```
!pip3 install kaggle
!mkdir ~/.kaggle/
! cp kaggle.json ~/.kaggle/
```

```
Requirement already satisfied: kaggle in /usr/local/lib/python3.7/dist-packages (1.5
Requirement already satisfied: python-slugify in /usr/local/lib/python3.7/dist-packa
Requirement already satisfied: six>=1.10 in /usr/local/lib/python3.7/dist-packages (
Requirement already satisfied: certifi in /usr/local/lib/python3.7/dist-packages (fr
Requirement already satisfied: urllib3 in /usr/local/lib/python3.7/dist-packages (fr
Requirement already satisfied: requests in /usr/local/lib/python3.7/dist-packages (f
Requirement already satisfied: tqdm in /usr/local/lib/python3.7/dist-packages (from
Requirement already satisfied: python-dateutil in /usr/local/lib/python3.7/dist-pack
Requirement already satisfied: text-unidecode>=1.3 in /usr/local/lib/python3.7/dist-
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-package
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.7/dist-pa
mkdir: cannot create directory ‘/root/.kaggle/’: File exists
```

```
! chmod 600 ~/.kaggle/kaggle.json
```

```
!kaggle datasets download -d vjchoudhary7/customer-segmentation-tutorial-in-python
```

```
customer-segmentation-tutorial-in-python.zip: Skipping, found more recently modified
```

```
!unzip customer-segmentation-tutorial-in-python.zip
```

```
Archive:  customer-segmentation-tutorial-in-python.zip
replace Mall_Customers.csv? [y]es, [n]o, [A]ll, [N]one, [r]ename: a
error:  invalid response [a]
replace Mall_Customers.csv? [y]es, [n]o, [A]ll, [N]one, [r]ename: A
  inflating: Mall_Customers.csv
```

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

```
data = pd.read_csv('./Mall_Customers.csv')
```

```
data.head()
```

	CustomerID	Gender	Age	Annual Income (k$)	Spending Score (1-100)
0	1	Male	19	15	39
1	2	Male	21	15	81

```
data=data.drop(columns=['Gender'])
bar=data
```

Applying Scaler and PAC to normalize our data to apply algorithm.

```
scaler = StandardScaler()
X_scaled = scaler.fit_transform(data)
X_normalized = normalize(X_scaled)

# Converting the numpy array into a pandas DataFrame
X_normalized = pd.DataFrame(X_normalized)

# Reducing the dimensions of the data
pca = PCA(n_components = 2)
X_principal = pca.fit_transform(X_normalized)
data = pd.DataFrame(X_principal)
data.columns = ['P1', 'P2']

data.head()
```

	P1	P2
0	-0.616450	-0.688409
1	-0.505240	-0.831002
2	-0.604943	-0.427461
3	-0.545842	-0.807508
4	-0.808550	-0.504300

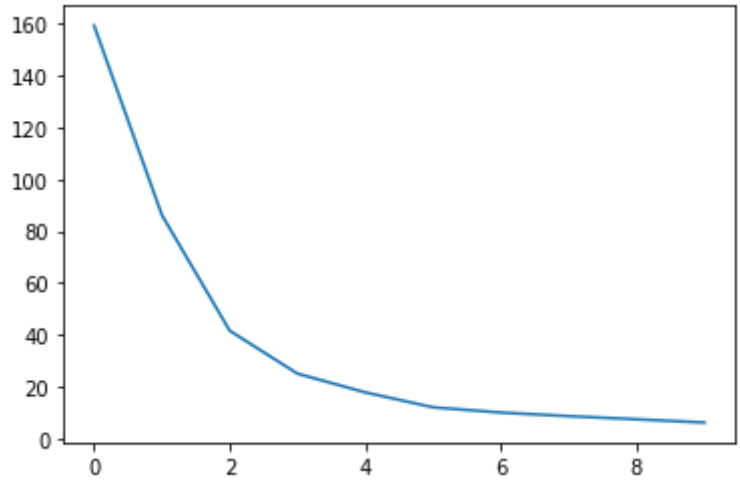
Applying k-means ALgorithm.

```
'''Age and spending Score'''
X1 = data
inertia = []
for n in range(1 , 11):
    algorithm = (KMeans(n_clusters = n ,init='k-means++', n_init = 10 ,max_iter=300,
                        tol=0.0001,  random_state= 111  , algorithm='elkan') )
    algorithm.fit(X1)
    inertia.append(algorithm.inertia_)
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/cluster/_kmeans.py:968: RuntimeWarnin
  RuntimeWarning,
```

```
!pip3 install mplcursors
import mplcursors
plt.plot(inertia)
mplcursors.cursor(hover=True)
```

```
Requirement already satisfied: mplcursors in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: matplotlib>=3.1 in /usr/local/lib/python3.7/dist-pack
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in /usr/loca
Requirement already satisfied: numpy>=1.11 in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.7/dist-package
Requirement already satisfied: python-dateutil>=2.1 in /usr/local/lib/python3.7/dist
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.7/dist-pa
Requirement already satisfied: typing-extensions in /usr/local/lib/python3.7/dist-pa
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-packages (f
<mplcursors._mplcursors.Cursor at 0x7fedb8b44a90>
```




```
inertia
```

```
[159.30601324222272,
 86.2217721866943,
 41.62362392559551,
 25.051857159233393,
 17.82251059542758,
 12.067410250145638,
 10.050024265127114,
 8.668470605190478,
 7.464136566411021,
 6.202093468097921]
```

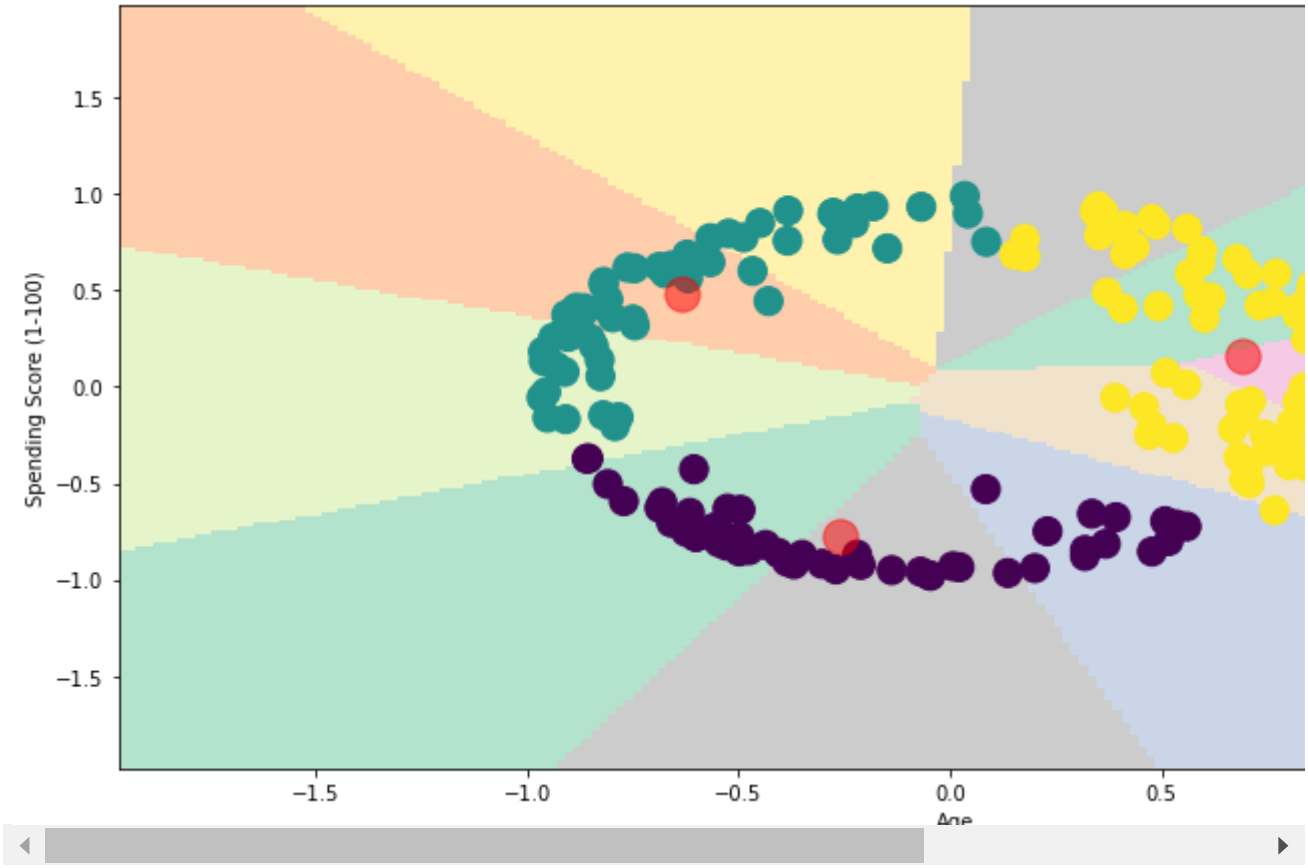
```
algorithm_kmeans= (KMeans(n_clusters = 3 ,init='k-means++', n_init = 10 ,max_iter=300,
                        tol=0.0001,  random_state= 111  , algorithm='elkan') )
algorithm_kmeans.fit(X1)
labels1 = algorithm_kmeans.labels_
centroids1 = algorithm_kmeans.cluster_centers_
```

```
smallData = pd.DataFrame(np.c_[xx.ravel(), yy.ravel()], columns = ['P1', 'P2'])
```

```
h = 0.02
x_min, x_max = X1['P1'].min() - 1, X1['P1'].max() + 1
y_min, y_max = X1['P2'].min() - 1, X1['P2'].max() + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max, h))
Z = algorithm.predict(smallData)

plt.figure(1 , figsize = (15 , 7) )
plt.clf()
Z = Z.reshape(xx.shape)
plt.imshow(Z , interpolation='nearest',
           extent=(xx.min(), xx.max(), yy.min(), yy.max()),
           cmap = plt.cm.Pastel2, aspect = 'auto', origin='lower')

plt.scatter( x = 'P1' ,y = 'P2' , data = data , c = labels1 ,
            s = 200 )
plt.scatter(x = centroids1[: , 0] , y =  centroids1[: , 1] , s = 300 , c = 'red' , alpha =
plt.ylabel('Spending Score (1-100)') , plt.xlabel('Age')
plt.show()
```



Applying Spectral clutering with rbf affinity.

```
from sklearn.cluster import SpectralClustering
sc=SpectralClustering(n_clusters=3)
sc=sc.fit_predict(X1)
```

```
sc
```

```
array([1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1,
       0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1,
       0, 1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1,
       0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1,
       1, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0,
       0, 1, 2, 2, 2, 2, 0, 0, 0, 0, 2, 2, 2, 2, 2, 2, 2, 2, 0, 2, 0, 2,
       2, 2, 2, 2, 0, 2, 2, 2, 0, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2], dtype=int32)
```

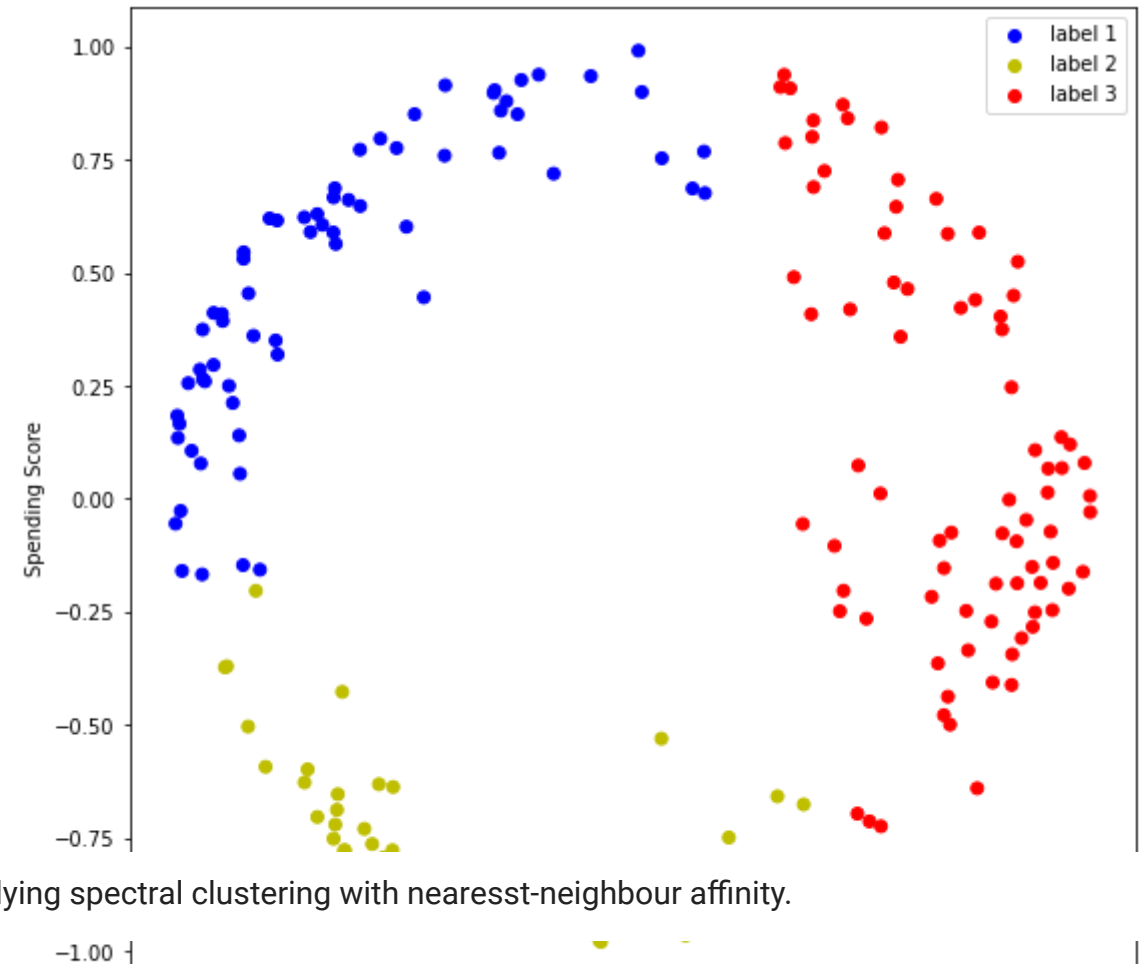
X1

	P1	P2
0	-0.616450	-0.688409
1	-0.505240	-0.831002
2	-0.604943	-0.427461
3	-0.545842	-0.807508
4	-0.808550	-0.504300
...	...	...
195	0.949111	0.135969
196	0.771409	0.588641
197	0.968015	0.119877
198	0.821200	0.374288
199	0.950343	0.067703

200 rows × 2 columns

```
colors={}
colors[0]='b'
colors[1]='y'
colors[2]='r'
cvec=[colors[i] for i in sc]
```

```
plt.figure(figsize =(9,9))
plt.scatter(x='P1',y='P2',data=X1,c=cvec)
plt.xlabel('Age'),plt.ylabel('Spending Score')
plt.legend((b,y,r),('label 1','label 2','label 3'))
plt.show()
```

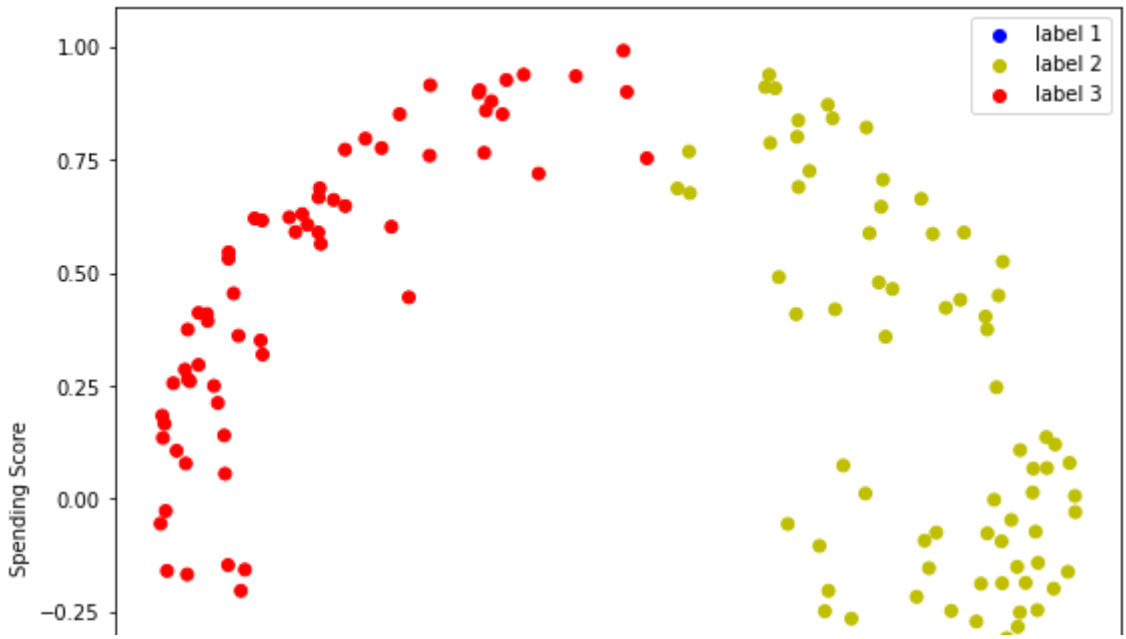


Applying spectral clustering with nearesst-neighbour affinity.

```
nearest=SpectralClustering(n_clusters=3,affinity='nearest_neighbors')
nearest_model=nearest.fit_predict(X1)


colors={}
colors[0]='b'
colors[1]='y'
colors[2]='r'
cvec=[colors[i] for i in nearest_model]


plt.figure(figsize =(9,9))
plt.scatter(x='P1',y='P2',data=X1,c=cvec)
plt.xlabel('Age'),plt.ylabel('Spending Score')
plt.legend((b,y,r),('label 1','label 2','label 3'))
plt.show()
```



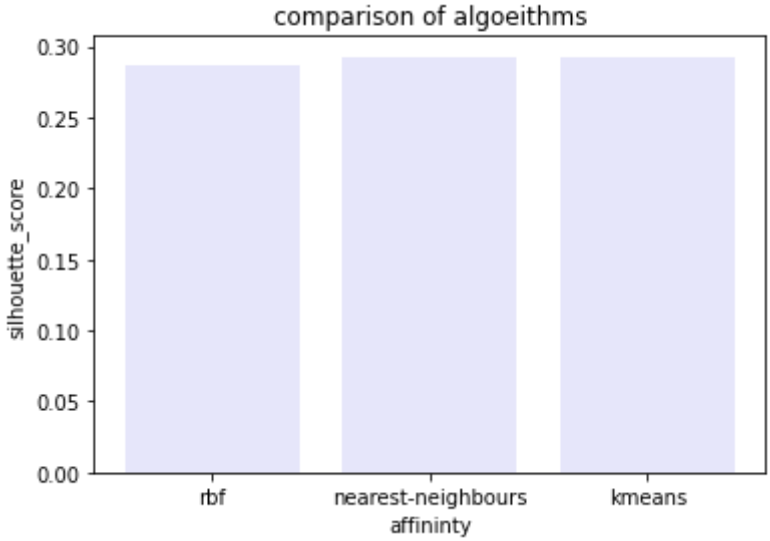
Comparing the three algorithms.

```
affininty=['rbf','nearest-neighbours','kmeans']
s_scores=[]
s_scores.append(silhouette_score(bar,sc))
s_scores.append(silhouette_score(bar,nearest_model))
s_scores.append(silhouette_score(bar,labels1))
```

```
print(s_scores)
```

```
[0.2864728854110913, 0.29270719573982384, 0.29270719573982384]
```

```
plt.bar(affininty,s_scores,color='lavender')
plt.xlabel('affininty'),plt.ylabel('silhouette_score')
plt.title("comparison of algoeithms")
plt.show()
```



Since all the algorithms are in close proximity from each other and k-means and specteral clusterin with affinity towards nearest neighbour are identical, we preffred k means, when training an algorith with preset data base.

✓ 0s completed at 11:54 PM