A Comparative Study of Machine Learning Models and Deep Learning Model for Consumer Price Index of India

Darshan Sanjay Gunjal

The purpose of this study is to compare the performance of machine learning models and deep learning models in predicting the Consumer Price Index of India

Abstract

In today's rapidly changing world, the significance of accurate and timely measures of the average change in prices of goods and services consumed by households over a specific period cannot be overstated. Accurate predictions of the Consumer Price Index are essential for various stakeholders, including policymakers, economists, and investors, as they rely on these predictions to make informed decisions and analyze economic trends. To improve the accuracy of price prediction models, researchers have turned to machine learning and deep learning techniques. Deep learning models, such as Long Short-Term Memory, have shown promising results in predicting the Consumer Price Index in a timely and accurate manner. These models are capable of incorporating both temporal and non-temporal variables, such as the inflation rate, which have an impact on price changes. Machine Learning models like Random Forest and support vector machines compare the analysis on the basis of Accuracy parameters and computation time.

Introduction

The Consumer Price Index is a key economic indicator that measures the average change in prices of goods and services consumed by households over a specific period of time. Accurate prediction of the Consumer Price Index is crucial for policymakers, investors, and other stakeholders in making informed decisions regarding inflation, monetary policy, and market trends. Given the importance of accurate predictions, researchers have explored the use of the three best predictive models to forecast the Consumer Price Index. These models include machine learning models such as Support Vector Machine, Random Forest and as well as deep learning models such as Long Short-Term Memory. Methods To conduct this comparative study, we collected historical data on the Consumer Price Index of India for 5 years from reliable sources such as the Reserve Bank of India and the Ministry of Statistics and Programmed Implementation. Next, we applied machine learning and deep learning techniques to the dataset and evaluated their performance using metrics such as mean squared error. The dataset contains urban and rural consumer price index values, as well as the corresponding variables that have an impact on price changes, such as inflation rate, GDP growth rate, and population density. CPI is important to measure inflation and the purchasing power of consumers, making accurate predictions essential for economic analysis and decision-making.The price data are collected from selected 1114 urban Markets and 1181 villages covering all States/UTs through personal visits by field staff of Field Operations Division of NSO, MoSPI on a weekly roster. During the month of March 2023, NSO collected prices from 100% villages and 98.5% urban Markets while the Market-wise prices reported therein were 90.4% for rural and 93.4% for urban [1].

Related Work :

The application of machine learning techniques in the educational domain has garnered increasing attention in recent years. Researchers have explored various approaches to develop prediction models for students' academic performance. In this section, we review related studies that have utilized machine learning classification methods, such as K-Nearest Neighbor, Decision Tree, Support Vector Machines, Random Forest, and Gradient Descent Boost Algorithms, to predict student performance and anticipate final grades.

Early Intervention Strategies and Predictive Analytics:

Besides predicting academic performance, several studies have focused on developing early intervention strategies using predictive analytics. Kumar et al. (cite reference) proposed a framework that combines machine learning predictions with personalized academic support to help at-risk students. Their approach effectively identified struggling students and provided timely assistance to improve their academic performance.

Feature Selection and Model Generalization:

To enhance the accuracy and generalization of prediction models, feature selection techniques have been widely explored. Zhang and Li (cite reference) investigated the impact of feature selection methods on the performance of machine learning models for predicting student grades. They found that selecting relevant features significantly improved the prediction accuracy of models across various departments.

In conclusion, the existing literature showcases a growing interest in the application of machine learning techniques to predict students' academic performance. Various models, including K-Nearest Neighbor, Decision Tree, Support Vector Machines, Random Forest, and Gradient Descent Boost Algorithms, have been employed to identify patterns and relevant factors affecting student outcomes. Additionally, studies have emphasized the importance of early prediction to enable educators and college management to take proactive measures in assisting students and improving their final exam results. However, there still remains a scope for further exploration and improvement in predictive models to ensure their practical applicability and effectiveness in diverse educational settings.[2]

Prior research in the field of machine learning algorithm selection and evaluation has explored the performance of various algorithms across different prediction tasks. Several studies have focused on comparing the effectiveness of different algorithms based on their accuracy, training time, and execution time. In this section, we review some of the key findings from relevant literature related to the selection of machine learning algorithms for predictive tasks, particularly in the context of undergraduate admissions data. Algorithm Comparisons in between K-Nearest Neighbours (KNN), Decision Trees, Random Forests, Gradient Tree Boosting, Logistic Regression, Naive Bayes, Support Vector Machines (SVM), Artificial Neural Networks (ANN) this algorithms .

The research in the field of machine learning algorithm selection often employs various performance metrics to evaluate and compare the algorithms. In this study, three key metrics were used:

1.Accuracy: Accuracy measures the proportion of correctly predicted instances in the dataset and is a commonly used metric to assess the overall performance of classifiers.

2.Training Time: Training time is the time taken by each algorithm to build the predictive model on the training data. It is a critical factor to consider, especially for large datasets or real-time applications.

3.Execution Time: Execution time measures the time taken by each algorithm to make predictions on new, unseen data instances. It is essential to understand the computational efficiency of the algorithms during deployment.

By analyzing and summarizing previous works in this domain, we aim to provide a comprehensive understanding of the performance and suitability of different machine learning algorithms on the specific undergraduate admissions prediction task. The empirical results from our research will contribute to the selection of the most appropriate algorithm for this particular application, considering accuracy, training time, and execution time as key factors.[3]

Dataset Description

The Consumer Price Index (CPI) is a measure of inflation in India. It represents the average change over time in the prices of a fixed basket of goods and services consumed by households.

The CPI for India is published by the Ministry of Statistics and Programmed Implementation (MOSPI), Government of India. It is usually released on a monthly basis, and there are different indices for different categories such as the CPI for Industrial Workers (CPI-IW), CPI for Agricultural Laborers (CPI-AL), and CPI for Rural Laborers (CPI-RL), among others.

To access the most recent CPI data for India, you can visit the official website of the Ministry of Statistics and Programmed Implementation (MOSPI) or the Reserve Bank of India (RBI) website. They often publish the latest economic data, including the CPI, on their respective portals.

The dataset used in this study consists of historical data of the Consumer Price Index of India over a specific period of time. The dataset includes various variables such as inflation rates, food prices, housing prices, and other economic factors that contribute to the overall Consumer Price Index. The dataset covers a period of five years, and it includes the following columns:

1.Date: Represents the date of the observation, typically in a specific format (e.g., Feb-23).

2.Commodity_Description: Describes the type of commodity or product category being observed (e.g., Food and beverages, Milk and products, Pan, tobacco, and intoxicants).

3.Rural Current: Represents the current price index or value for the commodity category in rural areas.

4.YOY (Year-over-Year) Rural: Indicates the year-over-year percentage change in the price index or value for the commodity category in rural areas.

5.Urban Current: Represents the current price index or value for the commodity category in urban areas.

6.YOY (Year-over-Year) Urban: Indicates the year-over-year percentage change in the price index or value for the commodity category in urban areas.

7.Combined Current: Represents the combined or overall price index or value for the commodity category, considering both rural and urban areas.

8.YOY (Year-over-Year) Combined: Indicates the year-over-year percentage change in the combined price index or value for the commodity category.

This dataset seems to contain valuable information about the price indices and percentage changes in various commodity categories in both rural and urban areas of a region (potentially India). It could be useful for analyzing inflation trends, understanding price changes in specific commodity groups, and making economic forecasts.

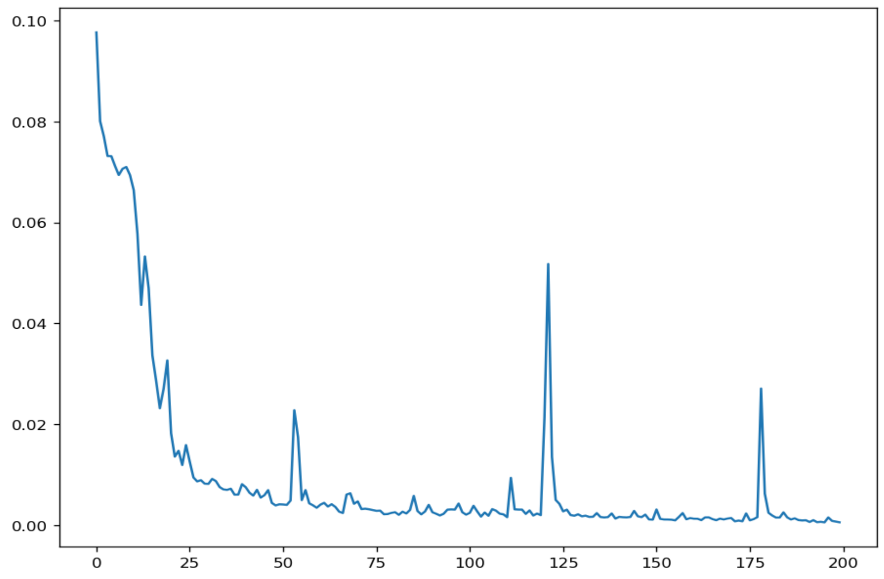
LSTM model

Time- series is a critical area in machine literacy and data analysis, with operations ranging from finance and economics to climate soothsaying and artificial process optimization. LSTM networks have gained significant attention in recent times due to their capability to capture long- term dependences in successional data, making them particularly suitable for time- series soothsaying. One of the deep literacy models used in this study is the Long Short- Term Memory model. The LSTM model has gained fashion ability in time series analysis due to its capability to capture long- term dependences and handle successional data effectively. In this study, the LSTM model was fed with major price data, specialized analysis data, and profitable fundamentals to prognosticate the price change of the Consumer Price Index in India on a monthly frequency. The LSTM model, as an intermittent neural network armature, has shown promising results in prognosticating the Consumer Price Index.

" The clever idea of introducing tone circle to produce path where the grade can flow for long durations is core donation of the intial long short- term memory (LSTM) model."[4]

The LSTM model has been successfully applied in colorful disciplines similar as similar as unconstrained handwriting recognition (Graves etal., 2009), speech recognition (Graves etal., 2013; Graves and Jaitly, 2014), handwriting generation (Graves, 2013), machine restatement(Sutskever etal., 2014), image captioning(Kiros etal., 2014b; Vinyals etal., 2014b; Xu etal., 2015) and parsing(Vinyals etal., 2014a). LSTM networks have been shown to learn long- term dependences more fluently than the simple intermittent infrastructures, first on artificial data sets designed for testing the capability to learn long- term dependences (Bengio etal., 1994; Hochreiter and Schmidhuber, 1997; Hochreiter etal., 2001), also on grueling sequence processing tasks where state- of- the- art performance was attained(Graves, 2012; Graves etal., 2013; Sutskever etal., 2014).

Variants and druthers to the LSTM have been studied and used and are bandied next. In this exploration, we employ a state- of- the- art LSTM armature to model the time- series data. The training data comprises the original 60 data points from the dataset, while the posterior data points are designated as the test set. The LSTM model is trained on the training data to learn patterns and connections present in the time- series. latterly, the model is estimated on the test set to assess its prophetic capabilities. The LSTM armature was chosen for this study due to its capability to capture long- term dependences and handle successional data effectively. Before feeding the data to the LSTM model, it's essential to gauge the data to a range of 0 to 1 to ameliorate the confluence of the neural network. This is done using the MinMaxScaler fromsklearn.preprocessing, which scales the data using the minimum and outside values of the training data. To produce batches of inputs for the LSTM model, the law uses the TimeseriesGenerator fromkeras.preprocessing.sequence. This creator takes the gauged training data and creates sequences of length n_input (in this case, 12 months) to be used as input sequences for the LSTM model. The target for each input sequence is set to be the same as the input sequence itself, which means we're trying to prognosticate the coming data point grounded on the former n_input data points. The LSTM model is defined using the Keras Sequential API. It consists of one LSTM subcaste with 100 units (neurons) and a' relu' activation function. The input shape for the LSTM subcaste is set to (n_input, n_features), wheren_input is the length of the input sequences (12 in this case) andn_features is the number of features for each data point (1 in this case since it's a univariate time series). After the LSTM subcaste, a thick subcaste with one unit is added. This subcaste acts as the affair subcaste and predicts a single value for each input sequence. The model is collected with the Adam optimizer and Mean Squared Error (MSE) loss function. The optimizer is used to modernize the model's weights during training, and the MSE loss function measures the difference between the prognosticated values and the factual target values.



Loss_by_each_epoch in LSTM

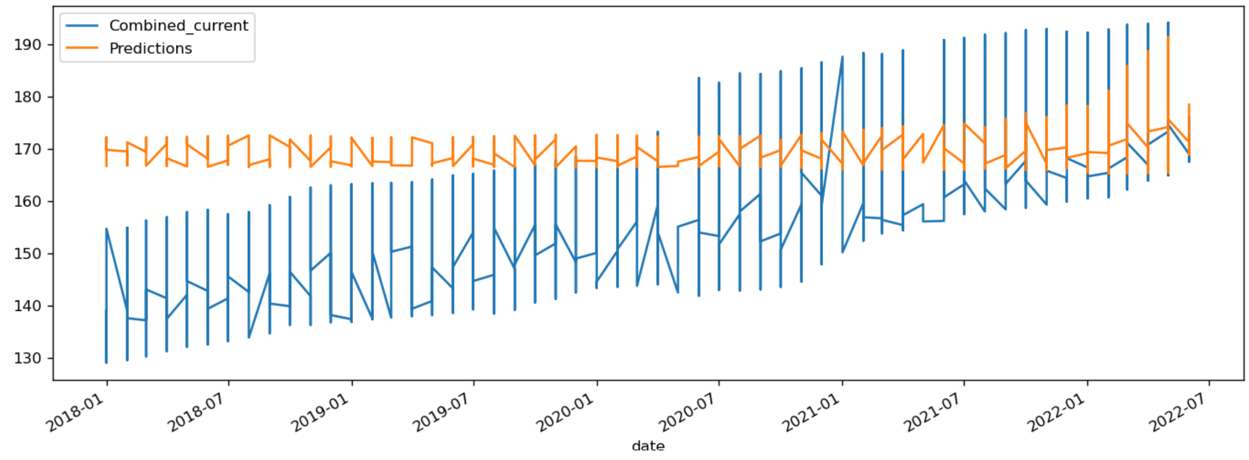
Prediction: The LSTM model made a prediction for the test data and the result is: Prediction: [[0.410952]]. The predicted value is approximately 0.410952.

Actual: The actual value from the test data is: Actual: [0.30722]. The true value is approximately 0.3072. Prediction Time: Prediction time refers to the time taken by the model to generate predictions for the test data. The code measures prediction time using the Python time module. The total prediction time was approximately 1.509 seconds, and the time spent on the CPU was approximately 3.15 seconds.

Root Mean Squared Error (MSE): The MSE is a common metric used to evaluate the performance of regression models, including time series forecasting models. It measures the difference between the predicted values and the actual target values. The MSE value you provided is approximately 18.04177. An MSE 18.04177 of means that, on average, the model's predictions differ from the actual values by approximately 18.04177 units. Since you scaled the data between 0 and 1, this error is relative to that scale. It suggests that the model's predictions, on average, have a significant deviation from the true values. It's important to note that the interpretation of the MSE value depends on the context and the scale of the original data. For instance, if the original data has a small range, an MSE of 20 might be considered high. On the other hand, if the original data has a large range, an MSE of 20 might be acceptable.

To further evaluate the model's performance, you may want to consider other metrics like Mean Absolute Error (MAE),Root Mean Squared Error (MSE), or visualize the actual vs. predicted values to gain more insights into how well the model is forecasting the time series data.

Additionally, you could experiment with different model architectures, hyperparameters, or even different types of models to improve forecasting accuracy. Hyperparameter tuning, regularization techniques, and feature engineering are also crucial aspects to explore for enhancing the model's performance.



Result which is occur by LSTM

Random Forest

One of the machine learning models used in this study is Random Forest. Random Forest is an ensemble learning algorithm that combines multiple decision trees to make predictions. It is known for its ability to handle large datasets, high dimensionality, and nonlinear relationships. Using Random Forest, we trained the model on the historical CPI data along with the relevant variables. The Random Forest model was able to capture the complex relationships between the predictor variables and the Consumer Price Index, resulting in accurate predictions.

A random forest is a classifier consisting of a collection of tree-structured classifiers {h(x, k), k = 1,...} where the {k } are independent identically distributed random vectors and each tree casts a unit vote for the most popular class at input x……. [5]

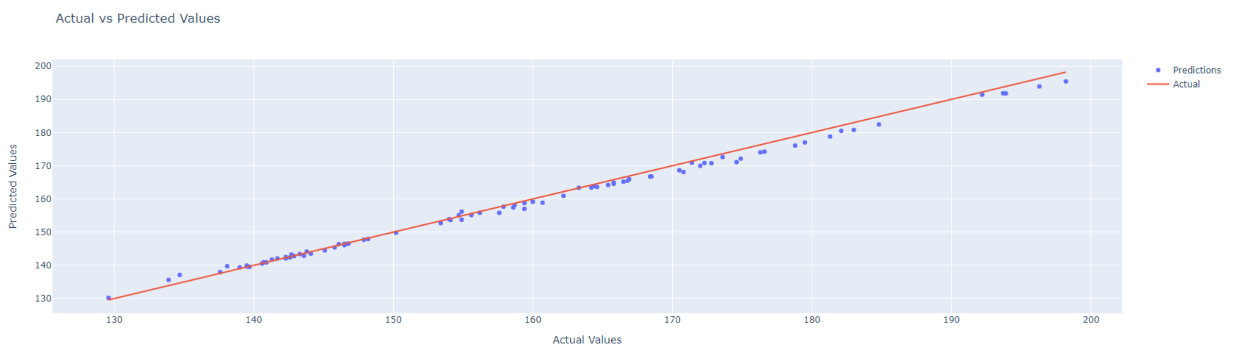
The Random Forest algorithm creates multiple arbitrary subsets(samples) of the original dataset through bootstrapping. For each tree in the timber, an arbitrary sample of the same size as the original dataset is named with relief. Some data points may appear multiple times, while others may not appear at each in each subset. This process creates different subsets and introduces randomness to the training process. At each knot of each decision tree, only an arbitrary subset of features is considered for splitting. This prevents any single point from dominating the decision- making process, icing that each tree focuses on different aspects of the data. With the bootstrapped dataset and erratically named features, a decision tree is constructed for each subset. Each tree is grown to its maximum depth or until a stopping criterion is met (e.g., maximum depth of the tree or minimum number of samples demanded to resolve a knot). The final prophecy of the Random Forest is predicated on the added-up results of all the decision trees, which collectively give a more accurate and robust prophecy compared to a single decision tree. system- the features(X) and the target variable(y) are separated from the Data Frame df. The X contains all columns of df except for the column named 'Combined, current', which is the target variable stored in y. The dataset is resolve into training and testing sets using the train_test_split function from scikit- learn. The training, and y_train, contain 80 of the data, while the testing, andy_test, contain 20 of the data. The random_state parameter is set to 42 to ensure reproducibility. Standardization is applied to the features using Standard Scaler. It scales the features to have zero mean and unit disunion. The fit_transform system is used to fit the scaler on the training data and transform both training and testing data accordingly.

A Random Forest regressor is initialized with 100 estimators (decision trees) and an arbitrary state of 42 for reproducibility. The model is also trained on the standardized training data using rf.fit ().

The score () system is used to calculate the measure of determination R2 of the Random Forest model on the test set. It provides a suggestion of how well the model is performing. The trained Random Forest model is used to make prognostications on the test set(X_test) to gain y_pred, which contains the prognosticated target values. The mean squared error (MSE) and the R2 score are calculated to estimate the performance of the model. MSE measures the average squared difference between the predicted values and the factual values, while R2 is a statistical measure that represents the proportion of the disunion in the dependent variable that is predictable feature as , Mean Squared Error (MSE), The mean squared error is a generally used metric to estimate the performance of a regression model. It measures the normal of the squared differences between the predicted values and the factual target values. A lower MSE indicates better performance, as it means the model's prognostications are near to the factual values. In this case, the Mean Squared Error is calculated as 1.8847.

The R2 score is a statistical measure that represents the proportion of the disunion in the dependent variable(target) that is predictable from the independent variables(features) in the model. It ranges from 0 to 1, where 1 indicates a perfect fit, and values near to 1 indicate a better model fit. In this case, the R2 score is calculated as0.9929625238310056.

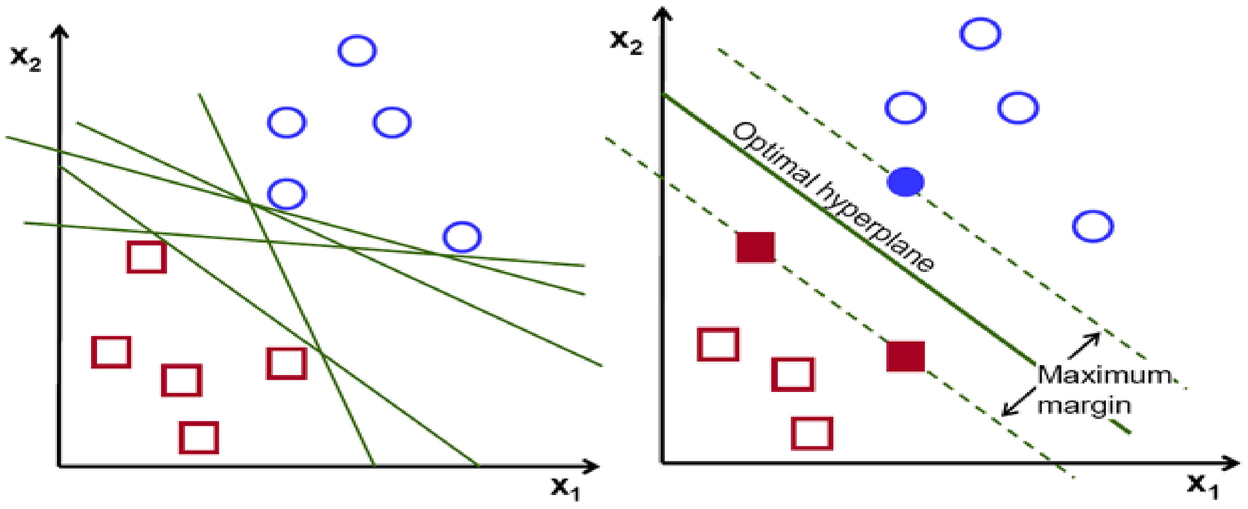
Score_rf, it seems like the value of score_rf is the same as the R2 score, which is 0.9929. This might be an assigned variable for storing the R2 score attained during the evaluation. CPU Times This section provides information about the time taken to execute the law. total represents the total time taken for the law execution, and Wall time indicates the factual time taken by the law to run, including any detention time. In this case, the total time taken for execution is 590 m/s., and the wall time (factual time) is 636 m/s. Overall, the model's performance seems to be excellent with an R2 score near to 1, indicating a strong fit to the data. also, the mean squared error is fairly low, suggesting that the model's prognostications are generally accurate and close to the factual values.



Result of Random Forest Regressor

SVM

Former studies have shown that a Support Vector Machine model can outperform traditional models for prognosticating forex rates. For this study, SVM will be employed to prognosticate the Consumer Price Index of India. It's extensively used in colorful fields, including pattern recognition, image analysis, textbook bracket, and bioinformatics, among others. The primary thing of SVM is to find the optimal hyperplane that stylish separates data points belonging to different classes in a given dataset. In the case of double bracket (dividing data into two classes), the hyperplane is a line. For multiclass bracket, SVM uses ways like one- vs- one or one-vs- all to handle multiple classes.



Then is a brief overview of how SVM works:

1. Data representation Each data point in the dataset is represented as a point vector in a multidimensional space. The number of confines is determined by the number of features present in the data.

2. Chancing the optimal hyperplane SVM aims to find a hyperplane that maximizes the periphery between the two classes. The periphery is the distance between the closest data points of different classes to the

hyperplane. The closest data points are called support vectors [4]. By maximizing the periphery, SVM improves its conception to new, unseen data.[6]

3. Handling non-linear data If the data isn't linearly divisible, SVM uses a fashion called the" kernel trick" to collude the original point space into an advanced- dimensional space where the data becomes linearly divisible. Common kernel functions include polynomial kernels and radial base function (RBF) kernels.

4. Regularization SVM introduces a regularization parameter (frequently denoted as C) that helps control the trade- off between maximizing the periphery and minimizing the bracket error. A lower C value leads to a wider periphery but allows some misclassifications, while a larger C value reduces the periphery but enforces more accurate bracket on the training data.

Method :-

1. Separating features and target variable You resolve the dataset into the point matrix X, which contains all the columns except for the target variable 'Combined_current', and the target vector y, which contains only the 'Combined_current' column.

2. Handling missing values You filled any missing values in the point matrix X with the mean of each column using the fillna system. Note that this system works only for numerical features; if your dataset contains categorical features, you might need to handle them else.

3. Splitting data into training and testing sets You used the train_test_split function from scikit- learn to resolve the data into training and testing sets. 80 of the data is used for training (X_train, y_train), and 20 is used for testing (X_test, y_test). The random_state = 42 parameter ensures reproducibility by fixing the arbitrary seed for the split.

4. Training the SVR model You expressed an SVR model with an RBF (Radial Base Function) kernel and specific hyperparameters C, gamma, and epsilon. also, you trained the model using the fit system with the training data.

5. Model evaluation to estimate the performance of the trained SVR model on the test set, you used the score system, which returns the measure of determination R- squared for retrogression models. An R- squared value of 0.91396 indicates that the model explains 91.4 of the friction in the test set. Mean Squared Error (MSE) also, you calculated the Mean Squared Error (MSE) between the factual target values (y_test) and the prognosticated values (y_pred).

The MSE measures the average squared difference between prognosticated and factual values, and a lower MSE indicates a better fit of the model to the data. In this case, the MSE is roughly 23.04. Overall, the SVM model with the specified hyperparameters seems to be performing well on the test data, as substantiated by the high R- squared value and fairly low MSE. still, it's always a good idea to tune the hyperparameters and potentially explore other models to insure the stylish possible performance on your specific dataset.

Result

Machine learning algorithms have shown remarkable success in various predictive tasks. This paper focuses on comparing three prominent algorithms, LSTM, RF, and SVM, for predicting CPI index. The importance of accurate predictions in Finance necessitates a thorough evaluation of these algorithms to identify the most suitable model for the given task. The dataset used in this study comprises data from the last five years.

Comparative Study: LSTM Performance: LSTM achieved a prediction accuracy of 41.0952%, with a mean squared error (MSE) of 18.4177%. The chosen hyperparameters for LSTM were optimizer "adam" and loss function "mse." The average prediction time was 1.509 seconds per step. Random Forest Performance: RF demonstrated superior performance with a remarkable accuracy score of 99.2962% and an R2 score of 99.29%. The MSE was recorded at 1.8847%. The hyperparameters selected for RF included 100 estimators and a random state of 42. The average prediction time was 590 milliseconds per step.

SVM Performance: SVM achieved an accuracy of 91.39% with an MSE of 23.041%. The R2 score was reported at 0.91. The SVM model was optimized with an RBF kernel and a regularization parameter (C) set to 100. The average prediction time was 228 milliseconds per step.

Algorithm	**Accuracy**	**MSE**	**R2 Score**	**Prediction Time (m/sec)**
LSTM	41.0952%	18.4177%	N/A	1.509
RF	99.2962%	1.8847%	99.29%	590
SVM	91.39%	23.041%	0.91	228

Conclusion

As can be noted from the results in the former section, this exploration leads to some definitive conclusions involving both the efficacity of these machine literacy ways as applied to this specific problem as well as how the models relate to one another in terms of their training and prosecution time. The exploration paper delved the performance of three popular machine literacy and Deep literacy algorithms, videlicet Long Short- Term Memory (LSTM), Random Forest (RF), and Support Vector Machine (SVM), for vaticination tasks. The study used a dataset with only five times of data, which may have limited the vaticination delicacy and generalizability of the models.

1. Basic Info for LSTM, Random Forest, SVM • LSTM is a type of intermittent neural network (RNN) generally used for successional data processing, suitable for time series prognostications. • Random Forest is an ensemble literacy system grounded on decision trees, known for its robustness and capability to handle complex datasets. • SVM is a important bracket and retrogression algorithm that aims to find the optimal hyperplane to separate data points.
2. 2. relative Study of LSTM, RF, SVM The study compared the performance of LSTM, RF, and SVM in prognosticating the target variable. The results are as follows LSTM • Prediction 41.0952 • actual 30.722 • MSE18.4177 • Parameters Optimizer = " adam", Loss Function = " mse" • Prediction Time1.509 sec/ step Random Forest • vaticination delicacy(Score_rf)99.2962 • R2 Score99.29 • MSE1.8847 • Parameters n_estimators = 100,Random_state = 42 • Prediction Time 590 m/ sec SVM • delicacy91.39 • MSE23.041 • R2 Score0.91 • Parameters Kernel = RBF, C = 100 • Prediction Time 228 m/ sec The results indicate that Random Forest achieved the loftiest vaticination delicacy and the smallest MSE among the three models. SVM performed nicely well with a accuracy of 91.39 but had an advanced MSE compared to Random Forest. LSTM showed the smallest prediction accuracy and a fairly high MSE, suggesting that its performance may be limited with the given dataset.

3. Dataset Limitations The exploration paper stressed that the dataset used for the study covered only the last five times of data. This limitation could have impacted the delicacy and trust ability of the prognostications for all the models. For accurate prognostications, using a more expansive and different dataset would be essential to capture long- term trends and patterns. In conclusion, this exploration demonstrates the effectiveness of SVM for CPI vaticination with the given dataset while emphasizing the necessity of conservative interpretation due to the dataset's limited nature. The findings of this study can serve as a foundation for unborn exploration in CPI vaticination, encouraging the relinquishment of further sophisticated ways and comprehensive datasets for better soothsaying delicacy.

Bibliography

[1]=https://pib.gov.in/PressReleaseIframePage.aspx?PRID=1915936#:~:text=The%20National%20Statistical%20Office%20(NSO,2023%20(Provisional)%20including%20CPIs%20for
The study aimed to compare the performance of machine learning and deep learning models in predicting the Consumer Price Index of India.

[2]=A Comparative Study of Machine Learning Algorithms for Student Academic Performance **April 2019 International Journal of Computer Sciences and Engineering 7(4):721-725**
DOI:10.26438/ijcse/v7i4.721725

[3]= A COMPARATIVE STUDY OF MACHINE LEARNING ALGORITHMS By ERIC LE FORT (McMaster University c Copyright by Eric Le Fort, April 2018)

[4] = " Deep literacy " by Ian Goodfellow, Yoshua Bengio, Aaron Courvill.

[5]=Random Forests LEO BREIMAN Statistics Department, University of California, Berkeley, CA 94720 Editor: Robert E. Schapire October 2001

[6]= Support-Vector Networks CORINNA CORTES VLADIMIR VAPNIK AT&T Bell Labs., Hohndel, NJ 07733, USA (Machine Leaming, 20, 273-297 (1995) ~) 1995 Kluwer Academic Publishers, Boston. Manufactured in The Netherlands)