PROBLEM_B

February 4, 2023

0.1 Problem B

0.1.1 B1

[6]:
```
#import  depepndencies

import numpy as np
import pandas as pd
from pandas import Series, DataFrame
import matplotlib.pyplot as plt
import mplfinance as mpf

import scipy
from scipy import stats

% matplotlib inline
```

```
UsageError: Line magic function `%` not found.
```

[7]:
```
# load the data

apple_model = pd.read_csv(r'/home/dsm/PART_A/AAPL.csv')
ibm_model = pd.read_csv(r'/home/dsm/PART_A/IBM.csv')
msft_model = pd.read_csv(r'/home/dsm/PART_A/MSFT.csv')


apple = apple_model.head(252)
ibm = ibm_model.head(252)
msft = msft_model.head(252)
```

[8]:
```
#print finance entries

print(apple)
```

```
          Date        Open        High         Low       Close   Adj Close  \
0   2022-02-03  174.479996  176.240005  172.119995  172.899994  171.902313
1   2022-02-04  171.679993  174.100006  170.679993  172.389999  171.613632
2   2022-02-07  172.860001  173.949997  170.949997  171.660004  170.886917
3   2022-02-08  171.729996  175.350006  171.429993  174.830002  174.042633
```

```
4    2022-02-09  176.050003  176.649994  174.899994  176.279999  175.486115
..          …           …           …           …           …           …
246  2023-01-27  143.160004  147.229996  143.080002  145.929993  145.929993
247  2023-01-30  144.960007  145.550003  142.850006  143.000000  143.000000
248  2023-01-31  142.699997  144.339996  142.279999  144.289993  144.289993
249  2023-02-01  143.970001  146.610001  141.320007  145.429993  145.429993
250  2023-02-02  148.899994  151.179993  148.169998  150.820007  150.820007

        Volume
0     89418100
1     82465400
2     77251200
3     74829200
4     71285000
..         …
246   70492800
247   64015300
248   65874500
249   77663600
250  116868600

[251 rows x 7 columns]
```

[9]:
```
print(ibm)
```

```
           Date        Open        High         Low       Close   Adj Close  \
0    2022-02-03  137.000000  138.759995  135.830002  137.779999  131.269730
1    2022-02-04  137.860001  138.820007  136.220001  137.149994  130.669495
2    2022-02-07  137.449997  137.820007  136.270004  137.240005  130.755249
3    2022-02-08  137.229996  137.520004  135.779999  137.020004  130.545639
4    2022-02-09  137.839996  138.350006  136.830002  137.789993  131.279236
..          …           …           …           …           …           …
246  2023-01-27  134.440002  135.490005  133.770004  134.389999  134.389999
247  2023-01-30  134.320007  136.110001  133.979996  135.300003  135.300003
248  2023-01-31  135.500000  135.649994  133.759995  134.729996  134.729996
249  2023-02-01  134.490005  135.789993  132.800003  135.089996  135.089996
250  2023-02-02  135.960007  136.720001  134.850006  136.389999  136.389999

      Volume
0    6100800
1    4142000
2    3759000
3    4181800
4    5393500
..       …
246  8140700
247  5375700
248  7206400
```

```
249   5428900
250   6104900

[251 rows x 7 columns]
```

[10]:
```
print(msft)
```

```
           Date        Open        High         Low       Close   Adj Close  \
0    2022-02-03  309.489990  311.230011  299.959991  301.250000  298.453430
1    2022-02-04  300.209991  308.799988  299.970001  305.940002  303.099884
2    2022-02-07  306.170013  307.839996  299.899994  300.950012  298.156219
3    2022-02-08  301.250000  305.559998  299.950012  304.559998  301.732697
4    2022-02-09  309.869995  311.929993  307.390015  311.209991  308.320984
..          …           …           …           …           …           …
246  2023-01-27  248.990005  249.830002  246.830002  248.160004  248.160004
247  2023-01-30  244.509995  245.600006  242.199997  242.710007  242.710007
248  2023-01-31  243.449997  247.949997  242.949997  247.809998  247.809998
249  2023-02-01  248.000000  255.179993  245.470001  252.750000  252.750000
250  2023-02-02  258.820007  264.690002  257.250000  264.600006  264.600006

       Volume
0    43730000
1    35096500
2    28533300
3    32421200
4    31284700
..        …
246  26480800
247  25867400
248  26541100
249  31259900
250  39855300

[251 rows x 7 columns]
```

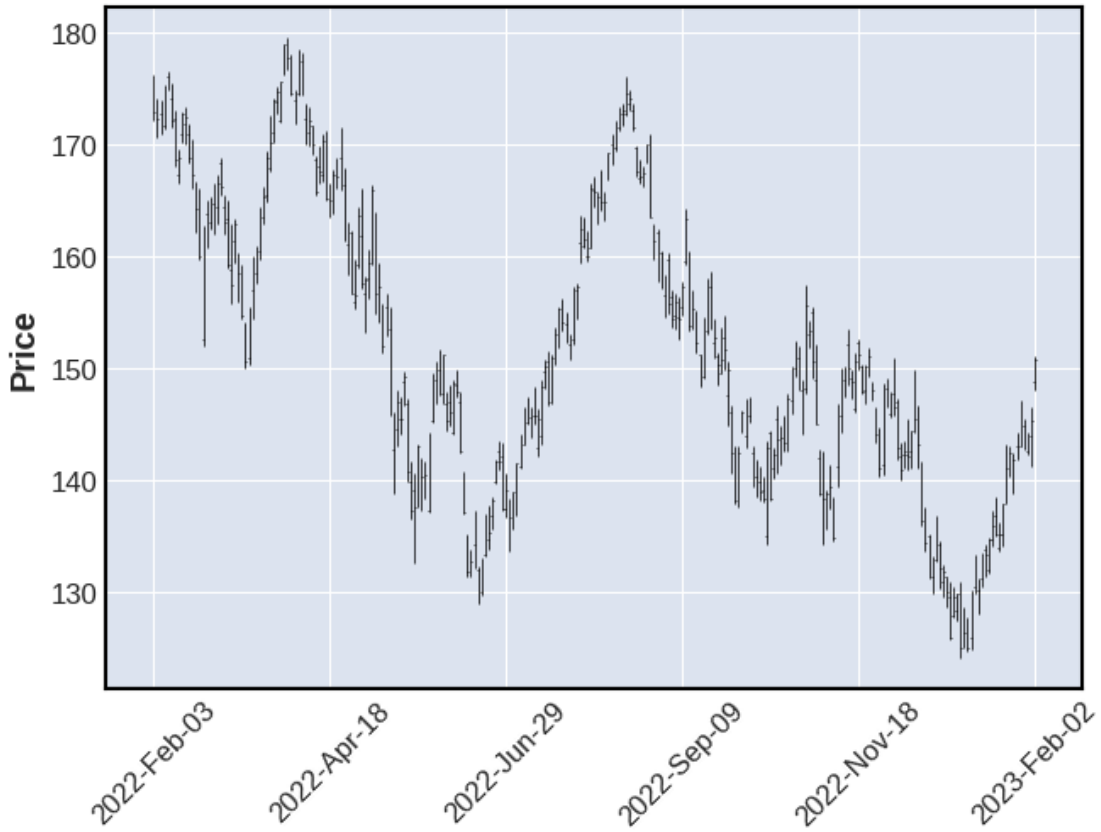
[11]:
```
#indicate columns

apple.columns
```

[11]:
```
Index(['Date', 'Open', 'High', 'Low', 'Close', 'Adj Close', 'Volume'],
dtype='object')
```

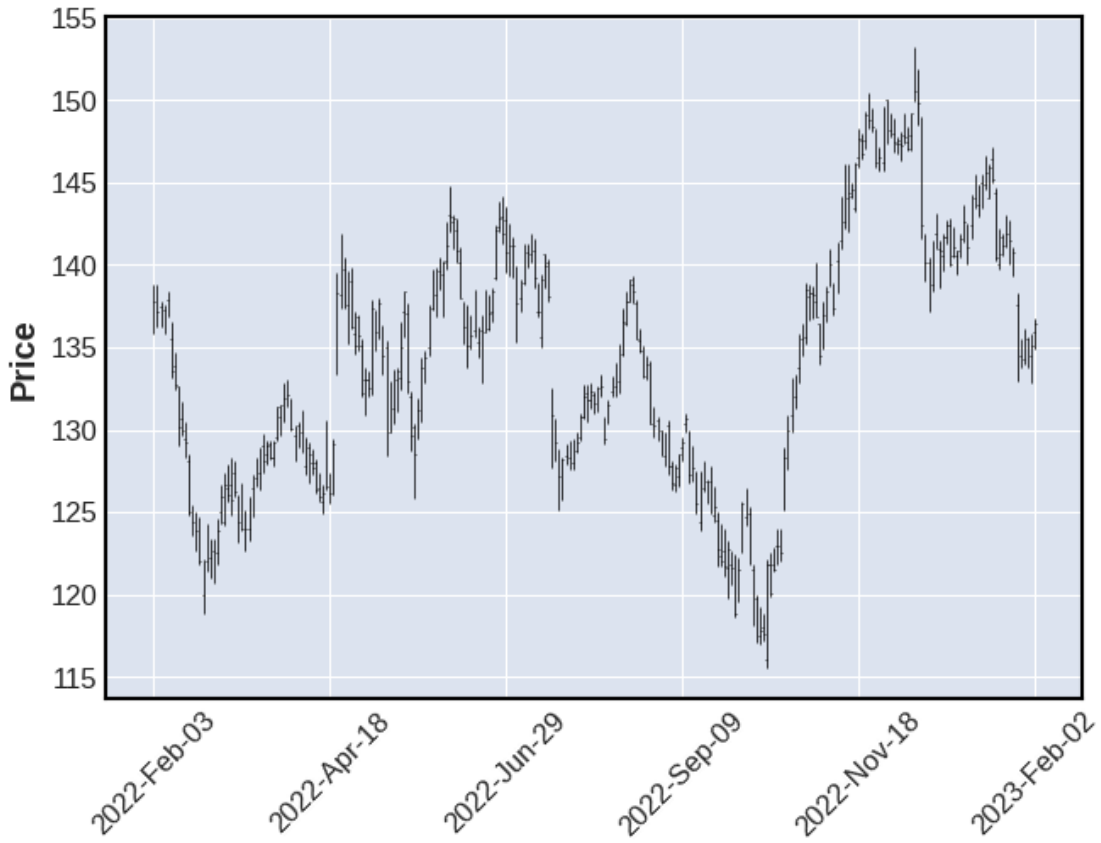
[12]:
```
apple.Date = pd.to_datetime(apple.Date)
ibm.Date = pd.to_datetime(ibm.Date)
msft.Date = pd.to_datetime(msft.Date)
```

[13]:
```
apple = apple.set_index('Date')
ibm = ibm.set_index('Date')
msft = msft.set_index('Date')
```

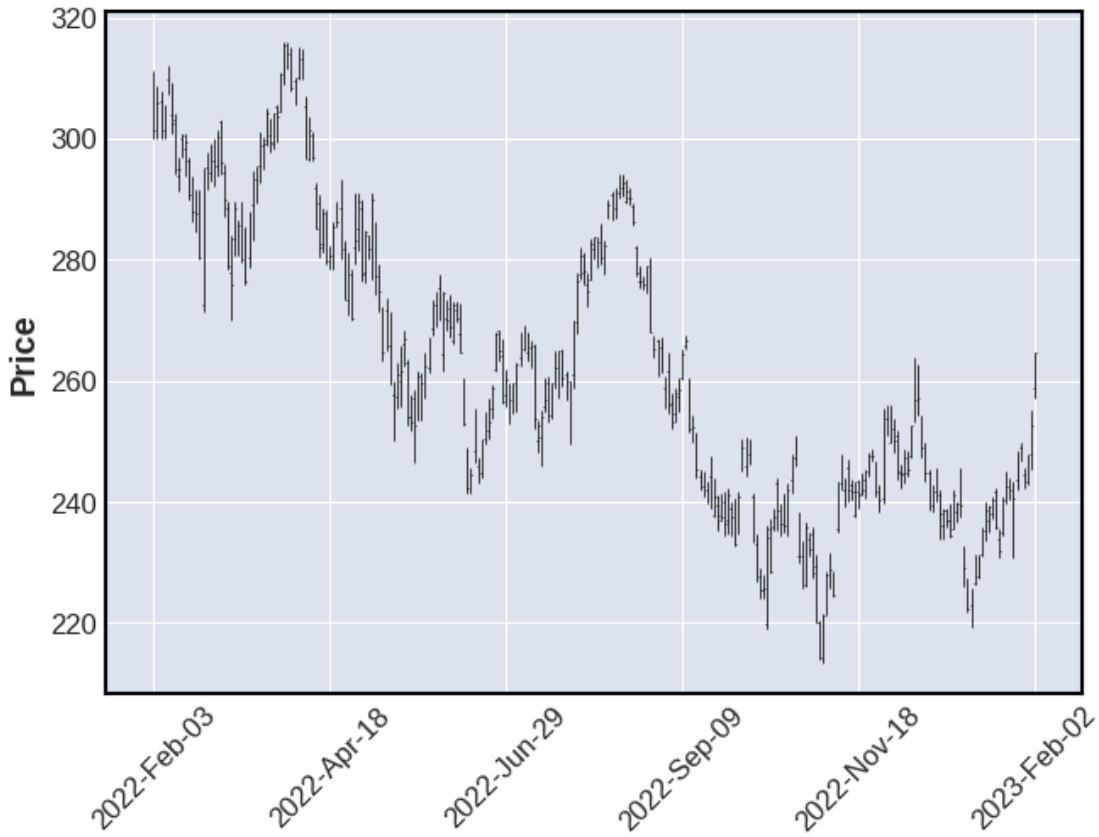
[14]:
```
mpf.plot(apple)
```



[15]:
```
mpf.plot(ibm)
```



[16]:
```
mpf.plot(msft)
```



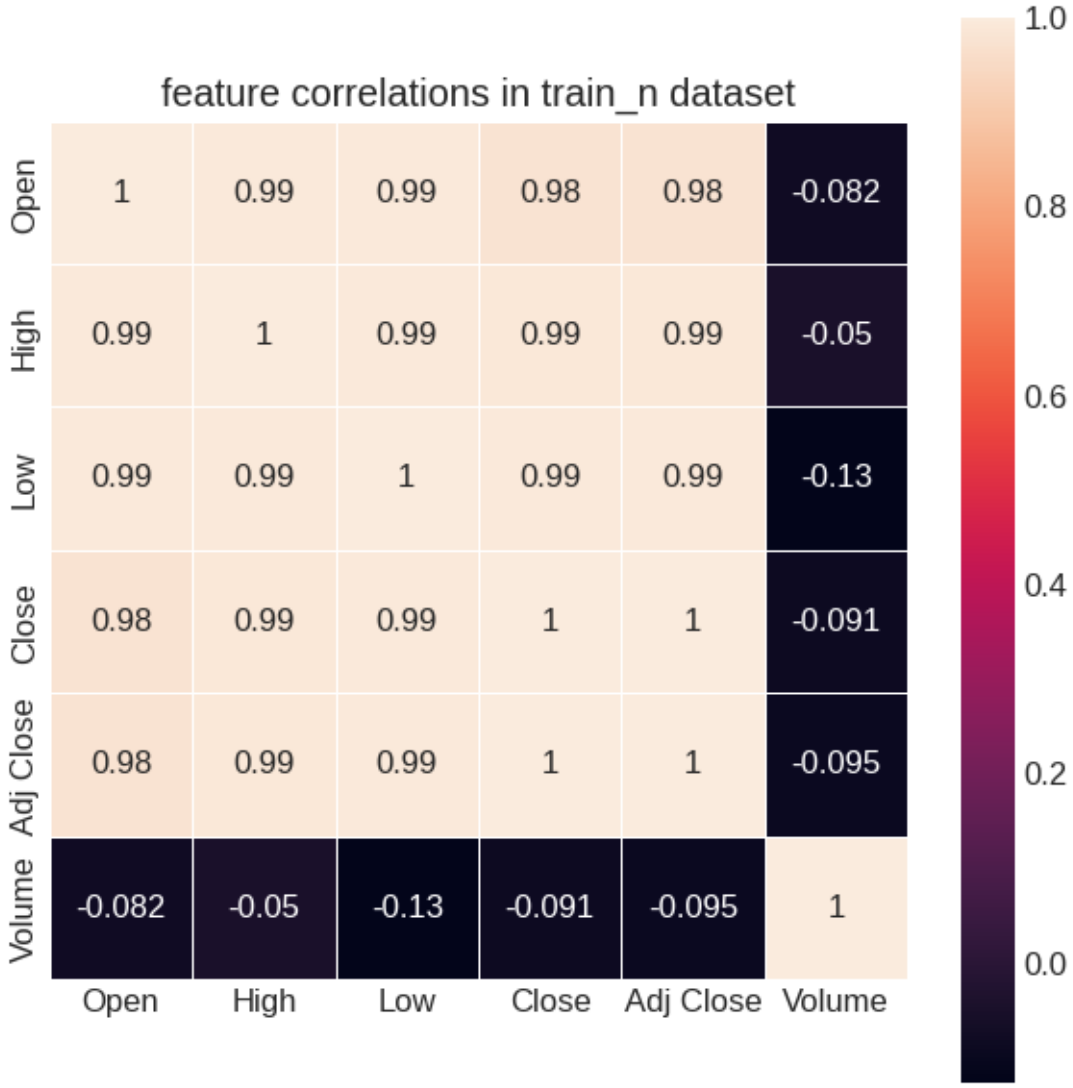
0.1.2 B2

[64]:
```
from speedml import Speedml
```

[66]:
```
# PLOTTING SML

sml = Speedml('AAPL.csv' ,  'MSFT.csv',
              target='Close', uid='Date')
sml.plot.correlate()
sml.eda()
```

[66]:
```
                                                          Results  \
Speedml Release                                            v0.9.3
Shape                                  train (251, 6) | test (251, 6)
Numerical Continuous      [Open, High, Low, Close, Adj Close, Volume]
Target Analysis (Close)                                  Model ready.

                                                        Observations
Speedml Release           Visit https://speedml.com for release notes.
Shape
```

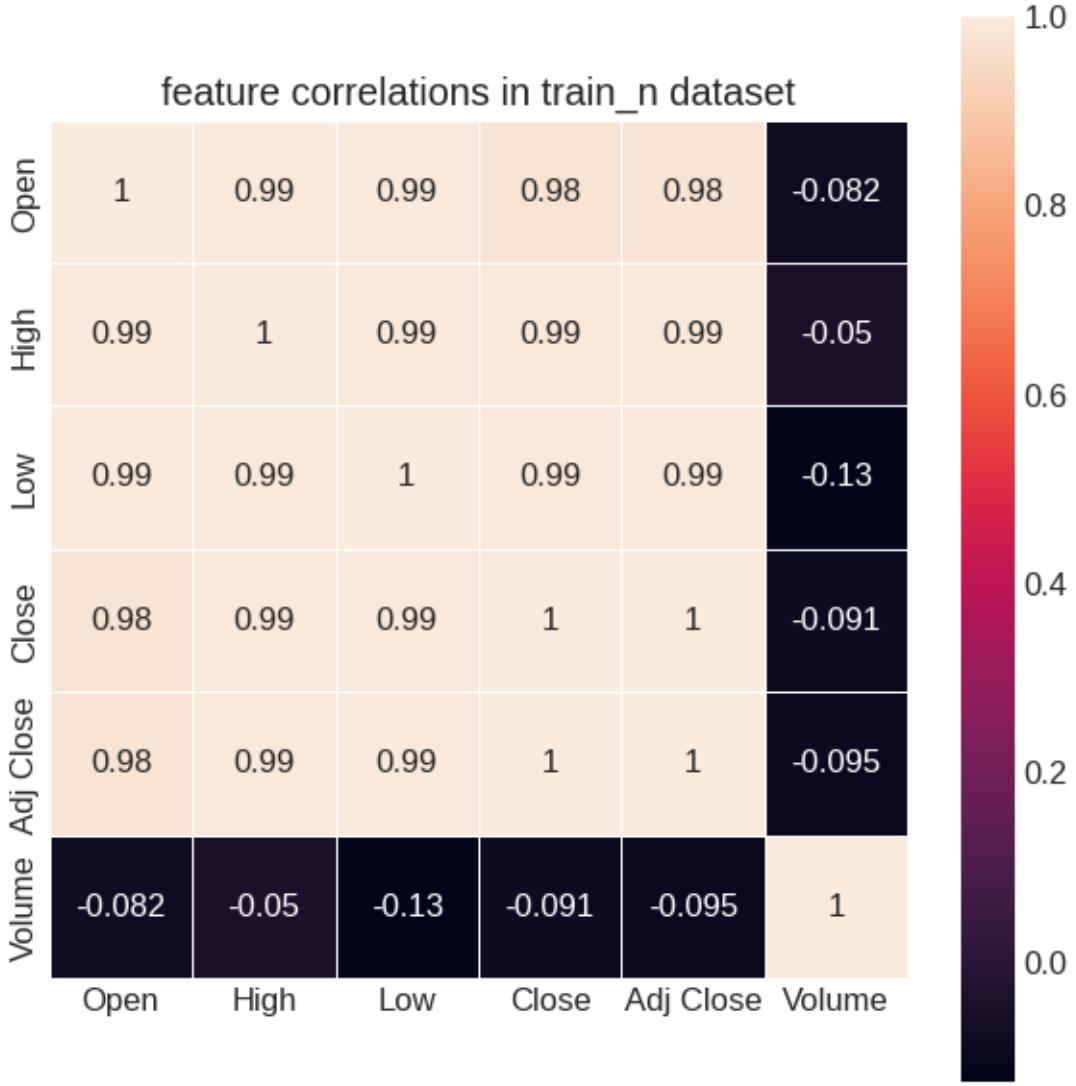
```
Numerical Continuous                    ~80% unique. Use plot.continuous.
Target Analysis (Close)                              Use regression models.
```



[67]:
```
# PLOTTING SML

sml = Speedml('AAPL.csv' ,  'IBM.csv',
              target='Close', uid='Date')
sml.plot.correlate()
sml.eda()
```

[67]:
```
                                                          Results  \
Speedml Release                                            v0.9.3
Shape                                  train (251, 6) | test (251, 6)
```

```
Numerical Continuous      [Open, High, Low, Close, Adj Close, Volume]
Target Analysis (Close)                                  Model ready.

                                                          Observations
Speedml Release          Visit https://speedml.com for release notes.
Shape
Numerical Continuous                ~80% unique. Use plot.continuous.
Target Analysis (Close)                        Use regression models.
```



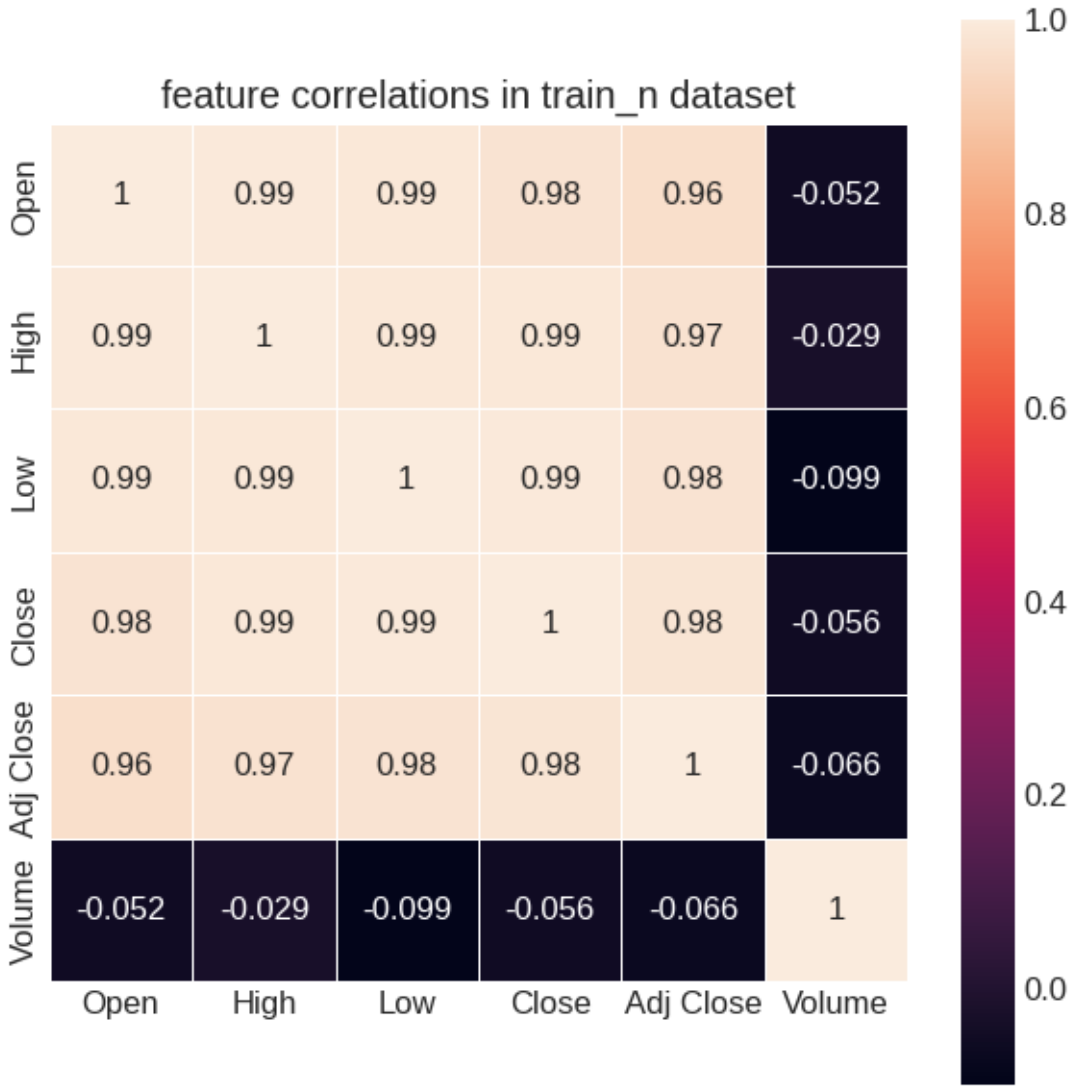
[68]:
```
# PLOTTING SML

sml = Speedml('IBM.csv' ,  'MSFT.csv',
              target='Close', uid='Date')
```

```
sml.plot.correlate()
sml.eda()
```

[68]:

	Results
Speedml Release	v0.9.3
Outliers Upper	[Volume]
Shape	train (251, 6) | test (251, 6)
Numerical Continuous	[Open, High, Low, Close, Adj Close, Volume]
Target Analysis (Close)	Model ready.

	Observations
Speedml Release	Visit https://speedml.com for release notes.
Outliers Upper	Positive skew (> 3). Use feature.outliers(upper).
Shape	
Numerical Continuous	~80% unique. Use plot.continuous.
Target Analysis (Close)	Use regression models.



0.1.3 B3

[53]:
```
import yfinance as yf
import pandas as pd

ticker = yf.Tickers('msft aapl ibm')

def get_historical_data(ticker, start_date):
    data = yf.download(ticker, start=start_date)
    #daily return
    data['Daily Return'] = data ['Adj Close'].pct_change()
    return data.dropna()
```

[54]:
```
def std_dev(data):
    #get number of observation
    n=len(data)
    #calculate mean
    mean=sum(data)/n
    #calculate deviations from mean
    deviations = sum([(x-mean)**2 for x in data])
    #calculate variance&standard deviation
    variance = deviations/(n-1)
    s = variance**(1/2)
    return s
```

[55]:
```
# calculate sharpe ratio

def sharpe_ratio(data, risk_rate=0.0):
    # Calculate average daily return
    mean_daily_return = sum(data)/len(data)
    # Calculate Standard Deviation
    s = std_dev(data)
    # Calculate Daily Sharpe Ratio
    daily_sharpe_ratio = (mean_daily_return - risk_rate)/s
    #Annualize Daily Sharpe Ratio
    sharpe_ratio = 252**(1/2)*daily_sharpe_ratio
    return sharpe_ratio
```

[56]:
```
AAPL = get_historical_data('AAPL', start_date='2022-02-03')
sharpe_ratio(AAPL['Daily Return'])
```

```
[**********************100%**********************]  1 of 1 completed
```

[56]:
```
-0.12577470355156317
```

[59]:
```
MSFT = get_historical_data('MSFT', start_date='2022-02-03')
sharpe_ratio(MSFT['Daily Return'])
```

```
[**********************100%**********************]  1 of 1 completed
```

[59]:
```
-0.2294137261580235
```

[60]:
```
IBM = get_historical_data('IBM', start_date='2022-02-03')
sharpe_ratio(IBM['Daily Return'])
```

```
[**********************100%**********************]  1 of 1 completed
```

[60]:
```
0.29787070806813176
```

0.1.4 B4

[70]:
```
AAPL
```

[70]:
```
                                 Open        High         Low       Close  \
Date
2022-02-04 00:00:00-05:00  171.679993  174.100006  170.679993  172.389999
2022-02-07 00:00:00-05:00  172.860001  173.949997  170.949997  171.660004
2022-02-08 00:00:00-05:00  171.729996  175.350006  171.429993  174.830002
2022-02-09 00:00:00-05:00  176.050003  176.649994  174.899994  176.279999
2022-02-10 00:00:00-05:00  174.139999  175.479996  171.550003  172.119995
…                                 …           …           …           …
2023-01-30 00:00:00-05:00  144.960007  145.550003  142.850006  143.000000
2023-01-31 00:00:00-05:00  142.699997  144.339996  142.279999  144.289993
2023-02-01 00:00:00-05:00  143.970001  146.610001  141.320007  145.429993
2023-02-02 00:00:00-05:00  148.899994  151.179993  148.169998  150.820007
2023-02-03 00:00:00-05:00  148.029999  157.380005  147.830002  154.500000

                           Adj Close     Volume  Daily Return
Date
2022-02-04 00:00:00-05:00  171.613632   82465400     -0.001679
2022-02-07 00:00:00-05:00  170.886917   77251200     -0.004235
2022-02-08 00:00:00-05:00  174.042633   74829200      0.018467
2022-02-09 00:00:00-05:00  175.486099   71285000      0.008294
2022-02-10 00:00:00-05:00  171.344833   90865900     -0.023599
…                                 …          …             …
2023-01-30 00:00:00-05:00  143.000000   64015300     -0.020078
2023-01-31 00:00:00-05:00  144.289993   65874500      0.009021
2023-02-01 00:00:00-05:00  145.429993   77663600      0.007901
2023-02-02 00:00:00-05:00  150.820007  118339000      0.037063
2023-02-03 00:00:00-05:00  154.500000  149918017      0.024400

[251 rows x 7 columns]
```

MSFT

[71]:
```
IBM
```

[71]:
```
                                 Open        High         Low       Close  \
Date
2022-02-04 00:00:00-05:00  137.860001  138.820007  136.220001  137.149994
2022-02-07 00:00:00-05:00  137.449997  137.820007  136.270004  137.240005
2022-02-08 00:00:00-05:00  137.229996  137.520004  135.779999  137.020004
2022-02-09 00:00:00-05:00  137.839996  138.350006  136.830002  137.789993
2022-02-10 00:00:00-05:00  135.470001  136.559998  133.169998  133.520004
…                                 …           …           …           …
2023-01-30 00:00:00-05:00  134.320007  136.110001  133.979996  135.300003
2023-01-31 00:00:00-05:00  135.500000  135.649994  133.759995  134.729996
```

```
2023-02-01 00:00:00-05:00  134.490005  135.789993  132.800003  135.089996
2023-02-02 00:00:00-05:00  135.960007  136.720001  134.850006  136.389999
2023-02-03 00:00:00-05:00  136.350006  136.949997  135.529999  136.940002

                            Adj Close   Volume  Daily Return
Date
2022-02-04 00:00:00-05:00  130.669495  4142000     -0.004573
2022-02-07 00:00:00-05:00  130.755249  3759000      0.000656
2022-02-08 00:00:00-05:00  130.545654  4181800     -0.001603
2022-02-09 00:00:00-05:00  131.279236  5393500      0.005619
2022-02-10 00:00:00-05:00  128.743332  5978600     -0.019317
…                                …       …           …
2023-01-30 00:00:00-05:00  135.300003  5375700      0.006771
2023-01-31 00:00:00-05:00  134.729996  7206400     -0.004213
2023-02-01 00:00:00-05:00  135.089996  5428900      0.002672
2023-02-02 00:00:00-05:00  136.389999  6107800      0.009623
2023-02-03 00:00:00-05:00  136.940002  3753101      0.004033

[251 rows x 7 columns]
```

0.1.5 B5

[ ]: