# MENENTUKAN JENIS BAHAN BAKAR KENDARAAN DENGAN TIPE DAN TRANSMISI

Laporan ini dibuat untuk memenuhi tugas besar

Mata Pembelajaran Mesin



Disusun oleh:

Edi Dwi Nugroho 1301174409

**S1 INFORMATIKA**

**FAKULTAS INFORMATIKA**

**UNIVERSITAS TELKOM**

**BANDUNG**

**2020**

# DAFTAR ISI

*Dibuat secara otomatis. Lihat tutorialnya di:*<https://www.youtube.com/watch?v=QbX0zYkIQyo>

## 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.

## Hasil Yang Diharapkan

Didalam tugas besar ini akan ada 5 klasifikasi pelatian yaitu :

'gas' : 1,

'diesel' : 2,

'electric' : 3,

'hybrid' : 4,

'other' : 5

## 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.

## Langkah Pengerjaan

## 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]

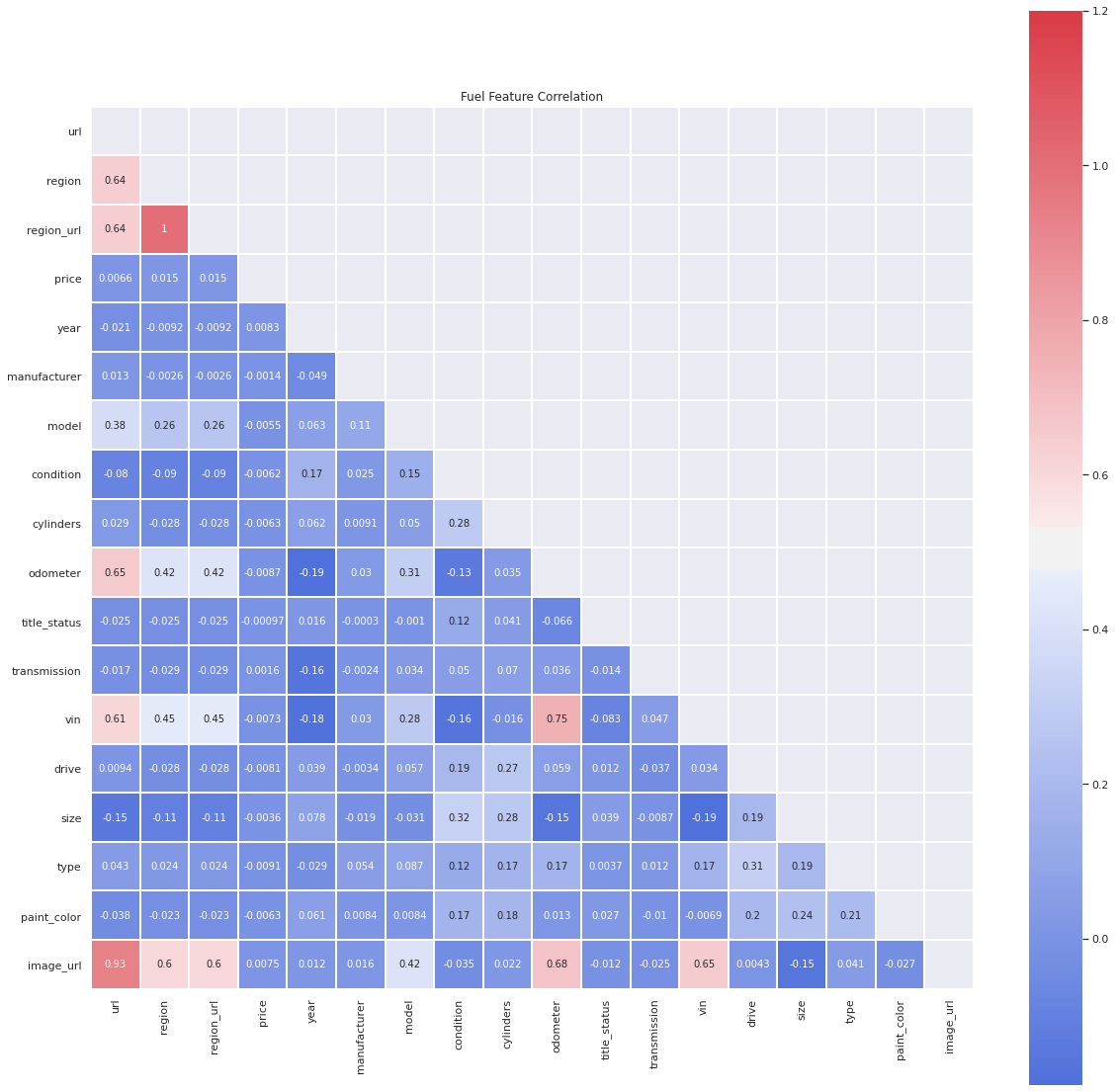
used\_cars['state'] = pd.factorize(used\_cars['state'])[0]

simpan data bersih ke dalam file csv baru

used\_cars.to\_csv('clean-data.csv')

## 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.shape)

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.shape)

Lalu di normalisasi dan dilakukan feature extraction

from sklearn.preprocessing import StandardScaler

# Normalisasi.

scaler =StandardScaler()

Xs = scaler.fit\_transform(X)

from sklearn.decomposition import PCA

# feature extraction

pca = PCA(n\_components=2)

fit = pca.fit(Xs)

langkah selanjutnya adalah membuat PCA, dengan pca ini data dapat dibaca dan dikurangi dimensinya agar lebih mudah dibaca.

X\_pca = pca.transform(Xs)

PCA\_df = pd.DataFrame()

PCA\_df['PCA\_1'] = X\_pca[:,0]

PCA\_df['PCA\_2'] = X\_pca[:,1]

plt.plot(PCA\_df['PCA\_1'][used\_cars.fuel == 'gas'],PCA\_df['PCA\_2'][used\_cars.fuel == 'gas'],'o', alpha = 0.7, color = 'r')

plt.plot(PCA\_df['PCA\_1'][used\_cars.fuel == 'diesel'],PCA\_df['PCA\_2'][used\_cars.fuel == 'diesel'],'o', alpha = 0.7, color = 'b')

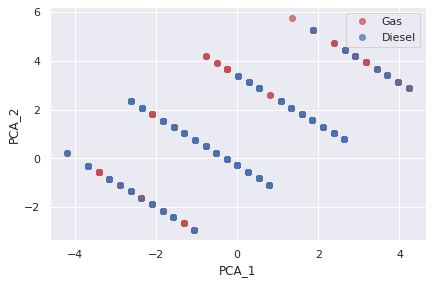
plt.xlabel('PCA\_1')

plt.ylabel('PCA\_2')

plt.legend(['Gas','Diesel'])

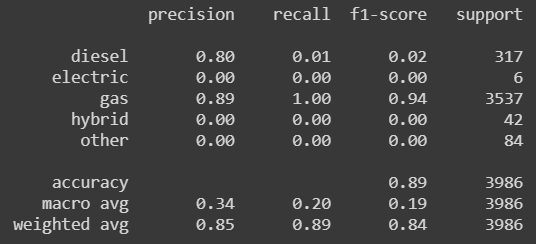
plt.show()

berikut keluaran dari proses PCA

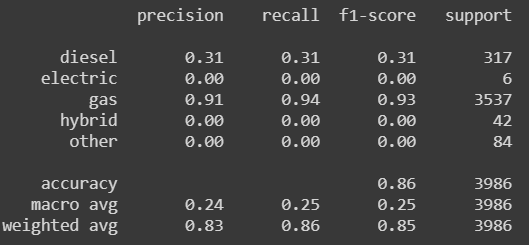


## Predictive model

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%



Sedangkan untuk algoritma knn menghasilkan 86%



## Penutup

## 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.

## Daftar Pustaka

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