Rajleen Kaur

00713207218

CSE- 1

MACHINE LEARNING

LAB PROGRAM

Submission -6

Github link : LAB Program -6

EXPERIMENT-6

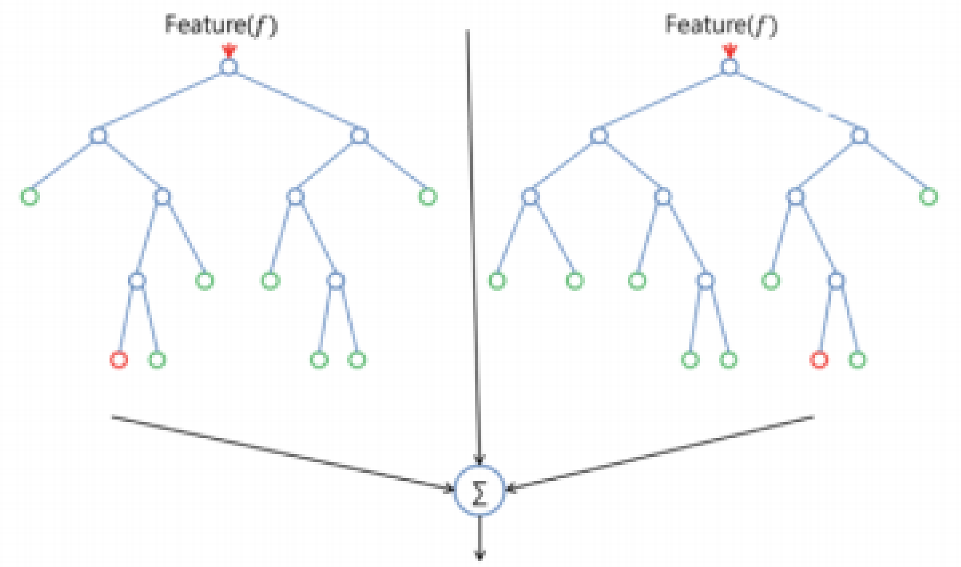
Problem Statement

Develop a machine learning method to predict stock price based on past price variation.

Algorithm

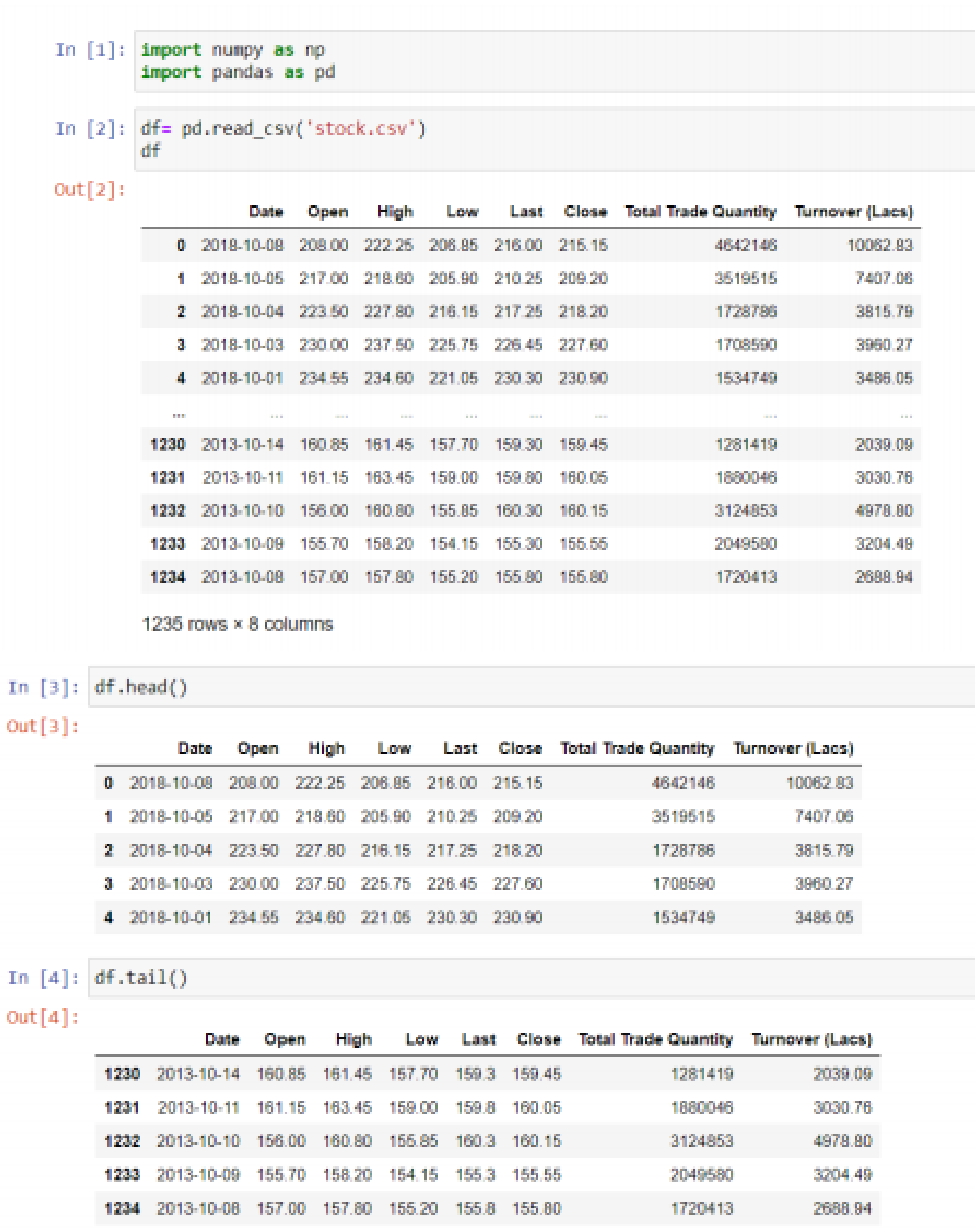
Random forest is a flexible, easy to use machine learning algorithm that produces, even without hyper-parameter tuning, a great result most of the time. It is also one of the most used algorithms, because of its simplicity and diversity (it can be used for both classification and regression tasks). In this post we'll learn how the random forest algorithm works, how it differs from other algorithms and how to use it.

One big advantage of random forest is that it can be used for both classification and regression problems, which form the majority of current machine learning systems. Let's look at random forest in classification, since classification is sometimes considered the building block of machine learning. Below you can see how a random forest would look like with two trees:



Random forest has nearly the same hyperparameters as a decision tree or a bagging classifier. Fortunately, there's no need to combine a decision tree with a bagging classifier because you can easily use the classifier-class of random forest. With random forest, you can also deal with regression tasks by using the algorithm's regressor.

Program Snippet



In [1]:
```
import numpy as np
import pandas as pd
```

In [2]:
```
df= pd.read_csv('stock.csv')
df
```

Out[2]:

	Date	Open	High	Low	Last	Close	Total Trade Quantity	Turnover (Lacs)
0	2018-10-08	208.00	222.25	206.85	216.00	215.15	4642146	10062.83
1	2018-10-05	217.00	218.60	205.90	210.25	209.20	3519515	7407.06
2	2018-10-04	223.50	227.80	216.15	217.25	218.20	1728786	3815.79
3	2018-10-03	230.00	237.50	225.75	226.45	227.60	1708590	3960.27
4	2018-10-01	234.55	234.60	221.05	230.30	230.90	1534749	3486.05
...	...	...	...	...	...	...	...	...
1230	2013-10-14	160.85	161.45	157.70	159.30	159.45	1281419	2039.09
1231	2013-10-11	161.15	163.45	159.00	159.80	160.05	1880046	3030.76
1232	2013-10-10	156.00	160.80	155.85	160.30	160.15	3124853	4978.80
1233	2013-10-09	155.70	158.20	154.15	155.30	155.55	2049580	3204.49
1234	2013-10-08	157.00	157.80	155.20	155.80	155.80	1720413	2688.94

1235 rows × 8 columns

In [3]:
```
df.head()
```

Out[3]:

	Date	Open	High	Low	Last	Close	Total Trade Quantity	Turnover (Lacs)
0	2018-10-08	208.00	222.25	206.85	216.00	215.15	4642146	10062.83
1	2018-10-05	217.00	218.60	205.90	210.25	209.20	3519515	7407.06
2	2018-10-04	223.50	227.80	216.15	217.25	218.20	1728786	3815.79
3	2018-10-03	230.00	237.50	225.75	226.45	227.60	1708590	3960.27
4	2018-10-01	234.55	234.60	221.05	230.30	230.90	1534749	3486.05

In [4]:
```
df.tail()
```

Out[4]:

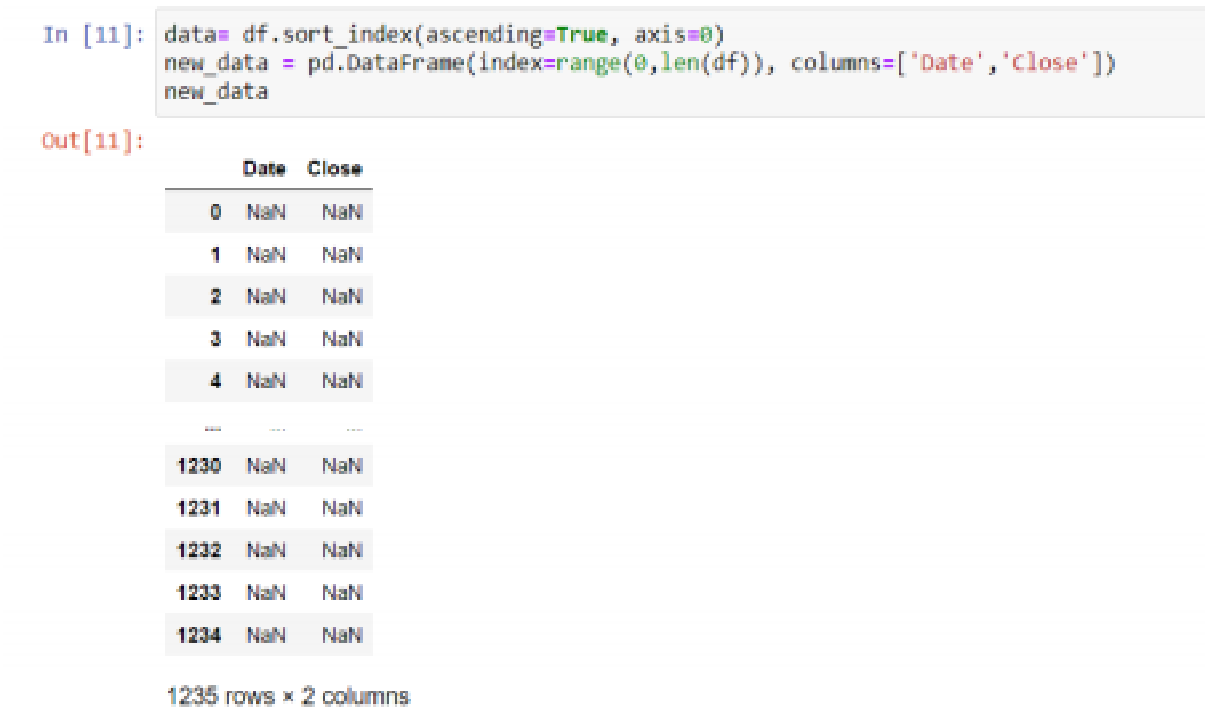
	Date	Open	High	Low	Last	Close	Total Trade Quantity	Turnover (Lacs)
1230	2013-10-14	160.85	161.45	157.70	159.3	159.45	1281419	2039.09
1231	2013-10-11	161.15	163.45	159.00	159.8	160.05	1880046	3030.76
1232	2013-10-10	156.00	160.80	155.85	160.3	160.15	3124853	4978.80
1233	2013-10-09	155.70	158.20	154.15	155.3	155.55	2049580	3204.49
1234	2013-10-08	157.00	157.80	155.20	155.8	155.80	1720413	2688.94

Long Short Term Memory (LSTM)

LSTMs are widely used for sequence prediction problems and have proven to be extremely effective. The reason they work so well is because LSTM is able to store past information that is important, and forget the information that is not. LSTM has three gates:

The input gate: The input gate adds information to the cell state

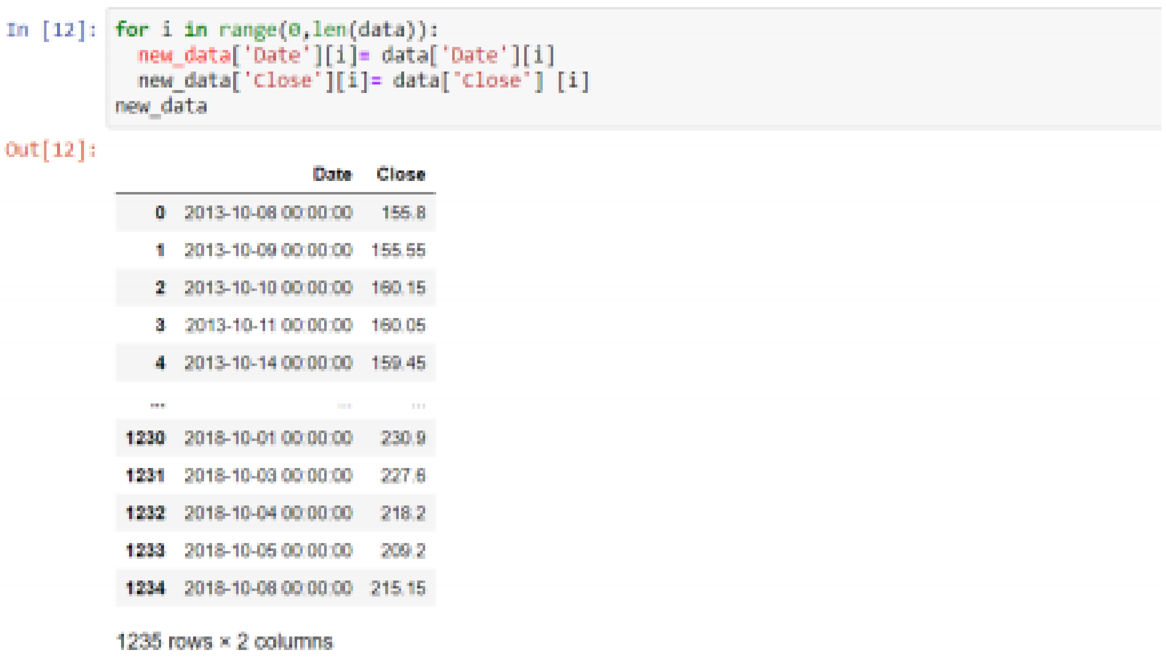
The forget gate: It removes the information that is no longer required by the model The output gate: Output Gate at LSTM selects the information to be shown as output



```
In [11]: data= df.sort_index(ascending=True, axis=0)
         new_data = pd.DataFrame(index=range(0,len(df)), columns=['Date','Close'])
         new_data
```

```
Out[11]:
         Date  Close
   0     NaN   NaN
   1     NaN   NaN
   2     NaN   NaN
   3     NaN   NaN
   4     NaN   NaN
 ...     ...   ...
1230     NaN   NaN
1231     NaN   NaN
1232     NaN   NaN
1233     NaN   NaN
1234     NaN   NaN

1235 rows × 2 columns
```



```
In [12]: for i in range(0,len(data)):
           new_data['Date'][i]= data['Date'][i]
           new_data['Close'][i]= data['Close'] [i]
         new_data
```

```
Out[12]:
                      Date   Close
   0   2013-10-08 00:00:00   155.8
   1   2013-10-09 00:00:00  155.55
   2   2013-10-10 00:00:00  160.15
   3   2013-10-11 00:00:00  160.05
   4   2013-10-14 00:00:00  159.45
 ...                   ...     ...
1230   2018-10-01 00:00:00   230.9
1231   2018-10-03 00:00:00   227.6
1232   2018-10-04 00:00:00   218.2
1233   2018-10-05 00:00:00   209.2
1234   2018-10-08 00:00:00  215.15

1235 rows × 2 columns
```