2.1.Random-Forest-Feature-Selection

May 15, 2023

[262]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import time
from subprocess import check_output

from scipy import stats
plt.style.use("ggplot")
import warnings
warnings.filterwarnings("ignore")
from scipy import stats
```

[263]:
```
data=pd.read_csv('wdbc.data',header=None)
```

data.head()

[264]:
```
headers=['id','diagnosis','mean_radius','mean_texture','mean_perimeter','mean_area','mean_smoo
  ↪points','mean_symmetry','mean_fractal␣
  ↪dimension','SE_radius','SE_texture','SE_perimeter','SE_area','SE_smoothness','SE_compactnes
  ↪points','SE_symmetry','SE_fractal␣
  ↪dimension','worst_radius','worst_texture','worst_perimeter','worst_area','worst_smoothness'
  ↪points','worst_symmetry','worst_fractal dimension']
```

[265]:
```
data.to_csv('labeledData.csv',header=headers,index=False)
```

[266]:
```
data=pd.read_csv('labeledData.csv')
data.head()
```

[266]:
```
         id diagnosis  mean_radius  mean_texture  mean_perimeter  mean_area
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

   mean_smoothness  mean_compactness  mean_concavity  mean_concave points
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

   …  worst_radius  worst_texture  worst_perimeter  worst_area  \
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

   worst_smoothness  worst_compactness  worst_concavity  worst_concave points  \
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

   worst_symmetry  worst_fractal dimension
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]
```

[267]:
```
data.shape
```

[267]:
```
(569, 32)
```

[268]:
```
data.isna().sum()
```

[268]:
```
id                         0
diagnosis                  0
mean_radius                0
mean_texture               0
mean_perimeter             0
mean_area                  0
mean_smoothness            0
mean_compactness           0
mean_concavity             0
mean_concave points        0
mean_symmetry              0
mean_fractal dimension     0
SE_radius                  0
```

```
SE_texture                 0
SE_perimeter               0
SE_area                    0
SE_smoothness              0
SE_compactness             0
SE_concavity               0
SE_concave points          0
SE_symmetry                0
SE_fractal dimension       0
worst_radius               0
worst_texture              0
worst_perimeter            0
worst_area                 0
worst_smoothness           0
worst_compactness          0
worst_concavity            0
worst_concave points       0
worst_symmetry             0
worst_fractal dimension    0
dtype: int64
```

[269]:
```
data['diagnosis'].value_counts()
```

[269]:
```
diagnosis
B    357
M    212
Name: count, dtype: int64
```

[270]:
```
data.dtypes
```

[270]:
```
id                           int64
diagnosis                   object
mean_radius                float64
mean_texture               float64
mean_perimeter             float64
mean_area                  float64
mean_smoothness            float64
mean_compactness           float64
mean_concavity             float64
mean_concave points        float64
mean_symmetry              float64
mean_fractal dimension     float64
SE_radius                  float64
SE_texture                 float64
SE_perimeter               float64
SE_area                    float64
SE_smoothness              float64
```

```
SE_compactness            float64
SE_concavity              float64
SE_concave points         float64
SE_symmetry               float64
SE_fractal dimension      float64
worst_radius              float64
worst_texture             float64
worst_perimeter           float64
worst_area                float64
worst_smoothness          float64
worst_compactness         float64
worst_concavity           float64
worst_concave points      float64
worst_symmetry            float64
worst_fractal dimension   float64
dtype: object
```

[271]:

```
list=['id','diagnosis']
y=data.diagnosis
x=data.drop(list,axis=1)
x.head()
```

[271]:

```
   mean_radius  mean_texture  mean_perimeter  mean_area  mean_smoothness  \
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

   mean_compactness  mean_concavity  mean_concave points  mean_symmetry  \
0           0.27760          0.3001              0.14710         0.2419
1           0.07864          0.0869              0.07017         0.1812
2           0.15990          0.1974              0.12790         0.2069
3           0.28390          0.2414              0.10520         0.2597
4           0.13280          0.1980              0.10430         0.1809

   mean_fractal dimension  …  worst_radius  worst_texture  worst_perimeter  \
0                 0.07871  …         25.38          17.33           184.60
1                 0.05667  …         24.99          23.41           158.80
2                 0.05999  …         23.57          25.53           152.50
3                 0.09744  …         14.91          26.50            98.87
4                 0.05883  …         22.54          16.67           152.20

   worst_area  worst_smoothness  worst_compactness  worst_concavity  \
0      2019.0            0.1622             0.6656           0.7119
1      1956.0            0.1238             0.1866           0.2416
2      1709.0            0.1444             0.4245           0.4504
```

```
3      567.7            0.2098            0.8663            0.6869
4     1575.0            0.1374            0.2050            0.4000

   worst_concave points  worst_symmetry  worst_fractal dimension
0                0.2654          0.4601                  0.11890
1                0.1860          0.2750                  0.08902
2                0.2430          0.3613                  0.08758
3                0.2575          0.6638                  0.17300
4                0.1625          0.2364                  0.07678

[5 rows x 30 columns]
```

[272]:
```
x.describe()
```

[272]:
```
       mean_radius  mean_texture  mean_perimeter    mean_area  \
count   569.000000    569.000000      569.000000   569.000000
mean     14.127292     19.289649       91.969033   654.889104
std       3.524049      4.301036       24.298981   351.914129
min       6.981000      9.710000       43.790000   143.500000
25%      11.700000     16.170000       75.170000   420.300000
50%      13.370000     18.840000       86.240000   551.100000
75%      15.780000     21.800000      104.100000   782.700000
max      28.110000     39.280000      188.500000  2501.000000

       mean_smoothness  mean_compactness  mean_concavity  mean_concave points  \
count       569.000000        569.000000      569.000000           569.000000
mean          0.096360          0.104341        0.088799             0.048919
std           0.014064          0.052813        0.079720             0.038803
min           0.052630          0.019380        0.000000             0.000000
25%           0.086370          0.064920        0.029560             0.020310
50%           0.095870          0.092630        0.061540             0.033500
75%           0.105300          0.130400        0.130700             0.074000
max           0.163400          0.345400        0.426800             0.201200

       mean_symmetry  mean_fractal dimension  …  worst_radius  \
count     569.000000              569.000000  …    569.000000
mean        0.181162                0.062798  …     16.269190
std         0.027414                0.007060  …      4.833242
min         0.106000                0.049960  …      7.930000
25%         0.161900                0.057700  …     13.010000
50%         0.179200                0.061540  …     14.970000
75%         0.195700                0.066120  …     18.790000
max         0.304000                0.097440  …     36.040000

       worst_texture  worst_perimeter  worst_area  worst_smoothness  \
count     569.000000       569.000000  569.000000        569.000000
mean       25.677223       107.261213  880.583128          0.132369
```

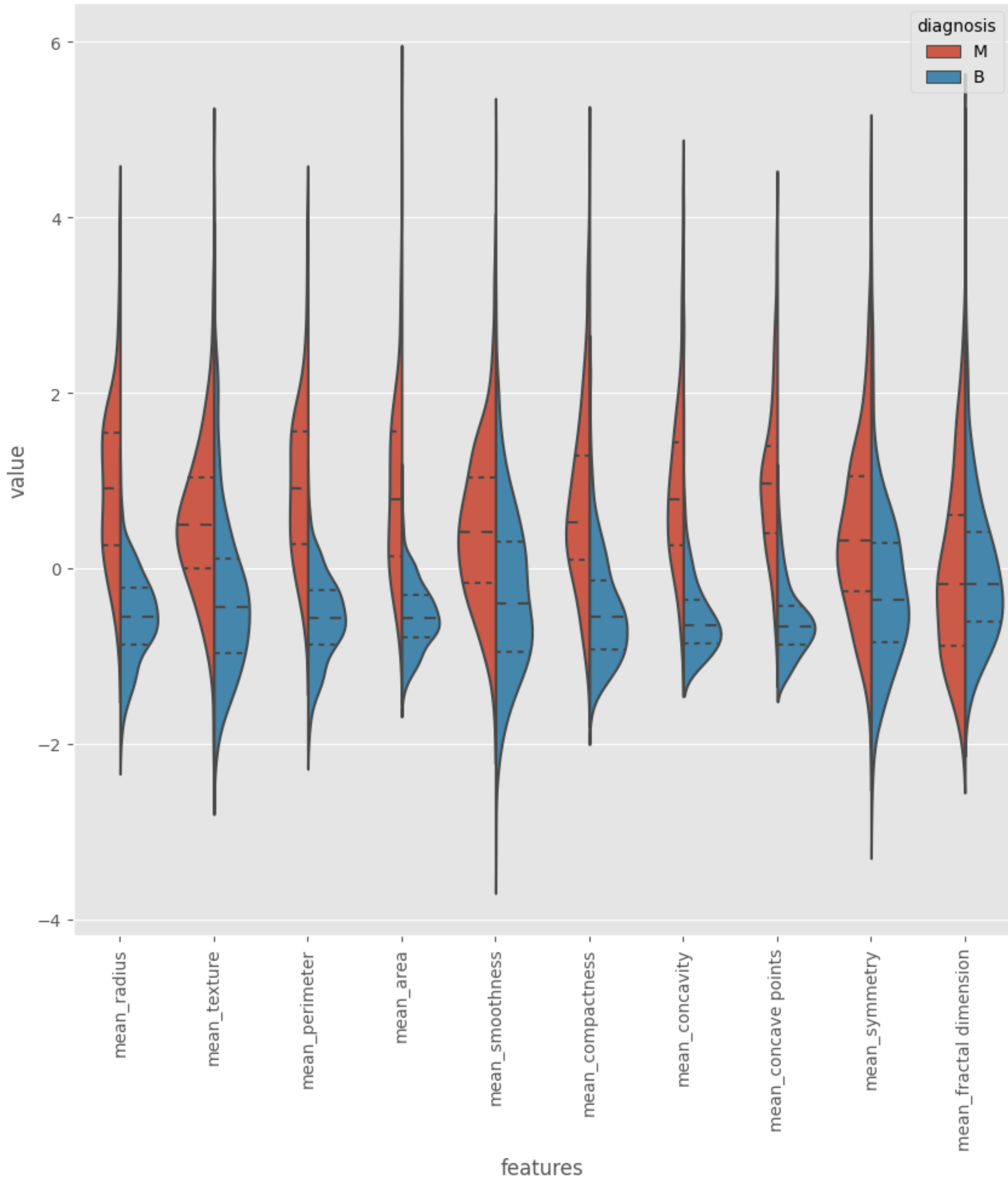
```
std         6.146258        33.602542   569.356993          0.022832
min        12.020000        50.410000   185.200000          0.071170
25%        21.080000        84.110000   515.300000          0.116600
50%        25.410000        97.660000   686.500000          0.131300
75%        29.720000       125.400000  1084.000000          0.146000
max        49.540000       251.200000  4254.000000          0.222600

       worst_compactness  worst_concavity  worst_concave points  
count         569.000000       569.000000            569.000000  \
mean            0.254265         0.272188              0.114606
std             0.157336         0.208624              0.065732
min             0.027290         0.000000              0.000000
25%             0.147200         0.114500              0.064930
50%             0.211900         0.226700              0.099930
75%             0.339100         0.382900              0.161400
max             1.058000         1.252000              0.291000

       worst_symmetry  worst_fractal dimension
count      569.000000               569.000000
mean         0.290076                 0.083946
std          0.061867                 0.018061
min          0.156500                 0.055040
25%          0.250400                 0.071460
50%          0.282200                 0.080040
75%          0.317900                 0.092080
max          0.663800                 0.207500

[8 rows x 30 columns]
```

[273]:
```python
diag=y
data=x
data_std=(data-data.mean())/(data.std())
```

[274]:
```python
data=pd.concat([y,data_std.iloc[:,0:10]],axis=1)
data=pd.melt(data,id_vars='diagnosis',var_name='features',value_name='value')
plt.figure(figsize=(10,10))
sns.
 ↪violinplot(x='features',y='value',hue='diagnosis',data=data,split=True,inner='quart')
plt.xticks(rotation=90)
```

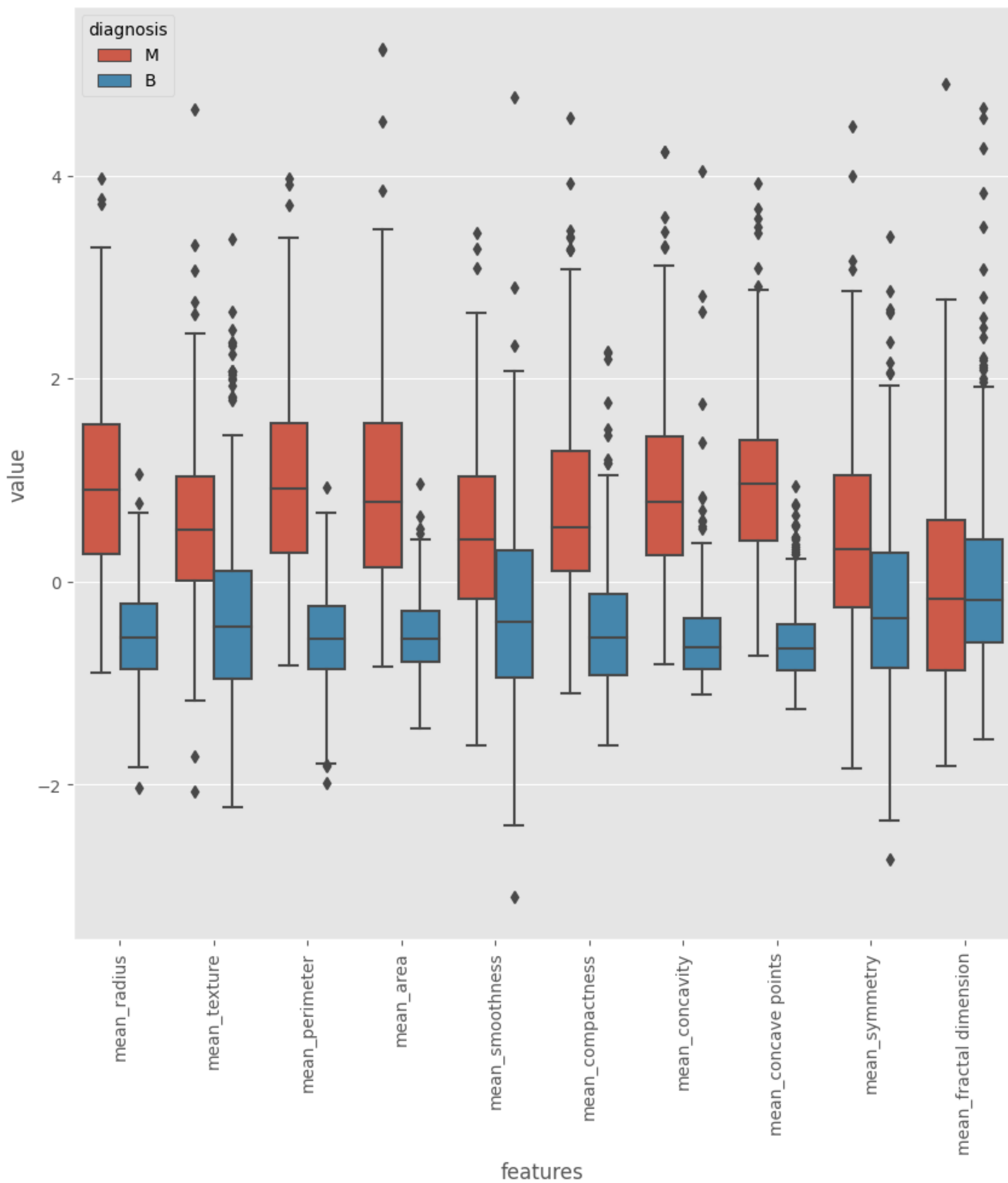
[274]:
```
(array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9]),
 [Text(0, 0, 'mean_radius'),
  Text(1, 0, 'mean_texture'),
  Text(2, 0, 'mean_perimeter'),
  Text(3, 0, 'mean_area'),
  Text(4, 0, 'mean_smoothness'),
  Text(5, 0, 'mean_compactness'),
```

```
 Text(6, 0, 'mean_concavity'),
 Text(7, 0, 'mean_concave points'),
 Text(8, 0, 'mean_symmetry'),
 Text(9, 0, 'mean_fractal dimension')])
```



[275]:
```
plt.figure(figsize=(10,10))
sns.boxplot(x="features",y="value",hue='diagnosis',data=data)
plt.xticks(rotation=90)
```

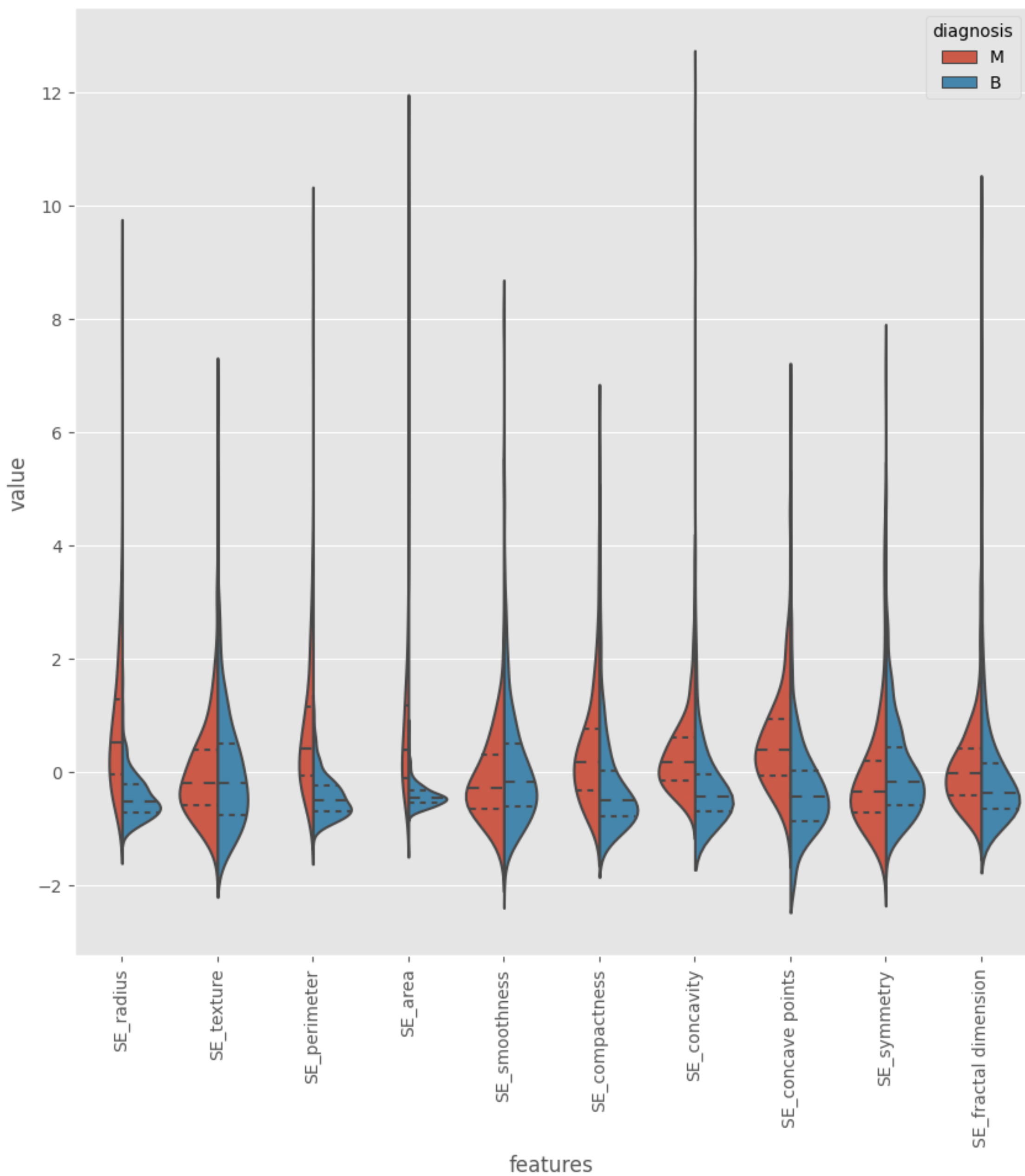
```
[275]: (array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9]),
        [Text(0, 0, 'mean_radius'),
         Text(1, 0, 'mean_texture'),
         Text(2, 0, 'mean_perimeter'),
         Text(3, 0, 'mean_area'),
         Text(4, 0, 'mean_smoothness'),
         Text(5, 0, 'mean_compactness'),
         Text(6, 0, 'mean_concavity'),
         Text(7, 0, 'mean_concave points'),
         Text(8, 0, 'mean_symmetry'),
         Text(9, 0, 'mean_fractal dimension')])
```



[276]:

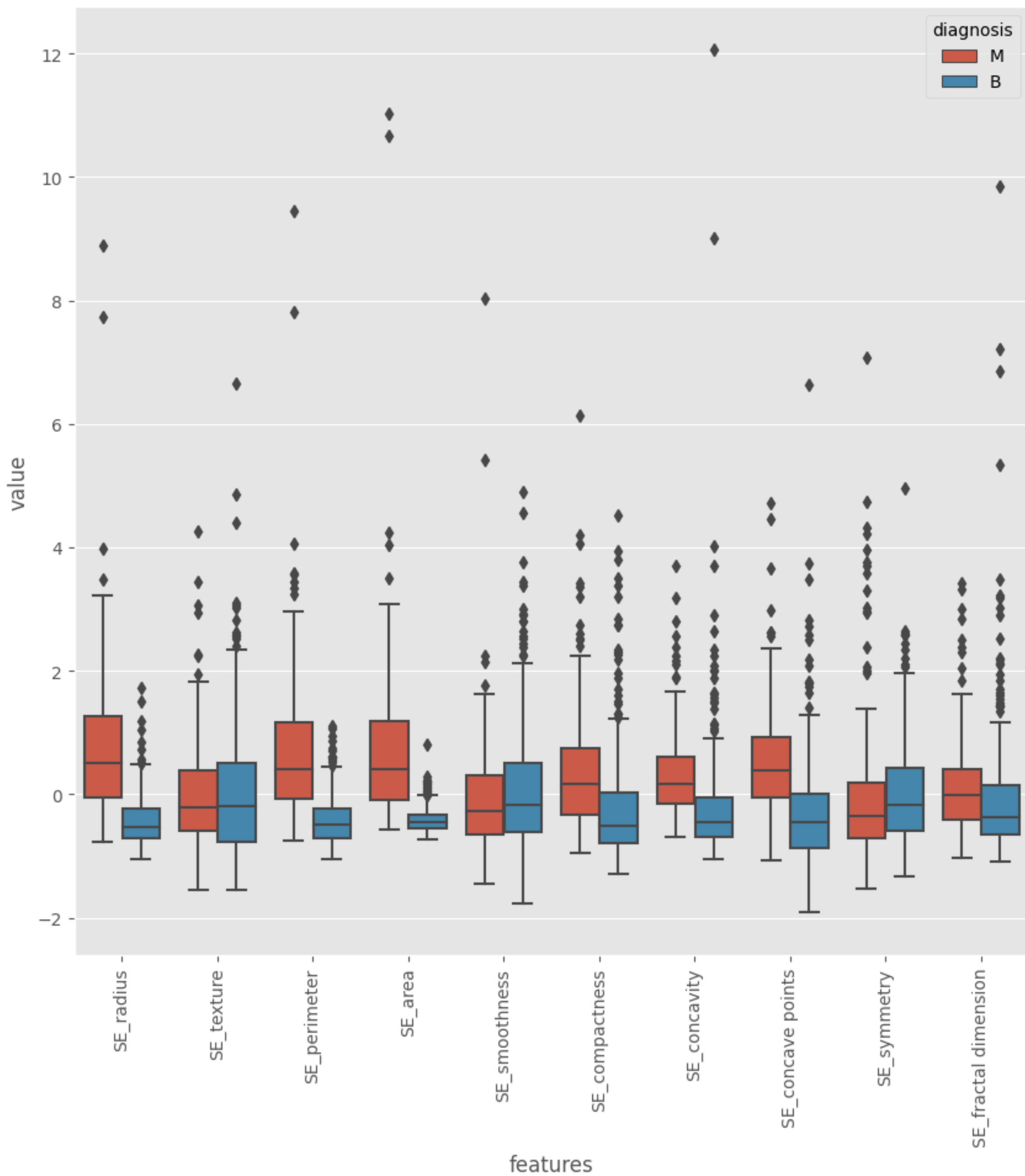
```
data = pd.concat([y,data_std.iloc[:,10:20]],axis=1)
data = pd.melt(data,id_vars="diagnosis",
                    var_name="features",
                    value_name='value')
plt.figure(figsize=(10,10))
sns.violinplot(x="features", y="value", hue="diagnosis", data=data,split=True,␣
 ↪inner="quart")
plt.xticks(rotation=90)
```

```
[276]: (array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9]),
 [Text(0, 0, 'SE_radius'),
  Text(1, 0, 'SE_texture'),
  Text(2, 0, 'SE_perimeter'),
  Text(3, 0, 'SE_area'),
  Text(4, 0, 'SE_smoothness'),
  Text(5, 0, 'SE_compactness'),
  Text(6, 0, 'SE_concavity'),
  Text(7, 0, 'SE_concave points'),
  Text(8, 0, 'SE_symmetry'),
  Text(9, 0, 'SE_fractal dimension')])
```



[277]:
```
plt.figure(figsize=(10,10))
sns.boxplot(x="features",y="value",hue='diagnosis',data=data)
plt.xticks(rotation=90)
```

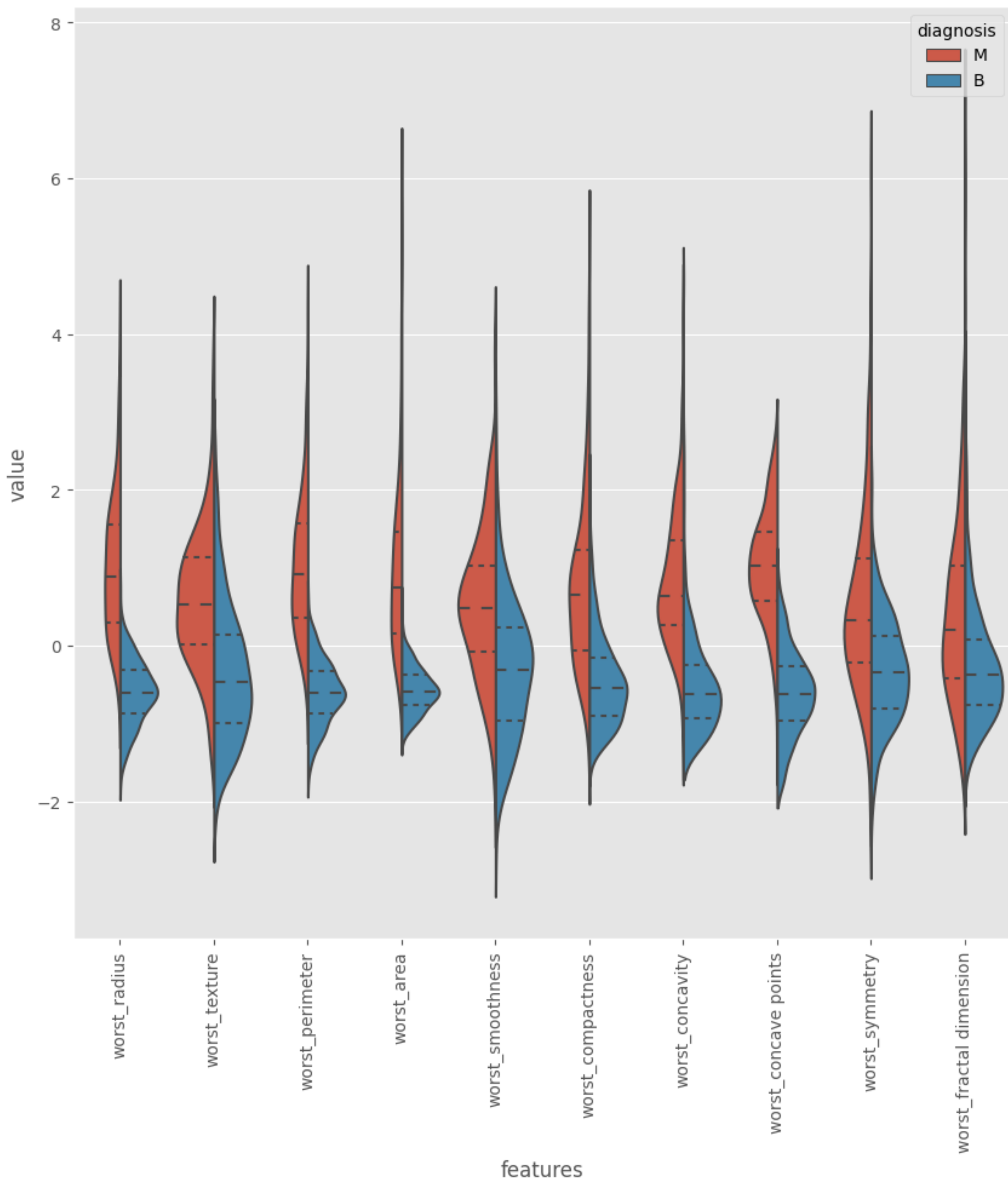
[277]:
```
(array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9]),
 [Text(0, 0, 'SE_radius'),
  Text(1, 0, 'SE_texture'),
  Text(2, 0, 'SE_perimeter'),
  Text(3, 0, 'SE_area'),
  Text(4, 0, 'SE_smoothness'),
  Text(5, 0, 'SE_compactness'),
  Text(6, 0, 'SE_concavity'),
  Text(7, 0, 'SE_concave points'),
  Text(8, 0, 'SE_symmetry'),
  Text(9, 0, 'SE_fractal dimension')])
```



[278]:

```
data = pd.concat([y,data_std.iloc[:,20:31]],axis=1)
data = pd.melt(data,id_vars="diagnosis",
                    var_name="features",
                    value_name='value')
plt.figure(figsize=(10,10))
sns.violinplot(x="features", y="value", hue="diagnosis", data=data,split=True,␣
 ↪inner="quart")
plt.xticks(rotation=90)
```

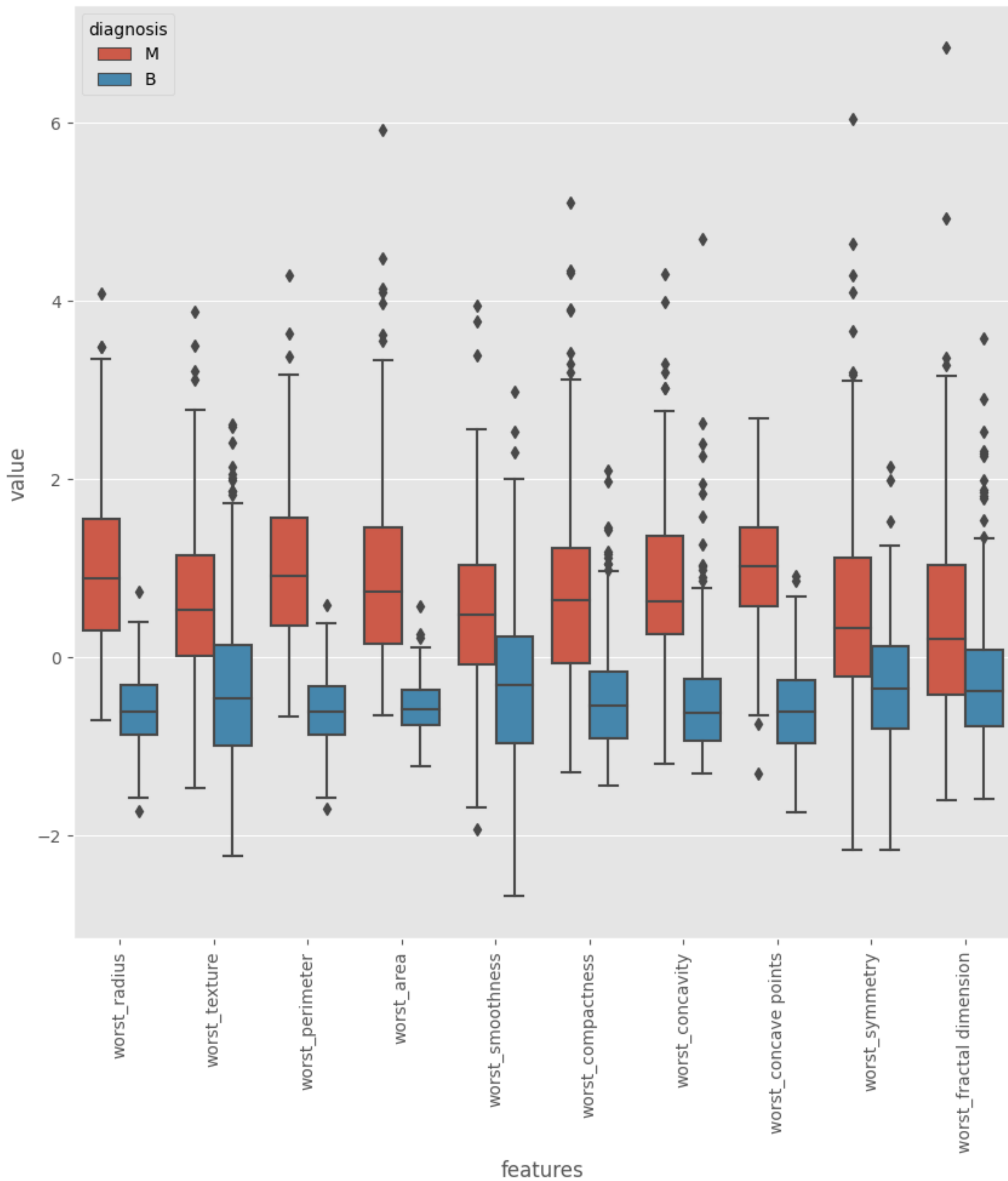
[278]:
```
(array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9]),
 [Text(0, 0, 'worst_radius'),
  Text(1, 0, 'worst_texture'),
  Text(2, 0, 'worst_perimeter'),
  Text(3, 0, 'worst_area'),
  Text(4, 0, 'worst_smoothness'),
  Text(5, 0, 'worst_compactness'),
  Text(6, 0, 'worst_concavity'),
  Text(7, 0, 'worst_concave points'),
  Text(8, 0, 'worst_symmetry'),
  Text(9, 0, 'worst_fractal dimension')])
```



[279]:
```
plt.figure(figsize=(10,10))
sns.boxplot(x="features",y="value",hue='diagnosis',data=data)
plt.xticks(rotation=90)
```

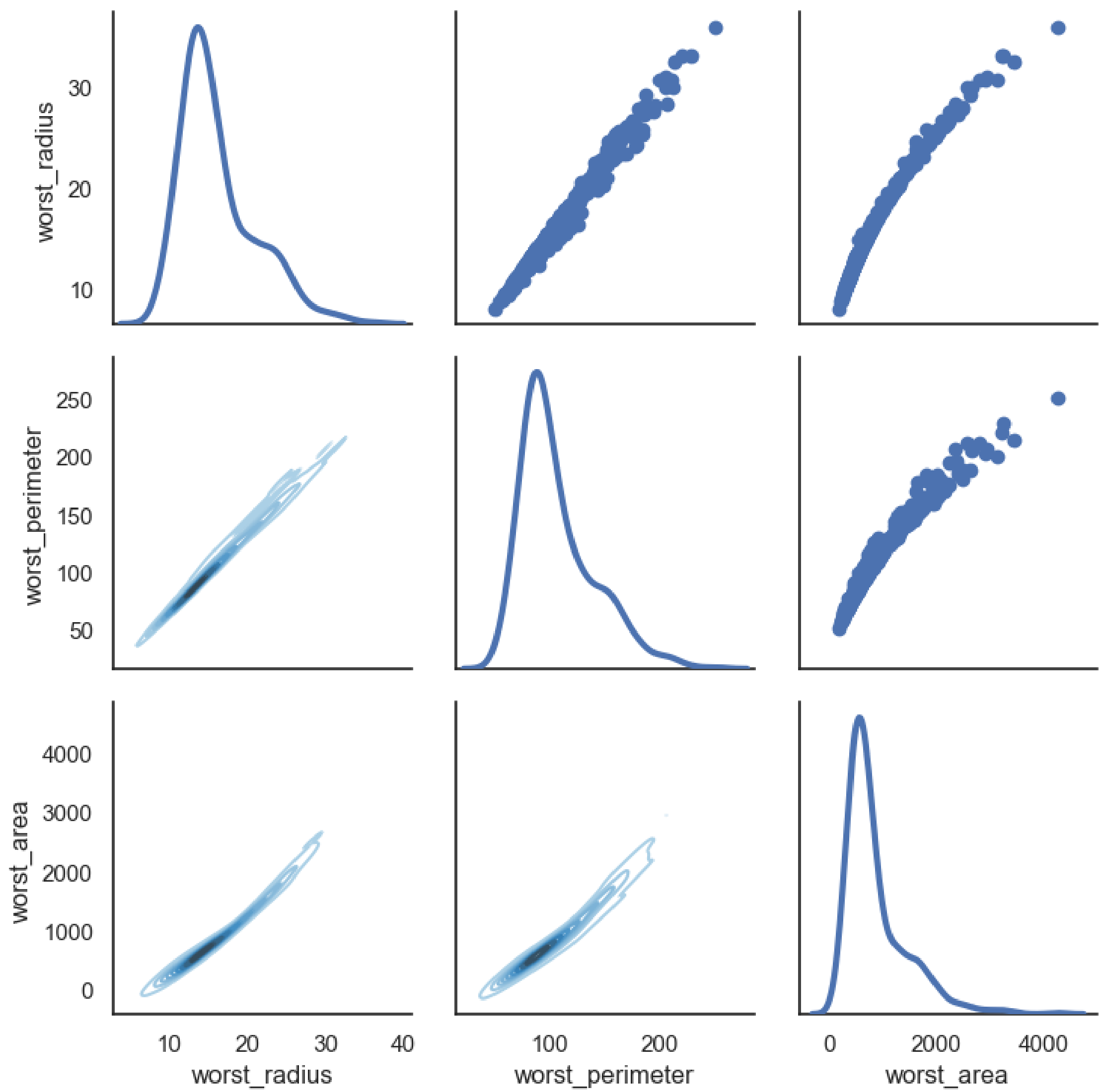
[279]:
```
(array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9]),
 [Text(0, 0, 'worst_radius'),
  Text(1, 0, 'worst_texture'),
  Text(2, 0, 'worst_perimeter'),
```

```
Text(3, 0, 'worst_area'),
Text(4, 0, 'worst_smoothness'),
Text(5, 0, 'worst_compactness'),
Text(6, 0, 'worst_concavity'),
Text(7, 0, 'worst_concave points'),
Text(8, 0, 'worst_symmetry'),
Text(9, 0, 'worst_fractal dimension')])
```



[280]:
```
sns.set(style='white')
df=x.loc[:,['worst_radius','worst_perimeter','worst_area']]
g=sns.PairGrid(df,diag_sharey=False)
g.map_lower(sns.kdeplot,cmap='Blues_d')
g.map_upper(plt.scatter)
g.map_diag(sns.kdeplot,lw=3)
```

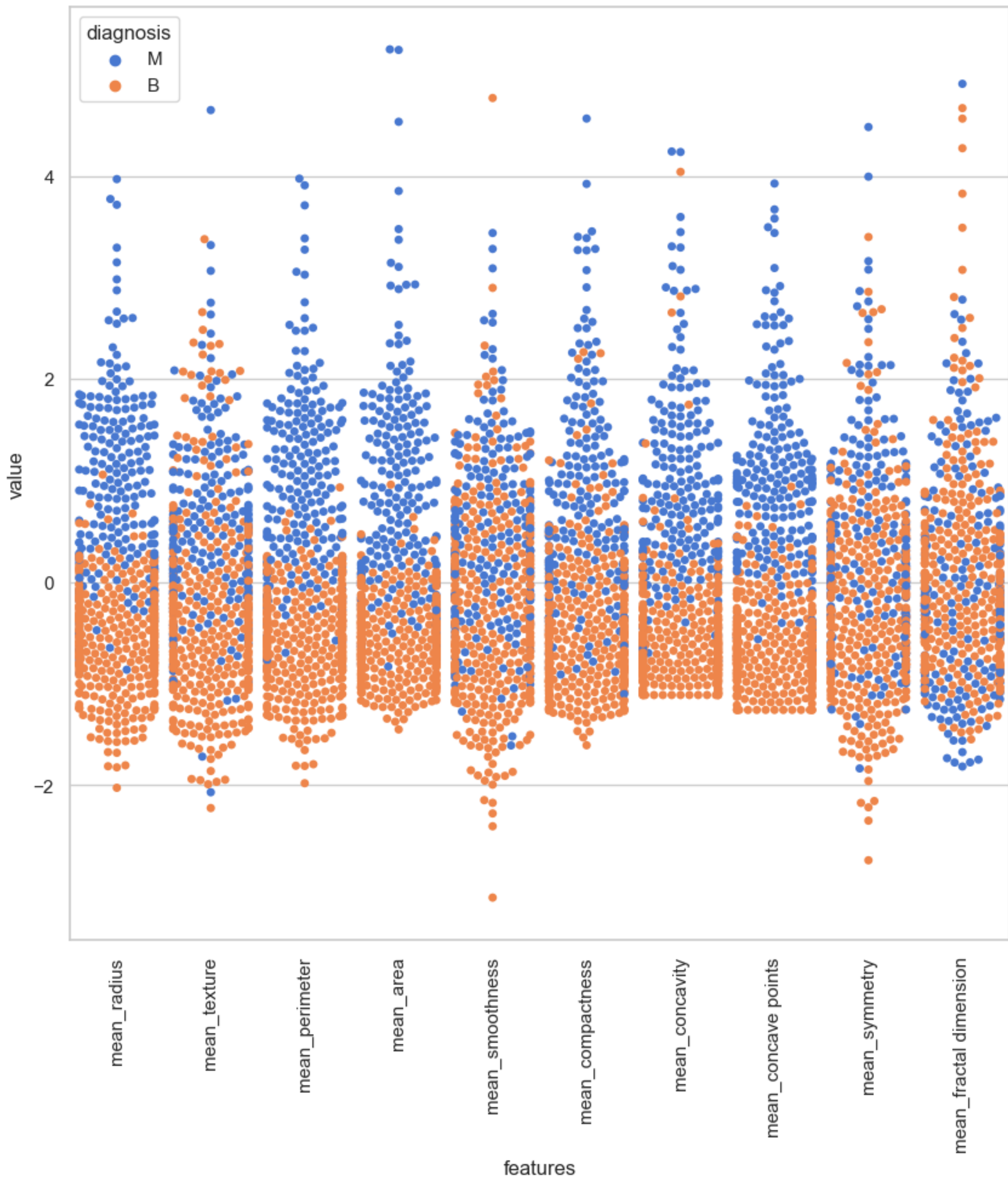
[280]: <seaborn.axisgrid.PairGrid at 0x25871c93b50>



[281]:
```
sns.set(style='whitegrid',palette='muted')
diag=y
data=x
data_n=(data-data.mean())/(data.std())
```

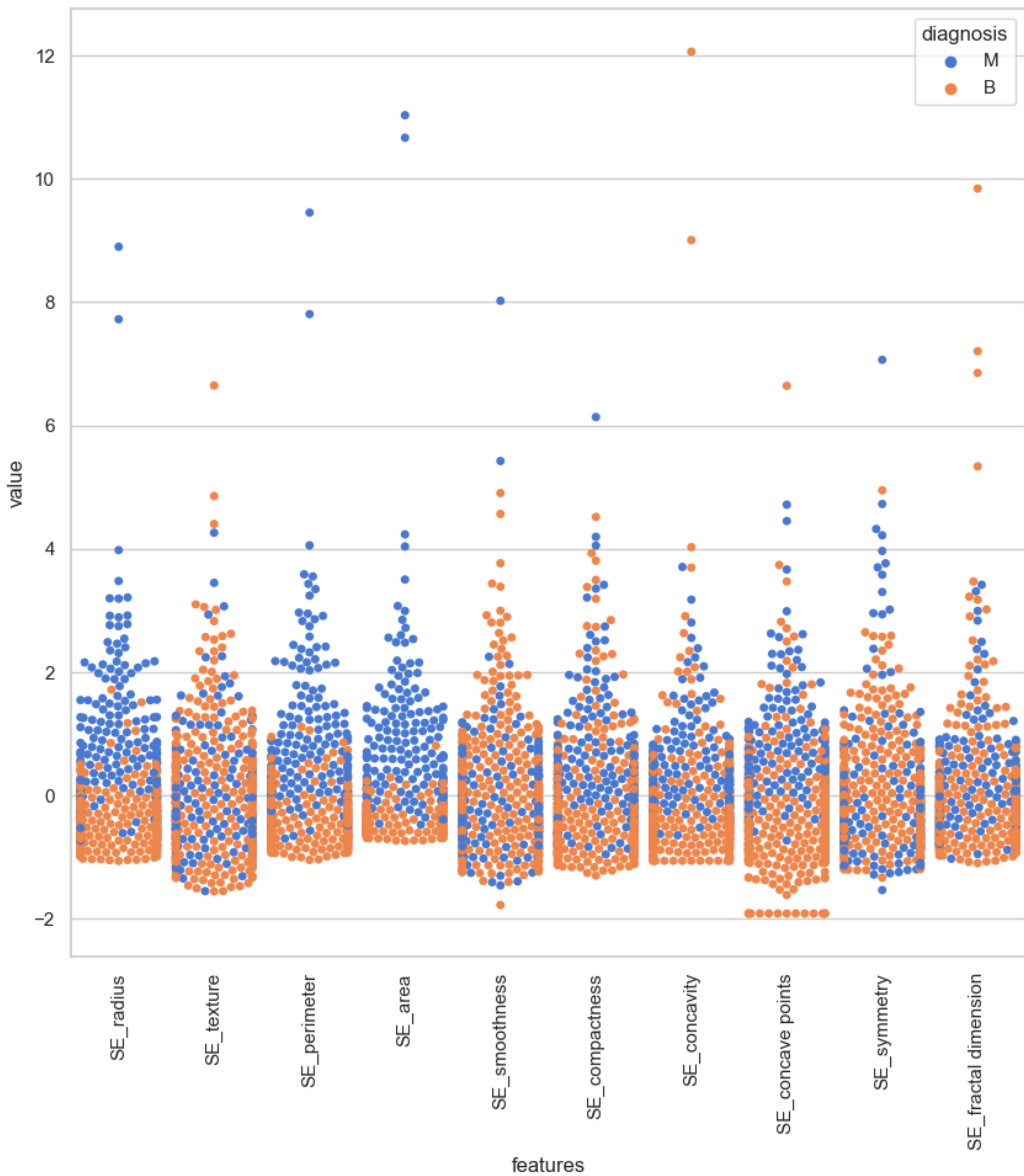
```
data=pd.concat([y,data_n.iloc[:,0:10]],axis=1)
data=pd.melt(data,id_vars='diagnosis',var_name='features',value_name='value')
plt.figure(figsize=(10,10))
tic=time.time()
sns.swarmplot(x='features',y='value',hue='diagnosis',data=data)
plt.xticks(rotation=90)
```

[281]:
```
([0, 1, 2, 3, 4, 5, 6, 7, 8, 9],
 [Text(0, 0, 'mean_radius'),
  Text(1, 0, 'mean_texture'),
  Text(2, 0, 'mean_perimeter'),
  Text(3, 0, 'mean_area'),
  Text(4, 0, 'mean_smoothness'),
  Text(5, 0, 'mean_compactness'),
  Text(6, 0, 'mean_concavity'),
  Text(7, 0, 'mean_concave points'),
  Text(8, 0, 'mean_symmetry'),
  Text(9, 0, 'mean_fractal dimension')])
```



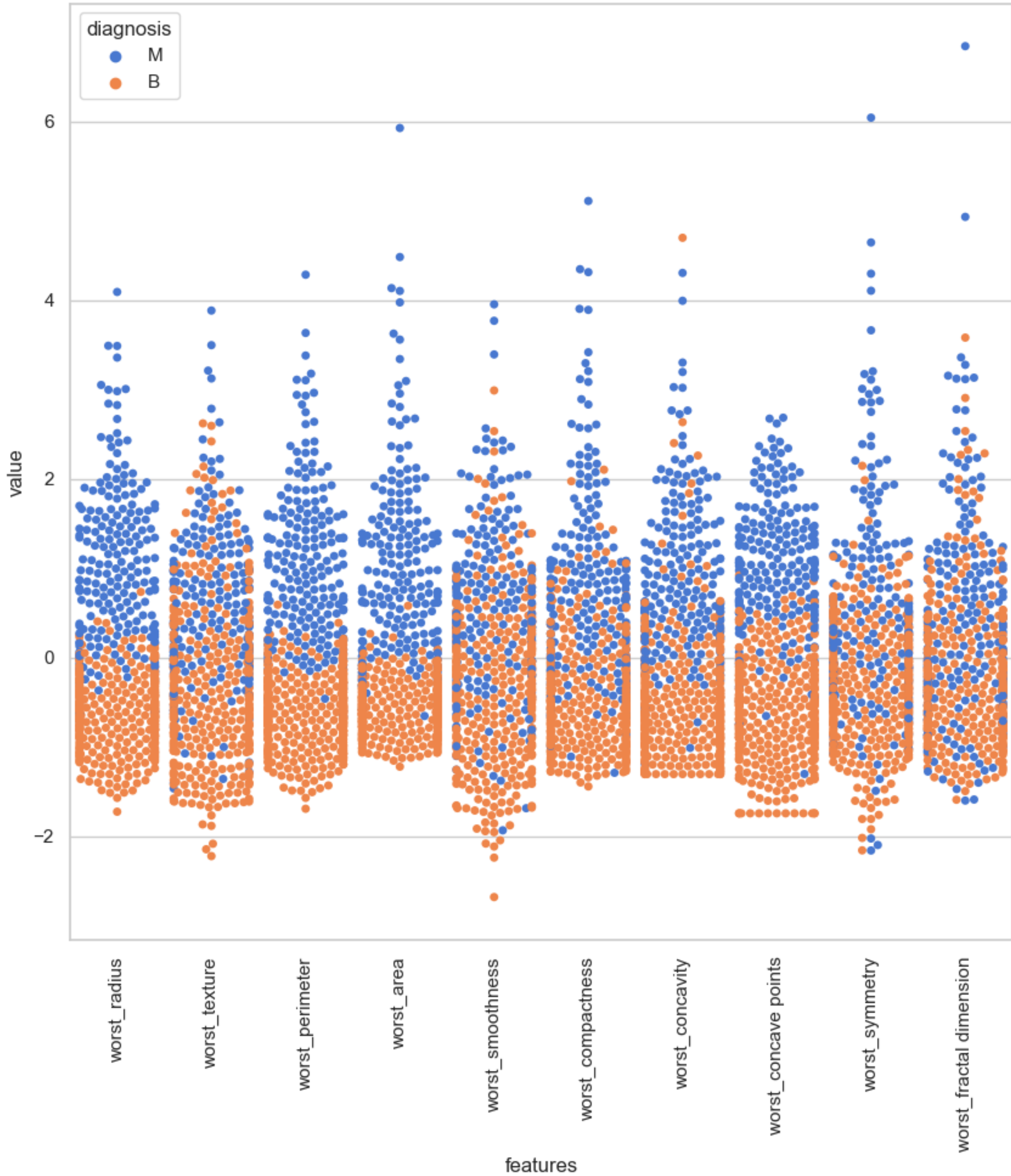
[282]:
```
data=pd.concat([y,data_n.iloc[:,10:20]],axis=1)
data=pd.melt(data,id_vars='diagnosis',var_name='features',value_name='value')
plt.figure(figsize=(10,10))
tic=time.time()
sns.swarmplot(x='features',y='value',hue='diagnosis',data=data)
plt.xticks(rotation=90)
```

```
[282]: ([0, 1, 2, 3, 4, 5, 6, 7, 8, 9],
 [Text(0, 0, 'SE_radius'),
  Text(1, 0, 'SE_texture'),
  Text(2, 0, 'SE_perimeter'),
  Text(3, 0, 'SE_area'),
  Text(4, 0, 'SE_smoothness'),
  Text(5, 0, 'SE_compactness'),
  Text(6, 0, 'SE_concavity'),
  Text(7, 0, 'SE_concave points'),
  Text(8, 0, 'SE_symmetry'),
  Text(9, 0, 'SE_fractal dimension')])
```



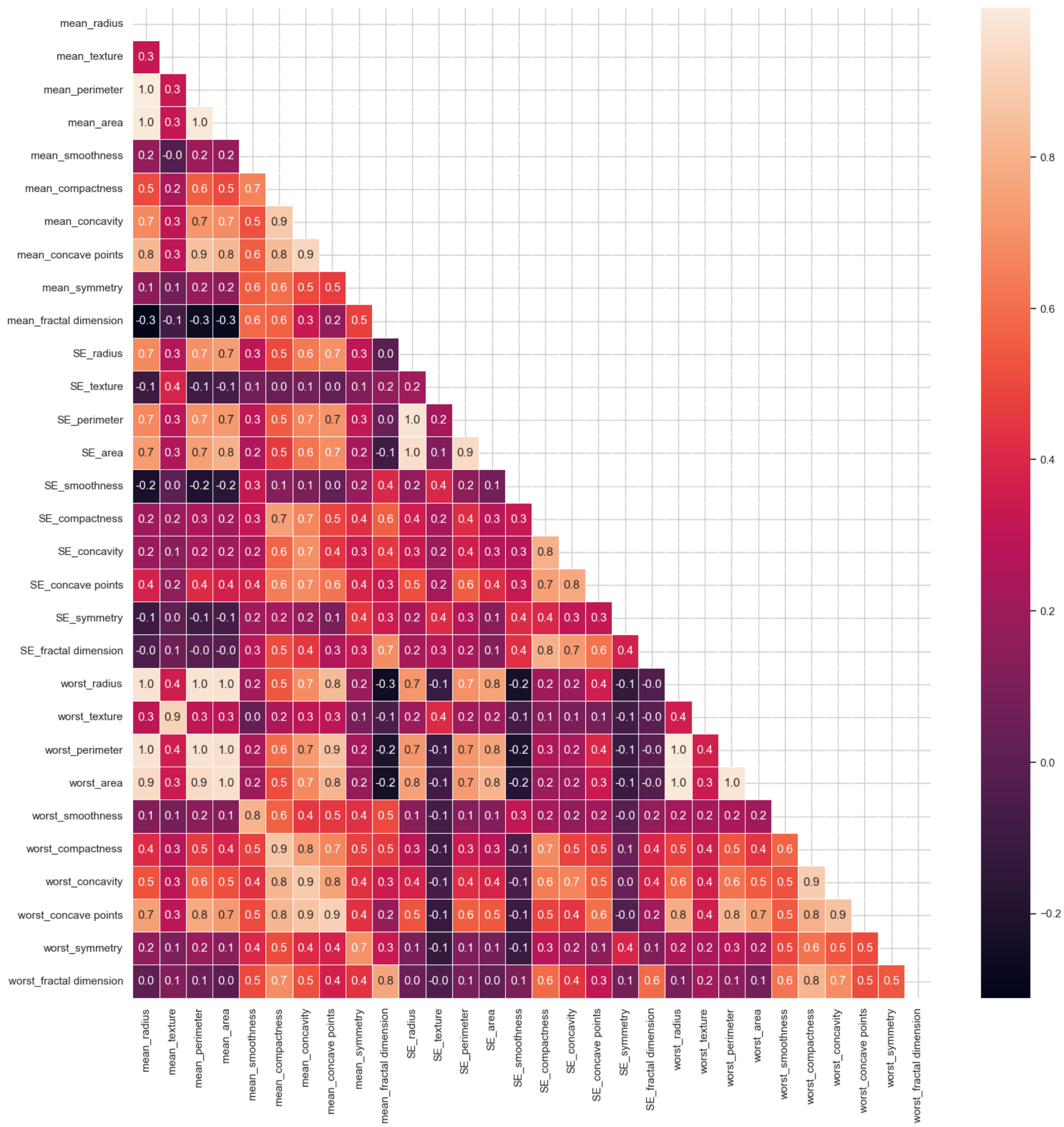
[283]:
```
data=pd.concat([y,data_n.iloc[:,20:31]],axis=1)
data=pd.melt(data,id_vars='diagnosis',var_name='features',value_name='value')
plt.figure(figsize=(10,10))
tic=time.time()
sns.swarmplot(x='features',y='value',hue='diagnosis',data=data)
plt.xticks(rotation=90)
```

[283]:
```
([0, 1, 2, 3, 4, 5, 6, 7, 8, 9],
 [Text(0, 0, 'worst_radius'),
  Text(1, 0, 'worst_texture'),
  Text(2, 0, 'worst_perimeter'),
  Text(3, 0, 'worst_area'),
  Text(4, 0, 'worst_smoothness'),
  Text(5, 0, 'worst_compactness'),
  Text(6, 0, 'worst_concavity'),
  Text(7, 0, 'worst_concave points'),
  Text(8, 0, 'worst_symmetry'),
  Text(9, 0, 'worst_fractal dimension')])
```



[284]:
```
f,ax = plt.subplots(figsize=(18, 18))
matrix = np.triu(x.corr())
sns.heatmap(x.corr(), annot=True, linewidths=.5, fmt= '.1f',ax=ax, mask=matrix)
```

[284]: <Axes: >



```
[285]: # Create correlation matrix
corr_matrix = x.corr().abs()# Select upper triangle of correlation matrix
upper = corr_matrix.where(np.triu(np.ones(corr_matrix.shape), k=1).astype(bool))


# Find index of feature columns with correlation greater than 0.8
to_drop = [column for column in upper.columns if any(upper[column] > 0.8)]
```

```
[286]: to_drop
```

[286]:
```
['mean_perimeter',
 'mean_area',
 'mean_concavity',
 'mean_concave points',
 'SE_perimeter',
 'SE_area',
 'SE_concavity',
 'SE_fractal dimension',
 'worst_radius',
 'worst_texture',
 'worst_perimeter',
 'worst_area',
 'worst_smoothness',
 'worst_compactness',
 'worst_concavity',
 'worst_concave points',
 'worst_fractal dimension']
```

[287]:
```
# Drop features
x1 = x.drop(x[to_drop], axis=1)
x1.columns
```

[287]:
```
Index(['mean_radius', 'mean_texture', 'mean_smoothness', 'mean_compactness',
       'mean_symmetry', 'mean_fractal dimension', 'SE_radius', 'SE_texture',
       'SE_smoothness', 'SE_compactness', 'SE_concave points', 'SE_symmetry',
       'worst_symmetry'],
      dtype='object')
```

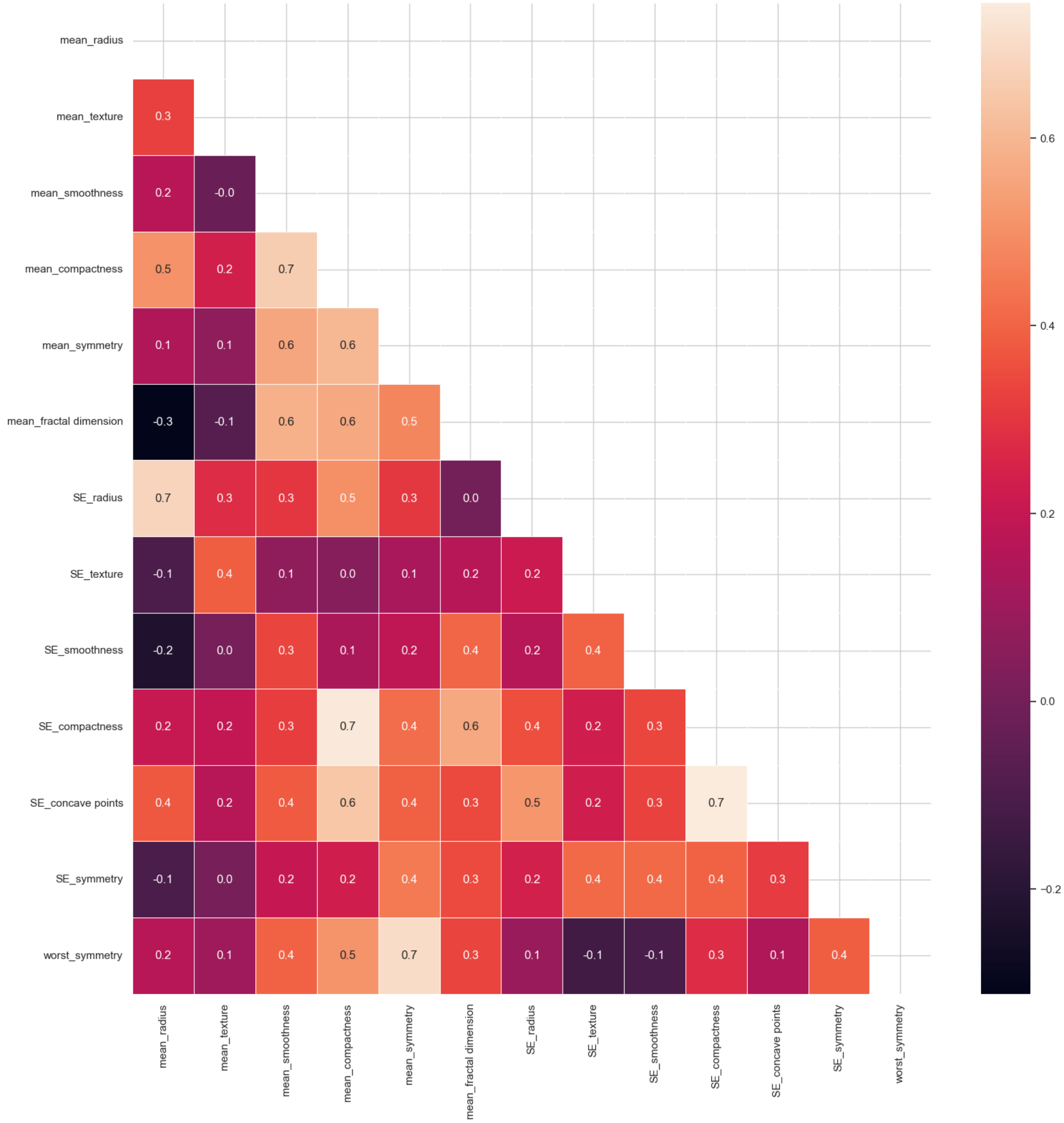
[288]:
```
x1.head()
```

[288]:
```
   mean_radius  mean_texture  mean_smoothness  mean_compactness  \
0        17.99         10.38          0.11840           0.27760
1        20.57         17.77          0.08474           0.07864
2        19.69         21.25          0.10960           0.15990
3        11.42         20.38          0.14250           0.28390
4        20.29         14.34          0.10030           0.13280

   mean_symmetry  mean_fractal dimension  SE_radius  SE_texture  \
0         0.2419                 0.07871     1.0950      0.9053
1         0.1812                 0.05667     0.5435      0.7339
2         0.2069                 0.05999     0.7456      0.7869
3         0.2597                 0.09744     0.4956      1.1560
4         0.1809                 0.05883     0.7572      0.7813

   SE_smoothness  SE_compactness  SE_concave points  SE_symmetry  \
0       0.006399         0.04904            0.01587      0.03003
1       0.005225         0.01308            0.01340      0.01389
```

```
2       0.006150          0.04006            0.02058      0.02250
3       0.009110          0.07458            0.01867      0.05963
4       0.011490          0.02461            0.01885      0.01756

   worst_symmetry
0          0.4601
1          0.2750
2          0.3613
3          0.6638
4          0.2364
```

[289]:
```
f,ax = plt.subplots(figsize=(18, 18))
matrix = np.triu(x1.corr())
sns.heatmap(x1.corr(), annot=True, linewidths=.5, fmt= '.1f',ax=ax, mask=matrix)
```

[289]: <Axes: >



[290]:
```
from sklearn.model_selection import train_test_split
```

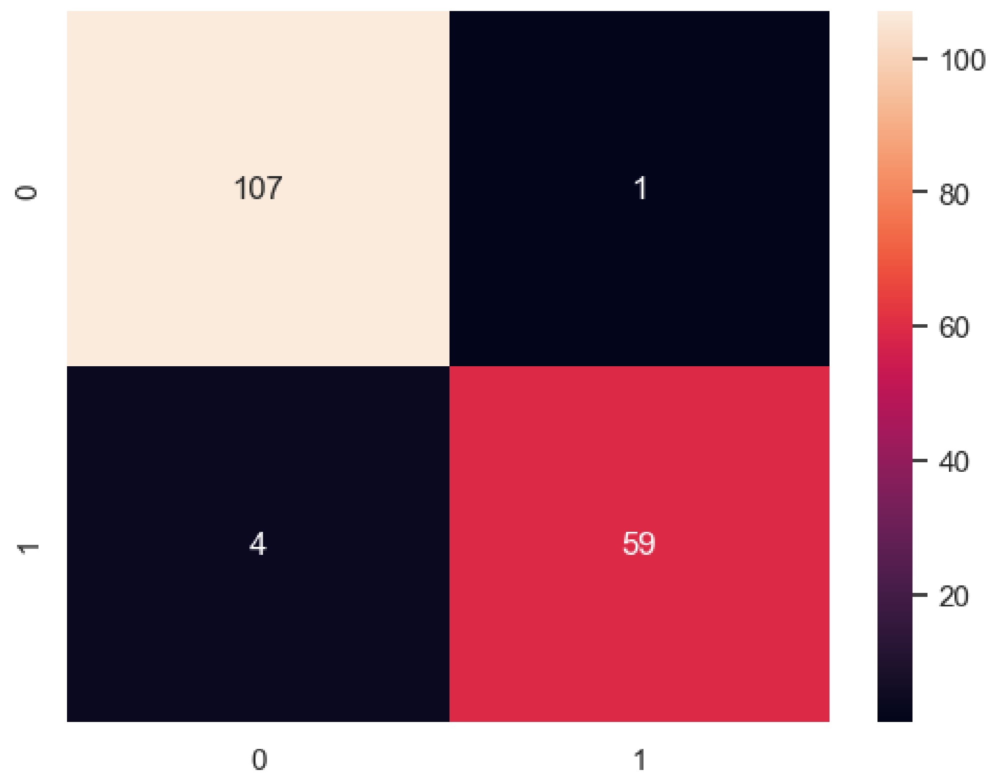
[291]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.
 ↪3,random_state=42)
```

[292]:
```
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import f1_score,confusion_matrix
from sklearn.metrics import accuracy_score
```

```
RFCl=RandomForestClassifier(random_state=42)
RFCl=RFCl.fit(x_train,y_train)
acc=accuracy_score(y_test,RFCl.predict(x_test))
print('Accuracy is:',acc)
cm=confusion_matrix(y_test,RFCl.predict(x_test))
sns.heatmap(cm,annot=True,fmt='d')
```

```
Accuracy is: 0.9707602339181286
```

[292]: <Axes: >



[293]:
```
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import chi2
```

[294]:
```
feature_select=SelectKBest(chi2,k=5).fit(x_train,y_train)
print('Score list:',feature_select.scores_)
print('Feature list:',x_train.columns)
```

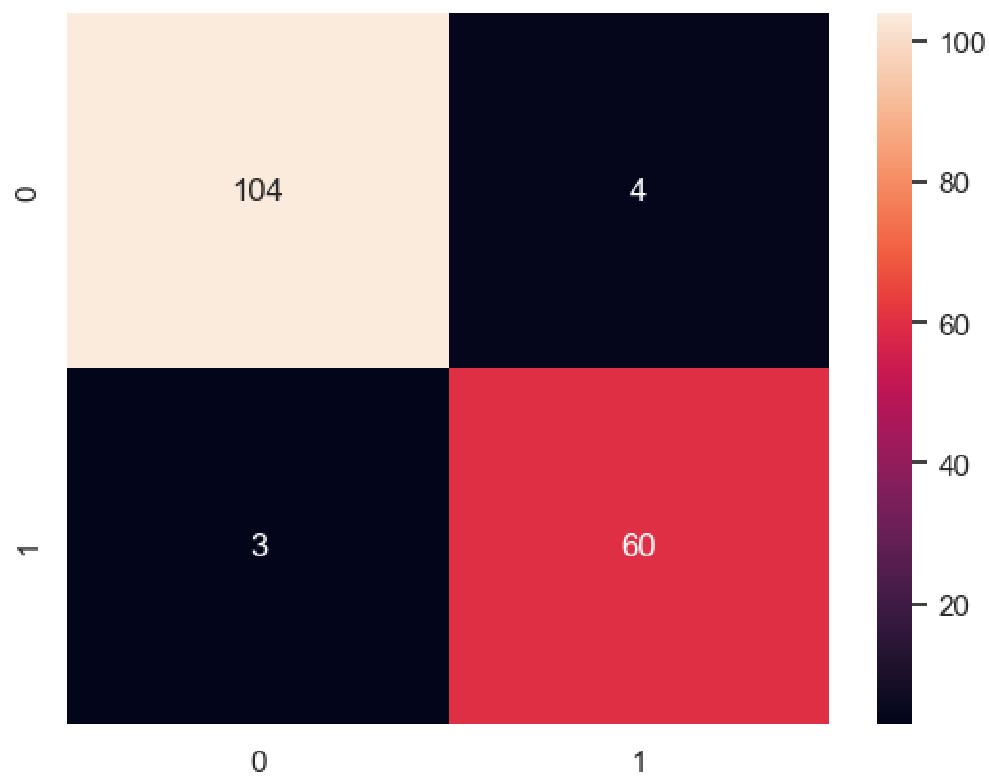
```
Score list: [1.77946492e+02 6.06916433e+01 1.34061092e+03 3.66899557e+04
 1.00015175e-01 3.41839493e+00 1.30547650e+01 7.09766457e+00
 1.95982847e-01 3.42575072e-04 2.45882967e+01 4.07131026e-02
 1.72696840e+02 6.12741067e+03 1.32470372e-03 3.74071521e-01
```

```
 6.92896719e-01 2.01587194e-01 1.39557806e-03 2.65927071e-03
 3.25782599e+02 1.16958562e+02 2.40512835e+03 7.50217341e+04
 2.63226314e-01 1.19077581e+01 2.58858117e+01 8.90751003e+00
 1.00635138e+00 1.23087347e-01]
Feature list: Index(['mean_radius', 'mean_texture', 'mean_perimeter',
'mean_area',
       'mean_smoothness', 'mean_compactness', 'mean_concavity',
       'mean_concave points', 'mean_symmetry', 'mean_fractal dimension',
       'SE_radius', 'SE_texture', 'SE_perimeter', 'SE_area', 'SE_smoothness',
       'SE_compactness', 'SE_concavity', 'SE_concave points', 'SE_symmetry',
       'SE_fractal dimension', 'worst_radius', 'worst_texture',
       'worst_perimeter', 'worst_area', 'worst_smoothness',
       'worst_compactness', 'worst_concavity', 'worst_concave points',
       'worst_symmetry', 'worst_fractal dimension'],
      dtype='object')
```

[295]:
```
x_train_2=feature_select.transform(x_train)
x_test_2=feature_select.transform(x_test)
RFCl2=RandomForestClassifier()
RFCl2.fit(x_train_2,y_train)
acc2=accuracy_score(y_test,RFCl2.predict(x_test_2))
print('Accuracy is:',acc2)
cm2=confusion_matrix(y_test,RFCl2.predict(x_test_2))
sns.heatmap(cm2,annot=True,fmt='d')
```

```
Accuracy is: 0.9590643274853801
```

[295]:
```
<Axes: >
```



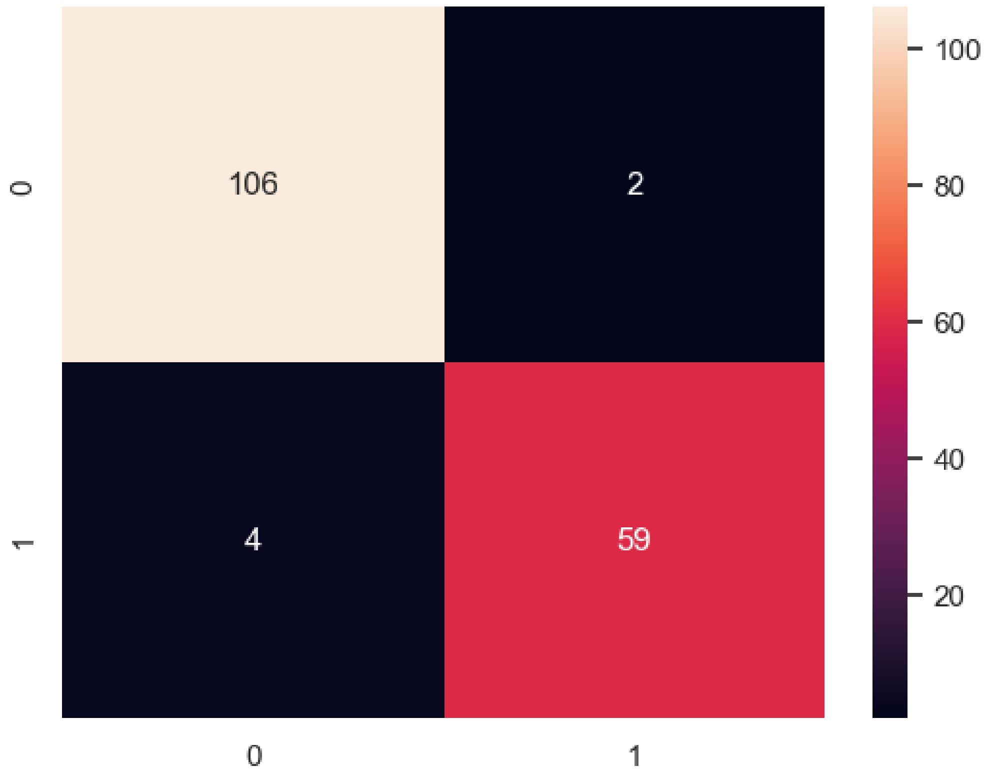
[296]:
```
from sklearn.feature_selection import RFE
RFCl3=RandomForestClassifier()

rfe=RFE(estimator=RFCl3,n_features_to_select=5,step=1)
rfe=rfe.fit(x_train,y_train)
x_train_3=rfe.transform(x_train)
x_test_3=rfe.transform(x_test)
print('Chosen best 5 features by rfe:',x_train.columns[rfe.support_])
```

```
Chosen best 5 features by rfe: Index(['mean_concave points', 'worst_radius',
'worst_perimeter', 'worst_area',
       'worst_concave points'],
      dtype='object')
```

[300]:
```
RFCl3.fit(x_train_3,y_train)
acc3=accuracy_score(y_test,RFCl3.predict(x_test_3))
print('Accuracy is:',acc3)
cm2=confusion_matrix(y_test,RFCl3.predict(x_test_3))
sns.heatmap(cm2,annot=True,fmt='d')
```

```
Accuracy is: 0.9649122807017544
```

[300]: <Axes: >



[301]:
```
from sklearn.feature_selection import RFECV
RFC14=RandomForestClassifier()
rfecv=RFECV(estimator=RFC14,step=1,cv=5,scoring='accuracy')
rfecv=rfecv.fit(x_train,y_train)
print('Optimal number of features:',rfecv.n_features_)
print('Best features:',x_train.columns[rfecv.support_])
```

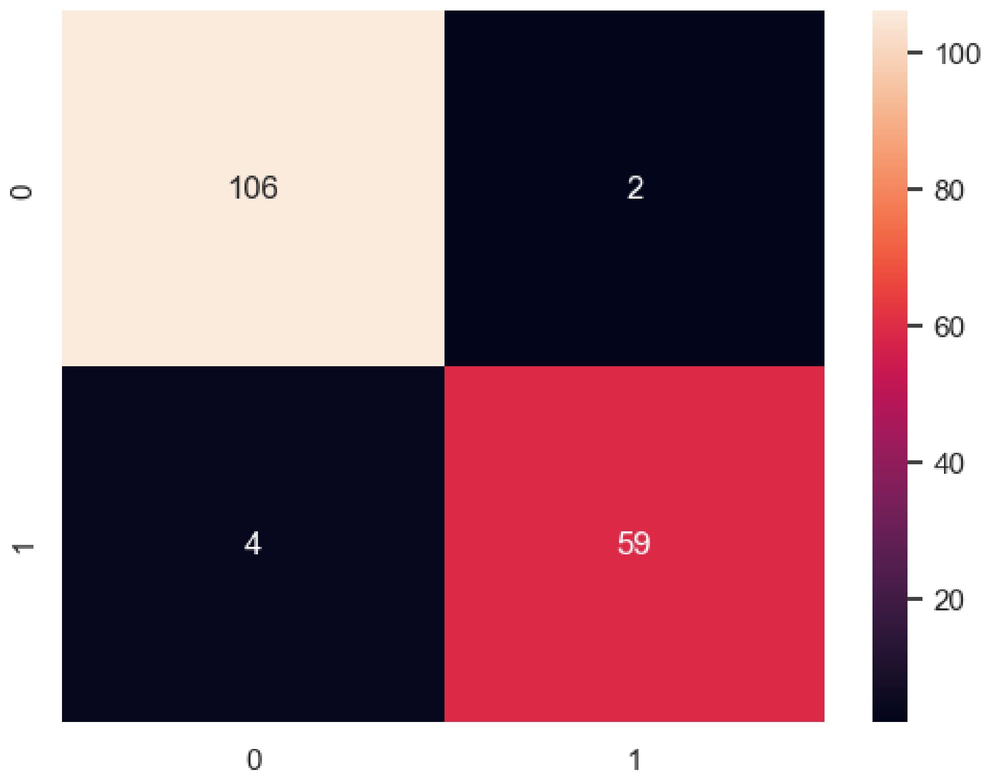
```
Optimal number of features: 13
Best features: Index(['mean_radius', 'mean_texture', 'mean_perimeter',
'mean_area',
       'mean_concavity', 'mean_concave points', 'SE_area', 'worst_radius',
       'worst_texture', 'worst_perimeter', 'worst_area', 'worst_concavity',
       'worst_concave points'],
      dtype='object')
```

[302]:
```
rfecv.fit(x_train,y_train)
x_train_4=rfecv.transform(x_train)
x_test_4=rfecv.transform(x_test)
RFC14.fit(x_train_4,y_train)
```

```
acc4=accuracy_score(y_test,RFC14.predict(x_test_4))
print('Accuracy is:',acc4)
cm2=confusion_matrix(y_test,RFC14.predict(x_test_4))
sns.heatmap(cm2,annot=True,fmt='d')
```

```
Accuracy is: 0.9649122807017544
```

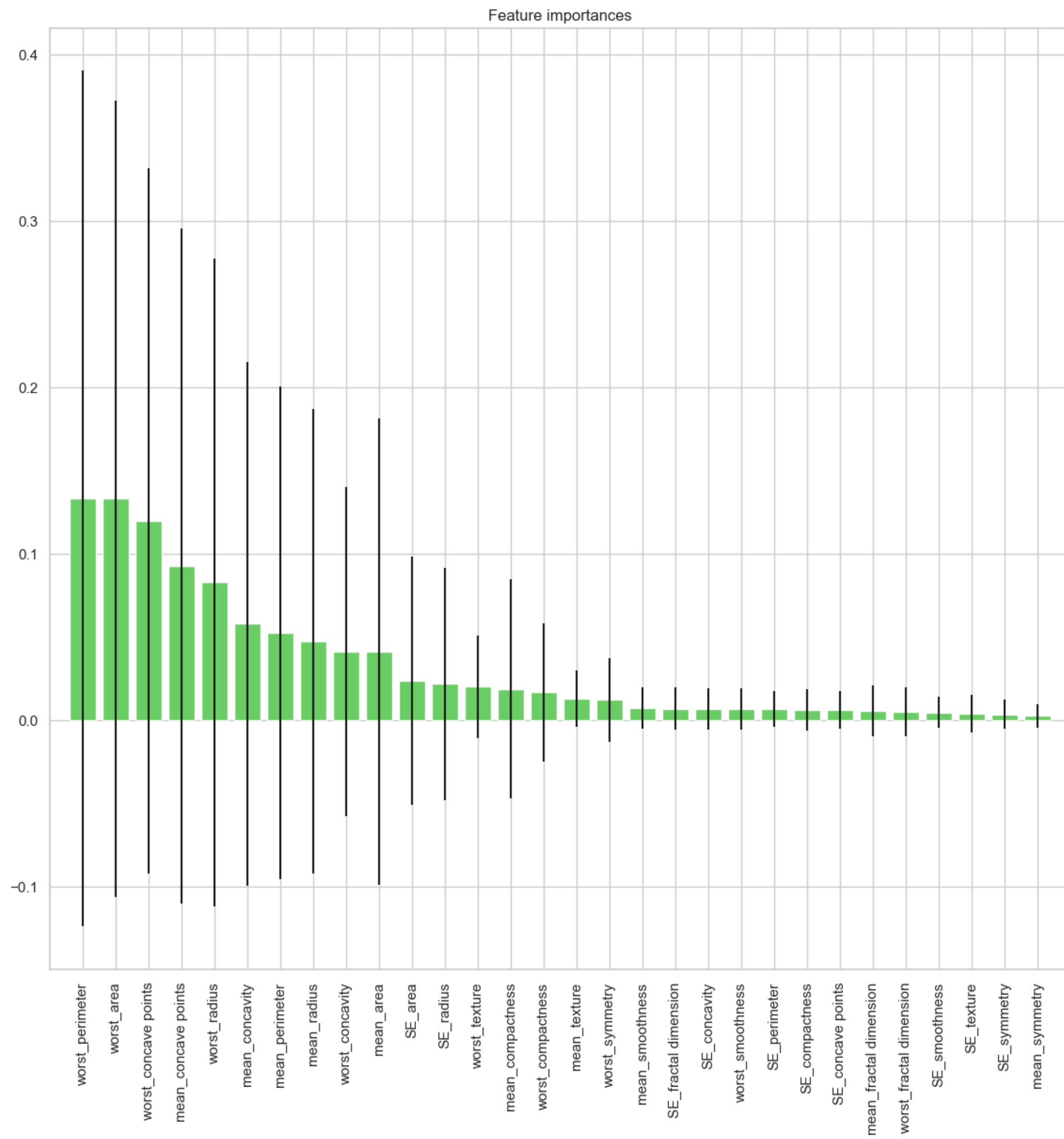
[302]: <Axes: >



[303]:
```
RFC15=RandomForestClassifier()
RFC15=RFC15.fit(x_train,y_train)
importances=RFC15.feature_importances_
std=np.std([tree.feature_importances_ for tree in RFC15.estimators_],axis=0)
indices=np.argsort(importances)[::-1]
print('Feature ranking:')
for f in range(x_train.shape[1]):
    print( '%d. feature %d )=(%f)' % (f+1,indices[f],importances[indices[f]]))
plt.figure(1,figsize=(14,13))
plt.title('Feature importances')
plt.bar(range(x_train.
  ↪shape[1]),importances[indices],color='g',yerr=std[indices],align='center')
plt.xticks(range(x_train.shape[1]),x_train.columns[indices],rotation=90)
```

```
plt.xlim([-1,x_train.shape[1]])
plt.show()
```

```
Feature ranking:
1. feature 22 )=(0.133143)
2. feature 23 )=(0.133068)
3. feature 27 )=(0.119613)
4. feature 7 )=(0.092584)
5. feature 20 )=(0.082885)
6. feature 6 )=(0.057896)
7. feature 2 )=(0.052447)
8. feature 0 )=(0.047453)
9. feature 26 )=(0.041349)
10. feature 3 )=(0.041249)
11. feature 13 )=(0.023812)
12. feature 10 )=(0.021853)
13. feature 21 )=(0.020053)
14. feature 5 )=(0.018697)
15. feature 25 )=(0.016667)
16. feature 1 )=(0.012959)
17. feature 28 )=(0.012210)
18. feature 4 )=(0.007435)
19. feature 19 )=(0.006828)
20. feature 16 )=(0.006770)
21. feature 24 )=(0.006609)
22. feature 12 )=(0.006606)
23. feature 15 )=(0.006181)
24. feature 17 )=(0.006067)
25. feature 9 )=(0.005679)
26. feature 29 )=(0.005058)
27. feature 14 )=(0.004739)
28. feature 11 )=(0.003873)
29. feature 18 )=(0.003526)
30. feature 8 )=(0.002691)
```



[ ]:
```
#Since dimensionality reduction improved efficiency but not importance, we'll
↪assess the results of the Random Forest algorithm using all features.
```