DATA MINING

ASSIGNMENT: MILESTONE 3

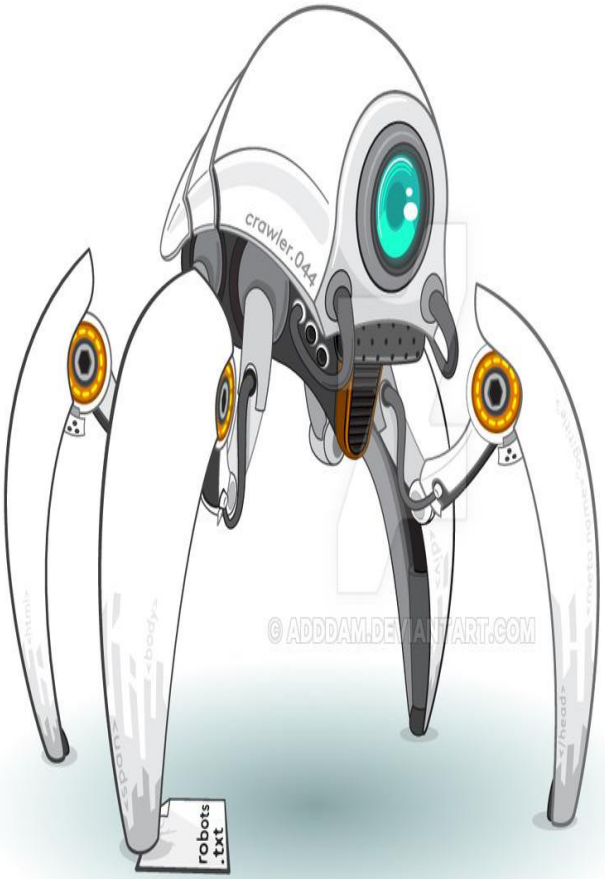
NORAISHA YUSUF (WQD180008)

NORBAIZURA MOHAMAD (WQD180043)

MARINA SHAH MUHAMMAD ZABRI TAN (WQD180011)

AMIRA AN-NUR BINTI RUSLI (WQD180016)

AMINAH SOFIA BINTI MAHAYUDIN (WQD180032)



Update

- Crawl historical stock data prices
- Compute covariance & correlation – co-moving stocks
- PAA & SAX – representation of time series data

```
# ================================================================================
# Part 2:
# Crawling historical prices (only once)
# ================================================================================

#using the stockname crawled and saved in csv. Then transform dataframe into list
df1 = pd.read_csv('stockname.csv',usecols=[1])
datanames = df1['name'].tolist()

sl=[];cl=[];ol=[];hl=[];ll=[];dl=[];vl=[];stocknames2=[]

#set timeframe to crawl e.g. 3 months
startdate=str(1546343431) #date = Tuesday, January 1, 2019 7:50:31 PM
enddate=str(1554205831) #date = Tuesday, April 2, 2019 7:50:31 PM

for name in datanames:
    url = 'https://charts.thestar.com.my/datafeed-udf/history?symbol='+name+'&reso
    r = requests.get(url).json()
    if r["s"] == "ok":
        stocknames2.append(name)
        for t in r["t"]:
            day=time.strftime("%Y-%m-%d",time.localtime(int(t)))
            dl.append(day)
            sl.append(name)
        for o in r["o"]:ol.append(o) #open price
        for c in r["c"]:cl.append(c) #closing price
        for h in r["h"]:hl.append(h) #high price
        for l in r["l"]:ll.append(l) #low price
        for v in r["v"]:vl.append(v) #volume
    print("Done for "+ name)
    #break

df = pd.DataFrame({'name':sl,'day':dl,'close':cl,'open':ol,'high':hl,'low':ll,'vol
df.to_csv('price_df.csv')
```

Part 1: Crawl historical stock data

- **3 months** worth of stock prices data for **1854 stocks**
- To focus on **closing prices, and changes between daily closing prices (dif)**

```
In [4]: df['dif'] = df.groupby('name')['close'].diff()

In [5]: print(df)
```

	name	day	close	open	high	low	dif
0	A50CHIN-C26	2019-01-02	0.185	0.200	0.200	0.185	NaN
1	A50CHIN-C26	2019-01-04	0.195	0.185	0.195	0.185	0.010
2	A50CHIN-C26	2019-01-09	0.230	0.225	0.230	0.225	0.035
3	A50CHIN-C26	2019-01-11	0.245	0.245	0.245	0.245	0.015
4	A50CHIN-C26	2019-01-18	0.260	0.260	0.260	0.260	0.015
5	A50CHIN-C26	2019-01-22	0.250	0.250	0.250	0.250	-0.010
6	A50CHIN-C26	2019-01-24	0.245	0.245	0.245	0.245	-0.005
7	A50CHIN-C26	2019-01-25	0.280	0.275	0.280	0.275	0.035
8	A50CHIN-C26	2019-01-28	0.270	0.290	0.290	0.270	-0.010
9	A50CHIN-C26	2019-01-30	0.295	0.295	0.300	0.295	0.025
10	A50CHIN-C26	2019-01-31	0.330	0.310	0.330	0.310	0.035
11	A50CHIN-C26	2019-02-13	0.365	0.340	0.365	0.340	0.035
12	A50CHIN-C26	2019-02-14	0.355	0.355	0.355	0.355	-0.010
13	A50CHIN-C26	2019-02-15	0.310	0.310	0.310	0.310	-0.045
14	A50CHIN-C26	2019-02-18	0.350	0.355	0.355	0.350	0.040
15	A50CHIN-C26	2019-02-19	0.340	0.365	0.365	0.340	-0.010
16	A50CHIN-C26	2019-02-20	0.365	0.365	0.370	0.360	0.025
17	A50CHIN-C26	2019-02-21	0.360	0.360	0.375	0.355	-0.005
18	A50CHIN-C26	2019-02-22	0.380	0.345	0.380	0.340	0.020
19	A50CHIN-C26	2019-02-25	0.495	0.410	0.495	0.410	0.115
20	A50CHIN-C26	2019-02-28	0.460	0.460	0.460	0.460	-0.035
21	A50CHIN-C26	2019-03-01	0.505	0.465	0.505	0.465	0.045
22	A50CHIN-C26	2019-03-11	0.430	0.430	0.430	0.430	-0.075
23	A50CHIN-C26	2019-03-18	0.520	0.520	0.520	0.520	0.090
24	A50CHIN-C26	2019-03-19	0.495	0.505	0.505	0.495	-0.025
25	A50CHIN-C26	2019-03-21	0.495	0.495	0.495	0.495	0.000
26	A50CHIN-C28	2019-01-02	0.085	0.085	0.085	0.085	NaN
27	A50CHIN-C28	2019-01-09	0.110	0.110	0.110	0.110	0.025
28	A50CHIN-C28	2019-01-10	0.110	0.110	0.110	0.110	0.000
29	A50CHIN-C28	2019-01-11	0.110	0.110	0.110	0.110	0.000
...	...	...	...	...	...	...	...
71192	3A	2019-02-19	0.855	0.850	0.855	0.845	0.005
71193	3A	2019-02-20	0.860	0.855	0.860	0.855	0.005
71194	3A	2019-02-21	0.860	0.850	0.860	0.835	0.000
71195	3A	2019-02-22	0.860	0.850	0.865	0.850	0.000
71196	3A	2019-02-25	0.860	0.855	0.860	0.855	0.000
71197	3A	2019-02-26	0.845	0.860	0.860	0.845	-0.015

Part 2: co-variance and correlation of stocks

Results of covariance
Sample size matters!

For various sample sizes of closing prices

```
In [12]: df_return.cov()
Out[12]:
name                    3A  A50CHIN-C26    ..
name                                       ..
3A            4.371786e-04     0.000292    ..
A50CHIN-C26   2.924015e-04     0.004949    ..
A50CHIN-C28   4.183095e-05     0.004657    ..
A50CHIN-C30   2.586507e-04     0.005320    ..
A50CHIN-C32   1.397193e-04     0.004857    ..
A50CHIN-C34  -2.988471e-04     0.000985    ..
A50CHIN-C36  -1.948861e-04     0.004684    ..
A50CHIN-C38  -1.306222e-04     0.000000    ..
A50CHIN-H25   6.163974e-05     0.000402    ..
A50CHIN-H27  -9.622552e-05    -0.003304    ..
A50CHIN-H29   2.166011e-04     0.000000    ..
AASIA         6.513705e-05     0.000097    ..
AAX           5.187487e-05    -0.000035    ..
AAX-WA        3.132140e-04     0.000163    ..
ABFMY1       -1.687660e-05    -0.000087    ..
ABLEGRP       2.327588e-04     0.000035    ..
ABMB         -2.952496e-05     0.000205    ..
```

For sample sizes of at least 60 days worth of closing prices

```
In [11]: df_return1.cov()
Out[11]:
name                    3A       AAX    ..
name                                    ..
3A            4.371786e-04  0.000052    ..
AAX           5.187487e-05  0.001458    ..
AAX-WA        3.132140e-04  0.002262    ..
ABMB         -2.952496e-05  0.000135    ..
ACOSTEC       4.172288e-06 -0.000023    ..
ADVCON        1.676773e-04  0.000305    ..
AEMULUS      -2.017843e-05  0.000072    ..
AEON         -2.991779e-05 -0.000092    ..
AEONCR       -3.231643e-06  0.000020    ..
AFFIN         6.774889e-05  0.000042    ..
AIRASIA       8.502528e-05  0.000367    ..
AIRASIAC67    4.346416e-04  0.001553    ..
AIRPORT       1.158398e-06 -0.000003    ..
AJI          -2.189591e-05 -0.000030    ..
ALAM         -6.250936e-05  0.000232    ..
ALLIANZ      -4.068133e-06  0.000029    ..
```

*The **various sample size** influence the covariance and correlation values*

Top 10 Positive Correlations
For different range of records:

name	name	
FBMKLCI-H6U	WCT-C15	1.0
POS-C23	SNTORIA-WA	1.0
	POS-C28	1.0
	PPB-CJ	1.0
AIRPORT-C5	ECOWLD-CT	1.0
TM-C40	YILAI	1.0
CHGP	ECOWLD-CT	1.0
ECOWLD-CT	KARYON	1.0
OIB	YTL-C24	1.0
ECOWLD-CT	MISC-C19	1.0

dtype: float64

It is more meaningful to interpret the co-variance & correlation of stocks ***with same sample size***

Top 10 Negative Correlations
For stocks with >60 records:

name	name	
KUB	SUPERMX	-0.677888
HSI-C5A	SP500-HG	-0.603021
HSI-C3Z	SP500-HG	-0.591305
HSI-C5B	SP500-HG	-0.579057
MUHIBAH	NYLEX	-0.568501

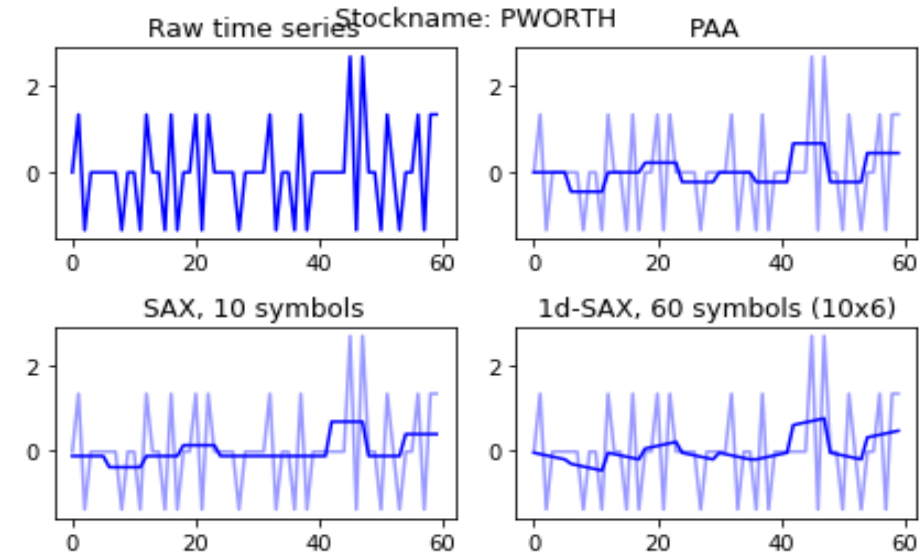
```
stocknameYONGTAI-WA
[[[-1.21540267]
  [ 1.48549215]
  [ 0.81026845]
  [ 0.13504474]
  [ 1.48549215]
  [ 1.48549215]
  [ 4.18638697]
  [-0.54017896]
  [ 0.13504474]
  [ 1.48549215]
  [-2.56585008]
  [-0.54017896]
  [-1.21540267]
  [ 0.13504474]
  [ 0.13504474]
  [-0.54017896]
  [-1.21540267]
  [-0.54017896]
  [ 0.13504474]
  [-0.54017896]
  [ 0.13504474]
```

Part 3: Normalization

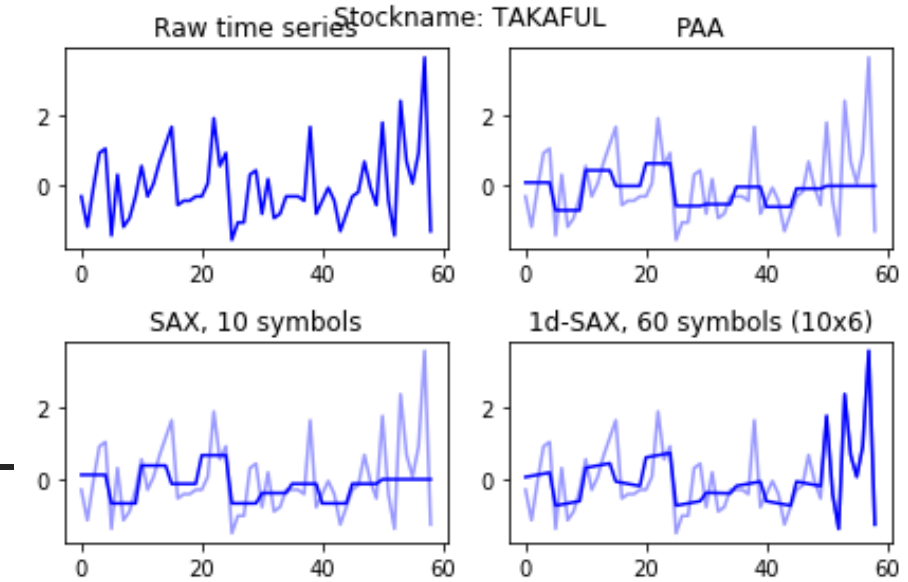
- Normalize stock data for **487 stocks that has at least 60 data points**
- Focus on **changes in daily closing prices**
- **Use tslearn package**
- **scaler = TimeSeriesScalerMeanVariance(mu=0., std=1.)**

Part 4: PAA, SAX, 1D-SAX

stocknamePWORTH



stocknameTAKAFUL



For >=60 data points	
PAA	• 10 levels
SAX	• 10 levels
1D-SAX	• 10 levels for average value • 6 levels for slope values

```
177 # ================================================================
178 df_new = pd.read_csv('stocks60.csv',usecols=[i for i in range(8) if i != 0]);print(df_new)
179 #No. of companies with >60 records
180 listnew = df_new["name"].unique().tolist()
181 len(listnew)
182 df_red = df_new.set_index(['name','day']).dif.dropna()
183 print(df_red)
184
185 scaler = TimeSeriesScalerMeanVariance(mu=0., std=1.)  # Rescale time series
186 n_paa_segments = 10
187 n_sax_symbols = 10
188 n_sax_symbols_avg = 10
189 n_sax_symbols_slope = 6
190 for i in listnew:
191     records = len(df_red[[i]])
192     print("stockname"+str(i))
193     scaleddata = scaler.fit_transform(df_red[[i]])
194     #print(scaleddata)
195     paa = PiecewiseAggregateApproximation(n_segments=n_paa_segments)
196     paa_dataset_inv = paa.inverse_transform(paa.fit_transform(scaleddata))
197     # SAX transform
198     sax = SymbolicAggregateApproximation(n_segments=n_paa_segments, alphabet_size_avg=n_sax_symbols)
199     sax_dataset_inv = sax.inverse_transform(sax.fit_transform(scaleddata))
200     # 1d-SAX transform
201     one_d_sax = OneD_SymbolicAggregateApproximation(n_segments=n_paa_segments, alphabet_size_avg=n_sax_symbols_a
202                                                     alphabet_size_slope=n_sax_symbols_slope)
203     one_d_sax_dataset_inv = one_d_sax.inverse_transform(one_d_sax.fit_transform(scaleddata))
204     plt.figure()
205     # First, raw time series
206     plt.subplot(2, 2, 1)
207     plt.plot(scaleddata[0].ravel(), "b-")
208     plt.title("Raw time series")
209     # Second, PAA
210     plt.subplot(2, 2, 2)
211     plt.plot(scaleddata[0].ravel(), "b-", alpha=0.4)
212     plt.plot(paa_dataset_inv[0].ravel(), "b-")
213     plt.title("PAA")
214     #SAX plot
215     plt.subplot(2, 2, 3)  # Then SAX
216     plt.plot(scaleddata[0].ravel(), "b-", alpha=0.4)
217     plt.plot(sax_dataset_inv[0].ravel(), "b-")
218     plt.title("SAX, %d symbols" % n_sax_symbols)
```

THE END