Importing the required packages for the datasets

In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib import rcParams
from matplotlib.cm import rainbow
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
```

In [2]:

```
from sklearn.neighbors import KNeighborsClassifier
```

In [3]:

```
df=pd.read_csv('heart.csv') # Reading the data files
df
```

Out[3]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal
0	52	1	0	125	212	0	1	168	0	1.0	2	2	3
1	53	1	0	140	203	1	0	155	1	3.1	0	0	3
2	70	1	0	145	174	0	1	125	1	2.6	0	0	3
3	61	1	0	148	203	0	1	161	0	0.0	2	1	3
4	62	0	0	138	294	1	1	106	0	1.9	1	3	2
...	...	...	...	...	...	...	...	...	...	...	...	...	...
1020	59	1	1	140	221	0	1	164	1	0.0	2	0	2
1021	60	1	0	125	258	0	0	141	1	2.8	1	1	3
1022	47	1	0	110	275	0	0	118	1	1.0	1	1	2
1023	50	0	0	110	254	0	0	159	0	0.0	2	0	2
1024	54	1	0	120	188	0	1	113	0	1.4	1	1	3

1025 rows × 14 columns

In [4]:

```
df.info() # finding the index,data-type,& memory information
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1025 entries, 0 to 1024
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   age       1025 non-null   int64  
 1   sex       1025 non-null   int64  
 2   cp        1025 non-null   int64  
 3   trestbps  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   int64  
 9   oldpeak   1025 non-null   float64
 10  slope     1025 non-null   int64  
 11  ca        1025 non-null   int64  
 12  thal      1025 non-null   int64  
 13  target    1025 non-null   int64  
dtypes: float64(1), int64(13)
memory usage: 112.2 KB
```

Data Processing

In [5]:

```
df=pd.get_dummies(df,columns=['sex','cp','fbs','restecg','exang','slope','ca','thal'])
```

In [6]:

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler #transforming the data to understand i
t in a better way.
standardScaler=StandardScaler()
columns_to_scale=['age','trestbps','chol','thalach','oldpeak']
df[columns_to_scale]=standardScaler.fit_transform(df[columns_to_scale])
```

In [7]:

```
df.head()
```

Out[7]:

	age	trestbps	chol	thalach	oldpeak	target	sex_0	sex_1	cp_0	cp_1	.
0	-0.268437	-0.377636	-0.659332	0.821321	-0.060888	0	0	1	1	0	.
1	-0.158157	0.479107	-0.833861	0.255968	1.727137	0	0	1	1	0	.
2	1.716595	0.764688	-1.396233	-1.048692	1.301417	0	0	1	1	0	.
3	0.724079	0.936037	-0.833861	0.516900	-0.912329	0	0	1	1	0	.
4	0.834359	0.364875	0.930822	-1.874977	0.705408	0	1	0	1	0	.

5 rows × 31 columns

In [8]:

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

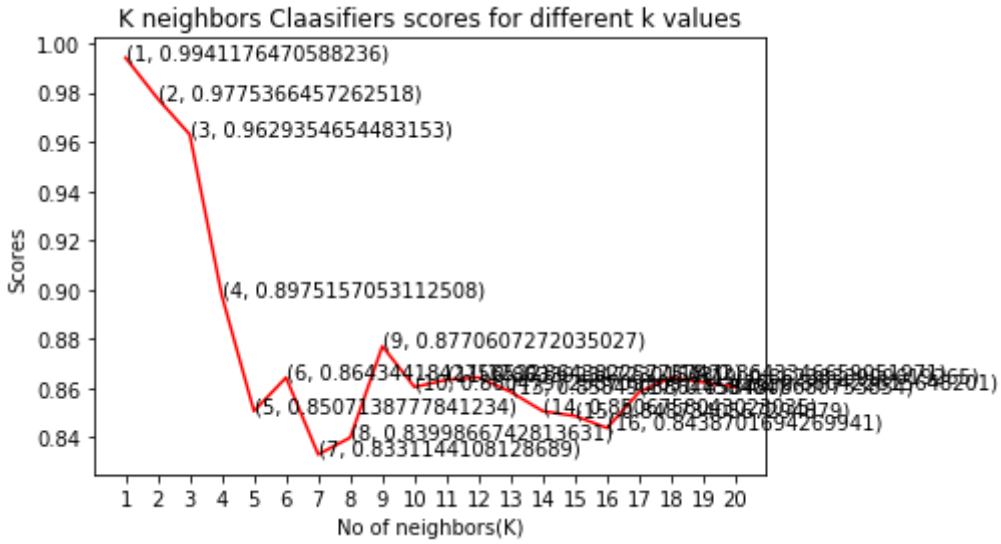
In [9]:

```
from sklearn.model_selection import cross_val_score #using cross validation in Knn_scor
es and getting the perfect value.
knn_scores=[]
for k in range(1,21):
    knn_classifier=KNeighborsClassifier(n_neighbors=k)
    score=cross_val_score(knn_classifier,x,y,cv=10)
    knn_scores.append(score.mean())
```

Data Visualization process

In [10]:

```
plt.plot([k for k in range(1,21)],knn_scores,color='red')
for i in range(1,21):
    plt.text(i,knn_scores[i-1],(i,knn_scores[i-1]))
    plt.xticks([i for i in range(1,21)])
    plt.xlabel('No of neighbors(K)')
    plt.ylabel('Scores')
    plt.title('K neighbors Claasifiers scores for different k values')
```



In [11]:

```
knn_classfier=KNeighborsClassifier(n_neighbors=1) #from the above visualization K neigh
bors highest value is at k=1
score=cross_val_score(knn_classfier,x,y,cv=10)
```

In [12]:

```
score.mean()
```

Out[12]:

```
0.9941176470588236
```

Conculsion: So we can conclude by using K-neighbors classfier,cross-validation,& other standard techniques for the datasets that this model is 99% accurate.