dsbda2-1

April 28, 2025

[1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

[2]:
```
dataset = pd.read_csv('cancer_data_set - cancer_data_set.csv')
```

[3]:
```
dataset.shape
```

[3]:
```
(569, 32)
```

[4]:
```
dataset.head()
```

[4]:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	\
0	842302	M	17.99	10.38	122.80	1001.0	
1	842517	M	20.57	17.77	132.90	1326.0	
2	84300903	M	19.69	21.25	130.00	1203.0	
3	84348301	M	11.42	20.38	77.58	386.1	
4	84358402	M	20.29	14.34	135.10	1297.0	

	smoothness_mean	compactness_mean	concavity_mean	concave points_mean	\
0	0.11840	0.27760	0.3001	0.14710	
1	0.08474	0.07864	0.0869	0.07017	
2	0.10960	0.15990	0.1974	0.12790	
3	0.14250	0.28390	0.2414	0.10520	
4	0.10030	0.13280	0.1980	0.10430	

	…	radius_worst	texture_worst	perimeter_worst	area_worst	\
0	…	25.38	17.33	184.60	2019.0	
1	…	24.99	23.41	158.80	1956.0	
2	…	23.57	25.53	152.50	1709.0	
3	…	14.91	26.50	98.87	567.7	
4	…	22.54	16.67	152.20	1575.0	

	smoothness_worst	compactness_worst	concavity_worst	concave points_worst	\
0	0.1622	0.6656	0.7119	0.2654	
1	0.1238	0.1866	0.2416	0.1860	

```
2            0.1444             0.4245          0.4504               0.2430
3            0.2098             0.8663          0.6869               0.2575
4            0.1374             0.2050          0.4000               0.1625

   symmetry_worst  fractal_dimension_worst
0          0.4601                  0.11890
1          0.2750                  0.08902
2          0.3613                  0.08758
3          0.6638                  0.17300
4          0.2364                  0.07678

[5 rows x 32 columns]
```

[5]:
```
dataset.dtypes
```

[5]:
```
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
```

```
dtype: object
```

[6]:
```
dataset.isnull().sum()
```

[6]:
```
id                        0
diagnosis                 0
radius_mean               0
texture_mean              0
perimeter_mean            0
area_mean                 2
smoothness_mean           0
compactness_mean          1
concavity_mean            1
concave points_mean       0
symmetry_mean             0
fractal_dimension_mean    1
radius_se                 2
texture_se                0
perimeter_se              0
area_se                   1
smoothness_se             0
compactness_se            0
concavity_se              1
concave points_se         2
symmetry_se               0
fractal_dimension_se      0
radius_worst              0
texture_worst             0
perimeter_worst           0
area_worst                0
smoothness_worst          0
compactness_worst         1
concavity_worst           0
concave points_worst      0
symmetry_worst            2
fractal_dimension_worst   0
dtype: int64
```

[7]:
```
dataset["area_mean"]=dataset["area_mean"].fillna(dataset["area_mean"].mean())
```

[8]:
```
dataset["compactness_mean"]=dataset["compactness_mean"].
 ↪fillna(dataset["compactness_mean"].mean())
```

[9]:
```
dataset["concavity_mean"]=dataset["concavity_mean"].
 ↪fillna(dataset["concavity_mean"].mean())
```

[10]:
```
dataset["fractal_dimension_mean"]=dataset["fractal_dimension_mean"].
↪fillna(dataset["fractal_dimension_mean"].mean())
```

[11]:
```
dataset["radius_se"]=dataset["radius_se"].fillna(dataset["radius_se"].mean())
```

[12]:
```
dataset["area_se"]=dataset["area_se"].fillna(dataset["area_se"].mean())
```

[13]:
```
dataset["concavity_se"]=dataset["concavity_se"].fillna(dataset["concavity_se"].
↪mean())
```

[14]:
```
dataset["concave points_se"]=dataset["concave points_se"].
↪fillna(dataset["concave points_se"].mean())
```

[15]:
```
dataset["compactness_worst"]=dataset["compactness_worst"].
↪fillna(dataset["compactness_worst"].mean())
```

[16]:
```
dataset["symmetry_worst"]=dataset["symmetry_worst"].
↪fillna(dataset["symmetry_worst"].mean())
```

[17]:
```
dataset.isnull().sum().sum()
```

[17]:
```
0
```

1 SVC

[18]:
```
from sklearn.svm import SVC
model = SVC(kernel='rbf', random_state=0)
```

[19]:
```
X = dataset.drop(['diagnosis','id'], axis=1)
```

[20]:
```
X
```

[20]:
```
     radius_mean  texture_mean  perimeter_mean  area_mean  smoothness_mean  \
0          17.99         10.38          122.80     1001.0          0.11840
1          20.57         17.77          132.90     1326.0          0.08474
2          19.69         21.25          130.00     1203.0          0.10960
3          11.42         20.38           77.58      386.1          0.14250
4          20.29         14.34          135.10     1297.0          0.10030
..           …             …               …          …               …
564        21.56         22.39          142.00     1479.0          0.11100
565        20.13         28.25          131.20     1261.0          0.09780
566        16.60         28.08          108.30      858.1          0.08455
567        20.60         29.33          140.10     1265.0          0.11780
568         7.76         24.54           47.92      181.0          0.05263

     compactness_mean  concavity_mean  concave points_mean  symmetry_mean  \
```

```
0             0.27760         0.30010              0.14710         0.2419
1             0.07864         0.08690              0.07017         0.1812
2             0.15990         0.19740              0.12790         0.2069
3             0.28390         0.24140              0.10520         0.2597
4             0.13280         0.19800              0.10430         0.1809
..                …               …                    …             …
564           0.11590         0.24390              0.13890         0.1726
565           0.10340         0.14400              0.09791         0.1752
566           0.10230         0.09251              0.05302         0.1590
567           0.27700         0.35140              0.15200         0.2397
568           0.04362         0.00000              0.00000         0.1587

     fractal_dimension_mean  …  radius_worst  texture_worst  \
0                   0.07871  …        25.380          17.33
1                   0.05667  …        24.990          23.41
2                   0.05999  …        23.570          25.53
3                   0.09744  …        14.910          26.50
4                   0.05883  …        22.540          16.67
..                      …  …           …             …
564                 0.05623  …        25.450          26.40
565                 0.05533  …        23.690          38.25
566                 0.05648  …        18.980          34.12
567                 0.07016  …        25.740          39.42
568                 0.05884  …         9.456          30.37

     perimeter_worst  area_worst  smoothness_worst  compactness_worst  \
0             184.60      2019.0           0.16220            0.66560
1             158.80      1956.0           0.12380            0.18660
2             152.50      1709.0           0.14440            0.42450
3              98.87       567.7           0.20980            0.86630
4             152.20      1575.0           0.13740            0.20500
..               …          …                …                  …
564           166.10      2027.0           0.14100            0.21130
565           155.00      1731.0           0.11660            0.19220
566           126.70      1124.0           0.11390            0.30940
567           184.60      1821.0           0.16500            0.86810
568            59.16       268.6           0.08996            0.06444

     concavity_worst  concave points_worst  symmetry_worst  \
0             0.7119                0.2654          0.4601
1             0.2416                0.1860          0.2750
2             0.4504                0.2430          0.3613
3             0.6869                0.2575          0.6638
4             0.4000                0.1625          0.2364
..               …                     …               …
564           0.4107                0.2216          0.2060
565           0.3215                0.1628          0.2572
```

```
566           0.3403                0.1418          0.2218
567           0.9387                0.2650          0.4087
568           0.0000                0.0000          0.2871

     fractal_dimension_worst
0                    0.11890
1                    0.08902
2                    0.08758
3                    0.17300
4                    0.07678
..                       …
564                  0.07115
565                  0.06637
566                  0.07820
567                  0.12400
568                  0.07039

[569 rows x 30 columns]
```

[21]:
```
Y = dataset['diagnosis']
```

[22]:
```
Y
```

[22]:
```
0      M
1      M
2      M
3      M
4      M
      ..
564    M
565    M
566    M
567    M
568    B
Name: diagnosis, Length: 569, dtype: object
```

[51]:
```
from sklearn.preprocessing import LabelEncoder
```

[52]:
```
LE = LabelEncoder()
```

[73]:
```
Y=LE.fit_transform(Y)
Y
```

[73]:
```
array([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, 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])
```

[56]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.20,␣
 ↪random_state = 0)
```

[57]:
```
model.fit(X_train,Y_train)
```

[57]:
```
SVC(random_state=0)
```

[58]:
```
y_pred = model.predict(X_test)
```

[59]:
```
from sklearn.metrics import accuracy_score, confusion_matrix
```

[60]:
```
accuracy_score(Y_test, y_pred)
```

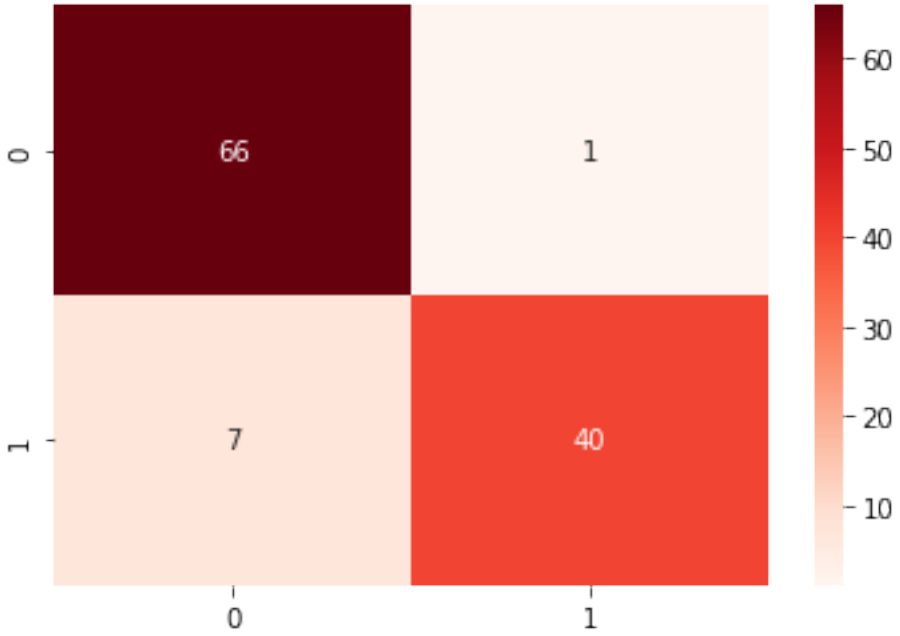
[60]:
```
0.9298245614035088
```

[61]:
```
cm = confusion_matrix(Y_test, y_pred)
cm
```

[61]:
```
array([[66,  1],
       [ 7, 40]])
```

[62]:
```
sns.heatmap(cm, annot=True,cmap='Reds')
```

[62]: <AxesSubplot: >



2 Random forest

```
[63]: from sklearn.ensemble import RandomForestClassifier
      model1 = RandomForestClassifier(n_estimators=100, random_state=0)
      model1.fit(X_train,Y_train)
```

[63]: RandomForestClassifier(random_state=0)

```
[64]: y_pred1 = model1.predict(X_test)
```

```
[65]: accuracy_score(Y_test,y_pred1)
```

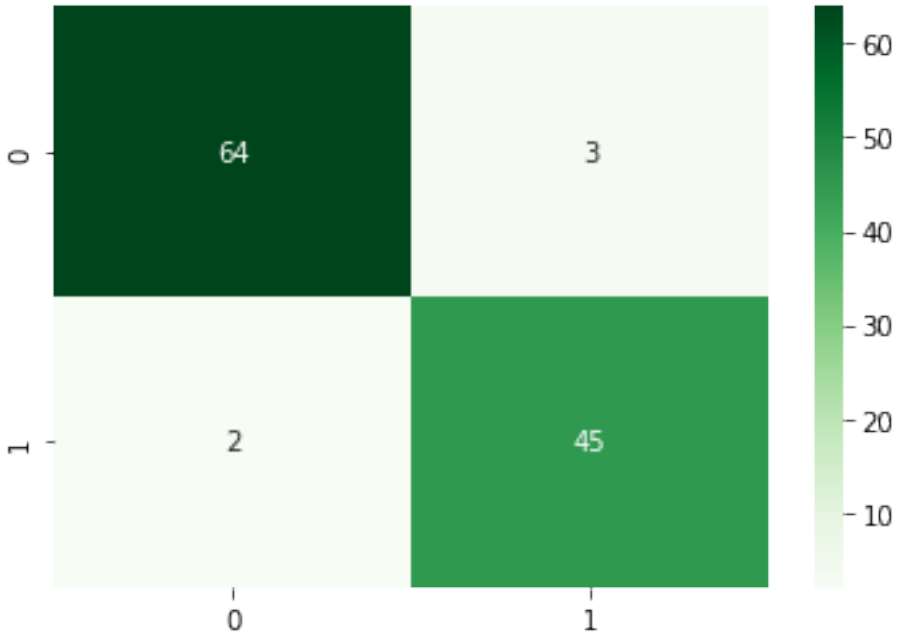
[65]: 0.956140350877193

```
[66]: cm1 = confusion_matrix(Y_test, y_pred1)
      cm1
```

[66]: array([[64, 3],
       [2, 45]])

```
[67]: sns.heatmap(cm1, annot=True,cmap='Greens')
```

[67]: <AxesSubplot: >

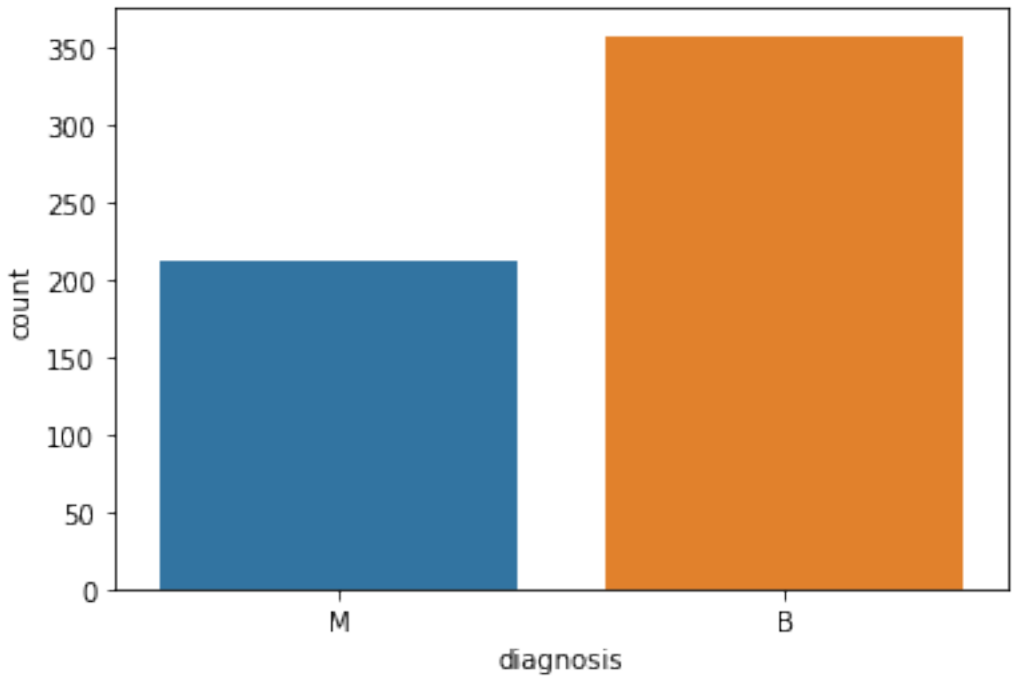


[68]:
```
dataset.duplicated().sum()
```

[68]: 0

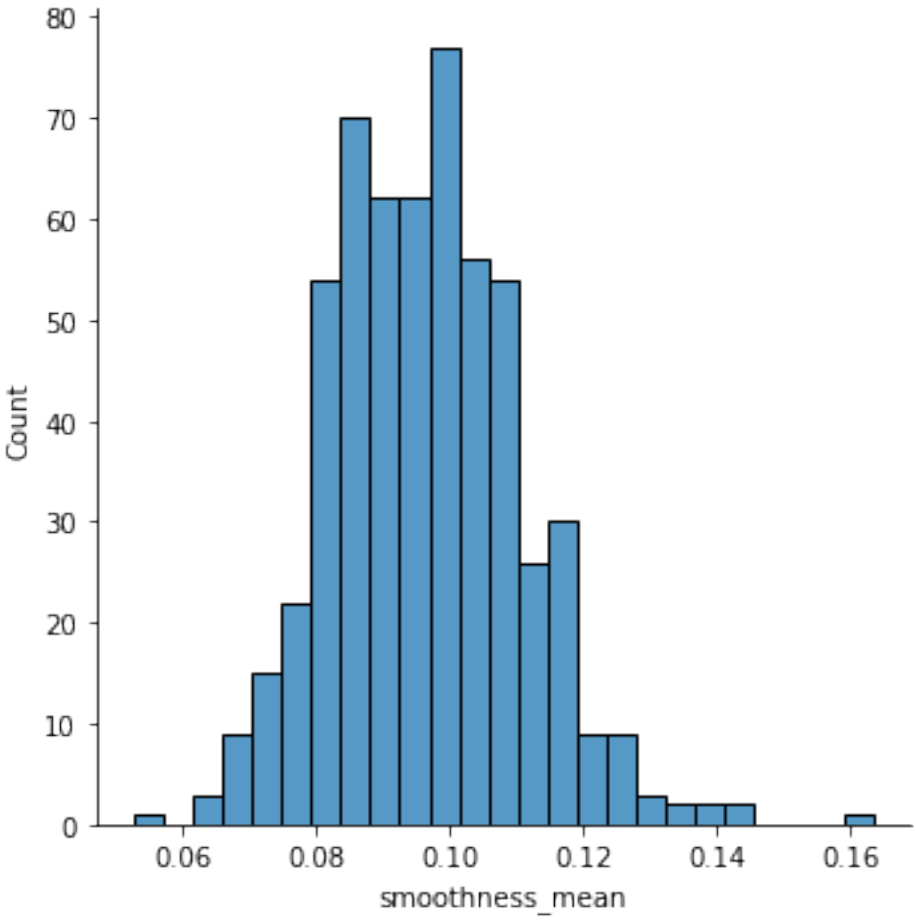
[69]:
```
sns.countplot(dataset,x='diagnosis')
```

[69]: <AxesSubplot: xlabel='diagnosis', ylabel='count'>



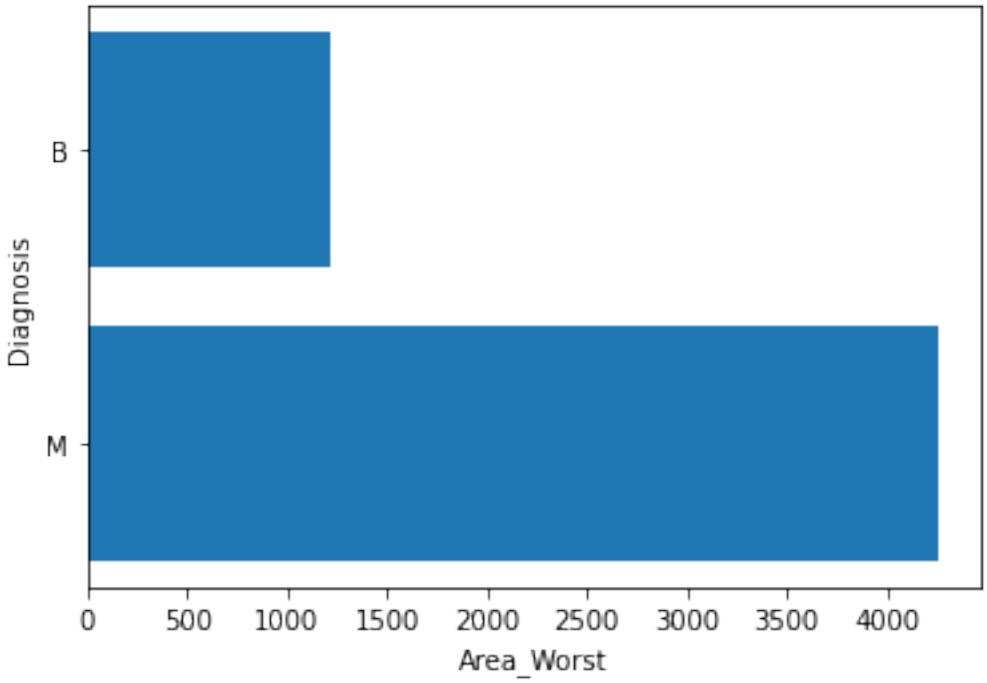
[70]:
```
sns.displot(dataset['smoothness_mean'])
```

[70]: <seaborn.axisgrid.FacetGrid at 0x7fb0a23e5c40>



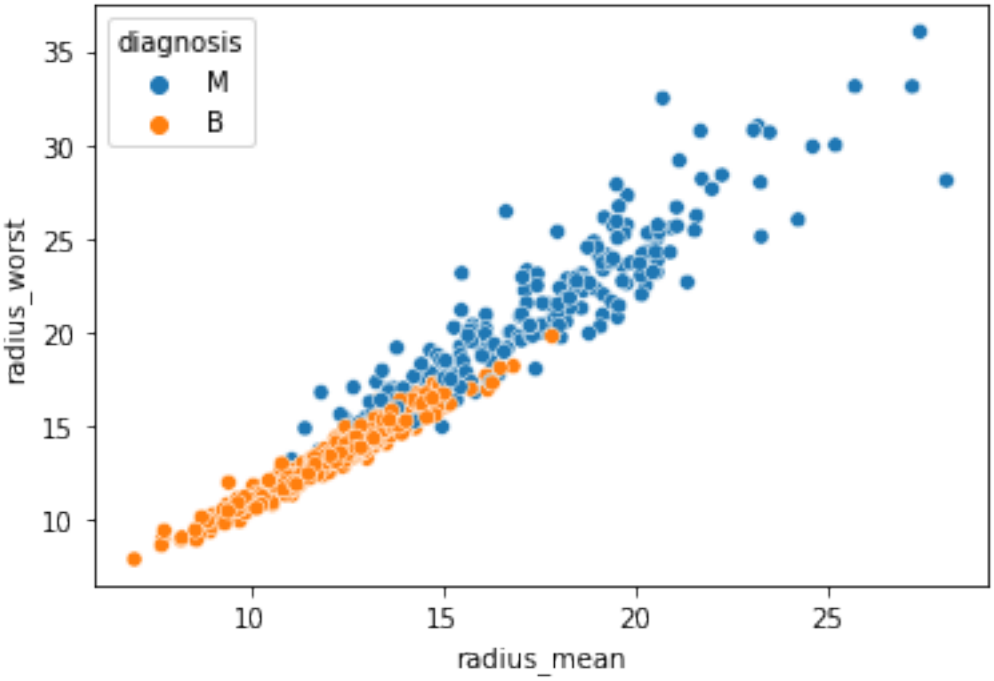
[71]:
```
plt.barh(dataset['diagnosis'],dataset['area_worst'])
plt.xlabel('Area_Worst')
plt.ylabel('Diagnosis')
```

[71]: Text(0, 0.5, 'Diagnosis')



[72]:
```
sns.scatterplot(dataset,x='radius_mean',y='radius_worst',hue='diagnosis')
```

[72]: <AxesSubplot: xlabel='radius_mean', ylabel='radius_worst'>



[78]:
```
X_test.iloc[0,:].array
```

[78]:
```
<PandasArray>
[    13.4,    20.52,    88.64,    556.7,   0.1106,   0.1469,   0.1445,
  0.08172,   0.2116,  0.07325,   0.3906,   0.9306,    3.093,    33.67,
 0.005414,  0.02265,  0.03452,  0.01334,  0.01705, 0.004005,    16.41,
    29.66,    113.3,    844.4,   0.1574,   0.3856,   0.5106,   0.2051,
   0.3585,   0.1109]
Length: 30, dtype: float64
```

[87]:
```
test = model1.predict([[    13.4,    20.52,    88.64,    556.7,   0.1106,   0.
↪1469,   0.1445,
  0.08172,   0.2116,  0.07325,   0.3906,   0.9306,    3.093,    33.67,
 0.005414,  0.02265,  0.03452,  0.01334,  0.01705, 0.004005,    16.41,
    29.66,    113.3,    844.4,   0.1574,   0.3856,   0.5106,   0.2051,
   0.3585,   0.1109]])
```

```
/home/student/.local/lib/python3.8/site-packages/sklearn/base.py:464:
UserWarning: X does not have valid feature names, but RandomForestClassifier was
fitted with feature names
  warnings.warn(
```

[86]:
```
Y_test[0]
```

[86]:
```
1
```

[95]:
```
def cancer(test):
    if test==1:
        print("Malignant")
    else:
        print("Benign")
```

[96]:
```
cancer(test)
```

```
Malignant
```

[97]:
```
test1 = model.predict([[    13.4,    20.52,    88.64,    556.7,   0.1106,   0.
↪1469,   0.1445,
  0.08172,   0.2116,  0.07325,   0.3906,   0.9306,    3.093,    33.67,
 0.005414,  0.02265,  0.03452,  0.01334,  0.01705, 0.004005,    16.41,
    29.66,    113.3,    844.4,   0.1574,   0.3856,   0.5106,   0.2051,
   0.3585,   0.1109]])
```

```
/home/student/.local/lib/python3.8/site-packages/sklearn/base.py:464:
UserWarning: X does not have valid feature names, but SVC was fitted with
```

```
feature names
  warnings.warn(
```

[98]:
```
cancer(test1)
```

```
Benign
```

[100]:
```
from sklearn.metrics import classification_report
```

[101]:
```
print(classification_report(Y_test,y_pred1))
```

```
              precision    recall  f1-score   support

           0       0.97      0.96      0.96        67
           1       0.94      0.96      0.95        47

    accuracy                           0.96       114
   macro avg       0.95      0.96      0.95       114
weighted avg       0.96      0.96      0.96       114
```

[ ]:
```

```