16yqrfnea

April 24, 2023

#Importing the Dependencies

[ ]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import LabelEncoder
```

1 Data Collection

[ ]:
```
# loading the breast cancer dataset from csv file to pandas data frame
breast_cancer_data = pd.read_csv('/content/data.csv')
```

2 Exploratory Data Analysis

[ ]:
```
# printing the first five rows of the dataframe
breast_cancer_data.head()
```

[ ]:
```
         id diagnosis  …  fractal_dimension_worst  Unnamed: 32
0    842302         M  …                  0.11890          NaN
1    842517         M  …                  0.08902          NaN
2  84300903         M  …                  0.08758          NaN
3  84348301         M  …                  0.17300          NaN
4  84358402         M  …                  0.07678          NaN

[5 rows x 33 columns]
```

[ ]:
```
# removing the unnmaed column
breast_cancer_data.drop(columns='Unnamed: 32', axis = 1, inplace=True)
```

[ ]:
```
breast_cancer_data.head()
```

[ ]:
```
         id diagnosis  …  symmetry_worst  fractal_dimension_worst
0    842302         M  …          0.4601                  0.11890
1    842517         M  …          0.2750                  0.08902
2  84300903         M  …          0.3613                  0.08758
```

```
3  84348301         M  …          0.6638                  0.17300
4  84358402         M  …          0.2364                  0.07678

[5 rows x 32 columns]
```

[ ]:
```
breast_cancer_data.shape
```

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

[ ]:
```
# checking the data types
breast_cancer_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 32 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
dtypes: float64(30), int64(1), object(1)
memory usage: 142.4+ KB
```

[ ]:
```
# removing the id column
breast_cancer_data.drop(columns='id', axis=1, inplace=True)
```

Diagnosis column is a CATEGORICAL columnm whereas remIning are continuous values

[ ]:
```
# checking for missing values
breast_cancer_data.isnull().sum()
```

[ ]:
```
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
dtype: int64
```

As we can see, the dataset has no missing values

Statistical summary of the data - Descriptive Statistics

[ ]:
```
breast_cancer_data.describe()
```

[ ]:
```
       radius_mean  texture_mean  …  symmetry_worst  fractal_dimension_worst
count   569.000000    569.000000  …      569.000000               569.000000
mean     14.127292     19.289649  …        0.290076                 0.083946
std       3.524049      4.301036  …        0.061867                 0.018061
min       6.981000      9.710000  …        0.156500                 0.055040
25%      11.700000     16.170000  …        0.250400                 0.071460
50%      13.370000     18.840000  …        0.282200                 0.080040
75%      15.780000     21.800000  …        0.317900                 0.092080
max      28.110000     39.280000  …        0.663800                 0.207500

[8 rows x 30 columns]
```

Check whether mean & median (50th Percentile) are close to each other

Checkin the distribution of target Variable

[ ]:
```
breast_cancer_data['diagnosis'].value_counts()
```

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

[ ]:
```
# encoding the target column
label_encode = LabelEncoder()

labels = label_encode.fit_transform(breast_cancer_data['diagnosis'])

breast_cancer_data['target'] = labels

breast_cancer_data.drop(columns='diagnosis', axis=1, inplace=True)
```

[ ]:
```
# diagnosis column removed
breast_cancer_data.head()
```

[ ]:
```
   radius_mean  texture_mean  …  fractal_dimension_worst  target
0        17.99         10.38  …                  0.11890       1
1        20.57         17.77  …                  0.08902       1
2        19.69         21.25  …                  0.08758       1
3        11.42         20.38  …                  0.17300       1
4        20.29         14.34  …                  0.07678       1

[5 rows x 31 columns]
```

[ ]:
```
breast_cancer_data['target'].value_counts()
```

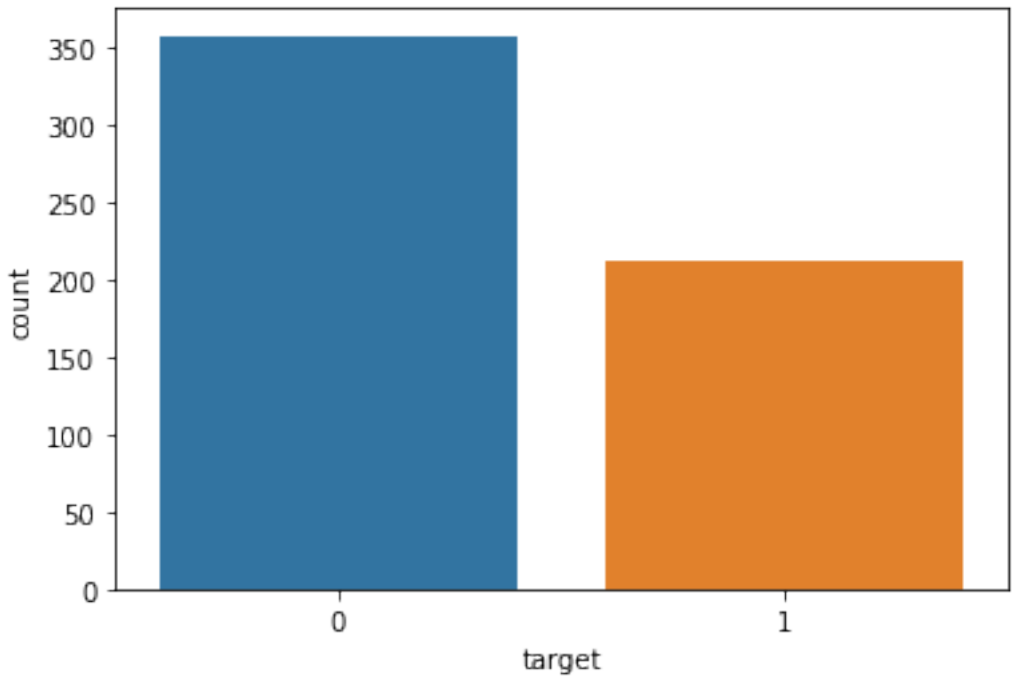
[ ]:
```
0    357
1    212
Name: target, dtype: int64
```

Benign –> 0

Malignant –> 1

[ ]:
```
sns.countplot(x='target', data=breast_cancer_data)
```

[ ]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f79da92c890>
```



There is a slight imbalance in the data. But it is fine in this case

Grouping the data based on the target

[ ]:
```
breast_cancer_data.groupby('target').mean()
```

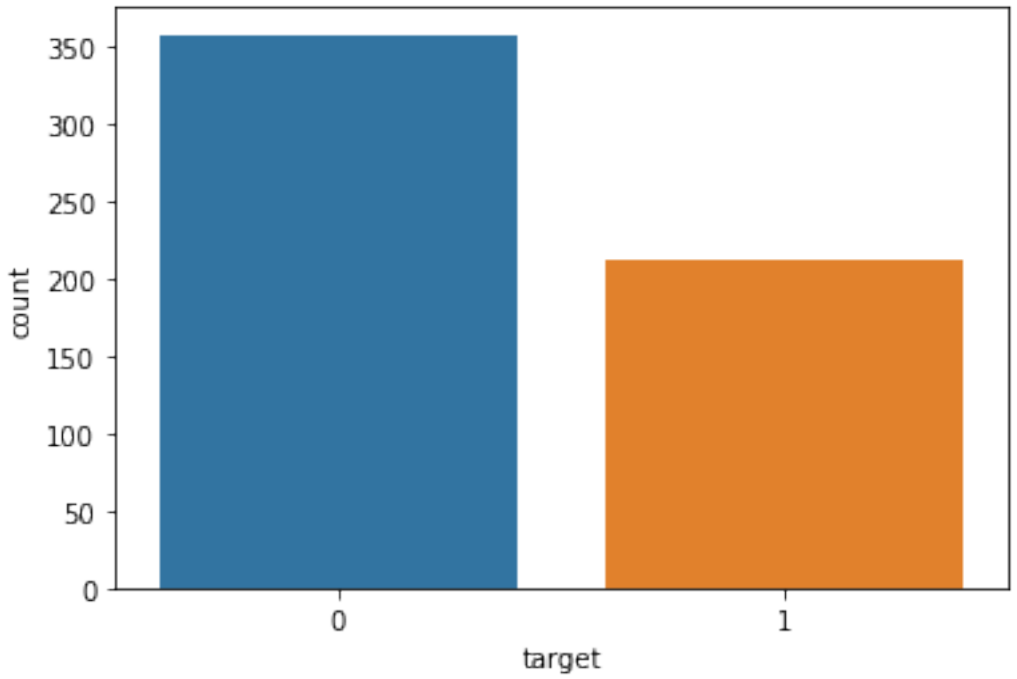
[ ]:
```
        radius_mean  texture_mean  …  symmetry_worst  fractal_dimension_worst
target                             …
0         12.146524     17.914762  …        0.270246                 0.079442
1         17.462830     21.604906  …        0.323468                 0.091530

[2 rows x 30 columns]
```

Inference: We can clearly see that for most of the features, the mean values are higher for Malignant(1) cases and lower for Benign(0) cases

3 Data Visualization

[ ]:
```
# countplot for the target column for checkin gthe distribution of target
sns.countplot(x='target', data=breast_cancer_data)
```

[ ]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f79da413d10>
```



Distribution plot for all columns

[ ]:
```
# this is how we can get all the column names of the dataframe
for column in breast_cancer_data:
  print(column)
```

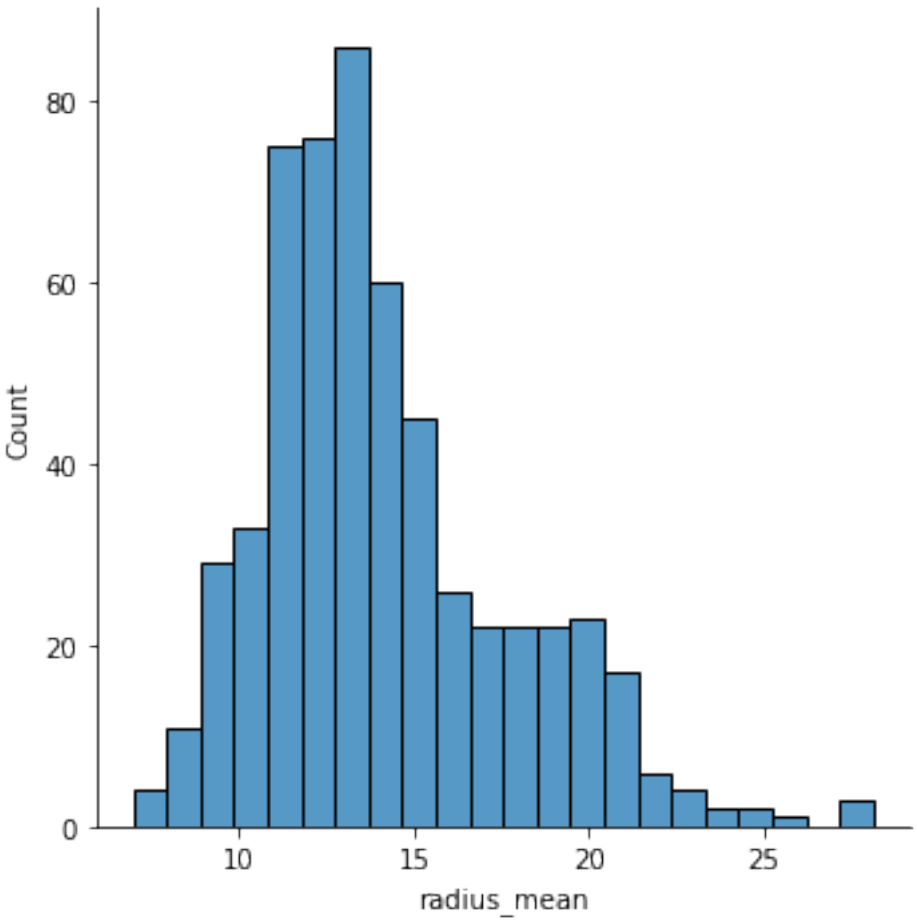
```
radius_mean
texture_mean
perimeter_mean
area_mean
smoothness_mean
compactness_mean
concavity_mean
concave points_mean
symmetry_mean
fractal_dimension_mean
radius_se
texture_se
perimeter_se
```

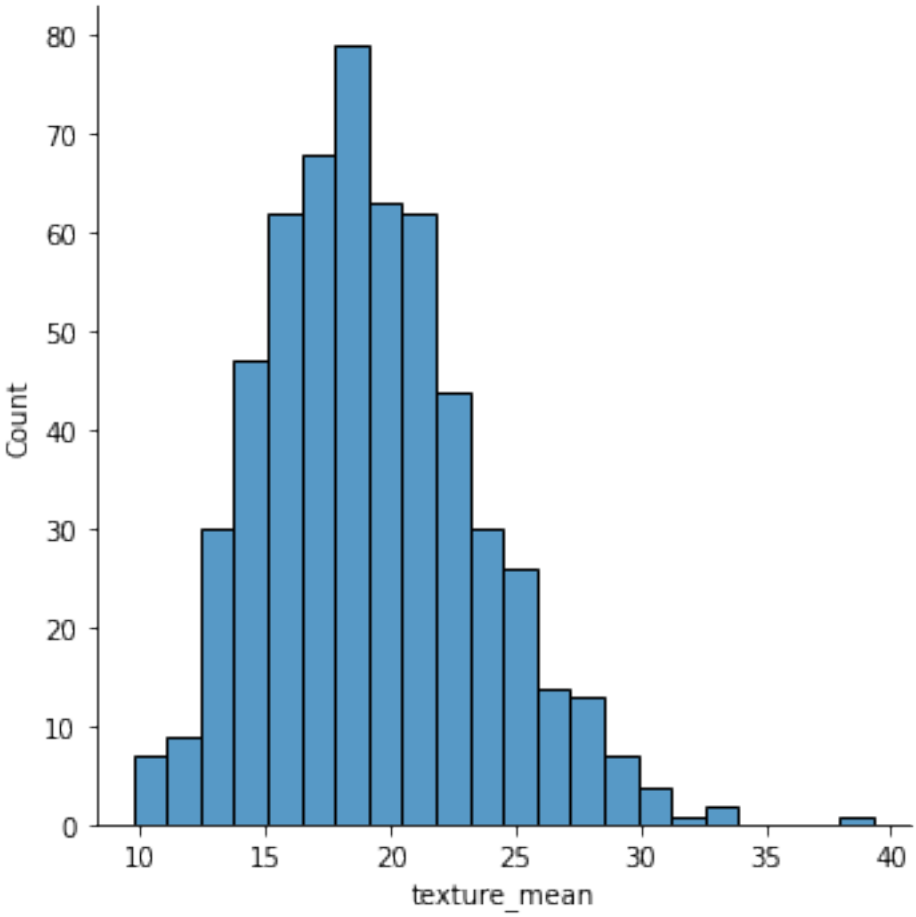
```
area_se
smoothness_se
compactness_se
concavity_se
concave points_se
symmetry_se
fractal_dimension_se
radius_worst
texture_worst
perimeter_worst
area_worst
smoothness_worst
compactness_worst
concavity_worst
concave points_worst
symmetry_worst
fractal_dimension_worst
target
```

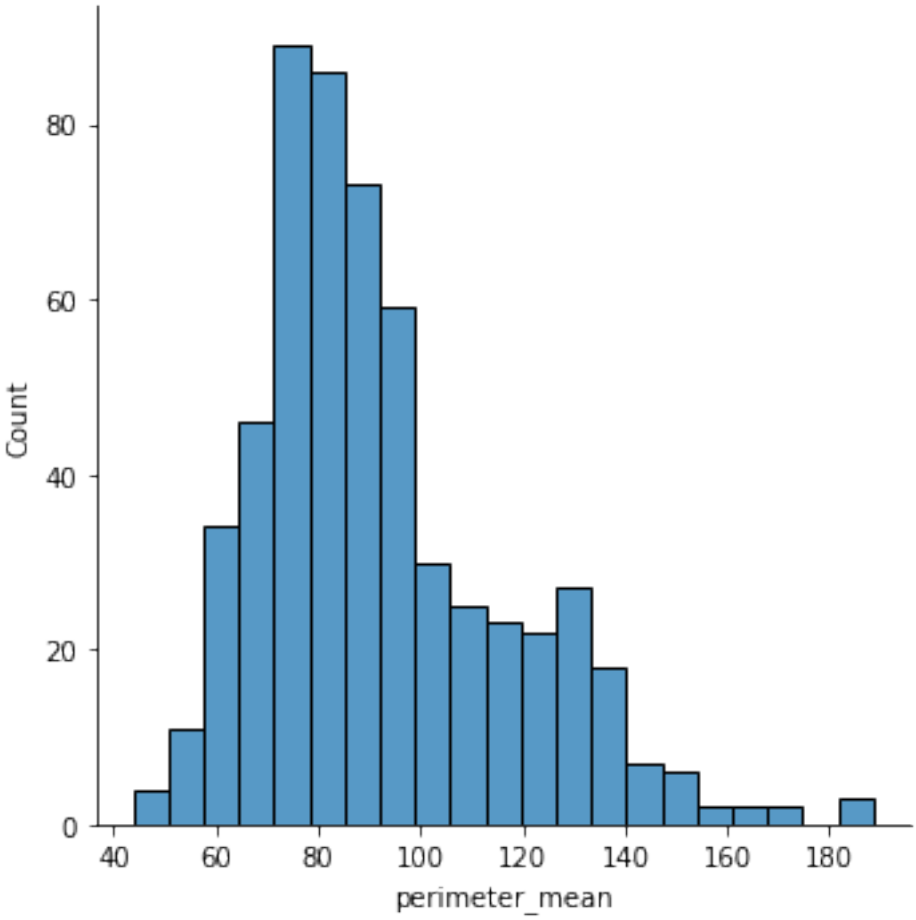
[ ]:
```python
# creating a for loop to get the distribution plot for all columns
for column in breast_cancer_data:
  sns.displot(x=column, data=breast_cancer_data)
```

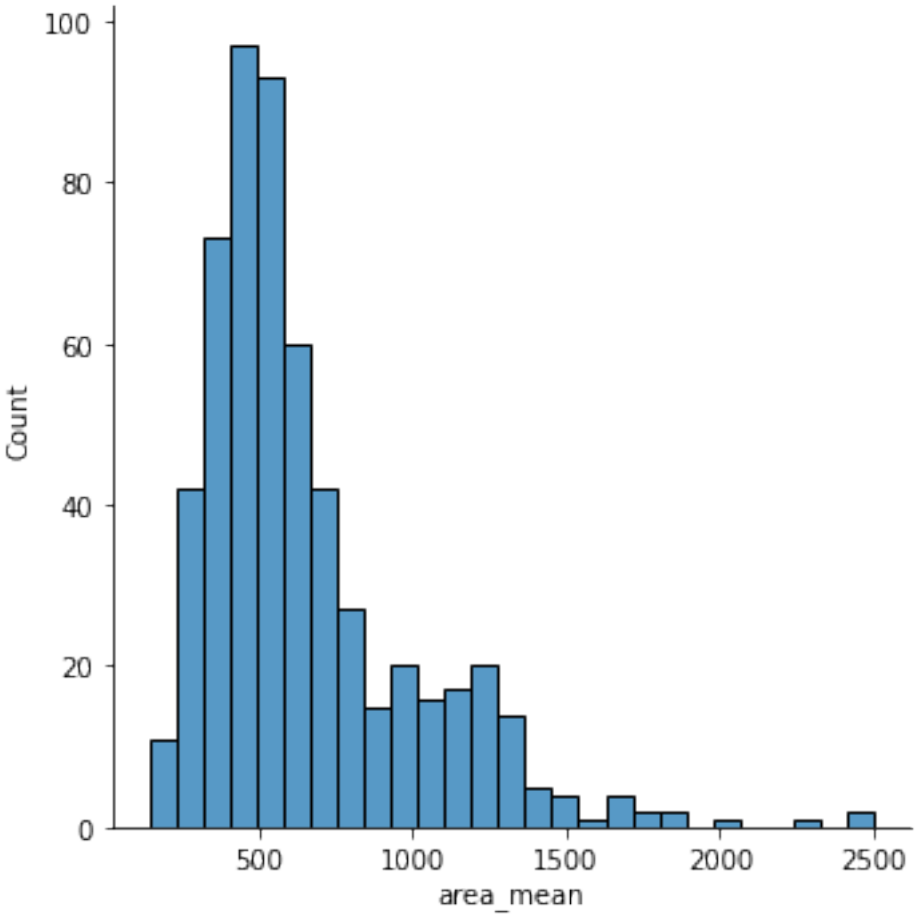
```
/usr/local/lib/python3.7/dist-packages/seaborn/axisgrid.py:409: RuntimeWarning:
More than 20 figures have been opened. Figures created through the pyplot
interface (`matplotlib.pyplot.figure`) are retained until explicitly closed and
may consume too much memory. (To control this warning, see the rcParam
`figure.max_open_warning`).
  fig = plt.figure(figsize=figsize)
/usr/local/lib/python3.7/dist-packages/seaborn/axisgrid.py:409: RuntimeWarning:
More than 20 figures have been opened. Figures created through the pyplot
interface (`matplotlib.pyplot.figure`) are retained until explicitly closed and
may consume too much memory. (To control this warning, see the rcParam
`figure.max_open_warning`).
  fig = plt.figure(figsize=figsize)
/usr/local/lib/python3.7/dist-packages/seaborn/axisgrid.py:409: RuntimeWarning:
More than 20 figures have been opened. Figures created through the pyplot
interface (`matplotlib.pyplot.figure`) are retained until explicitly closed and
may consume too much memory. (To control this warning, see the rcParam
`figure.max_open_warning`).
  fig = plt.figure(figsize=figsize)
/usr/local/lib/python3.7/dist-packages/seaborn/axisgrid.py:409: RuntimeWarning:
More than 20 figures have been opened. Figures created through the pyplot
interface (`matplotlib.pyplot.figure`) are retained until explicitly closed and
may consume too much memory. (To control this warning, see the rcParam
`figure.max_open_warning`).
  fig = plt.figure(figsize=figsize)
/usr/local/lib/python3.7/dist-packages/seaborn/axisgrid.py:409: RuntimeWarning:
```

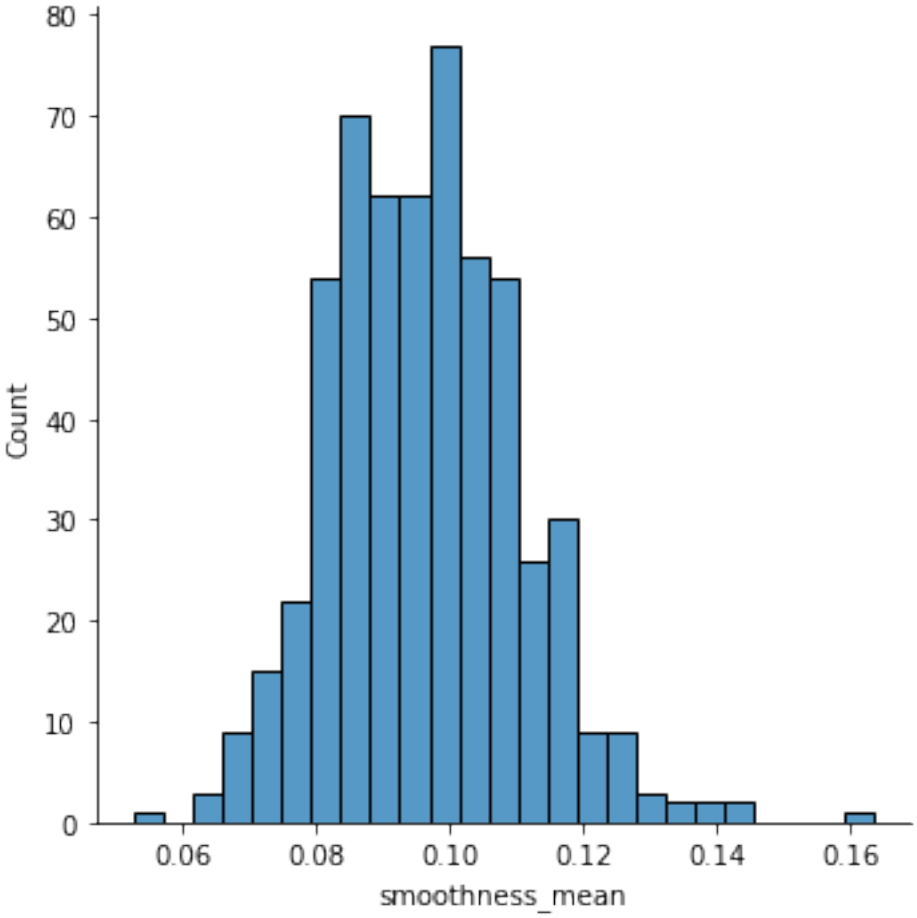
```
More than 20 figures have been opened. Figures created through the pyplot
interface (`matplotlib.pyplot.figure`) are retained until explicitly closed and
may consume too much memory. (To control this warning, see the rcParam
`figure.max_open_warning`).
  fig = plt.figure(figsize=figsize)
/usr/local/lib/python3.7/dist-packages/seaborn/axisgrid.py:409: RuntimeWarning:
More than 20 figures have been opened. Figures created through the pyplot
interface (`matplotlib.pyplot.figure`) are retained until explicitly closed and
may consume too much memory. (To control this warning, see the rcParam
`figure.max_open_warning`).
  fig = plt.figure(figsize=figsize)
/usr/local/lib/python3.7/dist-packages/seaborn/axisgrid.py:409: RuntimeWarning:
More than 20 figures have been opened. Figures created through the pyplot
interface (`matplotlib.pyplot.figure`) are retained until explicitly closed and
may consume too much memory. (To control this warning, see the rcParam
`figure.max_open_warning`).
  fig = plt.figure(figsize=figsize)
/usr/local/lib/python3.7/dist-packages/seaborn/axisgrid.py:409: RuntimeWarning:
More than 20 figures have been opened. Figures created through the pyplot
interface (`matplotlib.pyplot.figure`) are retained until explicitly closed and
may consume too much memory. (To control this warning, see the rcParam
`figure.max_open_warning`).
  fig = plt.figure(figsize=figsize)
/usr/local/lib/python3.7/dist-packages/seaborn/axisgrid.py:409: RuntimeWarning:
More than 20 figures have been opened. Figures created through the pyplot
interface (`matplotlib.pyplot.figure`) are retained until explicitly closed and
may consume too much memory. (To control this warning, see the rcParam
`figure.max_open_warning`).
  fig = plt.figure(figsize=figsize)
/usr/local/lib/python3.7/dist-packages/seaborn/axisgrid.py:409: RuntimeWarning:
More than 20 figures have been opened. Figures created through the pyplot
interface (`matplotlib.pyplot.figure`) are retained until explicitly closed and
may consume too much memory. (To control this warning, see the rcParam
`figure.max_open_warning`).
  fig = plt.figure(figsize=figsize)
/usr/local/lib/python3.7/dist-packages/seaborn/axisgrid.py:409: RuntimeWarning:
More than 20 figures have been opened. Figures created through the pyplot
interface (`matplotlib.pyplot.figure`) are retained until explicitly closed and
may consume too much memory. (To control this warning, see the rcParam
`figure.max_open_warning`).
  fig = plt.figure(figsize=figsize)
```

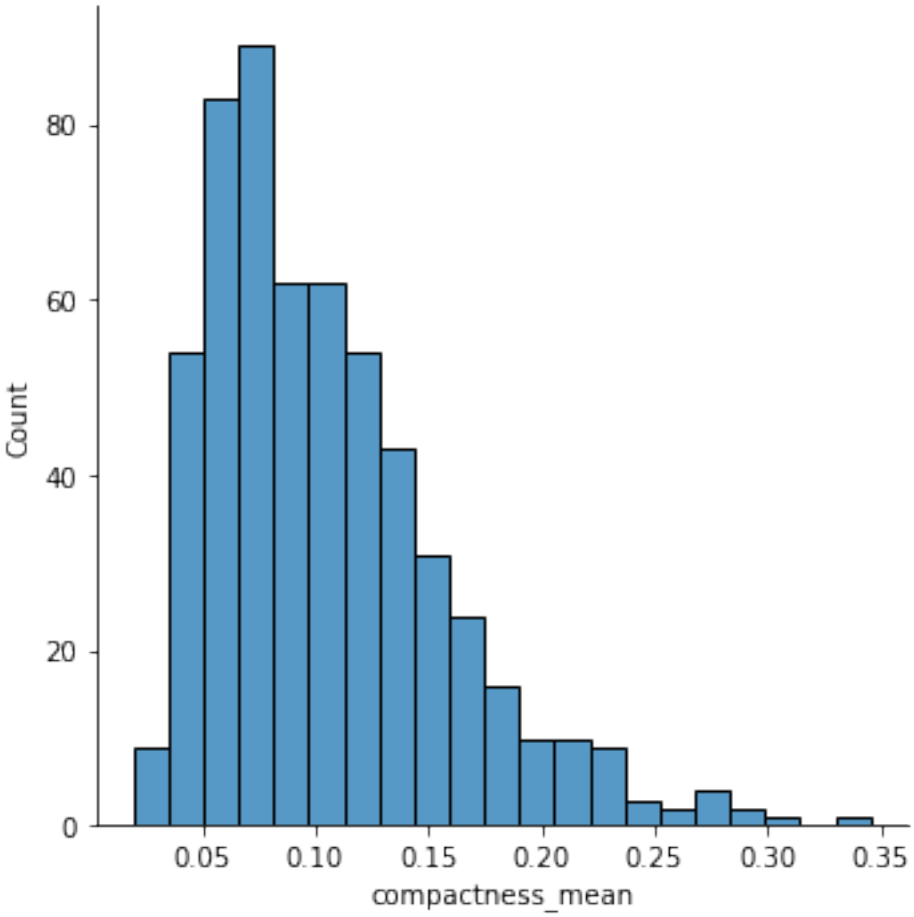

Count
80
60
40
20
0
10
15
20
25
radius_mean

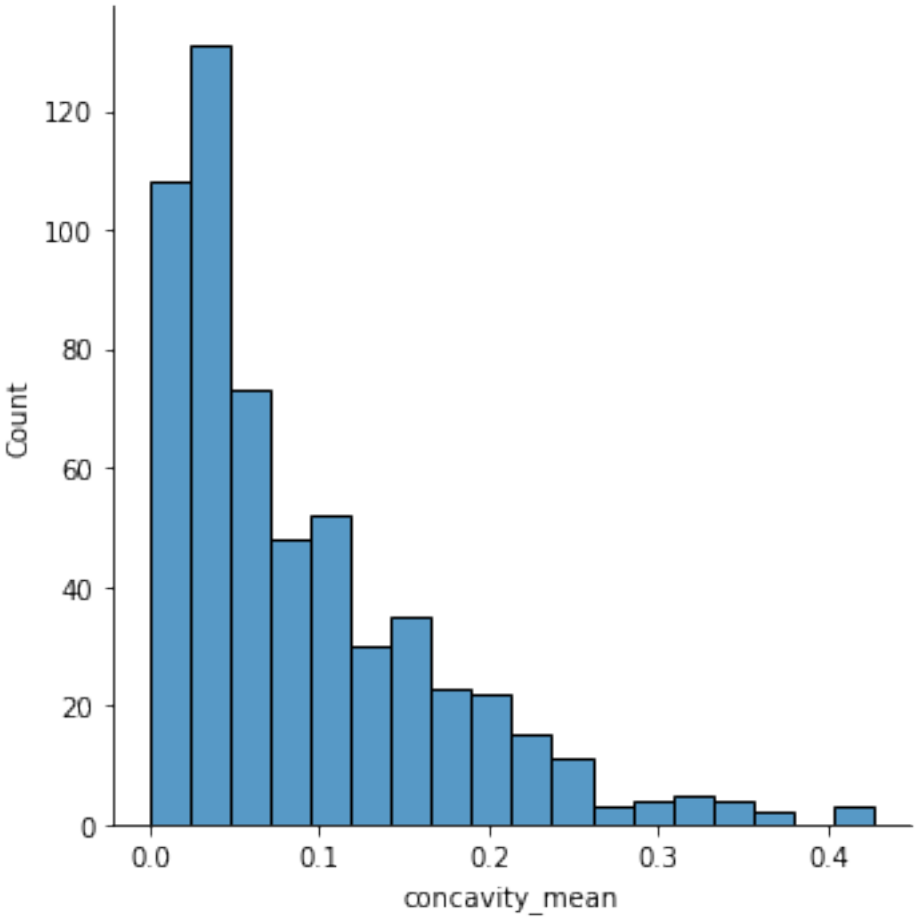

Count
texture_mean
0
10
20
30
40
50
60
70
80
10
15
20
25
30
35
40

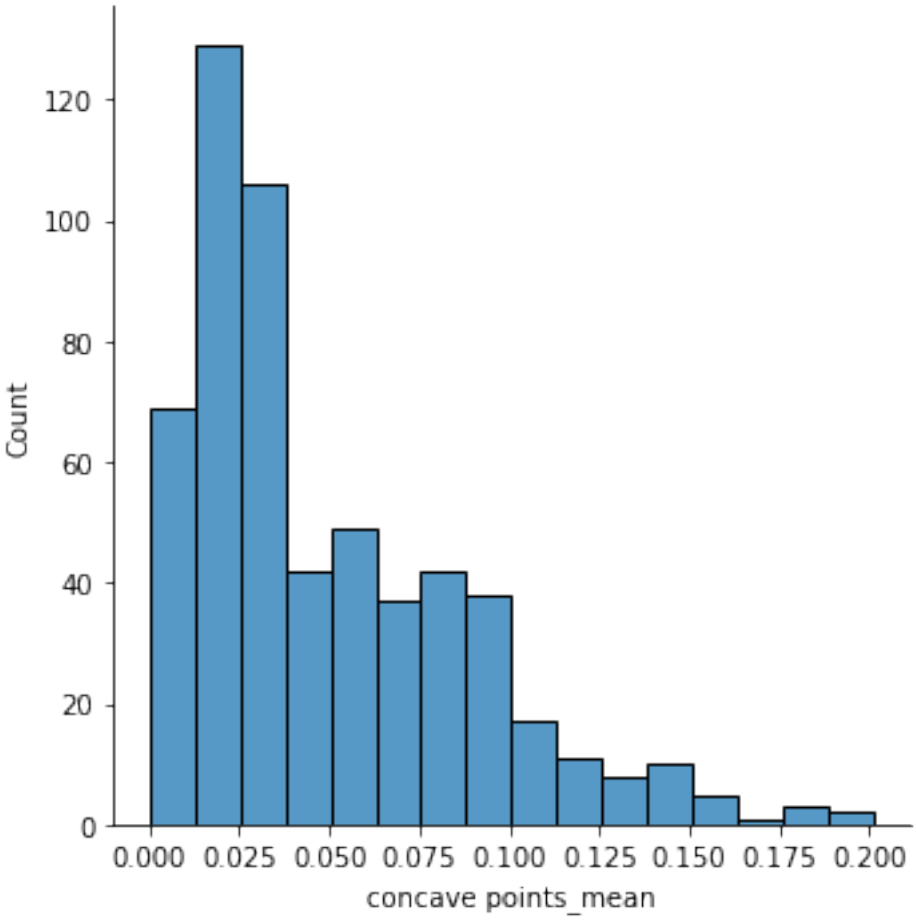

Count
0
20
40
60
80
40
60
80
100
120
140
160
180
perimeter_mean

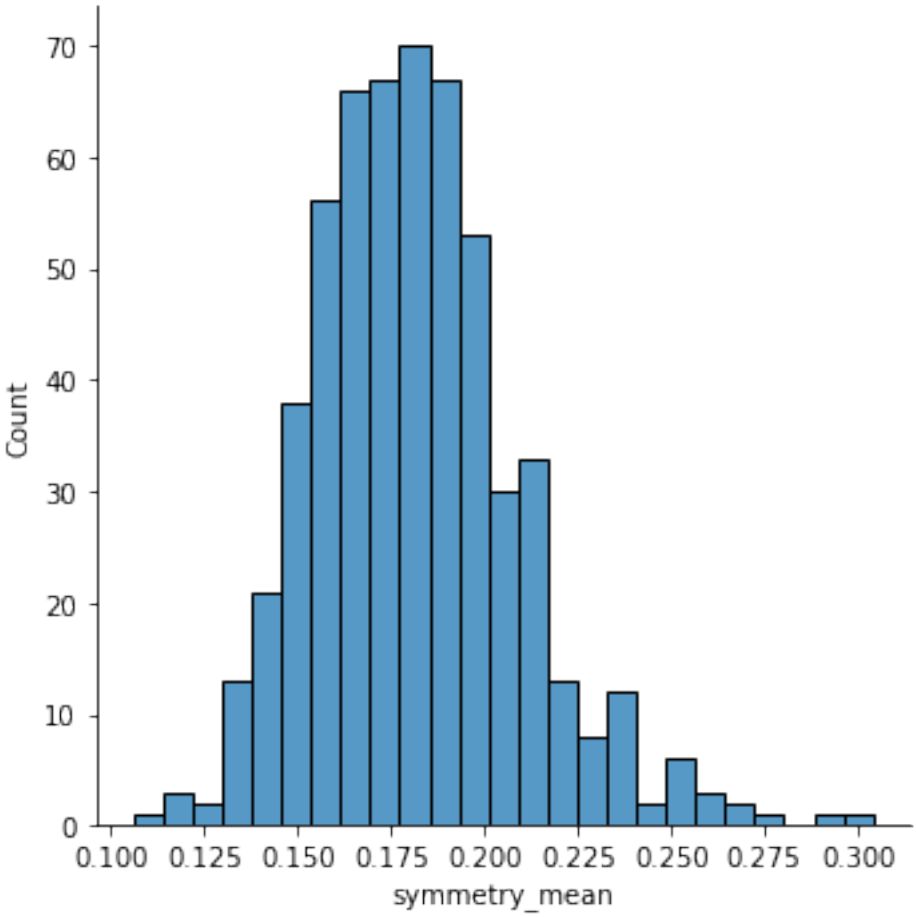

100
80
60
40
20
0
Count
500
1000
1500
2000
2500
area_mean

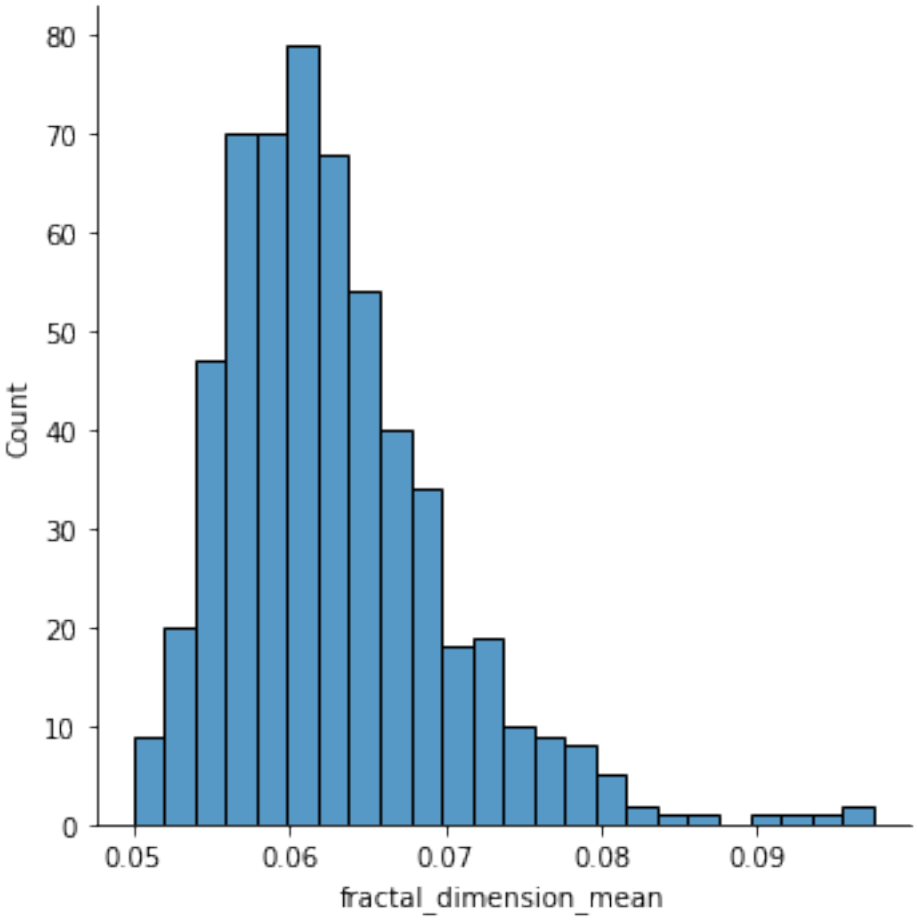

80
70
60
50
40
30
20
10
0
Count
0.06
0.08
0.10
0.12
0.14
0.16
smoothness_mean

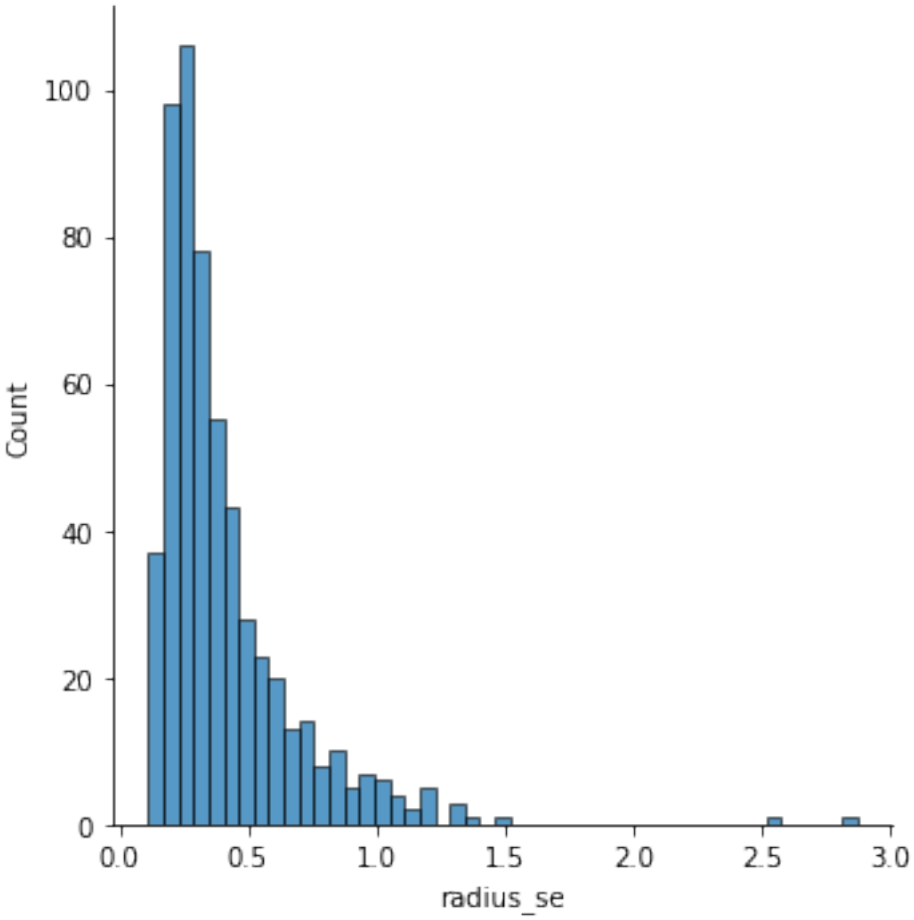

Count
80
60
40
20
0
0.05
0.10
0.15
0.20
0.25
0.30
0.35
compactness_mean

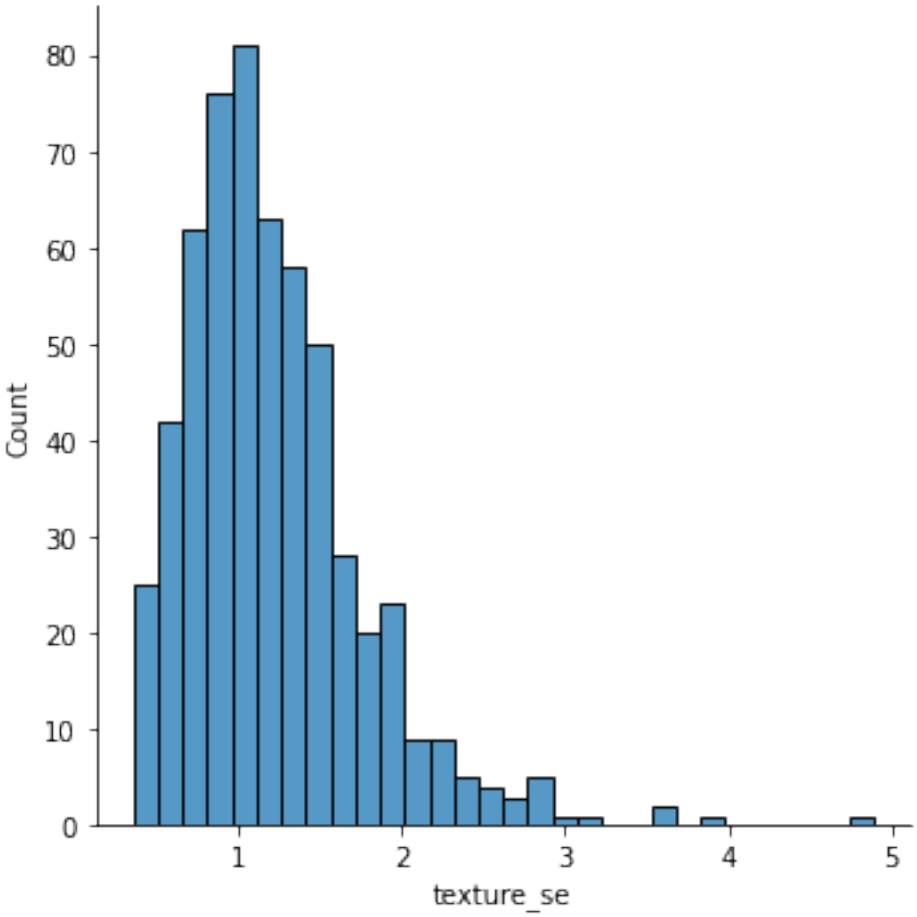

120
100
80
60
40
20
0
Count
0.0
0.1
0.2
0.3
0.4
concavity_mean

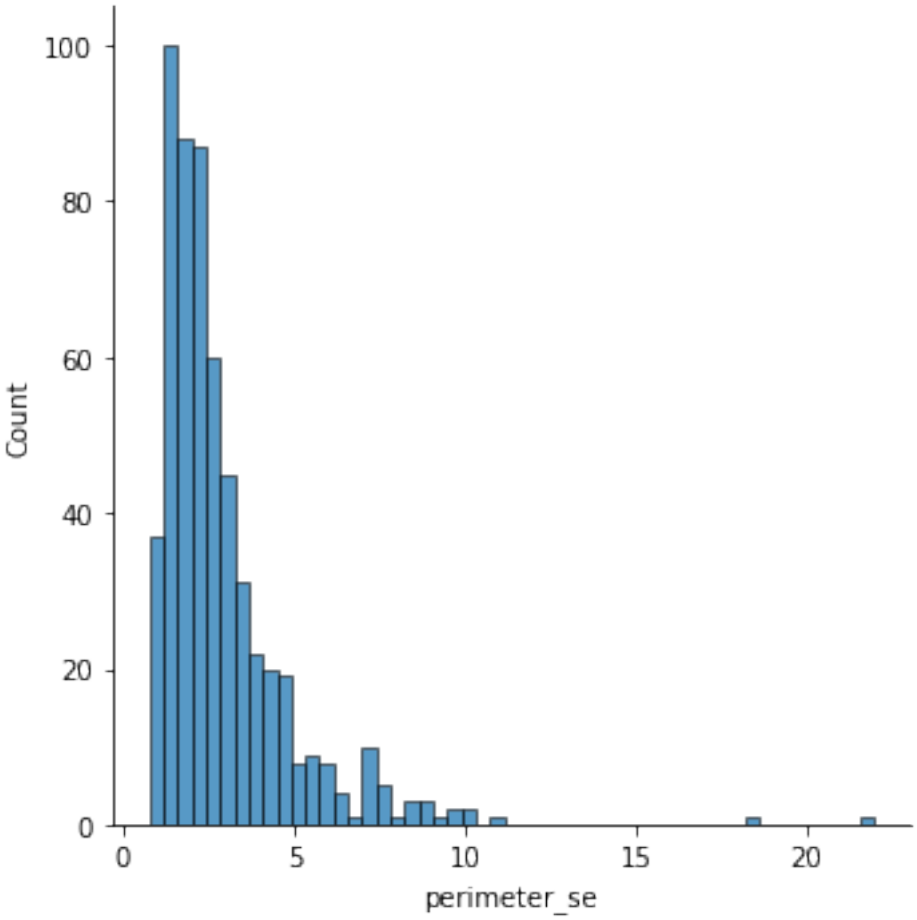

120
100
80
60
40
20
0
Count
0.000
0.025
0.050
0.075
0.100
0.125
0.150
0.175
0.200
concave points_mean

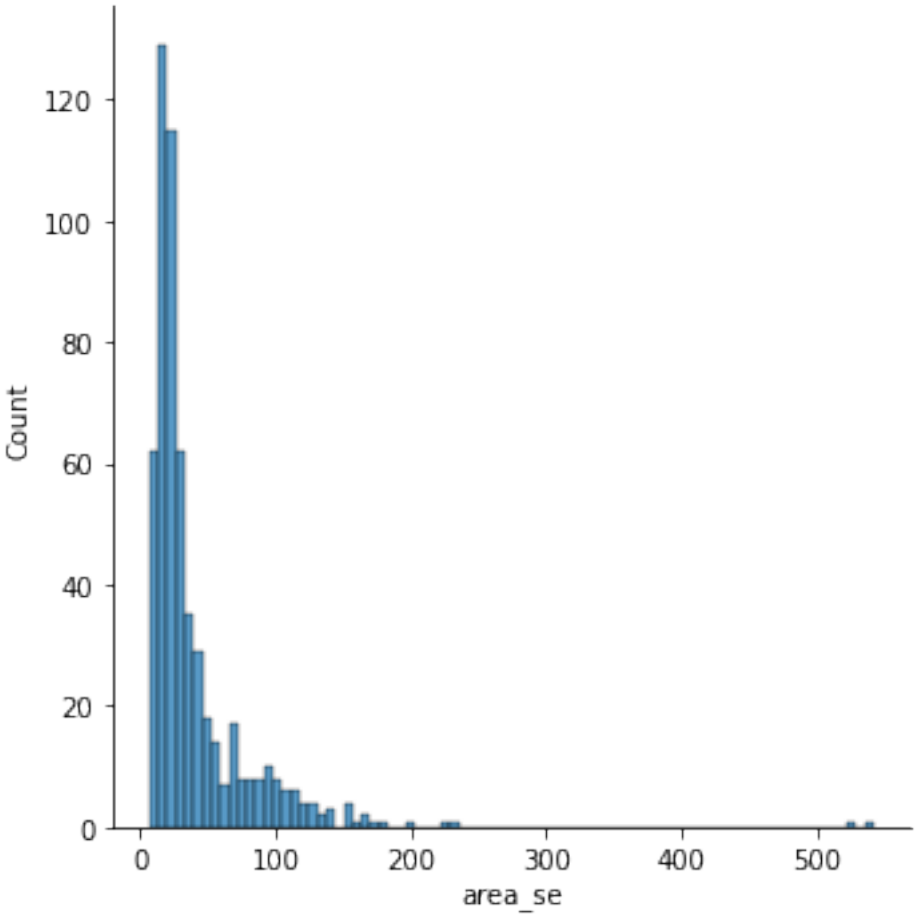

70
60
50
40
30
20
10
0
Count
0.100
0.125
0.150
0.175
0.200
0.225
0.250
0.275
0.300
symmetry_mean

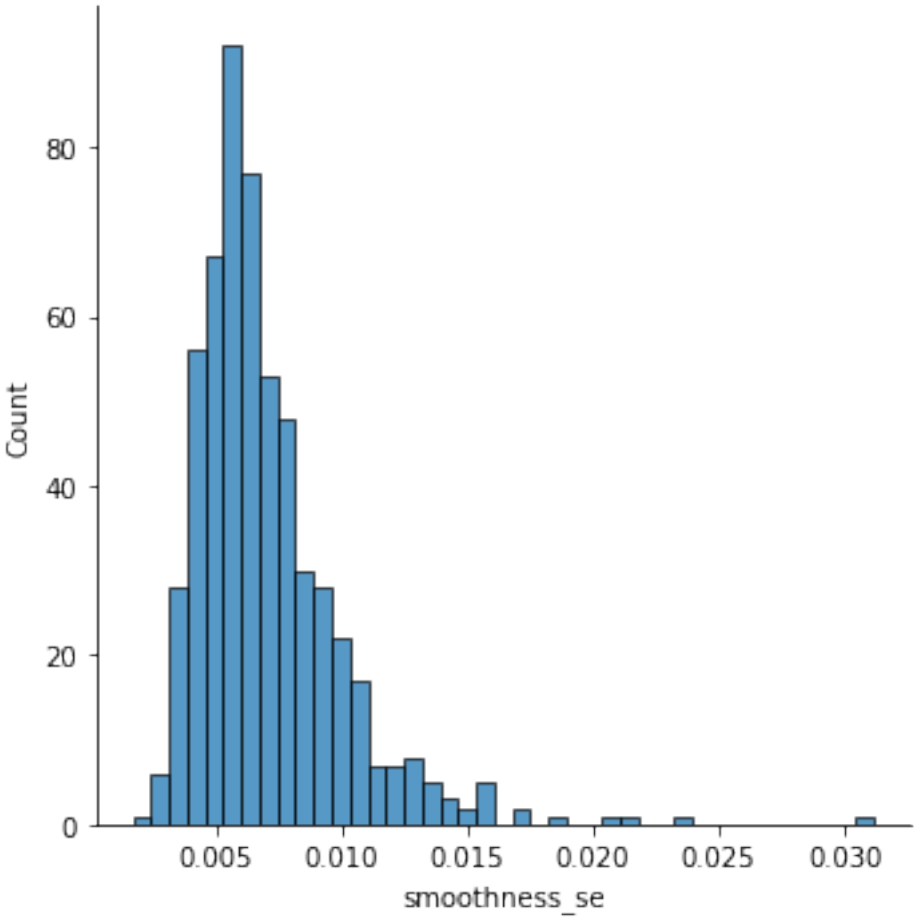

80
70
60
50
40
30
20
10
0
Count
0.05
0.06
0.07
0.08
0.09
fractal_dimension_mean

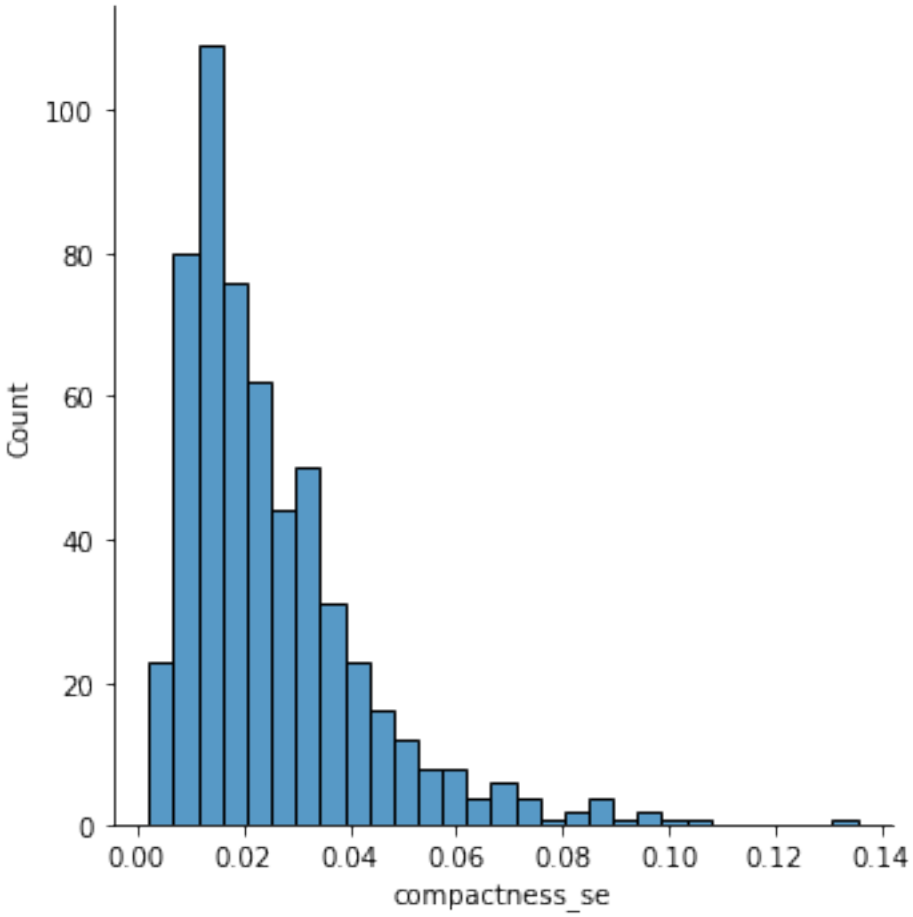

100
80
60
40
20
0
Count
0.0
0.5
1.0
1.5
2.0
2.5
3.0
radius_se

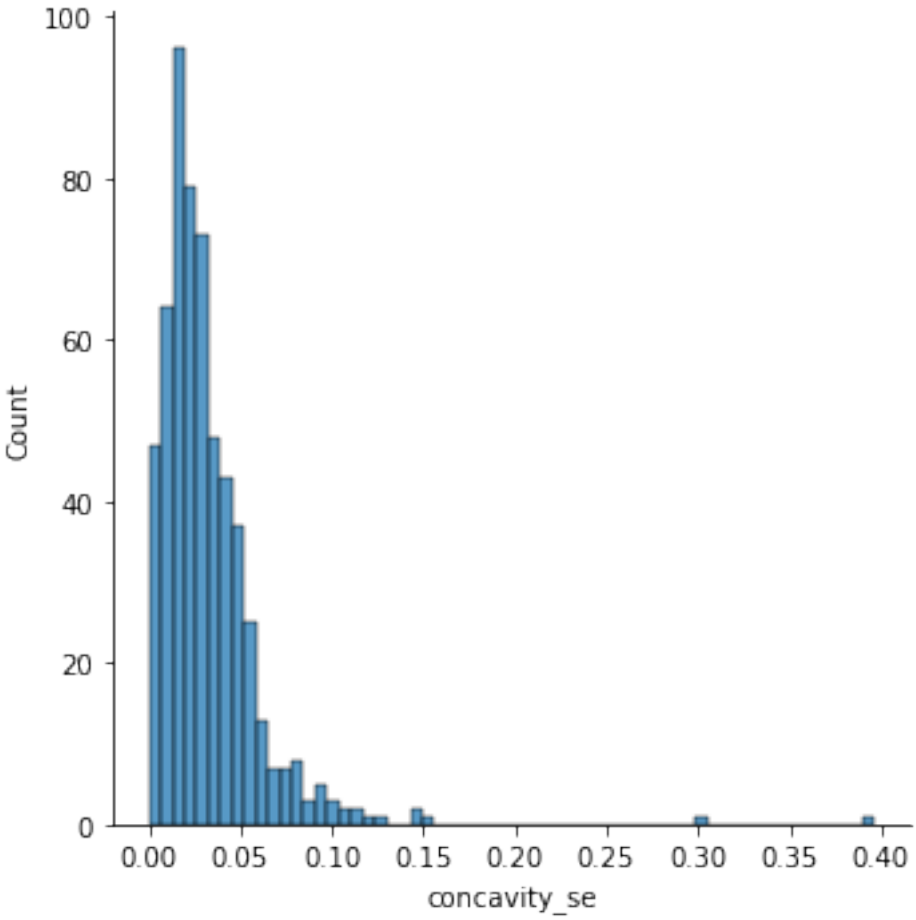

80
70
60
50
40
30
20
10
0
Count
1
2
3
4
5
texture_se

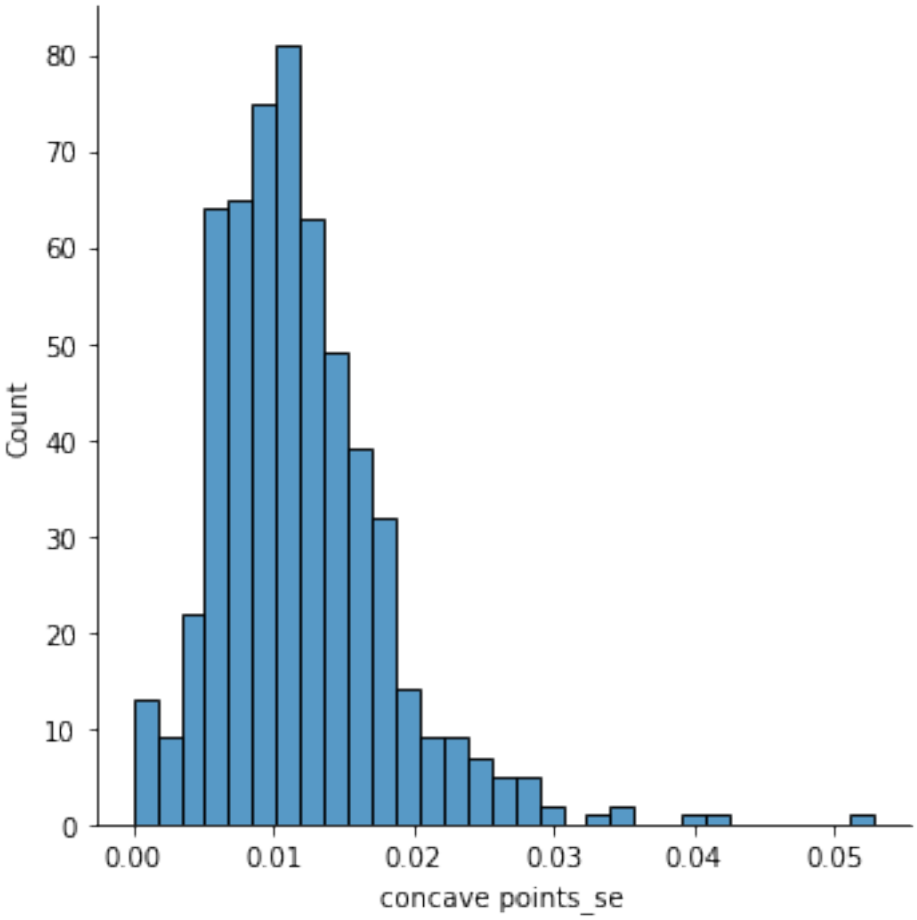

100
80
60
40
20
0
Count
0
5
10
15
20
perimeter_se

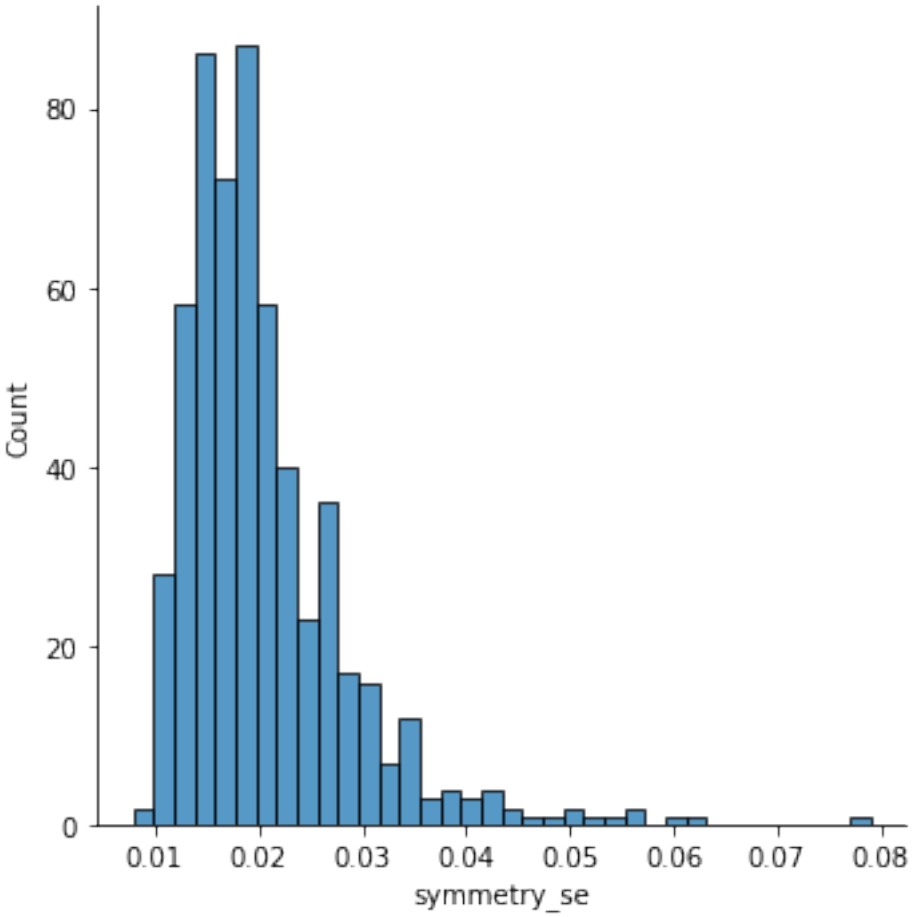

120
100
80
60
40
20
0
Count
0
100
200
300
400
500
area_se

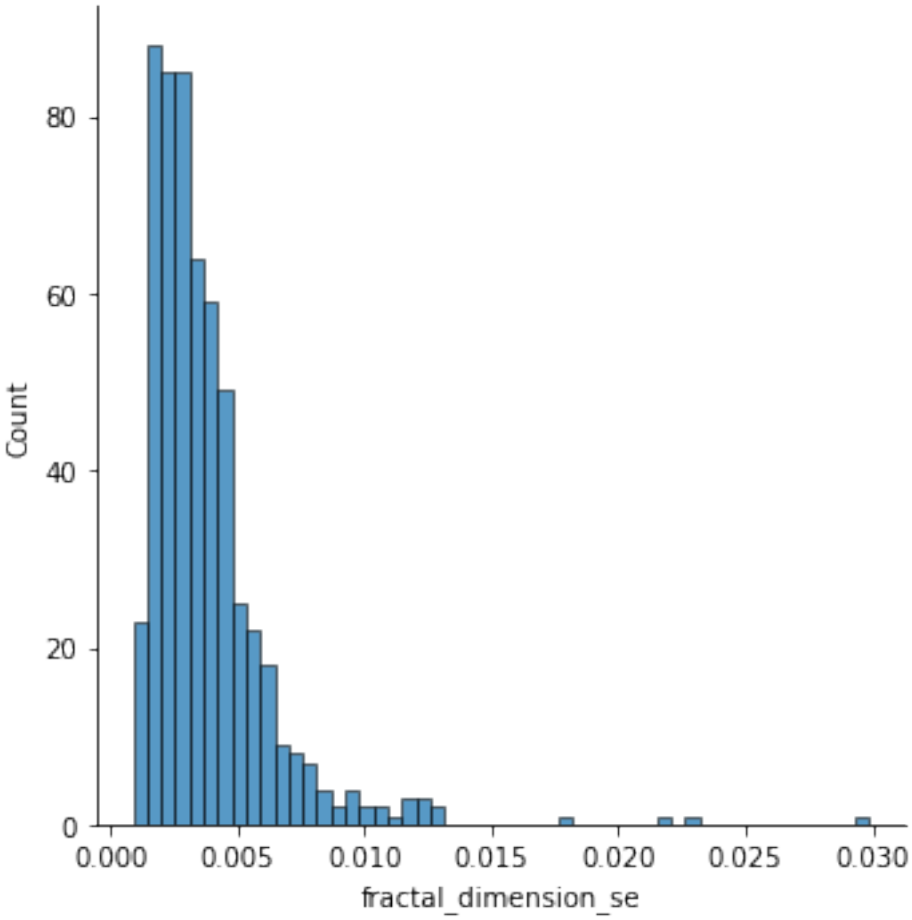

Count
80
60
40
20
0
0.005
0.010
0.015
0.020
0.025
0.030
smoothness_se

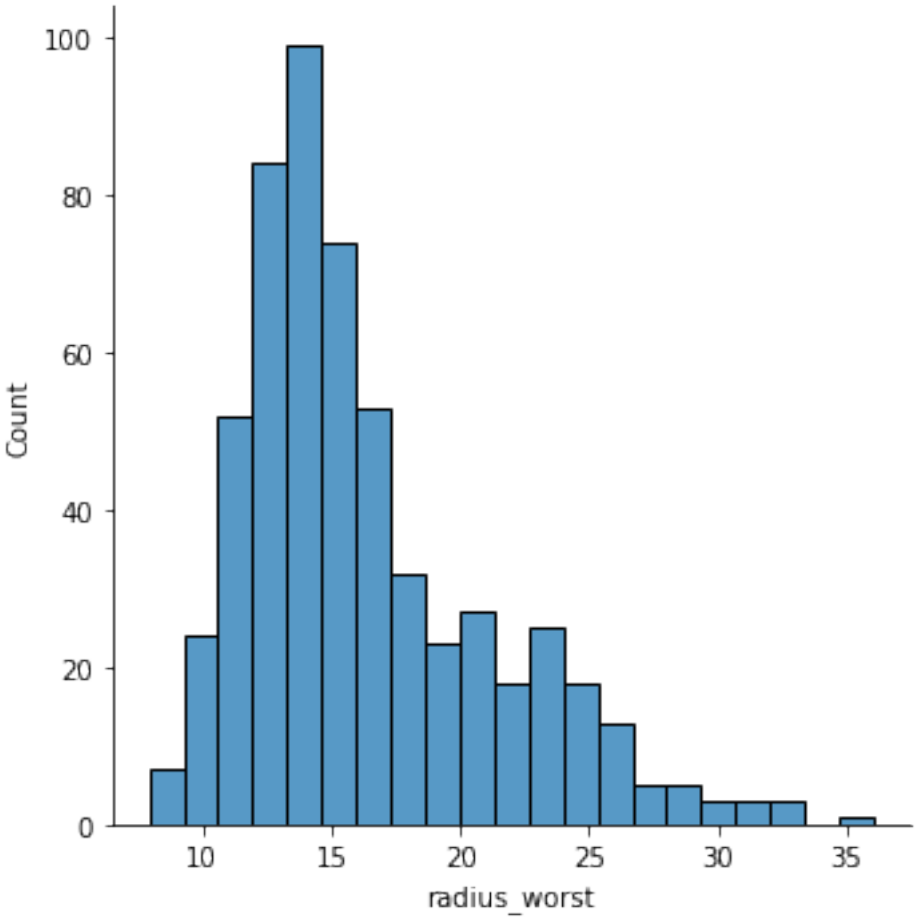

Count
100
80
60
40
20
0
0.00
0.02
0.04
0.06
0.08
0.10
0.12
0.14
compactness_se

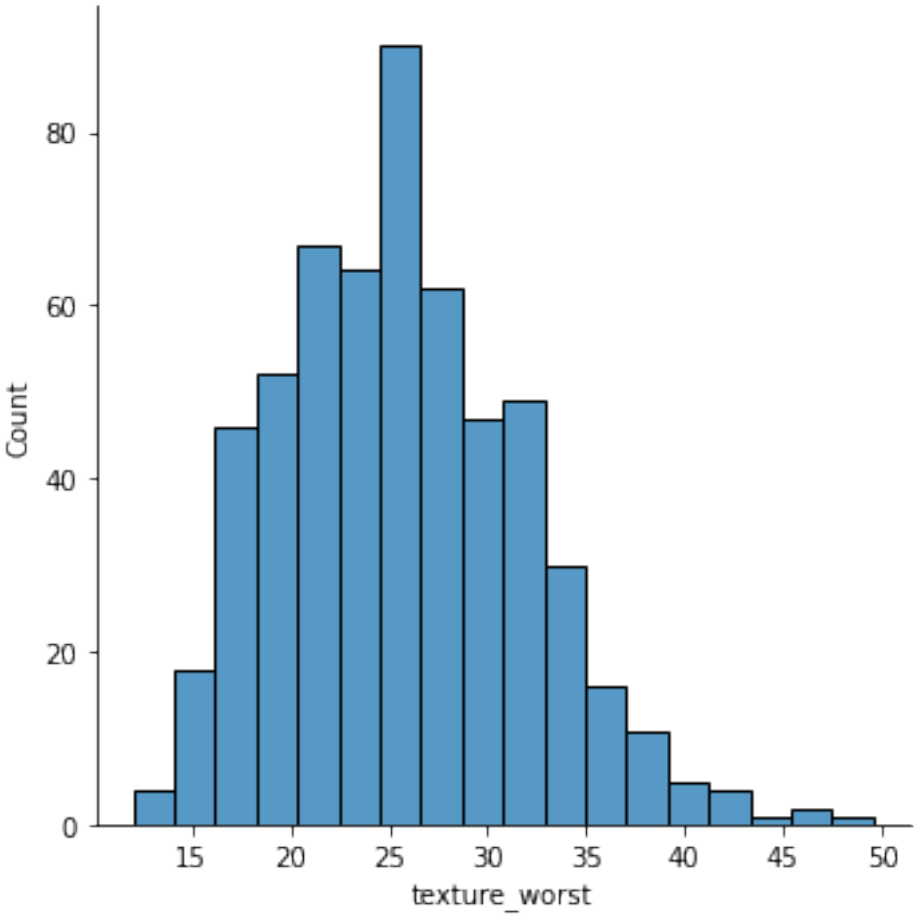

100
80
60
40
20
0
Count
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
0.40
concavity_se

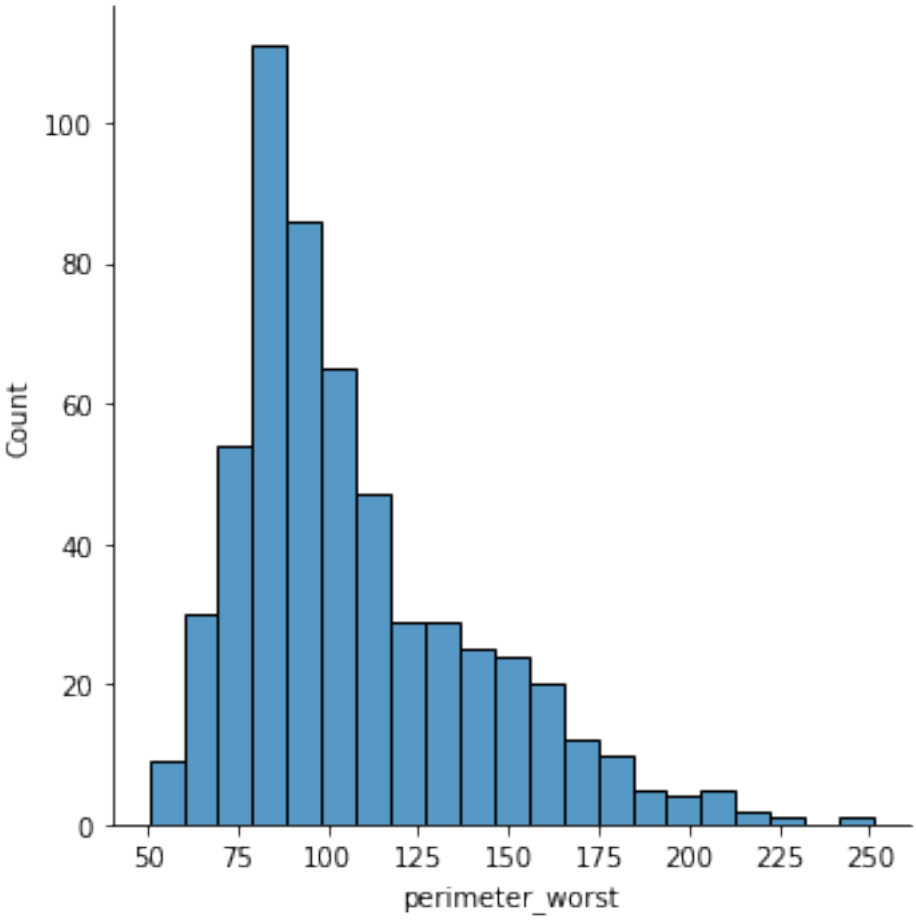

Count
80
70
60
50
40
30
20
10
0
0.00
0.01
0.02
0.03
0.04
0.05
concave points_se

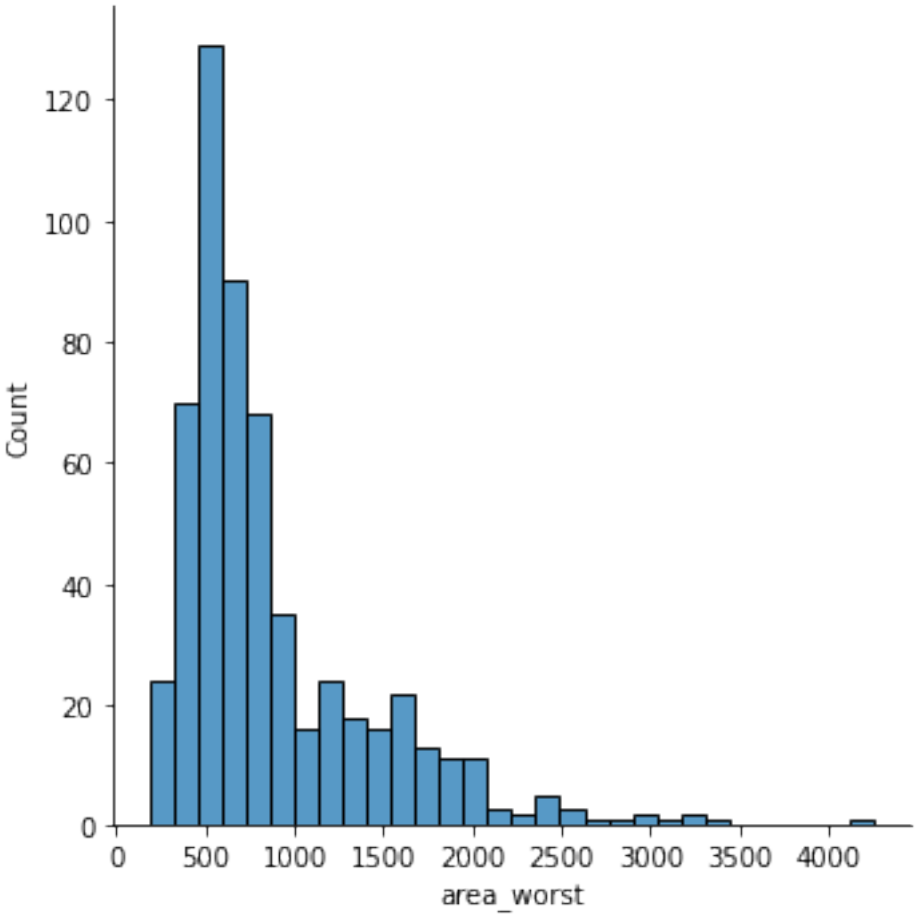

Count
80
60
40
20
0
0.01
0.02
0.03
0.04
0.05
0.06
0.07
0.08
symmetry_se

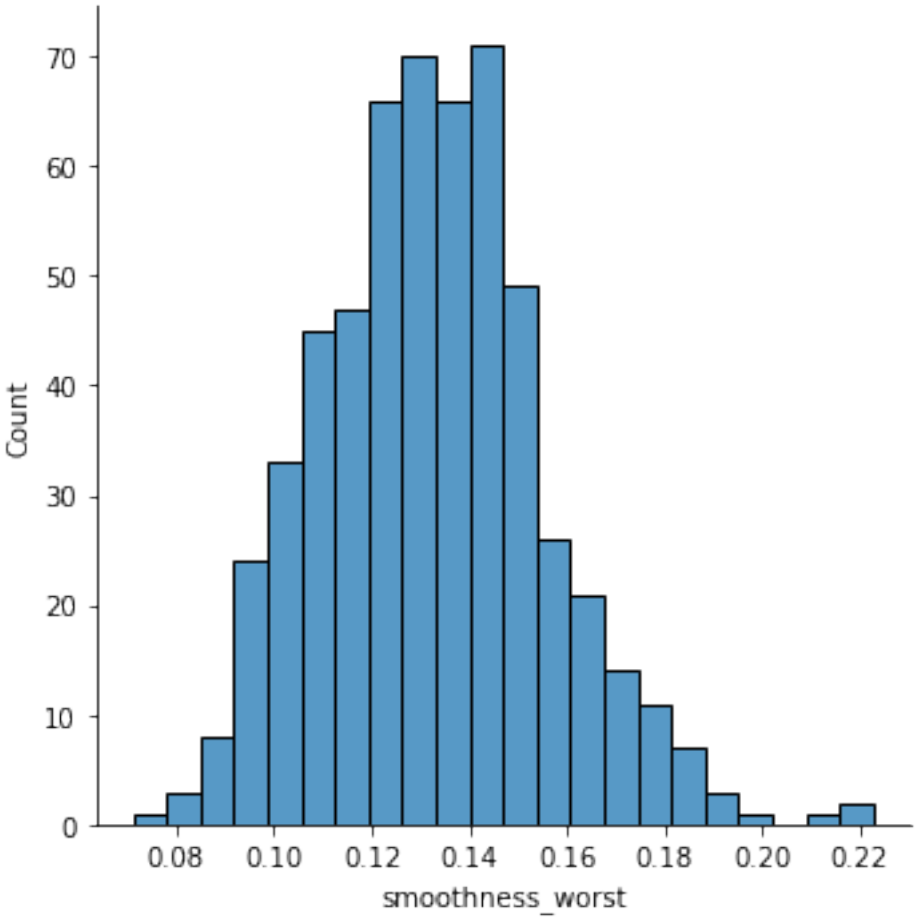

Count
80
60
40
20
0
0.000
0.005
0.010
0.015
0.020
0.025
0.030
fractal_dimension_se

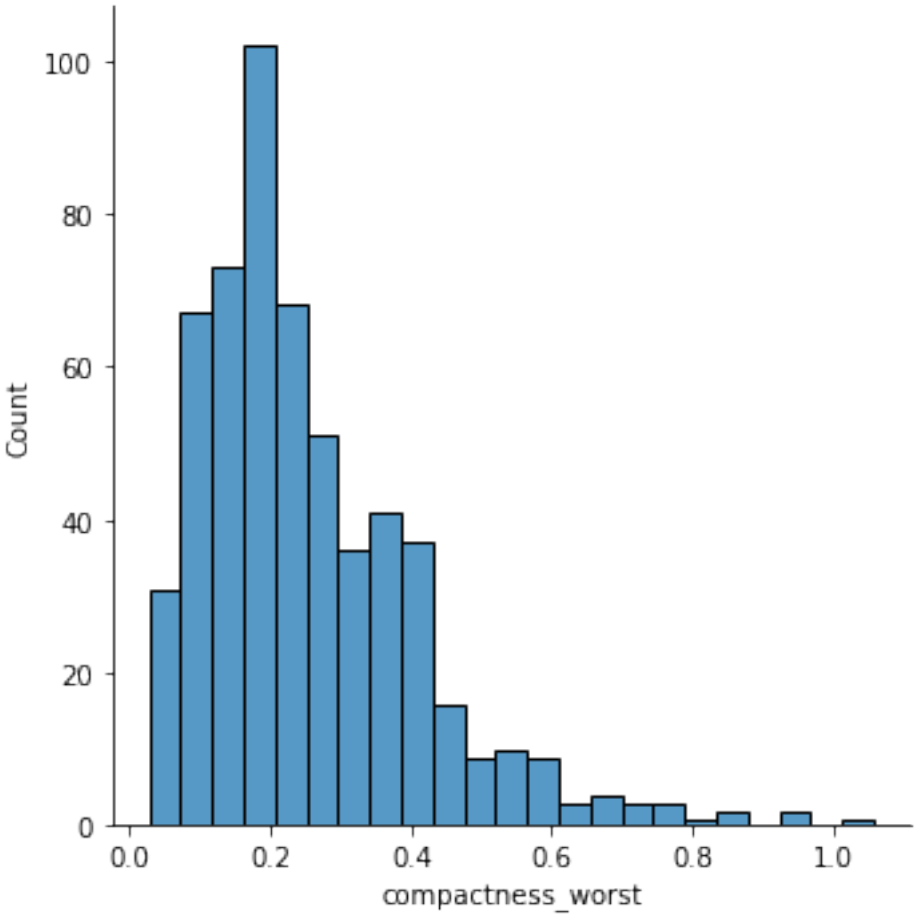

100
80
60
40
20
0
Count
10
15
20
25
30
35
radius_worst

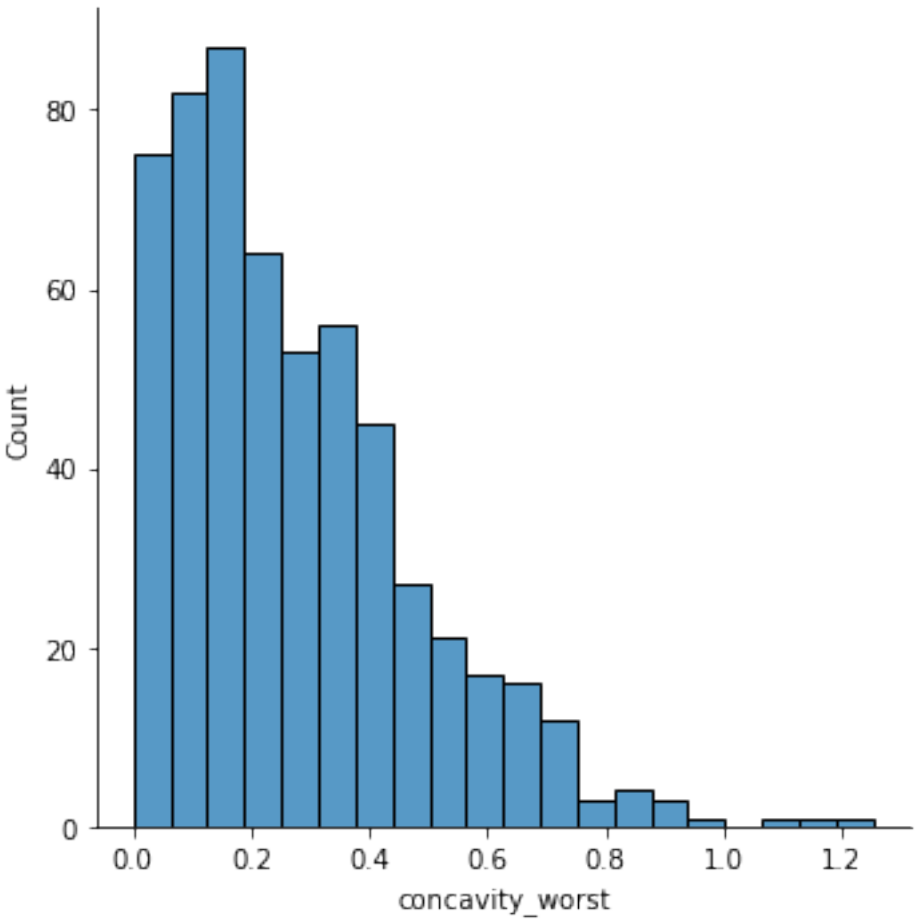

Count
80
60
40
20
0
15
20
25
30
35
40
45
50
texture_worst

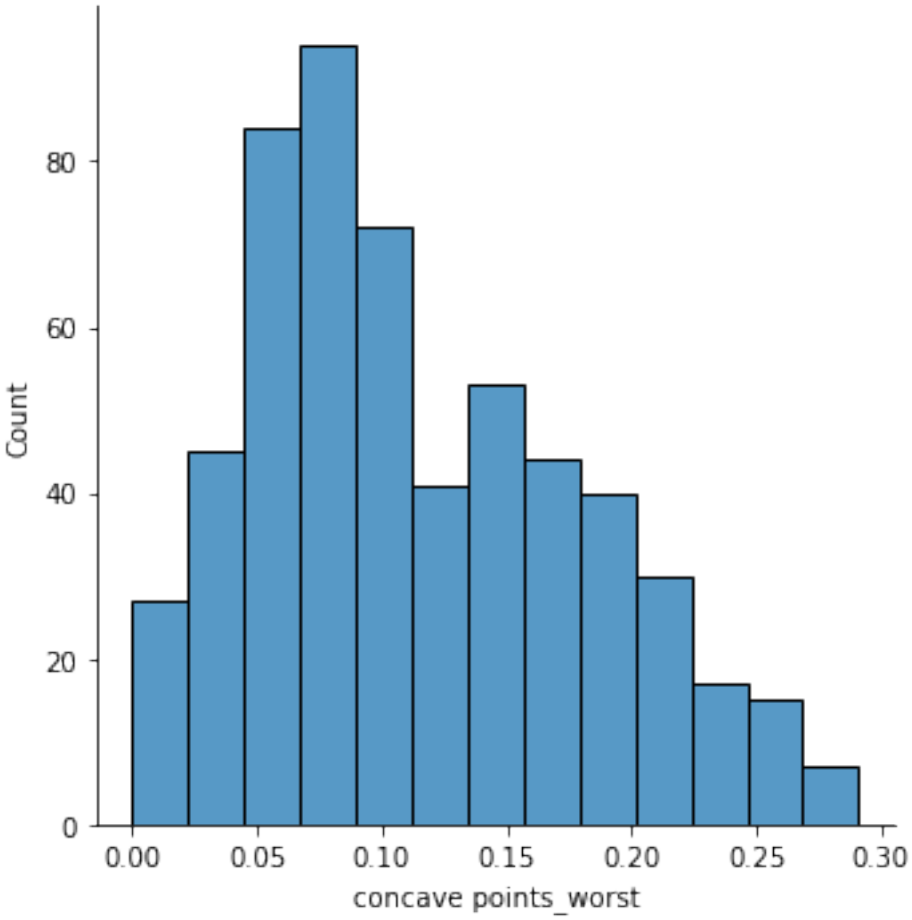

Count
100
80
60
40
20
0
50
75
100
125
150
175
200
225
250
perimeter_worst

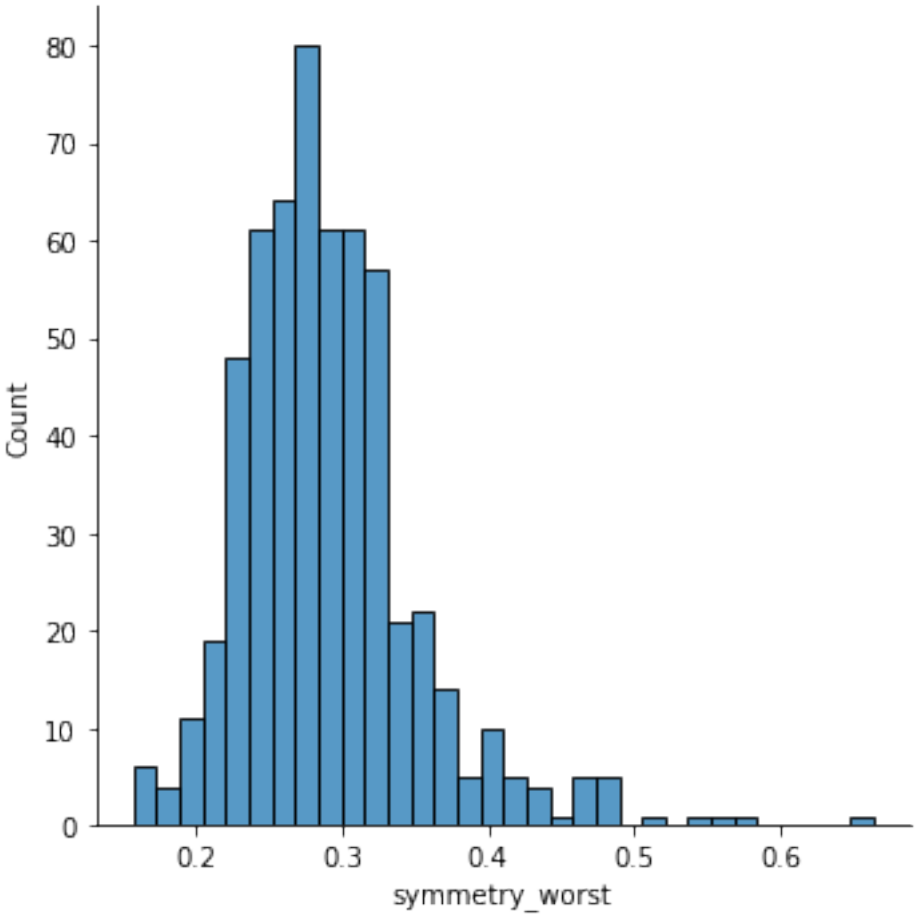

Count
120
100
80
60
40
20
0
0
500
1000
1500
2000
2500
3000
3500
4000
area_worst

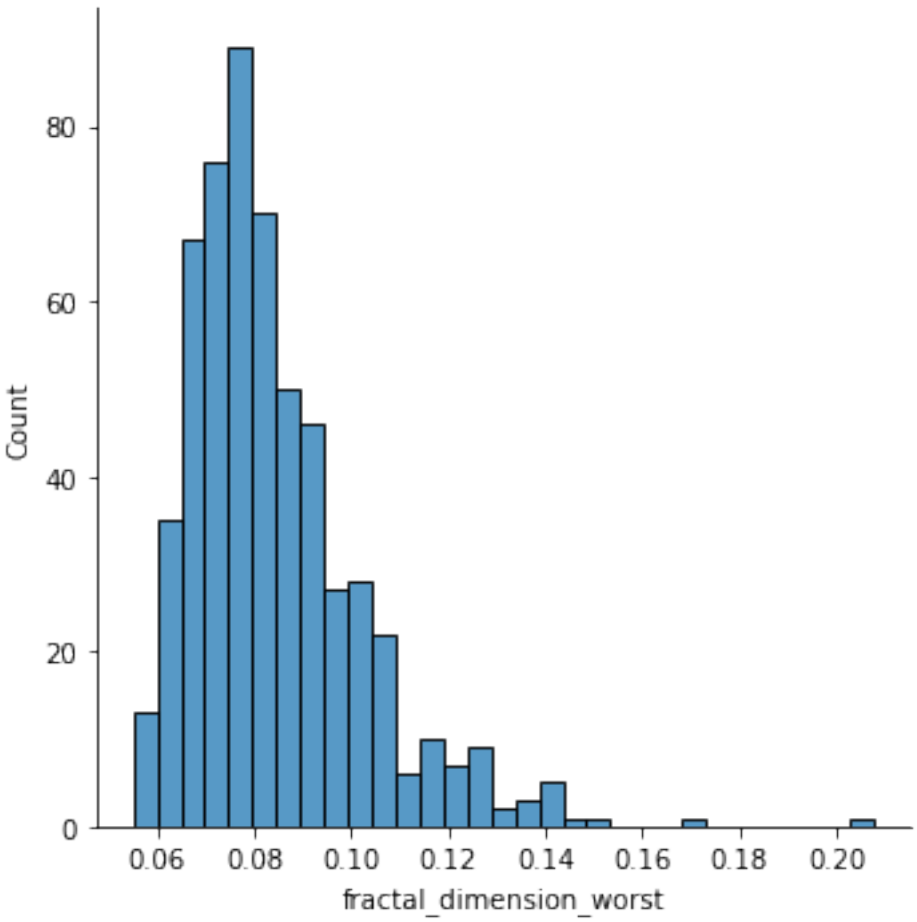

70
60
50
40
30
20
10
0
Count
0.08
0.10
0.12
0.14
0.16
0.18
0.20
0.22
smoothness_worst

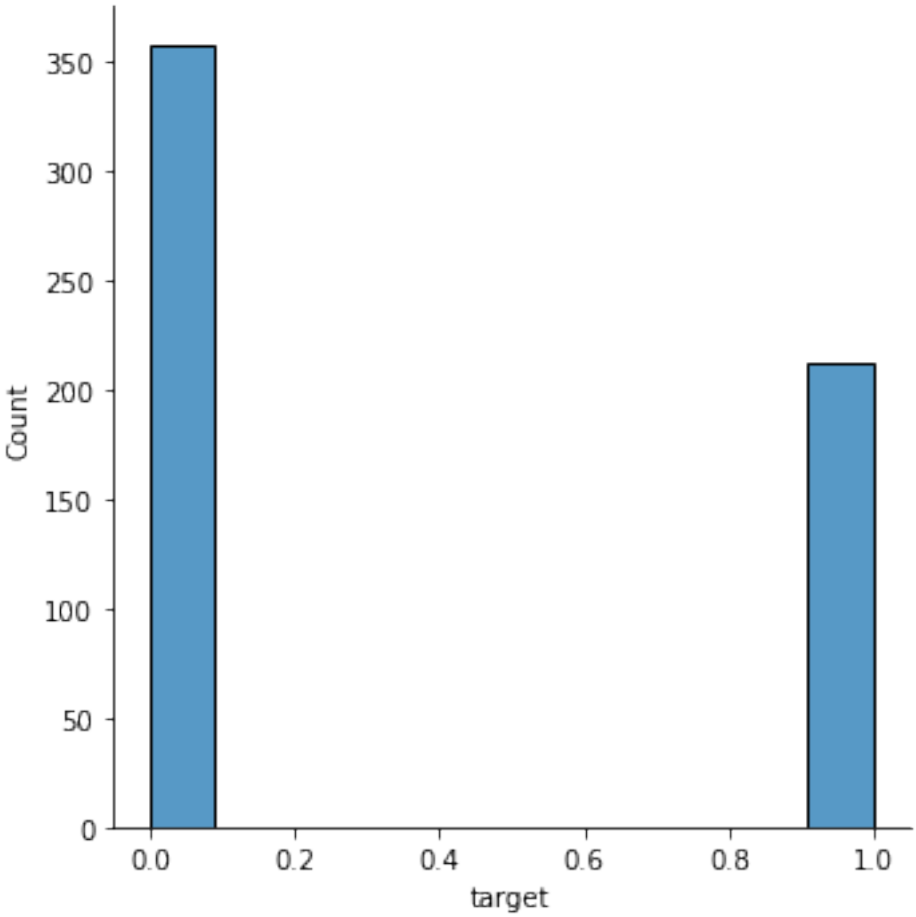

100
80
60
40
20
0
Count
0.0
0.2
0.4
0.6
0.8
1.0
compactness_worst


Count
80
60
40
20
0
0.0
0.2
0.4
0.6
0.8
1.0
1.2
concavity_worst


Count
0
20
40
60
80
0.00
0.05
0.10
0.15
0.20
0.25
0.30
concave points_worst


Count
80
70
60
50
40
30
20
10
0
0.2
0.3
0.4
0.5
0.6
symmetry_worst

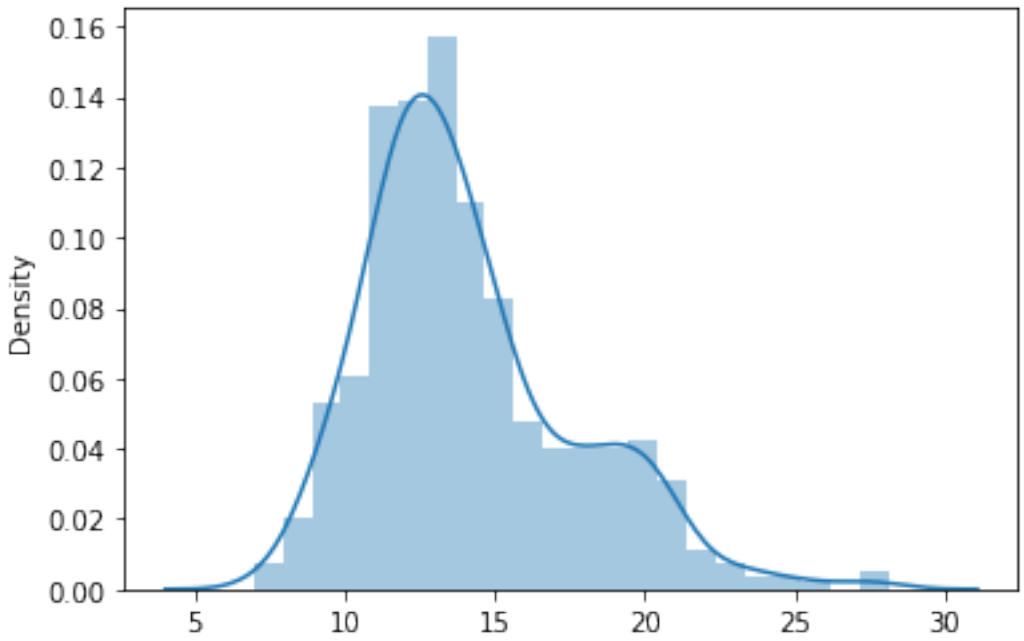

Count
80
60
40
20
0
0.06
0.08
0.10
0.12
0.14
0.16
0.18
0.20
fractal_dimension_worst



[ ]:
```
sns.distplot(x=breast_cancer_data.radius_mean)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619:
FutureWarning: `distplot` is a deprecated function and will be removed in a
future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `histplot` (an axes-level function for
histograms).
  warnings.warn(msg, FutureWarning)
```

[ ]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f79b9e29b50>
```



Inference about distribution: Most of the features are right skewed

Pair plot

Pair plot takes a lot of time if the number of features is more. So we are going to take a random sample of the original dataset to make the pairplot (Not plotting here)

[ ]:
```
# pair plot
#sns.pairplot(df)
#plt.show()
```

Scatter plot of first 2 columns

[ ]:
```
# Select first column of the dataframe as a series
first_column = breast_cancer_data.iloc[:, 0]

# Select second column of the dataframe as a series
second_column = breast_cancer_data.iloc[:, 1]
```

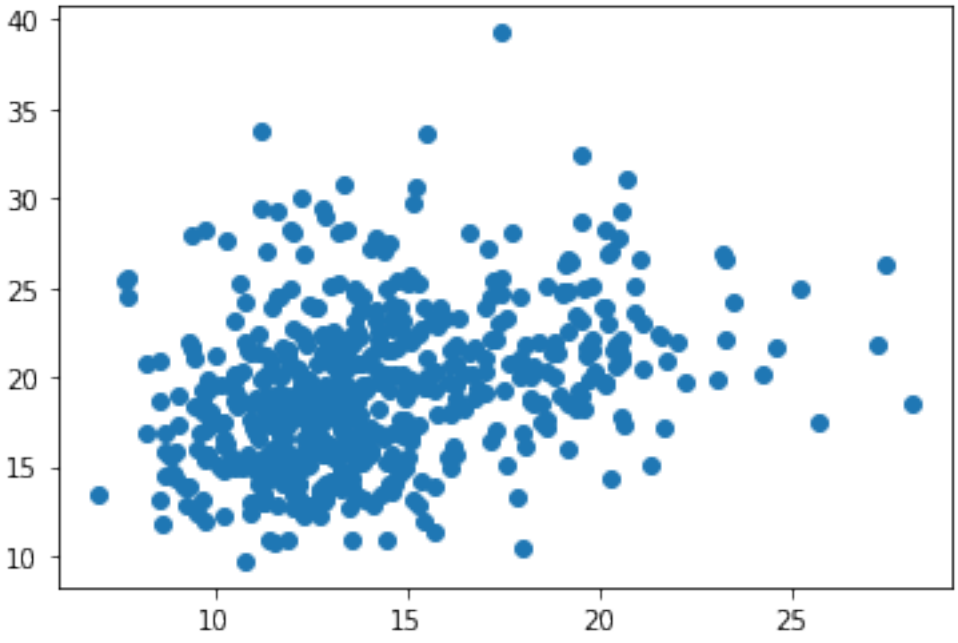
[ ]:
```
print(first_column)
print('-----')
print(second_column)
```

```
0      17.99
1      20.57
2      19.69
3      11.42
4      20.29
```

```
       …
564    21.56
565    20.13
566    16.60
567    20.60
568     7.76
Name: radius_mean, Length: 569, dtype: float64
-----
0      10.38
1      17.77
2      21.25
3      20.38
4      14.34
       …
564    22.39
565    28.25
566    28.08
567    29.33
568    24.54
Name: texture_mean, Length: 569, dtype: float64
```

[ ]:
```
# let's plot a scatter plot for 1st feature vs second feature
plt.scatter(x=first_column, y=second_column)
```

[ ]: <matplotlib.collections.PathCollection at 0x7f79b7953310>



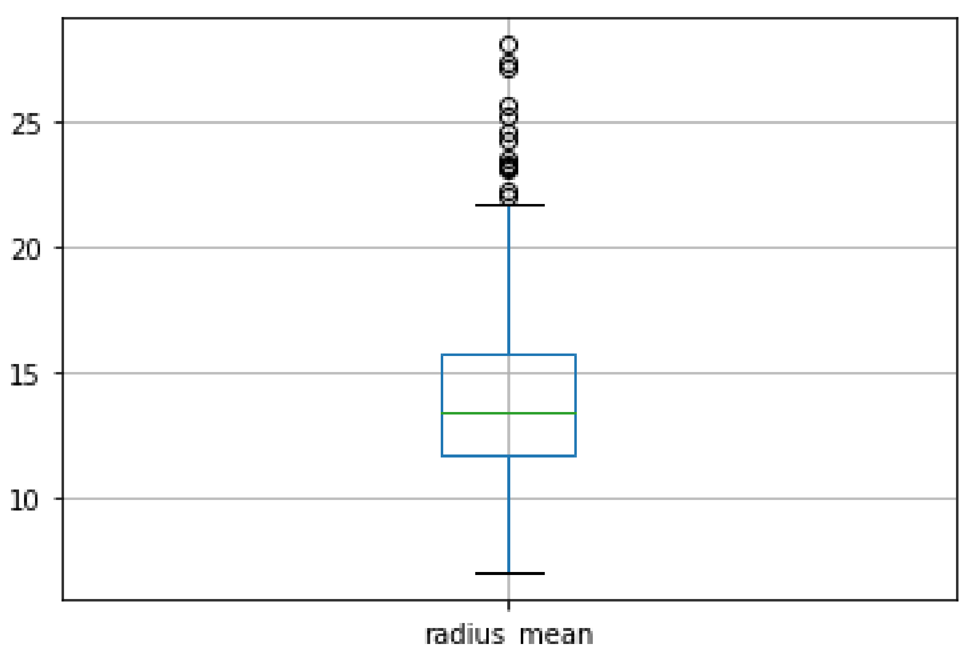
Outliers Detection

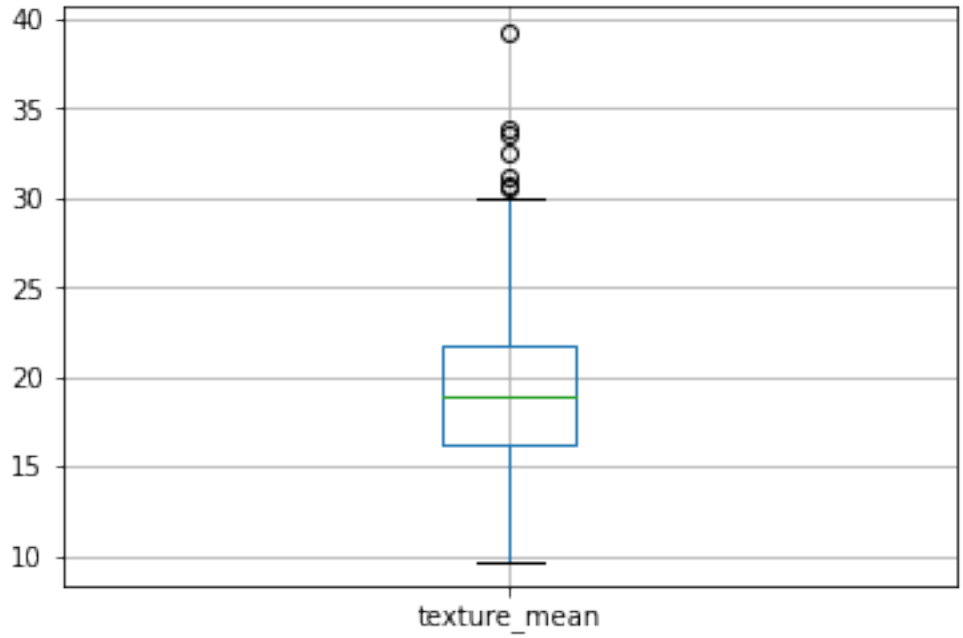
Box plot for visualizing the outliers in the dataset

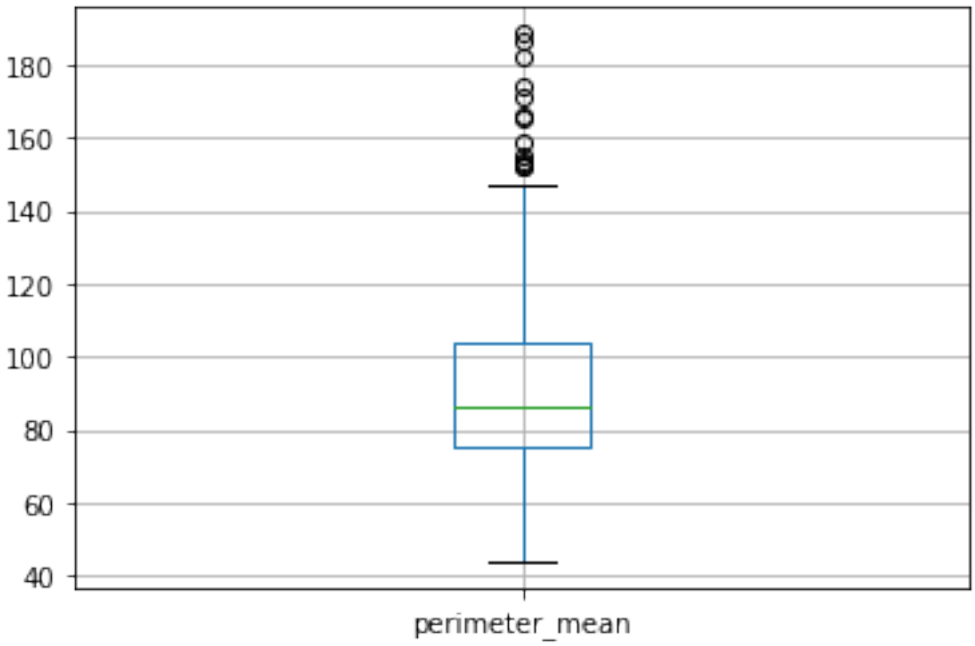
[ ]:
```
for column in breast_cancer_data:
    plt.figure()
    breast_cancer_data.boxplot([column])
```

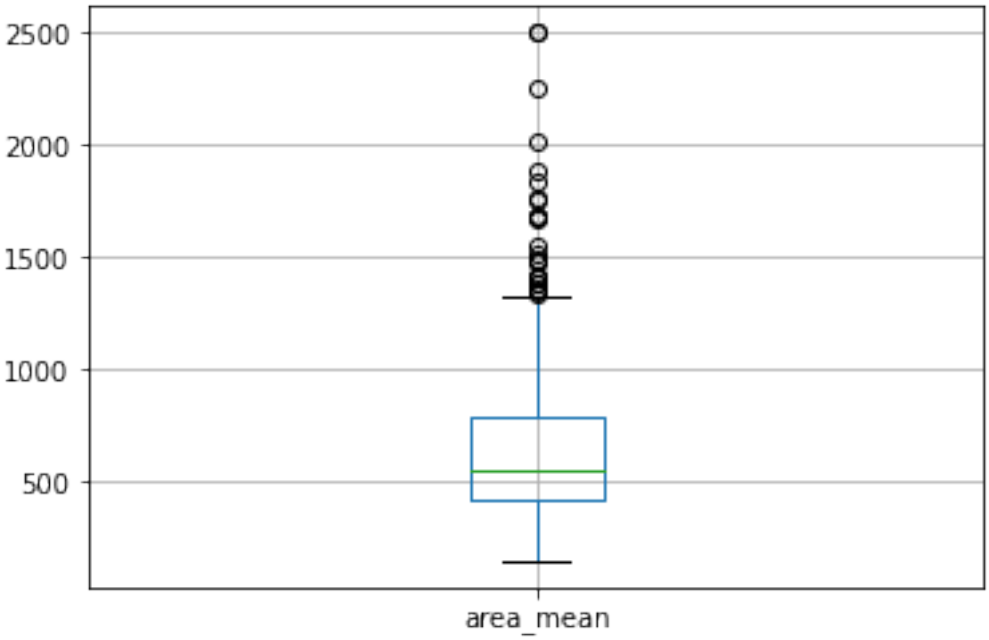
```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: RuntimeWarning:
More than 20 figures have been opened. Figures created through the pyplot
interface (`matplotlib.pyplot.figure`) are retained until explicitly closed and
may consume too much memory. (To control this warning, see the rcParam
`figure.max_open_warning`).

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: RuntimeWarning:
More than 20 figures have been opened. Figures created through the pyplot
interface (`matplotlib.pyplot.figure`) are retained until explicitly closed and
may consume too much memory. (To control this warning, see the rcParam
`figure.max_open_warning`).

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: RuntimeWarning:
More than 20 figures have been opened. Figures created through the pyplot
interface (`matplotlib.pyplot.figure`) are retained until explicitly closed and
may consume too much memory. (To control this warning, see the rcParam
`figure.max_open_warning`).

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: RuntimeWarning:
More than 20 figures have been opened. Figures created through the pyplot
interface (`matplotlib.pyplot.figure`) are retained until explicitly closed and
may consume too much memory. (To control this warning, see the rcParam
`figure.max_open_warning`).

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: RuntimeWarning:
More than 20 figures have been opened. Figures created through the pyplot
interface (`matplotlib.pyplot.figure`) are retained until explicitly closed and
may consume too much memory. (To control this warning, see the rcParam
`figure.max_open_warning`).

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: RuntimeWarning:
More than 20 figures have been opened. Figures created through the pyplot
interface (`matplotlib.pyplot.figure`) are retained until explicitly closed and
may consume too much memory. (To control this warning, see the rcParam
`figure.max_open_warning`).

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: RuntimeWarning:
More than 20 figures have been opened. Figures created through the pyplot
interface (`matplotlib.pyplot.figure`) are retained until explicitly closed and
may consume too much memory. (To control this warning, see the rcParam
```

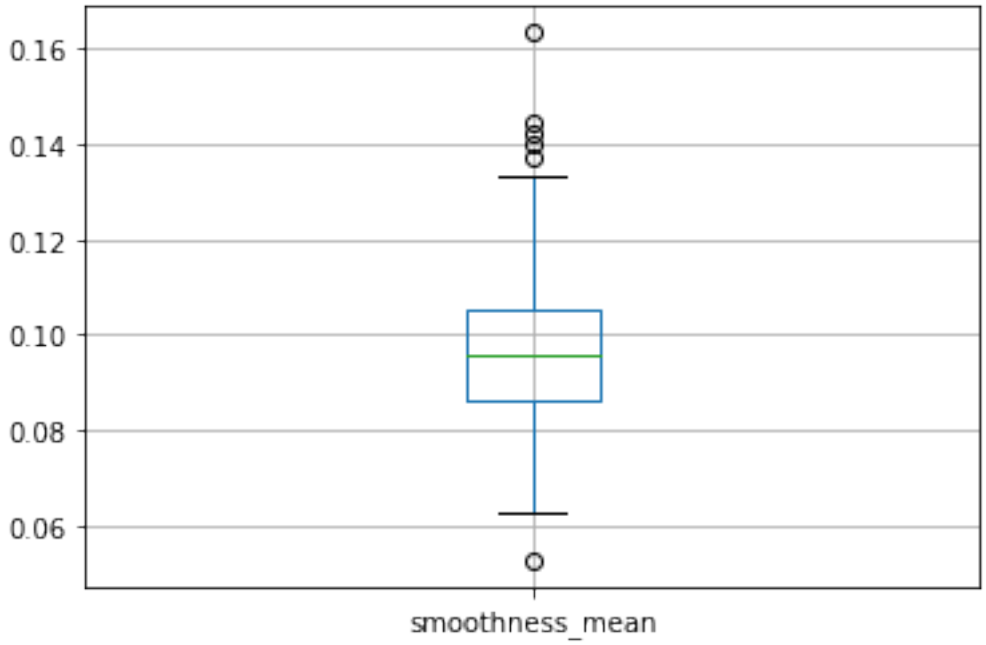
```
`figure.max_open_warning`).

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: RuntimeWarning:
More than 20 figures have been opened. Figures created through the pyplot
interface (`matplotlib.pyplot.figure`) are retained until explicitly closed and
may consume too much memory. (To control this warning, see the rcParam
`figure.max_open_warning`).

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: RuntimeWarning:
More than 20 figures have been opened. Figures created through the pyplot
interface (`matplotlib.pyplot.figure`) are retained until explicitly closed and
may consume too much memory. (To control this warning, see the rcParam
`figure.max_open_warning`).

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: RuntimeWarning:
More than 20 figures have been opened. Figures created through the pyplot
interface (`matplotlib.pyplot.figure`) are retained until explicitly closed and
may consume too much memory. (To control this warning, see the rcParam
`figure.max_open_warning`).

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: RuntimeWarning:
More than 20 figures have been opened. Figures created through the pyplot
interface (`matplotlib.pyplot.figure`) are retained until explicitly closed and
may consume too much memory. (To control this warning, see the rcParam
`figure.max_open_warning`).
```

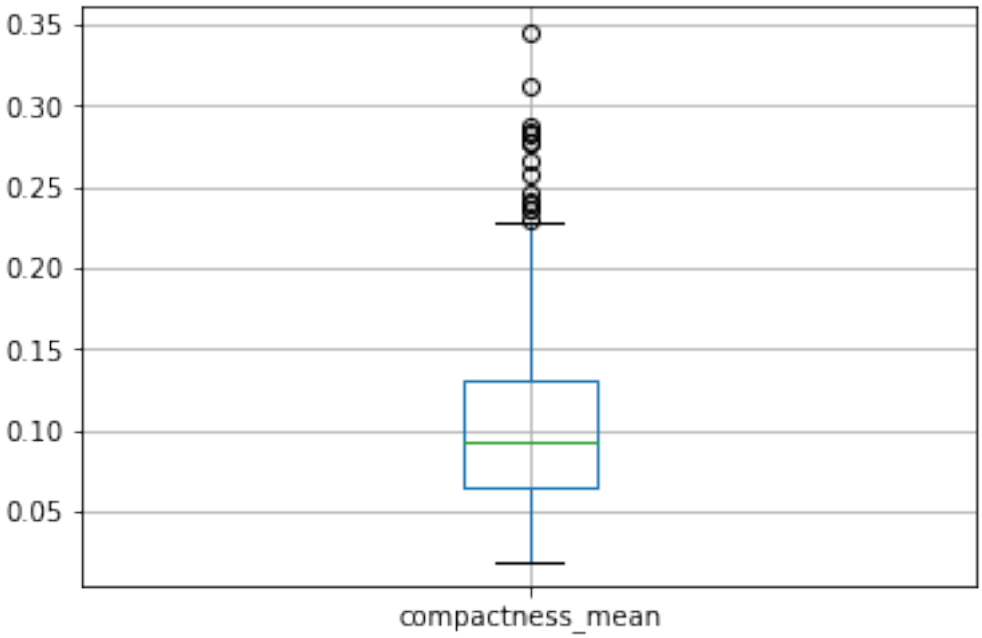


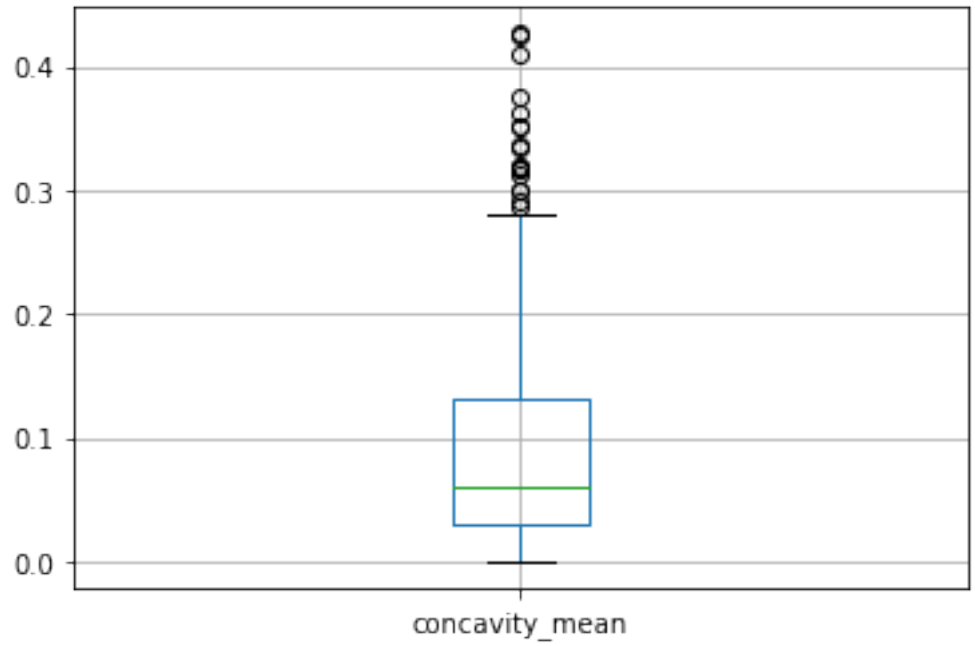

40
35
30
25
20
15
10
texture_mean

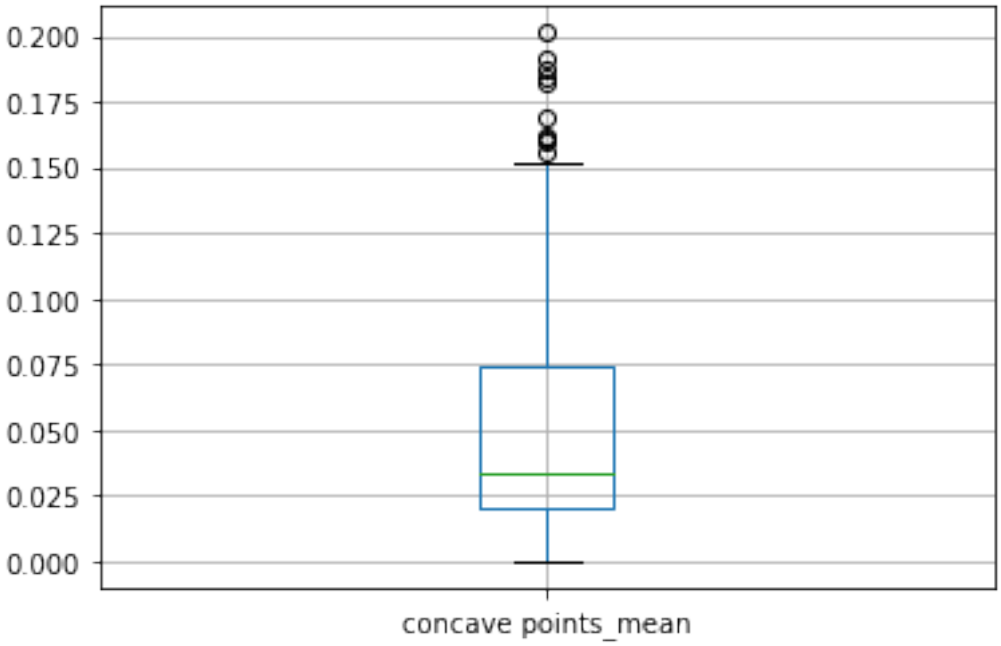

180
160
140
120
100
80
60
40
perimeter_mean

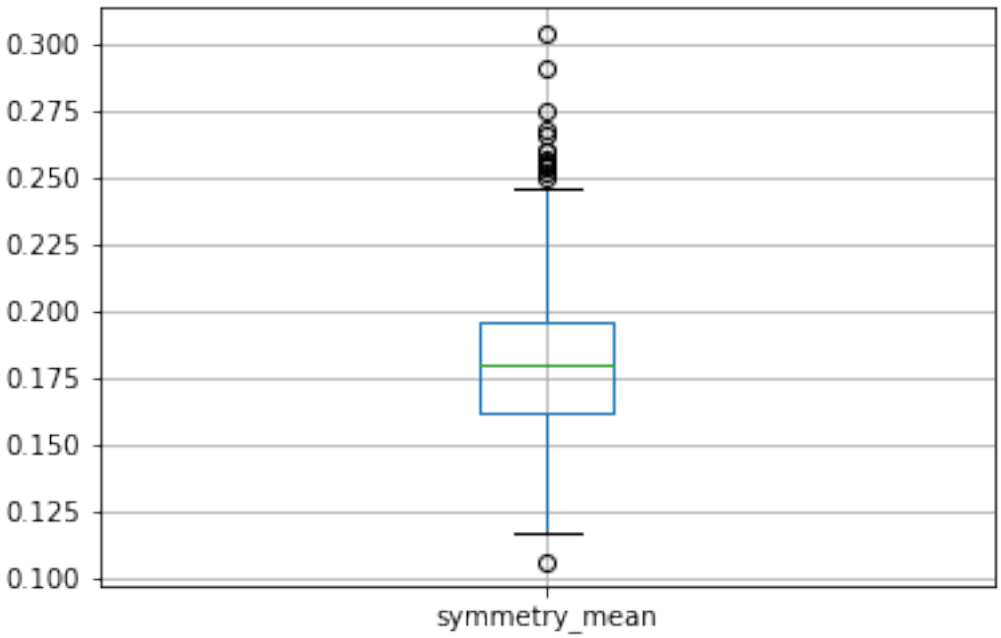

2500
2000
1500
1000
500
area_mean

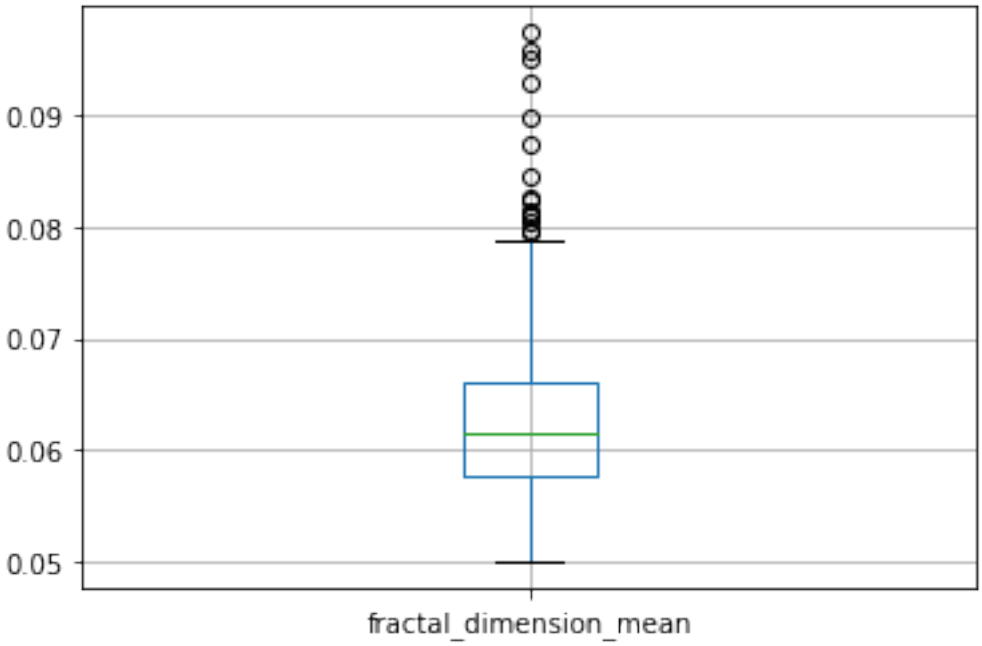

0.16
0.14
0.12
0.10
0.08
0.06
smoothness_mean

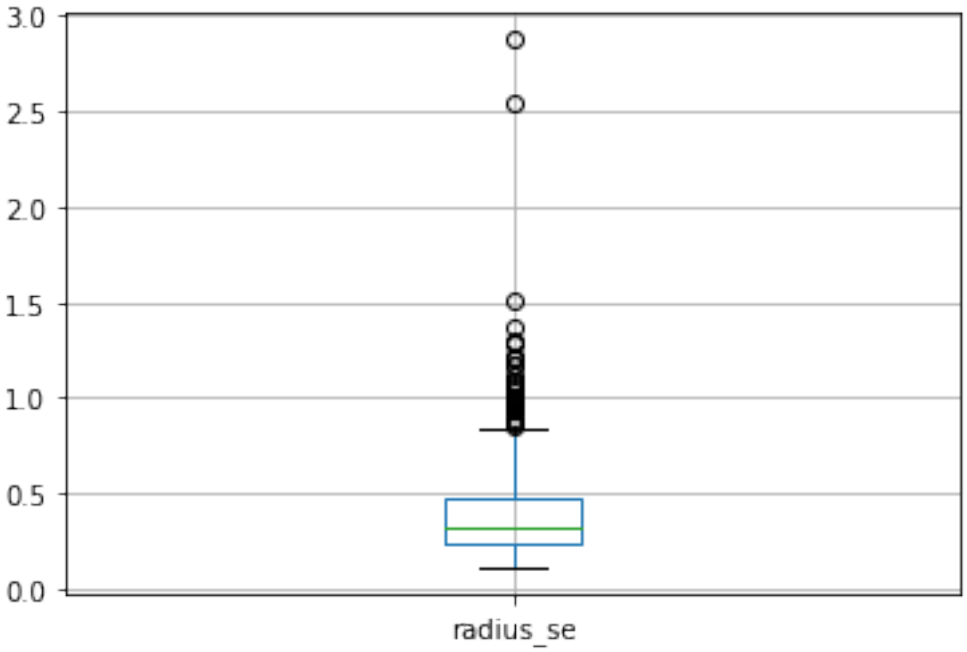

0.35
0.30
0.25
0.20
0.15
0.10
0.05
compactness_mean

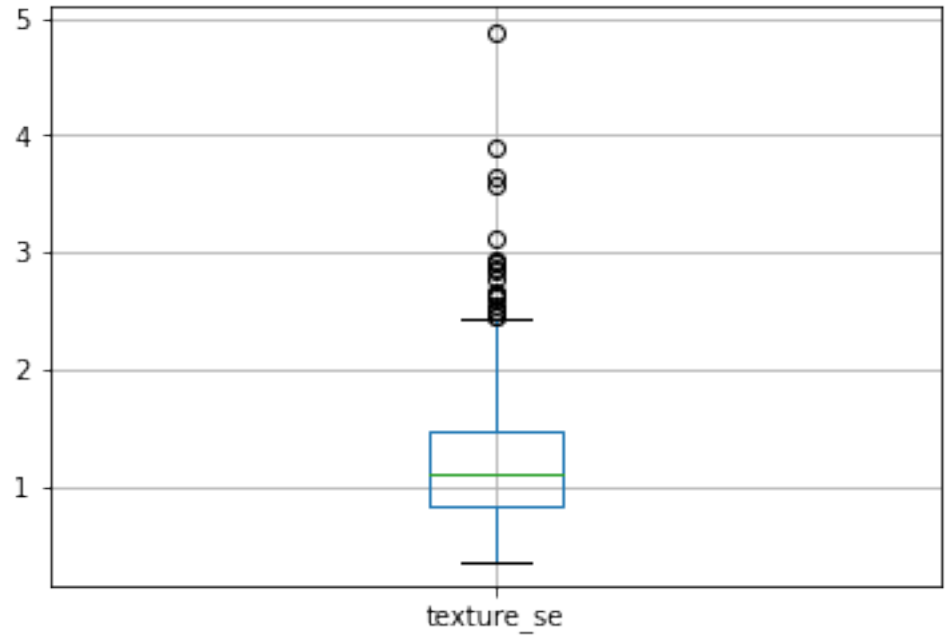

0.4
0.3
0.2
0.1
0.0
concavity_mean

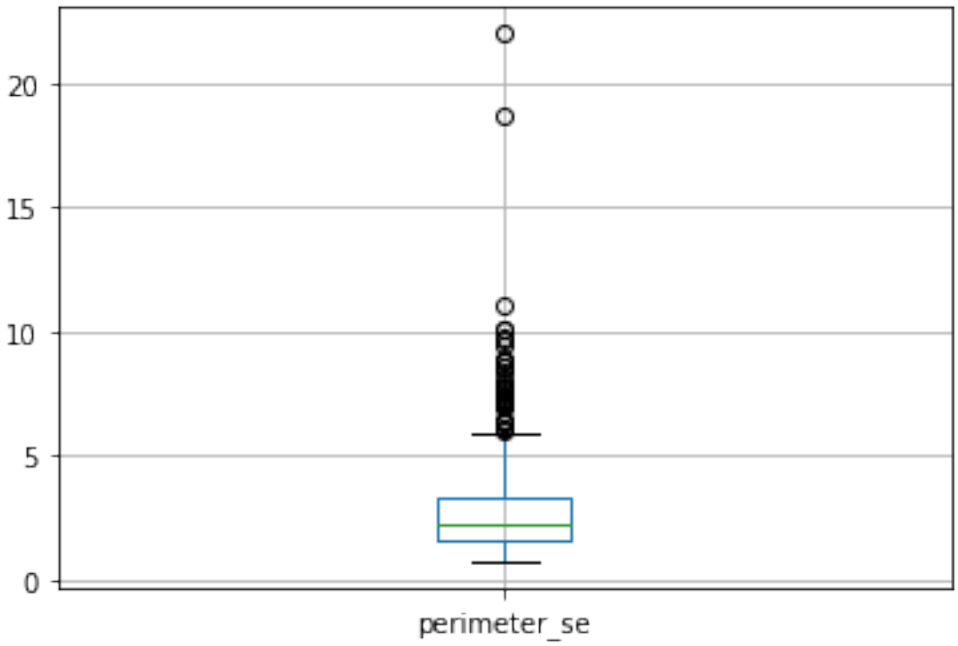

0.200
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0.000
concave points_mean

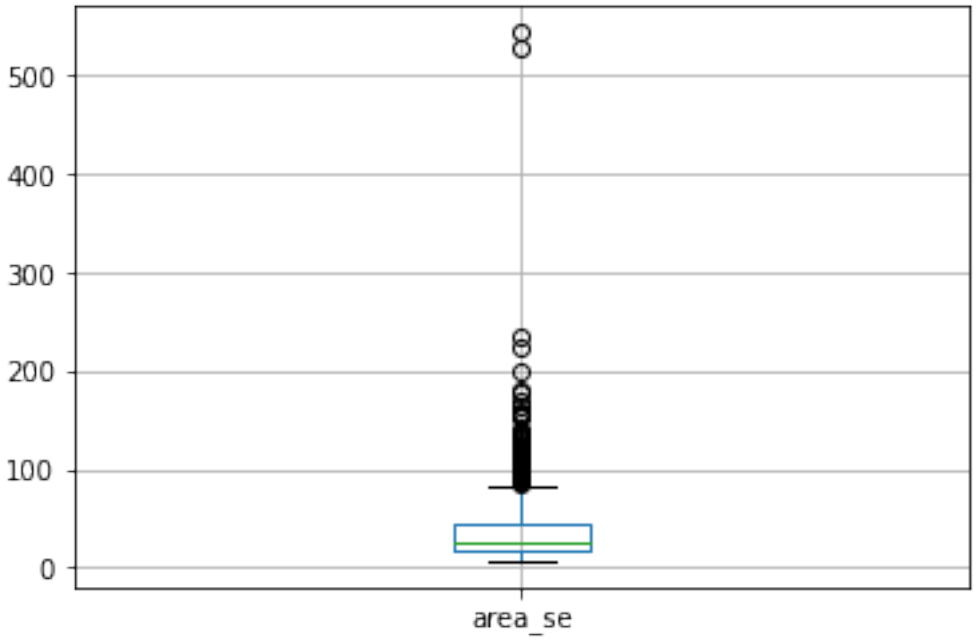

0.300
0.275
0.250
0.225
0.200
0.175
0.150
0.125
0.100
symmetry_mean


0.09
0.08
0.07
0.06
0.05
fractal_dimension_mean

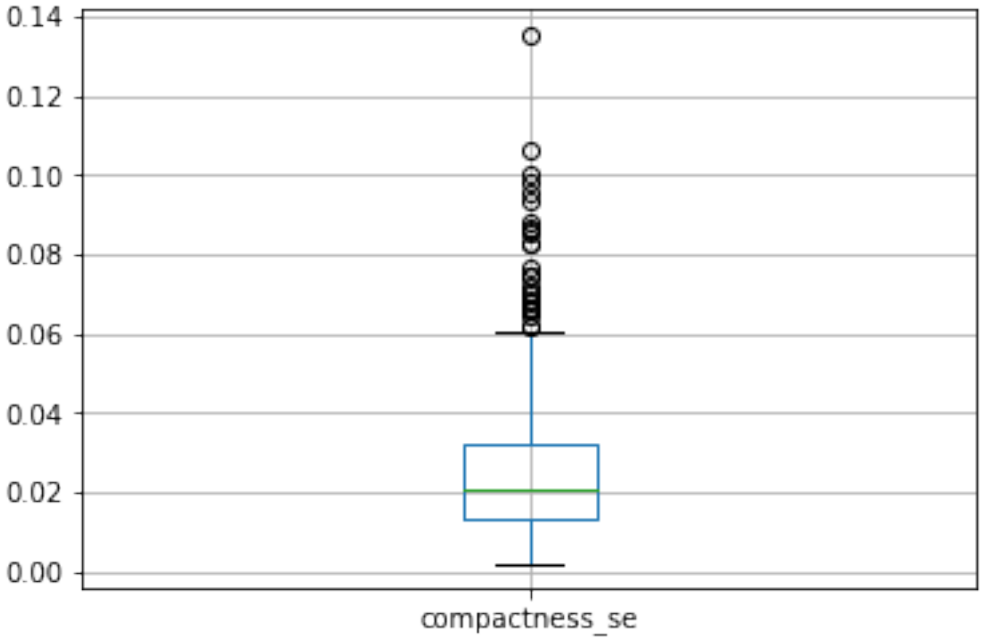

3.0
2.5
2.0
1.5
1.0
0.5
0.0
radius_se

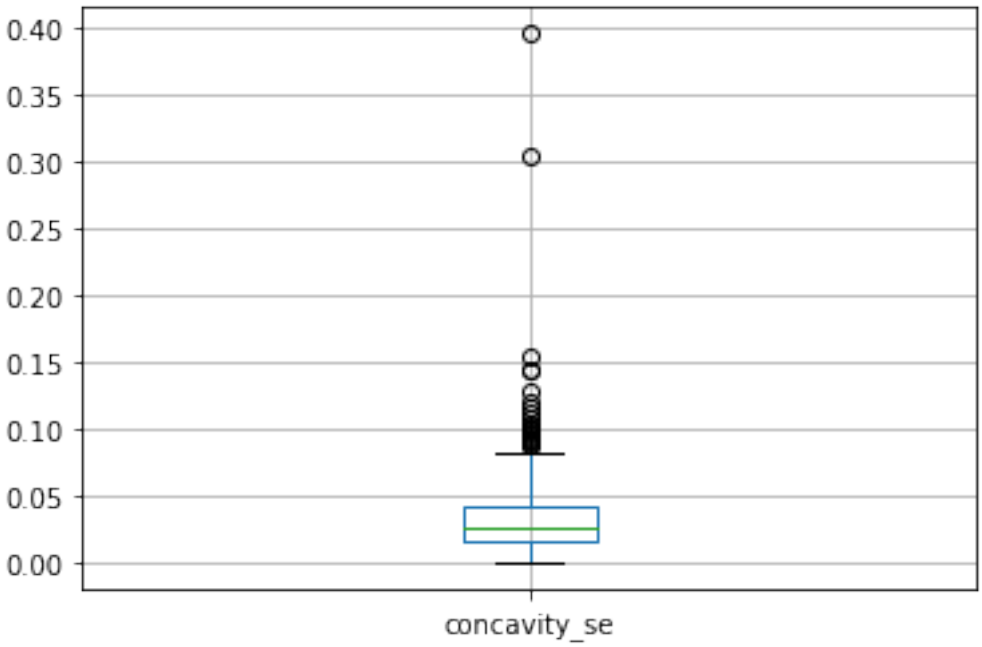

5
4
3
2
1
texture_se

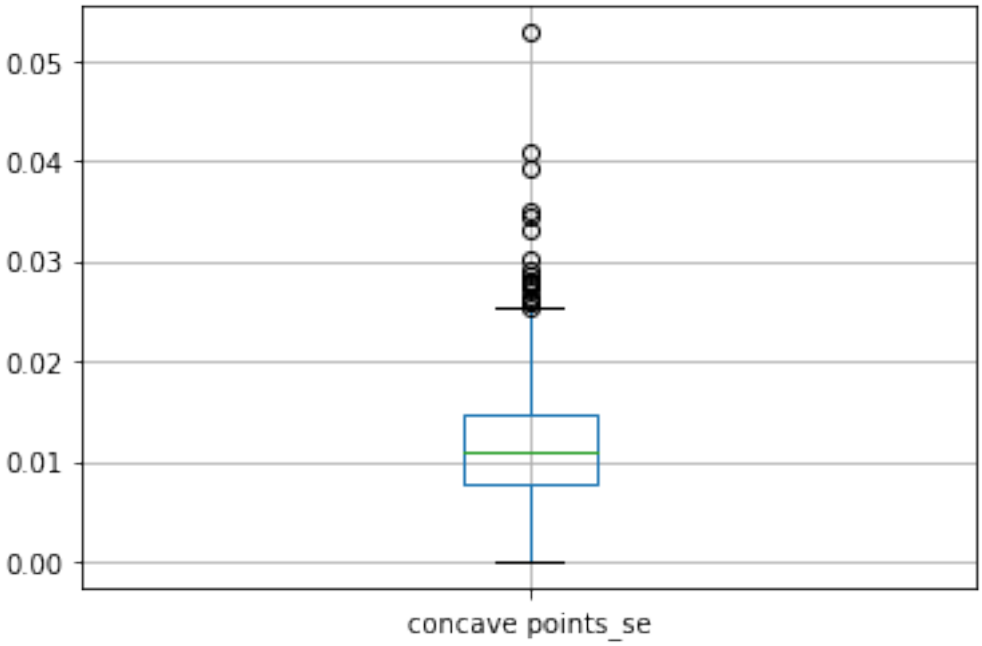

20
15
10
5
0
perimeter_se

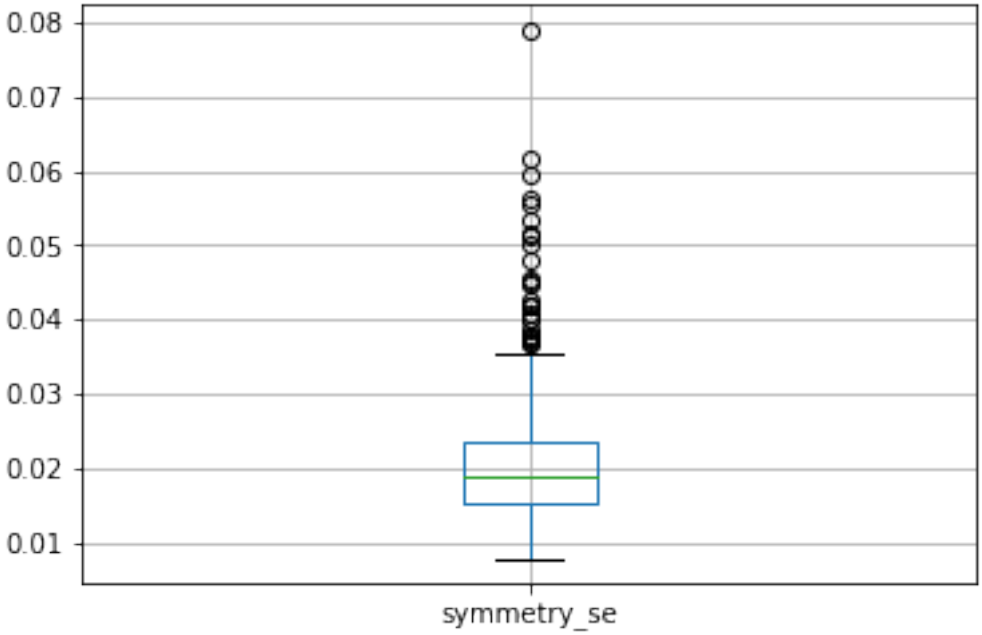

500
400
300
200
100
0
area_se

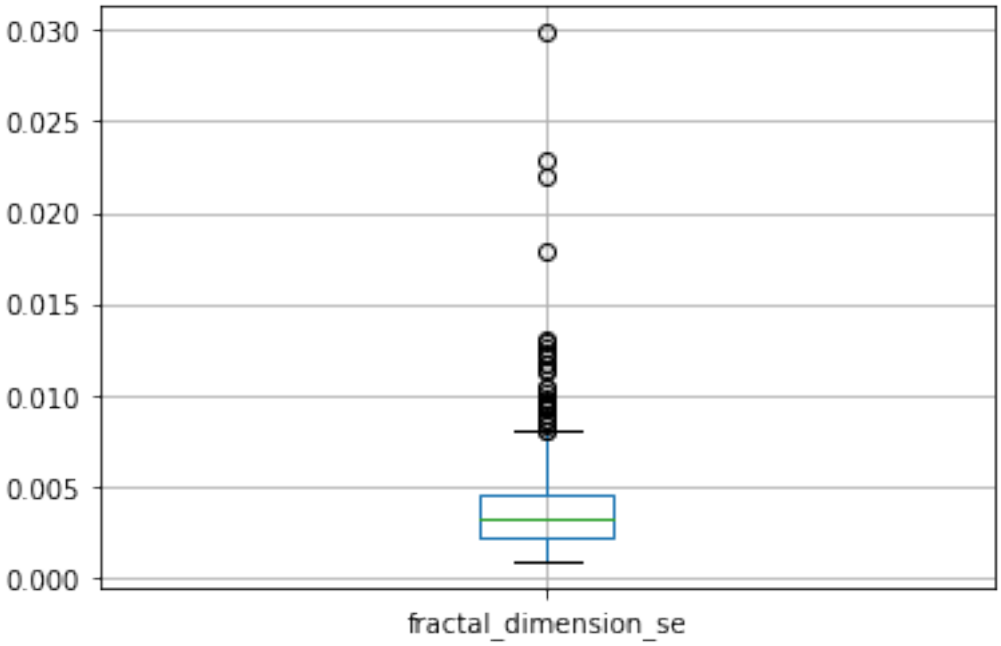

0.030
0.025
0.020
0.015
0.010
0.005
smoothness_se

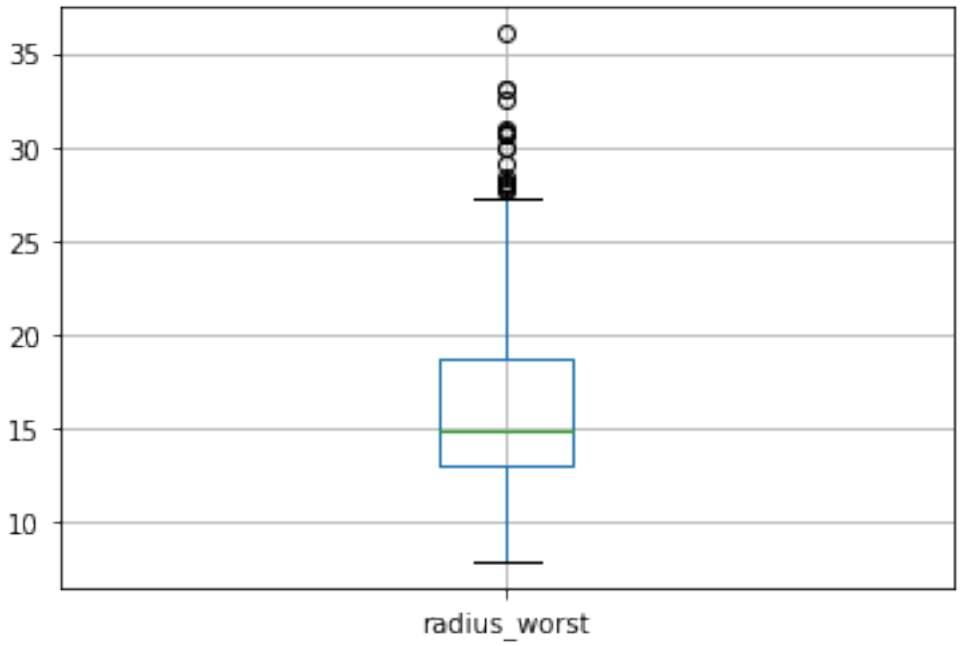

0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
compactness_se

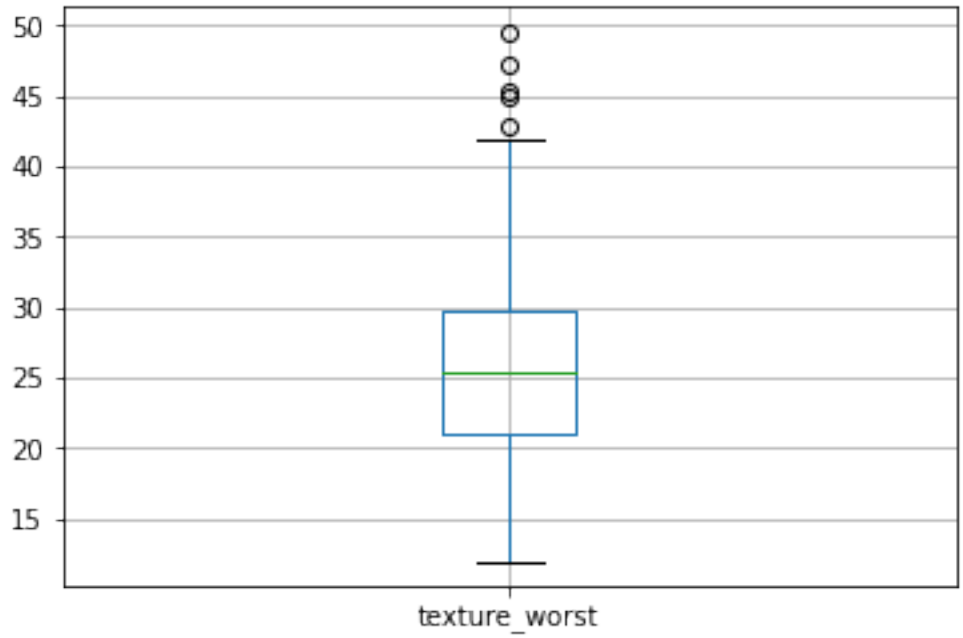

0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
concavity_se

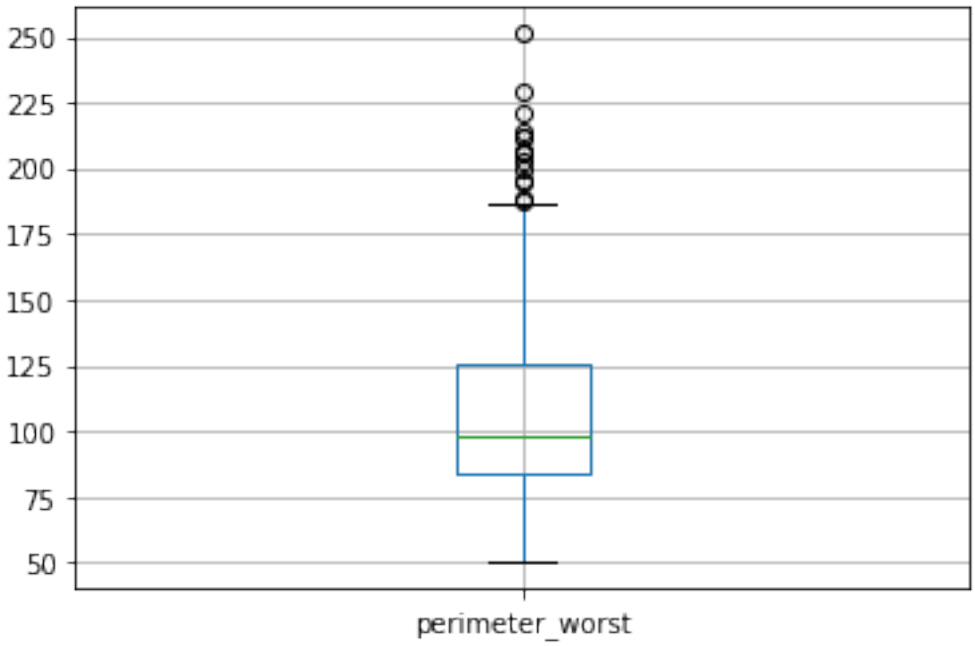

0.05
0.04
0.03
0.02
0.01
0.00
concave points_se

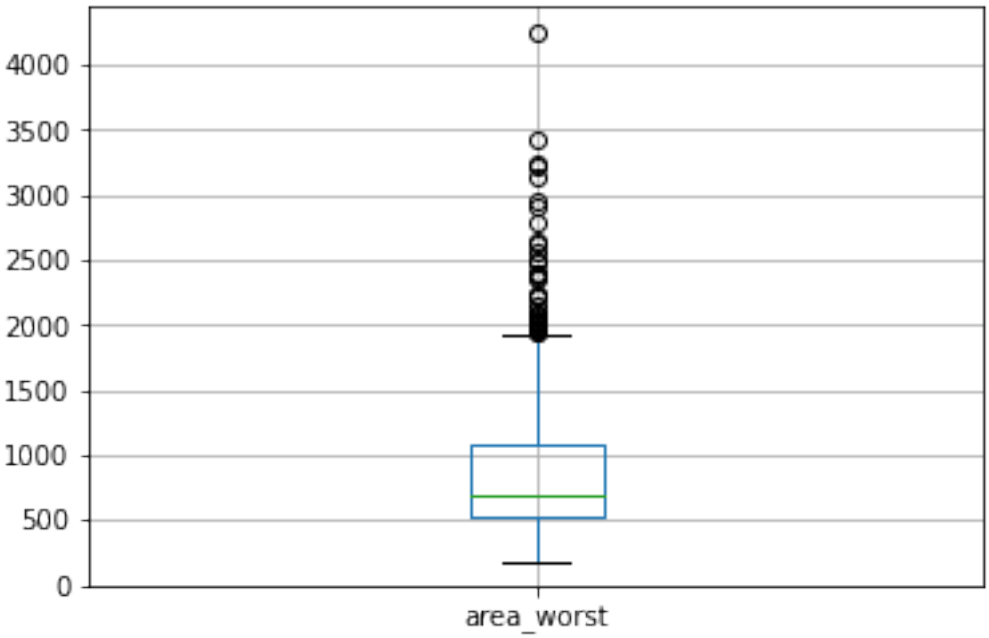

0.08
0.07
0.06
0.05
0.04
0.03
0.02
0.01
symmetry_se

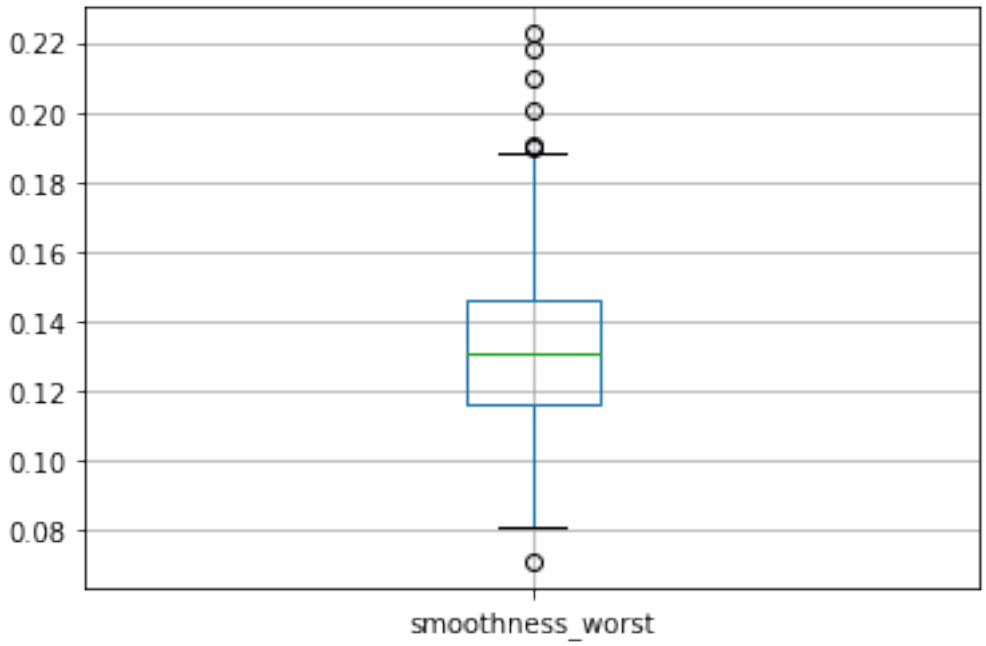

0.030
0.025
0.020
0.015
0.010
0.005
0.000
fractal_dimension_se

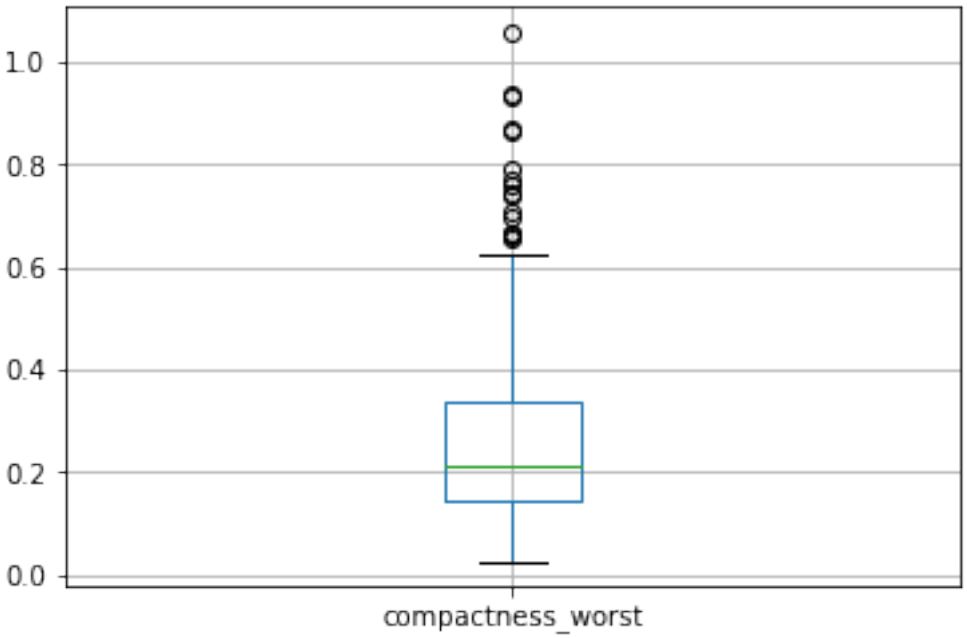

35
30
25
20
15
10
radius_worst

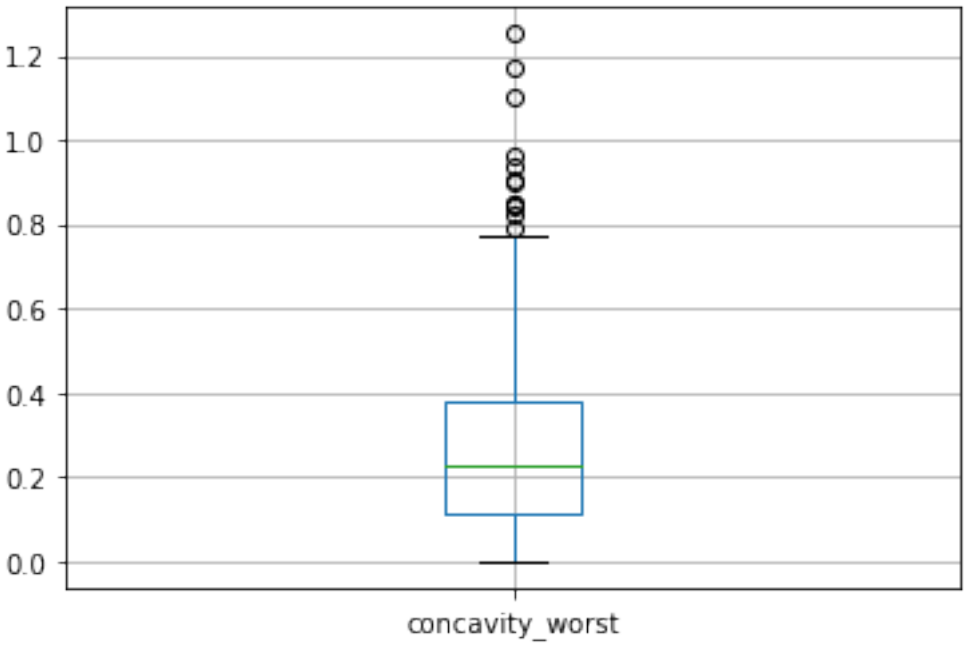

50
45
40
35
30
25
20
15
texture_worst

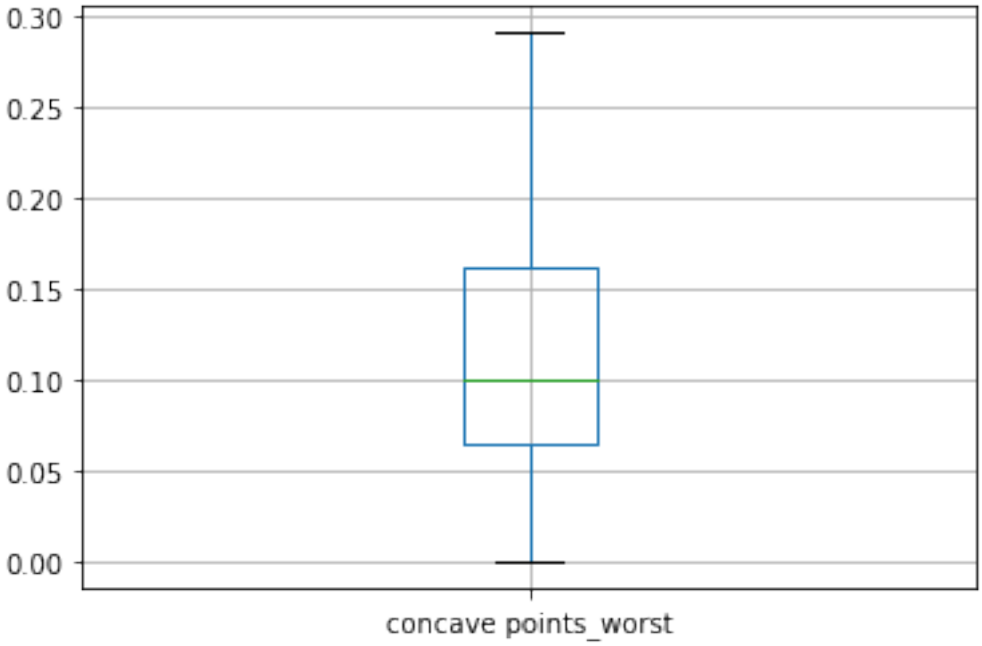

250
225
200
175
150
125
100
75
50
perimeter_worst

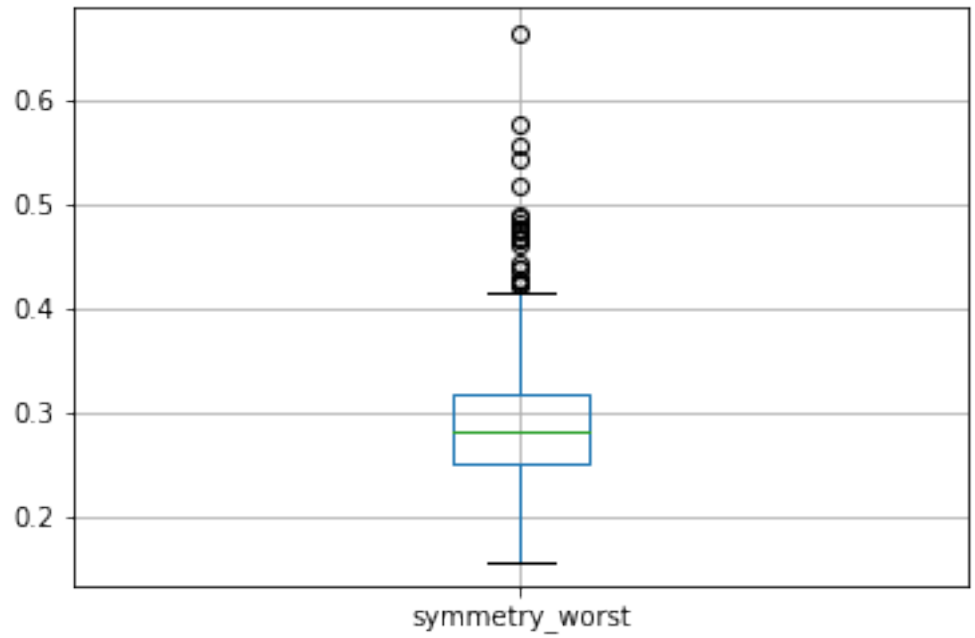

4000
3500
3000
2500
2000
1500
1000
500
0
area_worst

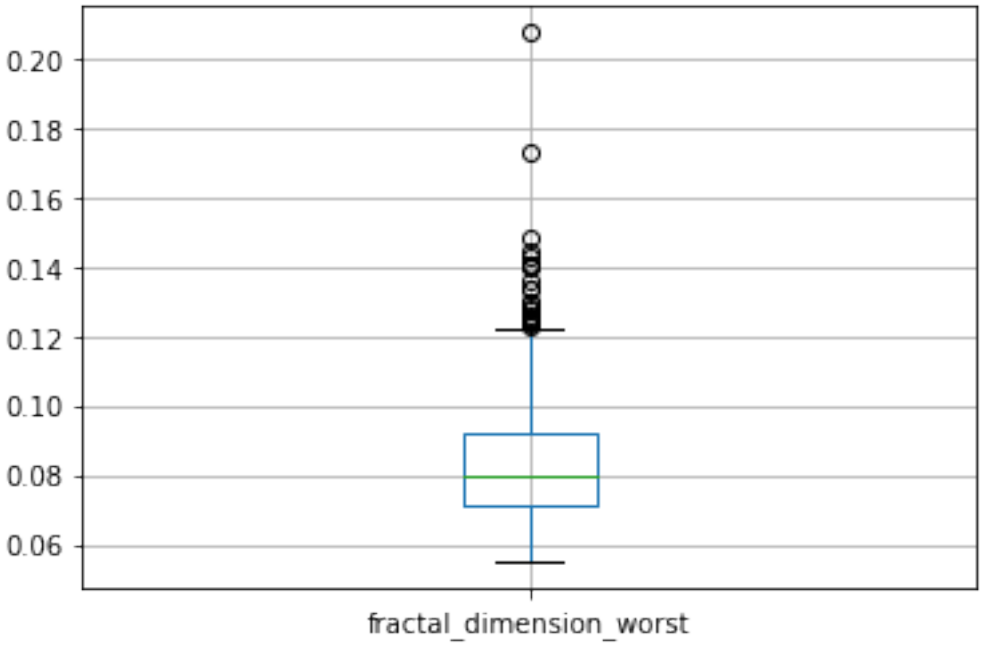

0.22
0.20
0.18
0.16
0.14
0.12
0.10
0.08
smoothness_worst

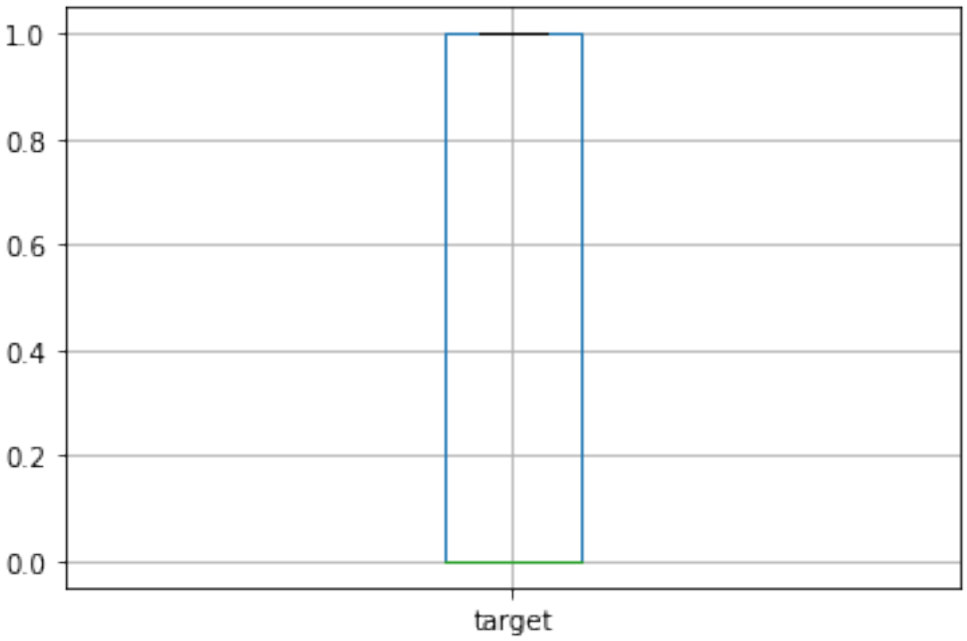

1.0
0.8
0.6
0.4
0.2
0.0
compactness_worst


1.2
1.0
0.8
0.6
0.4
0.2
0.0
concavity_worst


0.30
0.25
0.20
0.15
0.10
0.05
0.00
concave points_worst


0.6
0.5
0.4
0.3
0.2
symmetry_worst



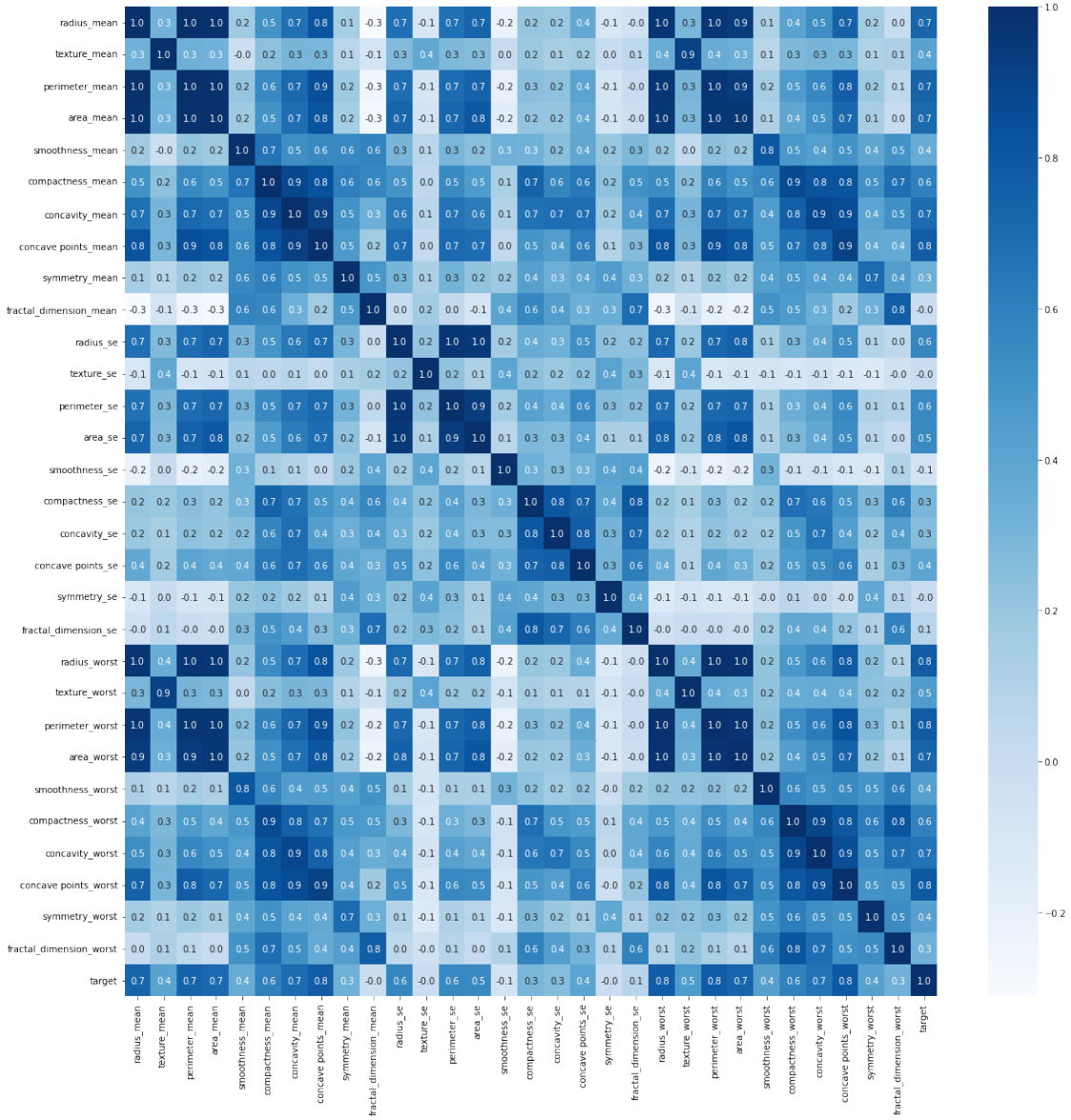


As we can see here that most of the skewed features have Outliers

Correlation Matrix

[ ]:
```
correlation_matrix = breast_cancer_data.corr()
```

```
[ ]: # constructing a heat map to visualize the correlation matrix
plt.figure(figsize=(20,20))
sns.heatmap(correlation_matrix, cbar=True, fmt='.1f', annot=True, cmap='Blues')
plt.savefig('Correlation Heat map')
```



Multicollinearity problem:

Multicollinearity exists when an independent variable is highly correlated with one or more independent variables

We can remove the features if they have high +ve or -ve correlation between them

Inference from EDA & Data Visualization: 1. Mean is slightly more than the median for

most of the features. So it is right skewed. 2. Slight imbalance in the dataset (Benign(0) cases are more than Malignant(1) cases 3. Mean of most features are clearly larger for Malignant cases compared to the benign cases (Groupby) 4. Most of the features have Outliers 5. Correlation Matrix reveal that most of the features are highly correlated. So we can remove certain features during Feature Selection

[ ]: