in fig 8. To overcome the random sampling issue, this Stratified k-fold is used, and we have observed mixed results, i.e., in some cases, the accuracy has increased, but in a few, it has depreciated.

Table 3: Experimental result of various algorithms with SKCV

DATASET	ALGORITHM	WITHOUT CV (ACCURACY)	STRATIFIED KFOLD	No of splits
Dataset 1	Decision Tree	99.7669 %	99.7468 %	5
	Logistic Regression	26.0838 %	36.1322 %	5
	KNN(1)	67.5889 %	66.2979 %	5
	Naïve Bayes	37.0481 %	36.9109 %	5
	Random Forest	90.0209 %	89.0557 %	5
Dataset 2	Decision Tree	99.3317 %	90.2125 %	5
	Logistic Regression	81.3462 %	90.9972 %	5
	KNN(25)	83.1342 %	84.3244 %	5
	Naïve Bayes	90.2841 %	90.1516 %	5
	Random Forest	84.9633 %	82.9972 %	5
Dataset 3	Decision Tree	38.6556 %	38.4288 %	5
	Logistic Regression	37.2604 %	37.1099 %	5
	KNN(1)	48.8449 %	43.9988 %	5
	Random Forest	37.2604 %	46.5523 %	5
Dataset 4	Decision Tree	46.1417%	45.8936%	5
	Logistic Regression	51.7334%	51.8499%	5
	KNN(25)	50.9153%	51.0436%	5
	Naïve Bayes	52.8070%	52.9492%	5
	Random Forest	55.5888%	55.1983%	5
Dataset 5	Decision Tree	49.2648%	49.6568%	5
	Logistic Regression	45.1613%	45.0441%	5
	KNN(25)	53.7710%	52.5255%	5
	Naïve Bayes	16.3760%	16.2053%	5
	Random Forest	77.5409%	77.0353%	5

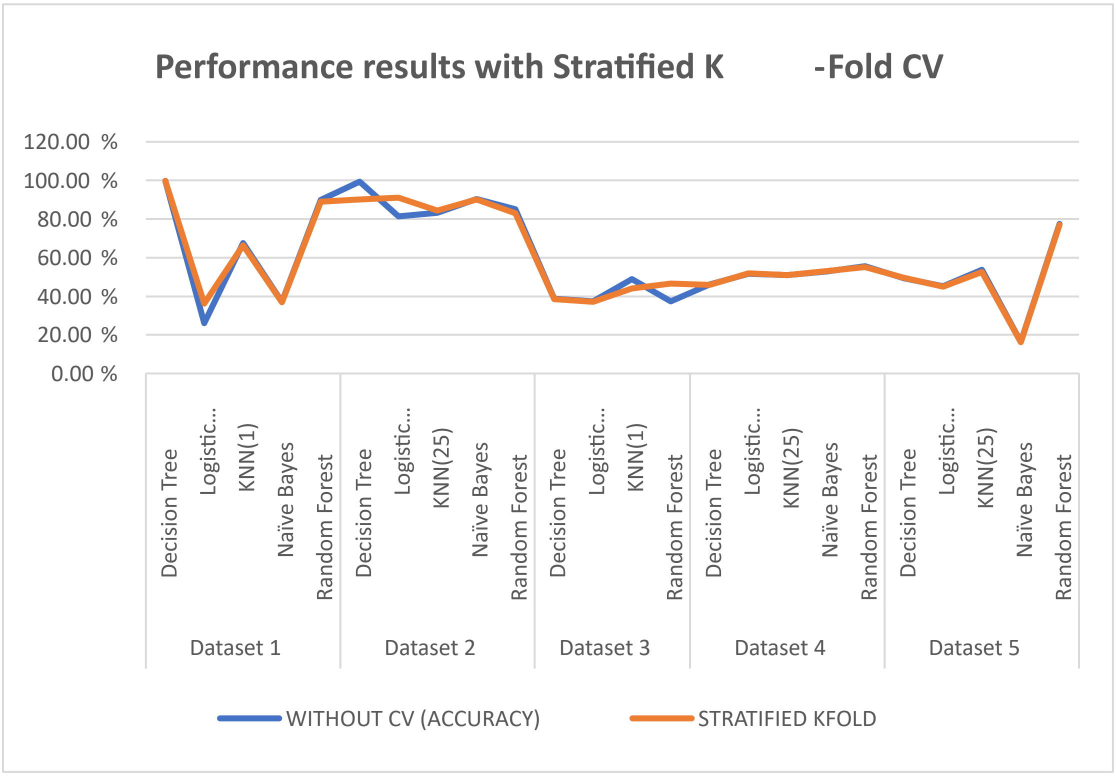


Fig 8: Visualization of the result of different classification algorithms with S K-fold CV

Performance results with Repeated K-Fold CV: Repeatedly applying the K-folds, which select different folds per each repeat, on the datasets has shown similar kinds of results as other k-fold, depicted in fig 9.

Table 4: Experimental result of various ML algorithms with RK-fold

Dataset	Algorithm	Without CV (accuracy)	Repeated k fold	No of splits	No of repeats
Dataset 1	Decision Tree	99.7669 %	99.7436 %	5	3
	Logistic Regression	26.0838 %	36.1051 %	5	3
	KNN(1)	67.5889 %	66.9345 %	5	3
	Naïve Bayes	37.0481 %	36.8922 %	5	3
	Random Forest	90.0209 %	89.2344 %	5	3
Dataset 2	Decision Tree	99.3317 %	92.9612 %	5	3
	Logistic Regression	81.3462 %	90.9849 %	5	3
	KNN(25)	83.1342 %	85.1103 %	5	3
	Naïve Bayes	90.2841 %	90.1977 %	5	3
	Random Forest	84.9633 %	83.4399 %	5	3
Dataset 3	Decision Tree	38.6556 %	38.7062 %	5	3
	Logistic Regression	37.2604 %	37.1099 %	5	3
	KNN(1)	48.8449 %	44.2014 %	5	3
	Random Forest	37.2604 %	46.7735 %	5	3
Dataset 4	Decision Tree	46.1417%	46.0904%	5	3
	Logistic Regression	51.7334%	51.8500%	5	3
	KNN(25)	50.9153%	51.0975%	5	3
	Naïve Bayes	52.8070%	52.9493%	5	3
	Random Forest	55.5888%	55.8168%	5	3
Dataset 5	Decision Tree	49.2648%	49.9095%	5	3
	Logistic Regression	45.1613%	44.8592%	5	3
	KNN(25)	53.7710%	53.2139%	5	3
	Naïve Bayes	16.3760%	16.1825%	5	3
	Random Forest	77.5409	77.8618	5	3

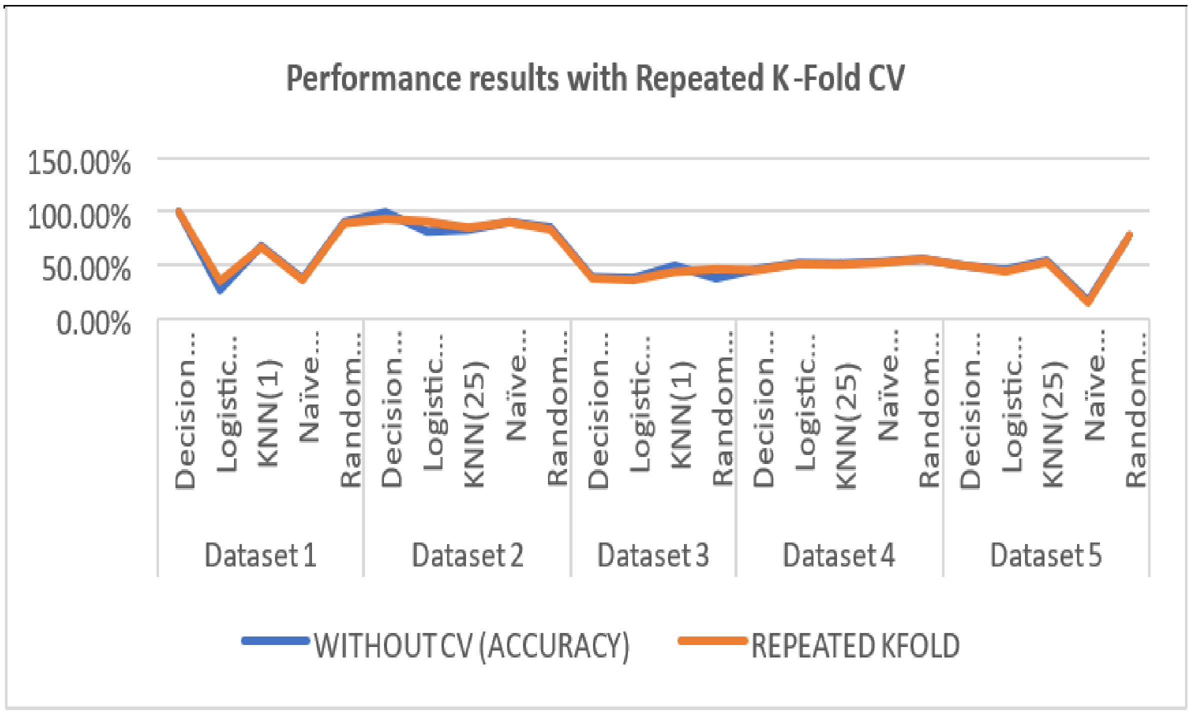


Fig 9: Visualization of the result of different classification algorithms with R K-fold CV

Performance results with shuffle split CV: validation methods. Fig 10 illustrates the accuracy of various ML algorithms with shuffle split CV. Shuffle Split brings out various indices each time. The result is relatively better than other cross-

Table 5: Experimental result of various ML algorithms with shuffle split

DATASET	ALGORITHM	WITHOUT CV (ACCURACY)	SHUFFLE SPLIT	Train size	Test size	No of splits
Dataset 1	Decision Tree	99.7669 %	99.5962 %	50 %	30 %	10
	Logistic Regression	26.0838 %	35.9979 %	50 %	30 %	10
	KNN(1)	67.5889 %	63.4134 %	50 %	30 %	10
	Naïve Bayes	37.0481 %	36.9749 %	50 %	30 %	10
	Random Forest	90.0209 %	87.9633 %	50 %	30 %	10
Dataset 2	Decision Tree	99.3317 %	81.7428 %	50 %	30 %	10
	Logistic Regression	81.3462 %	90.9686 %	50 %	30 %	10
	KNN(25)	83.1342 %	82.4188 %	50 %	30 %	10
	Naïve Bayes	90.2841 %	90.3474 %	50 %	30 %	10
	Random Forest	84.9633 %	81.6109 %	50 %	30 %	10
Dataset 3	Decision Tree	38.6556 %	37.3134 %	50 %	30 %	10
	Logistic Regression	37.2604 %	37.1117 %	50 %	30 %	10
	KNN(1)	48.8449 %	43.1021 %	50 %	30 %	10
	Random Forest	37.2604 %	45.8099 %	50 %	30 %	10
Dataset 4	Decision Tree	46.1417%	45.6019%	50 %	30 %	10
	Logistic Regression	51.7334%	51.8137%	50 %	30 %	10
	KNN(25)	50.9153%	51.1267%	50 %	30 %	10
	Naïve Bayes	52.8070%	52.9293%	50 %	30 %	10
	Random Forest	55.5888%	54.8771%	50 %	30 %	10
Dataset 5	Decision Tree	49.2648%	46.0970%	50 %	30 %	10
	Logistic Regression	45.1613%	44.9350%	50 %	30 %	10
	KNN(25)	53.7710%	49.9125%	50 %	30 %	10
	Naïve Bayes	16.3760%	19.9452%	50 %	30 %	10
	Random Forest	77.5409%	74.2253%	50 %	30 %	10

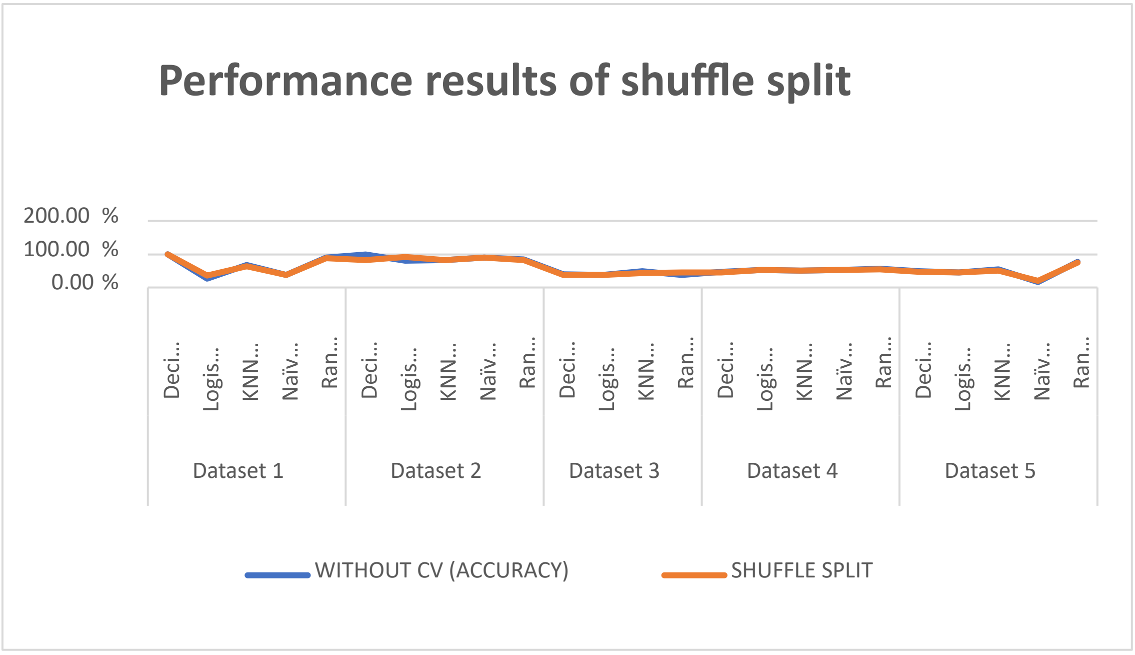


Fig 10: Visualization of the result of different classification algorithms with shuffle split

Performance analysis of Ensemble Classifiers: Stacking:

Table 6 and fig 11 represents the results obtained from each dataset applying stack generalization. From the following table, we observed that stacking an ensemble of the three best classifier models combined with Best among all meta-models yields a better result. When weak learners have stacked up with the best learner metamodel, it has improved the accuracy at a great rate. The combinations hence prove to be more powerful than the single classification model. The stacking is traditionally performed on stacking the weak learners and using the strong meta models to attain better results. But here to explore the behavior of the stacking process. Whether it shows the same improvement while stacking the best performers. Observation shows that in all aspects of combination, the stacking has performed pretty well compared to basic models.

Table 6: performance analysis of dynamic ensemble classifier

Dataset	Model1	Model2	Model3	Meta Model	Acc	Recall	Precision	F1 score
Dataset 1	KNN	DT	RF	DT	99.69%	99.69%	99.69%	99.69%
Dataset 1	KNN	DT	GB	DT	99.94%	99.94%	99.94%	99.94%
Dataset 1	KNN	LR	NB	KNN	64.21%	64.21%	64.46%	64.21%
Dataset 3	MLP	RF	LR	LR	84.70%	84.70%	84.70%	84.70%
Dataset 3	DT	RF	NB	NB	96.12%	96.12%	96.12%	96.12%
Dataset 3	KNN	LR	NB	NB	86.39%	86.39%	86.39%	86.39%
Dataset 2	MLP	RF	LR	LR	76.74%	76.74%	76.74%	76.74%
Dataset 2	DT	RF	GB	DT	98.88%	98.88%	98.88%	98.88%
Dataset 2	KNN	DT	MLP	NB	92.92%	92.92%	92.92%	92.92%
Dataset 4	KNN	LR	NB	RF	51.73%	51.73%	51.73%	51.73%
Dataset 4	GB	RF	LR	RF	56.14%	56.14%	48.76%	56.14%
Dataset 4	NB	DT	KNN	RF	53.80%	53.80%	38.60%	53.80%
Dataset 5	DT	RF	KNN	RF	74.80%	74.80%	75.48%	74.80%
Dataset 5	LR	KNN	NB	DT	56.06%	56.06%	46.09%	56.06%

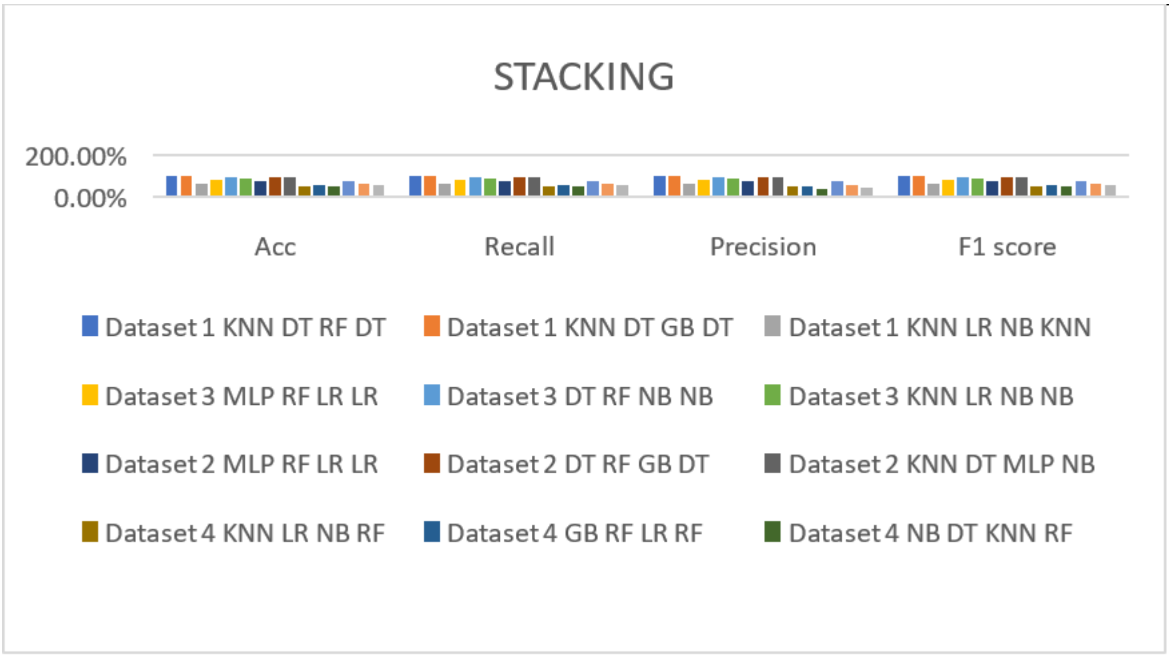


Fig 11: Visualization of the performance of different benchmark datasets with Stacking

Voting: hard voting, we can observe a little increase in decimal values of accuracy percentage. The further hard vote is selected over Soft

Table 7 and fig 12 below shows the voting because not all the models are observation of the Hard Voting technique suitable or work well with the probability applied in our work for the selected distribution. datasets. After validating the models using

Table 7: Result with Hard voting techniques

Dataset	Model 1	Model 2	Model 3	Acc	Recall	Precision	F1 score
Dataset 1	KNN	DT	RF	96.22%	96.22%	96.20%	96.22%
Dataset 3	KNN	DT	NB	96.81%	96.81%	96.81%	96.81%
Dataset 2	KNN	DT	NB	91.24%	91.24%	91.24%	91.24%
Dataset 4	NB	LR	RF	53.52%	53.52%	35.01%	53.52%
Dataset 5	KNN	DT	RF	67.69%	67.69%	77.48%	67.69%

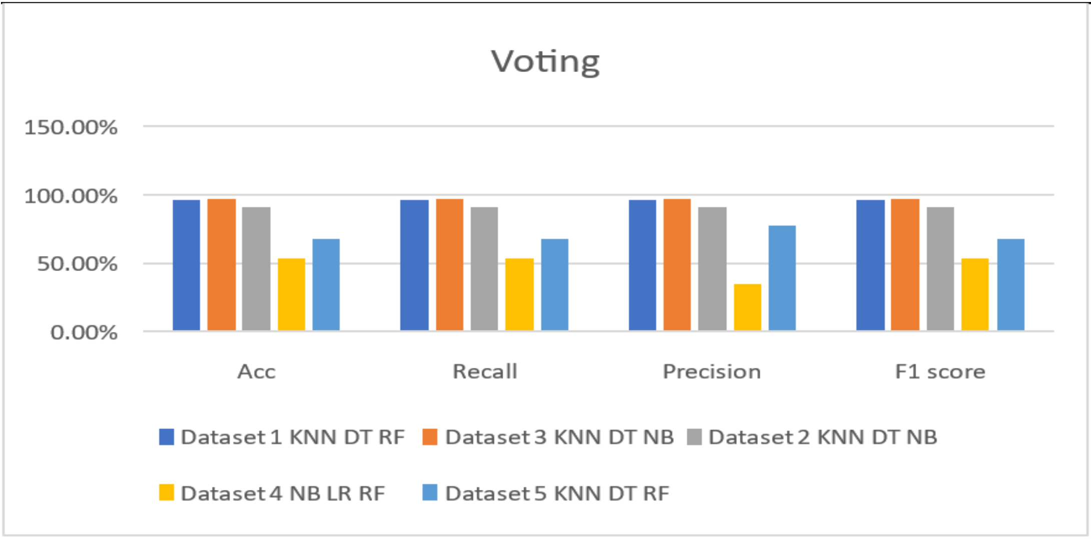


Fig 12: Visualization of the performance of different benchmark datasets with Voting

Boosting:
In boosting, adaptive boosting and gradient boosting were applied to all the datasets, and the following results shown in table 8 is obtained. For Dataset 2, the gradient boosting technique didn't work due to the overfitting because of adding too many trees.

Table 8: Comparison of Adaptive boosting and Gradient boosting techniques.

Dataset	Adaptive Boosting				Gradient Boosting			
	Acc	Recall	Precision	F1 score	Acc	Recall	Precision	F1 score
Dataset 1	33.68%	33.68%	19.52%	30.68%	99.88%	99.88%	99.88%	99.88%
Dataset 3	78.60%	78.60%	78.60%	78.60%	90.23%	90.23%	90.23%	90.23%
Dataset 2	87.36%	87.36%	87.36%	87.36%	-	-	-	-
Dataset 4	53.36%	53.36%	41.89%	53.36%	55.91%	55.91%	48.88%	55.91%
Dataset 5	54.54%	54.54%	44.50%	54.54%	68.49%	68.49%	73.87%	68.49%

Fig 13, depicts the performance of Adaptive and Gradient boosting in various benchmark datasets.

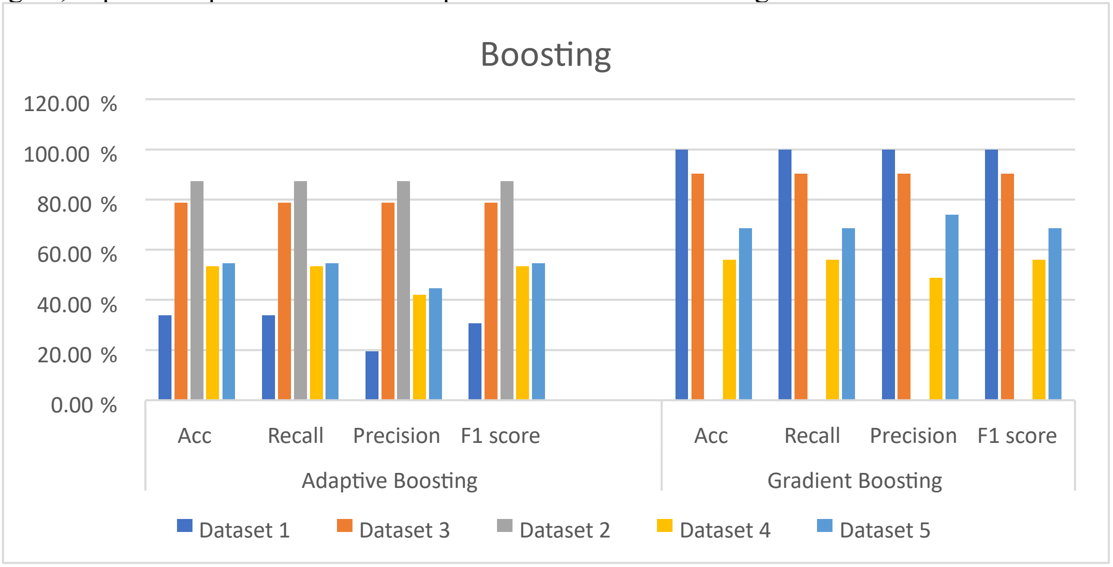


Fig 13: Visualization of the performance of different benchmark datasets with Boosting (Adaptive and Gradient boosting)

Result of Dynamic Classifier and Dynamic Ensemble Classifier algorithms:
The dynamic ensemble algorithm works better than the other algorithms discussed in this research because members are selected just in time, depending on the specific input pattern that requires forecasting. Table 9 and Fig 14 show the result of various dynamic classifiers (OLA) and dynamic ensemble algorithms used for the different datasets that we have taken:

Table 9: Results of various dynamic classifiers

DATASET	OLA	KNORA-E	KNORA-U
Dataset 1	99.5%	99.8%	93.6%
Dataset 2	95.7%	96.3%	96.1%
Dataset 3	78.1%	85.7%	82.6%
Dataset 4	46.3%	48.5%	93.6%
Dataset 5	49.1%	52.6%	93.5%

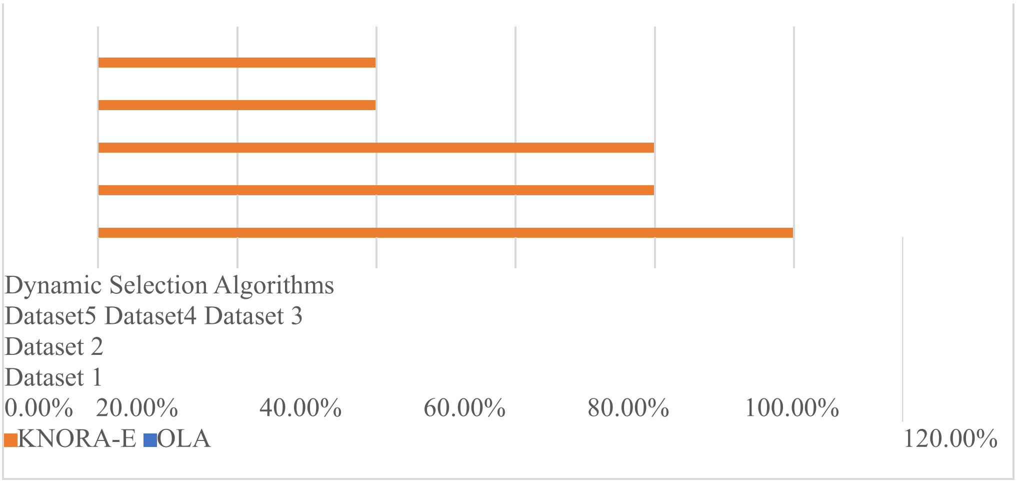


Fig 14: Visualization of the performance of different benchmark datasets with dynamic selection algorithms

The DES is promising because it relies entirely on selecting the best competent ensemble classifier for forecasting each split of sample data. Moreover, it performs all the integration, pooling, and selection independently. The result further shows that dynamic ensemble selection can perform better than any single model in the pool and is more beneficial than averaging all the static ensemble selections

5. CONCLUSION

We experimented with many machine learning models for classifying crime. The evaluation of these ML classifiers was done mainly in terms of accuracy. Overall, our results reveal a powerful impact of the dynamic algorithms, i.e., KNORA-U, and KNORA-E algorithms, achieved a reliable accuracy above 90% in classifying and identifying the crime types or categories. This outcome is anticipated because of the selection of the classifiers pool to predict every sample test data split. For every dataset apt classifier is chosen automatically without any pre-fixation. This research has also paved a path to discussing many more general learnings. Not all the data can be compatible with the desired algorithm, which has performed well with other data. Hence the algorithms can create either positive or negative on their performances on a particular dataset. Data pre-processing steps like feature engineering, outliers removal, and hyperparameter tuning have played a vital role in better accurate crime classification. When applied to the data, ensemble algorithms yield better accuracy than single algorithms. After several trials, briefly translated, our findings indicate that the Stacking of the algorithms, irrespective of their performance individually, can outperform the algorithm's accuracy in forecasting the test data to a reasonable extent. Altogether, our outcomes reinforce the essence of choosing dynamic ensemble classification algorithms for the crime domain, which in turn helps save many unfortunate situations that are yet to happen. Future studies could fruitfully explore the possibility of using Deep learning algorithms further by using them dynamically, which can handle larger quantities of data. Additionally, web and mobile applications can be built to be used by law enforcement to update crime data and analyze and forecast the same.

REFERENCE

1. S. Ahmed, M. Gentili, D. SierraSosa, and A. S. Elmaghraby, “Multi-layer data integration technique for combining heterogeneous crime data,” Inf. Process. Manag., vol. 59, no. 3, p. 102879, 2022, doi:10.1016/j.ipm.2022.102879.
2. M. Jangra, M.-T. Cse, and S. Kalsi, “Naïve Bayes Approach for the Crime Prediction in Data Mining,” 2019.
3. J. Q. Yuki, M. Mahfil Quader Sakib, Z. Zamal, K. M. Habibullah, and A. K. Das, “Predicting crime using time and location data,” in Pervasive Health: Pervasive Computing Technologies for Healthcare, Jul. 2019, pp. 124–128, doi: 10.1145/3348445.3348483.

4. B. S. Aldossari et al., "A comparative study of decision tree and naive bayes machine learning model for crime category prediction in chicago," in Pervasive Health: Pervasive Computing Technologies for Healthcare, Jan. 2020, pp. 34–38, doi:10.1145/3379247.3379279
5. N. Qazi and B. L. W. Wong, "An interactive human centered data science approach towards crime pattern analysis," Inf. Process. Manag., vol. 56, no. 6, 2019, doi: 10.1016/j.ipm.2019.102066.
6. A. H. Wibowo and T. I. Oesman, "The comparative analysis on the accuracy of kNN, Naive Bayes, and Decision Tree Algorithms in predicting crimes and criminal actions in Sleman Regency," in Journal of Physics: Conference Series, Mar. 2020, vol. 1450, no. 1, doi: 10.1088/1742-6596/1450/1/012076.
7. S. Albahli, A. Alsaqabi, F. Aldhubayi, H.T. Rauf, M. Arif, and M. A. Mohammed, "Predicting the type of crime: Intelligence gathering and crime analysis," Comput. Mater. Contin., vol. 66, no. 3, pp. 2317– 2341, 2020, doi: 10.32604/cmc.2021.014113.
8. L. A. Sri, K. Manvitha, G. Amulya, I. S. Sanjuna, and V. Pavani, "FBI CRIME ANALYSIS AND PREDICTION USING MACHINE LEARNING," vol. 11, 2020, [Online]. Available: www.jespublication.com.
9. H. Wang and S. Ma, "Preventing crimes against public health with artificial intelligence and machine learning capabilities," Socioecon. Plann. Sci., 2021, doi: 10.1016/j.seps.2021.101043.
10. W. Safat, S. Asghar, and S. A. Gillani, "Empirical Analysis for Crime Prediction and Forecasting Using Machine Learning and Deep Learning Techniques," IEEE Access, vol. 9, pp. 70080–70094, 2021, doi: 10.1109/ACCESS.2021.3078117.
11. S. S. Kshatri, D. Singh, B. Narain, S. Bhatia, M. T. Quasim, and G. R. Sinha, "An Empirical Analysis of Machine Learning Algorithms for Crime Prediction Using Stacked Generalization: An Ensemble Approach," IEEE Access, vol. 9, pp. 67488–67500, 2021, doi: 10.1109/ACCESS.2021.3075140.
12. M. R. Khatun, S. I. Ayon, M. R. Hossain, and M. J. Alam, "Data mining technique to analyse and predict crime using crime categories and arrest records," Indones. J. Electr. Eng. Comput. Sci., vol. 22, no. 2, 2021, doi: 10.11591/ijeecs.v22.i2.pp10521060.
13. Z. R. Tembusai, H. Mawengkang, and M. Zarlis, "K-Nearest Neighbor with K-Fold Cross Validation and Analytic Hierarchy Process on Data Classification," Int. J. Adv. Data Inf. Syst., vol. 2, no. 1, Jan. 2021, doi: 10.25008/ijadis.v2i1.1204.
14. A. Tamir, E. Watson, B. Willett, Q. Hasan, and J.-S. Yuan, "Crime Prediction and Forecasting using Machine Learning Algorithms," 2021. [Online]. Available: https://www.researchgate.net/publication/355872171.
15. M. Khan, A. Ali, and Y. Alharbi, "Predicting and Preventing Crime: A Crime Prediction Model Using San Francisco Crime Data by Classification Techniques," Complexity, vol. 2022, 2022, doi: 10.1155/2022/4830411.
16. A. C. M. da Silveira, Á. Sobrinho, L. D. da Silva, E. de B. Costa, M. E. Pinheiro, and A. Perkusich, "Exploring Early Prediction of Chronic Kidney Disease Using Machine Learning Algorithms for Small and Imbalanced Datasets," Appl. Sci., vol. 12, no. 7, Apr. 2022, doi: 10.3390/app12073673.
17. S. More, S. Mench, S. Kuge, H. Bagwan, and A. Professor, "Crime Prediction Using Machine Learning Approach," Int. J. Adv. Res. Comput. Commun. Eng., vol. 10, no. 5, 2021, doi:10.17148/IJARCCE.2021.10537.
18. Keerthi R, Kirthika B, Pavithraa S, and V. Gowri, "PREDICTION of CRIME RATE ANALYSIS using MACHINE LEARNING APPROACH," Int. Res. J. Eng. Technol., 2020.
19. Z. Wang and J. Wang, "Applications of Machine Learning in Public Security Information and Resource Management," Sci. Program., vol. 2021, 2021, doi: 10.1155/2021/4734187.
20. "Machine Learning and the Internet of Medical Things in Healthcare - Google Books." Accessed: Jan. 09, 2023. [Online]. Available:https://books.google.co.in/books?hl=en&lr=&id=wHQJEAAAQBAJ &oi=fnd&pg=PP1&dq=Saravanan,+Vijayalakshmi,+and+Singh,+Akansha.+Machine+Learning+%26+I Internet +of+ +Things+(IoT)+for+Urban+Intelligence.+In+Machine+Learning+%26+Internet +of+Things+(IoT)+ for+ Urban+Inte.
21. C. Hale and F. Liu, "CS 229 Project Report: San Francisco Crime Classification." S. Hossain, A. Abtahee, I. Kashem, M. M. Hoque, and I. H. Sarker, "Crime Prediction Using Spatio-Temporal Data," Mar. 2020, [Online]. Available: http://arxiv.org/abs/2003.09322.
22. Y. Abouelnaga, "San Francisco Crime Classification," Jul. 2016, [Online]. Available: http://arxiv.org/abs/1607.03626.
23. I. Pradhan, M. Eirinaki, K. Potika, and P. Potikas, "Exploratory data analysis and crime

prediction for smart cities," 2019, doi: 10.1145/3331076.3331114.
24. S. A. Alasadi and W. S. Bhaya, "Review of data preprocessing techniques in data mining," J. Eng. Appl. Sci., vol. 12, no. 16, 2017, doi: 10.3923/jeasci.2017.4102.4107.
25. R. M. O. Cruz, L. G. Hafemann, R. Sabourin, and G. D. C. Cavalcanti, "DESlib: A dynamic ensemble selection library in python," J. Mach. Learn. Res., vol. 21, 2020.

a. J. Wang, X. Wang, Y. Yang, H. Zhang, and Fang, "A review of data cleaning methods for web information system," Computers, Materials and Continua, vol. 62, no. 3. 2020, doi:10.32604/cmc.2020.08675.

26. B. Calabrese, "Data cleaning," in Encyclopedia of Bioinformatics and Computational Biology: ABC of Bioinformatics, vol. 1–3, 2018.
27. X. Wang and C. Wang, "Time Series Data Cleaning: A Survey," IEEE Access, vol. 8, 2020, doi: 10.1109/ACCESS.2019.2962152.
28. B. Doshi, "Handling Missing Values in Data Mining," Cs.Rit.Edu, 2011.
29. T. Maddileti, V. Sai Madhav, S. Sashank, and G. S. Rao, "Crime Data Analysis Using Machine Learning Models," Int. J. Adv. Sci. Technol., vol. 29, no. 9s, 2020.
30. J. Josse and F. Husson, "Handling missing values in exploratory multivariate data analysis methods," J. la Société Française …, vol. 153, no. 2, 2012.
31. T. F. Johnson, N. J. B. Isaac, A. Paviolo, and M. González-Suárez, "Handling missing values in trait data," Glob. Ecol. Biogeogr., vol. 30, no. 1, 2021, doi: 10.1111/geb.13185.
32. M. Vink, N. Netten, M. S. Bargh, S. Van Den Braak, and S. Choenni, "Mapping crime descriptions to law articles using deep learning," 2020, doi: 10.1145/3428502.3428507.
33. J. M. Chiou, Y. C. Zhang, W. H. Chen, and W. Chang, "A functional data approach to missing value imputation and outlier detection for traffic flow data," Transp. B, vol. 2, no. 2, 2014, doi: 10.1080/21680566.2014.892847.
34. O. Ghorbanzadeh et al., "Gully erosion susceptibility mapping (GESM) using machine learning methods optimized by the multi-collinearity analysis and K-fold cross-validation," Geomatics, Nat. Hazards Risk, vol. 11, no. 1, 2020, doi: 10.1080/19475705.2020.1810138.
35. K. Singh and S. Upadhyaya, "Outlier Detection: Applications And Techniques.," Int. J. Comput. …, vol. 9, no. 1, 2012.
36. A. Hamdi, K. Shaban, A. Erradi, A. Mohamed, S. K. Rumi, and F. D. Salim, "Spatiotemporal data mining: a survey on challenges and open problems," Artif. Intell. Rev., vol. 55, no. 2, 2022, doi: 10.1007/s10462-021-09994-y.
37. T. D. K. Thara, P. S. Prema, and F. Xiong, "Auto-detection of epileptic seizure events using deep neural network with different feature scaling techniques," Pattern Recognit. Lett., vol. 128, 2019, doi: 10.1016/j.patrec.2019.10.029.
38. X. Wan, "Influence of feature scaling on convergence of gradient iterative algorithm," in Journal of Physics: Conference Series, 2019, vol. 1213, no. 3, doi: 10.1088/1742-6596/1213/3/032021.
39. U. M. Khaire and R. Dhanalakshmi, "Stability of feature selection algorithm: A review," Journal of King Saud University - Computer and Information Sciences, vol. 34, no. 4. 2022, doi: 10.1016/j.jksuci.2019.06.012.
40. J. Li et al., "Feature selection: A data perspective," ACM Computing Surveys, vol. 50, no. 6. 2017, doi: 10.1145/3136625.
41. N. Pilnenskiy and I. Smetannikov, "Feature selection algorithms as one of the python data analytical tools," Futur. Internet, vol. 12, no. 3, 2020, doi: 10.3390/fi12030054.
42. Z. Wang, Z. Song, and T. Zhou, "Machine learning for ionic liquid toxicity prediction," Processes, vol. 9, no. 1, 2021, doi: 10.3390/pr9010065.
43. A. Nurhopipah and U. Hasanah, "Dataset Splitting Techniques Comparison For Face Classification on CCTV Images," IJCCS (Indonesian J. Comput. Cybern. Syst., vol. 14, no. 4, 2020, doi: 10.22146/ijccs.58092.
44. O. Bardhi and B. G. Zapirain, "Machine learning techniques applied to electronic healthcare records to predict cancer patient survivability," Comput. Mater. Contin., vol. 68, no. 2, 2021, doi: 10.32604/cmc.2021.015326.
45. M. Tuson, B. Turlach, K. Murray, M. R. Kok, A. Vickery, and D. Whyatt, "Predicting future geographic hotspots of potentially preventable hospitalisations using all subset model selection and repeated K-fold cross-validation," Int. J. Environ. Res. Public Health, vol. 18, no. 19, 2021, doi: 10.3390/ijerph181910253.
46. Q. S. Xu and Y. Z. Liang, "Monte Carlo cross validation," Chemom. Intell. Lab. Syst., vol. 56, no. 1, 2001, doi: 10.1016/S0169-7439(00)00122-2.
47. A. Almaw and K. Kadam, "Survey Paper on Crime Prediction using Ensemble Approach," Int. J. Pure Appl. Math., vol. 118, no. 8, 2018.

48. R. Lu and L. Li, "Application of an ensemble learning based classifier in crime prediction," 2020, doi: 10.18178/wcse.2019.06.019.
49. Y. Lamari, B. Freskura, A. Abdessamad, S. Eichberg, and S. de Bonviller, "Predicting spatial crime occurrences through an efficient ensemble-learning model," ISPRS Int. J. Geo-Information, vol. 9, no. 11, 2020, doi: 10.3390/ijgi9110645.
50. R. Atallah and A. Al-Mousa, "Heart Disease Detection Using Machine Learning Majority Voting Ensemble Method," 2019, doi: 10.1109/ICTCS.2019.8923053.
51. M. Malikhah, R. Sarno, and S. I. Sabilla, "Ensemble Learning for Optimizing Classification of Pork Adulteration in Beef Based on Electronic Nose Dataset," Int. J. Intell. Eng. Syst., vol. 14, no. 4, 2021, doi: 10.22266/ijies2021.0831.05.
52. A. Puurula, J. Read, and A. Bifet, "Kaggle LSHTC4 Winning Solution," May 2014, [Online]. Available: https://deepai.org/publication/k-nearestoracles-borderline-dynamic-classifierensemble-selection.
53. A. S. Britto, R. Sabourin, and L. E. S. Oliveira, "Dynamic selection of classifiers - A comprehensive review," Pattern Recognit., vol. 47, no. 11, pp. 3665–3680, 2014, doi: 10.1016/j.patcog.2014.05.003.
54. W. Ali, "Phishing Website Detection based on Supervised Machine Learning with Wrapper Features Selection," Int. J. Adv. Comput. Sci. Appl., vol. 8, no. 9, 2017, doi: 10.14569/ijacsa.2017.080910.
55. S. Xu et al., "Data cleaning in the process industries," Rev. Chem. Eng., vol. 31, no. 5, 2015, doi: 10.1515/revce-2015-0022.
56. N. Janardhan and N. Kumaresh, "Improving Depression Prediction Accuracy Using Fisher Score-Based Feature Selection and Dynamic Ensemble Selection Approach Based on Acoustic Features of Speech," Trait. du Signal, vol. 39, no. 1, pp. 87–107, Feb. 2022, doi: 10.18280/ts.390109.
57. D. V. R. Oliveira, G. D. C. Cavalcanti, T. N. Porpino, R. M. O. Cruz, and R. Sabourin, "K-Nearest Oracles Borderline Dynamic Classifier Ensemble Selection."
58. L. Yu, R. Zhou, R. Chen, and K. K. Lai, "Missing Data Preprocessing in Credit Classification: One-Hot Encoding or Imputation?," Emerg. Mark. Financ. Trade, vol. 58, no. 2, 2022, doi: 10.1080/1540496X.2020.1825935.
59. P. Kathiravan, R. Saranya, and S. Sekar, "Sentiment Analysis of COVID-19 Tweets Using Text Blob and Machine Learning Classifiers," pp. 89–106, 2023, doi: 10.1007/978-981-19-6634-7_8.