Laporan Tugas Besar Mandiri
Pembelajaran Mesin

Oleh: Alif Ranadian Nadhifah – 1301184255

Formulasi Masalah

Diberikan sebuah dataset yang menyimpan informasi pelanggan mengenai kendaraan untuk dibangun sebuah model clustering untuk memprediksi apakah pelanggan ingin membeli kendaraan baru atau tidak

Eksplorasi dan Persiapan Data

Sebelum dataset diolah, langkah pertama yang harus dilakukan adalah kenali dahulu datasetnya. Pada gambar berikut menunjukkan code untuk mengetahui info terkait dataset.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 285831 entries, 0 to 285830
Data columns (total 12 columns):
 #   Column              Non-Null Count   Dtype
---  ------              --------------   -----
 0   id                  285831 non-null  int64
 1   Jenis_Kelamin       271391 non-null  object
 2   Umur                271617 non-null  float64
 3   SIM                 271427 non-null  float64
 4   Kode_Daerah         271525 non-null  float64
 5   Sudah_Asuransi      271602 non-null  float64
 6   Umur_Kendaraan      271556 non-null  object
 7   Kendaraan_Rusak     271643 non-null  object
 8   Premi               271262 non-null  float64
 9   Kanal_Penjualan     271532 non-null  float64
 10  Lama_Berlangganan   271839 non-null  float64
 11  Tertarik            285831 non-null  int64
dtypes: float64(7), int64(2), object(3)
memory usage: 26.2+ MB
```

Terlihat bahwa dataset kendaraan memiliki kolom berjumlah 12, dengan nama masing-masing kolomnya diantaranya adalah: ‘id, ‘Jenis_Kelamin’, ‘Umur’, ‘SIM’, ‘Kode_Daerah’, ‘Sudah_Asuransi’, ‘Umur_Kendaraan’, ‘Kendaraan_Rusak’, ‘Premi’, ‘Kanal_Penjualan’, ‘Lama_Berlangganan’, dan 1 label yaitu: ‘Tertarik’. Namun, dari 12 kolom tersebut, ada beberapa kolom yang tidak akan digunakan dalam pembangunan model nantinya. Hal ini dikarenakan tidak mempunyai efek terhadap pembangunan model dan akan berakibat pada keberhasilan model yang dibangun nantinya.

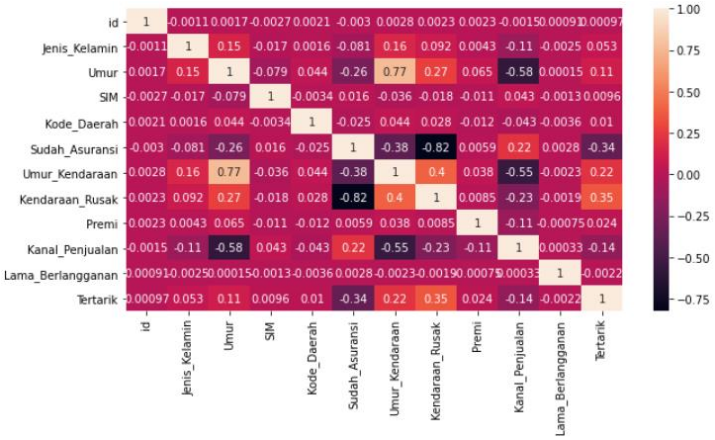
Untuk mengetahui kolom apa saja yang akan ataupun tidak digunakan dalam pembangunan model, maka dilakukan pemvisualisasian data menggunakan fungsi heatmap. Namun sebelum memvisualisasikan data ke dalam bentuk heatmap, perlu dilakukan encode data bertipe object menjadi data yang terdiri dari data bertipe integer. Hal ini dilakukan karena heatmpa hanya menampilkan data yang bertipe integer ataupun float. Sedangkan dataset masih memiliki 3 kolom yang bertipe object, yaitu: ‘Jenis_Kelamin’, ‘Umur_Kendaraan’, dan ‘Kendaraan_Rusak’. Berikut ini merupakan code dan hasil dari encode data yang bertipe object:

```
encode_dfk = {"Jenis_Kelamin": {"Wanita": 0, "Pria": 1},
              "Umur_Kendaraan": {"< 1 Tahun":0 , "1-2 Tahun":1, "> 2 Tahun":2},
              "Kendaraan_Rusak": {"Tidak":0, "Pernah":1}}
df_kendaraan.replace(encode_dfk, inplace=True)
```

```
df_kendaraan.head()
```

	id	Jenis_Kelamin	Umur	SIM	Kode_Daerah	Sudah_Asuransi	Umur_Kendaraan	Kendaraan_Rusak	Premi	Kanal_Penjualan	Lama_Berlangganan	Tertarik
0	1	0.0	30.0	1.0	33.0	1.0	0.0	0.0	28029.0	152.0	97.0	0
1	2	1.0	48.0	1.0	39.0	0.0	2.0	1.0	25800.0	29.0	158.0	0
2	3	NaN	21.0	1.0	46.0	1.0	0.0	0.0	32733.0	160.0	119.0	0
3	4	0.0	58.0	1.0	48.0	0.0	1.0	0.0	2630.0	124.0	63.0	0
4	5	1.0	50.0	1.0	35.0	0.0	2.0	NaN	34857.0	88.0	194.0	0

Fungsi dari memvisualisasikan heatmap adalah tak lain untuk memperlihatkan seberapa erat hubungan antar varibel kolom ataupun antara variabel kolom dengan label.



Pada gambar diatas terlihat bahwa, semakin gelap warna angka menunjukkan mendekati 0 yang berartikan hubungan antar kolom terlalu jauh. Namun sebaliknya, bila warna semakin terang, angka menunjukkan mendekati angka 1 yang berartikan hubungan antar kolom hampir saling berkaitan. Dari gambar tersebut, terlihat bahwa 3 variabel kolom yang memiliki keterkaitan erat dengan label data dan memiliki nilai tertinggi adalah kolom ‘Kendaraan_Rusak’, ‘Umur_Kendaraan’, dan ‘Umur’. Sehingga selain ketiga kolom tersebut, kolom akan didrop dan hasilnya sebagai berikut:

```
cols = ['Umur','Umur_Kendaraan', 'Kendaraan_Rusak']
df_kendaraan = df_kendaraan[cols]

df_kendaraan.head()
```

	Umur	Umur_Kendaraan	Kendaraan_Rusak
0	30.0	0.0	0.0
1	48.0	2.0	1.0
2	21.0	0.0	0.0
3	58.0	1.0	0.0
4	50.0	2.0	NaN

Setelah data diseleksi, perlu diketahui bahwa dataset masih memiliki nilai null / data kosong. Pembangunan model akan berbengaruh hasilnya jika dataset memiliki nilai yang

kosong. Maka langkah yang tepat untuk menangi hal ini adalah mengisi data kosong dengan nilai modus pada masing-masing kolom seperti pada gambar dibawah ini:

```
df_kendaraan["Umur"] = df_kendaraan["Umur"].fillna(df_kendaraan["Umur"].mode()[0])
df_kendaraan["Umur_Kendaraan"] = df_kendaraan["Umur_Kendaraan"].fillna(df_kendaraan["Umur_Kendaraan"].mode()[0])
df_kendaraan["Kendaraan_Rusak"] = df_kendaraan["Kendaraan_Rusak"].fillna(df_kendaraan["Kendaraan_Rusak"].mode()[0])

# Cek kembali apakah masih terdapat data null/tidak
df_kendaraan.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 285831 entries, 0 to 285830
Data columns (total 3 columns):
 #   Column           Non-Null Count   Dtype  
---  ------           --------------   -----  
 0   Umur             285831 non-null  float64
 1   Umur_Kendaraan   285831 non-null  float64
 2   Kendaraan_Rusak  285831 non-null  float64
dtypes: float64(3)
memory usage: 6.5 MB
```

Pemodelan

Pada pemodelan pertama ini, model yang akan dipakai adalah model Unsupervised Learning yakni Clustering. Hal pertama yang harus dilakukan adalah membagi dataset train set menjadi 2 bagian. Hal ini diperlukan untuk meminimalisir kesalahan atau eror yang akan terjadi jika dilakukan pada model lain atau pada final data test set. Contohnya seperti berikut:

```
# Ambil score 2 data yang memiliki keterkaitan erat tertinggi untuk dimodelkan
X = df_kendaraan.iloc[:, [1, 2]].values
```

```
m=X.shape[0] #Train Set
n=X.shape[1] #Validation set
n_iter=5 #Jumlah Iterasi
K=2 #Jumlah Kluster
```

Setelah itu, inisialisasi variabel yang diperlukan dalam pembangunan model. Langkah selanjutnya adalah buat nilai centroid untuk menentukan titik centroid dari masing-masing klaster data nantinya.

```
import numpy as np
Centroids=np.array([]).reshape(n,0)
```

```
import random
for i in range(K):
    rand = random.randint(0,m-1)
    Centroids = np.c_[Centroids,X[rand]]
```

```
# Inisialisasi Hasil keluaran ke dalam bentuk dictionary
Output={}
```

Langkah penting dalam klasterisasi ini adalah perhitungan jarak euclid. Euclide merupakan jarak antara 2 titik. Euclid diperlukan untuk menghitung jaraka antar titik ataupun dengan centroid untuk menentukan letak masing-masing titik data dan centroid.

```
euclid=np.array([]).reshape(m,0)
for k in range(K):
    tempDist = np.sum((X-Centroids[:,k])**2,axis=1)
    euclid = np.c_[euclid,tempDist]
#Mencari jarak minimum dari setiap baris matriks dan menyimpan indeks kolom dalam vektor Z
Z = np.argmin(euclid,axis=1)+1
```

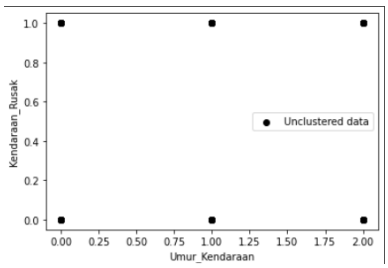
Setelah titik centroid dan jarak euclid ditentukan barulah model cluster dibangun untuk menentukan titik-titik data serta centroid masing-masing cluster dalam 1 frame data yang sama.

```
Y={} # Kamus sementara untuk menyimpan solusi dari 1 iterasi
for k in range(K):
    Y[k+1]=np.array([]).reshape(2,0)
for M in range(m):
    Y[Z[M]]=np.c_[Y[Z[M]],X[M]]

for k in range(K):
    Y[k+1]=Y[k+1].T

for k in range(K):
     Centroids[:,k]=np.mean(Y[k+1],axis=0)
```

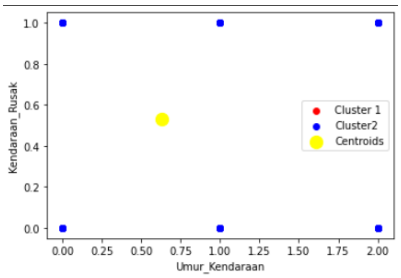
```
for i in range(n_iter):
    euclid=np.array([]).reshape(m,0)
    for k in range(K):
        tempDist = np.sum((X-Centroids[:,k])**2,axis=1)
        euclid = np.c_[euclid,tempDist]
        Z=np.argmin(euclid,axis=1)+1
    Y={}
    for k in range(K):
        Y[k+1]=np.array([]).reshape(2,0)
    for M in range(m):
        Y[Z[M]]=np.c_[Y[Z[M]],X[M]]

    for k in range(K):
        Y[k+1]=Y[k+1].T

    for k in range(K):
        Centroids[:,k]=np.mean(Y[k+1],axis=0)

    Output=Y
```

Eksperimen

Banyaknya jumlah centroid yang dibangun adalah K=2, hal ini dikarenakan untuk penyesuaian terhadap nilai yang akan dicari yaitu ‘Tertarik’ yang mempunyai dua nilai (‘Ya’ dan ‘Tidak’). Kedua centroid tersebut mempunyai koordinat awal acak, kemudian koordinat kedua centroid tersebut akan diperbarui berdasarkan rumus euclidean terhadap koordinat data 'Umur_Kendaraan' dan 'Kendaraan_Rusak' terdekat. Berikut merupakan perbandingan gambar plot data sebelum klasterisasi dan setelah klasterisasi:



Sebelum Klasterisasi



Setelah Klasterisasi

Kesimpulan

Dengan melihat dari plot data pada gambar diatas maka didapatkan bahwa cluster terbaik adalah K=2, sesuai dengan apa yang telah diterapkan pada model yang dibangun.