Lab Assignment 2: Data Preprocessing Pipeline

Aim: To implement a data preprocessing pipeline, including data cleaning, feature engineering, and basic classification, using the Heart Disease UCI dataset.

Task 1: Load and Explore the Dataset

1. Load the Heart Disease UCI dataset using pandas.

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')

df = pd.read_csv('heart_disease_uci.csv')
df.head()
```

```
   id  age     sex   dataset               cp  trestbps   chol    fbs
\
0   1   63    Male  Cleveland   typical angina     145.0  233.0   True

1   2   67    Male  Cleveland     asymptomatic     160.0  286.0  False

2   3   67    Male  Cleveland     asymptomatic     120.0  229.0  False

3   4   37    Male  Cleveland      non-anginal     130.0  250.0  False

4   5   41  Female  Cleveland  atypical angina     130.0  204.0  False


          restecg  thalch  exang  oldpeak        slope   ca  \
0  lv hypertrophy   150.0  False      2.3  downsloping  0.0
1  lv hypertrophy   108.0   True      1.5         flat  3.0
2  lv hypertrophy   129.0   True      2.6         flat  2.0
3          normal   187.0  False      3.5  downsloping  0.0
4  lv hypertrophy   172.0  False      1.4    upsloping  0.0

                thal  num
0      fixed defect    0
1             normal   2
2  reversable defect   1
3             normal   0
4             normal   0
```

2. Display dataset characteristics:

– Number of records and features

```
df.shape

(920, 16)
```

– Data types of columns

```
df.dtypes

id            int64
age           int64
sex          object
dataset      object
cp           object
trestbps    float64
chol        float64
fbs          object
restecg      object
thalch      float64
exang        object
oldpeak     float64
slope        object
ca          float64
thal         object
num           int64
dtype: object
```

– Summary statistics (mean, median, standard deviation, etc.).

```
df.describe()

              id         age    trestbps        chol      thalch  
oldpeak  \
count  920.000000  920.000000  861.000000  890.000000  865.000000  
858.000000
mean   460.500000   53.510870  132.132404  199.130337  137.545665  
0.878788
std    265.725422    9.424685   19.066070  110.780810   25.926276  
1.091226
min      1.000000   28.000000    0.000000    0.000000   60.000000   -
2.600000
25%    230.750000   47.000000  120.000000  175.000000  120.000000  
0.000000
50%    460.500000   54.000000  130.000000  223.000000  140.000000  
0.500000
75%    690.250000   60.000000  140.000000  268.000000  157.000000  
1.500000
max    920.000000   77.000000  200.000000  603.000000  202.000000  
6.200000
```

```
                ca         num
count  309.000000  920.000000
mean     0.676375    0.995652
std      0.935653    1.142693
min      0.000000    0.000000
25%      0.000000    0.000000
50%      0.000000    1.000000
75%      1.000000    2.000000
max      3.000000    4.000000
```

```
df.median(numeric_only = True)
```

```
id          460.5
age          54.0
trestbps    130.0
chol        223.0
thalch      140.0
oldpeak       0.5
ca            0.0
num           1.0
dtype: float64
```

3. Identify missing values and check for duplicate records.

```
df.isnull()
```

```
        id    age    sex  dataset     cp  trestbps   chol    fbs
restecg  \
0    False  False  False    False  False     False  False  False
False
1    False  False  False    False  False     False  False  False
False
2    False  False  False    False  False     False  False  False
False
3    False  False  False    False  False     False  False  False
False
4    False  False  False    False  False     False  False  False
False
..     ...    ...    ...      ...    ...       ...    ...    ...
...
915  False  False  False    False  False     False  False  False
False
916  False  False  False    False  False      True  False  False
False
917  False  False  False    False  False     False  False  False
False
918  False  False  False    False  False      True  False  False
False
919  False  False  False    False  False     False  False  False
```

```
False

     thalch  exang  oldpeak  slope     ca   thal    num
0     False  False    False  False  False  False  False
1     False  False    False  False  False  False  False
2     False  False    False  False  False  False  False
3     False  False    False  False  False  False  False
4     False  False    False  False  False  False  False
..      ...    ...      ...    ...    ...    ...    ...
915   False  False    False   True   True   True  False
916    True   True     True   True   True   True  False
917   False  False    False   True   True  False  False
918    True   True     True   True   True   True  False
919   False  False    False   True   True   True  False

[920 rows x 16 columns]
```

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

```
id            0
age           0
sex           0
dataset       0
cp            0
trestbps     59
chol         30
fbs          90
restecg       2
thalch       55
exang        55
oldpeak      62
slope       309
ca          611
thal        486
num           0
dtype: int64
```

```
df.duplicated().sum()
```

```
0
```

Task 2: Data Cleaning

1. Handle missing values using:

Mean/median/mode imputation for numerical columns.

```
numeric = df.select_dtypes(include =['float64','int64']).columns
df[numeric] = df[numeric].fillna(df[numeric].median())
```

Most frequent category imputation for categorical columns.

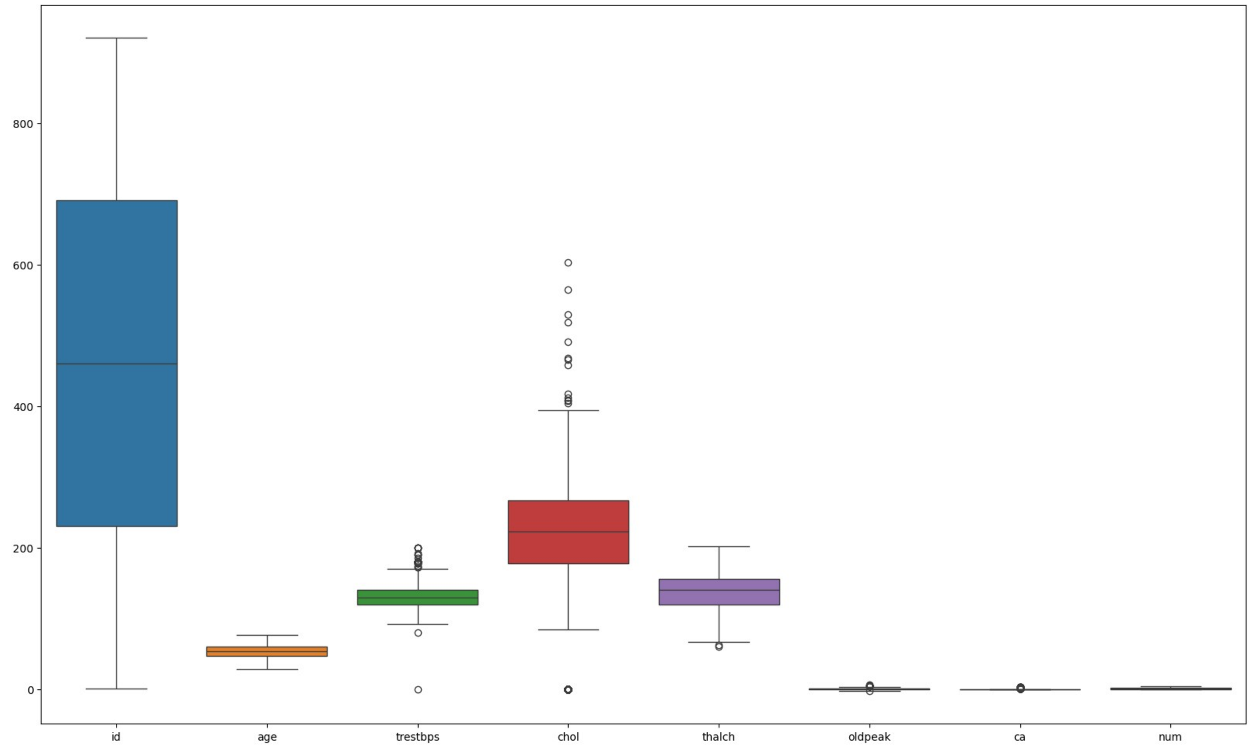
```
categoric = df.select_dtypes(include=['object']).columns
df[categoric] = df[categoric].fillna(df[categoric].mode())
```

df.isnull().sum()

3. Detect and handle outliers using:

– Box plots

```
num_columns = df.select_dtypes(include=['float64', 'int64']).columns
plt.figure(figsize = (20,12))
sns.boxplot(data=df[num_columns])
plt.show()
```



– Z-score method

```
from scipy.stats import zscore
z_scores = df[numeric].apply(zscore)

zero_outliers = df[(z_scores < 3).all(axis=1)]
df = zero_outliers

df_shape = df.shape

print('Data after removing outliers:',df_shape)
```

```
Data after removing outliers: (887, 16)
```

Task 3: Feature Engineering

1. Convert categorical features into numerical format using:

– One-hot encoding

```
df = pd.get_dummies(df, columns=categoric, drop_first=True)

df.head()

   id  age  trestbps   chol  thalch  oldpeak   ca  num  sex_Male  \
0   1   63     145.0  233.0   150.0      2.3  0.0    0      True
2   3   67     120.0  229.0   129.0      2.6  2.0    1      True
3   4   37     130.0  250.0   187.0      3.5  0.0    0      True
4   5   41     130.0  204.0   172.0      1.4  0.0    0     False
5   6   56     120.0  236.0   178.0      0.8  0.0    0      True

   dataset_Hungary  ...  cp_non-anginal  cp_typical angina
fbs_True  \
0            False  ...           False               True       True

2            False  ...           False              False      False

3            False  ...            True              False      False

4            False  ...           False              False      False

5            False  ...           False              False      False


   restecg_normal  restecg_st-t abnormality  exang_True  slope_flat  \
0           False                     False       False       False
2           False                     False        True        True
3            True                     False       False       False
4           False                     False       False       False
5            True                     False       False       False

   slope_upsloping  thal_normal  thal_reversable defect
0            False        False                  False
2            False        False                   True
3            False         True                  False
4             True         True                  False
5             True         True                  False

[5 rows x 23 columns]
```

– Label encoding

```
from sklearn.preprocessing import LabelEncoder
categoric = df.select_dtypes(include=['object']).columns

label_encoder = LabelEncoder()
```

```
for col in categoric:
    if col in df.columns:
        df[col] = label_encoder.fit_transform(df[col])
    else:
        print(f"Column '{col}' not found in the dataset.")
```

2. Normalize numerical features using:

– Min-Max Scaling

```
from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler()
df[numeric] = scaler.fit_transform(df[numeric])
```

– Standardization

```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
df[numeric] = scaler.fit_transform(df[numeric])
```

3. Analyse feature importance using correlation analysis.

```
corr= df.corr()
corr

                          id       age  trestbps      chol
thalch  \
id                  1.000000  0.269867  0.052029 -0.367751 -
0.446318
age                 0.269867  1.000000  0.232077 -0.098067 -
0.361037
trestbps            0.052029  0.232077  1.000000  0.102173 -
0.102014
chol               -0.367751 -0.098067  0.102173  1.000000
0.231315
thalch             -0.446318 -0.361037 -0.102014  0.231315
1.000000
oldpeak             0.055014  0.231706  0.144652  0.026499 -
0.157453
ca                 -0.363204  0.208402  0.002857  0.127913
0.063583
num                 0.318765  0.332769  0.078398 -0.264661 -
0.359543
sex_Male            0.287763  0.069513  0.008305 -0.203249 -
0.179379
dataset_Hungary    -0.047275 -0.400148  0.032937  0.322291
0.036979
dataset_Switzerland  0.286409  0.077934 -0.051838 -0.729107 -
0.249525
```

```
dataset_VA Long Beach     0.716469  0.342518  0.032815 -0.087155 -
0.225492
cp_atypical angina       -0.167262 -0.217299 -0.052948  0.162080
0.247820
cp_non-anginal           -0.096052 -0.011819 -0.030240  0.000786
0.142558
cp_typical angina        -0.068768  0.033895  0.042978  0.018260
0.100555
fbs_True                  0.160651  0.227493  0.148356  0.076576 -
0.028136
restecg_normal           -0.042409 -0.215523 -0.096619 -0.024756
0.015796
restecg_st-t abnormality  0.391885  0.147810  0.090487 -0.127878 -
0.135210
exang_True                0.169577  0.162084  0.150267 -0.038586 -
0.364215
slope_flat               -0.039195  0.111591  0.064928 -0.039741 -
0.240757
slope_upsloping          -0.333088 -0.010953 -0.051488 -0.004160
0.304255
thal_normal              -0.481388 -0.031038 -0.041228  0.125905
0.295141
thal_reversable defect   -0.169329  0.124124  0.070475 -0.109566 -
0.104733

                           oldpeak        ca       num  sex_Male  \
id                        0.055014 -0.363204  0.318765  0.287763
age                       0.231706  0.208402  0.332769  0.069513
trestbps                  0.144652  0.002857  0.078398  0.008305
chol                      0.026499  0.127913 -0.264661 -0.203249
thalch                   -0.157453  0.063583 -0.359543 -0.179379
oldpeak                   1.000000  0.173369  0.406139  0.116991
ca                        0.173369  1.000000  0.204365 -0.047612
num                       0.406139  0.204365  1.000000  0.288093
sex_Male                  0.116991 -0.047612  0.288093  1.000000
dataset_Hungary          -0.168533 -0.239184 -0.375735 -0.119723
dataset_Switzerland      -0.065446 -0.082028  0.295718  0.132948
dataset_VA Long Beach     0.147164 -0.185516  0.266000  0.238220
cp_atypical angina       -0.242570 -0.090715 -0.345873 -0.174806
cp_non-anginal           -0.110003 -0.036527 -0.152485 -0.069607
cp_typical angina         0.028843  0.060215 -0.046499 -0.006461
fbs_True                  0.038580  0.037424  0.127674  0.081332
restecg_normal           -0.080490 -0.114772 -0.151280 -0.019056
restecg_st-t abnormality  0.028249 -0.153849  0.143632  0.063101
exang_True                0.422426  0.053196  0.357586  0.155638
slope_flat                0.402720  0.133087  0.245719  0.028297
slope_upsloping          -0.098867  0.156733 -0.140298 -0.102855
thal_normal              -0.112379  0.150625 -0.232888 -0.300040
thal_reversable defect    0.124522  0.238635  0.268577  0.145117
```

```
                          dataset_Hungary  ...  cp_non-anginal  \
id                              -0.047275  ...       -0.096052
age                             -0.400148  ...       -0.011819
trestbps                         0.032937  ...       -0.030240
chol                             0.322291  ...        0.000786
thalch                           0.036979  ...        0.142558
oldpeak                         -0.168533  ...       -0.110003
ca                              -0.239184  ...       -0.036527
num                             -0.375735  ...       -0.152485
sex_Male                        -0.119723  ...       -0.069607
dataset_Hungary                  1.000000  ...       -0.057715
dataset_Switzerland             -0.275589  ...       -0.085929
dataset_VA Long Beach           -0.372920  ...        0.018223
cp_atypical angina               0.295872  ...       -0.261126
cp_non-anginal                  -0.057715  ...        1.000000
cp_typical angina               -0.039617  ...       -0.123526
fbs_True                        -0.158002  ...        0.008646
restecg_normal                   0.276094  ...        0.007164
restecg_st-t abnormality        -0.037100  ...       -0.041412
exang_True                      -0.094440  ...       -0.178894
slope_flat                      -0.090696  ...       -0.099093
slope_upsloping                 -0.300994  ...        0.079916
thal_normal                     -0.318365  ...        0.180820
thal_reversable defect          -0.277215  ...       -0.074065

                          cp_typical angina  fbs_True  restecg_normal
\
id                                -0.068768  0.160651       -0.042409

age                                0.033895  0.227493       -0.215523

trestbps                           0.042978  0.148356       -0.096619

chol                               0.018260  0.076576       -0.024756

thalch                             0.100555 -0.028136        0.015796

oldpeak                            0.028843  0.038580       -0.080490

ca                                 0.060215  0.037424       -0.114772

num                               -0.046499  0.127674       -0.151280

sex_Male                          -0.006461  0.081332       -0.019056

dataset_Hungary                   -0.039617 -0.158002        0.276094

dataset_Switzerland               -0.032011 -0.118266        0.070449
```

```
dataset_VA Long Beach                -0.025810  0.289114        -0.226524

cp_atypical angina                   -0.112977 -0.057066         0.105877

cp_non-anginal                       -0.123526  0.008646         0.007164

cp_typical angina                     1.000000  0.049457        -0.066759

fbs_True                              0.049457  1.000000        -0.169721

restecg_normal                       -0.066759 -0.169721         1.000000

restecg_st-t abnormality             -0.011967  0.137522        -0.619193

exang_True                           -0.121568 -0.008873        -0.034914

slope_flat                           -0.006279 -0.012723        -0.003383

slope_upsloping                       0.024789 -0.068045        -0.026165

thal_normal                           0.068662 -0.058917        -0.052210

thal_reversable defect               -0.024247  0.026848        -0.030992


                          restecg_st-t abnormality  exang_True  
slope_flat  \
id                                        0.391885    0.169577   -
0.039195
age                                       0.147810    0.162084    
0.111591
trestbps                                  0.090487    0.150267    
0.064928
chol                                     -0.127878   -0.038586   -
0.039741
thalch                                   -0.135210   -0.364215   -
0.240757
oldpeak                                   0.028249    0.422426    
0.402720
ca                                       -0.153849    0.053196    
0.133087
num                                       0.143632    0.357586    
0.245719
sex_Male                                  0.063101    0.155638    
0.028297
dataset_Hungary                          -0.037100   -0.094440   -
0.090696
dataset_Switzerland                       0.041107    0.061012    
0.104798
dataset_VA Long Beach                     0.362643    0.120947   -
```

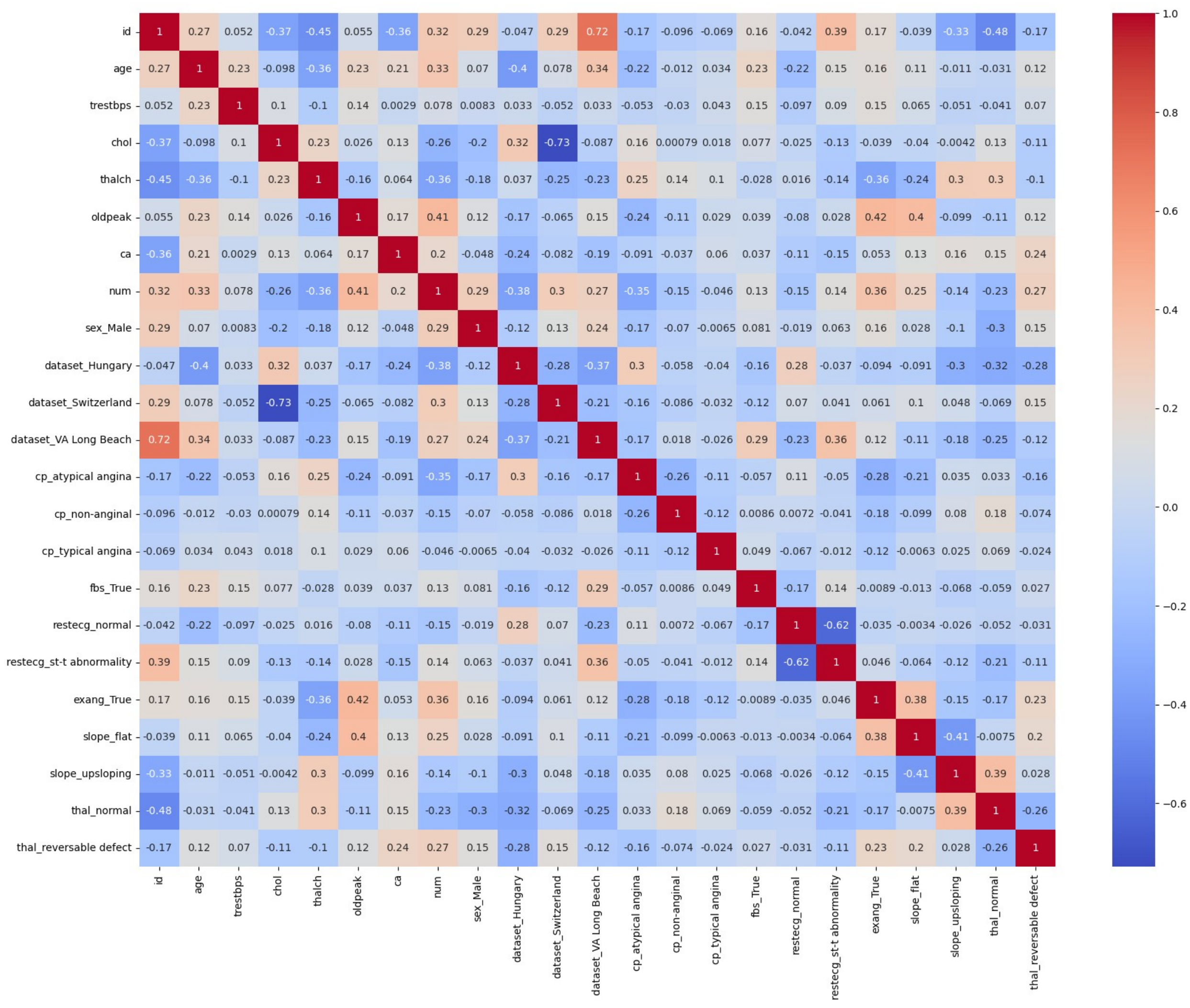
```
0.114072
cp_atypical angina                              -0.049661   -0.280743   -
0.213498
cp_non-anginal                                  -0.041412   -0.178894   -
0.099093
cp_typical angina                               -0.011967   -0.121568   -
0.006279
fbs_True                                         0.137522   -0.008873   -
0.012723
restecg_normal                                  -0.619193   -0.034914   -
0.003383
restecg_st-t abnormality                         1.000000    0.046468   -
0.063775
exang_True                                       0.046468    1.000000
0.378427
slope_flat                                      -0.063775    0.378427
1.000000
slope_upsloping                                 -0.116223   -0.145074   -
0.408309
thal_normal                                     -0.209606   -0.168888   -
0.007478
thal_reversable defect                          -0.111694    0.225334
0.202500

                          slope_upsloping  thal_normal
thal_reversable defect
id                               -0.333088    -0.481388               -
0.169329
age                              -0.010953    -0.031038
0.124124
trestbps                         -0.051488    -0.041228
0.070475
chol                             -0.004160     0.125905               -
0.109566
thalch                            0.304255     0.295141               -
0.104733
oldpeak                          -0.098867    -0.112379
0.124522
ca                                0.156733     0.150625
0.238635
num                              -0.140298    -0.232888
0.268577
sex_Male                         -0.102855    -0.300040
0.145117
dataset_Hungary                  -0.300994    -0.318365               -
0.277215
dataset_Switzerland               0.048412    -0.069489
0.149621
dataset_VA Long Beach            -0.183338    -0.252464               -
```

```
0.117226
cp_atypical angina                 0.034525     0.033261                -
0.156105
cp_non-anginal                     0.079916     0.180820                -
0.074065
cp_typical angina                  0.024789     0.068662                -
0.024247
fbs_True                          -0.068045    -0.058917
0.026848
restecg_normal                    -0.026165    -0.052210                -
0.030992
restecg_st-t abnormality          -0.116223    -0.209606                -
0.111694
exang_True                        -0.145074    -0.168888
0.225334
slope_flat                        -0.408309    -0.007478
0.202500
slope_upsloping                    1.000000     0.386556
0.028169
thal_normal                        0.386556     1.000000                -
0.257110
thal_reversable defect             0.028169    -0.257110
1.000000

[23 rows x 23 columns]
```

```
plt.figure(figsize=(20, 15))
sns.heatmap(corr, annot=True, cmap="coolwarm")
plt.show()
```



Task 4: Basic Classification using k-Nearest Neighbors (kNN)

1. Split the dataset into training (80%) and testing (20%) sets.

```
from sklearn.model_selection import train_test_split

X = df.drop('num',axis=1)
y = df['num']

X_train,X_test,y_train,y_test =
train_test_split(X,y,test_size=0.2,random_state=42)

y_train = y_train.astype(int)
y_test = y_test.astype(int)

print("Unique values in y_train:", y_train.unique())

Unique values in y_train: [1 0 2]
```

2. Train a k-Nearest Neighbors (kNN) classifier using Scikit-learn.

```
from sklearn.neighbors import KNeighborsClassifier

KNN = KNeighborsClassifier(n_neighbors=5)
KNN.fit(X_train, y_train)

KNeighborsClassifier()
```

3. Evaluate the model's performance using:

```
from sklearn.metrics import accuracy_score, classification_report, 
confusion_matrix
y_pred = KNN.predict(X_test)
accuracy = accuracy_score(y_test,y_pred)
print(accuracy)

0.8595505617977528

cr = classification_report(y_test, y_pred)
print(cr)

              precision    recall  f1-score   support

           0       0.88      0.97      0.92       155
           1       0.33      0.10      0.15        20
           2       0.00      0.00      0.00         3

    accuracy                           0.86       178
   macro avg       0.40      0.36      0.36       178
weighted avg       0.80      0.86      0.82       178


cm = confusion_matrix(y_test, y_pred)
print(cm)

[[151   4   0]
 [ 18   2   0]
 [  3   0   0]]
```