# Report of laboratory works 1-2

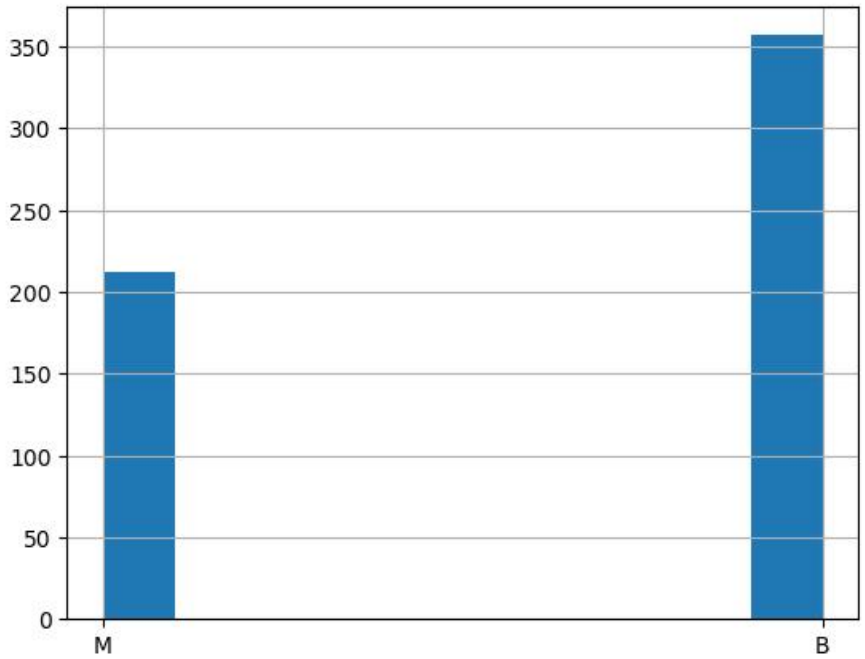
This analysis aims to observe which features are most helpful in predicting malignant or benign cancer and to see general trends that may aid us in model selection and hyper parameter selection. The goal is to classify whether the breast cancer is benign or malignant. To achieve this i have used machine learning classification methods to fit a function that can predict the discrete class of new input.

The dataset I am working with contains the following attribute information:

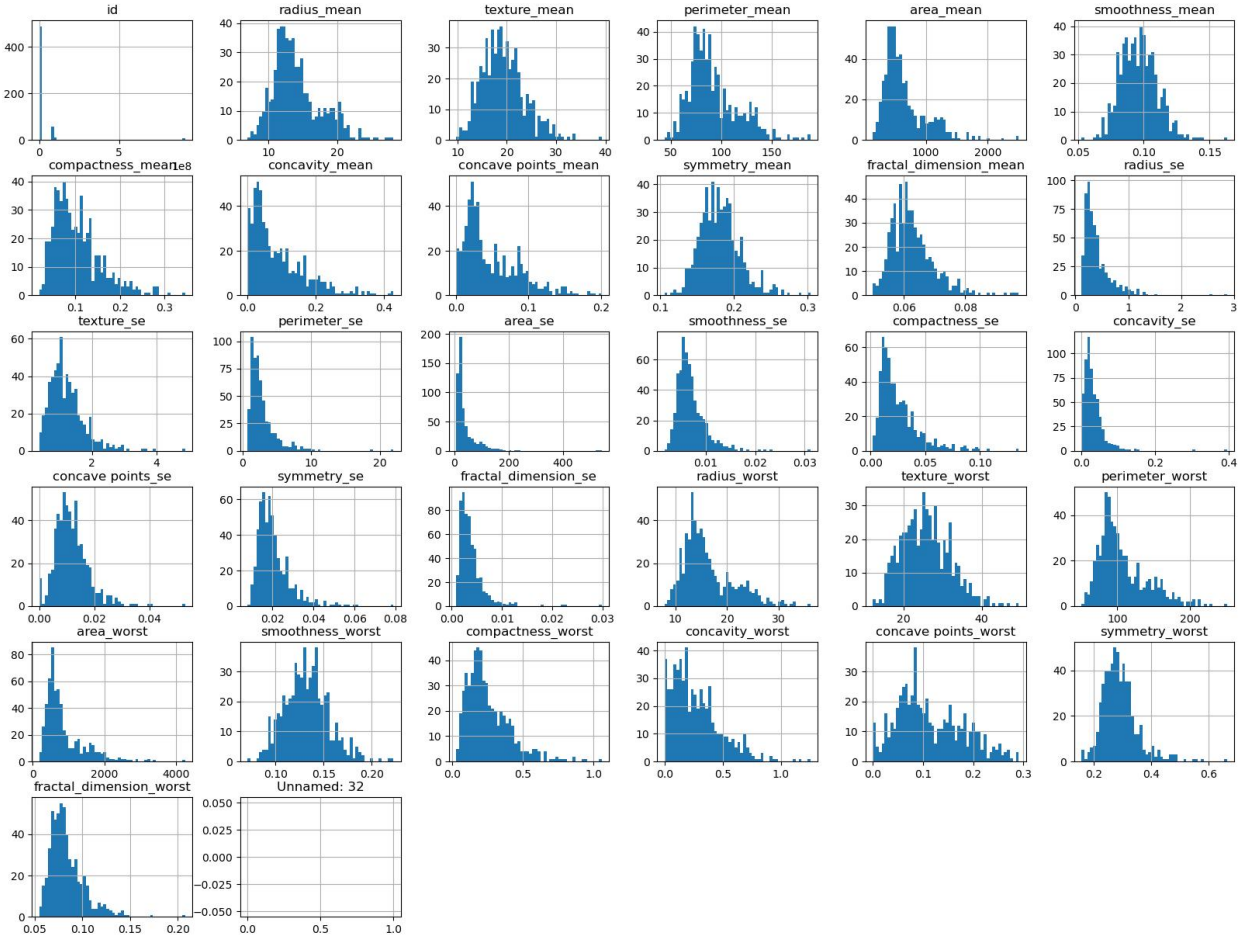
- ID number
- Diagnosis (M = malignant, B = benign)
- Computed features for each cell nucleus including radius, texture, perimeter, area, smoothness, compactness, concavity, concave points, symmetry and fractal dimension. The mean, standard error and “worst” of these features were computed for each image.

The following histogram shows the number of benign and malignant cancers.

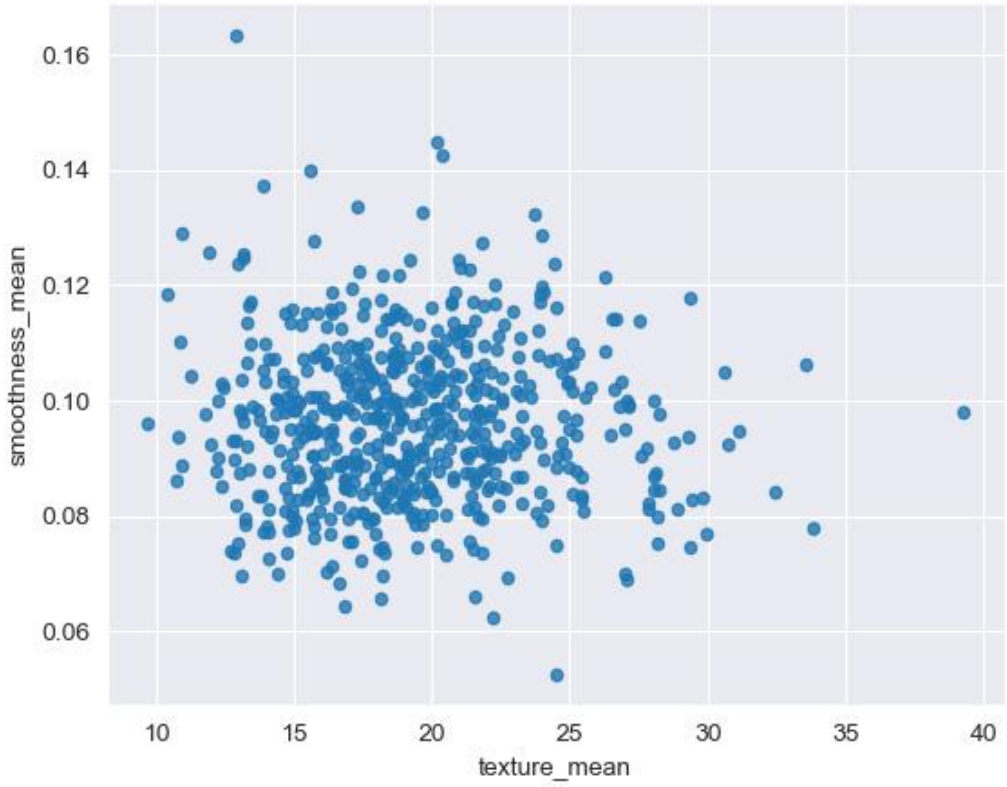
- B-357
- M-212



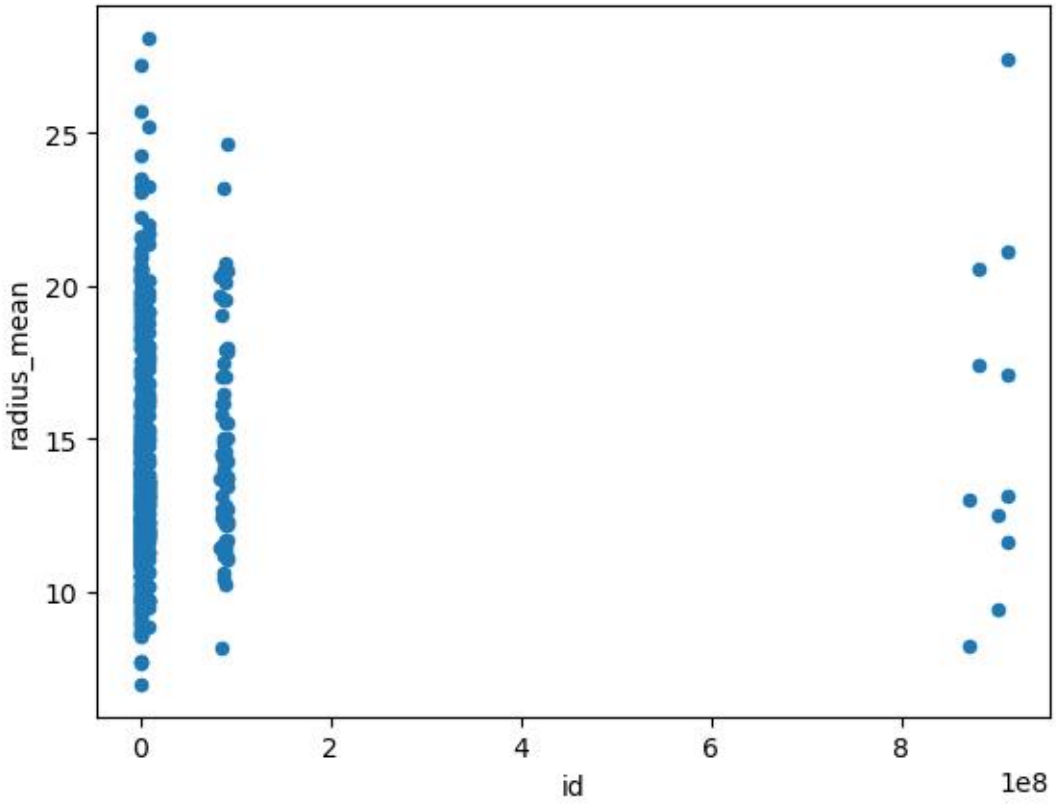

❖ The next stage was to visualize the data and see the distributions for each of the set of datasets that I was studying. The information is shown in the graphs below:

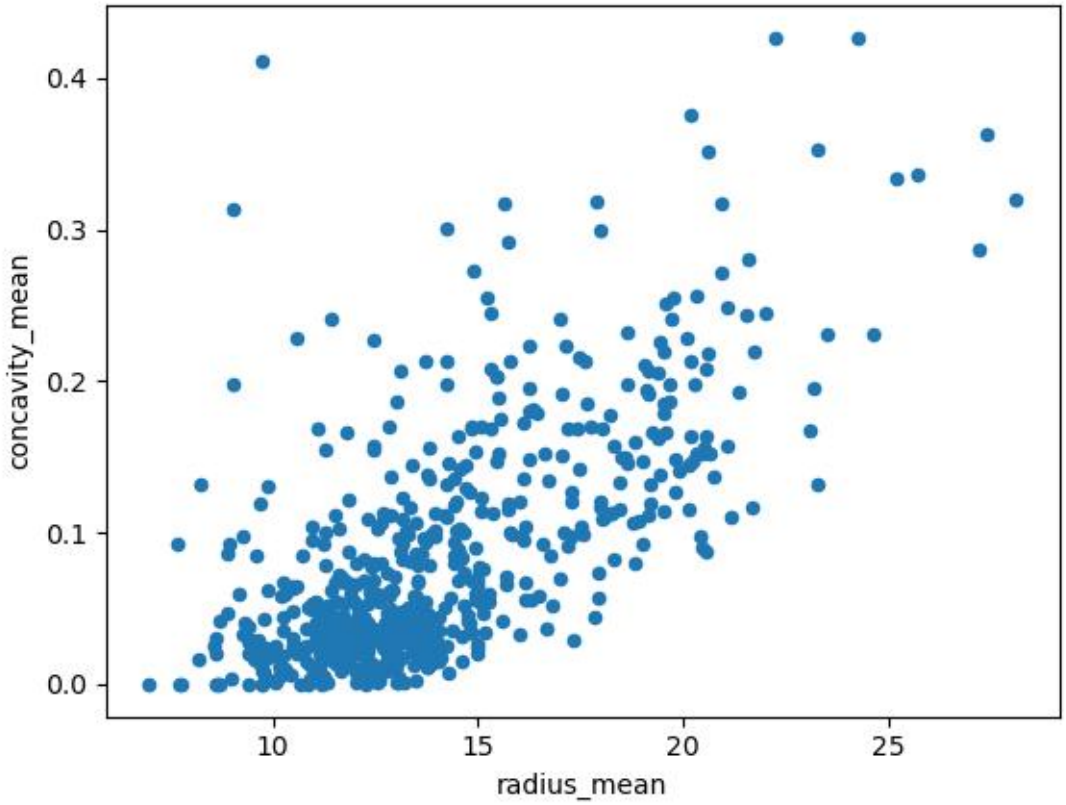



❖ Using scatter matrix, I scattered the data from the texture mean and smoothness mean of each cell nucleus with the goal to determine the relationship between them but the information was inconclusive.

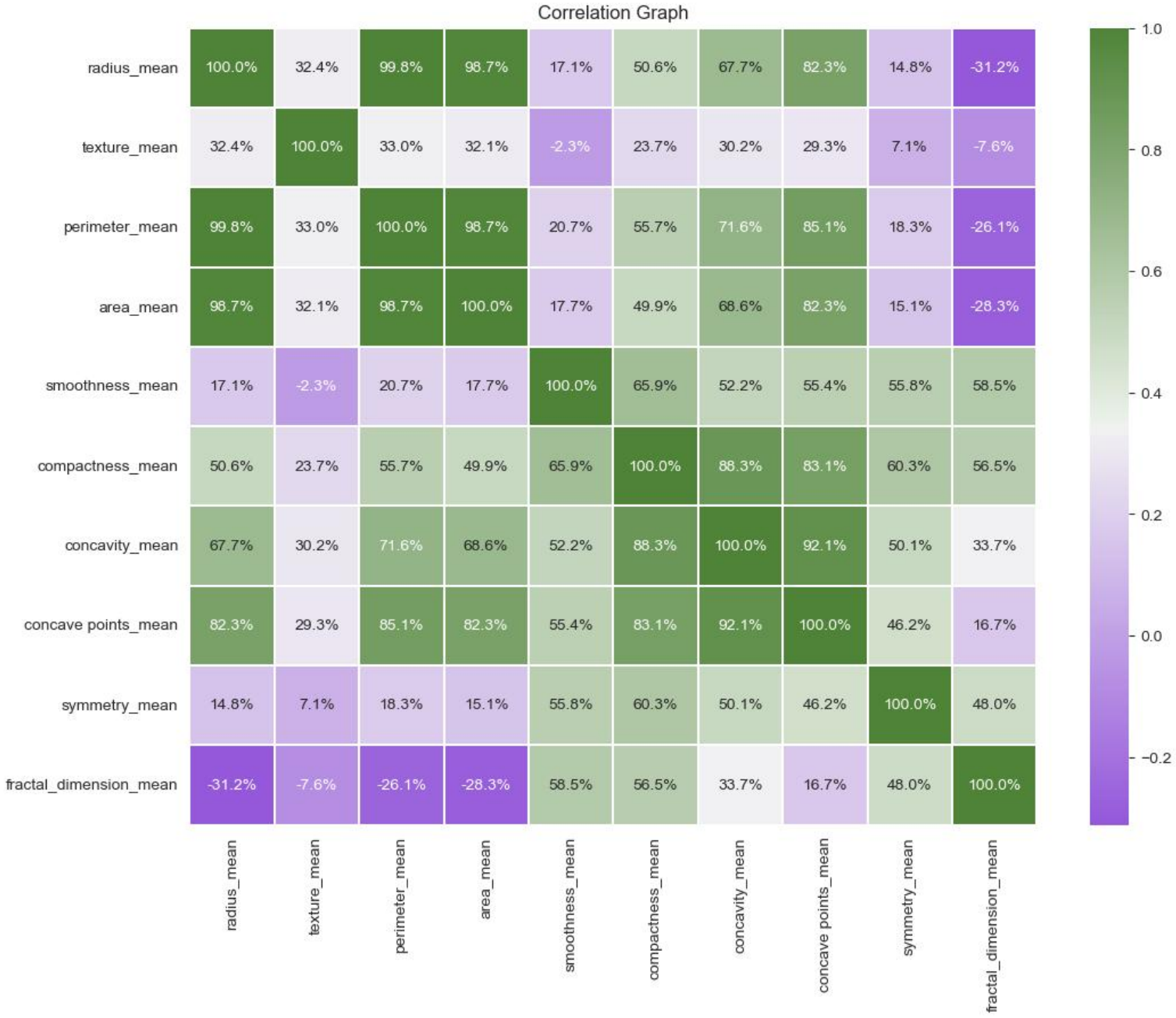


- Scatter matrix does not provide any conclusive information about the relationships of some of our attribute features of the cell nuclei that we were studying as shown in the graphs below:




- I was also able to find the correlations between the attribute features and I compiled the information in the graph below. Visualization of the correlation graphs showed that the perimeter mean and the radius mean were the most correlated features of the cells and the least correlated were the fractal dimension mean and radius mean.




❖ After building my models in the second part of the laboratory work, I trained them and tested them.

❖ To check the accuracy, I imported confusion matrix method of metrics class. The confusion matrix is a way of tabulating the number of mis-classifications, i.e., the number of predicted classes which ended up in a wrong classification bin based on the true classes and I found just 4 mis-calculations.

```
print(len(confusion_matrixs))
```

```
4
```

❖ The classification report for each of my models showing the precision scores in predicting the type of cancer cells is represented in the tables below:

Classification Report of 'LogisticRegression '

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| B | 0.90 | 0.96 | 0.93 | 115 |
| M | 0.92 | 0.84 | 0.88 | 73 |
| accuracy | | | 0.91 | 188 |
| macro avg | 0.91 | 0.90 | 0.90 | 188 |
| weighted avg | 0.91 | 0.91 | 0.91 | 188 |

Classification Report of 'RandomForestClassifier '

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| B | 0.92 | 0.96 | 0.94 | 115 |
| M | 0.93 | 0.88 | 0.90 | 73 |
| accuracy | | | 0.93 | 188 |
| macro avg | 0.93 | 0.92 | 0.92 | 188 |
| weighted avg | 0.93 | 0.93 | 0.93 | 188 |

Classification Report of 'DecisionTreeClassifier '

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| B | 0.90 | 0.96 | 0.93 | 115 |
| M | 0.92 | 0.84 | 0.88 | 73 |
| accuracy | | | 0.91 | 188 |
| macro avg | 0.91 | 0.90 | 0.90 | 188 |
| weighted avg | 0.91 | 0.91 | 0.91 | 188 |

Classification Report of 'SVC '

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| B | 0.90 | 0.97 | 0.93 | 115 |
| M | 0.94 | 0.84 | 0.88 | 73 |
| accuracy | | | 0.91 | 188 |
| macro avg | 0.92 | 0.90 | 0.91 | 188 |
| weighted avg | 0.92 | 0.91 | 0.91 | 188 |

❖ I managed to find the accuracy of each of my models based on the precision score and sorted them on which one gave the most accurate prediction as a percentage.

| | model_name | score | accuracy_score | accuracy_percentage |
|---|---|---|---|---|
| 0 | LogisticRegression | 0.916010 | 0.909574 | 90.96% |
| 1 | RandomForestClassifier | 0.992126 | 0.925532 | 92.55% |
| 2 | DecisionTreeClassifier | 1.000000 | 0.909574 | 90.96% |
| 3 | SVC | 0.923885 | 0.914894 | 91.49% |

```
df_pred.sort_values('accuracy_percentage', ascending=False)
```

| | model_name | score | accuracy_score | accuracy_percentage |
|---|---|---|---|---|
| 1 | RandomForestClassifier | 0.992126 | 0.925532 | 92.55% |
| 3 | SVC | 0.923885 | 0.914894 | 91.49% |
| 0 | LogisticRegression | 0.916010 | 0.909574 | 90.96% |
| 2 | DecisionTreeClassifier | 1.000000 | 0.909574 | 90.96% |

❖ FROM THE ABOVE INFORMATION FROM THE MODELS, I CHOSE RANDOM FOREST AS THE MODEL THAT I'LL DEPLOY AND IMPLEMENT AS IT GAVE ME THE BEST ACCURACY SCORES.