In [49]:

```
import pandas as pd
```

In [50]:

```
df=pd.read_csv("cancer.csv")
df.head()
```

Out[50]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave points_mean	...	texture_worst	perimeter_worst	area_worst
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	...	17.33	184.60	2019.0
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	...	23.41	158.80	1956.0
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	...	25.53	152.50	1709.0
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	...	26.50	98.87	567.7
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	...	16.67	152.20	1575.0

5 rows × 33 columns

In [65]:

```
df.shape
```

Out[65]:

```
(569, 33)
```

In [3]:

```
df.isnull().sum()
```

Out[3]:

```
id                           0
diagnosis                    0
radius_mean                  0
texture_mean                 0
perimeter_mean               0
area_mean                    0
smoothness_mean              0
compactness_mean             0
concavity_mean               0
concave points_mean          0
symmetry_mean                0
fractal_dimension_mean       0
radius_se                    0
texture_se                   0
perimeter_se                 0
area_se                      0
smoothness_se                0
compactness_se               0
concavity_se                 0
concave points_se            0
symmetry_se                  0
fractal_dimension_se         0
radius_worst                 0
texture_worst                0
perimeter_worst              0
area_worst                   0
smoothness_worst             0
compactness_worst            0
concavity_worst              0
concave points_worst         0
symmetry_worst               0
fractal_dimension_worst      0
Unnamed: 32                569
dtype: int64
```

In [7]:

```
df.diagnosis.value_counts()
```

Out[7]:

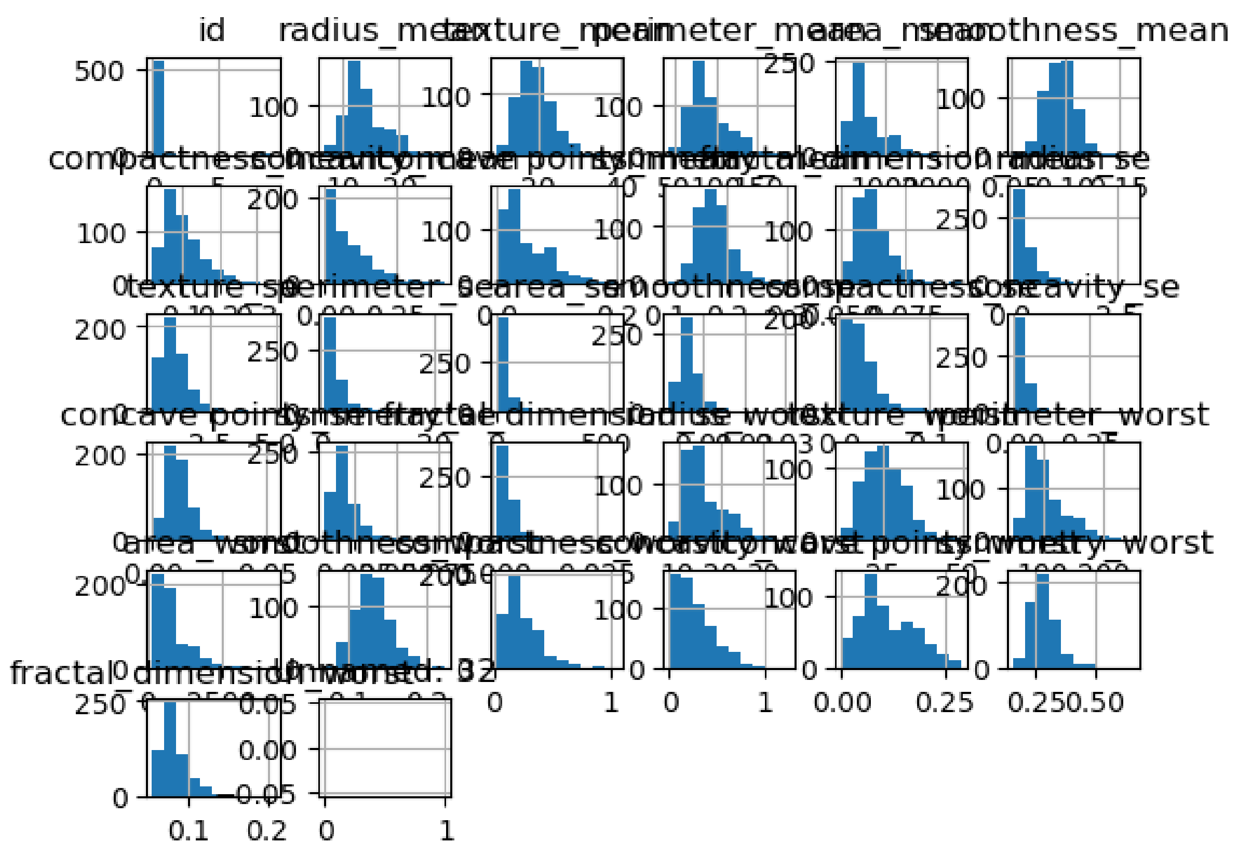
```
B    357
M    212
Name: diagnosis, dtype: int64
```

In [9]:

```
df.hist()
```

Out[9]:

```
array([[<AxesSubplot:title={'center':'id'}>,
        <AxesSubplot:title={'center':'radius_mean'}>,
        <AxesSubplot:title={'center':'texture_mean'}>,
        <AxesSubplot:title={'center':'perimeter_mean'}>,
        <AxesSubplot:title={'center':'area_mean'}>,
        <AxesSubplot:title={'center':'smoothness_mean'}>],
       [<AxesSubplot:title={'center':'compactness_mean'}>,
        <AxesSubplot:title={'center':'concavity_mean'}>,
        <AxesSubplot:title={'center':'concave points_mean'}>,
        <AxesSubplot:title={'center':'symmetry_mean'}>,
        <AxesSubplot:title={'center':'fractal_dimension_mean'}>,
        <AxesSubplot:title={'center':'radius_se'}>],
       [<AxesSubplot:title={'center':'texture_se'}>,
        <AxesSubplot:title={'center':'perimeter_se'}>,
        <AxesSubplot:title={'center':'area_se'}>,
        <AxesSubplot:title={'center':'smoothness_se'}>,
        <AxesSubplot:title={'center':'compactness_se'}>,
        <AxesSubplot:title={'center':'concavity_se'}>],
       [<AxesSubplot:title={'center':'concave points_se'}>,
        <AxesSubplot:title={'center':'symmetry_se'}>,
        <AxesSubplot:title={'center':'fractal_dimension_se'}>,
        <AxesSubplot:title={'center':'radius_worst'}>,
        <AxesSubplot:title={'center':'texture_worst'}>,
        <AxesSubplot:title={'center':'perimeter_worst'}>],
       [<AxesSubplot:title={'center':'area_worst'}>,
        <AxesSubplot:title={'center':'smoothness_worst'}>,
        <AxesSubplot:title={'center':'compactness_worst'}>,
        <AxesSubplot:title={'center':'concavity_worst'}>,
        <AxesSubplot:title={'center':'concave points_worst'}>,
        <AxesSubplot:title={'center':'symmetry_worst'}>],
       [<AxesSubplot:title={'center':'fractal_dimension_worst'}>,
        <AxesSubplot:title={'center':'Unnamed: 32'}>, <AxesSubplot:>,
        <AxesSubplot:>, <AxesSubplot:>, <AxesSubplot:>]], dtype=object)
```



In [52]:

```
x=df.iloc[:, 2:32].values
y=df.diagnosis
```

In [106…

```
from sklearn.preprocessing import StandardScaler  #scaling
Scaler=StandardScaler()
x_Scaled=Scaler.fit_transform(x)
x_Scaled
```

Out[106]:

```
array([[ 1.09706398, -2.07333501,  1.26993369, ...,  2.29607613,
         2.75062224,  1.93701461],
       [ 1.82982061, -0.35363241,  1.68595471, ...,  1.0870843 ,
        -0.24388967,  0.28118999],
       [ 1.57988811,  0.45618695,  1.56650313, ...,  1.95500035,
         1.152255  ,  0.20139121],
       ...,
       [ 0.70228425,  2.0455738 ,  0.67267578, ...,  0.41406869,
        -1.10454895, -0.31840916],
       [ 1.83834103,  2.33645719,  1.98252415, ...,  2.28998549,
         1.91908301,  2.21963528],
       [-1.80840125,  1.22179204, -1.81438851, ..., -1.74506282,
        -0.04813821, -0.75120669]])
```

In [ ]:

In [47]:

```
x_train.shape
```

Out[47]:

```
(426, 30)
```

In [22]:

```
y_train.shape
```

Out[22]:

```
(426,)
```

In [23]:

```
x_test.shape
```

Out[23]:

```
(143, 30)
```

In [24]:

```
y_test.shape
```

Out[24]:

```
(143,)
```

In [25]:

```
y_train.value_counts()
```

Out[25]:

```
0    266
1    160
Name: diagnosis, dtype: int64
```

In [74]:

```
#using kernal=rbf
from sklearn.svm import SVC
model=SVC()
model.fit(x_train,y_train)
```

Out[74]:

```
SVC()
```

In [75]:

```
model.score(x_test,y_test)
```

Out[75]:

```
0.9790209790209791
```

In [77]:

```
model.predict(x_test)
```

Out[77]:

```
array([1, 0, 0, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 1,
       0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
       0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0,
       0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 1,
       0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0,
       0, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0,
       1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1], dtype=int64)
```

In [104…

```
#using kernal=linear
from sklearn.svm import SVC
model=SVC(kernel='linear')
model.fit(x_train,y_train)
```

Out[104]:

```
SVC(kernel='linear')
```

In [105…

```
model.score(x_test,y_test)
```

Out[105]:

```
0.9370629370629371
```

In [101…

```
#using Decision Tree
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import cross_val_score
scores=cross_val_score(DecisionTreeClassifier(),x,y,cv=6)
```

In [102…

```
scores.mean()
```

Out[102]:

```
0.9173572228443448
```

In [62]:

```
#using bagging classifier
bag_model=BaggingClassifier(
            base_estimator=DecisionTreeClassifier(),
            n_estimators=100,
            max_samples=0.8,
            oob_score=True,
            random_state=0
        )
scores=cross_val_score(bag_model,x,y,cv=5)
scores.mean()
```

Out[62]:

```
0.9578636857630801
```

In [ ]:

In [ ]: