Name: Saloni Vishwakarma

Batch-Roll no: C1-13

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

```
d1=pd.read_csv('Diabetes.csv')
```

```
d1.head()
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1

```
d1.describe()
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	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

```
d1.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   DiabetesPedigreeFunction  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
```

```
d=d1.loc[(d1['Glucose']!=0) & (d1['BloodPressure']!=0) &
          (d1['SkinThickness']!=0) & (d1['Insulin']!=0) & (d1['BMI']!=0)]
```

```
d.describe()
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
count	392.000000	392.000000	392.000000	392.000000	392.000000	392.000000	392.000000	392.000000	392.000000

```
d.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 392 entries, 3 to 765
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype  
---  ------                    --------------  -----  
 0   Pregnancies               392 non-null    int64  
 1   Glucose                   392 non-null    int64  
 2   BloodPressure             392 non-null    int64  
 3   SkinThickness             392 non-null    int64  
 4   Insulin                   392 non-null    int64  
 5   BMI                       392 non-null    float64
 6   DiabetesPedigreeFunction  392 non-null    float64
 7   Age                       392 non-null    int64  
 8   Outcome                   392 non-null    int64  
dtypes: float64(2), int64(7)
memory usage: 30.6 KB
```

```
d1['Glucose'].replace(0,d['Glucose'].mean(),inplace=True)
d1['BloodPressure'].replace(0,d['BloodPressure'].mean(),inplace=True)
d1['SkinThickness'].replace(0,d['SkinThickness'].mean(),inplace=True)
d1['Insulin'].replace(0,d['Insulin'].mean(),inplace=True)
d1['BMI'].replace(0,d['BMI'].mean(),inplace=True)
```

```
d1.describe()
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	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	121.692888	72.325800	29.151052	155.795560	32.466469	0.471876	33.240885	0.348958
std	3.369578	30.436043	12.101807	8.790943	85.021487	6.875558	0.331329	11.760232	0.476951
min	0.000000	44.000000	24.000000	7.000000	14.000000	18.200000	0.078000	21.000000	0.000000
25%	1.000000	99.750000	64.000000	25.000000	121.500000	27.500000	0.243750	24.000000	0.000000
50%	3.000000	117.000000	72.000000	29.145408	156.056122	32.400000	0.372500	29.000000	0.000000
75%	6.000000	140.250000	80.000000	32.000000	156.056122	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

```
d1.head()
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148.0	72.0	35.000000	156.056122	33.6	0.627	50	1
1	1	85.0	66.0	29.000000	156.056122	26.6	0.351	31	0
2	8	183.0	64.0	29.145408	156.056122	23.3	0.672	32	1
3	1	89.0	66.0	23.000000	94.000000	28.1	0.167	21	0
4	0	137.0	40.0	35.000000	168.000000	43.1	2.288	33	1

```
d1.corr()
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age
Pregnancies	1.000000	0.127849	0.208850	0.082926	0.056535	0.021589	-0.033523	0.544341
Glucose	0.127849	1.000000	0.219028	0.192985	0.419998	0.230189	0.137004	0.266453
BloodPressure	0.208850	0.219028	1.000000	0.192796	0.072908	0.281531	-0.001108	0.325860
SkinThickness	0.082926	0.192985	0.192796	1.000000	0.158154	0.542239	0.101030	0.127780
Insulin	0.056535	0.419998	0.072908	0.158154	1.000000	0.166212	0.098136	0.137366
BMI	0.021589	0.230189	0.281531	0.542239	0.166212	1.000000	0.153238	0.025207
DiabetesPedigreeFunction	-0.033523	0.137004	-0.001108	0.101030	0.098136	0.153238	1.000000	0.033561
Age	0.544341	0.266453	0.325860	0.127780	0.137366	0.025207	0.033561	1.000000
Outcome	0.221898	0.492948	0.164509	0.215277	0.214532	0.311446	0.173844	0.238356

```
x=d1.iloc[:,0:8]
```

```
y=d1.iloc[:,-1] #y=d1.iloc[:,8]
```

```
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=0)
```

```
#feature Scaling
from sklearn.preprocessing import StandardScaler
st_x= StandardScaler()
x_train= st_x.fit_transform(x_train)
x_test= st_x.transform(x_test)
```

```
#Fitting K-NN classifier to the training set
from sklearn.neighbors import KNeighborsClassifier
KNNclassifier= KNeighborsClassifier(n_neighbors=5, metric='minkowski',p=2)  # Hyper Parameter
KNNclassifier.fit(x_train, y_train)
```

```
     KNeighborsClassifier()
```

```
#Predicting the test set result
y_pred= KNNclassifier.predict(x_test)
print(y_pred)
```

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

```
#Creating the Confusion matrix
from sklearn.metrics import confusion_matrix
cm= confusion_matrix(y_test, y_pred)
```

```
print(cm)
```

```
     [[89 18]
      [17 30]]
```

```
from sklearn import metrics
print("accuracy",metrics.accuracy_score(y_test,y_pred))
```

```
     accuracy 0.7727272727272727
```

```
from sklearn.metrics import classification_report
print(classification_report(y_test,y_pred))
```

```
                   precision    recall  f1-score   support

                0       0.84      0.83      0.84       107
                1       0.62      0.64      0.63        47

         accuracy                           0.77       154
        macro avg       0.73      0.74      0.73       154
     weighted avg       0.77      0.77      0.77       154
```

K Fold

```
import sklearn.metrics as metrics
from sklearn.metrics import accuracy_score
from sklearn.model_selection import KFold
from sklearn import model_selection
kfold = KFold(n_splits=6)
cv_results = model_selection.cross_val_score(KNNclassifier, x, y, cv=kfold)
```

```
print(cv_results)
```

```
     [0.6875    0.6796875 0.6875    0.7265625 0.8046875 0.703125 ]
```

```
print(cv_results.mean())
```

```
     0.71484375
```

GridSearchCV

```
from sklearn.model_selection import GridSearchCV
parameters = {'n_neighbors': range(30),
              'metric':['manhattan','euclidean']}
c1 = KNeighborsClassifier()
grid = GridSearchCV(c1, parameters, cv=10)
grid.fit(x,y)
print(grid.best_params_)
print(grid.best_estimator_)
```

```
C:\Users\acer\anaconda3\lib\site-packages\sklearn\model_selection\_validation.py:615: FitFailedWarning: Estimator fit failed. The
Traceback (most recent call last):
  File "C:\Users\acer\anaconda3\lib\site-packages\sklearn\model_selection\_validation.py", line 598, in _fit_and_score
    estimator.fit(X_train, y_train, **fit_params)
  File "C:\Users\acer\anaconda3\lib\site-packages\sklearn\neighbors\_classification.py", line 179, in fit
    return self._fit(X, y)
  File "C:\Users\acer\anaconda3\lib\site-packages\sklearn\neighbors\_base.py", line 514, in _fit
    raise ValueError(
ValueError: Expected n_neighbors > 0. Got 0

  warnings.warn("Estimator fit failed. The score on this train-test"
C:\Users\acer\anaconda3\lib\site-packages\sklearn\model_selection\_validation.py:615: FitFailedWarning: Estimator fit failed. The
Traceback (most recent call last):
  File "C:\Users\acer\anaconda3\lib\site-packages\sklearn\model_selection\_validation.py", line 598, in _fit_and_score
    estimator.fit(X_train, y_train, **fit_params)
  File "C:\Users\acer\anaconda3\lib\site-packages\sklearn\neighbors\_classification.py", line 179, in fit
    return self._fit(X, y)
  File "C:\Users\acer\anaconda3\lib\site-packages\sklearn\neighbors\_base.py", line 514, in _fit
    raise ValueError(
ValueError: Expected n_neighbors > 0. Got 0

  warnings.warn("Estimator fit failed. The score on this train-test"
C:\Users\acer\anaconda3\lib\site-packages\sklearn\model_selection\_validation.py:615: FitFailedWarning: Estimator fit failed. The
Traceback (most recent call last):
  File "C:\Users\acer\anaconda3\lib\site-packages\sklearn\model_selection\_validation.py", line 598, in _fit_and_score
    estimator.fit(X_train, y_train, **fit_params)
  File "C:\Users\acer\anaconda3\lib\site-packages\sklearn\neighbors\_classification.py", line 179, in fit
    return self._fit(X, y)
  File "C:\Users\acer\anaconda3\lib\site-packages\sklearn\neighbors\_base.py", line 514, in _fit
    raise ValueError(
ValueError: Expected n_neighbors > 0. Got 0

  warnings.warn("Estimator fit failed. The score on this train-test"
C:\Users\acer\anaconda3\lib\site-packages\sklearn\model_selection\_validation.py:615: FitFailedWarning: Estimator fit failed. The
Traceback (most recent call last):
  File "C:\Users\acer\anaconda3\lib\site-packages\sklearn\model_selection\_validation.py", line 598, in _fit_and_score
    estimator.fit(X_train, y_train, **fit_params)
  File "C:\Users\acer\anaconda3\lib\site-packages\sklearn\neighbors\_classification.py", line 179, in fit
    return self._fit(X, y)
  File "C:\Users\acer\anaconda3\lib\site-packages\sklearn\neighbors\_base.py", line 514, in _fit
    raise ValueError(
ValueError: Expected n_neighbors > 0. Got 0

  warnings.warn("Estimator fit failed. The score on this train-test"
C:\Users\acer\anaconda3\lib\site-packages\sklearn\model_selection\_validation.py:615: FitFailedWarning: Estimator fit failed. The
Traceback (most recent call last):
  File "C:\Users\acer\anaconda3\lib\site-packages\sklearn\model_selection\_validation.py", line 598, in _fit_and_score
    estimator.fit(X_train, y_train, **fit_params)
  File "C:\Users\acer\anaconda3\lib\site-packages\sklearn\neighbors\_classification.py", line 179, in fit
    return self._fit(X, y)
  File "C:\Users\acer\anaconda3\lib\site-packages\sklearn\neighbors\_base.py", line 514, in _fit
    raise ValueError(
ValueError: Expected n_neighbors > 0. Got 0

  warnings.warn("Estimator fit failed. The score on this train-test"
C:\Users\acer\anaconda3\lib\site-packages\sklearn\model_selection\_validation.py:615: FitFailedWarning: Estimator fit failed. The
Traceback (most recent call last):
```

```
print(grid.best_score_)
print(grid.best_params_)
print(grid.best_estimator_)
```

```
0.7592276144907724
{'metric': 'manhattan', 'n_neighbors': 15}
KNeighborsClassifier(metric='manhattan', n_neighbors=15)
```