1 Import Libraries

1.1 Data Processing Libraries

In [1]:

```
# data processing
import numpy as np
import pandas as pd
from scipy.stats.mstats import winsorize
from scipy import stats
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
```

1.2 Visualization

In [2]:

```
#import visualizing libraries
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

1.3 Utilization

In [3]:

```
import random
import pickle

random_state = 42
random.seed(random_state)
```

2 Data Loading

In [4]:

```
wdbc = pd.read_csv('../dataset/data.csv')
```

2.1 Lihat 5 data teratas

In [5]:

```
wdbc.head()
```

Out[5]:

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

5 rows × 32 columns

2.2 Lihat dimensi data

In [6]:

```
wdbc.shape
```

Out[6]:

```
(569, 32)
```

2.3 Melihat tipe data setiap atribut

In [7]:

```
wdbc.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 32 columns):
 #   Column                   Non-Null Count  Dtype  
---  ------                   --------------  -----  
 0   IDNumber                 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_largest           569 non-null    float64
 23  texture_largest          569 non-null    float64
 24  perimeter_largest        569 non-null    float64
 25  area_largest             569 non-null    float64
 26  smoothness_largest       569 non-null    float64
 27  compactness_largest      569 non-null    float64
 28  concavity_largest        569 non-null    float64
 29  concave_points_largest   569 non-null    float64
 30  symmetry_largest         569 non-null    float64
 31  fractal_dimension_largest  569 non-null    float64
dtypes: float64(30), int64(1), object(1)
memory usage: 142.4+ KB
```

2.4 Mengecek adanya data duplikat

In [8]:

```
wdbc[
    wdbc.duplicated()
]
```

Out[8]:

IDNumber	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_n

0 rows × 32 columns

2.5 Menghapus ID pasien pada data

In [9]:

```
wdbc.drop(['IDNumber'], axis=1, inplace=True)
```

2.6 Memisahkan atribut independen (X) dan dependen (Y)

In [10]:

```
target_name = 'diagnosis'
X = wdbc.drop(target_name,axis=1).copy()
y = wdbc[target_name].copy()
features_name = X.columns.tolist()
```

3 Exploratory Data Analysis

Exploratory Data Analysis memungkinkan analyst memahami isi data yang digunakan, mulai dari distribusi, frekuensi, korelasi dan lainnya.

1. Mengetahui jumlah nan pada data
2. Menampilkan persentase data di setiap kelas
3. Visualisasi histogram setiap atribut dan boxplot
4. Korelasi setiap atribut
5. Menghitung jumlah data outliers di setiap atribut

3.1 Mengetahui jumlah NaN pada data

In [11]:

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

Out[11]:

```
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_largest             0
texture_largest            0
perimeter_largest          0
area_largest               0
smoothness_largest         0
compactness_largest        0
concavity_largest          0
concave_points_largest     0
symmetry_largest           0
fractal_dimension_largest  0
dtype: int64
```

3.2 Deskriptif Statistik Data

In [12]:

```
wdbc.describe()
```

Out[12]:

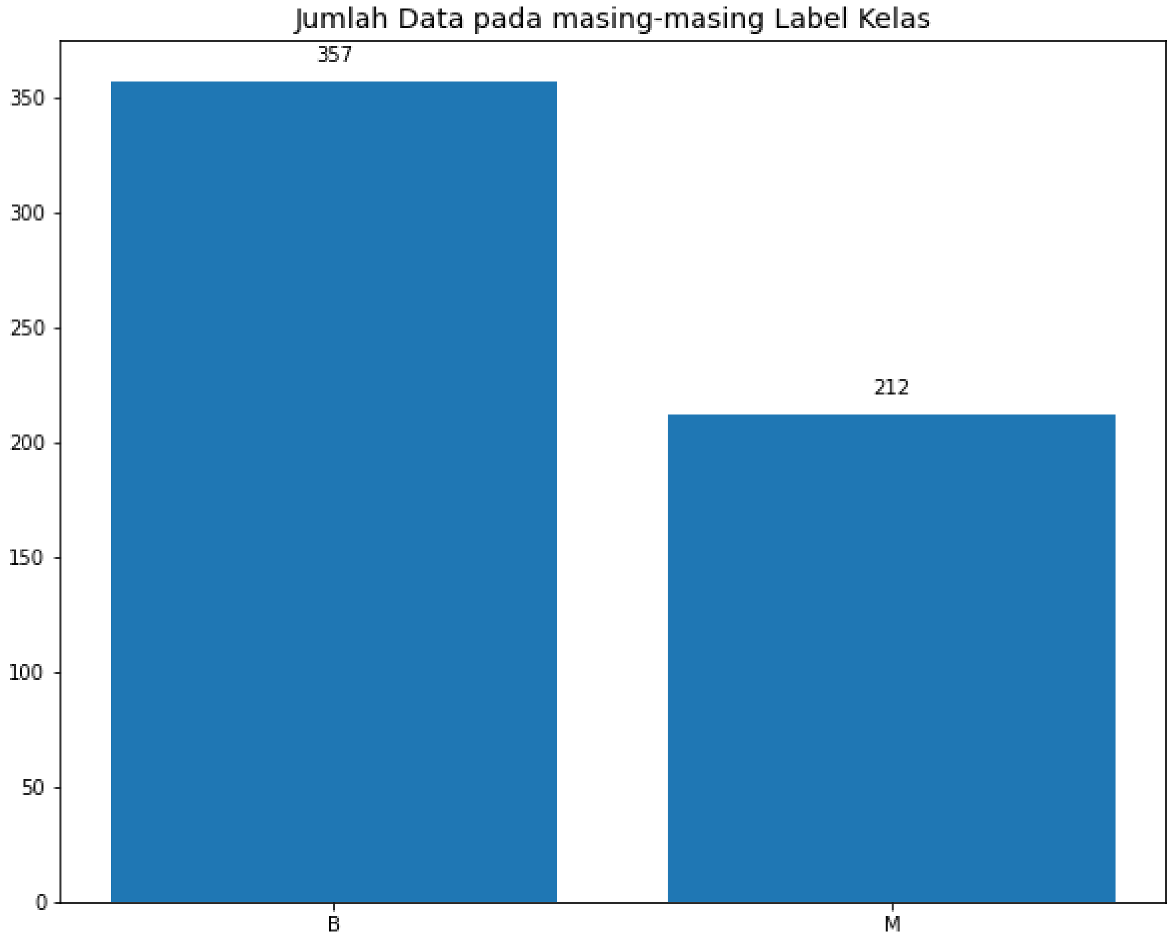
	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactne
count	569.000000	569.000000	569.000000	569.000000	569.000000	5
mean	14.127292	19.289649	91.969033	654.889104	0.096360	
std	3.524049	4.301036	24.298981	351.914129	0.014064	
min	6.981000	9.710000	43.790000	143.500000	0.052630	
25%	11.700000	16.170000	75.170000	420.300000	0.086370	
50%	13.370000	18.840000	86.240000	551.100000	0.095870	
75%	15.780000	21.800000	104.100000	782.700000	0.105300	
max	28.110000	39.280000	188.500000	2501.000000	0.163400	

8 rows × 30 columns

3.3 Persentase data di setiap kelas

In [13]:

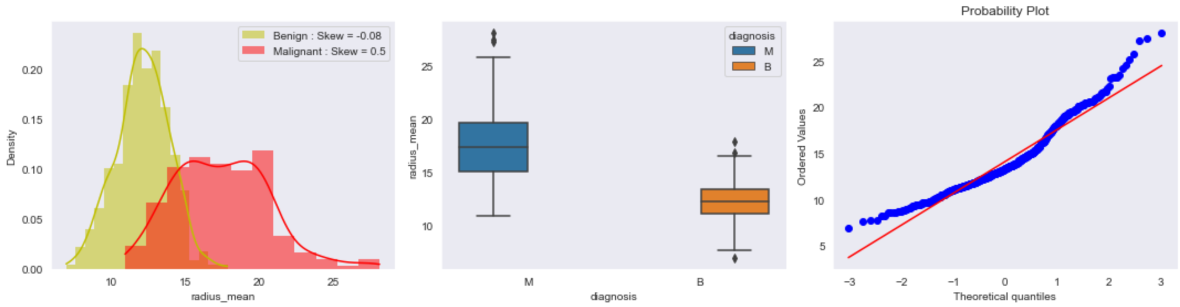
```
xs = wdbc[target_name].value_counts().index
ys = wdbc[target_name].value_counts().values


plt.figure(figsize=(10,8))
plt.bar(xs,ys)
for x,y in zip(xs,ys):
    plt.annotate(
        y,
        (x,y),
        textcoords='offset points',
        xytext=(0,10),
        ha='center'
    )

plt.title('Jumlah Data pada masing-masing Label Kelas', size=14)
plt.show()
```

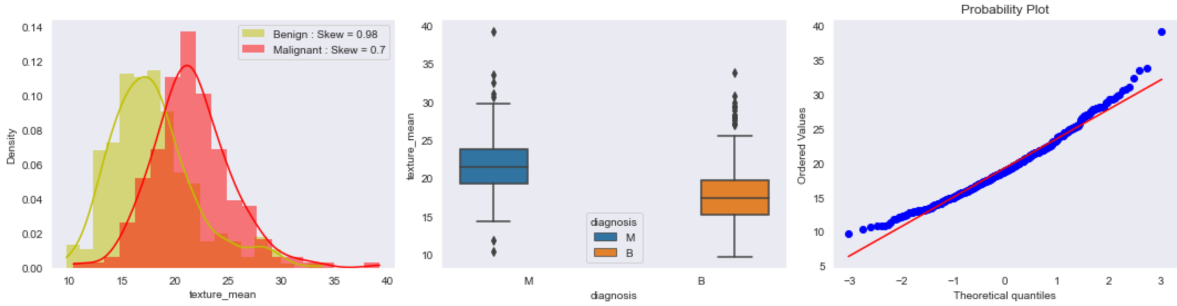


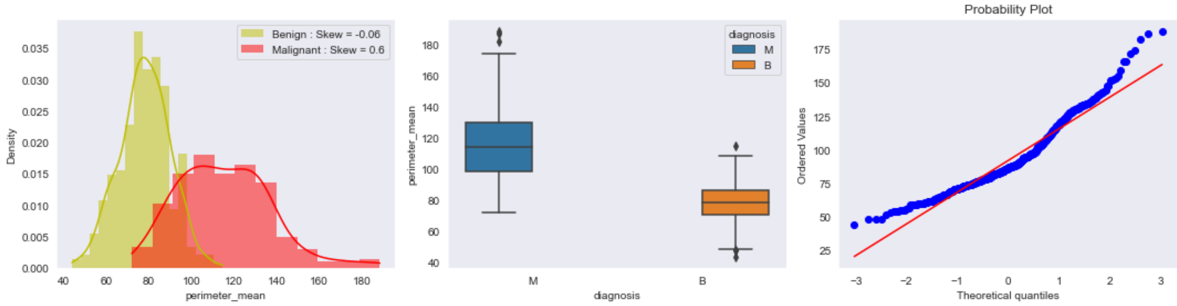
3.4 Visualisasi Histogram setiap atribut

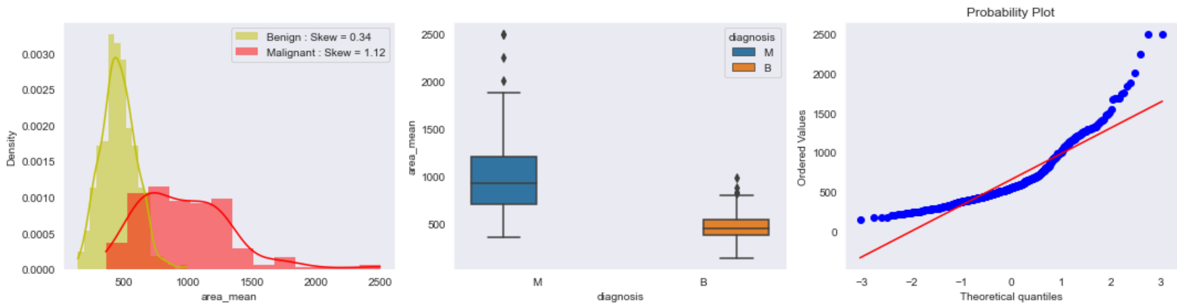
In [14]:

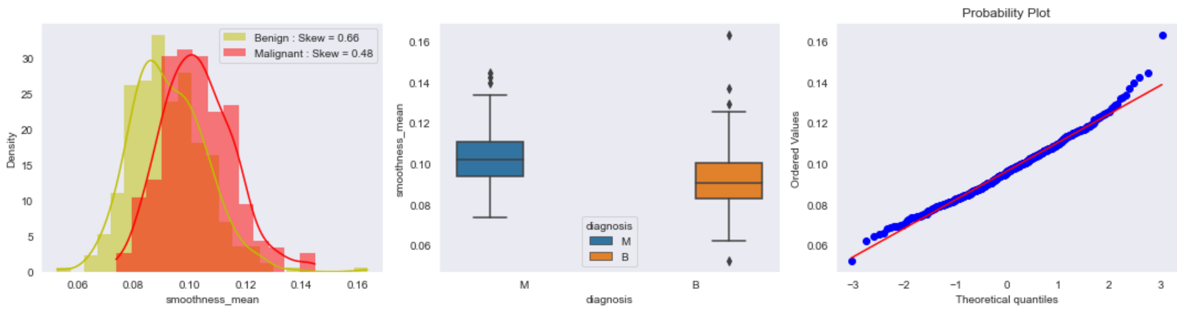
```
sns.set_style('dark')
for col in features_name:
    plt.figure(figsize=(15, 4))
    plt.title(col)
    plt.subplot(131)
    sns.histplot(
        wdbc[col][wdbc[target_name] == "B"],
        label="Benign :"+" Skew = " +
        str(np.round(wdbc[col][wdbc[target_name] == "B"].skew(), 2)),
        kde=True,
        color='y',
        stat="density",
        linewidth=0
    )
    sns.histplot(
        wdbc[col][wdbc[target_name] == "M"],
        label="Malignant :"+" Skew = " +
        str(np.round(wdbc[col][wdbc[target_name] == "M"].skew(), 2)),
        kde=True,
        color='r',
        stat="density",
        linewidth=0
    )
    plt.legend()
    plt.subplot(132)
    sns.boxplot(x=wdbc[target_name],
                y=wdbc[col],
                hue=wdbc[target_name])
    plt.subplot(133)
    stats.probplot(x=wdbc[col], plot=plt)
    plt.tight_layout()
    plt.show()
```

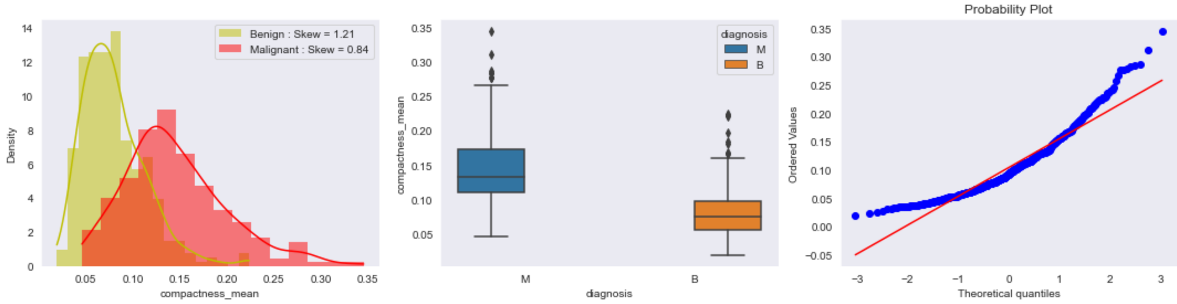


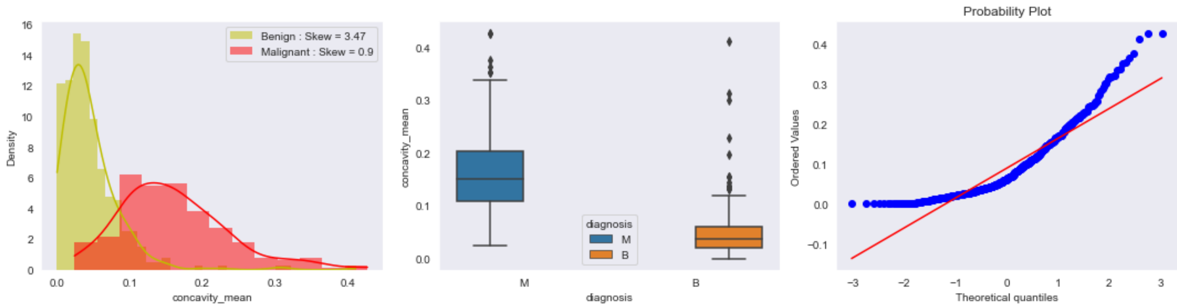

Benign : Skew = 0.98
Malignant : Skew = 0.7
Density
texture_mean
diagnosis
M
B
Probability Plot
Ordered Values
Theoretical quantiles

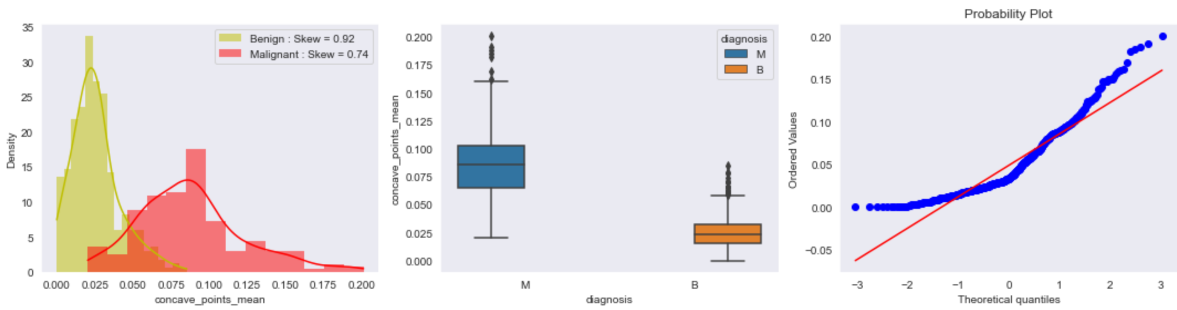

Benign : Skew = -0.06
Malignant : Skew = 0.6
Density
perimeter_mean
diagnosis
M
B
Probability Plot
Ordered Values
Theoretical quantiles

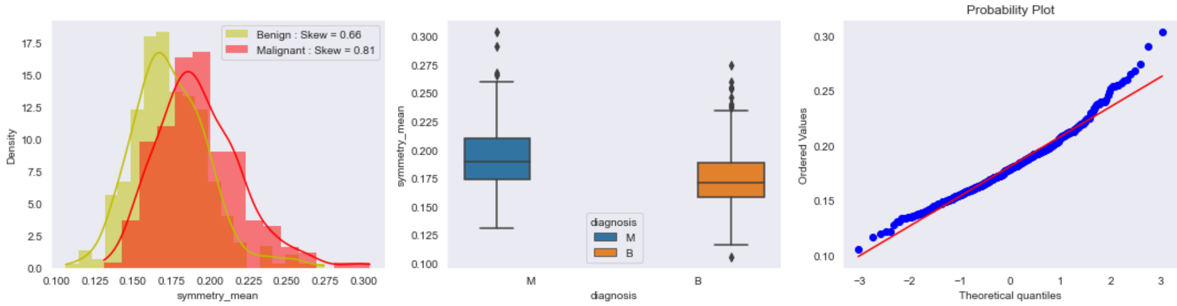

Benign : Skew = 0.34
Malignant : Skew = 1.12
Density
area_mean
diagnosis
M
B
Probability Plot
Ordered Values
Theoretical quantiles

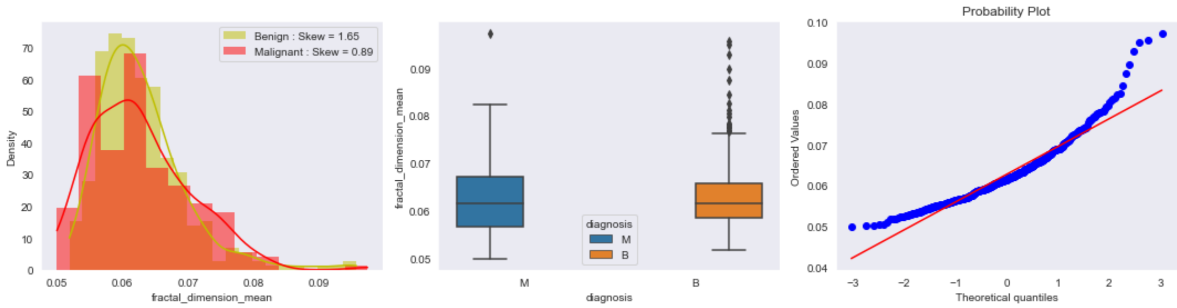

Benign : Skew = 0.66
Malignant : Skew = 0.48
Density
smoothness_mean
diagnosis
M
B
Probability Plot
Ordered Values
Theoretical quantiles

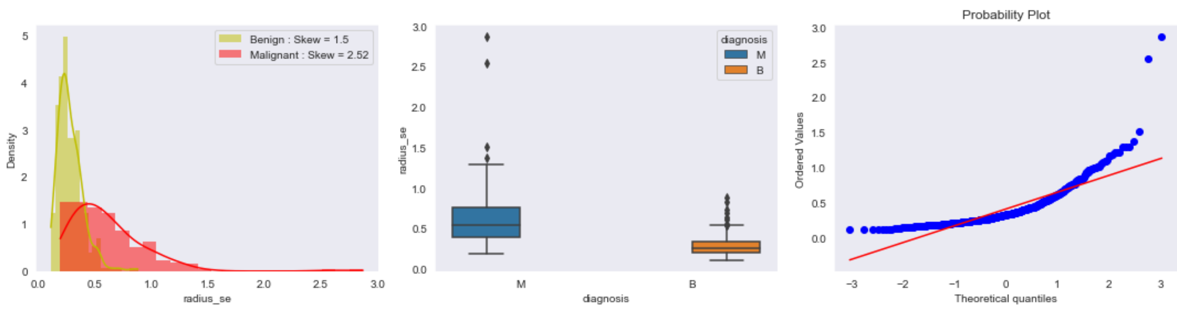

Benign : Skew = 1.21
Malignant : Skew = 0.84
Density
compactness_mean
diagnosis
M
B
compactness_mean
diagnosis
Probability Plot
Ordered Values
Theoretical quantiles

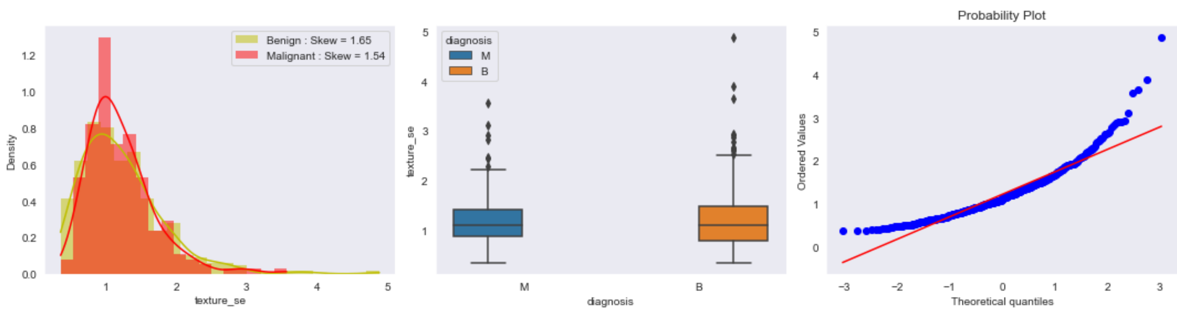

Benign : Skew = 3.47
Malignant : Skew = 0.9
Density
concavity_mean
diagnosis
M
B
concavity_mean
diagnosis
Probability Plot
Ordered Values
Theoretical quantiles

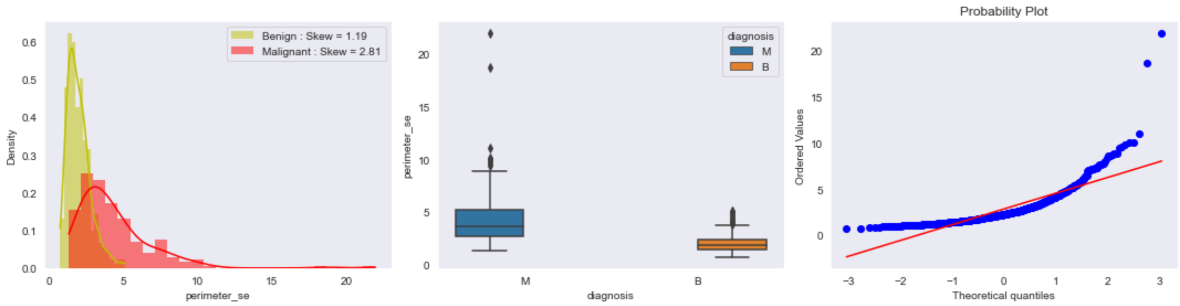

Benign : Skew = 0.92
Malignant : Skew = 0.74
Density
concave_points_mean
diagnosis
M
B
concave_points_mean
diagnosis
Probability Plot
Ordered Values
Theoretical quantiles

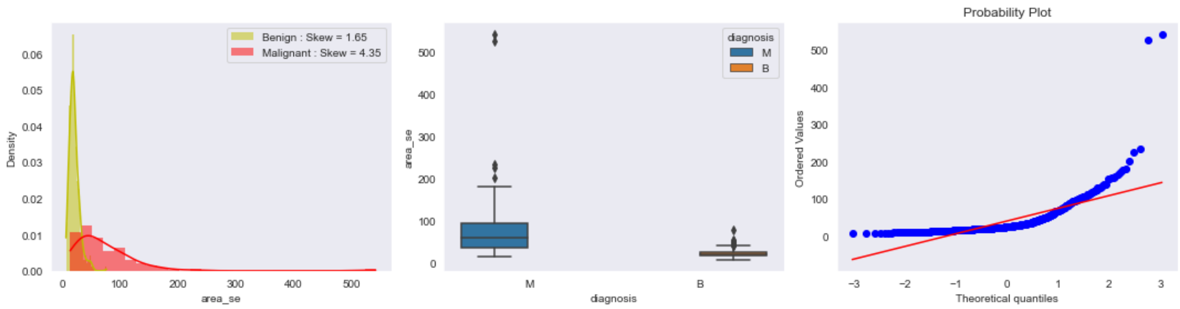

Benign : Skew = 0.66
Malignant : Skew = 0.81
Density
symmetry_mean
diagnosis
M
B
Probability Plot
Ordered Values
Theoretical quantiles

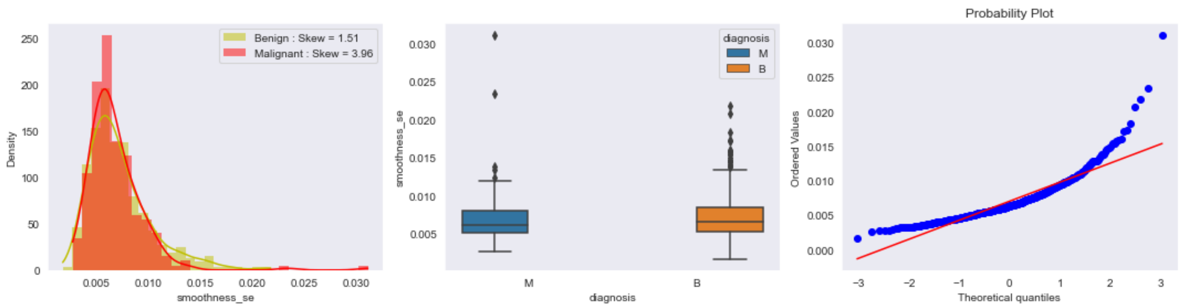

Benign : Skew = 1.65
Malignant : Skew = 0.89
Density
fractal_dimension_mean
diagnosis
M
B
Probability Plot
Ordered Values
Theoretical quantiles

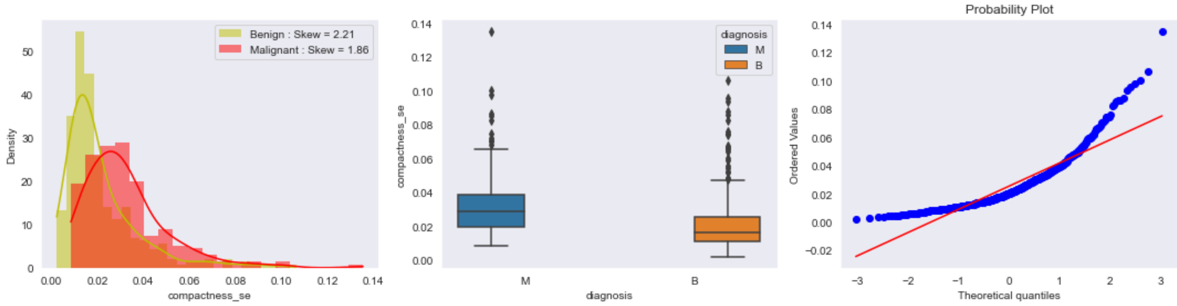

Benign : Skew = 1.5
Malignant : Skew = 2.52
Density
radius_se
diagnosis
M
B
Probability Plot
Ordered Values
Theoretical quantiles

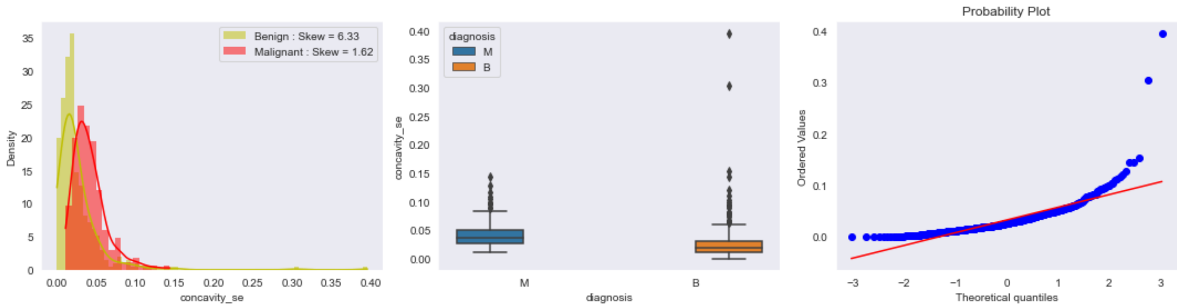

Benign : Skew = 1.65
Malignant : Skew = 1.54
Density
texture_se
diagnosis
M
B
Probability Plot
Ordered Values
Theoretical quantiles

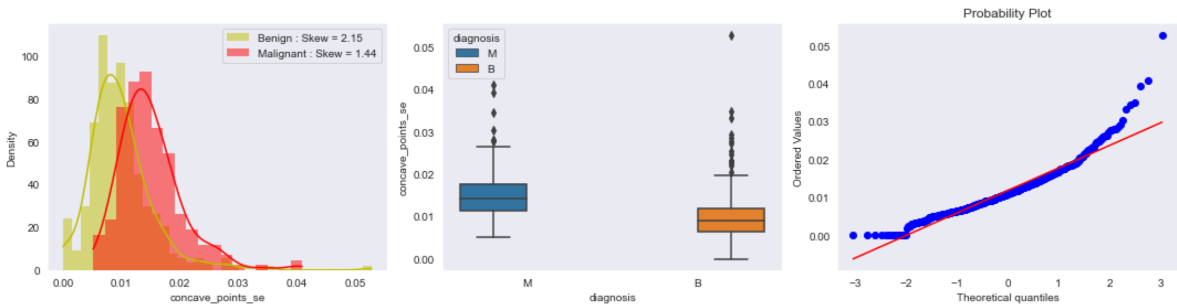

Benign : Skew = 1.19
Malignant : Skew = 2.81
Density
perimeter_se
perimeter_se
diagnosis
M
B
Probability Plot
Ordered Values
Theoretical quantiles

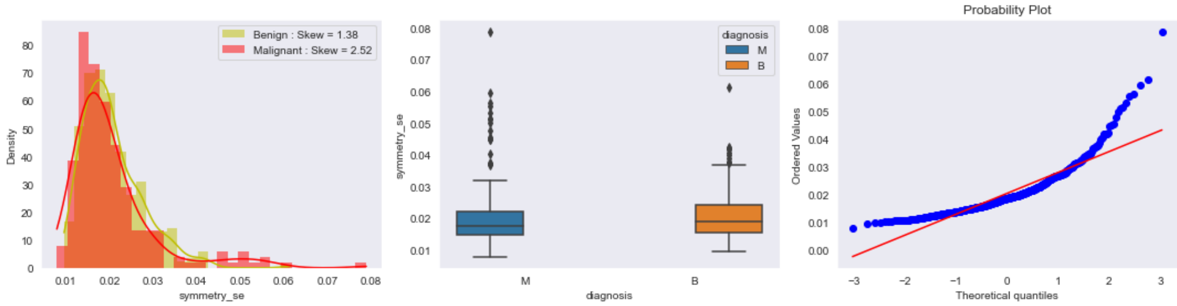

Benign : Skew = 1.65
Malignant : Skew = 4.35
Density
area_se
area_se
diagnosis
M
B
Probability Plot
Ordered Values
Theoretical quantiles

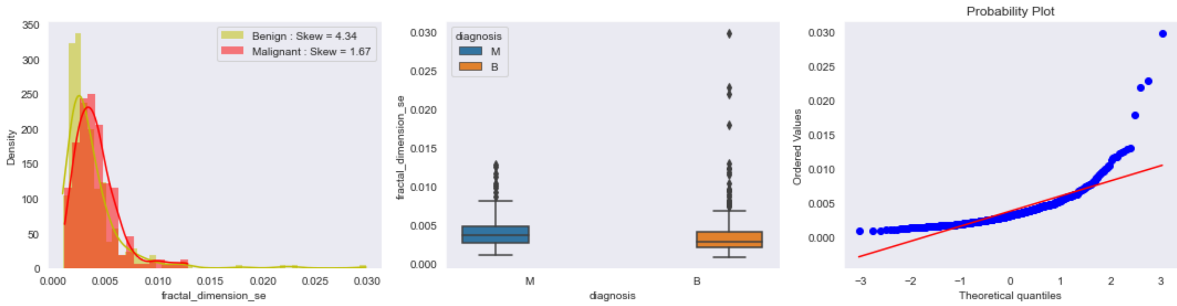

Benign : Skew = 1.51
Malignant : Skew = 3.96
Density
smoothness_se
smoothness_se
diagnosis
M
B
Probability Plot
Ordered Values
Theoretical quantiles

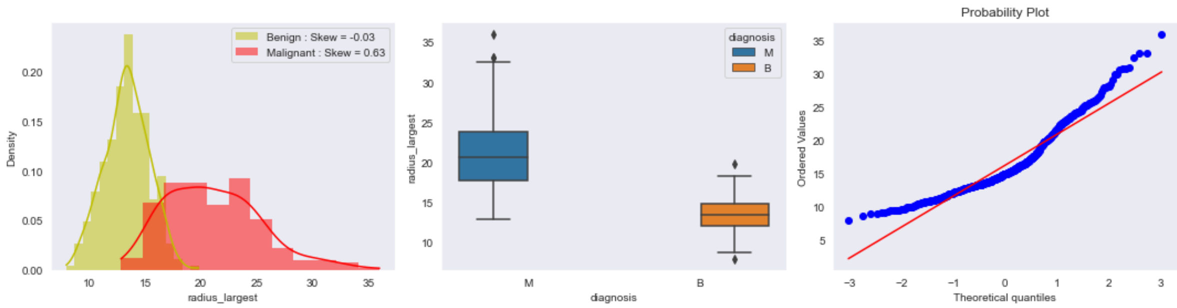

Benign : Skew = 2.21
Malignant : Skew = 1.86
Density
compactness_se
diagnosis
M
B
compactness_se
diagnosis
Probability Plot
Ordered Values
Theoretical quantiles

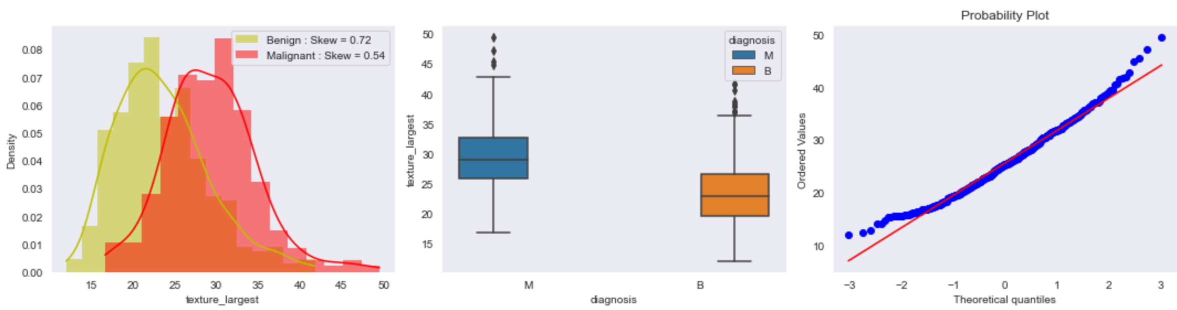

Benign : Skew = 6.33
Malignant : Skew = 1.62
Density
concavity_se
diagnosis
M
B
concavity_se
diagnosis
Probability Plot
Ordered Values
Theoretical quantiles

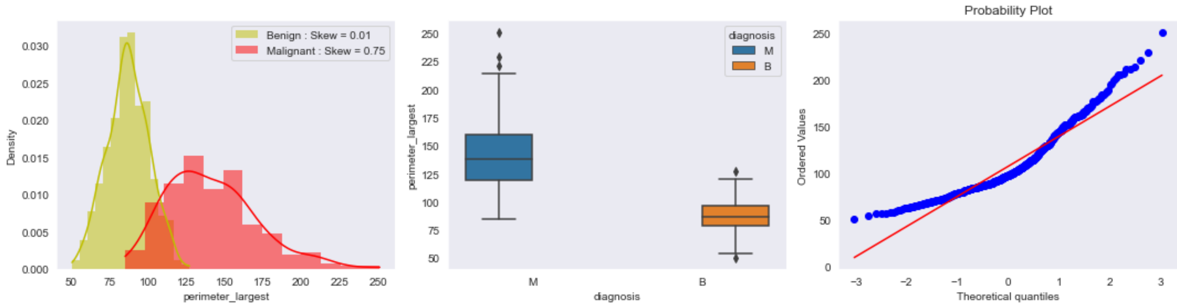

Benign : Skew = 2.15
Malignant : Skew = 1.44
Density
concave_points_se
diagnosis
M
B
concave_points_se
diagnosis
Probability Plot
Ordered Values
Theoretical quantiles

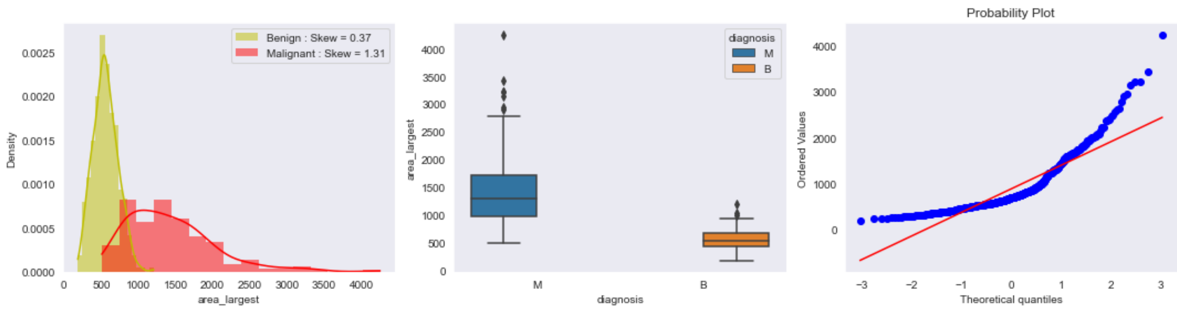

Benign : Skew = 1.38
Malignant : Skew = 2.52
Density
symmetry_se
diagnosis
M
B
Probability Plot
Ordered Values
Theoretical quantiles

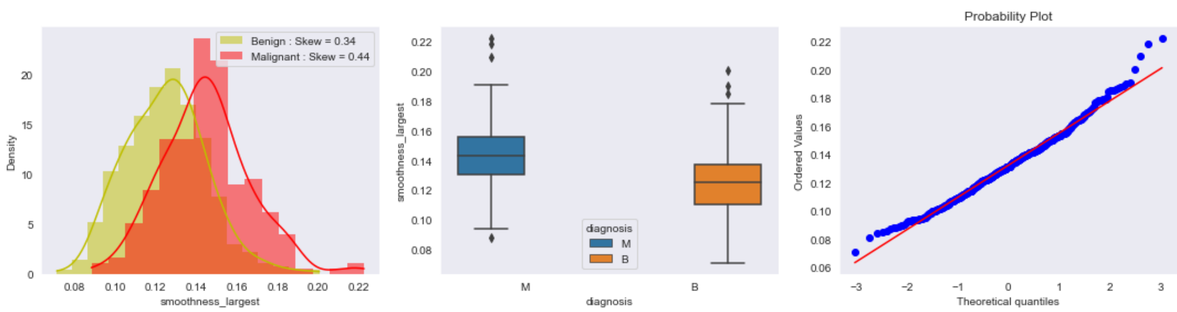

Benign : Skew = 4.34
Malignant : Skew = 1.67
Density
fractal_dimension_se
diagnosis
M
B
Probability Plot
Ordered Values
Theoretical quantiles

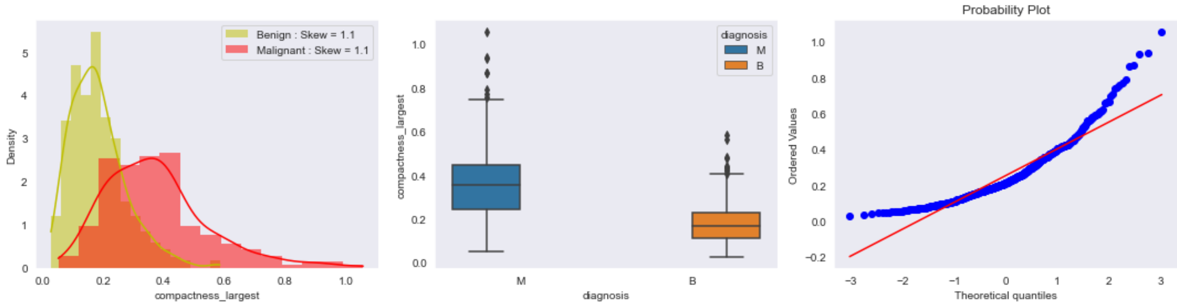

Benign : Skew = -0.03
Malignant : Skew = 0.63
Density
radius_largest
diagnosis
M
B
Probability Plot
Ordered Values
Theoretical quantiles

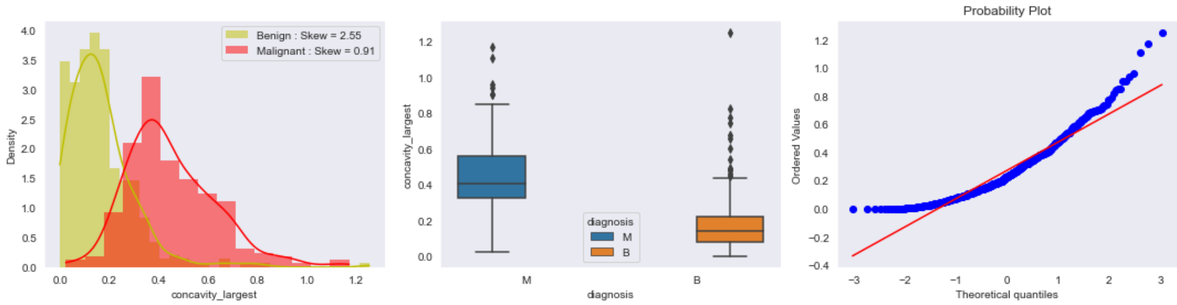

Benign : Skew = 0.72
Malignant : Skew = 0.54
Density
texture_largest
diagnosis
M
B
Probability Plot
Ordered Values
Theoretical quantiles

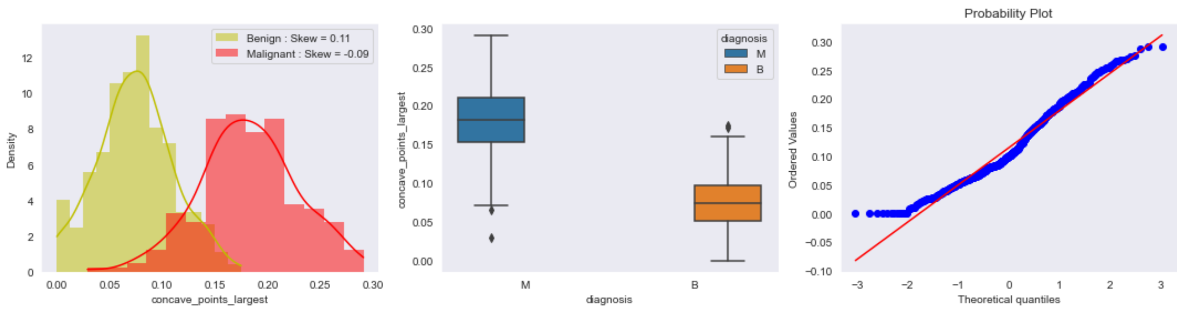

Benign : Skew = 0.01
Malignant : Skew = 0.75
Density
perimeter_largest
perimeter_largest
diagnosis
M
B
diagnosis
Probability Plot
Ordered Values
Theoretical quantiles

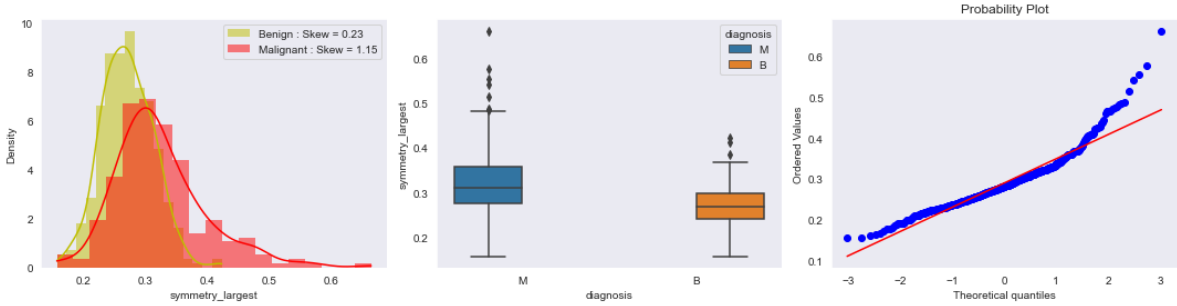

Benign : Skew = 0.37
Malignant : Skew = 1.31
Density
area_largest
area_largest
diagnosis
M
B
diagnosis
Probability Plot
Ordered Values
Theoretical quantiles

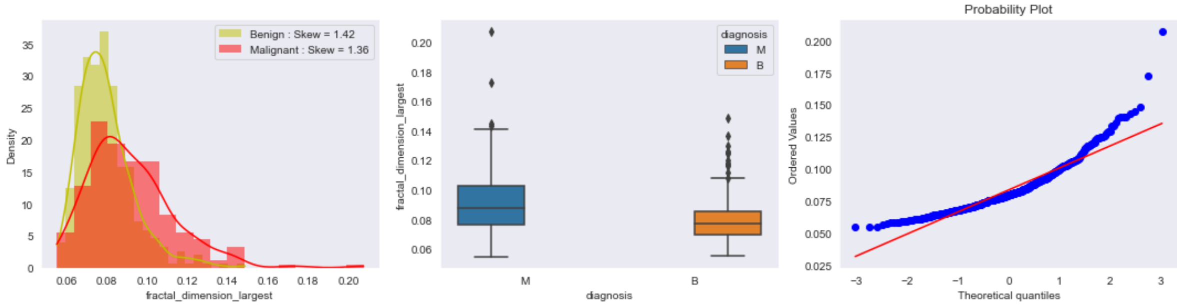

Benign : Skew = 0.34
Malignant : Skew = 0.44
Density
smoothness_largest
smoothness_largest
diagnosis
M
B
diagnosis
Probability Plot
Ordered Values
Theoretical quantiles


Benign : Skew = 1.1
Malignant : Skew = 1.1
Density
compactness_largest
diagnosis
M
B
compactness_largest
diagnosis
Probability Plot
Ordered Values
Theoretical quantiles


Benign : Skew = 2.55
Malignant : Skew = 0.91
Density
concavity_largest
diagnosis
M
B
concavity_largest
diagnosis
Probability Plot
Ordered Values
Theoretical quantiles


Benign : Skew = 0.11
Malignant : Skew = -0.09
Density
concave_points_largest
diagnosis
M
B
concave_points_largest
diagnosis
Probability Plot
Ordered Values
Theoretical quantiles





3.5 Correlations

3.5.1 Korelasi setiap atribut dengan label kelas

In [15]:

```
wdbc[target_name] = wdbc[target_name].map({
    'B':0,
    'M':1
})
```

In [16]:

```
wdbc.corr().loc[target_name].sort_values(ascending=False)
```

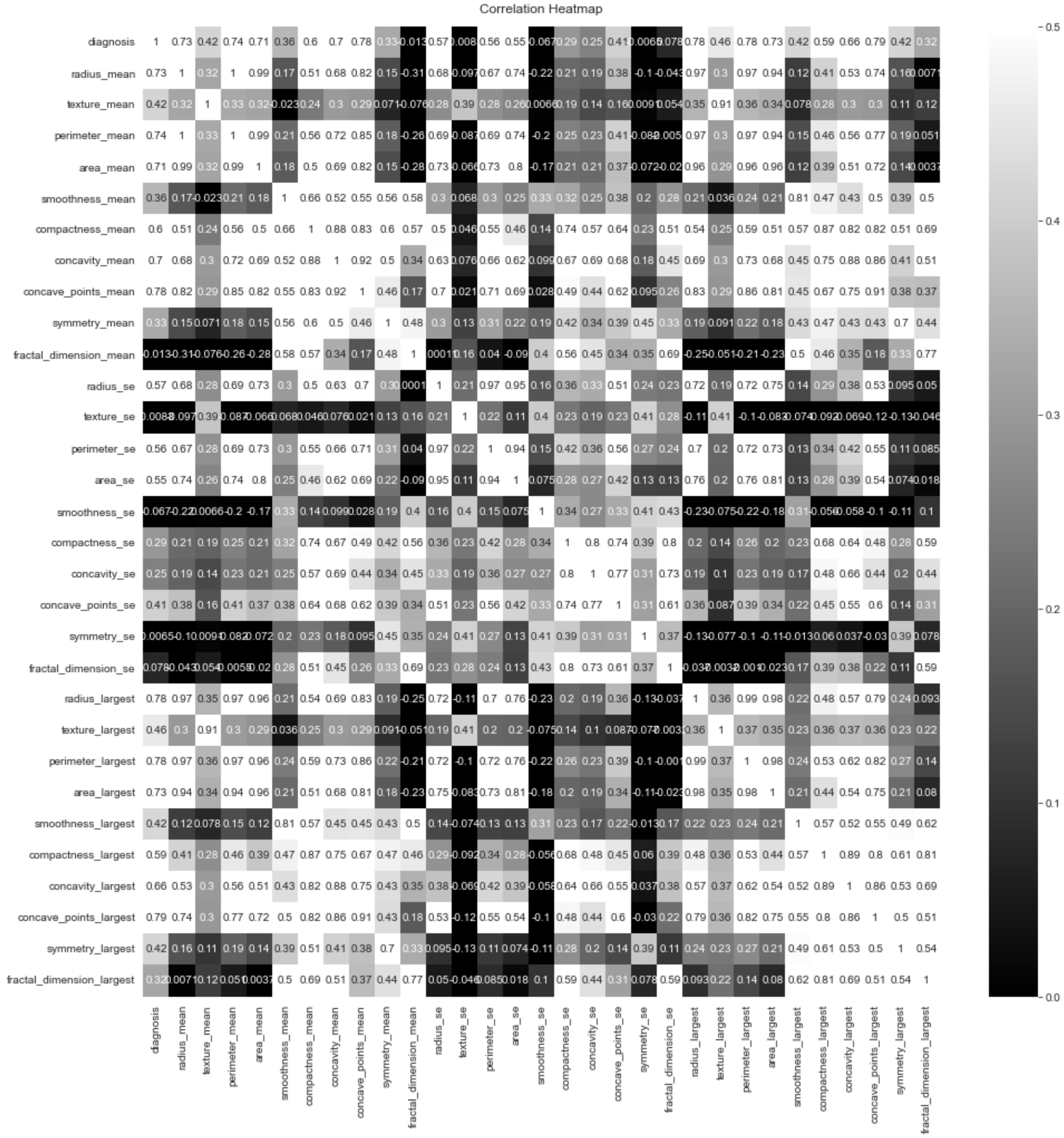
Out[16]:

```
diagnosis                  1.000000
concave_points_largest     0.793566
perimeter_largest          0.782914
concave_points_mean        0.776614
radius_largest             0.776454
perimeter_mean             0.742636
area_largest               0.733825
radius_mean                0.730029
area_mean                  0.708984
concavity_mean             0.696360
concavity_largest          0.659610
compactness_mean           0.596534
compactness_largest        0.590998
radius_se                  0.567134
perimeter_se               0.556141
area_se                    0.548236
texture_largest            0.456903
smoothness_largest         0.421465
symmetry_largest           0.416294
texture_mean               0.415185
concave_points_se          0.408042
smoothness_mean            0.358560
symmetry_mean              0.330499
fractal_dimension_largest  0.323872
compactness_se             0.292999
concavity_se               0.253730
fractal_dimension_se       0.077972
symmetry_se               -0.006522
texture_se                -0.008303
fractal_dimension_mean    -0.012838
smoothness_se             -0.067016
Name: diagnosis, dtype: float64
```

3.5.2 Matrix heatmap correlations

In [17]:

```
plt.figure(figsize=(16, 16))
# Store heatmap object in a variable to easily access it when you want to include more feat
# Set the range of values to be displayed on the colormap from -1 to 1, and set the annotat
heatmap = sns.heatmap(wdbc.corr(), vmin=0, vmax=0.5, annot=True, cmap="gray")
# Give a title to the heatmap. Pad defines the distance of the title from the top of the he
heatmap.set_title('Correlation Heatmap', fontdict={'fontsize':12}, pad=12);

# save heatmap
# plt.savefig('heatmap.png', dpi=300, bbox_inches='tight')
```



4 Data Splitting

Split dataset into training (80%) and testing (20%)

In [18]:

```
wdbc = pd.read_csv('../dataset/data.csv')
wdbc.drop(['IDNumber'],axis=1, inplace=True)
target_name = 'diagnosis'
X = wdbc.drop(target_name,axis=1).copy()
y = wdbc[target_name].copy()
features_name = X.columns.tolist()
```

4.1 Label Encoding

In [19]:

```
targetEncoder = LabelEncoder()
y = targetEncoder.fit_transform(y)
```

In [20]:

```
targetEncoder.inverse_transform([1,0])
```

Out[20]:

```
array(['M', 'B'], dtype=object)
```

4.2 Splitting

In [21]:

```
test_size = 0.2
train_size = 1 - test_size
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=test_size, random_state
```

In [22]:

```
pd.Series(y_train).value_counts(),pd.Series(y_test).value_counts()
```

Out[22]:

```
(0    285
 1    170
 dtype: int64,
 0    72
 1    42
 dtype: int64)
```

4.3 Simpan Dataset Hasil Splitting

In [23]:

```
# simpan dataset hasil splitting awal ke dalam pickle untuk ditampilkan pada website
wdbc_set = {
    'X': X.to_numpy(),
    'y': y
}
training_set = {
    'X_train': X_train.to_numpy(),
    'y_train': y_train,
}
testing_set = {
    'X_test': X_test.to_numpy(),
    'y_test': y_test,
}
```

4.3.1 Simpan Keseluruhan Data sebelum splitting

In [24]:

```
X = wdbc_set['X']
y = np.reshape(wdbc_set['y'], (-1,1))
wdbc_set = np.concatenate((X, y),axis=1)
```

4.3.2 Simpan Himpunan data latih

In [25]:

```
X_latih = training_set['X_train']
y_latih = np.reshape(training_set['y_train'], (-1,1))
n_sampel_training = X_latih.shape[0]
persentase_sampel_training = train_size
training_set = np.concatenate((X_latih, y_latih),axis=1)
df_train = pd.DataFrame(training_set, columns=features_name+['diagnosis'])


df_train.to_excel("informations/data_train.xlsx")
```

4.3.3 Simpan Himpunan data uji

In [26]:

```
X_uji = testing_set['X_test']
y_uji = np.reshape(testing_set['y_test'], (-1,1))
n_sampel_testing = X_uji.shape[0]
persentase_sampel_testing = test_size
testing_set = np.concatenate((X_uji, y_uji),axis=1)
df_testing = pd.DataFrame(testing_set, columns=features_name+['diagnosis'])

df_testing.to_excel("informations/data_testing.xlsx")
```

4.3.4 Simpan ke dalam format pickle

In [27]:

```
dataset = {
    'all': wdbc_set,
    'training':{
        'data': training_set,
        'n_sampel': n_sampel_training,
        'persentase_sampel': persentase_sampel_training
    },
    'testing':{
        'data': testing_set,
        'n_sampel': n_sampel_testing,
        'persentase_sampel': persentase_sampel_testing
    },
    'features_name': features_name,
}
# save the dataset to disk
pickle.dump(dataset, open('results/dataset.pkl', 'wb'))
```