knn2-1

November 21, 2023

1 ASSIGNMENT 4

NAME: RISHAV KUMAR

ROLL NO. 2301560042 My github account link : Github

2 Question 2

Dataset Link: Link

[120]:

```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score, classification_report,␣
 ↪confusion_matrix
from sklearn.preprocessing import StandardScaler
```

[121]:

```
df = pd.read_csv(r"C:\Users\risha\Documents\KRMU\AIML_assigment\datasets\cancer.
 ↪csv").drop(columns=['id','bare_nuclei'])
df.head()
```

[121]:

```
   clump_thickness  unif_cell_size  unif_cell_shape  marg_adhesion  \
0                5               1                1              1
1                5               4                4              5
2                3               1                1              1
3                6               8                8              1
4                4               1                1              3

   single_epith_cell_size  bland_chrom  norm_nucleoli  mitoses  classes
0                       2            3              1        1        0
1                       7            3              2        1        0
2                       2            3              1        1        0
3                       3            3              7        1        0
4                       2            3              1        1        0
```

[122]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 699 entries, 0 to 698
Data columns (total 9 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   clump_thickness         699 non-null    int64
 1   unif_cell_size          699 non-null    int64
 2   unif_cell_shape         699 non-null    int64
 3   marg_adhesion           699 non-null    int64
 4   single_epith_cell_size  699 non-null    int64
 5   bland_chrom             699 non-null    int64
 6   norm_nucleoli           699 non-null    int64
 7   mitoses                 699 non-null    int64
 8   classes                 699 non-null    int64
dtypes: int64(9)
memory usage: 49.3 KB
```

[123]:

```
df.isna().sum()
```

[123]:

```
clump_thickness           0
unif_cell_size            0
unif_cell_shape           0
marg_adhesion             0
single_epith_cell_size    0
bland_chrom               0
norm_nucleoli             0
mitoses                   0
classes                   0
dtype: int64
```

[124]:

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

[124]:

```
258
```

[125]:

```
df=df.drop_duplicates()
```

[126]:

```
df.columns
```

[126]:

```
Index(['clump_thickness', 'unif_cell_size', 'unif_cell_shape', 'marg_adhesion',
       'single_epith_cell_size', 'bland_chrom', 'norm_nucleoli', 'mitoses',
       'classes'],
      dtype='object')
```

[127]:

```
knn = []
for i in range(1,21):
    classifier = KNeighborsClassifier(n_neighbors=i)
    trained_model=classifier.fit(X_train,y_train)
    trained_model.fit(X_train,y_train )
    y_pred = classifier.predict(X_test)
    cm_KNN = confusion_matrix(y_test, y_pred)

    print(cm_KNN)
    print("Accuracy score of train KNN")
    print(accuracy_score(y_train, trained_model.predict(X_train))*100)

    print("Accuracy score of test KNN")
    print(accuracy_score(y_test, y_pred)*100)

    knn.append(accuracy_score(y_test, y_pred)*100)
```

```
[[38  4]
 [ 9 38]]
Accuracy score of train KNN
99.7159090909091
Accuracy score of test KNN
85.39325842696628
[[40  2]
 [10 37]]
Accuracy score of train KNN
94.60227272727273
Accuracy score of test KNN
86.51685393258427
[[39  3]
 [ 2 45]]
Accuracy score of train KNN
95.17045454545455
Accuracy score of test KNN
94.3820224719101
[[39  3]
 [ 6 41]]
Accuracy score of train KNN
94.31818181818183
Accuracy score of test KNN
89.8876404494382
[[38  4]
 [ 4 43]]
Accuracy score of train KNN
94.0340909090909
Accuracy score of test KNN
91.01123595505618
[[40  2]
 [ 5 42]]
Accuracy score of train KNN
94.60227272727273
```

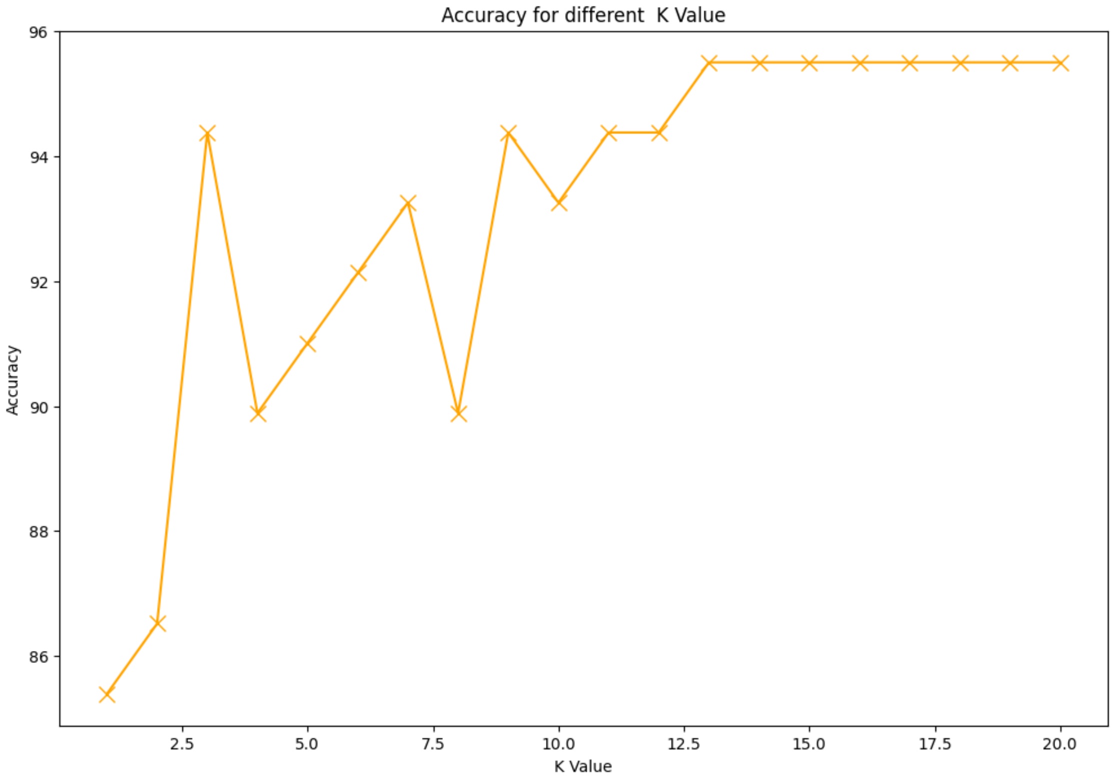
```
Accuracy score of test KNN
92.13483146067416
[[40  2]
 [ 4 43]]
Accuracy score of train KNN
94.88636363636364
Accuracy score of test KNN
93.25842696629213
[[40  2]
 [ 7 40]]
Accuracy score of train KNN
94.60227272727273
Accuracy score of test KNN
89.8876404494382
[[40  2]
 [ 3 44]]
Accuracy score of train KNN
94.88636363636364
Accuracy score of test KNN
94.3820224719101
[[40  2]
 [ 4 43]]
Accuracy score of train KNN
94.88636363636364
Accuracy score of test KNN
93.25842696629213
[[40  2]
 [ 3 44]]
Accuracy score of train KNN
94.31818181818183
Accuracy score of test KNN
94.3820224719101
[[40  2]
 [ 3 44]]
Accuracy score of train KNN
94.0340909090909
Accuracy score of test KNN
94.3820224719101
[[40  2]
 [ 2 45]]
Accuracy score of train KNN
94.0340909090909
Accuracy score of test KNN
95.50561797752809
[[40  2]
 [ 2 45]]
Accuracy score of train KNN
94.60227272727273
```

```
Accuracy score of test KNN
95.50561797752809
[[40  2]
 [ 2 45]]
Accuracy score of train KNN
94.31818181818183
Accuracy score of test KNN
95.50561797752809
[[40  2]
 [ 2 45]]
Accuracy score of train KNN
94.0340909090909
Accuracy score of test KNN
95.50561797752809
[[40  2]
 [ 2 45]]
Accuracy score of train KNN
93.75
Accuracy score of test KNN
95.50561797752809
[[40  2]
 [ 2 45]]
Accuracy score of train KNN
92.89772727272727
Accuracy score of test KNN
95.50561797752809
[[40  2]
 [ 2 45]]
Accuracy score of train KNN
93.18181818181817
Accuracy score of test KNN
95.50561797752809
[[40  2]
 [ 2 45]]
Accuracy score of train KNN
92.89772727272727
Accuracy score of test KNN
95.50561797752809
```

[128]:

```
plt.figure(figsize=(12, 8))
plt.plot(range(1, 21),knn, marker='x', markerfacecolor='blue', color='orange', markersize=10)
plt.title('Accuracy for different  K Value')
plt.xlabel('K Value')
plt.ylabel('Accuracy')

plt.show()
```



2.0.1 KNN

[129]:
```
X = df.drop(columns=['classes'])
y = df['classes']
```

[130]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
  ↪random_state=42)
```

[131]:
```
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

[132]:
```
# Train a KNN classifier on the training data
knn_classifier = KNeighborsClassifier(n_neighbors=3)
knn_classifier.fit(X_train_scaled, y_train)
```

[132]:
```
KNeighborsClassifier(n_neighbors=3)
```

[133]:
```
y_pred = knn_classifier.predict(X_test_scaled)
```

[134]:
```
accuracy = accuracy_score(y_test, y_pred)
```

[135]:

```python
print("Model Evaluation:")
print(f"Accuracy: {accuracy:.2f}")
```

```
Model Evaluation:
Accuracy: 0.92
```

[136]:

```python
print("\nClassification Report:")
print(classification_report(y_test, y_pred))
```

```

Classification Report:
              precision    recall  f1-score   support

           0       0.93      0.90      0.92        42
           1       0.92      0.94      0.93        47

    accuracy                           0.92        89
   macro avg       0.92      0.92      0.92        89
weighted avg       0.92      0.92      0.92        89
```

[137]:

```python
print("\nConfusion Matrix:")
print(confusion_matrix(y_test, y_pred))
```

```

Confusion Matrix:
[[38  4]
 [ 3 44]]
```

[ ]:

```python

```