# Cryptocurrency Price Prediction

## Setup

```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import seaborn; seaborn.set()
import datetime
import pandas as pd
from keras.models import Sequential
from keras.layers import Dense, Activation
import seaborn as sns
sns.set()
%load_ext autoreload
%autoreload 2
%matplotlib inline
from fastai.imports import *
# from fastai.structured import *
from fastai.tabular import *
from fastai.tabular.all import *
# from pandas_summary import DataFrameSummary
from sklearn.ensemble import RandomForestRegressor,
RandomForestClassifier
from IPython.display import display
from sklearn import metrics



coinbase=pd.read_csv('coinbase.csv')
```

## Preprocessing

## Data Exploration

```
coinbase.head()

    Timestamp   Open   High    Low  Close  Volume_(BTC)
Volume_(Currency)  \
0  1417411980  300.0  300.0  300.0  300.0          0.01
3.0
1  1417412040  300.0  300.0  300.0  300.0          0.01
3.0
2  1417412100  300.0  300.0  300.0  300.0          0.01
3.0
3  1417412160  300.0  300.0  300.0  300.0          0.01
3.0
```

```
4  1417412220  300.0  300.0  300.0  300.0          0.01
3.0

   Weighted_Price
0           300.0
1           300.0
2           300.0
3           300.0
4           300.0
```

```
coinbase.describe()
```

```
          Timestamp          Open          High           Low
Close  \
count  1.574274e+06  1.574274e+06  1.574274e+06  1.574274e+06
1.574274e+06
mean   1.468131e+09  1.705118e+03  1.706025e+03  1.704113e+03
1.705123e+03
std    2.728500e+07  3.059038e+03  3.061434e+03  3.056505e+03
3.059105e+03
min    1.417412e+09  6.000000e-02  6.000000e-02  6.000000e-02
6.000000e-02
25%    1.444527e+09  2.903000e+02  2.904100e+02  2.901800e+02
2.903000e+02
50%    1.468141e+09  5.900500e+02  5.902100e+02  5.899800e+02
5.900200e+02
75%    1.491756e+09  1.224490e+03  1.224810e+03  1.224090e+03
1.224490e+03
max    1.515370e+09  1.989199e+04  1.989199e+04  1.989198e+04
1.989199e+04

       Volume_(BTC)  Volume_(Currency)  Weighted_Price
count  1.574274e+06       1.574274e+06    1.574274e+06
mean   7.073412e+00       2.267928e+04    1.705069e+03
std    1.698569e+01       1.225156e+05    3.058976e+03
min    1.000000e-08       2.641700e-06    6.000000e-02
25%    6.915000e-01       3.162361e+02    2.903031e+02
50%    2.381500e+00       1.398624e+03    5.900207e+02
75%    7.032457e+00       7.601787e+03    1.224453e+03
max    1.563267e+03       1.997076e+07    1.989199e+04
```

```
coinbase.isna().sum()/coinbase.count()
```

```
Timestamp            0.0
Open                 0.0
High                 0.0
Low                  0.0
Close                0.0
Volume_(BTC)         0.0
Volume_(Currency)    0.0
```

```
Weighted_Price       0.0
dtype: float64

def timestampToDateTime(timestamp):
    from datetime import datetime
    return datetime.fromtimestamp(timestamp)

coinbase['Date']=coinbase['Timestamp'].apply(timestampToDateTime)

coinbase.head()

    Timestamp   Open   High    Low  Close  Volume_(BTC)
Volume_(Currency)  \
0  1417411980  300.0  300.0  300.0  300.0          0.01
3.0
1  1417412040  300.0  300.0  300.0  300.0          0.01
3.0
2  1417412100  300.0  300.0  300.0  300.0          0.01
3.0
3  1417412160  300.0  300.0  300.0  300.0          0.01
3.0
4  1417412220  300.0  300.0  300.0  300.0          0.01
3.0

   Weighted_Price                Date
0           300.0 2014-12-01 00:33:00
1           300.0 2014-12-01 00:34:00
2           300.0 2014-12-01 00:35:00
3           300.0 2014-12-01 00:36:00
4           300.0 2014-12-01 00:37:00

coinbase.drop('Timestamp',axis=1,inplace=True)
```

Let's verify if everything looks good

```
coinbase.head()

    Open   High    Low  Close  Volume_(BTC)  Volume_(Currency)  \
0  300.0  300.0  300.0  300.0          0.01                3.0
1  300.0  300.0  300.0  300.0          0.01                3.0
2  300.0  300.0  300.0  300.0          0.01                3.0
3  300.0  300.0  300.0  300.0          0.01                3.0
4  300.0  300.0  300.0  300.0          0.01                3.0

   Weighted_Price                Date
0           300.0 2014-12-01 00:33:00
1           300.0 2014-12-01 00:34:00
2           300.0 2014-12-01 00:35:00
3           300.0 2014-12-01 00:36:00
4           300.0 2014-12-01 00:37:00
```

```
coinbase[coinbase['Date']=='2016']
```

```
          Open    High     Low   Close  Volume_(BTC)
Volume_(Currency)  \
511853  436.12  436.13  436.12  436.13        1.4387
627.460131

        Weighted_Price       Date
511853       436.12993 2016-01-01
```

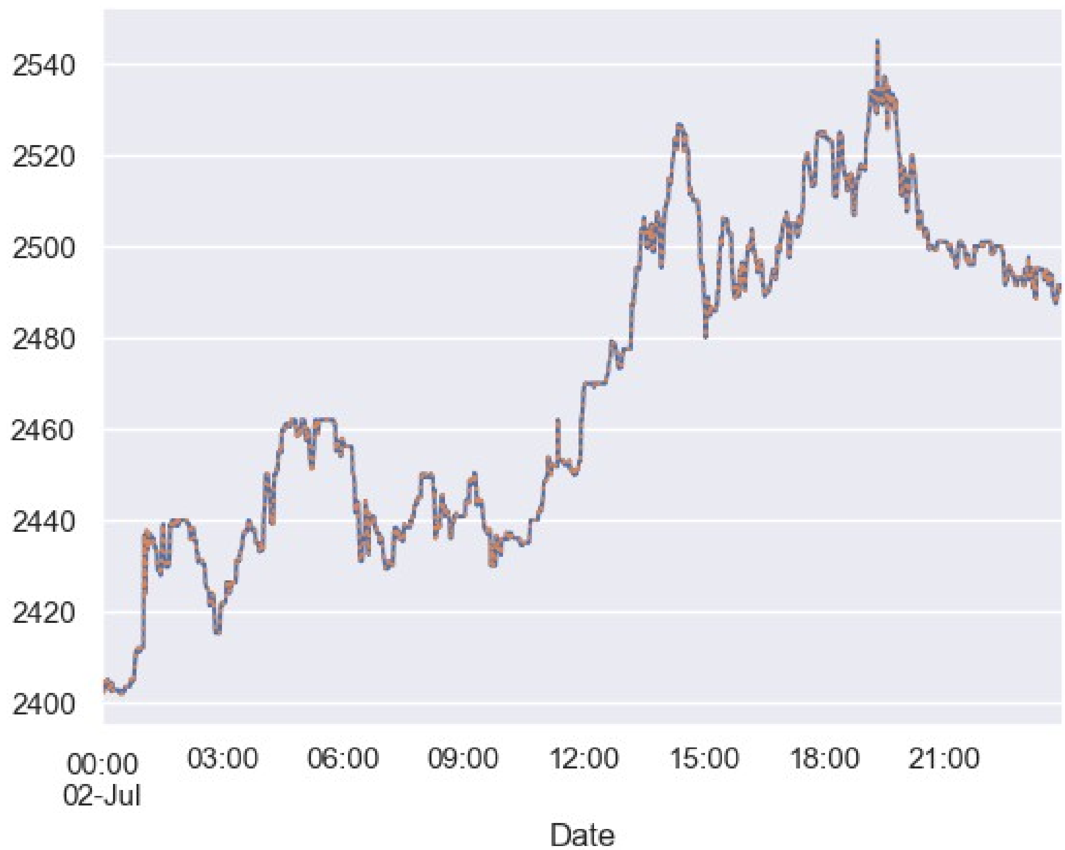
To fetch all the data for a year, we need to first index the dataframe by date and then we can query intresting things and explore the data more. This is pretty common in time series analysis and useful for data exploration too.

```
timeindex=pd.DatetimeIndex(coinbase['Date'])

coinbase.set_index(timeindex,inplace=True)

coinbase['2016-01-21'].head()
```

```
<ipython-input-17-fa20f238523a>:1: FutureWarning: Indexing a DataFrame
with a datetimelike index using a single string to slice the rows,
like `frame[string]`, is deprecated and will be removed in a future
version. Use `frame.loc[string]` instead.
  coinbase['2016-01-21'].head()

                       Open    High     Low   Close  Volume_(BTC)  \
Date
2016-01-21 00:00:00  414.57  414.59  414.54  414.59     21.464480
2016-01-21 00:01:00  414.56  414.56  414.27  414.27      6.672710
2016-01-21 00:02:00  414.24  414.24  414.00  414.00      8.210732
2016-01-21 00:03:00  414.00  414.01  414.00  414.01      1.620680
2016-01-21 00:04:00  414.00  414.01  413.55  413.55     42.184428

                     Volume_(Currency)  Weighted_Price
Date
Date

2016-01-21 00:00:00        8898.054200      414.547858 2016-01-21
00:00:00
2016-01-21 00:01:00        2766.101840      414.539496 2016-01-21
00:01:00
2016-01-21 00:02:00        3399.333643      414.011022 2016-01-21
00:02:00
2016-01-21 00:03:00         670.972006      414.006470 2016-01-21
00:03:00
2016-01-21 00:04:00       17453.142163      413.734233 2016-01-21
00:04:00
```

```
data=coinbase['2017-07-02']
data['High'].plot(style="-")
data['High'].resample('BM').plot(style=":")
```

```
<ipython-input-19-69aa7b0f3776>:1: FutureWarning: Indexing a DataFrame
with a datetimelike index using a single string to slice the rows,
like `frame[string]`, is deprecated and will be removed in a future
version. Use `frame.loc[string]` instead.
  data=coinbase['2017-07-02']

Date
2017-07-31    AxesSubplot(0.125,0.11;0.775x0.77)
Freq: BM, Name: High, dtype: object
```




```
def preprocess(dataframe):
    data=data.fillna(method='ffill')
    data=add_datepart(data, 'Date')
    return data
```

As we noticed there are few NAN values and we dont want to drop those values. Since it is a time series data, we need to be extremely careful how we handle these values. There are two options:

1. Fill the missing values with previous values (forward fill)

2. Fill the missing values with future values (backward fill)

We choose forward Fill

```
coinbase=coinbase.fillna(method='ffill')

coinbase.corr()

<ipython-input-22-6f0edc0a89c4>:1: FutureWarning: The default value of
numeric_only in DataFrame.corr is deprecated. In a future version, it
will default to False. Select only valid columns or specify the value
of numeric_only to silence this warning.
  coinbase.corr()

                       Open      High       Low     Close
Volume_(BTC)  \
Open               1.000000  0.999997  0.999997  0.999996
0.204421
High               0.999997  1.000000  0.999994  0.999998
0.204978
Low                0.999997  0.999994  1.000000  0.999997
0.203783
Close              0.999996  0.999998  0.999997  1.000000
0.204399
Volume_(BTC)       0.204421  0.204978  0.203783  0.204399
1.000000
Volume_(Currency)  0.497802  0.498775  0.496770  0.497814
0.575376
Weighted_Price     0.999999  0.999998  0.999998  0.999999
0.204366

                   Volume_(Currency)  Weighted_Price
Open                        0.497802        0.999999
High                        0.498775        0.999998
Low                         0.496770        0.999998
Close                       0.497814        0.999999
Volume_(BTC)                0.575376        0.204366
Volume_(Currency)           1.000000        0.497755
Weighted_Price              0.497755        1.000000

coinbase['PriceClose2D']=coinbase['Close']

shift=24 # 24 hours = 2days
coinbase['PriceClose2D']=coinbase['PriceClose2D'].shift(-shift)
coinbase=coinbase[:-shift]

coinbase[73:90]

                      Open   High    Low  Close  Volume_(BTC)  \
Date
2014-12-01 01:46:00  370.0  370.0  370.0  370.0      0.010000
```

```
2014-12-01 01:47:00  370.0  370.0  370.0  370.0      0.010000
2014-12-01 01:48:00  370.0  370.0  370.0  370.0      0.010000
2014-12-01 01:49:00  370.0  370.0  370.0  370.0      0.010000
2014-12-01 01:50:00  370.0  370.0  370.0  370.0      0.026556
2014-12-01 01:51:00  370.0  370.0  370.0  370.0      0.026556
2014-12-01 01:52:00  370.0  370.0  370.0  370.0      0.026556
2014-12-01 01:53:00  370.0  370.0  370.0  370.0      0.026556
2014-12-01 01:54:00  370.0  370.0  370.0  370.0      0.026556
2014-12-01 01:55:00  370.0  370.0  370.0  370.0      0.026556
2014-12-01 01:56:00  370.0  370.0  370.0  370.0      0.026556
2014-12-01 01:57:00  370.0  370.0  370.0  370.0      0.026556
2014-12-01 01:58:00  370.0  370.0  370.0  370.0      0.026556
2014-12-01 01:59:00  370.0  370.0  370.0  370.0      0.026556
2014-12-01 02:00:00  370.0  370.0  370.0  370.0      0.026556
2014-12-01 02:01:00  370.0  370.0  370.0  370.0      0.026556
2014-12-01 02:02:00  370.0  370.0  370.0  370.0      0.026556

                     Volume_(Currency)  Weighted_Price
Date  \
Date

2014-12-01 01:46:00            3.70000           370.0 2014-12-01
01:46:00
2014-12-01 01:47:00            3.70000           370.0 2014-12-01
01:47:00
2014-12-01 01:48:00            3.70000           370.0 2014-12-01
01:48:00
2014-12-01 01:49:00            3.70000           370.0 2014-12-01
01:49:00
2014-12-01 01:50:00            9.82555           370.0 2014-12-01
01:50:00
2014-12-01 01:51:00            9.82555           370.0 2014-12-01
01:51:00
2014-12-01 01:52:00            9.82555           370.0 2014-12-01
01:52:00
2014-12-01 01:53:00            9.82555           370.0 2014-12-01
01:53:00
2014-12-01 01:54:00            9.82555           370.0 2014-12-01
01:54:00
2014-12-01 01:55:00            9.82555           370.0 2014-12-01
01:55:00
2014-12-01 01:56:00            9.82555           370.0 2014-12-01
01:56:00
2014-12-01 01:57:00            9.82555           370.0 2014-12-01
01:57:00
2014-12-01 01:58:00            9.82555           370.0 2014-12-01
01:58:00
2014-12-01 01:59:00            9.82555           370.0 2014-12-01
01:59:00
```

```
2014-12-01 02:00:00          9.82555           370.0 2014-12-01
02:00:00
2014-12-01 02:01:00          9.82555           370.0 2014-12-01
02:01:00
2014-12-01 02:02:00          9.82555           370.0 2014-12-01
02:02:00

                     PriceClose2D
Date
2014-12-01 01:46:00         370.0
2014-12-01 01:47:00         370.0
2014-12-01 01:48:00         370.0
2014-12-01 01:49:00         370.0
2014-12-01 01:50:00         370.0
2014-12-01 01:51:00         370.0
2014-12-01 01:52:00         370.0
2014-12-01 01:53:00         370.0
2014-12-01 01:54:00         370.0
2014-12-01 01:55:00         370.0
2014-12-01 01:56:00         370.0
2014-12-01 01:57:00         370.0
2014-12-01 01:58:00         370.0
2014-12-01 01:59:00         370.0
2014-12-01 02:00:00         370.0
2014-12-01 02:01:00         370.0
2014-12-01 02:02:00         370.0
```

```
add_datepart(coinbase,'Date')
```

```
                         Open      High       Low     Close
Volume_(BTC)  \
Date

2014-12-01 00:33:00    300.00    300.00    300.00    300.00
0.010000
2014-12-01 00:34:00    300.00    300.00    300.00    300.00
0.010000
2014-12-01 00:35:00    300.00    300.00    300.00    300.00
0.010000
2014-12-01 00:36:00    300.00    300.00    300.00    300.00
0.010000
2014-12-01 00:37:00    300.00    300.00    300.00    300.00
0.010000
...                       ...       ...       ...       ...
...
2018-01-07 18:32:00  16349.00  16349.00  16329.00  16329.00
4.303270
2018-01-07 18:33:00  16329.01  16329.01  16300.00  16302.85
8.183248
2018-01-07 18:34:00  16302.84  16302.84  16279.74  16279.74
```

```
7.083056
2018-01-07 18:35:00  16279.75  16279.76  16266.06  16266.06
8.379655
2018-01-07 18:36:00  16266.07  16266.07  16266.06  16266.06
4.943145

                     Volume_(Currency)  Weighted_Price  PriceClose2D
Year  \
Date

2014-12-01 00:33:00           3.000000      300.000000        300.00
2014
2014-12-01 00:34:00           3.000000      300.000000        300.00
2014
2014-12-01 00:35:00           3.000000      300.000000        300.00
2014
2014-12-01 00:36:00           3.000000      300.000000        300.00
2014
2014-12-01 00:37:00           3.000000      300.000000        300.00
2014
...                                ...             ...           ...
...
2018-01-07 18:32:00       70338.446619    16345.349090      16174.23
2018
2018-01-07 18:33:00      133574.283340    16322.892267      16174.22
2018
2018-01-07 18:34:00      115426.384420    16296.128527      16174.21
2018
2018-01-07 18:35:00      136373.204380    16274.321438      16174.22
2018
2018-01-07 18:36:00       80405.530720    16266.068487      16174.22
2018

                     Month  ...  Day  Dayofweek  Dayofyear
Is_month_end  \
Date                        ...

2014-12-01 00:33:00     12  ...    1          0        335
False
2014-12-01 00:34:00     12  ...    1          0        335
False
2014-12-01 00:35:00     12  ...    1          0        335
False
2014-12-01 00:36:00     12  ...    1          0        335
False
2014-12-01 00:37:00     12  ...    1          0        335
False
...                    ...  ...  ...        ...        ...            .
..
2018-01-07 18:32:00      1  ...    7          6          7
```

```
False
2018-01-07 18:33:00      1  ...    7          6          7
False
2018-01-07 18:34:00      1  ...    7          6          7
False
2018-01-07 18:35:00      1  ...    7          6          7
False
2018-01-07 18:36:00      1  ...    7          6          7
False

                     Is_month_start  Is_quarter_end  Is_quarter_start
\
Date

2014-12-01 00:33:00            True           False             False

2014-12-01 00:34:00            True           False             False

2014-12-01 00:35:00            True           False             False

2014-12-01 00:36:00            True           False             False

2014-12-01 00:37:00            True           False             False

...                             ...             ...               ...

2018-01-07 18:32:00           False           False             False

2018-01-07 18:33:00           False           False             False

2018-01-07 18:34:00           False           False             False

2018-01-07 18:35:00           False           False             False

2018-01-07 18:36:00           False           False             False


                     Is_year_end  Is_year_start       Elapsed
Date
2014-12-01 00:33:00        False          False  1.417394e+09
2014-12-01 00:34:00        False          False  1.417394e+09
2014-12-01 00:35:00        False          False  1.417394e+09
2014-12-01 00:36:00        False          False  1.417394e+09
2014-12-01 00:37:00        False          False  1.417394e+09
...                          ...            ...           ...
2018-01-07 18:32:00        False          False  1.515350e+09
2018-01-07 18:33:00        False          False  1.515350e+09
2018-01-07 18:34:00        False          False  1.515350e+09
2018-01-07 18:35:00        False          False  1.515350e+09
2018-01-07 18:36:00        False          False  1.515350e+09
```

```
[1574250 rows x 21 columns]
```

```python
coinbase['2015']
```

```
<ipython-input-27-2868ae9b1f97>:1: FutureWarning: Indexing a DataFrame
with a datetimelike index using a single string to slice the rows,
like `frame[string]`, is deprecated and will be removed in a future
version. Use `frame.loc[string]` instead.
  coinbase['2015']
```

```
                       Open    High     Low   Close  Volume_(BTC)  \
Date
2015-01-07 20:24:00  360.00  360.00  360.00  360.00      0.010000
2015-01-07 20:25:00  360.00  360.00  360.00  360.00      0.010000
2015-01-07 20:26:00  360.00  360.00  360.00  360.00      0.010000
2015-01-07 20:27:00  271.84  276.34  271.84  276.34      0.020000
2015-01-07 20:28:00  295.19  319.84  271.60  271.60      0.030000
...                     ...     ...     ...     ...           ...
2015-12-31 23:55:00  437.11  437.11  437.02  437.02      1.308700
2015-12-31 23:56:00  437.02  437.07  437.02  437.07      1.017000
2015-12-31 23:57:00  437.02  437.12  436.03  436.03     23.060550
2015-12-31 23:58:00  436.02  436.36  436.02  436.13      0.312749
2015-12-31 23:59:00  436.12  436.13  436.12  436.12      6.139053

                     Volume_(Currency)  Weighted_Price  PriceClose2D
Year  \
Date

2015-01-07 20:24:00           3.600000      360.000000        317.98
2015
2015-01-07 20:25:00           3.600000      360.000000        301.99
2015
2015-01-07 20:26:00           3.600000      360.000000        333.28
2015
2015-01-07 20:27:00           5.481800      274.090000        329.03
2015
2015-01-07 20:28:00           8.866300      295.543333        318.88
2015
...                                ...             ...           ...
...
2015-12-31 23:55:00         572.007400      437.080614        435.51
2015
2015-12-31 23:56:00         444.467690      437.038043        435.41
2015
2015-12-31 23:57:00       10072.788522      436.797410        435.67
2015
2015-12-31 23:58:00         136.409927      436.163856        435.53
2015
2015-12-31 23:59:00        2677.366208      436.120393        435.78
```

```
2015

                     Month  ...  Day  Dayofweek  Dayofyear  
Is_month_end  \
Date                        ...                              
                                                             
2015-01-07 20:24:00      1  ...    7          2          7  
False
2015-01-07 20:25:00      1  ...    7          2          7  
False
2015-01-07 20:26:00      1  ...    7          2          7  
False
2015-01-07 20:27:00      1  ...    7          2          7  
False
2015-01-07 20:28:00      1  ...    7          2          7  
False
...                    ...  ...  ...        ...        ...             .
..
2015-12-31 23:55:00     12  ...   31          3        365  
True
2015-12-31 23:56:00     12  ...   31          3        365  
True
2015-12-31 23:57:00     12  ...   31          3        365  
True
2015-12-31 23:58:00     12  ...   31          3        365  
True
2015-12-31 23:59:00     12  ...   31          3        365  
True

                     Is_month_start  Is_quarter_end  Is_quarter_start  
\
Date                                                                    

2015-01-07 20:24:00           False           False             False  

2015-01-07 20:25:00           False           False             False  

2015-01-07 20:26:00           False           False             False  

2015-01-07 20:27:00           False           False             False  

2015-01-07 20:28:00           False           False             False  

...                             ...             ...               ...  

2015-12-31 23:55:00           False            True             False  

2015-12-31 23:56:00           False            True             False  

2015-12-31 23:57:00           False            True             False  
```

```
2015-12-31 23:58:00           False           True             False
2015-12-31 23:59:00           False           True             False


                     Is_year_end  Is_year_start       Elapsed
Date
2015-01-07 20:24:00        False          False  1.420662e+09
2015-01-07 20:25:00        False          False  1.420662e+09
2015-01-07 20:26:00        False          False  1.420662e+09
2015-01-07 20:27:00        False          False  1.420662e+09
2015-01-07 20:28:00        False          False  1.420662e+09
...                          ...            ...           ...
2015-12-31 23:55:00         True          False  1.451606e+09
2015-12-31 23:56:00         True          False  1.451606e+09
2015-12-31 23:57:00         True          False  1.451606e+09
2015-12-31 23:58:00         True          False  1.451606e+09
2015-12-31 23:59:00         True          False  1.451606e+09

[507735 rows x 21 columns]
```

```
coinbase.corr()
```

```
                         Open      High       Low     Close
Volume_(BTC)  \
Open                 1.000000  0.999997  0.999997  0.999996
0.204456
High                 0.999997  1.000000  0.999994  0.999998
0.205014
Low                  0.999997  0.999994  1.000000  0.999997
0.203819
Close                0.999996  0.999998  0.999997  1.000000
0.204435
Volume_(BTC)         0.204456  0.205014  0.203819  0.204435
1.000000
Volume_(Currency)    0.497838  0.498812  0.496806  0.497851
0.575377
Weighted_Price       0.999999  0.999998  0.999998  0.999999
0.204402
PriceClose2D         0.999926  0.999927  0.999926  0.999929
0.204717
Year                 0.549947  0.549839  0.550051  0.549945
0.109475
Month                0.316950  0.316947  0.316959  0.316954
0.073286
Week                 0.306384  0.306383  0.306391  0.306387
0.069306
Day                 -0.008167 -0.008150 -0.008184 -0.008171      -
0.013532
```

```
Dayofweek          0.008783  0.008802  0.008764  0.008784     -
0.039841
Dayofyear          0.314668  0.314666  0.314676  0.314671
0.071370
Is_month_end      -0.000271 -0.000270 -0.000276 -0.000273     -
0.003416
Is_month_start    -0.000525 -0.000536 -0.000514 -0.000524     -
0.005697
Is_quarter_end     0.013476  0.013469  0.013482  0.013476     -
0.008200
Is_quarter_start   0.013526  0.013518  0.013536  0.013525     -
0.013314
Is_year_end        0.055735  0.055732  0.055735  0.055735     -
0.003081
Is_year_start      0.055640  0.055635  0.055645  0.055638     -
0.004432
Elapsed            0.634166  0.634061  0.634269  0.634166
0.129252

                   Volume_(Currency)  Weighted_Price  PriceClose2D
Year  \
Open                        0.497838        0.999999      0.999926
0.549947
High                        0.498812        0.999998      0.999927
0.549839
Low                         0.496806        0.999998      0.999926
0.550051
Close                       0.497851        0.999999      0.999929
0.549945
Volume_(BTC)                0.575377        0.204402      0.204717
0.109475
Volume_(Currency)           1.000000        0.497791      0.498212
0.210434
Weighted_Price              0.497791        1.000000      0.999928
0.549944
PriceClose2D                0.498212        0.999928      1.000000
0.549956
Year                        0.210434        0.549944      0.549956
1.000000
Month                       0.165267        0.316952      0.316942 -
0.056114
Week                        0.160392        0.306385      0.306371 -
0.069203
Day                        -0.009471       -0.008169     -0.008236 -
0.022844
Dayofweek                  -0.000594        0.008783      0.008779
0.011033
Dayofyear                   0.163651        0.314669      0.314655 -
0.057633
```

```
Is_month_end                -0.004950       -0.000274      -0.000274 -
0.004109
Is_month_start              -0.004606       -0.000526      -0.000496
0.001939
Is_quarter_end              -0.003706        0.013475       0.013491 -
0.002346
Is_quarter_start            -0.005869        0.013526       0.013505
0.029331
Is_year_end                  0.013267        0.055733       0.055763 -
0.001168
Is_year_start                0.010884        0.055639       0.055597
0.061923
Elapsed                      0.257106        0.634164       0.634171
0.943108

                      Month  ...       Day  Dayofweek  Dayofyear  \
Open               0.316950  ... -0.008167   0.008783   0.314668
High               0.316947  ... -0.008150   0.008802   0.314666
Low                0.316959  ... -0.008184   0.008764   0.314676
Close              0.316954  ... -0.008171   0.008784   0.314671
Volume_(BTC)       0.073286  ... -0.013532  -0.039841   0.071370
Volume_(Currency)  0.165267  ... -0.009471  -0.000594   0.163651
Weighted_Price     0.316952  ... -0.008169   0.008783   0.314669
PriceClose2D       0.316942  ... -0.008236   0.008779   0.314655
Year              -0.056114  ... -0.022844   0.011033  -0.057633
Month              1.000000  ... -0.000243   0.001834   0.996452
Week               0.975682  ...  0.063371  -0.002510   0.977617
Day               -0.000243  ...  1.000000   0.004559   0.083622
Dayofweek          0.001834  ...  0.004559   1.000000   0.002194
Dayofyear          0.996452  ...  0.083622   0.002194   1.000000
Is_month_end      -0.003516  ...  0.307668   0.000601   0.022366
Is_month_start     0.004604  ... -0.311871  -0.003963  -0.021595
Is_quarter_end     0.028601  ...  0.176312   0.022318   0.043184
Is_quarter_start  -0.032616  ... -0.175498   0.044293  -0.047310
Is_year_end        0.082826  ...  0.090760   0.043939   0.090215
Is_year_start     -0.084825  ... -0.087386   0.008925  -0.091521
Elapsed            0.277860  ...  0.005868   0.011340   0.277576

                   Is_month_end  Is_month_start  Is_quarter_end  \
Open                  -0.000271       -0.000525        0.013476
High                  -0.000270       -0.000536        0.013469
Low                   -0.000276       -0.000514        0.013482
Close                 -0.000273       -0.000524        0.013476
Volume_(BTC)          -0.003416       -0.005697       -0.008200
Volume_(Currency)     -0.004950       -0.004606       -0.003706
Weighted_Price        -0.000274       -0.000526        0.013475
PriceClose2D          -0.000274       -0.000496        0.013491
Year                  -0.004109        0.001939       -0.002346
Month                 -0.003516        0.004604        0.028601
```

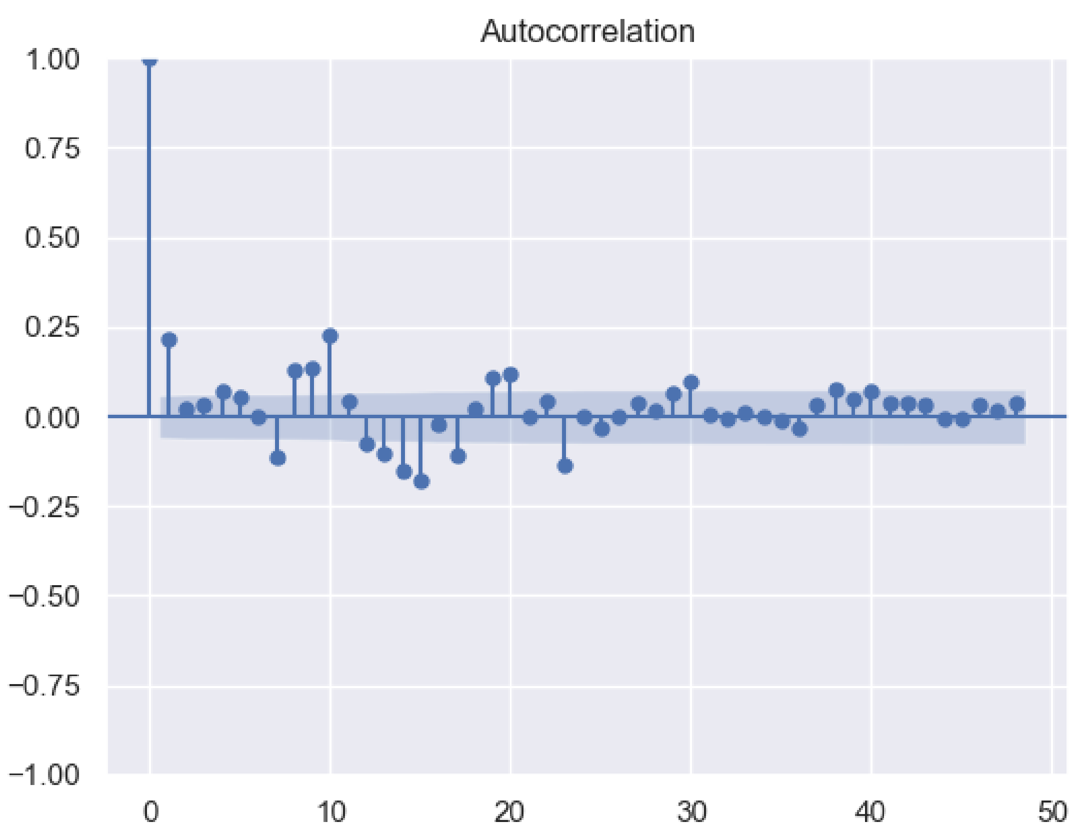
```
Week                  0.020027        0.011443        0.041412
Day                   0.307668       -0.311871        0.176312
Dayofweek             0.000601       -0.003963        0.022318
Dayofyear             0.022366       -0.021595        0.043184
Is_month_end          1.000000       -0.034546        0.570907
Is_month_start       -0.034546        1.000000       -0.019723
Is_quarter_end        0.570907       -0.019723        1.000000
Is_quarter_start     -0.019440        0.562725       -0.011098
Is_year_end           0.284273       -0.009821        0.497932
Is_year_start        -0.009680        0.280199       -0.005526
Elapsed               0.003496       -0.005321        0.012124

                   Is_quarter_start  Is_year_end  Is_year_start
Elapsed
Open                       0.013526     0.055735       0.055640
0.634166
High                       0.013518     0.055732       0.055635
0.634061
Low                        0.013536     0.055735       0.055645
0.634269
Close                      0.013525     0.055735       0.055638
0.634166
Volume_(BTC)              -0.013314    -0.003081      -0.004432
0.129252
Volume_(Currency)         -0.005869     0.013267       0.010884
0.257106
Weighted_Price             0.013526     0.055733       0.055639
0.634164
PriceClose2D               0.013505     0.055763       0.055597
0.634171
Year                       0.029331    -0.001168       0.061923
0.943108
Month                     -0.032616     0.082826      -0.084825
0.277860
Week                       0.011628     0.088665       0.029344
0.258994
Day                       -0.175498     0.090760      -0.087386
0.005868
Dayofweek                  0.044293     0.043939       0.008925
0.011340
Dayofyear                 -0.047310     0.090215      -0.091521
0.277576
Is_month_end              -0.019440     0.284273      -0.009680
0.003496
Is_month_start             0.562725    -0.009821       0.280199 -
0.005321
Is_quarter_end            -0.011098     0.497932      -0.005526
0.012124
Is_quarter_start           1.000000    -0.005526       0.497932
```

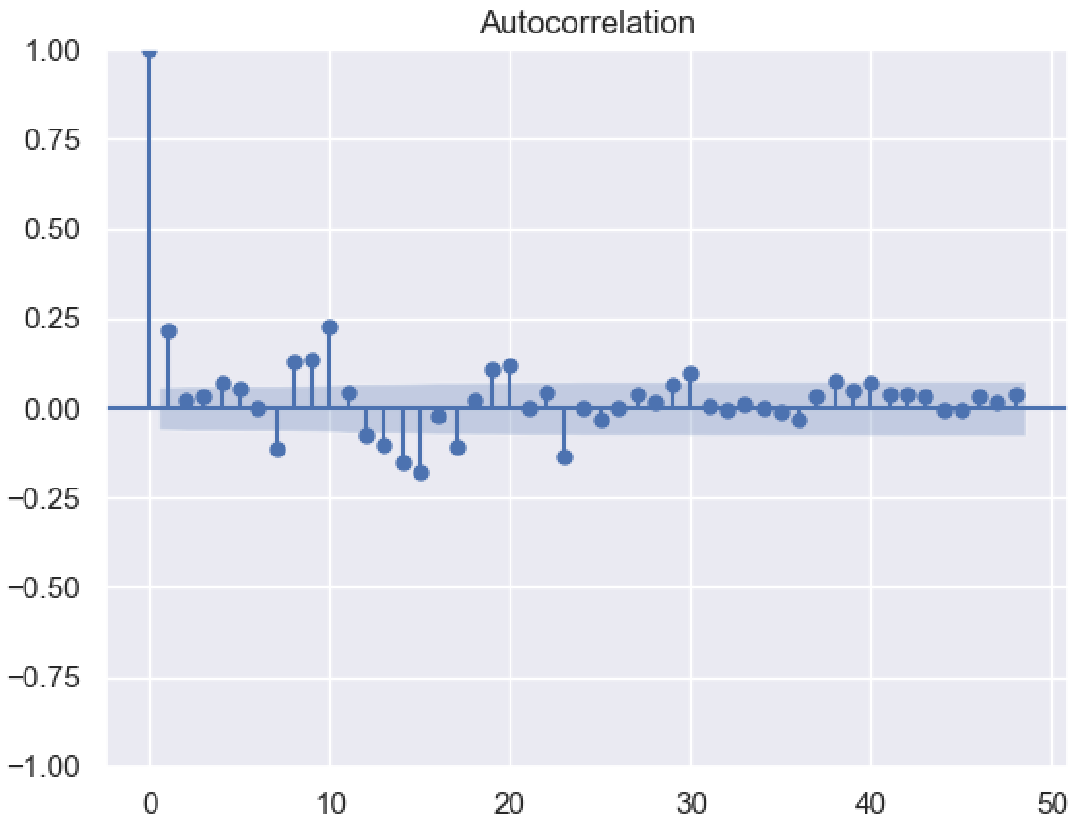
```
0.012458
Is_year_end                      -0.005526      1.000000      -0.002752
0.028919
Is_year_start                     0.497932     -0.002752       1.000000
0.029085
Elapsed                           0.012458      0.028919       0.029085
1.000000

[21 rows x 21 columns]
```

## Financial Stock Market Analysis

```
arima2015day=coinbase['2015':].resample('D').mean().fillna(method='ffi
ll')['Close']

from statsmodels.graphics.tsaplots import plot_acf
plot_acf(arima2015day.diff().dropna(), lags= 48, alpha=0.05)
```

```
<ipython-input-30-5876ad1fb838>:1: FutureWarning: Value based partial
slicing on non-monotonic DatetimeIndexes with non-existing keys is
deprecated and will raise a KeyError in a future Version.

arima2015day=coinbase['2015':].resample('D').mean().fillna(method='ffi
ll')['Close']
```






# Training & Testing

## Model 1: RANDOM FOREST :|

```
from sklearn.ensemble import RandomForestRegressor
from sklearn.datasets import make_regression
from sklearn.metrics import mean_squared_error
coinbase

                         Open      High       Low     Close
Volume_(BTC)  \
Date

2014-12-01 00:33:00    300.00    300.00    300.00    300.00
0.010000
2014-12-01 00:34:00    300.00    300.00    300.00    300.00
0.010000
2014-12-01 00:35:00    300.00    300.00    300.00    300.00
0.010000
2014-12-01 00:36:00    300.00    300.00    300.00    300.00
0.010000
2014-12-01 00:37:00    300.00    300.00    300.00    300.00
```

```
0.010000
...                        ...       ...       ...       ...
...
2018-01-07 18:32:00  16349.00  16349.00  16329.00  16329.00
4.303270
2018-01-07 18:33:00  16329.01  16329.01  16300.00  16302.85
8.183248
2018-01-07 18:34:00  16302.84  16302.84  16279.74  16279.74
7.083056
2018-01-07 18:35:00  16279.75  16279.76  16266.06  16266.06
8.379655
2018-01-07 18:36:00  16266.07  16266.07  16266.06  16266.06
4.943145

                     Volume_(Currency)  Weighted_Price  PriceClose2D
Year  \
Date

2014-12-01 00:33:00           3.000000      300.000000        300.00
2014
2014-12-01 00:34:00           3.000000      300.000000        300.00
2014
2014-12-01 00:35:00           3.000000      300.000000        300.00
2014
2014-12-01 00:36:00           3.000000      300.000000        300.00
2014
2014-12-01 00:37:00           3.000000      300.000000        300.00
2014
...                                ...             ...           ...
...
2018-01-07 18:32:00       70338.446619    16345.349090      16174.23
2018
2018-01-07 18:33:00      133574.283340    16322.892267      16174.22
2018
2018-01-07 18:34:00      115426.384420    16296.128527      16174.21
2018
2018-01-07 18:35:00      136373.204380    16274.321438      16174.22
2018
2018-01-07 18:36:00       80405.530720    16266.068487      16174.22
2018

                     Month  ...  Day  Dayofweek  Dayofyear
Is_month_end  \
Date                        ...

2014-12-01 00:33:00     12  ...    1          0        335
False
2014-12-01 00:34:00     12  ...    1          0        335
False
2014-12-01 00:35:00     12  ...    1          0        335
```

```
False
2014-12-01 00:36:00     12  ...    1          0        335
False
2014-12-01 00:37:00     12  ...    1          0        335
False
...                    ...  ...  ...        ...        ...           .
..
2018-01-07 18:32:00      1  ...    7          6          7
False
2018-01-07 18:33:00      1  ...    7          6          7
False
2018-01-07 18:34:00      1  ...    7          6          7
False
2018-01-07 18:35:00      1  ...    7          6          7
False
2018-01-07 18:36:00      1  ...    7          6          7
False

                     Is_month_start  Is_quarter_end  Is_quarter_start
\
Date

2014-12-01 00:33:00            True           False             False

2014-12-01 00:34:00            True           False             False

2014-12-01 00:35:00            True           False             False

2014-12-01 00:36:00            True           False             False

2014-12-01 00:37:00            True           False             False

...                             ...             ...               ...

2018-01-07 18:32:00           False           False             False

2018-01-07 18:33:00           False           False             False

2018-01-07 18:34:00           False           False             False

2018-01-07 18:35:00           False           False             False

2018-01-07 18:36:00           False           False             False


                     Is_year_end  Is_year_start       Elapsed
Date
2014-12-01 00:33:00        False          False  1.417394e+09
2014-12-01 00:34:00        False          False  1.417394e+09
2014-12-01 00:35:00        False          False  1.417394e+09
2014-12-01 00:36:00        False          False  1.417394e+09
```

```
2014-12-01 00:37:00        False          False  1.417394e+09
...                          ...            ...           ...
2018-01-07 18:32:00        False          False  1.515350e+09
2018-01-07 18:33:00        False          False  1.515350e+09
2018-01-07 18:34:00        False          False  1.515350e+09
2018-01-07 18:35:00        False          False  1.515350e+09
2018-01-07 18:36:00        False          False  1.515350e+09

[1574250 rows x 21 columns]
```

```
from sklearn.model_selection import train_test_split
trainColumns=['Open', 'High', 'Low', 'Close', 'Volume_(BTC)',
'Volume_(Currency)',
       'Weighted_Price', 'Month', 'Week', 'Day',
       'Dayofweek', 'Dayofyear', 'Is_month_end', 'Is_month_start',
       'Is_quarter_end', 'Is_quarter_start', 'Is_year_end',
'Is_year_start']
predictColumn='PriceClose2D'
X=coinbase[trainColumns]
y=coinbase[predictColumn]
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.10,shuffle=False)

train_error=[]
test_error=[]
minDepth=20
maxDepth=40
models=[]
for depth in range(minDepth,maxDepth,5):
    regr=RandomForestRegressor(max_depth=depth,
random_state=0,n_estimators=5,verbose=2)
    regr.fit(X_train, y_train)
    models.append(regr)

tr_error=math.sqrt(mean_squared_error(regr.predict(X_train),y_train))

te_error=math.sqrt(mean_squared_error(regr.predict(X_test),y_test))
    test_error.append(tr_error)
    train_error.append(te_error)
    print (depth,tr_error,te_error)
```

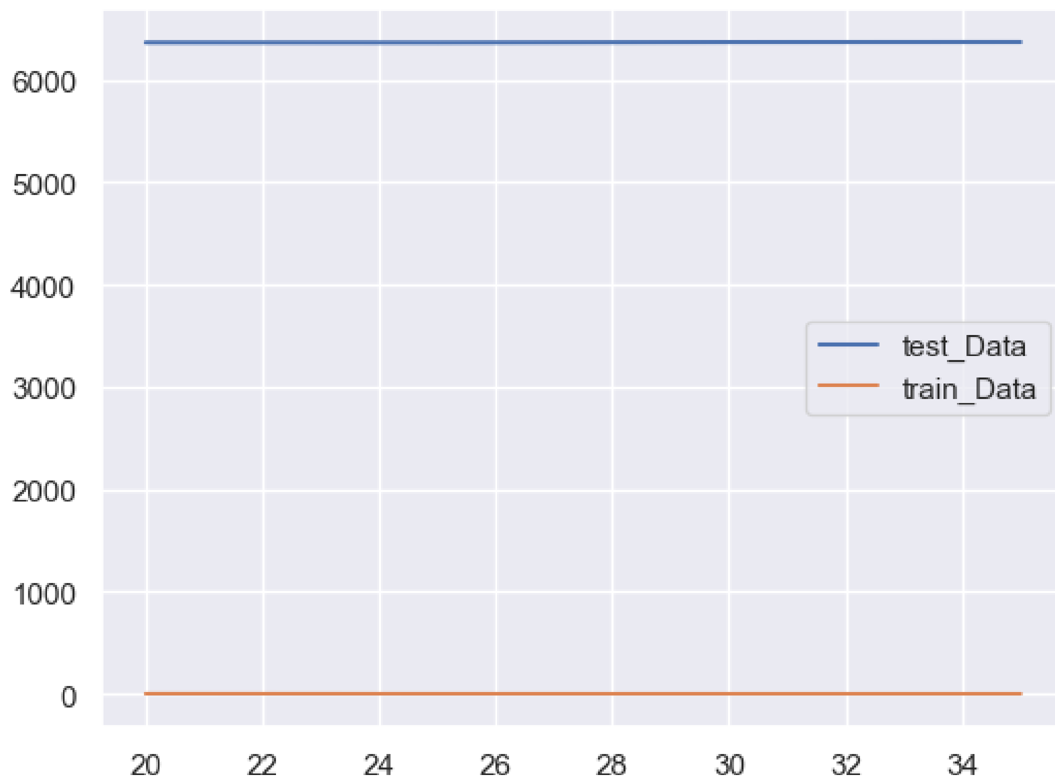
```
[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1
concurrent workers.

building tree 1 of 5

[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:   10.1s
remaining:    0.0s

building tree 2 of 5
building tree 3 of 5
```

```
building tree 4 of 5
building tree 5 of 5

[Parallel(n_jobs=1)]: Done   5 out of   5 | elapsed:   50.8s finished
[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1
concurrent workers.
[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:    0.1s
remaining:    0.0s
[Parallel(n_jobs=1)]: Done   5 out of   5 | elapsed:    0.5s finished
[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1
concurrent workers.
[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:    0.0s
remaining:    0.0s
[Parallel(n_jobs=1)]: Done   5 out of   5 | elapsed:    0.0s finished

20 4.657279640891137 6369.506613027702

[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1
concurrent workers.

building tree 1 of 5

[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:   11.4s
remaining:    0.0s

building tree 2 of 5
building tree 3 of 5
building tree 4 of 5
building tree 5 of 5

[Parallel(n_jobs=1)]: Done   5 out of   5 | elapsed:   57.3s finished
[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1
concurrent workers.
[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:    0.2s
remaining:    0.0s
[Parallel(n_jobs=1)]: Done   5 out of   5 | elapsed:    0.8s finished
[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1
concurrent workers.
[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:    0.0s
remaining:    0.0s
[Parallel(n_jobs=1)]: Done   5 out of   5 | elapsed:    0.0s finished

25 4.012775600613368 6368.654213519618

[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1
concurrent workers.

building tree 1 of 5

[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:   12.5s
remaining:    0.0s
```

```
building tree 2 of 5
building tree 3 of 5
building tree 4 of 5
building tree 5 of 5

[Parallel(n_jobs=1)]: Done   5 out of   5 | elapsed:  1.0min finished
[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1
concurrent workers.
[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:    0.3s
remaining:    0.0s
[Parallel(n_jobs=1)]: Done   5 out of   5 | elapsed:    1.3s finished
[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1
concurrent workers.
[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:    0.0s
remaining:    0.0s
[Parallel(n_jobs=1)]: Done   5 out of   5 | elapsed:    0.0s finished

30 3.852380995818222 6373.337580578217

[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1
concurrent workers.

building tree 1 of 5

[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:   13.2s
remaining:    0.0s

building tree 2 of 5
building tree 3 of 5
building tree 4 of 5
building tree 5 of 5

[Parallel(n_jobs=1)]: Done   5 out of   5 | elapsed:  1.1min finished
[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1
concurrent workers.
[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:    0.3s
remaining:    0.0s
[Parallel(n_jobs=1)]: Done   5 out of   5 | elapsed:    1.3s finished

35 3.820660838623674 6374.10942696613

[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1
concurrent workers.
[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:    0.0s
remaining:    0.0s
[Parallel(n_jobs=1)]: Done   5 out of   5 | elapsed:    0.0s finished

train_error

[6369.506613027702, 6368.654213519618, 6373.337580578217,
6374.10942696613]
```

```python
from sklearn.metrics import confusion_matrix
# print(confusion_matrix(models[2].predict(X_test), y_test))
train_plot=pd.DataFrame(train_error,index=range(20,40,5),columns=["tes
t_Data"])
test_plot=pd.DataFrame(test_error,index=range(20,40,5),columns=["train
_Data"])
plotdata=pd.concat([train_plot,test_plot],axis=1)
plotdata.plot()
X_test.size
```

```
2833650
```




```python
y_test.head()
```

```
Date
2017-09-20 11:52:00    3990.59
2017-09-20 11:53:00    3990.59
2017-09-20 11:54:00    3995.06
2017-09-20 11:55:00    3997.58
2017-09-20 11:56:00    3998.57
Name: PriceClose2D, dtype: float64
```

```python
# from sklearn.metrics import mean_squared_error
# print('testing
error',mean_squared_error(regr.predict(X_test),y_test))
```

```
# print('training
error',mean_squared_error(regr.predict(X_train),y_train))
```

## Model 2: Recurrent Neural Networks

```
import numpy
import matplotlib.pyplot as plt
from pandas import read_csv
import math
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import LSTM,GRU
from sklearn.preprocessing import
MinMaxScaler,RobustScaler,StandardScaler
from sklearn.metrics import mean_squared_error
from pandas import Series

data=pd.read_csv('Bitcoin2015Daily.csv')
data.head(3)
```

```
         Date   Open  Close   High    Low  Volume_(BTC)
Volume_(Currency)  \
0  2015-01-01  345.0  340.0  345.0  340.0           0.0
0.0
1  2015-01-02  345.0  340.0  345.0  340.0           0.0
0.0
2  2015-01-03  345.0  340.0  345.0  340.0           0.0
0.0

   Weighted_Price
0           342.5
1           342.5
2           342.5
```

```
# Prepare data. Set date as index and choose closing price as target.
data=data.set_index(pd.DatetimeIndex(data['Date']))['Close']
data.head(3)
```

```
Date
2015-01-01    340.0
2015-01-02    340.0
2015-01-03    340.0
Name: Close, dtype: float64
```

```
# make the signal stationary -- subtract the previous value from the
current value (Technical jargon: First order difference)
def difference(dataset, interval=1):
     diff = list()
     for i in range(interval, len(dataset)):
          value = dataset[i] - dataset[i - interval]
```

```
            diff.append(value)
      return Series(diff)

look_back=3
#data=difference(data,look_back)

#convert an array of values into a dataset matrix
def create_dataset(dataset, look_back=1):
    dataX, dataY = [], []
    for i in range(len(dataset)-look_back-1):
        #takes
        a = dataset[i:(i+look_back), 0]
        dataX.append(a)
        dataY.append(dataset[i+look_back, 0])
    return numpy.array(dataX), numpy.array(dataY)

# fix random seed for reproducibility
numpy.random.seed(0)

# load the dataset
dataframe = data
dataset = dataframe.values
dataset = dataset.astype('float64').reshape(-1, 1)
dataset
```

```
array([[  340.  ],
       [  340.  ],
       [  340.  ],
       ...,
       [16550.02],
       [16635.31],
       [16266.06]])
```

```
# normalize the dataset
scaler = MinMaxScaler()
#scaler=RobustScaler()
#scaler=StandardScaler()
dataset = scaler.fit_transform(dataset)
dataset
```

```
array([[0.00988583],
       [0.00988583],
       [0.00988583],
       ...,
       [0.85330463],
       [0.85774233],
       [0.83852999]])
```

```
# split into train and test sets
train_size = int(len(dataset) * 0.67)
test_size = len(dataset) - train_size
```

```
train, test = dataset[0:train_size,:],
dataset[train_size:len(dataset),:]
print(len(train), len(test))

739 364

# reshape into X=t and Y=t+1
trainX, trainY = create_dataset(train, look_back)
testX, testY = create_dataset(test, look_back)
print(len(trainX), len(testX))

735 360

# trainX

# reshape input to be [samples, time steps, features]
trainX = numpy.reshape(trainX, (trainX.shape[0], 1, trainX.shape[1]))
testX = numpy.reshape(testX, (testX.shape[0], 1, testX.shape[1]))

# testX

# create and fit the LSTM network
from keras.layers import Activation, Dense,Dropout
model = Sequential()

model.add(LSTM(256, return_sequences=True,input_shape=(1, look_back)))
#model.add(LSTM(256, return_sequences=True,input_shape=(1,
look_back)))
model.add(LSTM(256))
#model.add(LSTM(100, input_shape=(1, look_back)))

model.add(Dense(1))

import keras
from keras import optimizers

model.compile(loss='mean_squared_error', optimizer='adam')
model.fit(trainX, trainY, epochs=50,
verbose=1,shuffle=False,batch_size=50)

Epoch 1/50
15/15 [==============================] - 3s 6ms/step - loss: 3.3415e-
05
Epoch 2/50
15/15 [==============================] - 0s 6ms/step - loss: 4.8547e-
04
Epoch 3/50
15/15 [==============================] - 0s 7ms/step - loss: 8.7393e-
05
Epoch 4/50
15/15 [==============================] - 0s 7ms/step - loss: 1.7436e-
04
```

```
Epoch 5/50
15/15 [==============================] - 0s 5ms/step - loss: 1.1437e-
04
Epoch 6/50
15/15 [==============================] - 0s 5ms/step - loss: 1.2203e-
04
Epoch 7/50
15/15 [==============================] - 0s 5ms/step - loss: 1.0580e-
04
Epoch 8/50
15/15 [==============================] - 0s 5ms/step - loss: 9.1606e-
05
Epoch 9/50
15/15 [==============================] - 0s 5ms/step - loss: 7.5145e-
05
Epoch 10/50
15/15 [==============================] - 0s 5ms/step - loss: 5.1899e-
05
Epoch 11/50
15/15 [==============================] - 0s 5ms/step - loss: 2.8266e-
05
Epoch 12/50
15/15 [==============================] - 0s 5ms/step - loss: 8.9846e-
06
Epoch 13/50
15/15 [==============================] - 0s 5ms/step - loss: 2.3169e-
06
Epoch 14/50
15/15 [==============================] - 0s 5ms/step - loss: 8.2728e-
06
Epoch 15/50
15/15 [==============================] - 0s 5ms/step - loss: 9.2826e-
06
Epoch 16/50
15/15 [==============================] - 0s 5ms/step - loss: 4.4700e-
06
Epoch 17/50
15/15 [==============================] - 0s 5ms/step - loss: 1.4589e-
06
Epoch 18/50
15/15 [==============================] - 0s 5ms/step - loss: 1.0217e-
06
Epoch 19/50
15/15 [==============================] - 0s 5ms/step - loss: 1.2147e-
06
Epoch 20/50
15/15 [==============================] - 0s 5ms/step - loss: 1.2074e-
06
Epoch 21/50
```

```
15/15 [==============================] - 0s 5ms/step - loss: 1.1188e-
06
Epoch 22/50
15/15 [==============================] - 0s 5ms/step - loss: 1.0734e-
06
Epoch 23/50
15/15 [==============================] - 0s 5ms/step - loss: 1.0654e-
06
Epoch 24/50
15/15 [==============================] - 0s 5ms/step - loss: 1.0746e-
06
Epoch 25/50
15/15 [==============================] - 0s 5ms/step - loss: 1.0906e-
06
Epoch 26/50
15/15 [==============================] - 0s 6ms/step - loss: 1.1134e-
06
Epoch 27/50
15/15 [==============================] - 0s 6ms/step - loss: 1.1493e-
06
Epoch 28/50
15/15 [==============================] - 0s 8ms/step - loss: 1.2068e-
06
Epoch 29/50
15/15 [==============================] - 0s 18ms/step - loss: 1.2987e-
06
Epoch 30/50
15/15 [==============================] - 0s 8ms/step - loss: 1.4476e-
06
Epoch 31/50
15/15 [==============================] - 0s 5ms/step - loss: 1.6951e-
06
Epoch 32/50
15/15 [==============================] - 0s 5ms/step - loss: 2.1181e-
06
Epoch 33/50
15/15 [==============================] - 0s 5ms/step - loss: 2.8661e-
06
Epoch 34/50
15/15 [==============================] - 0s 5ms/step - loss: 4.2427e-
06
Epoch 35/50
15/15 [==============================] - 0s 10ms/step - loss: 6.8920e-
06
Epoch 36/50
15/15 [==============================] - 0s 8ms/step - loss: 1.2205e-
05
Epoch 37/50
15/15 [==============================] - 0s 7ms/step - loss: 2.3027e-
```

```
05
Epoch 38/50
15/15 [==============================] - 0s 7ms/step - loss: 4.3362e-
05
Epoch 39/50
15/15 [==============================] - 0s 7ms/step - loss: 6.6806e-
05
Epoch 40/50
15/15 [==============================] - 0s 7ms/step - loss: 5.8458e-
05
Epoch 41/50
15/15 [==============================] - 0s 7ms/step - loss: 1.9142e-
05
Epoch 42/50
15/15 [==============================] - 0s 9ms/step - loss: 3.6473e-
06
Epoch 43/50
15/15 [==============================] - 0s 7ms/step - loss: 1.5772e-
05
Epoch 44/50
15/15 [==============================] - 0s 5ms/step - loss: 7.9682e-
06
Epoch 45/50
15/15 [==============================] - 0s 5ms/step - loss: 1.3699e-
06
Epoch 46/50
15/15 [==============================] - 0s 5ms/step - loss: 1.6211e-
06
Epoch 47/50
15/15 [==============================] - 0s 6ms/step - loss: 1.8391e-
06
Epoch 48/50
15/15 [==============================] - 0s 6ms/step - loss: 1.0952e-
06
Epoch 49/50
15/15 [==============================] - 0s 10ms/step - loss: 1.1431e-
06
Epoch 50/50
15/15 [==============================] - 0s 16ms/step - loss: 1.1468e-
06

<keras.callbacks.History at 0x7f8de1b17a00>
```

```python
# make predictions
trainPredict = model.predict(trainX)
testPredict = model.predict(testX)

# invert predictions
trainPredict = scaler.inverse_transform(trainPredict)
trainY = scaler.inverse_transform([trainY])
```

```
testPredict = scaler.inverse_transform(testPredict)
testY = scaler.inverse_transform([testY])

# calculate root mean squared error
trainScore = math.sqrt(mean_squared_error(trainY[0],
trainPredict[:,0]))
print('Train Score: %.2f RMSE' % (trainScore))
testScore = math.sqrt(mean_squared_error(testY[0], testPredict[:,0]))
print('Test Score: %.2f RMSE' % (testScore))
```

```
Train Score: 19.83 RMSE
Test Score: 570.53 RMSE
```

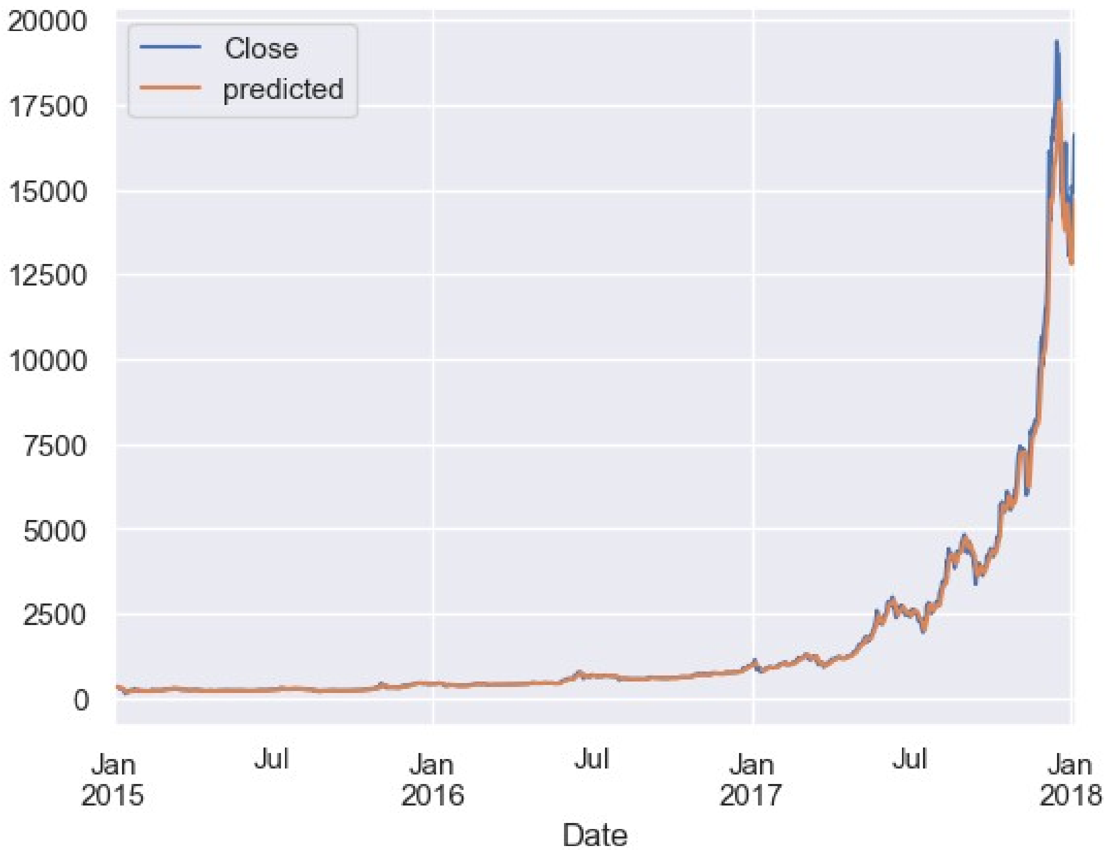
```
predictions = numpy.empty_like(dataset)
predictions[:, :] = numpy.nan
predictions[look_back:len(trainPredict)+look_back, :] = trainPredict
predictions[len(trainPredict)+(look_back*2)+1:len(dataset)-1, :] =
testPredict
#data=pd.DataFrame(numpy.concatenate((trainPredict[0:len(trainPredict)
-look_back-1],testPredict[0:len(testPredict)-look_back-
1])),columns=["predicted"])
#print('one',data.count())
#print('two',dataframe.count())
predictionsDF=pd.DataFrame(predictions,columns=["predicted"],index=dat
aframe.index)
ans=pd.concat([dataframe,predictionsDF],axis=1)
print( ans,[look_back,trainScore,testScore])
```

```
                 Close     predicted
Date
2015-01-01    340.00           NaN
2015-01-02    340.00           NaN
2015-01-03    340.00           NaN
2015-01-04    340.00    336.613922
2015-01-05    340.00    336.613922
...              ...           ...
2018-01-03  14986.76  13404.695312
2018-01-04  14938.79  13813.471680
2018-01-05  16550.02  14288.621094
2018-01-06  16635.31  14716.493164
2018-01-07  16266.06           NaN

[1103 rows x 2 columns] [3, 19.832386331414433, 570.5267779114859]
```

Let's plot and compare the prices predicted and actual price.

```
ans.plot()
```

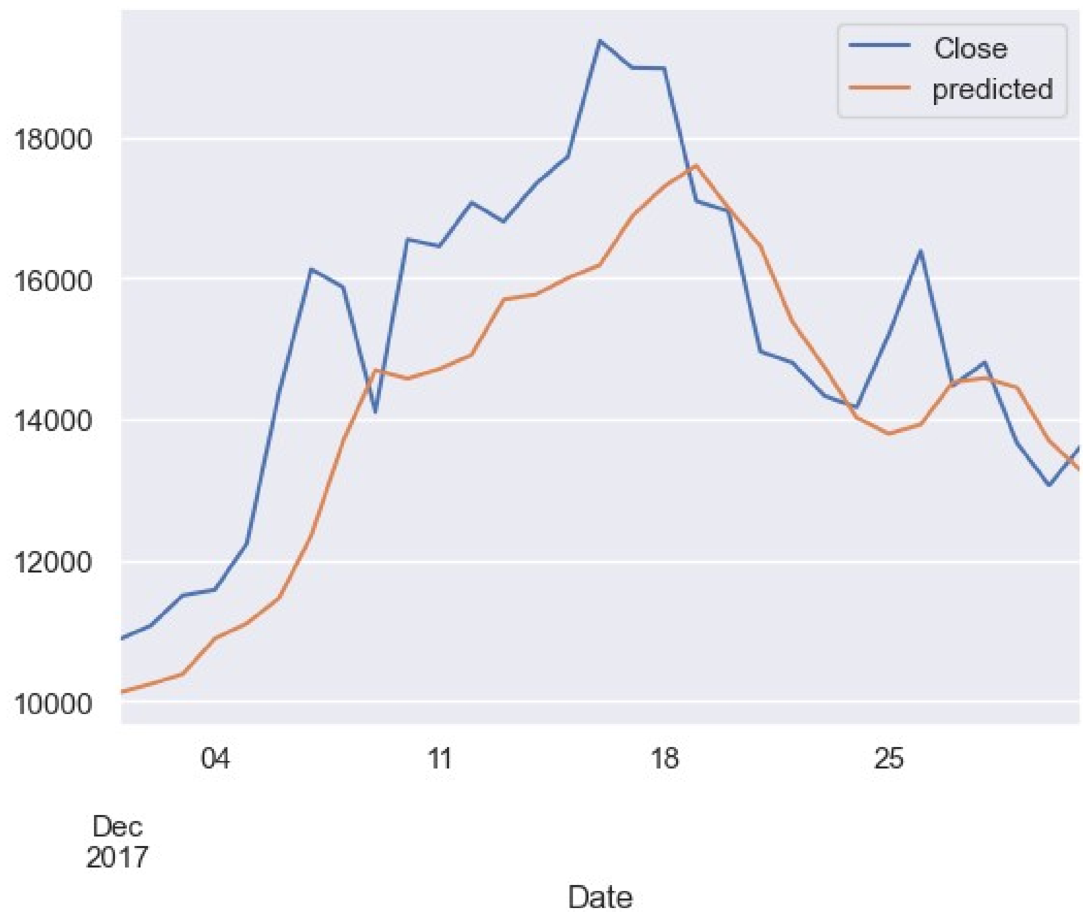
```
<AxesSubplot:xlabel='Date'>
```




```
ans['2017-12'].plot()
```

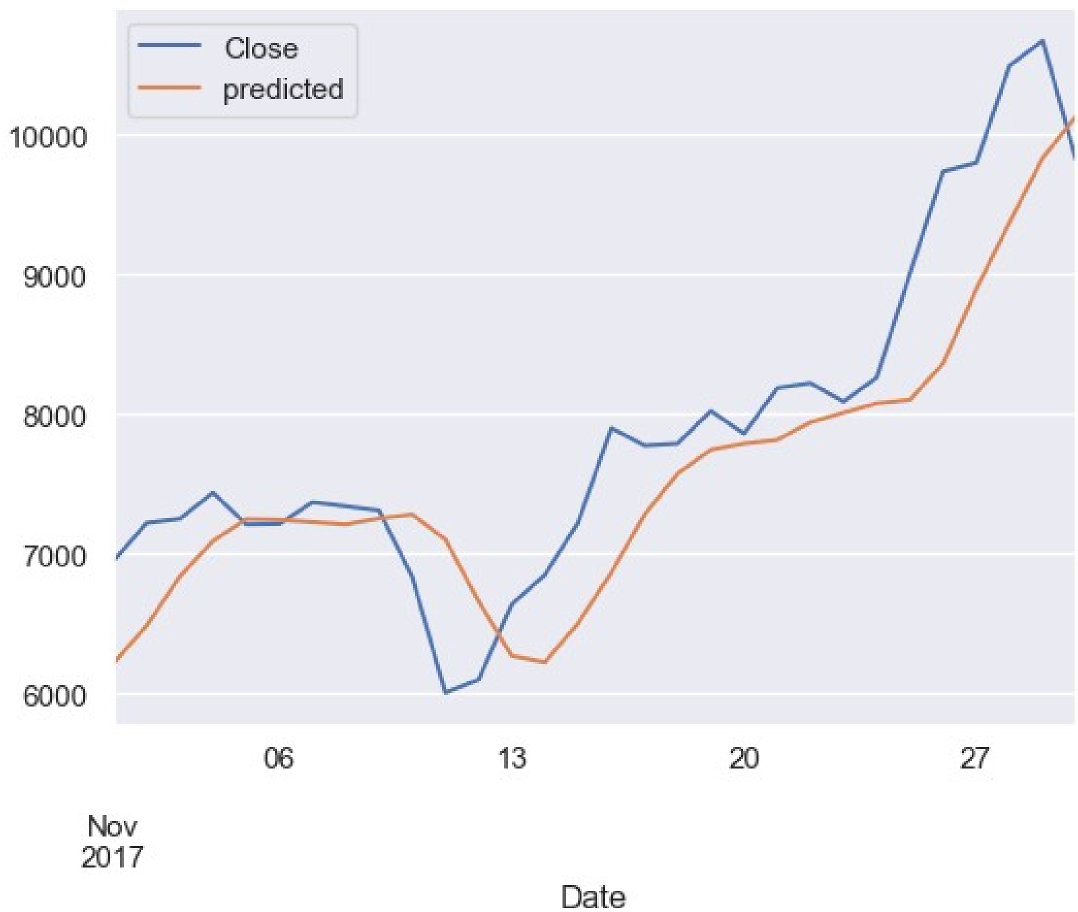
```
<ipython-input-57-4067f2678e90>:1: FutureWarning: Indexing a DataFrame
with a datetimelike index using a single string to slice the rows,
like `frame[string]`, is deprecated and will be removed in a future
version. Use `frame.loc[string]` instead.
  ans['2017-12'].plot()
```

```
<AxesSubplot:xlabel='Date'>
```



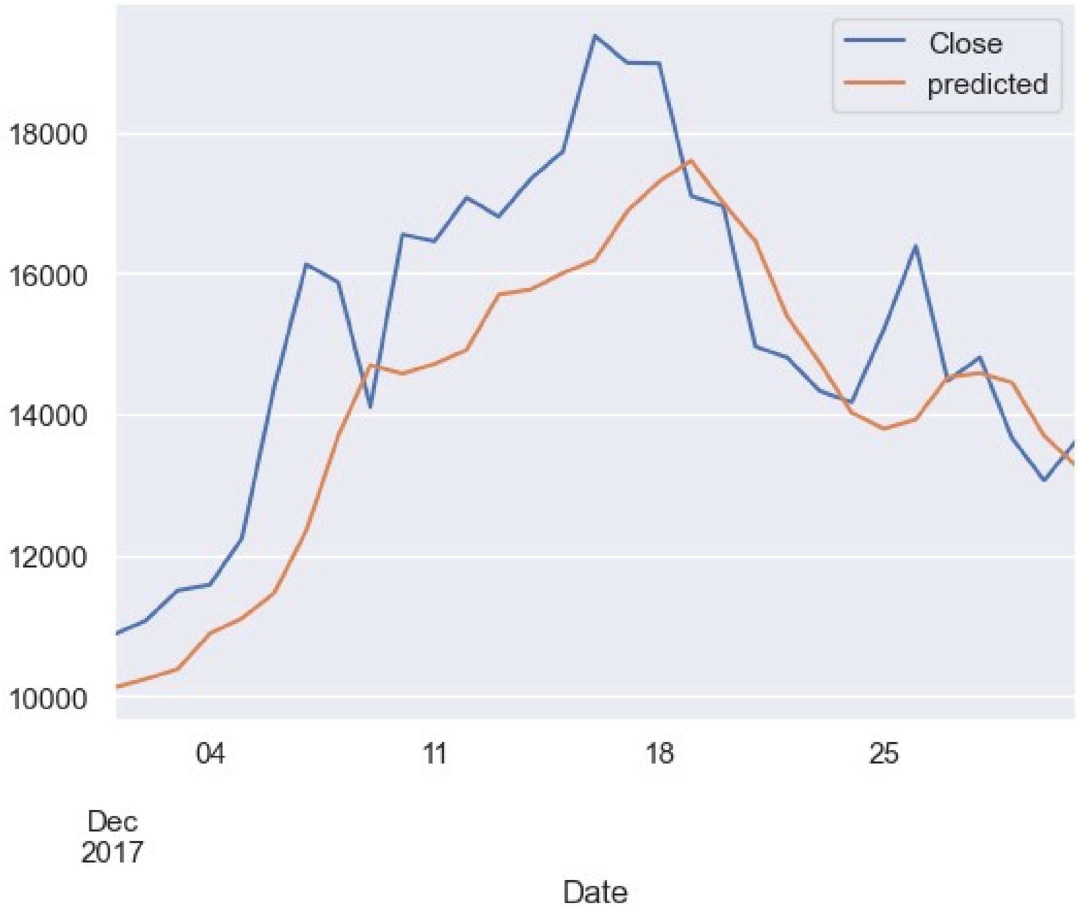

```
ans['2017-11'].plot()
```

```
<ipython-input-58-2aaf386c1058>:1: FutureWarning: Indexing a DataFrame
with a datetimelike index using a single string to slice the rows,
like `frame[string]`, is deprecated and will be removed in a future
version. Use `frame.loc[string]` instead.
  ans['2017-11'].plot()

<AxesSubplot:xlabel='Date'>
```

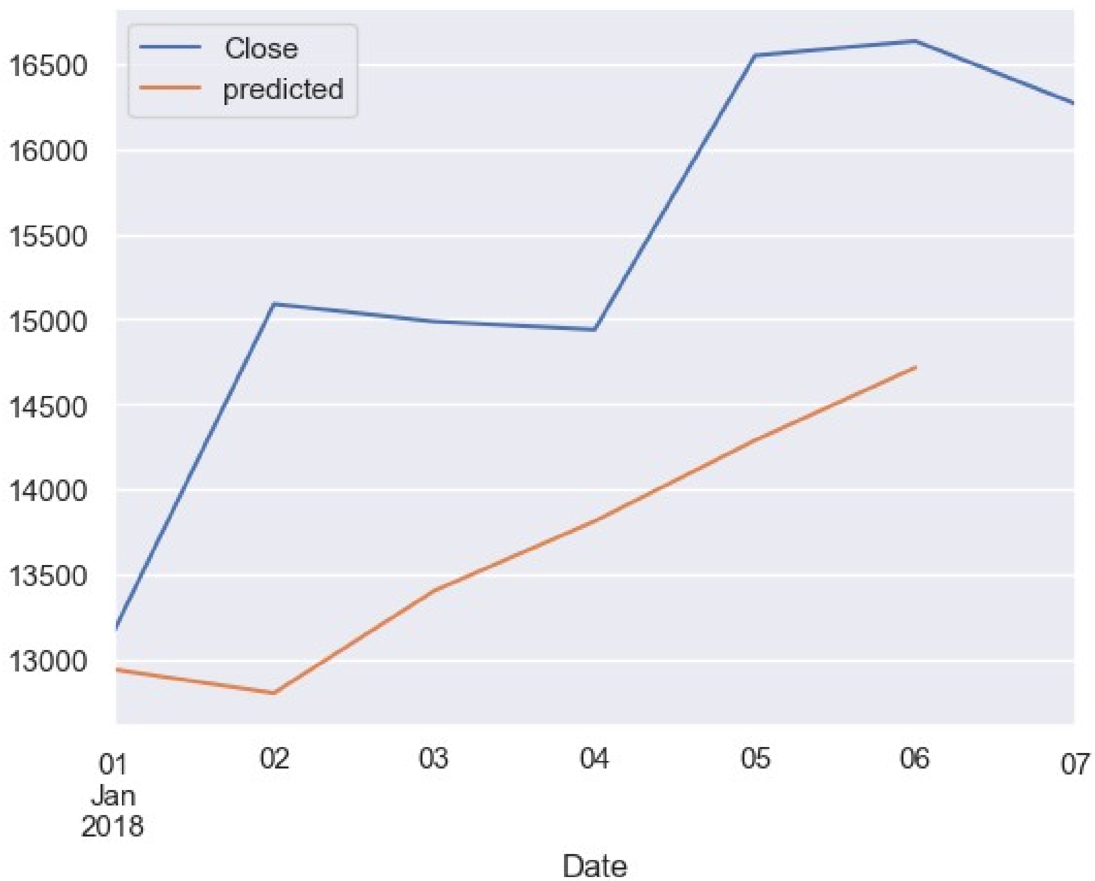



```
ans['2017-12'].plot()
```

```
<ipython-input-59-05412888f0a6>:1: FutureWarning: Indexing a DataFrame with a datetimelike index using a single string to slice the rows, like `frame[string]`, is deprecated and will be removed in a future version. Use `frame.loc[string]` instead.
  ans['2017-12'].plot()

<AxesSubplot:xlabel='Date'>
```



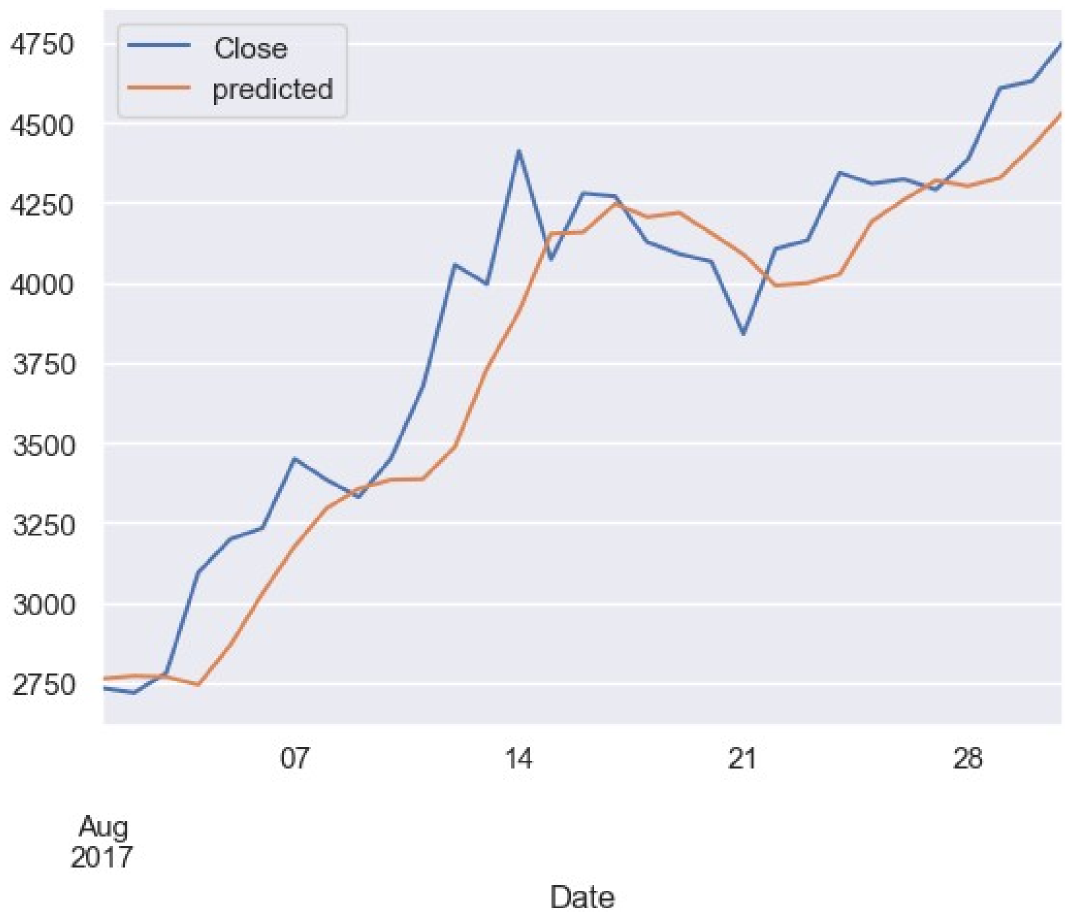

```
ans['2018-01'].plot()

<ipython-input-60-d26fce375b4e>:1: FutureWarning: Indexing a DataFrame
with a datetimelike index using a single string to slice the rows,
like `frame[string]`, is deprecated and will be removed in a future
version. Use `frame.loc[string]` instead.
  ans['2018-01'].plot()

<AxesSubplot:xlabel='Date'>
```




```
ans['2017-08'].plot()
```

```
<ipython-input-61-23e9907c9217>:1: FutureWarning: Indexing a DataFrame
with a datetimelike index using a single string to slice the rows,
like `frame[string]`, is deprecated and will be removed in a future
version. Use `frame.loc[string]` instead.
  ans['2017-08'].plot()
```

```
<AxesSubplot:xlabel='Date'>
```

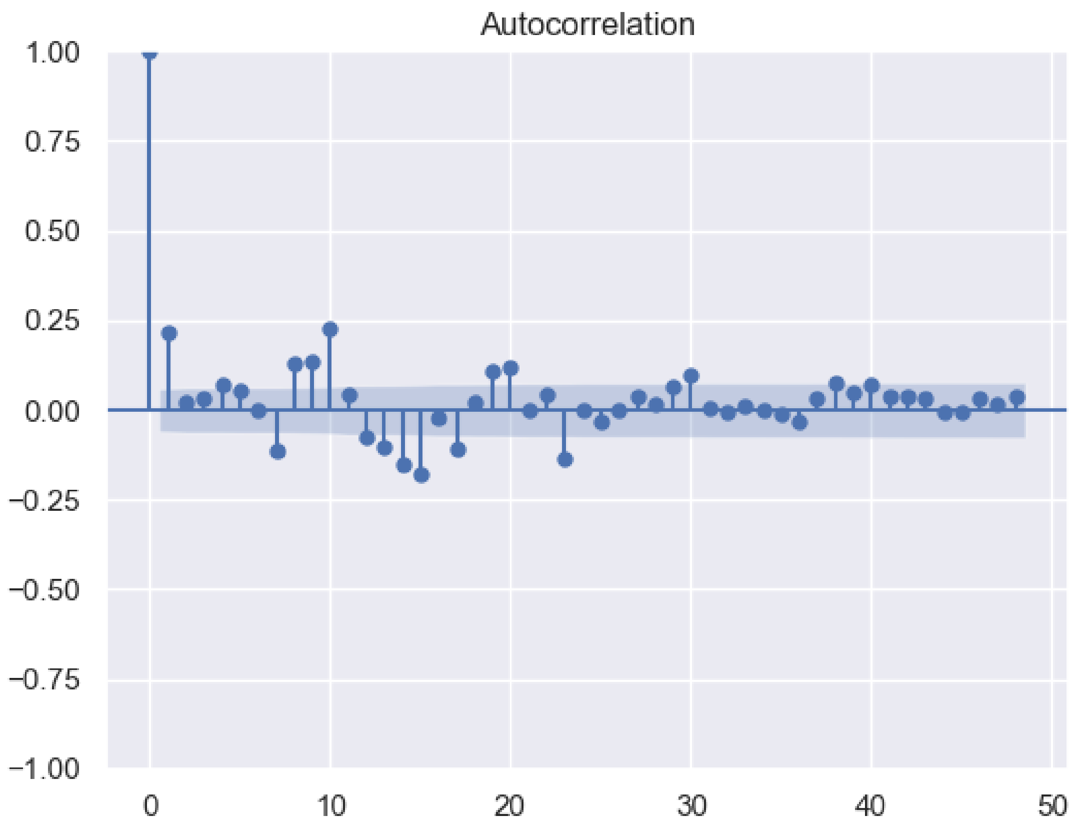


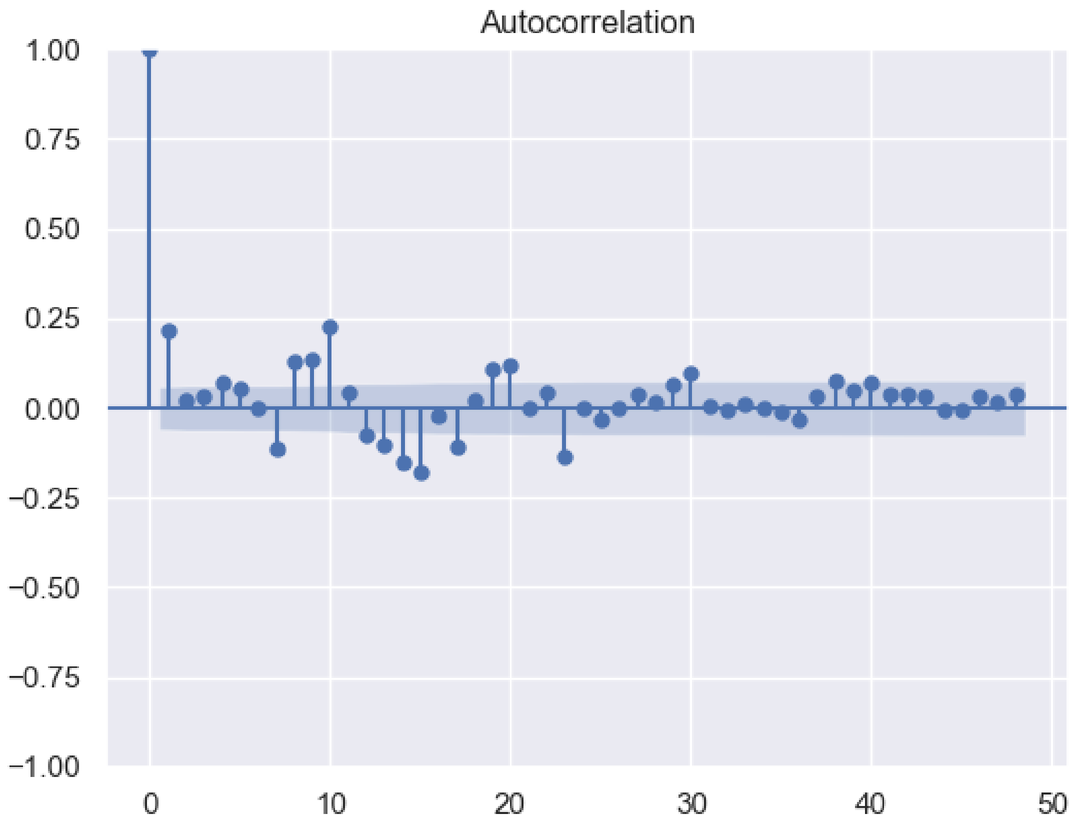

# Model 3: ARIMA MODEL

```
arima2015hour=coinbase['2015':].resample('D').mean().fillna(method='ff
ill')['Close']

<ipython-input-62-9e0c7cd310a2>:1: FutureWarning: Value based partial
slicing on non-monotonic DatetimeIndexes with non-existing keys is
deprecated and will raise a KeyError in a future Version.

arima2015hour=coinbase['2015':].resample('D').mean().fillna(method='ff
ill')['Close']

arima2015hour

Date
2015-01-07      302.148426
2015-01-08      288.934674
2015-01-09      288.934674
2015-01-10      288.934674
2015-01-11      288.934674
                   ...
```

```
2018-01-03    15033.472139
2018-01-04    14825.769069
2018-01-05    16150.067569
2018-01-06    16691.451653
2018-01-07    16440.150027
Freq: D, Name: Close, Length: 1097, dtype: float64
```

```python
from statsmodels.graphics.tsaplots import plot_acf
#hourarim=arima2015hour.resample('H').mean()['Close']
plot_acf(arima2015hour.diff().dropna(), lags= 48, alpha=0.05)
#arima2015hour['Close'].pct_change().autocorr()
#dayarima.diff()
#arima2015hour.dropna().diff().plot()
```






```python
from statsmodels.tsa.stattools import adfuller
adfuller(arima2015hour)[1]
```

```
0.9987867392172342
```

```python
from sklearn.metrics import mean_squared_error
from sklearn.model_selection import train_test_split
train,test =
train_test_split(arima2015hour,test_size=0.24,shuffle=False)
print(train)
```

```
Date
2015-01-07     302.148426
2015-01-08     288.934674
2015-01-09     288.934674
2015-01-10     288.934674
2015-01-11     288.934674
                  ...
2017-04-14    1181.642604
2017-04-15    1181.556806
2017-04-16    1181.432667
2017-04-17    1183.139410
2017-04-18    1200.347361
Freq: D, Name: Close, Length: 833, dtype: float64
```

```python
# from statsmodels.tsa.arima_model import ARMA
from statsmodels.tsa.arima.model import ARIMA
mod = ARIMA(arima2015hour, order=(4,4,0))
result = mod.fit()

history = [x for x in train]
predictions = list()
for i in range(len(test)):
    # predict
    model = ARIMA(history,order=(5,1,0))
    model_fit = mod.fit()
    yhat = model_fit.forecast()
    yhat_p = model_fit.predict(start=len(history), end=len(history))
[0]
    predictions.append(yhat_p)
    # observation
    obs = test[i]
    history.append(obs)
    print(str(yhat_p)+' '+' '+ str(history[-4:])+' '+str(obs)+'
'+str(i)+' ')
```

```
1233.5311977089639  [1181.4326666666666, 1183.1394097222221,
1200.347361111111, 1203.6566319444444] 1203.6566319444444 0
1196.4491797395754  [1183.1394097222221, 1200.347361111111,
1203.6566319444444, 1229.8685] 1229.8685 1
1264.5372030348337  [1200.347361111111, 1203.6566319444444, 1229.8685,
1248.6709097222224] 1248.6709097222224 2
1273.6717663141067  [1203.6566319444444, 1229.8685,
1248.6709097222224, 1244.4463333333333] 1244.4463333333333 3
1229.5095768231517  [1229.8685, 1248.6709097222224,
1244.4463333333333, 1248.4556458333334] 1248.4556458333334 4
1239.2875211576163  [1248.6709097222224, 1244.4463333333333,
1248.4556458333334, 1254.437798611111] 1254.437798611111 5
1255.9847788294128  [1244.4463333333333, 1248.4556458333334,
1254.437798611111, 1277.1488333333334] 1277.1488333333334 6
1300.1981485471072  [1248.4556458333334, 1254.437798611111,
1277.1488333333334, 1298.5008819444445] 1298.5008819444445 7
1339.181061742929  [1254.437798611111, 1277.1488333333334,
1298.5008819444445, 1334.0329652777777] 1334.0329652777777 8
1390.7687938842123  [1277.1488333333334, 1298.5008819444445,
1334.0329652777777, 1339.6038472222222] 1339.6038472222222 9
1338.416686529859  [1298.5008819444445, 1334.0329652777777,
1339.6038472222222, 1357.0315555555555] 1357.0315555555555 10
1363.8044017823427  [1334.0329652777777, 1339.6038472222222,
1357.0315555555555, 1360.206548611111] 1360.206548611111 11
1346.337078806807  [1339.6038472222222, 1357.0315555555555,
1360.206548611111, 1432.8617708333334] 1432.8617708333334 12
1525.9874615540734  [1357.0315555555555, 1360.206548611111,
1432.8617708333334, 1466.8994375] 1466.8994375 13
1520.5175389863891  [1360.206548611111, 1432.8617708333334,
```

```
1466.8994375, 1492.898576388889] 1492.898576388889 14
1531.1036673918538  [1432.8617708333334, 1466.8994375,
1492.898576388889, 1579.7366944444443] 1579.7366944444443 15
1680.8903449311365  [1466.8994375, 1492.898576388889,
1579.7366944444443, 1589.2706180555556] 1589.2706180555556 16
1582.271759211965  [1492.898576388889, 1579.7366944444443,
1589.2706180555556, 1586.8826180555557] 1586.8826180555557 17
1537.7327707704958  [1579.7366944444443, 1589.2706180555556,
1586.8826180555557, 1595.6256180555556] 1595.6256180555556 18
1596.0902258353499  [1589.2706180555556, 1586.8826180555557,
1595.6256180555556, 1653.2495416666666] 1653.2495416666666 19
1712.295650254206  [1586.8826180555557, 1595.6256180555556,
1653.2495416666666, 1735.7519305555554] 1735.7519305555554 20
1851.2398753682464  [1595.6256180555556, 1653.2495416666666,
1735.7519305555554, 1767.118125] 1767.118125 21
1848.9306580762106  [1653.2495416666666, 1735.7519305555554,
1767.118125, 1844.0285625] 1844.0285625 22
1932.5446804088358  [1735.7519305555554, 1767.118125, 1844.0285625,
1747.6833888888889] 1747.6833888888889 23
1568.010237766609  [1767.118125, 1844.0285625, 1747.6833888888889,
1738.5881458333336] 1738.5881458333336 24
1652.2212391171333  [1844.0285625, 1747.6833888888889,
1738.5881458333336, 1798.8772708333333] 1798.8772708333333 25
1872.4603139886974  [1747.6833888888889, 1738.5881458333336,
1798.8772708333333, 1743.3113055555555] 1743.3113055555555 26
1700.0572454577182  [1738.5881458333336, 1798.8772708333333,
1743.3113055555555, 1752.6138888888888] 1752.6138888888888 27
1783.5288023240003  [1798.8772708333333, 1743.3113055555555,
1752.6138888888888, 1821.8831319444444] 1821.8831319444444 28
1971.446905465915  [1743.3113055555555, 1752.6138888888888,
1821.8831319444444, 1853.3229930555556] 1853.3229930555556 29
1861.896292540223  [1752.6138888888888, 1821.8831319444444,
1853.3229930555556, 1950.512076388889] 1950.512076388889 30
2085.8970536275547  [1821.8831319444444, 1853.3229930555556,
1950.512076388889, 2013.7615] 2013.7615 31
2129.279440055406  [1853.3229930555556, 1950.512076388889, 2013.7615,
2057.812270833333] 2057.812270833333 32
2056.3205764109653  [1950.512076388889, 2013.7615, 2057.812270833333,
2180.656097222222] 2180.656097222222 33
2305.162047413192  [2013.7615, 2057.812270833333, 2180.656097222222,
2241.8767152777777] 2241.8767152777777 34
2310.68443854946  [2057.812270833333, 2180.656097222222,
2241.8767152777777, 2404.0440694444446] 2404.0440694444446 35
2585.418147875304  [2180.656097222222, 2241.8767152777777,
2404.0440694444446, 2594.9237013888887] 2594.9237013888887 36
2861.7974904122716  [2241.8767152777777, 2404.0440694444446,
2594.9237013888887, 2413.1511111111113] 2413.1511111111113 37
2109.397913730734  [2404.0440694444446, 2594.9237013888887,
2413.1511111111113, 2107.2662083333335] 2107.2662083333335 38
```

```
1535.3008317513663  [2594.9237013888887, 2413.1511111111113,
2107.2662083333335, 2236.9278958333334] 2236.9278958333334 39
2332.7458313700827  [2413.1511111111113, 2107.2662083333335,
2236.9278958333334, 2264.007013888889] 2264.007013888889 40
2338.9510216042995  [2107.2662083333335, 2236.9278958333334,
2264.007013888889, 2256.295465277778] 2256.295465277778 41
2409.283124588503  [2236.9278958333334, 2264.007013888889,
2256.295465277778, 2273.0040277777775] 2273.0040277777775 42
2488.284110183724  [2264.007013888889, 2256.295465277778,
2273.0040277777775, 2415.8703958333335] 2415.8703958333335 43
2559.755588053562  [2256.295465277778, 2273.0040277777775,
2415.8703958333335, 2437.727319444444] 2437.727319444444 44
2321.9923426089663  [2273.0040277777775, 2415.8703958333335,
2437.727319444444, 2527.356041666667] 2527.356041666667 45
2685.845388726714  [2415.8703958333335, 2437.727319444444,
2527.356041666667, 2526.7166319444445] 2526.7166319444445 46
2502.6375845883863  [2437.727319444444, 2527.356041666667,
2526.7166319444445, 2636.0438263888886] 2636.0438263888886 47
2714.9686578448363  [2527.356041666667, 2526.7166319444445,
2636.0438263888886, 2850.8538819444443] 2850.8538819444443 48
3119.8823324877894  [2526.7166319444445, 2636.0438263888886,
2850.8538819444443, 2789.415673611111] 2789.415673611111 49
2738.6890262166535  [2636.0438263888886, 2850.8538819444443,
2789.415673611111, 2769.859229166667] 2769.859229166667 50
2660.2754810083607  [2850.8538819444443, 2789.415673611111,
2769.859229166667, 2824.5085763888887] 2824.5085763888887 51
2847.688324149715  [2789.415673611111, 2769.859229166667,
2824.5085763888887, 2872.589375] 2872.589375 52
2867.180139486781  [2769.859229166667, 2824.5085763888887,
2872.589375, 2942.9690555555558] 2942.9690555555558 53
3071.157638176443  [2824.5085763888887, 2872.589375,
2942.9690555555558, 2774.581277777778] 2774.581277777778 54
2630.846501063523  [2872.589375, 2942.9690555555558,
2774.581277777778, 2727.7749027777777] 2727.7749027777777 55
2570.0522736948265  [2942.9690555555558, 2774.581277777778,
2727.7749027777777, 2621.745722222222] 2621.745722222222 56
2429.5716200297893  [2774.581277777778, 2727.7749027777777,
2621.745722222222, 2333.815798611111] 2333.815798611111 57
1943.8031617348106  [2727.7749027777777, 2621.745722222222,
2333.815798611111, 2457.931395833333] 2457.931395833333 58
2691.1063691204317  [2621.745722222222, 2333.815798611111,
2457.931395833333, 2586.183145833333] 2586.183145833333 59
2945.259684619778  [2333.815798611111, 2457.931395833333,
2586.183145833333, 2547.8353958333337] 2547.8353958333337 60
2550.7294856482827  [2457.931395833333, 2586.183145833333,
2547.8353958333337, 2565.492277777778] 2565.492277777778 61
2640.462621675172  [2586.183145833333, 2547.8353958333337,
2565.492277777778, 2671.0210625] 2671.0210625 62
2767.6963165091897  [2547.8353958333337, 2565.492277777778,
```

```
2671.0210625, 2678.73575] 2678.73575 63
2517.2756262246244  [2565.492277777778, 2671.0210625, 2678.73575,
2677.321270833333] 2677.321270833333 64
2694.6223628058256  [2671.0210625, 2678.73575, 2677.321270833333,
2704.0282500000003] 2704.0282500000003 65
2767.9185894871457  [2678.73575, 2677.321270833333,
2704.0282500000003, 2632.3170416666667] 2632.3170416666667 66
2471.9421108513534  [2677.321270833333, 2704.0282500000003,
2632.3170416666667, 2554.0734930555554] 2554.0734930555554 67
2408.951201223086  [2704.0282500000003, 2632.3170416666667,
2554.0734930555554, 2433.8382291666667] 2433.8382291666667 68
2295.74780226473  [2632.3170416666667, 2554.0734930555554,
2433.8382291666667, 2408.231861111111] 2408.231861111111 69
2379.3036985549857  [2554.0734930555554, 2433.8382291666667,
2408.231861111111, 2541.074326388889] 2541.074326388889 70
2788.7778292337152  [2433.8382291666667, 2408.231861111111,
2541.074326388889, 2554.110701388889] 2554.110701388889 71

2685.1524163235895  [2408.231861111111, 2541.074326388889,
2554.110701388889, 2498.9669375000003] 2498.9669375000003 72
2439.337101813411  [2541.074326388889, 2554.110701388889,
2498.9669375000003, 2452.1534097222225] 2452.1534097222225 73
2339.848494021357  [2554.110701388889, 2498.9669375000003,
2452.1534097222225, 2469.5617847222225] 2469.5617847222225 74
2398.320487287905  [2498.9669375000003, 2452.1534097222225,
2469.5617847222225, 2534.098986111111] 2534.098986111111 75
2607.715099902971  [2452.1534097222225, 2469.5617847222225,
2534.098986111111, 2598.409180555556] 2598.409180555556 76
2784.766833200126  [2469.5617847222225, 2534.098986111111,
2598.409180555556, 2586.3999444444444] 2586.3999444444444 77
2598.5718275335657  [2534.098986111111, 2598.409180555556,
2586.3999444444444, 2598.2301736111112] 2598.2301736111112 78
2566.3508099847936  [2598.409180555556, 2586.3999444444444,
2598.2301736111112, 2533.6880694444444] 2533.6880694444444 79
2378.782151793359  [2586.3999444444444, 2598.2301736111112,
2533.6880694444444, 2531.865479166667] 2531.865479166667 80
2491.2613727455637  [2598.2301736111112, 2533.6880694444444,
2531.865479166667, 2542.6523680555556] 2542.6523680555556 81
2582.078766610169  [2533.6880694444444, 2531.865479166667,
2542.6523680555556, 2431.2631944444443] 2431.2631944444443 82
2315.8752590247577  [2531.865479166667, 2542.6523680555556,
2431.2631944444443, 2333.194284722222] 2333.194284722222 83
2190.4457181044595  [2542.6523680555556, 2431.2631944444443,
2333.194284722222, 2358.6999166666665] 2358.6999166666665 84
2414.5391554538323  [2431.2631944444443, 2333.194284722222,
2358.6999166666665, 2352.3587222222222] 2352.3587222222222 85
2361.7084366699432  [2333.194284722222, 2358.6999166666665,
2352.3587222222222, 2252.127972222222] 2252.127972222222 86
2167.9138060974947  [2358.6999166666665, 2352.3587222222222,
2252.127972222222, 2051.7815555555553] 2051.7815555555553 87
```

```
1806.6226716859132  [2352.3587222222222, 2252.127972222222,
2051.7815555555553, 1919.1934652777777] 1919.1934652777777 88
1680.1842992167567  [2252.127972222222, 2051.7815555555553,
1919.1934652777777, 2092.9771458333335] 2092.9771458333335 89
2337.6472985512355  [2051.7815555555553, 1919.1934652777777,
2092.9771458333335, 2288.6526458333333] 2288.6526458333333 90
2720.9245646802015  [1919.1934652777777, 2092.9771458333335,
2288.6526458333333, 2311.8397083333334] 2311.8397083333334 91
2477.3870556784254  [2092.9771458333335, 2288.6526458333333,
2311.8397083333334, 2561.0707291666668] 2561.0707291666668 92
2896.0638127499296  [2288.6526458333333, 2311.8397083333334,
2561.0707291666668, 2712.290625] 2712.290625 93
2788.619675765446  [2311.8397083333334, 2561.0707291666668,
2712.290625, 2797.827402777778] 2797.827402777778 94
2733.9535000149626  [2561.0707291666668, 2712.290625,
2797.827402777778, 2767.828611111111] 2767.828611111111 95
2667.5168467804483  [2712.290625, 2797.827402777778,
2767.828611111111, 2759.0903194444445] 2759.0903194444445 96
2673.5790931646443  [2797.827402777778, 2767.828611111111,
2759.0903194444445, 2590.062986111111] 2590.062986111111 97
2225.7338447296515  [2767.828611111111, 2759.0903194444445,
2590.062986111111, 2496.3023749999998] 2496.3023749999998 98
2387.606549970251  [2759.0903194444445, 2590.062986111111,
2496.3023749999998, 2593.9644930555555] 2593.9644930555555 99
2805.484209804464  [2590.062986111111, 2496.3023749999998,
2593.9644930555555, 2747.5810208333332] 2747.5810208333332 100
3090.5756237827286  [2496.3023749999998, 2593.9644930555555,
2747.5810208333332, 2713.5360625000003] 2713.5360625000003 101
2749.5278595015493  [2593.9644930555555, 2747.5810208333332,
2713.5360625000003, 2689.9854652777776] 2689.9854652777776 102
2668.2459257616383  [2747.5810208333332, 2713.5360625000003,
2689.9854652777776, 2792.5772569444443] 2792.5772569444443 103
2812.245953717826  [2713.5360625000003, 2689.9854652777776,
2792.5772569444443, 2774.0704097222224] 2774.0704097222224 104
2657.9760756098212  [2689.9854652777776, 2792.5772569444443,
2774.0704097222224, 2712.1440486111114] 2712.1440486111114 105
2631.9721302352295  [2792.5772569444443, 2774.0704097222224,
2712.1440486111114, 2753.2461805555554] 2753.2461805555554 106
2858.8042842020423  [2774.0704097222224, 2712.1440486111114,
2753.2461805555554, 2849.1954236111114] 2849.1954236111114 107
2961.0516630325947  [2712.1440486111114, 2753.2461805555554,
2849.1954236111114, 3191.7712291666667] 3191.7712291666667 108
3685.8608766379753  [2753.2461805555554, 2849.1954236111114,
3191.7712291666667, 3213.771909722222] 3213.771909722222 109
3339.621643054149  [2849.1954236111114, 3191.7712291666667,
3213.771909722222, 3336.721229166667] 3336.721229166667 110
3430.0410754689424  [3191.7712291666667, 3213.771909722222,
3336.721229166667, 3420.602715277778] 3420.602715277778 111
3390.1576612433832  [3213.771909722222, 3336.721229166667,
```

```
3420.602715277778, 3336.4065069444446] 3336.4065069444446 112
3073.4962555726265  [3336.721229166667, 3420.602715277778,
3336.4065069444446, 3411.066201388889] 3411.066201388889 113
3418.1191372203475  [3420.602715277778, 3336.4065069444446,
3411.066201388889, 3552.4018055555553] 3552.4018055555553 114
3851.588250108057  [3336.4065069444446, 3411.066201388889,
3552.4018055555553, 3849.182027777778] 3849.182027777778 115
4262.294099265629  [3411.066201388889, 3552.4018055555553,
3849.182027777778, 4052.7570694444444] 4052.7570694444444 116
4412.5045793439995  [3552.4018055555553, 3849.182027777778,
4052.7570694444444, 4228.112034722222] 4228.112034722222 117
4453.424506383397  [3849.182027777778, 4052.7570694444444,
4228.112034722222, 4135.742472222222] 4135.742472222222 118
3810.906971733409  [4052.7570694444444, 4228.112034722222,
4135.742472222222, 4207.153243055555] 4207.153243055555 119
4105.943476716401  [4228.112034722222, 4135.742472222222,
4207.153243055555, 4338.1511736111115] 4338.1511736111115 120
4423.197671655926  [4135.742472222222, 4207.153243055555,
4338.1511736111115, 4211.887145833333] 4211.887145833333 121
4081.0962237290255  [4207.153243055555, 4338.1511736111115,
4211.887145833333, 4090.9302777777775] 4090.9302777777775 122
3940.759558340353  [4338.1511736111115, 4211.887145833333,
4090.9302777777775, 4102.02275] 4102.02275 123
4150.825860065823  [4211.887145833333, 4090.9302777777775, 4102.02275,
4000.476423611111] 4000.476423611111 124
3806.1151738908356  [4090.9302777777775, 4102.02275,
4000.476423611111, 3968.6874236111107] 3968.6874236111107 125
3988.4297780857823  [4102.02275, 4000.476423611111,
3968.6874236111107, 4161.1070625] 4161.1070625 126
4573.496531136574  [4000.476423611111, 3968.6874236111107,
4161.1070625, 4229.312861111111] 4229.312861111111 127
4357.3822470266105  [3968.6874236111107, 4161.1070625,
4229.312861111111, 4357.201] 4357.201 128
4515.498890733864  [4161.1070625, 4229.312861111111, 4357.201,
4323.121375000001] 4323.121375000001 129
4266.942103087509  [4229.312861111111, 4357.201, 4323.121375000001,
4337.465625] 4337.465625 130
4208.378401923291  [4357.201, 4323.121375000001, 4337.465625,
4322.8015000000005] 4322.8015000000005 131
4192.864095901293  [4323.121375000001, 4337.465625,
4322.8015000000005, 4505.445798611111] 4505.445798611111 132
4794.127904105307  [4337.465625, 4322.8015000000005,
4505.445798611111, 4584.305944444444] 4584.305944444444 133
4736.463763763839  [4322.8015000000005, 4505.445798611111,
4584.305944444444, 4703.231305555556] 4703.231305555556 134
4919.515434906642  [4505.445798611111, 4584.305944444444,
4703.231305555556, 4836.6599375] 4836.6599375 135
4971.541125076544  [4584.305944444444, 4703.231305555556,
4836.6599375, 4678.04975] 4678.04975 136
```

```
4370.966236141769  [4703.231305555556, 4836.6599375, 4678.04975,
4624.408520833333] 4624.408520833333 137
4396.173659532125  [4836.6599375, 4678.04975, 4624.408520833333,
4447.767423611111] 4447.767423611111 138
4188.058094032236  [4678.04975, 4624.408520833333, 4447.767423611111,
4417.296958333333] 4417.296958333333 139
4358.683843197589  [4624.408520833333, 4447.767423611111,
4417.296958333333, 4594.3478888888885] 4594.3478888888885 140
4966.431289091662  [4447.767423611111, 4417.296958333333,
4594.3478888888885, 4600.271472222222] 4600.271472222222 141
4798.962844845226  [4417.296958333333, 4594.3478888888885,
4600.271472222222, 4453.546888888888] 4453.546888888888 142
4234.03252676022  [4594.3478888888885, 4600.271472222222,
4453.546888888888, 4329.529534722223] 4329.529534722223 143
```

```
4105.669851111961  [4600.271472222222, 4453.546888888888,
4329.529534722223, 4250.614541666667] 4250.614541666667 144
4005.7152862728967  [4453.546888888888, 4329.529534722223,
4250.614541666667, 4212.8381875000005] 4212.8381875000005 145
4152.872529729212  [4329.529534722223, 4250.614541666667,
4212.8381875000005, 4229.4228125] 4229.4228125 146
4421.861438942625  [4250.614541666667, 4212.8381875000005,
4229.4228125, 3899.4992291666663] 3899.4992291666663 147
3533.2279884602553  [4212.8381875000005, 4229.4228125,
3899.4992291666663, 3545.2712430555553] 3545.2712430555553 148
2987.954169534215  [4229.4228125, 3899.4992291666663,
3545.2712430555553, 3495.2340694444442] 3495.2340694444442 149
3394.5817141655707  [3899.4992291666663, 3545.2712430555553,
3495.2340694444442, 3756.6529375000005] 3756.6529375000005 150
4240.261181614286  [3545.2712430555553, 3495.2340694444442,
3756.6529375000005, 3691.706048611111] 3691.706048611111 151
3843.7368094875237  [3495.2340694444442, 3756.6529375000005,
3691.706048611111, 3991.9329374999998] 3991.9329374999998 152
4599.403610080371  [3756.6529375000005, 3691.706048611111,
3991.9329374999998, 3951.8083472222224] 3951.8083472222224 153
3829.396007785381  [3691.706048611111, 3991.9329374999998,
3951.8083472222224, 3933.807166666667] 3933.807166666667 154
3640.2548847248627  [3991.9329374999998, 3951.8083472222224,
3933.807166666667, 3759.2553541666666] 3759.2553541666666 155
3372.808668641079  [3951.8083472222224, 3933.807166666667,
3759.2553541666666, 3621.531145833333] 3621.531145833333 156
3426.0171694463456  [3933.807166666667, 3759.2553541666666,
3621.531145833333, 3752.771909722222] 3752.771909722222 157
3875.3799755217883  [3759.2553541666666, 3621.531145833333,
3752.771909722222, 3690.0590138888892] 3690.0590138888892 158
3830.357195061756  [3621.531145833333, 3752.771909722222,
3690.0590138888892, 3860.085090277778] 3860.085090277778 159
4166.272616058257  [3752.771909722222, 3690.0590138888892,
3860.085090277778, 3914.9755] 3914.9755 160
4072.2653881285237  [3690.0590138888892, 3860.085090277778, 3914.9755,
```

```
4082.772055555555] 4082.772055555555 161
4232.823665038231  [3860.085090277778, 3914.9755, 4082.772055555555,
4186.487548611111] 4186.487548611111 162
4241.042650803309  [3914.9755, 4082.772055555555, 4186.487548611111,
4150.750458333334] 4150.750458333334 163
4080.694414856681  [4082.772055555555, 4186.487548611111,
4150.750458333334, 4290.905805555556] 4290.905805555556 164
4342.124399749725  [4186.487548611111, 4150.750458333334,
4290.905805555556, 4328.2296875] 4328.2296875 165
4377.33498940112  [4150.750458333334, 4290.905805555556, 4328.2296875,
4410.164319444444] 4410.164319444444 166
4471.771304877578  [4290.905805555556, 4328.2296875,
4410.164319444444, 4300.587597222222] 4300.587597222222 167
4171.34754067567  [4328.2296875, 4410.164319444444, 4300.587597222222,
4239.825791666666] 4239.825791666666 168
4113.8802012310925  [4410.164319444444, 4300.587597222222,
4239.825791666666, 4275.225909722222] 4275.225909722222 169
4251.72318457769  [4300.587597222222, 4239.825791666666,
4275.225909722222, 4368.1047708333335] 4368.1047708333335 170
4568.171823286472  [4239.825791666666, 4275.225909722222,
4368.1047708333335, 4366.780847222222] 4366.780847222222 171
4428.305490560044  [4275.225909722222, 4368.1047708333335,
4366.780847222222, 4530.140409722222] 4530.140409722222 172
4822.766692708458  [4368.1047708333335, 4366.780847222222,
4530.140409722222, 4684.224625] 4684.224625 173
4858.285596619558  [4366.780847222222, 4530.140409722222, 4684.224625,
4807.587493055556] 4807.587493055556 174
4916.507120402977  [4530.140409722222, 4684.224625, 4807.587493055556,
4807.469777777778] 4807.469777777778 175
4751.96275411858  [4684.224625, 4807.587493055556, 4807.469777777778,
5232.137243055556] 5232.137243055556 176
5763.309019714363  [4807.587493055556, 4807.469777777778,
5232.137243055556, 5628.263541666666] 5628.263541666666 177
6134.578077128435  [4807.469777777778, 5232.137243055556,
5628.263541666666, 5702.026611111111] 5702.026611111111 178
5793.257896139915  [5232.137243055556, 5628.263541666666,
5702.026611111111, 5610.527729166666] 5610.527729166666 179
5350.153571298761  [5628.263541666666, 5702.026611111111,
5610.527729166666, 5683.467923611111] 5683.467923611111 180
5550.294834214996  [5702.026611111111, 5610.527729166666,
5683.467923611111, 5601.248423611111] 5601.248423611111 181
5248.716800068175  [5610.527729166666, 5683.467923611111,
5601.248423611111, 5425.076215277778] 5425.076215277778 182
5240.6205276307865  [5683.467923611111, 5601.248423611111,
5425.076215277778, 5673.920055555555] 5673.920055555555 183
6184.845499517788  [5601.248423611111, 5425.076215277778,
5673.920055555555, 5850.480736111111] 5850.480736111111 184
6214.891425427803  [5425.076215277778, 5673.920055555555,
5850.480736111111, 6078.082340277778] 6078.082340277778 185
```

```
6419.913631014326  [5673.920055555555, 5850.480736111111,
6078.082340277778, 5926.425152777778] 5926.425152777778 186
5758.113917174168  [5850.480736111111, 6078.082340277778,
5926.425152777778, 5841.58825] 5841.58825 187
5525.973959645191  [6078.082340277778, 5926.425152777778, 5841.58825,
5632.815847222222] 5632.815847222222 188
5084.833423229178  [5926.425152777778, 5841.58825, 5632.815847222222,
5603.770652777778] 5603.770652777778 189
5570.237647536046  [5841.58825, 5632.815847222222, 5603.770652777778,
5865.141673611111] 5865.141673611111 190
6374.08673013177  [5632.815847222222, 5603.770652777778,
5865.141673611111, 5815.4378541666665] 5815.4378541666665 191
6004.675895074523  [5603.770652777778, 5865.141673611111,
5815.4378541666665, 5762.395618055556] 5762.395618055556 192
5678.5715276937  [5865.141673611111, 5815.4378541666665,
5762.395618055556, 5950.888472222223] 5950.888472222223 193
6169.593237827586  [5815.4378541666665, 5762.395618055556,
5950.888472222223, 6125.478881944445] 6125.478881944445 194
6248.472582827596  [5762.395618055556, 5950.888472222223,
6125.478881944445, 6295.324097222222] 6295.324097222222 195
6501.79747145486  [5950.888472222223, 6125.478881944445,
6295.324097222222, 6595.392118055556] 6595.392118055556 196
7115.429476535096  [6125.478881944445, 6295.324097222222,
6595.392118055556, 7032.427291666667] 7032.427291666667 197
7553.878407030361  [6295.324097222222, 6595.392118055556,
7032.427291666667, 7260.6722291666665] 7260.6722291666665 198
7433.912625369664  [6595.392118055556, 7032.427291666667,
7260.6722291666665, 7318.0558125] 7318.0558125 199
7264.767035180797  [7032.427291666667, 7260.6722291666665,
7318.0558125, 7465.924360000001] 7465.924360000001 200
7458.640346426323  [7260.6722291666665, 7318.0558125,
7465.924360000001, 7195.7843125] 7195.7843125 201
6601.58562637387  [7318.0558125, 7465.924360000001, 7195.7843125,
7148.5320625] 7148.5320625 202
6951.801293276872  [7465.924360000001, 7195.7843125, 7148.5320625,
7404.7183958333335] 7404.7183958333335 203
7881.196135001351  [7195.7843125, 7148.5320625, 7404.7183958333335,
7225.421298611111] 7225.421298611111 204
7116.291987512754  [7148.5320625, 7404.7183958333335,
7225.421298611111, 6864.494159722222] 6864.494159722222 205
6384.051370711149  [7404.7183958333335, 7225.421298611111,
6864.494159722222, 6438.928145833333] 6438.928145833333 206
5866.091754897733  [7225.421298611111, 6864.494159722222,
6438.928145833333, 6046.0095972222225] 6046.0095972222225 207
5322.822043799765  [6864.494159722222, 6438.928145833333,
6046.0095972222225, 6498.045208333333] 6498.045208333333 208
7247.937987368932  [6438.928145833333, 6046.0095972222225,
6498.045208333333, 6624.604013888888] 6624.604013888888 209
7330.034945486718  [6046.0095972222225, 6498.045208333333,
```

```
6624.604013888888, 7134.8696875] 7134.8696875 210
8090.530756002121  [6498.045208333333, 6624.604013888888,
7134.8696875, 7606.2153263888895] 7606.2153263888895 211
8298.8275957037  [6624.604013888888, 7134.8696875, 7606.2153263888895,
7775.014708333333] 7775.014708333333 212
7714.562046434106  [7134.8696875, 7606.2153263888895,
7775.014708333333, 7752.3725] 7752.3725 213
7220.532990385738  [7606.2153263888895, 7775.014708333333, 7752.3725,
7885.3930625] 7885.3930625 214
7894.153969624316  [7775.014708333333, 7752.3725, 7885.3930625,
8149.793006944445] 8149.793006944445 215

8276.954416086804  [7752.3725, 7885.3930625, 8149.793006944445,
8192.566888888889] 8192.566888888889 216
8331.261826206815  [7885.3930625, 8149.793006944445,
8192.566888888889, 8224.038] 8224.038 217
8370.585854409303  [8149.793006944445, 8192.566888888889, 8224.038,
8158.045340277778] 8158.045340277778 218
8014.755332471236  [8192.566888888889, 8224.038, 8158.045340277778,
8221.748743055556] 8221.748743055556 219
8152.034790344698  [8224.038, 8158.045340277778, 8221.748743055556,
8639.591048611112] 8639.591048611112 220
9236.281497472404  [8158.045340277778, 8221.748743055556,
8639.591048611112, 9295.825416666667] 9295.825416666667 221
10403.673898876723  [8221.748743055556, 8639.591048611112,
9295.825416666667, 9696.481555555556] 9696.481555555556 222
10325.36365546818  [8639.591048611112, 9295.825416666667,
9696.481555555556, 9977.966125] 9977.966125 223
10232.124058537678  [9295.825416666667, 9696.481555555556,
9977.966125, 10541.490055555556] 10541.490055555556 224
10957.524136302296  [9696.481555555556, 9977.966125,
10541.490055555556, 9875.352854166666] 9875.352854166666 225
8537.401963624347  [9977.966125, 10541.490055555556,
9875.352854166666, 10406.63023611111] 10406.63023611111 226
10802.10469215122  [10541.490055555556, 9875.352854166666,
10406.63023611111, 10956.278756944444] 10956.278756944444 227
11856.290372004845  [9875.352854166666, 10406.63023611111,
10956.278756944444, 11415.233701388888] 11415.233701388888 228
12166.224767726177  [10406.63023611111, 10956.278756944444,
11415.233701388888, 11442.076930555555] 11442.076930555555 229
11558.703634425512  [10956.278756944444, 11415.233701388888,
11442.076930555555, 11811.547597222221] 11811.547597222221 230
12314.993850384064  [11415.233701388888, 11442.076930555555,
11811.547597222221, 13179.976791666666] 13179.976791666666 231
14477.208137713795  [11442.076930555555, 11811.547597222221,
13179.976791666666, 16110.155229166667] 16110.155229166667 232
20412.57486517842  [11811.547597222221, 13179.976791666666,
16110.155229166667, 15934.581388888888] 15934.581388888888 233
15922.529061822326  [13179.976791666666, 16110.155229166667,
15934.581388888888, 15195.640243055555] 15195.640243055555 234
```

```
13338.216637072186  [16110.155229166667, 15934.581388888888,
15195.640243055555, 15179.819444444445] 15179.819444444445 235
13686.63157814784  [15934.581388888888, 15195.640243055555,
15179.819444444445, 16796.45004861111] 16796.45004861111 236
18071.099013139523  [15195.640243055555, 15179.819444444445,
16796.45004861111, 17274.807381944443] 17274.807381944443 237
18417.535025510908  [15179.819444444445, 16796.45004861111,
17274.807381944443, 16978.164395833333] 16978.164395833333 238
17933.933327259714  [16796.45004861111, 17274.807381944443,
16978.164395833333, 16915.718305555554] 16915.718305555554 239
16351.345434289995  [17274.807381944443, 16978.164395833333,
16915.718305555554, 17651.504868055556] 17651.504868055556 240
17711.965150369437  [16978.164395833333, 16915.718305555554,
17651.504868055556, 18805.654840277777] 18805.654840277777 241
20137.582159684614  [16915.718305555554, 17651.504868055556,
18805.654840277777, 19419.864097222224] 19419.864097222224 242
21071.72464768752  [17651.504868055556, 18805.654840277777,
19419.864097222224, 18908.39645138889] 18908.39645138889 243
18171.765681227644  [18805.654840277777, 19419.864097222224,
18908.39645138889, 17985.722729166668] 17985.722729166668 244
15783.889106055409  [19419.864097222224, 18908.39645138889,
17985.722729166668, 16807.2515625] 16807.2515625 245
14267.7767637228  [18908.39645138889, 17985.722729166668,
16807.2515625, 15955.282680555554] 15955.282680555554 246
14671.348724757578  [17985.722729166668, 16807.2515625,
15955.282680555554, 13873.203222222222] 13873.203222222222 247
11721.582468045413  [16807.2515625, 15955.282680555554,
13873.203222222222, 15092.708374999998] 15092.708374999998 248
17716.74576287361  [15955.282680555554, 13873.203222222222,
15092.708374999998, 14061.63598611111] 14061.63598611111 249
13714.028455406347  [13873.203222222222, 15092.708374999998,
14061.63598611111, 14278.775576388887] 14278.775576388887 250
15019.811852249017  [15092.708374999998, 14061.63598611111,
14278.775576388887, 15673.237041666667] 15673.237041666667 251
17979.538187468104  [14061.63598611111, 14278.775576388887,
15673.237041666667, 15445.97476388889] 15445.97476388889 252
15218.47338984324  [14278.775576388887, 15673.237041666667,
15445.97476388889, 14235.0436875] 14235.0436875 253
11417.946328841164  [15673.237041666667, 15445.97476388889,
14235.0436875, 14447.896229166667] 14447.896229166667 254
14884.42462593866  [15445.97476388889, 14235.0436875,
14447.896229166667, 13377.130444444445] 13377.130444444445 255
11110.66774167019  [14235.0436875, 14447.896229166667,
13377.130444444445, 13471.729645833335] 13471.729645833335 256
13632.806114735446  [14447.896229166667, 13377.130444444445,
13471.729645833335, 13426.872152777776] 13426.872152777776 257
14560.364019462733  [13377.130444444445, 13471.729645833335,
13426.872152777776, 14191.70517361111] 14191.70517361111 258
15662.099877078163  [13471.729645833335, 13426.872152777776,
```

```
14191.70517361111, 15033.472138888888] 15033.472138888888 259
16173.182337933158  [13426.872152777776, 14191.70517361111,
15033.472138888888, 14825.769069444445] 14825.769069444445 260
14834.200073162145  [14191.70517361111, 15033.472138888888,
14825.769069444445, 16150.067569444445] 16150.067569444445 261
17234.518239161298  [15033.472138888888, 14825.769069444445,
16150.067569444445, 16691.451652777778] 16691.451652777778 262
17229.68492186877  [14825.769069444445, 16150.067569444445,
16691.451652777778, 16440.150026857653] 16440.150026857653 263
```

```
resultsall =
pd.concat([pd.DataFrame(predictions,index=test.index,columns=['predict
ions']),test],axis=1)
resultsall.head(3)
```

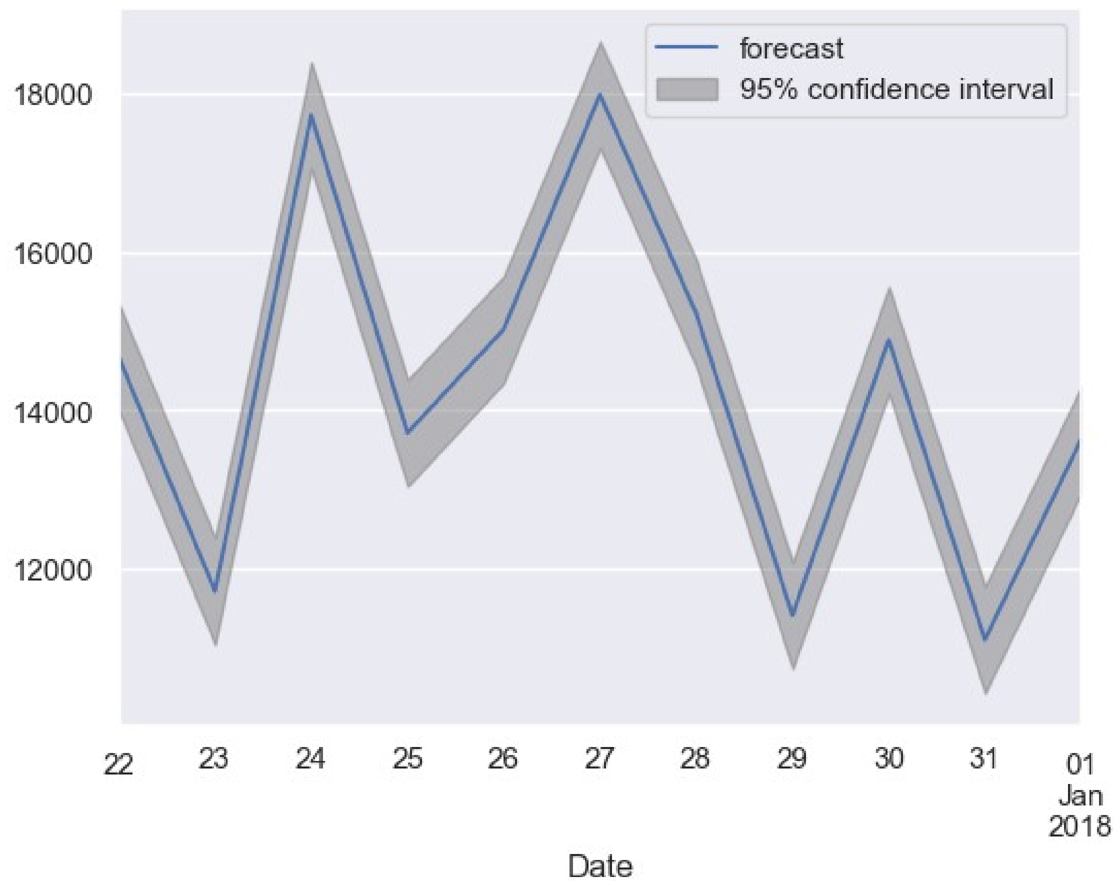
| | predictions | Close |
|---|---|---|
| Date | | |
| 2017-04-19 | 1233.531198 | 1203.656632 |
| 2017-04-20 | 1196.449180 | 1229.868500 |
| 2017-04-21 | 1264.537203 | 1248.670910 |

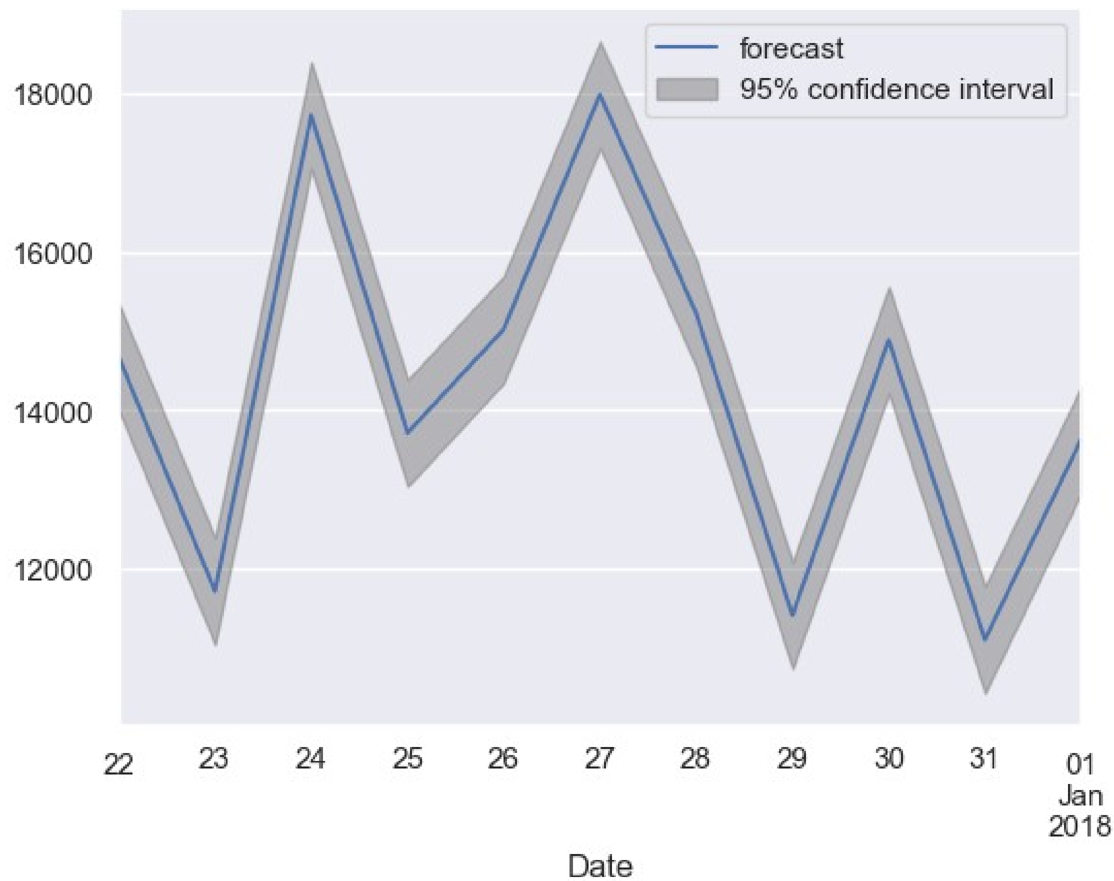
```
error=math.sqrt(mean_squared_error(test,predictions))
error
```

```
695.2002797477246
```

```
from statsmodels.tsa.arima.model import ARIMA
mod = ARIMA(arima2015hour, order=(4,4,0))
result = mod.fit()
```

```
from statsmodels.graphics.tsaplots import plot_predict
plot_predict(result, start=1080,end=1090)
```

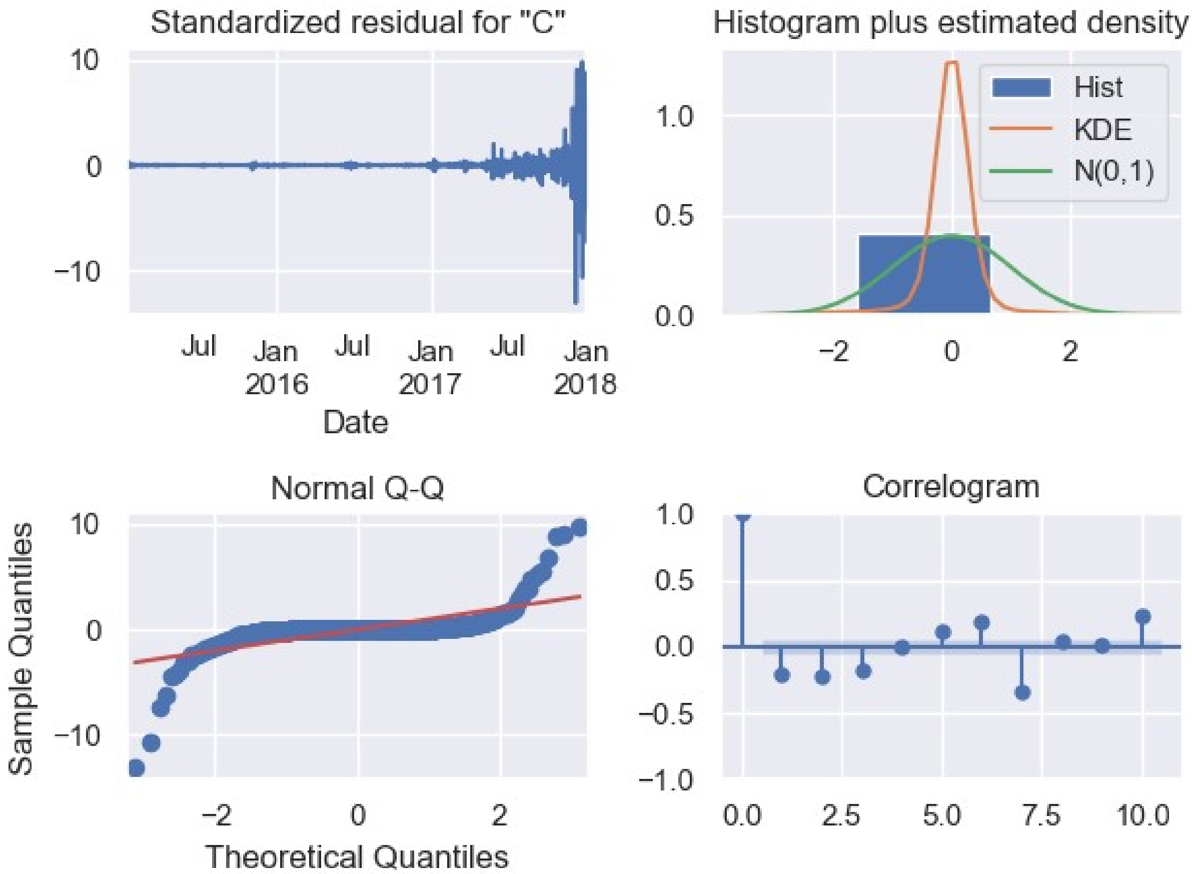

forecast
95% confidence interval
18000
16000
14000
12000
22
23
24
25
26
27
28
29
30
31
01
Jan
2018
Date




```
print(result.summary())
# result.plot_predict()
```

```
                              SARIMAX Results
==========================================================================
========
Dep. Variable:                  Close   No. Observations:
1097
Model:                 ARIMA(4, 4, 0)   Log Likelihood               -
7930.705
Date:                Wed, 10 May 2023   AIC
15871.409
Time:                        13:49:53   BIC
15896.393
Sample:                    01-07-2015   HQIC
15880.863
                         - 01-07-2018

Covariance Type:                  opg

==========================================================================
```

```
========
                 coef    std err          z      P>|z|      [0.025
0.975]
----------------------------------------------------------------------
--------
ar.L1         -1.5567      0.007   -229.176      0.000      -1.570
-1.543
ar.L2         -1.5253      0.011   -135.312      0.000      -1.547
-1.503
ar.L3         -1.0545      0.012    -84.428      0.000      -1.079
-1.030
ar.L4         -0.4237      0.007    -57.649      0.000      -0.438
-0.409
sigma2      1.177e+05    927.211    126.976      0.000    1.16e+05
1.2e+05
======================================================================
=============
Ljung-Box (L1) (Q):                  48.99   Jarque-Bera (JB):
204546.54
Prob(Q):                              0.00   Prob(JB):
0.00
Heteroskedasticity (H):             978.81   Skew:
-0.91
Prob(H) (two-sided):                  0.00   Kurtosis:
69.99
======================================================================
=============

Warnings:
[1] Covariance matrix calculated using the outer product of gradients
(complex-step).

# result.plot_predict(start=1060, end=1090)
result.plot_diagnostics()
plt.tight_layout()
plt.show()
```


Standardized residual for "C"
10
0
−10
Jul
Jan 2016
Jul
Jan 2017
Jul
Jan 2018
Date
Histogram plus estimated density
Hist
KDE
N(0,1)
1.0
0.5
0.0
−2
0
2
Normal Q-Q
Sample Quantiles
10
0
−10
−2
0
2
Theoretical Quantiles
Correlogram
1.0
0.5
0.0
−0.5
−1.0
0.0
2.5
5.0
7.5
10.0