CIND820 Initial Coding

November 17, 2020

[308]:
```
import os
import warnings
warnings.filterwarnings('ignore')
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
plt.style.use('fivethirtyeight')
from pylab import rcParams
rcParams['figure.figsize'] = 10, 6
from datetime import datetime
from statsmodels.tsa.stattools import adfuller
from statsmodels.tsa.seasonal import seasonal_decompose
from statsmodels.tsa.arima_model import ARIMA
from sklearn.metrics import mean_squared_error, mean_absolute_error
import math
from statsmodels.tsa.stattools import acf, pacf
```

[255]:
```
pip install pmdarima
```

```
Requirement already satisfied: pmdarima in /opt/conda/lib/python3.7/site-
packages (1.7.1)
Requirement already satisfied: urllib3 in /opt/conda/lib/python3.7/site-packages
(from pmdarima) (1.25.9)
Requirement already satisfied: joblib>=0.11 in /opt/conda/lib/python3.7/site-
packages (from pmdarima) (0.15.1)
Requirement already satisfied: statsmodels<0.12,>=0.11 in
/opt/conda/lib/python3.7/site-packages (from pmdarima) (0.11.1)
Requirement already satisfied: Cython<0.29.18,>=0.29 in
/opt/conda/lib/python3.7/site-packages (from pmdarima) (0.29.17)
Requirement already satisfied: setuptools<50.0.0 in
/opt/conda/lib/python3.7/site-packages (from pmdarima) (46.1.3.post20200325)
Requirement already satisfied: numpy>=1.17.3 in /opt/conda/lib/python3.7/site-
packages (from pmdarima) (1.18.4)
Requirement already satisfied: pandas>=0.19 in /opt/conda/lib/python3.7/site-
packages (from pmdarima) (1.0.3)
Requirement already satisfied: scikit-learn>=0.22 in
/opt/conda/lib/python3.7/site-packages (from pmdarima) (0.22.2.post1)
Requirement already satisfied: scipy>=1.3.2 in /opt/conda/lib/python3.7/site-
```

```
packages (from pmdarima) (1.4.1)
Requirement already satisfied: patsy>=0.5 in /opt/conda/lib/python3.7/site-
packages (from statsmodels<0.12,>=0.11->pmdarima) (0.5.1)
Requirement already satisfied: python-dateutil>=2.6.1 in
/opt/conda/lib/python3.7/site-packages (from pandas>=0.19->pmdarima) (2.8.1)
Requirement already satisfied: pytz>=2017.2 in /opt/conda/lib/python3.7/site-
packages (from pandas>=0.19->pmdarima) (2020.1)
Requirement already satisfied: six in /opt/conda/lib/python3.7/site-packages
(from patsy>=0.5->statsmodels<0.12,>=0.11->pmdarima) (1.14.0)
Note: you may need to restart the kernel to use updated packages.
```

[256]:
```
from pmdarima.arima import auto_arima
```

[257]:
```
#import NASDAQ data
df=pd.read_csv("IXIC_v1.csv", sep=",")
```

[258]:
```
#understand data format and clean up data
from datetime import datetime
con=df['Date']
df['Date']=pd.to_datetime(df['Date'])
df.set_index('Date', inplace=True)
#check datatype of index
df.index
```

[258]:
```
DatetimeIndex(['2010-01-04', '2010-01-05', '2010-01-06', '2010-01-07',
               '2010-01-08', '2010-01-11', '2010-01-12', '2010-01-13',
               '2010-01-14', '2010-01-15',
               …
               '2020-09-17', '2020-09-18', '2020-09-21', '2020-09-22',
               '2020-09-23', '2020-09-24', '2020-09-25', '2020-09-28',
               '2020-09-29', '2020-09-30'],
              dtype='datetime64[ns]', name='Date', length=2705, freq=None)
```

[259]:
```
df['year'] = df.index.year
df['month'] = df.index.month
df['day'] = df.index.day
```

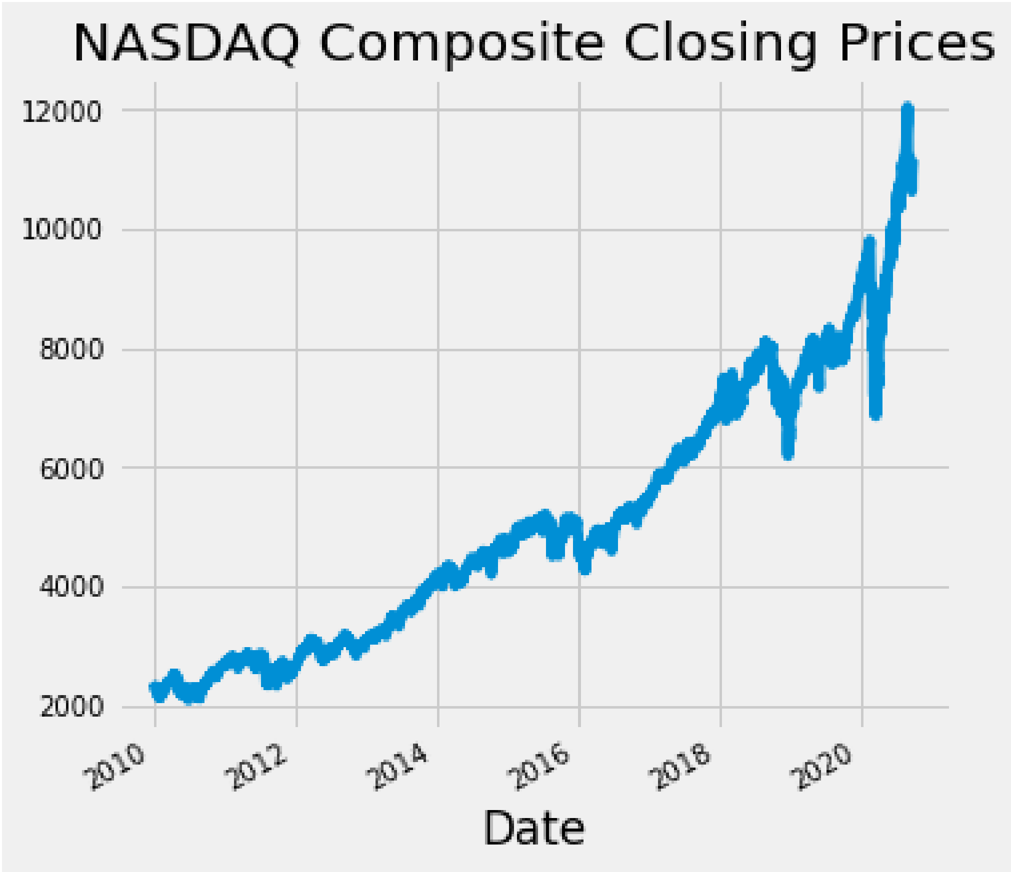
[260]:
```
df.head()
```

[260]:

Date	Close	year	month	day
2010-01-04	2308.419922	2010	1	4
2010-01-05	2308.709961	2010	1	5
2010-01-06	2301.090088	2010	1	6
2010-01-07	2300.050049	2010	1	7
2010-01-08	2317.169922	2010	1	8

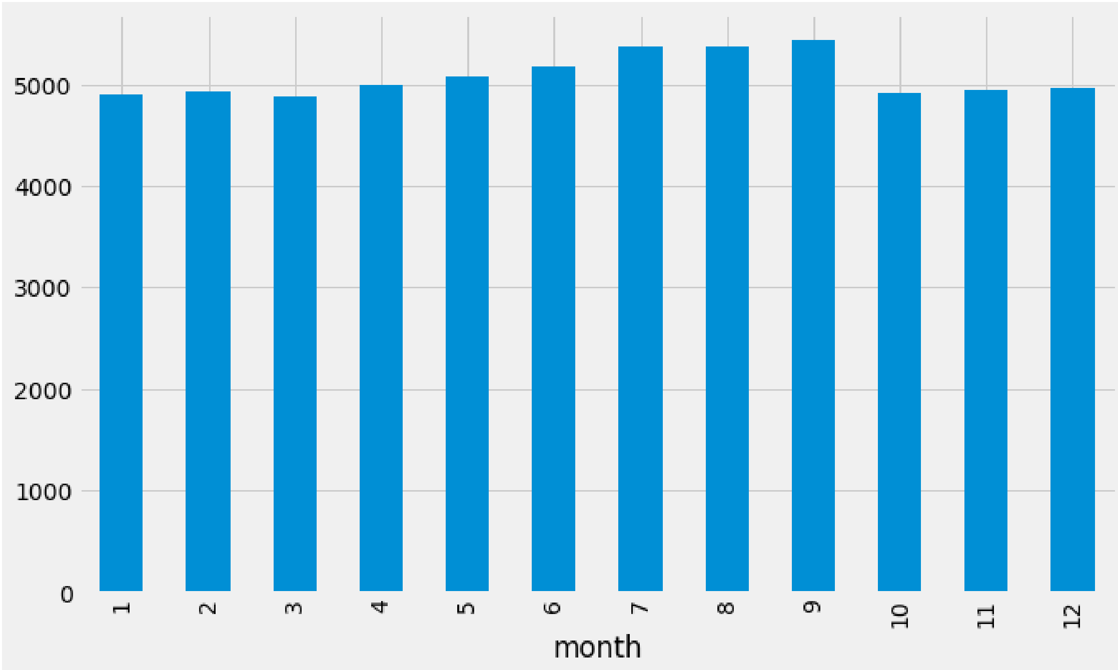
[261]:
```python
#plot NASDAQ trend
temp=df.groupby(['Date'])['Close'].mean()
temp.plot(figsize=(5,5), title= 'NASDAQ Composite Closing Prices', fontsize=10)
```

[261]: <matplotlib.axes._subplots.AxesSubplot at 0x7f9e3475b890>



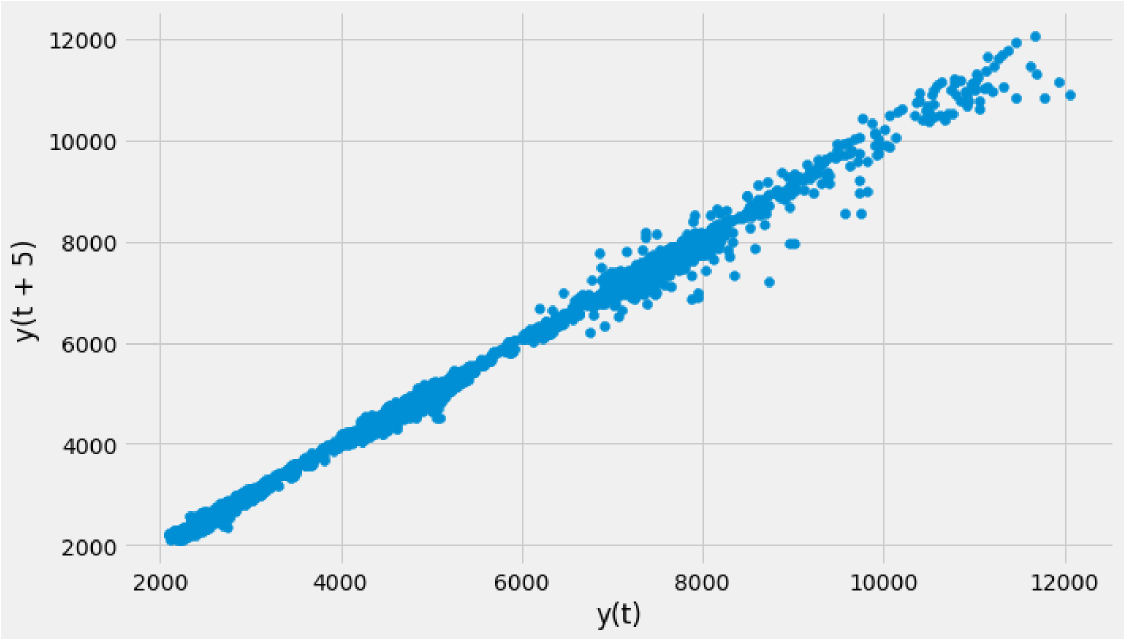
[262]:
```python
df.groupby('month')['Close'].mean().plot.bar()
#on average, september has the highest average price compares to the other␣
↪months.
```

[262]: <matplotlib.axes._subplots.AxesSubplot at 0x7f9e3531ed90>



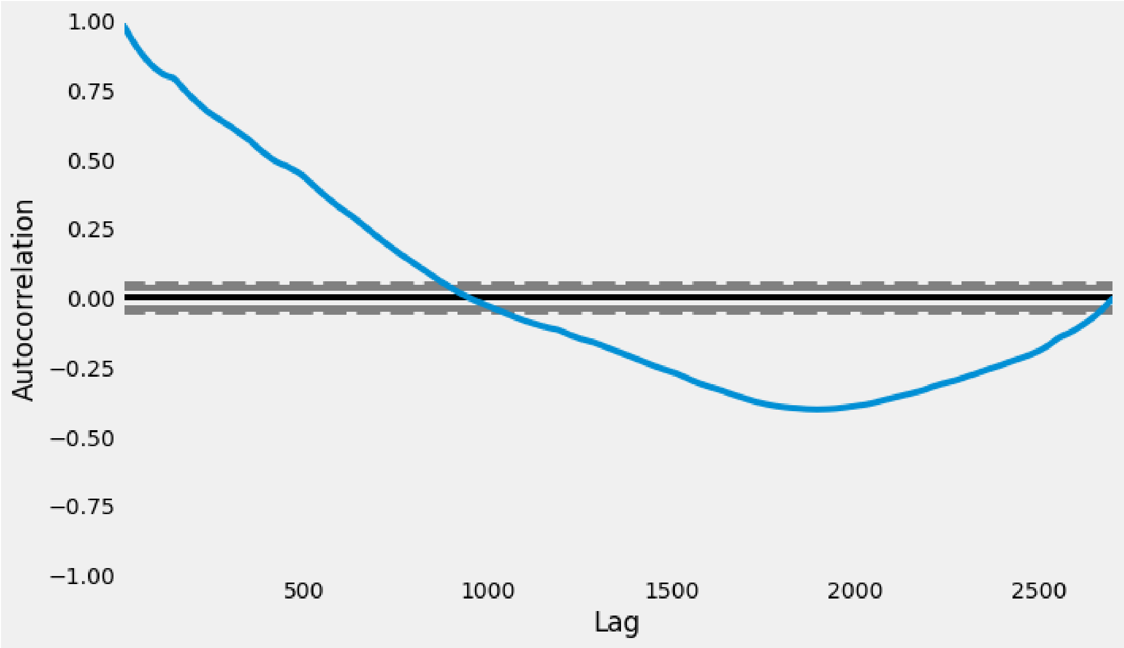
[263]:
```
#lag plot
from pandas.plotting import lag_plot
lag_plot(df['Close'],lag=5)

#Graph shows a linear pattern. Implies data points are non random and suggests␣
↪that an autoregressive model might be appropriate.
```

[263]: <matplotlib.axes._subplots.AxesSubplot at 0x7f9e3bf1fa50>



[264]:
```
from pandas.plotting import autocorrelation_plot
autocorrelation_plot(df['Close'])
#there is high level of correlation
```

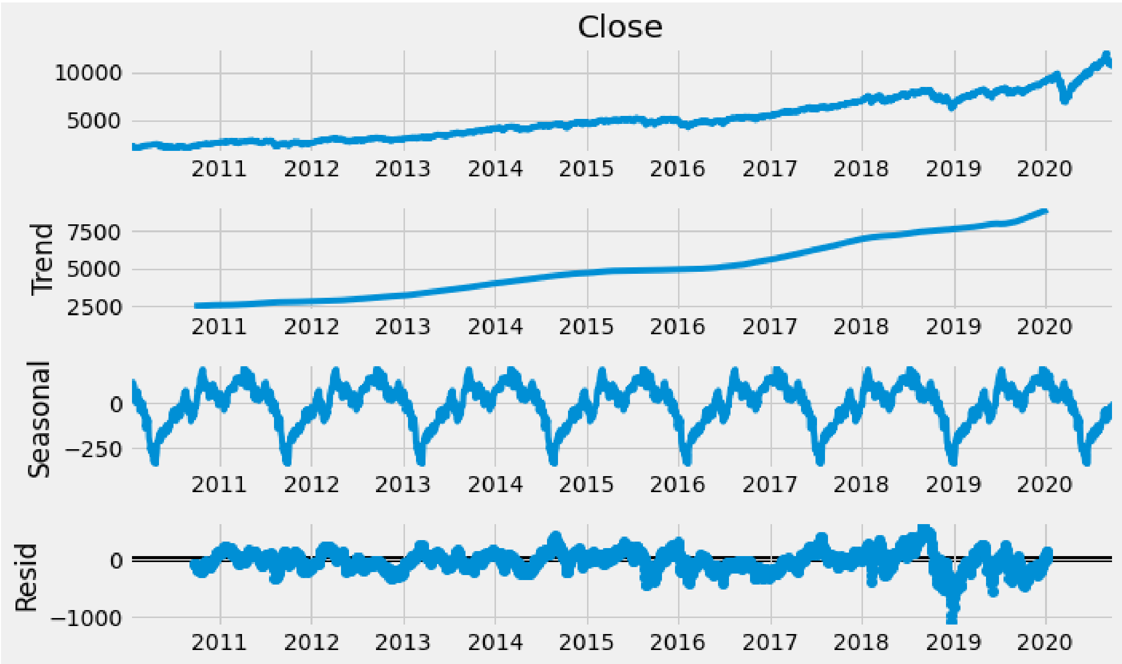
[264]: <matplotlib.axes._subplots.AxesSubplot at 0x7f9e34e2fe90>



[265]:
```
#data is not stationary based on high p value
from statsmodels.tsa.stattools import adfuller
result = adfuller(df.Close.dropna())
print(f"ADF Statstic: {result[0]}")
print(f"p-value:{result[1]}")
```

```
ADF Statstic: 1.4430465972942679
p-value:0.9973011850493003
```

[266]:
```
#decompose data
import statsmodels.api as sm
res = sm.tsa.seasonal_decompose(df['Close'],model= 'addictive',period = 365)
resplot = res.plot()
#data shows upward trend and presents seasonlity
```



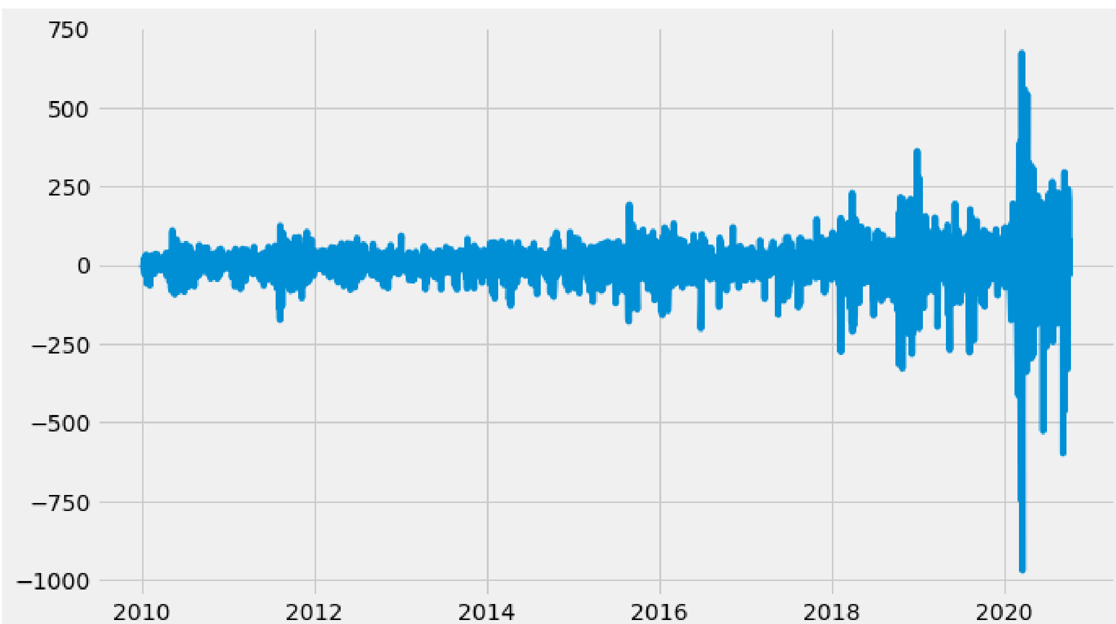
[267]:
```
def plot_df(df,x,y,title= "", xlabel = "Date", ylabel='Value',dpi=50):
    plt.plot(x,y)
    plt.show()
```

[268]:
```
#apply log transformation to stablize data
plt.plot(df.apply(np.log)['Close'])
```

[268]:
```
[<matplotlib.lines.Line2D at 0x7f9e3acc9910>]
```

[269]:
```
#To covert data into stationary dataset, first differencing has to be applied.␣
↪With first differencing, empty field needs to be filled as 0.
plt.plot(df['Close'].diff(1).fillna(0))
```

[269]: [<matplotlib.lines.Line2D at 0x7f9e3ac4af90>]



[270]:
```
#confirm stationarity
from statsmodels.tsa.stattools import adfuller
result = adfuller(np.log(df['Close']).diff(1).fillna(0))
print(f"ADF Statstic: {result[0]}")
print(f"p-value:{result[1]}")
```

```
ADF Statstic: -11.770291597674845
p-value:1.0940406662618215e-21
```

[271]:
```
df_st= df.diff(1).fillna(0)
```

[272]:
```
#understand the structure of the stationary dataset
df_st.head()
```

[272]:

	Close	year	month	day
Date				
2010-01-04	0.000000	0.0	0.0	0.0
2010-01-05	0.290039	0.0	0.0	1.0
2010-01-06	-7.619873	0.0	0.0	1.0
2010-01-07	-1.040039	0.0	0.0	1.0
2010-01-08	17.119873	0.0	0.0	1.0

[275]:
```
#With transformation to maintain stationarity, we need to be model back to the␣
↪original dataset in order to predict stock price.
df_revert=df_st.copy()
df_revert=df_revert.cumsum()
```

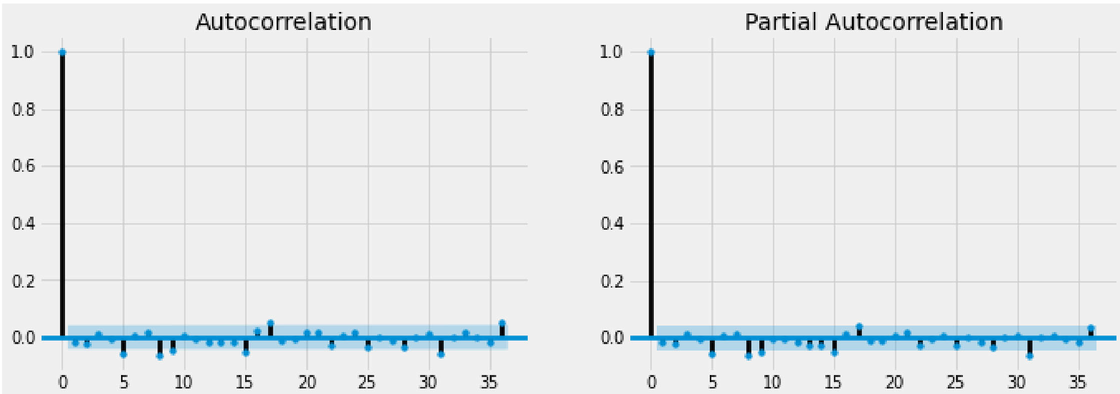
[324]:
```
df_revert.iloc[0,:]=df.iloc[0,:]
df_revert = df_revert.cumsum()
```

[325]:
```
#define data
class TimeSeriesData():
    def __init__(self, df):
        self.data = df
        self.stationary = self.stationarize(df)
        self.revert = self.revert(self.stationary, self.data)

    def revert(self, st, org):
        x = st.copy()
        x.iloc[0,:] = org.iloc[0,:]
        return x.cumsum()

    def stationarize(self, data):
        return data.diff(1).fillna(0)
```

[283]:
```
#split dataset
x_train = TimeSeriesData(df[:int((len(df)*0.8))])
```

```
x_test = TimeSeriesData(df[int((len(df)*0.8)):])
```

[284]:
```
#plot ACF and PACF for the stationary dataset
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
fig, axes = plt.subplots(1,2,figsize=(15,5), dpi= 50)
plot_acf(x_train.stationary['Close'].values.tolist(), lags=36, ax=axes[0]);
plot_pacf(x_train.stationary['Close'].values.tolist(), lags=36, ax=axes[1]);
```



[285]:
```
#using auto_arima to aquire p and q value with min AIC.
from pmdarima import auto_arima
model = auto_arima(x_train.data['Close'], trace=True, error_action='ignore',
    suppress_warnings=True)
model.fit(x_train.data['Close'])
```

```
Performing stepwise search to minimize aic
 ARIMA(2,1,2)(0,0,0)[0] intercept   : AIC=22406.770, Time=1.32 sec
 ARIMA(0,1,0)(0,0,0)[0] intercept   : AIC=22400.729, Time=0.10 sec
 ARIMA(1,1,0)(0,0,0)[0] intercept   : AIC=22402.090, Time=0.15 sec
 ARIMA(0,1,1)(0,0,0)[0] intercept   : AIC=22402.065, Time=0.14 sec
 ARIMA(0,1,0)(0,0,0)[0]             : AIC=22406.532, Time=0.04 sec
 ARIMA(1,1,1)(0,0,0)[0] intercept   : AIC=22393.839, Time=1.55 sec
 ARIMA(2,1,1)(0,0,0)[0] intercept   : AIC=22404.619, Time=0.71 sec
 ARIMA(1,1,2)(0,0,0)[0] intercept   : AIC=22404.702, Time=1.86 sec
 ARIMA(0,1,2)(0,0,0)[0] intercept   : AIC=22402.822, Time=0.84 sec
 ARIMA(2,1,0)(0,0,0)[0] intercept   : AIC=22402.801, Time=0.23 sec
 ARIMA(1,1,1)(0,0,0)[0]             : AIC=22406.446, Time=0.31 sec

Best model:  ARIMA(1,1,1)(0,0,0)[0] intercept
Total fit time: 7.278 seconds
```

[285]:
```
ARIMA(maxiter=50, method='lbfgs', order=(1, 1, 1), out_of_sample_size=0,
      scoring='mse', scoring_args={}, seasonal_order=(0, 0, 0, 0),
      start_params=None, suppress_warnings=True, trend=None,
```

```
    with_intercept=True)
```

[286]:
```
model_arima = ARIMA(x_train.data['Close'].values, order=(1,1,1))
```

[287]:
```
result_arima = model_arima.fit(disp=-1)
```

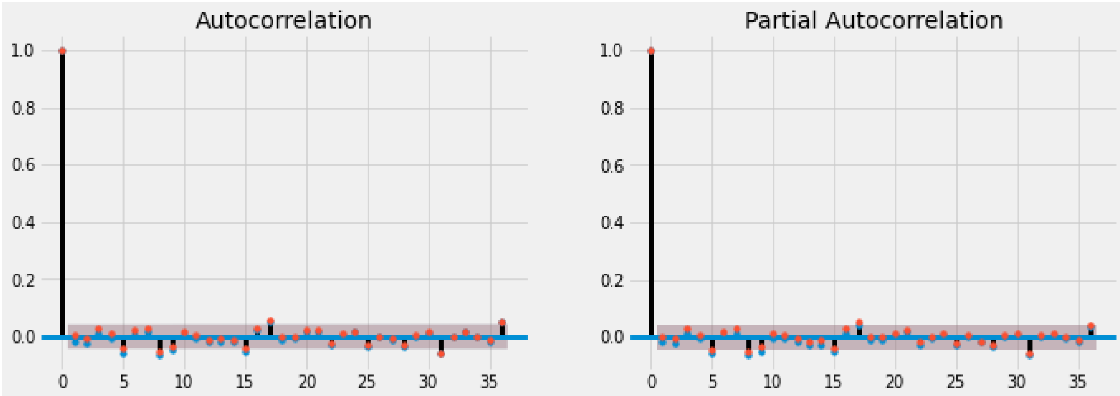
[288]:
```
print(result_arima.summary())
```

```
                              ARIMA Model Results
==============================================================================
Dep. Variable:                    D.y   No. Observations:                 2163
Model:                 ARIMA(1, 1, 1)   Log Likelihood              -11192.925
Method:                       css-mle   S.D. of innovations             42.767
Date:                Tue, 17 Nov 2020   AIC                          22393.851
Time:                        01:05:02   BIC                          22416.568
Sample:                             1   HQIC                         22402.159

==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
const          2.5691      0.570      4.506      0.000       1.452       3.687
ar.L1.D.y      0.9330      0.027     34.756      0.000       0.880       0.986
ma.L1.D.y     -0.9587      0.021    -45.615      0.000      -1.000      -0.917
                                    Roots
=============================================================================
                  Real          Imaginary           Modulus         Frequency
-----------------------------------------------------------------------------
AR.1            1.0718           +0.0000j            1.0718            0.0000
MA.1            1.0431           +0.0000j            1.0431            0.0000
-----------------------------------------------------------------------------
```

[289]:
```
#understand residual
residuals = pd.DataFrame(result_arima.resid)
```

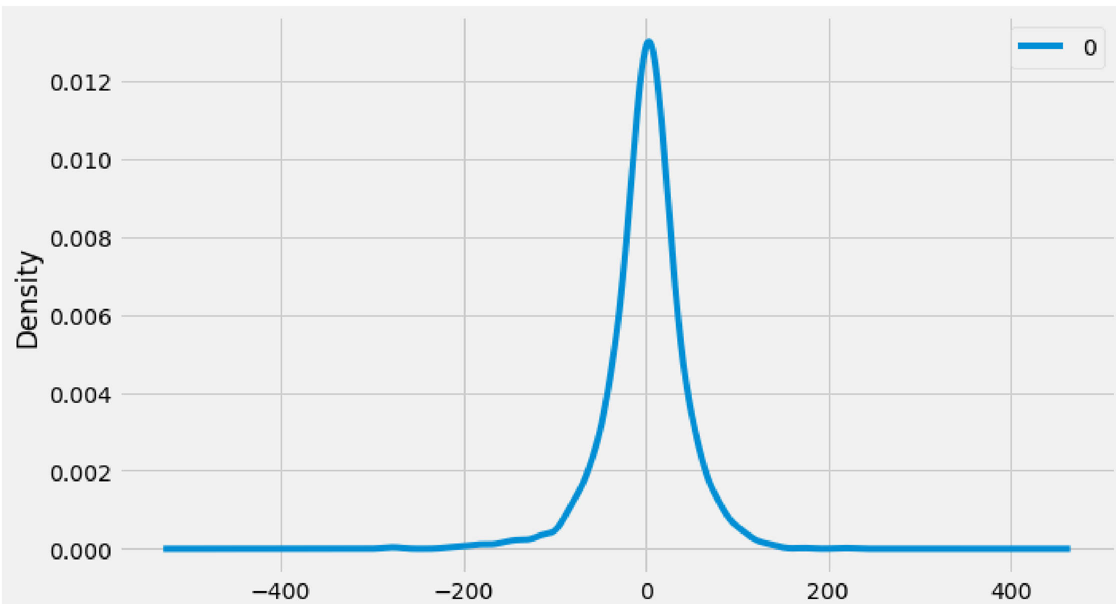
[290]:
```
plot_acf(residuals, lags=36, ax=axes[0])
plot_pacf(residuals, lags=36, ax=axes[1])
```

[290]:

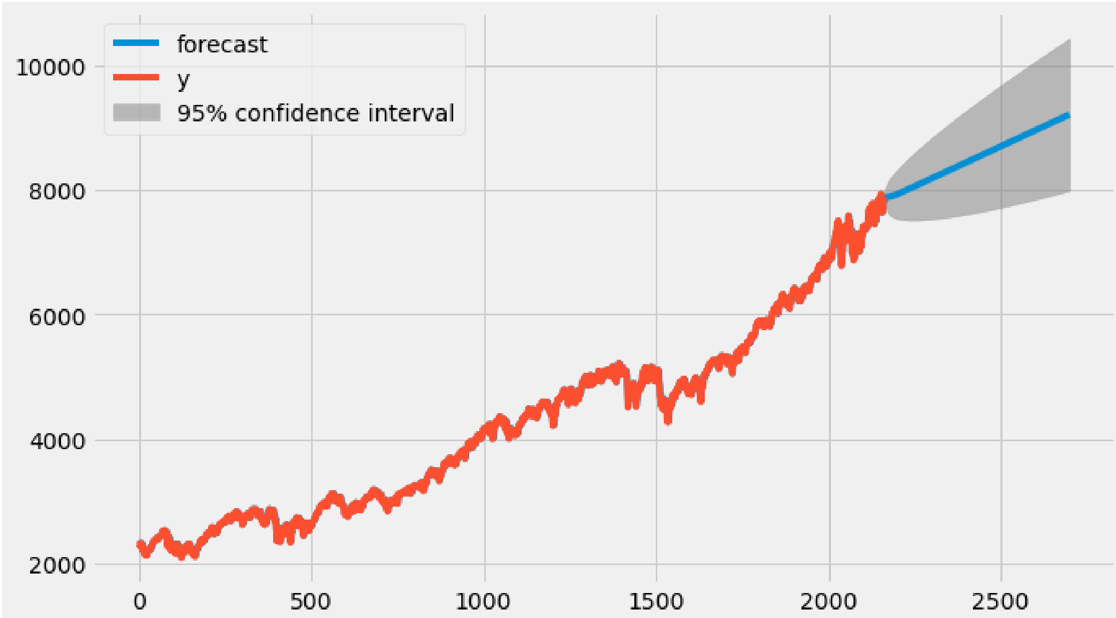


[291]:
```
residuals.plot(kind='kde')
```

[291]: <matplotlib.axes._subplots.AxesSubplot at 0x7f9e3aac3290>



[292]:
```
result_arima.plot_predict(1,2700);
```



[301]:
```
prediction = result_arima.predict(len(df)-200,len(df)-1)
```

[302]:
```
from statsmodels.tools.eval_measures import rmse
#RMSE for ARIMA Model
err_ARIMA = rmse(x_test.data['Close'], np.append(x_train.data.iloc[-1,:
↪]['Close'], prediction).cumsum()[1:])
print('RMSE with ARIMA', err_ARIMA)
```

```
RMSE with ARIMA 889.678717079625
```

[ ]:
```

```

[ ]:
```
#same analysis for TSX price
```

[305]:
```
#upload TSX price
df=pd.read_csv("GSPTSE_v1.csv", sep=",")
```

[309]:
```
#understand the data and covert date format
from datetime import datetime
con=df['Date']
df['Date']=pd.to_datetime(df['Date'])
df.set_index('Date', inplace=True)
#check datatype of index
df.index
```

[309]:
```
DatetimeIndex(['2010-01-04', '2010-01-05', '2010-01-06', '2010-01-07',
               '2010-01-08', '2010-01-11', '2010-01-12', '2010-01-13',
```

```
              '2010-01-14', '2010-01-15',
              …
              '2020-09-17', '2020-09-18', '2020-09-21', '2020-09-22',
              '2020-09-23', '2020-09-24', '2020-09-25', '2020-09-28',
              '2020-09-29', '2020-09-30'],
             dtype='datetime64[ns]', name='Date', length=2697, freq=None)
```

[310]:
```
df['year'] = df.index.year
df['month'] = df.index.month
df['day'] = df.index.day
```

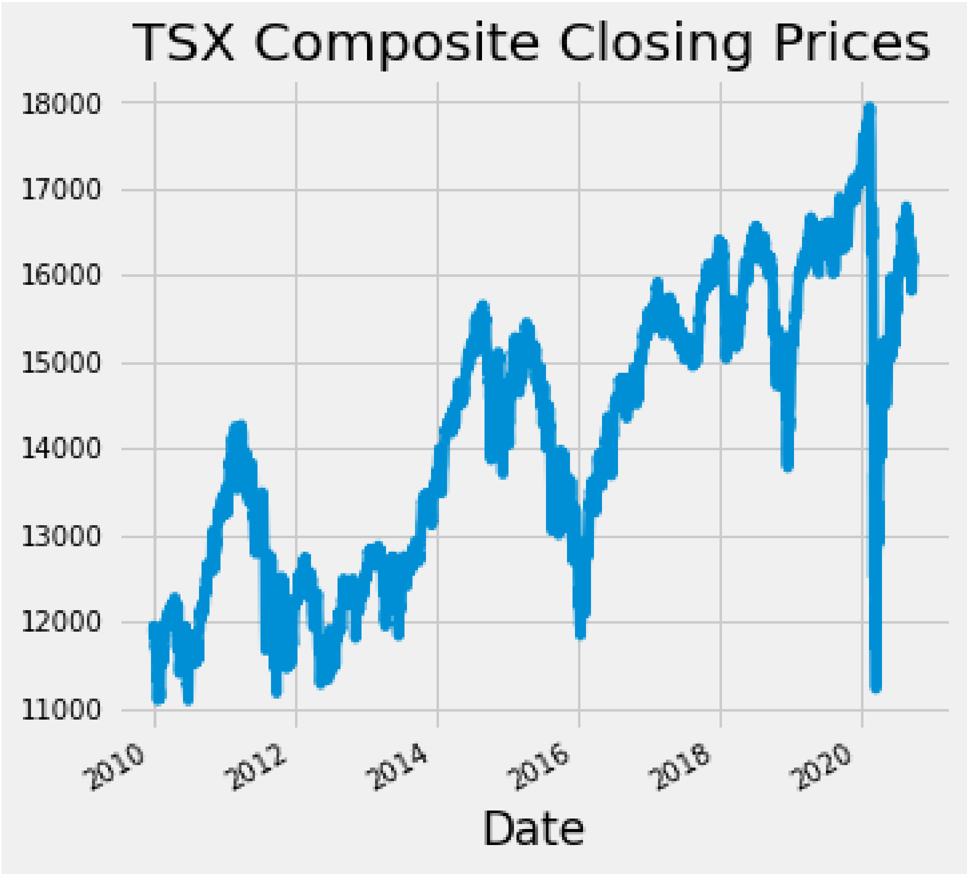
[311]:
```
df.head()
```

[311]:

Date	Close	year	month	day
2010-01-04	11866.90039	2010	1	4
2010-01-05	11888.09961	2010	1	5
2010-01-06	11944.50000	2010	1	6
2010-01-07	11887.50000	2010	1	7
2010-01-08	11953.79981	2010	1	8

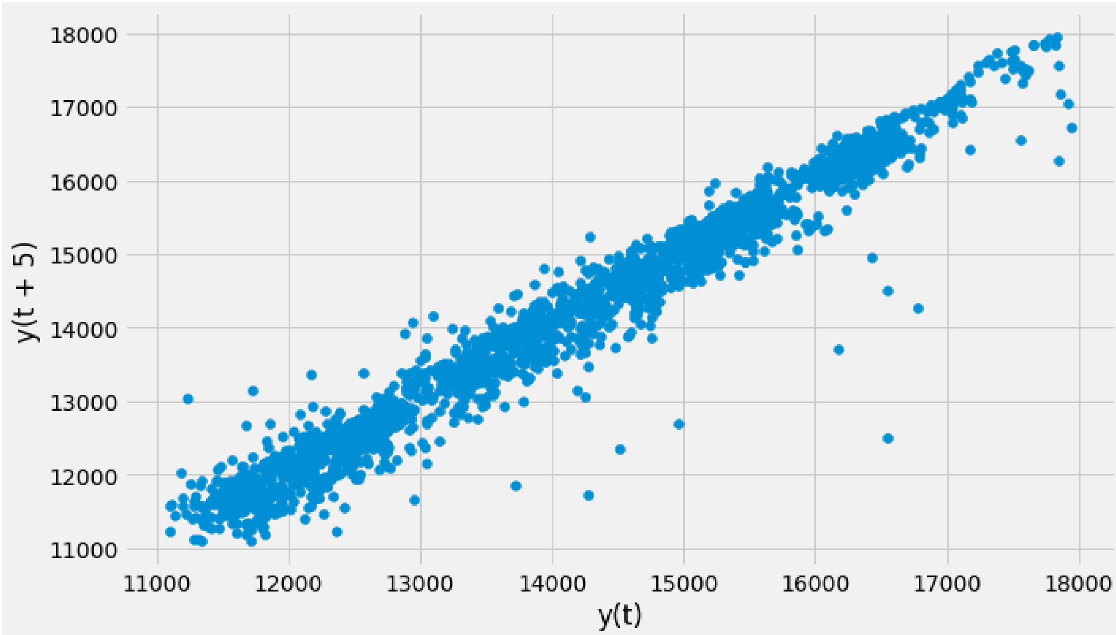
[312]:
```
#plot TSX trend
temp=df.groupby(['Date'])['Close'].mean()
temp.plot(figsize=(5,5), title= 'TSX Composite Closing Prices', fontsize=10)
```

[312]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f9e5a0896d0>
```



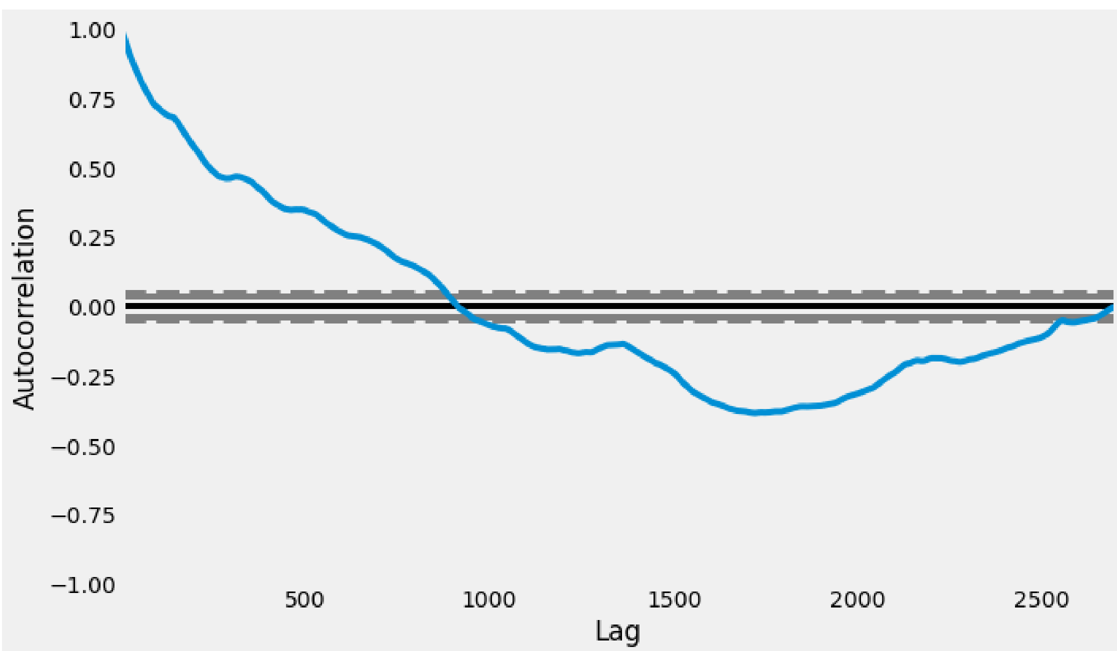
[307]:
```
#lag plot
from pandas.plotting import lag_plot
lag_plot(df['Close'],lag=5)
#a linear plot also indicates non random dataset
```

[307]: <matplotlib.axes._subplots.AxesSubplot at 0x7f9e34de1410>



[313]:
```
from pandas.plotting import autocorrelation_plot
autocorrelation_plot(df['Close'])
#autocorrelation suggests arima might be a good model
```

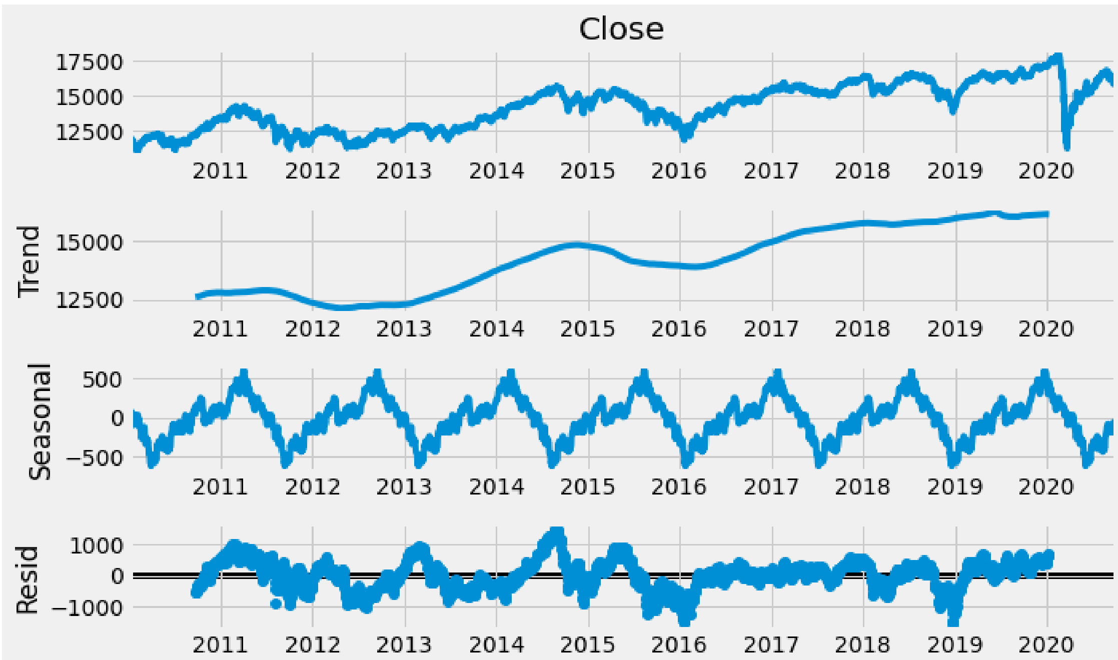
[313]: <matplotlib.axes._subplots.AxesSubplot at 0x7f9e5a0b2210>



[314]:
```
#original data shows as not stationary due to high p level
from statsmodels.tsa.stattools import adfuller
result = adfuller(df.Close.dropna())
print(f"ADF Statstic: {result[0]}")
print(f"p-value:{result[1]}")
```

```
ADF Statstic: -2.1272017170757755
p-value:0.23371823999228158
```

[315]:
```
#decompose data
import statsmodels.api as sm
res = sm.tsa.seasonal_decompose(df['Close'],model= 'addictive',period = 365)
resplot = res.plot()
#upward trend
```



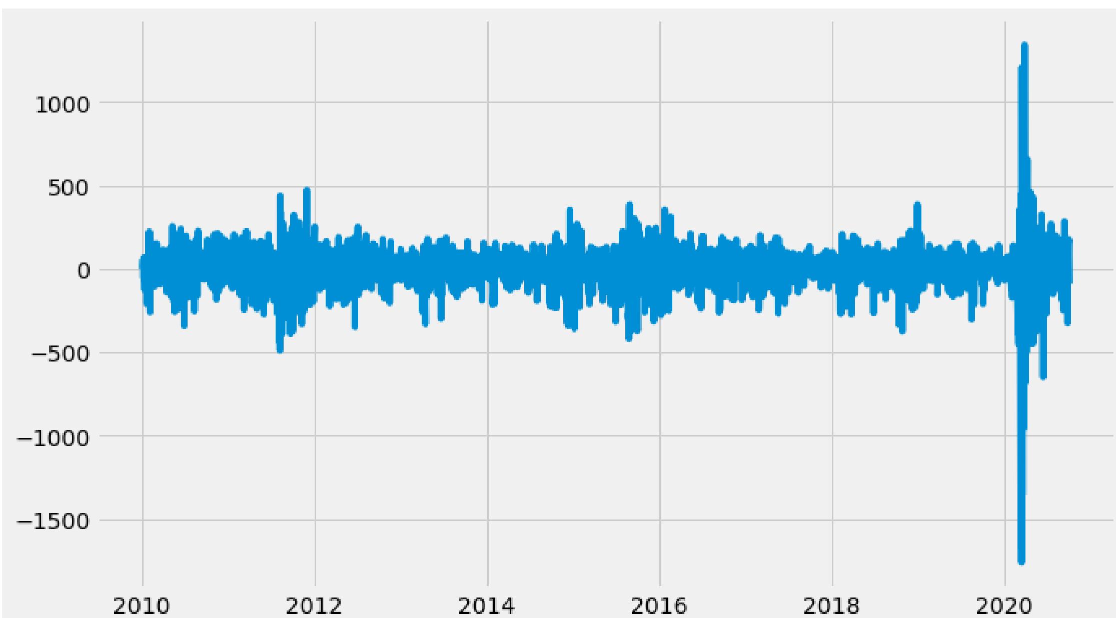
[317]:
```
def plot_df(df,x,y,title= "", xlabel = "Date", ylabel='Value',dpi=50):
    plt.plot(x,y)
    plt.show()
```

[318]:
```
#apply log transformation to stablize data
plt.plot(df.apply(np.log)['Close'])
```

[318]: [<matplotlib.lines.Line2D at 0x7f9e376e20d0>]

[319]:
```
#To covert data into stationary dataset, first differencing has to be applied.␣
 ↪With first differencing, empty field needs to be filled as 0.
plt.plot(df['Close'].diff(1).fillna(0))
```

[319]: [<matplotlib.lines.Line2D at 0x7f9e5af03e10>]



[320]:
```
#confirm stationarity
from statsmodels.tsa.stattools import adfuller
result = adfuller(np.log(df['Close']).diff(1).fillna(0))
print(f"ADF Statstic: {result[0]}")
print(f"p-value:{result[1]}")
```

```
ADF Statstic: -10.92108896454532
p-value:1.037610106877016e-19
```

[321]:
```
df_st= df.diff(1).fillna(0)
```

[322]:
```
#understand the structure of the stationary dataset
df_st.head()
```

[322]:

Date	Close	year	month	day
2010-01-04	0.00000	0.0	0.0	0.0
2010-01-05	21.19922	0.0	0.0	1.0
2010-01-06	56.40039	0.0	0.0	1.0
2010-01-07	-57.00000	0.0	0.0	1.0
2010-01-08	66.29981	0.0	0.0	1.0

[323]:
```
#With transformation to maintain stationarity, we need to be model back to the␣
↪original dataset in order to predict stock price.
df_revert=df_st.copy()
df_revert=df_revert.cumsum()
```

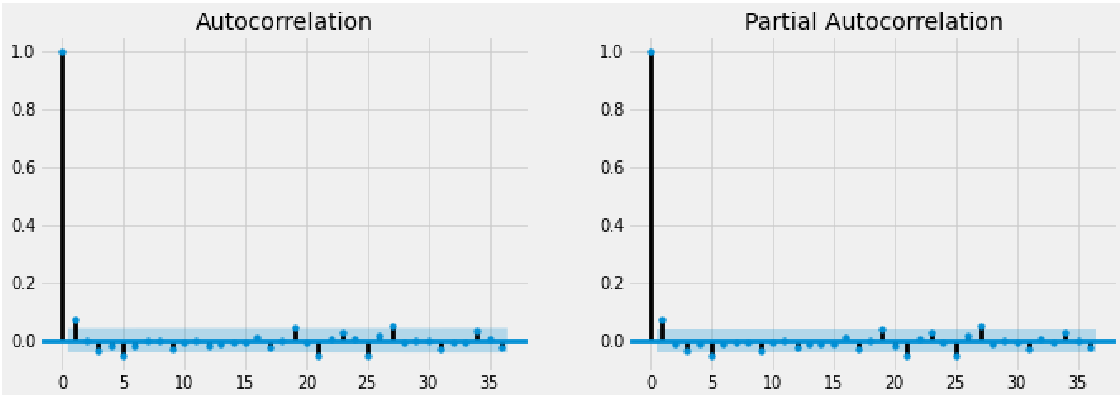
[331]:
```
df_revert.iloc[0,:]=df.iloc[0,:]
df_revert = df_revert.cumsum()
```

[332]:
```
#define data
class TimeSeriesData():
    def __init__(self, df):
        self.data = df
        self.stationary = self.stationarize(df)
        self.revert = self.revert(self.stationary, self.data)

    def revert(self, st, org):
        x = st.copy()
        x.iloc[0,:] = org.iloc[0,:]
        return x.cumsum()

    def stationarize(self, data):
        return data.diff(1).fillna(0)
```

[352]:
```
#split dataset
x_train = TimeSeriesData(df[:int((len(df)*0.85))])
```

```
x_test = TimeSeriesData(df[int((len(df)*0.85)):])
```

[353]:
```
#plot ACF and PACF for the stationary dataset
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
fig, axes = plt.subplots(1,2,figsize=(15,5), dpi= 50)
plot_acf(x_train.stationary['Close'].values.tolist(), lags=36,ax=axes[0]);
plot_pacf(x_train.stationary['Close'].values.tolist(), lags=36,ax=axes[1]);
```



[354]:
```
#using auto_arima to aquire p and q value with min AIC.
from pmdarima import auto_arima
model = auto_arima(x_train.data['Close'], trace=True, error_action='ignore',␣
↪suppress_warnings=True)
model.fit(x_train.data['Close'])
```

```
Performing stepwise search to minimize aic
 ARIMA(2,1,2)(0,0,0)[0] intercept   : AIC=27707.940, Time=1.57 sec
 ARIMA(0,1,0)(0,0,0)[0] intercept   : AIC=27722.152, Time=0.06 sec
 ARIMA(1,1,0)(0,0,0)[0] intercept   : AIC=27711.070, Time=0.12 sec
 ARIMA(0,1,1)(0,0,0)[0] intercept   : AIC=27711.001, Time=0.16 sec
 ARIMA(0,1,0)(0,0,0)[0]             : AIC=27720.872, Time=0.03 sec
 ARIMA(1,1,2)(0,0,0)[0] intercept   : AIC=27714.994, Time=0.27 sec
 ARIMA(2,1,1)(0,0,0)[0] intercept   : AIC=27707.115, Time=1.64 sec
 ARIMA(1,1,1)(0,0,0)[0] intercept   : AIC=27712.993, Time=0.16 sec
 ARIMA(2,1,0)(0,0,0)[0] intercept   : AIC=27712.917, Time=0.19 sec
 ARIMA(3,1,1)(0,0,0)[0] intercept   : AIC=27707.866, Time=1.87 sec
 ARIMA(3,1,0)(0,0,0)[0] intercept   : AIC=27712.282, Time=0.26 sec
 ARIMA(3,1,2)(0,0,0)[0] intercept   : AIC=27708.220, Time=2.53 sec
 ARIMA(2,1,1)(0,0,0)[0]             : AIC=27706.090, Time=0.50 sec
 ARIMA(1,1,1)(0,0,0)[0]             : AIC=27711.613, Time=0.21 sec
 ARIMA(2,1,0)(0,0,0)[0]             : AIC=27711.544, Time=0.08 sec
 ARIMA(3,1,1)(0,0,0)[0]             : AIC=27706.835, Time=0.48 sec
 ARIMA(2,1,2)(0,0,0)[0]             : AIC=27706.903, Time=0.53 sec
 ARIMA(1,1,0)(0,0,0)[0]             : AIC=27709.688, Time=0.05 sec
```

```
 ARIMA(1,1,2)(0,0,0)[0]             : AIC=27713.616, Time=0.11 sec
 ARIMA(3,1,0)(0,0,0)[0]             : AIC=27710.951, Time=0.10 sec
 ARIMA(3,1,2)(0,0,0)[0]             : AIC=27707.194, Time=0.85 sec

Best model:  ARIMA(2,1,1)(0,0,0)[0]
Total fit time: 11.772 seconds
```

[354]:
```
ARIMA(maxiter=50, method='lbfgs', order=(2, 1, 1), out_of_sample_size=0,
      scoring='mse', scoring_args={}, seasonal_order=(0, 0, 0, 0),
      start_params=None, suppress_warnings=True, trend=None,
      with_intercept=False)
```

[355]:
```
model_arima = ARIMA(x_train.data['Close'].values, order=(2,1,1))
```

[356]:
```
result_arima = model_arima.fit(disp=-1)
```

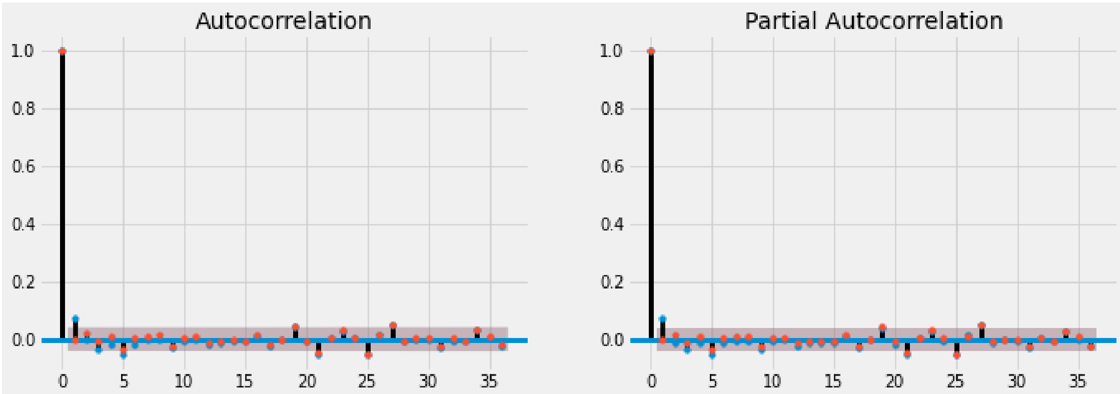
[357]:
```
print(result_arima.summary())
```

```
                             ARIMA Model Results
==============================================================================
Dep. Variable:                    D.y   No. Observations:                 2291
Model:                 ARIMA(2, 1, 1)   Log Likelihood              -13848.621
Method:                       css-mle   S.D. of innovations            102.089
Date:                Tue, 17 Nov 2020   AIC                          27707.242
Time:                        01:20:14   BIC                          27735.926
Sample:                             1   HQIC                         27717.702

==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
const          1.7918      1.819      0.985      0.325      -1.773       5.357
ar.L1.D.y      0.9619      0.058     16.567      0.000       0.848       1.076
ar.L2.D.y     -0.0939      0.021     -4.491      0.000      -0.135      -0.053
ma.L1.D.y     -0.8876      0.055    -16.179      0.000      -0.995      -0.780
                                    Roots
=============================================================================
                  Real          Imaginary           Modulus         Frequency
-----------------------------------------------------------------------------
AR.1            1.1741           +0.0000j            1.1741            0.0000
AR.2            9.0727           +0.0000j            9.0727            0.0000
MA.1            1.1267           +0.0000j            1.1267            0.0000
-----------------------------------------------------------------------------
```

[358]:
```
#understand residual
residuals = pd.DataFrame(result_arima.resid)
```

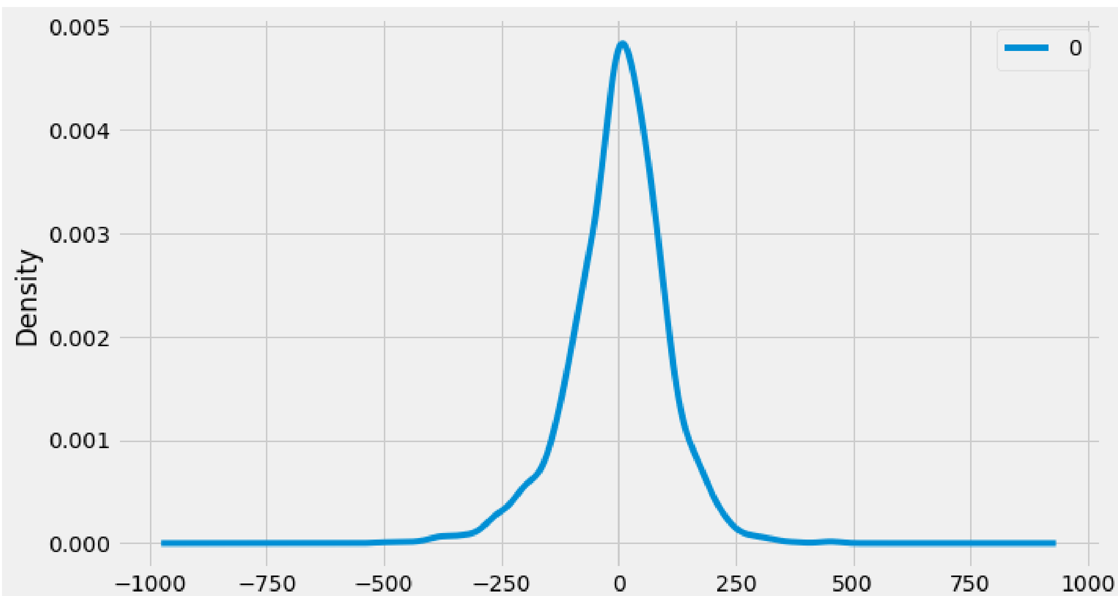
[359]:
```
plot_acf(residuals, lags=36, ax=axes[0])
plot_pacf(residuals, lags=36, ax=axes[1])
```

[359]:

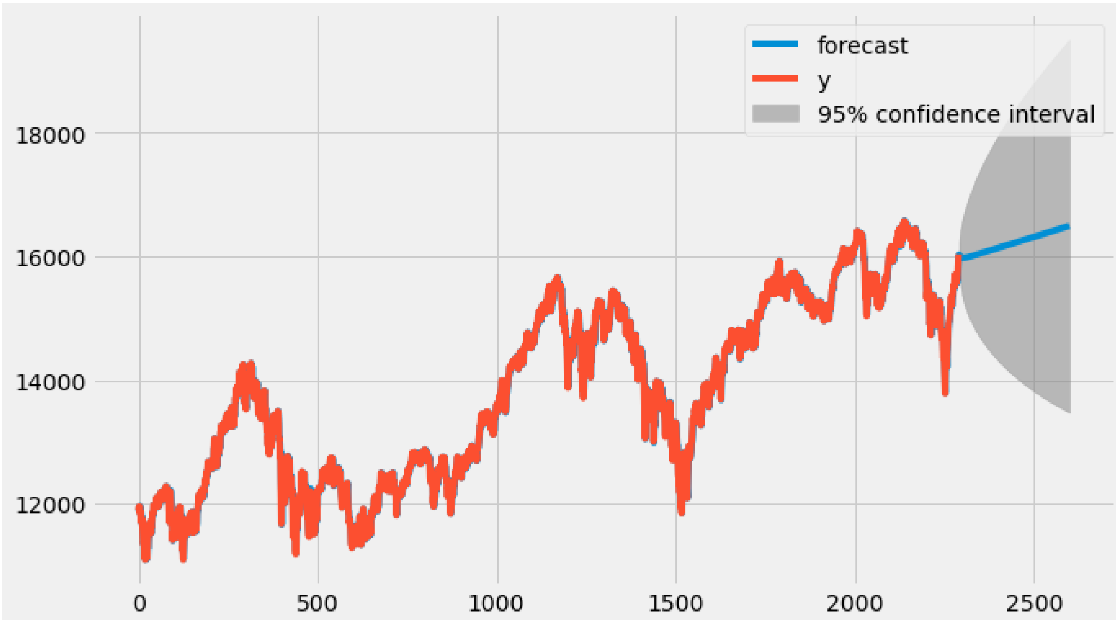


[360]:
```
residuals.plot(kind='kde')
```

[360]: <matplotlib.axes._subplots.AxesSubplot at 0x7f9e5a422ed0>



[364]:
```
result_arima.plot_predict(1,2600);
```



[365]:
```
prediction = result_arima.predict(len(df)-208,len(df)-1)
```

[366]:
```
from statsmodels.tools.eval_measures import rmse
#RMSE for ARIMA Model
err_ARIMA = rmse(x_test.data['Close'], np.append(x_train.data.iloc[-1,:
↪]['Close'], prediction).cumsum()[1:])
print('RMSE with ARIMA', err_ARIMA)
```

```
RMSE with ARIMA 1145.9902860642562
```

[ ]:
```

```

[ ]:
```

```

[ ]:
```
#using LSTM to predict stock price
```

[44]:
```
pip install tensorflow
```

```
Collecting tensorflow
  Using cached tensorflow-2.3.1-cp37-cp37m-manylinux2010_x86_64.whl (320.4 MB)
Requirement already satisfied: six>=1.12.0 in /opt/conda/lib/python3.7/site-
packages (from tensorflow) (1.14.0)
Collecting grpcio>=1.8.6
  Using cached grpcio-1.33.2-cp37-cp37m-manylinux2014_x86_64.whl (3.8 MB)
Collecting google-pasta>=0.1.8
  Using cached google_pasta-0.2.0-py3-none-any.whl (57 kB)
Collecting astunparse==1.6.3
```

```
  Using cached astunparse-1.6.3-py2.py3-none-any.whl (12 kB)
Requirement already satisfied: h5py<2.11.0,>=2.10.0 in
/opt/conda/lib/python3.7/site-packages (from tensorflow) (2.10.0)
Collecting keras-preprocessing<1.2,>=1.1.1
  Using cached Keras_Preprocessing-1.1.2-py2.py3-none-any.whl (42 kB)
Collecting tensorboard<3,>=2.3.0
  Using cached tensorboard-2.4.0-py3-none-any.whl (10.6 MB)
Collecting tensorflow-estimator<2.4.0,>=2.3.0
  Using cached tensorflow_estimator-2.3.0-py2.py3-none-any.whl (459 kB)
Collecting opt-einsum>=2.3.2
  Using cached opt_einsum-3.3.0-py3-none-any.whl (65 kB)
Collecting absl-py>=0.7.0
  Using cached absl_py-0.11.0-py3-none-any.whl (127 kB)
Requirement already satisfied: numpy<1.19.0,>=1.16.0 in
/opt/conda/lib/python3.7/site-packages (from tensorflow) (1.18.4)
Processing ./.cache/pip/wheels/3f/e3/ec/8a8336ff196023622fbcb36de0c5a5c218cbb241
11d1d4c7f2/termcolor-1.1.0-py3-none-any.whl
Requirement already satisfied: wheel>=0.26 in /opt/conda/lib/python3.7/site-
packages (from tensorflow) (0.34.2)
Requirement already satisfied: protobuf>=3.9.2 in /opt/conda/lib/python3.7/site-
packages (from tensorflow) (3.11.4)
Processing ./.cache/pip/wheels/62/76/4c/aa25851149f3f6d9785f6c869387ad82b3fd3758
2fa8147ac6/wrapt-1.12.1-cp37-cp37m-linux_x86_64.whl
Collecting gast==0.3.3
  Using cached gast-0.3.3-py2.py3-none-any.whl (9.7 kB)
Requirement already satisfied: requests<3,>=2.21.0 in
/opt/conda/lib/python3.7/site-packages (from tensorboard<3,>=2.3.0->tensorflow)
(2.23.0)
Requirement already satisfied: setuptools>=41.0.0 in
/opt/conda/lib/python3.7/site-packages (from tensorboard<3,>=2.3.0->tensorflow)
(46.1.3.post20200325)
Collecting werkzeug>=0.11.15
  Using cached Werkzeug-1.0.1-py2.py3-none-any.whl (298 kB)
Collecting markdown>=2.6.8
  Using cached Markdown-3.3.3-py3-none-any.whl (96 kB)
Collecting tensorboard-plugin-wit>=1.6.0
  Using cached tensorboard_plugin_wit-1.7.0-py3-none-any.whl (779 kB)
Collecting google-auth-oauthlib<0.5,>=0.4.1
  Using cached google_auth_oauthlib-0.4.2-py2.py3-none-any.whl (18 kB)
Requirement already satisfied: google-auth<2,>=1.6.3 in
/opt/conda/lib/python3.7/site-packages (from tensorboard<3,>=2.3.0->tensorflow)
(1.16.1)
Requirement already satisfied: idna<3,>=2.5 in /opt/conda/lib/python3.7/site-
packages (from requests<3,>=2.21.0->tensorboard<3,>=2.3.0->tensorflow) (2.9)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in
/opt/conda/lib/python3.7/site-packages (from
requests<3,>=2.21.0->tensorboard<3,>=2.3.0->tensorflow) (1.25.9)
Requirement already satisfied: certifi>=2017.4.17 in
```

```
/opt/conda/lib/python3.7/site-packages (from
requests<3,>=2.21.0->tensorboard<3,>=2.3.0->tensorflow) (2020.4.5.2)
Requirement already satisfied: chardet<4,>=3.0.2 in
/opt/conda/lib/python3.7/site-packages (from
requests<3,>=2.21.0->tensorboard<3,>=2.3.0->tensorflow) (3.0.4)
Requirement already satisfied: importlib-metadata; python_version < "3.8" in
/opt/conda/lib/python3.7/site-packages (from
markdown>=2.6.8->tensorboard<3,>=2.3.0->tensorflow) (1.6.0)
Requirement already satisfied: requests-oauthlib>=0.7.0 in
/opt/conda/lib/python3.7/site-packages (from google-auth-
oauthlib<0.5,>=0.4.1->tensorboard<3,>=2.3.0->tensorflow) (1.3.0)
Requirement already satisfied: pyasn1-modules>=0.2.1 in
/opt/conda/lib/python3.7/site-packages (from google-
auth<2,>=1.6.3->tensorboard<3,>=2.3.0->tensorflow) (0.2.8)
Requirement already satisfied: rsa<4.1,>=3.1.4 in /opt/conda/lib/python3.7/site-
packages (from google-auth<2,>=1.6.3->tensorboard<3,>=2.3.0->tensorflow) (4.0)
Requirement already satisfied: cachetools<5.0,>=2.0.0 in
/opt/conda/lib/python3.7/site-packages (from google-
auth<2,>=1.6.3->tensorboard<3,>=2.3.0->tensorflow) (4.1.0)
Requirement already satisfied: zipp>=0.5 in /opt/conda/lib/python3.7/site-
packages (from importlib-metadata; python_version <
"3.8"->markdown>=2.6.8->tensorboard<3,>=2.3.0->tensorflow) (3.1.0)
Requirement already satisfied: oauthlib>=3.0.0 in /opt/conda/lib/python3.7/site-
packages (from requests-oauthlib>=0.7.0->google-auth-
oauthlib<0.5,>=0.4.1->tensorboard<3,>=2.3.0->tensorflow) (3.0.1)
Requirement already satisfied: pyasn1<0.5.0,>=0.4.6 in
/opt/conda/lib/python3.7/site-packages (from pyasn1-modules>=0.2.1->google-
auth<2,>=1.6.3->tensorboard<3,>=2.3.0->tensorflow) (0.4.8)
Installing collected packages: grpcio, google-pasta, astunparse, keras-
preprocessing, absl-py, werkzeug, markdown, tensorboard-plugin-wit, google-auth-
oauthlib, tensorboard, tensorflow-estimator, opt-einsum, termcolor, wrapt, gast,
tensorflow
Successfully installed absl-py-0.11.0 astunparse-1.6.3 gast-0.3.3 google-auth-
oauthlib-0.4.2 google-pasta-0.2.0 grpcio-1.33.2 keras-preprocessing-1.1.2
markdown-3.3.3 opt-einsum-3.3.0 tensorboard-2.4.0 tensorboard-plugin-wit-1.7.0
tensorflow-2.3.1 tensorflow-estimator-2.3.0 termcolor-1.1.0 werkzeug-1.0.1
wrapt-1.12.1
Note: you may need to restart the kernel to use updated packages.
```

[45]:
```
pip install keras
```

```
Collecting keras
  Using cached Keras-2.4.3-py2.py3-none-any.whl (36 kB)
Requirement already satisfied: scipy>=0.14 in /opt/conda/lib/python3.7/site-
packages (from keras) (1.4.1)
Requirement already satisfied: numpy>=1.9.1 in /opt/conda/lib/python3.7/site-
packages (from keras) (1.18.4)
Requirement already satisfied: h5py in /opt/conda/lib/python3.7/site-packages
```

```
(from keras) (2.10.0)
Requirement already satisfied: pyyaml in /opt/conda/lib/python3.7/site-packages
(from keras) (5.3.1)
Requirement already satisfied: six in /opt/conda/lib/python3.7/site-packages
(from h5py->keras) (1.14.0)
Installing collected packages: keras
Successfully installed keras-2.4.3
Note: you may need to restart the kernel to use updated packages.
```

[46]:
```
pip install pandas-datareader
```

```
Requirement already satisfied: pandas-datareader in
/opt/conda/lib/python3.7/site-packages (0.9.0)
Requirement already satisfied: lxml in /opt/conda/lib/python3.7/site-packages
(from pandas-datareader) (4.5.1)
Requirement already satisfied: requests>=2.19.0 in
/opt/conda/lib/python3.7/site-packages (from pandas-datareader) (2.23.0)
Requirement already satisfied: pandas>=0.23 in /opt/conda/lib/python3.7/site-
packages (from pandas-datareader) (1.0.3)
Requirement already satisfied: certifi>=2017.4.17 in
/opt/conda/lib/python3.7/site-packages (from requests>=2.19.0->pandas-
datareader) (2020.4.5.2)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in
/opt/conda/lib/python3.7/site-packages (from requests>=2.19.0->pandas-
datareader) (1.25.9)
Requirement already satisfied: chardet<4,>=3.0.2 in
/opt/conda/lib/python3.7/site-packages (from requests>=2.19.0->pandas-
datareader) (3.0.4)
Requirement already satisfied: idna<3,>=2.5 in /opt/conda/lib/python3.7/site-
packages (from requests>=2.19.0->pandas-datareader) (2.9)
Requirement already satisfied: python-dateutil>=2.6.1 in
/opt/conda/lib/python3.7/site-packages (from pandas>=0.23->pandas-datareader)
(2.8.1)
Requirement already satisfied: pytz>=2017.2 in /opt/conda/lib/python3.7/site-
packages (from pandas>=0.23->pandas-datareader) (2020.1)
Requirement already satisfied: numpy>=1.13.3 in /opt/conda/lib/python3.7/site-
packages (from pandas>=0.23->pandas-datareader) (1.18.4)
Requirement already satisfied: six>=1.5 in /opt/conda/lib/python3.7/site-
packages (from python-dateutil>=2.6.1->pandas>=0.23->pandas-datareader) (1.14.0)
Note: you may need to restart the kernel to use updated packages.
```

[380]:
```
from pandas_datareader import data
import datetime as dt
from matplotlib import pyplot as plt
from sklearn import model_selection
from sklearn.metrics import confusion_matrix
from sklearn.preprocessing import StandardScaler
```

```
from sklearn.model_selection import train_test_split
import numpy as np
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import LSTM
from keras.layers import Dropout
```

[449]:
```
#similar to before, upload NASDAQ data
df=pd.read_csv("IXIC_v1.csv", sep=",")
from datetime import datetime
con=df['Date']
df['Date']=pd.to_datetime(df['Date'])
df.set_index('Date', inplace=True)
test = df[2164:]
train = df[:2163]
```

[450]:
```
df['Date'] = df.index
data2 = pd.DataFrame(columns = ['Date', 'Close'])
data2['Date'] = df['Date']
data2['Close'] = df['Close']
```

[451]:
```
#scale and reshape data
train_set = data2.iloc[:, 1:2].values
sc = MinMaxScaler(feature_range = (0, 1))
training_set_scaled = sc.fit_transform(train_set)
X_train = []
y_train = []
for i in range(60, 2600):
    X_train.append(training_set_scaled[i-60:i, 0])
    y_train.append(training_set_scaled[i, 0])
X_train, y_train = np.array(X_train), np.array(y_train)
X_train = np.reshape(X_train, (X_train.shape[0], X_train.shape[1], 1))
```

[452]:
```
#add layers
regressor = Sequential()
regressor.add(LSTM(units = 50, return_sequences = True, input_shape = (X_train.
 ↪shape[1], 1)))
regressor.add(Dropout(0.2))
regressor.add(LSTM(units = 50, return_sequences = True))
regressor.add(Dropout(0.2))
regressor.add(LSTM(units = 50, return_sequences = True))
regressor.add(Dropout(0.2))
regressor.add(LSTM(units = 50))
regressor.add(Dropout(0.2))
regressor.add(Dense(units = 1))
```

[453]:
```
#add optimizer and build model
regressor.compile(optimizer = 'adam', loss = 'mean_squared_error')
regressor.fit(X_train, y_train, epochs = 30, batch_size = 30)
```

```
Epoch 1/30
85/85 [==============================] - 13s 154ms/step - loss: 0.0071
Epoch 2/30
85/85 [==============================] - 12s 146ms/step - loss: 0.0018
Epoch 3/30
85/85 [==============================] - 12s 143ms/step - loss: 0.0016
Epoch 4/30
85/85 [==============================] - 12s 143ms/step - loss: 0.0014
Epoch 5/30
85/85 [==============================] - 12s 141ms/step - loss: 0.0013
Epoch 6/30
85/85 [==============================] - 12s 140ms/step - loss: 0.0013
Epoch 7/30
85/85 [==============================] - 12s 138ms/step - loss: 0.0012
Epoch 8/30
85/85 [==============================] - 11s 135ms/step - loss: 0.0012
Epoch 9/30
85/85 [==============================] - 13s 151ms/step - loss: 0.0011
Epoch 10/30
85/85 [==============================] - 12s 147ms/step - loss: 0.0010
Epoch 11/30
85/85 [==============================] - 12s 137ms/step - loss: 0.0010
Epoch 12/30
85/85 [==============================] - 12s 136ms/step - loss: 9.9016e-04
Epoch 13/30
85/85 [==============================] - 12s 137ms/step - loss: 9.7918e-04
Epoch 14/30
85/85 [==============================] - 12s 137ms/step - loss: 9.2474e-04
Epoch 15/30
85/85 [==============================] - 12s 136ms/step - loss: 7.9696e-04
Epoch 16/30
85/85 [==============================] - 12s 136ms/step - loss: 7.9756e-04
Epoch 17/30
85/85 [==============================] - 12s 135ms/step - loss: 7.4945e-04
Epoch 18/30
85/85 [==============================] - 11s 134ms/step - loss: 7.2670e-04
Epoch 19/30
85/85 [==============================] - 12s 136ms/step - loss: 7.6239e-04
Epoch 20/30
85/85 [==============================] - 12s 136ms/step - loss: 7.3629e-04
Epoch 21/30
85/85 [==============================] - 11s 135ms/step - loss: 7.2187e-04
Epoch 22/30
```

```
85/85 [==============================] - 13s 147ms/step - loss: 6.8401e-04
Epoch 23/30
85/85 [==============================] - 12s 135ms/step - loss: 7.4183e-04
Epoch 24/30
85/85 [==============================] - 12s 135ms/step - loss: 8.4028e-04
Epoch 25/30
85/85 [==============================] - 12s 136ms/step - loss: 6.9651e-04
Epoch 26/30
85/85 [==============================] - 11s 135ms/step - loss: 6.5826e-04
Epoch 27/30
85/85 [==============================] - 11s 133ms/step - loss: 6.7812e-04
Epoch 28/30
85/85 [==============================] - 11s 134ms/step - loss: 6.5652e-04
Epoch 29/30
85/85 [==============================] - 11s 134ms/step - loss: 6.5054e-04
Epoch 30/30
85/85 [==============================] - 11s 134ms/step - loss: 6.2934e-04
```

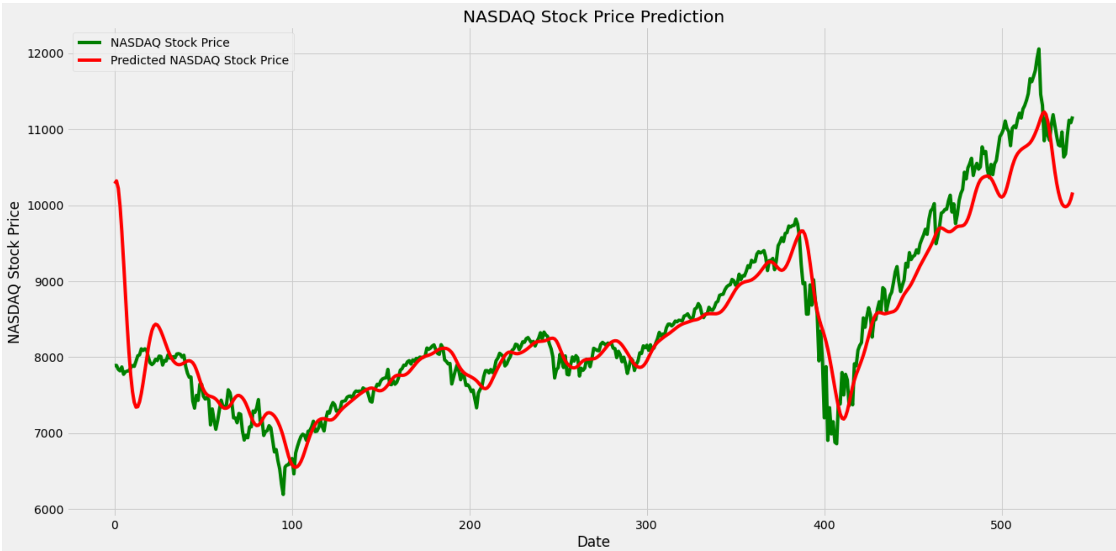
[453]:
```
<tensorflow.python.keras.callbacks.History at 0x7f9de2d63510>
```

[454]:
```
#train the model
testdataframe= test
testdataframe['Date'] = testdataframe.index
testdata = pd.DataFrame(columns = ['Date','Close'])
testdata['Data'] = testdataframe['Date']
testdata['Close'] = testdataframe['Close']
real_stock_price = testdata.iloc[:, 1:2].values
dataset_total = pd.concat((data2['Close'], testdata['Close']), axis = 0)
inputs = dataset_total[len(dataset_total) - len(testdata) - 60:].values
inputs = inputs.reshape(-1,1)
inputs = sc.transform(inputs)
X_test = []
for i in range(60, 601):
    X_test.append(inputs[i-60:i, 0])
X_test = np.array(X_test)
X_test = np.reshape(X_test, (X_test.shape[0], X_test.shape[1], 1))
```

[455]:
```
predicted_stock_price = regressor.predict(X_test)
predicted_stock_price = sc.inverse_transform(predicted_stock_price)
```

[456]:
```
plt.figure(figsize=(20,10))
plt.plot(real_stock_price, color = 'green', label = 'NASDAQ Stock Price')
plt.plot(predicted_stock_price, color = 'red', label = 'Predicted NASDAQ Stock␣
↪Price')
plt.title('NASDAQ Stock Price Prediction')
plt.xlabel('Date')
plt.ylabel('NASDAQ Stock Price')
```

```
plt.legend()
plt.show()
```



[457]:
```
rmse_predict= np.reshape(predicted_stock_price,541)
```

[458]:
```
test["Close"].values.shape
```

[458]: (541,)

[459]:
```
rmse_predict.shape
```

[459]: (541,)

[460]:
```
#RMSE for LSTM Model
err_LSTM = rmse(test["Close"].values, rmse_predict)
print('RMSE with LSTM', err_LSTM)
```

RMSE with LSTM 391.79083600796884

[405]:
```
#repeat the same process for TSX
df=pd.read_csv("GSPTSE_v1.csv", sep=",")
from datetime import datetime
con=df['Date']
df['Date']=pd.to_datetime(df['Date'])
df.set_index('Date', inplace=True)
test = df[2164:]
train = df[:2163]
```

[406]:
```python
df['Date'] = df.index
data2 = pd.DataFrame(columns = ['Date', 'Close'])
data2['Date'] = df['Date']
data2['Close'] = df['Close']
```

[407]:
```python
train_set = data2.iloc[:, 1:2].values
sc = MinMaxScaler(feature_range = (0, 1))
training_set_scaled = sc.fit_transform(train_set)
X_train = []
y_train = []
for i in range(60, 2600):
    X_train.append(training_set_scaled[i-60:i, 0])
    y_train.append(training_set_scaled[i, 0])
X_train, y_train = np.array(X_train), np.array(y_train)
X_train = np.reshape(X_train, (X_train.shape[0], X_train.shape[1], 1))
```

[408]:
```python
regressor = Sequential()
regressor.add(LSTM(units = 50, return_sequences = True, input_shape = (X_train.
 ↪shape[1], 1)))
regressor.add(Dropout(0.2))
regressor.add(LSTM(units = 50, return_sequences = True))
regressor.add(Dropout(0.2))
regressor.add(LSTM(units = 50, return_sequences = True))
regressor.add(Dropout(0.2))
regressor.add(LSTM(units = 50))
regressor.add(Dropout(0.2))
regressor.add(Dense(units = 1))
```

[417]:
```python
regressor.compile(optimizer = 'adam', loss = 'mean_squared_error')
regressor.fit(X_train, y_train, epochs = 80, batch_size = 32)
```

```
Epoch 1/80
80/80 [==============================] - 13s 162ms/step - loss: 0.0013
Epoch 2/80
80/80 [==============================] - 13s 163ms/step - loss: 0.0013
Epoch 3/80
80/80 [==============================] - 12s 151ms/step - loss: 0.0013
Epoch 4/80
80/80 [==============================] - 13s 158ms/step - loss: 0.0011
Epoch 5/80
80/80 [==============================] - 13s 160ms/step - loss: 0.0013
Epoch 6/80
80/80 [==============================] - 12s 155ms/step - loss: 0.0011
Epoch 7/80
80/80 [==============================] - 14s 178ms/step - loss: 0.0011
Epoch 8/80
80/80 [==============================] - 12s 154ms/step - loss: 0.0012
```

```
Epoch 9/80
80/80 [==============================] - 13s 156ms/step - loss: 0.0011
Epoch 10/80
80/80 [==============================] - 13s 162ms/step - loss: 0.0011
Epoch 11/80
80/80 [==============================] - 11s 143ms/step - loss: 0.0012
Epoch 12/80
80/80 [==============================] - 11s 139ms/step - loss: 0.0011
Epoch 13/80
80/80 [==============================] - 11s 132ms/step - loss: 0.0011
Epoch 14/80
80/80 [==============================] - 11s 133ms/step - loss: 0.0013
Epoch 15/80
80/80 [==============================] - 11s 133ms/step - loss: 0.0011
Epoch 16/80
80/80 [==============================] - 11s 133ms/step - loss: 0.0010
Epoch 17/80
80/80 [==============================] - 11s 133ms/step - loss: 0.0010
Epoch 18/80
80/80 [==============================] - 11s 133ms/step - loss: 9.8654e-04
Epoch 19/80
80/80 [==============================] - 11s 133ms/step - loss: 0.0010
Epoch 20/80
80/80 [==============================] - 11s 134ms/step - loss: 0.0011
Epoch 21/80
80/80 [==============================] - 11s 134ms/step - loss: 0.0010
Epoch 22/80
80/80 [==============================] - 11s 131ms/step - loss: 9.8064e-04
Epoch 23/80
80/80 [==============================] - 11s 136ms/step - loss: 0.0011
Epoch 24/80
80/80 [==============================] - 11s 136ms/step - loss: 0.0010
Epoch 25/80
80/80 [==============================] - 11s 132ms/step - loss: 9.5185e-04
Epoch 26/80
80/80 [==============================] - 11s 133ms/step - loss: 9.8578e-04
Epoch 27/80
80/80 [==============================] - 11s 133ms/step - loss: 9.8236e-04
Epoch 28/80
80/80 [==============================] - 11s 133ms/step - loss: 9.2821e-04
Epoch 29/80
80/80 [==============================] - 11s 133ms/step - loss: 0.0011
Epoch 30/80
80/80 [==============================] - 11s 132ms/step - loss: 9.8527e-04
Epoch 31/80
80/80 [==============================] - 11s 134ms/step - loss: 9.9202e-04
Epoch 32/80
80/80 [==============================] - 11s 133ms/step - loss: 9.8537e-04
```

```
Epoch 33/80
80/80 [==============================] - 11s 132ms/step - loss: 9.0084e-04
Epoch 34/80
80/80 [==============================] - 11s 131ms/step - loss: 9.0383e-04
Epoch 35/80
80/80 [==============================] - 12s 154ms/step - loss: 9.6011e-04
Epoch 36/80
80/80 [==============================] - 11s 132ms/step - loss: 9.8266e-04
Epoch 37/80
80/80 [==============================] - 12s 144ms/step - loss: 9.4536e-04
Epoch 38/80
80/80 [==============================] - 11s 132ms/step - loss: 0.0010
Epoch 39/80
80/80 [==============================] - 11s 132ms/step - loss: 9.3070e-04
Epoch 40/80
80/80 [==============================] - 11s 133ms/step - loss: 9.3518e-04
Epoch 41/80
80/80 [==============================] - 11s 132ms/step - loss: 8.8781e-04
Epoch 42/80
80/80 [==============================] - 11s 132ms/step - loss: 9.3565e-04
Epoch 43/80
80/80 [==============================] - 11s 132ms/step - loss: 9.3241e-04
Epoch 44/80
80/80 [==============================] - 10s 131ms/step - loss: 8.5825e-04
Epoch 45/80
80/80 [==============================] - 10s 130ms/step - loss: 0.0011
Epoch 46/80
80/80 [==============================] - 10s 130ms/step - loss: 9.4952e-04
Epoch 47/80
80/80 [==============================] - 10s 131ms/step - loss: 9.0473e-04
Epoch 48/80
80/80 [==============================] - 11s 132ms/step - loss: 9.4199e-04
Epoch 49/80
80/80 [==============================] - 11s 132ms/step - loss: 8.8700e-04
Epoch 50/80
80/80 [==============================] - 10s 131ms/step - loss: 9.0246e-04
Epoch 51/80
80/80 [==============================] - 11s 132ms/step - loss: 9.4137e-04
Epoch 52/80
80/80 [==============================] - 11s 142ms/step - loss: 9.3281e-04
Epoch 53/80
80/80 [==============================] - 11s 132ms/step - loss: 8.8455e-04
Epoch 54/80
80/80 [==============================] - 11s 131ms/step - loss: 8.6919e-04
Epoch 55/80
80/80 [==============================] - 11s 133ms/step - loss: 9.1682e-04
Epoch 56/80
80/80 [==============================] - 11s 137ms/step - loss: 9.2724e-04
```

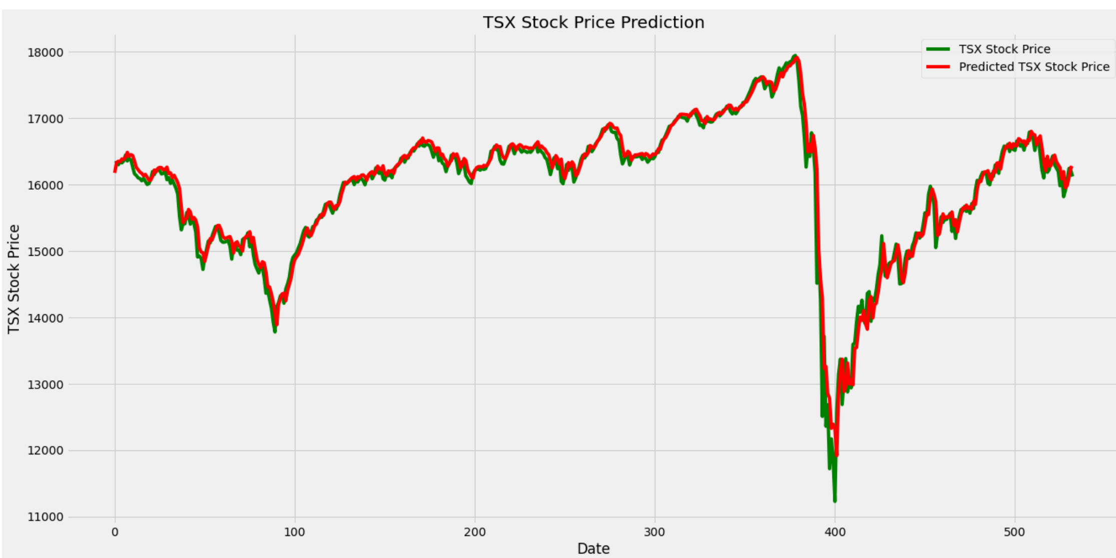
```
Epoch 57/80
80/80 [==============================] - 11s 136ms/step - loss: 8.7430e-04
Epoch 58/80
80/80 [==============================] - 11s 138ms/step - loss: 8.9795e-04
Epoch 59/80
80/80 [==============================] - 11s 140ms/step - loss: 8.3710e-04
Epoch 60/80
80/80 [==============================] - 11s 138ms/step - loss: 8.6120e-04
Epoch 61/80
80/80 [==============================] - 11s 139ms/step - loss: 8.9001e-04
Epoch 62/80
80/80 [==============================] - 11s 138ms/step - loss: 9.2847e-04
Epoch 63/80
80/80 [==============================] - 12s 151ms/step - loss: 8.5671e-04
Epoch 64/80
80/80 [==============================] - 12s 145ms/step - loss: 8.7639e-04
Epoch 65/80
80/80 [==============================] - 11s 138ms/step - loss: 9.1322e-04
Epoch 66/80
80/80 [==============================] - 11s 138ms/step - loss: 9.2609e-04
Epoch 67/80
80/80 [==============================] - 11s 138ms/step - loss: 0.0010
Epoch 68/80
80/80 [==============================] - 11s 139ms/step - loss: 8.7199e-04
Epoch 69/80
80/80 [==============================] - 11s 141ms/step - loss: 8.3027e-04
Epoch 70/80
80/80 [==============================] - 11s 139ms/step - loss: 8.4242e-04
Epoch 71/80
80/80 [==============================] - 11s 141ms/step - loss: 8.7307e-04
Epoch 72/80
80/80 [==============================] - 11s 140ms/step - loss: 7.9310e-04
Epoch 73/80
80/80 [==============================] - 11s 143ms/step - loss: 8.0419e-04
Epoch 74/80
80/80 [==============================] - 11s 142ms/step - loss: 8.1921e-04
Epoch 75/80
80/80 [==============================] - 11s 141ms/step - loss: 8.6678e-04
Epoch 76/80
80/80 [==============================] - 11s 138ms/step - loss: 8.1471e-04
Epoch 77/80
80/80 [==============================] - 11s 137ms/step - loss: 8.7284e-04
Epoch 78/80
80/80 [==============================] - 11s 135ms/step - loss: 8.3848e-04
Epoch 79/80
80/80 [==============================] - 12s 149ms/step - loss: 8.1106e-04
Epoch 80/80
80/80 [==============================] - 11s 142ms/step - loss: 8.4214e-04
```

[417]: <tensorflow.python.keras.callbacks.History at 0x7f9e06503190>

[418]:
```
testdataframe= test
testdataframe['Date'] = testdataframe.index
testdata = pd.DataFrame(columns = ['Date','Close'])
testdata['Date'] = testdataframe['Date']
testdata['Close'] = testdataframe['Close']
real_stock_price = testdata.iloc[:, 1:2].values
dataset_total = pd.concat((data2['Close'], testdata['Close']), axis = 0)
inputs = dataset_total[len(dataset_total) - len(testdata) - 60:].values
inputs = inputs.reshape(-1,1)
inputs = sc.transform(inputs)
X_test = []
for i in range(60, 593):
    X_test.append(inputs[i-60:i, 0])
X_test = np.array(X_test)
X_test = np.reshape(X_test, (X_test.shape[0], X_test.shape[1], 1))
```

[419]:
```
predicted_stock_price = regressor.predict(X_test)
predicted_stock_price = sc.inverse_transform(predicted_stock_price)
```

[420]:
```
plt.figure(figsize=(20,10))
plt.plot(real_stock_price, color = 'green', label = 'TSX Stock Price')
plt.plot(predicted_stock_price, color = 'red', label = 'Predicted TSX Stock␣
↪Price')
plt.title('TSX Stock Price Prediction')
plt.xlabel('Date')
plt.ylabel('TSX Stock Price')
plt.legend()
plt.show()
```



[421]:
```
rmse_predict= np.reshape(predicted_stock_price,533)
```

[422]:
```
test["Close"].values.shape
```

[422]: (533,)

[423]:
```
rmse_predict.shape
```

[423]: (533,)

[424]:
```
#RMSE for LSTM Model
err_LSTM = rmse(test["Close"].values, rmse_predict)
print('RMSE with LSTM', err_LSTM)
```

```
RMSE with LSTM 199.6917074408309
```

[ ]:
```
```