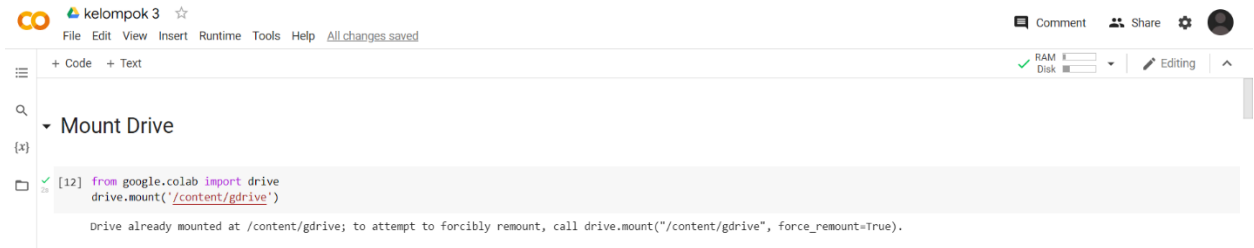


IMPLEMENTASI PROGRAM

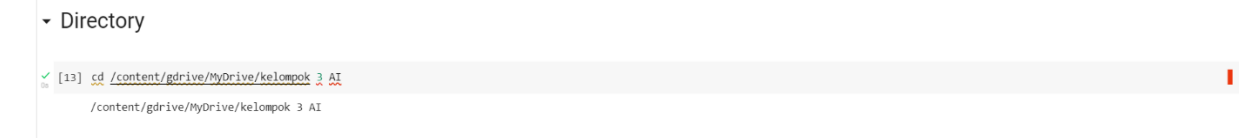
- Pertama-tama buka google colabratoty untuk masukkan script code
- Ketik “Mount Drive” untuk penamaan di google colaboratory. Lalu hubungkan dengan google drive menggunakan script code dibawah ini



```
[12] from google.colab import drive
drive.mount('/content/gdrive')

Drive already mounted at /content/gdrive; to attempt to forcibly remount, call drive.mount("/content/gdrive", force_remount=True).
```

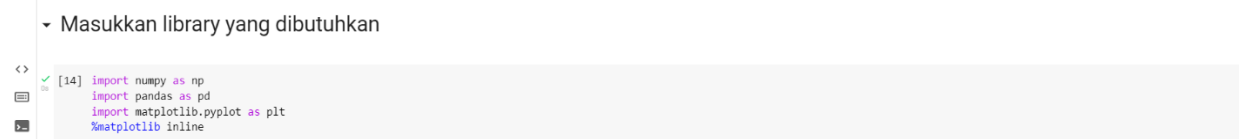
- Ketik “Directory” untuk penamaan di google colaboratory untuk memanggil jalur folder “kelompok 3 AI” yang telah dibuat di google drive menggunakan script code dibawah ini



```
[13] cd /content/gdrive/MyDrive/kelompok 3 AI

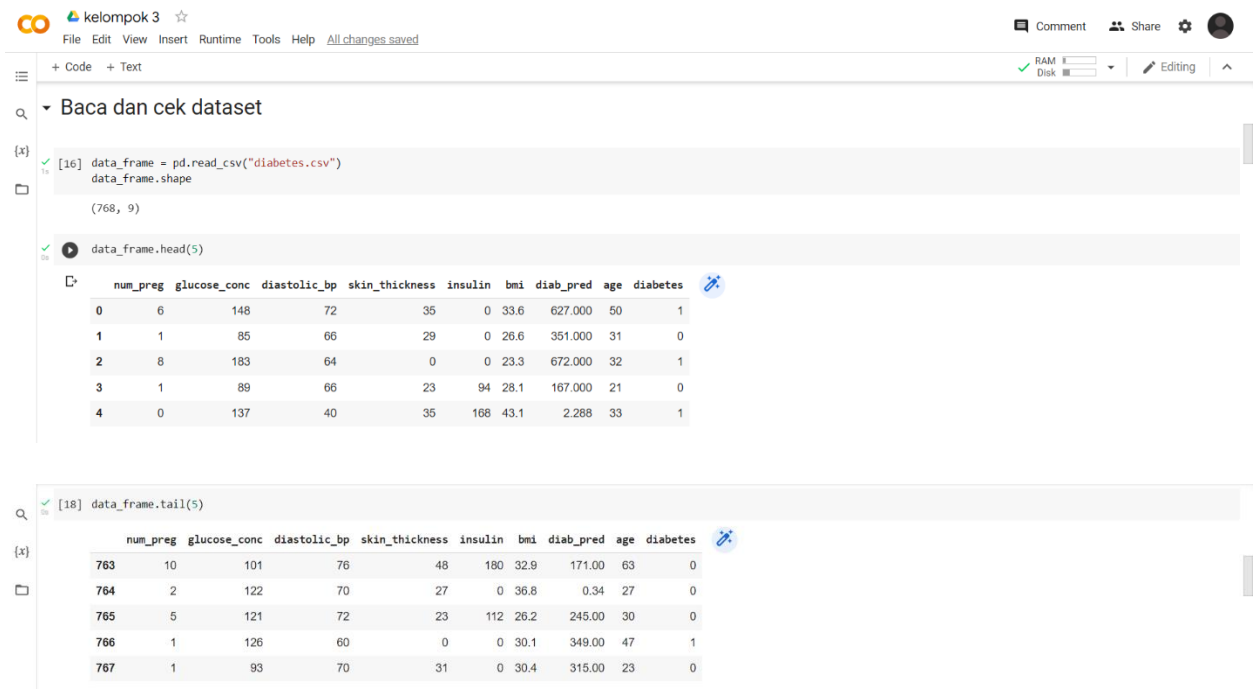
/content/gdrive/MyDrive/kelompok 3 AI
```

- Ketik “Masukkan library yang dibutuhkan” untuk package di google colaboratory



```
[14] import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
```

- Ketik data frame untuk membaca file csv atau dataset yang akan digunakan



```
[16] data_frame = pd.read_csv("diabetes.csv")
data_frame.shape

(768, 9)

data_frame.head(5)
```

	num_preg	glucose_conc	diastolic_bp	skin_thickness	insulin	bmi	diab_pred	age	diabetes
0	6	148	72	35	0	33.6	627.000	50	1
1	1	85	66	29	0	26.6	351.000	31	0
2	8	183	64	0	0	23.3	672.000	32	1
3	1	89	66	23	94	28.1	167.000	21	0
4	0	137	40	35	166	43.1	2.288	33	1

```
[18] data_frame.tail(5)
```

	num_preg	glucose_conc	diastolic_bp	skin_thickness	insulin	bmi	diab_pred	age	diabetes
763	10	101	76	48	180	32.9	171.00	63	0
764	2	122	70	27	0	36.8	0.34	27	0
765	5	121	72	23	112	26.2	245.00	30	0
766	1	126	60	0	0	30.1	349.00	47	1
767	1	93	70	31	0	30.4	315.00	23	0

- Ketik print untuk mencetak suatu fungsi yang dibutuhkan

```
[19] print (data_frame.isnull().values.any())
False
```

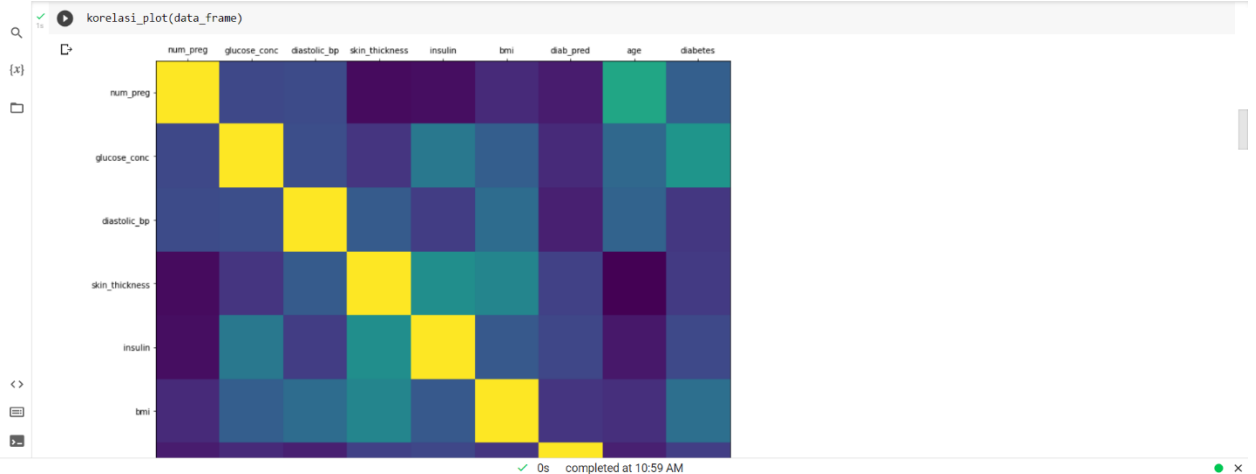
- Membuat fungsi class korelasi plot

```
[20] def korelasi_plot(data_frame):
    corr = data_frame.corr()
    fig, ax = plt.subplots(figsize=(13, 13))
    ax.matshow(corr)
    plt.xticks(range(len(corr.columns)), corr.columns)
    plt.yticks(range(len(corr.columns)), corr.columns)

[21] korelasi_plot(data_frame)
```

num_preg glucose_conc diastolic_bp skin_thickness insulin bmi diab_pred age diabetes
0s completed at 10:59 AM

- Menampilkan grafik data yang ada di dataset



data_frame.corr()

	num_preg	glucose_conc	diastolic_bp	skin_thickness	insulin	bmi	diab_pred	age	diabetes
num_preg	1.000000	0.129459	0.141282	-0.081672	-0.073535	0.017683	-0.027129	0.544341	0.221898
glucose_conc	0.129459	1.000000	0.152590	0.057328	0.331357	0.221071	0.019326	0.263514	0.466581
diastolic_bp	0.141282	0.152590	1.000000	0.207371	0.088933	0.281805	-0.014605	0.239528	0.065068
skin_thickness	-0.081672	0.057328	0.207371	1.000000	0.436783	0.392573	0.103279	-0.113970	0.074752
insulin	-0.073535	0.331357	0.088933	0.436783	1.000000	0.197859	0.122250	-0.042163	0.130548
bmi	0.017683	0.221071	0.281805	0.392573	0.197859	1.000000	0.055877	0.036242	0.292695
diab_pred	-0.027129	0.019326	-0.014605	0.103279	0.122250	0.055877	1.000000	-0.016418	0.092093
age	0.544341	0.263514	0.239528	-0.113970	-0.042163	0.036242	-0.016418	1.000000	0.238356
diabetes	0.221898	0.466581	0.065068	0.074752	0.130548	0.292695	0.092093	0.238356	1.000000

```
[23] data_frame.head(5)
```

	num_preg	glucose_conc	diastolic_bp	skin_thickness	insulin	bmi	diab_pred	age	diabetes
0	6	148	72	35	0	33.6	627.000	50	1
1	1	85	66	29	0	26.6	351.000	31	0
2	8	183	64	0	0	23.3	672.000	32	1
3	1	89	66	23	94	28.1	167.000	21	0
4	0	137	40	35	168	43.1	2.288	33	1

```
[24] obs = len(data_frame)
true = len(data_frame.loc[data_frame['diabetes'] == 1])
false = len(data_frame.loc[data_frame['diabetes'] == 0])
print("Persentase Orang Diabetes : {} ({:1.2f}%)".format(true, ((1.0 * true)/(1.0 * obs)) * 100))
print("Persentase Orang Tidak Diabetes: {} ({:1.2f}%)".format(false, ((1.0 * false)/(1.0 * obs)) * 100))

Persentase Orang Diabetes : 268 (34.90%)
Persentase Orang Tidak Diabetes: 500 (65.10%)

from sklearn.model_selection import train_test_split

nama_fitur = ['num_preg', 'glucose_conc', 'diastolic_bp', 'skin_thickness', 'insulin', 'bmi', 'diab_pred', 'age']
prediksi = ['diabetes']

X = data_frame[nama_fitur].values
y = data_frame[prediksi].values

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30, random_state=5)
```

```
[26] train_value = (1.0 * len(X_train)) / (1.0 * len(data_frame.index))
test_value = (1.0 * len(X_test)) / (1.0 * len(data_frame.index))
print("{:0.2f}% pada set training".format(train_value * 100))
print("{:0.2f}% pada set test".format(test_value * 100))

69.92% pada set training
30.08% pada set test

int("Original True : {} ({:1.0.2f}%)".format(len(data_frame.loc[data_frame['diabetes'] == 1]), (len(data_frame.loc[data_frame['diabetes'] == 1])/len(data_frame.index)) * 100.0))
int("Original False : {} ({:1.0.2f}%)".format(len(data_frame.loc[data_frame['diabetes'] == 0]), (len(data_frame.loc[data_frame['diabetes'] == 0])/len(data_frame.index)) * 100.0))
int("")
int("Training True : {} ({:1.0.2f}%)".format(len(y_train[y_train[:] == 1]), (len(y_train[y_train[:] == 1])/len(y_train)) * 100.0))
int("Training False : {} ({:1.0.2f}%)".format(len(y_train[y_train[:] == 0]), (len(y_train[y_train[:] == 0])/len(y_train)) * 100.0))
int("")
int("Test True : {} ({:1.0.2f}%)".format(len(y_test[y_test[:] == 1]), (len(y_test[y_test[:] == 1])/len(y_test)) * 100.0))
int("Test False : {} ({:1.0.2f}%)".format(len(y_test[y_test[:] == 0]), (len(y_test[y_test[:] == 0])/len(y_test)) * 100.0))

Original True : 268 (34.90%)
Original False : 500 (65.10%)

Training True : 197 (36.69%)
Training False : 340 (63.31%)

Test True : 71 (30.74%)
Test False : 160 (69.26%)

[28] data_frame.head(5)
```

	num_preg	glucose_conc	diastolic_bp	skin_thickness	insulin	bmi	diab_pred	age	diabetes
0	6	148	72	35	0	33.6	627.000	50	1
1	1	85	66	29	0	26.6	351.000	31	0
2	8	183	64	0	0	23.3	672.000	32	1
3	1	89	66	23	94	28.1	167.000	21	0
4	0	137	40	35	168	43.1	2.288	33	1

```
[29] print("Baris yang ada di dataframe {}".format(len(data_frame)))
print("Baris yang mempunyai nilai 0 di glucose_conc: {}".format(len(data_frame.loc[data_frame['glucose_conc'] == 0])))
print("Baris yang mempunyai nilai 0 di diastolic_bp: {}".format(len(data_frame.loc[data_frame['diastolic_bp'] == 0])))
print("Baris yang mempunyai nilai 0 di skin_thickness: {}".format(len(data_frame.loc[data_frame['skin_thickness'] == 0])))
print("Baris yang mempunyai nilai 0 di insulin: {}".format(len(data_frame.loc[data_frame['insulin'] == 0])))
print("Baris yang mempunyai nilai 0 di bmi: {}".format(len(data_frame.loc[data_frame['bmi'] == 0])))
print("Baris yang mempunyai nilai 0 di diab_pred: {}".format(len(data_frame.loc[data_frame['diab_pred'] == 0])))
print("Baris yang mempunyai nilai 0 di age: {}".format(len(data_frame.loc[data_frame['age'] == 0])))

Baris yang ada di dataframe 768
Baris yang mempunyai nilai 0 di glucose_conc: 5
Baris yang mempunyai nilai 0 di diastolic_bp: 35
Baris yang mempunyai nilai 0 di skin_thickness: 227
Baris yang mempunyai nilai 0 di insulin: 374
Baris yang mempunyai nilai 0 di bmi: 11
Baris yang mempunyai nilai 0 di diab_pred: 0
Baris yang mempunyai nilai 0 di age: 0
```

```
[30] from sklearn.impute import SimpleImputer

imputer = SimpleImputer(missing_values=np.nan, strategy='mean')

X_train = imputer.fit_transform(X_train)
X_test = imputer.fit_transform(X_test)

[31] from sklearn.svm import SVC
svm = SVC(kernel='linear', C=1, random_state=0)

svm.fit(X_train, y_train.ravel())

SVC(C=1, kernel='linear', random_state=0)

[32] # mengembalikan array ke hasil prediksi
prediksi_svm = svm.predict(X_train)

[33] from sklearn import metrics

akurasi_train = metrics.accuracy_score(y_train, prediksi_svm)

print ("Akurasi train : {:.4f}".format(akurasi_train))

Akurasi train : 0.7561

[34] prediksi_svm = svm.predict(X_test)

akurasi_test = metrics.accuracy_score(y_test, prediksi_svm)

print ("Akurasi tes : {:.4f}".format(akurasi_test))

Akurasi tes : 0.7922

[35] print ("Confusion Matrix untuk Support Vector Machine")
# Label di set ke 1=True untuk kiri atas dan 0=false untuk kanan bawah
print ("{}".format(metrics.confusion_matrix(y_test, prediksi_svm, labels=[1, 0])))

Confusion Matrix untuk Support Vector Machine
[[ 47  24]
 [ 24 136]]

[36] print ("Laporan Klasifikasi\n")
# Label di set ke 1=True untuk kiri atas dan 0=false untuk kanan bawah
print ("{}".format(metrics.classification_report(y_test, prediksi_svm, labels=[1, 0])))

Laporan Klasifikasi

              precision    recall  f1-score   support

         1       0.66       0.66       0.66         71
         0       0.85       0.85       0.85        160

 accuracy          0.79          0.79          0.79        231
 macro avg          0.76          0.76          0.76        231
 weighted avg          0.79          0.79          0.79        231

print ("Laporan Klasifikasi\n")
# Label di set ke 1=True untuk kiri atas dan 0=false untuk kanan bawah
print ("{}".format(metrics.classification_report(y_test, prediksi_svm, labels=[1, 0])))

Laporan Klasifikasi

              precision    recall  f1-score   support

         1       0.66       0.66       0.66         71
         0       0.85       0.85       0.85        160

 accuracy          0.79          0.79          0.79        231
 macro avg          0.76          0.76          0.76        231
 weighted avg          0.79          0.79          0.79        231
```