In [1]:
```
import pandas as pd
import numpy as np
```

In [3]:
```
from sklearn.datasets import load_breast_cancer
```

Loading of breast cancer data¶

In [6]:
```
data = load_breast_cancer()
data
print(data.feature_names)
```

```
['mean radius' 'mean texture' 'mean perimeter' 'mean area'
 'mean smoothness' 'mean compactness' 'mean concavity'
 'mean concave points' 'mean symmetry' 'mean fractal dimension'
 'radius error' 'texture error' 'perimeter error' 'area error'
 'smoothness error' 'compactness error' 'concavity error'
 'concave points error' 'symmetry error' 'fractal dimension error'
 'worst radius' 'worst texture' 'worst perimeter' 'worst area'
 'worst smoothness' 'worst compactness' 'worst concavity'
 'worst concave points' 'worst symmetry' 'worst fractal dimension']
```

In [8]:
```
print(data.target_names)
```

```
['malignant' 'benign']
```

In [10]:
```
# Convert to DataFrame
df=pd.DataFrame(data.data,columns=data.feature_names)
df
```

Out[10]:

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	r fr dimer
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.30010	0.14710	0.2419	0.0
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.08690	0.07017	0.1812	0.0
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.19740	0.12790	0.2069	0.0
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.24140	0.10520	0.2597	0.0
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.19800	0.10430	0.1809	0.0
...	...	...	...	...	...	...	...	...	...	
564	21.56	22.39	142.00	1479.0	0.11100	0.11590	0.24390	0.13890	0.1726	0.0
565	20.13	28.25	131.20	1261.0	0.09780	0.10340	0.14400	0.09791	0.1752	0.0
566	16.60	28.08	108.30	858.1	0.08455	0.10230	0.09251	0.05302	0.1590	0.0
567	20.60	29.33	140.10	1265.0	0.11780	0.27700	0.35140	0.15200	0.2397	0.0
568	7.76	24.54	47.92	181.0	0.05263	0.04362	0.00000	0.00000	0.1587	0.0

569 rows × 30 columns

In [12]:
```
## Adding the target to the DataFrame(df)
df['target']=data.target
```

```
df
```

Out[12]:

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	r fr dimer
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.30010	0.14710	0.2419	0.0
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.08690	0.07017	0.1812	0.0
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.19740	0.12790	0.2069	0.0
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.24140	0.10520	0.2597	0.0
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.19800	0.10430	0.1809	0.0
...	...	...	...	...	...	...	...	...	...	
564	21.56	22.39	142.00	1479.0	0.11100	0.11590	0.24390	0.13890	0.1726	0.0
565	20.13	28.25	131.20	1261.0	0.09780	0.10340	0.14400	0.09791	0.1752	0.0
566	16.60	28.08	108.30	858.1	0.08455	0.10230	0.09251	0.05302	0.1590	0.0
567	20.60	29.33	140.10	1265.0	0.11780	0.27700	0.35140	0.15200	0.2397	0.0
568	7.76	24.54	47.92	181.0	0.05263	0.04362	0.00000	0.00000	0.1587	0.0

569 rows × 31 columns

In [14]:
```
df.head(5)
```

Out[14]:

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	me fract dimensi
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	0.2419	0.078
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	0.1812	0.056
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	0.2069	0.059
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	0.2597	0.097
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	0.1809	0.058

5 rows × 31 columns

preprocessing of data¶¶

In [17]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 31 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
 30  target                   569 non-null    int32  
dtypes: float64(30), int32(1)
memory usage: 135.7 KB
```

In [19]:
```
df.isnull().sum()
```

Out[19]:
```
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
radius error               0
texture error              0
perimeter error            0
area error                 0
smoothness error           0
compactness error          0
concavity error            0
concave points error       0
symmetry error             0
fractal dimension error    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
target                     0
dtype: int64
```

In [21]:
```
df.duplicated().sum()
```

Out[21]:
```
0
```

In [23]:
```
## There are no missing values and duplicate values in the dataset.
```

In [25]:
```
df.describe()
```

Out[25]:

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave point
count	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000
mean	14.127292	19.289649	91.969033	654.889104	0.096360	0.104341	0.088799	0.048919
std	3.524049	4.301036	24.298981	351.914129	0.014064	0.052813	0.079720	0.038803
min	6.981000	9.710000	43.790000	143.500000	0.052630	0.019380	0.000000	0.000000
25%	11.700000	16.170000	75.170000	420.300000	0.086370	0.064920	0.029560	0.020310
50%	13.370000	18.840000	86.240000	551.100000	0.095870	0.092630	0.061540	0.033500
75%	15.780000	21.800000	104.100000	782.700000	0.105300	0.130400	0.130700	0.074000
max	28.110000	39.280000	188.500000	2501.000000	0.163400	0.345400	0.426800	0.201200

8 rows × 31 columns

In [27]:
```
df=df.select_dtypes(include='number')
df
correlation=df.corr()
correlation
print("\nCorrelation:")
print(correlation)
```

Correlation:

	mean radius	mean texture	mean perimeter	mean area
mean radius	1.000000	0.323782	0.997855	0.987357
mean texture	0.323782	1.000000	0.329533	0.321086
mean perimeter	0.997855	0.329533	1.000000	0.986507
mean area	0.987357	0.321086	0.986507	1.000000
mean smoothness	0.170581	-0.023389	0.207278	0.177028
mean compactness	0.506124	0.236702	0.556936	0.498502
mean concavity	0.676764	0.302418	0.716136	0.685983
mean concave points	0.822529	0.293464	0.850977	0.823269
mean symmetry	0.147741	0.071401	0.183027	0.151293
mean fractal dimension	-0.311631	-0.076437	-0.261477	-0.283110
radius error	0.679090	0.275869	0.691765	0.732562
texture error	-0.097317	0.386358	-0.086761	-0.066280
perimeter error	0.674172	0.281673	0.693135	0.726628
area error	0.735864	0.259845	0.744983	0.800086
smoothness error	-0.222600	0.006614	-0.202694	-0.166777
compactness error	0.206000	0.191975	0.250744	0.212583
concavity error	0.194204	0.143293	0.228082	0.207660
concave points error	0.376169	0.163851	0.407217	0.372320
symmetry error	-0.104321	0.009127	-0.081629	-0.072497
fractal dimension error	-0.042641	0.054458	-0.005523	-0.019887
worst radius	0.969539	0.352573	0.969476	0.962746
worst texture	0.297008	0.912045	0.303038	0.287489
worst perimeter	0.965137	0.358040	0.970387	0.959120
worst area	0.941082	0.343546	0.941550	0.959213
worst smoothness	0.119616	0.077503	0.150549	0.123523
worst compactness	0.413463	0.277830	0.455774	0.390410
worst concavity	0.526911	0.301025	0.563879	0.512606
worst concave points	0.744214	0.295316	0.771241	0.722017
worst symmetry	0.163953	0.105008	0.189115	0.143570
worst fractal dimension	0.007066	0.119205	0.051019	0.003738
target	-0.730029	-0.415185	-0.742636	-0.708984

	mean smoothness	mean compactness	mean concavity
mean radius	0.170581	0.506124	0.676764
mean texture	-0.023389	0.236702	0.302418
mean perimeter	0.207278	0.556936	0.716136
mean area	0.177028	0.498502	0.685983
mean smoothness	1.000000	0.659123	0.521984
mean compactness	0.659123	1.000000	0.883121
mean concavity	0.521984	0.883121	1.000000
mean concave points	0.553695	0.831135	0.921391
mean symmetry	0.557775	0.602641	0.500667
mean fractal dimension	0.584792	0.565369	0.336783
radius error	0.301467	0.497473	0.631925
texture error	0.068406	0.046205	0.076218
perimeter error	0.296092	0.548905	0.660391
area error	0.246552	0.455653	0.617427
smoothness error	0.332375	0.135299	0.098564
compactness error	0.318943	0.738722	0.670279
concavity error	0.248396	0.570517	0.691270
concave points error	0.380676	0.642262	0.683260
symmetry error	0.200774	0.229977	0.178009
fractal dimension error	0.283607	0.507318	0.449301
worst radius	0.213120	0.535315	0.688236
worst texture	0.036072	0.248133	0.299879
worst perimeter	0.238853	0.590210	0.729565
worst area	0.206718	0.509604	0.675987
worst smoothness	0.805324	0.565541	0.448822
worst compactness	0.472468	0.865809	0.754968
worst concavity	0.434926	0.816275	0.884103

```
worst concave points            0.503053          0.815573        0.861323
worst symmetry                  0.394309          0.510223        0.409464
worst fractal dimension         0.499316          0.687382        0.514930
target                         -0.358560         -0.596534       -0.696360

                        mean concave points  mean symmetry  \
mean radius                        0.822529       0.147741
mean texture                       0.293464       0.071401
mean perimeter                     0.850977       0.183027
mean area                          0.823269       0.151293
mean smoothness                    0.553695       0.557775
mean compactness                   0.831135       0.602641
mean concavity                     0.921391       0.500667
mean concave points                1.000000       0.462497
mean symmetry                      0.462497       1.000000
mean fractal dimension             0.166917       0.479921
radius error                       0.698050       0.303379
texture error                      0.021480       0.128053
perimeter error                    0.710650       0.313893
area error                         0.690299       0.223970
smoothness error                   0.027653       0.187321
compactness error                  0.490424       0.421659
concavity error                    0.439167       0.342627
concave points error               0.615634       0.393298
symmetry error                     0.095351       0.449137
fractal dimension error            0.257584       0.331786
worst radius                       0.830318       0.185728
worst texture                      0.292752       0.090651
worst perimeter                    0.855923       0.219169
worst area                         0.809630       0.177193
worst smoothness                   0.452753       0.426675
worst compactness                  0.667454       0.473200
worst concavity                    0.752399       0.433721
worst concave points               0.910155       0.430297
worst symmetry                     0.375744       0.699826
worst fractal dimension            0.368661       0.438413
target                            -0.776614      -0.330499

                        mean fractal dimension  ...  worst texture  \
mean radius                          -0.311631  ...       0.297008
mean texture                         -0.076437  ...       0.912045
mean perimeter                       -0.261477  ...       0.303038
mean area                            -0.283110  ...       0.287489
mean smoothness                       0.584792  ...       0.036072
mean compactness                      0.565369  ...       0.248133
mean concavity                        0.336783  ...       0.299879
mean concave points                   0.166917  ...       0.292752
mean symmetry                         0.479921  ...       0.090651
mean fractal dimension                1.000000  ...      -0.051269
radius error                          0.000111  ...       0.194799
texture error                         0.164174  ...       0.409003
perimeter error                       0.039830  ...       0.200371
area error                           -0.090170  ...       0.196497
smoothness error                      0.401964  ...      -0.074743
compactness error                     0.559837  ...       0.143003
concavity error                       0.446630  ...       0.100241
concave points error                  0.341198  ...       0.086741
symmetry error                        0.345007  ...      -0.077473
fractal dimension error               0.688132  ...      -0.003195
worst radius                         -0.253691  ...       0.359921
worst texture                        -0.051269  ...       1.000000
worst perimeter                      -0.205151  ...       0.365098
```

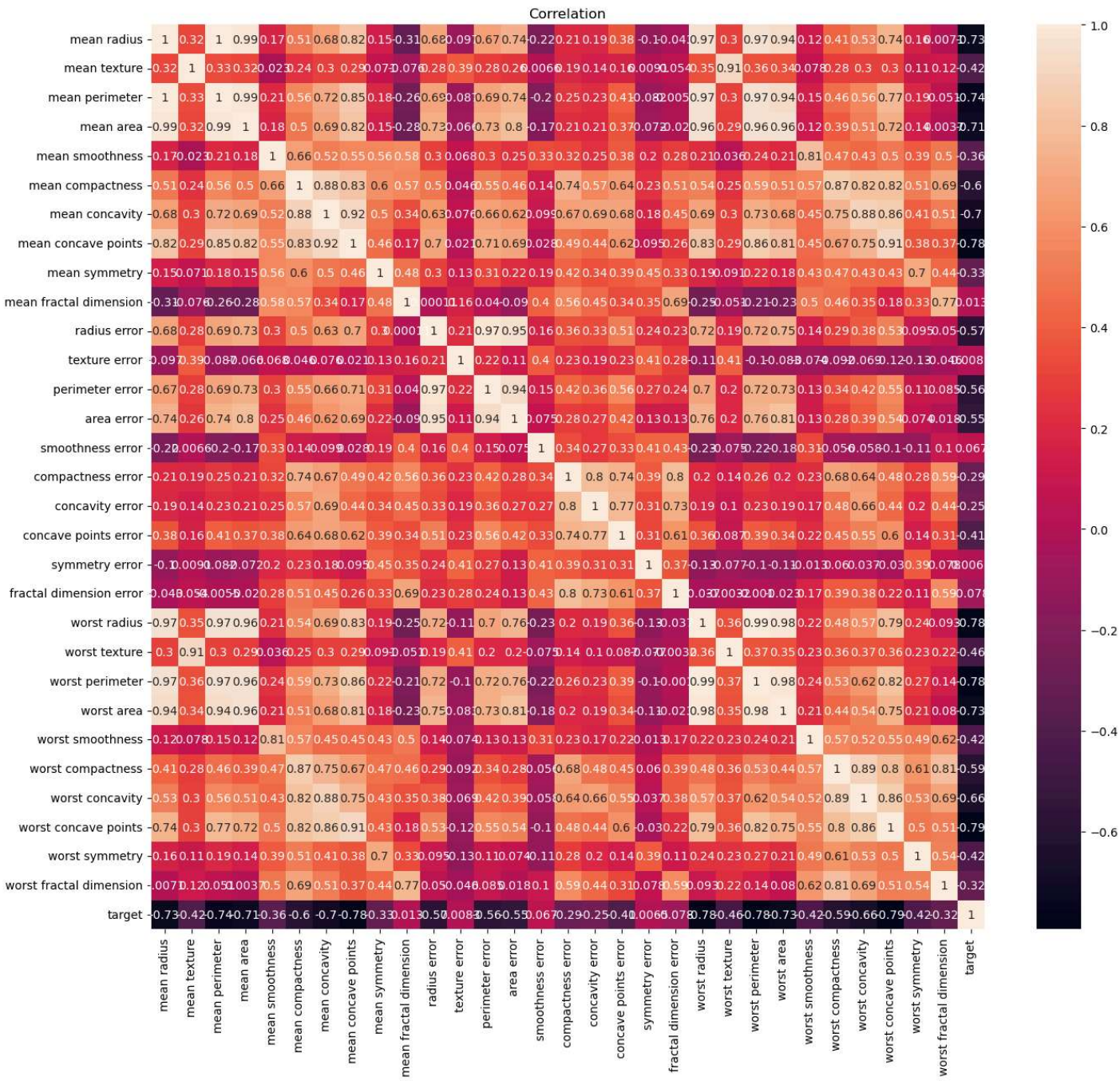
```
worst area                      -0.231854  ...       0.345842
worst smoothness                 0.504942  ...       0.225429
worst compactness                0.458798  ...       0.360832
worst concavity                  0.346234  ...       0.368366
worst concave points             0.175325  ...       0.359755
worst symmetry                   0.334019  ...       0.233027
worst fractal dimension          0.767297  ...       0.219122
target                           0.012838  ...      -0.456903

                        worst perimeter  worst area  worst smoothness  \
mean radius                    0.965137    0.941082          0.119616
mean texture                   0.358040    0.343546          0.077503
mean perimeter                 0.970387    0.941550          0.150549
mean area                      0.959120    0.959213          0.123523
mean smoothness                0.238853    0.206718          0.805324
mean compactness               0.590210    0.509604          0.565541
mean concavity                 0.729565    0.675987          0.448822
mean concave points            0.855923    0.809630          0.452753
mean symmetry                  0.219169    0.177193          0.426675
mean fractal dimension        -0.205151   -0.231854          0.504942
radius error                   0.719684    0.751548          0.141919
texture error                 -0.102242   -0.083195         -0.073658
perimeter error                0.721031    0.730713          0.130054
area error                     0.761213    0.811408          0.125389
smoothness error              -0.217304   -0.182195          0.314457
compactness error              0.260516    0.199371          0.227394
concavity error                0.226680    0.188353          0.168481
concave points error           0.394999    0.342271          0.215351
symmetry error                -0.103753   -0.110343         -0.012662
fractal dimension error       -0.001000   -0.022736          0.170568
worst radius                   0.993708    0.984015          0.216574
worst texture                  0.365098    0.345842          0.225429
worst perimeter                1.000000    0.977578          0.236775
worst area                     0.977578    1.000000          0.209145
worst smoothness               0.236775    0.209145          1.000000
worst compactness              0.529408    0.438296          0.568187
worst concavity                0.618344    0.543331          0.518523
worst concave points           0.816322    0.747419          0.547691
worst symmetry                 0.269493    0.209146          0.493838
worst fractal dimension        0.138957    0.079647          0.617624
target                        -0.782914   -0.733825         -0.421465

                        worst compactness  worst concavity  \
mean radius                      0.413463         0.526911
mean texture                     0.277830         0.301025
mean perimeter                   0.455774         0.563879
mean area                        0.390410         0.512606
mean smoothness                  0.472468         0.434926
mean compactness                 0.865809         0.816275
mean concavity                   0.754968         0.884103
mean concave points              0.667454         0.752399
mean symmetry                    0.473200         0.433721
mean fractal dimension           0.458798         0.346234
radius error                     0.287103         0.380585
texture error                   -0.092439        -0.068956
perimeter error                  0.341919         0.418899
area error                       0.283257         0.385100
smoothness error                -0.055558        -0.058298
compactness error                0.678780         0.639147
concavity error                  0.484858         0.662564
concave points error             0.452888         0.549592
symmetry error                   0.060255         0.037119
```

```
fractal dimension error           0.390159         0.379975
worst radius                      0.475820         0.573975
worst texture                     0.360832         0.368366
worst perimeter                   0.529408         0.618344
worst area                        0.438296         0.543331
worst smoothness                  0.568187         0.518523
worst compactness                 1.000000         0.892261
worst concavity                   0.892261         1.000000
worst concave points              0.801080         0.855434
worst symmetry                    0.614441         0.532520
worst fractal dimension           0.810455         0.686511
target                           -0.590998        -0.659610

                         worst concave points  worst symmetry  \
mean radius                          0.744214        0.163953
mean texture                         0.295316        0.105008
mean perimeter                       0.771241        0.189115
mean area                            0.722017        0.143570
mean smoothness                      0.503053        0.394309
mean compactness                     0.815573        0.510223
mean concavity                       0.861323        0.409464
mean concave points                  0.910155        0.375744
mean symmetry                        0.430297        0.699826
mean fractal dimension               0.175325        0.334019
radius error                         0.531062        0.094543
texture error                       -0.119638       -0.128215
perimeter error                      0.554897        0.109930
area error                           0.538166        0.074126
smoothness error                    -0.102007       -0.107342
compactness error                    0.483208        0.277878
concavity error                      0.440472        0.197788
concave points error                 0.602450        0.143116
symmetry error                      -0.030413        0.389402
fractal dimension error              0.215204        0.111094
worst radius                         0.787424        0.243529
worst texture                        0.359755        0.233027
worst perimeter                      0.816322        0.269493
worst area                           0.747419        0.209146
worst smoothness                     0.547691        0.493838
worst compactness                    0.801080        0.614441
worst concavity                      0.855434        0.532520
worst concave points                 1.000000        0.502528
worst symmetry                       0.502528        1.000000
worst fractal dimension              0.511114        0.537848
target                              -0.793566       -0.416294

                         worst fractal dimension    target
mean radius                             0.007066 -0.730029
mean texture                            0.119205 -0.415185
mean perimeter                          0.051019 -0.742636
mean area                               0.003738 -0.708984
mean smoothness                         0.499316 -0.358560
mean compactness                        0.687382 -0.596534
mean concavity                          0.514930 -0.696360
mean concave points                     0.368661 -0.776614
mean symmetry                           0.438413 -0.330499
mean fractal dimension                  0.767297  0.012838
radius error                            0.049559 -0.567134
texture error                          -0.045655  0.008303
perimeter error                         0.085433 -0.556141
area error                              0.017539 -0.548236
smoothness error                        0.101480  0.067016
```

```
compactness error                0.590973 -0.292999
concavity error                  0.439329 -0.253730
concave points error             0.310655 -0.408042
symmetry error                   0.078079  0.006522
fractal dimension error          0.591328 -0.077972
worst radius                     0.093492 -0.776454
worst texture                    0.219122 -0.456903
worst perimeter                  0.138957 -0.782914
worst area                       0.079647 -0.733825
worst smoothness                 0.617624 -0.421465
worst compactness                0.810455 -0.590998
worst concavity                  0.686511 -0.659610
worst concave points             0.511114 -0.793566
worst symmetry                   0.537848 -0.416294
worst fractal dimension          1.000000 -0.323872
target                          -0.323872  1.000000

[31 rows x 31 columns]
```

In [29]:
```
import seaborn as sns
import matplotlib.pyplot as plt
```

In [31]:
```
# Visualize the correlation matrix
plt.figure(figsize=(16,14))
sns.heatmap(correlation,annot=True)
plt.title("Correlation")
plt.show()
```



In [33]:
```
## Correlation is analyzed to understand relationships between features.
##High correlation between features might indicate redundancy and affect some models (like Logis
```

In [35]:
```
from sklearn.model_selection import train_test_split
```

In [37]:
```
## features and target
X = df.drop("target", axis=1)
y = df["target"]
```

In [39]:
```
from sklearn.preprocessing import StandardScaler
```

In [41]:
```
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
X_scaled
```

Out[41]:
```
array([[ 1.09706398, -2.07333501,  1.26993369, ...,  2.29607613,
         2.75062224,  1.93701461],
       [ 1.82982061, -0.35363241,  1.68595471, ...,  1.0870843 ,
        -0.24388967,  0.28118999],
       [ 1.57988811,  0.45618695,  1.56650313, ...,  1.95500035,
         1.152255  ,  0.20139121],
       ...,
       [ 0.70228425,  2.0455738 ,  0.67267578, ...,  0.41406869,
        -1.10454895, -0.31840916],
       [ 1.83834103,  2.33645719,  1.98252415, ...,  2.28998549,
         1.91908301,  2.21963528],
       [-1.80840125,  1.22179204, -1.81438851, ..., -1.74506282,
        -0.04813821, -0.75120669]])
```

In [43]:
```
# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)
X_train, X_test, y_train, y_test
```

Out[43]:
```
(array([[-1.44798723, -0.45602336, -1.36665103, ...,  0.91959172,
          2.14719008,  1.85943247],
        [ 1.97750799,  1.69418666,  2.08961867, ...,  2.6752184 ,
          1.9368786 ,  2.46346488],
        [-1.40708919, -1.26351565, -1.34976305, ..., -0.97048581,
          0.61676962,  0.05287682],
        ...,
        [ 0.04621146, -0.57470379, -0.06874782, ..., -1.23756033,
         -0.71628161, -1.26047806],
        [-0.04183295,  0.07687501, -0.03497186, ...,  1.03683652,
          0.45013821,  1.19444266],
        [-0.5530585 ,  0.28631105, -0.60751564, ..., -0.61357437,
         -0.33448538, -0.84042616]]),
 array([[-0.47069438, -0.16048584, -0.44810956, ..., -0.19956318,
          0.18320441,  0.19695794],
        [ 1.36687747,  0.47014935,  1.30288585, ...,  0.97897545,
         -0.56582801, -1.00057787],
        [ 0.37850807,  0.04429607,  0.40082046, ...,  0.56024403,
         -0.10314275, -0.20813168],
        ...,
        [-0.74050787, -1.01451947, -0.74550281, ..., -0.28209134,
         -0.3830188 , -0.3245049 ],
        [ 0.02633046,  1.99205126,  0.02393013, ..., -0.49358878,
         -1.63518099, -0.33170895],
        [ 1.87526288,  2.75300221,  1.80128727, ...,  0.78102969,
         -0.05299156, -0.0978542 ]]),
 68     1
 181    0
 63     1
 248    1
 60     1
       ..
 71     1
 106    1
 270    1
 435    0
 102    1
 Name: target, Length: 455, dtype: int32,
 204    1
 70     0
 131    0
 431    1
 540    1
       ..
 486    1
 75     0
 249    1
 238    1
 265    0
 Name: target, Length: 114, dtype: int32)
```

In [104…

```python
# Function to train and evaluate models
def train_and_evaluate_model(model, model_name):
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    accuracy = accuracy_score(y_test, y_pred)
    print(f"{model_name} Accuracy: {accuracy:.2f}")
    print(f"{model_name} Classification Report:\n{classification_report(y_test, y_pred)}\n")
    print(f"{model_name} Confusion Matrix:\n{confusion_matrix(y_test, y_pred)}\n")
    return y_pred
```

In [106…
```
# Initialize results dictionary
results = {
    "Model": [],
    "Accuracy": [],
    "Precision": [],
    "Recall": [],
    "F1_Score": []
}
```

LogisticRegression

In [108…
```
from sklearn.linear_model import LogisticRegression
```

In [110…
```
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, classificat
```

In [112…
```
# Logistic Regression
log_reg = LogisticRegression()
log_reg
y_pred_log_reg = train_and_evaluate_model(log_reg, "Logistic Regression")
y_pred_log_reg
```

```
Logistic Regression Accuracy: 0.97
Logistic Regression Classification Report:
              precision    recall  f1-score   support

           0       0.98      0.95      0.96        43
           1       0.97      0.99      0.98        71

    accuracy                           0.97       114
   macro avg       0.97      0.97      0.97       114
weighted avg       0.97      0.97      0.97       114


Logistic Regression Confusion Matrix:
[[41  2]
 [ 1 70]]
```

Out[112…
```
array([1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1,
       0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1,
       1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 1,
       0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0,
       1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 1, 0, 1, 1,
       0, 1, 0, 0])
```

In [142…
```
## Logistic Regression is classification model.It works well for binary & multiclass classificat
```

DecisionTreeClassifier

In [115…
```
from sklearn.tree import DecisionTreeClassifier
```

In [117…
```
# Decision Tree Classifier
dt = DecisionTreeClassifier()
dt
y_pred_dt = train_and_evaluate_model(dt, "Decision Tree")
y_pred_dt
```

```
Decision Tree Accuracy: 0.94
Decision Tree Classification Report:
              precision    recall  f1-score   support

           0       0.91      0.93      0.92        43
           1       0.96      0.94      0.95        71

    accuracy                           0.94       114
   macro avg       0.93      0.94      0.93       114
weighted avg       0.94      0.94      0.94       114


Decision Tree Confusion Matrix:
[[40  3]
 [ 4 67]]
```

Out[117…

```
array([1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1,
       0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1,
       1, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 1,
       0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0,
       1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 1,
       0, 1, 1, 0])
```

In [144…

```
#Decision Tree splits data based on feature thresholds to make decisions. It handles non-linear r
```

RandomForestClassifier

In [120…

```
from sklearn.ensemble import RandomForestClassifier
```

In [122…

```
rf = RandomForestClassifier()
rf
y_pred_rf= train_and_evaluate_model(rf, "Random Forest")
y_pred_rf
```

```
Random Forest Accuracy: 0.96
Random Forest Classification Report:
              precision    recall  f1-score   support

           0       0.98      0.93      0.95        43
           1       0.96      0.99      0.97        71

    accuracy                           0.96       114
   macro avg       0.97      0.96      0.96       114
weighted avg       0.97      0.96      0.96       114


Random Forest Confusion Matrix:
[[40  3]
 [ 1 70]]
```

Out[122…

```
array([1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1,
       0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1,
       1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 1,
       0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0,
       1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 1, 0, 1, 1,
       0, 1, 1, 0])
```

Support Vector Machine (SVM)

In [125…
```
from sklearn.svm import SVC
```

In [127…
```
svm = SVC()
svm
y_pred_svm = train_and_evaluate_model(svm, "SVM")
y_pred_svm
```

```
SVM Accuracy: 0.97
SVM Classification Report:
              precision    recall  f1-score   support

           0       0.98      0.95      0.96        43
           1       0.97      0.99      0.98        71

    accuracy                           0.97       114
   macro avg       0.97      0.97      0.97       114
weighted avg       0.97      0.97      0.97       114


SVM Confusion Matrix:
[[41  2]
 [ 1 70]]
```

Out[127…
```
array([1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1,
       0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1,
       1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 1,
       0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0,
       1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 1, 0, 1, 1,
       0, 1, 1, 0])
```

KNeighborsClassifier

In [130…
```
from sklearn.neighbors import KNeighborsClassifier
```

In [132…
```
knn = KNeighborsClassifier()
knn
y_pred_knn = train_and_evaluate_model(knn, "k-NN")
y_pred_knn
```

```
k-NN Accuracy: 0.95
k-NN Classification Report:
              precision    recall  f1-score   support

           0       0.93      0.93      0.93        43
           1       0.96      0.96      0.96        71

    accuracy                           0.95       114
   macro avg       0.94      0.94      0.94       114
weighted avg       0.95      0.95      0.95       114


k-NN Confusion Matrix:
[[40  3]
 [ 3 68]]
```

Out[132…
```
array([1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1,
       0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1,
       1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 1,
       0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0,
       1, 1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 1, 0, 1, 1,
       0, 1, 1, 0])
```

In [134…
```
# Step 3: Model Evaluation and Comparison
predictions = {
    "Logistic Regression": y_pred_log_reg,
    "Decision Tree": y_pred_dt,
    "Random Forest": y_pred_rf,
    "SVM": y_pred_svm,
    "k-NN": y_pred_knn
}
```

In [136…
```
for model_name, y_pred in predictions.items():
    results["Model"].append(model_name)
    results["Accuracy"].append(accuracy_score(y_test, y_pred))
    results["Precision"].append(precision_score(y_test, y_pred))
    results["Recall"].append(recall_score(y_test, y_pred))
    results["F1_Score"].append(f1_score(y_test, y_pred))
```

In [138…
```
# Convert results to a DataFrame for better visualization
results_df = pd.DataFrame(results)
print("\nModel Evaluation Results:")
print(results_df)
```

```
Model Evaluation Results:
                 Model  Accuracy  Precision    Recall  F1_Score
0  Logistic Regression  0.973684   0.972222  0.985915  0.979021
1        Decision Tree  0.938596   0.957143  0.943662  0.950355
2        Random Forest  0.964912   0.958904  0.985915  0.972222
3                  SVM  0.973684   0.972222  0.985915  0.979021
4                 k-NN  0.947368   0.957746  0.957746  0.957746
```

In [140…
```
## algorithm performed the best and which one performed the worst?
#Best Performing Models Logistic Regression  are  & SVM both models performed similarly across al
#Best Performing Model is  Logistic Regression
#Worst Performing Model is  Decision Tree
```

In [ ]:
```
```