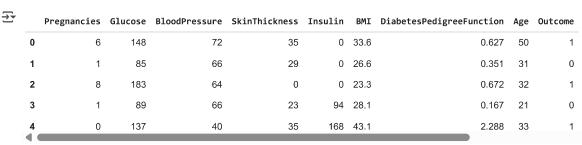
```
import pandas as pd
import numpy as np
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn import svm
from sklearn.metrics import accuracy_score
df=pd.read_csv('/content/diabetes.csv')
df.head()
```



df.shape

→ (768, 9)

df.describe()

→ *		Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	ВМІ	DiabetesPedigreeFunction	Age	Outcome
	count	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000
	mean	3.845052	120.894531	69.105469	20.536458	79.799479	31.992578	0.471876	33.240885	0.348958
	std	3.369578	31.972618	19.355807	15.952218	115.244002	7.884160	0.331329	11.760232	0.476951
	min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.078000	21.000000	0.000000
	25%	1.000000	99.000000	62.000000	0.000000	0.000000	27.300000	0.243750	24.000000	0.000000
	50%	3.000000	117.000000	72.000000	23.000000	30.500000	32.000000	0.372500	29.000000	0.000000
	75%	6.000000	140.250000	80.000000	32.000000	127.250000	36.600000	0.626250	41.000000	1.000000
	max	17.000000	199.000000	122.000000	99.000000	846.000000	67.100000	2.420000	81.000000	1.000000

df['Outcome'].value_counts()



count

Dutcome						
0	500					
1	268					

x=df.drop(columns='Outcome',axis=1)
y=df['Outcome']

scaler=StandardScaler()
scaler.fit(x)

dtuna intal



v StandardScaler ① ?
StandardScaler()

standardized_data=scaler.transform(x)
standardized_data

```
array([[ 0.63994726, 0.84832379, 0.14964075, ..., 0.20401277, 0.46849198, 1.4259954 ], [-0.84488505, -1.12339636, -0.16054575, ..., -0.68442195, -0.36506078, -0.19067191],
```

```
[ 1.23388019, 1.94372388, -0.26394125, ..., -1.10325546, 0.60439732, -0.10558415],
              [ 0.3429808 , 0.00330087, 0.14964075, ..., -0.73518964, -0.68519336, -0.27575966],
              [-0.84488505, 0.1597866, -0.47073225, ..., -0.24020459, -0.37110101, 1.17073215], [-0.84488505, -0.8730192, 0.04624525, ..., -0.20212881,
                -0.47378505, -0.87137393]])
x=standardized data
⇒ array([[ 0.63994726, 0.84832379, 0.14964075, ..., 0.20401277,
              0.46849198, 1.4259954 ],
[-0.84488505, -1.12339636, -0.16054575, ..., -0.68442195,
                -0.36506078, -0.19067191],
              [ 1.23388019, 1.94372388, -0.26394125, ..., -1.10325546, 0.60439732, -0.10558415],
              [0.3429808, 0.00330087, 0.14964075, ..., -0.73518964,
                -0.68519336, -0.27575966],
              [-0.84488505, 0.1597866, -0.47073225, ..., -0.24020459, -0.37110101, 1.17073215],
              [-0.84488505, -0.8730192, 0.04624525, ..., -0.20212881,
                -0.47378505, -0.87137393]])
у
<del>_</del>
             Outcome
        0
                    0
        2
                    0
        3
        4
       763
                    0
       764
                    0
       765
                    0
       766
                     1
       767
                    0
      768 rows × 1 columns
      dtype: int64
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2)
x train.shape
→ (614, 8)
x_test.shape
→ (154, 8)
clf=svm.SVC(kernel='linear')
clf.fit(x_train,y_train)
<del>_</del>
                         (i) (?)
               SVC
      SVC(kernel='linear')
x_train_prediction=clf.predict(x_train)
accuracy_score(x_train_prediction,y_train)
```

```
0.7833876221498371
x_test_prediction=clf.predict(x_test)
accuracy_score(x_test_prediction,y_test)
→ 0.7857142857142857
input_sample=(5,166,72,19,175,22.7,0.6,51)
input_np_array=np.asarray(input_sample)
input_np_array_reshaped=input_np_array.reshape(1,-1)
std_data=scaler.transform(input_np_array_reshaped)
🚁 /usr/local/lib/python3.11/dist-packages/sklearn/utils/validation.py:2739: UserWarning: X does not have valid feature names, but Standarc
       warnings.warn(
std_data
→ array([[ 0.3429808 , 1.41167241, 0.14964075, -0.09637905, 0.82661621,
             -1.179407 , 0.38694877, 1.51108316]])
prediction=clf.predict(std_data)
prediction
→ array([1])
if(prediction[0]==0):
 print('person is not Diabetic')
else:
 print('Person is Diabetic')
→ Person is Diabetic
Start coding or generate with AI.
```