Table of Contents

[Project Overview 1](#_Toc158899940)

[Problem Statement 1 1](#_Toc158899941)

[Question 1](#_Toc158899942)

[Code 1](#_Toc158899943)

[Output 1](#_Toc158899944)

[Problem Statement 2 3](#_Toc158899945)

[Question 3](#_Toc158899946)

[Code 3](#_Toc158899947)

[Output 3](#_Toc158899948)

[Problem Statement 3 3](#_Toc158899949)

[Question 3](#_Toc158899950)

[Code 3](#_Toc158899951)

[Output 3](#_Toc158899952)

[Observations 4](#_Toc158899953)

[Problem Statement 4 5](#_Toc158899954)

[Question 5](#_Toc158899955)

[Code 5](#_Toc158899956)

[Output 5](#_Toc158899957)

[Observations 6](#_Toc158899958)

[Problem Statement 5 7](#_Toc158899959)

[Question 7](#_Toc158899960)

[Code 8](#_Toc158899961)

[Output 8](#_Toc158899962)

[Observations 10](#_Toc158899963)

[Project 2 Conclusion 10](#_Toc158899964)

[Appendix 11](#_Toc158899965)

[Problem Statement 1 11](#_Toc158899966)

[Code 11](#_Toc158899967)

[Problem Statement 2 12](#_Toc158899968)

[Code 12](#_Toc158899969)

[Problem Statement 3 13](#_Toc158899970)

[Code 13](#_Toc158899971)

[Problem Statement 4 14](#_Toc158899972)

[Code 14](#_Toc158899973)

[Problem Statement 5 15](#_Toc158899974)

[Code 15](#_Toc158899975)

**List of Tables**

[Table 1: Split of Data into Training, Validation and Testing with values. 1](#_Toc160069853)

[Table 2: Cross-Validation Scores for each iteration and the average. 2](#_Toc160069854)

# Project Overview

The project 3 aims to expand learning through practice upon the concepts of Neural Network multilayer perceptron learning algorithms with focus upon MLPClassifier and the MLPRegressor.

# Problem Statement 1

## Question

The expectation is to write a python code that splits the Wisconsin Breast Cancer data set available in the *Sklearn* dataset library into Training/Validation and Testing subsets through a split ratio of 80% and 20% respectively. The key objective is to mention and describe how the split is conducted along with providing the random value used for the split for the purpose of recreation.

## Code Snippet

The following is the code snippet for the question given:



The entire code block for the Problem Statement 1 can be found in the appendix within the Code 1 of this document.

## Output



The above output clearly identifies the total number of samples in the Wisconsin Breast Cancer Dataset which is 569 and the number of samples divided for Training and Validation is 455 i.e. 80% (79.96%). And the number of samples divided for Testing is 114 i.e. 20% (20.03%). The random state value provided for the split is given as 19.

|  |  |  |
| --- | --- | --- |
| **Category** | **Percentage split** | **Number of rows (samples)** |
| Training and Validation | 80% | 455 |
| Testing | 25% | 114 |

Table 1: Split of Data into Training, Validation and Testing with values.

# Problem Statement 2

## Question

The requirement here is to either create an additional split for the validation sub-dataset or to use cross validation technique to tune the overall program’s MLP Hyperparameters.

## Code Snippet

The following is the code snippet for the question given:



The entire code block for the Problem Statement 2 can be found in the appendix within the Code 1 of this document.

## Output



The output at first displays the cross-validation scores with a cross validation count of number of folds is given as 10. This means that the validation subset is divided into 10 equal folds and each fold is compared with the remaining 9 other to evaluate the model’s performance based on 10 iterations (in this case). The average value of these 10-fold iterations is also displayed and the value obtained here is 0.9035. This shows that the model is already leaning on a good learning curve.

|  |  |
| --- | --- |
| **Iteration Count** | **Cross Validation Scores** |
| 1 | 0.95652174 |
| 2 | 0.86956522 |
| 3 | 0.89130435 |
| 4 | 0.80434783 |
| 5 | 0.89130435 |
| 6 | 0.95555556 |
| 7 | 0.91111111 |
| 8 | 0.91111111 |
| 9 | 0.88888889 |
| 10 | 0.95555556 |
| **Average** | **0.9035265700483093** |

Table 2: Cross-Validation Scores for each iteration and the average.

# Problem Statement 3

## Question

The expectation with this question is to provide a procedure to document the design process and to identify and mention the tradeoffs considered in using a MLPClassifier.

## Design Process Procedure

The following are the general steps involved (along with short descriptions in relation to the Project 3 fulfillment) in the establishment of design as a precursor to the MLPClassifier:

1. Problem Definition:

In this case, the problems are clearly defined in what is to be achieved along with the necessary inputs (Ex. Choice of Dataset), procedural steps and the necessary outputs (Ex. MLP and Cross Validation scores).

1. Data Collection and Preprocessing:

This step involved prepping the data (cleaning, normalization etc.) before operations can be done using it. However, in this case, since we are already using a prepped dataset such as the Wisconsin Breast Cancer Dataset from the Sklearn dataset library, it is directly used for operations.

1. Model Architecture:

The model architecture focusses on the finer details of how the data is to be analyzed based on a certain set of hyperparameters such as number of layers with the number of neurons in each layer along with the number of epochs that is to be conducted et al.

1. Training and Hyperparameter tuning:

This stage of the process trains the model from the split training sub-dataset and tuning the hyperparameters such as the number of neurons, learning rate etc. The usage of cross validation technique is also considered for hyperparameter to achieve the best score for the model.

1. Model Evaluation:

This step focusses on the model’s performance upon the unseen data i.e. testing sub-dataset. This evaluation is done by

## Output

This code uses the Linear Lasso (L1 Regularization) method in determining the best five features for the job to be done. This program uses the Breast Cancer dataset from the *SKLearn* database to compute the following results.



## Observations

There is an immediate observation that can be noticed is that the efficiency score of the model has improved from the validation set to the final test set. Although, this score is a very minor improvement, it still is significant.

The two major hyperparameters that can be altered which brings out a visible and desirable change to the out are:

* Alpha value of the lasso function
* Train, Validation and Test split ratio

|  |  |  |  |
| --- | --- | --- | --- |
| Alpha value | Train : Validation : Test (ratio) | Efficiency Score | Names of selected features |
| 0.2 | 80:15:5 | 0.487537649389 | Worst fractal dimension; worst symmetry; mean texture; mean perimeter; mean area |
| 0.1 | 70:24:6 | 0.613257188968 | Worst fractal dimension; worst symmetry; mean texture; mean perimeter; mean area |
| 0.4 | 60:20:20 | -0.00460162674695 | Worst fractal dimension; worst symmetry; mean texture; mean perimeter; mean area |

From the table it can inferred that the most significant impacting hyperparameter for the model is the alpha score of the Lasso model. This hyperparameter determines the strength of the regularization model. Although, the ratio of training, validation and test does impact the overall efficiency score, but only to a certain degree of manipulation.

# Problem Statement 4

## Question

This question requires the usage of any of the fundamental regression algorithm to predict disease progression from a dataset using the top 5 best features that perform the best job for the same. The change and tuning done to the input via hyperparameters is to be documented along with the list of chosen final features in the output, the score value of the final model and the other output parameters. It is also important to note down observations on the efficiency of the model in data prediction, one example each from the test data that the model predicts poorly and well, along with a few pointers to improve model’s efficiency.

## Code

The entire code block for the Problem Statement 4 can be found in the appendix of this document.

## Output

The code for this problem focusses on the breast cancer dataset from the *SKLearn* repository as well. Here, the Decision Tree Regressor is used to train the model and a score is drawn for the model’s efficiency. The following the output drawn from a randomly chosen set of hyperparameters.



## Observations

It is to note the above output derived is from the Train: Validate: Test ratio of 70: 15: 15. Upon multiple iteration of change of this hyperparameter, it is noted that the Decision Tree Regressor is providing range of outputs that are falling out of pattern when the validation score (Model Efficiency score) and the test score (Final Score) are compared with one another.

|  |  |  |  |
| --- | --- | --- | --- |
| Train: Validation: Test (Ratio) | Validation Score (Model Efficiency Score) | Model Testing Score (Final Score) | List of top 5 best features |
| 70: 15: 15 | 0.7319263583295914 | 0.5859564164648912 | worst perimeter; worst concave points; mean texture; mean fractal dimension; compactness error |
| 50: 25: 25 | 0.4964539007092198 | 0.6984126984126984 | worst concave points; area error; mean concave points; radius error; worst texture |
| 40: 42: 18 | 0.6461538461538461 | 0.6915584415584415 | worst concave points; worst perimeter; mean fractal dimension; worst fractal dimension; area error |
| 95: 2.5: 2.5 | 0.5679012345679013 | 0.7583333333333333 | worst radius; worst concave points; mean texture; mean concavity; mean radius |

The above table indicates the validation that the Decision Tree Regression isn’t an ideal choice to perform regression on this dataset. It is due to the results that vary in disproportion to the varying hyperparameters. With the certain cases where the training is very high and validation is low or when training is low and validation is the highest, the data points being output cannot be linearly matched for a pattern. Due to this, the list of selected features for each iteration is varying as well. Ideally, the **best-case example** would be the Split ratio of data of **70:15:15** and the **worst** would be **95:2.5:2.5**.

The suggestion(s) that one can provide for model improvement would be:

1. To define the problem further by collecting further information upon the desired output and the original behavior of the data along with trends and patterns.
2. To elaborately define the Decision Tree Regressor along with the path to be travelled by assigning the number of leaves.
3. To assign weights to the features and converting the solution to this problem via a more biased method.
4. To change the algorithm altogether and use a linear model to derive better results.

# Problem Statement 5

## Question

The above problem statements are to be conducted here as well for the cancer dataset and there is a need for the following:

1. 3 plots generation for features that are doing a good job.
2. 3 plots generation for features that are doing a poor job.

By separating classes for both. Added to this, the original dataset is to be split into Training, Validation and Testing. Upon this split dataset Linear Logistic Regression (with L1 Regularization) is to be performed to find a maximum 5 good features. These features will be provided as an input to a classifier model that classifies examples as benign and malignant.

## Code

The entire code block for the Problem Statement 5 can be found in the appendix of this document.

## Output

The program runs over the dataset of Breast Cancer from the *SKLearn* repository. The following are the plots obtained for the three good and three poor performing features. Both the features that perform a good job and a poor job are separated by classes and are plotted alongside each other.

A graph of blue squares

Description automatically generated

Figure 4: Three features that do a good job as per their Logistic Regressor Coefficients

Figure 4 has the score of each feature’s logistic regressor coefficient upon the bar chart plotted The model has selected “Worst Area”, “Worst Radius Feature” and “Worst Texture” to be the features that do a good job in predicting disease progression.

A white rectangular object with black text

Description automatically generated with medium confidence

Figure 5:Three features that do a poor job as per their Logistic Regressor Coefficients

Figure 5 has the score of each feature’s logistic regressor coefficient upon the bar chart plotted The model has selected “Mean Radius”, “Worst Compactness Feature” and “Worst Perimeter” to be the features that do a poor job in predicting disease progression.



The above is also the output from the program upon the terminal. Here, the score of all features is first printed and there after the Model Efficiency Score is printed.

This Model Efficiency score is the score based on the Validation split of the dataset. The five best features are then selected based on the score from the All-features list. The Final score is the based on the Testing split of the dataset. The top three example values for Benign and Malignant is also provided upon using K Nearest Neighbor Classifier (with n\_neighbors = 5).

## Observations

The flow of the program with the plotting of the good and poor features to the validation score and then the testing score, exemplifies that the data learning curve by the model has improved based on the ratio split of the Training, Validation and Testing.

The benign and malignant set of features after the K Nearest Neighbor classification is also displayed along with score to the top three values in each category.

# Project 2 Conclusion

Project 2 has been the most challenging task that has been endured so far. The convolution in understanding and further continued learnings from each problem statement has not only been frustrating but also fruitful towards understanding each model and the concept of splitting data for Training, Validation and Testing in Machine Learning. The problems revolve only around this splitting concept and helps one solidify their reasoning upon the knowledge of these Machine Learning concepts.

# Appendix

## Code 1



## Problem Statement 2

### Code



## Problem Statement 3

### Code



## Problem Statement 4

### Code





## Problem Statement 5

### Code

