```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.neighbors import KNeighborsClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import mean_absolute_percentage_error,r2_score
df=pd.read_csv('/content/drive/MyDrive/project_files/breast_cancer.csv')
df
```

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean
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
...	...	...	...	...	...	...	...	...
564	926424	M	21.56	22.39	142.00	1479.0	0.11100	0.11590
565	926682	M	20.13	28.25	131.20	1261.0	0.09780	0.10340
566	926954	M	16.60	28.08	108.30	858.1	0.08455	0.10230
567	927241	M	20.60	29.33	140.10	1265.0	0.11780	0.27700
568	92751	B	7.76	24.54	47.92	181.0	0.05263	0.04362

569 rows × 33 columns

```
df.head()
```

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	cc
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

```
df.tail()
```

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	cc
564	926424	M	21.56	22.39	142.00	1479.0	0.11100	0.11590	
565	926682	M	20.13	28.25	131.20	1261.0	0.09780	0.10340	
566	926954	M	16.60	28.08	108.30	858.1	0.08455	0.10230	
567	927241	M	20.60	29.33	140.10	1265.0	0.11780	0.27700	
568	92751	B	7.76	24.54	47.92	181.0	0.05263	0.04362	

5 rows × 33 columns

```
df.columns
```

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

```
df.shape
```

```
(569, 33)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 33 columns):
 #   Column                   Non-Null Count  Dtype  
---  ------                   --------------  -----  
 0   id                       569 non-null    int64  
 1   diagnosis                569 non-null    object 
 2   radius_mean              569 non-null    float64
 3   texture_mean             569 non-null    float64
 4   perimeter_mean           569 non-null    float64
 5   area_mean                569 non-null    float64
 6   smoothness_mean          569 non-null    float64
 7   compactness_mean         569 non-null    float64
 8   concavity_mean           569 non-null    float64
 9   concave points_mean      569 non-null    float64
 10  symmetry_mean            569 non-null    float64
 11  fractal_dimension_mean   569 non-null    float64
 12  radius_se                569 non-null    float64
 13  texture_se               569 non-null    float64
 14  perimeter_se             569 non-null    float64
 15  area_se                  569 non-null    float64
 16  smoothness_se            569 non-null    float64
 17  compactness_se           569 non-null    float64
 18  concavity_se             569 non-null    float64
 19  concave points_se        569 non-null    float64
 20  symmetry_se              569 non-null    float64
 21  fractal_dimension_se     569 non-null    float64
 22  radius_worst             569 non-null    float64
 23  texture_worst            569 non-null    float64
 24  perimeter_worst          569 non-null    float64
 25  area_worst               569 non-null    float64
 26  smoothness_worst         569 non-null    float64
 27  compactness_worst        569 non-null    float64
 28  concavity_worst          569 non-null    float64
 29  concave points_worst     569 non-null    float64
 30  symmetry_worst           569 non-null    float64
 31  fractal_dimension_worst  569 non-null    float64
 32  Unnamed: 32              0 non-null      float64
dtypes: float64(31), int64(1), object(1)
memory usage: 146.8+ KB
```

Data cleaning

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

```
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
```

```
df.dtypes
```

```
id                         int64
diagnosis                 object
radius_mean              float64
texture_mean             float64
perimeter_mean           float64
area_mean                float64
smoothness_mean          float64
compactness_mean         float64
concavity_mean           float64
concave points_mean      float64
symmetry_mean            float64
fractal_dimension_mean   float64
radius_se                float64
texture_se               float64
perimeter_se             float64
area_se                  float64
smoothness_se            float64
compactness_se           float64
concavity_se             float64
concave points_se        float64
symmetry_se              float64
fractal_dimension_se     float64
radius_worst             float64
texture_worst            float64
perimeter_worst          float64
area_worst               float64
smoothness_worst         float64
compactness_worst        float64
concavity_worst          float64
concave points_worst     float64
symmetry_worst           float64
fractal_dimension_worst  float64
Unnamed: 32              float64
dtype: object
```

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

```
0
```

```
df.drop(['id','Unnamed: 32'],axis=1,inplace=True)
df.head()
```

	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_m
0	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3
1	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0
2	M	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1
3	M	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2
4	M	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1

5 rows × 31 columns

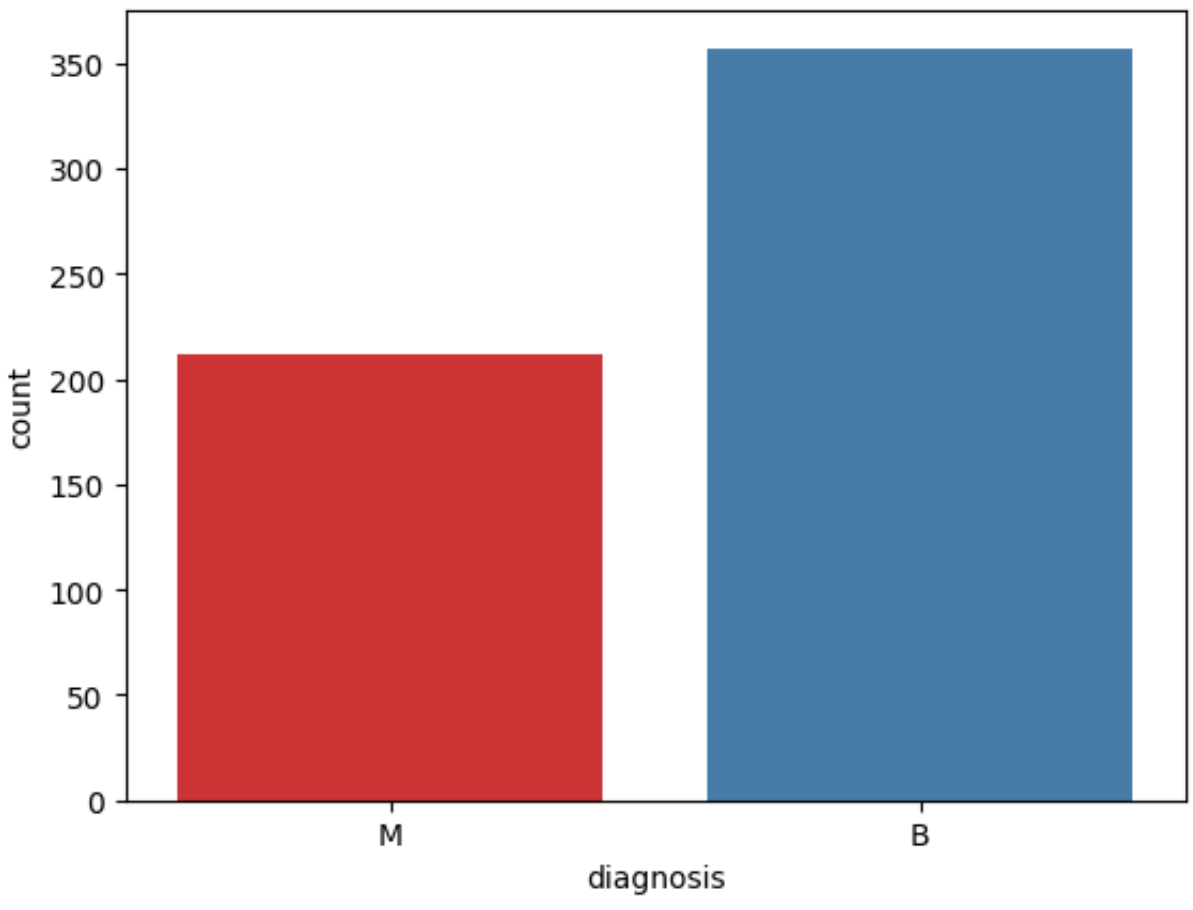
Visualization

```
diagnosis=df['diagnosis'].value_counts().reset_index()
diagnosis
```

	diagnosis	count
0	B	357
1	M	212

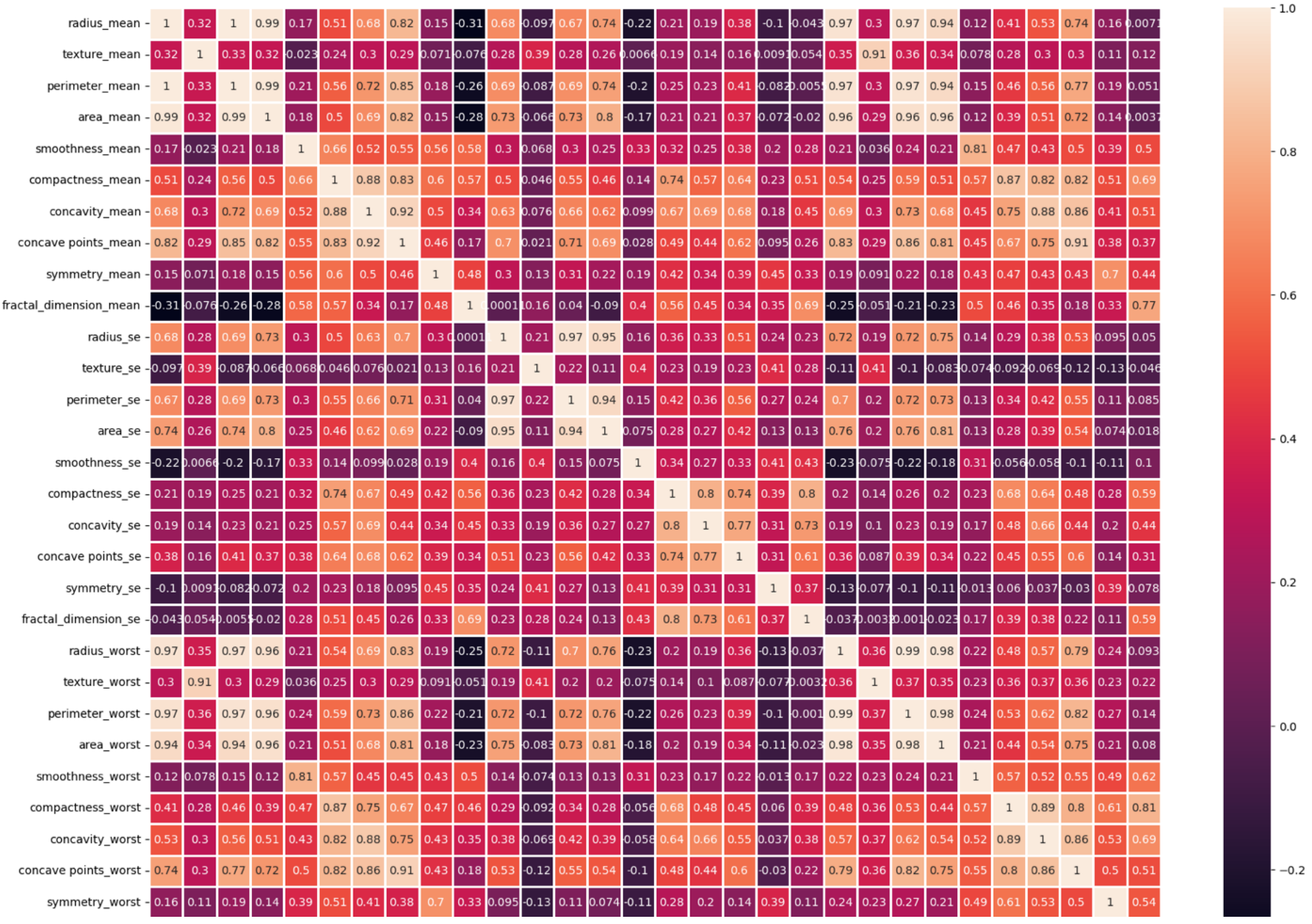
```
sns.countplot(x='diagnosis',data=df,palette='Set1')
plt.show()
```

```
<ipython-input-13-575777ea0a19>:1: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable

  sns.countplot(x='diagnosis',data=df,palette='Set1')
```



```
plt.figure(figsize=(20,15))
sns.heatmap(df.corr(numeric_only=True),annot=True,linewidth=1)
```

```
<Axes: >
```



```
le=LabelEncoder()
df['diagnosis']=le.fit_transform(df['diagnosis'])
df.dtypes
```

```
diagnosis                  int64
radius_mean              float64
texture_mean             float64
perimeter_mean           float64
area_mean                float64
smoothness_mean          float64
compactness_mean         float64
concavity_mean           float64
concave points_mean      float64
symmetry_mean            float64
fractal_dimension_mean   float64
radius_se                float64
texture_se               float64
perimeter_se             float64
area_se                  float64
smoothness_se            float64
compactness_se           float64
concavity_se             float64
concave points_se        float64
symmetry_se              float64
fractal_dimension_se     float64
radius_worst             float64
texture_worst            float64
perimeter_worst          float64
area_worst               float64
smoothness_worst         float64
compactness_worst        float64
concavity_worst          float64
concave points_worst     float64
symmetry_worst           float64
fractal_dimension_worst  float64
dtype: object
```

```
x=df.drop(columns='diagnosis',axis=1)
x
```

	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	poin
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.30010	
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.08690	
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.19740	
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.24140	
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.19800	
...	...	...	...	...	...	...	...	
564	21.56	22.39	142.00	1479.0	0.11100	0.11590	0.24390	
565	20.13	28.25	131.20	1261.0	0.09780	0.10340	0.14400	
566	16.60	28.08	108.30	858.1	0.08455	0.10230	0.09251	
567	20.60	29.33	140.10	1265.0	0.11780	0.27700	0.35140	
568	7.76	24.54	47.92	181.0	0.05263	0.04362	0.00000	

569 rows × 30 columns

```
y=df['diagnosis']
y
```

```
0      1
1      1
2      1
3      1
4      1
      ..
564    1
565    1
566    1
567    1
568    0
Name: diagnosis, Length: 569, dtype: int64
```

Train Test Split

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.30,random_state=42)
x_train
```

	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	poin
149	13.740	17.91	88.12	585.0	0.07944	0.06376	0.02881	
124	13.370	16.39	86.10	553.5	0.07115	0.07325	0.08092	
421	14.690	13.98	98.22	656.1	0.10310	0.18360	0.14500	
195	12.910	16.33	82.53	516.4	0.07941	0.05366	0.03873	
545	13.620	23.23	87.19	573.2	0.09246	0.06747	0.02974	
...	...	...	...	...	...	...	...	
71	8.888	14.64	58.79	244.0	0.09783	0.15310	0.08606	
106	11.640	18.33	75.17	412.5	0.11420	0.10170	0.07070	
270	14.290	16.82	90.30	632.6	0.06429	0.02675	0.00725	
435	13.980	19.62	91.12	599.5	0.10600	0.11330	0.11260	
102	12.180	20.52	77.22	458.7	0.08013	0.04038	0.02383	

398 rows × 30 columns

```
x_test
```

	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	poin
204	12.470	18.60	81.09	481.9	0.09965	0.10580	0.08005	
70	18.940	21.31	123.60	1130.0	0.09009	0.10290	0.10800	
131	15.460	19.48	101.70	748.9	0.10920	0.12230	0.14660	
431	12.400	17.68	81.47	467.8	0.10540	0.13160	0.07741	
540	11.540	14.44	74.65	402.9	0.09984	0.11200	0.06737	
...	...	...	...	...	...	...	...	
69	12.780	16.49	81.37	502.5	0.09831	0.05234	0.03653	
542	14.740	25.42	94.70	668.6	0.08275	0.07214	0.04105	
176	9.904	18.06	64.60	302.4	0.09699	0.12940	0.13070	
501	13.820	24.49	92.33	595.9	0.11620	0.16810	0.13570	
247	12.890	14.11	84.95	512.2	0.08760	0.13460	0.13740	

171 rows × 30 columns

```
y_train
```

```
149    0
124    0
421    0
195    0
545    0
      ..
71     0
106    0
270    0
435    1
102    0
Name: diagnosis, Length: 398, dtype: int64
```

```
y_test
```

```
204    0
70     1
131    1
431    0
540    0
      ..
69     0
542    0
176    0
501    1
247    0
Name: diagnosis, Length: 171, dtype: int64
```

Normalization

```
scaler=StandardScaler()
x_train=scaler.fit_transform(x_train)
x_test=scaler.fit_transform(x_test)
x_test
```

```
array([[-0.44180872, -0.22163893, -0.42004481, ..., -0.21335976,
         0.11865301,  0.13539217],
       [ 1.40986153,  0.38303372,  1.32877902, ...,  0.89553559,
        -0.64528246, -0.98835798],
       [ 0.41390906, -0.02528766,  0.42783237, ...,  0.50154822,
        -0.17339144, -0.24473803],
       ...,
       [-1.17618057, -0.34212721, -1.09842884, ..., -0.24774412,
        -0.54133444,  1.62783221],
       [-0.05544785,  1.09257579,  0.04235885, ...,  0.51157699,
         1.16968301,  1.73703514],
       [-0.32160756, -1.22347664, -0.26124782, ...,  0.56888424,
        -0.50008522,  1.71103444]])
```

```
x_train
```

```
array([[-0.12348985, -0.29680142, -0.17050713, ..., -0.84082156,
        -0.8563616 , -0.76574773],
       [-0.22826757, -0.65795149, -0.25377521, ..., -0.37706655,
        -1.3415819 , -0.41480748],
       [ 0.14553402, -1.23056444,  0.24583328, ..., -0.04762652,
        -0.08997059,  0.4882635 ],
       ...,
       [ 0.03226081, -0.55578404, -0.08064356, ..., -1.26179013,
        -0.6828391 , -1.27672587],
       [-0.05552593,  0.10949242, -0.04684166, ...,  1.07924018,
         0.4755842 ,  1.25530227],
       [-0.56525537,  0.32333128, -0.619825  , ..., -0.61952313,
        -0.30366032, -0.84348042]])
```

Model Creation

K-Nearest Neighbors (KNN)

```
knn=KNeighborsClassifier(n_neighbors=7)
knn.fit(x_train,y_train)
y_pred=knn.predict(x_test)
y_pred
```

```
array([0, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0,
       1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0,
       0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0,
       1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1,
       0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0, 0, 1, 0, 0,
       1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 1,
       0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 0, 0,
       0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0])
```

```
y_test
```

```
204    0
70     1
131    1
431    0
540    0
      ..
69     0
542    0
176    0
501    1
247    0
Name: diagnosis, Length: 171, dtype: int64
```

```
cm=confusion_matrix(y_test,y_pred)
cm
```

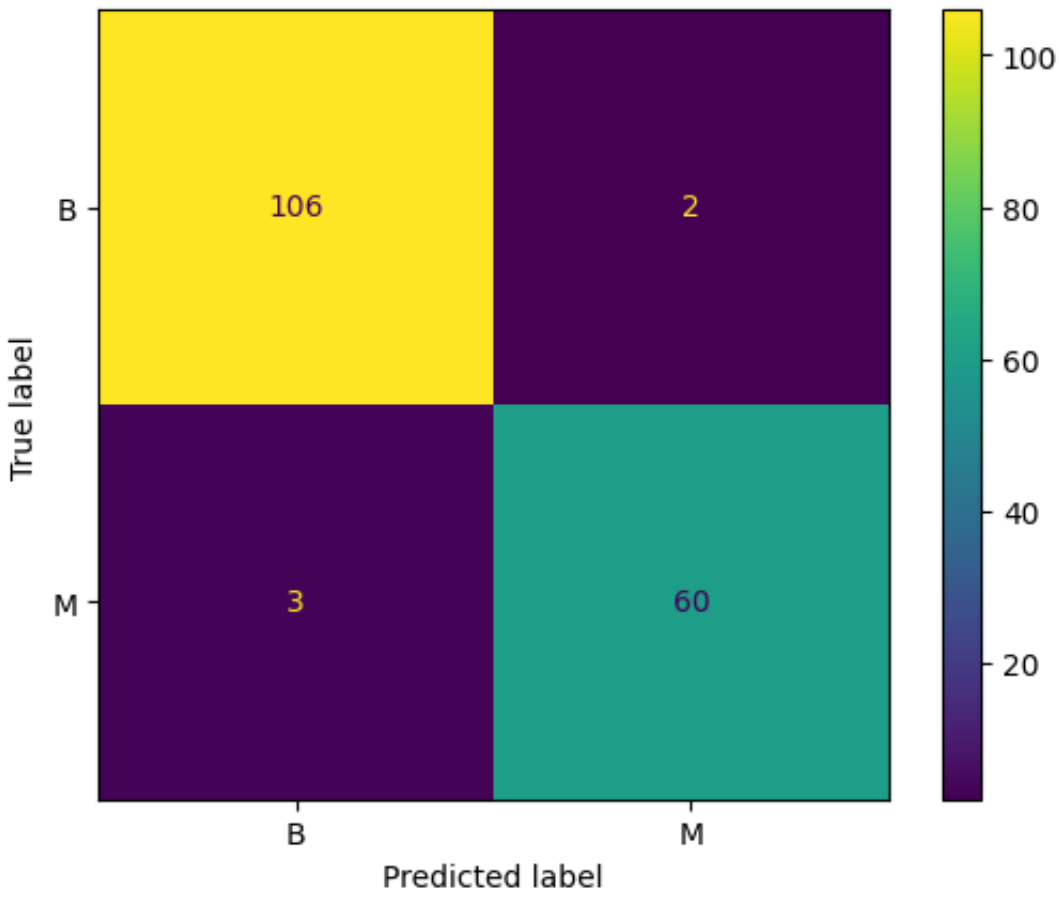
```
array([[106,   2],
       [  3,  60]])
```

```
score1=accuracy_score(y_test,y_pred)
score1
```

```
0.9707602339181286
```

```
from sklearn.metrics import ConfusionMatrixDisplay
labels=['B','M']
cmd=ConfusionMatrixDisplay(cm,display_labels=labels)
cmd.plot()
```

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7f5922115990>
```




```
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.97      0.98      0.98       108
           1       0.97      0.95      0.96        63

    accuracy                           0.97       171
   macro avg       0.97      0.97      0.97       171
weighted avg       0.97      0.97      0.97       171
```

Random Forest Classifier

```
RF=RandomForestClassifier()
RF.fit(x_train,y_train)
y_pred=RF.predict(x_test)
y_pred
```

```
array([0, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0,
       1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0,
       0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0,
       1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 1, 1, 1,
       0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0, 0, 1, 0, 0,
       1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 1,
       0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 0, 0,
       0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0])
```

```
y_test
```

```
204    0
70     1
131    1
431    0
540    0
      ..
69     0
542    0
176    0
501    1
247    0
Name: diagnosis, Length: 171, dtype: int64
```

```
cm=confusion_matrix(y_test,y_pred)
cm
```

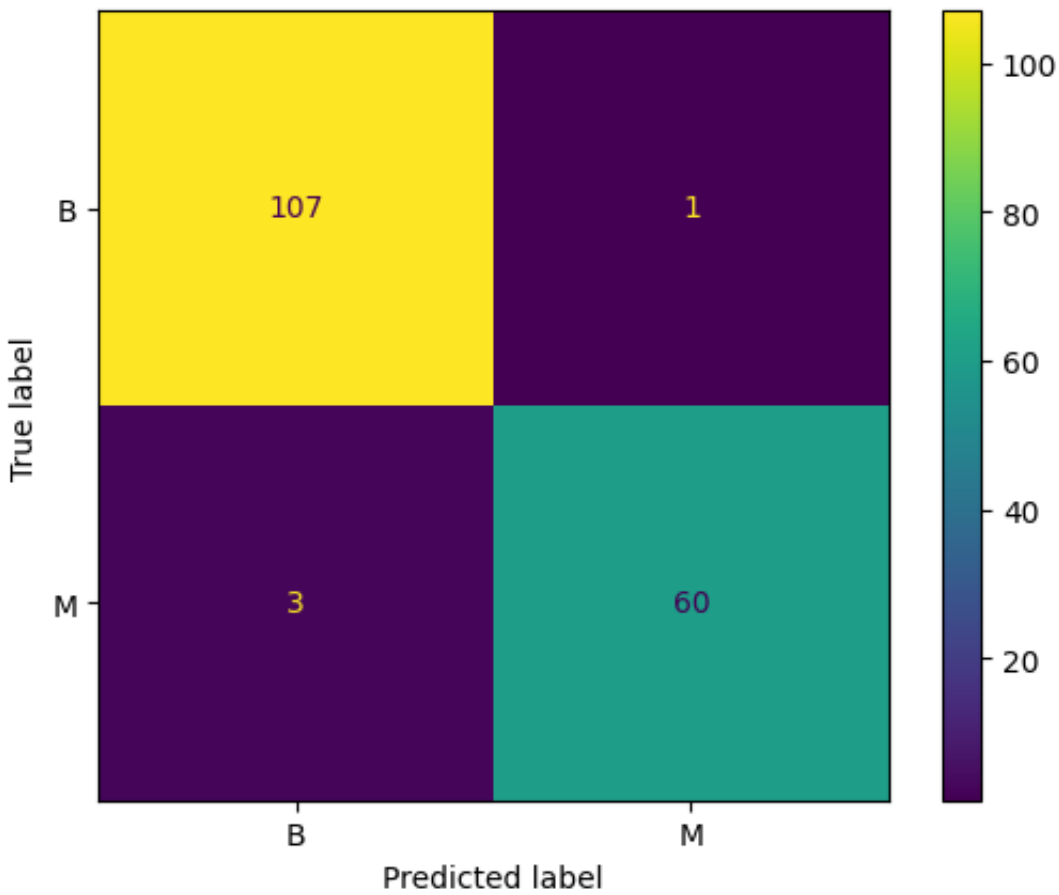
```
array([[107,   1],
       [  3,  60]])
```

```
score2=accuracy_score(y_test,y_pred)
score2
```

```
0.9766081871345029
```

```
labels=['B','M']
cmd=ConfusionMatrixDisplay(cm,display_labels=labels)
cmd.plot()
```

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7f5922010ca0>
```



```
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.97      0.99      0.98       108
           1       0.98      0.95      0.97        63

    accuracy                           0.98       171
   macro avg       0.98      0.97      0.97       171
weighted avg       0.98      0.98      0.98       171
```

Decision Tree Classifier

```
dt=DecisionTreeClassifier(criterion='entropy')
dt.fit(x_train,y_train)
y_pred=dt.predict(x_test)
y_pred
```

```
array([0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0,
       1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0,
       0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0,
       1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 1, 1, 1,
       0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0,
       1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 1,
       0, 0, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 0, 0,
       0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0])
```

```
y_test
```

```
204    0
70     1
131    1
431    0
540    0
      ..
69     0
542    0
176    0
501    1
247    0
Name: diagnosis, Length: 171, dtype: int64
```

```
cm=confusion_matrix(y_test,y_pred)
cm
```

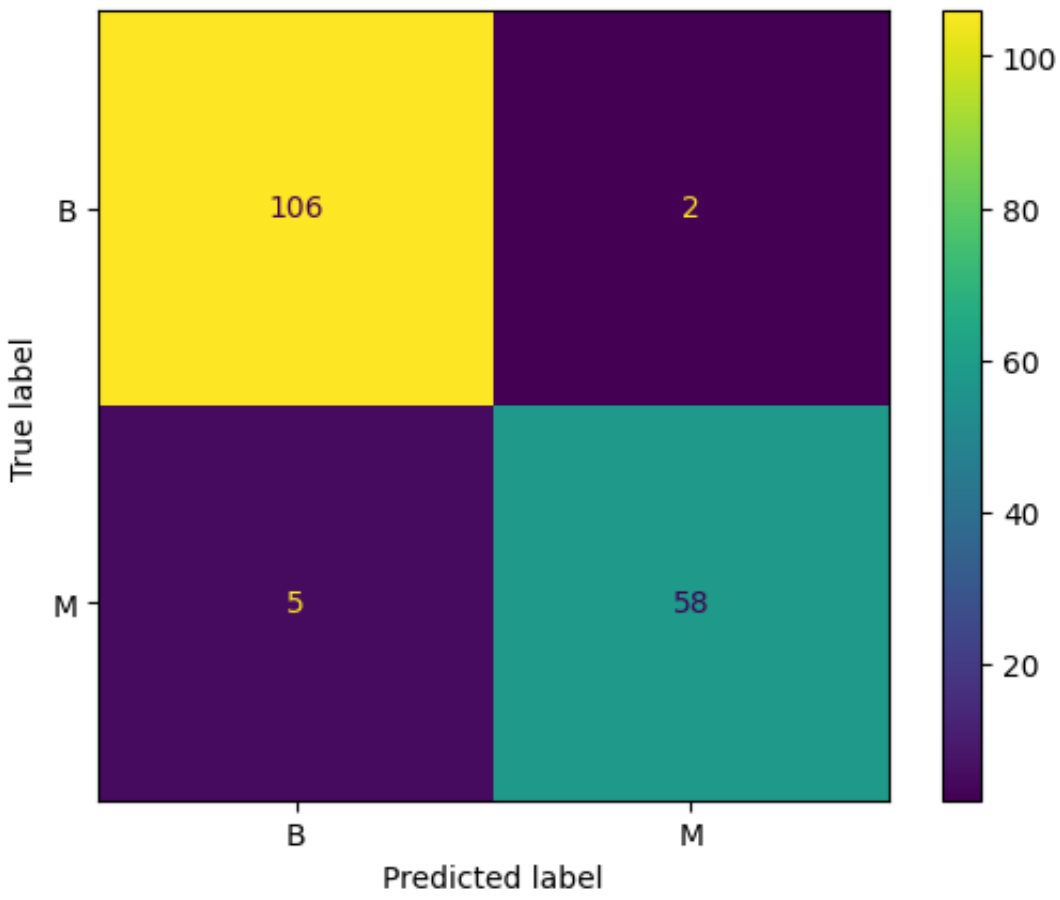
```
array([[106,   2],
       [  5,  58]])
```

```
score3=accuracy_score(y_test,y_pred)
score3
```

```
0.9590643274853801
```

```
labels=['B','M']
cmd=ConfusionMatrixDisplay(cm,display_labels=labels)
cmd.plot()
```

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7f5922a1b6d0>
```




```
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.95      0.98      0.97       108
           1       0.97      0.92      0.94        63

    accuracy                           0.96       171
   macro avg       0.96      0.95      0.96       171
weighted avg       0.96      0.96      0.96       171
```

LogisticRegression

```
lr=LogisticRegression()
lr.fit(x_train,y_train)
y_pred=lr.predict(x_test)
```

```
print("MAPE:",mean_absolute_percentage_error(y_test,y_pred))
print("R2 score:",r2_score(y_test,y_pred))
```

```
MAPE: 26336839926143.27
R2 score: 0.8994708994708994
```

```
score4=accuracy_score(y_test,y_pred)
score4
```

```
0.9766081871345029
```

```
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.97      0.99      0.98       108
           1       0.98      0.95      0.97        63

    accuracy                           0.98       171
   macro avg       0.98      0.97      0.97       171
weighted avg       0.98      0.98      0.98       171
```