In [1]:

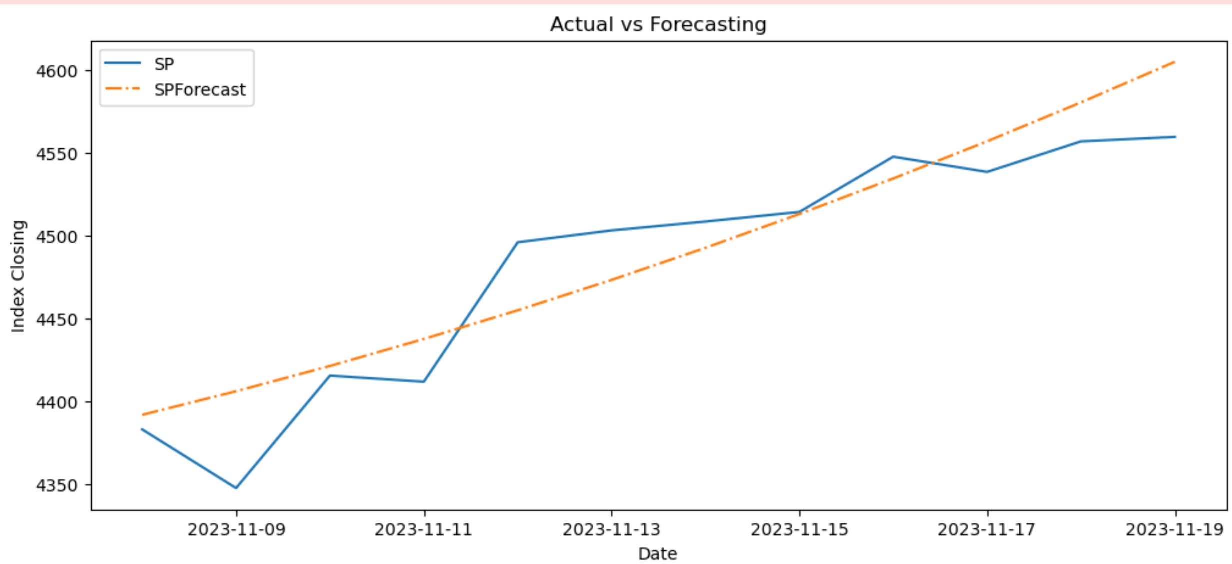
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from statsmodels.tsa.stattools import adfuller
import statsmodels.api as sm
from statsmodels.tsa.api import VAR
from statsmodels.tsa.base import  datetools
from sklearn.metrics import mean_squared_error
# importing all standard library for Vector Autoregression

df = pd.read_csv('SPDJ.csv')
# read data of S&P500, Dow Jones Index from 11/25/2013 to 11/24/2023

def adf_test(series,title=''):
    """
    Pass in a time series and an optional title, returns an ADF report
    """
    print(f'Augmented Dickey-Fuller Test: {title}')
    result = adfuller(series.dropna(),autolag='AIC') # .dropna() handles differenced d
    labels = ['ADF test statistic','p-value','# lags used','# observations']
    out = pd.Series(result[0:4],index=labels)
    for key,val in result[4].items():
        out[f'critical value ({key})']=val
    print(out.to_string())          # .to_string() removes the line "dtype: float64"
    if result[1] <= 0.05:
        print("Strong evidence against the null hypothesis")
        print("Reject the null hypothesis")
        print("Data has no unit root and is stationary")
    else:
        print("Weak evidence against the null hypothesis")
        print("Fail to reject the null hypothesis")
        print("Data has a unit root and is non-stationary")

        # Augmented Dickey Fuller test for Unit root to decide stationarity

        adf_test(df['SP'])

        adf_test(df['DJ'])

df_difference = df.diff()
# differencing daily closing as most of financial data itself are not stationary
adf_test(df_difference['SP'])
adf_test(df_difference['DJ'])

new_df_difference = df_difference.dropna(axis=0, how='any')
#dropping first line as it is n/a

test_obs = 12
train = new_df_difference[:-test_obs]
test = new_df_difference[-test_obs:]
# Last 12 data point as test points

for i in [1,2,3,4,5,6,7,8,9,10]:
    model = VAR(train)
    results = model.fit(i)
    print('Order =', i)
    print('AIC: ', results.aic)
    print('BIC: ', results.bic)
```

```
    print()
# Run AIC and BIC of VAR(1) to VAR(10)

result = model.fit(1)
result.summary()
#VAR(1) is most logical choice for simple and clean model

lagged_Values = train.values[-8:]
pred = result.forecast(y=lagged_Values, steps=12)
#forecasting 12 steps ahead

idx = pd.date_range('2023-11-08', periods=12, freq='1D')
df_forecast=pd.DataFrame(data=pred, index=idx, columns=['SP_2d', 'DJ_2d'])
#adding forecasted difference back to original index closing

df_forecast['SP1d'] = (df['SP'].iloc[-test_obs-1]-df['SP'].iloc[-test_obs-2]) + df_for
df_forecast['SPForecast'] = df['SP'].iloc[-test_obs-1] + df_forecast['SP1d'].cumsum()

df_forecast['DJ1d'] = (df['DJ'].iloc[-test_obs-1]-df['DJ'].iloc[-test_obs-2]) + df_for
df_forecast['DJForecast'] = df['DJ'].iloc[-test_obs-1] + df_forecast['DJ1d'].cumsum()

test_original = df[-test_obs:]
test_original.index = pd.to_datetime(test_original.index)

f = plt.figure()
f.set_figwidth(12)
f.set_figheight(5)

plt.plot(df_forecast['SPForecast'].index, test_original['SP'], label ='SP')
plt.plot(df_forecast['SPForecast'].index, df_forecast['SPForecast'], '-.', label ='SPF

plt.xlabel("Date")
plt.ylabel("Index Closing")
plt.legend()
plt.title('Actual vs Forecasting')

plt.show()

MSE_SP = mean_squared_error(test_original['SP'], df_forecast['SPForecast'])
print(" Mean Squared Error of S&P Forecasting vs actual = ", MSE_SP)

f = plt.figure()
f.set_figwidth(12)
f.set_figheight(5)

plt.plot(df_forecast['DJForecast'].index, test_original['DJ'], label ='DJ')
plt.plot(df_forecast['DJForecast'].index, df_forecast['DJForecast'], '-.', label ='DJF

plt.xlabel("Date")
plt.ylabel("Index Closing")
plt.legend()
plt.title('Actual vs Forecasting')

plt.show()

MSE_DJ = mean_squared_error(test_original['DJ'], df_forecast['DJForecast'])
print(" Mean Squared Error of DJ Index Forecasting vs actual = ", MSE_DJ)

# plotting forecasted valued of S&P and Dow Index against actual indices
# calculating mean squared errors of S&P & Dow foreastings vs actuals
```

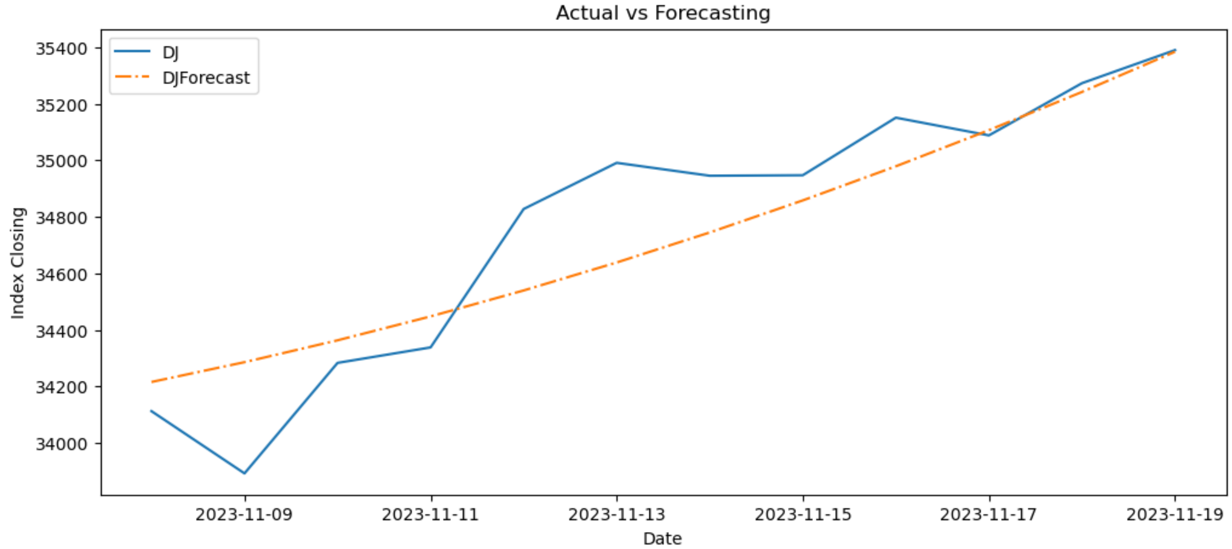
```
Augmented Dickey-Fuller Test:
ADF test statistic     -1.572177e+01
p-value                 1.321288e-28
# lags used             9.000000e+00
# observations          2.506000e+03
critical value (1%)    -3.432962e+00
critical value (5%)    -2.862694e+00
critical value (10%)   -2.567384e+00
Strong evidence against the null hypothesis
Reject the null hypothesis
Data has no unit root and is stationary
Augmented Dickey-Fuller Test:
ADF test statistic     -1.565104e+01
p-value                 1.596371e-28
# lags used             9.000000e+00
# observations          2.506000e+03
critical value (1%)    -3.432962e+00
critical value (5%)    -2.862694e+00
critical value (10%)   -2.567384e+00
Strong evidence against the null hypothesis
Reject the null hypothesis
Data has no unit root and is stationary
Order = 1
AIC:  16.086664317766086
BIC:  16.100628149867003

Order = 2
AIC:  16.081833051120153
BIC:  16.105113809277736

Order = 3
AIC:  16.084552300518894
BIC:  16.11715615618954

Order = 4
AIC:  16.08428213729433
BIC:  16.126215268572498

Order = 5
AIC:  16.08515356323937
BIC:  16.13642215486737

Order = 6
AIC:  16.079894194369633
BIC:  16.140504437747463

Order = 7
AIC:  16.06642149452195
BIC:  16.13637958771714

Order = 8
AIC:  16.063054049984775
BIC:  16.14236619774224

Order = 9
AIC:  16.054513415021052
BIC:  16.143185828772978

Order = 10
AIC:  16.05608462475179
```

```
BIC:  16.154123522627522
```

```
C:\Users\New User\anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:471:
ValueWarning: An unsupported index was provided and will be ignored when e.g. forecas
ting.
  self._init_dates(dates, freq)
C:\Users\New User\anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:471:
ValueWarning: An unsupported index was provided and will be ignored when e.g. forecas
ting.
  self._init_dates(dates, freq)
C:\Users\New User\anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:471:
ValueWarning: An unsupported index was provided and will be ignored when e.g. forecas
ting.
  self._init_dates(dates, freq)
C:\Users\New User\anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:471:
ValueWarning: An unsupported index was provided and will be ignored when e.g. forecas
ting.
  self._init_dates(dates, freq)
C:\Users\New User\anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:471:
ValueWarning: An unsupported index was provided and will be ignored when e.g. forecas
ting.
  self._init_dates(dates, freq)
C:\Users\New User\anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:471:
ValueWarning: An unsupported index was provided and will be ignored when e.g. forecas
ting.
  self._init_dates(dates, freq)
C:\Users\New User\anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:471:
ValueWarning: An unsupported index was provided and will be ignored when e.g. forecas
ting.
  self._init_dates(dates, freq)
C:\Users\New User\anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:471:
ValueWarning: An unsupported index was provided and will be ignored when e.g. forecas
ting.
  self._init_dates(dates, freq)
C:\Users\New User\anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:471:
ValueWarning: An unsupported index was provided and will be ignored when e.g. forecas
ting.
  self._init_dates(dates, freq)
C:\Users\New User\anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:471:
ValueWarning: An unsupported index was provided and will be ignored when e.g. forecas
ting.
  self._init_dates(dates, freq)
```



```
Mean Squared Error of S&P Forecasting vs actual =  845.6290282401527
```




```
Mean Squared Error of DJ Index Forecasting vs actual =  39237.28143171107
```

In [ ]: