K-Nearest Neighbor Algorithm

In [1]:
```
#Exp no.:9
```

In [2]:
```
#Aim : Understanding K-Nearest Neighbor Algorithm
```

In [3]:
```
#Name:Vedant M.Padole
#Roll no:42
#Sec:C
#Subject:ET1
#Date:
```

Importing The Libraries

In [4]:
```
import pandas as pd
import numpy as np
```

Data Acquisition using Pandas

In [5]:
```
import os
```

In [6]:
```
os.getcwd()
```

Out[6]:
```
'C:\\Users\\DELL'
```

In [7]:
```
os.chdir('C:\\Users\\DELL\\Desktop')
```

In [8]:
```
data=pd.read_csv("heart.csv")
```

In [9]:
```
data.head()
```

Out[9]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	52	1	0	125	212	0	1	168	0	1.0	2	2	3	0
1	53	1	0	140	203	1	0	155	1	3.1	0	0	3	0
2	70	1	0	145	174	0	1	125	1	2.6	0	0	3	0
3	61	1	0	148	203	0	1	161	0	0.0	2	1	3	0
4	62	0	0	138	294	1	1	106	0	1.9	1	3	2	0

In [11]:
```
data.tail()
```

Out[11]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
1020	59	1	1	140	221	0	1	164	1	0.0	2	0	2	1
1021	60	1	0	125	258	0	0	141	1	2.8	1	1	3	0
1022	47	1	0	110	275	0	0	118	1	1.0	1	1	2	0
1023	50	0	0	110	254	0	0	159	0	0.0	2	0	2	1
1024	54	1	0	120	188	0	1	113	0	1.4	1	1	3	0

In [12]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1025 entries, 0 to 1024
Data columns (total 14 columns):
 #   Column    Non-Null Count Dtype
--- ------     -------------- -----
a0ge           1025int64 non-null
s1ex           1025int64 non-null
c2p            1025int64 non-null
t3estbps 1025 non-null int64
1025 non-null  int64   4 chol
1025 non-null  int64   5 fbs
1025 non-null  int64   6 restecg
1025 non-null  int64   7 thalach
1025 non-null  int64   8 exang
1025 non-null  float64 9 oldpeak
1025 non-null  int64   10 slope
1025 non-null  int64   11 ca
1025 non-null  int64   12 thal
1025 non-null  int64   13 target
dtypes: float64(1), int64(13)
memory usage: 112.2 KB
```

In [13]:

```
data.describe()
```

Out[13]:

	age	sex	cp	trestbps	chol	fbs	restecg	
count	1025.000000	1025.000000	1025.000000	1025.000000	1025.00000	1025.000000	1025.000000	102
mean	54.434146	0.695610	0.942439	131.611707	246.00000	0.149268	0.529756	14
std	9.072290	0.460373	1.029641	17.516718	51.59251	0.356527	0.527878	2
min	29.000000	0.000000	0.000000	94.000000	126.00000	0.000000	0.000000	7
25%	48.000000	0.000000	0.000000	120.000000	211.00000	0.000000	0.000000	13
50%	56.000000	1.000000	1.000000	130.000000	240.00000	0.000000	1.000000	15
75%	61.000000	1.000000	2.000000	140.000000	275.00000	0.000000	1.000000	16
max	77.000000	1.000000	3.000000	200.000000	564.00000	1.000000	2.000000	20

In [15]:

```
data.shape
```

Out[15]: (1025, 14)

In [16]:

```
data.size
```

Out[16]: 14350

In [17]:

```
data.ndim
```

Out[17]: 2

Data preprocessing *data cleaning* missing value treatment

In [18]:

```
# check Missing Value by record
data.isna()
```

Out[18]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal
0	False	False	False	False	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...	...	...	...	...	...
1020	False	False	False	False	False	False	False	False	False	False	False	False	False
1021	False	False	False	False	False	False	False	False	False	False	False	False	False
1022	False	False	False	False	False	False	False	False	False	False	False	False	False
1023	False	False	False	False	False	False	False	False	False	False	False	False	False
1024	False	False	False	False	False	False	False	False	False	False	False	False	False

1025 rows × 14 columns

In [19]:

```
# check Missing Value by record
data.isna().any()
```

Out[19]:

```
age         False
sex         False
cp          False
trestbps    False
chol        False
fbs         False
restecg     False
thalach     False
exang       False
oldpeak     False
slope       False
ca          False
thal        False
target      False
dtype: bool
```

Independent and Dependent Variables

In [20]:

```
x=data.drop("target", axis=1)
y=data["target"]
```

Splitting of DataSet into train and Test

```
In [21]: 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_stat
```

KNN Classifier

```
In [22]: from sklearn.neighbors import KNeighborsClassifier
         from sklearn.metrics import accuracy_score
```

```
In [23]: knn=KNeighborsClassifier()
```

```
In [24]: knn.fit(x_train, y_train)

Out[24]: KNeighborsClassifier()
```

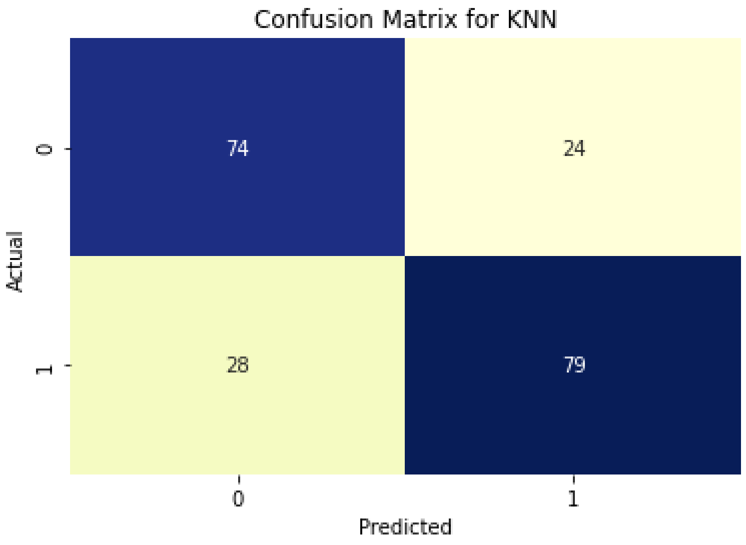
```
In [25]: y_pred2=knn.predict(x_test)
```

```
In [26]: accuracy = accuracy_score(y_test, y_pred2)
```

```
In [27]: accuracy

Out[27]: 0.7463414634146341
```

```
In [29]: import matplotlib.pyplot as plt
         import seaborn as sns
         from sklearn.metrics import accuracy_score, confusion_matrix, classification
         cm = confusion_matrix(y_test, y_pred2)
         plt.figure(figsize=(6,4))
         sns.heatmap(cm, annot=True, fmt="d", cmap="YlGnBu", cbar=False)
         plt.title("Confusion Matrix for KNN")
         plt.xlabel("Predicted")
         plt.ylabel("Actual")
         plt.show()
```

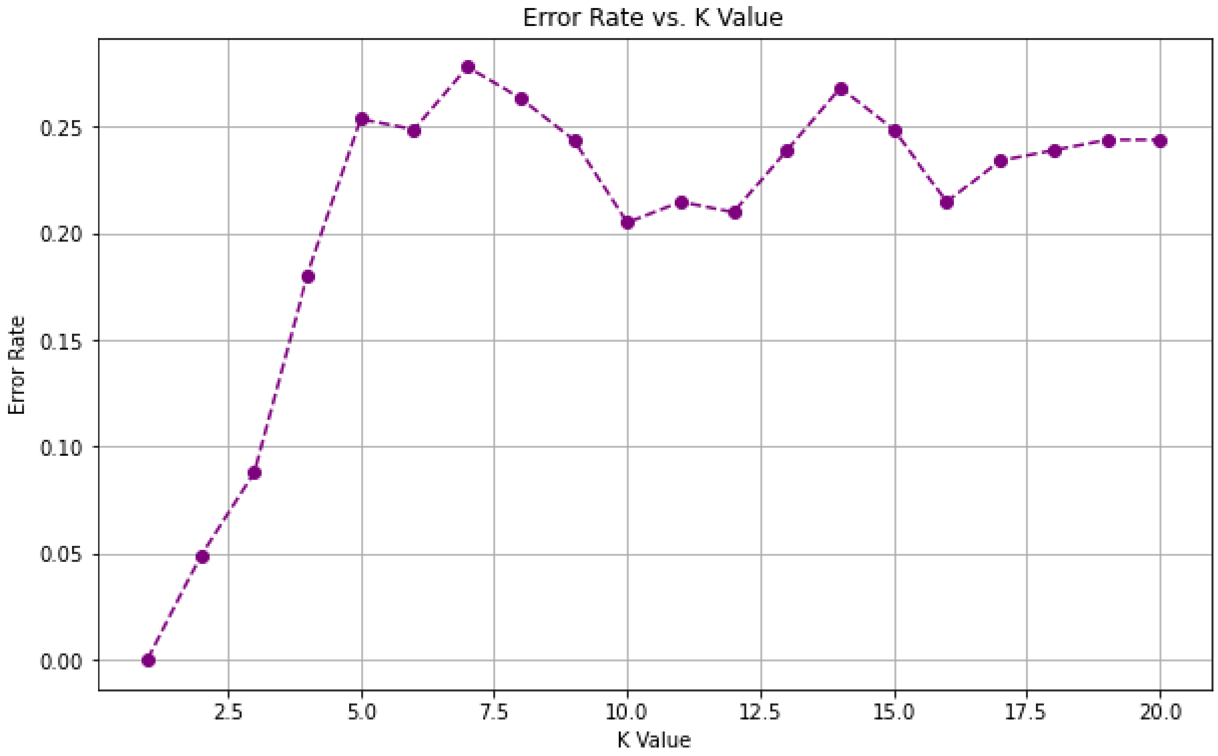


```
In [35]: from sklearn.neighbors import KNeighborsClassifier
         from sklearn.metrics import accuracy_score
         import matplotlib.pyplot as plt
```

```
error_rates = []

for k in range(1, 21):
    knn_temp = KNeighborsClassifier(n_neighbors=k)
    knn_temp.fit(x_train, y_train)
    pred_k = knn_temp.predict(x_test)
    error_rates.append(1 - accuracy_score(y_test, pred_k))

plt.figure(figsize=(10, 6))
plt.plot(range(1, 21), error_rates, marker='o', linestyle='--', color='purple')  # <
plt.title("Error Rate vs. K Value")
plt.xlabel("K Value")
plt.ylabel("Error Rate")
plt.grid(True)
plt.show()
```



Conclusion :

The experiment successfully implemented the KNearest Neighbour (KNN) algorithm, showcasing its simplicity and effectiveness in classification tasks. The analysis confirmed that the model's accuracy depends significantly on the choice of 'K' and distance metrics.

In [ ]: