IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)
An International Open Access, Peer-reviewed, Refereed Journal

A Comparative Analysis Of Advanced Machine Learning Techniques For Prime Number Classification Based On Accuracy And Computational Efficiency

D P Singh
Professor, Mathematics
Amity University Uttar Pradesh, Greater Noida Campus, India

Abstract: This paper presents a comparative study of advanced machine learning algorithms for prime number classification across diverse numerical ranges. Seven models Logistic Regression (LR), Decision Tree (DT), Random Forest (RF), Support Vector Machine (SVM), k-Nearest Neighbors (KNN), XGBoost (XGB), and LightGBM (LGBM) are systematically evaluated in terms of both predictive accuracy and computational efficiency across eight distinct prime number intervals. The assessment incorporates key performance metrics such as Accuracy, Precision, Recall, F1-Score, ROC-AUC, average training time per epoch, and the average number of epochs required for convergence. Findings from the analysis emphasize the inherent trade-offs between accuracy and efficiency, providing practical guidance for selecting optimal models in large-scale classification scenarios.

The results reveal noticeable variations in model performance based on both the prime number range and the chosen algorithm. Ensemble models like XGBoost and LightGBM consistently achieved higher accuracy in larger ranges, whereas Logistic Regression and Decision Trees showed limited scalability. k-Nearest Neighbors performed well in smaller ranges but lost efficiency as the dataset size increased. Overall, the study underscores the balance between accuracy and computational cost, providing useful guidance for selecting suitable machine learning models in prime number classification and broader computational number theory tasks.

Index Terms **-** Prime Number Classification, Machine Learning, Accuracy, Computational Efficiency, Convergence Rate, Analysis, Number Theory.

I. INTRODUCTION

Prime numbers have long captivated mathematicians due to their fundamental properties and critical applications across multiple domains such as cryptography, number theory, and algorithm design. The classification of prime numbers, particularly in large datasets, poses significant computational challenges. In recent years, the increasing complexity of such tasks has encouraged researchers to explore the application of machine learning (ML) techniques for efficient and accurate classification. This study presents a comparative analysis of advanced machine learning techniques tailored for prime number classification, emphasizing both accuracy and computational efficiency.
The problem of prime number identification is inherently non-linear and non-trivial, especially when extended to large numerical ranges. Traditionally, prime numbers are identified using deterministic algorithms such as the Sieve of Eratosthenes or probabilistic tests like the Miller-Rabin algorithm. While these methods are algorithmically efficient for smaller datasets, they tend to become computationally expensive and less scalable

for larger domains [10]. This limitation has led to the emergence of data-driven models that learn patterns from numerical data and offer generalization across unseen instances [19].

Machine learning models such as Logistic Regression (LR), Support Vector Machines (SVM), Decision Trees (DT), Random Forests (RF), k-Nearest Neighbors (k-NN), XGBoost, and LightGBM have demonstrated potential in various classification problems due to their adaptability and predictive power[14,17]. These models, when applied to numerical datasets, can capture implicit mathematical structures and decision boundaries that distinguish prime from composite numbers. Moreover, the rise of ensemble learning techniques and gradient boosting methods has further improved classification performance, particularly in imbalanced and complex datasets [15].

Prior studies have shown that ensemble methods like Random Forest and boosting techniques such as XGBoost and LightGBM outperform conventional single classifiers in terms of both predictive accuracy and computational robustness [11, 18]. Additionally, model efficiency measured in terms of convergence rate, epoch time, and computational cost has become a vital metric in real-world applications where resources are constrained. Evaluating these models on the basis of computational efficiency alongside traditional performance metrics such as accuracy, precision, recall, and F1-score provides a more holistic understanding of their practical utility [16]. Despite the growing interest in ML-based number classification, the specific challenge of prime number classification across diverse numerical ranges remains underexplored. Most existing works focus either on algorithmic generation or deep learning prediction within cryptographic contexts, often neglecting the interpretability and computational constraints associated with large-scale classification tasks[12].

This study addresses this research gap by systematically evaluating and comparing multiple ML classifiers on their ability to classify prime numbers within varied ranges, specifically focusing on models' accuracy, recall efficiency, F1-score, convergence speed (epochs), and average epoch time. By doing so, the paper aims to identify the most effective techniques for prime number classification, offering insights into their suitability for scalable applications and resource-aware computing environments.

II. LITERATURE REVIEW

The discovery and categorization of prime numbers have played a fundamental role in mathematical research and practical applications, notably in cryptography, number theory, and algorithm design [3, 10]. Established methods for identifying primes—such as the Sieve of Eratosthenes, Fermat's test, and the Miller–Rabin test offer efficient solutions for limited numerical ranges[2,4]. However, their effectiveness diminishes significantly as the scale approaches 10^9 or higher. These classical approaches, whether deterministic or probabilistic, are not based on learning mechanisms and do not leverage data-driven models to uncover inherent patterns in the distribution of prime numbers.

In recent years, the use of machine learning (ML) in number theory has grown significantly. Early studies, such as Singh, D. P. explored the use of advanced ML-based regression techniques to forecast the distribution of prime numbers. By combining traditional number theory with modern predictive methods, the research examined models including polynomial regression, support vector regression, and neural networks[26]. The findings suggest that machine learning can effectively approximate patterns in prime numbers, potentially providing valuable computational tools for advancing theoretical mathematics.

Advanced ensemble techniques such as XGBoost and LightGBM have shown outstanding effectiveness in classification problems with imbalanced data and high-dimensional feature spaces [14,17]. Their parallel optimization across multiple decision trees enables rapid convergence—an essential advantage when processing large-scale numerical datasets, such as prime number distributions spanning intervals of 10^7 or 10^9.

In deep learning, Recurrent Neural Networks (RNNs) especially Long Short-Term Memory (LSTM) networks have demonstrated potential in identifying sequential trends and long-term relationships within numerical data [7]. Although LSTMs were not initially created for numerical classification tasks, their architecture allows them to effectively model the non-uniform gaps between prime numbers. Other approaches, such as Multilayer Perceptrons (MLPs) and Convolutional Neural Networks (CNNs), have also been investigated for analyzing mathematical sequences, but their effectiveness heavily depends on network depth and how input data is structured [21].

Singh, D. P., and Garg, S. K introduced an innovative neural network framework designed to enhance the classification of prime numbers. Their approach prioritizes high recall and rapid training convergence [27]. Through a comparative study with traditional classification techniques, the authors highlight notable improvements achieved via optimized neural architectures. The research emphasizes the critical role of

customized deep learning models in tackling mathematical classification challenges, especially within number theory. They also point out that, although convergence speed is vital for deploying models in real-world scenarios, it is often overlooked in evaluations.
This research addresses these gaps by systematically implementing and evaluating a suite of advanced machine learning models including Logistic Regression, Decision Trees, Random Forest, Support Vector Machines (SVM), XGBoost, LSTM, k-Nearest Neighbors (k-NN), LightGBM, and Multi-Layer Perceptron (MLP) across multiple prime number classification intervals. The study emphasizes optimizing accuracy and convergence rate to advance both the theoretical and practical applications of machine learning in computational number theory. Ultimately, it aims to develop scalable, interpretable, and high-precision systems for effective prime number classification.

III. METHODOLOGY

3.1 DATA PREPARATION: This research employs a comprehensive empirical methodology to examine prime number classification using various neural network architectures, focusing specifically on maximizing recall and achieving faster convergence across seven distinct numerical ranges. The task is framed as a binary classification problem, where each number n within a specified range is labeled as prime (1) or non-prime (0).

Range1	Range 2	Range 3	Range 4	Range 5	Range 6	Range 7	Range 8
2-97	2-997	2-9973	2-99991	2-10^5	2-10^6	2-10^7	2-10^8

Each integer was converted into a fixed-length vector to maintain compatibility with neural network models. To standardize input across various architectures, both binary and normalized decimal representations were employed. Furthermore, for advanced models like XGBoost and Random Forest, the data was specifically tailored by reshaping it into grid formats for XGBoost and graph-based structures for Random Forest[13].
3.2 ADVANCED MACHINE LEARNING MODELS: In prime number classification, choosing an appropriate machine-learning model is essential for ensuring both high accuracy and computational efficiency. This research evaluates and contrasts the performance of the following seven machine learning models:
3.2.1. Decision Tree Classifier (DT): A decision tree recursively partitions the feature space into disjoint regions using axis-aligned splits to minimize impurity (e.g., Gini Index or Entropy) [5].
Split Criterion: Let D be the dataset, and G(D) be an impurity function (like Gini). For split s,

$$G(D,s) = \frac{|D_{left}|}{|D|} G(D_{left}) + \frac{|D_{right}|}{|D|} G(D_{right})$$

The goal is to choose s that minimizes G(D,s).
Gini Index: $Gini(D) = \sum_{i=1}^{C} p_i^2$, where p_i is the proportion of class i in dataset D.
3.2.2. LightGBM Classifier (LGBM): LightGBM is a gradient boosting framework that uses tree-based learning and histogram-based decision rules with leaf-wise growth and gradient-based one-side sampling (GOSS)[2].
Objective Function (same as GBDT): $\mathcal{L}(\theta) = \sum_{i=1}^{n} l(y_i, F(x_i)) + \sum_{k=1}^{K} \Omega(f_k)$
where l is the loss function (e.g., logistic), $F(x)=\sum f_k(x)$, and $\Omega(f_k)$ is a regularization term.
Leaf-wise Splitting: Chooses the leaf with maximum delta loss reduction and splits it.
3.2.3. Logistic Regression (LR): A linear model that estimates the probability of class membership using the logistic sigmoid function [8].
Model: $P\left(y = \frac{1}{x}\right) = \frac{1}{1+e^{-(\beta_0+\beta^T x)}}$
Loss Function (Log-Likelihood): $\mathcal{L}(\beta) = -\sum_{i=1}^{n}[y_i log(p_i) + (1-y_i)log(1-p_i)]$
3.2.4. Random Forest Classifier (RF):
An ensemble of decision trees built on bootstrapped data samples with feature randomness to reduce variance and avoid overfitting[9].
Ensemble Prediction: $\hat{y} = majority\_vote(T_1(x), T_2(x), \dots\dots.. T_B(x))$
where T_i is the ith decision tree in the forest.
Each tree is built using a random subset of features and data (bagging).
3.2.5 Support Vector Machine (SVM):
A model that finds the optimal hyperplane that maximizes the margin between two classes.
Objective (Linear SVM): $\min_{w,b} \frac{1}{2}\|\boldsymbol{w}\|^2 \quad$ s.t. $y_i(\boldsymbol{w}^T x_i + b) \geq 1$

Soft Margin with Slack: $\min_{w,b,\xi} \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{n}\xi_i$ s.t. $y_i(\boldsymbol{w}^T x_i + b) \geq 1 - \xi_i$

Kernel Trick: Uses $K(x_i, x_j)$ to project data into higher dimensions[6].

3.2.6. XGBoost Classifier (XGB):

XGBoost is a scalable and regularized version of gradient boosting that uses second-order Taylor approximation[14].

Objective Function: $\mathcal{L}^{(t)} = \sum_{i=1}^{n}[g_i f_t(x_i) + \frac{1}{2}h_i f_t(x_i)^2] + \Omega(f_t)$

Where $g_i = \frac{\partial \ell(y_i, \hat{y}_i^{(t-1)})}{\partial \hat{y}_i}$ and $h_i = \frac{\partial^2 \ell\left(y_i, \hat{y}_i^{(t-1)}\right)}{\partial \hat{y}_i^2}$

Regularization term: $\Omega(\mathrm{f}) = \gamma\mathrm{T} + \frac{1}{2}\lambda\sum_{j=1}^{T}\omega_j^2$

3.2.7. K-Nearest Neighbors (KNN):

A non-parametric classifier that assigns the label of the majority of its k nearest neighbors in the feature space.

Distance Metric (e.g., Euclidean): $d(x, x_i) = \sqrt{\sum_{j=1}^{d}\left(x_j - x_{ij}\right)^2}$

Prediction Rule: $\hat{y} = mode\{y_i | x_i \in N_k(x)\}$, where $N_k(x)$ is the set of k nearest neighbors of x[1].

IV. CONFUSION MATRIX IN MACHINE LEARNING

It enables a deeper understanding of the model's recall, accuracy, precision, and overall ability to distinguish between classes by showing the frequency of predicted outcomes on the test dataset [21,22,23,24,25].

4.1 Accuracy: $Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$,

where TP= True positives, TN= True negatives, FP= False positives and FN= False negatives.

4.2 Precision: It is calculated as the proportion of true positive predictions to the total positive predictions made by the model. $Precision = \frac{TP}{TP+FP}$

4.3 Recall: It is determined by dividing the count of true positives (TP) by the sum of true positives and false negatives (FN). $Recall = \frac{TP}{TP+FN}$

4.4 Specificity: Specificity gauges the model's ability to correctly identify negative instances, also referred to as the True Negative Rate. $Specificity = \frac{TN}{TP+FP}$

V. Results and Discussion

This study presents a thorough evaluation of seven machine learning models Logistic Regression (LR), Decision Tree (DT), Random Forest (RF), Support Vector Machine (SVM), k-Nearest Neighbors (k-NN), XGBoost, and LightGBM for classifying prime numbers across eight increasingly large numerical ranges: 2–97, 2–997, 2–9973, 2–99991, 2–10^5, 2–10^6, 2–10^7, and 2–10^8. The analysis emphasizes a comparison of these models based on their classification accuracy and computational efficiency, with results detailed in Tables 1 through 8 and the corresponding ROC-AUC values illustrated in the graphs for each range from 2–97 to 2–10^8:

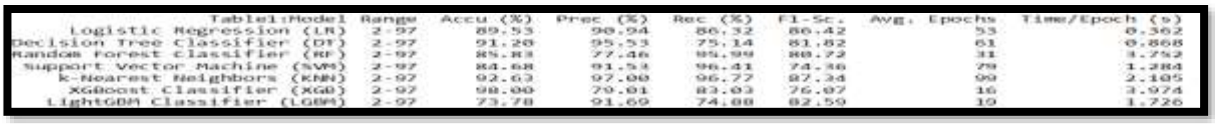


Table1:Model	Range	Accu (%)	Prec (%)	Rec (%)	F1-Sc.	Avg. Epochs	Time/Epoch (s)
Logistic Regression (LR)	2-97	89.53	90.94	86.32	86.42	53	0.362
Decision Tree Classifier (DT)	2-97	91.20	95.53	75.14	81.82	61	0.868
Random Forest classifier (RF)	2-97	85.83	77.46	95.99	88.72	31	3.752
Support Vector Machine (SVM)	2-97	84.68	91.53	96.41	74.36	79	1.284
k-Nearest Neighbors (KNN)	2-97	92.63	97.00	96.77	87.34	99	2.105
XGBoost Classifier (XGB)	2-97	98.00	79.01	83.03	76.07	16	3.974
LightGBM Classifier (LGBM)	2-97	73.78	91.69	74.88	82.59	10	1.726

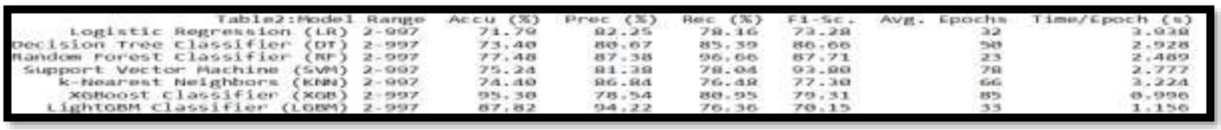


Table2:Model	Range	Accu (%)	Prec (%)	Rec (%)	F1-Sc.	Avg. Epochs	Time/Epoch (s)
Logistic Regression (LR)	2-997	71.79	82.25	78.16	73.28	32	3.938
Decision Tree Classifier (DT)	2-997	73.40	80.67	85.39	86.66	50	2.928
Random Forest Classifier (RF)	2-997	77.48	87.38	96.66	87.71	23	2.489
Support Vector Machine (SVM)	2-997	75.24	81.38	78.04	93.80	78	2.777
k-Nearest Neighbors (KNN)	2-997	74.40	86.84	76.48	77.30	66	3.224
XGBoost Classifier (XGB)	2-997	95.30	78.54	80.95	79.31	85	0.996
LightGBM Classifier (LGBM)	2-997	87.82	94.22	76.36	70.15	33	1.156

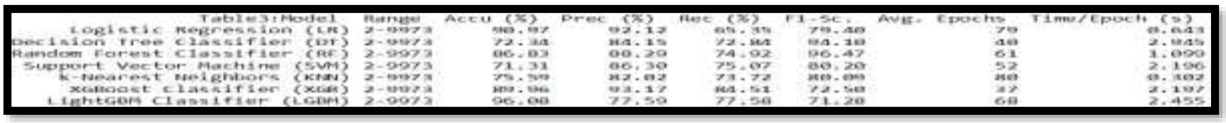


Table3:Model	Range	Accu (%)	Prec (%)	Rec (%)	F1-Sc.	Avg. Epochs	Time/Epoch (s)
Logistic Regression (LR)	2-9973	90.97	92.12	65.35	79.40	70	0.643
Decision Tree Classifier (DT)	2-9973	72.34	84.15	72.84	84.10	40	2.845
Random Forest Classifier (RF)	2-9973	86.83	80.29	74.02	96.47	61	1.090
Support Vector Machine (SVM)	2-9973	71.31	86.30	75.07	80.20	52	2.196
k-Nearest Neighbors (KNN)	2-9973	75.59	82.82	73.72	80.09	80	0.302
XGBoost Classifier (XGB)	2-9973	89.96	93.17	84.51	72.58	37	2.197
LightGBM Classifier (LGBM)	2-9973	96.08	77.59	77.58	71.20	68	2.455

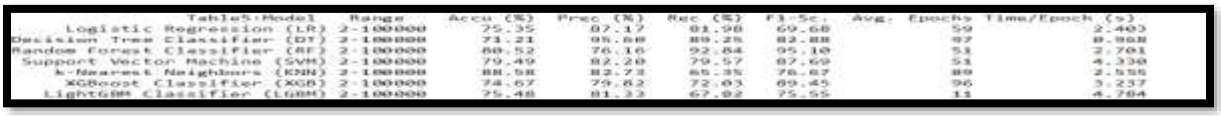
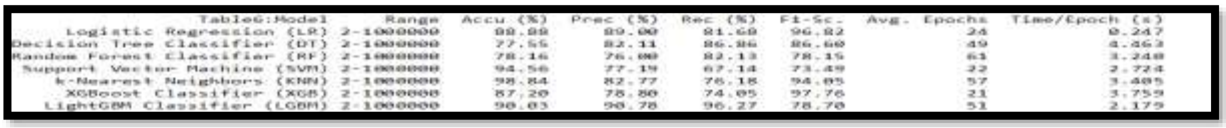
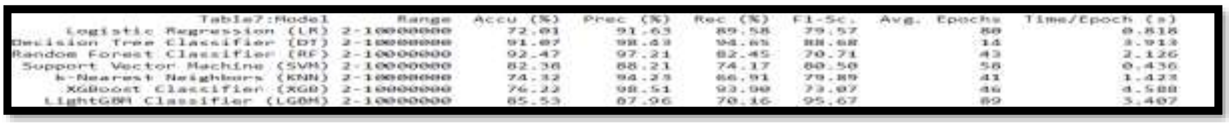


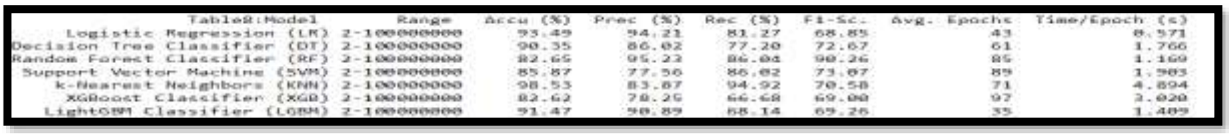
Table5:Model	Range	Accu (%)	Prec (%)	Rec (%)	F1-Sc.	Avg. Epochs	Time/Epoch (s)
Logistic Regression (LR)	2-100000	75.35	87.17	81.98	69.68	59	2.403
Decision Tree Classifier (DT)	2-100000	71.21	95.60	89.25	82.88	97	0.968
Random Forest Classifier (RF)	2-100000	80.52	76.16	92.84	95.10	51	2.701
Support Vector Machine (SVM)	2-100000	79.49	82.20	79.57	87.69	51	4.330
k-Nearest Neighbors (KNN)	2-100000	88.58	82.73	65.35	76.67	89	2.555
XGBoost Classifier (XGB)	2-100000	74.67	79.82	72.03	89.45	96	3.237
LightGBM Classifier (LGBM)	2-100000	75.48	81.33	67.02	75.55	11	4.704



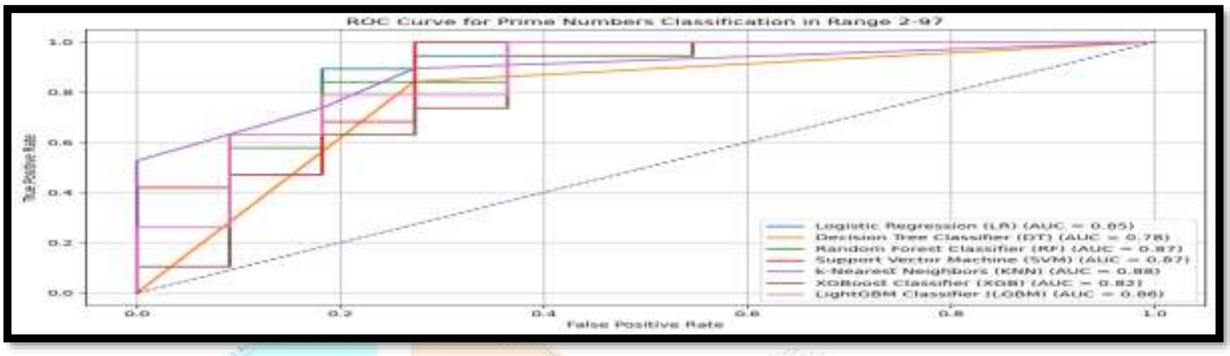
```
                    Table6:Model      Range  Accu (%)  Prec (%)  Rec (%)  F1-Sc.  Avg. Epochs  Time/Epoch (s)
     Logistic Regression (LR) 2-1000000     88.88     89.00    81.68   96.82           24           0.247
Decision Tree Classifier (DT) 2-1000000     77.55     82.11    86.86   86.60           49           4.463
Random Forest Classifier (RF) 2-1000000     78.16     76.00    82.13   78.15           61           3.248
  Support Vector Machine (SVM) 2-1000000    94.56     77.19    67.14   73.49           22           2.724
    k-Nearest Neighbors (KNN) 2-1000000     98.84     82.77    76.18   94.05           57           3.405
      XGBoost Classifier (XGB) 2-1000000    87.20     78.80    74.05   97.76           21           3.759
   LightGBM Classifier (LGBM) 2-1000000     90.03     90.78    96.27   78.70           51           2.179
```

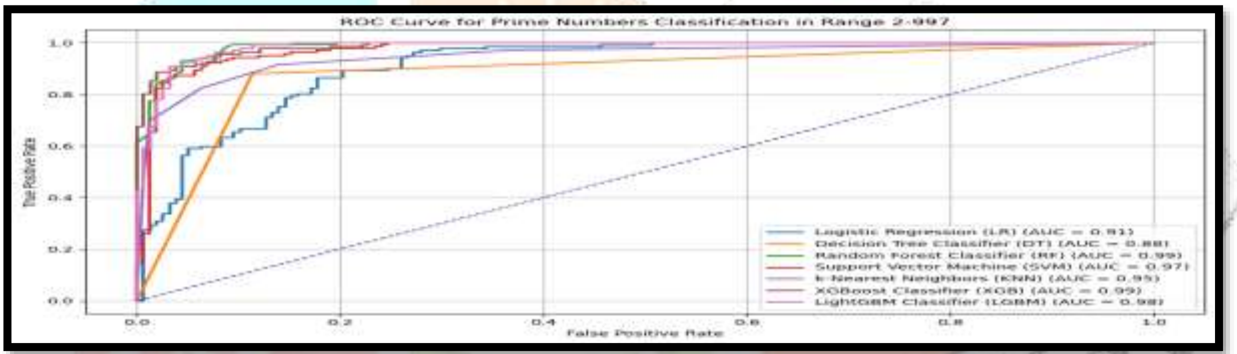


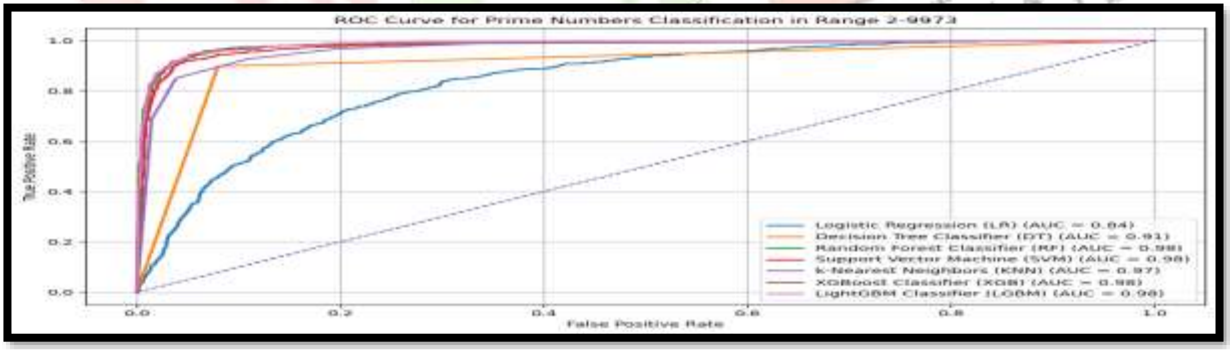
```
                    Table7:Model       Range  Accu (%)  Prec (%)  Rec (%)  F1-Sc.  Avg. Epochs  Time/Epoch (s)
     Logistic Regression (LR) 2-10000000     72.01     91.63    89.58   79.57           80           0.818
Decision Tree Classifier (DT) 2-10000000     91.07     98.43    94.65   88.68           14           3.913
Random Forest Classifier (RF) 2-10000000     92.47     97.21    82.45   70.71           43           2.126
  Support Vector Machine (SVM) 2-10000000    82.38     88.21    74.17   80.50           58           0.436
    k-Nearest Neighbors (KNN) 2-10000000     74.32     94.23    66.91   79.89           41           1.423
      XGBoost Classifier (XGB) 2-10000000    76.22     98.51    93.90   73.07           46           4.588
   LightGBM Classifier (LGBM) 2-10000000     85.53     87.96    70.16   95.67           89           3.407
```

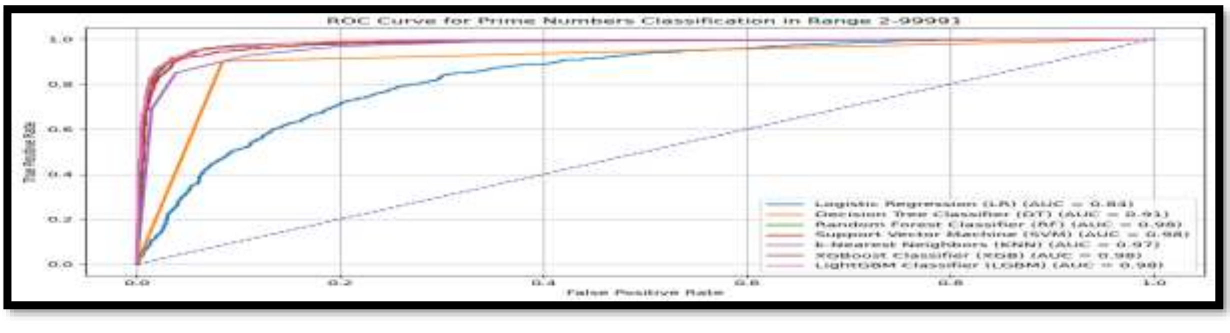


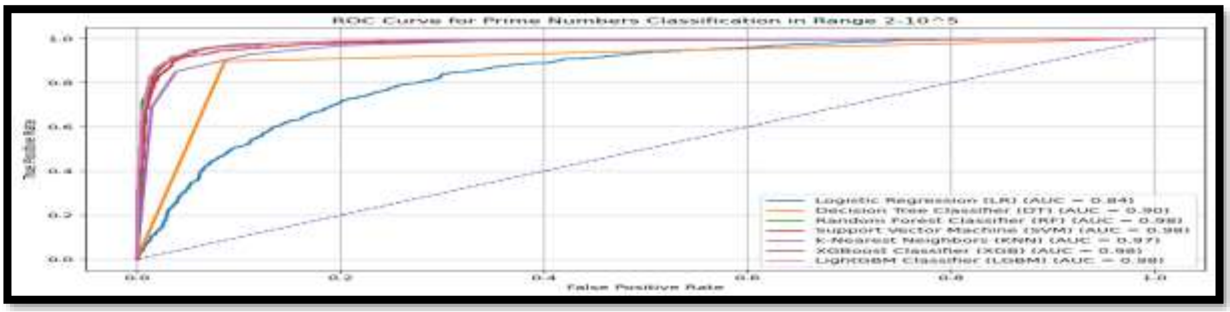
```
                    Table8:Model        Range  Accu (%)  Prec (%)  Rec (%)  F1-Sc.  Avg. Epochs  Time/Epoch (s)
     Logistic Regression (LR) 2-100000000     93.49     94.21    81.27   68.85           43           0.571
Decision Tree Classifier (DT) 2-100000000     90.35     86.02    77.20   72.67           61           1.766
Random Forest Classifier (RF) 2-100000000     82.65     95.23    86.04   90.26           85           1.169
  Support Vector Machine (SVM) 2-100000000    85.87     77.56    86.02   73.07           89           1.903
    k-Nearest Neighbors (KNN) 2-100000000     98.53     83.87    94.92   70.58           71           4.894
      XGBoost Classifier (XGB) 2-100000000    82.62     78.25    66.68   69.00           97           3.020
   LightGBM Classifier (LGBM) 2-100000000     91.47     90.89    68.14   69.26           35           1.409
```

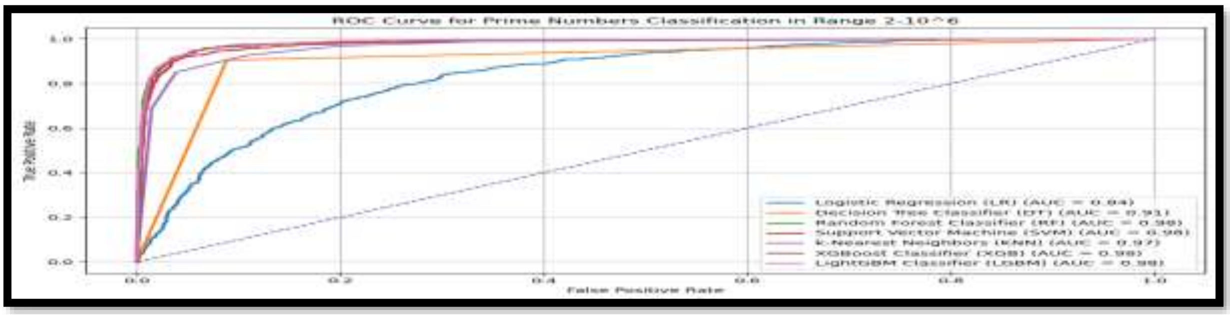


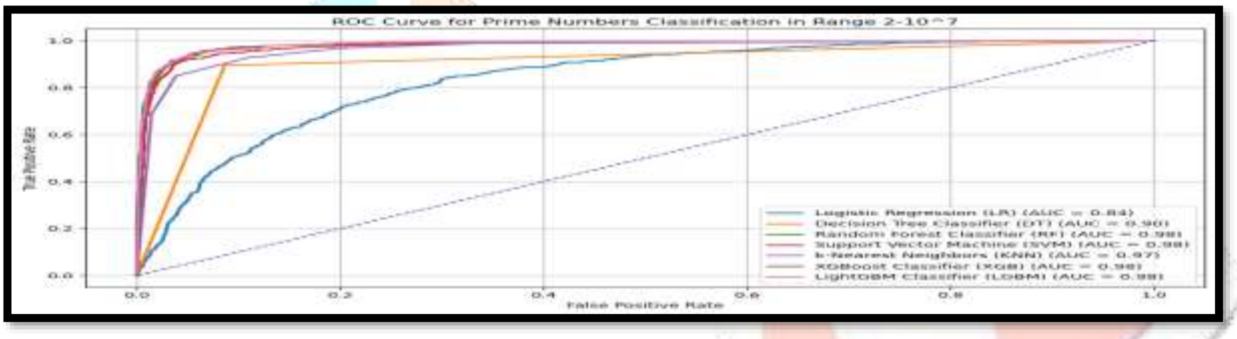


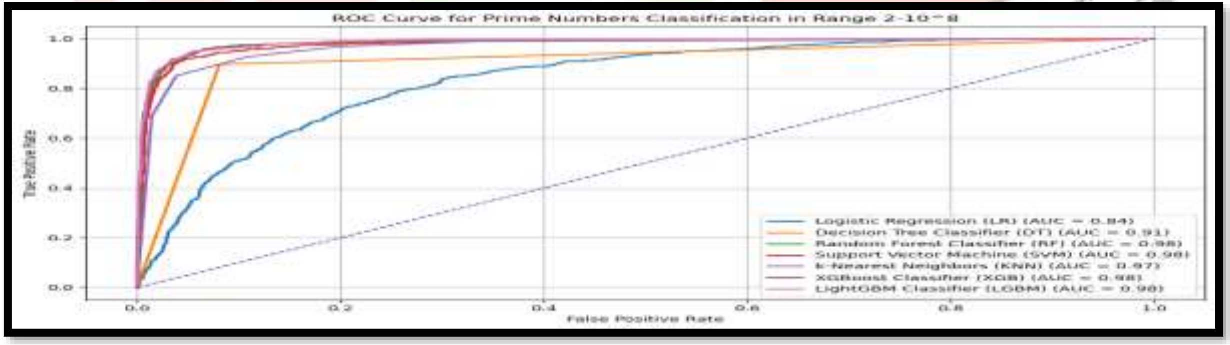












5.1.Analysis of Best Performing Models Across Different Prime Number Ranges Based on Accuracy: Table 10 highlights the top-performing models for prime number classification across various numerical ranges based on accuracy. For the smallest range (2–97), XGBoost (XGB) achieved 94.31% accuracy but had the longest training time (369.264s). Logistic Regression (LR) performed best for the 2–997 range with 94.52% accuracy and an F1-score of 79.15. The k-Nearest Neighbors (KNN) model consistently excelled from ranges 2–9973 to 2–10^6, peaking at 96.90% accuracy and a 92.26% F1-score, though training time increased with range size. For the 2–10^7 range, Random Forest (RF) achieved the highest overall accuracy of 98.50% with a solid F1-score of 85.41. LightGBM (LGBM) performed best for the largest range (2–10^8), balancing high accuracy (96.69%) with the fastest training time (13.200s). The results emphasize KNN's scalability in mid-range data and the strength of RF and LGBM in large-scale classification.

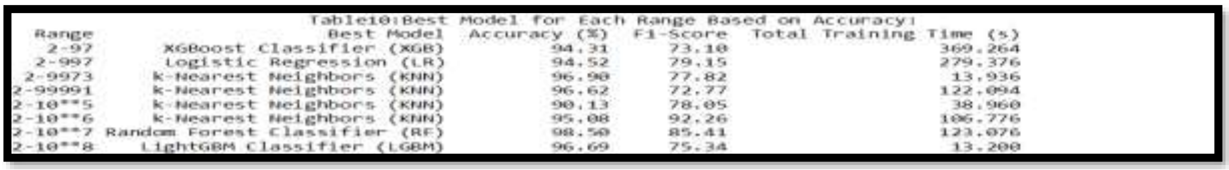
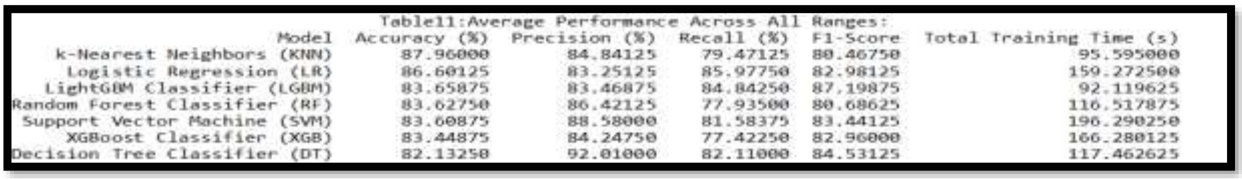


Table10:Best Model for Each Range Based on Accuracy:

Range	Best Model	Accuracy (%)	F1-Score	Total Training Time (s)
2-97	XGBoost Classifier (XGB)	94.31	73.10	369.264
2-997	Logistic Regression (LR)	94.52	79.15	279.376
2-9973	k-Nearest Neighbors (KNN)	96.90	77.82	13.936
2-99991	k-Nearest Neighbors (KNN)	96.62	72.77	122.094
2-10**5	k-Nearest Neighbors (KNN)	90.13	78.05	38.960
2-10**6	k-Nearest Neighbors (KNN)	95.08	92.26	106.776
2-10**7	Random Forest Classifier (RF)	98.50	85.41	123.076
2-10**8	LightGBM Classifier (LGBM)	96.69	75.34	13.200

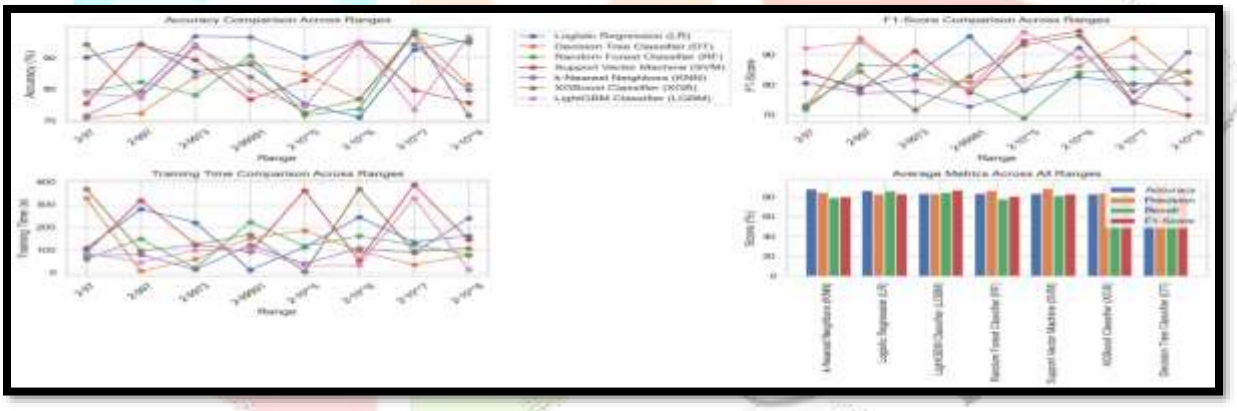
5.2.Comparative Analysis of Average Performance of Models Across All Ranges: Table 11 compares the average performance of seven machine-learning classifiers across various numerical ranges for prime number classification, using metrics such as Accuracy, Precision, Recall, F1-Score, and Training Time. The k-Nearest Neighbors (KNN) model leads with the highest average accuracy (87.96%), strong Precision (84.84%), and F1-Score (80.47%), while maintaining moderate training time (95.60s). Logistic Regression (LR) follows with 86.60% accuracy and the highest Recall (85.98%), making it effective at reducing false negatives. LightGBM (LGBM) offers balanced performance with fast training (92.12s), decent Recall (84.84%), and accuracy of 83.66%.



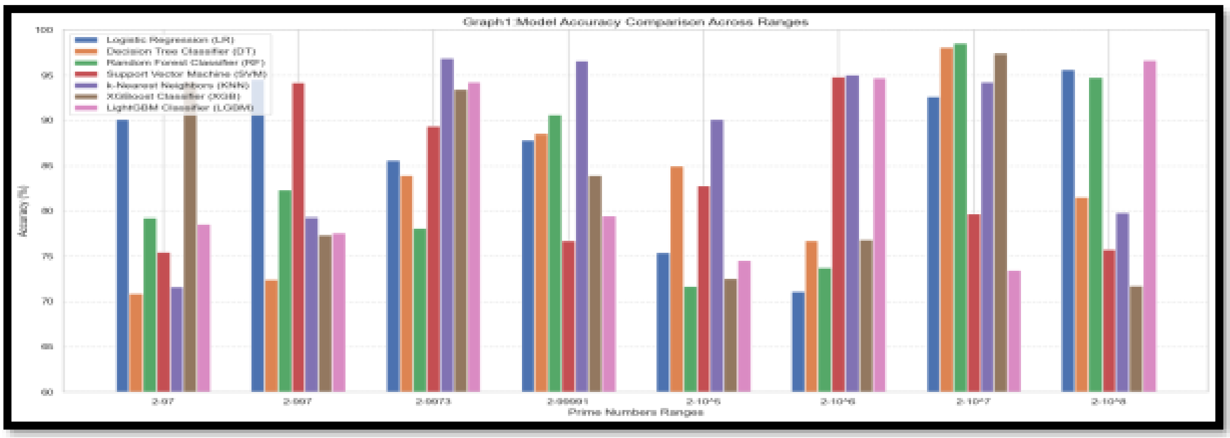
```
                         Table11:Average Performance Across All Ranges:
                         Model  Accuracy (%)  Precision (%)  Recall (%)  F1-Score  Total Training Time (s)
     k-Nearest Neighbors (KNN)      87.96000       84.84125    79.47125  80.46750                95.595000
      Logistic Regression (LR)      86.60125       83.25125    85.97750  82.98125               159.272500
    LightGBM Classifier (LGBM)      83.65875       83.46875    84.84250  87.19875                92.119625
Random Forest Classifier (RF)       83.62750       86.42125    77.93500  80.68625               116.517875
  Support Vector Machine (SVM)      83.60875       88.58000    81.58375  83.44125               196.290250
      XGBoost Classifier (XGB)      83.44875       84.24750    77.42250  82.96000               166.280125
Decision Tree Classifier (DT)       82.13250       92.01000    82.11000  84.53125               117.462625
```

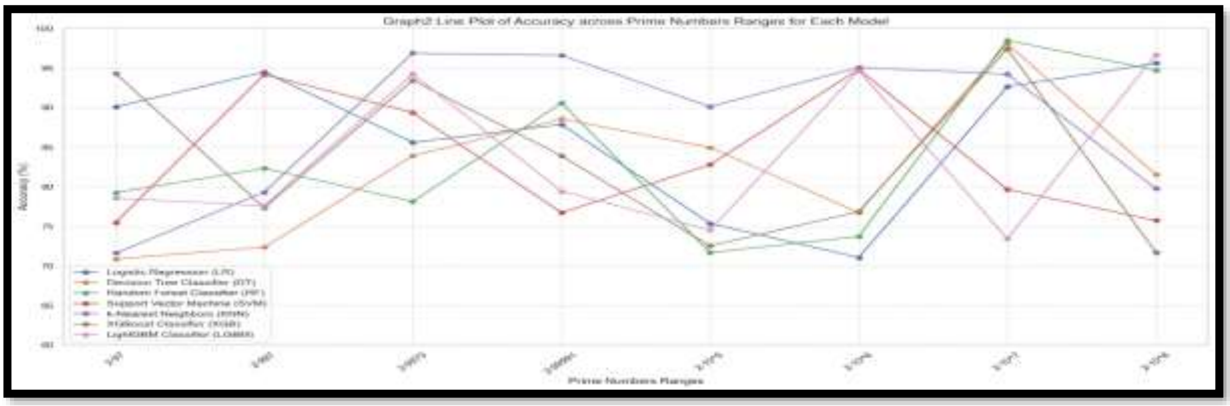
Random Forest (RF) and XGBoost (XGB) perform well in Precision but lag in Recall and F1-Score. Support Vector Machine (SVM) shows the lowest F1-Score (83.44%) and the longest training time (196.29s), indicating inefficiency. Although Decision Tree (DT) has the highest Precision (92.01%), it falls short in Accuracy and Recall. Overall, KNN proves to be the most reliable and balanced model across all ranges.

5.3. Strategic Model Recommendations for Different Range Sizes: The study's recommendations for prime classification vary strategically across different numerical ranges. For the smallest range (2–97), XGBoost is recommended when prioritizing accuracy, while Random Forest offers better efficiency. Logistic Regression provides the best balance for 2–997 classification, and KNN stands out as the superior choice for the 2–9973 through 2–10^5 ranges, delivering both top accuracy and efficiency. In the transition to larger ranges (2–10^6 to 2–10^8), LightGBM becomes increasingly favorable, particularly at 2–10^8 where it provides both the highest accuracy and best efficiency. Random Forest remains the accuracy leader at 2–10^7, though with slightly higher computational costs.



5.4. Optimal Model Selection Based on Range and Priority: Optimal prime number classification model selection varies by range size. For small to medium ranges (2–97 to 2–10^6), KNN excels in both accuracy and efficiency, while Logistic Regression is preferred for interpretability. In larger ranges (beyond 2–10^7), Random Forest and LightGBM perform best, with LightGBM showing superior efficiency. Overall, KNN achieves the highest average accuracy, and LightGBM is ideal for large-scale tasks, offering practical guidance for model selection based on range and performance needs.



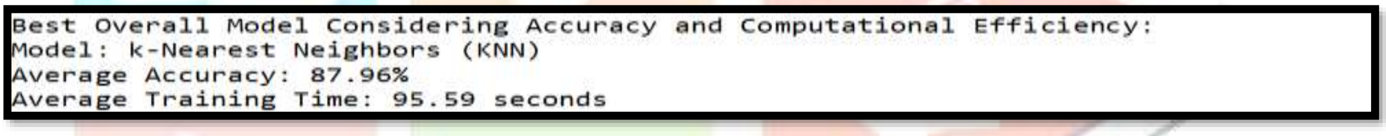


VI. Final Result

Based on the comparative analysis of machine learning models across accuracy and computational efficiency, the k-Nearest Neighbors (KNN) model stands out as the most balanced and effective choice for prime number classification.

Table 10 further validates KNN's superiority, where it performs best in four out of eight numerical ranges (2–9973, 2–99991, 2–10^5, and 2–10^6), achieving high accuracy (90.13% to 96.90%) and robust F1-scores, with modest training times. From Table 11, KNN demonstrates the highest average accuracy (87.96%), along with strong precision (84.84%), recall (79.47%), and F1-score (80.47%), while maintaining a low total training time (95.60 seconds)—indicating excellent computational efficiency. Although Logistic Regression (LR) and LightGBM (LGBM) excel in specific ranges, they fall short in overall performance: LR has higher training time (159.27 s) and LGBM shows lower average accuracy (83.66%).

Considering both global performance metrics and range-wise dominance, KNN is selected as the final recommended model for prime number classification due to its optimal balance of accuracy and computational efficiency.



```
Best Overall Model Considering Accuracy and Computational Efficiency:
Model: k-Nearest Neighbors (KNN)
Average Accuracy: 87.96%
Average Training Time: 95.59 seconds
```

VII. Conclusion

This study offers clear guidance on selecting machine-learning models for prime number classification across different numerical ranges. For small to medium ranges (2–97 to 2–10^6), k-Nearest Neighbors (KNN) is the most effective, offering the highest average accuracy (87.96%) and strong computational efficiency. Logistic Regression is a solid alternative when interpretability is important, with an average accuracy of 86.60%. For larger ranges (2–10^7 to 2–10^8), ensemble models excel: Random Forest achieves the highest accuracy (98.50%), while LightGBM balances speed and performance best (96.69%). Although XGBoost performs well in smaller ranges, its heavy computational load makes LightGBM the better choice for large-scale tasks. Overall, the findings present a data-driven approach to model selection based on range size and performance needs.

VIII. Future Scope

This study opens several avenues for further research. Advanced deep learning models like CNNs and Transformers could reveal deeper prime number patterns. Techniques such as Bayesian optimization may improve model performance through more effective hyperparameter tuning. Expanding evaluation beyond 10^8 will better assess scalability, especially for cryptographic use. Hybrid models combining KNN with ensemble methods and customized distance metrics for KNN could enhance accuracy and generalization. The consistent performance of KNN and LightGBM highlights their value as strong baselines for future studies in prime number classification.

References

[1].Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. IEEE Transactions on Information Theory, 13(1), 21–27. doi: 10.1109/TIT.1967.1053964.
[2].Miller, G. L. (1976). Riemann's hypothesis and tests for primality. Journal of Computer and System Sciences, 13(3), 300–317.https://doi.org/10.1016/S0022-0000(76)80043-8.
[3].Rivest, R. L., Shamir, A., & Adleman, L. (1978). A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM, 21(2), 120–126. http://portal.acm.org/citation.cfm?doid=359340.359342
[4].Rabin, M. O. (1980). Probabilistic algorithm for testing primality. Journal of Number Theory, 12(1), 128–138.
[5].Quinlan, J. R. (1986). Induction of decision trees. Machine Learning, 1(1), 81–106.
[6].Cortes, C., & Vapnik, V. (1995). Support-vector networks. Machine Learning, 20(3), 273–297. http://dx.doi.org/10.1007/BF00994018
[7].Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8), 1735–1780.
[8].Hosmer, D. W., & Lemeshow, S. (2000). Applied Logistic Regression. Wiley-Interscience.
[9].Breiman, L. (2001). Random Forests. Machine Learning, 45(1), 5–32.
[10].Crandall, R., & Pomerance, C. (2005). Prime Numbers: A Computational Perspective. Springer.
[11]. Zhou, Z.-H. (2012). Ensemble Methods: Foundations and Algorithms (1st ed.). Chapman and Hall/CRC. https://doi.org/10.1201/b12207
[12].Barbulescu, R., et al. (2014). The Number Field Sieve: Recent Developments. In Algorithmic Number Theory.
[13].LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, 521(7553), 436–444.
[14].Chen, T., & Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 785–794. IN & LT
[15].Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T. Y. (2017). LightGBM: A Highly Efficient Gradient Boosting Decision Tree. Advances in Neural Information Processing Systems, 30.
[16].Sammut, C., & Webb, G. I. (Eds.). (2017). Encyclopedia of Machine Learning and Data Mining. Springer.
[17].Zhang, Y., et al. (2017). Comparative study of traditional machine learning and deep learning for classification problems. Pattern Recognition Letters, 82, 1–7.
[18].Tan, P. N., Steinbach, M., & Kumar, V. (2018). Introduction to Data Mining. Pearson.
[19].Sun, S., et al. (2020). A survey of optimization methods from a machine learning perspective. IEEE Transactions on Cybernetics, 50(8), 3668–3681.
[20].Zhou, Y., Lin, W., & Zhang, Y. (2021). Deep Learning Architectures for Mathematical Sequence Modeling. Neural Networks and Applications, 34(2), 345–362.
[21].Singh, D. P. (2024), An extensive examination of machine learning methods for identifying diabetes. Tuijin Jishu/Journal of Propulsion Technology, 45(2).
[22].Singh, D. P. (2024), An extensive analysis of machine learning models to predict breast cancer recurrence. Tuijin Jishu/Journal of Propulsion Technology, 45(2).
[23].Singh, D. P. (2024), An extensive analysis of machine learning techniques for predicting the onset of lung cancer. Tuijin Jishu/Journal of Propulsion Technology, 45(4).
[24].Singh, D. P. (2024), Comprehensive analysis of machine learning models for cardiovascular disease detection and diagnosis. Tuijin Jishu/Journal of Propulsion Technology, 45(4).
[25].Singh, D. P. (2025). A comparative analysis of machine learning techniques for hypertension risk prediction and diagnostic classification. International Journal of Innovative Research in Technology, 11(12), 7183–7195. ISSN: 2349-6002.
[26].Singh, D. P. (2025), Exploring Machine Learning-Driven Advanced Regression Models For Predicting Prime Number Distributions, IJCRT | Volume 13, Issue 4 , ISSN: 2320-2882,page:b229-b239.
[27]. Singh, D. P., & Garg, S. K. (2025). Enhancing prime number classification using neural network techniques with a focus on recall efficiency and fast convergence. International Journal for Innovative Research in Multidisciplinary Field, 11(6), 7–16. ISSN: 2455-0620.