Anandh K CSE-A 3122215001009

## UCS2612 Machine Learning Laboratory

ASSIGNMENT 8 : Applications of Random Forest and AdaBoost Ensemble Techniques

import pandas as pd import numpy as np

import matplotlib.pyplot as plt import seaborn as sns

from sklearn.model\_selection import train\_test\_split from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import accuracy\_score, classification\_report, confusion\_matrix

from sklearn.preprocessing import StandardScaler, LabelEncoder

### LOADING DATASET

url =

"https://archive.ics.uci.edu/ml/machine-learning-databases/breast- cancer-wisconsin/wdbc.data"

names = ['ID', 'Diagnosis', 'mean\_radius', 'mean\_texture','mean\_perimeter', 'mean\_area',c

'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']data

= pd.read\_csv(url, names=names) data

ID Diagnosis mean\_radius mean\_texture mean\_perimeter mean\_area \

842302 M 17.99 10.38 122.80

0

1001.0

842517 M 20.57 17.77 132.90

1

1326.0

84300903 M 19.69 21.25 130.00

2

1203.0

84348301 M 11.42 20.38 77.58

3

386.1

84358402 M 20.29 14.34 135.10

4

1297.0

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |
| 564 | 926424 | M | 21.56 | 22.39 | 142.00 |
| 1479.0 |  |  |  |  |  |
| 565 | 926682 | M | 20.13 | 28.25 | 131.20 |
| 1261.0 |  |  |  |  |  |
| 566 | 926954 | M | 16.60 | 28.08 | 108.30 |
| 858.1 |  |  |  |  |  |
| 567 | 927241 | M | 20.60 | 29.33 | 140.10 |
| 1265.0 |  |  |  |  |  |
| 568 | 92751 | B | 7.76 | 24.54 | 47.92 |
|  | 181.0 |  |  |  |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| mean\_smoothness mean\_compactness mean\_concavity mean\_concave\_points \ | | | |  | |
| 0 | 0.11840 | 0.27760 | 0.30010 |  |  |
| 0.14710 |  |  |  |  |  |
| 1 | 0.08474 | 0.07864 | 0.08690 |  |  |
| 0.07017 |  |  |  |  |  |
| 2 | 0.10960 | 0.15990 | 0.19740 |  |  |
| 0.12790 |  |  |  |  |  |
| 3 | 0.14250 | 0.28390 | 0.24140 |  |  |
| 0.10520 |  |  |  |  |  |
| 4 | 0.10030 | 0.13280 | 0.19800 |  |  |
| 0.10430  .. | ... | ... | ... |  |  |
| ... |  |  |  |  |  |
| 564 | 0.11100 | 0.11590 | 0.24390 |  |  |
| 0.13890 |  |  |  |  |  |
| 565 | 0.09780 | 0.10340 | 0.14400 |  |  |
| 0.09791 |  |  |  |  |  |
| 566 | 0.08455 | 0.10230 | 0.09251 |  |  |
| 0.05302 |  |  |  |  |  |
| 567 | 0.11780 | 0.27700 | 0.35140 |  |  |
| 0.15200 |  |  |  |  |  |
| 568 | 0.05263 | 0.04362 | 0.00000 |  |  |
| 0.00000 |  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ... worst\_radius | worst\_texture | worst\_perimeter | worst\_area | \ |
| 0 ... 25.380 | 17.33 | 184.60 | 2019.0 |  |
| 1 ... 24.990 | 23.41 | 158.80 | 1956.0 |  |
| 2 ... 23.570 | 25.53 | 152.50 | 1709.0 |  |
| 3 ... 14.910 | 26.50 | 98.87 | 567.7 |  |
| 4 ... 22.540  .. ... ... | 16.67  ... | 152.20  ... | 1575.0  ... |  |
| 564 ... 25.450 | 26.40 | 166.10 | 2027.0 |  |
| 565 ... 23.690 | 38.25 | 155.00 | 1731.0 |  |
| 566 ... 18.980 | 34.12 | 126.70 | 1124.0 |  |
| 567 ... 25.740 | 39.42 | 184.60 | 1821.0 |  |
| 568 ... 9.456 | 30.37 | 59.16 | 268.6 |  |

worst\_smoothness worst\_compactness worst\_concavity \

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
| 0 | 0.16220 | 0.66560 | 0.7119 |
| 1 | 0.12380 | 0.18660 | 0.2416 |  |
| 2 | 0.14440 | 0.42450 | 0.4504 |  |
| 3 | 0.20980 | 0.86630 | 0.6869 |  |
| 4 | 0.13740 | 0.20500 | 0.4000 |  |
| .. | ... | ... | ... |  |
| 564 | 0.14100 | 0.21130 | 0.4107 |  |
| 565 | 0.11660 | 0.19220 | 0.3215 |  |
| 566 | 0.11390 | 0.30940 | 0.3403 |  |
| 567 | 0.16500 | 0.86810 | 0.9387 |  |
|  | 568 | 0.08996 | 0.06444 | 0.0000 |  |

|  |  |  |  |
| --- | --- | --- | --- |
| 0 | worst\_concave\_points  0.2654 | worst\_symmetry  0.4601 | worst\_fractal\_dimension  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 |
| .. | ... | ... | ... |
| 564 | 0.2216 | 0.2060 | 0.07115 |
| 565 | 0.1628 | 0.2572 | 0.06637 |
| 566 | 0.1418 | 0.2218 | 0.07820 |
| 567 | 0.2650 | 0.4087 | 0.12400 |
| 568 | 0.0000 | 0.2871 | 0.07039 |

[569 rows x 32 columns] data.head()

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 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] data.describe()

ID mean\_radius mean\_texture mean\_perimeter mean\_area \

count 5.690000e+02 569.000000 569.000000 569.000000

569.000000

mean 3.037183e+07 14.127292 19.289649 91.969033

654.889104

std 1.250206e+08 3.524049 4.301036 24.298981

351.914129

min 8.670000e+03 6.981000 9.710000 43.790000

143.500000

25% 8.692180e+05 11.700000 16.170000 75.170000

420.300000

50% 9.060240e+05 13.370000 18.840000 86.240000

551.100000

75% 8.813129e+06 15.780000 21.800000 104.100000

782.700000

max 9.113205e+08 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 ... worst\_radius worst\_texture worst\_perimeter \

count 569.000000 ... 569.000000 569.000000

569.000000

mean 0.181162 ... 16.269190 25.677223

107.261213

std 0.027414 ... 4.833242 6.146258

33.602542

min 0.106000 ... 7.930000 12.020000

50.410000

25% 0.161900 ... 13.010000 21.080000

84.110000

50% 0.179200 ... 14.970000 25.410000

97.660000

75% 0.195700 ... 18.790000 29.720000

125.400000

max 0.304000 ... 36.040000 49.540000

251.200000

worst\_area worst\_smoothness worst\_compactness worst\_concavity \

count 569.000000 569.000000 569.000000

569.000000

|  |  |  |  |
| --- | --- | --- | --- |
| mean 880.583128 | | 0.132369 | 0.254265 |
| 0.272188  std |  | | |
| 569.356993 | 0.022832 | 0.157336 |

1.252000

0.208624

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| min | 185.200000 | | 0.071170 | 0.027290 |
| 0.000000 | |  | | |
| 25% | 515.300000 | | 0.116600 | 0.147200 |
| 0.114500 | |  | | |
| 50% | 686.500000 | | 0.131300 | 0.211900 |
| 0.226700 | |  | | |
| 75% | 1084.000000 | | 0.146000 | 0.339100 |
| 0.382900 | |  | | |
| max | 4254.000000 | | 0.222600 | 1.058000 |

|  |  |  |  |
| --- | --- | --- | --- |
| count | worst\_concave\_points  569.000000 | worst\_symmetry 569.000000 | worst\_fractal\_dimension  569.000000 |
| mean | 0.114606 | 0.290076 | 0.083946 |
| std | 0.065732 | 0.061867 | 0.018061 |
| min | 0.000000 | 0.156500 | 0.055040 |
| 25% | 0.064930 | 0.250400 | 0.071460 |
| 50% | 0.099930 | 0.282200 | 0.080040 |
| 75% | 0.161400 | 0.317900 | 0.092080 |
| max | 0.291000 | 0.663800 | 0.207500 |

[8 rows x 31 columns]

num\_rows, num\_columns = data.shape

print("Number of rows:", num\_rows) print("Number of columns:", num\_columns)

Number of rows: 569 Number of columns: 32

data.nunique()

|  |  |
| --- | --- |
| ID | 569 |
| Diagnosis | 2 |
| mean\_radius | 456 |
| mean\_texture | 479 |
| mean\_perimeter | 522 |
| mean\_area | 539 |
| mean\_smoothness | 474 |
| mean\_compactness | 537 |
| mean\_concavity | 537 |
| mean\_concave\_points | 542 |
| mean\_symmetry | 432 |
| mean\_fractal\_dimension | 499 |
| se\_radius | 540 |
| se\_texture | 519 |
| se\_perimeter | 533 |
| se\_area | 528 |

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

print("The Number of Missing Values in the dataset\n") data.isnull().sum()

The Number of Missing Values in the dataset

dtype: int64

### 2. ENCODING CATEGORICAL TARGET VARIABLE

label\_encoder = LabelEncoder()

data['Diagnosis'] = label\_encoder.fit\_transform(data['Diagnosis'])

data

mean\_concave\_points ... worst\_radius worst\_texture worst\_perimeter \

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | ID | Diagnosis | mean\_radius | mean\_texture | mean\_perimeter | \ |
| 0 | 842302 | 1 | 17.99 | 10.38 | 122.80 |  |
| 1 | 842517 | 1 | 20.57 | 17.77 | 132.90 |  |
| 2 | 84300903 | 1 | 19.69 | 21.25 | 130.00 |  |
| 3 | 84348301 | 1 | 11.42 | 20.38 | 77.58 |  |
| 4 | 84358402 | 1 | 20.29 | 14.34 | 135.10 |  |
| .. | ... | ... | ... | ... | ... |  |
| 564 | 926424 | 1 | 21.56 | 22.39 | 142.00 |  |
| 565 | 926682 | 1 | 20.13 | 28.25 | 131.20 |  |
| 566 | 926954 | 1 | 16.60 | 28.08 | 108.30 |  |
| 567 | 927241 | 1 | 20.60 | 29.33 | 140.10 |  |
| 568 | 92751 | 0 | 7.76 | 24.54 | 47.92 |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | mean\_area | mean\_smoothness | mean\_compactness | mean\_concavity | \ |
| 0 | 1001.0 | 0.11840 | 0.27760 | 0.30010 |  |
| 1 | 1326.0 | 0.08474 | 0.07864 | 0.08690 |  |
| 2 | 1203.0 | 0.10960 | 0.15990 | 0.19740 |  |
| 3 | 386.1 | 0.14250 | 0.28390 | 0.24140 |  |
| 4 | 1297.0 | 0.10030 | 0.13280 | 0.19800 |  |
| .. | ... | ... | ... | ... |  |
| 564 | 1479.0 | 0.11100 | 0.11590 | 0.24390 |  |
| 565 | 1261.0 | 0.09780 | 0.10340 | 0.14400 |  |
| 566 | 858.1 | 0.08455 | 0.10230 | 0.09251 |  |
| 567 | 1265.0 | 0.11780 | 0.27700 | 0.35140 |  |
| 568 | 181.0 | 0.05263 | 0.04362 | 0.00000 |  |

|  |  |  |
| --- | --- | --- |
| 0 | 0.14710 ... 25.380 | 17.33 |
| 184.60 |  |  |
| 1 | 0.07017 ... 24.990 | 23.41 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 158.80 | |  | | | |
| 2 | | 0.12790 ... 23.570 | | 25.53 | |
| 152.50 | |  | |  |  |
| 3 | | 0.10520 ... 14.910 | | 26.50 |  |
| 98.87 | |  | |  |  |
| 4 | | 0.10430 ... 22.540 | | 16.67 |  |
| 152.20 | |  | |  |  |
| .. | | ... ... ... | | ... |  |
| ... | |  | |  |  |
| 564 | | 0.13890 ... 25.450 | | 26.40 |  |
| 166.10 | |  | |  |  |
| 565 | | 0.09791 ... 23.690 | | 38.25 |  |
| 155.00 | |  | |  |  |
| 566 | | 0.05302 ... 18.980 | | 34.12 |  |
| 126.70 | |  | |  |  |
| 567 | | 0.15200 ... 25.740 | | 39.42 |  |
| 184.60 | |  | |  |  |
| 568 | | 0.00000 ... 9.456 | | 30.37 |  |
| 59.16 | |  |  |  |  | |
|  | | worst\_area | worst\_smoothness | worst\_compactness | worst\_concavity | |
| \ | |  |  |  |  | |
| 0 | | 2019.0 | 0.16220 | 0.66560 | 0.7119 | |
|  | |  |  |  |  | |
| 1 | | 1956.0 | 0.12380 | 0.18660 | 0.2416 | |
|  | |  |  |  |  | |
| 2 | | 1709.0 | 0.14440 | 0.42450 | 0.4504 | |
|  | |  |  |  |  | |
| 3 | | 567.7 | 0.20980 | 0.86630 | 0.6869 | |
|  | |  |  |  |  | |
| 4 | | 1575.0 | 0.13740 | 0.20500 | 0.4000 | |
|  | |  |  |  |  | |
| .. | | ... | ... | ... | ... | |
|  | |  |  |  |  | |
| 564 | | 2027.0 | 0.14100 | 0.21130 | 0.4107 | |
|  | |  |  |  |  | |
| 565 | | 1731.0 | 0.11660 | 0.19220 | 0.3215 | |
|  | |  |  |  |  | |
| 566 | | 1124.0 | 0.11390 | 0.30940 | 0.3403 | |
|  | |  |  |  |  | |
| 567 | | 1821.0 | 0.16500 | 0.86810 | 0.9387 | |
|  | |  |  |  |  | |
| 568 | | 268.6 | 0.08996 | 0.06444 | 0.0000 | |

|  |  |  |  |
| --- | --- | --- | --- |
| 0 | worst\_concave\_points  0.2654 | worst\_symmetry  0.4601 | worst\_fractal\_dimension  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 |
| .. | ... | ... | ... |
| 564 | 0.2216 | 0.2060 | 0.07115 |
| 565 | 0.1628 | 0.2572 | 0.06637 |
| 566 | 0.1418 | 0.2218 | 0.07820 |
| 567 | 0.2650 | 0.4087 | 0.12400 |
|  | 568 | 0.0000 | 0.2871 | 0.07039 |

[569 rows x 32 columns] data.info()

|  |  |  |  |
| --- | --- | --- | --- |
| <class 'pandas.core.frame.DataFrame'> RangeIndex: 569 entries, 0 to 568 Data columns (total 32 columns):  # Column Non-Null Count | | | Dtype |
| 0 | ID | 569 non-null | int64 |
| 1 | Diagnosis | 569 non-null | int32 |
| 2 | mean\_radius | 569 non-null | float64 |
| 3 | mean\_texture | 569 non-null | float64 |
| 4 | mean\_perimeter | 569 non-null | float64 |
| 5 | mean\_area | 569 non-null | float64 |
| 6 | mean\_smoothness | 569 non-null | float64 |
| 7 | mean\_compactness | 569 non-null | float64 |
| 8 | mean\_concavity | 569 non-null | float64 |
| 9 | mean\_concave\_points | 569 non-null | float64 |
| 10 | mean\_symmetry | 569 non-null | float64 |
| 11 | mean\_fractal\_dimension | 569 non-null | float64 |
| 12 | se\_radius | 569 non-null | float64 |
| 13 | se\_texture | 569 non-null | float64 |
| 14 | se\_perimeter | 569 non-null | float64 |
| 15 | se\_area | 569 non-null | float64 |
| 16 | se\_smoothness | 569 non-null | float64 |
| 17 | se\_compactness | 569 non-null | float64 |
| 18 | se\_concavity | 569 non-null | float64 |
| 19 | se\_concave\_points | 569 non-null | float64 |
| 20 | se\_symmetry | 569 non-null | float64 |
| 21 | se\_fractal\_dimension | 569 non-null | float64 |
| 22 | worst\_radius | 569 non-null | float64 |
| 23 | worst\_texture | 569 non-null | float64 |
| 24 | worst\_perimeter | 569 non-null | float64 |
| 25 | worst\_area | 569 non-null | float64 |
| 26 | worst\_smoothness | 569 non-null | float64 |
| 27 | worst\_compactness | 569 non-null | float64 |
| 28 | worst\_concavity | 569 non-null | float64 |
| 29 | worst\_concave\_points | 569 non-null | float64 |
| 30 | worst\_symmetry | 569 non-null | float64 |
| 31 | worst\_fractal\_dimension | 569 non-null | float64 |

dtypes: float64(30), int32(1), int64(1) memory usage: 140.2 KB

### 1. NORMALIZATION AND STANDARDIZATION

scaler = StandardScaler()

data\_scaled = scaler.fit\_transform(data.drop(['ID', 'Diagnosis'],axis=1)) data\_scaled = pd.DataFrame(data\_scaled, columns=data.columns[2:])

data

ID Diagnosis mean\_radius mean\_texture mean\_perimeter \

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0 | 842302 | 1 | 17.99 | 10.38 | 122.80 |
| 1 | 842517 | 1 | 20.57 | 17.77 | 132.90 |
| 2 | 84300903 | 1 | 19.69 | 21.25 | 130.00 |
| 3 | 84348301 | 1 | 11.42 | 20.38 | 77.58 |
| 4 | 84358402 | 1 | 20.29 | 14.34 | 135.10 |
| .. | ... | ... | ... | ... | ... |
| 564 | 926424 | 1 | 21.56 | 22.39 | 142.00 |
| 565 | 926682 | 1 | 20.13 | 28.25 | 131.20 |
| 566 | 926954 | 1 | 16.60 | 28.08 | 108.30 |
| 567 | 927241 | 1 | 20.60 | 29.33 | 140.10 |
| 568 | 92751 | 0 | 7.76 | 24.54 | 47.92 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | mean\_area | mean\_smoothness | mean\_compactness | mean\_concavity | \ |
| 0 | 1001.0 | 0.11840 | 0.27760 | 0.30010 |  |
| 1 | 1326.0 | 0.08474 | 0.07864 | 0.08690 |  |
| 2 | 1203.0 | 0.10960 | 0.15990 | 0.19740 |  |
| 3 | 386.1 | 0.14250 | 0.28390 | 0.24140 |  |
| 4  .. | 1297.0  ... | 0.10030  ... | 0.13280  ... | 0.19800  ... |  |
| 564 | 1479.0 | 0.11100 | 0.11590 | 0.24390 |  |
| 565 | 1261.0 | 0.09780 | 0.10340 | 0.14400 |  |
| 566 | 858.1 | 0.08455 | 0.10230 | 0.09251 |  |
| 567 | 1265.0 | 0.11780 | 0.27700 | 0.35140 |  |
| 568 | 181.0 | 0.05263 | 0.04362 | 0.00000 |  |

mean\_concave\_points ... worst\_radius worst\_texture worst\_perimeter \

|  |  |  |
| --- | --- | --- |
| 0 | 0.14710 ... 25.380 | 17.33 |
| 184.60 |  |  |
| 1 | 0.07017 ... 24.990 | 23.41 |
| 158.80 |  |  |
| 2 | 0.12790 ... 23.570 | 25.53 |
| 152.50 |  |  |
| 3 | 0.10520 ... 14.910 | 26.50 |
| 98.87 |  |  |
| 4 | 0.10430 ... 22.540 | 16.67 |
| 152.20 |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | |  | |  |  |
| .. | | ... ... ... | | ... | |
| ... | |  | |  |  |
| 564 | | 0.13890 ... 25.450 | | 26.40 |  |
| 166.10 | |  | |  |  |
| 565 | | 0.09791 ... 23.690 | | 38.25 |  |
| 155.00 | |  | |  |  |
| 566 | | 0.05302 ... 18.980 | | 34.12 |  |
| 126.70 | |  | |  |  |
| 567 | | 0.15200 ... 25.740 | | 39.42 |  |
| 184.60 | |  | |  |  |
| 568 | | 0.00000 ... 9.456 | | 30.37 |  |
| 59.16 | |  |  |  |  | |
|  | | worst\_area | worst\_smoothness | worst\_compactness | worst\_concavity | |
| \ | |  |  |  |  | |
| 0 | | 2019.0 | 0.16220 | 0.66560 | 0.7119 | |
|  | |  |  |  |  | |
| 1 | | 1956.0 | 0.12380 | 0.18660 | 0.2416 | |
|  | |  |  |  |  | |
| 2 | | 1709.0 | 0.14440 | 0.42450 | 0.4504 | |
|  | |  |  |  |  | |
| 3 | | 567.7 | 0.20980 | 0.86630 | 0.6869 | |
|  | |  |  |  |  | |
| 4 | | 1575.0 | 0.13740 | 0.20500 | 0.4000 | |
|  | |  |  |  |  | |
| .. | | ... | ... | ... | ... | |
|  | |  |  |  |  | |
| 564 | | 2027.0 | 0.14100 | 0.21130 | 0.4107 | |
|  | |  |  |  |  | |
| 565 | | 1731.0 | 0.11660 | 0.19220 | 0.3215 | |
|  | |  |  |  |  | |
| 566 | | 1124.0 | 0.11390 | 0.30940 | 0.3403 | |
|  | |  |  |  |  | |
| 567 | | 1821.0 | 0.16500 | 0.86810 | 0.9387 | |
|  | |  |  |  |  | |
| 568 | | 268.6 | 0.08996 | 0.06444 | 0.0000 | |

|  |  |  |  |
| --- | --- | --- | --- |
|  | 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 |
| .. | ... | ... | ... |
| 564 | 0.2216 | 0.2060 | 0.07115 |
| 565 | 0.1628 | 0.2572 | 0.06637 |
| 566 | 0.1418 | 0.2218 | 0.07820 |
| 567 | 0.2650 | 0.4087 | 0.12400 |
| 568 | 0.0000 | 0.2871 | 0.07039 |

[569 rows x 32 columns]

### EXPLORATORY DATA ANALYSIS 1.CORRELATION HEATMAP

import matplotlib.pyplot as plt import seaborn as sns correlation\_matrix = data.corr()

plt.figure(figsize=(25, 15))

sns.heatmap(correlation\_matrix, annot=True, linecolor='black', fmt='.2f', linewidths=.1)

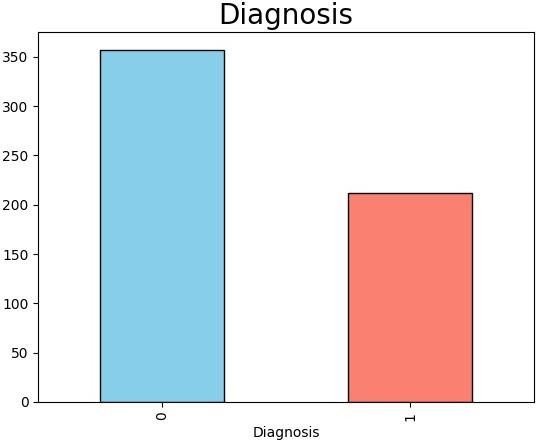
plt.title('Correlation Heatmap') plt.show()

1. BAR-CHART

data['Diagnosis'].value\_counts().plot(kind='bar',edgecolor='black',col or=['skyblue','salmon'])

plt.title("Diagnosis",fontsize=20)

plt.show() data['Diagnosis'].value\_counts()



### 1. PIE-CHART OF TARGET COLUMN

*# Count occurrences of each unique value in the 'Diagnosis' column*

diagnosis\_counts = data['Diagnosis'].value\_counts()

*# Define colors for each slice*

colors = ['skyblue', 'salmon']

*# Plotting the pie chart with custom colors*

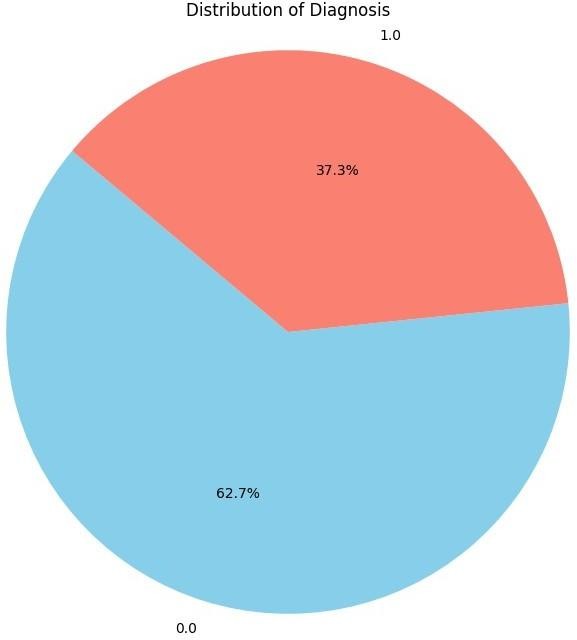
plt.figure(figsize=(8, 8))

plt.pie(diagnosis\_counts, labels=diagnosis\_counts.index, autopct='%1.1f%%', startangle=140, colors=colors) plt.title('Distribution of Diagnosis')

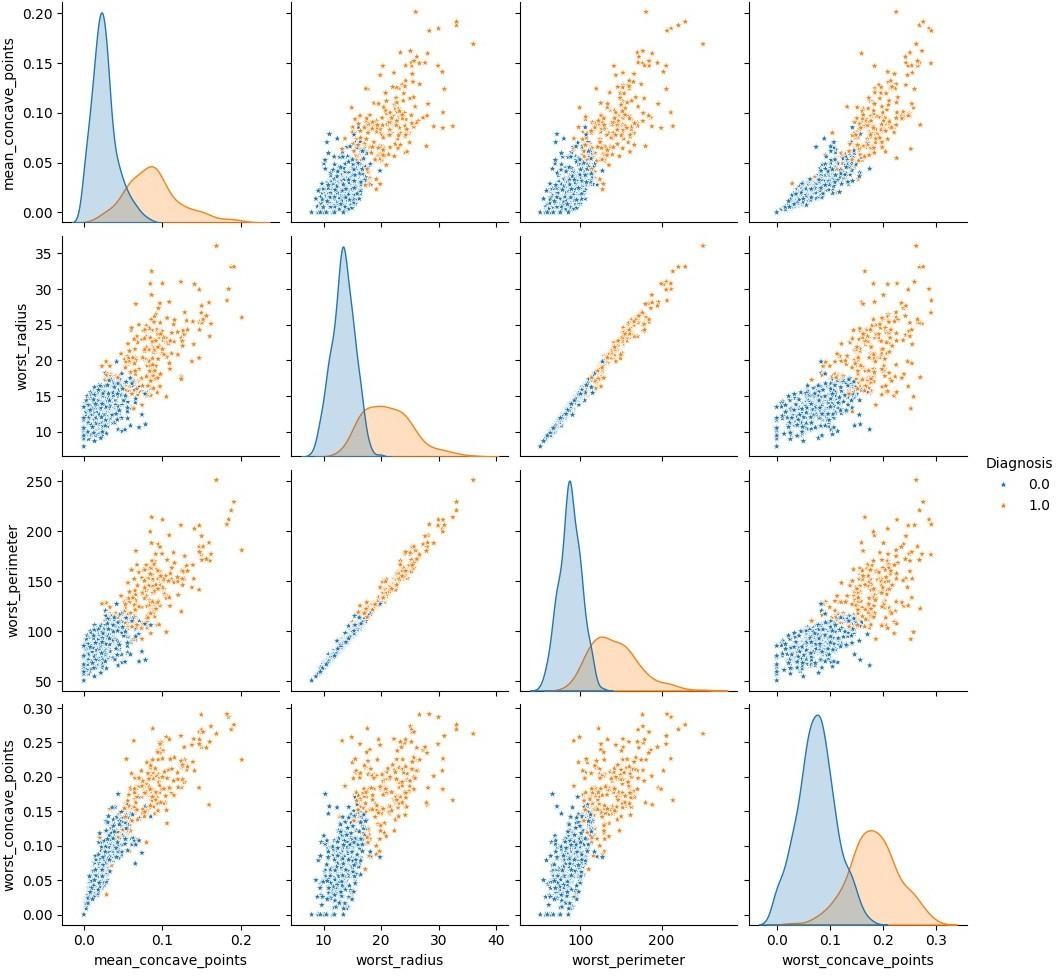
plt.axis('equal') *# Equal aspect ratio ensures that pie is drawn as a*

*circle.*

plt.show()



### 1. PAIRPLOT FOR HIGHLY CORRELATED FEATURES



*# Remove 'ID' column*

data.drop('ID', axis=1, inplace=True)

*# Convert 'Diagnosis' column to float64*

data['Diagnosis'] = data['Diagnosis'].astype(float) data.info()

<class 'pandas.core.frame.DataFrame'> RangeIndex: 569 entries, 0 to 568 Data columns (total 31 columns):

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # |  | Column | Non-Null Count | Dtype |
|  |  |  |  |  |
| 0 |  | Diagnosis | 569 non-null | float64 |
| 1 |  | mean\_radius | 569 non-null | float64 |
| 2 |  | mean\_texture | 569 non-null | float64 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
| 3 mean\_perimeter | 569 | non-null | float64 |
| 4 mean\_area | 569 | non-null | float64 |
| 5 mean\_smoothness | 569 | non-null | float64 |
| 6 mean\_compactness | 569 | non-null | float64 |
| 7 mean\_concavity | 569 | non-null | float64 |
| 8 mean\_concave\_points | 569 | non-null | float64 |
| 9 mean\_symmetry | 569 | non-null | float64 |
| 10 mean\_fractal\_dimension | 569 | non-null | float64 |
| 11 se\_radius | 569 | non-null | float64 |
| 12 se\_texture | 569 | non-null | float64 |
| 13 se\_perimeter | 569 | non-null | float64 |
| 14 se\_area | 569 | non-null | float64 |
| 15 se\_smoothness | 569 | non-null | float64 |
| 16 se\_compactness | 569 | non-null | float64 |
| 17 se\_concavity | 569 | non-null | float64 |
| 18 se\_concave\_points | 569 | non-null | float64 |
| 19 se\_symmetry | 569 | non-null | float64 |
| 20 se\_fractal\_dimension | 569 | non-null | float64 |
| 21 worst\_radius | 569 | non-null | float64 |
| 22 worst\_texture | 569 | non-null | float64 |
| 23 worst\_perimeter | 569 | non-null | float64 |
| 24 worst\_area | 569 | non-null | float64 |
| 25 worst\_smoothness | 569 | non-null | float64 |
| 26 worst\_compactness | 569 | non-null | float64 |
| 27 worst\_concavity | 569 | non-null | float64 |
| 28 worst\_concave\_points | 569 | non-null | float64 |
| 29 worst\_symmetry | 569 | non-null | float64 |
|  | 30 worst\_fractal\_dimension | 569 | non-null | float64 |

dtypes: float64(31) memory usage: 137.9 KB

### SPLITTING DATA INTO TRAIN , TEST AND VALIDATION SETS

from sklearn.model\_selection import train\_test\_split from sklearn.metrics import roc\_curve, auc

from sklearn.ensemble import BaggingClassifier, RandomForestClassifier, AdaBoostClassifier import matplotlib.pyplot as plt

*# Step 5: Split the data into training, testing, and validation sets*

X = data.drop('Diagnosis', axis=1) y = data['Diagnosis']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

X\_train, X\_val, y\_train, y\_val = train\_test\_split(X\_train, y\_train, test\_size=0.25, random\_state=42) *# 60% train, 20% validation, 20% test*

from sklearn.ensemble import RandomForestClassifier

classifier = RandomForestClassifier(n\_estimators = 10, criterion = 'entropy', random\_state = 0)

classifier.fit(X\_train, y\_train)

RandomForestClassifier(criterion='entropy', n\_estimators=10, random\_state=0)

### TRAINING AND TESTING MODEL

*#Train the model*

models = {

"Bagging": BaggingClassifier(),

"Random Forest": RandomForestClassifier(), "AdaBoost": AdaBoostClassifier()

}

for name, model in models.items(): model.fit(X\_train, y\_train)

*#Test the model*

results = {}

for name, model in models.items(): y\_pred = model.predict(X\_test) results[name] = y\_pred

c:\Users\HP\AppData\Local\Programs\Python\Python311\Lib\site-packages\ sklearn\ensemble\\_weight\_boosting.py:519: FutureWarning: The SAMME.R algorithm (the default) is deprecated and will be removed in 1.6. Use the SAMME algorithm to circumvent this warning.

warnings.warn(

*# Step 8: Measure the performance of the trained model # (Assuming binary classification)*

*# (Assuming binary classification)*

def calculate\_roc(y\_true, y\_pred):

fpr, tpr, thresholds = roc\_curve(y\_true, y\_pred) roc\_auc = auc(fpr, tpr)

return fpr, tpr, roc\_auc plt.figure(figsize=(8, 6))

<Figure size 800x600 with 0 Axes>

*# Step 9: Compare the results of each ensemble model using graphs*

for name, y\_pred in results.items():

fpr, tpr, roc\_auc = calculate\_roc(y\_test, y\_pred) plt.plot(fpr, tpr, label=f'{name} (AUC = {roc\_auc:.2f})')

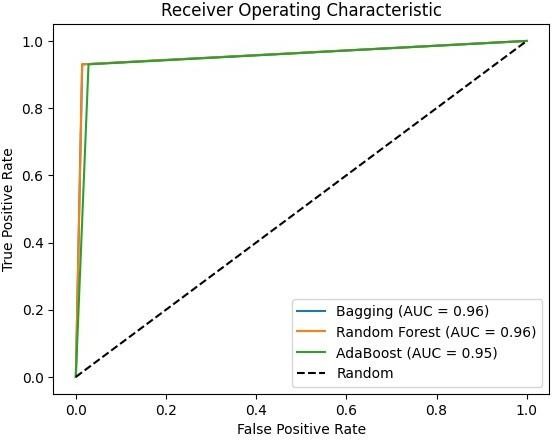
accuracy = accuracy\_score(y\_test, y\_pred)

print(f"{name}: Accuracy = {accuracy:.4f}")

plt.plot([0, 1], [0, 1], 'k--', label='Random') plt.xlabel('False Positive Rate') plt.ylabel('True Positive Rate') plt.title('Receiver Operating Characteristic') plt.legend(loc='lower right')

plt.show()

Bagging: Accuracy = 0.9649 Random Forest: Accuracy = 0.9649 AdaBoost: Accuracy = 0.9561



*# Print classification report*

print(f"\n{name} Classification Report:\ n{classification\_report(y\_test, y\_pred)}")

*# Plot confusion matrix*

plt.figure()

cm = confusion\_matrix(y\_test, y\_pred) sns.heatmap(cm, annot=True, cmap='Blues', fmt='g') plt.title(f'Confusion Matrix - {name}') plt.xlabel('Predicted')

plt.ylabel('True') plt.show()

Bagging: Accuracy = 0.9649

Bagging Classification Report:

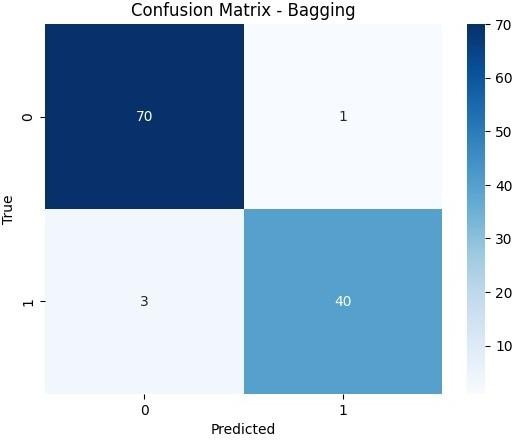
precision recall f1-score support

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 0.0 | 0.96 | 0.99 | 0.97 | 71 |
| 1.0 | 0.98 | 0.93 | 0.95 | 43 |
|  |  |  |  |  |
| accuracy |  |  | 0.96 | 114 |
| macro avg | 0.97 | 0.96 | 0.96 | 114 |
| weighted avg | 0.97 | 0.96 | 0.96 | 114 |

Random Forest: Accuracy = 0.9649

Random Forest Classification Report:

precision recall f1-score support

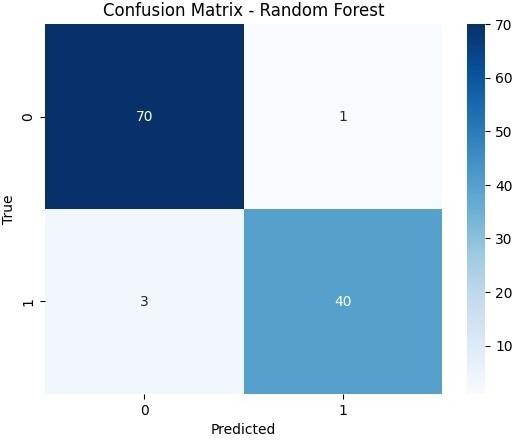


|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 0.0 | 0.96 | 0.99 | 0.97 | 71 |
| 1.0 | 0.98 | 0.93 | 0.95 | 43 |
|  |  |  |  |  |
| accuracy |  |  | 0.96 | 114 |
| macro avg | 0.97 | 0.96 | 0.96 | 114 |
| weighted avg | 0.97 | 0.96 | 0.96 | 114 |

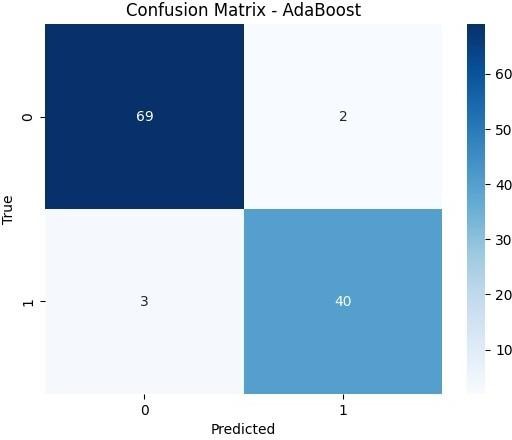
AdaBoost: Accuracy = 0.9561

AdaBoost Classification Report:

precision recall f1-score support



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 0.0 | 0.96 | 0.97 | 0.97 | 71 |
| 1.0 | 0.95 | 0.93 | 0.94 | 43 |
|  |  |  |  |  |
| accuracy |  |  | 0.96 | 114 |
| macro avg | 0.96 | 0.95 | 0.95 | 114 |
| weighted avg | 0.96 | 0.96 | 0.96 | 114 |



*# Step 10: Represent the ROC of training and test results in the graphs*

plt.figure(figsize=(8, 6))

for name, model in models.items(): y\_train\_pred = model.predict(X\_train)

fpr\_train, tpr\_train, roc\_auc\_train = calculate\_roc(y\_train, y\_train\_pred)

plt.plot(fpr\_train, tpr\_train, label=f'{name} Train (AUC =

{roc\_auc\_train:.2f})', linestyle='--')

fpr\_test, tpr\_test, roc\_auc\_test = calculate\_roc(y\_test, results[name])

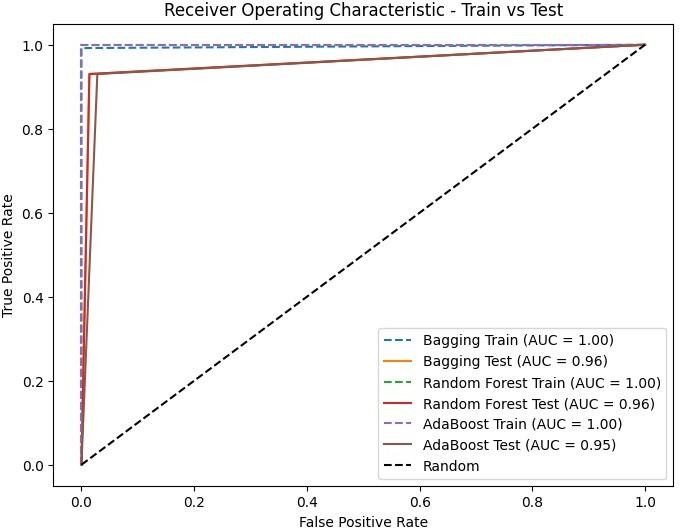
plt.plot(fpr\_test, tpr\_test, label=f'{name} Test (AUC =

{roc\_auc\_test:.2f})')

plt.plot([0, 1], [0, 1], 'k--', label='Random') plt.xlabel('False Positive Rate') plt.ylabel('True Positive Rate')

plt.title('Receiver Operating Characteristic - Train vs Test') plt.legend(loc='lower right')

plt.show()



# INFERENCE :

* Bagging and Random Forest models achieved identical accuracy scores of 0.9649, while AdaBoost had a slightly lower accuracy of 0.9561.
* Precision, recall, and F1-score metrics indicate high performance across all models, suggesting strong predictive capability.
* Detailed classification reports provide insights into the performance of each model for both classes (0 and 1), demonstrating their ability to correctly classify instances.
* Overall, Bagging and Random Forest models slightly outperform AdaBoost in diagnosing breast cancer.
* The models exhibit high accuracy and robustness, as evidenced by the AUC values.
* Visualizations of ROC curves can offer further insights into the comparative performance of the models.

# LEARNING OUTCOMES :

* Ensemble Learning: Understanding and applying ensemble techniques for classification tasks.
* Data Preprocessing: Handling missing data, encoding categorical variables, and scaling features for model training.
* Model Evaluation: Using classification metrics to assess model performance.
* Feature Engineering: Implementing techniques to enhance model predictive capability.
* Model Evaluation: Familiarity with classification metrics like accuracy, precision, recall, and F1-score for assessing model performance.

# GITHUB LINK :

[**https://github.com/Anandh-007/Machine-learning-lab/tree/main/ML\_A8**](ML_A8.docx)