PEMBELAJARAN MESIN

LAPORAN TUGAS BESAR TAHAP PERTAMA

(TASK UNSUPERVISED CLUSTERING)

Laporan

Disusun untuk Memenuhi Tugas Besar Mata Kuliah Pembelajaran Mesin

Oleh:

Muhammad Aziz Pratama (NIM 1301180018)

Kelas : IF-42-11

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS INFORMATIKA

UNIVERSITAS TELKOM

BANDUNG

2021

1. **Formulasi Masalah**

Clustering atau klasterisasi adalah metode pengelompokan data. Clustering merupakan proses partisi satu set objek data ke dalam himpunan bagian yang disebut dengan cluster. Objek yang di dalam cluster memiliki kemiripan karakteristik antar satu sama lainnya dan berbeda dengan cluster yang lain. Oleh karena itu, clustering sangat berguna dan bisa menemukan group atau kelompok yang tidak dikenal dalam data.

Task yang dilakukan adalah mengelompokkan data pelanggan tertarik untuk membeli kendaraan baru atau tidak berdasarkan data pelanggan di dealer. K-Means dipilih sebagai metode dalam task ini. Tujuan dari clustering adalah meminimumkan jarak antara data point dan centroid, serta memaksimumkan jarak antara centroid yang dihitung menggunakan within-cluster sum of squares atau WCSS.

2. **Eksplorasi dan Persiapan Data**

- Dataset dan featurenya diikuti dengan type datanya

```
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)
```

- Dapat kita lihat, bahwa terdapat nilai Nan atau nilai kosong didalam dataset. Karena Dataset memiliki sekitar 200-an ribu dan rata-rata feature hanya memiliki 5% data Nan, maka saya memilih untuk drop saja data Nan dan masih tersedia data sebanyak 171068 data.

```
Premi               5.097068
Jenis_Kelamin       5.051936
SIM                 5.039341
Kode_Daerah         5.005055
Kanal_Penjualan     5.002606
Umur_Kendaraan      4.994210
Sudah_Asuransi      4.978116
Umur                4.972869
Kendaraan_Rusak     4.963772
Lama_Berlangganan   4.895200
Tertarik            0.000000
id                  0.000000
dtype: float64
```

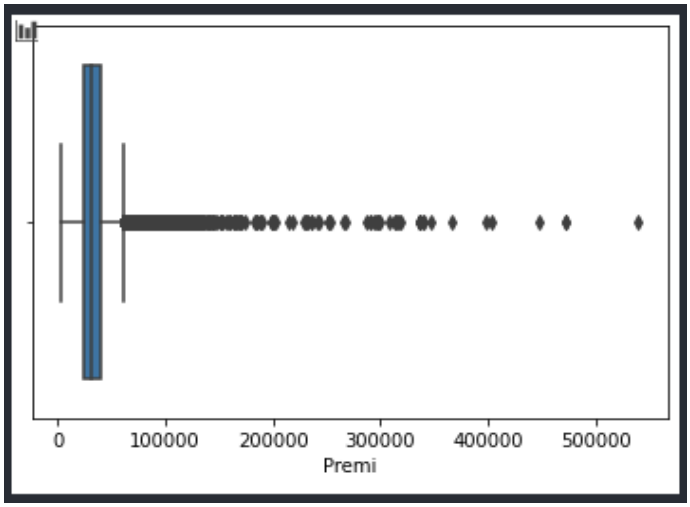
```
data_train = data_train.dropna()
len(data_train)
```

```
171068
```

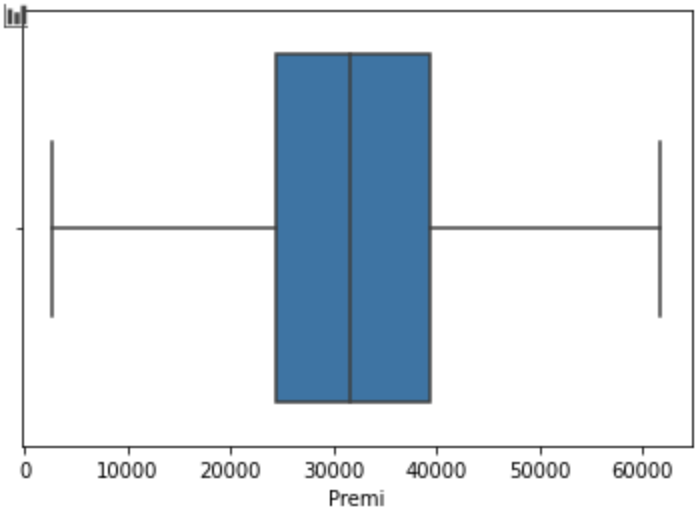
- Setelah itu dipilih lagi mana feature yang berguna dan tidak, oleh karena itu saya meghilangkan feature id(karena tidak berarti apa-apa), Tertarik(karena unsupervised tidak memakai label), dan Kanal_Penjualan.

- Untuk melihat korelasi data antar feature, kita dapat melihatnya dengan heatmap, tetapi bisa dilihat bahwa antar feature tidak terlalu berkorelasi, karena feature didataset ini sendiri banyak yang berbentuk categorical.

- Selanjutnya, melakukan handle terhadap data *outlier* / pencilan. Pada dataset, saya menemukan data outlier pada feature Premi.



Untuk menangani data outlier ini, saya mengganti data outlier tersebut dengan menggunakan nilai upperbound dan lowerbound. Karena pada data feature premi ini outlier ada diatas upperbound semua, maka saya hanya menggunakan nilai upperbound. Dengan nilai upperbound itu sendiri 61753.5. Sehingga tidak terdapat nilai outlier lagi didalam feature ini.



- Data *Categorical* juga harus dihandle, pada dataset ini terdapat beberapa feature categorical seperti Jenis_Kelamin, SIM, Kode_Daerah, Sudah_Asuransi, Umur_Kendaraan, Kendaraan_Rusak.
    - Untuk jenis data categorical nominal (Jenis_Kelamin, SIM, Kode_Daerah, Sudah_Asuransi, Kendaraan_Rusak) saya menggunakan cara one hot encoding dengan pandas.dummies.
    - Sedangkan jenis data categorical ordinal (Umur_Kendaraan) saya menggunakan cara label encoder biasa.

	Jenis_Kelamin	Umur	SIM	Kode_Daerah	Sudah_Asuransi	Umur_Kendaraan	Kendaraan_Rusak	Premi	Lama_Berlangganan
0	Wanita	30.0	1.0	33.0	1.0	< 1 Tahun	Tidak	28029.0	97.0
1	Pria	48.0	1.0	39.0	0.0	> 2 Tahun	Pernah	25800.0	158.0
3	Wanita	58.0	1.0	48.0	0.0	1-2 Tahun	Tidak	2630.0	63.0
5	Pria	21.0	1.0	35.0	1.0	< 1 Tahun	Tidak	22735.0	171.0
8	Wanita	20.0	1.0	8.0	1.0	< 1 Tahun	Tidak	30786.0	31.0

Data awal

	Umur	Umur_Kendaraan	Premi	Lama_Berlangganan	jenkel_Pria	jenkel_Wanita	daerah_0.0	daerah_1.0	daerah_2.0	daerah_3.0	daerah_4.0	daerah_5.0	daerah_6.0
0	30.0	1	28029.0	97.0	0	1	0	0	0	0	0	0	0
1	48.0	2	25800.0	158.0	1	0	0	0	0	0	0	0	0
3	58.0	0	2630.0	63.0	0	1	0	0	0	0	0	0	0
5	21.0	1	22735.0	171.0	1	0	0	0	0	0	0	0	0
8	20.0	1	30786.0	31.0	0	1	0	0	0	0	0	0	0

Setelah di label (kolom tidak terlihat semua, karena banyak)

- *Scaling* dilakukan pada feature bertype float yaitu Umur, Premi, Lama_Berlangganan. Scaling data dilakukan menggunakan MinMax Scaling yang berguna untuk mempercepat proses perhitungan pada kmeans dan membuat range nilai menjadi sama yaitu di rentang [0,1] karena kmeans ini sendiri melakukan perhitungan dengan jarak antar data. berikut data dari hasil MinMax Scaling :

	Umur	Umur_Kendaraan	Premi	Lama_Berlangganan	jenkel_Pria	jenkel_Wanita	daerah_0.0	daerah_1.0	daerah_2.0	daerah_3.0	d
0	0.153846	1	0.429592	0.301038	0	1	0	0	0	0	
1	0.430769	2	0.391892	0.512111	1	0	0	0	0	0	
3	0.584615	0	0.000000	0.183391	0	1	0	0	0	0	
5	0.015385	1	0.340051	0.557093	1	0	0	0	0	0	
8	0.000000	1	0.476223	0.072664	0	1	0	0	0	0	
...	...	...	...	...	...	...	...	...	...	...	

3. Pemodelan

- Pemodelan dilakukan menggunakan algoritma k-means seperti dibawah:

Algoritma 4.1 ***k-means clustering***

k-means(*D*, *k*)

Pilih sejumlah *k* objek secara acak dari himpunan data *D* sebagai *centroid* awal — **Langkah 1**

repeat

  for semua objek di dalam *D* — **Langkah 2**

    Masukkan setiap objek yang bukan *centroid* ke klaster yang paling dekat di antara *k* klaster yang ada

  end — **Langkah 3**

  Perbarui setiap *centroid* dengan menghitung rata-rata dari semua objek yang berada di dalam klaster tersebut

until tidak ada perubahan *centroid*

- Dengan perhitungan jarak antar centroid dan objek menggunakan rumus Euclidean, yaitu :

$$d(\mathbf{p}, \mathbf{q}) = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}$$

p, q = two points in Euclidean n-space

q_i, p_i = Euclidean vectors, starting from the origin of the space (initial point)

n = n-space

dengan mengikuti algoritma k-means tersebut, maka didapatlah model kmeans yang saya buat sebagai berikut:

```
class Kmeans:
  def __init__(self, k, max_iter):
    self.k = k
    self.max_iter = max_iter
    self.centroid = []

  def euclidean(self, x1, x2):
    distance = np.sqrt(((x2-x1)**2).sum(axis=0))
    return distance

  def fit(self,data):

    labels = [-1]*len(data)
    self.centroid = data.sample(self.k)
    self.centroid = self.centroid.to_numpy()
    data = data.to_numpy()

    for i in range(self.max_iter):
      self.inertia = 0
      clusters = [[] for i in range(self.k)]

      for x in range(len(data)):

        distance = []
        for indeks in range(len(self.centroid)):
          distance.append(self.euclidean(data[x], self.centroid[indeks]))

        label = distance.index(min(distance))
        clusters[label].append(data[x])
        labels[x] = label

        self.inertia += (min(distance))**2

      temp = np.copy(self.centroid)
      for indeks in range(self.k):
        self.centroid[indeks] = np.mean(clusters[indeks], axis=0)

      condition = temp == self.centroid
      if condition.all():
        break

    return labels
```

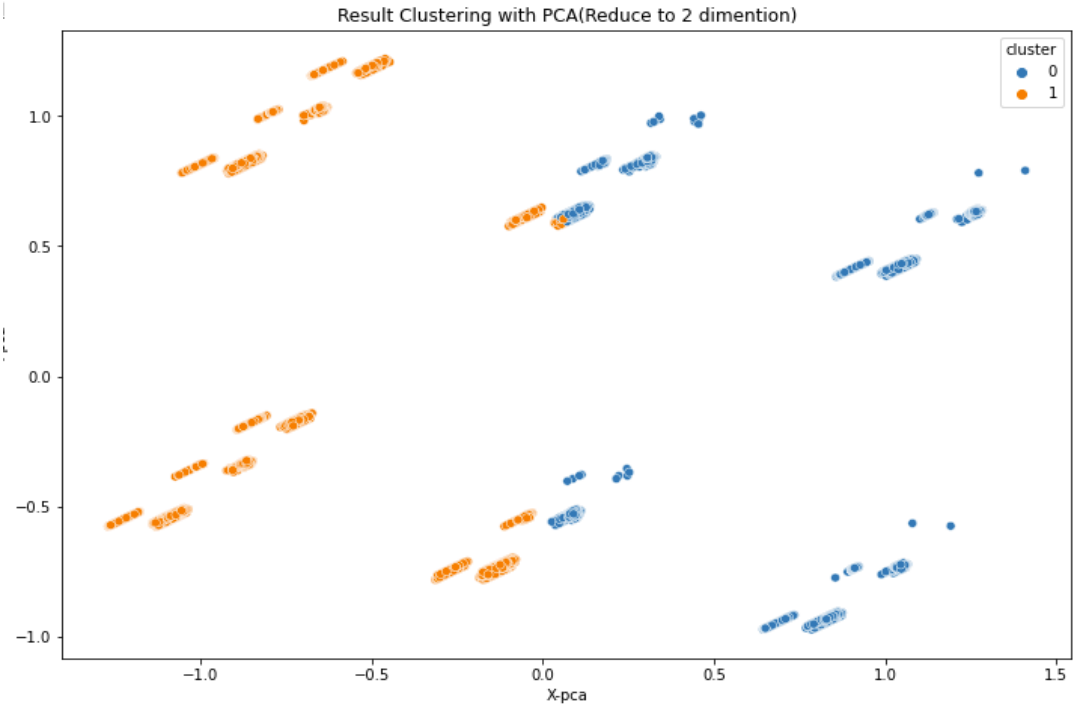
Idenya adalah :

- Centroid awal diambil dari dataset sebanyak K. Centroid dan Data diubah ke representasi array numpy agar lebih ringan saat perhitungan.
- mencari minimum distance(distance/jarak dihitung menggunakan rumus euclidean) dari tiap data ke para centroid tadi.

- Setelah semua data sudah dihitung distance nya, maka akan diupdate centroid nya, dengan mencari mean/rata-rata tiap cluster dari centroidnya.
- Algoritma ini akan berhenti apabila centroid tidak lagi memiliki perubahan(centroid sekarang nilai nya sama dengan centroid sebelumnya(n-1)) sehingga tidak perlu looping sampai maximum iterasi atau pada *worst-case*-nya akan berhenti pada maximum iterasi yang telah ditentukan jumlahnya.
- Pada kmeans ini saya membuat nilai inertia agar nantinya dapat digunakan untuk mencari wcss(within-cluster sum of squares) pada evaluasi kmeans dengan menggunakan elbow method.
- Pada class(model) Kmeans ini, clustering dilakukan di fungsi *fit()* yang akan me-return labels yaitu hasil cluster.

4. Eksperimen

- Saya melakukan eksperimen dengan memanggil model Kmeans dengan parameter K = 2 dan maximum iterasi = 100 kali.
- Setelah selesai fit model Kmeans, maka kita akan mendapatkan return label cluster dari dataset
- Selanjutnya saya melakukan plot dataset dengan label yang sudah didapat. Pada plot ini, saya menggunakan Principal Component Analysis (PCA). PCA ini digunakan untuk mereduksi feature yang kita punya sebelumnya, dengan PCA ini saya mereduksi feature menjadi hanya 2 dimensi agar dapat di visualisasikan plot.
- Berikut hasil dari plot model Kmeans yang telah saya buat:

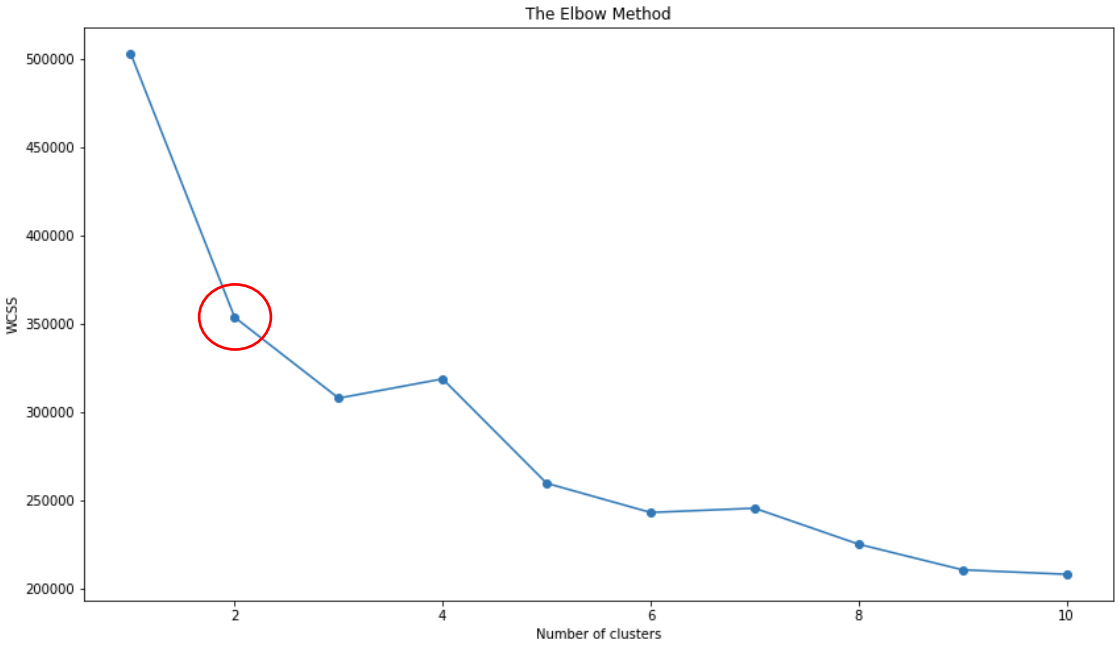


5. Evaluasi

Pada evaluasi model Kmeans ini, saya menggunakan elbow method yang dapat memberi tahu pada K berapa (banyak cluster) yang efektif pada data yang sudah saya eksplorasi sebelumnya.

Pada inertia yang terdapat di model kmeans yang sudah saya buat tadi, kita dapat melihat hasil elbow method yang menggunakan nilai inertia atau wcss(within-cluster sum of squares).

Pada saat memakai elbow method ini, saya menjalankan algoritma mencoba algoritma Kmeans sebanyak 10 kali (dicoba per-K sampai 10 K).



Dapat kita lihat dari hasil elbow method ini, bahwa K = 2 merupakan cluster yang efektif untuk data ini.

6. Kesimpulan

Dari proses reading data, exploratory data analysis, dan clustering yang telah dijalani, dapat disimpulkan bahwa :

- Berdasarkan elbow method, untuk clustering ini K=2 adalah jumlah k cluster yang efektif.
- Untuk visualisasi, hasil clustering harus direduksi dengan PCA terlebih dahulu agar terlihat jelas sebaran cluster nya.
- Bila dilihat dari elbow method, makin banyak K yang digunakan saat kmeans maka total minimum jarak makin sendikit karena centroid yang menyebar dapat memperkecil jarak antar data dan centroid.
- Dataset yang diberikan kurang baik untuk diclustering, bisa dilihat dari tiap feature yang kurang berkorelasi dan visualisasi yang perlu di-PCA.

Link video presentasi :

https://youtu.be/vmEFZlQ_al0

Link berkas csv, berisi data hasil eksplorasi digabung dengan hasil cluster nya:

(data_cluster.csv) di https://drive.google.com/drive/folders/1BGqszAdrgMq5jBEeh-LsUkZURi3jVnC8?usp=sharing