Forecasting sales of store using IBM watson studio

1 INTRODUCTION

1.1 OVERVIEW

Forecasting of sales is an essential task for the management of a store. It is very important to be able to predict the sales of the store so as to be prepared with the inventory that they will need. So we are building a system that analyses the previous trends of sales which includes sales on various days and predicts future sales. This will be beneficial for both the store owners and customers as their will be sufficient supplies at all times.

1.2 PROPOSED SYSTEM

In this project, we are building a system that analyses the previous trends of sales which includes sales on various days and predicts future sales. The goal of this project is to forecast the sales of stores by using time series analysis. Here time series analysis algorithms such as RNN (Recurrent Neural Network) & LSTM (Long Term Short Memory) are used to analyze the past trends of sales of stores. Create and deploy flask-based web Application and integrate AI model to it.
The objective of the project is to build a web application where the user gives the last ten days' sales values and gets the prediction for the 11 th day which is showcased on UI.

2 LITERATURE SURVEY

2.1 EXISTING PROBLEM

With the existing system it is difficult to manually calculate the sales of the store and also to predict the number of commodities that will be sold in the coming 10 days. By predicting the same, the shop owners are able to understand the stock of the product and how much will be sold.

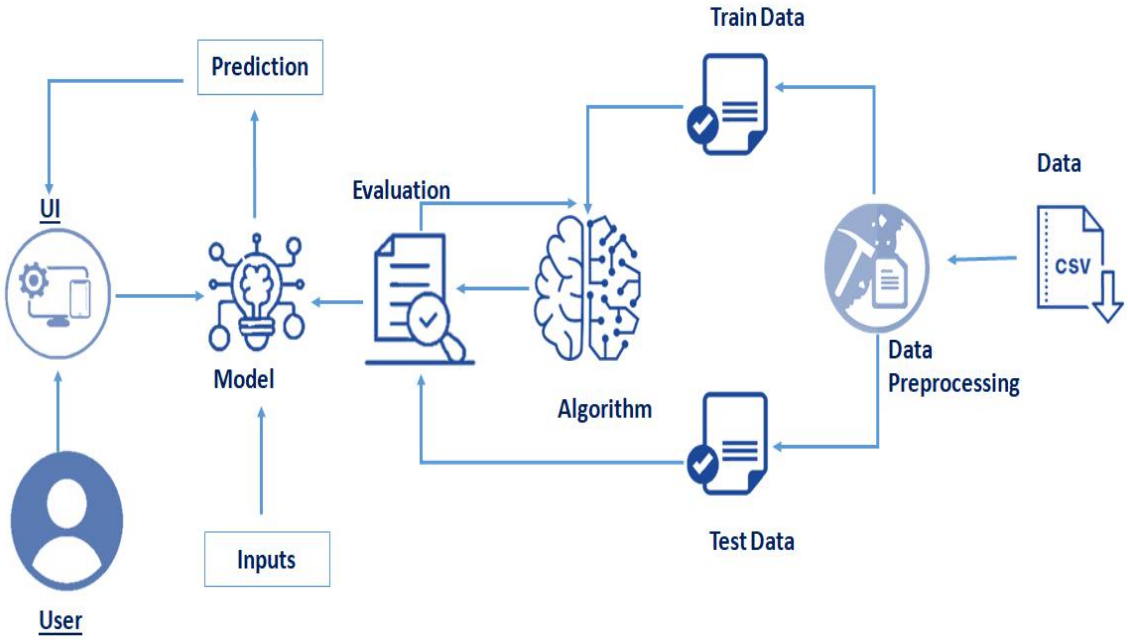
2.2 PROPOSED SYSTEM

In this project, we are building a system that analyses the previous trends of sales which includes sales on various days and predicts future sales. The goal of this project is to forecast the sales of stores by using time series analysis. Here time series analysis algorithms such as RNN (Recurrent Neural Network) & LSTM (Long Term Short Memory) are used to analyze the past trends of sales of stores. Create and deploy flask-based web Application and integrate AI model to it.

The objective of the project is to build a web application where the user gives the last ten days' sales values and gets the prediction for the 11 th day which is showcased on UI.

3 THEORITICAL ANALYSIS

3.1 BLOCK DIAGRAM



3.2 HARDWARE/SOFTWARE DESIGNING

Software requirements:

- Ananconda navigator
- Python packages
- IBM watson studio

4 EXPERIMENTAL INVESTIGATION

- Shape of inputs plays a major role in the correctness of the model.
- IBM Cloud helps to deploy machine learning models and test the correctness of our model.
- Integrating Flask with the machine learning model involves a lot of data preprocessing to make the predictions correctly.

5 FLOWCHART

- User interacts with the UI (User Interface) to enter the data of the previous 10 days to get the future prediction

- Entered data is analyzed by the model which is integrated
- Once the model analyses the input the prediction of the next day is showcased on the UI

To accomplish this, we have to complete all the activities and tasks listed below

Data Collection.

- Collect the dataset or Create the dataset

Data Preprocessing.

- Import the Libraries.
- Importing the dataset.
- Analyze the data
- Taking care of Missing Data
- Feature Scaling
- Data Visualization
- Splitting Data into Train and Test.
- Creating a dataset with a sliding window.

Model Building

- Import the model building Libraries
- Initializing the model
- Adding LSTM Layers
- Adding Output Layer
- Configure the Learning Process
- Training the model
- Model Evaluation
- Save the Model
- Test the Model

Application Building

- Create an HTML file
- Build Python Code

6 RESULT

SOURCE CODE:

5789 lines (5789 sloc) | 589 KB

Raw Blame

In [114]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import statsmodels.api as sm
```

In [117]:
```
dataset = pd.read_excel("Sales_Forecasting.xlsx")
```

In [118]:
```
dataset.head()
```

Out[118]:

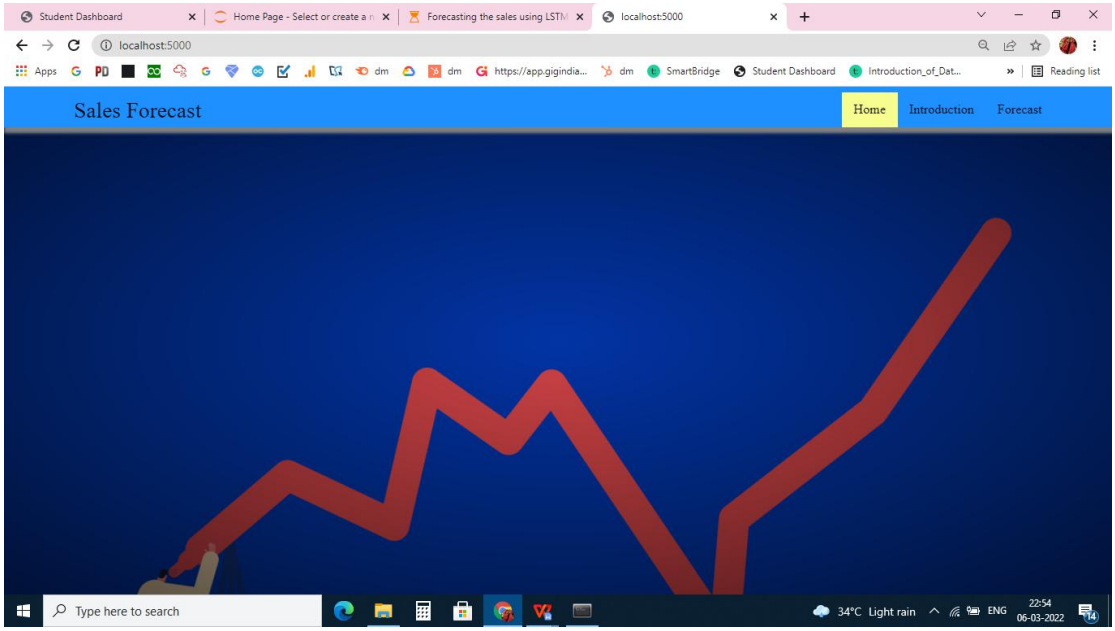
	HQ	Country	State_of_outlet	City_of_outlet	Month	Day	Year	Total_Sales
0	Asia	India	NaN	Bombay (Mumbai)	1	1	2005	72.2
1	Asia	India	NaN	Bombay (Mumbai)	1	2	2005	72.7
2	Asia	India	NaN	Bombay (Mumbai)	1	3	2005	74.3
3	Asia	India	NaN	Bombay (Mumbai)	1	4	2005	78.9
4	Asia	India	NaN	Bombay (Mumbai)	1	5	2005	81.5

In [119]:
```
dataset.tail()
```

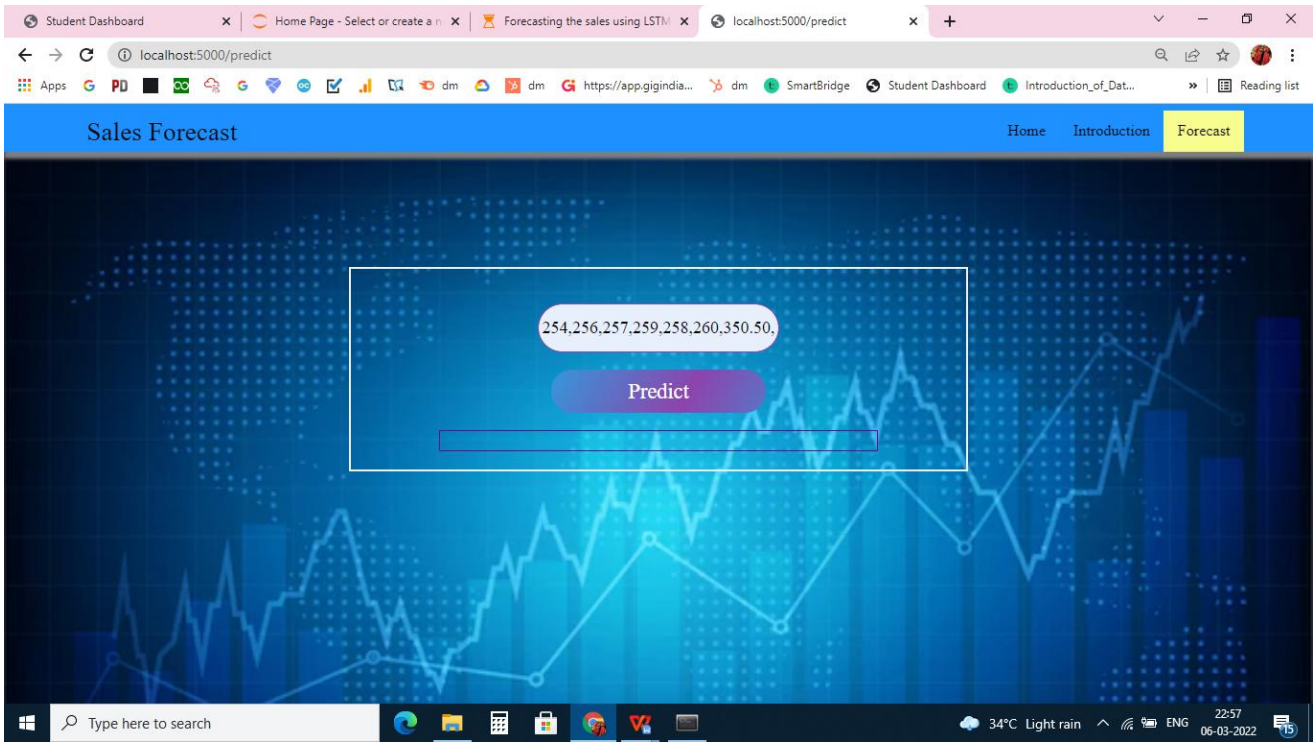
Out[119]:

	HQ	Country	State_of_outlet	City_of_outlet	Month	Day	Year	Total_Sales
17531	Asia	India	NaN	Delhi	12	27	2016	61.3
17532	Asia	India	NaN	Delhi	12	28	2016	61.7
17533	Asia	India	NaN	Delhi	12	29	2016	59.3
17534	Asia	India	NaN	Delhi	12	30	2016	57.1
17535	Asia	India	NaN	Delhi	12	31	2016	58.3

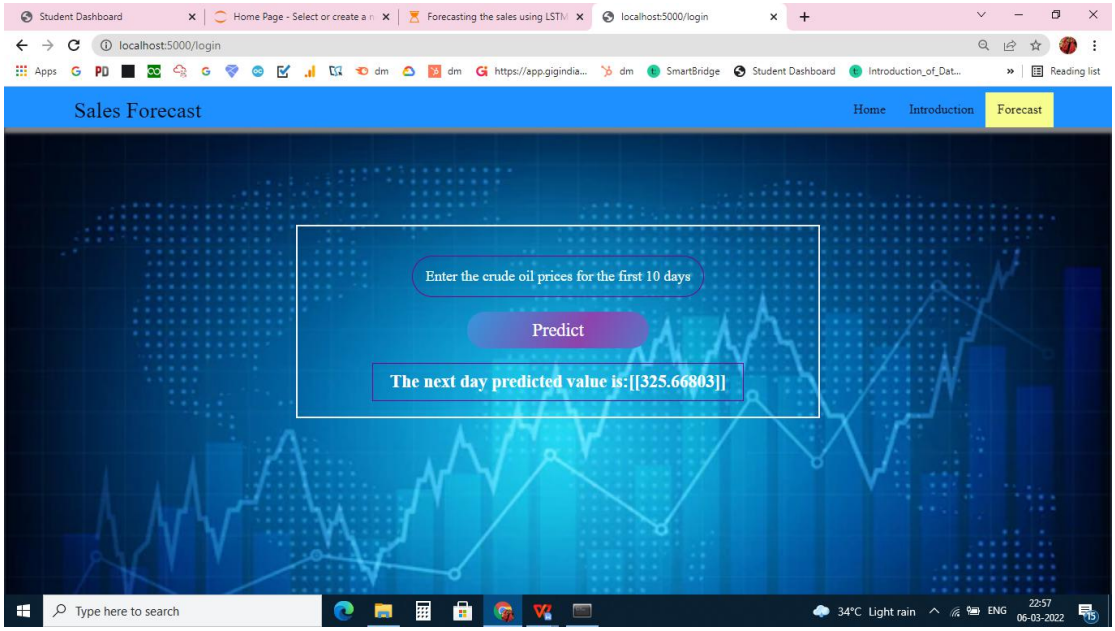
UI OUTPUT:



ENTERING 10 VALUES:



PREDICTED THE VALUE:



7 ADVANTAGES AND DISADVANTAGES

ADVANTAGES:

- Useful in forecasting sales for industrial products, consumer, durables and new products
- Relatively inexpensive and fast
- Alignment of Strategy and Results
- Opportunities Adjustments

DISADVANTAGES:

- Difficulty getting a panel experts
- Longer time for getting consensus
- Uncertain environment

8 APPLICATION

- Shopping malls
- Book store
- Medical store

9 CONCLUSION

- Know fundamental concepts and techniques of time series forecasting and LSTM
- Gain a broad understanding of time series data.
- Know how to split the data for time series forecasting.
- Know how to build a web application using the Flask framework.

10 FUTURE SCOPE

Sales forecasting allows companies to efficiently allocate resources for future growth and manage its cash flow. Sales forecasting also helps businesses to estimate their costs and revenue accurately based on which they are able to predict their short-term and long-term performance

11 BIBLIOGRAPHY

- https://youtu.be/pCuV_7H2vyQ
- https://www.neuraldesigner.com/solutions/sales-forecasting
- https://youtu.be/AtkWpgJJHgQ
- https://towardsdatascience.com/7-ways-to-handle-missing-values-in-machine-learning-1a6326adf79e

12 APPENDIX

```
import pandas as pd #Reading the dataset

import numpy as np #numerical calculations

import matplotlib.pyplot as plt #for data visualization

import statsmodels.api as sm #statistical data exploration

#tensorflow :open source used for both ML and DL for computation

from tensorflow.keras.models import Sequential#it is a plain stack of layers

from tensorflow.keras.layers import Dense#Dense layer is the regular deeply connected neural network layer

from tensorflow.keras.layers import LSTM #Long Short Trem Memory

data=pd.read_excel("Sales_Forecasting.xlsx")

data.head()

data.info()

data.describe()

from datetime import datetime

data['Date'] = data.apply(lambda row: datetime.strptime(f"{int(row.Year)}-{int(row.Month)}-{int(row.Day)}", '%Y-%m-%d'), axis=1)

data['Date'].min(), data['Date'].max()

data.shape

cols = ['Month','Day','Year','City_of_outlet','State_of_outlet','Country','HQ']

data.drop(cols, axis=1, inplace=True)
```

```
data = data.sort_values('Date')
data.head()
data.isnull().sum()
data = data.groupby('Date')['Total_Sales'].sum().reset_index()
data.head()
sale = np.array(data['Total_Sales'])
plt.figure(figsize=(12,6))
plt.plot(sale)
plt.title('SALES FROM 2005 TO 2016')
plt.xlabel('Days')
plt.ylabel('Sales')
plt.show()
training_size=int(len(sale)*0.80)
test_size=len(sale)-training_size
train_data,test_data=sale[0:training_size],sale[training_size:]
def prepare_data(timeseries_data, n_features):
        X, y =[],[]
        for i in range(len(timeseries_data)):
                # find the end of this pattern
                end_ix = i + n_features
                # check if we are beyond the sequence
                if end_ix > len(timeseries_data)-1:
                        break
                # gather input and output parts of the pattern
                seq_x, seq_y = timeseries_data[i:end_ix], timeseries_data[end_ix]
                X.append(seq_x)
                y.append(seq_y)
        return np.array(X), np.array(y)
```

```
xtrain, ytrain = prepare_data(train_data, 10)

xtest, ytest = prepare_data(test_data, 10)

xtrain.shape, xtest.shape

xtrain = xtrain.reshape((xtrain.shape[0], xtrain.shape[1], 1))

xtest = xtest.reshape((xtest.shape[0], xtest.shape[1], 1))

xtrain.shape, xtest.shape

model=Sequential()

model.add(LSTM(50,activation='relu',return_sequences=True,input_sha
pe=(10,1)))

model.add(LSTM(50,activation='relu'))

model.add(Dense(1))

model.compile(loss='mse',optimizer='adam')

model.summary()

model.fit(xtrain,ytrain,validation_data=(xtest,ytest),epochs=150,batch_si
ze=64,verbose=1)

a=[[275,281,279,287,271,281,288,255,278,298]]

A

model.predict(a)

model.save('forcast_sales.h5')

x_input=sale[4372:4382].reshape(1,-1)

x_input.shape

temp_input=list(x_input)

temp_input

temp_input=list(x_input)

temp_input

from numpy import array

lst_output=[]

n_steps=10

i=0
```

```
while(i<10):

    if(len(temp_input)>=10):
        #print("temp_input",temp_input)
        x_input=np.array(temp_input[0:])
        print("{} day input {}".format(i,x_input))
        x_input=np.expand_dims(x_input, axis=0)
        #x_input=scaler.transform(x_input)
        x_input = x_input.reshape((1, n_steps, 1))
        yhat = model.predict(x_input, verbose=0)
        #yhat=scaler.inverse_transform(yhat)
        print("{} day output {}".format(i,yhat))
        temp_input.extend(yhat[0].tolist())
        temp_input=temp_input[1:]
        #print(temp_input)
        lst_output.extend(yhat.tolist())
        i=i+1
    else:
        print("Please give 10 number of inputs")
```