LATIHAN PRAKTEK MACHINE LEARNING ML A, ML B

Rin Rin Nurmalasari

Sebelum membuat program pastikan terlebih dahulu menginstall code editor. Misal visual studio, jupyter notebook dll. Pilih salah satu editor untuk digunakan:

Cara install python dan vscode: https://youtu.be/xETkm9H6aaY?feature=shared

Jupyter notebook pada vscode: https://youtu.be/GoUh8CWNI7Y?feature=shared

Cara install jupyter notebook pada anaconda : https://youtu.be/AnIU_QHyUXE?feature=shared

LATIHAN 1: DETEKSI EMOSI MENGGUNAKAN ALGORTIMA SVM DAN NAÏVE BAYES

Dataset yang digunakan adalah dataset ISEAR (International Survey on Emotion Antecedents and Reactions). ISEAR adalah sebuah data tentang emosi yang dimulai dikumpulkan oleh psikolog di seluruh dunia, dipimpin oleh Klaus R. Scherer dan Harald Wallbott. Pada data isear terdiri dari 7 emosi utama (senang, takut, marah, sedih, jijik, malu, dan rasa bersalah). Data ini memuat laporan tentang tujuh emosi sekitar 3000 responden di 37 negara di 5 benua.

1	joy	On days when I feel close to my partner and other friends. When I feel at peace with myself and also experience a close contact with people whom I regard greatly.
2	fear	Every time I imagine that someone I love or I could contact a serious illness, even death.
3	anger	When I had been obviously unjustly treated and had no possibility of elucidating this.
4	sadness	When I think about the short time that we live and relate it to the periods of my life when I think that I did not use this short time.
5	disgust	At a gathering I found myself involuntarily sitting next to two people who expressed opinions that I considered very low and discriminating.
6	shame	When I realized that I was directing the feelings of discontent with myself at my partner and this way was trying to put the blame on him instead of sorting out my own feeliings.
7	guilt	I feel guilty when when I realize that I consider material things more important than caring for my relatives. I feel very self-centered.

Preprocessing

Hal pertama yang harus dilakukan pada data *text* adalah melakukan preprocessing. Text preprocessing adalah tahapan untuk mempersiapkan teks menjadi data yang akan diolah di tahapan berikutnya. Inputan awal pada proses ini adalah berupa dokumen teks. Teks yang akan dilakukan proses text mining pada umumnya memiliki beberapa karakteristik, diantaranya adalah memiliki dimensi yang tinggi, terdapat noise pada data, dan terdapat struktur teks yang tidak baik. Agar dapat dihasilkan fitur yang baik dan mewakili data dengan baik, perlu dilakukan

tahapan preprocessing . Adapun hal-hal yang dapat dilakukan dalam preprocessing pada kasus data seperti ini yang dilakukan antara lain case folding, tokenizing, stopword removal dan stemming.

1. Case Folding
   Pada tahap ini dilakukan perubahan pada semua huruf dalam dokumen menjadi huruf kecil dan menghilangkan karakter selain a-z, dengan tujuan untuk menyeragamkan karakter dalam dokumen tersebut
2. Convert Negation
   Convert Negation merupakan proses konversi kata-kata negasi yang terdapat pada suatu kalimat, karena kata negasi mempunyai pengaruh dalam merubah nilai emosi pada sebuah kalimat. Jika terdapat kata negasi maka akan disatukan dengan kata setelahnya. Kata - kata negasi tersebut meliputi kata "bukan", "tidak", "tak", "tanpa" dan "jangan". Langkah – langkah pada tahap convert negation adalah sebagai berikut :
   - Kata yang digunakan adalah hasil dari case folding
   - Jika ditemukan lirik yang mengandung kata – kata negasi maka akan disatukan kata negasi tersebut dengan kata setelah kata negasi tersebut
3. Tokenizing
   Tokenizing adalah tahap untuk memisahkan atau memecah teks menjadi bagian–bagian kata yang disebut token.
3. Stopwords Removing
   Stopwords removing adalah tahap untuk mengambil kata yang dianggap penting dari hasil tokenizing atau membuang kata yang dianggap tidak memiliki arti penting dalam proses text preprocessing
4. Stemming Stemming merupakan tahap untuk mentransformasi kata-kata yang terdapat dalam suatu dokumen ke kata-kata akarnya (root word) dengan menggunakan aturan-aturan tertentu.

Step latihan 1:

a. Buka anaconda, lalu pilih jupyter notebook dan buatlah folder yangs udah diisi data isear dan buatlah file python pada folder tersebut.
b. Pada latihan ini akan menggunakan data teks bahasa manusia, maka diperlukan untuk menginstal library NLTK. NLTK (https://www.nltk.org) adalah library yang digunakan untuk pengolahan data bahasa manusia. Cara install nltk dapat dilakukan di command prompt atau secara langsung di dokumen program. . Ikuti kode program berikut dan run

```
In [1]: !pip install nltk
```

```
Unable to create process using 'C:\Users\Rin Rin Nurmalasari\anaconda3\python.exe "C:\Users\Rin Rin Nurmalasari\anaconda3\Scripts\pip-script.py" install nltk'
```

```
In [2]: import nltk
```

```
In [3]: nltk.download('punkt')
```

```
[nltk_data] Downloading package punkt to C:\Users\Rin Rin
[nltk_data]     Nurmalasari\AppData\Roaming\nltk_data...
[nltk_data]   Unzipping tokenizers\punkt.zip.
```

```
Out[3]: True
```

c. Load library yang akan digunakan: (pada python harus melakukan import library)

```
In [4]: import pandas as pd
import numpy as np
from nltk.corpus import stopwords
from nltk import ngrams
from nltk import word_tokenize
import string
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn import svm
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
from nltk.stem import PorterStemmer
from sklearn import metrics
```

d. Load data ISEAR.csv. Isi sesuai lokasi folder. Jika file berada pada folder yang sama dengan lokasi program python maka dapat ditulis df = pd.read_csv("ISEAR.csv", header=None). Df.head() berfungsi menampilkan 5 data teratas

```
In [5]: df = pd.read_csv("E:/FILE KULIAH S2 TMDG ITB/SISTEM INTELIJEN/MACHINE LEARNING/TUGAS 4 PAPER/ISEAR.csv",header=None)

df.head()
```

Out[5]:

	0	1	2
0	joy	On days when I feel close to my partner and ot...	NaN
1	fear	Every time I imagine that someone I love or I ...	NaN
2	anger	When I had been obviously unjustly treated and...	NaN
3	sadness	When I think about the short time that we live...	NaN
4	disgust	At a gathering I found myself involuntarily si...	NaN

e. Lakukan pembersihan pada data dengan menghapus null values pada data dan menambahkan nama kolom pada data

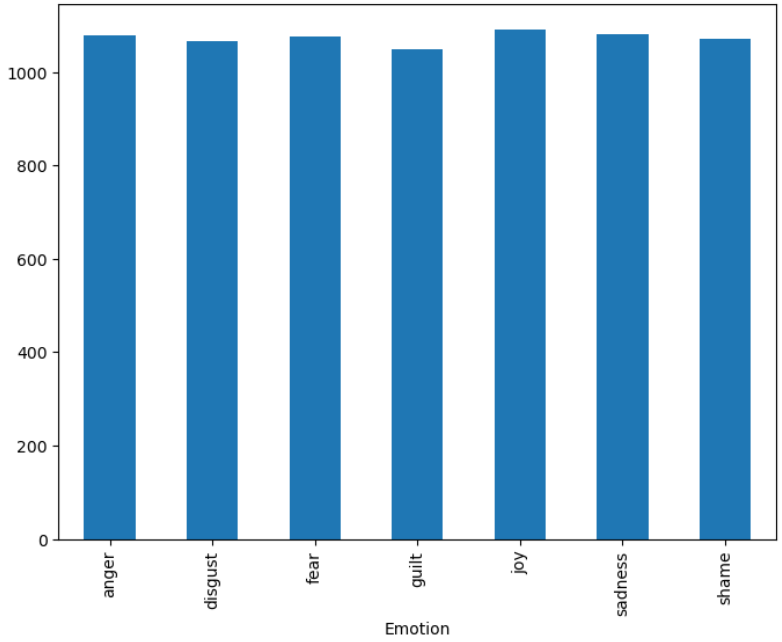
```
col = [0,1]
new_df = df[col]
new_df = new_df[pd.notnull(df[1])]
new_df.columns = ['Emotion','Text']
new_df.head()
```

Out[6]:

	Emotion	Text
0	joy	On days when I feel close to my partner and ot...
1	fear	Every time I imagine that someone I love or I ...
2	anger	When I had been obviously unjustly treated and...
3	sadness	When I think about the short time that we live...
4	disgust	At a gathering I found myself involuntarily si...

f. Melakukan pengecekan kelas emosi pada data seimbang atau tidak. Visualisasi menggunakan library matplotlib. Terlihat data cukup seimbang jadi dapat langsung dilakukan pada langkah selanjutnya. Dapat dilihat juga pada data ISEAR ini terdapat 7 emosi yaitu anger, disgust, fear, guilt, joy, sadness, shame.

```
In [7]: import matplotlib.pyplot as plt
        fig = plt.figure(figsize=(8,6))
        new_df.groupby('Emotion').Text.count().plot.bar(ylim=0)
        plt.show()
```



g. Lakukan preprocessing pada data text dengan melakukan Tokenisasi, menghapus baris baru, tanda baca.

```
new_df['Text']
new_df['Emotion']
```

```
def getTokenizedList(input_df):
    tokenizedList=[]
    for i in range(0,len(input_df)):
        curText=input_df[1].iloc[i]
        curText=curText.replace('\n','')
        curTokenized=word_tokenize(curText)
        tokenizedList.append(curTokenized)
    #print(tokenizedList)
    tokenziedListWithoutPunct=[]
    punctList=list(string.punctuation)
    for i in range(0,len(tokenizedList)):
        curList=tokenizedList[i]
        newList=[] #List without stopwords
        for word in curList:
            if (word.lower() not in punctList):
                newList.append(word.lower())
        tokenziedListWithoutPunct.append(newList)

    #print(tokenziedListWithoutPunct)
#Stemming
    mystemmer=PorterStemmer()
    tokenziedStemmed=[]
    for i in range(0,len(tokenziedListWithoutPunct)):
        curList=tokenziedListWithoutPunct[i]
        newList=[]
        for word in curList:
            newList.append(mystemmer.stem(word))
        tokenziedStemmed.append(newList)
    return tokenziedStemmed
```

```
def transformSentence(sent):
    s = []
    sent=sent.replace('\n','')
    sentTokenized=word_tokenize(sent)
    s.append(sentTokenized)
    sWithoutPunct = []
    punctList = list(string.punctuation)
    curSentList = s[0]
    newSentList = []
    for word in curSentList:
        if (word.lower() not in punctList):
            newSentList.append(word.lower())
    sWithoutPunct.append(newSentList)
    mystemmer = PorterStemmer()
    tokenziedStemmed = []
    for i in range(0,len(sWithoutPunct)):
        curList=sWithoutPunct[i]
        newList=[]
        for word in curList:
            newList.append(mystemmer.stem(word))
        tokenziedStemmed.append(newList)
    return tokenziedStemmed
```

h. Lalu lakukan Splitting data untuk train dan test. Proporsi data training 70 persen dan data test 30%. Lalu membuat fungsi untuk meneruskan list ke vektorizer Tfidf.

In [11]:
```
new_df['Text']=getTokenizedList(df)
#new_df['Text']
```

In [12]:
```
X_train, X_test, Y_train, Y_test=train_test_split(new_df['Text'],new_df['Emotion'], test_size=.3,random_state=1)
```

In [13]:
```
#Function to pass the list to the Tfidf vectorizer
def returnPhrase(inputList):
    return inputList
```

i. Selanjutnya ekstrasi fitur dengan naïve bayes . tulis kode program berikut:
#Extracting features for Naive Bayes

myVectorizer=TfidfVectorizer(analyzer='word',tokenizer=returnPhrase,preprocessor=retu
rnPhrase,token_pattern=None,ngram_range=(1,3))
myVectorizer.fit(X_train)
transformedTrain=myVectorizer.transform(X_train).toarray()
transformedTest=myVectorizer.transform(X_test).toarray()

j. Melakukan training menggunakan naïve bayes dan menampilkan metric clasification meliputi accuracy, precision, recall, f-1 score

```
In [15]: curAlpha=0.33 #smoothing factor in NB
         NBClassifier=MultinomialNB(alpha=curAlpha)
         NBClassifier.fit(transformedTrain,Y_train)
         myPredTest=NBClassifier.predict(transformedTest)
         print('Best Acc Naive Bayes')
         #print(curAlpha)
         print(np.sum(myPredTest==Y_test)/len(Y_test))

         Best Acc Naive Bayes
         0.5835920177383592
```

```
In [16]: print('Metrics Classification Report : Naive Bayes')
         print(metrics.classification_report(Y_test, myPredTest))

         Metrics Classification Report : Naive Bayes
                       precision    recall  f1-score   support

              anger       0.51      0.47      0.49       331
            disgust       0.72      0.50      0.59       324
               fear       0.62      0.70      0.66       311
              guilt       0.43      0.62      0.51       299
                joy       0.69      0.69      0.69       323
            sadness       0.68      0.60      0.64       351
              shame       0.52      0.51      0.52       316

           accuracy                           0.58      2255
          macro avg       0.60      0.58      0.58      2255
       weighted avg       0.60      0.58      0.59      2255
```

k. Selanjutnya lakukan Ekstrasi fitur untuk SVM dengan kode program berikut:
myVectorizer=TfidfVectorizer(analyzer='word',tokenizer=returnPhrase,preprocessor=retur
nPhrase,token_pattern=None,ngram_range=(1,3))
myVectorizer.fit(X_train)
transformedTrain=myVectorizer.transform(X_train).toarray()
transformedTest=myVectorizer.transform(X_test).toarray()

l. Melakukan training menggunakan SVM dan menampilkan metric clasification meliputi accuracy, precision, recall, f-1 score. Hasil akurasi SVM 61% dinilai cukup rendah.

In [18]:
```
curC=2   #cost factor in SVM
SVMClassifier=svm.LinearSVC(C=curC)
SVMClassifier.fit(transformedTrain,Y_train)
myPredTest=SVMClassifier.predict(transformedTest)
print('Best Acc SVM')
#print(curC)
print(np.sum(myPredTest==Y_test)/len(Y_test))
```

```
C:\Users\Rin Rin Nurmalasari\anaconda3\Lib\site-packages\sklearn\svm\_classes.py:32: FutureWarning: The default value of `dual` will change from `True` to `'auto'` in 1.5. Set the value of `dual` explicitly to suppress the warning.
  warnings.warn(
```

```
Best Acc SVM
0.6124168514412417
```

In [19]:
```
print('Metrics Classification Report : SVM')
print(metrics.classification_report(Y_test, myPredTest))
```

```
Metrics Classification Report : SVM
              precision    recall  f1-score   support

       anger       0.53      0.55      0.54       331
     disgust       0.67      0.60      0.63       324
        fear       0.65      0.72      0.69       311
       guilt       0.50      0.56      0.53       299
         joy       0.70      0.71      0.70       323
     sadness       0.67      0.65      0.66       351
       shame       0.56      0.50      0.53       316

    accuracy                           0.61      2255
   macro avg       0.61      0.61      0.61      2255
weighted avg       0.61      0.61      0.61      2255
```

J. Membuat fungsi untuk prediksi naïve bayes dan SVM. Serta menampilkan hasil prediksi. Isi kalimat variabel sent dapat diganti dengan kalimat apapun.

In [20]:
```
#To predict the emotion of a sentence using Naive Bayes
def predictSentNB(sent):
    sentPred = NBClassifier.predict(myVectorizer.transform(transformSentence(sent)).toarray())
    return sentPred
```

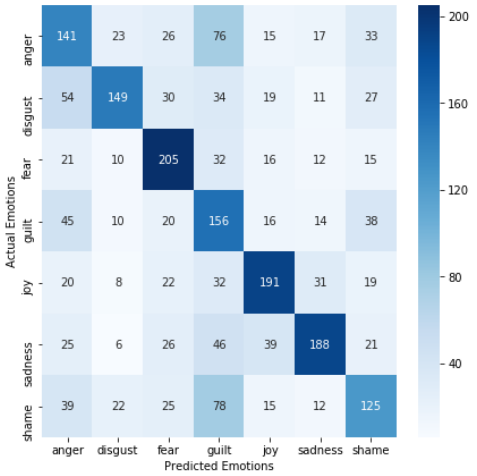
In [21]:
```
#To predict the emotion of a sentence using SVM
def predictSentSVM(sent):
    sentPred = SVMClassifier.predict(myVectorizer.transform(transformSentence(sent)).toarray())
    return sentPred
```

In [23]:
```
sent = "I love you to the moon and back"

#Printing the predicted emotion
print("Navie bayes prediction")
print(predictSentNB(sent))
print("SVM prediction")
print(predictSentSVM(sent))
```

```
Navie bayes prediction
['joy']
SVM prediction
['joy']
```

k. Menampilkan confusion matrix

```
In [48]: #printing the Confusion Matrix

from sklearn.metrics import confusion_matrix
import seaborn as sns
labels = ['anger', 'disgust','fear','guilt','joy','sadness','shame']
cm = confusion_matrix(Y_test, myPredTest,labels)
#print(cm)
fig, ax = plt.subplots(figsize=(7,7))
sns.heatmap(cm, annot=True,fmt='d',
            xticklabels=labels, yticklabels=labels,cmap='Blues')
plt.ylabel('Actual Emotions')
plt.xlabel('Predicted Emotions')
plt.show()
```



Soal Latihan: **Silahkan kalian perbaiki lagi dari preprocessing maupun aspek lainnya ataupun mencoba dengan metode baru untuk meningkatkan akurasi!**

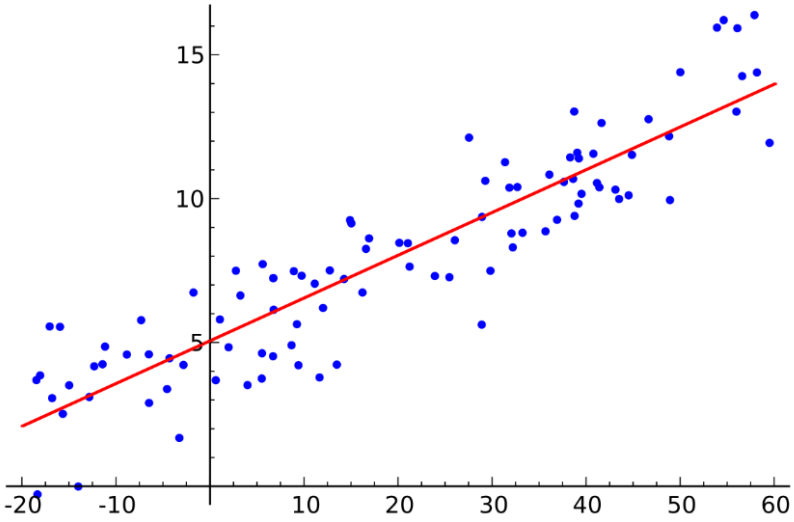
LATIHAN 2: SIMPLE LINIER REGRESI

kita akan mempelajari tentang simple linear regression. Linear Regression atau Regresi Linier adalah salah satu metode supervised learning dalam machine learning. Berbeda dengan metode klasifikasi yang memprediksi kelas atau kategori sebuah item, regresi linier berfungsi untuk memprediksi sebuah nilai berdasarkan atribut yang tersedia. Yang diprediksi adalah sejauh mana hubungan sebab akibat antara Variabel Faktor Penyebab (x) terhadap variabel akibat (y).

Apabila variabel bebasnya hanya satu, maka analisis regresinya menggunakan regresi linier sederhana. Apabila variabel bebasnya lebih dari satu, maka analisis regresinya menggunakan regresi linier berganda.

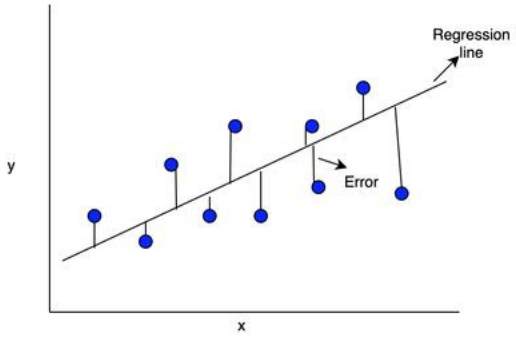
Contoh penggunaan analisis regresi linier sederhana antara lain:

- Hubungan antara jumlah pekerja dengan jumlah produksi
- Hubungan antara lama waktu bekerja dengan jabatan yang dimiliki saat ini
- Hubungan antara lama waktu PDKT dengan keberhasilan mendapat jodoh



Secara sederhana regresi linier adalah teknik untuk memprediksi sebuah nilai dari variable Y (variabel dependen) berdasarkan beberapa variabel tertentu X (variabel independen) jika terdapat hubungan linier antara X dan Y.

Dapat kita lihat juga bahwa garis regresi yang kita buat tidak tepat mengenai semua titik-titik variabelnya. Jarak dari garis regresi ke titik-titik variabel ini lah yang akan menjadi nilai error dari permodelan regresi kita.



Cobalah latihan berikut dengan menggunakan data sembarang:

Bermain dengan Data sembarang

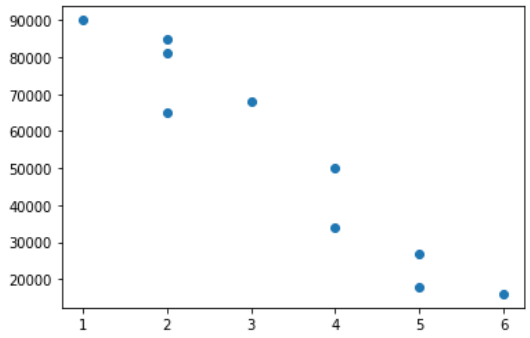
Untuk memberi gambaran mengenai regresi linier sederhana, kita coba buat suatu data buatan dan mencari hubungan antara keduanya

```
#import library dan package yang dibutuhkan

import numpy as np #untuk perhitungan saintifik
import matplotlib.pyplot as plt #untuk plotting
from sklearn.linear_model import LinearRegression #import library LinearRegression dari scikit-learn
```

```
#buat data

penjualan = np.array([6,5,5,4,4,3,2,2,2,1])
harga = np.array([16000, 18000, 27000, 34000, 50000, 68000, 65000, 81000, 85000, 90000])
```

```
#buat plot
%matplotlib inline
plt.scatter (penjualan, harga)
```

```
<matplotlib.collections.PathCollection at 0x7feff7299d68>
```

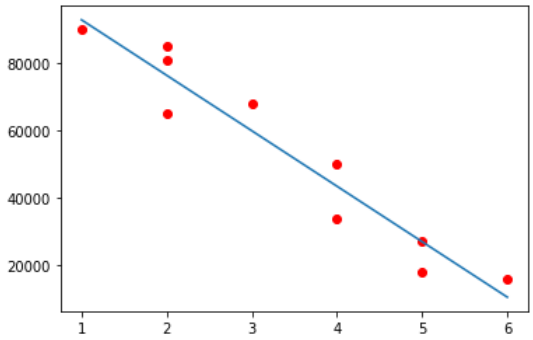



```
#buat permodelan regresi

penjualan = penjualan.reshape (-1,1) #kita tukar baris dan kolom variabel ini, agar bisa dikalikan dalam operasi matriks
#untuk lebih lengkapnya baca teori soal perhitungan regresi linier

linreg = LinearRegression()
linreg.fit(penjualan, harga)
```

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

```
#plot hasil regresi

plt.scatter(penjualan, harga, color='red')
plt.plot(penjualan, linreg.predict(penjualan))
```

```
[<matplotlib.lines.Line2D at 0x7feff72b32b0>]
```



Kita dapat melihat hubungan antara kedua variabel yang kita buat. Dalam hal ini saya membuat hubungan antara jumlah penjualan dengan harga barang. Dapat dilihat, semakin murah harga barang, maka jumlah penjualan naik.

LATIHAN LINEAR REGRESI DENGAN DATASET

kita akan belajar cara menggunakan library scikit-learn pada python untuk membuat permodelan regresi linier sederhana. Data yang akan kita gunakan adalah dataset konsumsi bahan bakar pada mobil yang kita dapat dari **IBM Object Storage.** Telah disediakan datasetnya **FuelConsumptionCo2.csv.**

Dataset ini berisi data konsumsi bahan bakal dan estimasi emisi karbon dioksida pada beberapa model kendaraan yang dijual di Canada. Dataset berisi data:

- MODELYEAR
- MAKE
- MODEL
- VEHICLE CLASS
- ENGINE SIZE
- CYLINDERS
- TRANSMISSION
- FUEL CONSUMPTION in CITY(L/100 km)
- FUEL CONSUMPTION in HWY (L/100 km)
- FUEL CONSUMPTION COMB (L/100 km)
- CO2 EMISSIONS (g/km)

Langkah pertama, yaitu load library yang dibutuhkan. Matplotlib, pandas dsb. Lalu load data yang fuel consumtion.

```
df = pd.read_csv("FuelConsumption.csv") #membaca data

# melihat 5 baris pertama data
df.head()
```

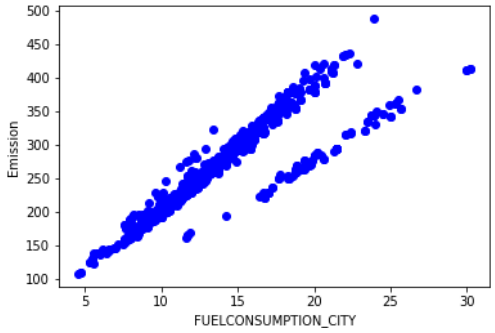
	MODELYEAR	MAKE	MODEL	VEHICLECLASS	ENGINESIZE	CYLINDERS	TRANSMISSION	FUELTYPE	FUELCONSUMPTION_CITY	FU
0	2014	ACURA	ILX	COMPACT	2.0	4	AS5	Z	9.9	
1	2014	ACURA	ILX	COMPACT	2.4	4	M6	Z	11.2	
2	2014	ACURA	ILX HYBRID	COMPACT	1.5	4	AV7	Z	6.0	
3	2014	ACURA	MDX 4WD	SUV - SMALL	3.5	6	AS6	Z	12.7	
4	2014	ACURA	RDX AWD	SUV - SMALL	3.5	6	AS6	Z	12.1	

```
#kita ambil kolom mana saja yang akan kita analisis, dan membuang sisanya

cdf = df[['ENGINESIZE','CYLINDERS','FUELCONSUMPTION_CITY','CO2EMISSIONS']]
cdf.head(9)
```

	ENGINESIZE	CYLINDERS	FUELCONSUMPTION_CITY	CO2EMISSIONS
0	2.0	4	9.9	196
1	2.4	4	11.2	221
2	1.5	4	6.0	136
3	3.5	6	12.7	255
4	3.5	6	12.1	244
5	3.5	6	11.9	230
6	3.5	6	11.8	232
7	3.7	6	12.8	255
8	3.7	6	13.4	267

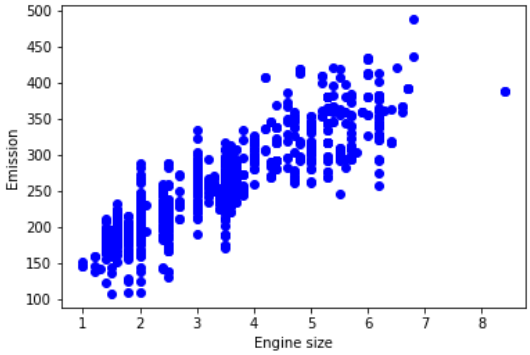
Sebagai contoh, kita lihat hubungan antara variabel konsumsi bahan bakar di kota dengan emisi CO2.

```
#Kita plot hubungannya

plt.scatter(cdf.FUELCONSUMPTION_CITY, cdf.CO2EMISSIONS,  color='blue')
plt.xlabel("FUELCONSUMPTION_CITY")
plt.ylabel("Emission")
plt.show()
```



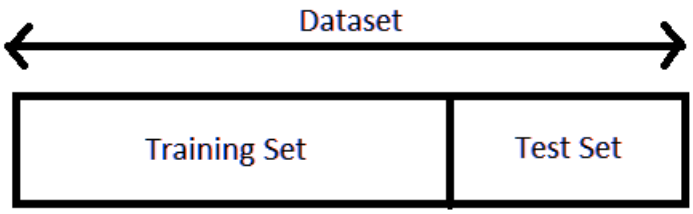
Sebagai contoh juga, kita lihat hubungan antara variabel ukuran mesin dengan emisi CO2.

```
#Kita plot hubungannya

plt.scatter(cdf.ENGINESIZE, cdf.CO2EMISSIONS,  color='blue')
plt.xlabel("Engine size")
plt.ylabel("Emission")
plt.show()
```



Melakukan Data Splitting

Dalam membuat permodelan Machine Learning, merupakan suatu hal yang lazim untuk membagi data ke dalam data latih (train) dan data uji (test). Kita gunakan data latih untuk melatih model, dan data uji untuk menilai performa dari permodelan yang kita buat. Hal ini kita lakukan untuk mensimulasikan, apakah model kita dapat bekerja dengan baik dalam menganalisa kasus yang belum pernah ia temui sebelumnya

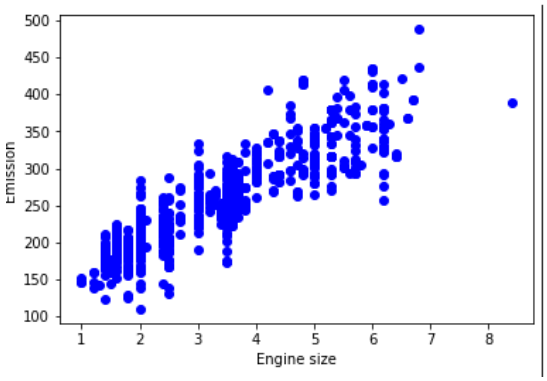


Biasanya data latih akan berjumlah sekitar 70-80% dari semua data, dan sisanya adalah data test.

```
#Membagi data

msk = np.random.rand(len(df)) < 0.8
train = cdf[msk]
test = cdf[~msk]
```

Di sini kita akan mencari hubungan antara ukuran mesin dengan emisi CO2 pada kendaraan

```
plt.scatter(train.ENGINESIZE, train.CO2EMISSIONS,  color='blue')
plt.xlabel("Engine size")
plt.ylabel("Emission")
plt.show()
```



Membuat Model Regresi

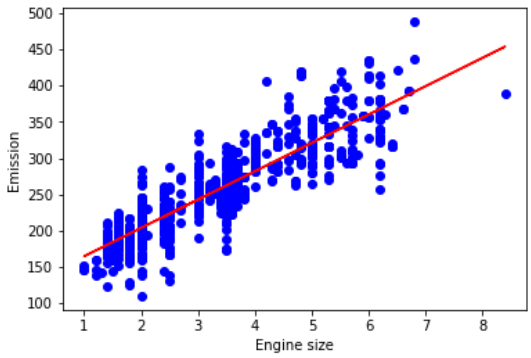
```
#Membuat model regresi
regr = LinearRegression()
train_x = np.asanyarray(train[['ENGINESIZE']])
train_y = np.asanyarray(train[['CO2EMISSIONS']])
regr.fit (train_x, train_y)

# Koefisien model
print ('Coefficients: ', regr.coef_)
print ('Intercept: ',regr.intercept_)
```

```
Coefficients:  [[39.09332068]]
Intercept:  [125.62023245]
```

Dalam membuat model regresi, sebenarnya kita mencari nilai-nilai *Coefficient* (gradien/kemiringan) dan *Intercept* yang kita sebut parameter model. Untuk memahami hal ini, disarankan untuk mempelajari persamaan garis dan teori dasar regresi linier pada tautan yang sudah diberikan.

```
#Plot hasil regresi

plt.scatter(train.ENGINESIZE, train.CO2EMISSIONS,  color='blue')
plt.plot(train_x, regr.coef_[0][0]*train_x + regr.intercept_[0], '-r')
plt.xlabel("Engine size")
plt.ylabel("Emission")
```

```
Text(0, 0.5, 'Emission')
```




```
#Menghitung error

from sklearn.metrics import r2_score

test_x = np.asanyarray(test[['ENGINESIZE']])
test_y = np.asanyarray(test[['CO2EMISSIONS']])
test_y_ = regr.predict(test_x)

print("Mean absolute error: %.2f" % np.mean(np.absolute(test_y_ - test_y)))
print("Residual sum of squares (MSE): %.2f" % np.mean((test_y_ - test_y) ** 2))
print("R2-score: %.2f" % r2_score(test_y_ , test_y) )
```

```
Mean absolute error: 27.50
Residual sum of squares (MSE): 1253.08
R2-score: 0.61
```

Dapat kita lihat, bahwa error yang dihasilkan model regresi yang kita buat cukup besar. Hal ini dikarenakan masih banyak faktor lain yang mempengaruhi emisi CO2 pada kasus di atas. Permodelan regresi linier sederhan masih tergolong metode *machine learning* yang masih dasar. Oleh karena itu, kita perlu menggunakan metode *machine learning* yang lebih *powerful* lagi.

LATIHAN 3: STUDI KASUS DEPLOYMENT MODEL MACHINE LEARNING PADA WEB UNTUK PREDIKSI KEUNTUNGAN START-UP BERDASARKAN PENGELUARAN MENGGUNAKAN FLASK

Pada tutorial kali ini akan melakukan deployment model machine learning kedalam website dengan studi kasus memprediksi keuntungan/profit yang didapatkan start-up berdasarkan pengeluaran yang digelontorkan di **Research and Development, Administration, dan Marketing**.

Data yang digunakan dapat dilihat di (sample data.csv) yang berisi 50 baris observasi. Data yang ada akan dilatih menggunakan metode regresi linear dari pustaka

Python **Scikit-Learn** yang mana hasil dari pelatihan model ini akan diekspor dan diimplementasikan kedalam website menggunakan pustaka Python **Flask**.

Persiapan Tool latihan 4:

1. Browser
2. Visual Studio Code/IDE lainnya
3. Python yang terinstall di local machine
4. Command Prompt

PEMBUATAN MODEL MACHINE LEARNING

Tujuan bagian ini untuk menyiapkan model yang akan digunakan dalam web. Pada latihan ini model ML sudah dibuat. Kunjungi https://colab.research.google.com/drive/1rS0QpD-7EeOCA1IdV1MAChshnFxdOTh-?usp=sharing untuk melihat keseluruhan isi kode dan dicoba.

	R&D Spend	Administration	Marketing Spend	State	Profit
0	165349.20	136897.80	471784.10	New York	192261.83
1	162597.70	151377.59	443898.53	California	191792.06
2	153441.51	101145.55	407934.54	Florida	191050.39
3	144372.41	118671.85	383199.62	New York	182901.99
4	142107.34	91391.77	366168.42	Florida	166187.94

Pengenalan Data

Berisi lima kolom dengan lima puluh baris observasi dengan empat kolom numerikal dan satu kolom kategorikal. Data ini akan diolah sederhana agar dapat memprediksi hasil yang tertera di kolom profit.

Pemrosesan Data

```
array([[165349.2 , 136897.8 , 471784.1 ],
       [162597.7 , 151377.59, 443898.53],
       [153441.51, 101145.55, 407934.54],
       [144372.41, 118671.85, 383199.62],
       [142107.34,  91391.77, 366168.42],
       [131876.9 ,  99814.71, 362861.36],
```

Data latih hanya berisi informasi pengeluaran di **Research & Development Spend, Administration, dan Marketing Spend** yang sudah dikonversi dalam bentuk numpy array.

Pelatihan Model Machine Learning

```
regression = LinearRegression()
regression.fit(X_train, y_train)
```

```
LinearRegression()
```

```
y_pred = regression.predict(X_test)

mean_absolute_error(y_test, y_pred)
```

Data dilatih menggunakan regresi linear dan menghasilkan model machine learning yang siap untuk diekspor.

Ekspor Model

```
joblib.dump(regression, "hasil_pelatihan_model.pkl")
```

```
['hasil_pelatihan_model.pkl']
```

Model diekspor kedalam bentuk pickle (.pkl) yang mana model ini akan di load di program Flask agar dapat berjalan di website. Selain pickle, model juga dapat diekspor ke bentuk lain seperti .h5, .hdf, ataupun .pt.

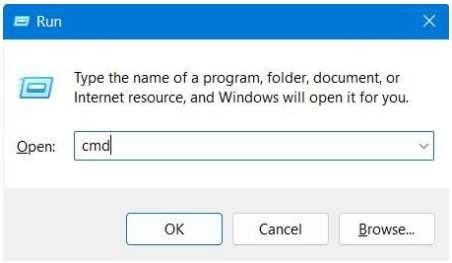
IMPLEMENTASI DAN PEMBUATAN WEBSITE MENGGUNAKAN FLASK

Sekilas Tentang Flask

Flask adalah framework juga library yang ada di bahasa pemrograman Python dengan salah satu fungsinya untuk mengimplementasikan model machine learning kedalam website. Flask memiliki sintaks yang simpel dan tidak rumit, namun memiliki struktur folder yang terorganisir untuk memisahkan file-file berdasarkan fungsionalitasnya.

Instalasi dan Setup Flask

1. Buka Command Prompt



2. Buat direktori baru untuk projek

```
mkdir tutorialFlask
cd tutorialFlask
```

3. Buat dan aktifkan virtual environment

```
// untuk windows
py -3 -m venv .venv
.venv\scripts\activate
// untuk linux
python3 -m venv .venv
source .venv/bin/activate
```

4. Masuk ke Visual Studio Code

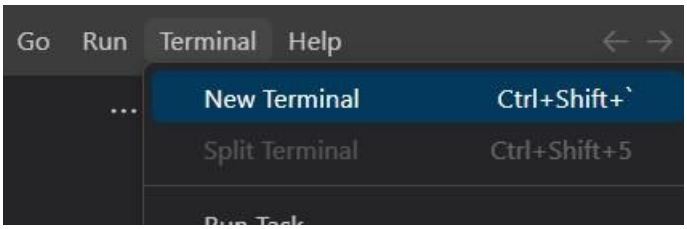
```
code .
```

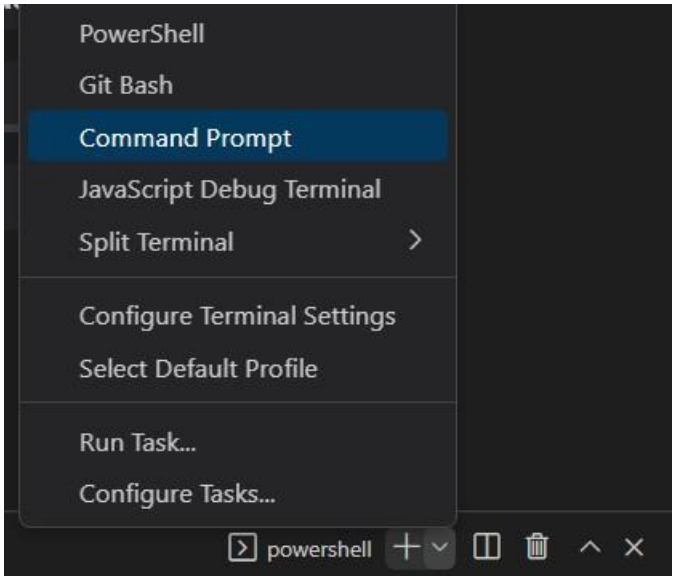
Atau buka visual studio code dan buka folder tutorialFlask

Berikut tampilan keseluruhan command dari step 2 sampai 4

```
C:\Users\hp>mkdir tutorialFlask

C:\Users\hp>cd tutorialFlask

C:\Users\hp\tutorialFlask>py -3 -m venv .venv

C:\Users\hp\tutorialFlask>.venv\scripts\activate

(.venv) C:\Users\hp\tutorialFlask>code .

(.venv) C:\Users\hp\tutorialFlask>
```

5. Buat terminal baru dan pindahkan ke command prompt atau git bash





```
(.venv) C:\Users\hp\tutorialFlask>
```

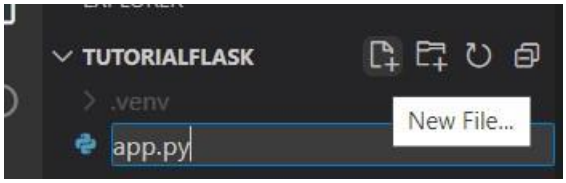
6. Install Flask di terminal Visual Studio Code

```
python -m pip install --upgrade pip
python -m pip install flask
```

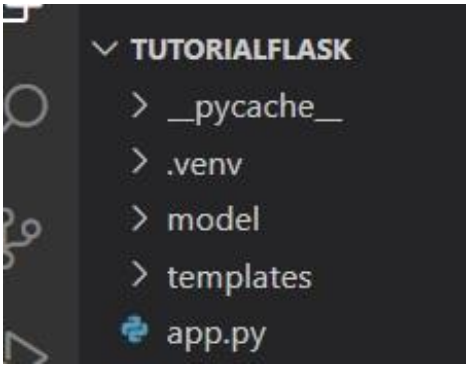
7. Buat direktori baru untuk template dan model

```
mkdir template
mkdir model
```

8. Buat file app.py



Pemahaman Folder Direktori Flask



1. .venv

   .venv atau virtual environment ini berisikan local dependencies yang terinstall untuk project, atau dalam bahasa lain virtual environment ini merupakan kumpulan resource yang dibutuhkan agar projek dapat berjalan semestinya. Semua package/library yang terinstall akan berada di folder .venv
2. model model disini untuk menyimpan file machine learning yang sudah diekspor kedalam ekstensi pickle (.pkl), maka dari itu pindahkan file pickle di notebook atau download disini dan pindahkan file tersebut di folder model.
3. templates

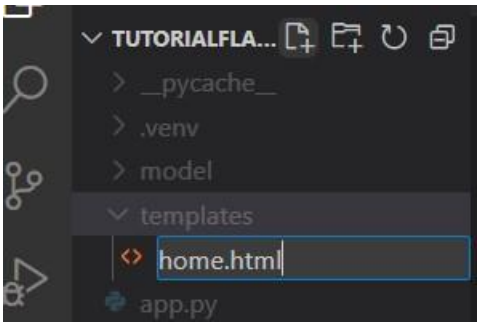
   Folder template akan digunakan untuk menyimpan file HTML yang akan dibuat.
4. static

   Folder static digunakan untuk menyimpan file css. Javascript, maupun gambar yang dipakai dalam website.
5. app.py

   File app.py digunakan untuk mengatur keseluruhan project flask di sisi backend

Pembuatan Tampilan Antarmuka

1. Buat file home.html di folder templates

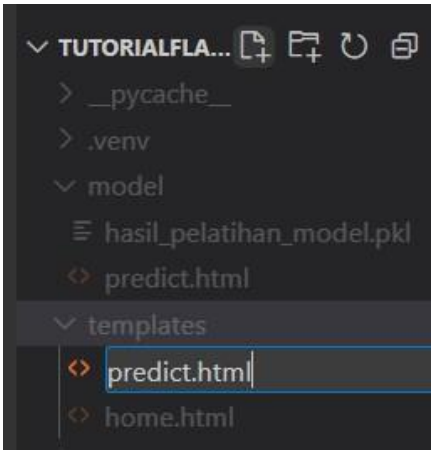


2. Masukkan kode berikut di home.html

```
<!DOCTYPE html>
<html>
<head>
  <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstr
ap.min.css" integrity="sha384-
BVYiiSIFeK1dGmJRAkycuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u"
crossorigin="anonymous">
  <title>Profit Prediction</title>
  </head>

  <body>
    <h1><div style="text-align:center"><font color="blue">Prediksi
profit dari pengeluaran startup</font></div></h1>
    <hr>
    </br>
```

```
    <h2><div  style="text-align:center">Masukkan  nilai  untuk  R&D
Spend, Administration Spend, dan Marketing Spend</div></h2>
    </br>
    <div class='container'>
      <div class='row'>
        <div class='col-6'>
          <form method="POST" action="/predict">
            <div class="form-group">
              <label    for="RnD_Spend"><p    class="font-weight-
bold">RnD_Spend</p></label>
              <input type="text" name="RnD_Spend">
              <small   id="RnDSpendHelp"   class="form-text   text-
muted">Masukkan nilai RnD Spend
(dalam dollar)</small>
            </div>

            <div class="form-group">
              <label    for="Admin_Spend"><p    class="font-weight-
bold">Admin_Spend</p></label>
              <input type="text" name="Admin_Spend">
              <small  id="AdminSpendHelp"  class="form-text  text-
muted">Masukkan nilai
Administration Spend (dalam dollar)</small>
            </div>
            <div class="form-group">
              <label   for="Market_Spend"><p   class="font-weight-
bold">Market_Spend</p></label>
              <input type="text" name="Market_Spend">
              <small  id="MarketSpendHelp"  class="form-text  text-
muted">Masukkan nilai Marketing
Spend (dalam dollar)</small>
            </div>

            <input    class='btn    btn-primary'    type='submit'
value='Submit'>
          </form>
          </div>
          </div>
          </div>
  </body>
</html>
```

3. Buat file **predict.html** di folder templates



4. Masukkan kode berikut di predict.html

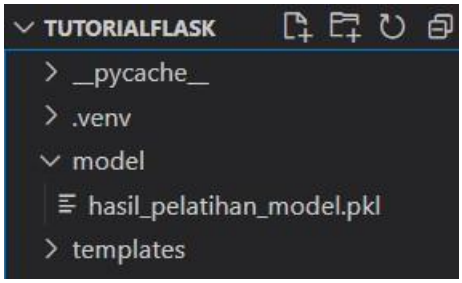
```
<!DOCTYPE html>
<html>
<head>
  <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css"
integrity="sha384
BVYiiSIFeK1dGmJRAkycuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u"
crossorigin="anonymous">
  <title>Final Result of Profit Prediction</title>
  </head>

  <body>
    <h1><div style="text-align:center"><font color="blue">Hasil Prediksi Profit</font></div></h1>
    <hr>
    <div class="card text-center" style="width:21.5em;margin:0 auto;">
      <div class="card-body">
        <p class="card-text"><h1><font color="blue">${{prediction}}</font></h1></p>
        </div>
        </div>
  </body>

</html>
```

Kode HTML yang telah dimasukan tadi akan menjadi kerangka web yang akan dibuat, dimana ada form di home.html yang akan diproses oleh flask lalu akan menampilkan hasil olahannya di predict.html.

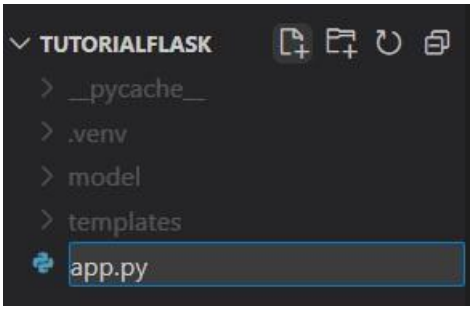
Integrasi Model Machine Learning

1. Load file pickle ke folder model

   File model yang telah diberikan hasil_pelatihan_model.pkl, pindahkan ke folder model



2. Buat file app.py di root folder



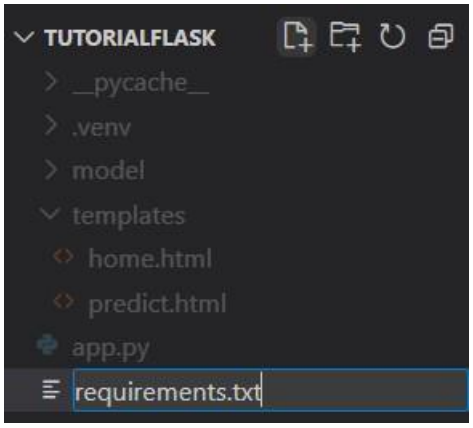
3. Masukkan kode berikut di app.py

```
from flask import Flask, render_template, request
import joblib
import pandas as pd
import numpy as np

app = Flask(__name__)

mul_reg = open("model/hasil_pelatihan_model.pkl", "rb")
ml_model = joblib.load(mul_reg)

@app.route("/")
def home():
    return render_template('home.html')

@app.route("/predict", methods=['GET', 'POST'])
def predict():
    print("Prediksi dimulai")
    if request.method == 'POST':
        print(request.form.get('RnD_Spend'))
        try:
            RnD_Spend =float(request.form['RnD_Spend'])
            Admin_Spend = float(request.form['Admin_Spend'])
            Market_Spend = float(request.form['Market_Spend'])
            pred_args = [RnD_Spend, Admin_Spend, Market_Spend]
            pred_args_arr = np.array(pred_args)
            pred_args_arr = pred_args_arr.reshape(1, -1)
            # mul_reg = open("multiple_regression_model.pkl", "rb")
            # ml_model = joblib.load(mul_reg)
            model_prediction = ml_model.predict(pred_args_arr)
            model_prediction = round(float(model_prediction), 2)
        except ValueError:
            return "Please check if the values are entered correctly"
    return render_template('predict.html', prediction=model_prediction)


if __name__ == "__main__":
    app.run(host='0.0.0.0')
```

Kode ini akan menginput form yang diisi, meload model machine learning, dan mengembalikan hasil prediksi yang ditampilkan dalam predict.html.

4. Install depedensi

   Kode yang sudah dibuat tidak akan berjalan karena library yang digunakan belum diinstal. Untuk menginstall, buat file requirements.txt di root folder.



Masukkan kode berikut

```
certifi
chardet
Click
Flask
idna
itsdangerous
Jinja2
MarkupSafe
numpy
pandas
python-dateutil
pytz
redis
requests
scikit-learn
scipy
six
urllib3
Werkzeug
wincertstore
```

Di terminal, jalankan command berikut untuk menginstall semua depedensi yang ditulis di requirements.txt

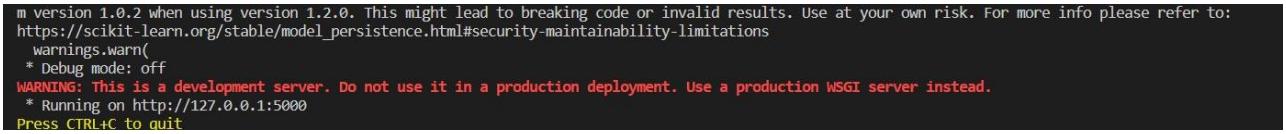
```
pip install-r requirements.txt
```

Menjalankan Project

Masukkan command berikut untuk menjalankan projek dan membukanya di browser.

```
flask run
```

Akan menghasilkan output sebagai berikut, copy paste url http://127.0.0.1:5000 pada web browser dan jalankan.



Berikut adalah tampilan dari **website**

Setelah data diisi dan disubmit, berikut adalah tampilan hasilnya.