Project-3 Report Document

Kiran Gowda – 018761559

Karthik Ganduri – 018779902

- ➔ **data = pd.read_csv("wine_quality.csv",header=None, names = table_header)**
- ➔ **print(data)** *# Displaying the dataset from the dataset file wine_quality.csv*

```
      fixedacid  volacid  citricacid  residualsugar  chlorides  freesulfur  \
0           7.0    0.270        0.36          20.70      0.045        45.0
1           7.2    0.230        0.32           8.50      0.058        47.0
2           7.2    0.230        0.32           8.50      0.058        47.0
3           7.0    0.270        0.36          20.70      0.045        45.0
4           6.3    0.300        0.34           1.60      0.049        14.0
...         ...      ...         ...            ...        ...         ...
4896        4.9    0.235        0.27          11.75      0.030        34.0
4897        6.1    0.340        0.29           2.20      0.036        25.0
4898        5.7    0.210        0.32           0.90      0.038        38.0
4899        6.5    0.230        0.38           1.30      0.032        29.0
4900        6.5    0.240        0.19           1.20      0.041        30.0

      totalsulfur  density    pH  sulphates  alcohol  quality
0           170.0  1.00100  3.00       0.45      8.8      6.0
1           186.0  0.99560  3.19       0.40      9.9      6.0
2           186.0  0.99560  3.19       0.40      9.9      6.0
3           170.0  1.00100  3.00       0.45      8.8      6.0
4           132.0  0.99400  3.30       0.49      9.5      6.0
...           ...      ...   ...        ...      ...      ...
4896        118.0  0.99540  3.07       0.50      9.4      6.0
4897        100.0  0.98938  3.06       0.44     11.8      6.0
4898        121.0  0.99074  3.24       0.46     10.6      6.0
4899        112.0  0.99298  3.29       0.54      9.7      5.0
4900        111.0  0.99254  2.99       0.46      9.4      6.0

[4901 rows x 12 columns]
```

➔ **data['new_quality'] = [0 if i <= 5 else 1 for i in data.quality]**

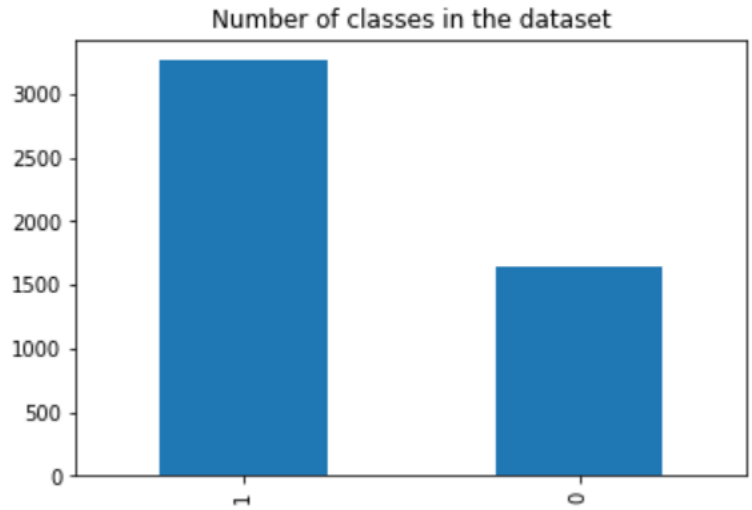
➔ **print(data1)** *# Displaying the data after adding new column 'new quality'.*

```
      fixedacid  volacid  citricacid  residualsugar  chlorides  freesulfur  \
0           7.0    0.270        0.36          20.70      0.045        45.0
1           7.2    0.230        0.32           8.50      0.058        47.0
2           7.2    0.230        0.32           8.50      0.058        47.0
3           7.0    0.270        0.36          20.70      0.045        45.0
4           6.3    0.300        0.34           1.60      0.049        14.0
...         ...      ...         ...            ...        ...         ...
4896        4.9    0.235        0.27          11.75      0.030        34.0
4897        6.1    0.340        0.29           2.20      0.036        25.0
4898        5.7    0.210        0.32           0.90      0.038        38.0
4899        6.5    0.230        0.38           1.30      0.032        29.0
4900        6.5    0.240        0.19           1.20      0.041        30.0

      totalsulfur  density    pH  sulphates  alcohol  quality  new_quality
0           170.0  1.00100  3.00       0.45      8.8      6.0            1
1           186.0  0.99560  3.19       0.40      9.9      6.0            1
2           186.0  0.99560  3.19       0.40      9.9      6.0            1
3           170.0  1.00100  3.00       0.45      8.8      6.0            1
4           132.0  0.99400  3.30       0.49      9.5      6.0            1
...           ...      ...   ...        ...      ...      ...          ...
4896        118.0  0.99540  3.07       0.50      9.4      6.0            1
4897        100.0  0.98938  3.06       0.44     11.8      6.0            1
4898        121.0  0.99074  3.24       0.46     10.6      6.0            1
4899        112.0  0.99298  3.29       0.54      9.7      5.0            0
4900        111.0  0.99254  2.99       0.46      9.4      6.0            1

[4901 rows x 13 columns]
```

➔ **data['new_quality'].value_counts().plot(kind = 'bar', title = 'Number of classes in the dataset')** *# Plotting the graph for the number of classes found in the dataset*



➔ **data1 = data.drop(columns = ['quality'])**

➔ **print(data1)** *# Displaying the data after dropping the column 'quality'.*

```
      fixedacid  volacid  citricacid  residualsugar  chlorides  freesulfur  \
0           7.0    0.270        0.36          20.70      0.045        45.0
1           7.2    0.230        0.32           8.50      0.058        47.0
2           7.2    0.230        0.32           8.50      0.058        47.0
3           7.0    0.270        0.36          20.70      0.045        45.0
4           6.3    0.300        0.34           1.60      0.049        14.0
...         ...      ...         ...            ...        ...         ...
4896        4.9    0.235        0.27          11.75      0.030        34.0
4897        6.1    0.340        0.29           2.20      0.036        25.0
4898        5.7    0.210        0.32           0.90      0.038        38.0
4899        6.5    0.230        0.38           1.30      0.032        29.0
4900        6.5    0.240        0.19           1.20      0.041        30.0

      totalsulfur  density    pH  sulphates  alcohol  new_quality
0           170.0  1.00100  3.00       0.45      8.8            1
1           186.0  0.99560  3.19       0.40      9.9            1
2           186.0  0.99560  3.19       0.40      9.9            1
3           170.0  1.00100  3.00       0.45      8.8            1
4           132.0  0.99400  3.30       0.49      9.5            1
...           ...      ...   ...        ...      ...          ...
4896        118.0  0.99540  3.07       0.50      9.4            1
4897        100.0  0.98938  3.06       0.44     11.8            1
4898        121.0  0.99074  3.24       0.46     10.6            1
4899        112.0  0.99298  3.29       0.54      9.7            0
4900        111.0  0.99254  2.99       0.46      9.4            1

[4901 rows x 12 columns]
```

➔ **data2 = data['new_quality']**

➔ **print(data2)** *# Displaying the class values.*

```
0       1
1       1
2       1
3       1
4       1
       ..
4896    1
4897    1
4898    1
4899    0
4900    1
Name: new_quality, Length: 4901, dtype: int64
```

➔ **print(pd.DataFrame(data2.value_counts()))** *# no of high classes and low classes.*

```
   new_quality
1         3260
0         1641
```

➔ X_train, X_test, y_train, y_test = train_test_split(data1, data2, test_size=0.2)
➔ print(X_train) *# Displaying trained split data.*

```
      fixedacid  volacid  citricacid  residualsugar  chlorides  freesulfur  \
745         6.8     0.21        0.36          18.10      0.046        32.0
4394        6.7     0.21        0.34           1.50      0.035        45.0
2507        7.1     0.85        0.49           8.70      0.028        40.0
4107        6.3     0.24        0.29          13.70      0.035        53.0
2567        7.6     0.18        0.49          18.05      0.046        36.0
...         ...      ...         ...            ...        ...         ...
3721        7.1     0.14        0.33           1.00      0.104        20.0
3027        7.7     0.39        0.34          10.00      0.056        35.0
2524        6.6     0.30        0.24           1.20      0.034        17.0
771         6.9     0.13        0.28          13.30      0.050        47.0
548         7.7     0.44        0.24          11.20      0.031        41.0

      totalsulfur  density    pH  sulphates  alcohol  new_quality
745         133.0  1.00000  3.27       0.48      8.8            0
4394        123.0  0.98949  3.24       0.36     12.6            1
2507        184.0  0.99620  3.22       0.36     10.7            0
4107        134.0  0.99567  3.17       0.38     10.6            1
2567        158.0  0.99960  3.06       0.41      9.2            0
...           ...      ...   ...        ...      ...          ...
3721         54.0  0.99057  3.19       0.64     11.5            1
3027        178.0  0.99740  3.26       0.60     10.2            0
2524        121.0  0.99330  3.13       0.36      9.2            0
771         132.0  0.99655  3.34       0.42     10.1            1
548         167.0  0.99480  3.12       0.43     11.3            1

[3920 rows x 12 columns]
```

➔ accuracy = {}
➔ knn = KNeighborsClassifier(n_neighbors = k)
➔ model = knn.fit(X_train,y_train)
➔ pred = knn.predict(X_test)
➔ accuracy = metrics.accuracy_score(y_test,pred)
➔ print(accuracy) *# calculating the accuracy.*

```
0.7145769622833843
```

➔ kfold = KFold(n_splits=5, shuffle=False)
➔ cv_score = cross_val_score(knn, data1, data2,cv = cv1), scoring='accuracy')
➔ print(cv_score) *# calculating accuracy for each fold.*

```
[0.67686035 0.67142857 0.65816327 0.7122449  0.69489796]
```

➔ print(cv_score.mean()) *# displaying mean accuracy.*

```
0.6827190080925336
```

- ➔ **grid = KNeighborsClassifier()**
- ➔ **grid1 = {"n_neighbors":np.arange(R1,R2)}**
- ➔ **knn_gridcv = GridSearchCV(grid,grid1, cv = cv1)**
- ➔ **knn_gridcv.fit(data1,data2)** *# calculating accuracy for each value of k in the range 1-25.*

```
GridSearchCV(cv=5, error_score='raise-deprecating',
             estimator=KNeighborsClassifier(algorithm='auto', leaf_size=30,
                                            metric='minkowski',
                                            metric_params=None, n_jobs=None,
                                            n_neighbors=5, p=2,
                                            weights='uniform'),
             iid='warn', n_jobs=None,
             param_grid={'n_neighbors': array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16, 17,
       18, 19, 20, 21, 22, 23, 24])},
             pre_dispatch='2*n_jobs', refit=True, return_train_score=False,
             scoring=None, verbose=0)
```

- ➔ **print(knn_gridcv.best_params_)** *# the best value of k*.

```
{'n_neighbors': 1}
```

- ➔ **print(knn_gridcv.best_score_)** *# the accuracy for gridsearchCV*

```
0.7463782901448683
```