```python
from sklearn.datasets import load_breast_cancer
import seaborn as sns
import pandas as pd

features,target = load_breast_cancer(as_frame=True,return_X_y=True)
print(features.shape)
print(target.shape)
features.describe()
```

```
(569, 30)
(569,)
```

```
       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]

features.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 30 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   mean radius              569 non-null    float64
 1   mean texture             569 non-null    float64
```

```
 2   mean perimeter           569 non-null    float64
 3   mean area                569 non-null    float64
 4   mean smoothness          569 non-null    float64
 5   mean compactness         569 non-null    float64
 6   mean concavity           569 non-null    float64
 7   mean concave points      569 non-null    float64
 8   mean symmetry            569 non-null    float64
 9   mean fractal dimension   569 non-null    float64
 10  radius error             569 non-null    float64
 11  texture error            569 non-null    float64
 12  perimeter error          569 non-null    float64
 13  area error               569 non-null    float64
 14  smoothness error         569 non-null    float64
 15  compactness error        569 non-null    float64
 16  concavity error          569 non-null    float64
 17  concave points error     569 non-null    float64
 18  symmetry error           569 non-null    float64
 19  fractal dimension error  569 non-null    float64
 20  worst radius             569 non-null    float64
 21  worst texture            569 non-null    float64
 22  worst perimeter          569 non-null    float64
 23  worst area               569 non-null    float64
 24  worst smoothness         569 non-null    float64
 25  worst compactness        569 non-null    float64
 26  worst concavity          569 non-null    float64
 27  worst concave points     569 non-null    float64
 28  worst symmetry           569 non-null    float64
 29  worst fractal dimension  569 non-null    float64
dtypes: float64(30)
memory usage: 133.5 KB
```

```
features.head(10)
```

```
   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
5        12.45         15.70           82.57      477.1
0.12780
6        18.25         19.98          119.60     1040.0
0.09463
7        13.71         20.83           90.20      577.9
```

```
0.11890
8        13.00         21.82           87.50      519.8
0.12730
9        12.46         24.04           83.97      475.9
0.11860

   mean compactness  mean concavity  mean concave points  mean
symmetry  \
0           0.27760         0.30010              0.14710
0.2419
1           0.07864         0.08690              0.07017
0.1812
2           0.15990         0.19740              0.12790
0.2069
3           0.28390         0.24140              0.10520
0.2597
4           0.13280         0.19800              0.10430
0.1809
5           0.17000         0.15780              0.08089
0.2087
6           0.10900         0.11270              0.07400
0.1794
7           0.16450         0.09366              0.05985
0.2196
8           0.19320         0.18590              0.09353
0.2350
9           0.23960         0.22730              0.08543
0.2030

   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
5                 0.07613  ...         15.47          23.75
103.40
6                 0.05742  ...         22.88          27.66
153.20
7                 0.07451  ...         17.06          28.14
110.60
8                 0.07389  ...         15.49          30.73
106.20
```

```
9                 0.08243  ...         15.09          40.68
97.65

   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
5       741.6            0.1791             0.5249           0.5355
6      1606.0            0.1442             0.2576           0.3784
7       897.0            0.1654             0.3682           0.2678
8       739.3            0.1703             0.5401           0.5390
9       711.4            0.1853             1.0580           1.1050

   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                0.1741          0.3985                  0.12440
6                0.1932          0.3063                  0.08368
7                0.1556          0.3196                  0.11510
8                0.2060          0.4378                  0.10720
9                0.2210          0.4366                  0.20750

[10 rows x 30 columns]
```

```
target.head(10)
```

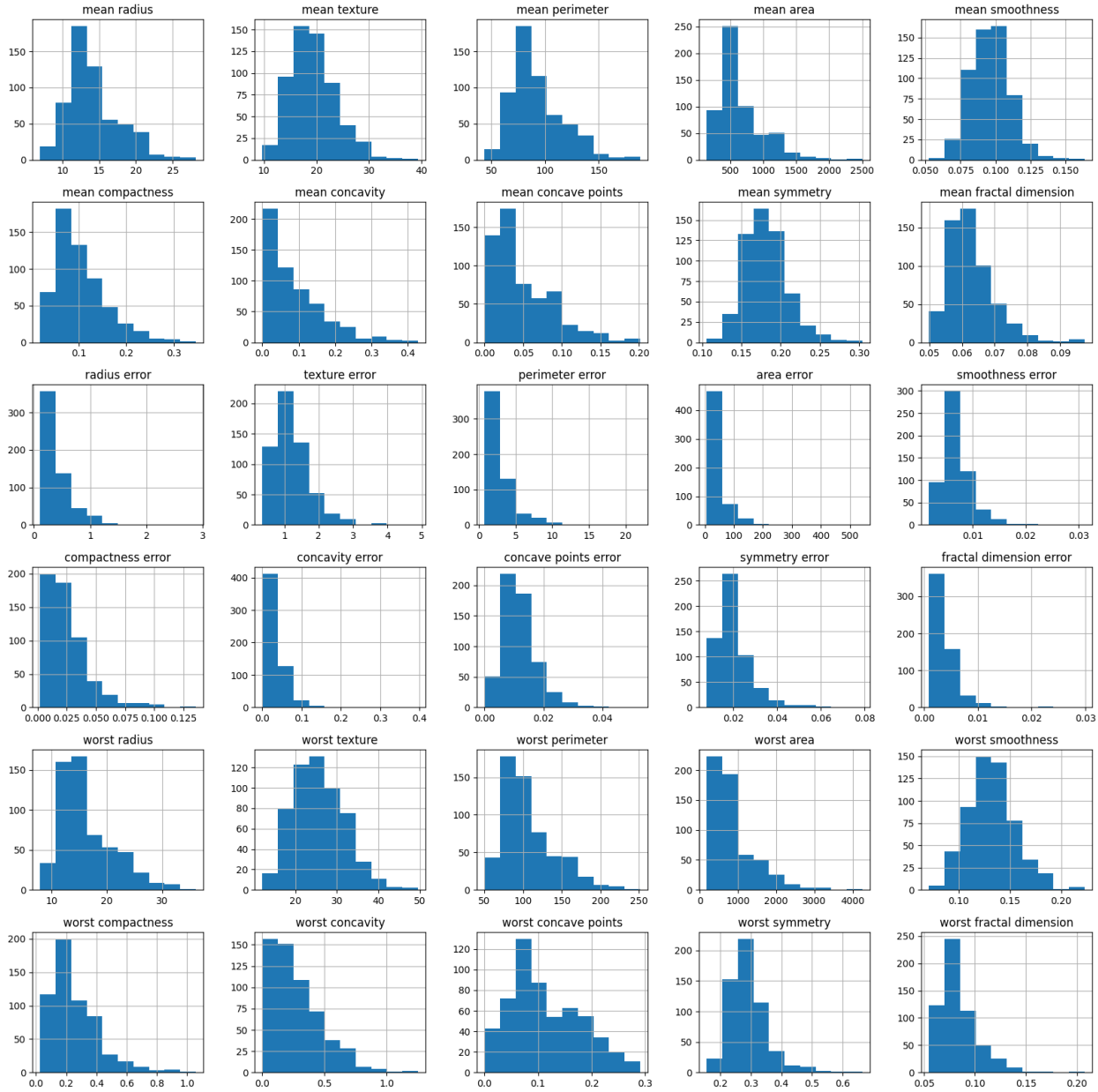
```
0    0
1    0
2    0
3    0
4    0
5    0
6    0
7    0
8    0
9    0
Name: target, dtype: int32
```

EXPLORING THE DATA

```
features.hist(figsize=(20,20))
```

```
array([[<AxesSubplot: title={'center': 'mean radius'}>,
        <AxesSubplot: title={'center': 'mean texture'}>,
        <AxesSubplot: title={'center': 'mean perimeter'}>,
```

```
        <AxesSubplot: title={'center': 'mean area'}>,
        <AxesSubplot: title={'center': 'mean smoothness'}>],
       [<AxesSubplot: title={'center': 'mean compactness'}>,
        <AxesSubplot: title={'center': 'mean concavity'}>,
        <AxesSubplot: title={'center': 'mean concave points'}>,
        <AxesSubplot: title={'center': 'mean symmetry'}>,
        <AxesSubplot: title={'center': 'mean fractal dimension'}>],
       [<AxesSubplot: title={'center': 'radius error'}>,
        <AxesSubplot: title={'center': 'texture error'}>,
        <AxesSubplot: title={'center': 'perimeter error'}>,
        <AxesSubplot: title={'center': 'area error'}>,
        <AxesSubplot: title={'center': 'smoothness error'}>],
       [<AxesSubplot: title={'center': 'compactness error'}>,
        <AxesSubplot: title={'center': 'concavity error'}>,
        <AxesSubplot: title={'center': 'concave points error'}>,
        <AxesSubplot: title={'center': 'symmetry error'}>,
        <AxesSubplot: title={'center': 'fractal dimension error'}>],
       [<AxesSubplot: title={'center': 'worst radius'}>,
        <AxesSubplot: title={'center': 'worst texture'}>,
        <AxesSubplot: title={'center': 'worst perimeter'}>,
        <AxesSubplot: title={'center': 'worst area'}>,
        <AxesSubplot: title={'center': 'worst smoothness'}>],
       [<AxesSubplot: title={'center': 'worst compactness'}>,
        <AxesSubplot: title={'center': 'worst concavity'}>,
        <AxesSubplot: title={'center': 'worst concave points'}>,
        <AxesSubplot: title={'center': 'worst symmetry'}>,
        <AxesSubplot: title={'center': 'worst fractal dimension'}>]],
      dtype=object)
```



DATA CLEANING

```
from sklearn.feature_selection import VarianceThreshold
from sklearn.preprocessing import StandardScaler
print(features.shape)
```

```
(569, 30)
```

```
#removing columns with low variance
transform = VarianceThreshold(threshold=0.01)
transform.fit(features)

#print names of dropped columns
```

```
dropped_columns = features.columns[~transform.get_support()]
print(dropped_columns)
```

```
Index(['mean smoothness', 'mean compactness', 'mean concavity',
       'mean concave points', 'mean symmetry', 'mean fractal
dimension',
       'smoothness error', 'compactness error', 'concavity error',
       'concave points error', 'symmetry error', 'fractal dimension
error',
       'worst smoothness', 'worst concave points', 'worst symmetry',
       'worst fractal dimension'],
      dtype='object')
```

```
features.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 30 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   mean radius              569 non-null    float64
 1   mean texture             569 non-null    float64
 2   mean perimeter           569 non-null    float64
 3   mean area                569 non-null    float64
 4   mean smoothness          569 non-null    float64
 5   mean compactness         569 non-null    float64
 6   mean concavity           569 non-null    float64
 7   mean concave points      569 non-null    float64
 8   mean symmetry            569 non-null    float64
 9   mean fractal dimension   569 non-null    float64
 10  radius error             569 non-null    float64
 11  texture error            569 non-null    float64
 12  perimeter error          569 non-null    float64
 13  area error               569 non-null    float64
 14  smoothness error         569 non-null    float64
 15  compactness error        569 non-null    float64
 16  concavity error          569 non-null    float64
 17  concave points error     569 non-null    float64
 18  symmetry error           569 non-null    float64
 19  fractal dimension error  569 non-null    float64
 20  worst radius             569 non-null    float64
 21  worst texture            569 non-null    float64
 22  worst perimeter          569 non-null    float64
 23  worst area               569 non-null    float64
 24  worst smoothness         569 non-null    float64
 25  worst compactness        569 non-null    float64
 26  worst concavity          569 non-null    float64
 27  worst concave points     569 non-null    float64
 28  worst symmetry           569 non-null    float64
 29  worst fractal dimension  569 non-null    float64
```

```
dtypes: float64(30)
memory usage: 133.5 KB
```

```
processed_features = features.drop(['mean smoothness', 'mean
compactness', 'mean concavity',
       'mean concave points', 'mean symmetry', 'mean fractal
dimension',
       'smoothness error', 'compactness error', 'concavity error',
       'concave points error', 'symmetry error', 'fractal dimension
error',
       'worst smoothness', 'worst concave points', 'worst symmetry',
       'worst fractal dimension'],axis=1)
```

```
processed_features
```

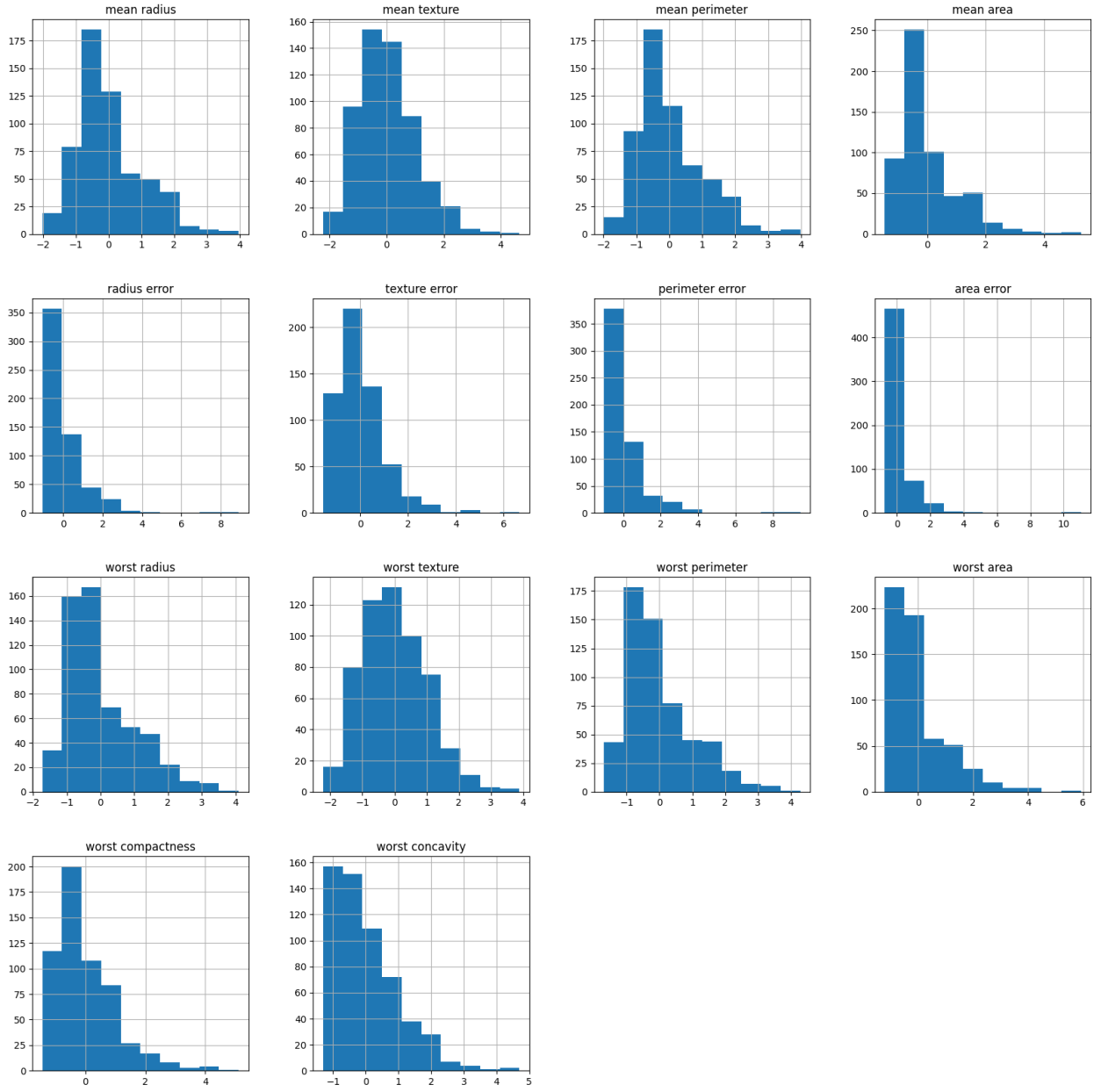
```
     mean radius  mean texture  mean perimeter  mean area  radius
error  \
0          17.99         10.38          122.80     1001.0
1.0950
1          20.57         17.77          132.90     1326.0
0.5435
2          19.69         21.25          130.00     1203.0
0.7456
3          11.42         20.38           77.58      386.1
0.4956
4          20.29         14.34          135.10     1297.0
0.7572
..           ...           ...             ...        ...           ..
.
564        21.56         22.39          142.00     1479.0
1.1760
565        20.13         28.25          131.20     1261.0
0.7655
566        16.60         28.08          108.30      858.1
0.4564
567        20.60         29.33          140.10     1265.0
0.7260
568         7.76         24.54           47.92      181.0
0.3857

     texture error  perimeter error  area error  worst radius  worst
texture  \
0           0.9053            8.589      153.40        25.380
17.33
1           0.7339            3.398       74.08        24.990
23.41
2           0.7869            4.585       94.03        23.570
25.53
3           1.1560            3.445       27.23        14.910
26.50
```

```
4           0.7813             5.438       94.44        22.540
16.67
..             ...               ...         ...           ...
...
564         1.2560             7.673      158.70        25.450
26.40
565         2.4630             5.203       99.04        23.690
38.25
566         1.0750             3.425       48.55        18.980
34.12
567         1.5950             5.772       86.22        25.740
39.42
568         1.4280             2.548       19.15         9.456
30.37

     worst perimeter  worst area  worst compactness  worst concavity
0              184.60      2019.0            0.66560           0.7119
1              158.80      1956.0            0.18660           0.2416
2              152.50      1709.0            0.42450           0.4504
3               98.87       567.7            0.86630           0.6869
4              152.20      1575.0            0.20500           0.4000
..                ...         ...                ...              ...
564            166.10      2027.0            0.21130           0.4107
565            155.00      1731.0            0.19220           0.3215
566            126.70      1124.0            0.30940           0.3403
567            184.60      1821.0            0.86810           0.9387
568             59.16       268.6            0.06444           0.0000

[569 rows x 14 columns]
```

```
#scaling columns
scaler = StandardScaler()
scaler.fit(processed_features)

#keep the same dataframe but scale it
processed_features =
pd.DataFrame(scaler.transform(processed_features),columns=processed_fe
atures.columns)

processed_features.hist(figsize=(20,20))
```

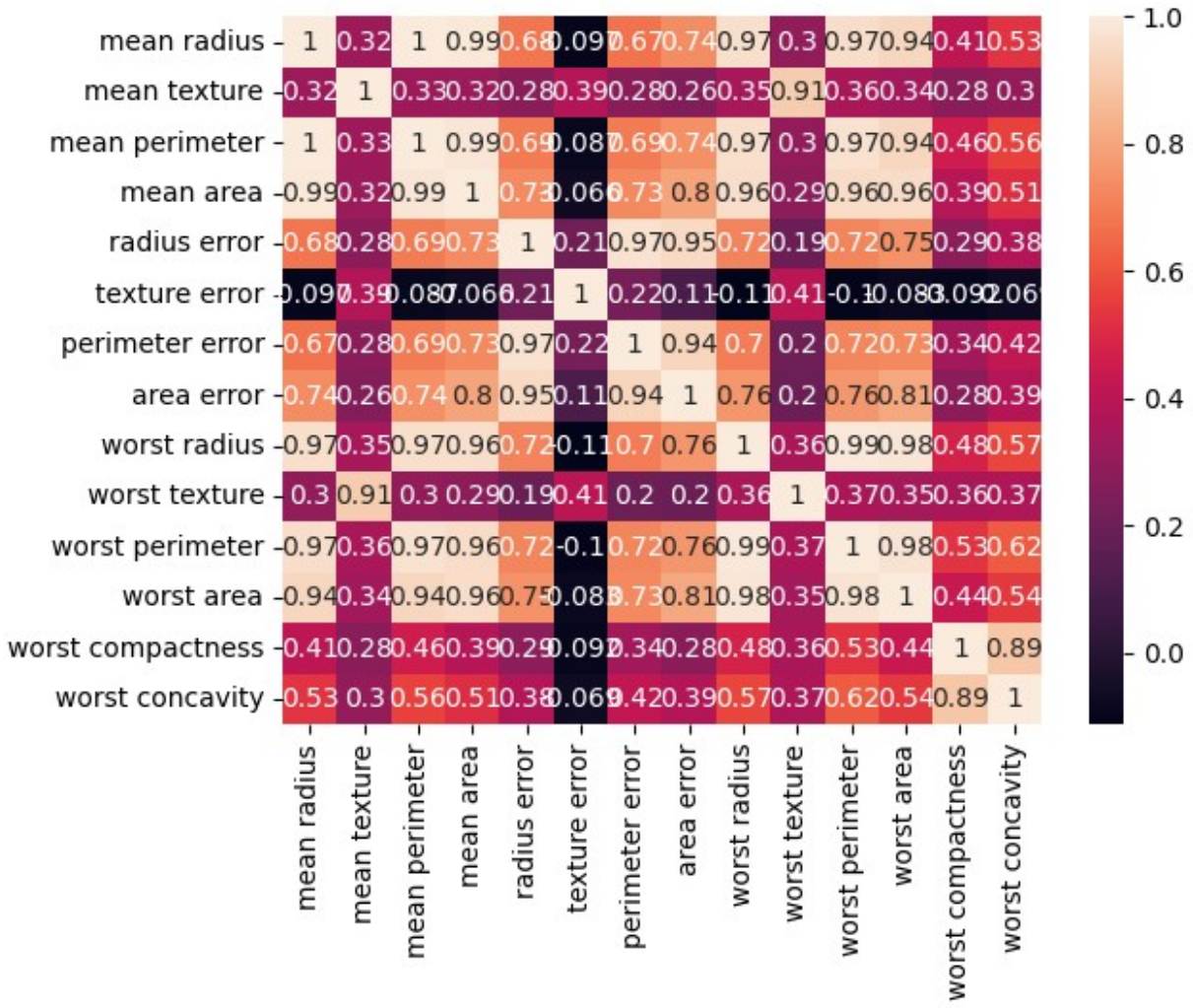
```
array([[<AxesSubplot: title={'center': 'mean radius'}>,
        <AxesSubplot: title={'center': 'mean texture'}>,
        <AxesSubplot: title={'center': 'mean perimeter'}>,
        <AxesSubplot: title={'center': 'mean area'}>],
       [<AxesSubplot: title={'center': 'radius error'}>,
        <AxesSubplot: title={'center': 'texture error'}>,
        <AxesSubplot: title={'center': 'perimeter error'}>,
        <AxesSubplot: title={'center': 'area error'}>],
       [<AxesSubplot: title={'center': 'worst radius'}>,
```

```
       <AxesSubplot: title={'center': 'worst texture'}>,
       <AxesSubplot: title={'center': 'worst perimeter'}>,
       <AxesSubplot: title={'center': 'worst area'}>],
      [<AxesSubplot: title={'center': 'worst compactness'}>,
       <AxesSubplot: title={'center': 'worst concavity'}>,
       <AxesSubplot: >, <AxesSubplot: >]], dtype=object)
```



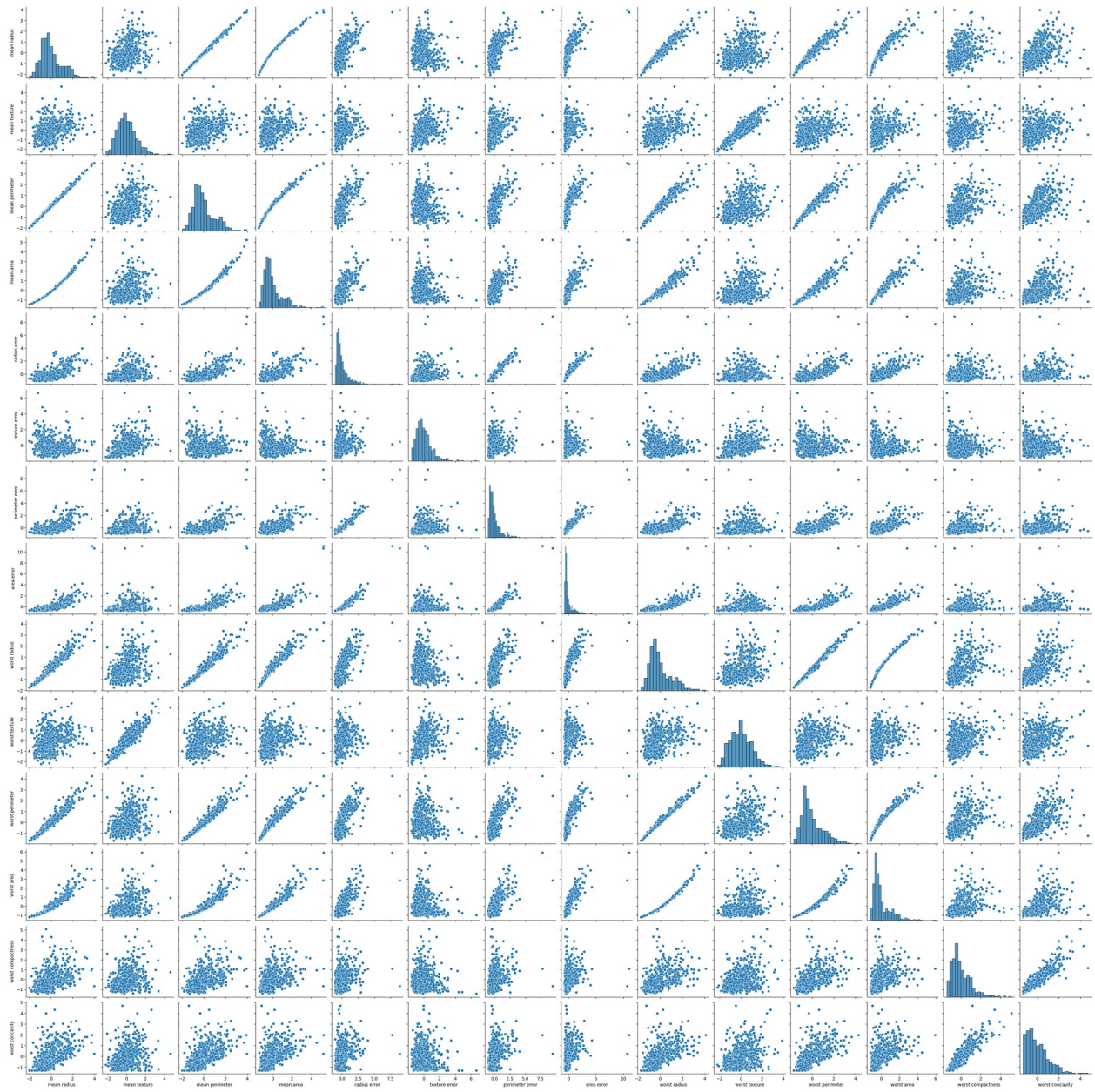

```
sns.heatmap(processed_features.corr(),annot=True)
```

```
<AxesSubplot: >
```




```
sns.pairplot(processed_features)
```

```
<seaborn.axisgrid.PairGrid at 0x1ffbfd93820>
```




```
#use PCA to reduce dimensionality to 2 features
from sklearn.decomposition import PCA
pca = PCA(n_components=2)
pca.fit(processed_features)
reduced_features = pca.transform(processed_features)
reduced_features =
pd.DataFrame(reduced_features,columns=['PC1','PC2'])
print(reduced_features)
```

```
          PC1       PC2
0    5.765261 -2.886527
1    3.669746 -1.991370
```

```
2    4.303308 -0.878462
3    0.261025  1.299997
4    3.740092 -2.948803
..        ...       ...
564  6.379410 -1.023936
565  4.745654  2.340874
566  1.958909  1.603952
567  6.846644  2.725112
568 -3.304229  1.869602

[569 rows x 2 columns]
```

PERCEPTRON CLASSIFICATION

```
from sklearn.linear_model import Perceptron,LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.inspection import DecisionBoundaryDisplay
from sklearn.model_selection import GridSearchCV

x_train,x_test,y_train,y_test =
train_test_split(reduced_features,target,test_size=0.3,random_state=42
)
print(x_train.shape)
print(x_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(398, 2)
(171, 2)
(398,)
(171,)
```

```
perceptron =Perceptron(max_iter=100,random_state=42)
perceptron.fit(x_train,y_train)
y_pred= perceptron.predict(x_test)
print(y_pred.shape)
accuracy =accuracy_score(y_true=y_test,y_pred=y_pred)
print(f"Accuracy: {accuracy * 100:.2f}%")
```

```
(171,)
Accuracy: 90.64%
```

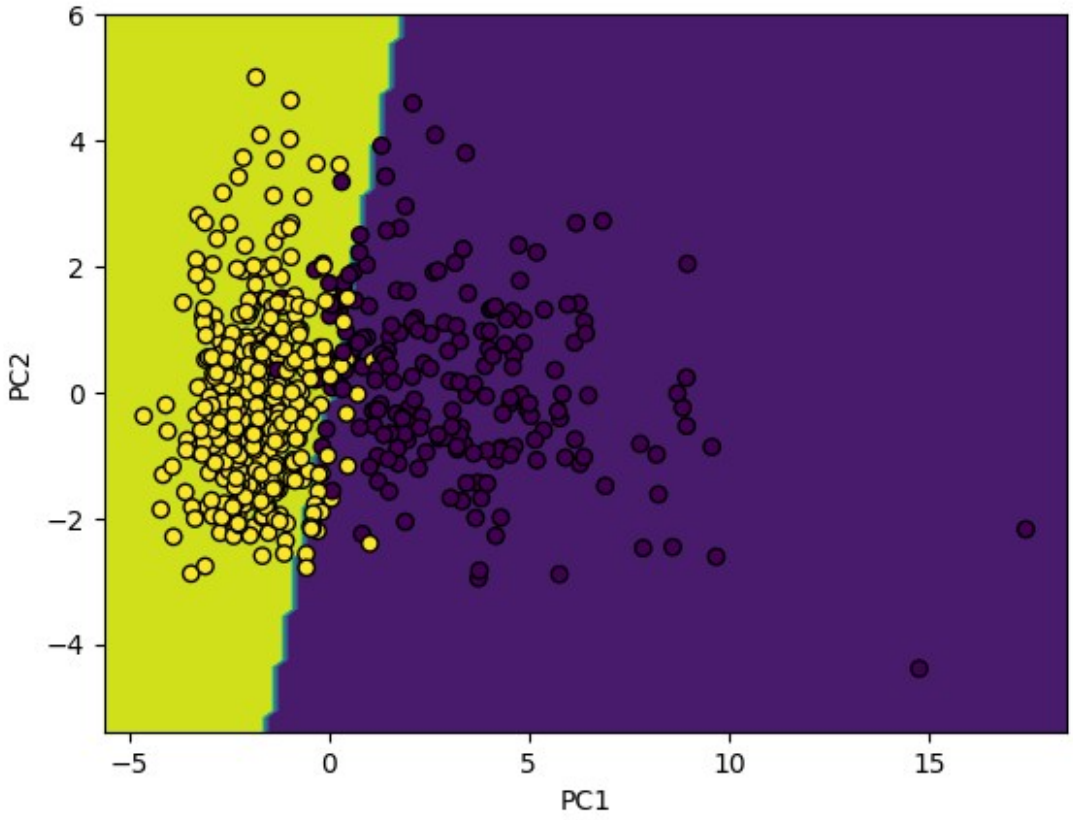
```
#Implementing GridSearchCV for perceptron classifier

parameters = {'max_iter':[1000],'eta0':[0.1,0.01,0.001,0.0001]}
perceptron = Perceptron()
clf = GridSearchCV(perceptron,parameters)
clf.fit(x_train,y_train)
print(clf.best_params_)
```

```
y_pred= clf.predict(x_test)
accuracy =accuracy_score(y_true=y_test,y_pred=y_pred)
print(f"Accuracy: {accuracy * 100:.2f}%")
```

```
{'eta0': 0.1, 'max_iter': 1000}
Accuracy: 95.32%
```

```
plot_dp =
DecisionBoundaryDisplay.from_estimator(estimator=clf,X=reduced_feature
s,response_method='predict')
plot_dp.ax_.scatter(x=reduced_features['PC1'],y=reduced_features['PC2'
],c=target,edgecolors="k")
print("DECISION BOUNDARY FOR PERCEPTRON CLASSIFIER")
```

```
DECISION BOUNDARY FOR PERCEPTRON CLASSIFIER
```



LOGISTIC REGRESSION

```
logisticregr=LogisticRegression()
logisticregr.fit(x_train,y_train)
y_pred=logisticregr.predict(x_test)
accuracy =accuracy_score(y_true=y_test,y_pred=y_pred)
print(f"Accuracy: {accuracy * 100:.2f}%")
```

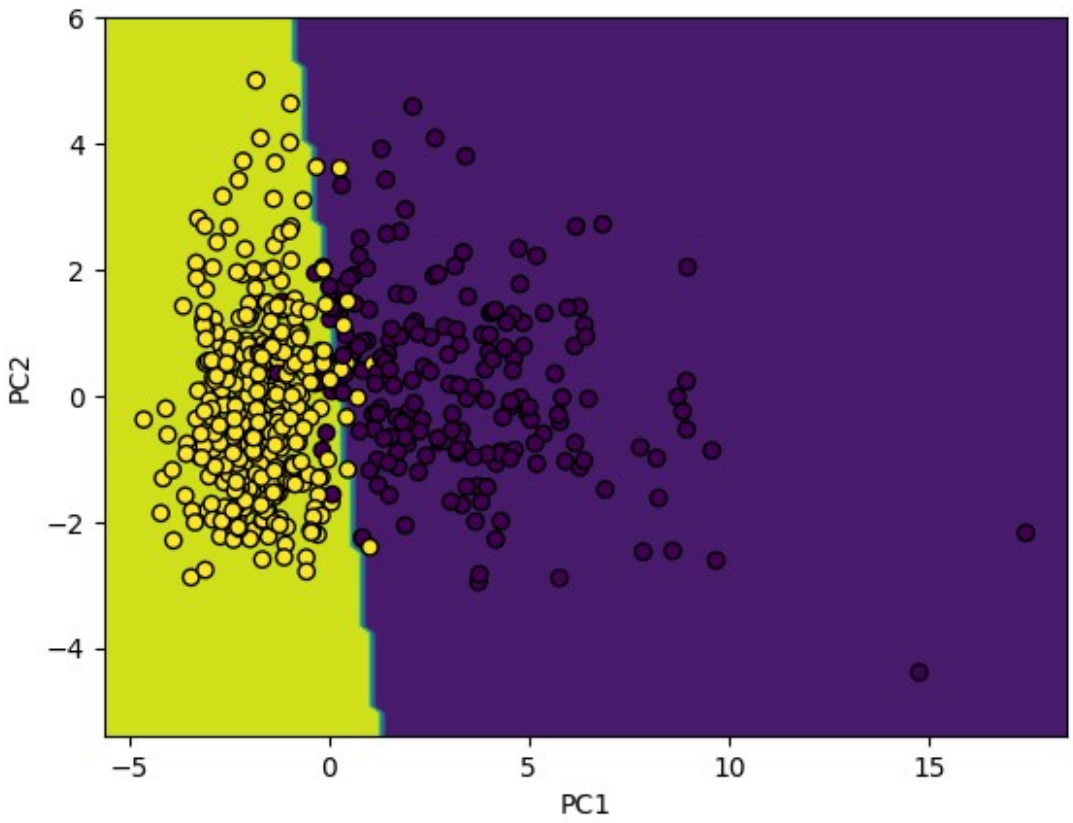
```
Accuracy: 96.49%
```

```
#Grid Search CV for logistic regression
parameters = {'C':[0.1,0.01,0.001,0.0001]}
logisticregr = LogisticRegression()
clf = GridSearchCV(logisticregr,parameters)
clf.fit(x_train,y_train)
print(clf.best_params_)
y_pred= clf.predict(x_test)
accuracy =accuracy_score(y_true=y_test,y_pred=y_pred)
print(f"Accuracy: {accuracy * 100:.2f}%")
```

```
{'C': 0.1}
Accuracy: 97.08%
```

```
plot_dp =
DecisionBoundaryDisplay.from_estimator(estimator=clf,X=reduced_feature
s,response_method='predict')
plot_dp.ax_.scatter(x=reduced_features['PC1'],y=reduced_features['PC2'
],c=target,edgecolors="k")
```

```
<matplotlib.collections.PathCollection at 0x1ffcd2d3c70>
```



SUPPORT VECTOR MACHINE

```
from sklearn.svm import SVC

svm = SVC(kernel="linear",gamma=0.5,C=1.0)
svm.fit(x_train,y_train)
y_pred=svm.predict(x_test)
accuracy =accuracy_score(y_true=y_test,y_pred=y_pred)
print(f"Accuracy: {accuracy * 100:.2f}%")
```

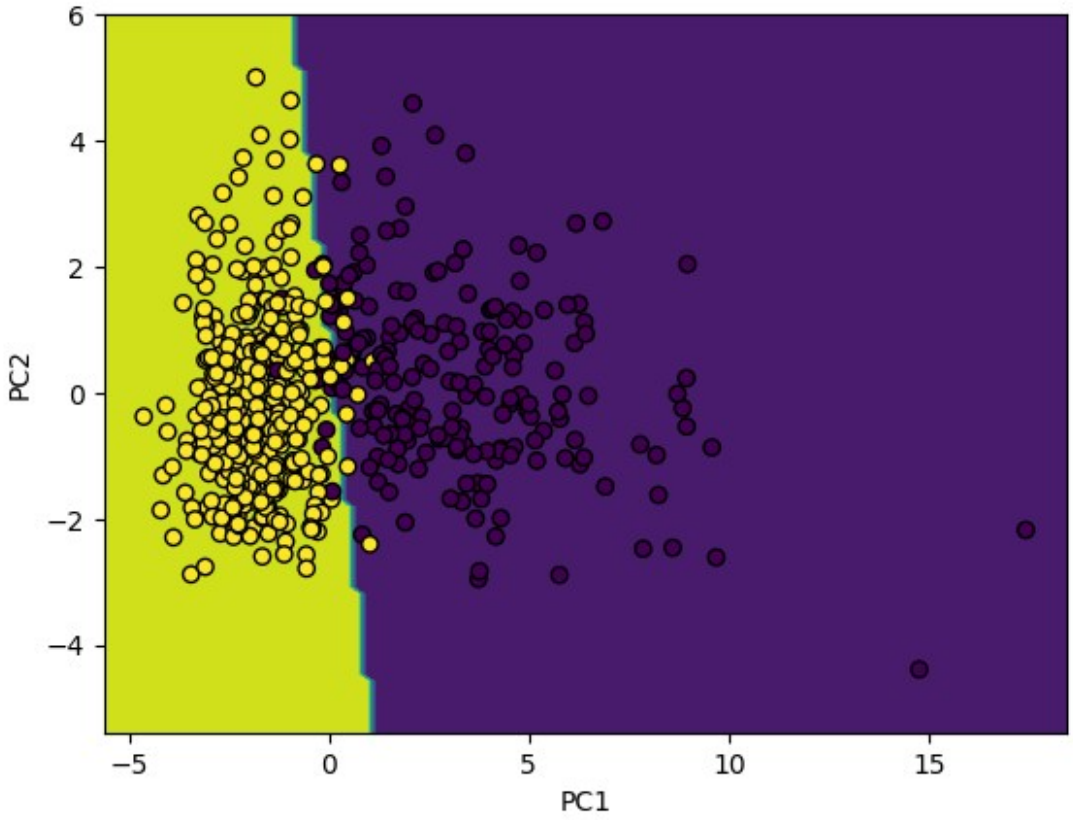
```
Accuracy: 96.49%
```

```
#Grid Search CV for svm
parameters = {'C':[0.1,0.01,0.001,0.0001]}
svm = SVC(kernel="linear",gamma=0.5)
clf = GridSearchCV(svm,parameters)
clf.fit(x_train,y_train)
print(clf.best_params_)
y_pred= clf.predict(x_test)
accuracy =accuracy_score(y_true=y_test,y_pred=y_pred)
print(f"Accuracy: {accuracy * 100:.2f}%")
```

```
{'C': 0.1}
Accuracy: 96.49%
```

```
plot_dp =
DecisionBoundaryDisplay.from_estimator(estimator=clf,X=reduced_feature
s,response_method='predict')
plot_dp.ax_.scatter(x=reduced_features['PC1'],y=reduced_features['PC2'
],c=target,edgecolors="k")
```

```
<matplotlib.collections.PathCollection at 0x1ffbfa55990>
```


6
4
2
0
−2
−4
PC2
−5
0
5
10
15
PC1