# IMPLEMENTASI KLASTERING K-NEAREST DAN KLASIFIKASI KNN DAN SVM

Laporan ini dibuat untuk memenuhi tugas besar

Mata Pembelajaran Mesin

Universitas Telkom


Disusun oleh:

Edi Dwi Nugroho 1301174409


**S1 INFORMATIKA**

**FAKULTAS INFORMATIKA**

**UNIVERSITAS TELKOM**

**BANDUNG**

**2020**

# BAB I
# Pendahuluan

## 1. Identifikasi Masalah Dan Pengolahan Data

Dalam tugas besar pembelajaran mesin ini akan dibuat sebuah pengimplementasian dari system klasifikasi dan system clustering dengan memanfaatkan datashet used_cars.csv yang telah diberikan. Untuk clustering metode yang digunakan adalah metode kmeans karena metode paling sederhana dan umum. Kmeans mempunyai kemampuan mengelompokkan data dalam jumlah cukup besar dengan waktu komputasi yang cukup cepat. Dan untuk clustering data yang dimanfaatkan adalah label “odometer” dan “model” dari dataset. Classification merupakan metode mengelompokkan masing-masing label untuk setiap kondisi. Dalam metode ini, membutuhkan untuk menemukan model yang dapat menjelaskan class attribute sebagai fungsi dari input attribute.

# BAB II
# Analisis dan Perancangan

## 2.1 Hasil Yang Diharapkan

Didalam tugas besar ini akan ada 5 klasifikasi pelatian yaitu :

'gas' : 1,
'diesel' : 2,
'electric' : 3,
'hybrid' : 4,
'other' : 5

## 2.2 Identifikasi Sumber Data

Dataset Used cars adalah dataset yang diberikan dalam tugas besar ini. Dataset ini tediri dari 20.000 baris data. Kolom yang akan digunakan dalam tugas besar ini adalah kolom type dan kolom transmisi.

# BAB II
# Analisis dan Perancangan

## 2.1 Persiapan Data

Pertama yang harus dilakukan adalah meload dataset

```
import math
import matplotlib as mpl
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
from pandas import DataFrame
import numpy as np
import seaborn as sns
sns.set(style="darkgrid")


url = 'https://raw.githubusercontent.com/edinugroho/Final-Task-
Machine-Learning/master/used_cars.csv'
used_cars = pd.read_csv(url)

used_cars.region = pd.factorize(used_cars.region)[0]
used_cars.url = pd.factorize(used_cars.url)[0]
used_cars.region_url = pd.factorize(used_cars.region_url)[0]
used_cars.manufacturer = pd.factorize(used_cars.manufacturer)[0]
used_cars.model = pd.factorize(used_cars.model)[0]
used_cars.year = pd.factorize(used_cars.year)[0]
```

```
used_cars.condition = pd.factorize(used_cars.condition)[0]
used_cars.cylinders = pd.factorize(used_cars.cylinders)[0]
used_cars.odometer = pd.factorize(used_cars.odometer)[0]
used_cars.title_status = pd.factorize(used_cars.title_status)[0]
used_cars.transmission = pd.factorize(used_cars.transmission)[0]
used_cars.vin = pd.factorize(used_cars.vin)[0]
used_cars.model = pd.factorize(used_cars.model)[0]
used_cars.image_url = pd.factorize(used_cars.image_url)[0]
used_cars.lat = pd.factorize(used_cars.lat)[0]
used_cars.drive = pd.factorize(used_cars.drive)[0]
used_cars['long'] = pd.factorize(used_cars['long'])[0]
used_cars['size'] = pd.factorize(used_cars['size'])[0]
used_cars['type'] = pd.factorize(used_cars['type'])[0]
used_cars['paint_color'] = pd.factorize(used_cars['paint_color'])[0
]
used_cars['description'] = pd.factorize(used_cars['description'])[0
]
used_cars['county'] = pd.factorize(used_cars['county'])[0]
used_cars['state'] = pd.factorize(used_cars['state'])[0]
```

simpan data bersih ke dalam file csv baru

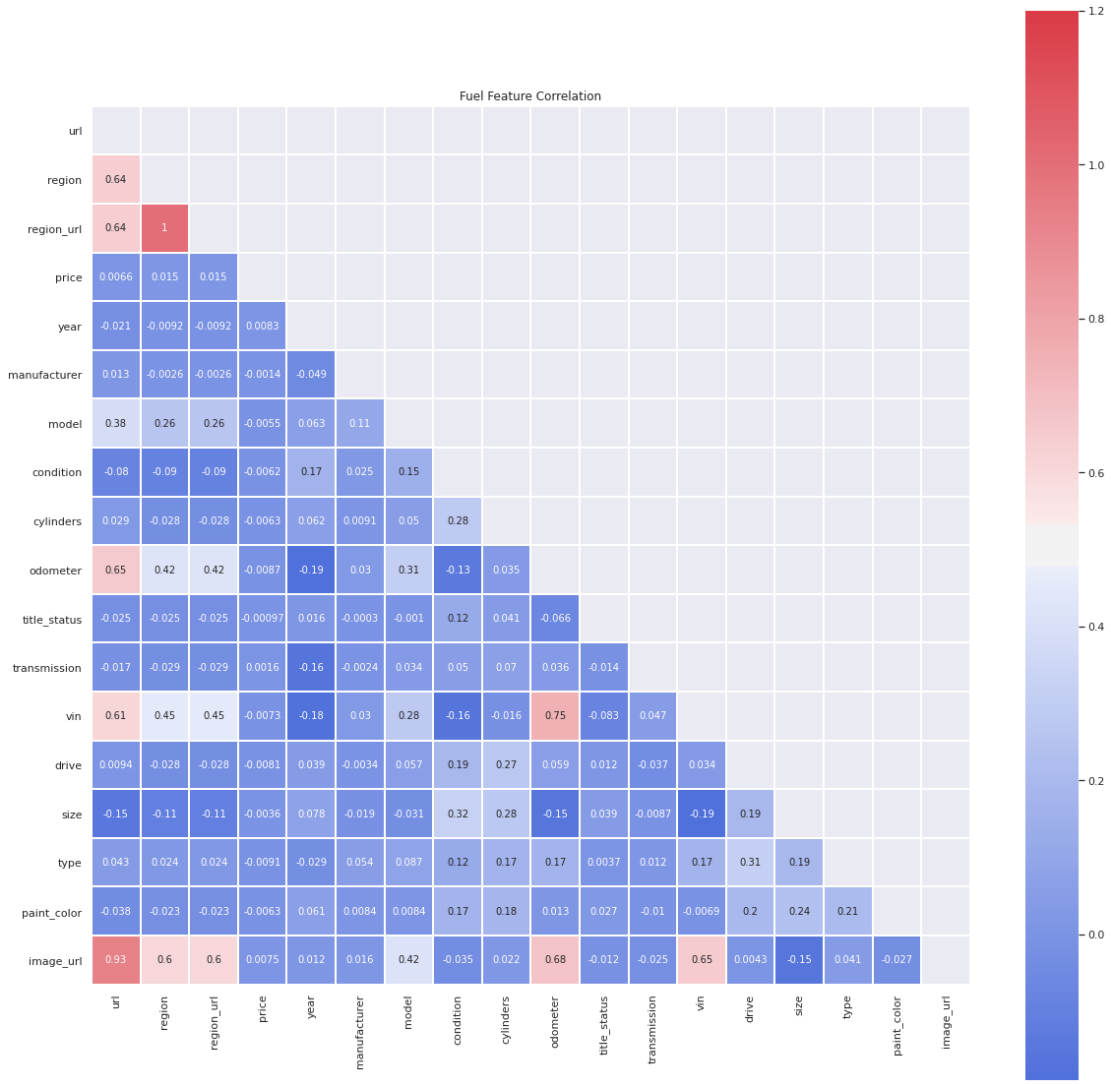
```
used_cars.to_csv('clean-data.csv')
```

## 1. Klasifikasi

Dalam klasifikasi ini saya menggunakan 2 metode yaitu KNN dan SVM. Setelah diperoleh data yang sudah bersih maka langkah langkah klasifikasi selanjutnya akan disebutkan dibawah ini.

### Eksplorasi Data

Setelah didapatkan data yang sudah bersih maka langkah selanjutnya adalah eksplorasi data. Agar preprocessing data berhasil, sangat penting untuk memiliki gambaran keseluruhan data. Deskripsi statistik dasar dapat digunakan untuk mengidentifikasi properti data dan menyoroti nilai data mana yang harus diperlakukan sebagai noise atau outlier.




Berikut matrix korelasi yang diperoleh

## Pre-Processing Data

Preprocessing data adalah langkah penting untuk setiap masalah analisis data. Misi utama pada tahap ini adalah menemukan fitur yang paling prediktif dari data dan sehingga meningkatkan daya prediksi.

Pertama load data yang sudah dibersihkan, lalu hapus baris yang ada nilai kosong di dalamnya. Dan assingn variable predictor. Disini saya menggunakan fuel sebagai predictor.

```
used_cars = pd.read_csv('clean-
data.csv',usecols=['fuel','transmission','type'], index_col=False)
used_cars.dropna(inplace=True)
X = used_cars.drop(['fuel'], axis = 1)
y = used_cars['fuel']
```

setelah di assign data akan di split untuk membagi antara data train dan data test. Saya membaginya menjadi 20% data test dan sisanya data train.

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size
 = 0.2, random_state = 20)
print ('The size of our training "X" (input features) is', X_train.
shape)
print ('The size of our testing "X" (input features) is', X_test.sh
ape)
print ('The size of our training "y" (output feature) is', y_train.
shape)
print ('The size of our testing "y" (output features) is', y_test.s
hape)
```

**Predictive model**

Tahap ini adalah tahap akhir dari tugas besar ini. Disini saya menggunakan algoritma knn dan support vector machine. Tiap tiap model menghasilkan akurasi yang berbeda. Pertama algoritma support vector machine menghasilkan akurasi 89%

```
akurasi :  0.8881083793276467
              precision    recall  f1-score   support

           0       0.80      0.01      0.02       317
           1       0.00      0.00      0.00         6
           2       0.89      1.00      0.94      3537
           3       0.00      0.00      0.00        42
           4       0.00      0.00      0.00        84

    accuracy                           0.89      3986
   macro avg       0.34      0.20      0.19      3986
weighted avg       0.85      0.89      0.84      3986

/usr/local/lib/python3.6/dist-packages/sklearn/metrics/_
  _warn_prf(average, modifier, msg_start, len(result))
```

| | gas | diesel | other | electric | hybrid |
|---|---|---|---|---|---|
| **gas** | 4 | 0 | 313 | 0 | 0 |
| **diesel** | 0 | 0 | 6 | 0 | 0 |
| **other** | 1 | 0 | 3536 | 0 | 0 |
| **electric** | 0 | 0 | 42 | 0 | 0 |
| **hybrid** | 0 | 0 | 84 | 0 | 0 |

Sedangkan untuk algoritma knn menghasilkan 86%

```
akurasi :  0.8627696939287506
              precision    recall  f1-score   support

           0       0.31      0.31      0.31       317
           1       0.00      0.00      0.00         6
           2       0.91      0.94      0.93      3537
           3       0.00      0.00      0.00        42
           4       0.00      0.00      0.00        84

    accuracy                           0.86      3986
   macro avg       0.24      0.25      0.25      3986
weighted avg       0.83      0.86      0.85      3986

/usr/local/lib/python3.6/dist-packages/sklearn/metrics/_
  _warn_prf(average, modifier, msg_start, len(result))
```

| | gas | diesel | other | electric | hybrid |
|---|---|---|---|---|---|
| gas | 99 | 0 | 218 | 0 | 0 |
| diesel | 0 | 0 | 6 | 0 | 0 |
| other | 197 | 0 | 3340 | 0 | 0 |
| electric | 0 | 0 | 42 | 0 | 0 |
| hybrid | 25 | 0 | 59 | 0 | 0 |

## 2. Klastering

Dalam Tugas besar ini saya menggunakan klastering k-means karena algoritmanya cukup mudah untuk diimplementasikan. Untuk memproses algoritma K-means, dimulai dengan kelompok pertama centroid yang dipilih secara acak, yang digunakan sebagai titik awal untuk setiap cluster, dan kemudian melakukan perhitungan berulang (berulang) untuk mengoptimalkan posisi centroid.

### Pilih K buah titik centroid secara acak

Dalam k-mean pertama yang dilakukan adalah generate nilai dari salah satu daya yang ada dalam dataset.

### Menentukan Kelompok Cluster

Pengelompokan data pada k-means adalah penentuan klister dari titik centroid yang telah dipilih pada tahap sebelumnya.

### Perbaharui nilai titik centroid

Setelah kluster terbentuk titik centroid akan diukur jaraknya dengan titik titik di dekatnya, setelah itu titik cluster akan dipindahkan sesuai dengan rata-rata jarak yang telah dihitung.

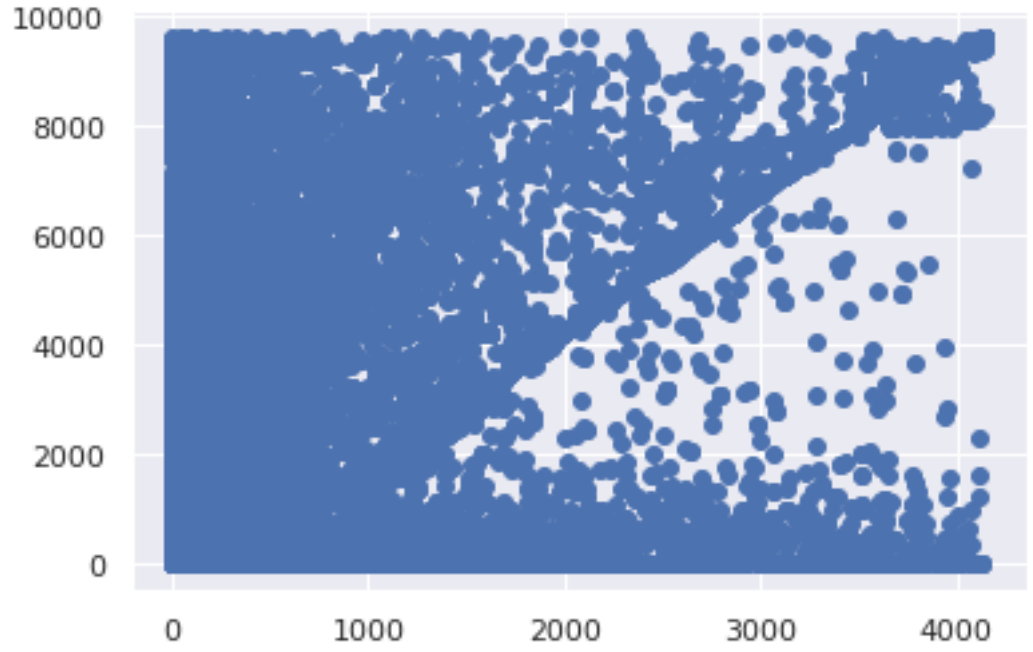
### Penentuan Centroid Klaster

Agar titik centroid berada di tengah tengah cluster maka tahap kedua dan tahap ketiga akan diulangi sampai nilai rata-rata yang didapatkan konstan atau tidak berubah.

Berikut code dari aalgoritma clustering k-means

```
import pandas as pd
import matplotlib.pyplot as plt
import random
used_cars = pd.read_csv('clean-data.csv', index_col=False)
used_cars.replace(to_replace = 0, value = 1)
x = used_cars['model']
y = used_cars['odometer']
plt.scatter(x,y)
plt.show()
```

diatas adalah code untuk mengambil file yang telah dibersihkan sebelumnya. Dan berikut adalah output dari code diatas




```
def distance(k,x,y,z):
  dist = [[],[],[]]
  for i in range(len(k)):
    for j in range(len(x)):
      x1 = np.sqrt((k[i][0]-x[j])**2 + (k[i][1]-y[j])**2)
      dist[i].append(x1)
```

```
  return (dist)
```

di atas adalah fungsi untuk mengukur jarak antar node.

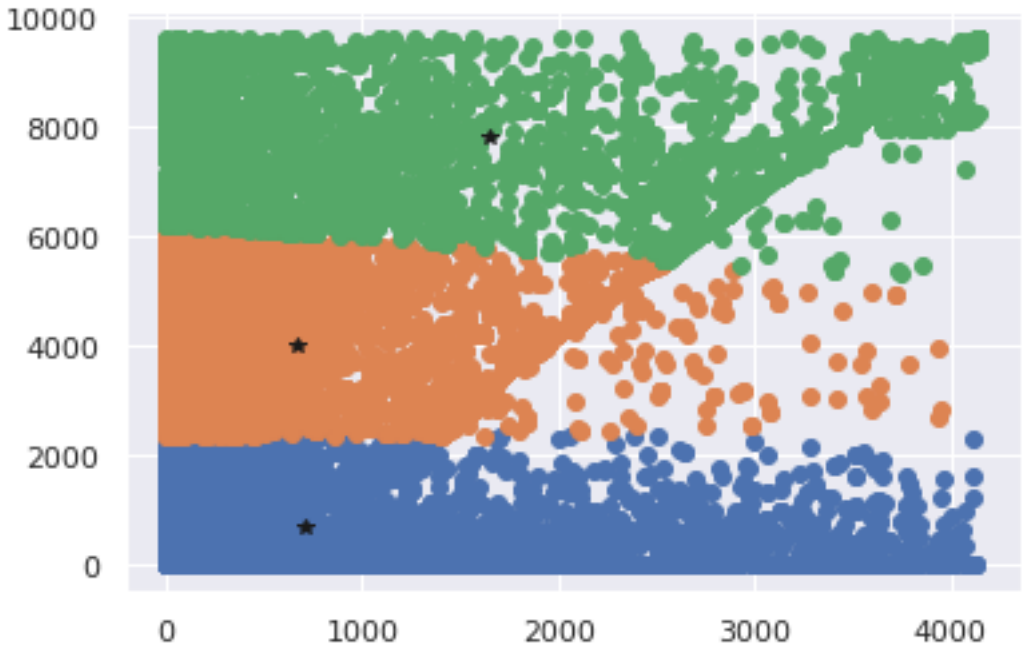
```
def newK(k):
  ncluster = [[],[],[]]
  nk0x = sum(k[0][0])/len(k[0][0])
  nk0y = sum(k[0][1])/len(k[0][1])
  ncluster[0].append(nk0x)
  ncluster[0].append(nk0y)

  nk1x = sum(k[1][0])/len(k[1][0])
  nk1y = sum(k[1][1])/len(k[1][1])
  ncluster[1].append(nk1x)
  ncluster[1].append(nk1y)

  nk2x = sum(k[2][0])/len(k[2][0])
  nk2y = sum(k[2][1])/len(k[2][1])
  ncluster[2].append(nk2x)
  ncluster[2].append(nk2y)
  return(ncluster)
def cluster(x,y,dist):
  clust = [[[],[]],[[],[]],[[],[]]]
  for i in range(len(x)):
    min = dist[0][i]
    for j in range(len(dist)):
      if(dist[j][i] < min):
        min = dist[j][i]
    for l in range(len(dist)):
      if(dist[l][i] == min):
        clust[l][0].append(x[i])
        clust[l][1].append(y[i])
  return (clust)
```

```
centeroid = [[],[],[]]
for i in range (3):
  a = random.randrange(len(used_cars))
  centeroid[i].append(used_cars.model[a])
  centeroid[i].append(used_cars.odometer[a])

cond = True
while(cond == True):
  jarak = distance(centeroid,x,y,used_cars)
  clust = cluster(x,y,jarak)
  centerold = newK(clust)
  if(centeroid == centerold):
    cond = False
  else:
    centeroid = centerold
```

```
for i in range (len(clust)):
  plt.scatter(clust[i][0],clust[i][1])

for i in range (len(centeroid)):
  plt.plot(centeroid[i][0],centeroid[i][1],'k*')
```

kode diatas adalah penggabungan dari fungsi-fungsi yang sudah dibuat diatas.

Berikut hasil dari klastering yang telah dibuat




# BAB III
# Hasil Implementasi

## 3.1 Penutup

## 3.1 Kesimpulan

Dengan dibuatnya tugas besar ini maka dapatkan kesimpulan yaitu :

1. Terdapat 4 langkah dalam pengerjaan tuugas besar ini, yaitu persiapan data, eksplorasi data, pre processing dan modelling. Keempat tahap tersebut saling berkaitan satu sama lain.
2. Setiap model mendapatkan hasil akurasi yang berbeda dengan algoritmanya masing-masing
3. Dari kedua algoritma, support vector machine adalah algoritma yang paling cocok dengan datasheet pada tugas besar ini.
4. Klastering k-means menghasilkan 3 klaster.

## 2.3 Daftar Pustaka


1. https://scikit-learn.org/stable/user_guide.html
2. https://drive.google.com/drive/folders/1VEtnsfMjWstIOv1s5vgruJk9Nm7hAE2y