In [1]:
```
import pandas as pd
```

In [2]:
```
df = pd.read_csv("https://raw.githubusercontent.com/patrickmlong/Breast-Cancer-Wisconsin-Diagnostic-DataSet/master/data.csv")
```

In [3]:
```
df.head(25)
```

Out[3]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean
0	842302	M	17.990	10.38	122.80	1001.0	0.11840	0.27760
1	842517	M	20.570	17.77	132.90	1326.0	0.08474	0.07864
2	84300903	M	19.690	21.25	130.00	1203.0	0.10960	0.15990
3	84348301	M	11.420	20.38	77.58	386.1	0.14250	0.28390
4	84358402	M	20.290	14.34	135.10	1297.0	0.10030	0.13280
5	843786	M	12.450	15.70	82.57	477.1	0.12780	0.17000
6	844359	M	18.250	19.98	119.60	1040.0	0.09463	0.10900
7	84458202	M	13.710	20.83	90.20	577.9	0.11890	0.16450
8	844981	M	13.000	21.82	87.50	519.8	0.12730	0.19320
9	84501001	M	12.460	24.04	83.97	475.9	0.11860	0.23960
10	845636	M	16.020	23.24	102.70	797.8	0.08206	0.06669
11	84610002	M	15.780	17.89	103.60	781.0	0.09710	0.12920
12	846226	M	19.170	24.80	132.40	1123.0	0.09740	0.24580
13	846381	M	15.850	23.95	103.70	782.7	0.08401	0.10020
14	84667401	M	13.730	22.61	93.60	578.3	0.11310	0.22930
15	84799002	M	14.540	27.54	96.73	658.8	0.11390	0.15950
16	848406	M	14.680	20.13	94.74	684.5	0.09867	0.07200
17	84862001	M	16.130	20.68	108.10	798.8	0.11700	0.20220
18	849014	M	19.810	22.15	130.00	1260.0	0.09831	0.10270
19	8510426	B	13.540	14.36	87.46	566.3	0.09779	0.08129
20	8510653	B	13.080	15.71	85.63	520.0	0.10750	0.12700
21	8510824	B	9.504	12.44	60.34	273.9	0.10240	0.06492
22	8511133	M	15.340	14.26	102.50	704.4	0.10730	0.21350
23	851509	M	21.160	23.04	137.20	1404.0	0.09428	0.10220
24	852552	M	16.650	21.38	110.00	904.6	0.11210	0.14570

25 rows × 33 columns

In [4]:
```
df.head()
```

Out[4]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	c
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990	
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	0.28390	
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.13280	

5 rows × 33 columns

In [5]:
```
y = df['diagnosis']
```

In [6]:
```
y.head()
```

Out[6]:
```
0    M
1    M
2    M
3    M
4    M
Name: diagnosis, dtype: object
```

In [7]:
```
x = df.drop(['id','Unnamed: 32','diagnosis'], axis=1)
```

In [8]:
```
x.head()
```

Out[8]:

	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	co points
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0

5 rows × 30 columns

In [9]:
```
from sklearn.preprocessing import LabelEncoder
```

In [10]:
```
le = LabelEncoder()
```

In [11]:
```
le.fit(y)
```

Out[11]:
```
LabelEncoder()
```

In [12]:
```
le.classes_
```

Out[12]:
```
array(['B', 'M'], dtype=object)
```

In [13]:
```
S = le.transform(y)
```

In [14]:
```
y = S
```

In [15]:
```
y
```

Out[15]:
```
array([1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1,
       1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 1,
       0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 0, 0, 1, 1, 0, 1, 1, 1,
       0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0,
       0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 1, 0, 0, 0, 1, 1, 0, 1, 0, 1,
       1, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0,
       0, 1, 0, 0, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 1,
       1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 1, 0, 0, 1, 1,
       0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 0,
       0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 1,
       1, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1,
       1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1,
       0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0,
       0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0,
       0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1,
       0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0,
       0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0,
       0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0])
```

In [17]:
```
from sklearn.model_selection import train_test_split
```

In [18]:
```
xTrain, xTest, yTrain, yTest = train_test_split(x, y, test_size = 0.2, random_state = 5)
```

In [19]:
```
from sklearn.neighbors import KNeighborsClassifier
```

In [20]:
```
knn = KNeighborsClassifier(n_neighbors=3)
```

In [21]:
```
knn.fit(xTrain,yTrain)
```

Out[21]:
```
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
           metric_params=None, n_jobs=None, n_neighbors=3, p=2,
           weights='uniform')
```

In [22]:
```
knn.score(xTest,yTest)
```

Out[22]:
```
0.9210526315789473
```

In [24]:
```
from sklearn.svm import SVC
svclassifier = SVC(kernel='linear')
svclassifier.fit(xTrain, yTrain)
```

Out[24]:
```
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape='ovr', degree=3, gamma='auto_deprecated',
  kernel='linear', max_iter=-1, probability=False, random_state=None,
  shrinking=True, tol=0.001, verbose=False)
```

In [25]:
```
svclassifier.score(xTest,yTest)
```

Out[25]:
```
0.9736842105263158
```

In [28]:
```
svcc = SVC(kernel='rbf')
svcc.fit(xTrain, yTrain)
```

```
C:\Users\RONY\Anaconda3\lib\site-packages\sklearn\svm\base.py:196: FutureWarning: The default va
lue of gamma will change from 'auto' to 'scale' in version 0.22 to account better for unscaled f
eatures. Set gamma explicitly to 'auto' or 'scale' to avoid this warning.
  "avoid this warning.", FutureWarning)
```

Out[28]:
```
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape='ovr', degree=3, gamma='auto_deprecated',
  kernel='rbf', max_iter=-1, probability=False, random_state=None,
  shrinking=True, tol=0.001, verbose=False)
```

In [29]:
```
svcc.score(xTest,yTest)
```

Out[29]:
```
0.5789473684210527
```

In [30]:
```
#Shift + Double Tab for details
```

In [ ]:
```

```