Unveiling the Future: Time Series Forecasting of Women's Crime Rates in India

Jasmine Das
Simran Minhas
Computer Science and Engineering Department
Thapar Institute of Engineering and Technology
Patiala, India

Dr Arun Pundir

Computer Science and Engineering Department
Thapar Institute of Engineering and Technology
Patiala, India

Abstract— **Predicting crime using machine learning and deep learning techniques has gained considerable attention from researchers in recent years, focusing on identifying patterns and trends in crime occurrences. This project examines over 4 datasets to explore the various machine learning and deep learning algorithms applied to predict crime rate. The study provides access to the datasets used for crime prediction by researchers and analyzes prominent approaches applied in machine learning and deep learning algorithms to predict crime, offering insights into different trends and factors related to criminal activities. Additionally, the paper highlights potential gaps and future directions that can enhance the accuracy of crime prediction. Finally, the comprehensive overview of research discussed in this paper on crime prediction using machine learning and deep learning approaches serves as a valuable reference for researchers in this field. By gaining a deeper understanding of crime prediction techniques, law enforcement agencies can develop strategies to prevent and respond to criminal activities more effectively.**

Keywords:crime prediction, crime datasets, deep learning, machine learning.

I. INTRODUCTION

Violence against women is a pervasive and concerning issue in India, posing a significant threat to public safety and social well-being. To effectively address this challenge, proactive measures are crucial. Here, time series forecasting emerges as a powerful tool with the potential to predict future trends in women's crime rates.This project delves into the application of time series forecasting models for predicting future occurrences of crimes against women in India. We explore the feasibility of leveraging historical crime data to identify patterns and seasonality, ultimately enabling forecasts that inform preventive strategies.To achieve the most accurate forecasts, we conduct a comparative study employing a diverse set of time series forecasting models. This encompasses both machine learning approaches like XGBoost and Random Forest, and deep learning models including LSTMs and Transformers. By evaluating a range of models, we aim to identify the one or combination that delivers the most effective forecasting performance for women's crime rates in the Indian context. By accurately predicting future crime trends, authorities can be empowered by allocating resources strategically to areas with anticipated spikes in crime rates. Foster safer communities for women by anticipating and mitigating potential threats.This project contributes significantly to the ongoing fight against women violence in India. By providing a forecast of future crime patterns, we aim to empower authorities and communities to take preventative actions, ultimately safeguarding the safety and well-being of women.

II. LITERATURE SURVEY

Addressing a pressing social issue, crime against women is a significant societal concern that poses serious threats to their safety, well-being, and empowerment. There is a critical need to develop effective methods and tools to combat such crimes and protect women from harm. Enhancing crime prevention strategies: Traditional crime prevention approaches often fall short in effectively addressing crimes against women due to their complex and dynamic nature. By leveraging machine learning algorithms, we can gain insights into patterns, trends, and risk factors associated with these crimes. This understanding can lead to the development of more targeted and proactive crime prevention strategies.

We provide a comprehensive review of existing research pertaining to each of the eight machine learning models employed in our study: GRU (Gated Recurrent Unit), LSTM (Long Short-Term Memory), RNN (Recurrent Neural Network), Gradient Boosting, XGBoost, Random Forest, DNN (Deep Neural Network), and Transformers.

A.Xgboost

XGBoost, or extreme Gradient Boosting, is a powerful and popular machine learning algorithm that falls under the umbrella of ensemble learning.This ensemble learning method that combines multiple weak decision trees into a powerful model.It's known for its efficiency, accuracy, and interpretability, making it popular for various tasks like classification, regression, and ranking, it is necessary to attach a fully connected layer. This layer takes the output information from CNN. Attaching a fully connected layer to the end of the network results in an N dimensional vector, where N is the number of classes from which the model selects the desired class.

B. Long short-term Memory (LSTM)

Long Short-Term Memory (LSTM) networks are a type of Recurrent Neural Network (RNN) capable of learning order dependence in sequence prediction problems. This is most used in complex problems like Machine Translation, Speech Recognition, and many more. The reason behind developing LSTM was, when we go deeper into a neural network if the gradients are very small or zero, then little to no training can take place, leading to poor predictive performance and this problem was encountered when training traditional RNNs. LSTM networks are well suited for classifying, processing, and making predictions based on time series data since there can be lags of unknown duration between important events in a time series. LSTM is way more effective and better compared to the traditional RNN as it overcomes the short term memory limitations of the RNN. LSTM can carry out relevant information throughout the processing of inputs and discards non-relevant information with a forget gate.

C. Transformers

Transformers are a revolutionary deep learning architecture that has achieved state-of-the-art results in various Natural Language Processing (NLP) tasks. Unlike Recurrent Neural Networks (RNNs) that process information sequentially, Transformers rely on an "attention mechanism" that allows them to focus on relevant parts of the input data simultaneously.This parallel processing makes them faster and more efficient than RNNs, especially for long sequences. Its components are Encoder- Processes the input sequence, creating a representation that captures the relationships between words.Decoder-Generates the output sequence, using the encoder's representation and its own attention mechanism.Attention Mechanism- The core of Transformers, it allows each part of the input sequence.

D. Deep Neural Network(DNN)

Deep Neural Networks (DNNs), also known as Deep Nets, are a powerful type of artificial neural network with a specific characteristic of multiple hidden layers. This distinguishes them from simpler neural networks that might have only one or two hidden layers.DNNs are stacked layers of interconnected artificial neurons, inspired by the structure of the human brain.Each layer performs a series of mathematical operations on the input it receives from the previous layer, transforming the data progressively.With multiple hidden layers, DNNs can learn complex relationships between input and output data, making them suitable for solving a wide range of tasks.

E. Recurrent neural networks(RNN)

RNNs are designed to handle sequential data like text, audio, or time series data. Unlike feedforward neural networks where information flows only once, RNNs have a loop in their architecture. This loop allows them to process information and maintain an internal state (memory) of what has been processed before.It has three layers that are Hidden Layer-This layer holds the internal state of the network, essentially its memory. It gets updated with information from the current input and the previous hidden state, allowing the network to remember relevant information across the sequence.Input Layer-Receives the individual elements of the sequence one at a time.Output Layer- Produces the final output based on the current input and the internal state.

F. Gated recurrent units(GRU)

GRUs are a variant of RNNs specifically designed to address the vanishing/exploding gradient problem. They introduce gates like update gate which decides how much of the previous hidden state to keep And reset gate which decides how much of the current input to let into the cell.

G. Random Forest

Random Forest is a powerful machine learning algorithm that belongs to the category of ensemble learning.Random Forest combines the predictions of multiple decision trees to make a single, more accurate prediction.Each decision tree in the forest is trained on a different subset of the data and uses a random subset of features at each split point.This randomness helps to reduce overfitting and improve the overall accuracy of the model. Steps followed as Bootstrapping-Random Forest creates multiple training sets by randomly sampling (with replacement) from the original data. This is called bootstrapping.Building Decision Trees-Each training set is used to build a separate decision tree.Random Feature Selection-At each split point in the tree, a random subset of features is considered for splitting, further increasing diversity among the trees.Prediction-To make a prediction, each tree in the forest votes on the class or value, and the final prediction is the majority vote (for classification) or the average (for regression).

H. Gradient Boost

Gradient boost is a fundamental optimization algorithm widely used in machine learning to train various models, including neural networks like RNNs, CNNs, and more. It aims to find the minimum of a function, which in machine learning often translates to minimizing the error between the model's predictions and the actual values.It works iteratively, meaning it takes small steps in the direction of the steepest descent (negative gradient) of the function at the current point. This gradually leads the model towards the minimum, improving its performance.Key components are Cost Function-This function measures the error between the model's predictions and the actual values. Gradient descent aims to minimize this cost function.Gradient-The gradient of the cost function indicates the direction of the steepest descent at any given point. It's calculated using calculus.

Learning Rate-This parameter controls the step size taken in each iteration. A smaller learning rate leads to smaller steps but may take longer to converge, while a larger learning rate can lead to faster convergence but might overshoot the minimum or even diverge.

III. TECHNOLOGY USED

A. Python

Python is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python's design philosophy emphasizes code readability with its notable use of significant whitespace. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects.Python is dynamically typed and garbage-collected. It supports multiple programming paradigms, including procedural, object oriented, and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library.

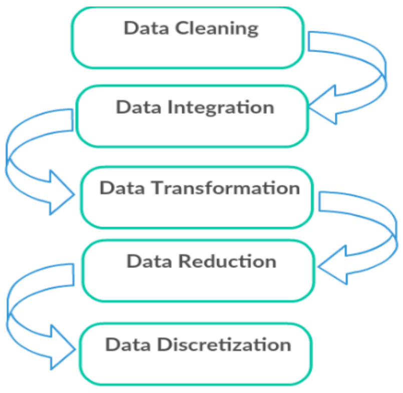
B. Google Colab

Colaboratory, or "Colab" for short, is a product from Google Research. Colab allows us to write and execute arbitrary python code through the browser, and is especially well suited to machine learning, data analysis and education. More technically, Colab is a hosted Jupyter notebook service that requires no setup to use, while providing free access to computing resources including GPUs

C. Python Libraries

1) Pandas
2) Numpy
3) Matplotlib
4) Keras
5) Nltk
6) TensorFlow
7) Transformers
8) Xgboost
9) Sklearn

IV. METHODOLOGY



A. Data Collection:

For data collection, we meticulously curated a diverse set of datasets from year 2001 to 2015 from data.gov.in (https://data.gov.in/catalog/district-wise-crimes-committed-against-women) specifically targeting common every year crime scenes to ensure relevance and inclusivity. Our dataset comprises 10677 labels meticulously paired with eleven descriptive features, resulting in a total of 10677 X 11. This dataset was chosen because of its size and variety of crime rate against women in India, which includes State or UT, District, Year,Rape, Kidnapping and Abduction, Dowry Deaths,Assault on Women with intent to outrage her Modesty, Insult to modesty of Women,Cruelty by Husband or his Relatives, and Importation of Girls.

This exhaustive collection spans a wide array of scenarios and contexts, providing ample training and evaluation data for our time series forecasting model. Such meticulous curation ensures the model's robustness and adaptability across diversing dataset, ultimately enhancing its performance and utility in real-world applications.

B. *Data Preprocessing:*

In our data preprocessing methodology, we meticulously clean and preprocess to analyze time series forecasting of crime rates against women in India effectively.

The primary source is likely data.gov.in where we combined three dataset from 2001 to 2012, 2013,2014 and 2015 is used for forecasting that consists of district wise crimes committed against women, which can be downloaded in various formats (CSV, Excel).

We begin by iterating through each row in the dataset, extracting its corresponding crime rate. For each crime in different states and districts in their respective year, we perform a series of preprocessing steps to ensure uniformity and coherence in the textual data.

Firstly, we convert all characters to uppercase to standardize the text and remove any potential case discrepancies. Next, we eliminate any non-alphabetic characters, including digits and special symbols, to focus solely on linguistic content. This step helps in enhancing the readability and interpretability of the captions. Furthermore, we remove any additional spaces to maintain consistency in text formatting. By condensing consecutive whitespace characters into single spaces, we ensure uniformity in the textual representation of captions. For missing values we identify and handle missing data points. This involves imputing missing values based on available information or removing entries with excessive missingness. If necessary, normalize numerical features to a common scale to improve the performance of machine learning algorithms used for further analysis.

To facilitate model training and enhance the quality of generated data, by adding new column year with date time value by using 'to_datetime'.This step aids the model in learning the structure and facilitating more accurate generation of descriptive narratives.Once the preprocessing steps are completed for all crimes in respective districts in the dataset, the cleaned data are stored back into the new cleaned csv file, ready for subsequent use in model training and evaluation.

Overall, our meticulous data preprocessing methodology ensures the consistency, coherence, and relevance of categorical data, laying a solid foundation for robust training and evaluating crime prediction on 2015 dataset.

C. Model Creation

The model architecture for time series forecasting is characterized by the seamless integration of eight algorithms on this data. This innovative approach ensures comprehensive representation learning from both visual graphs and textual modalities, facilitating accurate and contextually approximated crime rate values.

1.Random Forest

We initialize the Model by importing the Random Forest regressor from the sklearn-learn library and define the hyperparameters such as the number of trees (n_estimators), maximum depth of each tree (max_depth). Then we train the Model to fit the Random Forest model to the training data using the fit() method.

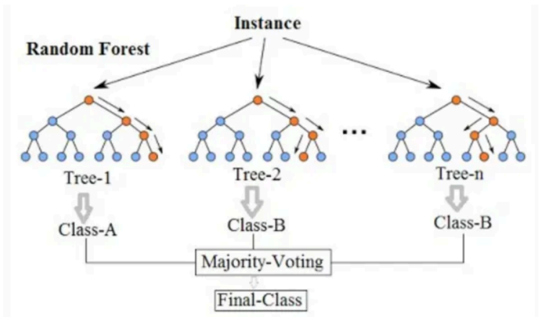


Fig-1: Architecture of Random Forest, Source: Adapted from [4]

2.Xgboost

First we import the XGBoost regressor from the XGBoost library. We define hyperparameters such as the learning rate (eta), maximum depth of trees (max_depth), number of trees (n_estimators). Then we fit the XGBoost model to the training data using the fit() method.

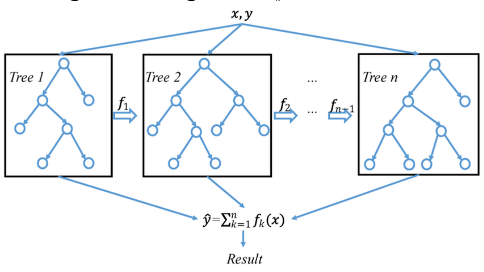


Fig-2: Architecture of XGBoost, Source: Adapted from [9]

3.Deep Neural Network

First we construct a sequential model with multiple fully connected (Dense) layers using TensorFlo. Then we specify the number of neurons in each layer, activation functions (sigmoid and ReLU), and dropout layers . We also specify the loss function like optimizer Adam. Fit the DNN model to the training data using the fit() method.

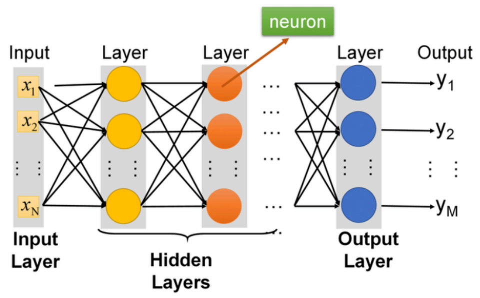


Fig-3: Architecture of DNN, Source: Adapted from [8]

4.Recurrent Neural Network

We constructed a sequential model using TensorFlow for Keras and added recurrent layers such as SimpleRNN, LSTM, or GRU. Next we specified the loss function as optimizer Adam and any additional metrics. We finally fit the RNN model to the training data using the fit() method.

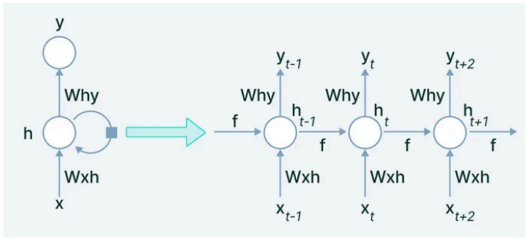


Fig-4: Architecture of RNN, Source: Adapted from [6]

5.Gated Recurrent Unit

We created an instance of the GRUModel class, specifying the input size, hidden size, number of layers, and output size. We then specified the loss function and optimizer to train the model. For classification tasks, commonly used loss functions include CrossEntropyLoss, and popular optimizers include Adam and SGD. We iterated through the dataset in batches. Performed forward pass by input data through the model to get predictions. Compared model predictions to actual labels using the defined loss function. Computed gradients of the loss with respect to model parameters. We use the optimizer to update model parameters based on computed gradients.

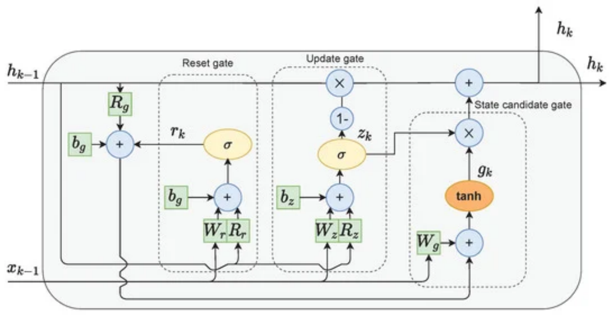


Fig-5: Architecture of GRU, Source: Adapted from [10]

6.LSTM

We constructed a sequential model using TensorFlow or Keras.Add LSTM layers with appropriate units (neurons), activation functions, and possibly dropout for regularization. Specify the loss function (e.g., Mean Squared Error),optimizer like Adam.Fit the LSTM model to the training data using the fit() method.

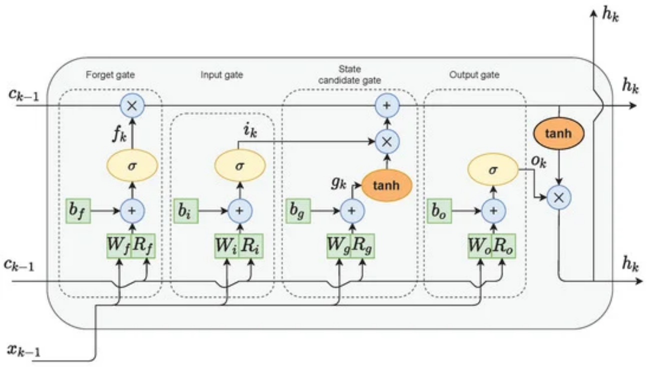


Fig-6: Architecture of LSTM, Source: Adapted from [10]

7.Transformer

The Transformer model consists of an encoder and a decoder, both of which are composed of multiple layers of self-attention mechanisms and feed-forward neural networks. In the encoder, each input sequence is processed independently in parallel through a stack of identical layers. Each layer contains two main sublayers: a multi-head self-attention mechanism and a position-wise feed-forward network. The self-attention mechanism allows each word in the input sequence to attend to all other words, capturing the relationships between different words in the sequence. The feed-forward network applies a non-linear transformation to each position separately and identically. To implement the encoder in PyTorch, we define the TransformerEncoderLayer class, which encapsulates the self-attention mechanism and feed-forward network. The decoder follows a similar structure to the encoder but also includes an additional attention mechanism called the encoder-decoder attention. This mechanism allows the decoder to focus on relevant parts of the input sequence when generating the output sequence.

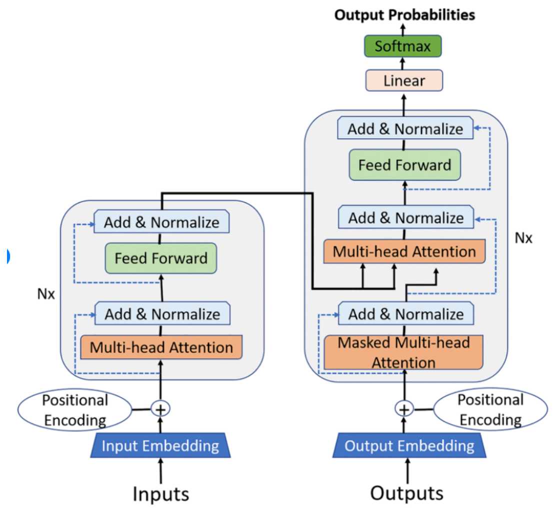


Fig-7: Architecture of Transformer, Source: Adapted from [11]

8.Gradient Boost

We start with a simple initial prediction, often the average of the target variable in your training data. This serves as the baseline for subsequent improvements. Then we calculate the difference between the actual target values and the initial prediction for each data point. These are called "pseudo-residuals". They represent the errors the model needs to learn from.Train a weak learner model, typically a shallow decision tree, on the calculated pseudo-residuals. This weak learner aims to capture the errors in the previous prediction. Combined the predictions of the weak learner with the initial prediction. This is done by multiplying the weak learner's predictions by a learning rate (usually between 0 and 1) to prevent overfitting. Repeated these steps iteratively. In each iteration, we calculate new pseudo-residuals based on the current ensemble prediction and train a new weak learner to improve the overall accuracy. We chose a loss function appropriate for this task, such as Mean Squared Error (MSE) for regression or cross-entropy for classification.

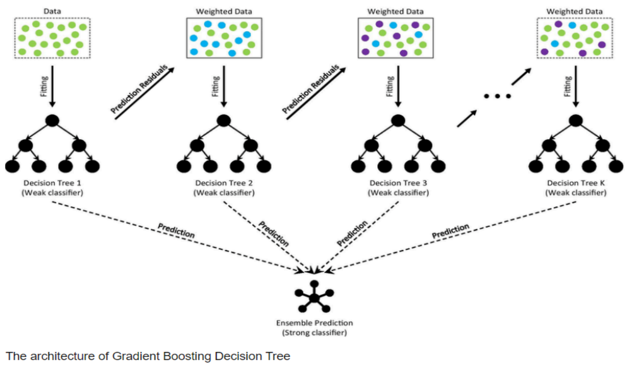
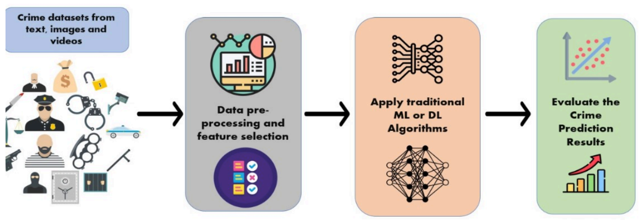

Data
Weighted Data
Weighted Data
Weighted Data
Fitting
Prediction Residuals
Prediction Residuals
Decision Tree 1 (Weak classifier)
Decision Tree 2 (Weak classifier)
Decision Tree 3 (Weak classifier)
Decision Tree K (Weak classifier)
Prediction
Ensemble Prediction (Strong classifier)
The architecture of Gradient Boosting Decision Tree

Fig-8: Architecture of Gradient Boost, Source: Adapted from [12]

Implementation Details:

The implementation of the model architecture is realized using the Sklearn for random forest, tensorflow for DNN, RNN, GRU and LSTM. Xgboost library for XGBoost and gradient descent .Then transformer and pipeline for transformer model and ensuring flexibility and efficiency in model development. During training, parameters of each RNN, DNN , LSTM, GRU, XGBoost, gradient descent and random forest components are jointly optimized, facilitated by the unified architecture. Model compilation utilizes categorical cross-entropy loss and the Adam optimizer, enabling effective parameter updates and convergence during training.

D. Model Training


Crime datasets from text, images and videos
Data pre-processing and feature selection
Apply traditional ML or DL Algorithms
Evaluate the Crime Prediction Results

This study investigates the effectiveness of eight diverse machine learning models for time series forecasting of crime rates against women in India. To ensure a comprehensive and comparative analysis, the following models were implemented and trained:

- *Recurrent Neural Networks (RNNs)*
- *Long Short-Term Memory (LSTM)*
- *Gated Recurrent Unit (GRU)*
- *Gradient Boosting*
- *XGBoost*
- *Random Forest*
- *Deep Neural Network (DNN)*
- *Transformer*

Ensuring high-quality, structured data is crucial for effective model training. Missing data points within the crime rate dataset were addressed using appropriate techniques like mean imputation or forward filling. This prevents the introduction of biases and ensures the model operates on complete information.

The crime rate data was analyzed for seasonality, such as yearly trends. If seasonality was present, appropriate techniques like seasonal differencing were applied to capture these recurring patterns effectively. This allows the model to learn the cyclical nature of crime rates and improve its forecasting accuracy.

Each model underwent training for a predefined number of epochs, with each epoch representing a complete iteration through the entire training dataset. During each epoch, the models were iteratively refined as they learned to capture the underlying patterns and trends within the crime rate data. This iterative process allowed the models to gradually improve their forecasting accuracy over successive epochs.

By implementing these comprehensive training procedures and hyperparameter tuning strategies, this study aims to provide a thorough comparison of the eight chosen Machine Learning models for time series forecasting of women's crime rates in India. This will ultimately lead to the identification of the model that demonstrates the most accurate forecasting ability on unseen data, including data from 2015 onwards.

V. RESULTS AND ANALYSIS

Evaluation of the eight diverse machine learning models for time series forecasting of women's crime rates in India was performed using the established evaluation metrics: Root Mean Squared Error (RMSE) and R-squared. This comparison provides insights into which model demonstrates the most accurate forecasting ability on the testing set, indicating its suitability for real-world predictions.

Table-1: Evaluation Metric Comparison

	XGBoost	GradientBoost	RNN	LSTM
RMSE	245.98	191.422	1789.74	1792.31
R-2 Score	0.981	0.97	0.012	0.009

Table-2: Evaluation Metric Comparison

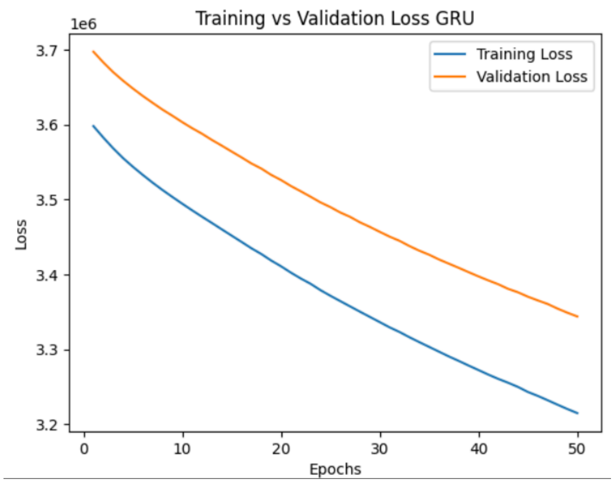
	Random Forest	GRU	Deep NN	Transformers
RMSE	311.94	1758.36	2012.98	220.16
R-2 Score	0.96	0.046	-0.0629	0.97

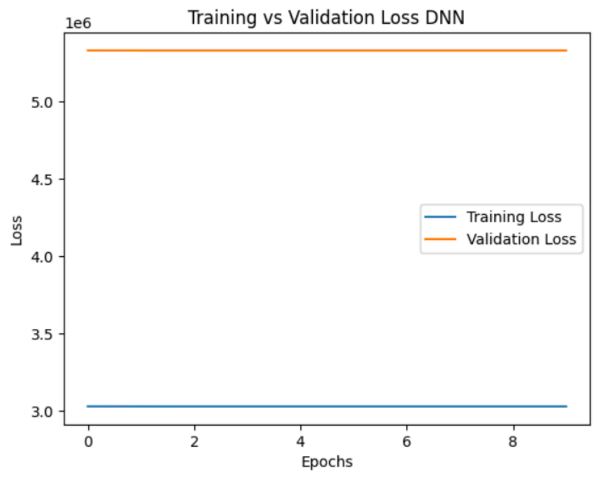
A lower RMSE and a R-2 score closer to 1 is generally regarded as indicative of good performance. Hence, we observe that XGBoost, Gradient Descent and transformers among the 8 models have the best performance. Whereas, DNN and LSTM have poor performance in this time series forecasting model.

Furthermore, the performance comparison between CPU computing resources has been done in the given *Table-3.*

Table-3: CPU Time Comparison

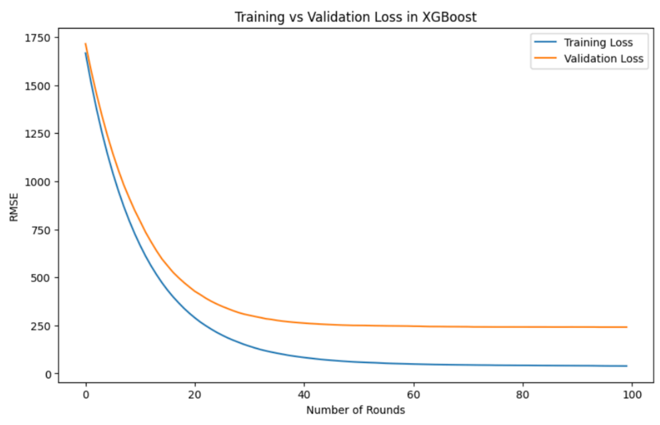
Machine Learning Model	CPU Time(s)
XGBoost	8.77
GradientBoost	13.60
RNN	143.94
LSTM	144.88
Random Forest	160.58
GRU	112.81
Deep NN	5.80
Transformers	8.85

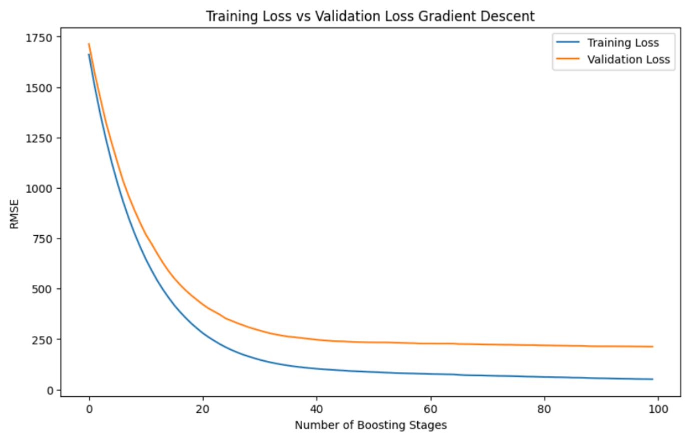


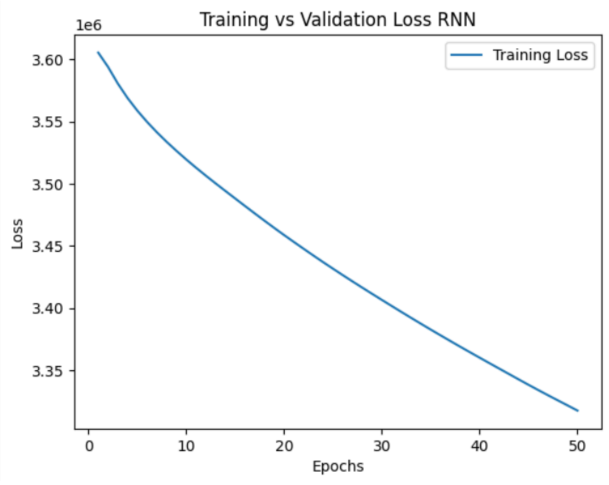


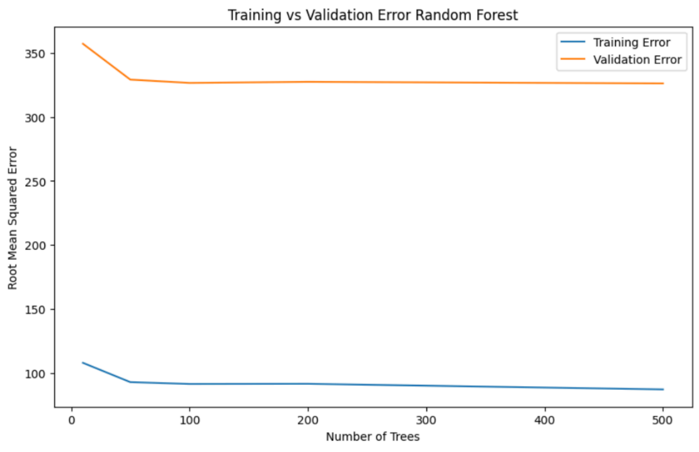
All figures below depict the comparison between the Training loss and Validation loss for each algorithm

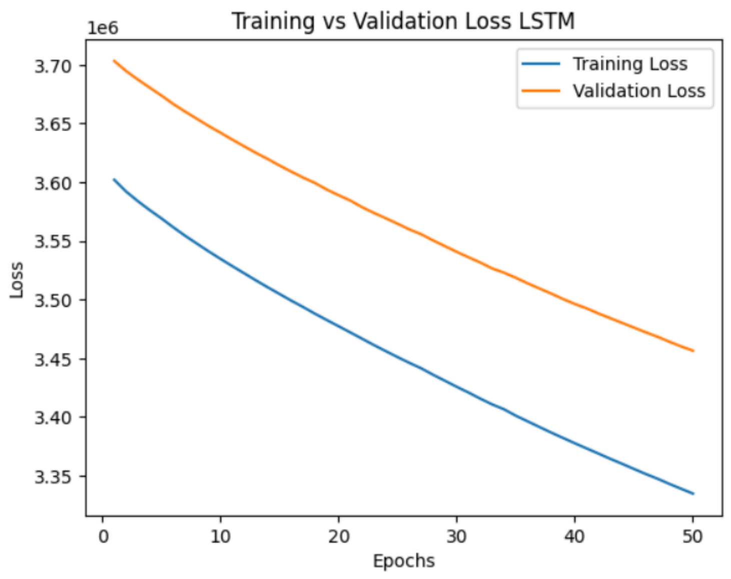
- For ***XGBOOST*** and ***GRADIENT BOOST,*** validation and training errors are close and seem to be converging which indicates that the model is likely generalizing well to unseen data and has learned the underlying patterns in the training data and can forecast on new data accurately.
- For ***RANDOMFOREST*** and ***DNN***, the training error is much lower than the validation error which indicates the model might be fitting too closely to the training data and not generalizing new unseen data and thus is not fit for forecasting on new data.
- For ***GRU*** and ***LSTM***, the training error is slightly lower than validation error but aren't converging which suggests that model might be slightly overfitting. It can capture some of the patterns in training data but is not generalizing well to unseen data due to high validation error and thus can be used for forecasting only if techniques like regularization(such as L1 or L2) are used to prevent overfitting.









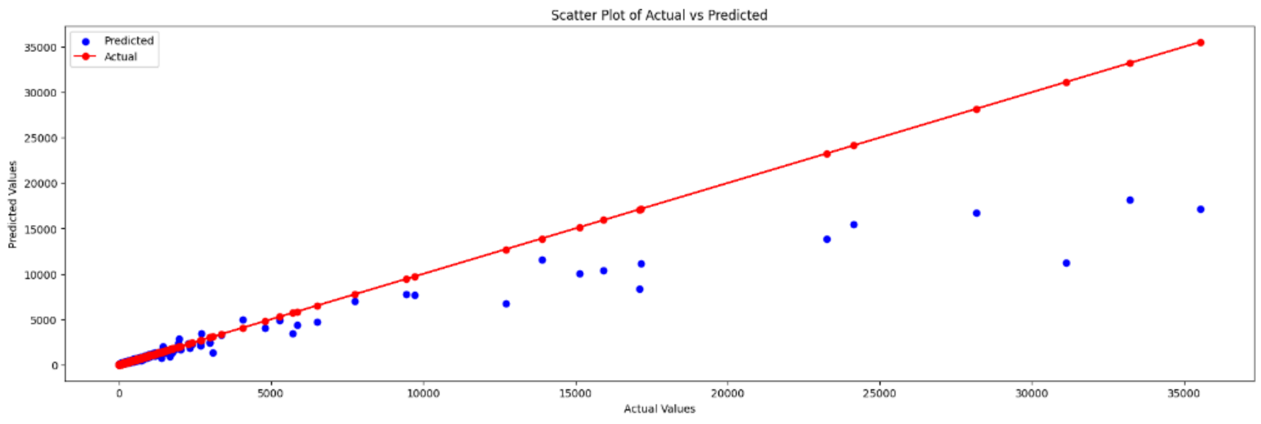


VI. FORECASTING ON 2015 DATA

Presenting forecasts generated by the best-performing model i.e XGBoost on new data from 2015

Results of forecasting the new data:

- The district having lowest predicted crime rate was **PAPUM PARE RURAL** with a predicted crime rate of : **7.94 crimes per 100,000 people**
- The district having highest predicted crime rate was **MUMBAI COMMR** with a predicted crime rate of : **4730.30 crimes per 100,000 people**



S.no	District	Actual Crime Rate	Predicted Crime Rate	Accuracy
1	Anantapur	952.0	631.747	66.35 %
2	Chittoor	366.0	347.24	94.8 %
3	Cuddapah	690.0	649.81	94 %
4	East Godavari	1438.0	1273.59	88.5n%
5	Guntakal Railway	1.0	-2.5	-25.0 %
6	Guntur	1325.0	1139.51	85.9 %
7	Guntur Urban	874.0	783.49	89.5 %
8	Krishna	1387.0	1340.71	96.6 %
9	Kurnool	1719.0	1225.79	71.2 %
10	Nellore	1063.0	992.71	93.3 %
11	Prakasham	906.0	774.31	85.4 %
12	Rajahmundry	435.0	446.77	97.5 %
13	Srikakulam	460.0	426.76	92.6 %
14	Tirupati Urban	228.0	202.83	88.5 %
15	Vijayawada City	641.0	550.49	85.8 %
16	Vijayawada Railway	22.0	4.27	19.4 %
17	Visakha Rural	343.0	306.81	89.2 %
18	Visakhapatnam	902.0	910.45	99.1 %
19	Vizianagaram	625.0	614.21	98.24 %
20	West Godavari	1554.0	1311.7	84.3 %

Table-4: Actual and Predicted crime rates in 2015 of first 20 districts

VII. REFERENCES

[1] Ruaa Mohammed Saeed, Husam Ali Abdulmohsin (2023) ‘A study on predicting crime rates through machine learning and data mining using text’. https://doi.org/10.1515/jisys-2022-0223

[2] Priyanka Das, Asit Kumar Das, "Crime analysis against women from online newspaper reports and an approach to apply it in dynamic environment", (ICBDAC) 2017 (IEEE Xplore: October 2017)

[3.] Bharathi, S., & Sengupta, S. (2021). A Machine Learning Approach to Predict Crime against Women in India. International Journal of Innovative Technology and Exploring Engineering, 10(8), 2077-2085.

[4] Will Koehrsen(2020), ‘Random Forest Simple Explanation by Medium’.

[5] R.N. Mangoli, Ganapati M. Tarase (2009),’Crime Against Women in India: A Statistical Review’. International Journal of Criminology and Sociological Theory, Vol. 2, No. 2, December 2009, 292-302. https://ijcst.journals.yorku.ca/index.php/ijcst/article/download/23401/21601/23695

[6] Mohsen Nabil(2023), ‘The Architecture of a Basic RNN’.

[7] Jha, A. K., & Kumar, R. (2020). Predicting crimes against women using machine learning techniques. International Journal of Advanced Science and Technology, 29(4), 1246-1255.

[8]Junxi Feng, Xiaohai He, Qizhi Teng, Chao Ren, Honggang Chen, Yang Li(2019), ‘Reconstruction of porous media from extremely limited information using conditional generative adversarial networks’.

[9]Y. Wa ng, Z. Pan, J. Zheng, L. Qian, M. Li1(2019), ‘A hybrid ensemble method for pulsar candidate Classification’

[10] Krzysztof Zarzycki and Maciej Ławryńczuk(2021), ‘ LSTM and GRU Neural Networks as Models of Dynamical Processes Used in Predictive Control’.

[11]Khalid Nassiri1, Moulay Akhloufi, ‘Transformer models used for text-based question answering Systems’.

[12]Haowen Deng, Youyou Zhou, Lin Wang, Cheng Zhang(2021), ‘Ensemble learning for early prediction’.