File Edit View Run Kernel Tabs Settings Help

DSM_Project5.ipynb

Code Python 3

[146]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from pandas.tools.plotting import autocorrelation_plot
from statsmodels.graphics.tsaplots import plot_pacf
from statsmodels.tsa.arima_model import ARIMA, ARMAResults
import datetime
import sys
import seaborn as sns
import statsmodels
import statsmodels.stats.diagnostic as diag
from statsmodels.tsa.stattools import adfuller
from scipy.stats.mstats import normaltest
from matplotlib.pyplot import acorr
#plt.style.use('fivethirtyeight')
import warnings
warnings.warn('ignore')
%matplotlib inline
```

```
C:\ProgramData\Anaconda3\lib\site-packages\ipykernel_launcher.py:17: UserWarning: ignore
```

[2]:
```
df = pd.read_csv('data_stocks.csv')
df.head()
```

[2]:

	DATE	SP500	NASDAQ.AAL	NASDAQ.AAPL	NASDAQ.ADBE	NASDAQ.ADI	NASDAQ.ADP	NASDAQ.ADSK	NASDAQ.AKAM	NASDAQ.ALXN	...	NYSE.WYN	NYSE.XEC	NYS

File Edit View Run Kernel Tabs Settings Help

DSM_Project5.ipynb

Code Python 3

[2]:

	DATE	SP500	NASDAQ.AAL	NASDAQ.AAPL	NASDAQ.ADBE	NASDAQ.ADI	NASDAQ.ADP	NASDAQ.ADSK	NASDAQ.AKAM	NASDAQ.ALXN	...	NYSE.WYN	NYSE.XEC	NYS
0	1491226200	2363.6101	42.3300	143.6800	129.6300	82.040	102.2300	85.2200	59.760	121.52	...	84.370	119.035	
1	1491226260	2364.1001	42.3600	143.7000	130.3200	82.080	102.1400	85.6500	59.840	121.48	...	84.370	119.035	
2	1491226320	2362.6799	42.3100	143.6901	130.2250	82.030	102.2125	85.5100	59.795	121.93	...	84.585	119.260	
3	1491226380	2364.3101	42.3700	143.6400	130.0729	82.000	102.1400	85.4872	59.620	121.44	...	84.460	119.260	
4	1491226440	2364.8501	42.5378	143.6600	129.8800	82.035	102.0600	85.7001	59.620	121.60	...	84.470	119.610	

5 rows × 502 columns

[3]:
```
#Pick up the following stocks and generate forecasts accordingly
```

[4]:
```
stock_features =['NASDAQ.AAPL','NASDAQ.ADP','NASDAQ.CBOE','NASDAQ.CSCO','NASDAQ.EBAY']
col_list = ['DATE'] + stock_features
df1 = df[col_list]
df1.head()
```

[4]:

	DATE	NASDAQ.AAPL	NASDAQ.ADP	NASDAQ.CBOE	NASDAQ.CSCO	NASDAQ.EBAY
0	1491226200	143.6800	102.2300	81.03	33.7400	33.3975
1	1491226260	143.7000	102.1400	81.21	33.8800	33.3950

File Edit View Run Kernel Tabs Settings Help

DSM_Project5.ipynb

Code

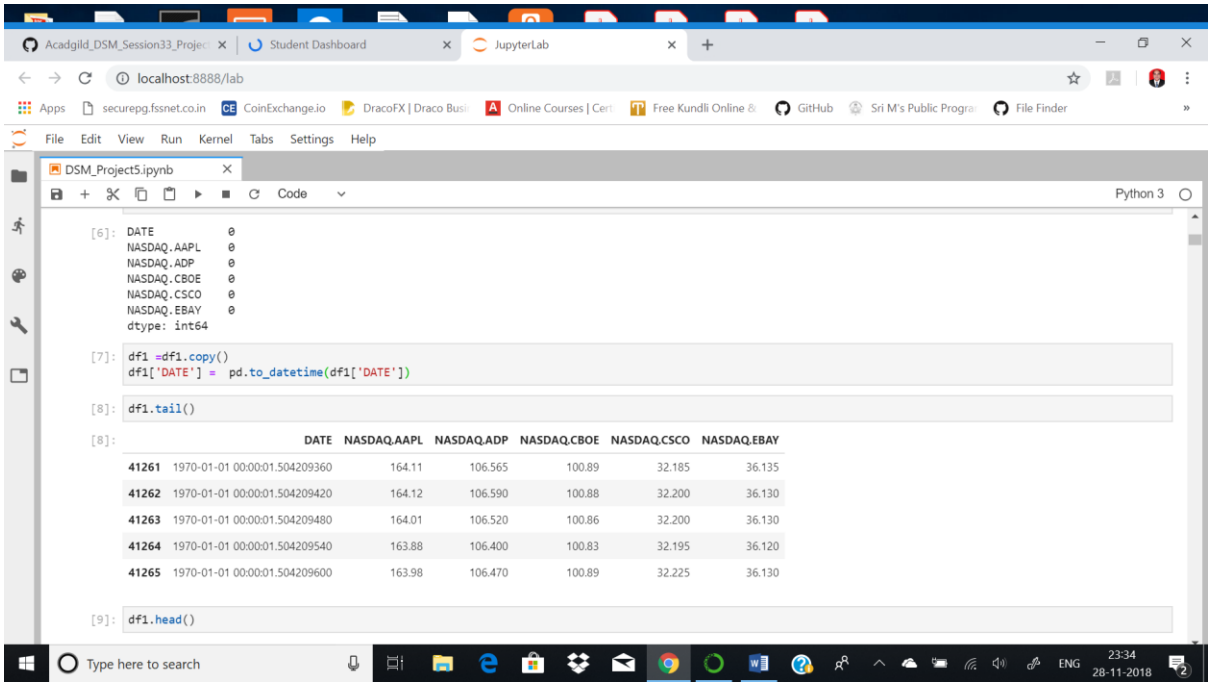
0	1491226200	143.6800	102.2300	81.03	33.7400	33.3975
1	1491226260	143.7000	102.1400	81.21	33.8800	33.3950
2	1491226320	143.6901	102.2125	81.21	33.9000	33.4100
3	1491226380	143.6400	102.1400	81.13	33.8499	33.3350
4	1491226440	143.6600	102.0600	81.12	33.8400	33.4000

[5]:
```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 41266 entries, 0 to 41265
Data columns (total 6 columns):
DATE           41266 non-null int64
NASDAQ.AAPL    41266 non-null float64
NASDAQ.ADP     41266 non-null float64
NASDAQ.CBOE    41266 non-null float64
NASDAQ.CSCO    41266 non-null float64
NASDAQ.EBAY    41266 non-null float64
dtypes: float64(5), int64(1)
memory usage: 1.9 MB
```

[6]:
```
df1.isnull().sum()
```

[6]:
```
DATE           0
NASDAQ.AAPL    0
```



```
[6]: DATE          0
     NASDAQ.AAPL   0
     NASDAQ.ADP    0
     NASDAQ.CBOE   0
     NASDAQ.CSCO   0
     NASDAQ.EBAY   0
     dtype: int64
```

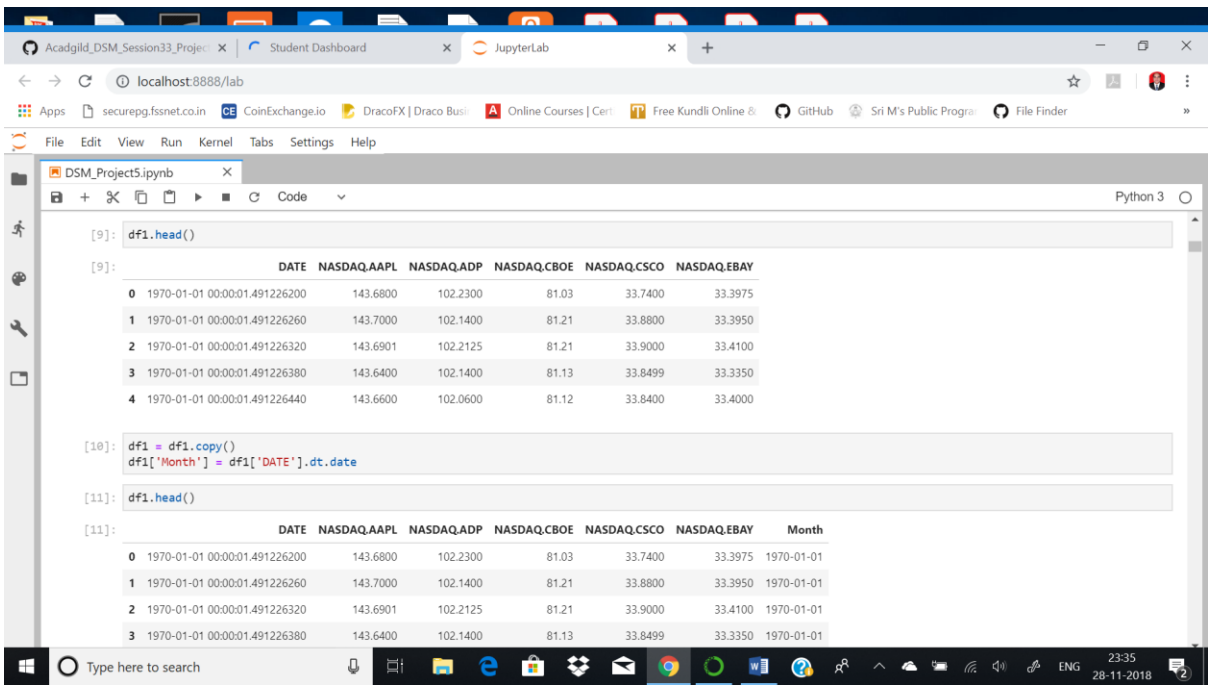
```
[7]: df1 =df1.copy()
     df1['DATE'] =  pd.to_datetime(df1['DATE'])
```

```
[8]: df1.tail()
```

[8]:

	DATE	NASDAQ.AAPL	NASDAQ.ADP	NASDAQ.CBOE	NASDAQ.CSCO	NASDAQ.EBAY
41261	1970-01-01 00:00:01.504209360	164.11	106.565	100.89	32.185	36.135
41262	1970-01-01 00:00:01.504209420	164.12	106.590	100.88	32.200	36.130
41263	1970-01-01 00:00:01.504209480	164.01	106.520	100.86	32.200	36.130
41264	1970-01-01 00:00:01.504209540	163.88	106.400	100.83	32.195	36.120
41265	1970-01-01 00:00:01.504209600	163.98	106.470	100.89	32.225	36.130

```
[9]: df1.head()
```



```
[9]: df1.head()
```

[9]:

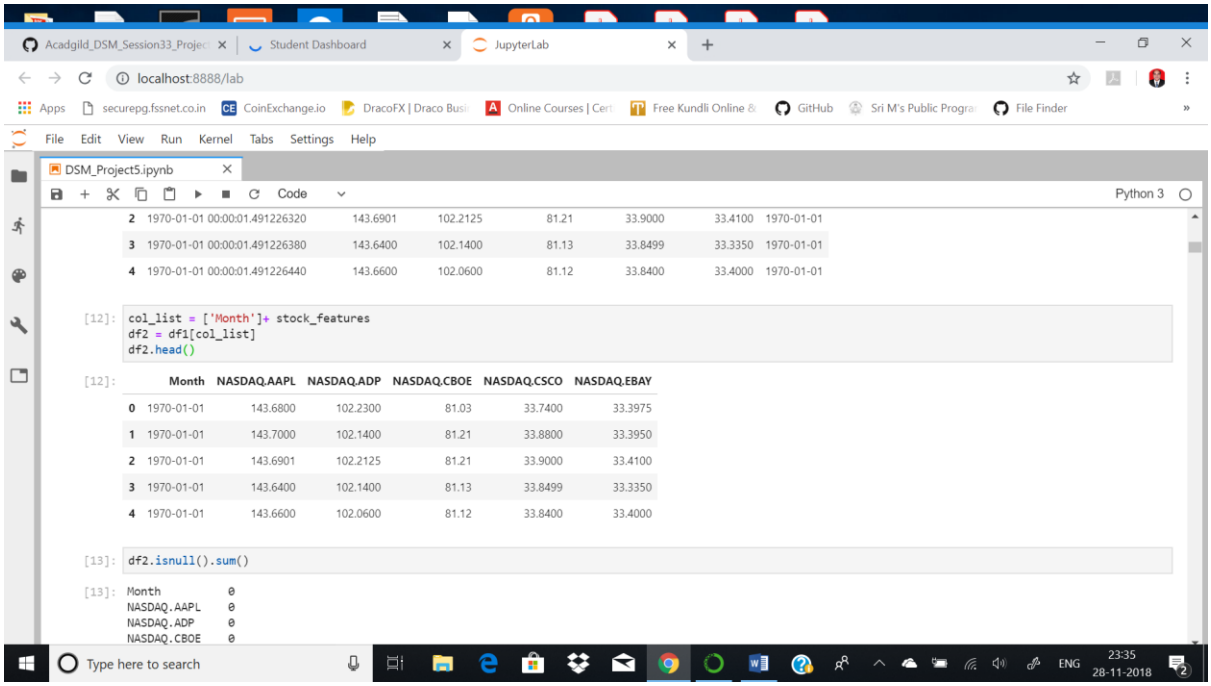
	DATE	NASDAQ.AAPL	NASDAQ.ADP	NASDAQ.CBOE	NASDAQ.CSCO	NASDAQ.EBAY
0	1970-01-01 00:00:01.491226200	143.6800	102.2300	81.03	33.7400	33.3975
1	1970-01-01 00:00:01.491226260	143.7000	102.1400	81.21	33.8800	33.3950
2	1970-01-01 00:00:01.491226320	143.6901	102.2125	81.21	33.9000	33.4100
3	1970-01-01 00:00:01.491226380	143.6400	102.1400	81.13	33.8499	33.3350
4	1970-01-01 00:00:01.491226440	143.6600	102.0600	81.12	33.8400	33.4000

```
[10]: df1 = df1.copy()
      df1['Month'] = df1['DATE'].dt.date
```

```
[11]: df1.head()
```

[11]:

	DATE	NASDAQ.AAPL	NASDAQ.ADP	NASDAQ.CBOE	NASDAQ.CSCO	NASDAQ.EBAY	Month
0	1970-01-01 00:00:01.491226200	143.6800	102.2300	81.03	33.7400	33.3975	1970-01-01
1	1970-01-01 00:00:01.491226260	143.7000	102.1400	81.21	33.8800	33.3950	1970-01-01
2	1970-01-01 00:00:01.491226320	143.6901	102.2125	81.21	33.9000	33.4100	1970-01-01
3	1970-01-01 00:00:01.491226380	143.6400	102.1400	81.13	33.8499	33.3350	1970-01-01



2	1970-01-01 00:00:01.491226320	143.6901	102.2125	81.21	33.9000	33.4100	1970-01-01
3	1970-01-01 00:00:01.491226380	143.6400	102.1400	81.13	33.8499	33.3350	1970-01-01
4	1970-01-01 00:00:01.491226440	143.6600	102.0600	81.12	33.8400	33.4000	1970-01-01

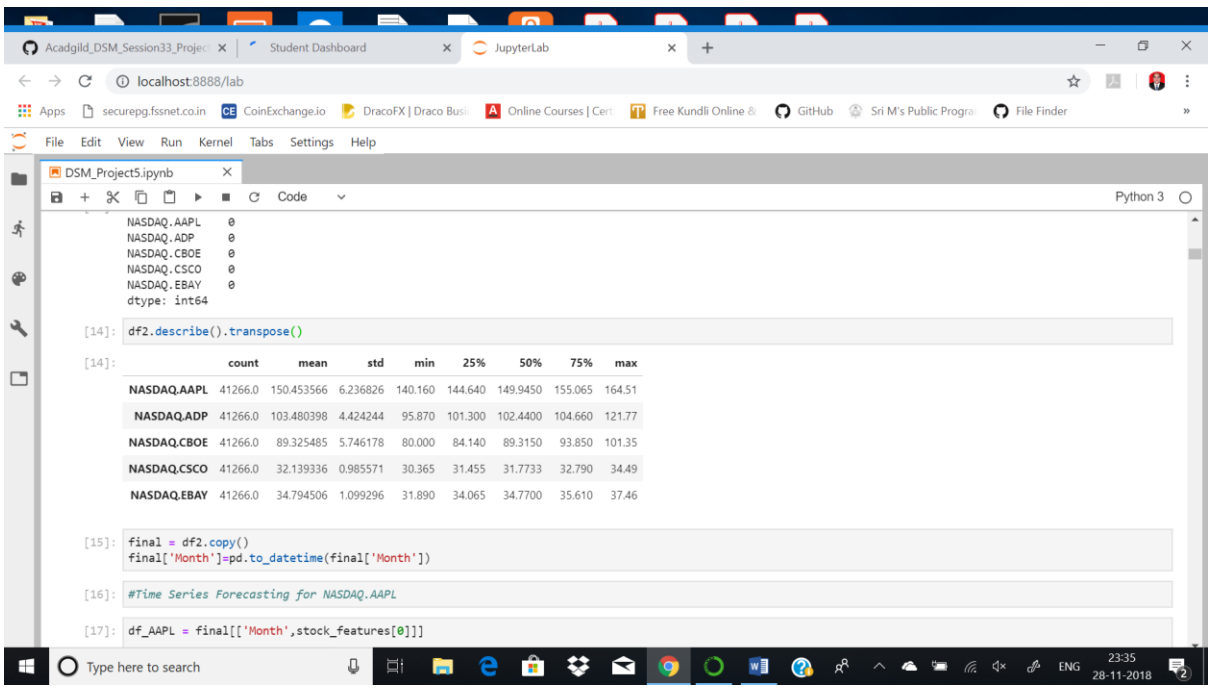
[12]:
```
col_list = ['Month']+ stock_features
df2 = df1[col_list]
df2.head()
```

[12]:

	Month	NASDAQ.AAPL	NASDAQ.ADP	NASDAQ.CBOE	NASDAQ.CSCO	NASDAQ.EBAY
0	1970-01-01	143.6800	102.2300	81.03	33.7400	33.3975
1	1970-01-01	143.7000	102.1400	81.21	33.8800	33.3950
2	1970-01-01	143.6901	102.2125	81.21	33.9000	33.4100
3	1970-01-01	143.6400	102.1400	81.13	33.8499	33.3350
4	1970-01-01	143.6600	102.0600	81.12	33.8400	33.4000

[13]:
```
df2.isnull().sum()
```

[13]:
```
Month          0
NASDAQ.AAPL    0
NASDAQ.ADP     0
NASDAQ.CBOE    0
```



```
NASDAQ.AAPL    0
NASDAQ.ADP     0
NASDAQ.CBOE    0
NASDAQ.CSCO    0
NASDAQ.EBAY    0
dtype: int64
```

[14]:
```
df2.describe().transpose()
```

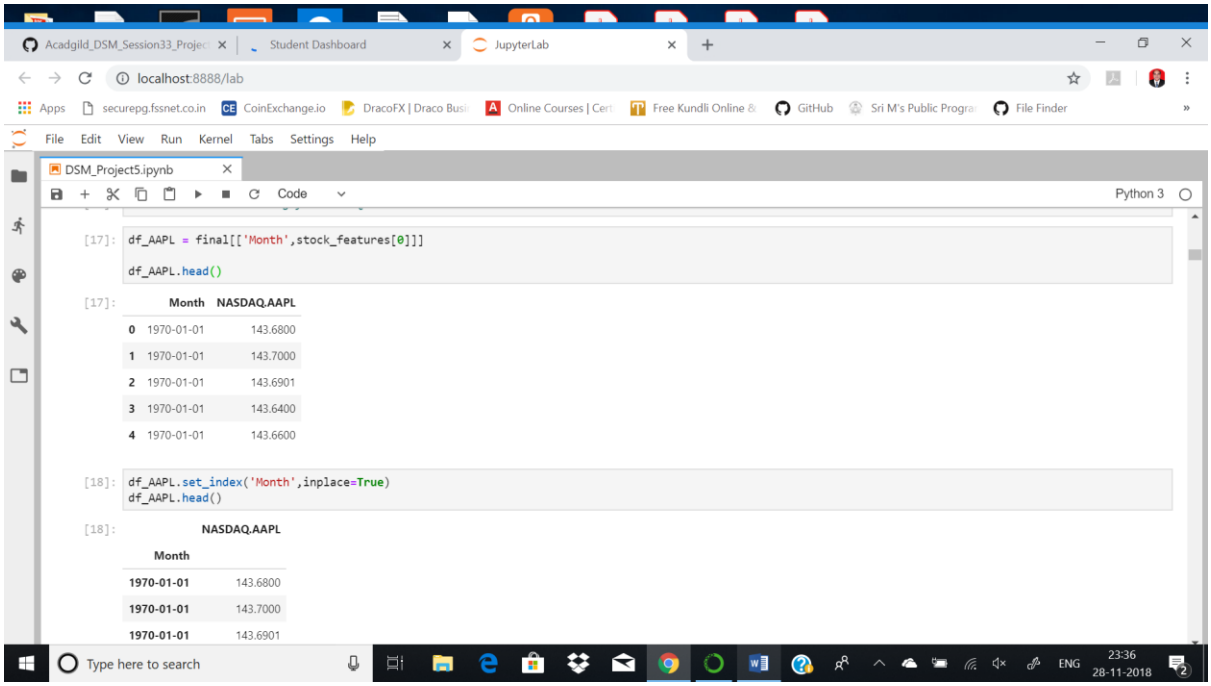
[14]:

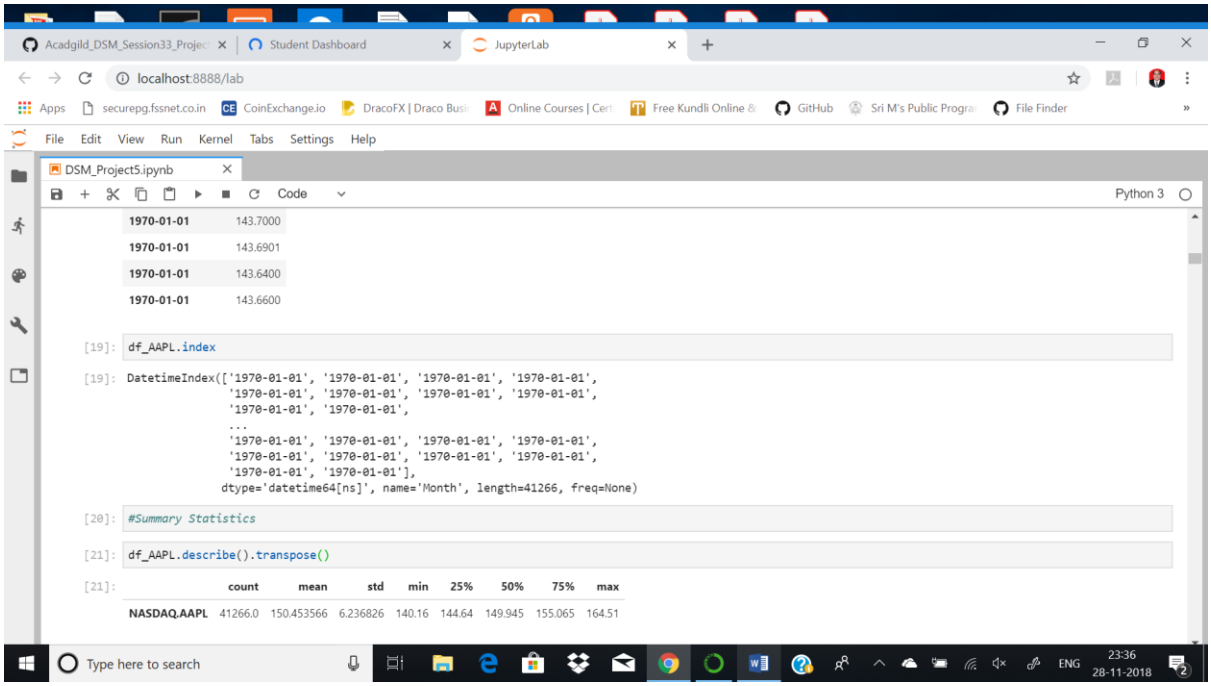
	count	mean	std	min	25%	50%	75%	max
NASDAQ.AAPL	41266.0	150.453566	6.236826	140.160	144.640	149.9450	155.065	164.51
NASDAQ.ADP	41266.0	103.480398	4.424244	95.870	101.300	102.4400	104.660	121.77
NASDAQ.CBOE	41266.0	89.325485	5.746178	80.000	84.140	89.3150	93.850	101.35
NASDAQ.CSCO	41266.0	32.139336	0.985571	30.365	31.455	31.7733	32.790	34.49
NASDAQ.EBAY	41266.0	34.794506	1.099296	31.890	34.065	34.7700	35.610	37.46

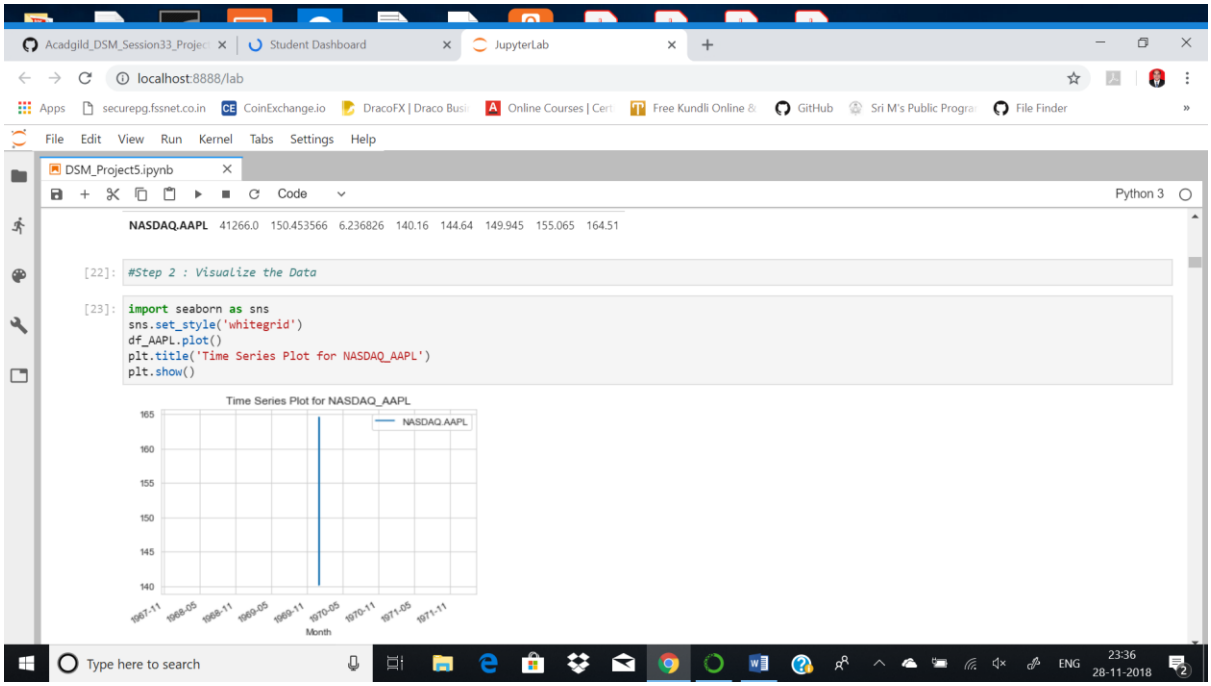
[15]:
```
final = df2.copy()
final['Month']=pd.to_datetime(final['Month'])
```

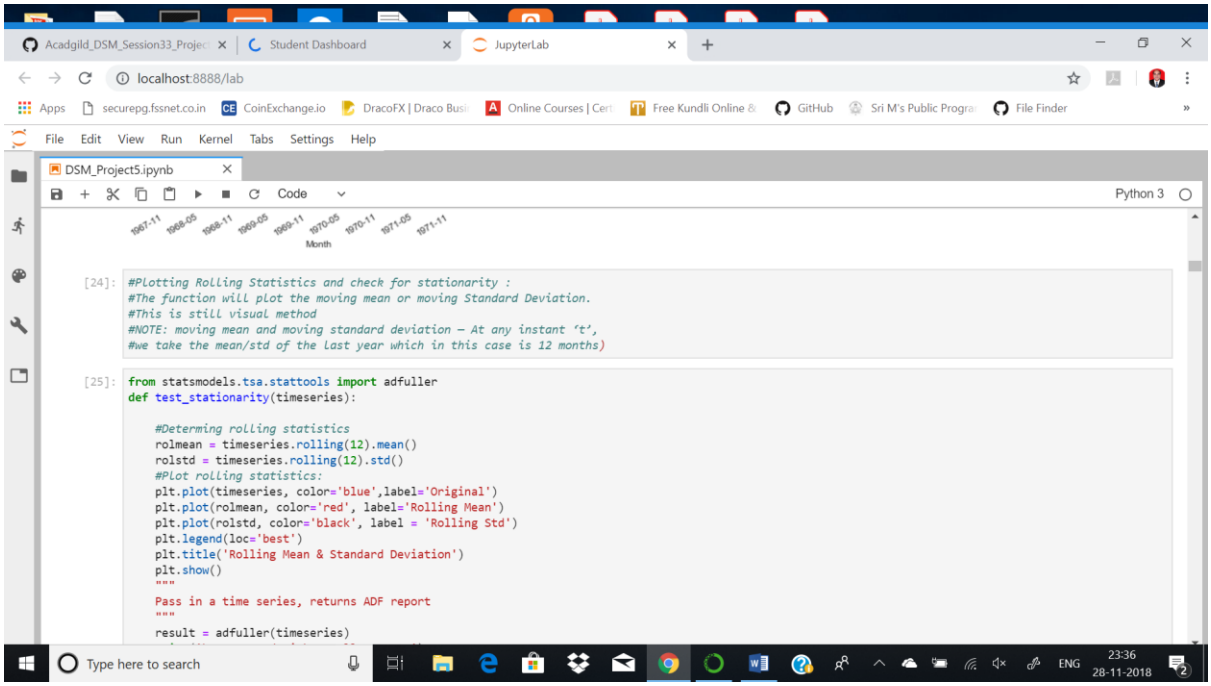
[16]:
```
#Time Series Forecasting for NASDAQ.AAPL
```

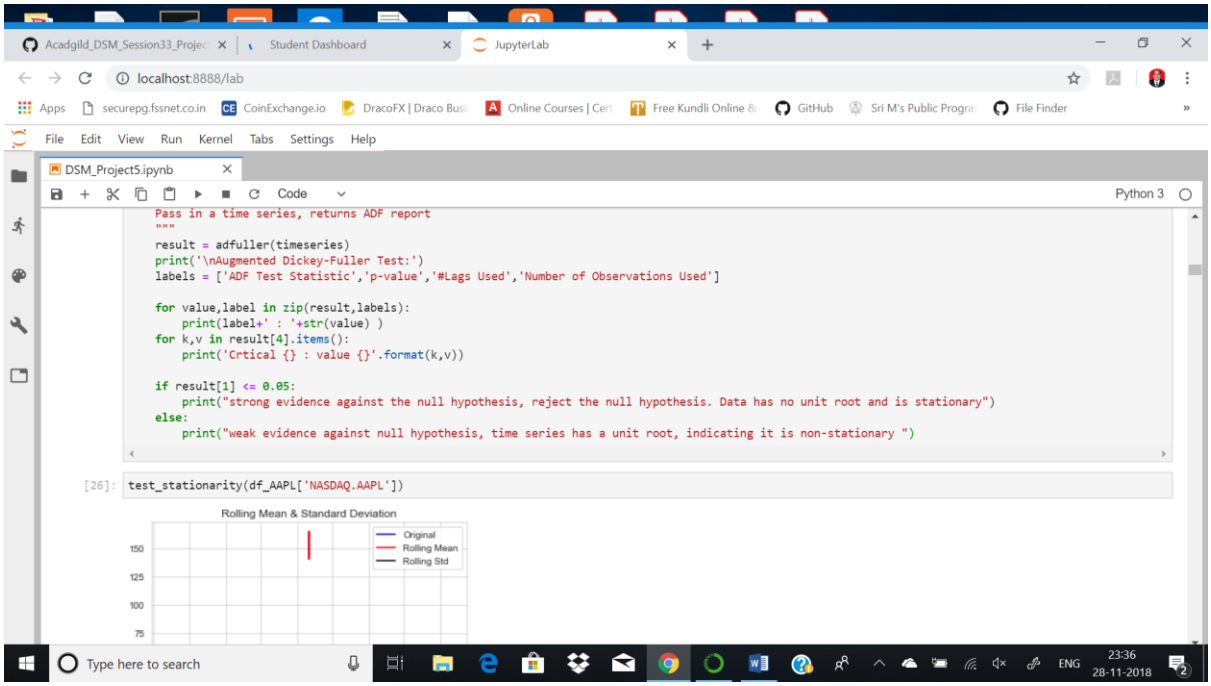
[17]:
```
df_AAPL = final[['Month',stock_features[0]]]
```

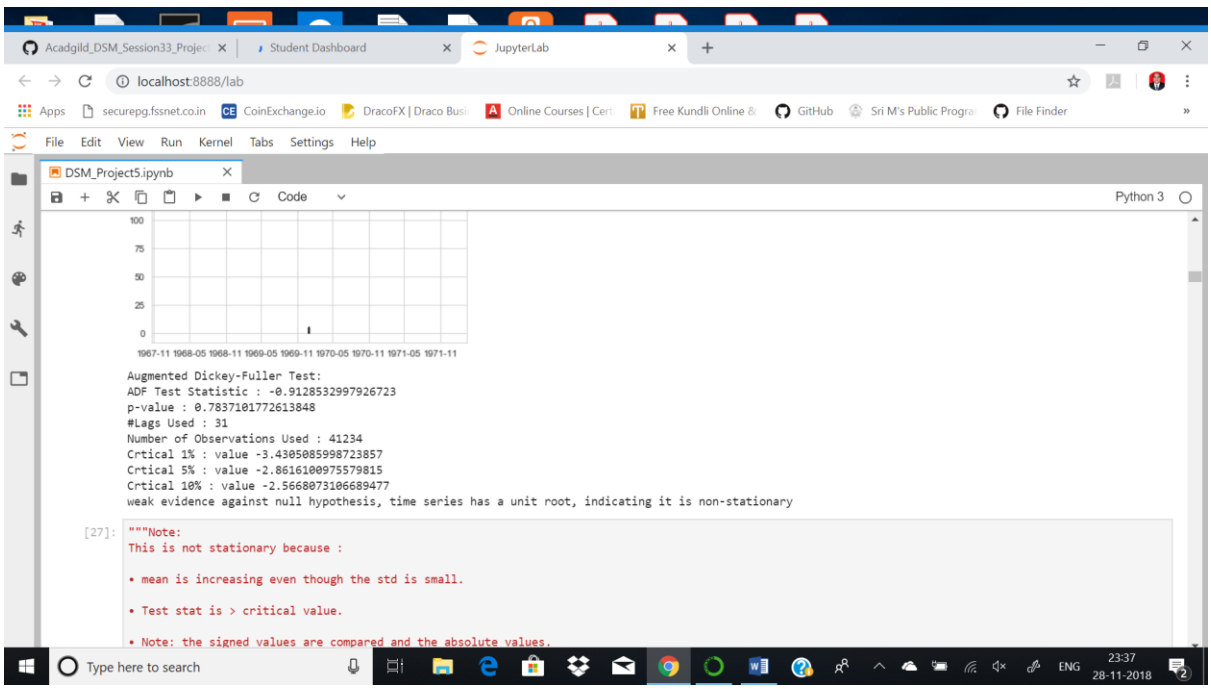

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
[17]: df_AAPL = final[['Month',stock_features[0]]]
df_AAPL.head()
[17]:
Month NASDAQ.AAPL
0 1970-01-01 143.6800
1 1970-01-01 143.7000
2 1970-01-01 143.6901
3 1970-01-01 143.6400
4 1970-01-01 143.6600
[18]: df_AAPL.set_index('Month',inplace=True)
df_AAPL.head()
[18]:
NASDAQ.AAPL
Month
1970-01-01 143.6800
1970-01-01 143.7000
1970-01-01 143.6901
Type here to search
ENG
23:36
28-11-2018

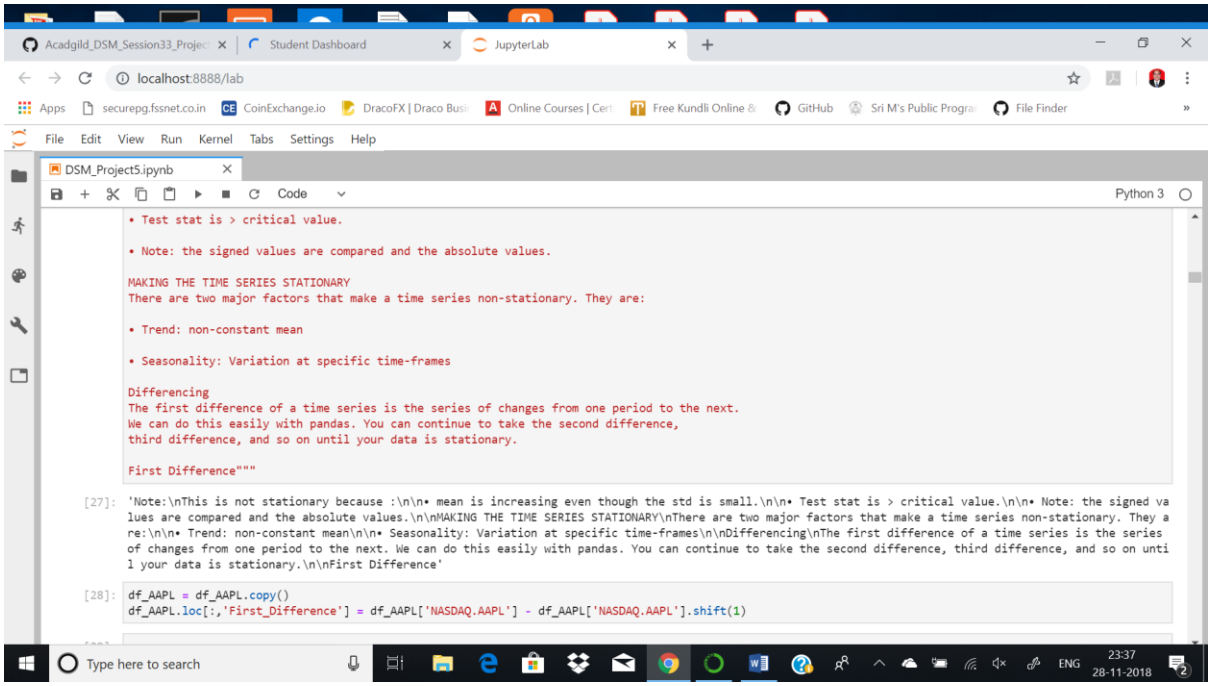

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
1970-01-01 143.7000
1970-01-01 143.6901
1970-01-01 143.6400
1970-01-01 143.6600
[19]: df_AAPL.index
[19]: DatetimeIndex(['1970-01-01', '1970-01-01', '1970-01-01', '1970-01-01',
'1970-01-01', '1970-01-01', '1970-01-01', '1970-01-01',
'1970-01-01', '1970-01-01',
...
'1970-01-01', '1970-01-01', '1970-01-01', '1970-01-01',
'1970-01-01', '1970-01-01', '1970-01-01', '1970-01-01',
'1970-01-01', '1970-01-01'],
dtype='datetime64[ns]', name='Month', length=41266, freq=None)
[20]: #Summary Statistics
[21]: df_AAPL.describe().transpose()
[21]:
count mean std min 25% 50% 75% max
NASDAQ.AAPL 41266.0 150.453566 6.236826 140.16 144.64 149.945 155.065 164.51
Type here to search
ENG
23:36
28-11-2018

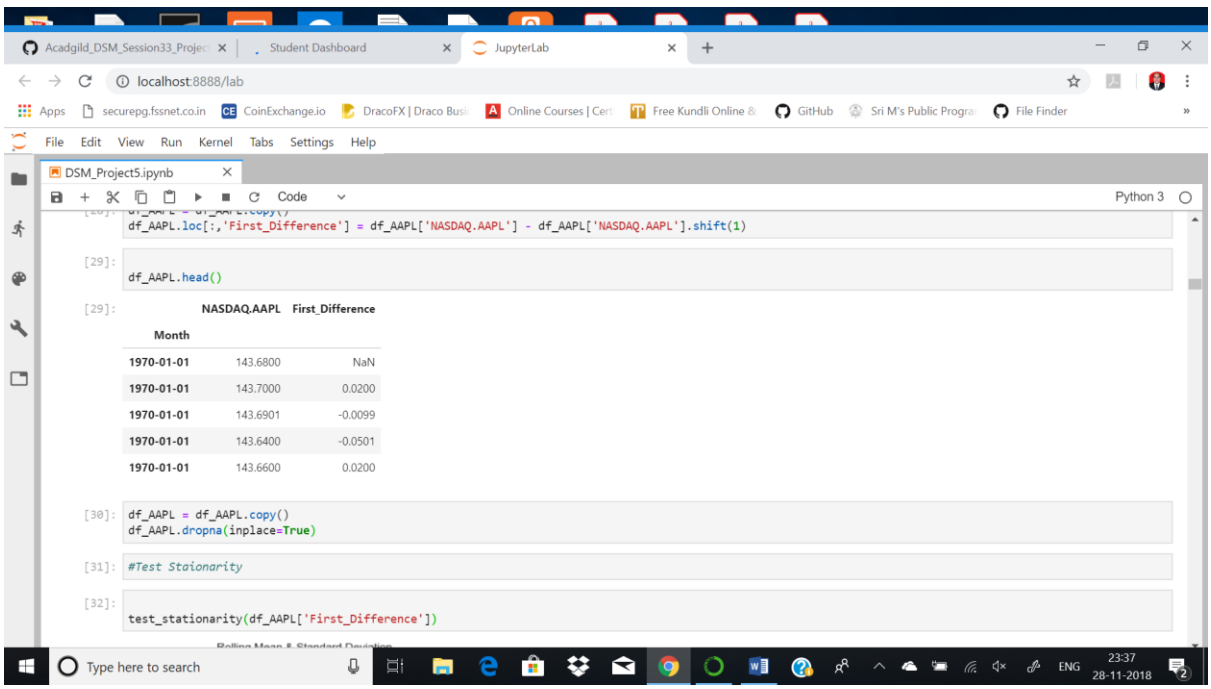

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
Apps
securepg.fssnet.co.in
CoinExchange.io
DracoFX | Draco Busi
Online Courses | Cert
Free Kundli Online &
GitHub
Sri M's Public Progra
File Finder
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
NASDAQ.AAPL 41266.0 150.453566 6.236826 140.16 144.64 149.945 155.065 164.51
[22]: #Step 2 : Visualize the Data
[23]: import seaborn as sns
sns.set_style('whitegrid')
df_AAPL.plot()
plt.title('Time Series Plot for NASDAQ_AAPL')
plt.show()
Time Series Plot for NASDAQ_AAPL
NASDAQ.AAPL
165
160
155
150
145
140
1967-11
1968-05
1968-11
1969-05
1969-11
1970-05
1970-11
1971-05
1971-11
Month
Type here to search
ENG
23:36
28-11-2018

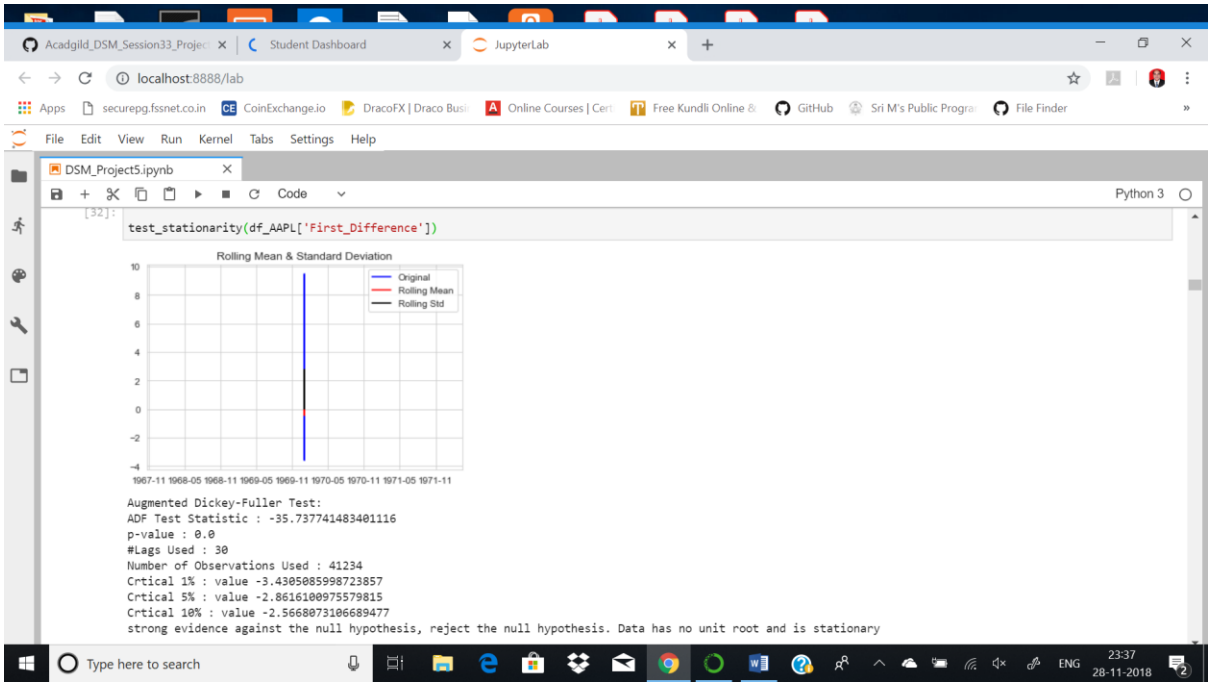

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
Apps
securepg.fssnet.co.in
CoinExchange.io
DracoFX | Draco Busi
Online Courses | Cert
Free Kundli Online &
GitHub
Sri M's Public Progra
File Finder
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
1967-11
1968-05
1968-11
1969-05
1969-11
1970-05
1970-11
1971-05
1971-11
Month
[24]: #Plotting Rolling Statistics and check for stationarity :
#The function will plot the moving mean or moving Standard Deviation.
#This is still visual method
#NOTE: moving mean and moving standard deviation – At any instant 't',
#we take the mean/std of the last year which in this case is 12 months)
[25]: from statsmodels.tsa.stattools import adfuller
def test_stationarity(timeseries):
#Determing rolling statistics
rolmean = timeseries.rolling(12).mean()
rolstd = timeseries.rolling(12).std()
#Plot rolling statistics:
plt.plot(timeseries, color='blue',label='Original')
plt.plot(rolmean, color='red', label='Rolling Mean')
plt.plot(rolstd, color='black', label = 'Rolling Std')
plt.legend(loc='best')
plt.title('Rolling Mean & Standard Deviation')
plt.show()
"""
Pass in a time series, returns ADF report
"""
result = adfuller(timeseries)
Type here to search
ENG
23:36
28-11-2018

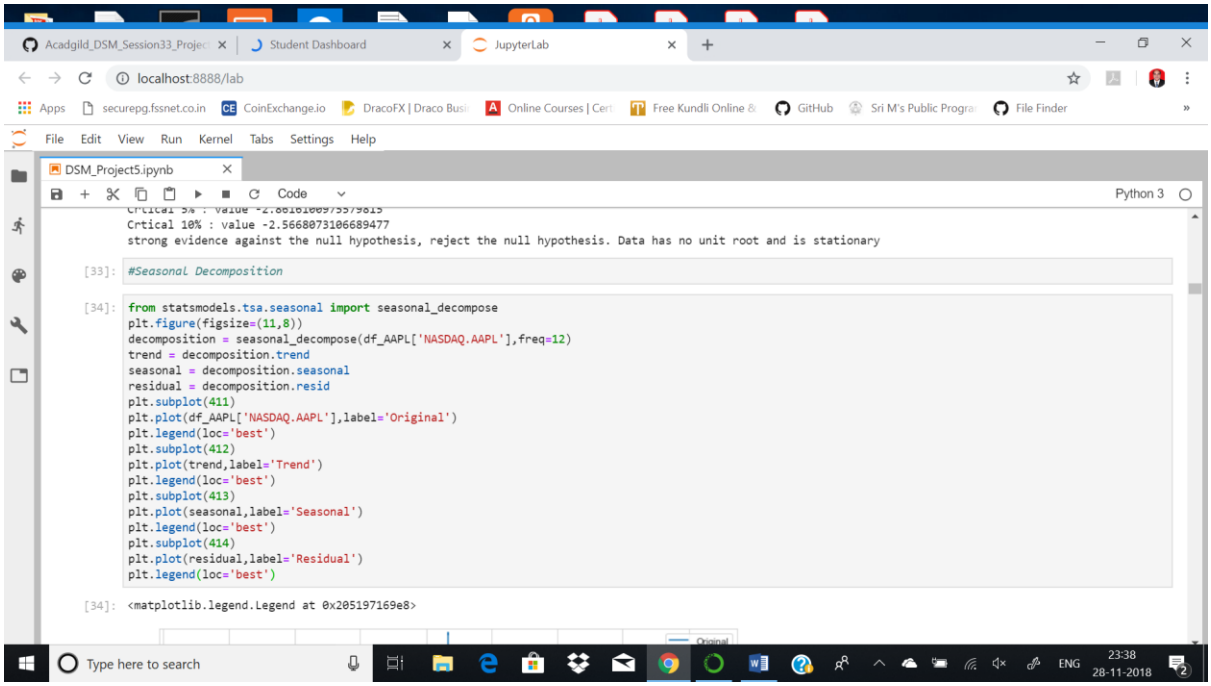

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
Pass in a time series, returns ADF report
"""
result = adfuller(timeseries)
print('\nAugmented Dickey-Fuller Test:')
labels = ['ADF Test Statistic','p-value','#Lags Used','Number of Observations Used']
for value,label in zip(result,labels):
print(label+' : '+str(value))
for k,v in result[4].items():
print('Crtical {} : value {}'.format(k,v))
if result[1] <= 0.05:
print("strong evidence against the null hypothesis, reject the null hypothesis. Data has no unit root and is stationary")
else:
print("weak evidence against null hypothesis, time series has a unit root, indicating it is non-stationary ")
[26]:
test_stationarity(df_AAPL['NASDAQ.AAPL'])
Rolling Mean & Standard Deviation
Original
Rolling Mean
Rolling Std
150
125
100
75
Type here to search
ENG
23:36
28-11-2018

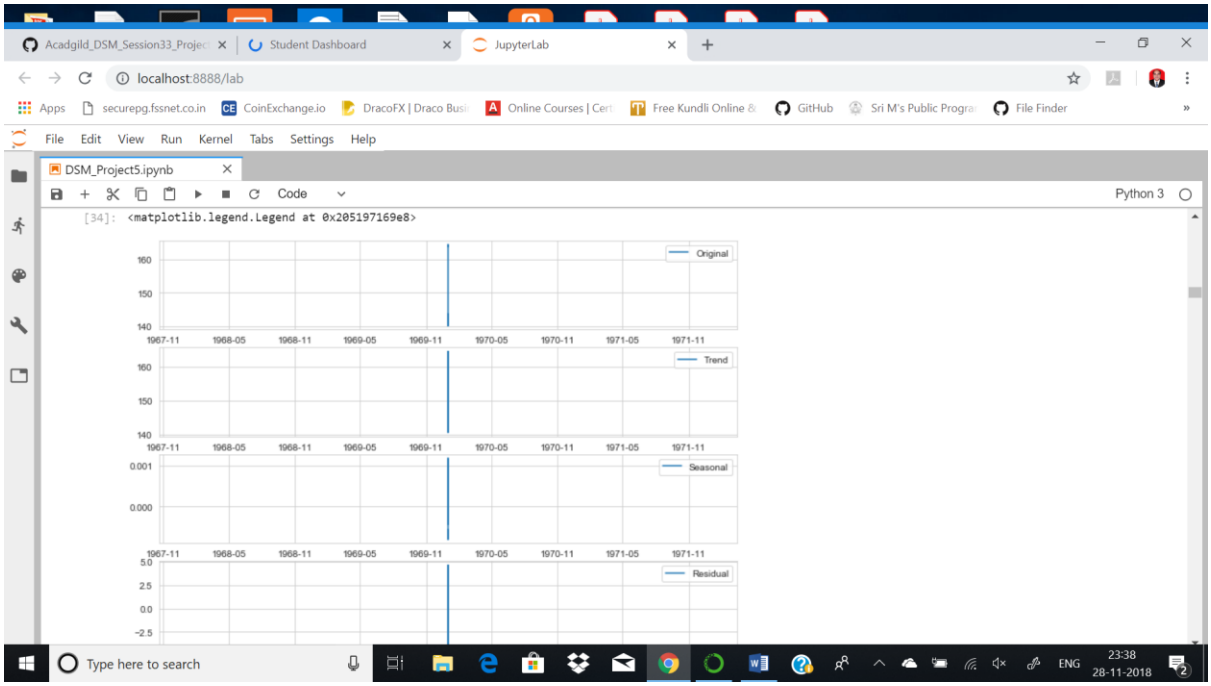

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
100
75
50
25
0
1967-11 1968-05 1968-11 1969-05 1969-11 1970-05 1970-11 1971-05 1971-11
Augmented Dickey-Fuller Test:
ADF Test Statistic : -0.9128532997926723
p-value : 0.7837101772613848
#Lags Used : 31
Number of Observations Used : 41234
Crtical 1% : value -3.4305085998723857
Crtical 5% : value -2.8616100975579815
Crtical 10% : value -2.5668073106689477
weak evidence against null hypothesis, time series has a unit root, indicating it is non-stationary
[27]:
"""Note:
This is not stationary because :
• mean is increasing even though the std is small.
• Test stat is > critical value.
• Note: the signed values are compared and the absolute values.
Type here to search
ENG
23:37
28-11-2018

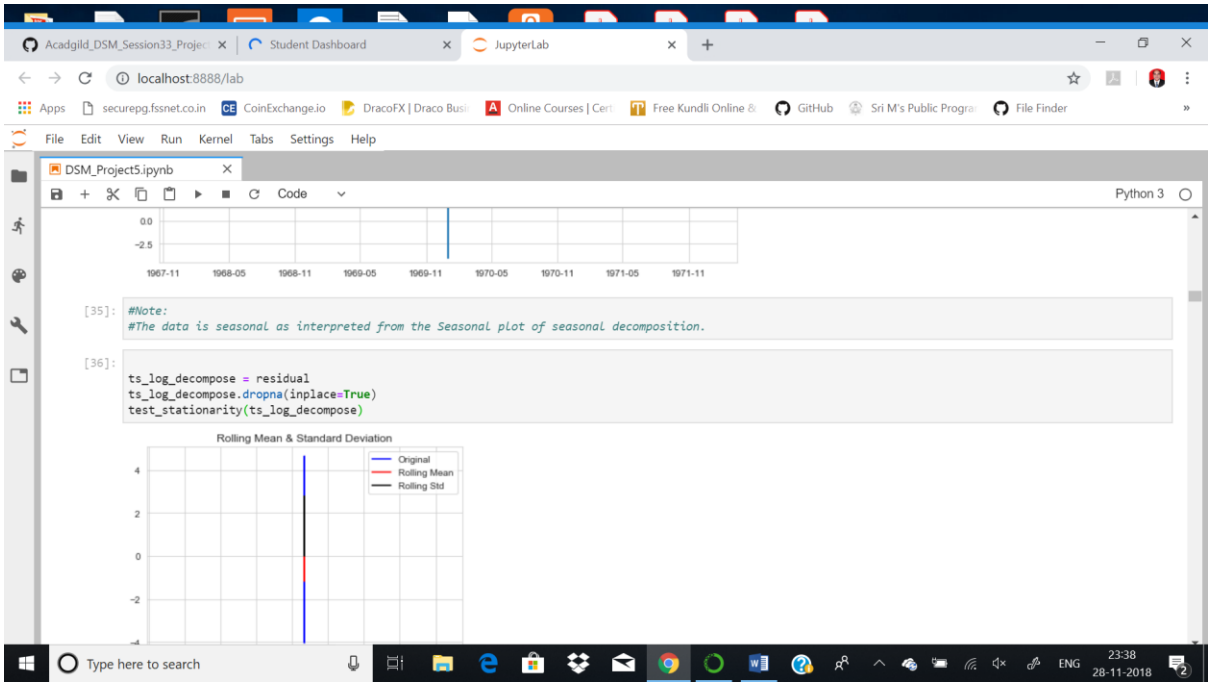

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
• Test stat is > critical value.
• Note: the signed values are compared and the absolute values.
MAKING THE TIME SERIES STATIONARY
There are two major factors that make a time series non-stationary. They are:
• Trend: non-constant mean
• Seasonality: Variation at specific time-frames
Differencing
The first difference of a time series is the series of changes from one period to the next.
We can do this easily with pandas. You can continue to take the second difference,
third difference, and so on until your data is stationary.
First Difference"""
[27]: 'Note:\nThis is not stationary because :\n\n• mean is increasing even though the std is small.\n\n• Test stat is > critical value.\n\n• Note: the signed values are compared and the absolute values.\n\nMAKING THE TIME SERIES STATIONARY\nThere are two major factors that make a time series non-stationary. They are:\n\n• Trend: non-constant mean\n\n• Seasonality: Variation at specific time-frames\n\nDifferencing\nThe first difference of a time series is the series of changes from one period to the next. We can do this easily with pandas. You can continue to take the second difference, third difference, and so on until your data is stationary.\n\nFirst Difference'
[28]: df_AAPL = df_AAPL.copy()
df_AAPL.loc[:,'First_Difference'] = df_AAPL['NASDAQ.AAPL'] - df_AAPL['NASDAQ.AAPL'].shift(1)
Type here to search
ENG
23:37
28-11-2018

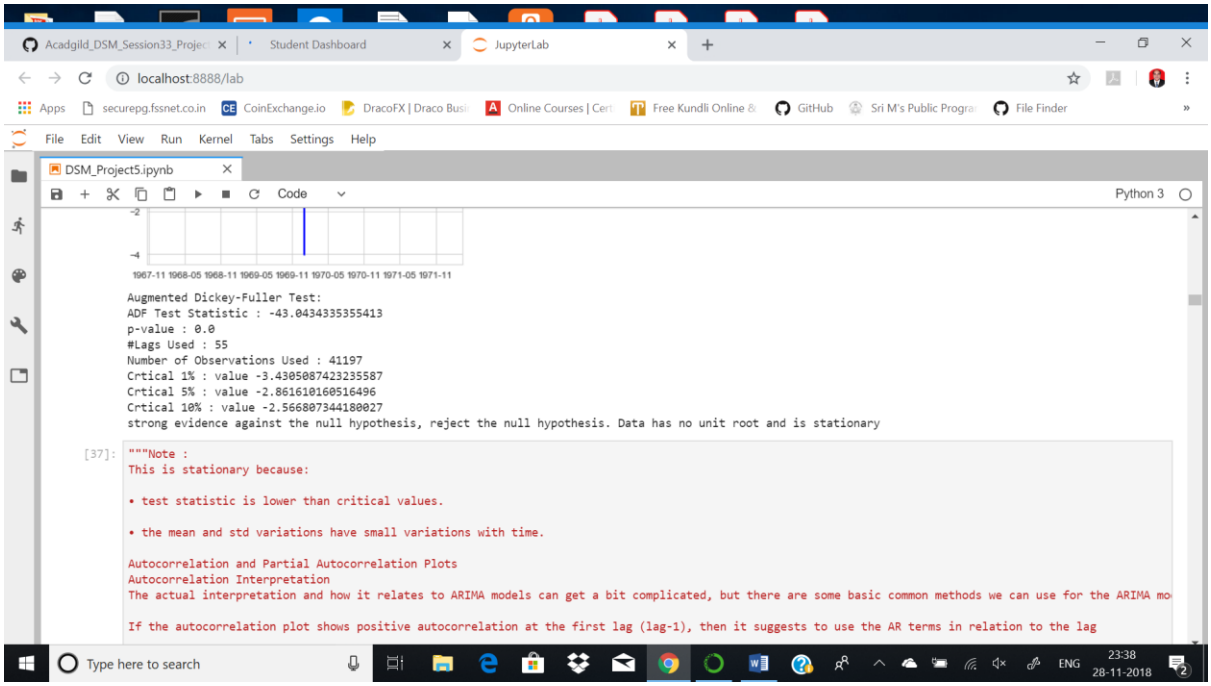

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
df_AAPL.loc[:,'First_Difference'] = df_AAPL['NASDAQ.AAPL'] - df_AAPL['NASDAQ.AAPL'].shift(1)
[29]: df_AAPL.head()
[29]:
NASDAQ.AAPL First_Difference
Month
1970-01-01 143.6800 NaN
1970-01-01 143.7000 0.0200
1970-01-01 143.6901 -0.0099
1970-01-01 143.6400 -0.0501
1970-01-01 143.6600 0.0200
[30]: df_AAPL = df_AAPL.copy()
df_AAPL.dropna(inplace=True)
[31]: #Test Staionarity
[32]: test_stationarity(df_AAPL['First_Difference'])
Type here to search
ENG
23:37
28-11-2018

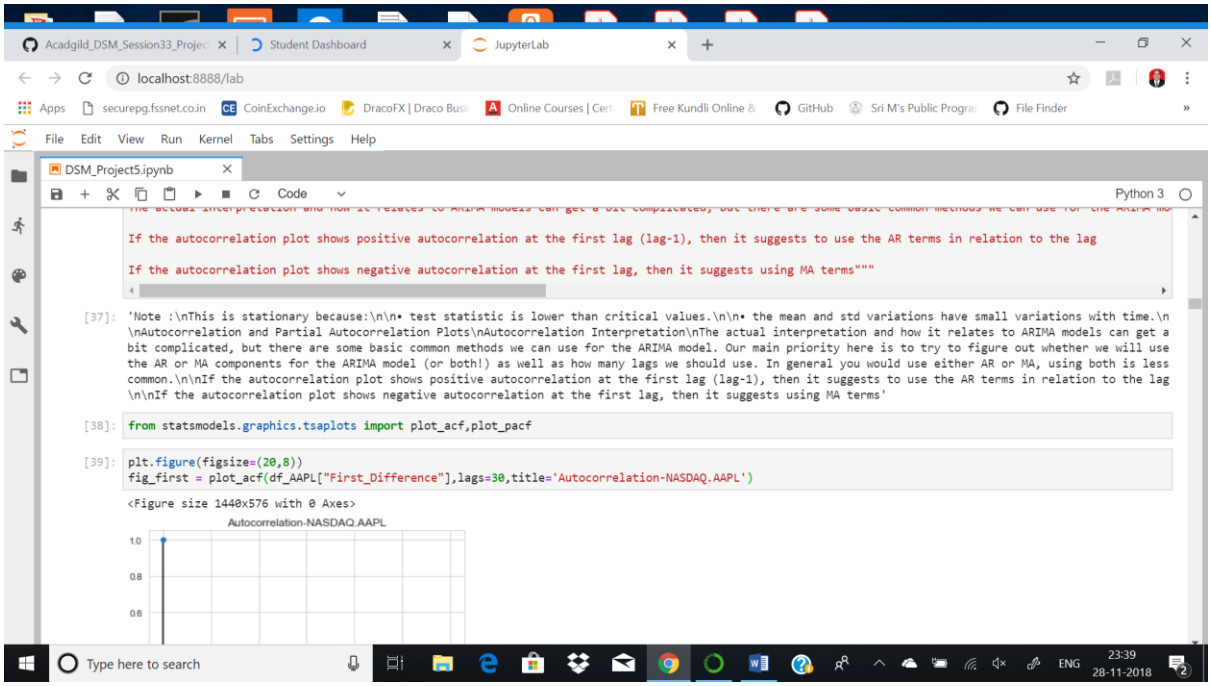

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
Apps
securepg.fssnet.co.in
CoinExchange.io
DracoFX | Draco Busi
Online Courses | Cert
Free Kundli Online &
GitHub
Sri M's Public Progra
File Finder
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
[32]:
test_stationarity(df_AAPL['First_Difference'])
Rolling Mean & Standard Deviation
Original
Rolling Mean
Rolling Std
10
8
6
4
2
0
-2
-4
1967-11 1968-05 1968-11 1969-05 1969-11 1970-05 1970-11 1971-05 1971-11
Augmented Dickey-Fuller Test:
ADF Test Statistic : -35.737741483401116
p-value : 0.0
#Lags Used : 30
Number of Observations Used : 41234
Crtical 1% : value -3.4305085998723857
Crtical 5% : value -2.8616100975579815
Crtical 10% : value -2.5668073106689477
strong evidence against the null hypothesis, reject the null hypothesis. Data has no unit root and is stationary
Type here to search
ENG
23:37
28-11-2018

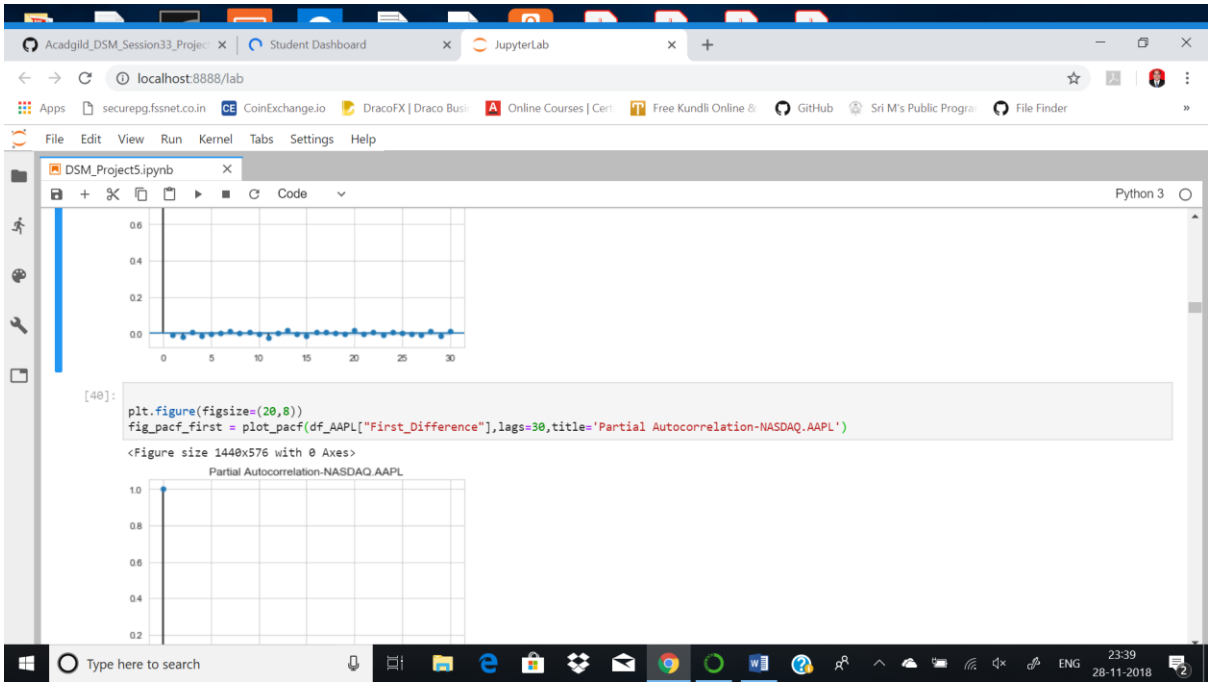

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
Apps
securepg.fssnet.co.in
CoinExchange.io
DracoFX | Draco Busi
Online Courses | Cert
Free Kundli Online &
GitHub
Sri M's Public Progra
File Finder
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
Crtical 5% : value -2.8616100975579815
Crtical 10% : value -2.5668073106689477
strong evidence against the null hypothesis, reject the null hypothesis. Data has no unit root and is stationary
[33]:
#Seasonal Decomposition
[34]:
from statsmodels.tsa.seasonal import seasonal_decompose
plt.figure(figsize=(11,8))
decomposition = seasonal_decompose(df_AAPL['NASDAQ.AAPL'],freq=12)
trend = decomposition.trend
seasonal = decomposition.seasonal
residual = decomposition.resid
plt.subplot(411)
plt.plot(df_AAPL['NASDAQ.AAPL'],label='Original')
plt.legend(loc='best')
plt.subplot(412)
plt.plot(trend,label='Trend')
plt.legend(loc='best')
plt.subplot(413)
plt.plot(seasonal,label='Seasonal')
plt.legend(loc='best')
plt.subplot(414)
plt.plot(residual,label='Residual')
plt.legend(loc='best')
[34]:
<matplotlib.legend.Legend at 0x205197169e8>
Type here to search
ENG
23:38
28-11-2018

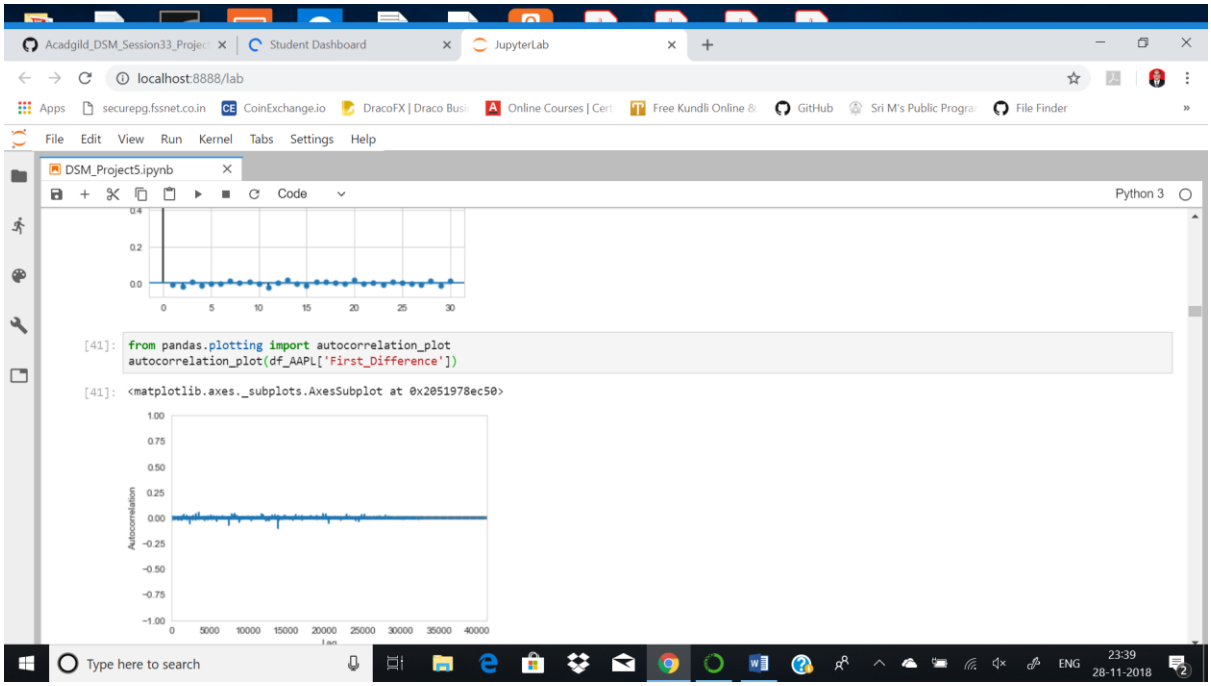

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
[34]: <matplotlib.legend.Legend at 0x205197169e8>
Original
Trend
Seasonal
Residual
Type here to search
ENG
23:38
28-11-2018

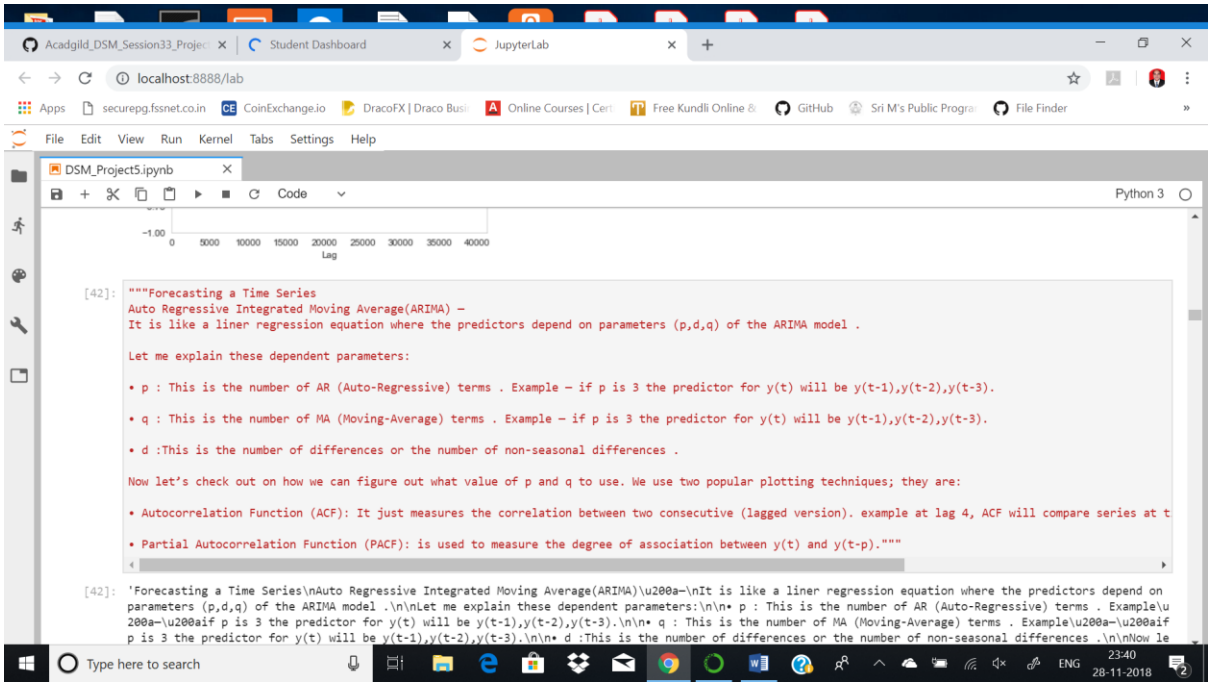

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
[35]: #Note:
#The data is seasonal as interpreted from the Seasonal plot of seasonal decomposition.
[36]: ts_log_decompose = residual
ts_log_decompose.dropna(inplace=True)
test_stationarity(ts_log_decompose)
Rolling Mean & Standard Deviation
Original
Rolling Mean
Rolling Std
Type here to search
ENG
23:38
28-11-2018

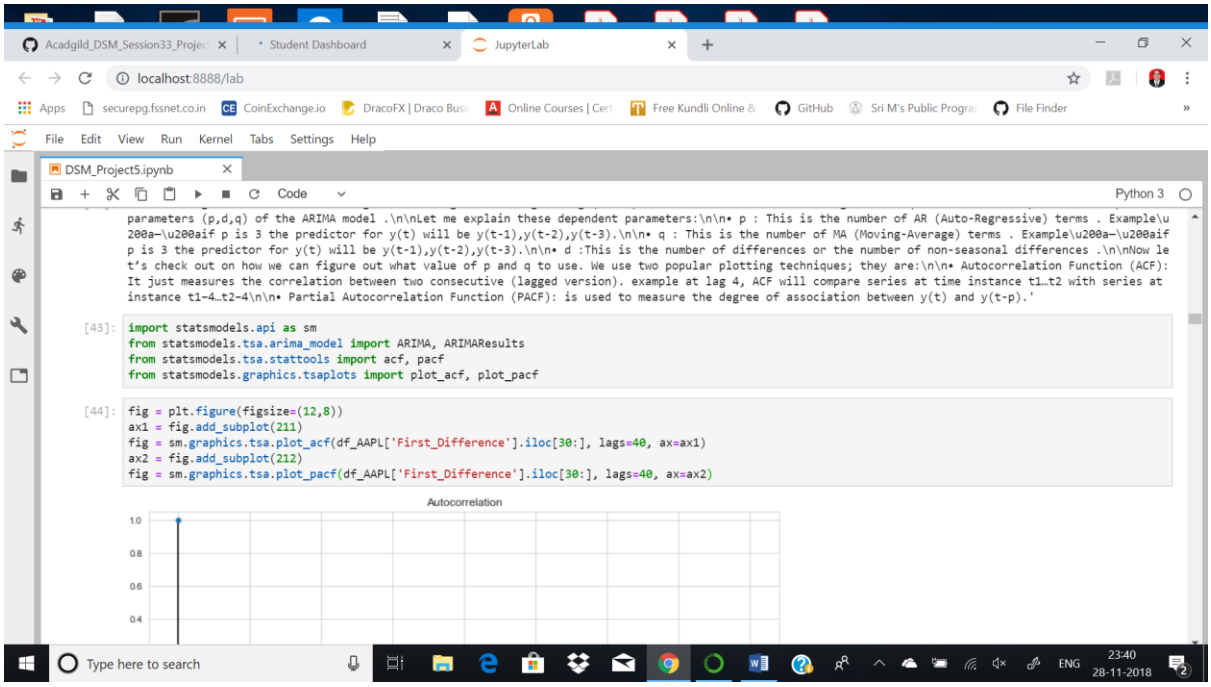

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
Apps
securepg.fssnet.co.in
CoinExchange.io
DracoFX | Draco Busi
Online Courses | Cert
Free Kundli Online &
GitHub
Sri M's Public Progra
File Finder
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
1967-11 1968-05 1968-11 1969-05 1969-11 1970-05 1970-11 1971-05 1971-11
Augmented Dickey-Fuller Test:
ADF Test Statistic : -43.04343355413
p-value : 0.0
#Lags Used : 55
Number of Observations Used : 41197
Crtical 1% : value -3.430508742335587
Crtical 5% : value -2.861610160516496
Crtical 10% : value -2.566807344180027
strong evidence against the null hypothesis, reject the null hypothesis. Data has no unit root and is stationary
[37]:
"""Note :
This is stationary because:
• test statistic is lower than critical values.
• the mean and std variations have small variations with time.
Autocorrelation and Partial Autocorrelation Plots
Autocorrelation Interpretation
The actual interpretation and how it relates to ARIMA models can get a bit complicated, but there are some basic common methods we can use for the ARIMA mo
If the autocorrelation plot shows positive autocorrelation at the first lag (lag-1), then it suggests to use the AR terms in relation to the lag
Type here to search
ENG
23:38
28-11-2018

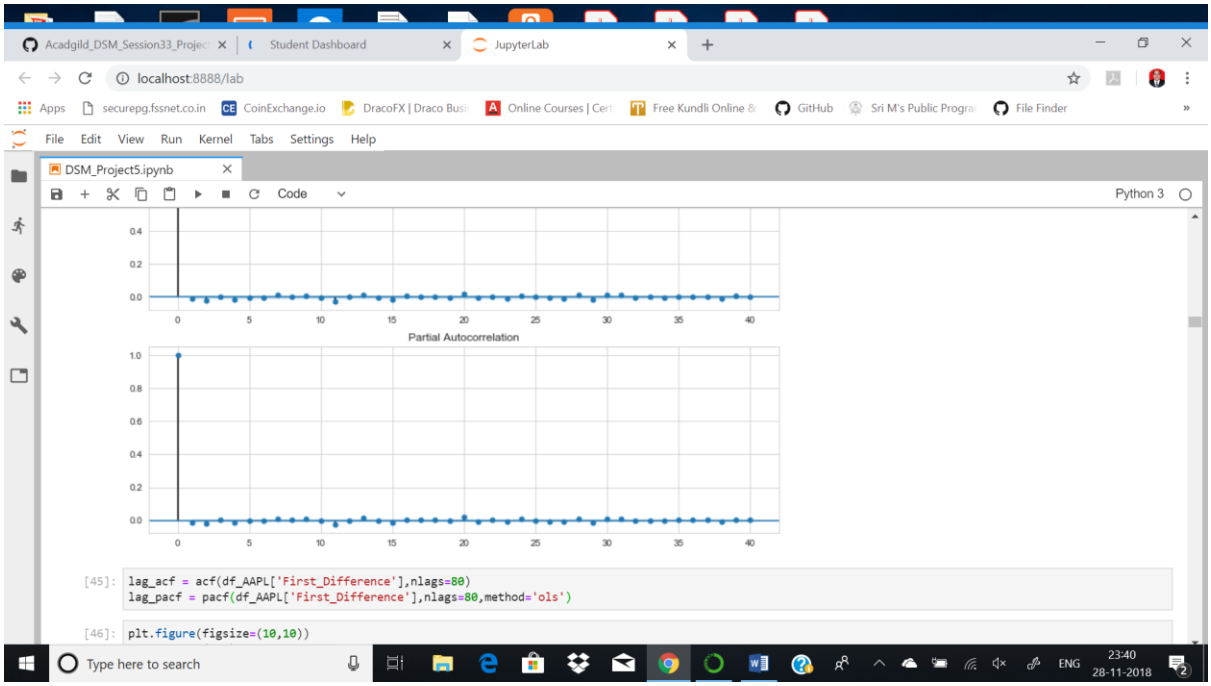

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
Apps
securepg.fssnet.co.in
CoinExchange.io
DracoFX | Draco Busi
Online Courses | Cert
Free Kundli Online &
GitHub
Sri M's Public Progra
File Finder
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
If the autocorrelation plot shows positive autocorrelation at the first lag (lag-1), then it suggests to use the AR terms in relation to the lag
If the autocorrelation plot shows negative autocorrelation at the first lag, then it suggests using MA terms"""
[37]:
'Note :\nThis is stationary because:\n\n• test statistic is lower than critical values.\n\n• the mean and std variations have small variations with time.\n
\nAutocorrelation and Partial Autocorrelation Plots\nAutocorrelation Interpretation\nThe actual interpretation and how it relates to ARIMA models can get a
bit complicated, but there are some basic common methods we can use for the ARIMA model. Our main priority here is to try to figure out whether we will use
the AR or MA components for the ARIMA model (or both!) as well as how many lags we should use. In general you would use either AR or MA, using both is less
common.\n\nIf the autocorrelation plot shows positive autocorrelation at the first lag (lag-1), then it suggests to use the AR terms in relation to the lag
\n\nIf the autocorrelation plot shows negative autocorrelation at the first lag, then it suggests using MA terms'
[38]:
from statsmodels.graphics.tsaplots import plot_acf,plot_pacf
[39]:
plt.figure(figsize=(20,8))
fig_first = plot_acf(df_AAPL["First_Difference"],lags=30,title='Autocorrelation-NASDAQ.AAPL')
<Figure size 1440x576 with 0 Axes>
Autocorrelation-NASDAQ.AAPL
1.0
0.8
0.6
Type here to search
ENG
23:39
28-11-2018

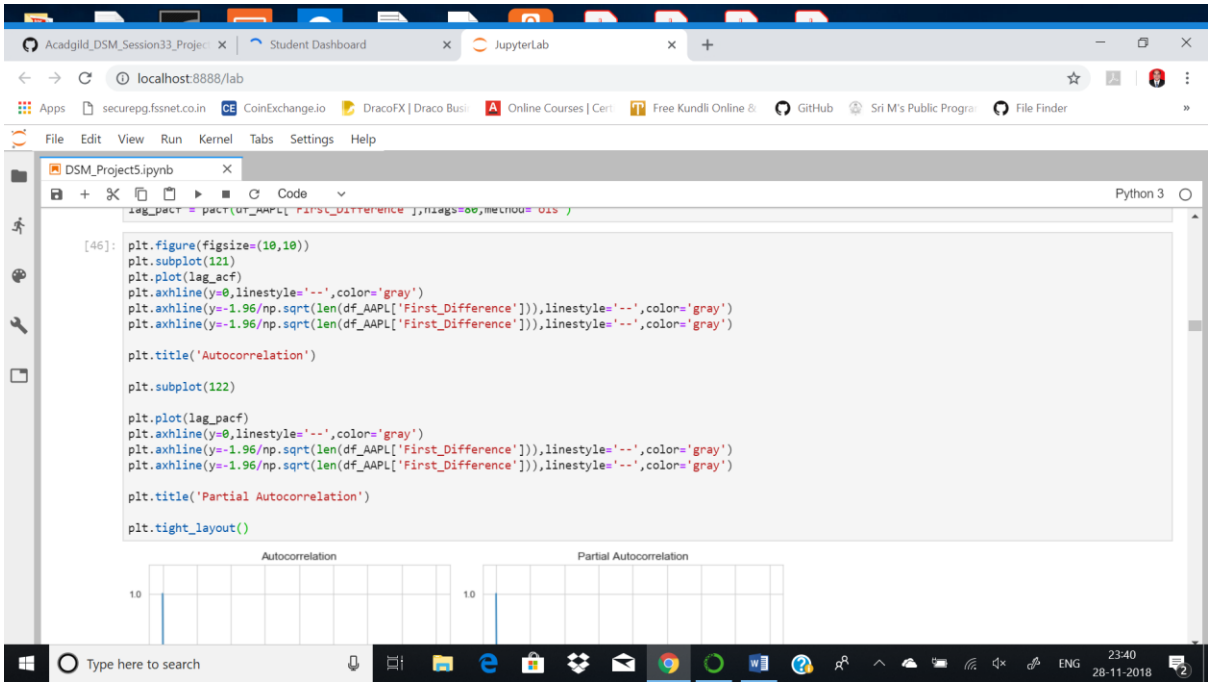

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
[40]:
plt.figure(figsize=(20,8))
fig_pacf_first = plot_pacf(df_AAPL["First_Difference"],lags=30,title='Partial Autocorrelation-NASDAQ.AAPL')
<Figure size 1440x576 with 0 Axes>
Partial Autocorrelation-NASDAQ.AAPL
Type here to search
ENG
23:39
28-11-2018

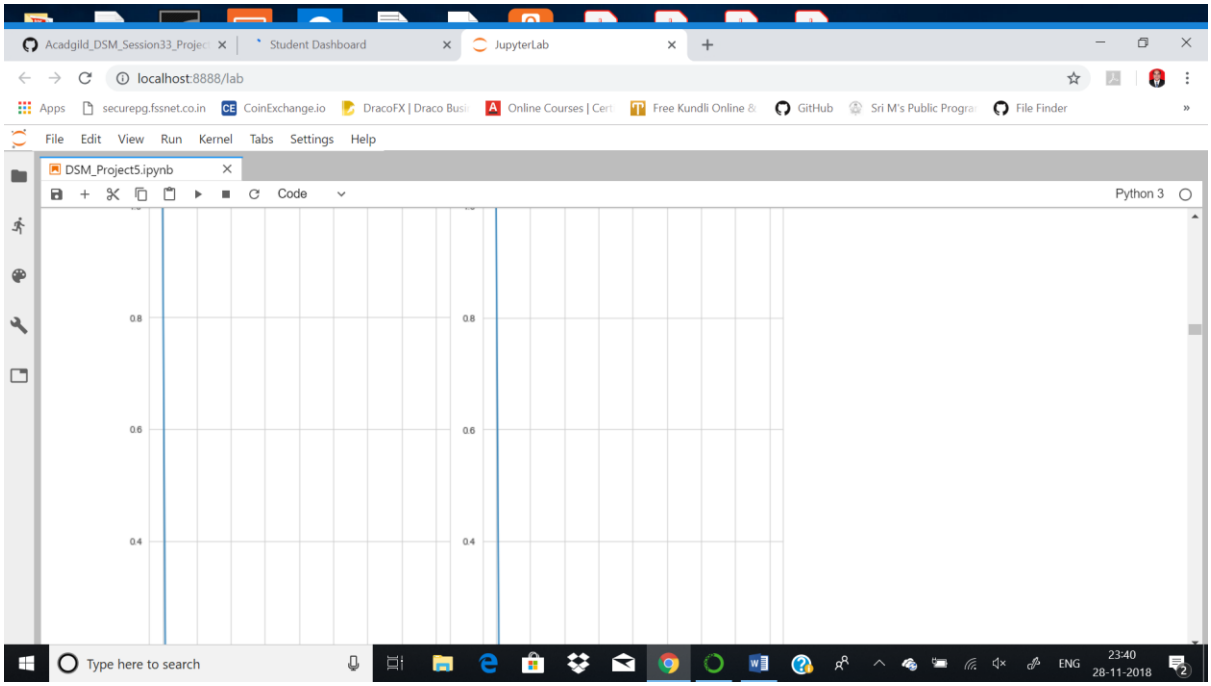

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
[41]:
from pandas.plotting import autocorrelation_plot
autocorrelation_plot(df_AAPL['First_Difference'])
[41]:
<matplotlib.axes._subplots.AxesSubplot at 0x2051978ec50>
Autocorrelation
Type here to search
ENG
23:39
28-11-2018

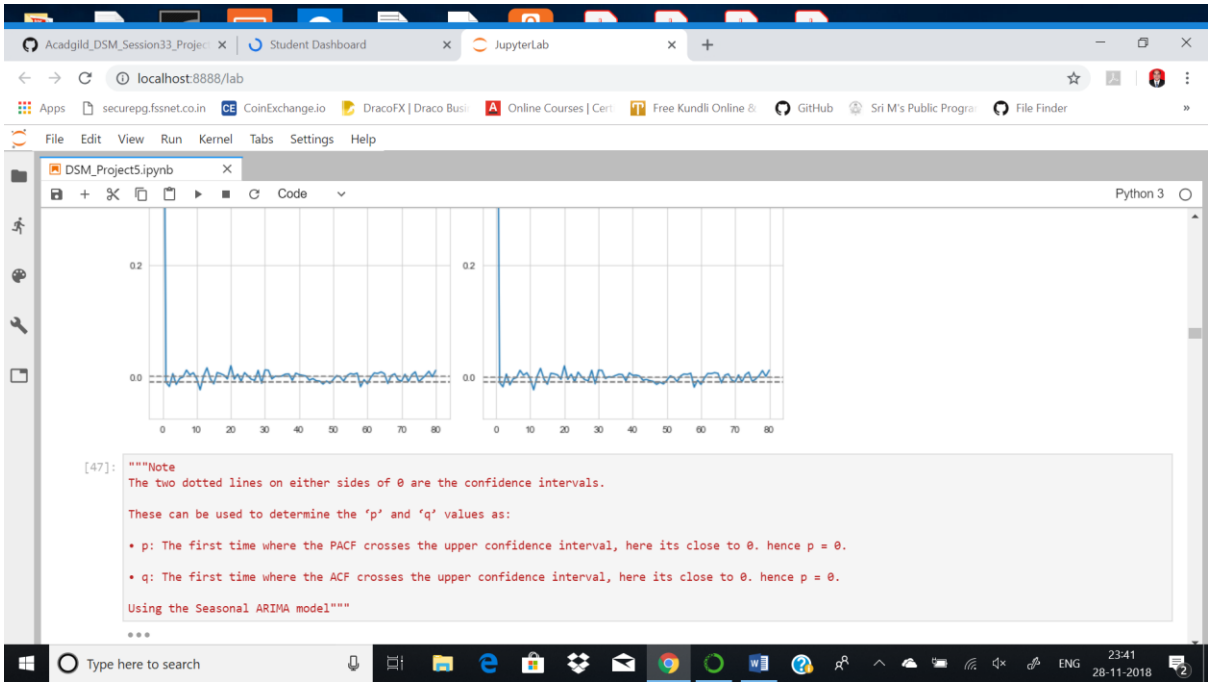

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
Lag
[42]:
"""Forecasting a Time Series
Auto Regressive Integrated Moving Average(ARIMA) –
It is like a liner regression equation where the predictors depend on parameters (p,d,q) of the ARIMA model .
Let me explain these dependent parameters:
• p : This is the number of AR (Auto-Regressive) terms . Example – if p is 3 the predictor for y(t) will be y(t-1),y(t-2),y(t-3).
• q : This is the number of MA (Moving-Average) terms . Example – if p is 3 the predictor for y(t) will be y(t-1),y(t-2),y(t-3).
• d :This is the number of differences or the number of non-seasonal differences .
Now let's check out on how we can figure out what value of p and q to use. We use two popular plotting techniques; they are:
• Autocorrelation Function (ACF): It just measures the correlation between two consecutive (lagged version). example at lag 4, ACF will compare series at t
• Partial Autocorrelation Function (PACF): is used to measure the degree of association between y(t) and y(t-p)."""
[42]:
'Forecasting a Time Series\nAuto Regressive Integrated Moving Average(ARIMA)\u200a–\nIt is like a liner regression equation where the predictors depend on parameters (p,d,q) of the ARIMA model .\n\nLet me explain these dependent parameters:\n\n• p : This is the number of AR (Auto-Regressive) terms . Example\u200a–\u200aif p is 3 the predictor for y(t) will be y(t-1),y(t-2),y(t-3).\n\n• q : This is the number of MA (Moving-Average) terms . Example\u200a–\u200aif p is 3 the predictor for y(t) will be y(t-1),y(t-2),y(t-3).\n\n• d :This is the number of differences or the number of non-seasonal differences .\n\nNow le
Type here to search
ENG
23:40
28-11-2018


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
parameters (p,d,q) of the ARIMA model .\n\nLet me explain these dependent parameters:\n\n• p : This is the number of AR (Auto-Regressive) terms . Example\u200a–\u200aif p is 3 the predictor for y(t) will be y(t-1),y(t-2),y(t-3).\n\n• q : This is the number of MA (Moving-Average) terms . Example\u200a–\u200aif p is 3 the predictor for y(t) will be y(t-1),y(t-2),y(t-3).\n\n• d :This is the number of differences or the number of non-seasonal differences .\n\nNow let's check out on how we can figure out what value of p and q to use. We use two popular plotting techniques; they are:\n\n• Autocorrelation Function (ACF): It just measures the correlation between two consecutive (lagged version). example at lag 4, ACF will compare series at time instance t1…t2 with series at instance t1-4…t2-4\n\n• Partial Autocorrelation Function (PACF): is used to measure the degree of association between y(t) and y(t-p).'
[43]:
import statsmodels.api as sm
from statsmodels.tsa.arima_model import ARIMA, ARIMAResults
from statsmodels.tsa.stattools import acf, pacf
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
[44]:
fig = plt.figure(figsize=(12,8))
ax1 = fig.add_subplot(211)
fig = sm.graphics.tsa.plot_acf(df_AAPL['First_Difference'].iloc[30:], lags=40, ax=ax1)
ax2 = fig.add_subplot(212)
fig = sm.graphics.tsa.plot_pacf(df_AAPL['First_Difference'].iloc[30:], lags=40, ax=ax2)
Autocorrelation
Type here to search
ENG
23:40
28-11-2018


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
Apps
securepg.fssnet.co.in
CoinExchange.io
DracoFX | Draco Busi
Online Courses | Cert
Free Kundli Online &
GitHub
Sri M's Public Progra
File Finder
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
Partial Autocorrelation
[45]:
lag_acf = acf(df_AAPL['First_Difference'],nlags=80)
lag_pacf = pacf(df_AAPL['First_Difference'],nlags=80,method='ols')
[46]:
plt.figure(figsize=(10,10))
Type here to search
ENG
23:40
28-11-2018


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
Apps
securepg.fssnet.co.in
CoinExchange.io
DracoFX | Draco Busi
Online Courses | Cert
Free Kundli Online &
GitHub
Sri M's Public Progra
File Finder
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
[46]:
plt.figure(figsize=(10,10))
plt.subplot(121)
plt.plot(lag_acf)
plt.axhline(y=0,linestyle='--',color='gray')
plt.axhline(y=-1.96/np.sqrt(len(df_AAPL['First_Difference'])),linestyle='--',color='gray')
plt.axhline(y=-1.96/np.sqrt(len(df_AAPL['First_Difference'])),linestyle='--',color='gray')
plt.title('Autocorrelation')
plt.subplot(122)
plt.plot(lag_pacf)
plt.axhline(y=0,linestyle='--',color='gray')
plt.axhline(y=-1.96/np.sqrt(len(df_AAPL['First_Difference'])),linestyle='--',color='gray')
plt.axhline(y=-1.96/np.sqrt(len(df_AAPL['First_Difference'])),linestyle='--',color='gray')
plt.title('Partial Autocorrelation')
plt.tight_layout()
Autocorrelation
Partial Autocorrelation
Type here to search
ENG
23:40
28-11-2018


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
Apps
securepg.fssnet.co.in
CoinExchange.io
DracoFX | Draco Busi
Online Courses | Cert
Free Kundli Online &
GitHub
Sri M's Public Progra
File Finder
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
0.8
0.6
0.4
Type here to search
ENG
23:40
28-11-2018


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
Apps
securepg.fssnet.co.in
CoinExchange.io
DracoFX | Draco Busi
Online Courses | Cert
Free Kundli Online &
GitHub
Sri M's Public Progra
File Finder
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
0.2
0.0
0 10 20 30 40 50 60 70 80
[47]:
"""Note
The two dotted lines on either sides of 0 are the confidence intervals.
These can be used to determine the 'p' and 'q' values as:
• p: The first time where the PACF crosses the upper confidence interval, here its close to 0. hence p = 0.
• q: The first time where the ACF crosses the upper confidence interval, here its close to 0. hence p = 0.
Using the Seasonal ARIMA model"""
Type here to search
ENG
23:41
28-11-2018

Using the Seasonal ARIMA model"""

[48]:

```
model= sm.tsa.statespace.SARIMAX(df_AAPL['NASDAQ.AAPL'],order=(0,1,0),seasonal_order=(0,1,0,12))
results = model.fit()
print(results.summary())
```

```
C:\ProgramData\Anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:225: ValueWarning: A date index has been provided, but it has no associated fr
equency information and so will be ignored when e.g. forecasting.
  ' ignored when e.g. forecasting.', ValueWarning)
                                 Statespace Model Results
==========================================================================================
Dep. Variable:                        NASDAQ.AAPL   No. Observations:                41265
Model:             SARIMAX(0, 1, 0)x(0, 1, 0, 12)   Log Likelihood               24925.552
Date:                            Wed, 28 Nov 2018   AIC                         -49849.104
Time:                                    22:50:24   BIC                         -49840.476
Sample:                                         0   HQIC                        -49846.377
                                            - 41265
Covariance Type:                              opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
sigma2         0.0175   4.57e-06   3828.712      0.000       0.017       0.017
===================================================================================
Ljung-Box (Q):                    10611.64   Jarque-Bera (JB):        3462262308.64
Prob(Q):                              0.00   Prob(JB):                         0.00
```


```
Ljung-Box (Q):                    10611.64   Jarque-Bera (JB):        3462262308.64
Prob(Q):                              0.00   Prob(JB):                         0.00
Heteroskedasticity (H):               2.92   Skew:                            -2.00
Prob(H) (two-sided):                  0.00   Kurtosis:                      1422.26
```

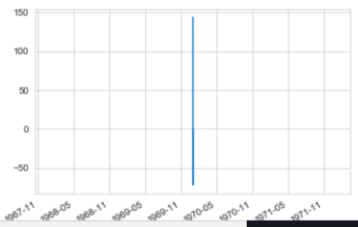

```
Warnings:
[1] Covariance matrix calculated using the outer product of gradients (complex-step).
```

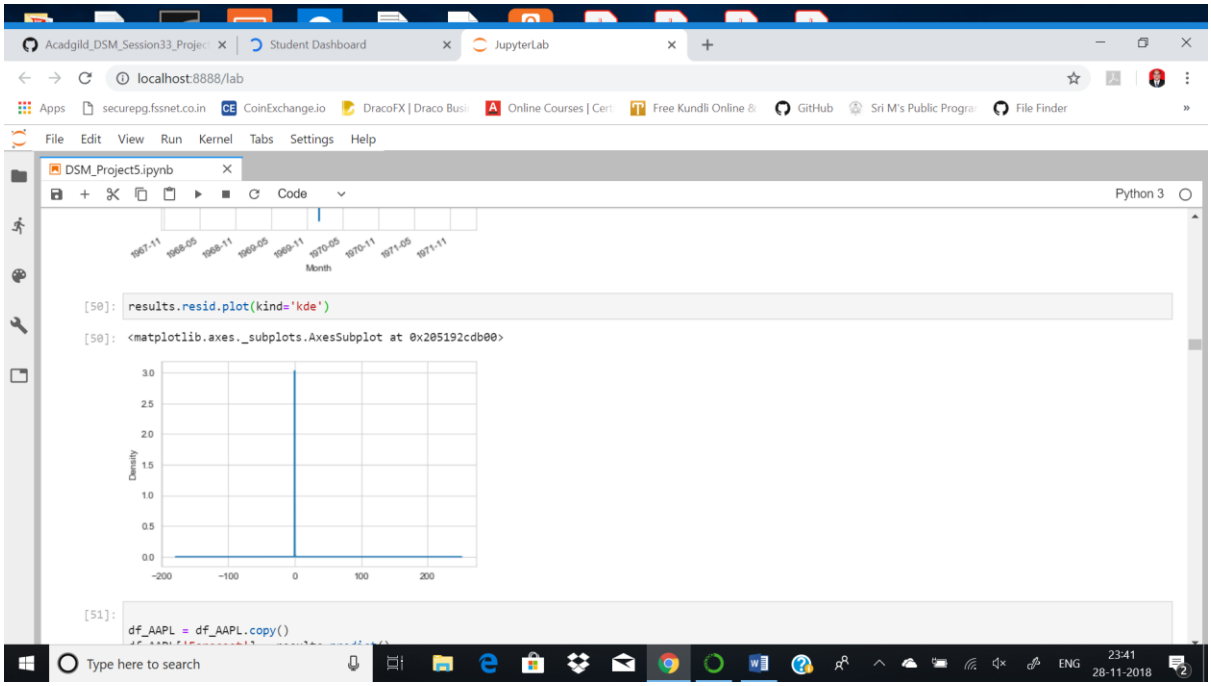
[49]:

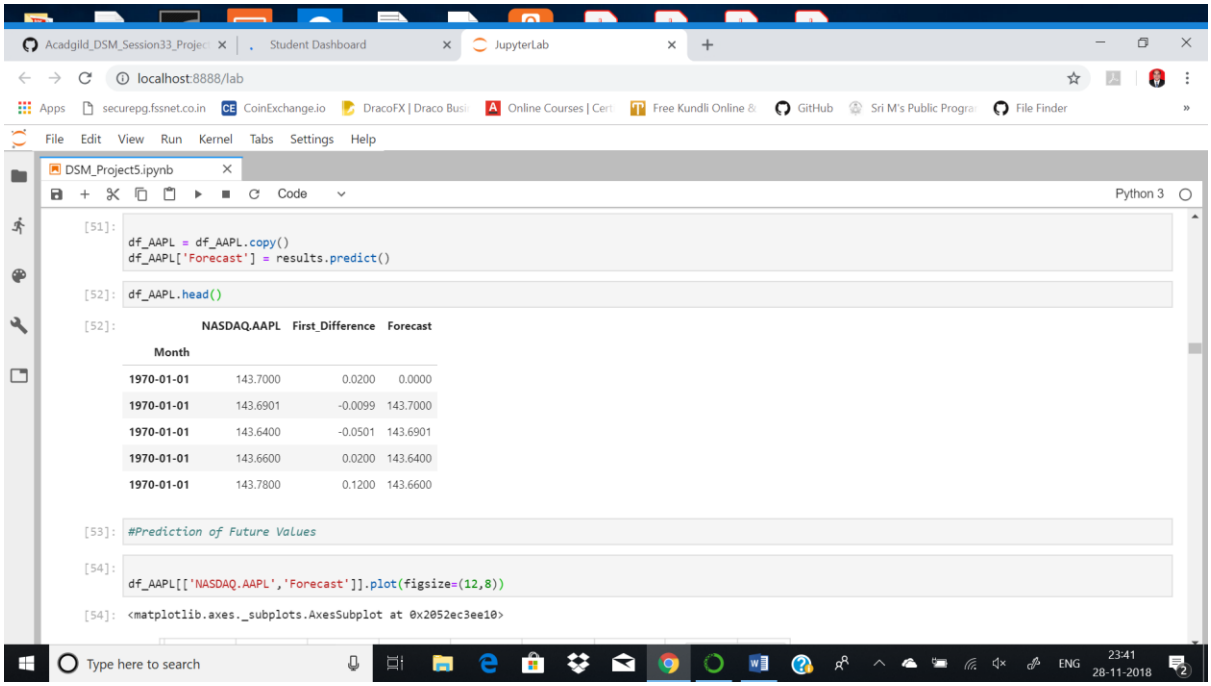
```
results.resid.plot()
```

[49]:

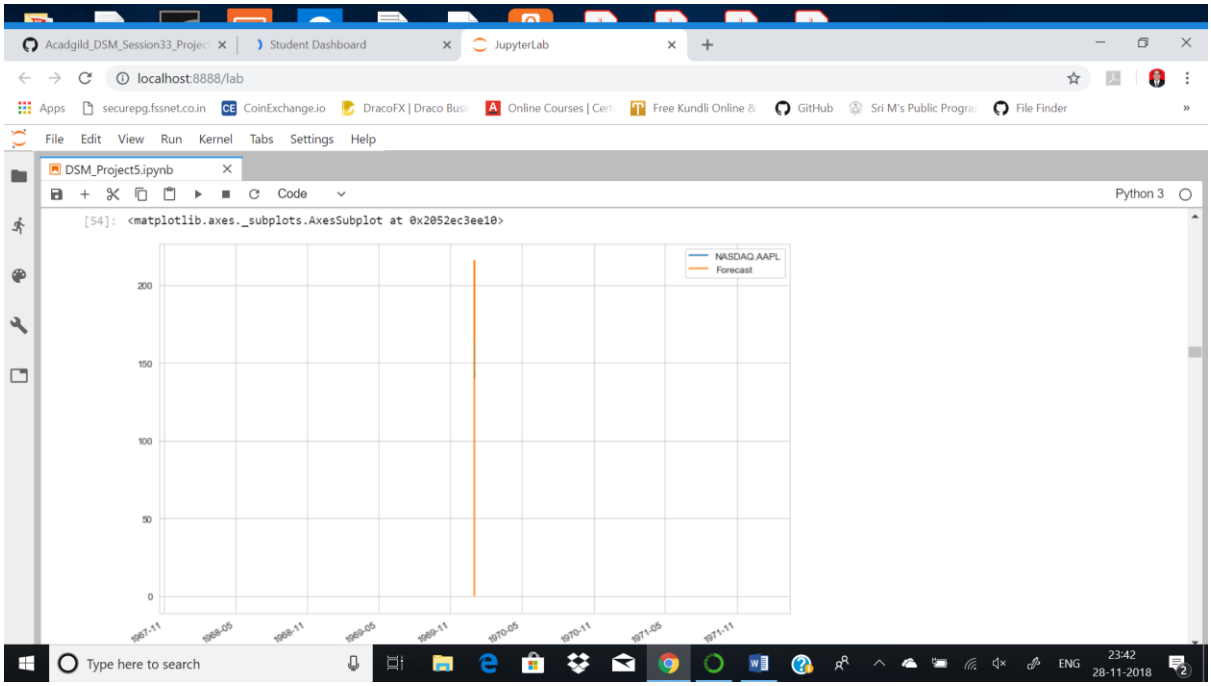
```
<matplotlib.axes._subplots.AxesSubplot at 0x2052ea7fcc0>
```

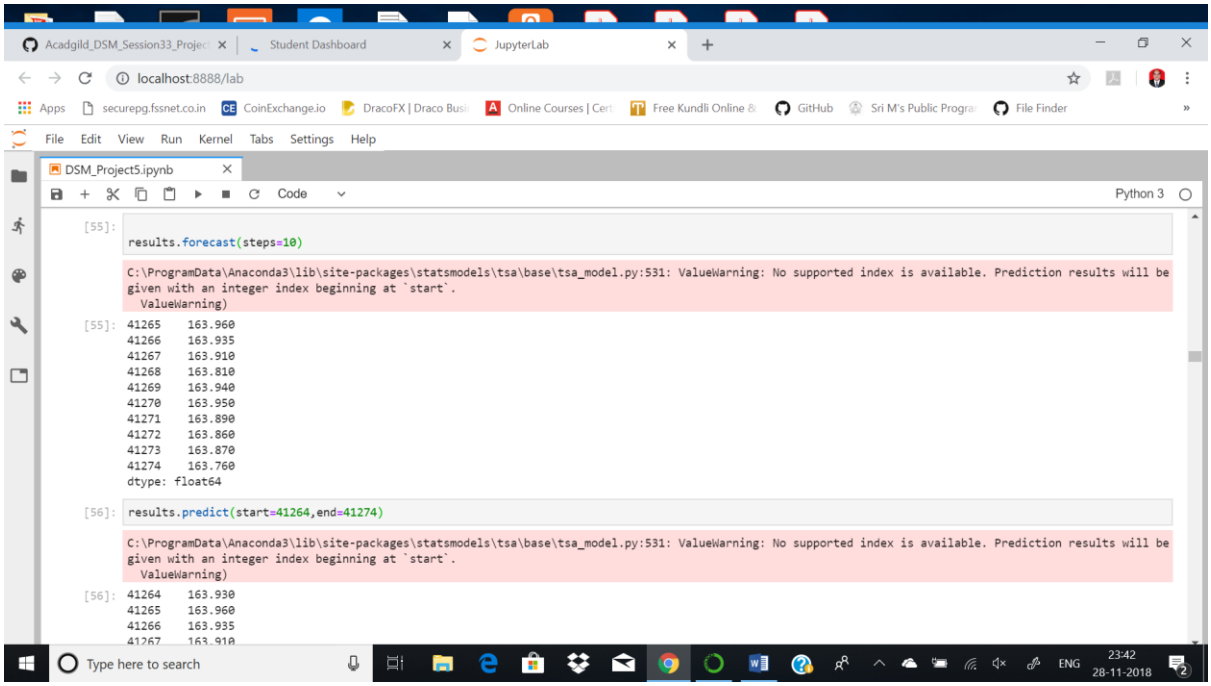


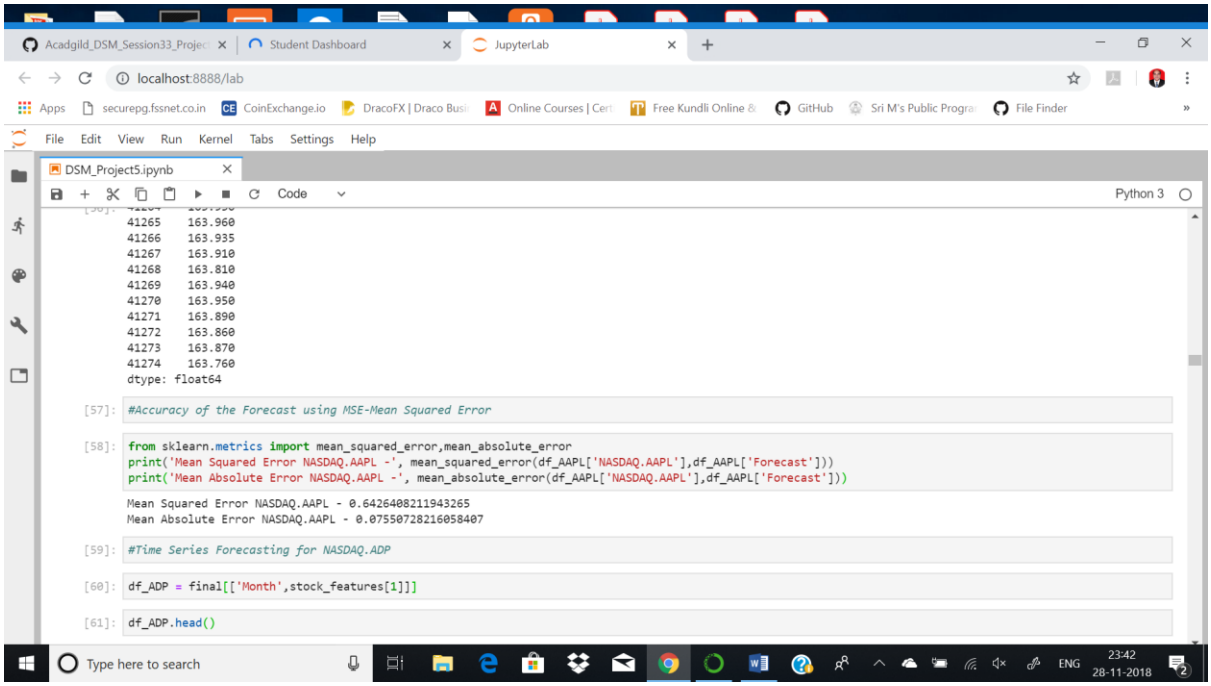


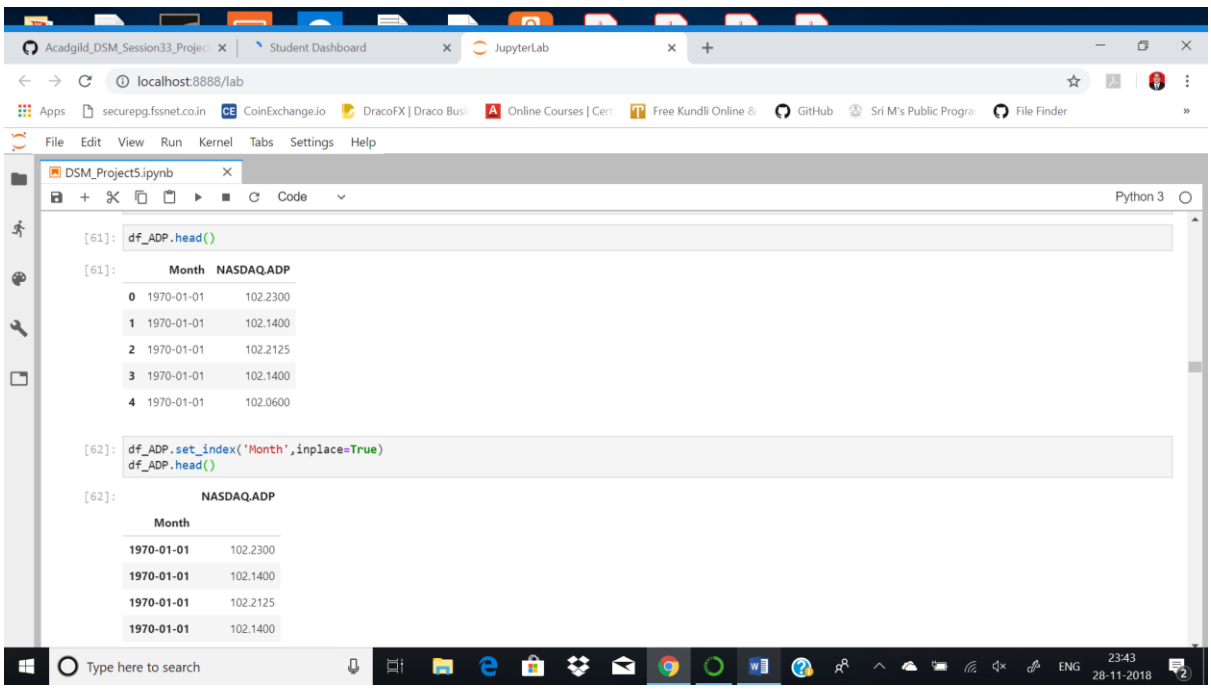


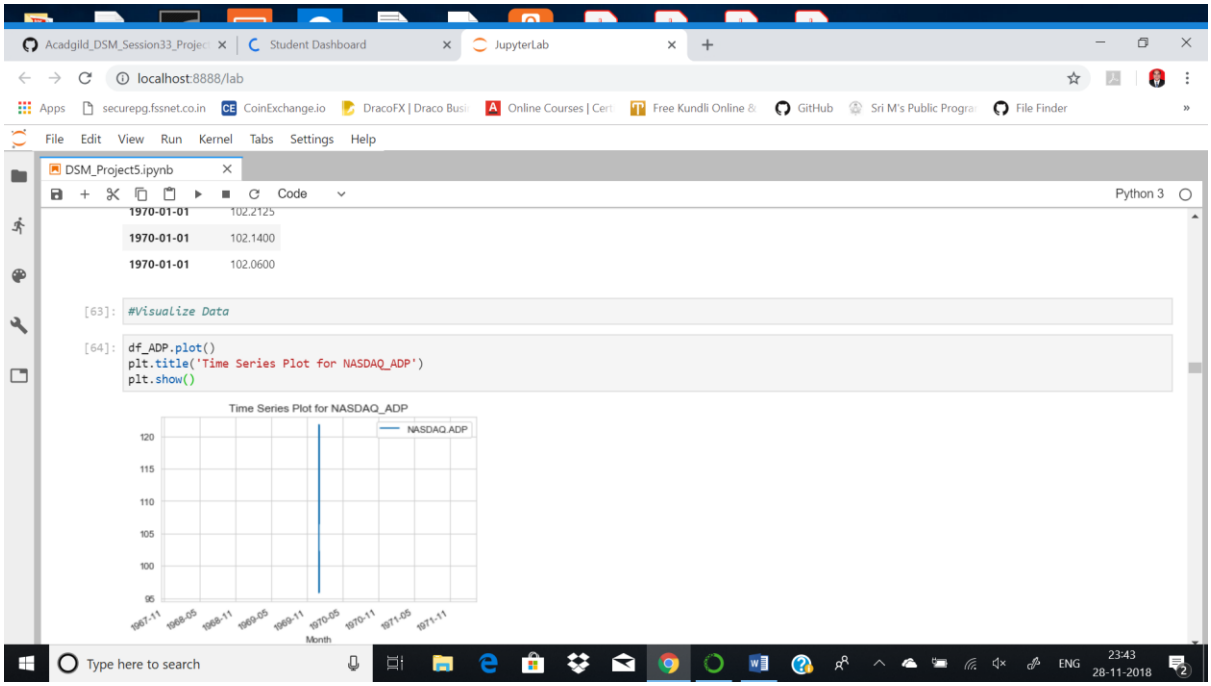
Month	NASDAQ.AAPL	First_Difference	Forecast
1970-01-01	143.7000	0.0200	0.0000
1970-01-01	143.6901	-0.0099	143.7000
1970-01-01	143.6400	-0.0501	143.6901
1970-01-01	143.6600	0.0200	143.6400
1970-01-01	143.7800	0.1200	143.6600

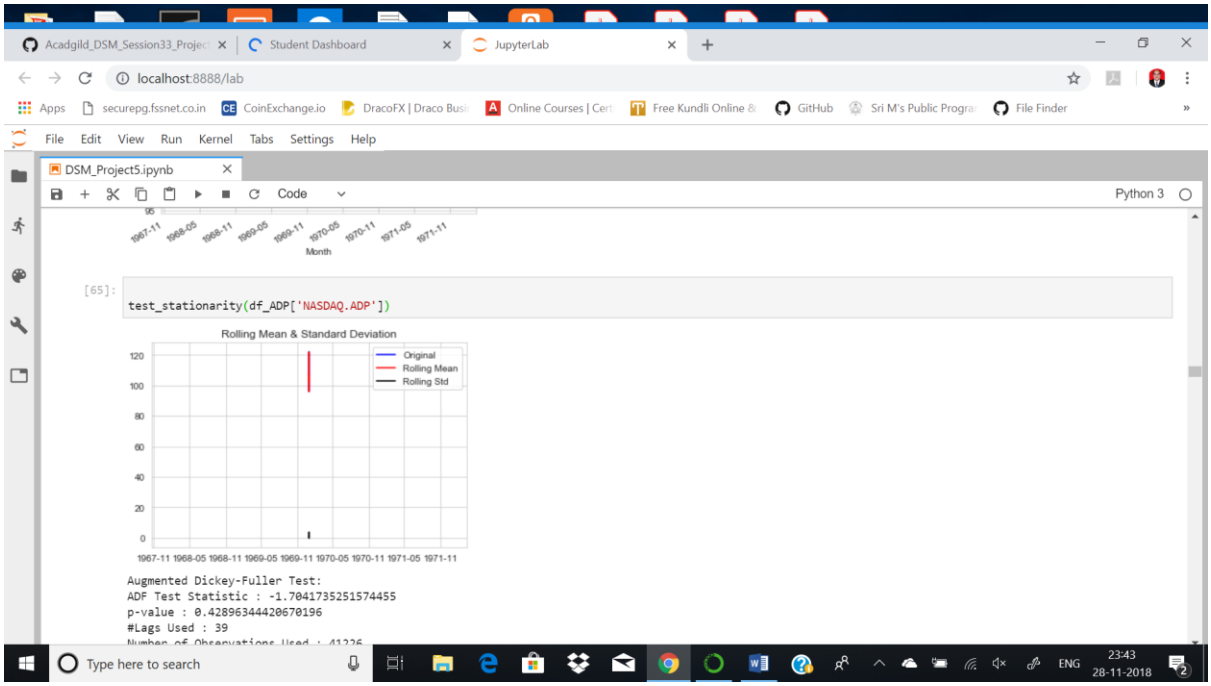

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
Apps
securepg.fssnet.co.in
CoinExchange.io
DracoFX | Draco Busi
Online Courses | Cert
Free Kundli Online &
GitHub
Sri M's Public Progra
File Finder
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
[54]: <matplotlib.axes._subplots.AxesSubplot at 0x2052ec3ee10>
NASDAQ.AAPL
Forecast
200
150
100
50
0
1967-11
1968-05
1968-11
1969-05
1969-11
1970-05
1970-11
1971-05
1971-11
Type here to search
ENG
23:42
28-11-2018


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
Apps
securepg.fssnet.co.in
CoinExchange.io
DracoFX | Draco Busi
Online Courses | Cert
Free Kundli Online &
GitHub
Sri M's Public Progra
File Finder
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
[55]:
results.forecast(steps=10)
C:\ProgramData\Anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:531: ValueWarning: No supported index is available. Prediction results will be given with an integer index beginning at `start`.
ValueWarning)
[55]: 41265 163.960
41266 163.935
41267 163.910
41268 163.810
41269 163.940
41270 163.950
41271 163.890
41272 163.860
41273 163.870
41274 163.760
dtype: float64
[56]:
results.predict(start=41264,end=41274)
C:\ProgramData\Anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:531: ValueWarning: No supported index is available. Prediction results will be given with an integer index beginning at `start`.
ValueWarning)
[56]: 41264 163.930
41265 163.960
41266 163.935
41267 163.910
Type here to search
ENG
23:42
28-11-2018


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
41265 163.960
41266 163.935
41267 163.910
41268 163.810
41269 163.940
41270 163.950
41271 163.890
41272 163.860
41273 163.870
41274 163.760
dtype: float64
[57]: #Accuracy of the Forecast using MSE-Mean Squared Error
[58]: from sklearn.metrics import mean_squared_error,mean_absolute_error
print('Mean Squared Error NASDAQ.AAPL -', mean_squared_error(df_AAPL['NASDAQ.AAPL'],df_AAPL['Forecast']))
print('Mean Absolute Error NASDAQ.AAPL -', mean_absolute_error(df_AAPL['NASDAQ.AAPL'],df_AAPL['Forecast']))
Mean Squared Error NASDAQ.AAPL - 0.6426408211943265
Mean Absolute Error NASDAQ.AAPL - 0.0755072821605840
[59]: #Time Series Forecasting for NASDAQ.ADP
[60]: df_ADP = final[['Month',stock_features[1]]]
[61]: df_ADP.head()
Type here to search
ENG
23:42
28-11-2018


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
[61]: df_ADP.head()
[61]:
Month NASDAQ.ADP
0 1970-01-01 102.2300
1 1970-01-01 102.1400
2 1970-01-01 102.2125
3 1970-01-01 102.1400
4 1970-01-01 102.0600
[62]: df_ADP.set_index('Month',inplace=True)
df_ADP.head()
[62]:
NASDAQ.ADP
Month
1970-01-01 102.2300
1970-01-01 102.1400
1970-01-01 102.2125
1970-01-01 102.1400
Type here to search
ENG
23:43
28-11-2018


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
1970-01-01 102.2125
1970-01-01 102.1400
1970-01-01 102.0600
[63]: #Visualize Data
[64]: df_ADP.plot()
plt.title('Time Series Plot for NASDAQ_ADP')
plt.show()
Time Series Plot for NASDAQ_ADP
NASDAQ.ADP
120
115
110
105
100
95
1967-11
1968-05
1968-11
1969-05
1969-11
1970-05
1970-11
1971-05
1971-11
Month
Type here to search
ENG
23:43
28-11-2018


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
Month
[65]: test_stationarity(df_ADP['NASDAQ.ADP'])
Rolling Mean & Standard Deviation
Original
Rolling Mean
Rolling Std
120
100
80
60
40
20
0
1967-11 1968-05 1968-11 1969-05 1969-11 1970-05 1970-11 1971-05 1971-11
Augmented Dickey-Fuller Test:
ADF Test Statistic : -1.7041735251574455
p-value : 0.42896344420670196
#Lags Used : 39
Type here to search
ENG
23:43
28-11-2018

```
p-value : 0.42896344420670196
#Lags Used : 39
Number of Observations Used : 41226
Crtical 1% : value -3.430508630650971 6
Crtical 5% : value -2.861610111161057
Crtical 10% : value -2.5668073179094897
weak evidence against null hypothesis, time series has a unit root, indicating it is non-stationary
```

[66]:
```
#MAKING THE TIME SERIES STATIONARY
#Differencing
```

[67]:
```
df_ADP = df_ADP.copy()
df_ADP['First_Difference'] = df_ADP['NASDAQ.ADP'] - df_ADP['NASDAQ.ADP'].shift(1)
```

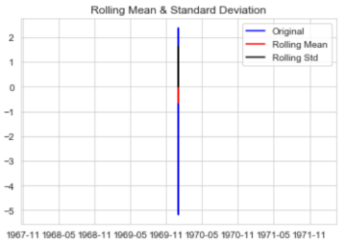
[68]:
```
df_ADP.head()
```

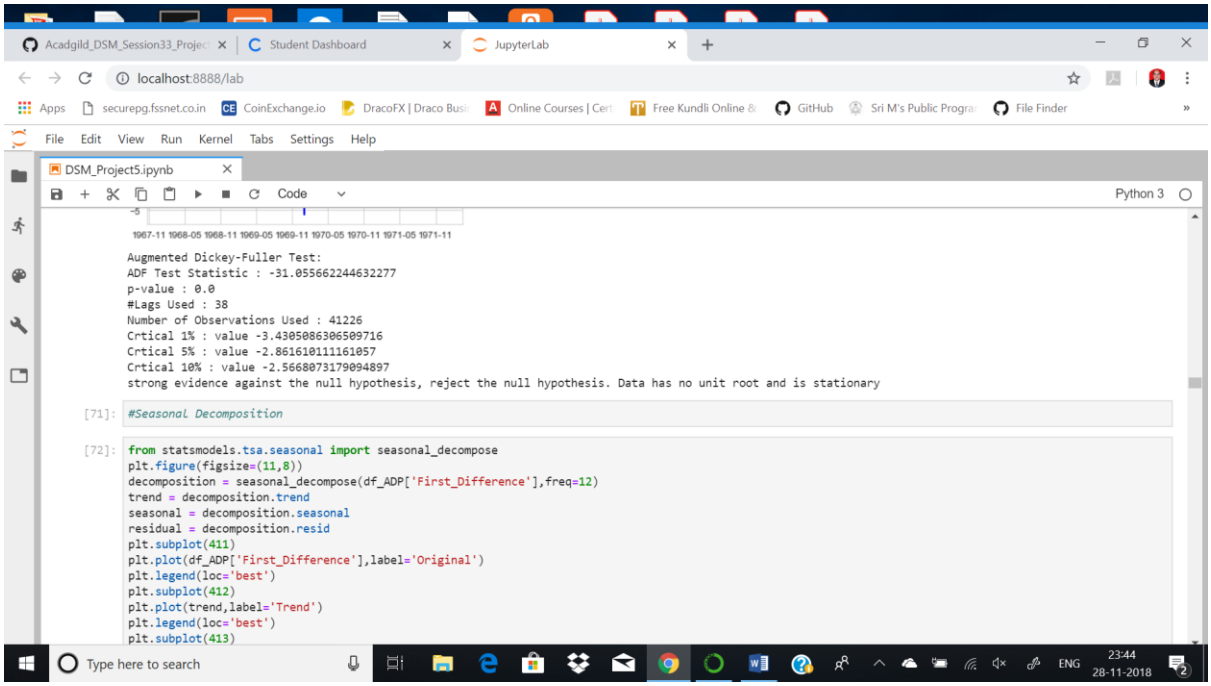
[68]:

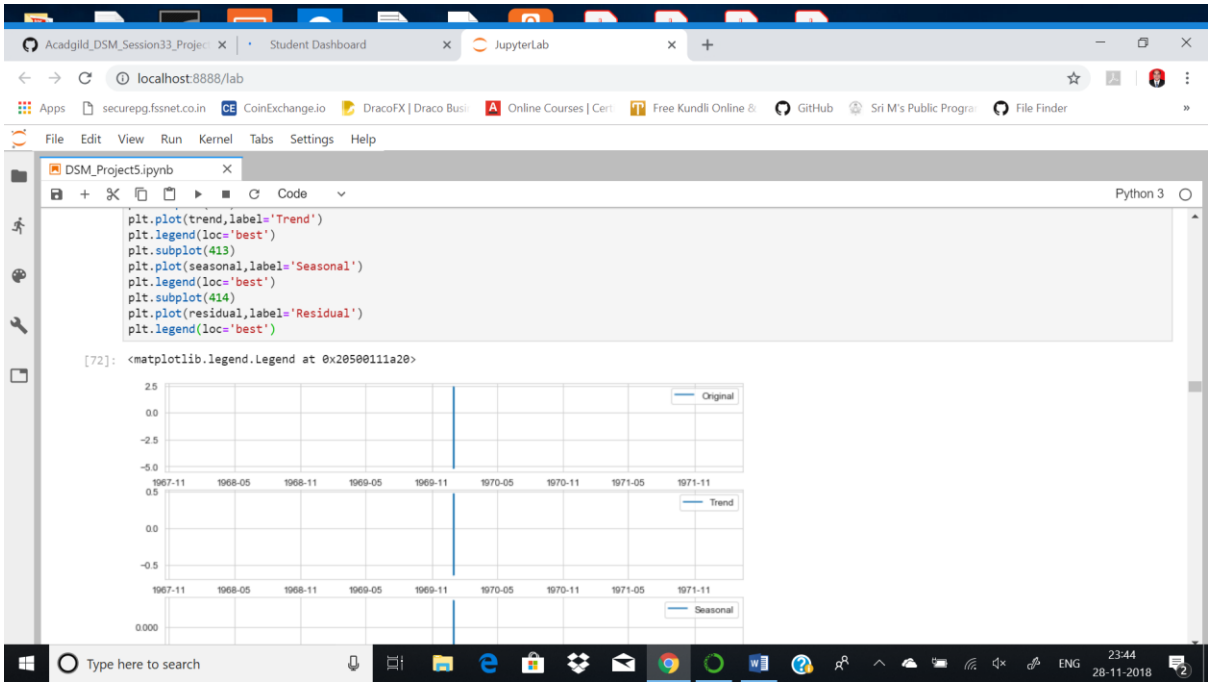
Month	NASDAQ.ADP	First_Difference
1970-01-01	102.2300	NaN
1970-01-01	102.1400	-0.0900
1970-01-01	102.2125	0.0725
1970-01-01	102.1400	-0.0725
1970-01-01	102.0600	-0.0800

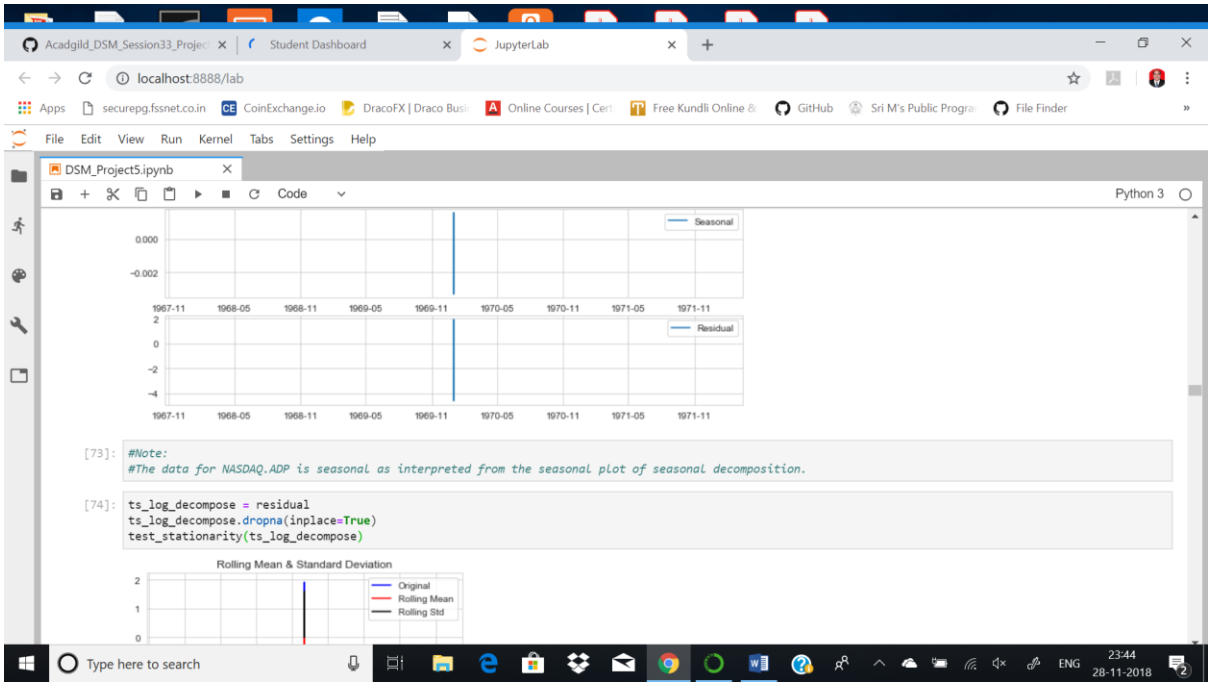
[69]:
```
df_ADP.dropna(inplace=True)
```

[70]:
```
test_stationarity(df_ADP['First_Difference'])
#Now subtract the rolling mean from the original series
```



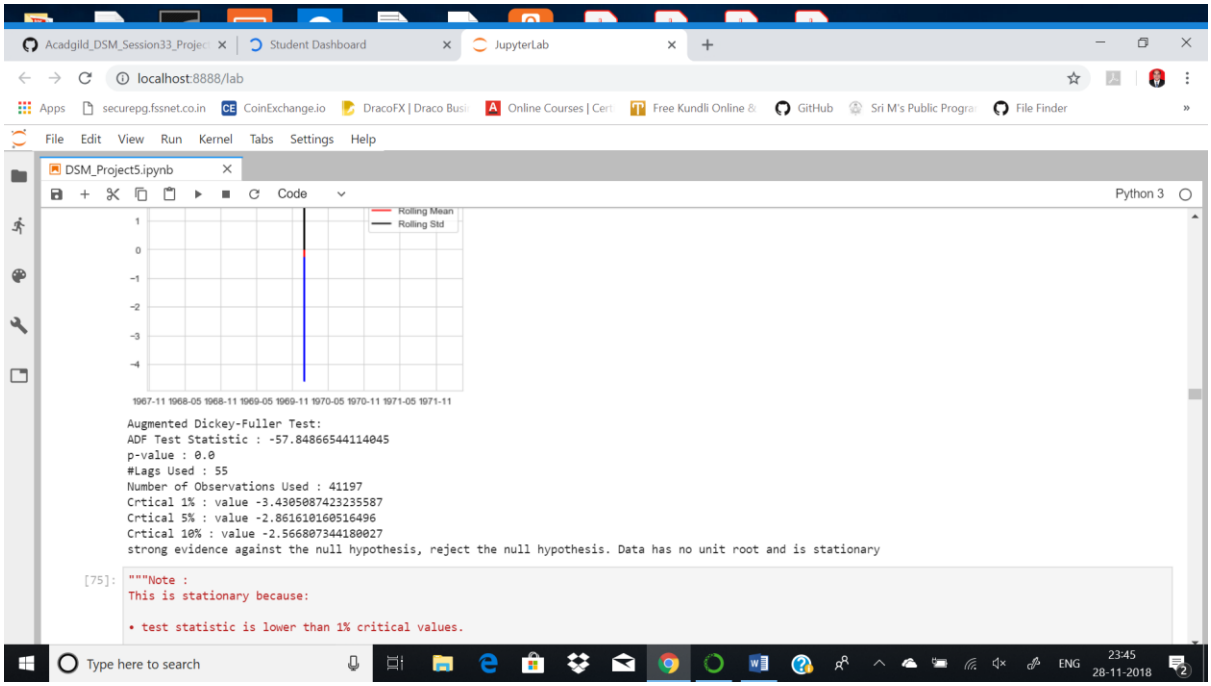

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
1967-11 1968-05 1968-11 1969-05 1969-11 1970-05 1970-11 1971-05 1971-11
Augmented Dickey-Fuller Test:
ADF Test Statistic : -31.055662244632277
p-value : 0.0
#Lags Used : 38
Number of Observations Used : 41226
Crtical 1% : value -3.4305086306509716
Crtical 5% : value -2.861610111161057
Crtical 10% : value -2.5668073179094897
strong evidence against the null hypothesis, reject the null hypothesis. Data has no unit root and is stationary
[71]: #Seasonal Decomposition
[72]: from statsmodels.tsa.seasonal import seasonal_decompose
plt.figure(figsize=(11,8))
decomposition = seasonal_decompose(df_ADP['First_Difference'],freq=12)
trend = decomposition.trend
seasonal = decomposition.seasonal
residual = decomposition.resid
plt.subplot(411)
plt.plot(df_ADP['First_Difference'],label='Original')
plt.legend(loc='best')
plt.subplot(412)
plt.plot(trend,label='Trend')
plt.legend(loc='best')
plt.subplot(413)
Type here to search
ENG
23:44
28-11-2018


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
plt.plot(trend,label='Trend')
plt.legend(loc='best')
plt.subplot(413)
plt.plot(seasonal,label='Seasonal')
plt.legend(loc='best')
plt.subplot(414)
plt.plot(residual,label='Residual')
plt.legend(loc='best')
[72]: <matplotlib.legend.Legend at 0x20500111a20>
Original
Trend
Seasonal
1967-11
1968-05
1968-11
1969-05
1969-11
1970-05
1970-11
1971-05
1971-11
Type here to search
ENG
23:44
28-11-2018



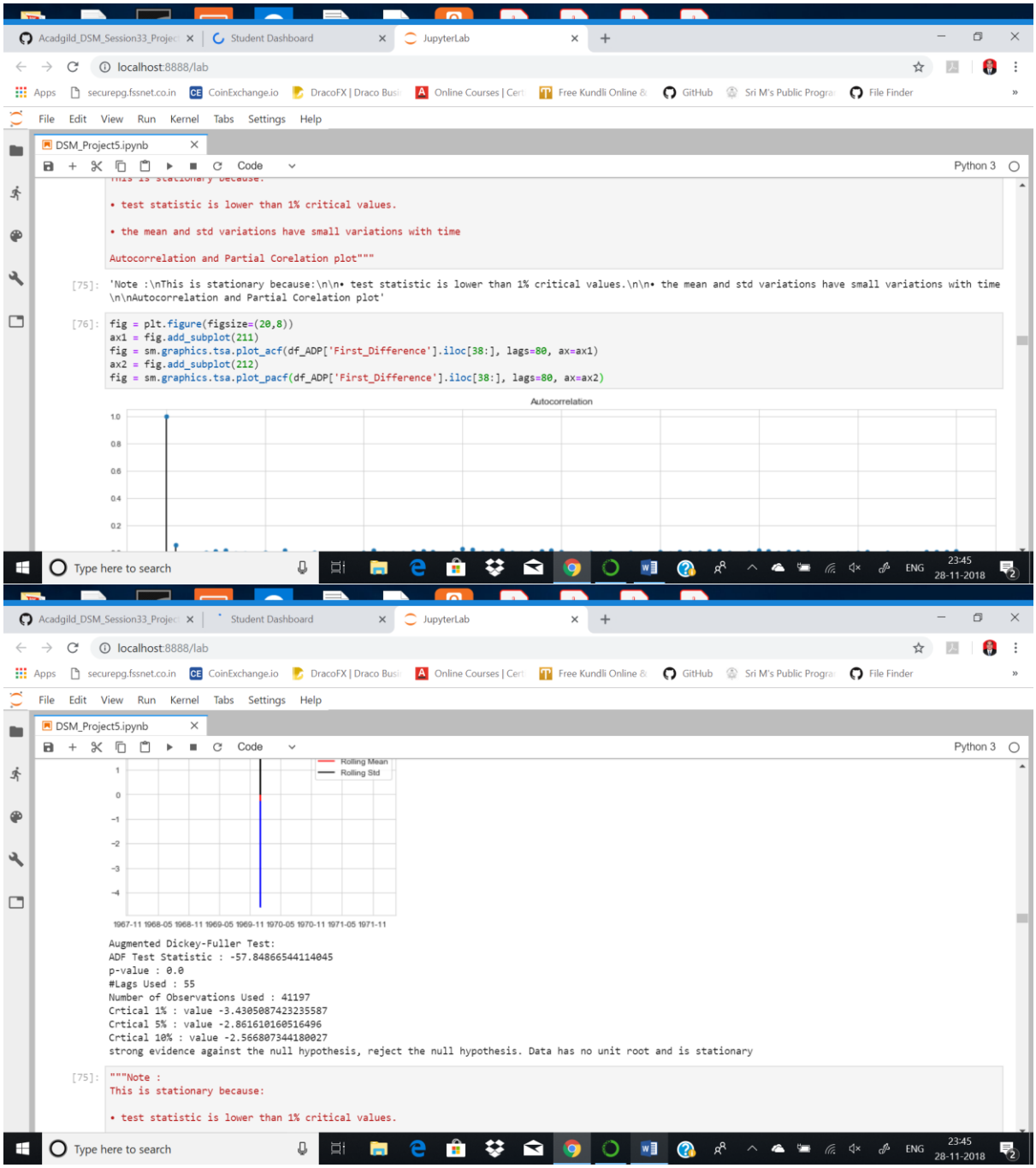
[73]:
```
#Note:
#The data for NASDAQ.ADP is seasonal as interpreted from the seasonal plot of seasonal decomposition.
```

[74]:
```
ts_log_decompose = residual
ts_log_decompose.dropna(inplace=True)
test_stationarity(ts_log_decompose)
```

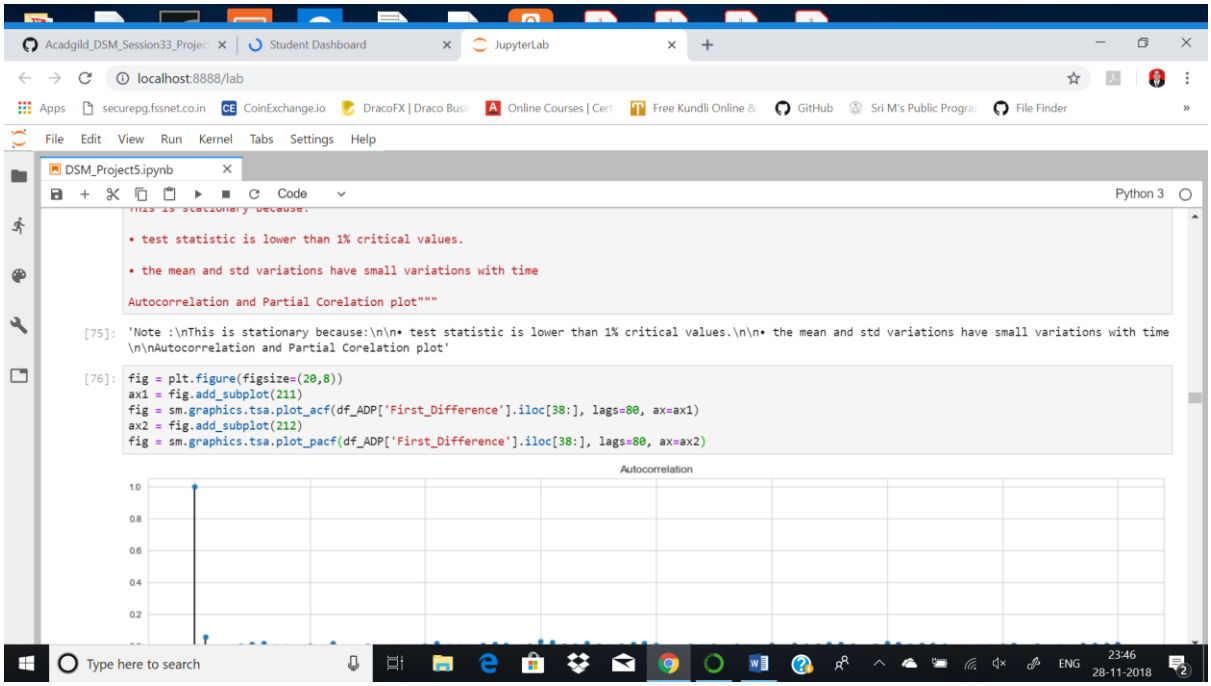



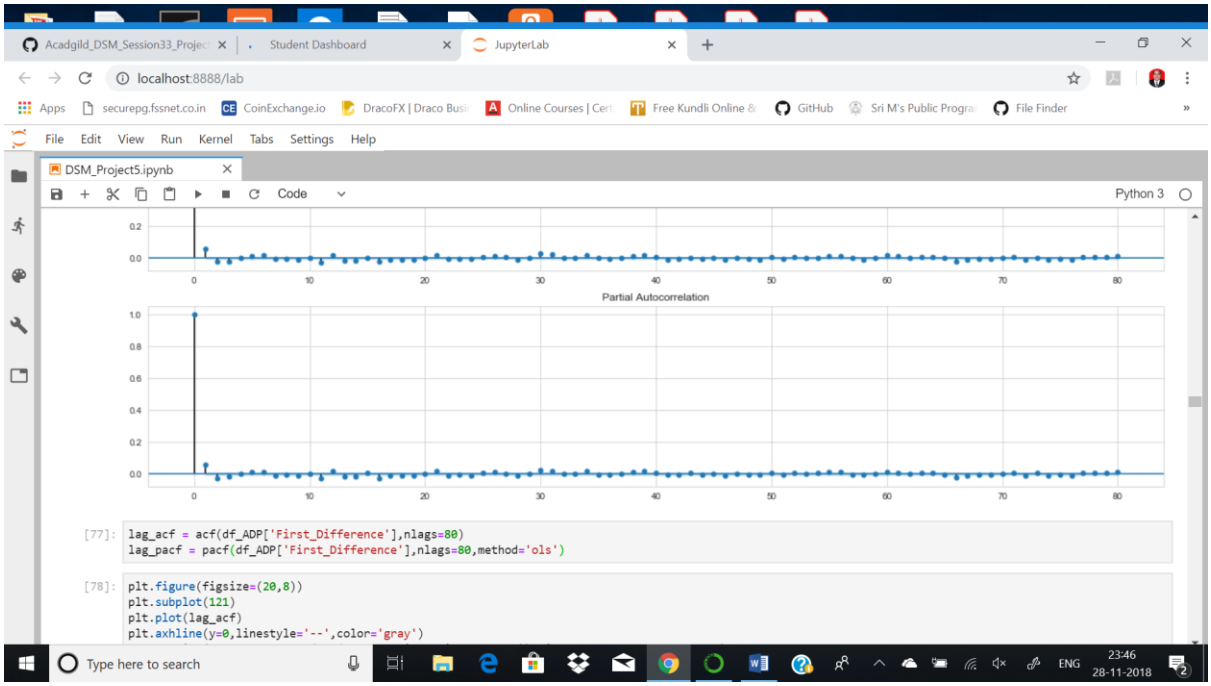

```
Augmented Dickey-Fuller Test:
ADF Test Statistic : -57.84866544114045
p-value : 0.0
#Lags Used : 55
Number of Observations Used : 41197
Crtical 1% : value -3.4305087423235587
Crtical 5% : value -2.86161160516496
Crtical 10% : value -2.566807344180027
strong evidence against the null hypothesis, reject the null hypothesis. Data has no unit root and is stationary
```

[75]:
```
"""Note :
This is stationary because:

• test statistic is lower than 1% critical values.
```


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
Apps
securepg.fssnet.co.in
CoinExchange.io
DracoFX | Draco Busi
Online Courses | Cert
Free Kundli Online &
GitHub
Sri M's Public Progra
File Finder
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
• test statistic is lower than 1% critical values.
• the mean and std variations have small variations with time
Autocorrelation and Partial Corelation plot"""
[75]: 'Note :\nThis is stationary because:\n\n• test statistic is lower than 1% critical values.\n\n• the mean and std variations have small variations with time \n\nAutocorrelation and Partial Corelation plot'
[76]: fig = plt.figure(figsize=(20,8))
ax1 = fig.add_subplot(211)
fig = sm.graphics.tsa.plot_acf(df_ADP['First_Difference'].iloc[38:], lags=80, ax=ax1)
ax2 = fig.add_subplot(212)
fig = sm.graphics.tsa.plot_pacf(df_ADP['First_Difference'].iloc[38:], lags=80, ax=ax2)
Autocorrelation
1.0
0.8
0.6
0.4
0.2
Type here to search
ENG
23:45
28-11-2018

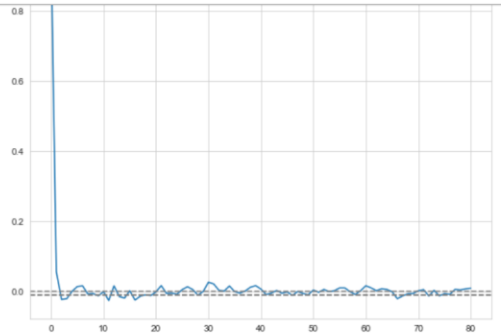
Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
Apps
securepg.fssnet.co.in
CoinExchange.io
DracoFX | Draco Busi
Online Courses | Cert
Free Kundli Online &
GitHub
Sri M's Public Progra
File Finder
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
Rolling Mean
Rolling Std
1
0
-1
-2
-3
-4
1967-11 1968-05 1968-11 1969-05 1969-11 1970-05 1970-11 1971-05 1971-11
Augmented Dickey-Fuller Test:
ADF Test Statistic : -57.84866544114045
p-value : 0.0
#Lags Used : 55
Number of Observations Used : 41197
Crtical 1% : value -3.4305087423235587
Crtical 5% : value -2.861610160516496
Crtical 10% : value -2.566807344180027
strong evidence against the null hypothesis, reject the null hypothesis. Data has no unit root and is stationary
[75]: """Note :
This is stationary because:
• test statistic is lower than 1% critical values.
Type here to search
ENG
23:45
28-11-2018

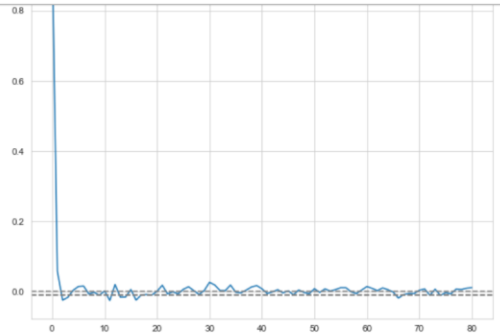

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
• test statistic is lower than 1% critical values.
• the mean and std variations have small variations with time
Autocorrelation and Partial Corelation plot"""
[75]: 'Note :\nThis is stationary because:\n\n• test statistic is lower than 1% critical values.\n\n• the mean and std variations have small variations with time \n\nAutocorrelation and Partial Corelation plot'
[76]: fig = plt.figure(figsize=(20,8))
ax1 = fig.add_subplot(211)
fig = sm.graphics.tsa.plot_acf(df_ADP['First_Difference'].iloc[38:], lags=80, ax=ax1)
ax2 = fig.add_subplot(212)
fig = sm.graphics.tsa.plot_pacf(df_ADP['First_Difference'].iloc[38:], lags=80, ax=ax2)
Autocorrelation
Type here to search
ENG
23:46
28-11-2018


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
Partial Autocorrelation
[77]: lag_acf = acf(df_ADP['First_Difference'],nlags=80)
lag_pacf = pacf(df_ADP['First_Difference'],nlags=80,method='ols')
[78]: plt.figure(figsize=(20,8))
plt.subplot(121)
plt.plot(lag_acf)
plt.axhline(y=0,linestyle='--',color='gray')
Type here to search
ENG
23:46
28-11-2018

```
plt.subplot(121)
plt.plot(lag_acf)
plt.axhline(y=0,linestyle='--',color='gray')
plt.axhline(y=-1.96/np.sqrt(len(df_ADP['First_Difference'])),linestyle='--',color='gray')
plt.axhline(y=-1.96/np.sqrt(len(df_ADP['First_Difference'])),linestyle='--',color='gray')

plt.title('Autocorrelation')

plt.subplot(122)

plt.plot(lag_pacf)
plt.axhline(y=0,linestyle='--',color='gray')
plt.axhline(y=-1.96/np.sqrt(len(df_ADP['First_Difference'])),linestyle='--',color='gray')
plt.axhline(y=-1.96/np.sqrt(len(df_ADP['First_Difference'])),linestyle='--',color='gray')

plt.title('Partial Autocorrelation')
```

[78]:
```
Text(0.5,1,'Partial Autocorrelation')
```





[79]:
```
"""Note
The two dotted lines on either sides of 0 are the confidence intervals.

These can be used to determine the 'p' and 'q' values as:
```

Acadgild_DSM_Session33_Projec | Student Dashboard | JupyterLab

localhost:8888/lab

Apps | securepg.fssnet.co.in | CoinExchange.io | DracoFX | Draco Busi | Online Courses | Cert | Free Kundli Online & | GitHub | Sri M's Public Progra | File Finder

File Edit View Run Kernel Tabs Settings Help

DSM_Project5.ipynb

Code | Python 3

These can be used to determine the 'p' and 'q' values as:

• p: The first time where the PACF crosses the upper confidence interval, here its close to 0. hence p = 0.

• q: The first time where the ACF crosses the upper confidence interval, here its close to 0. hence p = 0."""

[79]:
```
'Note\nThe two dotted lines on either sides of 0 are the confidence intervals.\n\nThese can be used to determine the 'p' and 'q' values as:\n\n• p: The fir
st time where the PACF crosses the upper confidence interval, here its close to 0. hence p = 0.\n\n• q: The first time where the ACF crosses the upper conf
idence interval, here its close to 0. hence p = 0.'
```

[80]:
```
model= sm.tsa.statespace.SARIMAX(df_ADP['NASDAQ.ADP'],order=(0,1,0),seasonal_order=(0,1,0,12))
results = model.fit()
print(results.summary())
```

```
C:\ProgramData\Anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:225: ValueWarning: A date index has been provided, but it has no associated fr
equency information and so will be ignored when e.g. forecasting.
  ' ignored when e.g. forecasting.', ValueWarning)
                           Statespace Model Results
==============================================================================================
Dep. Variable:                          NASDAQ.ADP   No. Observations:                41265
Model:             SARIMAX(0, 1, 0)x(0, 1, 0, 12)   Log Likelihood              34733.013
Date:                            Wed, 28 Nov 2018   AIC                         -69464.026
Time:                                    23:03:16   BIC                         -69455.399
Sample:                                         0   HQIC                        -69461.299
                                            - 41265
Covariance Type:                              opg
```

Type here to search | ENG 23:47 28-11-2018

Acadgild_DSM_Session33_Projec | Student Dashboard | JupyterLab

localhost:8888/lab

Apps | securepg.fssnet.co.in | CoinExchange.io | DracoFX | Draco Busi | Online Courses | Cert | Free Kundli Online & | GitHub | Sri M's Public Progra | File Finder

File Edit View Run Kernel Tabs Settings Help

DSM_Project5.ipynb

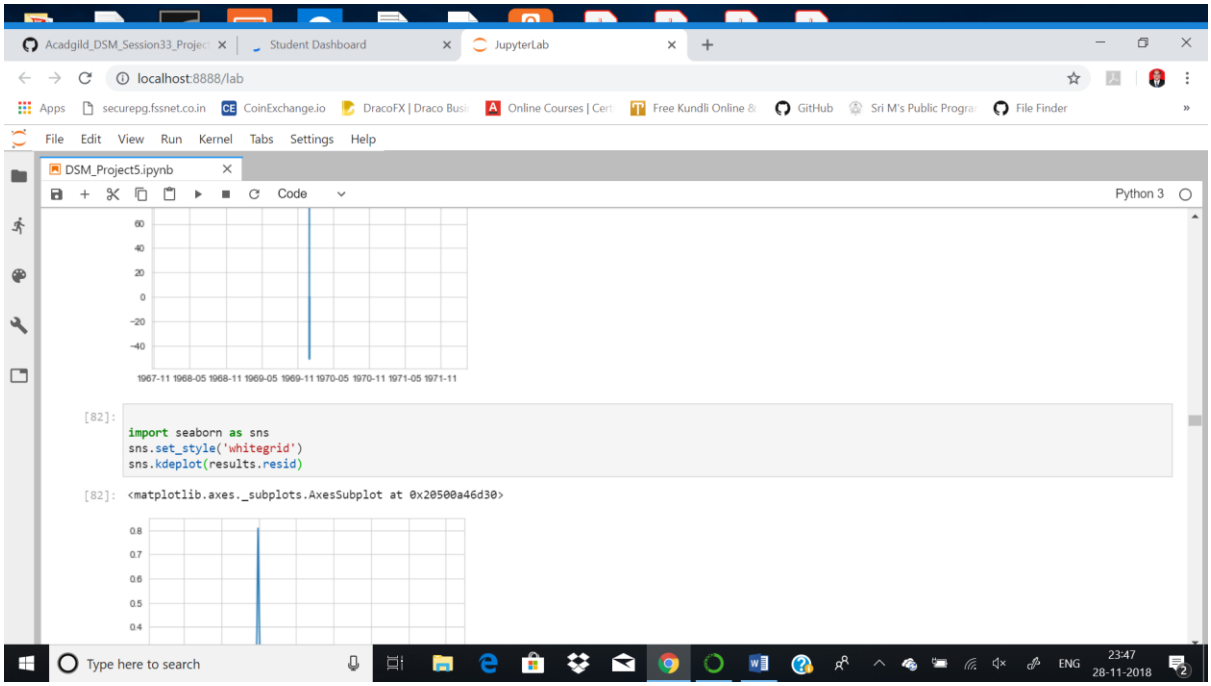
Code | Python 3

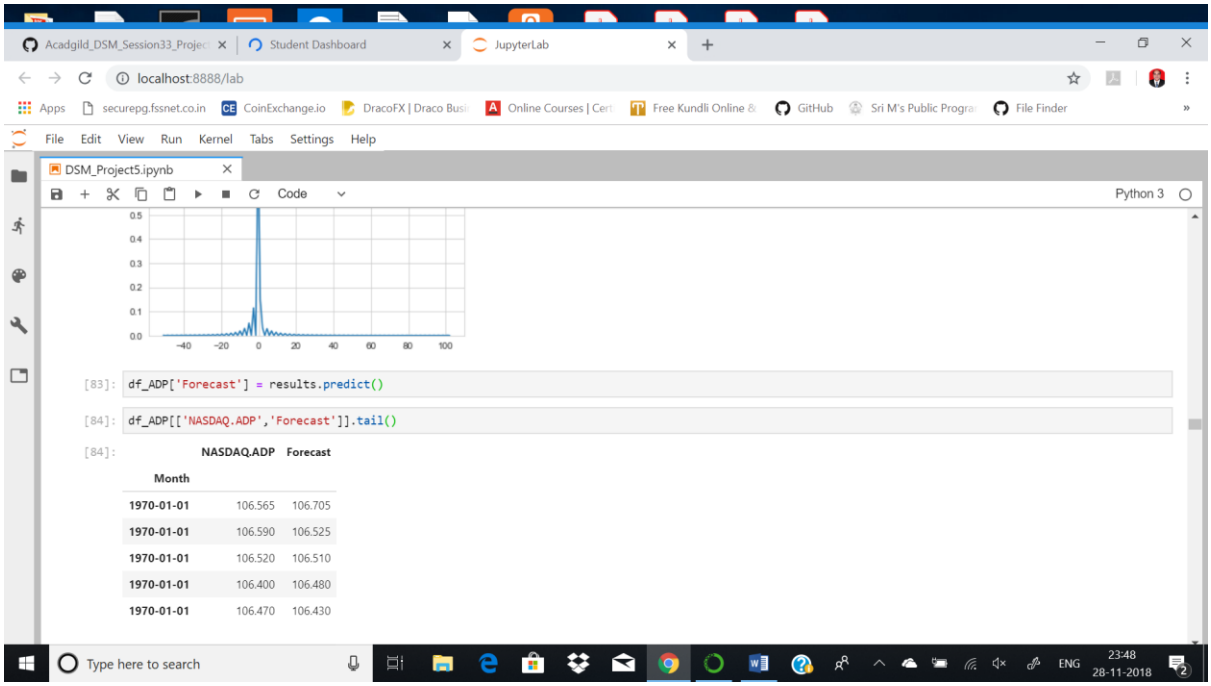
```
Sample:                                 0   HQIC                        -69461.299
                                    - 41265
Covariance Type:                      opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
sigma2         0.0109   5.34e-06   2036.759      0.000       0.011       0.011
===================================================================================
Ljung-Box (Q):                    10628.96   Jarque-Bera (JB):         275266215.34
Prob(Q):                              0.00   Prob(JB):                         0.00
Heteroskedasticity (H):               2.20   Skew:                            -1.59
Prob(H) (two-sided):                  0.00   Kurtosis:                       403.17
===================================================================================

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

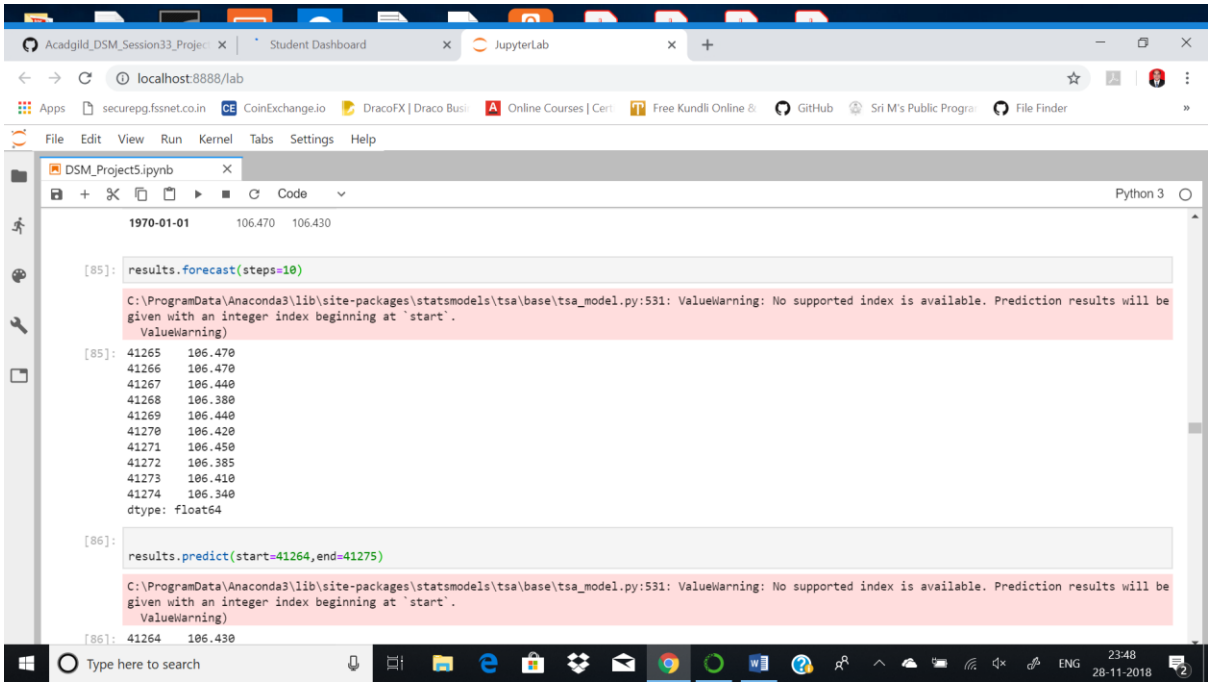
[81]:
```
plt.plot(results.resid)
```

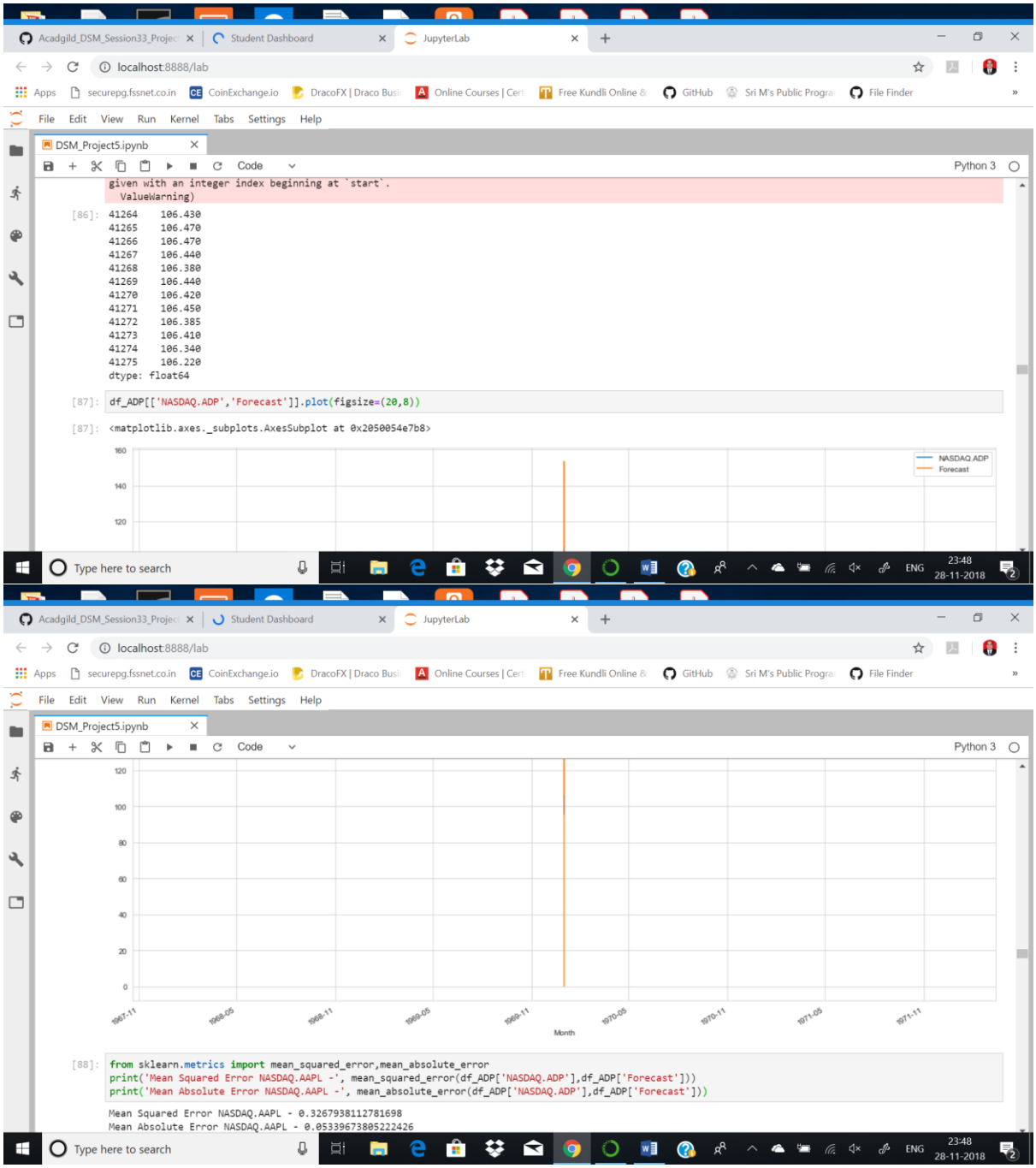
[81]: [<matplotlib.lines.Line2D at 0x20500a5cc50>]

Type here to search | ENG 23:47 28-11-2018


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
Apps
securepg.fssnet.co.in
CoinExchange.io
DracoFX | Draco Busi
Online Courses | Cert
Free Kundli Online &
GitHub
Sri M's Public Progra
File Finder
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
60
40
20
0
-20
-40
1967-11 1968-05 1968-11 1969-05 1969-11 1970-05 1970-11 1971-05 1971-11
[82]:
import seaborn as sns
sns.set_style('whitegrid')
sns.kdeplot(results.resid)
[82]: <matplotlib.axes._subplots.AxesSubplot at 0x20500a46d30>
0.8
0.7
0.6
0.5
0.4
Type here to search
ENG
23:47
28-11-2018


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
Apps
securepg.fssnet.co.in
CoinExchange.io
DracoFX | Draco Busi
Online Courses | Cert
Free Kundli Online &
GitHub
Sri M's Public Progra
File Finder
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
0.5
0.4
0.3
0.2
0.1
0.0
-40 -20 0 20 40 60 80 100
[83]: df_ADP['Forecast'] = results.predict()
[84]: df_ADP[['NASDAQ.ADP','Forecast']].tail()
[84]:
NASDAQ.ADP Forecast
Month
1970-01-01 106.565 106.705
1970-01-01 106.590 106.525
1970-01-01 106.520 106.510
1970-01-01 106.400 106.480
1970-01-01 106.470 106.430
Type here to search
ENG
23:48
28-11-2018


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
Apps
securepg.fssnet.co.in
CoinExchange.io
DracoFX | Draco Busi
Online Courses | Cert
Free Kundli Online &
GitHub
Sri M's Public Progra
File Finder
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
1970-01-01 106.470 106.430
[85]: results.forecast(steps=10)
C:\ProgramData\Anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:531: ValueWarning: No supported index is available. Prediction results will be given with an integer index beginning at `start`.
ValueWarning)
[85]: 41265 106.470
41266 106.470
41267 106.440
41268 106.380
41269 106.440
41270 106.420
41271 106.450
41272 106.385
41273 106.410
41274 106.340
dtype: float64
[86]: results.predict(start=41264,end=41275)
C:\ProgramData\Anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:531: ValueWarning: No supported index is available. Prediction results will be given with an integer index beginning at `start`.
ValueWarning)
[86]: 41264 106.430
Type here to search
ENG
23:48
28-11-2018


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
given with an integer index beginning at `start`.
ValueWarning)
[86]:
41264 106.430
41265 106.470
41266 106.470
41267 106.440
41268 106.380
41269 106.440
41270 106.420
41271 106.450
41272 106.385
41273 106.410
41274 106.340
41275 106.220
dtype: float64
[87]: df_ADP[['NASDAQ.ADP','Forecast']].plot(figsize=(20,8))
[87]: <matplotlib.axes._subplots.AxesSubplot at 0x2050054e7b8>
160
140
120
NASDAQ.ADP
Forecast
Type here to search
ENG
23:48
28-11-2018
Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
120
100
80
60
40
20
0
1967-11
1968-05
1968-11
1969-05
1969-11
1970-05
1970-11
1971-05
1971-11
Month
[88]:
from sklearn.metrics import mean_squared_error,mean_absolute_error
print('Mean Squared Error NASDAQ.AAPL -', mean_squared_error(df_ADP['NASDAQ.ADP'],df_ADP['Forecast']))
print('Mean Absolute Error NASDAQ.AAPL -', mean_absolute_error(df_ADP['NASDAQ.ADP'],df_ADP['Forecast']))
Mean Squared Error NASDAQ.AAPL - 0.3267938112781698
Mean Absolute Error NASDAQ.AAPL - 0.0533967380522242
Type here to search
ENG
23:48
28-11-2018

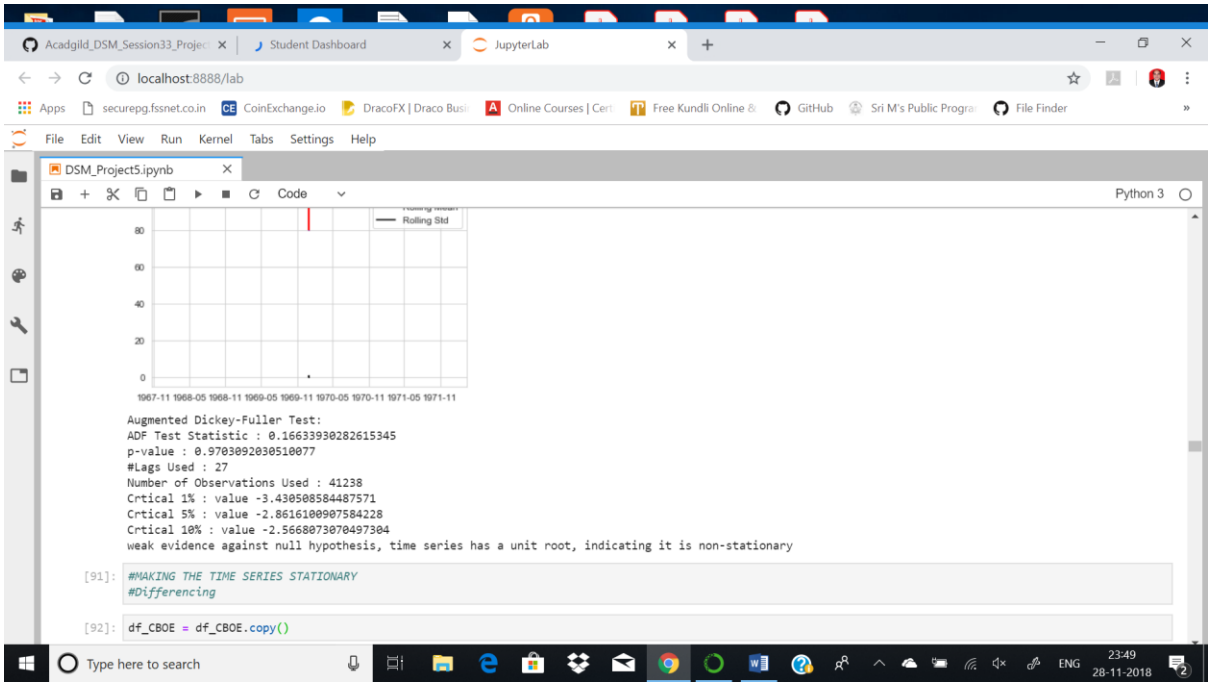
Mean Squared Error NASDAQ.AAPL - 0.3267938112781698
Mean Absolute Error NASDAQ.AAPL - 0.0533967380522242 6

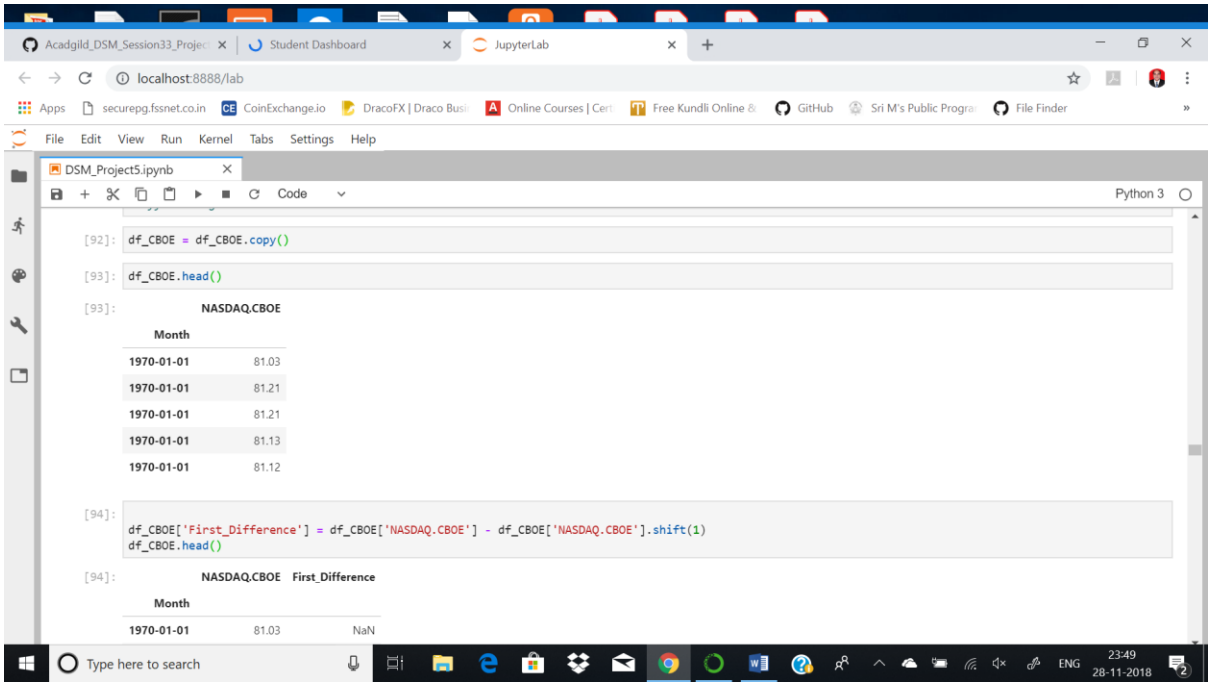
[89]:
```
#Times Series Forecasting for 'NASDAQ.CBOE'
```

[90]:
```
df_CBOE= final[['Month',stock_features[2]]]
print(df_CBOE.head())
df_CBOE.set_index('Month',inplace=True)
print(df_CBOE.head())

df_CBOE.plot()
plt.title('Time Series Plot for NASDAQ_CBOE')
plt.show()
#test Stationarity
test_stationarity(df_CBOE['NASDAQ.CBOE'])
```

```
       Month  NASDAQ.CBOE
0 1970-01-01        81.03
1 1970-01-01        81.21
2 1970-01-01        81.21
3 1970-01-01        81.13
4 1970-01-01        81.12
            NASDAQ.CBOE
Month
1970-01-01        81.03
1970-01-01        81.21
1970-01-01        81.21
```

1970-01-01 81.03
1970-01-01 81.21
1970-01-01 81.21
1970-01-01 81.13
1970-01-01 81.12


Rolling Std
80
60
40
20
0
1967-11 1968-05 1968-11 1969-05 1969-11 1970-05 1970-11 1971-05 1971-11
Augmented Dickey-Fuller Test:
ADF Test Statistic : 0.16633930282615345
p-value : 0.9703092030510077
#Lags Used : 27
Number of Observations Used : 41238
Crtical 1% : value -3.430508584487571
Crtical 5% : value -2.8616100907584228
Crtical 10% : value -2.5668073070497304
weak evidence against null hypothesis, time series has a unit root, indicating it is non-stationary
[91]: #MAKING THE TIME SERIES STATIONARY
#Differencing
[92]: df_CBOE = df_CBOE.copy()

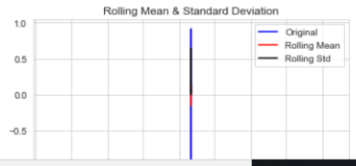

[92]: df_CBOE = df_CBOE.copy()
[93]: df_CBOE.head()
[93]:
NASDAQ.CBOE
Month
1970-01-01 81.03
1970-01-01 81.21
1970-01-01 81.21
1970-01-01 81.13
1970-01-01 81.12
[94]:
df_CBOE['First_Difference'] = df_CBOE['NASDAQ.CBOE'] - df_CBOE['NASDAQ.CBOE'].shift(1)
df_CBOE.head()
[94]:
NASDAQ.CBOE First_Difference
Month
1970-01-01 81.03 NaN

Month		
1970-01-01	81.03	NaN
1970-01-01	81.21	0.18
1970-01-01	81.21	0.00
1970-01-01	81.13	-0.08
1970-01-01	81.12	-0.01

[95]:
```
df_CBOE.dropna(inplace=True)
```

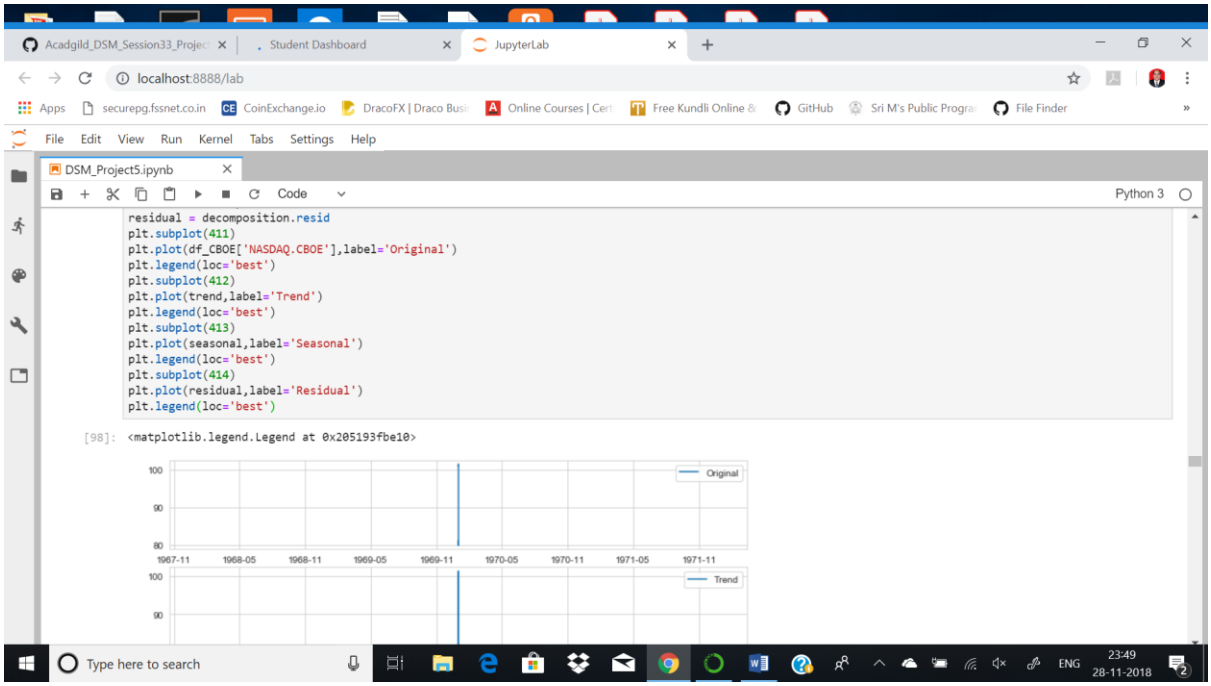
[96]:
```
#Test Seasonality
```

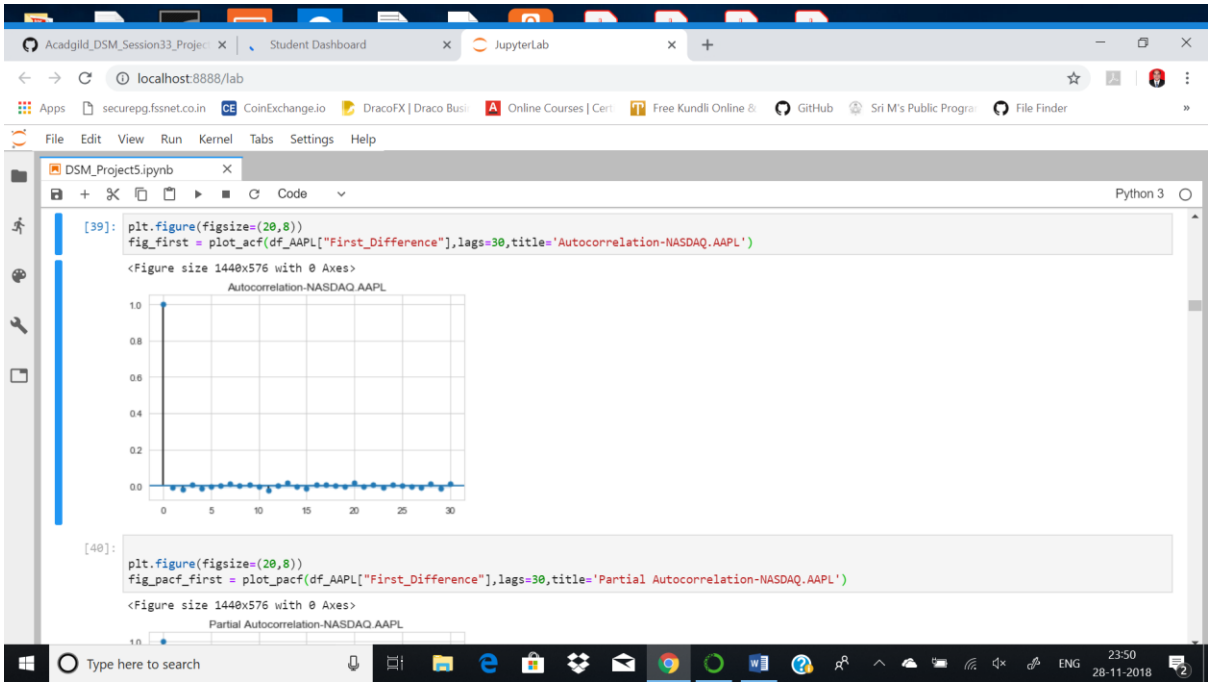
[97]:
```
test_stationarity(df_CBOE['First_Difference'])
```

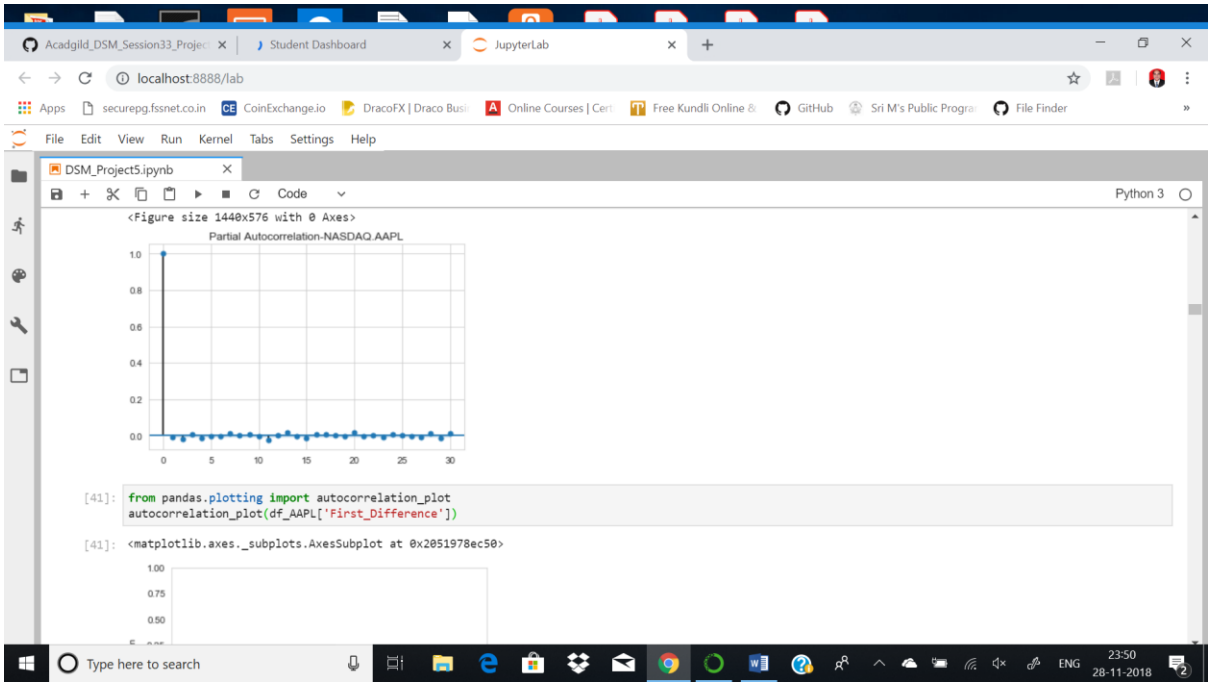


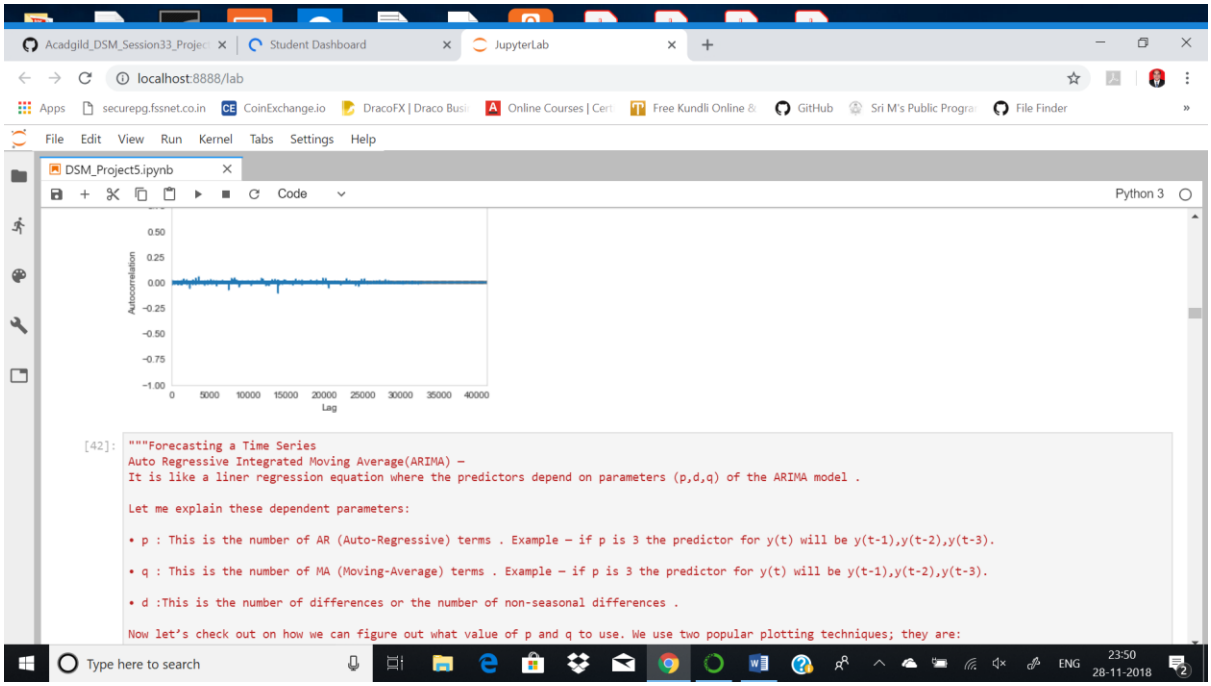

```
Augmented Dickey-Fuller Test:
ADF Test Statistic : -41.64209364543141
p-value : 0.0
#Lags Used : 26
Number of Observations Used : 41238
Crtical 1% : value -3.430508584487571
Crtical 5% : value -2.8616100907584228
Crtical 10% : value -2.5668073070497304
strong evidence against the null hypothesis, reject the null hypothesis. Data has no unit root and is stationary
```

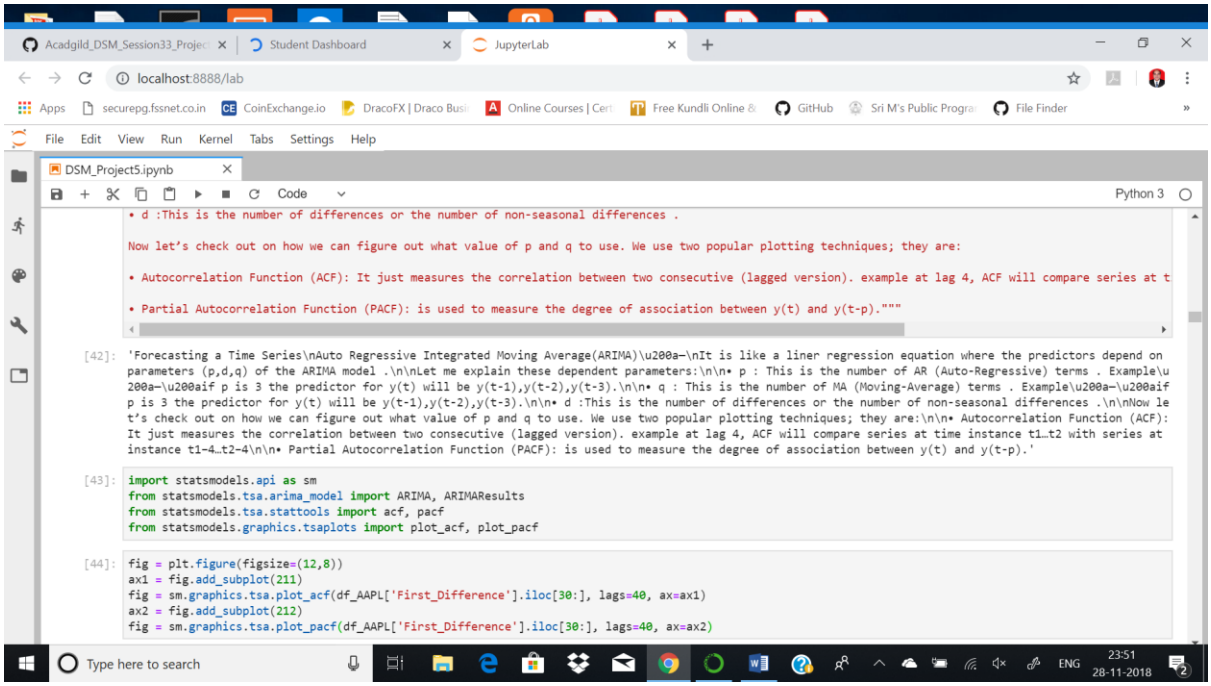
[98]:
```
#Seasonal Decomposition
from statsmodels.tsa.seasonal import seasonal_decompose
plt.figure(figsize=(11,8))
decomposition = seasonal_decompose(df_CBOE['NASDAQ.CBOE'],freq=12)
trend = decomposition.trend
seasonal = decomposition.seasonal
residual = decomposition.resid
plt.subplot(411)
plt.plot(df_CBOE['NASDAQ.CBOE'],label='Original')
```

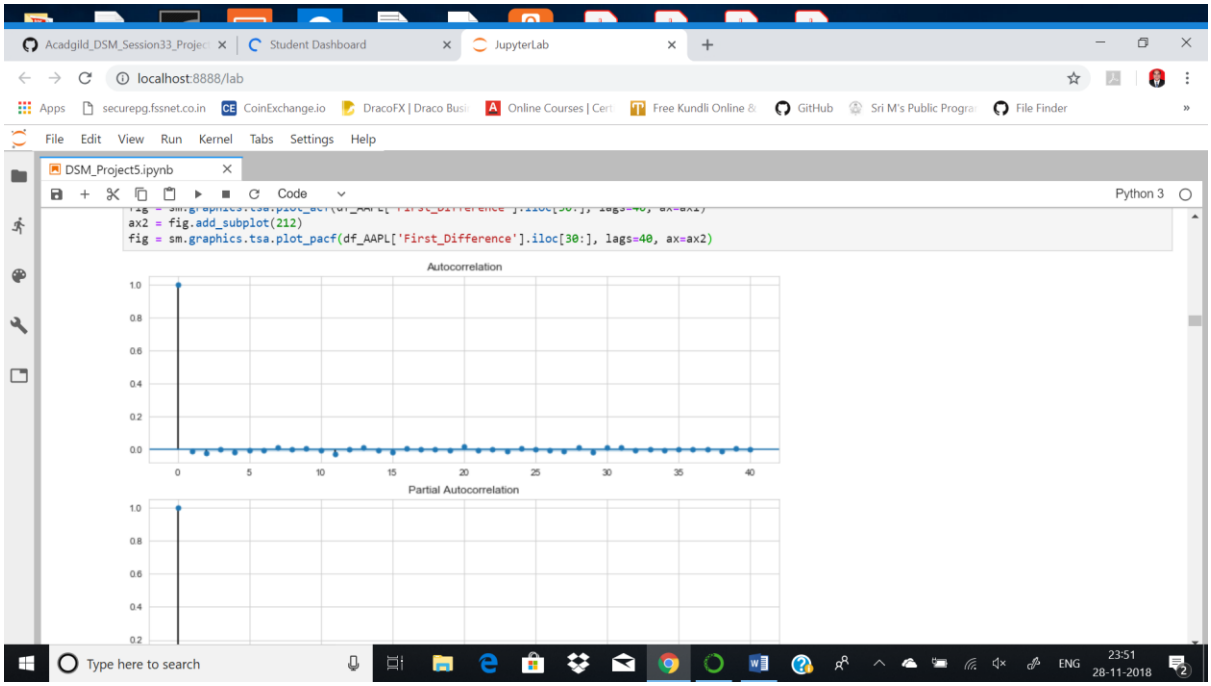

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
Apps
securepg.fssnet.co.in
CoinExchange.io
DracoFX | Draco Busi
Online Courses | Cert
Free Kundli Online &
GitHub
Sri M's Public Progra
File Finder
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
residual = decomposition.resid
plt.subplot(411)
plt.plot(df_CBOE['NASDAQ.CBOE'],label='Original')
plt.legend(loc='best')
plt.subplot(412)
plt.plot(trend,label='Trend')
plt.legend(loc='best')
plt.subplot(413)
plt.plot(seasonal,label='Seasonal')
plt.legend(loc='best')
plt.subplot(414)
plt.plot(residual,label='Residual')
plt.legend(loc='best')
[98]: <matplotlib.legend.Legend at 0x205193fbe10>
Original
Trend
1967-11
1968-05
1968-11
1969-05
1969-11
1970-05
1970-11
1971-05
1971-11
Type here to search
ENG
23:49
28-11-2018

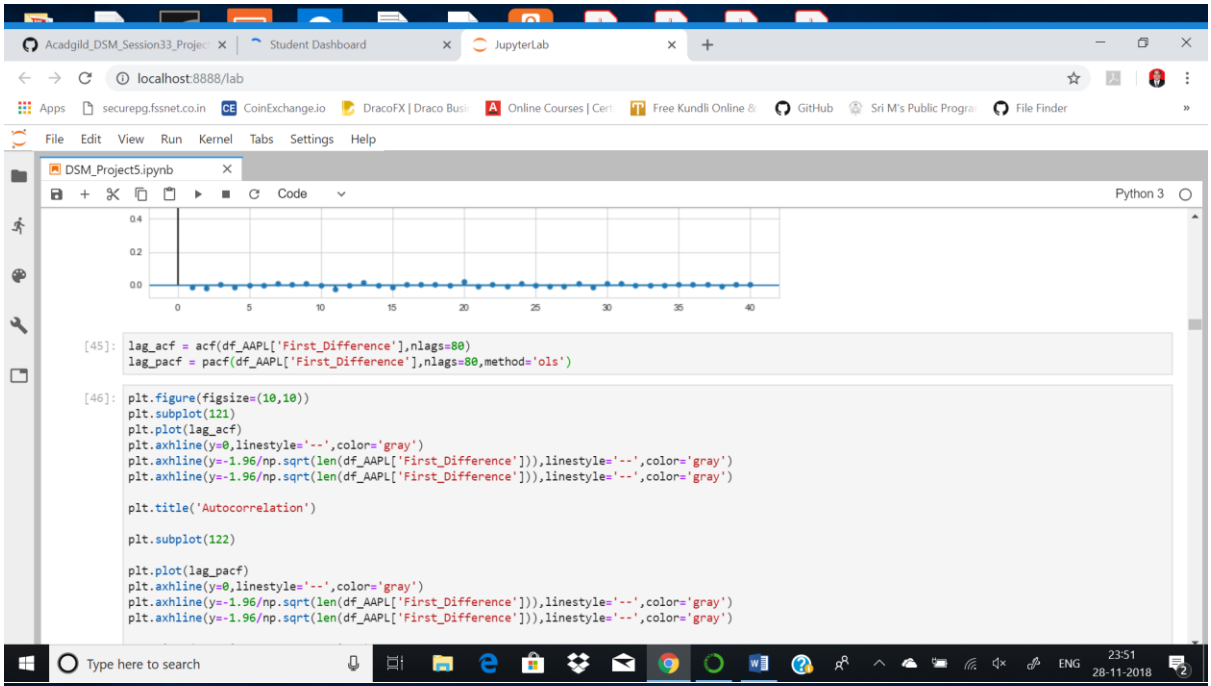

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
[39]: plt.figure(figsize=(20,8))
fig_first = plot_acf(df_AAPL["First_Difference"],lags=30,title='Autocorrelation-NASDAQ.AAPL')
<Figure size 1440x576 with 0 Axes>
Autocorrelation-NASDAQ.AAPL
[40]: plt.figure(figsize=(20,8))
fig_pacf_first = plot_pacf(df_AAPL["First_Difference"],lags=30,title='Partial Autocorrelation-NASDAQ.AAPL')
<Figure size 1440x576 with 0 Axes>
Partial Autocorrelation-NASDAQ.AAPL
Type here to search
ENG
23:50
28-11-2018

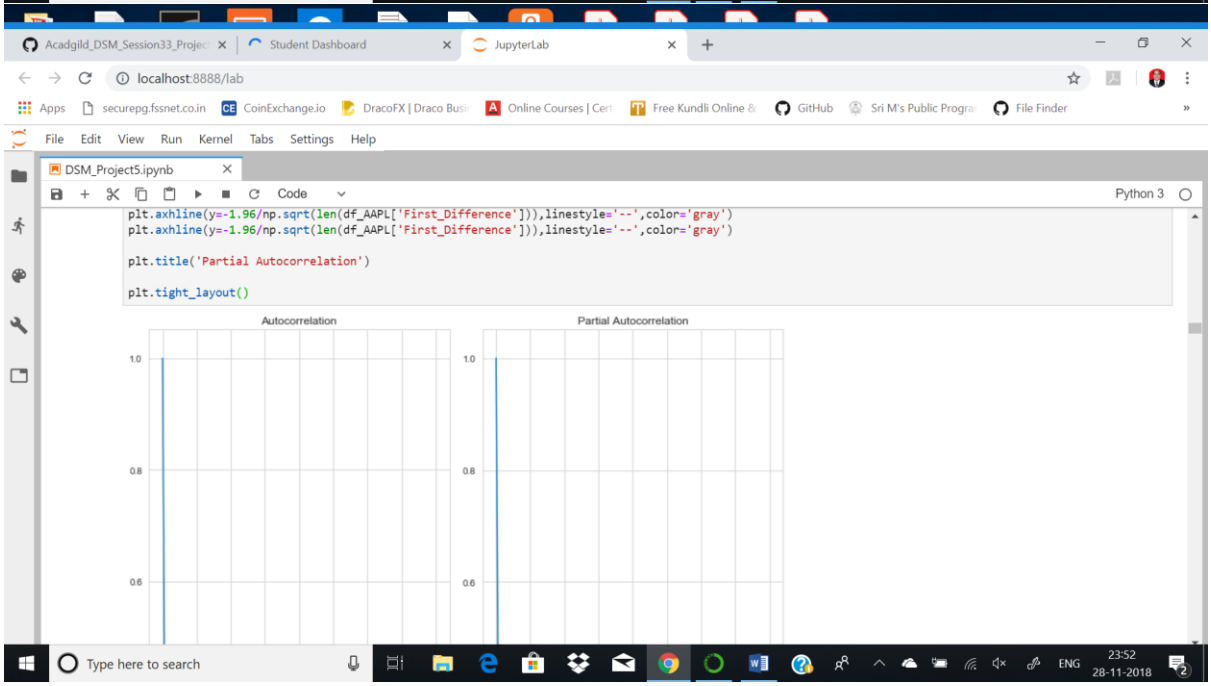

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
Apps
securepg.fssnet.co.in
CoinExchange.io
DracoFX | Draco Busi
Online Courses | Cert
Free Kundli Online &
GitHub
Sri M's Public Progra
File Finder
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
<Figure size 1440x576 with 0 Axes>
Partial Autocorrelation-NASDAQ.AAPL
1.0
0.8
0.6
0.4
0.2
0.0
0 5 10 15 20 25 30
[41]:
from pandas.plotting import autocorrelation_plot
autocorrelation_plot(df_AAPL['First_Difference'])
[41]:
<matplotlib.axes._subplots.AxesSubplot at 0x2051978ec50>
1.00
0.75
0.50
Type here to search
ENG
23:50
28-11-2018

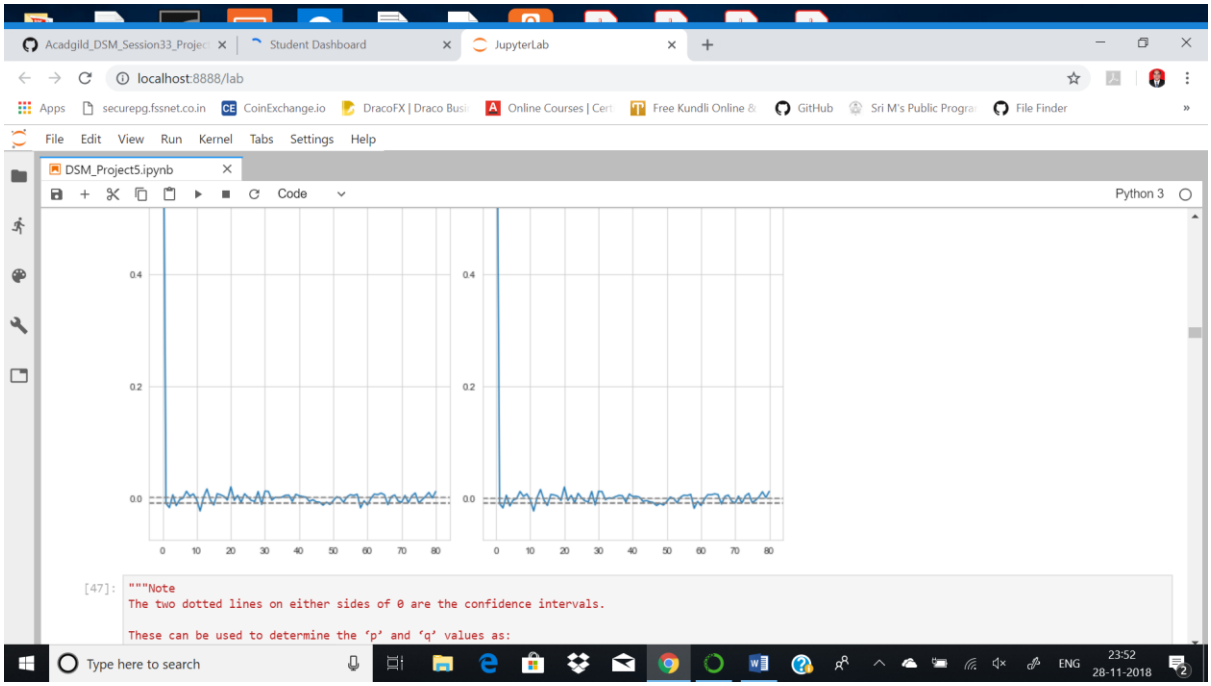

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
Apps
securepg.fssnet.co.in
CoinExchange.io
DracoFX | Draco Busi
Online Courses | Cert
Free Kundli Online &
GitHub
Sri M's Public Progra
File Finder
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
Autocorrelation
0 5000 10000 15000 20000 25000 30000 35000 40000
Lag
[42]:
"""Forecasting a Time Series
Auto Regressive Integrated Moving Average(ARIMA) –
It is like a liner regression equation where the predictors depend on parameters (p,d,q) of the ARIMA model .
Let me explain these dependent parameters:
• p : This is the number of AR (Auto-Regressive) terms . Example – if p is 3 the predictor for y(t) will be y(t-1),y(t-2),y(t-3).
• q : This is the number of MA (Moving-Average) terms . Example – if p is 3 the predictor for y(t) will be y(t-1),y(t-2),y(t-3).
• d :This is the number of differences or the number of non-seasonal differences .
Now let's check out on how we can figure out what value of p and q to use. We use two popular plotting techniques; they are:
Type here to search
ENG
23:50
28-11-2018


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
• d :This is the number of differences or the number of non-seasonal differences .
Now let's check out on how we can figure out what value of p and q to use. We use two popular plotting techniques; they are:
• Autocorrelation Function (ACF): It just measures the correlation between two consecutive (lagged version). example at lag 4, ACF will compare series at t
• Partial Autocorrelation Function (PACF): is used to measure the degree of association between y(t) and y(t-p)."""
[42]: 'Forecasting a Time Series\nAuto Regressive Integrated Moving Average(ARIMA)\u200a–\nIt is like a liner regression equation where the predictors depend on parameters (p,d,q) of the ARIMA model .\n\nLet me explain these dependent parameters:\n\n• p : This is the number of AR (Auto-Regressive) terms . Example\u200a–\u200aif p is 3 the predictor for y(t) will be y(t-1),y(t-2),y(t-3).\n\n• q : This is the number of MA (Moving-Average) terms . Example\u200a–\u200aif p is 3 the predictor for y(t) will be y(t-1),y(t-2),y(t-3).\n\n• d :This is the number of differences or the number of non-seasonal differences .\n\nNow let's check out on how we can figure out what value of p and q to use. We use two popular plotting techniques; they are:\n\n• Autocorrelation Function (ACF): It just measures the correlation between two consecutive (lagged version). example at lag 4, ACF will compare series at time instance t1…t2 with series at instance t1–4…t2–4\n\n• Partial Autocorrelation Function (PACF): is used to measure the degree of association between y(t) and y(t-p).'
[43]: import statsmodels.api as sm
from statsmodels.tsa.arima_model import ARIMA, ARIMAResults
from statsmodels.tsa.stattools import acf, pacf
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
[44]: fig = plt.figure(figsize=(12,8))
ax1 = fig.add_subplot(211)
fig = sm.graphics.tsa.plot_acf(df_AAPL['First_Difference'].iloc[30:], lags=40, ax=ax1)
ax2 = fig.add_subplot(212)
fig = sm.graphics.tsa.plot_pacf(df_AAPL['First_Difference'].iloc[30:], lags=40, ax=ax2)
Type here to search
ENG
23:51
28-11-2018


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
ax2 = fig.add_subplot(212)
fig = sm.graphics.tsa.plot_pacf(df_AAPL['First_Difference'].iloc[30:], lags=40, ax=ax2)
Autocorrelation
Partial Autocorrelation
Type here to search
ENG
23:51
28-11-2018


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
Apps
securepg.fssnet.co.in
CoinExchange.io
DracoFX | Draco Busi
Online Courses | Cert
Free Kundli Online &
GitHub
Sri M's Public Progra
File Finder
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
0.4
0.2
0.0
0 5 10 15 20 25 30 35 40
[45]:
lag_acf = acf(df_AAPL['First_Difference'],nlags=80)
lag_pacf = pacf(df_AAPL['First_Difference'],nlags=80,method='ols')
[46]:
plt.figure(figsize=(10,10))
plt.subplot(121)
plt.plot(lag_acf)
plt.axhline(y=0,linestyle='--',color='gray')
plt.axhline(y=-1.96/np.sqrt(len(df_AAPL['First_Difference'])),linestyle='--',color='gray')
plt.axhline(y=-1.96/np.sqrt(len(df_AAPL['First_Difference'])),linestyle='--',color='gray')
plt.title('Autocorrelation')
plt.subplot(122)
plt.plot(lag_pacf)
plt.axhline(y=0,linestyle='--',color='gray')
plt.axhline(y=-1.96/np.sqrt(len(df_AAPL['First_Difference'])),linestyle='--',color='gray')
plt.axhline(y=-1.96/np.sqrt(len(df_AAPL['First_Difference'])),linestyle='--',color='gray')
Type here to search
ENG
23:51
28-11-2018


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
Apps
securepg.fssnet.co.in
CoinExchange.io
DracoFX | Draco Busi
Online Courses | Cert
Free Kundli Online &
GitHub
Sri M's Public Progra
File Finder
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
plt.axhline(y=-1.96/np.sqrt(len(df_AAPL['First_Difference'])),linestyle='--',color='gray')
plt.axhline(y=-1.96/np.sqrt(len(df_AAPL['First_Difference'])),linestyle='--',color='gray')
plt.title('Partial Autocorrelation')
plt.tight_layout()
Autocorrelation
Partial Autocorrelation
1.0
0.8
0.6
Type here to search
ENG
23:52
28-11-2018


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
Apps
securepg.fssnet.co.in
CoinExchange.io
DracoFX | Draco Busi
Online Courses | Cert
Free Kundli Online &
GitHub
Sri M's Public Progra
File Finder
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
0.4
0.2
0.0
0 10 20 30 40 50 60 70 80
[47]:
"""Note
The two dotted lines on either sides of 0 are the confidence intervals.
These can be used to determine the 'p' and 'q' values as:
Type here to search
ENG
23:52
28-11-2018

Acadgild_DSM_Session33_Projec | Student Dashboard | JupyterLab

localhost:8888/lab

Apps | securepg.fssnet.co.in | CoinExchange.io | DracoFX | Draco Busi | Online Courses | Cert | Free Kundli Online & | GitHub | Sri M's Public Progra | File Finder

File Edit View Run Kernel Tabs Settings Help

DSM_Project5.ipynb

Code | Python 3

[48]:
```
model= sm.tsa.statespace.SARIMAX(df_AAPL['NASDAQ.AAPL'],order=(0,1,0),seasonal_order=(0,1,0,12))
results = model.fit()
print(results.summary())
```

```
C:\ProgramData\Anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:225: ValueWarning: A date index has been provided, but it has no associated fr
equency information and so will be ignored when e.g. forecasting.
  ' ignored when e.g. forecasting.', ValueWarning)
                                Statespace Model Results
==========================================================================================
Dep. Variable:                          NASDAQ.AAPL   No. Observations:                41265
Model:             SARIMAX(0, 1, 0)x(0, 1, 0, 12)   Log Likelihood               24925.552
Date:                            Wed, 28 Nov 2018   AIC                         -49849.104
Time:                                    22:50:24   BIC                         -49840.476
Sample:                                         0   HQIC                        -49846.377
                                            - 41265
Covariance Type:                              opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
sigma2         0.0175   4.57e-06   3828.712      0.000       0.017       0.017
===================================================================================
Ljung-Box (Q):                    10611.64   Jarque-Bera (JB):        3462262308.64
Prob(Q):                              0.00   Prob(JB):                         0.00
Heteroskedasticity (H):               2.92   Skew:                            -2.00
Prob(H) (two-sided):                  0.00   Kurtosis:                      1422.26
===================================================================================
```

Type here to search | ENG 23:52 28-11-2018

Acadgild_DSM_Session33_Projec | Student Dashboard | JupyterLab

localhost:8888/lab

Apps | securepg.fssnet.co.in | CoinExchange.io | DracoFX | Draco Busi | Online Courses | Cert | Free Kundli Online & | GitHub | Sri M's Public Progra | File Finder

File Edit View Run Kernel Tabs Settings Help

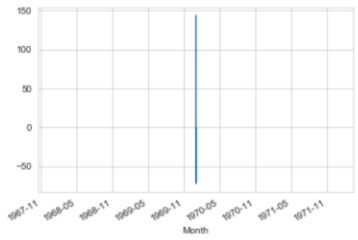
DSM_Project5.ipynb

Code | Python 3

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

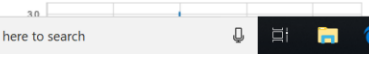
[49]:
```
results.resid.plot()
```

[49]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x2052ea7fcc0>
```

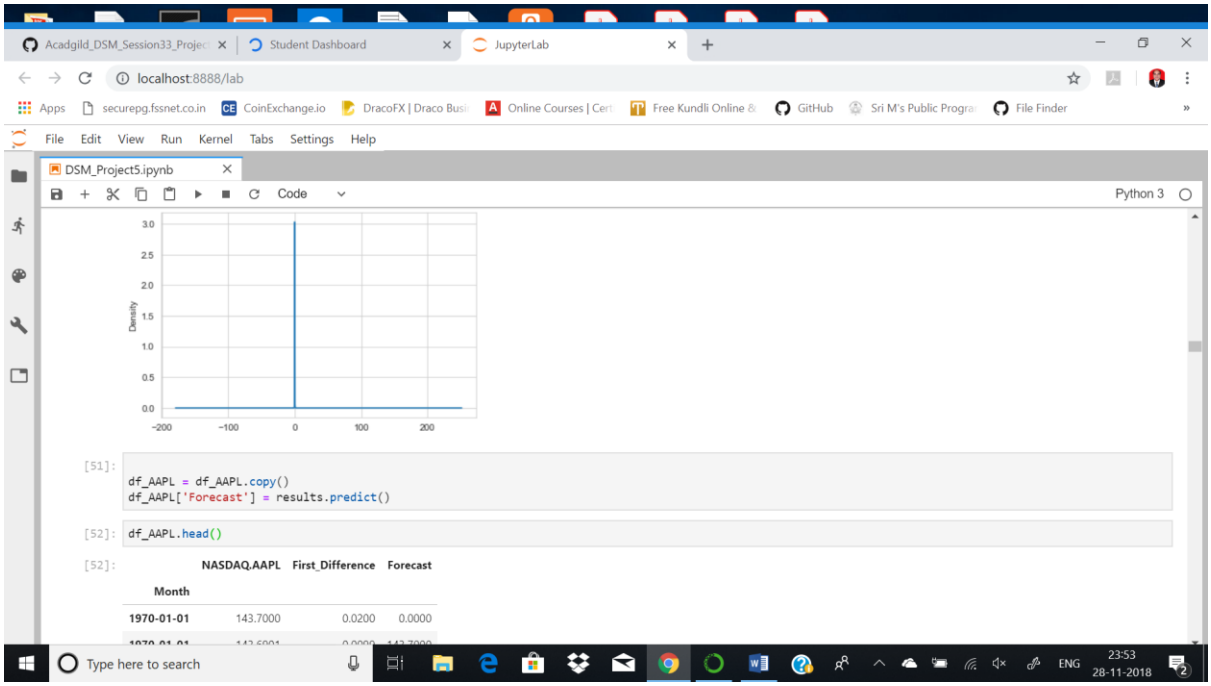


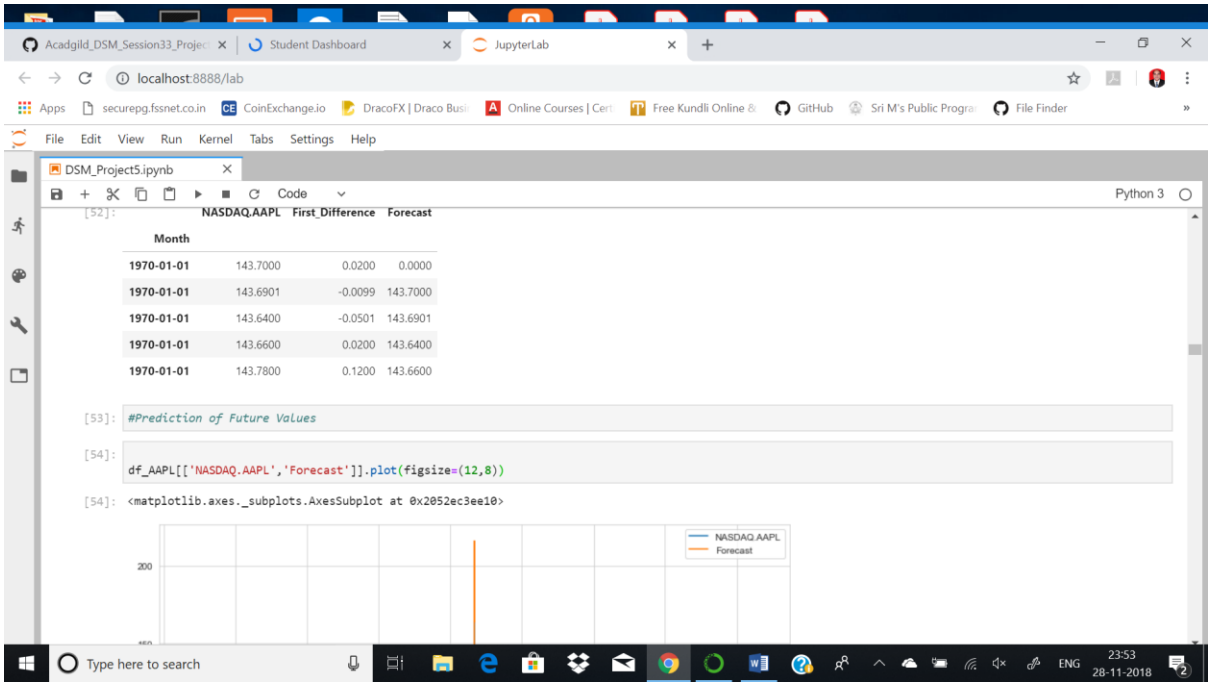
[50]:
```
results.resid.plot(kind='kde')
```

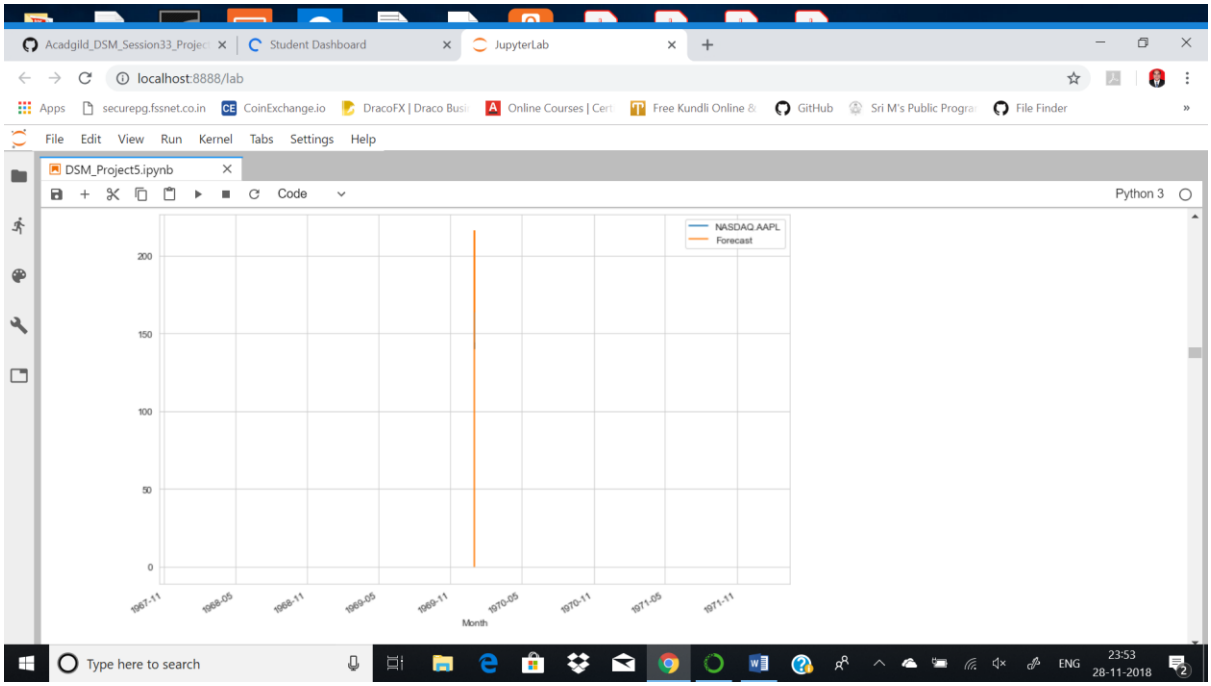
[50]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x205192cdb00>
```

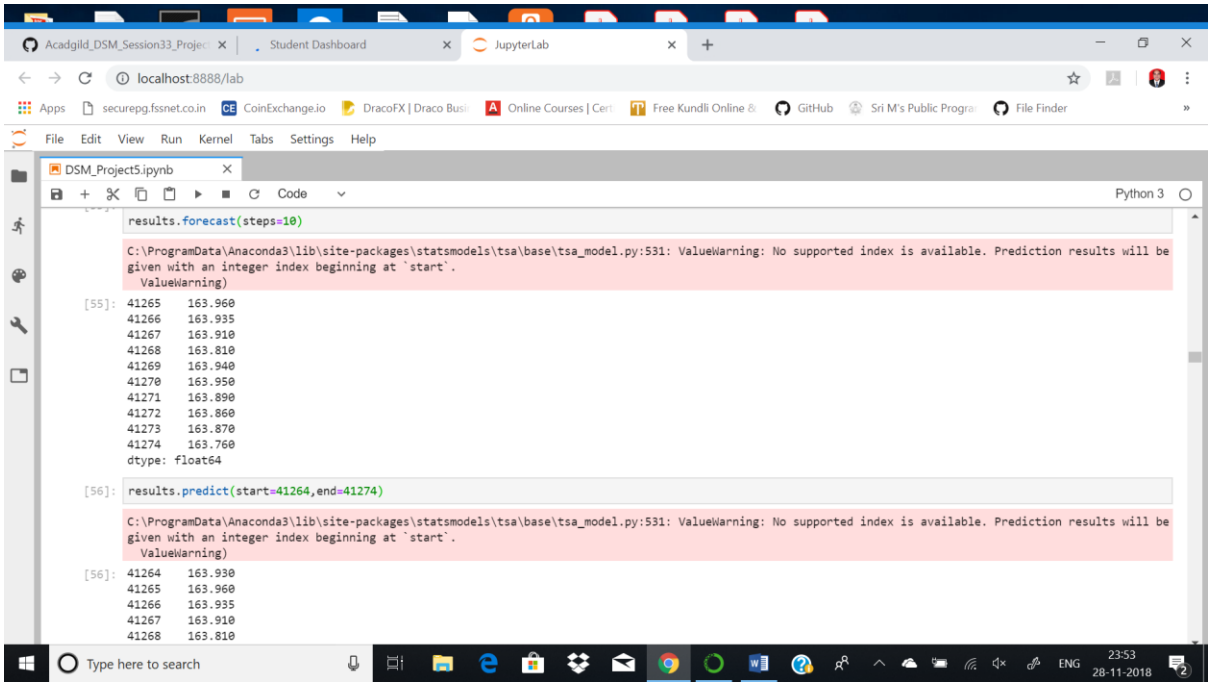


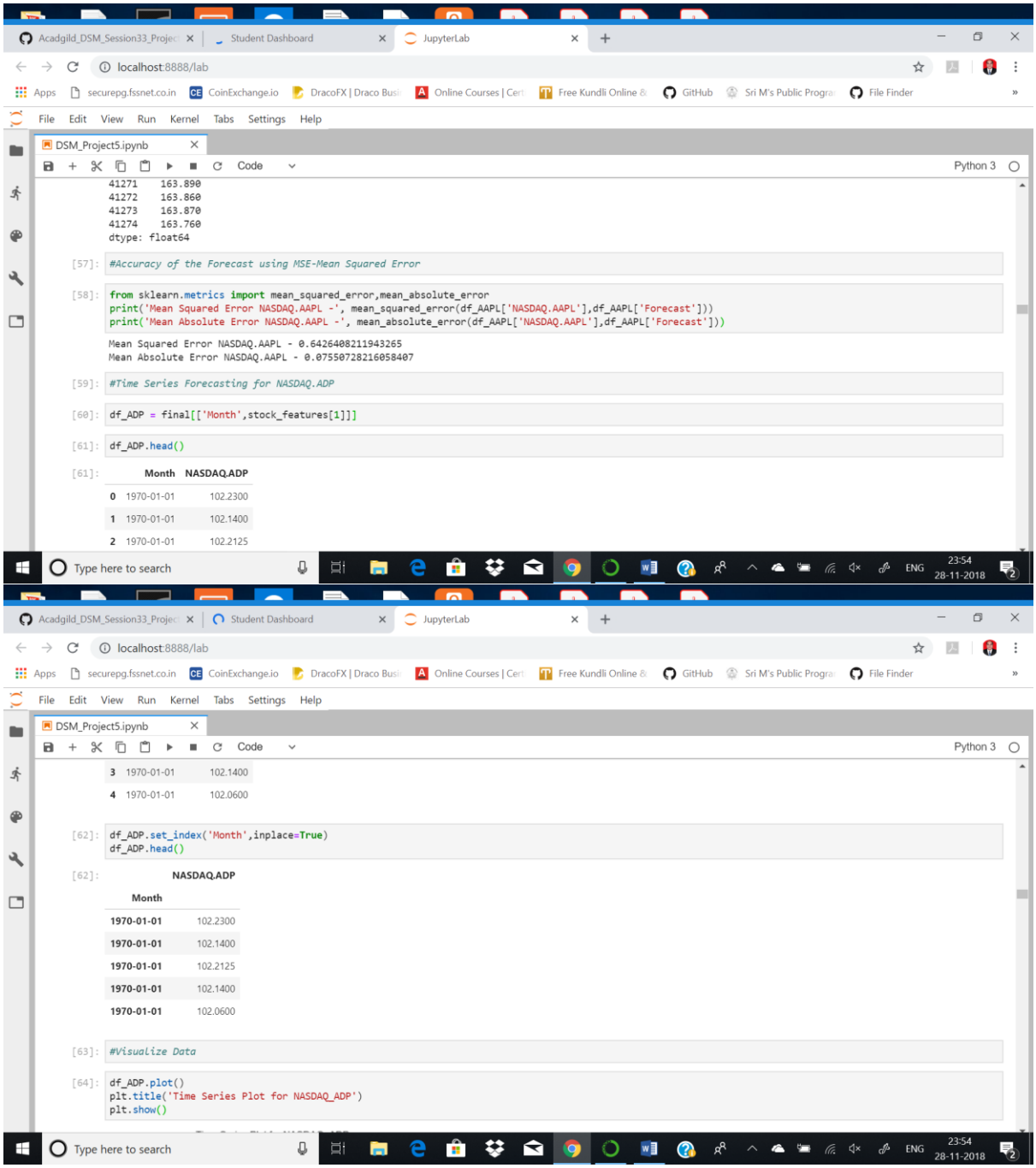
Type here to search | ENG 23:52 28-11-2018


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
Density
[51]:
df_AAPL = df_AAPL.copy()
df_AAPL['Forecast'] = results.predict()
[52]:
df_AAPL.head()
[52]:
NASDAQ.AAPL First_Difference Forecast
Month
1970-01-01 143.7000 0.0200 0.0000
Type here to search
ENG
23:53
28-11-2018

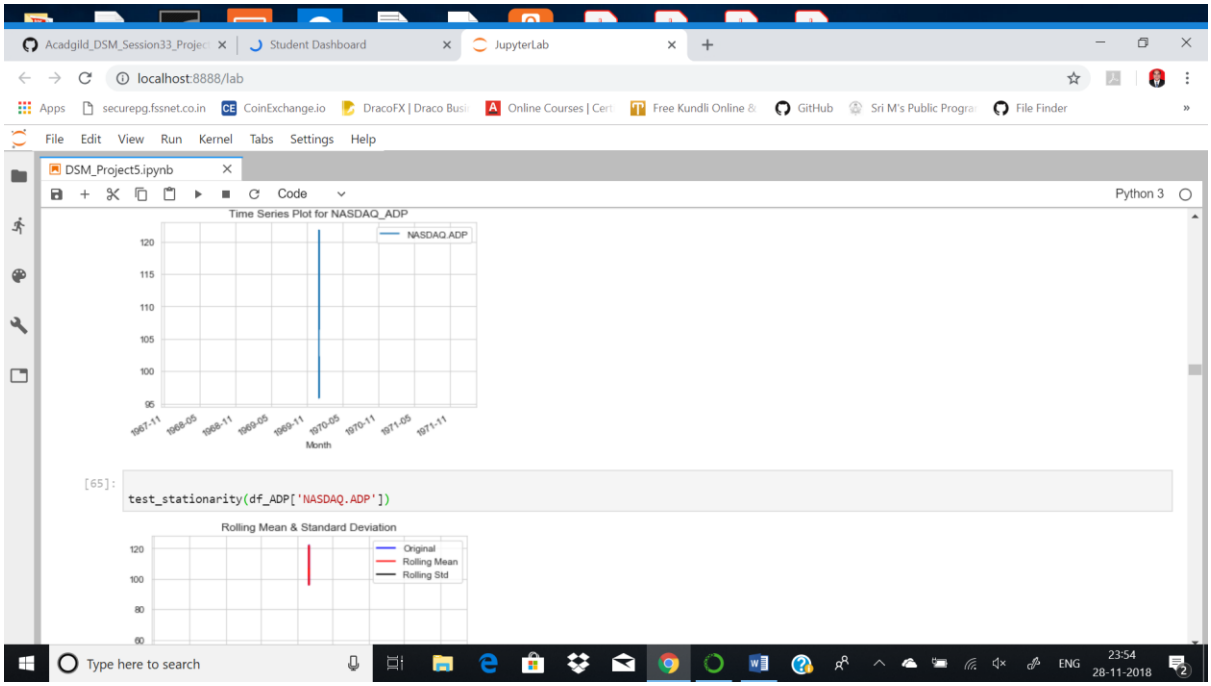

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
[52]:
NASDAQ.AAPL First_Difference Forecast
Month
1970-01-01 143.7000 0.0200 0.0000
1970-01-01 143.6901 -0.0099 143.7000
1970-01-01 143.6400 -0.0501 143.6901
1970-01-01 143.6600 0.0200 143.6400
1970-01-01 143.7800 0.1200 143.6600
[53]:
#Prediction of Future Values
[54]:
df_AAPL[['NASDAQ.AAPL','Forecast']].plot(figsize=(12,8))
[54]:
<matplotlib.axes._subplots.AxesSubplot at 0x2052ec3ee10>
NASDAQ.AAPL
Forecast
200
Type here to search
ENG
23:53
28-11-2018

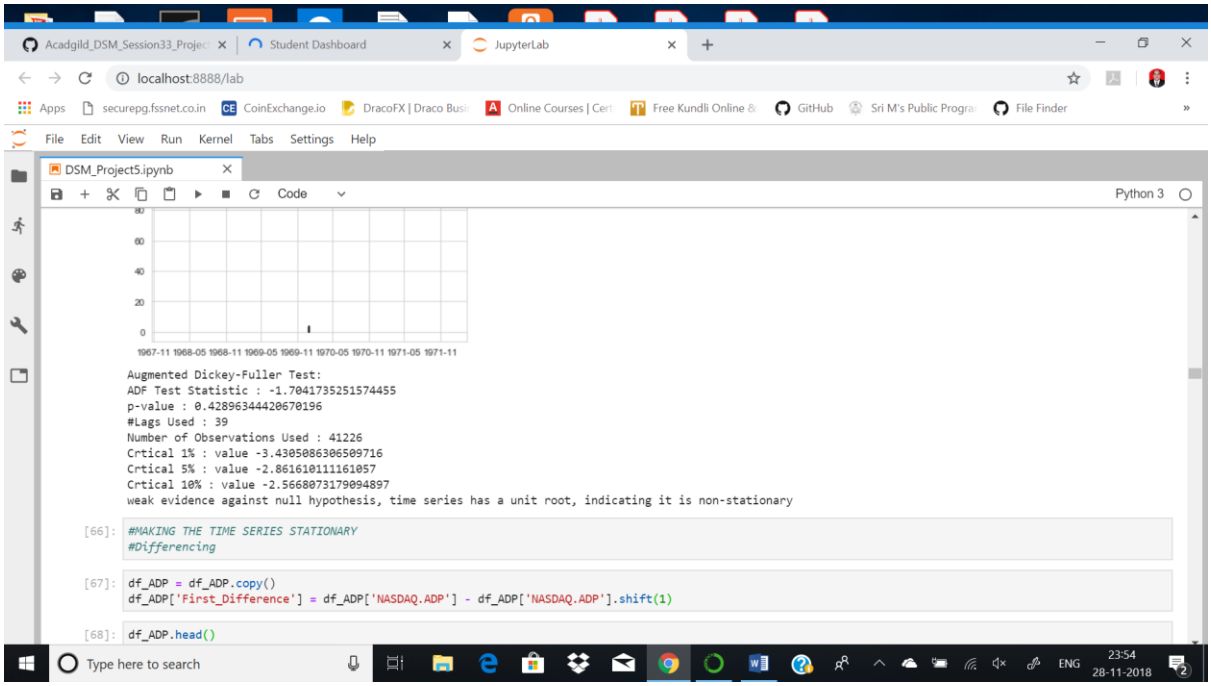

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
NASDAQ.AAPL
Forecast
200
150
100
50
0
1967-11
1968-05
1968-11
1969-05
1969-11
1970-05
1970-11
1971-05
1971-11
Month
Type here to search
ENG
23:53
28-11-2018


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
results.forecast(steps=10)
C:\ProgramData\Anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:531: ValueWarning: No supported index is available. Prediction results will be given with an integer index beginning at `start`.
ValueWarning)
[55]:
41265 163.960
41266 163.935
41267 163.910
41268 163.810
41269 163.940
41270 163.950
41271 163.890
41272 163.860
41273 163.870
41274 163.760
dtype: float64
[56]:
results.predict(start=41264,end=41274)
C:\ProgramData\Anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:531: ValueWarning: No supported index is available. Prediction results will be given with an integer index beginning at `start`.
ValueWarning)
[56]:
41264 163.930
41265 163.960
41266 163.935
41267 163.910
41268 163.810
Type here to search
ENG
23:53
28-11-2018


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
41271 163.890
41272 163.860
41273 163.870
41274 163.760
dtype: float64
[57]: #Accuracy of the Forecast using MSE-Mean Squared Error
[58]: from sklearn.metrics import mean_squared_error,mean_absolute_error
print('Mean Squared Error NASDAQ.AAPL -', mean_squared_error(df_AAPL['NASDAQ.AAPL'],df_AAPL['Forecast']))
print('Mean Absolute Error NASDAQ.AAPL -', mean_absolute_error(df_AAPL['NASDAQ.AAPL'],df_AAPL['Forecast']))
Mean Squared Error NASDAQ.AAPL - 0.6426408211943265
Mean Absolute Error NASDAQ.AAPL - 0.0755072821605840
[59]: #Time Series Forecasting for NASDAQ.ADP
[60]: df_ADP = final[['Month',stock_features[1]]]
[61]: df_ADP.head()
[61]:
Month NASDAQ.ADP
0 1970-01-01 102.2300
1 1970-01-01 102.1400
2 1970-01-01 102.2125
Type here to search
ENG
23:54
28-11-2018

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
3 1970-01-01 102.1400
4 1970-01-01 102.0600
[62]: df_ADP.set_index('Month',inplace=True)
df_ADP.head()
[62]:
NASDAQ.ADP
Month
1970-01-01 102.2300
1970-01-01 102.1400
1970-01-01 102.2125
1970-01-01 102.1400
1970-01-01 102.0600
[63]: #Visualize Data
[64]: df_ADP.plot()
plt.title('Time Series Plot for NASDAQ_ADP')
plt.show()
Type here to search
ENG
23:54
28-11-2018


Time Series Plot for NASDAQ_ADP
NASDAQ.ADP
Month
[65]:
test_stationarity(df_ADP['NASDAQ.ADP'])
Rolling Mean & Standard Deviation
Original
Rolling Mean
Rolling Std


Augmented Dickey-Fuller Test:
ADF Test Statistic : -1.7041735251574455
p-value : 0.42896344420670196
#Lags Used : 39
Number of Observations Used : 41226
Crtical 1% : value -3.430508306509716
Crtical 5% : value -2.861610111161057
Crtical 10% : value -2.5668073179094897
weak evidence against null hypothesis, time series has a unit root, indicating it is non-stationary
[66]:
#MAKING THE TIME SERIES STATIONARY
#Differencing
[67]:
df_ADP = df_ADP.copy()
df_ADP['First_Difference'] = df_ADP['NASDAQ.ADP'] - df_ADP['NASDAQ.ADP'].shift(1)
[68]:
df_ADP.head()

Acadgild_DSM_Session33_Projec | Student Dashboard | JupyterLab

localhost:8888/lab

Apps | securepg.fssnet.co.in | CoinExchange.io | DracoFX | Draco Busi | Online Courses | Cert | Free Kundli Online & | GitHub | Sri M's Public Progra | File Finder

File Edit View Run Kernel Tabs Settings Help

DSM_Project5.ipynb

Code | Python 3

[68]:
```
df_ADP.head()
```

[68]:

Month	NASDAQ.ADP	First_Difference
1970-01-01	102.2300	NaN
1970-01-01	102.1400	-0.0900
1970-01-01	102.2125	0.0725
1970-01-01	102.1400	-0.0725
1970-01-01	102.0600	-0.0800

[69]:
```
df_ADP.dropna(inplace=True)
```

[70]:
```
test_stationarity(df_ADP['First_Difference'])
#Now subtract the rolling mean from the original series
```

Type here to search

ENG 23:55 28-11-2018

Acadgild_DSM_Session33_Projec | Student Dashboard | JupyterLab

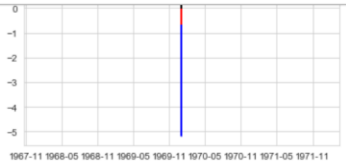
localhost:8888/lab

Apps | securepg.fssnet.co.in | CoinExchange.io | DracoFX | Draco Busi | Online Courses | Cert | Free Kundli Online & | GitHub | Sri M's Public Progra | File Finder

File Edit View Run Kernel Tabs Settings Help

DSM_Project5.ipynb

Code | Python 3




```
Augmented Dickey-Fuller Test:
ADF Test Statistic : -31.055662244632277
p-value : 0.0
#Lags Used : 38
Number of Observations Used : 41226
Crtical 1% : value -3.4305086306509716
Crtical 5% : value -2.861610111161057
Crtical 10% : value -2.5668073179094897
strong evidence against the null hypothesis, reject the null hypothesis. Data has no unit root and is stationary
```

[71]:
```
#Seasonal Decomposition
```

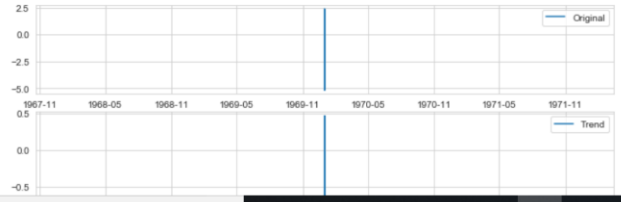
[72]:
```
from statsmodels.tsa.seasonal import seasonal_decompose
plt.figure(figsize=(11,8))
decomposition = seasonal_decompose(df_ADP['First_Difference'],freq=12)
trend = decomposition.trend
seasonal = decomposition.seasonal
```

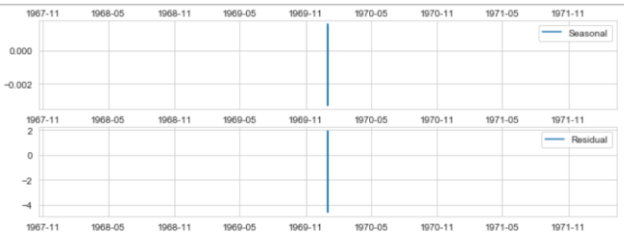
Type here to search

ENG 23:55 28-11-2018

```
residual = decomposition.resid
plt.subplot(411)
plt.plot(df_ADP['First_Difference'],label='Original')
plt.legend(loc='best')
plt.subplot(412)
plt.plot(trend,label='Trend')
plt.legend(loc='best')
plt.subplot(413)
plt.plot(seasonal,label='Seasonal')
plt.legend(loc='best')
plt.subplot(414)
plt.plot(residual,label='Residual')
plt.legend(loc='best')
```

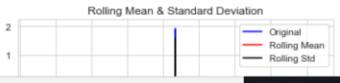
[72]: `<matplotlib.legend.Legend at 0x20500111a20>`

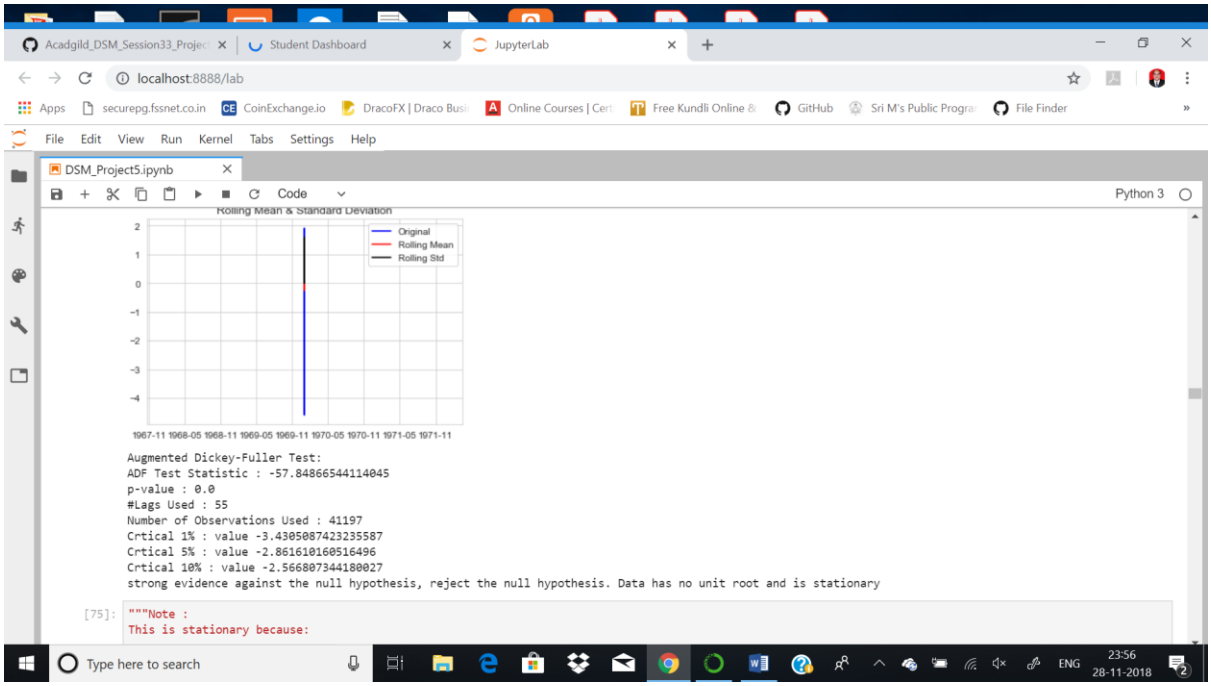




[73]:
```
#Note:
#The data for NASDAQ.ADP is seasonal as interpreted from the seasonal plot of seasonal decomposition.
```

[74]:
```
ts_log_decompose = residual
ts_log_decompose.dropna(inplace=True)
test_stationarity(ts_log_decompose)
```



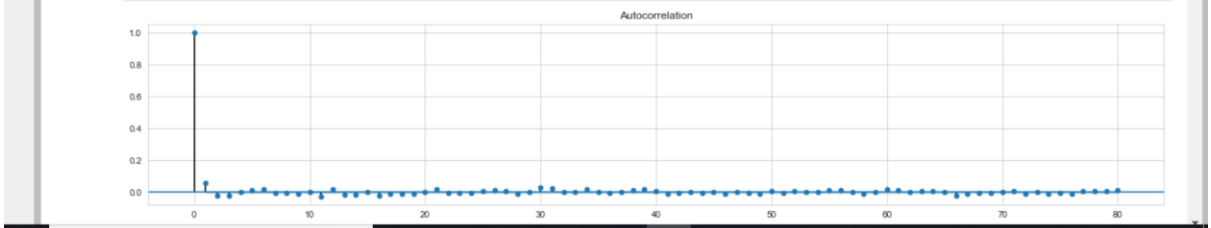


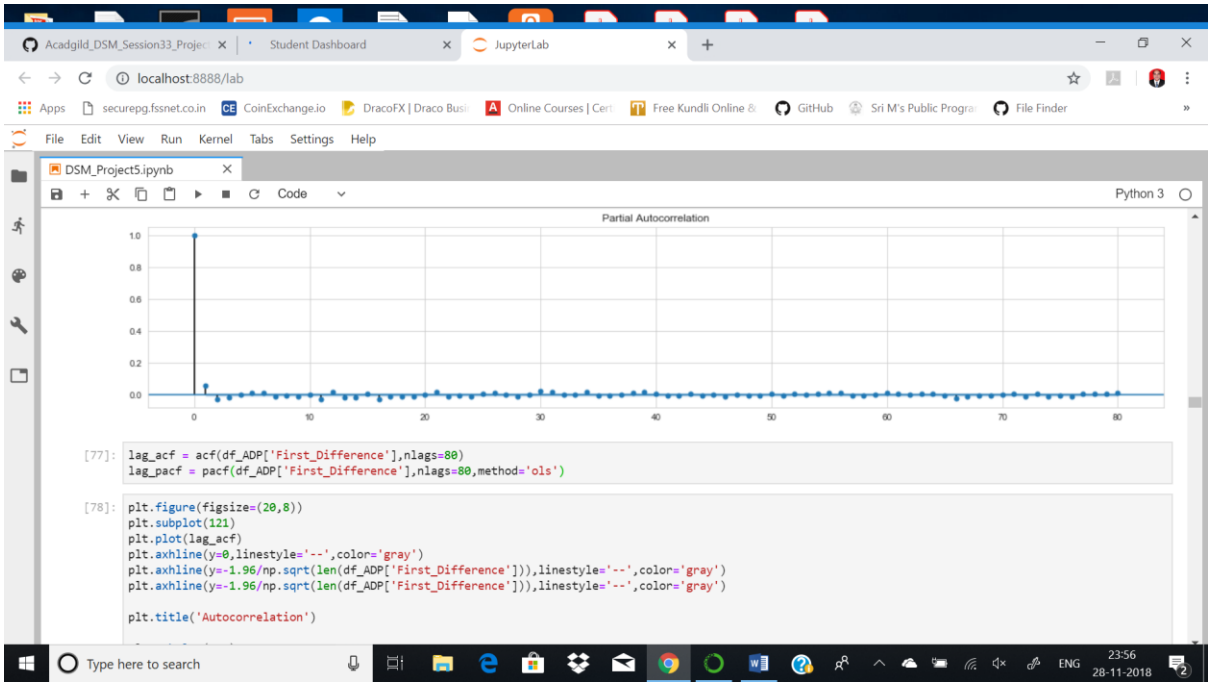

```
Augmented Dickey-Fuller Test:
ADF Test Statistic : -57.84866544114045
p-value : 0.0
#Lags Used : 55
Number of Observations Used : 41197
Crtical 1% : value -3.4305087423235587
Crtical 5% : value -2.861610160516496
Crtical 10% : value -2.566807344180027
strong evidence against the null hypothesis, reject the null hypothesis. Data has no unit root and is stationary
```

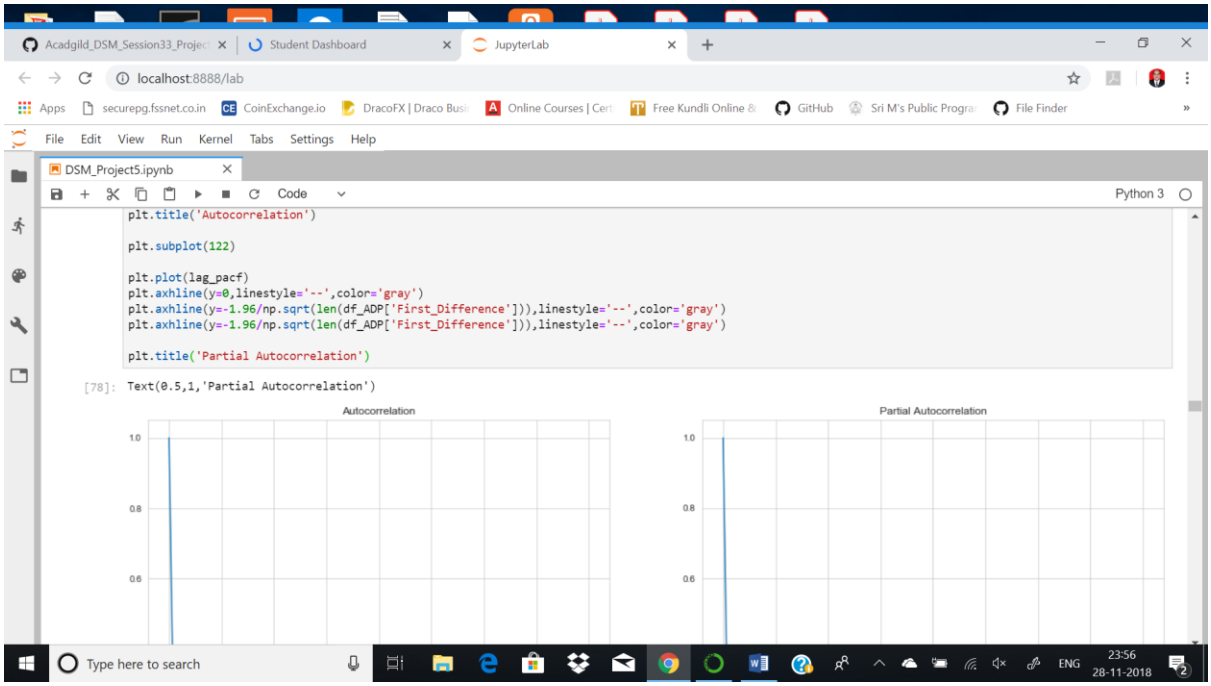
[75]:
```
"""Note :
This is stationary because:

• the mean and std variations have small variations with time

Autocorrelation and Partial Corelation plot"""
```

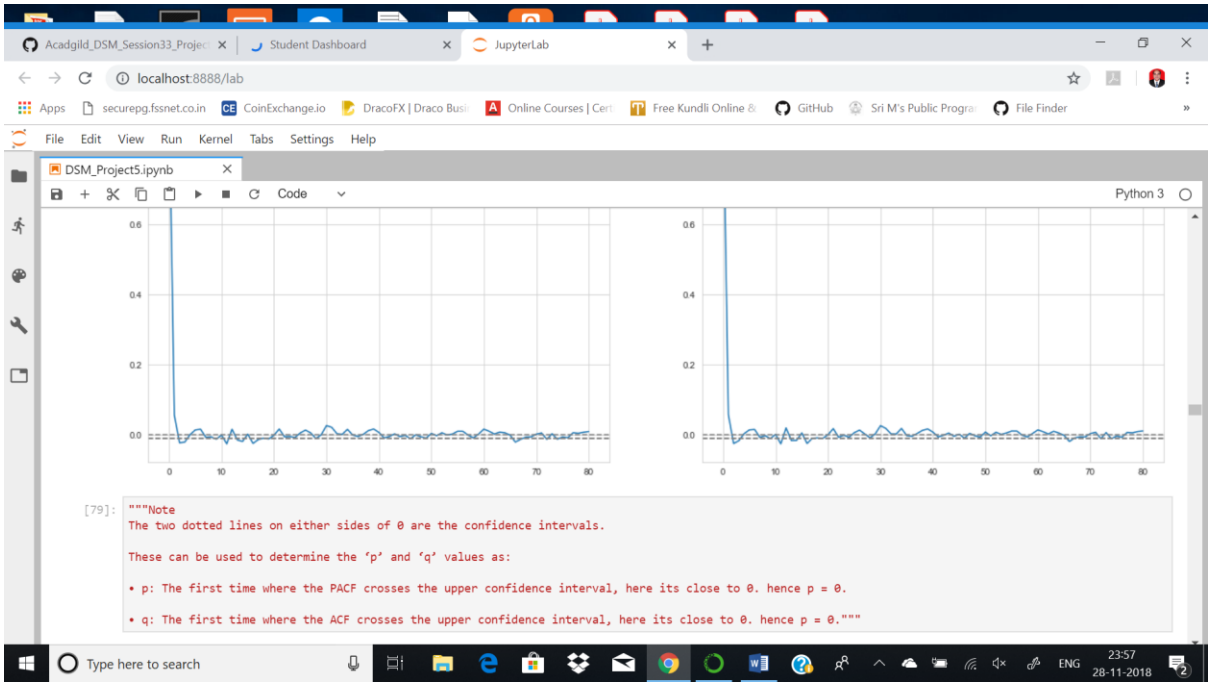
[75]:
```
'Note :\nThis is stationary because:\n\n• test statistic is lower than 1% critical values.\n\n• the mean and std variations have small variations with time
\n\nAutocorrelation and Partial Corelation plot'
```

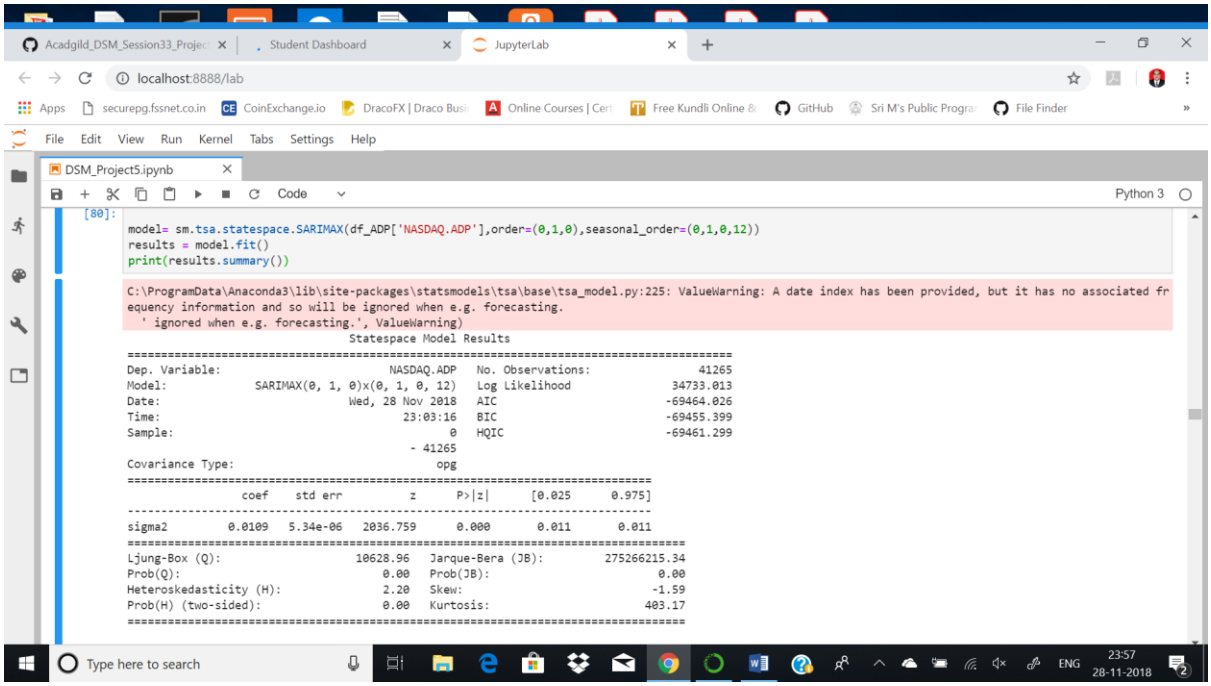
[76]:
```
fig = plt.figure(figsize=(20,8))
ax1 = fig.add_subplot(211)
fig = sm.graphics.tsa.plot_acf(df_ADP['First_Difference'].iloc[38:], lags=80, ax=ax1)
ax2 = fig.add_subplot(212)
fig = sm.graphics.tsa.plot_pacf(df_ADP['First_Difference'].iloc[38:], lags=80, ax=ax2)
```




Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
Apps
securepg.fssnet.co.in
CoinExchange.io
DracoFX | Draco Busi
Online Courses | Cert
Free Kundli Online &
GitHub
Sri M's Public Progra
File Finder
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
Partial Autocorrelation
1.0
0.8
0.6
0.4
0.2
0.0
0
10
20
30
40
50
60
70
80
[77]:
lag_acf = acf(df_ADP['First_Difference'],nlags=80)
lag_pacf = pacf(df_ADP['First_Difference'],nlags=80,method='ols')
[78]:
plt.figure(figsize=(20,8))
plt.subplot(121)
plt.plot(lag_acf)
plt.axhline(y=0,linestyle='--',color='gray')
plt.axhline(y=-1.96/np.sqrt(len(df_ADP['First_Difference'])),linestyle='--',color='gray')
plt.axhline(y=-1.96/np.sqrt(len(df_ADP['First_Difference'])),linestyle='--',color='gray')
plt.title('Autocorrelation')
Type here to search
ENG
23:56
28-11-2018


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
Apps
securepg.fssnet.co.in
CoinExchange.io
DracoFX | Draco Busi
Online Courses | Cert
Free Kundli Online &
GitHub
Sri M's Public Progra
File Finder
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
plt.title('Autocorrelation')
plt.subplot(122)
plt.plot(lag_pacf)
plt.axhline(y=0,linestyle='--',color='gray')
plt.axhline(y=-1.96/np.sqrt(len(df_ADP['First_Difference'])),linestyle='--',color='gray')
plt.axhline(y=-1.96/np.sqrt(len(df_ADP['First_Difference'])),linestyle='--',color='gray')
plt.title('Partial Autocorrelation')
[78]:
Text(0.5,1,'Partial Autocorrelation')
Autocorrelation
Partial Autocorrelation
1.0
0.8
0.6
Type here to search
ENG
23:56
28-11-2018


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
[79]:
"""Note
The two dotted lines on either sides of 0 are the confidence intervals.
These can be used to determine the 'p' and 'q' values as:
• p: The first time where the PACF crosses the upper confidence interval, here its close to 0. hence p = 0.
• q: The first time where the ACF crosses the upper confidence interval, here its close to 0. hence p = 0."""
Type here to search
ENG
23:57
28-11-2018


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
[80]:
model= sm.tsa.statespace.SARIMAX(df_ADP['NASDAQ.ADP'],order=(0,1,0),seasonal_order=(0,1,0,12))
results = model.fit()
print(results.summary())
C:\ProgramData\Anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:225: ValueWarning: A date index has been provided, but it has no associated frequency information and so will be ignored when e.g. forecasting.
' ignored when e.g. forecasting.', ValueWarning)
Statespace Model Results
Dep. Variable: NASDAQ.ADP No. Observations: 41265
Model: SARIMAX(0, 1, 0)x(0, 1, 0, 12) Log Likelihood 34733.013
Date: Wed, 28 Nov 2018 AIC -69464.026
Time: 23:03:16 BIC -69455.399
Sample: 0 HQIC -69461.299
- 41265
Covariance Type: opg
coef std err z P>|z| [0.025 0.975]
sigma2 0.0109 5.34e-06 2036.759 0.000 0.011 0.011
Ljung-Box (Q): 10628.96 Jarque-Bera (JB): 275266215.34
Prob(Q): 0.00 Prob(JB): 0.00
Heteroskedasticity (H): 2.20 Skew: -1.59
Prob(H) (two-sided): 0.00 Kurtosis: 403.17
Type here to search
ENG
23:57
28-11-2018

Acadgild_DSM_Session33_Projec | Student Dashboard | JupyterLab

localhost:8888/lab

Apps | securepg.fssnet.co.in | CoinExchange.io | DracoFX | Draco Busi | Online Courses | Cert | Free Kundli Online & | GitHub | Sri M's Public Progra | File Finder

File Edit View Run Kernel Tabs Settings Help

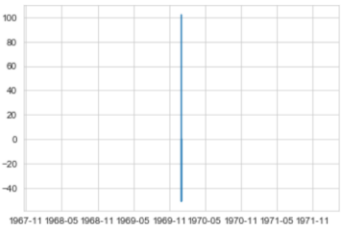
DSM_Project5.ipynb

Code | Python 3

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

[81]:
```
plt.plot(results.resid)
```

[81]: [<matplotlib.lines.Line2D at 0x20500a5cc50>]



[82]:
```
import seaborn as sns
sns.set_style('whitegrid')
sns.kdeplot(results.resid)
```

Type here to search

ENG 23:57 28-11-2018

Acadgild_DSM_Session33_Projec | Student Dashboard | JupyterLab

localhost:8888/lab

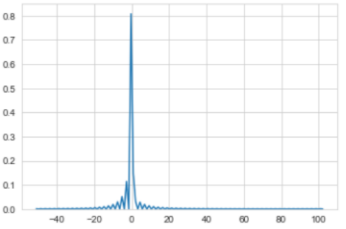
Apps | securepg.fssnet.co.in | CoinExchange.io | DracoFX | Draco Busi | Online Courses | Cert | Free Kundli Online & | GitHub | Sri M's Public Progra | File Finder

File Edit View Run Kernel Tabs Settings Help

DSM_Project5.ipynb

Code | Python 3

[82]: <matplotlib.axes._subplots.AxesSubplot at 0x20500a46d30>



[83]:
```
df_ADP['Forecast'] = results.predict()
```

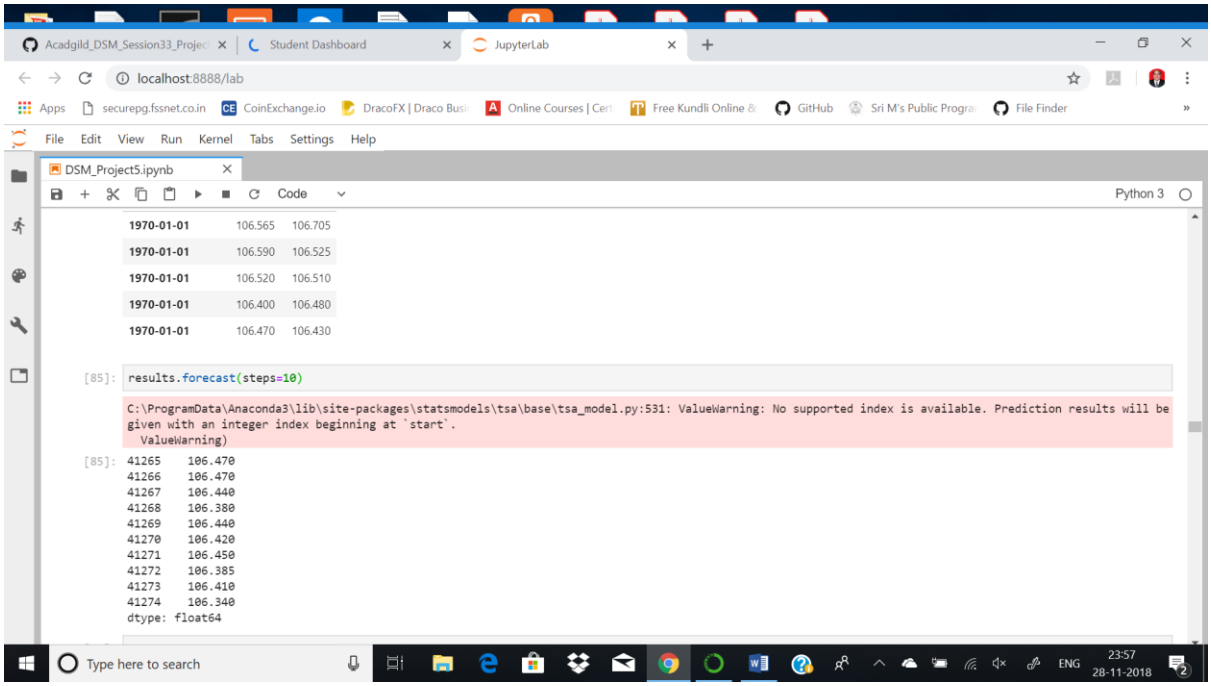
[84]:
```
df_ADP[['NASDAQ.ADP','Forecast']].tail()
```

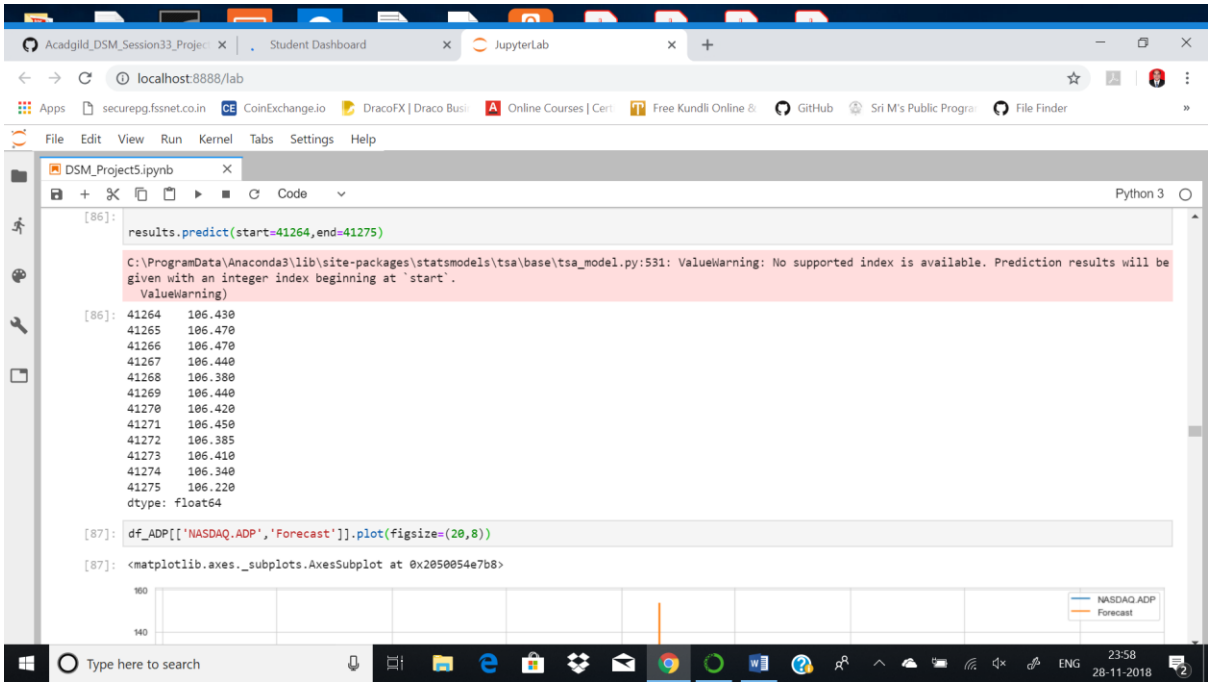
[84]:

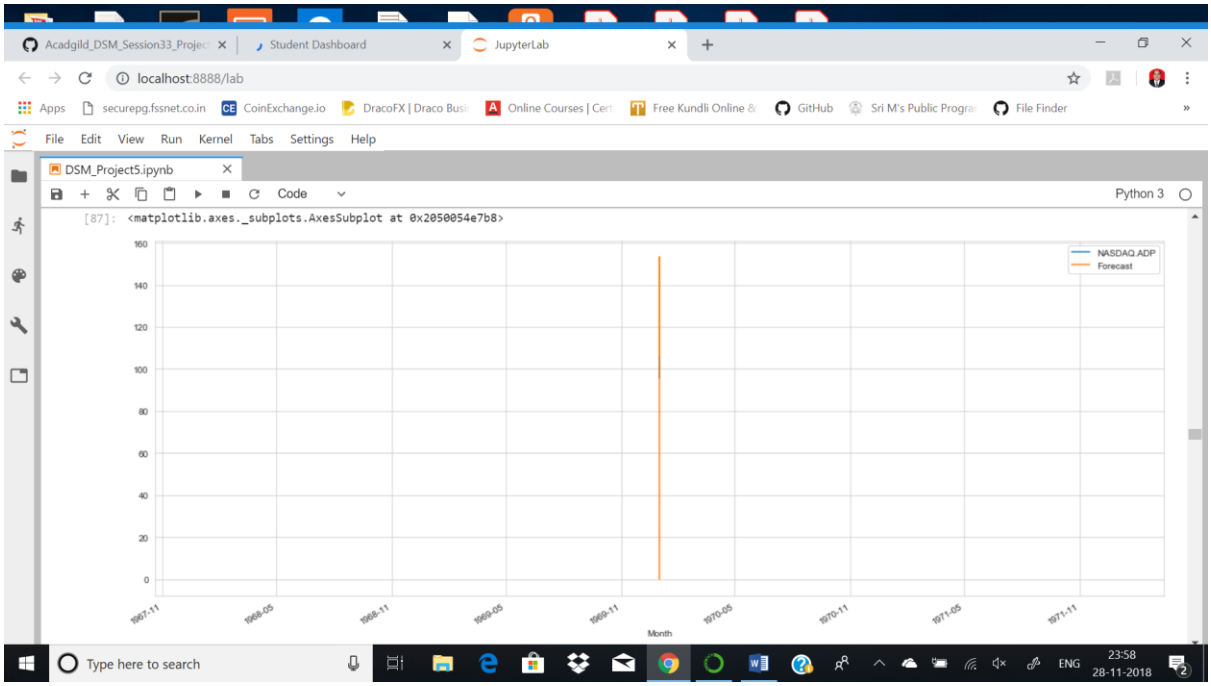
Month	NASDAQ.ADP	Forecast
1970-01-01	106.565	106.705

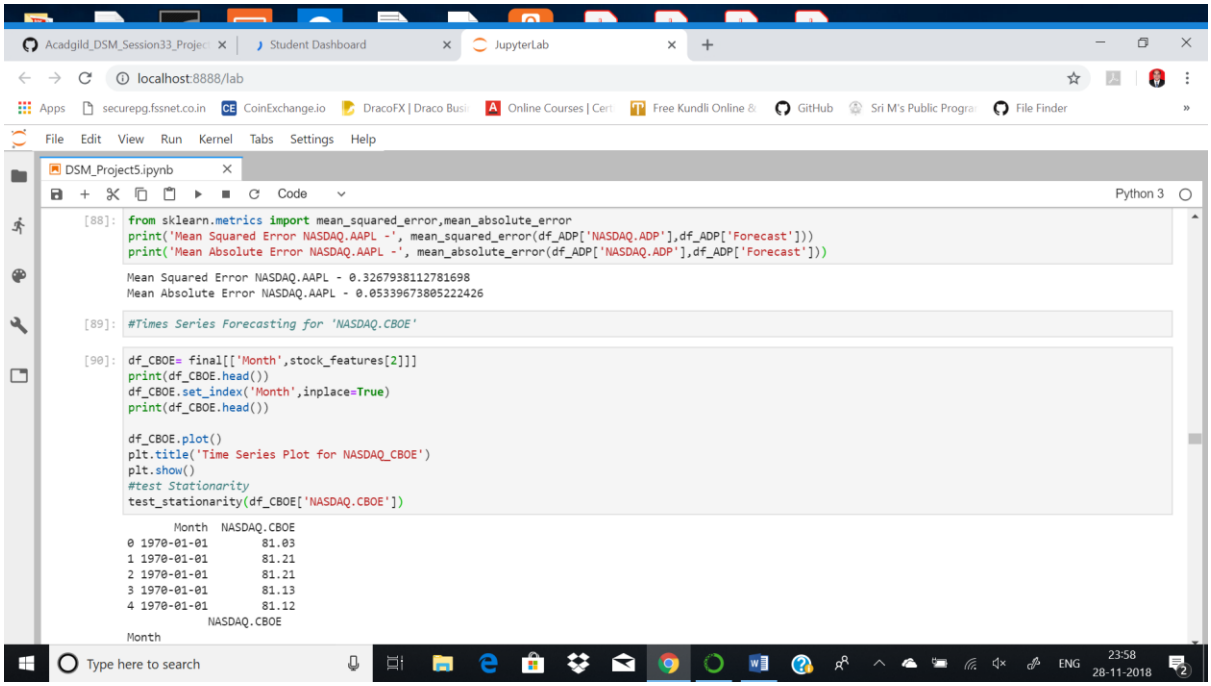
Type here to search

ENG 23:57 28-11-2018

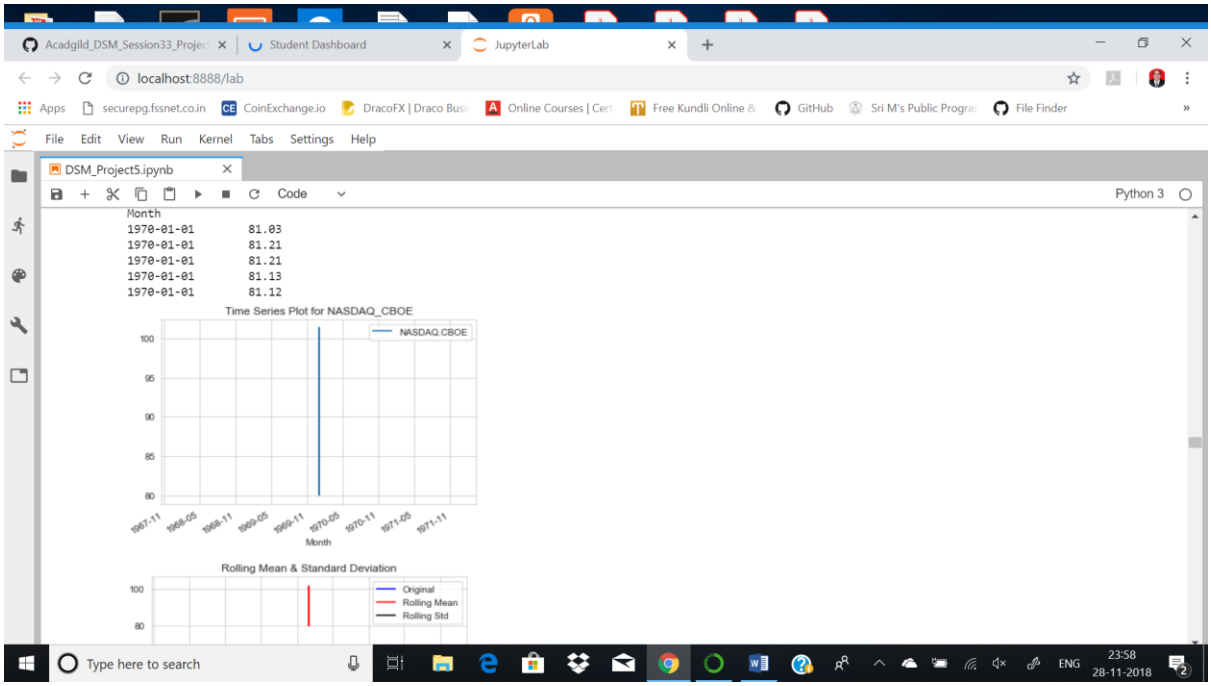

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
1970-01-01 106.565 106.705
1970-01-01 106.590 106.525
1970-01-01 106.520 106.510
1970-01-01 106.400 106.480
1970-01-01 106.470 106.430
[85]: results.forecast(steps=10)
C:\ProgramData\Anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:531: ValueWarning: No supported index is available. Prediction results will be given with an integer index beginning at `start`.
ValueWarning)
[85]: 41265 106.470
41266 106.470
41267 106.440
41268 106.380
41269 106.440
41270 106.420
41271 106.450
41272 106.385
41273 106.410
41274 106.340
dtype: float64
Type here to search
ENG
23:57
28-11-2018

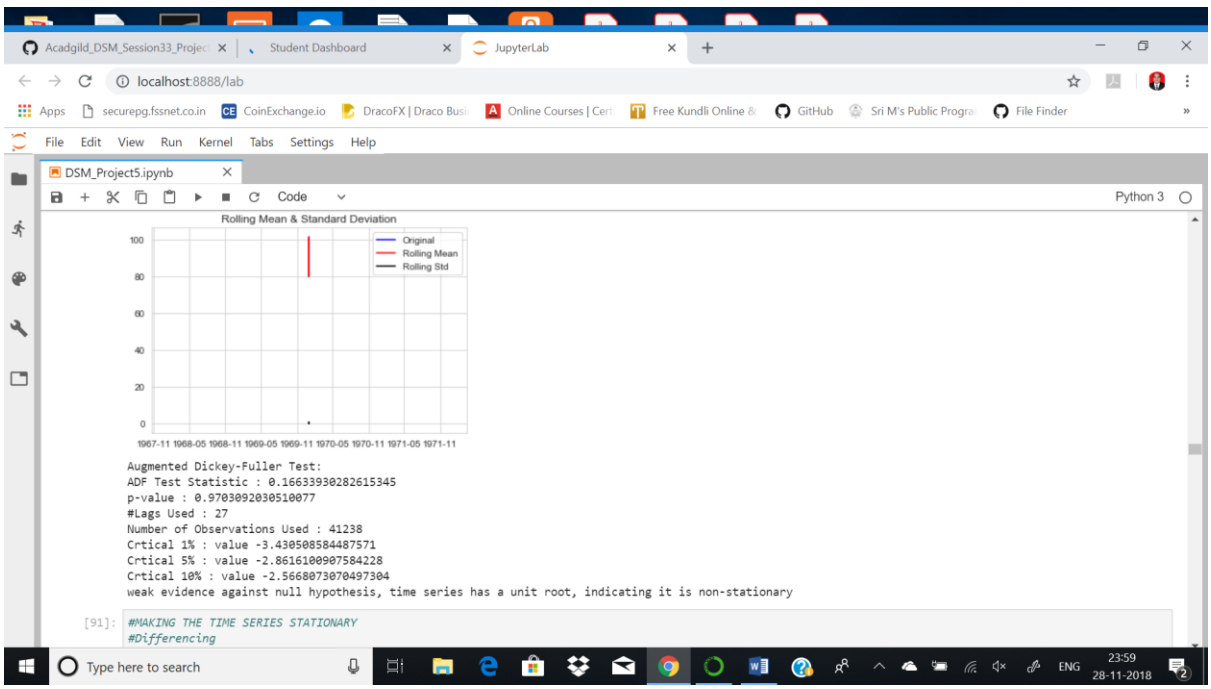

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
[86]: results.predict(start=41264,end=41275)
C:\ProgramData\Anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:531: ValueWarning: No supported index is available. Prediction results will be given with an integer index beginning at `start`.
ValueWarning)
[86]: 41264 106.430
41265 106.470
41266 106.470
41267 106.440
41268 106.380
41269 106.440
41270 106.420
41271 106.450
41272 106.385
41273 106.410
41274 106.340
41275 106.220
dtype: float64
[87]: df_ADP[['NASDAQ.ADP','Forecast']].plot(figsize=(20,8))
[87]: <matplotlib.axes._subplots.AxesSubplot at 0x2050054e7b8>
160
140
NASDAQ.ADP
Forecast
Type here to search
ENG
23:58
28-11-2018






```
[88]: from sklearn.metrics import mean_squared_error,mean_absolute_error
      print('Mean Squared Error NASDAQ.AAPL -', mean_squared_error(df_ADP['NASDAQ.ADP'],df_ADP['Forecast']))
      print('Mean Absolute Error NASDAQ.AAPL -', mean_absolute_error(df_ADP['NASDAQ.ADP'],df_ADP['Forecast']))

Mean Squared Error NASDAQ.AAPL - 0.3267938112781698
Mean Absolute Error NASDAQ.AAPL - 0.0533967380522242 6

[89]: #Times Series Forecasting for 'NASDAQ.CBOE'

[90]: df_CBOE= final[['Month',stock_features[2]]]
      print(df_CBOE.head())
      df_CBOE.set_index('Month',inplace=True)
      print(df_CBOE.head())

      df_CBOE.plot()
      plt.title('Time Series Plot for NASDAQ_CBOE')
      plt.show()
      #test Stationarity
      test_stationarity(df_CBOE['NASDAQ.CBOE'])

        Month  NASDAQ.CBOE
0 1970-01-01        81.03
1 1970-01-01        81.21
2 1970-01-01        81.21
3 1970-01-01        81.13
4 1970-01-01        81.12
             NASDAQ.CBOE
Month
```


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
Month
1970-01-01 81.03
1970-01-01 81.21
1970-01-01 81.21
1970-01-01 81.13
1970-01-01 81.12
Time Series Plot for NASDAQ_CBOE
NASDAQ.CBOE
Month
Rolling Mean & Standard Deviation
Original
Rolling Mean
Rolling Std
Type here to search
ENG
23:58
28-11-2018


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
Rolling Mean & Standard Deviation
Original
Rolling Mean
Rolling Std
Augmented Dickey-Fuller Test:
ADF Test Statistic : 0.16633930282615345
p-value : 0.9703092030510077
#Lags Used : 27
Number of Observations Used : 41238
Crtical 1% : value -3.430508584487571
Crtical 5% : value -2.8616100907584228
Crtical 10% : value -2.5668073070497304
weak evidence against null hypothesis, time series has a unit root, indicating it is non-stationary
[91]: #MAKING THE TIME SERIES STATIONARY
#Differencing
Type here to search
ENG
23:59
28-11-2018

Acadgild_DSM_Session33_Projec | Student Dashboard | JupyterLab

localhost:8888/lab

Apps | securepg.fssnet.co.in | CoinExchange.io | DracoFX | Draco Busi | Online Courses | Cert | Free Kundli Online & | GitHub | Sri M's Public Progra | File Finder

File Edit View Run Kernel Tabs Settings Help

DSM_Project5.ipynb

Code | Python 3

[92]:
```
df_CBOE = df_CBOE.copy()
```

[93]:
```
df_CBOE.head()
```

[93]:

Month	NASDAQ.CBOE
1970-01-01	81.03
1970-01-01	81.21
1970-01-01	81.21
1970-01-01	81.13
1970-01-01	81.12

[94]:
```
df_CBOE['First_Difference'] = df_CBOE['NASDAQ.CBOE'] - df_CBOE['NASDAQ.CBOE'].shift(1)
df_CBOE.head()
```

[94]:

Month	NASDAQ.CBOE	First_Difference
1970-01-01	81.03	NaN
1970-01-01	81.21	0.18

Type here to search | ENG 23:59 28-11-2018

Acadgild_DSM_Session33_Projec | Student Dashboard | JupyterLab

localhost:8888/lab

Apps | securepg.fssnet.co.in | CoinExchange.io | DracoFX | Draco Busi | Online Courses | Cert | Free Kundli Online & | GitHub | Sri M's Public Progra | File Finder

File Edit View Run Kernel Tabs Settings Help

DSM_Project5.ipynb

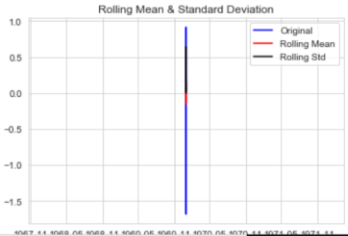
Code | Python 3

Month	NASDAQ.CBOE	First_Difference
1970-01-01	81.21	0.00
1970-01-01	81.13	-0.08
1970-01-01	81.12	-0.01

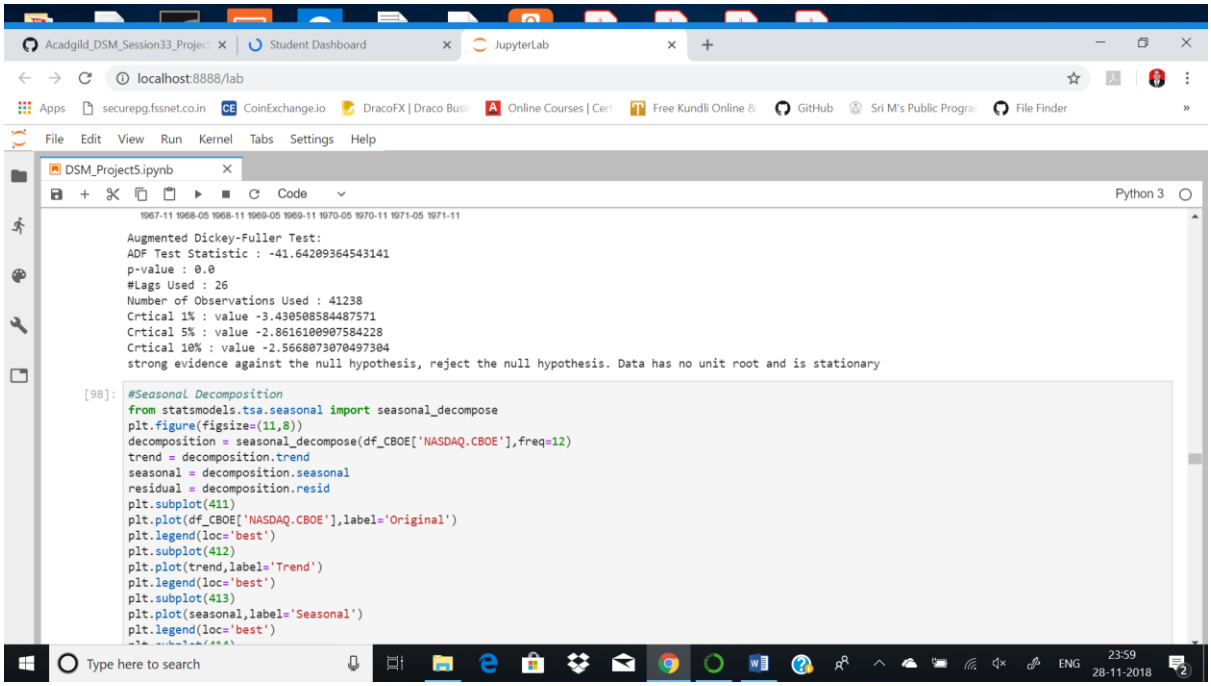
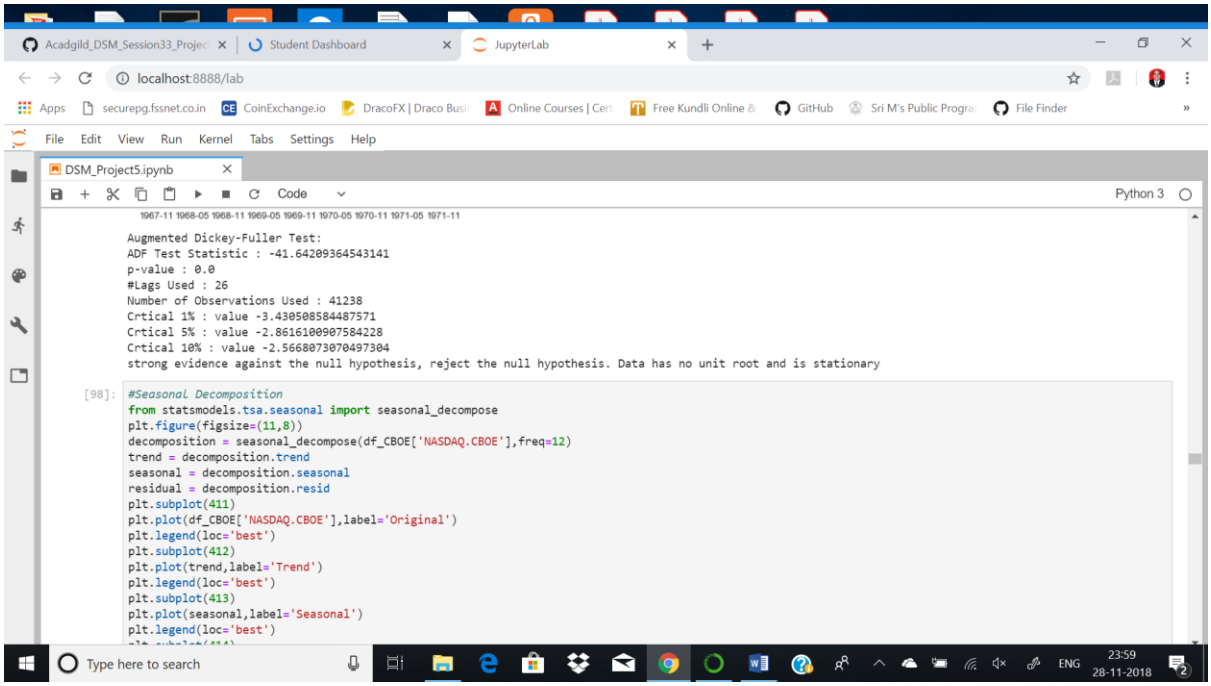
[95]:
```
df_CBOE.dropna(inplace=True)
```

[96]:
```
#Test Seasonality
```

[97]:
```
test_stationarity(df_CBOE['First_Difference'])
```

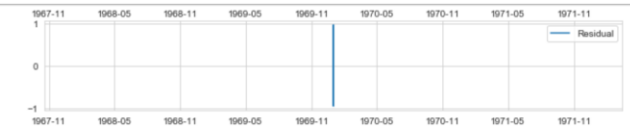


Type here to search | ENG 23:59 28-11-2018

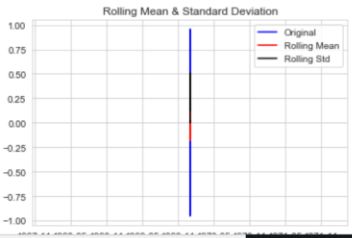

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
Apps
securepg.fssnet.co.in
CoinExchange.io
DracoFX | Draco Busi
Online Courses | Cert
Free Kundli Online &
GitHub
Sri M's Public Progra
File Finder
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
1967-11 1968-05 1968-11 1969-05 1969-11 1970-05 1970-11 1971-05 1971-11
Augmented Dickey-Fuller Test:
ADF Test Statistic : -41.64209364543141
p-value : 0.0
#Lags Used : 26
Number of Observations Used : 41238
Crtical 1% : value -3.430508584487571
Crtical 5% : value -2.8616100907584228
Crtical 10% : value -2.5668073070497304
strong evidence against the null hypothesis, reject the null hypothesis. Data has no unit root and is stationary
[98]:
#Seasonal Decomposition
from statsmodels.tsa.seasonal import seasonal_decompose
plt.figure(figsize=(11,8))
decomposition = seasonal_decompose(df_CBOE['NASDAQ.CBOE'],freq=12)
trend = decomposition.trend
seasonal = decomposition.seasonal
residual = decomposition.resid
plt.subplot(411)
plt.plot(df_CBOE['NASDAQ.CBOE'],label='Original')
plt.legend(loc='best')
plt.subplot(412)
plt.plot(trend,label='Trend')
plt.legend(loc='best')
plt.subplot(413)
plt.plot(seasonal,label='Seasonal')
plt.legend(loc='best')
Type here to search
ENG
23:59
28-11-2018

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
Apps
securepg.fssnet.co.in
CoinExchange.io
DracoFX | Draco Busi
Online Courses | Cert
Free Kundli Online &
GitHub
Sri M's Public Progra
File Finder
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
1967-11 1968-05 1968-11 1969-05 1969-11 1970-05 1970-11 1971-05 1971-11
Augmented Dickey-Fuller Test:
ADF Test Statistic : -41.64209364543141
p-value : 0.0
#Lags Used : 26
Number of Observations Used : 41238
Crtical 1% : value -3.430508584487571
Crtical 5% : value -2.8616100907584228
Crtical 10% : value -2.5668073070497304
strong evidence against the null hypothesis, reject the null hypothesis. Data has no unit root and is stationary
[98]:
#Seasonal Decomposition
from statsmodels.tsa.seasonal import seasonal_decompose
plt.figure(figsize=(11,8))
decomposition = seasonal_decompose(df_CBOE['NASDAQ.CBOE'],freq=12)
trend = decomposition.trend
seasonal = decomposition.seasonal
residual = decomposition.resid
plt.subplot(411)
plt.plot(df_CBOE['NASDAQ.CBOE'],label='Original')
plt.legend(loc='best')
plt.subplot(412)
plt.plot(trend,label='Trend')
plt.legend(loc='best')
plt.subplot(413)
plt.plot(seasonal,label='Seasonal')
plt.legend(loc='best')
Type here to search
ENG
23:59
28-11-2018

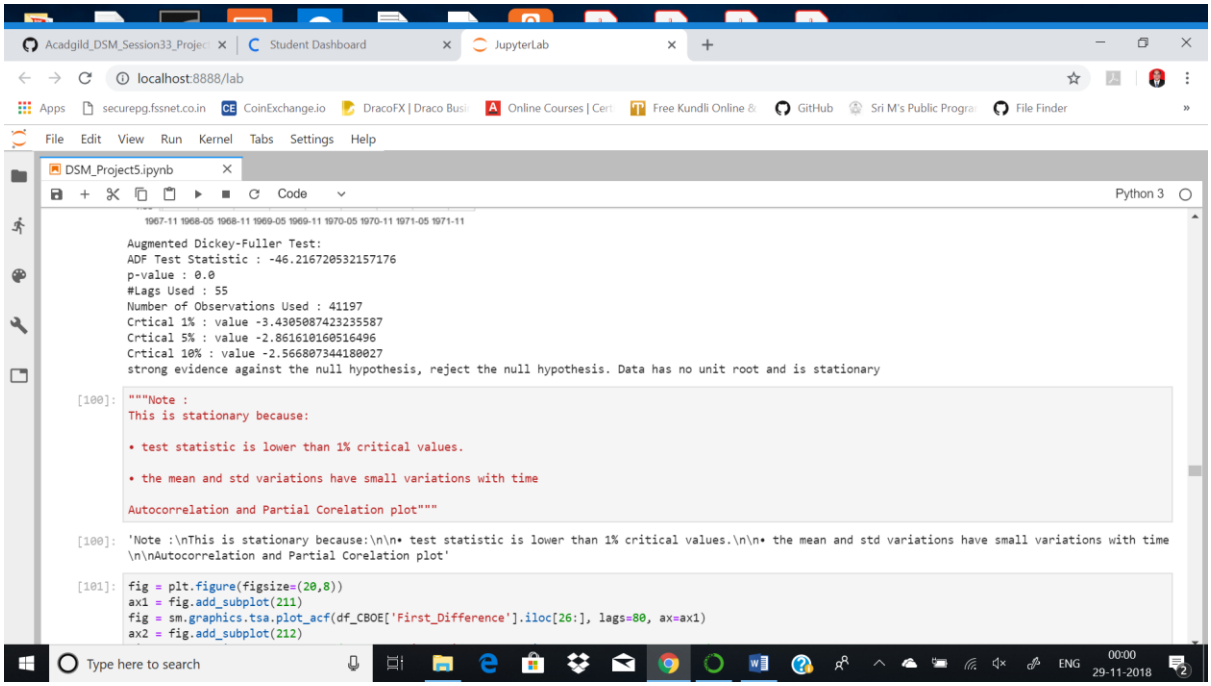
```
plt.subplot(414)
plt.plot(residual,label='Residual')
plt.legend(loc='best')
```

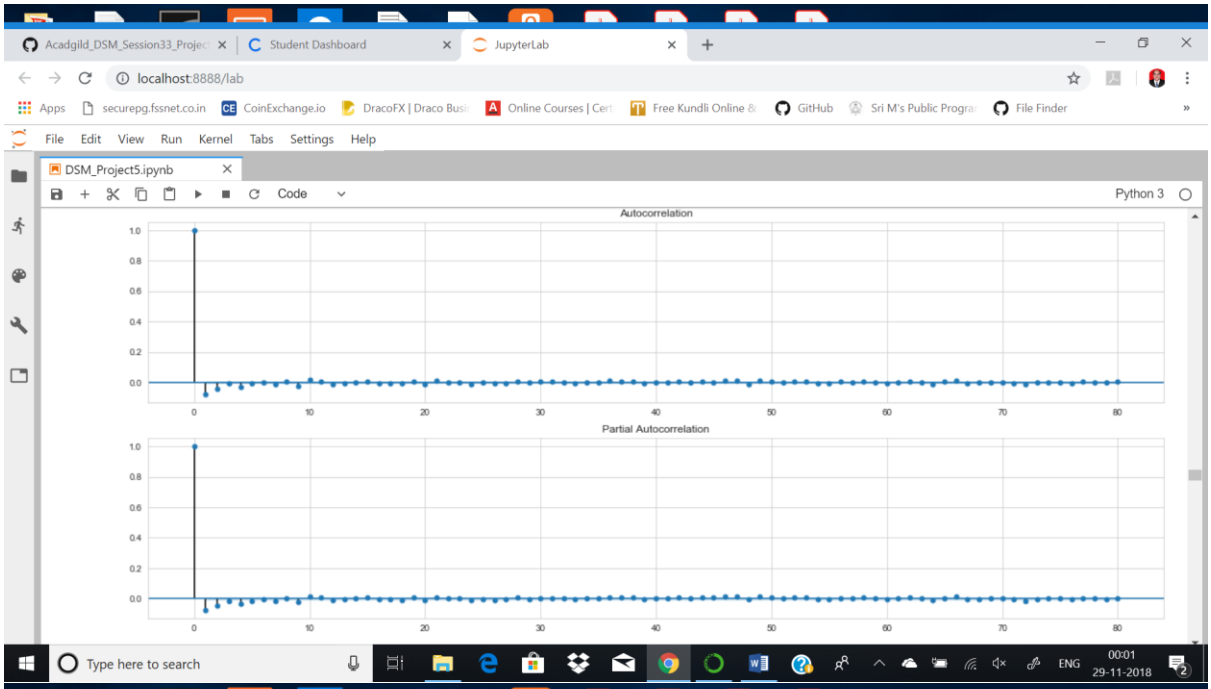
[98]: <matplotlib.legend.Legend at 0x205193fbe10>



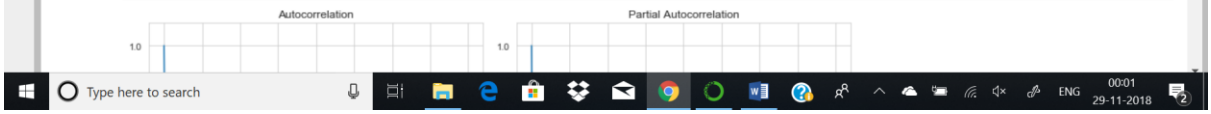
[99]:
```
ts_log_decompose = residual
ts_log_decompose.dropna(inplace=True)
test_stationarity(ts_log_decompose)
```

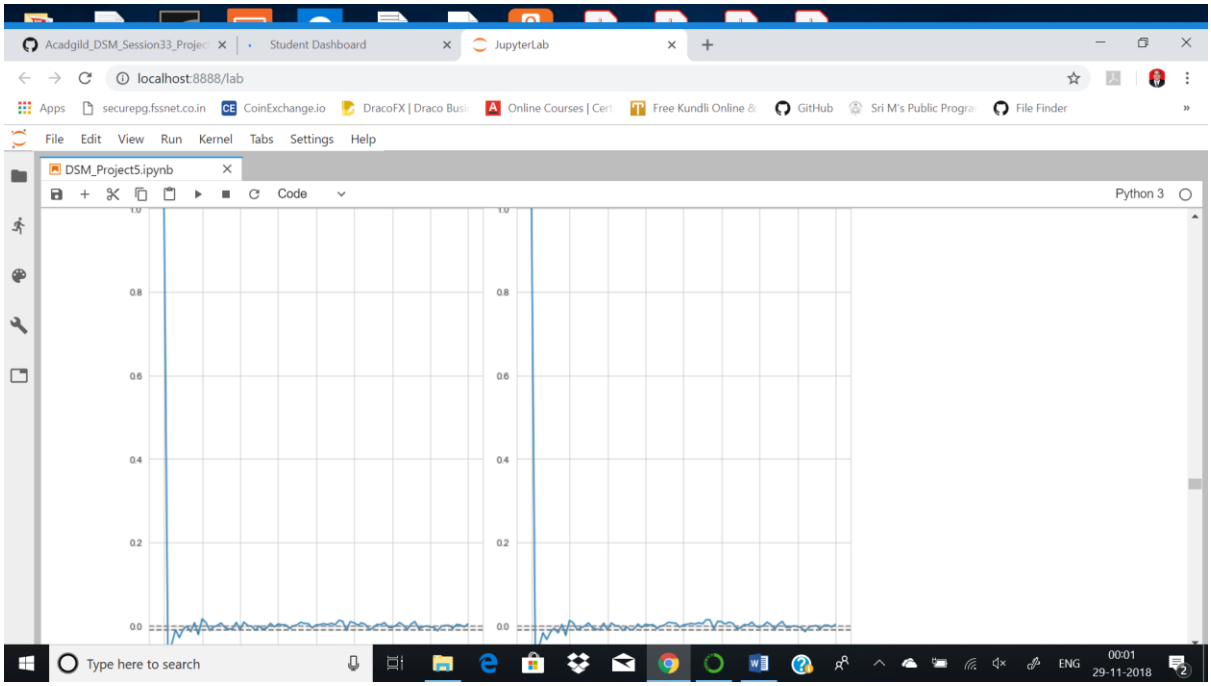


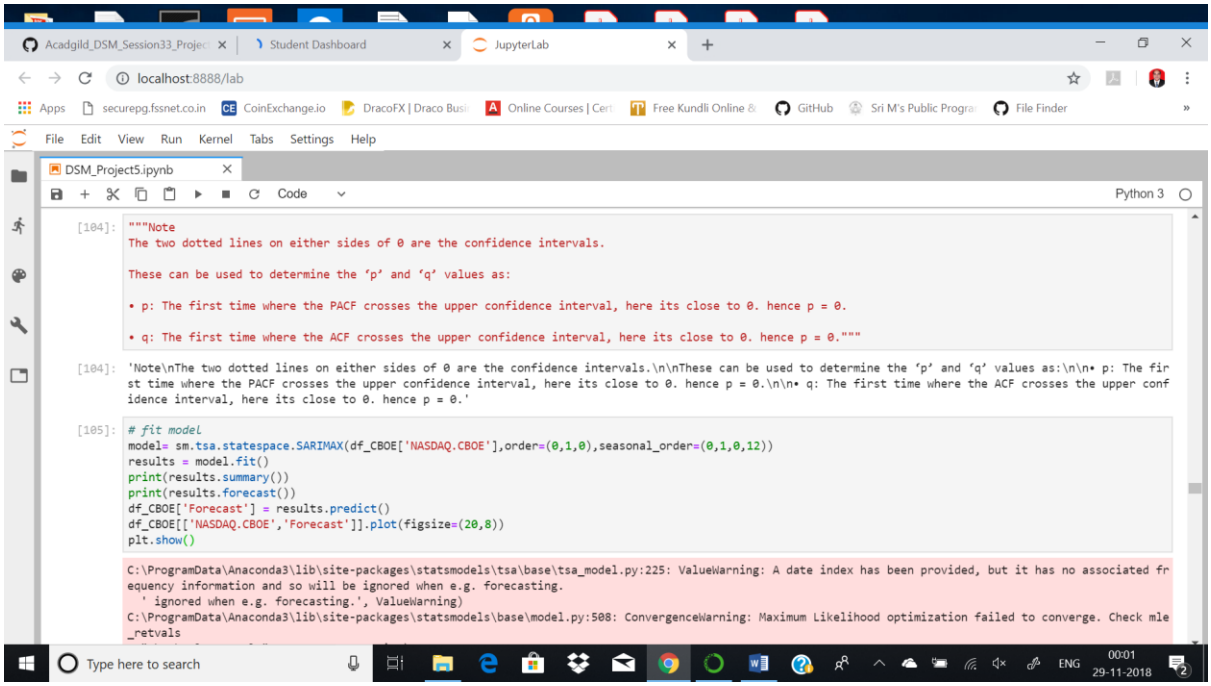

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
Apps
securepg.fssnet.co.in
CoinExchange.io
DracoFX | Draco Busi
Online Courses | Cert
Free Kundli Online &
GitHub
Sri M's Public Progra
File Finder
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
1967-11 1968-05 1968-11 1969-05 1969-11 1970-05 1970-11 1971-05 1971-11
Augmented Dickey-Fuller Test:
ADF Test Statistic : -46.216720532157176
p-value : 0.0
#Lags Used : 55
Number of Observations Used : 41197
Crtical 1% : value -3.4305087423235587
Crtical 5% : value -2.861610160516496
Crtical 10% : value -2.566807344180027
strong evidence against the null hypothesis, reject the null hypothesis. Data has no unit root and is stationary
[100]:
"""Note :
This is stationary because:
• test statistic is lower than 1% critical values.
• the mean and std variations have small variations with time
Autocorrelation and Partial Corelation plot"""
[100]:
'Note :\nThis is stationary because:\n\n• test statistic is lower than 1% critical values.\n\n• the mean and std variations have small variations with time \n\nAutocorrelation and Partial Corelation plot'
[101]:
fig = plt.figure(figsize=(20,8))
ax1 = fig.add_subplot(211)
fig = sm.graphics.tsa.plot_acf(df_CBOE['First_Difference'].iloc[26:], lags=80, ax=ax1)
ax2 = fig.add_subplot(212)
Type here to search
ENG
00:00
29-11-2018

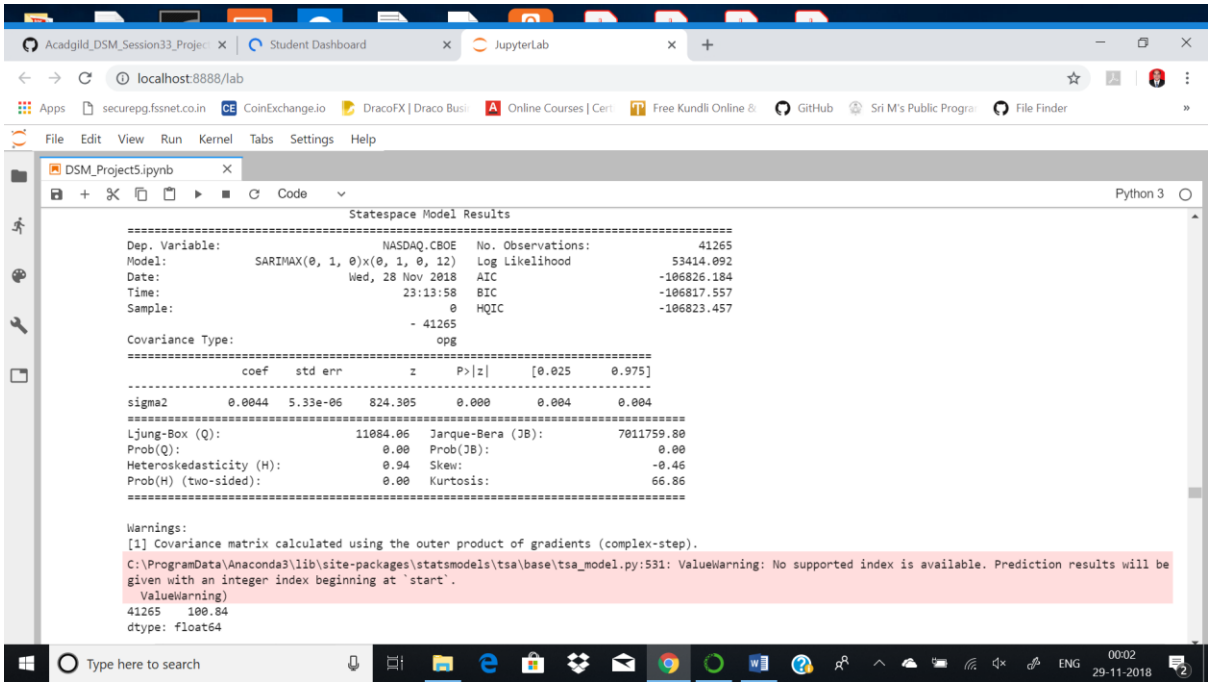


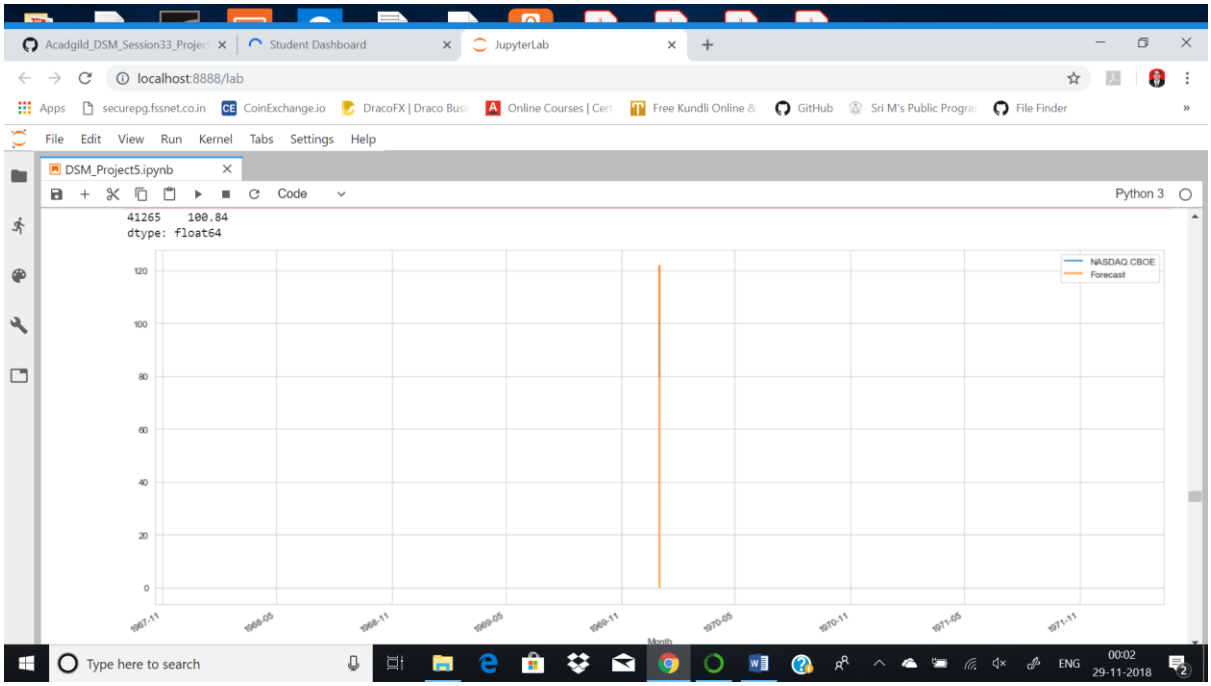

```
[102]: lag_acf = acf(df_CBOE['First_Difference'],nlags=80)
       lag_pacf = pacf(df_CBOE['First_Difference'],nlags=80,method='ols')

[103]: plt.figure(figsize=(11,8))
       plt.subplot(121)
       plt.plot(lag_acf)
       plt.axhline(y=0,linestyle='--',color='gray')
       plt.axhline(y=-1.96/np.sqrt(len(df_CBOE['First_Difference'])),linestyle='--',color='gray')
       plt.axhline(y=-1.96/np.sqrt(len(df_CBOE['First_Difference'])),linestyle='--',color='gray')

       plt.title('Autocorrelation')

       plt.subplot(122)

       plt.plot(lag_pacf)
       plt.axhline(y=0,linestyle='--',color='gray')
       plt.axhline(y=-1.96/np.sqrt(len(df_CBOE['First_Difference'])),linestyle='--',color='gray')
       plt.axhline(y=-1.96/np.sqrt(len(df_CBOE['First_Difference'])),linestyle='--',color='gray')

       plt.title('Partial Autocorrelation')

       plt.tight_layout()
```

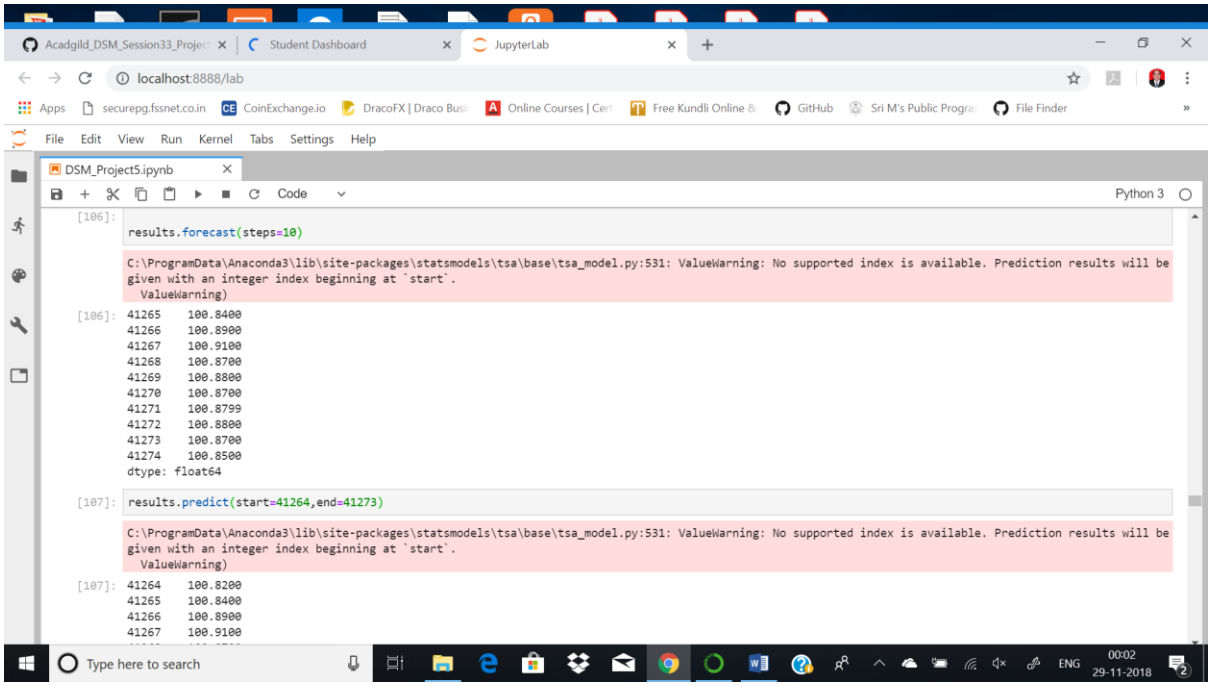


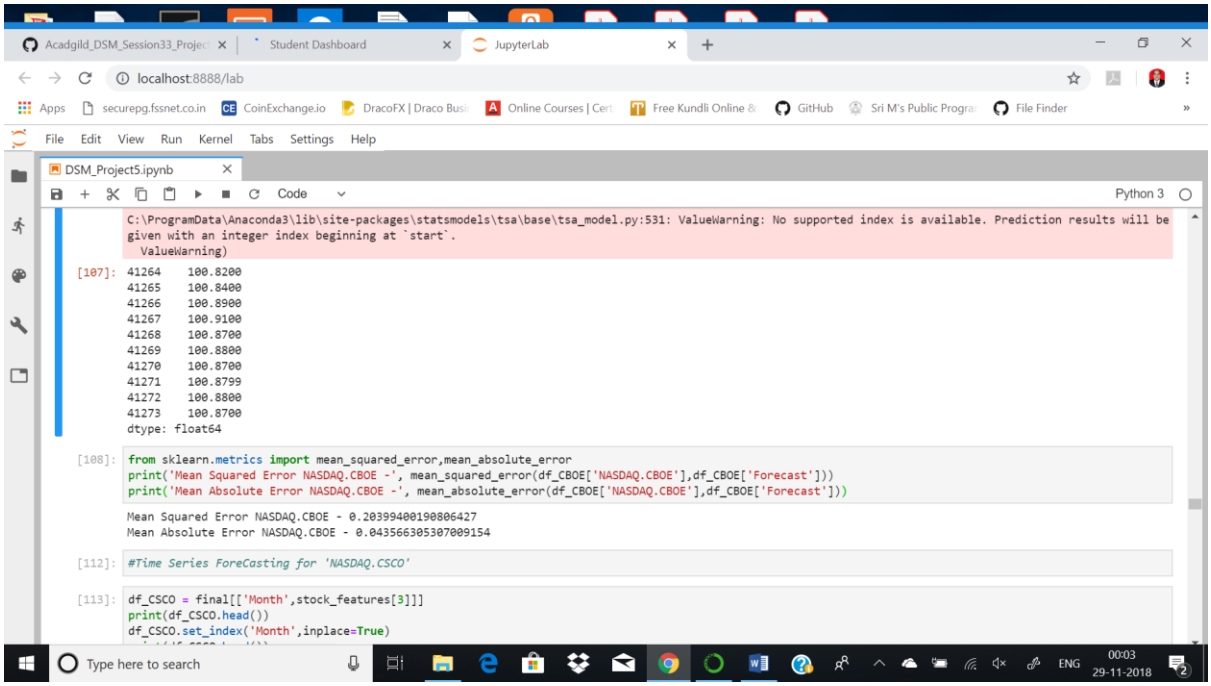

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
Apps
securepg.fssnet.co.in
CoinExchange.io
DracoFX | Draco Busi
Online Courses | Cert
Free Kundli Online &
GitHub
Sri M's Public Progra
File Finder
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
1.0
0.8
0.6
0.4
0.2
0.0
Type here to search
ENG
00:01
29-11-2018

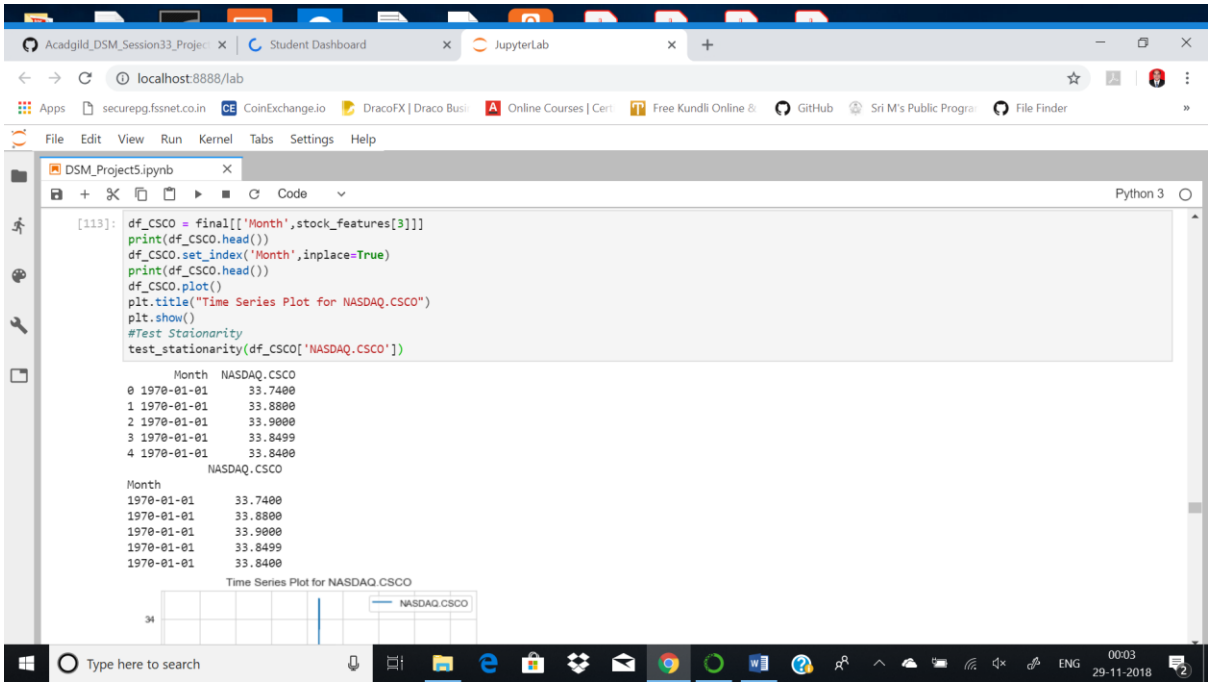

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
Apps
securepg.fssnet.co.in
CoinExchange.io
DracoFX | Draco Busi
Online Courses | Cert
Free Kundli Online &
GitHub
Sri M's Public Progra
File Finder
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
[104]:
"""Note
The two dotted lines on either sides of 0 are the confidence intervals.
These can be used to determine the 'p' and 'q' values as:
• p: The first time where the PACF crosses the upper confidence interval, here its close to 0. hence p = 0.
• q: The first time where the ACF crosses the upper confidence interval, here its close to 0. hence p = 0."""
[104]:
'Note\nThe two dotted lines on either sides of 0 are the confidence intervals.\n\nThese can be used to determine the 'p' and 'q' values as:\n\n• p: The fir
st time where the PACF crosses the upper confidence interval, here its close to 0. hence p = 0.\n\n• q: The first time where the ACF crosses the upper conf
idence interval, here its close to 0. hence p = 0.'
[105]:
# fit model
model= sm.tsa.statespace.SARIMAX(df_CBOE['NASDAQ.CBOE'],order=(0,1,0),seasonal_order=(0,1,0,12))
results = model.fit()
print(results.summary())
print(results.forecast())
df_CBOE['Forecast'] = results.predict()
df_CBOE[['NASDAQ.CBOE','Forecast']].plot(figsize=(20,8))
plt.show()
C:\ProgramData\Anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:225: ValueWarning: A date index has been provided, but it has no associated fr
equency information and so will be ignored when e.g. forecasting.
' ignored when e.g. forecasting.', ValueWarning)
C:\ProgramData\Anaconda3\lib\site-packages\statsmodels\base\model.py:508: ConvergenceWarning: Maximum Likelihood optimization failed to converge. Check mle
_retvals
Type here to search
ENG
00:01
29-11-2018

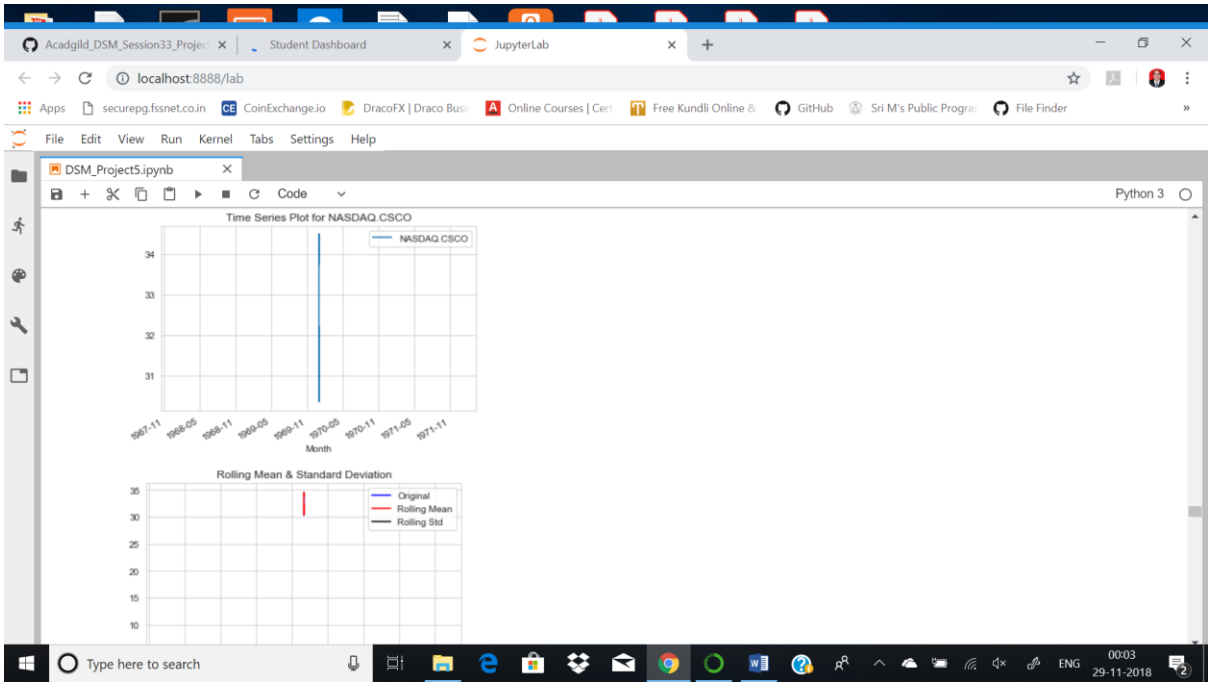

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
Statespace Model Results
Dep. Variable: NASDAQ.CBOE No. Observations: 41265
Model: SARIMAX(0, 1, 0)x(0, 1, 0, 12) Log Likelihood 53414.092
Date: Wed, 28 Nov 2018 AIC -106826.184
Time: 23:13:58 BIC -106817.557
Sample: 0 HQIC -106823.457
- 41265
Covariance Type: opg
coef std err z P>|z| [0.025 0.975]
sigma2 0.0044 5.33e-06 824.305 0.000 0.004 0.004
Ljung-Box (Q): 11084.06 Jarque-Bera (JB): 7011759.80
Prob(Q): 0.00 Prob(JB): 0.00
Heteroskedasticity (H): 0.94 Skew: -0.46
Prob(H) (two-sided): 0.00 Kurtosis: 66.86
Warnings:
[1] Covariance matrix calculated using the outer product of gradients (complex-step).
C:\ProgramData\Anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:531: ValueWarning: No supported index is available. Prediction results will be given with an integer index beginning at `start`.
ValueWarning)
41265 100.84
dtype: float64
Type here to search
ENG
00:02
29-11-2018

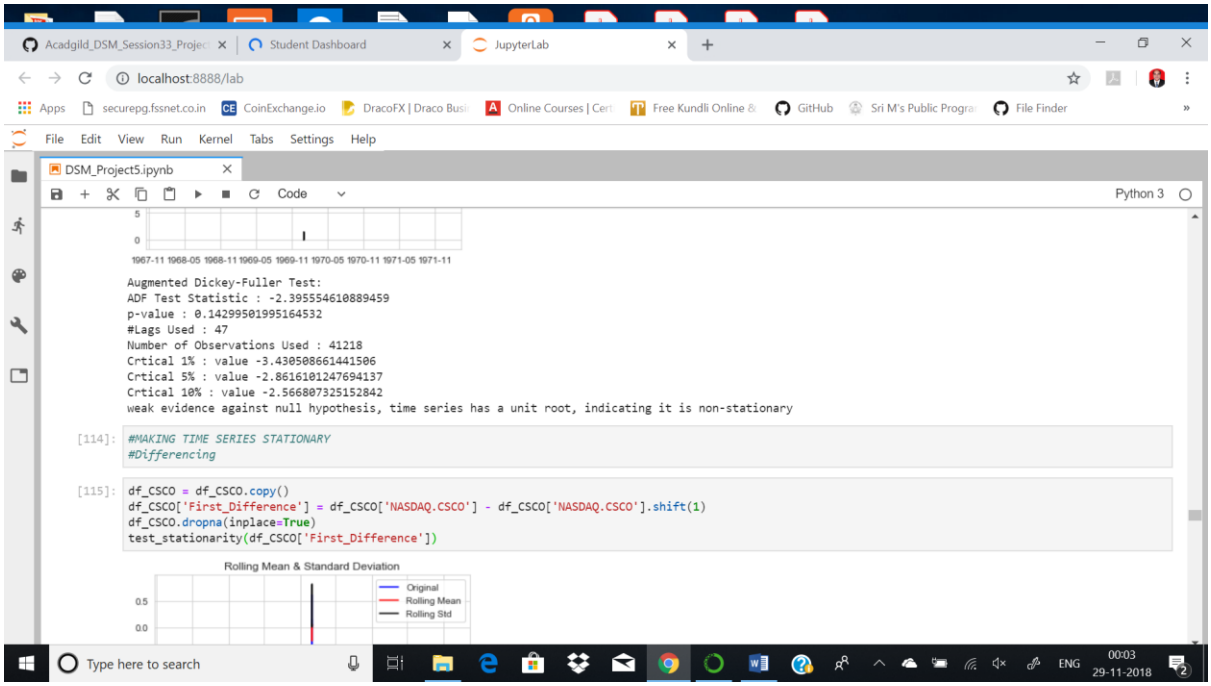

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
41265 100.84
dtype: float64
NASDAQ.CBOE
Forecast
120
100
80
60
40
20
0
1967-11
1968-05
1968-11
1969-05
1969-11
1970-05
1970-11
1971-05
1971-11
Month
Type here to search
ENG
00:02
29-11-2018

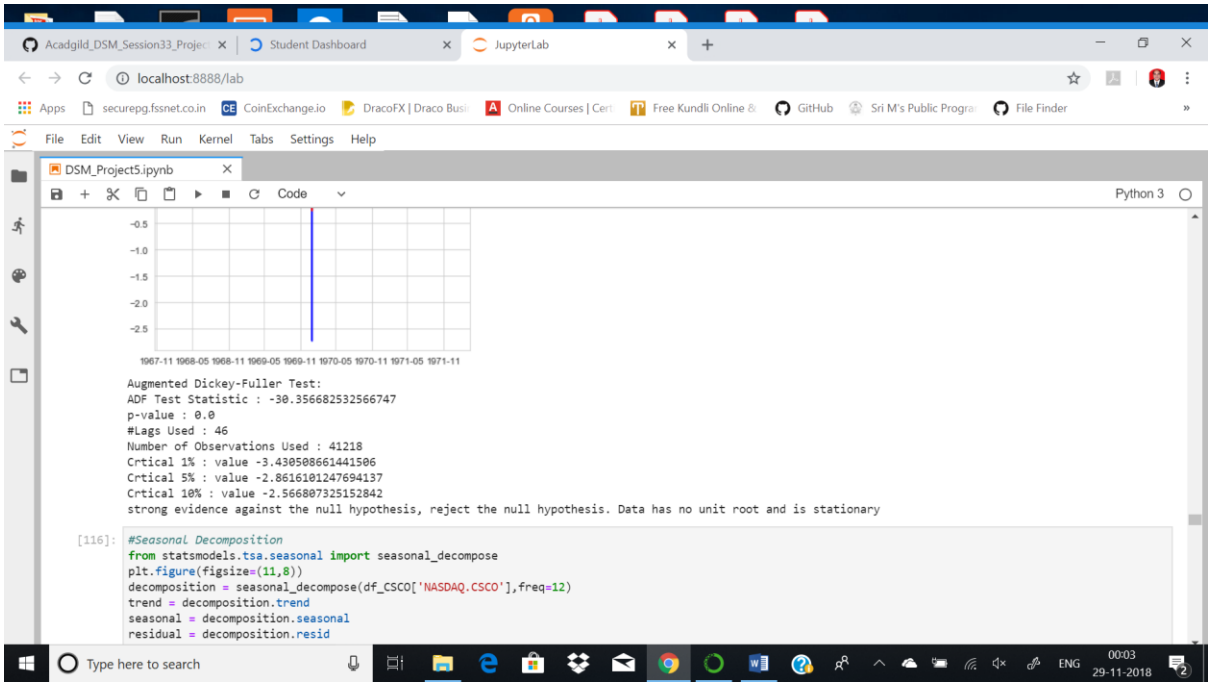

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
[106]: results.forecast(steps=10)
C:\ProgramData\Anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:531: ValueWarning: No supported index is available. Prediction results will be given with an integer index beginning at `start`.
ValueWarning)
[106]: 41265 100.8400
41266 100.8900
41267 100.9100
41268 100.8700
41269 100.8800
41270 100.8700
41271 100.8799
41272 100.8800
41273 100.8700
41274 100.8500
dtype: float64
[107]: results.predict(start=41264,end=41273)
C:\ProgramData\Anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:531: ValueWarning: No supported index is available. Prediction results will be given with an integer index beginning at `start`.
ValueWarning)
[107]: 41264 100.8200
41265 100.8400
41266 100.8900
41267 100.9100
Type here to search
ENG
00:02
29-11-2018


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
C:\ProgramData\Anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:531: ValueWarning: No supported index is available. Prediction results will be given with an integer index beginning at `start`.
ValueWarning)
[107]: 41264 100.8200
41265 100.8400
41266 100.8900
41267 100.9100
41268 100.8700
41269 100.8800
41270 100.8700
41271 100.8799
41272 100.8800
41273 100.8700
dtype: float64
[108]: from sklearn.metrics import mean_squared_error,mean_absolute_error
print('Mean Squared Error NASDAQ.CBOE -', mean_squared_error(df_CBOE['NASDAQ.CBOE'],df_CBOE['Forecast']))
print('Mean Absolute Error NASDAQ.CBOE -', mean_absolute_error(df_CBOE['NASDAQ.CBOE'],df_CBOE['Forecast']))
Mean Squared Error NASDAQ.CBOE - 0.20399400190806427
Mean Absolute Error NASDAQ.CBOE - 0.043566305307009154
[112]: #Time Series ForeCasting for 'NASDAQ.CSCO'
[113]: df_CSCO = final[['Month',stock_features[3]]]
print(df_CSCO.head())
df_CSCO.set_index('Month',inplace=True)
Type here to search
ENG
00:03
29-11-2018


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
Apps
securepg.fssnet.co.in
CoinExchange.io
DracoFX | Draco Busi
Online Courses | Cert
Free Kundli Online &
GitHub
Sri M's Public Progra
File Finder
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
[113]:
df_CSCO = final[['Month',stock_features[3]]]
print(df_CSCO.head())
df_CSCO.set_index('Month',inplace=True)
print(df_CSCO.head())
df_CSCO.plot()
plt.title("Time Series Plot for NASDAQ.CSCO")
plt.show()
#Test Staionarity
test_stationarity(df_CSCO['NASDAQ.CSCO'])
Month NASDAQ.CSCO
0 1970-01-01 33.7400
1 1970-01-01 33.8800
2 1970-01-01 33.9000
3 1970-01-01 33.8499
4 1970-01-01 33.8400
NASDAQ.CSCO
Month
1970-01-01 33.7400
1970-01-01 33.8800
1970-01-01 33.9000
1970-01-01 33.8499
1970-01-01 33.8400
Time Series Plot for NASDAQ.CSCO
NASDAQ.CSCO
34
Type here to search
ENG
00:03
29-11-2018

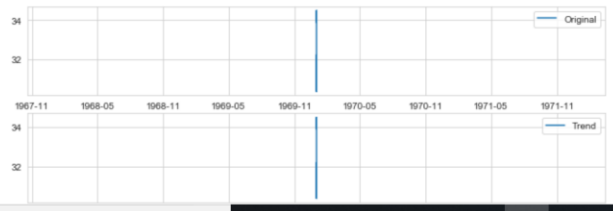

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
Apps
securepg.fssnet.co.in
CoinExchange.io
DracoFX | Draco Busi
Online Courses | Cert
Free Kundli Online &
GitHub
Sri M's Public Progra
File Finder
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
Time Series Plot for NASDAQ.CSCO
NASDAQ.CSCO
34
33
32
31
1967-11
1968-05
1968-11
1969-05
1969-11
1970-05
1970-11
1971-05
1971-11
Month
Rolling Mean & Standard Deviation
Original
Rolling Mean
Rolling Std
35
30
25
20
15
10
Type here to search
ENG
00:03
29-11-2018

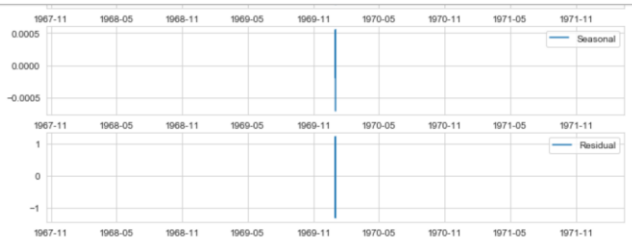

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
1967-11 1968-05 1968-11 1969-05 1969-11 1970-05 1970-11 1971-05 1971-11
Augmented Dickey-Fuller Test:
ADF Test Statistic : -2.395554610889459
p-value : 0.14299501995164532
#Lags Used : 47
Number of Observations Used : 41218
Crtical 1% : value -3.430508661441506
Crtical 5% : value -2.8616101247694137
Crtical 10% : value -2.566807325152842
weak evidence against null hypothesis, time series has a unit root, indicating it is non-stationary
[114]: #MAKING TIME SERIES STATIONARY
#Differencing
[115]: df_CSCO = df_CSCO.copy()
df_CSCO['First_Difference'] = df_CSCO['NASDAQ.CSCO'] - df_CSCO['NASDAQ.CSCO'].shift(1)
df_CSCO.dropna(inplace=True)
test_stationarity(df_CSCO['First_Difference'])
Rolling Mean & Standard Deviation
Original
Rolling Mean
Rolling Std
Type here to search
ENG
00:03
29-11-2018


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
1967-11 1968-05 1968-11 1969-05 1969-11 1970-05 1970-11 1971-05 1971-11
Augmented Dickey-Fuller Test:
ADF Test Statistic : -30.356682532566747
p-value : 0.0
#Lags Used : 46
Number of Observations Used : 41218
Crtical 1% : value -3.430508661441506
Crtical 5% : value -2.8616101247694137
Crtical 10% : value -2.566807325152842
strong evidence against the null hypothesis, reject the null hypothesis. Data has no unit root and is stationary
[116]: #Seasonal Decomposition
from statsmodels.tsa.seasonal import seasonal_decompose
plt.figure(figsize=(11,8))
decomposition = seasonal_decompose(df_CSCO['NASDAQ.CSCO'],freq=12)
trend = decomposition.trend
seasonal = decomposition.seasonal
residual = decomposition.resid
Type here to search
ENG
00:03
29-11-2018

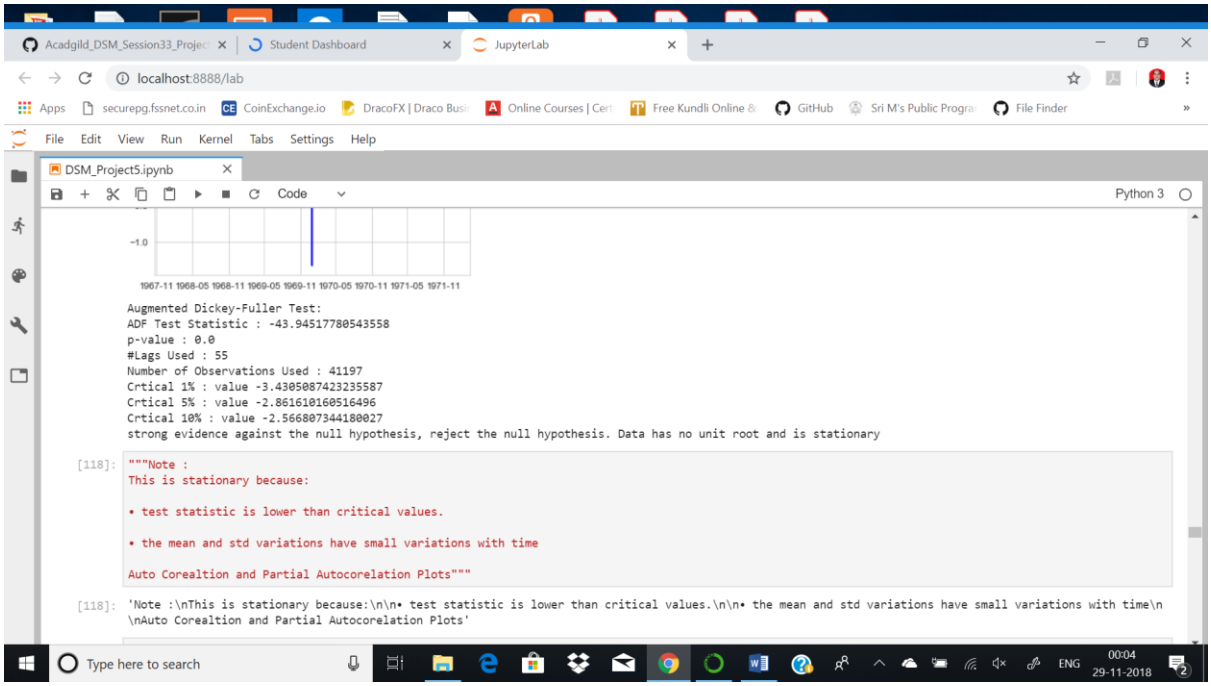
```
plt.subplot(411)
plt.plot(df_CSCO['NASDAQ.CSCO'],label='Original')
plt.legend(loc='best')
plt.subplot(412)
plt.plot(trend,label='Trend')
plt.legend(loc='best')
plt.subplot(413)
plt.plot(seasonal,label='Seasonal')
plt.legend(loc='best')
plt.subplot(414)
plt.plot(residual,label='Residual')
plt.legend(loc='best')
```

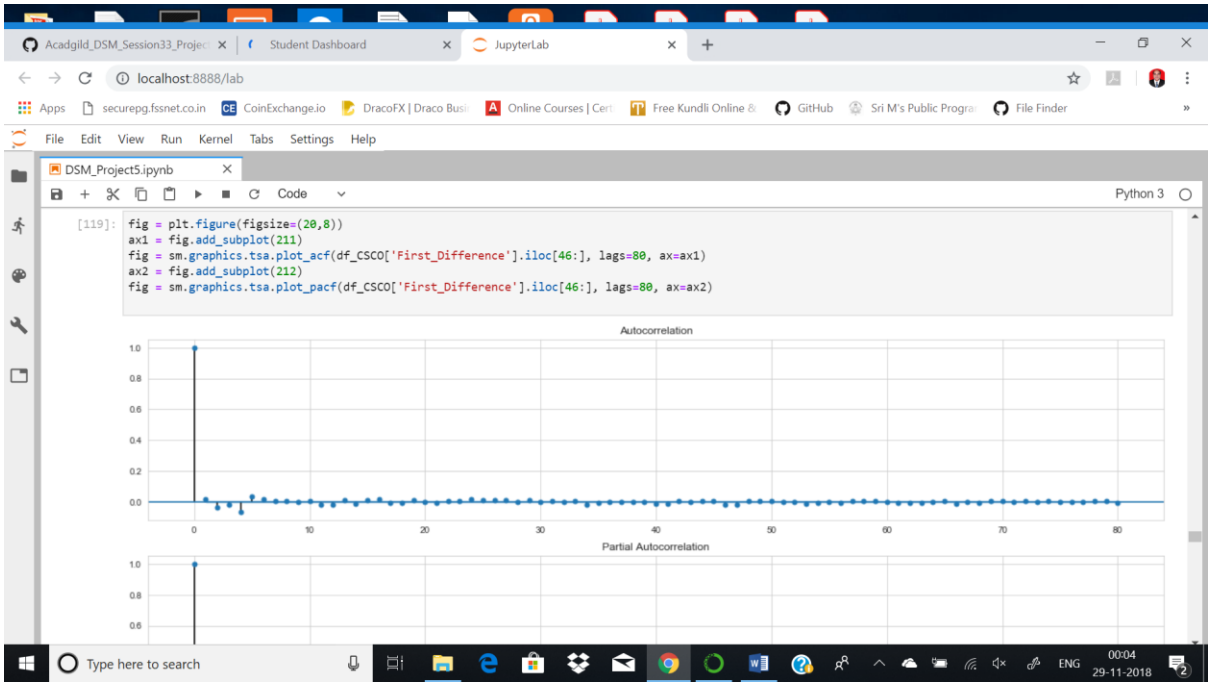
[116]: `<matplotlib.legend.Legend at 0x20501462518>`

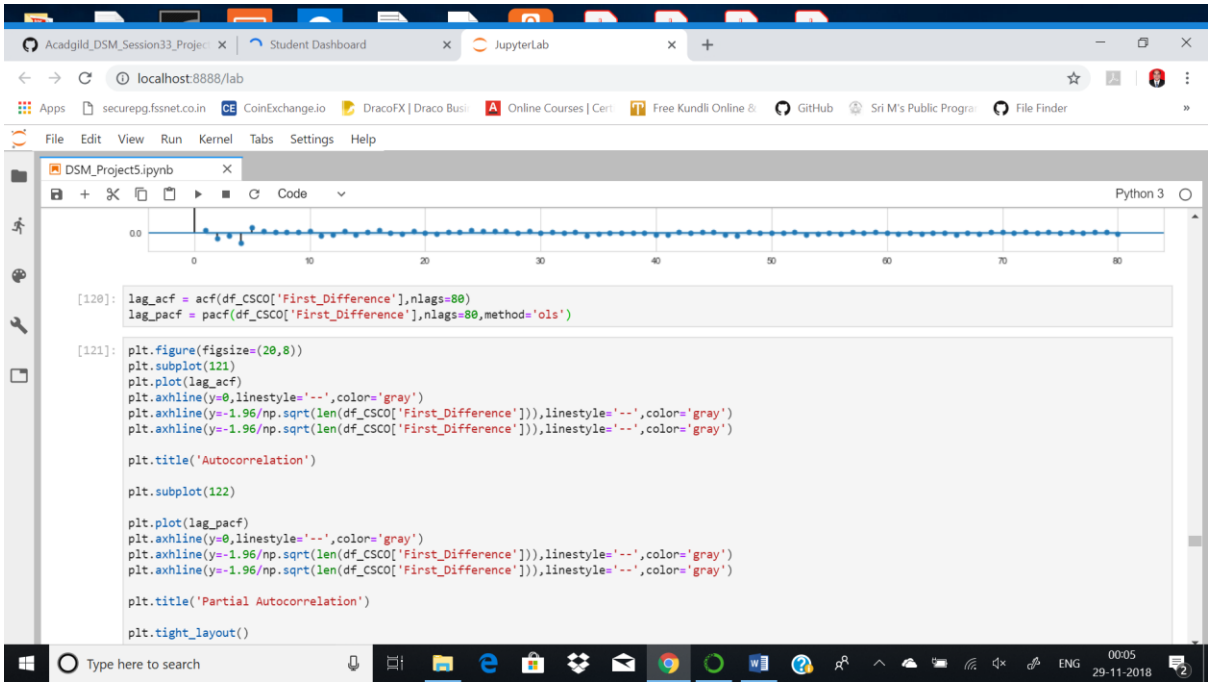


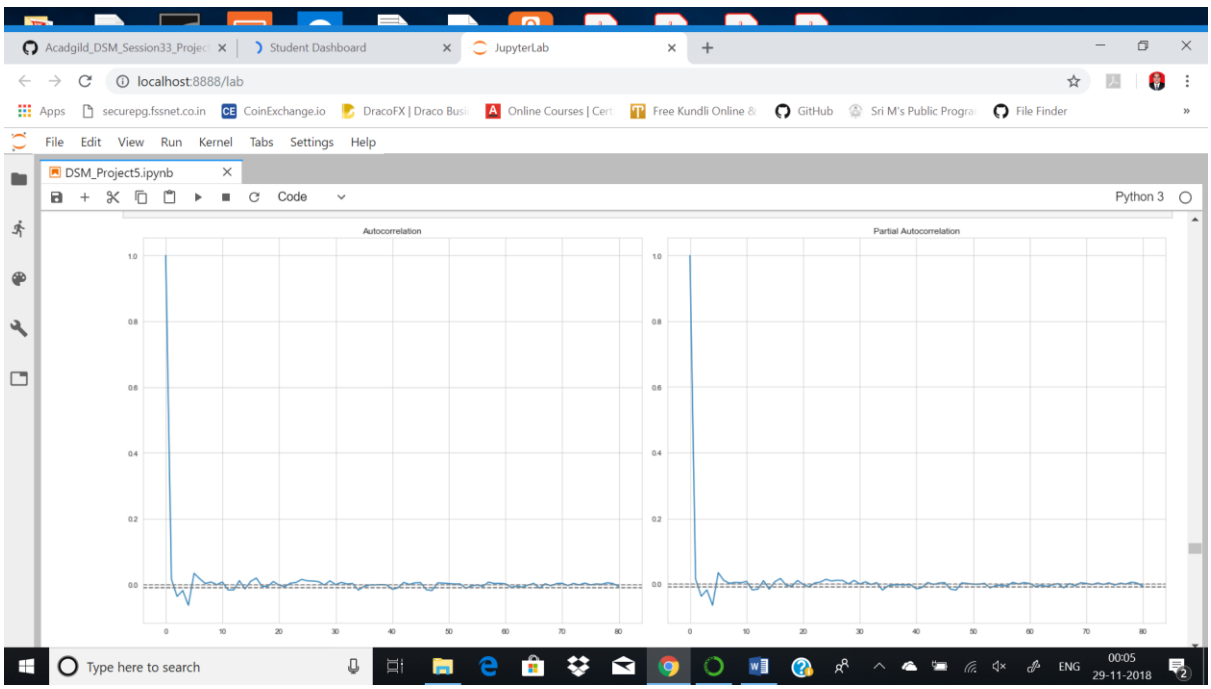


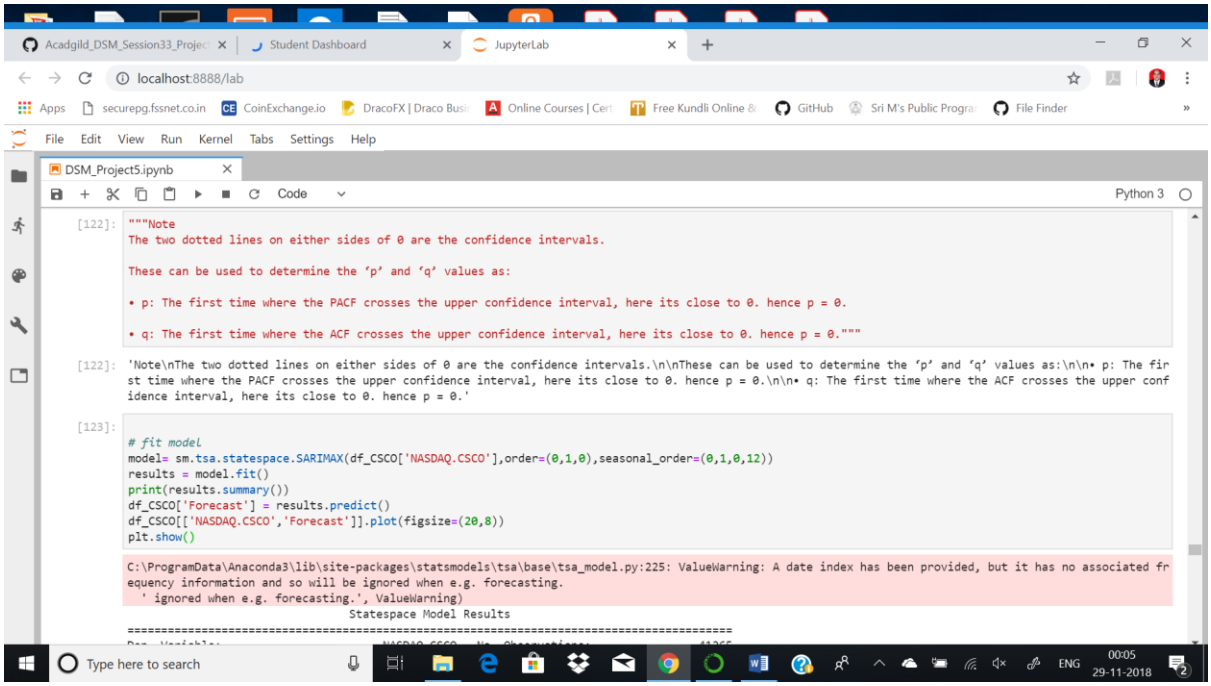
[117]:
```
ts_log_decompose = residual
ts_log_decompose.dropna(inplace=True)
test_stationarity(ts_log_decompose)
```

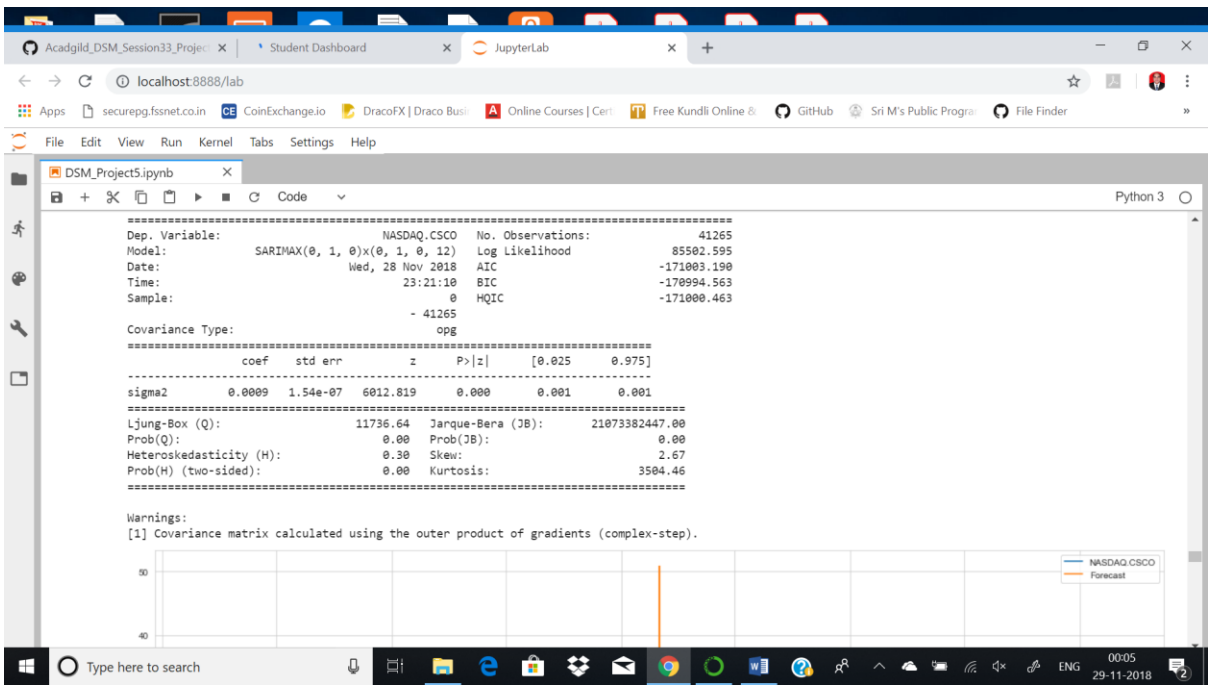

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
-1.0
1967-11 1968-05 1968-11 1969-05 1969-11 1970-05 1970-11 1971-05 1971-11
Augmented Dickey-Fuller Test:
ADF Test Statistic : -43.94517780543558
p-value : 0.0
#Lags Used : 55
Number of Observations Used : 41197
Crtical 1% : value -3.430508742323558 7
Crtical 5% : value -2.861610160516496
Crtical 10% : value -2.566807344180027
strong evidence against the null hypothesis, reject the null hypothesis. Data has no unit root and is stationary
[118]:
"""Note :
This is stationary because:
• test statistic is lower than critical values.
• the mean and std variations have small variations with time
Auto Corealtion and Partial Autocorelation Plots"""
[118]:
'Note :\nThis is stationary because:\n\n• test statistic is lower than critical values.\n\n• the mean and std variations have small variations with time\n
\nAuto Corealtion and Partial Autocorelation Plots'
Type here to search
ENG
00:04
29-11-2018

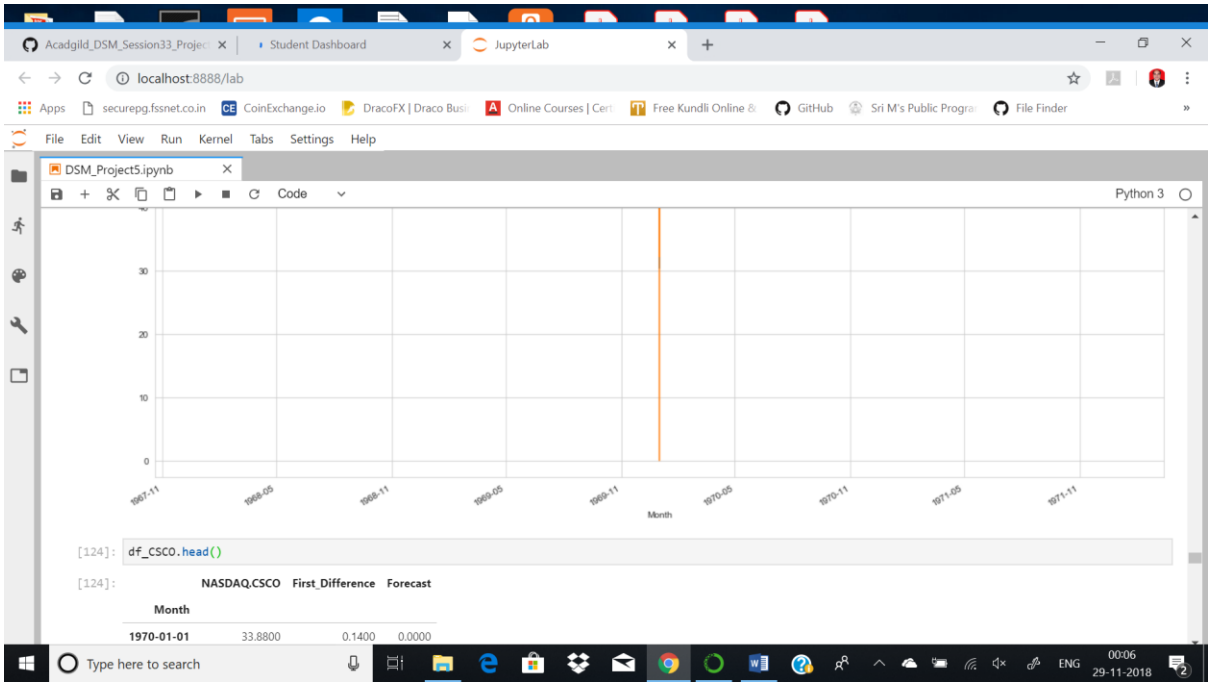

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
[119]:
fig = plt.figure(figsize=(20,8))
ax1 = fig.add_subplot(211)
fig = sm.graphics.tsa.plot_acf(df_CSCO['First_Difference'].iloc[46:], lags=80, ax=ax1)
ax2 = fig.add_subplot(212)
fig = sm.graphics.tsa.plot_pacf(df_CSCO['First_Difference'].iloc[46:], lags=80, ax=ax2)
Autocorrelation
1.0
0.8
0.6
0.4
0.2
0.0
0
10
20
30
40
50
60
70
80
Partial Autocorrelation
1.0
0.8
0.6
Type here to search
ENG
00:04
29-11-2018

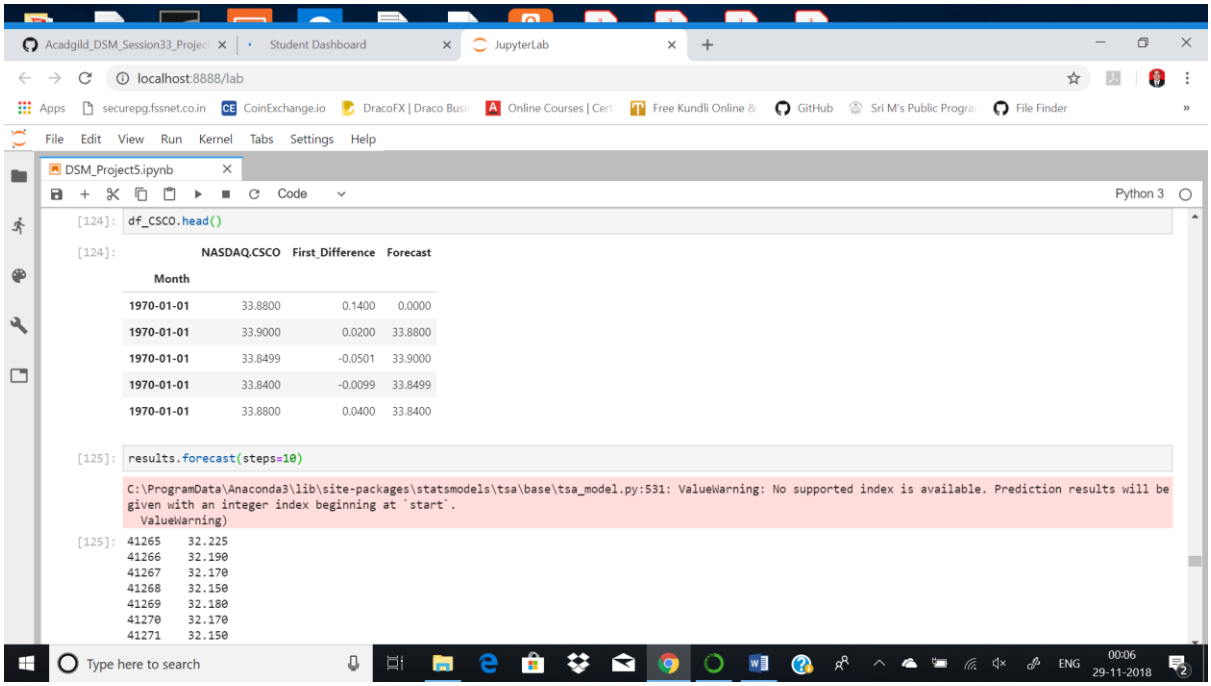

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
Apps
securepg.fssnet.co.in
CoinExchange.io
DracoFX | Draco Busi
Online Courses | Cert
Free Kundli Online &
GitHub
Sri M's Public Progra
File Finder
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
[120]:
lag_acf = acf(df_CSCO['First_Difference'],nlags=80)
lag_pacf = pacf(df_CSCO['First_Difference'],nlags=80,method='ols')
[121]:
plt.figure(figsize=(20,8))
plt.subplot(121)
plt.plot(lag_acf)
plt.axhline(y=0,linestyle='--',color='gray')
plt.axhline(y=-1.96/np.sqrt(len(df_CSCO['First_Difference'])),linestyle='--',color='gray')
plt.axhline(y=-1.96/np.sqrt(len(df_CSCO['First_Difference'])),linestyle='--',color='gray')
plt.title('Autocorrelation')
plt.subplot(122)
plt.plot(lag_pacf)
plt.axhline(y=0,linestyle='--',color='gray')
plt.axhline(y=-1.96/np.sqrt(len(df_CSCO['First_Difference'])),linestyle='--',color='gray')
plt.axhline(y=-1.96/np.sqrt(len(df_CSCO['First_Difference'])),linestyle='--',color='gray')
plt.title('Partial Autocorrelation')
plt.tight_layout()
Type here to search
ENG
00:05
29-11-2018

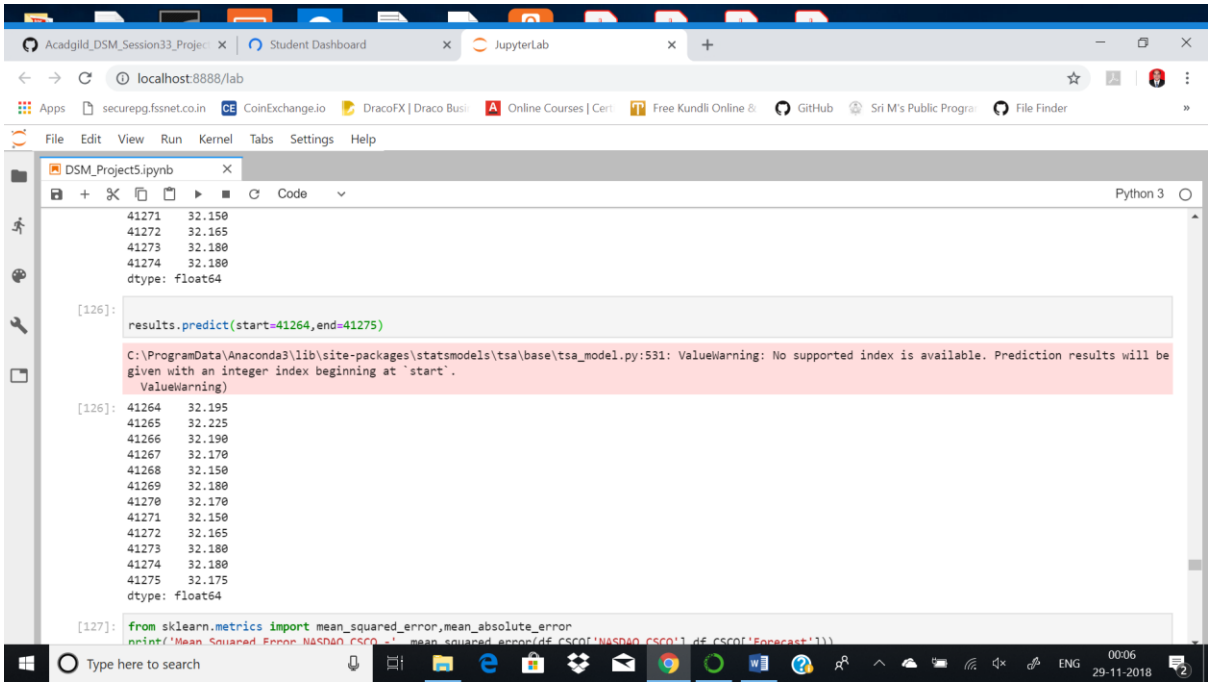

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
Apps
securepg.fssnet.co.in
CoinExchange.io
DracoFX | Draco Busi
Online Courses | Cert
Free Kundli Online &
GitHub
Sri M's Public Progra
File Finder
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
Autocorrelation
Partial Autocorrelation
Type here to search
ENG
00:05
29-11-2018

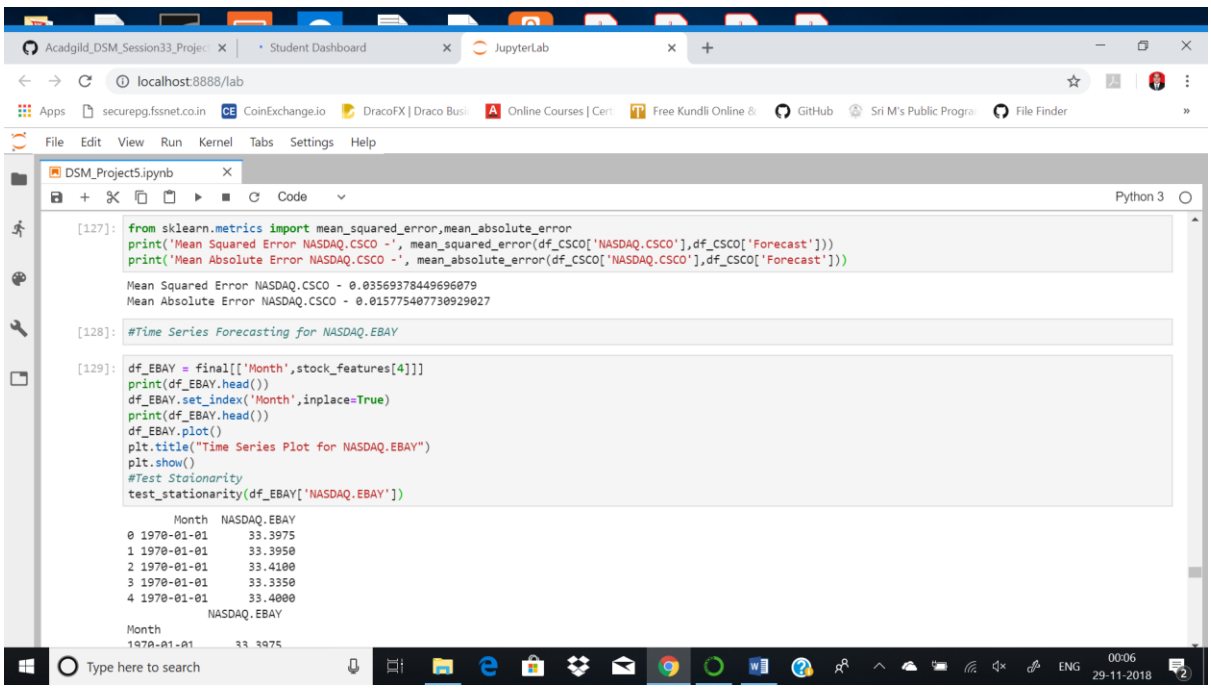

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
[122]:
"""Note
The two dotted lines on either sides of 0 are the confidence intervals.
These can be used to determine the 'p' and 'q' values as:
• p: The first time where the PACF crosses the upper confidence interval, here its close to 0. hence p = 0.
• q: The first time where the ACF crosses the upper confidence interval, here its close to 0. hence p = 0."""
[122]:
'Note\nThe two dotted lines on either sides of 0 are the confidence intervals.\n\nThese can be used to determine the 'p' and 'q' values as:\n\n• p: The fir
st time where the PACF crosses the upper confidence interval, here its close to 0. hence p = 0.\n\n• q: The first time where the ACF crosses the upper conf
idence interval, here its close to 0. hence p = 0.'
[123]:
# fit model
model= sm.tsa.statespace.SARIMAX(df_CSCO['NASDAQ.CSCO'],order=(0,1,0),seasonal_order=(0,1,0,12))
results = model.fit()
print(results.summary())
df_CSCO['Forecast'] = results.predict()
df_CSCO[['NASDAQ.CSCO','Forecast']].plot(figsize=(20,8))
plt.show()
C:\ProgramData\Anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:225: ValueWarning: A date index has been provided, but it has no associated fr
equency information and so will be ignored when e.g. forecasting.
' ignored when e.g. forecasting.', ValueWarning)
Statespace Model Results
Type here to search
ENG
00:05
29-11-2018

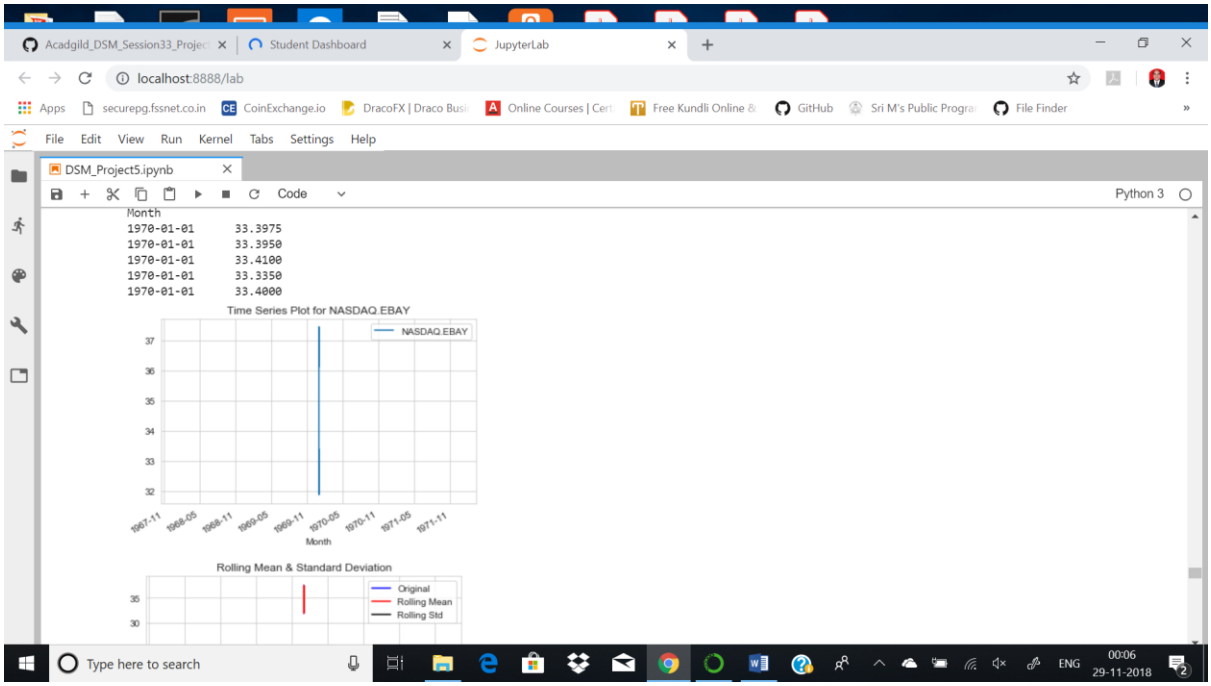

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
Dep. Variable: NASDAQ.CSCO No. Observations: 41265
Model: SARIMAX(0, 1, 0)x(0, 1, 0, 12) Log Likelihood 85502.595
Date: Wed, 28 Nov 2018 AIC -171003.190
Time: 23:21:10 BIC -170994.563
Sample: 0 HQIC -171000.463
- 41265
Covariance Type: opg
coef std err z P>|z| [0.025 0.975]
sigma2 0.0009 1.54e-07 6012.819 0.000 0.001 0.001
Ljung-Box (Q): 11736.64 Jarque-Bera (JB): 21073382447.00
Prob(Q): 0.00 Prob(JB): 0.00
Heteroskedasticity (H): 0.30 Skew: 2.67
Prob(H) (two-sided): 0.00 Kurtosis: 3504.46
Warnings:
[1] Covariance matrix calculated using the outer product of gradients (complex-step).
NASDAQ.CSCO
Forecast
50
40
Type here to search
ENG
00:05
29-11-2018

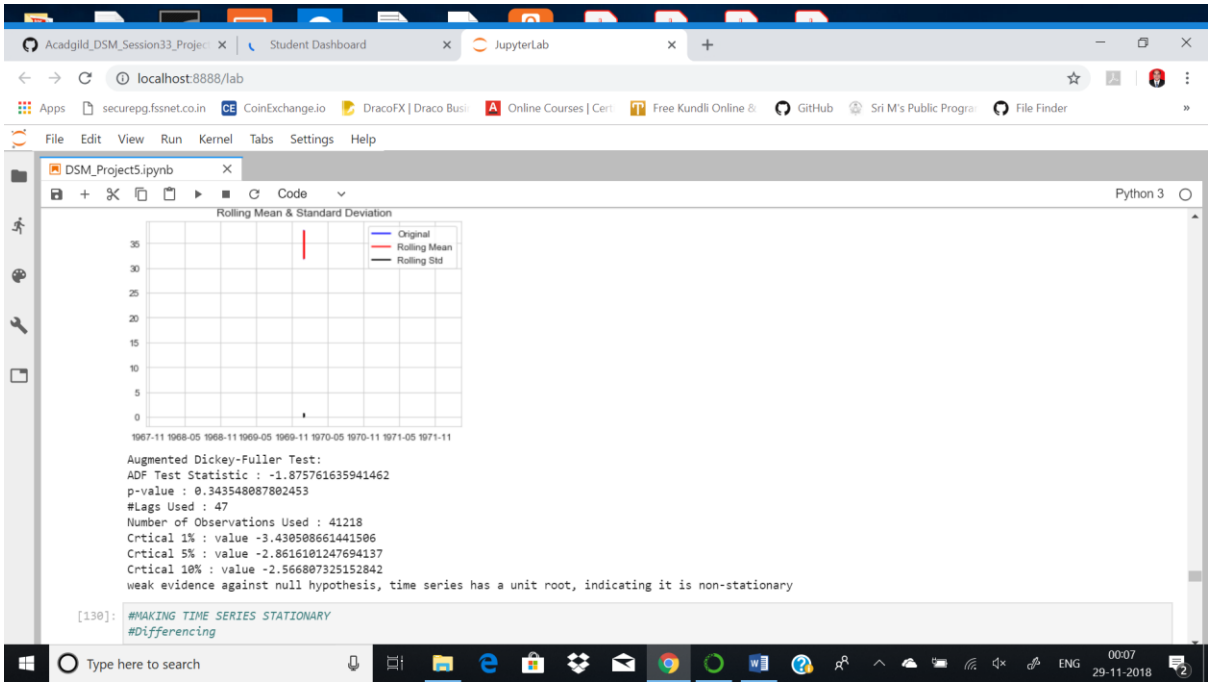

Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
Apps
securepg.fssnet.co.in
CoinExchange.io
DracoFX | Draco Busi
Online Courses | Cert
Free Kundli Online &
GitHub
Sri M's Public Progra
File Finder
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
30
20
10
0
1967-11
1968-05
1968-11
1969-05
1969-11
1970-05
1970-11
1971-05
1971-11
Month
[124]: df_CSCO.head()
[124]:
NASDAQ.CSCO First_Difference Forecast
Month
1970-01-01 33.8800 0.1400 0.0000
Type here to search
ENG
00:06
29-11-2018


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
Apps
securepg.fssnet.co.in
CoinExchange.io
DracoFX | Draco Busi
Online Courses | Cert
Free Kundli Online &
GitHub
Sri M's Public Progra
File Finder
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
[124]: df_CSCO.head()
[124]:
NASDAQ.CSCO First_Difference Forecast
Month
1970-01-01 33.8800 0.1400 0.0000
1970-01-01 33.9000 0.0200 33.8800
1970-01-01 33.8499 -0.0501 33.9000
1970-01-01 33.8400 -0.0099 33.8499
1970-01-01 33.8800 0.0400 33.8400
[125]: results.forecast(steps=10)
C:\ProgramData\Anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:531: ValueWarning: No supported index is available. Prediction results will be given with an integer index beginning at `start`.
ValueWarning)
[125]: 41265 32.225
41266 32.190
41267 32.170
41268 32.150
41269 32.180
41270 32.170
41271 32.150
Type here to search
ENG
00:06
29-11-2018


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
41271 32.150
41272 32.165
41273 32.180
41274 32.180
dtype: float64
[126]:
results.predict(start=41264,end=41275)
C:\ProgramData\Anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:531: ValueWarning: No supported index is available. Prediction results will be given with an integer index beginning at `start`.
ValueWarning)
[126]: 41264 32.195
41265 32.225
41266 32.190
41267 32.170
41268 32.150
41269 32.180
41270 32.170
41271 32.150
41272 32.165
41273 32.180
41274 32.180
41275 32.175
dtype: float64
[127]:
from sklearn.metrics import mean_squared_error,mean_absolute_error
Type here to search
ENG
00:06
29-11-2018


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
[127]:
from sklearn.metrics import mean_squared_error,mean_absolute_error
print('Mean Squared Error NASDAQ.CSCO -', mean_squared_error(df_CSCO['NASDAQ.CSCO'],df_CSCO['Forecast']))
print('Mean Absolute Error NASDAQ.CSCO -', mean_absolute_error(df_CSCO['NASDAQ.CSCO'],df_CSCO['Forecast']))
Mean Squared Error NASDAQ.CSCO - 0.035693784496960 79
Mean Absolute Error NASDAQ.CSCO - 0.015775407730929027
[128]:
#Time Series Forecasting for NASDAQ.EBAY
[129]:
df_EBAY = final[['Month',stock_features[4]]]
print(df_EBAY.head())
df_EBAY.set_index('Month',inplace=True)
print(df_EBAY.head())
df_EBAY.plot()
plt.title("Time Series Plot for NASDAQ.EBAY")
plt.show()
#Test Stationarity
test_stationarity(df_EBAY['NASDAQ.EBAY'])
Month NASDAQ.EBAY
0 1970-01-01 33.3975
1 1970-01-01 33.3950
2 1970-01-01 33.4100
3 1970-01-01 33.3350
4 1970-01-01 33.4000
NASDAQ.EBAY
Month
1970-01-01 33.3975
Type here to search
ENG
00:06
29-11-2018


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
Month
1970-01-01 33.3975
1970-01-01 33.3950
1970-01-01 33.4100
1970-01-01 33.3350
1970-01-01 33.4000
Time Series Plot for NASDAQ.EBAY
NASDAQ.EBAY
Month
Rolling Mean & Standard Deviation
Original
Rolling Mean
Rolling Std
Type here to search
ENG
00:06
29-11-2018


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
Rolling Mean & Standard Deviation
Original
Rolling Mean
Rolling Std
1967-11 1968-05 1968-11 1969-05 1969-11 1970-05 1970-11 1971-05 1971-11
Augmented Dickey-Fuller Test:
ADF Test Statistic : -1.875761635941462
p-value : 0.343548087802453
#Lags Used : 47
Number of Observations Used : 41218
Crtical 1% : value -3.430508661441506
Crtical 5% : value -2.8616101247694137
Crtical 10% : value -2.566807325152842
weak evidence against null hypothesis, time series has a unit root, indicating it is non-stationary
[130]:
#MAKING TIME SERIES STATIONARY
#Differencing
Type here to search
ENG
00:07
29-11-2018

Acadgild_DSM_Session33_Projec | Student Dashboard | JupyterLab

localhost:8888/lab

Apps | securepg.fssnet.co.in | CoinExchange.io | DracoFX | Draco Busi | Online Courses | Cert | Free Kundli Online & | GitHub | Sri M's Public Progra | File Finder

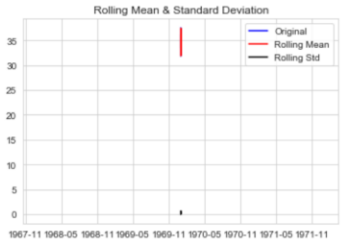
File Edit View Run Kernel Tabs Settings Help

DSM_Project5.ipynb

Code | Python 3

[130]:
```
#MAKING TIME SERIES STATIONARY
#Differencing
```

[131]:
```
df_EBAY = df_EBAY.copy()
df_EBAY['First_Difference'] = df_EBAY['NASDAQ.EBAY'] - df_EBAY['NASDAQ.EBAY'].shift(1)
df_EBAY.dropna(inplace=True)
#test Stationarity
test_stationarity(df_EBAY['NASDAQ.EBAY'])
```




```
Augmented Dickey-Fuller Test:
ADF Test Statistic : -1.8639133106583694
```

Type here to search | ENG 00:07 29-11-2018

Acadgild_DSM_Session33_Projec | Student Dashboard | JupyterLab

localhost:8888/lab

Apps | securepg.fssnet.co.in | CoinExchange.io | DracoFX | Draco Busi | Online Courses | Cert | Free Kundli Online & | GitHub | Sri M's Public Progra | File Finder

File Edit View Run Kernel Tabs Settings Help

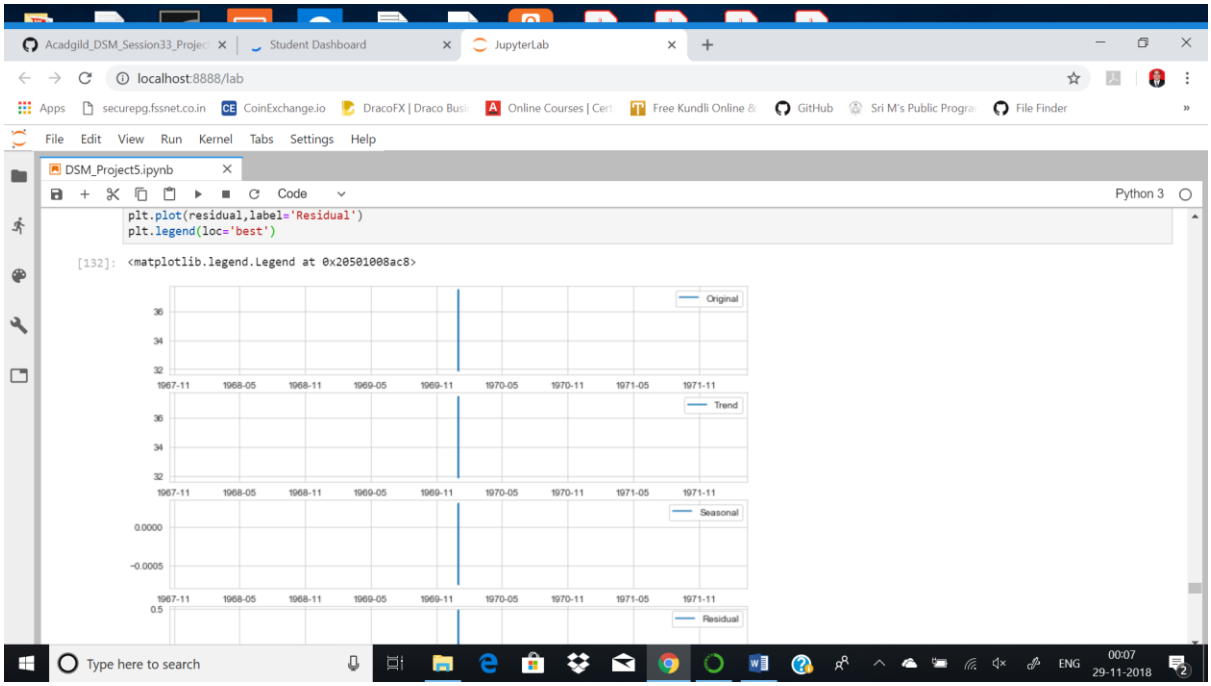
DSM_Project5.ipynb

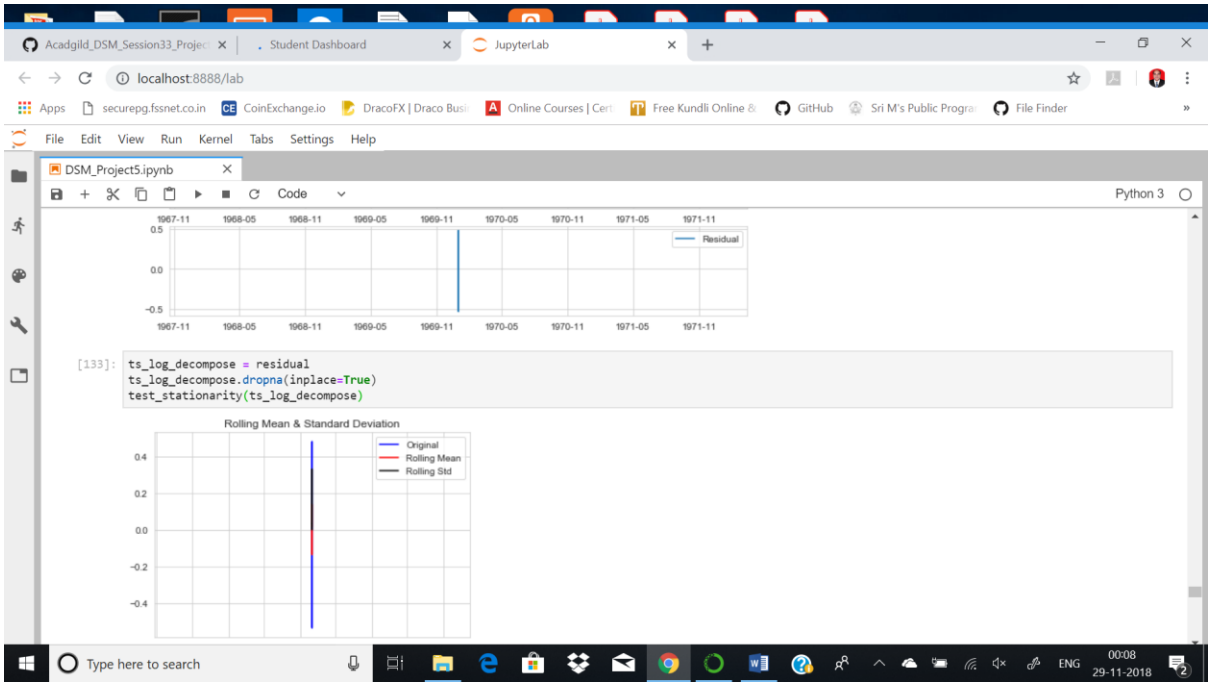
Code | Python 3

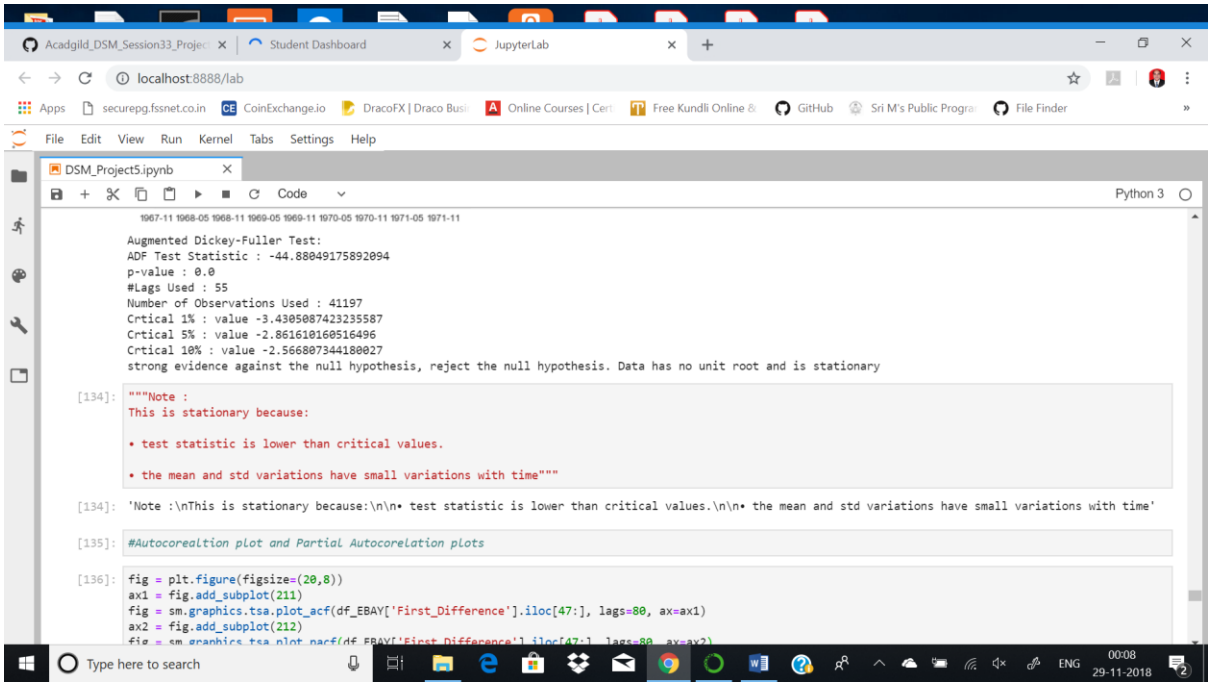
```
Augmented Dickey-Fuller Test:
ADF Test Statistic : -1.8639133106583694
p-value : 0.3492231149987619
#Lags Used : 47
Number of Observations Used : 41217
Crtical 1% : value -3.4305086652911636
Crtical 5% : value -2.8616101264708296
Crtical 10% : value -2.5668073260584587
weak evidence against null hypothesis, time series has a unit root, indicating it is non-stationary
```

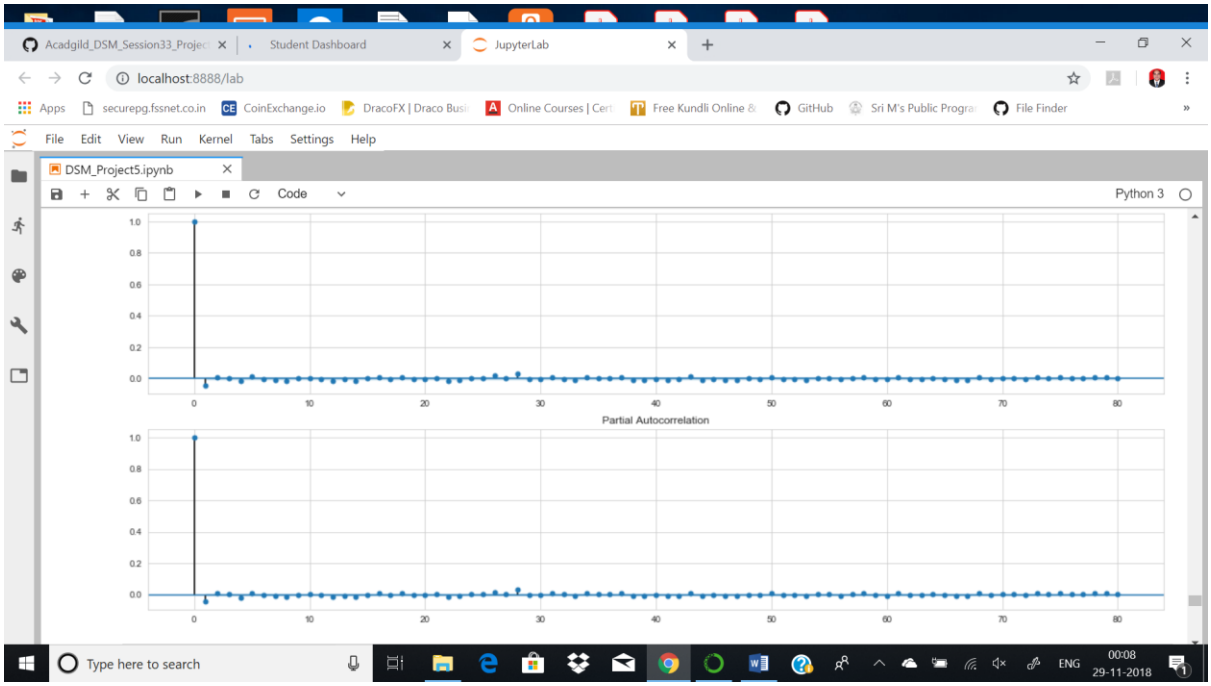
[132]:
```
#Seasonal Decomposition
from statsmodels.tsa.seasonal import seasonal_decompose
plt.figure(figsize=(11,8))
decomposition = seasonal_decompose(df_EBAY['NASDAQ.EBAY'],freq=12)
trend = decomposition.trend
seasonal = decomposition.seasonal
residual = decomposition.resid
plt.subplot(411)
plt.plot(df_EBAY['NASDAQ.EBAY'],label='Original')
plt.legend(loc='best')
plt.subplot(412)
plt.plot(trend,label='Trend')
plt.legend(loc='best')
plt.subplot(413)
plt.plot(seasonal,label='Seasonal')
plt.legend(loc='best')
plt.subplot(414)
plt.plot(residual,label='Residual')
```

Type here to search | ENG 00:07 29-11-2018


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
plt.plot(residual,label='Residual')
plt.legend(loc='best')
[132]: <matplotlib.legend.Legend at 0x20501008ac8>
Original
Trend
Seasonal
Residual
Type here to search
ENG
00:07
29-11-2018


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
Residual
[133]: ts_log_decompose = residual
ts_log_decompose.dropna(inplace=True)
test_stationarity(ts_log_decompose)
Rolling Mean & Standard Deviation
Original
Rolling Mean
Rolling Std
Type here to search
ENG
00:08
29-11-2018


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
Apps
securepg.fssnet.co.in
CoinExchange.io
DracoFX | Draco Busi
Online Courses | Cert
Free Kundli Online &
GitHub
Sri M's Public Progra
File Finder
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
1967-11 1968-05 1968-11 1969-05 1969-11 1970-05 1970-11 1971-05 1971-11
Augmented Dickey-Fuller Test:
ADF Test Statistic : -44.88049175892094
p-value : 0.0
#Lags Used : 55
Number of Observations Used : 41197
Crtical 1% : value -3.4305087423235587
Crtical 5% : value -2.861610160516496
Crtical 10% : value -2.566807344180027
strong evidence against the null hypothesis, reject the null hypothesis. Data has no unit root and is stationary
[134]:
"""Note :
This is stationary because:
• test statistic is lower than critical values.
• the mean and std variations have small variations with time"""
[134]:
'Note :\nThis is stationary because:\n\n• test statistic is lower than critical values.\n\n• the mean and std variations have small variations with time'
[135]:
#Autocorealtion plot and Partial Autocorelation plots
[136]:
fig = plt.figure(figsize=(20,8))
ax1 = fig.add_subplot(211)
fig = sm.graphics.tsa.plot_acf(df_EBAY['First_Difference'].iloc[47:], lags=80, ax=ax1)
ax2 = fig.add_subplot(212)
Type here to search
ENG
00:08
29-11-2018


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
Apps
securepg.fssnet.co.in
CoinExchange.io
DracoFX | Draco Busi
Online Courses | Cert
Free Kundli Online &
GitHub
Sri M's Public Progra
File Finder
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
Partial Autocorrelation
Type here to search
ENG
00:08
29-11-2018

Acadgild_DSM_Session33_Projec | Student Dashboard | JupyterLab

localhost:8888/lab

Apps | securepg.fssnet.co.in | CoinExchange.io | DracoFX | Draco Busi | Online Courses | Cert | Free Kundli Online & | GitHub | Sri M's Public Progra | File Finder

File Edit View Run Kernel Tabs Settings Help

DSM_Project5.ipynb

Code | Python 3

[137]:
```
lag_acf = acf(df_EBAY['First_Difference'],nlags=80)
lag_pacf = pacf(df_EBAY['First_Difference'],nlags=80,method='ols')
```

[138]:
```
plt.figure(figsize=(20,8))
plt.subplot(121)
plt.plot(lag_acf)
plt.axhline(y=0,linestyle='--',color='gray')
plt.axhline(y=-1.96/np.sqrt(len(df_EBAY['First_Difference'])),linestyle='--',color='gray')
plt.axhline(y=-1.96/np.sqrt(len(df_EBAY['First_Difference'])),linestyle='--',color='gray')

plt.title('Autocorrelation')

plt.subplot(122)

plt.plot(lag_pacf)
plt.axhline(y=0,linestyle='--',color='gray')
plt.axhline(y=-1.96/np.sqrt(len(df_EBAY['First_Difference'])),linestyle='--',color='gray')
plt.axhline(y=-1.96/np.sqrt(len(df_EBAY['First_Difference'])),linestyle='--',color='gray')

plt.title('Partial Autocorrelation')

plt.tight_layout()
```

Type here to search

ENG 00:08 29-11-2018

Acadgild_DSM_Session33_Projec | Student Dashboard | JupyterLab

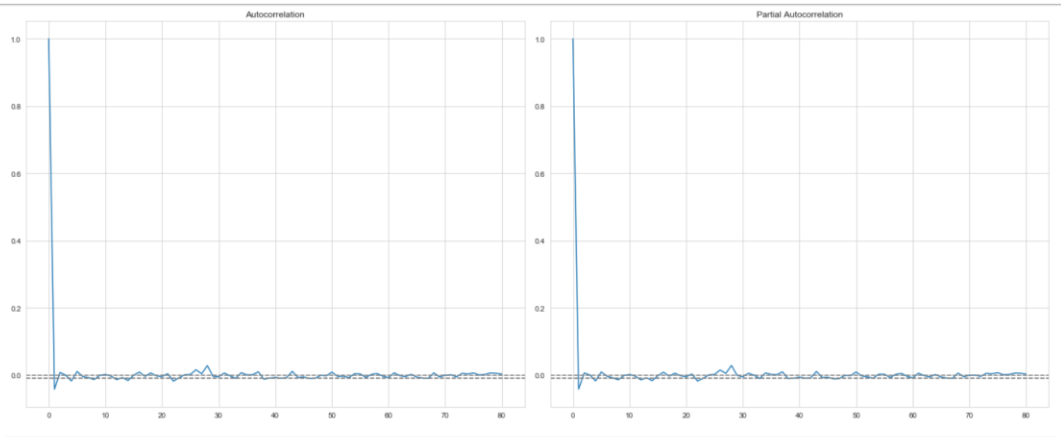
localhost:8888/lab

Apps | securepg.fssnet.co.in | CoinExchange.io | DracoFX | Draco Busi | Online Courses | Cert | Free Kundli Online & | GitHub | Sri M's Public Progra | File Finder

File Edit View Run Kernel Tabs Settings Help

DSM_Project5.ipynb

Code | Python 3



Type here to search

ENG 00:08 29-11-2018

Acadgild_DSM_Session33_Projec | Student Dashboard | JupyterLab

localhost:8888/lab

Apps | securepg.fssnet.co.in | CoinExchange.io | DracoFX | Draco Busi | Online Courses | Cert | Free Kundli Online & | GitHub | Sri M's Public Progra | File Finder

File Edit View Run Kernel Tabs Settings Help

DSM_Project5.ipynb

Code | Python 3

[139]:
```
"""Note
The two dotted lines on either sides of 0 are the confidence intervals.

These can be used to determine the 'p' and 'q' values as:

• p: The first time where the PACF crosses the upper confidence interval, here its close to 0. hence p = 0.

• q: The first time where the ACF crosses the upper confidence interval, here its close to 0. hence p = 0."""
```

[140]:
```
# fit model
model= sm.tsa.statespace.SARIMAX(df_EBAY['NASDAQ.EBAY'],order=(0,1,0),seasonal_order=(0,1,0,12))
results = model.fit()
print(results.summary())
df_EBAY['Forecast'] = results.predict()
df_EBAY[['NASDAQ.EBAY','Forecast']].plot(figsize=(20,8))
plt.show()
```

```
C:\ProgramData\Anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:225: ValueWarning: A date index has been provided, but it has no associated fr
equency information and so will be ignored when e.g. forecasting.
  ' ignored when e.g. forecasting.', ValueWarning)
                                 Statespace Model Results
==========================================================================================
Dep. Variable:                          NASDAQ.EBAY   No. Observations:                41265
Model:             SARIMAX(0, 1, 0)x(0, 1, 0, 12)   Log Likelihood              82104.712
Date:                            Wed, 28 Nov 2018   AIC                       -164207.425
Time:                                    23:28:45   BIC                       -164198.797
```

Type here to search | ENG 00:09 29-11-2018

Acadgild_DSM_Session33_Projec | Student Dashboard | JupyterLab

localhost:8888/lab

Apps | securepg.fssnet.co.in | CoinExchange.io | DracoFX | Draco Busi | Online Courses | Cert | Free Kundli Online & | GitHub | Sri M's Public Progra | File Finder

File Edit View Run Kernel Tabs Settings Help

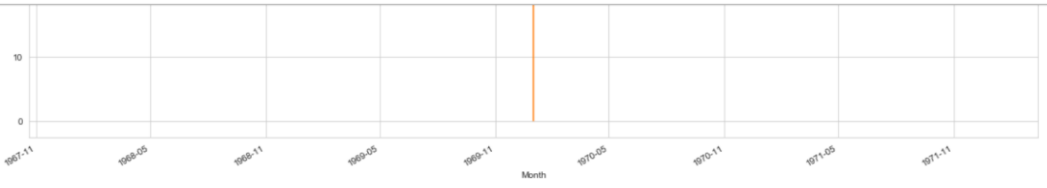
DSM_Project5.ipynb

Code | Python 3

```
C:\ProgramData\Anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:225: ValueWarning: A date index has been provided, but it has no associated fr
equency information and so will be ignored when e.g. forecasting.
  ' ignored when e.g. forecasting.', ValueWarning)
                                 Statespace Model Results
==========================================================================================
Dep. Variable:                          NASDAQ.EBAY   No. Observations:                41265
Model:             SARIMAX(0, 1, 0)x(0, 1, 0, 12)   Log Likelihood              82104.712
Date:                            Wed, 28 Nov 2018   AIC                       -164207.425
Time:                                    23:28:45   BIC                       -164198.797
Sample:                                         0   HQIC                      -164204.698
                                            - 41265
Covariance Type:                              opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
sigma2         0.0011   9.43e-07   1158.841      0.000       0.001       0.001
===================================================================================
Ljung-Box (Q):                    10939.63   Jarque-Bera (JB):          28223015.54
Prob(Q):                              0.00   Prob(JB):                         0.00
Heteroskedasticity (H):               1.21   Skew:                             0.35
Prob(H) (two-sided):                  0.00   Kurtosis:                       131.14
===================================================================================

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

Type here to search | ENG 00:09 29-11-2018

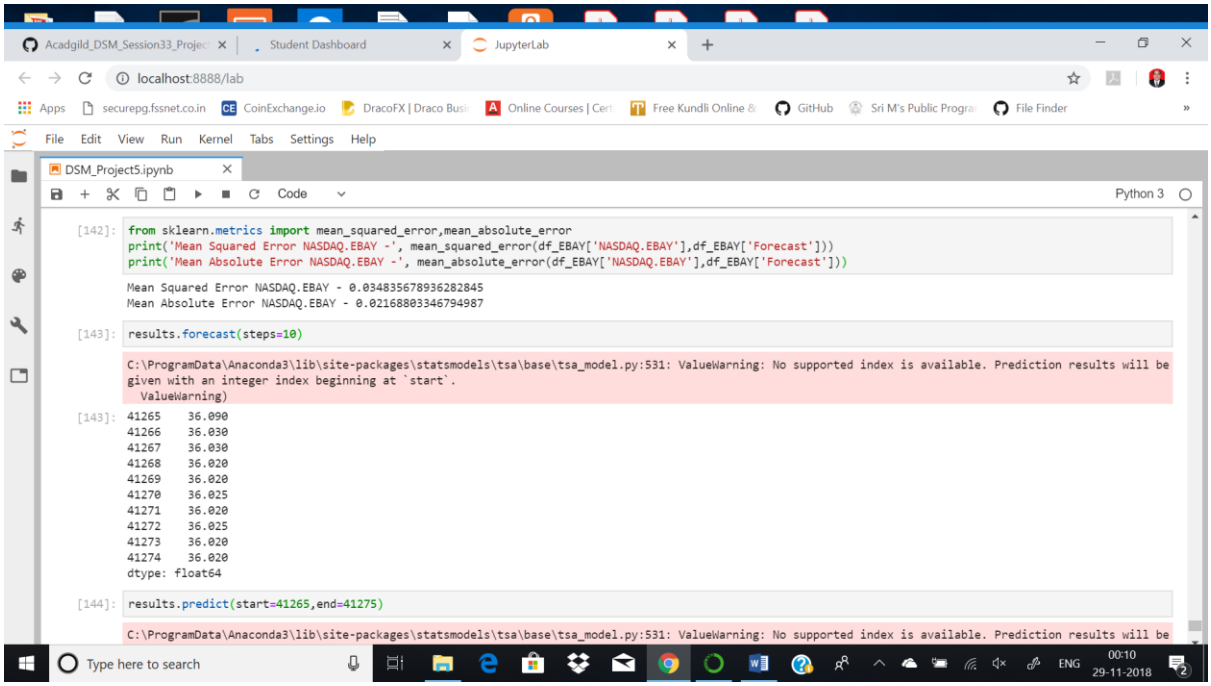
```
Ljung-Box (Q):                  10939.63   Jarque-Bera (JB):         28223015.54
Prob(Q):                            0.00   Prob(JB):                        0.00
Heteroskedasticity (H):             1.21   Skew:                            0.35
Prob(H) (two-sided):                0.00   Kurtosis:                      131.14

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

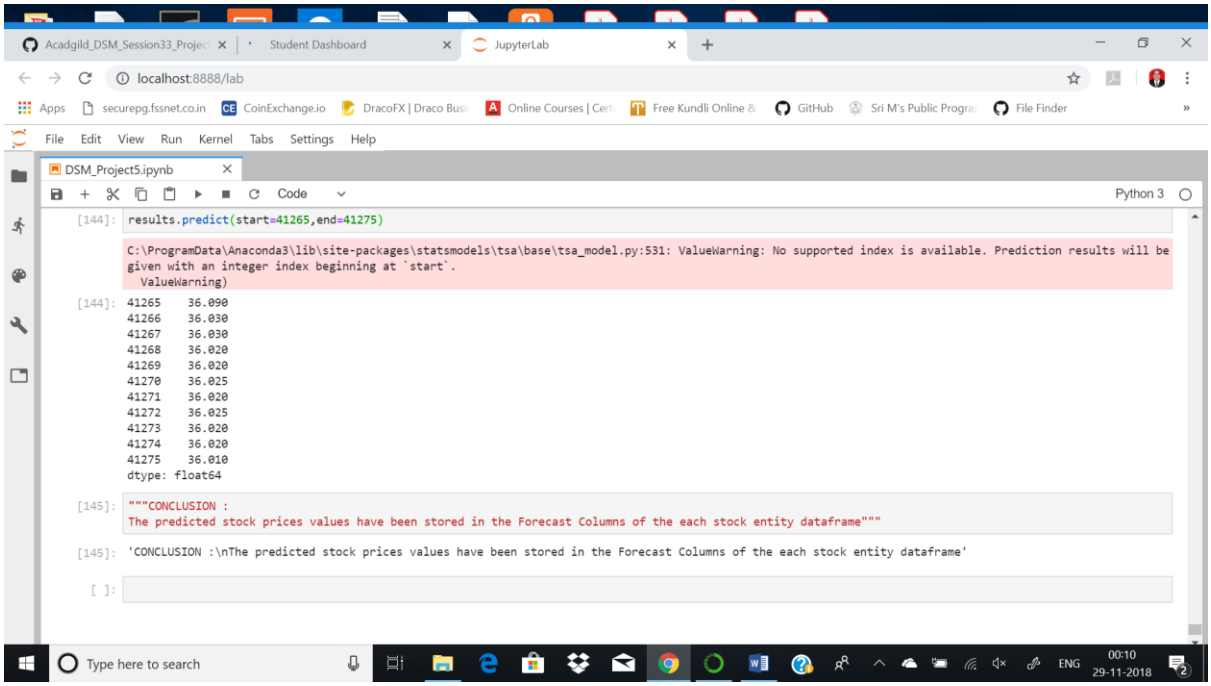


[141]:
```
df_EBAY.head()
```

[141]:

Month	NASDAQ.EBAY	First_Difference	Forecast
1970-01-01	33.395	-0.0025	0.000
1970-01-01	33.410	0.0150	33.395
1970-01-01	33.335	-0.0750	33.410
1970-01-01	33.400	0.0650	33.335
1970-01-01	33.430	0.0300	33.400


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
[142]: from sklearn.metrics import mean_squared_error,mean_absolute_error
print('Mean Squared Error NASDAQ.EBAY -', mean_squared_error(df_EBAY['NASDAQ.EBAY'],df_EBAY['Forecast']))
print('Mean Absolute Error NASDAQ.EBAY -', mean_absolute_error(df_EBAY['NASDAQ.EBAY'],df_EBAY['Forecast']))
Mean Squared Error NASDAQ.EBAY - 0.034835678936282845
Mean Absolute Error NASDAQ.EBAY - 0.02168803346794987
[143]: results.forecast(steps=10)
C:\ProgramData\Anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:531: ValueWarning: No supported index is available. Prediction results will be given with an integer index beginning at `start`.
ValueWarning)
[143]: 41265 36.090
41266 36.030
41267 36.030
41268 36.020
41269 36.020
41270 36.025
41271 36.020
41272 36.025
41273 36.020
41274 36.020
dtype: float64
[144]: results.predict(start=41265,end=41275)
C:\ProgramData\Anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:531: ValueWarning: No supported index is available. Prediction results will be
Type here to search
ENG
00:10
29-11-2018


Acadgild_DSM_Session33_Projec
Student Dashboard
JupyterLab
localhost:8888/lab
File Edit View Run Kernel Tabs Settings Help
DSM_Project5.ipynb
Code
Python 3
[144]: results.predict(start=41265,end=41275)
C:\ProgramData\Anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:531: ValueWarning: No supported index is available. Prediction results will be given with an integer index beginning at `start`.
ValueWarning)
[144]: 41265 36.090
41266 36.030
41267 36.030
41268 36.020
41269 36.020
41270 36.025
41271 36.020
41272 36.025
41273 36.020
41274 36.020
41275 36.010
dtype: float64
[145]: """CONCLUSION :
The predicted stock prices values have been stored in the Forecast Columns of the each stock entity dataframe"""
[145]: 'CONCLUSION :\nThe predicted stock prices values have been stored in the Forecast Columns of the each stock entity dataframe'
[ ]:
Type here to search
ENG
00:10
29-11-2018