

Course: Laboratory Practice III

Course Code: 410246

Name: Asmeeta Lalit Bardiya

Class: BE

Roll No. : 22

Div: A

Title: Implement K-Nearest Neighbors algorithm on diabetes.csv dataset. Compute confusion matrix, accuracy, error rate, precision and recall on the given dataset. Dataset link :

<https://www.kaggle.com/datasets/abdallamahgoub/diabetes>

```
import pandas as pd
import numpy as np
import seaborn as sns
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
```

```
data = pd.read_csv('/content/diabetes.csv')
```

```
data.head()
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	Pedigree	Age
0	6	148	72	35	0	33.6	0.627	51
1	1	85	66	29	0	26.6	0.351	33
2	8	183	64	0	0	23.3	0.672	33
3	1	89	66	23	94	28.1	0.167	24
4	0	137	40	35	168	43.1	2.288	33

```
data.tail()
```

**Pregnancies   Glucose   BloodPressure   SkinThickness   Insulin   BMI   Pedigree**

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
#   Column                Non-Null Count  Dtype
---  -
0   Pregnancies            768 non-null    int64
1   Glucose                768 non-null    int64
2   BloodPressure          768 non-null    int64
3   SkinThickness          768 non-null    int64
4   Insulin                768 non-null    int64
5   BMI                    768 non-null    float64
6   Pedigree               768 non-null    float64
7   Age                    768 non-null    int64
8   Outcome                768 non-null    int64
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

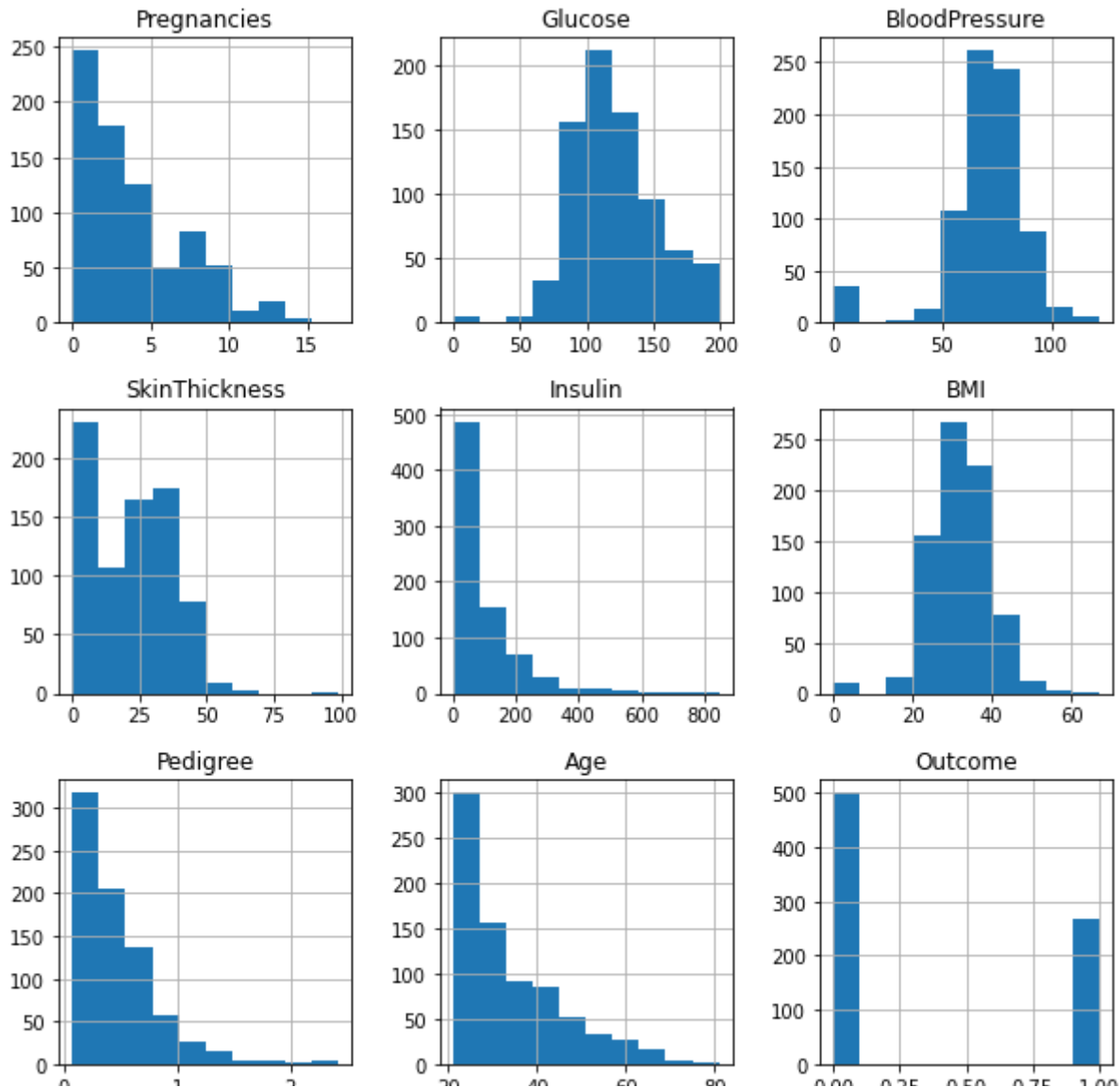
```
data.describe()
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI
<b>count</b>	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000
<b>mean</b>	3.845052	120.894531	69.105469	20.536458	79.799479	31.99257
<b>std</b>	3.369578	31.972618	19.355807	15.952218	115.244002	7.88416
<b>min</b>	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
<b>25%</b>	1.000000	99.000000	62.000000	0.000000	0.000000	27.300000
<b>50%</b>	3.000000	117.000000	72.000000	23.000000	30.500000	32.000000
<b>75%</b>	6.000000	140.250000	80.000000	32.000000	127.250000	36.600000
<b>max</b>	17.000000	199.000000	122.000000	99.000000	846.000000	67.100000

```
data.isnull().sum()
```

```
Pregnancies    0
Glucose         0
BloodPressure   0
SkinThickness   0
Insulin         0
BMI             0
Pedigree        0
Age             0
Outcome         0
dtype: int64
```

```
hist=data.hist(figsize=(10,10))
```



```
zero_not_accepted=['Glucose','BloodPressure','SkinThickness','BMI','Insulin']
for col in zero_not_accepted:
    data[col]=data[col].replace(0,np.NaN)
    mean=int(data[col].mean(skipna=True))
    data[col]=data[col].replace(np.NaN,mean)
```

```
x=data.iloc[:, :-1].values
print(x)
```

```
[[ 6.   148.   72.   ...  33.6   0.627  50.   ]
 [ 1.    85.   66.   ...  26.6   0.351  31.   ]
 [ 8.   183.   64.   ...  23.3   0.672  32.   ]
 ...
 [ 5.   121.   72.   ...  26.2   0.245  30.   ]
 [ 1.   126.   60.   ...  30.1   0.349  47.   ]
 [ 1.    93.   70.   ...  30.4   0.315  23.   ]]
```

```
y=data.iloc[:, -1].values
print(y)
```

```
[1 0 1 0 1 0 1 0 1 1 0 1 0 1 1 1 1 1 0 1 0 0 1 1 1 1 1 0 0 0 0 1 0 0 0 0 0
 1 1 1 0 0 0 1 0 1 0 0 1 0 0 0 0 1 0 0 1 0 0 0 0 1 0 0 1 0 1 0 0 0 1 0 1 0
 0 0 0 0 1 0 0 0 0 0 1 0 0 0 1 0 0 0 0 1 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 1 1]
```

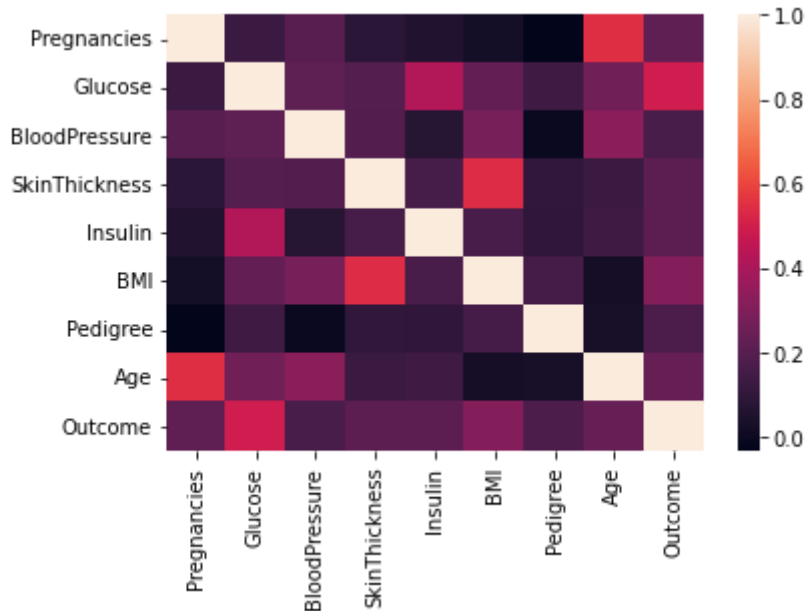
```

1 0 0 1 1 1 0 0 0 1 0 0 0 1 1 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0
0 0 0 0 1 0 1 1 0 0 0 1 0 0 0 0 1 1 0 0 0 0 1 1 0 0 0 1 0 1 0 1 0 0 0 0
1 1 1 1 1 0 0 1 1 0 1 0 1 1 1 0 0 0 0 0 0 1 1 0 1 0 0 0 1 1 1 1 0 1 1 1
0 0 0 0 0 1 0 0 1 1 0 0 0 1 1 1 1 0 0 0 1 1 0 1 0 0 0 0 0 0 0 0 1 1 0 0 0
1 0 1 0 0 1 0 1 0 0 1 1 0 0 0 0 0 1 0 0 0 1 0 0 1 1 0 0 1 0 0 0 1 1 1 0 0
1 0 1 0 1 1 0 1 0 0 1 0 1 1 0 0 1 0 1 0 0 1 0 1 0 1 1 1 0 0 1 0 1 0 0 0 1
0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 1 1 0 1 1 0 0 1 0 0 1 0 0 1
1 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 1 1 1 0 0 1 0 0 1 0 0 1 0 1 1 0 1 0 1 0 1
0 1 1 0 0 0 0 1 1 0 1 0 1 0 0 0 0 1 1 0 1 0 1 0 0 0 0 0 1 0 0 0 0 1 0 0 1
1 1 0 0 1 0 0 1 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 1
0 0 0 1 1 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 0 1 1 0
0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1 1 1 1 0 0 1 1 0 0 0 0 0 0 0 0
0 0 0 0 0 1 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0 1 1 0 0 0 1 0 1 0 1 0 1 0
1 0 0 1 0 0 1 0 0 0 0 1 1 0 1 0 0 0 0 1 1 0 1 0 0 0 1 1 0 0 0 0 0 0 0 0 0
0 1 0 0 0 0 1 0 0 1 0 0 0 1 0 0 0 1 1 1 0 0 0 0 0 0 1 0 0 0 1 0 1 1 1 1 0
1 1 0 0 0 0 0 0 0 1 1 0 1 0 0 1 0 1 0 0 0 0 0 0 1 0 1 0 1 0 1 1 0 0 0 0 1 1
0 0 0 1 0 1 1 0 0 1 0 0 1 1 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 1
1 0 0 1 0 0 1 0 1 1 1 0 0 1 1 1 0 1 0 1 0 1 0 0 0 0 0 1 0]

```

```
sns.heatmap(data.corr())
```

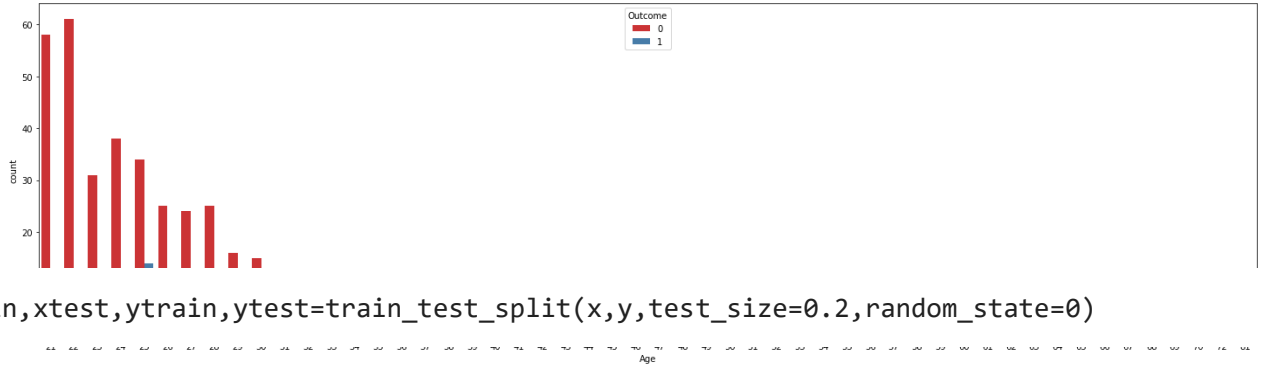
<matplotlib.axes.\_subplots.AxesSubplot at 0x7f72ff4ea710>



```
plt.figure(figsize=(25,7))
```

```
sns.countplot(x='Age',hue='Outcome',data=data,palette='Set1')
```

&lt;matplotlib.axes.\_subplots.AxesSubplot at 0x7f72ff475150&gt;



```
xtrain,xtest,ytrain,ytest=train_test_split(x,y,test_size=0.2,random_state=0)
```

```
scaler=StandardScaler()
```

```
xtrain=scaler.fit_transform(xtrain)
```

```
xtest=scaler.transform(xtest)
```

```
classifier=KNeighborsClassifier(n_neighbors=11,p=2,metric='euclidean')
```

```
classifier.fit(xtrain,ytrain)
```

```
KNeighborsClassifier(metric='euclidean', n_neighbors=11)
```

```
ypred=classifier.predict(xtest)
```

```
confusion_matrix2=confusion_matrix(ytest,ypred)
```

```
print(confusion_matrix2)
```

```
[[94 13]
 [15 32]]
```

```
print("F1 Score: ",f1_score(ytest,ypred))
```

```
F1 Score: 0.6956521739130436
```

```
print("Accuracy: ",accuracy_score(ytest,ypred))
```

```
Accuracy: 0.8181818181818182
```

```
print("Precision: ",precision_score(ytest,ypred))
```

```
print("Recall: ",recall_score(ytest,ypred))
```

```
Precision: 0.7111111111111111
```

```
Recall: 0.6808510638297872
```

```
print("Error Rate: ",1-accuracy_score(ytest,ypred))
```

```
Error Rate: 0.18181818181818177
```