

PROJECT REPORT

GDP FORECASTING IN INDIA USING MACHINE LEARNING AND DEEP LEARNING

PROJECT REPORT

SUBMITTED BY

NEERAJ V(KSD17CS074)

SANDEEP K(KSD17CS082)

THARA RAGHUNATH(KSD17CS094)

VISHNU CHANDRA M C(KSD17CS099)

To

the APJ Abdul Kalam Technological University

in partial fulfillment of the requirements for the award of the degree

of

BACHELOR OF TECHNOLOGY In *COMPUTER SCIENCE ENGINEERING*



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

LBS COLLEGE OF ENGINEERING KASARAGOD

KASARAGOD – 671542, KERALA

JANUARY 2021

DECLARATION

We undersigned hereby declare that the Project report, GDP FORECASTING USING MACHINE LEARNING AND DEEP LEARNING, submitted for partial fulfillment of the requirements for the award of degree of Bachelor of Technology of the APJ Abdul Kalam Technological University, Kerala is a bonafide work done by us under supervision of Mrs. Pragisha K. This submission represents our ideas in our own words and where ideas or words of others have been included, We have adequately and accurately cited and referenced the original sources. We also declare that We have adhered to ethics of academic honesty and integrity and have not misrepresented or fabricated any data or idea or fact or source in our submission. This report has not been previously formed the basis for the award of any degree, diploma or similar title of any other University.

PLACE: POVAL

DATE:14/01/21

NAME:

NEERAJ V

SANDEEP K

THARARAGHUNATH

VISHNU CHANDRA MC

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
LBS COLLEGE OF ENGINEERING KASARAGOD**



CERTIFICATE

This is to certify that the project report entitled “**GDP FORECASTING USING MACHINE LEARNING AND DEEP LEARNING**” submitted by **NEERAJ V, SANDEEP K, THARA RAGHUNATH, VISHNU CHANDRA MC** on 14/01/2021 to the APJ Abdul Kalam Technological University in partial fulfilment of the requirements for the award of the degree of Bachelor of Technology in Information Technology is a bonafide record of the work done by them under our supervision and guidance.

Mr. Sarith Divakar M

Assistant Professor
Department of CSE
(Guide)

Mrs. Pragisha K

Assistant Professor
Department of CSE
(Seminar coordinator)

Mrs. Smitha Mol MB

Department of CSE
(Head of Department)

ACKNOWLEDGEMENT

It is with extreme respect that We remember the names of all those who have been of great help and guidance throughout our project. Firstly, We would like to thank Almighty for giving us the wisdom and grace for presenting this project. With a profound sense of gratitude, We would like to express our heartfelt thanks to our guide, Mr Sarith Divakar, Assistant Professor, Department of Computer Science and Engineering for his expert guidance, co-operation and immense encouragement in pursuing this Project. We are extending our gratitude to our Project coordinator, Mrs.Pragisha K, Assistant Professor, Department of Computer Science and Engineering for her co-operation and support. We are very much thankful to Dr.Smitha Mol MB , Head of the Department of Computer Science and Engineering, for providing necessary facilities and for her sincere co-operation. We extend our sincere thanks to all the teachers of the Department of CSE and to all our friends for their help and support.

NEERAJ V

SANDEEP K

THARA RAGHUNATH

VISHNU CHANDRA MC

CONTENTS

- 1 INTRODUCTION
- 2 EXISTING SYSTEM
- 3 DRAWBACKS OF EXISTING SYSTEM
- 4 LITERATURE REVIEW
- 5 DESIGN
 - PROPOSED SYSTEM
 - DATASETS
 - PREPARATION AND PREPROCESSING
 - OVERVIEW OF TOOLS
 - NUMPY
 - PANDAS
 - IPYTHON
 - JUPYTER NOTEBOOK
 - EXPERIMENTAL STUDY
 - LINEAR REGRESSION
 - LOGISTIC REGRESSION
 - MLP REGRESSOR
 - LASSO
 - LSTM
 - ARCHITECTURE
 - ADVANTAGES
 - DISADVANTAGES
- 6 CONCLUSION
- 7 REFERENCES

LIST OF FIGURES

- 1 ARCHITECTURE OF PROPOSED SYSTEM
- 2 DATASET
- 3 LSTM STRUCTURE
- 4 LSTM CELL
- 5 LSTM STATE
- 6 FORGET STATE
- 7 INPUT GATE
- 8 OUTPUT GATE
- 9 HIDDEN GATE

ABSTRACT

Gross domestic product (GDP) is the single standard indicator used across the globe to indicate the health of a nation's economy: one single number that represents the monetary value of all the finished goods and services produced within a country's borders in a specific period. GDP may be easy to define but it is complex to calculate, and different countries employ different methods. In India, the rate of GDP is fixed on the basis of average enhancement or decline in the production in agriculture, manufacturing and service sector. If we say that there is a 2% increase in GDP of India, then it implies that the economy of India is growing at the rate of 2%. But often, in these statistics the inflation rate is not included. In India, the GDP is calculated in every three months and the figures of GDP depend on production rate of major economic sectors. We pursue a new approach to forecasting by employing a number of machine learning algorithms, a method that is data driven, and imposing limited restrictions on the nature of the true relationship between input and output variables. We apply the available Machine learning algorithms on raw data of advanced and emerging economies of India and find the best of these algorithms which can outperform traditional statistical models, thereby offering a relevant addition to the field of economic forecasting

CHAPTER 1

INTRODUCTION

GDP refers to the production of all goods and their services of a country or nation within a period of time and is one of the crucial factors of the economy which is to be measured annually. It is the aggregate statistic of all economic activities and captures the broadest coverage of the economy than other macro economic variables. It is the market value of all final goods and services produced within the borders of a nation in a year. It is often considered the best measure to see how the economy is performing. GDP can be measured in three ways. First, the Expenditure approach, it consists of household, business and government purchases of goods and services and net exports. Second the Production approach, it is equal to the sum of the value added at every stage of production (the intermediate stages) by all industries within the country, plus taxes and fewer subsidies on products in the period. Third is Income approach, it is equal to the sum of all factor income generated by production in the country (the sum of remuneration of employees, capital income, and gross operating surplus of enterprises i.e. profit, taxes on production and imports less subsidies) in a period (Yang, 2009- 2010). GDP can be calculated by using following formula:

$$\text{GDP} = \text{Consumption} + \text{Investment} + \text{Government Expenditure} + \text{Balance of trade.}$$

GDP prediction is a crucial job in the economy and business analysis. It provides the way to set up future business plan and take timely decision about the financial market and economy. The issues of GDP has become the most concerned amongst macro economy variables and data on GDP is regarded as the important index for assessing the national economic development and for judging the operating status of macro economy as a whole (Ning et al. 2010) . GDP is the aggregate statistic of all economic activity and captures the broadest coverage of the economy than other macro-economic variables. it is equal to the sum of all factor income generated by production in the country (the sum of remuneration of employees, capital income, and gross operating surplus of enterprises i.e. profit, taxes on production and imports less subsidies) in a period (Yang 2009; Ard 2010) Besides these, it is also the vital basis for government to set up economic developmental strategies and policies. Therefore, an accurate prediction of GDP is necessary to get an insightful idea of future

health of an economy since the data on GDP is actually represented the past activities in summary form, which is not very helpful to frame suitable economic development strategies, economic policies and allocation of funds on different priorities for government as well as individual firms in a particular industry. It needs a reliable estimate of GDP in some period ahead, which is only possible by forecasting GDP as accurately as possible by suitable sophisticated time series modeling, since it is not easy to identify the variables those effects on GDP precisely. The researcher was motivated to undertake this study dealing with the GDP issues in India as not many studies have been done attempting to forecast the GDP as well as prediction of growth rates in various forms in India. However an attempt to forecast this macro variable only as point estimates has been of very little help for the managers and policy makers since variability is the key in decision making when certain level of risk is involved.

1.1 EXISTING SYSTEM

- There has been an increasing amount of research on using large data sets to measure or predict macroeconomic indicators. The theoretical literature is adapting existing or developing new statistical and econometric methods for the analysis of large data sets with a large number of explanatory variables.
- Many of the Indian data scientists are working together to find effective Machine learning model which gives accurate results.
- There are so many effective algorithms used today, some of the algorithms gives effective results . XG Boost, Light GDM, Decision Trees are effective algorithms used for GDP prediction.

1.2 DRAWBACKS OF EXISTING SYSTEM

- Accuracy is not optimum and there is still room for improvement.
- **Focus on complete data:** missing or corrupt data is generally unsupported.
- **Focus on linear relationships:** assuming a linear relationship excludes more complex joint distributions.
- **Focus on fixed temporal dependence:** the relationship between observations at different times, and in turn the number of lag observations provided as input, must be diagnosed and specified.
- **Focus on univariate data:** many real-world problems have multiple input variables.
- **Focus on one-step forecasts:** many real-world problems require forecasts with a long time horizon

CHAPTER 2

LITERATURE REVIEW

GROSS DOMESTIC PRODUCT AS A MODERN ECONOMIC INDICATOR

**Published by Topi Tjukanov, Helsinki Metropolia University of Applied Sciences,
October 2011**

Economic growth of a country is measured in terms of an increase in the size of a nation's economy. A broad measure of an economy's size is its output. The most widely-used measure of economic output is the GDP. The three basic ways to determine a nation's GDP are; the Expenditure approach, the Production approach and the Income approach.

The Expenditure Approach of determining GDP adds up the market value of all domestic expenditures made on final goods and services in a single year, including consumption expenditures, investment expenditures, government expenditures, and net exports. Add all of the expenditures together and you determine GDP.

The Production approach, also called the Net Product or Value added method requires three stages of analysis. First gross value of output from all sectors is estimated. Then, intermediate consumption such as cost of materials, supplies and services used in production final output is derived. Then gross output is reduced by intermediate consumption to develop net production.

The Income Approach of determining GDP is to add up all the income earned by households and firms in the year. The total expenditures on all of the final goods and services are also income received as wages, profits, rents, and interest income. GDP is determined by adding together all of the wages, profits, rents, and interest income.

The three methods of measuring GDP should result in the same number, with some possible difference caused by statistical and rounding differences. The credibility of data is always a significant concern in any form of research. An advantage of using the Expenditure Method is data integrity. The source data for expenditure components is considered to be more reliable than for either income or production components. GDP as examined using the Expenditure Approach is reported as the sum of four components. The formula for determining GDP is:

$$C + I + G + (X - M) = \text{GDP} \quad (1)$$

Where: C = Personal Consumption Expenditures

I = Gross Private Fixed Investment

G = Government Expenditures and Investment

X = Net Exports

M = Net Imports

The GDP forecasting involves the application of both statistical and mathematical models to predict future developments in the economy. It allows economists to review past economic trends and forecast how recent economic changes will alter the patterns of past trends.

FORECASTING GDP USING ARIMA AND ARTIFICIAL NEURAL NETWORKS MODELS UNDER INDIAN ENVIRONMENT

**Published by Sunitha.G , Sampath Kumar.K , JyothiRani.S.A , Haragopal.V.V in
ResearchGate on April 2018**

Auto-regressive Integrated Moving Average (ARIMA) Models:

Autoregressive Integrated Moving Average models (ARIMA models) were popularized by George Box and Gwilym Jenkins in the early 1970s. It's an iterative process that involves four stages; identification, estimation, diagnostic checking and forecasting of time series. ARIMA models are a class of linear models that is capable of representing stationary as well as nonstationary. They do not involve independent variables in their construction, but rather make use of the information in the series itself to generate forecasts. ARIMA models therefore, rely heavily on autocorrelation patterns in the data. ARIMA methodology of forecasting is different from most methods because it does not assume any particular pattern in the historical data of the series to be forecast. It uses an interactive approach of identifying a possible model from a general class of models. The chosen model is then checked against the historical data to see if it accurately describes the series. Most of the traditional forecasting models therefore, provide a limited number of models relative to the complex behaviour of many time series with little guidelines and statistical tests for verifying the validity of the selected model.

Artificial Neural Networks:

ANNs imitate the learning process of the human brain and can process problems involving non-linear and complex data even if the data are imprecise and noisy. Thus they are ideally suited for the modeling of agricultural data which are known to be complex and often non-linear. A very important feature of these networks is their adaptive nature, where “learning by example” replaces “programming” in solving problems. This feature makes such computational models very appealing in application domains where one has little or incomplete understanding of the problem to be solved but where training data is readily available. These networks are “neural” in the sense that they may have been inspired by neuroscience but not necessarily because they are faithful models of biological neural or cognitive phenomena. In fact majority of the network are more closely related to traditional mathematical and/or statistical models such as non-parametric pattern classifiers, clustering algorithms, nonlinear filters, and statistical regression models than they are to neurobiology models. Neural networks (NNs) have been used for a wide variety of applications where statistical methods are traditionally employed. They have been used in classification problems, such as identifying underwater sonar currents, recognizing speech, and predicting the secondary structure of globular proteins. In time-series applications, NNs have been used in predicting stock market performance. As statisticians or users of statistics, these problems are normally solved through classical statistical methods, such as discriminant analysis, logistic regression, Bayes analysis, multiple regression, and ARIMA time-series models. It is, therefore, time to recognize neural networks as a powerful tool for data analysis.

Methodology

In this study, the GDP data analysed is collected from the official website (<http://www.rbi.org.in>) of Reserve Bank of India (RBI). The yearly aggregate GDP data (Market Price) at current prices is taken for the study from 1951-52 to 2015-16. For forecasting aggregate GDP, Box – Jenkins and Neural Network methods are used. In this section, the results of forecasting using these two methods are presented. The reported results are then analysed and compared. These two methods are compared by Root Mean Squared Error (RMSE) and Mean Absolute Percentage Error (MAPE).

NOWCASTING NEW ZEALAND GDP USING MACHINE LEARNING ALGORITHMS

**Published by Adam Richardson, Thomas van Florentestein Mulder, Tugrul Vehbi
– Reserve Bank of New Zealand on July 2018**

K-Nearest Neighbour Regression (KNN)

KNN is a widely used non-parametric method that uses a distance function, in our case the Euclidean distance, to measure the similarity between the data used for training and testing purposes. The algorithm finds which k most observations in the training set have features most similar to the features of the hold out set and predict the outcome of the new data to be the outcome of the average of the k observations previously defined. The prediction is based on the mean of the k –most similar instances. We choose k using the grid search method available in Python’s Scikit-Learn library.

Autoregressive Model (AR)

As our simple benchmark, we use a univariate AR model of order 1 for quarterly GDP growth (y_t)

$$y_t = \alpha_0 + \alpha_1 y_{t-1} + u_t$$

where α_0 and α_1 are parameters and u_t is the residual term.

KNN has RMSE of 0.491 and AR has RMSE of 0.543 by the calculations.

Here we evaluate the real-time performance of popular ML algorithms in obtaining accurate nowcasts of the real gross domestic product (GDP) growth for New Zealand. We estimate several ML models over the 2009-2018 period using multiple vintages of historical GDP data and multiple vintages of a large features set comprising approximately 550 domestic and international variables. We then compare the forecasts obtained from these models with the forecasting accuracy of a naive autoregressive benchmark as well as other data-rich methods such as a factor model, a large Bayesian VAR and the combined GDP nowcasts obtained from the suite of statistical models used at the RBNZ. We find that the majority of the ML models are able to produce more accurate forecasts than those of the AR and other statistical benchmarks.

CHAPTER 3

DESIGN

3.1 PROPOSED SYSTEM

- The main aim of this project was to create a ML model that would help in predicting the expected GDP of each state of India with minimum margin of error for any given year in the future such as 2020 and beyond.
- This is very beneficial since by this data we could predict the growth of different states of the country and it will be able to predict how the GDP must be used in an effective way for the benefit of the state.
- We will analyze the latest dataset from <https://niti.gov.in/>.
- We will train the data and validate the data and use various effective
- Machine Learning Algorithms which produce more accurate results.
- We will also use some deep learning techniques for accurate results.
- The major objective is to provide an effective ML model which produces accurate results than the existing system.

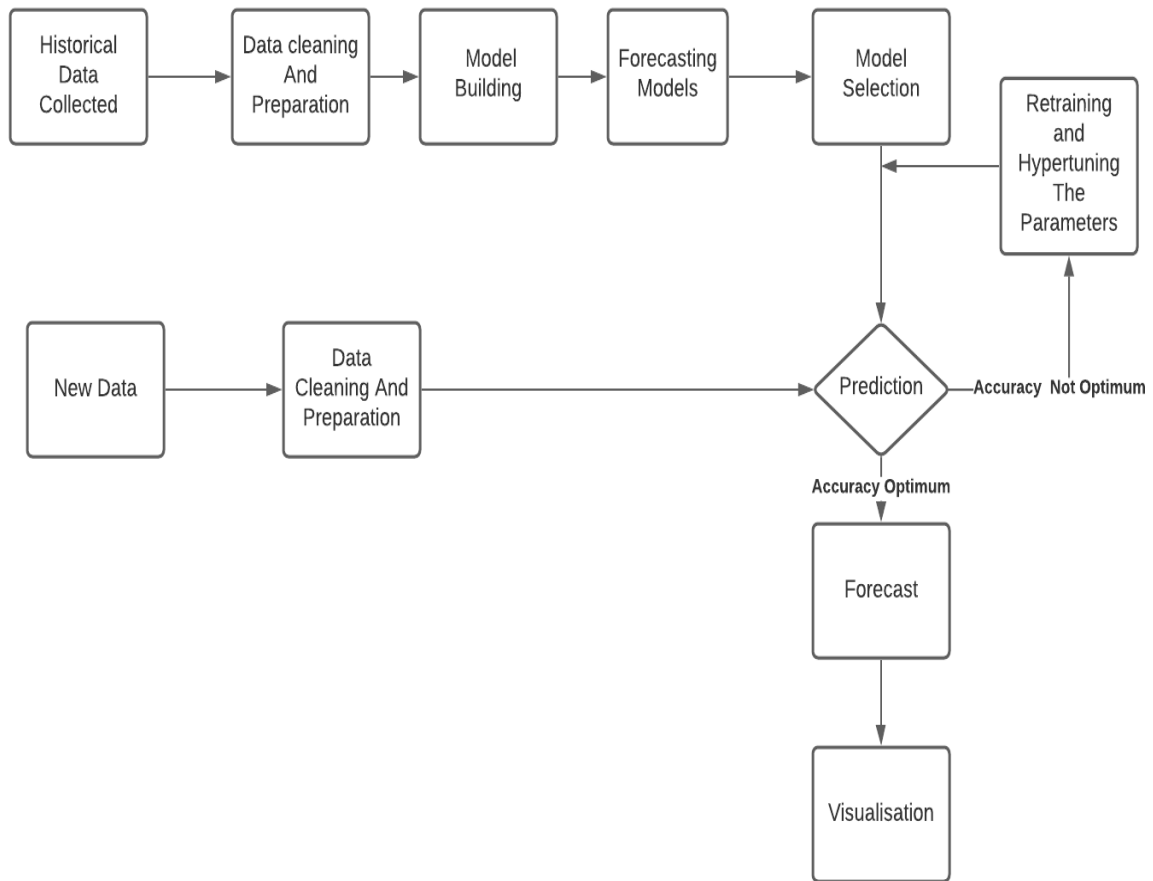














Figure: Architecture of proposed system

3.2 DATASETS

The data is harvested at scale from the Niti Ayog website and the Reserve Bank of India . While we harvest hundreds of features for each state, in order to avoid challenges of over fitting the prediction models, selected key indicators are utilised for model building purposes. The selection of these specific training features is equally guided by the availability of reliable and continuous time-series variables. These training features include the following:

-  Aggregate Expenditure
-  Aggregate Receipts
-  Capital Expenditure
-  Capital Receipts
-  Fiscal Deficits
-  Interest Payments
-  Outstanding Liabilities
-  Own Tax Revenues
-  Revenue Deficits
-  Revenue Receipts
-  Social Sector Expenditure
-  Nominal GSDP Series.

3.3 PREPARATION AND PREPROCESSING

The RAW data was in XLSX format which was converted to a well-known CSV format that could be later easily read by python pandas library. The unknown values that were originally represented by a '-' was replaced by NA's by pandas. The Data was given for 29 states and 2 Union Territories, from the year 1981 to 2016, some of the data was missing for some of the states, due to the fact that the data was started to be collected at a later period of time, or

maybe the state wasn't even formed in that year, such years were removed from the data by using remove NA's in python pandas, and the clean and preprocessed data was dumped to a CSV file.

[6]: `niti.describe().T`

	count	mean	std	min	25%	50%	75%	max
Capital_Receipts	660.0	2428.893939	4110.573309	-5116.00	238.7500	811.50	2583.750	37639.0
Aggregate_Receipts	660.0	7791.672727	10883.932609	38.00	1143.5000	3191.50	9882.000	72074.0
Social_Sector_Expenditure	660.0	2623.565152	3470.775608	9.00	381.2500	1114.00	3512.000	24268.0
Interest_Payments	660.0	1060.228030	1757.027349	0.00	81.7000	290.00	1240.000	11870.0
Own_Tax_Revenues	660.0	2604.877273	4224.076558	0.00	90.0000	879.50	3174.750	33540.0
Fiscal_Deficits	660.0	1630.949848	2688.436374	-360.00	159.9500	517.00	1736.150	18620.0
Outstanding_Liabilities	660.0	417.522303	596.079415	1.42	49.6725	177.21	505.725	4067.4
Aggregate_Expenditure	660.0	7663.422727	10658.752422	43.00	1175.5000	3078.50	9737.250	72668.0
Revenue_Receipts	660.0	5362.778788	7044.667651	37.00	892.5000	2347.50	7210.750	48438.0
Revenue_Expenditure	660.0	6022.030303	8317.578141	30.00	864.0000	2275.50	7755.000	52280.0
Revenue_Deficits	660.0	659.255455	1824.575250	-4328.00	-62.3750	41.25	594.675	18583.0
Capital_Expenditure	660.0	1641.445455	2554.712652	13.00	289.5000	797.00	1851.750	21622.0
Nominal_GSDP_Series	660.0	37596.312121	57944.592770	52.00	2963.0000	14311.00	44635.000	486766.0

[3]: `niti = pd.read_csv('NITIDATA.csv')`

[4]: `niti.head()`

	Capital_Receipts	Aggregate_Receipts	Social_Sector_Expenditure	Interest_Payments	Own_Tax_Revenues	Fiscal_Deficits	Outstanding_Liabilities	Aggregate_Expenditure	Revenue_Receipts	Revenue_Expenditure	Revenue_Deficits	Capital_Expenditure	Nominal_GSDP_Series
0	325.0	1590.0	532.0	81.7	582.0	222.1	81.50	1610.0	1265.0	1161.0	-104.0	449.0	8191.0
1	187.0	709.0	171.0	30.0	66.0	-27.4	43.41	758.0	522.0	357.0	-165.0	401.0	2516.0
2	702.0	1690.0	452.0	107.1	277.0	335.8	106.33	1791.0	988.0	929.0	-59.0	862.0	7353.0
3	345.0	1370.0	406.0	68.6	531.0	246.5	80.76	1442.0	1025.0	903.0	-122.0	539.0	7427.0
4	116.0	576.0	137.0	37.0	234.0	112.0	30.76	607.0	460.0	401.0	-59.0	207.0	3386.0

Figure: Dataset

3.4 OVERVIEW OF TOOLS

3.4.1 NUMPY

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. The ancestor of Numpy, Numeric, was originally created by Jim Hugunin with contributions from several other developers. In 2005, Travis Oliphant created Numpy by incorporating features of the competing Numarray into Numeric, with extensive modifications. Numpy is open-source.

3.4.2 PANDAS

Pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series.

3.4.3 IPYTHON

IPython (Interactive Python) is a command shell for interactive computing in multiple programming languages, originally developed for the Python programming language, that offers introspection, rich media, shell syntax, tab completion, and history. IPython provides the following features: Interactive shells (terminal and Qt-based), A browser-based notebook interface with support for code, text, mathematical expressions, inline plots and other media, Support for interactive data visualization and use of GUI toolkits, Flexible, embeddable interpreters to load into one's own projects, Tools for parallel computing

3.4.4 JUPYTOR NOTEBOOK

The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. Uses include: data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more.

3.5 EXPERIMENTAL STUDY

After the preprocessing of dataset, we experimentally used some machine learning and deep learning algorithms. These algorithms are especially regression models and every model is from python's *Scikit Learn* Library.

These are the Machine Learning and Deep Learning algorithms that we used so far:

- Linear Regression
- Logistic Regression
- MLP Regressor
- LASSO

Before using these algorithms, The preprocessed data is split into two, and stored in X and Y variable.

Data splitting is the act of partitioning available data into two portions; usually for cross-validatory purposes. One portion of the data is used to develop a predictive model and the other to evaluate the model's performance. Here the X portion is used to develop a predictive model and Y portion is used to evaluate the models performance.

```
In [10]: X = niti.iloc[:, 0:-1]

In [11]: X.shape
Out[11]: (660, 12)

In [12]: X.head()
Out[12]:
```

	Capital_Receipts	Aggregate_Receipts	Social_Sector_Expenditure	Interest_Payments	Own_Tax_Revenues	Fiscal_Deficits	Outstanding_Liabilities	Aggregate_Expe
0	325.0	1590.0	532.0	81.7	582.0	222.1	81.50	
1	187.0	709.0	171.0	30.0	66.0	-27.4	43.41	
2	702.0	1690.0	452.0	107.1	277.0	335.8	106.33	
3	345.0	1370.0	406.0	68.6	531.0	246.5	80.76	
4	116.0	576.0	137.0	37.0	234.0	112.0	30.76	

```

In [13]: Y = niti.iloc[:, -1]

In [14]: Y.head()
Out[14]: 0    8191.0
         1    2516.0
         2    7353.0
         3    7427.0
         4    3386.0
         Name: Nominal_GSDP_Series, dtype: float64

In [15]: Y.shape
Out[15]: (660,)

In [16]: X_train, X_test, Y_train, Y_test = train_test_split(X, Y)

```

Before attempting to fit a model to observed data, first need to perform a feature scaling. Feature scaling is a method used to normalize the range of independent variables or features of data. In data processing, it is also known as data normalization and is generally performed during the data preprocessing step.

Feature Scaling

```
In [17]: from sklearn.preprocessing import StandardScaler

In [18]: scaler = StandardScaler()

In [19]: X_train = scaler.fit_transform(X_train)

In [20]: X_test = scaler.transform(X_test)
```

Here we used Mean-Normalisation and Transform.

$$x' = \frac{x - \text{average}(x)}{\max(x) - \min(x)}$$

where x is an original value, x' is the normalized value. There is another form of the means normalization which is when we divide by the standard deviation which is also called standardization.

The Resultant value is stored in X_test variable. The above steps similar in all the models.

1.Linear Regression

Linear regression attempts to model the relationship between two variables by fitting a linear equation to observed data. One variable is considered to be an explanatory variable, and the other is considered to be a dependent variable (X and Y).

Model Creation

```
In [21]: from sklearn import datasets, linear_model
In [22]: regr = linear_model.LinearRegression()
In [23]: regr.fit(X_train, Y_train)
Out[23]: LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
In [24]: regr.score(X_test, Y_test)
Out[24]: 0.9589428413610699
```

Result

Model Analysing

```
In [25]: from sklearn.metrics import mean_squared_error, r2_score
In [26]: Ya = Y_test
          Yp = regr.predict(X_test)
          mse = mean_squared_error(Ya, Yp)
          R2 = r2_score(Ya, Yp)
In [27]: 'MSE : {}, RMSE : {}, R2-Score : {}'.format(mse, np.sqrt(mse), R2)
Out[27]: 'MSE : 149350783.38085592, RMSE : 12220.915815963053, R2-Score : 0.9589428413610698'
In [ ]:
```

Conclusion

Linear Regression model does not give accurate result that we expected, hence linear regression model is not feasible for our purpose.

2.Logistic Regression

Logistic regression is another technique borrowed by machine learning from the field of statistics.

It is the go-to method for binary classification problems (problems with two class values).

Logistic regression uses an equation as the representation, very much like linear regression.

Input values (x) are combined linearly using weights or coefficient values (referred to as the Greek capital letter Beta) to predict an output value (y). A key difference from linear regression is that the output value being modeled is a binary values (0 or 1) rather than a numeric value.

Below is an example logistic regression equation:

$$y = e^{(b_0 + b_1 * x)} / (1 + e^{(b_0 + b_1 * x)})$$

Where y is the predicted output, b0 is the bias or intercept term and b1 is the coefficient for the single input value (x). Each column in your input data has an associated b coefficient (a constant real value) that must be learned from our training data.

```

Model Creation

In [46]: from sklearn.linear_model import LogisticRegression

In [61]: logmodel=LogisticRegression(max_iter=5000)
logmodel.fit(X_train,Y_train)

Out[61]: LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
intercept_scaling=1, l1_ratio=None, max_iter=5000,
multi_class='auto', n_jobs=None, penalty='l2',
random_state=None, solver='lbfgs', tol=0.0001, verbose=0,
warm_start=False)

In [63]: from sklearn.metrics import mean_squared_error, r2_score

In [64]: Ya = Y_test
Yp = logmodel.predict(X_test)
mse = mean_squared_error(Ya, Yp)
R2 = r2_score(Ya, Yp)

In [66]: 'MSE : {}, RMSE : {}, R2-Score : {}'.format(mse, np.sqrt(mse), R2)

Out[66]: 'MSE : 628534808.8242425, RMSE : 25070.596499170944, R2-Score : 0.8639490906929266'

```

Conclusion

Logistic Regression model does not give accurate result that we expected, hence linear regression model is not feasible for our purpose.

3. MLP Regressor

A `sklearn.neural_network.MLPRegressor` is a multi-layer perceptron regression system within `sklearn.neural_network`.

Model Creation

```
In [58]: from sklearn.neural_network import MLPRegressor

In [59]: model = MLPRegressor(hidden_layer_sizes=(12, 12, 12, 12, 12, 12, 12, 12, 12, 12), max_iter=5000)

In [60]: model.fit(X_train, Y_train)

Out[60]: MLPRegressor(activation='relu', alpha=0.0001, batch_size='auto', beta_1=0.9,
    beta_2=0.999, early_stopping=False, epsilon=1e-08,
    hidden_layer_sizes=(12, 12, 12, 12, 12, 12, 12, 12, 12, 12),
    learning_rate='constant', learning_rate_init=0.001, max_fun=15000,
    max_iter=5000, momentum=0.9, n_iter_no_change=10,
    nesterovs_momentum=True, power_t=0.5, random_state=None,
    shuffle=True, solver='adam', tol=0.0001, validation_fraction=0.1,
    verbose=False, warm_start=False)

In [61]: model.score(X_test, Y_test)

Out[61]: 0.956749943353966
```

Model Analysing

```
In [62]: from sklearn.metrics import mean_squared_error, r2_score

In [63]: Ya = Y_test
    Yp = model.predict(X_test)
    mse = mean_squared_error(Ya, Yp)
    R2 = r2_score(Ya, Yp)

In [64]: 'MSE : {}, RMSE : {}, R2-Score : {}'.format(mse, np.sqrt(mse), R2)

Out[64]: 'MSE : 105890216.39165458, RMSE : 10290.297196468846, R2-Score : 0.956749943353966'
```

4. LASSO

lasso (least absolute shrinkage and selection operator; also Lasso or LASSO) is a regression analysis method that performs both variable selection and regularization in order to enhance the prediction accuracy and interpretability of the resulting statistical model

Lasso was originally formulated for linear regression models. This simple case reveals a substantial amount about the estimator. These include its relationship to ridge regression and best subset selection and the connections between lasso coefficient estimates and so-called soft thresholding. It also reveals that (like standard linear regression) the coefficient estimates do not need to be unique if covariates are collinear.

Model Creation

```
In [17]: from sklearn import linear_model

In [38]: clf = linear_model.Lasso(max_iter=5000)

In [39]: clf.fit(X_train, Y_train)

C:\Users\vishnuchandra.m\anaconda3\lib\site-packages\sklearn\linear_model\_coordinate_descent.py:476: ConvergenceWarning: Objective did not converge. You might want to increase the number of iterations. Duality gap: 2479857317.6875916, tolerance: 154536246.40130707
positive)

Out[39]: Lasso(alpha=1.0, copy_X=True, fit_intercept=True, max_iter=5000,
    normalize=False, positive=False, precompute=False, random_state=None,
    selection='cyclic', tol=0.0001, warm_start=False)

In [40]: clf.score(X_test, Y_test)

Out[40]: 0.9658163805298355

In [41]: from sklearn.metrics import mean_squared_error, r2_score

In [42]: Ya = Y_test
    Yp = clf.predict(X_test)
    mse = mean_squared_error(Ya, Yp)
    R2 = r2_score(Ya, Yp)

In [43]: 'MSE : {}, RMSE : {}, R2-Score : {}'.format(mse, np.sqrt(mse), R2)

Out[43]: 'MSE : 137866760.23046616, RMSE : 11741.667693750584, R2-Score : 0.9658163805298355'
```


From the above Models We can conclude that MLPRegressor is accurate compared to other regressor methods, but these results are not satisfactory. These results does not meet our requirements hence we should not recommend these models for our future purpose.

Our next plan is to test with popular ML and DL models like LSTM or ARIMA that are perfect for timeseries forecasting.

3.6 LSTM

The LSTM model proposed by Hochreiter et al. is a variant of the recurrent neural network (RNN). It builds a specialized memory storage unit that trains the data through a time backpropagation algorithm. It can solve the problem that the RNN has no long-term dependence. The schematic diagram of the LSTM structure is shown in Figure

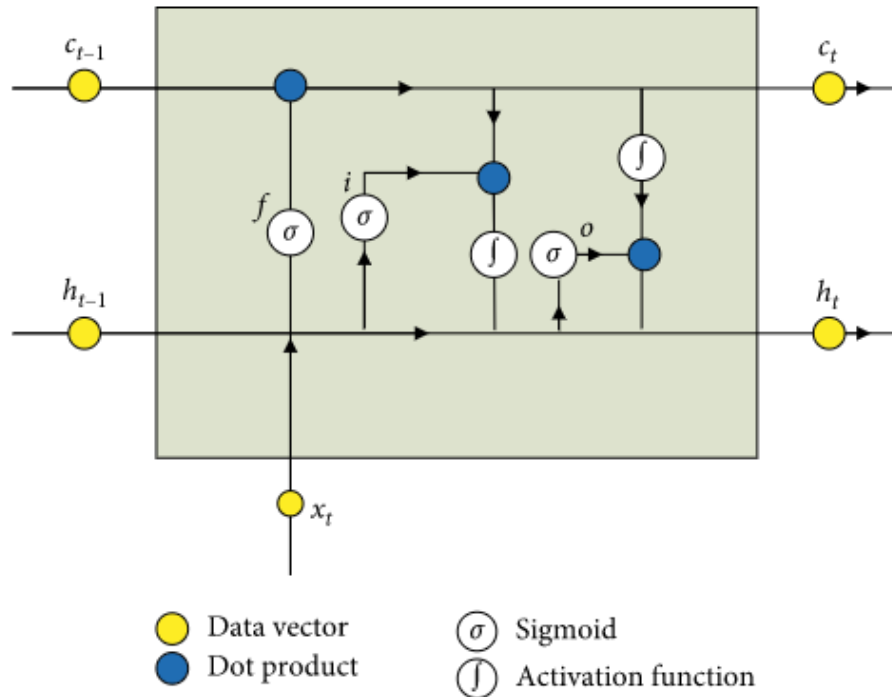


Figure :LSTM structure

The standard LSTM can be expressed as follows. Each step t and its corresponding input sequence are $X=\{X_1,X_2,\dots,X_t\}$, the input gate is i_t , the forget gate is f_t , and the output gate is o_t . Memory cell state controls data memory and oblivion through different gates. The formula is as follows:

$$i_t = \sigma(W_i x_t + U_i h_t),$$

$$f_t = \sigma(W_f x_t + U_f h_t),$$

$$o_t = \sigma(W_o x_t + U_o h_t),$$

$$\tilde{c}_t = \tanh(W_c x_t + U_c h_t).$$

The memory cell state c_t^j of the unit time t of the j th LSTM is as follows:

$$C_t^j = i_t^j \circ \tilde{c}_t + f_t^j \circ c_{t-1}^j$$

After the memory cell state is updated, calculate the current hidden layer h_t^j :

$$h_t^j = o_t^j \circ \tanh(c_t^j)$$

where W is the weight matrix of the input, U is the state transition weight matrix, σ is the sigmoid function, \tanh is the hyperbolic tangent function, h_t is the hidden state vector of the output, \tilde{c}_t the new cell state after the adjustment and update, and “ \circ ” indicates point multiplication. The three types of gates jointly control the information entering and leaving the memory cell state, and the input gates adjust new information into the memory cells; the forgetting gate controls how much information is stored in the memory cells and how much information can be output by the output gate definition. The gate structure of the LSTM allows the information in the time series to form a balanced long short-term dependency.

3.6.1 ARCHITECTURE

The **long-term memory** is usually called the **cell state**. The looping arrows indicate recursive nature of the cell. This allows information from previous intervals to be stored with in the

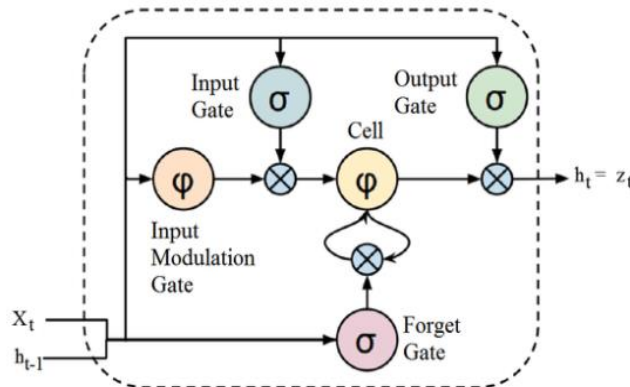


Figure :LSTM Cell

LSTM cell. Cell state is modified by the forget gate placed below the cell state and also adjust by the input modulation gate. From equation, the previous cell state forgets by multiply with the forget gate and adds new information through the output of the input gates.

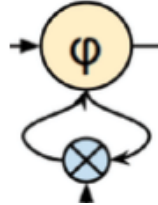


Figure: LSTM cell state

The **remember vector** is usually called the **forget gate**. The output of the forget gate tells the cell state which information to forget by multiplying 0 to a position in the matrix. If the output of the forget gate is 1, the information is kept in the cell state. From equation, sigmoid function is applied to the weighted input/observation and previous hidden state.

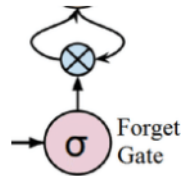


Figure: Forget Gate

The **save vector** is usually called the **input gate**. These gates determine which information should enter the cell state / long-term memory. The important parts are the activation functions for each gates. The input gate is a **sigmoid** function and have a range of [0,1]. Because the equation of the cell state is a summation between the previous cell state, sigmoid function alone will only add memory and not be able to remove/forget memory. If you can only add a float number between [0,1], that number will never be zero / turned-off / forget. This is why the input modulation gate has an **tanh** activation function. Tanh has a range of [-1, 1] and allows the cell state to forget memory.

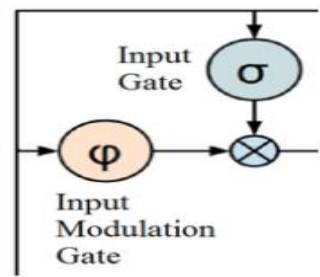


Figure: Input Gate

The **focus vector** is usually called the **output gate**. Out of all the possible values from the matrix, which should be moving forward to the next hidden state

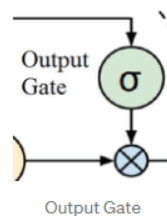


Figure : Output Gate

The **working memory** is usually called the **hidden state**. What information should I take to the next sequence? This is analogous to the hidden state in RNN and HMM.

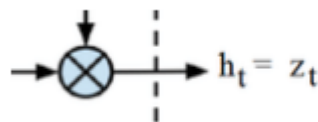


Figure :HiddenState

3.7 ADVANTAGES

- Traditional time series forecasting methods focus on univariate data with linear relationships and fixed and manually-diagnosed temporal dependence.
- Neural networks add the capability to learn possibly noisy and nonlinear relationships with arbitrarily defined but fixed numbers of inputs and outputs supporting multivariate and multi-step forecasting.
- Recurrent neural networks add the explicit handling of ordered observations and the promise of learning temporal dependence from context.
- LSTM generalises well.
- LSTM is powerful it learns what to keep and how much to keep from the past and the present state.
- Accuracy is increased.

3.8 DISADVANTAGES

- Difficulty in training.
- A lot of time and system resource required.

CONCLUSION

- Applications using Machine Learning are transforming our daily life.
- ML methods have recently been proposed as alternatives to time series regression models being typically used.
- Machine Learning algorithms outperform the statistical benchmarks.
- By comparing the predictive accuracy of each model we tried to understand which would give us a better output.
- Deep Learning models have an edge over ML models in terms of accuracy.
- For our future work we are going to predict gdp using LSTM a Deep Learning method.

REFERENCES

- [1] Forecasting GDP using ARIMA and Artificial Neural Networks models under Indian environment published on International Journal of Mathematics Trends and Technology (IJMTT) – Volume 56 Number 1- April 2018
- [2] IASSI Quarterly: Contributions to Indian Social Science, Vol. 37, Nos. 3 & 4, 2018
Predicting Regional Economic Activity using Artificial Intelligence (AI) Methods: Case Study with Indian States
- [3] MPRA Paper No. 95459, posted 08 Aug 2019, Nowcasting US GDP with artificial neural networks
- [4] Macroeconomic Indicator Forecasting with Deep Neural Networks Thomas R. Cook and Aaron Smalter Hall, September 29, 2017
- [5] International Business and Logistics Thesis, October 2011, GDP as a modern day economic indicator.