# Assignment 2

1. EDA
2. Summary statistics:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Feature** | **Count** | **Mean** | **Std Dev** | **Min** | **25%** | **50%** | **75%** | **Max** |
| Mean Radius | 198 | 17.4121 | 3.1572 | 10.95 | 15.0525 | 17.28 | 19.58 | 27.22 |
| Mean Texture | 198 | 22.3190 | 4.2831 | 10.38 | 19.5175 | 21.795 | 24.655 | 39.28 |
| Mean Perimeter | 198 | 114.8566 | 21.3834 | 71.9 | 98.16 | 113.7 | 129.65 | 182.1 |
| Mean Area | 198 | 970.0409 | 352.1492 | 361.6 | 702.525 | 929.1 | 1193.5 | 2250.0 |
| Mean Smoothness | 198 | 0.1027 | 0.0125 | 0.07497 | 0.0939 | 0.1019 | 0.1110 | 0.1447 |
| Mean Compactness | 198 | 0.1426 | 0.0499 | 0.04605 | 0.1102 | 0.1318 | 0.1722 | 0.3114 |
| Mean Concavity | 198 | 0.1562 | 0.0706 | 0.02398 | 0.10685 | 0.1513 | 0.2005 | 0.4268 |
| Mean Concave Points | 198 | 0.0868 | 0.0339 | 0.02031 | 0.06367 | 0.0861 | 0.1039 | 0.2012 |

1. total rows = 198

unique values = ['N' 'R']

top value = N

top value frequency = 151

1. Yes, we can encode the categorical variable ‘outcome’ to numerical data type. As the column only has two unique values and we are performing binary classification using logistic regression, I am going to use ‘*label encoding*. For complex classification we can also use OHE (One-hot Encoding).
2. Yes, there are multiple redundant features in the dataset. Those are the features that have high correlation or same data. For example, in the given dataset, mean\_radius and mean\_perimeter have correlation of 1. There are also other features which have high correlation. Since the features having high correlation doesn’t affect the outcome of the data, removing one of the features having high correlation with the other won’t impact the analysis.

One interesting observation is that there are different variables that have correlation of 1 even though the data is different. There are also many variables with high correlations suggesting many redundant features in the data.

A diagram of a heat map

AI-generated content may be incorrect.

Another interesting observation is that based on the statistics computed among the given variables, mean\_radius, mean\_texture, mean\_perimeter, mean\_area, mean\_smoothness, mean\_compactness, mean\_concavity, and mean\_concave\_points, median is close to the mean but slightly less in all the variables suggesting slight right skewness.

Finally, we can observe that there is an imbalance in the data. The number of data points for outcome R is much less than for N.

1. correlation between ‘mean\_perimeter’ and ‘se\_perimeter’ = 0.6099643781634989
2. **Encoding**: [R = 1, N = 0]

**Note:**

Since there is imbalance in the data and overlaps in the data points for outcome of 1 and 0 as you can see in the diagram below, the model is not able to fit a sigmoid line properly. The line seemed to have been dipped instead of taking the shape of the logistic curve. I also tried to stratify the data and check if the model works. The model predicted similar output. I tried this with all the 12 features individually, and it gave a similar prediction.

Finally, I used z-score normalization and use data pooling and balanced the data points by down sampling the data with high frequency. Then it performed better than without normalization. I cross validated the performance of the model with scikit-learn to see how it will perform. The performance with the scikit-learn model was comparable enough to the model I implemented. So, I decided to go with the model using down sampling and z-score normalization.

A graph of a number of red and blue dots

AI-generated content may be incorrect.

As we can observe from the figure, there is an imbalance in the data points and many overlaps. The case is the same for every feature in the data set.

1. Yes, we can map the likelihood of breast cancer recurrence based on “mean\_area” feature by observing the performance of the model I implemented. However, the model is not on par with what we would expect. The F1 score in the range of 0.64 from the down sampled data. If we had sufficient data points to balance the data in two categories, I would say that the model would have performed a lot better.

A diagram of a graph

AI-generated content may be incorrect.

1. TP = 9 FP = 10 TN = 1 FN= 0 Threshold: 0.3

Precision = 0.47368421052631576 Recall = 1.0

F1 Score = 0.6428571428571429

Again, the performance is not on par but if we could get a more balanced dataset, the performance could be improved.

1. The implementation is done on the jupyter-notebook file. The performance of the model decreased. Below are the evaluation data.

TP = 18 FP = 23 TN = 88 FN= 16 Threshold = 0.3

Precision = 0.43902439024390244 Recall = 0.5294117647058824

F1 Score = 0.48

1. For the forward selection process, first I performed logistic regression for all the features individually and evaluated the performance using F1 score. The scores are shown in the Table below.

A table with numbers and text

AI-generated content may be incorrect.

From the features, I selected the highest feature with highest performance and dropped others. During the process, I also evaluated the correlation between each feature and dropped the features with high correlation as the F1 score from the features were same.

The correlation can be observed from the correlation matrix heatmap below.

A colorful chart with black text

AI-generated content may be incorrect.

Since mean\_area, mean\_radius, and mean\_perimeter result in same F1 score and have high correlation, I eliminated mean\_radius, and mean\_perimeter.

Similarly, we can observe from the heatmap that the mean\_concavity and mean\_compactness also have very high correlation. From those two, I chose mean\_concavity. Mean concavity also has high correlation with mean\_concave points so I am also dropped mean\_concave\_points.

se\_perimeter and se\_area also have high correlation. Therefore, I dropped se\_perimeter.

**dropped**: mean\_radius, mean\_perimeter, mean\_compactness, mean\_concave\_points, se\_perimeter

**remaining**: mean\_texture, mean\_area, mean\_smoothness, mean\_concavity, mean\_fractal\_dimension, se\_texture, se\_area

Then I compared the combination of the remaining features with mean\_area and the performance is shown in the table below.

A screenshot of a computer

AI-generated content may be incorrect.

Since the performance was not improving, no selection would not be able to improve the performance of the model.

Therefore, the best performance was obtained using mean\_area feature alone.

1. The performance of the full model was:

TP = 18 FP = 23 TN = 88 FN= 16 Threshold = 0.3

Precision = 0.43902439024390244 Recall = 0.5294117647058824

F1 Score = 0.48

This is clearly an underperforming model than the model selected from 3.b. The performance of the model from 3b is given below:

TP = 9 FP = 10 TN = 1 FN= 0 Threshold: 0.3

Precision = 0.47368421052631576 Recall = 1.0

F1 Score = 0.6428571428571429

1. For the model to perform as per expectations, I had already done z-score normalization and checked the performance. So, yes, the model performs better if we use scale the features.

After regularization, the model underperformed. Here is the performance of the model:

TP = 9 FP = 11 TN = 0 FN= 0 Threshold=0.3

Precision = 0.45 Recall = 1.0

F1 Score = 0.6206896551724138

2. I have implemented different function MSE\_gradient\_descent in the LogisticRegression class to evaluate the performance of MSE in the logistic function.
3. After implementing LL\_gradient\_descent where I put the stopping criteria of 0.0001difference between previous cost and current cost. The program terminated before the total number of iterations met but the F1 score dropped. However, the performance with both cost were same.

MSE:

final bias = -0.03381127604770039 final weights = [0.0954779]

TP = 9 FP = 11 TN = 0 FN= 0

Precision = 0.45 Recall = 1.0

F1 Score = 0.6206896551724138

LL:

final bias = -0.2616166278432413 final weights = [0.21339542]

TP = 9 FP = 11 TN = 0 FN= 0

Precision = 0.45 Recall = 1.0

F1 Score = 0.6206896551724138