National College of Ireland

Configuration Manual

MSc Research Project
MSc in Data Analytics

Himanshu Rathee
20132689

School of Computing
National College of Ireland

Submitted to:
Dr. Rashmi Gupta

National College of Ireland

National College of Ireland

Project Submission Sheet – 2019/2020

Student Name:	Himanshu Rathee		
Student ID:	20132689		
Programme:	MSc in Data Analytics	**Year:**	2020-2021
Module:	Configuration module for research project		
Lecturer:	Dr. Rashmi Gupta		
Submission Due Date:	16/08/2021		
Project Title:	Repo Rate modeling based on financial and economic variables		
Word Count:			

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the references section. Students are encouraged to use the Harvard Referencing Standard supplied by the Library. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action. Students may be required to undergo a viva (oral examination) if there is suspicion about the validity of their submitted work.

Signature: Himanshu Rathee

Date: 16/08/2021

PLEASE READ THE FOLLOWING INSTRUCTIONS:

1. Please attach a completed copy of this sheet to each project (including multiple copies).
2. Projects should be submitted to your Programme Coordinator.
3. **You must ensure that you retain a HARD COPY of ALL projects**, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer. Please do not bind projects or place in covers unless specifically requested.
4. You must ensure that all projects are submitted to your Programme Coordinator on or before the required submission date. **Late submissions will incur penalties.**
5. All projects must be submitted and passed in order to successfully complete the year. **Any project/assignment not submitted will be marked as a fail.**

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

1. Introduction to the document

This document can be used to configure and run the python code written by the author for this research project. This document includes all the necessary steps along with the appropriate screenshot to replicate the research work.

2. System configuration

The basic requirements to carry the research work are:

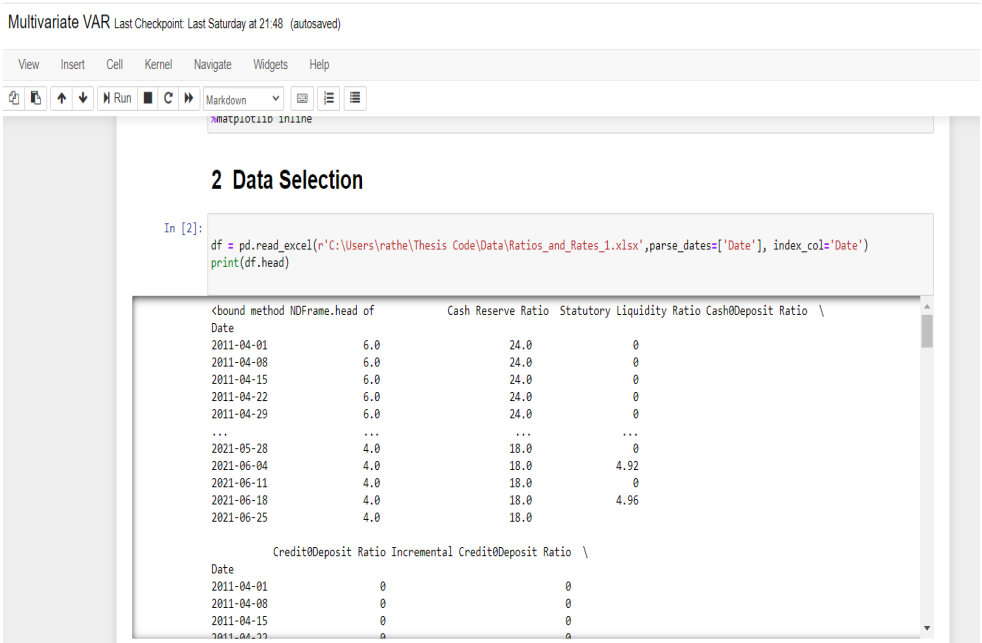
- PC
- Windows 10
- Anaconda
- Jupyter Notebook
- Google colab
- Microsoft Office

A major part of the coding has been done using jupyter notebook in python 3.8. However, Microsoft office tools such as MS-Word, and Ms-Excel have also been utilized in the data-gathering stage.

3. Data Selection

Two sets of data have been downloaded and used for this research.

- Financial and Economic Variables: This data set has been downloaded from Indian Economy database. The link to the page is https://dbie.rbi.org.in/DBIE/dbie.rbi?site=publications. The file Ratios and Rates has been downloaded in .xlsx format and analyzed in Ms-Excel. This file has been read using python and stored as a dataframe. Alternatively, Ratio_Final csv file can be exported to access all the financial variables.



- Twitter scraping: The python code in the file scraping_tweets.ipynb could be replicated in jupyter notebook or google colab to extract tweets based on search strings. The data retrieved has been stored in a csv file named Tweets3.csv

4. Data Transformation

The transformation of the financial and economic variable dataset has been performed in each python file before the model building. However, the transformation of the data scraped from Twitter has been performed in twitter_transformation.ipynb. This file can be replicated and run to assign sentiments to tweets using the Textblob.

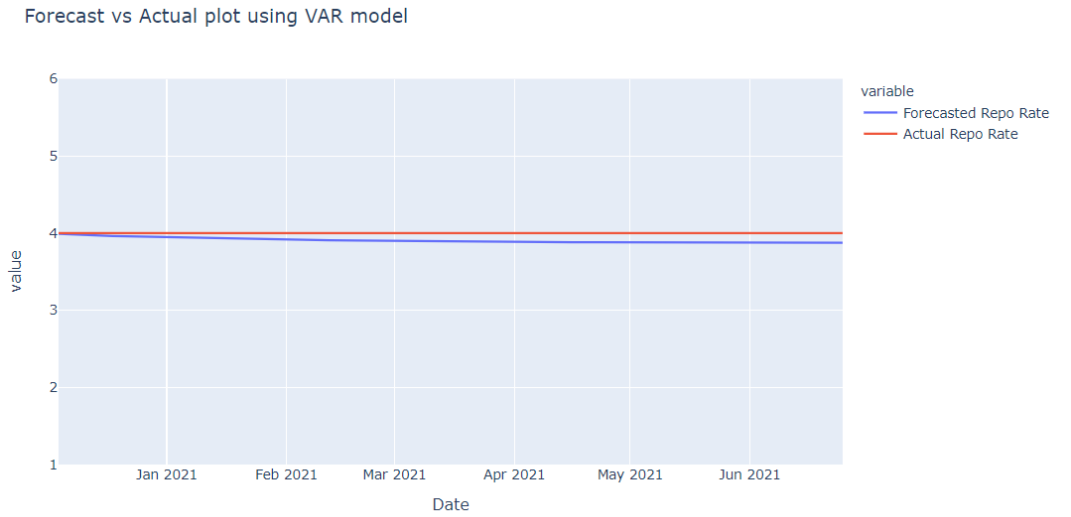
```
In [22]: sentyEng = []

for i,text in enumerate(MixedTweet.tweet):
  blob = TextBlob(text)
  print(blob.sentiment)
  if blob.sentiment[0]>0:
     print('Positive')
     sentyEng.append('Positive')
  elif blob.sentiment[0]<0:
     print('Negative')
     sentyEng.append('Negative')
  else:
     print('Neutral')
     sentyEng.append('Neutral')
```

5. Machine Learning Models

- Multivariate VAR model: The python file Multivariate VAR.ipynb can be replicated and run to get the results of the VAR model. All the libraries that have been used have been declared in the file. This file has been labeled and has all the steps required to apply a VAR model to the financial dataset. The evaluations and forecasts are at the end of the file.

```
In [38]: mse1 = mean_squared_error(df_forecast['Policy Repo Rate_2d'],df_test['Policy Repo Rate'])
rmse1 = math.sqrt(mse1)
mae1 = mean_absolute_error(df_forecast['Policy Repo Rate_2d'],df_test['Policy Repo Rate'])
MAPE1 = np.mean(np.abs((df_test['Policy Repo Rate'] - df_forecast['Policy Repo Rate_2d']) / df_forecast['Policy Repo Rate_2d']))
print('The Root Mean Square Error is for VAR model is {:.2f}'.format(rmse1))
print('The Mean Absolute Error is for VAR model is {:.2f}'.format(mae1))
print('The Mean Absolute Percentage Error is for VAR model is {:.2f}%'.format(MAPE1))

The Root Mean Square Error is for VAR model is 0.10
The Mean Absolute Error is for VAR model is 0.09
The Mean Absolute Percentage Error is for VAR model is 2.43%
```

In [36]:
```
import plotly.express as px
fig = px.line(df_plot, x="Date", y=df_plot.columns,
              title='Forecast vs Actual plot using VAR model')
fig.update_layout(yaxis_range=[1,6])
fig.show()
```



- LSTM /baseline Sequential and MLP model: The LSTM_MLP_NN_Baseline.ipynb file can be replicated or run to get the output of these three models. The dataset used for these models is the financial and economic variable data from the Indian Economy website itself.

4 Building LSTM model

In [13]:
```
#Defining the LSTM model
n_features=x_train.shape[1]
model=Sequential()
model.add(LSTM(100,activation='relu',input_shape=(1,1)))
model.add(Dense(n_features))

#Model summary
model.summary()
```

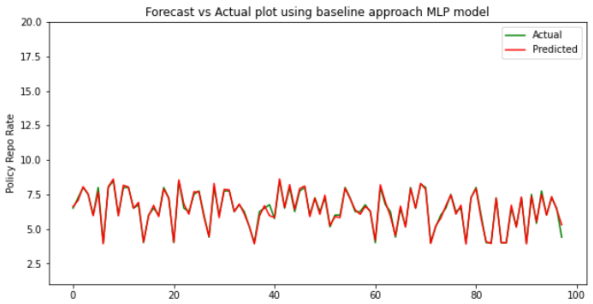
```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm (LSTM)                  (None, 100)               40800
_________________________________________________________________
dense (Dense)                (None, 1)                 101
=================================================================
Total params: 40,901
Trainable params: 40,901
Non-trainable params: 0
_________________________________________________________________
```

In [14]:
```
#Compiling
model.compile(optimizer='adam', loss = 'mse')
```

In [15]:
```
#Training
model.fit(x_train,y_train, epochs = 5, batch_size=1)
```

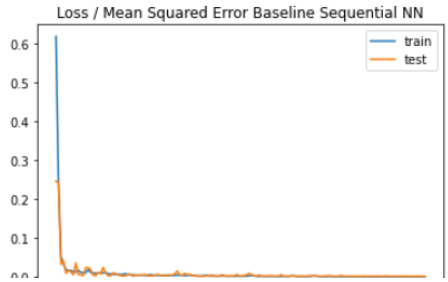
MLP model output

```
In [150]: msemlp = mean_squared_error(y_predmlp,y_testmlp)
          rmsemlp = math.sqrt(msemlp)
          maemlp = mean_absolute_error(y_predmlp,y_testmlp)
          MAPEmlp = np.mean(np.abs((y_testmlp - y_predmlp) / y_predmlp)) * 100
          print('The Root Mean Square Error for baseline MLP model is {:.2f}'.format(rmsemlp))
          print('The Mean Absolute Error for baseline MLP model is {:.2f}'.format(maemlp))
          print('The Mean Absolute Percentage Error for baseline MLP model is {:.2f}%'.format(MAPEmlp))

          The Root Mean Square Error for baseline MLP model is 0.18
          The Mean Absolute Error for baseline MLP model is 0.13
          The Mean Absolute Percentage Error for baseline MLP model is 2.09%

In [159]: plt.figure(figsize=(10,5))
          plt.title('Forecast vs Actual plot using baseline approach MLP model')
          plt.plot(tempval , label = 'Actual', color = 'g')
          plt.plot(y_predmlp , label = 'Predicted', color = 'r')
          plt.ylim([1, 20])
          plt.ylabel("Policy Repo Rate")
          plt.legend()

Out[159]: <matplotlib.legend.Legend at 0x25ad6d1e7c0>
```



Sequential Model

```
In [84]: model = Sequential()
         model.add(Dense(25, input_dim=11, activation='sigmoid', kernel_initializer='he_uniform'))
         model.add(Dense(10, activation='sigmoid'))
         model.add(Dense(1, activation='linear'))

In [85]: opt = SGD(learning_rate=0.1,momentum=0.9)

In [86]: model.compile(loss='mean_squared_error', optimizer=opt)

In [87]: history = model.fit(trainX, trainy, validation_data=(testX, testy), epochs=150, verbose=0)
         # evaluate the model
         train_mse = model.evaluate(trainX, trainy, verbose=0)
         test_mse = model.evaluate(testX, testy, verbose=0)
         print('Train mse: %.5f, Test mse: %.5f' % (train_mse, test_mse))

         Train mse: 0.00094, Test mse: 0.00139

In [88]: pyplot.title('Loss / Mean Squared Error Baseline Sequential NN')
         pyplot.plot(history.history['loss'], label='train')
         pyplot.plot(history.history['val_loss'], label='test')
         pyplot.legend()
         pyplot.show()
```



- Novel Sequential and MLP model: The file Novel_Model.ipynb could be utilized and run to get the merged data of both the sentiment analysis as well as the financial variable data. These data sets have been merged by taking the weeks and years from both the data sets

```
In [2]: df = pd.read_csv("Sentitweets.csv")

In [3]: def weekNumberFromDate(datee):
            datee = datetime.datetime.strptime(datee, "%Y-%m-%d")
            weekNumber = datetime.date(datee.year, datee.month, datee.day).isocalendar()[1]
            return weekNumber

In [4]: def yearFromDate(datee):
            datee = datetime.datetime.strptime(datee, "%Y-%m-%d")
            return datee.year

In [5]: df["Year"] = df["date"]

In [6]: df = df.rename(columns={'date': 'Week'})

In [7]: df["Week"] = df["Week"].apply(weekNumberFromDate)

In [8]: df["Year"] = df["Year"].apply(yearFromDate)

In [9]: df = df.drop(columns="tweet")

In [10]: df = df.groupby(['Week', 'Year'], as_index=False)['Sentiments'].mean()

In [11]: df
```

Out[11]:

	Week	Year	Sentiments
0	1	2012	0.0
1	1	2014	0.0
2	1	2016	1.0
3	1	2017	-1.0

MERGED dataset

```
In [19]: new_df = rf.merge(df,  how='left')

In [20]: new_df
```

Out[20]:

	Date	Cash Reserve Ratio	Statutory Liquidity Ratio	Policy Repo Rate	Reverse Repo Rate	Marginal Standing Facility (MSF) Rate	Bank Rate	Call Money Rate (Weighted Average)	910Day Treasury Bill (Primary) Yield	INR0US$ Spot Rate (Rs. Per Foreign Currency)	INR0Euro Spot Rate (Rs. Per Foreign Currency)	Forward Premia of US$ 10month	Forward Premia of US$ 30month	Year	Week	Sentiments
0	2011-04-01	6.0	24.0	6.75	5.75	0.00	6.00	7.60	0.00	0.00	0.00	0.00	0.00	2011	13	NaN
1	2011-04-08	6.0	24.0	6.75	5.75	0.00	6.00	6.22	0.00	0.00	0.00	0.00	0.00	2011	14	0.000000
2	2011-04-15	6.0	24.0	6.75	5.75	0.00	6.00	6.77	0.00	0.00	0.00	0.00	0.00	2011	15	NaN
3	2011-04-22	6.0	24.0	6.75	5.75	0.00	6.00	6.40	0.00	0.00	0.00	0.00	0.00	2011	16	NaN
4	2011-04-29	6.0	24.0	6.75	5.75	0.00	6.00	6.87	0.00	0.00	0.00	0.00	0.00	2011	17	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
485	2021-05-28	4.0	18.0	4.00	3.35	4.25	4.25	3.18	3.40	72.48	88.23	5.46	5.63	2021	21	0.000000
486	2021-06-04	4.0	18.0	4.00	3.35	4.25	4.25	3.13	3.41	73.03	88.45	3.62	4.08	2021	22	0.133333
487	2021-06-11	4.0	18.0	4.00	3.35	4.25	4.25	3.11	3.40	72.98	88.98	4.19	4.30	2021	23	0.000000
488	2021-06-18	4.0	18.0	4.00	3.35	4.25	4.25	3.16	3.47	74.14	88.26	3.97	4.15	2021	24	NaN
489	2021-06-25	4.0	18.0	4.00	3.35	4.25	4.25	3.15	3.47	74.18	88.57	3.80	3.99	2021	25	1.000000

490 rows × 16 columns

Data Transformation

1 Creating functions for imputing, handling NAs, null and empty values ¶

Function to impute missing values in a data frame

```
In [23]: new_df_WithoutDate = new_df.drop(columns = "Date")

In [24]: from sklearn.preprocessing import MinMaxScaler
         scaler = MinMaxScaler()
         new_df_WithoutDate = pd.DataFrame(scaler.fit_transform(new_df_WithoutDate), columns = new_df_WithoutDate.columns)
         new_df_WithoutDate.head()
```

Out[24]:

	Cash Reserve Ratio	Statutory Liquidity Ratio	Policy Repo Rate	Reverse Repo Rate	Marginal Standing Facility (MSF) Rate	Bank Rate	Call Money Rate (Weighted Average)	910Day Treasury Bill (Primary) Yield	INR0US$ Spot Rate (Rs. Per Foreign Currency)	INR0Euro Spot Rate (Rs. Per Foreign Currency)	Forward Premia of US$ 10month	Forward Premia of US$ 30month	Year	Week	Sentiments
0	1.0	1.0	0.611111	0.578313	0.0	0.291667	0.619699	0.0	0.0	0.0	0.0	0.0	0.0	0.230769	NaN
1	1.0	1.0	0.611111	0.578313	0.0	0.291667	0.430917	0.0	0.0	0.0	0.0	0.0	0.0	0.250000	0.5
2	1.0	1.0	0.611111	0.578313	0.0	0.291667	0.506156	0.0	0.0	0.0	0.0	0.0	0.0	0.269231	NaN
3	1.0	1.0	0.611111	0.578313	0.0	0.291667	0.455540	0.0	0.0	0.0	0.0	0.0	0.0	0.288462	NaN
4	1.0	1.0	0.611111	0.578313	0.0	0.291667	0.519836	0.0	0.0	0.0	0.0	0.0	0.0	0.307692	NaN

```
In [25]: imputer = KNNImputer(n_neighbors=5)
         impute_df = pd.DataFrame(imputer.fit_transform(new_df_WithoutDate),columns = new_df_WithoutDate.columns)

In [26]: impute_df
```

Out[26]:

Now with this merged dataset, the novel approach models have been trained. The results of both the models have been labeled and commented on in the file.