Breast Cancer Prediction Utilizing Machine Learning Supervised Classification Models:

Importing the libraries:

- Pandas
- Numpy
- Plotly
- Seaborn

Importing The dataset and Displaying some observations from the dataset:

- The Dataset Source is : **Breast Cancer Dataset**

The first 5 records:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave_points_mean	...	radius_
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	...	
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	...	
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	...	
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	...	
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	...	

5 rows × 32 columns

The last 5 records:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave_points_mean	...	radius_
564	926424	M	21.56	22.39	142.00	1479.0	0.11100	0.11590	0.24390	0.13890	...	2
565	926682	M	20.13	28.25	131.20	1261.0	0.09780	0.10340	0.14400	0.09791	...	2
566	926954	M	16.60	28.08	108.30	858.1	0.08455	0.10230	0.09251	0.05302	...	1
567	927241	M	20.60	29.33	140.10	1265.0	0.11780	0.27700	0.35140	0.15200	...	2
568	92751	B	7.76	24.54	47.92	181.0	0.05263	0.04362	0.00000	0.00000	...	

5 rows × 32 columns

Repalcing the unique values of the diagnosis columns:

- From M (malignant) and B (Benign) to 1 and 0:

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave_points_mean	...	radius_
0	842302	1	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	...	
1	842517	1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	...	
2	84300903	1	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	...	
3	84348301	1	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	...	
4	84358402	1	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	...	

5 rows × 32 columns

Rearrange the columns in the dataset:

	id	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave_points_mean	symmetry_mean	...
0	842302	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	0.2419	...
1	842517	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	0.1812	...
2	84300903	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	0.2069	...
3	84348301	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	0.2597	...
4	84358402	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	0.1809	...

5 rows × 32 columns

Selecting the columns for the prediction:

	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave_points_mean	symmetry_mean	fractal_dimensio
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	0.2419	
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	0.1812	
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	0.2069	
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	0.2597	
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	0.1809	

5 rows × 31 columns

Viewing some insights about the dataset:

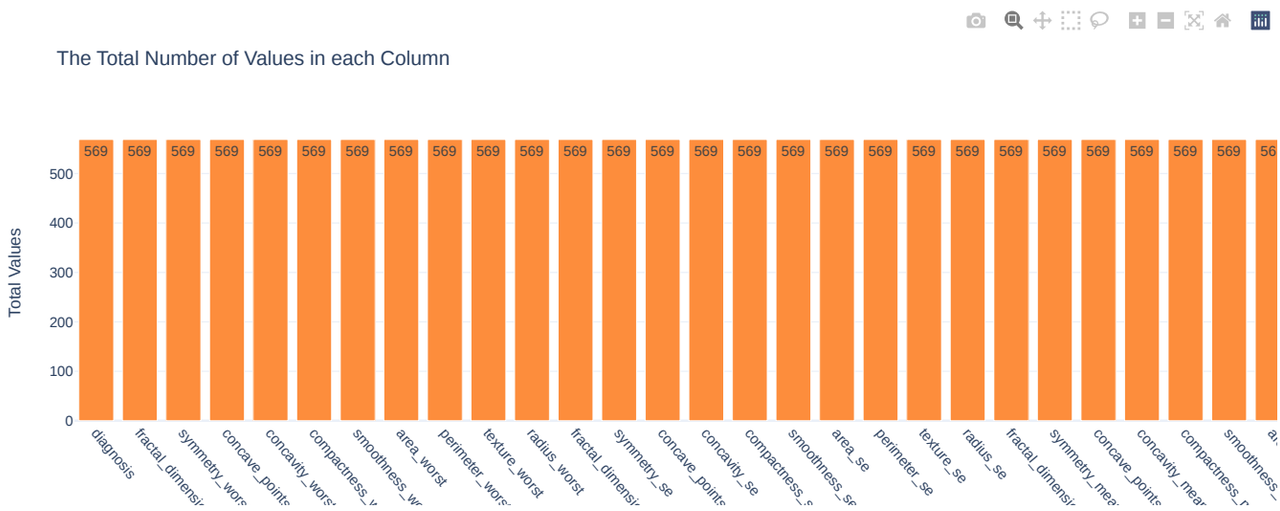
Column names, data typse and number of values in each column:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 31 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   radius_mean              569 non-null    float64
 1   texture_mean             569 non-null    float64
 2   perimeter_mean           569 non-null    float64
 3   area_mean                569 non-null    float64
 4   smoothness_mean          569 non-null    float64
 5   compactness_mean         569 non-null    float64
 6   concavity_mean           569 non-null    float64
 7   concave_points_mean      569 non-null    float64
 8   symmetry_mean            569 non-null    float64
 9   fractal_dimension_mean   569 non-null    float64
 10  radius_se                569 non-null    float64
 11  texture_se               569 non-null    float64
 12  perimeter_se             569 non-null    float64
 13  area_se                  569 non-null    float64
 14  smoothness_se            569 non-null    float64
 15  compactness_se           569 non-null    float64
 16  concavity_se             569 non-null    float64
 17  concave_points_se        569 non-null    float64
 18  symmetry_se              569 non-null    float64
 19  fractal_dimension_se     569 non-null    float64
 20  radius_worst             569 non-null    float64
 21  texture_worst            569 non-null    float64
 22  perimeter_worst          569 non-null    float64
 23  area_worst               569 non-null    float64
 24  smoothness_worst         569 non-null    float64
 25  compactness_worst        569 non-null    float64
 26  concavity_worst          569 non-null    float64
 27  concave_points_worst     569 non-null    float64
 28  symmetry_worst           569 non-null    float64
 29  fractal_dimension_worst  569 non-null    float64
 30  diagnosis                569 non-null    int64
dtypes: float64(30), int64(1)
memory usage: 137.9 KB
```

The total number of values of each column in the dataset:

column	total_values
radius_mean	569
texture_mean	569
perimeter_mean	569
area_mean	569
smoothness_mean	569
compactness_mean	569
concavity_mean	569
concave_points_mean	569
symmetry_mean	569
fractal_dimension_mean	569
radius_se	569
texture_se	569
perimeter_se	569
area_se	569
smoothness_se	569
compactness_se	569
concavity_se	569
concave_points_se	569
symmetry_se	569
fractal_dimension_se	569
radius_worst	569
texture_worst	569

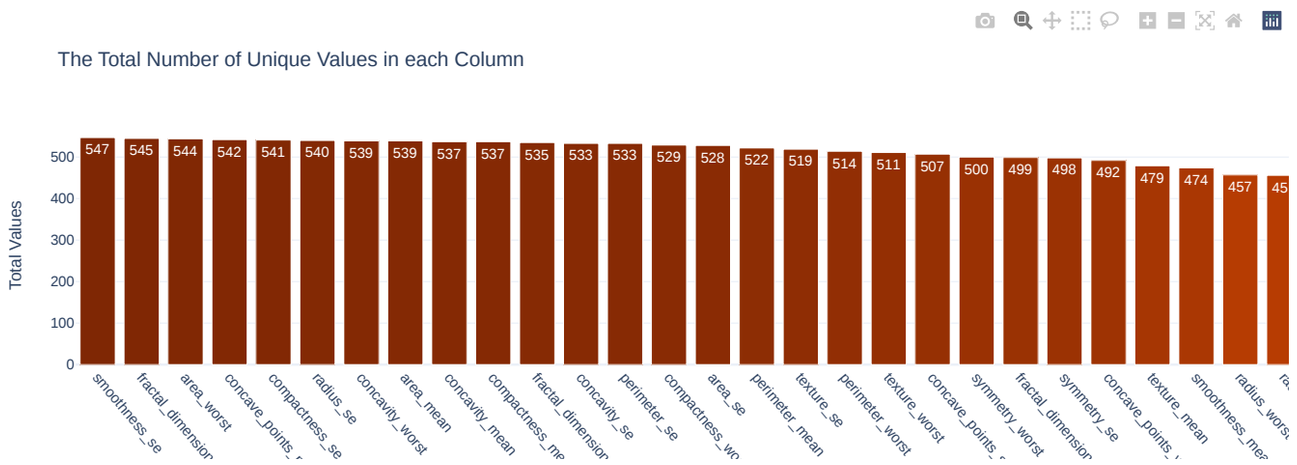
perimeter_worst	569
area_worst	569
smoothness_worst	569
compactness_worst	569
concavity_worst	569
concave_points_worst	569
symmetry_worst	569
fractal_dimension_worst	569
diagnosis	569



The Total number of the unique values in each columns:

column	total_values
radius_mean	456
texture_mean	479
perimeter_mean	522
area_mean	539
smoothness_mean	474
compactness_mean	537
concavity_mean	537
concave_points_mean	542
symmetry_mean	432
fractal_dimension_mean	499
radius_se	540
texture_se	519
perimeter_se	533
area_se	528
smoothness_se	547
compactness_se	541
concavity_se	533
concave_points_se	507
symmetry_se	498
fractal_dimension_se	545
radius_worst	457
texture_worst	511
perimeter_worst	514
area_worst	544
smoothness_worst	411
compactness_worst	529
concavity_worst	539
concave_points_worst	492

symmetry_worst	500
fractal_dimension_worst	535
diagnosis	2



Figuring out if there are missing values in the dataset:

	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave_points_mean	symmetry_mean	fractal_dimer
0	False	False	False	False	False	False	False	False	False	
1	False	False	False	False	False	False	False	False	False	
2	False	False	False	False	False	False	False	False	False	
3	False	False	False	False	False	False	False	False	False	
4	False	False	False	False	False	False	False	False	False	
...	...	...	...	...	...	...	...	...	...	
564	False	False	False	False	False	False	False	False	False	
565	False	False	False	False	False	False	False	False	False	
566	False	False	False	False	False	False	False	False	False	
567	False	False	False	False	False	False	False	False	False	
568	False	False	False	False	False	False	False	False	False	

569 rows × 31 columns

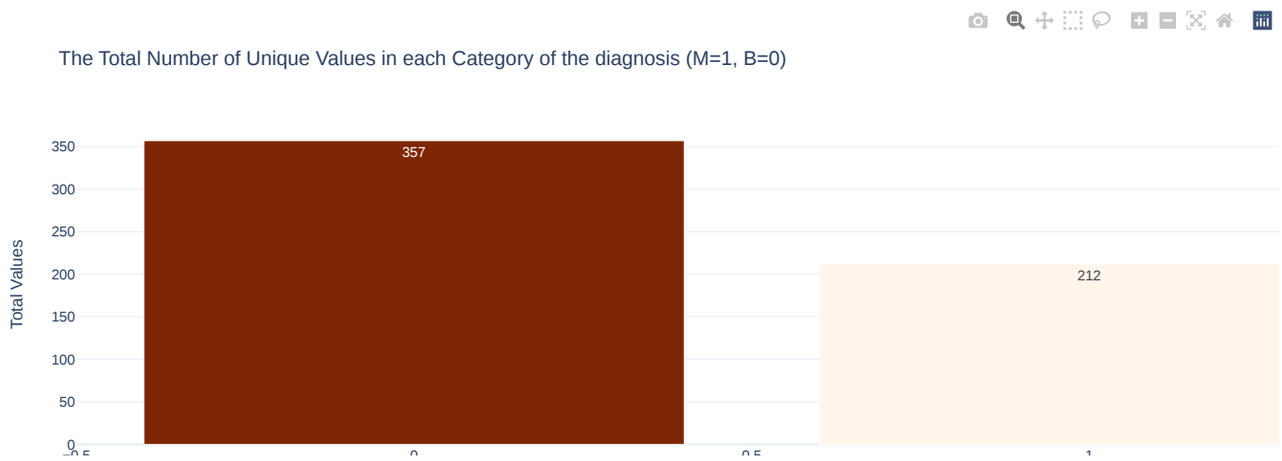
Counting the number of the missing values in each column:

	total_values
column	
radius_mean	0
texture_mean	0
perimeter_mean	0
area_mean	0
smoothness_mean	0
compactness_mean	0
concavity_mean	0
concave_points_mean	0
symmetry_mean	0
fractal_dimension_mean	0
radius_se	0
texture_se	0
perimeter_se	0
area_se	0
smoothness_se	0
compactness_se	0
concavity_se	0
concave_points_se	0
symmetry_se	0

fractal_dimension_se	0
radius_worst	0
texture_worst	0
perimeter_worst	0
area_worst	0
smoothness_worst	0
compactness_worst	0
concavity_worst	0
concave_points_worst	0
symmetry_worst	0
fractal_dimension_worst	0
diagnosis	0

Finding out the total number of values based on the unique values of the diagnosis:

diagnosis	total_values
0	357
1	212



Getting the total number of missing values in the entire dataset:

```
0
```

Checking for the duplicated data:

```
0      False
1      False
2      False
3      False
4      False
       ...
564    False
565    False
566    False
567    False
568    False
Length: 569, dtype: bool
```

Finding out the number of duplicated data in each column:

```
0
```

Checking for the total number of duplicated data in the entire datset:

```
0
```

Calculating some statistcal information about the dataset:

	count	mean	std	min	25%	50%	75%	max
radius_mean	569.0	14.127292	3.524049	6.981000	11.700000	13.370000	15.780000	28.11000
texture_mean	569.0	19.289649	4.301036	9.710000	16.170000	18.840000	21.800000	39.28000
perimeter_mean	569.0	91.969033	24.298981	43.790000	75.170000	86.240000	104.100000	188.50000
area_mean	569.0	654.889104	351.914129	143.500000	420.300000	551.100000	782.700000	2501.00000
smoothness_mean	569.0	0.096360	0.014064	0.052630	0.086370	0.095870	0.105300	0.16340
compactness_mean	569.0	0.104341	0.052813	0.019380	0.064920	0.092630	0.130400	0.34540

concavity_mean	569.0	0.088799	0.079720	0.000000	0.029560	0.061540	0.130700	0.42680
concave_points_mean	569.0	0.048919	0.038803	0.000000	0.020310	0.033500	0.074000	0.20120
symmetry_mean	569.0	0.181162	0.027414	0.106000	0.161900	0.179200	0.195700	0.30400
fractal_dimension_mean	569.0	0.062798	0.007060	0.049960	0.057700	0.061540	0.066120	0.09744
radius_se	569.0	0.405172	0.277313	0.111500	0.232400	0.324200	0.478900	2.87300
texture_se	569.0	1.216853	0.551648	0.360200	0.833900	1.108000	1.474000	4.88500
perimeter_se	569.0	2.866059	2.021855	0.757000	1.606000	2.287000	3.357000	21.98000
area_se	569.0	40.337079	45.491006	6.802000	17.850000	24.530000	45.190000	542.20000
smoothness_se	569.0	0.007041	0.003003	0.001713	0.005169	0.006380	0.008146	0.03113
compactness_se	569.0	0.025478	0.017908	0.002252	0.013080	0.020450	0.032450	0.13540
concavity_se	569.0	0.031894	0.030186	0.000000	0.015090	0.025890	0.042050	0.39600
concave_points_se	569.0	0.011796	0.006170	0.000000	0.007638	0.010930	0.014710	0.05279
symmetry_se	569.0	0.020542	0.008266	0.007882	0.015160	0.018730	0.023480	0.07895
fractal_dimension_se	569.0	0.003795	0.002646	0.000895	0.002248	0.003187	0.004558	0.02984
radius_worst	569.0	16.269190	4.833242	7.930000	13.010000	14.970000	18.790000	36.04000
texture_worst	569.0	25.677223	6.146258	12.020000	21.080000	25.410000	29.720000	49.54000
perimeter_worst	569.0	107.261213	33.602542	50.410000	84.110000	97.660000	125.400000	251.20000
area_worst	569.0	880.583128	569.356993	185.200000	515.300000	686.500000	1084.000000	4254.00000
smoothness_worst	569.0	0.132369	0.022832	0.071170	0.116600	0.131300	0.146000	0.22260
compactness_worst	569.0	0.254265	0.157336	0.027290	0.147200	0.211900	0.339100	1.05800
concavity_worst	569.0	0.272188	0.208624	0.000000	0.114500	0.226700	0.382900	1.25200
concave_points_worst	569.0	0.114606	0.065732	0.000000	0.064930	0.099930	0.161400	0.29100
symmetry_worst	569.0	0.290076	0.061867	0.156500	0.250400	0.282200	0.317900	0.66380
fractal_dimension_worst	569.0	0.083946	0.018061	0.055040	0.071460	0.080040	0.092080	0.20750
diagnosis	569.0	0.372583	0.483918	0.000000	0.000000	0.000000	1.000000	1.00000

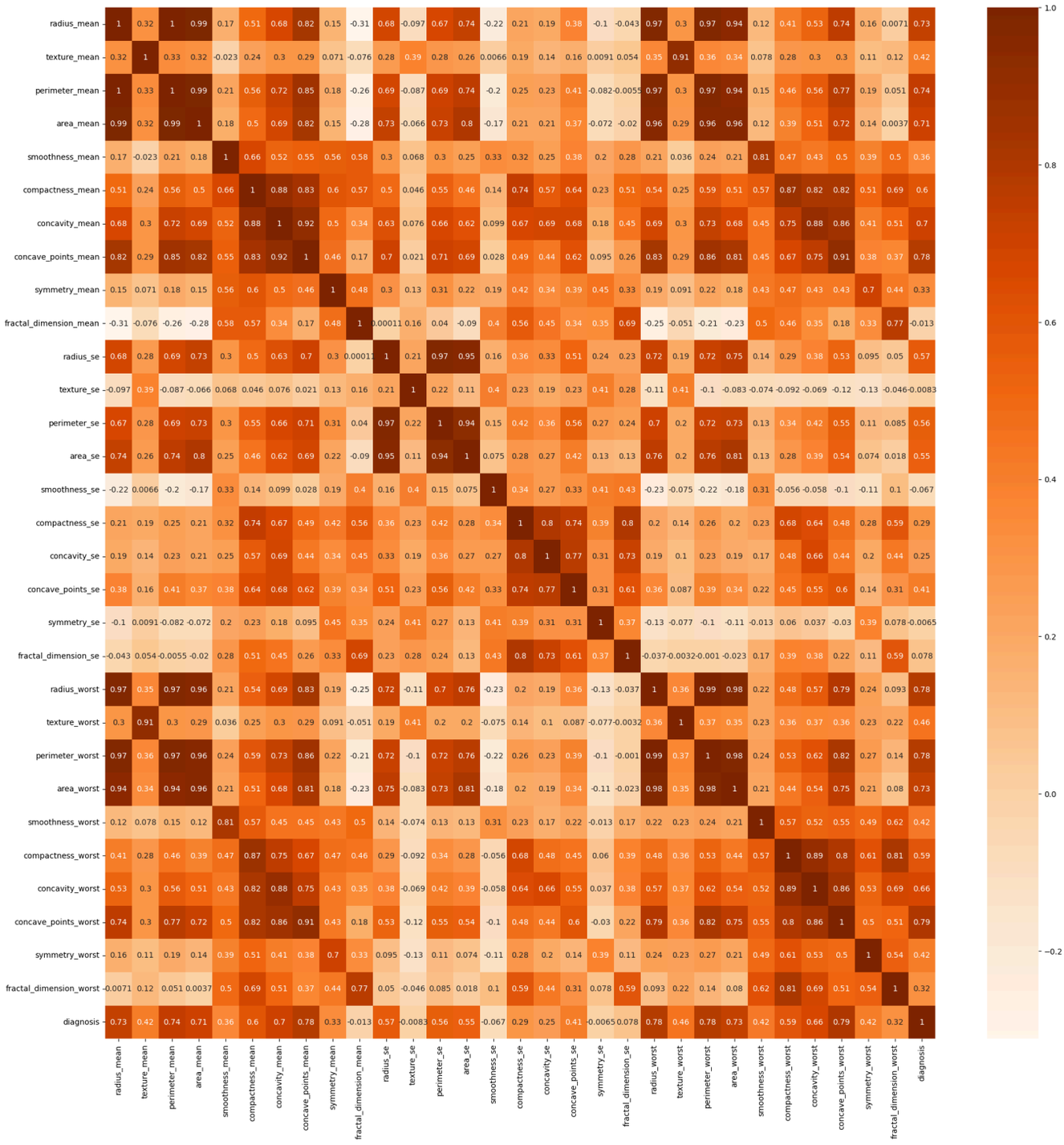
Identify the correlation between the "diagnosis" column and the other columns:

	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave_points_mean	symmetry_
radius_mean	1.000000	0.323782	0.997855	0.987357	0.170581	0.506124	0.676764	0.822529	0.14
texture_mean	0.323782	1.000000	0.329533	0.321086	-0.023389	0.236702	0.302418	0.293464	0.07
perimeter_mean	0.997855	0.329533	1.000000	0.986507	0.207278	0.556936	0.716136	0.850977	0.18
area_mean	0.987357	0.321086	0.986507	1.000000	0.177028	0.498502	0.685983	0.823269	0.15
smoothness_mean	0.170581	-0.023389	0.207278	0.177028	1.000000	0.659123	0.521984	0.553695	0.55
compactness_mean	0.506124	0.236702	0.556936	0.498502	0.659123	1.000000	0.883121	0.831135	0.60
concavity_mean	0.676764	0.302418	0.716136	0.685983	0.521984	0.883121	1.000000	0.921391	0.50
concave_points_mean	0.822529	0.293464	0.850977	0.823269	0.553695	0.831135	0.921391	1.000000	0.46
symmetry_mean	0.147741	0.071401	0.183027	0.151293	0.557775	0.602641	0.500667	0.462497	1.00
fractal_dimension_mean	-0.311631	-0.076437	-0.261477	-0.283110	0.584792	0.565369	0.336783	0.166917	0.47
radius_se	0.679090	0.275869	0.691765	0.732562	0.301467	0.497473	0.631925	0.698050	0.30
texture_se	-0.097317	0.386358	-0.086761	-0.066280	0.068406	0.046205	0.076218	0.021480	0.12
perimeter_se	0.674172	0.281673	0.693135	0.726628	0.296092	0.548905	0.660391	0.710650	0.31
area_se	0.735864	0.259845	0.744983	0.800086	0.246552	0.455653	0.617427	0.690299	0.22
smoothness_se	-0.222600	0.006614	-0.202694	-0.166777	0.332375	0.135299	0.098564	0.027653	0.18
compactness_se	0.206000	0.191975	0.250744	0.212583	0.318943	0.738722	0.670279	0.490424	0.42
concavity_se	0.194204	0.143293	0.228082	0.207660	0.248396	0.570517	0.691270	0.439167	0.34
concave_points_se	0.376169	0.163851	0.407217	0.372320	0.380676	0.642262	0.683260	0.615634	0.39
symmetry_se	-0.104321	0.009127	-0.081629	-0.072497	0.200774	0.229977	0.178009	0.095351	0.44
fractal_dimension_se	-0.042641	0.054458	-0.005523	-0.019887	0.283607	0.507318	0.449301	0.257584	0.33
radius_worst	0.969539	0.352573	0.969476	0.962746	0.213120	0.535315	0.688236	0.830318	0.18
texture_worst	0.297008	0.912045	0.303038	0.287489	0.036072	0.248133	0.299879	0.292752	0.09
perimeter_worst	0.965137	0.358040	0.970387	0.959120	0.238853	0.590210	0.729565	0.855923	0.21
area_worst	0.941082	0.343546	0.941550	0.959213	0.206718	0.509604	0.675987	0.809630	0.17
smoothness_worst	0.119616	0.077503	0.150549	0.123523	0.805324	0.565541	0.448822	0.452753	0.42
compactness_worst	0.413463	0.277830	0.455774	0.390410	0.472468	0.865809	0.754968	0.667454	0.47

concavity_worst	0.526911	0.301025	0.563879	0.512606	0.434926	0.816275	0.884103	0.752399	0.43
concave_points_worst	0.744214	0.295316	0.771241	0.722017	0.503053	0.815573	0.861323	0.910155	0.43
symmetry_worst	0.163953	0.105008	0.189115	0.143570	0.394309	0.510223	0.409464	0.375744	0.69
fractal_dimension_worst	0.007066	0.119205	0.051019	0.003738	0.499316	0.687382	0.514930	0.368661	0.43
diagnosis	0.730029	0.415185	0.742636	0.708984	0.358560	0.596534	0.696360	0.776614	0.33

31 rows × 31 columns

<Axes: >



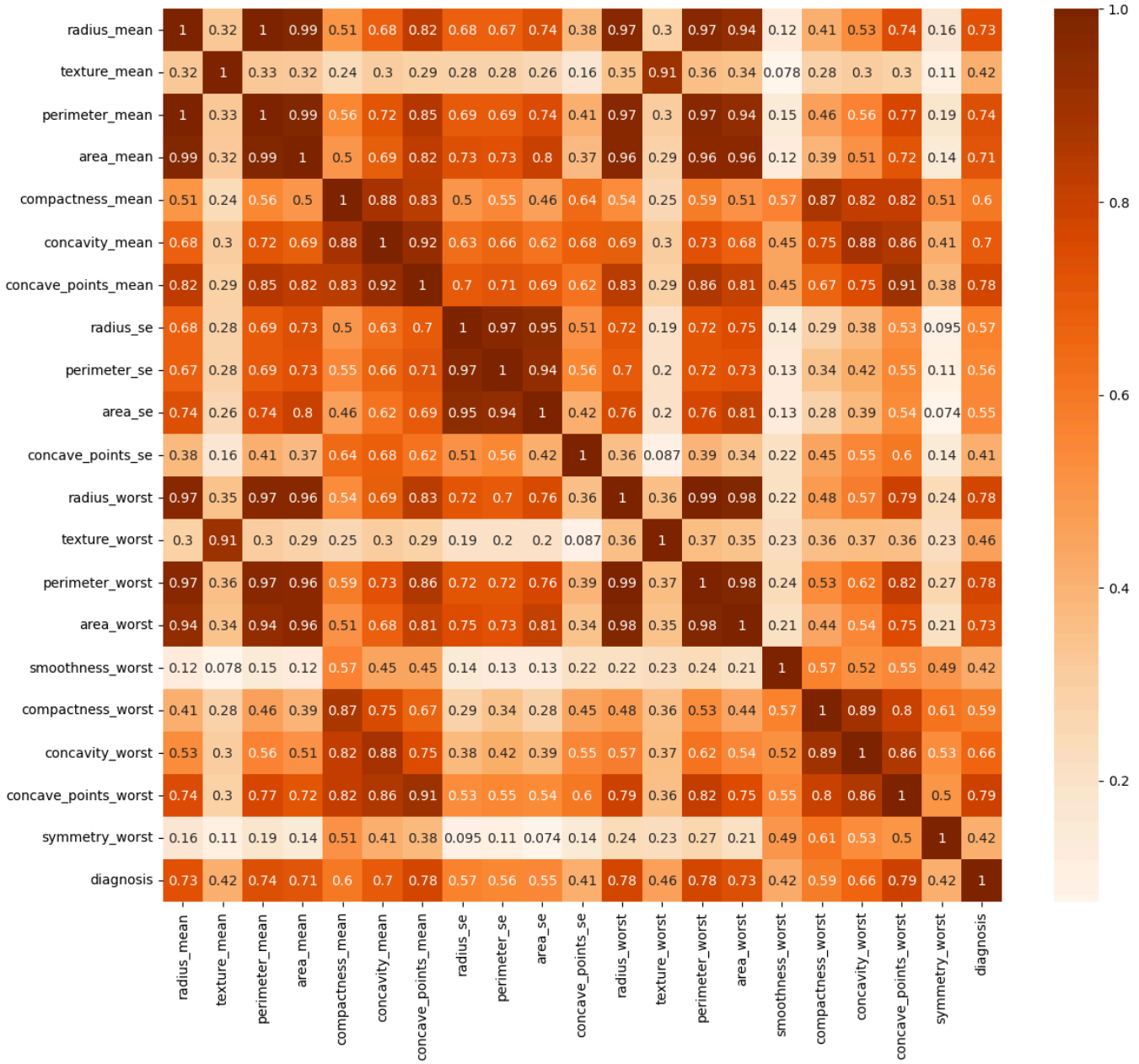
Selecting the columns with a high correlation with the diagnosis column:

- Columns with correaltion more than 40%:

	radius_mean	texture_mean	perimeter_mean	area_mean	compactness_mean	concavity_mean	concave_points_mean	radius_se	perimeter_se	area_se	...	radius_worst
0	17.99	10.38	122.80	1001.0	0.27760	0.3001	0.14710	1.0950	8.589	153.40	...	25.38
1	20.57	17.77	132.90	1326.0	0.07864	0.0869	0.07017	0.5435	3.398	74.08	...	24.99
2	19.69	21.25	130.00	1203.0	0.15990	0.1974	0.12790	0.7456	4.585	94.03	...	23.57
3	11.42	20.38	77.58	386.1	0.28390	0.2414	0.10520	0.4956	3.445	27.23	...	14.91
4	20.29	14.34	135.10	1297.0	0.13280	0.1980	0.10430	0.7572	5.438	94.44	...	22.54

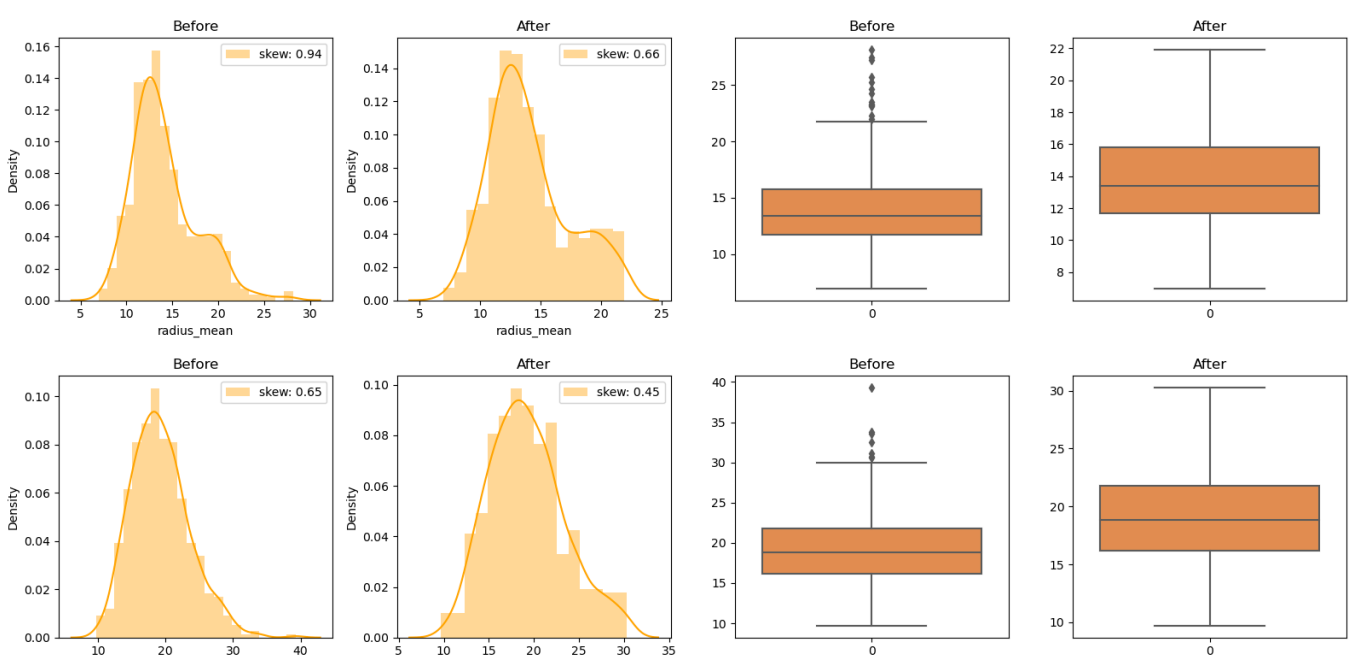
5 rows × 21 columns

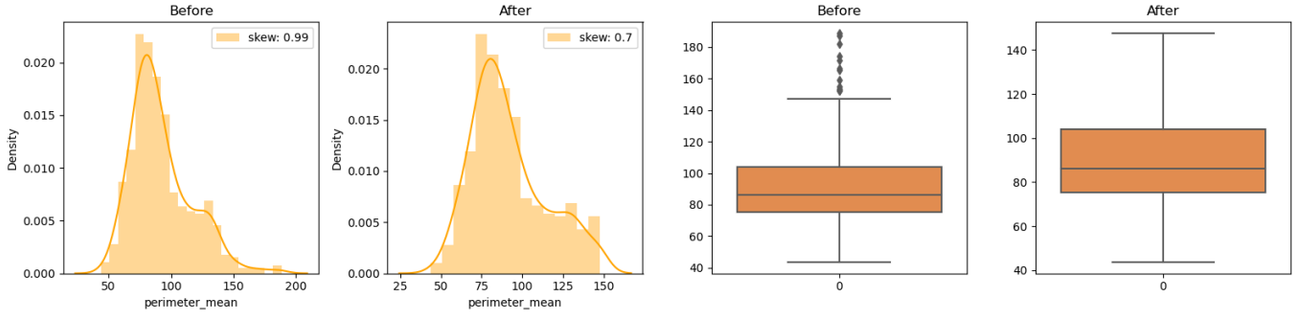
<Axes: >

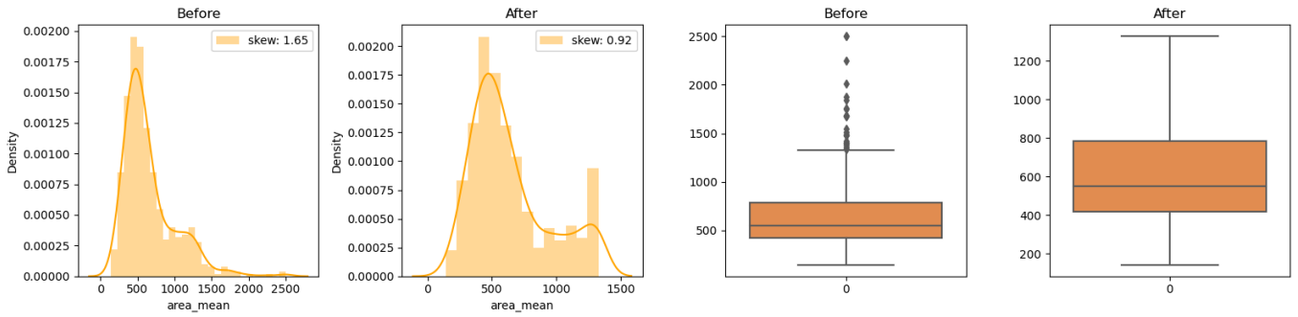


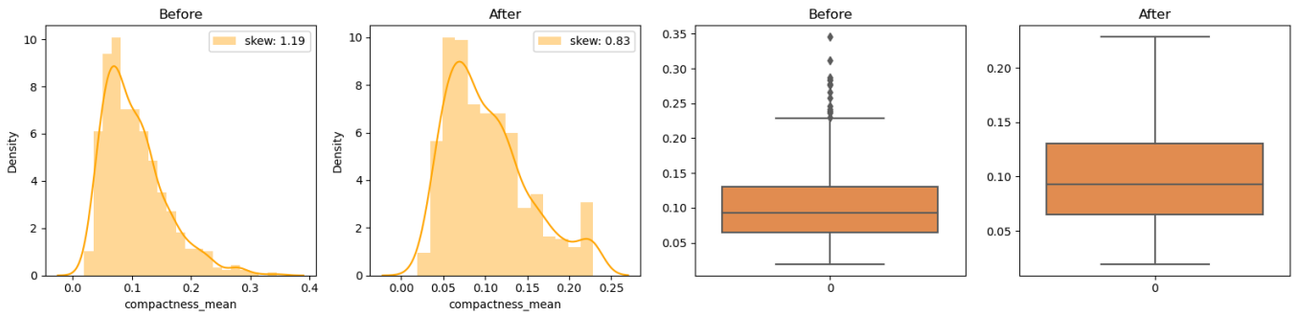
Dealing with the outliers in the dataset:

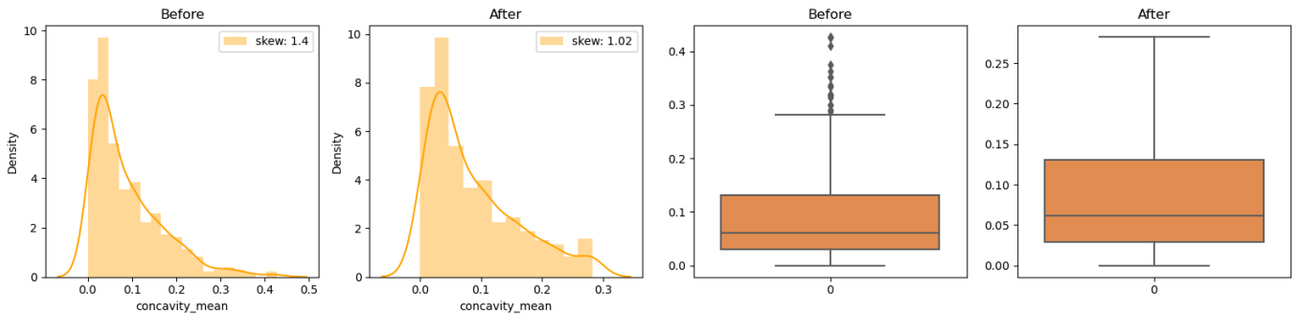
Before and after removing the outliers:

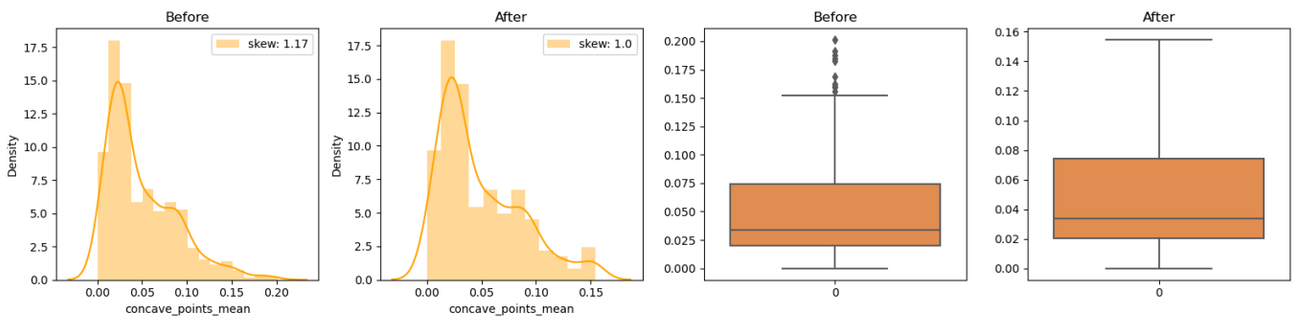


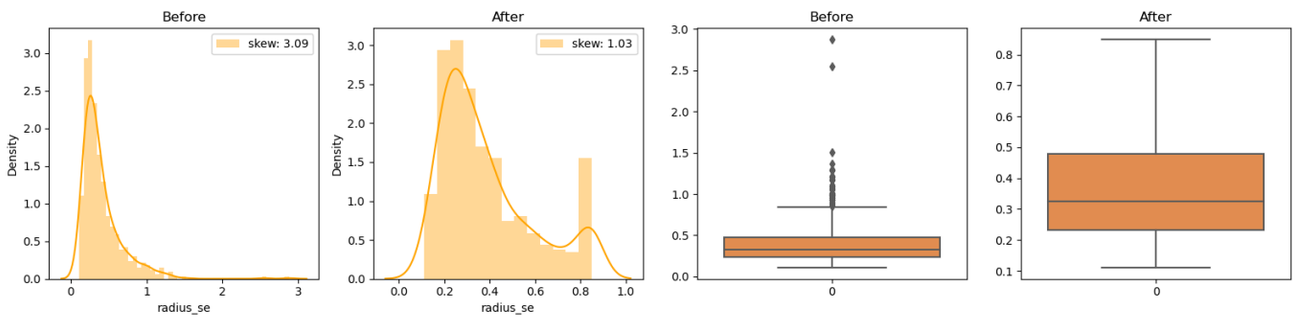

Before
After
Before
After
skew: 0.99
skew: 0.7
Density
perimeter_mean
perimeter_mean
0

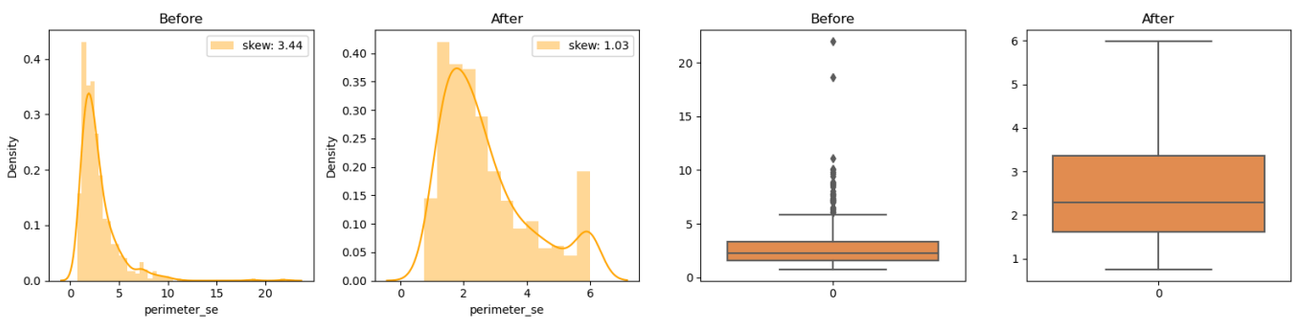

Before
After
Before
After
skew: 1.65
skew: 0.92
Density
area_mean
area_mean
0

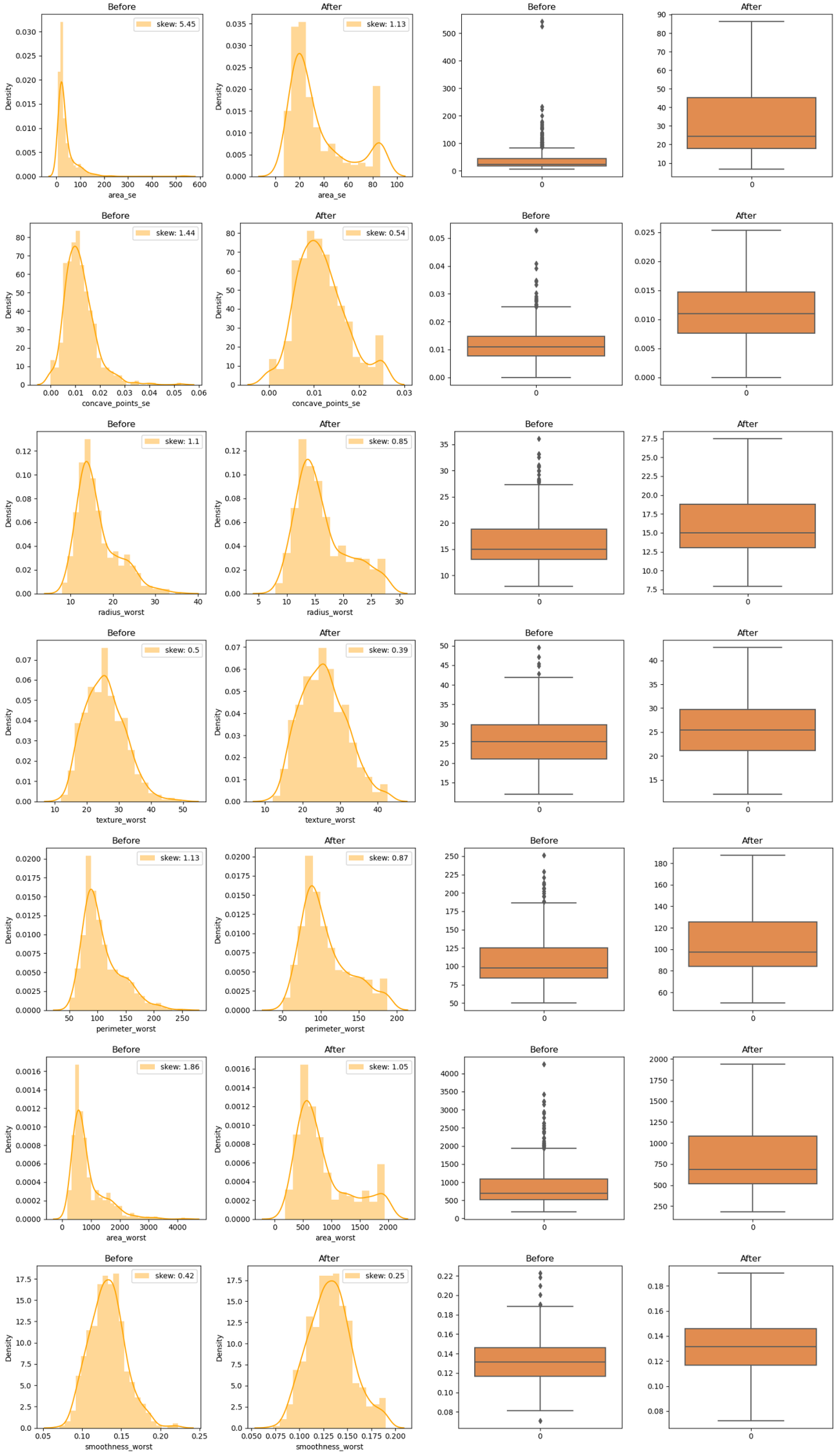

Before
After
Before
After
skew: 1.19
skew: 0.83
Density
compactness_mean
compactness_mean
0

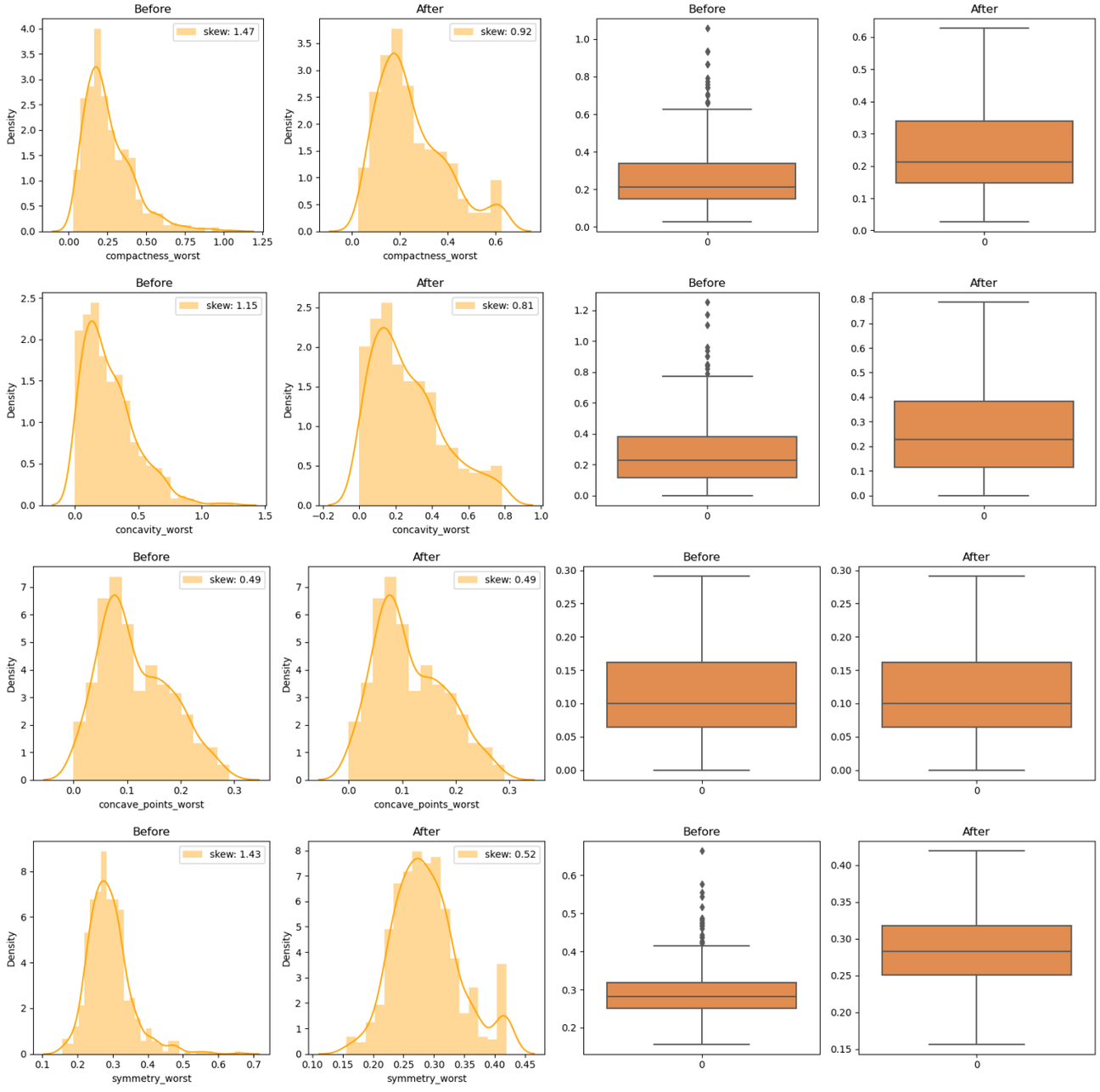

Before
After
Before
After
skew: 1.4
skew: 1.02
Density
concavity_mean
concavity_mean
0


Before
After
Before
After
skew: 1.17
skew: 1.0
Density
concave_points_mean
concave_points_mean
0


Before
After
Before
After
skew: 3.09
skew: 1.03
Density
radius_se
radius_se
0


Before
After
Before
After
skew: 3.44
skew: 1.03
Density
perimeter_se
perimeter_se
0


Before
After
skew: 5.45
skew: 1.13
area_se
skew: 1.44
skew: 0.54
concave_points_se
skew: 1.1
skew: 0.85
radius_worst
skew: 0.5
skew: 0.39
texture_worst
skew: 1.13
skew: 0.87
perimeter_worst
skew: 1.86
skew: 1.05
area_worst
skew: 0.42
skew: 0.25
smoothness_worst
Density



The dataset is ready for the ML Models:

Viewing the dataset:

	radius_mean	texture_mean	perimeter_mean	area_mean	compactness_mean	concavity_mean	concave_points_mean	radius_se	perimeter_se	area_se	...	radius_worst
0	17.99	10.38	122.80	1001.0	0.22862	0.28241	0.14710	0.84865	5.9835	86.20	...	25.38
1	20.57	17.77	132.90	1326.0	0.07864	0.08690	0.07017	0.54350	3.3980	74.08	...	24.99
2	19.69	21.25	130.00	1203.0	0.15990	0.19740	0.12790	0.74560	4.5850	86.20	...	23.57
3	11.42	20.38	77.58	386.1	0.22862	0.24140	0.10520	0.49560	3.4450	27.23	...	14.91
4	20.29	14.34	135.10	1297.0	0.13280	0.19800	0.10430	0.75720	5.4380	86.20	...	22.54

5 rows × 21 columns

Splitting the dataset into feature and target sets:

- **Feature set includes**: 'radius_mean', 'texture_mean', 'perimeter_mean', 'area_mean', 'compactness_mean', 'concavity_mean', 'concave_points_mean', 'radius_se', 'perimeter_se', 'area_se', 'concave_points_se', 'radius_worst', 'texture_worst', 'perimeter_worst', 'area_worst', 'smoothness_worst', 'compactness_worst', 'concavity_worst', 'concave_points_worst', 'symmetry_worst'.
- **Target set contains**: 'diagnosis'.

Splitting the dataset into:

- **Training set** (X_train, y_train)
- **Test set** (X_test, y_train)

Training multiple models on the "training set" and then fit the model:

- **Models**:
  - Gradient Boosting Classifier
  - Random Forest Classifier

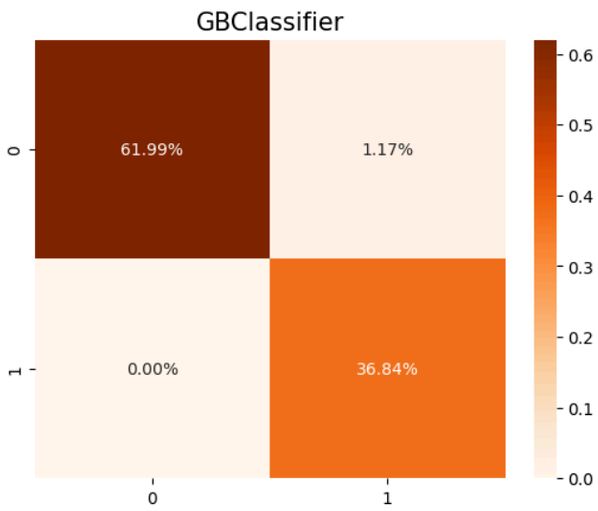
- Adaptive Boosting Classifier
- Logistic Regression
- KNeighbors Classifier
- Decision Tree Classifier
- Support Vector Classifier
- Linear Discriminant Analysis
- Quadratic Discriminant Analysis

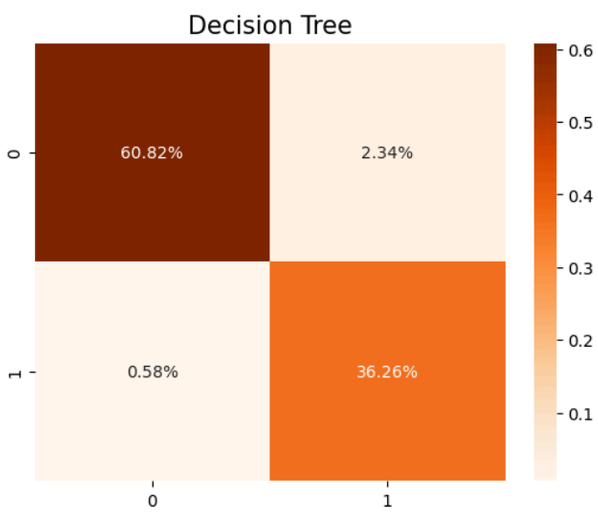
- Metrics:
  - Precision Score
  - Recall Score
  - F1 Score
  - Confusion Matrix

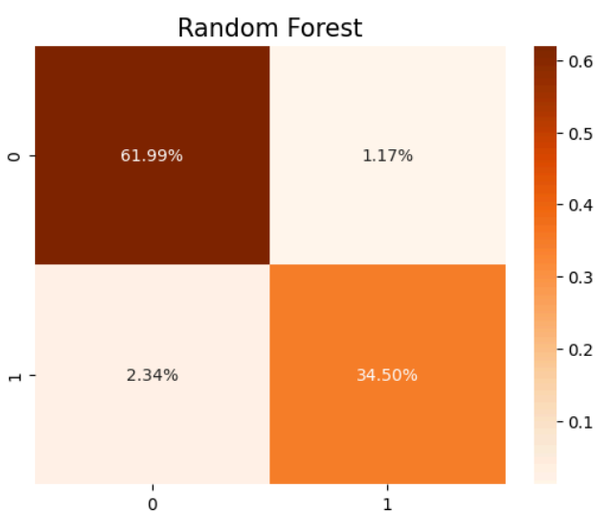
Evaluating the models with certain metrics:

Confusion Matrix outcomes:

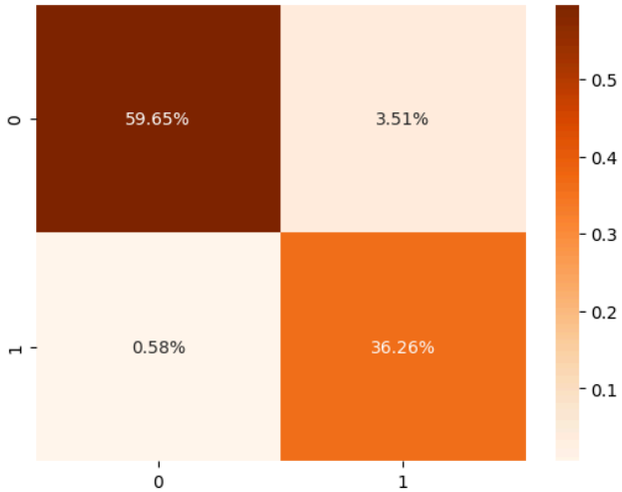
- **True Positive: TP**
  - We predicted positive and it's true.
- **True Negative: TN**
  - We predicted negative and it's true.
- **False Positive: FP (Type 1 Error)**
  - We predicted positive and it's false.
- **False Negative: FN (Type 2 Error)**
  - We predicted negative and it's false.



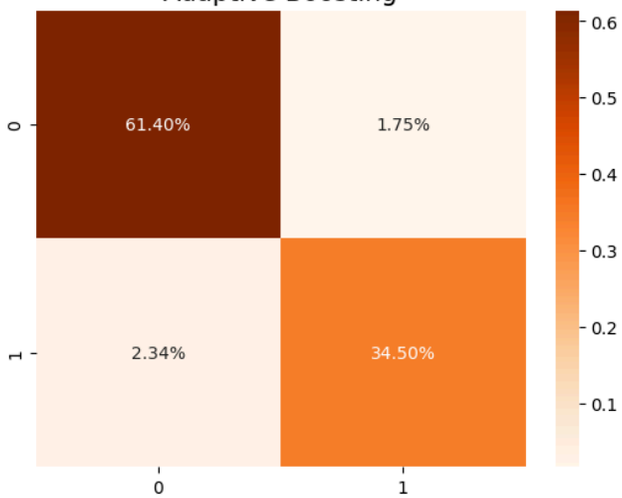




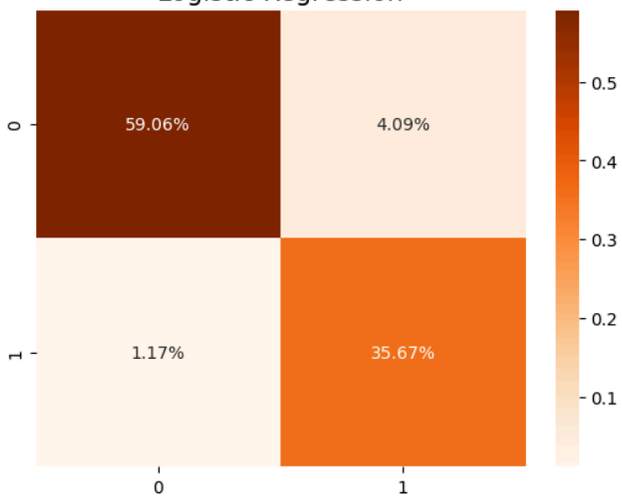
SVClassifier



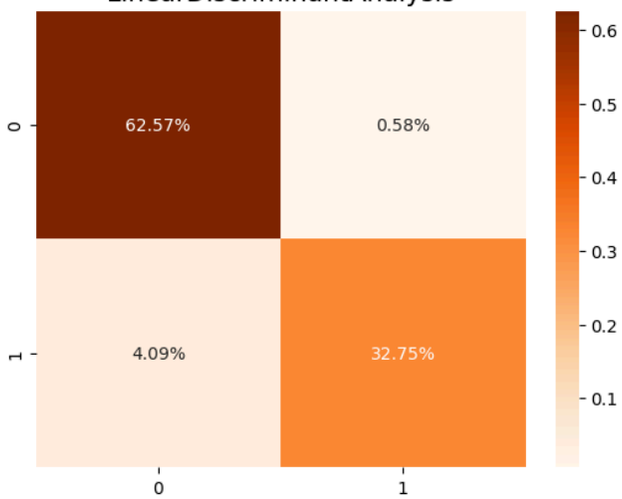
Adaptive Boosting

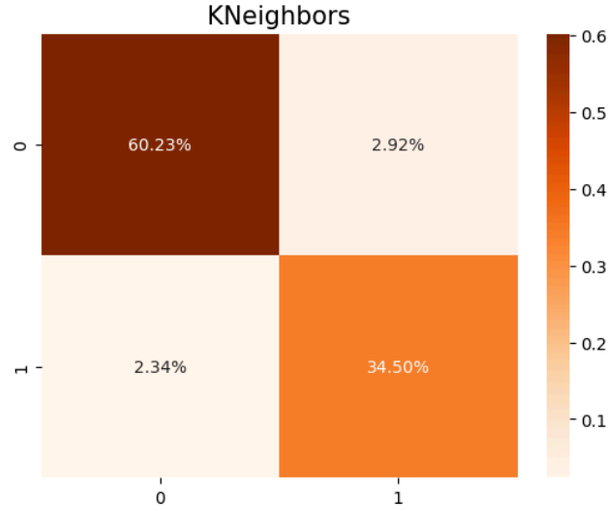


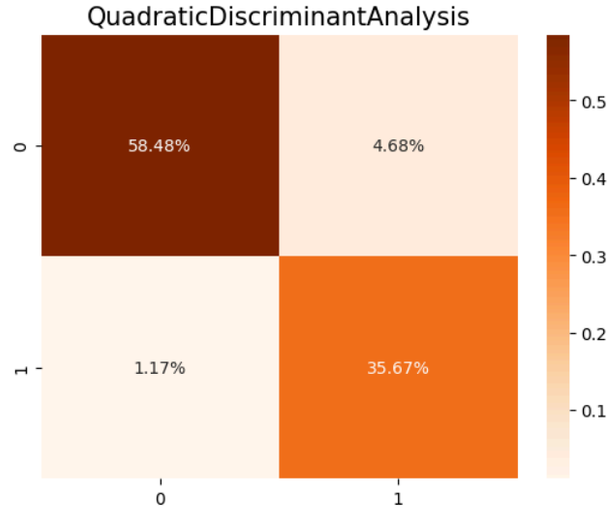
Logistic Regression



LinearDiscriminantAnalysis



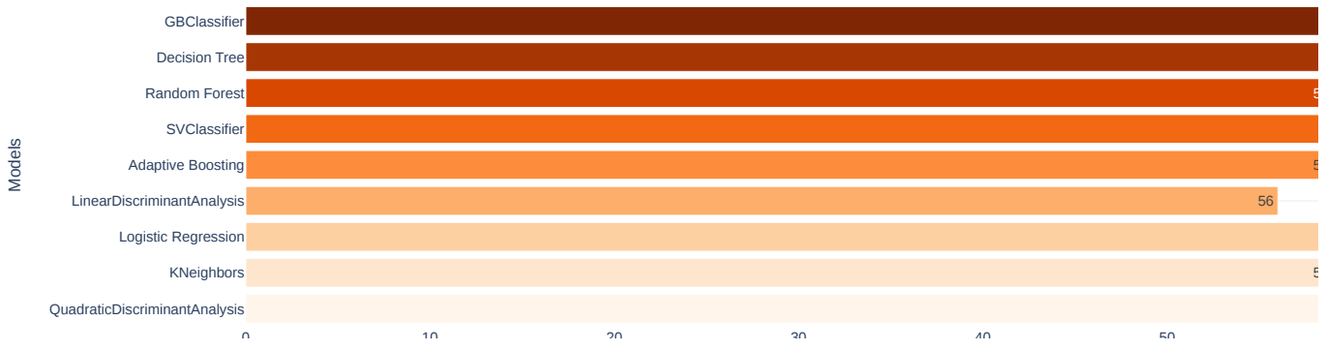




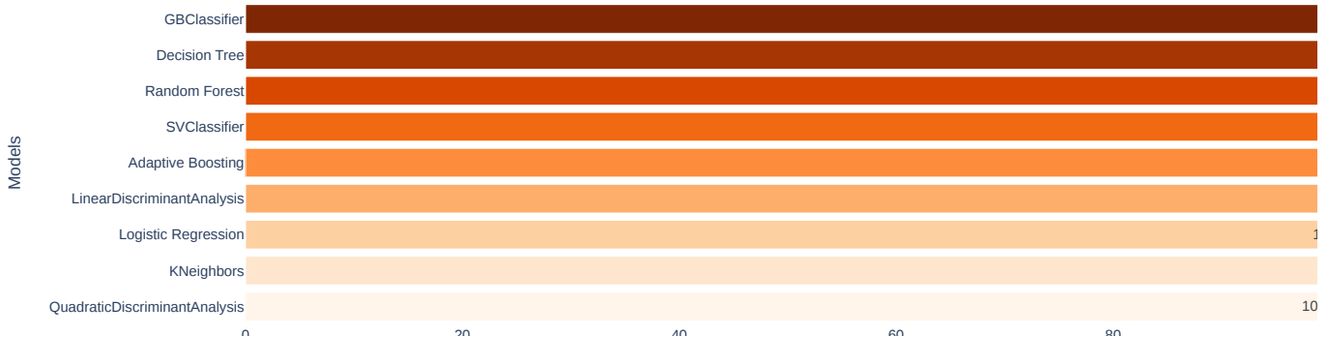
Summarizing the Confusion Matrix outcomes:

	TP	TN	FP	FN
GBClassifier	63	106	2	0
Decision Tree	62	104	4	1
Random Forest	59	106	2	4
SVClassifier	62	102	6	1
Adaptive Boosting	59	105	3	4
LinearDiscriminantAnalysis	56	107	1	7
Logistic Regression	61	101	7	2
KNeighbors	59	103	5	4
QuadraticDiscriminantAnalysis	61	100	8	2

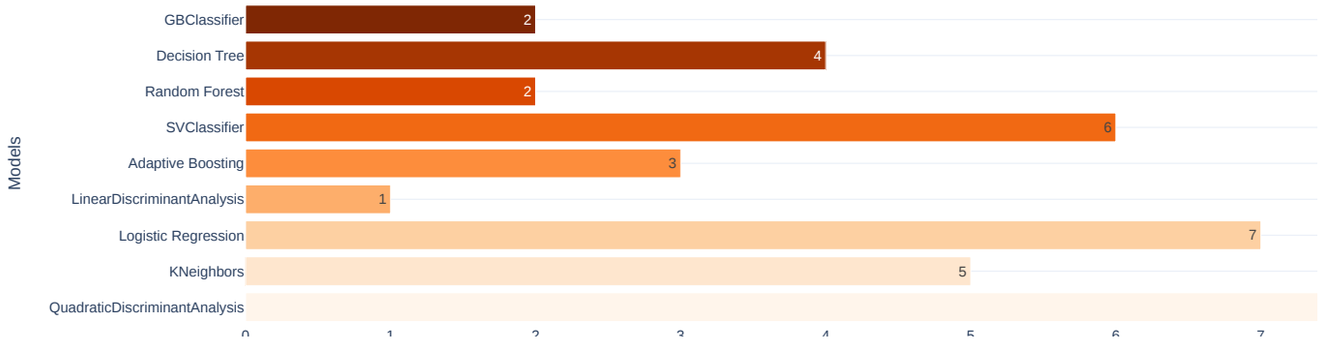
Confusion Matrix: True Positive of Each Model



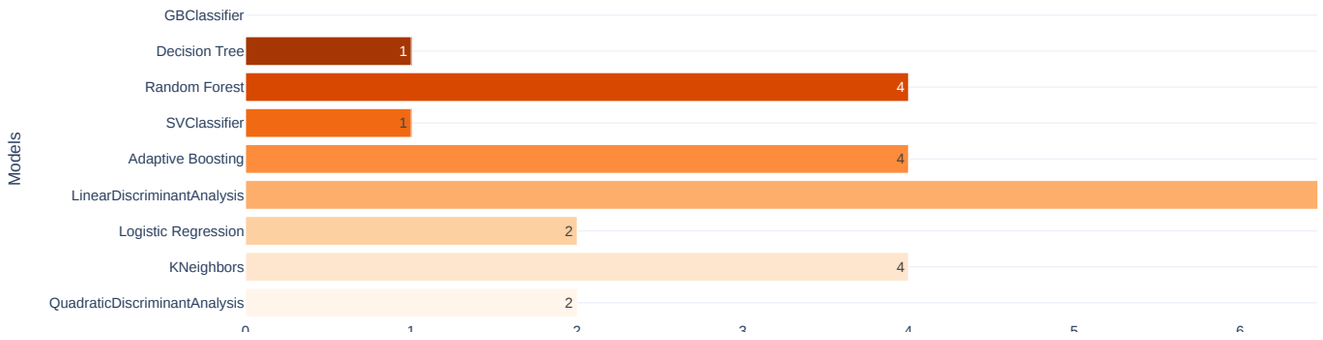
Confusion Matrix: True Negative of Each Model



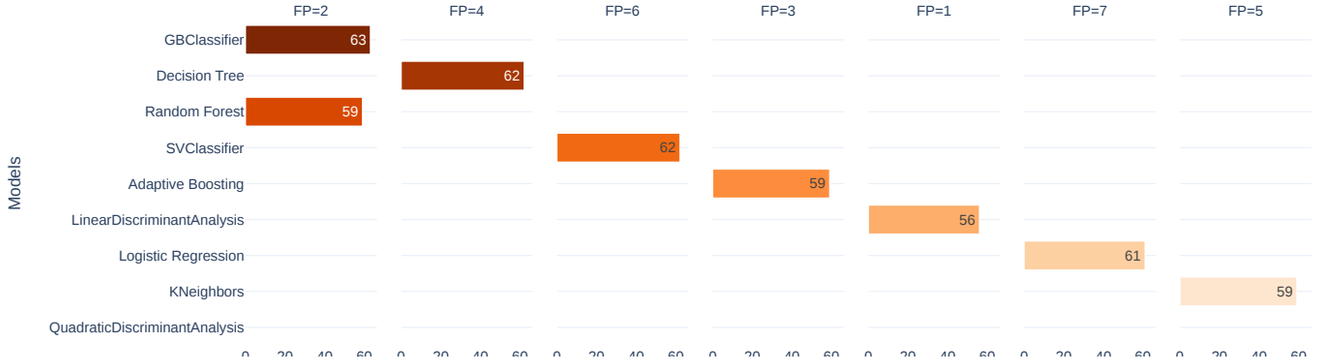
Confusion Matrix: False Positive (Type 1 Error) of Each Model



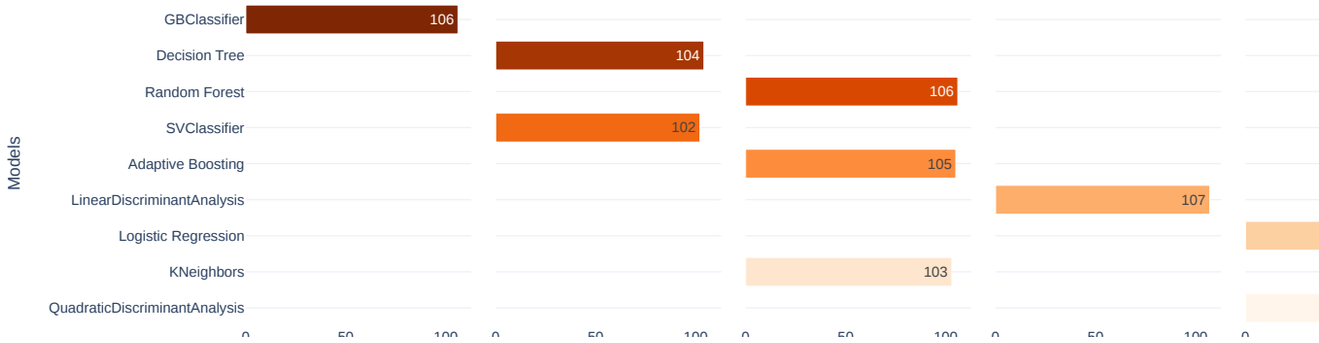
Confusion Matrix: False Negative (Type 2 Error) of Each Model

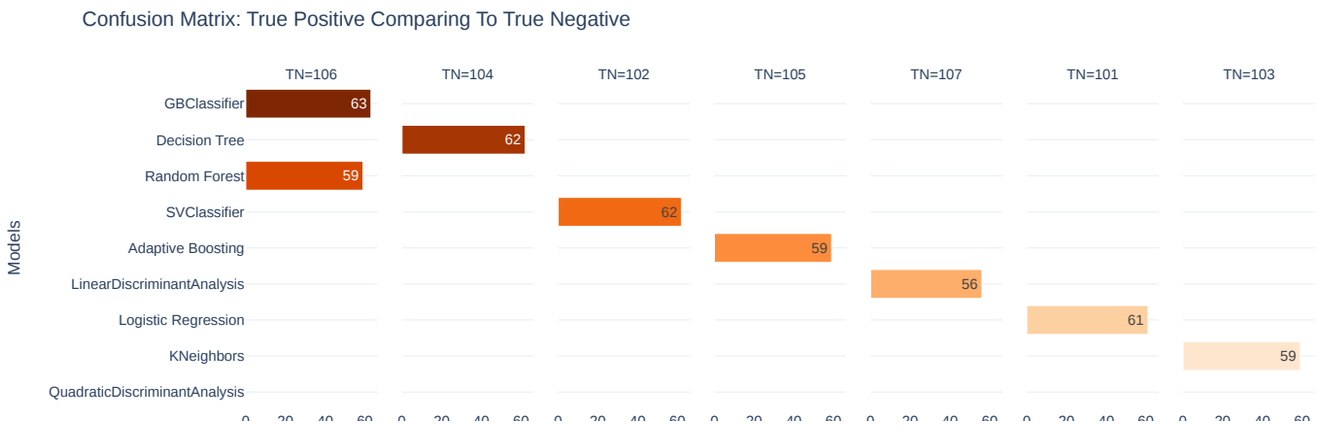


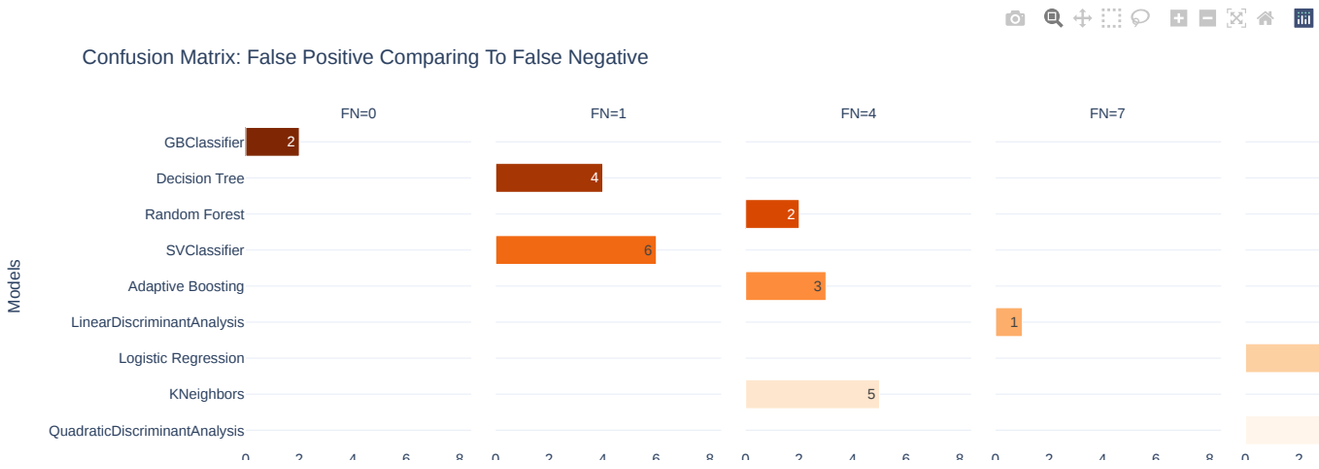
Confusion Matrix: True Positive Comparing To False Positive



Confusion Matrix: True Negative Comparing To False Negative

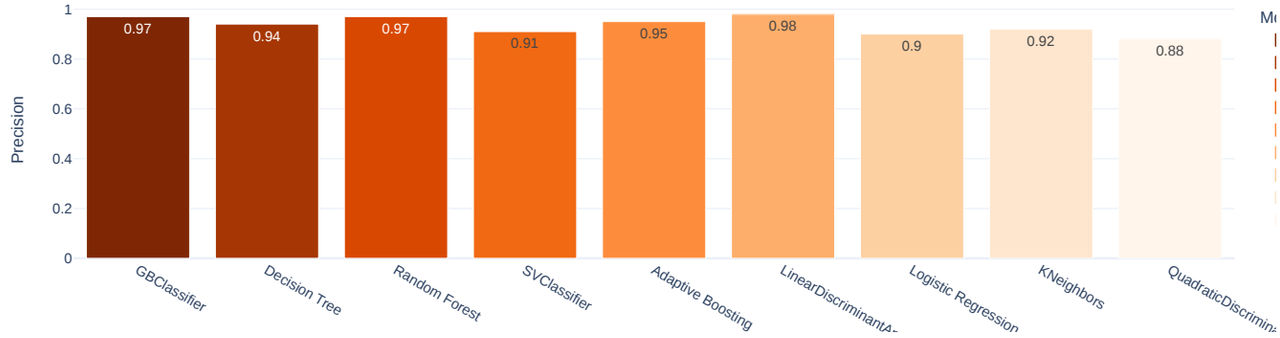


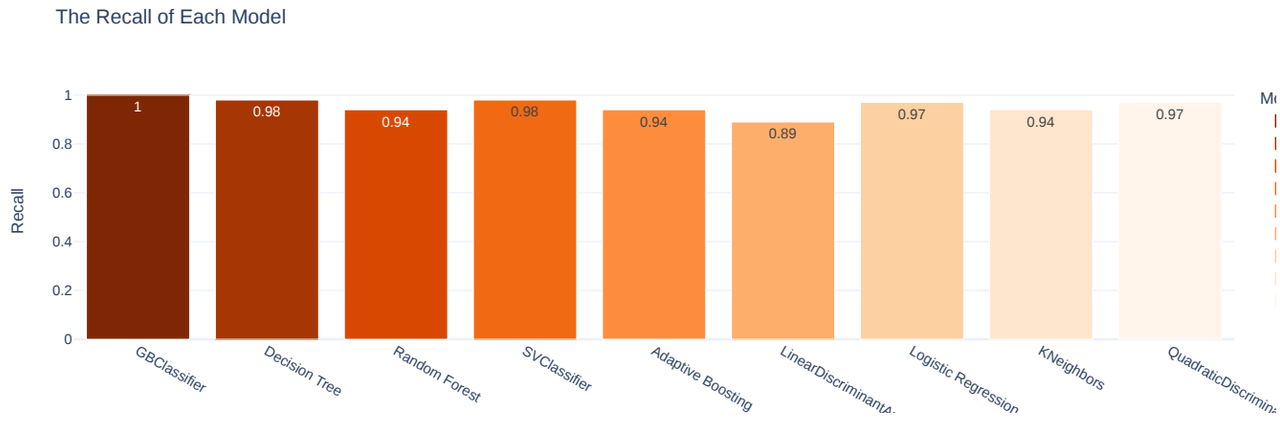




Precision and Recall of each model:

	Precision	Recall
GBClassifier	0.97	1.00
Decision Tree	0.94	0.98
Random Forest	0.97	0.94
SVClassifier	0.91	0.98
Adaptive Boosting	0.95	0.94
LinearDiscriminantAnalysis	0.98	0.89
Logistic Regression	0.90	0.97
KNeighbors	0.92	0.94
QuadraticDiscriminantAnalysis	0.88	0.97

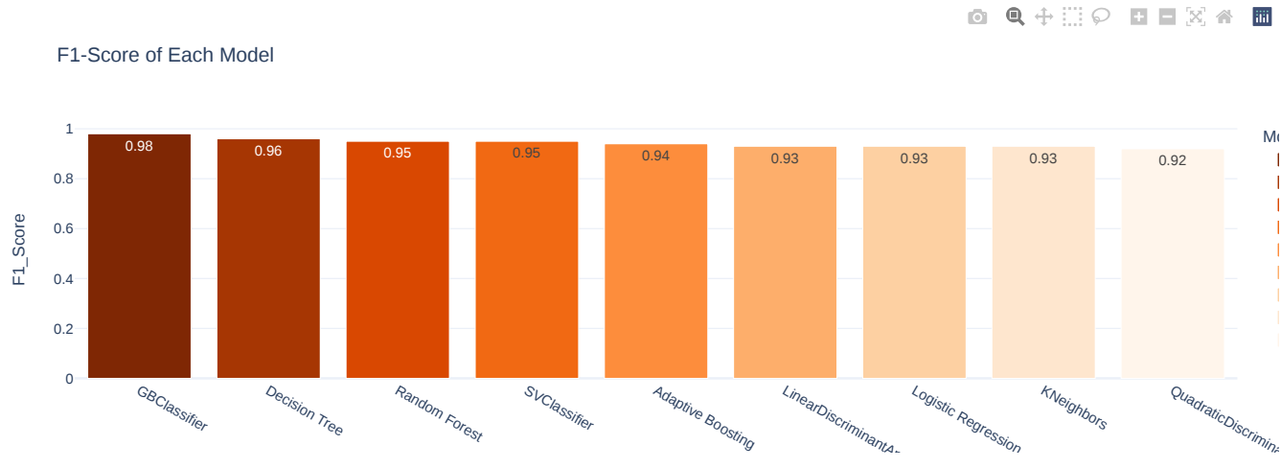




F1-Score which is also known as the balanced F-score or F-measure:

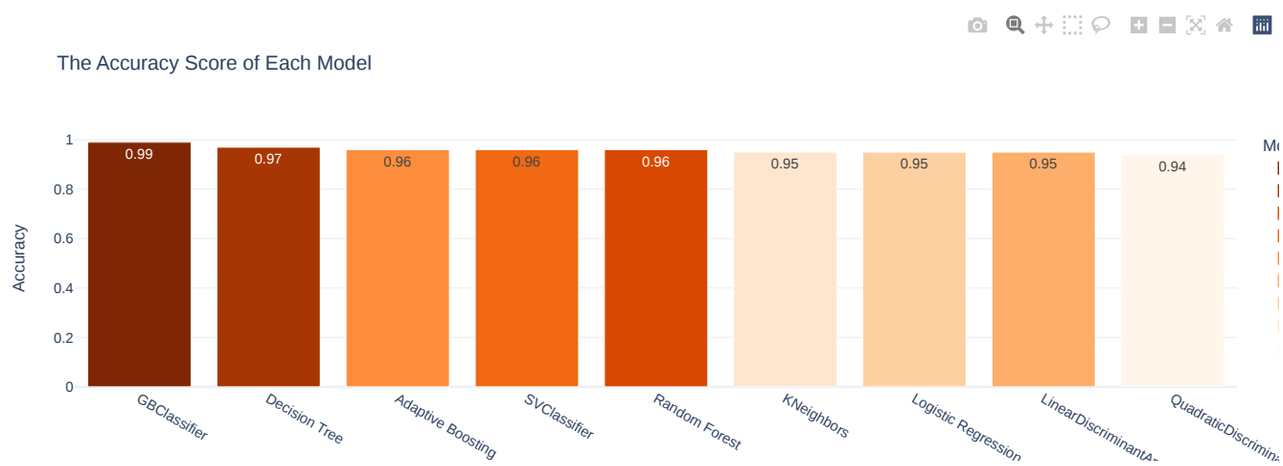
- **F-Score** is the **harmonic mean** of precision and recall.
- **F-Score** is utilized to deal with the difficultly of **comparing** two models with high precision or low recall or vice versa.

	F1_Score
GBClassifier	0.98
Decision Tree	0.96
Random Forest	0.95
SVClassifier	0.95
Adaptive Boosting	0.94
LinearDiscriminantAnalysis	0.93
Logistic Regression	0.93
KNeighbors	0.93
QuadraticDiscriminantAnalysis	0.92



The Accuracy score of each model:

	Accuracy
GBClassifier	0.99
Decision Tree	0.97
Random Forest	0.96
SVClassifier	0.96
Adaptive Boosting	0.96
LinearDiscriminantAnalysis	0.95
Logistic Regression	0.95
KNeighbors	0.95
QuadraticDiscriminantAnalysis	0.94



Combining all the metrics together:

	Accuracy	Precision	Recall	F1_Score	TP	TN	FP	FN
GBClassifier	0.99	0.97	1.00	0.98	63	106	2	0
Decision Tree	0.97	0.94	0.98	0.96	62	104	4	1
Random Forest	0.96	0.97	0.94	0.95	59	106	2	4
SVClassifier	0.96	0.91	0.98	0.95	62	102	6	1
Adaptive Boosting	0.96	0.95	0.94	0.94	59	105	3	4
LinearDiscriminantAnalysis	0.95	0.98	0.89	0.93	56	107	1	7
Logistic Regression	0.95	0.90	0.97	0.93	61	101	7	2
KNeighbors	0.95	0.92	0.94	0.93	59	103	5	4
QuadraticDiscriminantAnalysis	0.94	0.88	0.97	0.92	61	100	8	2

From the observations:

After **training**, **testing**, and **evaluating** the models, **all the models** achieved and relized excellent accuracy scores since they were **more than 90%**. The **best accuracy score** was obtained by **Gradient Boosting Classifier** which was **99%**.

Lastly, comparing the prediction results with the actul ones including in the test set:

- Using Gradient Boosting Classifier with 99% accuracy

	Y_Prediction	Y_Test_Set
0	1	1
1	0	0
2	0	0
3	0	0
4	0	0
5	0	0
6	0	0
7	0	0
8	0	0
9	0	0
10	0	0
11	0	0
12	0	0
13	0	0
14	0	0

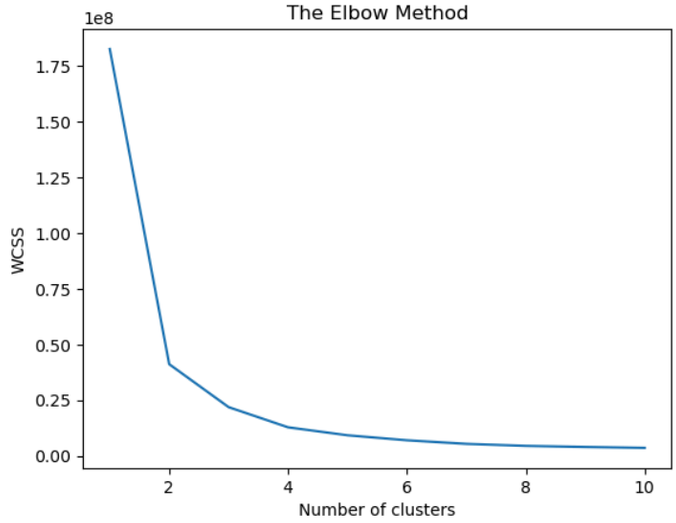
Unsupervised Machine Learning (Clustering):

K-Means Clustering:

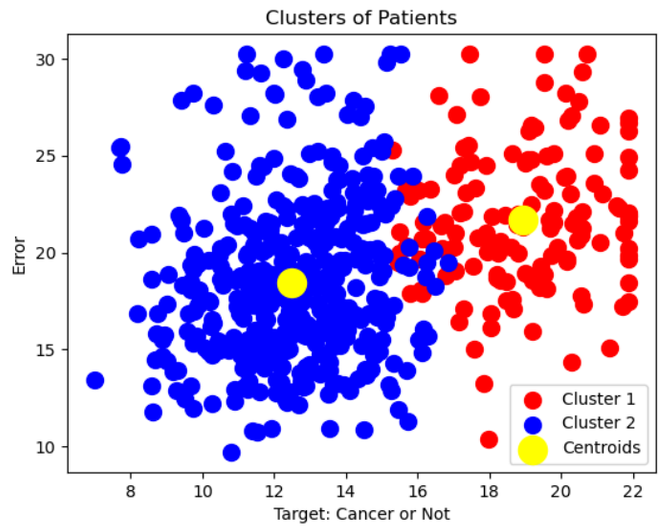
Splitting the dataset into:

- **Training set** (X_train, y_train)
- **Test set** (X_test, y_train)

Identifying The Number of Clusters:



Ploting The Result of The Clustering:



Most Common Metrics for Clustering Models:

Davies-Bouldin Index:

```
0.4377786031768084
```

Mutual Information Score:

```
0.3028781228178782
```

Normalized Mutual Information Score:

```
0.4980364328681648
```

Adjusted Mutual Information Score:

```
0.4973069621902457
```

Adjusted Rand Index:

```
0.5337690933348298
```

Rand Index:

```
0.7707245228842298
```

Silhouette Score aka Silhouette Coefficient:

```
0.7060581296125275
```

Fowlkes Mallows Score:

```
0.8055498414192741
```

V-Measure Score:

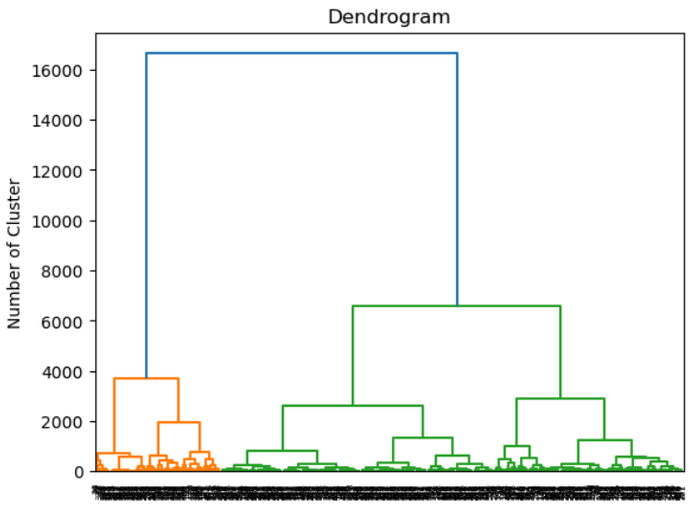
```
0.49803643286816485
```

Hierachical Clustering:

Splitting the dataset into:

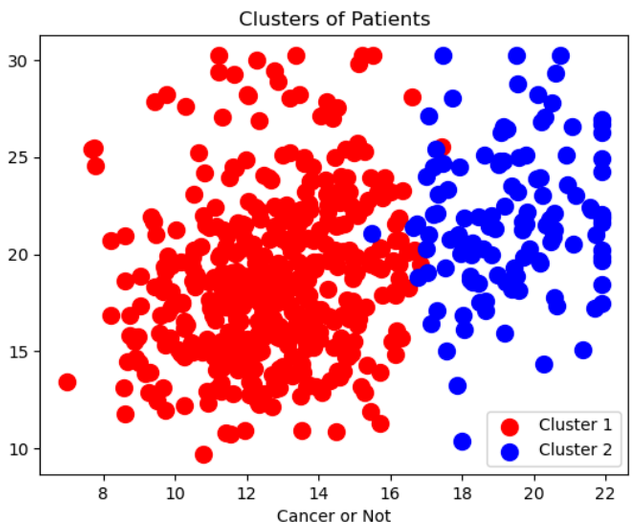
- **Training set** (X_train, y_train)
- **Test set** (X_test, y_train)

Using the dendrogram to find the optimal number of clusters



Training the Hierarchical Clustering model on the dataset

Visualising the clusters



Most Common Metrics for Clustering Models:

Davies-Bouldin Index:

```
0.4089704251999372
```

Mutual Information Score:

```
0.25037060976914904
```

Normalized Mutual Information Score:

```
0.4251611983264271
```

Adjusted Mutual Information Score:

```
0.4242979885293028
```

Adjusted Rand Index:

```
0.44046024502621134
```

Rand Index:

```
0.7260575756825663
```

Silhouette Score aka Silhouette Coefficient:

```
0.7017688156922348
```

Fowlkes Mallows Score:

```
0.7757910094078465
```

V-Measure Score:

```
0.7757910094078465
```

Deep Learning Model:

```
WARNING:tensorflow:From D:\Anaconda\Lib\site-packages\keras\src\losses.py:2976: The name tf.losses.sparse_softmax_cross_entropy is deprecate
d. Please use tf.compat.v1.losses.sparse_softmax_cross_entropy instead.
```

Splitting the dataset into:

- **Training set** (X_train, y_train)
- **Test set** (X_test, y_train)

Feature Scaling

Train Set After Scaling:

```
array([[-0.78, -1.11, -0.77, ..., -0.04, -0.62,  0.14],
       [-1.08, -0.14, -1.08, ..., -0.97, -0.76, -1.19],
       [-0.55, -0.3 , -0.59, ..., -0.39, -0.5 ,  0.45],
       ...,
       [-1.39, -0.21, -1.39, ..., -0.81, -0.99, -0.75],
       [-1.31, -0.23, -1.35, ..., -1.42, -1.76, -1.75],
       [-0.77,  1.16, -0.75, ...,  0.51, -0.28, -1.38]])
```

Test Set After Scaling:

```
array([[-0.21,  0.33, -0.14, ...,  1.29,  1.37,  1.32],
       [-0.27,  1.48, -0.33, ..., -0.68, -0.84, -0.77],
       [-0.02, -0.85, -0.09, ..., -1.09, -0.5 , -1.33],
       ...,
       [-0.12, -0.12, -0.13, ..., -0.39,  0.2 , -0.03],
       [-0.24,  0.59, -0.28, ..., -0.78, -0.9 , -0.42],
       [-1.42,  0.68, -1.42, ..., -0.99, -1.37, -0.79]])
```

Part 2 - Building the ANN

Initializing the ANN

```
WARNING:tensorflow:From D:\Anaconda\Lib\site-packages\keras\src\backend.py:873: The name tf.get_default_graph is deprecated. Please use tf.c
ompat.v1.get_default_graph instead.
```

Adding the input layer and the first hidden layer:

- **Unit**: 6
- **Activation**: 'relu'

Adding the second hidden layer:

- **Unit**: 6
- **Activation**: 'relu'

Adding the output layer:

- **Unit**: 1
- **Activation**: 'sigmoid'

Part 3 - Training the ANN

Compiling the ANN

```
WARNING:tensorflow:From D:\Anaconda\Lib\site-packages\keras\src\optimizers\__init__.py:309: The name tf.train.Optimizer is deprecated. Pleas
e use tf.compat.v1.train.Optimizer instead.
```

Training the ANN on the Training set

```
Epoch 1/100
WARNING:tensorflow:From D:\Anaconda\Lib\site-packages\keras\src\utils\tf_utils.py:492: The name tf.ragged.RaggedTensorValue is deprecated. P
lease use tf.compat.v1.ragged.RaggedTensorValue instead.

WARNING:tensorflow:From D:\Anaconda\Lib\site-packages\keras\src\engine\base_layer_utils.py:384: The name tf.executing_eagerly_outside_functi
ons is deprecated. Please use tf.compat.v1.executing_eagerly_outside_functions instead.
```

```
13/13 [==============================] - 2s 4ms/step - loss: 0.5963 - accuracy: 0.7035
Epoch 2/100
13/13 [==============================] - 0s 3ms/step - loss: 0.5415 - accuracy: 0.8317
Epoch 3/100
13/13 [==============================] - 0s 3ms/step - loss: 0.4952 - accuracy: 0.9095
Epoch 4/100
13/13 [==============================] - 0s 3ms/step - loss: 0.4529 - accuracy: 0.9296
Epoch 5/100
13/13 [==============================] - 0s 3ms/step - loss: 0.4089 - accuracy: 0.9397
Epoch 6/100
13/13 [==============================] - 0s 3ms/step - loss: 0.3659 - accuracy: 0.9523
Epoch 7/100
13/13 [==============================] - 0s 3ms/step - loss: 0.3240 - accuracy: 0.9598
Epoch 8/100
13/13 [==============================] - 0s 3ms/step - loss: 0.2865 - accuracy: 0.9598
Epoch 9/100
13/13 [==============================] - 0s 4ms/step - loss: 0.2538 - accuracy: 0.9623
Epoch 10/100
13/13 [==============================] - 0s 4ms/step - loss: 0.2261 - accuracy: 0.9623
Epoch 11/100
13/13 [==============================] - 0s 4ms/step - loss: 0.2031 - accuracy: 0.9648
Epoch 12/100
13/13 [==============================] - 0s 3ms/step - loss: 0.1845 - accuracy: 0.9648
Epoch 13/100
13/13 [==============================] - 0s 4ms/step - loss: 0.1693 - accuracy: 0.9648
Epoch 14/100
13/13 [==============================] - 0s 4ms/step - loss: 0.1571 - accuracy: 0.9648
Epoch 15/100
13/13 [==============================] - 0s 3ms/step - loss: 0.1472 - accuracy: 0.9648
Epoch 16/100
13/13 [==============================] - 0s 3ms/step - loss: 0.1389 - accuracy: 0.9648
Epoch 17/100
13/13 [==============================] - 0s 4ms/step - loss: 0.1320 - accuracy: 0.9648
Epoch 18/100
13/13 [==============================] - 0s 4ms/step - loss: 0.1268 - accuracy: 0.9648
Epoch 19/100
13/13 [==============================] - 0s 4ms/step - loss: 0.1216 - accuracy: 0.9673
Epoch 20/100
13/13 [==============================] - 0s 4ms/step - loss: 0.1176 - accuracy: 0.9673
Epoch 21/100
13/13 [==============================] - 0s 3ms/step - loss: 0.1143 - accuracy: 0.9673
Epoch 22/100
13/13 [==============================] - 0s 4ms/step - loss: 0.1112 - accuracy: 0.9673
Epoch 23/100
13/13 [==============================] - 0s 4ms/step - loss: 0.1086 - accuracy: 0.9698
Epoch 24/100
13/13 [==============================] - 0s 3ms/step - loss: 0.1062 - accuracy: 0.9698
Epoch 25/100
13/13 [==============================] - 0s 3ms/step - loss: 0.1042 - accuracy: 0.9698
Epoch 26/100
13/13 [==============================] - 0s 3ms/step - loss: 0.1021 - accuracy: 0.9724
Epoch 27/100
13/13 [==============================] - 0s 4ms/step - loss: 0.1004 - accuracy: 0.9724
Epoch 28/100
13/13 [==============================] - 0s 8ms/step - loss: 0.0984 - accuracy: 0.9724
Epoch 29/100
13/13 [==============================] - 0s 4ms/step - loss: 0.0968 - accuracy: 0.9724
Epoch 30/100
13/13 [==============================] - 0s 4ms/step - loss: 0.0954 - accuracy: 0.9724
Epoch 31/100
13/13 [==============================] - 0s 3ms/step - loss: 0.0941 - accuracy: 0.9724
Epoch 32/100
13/13 [==============================] - 0s 3ms/step - loss: 0.0931 - accuracy: 0.9724
Epoch 33/100
13/13 [==============================] - 0s 3ms/step - loss: 0.0919 - accuracy: 0.9724
Epoch 34/100
13/13 [==============================] - 0s 3ms/step - loss: 0.0908 - accuracy: 0.9724
Epoch 35/100
13/13 [==============================] - 0s 3ms/step - loss: 0.0898 - accuracy: 0.9749
Epoch 36/100
13/13 [==============================] - 0s 4ms/step - loss: 0.0888 - accuracy: 0.9749
Epoch 37/100
13/13 [==============================] - 0s 4ms/step - loss: 0.0878 - accuracy: 0.9749
Epoch 38/100
13/13 [==============================] - 0s 4ms/step - loss: 0.0870 - accuracy: 0.9774
Epoch 39/100
13/13 [==============================] - 0s 4ms/step - loss: 0.0861 - accuracy: 0.9749
Epoch 40/100
13/13 [==============================] - 0s 4ms/step - loss: 0.0854 - accuracy: 0.9749
Epoch 41/100
13/13 [==============================] - 0s 3ms/step - loss: 0.0846 - accuracy: 0.9749
Epoch 42/100
13/13 [==============================] - 0s 3ms/step - loss: 0.0839 - accuracy: 0.9749
Epoch 43/100
13/13 [==============================] - 0s 3ms/step - loss: 0.0832 - accuracy: 0.9774
Epoch 44/100
13/13 [==============================] - 0s 3ms/step - loss: 0.0825 - accuracy: 0.9774
Epoch 45/100
13/13 [==============================] - 0s 3ms/step - loss: 0.0816 - accuracy: 0.9774
Epoch 46/100
13/13 [==============================] - 0s 3ms/step - loss: 0.0808 - accuracy: 0.9774
Epoch 47/100
13/13 [==============================] - 0s 3ms/step - loss: 0.0801 - accuracy: 0.9774
Epoch 48/100
13/13 [==============================] - 0s 3ms/step - loss: 0.0793 - accuracy: 0.9774
Epoch 49/100
13/13 [==============================] - 0s 3ms/step - loss: 0.0786 - accuracy: 0.9774
Epoch 50/100
13/13 [==============================] - 0s 3ms/step - loss: 0.0779 - accuracy: 0.9774
Epoch 51/100
13/13 [==============================] - 0s 3ms/step - loss: 0.0774 - accuracy: 0.9774
Epoch 52/100
```

```
13/13 [==============================] - 0s 3ms/step - loss: 0.0769 - accuracy: 0.9774
Epoch 53/100
13/13 [==============================] - 0s 3ms/step - loss: 0.0763 - accuracy: 0.9774
Epoch 54/100
13/13 [==============================] - 0s 3ms/step - loss: 0.0758 - accuracy: 0.9774
Epoch 55/100
13/13 [==============================] - 0s 3ms/step - loss: 0.0754 - accuracy: 0.9774
Epoch 56/100
13/13 [==============================] - 0s 3ms/step - loss: 0.0749 - accuracy: 0.9774
Epoch 57/100
13/13 [==============================] - 0s 3ms/step - loss: 0.0744 - accuracy: 0.9774
Epoch 58/100
13/13 [==============================] - 0s 3ms/step - loss: 0.0740 - accuracy: 0.9774
Epoch 59/100
13/13 [==============================] - 0s 4ms/step - loss: 0.0735 - accuracy: 0.9774
Epoch 60/100
13/13 [==============================] - 0s 3ms/step - loss: 0.0731 - accuracy: 0.9774
Epoch 61/100
13/13 [==============================] - 0s 3ms/step - loss: 0.0728 - accuracy: 0.9774
Epoch 62/100
13/13 [==============================] - 0s 3ms/step - loss: 0.0724 - accuracy: 0.9774
Epoch 63/100
13/13 [==============================] - 0s 3ms/step - loss: 0.0719 - accuracy: 0.9774
Epoch 64/100
13/13 [==============================] - 0s 3ms/step - loss: 0.0715 - accuracy: 0.9774
Epoch 65/100
13/13 [==============================] - 0s 3ms/step - loss: 0.0713 - accuracy: 0.9774
Epoch 66/100
13/13 [==============================] - 0s 3ms/step - loss: 0.0708 - accuracy: 0.9799
Epoch 67/100
13/13 [==============================] - 0s 3ms/step - loss: 0.0705 - accuracy: 0.9799
Epoch 68/100
13/13 [==============================] - 0s 4ms/step - loss: 0.0702 - accuracy: 0.9799
Epoch 69/100
13/13 [==============================] - 0s 4ms/step - loss: 0.0699 - accuracy: 0.9799
Epoch 70/100
13/13 [==============================] - 0s 3ms/step - loss: 0.0696 - accuracy: 0.9799
Epoch 71/100
13/13 [==============================] - 0s 3ms/step - loss: 0.0692 - accuracy: 0.9799
Epoch 72/100
13/13 [==============================] - 0s 4ms/step - loss: 0.0689 - accuracy: 0.9799
Epoch 73/100
13/13 [==============================] - 0s 4ms/step - loss: 0.0686 - accuracy: 0.9799
Epoch 74/100
13/13 [==============================] - 0s 4ms/step - loss: 0.0684 - accuracy: 0.9824
Epoch 75/100
13/13 [==============================] - 0s 4ms/step - loss: 0.0681 - accuracy: 0.9824
Epoch 76/100
13/13 [==============================] - 0s 3ms/step - loss: 0.0679 - accuracy: 0.9824
Epoch 77/100
13/13 [==============================] - 0s 4ms/step - loss: 0.0675 - accuracy: 0.9824
Epoch 78/100
13/13 [==============================] - 0s 4ms/step - loss: 0.0672 - accuracy: 0.9824
Epoch 79/100
13/13 [==============================] - 0s 5ms/step - loss: 0.0668 - accuracy: 0.9824
Epoch 80/100
13/13 [==============================] - 0s 4ms/step - loss: 0.0666 - accuracy: 0.9849
Epoch 81/100
13/13 [==============================] - 0s 4ms/step - loss: 0.0663 - accuracy: 0.9849
Epoch 82/100
13/13 [==============================] - 0s 4ms/step - loss: 0.0660 - accuracy: 0.9849
Epoch 83/100
13/13 [==============================] - 0s 4ms/step - loss: 0.0657 - accuracy: 0.9849
Epoch 84/100
13/13 [==============================] - 0s 5ms/step - loss: 0.0656 - accuracy: 0.9849
Epoch 85/100
13/13 [==============================] - 0s 4ms/step - loss: 0.0652 - accuracy: 0.9849
Epoch 86/100
13/13 [==============================] - 0s 4ms/step - loss: 0.0651 - accuracy: 0.9849
Epoch 87/100
13/13 [==============================] - 0s 4ms/step - loss: 0.0647 - accuracy: 0.9849
Epoch 88/100
13/13 [==============================] - 0s 4ms/step - loss: 0.0644 - accuracy: 0.9849
Epoch 89/100
13/13 [==============================] - 0s 5ms/step - loss: 0.0643 - accuracy: 0.9849
Epoch 90/100
13/13 [==============================] - 0s 4ms/step - loss: 0.0641 - accuracy: 0.9849
Epoch 91/100
13/13 [==============================] - 0s 4ms/step - loss: 0.0638 - accuracy: 0.9849
Epoch 92/100
13/13 [==============================] - 0s 4ms/step - loss: 0.0635 - accuracy: 0.9849
Epoch 93/100
13/13 [==============================] - 0s 4ms/step - loss: 0.0633 - accuracy: 0.9849
Epoch 94/100
13/13 [==============================] - 0s 4ms/step - loss: 0.0630 - accuracy: 0.9849
Epoch 95/100
13/13 [==============================] - 0s 4ms/step - loss: 0.0628 - accuracy: 0.9849
Epoch 96/100
13/13 [==============================] - 0s 4ms/step - loss: 0.0625 - accuracy: 0.9849
Epoch 97/100
13/13 [==============================] - 0s 3ms/step - loss: 0.0623 - accuracy: 0.9849
Epoch 98/100
13/13 [==============================] - 0s 4ms/step - loss: 0.0620 - accuracy: 0.9849
Epoch 99/100
13/13 [==============================] - 0s 4ms/step - loss: 0.0618 - accuracy: 0.9849
Epoch 100/100
13/13 [==============================] - 0s 4ms/step - loss: 0.0615 - accuracy: 0.9849
<keras.src.callbacks.History at 0x1abe9412890>
```

Making the Confusion Matrix

```
[[106   2]
 [  0  63]]
0.9883040935672515
```