Machine Learning Project

Breast Cancer Detection

Final Report

Names	BN	Sec	Contribution
Abdelaziz Salah Mohammed	2	2	25%
Abdelrahman Noaman	4	2	25%
Khaled Hesham	21	1	25%
Kirollos Samy	13	2	25%

Delivered to:

Eng/ Mohammed Shawky

Date: 9/5/2024

Problem Definition:

Develop a machine learning model to classify breast cancer tumors as either benign or malignant based on features extracted from diagnostic images and patient data.

Motivation:

Appling the machine learning model in the medical field could be very helpful, because we may save a lot of souls, if we could the malignant cells early.

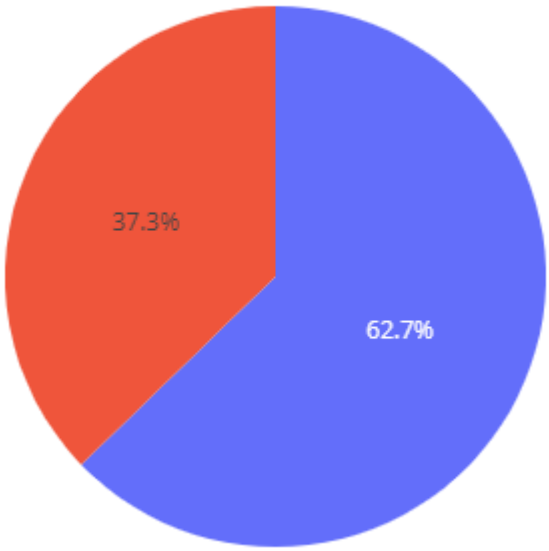
Moreover, it is a good chance to apply all what we have learned in the course on a classification problem.

Dataset:

The dataset contains a collection of features computed from digitized images of fine needle aspirate (FNA) of breast mass and patient data. Each instance in the dataset represents a breast tumor and is labeled as either benign or malignant.

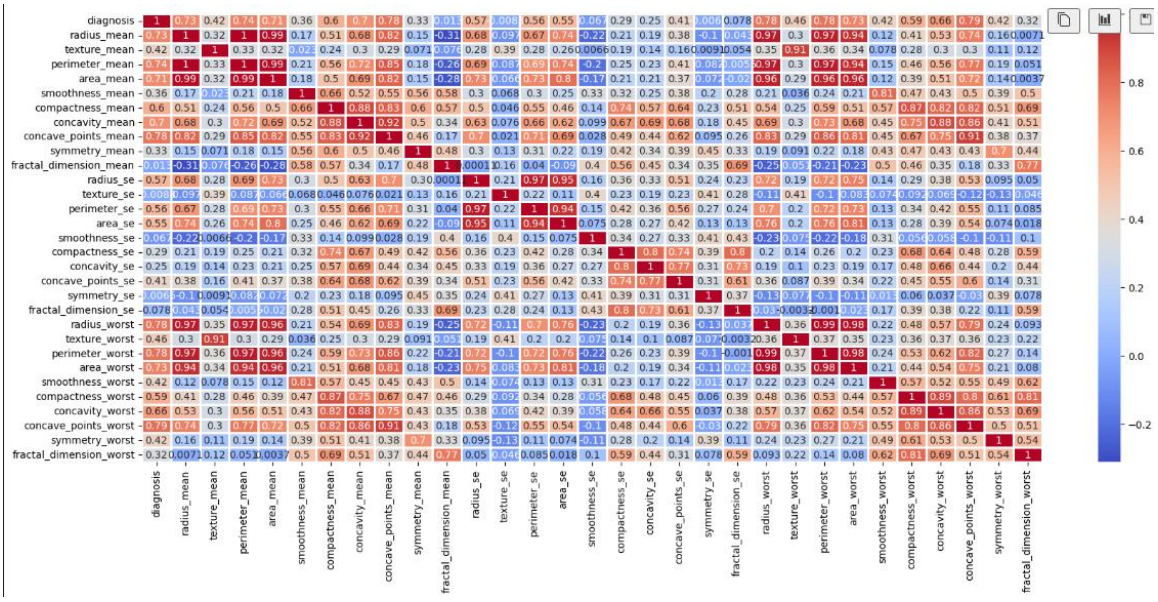
Data Preparation and analysis:

- Pie chart for the benign (in blue) and malignant (in orange):

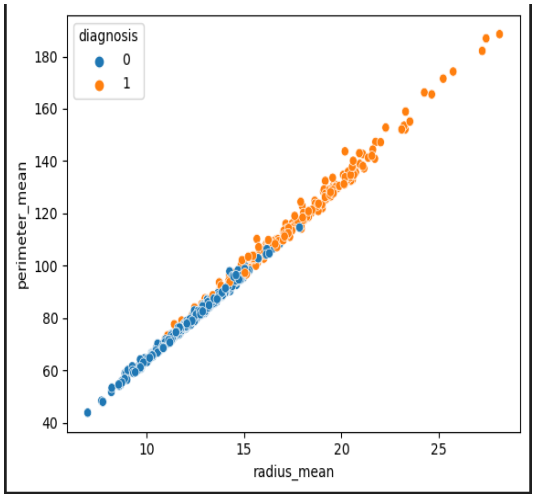


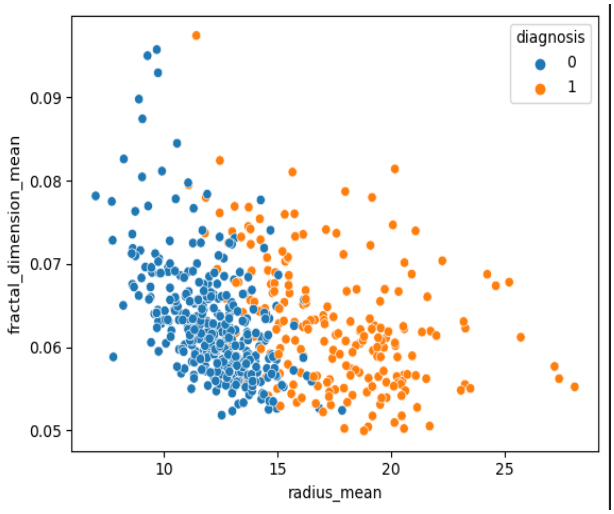
Date: 9/5/2024

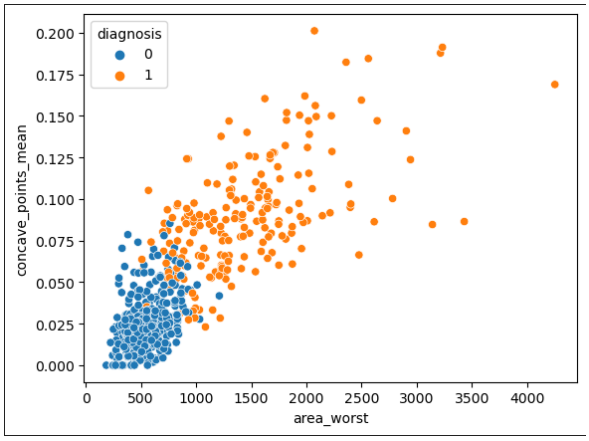
- The Correlation matrix:



- Snapshots for some of the features, to show the correlation between them:







Date: 9/5/2024

Results and Analysis:

- We Have trained 9 models in this project, to figure out what is the best accuracy we can get.
- So, lets see our analysis and results:
  1. **ZeroR base model:**
     - We used this model as our base model, to compare the other accuracies on it, and it is just a simple model which get the class with the highest frequency in the training dataset, and just predict any test point with this class, and on evaluating its accuracy, we got than benign is the most frequent class, and the accuracy of this model was around 62.7%.

  2. **Perceptron:**
     - Another simple model we tried to use for our binary classification problem was the perceptron simple model, however, we find that the perceptron is not suitable for our dataset, since it is not linearly separable, and one of the main assumptions of the perceptron is that the data must be linearly separable.
     - We noticed that because the model always reached the maximum number of iterations without finding a completely correct classification hyperplane.
  3. **Linear Regression:**
     - We have implemented the Linear Regression model ourselves and compared its results with the results of the **skLearn model**, and we got the exact same results.
     - Also, we have tried it on scaled, and non-scaled features, and the difference in the accuracy was significant, and that was

Date: 9/5/2024

an interesting point, to show how feature scaling and overfitting can significantly matter.

- In the below two tables, we will show the results of the logistic regression on 5 different values of C (reciprocal of the regularization factor), and our main evaluation metrics which are:
  - F1 Score
  - Training Accuracy
  - Testing Accuracy
  - Precision
  - Recall
- Also, we used the penalty of l2, which is ridge regression, which is a method used to prevent overfitting.

1. **Without scaling:**

C	Training Accuracy	Testing Accuracy	F1 Score	Precision	Recall
0.001	93.84	96.49	95.23	97.5	93.0
0.01	92.96	94.73	92.6	97.4	88.3
0.1	94.28	96.49	95.2	97.5	93.0
1	94.50	96.49	95.2	97.5	93.0
10	94.72	95.61	93.9	97.5	90.6
100	94.50	96.49	95.2	97.5	93.0

2. **With Scaling**

C	Training Accuracy	Testing Accuracy	F1 Score	Precision	Recall
0.001	90.32	88.59	82.19	1.0	69.76

Date: 9/5/2024

0.01	95.60	96.49	95.12	1.0	90.69
0.1	**98.02**	**98.24**	**97.61**	**1.0**	**95.34**
1	98.68	97.36	96.47	97.61	95.34
10	98.90	97.36	96.55	95.45	97.67
100	99.34	93.85	92.30	87.5	97.67

- We have also applied PCA to reduce the input space, and avoid the curse of dimensionality, however, this did not enhance the accuracy of the model, we have tried to reduce the components to 10-19 principle components, but all of them was with the exact same training and testing accuracy of:
    - Training Accuracy: 85%
    - Testing Accuracy: 92%
- However, changing the principal components, has just changed the precision and the recall of the model, but we get almost get the exact same F1 Score.

4. **SVM:**

- We have implemented 4 different kernels, on the scaled data, to check their performance, and get the best of them.
- However, usually on choosing the most suitable kernel we need to have a look on the following factors:

    - Linear Kernel:
        - Use when your data is linearly separable or when you have a large number of features compared to the number of samples.
        - It's computationally less expensive compared to non-linear kernels and can be a good choice for high-dimensional data.
    - Polynomial Kernel:

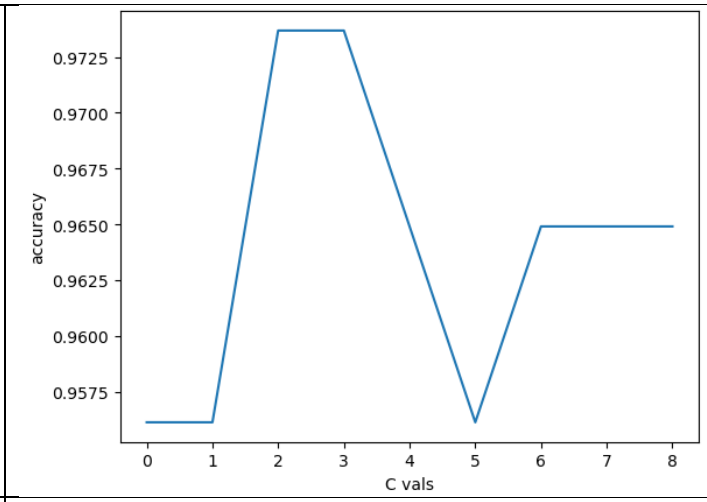
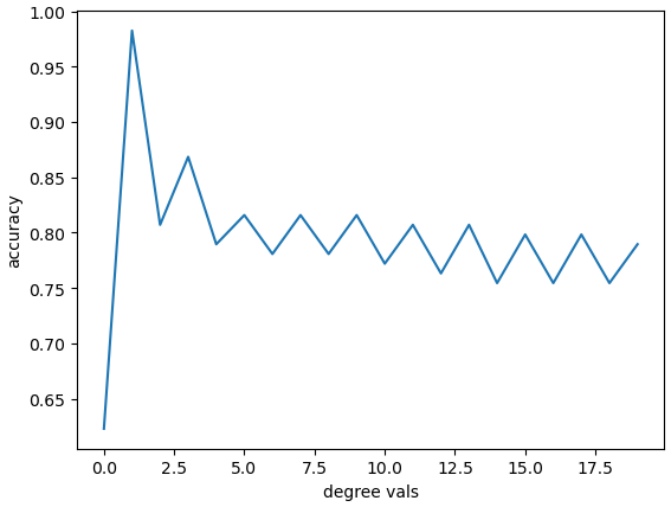
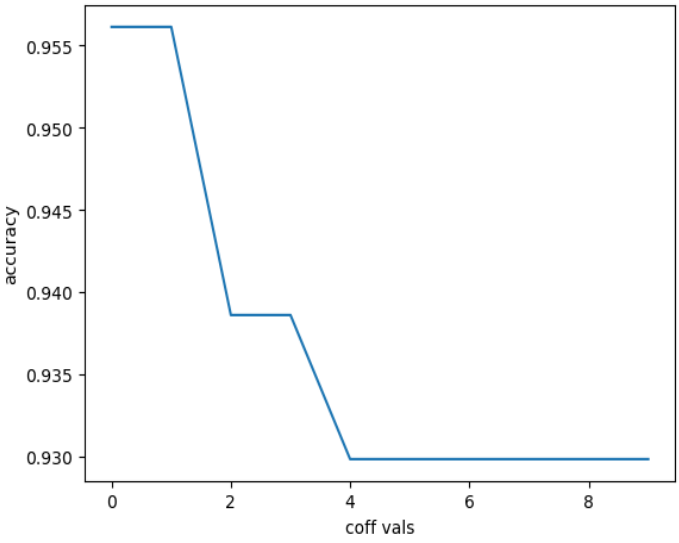
Date: 9/5/2024

        - Use when you suspect that the decision boundary is polynomial but not too complex.
        - It's suitable for data with moderate complexity and can capture non-linear relationships to some extent.
        - Be cautious with the degree parameter as higher degrees can lead to overfitting.
    - Radial Basis Function (RBF) Kernel:
        - Use as a default choice when you don't have prior knowledge about the dataset or when the dataset is not linearly separable.
        - It's effective for capturing complex, non-linear relationships in the data.
        - Be careful with the gamma parameter, as it controls the kernel's width and can greatly impact the model's performance.
        - Higher values of gamma can lead to overfitting.

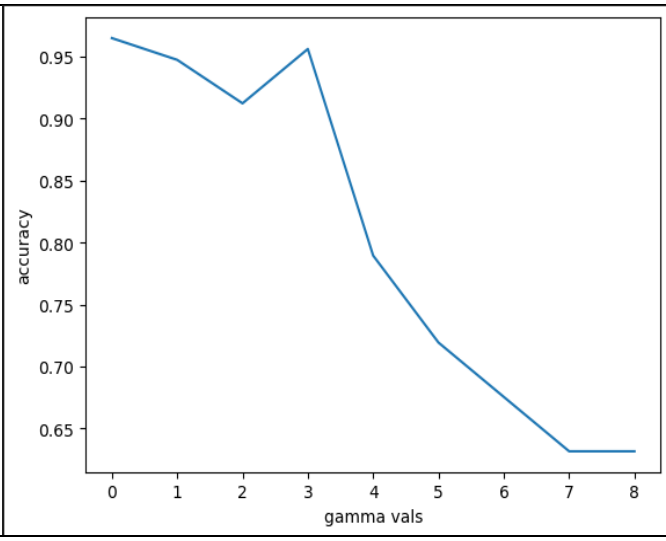
    - Sigmoid Kernel:
        - Use with caution as it's less commonly used compared to the other kernels.
        - It's suitable for problems where the input data can be mapped into a higher-dimensional space and exhibits a sigmoidal relationship.
        - Tends to be less robust and less stable compared to other kernels.
    - Custom Kernels:
        - Use when none of the predefined kernels suit your problem, or when you have domain-specific knowledge that can be encoded into a custom kernel.
        - Ensure that the custom kernel satisfies the Mercer's condition for the SVM algorithm's convergence and stability.

Date: 9/5/2024

Results:

Linear	
Poly	
Sigmoid	

Date: 9/5/2024

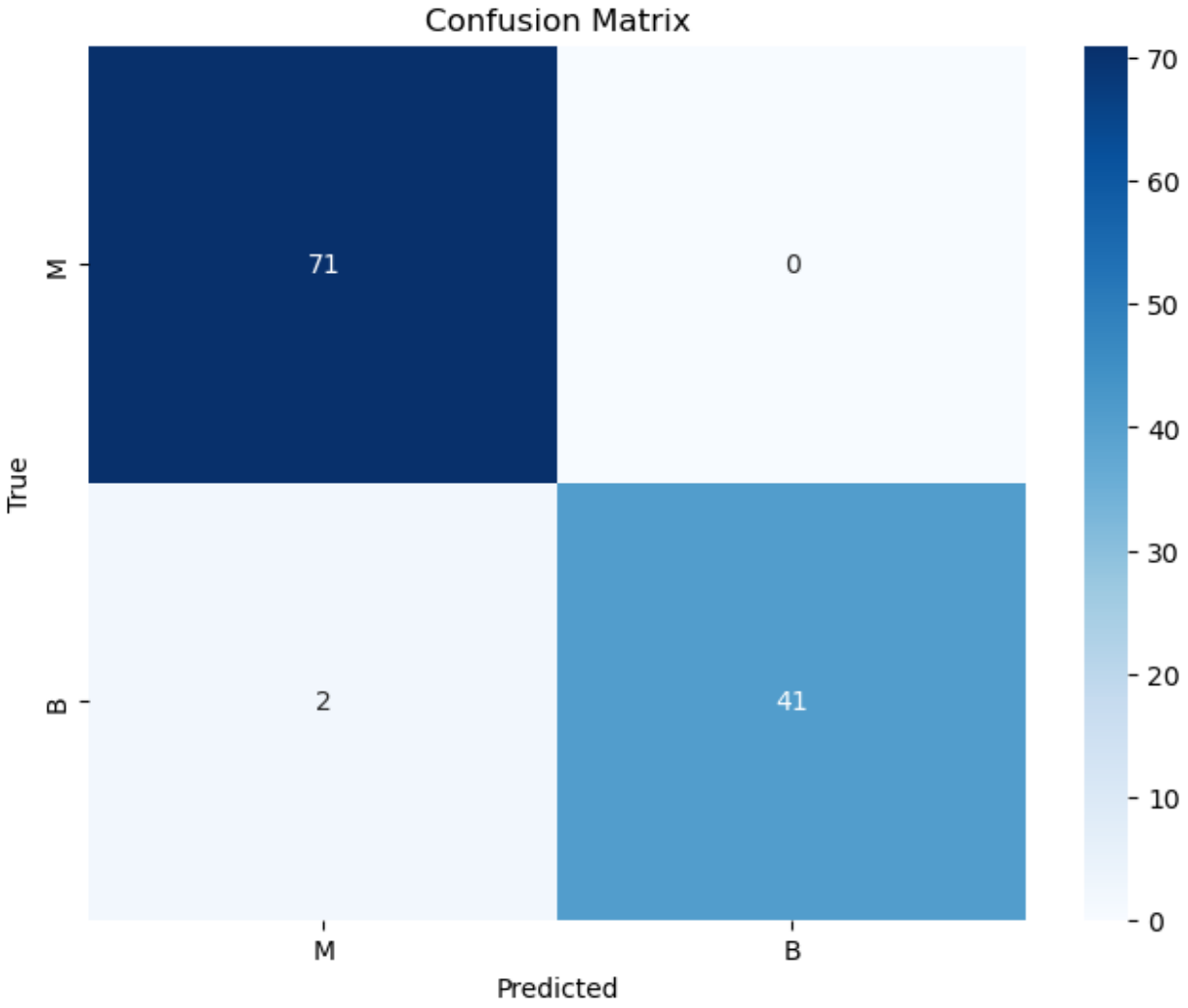
RBF			

- After this analysis, and fine tuning on all the hyper parameters, we figured out that the best model for SVM is the ploy kernel with second degree, it has accuracy of **98.24.**

5. <u>Best Model:</u>

- At the end of the analysis, we have made a list containing 9 different models to figure out which is the best and most suitable model for this problem, and we figured out that the winner is the SVM with poly kernel of degree 2, as it achieved the highest accuracy on the test set with 98.24%, and here is the confusion matrix for it.

<u>Date: 9/5/2024</u>


Confusion Matrix
71
0
2
41
M
B
True
M
B
Predicted
70
60
50
40
30
20
10
0

Date: 9/5/2024