PROJECT 3

The project uses machine learning to detect the various types of *breast cancer*

Exploring the dataset

```
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
```

```
for column in df:
    if df[column].dtype == 'object':
        print(column)
```

```
diagnosis
```

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

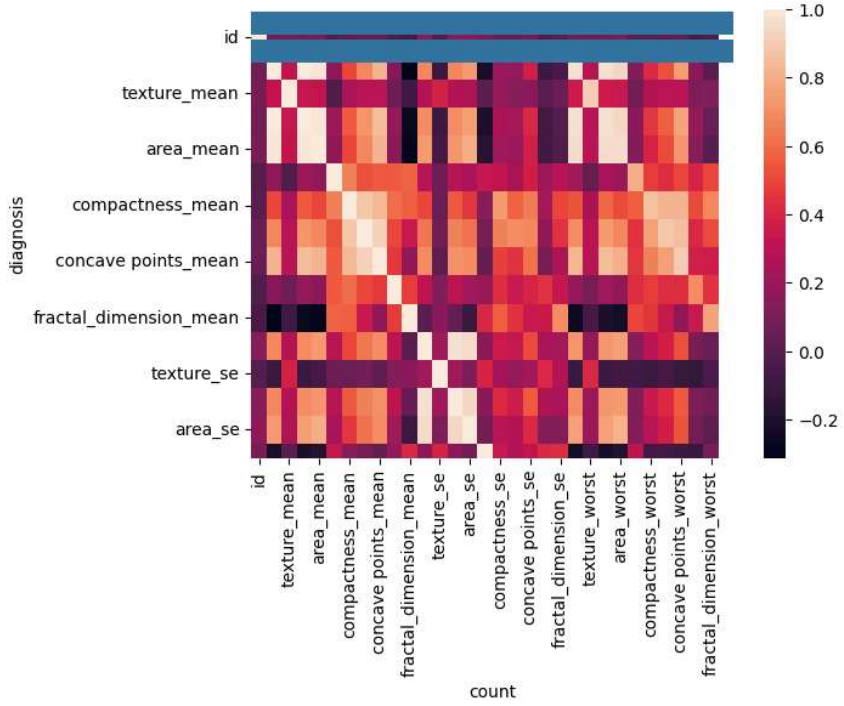
```
id                          0
diagnosis                   0
radius_mean                 0
texture_mean                0
perimeter_mean              0
area_mean                   0
smoothness_mean             0
compactness_mean            0
concavity_mean              0
concave points_mean         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
```

Performing Exploratory Data Analysis on the dataset

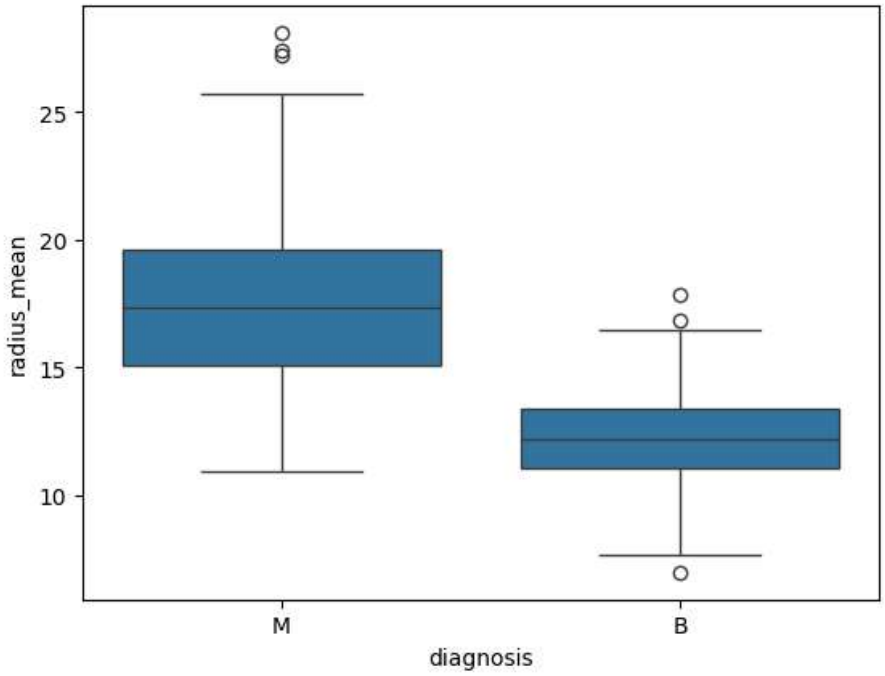
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Load the dataset
df = pd.read_csv('/content/sample_data/data.csv')

# Display the first few rows of the dataset
print(df.head())

# Display the summary statistics of the dataset
print(df.describe())

# Display the correlation matrix
corr = df.corr()
sns.heatmap(corr)

# Display the distribution of the 'diagnosis' column
sns.countplot(df['diagnosis'])
```

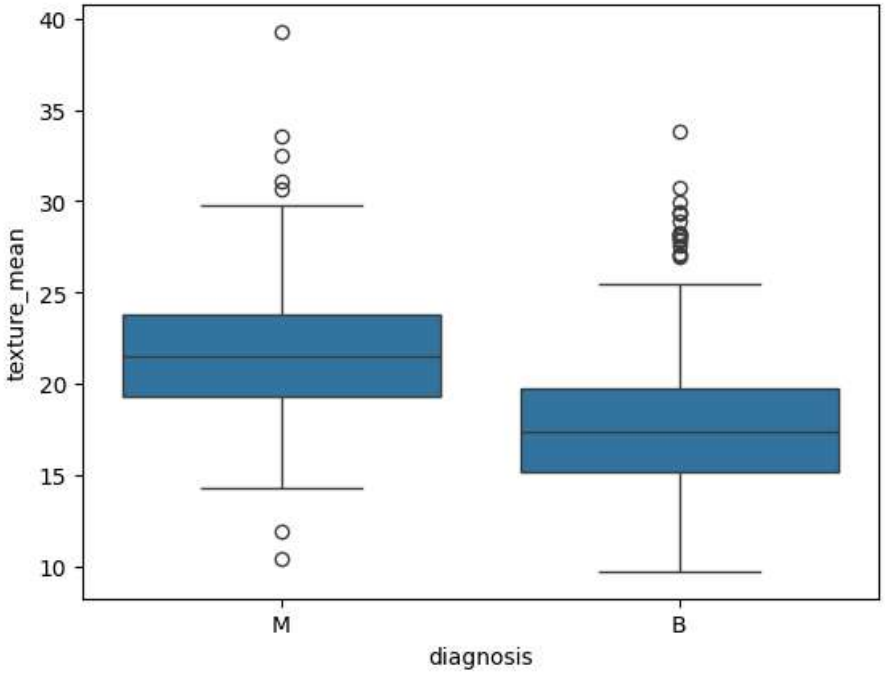
```
         id diagnosis  radius_mean  texture_mean  perimeter_mean  are
0    842302         M        17.99         10.38          122.80
1    842517         M        20.57         17.77          132.90
2  84300903         M        19.69         21.25          130.00
3  84348301         M        11.42         20.38           77.58
4  84358402         M        20.29         14.34          135.10

   smoothness_mean  compactness_mean  concavity_mean  concave points_
0          0.11840           0.27760          0.3001              0.1
1          0.08474           0.07864          0.0869              0.0
2          0.10960           0.15990          0.1974              0.1
3          0.14250           0.28390          0.2414              0.1
4          0.10030           0.13280          0.1980              0.1

   ...  texture_worst  perimeter_worst  area_worst  smoothness_worst
0  ...          17.33           184.60      2019.0            0.1622
1  ...          23.41           158.80      1956.0            0.1238
2  ...          25.53           152.50      1709.0            0.1444
3  ...          26.50            98.87       567.7            0.2098
4  ...          16.67           152.20      1575.0            0.1374

   compactness_worst  concavity_worst  concave points_worst  symmetry
0             0.6656           0.7119                0.2654
1             0.1866           0.2416                0.1860
2             0.4245           0.4504                0.2430
3             0.8663           0.6869                0.2575
4             0.2050           0.4000                0.1625

   fractal_dimension_worst  Unnamed: 32
0                  0.11890          NaN
1                  0.08902          NaN
2                  0.08758          NaN
3                  0.17300          NaN
4                  0.07678          NaN

[5 rows x 33 columns]
                 id  radius_mean  texture_mean  perimeter_mean    are
count  5.690000e+02   569.000000    569.000000      569.000000   569.
mean   3.037183e+07    14.127292     19.289649       91.969033   654.
std    1.250206e+08     3.524049      4.301036       24.298981   351.
min    8.670000e+03     6.981000      9.710000       43.790000   143.
25%    8.692180e+05    11.700000     16.170000       75.170000   420.
50%    9.060240e+05    13.370000     18.840000       86.240000   551.
75%    8.813129e+06    15.780000     21.800000      104.100000   782.
max    9.113205e+08    28.110000     39.280000      188.500000  2501.

       smoothness_mean  compactness_mean  concavity_mean  concave poi
count       569.000000        569.000000      569.000000           56
mean          0.096360          0.104341        0.088799
std           0.014064          0.052813        0.079720
min           0.052630          0.019380        0.000000
25%           0.086370          0.064920        0.029560
50%           0.095870          0.092630        0.061540
75%           0.105300          0.130400        0.130700
max           0.163400          0.345400        0.426800

       symmetry_mean  ...  texture_worst  perimeter_worst   area_wors
count     569.000000  ...     569.000000       569.000000   569.00000
mean        0.181162  ...      25.677223       107.261213   880.58312
std         0.027414  ...       6.146258        33.602542   569.35699
min         0.106000  ...      12.020000        50.410000   185.20000
25%         0.161900  ...      21.080000        84.110000   515.30000
50%         0.179200  ...      25.410000        97.660000   686.50000
75%         0.195700  ...      29.720000       125.400000  1084.00000
max         0.304000  ...      49.540000       251.200000  4254.00000

       smoothness_worst  compactness_worst  concavity_worst  \
count        569.000000         569.000000       569.000000
mean           0.132369           0.254265         0.272188
std            0.022832           0.157336         0.208624
min            0.071170           0.027290         0.000000
25%            0.116600           0.147200         0.114500
50%            0.131300           0.211900         0.226700
75%            0.146000           0.339100         0.382900
max            0.222600           1.058000         1.252000
```

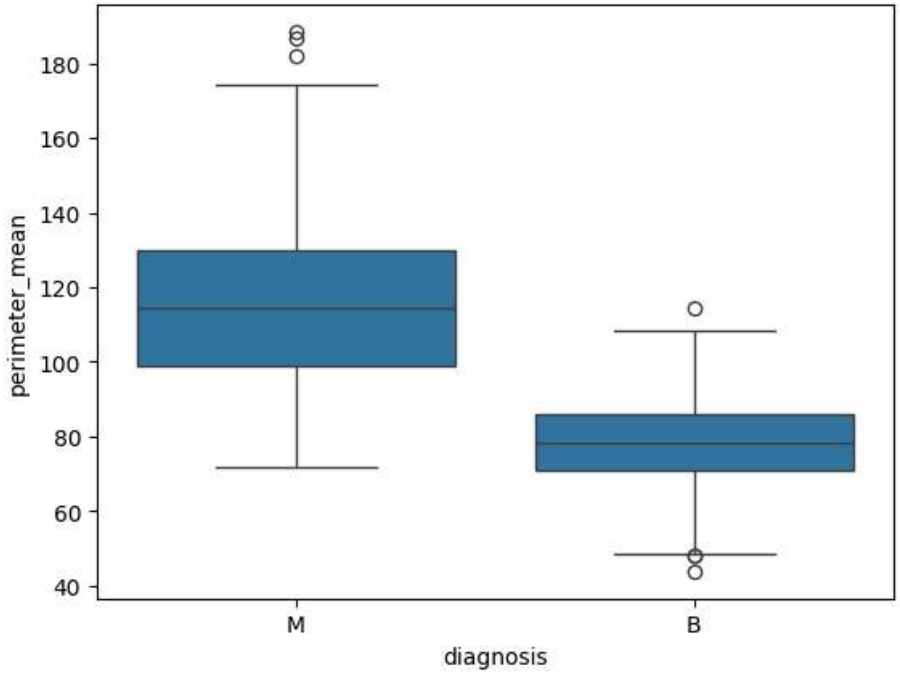
```
       concave points_worst  symmetry_worst  fractal_dimension_worst
count            569.000000      569.000000               569.000000
mean               0.114606        0.290076                 0.083946
std                0.065732        0.061867                 0.018061
min                0.000000        0.156500                 0.055040
25%                0.064930        0.250400                 0.071460
50%                0.099930        0.282200                 0.080040
75%                0.161400        0.317900                 0.092080
max                0.291000        0.663800                 0.207500

       Unnamed: 32
count          0.0
mean           NaN
std            NaN
min            NaN
25%            NaN
50%            NaN
75%            NaN
max            NaN

[8 rows x 32 columns]
<ipython-input-16-f95bc6d7aa74>:15: FutureWarning: The default value
  corr = df.corr()
<Axes: xlabel='count', ylabel='diagnosis'>
```

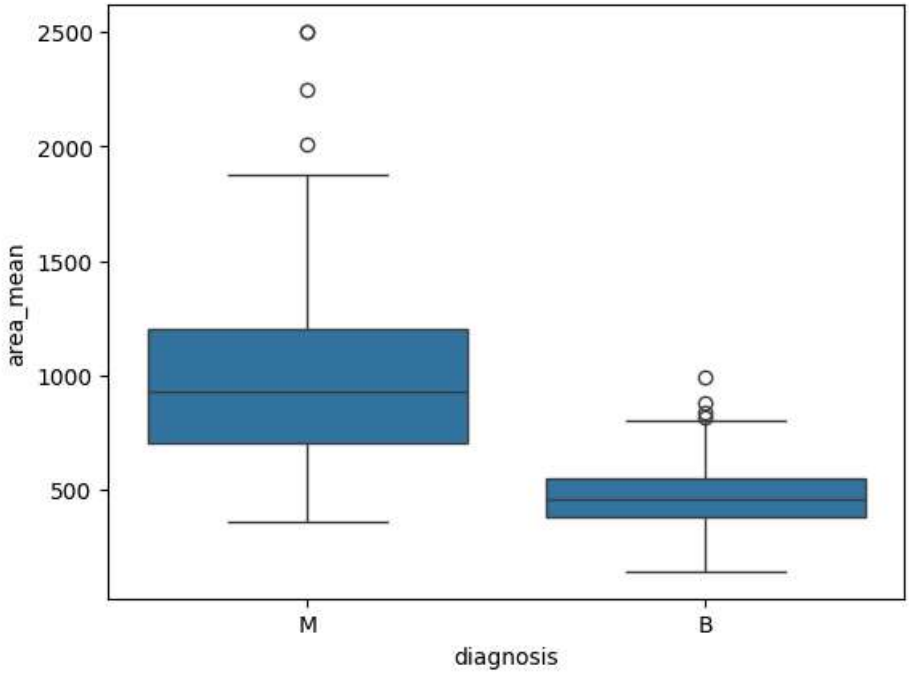


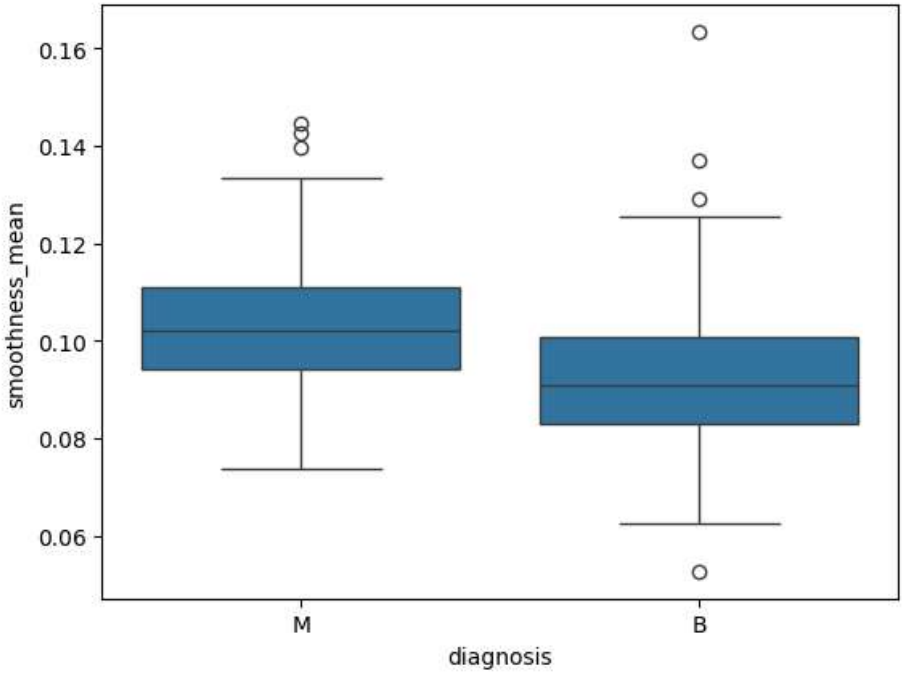

```
for column in df.columns:
    if column not in ['id', 'diagnosis', 'Unnamed: 32']:
        sns.boxplot(x='diagnosis', y=column, data=df)
        plt.show()
```

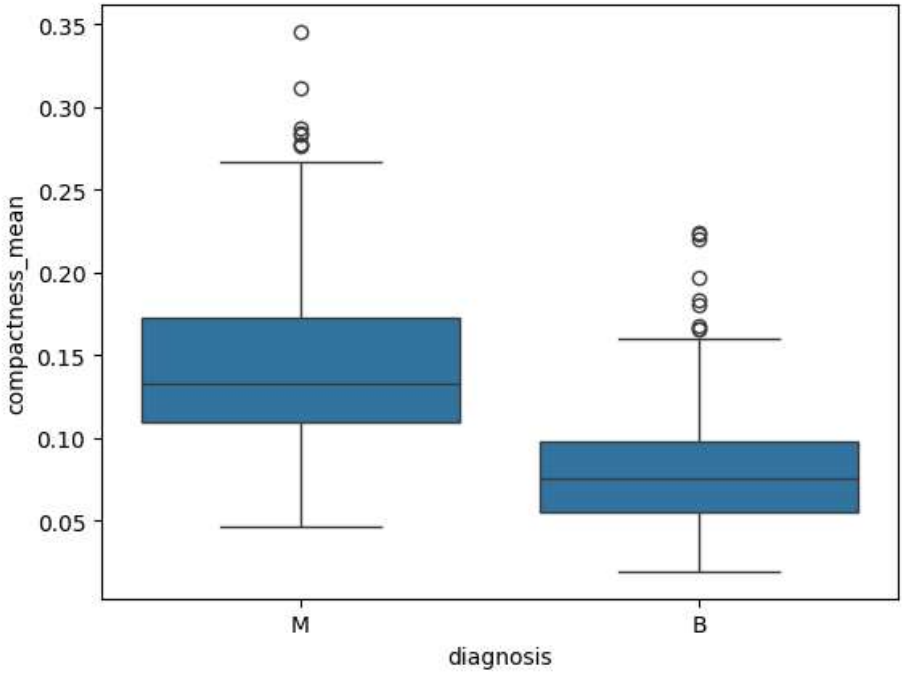

radius_mean
25
20
15
10
M
B
diagnosis

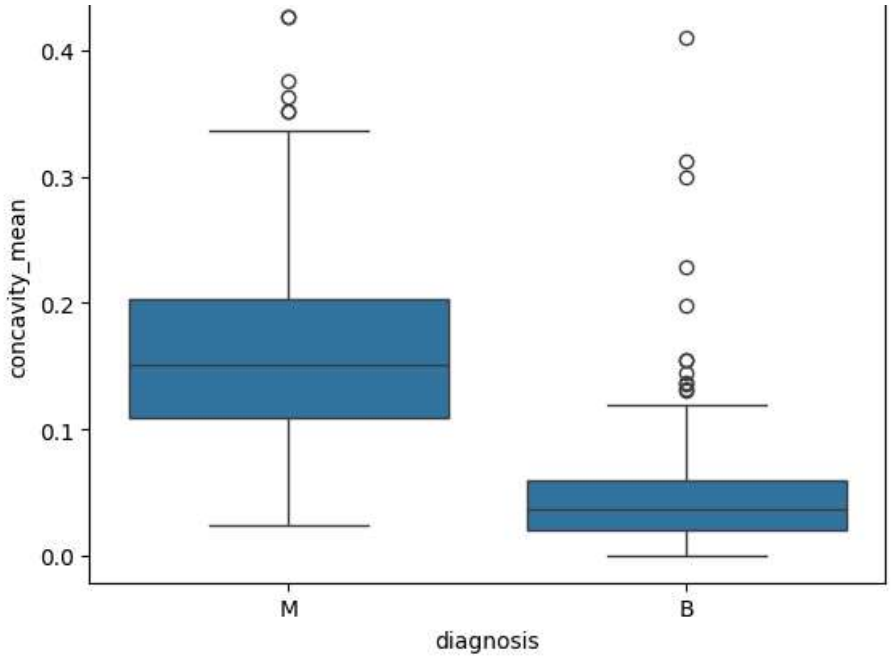

texture_mean
40
35
30
25
20
15
10
M
B
diagnosis

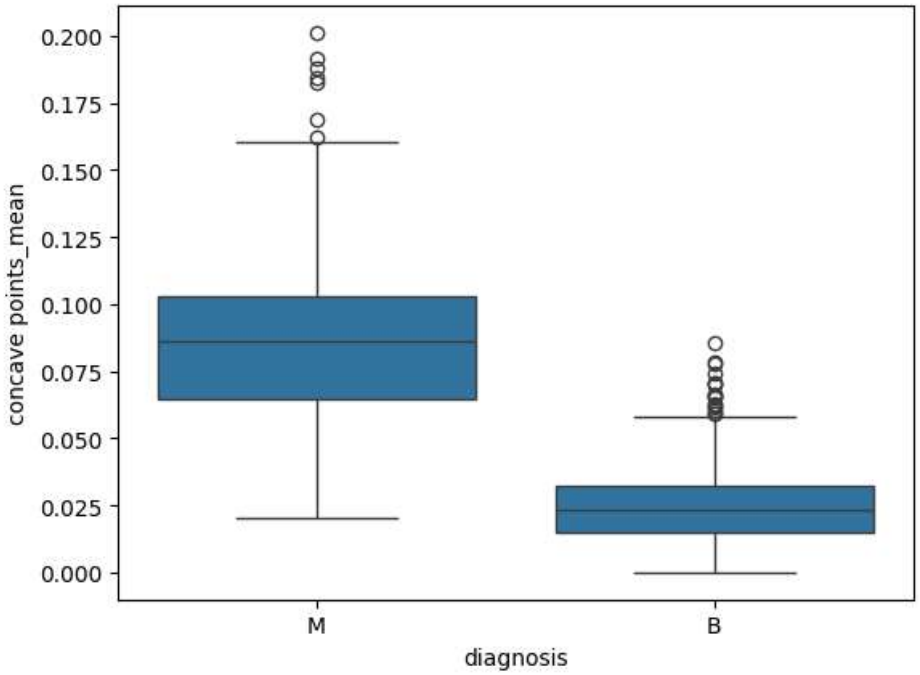

perimeter_mean
180
160
140
120
100
80
60
40
M
B
diagnosis

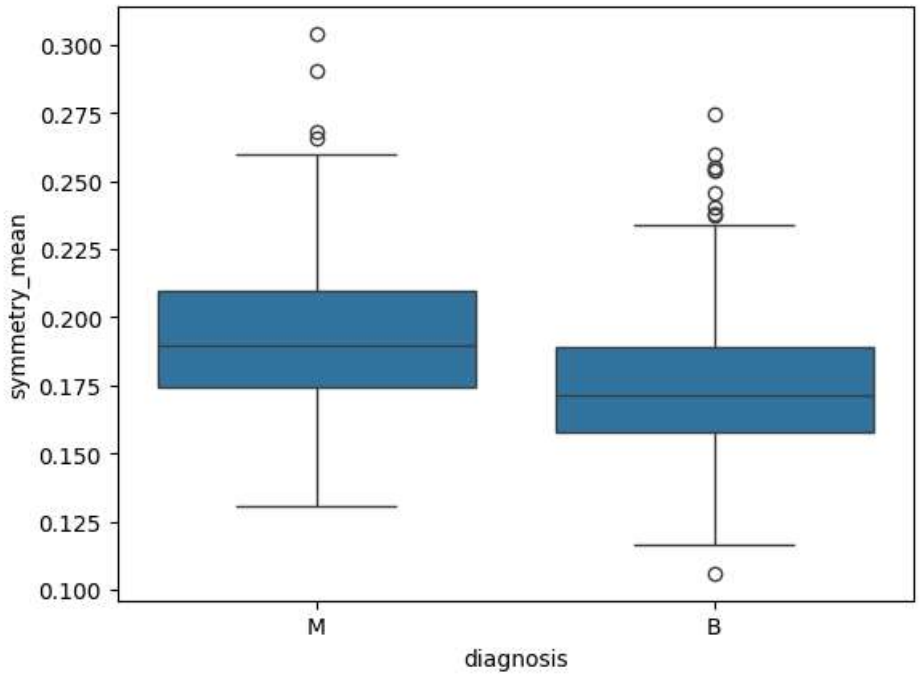

2500
2000
1500
1000
500
area_mean
M
B
diagnosis

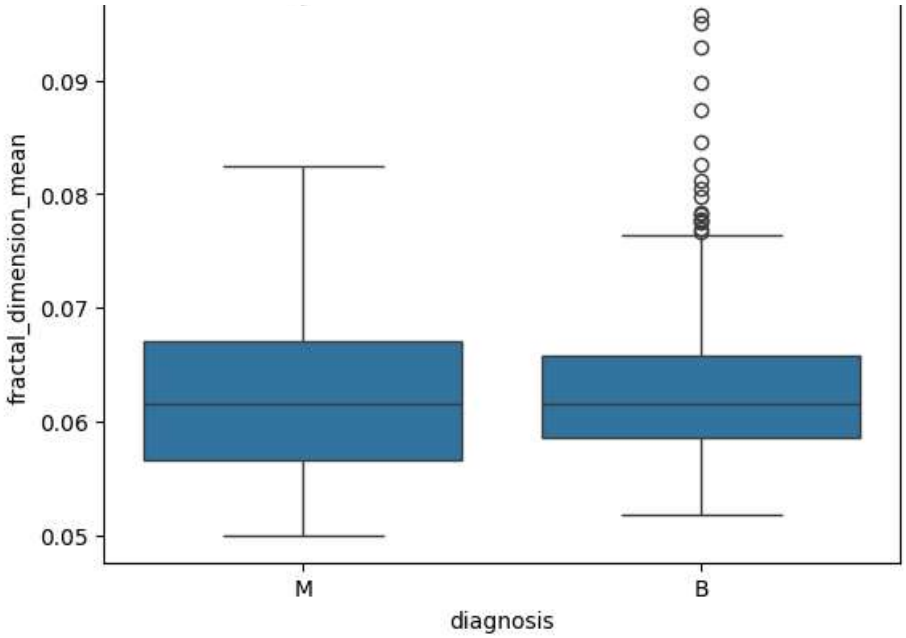

0.16
0.14
0.12
0.10
0.08
0.06
smoothness_mean
M
B
diagnosis

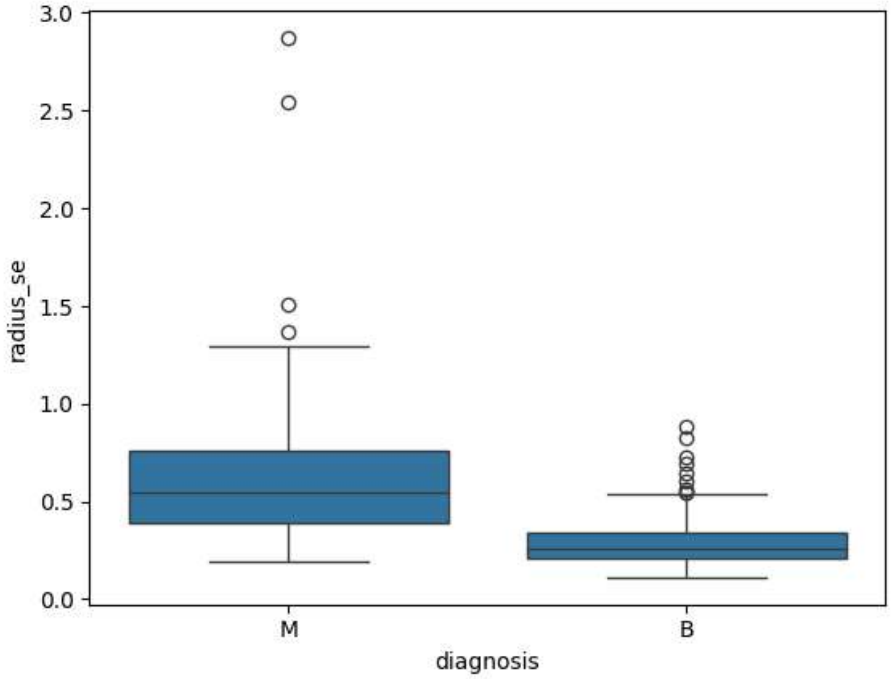

0.35
0.30
0.25
0.20
0.15
0.10
0.05
compactness_mean
M
B
diagnosis

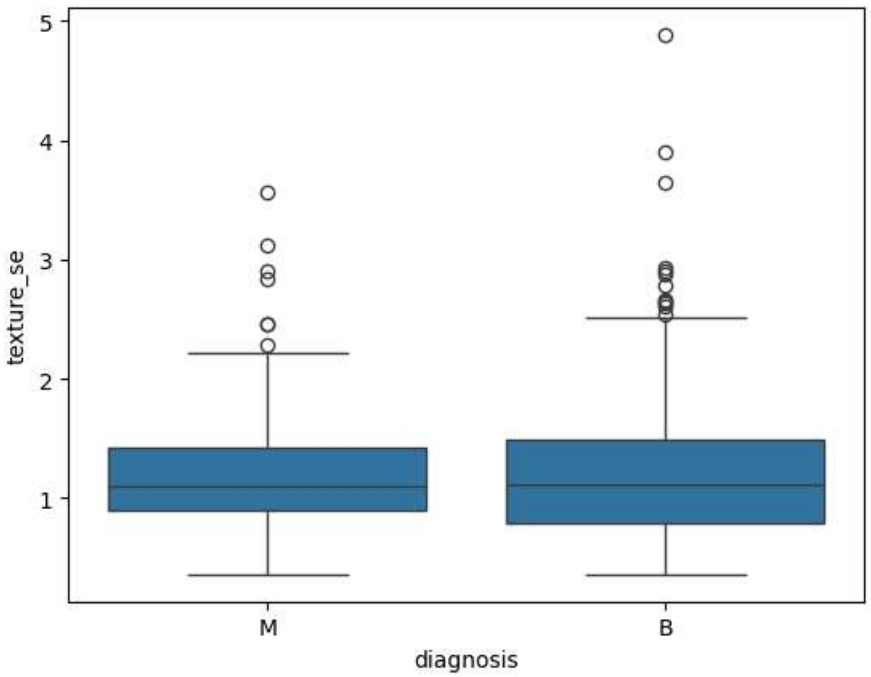

concavity_mean
0.4
0.3
0.2
0.1
0.0
M
B
diagnosis

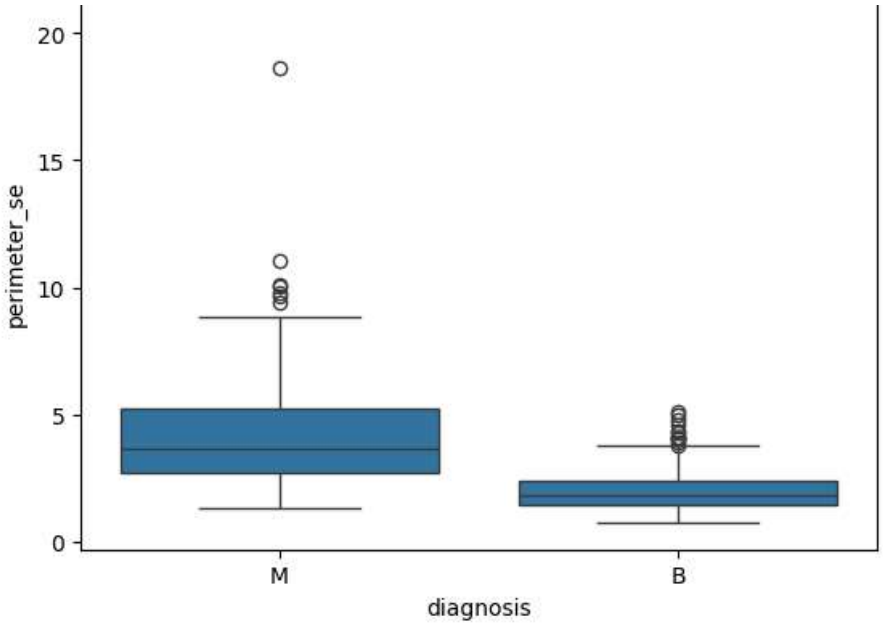

concave points_mean
0.200
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0.000
M
B
diagnosis

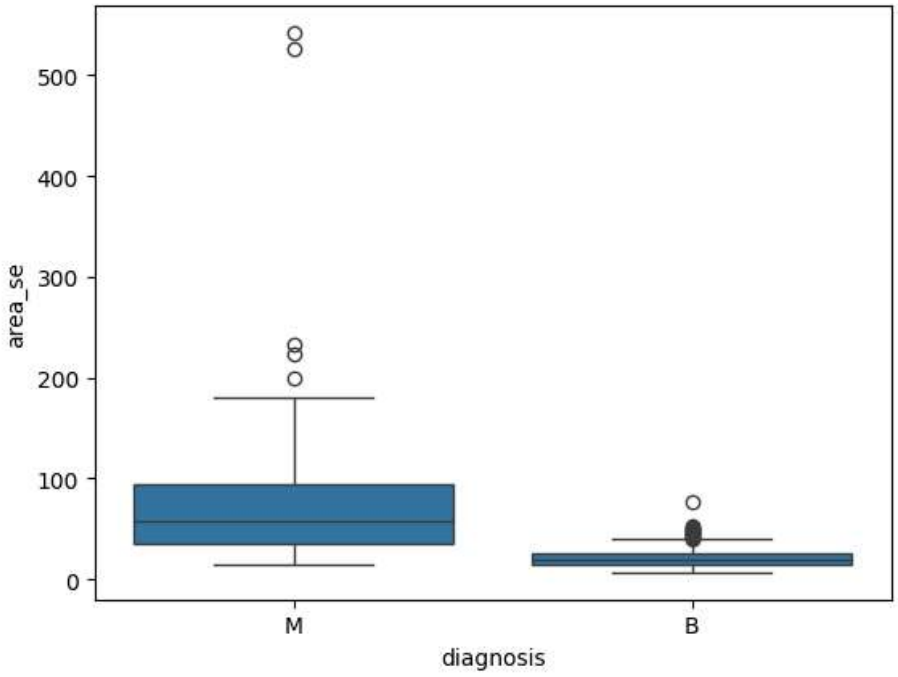

symmetry_mean
0.300
0.275
0.250
0.225
0.200
0.175
0.150
0.125
0.100
M
B
diagnosis

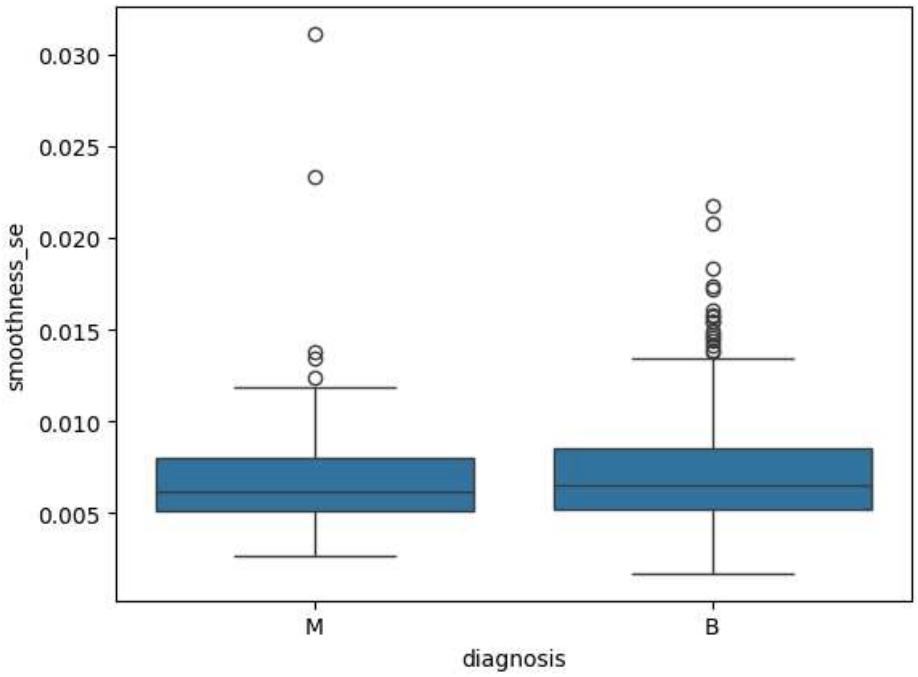

fractal_dimension_mean
0.09
0.08
0.07
0.06
0.05
M
B
diagnosis


radius_se
3.0
2.5
2.0
1.5
1.0
0.5
0.0
M
B
diagnosis

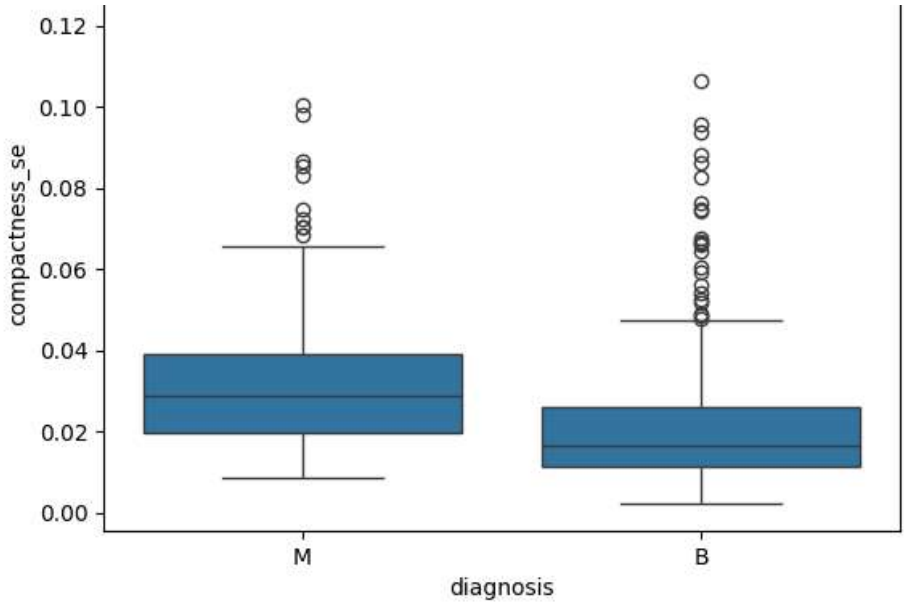

texture_se
5
4
3
2
1
M
B
diagnosis

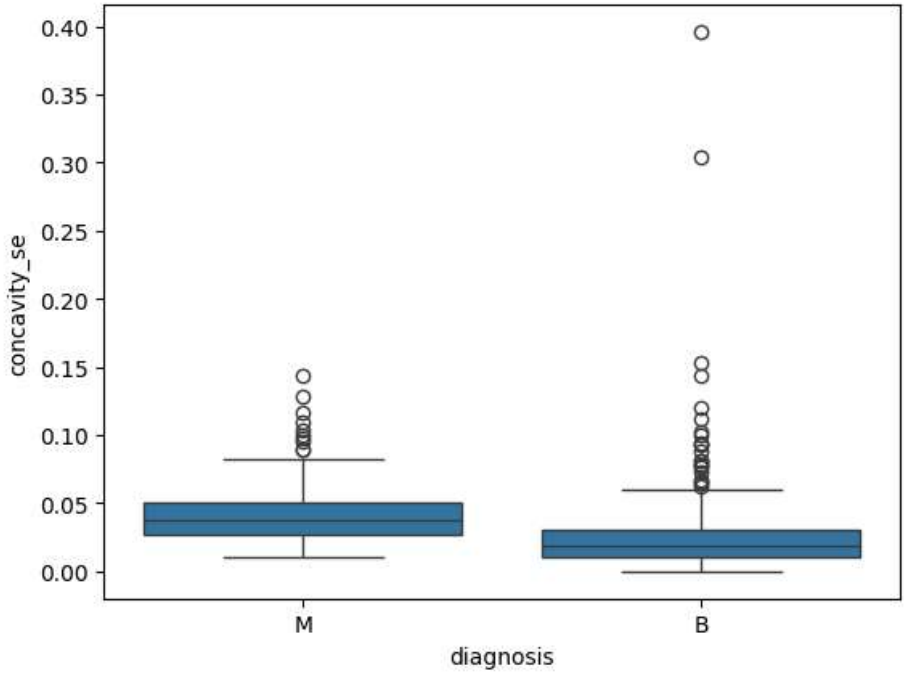

perimeter_se
20
15
10
5
0
M
B
diagnosis

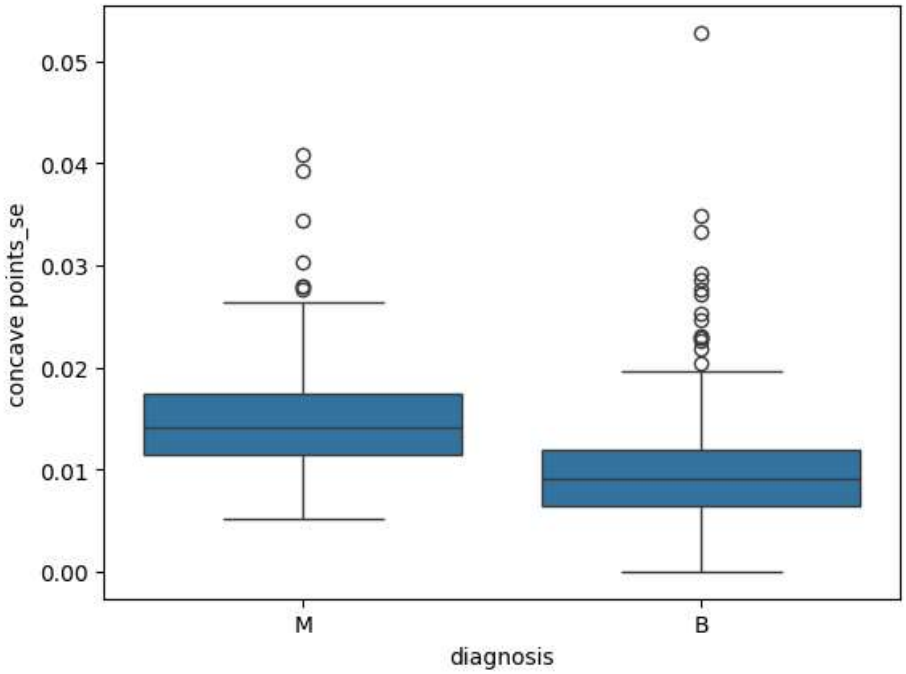

area_se
500
400
300
200
100
0
M
B
diagnosis


smoothness_se
0.030
0.025
0.020
0.015
0.010
0.005
M
B
diagnosis

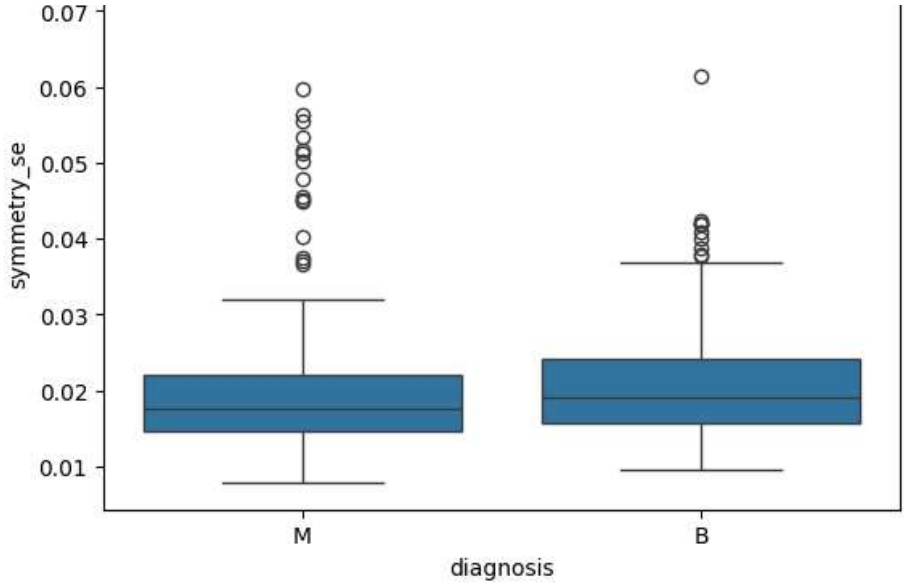

0.14

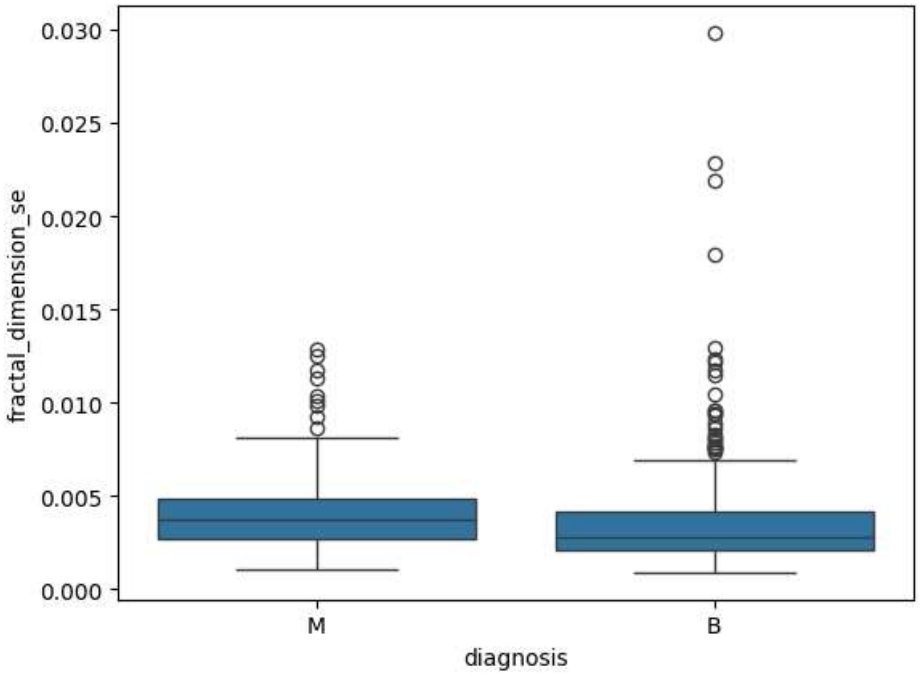

compactness_se
0.12
0.10
0.08
0.06
0.04
0.02
0.00
M
B
diagnosis

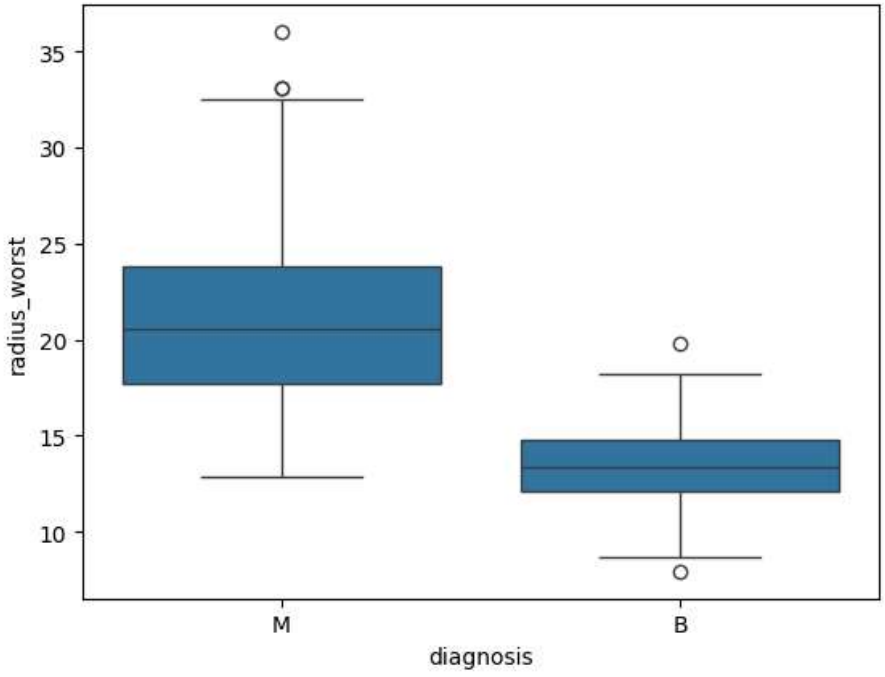

concavity_se
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
M
B
diagnosis


concave points_se
0.05
0.04
0.03
0.02
0.01
0.00
M
B
diagnosis

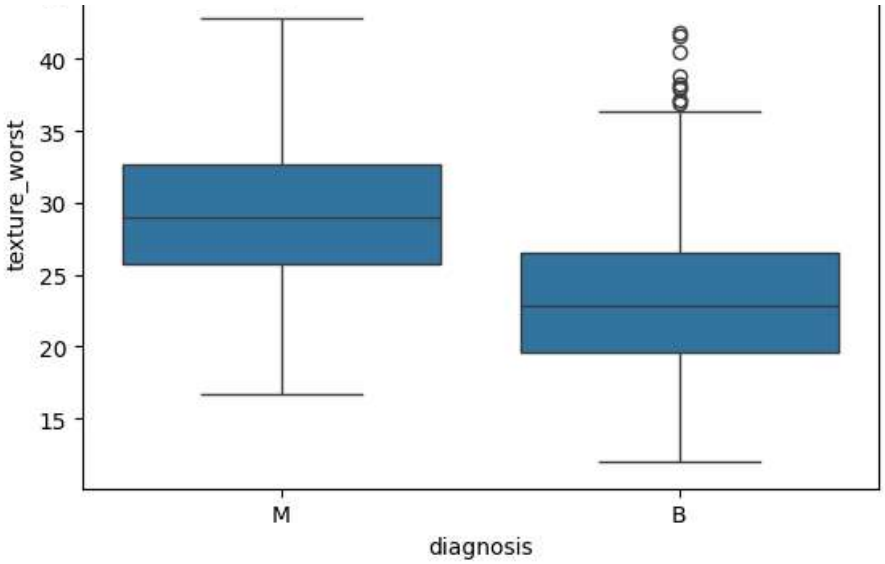

0.08

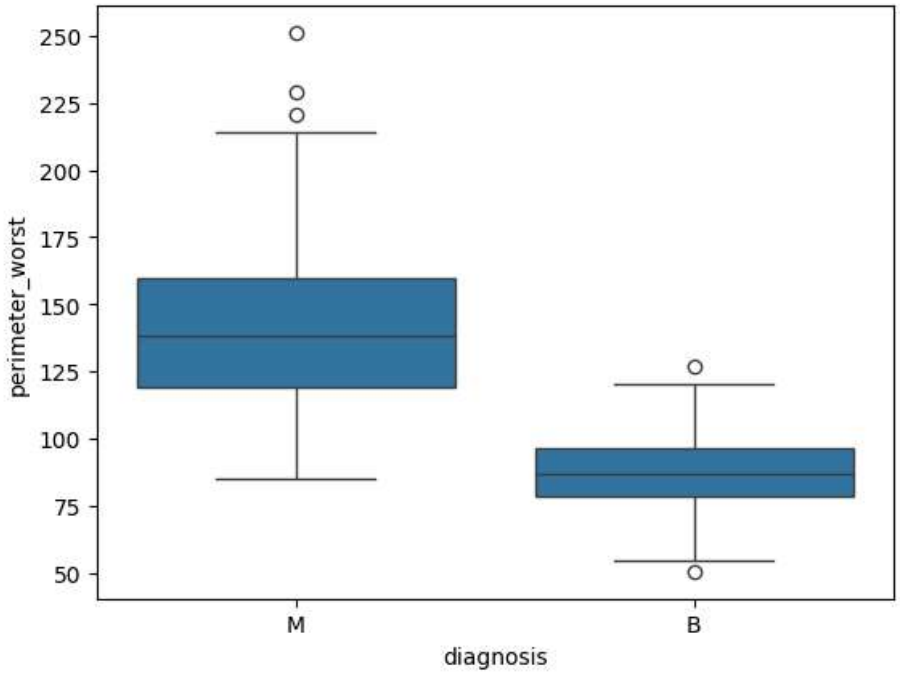

0.07
0.06
0.05
0.04
0.03
0.02
0.01
symmetry_se
M
B
diagnosis

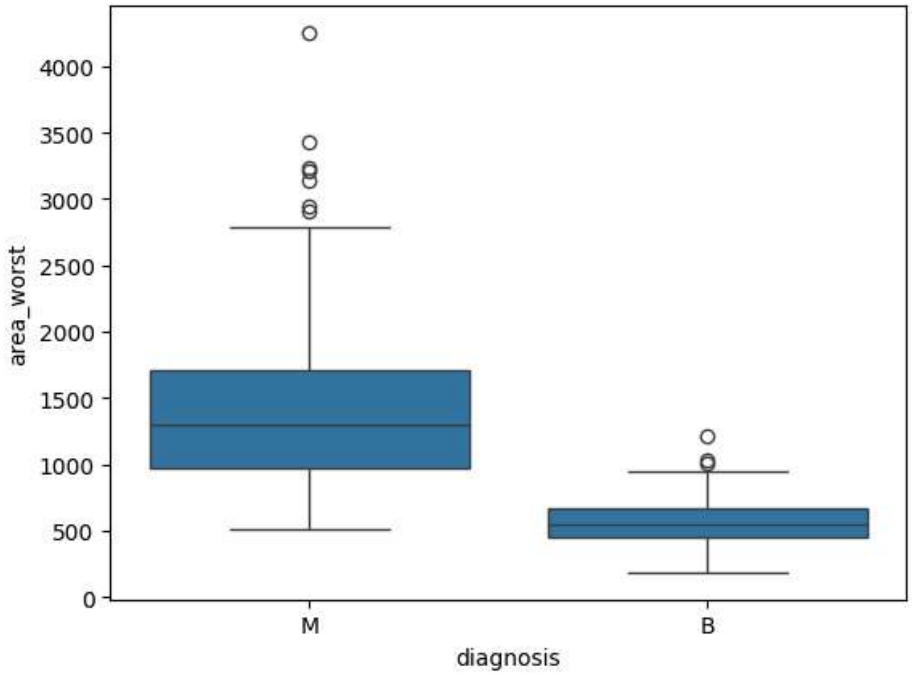

0.030
0.025
0.020
0.015
0.010
0.005
0.000
fractal_dimension_se
M
B
diagnosis


35
30
25
20
15
10
radius_worst
M
B
diagnosis

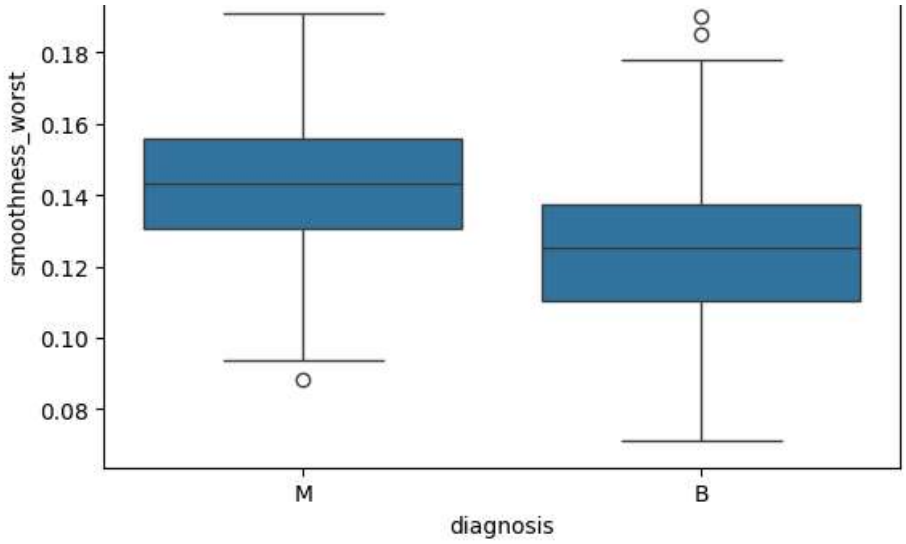

50
45

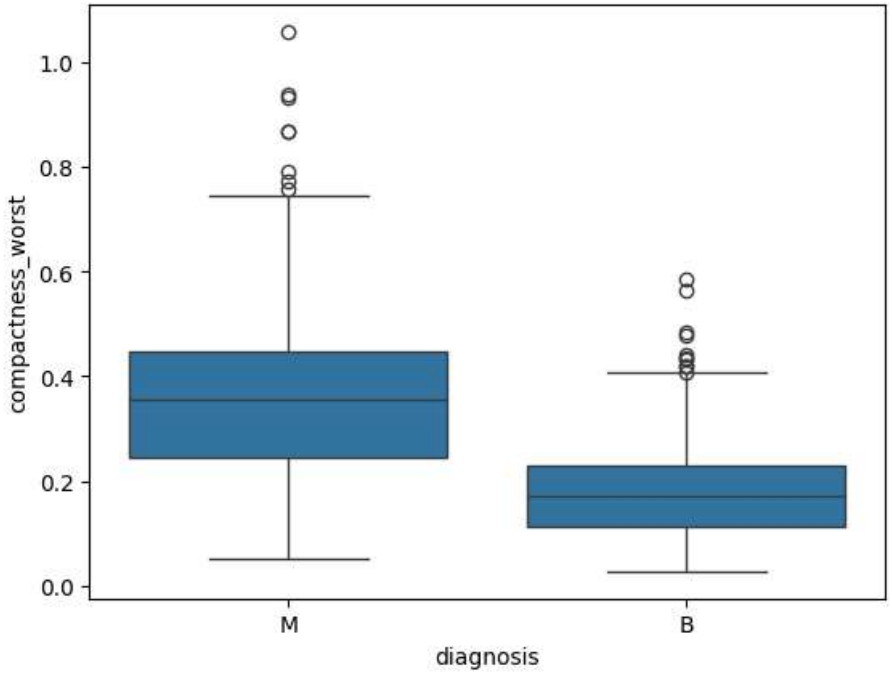

texture_worst
40
35
30
25
20
15
M
B
diagnosis

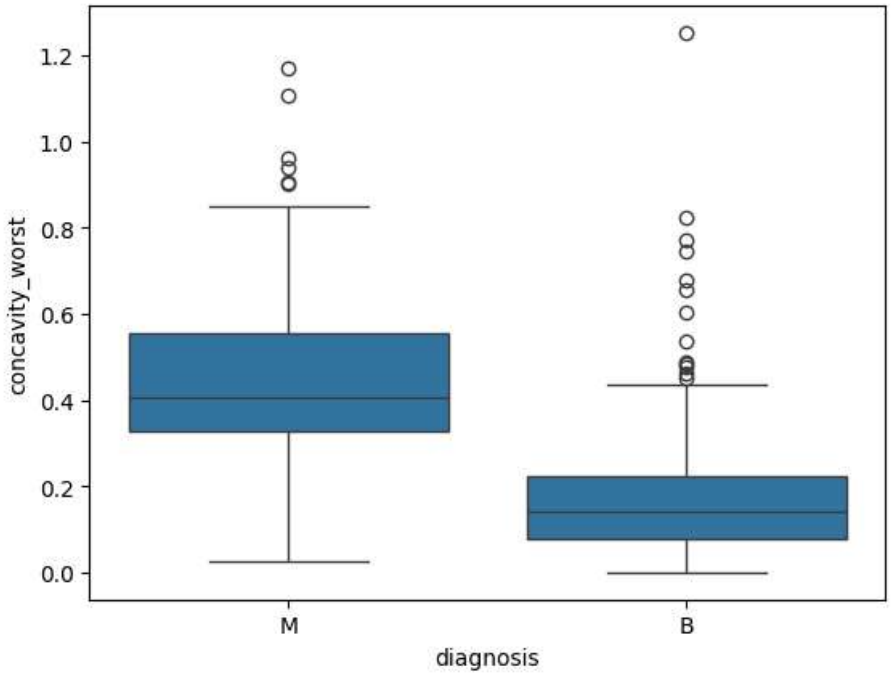

perimeter_worst
250
225
200
175
150
125
100
75
50
M
B
diagnosis

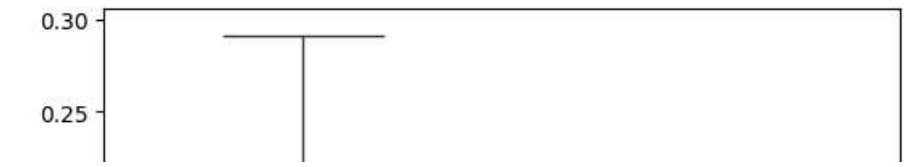

area_worst
4000
3500
3000
2500
2000
1500
1000
500
0
M
B
diagnosis

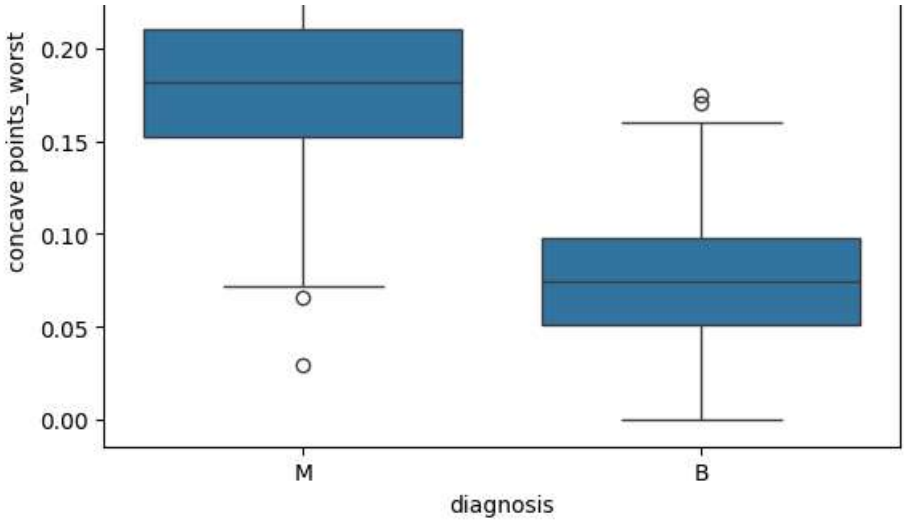

0.22
0.20

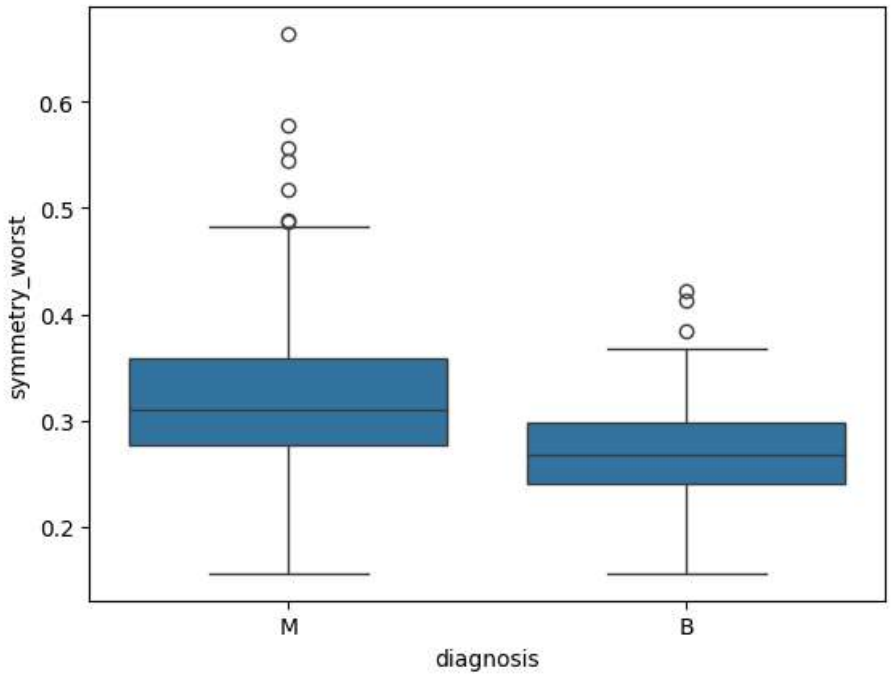

smoothness_worst
0.18
0.16
0.14
0.12
0.10
0.08
M
B
diagnosis

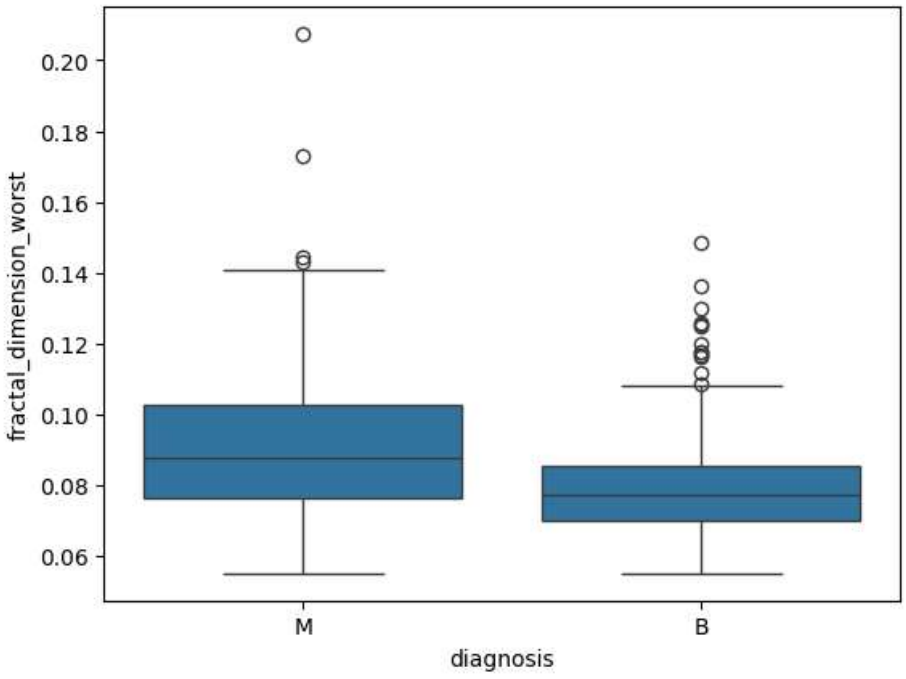

compactness_worst
1.0
0.8
0.6
0.4
0.2
0.0
M
B
diagnosis


concavity_worst
1.2
1.0
0.8
0.6
0.4
0.2
0.0
M
B
diagnosis


0.30
0.25


concave points_worst
0.20
0.15
0.10
0.05
0.00
M
B
diagnosis


symmetry_worst
0.6
0.5
0.4
0.3
0.2
M
B
diagnosis


fractal_dimension_worst
0.20
0.18
0.16
0.14
0.12
0.10
0.08
0.06
M
B
diagnosis

```
import pandas as pd

# Assuming 'df' is your DataFrame
df = pd.read_csv('/content/sample_data/data.csv')

# Check for missing values in the entire DataFrame
print(df.isnull().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
```

```
import pandas as pd

# Assuming 'df' is your DataFrame
df = pd.read_csv('/content/sample_data/data.csv')

# Drop the rows where at least one element is missing
df = df.dropna()

# Now, 'df' is your DataFrame with rows containing null values dropped
```

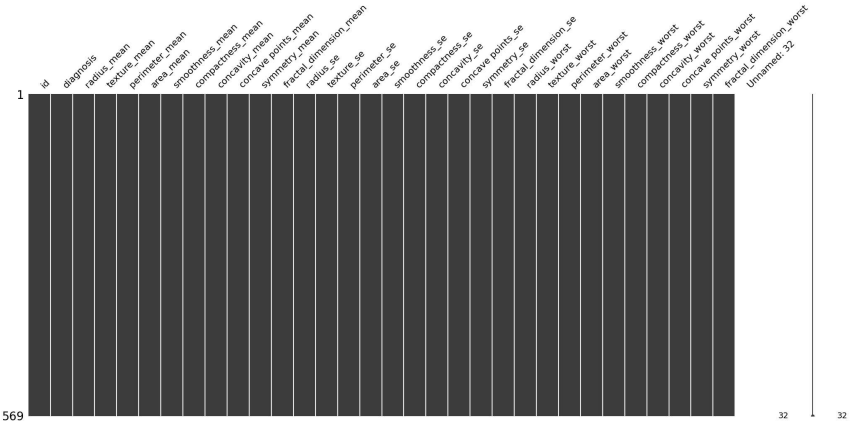
```
!pip install missingno
import missingno as msno
```

```
Requirement already satisfied: missingno in /usr/local/lib/python3.10/dist-packages (0.5.2)
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (from missingno) (1.25.2)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-packages (from missingno) (3.7.1
Requirement already satisfied: scipy in /usr/local/lib/python3.10/dist-packages (from missingno) (1.11.4)
Requirement already satisfied: seaborn in /usr/local/lib/python3.10/dist-packages (from missingno) (0.13.1)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib-
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from matplotlib->mis
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->mi
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib-
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.10/dist-packages (from matplot
```

```
Requirement already satisfied: pandas>=1.2 in /usr/local/lib/python3.10/dist-packages (from seaborn->missing
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=1.2->se
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.
```

```
msno.matrix(df)
```

```
<Axes: >
```




```
from sklearn.impute import SimpleImputer

imputer = SimpleImputer(strategy='mean')
X = imputer.fit_transform(X)
```

Training and testing the model

```
# Split the data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Train the model
model.fit(X_train, y_train)

# Evaluate the model
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy*100:.2f}%")
```

```
Accuracy: 96.49%
```