```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
from sklearn.preprocessing import scale
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn import metrics
```

```
df = pd.read_csv('diabetes.csv')
df
```

```
     Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin   BMI  \
0              6      148             72             35        0  33.6   

1              1       85             66             29        0  26.6   

2              8      183             64              0        0  23.3   

3              1       89             66             23       94  28.1   

4              0      137             40             35      168  43.1   

..           ...      ...            ...            ...      ...   ...   

763           10      101             76             48      180  32.9   

764            2      122             70             27        0  36.8   

765            5      121             72             23      112  26.2   

766            1      126             60              0        0  30.1   

767            1       93             70             31        0  30.4   


     Pedigree  Age  Outcome  
0       0.627   50        1  
1       0.351   31        0  
2       0.672   32        1  
3       0.167   21        0  
4       2.288   33        1  
..        ...  ...      ...  
763     0.171   63        0  
764     0.340   27        0  
765     0.245   30        0  
```

```
766      0.349   47         1
767      0.315   23         0

[768 rows x 9 columns]
```

```
df.columns
```

```
Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness',
'Insulin',
       'BMI', 'Pedigree', 'Age', 'Outcome'],
      dtype='object')
```

```
df.isnull().sum()
```

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

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

```
x = scale(x)
# Split into Train & Test Data
x_train, x_test, y_train, y_test = train_test_split(x, y,
test_size=0.3,
                                                    random_state=42)
```

```
knn = KNeighborsClassifier(n_neighbors=7)
knn.fit(x_train, y_train)
y_pred = knn.predict(x_test)
```

```
cs = metrics.confusion_matrix(y_test,y_pred)
print('Confusion Matrix:\n',cs)
```

```
Confusion Matrix:
 [[123  28]
 [ 37  43]]
```

```
print('Acccuracy:\n',metrics.accuracy_score(y_test,y_pred))
```

```
Acccuracy:
 0.7186147186147186
```

```
total_misclassified = cs[0,1] + cs[1,0]
print('Total Misclassified Entries:\n',total_misclassified)
```

```
total_examples = cs[0,0]+cs[0,1]+cs[1,0]+cs[1,1]
print('Total Entries:\n',total_examples)
print('Error Rate:\n',total_misclassified/total_examples)
print('Error Rate:\n',1-metrics.accuracy_score(y_test,y_pred))
```

```
Total Misclassified Entries:
 65
Total Entries:
 231
Error Rate:
 0.2813852813852814
Error Rate:
 0.2813852813852814
```

```
print('Precision Score:\n',metrics.precision_score(y_test,y_pred))
```

```
Precision Score:
 0.6056338028169014
```

```
print('Recall Score:\n',metrics.recall_score(y_test,y_pred))
```

```
Recall Score:
 0.5375
```

```
print('Classification Report\
n',metrics.classification_report(y_test,y_pred))
```

```
Classification Report
               precision    recall  f1-score   support

           0       0.77      0.81      0.79       151
           1       0.61      0.54      0.57        80

    accuracy                           0.72       231
   macro avg       0.69      0.68      0.68       231
weighted avg       0.71      0.72      0.71       231
```