Clustering

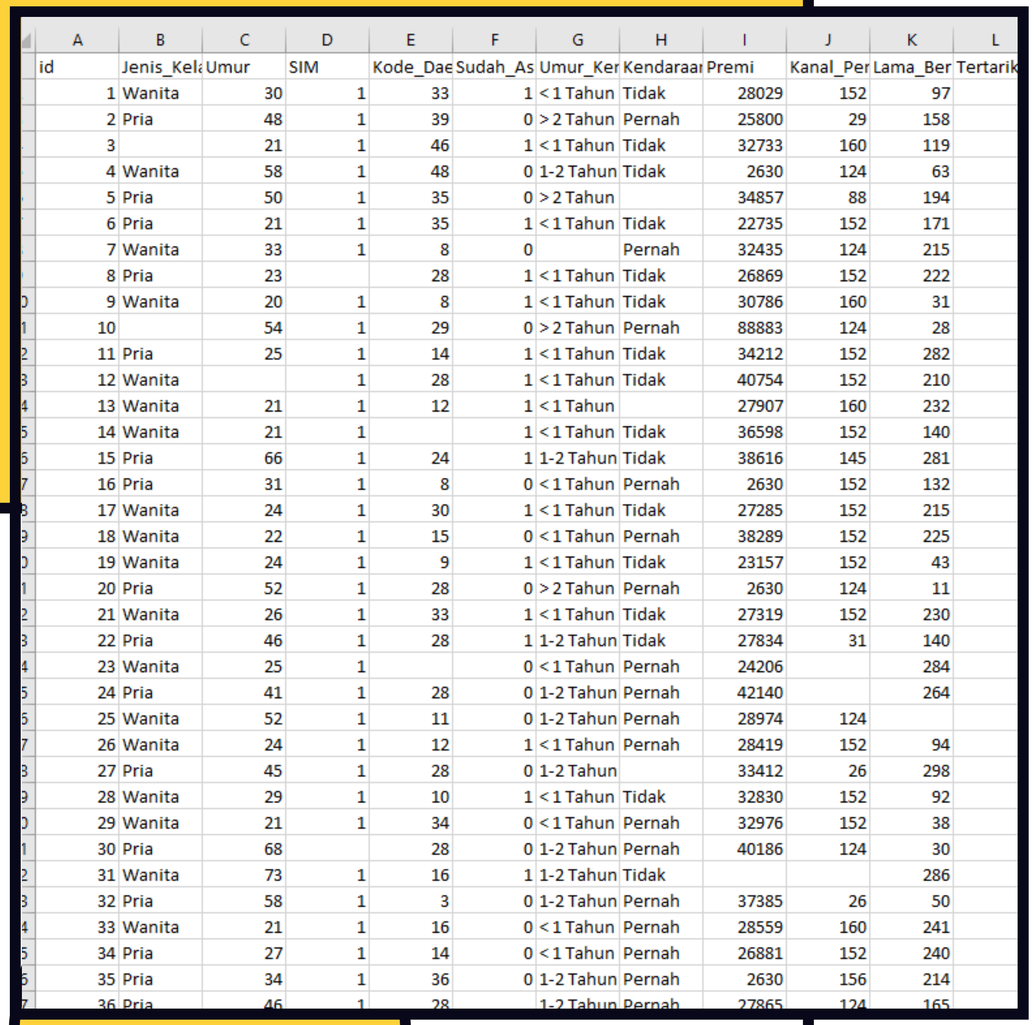
Muhammad Furqon Fahlevi / 1301194214
Designer at Templatery

Clustering

Topics List

1. Problem Formulation
2. Data Exploration & Preparation
3. Modeling
4. Evaluation
5. Experiment
6. Conclusion

Problem Formulation



	A	B	C	D	E	F	G	H	I	J	K	L
	id	Jenis_Kel	Umur	SIM	Kode_Dae	Sudah_As	Umur_Ker	Kendaraa	Premi	Kanal_Per	Lama_Ber	Tertarik
	1	Wanita	30	1	33	1	< 1 Tahun	Tidak	28029	152	97	
	2	Pria	48	1	39	0	> 2 Tahun	Pernah	25800	29	158	
	3		21	1	46	1	< 1 Tahun	Tidak	32733	160	119	
	4	Wanita	58	1	48	0	1-2 Tahun	Tidak	2630	124	63	
	5	Pria	50	1	35	0	> 2 Tahun		34857	88	194	
	6	Pria	21	1	35	1	< 1 Tahun	Tidak	22735	152	171	
	7	Wanita	33	1	8	0		Pernah	32435	124	215	
	8	Pria	23		28	1	< 1 Tahun	Tidak	26869	152	222	
0	9	Wanita	20	1	8	1	< 1 Tahun	Tidak	30786	160	31	
1	10		54	1	29	0	> 2 Tahun	Pernah	88883	124	28	
2	11	Pria	25	1	14	1	< 1 Tahun	Tidak	34212	152	282	
3	12	Wanita		1	28	1	< 1 Tahun	Tidak	40754	152	210	
4	13	Wanita	21	1	12	1	< 1 Tahun		27907	160	232	
5	14	Wanita	21	1		1	< 1 Tahun	Tidak	36598	152	140	
6	15	Pria	66	1	24	1	1-2 Tahun	Tidak	38616	145	281	
7	16	Pria	31	1	8	0	< 1 Tahun	Pernah	2630	152	132	
8	17	Wanita	24	1	30	1	< 1 Tahun	Tidak	27285	152	215	
9	18	Wanita	22	1	15	0	< 1 Tahun	Pernah	38289	152	225	
0	19	Wanita	24	1	9	1	< 1 Tahun	Tidak	23157	152	43	
1	20	Pria	52	1	28	0	> 2 Tahun	Pernah	2630	124	11	
2	21	Wanita	26	1	33	1	< 1 Tahun	Tidak	27319	152	230	
3	22	Pria	46	1	28	1	1-2 Tahun	Tidak	27834	31	140	
4	23	Wanita	25	1		0	< 1 Tahun	Pernah	24206		284	
5	24	Pria	41	1	28	0	1-2 Tahun	Pernah	42140		264	
6	25	Wanita	52	1	11	0	1-2 Tahun	Pernah	28974	124		
7	26	Wanita	24	1	12	1	< 1 Tahun	Pernah	28419	152	94	
8	27	Pria	45	1	28	0	1-2 Tahun		33412	26	298	
9	28	Wanita	29	1	10	1	< 1 Tahun	Tidak	32830	152	92	
0	29	Wanita	21	1	34	0	< 1 Tahun	Pernah	32976	152	38	
1	30	Pria	68		28	0	1-2 Tahun	Pernah	40186	124	30	
2	31	Wanita	73	1	16	1	1-2 Tahun	Tidak			286	
3	32	Pria	58	1	3	0	1-2 Tahun	Pernah	37385	26	50	
4	33	Wanita	21	1	16	0	< 1 Tahun	Pernah	28559	160	241	
5	34	Pria	27	1	14	0	< 1 Tahun	Pernah	26881	152	240	
6	35	Pria	34	1	36	0	1-2 Tahun	Pernah	2630	156	214	
7	36	Pria	46	1	28		1-2 Tahun	Pernah	27865	124	165	

Problem Formulation

Vehicle datasets with multiple variables or columns. There are 2 selected columns from datasets that will go through several stages (Pre-Processing, Clustering).

Data Exploration & Preparation

Import Library

Several libraries used for clustering.

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
from scipy import stats
```

Import Datasets

```
# Import dataset into new dataframe named data_train
data_train = pd.read_csv("kendaraan_train.csv")
data_train.head()
```

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

Using pandas as pd to import csv file into colab with name of variable “data_train”.

Check NaN data

```
# Check NaN data
data_train.isna()
```

285831 rows × 12 columns

```
# Drop NaN data
data_train = data_train.dropna()
data_train
```

171068 rows × 12 columns

Cleansing data or drop the data column or variable that has a NaN value. This serves to avoid empty data while doing the processing.

There are 114.763 NaN data.

Check duplicated data

To check is there any duplicated data or no.

```
# Check duplicate data
print("Duplicated data:",data_train.duplicated().sum())
```

```
Duplicated data: 0
```

Catogorical into Numeric data

```
# Categorical -> Numeric data
data_train['Jenis_Kelamin'] = data_train['Jenis_Kelamin'].map({'Pria':0, 'wanita':1})
data_train['Kendaraan_Rusak'] = data_train['Kendaraan_Rusak'].map({'Tidak':0, 'Pernah':1})
data_train['Umur_Kendaraan'] = data_train['Umur_Kendaraan'].map({'< 1 Tahun':0, '1-2 Tahun':1, '> 2 Tahun':2})
data_train.head()
```

	id	Jenis_Kelamin	Umur	SIM	Kode_Daerah	Sudah_Asuransi	Umur_Kendaraan	Kendaraan_Rusak	Premi	Kanal_Penjualan	Lama_Berlangganan	Tertar
0	1	1	30.0	1.0	33.0	1.0	0	0	28029.0	152.0	97.0	
1	2	0	48.0	1.0	39.0	0.0	2	1	25800.0	29.0	158.0	
3	4	1	58.0	1.0	48.0	0.0	1	0	2630.0	124.0	83.0	
5	6	0	21.0	1.0	35.0	1.0	0	0	32735.0	152.0	171.0	
8	9	1	20.0	1.0	8.0	1.0	0	0	30786.0	160.0	31.0	

Algorithm cannot operate on label data directly. They require all input and output variables to be numeric.

Drop unnecessary data

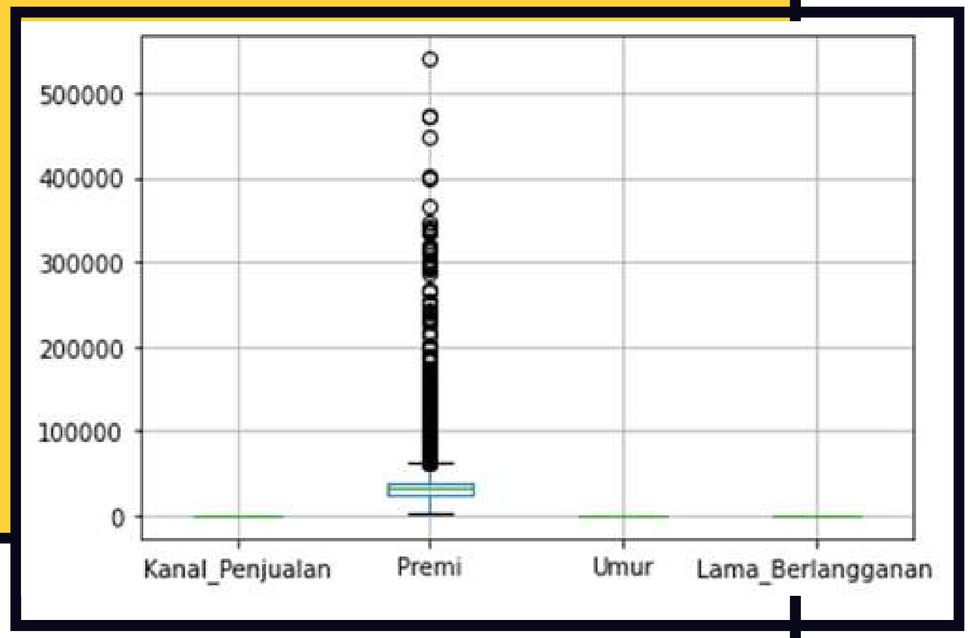
```
# Drop unnecessary data
del data_train['id']
del data_train['Tertarik']
data_train.head()
```

	Jenis_Kelamin	Umur	SIM	Kode_Daerah	Sudah_Asuransi	Umur_Kendaraan	Kendaraan_Rusak	Premi	Kanal_Penjualan	Lama_Berlangganan
0	1	30.0	1.0	33.0	1.0	0	0	28029.0	152.0	97.0
1	0	48.0	1.0	39.0	0.0	2	1	25800.0	29.0	158.0
3	1	58.0	1.0	46.0	0.0	1	0	2630.0	124.0	63.0
5	0	21.0	1.0	35.0	1.0	0	0	22735.0	152.0	171.0
8	1	20.0	1.0	8.0	1.0	0	0	30786.0	160.0	31.0

Drop unnecessary data

Since ‘Id’ and ‘Tertarik’ is not related with clustering, therefore the column will be dropped.

Check outliers



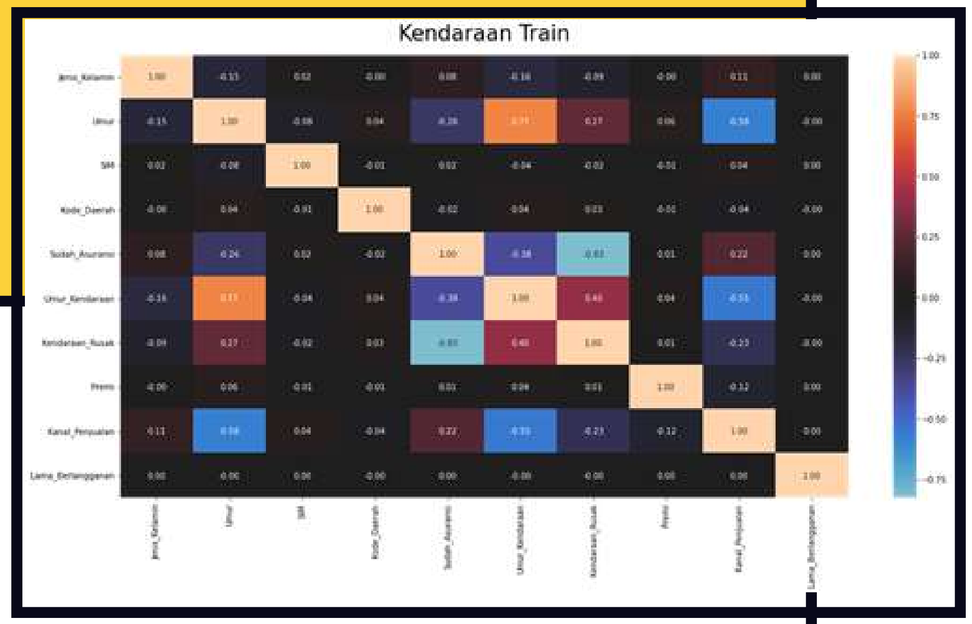

```
z = np.abs(stats.zscore(data_train[['Kanal_Penjualan', 'Premi', 'Umur', 'Lama_Berlangganan']]))
threshold = 3

data_new = data_train[(z < threshold).all(axis=1)]
data_new
```

	Jenis_Kelamin	Umur	SIM	Kode_Daerah	Sudah_Asuransi	Umur_Kendaraan	Kendaraan_Rusak	Premi	Kanal_Penjualan	Lama_Berlangganan
0	1	30.0	1.0	33.0	1.0	0	0	28029.0	152.0	97.0
1	0	48.0	1.0	39.0	0.0	2	1	25800.0	29.0	158.0
3	1	58.0	1.0	48.0	0.0	1	0	2630.0	124.0	63.0
5	0	21.0	1.0	35.0	1.0	0	0	22735.0	152.0	171.0
8	1	20.0	1.0	8.0	1.0	0	0	30786.0	160.0	31.0

Check the outliers for these columns because this is the candidate who may be continued to the clustering stage. The outliers will be dropped using z-score

Heatmap



This the correlation between columns

Modeling

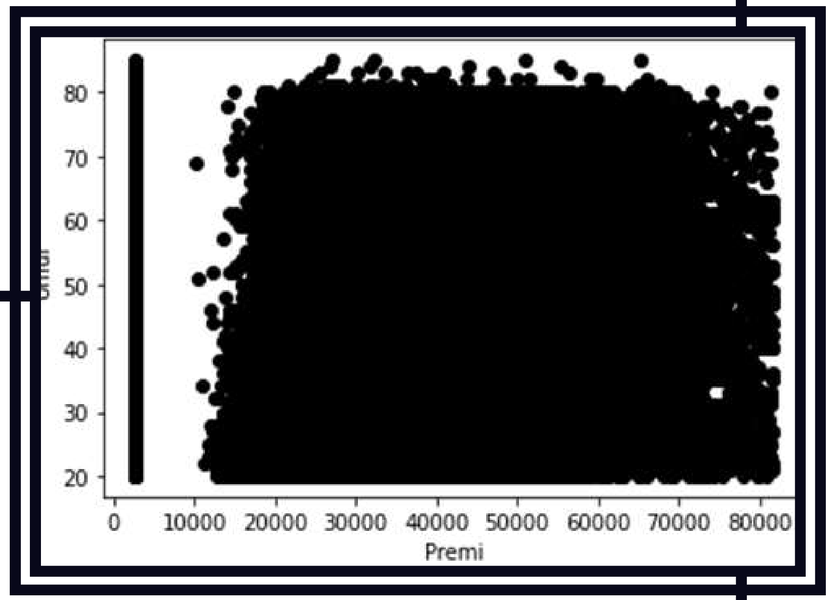
Select datas for clustering

```
# Select 2 datas / columns for clustering
data_selected = data_new.loc[:, ['Premi', 'Umur']]
data_selected.head()
```

	Premi	Umur
0	28029.0	30.0
1	25800.0	48.0
3	2630.0	58.0
5	22735.0	21.0
8	30786.0	20.0

‘Premi’ and ‘Umur’ is the selected data and the correlation between them is 0.06.

Data before clustering

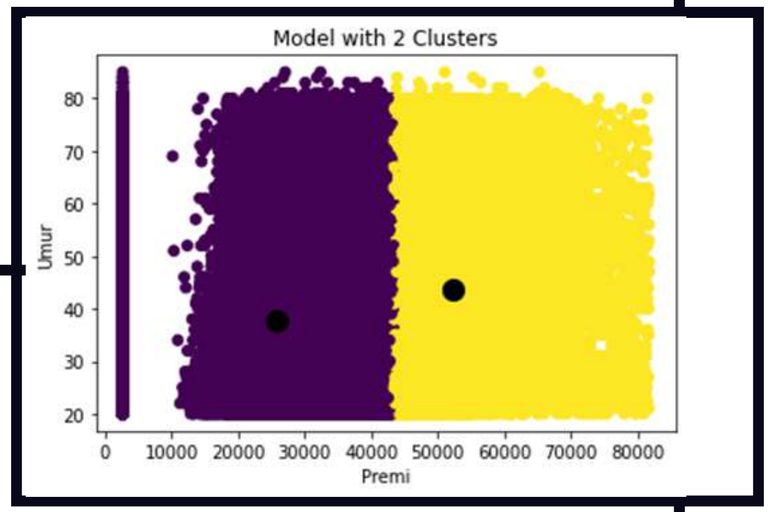


Define K-Means

```
# Define K-Means
def kMeans(ds_val, k):
  diff = 1
  cluster = np.zeros(ds_val.shape[0])
  centroids = data_selected.sample(n=k).values
  while diff:
    for i, row in enumerate(ds_val):
      distance = float('inf')
      for idx, centroid in enumerate(centroids):
        eu_distance = np.sqrt((centroid[0]-row[0])**2 + (centroid[1]-row[1])**2)
        if distance > eu_distance:
          distance = eu_distance
          cluster[i] = idx
    new_centroids = pd.DataFrame(ds_val).groupby(by=cluster).mean().values
    if np.count_nonzero(centroids-new_centroids) == 0:
      diff = 0
    else:
        centroids = new_centroids
    return centroids, cluster
```

Using K-Means algorithm for clustering with Euclidian Distance.

Data after clustering



Data visualization with K = 2

Calculate WCSS

$$\text{WCSS} = \sum_{C_k}^{C_n} \left(\sum_{d_i in C_i}^{d_m} distance(d_i, C_k)^2 \right)$$

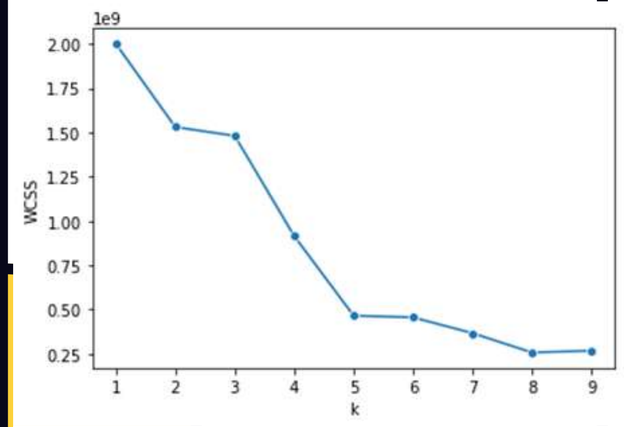
Where,
C is the cluster centroids and d is the data point in each Cluster.

```python
# Calculate WCSS
def calculate_cost(X, centroids, cluster):
  sum = 0
  for i, val in enumerate(X):
    sum += np.sqrt((centroids[int(cluster[i]), 0]-val[0])**2 +(centroids[int(cluster[i]), 1]-val[1])**2)
  return sum
```

WCSS is the sum of squared distance between each point and the centroid in a cluster. Calculate the WCSS based on the formula, and we implement into the code.

Elbow Line & Plot

```
# Find K value
cost_list = []
for k in range(1,10):
    centroids, cluster = kMeans(ds_val, k)
    cost = calculate_cost(ds_val, centroids, cluster)
    cost_list.append(cost)
```



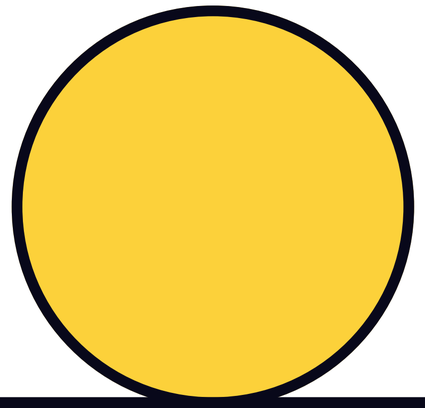
Varying the number of clusters (k) with range 1-10. For each value of k, we are calculating WCSS. After that, we make plot with similar range.

Evaluation

Silhouette value

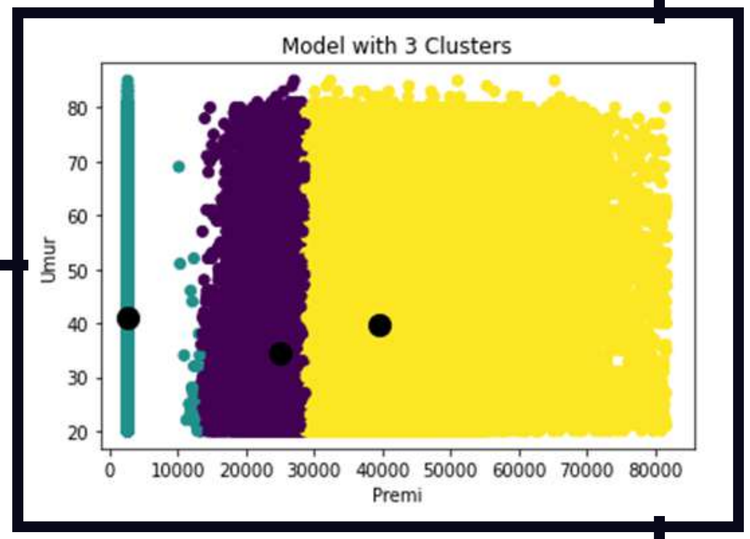
```
0.34022689799573286
```

The evaluation is done by finding Silhouette value. This is the score with K = 2.

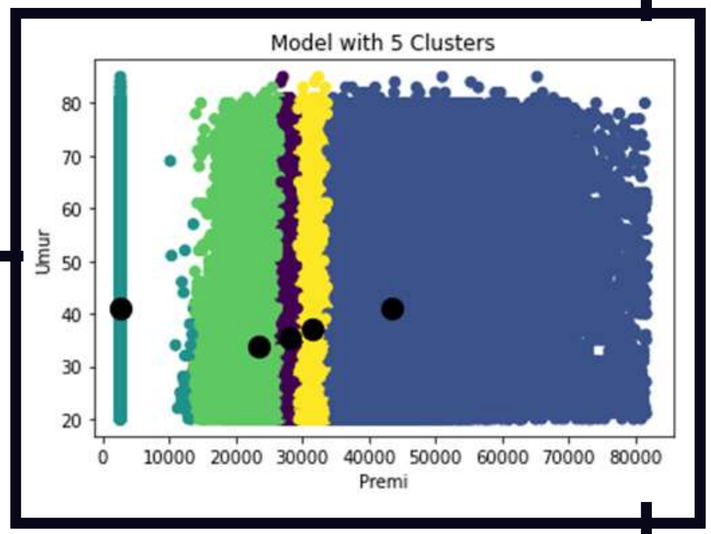


Experiment

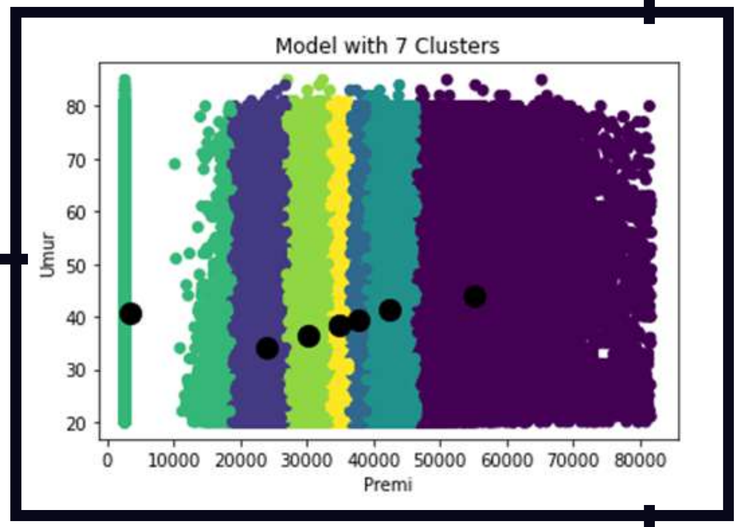
Model with 3 clusters



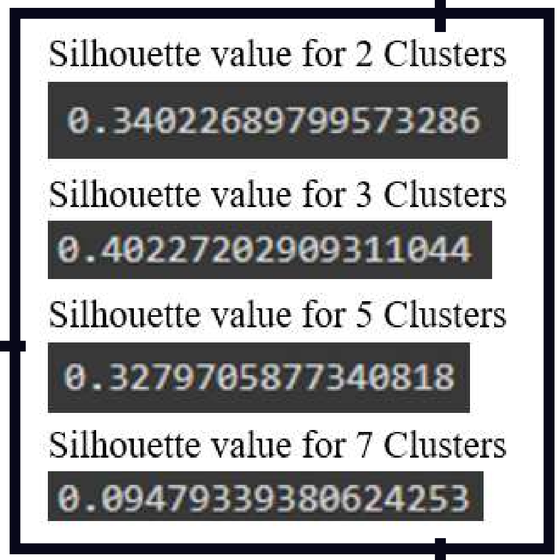
Model with 5 clusters



Model with 7 clusters



Conclusion



Conclusion

After doing the experiment, we got the Silhouette values from the selected clusters.

Based on Silhouette method that a value close to 1 is the most optimal. We can conclude that 3 clusters have the most optimal value with K = 3.

Thank you!

Muhammad Furqon Fahlevi | 1301194214 | IF-43-INT