Report

On

Demand forecasting of Wheat using Neural Networks in India's context

BY

Shubham Kumar 2017B4A40882P

Under the guidance of

Dr. Srikanta Routroy

For

BITS F424T (9 units)

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI

Pilani Campus

(Jan-May 2022)

Contents

Acknowledgment

I would like to thank Dr. Srikanta Routroy sir for providing me an opportunity to this 9 units thesis (BITS F424T). I would also like to thank Khalid Ansari sir and Nikita Dhankar ma'am for continuous guidance and support throughout these days. Despite, your busy schedule I was never neglected with support and assistance. I would also like to thank my friends for helping with my doubts and suggesting me improvements.

Abstract

Wheat is a very crucial staple food of India. India is the second-largest producer of wheat after China. Almost every household has wheat consumption on a daily basis. With the bursting population of India, a platform to predict wheat demand is essential. This study is a prediction module for the demand for wheat using a Long Short Term Network (LSTM) and Recurrent Neural Network (RNN) in India's context. The model proposed and studied shows accurate results for short time-series data. Demand forecasting with good accuracy can prove to be a very great tool for smooth supply chain management which on behalf will reduce stock-out situations and increase customer satisfaction.

Introduction

India is a land of agriculture where the major households depend on farming as a means of income to feed their families. Around seventy percent of the Indian village population is involved in farming directly or indirectly for their daily needs. According to a study done by the Food and Agriculture Organization of India, we have eighty-two percent of the farmers which a having only small landholdings; they are likely to increase in the future as per the Food and Agriculture Organisation of India by ninety-one percent in 2030. In India, farmers face various challenges linked to the demand for crops like sometimes, they get irregular demands for a certain crop, which lead to storage challenges for the farmer. Storage of crops is an overhead cost for the farmers and the crops are likely to rot by improper storage or rodents etc. Irregularity in demands is generally more common in India in comparison to other countries, which causes poor farmers to suffer. Presently the whole demand forecasting is dependent on manual surveys, which are done after harvesting and before crop storage. Such manual surveys are very costly and don't serve the purpose of predicting demand since it is very error-prone and time-consuming as well. This affects the supply chain of crops in our country as such not up to date creates a gap between demand and supply and major times either is short of crop where he could have earn more by growing more or the farmer is left with an excess crop which burns his pocket. A seasonal crop demand forecasting for various critical crops is therefore important to handle such imbalances.

Wheat is an important crop not only in India but also from a global perspective. It is one of the oldest crops grown by humankind for its requirement for cereals. Some early transcripts from Mohen-jo-Daro suggest that wheat was cultivated even 5,000 years ago. From an Indian kitchen point of view, one cannot see any household missing wheat in its kitchen, where its consumption in the form of chappatis, bread, etc., is extensively consumed in every household for multiple meals.

Around 29.8 million [2] hectares of land in India are under crop cultivation. The majority of the land under cultivation is for wheat and rice as they are the most cultivated cereal. The production of wheat was very low when India gained its independence in 1947. Based on data from 1950-to 1951, India produced only 6.46 million tonnes of wheat, which was not enough to feed the Indian population. So based on such circumstances, India started importing wheat in large quantities from countries like the USA to fulfilling the wheat demands of the Indian population.

Later in 1961 Government of India (GoI) appointed a commission to improve crop production output based on Indian climatic and economic conditions. Based on the result of the steps taken by the government, the scenario of wheat production completely changed; a bumper rise in wheat production output was seen. The early sixties were the golden period in Indian agriculture as the 'Green Revolution' made India somewhat self-dependent in the agricultural aspect. At present, India is producing even more wheat than what is required to feed its population, storehouses are flooded with grains, and there is no shortage of grains. India started exporting the excess wheat to various countries. Recently India has an agreement with Egypt for wheat exports.

Wheat is one of the most chosen cereals in India. It falls just behind rice in its demand. According to a survey (National Survey) done in 2011-2012, they found the rural parts of India has more penetration of rice in comparison to the urban parts. An urban Indian kitchen consists of 4 Kg of wheat on 4.5 kg of rice. While in the rural parts of India, their kitchen constitutes 4.3 kg of wheat on 6 kg of rice. Both of the above-mentioned data are monthly averages.

Table 1. Rice consumption based on demographics

Commodity	Estimated based on NSS			Projected		
	1983	1993-94	2004-05	2011-12	2016-17	2021-22
Rural						
Rice	80.7	85.4	79.7	72.4	72.2	72.4
Wheat	54.3	53.5	52.2	47.9	49.0	48.1
Coarse cereals	45.1	24.3	15.5	14.9	14.7	14.5
Total cereals	180.1	163.3	147.4	135.2	135.9	135.1
Pulses	11.07	9.3	8.6	8.7	9.2	9.5
Foodgrains	191.1	172.5	156.0	143.9	145.1	144.6
Urban						
Rice	64.7	64.2	59.0	48.8	48.2	47.8
Wheat	58.6	57.4	56.5	51.2	49.8	46.6
Coarse cereals	14.1	7.7	4.4	4.4	3.9	3.6
Total cereals	137.5	129.3	119.9	104.4	101.9	98.0
Pulses	12.40	10.5	10.4	11.0	12.3	13.5
Foodgrains	149.9	139.8	130.3	115.4	114.2	111.6
Rural + Urban						
Rice	76.9	79.9	73.8	64.8	64.1	64.1
Wheat	55.3	54.6	53.5	49.0	49.2	47.6
Coarse cereals	37.8	19.8	12.6	11.5	11.0	10.8
Total cereals	169.9	154.2	139.9	125.3	124.4	122.6
Pulses	11.7	9.6	9.0	9.5	10.2	10.9
Foodgrains	181.6	163.8	148.8	134.8	134.6	133.4

Trends in per capita consumption of foodgrains in India (kg/year)

Source: ResearchGate

In this study of demand forecasting of wheat in India, two algorithms are used i) an LSTM network ii) an RNN. A recurrent neural network (RNN) is commonly used for sequential data like speech recognition, just like in Google Home and Alexa. Recurrent Neural Network also works great with time series and uses the previous timestamp to predict the future data. The issue with an RNN algorithm is vanishing gradient, which is dealt with in our second algorithm, LSTM. LSTM consists of special neurons capable of remembering important information and forgetting the useless information. While RNN neural networks have memory, LSTM has a better memory. The performance of the two algorithms is tested on the demand data related to India, which is dependent upon per capita income, substitute crop, and population on an annual basis [9]. Both of the algorithms perform well on the seventy-four-year dataset.

Literature Review

Much of the studies are focused on utilizing time series analysis approaches to estimate crop yields, but using neural network models to predict crop demands is relatively recent.

Muthusinghe et al., 2018 [9]; proposed a study that uses ML algorithms to study the yield and demand of rice in Sri Lanka. The whole study is to predict yield and demand so that better farming techniques could be employed in the country. The two algorithms taken into consideration are LSTM and the RNN algorithm.
Just before the study, they did a survey on the importance of such a prediction system. Through that survey, they found that farmers and mill owners are eager for the presence of such a facility. To carry out this study, they collected various raw data for about twenty-seven years period from various central government and village level data sources. The whole point of the study was to check out LSTM and RNN which serve better for such a prediction system. Their raw data consisted of population, income, and other cereals' demand as the factor affecting demand. Their study was compared using MSE error functions. The conclusion of the study showed LSTM to have performed better than RNN.

Balaji et al., 2020 [15]; suggested a model that applies an effective multiple linear regression-based forecasting to aid farmers in demand-based constructive farming. Demand is a dependent variable in the proposed model, while population, per capita income, taste, and per capita income are independent factors.

Devi et al., 2021 [20]; In their investigation, they discovered a growth in wheat production, the area under cultivation, and yield in Haryana utilising an Artificial Neural Network and an AutoRegressive Integrated Moving Average. After Punjab, Haryana is the second-largest wheat grower.

Amin et al., 2014 [21]; In Pakistan, a time series model for wheat forecasting has been proposed. They used Statgraphics and JMP software to fit different time series data. They discovered that Pakistan's wheat production will increase to 26623.5 thousand tonnes in 2020 and quadruple in 2060 as compared to 2010, assuming no abnormalities.

Kani et al., 2007 [22]; ANN and PSO algorithms were used to anticipate annual electricity demand in Iran's agriculture sector. They fine-tuned the coefficients and reduced the mean absolute error using the PSO algorithm. Then, utilising data from 1981 to 2005, the ANN algorithm is utilised to forecast demand.

Vijai et al., 2018 [23]; recommended a comparison study of various strategies for water demand forecasting. Artificial Neural Networks, Least Square Support Vector Machines, Deep Neural Networks, Random Forests, and multiple regression techniques were used. The algorithms were all conducted with the same purpose in mind: to forecast water consumption. In comparison to the other algorithms, the results showed that ANN performed better for short-term forecasting.

Abbasimehr et al., 2020 [25]; proposed an optimized multi-layer LSTM model which improves forecasting accuracy as it captures non-linear patterns in time series data. Their results showed that it outperformed traditional methods.

Bedi et al., 2018 [28]; proposed an empirical mode decomposition type of model which decompose the time series data into intrinsic mode functions. Each intrinsic mode function are run separately using LSTM and RNN algorithm and the output is combined and the results are compared. Jatin used it for predicting the electricity demand in Chandigarh.

Abdel-Nasser et al., 2017 [29]; proposed a forecasting model for photovoltaic power generation using three algorithms multiple regression model, boosted regression tree, and LSTM. Further proposed implementation on other renewable energy sources like wind and biomass.

Kumar et al., 2020 [30]; energy prediction model in the wind and solar to analyze the uncertainty in the microgrid system and to encourage the use of renewable energy using LSTM which is an improvement over the Shallow Neural Network.

Gamage et al., 2019 [31]; proposed a profitable crop cultivation predictor which suggests profitable vegetables to the farmers using LSTM-RNN and predicts the future prices using ARIMA. Their study is in Srilanka's context, it will help Sri Lankan Farmers to smartly choose crops to maximize their profits.

Márquez et al., 2021 [32]; proposed an ethanol fuel demand prediction model in Brazil's context. They created a univariate and multivariate model for the same. They took the substitute of ethanol as a factor for demand. Then they compared the results from LSTM-RNN and ARIMA models.

Paidipiti et al., 2020 [33]; proposed a yield prediction model for rice in India's context. They collected data from 1950 to 2017 and used ARIMA and LSTM for prediction they used Root Mean Square Error for performance checking.

Saini et al., 2020 [34]; proposed a prediction model for Energy consumption in Agricultural sectors in India using LSTM and RNN on an hourly basis. This study was useful for meeting the unexpected as well as seasonal demands.

Marndi et al., 2021 [35]; proposed the prediction of rice yields in India. The goal of this research is to assist policymakers in determining import-export volumes based on yield predictions.

Rani et al., 2020 [36]; proposed a prediction model for foodgrains using ARIMA, RNN, and CNN. The performance of the different algorithms is compared using mean square error (MSE), Root mean square error (RMSE).

Geetha et al., 2022 [37]; proposed a prediction model in the Cauvery region of India for rice production. The model is based on LSTM, and Tamil Nadu is an important state in terms of rice production.

Majhi et al., 2020 [38]; proposed a prediction model for evaporation measurement to manage water in drought-struck regions. They took temperature, wind speed, humidity, etc for the prediction using the LSTM algorithm.

Bhartra et al., 2021 [39]; proposed a prediction model for the yield of various crops in Southern parts of India such that the profit of farmers is maximized, they used Normalized Difference Vegetation Index for better prediction.

Reddy et al., 2021 [40]; proposed a prediction model for crop yield using parameters such as climate and weather conditions, disease outbursts in crops, and growing phases in crops using various machine learning algorithms.

Kowshik et al., 2021 [41]; proposed an Indian crop prediction model, they created the model taking into account acidity of the soil, weather, and percentage of essential minerals. The model also recommends fertilizers ratios for increasing crop yields.

Set of Equations

RNN Network:-

1) $h_t = f (h_{t-1}, X_t)$
2) $h_t = \tanh (W_h . h_{t-1} + W_x . X_t)$
3) $Y_t = W_y . h_t$

h_t - Current State

h_{t-1} - Previous State

X_t - Input State

f- Activation Function

W_h - Weight of Previous State

W_x - Weight of Current State

W_y - Weight of Output State

LSTM Network:-

1) Equation of Forget gate:-

   $f_t = \sigma (W_f [h_{t-1}, X_t] + b_f)$
2) Equation of Memory cell:-

   $C_t = f_t . C_{t-1} + i_t . C_t$
3) Equation of Input Gate:-

   $i_t = \sigma (W_i . [h_{t-1}, x_t] + b_i)$

   $C_t = \tanh (W_c . [h_{t-1}, x_t] + b_c)$
4) Equation of Output Gate:-

   $o_t = \sigma (W_o . [h_{t-1}, x_t] + b_o)$

   $h_t = o_t . \tanh (C_t)$

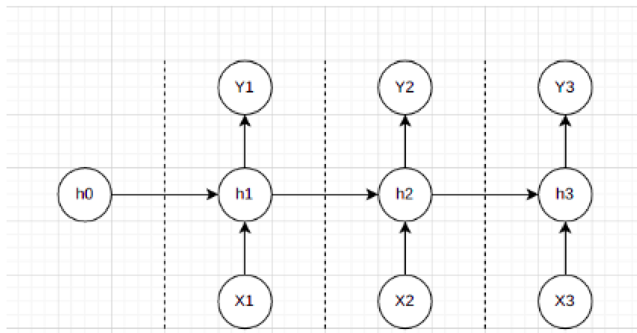
Recurrent Neural Network

Machine Learning was built upon a Neural network, a core component of its existence.; artificial learning algorithms without its existence have no existence too. A neural network is a mimic of the human brain; just like neurons of the brain are connected to each other neural network is also a connection of artificial neurons. They have brought a revolution in research around everyday activities using computer machines.

RNN (recurrent neural network) is an up-gradation to the artificial neural network. An RNN is a stronger and modified form of ANN. They are used to work with data that has some pattern in it or has a seasonality or is related to time series. David Rumelhart was the first to discover RNN in the year 1986. The whole idea of RNN is to take processed data from previous output or so-called hidden neurons for the prediction of present data. At a particular stage of RNN, data consist partially from the previous output layer and the data of that particular time period. This neural network has memory though not a very strong one; it stores information about the current stage and its previous stage.

An RNN, though having pros of memory like property, also have some significant cons like vanishing gradient and exponential gradient problem. In the first case, the neural networks start to forget about their previous data and become clueless while in the second case the neural network learns beyond what is required which makes the prediction system out of order.

Sepp Hochreiter found LSTM to deal with the cons of simple RNN network. LSTM was invented in 1997 was able to deal with vanishing gradient problem.



ht = f (ht-1 , Xt)

ht = tanh (Wh_h * ht-1 + WX_ h * Xt)

Yt = WY_h * ht

ht - Current State   ht-1 - Previous State   Xt - Current Input State   f - Activation Function

Wh_h - Weight of previous state WX_h - Weight of Current Input State WY_h - Weight of Output State

Figure 1. Structure and Formula behind a Recurrent Neural Network

Working of a Recurrent Neural Network

An RNN works in a recurrent fashion since the output of the previous act as an input to the current layer; it is possible due to its three components output layer, hidden layer, and an inner layer. An RNN is firstly fed with input in a feed forward network which is used to assign initial weights then the output is passed to the next time period network. Except for the first input at every step, there is a collaboration of two data, one coming from the present source while the other coming from the previous source. Since there is a learning in the network in a pattern of the sequence, the RNN works very greatly with sequential data or time-series data. An initial weight is assigned when the input is fit for the first time, which is altered by backward propagation as the network starts learning with more inputs. Backpropagation is a process of adjusting weights as the network goes back to reduce the error function of the network using partial derivative computation; these gradients are then subtracted from the weight. Then it adjusts the weights depending on which decreases the error of the model. That's how a Recurrent Neural Network trains itself.

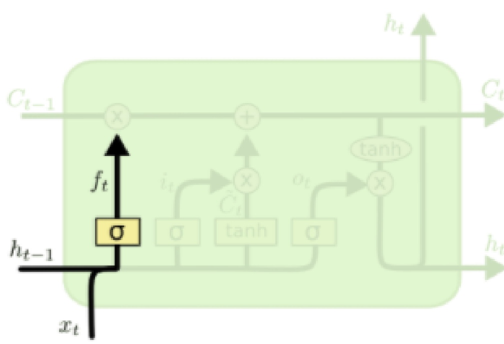
Long Short Term Memory

An LSTM network is a modified and upgraded form an RNN network that is capable of covering the cons of RNN. LSTM has a longer memory than RNN and hence deals with vanishing gradient issues of an RNN. It smartly stores data and rejects useless ones.An LSTM cell consists of four major components:- i) Input Gate;
ii) Output Gate; iii) Memory Cell; iv) Forget Gate.

Memory Cell:- This is the storage section of the LSTM neuron; it stores information after it is passed through the forget gate and the information that is passed through the input gate. Finally, this information is passed through the output gate from this memory cell to be carried forward to the next neuron.

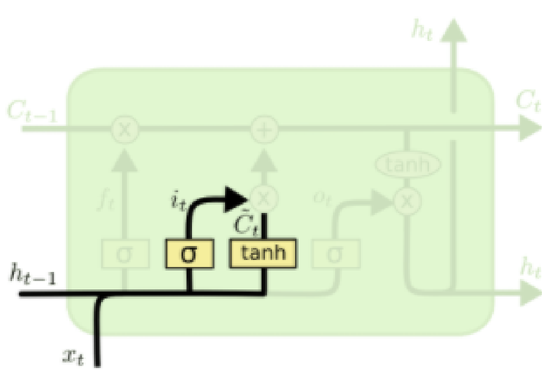
Forget Gate:- A forget gate is a special type of gate that reduce the overhead and useless information from the network. It has filters that act as a multiplier for data before proceeding forward. The value of the multiplier lies between 0 and 1. Suppose a network experience some useless information in the network; it deals with it by converting the filter multiplier value close to a 0. When the network experience some important data from the past it changes the filter multiplier close to 1 to allow the network to retain such information. This is required to improve the performance of this network.

Input Gate:- The input gate is a special type of gate that also consists of the multiplier. It is present at the input source. Suppose there is a change in context in the network the network will make the filter value close to 1 to store more of the input information. While if the context of data is more or less related to the previous context it will keep its filter value close to 0 since it already has the required information.



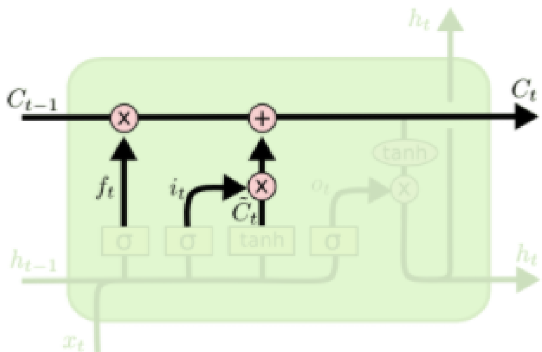
$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \ + \ b_f\right)$$

Figure 2. Forget Gate of an LSTM



$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \ + \ b_C)$$

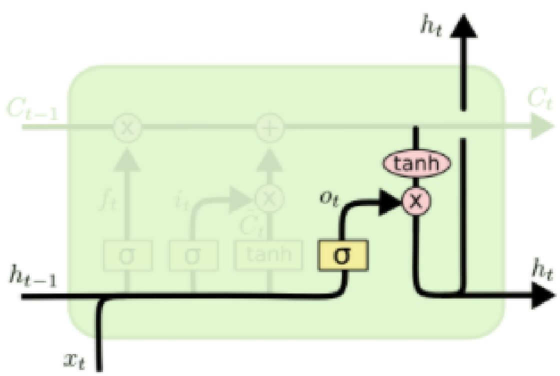
Figure 3. Input Gate of an LSTM



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Figure 4. Memory Cell of an LSTM

Output Gate:- The output gate is used for collaborating data coming from forget gate and input gate. It combines the data using computations to feed it ahead in the network as an input to the next forget gate.

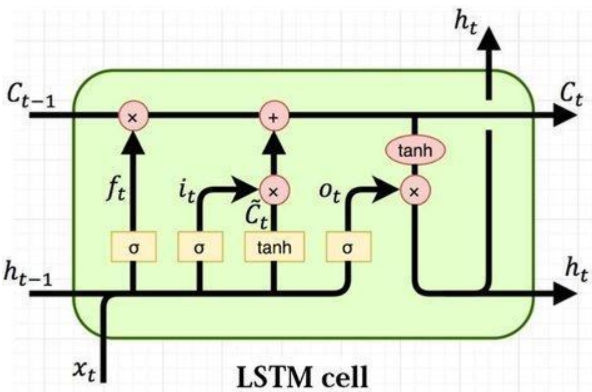


$$o_t = \sigma\left(W_o\ [h_{t-1}, x_t]\ +\ b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

Figure 5. Output gate of an LSTM

Working of an LSTM

An LSTM works just like an RNN; the only difference is that it can perform well with large data points since it deals with the vanishing gradient problem of an RNN. Otherwise, all other steps like training with train data set for assigning weights. And Backpropagation for weights updation is also present in the LSTM network too.



$$i_t = \sigma\left(x_t U^i + h_{t-1} W^i\right)$$
$$f_t = \sigma\left(x_t U^f + h_{t-1} W^f\right)$$
$$o_t = \sigma\left(x_t U^o + h_{t-1} W^o\right)$$
$$\tilde{C}_t = \tanh\left(x_t U^g + h_{t-1} W^g\right)$$
$$C_t = \sigma\left(f_t * C_{t-1} + i_t * \tilde{C}_t\right)$$
$$h_t = \tanh(C_t) * o_t$$

Figure 6. Structure and Formula behind a Long Short Term Memory

Methodology

As per the studies done in [9], the first step for the given study was to collect raw data. The first step in raw data collection was to get year-wise wheat demand data. The second step was to collect population data of India. The third step was to collect the per capita income of the Indian population. The fourth step was to collect substitute crop demand which was calculated using first retrieving cereals demand data of India and subtracting wheat demand from it. Raw data of around seventy years are collected and arranged properly in a CSV format.

Factors used for demand prediction is:

1. inex number: just for index
2. Year: year of record for wheat
3. income: per capita income of people of india
4. substitute: consumption taken in annual average of other substitute cereals in India
5. population: total population of India
6. Demand: consumption of wheat in India

Table 2. Raw Demand Data

Year	Per Capita Income (in Rs)	Substitute (in 000 Tonnes)	Population	Demand (in 000 Tonnes)
1948	1299	28311	376325200	6400
1949	1303	29490	382376948	6462
1950	1309	31210	388799073	6183
1951	1313	35010	395544369	7501
1952	1321	43169	402578596	8017
1953	1325	38999	409880595	9043
1954	1328	38285	417442811	8760
1955	1347	39498	425270695	9403
1956	1350	38753	433380978	7998
1957	1355	44076	441798578	9958
1958	1347	44225	450547679	10324
1959	1384	47320	459642165	10997
1960	1454	46807	469077190	12072
1961	1355	47071	478825608	10776
1962	1335	50863	488848135	9853
1963	1413	52425	499123324	12257
1964	1415	41615	509631500	10394
1965	1478	43098	520400576	11393
1966	1520	49870	531513824	16540
1967	1492	46293	543084336	18651
1968	1446	47624	555189792	20093
1969	1483	48939	567868018	23832
1970	1469	41254	581087256	26410
1971	1572	37649	594770134	24735
1972	1552	51101	608802600	21778
1973	1635	41604	623102897	24104
1974	1689	50303	637630087	28846

Source: Ministry of Agriculture & Farmer Welfare, Ministry of Statistics and Programme Implementation, United Nations Populations

The raw data is then tested upon the two algorithms separately for a comparative study. The parameter used for comparison are:-

Mean Absolute Error:- It is the difference between predicted and true value divided by total number of values.

$$\mathrm{MAE} = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i|$$

Mean Square Error:- It is the difference of the square of predicted and true values to power two which is divided by total number of variables.

$$\mathbf{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2$$

Root Mean Square Deviation:- It is the square root of difference between predicted and actual value to the whole square divided by total number of value.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

Normalization tools

They are used to normalize the input values.

i) Standard Scaler:- The Sklearn Python package includes a normalizing utility. It adjusts the values to the point where the mean is zero and the standard deviation is one.

ii) MinMax Scaler:- It is a normalization tool available in the Sklearn Python library. It scales the values in such a way that the value lies between [0,1]

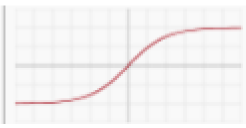
$$\boxed{x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}}}$$

Activation Functions

This function is used to do the changes of input before it is sent to the next neuron for proper functioning of a neural network or the before it is sent as an output.

i) Tangent hyperbolic function:-

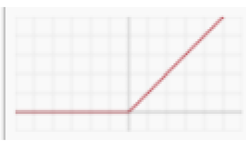
TanH



$$f(x) = \tanh(x) = \frac{2}{1+e^{-2x}} - 1$$

ii) Relu function:-

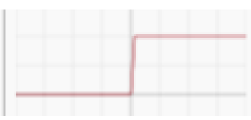
Rectified Linear Unit (ReLU)



$$f(x) = \begin{cases} 0 & \text{for} \quad x < 0 \\ x & \text{for} \quad x \geq 0 \end{cases}$$

iii) Binary Step Function:-

Binary step



$$f(x) = \begin{cases} 0 & \text{for} \quad x < 0 \\ 1 & \text{for} \quad x \geq 0 \end{cases}$$

For Normalization purposes, the MinMax Scaler function is used and the Relu function was used as an activation function. Finally, all the three error functions are used to make a comparative study.

Code

RNN Code

```
import math
import pandas as pd;
import matplotlib.pyplot as plt
from numpy import array
from numpy import hstack
from keras.layers import Dense
from keras.models import Sequential
from keras.layers import RNN, SimpleRNN
from keras.preprocessing.sequence import TimeseriesGenerator
from keras.layers import Dropout
from tensorflow.keras.optimizers import Adam
from sklearn.preprocessing import MinMaxScaler
```

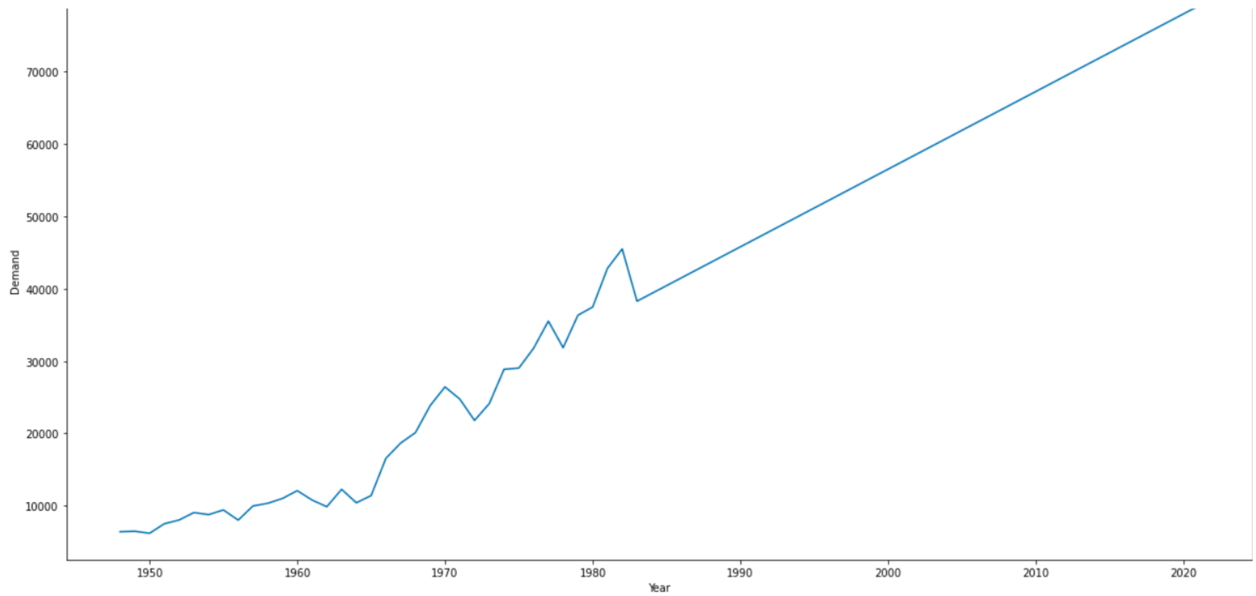
Firstly import standard Python libraries like NumPy for working with array datasets, Matplotlib for plotting, pandas for data manipulation and analysis, math libraries for various inbuilt mathematical functions, Sequential layer to define the architecture of our neural network, Dense and LSTM layers from Keras, MinMaxScaler is used to normalize the data to be able to work with an activation function.

```
import os
print(os.listdir("/content/"))
dataset = pd.read_csv('fin_dat1.csv')
```

```
dataset.head()
```

Year	Per Capita Income (in Rs)	Substitute (in 000 Tonnes)	Population	Demand (in 000 Tonnes)
1948	1299	28311	376325200	6400
1949	1303	29490	382376948	6462
1950	1309	31210	388799073	6183
1951	1313	35010	395544369	7501

```
df[["Demand (in 000' Tonnes)"]].plot(figsize = (20,10), alpha = 1)
plt.title('Demand')
plt.xlabel('Year')
plt.ylabel('Demand')
```



Now the data is normalized using a MinMax Scaler.

```
scaler = MinMaxScaler(feature_range=(0, 1))
scaled = scaler.fit_transform(array(dataset['Demand']).reshape(len(dataset['Demand']), 1))
series = pd.DataFrame(scaled)
series.columns = ['DemandScl']
```

Now the data is divided into testing and training parts.

```
number_of_test_data = 5
number_of_holdout_data = 5
number_of_training_data = len(dataset) - number_of_holdout_data - number_of_test_data
print ("total, train, test, holdout:", len(dataset), number_of_training_data, number_of_test_data, number_of_holdout_data)
```

```
datatrain = dataset[:number_of_training_data]
datatest = dataset[-(number_of_test_data+number_of_holdout_data):-number_of_holdout_data]
datahold = dataset[-number_of_holdout_data:]
```

Now the input data is prepared for the sequential model using TimeSeriesGenerator.

```
in_seq1 = array(datatrain['Per Capita Income (in Rs)'])
in_seq2 = array(datatrain['Substitute'])
in_seq3 = array(datatrain['Population'])
out_seq_train = array(datatrain['DemandScl'])
```

```
in_seq1 = in_seq1.reshape((len(in_seq1), 1))
in_seq2 = in_seq2.reshape((len(in_seq2), 1))
in_seq3 = in_seq3.reshape((len(in_seq3), 1))
out_seq_train = out_seq_train.reshape((len(out_seq_train), 1))
```

```
datatrain_feed = hstack((in_seq1, in_seq2, in_seq3, out_seq_train))
```

Preparing the RNN Model with Relu as an activation function and optimizer as Adam.

```
model = Sequential()

model.add(SimpleRNN(10, activation='relu', input_shape=(n_input, n_features), return_sequences = False))
model.add(Dense(1, activation='relu'))

adam = Adam(learning_rate=0.0001)
model.compile(optimizer=adam, loss='mse')
```

Calculating the score of the model.

```
score = model.fit_generator(generator_train, epochs=3000, verbose=2, validation_data=generator_test)
```

Code for calculating various errors functions.

```
mean = df_result['Actual'].mean()
mae = (df_result['Actual'] - df_result['Prediction']).abs().mean()
mse = (df_result['Actual']*df_result['Actual'] - df_result['Prediction']*df_result['Prediction']).abs().mean()
rmse=math.sqrt(mse)
print("mean: ", mean)
print("mae:", mae)
print("mse:", mse)
print("rmse:", rmse)
```

LSTM CODE

Firstly import standard Python libraries like NumPy for working with array datasets, Matplotlib for plotting, pandas for data manipulation and analysis, math libraries for various inbuilt mathematical functions, Sequential layer to define the architecture of our neural network, Dense and LSTM layers from Keras, MinMaxScaler is used to normalize the data to be able to work with an activation function.

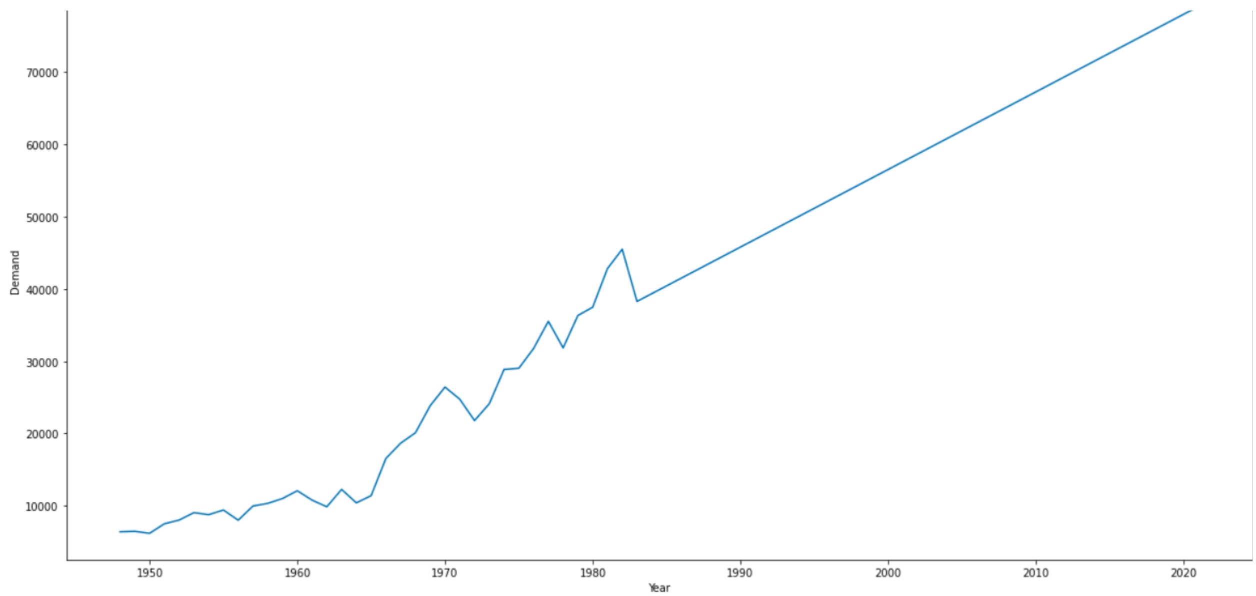
```
import math
import pandas as pd;
import matplotlib.pyplot as plt
from numpy import array
from numpy import hstack
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import LSTM
from keras.preprocessing.sequence import TimeseriesGenerator
from keras.layers import Dropout
from tensorflow.keras.optimizers import Adam
from sklearn.preprocessing import MinMaxScaler
```

```
import os
print(os.listdir("/content/"))
dataset = pd.read_csv('fin_dat1.csv')
```

```
dataset.head()
```

Year	Per Capita Income (in Rs)	Substitute (in 000 Tonnes)	Population	Demand (in 000 Tonnes)
1948	1299	28311	376325200	6400
1949	1303	29490	382376948	6462
1950	1309	31210	388799073	6183
1951	1313	35010	395544369	7501

```
df[["Demand (in 000' Tonnes)"]].plot(figsize = (20,10), alpha = 1)
plt.title('Demand')
plt.xlabel('Year')
plt.ylabel('Demand')
```



Now the data is normalized using a MinMax Sclaer

```
scaler = MinMaxScaler(feature_range=(0, 1))
scaled = scaler.fit_transform(array(dataset['Demand']).reshape(len(dataset['Demand']), 1))
series = pd.DataFrame(scaled)
series.columns = ['DemandScl']
```

Now the data is divided into testing and training parts.

```
number_of_test_data = 5
number_of_holdout_data = 5
number_of_training_data = len(dataset) - number_of_holdout_data - number_of_test_data
print ("total, train, test, holdout:", len(dataset), number_of_training_data, number_of_test_data, number_of_holdout_data)
```

```
datatrain = dataset[:number_of_training_data]
datatest = dataset[-(number_of_test_data+number_of_holdout_data):-number_of_holdout_data]
datahold = dataset[-number_of_holdout_data:]
```

Now the input data is prepared for the sequential model using TimeSeriesGenerator.

```
in_seq1 = array(datatrain['Per Capita Income (in Rs)'])
in_seq2 = array(datatrain['Substitute'])
in_seq3 = array(datatrain['Population'])
out_seq_train = array(datatrain['DemandScl'])
```

```
in_seq1 = in_seq1.reshape((len(in_seq1), 1))
in_seq2 = in_seq2.reshape((len(in_seq2), 1))
in_seq3 = in_seq3.reshape((len(in_seq3), 1))
out_seq_train = out_seq_train.reshape((len(out_seq_train), 1))
```

```
datatrain_feed = hstack((in_seq1, in_seq2, in_seq3, out_seq_train))
```

Preparing the LSTM Model with Relu as an activation function and optimizer as Adam.

```
model = Sequential()

model.add(LSTM(10, activation='relu', input_shape=(n_input, n_features), return_sequences = False))
model.add(Dense(1, activation='relu'))

adam = Adam(learning_rate=0.0001)
model.compile(optimizer=adam, loss='mse')
```

Calculating the score of the model.

```
score = model.fit_generator(generator_train, epochs=3000, verbose=2, validation_data=generator_test)
```

Code for calculating various error functions.

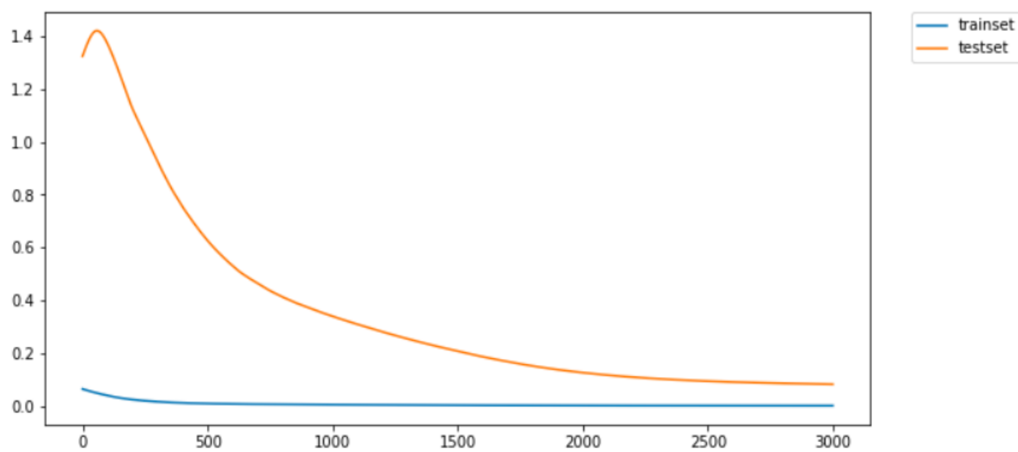
```
mean = df_result['Actual'].mean()
mae = (df_result['Actual'] - df_result['Prediction']).abs().mean()
mse = (df_result['Actual']*df_result['Actual'] - df_result['Prediction']*df_result['Prediction']).abs().mean()
rmse=math.sqrt(mse)
print("mean: ", mean)
print("mae:", mae)
print("mse:", mse)
print("rmse:", rmse)
```

Result

RNN Model

Figure 7. RNN loss function

```
losses = score.history['loss']
val_losses = score.history['val_loss']
plt.figure(figsize=(10,5))
plt.plot(losses, label="trainset")
plt.plot(val_losses, label="testset")
plt.legend(bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0.)
plt.show()
```




```
df_result['Diff'] = 100 * (df_result['Prediction'] - df_result['Actual']) / df_result['Actual']
```

```
df_result
```

	Actual	Prediction	Diff
0	71.524	90.550346	26.601346
1	72.597	92.477455	27.384679
2	73.670	97.061958	31.752353

```
mae:  40.64380407714844
mse:  7999.386914684693
rmse:  89.43929178322406
```

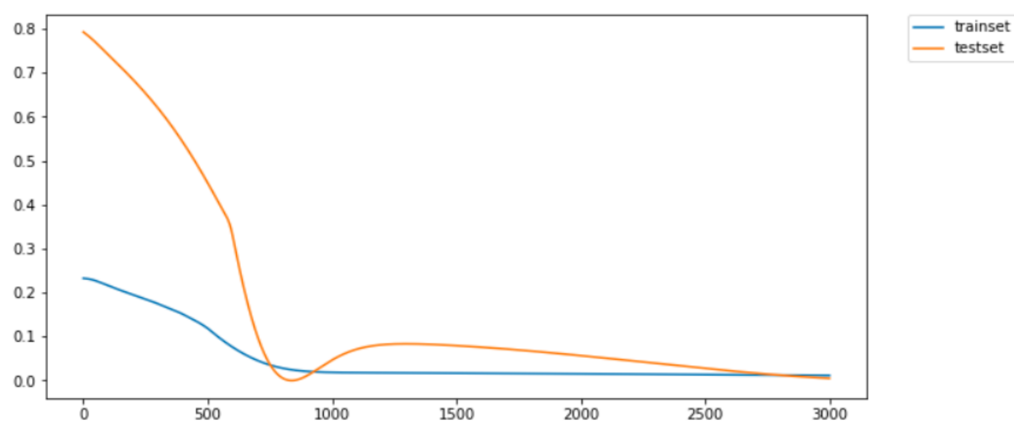
The RNN algorithm runs on the raw data of seventy years with three thousand epochs or iterations to achieve a proper training of model.

This demand prediction model using this algorithm of network gave a RMSE of 89.43, MSE of 7999.38 and MAE of 40.64.

LSTM Model

Figure 8. RNN loss function

```
losses = score.history['loss']
val_losses = score.history['val_loss']
plt.figure(figsize=(10,5))
plt.plot(losses, label="trainset")
plt.plot(val_losses, label="testset")
plt.legend(bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0.)
plt.show()
```




```
df_result['Diff'] = 100 * (df_result['Prediction'] - df_result['Actual']) / df_result['Actual']
```

```
df_result
```

	Actual	Prediction	Diff
0	71.524	77.630447	8.537620
1	72.597	77.298164	6.475701
2	73.670	78.032463	5.921628

```
mae: 1.401418212890628
mse: 219.73678141826895
rmse: 14.823521221972495
```

LSTM algorithm compiles on the raw data of seventy years with three thousand epochs or iterations to achieve a proper demand prediction.
This demand prediction model using this algorithm of network gave RMSE as 14.82, MSE as 219.73 and MAE as 1.40.

Conclusion & Future Work

For this study of wheat demand in the Indian context we considered three hyperparameters population, substitute cereal demand, and per capita income. Seventy years of data is collected for this purpose. The collected raw data is run upon two algorithms 1) LSTM and 2) RNN.
The results of both the algorithms are calculated and compared numerically.
LSTM gave an RMSE of 14.82 while RNN gave 89.43.
LSTM gave an MSE of 219.73 while RNN gave 7999.38.
LSTM gave an MAE of 1.40 while RNN gave 40.64.
Clearly, LSTM gives better results in comparison to the RNN model. The reason for the better results in the case of LSTM is they handle vanishing gradient problems efficiently and is an algorithm better in memory management.
Future prospect is still there in such studies to improve the results and accuracy, which can be achieved by collecting more raw data, our present study is on annual data it can be converted into monthly data to increase the raw data, which will improve the algorithm in learning and proper weight assigning and we can look for some approximation algorithms such as Iterated Local Search, Genetic Algo.[26] etc. for better resource optimization.

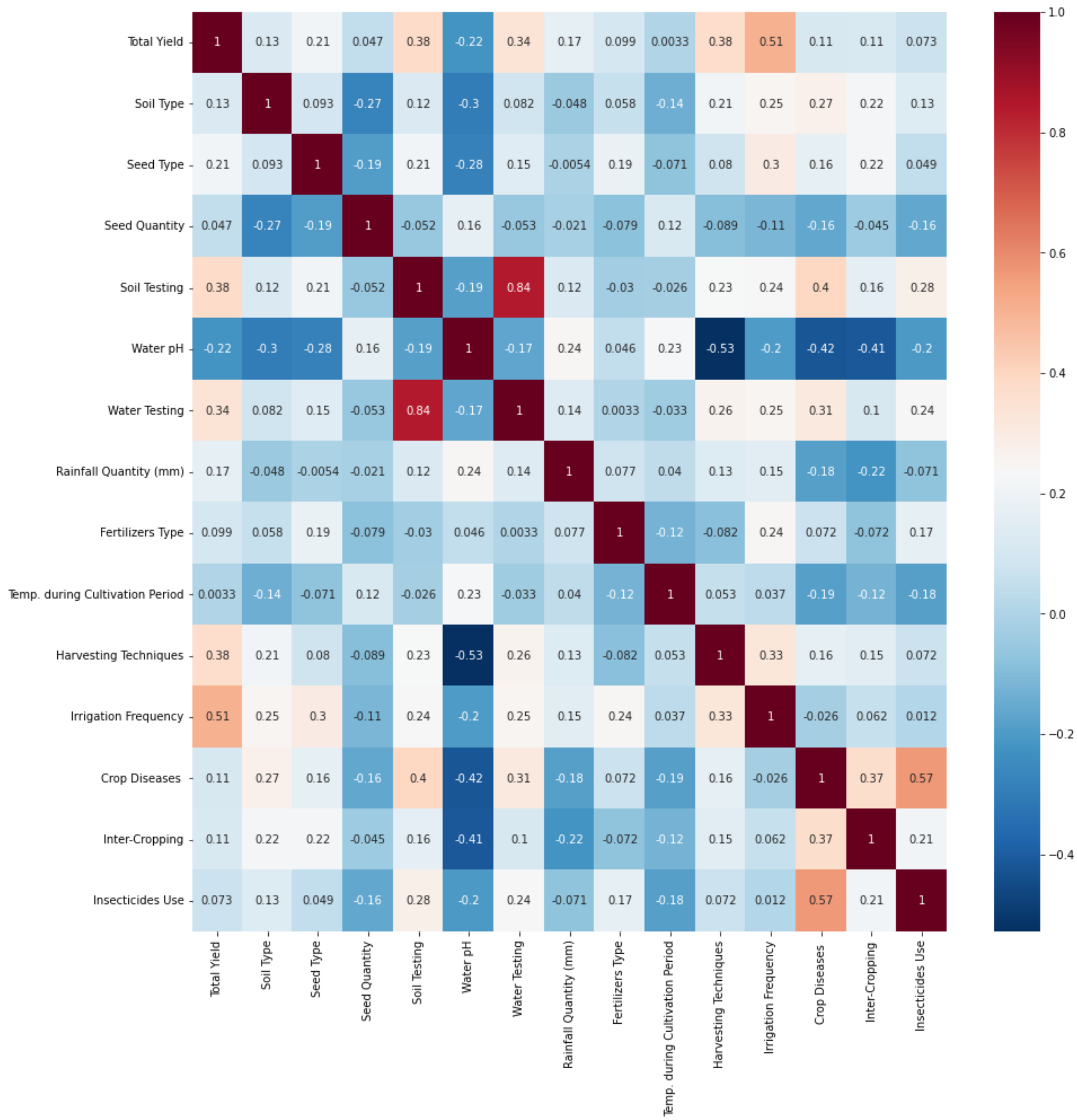
Total Crop Yield Prediction

Total Yield	Soil Type	Seed Type	Seed Quantity	Soil Testing	Water pH	Water Testing	Rainfall Quantity (mm)	Fertilizers Type	Fertilizers Type Quantity	Temp. during Cultivation Period	Harvesting Techniques	Irrigation Frequency	Crop Rotation Frequency	Crop Diseases	Inter-Cropping	Insecticides Use
1.71	1	1.0	0.90	0	7.63	0	3	0.0	85	45	0.0	0	3	0	1	0
2.50	1	1.0	1.33	0	7.63	0	12	0.0	1000	45	0.0	2	4	0	1	0
0.75	1	1.0	0.35	0	7.63	0	22	0.0	60	48	1.0	2	1	0	1	0

It is a prediction model created for total yield prediction using parameters that are both categorical and numerical. Some of the various parameters used in this study were soil type, seed type, seed quantity, fertilizer type, soil testing, water pH, water testing, rainfall in mm, temperature, irrigation frequency, harvesting technique, an insecticide used, intercropping, and crop disease.

Process followed:-

First, the important libraries are called and the required files are imported. The cleaning of data files is done. In the present case we are dealing with a categorical type of data like yes and no; organic and inorganic; Domat and Balu; cutting techniques, etc. Then the data is scaled using a MinMax Scaler to bring it in the range between 0 and 1. Now the data is divided into parts like testing portion and training portion. We used 200 data for training the network and 60 data for testing the network. The data is passed to Hstack for time series data generation. As the data is prepared using the above steps it is passed to RNN and LSTM models with 3000 iterations. Followed by it, an inverse transform is taken to get the value in its original form. Finally, actual and predicted values are plotted using the Matplotlib library. Now the actual and predicted values are used for comparison using MSE, MAE, and RMSE as the metric. These metrics are covered above in detail.

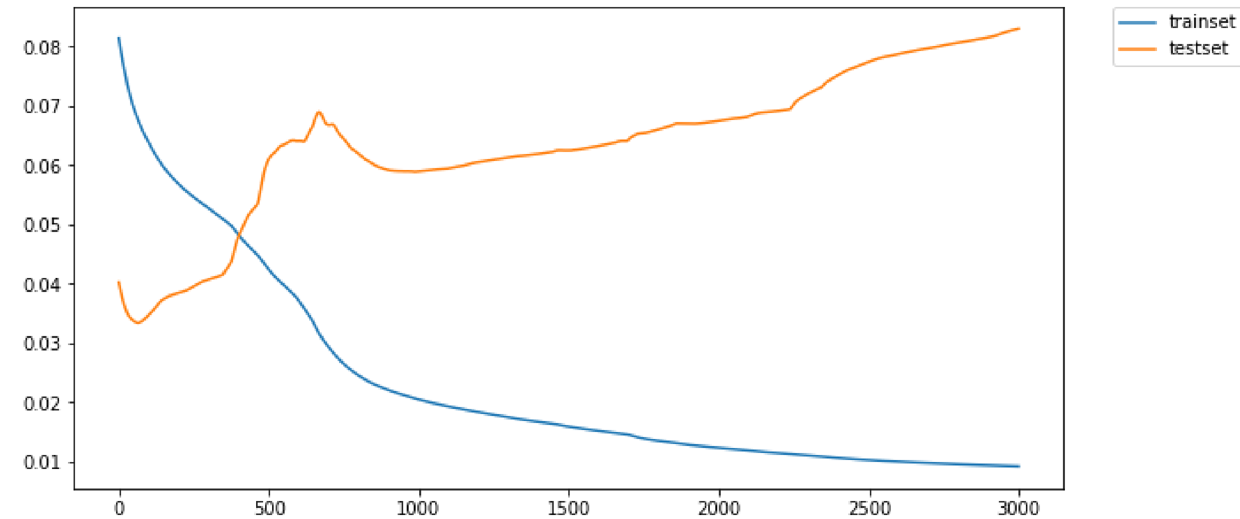


Heat Map of Total Yield

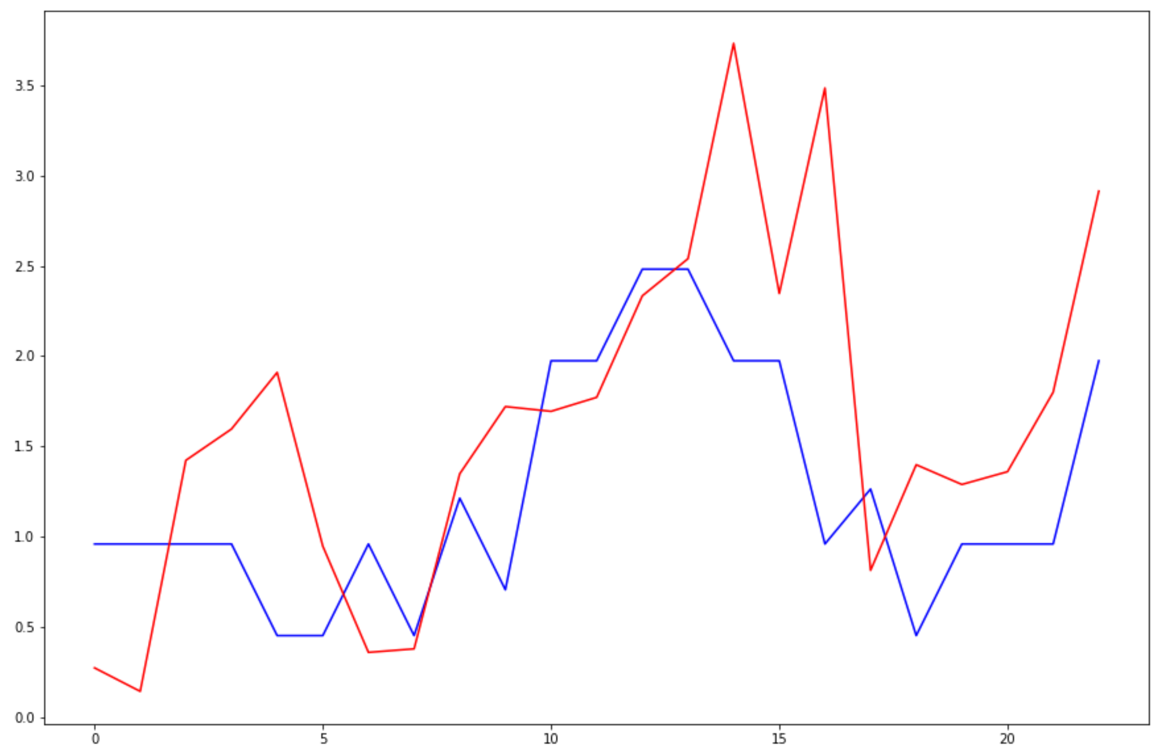
From the given diagram we can clearly see the total yield correlation with various other parameters used in this study. Yield is strongly correlated to irrigation frequency which is an important factor for any agricultural crop.

Results:-

RNN Algorithm



The error function of RNN after achieving 3000 epochs(Iteration)

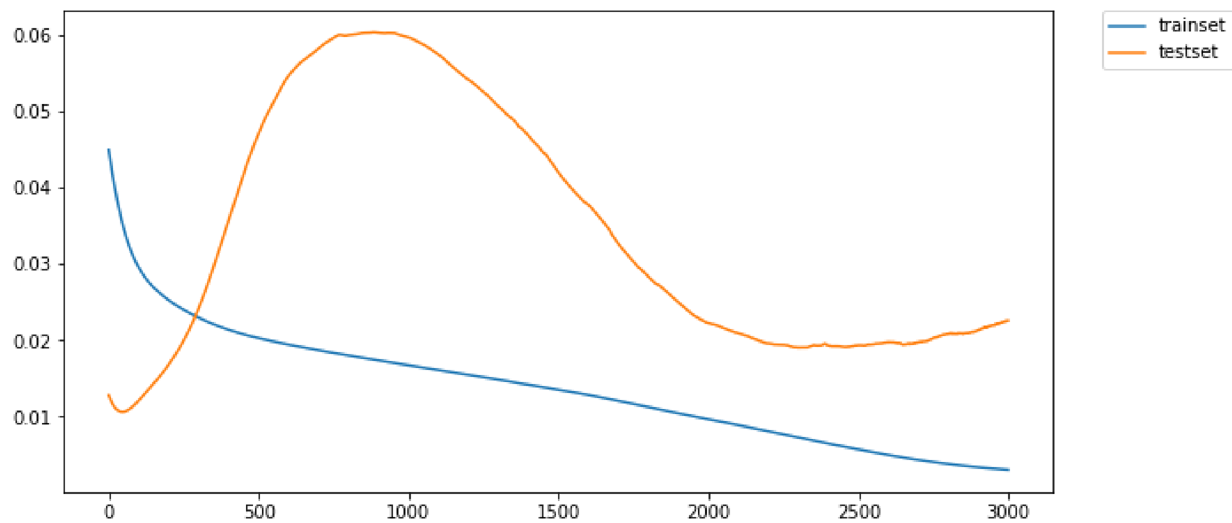


Actual vs. Predicted output plot. [Red:- Predicted ; Blue:- Actual]

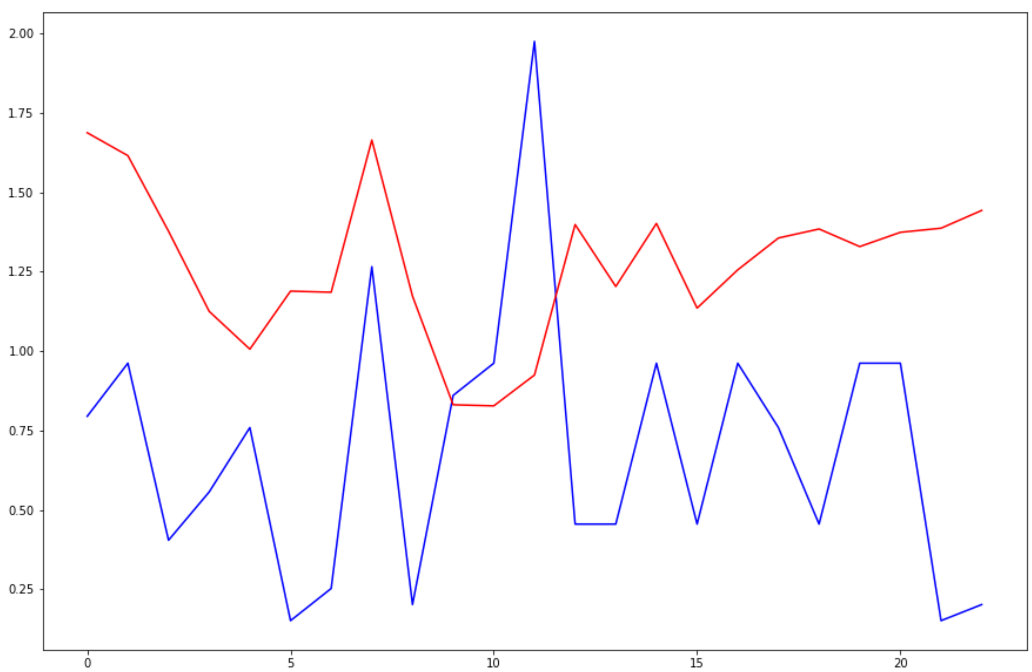
```
mean:  1.2394056136488714
mae: 0.678192916189371
mse: 2.1364392223388746
rmse: 1.4616563283955892
```

These are the metrics on which we judge the RNN algorithm.

LSTM Algorithm



The error function of LSTM after achieving 3000 epochs(Iteration)



This is the Actual vs. Predicted output plot. [Red:- Predicted ; Blue:- Actual]

```
mean:  1.2394056136488714
mae: 0.6630786299525083
mse: 1.6720646567932744
rmse: 1.2930833912757809
```

These are the metrics on which we judge the LSTM algorithm.

Conclusion

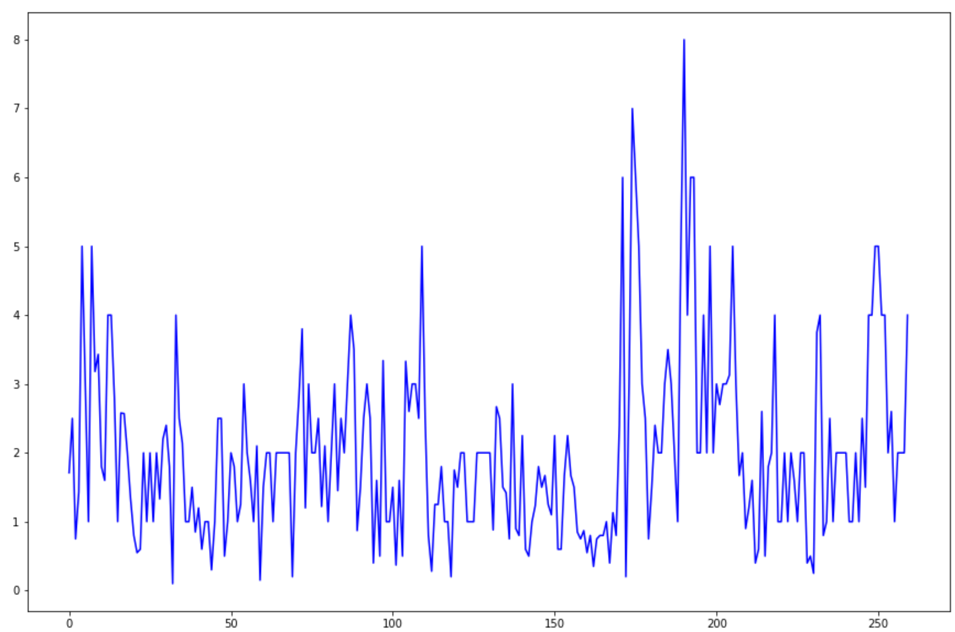
The results of both the algorithms are calculated and compared numerically.

LSTM gave an RMSE of 1.29 while RNN gave 1.46.

LSTM gave an MSE of 1.67 while RNN gave 2.13.

LSTM gave an MAE of 0.66 while RNN gave 0.67.

Clearly, LSTM gives better results in comparison to the RNN model.



Total Yield vs Data points graph.

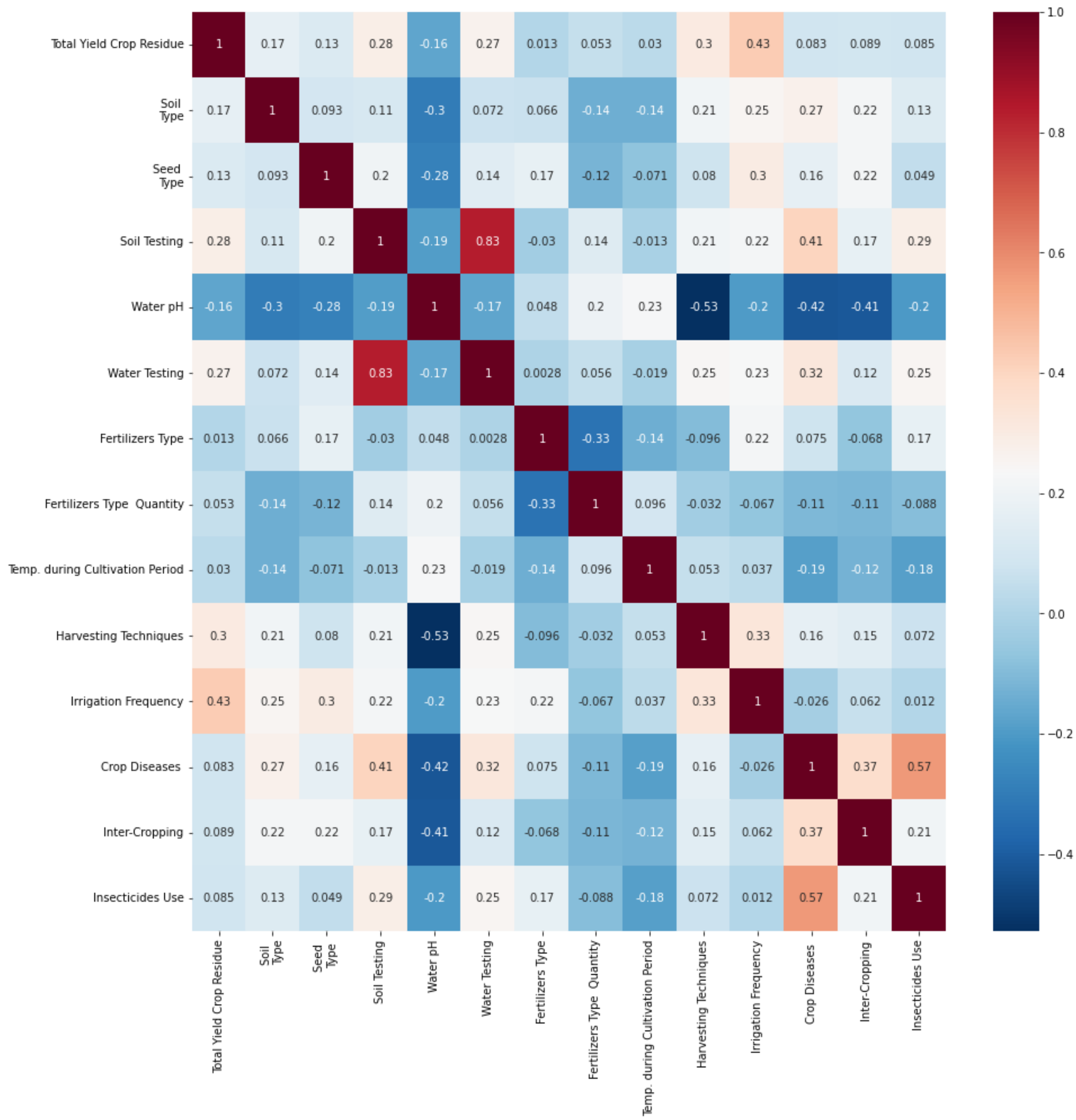
Total Yield Crop Residue Prediction

Total Yield Crop Residue	Soil \nType	Seed \nType	Soil Testing	Water pH	Water Testing	Rainfall Quantity (mm)	Fertilizers Type	Fertilizers Type Quantity	Temp. during Cultivation Period	Harvesting Techniques	Irrigation Frequency	Crop Rotation Frequency	Crop Diseases	Inter-Cropping	Insecticides Use
3.42	1	1.0	0	7.63	0	3	0.0	85	45	0.0	0	3	0	1	0
5.67	1	1.0	0	7.63	0	12	0.0	1000	45	0.0	2	4	0	1	0
5.00	1	1.0	0	7.63	0	22	0.0	60	48	1.0	2	1	0	1	0

It is a prediction model created for total crop residue prediction using parameters that are both categorical and numerical. Some of the various parameters used in this study were soil type, seed type, fertilizer type, soil testing, water pH, water testing, rainfall in mm, temperature, irrigation frequency, harvesting technique, an insecticide used, intercropping, and crop disease.

Process followed:-

First, the important libraries are called and the required files are imported. The cleaning of data files is done. In the present case, we are dealing with a categorical type of data like yes and no; organic and inorganic; Domat and Balu; cutting techniques, etc. Then the data is scaled using a MinMax Scaler to bring it in the range between 0 and 1. Now the data is divided into testing portion and training portion. We used 200 data for training the network and 60 data for testing the network. The data is passed to Hstack for time series data generation. As the data is prepared using the above steps it is passed to RNN and LSTM models with 3000 iterations. Followed by it, an inverse transform is taken to get the value in its original form. Finally, actual and predicted values are plotted using the Matplotlib library. Now the actual and predicted values are used for comparison using MSE, MAE, and RMSE as the metric. These metrics are covered above in detail.

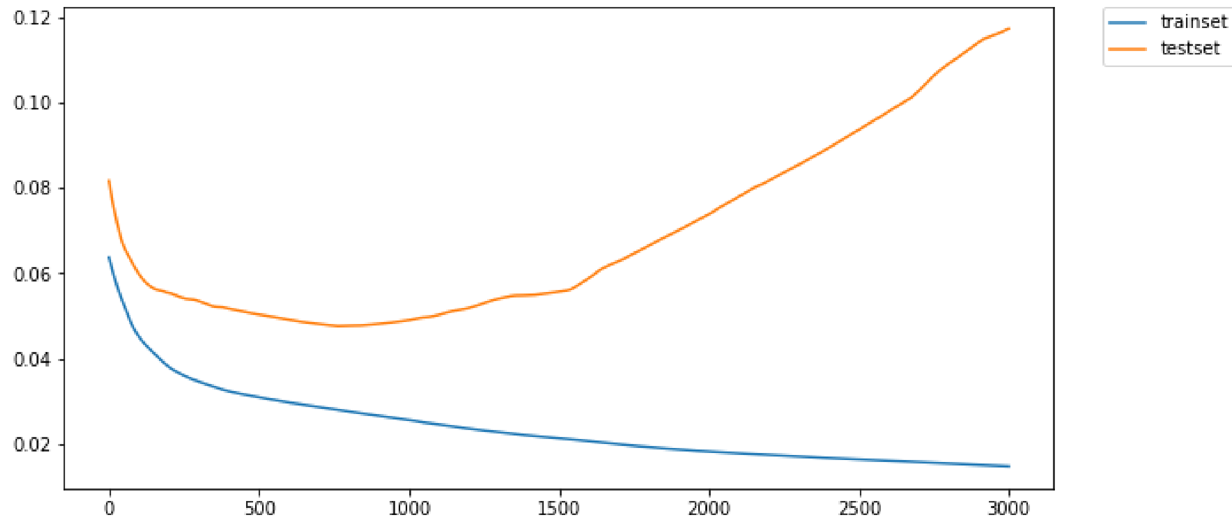


Heat Map of Total Yield Crop Residue

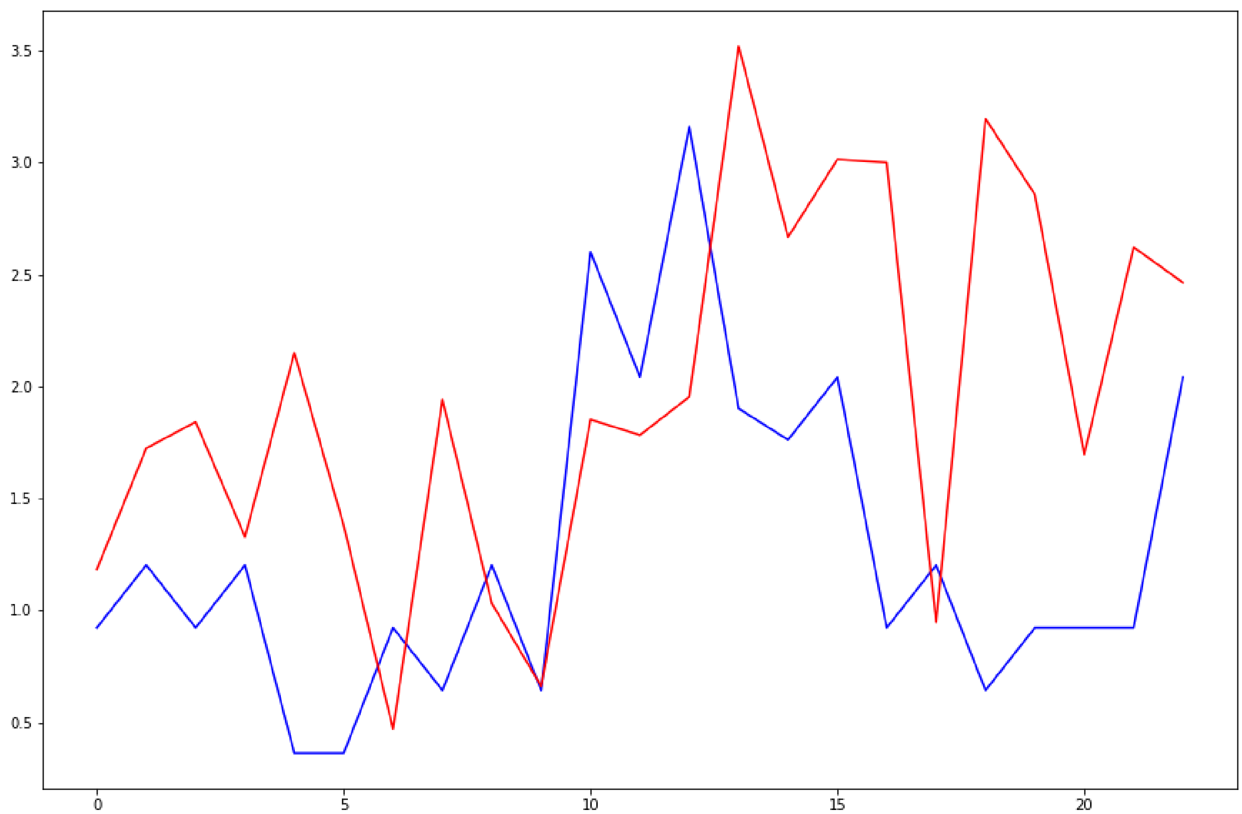
From the given diagram we can clearly see the total yield crop residue correlation with various other parameters used in this study. Yield is strongly correlated to irrigation frequency which is an important factor for any agricultural crop.

Results:-

RNN algorithm



The error function of RNN after achieving 3000 epochs(Iteration)

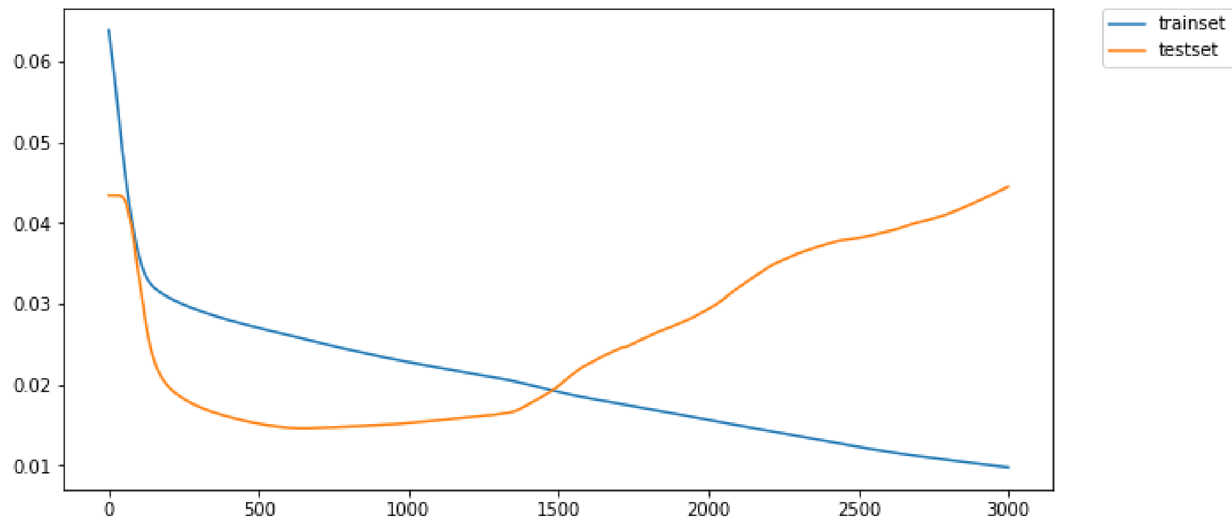


This is the Actual vs. Predicted output plot. [Red:- Predicted ; Blue:- Actual]

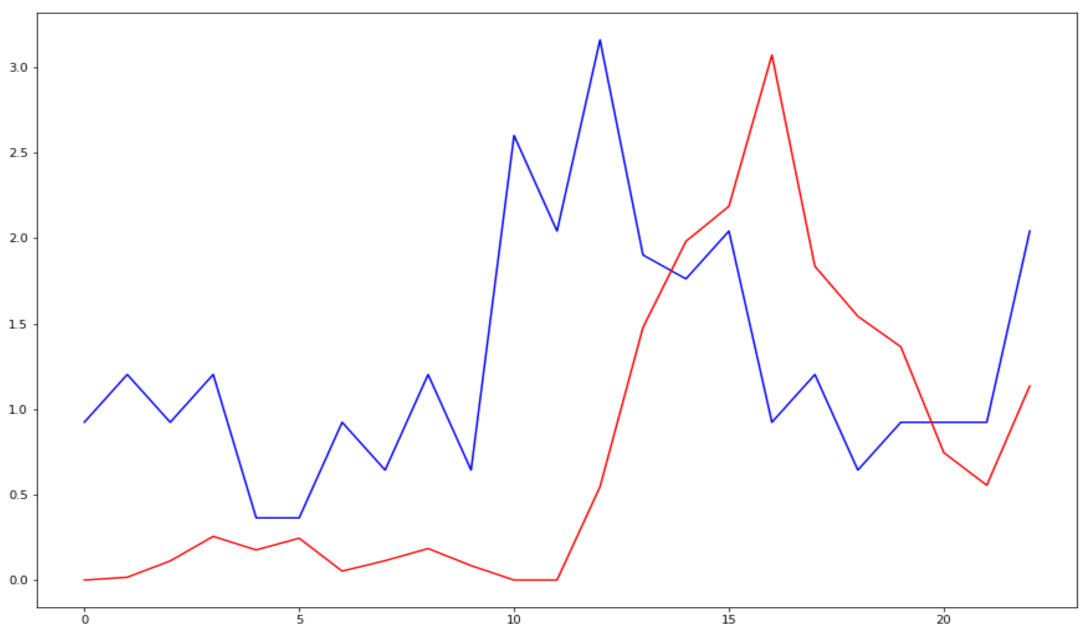
```
mean:  1.2818485861964122
mae: 0.9563310150137494
mse: 3.4592746373652754
rmse: 1.8599125348696577
```

These are the metrics on which we judge the RNN algorithm.

LSTM Algorithm



The error function of LSTM after achieving 3000 epochs(Iteration).



This is the Actual vs. Predicted output plot. [Red:- Predicted ; Blue:- Actual]

```
mean:  1.2818485861964122
mae: 0.903375605278127
mse: 2.1076202396723205
rmse: 1.451764526248083
```

These are the metrics on which we judge the LSTM algorithm.

Conclusion

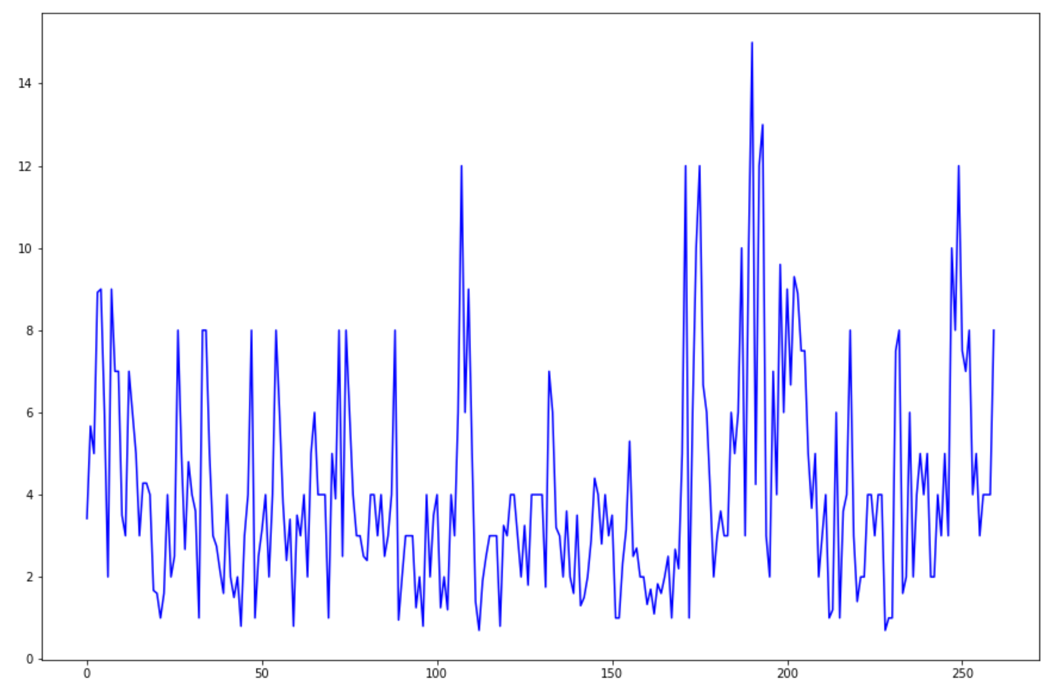
The results of both the algorithms are calculated and compared numerically.

LSTM gave an RMSE of 1.45 while RNN gave 1.85.

LSTM gave an MSE of 2.107 while RNN gave 3.45.

LSTM gave an MAE of 0.9 while RNN gave 0.95.

Clearly, LSTM gives better results in comparison to the RNN model.



Total Yield crop Residue vs Data points graph.

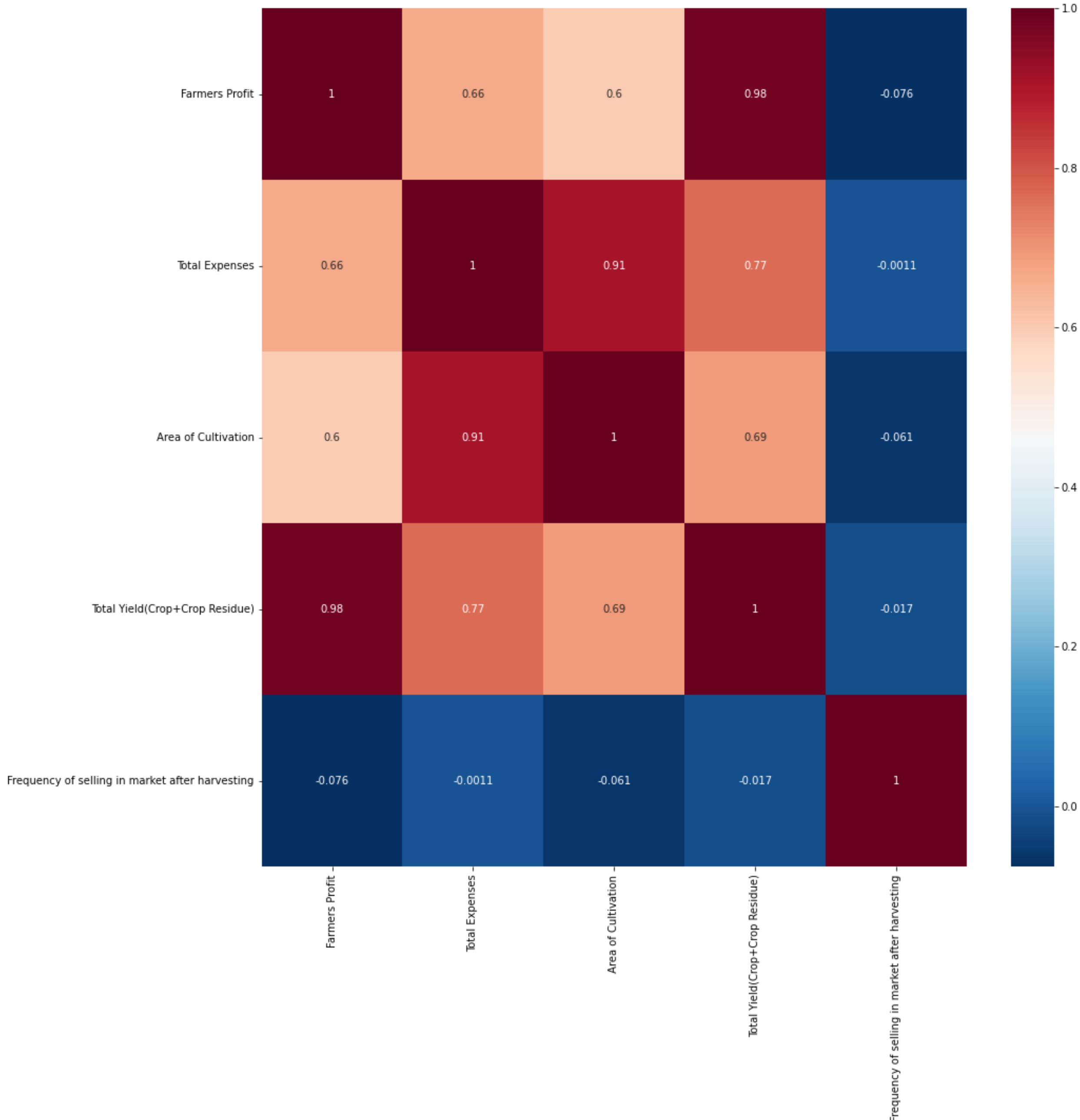
Farmers Profit Prediction

Farmers Profit	Total Expenses	Area of Cultivation	Total Yield(Crop+Crop Residue)	Frequency of selling in market after harvesting
240990.0	64245.0	35.0	180	1
91995.0	18300.0	6.0	49	0
196000.0	34000.0	20.0	115	1
597950.0	54100.0	28.0	290	0
167500.0	14500.0	5.0	70	1

It is a prediction model created for farmers profit prediction using parameters that are numerical. Some of the various parameters used in this study were total expense, area of cultivation, frequency of selling and Total yield.

Process followed:-

First, the important libraries are called and the required files are imported. The cleaning of data files is done. In the present case, we are dealing with a numerical type of data so any ambiguity is removed using Numpy. Then the data is scaled using a MinMax Scaler to bring it in the range between 0 and 1. Now the data is divided into testing portion and training portion. We used 200 data for training the network and 60 data for testing the network. The data is passed to Hstack for time series data generation. As the data is prepared using the above steps it is passed to RNN and LSTM models with 1000 iterations. Followed by it, an inverse transform is taken to get the value in its original form. Finally, actual and predicted values are plotted using the Matplotlib library. Now the actual and predicted values are used for comparison using MSE, MAE, and RMSE as the metric. These metrics are covered above in detail.

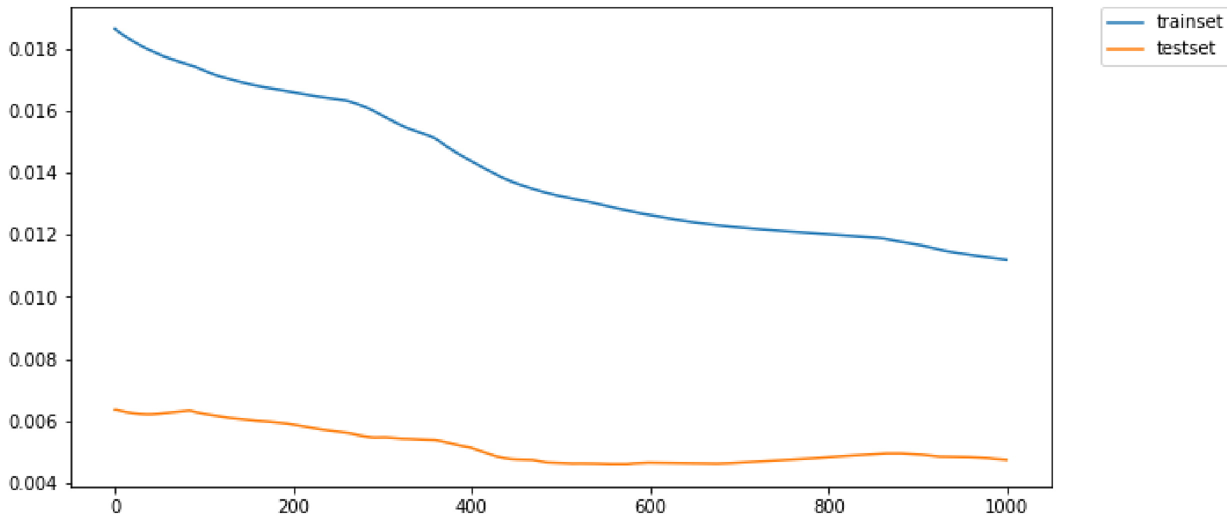


Heat Map of Total Farmers Profit

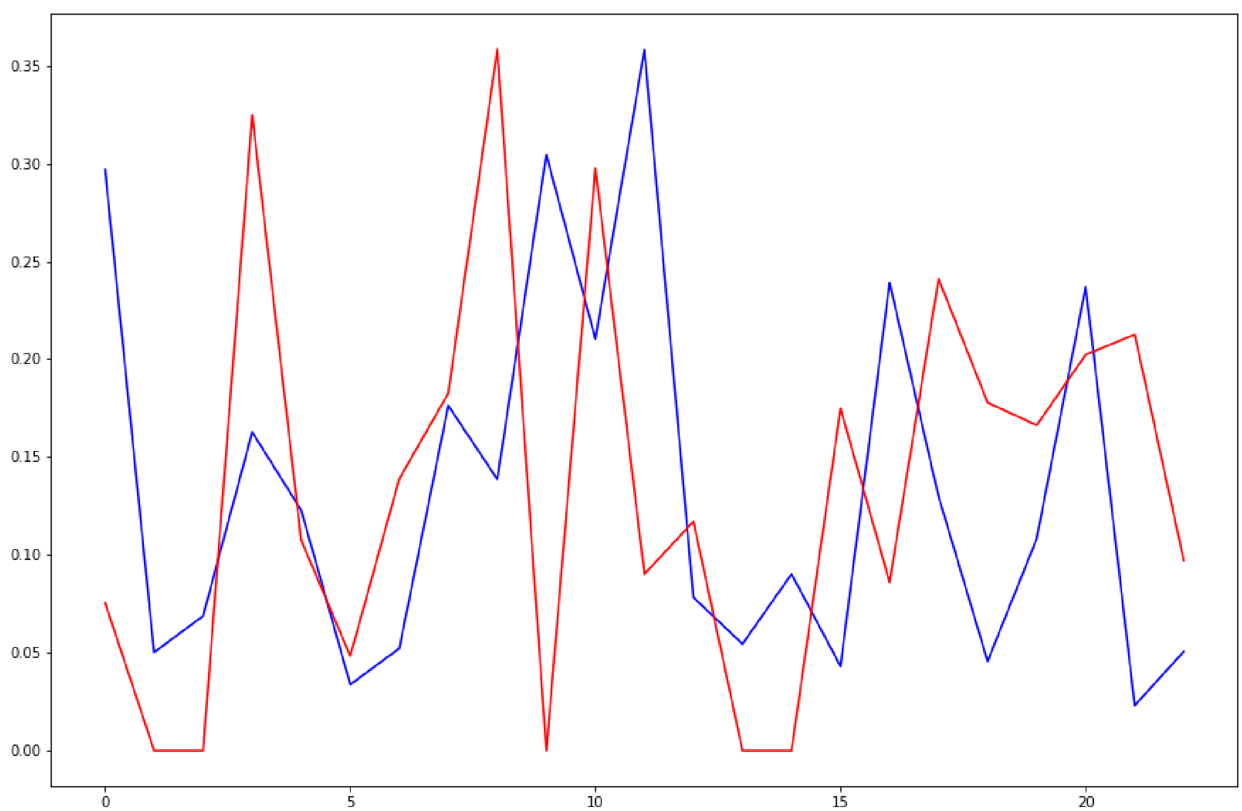
From the given diagram, we can clearly see the total profit correlation with various other parameters used in this study. Profit is strongly correlated to Total Yield, which is an important factor for any agricultural business.

Results

RNN Algorithm



The error function of RNN after achieving 1000 epochs(Iteration).

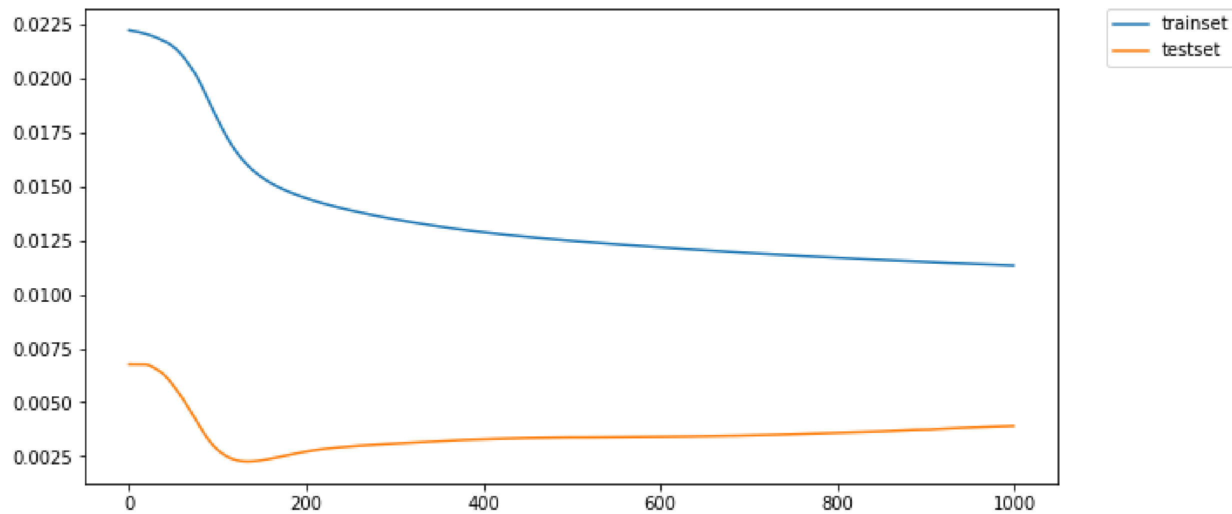


This is the Actual vs. Predicted output plot. [Red:- Predicted ; Blue:- Actual]

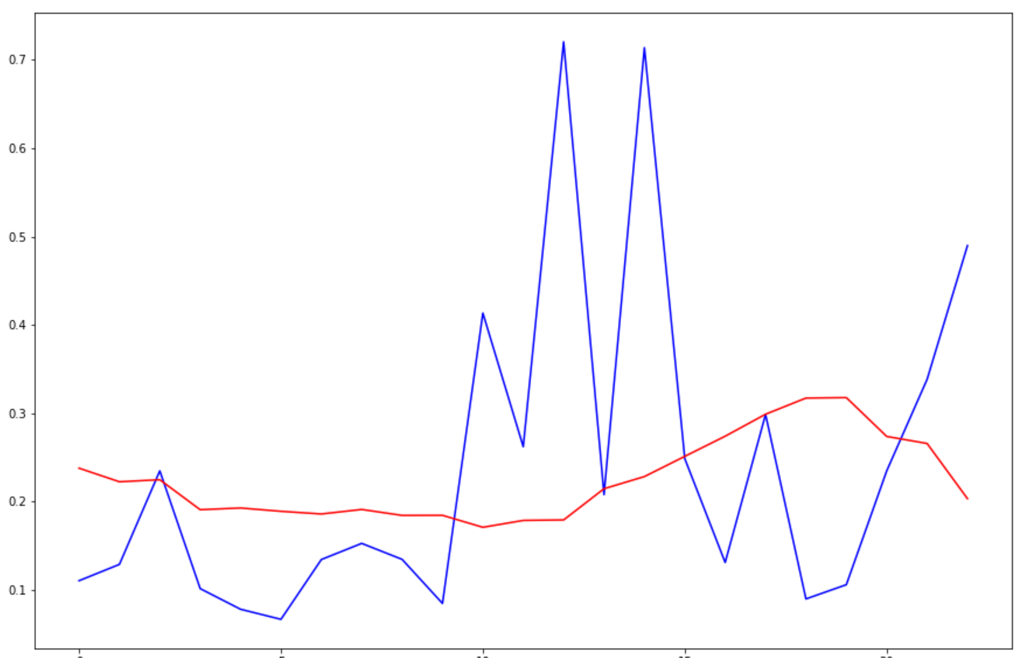
```
mean:  0.2381830261108821
mae: 0.14416770281045446
mse: 0.0837514788187446
rmse: 0.2893984775681182
```

These are the metrics on which we judge the RNN algorithm.

LSTM Algorithm



The error function of LSTM after achieving 1000 epochs(Iteration).



This is the Actual vs. Predicted output plot. [Red:- Predicted ; Blue:- Actual]

```
mean:  0.2381830261108821
mae: 0.13652247577886747
mse: 0.08139131591785116
rmse: 0.28529163310172834
```

These are the metrics on which we judge the LSTM algorithm.

Conclusion

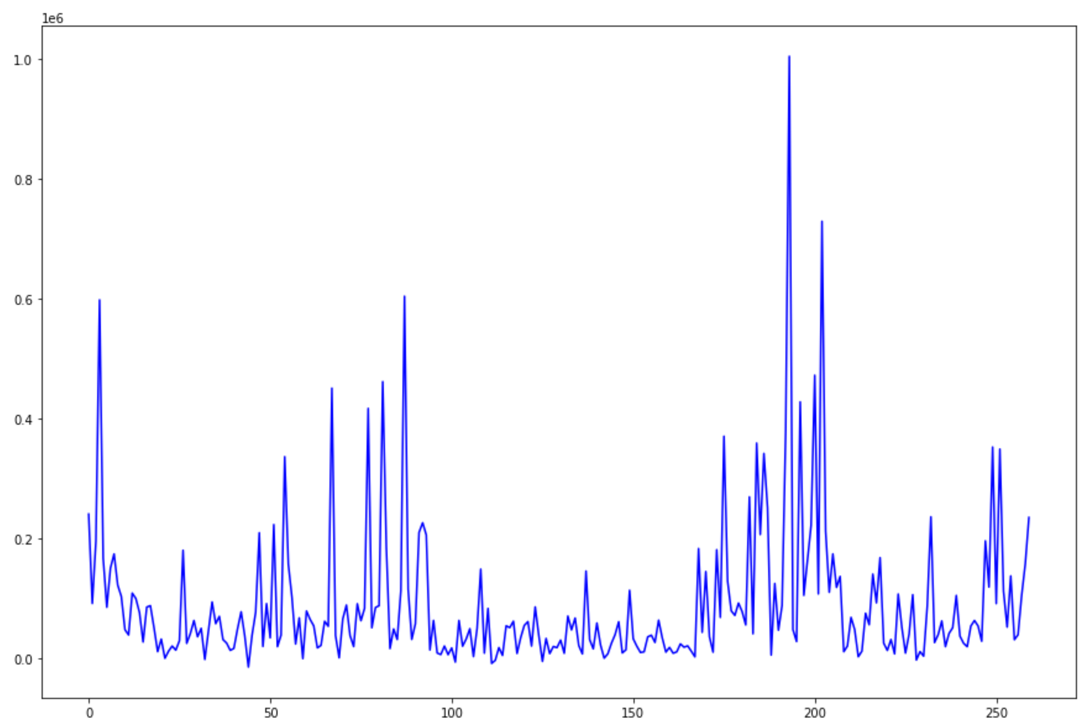
The results of both the algorithms are calculated and compared numerically.

LSTM gave an RMSE of 0.285 while RNN gave 0.289.

LSTM gave an MSE of 0.0813 while RNN gave 0.0837.

LSTM gave an MAE of 0.13 while RNN gave 0.144.

Clearly, LSTM gives better results in comparison to the RNN model.



Farmers profit (10^6) vs Data points graph.

Reference

1. Paidipati, K. K., & Banik, A. (2020). Forecasting of rice cultivation in India–a comparative analysis with ARIMA and LSTM-NN models. *EAI Endorsed Transactions on Scalable Information Systems*, *7*(24).
2. (2022). Retrieved 18 April 2022, from https://farmer.gov.in/m_cropstaticswheat.aspx
3. info@sustainablefoodtrust.org, S. (2022). Sustainable Food Trust. Retrieved 18 April 2022, from https://sustainablefoodtrust.org/articles/a-brief-history-of-wheat/
4. Vij, S. (2022). Rice and wheat maps of India: Rajasthan doesn't eat rice, rotis a rarity in Manipur. Retrieved 18 April 2022, from https://scroll.in/article/670473/rice-and-wheat-maps-of-india-rajasthan-doesnt-eat-rice-rotis-a-rarity-in-manipur#:~:text=On%20average%2C%20a%20rural%20Indian,to%204%20kgs%20of%20wheat.
5. Abbasimehr, H., Shabani, M., & Yousefi, M. (2020). An optimized model using LSTM network for demand forecasting. *Computers & Industrial Engineering*, *143*, 106435. https://doi.org/10.1016/j.cie.2020.106435
6. Bai, L.-H., & Xu, H. (2022). Accurate storm surge forecasting using the encoder–decoder long short term memory recurrent neural network. *Physics of Fluids*, *34*(1), 016601. https://doi.org/10.1063/5.0081858
7. Long short-term memory - Wikipedia. (2022). Retrieved 2 May 2022, from https://en.wikipedia.org/wiki/Long_short-term_memory
8. Karim, M. R., Awal, M. A., & Akter, M. (1970). Forecasting of wheat production in Bangladesh. *Bangladesh Journal of Agricultural Research*, *35*(1), 17–28. https://doi.org/10.3329/bjar.v35i1.5863
9. Muthusinghe, M. R. S., S.T., P., Weerakkody, W. A. N. D., Saranga, A. M. H., & Rankothge, W. H. (2018). Towards smart farming: Accurate prediction of Paddy Harvest and rice demand. *2018 IEEE Region 10 Humanitarian Technology Conference (R10-HTC)*. https://doi.org/10.1109/r10-htc.2018.8629843
10. Purohit, S. K., Panigrahi, S., Sethy, P. K., & Behera, S. K. (2021). Time series forecasting of price of agricultural products using hybrid methods. *Applied Artificial Intelligence*, 1–19. https://doi.org/10.1080/08839514.2021.1981659
11. Sharma, S., & Patil, S. V. (2015). Key indicators of rice production and consumption, correlation between them and supply-demand prediction. *International Journal of Productivity and Performance Management*, *64*(8), 1113–1137. https://doi.org/10.1108/ijppm-06-2014-0088
12. Gandhi, N., Armstrong, L., Petkar, O., & Tripathy, A. (2016). Rice crop yield prediction in India using support vector machines. *2016 13Th International Joint Conference On Computer Science And Software Engineering (JCSSE)*. doi: 10.1109/jcsse.2016.7748856
13. Chandriah, K., & Naraganahalli, R. (2021). RNN / LSTM with modified Adam optimizer in deep learning approach for automobile spare parts demand forecasting. Multimedia Tools And Applications. doi: 10.1007/s11042-021-10913-0

14. “FAO.org.” India at a Glance — FAO in India — Food and Agriculture
    a. Organization of the United Nations.
15. Balaji Prabhu B.V., & Dakshayini, M. (2020). An Effective Multiple Linear Regression-Based Forecasting Model for Demand-Based Constructive Farming. *International Journal Of Web-Based Learning And Teaching Technologies*, *15*(2), 1-18. doi: 10.4018/ijwltt.2020040101
16. Vanichrujee, U., Horanont, T., Pattara-atikom, W., Theeramunkong, T., & Shinozaki, T. (2018). Taxi Demand Prediction using Ensemble Model Based on RNNs and XGBOOST. *2018 International Conference On Embedded Systems And Intelligent Technology &Amp; International Conference On Information And Communication Technology For Embedded Systems (ICESIT-ICICTES)*. doi: 10.1109/icesit-icictes.2018.8442063
17. Bali, N., & Singla, A. (2021). Deep Learning Based Wheat Crop Yield Prediction Model in Punjab Region of North India. Applied Artificial Intelligence, 35(15), 1304-1328. doi: 10.1080/08839514.2021.1976091
18. Haider, S., Naqvi, S., Akram, T., Umar, G., Shahzad, A., & Sial, M. et al. (2019). LSTM Neural Network Based Forecasting Model for Wheat Production in Pakistan. Agronomy, 9(2), 72. doi: 10.3390/agronomy9020072
19. A Guide to RNN: Understanding Recurrent Neural Networks and LSTM Networks. (2022). Retrieved 18 April 2022, from https://builtin.com/data-science/recurrent-neural-networks-and-lstm
20. Devi, M., Kumar, J., Malik, D., & Mishra, P. (2021). Forecasting of wheat production in Haryana using hybrid time series model. *Journal Of Agriculture And Food Research*, *5*, 100175. doi: 10.1016/j.jafr.2021.100175
21. Amin, M., Amanullah, M., & Akbar, A. (2014). Time series modeling for forecasting wheat production of Pakistan. JAPS: Journal of Animal & Plant Sciences, 24(5).
22. Kani, S., & Ershad, N. (2007). Annual Electricity Demand Prediction for Iranian Agriculture Sector Using ANN and PSO. 2007 IEEE Canada Electrical Power Conference. doi: 10.1109/epc.2007.4520373
23. Vijai, P., & Bagavathi Sivakumar, P. (2018). Performance comparison of techniques for water demand forecasting. *Procedia Computer Science*, *143*, 258-266. doi: 10.1016/j.procs.2018.10.394
24. Bhojani, S., & Bhatt, N. (2020). Wheat crop yield prediction using new activation functions in neural network. *Neural Computing And Applications*, *32*(17), 13941-13951. doi: 10.1007/s00521-020-04797-8
25. Abbasimehr, H., Shabani, M., & Yousefi, M. (2020). An optimized model using LSTM network for demand forecasting. *Computers &Amp; Industrial Engineering*, *143*, 106435. doi: 10.1016/j.cie.2020.106435
26. "Recurrent neural network", En.wikipedia.org, 2018. [Online]. Available: https://en.wikipedia.org/wiki/Recurrent_neural_network.
27. A. Ishigaki and S. Takaki, “Iterated Local Search Algorithm for Flexible Job Shop Scheduling”, 2017
28. Bedi, J., & Toshniwal, D. (2018). Empirical Mode Decomposition Based Deep Learning for Electricity Demand Forecasting. *IEEE Access*, *6*, 49144-49156. doi: 10.1109/access.2018.2867681

29. Abdel-Nasser, M., & Mahmoud, K. (2017). Accurate photovoltaic power forecasting models using deep LSTM-RNN. *Neural Computing And Applications*, *31*(7), 2727-2740. doi: 10.1007/s00521-017-3225-z
30. Kumar, D., Mathur, H., Bhanot, S., & Bansal, R. (2020). Forecasting of solar and wind power using LSTM RNN for load frequency control in isolated microgrid. *International Journal Of Modelling And Simulation*, *41*(4), 311-323. doi: 10.1080/02286203.2020.1767840
31. Gamage, A., & Kasthurirathna, D. (2019). Agro-Genius: Crop Prediction Using Machine Learning. *International Journal of Innovative Science and Research Technology*, *4*(10).
32. Márquez, J. P., de Oliveira Ribeiro, C., Santoyo, E. R., & Fernández, V. F. (2021). Ethanol Fuel Demand Forecasting in Brazil Using a LSTM Recurrent Neural Network Approach. *IEEE Latin America Transactions*, *19*(4), 551-558.
33. Paidipati, K. K., & Banik, A. (2020). Forecasting of rice cultivation in India–a comparative analysis with ARIMA and LSTM-NN models. *EAI Endorsed Transactions on Scalable Information Systems*, *7*(24).
34. Saini, U., Kumar, R., Jain, V., & Krishnajith, M. U. (2020, July). Univariant time series forecasting of agriculture load by using lstm and gru rnns. In *2020 IEEE Students Conference on Engineering & Systems (SCES)* (pp. 1-6). IEEE.
35. Marndi, A., Ramesh, K. V., & Patra, G. K. (2021). Crop production estimation using deep learning technique. *Current Science*, *121*(8), 1073-1079.
36. Rani, S. J., & Babu, N. C. (2020). Forecasting production of rice in India–using Arima and deep learning methods. *Int J Math Trends Technol (IJMTT)*, *66*(4).
37. Geetha, M., Suganthe, R. C., Latha, R. S., Anju, R., Sastimalar, K., & Shobana, P. (2022, January). Deep Learning Based Yield Prediction Model To Predict The Yield of Paddy In Cauvery Delta Region. In *2022 International Conference on Computer Communication and Informatics (ICCCI)* (pp. 1-6). IEEE.
38. Majhi, B., Naidu, D., Mishra, A. P., & Satapathy, S. C. (2020). Improved prediction of daily pan evaporation using Deep-LSTM model. *Neural Computing and Applications*, *32*(12), 7823-7838.
39. Bhartra, D., Manohar, A. M., Vohra, D., Bharadwaj, S., & Srinivas, K. S. (2021, November). Agricultural Yield Estimation of Various Crops in Southern India Using Vegetation Index. In *2021 8th International Conference on Soft Computing & Machine Intelligence (ISCMI)* (pp. 96-100). IEEE.
40. Reddy, D. J., & Kumar, M. R. (2021, May). Crop yield prediction using machine learning algorithm. In *2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS)* (pp. 1466-1470). IEEE.
41. Kowshik, A., Kishor Gowda, H. K., Rithik Somesh, B. R., Yashas, S., Ramesh, B., & Nithyashree, R. Crop yield prediction based on Indian agriculture using machine learning.