National College of Ireland

Project Submission Sheet – 2016/2017

School of Computing

Student Name:	Vimalraj Kumar		
Student ID:	15009025		
Programme:	M.Sc. Data Analytics	**Year:**	2016-2017
Module:	Configuration Manual		
Lecturer:	Catherine Mulwa		
Submission Due Date:	22/08/2016		
Project Title:	Prediction of FOrex		
Word Count:	816		

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 bibliography 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:

Date: 22/08/2016

PLEASE READ THE FOLLOWING INSTRUCTIONS:

1. Please attach a completed copy of this sheet to each project (including multiple copies).
2. **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.
3 Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

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

Contents

1.Introduction

This project presents the implementation prediction that can accuracy predict the foreign exchange rate. how the prediction accuracy can be improved by developing an ensemble model of the deep learning algorithm, Distributed Random forest and generalised linear model using sparkling water (Spark +H20). According to the researchers of literature review from 2000-2016, there are several models that has been used for predicting the foreign exchange rate. Among which Artificial Neural Network, Linear regression, Support vector machine, Arima are best-suited models for predicting the time series data. By having the above models as a base for this project and also by considering this project with the time series data of exchange rate, an additional feature of an ensemble is done on the output of these three models. This model is also implemented on the big data for getting better accuracy and faster predictions. Each model by using sparkling water produces the accuracy of more the 90\% and the ensemble models increase the accuracy of the model by 3\% with the accuracy of 93\%. The evaluation and the results of this model clearly delivers the ensemble method, on the sparkling water which can efficiently improve the accuracy and performance of the model. This implementation of the Project are achieved by following this configuration manual and the steps provided through it.

2.Environment Specification

The environmental specification provides details about what system are required to develop and implement this Forex prediction model. The Environmental specification consists of two Main systems,

1. The Host Laptop
2. The Distributed system, using Oracle virtual box,

2.1 Laptop Specification

The host laptop is the place where the complete demonstration of the project resides, the lap with good configuration are considered to hold the Distributed system using Oracle virtual box, because the system with less configuration, may lead to the hard in writing and executing the command and the program for this project.

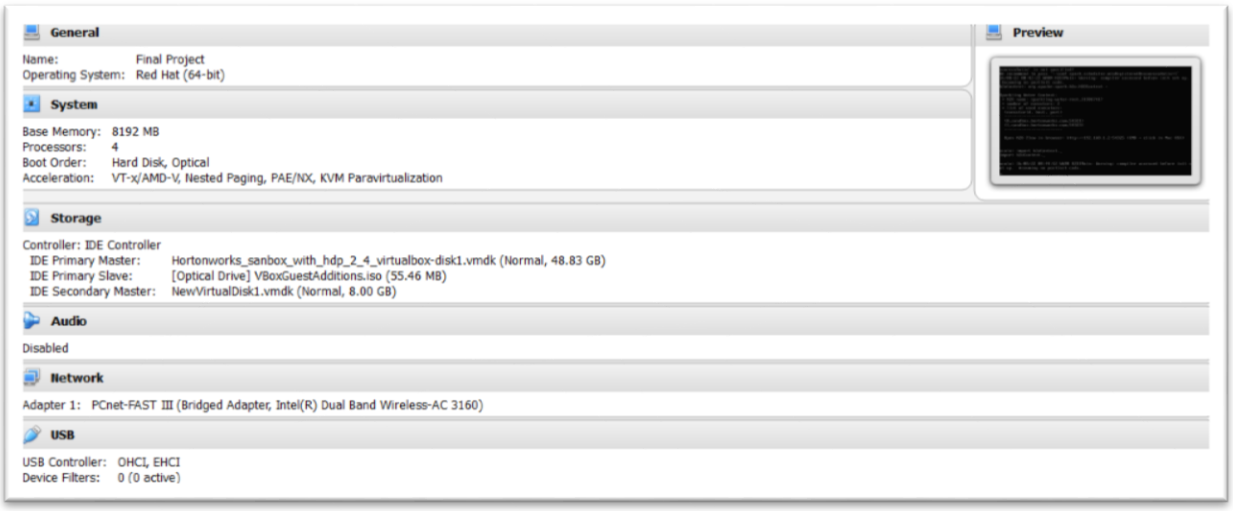
The Laptop Specification required are

Laptop Brand: HP Envy Series,
Operating Systems: Windows 10,
Processor: Intel core i7 processor,
Installed RAM Memory: 12 Giga Bytes,
System type: 64bit Operating System, x64 based processor,

2.2 Virtual System Environment

The Virtual system environment is the place where our distributed file system work with the help of Horton work sandbox. I order to develop the virtual system, the virtual machine tool like Oracle virtual box is used. The Specification of the Oracle virtual box to install Horton work sand box are given below
The Virtual System Specification required are

Laptop Brand: Red Hat,
Operating Systems: Windows 10,
No. of Processors: 4 processors,
Installed RAM Memory: 8 Giga Bytes,
System type: 64bit Operating System,
Network Setting: PC net -FAST III (Bridge Adapter, Intel(R) Dual Band wireless AC 3160)



3. Setup of Technologies

After meeting all the Environment and system specification, the next step is stalling all the required softwares and packages to develop our prediction model using sparkling water.

3.1 Oracle Virtual Box Environment

In order to implement the virtual system in the host laptop, Oracle virtual box is used which is open source tool for creating virtual machines.

Summary:

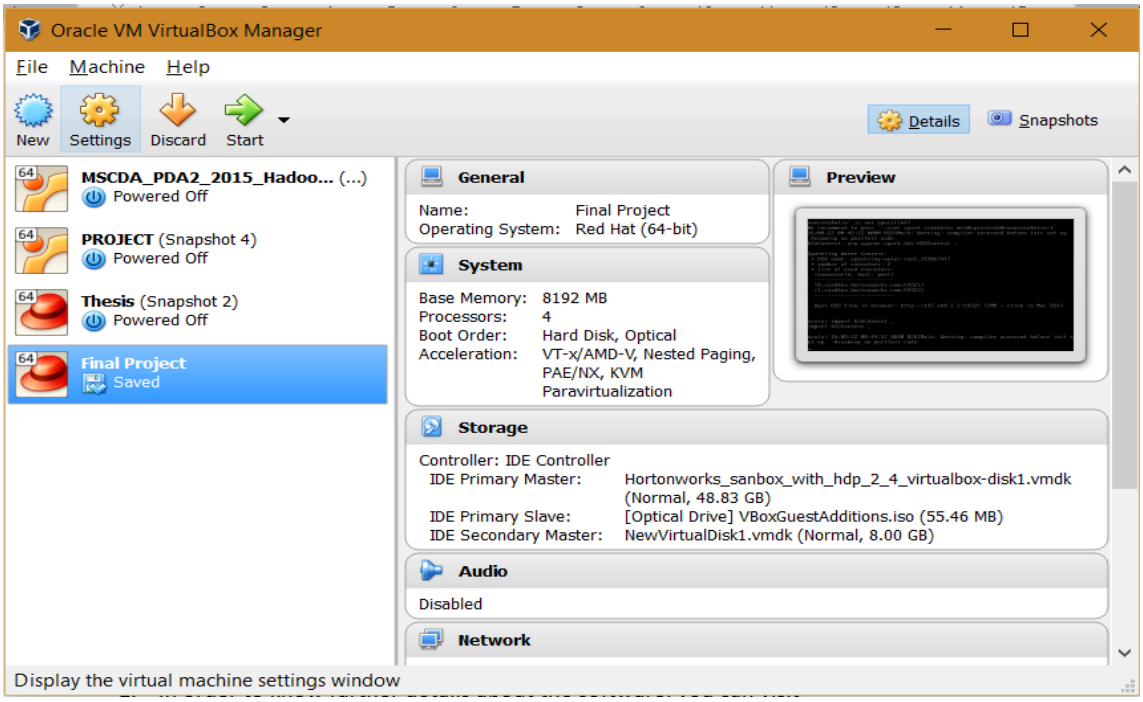
It is a powerful open source virtualization tool and it can run on several operating system including Windows, Linux, Macintosh, and Solaris hosts and supports a large number of guest operating systems as well.

Use:

- It is used to create the Virtual machine for implementing the Hadoop distributed file system using the Horton work sand box.
- Integrating with the Horton work sand box provides ability to all the big data technologies available in the market.
- In this project, to make use of the spark for implementing the sparkling water and the Hive for generating the ensemble this helps in achieving the virtual distributed system.

Installation:

1. In order to know further details about the software, you can visit https://www.virtualbox.org/
2. The Virtual Box binaries, source code and executable files are available in the link https://www.virtualbox.org/wiki/Downloads
3. Download and Install the recent version of the virtual box.
4. After installation, your oracle virtual box looks something similar to this.



3.2 Horton Work Sandbox

Hortonworks Sandbox is a highly portable application of Hadoop distributed file system along several big data tools available with the Graphical user interface.

Summary:

It is the famous Sandbox that provides the Hadoop distributed file system and parallel processing system. It also provides More Flexible Upgrades, Simplified Security Operations, Speed up Spark Streaming etc.

Use:

1. In order to predict the Forex, the big data technology of spark is used, on which the H20 high level deep learning application runs which in short called as Sparkling water.
2. In order to achieve this sparkling water, we are in the need of distributed environment to proceed with parallel computing so that the model implemented is efficient and realiable.

Prerequisites:

Before integrating it with the Oracle virtual box you should check its requirement once again and which are provided below,

1. Hosts of A 64-bit machine with a chip that supports virtualization machine.
2. Host Operating Systems of Windows 7, 8 or 10.
3. Supported Browsers: Google Chrome – latest stable release
4. At least 4 GB of RAM
5. Virtual Machine Environments like Oracle Virtual Box, version 4.2

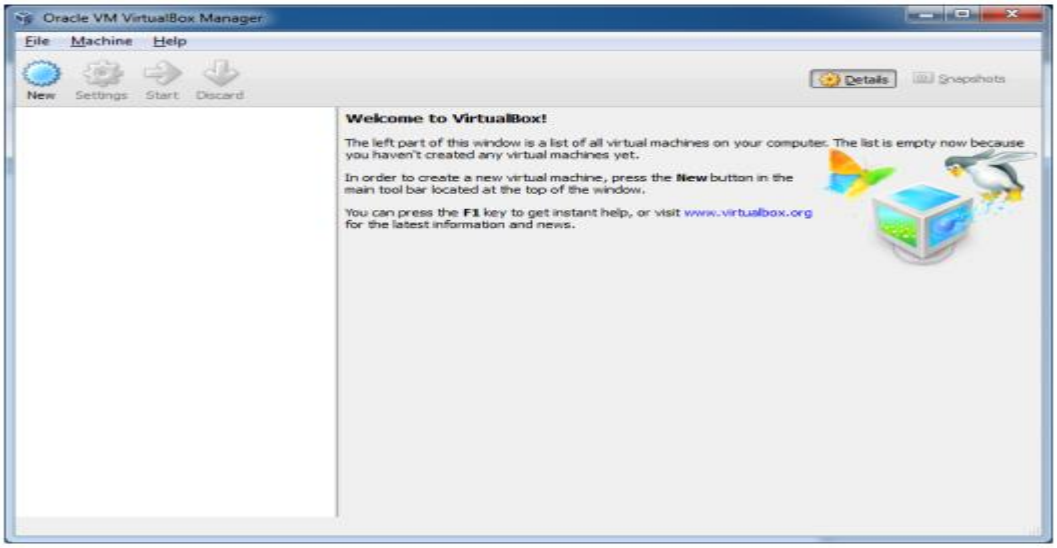
Installation and Integrating:

The following steps are followed to install and integrate the Horton work in the host environment.

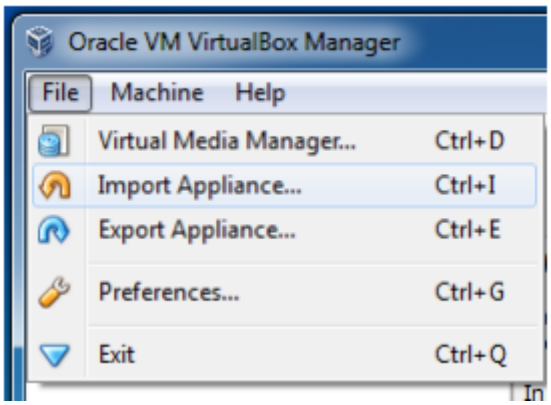
1. Download the Horton work sand box from the official link of Horton work http://hortonworks.com/downloads/#sandbox
2. It downloads the Virtual machine file which can be attached with the oracle virtual box.

3. Then open the Oracle Virtual box application by clicking the Oracle virtual box manager

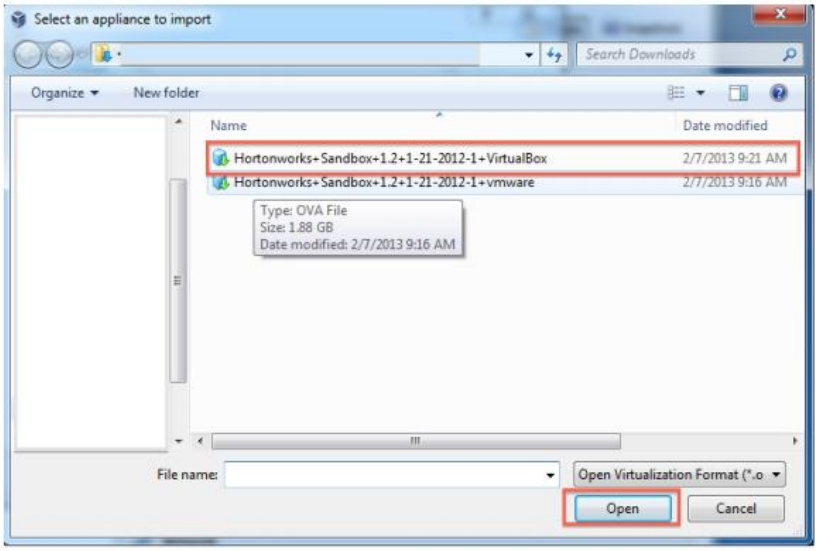
4. After the opening the oracle Virtual box manager, it looks similar to this



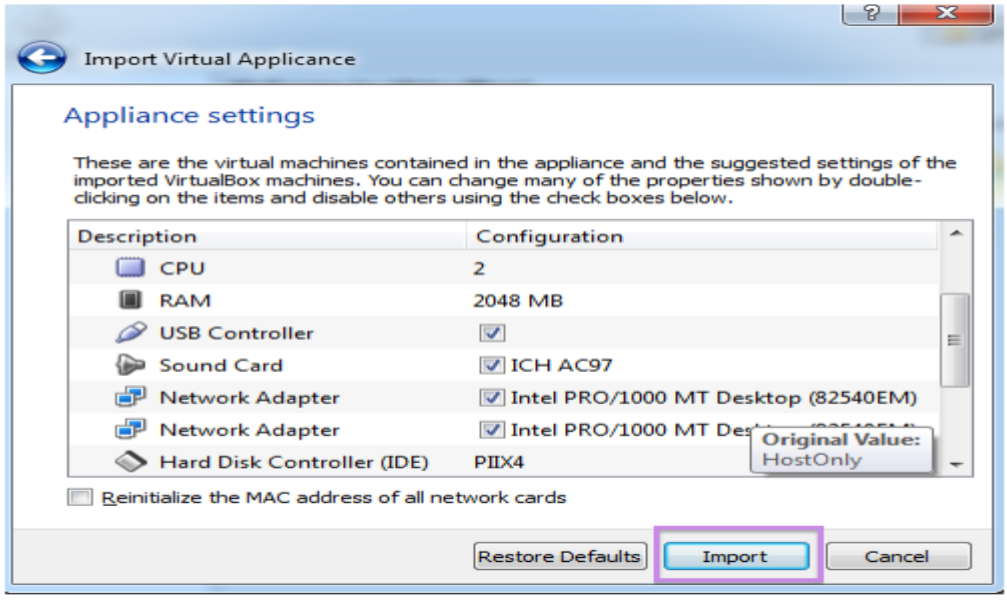
5. Importing the Horton work sandbox virtual machine file downloaded into the oracle virtual box manager.



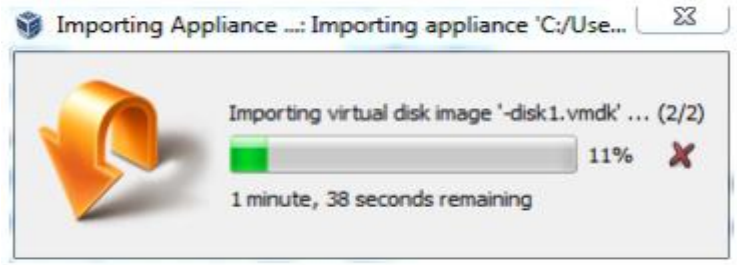
6. Click the Open appliance button; the file browser opens which load the Horton work with red had based operating system



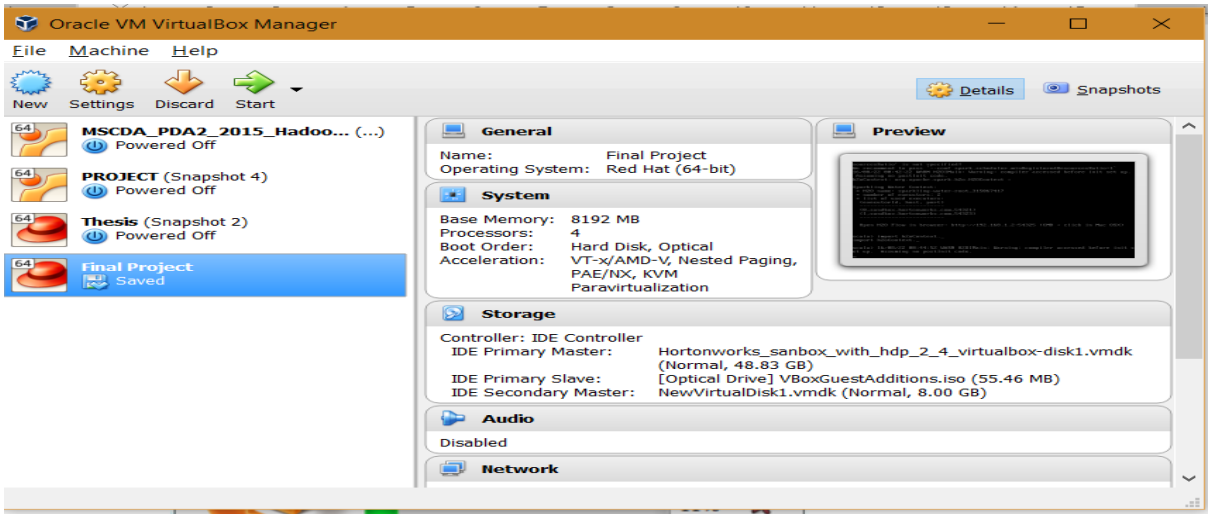
7. Click next to import the Virtual appliance screen
8. It throws up the Application setting dialog box where the default specification are shown and configured. If required you can change the configuration and click import button



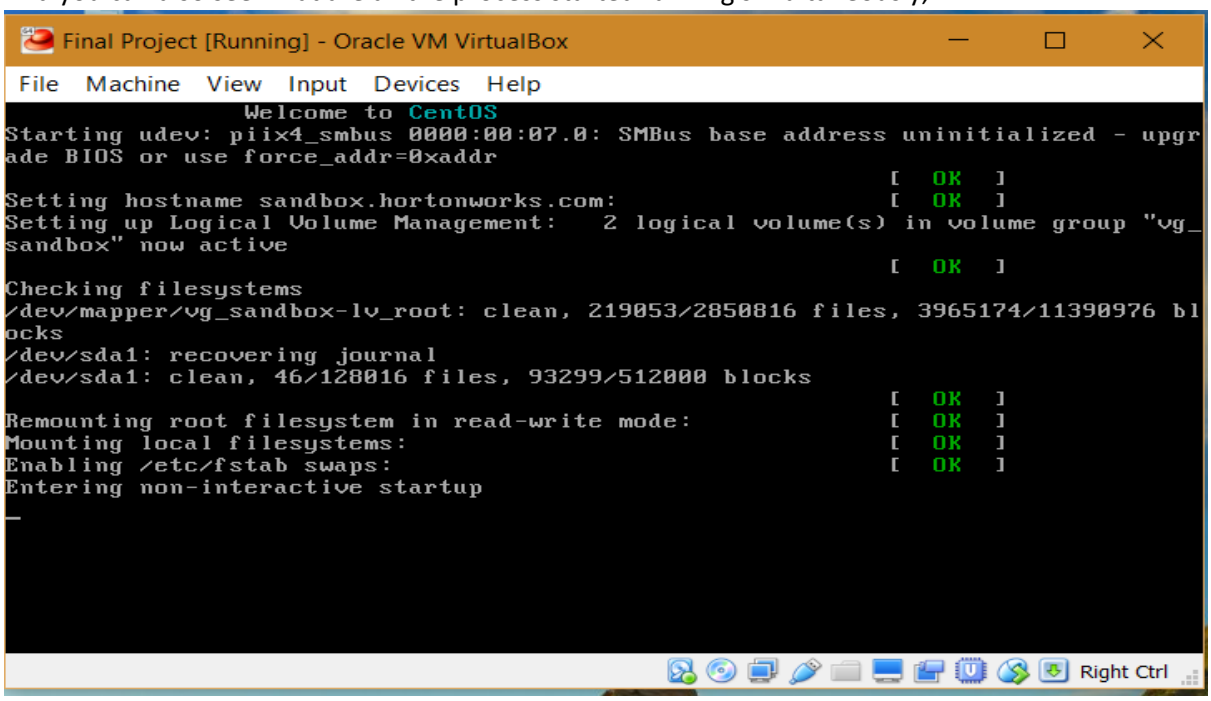
9. The appliance will import all the application



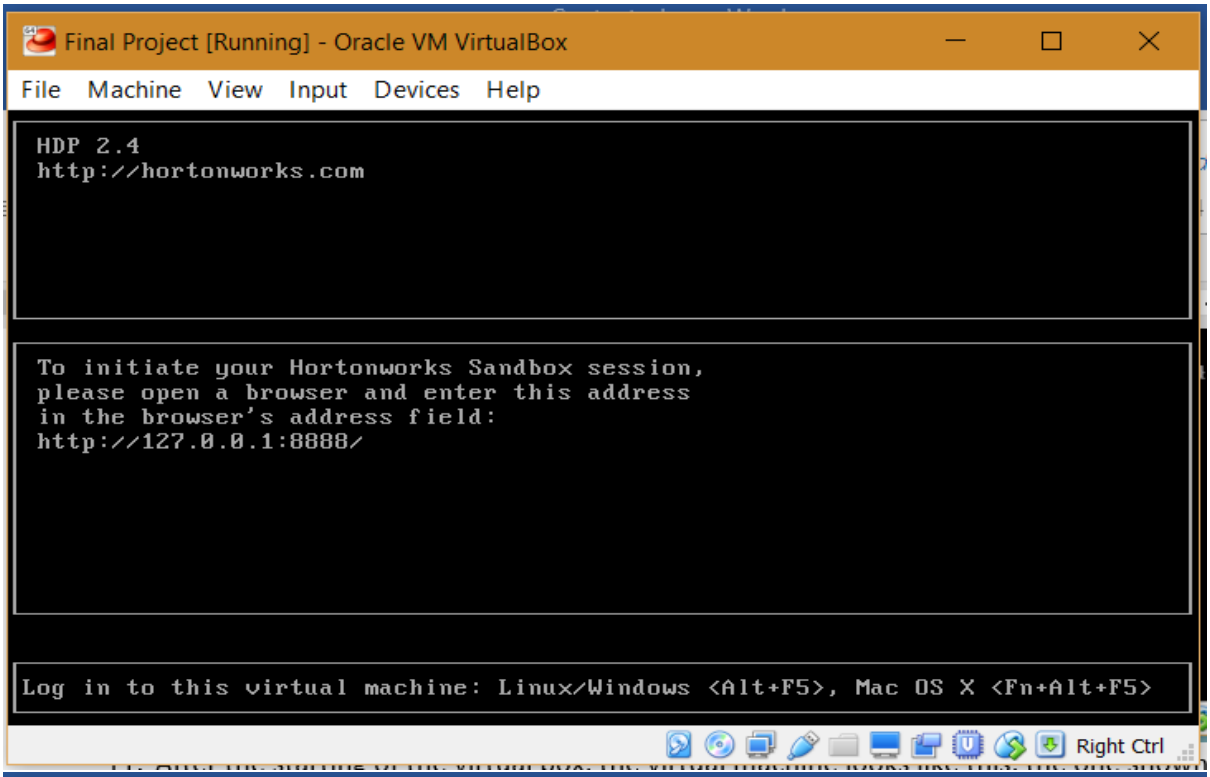
10. After finishing the importing, you can switch on the virtual machine by clicking the start of the oracle virtual box machine manager

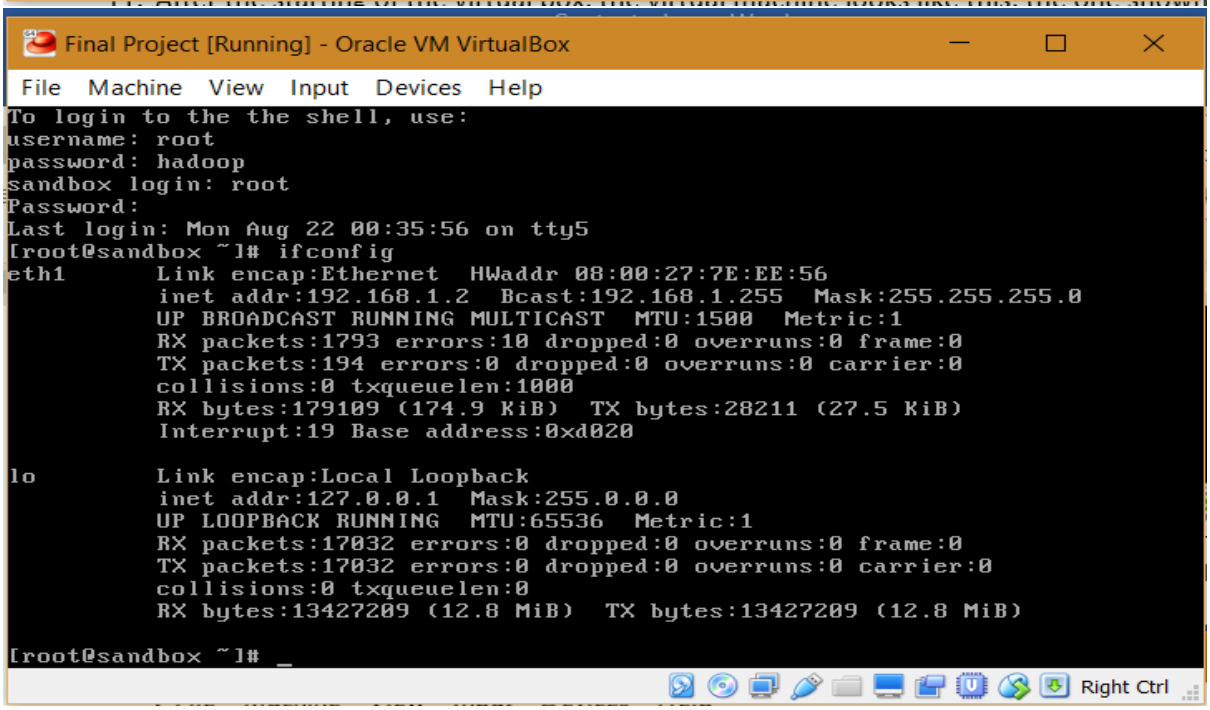


And you can also see what are all the process started running simultaneously,



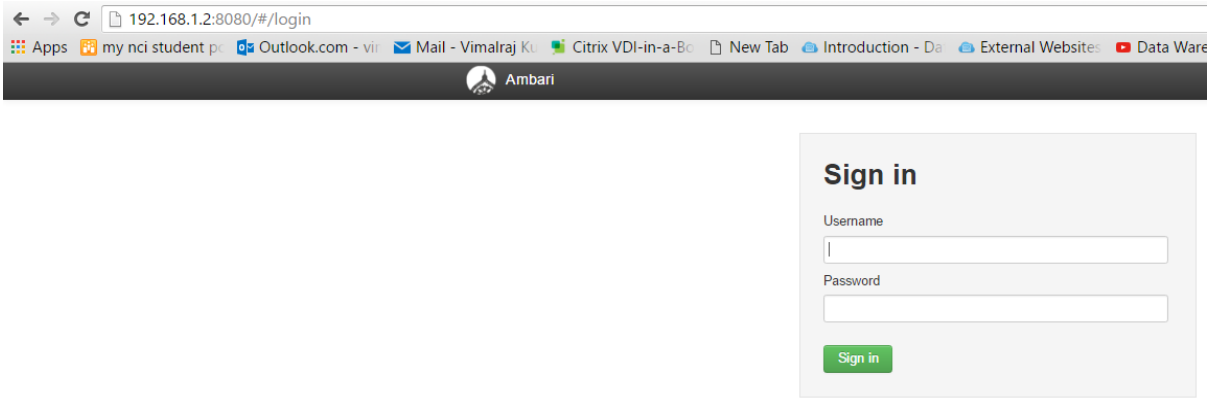
11. After the starting of the virtual box, the virtual machine looks like this, the one shown in the below diagram with prompting for user name and password of the root. By default, the password of the root is Hadoop which you to change after the first time login.



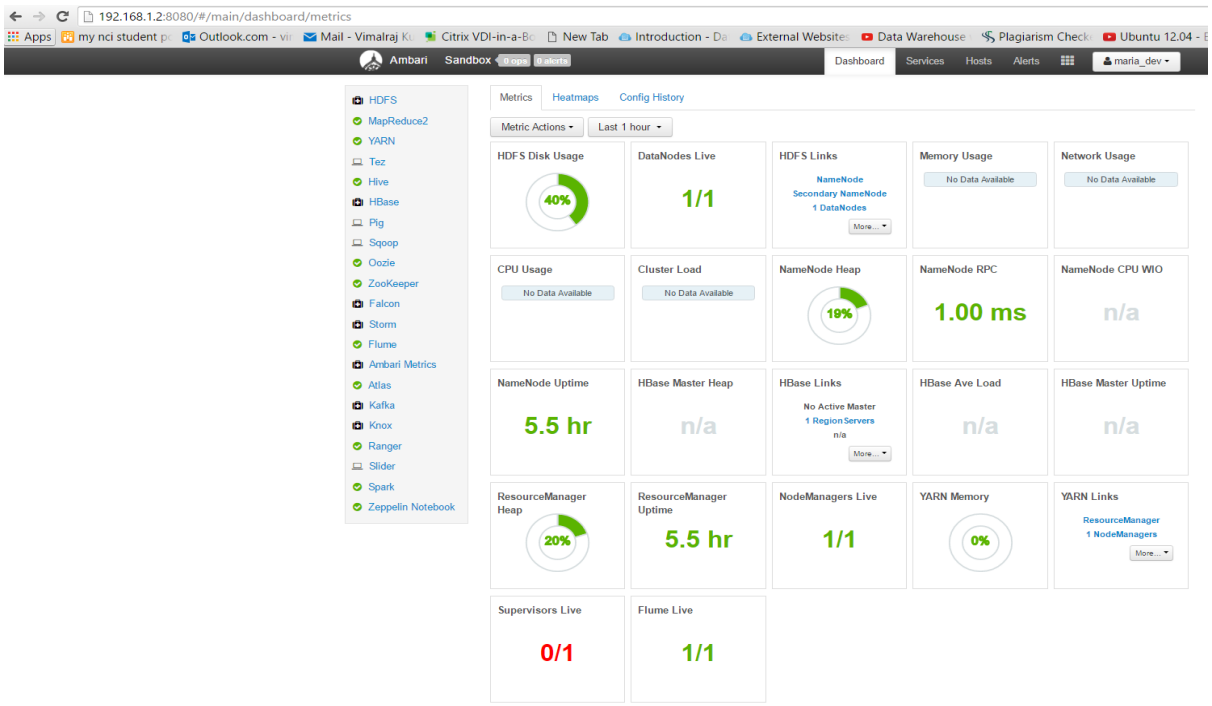


12. After the VM started, it provides the url through which you can communicate with the Hadoop ecosystem like HDFS, Hive, Pig, Spark, etc. you can find the ip address of the system by giving ipconfig so you are allowed to connect with ambary server with the ip address. In

my case, the url available is http://192.168.1.2:8000/ .



13. On navigating to that link, you can see that you can connect with the help of Ambari server with the default username and password of maria_dev. After Logging in it takes to the dashboard, where the performance of all the application are seen.



3.3 Putty

PuTTY is an SSH and telnet client, developed and used for interacting with the several ssh system through this application.

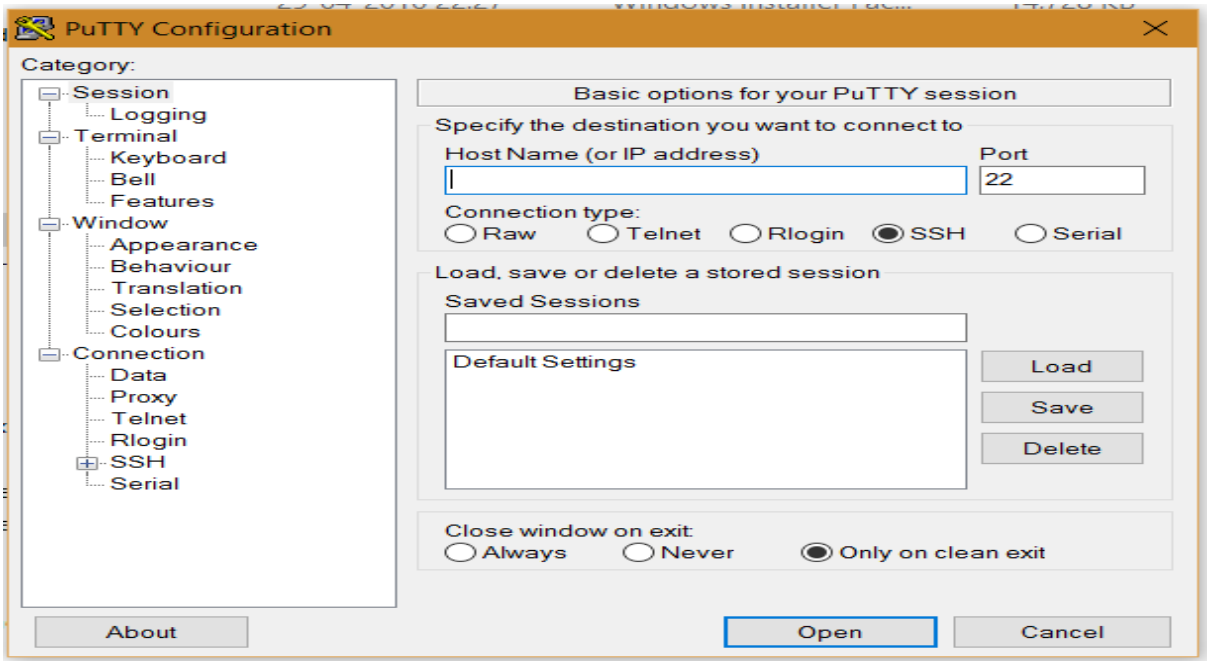
Use:

The Putty is used in our project , to insert few commands in the Hortonwork sandbox for starting and running the Sparkling water and moving file in the HDFS etc.

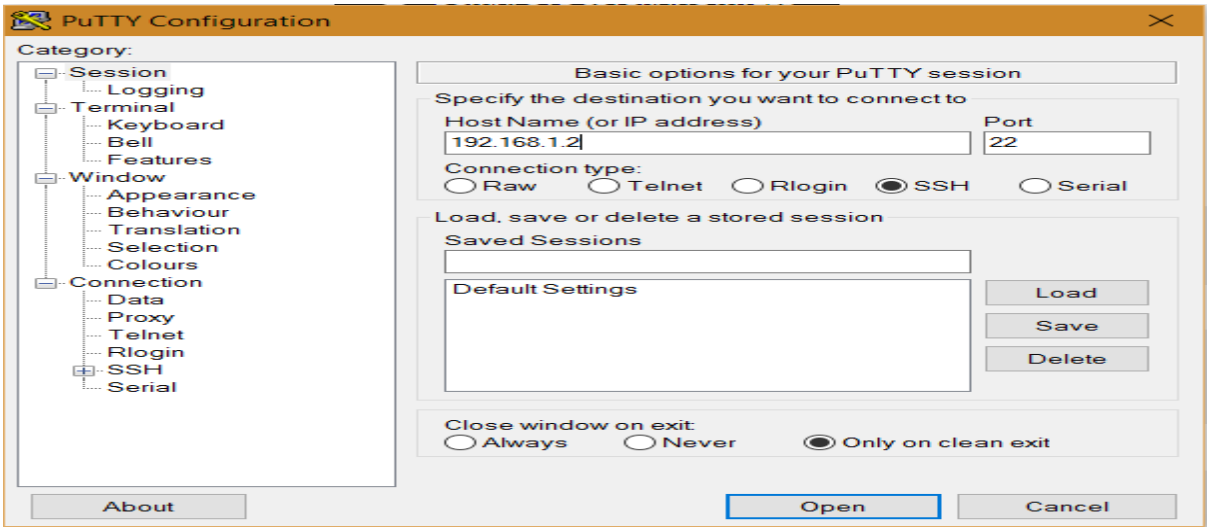
Installation:

1. You can download the putty and also can know more about the putty in this link http://www.putty.org/ .
2. After downloading you can start the application by directly double clicking in the application.

3. After the application started, the application looks something like the below diagram based on the version is used.

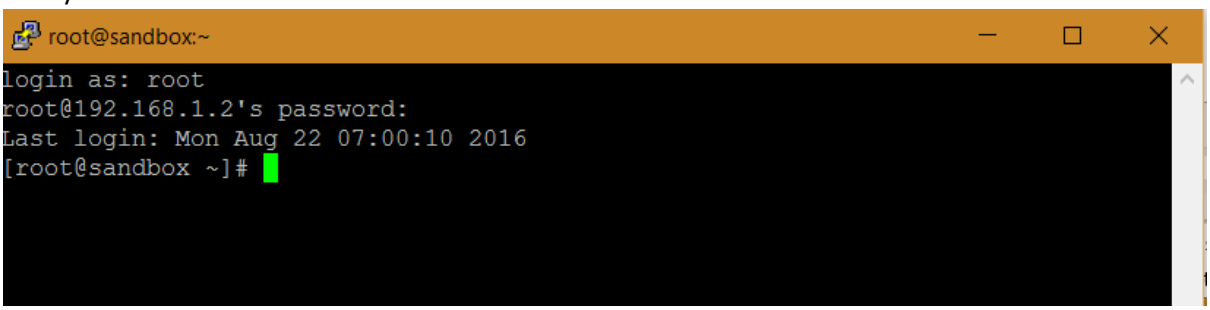


4. Accessing the Putty, Inorder to install software and run several program the putty terminal is used. It as for the Ip address, provide the Ip address got form your horton work sand box.



It then prompt for username and password of the virtual system in my case it is root and the password is xxx.

5. After the correct username and password of the virtual system, you will be able to access the system.



3.4 Sparkling Water

Summary

Sparkling Water is an application that integrates H2O's fast scalable machine learning engine with Spark. It provides several features like to publish Spark data structures as H2O's frames and vice versa.

Use:

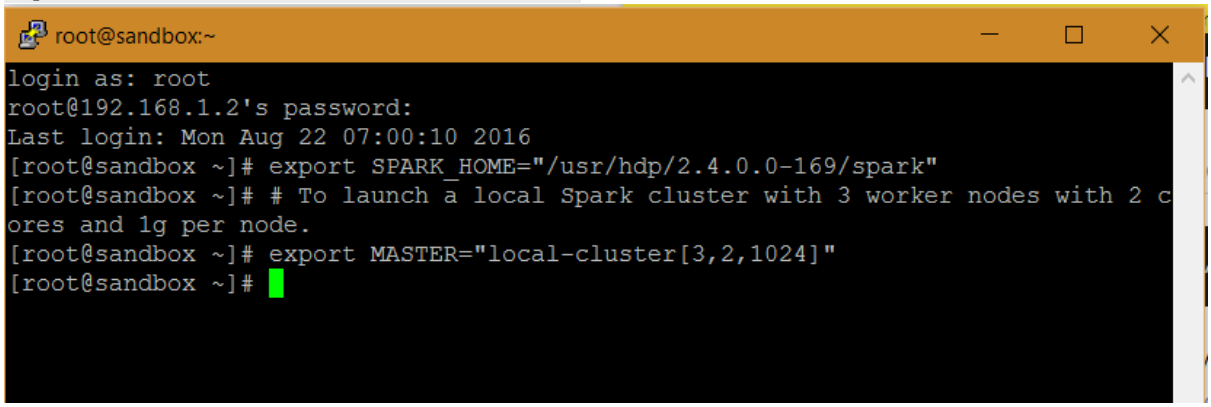
1. It is used for developing our ensemble data mining model for predicting the forex rates using sparkling water.
2. It is also used for cleansing of data for parsing and data exploration.
3. We have used deep learning, distributed random forest, and generalized linear algorithm for predicting the forex rate.

Installation:

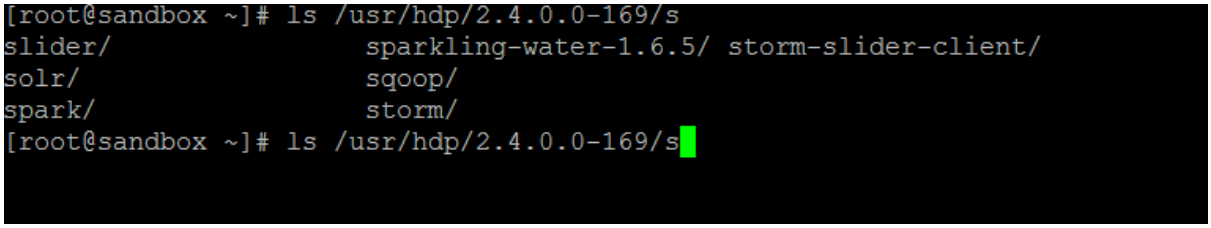
There few steps which needs to be followed to download, install, and integrate it with the spark.

1. First step is downloading and sparkling water from the link http://h2o-release.s3.amazonaws.com/sparkling-water/rel-1.5/6/index.html
2. Setting up the spark environmental variable is the command prompt using putty. And the Environmental variables are given below,

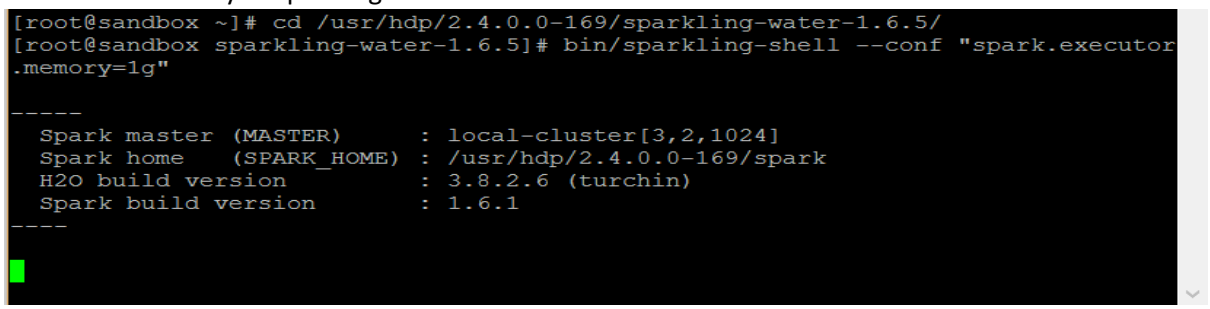
```
export SPARK_HOME="/path/to/spark/installation"
# To launch a local Spark cluster with 3 worker nodes with 2 cores and 1g
per node.
export MASTER="local-cluster[3,2,1024]"
```



3. Then the file downloaded is unzip and placed folder near to spark



4. After moving to the folder where the sparkling water extracted folder resides give this command to start you sparkling water

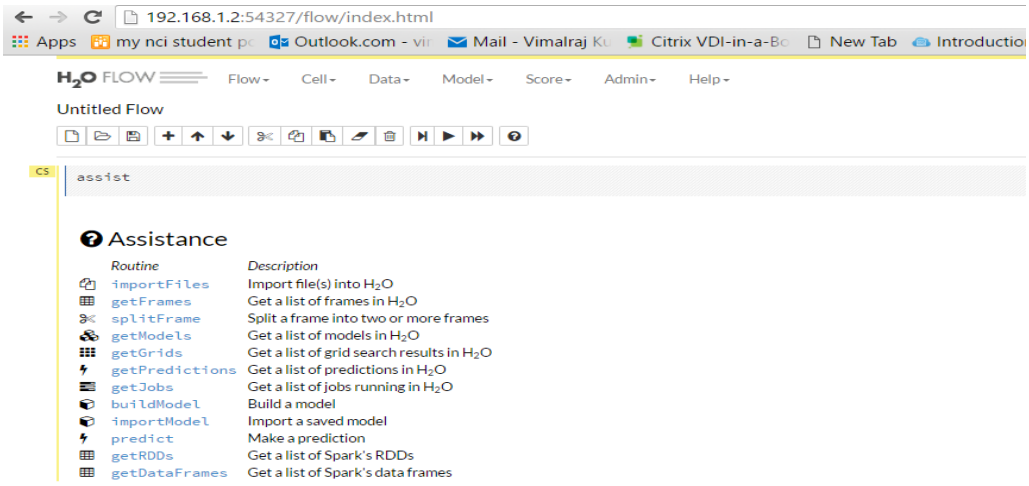


5. After the sucessfully, starting of the spark and then H2O cloud instance is created inside the Spark cluster by importing few packages.

```
import org.apache.spark.h2o._
val h2oContext = new H2OContext(sc).start()
import h2oContext._
```

6. After starting the H20 instace looks something like as shown in th ebelow diagram.

7. You can navigate to the H2o flow using the URL provided in my case it is http://192.168.1.2:54327/



3.5 R Studio

Summary:

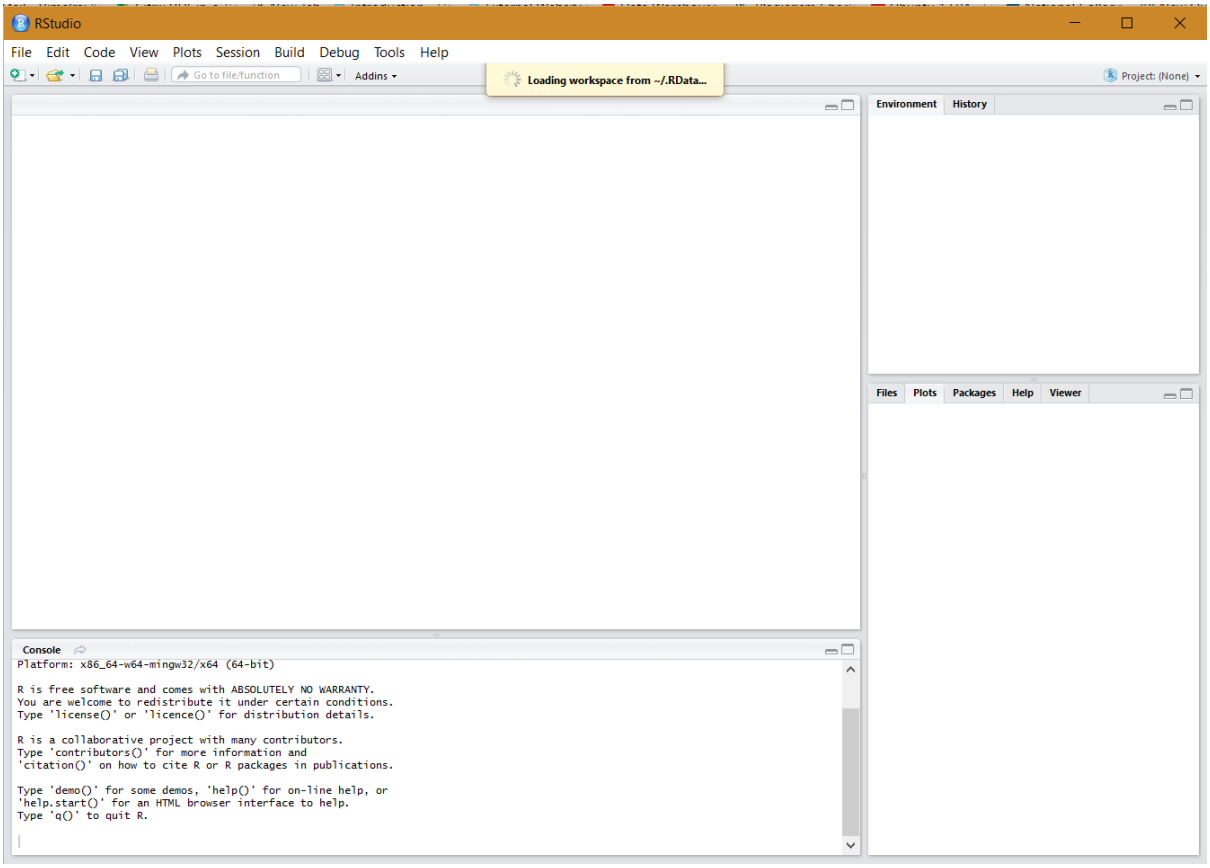
It is used for programming one of the high level programing language called R and it also provided with a set of integrated tools designed to help you be more productive with R.

Use:

1. It mainly uses for treating the data, exploring and Integrating the dataset. Since the dataset is collected from several resource we need to make them as one for that we use.
2. Several cleaning process like handling NA values and also cleansing of the data.

Installation:

1. You can download the R-studio software from https://www.rstudio.com/products/rstudio/download/ and find the software version based on your operating system.
2. After Installing the R-Studio, the application looks like this



3.6 Tableau

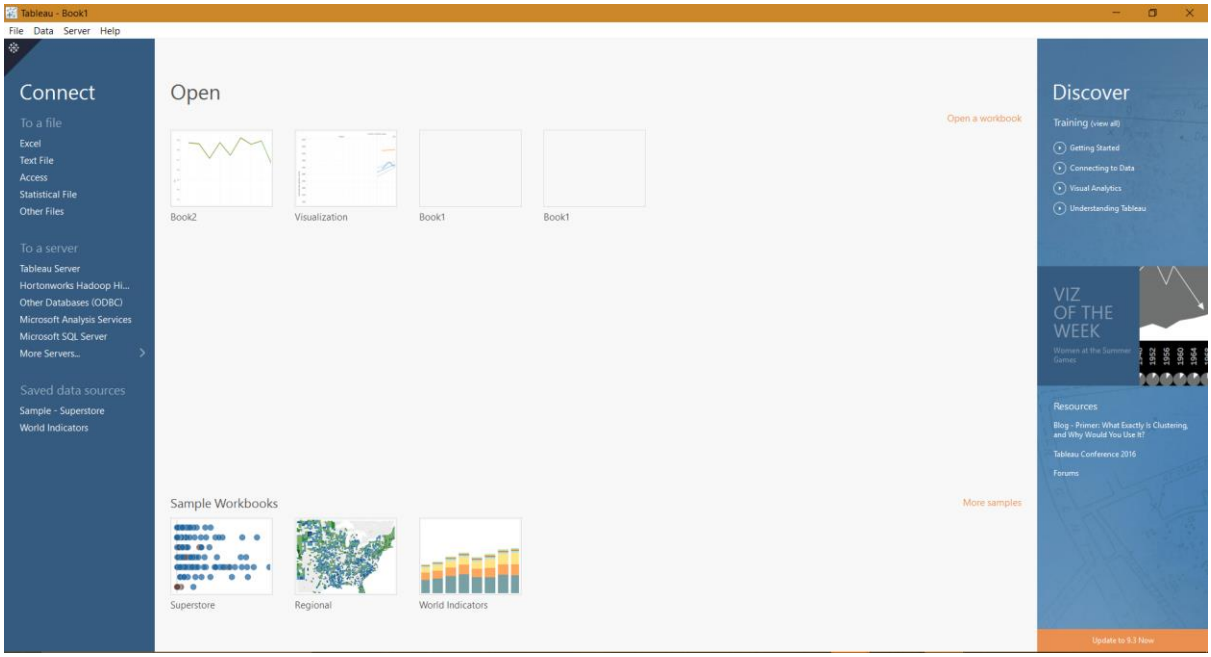
It is a powerful visualization tool used for visualizing the output of the model and the predicted data and comparing with several results of the several model.

Use:

It is used to compare the performance of the three model according to our requirement through the best line chart connected using Tableau to Hive.

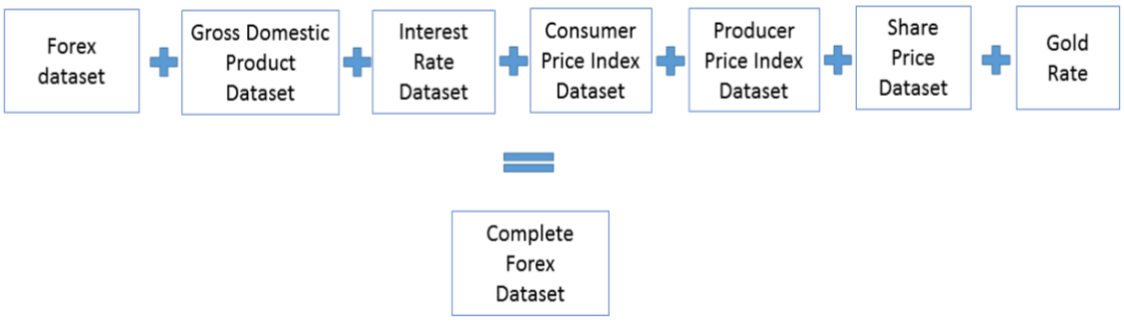
Installation:

1. Download Tableau Desktop or Tableau Reader to your computer from this link http://www.tableau.com/products/desktop .
2. Click the Start button.
3. The application looks like this as shown below



Data Collection

Our model has to predict three countries forex rate against INR, the three Countries are USA, Canada and Great Britain. The also several factors are also considered, the factors include Consumer price index, Producer price index, interest rate of the country, Gross domestic product .



Integration of different dataset into single dataset for data analysis

The dataset are collected from several resources and they are provided below, the dataset for the USA the Exchange Rate against India is collected from http://www.investing.com/currencies/usd-inr-historical-data for the Duration : Jan 2000 - Jun 2016, similarly, the Interest Rate http://data.okfn.org/data/core/bond-yields-us-10y#data the Duration : Jan 1953 - Jun 2014, the Consumer Price Index: http://stats.oecd.org/viewhtml.aspx?datasetcode=MEI_PRICES&lang=en# for the duration from Jan 2000 - Jun 2016. Producer price indices from https://data.oecd.org/price/producer-price-indices-ppi.htm#indicator-chart for the Duration : Jan 2000 - Jun 2016, Share Prices from https://data.oecd.org/price/share-prices.htm#indicator-chart , the Gold http://www.gold.org/research/download-the-gold-price-since-1978. Similarly, the dataset is collected for other three countries and they merged based on several condition.

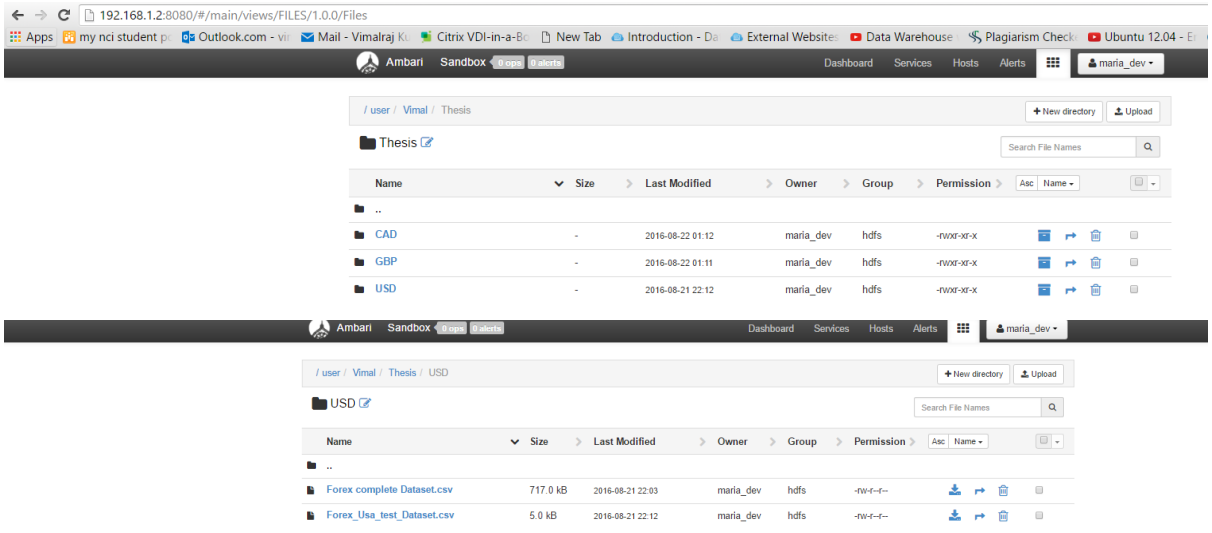
In order to merge them, the 1000 lines of R program is written which are provided along appendix of this report and they are executed which then saves three file in the local system.

```
require(zoo)
# LOading the main foreign exchange rate dataset--
setwd('E:/Study Material/Course Content/Data_Analytics-2015-11-30/Data Analytics/Thesis/Dataset')
forex_all <- read.csv('/Study Material/Course Content/Data_Analytics-2015-11-30/Data Analytics/Thesis/Dataset/ALL Cou
View(forex_all)
head(forex_all)
tail(forex_all)
#-- removing the footer content at the end of the foreign exchange rate dataset
forex_usa <-forex_all[c(1:7862),c(1,2)]
head(forex_usa)
dim(forex_usa)
tail(forex_usa)
#-- Assigning the proper name to the Dataset
colnames(forex_usa)<-c('Date','Exchange_rate')
head(forex_usa)
dim(forex_usa)
tail(forex_usa)
#-- Formating the Date column to exact date format
forex_usa$Date<-as.Date(paste0(substr(forex_usa$Date,7,10),'/',substr(forex_usa$Date,4,5),'/',substr(forex_usa$Date,1

#-- Creating a new field for comparing and merging the datasets
#forex_usa$DatePart <- paste0(format(forex_usa$Date, "%m"),'/',format(forex_usa$Date, "%Y"))

head(forex_usa)
#-- checking min and max date in the dataset
min(forex_usa$Date,na.rm = T)
```

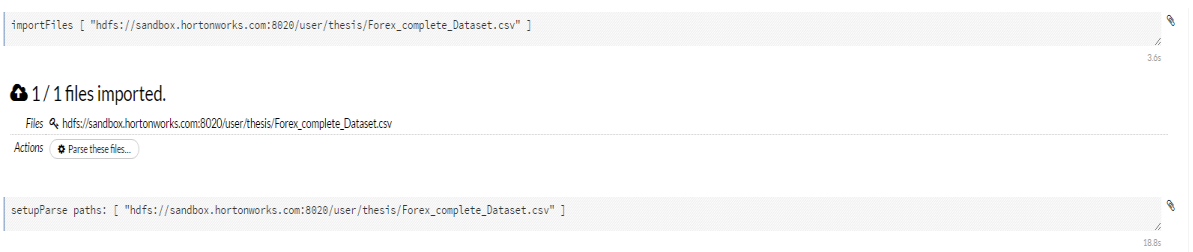
The output of the file are shown in the below diagram,

Forex complete Dataset.csv	21-08-2016 19:43	Microsoft Excel Co...	717 KB
Forex_CAD_complete_Dataset.csv	22-08-2016 00:39	Microsoft Excel Co...	668 KB
Forex_GBD_complete_Dataset.csv	21-08-2016 23:43	Microsoft Excel Co...	670 KB
Forex_Usa_test_Dataset.csv	21-08-2016 19:43	Microsoft Excel Co...	6 KB
Forex_Usa_train_Dataset.csv	21-08-2016 19:43	Microsoft Excel Co...	713 KB
Forex_Usa_Validate_Dataset.csv	21-08-2016 21:49	Microsoft Excel Co...	24 KB

Then these files are transferred using Horton work sandbox HDFS file manager



Once the files are loaded into the HDFS, the next step is doing Data mining in H20 which is discussed below,



The imported data is now parsed by clicking the parse button and then the respective data type are selected

Setup Parse

PARSE CONFIGURATION

Sources hdfs://sandbox.hortonworks.com:8020/user/thesis/Forex_complete_Dataset.csv
ID Forex_complete_Dataset.hex
Parser CSV
Separator ',' 44'
Column Headers Auto
First row contains column names
First row contains data
Options Enable single quotes as a field quotation character
Delete on done

EDIT COLUMN NAMES AND TYPES

Search by column name...

1		Numeric	1	2	3	4	5	6	7	8	9
2	Date	Time	2000-01-01	2000-01-02	2000-01-03	2000-01-04	2000-01-05	2000-01-06	2000-01-07	2000-01-08	2000-01-09
3	month	Numeric	01	01	01	01	01	01	01	01	01
4	year	Numeric	2000	2000	2000	2000	2000	2000	2000	2000	2000
5	Day	Numeric	01	02	03	04	05	06	07	08	09
6	DayofWeek	Enum	Saturday	Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
7	Tuesday	Numeric	0	0	0	1	0	0	0	0	0
8	Monday	Numeric	0	0	1	0	0	0	0	0	0
9	Wednesday	Numeric	0	0	0	0	1	0	0	0	0
10	Friday	Numeric	0	0	0	0	0	0	1	0	0
11	Thursday	Numeric	0	0	0	0	0	1	0	0	0
12	Saturday	Numeric	1	0	0	0	0	0	0	1	0
13	Sunday	Numeric	0	0	0	0	0	0	0	0	0
14	isWeekend	Numeric	1	0	0	0	0	0	0	1	0
15	isWorkDay	Numeric	0	0	1	1	1	1	1	0	0

After the data parse the data looks like this as shown in the below diagram,

Forex_complete_Dataset.hex

Actions: View Data | Split... | Build Model... | Predict | Download | Export

Rows	Columns	Compressed Size
6037	21	235KB

COLUMN SUMMARIES

label	type	Missing	Zeros	+Inf	-Inf	min	max	mean	sigma	cardinality	Actions
C1	int	0	0	0	0	1.0	6037.0	3019.0	1742.8761	·	Convert to enum
Date	time	0	0	0	0	946684800000.0	1468195200000.0	1207440000000.0	150584496693.6504	·	·
month	enum	0	527	0	0	0	11.0	·	·	12	Convert to numeric
year	enum	0	366	0	0	0	16.0	·	·	17	Convert to numeric
Day	enum	0	199	0	0	0	30.0	·	·	31	Convert to numeric
DayofWeek	enum	0	862	0	0	0	6.0	·	·	7	Convert to numeric
Tuesday	enum	0	5175	0	0	0	1.0	0.1428	0.3499	2	Convert to numeric
Monday	enum	0	5174	0	0	0	1.0	0.1430	0.3501	2	Convert to numeric
Wednesday	enum	0	5175	0	0	0	1.0	0.1428	0.3499	2	Convert to numeric
Friday	enum	0	5175	0	0	0	1.0	0.1428	0.3499	2	Convert to numeric
Thursday	enum	0	5175	0	0	0	1.0	0.1428	0.3499	2	Convert to numeric
Saturday	enum	0	5174	0	0	0	1.0	0.1430	0.3501	2	Convert to numeric
Sunday	enum	0	6037	0	0	0	0	0	0	1	Convert to numeric
isWeekend	enum	0	5174	0	0	0	1.0	0.1430	0.3501	2	Convert to numeric
isWorkDay	enum	0	1726	0	0	0	1.0	0.7141	0.4519	2	Convert to numeric
Interest_rate	real	711	0	0	0	1.5300	6.6600	3.8600	1.1615	·	·
Cpi	real	41	0	0	0	77.4115	110.1720	95.3965	10.0301	·	·
PPI	real	41	0	0	0	74.5512	112.5677	93.8427	12.9200	·	·
Share	real	11	0	0	0	65.3644	154.2413	108.3521	23.6125	·	·
Gold	real	3505	0	0	0	256.0	999.5000	505.3970	225.8524	·	·

Previous 20 Columns | Next 20 Columns

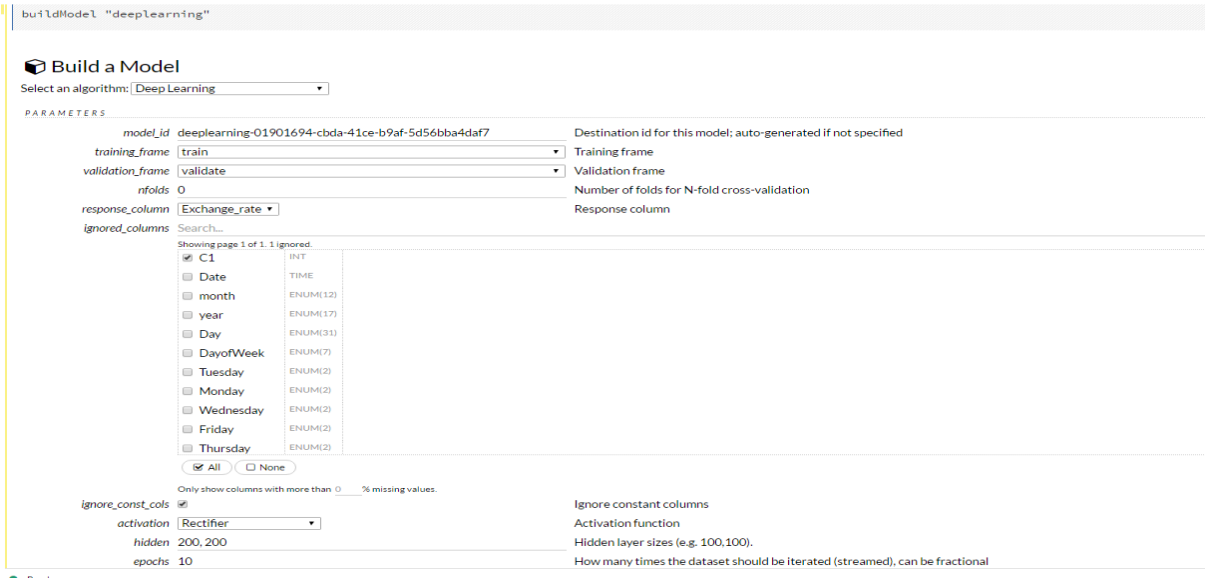
Then the dataset is split into three sets, Train, Test and validate

Split Frames

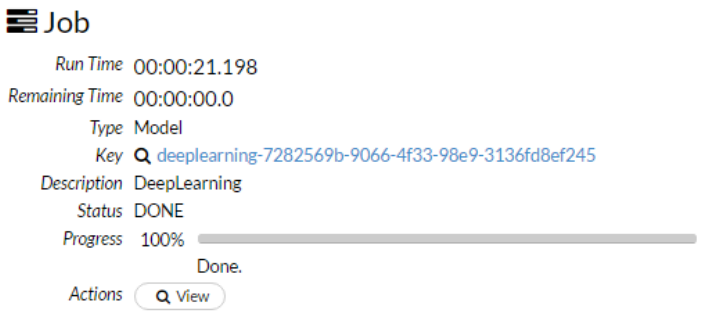
Type	Key
	validate
	test
	train

```
getFrameSummary "train"
```

The deep learning algorithm is selected from the model tab and the configuration are done

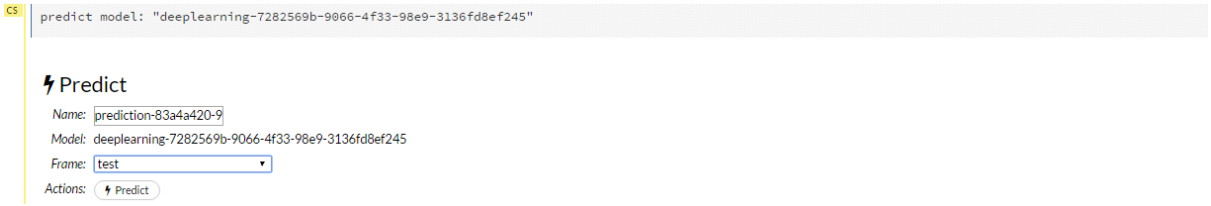


After clicking the run job, the job runs and shows the parameters of the model




```
getModel "deeplearning-7282569b-9066-4f33-98e9-3136fd8ef245"
```

And then the predict dataset of test is assigned to predicted by clicking the predict button of the trained model.



The predicted models performances are seen which is shown below,

Prediction

Actions: Inspect

PREDICTION

model	deeplearning-7282569b-9066-4f33-98e9-3136fd8ef245
model_checksum	6911522010093010944
frame	test
frame_checksum	-5675281573550583808
description	-
model_category	Regression
scoring_time	1471855746409
predictions	prediction-1252785c-5b2b-4132-9ba9-b79903b56823
MSE	0.000001
r2	0.920259
mean_residual_deviance	0.000001

Combine predictions with frame

```
inspect getPrediction model: "deeplearning-7282569b-9066-4f33-98e9-3136fd8ef245", frame: "test"
```

After the successful prediction of the data, the predicted columns get combined test dataset by clicking the combine prediction with frame button at the bottom.

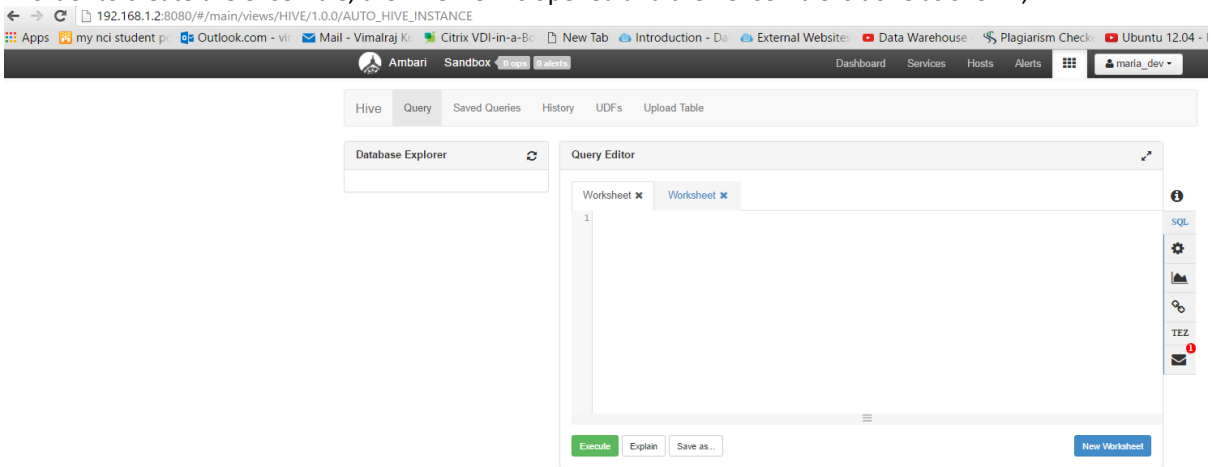
combined-prediction-1252785c-5b2b-4132-9ba9-b79903b56823

DATA

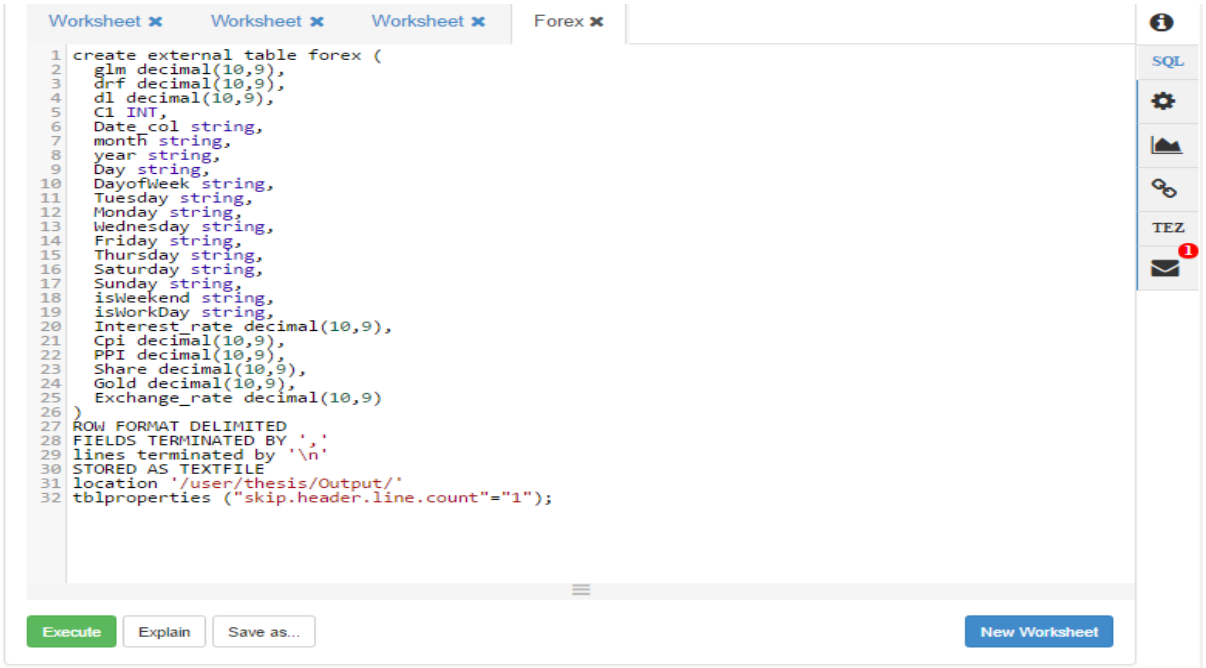
Previous 20 Columns | Next 20 Columns

Row	predict	C1	Date	month	year	Day	DayofWeek	Tuesday	Monday	Wednesday	Friday	Thursday	Saturday	Sunday	isWeekend	isWorkDay	Interest_rate	Cpi	PPI	Share
1	0.0224	1.0	946684800000	01	2000	01	Saturday	0	0	0	0	0	1	0	1	0	6.6600	77.4115	74.5512	92.8727
2	0.0221	7.0	947203200000	01	2000	07	Friday	0	0	0	1	0	0	0	0	1	6.6600	77.4115	74.5512	92.8727
3	0.0224	14.0	947808000000	01	2000	14	Friday	0	0	0	1	0	0	0	0	1	6.6600	77.4115	74.5512	92.8727
4	0.0225	15.0	947894400000	01	2000	15	Saturday	0	0	0	0	0	1	0	1	0	6.6600	77.4115	74.5512	92.8727
5	0.0222	20.0	948326400000	01	2000	20	Thursday	0	0	0	0	1	0	0	0	1	6.6600	77.4115	74.5512	92.8727
6	0.0225	23.0	948585600000	01	2000	23	Sunday	0	0	0	0	0	0	0	0	0	6.6600	77.4115	74.5512	92.8727
7	0.0222	25.0	948758400000	01	2000	25	Tuesday	1	0	0	0	0	0	0	0	1	6.6600	77.4115	74.5512	92.8727
8	0.0222	35.0	949622400000	02	2000	04	Friday	0	0	0	1	0	0	0	0	1	6.5200	77.8701	75.3491	88.5262
9	0.0221	39.0	949968000000	02	2000	08	Tuesday	1	0	0	0	0	0	0	0	1	6.5200	77.8701	75.3491	88.5262
10	0.0223	40.0	950054400000	02	2000	09	Wednesday	0	0	1	0	0	0	0	0	1	6.5200	77.8701	75.3491	88.5262
11	0.0220	47.0	950659200000	02	2000	16	Wednesday	0	0	1	0	0	0	0	0	1	6.5200	77.8701	75.3491	88.5262
12	0.0223	57.0	951523200000	02	2000	26	Saturday	0	0	0	0	0	1	0	1	0	6.5200	77.8701	75.3491	88.5262
13	0.0221	61.0	951868800000	03	2000	01	Wednesday	0	0	1	0	0	0	0	0	1	6.2600	78.5121	75.7481	91.0388
14	0.0222	62.0	951955200000	03	2000	02	Thursday	0	0	0	0	1	0	0	0	1	6.2600	78.5121	75.7481	91.0388
15	0.0222	68.0	952473600000	03	2000	08	Wednesday	0	0	1	0	0	0	0	0	1	6.2600	78.5121	75.7481	91.0388
16	0.0220	81.0	953596800000	03	2000	21	Tuesday	1	0	0	0	0	0	0	0	1	6.2600	78.5121	75.7481	91.0388

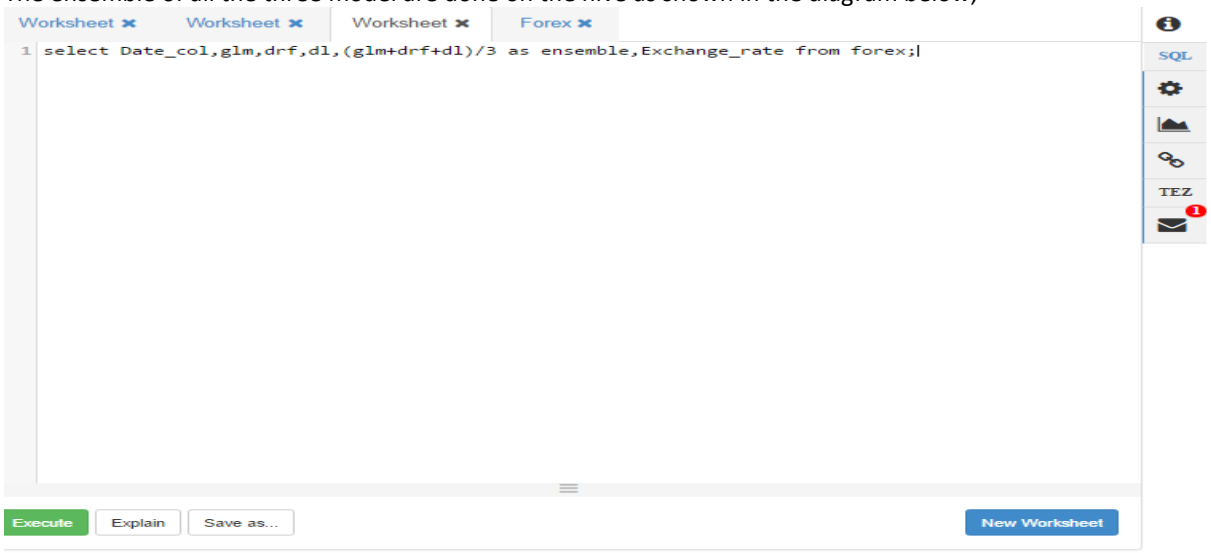
In order to create the ensemble, the hive view is opened and then ensemble is done as shown,



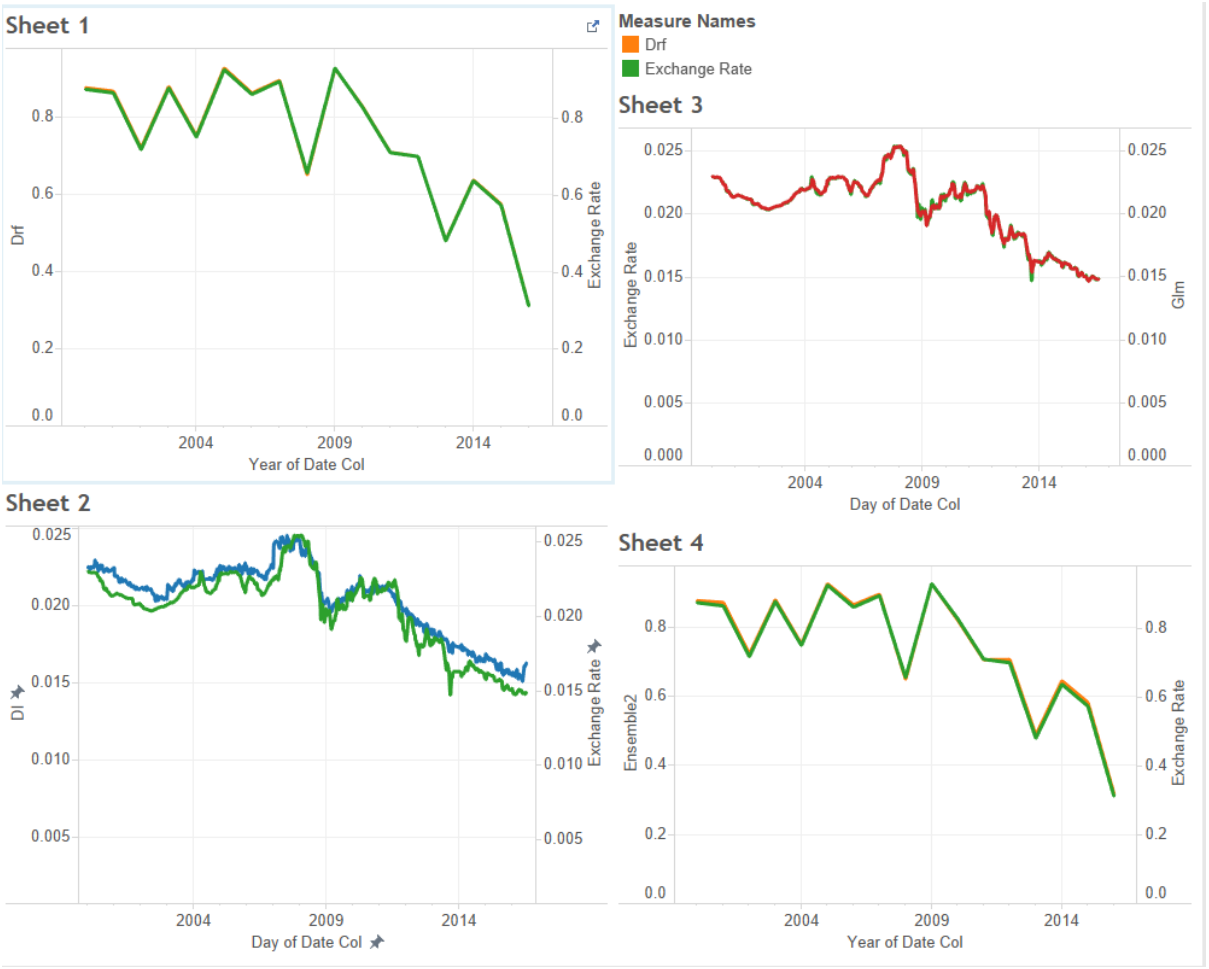
The data after prediction is saved to hdfs and then it is loaded to Hive as shown in the diagram below



The ensemble of all the three model are done on the hive as shown in the diagram below,



Then the tableau is connected to the Hive, and then the results are compared using the line graph as shown in the diagram below,



Conclusion:

The comparison of the predicted values by individual model are done and it is seen clearly that the prediction of the forex using sparkling is highly achieved. And I am sure that by following all these steps you will be also able to predict the forex accurately using sparkling water.

Appendix:
R – Program for Data Extraction, Integration, Cleansing

```
require(zoo)
# LOading the main foreign exchange rate dataset--
setwd('E:/Study Material/Course Content/Data_Analytics-2015-11-30/Data Analytics/Thesis/Dataset')
forex_all <- read.csv('/Study Material/Course Content/Data_Analytics-2015-11-30/Data
Analytics/Thesis/Dataset/ALL Country Dataset/rates_data_USA_GDB_AUD_CAD.csv')
View(forex_all)
head(forex_all)
tail(forex_all)
#-- removing the footer content at the end of the foreign exchange rate dataset
forex_usa <-forex_all[c(1:7862),c(1,2)]
head(forex_usa)
dim(forex_usa)
tail(forex_usa)
#-- Assigning the proper name to the Dataset
colnames(forex_usa)<-c('Date','Exchange_rate')
head(forex_usa)
dim(forex_usa)
tail(forex_usa)
#-- Formating the Date column to exact date format
forex_usa$Date<-
as.Date(paste0(substr(forex_usa$Date,7,10),'/',substr(forex_usa$Date,4,5),'/',substr(forex_usa$Date,1,2)))

#-- Creating a new field for comparing and merging the dataseta
#forex_usa$DatePart <- paste0(format(forex_usa$Date, "%m"),'/',format(forex_usa$Date, "%Y"))

head(forex_usa)
#-- checking min and max date in the dataset
min(forex_usa$Date,na.rm = T)
max(forex_usa$Date,na.rm = T)

#-- considering the data after 2000
forex_usa <- forex_usa[format(forex_usa$Date, "%Y")>= 2000,]

#-- checking min and max date in the dataset after subseting
min(forex_usa$Date,na.rm = T)
max(forex_usa$Date,na.rm = T)

forex_usa$month <- as.factor(format(forex_usa$Date, "%m"))
forex_usa$year <- as.factor(format(forex_usa$Date, "%Y"))

forex_usa<-forex_usa[,c(1,3,4,2)]
head(forex_usa)

#-- Loading the GDP of the USA
gdp_allcountries <-
  read.csv('/Study Material/Course Content/Data_Analytics-2015-11-30/Data Analytics/Thesis/Dataset/ALL
Country Dataset/GDP_ALL_COUNTRIES.csv')
head(gdp_allcountries)


gdp_usa <- gdp_allcountries[gdp_allcountries$LOCATION=='USA',c("TIME","Value")]
head(gdp_usa)
#-- Assigning the proper column name to this GDP dataset
```

```
colnames(gdp_usa)<-c('Time','GDP')

gdp_usa$year = as.factor(substr(gdp_usa$Time,1,4))
gdp_usa$quarters = as.factor(substr(gdp_usa$Time,6,7))
head(gdp_usa)

head(forex_usa)

forex_usa$month <- as.numeric(forex_usa$month)
firstquarter <- c(1:3)
secondquarter <- c(4:6)
thirdquarter <- c(7:9)
fourthquarter <- c(10:12)

forex_usa$quarters <- ifelse(forex_usa$month %in% firstquarter ,"Q1",ifelse(forex_usa$month  %in%
secondquarter,"Q2",ifelse(forex_usa$month  %in% thirdquarter,"Q3","Q4")))

View(forex_usa)
tail(gdp_usa)

merge_forex_usa_gdp <- merge(x=forex_usa,y=gdp_usa,by=c('quarters','year'),all.x =   TRUE)
head(merge_forex_usa_gdp)

merge_forex_usa_gdp$quarters <- NULL
merge_forex_usa_gdp$Time <- NULL

head(merge_forex_usa_gdp)


merge_forex_usa_gdp<-merge_forex_usa_gdp[with(merge_forex_usa_gdp,
order(merge_forex_usa_gdp$Date)), ]



#-- Loading the Interest rate of the USA
interest_rate_usa <-
  read.csv('/Study Material/Course Content/Data_Analytics-2015-11-30/Data
Analytics/Thesis/Dataset/USA/Interest rate.csv')


#-- Assigning the proper column name to this interest rate dataset
colnames(interest_rate_usa)<-c('Date','Interest_rate')

#-- Viewing the Head and tail of the interest rate dataset
head(interest_rate_usa)
tail(interest_rate_usa)

#-- converting the date column for the interest rate dataset
interest_rate_usa$Date=as.Date(paste0(substr(interest_rate_usa$Date,7,10),'/',substr(interest_rate_usa$Date
,4,5),'/',substr(interest_rate_usa$Date,1,2)))

#-- Adding the datepart column for the interest rate dataset for merging
#interest_rate_usa$DatePart <- paste0(format(interest_rate_usa$Date,
"%m"),'/',format(interest_rate_usa$Date, "%Y"))

#-- considering the data after 2000
interest_rate_usa <- interest_rate_usa[format(interest_rate_usa$Date, "%Y") >= 2000,]
```

```
min(interest_rate_usa$Date,na.rm = T)
max(interest_rate_usa$Date,na.rm = T)

head(interest_rate_usa)

interest_rate_usa$month <- as.factor(format(interest_rate_usa$Date, "%m"))
interest_rate_usa$year <- as.factor(format(interest_rate_usa$Date, "%Y"))


interest_rate_usa <- interest_rate_usa[,c(1,3,4,2)]

merge_forex_usa_gdp$month <- as.factor(format(merge_forex_usa_gdp$Date, "%m"))

head(interest_rate_usa)
head(merge_forex_usa_gdp)


merge_forex_usa <- merge(x=merge_forex_usa_gdp,y=interest_rate_usa,by=c('Date'),all.x =   TRUE)

head(merge_forex_usa)


merge_forex_usa$Date.y <- NULL
merge_forex_usa$month.y <- NULL
merge_forex_usa$year.y <- NULL
head(merge_forex_usa)

colnames(merge_forex_usa)<-c('Date','year','month','Exchange_rate',"GDP",'Interest_rate')
merge_forex_usa <- merge_forex_usa[c(3,1,2,5,6,4)]
head(merge_forex_usa,5)
merge_forex_usa <- merge_forex_usa[c(2,1,3,4,5,6)]

head(merge_forex_usa)
merge_forex_usa<-merge_forex_usa[with(merge_forex_usa, order(merge_forex_usa$Date)), ]
View(merge_forex_usa)


#-- Loading the CPI of the United states
Cpi_all <-
  read.csv('/Study Material/Course Content/Data_Analytics-2015-11-30/Data Analytics/Thesis/Dataset/ALL
Country Dataset/CPI FOR CAN_USA_GBD_IND.csv')
head(Cpi_all)
Cpi_US <- Cpi_all[which(Cpi_all$LOCATION=='USA' & Cpi_all$MEASURE=='IXOB' & Cpi_all$Subject=='Consumer
prices - all items'),c('Time','Value')]



Cpi_US$dates <- paste0('01',tolower(substr(Cpi_US$Time,1,3)),"20",substr(Cpi_US$Time,5,6))


dim(Cpi_US)
head(Cpi_US)
Cpi_US$Date <- strptime(Cpi_US$dates, "%d%b%Y")
Cpi_US$Date<- as.Date(Cpi_US$Date)
Cpi_US$Time <- NULL
Cpi_US$dates <- NULL
```

```
head(Cpi_US)
Cpi_US <- Cpi_US[,c(2,1)]

Cpi_US$month <- as.factor(format(Cpi_US$Date, "%m"))
Cpi_US$year <- as.factor(format(Cpi_US$Date, "%Y"))

#Cpi_US$DatePart <- paste0(format(Cpi_US$Date, "%m"),'/',format(Cpi_US$Date, "%Y"))
Cpi_US<-Cpi_US[,c(1,3,4,2)]
head(Cpi_US)
View(Cpi_US)

min(Cpi_US$Date,na.rm = T)
max(Cpi_US$Date,na.rm = T)

head(merge_forex_usa)
head(Cpi_US)

colnames(merge_forex_usa)
colnames(Cpi_US)

#-- merging the CPI with the existing merged dataset

merge_forex_usa_cpi <- merge(x=merge_forex_usa,y=Cpi_US,by=c('month','year'),all.x =TRUE)

merge_forex_usa_cpi <- merge_forex_usa_cpi[,c(1,2,3,4,5,8,6)]

head(merge_forex_usa_cpi)
colnames(merge_forex_usa_cpi) <- c('month','year','Date','GDP','Interest_rate','Cpi','Exchange_rate')


merge_forex_usa_cpi[1:length(merge_forex_usa_cpi)] <- merge_forex_usa_cpi[with(merge_forex_usa_cpi,
order(merge_forex_usa_cpi$Date)), ]
View(merge_forex_usa_cpi)

# -- Loading the product prize index ..

porduct_price_index_all <- read.csv('/Study Material/Course Content/Data_Analytics-2015-11-30/Data
Analytics/Thesis/Dataset/ALL Country Dataset/PPI_FOR_CAN_USA_GBD.csv')

head(porduct_price_index_all)

porduct_price_index_us<-
porduct_price_index_all[porduct_price_index_all$ï..LOCATION=='USA',c('TIME','Value')]
dim(porduct_price_index_us)
porduct_price_index_us$Date <- paste0(porduct_price_index_us$TIME,'-01')
porduct_price_index_us$Date <- as.Date(porduct_price_index_us$Date)
porduct_price_index_us$TIME <- NULL
porduct_price_index_us<- porduct_price_index_us[format(porduct_price_index_us$Date, "%Y") >= 2000,]

porduct_price_index_us$month <- as.factor(format(porduct_price_index_us$Date, "%m"))
porduct_price_index_us$year <- as.factor(format(porduct_price_index_us$Date, "%Y"))

#porduct_price_index_us$DatePart <- paste0(format(porduct_price_index_us$Date,
"%m"),'/',format(porduct_price_index_us$Date, "%Y"))

min(porduct_price_index_us$Date,na.rm = T)
max(porduct_price_index_us$Date,na.rm = T)
```

```
head(porduct_price_index_us)
head(merge_forex_usa_cpi)
colnames(porduct_price_index_us)
colnames(merge_forex_usa_cpi)


merge_forex_usa_ppi <- merge(x=merge_forex_usa_cpi,y=porduct_price_index_us,by=c('month','year'),all.x =
TRUE)

merge_forex_usa_ppi <- merge_forex_usa_ppi[,c(1,2,3,4,5,6,8,7)]
head(merge_forex_usa_ppi)
colnames(merge_forex_usa_ppi) <- c('month','year','Date','GDP','Interest_rate','Cpi','PPI','Exchange_rate')
View(merge_forex_usa_ppi)

merge_forex_usa_ppi[1:length(merge_forex_usa_ppi)] <-merge_forex_usa_ppi[with(merge_forex_usa_ppi,
order(merge_forex_usa_ppi$Date)), ]

# -- Loading the product prize index ..

share_price_all <- read.csv('/Study Material/Course Content/Data_Analytics-2015-11-30/Data
Analytics/Thesis/Dataset/ALL Country Dataset/Share Price CAN GBD USA IND.csv')
head(share_price_all)

share_price_usa <- share_price_all[share_price_all$ï..LOCATION=='USA',c(6,7)]
head(share_price_usa)

dim(share_price_usa)
share_price_usa$Date <- paste0(share_price_usa$TIME,'-01')
share_price_usa$Date <- as.Date(share_price_usa$Date)
share_price_usa$TIME <- NULL
share_price_usa<- share_price_usa[format(share_price_usa$Date, "%Y") >= 2000,]
head(share_price_usa)

share_price_usa$month <- as.factor(format(share_price_usa$Date, "%m"))
share_price_usa$year <- as.factor(format(share_price_usa$Date, "%Y"))

min(share_price_usa$Date,na.rm = T)
max(share_price_usa$Date,na.rm = T)

merge_forex_usa_share <- merge(x=merge_forex_usa_ppi,y=share_price_usa,by=c('month','year'),all.x =
TRUE)

merge_forex_usa_share <- merge_forex_usa_share[,c(1,2,3,4,5,6,7,9,8)]
head(merge_forex_usa_share)
colnames(merge_forex_usa_share) <-
c('month','year','Date','GDP','Interest_rate','Cpi','PPI','Share','Exchange_rate')


merge_forex_usa_share <-
merge_forex_usa_share[with(merge_forex_usa_share,order(merge_forex_usa_share$Date)),]
merge_forex_usa_share$Day <- as.factor(format(merge_forex_usa_share$Date, "%d"))
View(merge_forex_usa_share)

#-- gold prize dataset
```

```
gold_price_all <- read.csv('/Study Material/Course Content/Data_Analytics-2015-11-30/Data
Analytics/Thesis/Dataset/ALL Country Dataset/Gold rate dataset.csv')
head(gold_price_all)
gold_price_usd <- gold_price_all[,c(1,2)]


gold_price_usd$Date<-as.Date(paste0(substr(gold_price_usd$Date,7,10),'/',
                            substr(gold_price_usd$Date,4,5),'/',substr(gold_price_usd$Date,1,2)))

gold_price_usd <- gold_price_usd[format(gold_price_usd$Date, "%Y") >= 2000,]
head(gold_price_usd)

dim(gold_price_usd)
dim(merge_forex_usa_share)
merge_forex_usa_gold <- merge(x=merge_forex_usa_share,y=gold_price_usd,by='Date',all.x =   TRUE)
dim(merge_forex_usa_gold)
head(merge_forex_usa_gold)

merge_forex_usa_gold <- merge_forex_usa_gold[,c(1,2,3,10,4,5,6,7,8,11,9)]

colnames(merge_forex_usa_gold) <-
c('Date','month','year','Day','GDP','Interest_rate','Cpi','PPI','Share','Gold','Exchange_rate')
head(merge_forex_usa_gold)

merge_forex_usa_gold$DayofWeek <- weekdays(as.Date(merge_forex_usa_gold$Date))
merge_forex_usa_gold$DayofWeek <- as.factor(merge_forex_usa_gold$DayofWeek)

levels(merge_forex_usa_gold$DayofWeek)

dim(merge_forex_usa_gold)
View(merge_forex_usa_gold)

library(timeDate)

extractdayOfWeek<-function(x){
  x$Monday<-isWeekday(x$Date, wday=1)
  x$Tuesday<-isWeekday(x$Date, wday=2)
  x$Wednesday<-isWeekday(x$Date, wday=3)
  x$Thursday<-isWeekday(x$Date, wday=4)
  x$Friday<-isWeekday(x$Date, wday=5)
  x$Saturday<-isWeekday(x$Date, wday=6)
  x$Sunday <-isWeekday(x$Date, wday=7)
  x$isWorkDay<-isWeekday(x$Date, wday=1:5)
  x$isWeekend<-isWeekday(x$Date, wday=6:7)
  return(x)
}


merge_forex_usa_gold_new<-data.frame(extractdayOfWeek(merge_forex_usa_gold))

cols <- sapply(merge_forex_usa_gold_new, is.logical)
merge_forex_usa_gold_new[,cols] <- lapply(merge_forex_usa_gold_new[,cols], as.numeric)

head(merge_forex_usa_gold_new)

merge_forex_usa_complete <- merge_forex_usa_gold_new[,c(1,2,3,4,12:21,5:11)]
```

```
head(merge_forex_usa_complete)

na_count <-sapply(merge_forex_usa_complete,
            function(y)
              paste0(round(sum(length(which(is.na(y))))/nrow(merge_forex_usa_complete),3),' %')
)

View(na_count)


min(merge_forex_usa_complete[is.na(merge_forex_usa_complete$GDP),"Date"])
max(merge_forex_usa_complete[is.na(merge_forex_usa_complete$GDP),"Date"])
merge_forex_usa_complete[is.na(merge_forex_usa_complete$GDP),]$GDP <- 0.021331

merge_forex_usa_complete[is.na(merge_forex_usa_complete$Cpi),]
min(merge_forex_usa_complete[is.na(merge_forex_usa_complete$Cpi),"Date"])
max(merge_forex_usa_complete[is.na(merge_forex_usa_complete$Cpi),"Date"])
merge_forex_usa_complete[is.na(merge_forex_usa_complete$Cpi),]$Cpi <-
max(merge_forex_usa_complete[merge_forex_usa_complete$year==2016 &
merge_forex_usa_complete$month=="05",]$Cpi)

merge_forex_usa_complete[is.na(merge_forex_usa_complete$PPI),]
min(merge_forex_usa_complete[is.na(merge_forex_usa_complete$PPI),"Date"])
max(merge_forex_usa_complete[is.na(merge_forex_usa_complete$PPI),"Date"])
merge_forex_usa_complete[is.na(merge_forex_usa_complete$PPI),]$PPI <-
max(merge_forex_usa_complete[merge_forex_usa_complete$year==2016 &
merge_forex_usa_complete$month=="05",]$PPI)

merge_forex_usa_complete[is.na(merge_forex_usa_complete$Share),]$Share <-
max(merge_forex_usa_complete[merge_forex_usa_complete$year==2016 &
merge_forex_usa_complete$month=="06",]$Share)


merge_forex_usa_complete$Gold <- as.numeric(merge_forex_usa_complete$Gold)

merge_forex_usa_complete[is.na(merge_forex_usa_complete$Gold),]$Gold <-
mean(merge_forex_usa_complete[!is.na(merge_forex_usa_complete$Gold) &
merge_forex_usa_complete$Gold!='<NA>',]$Gold)

forex_usa <- NULL
interest_rate_usa <- NULL
merge_forex_usa <- NULL
Cpi_US <- NULL
merge_forex_usa_cpi <- NULL
porduct_price_index_us <- NULL
merge_forex_usa_ppi <- NULL
share_price_usa <- NULL
merge_forex_usa_share <- NULL
gold_price_usd <- NULL
merge_forex_usa_gold <- NULL
merge_forex_usa_gold_new <- NULL

head(merge_forex_usa_complete)

min(merge_forex_usa_complete$Date)
max(merge_forex_usa_complete$Date)
```

```
Usa_traindata <- merge_forex_usa_complete[as.Date(merge_forex_usa_complete$Date) < as.Date("2016-06-01")  ,]
min(Usa_traindata$Date)
max(Usa_traindata$Date)
set.seed(1000)

Usa_validatedata <- merge_forex_usa_complete[format(merge_forex_usa_complete$Date, "%Y") == 2016, ]


Usa_testdata <- merge_forex_usa_complete[as.Date(merge_forex_usa_complete$Date) > as.Date("2016-05-31")  ,]

dim(Usa_traindata)
min(Usa_traindata$Date)
max(Usa_traindata$Date)
dim(Usa_validatedata)
dim(Usa_testdata)
min(Usa_testdata$Date)
max(Usa_testdata$Date)

getwd()
write.csv(file='Forex complete Dataset.csv', x=merge_forex_usa_complete)

write.csv(file='Forex_Usa_train_Dataset.csv', x=Usa_traindata)
write.csv(file='Forex_Usa_Validate_Dataset.csv', x=Usa_validatedata)
write.csv(file='Forex_Usa_test_Dataset.csv', x=Usa_testdata)



require(zoo)
# LOading the main foreign exchange rate dataset--
setwd('E:/Study Material/Course Content/Data_Analytics-2015-11-30/Data Analytics/Thesis/Dataset')
forex_all <- read.csv('/Study Material/Course Content/Data_Analytics-2015-11-30/Data Analytics/Thesis/Dataset/ALL Country Dataset/rates_data_USA_GDB_AUD_CAD.csv')
View(forex_all)
head(forex_all)
tail(forex_all)
#-- removing the footer content at the end of the foreign exchange rate dataset
forex_gbd <-forex_all[c(1:7862),c(1,3)]
head(forex_gbd)
dim(forex_gbd)
tail(forex_gbd)
#-- Assigning the proper name to the Dataset
colnames(forex_gbd)<-c('Date','Exchange_rate')
head(forex_gbd)
dim(forex_gbd)
tail(forex_gbd)
#-- Formating the Date column to exact date format
forex_gbd$Date<-
as.Date(paste0(substr(forex_gbd$Date,7,10),'/',substr(forex_gbd$Date,4,5),'/',substr(forex_gbd$Date,1,2)))

#-- Creating a new field for comparing and merging the dataseta

head(forex_gbd)
#-- checking min and max date in the dataset
```

```
min(forex_gbd$Date,na.rm = T)
max(forex_gbd$Date,na.rm = T)

#-- considering the data after 2000
forex_gbd <- forex_gbd[format(forex_gbd$Date, "%Y")>= 2000,]

#-- checking min and max date in the dataset after subseting
min(forex_gbd$Date,na.rm = T)
max(forex_gbd$Date,na.rm = T)

forex_gbd$month <- as.factor(format(forex_gbd$Date, "%m"))
forex_gbd$year <- as.factor(format(forex_gbd$Date, "%Y"))

forex_gbd<-forex_gbd[,c(1,3,4,2)]
head(forex_gbd)


#-- Loading the Interest rate of the GBR
interest_rate_gbd <-
  read.csv('/Study Material/Course Content/Data_Analytics-2015-11-30/Data
Analytics/Thesis/Dataset/GBD/Interest Rate 1975 - 2016.csv')
head(interest_rate_gbd)

#-- Assigning the proper column name to this interest rate dataset
colnames(interest_rate_gbd)<-c('Date','Interest_rate')

#-- Viewing the Head and tail of the interest rate dataset
head(interest_rate_gbd)
View(interest_rate_gbd)
tail(interest_rate_gbd)

interest_rate_gbd$Date<-

as.Date(paste0(substr(interest_rate_gbd$Date,7,10),'/',substr(interest_rate_gbd$Date,4,5),'/',substr(interest_
rate_gbd$Date,1,2)))


#-- converting the date column for the interest rate dataset
interest_rate_gbd$Date=as.Date(interest_rate_gbd$Date)

#-- Adding the datepart column for the interest rate dataset for merging

#-- considering the data after 2000
interest_rate_gbd <- interest_rate_gbd[format(interest_rate_gbd$Date, "%Y") >= 2000,]
head(interest_rate_gbd)

min(interest_rate_gbd$Date,na.rm = T)
max(interest_rate_gbd$Date,na.rm = T)


interest_rate_gbd$month <- as.factor(format(interest_rate_gbd$Date, "%m"))
interest_rate_gbd$year <- as.factor(format(interest_rate_gbd$Date, "%Y"))


interest_rate_gbd <- interest_rate_gbd[,c(1,3,4,2)]
head(interest_rate_gbd)
head(forex_gbd)
```

```
merge_forex_gbd <- merge(x=forex_gbd,y=interest_rate_gbd,by=c('Date'),all.x =   TRUE)

head(merge_forex_gbd)

merge_forex_gbd<-merge_forex_gbd[,c(1,2,3,7,4)]

head(merge_forex_gbd)

colnames(merge_forex_gbd)<-c('Date','month','year','Interest_rate','Exchange_rate')
head(merge_forex_gbd,15)
View(merge_forex_gbd)

#-- Loading the CPI of the United states
Cpi_all <-
  read.csv('/Study Material/Course Content/Data_Analytics-2015-11-30/Data Analytics/Thesis/Dataset/ALL
Country Dataset/CPI FOR CAN_USA_GBD_IND.csv')
head(Cpi_all)
Cpi_GBD <- Cpi_all[which(Cpi_all$LOCATION=='GBR' & Cpi_all$MEASURE=='IXOB' &
Cpi_all$Subject=='Consumer prices - all items'),c('Time','Value')]
head(Cpi_GBD)


Cpi_GBD$dates <- paste0('01',tolower(substr(Cpi_GBD$Time,1,3)),"20",substr(Cpi_GBD$Time,5,6))



dim(Cpi_GBD)
head(Cpi_GBD)
Cpi_GBD$Date <- strptime(Cpi_GBD$dates, "%d%b%Y")
Cpi_GBD$Date<- as.Date(Cpi_GBD$Date)
Cpi_GBD$Time <- NULL
Cpi_GBD$dates <- NULL
head(Cpi_GBD)
Cpi_GBD <- Cpi_GBD[,c(2,1)]

Cpi_GBD$month <- as.factor(format(Cpi_GBD$Date, "%m"))
Cpi_GBD$year <- as.factor(format(Cpi_GBD$Date, "%Y"))

#Cpi_US$DatePart <- paste0(format(Cpi_US$Date, "%m"),'/',format(Cpi_US$Date, "%Y"))
Cpi_GBD<-Cpi_GBD[,c(1,3,4,2)]
head(Cpi_GBD)
View(Cpi_GBD)

min(Cpi_GBD$Date,na.rm = T)
max(Cpi_GBD$Date,na.rm = T)

head(merge_forex_gbd)
head(Cpi_GBD)

colnames(merge_forex_gbd)
colnames(Cpi_GBD)

#-- merging the CPI with the existing merged dataset

merge_forex_gbd_cpi <- merge(x=merge_forex_gbd,y=Cpi_GBD,by=c('month','year'),all.x =TRUE)
```

```
merge_forex_gbd_cpi <- merge_forex_gbd_cpi[,c(1,2,3,4,7,5)]

head(merge_forex_gbd_cpi)
colnames(merge_forex_gbd_cpi) <- c('month','year','Date','Interest_rate','Cpi','Exchange_rate')


merge_forex_gbd_cpi[1:length(merge_forex_gbd_cpi)] <- merge_forex_gbd_cpi[with(merge_forex_gbd_cpi,
order(merge_forex_gbd_cpi$Date)), ]
View(merge_forex_gbd_cpi)

# -- Loading the product prize index ..
porduct_price_index_all <- read.csv('/Study Material/Course Content/Data_Analytics-2015-11-30/Data
Analytics/Thesis/Dataset/ALL Country Dataset/PPI_FOR_CAN_USA_GBD.csv')

head(porduct_price_index_all)

porduct_price_index_gbr<-
porduct_price_index_all[porduct_price_index_all$ï..LOCATION=='GBR',c('TIME','Value')]
dim(porduct_price_index_gbr)
porduct_price_index_gbr$Date <- paste0(porduct_price_index_gbr$TIME,'-01')
porduct_price_index_gbr$Date <- as.Date(porduct_price_index_gbr$Date)
porduct_price_index_gbr$TIME <- NULL
porduct_price_index_gbr<- porduct_price_index_gbr[format(porduct_price_index_gbr$Date, "%Y") >= 2000,]


porduct_price_index_gbr$month <- as.factor(format(porduct_price_index_gbr$Date, "%m"))
porduct_price_index_gbr$year <- as.factor(format(porduct_price_index_gbr$Date, "%Y"))

#porduct_price_index_us$DatePart <- paste0(format(porduct_price_index_us$Date,
"%m"),'/',format(porduct_price_index_us$Date, "%Y"))

min(porduct_price_index_gbr$Date,na.rm = T)
max(porduct_price_index_gbr$Date,na.rm = T)


head(porduct_price_index_gbr)
head(merge_forex_gbd_cpi)
colnames(porduct_price_index_gbr)
colnames(merge_forex_gbd_cpi)


merge_forex_gbd_ppi <- merge(x=merge_forex_gbd_cpi,y=porduct_price_index_gbr,by=c('month','year'),all.x
=  TRUE)

merge_forex_gbd_ppi <- merge_forex_gbd_ppi[,c(1,2,3,4,5,7,6)]
head(merge_forex_gbd_ppi)
colnames(merge_forex_gbd_ppi) <- c('month','year','Date','Interest_rate','Cpi','PPI','Exchange_rate')
View(merge_forex_gbd_ppi)

merge_forex_gbd_ppi <-
merge_forex_gbd_ppi[with(merge_forex_gbd_ppi,order(merge_forex_gbd_ppi$Date)),]
# -- Loading the share prize index ..

share_price_all <- read.csv('/Study Material/Course Content/Data_Analytics-2015-11-30/Data
Analytics/Thesis/Dataset/ALL Country Dataset/Share Price CAN GBD USA IND.csv')
head(share_price_all)
```

```
share_price_gbd <- share_price_all[share_price_all$ï..LOCATION=='GBR',c(6,7)]
head(share_price_gbd)

dim(share_price_gbd)
share_price_gbd$Date <- paste0(share_price_gbd$TIME,'-01')
share_price_gbd$Date <- as.Date(share_price_gbd$Date)
share_price_gbd$TIME <- NULL
share_price_gbd<- share_price_gbd[format(share_price_gbd$Date, "%Y") >= 2000,]
head(share_price_gbd)

share_price_gbd$month <- as.factor(format(share_price_gbd$Date, "%m"))
share_price_gbd$year <- as.factor(format(share_price_gbd$Date, "%Y"))

min(share_price_gbd$Date,na.rm = T)
max(share_price_gbd$Date,na.rm = T)

merge_forex_gbd_share <- merge(x=merge_forex_gbd_ppi,y=share_price_gbd,by=c('month','year'),all.x =
TRUE)

merge_forex_gbd_share <- merge_forex_gbd_share[,c(1,2,3,4,5,6,8,7)]
head(merge_forex_gbd_share)
colnames(merge_forex_gbd_share) <- c('month','year','Date','Interest_rate','Cpi','PPI','Share','Exchange_rate')


merge_forex_gbd_share <-
merge_forex_gbd_share[with(merge_forex_gbd_share,order(merge_forex_gbd_share$Date)),]
merge_forex_gbd_share$Day <- as.factor(format(merge_forex_gbd_share$Date, "%d"))
View(merge_forex_gbd_share)

#-- gold prize dataset

gold_price_all <- read.csv('/Study Material/Course Content/Data_Analytics-2015-11-30/Data
Analytics/Thesis/Dataset/ALL Country Dataset/Gold rate dataset.csv')
head(gold_price_all)
gold_price_gbd <- gold_price_all[,c(1,3)]


gold_price_gbd$Date<-as.Date(paste0(substr(gold_price_gbd$Date,7,10),'/',
                        substr(gold_price_gbd$Date,4,5),'/',substr(gold_price_gbd$Date,1,2)))

gold_price_gbd <- gold_price_gbd[format(gold_price_gbd$Date, "%Y") >= 2000,]
head(gold_price_gbd)

dim(gold_price_gbd)
dim(merge_forex_gbd_share)
merge_forex_gbd_gold <- merge(x=merge_forex_gbd_share,y=gold_price_gbd,by='Date',all.x =   TRUE)
dim(merge_forex_gbd_gold)
head(merge_forex_gbd_gold)

merge_forex_gbd_gold <- merge_forex_gbd_gold[,c(1,2,3,9,4,5,6,7,10,8)]

colnames(merge_forex_gbd_gold) <-
c('Date','month','year','Day','Interest_rate','Cpi','PPI','Share','Gold','Exchange_rate')
head(merge_forex_gbd_gold)

merge_forex_gbd_gold$DayofWeek <- weekdays(as.Date(merge_forex_gbd_gold$Date))
merge_forex_gbd_gold$DayofWeek <- as.factor(merge_forex_gbd_gold$DayofWeek)
```

```
levels(merge_forex_gbd_gold$DayofWeek)

dim(merge_forex_gbd_gold)
View(merge_forex_gbd_gold)



#-- Loading the GDP of the GBR
gdp_allcountries <-
  read.csv('/Study Material/Course Content/Data_Analytics-2015-11-30/Data Analytics/Thesis/Dataset/ALL
Country Dataset/GDP_ALL_COUNTRIES.csv')
head(gdp_allcountries)


gdp_gbr<- gdp_allcountries[gdp_allcountries$LOCATION=='GBR',c("TIME","Value")]
head(gdp_gbr)
#-- Assigning the proper column name to this GDP dataset
colnames(gdp_gbr)<-c('Time','GDP')

gdp_gbr$year = as.factor(substr(gdp_gbr$Time,1,4))
gdp_gbr$quarters = as.factor(substr(gdp_gbr$Time,6,7))
head(gdp_gbr)

head(merge_forex_gbd_gold)

merge_forex_gbd_gold$month <- as.numeric(merge_forex_gbd_gold$month)
firstquarter <- c(1:3)
secondquarter <- c(4:6)
thirdquarter <- c(7:9)
fourthquarter <- c(10:12)

merge_forex_gbd_gold$quarters <- ifelse(merge_forex_gbd_gold$month %in% firstquarter
,"Q1",ifelse(merge_forex_gbd_gold$month  %in% secondquarter,"Q2",
                                                    ifelse(merge_forex_gbd_gold$month  %in%
thirdquarter,"Q3","Q4")))

View(merge_forex_gbd_gold)

merge_forex_gbd_gold <- merge(x=merge_forex_gbd_gold,y=gdp_gbr,by=c('quarters','year'),all.x =   TRUE)
head(merge_forex_gbd_gold)

merge_forex_gbd_gold$quarters <- NULL
merge_forex_gbd_gold$Time <- NULL

head(merge_forex_gbd_gold)


merge_forex_gbd_gold<-merge_forex_gbd_gold[with(merge_forex_gbd_gold,
order(merge_forex_gbd_gold$Date)), ]
```

```
#-----------------------------------------------------------------------------------------------------------------------------


library(timeDate)

extractdayOfWeek<-function(x){
 x$Monday<-isWeekday(x$Date, wday=1)
 x$Tuesday<-isWeekday(x$Date, wday=2)
 x$Wednesday<-isWeekday(x$Date, wday=3)
 x$Thursday<-isWeekday(x$Date, wday=4)
 x$Friday<-isWeekday(x$Date, wday=5)
 x$Saturday<-isWeekday(x$Date, wday=6)
 x$Sunday <-isWeekday(x$Date, wday=7)
 x$isWorkDay<-isWeekday(x$Date, wday=1:5)
 x$isWeekend<-isWeekday(x$Date, wday=6:7)
 return(x)
}


merge_forex_gbd_gold_new<-data.frame(extractdayOfWeek(merge_forex_gbd_gold))

cols <- sapply(merge_forex_gbd_gold_new, is.logical)
merge_forex_gbd_gold_new[,cols] <- lapply(merge_forex_gbd_gold_new[,cols], as.numeric)

head(merge_forex_gbd_gold_new)

merge_forex_gbd_complete <- merge_forex_gbd_gold_new[,c(1,2,3,4,11,13:21,5,6,7,8,9,12,10)]


head(merge_forex_gbd_complete)

na_count <-sapply(merge_forex_gbd_complete,
            function(y)
              paste0(round(sum(length(which(is.na(y))))/nrow(merge_forex_gbd_complete),3),' %')
)

View(na_count)


forex_gbd <- NULL
interest_rate_gbd <- NULL
merge_forex_gbd <- NULL
Cpi_GBD <- NULL
merge_forex_gbd_cpi <- NULL
porduct_price_index_gbr <- NULL
merge_forex_gbd_ppi <- NULL
share_price_gbd <- NULL
merge_forex_gbd_share <- NULL
gold_price_gbd <- NULL
merge_forex_gbd_gold <- NULL
merge_forex_gbd_gold_new <- NULL

head(merge_forex_gbd_complete)
View(merge_forex_gbd_complete)

merge_forex_gbd_complete[as.Date(merge_forex_gbd_complete$Date) ==as.Date("2016-06-01"),]$GDP
```

```
merge_forex_gbd_complete[as.Date(merge_forex_gbd_complete$Date) ==as.Date("2016-05-11"),]$Cpi

min(merge_forex_gbd_complete[is.na(merge_forex_gbd_complete$GDP),"Date"])
max(merge_forex_gbd_complete[is.na(merge_forex_gbd_complete$GDP),"Date"])
merge_forex_gbd_complete[is.na(merge_forex_gbd_complete$GDP),]$GDP <- 0.599941

min(merge_forex_gbd_complete[is.na(merge_forex_gbd_complete$Interest_rate),"Date"])
max(merge_forex_gbd_complete[is.na(merge_forex_gbd_complete$Interest_rate),"Date"])
merge_forex_gbd_complete[is.na(merge_forex_gbd_complete$Interest_rate),]$Interest_rate <-
mean(merge_forex_gbd_complete$Interest_rate,na.rm = T)


merge_forex_gbd_complete[is.na(merge_forex_gbd_complete$Cpi),]
min(merge_forex_gbd_complete[is.na(merge_forex_gbd_complete$Cpi),"Date"])
max(merge_forex_gbd_complete[is.na(merge_forex_gbd_complete$Cpi),"Date"])
merge_forex_gbd_complete[is.na(merge_forex_gbd_complete$Cpi),]$Cpi <- 112.2624

merge_forex_gbd_complete[is.na(merge_forex_gbd_complete$PPI),]
min(merge_forex_gbd_complete[is.na(merge_forex_gbd_complete$PPI),"Date"])
max(merge_forex_gbd_complete[is.na(merge_forex_gbd_complete$PPI),"Date"])
merge_forex_gbd_complete$PPI <- NULL

merge_forex_gbd_complete[is.na(merge_forex_gbd_complete$Share),]
min(merge_forex_gbd_complete[is.na(merge_forex_gbd_complete$Share),]$Date)
max(merge_forex_gbd_complete[is.na(merge_forex_gbd_complete$Share),]$Date)
merge_forex_gbd_complete[is.na(merge_forex_gbd_complete$Share),]$Share <-
mean(merge_forex_gbd_complete$Share,na.rm = T)


merge_forex_gbd_complete$Gold <- as.numeric(merge_forex_gbd_complete$Gold)
min(merge_forex_gbd_complete$Gold,na.rm = T)
hist(merge_forex_gbd_complete$Gold)
min(merge_forex_gbd_complete[is.na(merge_forex_gbd_complete$Gold),]$Date)
max(merge_forex_gbd_complete[is.na(merge_forex_gbd_complete$Gold),]$Date)
merge_forex_gbd_complete[is.na(merge_forex_gbd_complete$Gold),]$Gold <-
mean(merge_forex_gbd_complete[!is.na(merge_forex_gbd_complete$Gold) &
merge_forex_gbd_complete$Gold!='<NA>',]$Gold)


getwd()

write.csv(file='Forex_GBD_complete_Dataset.csv', x=merge_forex_gbd_complete)

require(zoo)
# LOading the main foreign exchange rate dataset--
setwd('E:/Study Material/Course Content/Data_Analytics-2015-11-30/Data Analytics/Thesis/Dataset')
forex_all <- read.csv('/Study Material/Course Content/Data_Analytics-2015-11-30/Data
Analytics/Thesis/Dataset/ALL Country Dataset/rates_data_USA_GDB_AUD_CAD.csv')
View(forex_all)
head(forex_all)
tail(forex_all)
#-- removing the footer content at the end of the foreign exchange rate dataset
forex_cad <-forex_all[c(1:7862),c(1,4)]
head(forex_cad)
dim(forex_cad)
tail(forex_cad)
#-- Assigning the proper name to the Dataset
```

```
colnames(forex_cad)<-c('Date','Exchange_rate')
head(forex_cad)
dim(forex_cad)
tail(forex_cad)
#-- Formating the Date column to exact date format
forex_cad$Date<-
as.Date(paste0(substr(forex_cad$Date,7,10),'/',substr(forex_cad$Date,4,5),'/',substr(forex_cad$Date,1,2)))

#-- Creating a new field for comparing and merging the dataseta

head(forex_cad)
#-- checking min and max date in the dataset
min(forex_cad$Date,na.rm = T)
max(forex_cad$Date,na.rm = T)

#-- considering the data after 2000
forex_cad <- forex_cad[format(forex_cad$Date, "%Y")>= 2000,]

#-- checking min and max date in the dataset after subseting
min(forex_cad$Date,na.rm = T)
max(forex_cad$Date,na.rm = T)

forex_cad$month <- as.factor(format(forex_cad$Date, "%m"))
forex_cad$year <- as.factor(format(forex_cad$Date, "%Y"))

forex_cad<-forex_cad[,c(1,3,4,2)]
head(forex_cad)


#-- Loading the Interest rate of the USA
interest_rate_cad <-
  read.csv('/Study Material/Course Content/Data_Analytics-2015-11-30/Data
Analytics/Thesis/Dataset/CAD/interestrate-2006.csv')
head(interest_rate_cad)

#-- Assigning the proper column name to this interest rate dataset
colnames(interest_rate_cad)<-c('Date','Interest_rate')

#-- Viewing the Head and tail of the interest rate dataset
head(interest_rate_cad)
View(interest_rate_cad)
tail(interest_rate_cad)

interest_rate_cad$Date<-

as.Date(paste0(substr(interest_rate_cad$Date,7,10),'/',substr(interest_rate_cad$Date,4,5),'/',substr(interest_r
ate_cad$Date,1,2)))


#-- converting the date column for the interest rate dataset
interest_rate_cad$Date=as.Date(interest_rate_cad$Date)

#-- considering the data after 2000
interest_rate_cad <- interest_rate_cad[format(interest_rate_cad$Date, "%Y") >= 2000,]
head(interest_rate_cad)

min(interest_rate_cad$Date,na.rm = T)
```

```
max(interest_rate_cad$Date,na.rm = T)


interest_rate_cad$month <- as.factor(format(interest_rate_cad$Date, "%m"))
interest_rate_cad$year <- as.factor(format(interest_rate_cad$Date, "%Y"))


interest_rate_cad <- interest_rate_cad[,c(1,3,4,2)]
head(interest_rate_cad)
head(forex_cad)

merge_forex_cad <- merge(x=forex_cad,y=interest_rate_cad,by=c('Date'),all.x =   TRUE)

head(merge_forex_cad)

merge_forex_cad<-merge_forex_cad[,c(1,2,3,7,4)]

head(merge_forex_cad)

colnames(merge_forex_cad)<-c('Date','month','year','Interest_rate','Exchange_rate')
head(merge_forex_cad,15)
View(merge_forex_cad)

#-- Loading the CPI of the United states
Cpi_all <-
  read.csv('/Study Material/Course Content/Data_Analytics-2015-11-30/Data Analytics/Thesis/Dataset/ALL
Country Dataset/CPI FOR CAN_USA_GBD_IND.csv')
head(Cpi_all)

Cpi_cad <- Cpi_all[which(Cpi_all$LOCATION=='CAN' & Cpi_all$MEASURE=='IXOB' &
Cpi_all$Subject=='Consumer prices - all items'),c('Time','Value')]
head(Cpi_cad)


Cpi_cad$dates <- paste0('01',tolower(substr(Cpi_cad$Time,1,3)),"20",substr(Cpi_cad$Time,5,6))



dim(Cpi_cad)
head(Cpi_cad)
Cpi_cad$Date <- strptime(Cpi_cad$dates, "%d%b%Y")
Cpi_cad$Date<- as.Date(Cpi_cad$Date)
Cpi_cad$Time <- NULL
Cpi_cad$dates <- NULL
head(Cpi_cad)
Cpi_cad <- Cpi_cad[,c(2,1)]

Cpi_cad$month <- as.factor(format(Cpi_cad$Date, "%m"))
Cpi_cad$year <- as.factor(format(Cpi_cad$Date, "%Y"))


Cpi_cad<-Cpi_cad[,c(1,3,4,2)]
head(Cpi_cad)
View(Cpi_cad)

min(Cpi_cad$Date,na.rm = T)
max(Cpi_cad$Date,na.rm = T)
```

```
head(merge_forex_cad)
head(Cpi_cad)

#-- merging the CPI with the existing merged dataset

merge_forex_cad_cpi <- merge(x=merge_forex_cad,y=Cpi_cad,by=c('month','year'),all.x =TRUE)

merge_forex_cad_cpi <- merge_forex_cad_cpi[,c(1,2,3,4,7,5)]

head(merge_forex_cad_cpi)
colnames(merge_forex_cad_cpi) <- c('month','year','Date','Interest_rate','Cpi','Exchange_rate')


merge_forex_cad_cpi[1:length(merge_forex_cad_cpi)] <- merge_forex_cad_cpi[with(merge_forex_cad_cpi,
order(merge_forex_cad_cpi$Date)), ]
View(merge_forex_cad_cpi)

# -- Loading the product prize index ..
porduct_price_index_all <- read.csv('/Study Material/Course Content/Data_Analytics-2015-11-30/Data
Analytics/Thesis/Dataset/ALL Country Dataset/PPI_FOR_CAN_USA_GBD.csv')

head(porduct_price_index_all)

porduct_price_index_can <-
porduct_price_index_all[porduct_price_index_all$ï..LOCATION=='CAN',c('TIME','Value')]

dim(porduct_price_index_can)
porduct_price_index_can$Date <- paste0(porduct_price_index_can$TIME,'-01')
porduct_price_index_can$Date <- as.Date(porduct_price_index_can$Date)
porduct_price_index_can$TIME <- NULL
porduct_price_index_can<- porduct_price_index_can[format(porduct_price_index_can$Date, "%Y") >= 2000,]


porduct_price_index_can$month <- as.factor(format(porduct_price_index_can$Date, "%m"))
porduct_price_index_can$year <- as.factor(format(porduct_price_index_can$Date, "%Y"))

min(porduct_price_index_can$Date,na.rm = T)
max(porduct_price_index_can$Date,na.rm = T)


head(porduct_price_index_can)
head(merge_forex_cad_cpi)

merge_forex_cad_ppi <- merge(x=merge_forex_cad_cpi,y=porduct_price_index_can,by=c('month','year'),all.x
=   TRUE)

merge_forex_cad_ppi <- merge_forex_cad_ppi[,c(1,2,3,4,5,7,6)]
head(merge_forex_cad_ppi)
colnames(merge_forex_cad_ppi) <- c('month','year','Date','Interest_rate','Cpi','PPI','Exchange_rate')
View(merge_forex_cad_ppi)

# -- Loading the share prize index ..

share_price_all <- read.csv('/Study Material/Course Content/Data_Analytics-2015-11-30/Data
Analytics/Thesis/Dataset/ALL Country Dataset/Share Price CAN GBD USA IND.csv')
head(share_price_all)
```

```
share_price_can <- share_price_all[share_price_all$ï..LOCATION=='CAN',c(6,7)]
head(share_price_can)

dim(share_price_can)
share_price_can$Date <- paste0(share_price_can$TIME,'-01')
share_price_can$Date <- as.Date(share_price_can$Date)
share_price_can$TIME <- NULL
share_price_can<- share_price_can[format(share_price_can$Date, "%Y") >= 2000,]

head(share_price_can)

share_price_can$month <- as.factor(format(share_price_can$Date, "%m"))
share_price_can$year <- as.factor(format(share_price_can$Date, "%Y"))

min(share_price_can$Date,na.rm = T)
max(share_price_can$Date,na.rm = T)

merge_forex_cad_share <- merge(x=merge_forex_cad_ppi,y=share_price_can,by=c('month','year'),all.x =
TRUE)

merge_forex_cad_share <- merge_forex_cad_share[,c(1,2,3,4,5,6,8,7)]
head(merge_forex_cad_share)
colnames(merge_forex_cad_share) <- c('month','year','Date','Interest_rate','Cpi','PPI','Share','Exchange_rate')


merge_forex_cad_share <-
merge_forex_cad_share[with(merge_forex_cad_share,order(merge_forex_cad_share$Date)),]
merge_forex_cad_share$Day <- as.factor(format(merge_forex_cad_share$Date, "%d"))
View(merge_forex_cad_share)

#-- gold prize dataset

gold_price_all <- read.csv('/Study Material/Course Content/Data_Analytics-2015-11-30/Data
Analytics/Thesis/Dataset/ALL Country Dataset/Gold rate dataset.csv')
head(gold_price_all)
gold_price_cad <- gold_price_all[,c(1,4)]


gold_price_cad$Date<-as.Date(paste0(substr(gold_price_cad$Date,7,10),'/',
                        substr(gold_price_cad$Date,4,5),'/',substr(gold_price_cad$Date,1,2)))

gold_price_cad <- gold_price_cad[format(gold_price_cad$Date, "%Y") >= 2000,]

head(gold_price_cad)

dim(gold_price_cad)

dim(merge_forex_cad_share)
merge_forex_cad_gold <- merge(x=merge_forex_cad_share,y=gold_price_cad,by='Date',all.x =   TRUE)
dim(merge_forex_cad_gold)
head(merge_forex_cad_gold)

merge_forex_cad_gold <- merge_forex_cad_gold[,c(1,2,3,9,4,5,6,7,10,8)]

colnames(merge_forex_cad_gold) <-
c('Date','month','year','Day','Interest_rate','Cpi','PPI','Share','Gold','Exchange_rate')
```

```
head(merge_forex_cad_gold)

merge_forex_cad_gold$DayofWeek <- weekdays(as.Date(merge_forex_cad_gold$Date))
merge_forex_cad_gold$DayofWeek <- as.factor(merge_forex_cad_gold$DayofWeek)

levels(merge_forex_cad_gold$DayofWeek)

dim(merge_forex_cad_gold)
View(merge_forex_cad_gold)

na_count <-sapply(merge_forex_cad_gold,
            function(y)
              paste0(round(sum(length(which(is.na(y))))/nrow(merge_forex_cad_gold),3),' %')
)

library(timeDate)

extractdayOfWeek<-function(x){
 x$Monday<-isWeekday(x$Date, wday=1)
 x$Tuesday<-isWeekday(x$Date, wday=2)
 x$Wednesday<-isWeekday(x$Date, wday=3)
 x$Thursday<-isWeekday(x$Date, wday=4)
 x$Friday<-isWeekday(x$Date, wday=5)
 x$Saturday<-isWeekday(x$Date, wday=6)
 x$Sunday <-isWeekday(x$Date, wday=7)
 x$isWorkDay<-isWeekday(x$Date, wday=1:5)
 x$isWeekend<-isWeekday(x$Date, wday=6:7)
 return(x)
}


merge_forex_cad_gold_new<-data.frame(extractdayOfWeek(merge_forex_cad_gold))

cols <- sapply(merge_forex_cad_gold_new, is.logical)
merge_forex_cad_gold_new[,cols] <- lapply(merge_forex_cad_gold_new[,cols], as.numeric)

head(merge_forex_cad_gold_new)

merge_forex_cad_complete <-
merge_forex_cad_gold_new[,c(1,2,3,4,11,13,12,14,16,15,17,18,19,20,5,6,7,8,9,10)]

forex_cad <- NULL
interest_rate_cad <- NULL
merge_forex_cad <- NULL
Cpi_cad <- NULL
merge_forex_cad_cpi <- NULL
porduct_price_index_can <- NULL
merge_forex_cad_ppi <- NULL
share_price_can <- NULL
merge_forex_cad_share <- NULL
gold_price_cad <- NULL
merge_forex_cad_gold <- NULL
merge_forex_cad_gold_new <- NULL

head(merge_forex_cad_complete)
getwd()
write.csv(file='Forex_CAD_complete_Dataset.csv', x=merge_forex_cad_complete)
```