Time_Series_Project_Code

November 18, 2021

1 Importing Necessary Librarires

[26]:

```
import requests
from functools import reduce
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

#cointegration test
from statsmodels.tsa.stattools import coint
from statsmodels.tsa.vector_ar.vecm import coint_johansen

#causality test
from statsmodels.tsa.stattools import grangercausalitytests

%matplotlib inline
```

2 Importing Data

[27]:

```
btc = pd.read_csv('https://query1.finance.yahoo.com/v7/finance/download/BTC-USD?
 ↪period1=1410912000&period2=1635984000&interval=1d&events=history&includeAdjustedClose=true'
eth = pd.read_csv('https://query1.finance.yahoo.com/v7/finance/download/ETH-USD?
 ↪period1=1438905600&period2=1635984000&interval=1d&events=history&includeAdjustedClose=true'
bnc = pd.read_csv('https://query1.finance.yahoo.com/v7/finance/download/BNB-USD?
 ↪period1=1500940800&period2=1635984000&interval=1d&events=history&includeAdjustedClose=true'



btc = btc[['Date','Close']]
eth = eth[['Date','Close']]
bnc = bnc[['Date','Close']]

df = reduce(lambda  left,right: pd.merge(left,right,on=['Date']), [btc,eth,bnc])
df_btc_eth = pd.merge(btc, eth, on = 'Date')
df_btc_bnc = pd.merge(btc, bnc, on = 'Date')
```

```
df.columns = ['Date', 'btc', 'eth', 'bnc']
df_btc_eth.columns = ['Date', 'btc', 'eth']
df_btc_bnc.columns = ['Date', 'btc', 'bnc']

df.Date = pd.to_datetime(df.Date)
df_btc_eth.Date = pd.to_datetime(df_btc_eth.Date)
df_btc_bnc.Date = pd.to_datetime(df_btc_bnc.Date)

df.dropna(inplace = True)
df_btc_eth.dropna(inplace = True)
df_btc_bnc.dropna(inplace = True)
```

[3]:

```
print('Toatal merged dataframe length : ',len(df))
df.tail()
```

```
Toatal merged dataframe length :  1560
```

[3]:

```
           Date           btc          eth         bnc
1559 2021-10-31  61318.957031  4288.074219  524.364441
1560 2021-11-01  61004.406250  4324.626953  551.255920
1561 2021-11-02  63226.402344  4584.798828  554.447632
1562 2021-11-03  62970.046875  4607.193848  568.578796
1563 2021-11-04  61452.230469  4537.324219  559.737305
```

[4]:

```
print('Bitcoin-Etherium Data length: ', len(df_btc_eth))
df_btc_eth.tail()
```

```
Bitcoin-Etherium Data length:  2278
```

[4]:

```
           Date           btc          eth
2277 2021-10-31  61318.957031  4288.074219
2278 2021-11-01  61004.406250  4324.626953
2279 2021-11-02  63226.402344  4584.798828
2280 2021-11-03  62970.046875  4607.193848
2281 2021-11-04  61452.230469  4537.324219
```

[5]:

```
print('Bitcoin-Binance Data Length',len(df_btc_bnc))
df_btc_bnc.tail()
```

```
Bitcoin-Binance Data Length 1560
```

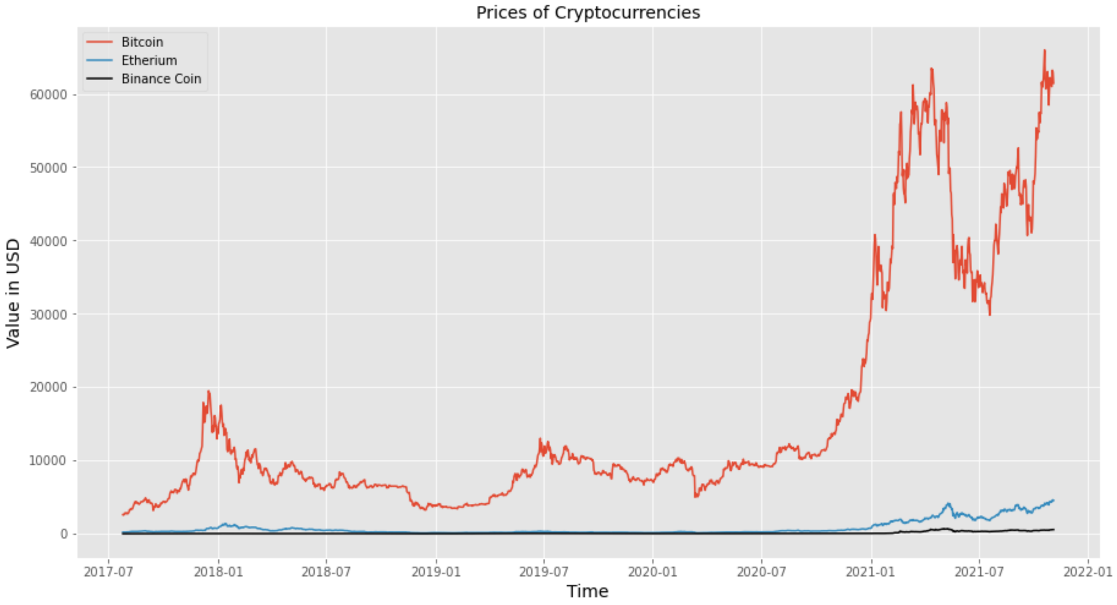
[5]:

```
           Date           btc         bnc
1559 2021-10-31  61318.957031  524.364441
1560 2021-11-01  61004.406250  551.255920
1561 2021-11-02  63226.402344  554.447632
1562 2021-11-03  62970.046875  568.578796
1563 2021-11-04  61452.230469  559.737305
```

3 Visualisation

[6]:

```
plt.style.use('ggplot')
plt.figure(figsize = (15,8))
plt.title('Prices of Cryptocurrencies', color = 'k', fontsize = 14)
plt.ylabel('Value in USD', color = 'k', fontsize = 14)
plt.xlabel('Time', color = 'k', fontsize = 14)
plt.plot(df['Date'],df['btc'], label = 'Bitcoin')
plt.plot(df['Date'],df['eth'], label = 'Etherium')
plt.plot(df['Date'],df['bnc'], label = 'Binance Coin', color = 'k')
plt.legend(loc = 'upper left')
plt.show()
```



[7]:

```
import plotly.express as px
df = df.set_index('Date').rename_axis('cryptocurrency', axis=1)

fig = px.line(df, facet_col="cryptocurrency", facet_col_wrap=1)
fig.update_yaxes(matches=None)
fig.show()
```

[8]:

```
fig = px.area(df, facet_col='cryptocurrency', facet_col_wrap=1)
fig.update_yaxes(matches=None)
fig.show()
```

4 Stationarity Test

[9]:

```python
from statsmodels.tsa.stattools import adfuller

def adf_test(df):
    result = adfuller(df.values)
    print('ADF Statistics: %f' % result[0])
    print('p-value: %f' % result[1])
    print('Critical values:')
    for key, value in result[4].items():
        print('\t%s: %.3f' % (key, value))

print('ADF Test: Bitcoin time series')
adf_test(df['btc'])
print('')
print('ADF Test: Etherium time series')
adf_test(df['eth'])
print('')
print('ADF Test: Binance Coin time series')
adf_test(df['bnc'])
```

```
ADF Test: Bitcoin time series
ADF Statistics: 0.497392
p-value: 0.984794
Critical values:
        1%: -3.435
        5%: -2.863
        10%: -2.568

ADF Test: Etherium time series
ADF Statistics: 1.898664
p-value: 0.998526
Critical values:
        1%: -3.435
        5%: -2.863
        10%: -2.568

ADF Test: Binance Coin time series
ADF Statistics: 0.193097
p-value: 0.971872
Critical values:
        1%: -3.435
        5%: -2.863
        10%: -2.568
```

[10]:

```python
df_differenced = df.diff().dropna()

fig = px.line(df_differenced, facet_col="cryptocurrency", facet_col_wrap=1)
```

```
fig.update_yaxes(matches=None)
fig.show()
```

[11]:

```
print('ADF Test: Bitcoin time series transformed')
adf_test(df_differenced['btc'])
print('')
print('ADF Test: Etherium time series transformed')
adf_test(df_differenced['eth'])
print('')
print('ADF Test: Binance Coin time series transformed')
adf_test(df_differenced['bnc'])
```

```
ADF Test: Bitcoin time series transformed
ADF Statistics: -8.252043
p-value: 0.000000
Critical values:
        1%: -3.435
        5%: -2.863
        10%: -2.568

ADF Test: Etherium time series transformed
ADF Statistics: -11.127701
p-value: 0.000000
Critical values:
        1%: -3.435
        5%: -2.863
        10%: -2.568

ADF Test: Binance Coin time series transformed
ADF Statistics: -8.013770
p-value: 0.000000
Critical values:
        1%: -3.435
        5%: -2.863
        10%: -2.568
```

5 Co-integration Test

5.1 Engle-Granger Causality Test

[30]:

```
from statsmodels.api import OLS
from statsmodels.tsa.stattools import adfuller

def print_adf_test_result(series):
    adf, pvalue, _, _, _, _ = adfuller(series)
    print(f"Test Statistics: {adf}\np-value: {pvalue}")

```

```
model = OLS(df_btc_eth['btc'], df_btc_eth['eth'])
res = model.fit()

# print(res.params[0])
err = df_btc_eth['btc'] - res.params[0] * df_btc_eth['eth']
print_adf_test_result(err)

model = OLS(df_btc_bnc['btc'], df_btc_bnc['bnc'])
res = model.fit()

# print(res.params[0])
err = df_btc_bnc['btc'] - res.params[0] * df_btc_bnc['bnc']
print_adf_test_result(err)
```

```
Test Statistics: -3.1121059433534177
p-value: 0.02567016974030872
Test Statistics: -3.057243438119709
p-value: 0.029883473062010103
```

5.2 Johansen's Test

[38]:

```
from statsmodels.tsa.vector_ar.vecm import coint_johansen

def print_johansen_test_result(*args, **kwargs):
    result = coint_johansen(*args, **kwargs)
    print('Trace Statistics:')
    print('variable statistic Crit-90% Crit-95%  Crit-99%')
    for i in range(len(result.lr1)):
        print('r =', i, '\t', round(result.lr1[i], 4), result.cvt[i, 0], result.cvt[i, 1], result.cvt[i, 2])
    print('--------------------------------------------------')
    print('--> Eigen Statistics')
    print('variable statistic Crit-90% Crit-95%  Crit-99%')
    for i in range(len(result.lr2)):
        print('r =', i, '\t', round(result.lr2[i], 4), result.cvm[i, 0], result.cvm[i, 1], result.cvm[i, 2])
    print('--------------------------------------------------')
    print('eigenvectors:\n', result.evec)
    print('--------------------------------------------------')
    print('eigenvalues:\n', result.eig)
    print('--------------------------------------------------')

print_johansen_test_result(df.drop(['Date'],axis = 1), 1, 1)
```

```
Trace Statistics:
variable statistic Crit-90% Crit-95%  Crit-99%
r = 0    79.9731 32.0645 35.0116 41.0815
r = 1    36.2703 16.1619 18.3985 23.1485
```

```
r = 2    0.2537 2.7055 3.8415 6.6349
--------------------------------------------------
--> Eigen Statistics
variable statistic Crit-90% Crit-95%  Crit-99%
r = 0    43.7028 21.8731 24.2522 29.2631
r = 1    36.0166 15.0006 17.1481 21.7465
r = 2    0.2537 2.7055 3.8415 6.6349
--------------------------------------------------
eigenvectors:
 [[ 1.41942657e-05  1.84264074e-04 -1.46224251e-05]
 [ 2.03029257e-03 -1.76194816e-03  2.34657934e-03]
 [-2.06708702e-02 -3.04795525e-03 -7.45071957e-03]]
--------------------------------------------------
eigenvalues:
 [0.02766082 0.02285204 0.00016282]
--------------------------------------------------
```

6 Causality Test

[16]:
```
maxlag=15
test = 'ssr_chi2test'

def grangers_causation_matrix(data, variables, test='ssr_chi2test',␣
 ↪verbose=False):

    df = pd.DataFrame(np.zeros((len(variables), len(variables))),␣
 ↪columns=variables, index=variables)
    for c in df.columns:
        for r in df.index:
            test_result = grangercausalitytests(data[[r, c]], maxlag=maxlag,␣
 ↪verbose=False)
            p_values = [round(test_result[i+1][0][test][1],4) for i in␣
 ↪range(maxlag)]
            if verbose: print(f'Y = {r}, X = {c}, P Values = {p_values}')
            min_p_value = np.min(p_values)
            df.loc[r, c] = min_p_value
    df.columns = [var + '_x' for var in variables]
    df.index = [var + '_y' for var in variables]
    return df

grangers_causation_matrix(df_differenced, variables = df_differenced.columns)
```

[16]:
```
       btc_x  eth_x  bnc_x
btc_y    1.0    0.0    0.0
eth_y    0.0    1.0    0.0
bnc_y    0.0    0.0    1.0
```