Studi kasus:
segmentasi pelanggan berdasarkan 3 atribut utama yaitu ***income*** (dalam satuan puluhan ribu dollar), ***average purchases*** (dalam satuan ratusan dollar) dan ***last year purchases*** (dalam satuan unit).

1. Import libraries

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
```

2. Import file
3. Membaca file
4. Menentukan tipe data atribut
5. Menentukan jumlah missing data
6. Identifikasi data outlier dengan boxplot

```
ax=sns.boxplot(data=df,orient = "h",palette="Set2")
```

*hasil tdk ada data outlier

7. Data preprocessing : dilakukan normalisasi/standarisasi z-score dengan menggunakan fungsi StandardScaler kemudian hasilnya disimpan sebagai sebuah dataframe dengan nama “std_atr” (standardized attibutes)

```
from sklearn.preprocessing import StandardScaler
scale=StandardScaler()
std_atr=scale.fit_transform(df)
std_atr=pd.DataFrame(std_atr,columns=df.columns)
print(std_atr)
```

8. Model tuning

Pada algoritma k-means clustering terdapat nilai parameter k, sehingga sebelum diputuskan subjektif nilai k, dapat dilakukan model tuning, misalnya dengan metode Elbow.

Nilai inertia akan semakin kecil dengan bertambahnya jumlah cluster dengan nilai k optimal diperoleh dari titik balik, yang mana penurunan inertia sudah relatif “kecil”, dicirikan dengan kurva mulai landai.
Input fungsi KMeans:

```
from sklearn.cluster import KMeans
wcss = []
for i in range(1,11):
  km=KMeans(n_clusters=i)
```

```
  km.fit(std_atr)
  wcss.append(km.inertia_)
plt.figure(figsize=(12,6))
plt.plot(range(1,11),wcss)
plt.plot(range(1,11),wcss,linewidth=2,color="red",marker="8")
plt.xlabel("K Value")
plt.xticks(np.arange(1,11,1))
plt.ylabel("Within-cluster Some of Square")
plt.show()
```

*pada Kvalue ketiga mulai terjadi pengecilan penurunan, kemungkinan pada nilai tersebut merupakan KValue yg optimal

Untuk lebih meyakinkan berapa nilai k optimal, dilakukan analisa silhouette coefficient berdasarkan konfigurasi setiap anggota clusternya. Visualisasi dilakukan dengan menggunakan fungsi SilhouetteVisualizer.

```
from yellowbrick.cluster import SilhouetteVisualizer
model=KMeans(3,random_state=42)
visualizer=SilhouetteVisualizer(model,colors='yellowbrick')
visualizer.fit(std_atr)
```

*Hasil : semua objek data memiliki skor silhouette positif, hal ini mengindikasikan bahwa objek-objek sudah dikelompokkan pada konfigurasi yang sesuai.

Menampilkan skor silhouette coefficient pada k=3, dan nilai-nilai k yang lain

```
from sklearn.metrics import silhouette_score
km=KMeans(n_clusters=3,random_state=42)
km.fit(std_art)
score=silhouette_score(std_atr,km.labels_)
print('SilhouetterScore : %.3f' % score)
```

```
from sklearn.metrics import silhouette_score
k_cluster = []
sil_coeffecients = []

for n_cluster in range(2,15):
```

```
  kmeans = KMeans(n_clusters=n_cluster).fit(std_atr)
  label = kmeans.labels_
  sil_coeff = silhouette_score(std_atr, label)
  print("For n_clusters={}, Silhouette coeffecient = {}".format(n_cluster
, sil_coeff))
  sil_coeffecients.append(sil_coeff)
  k_clusters.append(n_cluster)


plt.plot(k_cluster,sil_coeffecients)
plt.ylabel('Silhouette Coeffecient')
plt.xlabel('No. of Clusters')
plt.show()
```

9. Model building

Setelah nilai k terpilih (k=3), selanjutnya dilakukan pembangunan model 3-means clustering dengan menggunakan fungsi Kmeans dan input atribut-atribut yang sudah dinormalisasi. Hasil dari clustering disajikan dalam sebuah dataframe dengan nama “hasil”.

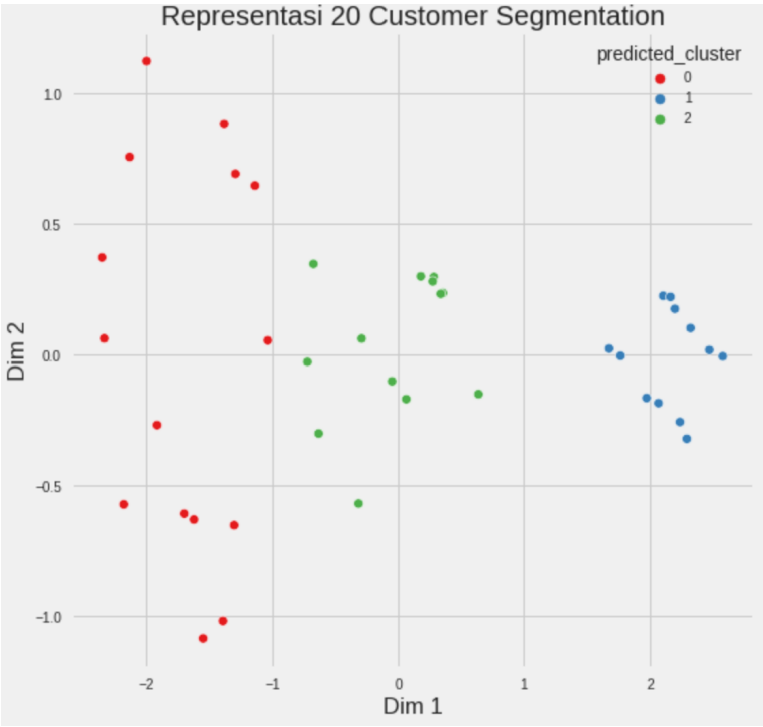
```
model=KMeans(init="random",n_clusters=3)
model.fit(std_atr)
print('inertia:',model.inertia_)
print('cluster_centroids:',model.cluster_centers_)

clt=model.labels_
hasil=pd.Series(clt,name="Cluster")
hasil=pd.DataFrame(hasil)
```

10. Model visualization

Hasil dari clustering selanjutnya dapat dilakukan visualisasi kedalam plot 2 dimensi. Dengan jumlah atribut sebanyak 3, untuk membuat plot 2 dimensi dapat dilakukan teknik reduksi data dengan PCA.

```
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import LabelEncoder, StandardScaler
```

```
from sklearn.decomposition import PCA

preprocessor=Pipeline([("scaler",StandardScaler()),("pca",PCA(n_components
=2,random_state=42)),])

clusterer=Pipeline([("kmenas",KMeans(n_clusters=3,init="k-means++",n_init=
50,max_iter=500,random_state=42,),),])

pipe=Pipeline([("preprocessor",preprocessor),("clusterer",clusterer)])

pipe.fit(df)


preprocessor_data=pipe["preprocessor"].transform(df)

predicted_labels=pipe["clusterer"]["kmeans"].labels_


pcadf=pd.DataFrame(pipe["preprocessor"].transform(df),columns=["Dim
1","Dim 2"],)

pcadf["predicted_cluster"]=pipe["clusterer"]["kmeans"].labels_


plt.style.use("fivethirtyeight")

plt.figure(figsize=(8,8))

scat=sns.scatterplot("Dim                    1","Dim                    2",
s=50,data=pcadf,hue="predicted_cluster", palette="Set1",)

scat.set_title("Representasi 20 Customer Segmentation")

plt.show()
```



11. Result

Dari data std_atr dan centroid, didapatkan pembagian klusterisasi utk masing2 data:

Cluster	Objects
0	0, 3, 6, 9, 10, 11, 14, 18, 19, 25, 26, 29, 31, 32, 38
1	1, 4, 5, 7, 8, 16, 17, 21, 23, 27, 30, 35
2	2, 12, 13, 15, 20, 22, 24, 28, 33, 34, 36, 37, 39

Cluster mana yg terbaik dilihat dari hasil point 9. Centroid-nya

```
inertia: 19.812698382869723
cluster_centroids: [[ 0.21435809 -0.18652161  0.04739934]
 [ 0.90384157  1.05915376  0.9607285 ]
 [-1.36202323 -1.12187713 -1.25225991]]
```

Cluster	Centroids		
	Income	Average_purchases	Last_year_purchases
0	0.904	1.059	0.961
1	-1.362	-1.122	-1.252
2	0.214	-0.186	0.047

Kesimpulan: cluster 0 terbaik