```
!pip install talib-binary
!pip install yfinance

import talib as ta
from pandas_datareader import data as pdr
import yfinance as yf

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import SpectralClustering
from sklearn.preprocessing import StandardScaler, normalize
from sklearn.decomposition import PCA
from sklearn.metrics import silhouette_score

import warnings
warnings.filterwarnings('ignore')
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/pub
Requirement already satisfied: talib-binary in /usr/local/lib/python3.7/dist-packages (0
Requirement already satisfied: numpy in /usr/local/lib/python3.7/dist-packages (from ta
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/pub
Requirement already satisfied: yfinance in /usr/local/lib/python3.7/dist-packages (0.1.7
Requirement already satisfied: numpy>=1.15 in /usr/local/lib/python3.7/dist-packages (fr
Requirement already satisfied: pandas>=0.24.0 in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: multitasking>=0.0.7 in /usr/local/lib/python3.7/dist-pack
Requirement already satisfied: requests>=2.26 in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: lxml>=4.5.1 in /usr/local/lib/python3.7/dist-packages (fr
Requirement already satisfied: pytz>=2017.3 in /usr/local/lib/python3.7/dist-packages (f
Requirement already satisfied: python-dateutil>=2.7.3 in /usr/local/lib/python3.7/dist-p
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-packages (from
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.7/dist-packages (f
Requirement already satisfied: charset-normalizer<3,>=2 in /usr/local/lib/python3.7/dist
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /usr/local/lib/python3.7/dist-pa
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/dist-packa
```

```
yf.pdr_override()
threshold = 0.75

df = pdr.get_data_yahoo('GOOG', '2000-01-01', '2018-01-01')
df = df[df.columns[df.isnull().mean() < threshold]]
df = df.loc[df.isnull().mean(axis=1) < threshold]
df = df.dropna()
df = df.iloc[:,:4]
df.head()
```

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

Date	Open	High	Low	Close
2004-08-19	2.490664	2.591785	2.390042	2.499133
2004-08-20	2.515820	2.716817	2.503118	2.697639
2004-08-23	2.758411	2.826406	2.716070	2.724787
2004-08-24	2.770615	2.779581	2.579581	2.611960

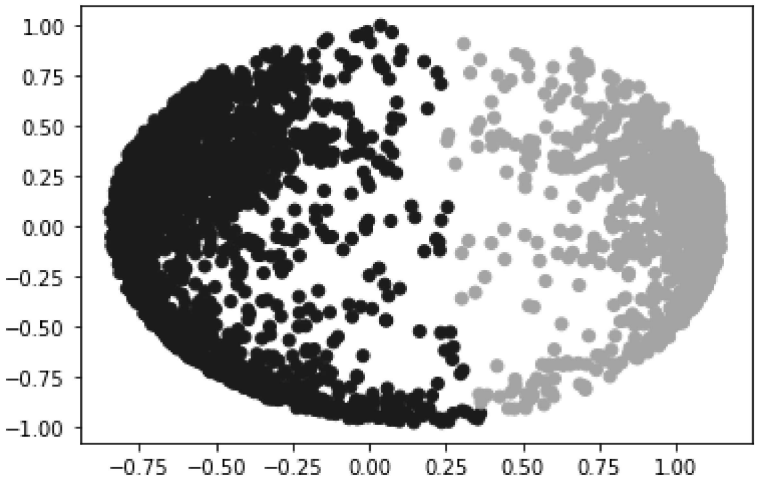
```
upper_lim = df['Close'].quantile(.95)
lower_lim = df['Close'].quantile(.05)
df = df[(df['Close'] < upper_lim) & (df['Close'] > lower_lim)]

upper_lim = df['Open'].quantile(.95)
lower_lim = df['Open'].quantile(.05)
df = df[(df['Open'] < upper_lim) & (df['Open'] > lower_lim)]


df['S_10'] = df['Close'].rolling(window=10).mean()
df['Corr'] = df['Close'].rolling(window=10).corr(df['S_10'])
df['RSI'] = ta.RSI(np.array(df['Close']), timeperiod =10)
df['Open-Close'] = df['Open'] - df['Close'].shift(1)
df['Open-Open'] = df['Open'] - df['Open'].shift(1)
df = df.dropna()
```

Date	Open	High	Low	Close	S_10	Corr	RSI	Open-Close
2005-11-21	9.941985	10.211226	9.800515	10.195784	9.841835	0.864168	76.423164	-0.025903

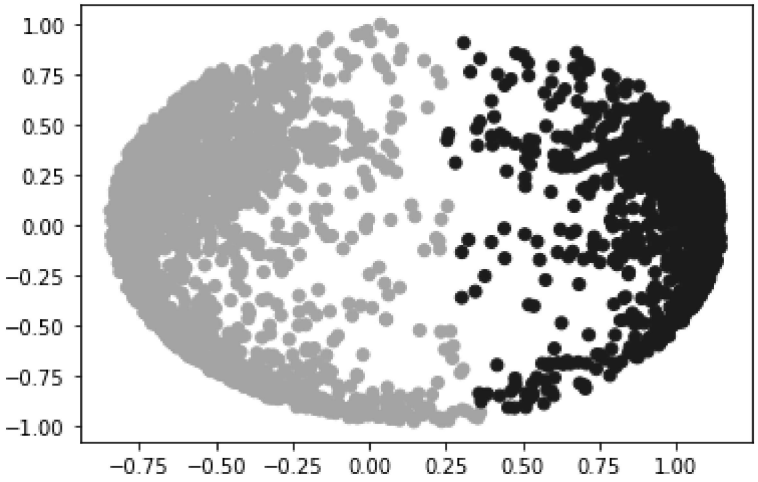
```
scaler = StandardScaler()
X_scaled = scaler.fit_transform(df)

X_normalized = normalize(X_scaled)

X_normalized = pd.DataFrame(X_normalized)

pca = PCA(n_components = 2)
X_principal = pca.fit_transform(X_normalized)
X_principal = pd.DataFrame(X_principal)
X_principal.columns = ['Open-Close', 'Open-Open']

X_principal.head(2)
```

	Open-Close	Open-Open
0	-0.607197	-0.554844

```
spectral_model_rbf = SpectralClustering(n_clusters = 2, affinity ='rbf')
labels_rbf = spectral_model_rbf.fit_predict(X_principal)


plt.scatter(X_principal['Open-Close'], X_principal['Open-Open'],
           c = SpectralClustering(n_clusters = 2, affinity ='nearest_neighbors') .fit_predict
plt.show()
```




```
spectral_model_nn = SpectralClustering(n_clusters = 2, affinity ='nearest_neighbors')
labels_nn = spectral_model_nn.fit_predict(X_principal)


plt.scatter(X_principal['Open-Close'], X_principal['Open-Open'],
           c = SpectralClustering(n_clusters = 2, affinity ='nearest_neighbors') .fit_predict
plt.show()
```




```
affinity = ['rbf', 'nearest-neighbours']

s scores = []
```

```
s_scores.append(silhouette_score(df, labels_rbf))
s_scores.append(silhouette_score(df, labels_nn))

plt.bar(affinity, s_scores)
plt.xlabel('Affinity')
plt.ylabel('Silhouette Score')
plt.title('Comparison of different Clustering Models')
plt.show()

print(s_scores)
```


```
[0.4735833256800287, 0.4714920772028157]
```

✓ 0s completed at 3:02 PM