Abstract

Developing and testing four classifiers, two deep learning neural networks and two k-nearest neighbor classifiers on gene expression data. Comparing and discussing the developed feature selection algorithm and its effect on classifier success.

Comparing Machine Learning and Deep Learning Classifiers on Gene Expression Data

December 2021

Hannah Nissenbaum

Student Number: 20049077

All source code written for this report and additional plots relating to the algorithms can be found on my [GitHub](https://github.com/hnissenbaum/BioAnalytics).

Table of Contents

[Introduction – Aims and Objectives 2](#_Toc89996396)

[Methodology 2](#_Toc89996397)

[Findings 5](#_Toc89996398)

[Feature Selection 5](#_Toc89996399)

[K-Nearest Neighbor Classifier 6](#_Toc89996400)

[Class Predictions vs True Classes 6](#_Toc89996401)

[Contingency Tables 7](#_Toc89996402)

[Neural Networks 7](#_Toc89996403)

[Class Predictions vs True Classes 7](#_Toc89996404)

[Contingency Tables 8](#_Toc89996405)

[Discussion 8](#_Toc89996406)

[K-Nearest Neighbor Classifier 8](#_Toc89996407)

[Neural Networks 10](#_Toc89996408)

[Narrow Neural Network with Feature Selection 12](#_Toc89996409)

[Overall Results Achieved with 4 Classifiers 12](#_Toc89996410)

[Conclusions and Recommendations 13](#_Toc89996411)

[References 13](#_Toc89996412)

[Appendix A: MATLAB Code for Data Randomization 14](#_Toc89996413)

[Appendix B: Python Source Code 15](#_Toc89996414)

List of Tables and Figures

[Figure 1: A Comparison of the Feature Averages across all datapoints for before and after feature selection. 7](#_Toc90039854)

[Figure 2: Comparison of the Class 1 and Class 0 Prediction Numbers of Each Predictor Compared to the True Predictions 15](#_Toc90039855)

[Table 1,2,3: Results Achieved for KNN Predictors 7](#_Toc89996415)

[Table 4,5,6: Results Achieved for Neural Networks 8](#_Toc89996418)

[Table 7,8,9,10: Contingency Tables for developed KNN with feature selection 10](#_Toc89996421)

[Table 11,12,13,14: Contingency Tables for developed KNN without feature selection 11](#_Toc89996425)

[Table 15,16,17,18: Contingency Tables for Bilayer Neural Network 12](#_Toc89996429)

[Table 19: Contingency Table for Narrow Neural Network 13](#_Toc89996433)

[Table 20: Summary of Accuracy and Fisher's Two Tailed Probability for all classifiers 13](#_Toc89996434)

Section I: Introduction

The objective of this report is to construct a predictor that can accurately classify datapoints in one of two classes. These results will then be compared with a second predictor’s results to determine which predictor can provide the most accurate classification of new datapoints.

The data used to train and test the two predictors contains 7129 features and 60 datapoints and was collected by (Pomeroy, et al., 2001). The features represent specific individual genes, which have all been collected from patients with medulloblastomas, a malignant brain tumor. The gene expressions are calculated and then a log operation is performed on them to normalize the data for processing.

For the purposes of this report, the first group, non-responders will henceforth be referred to as Class 0 and the second group, responders, as Class 1. The genes (rows in the data matrix) will be referred to as features, and the patients (each full set of gene information) will be referred to as datapoints.

# Section II: Methodology

The first step is processing the data. With 60 datapoints, the data can be evenly split for a train set and a test set. The number of datapoints in the non-responders class, Class 0 is 21, whereas the number of datapoints in the responders class, Class 1 is 39. It is prudent to ensure that the split of classes over both datasets is relatively equal to avoid overfitting the data to a specific class. Therefore, the data will be split into two matrices, Class 0 datapoints and Class 1 datapoints. To ensure randomness and objectivity, all separation of the data was randomized using MATLAB random integer generator functions. Due to the uneven values of datapoints in each set, a bias will be assigned with a randomly generated integer between 0 and 1. The value of the bias represents the dataset (training or testing) which will contain 11 datapoints in Class 0 and 19 datapoints in Class 1. The ‘non-biased’ dataset will contain 10 datapoints in Class 0 and 20 datapoints in Class 1. To randomize the data, two strings of unique random integers will be generated, one of length 21 and one of length 39. These strings will serve as the indices for which datapoints are picked to go in each set. The first and last ten randomly generated integers will be used to select datapoints in Class 0 to be placed into the training and test set, respectively. Then, the bias will be consulted, and the biased set will receive the leftover datapoint in Class 0. At the same time, the unbiased set will receive the Class 1 datapoint corresponding to the 39th random integer in the second string. Then, the first and second last 19 integers in the second string will be used to split the datapoints in the Class 1 data, completing both the training set and test set. Both sets will have 30 datapoints, and 7129 features per datapoint. Two 30x1 arrays will be created, one for the training set and one for the testing set, storing the class labels for each datapoint at the same indices as that datapoint is stored.

The accuracy of the classifiers will be measured as a function of the prediction errors over the total predictions made. It will therefore be a value normalized between 0 and 1. Class 0 errors have been defined as those where the predictor selects 1 as the class, however the true class is 0. Conversely, Class 1 errors have been defined as those where the predictor selects 0 as the class and the true class is 1. Class 0 errors are expected to be more common, as there are significantly more Class 1 datapoints in both the training set and the testing set as a function of the raw data.

Four predictors will be used for this report: two K-nearest neighbor (KNN) predictors and two trained neural networks. One of the neural networks and one of the KNN classifiers have been through a feature selection step and the other two will use all 7129 features. Therefore, the feature selection step will be evaluated for validity based on the classifiers that used it, and those that didn’t. Additionally, there will be a direct comparison for each training type using the same number of features.

These two predictors were chosen because KNN represents a classical machine learning technique commonly used for classifiers and neural networks represent the new phase of deep learning gaining in popularity since AlexNet was introduced in 2012. (Krizhevsky, Sutskever, & Hinton, 2012)

The feature selection algorithm will be implemented for one of the KNN predictors and Narrow Neural Network trained and follows (Golub, et al., 1999) suggestions on fair feature selection for gene expressions. The feature selection function is written in python and compiled through Visual Studio Code. It takes in training dataset array that is previously randomized and calculates the number of Class 0 and Class 1 datapoints, storing those values. Then, looping through every cell in the dataset, the algorithm calculates the average values for each feature in Class 0 and Class 1. Then, using those same values, the standard deviation of each cell is calculated from it’s respective average using Equation 1.

After calculating the standard deviation for the feature sums in Class 0 and Class 1, the total feature score is then calculated for every feature using Equation 2, where j is the feature array.

The feature scores equation is such that values that are large and positive will be strongly correlated to Class 0, those that are large and negative will be strongly correlated to Class 1, and those close to 0 provide little to no useful classification information. Therefore, the center values must be immediately filtered out. Furthermore, due to the data classes being 60% Class 1, the features with the largest positive scores are chosen to represent the dataset. Feature selection is also implemented as an additional parameter during training; the algorithm will run through the KNN algorithm and parameter selection for k during the training stage, storing accuracy values for the range 5-500 of the most correlated features to Class 0. It then looks for the maximum accuracy achieved on the training set, and picks how many correlated features to use for testing.

The feature selection is also implemented for the Narrow Neural Network, however, the number of genes chosen for the neural network training is reliant on the number of genes the KNN algorithm uses. The reason for this is that the dataset used for both algorithms was intended to be the same for each of the two predictor pairs.

Both neural networks that will used are from the MATLAB machine learning app, using the classification learner. They will both be trained on the same number of features as their respective KNN predictor pairs.

The KNN predictor was developed specifically for this report. It is written in Python and compiled using Visual Studio Code. The algorithm takes the previously randomized data from a csv file and uses the Skikit-learn python library to fit a KNN model to the training set, looping through k values in a range of 2 to 10. For each of the k values, leave-one-out testing is implemented, cross validating the training accuracy and scoring each k value according to its mean accuracy achieved over 30 folds. The k score array is then evaluated for the maximum score, and the corresponding k is chosen as the best k value. A KNN model is then retrained on the training set, using the best k value, and the accuracy score is evaluated, calculating the percentage of correct predictions over total datapoints. For the KNN predictor with feature selection, the algorithm then uses the best k value for it’s trained KNN model and loops through different models with different feature lengths, finding the maximum accuracy. It then sets the feature number as a parameter, and selects those features from the test set, calculating the accuracy.

# Section III: Findings

## Feature Selection

For feature selection, the largest positive feature scores were targeted. As can be seen in Figure 1, this results in a data selection that has far fewer outliers and normalized averages over the 30 datapoints. Because feature scores prioritize low standard deviation values, the standard deviation for the selected values is either low, or high on either side of zero, cancelling each other out. That is why the average value of most features selected hover closer to zero then the rest of the datapoints.

Chart, scatter chart

Description automatically generated

Figure : A Comparison of the Feature Averages across all datapoints for before and after feature selection.

## K-Nearest Neighbor Classifier

The KNN classifier was run two times, first with the developed feature selection algorithm implemented to solely use the heavily correlated features. The accuracy found for KNN With Feature Selection was 73.3%, with 4 Class 0 errors and 4 Class 1 errors. Leave-one-out cross validation was also implemented when calculating the parameter k.

### Class Predictions vs True Classes

Table : Results Achieved for KNN Predictors

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | Results Achieved with KNN and Feature Selection – 249 Genes Used | | | | Data Point Index | Predicted Class | True Class | | 1 | 1 | 0 | | 2 | 1 | 0 | | 3 | 0 | 0 | | 4 | 0 | 0 | | 5 | 0 | 0 | | 6 | 1 | 0 | | 7 | 0 | 0 | | 8 | 0 | 0 | | 9 | 1 | 0 | | 10 | 0 | 0 | | 11 | 0 | 0 | | 12 | 1 | 1 | | 13 | 1 | 1 | | 14 | 1 | 1 | | 15 | 1 | 1 | | 16 | 1 | 1 | | 17 | 1 | 1 | | 18 | 1 | 1 | | 19 | 1 | 1 | | 20 | 1 | 1 | | 21 | 1 | 1 | | 22 | 0 | 1 | | 23 | 1 | 1 | | 24 | 1 | 1 | | 25 | 1 | 1 | | 26 | 0 | 1 | | 27 | 0 | 1 | | 28 | 1 | 1 | | 29 | 1 | 1 | | 30 | 0 | 1 | | **Accuracy over All Data Points: 73.3%** | | | | |  |  |  | | --- | --- | --- | | Results Achieved with KNN, without Feature Selection – 7129 Genes | | | | Data Point Index | Predicted Class | True Class | | 1 | 1 | 0 | | 2 | 1 | 0 | | 3 | 0 | 0 | | 4 | 1 | 0 | | 5 | 1 | 0 | | 6 | 1 | 0 | | 7 | 0 | 0 | | 8 | 1 | 0 | | 9 | 1 | 0 | | 10 | 0 | 0 | | 11 | 1 | 0 | | 12 | 1 | 1 | | 13 | 1 | 1 | | 14 | 1 | 1 | | 15 | 0 | 1 | | 16 | 1 | 1 | | 17 | 1 | 1 | | 18 | 1 | 1 | | 19 | 1 | 1 | | 20 | 1 | 1 | | 21 | 0 | 1 | | 22 | 1 | 1 | | 23 | 1 | 1 | | 24 | 1 | 1 | | 25 | 1 | 1 | | 26 | 1 | 1 | | 27 | 1 | 1 | | 28 | 1 | 1 | | 29 | 1 | 1 | | 30 | 1 | 1 | | **Accuracy over All Data Points: 66.6%** | | | | |
| Contingency Tables  |  |  | | --- | --- | | Table : Results Achieved with KNN and Feature Selection – 250 Genes Used | Table : Results Achieved with KNN, without Feature Selection – 7129 Genes | | | |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | Contingency Table | | *Predicted Classes* | | *Marginal Totals* | | *Class 0* | *Class 1* | | **True Classes** | *Class 0* | **7** | **4** | 11 | | *Class 1* | **4** | **15** | 19 | | Marginal Totals | | 11 | 19 | **30** | | | |  |  |  |  |  | | --- | --- | --- | --- | --- | | Contingency Table | | *Predicted Classes* | | *Marginal Totals* | | *Class 0* | *Class 1* | | **True Classes** | *Class 0* | **3** | **8** | 11 | | *Class 1* | **2** | **17** | 19 | | Marginal Totals | | 5 | 25 | **30** | |

## Neural Networks

### Class Predictions vs True Classes

Table : Results Achieved for Neural Networks

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | Results Achieved on Narrow Neural Network with Feature Selection – 249 Genes Used | | | | Data Point Index | Predicted Class | True Class | | 1 | 1 | 0 | | 2 | 1 | 0 | | 3 | 0 | 0 | | 4 | 1 | 0 | | 5 | 0 | 0 | | 6 | 1 | 0 | | 7 | 0 | 0 | | 8 | 0 | 0 | | 9 | 1 | 0 | | 10 | 1 | 0 | | 11 | 1 | 0 | | 12 | 1 | 1 | | 13 | 0 | 1 | | 14 | 1 | 1 | | 15 | 1 | 1 | | 16 | 0 | 1 | | 17 | 1 | 1 | | 18 | 1 | 1 | | 19 | 1 | 1 | | 20 | 1 | 1 | | 21 | 1 | 1 | | 22 | 0 | 1 | | 23 | 1 | 1 | | 24 | 1 | 1 | | 25 | 1 | 1 | | 26 | 1 | 1 | | 27 | 1 | 1 | | 28 | 1 | 1 | | 29 | 1 | 1 | | 30 | 1 | 1 | | **Accuracy over All Data Points: 66.7%** | | | | |  |  |  | | --- | --- | --- | | Results Achieved on Bilayered Neural Network without Feature Selection – 7129 Genes Used | | | | Data Point Index | Predicted Class | True Class | | 1 | 0 | 0 | | 2 | 1 | 0 | | 3 | 1 | 0 | | 4 | 0 | 0 | | 5 | 1 | 0 | | 6 | 1 | 0 | | 7 | 0 | 0 | | 8 | 0 | 0 | | 9 | 1 | 0 | | 10 | 0 | 0 | | 11 | 1 | 0 | | 12 | 1 | 1 | | 13 | 0 | 1 | | 14 | 1 | 1 | | 15 | 1 | 1 | | 16 | 1 | 1 | | 17 | 1 | 1 | | 18 | 1 | 1 | | 19 | 1 | 1 | | 20 | 1 | 1 | | 21 | 0 | 1 | | 22 | 1 | 1 | | 23 | 1 | 1 | | 24 | 1 | 1 | | 25 | 1 | 1 | | 26 | 1 | 1 | | 27 | 1 | 1 | | 28 | 0 | 1 | | 29 | 1 | 1 | | 30 | 1 | 1 | | **Accuracy over All Data Points: 70.0%** | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Contingency Tables  |  |  | | --- | --- | | Table : Results Achieved on Narrow Neural Network with Feature Selection – 249 Genes Used | Table : Results Achieved on Bilayered Neural Network without Feature Selection – 7129 Genes Used | | |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | Contingency Table | | *Predicted Classes* | | *Marginal Totals* | | *Class 0* | *Class 1* | | **True Classes** | *Class 0* | **4** | **7** | 11 | | *Class 1* | **3** | **16** | 19 | | Marginal Totals | | 7 | 23 | **30** | | |  |  |  |  |  | | --- | --- | --- | --- | --- | | Contingency Table | | *Predicted Classes* | | *Marginal Totals* | | *Class 0* | *Class 1* | | **True Classes** | *Class 0* | **5** | **6** | 11 | | *Class 1* | **3** | **16** | 19 | | Marginal Totals | | 8 | 22 | **30** | |

The findings in this report are all the result of a single randomized split in the data. However, other randomized iterations may have produced different results. In this dataset, the training set was randomly assigned to have 10 Class 0 datapoints and 20 Class 1 datapoints. After receiving the results, the training set should have prioritized as even a split between Class 0 and Class 1 as possible, therefore the training set should have been forcibly biased to have 11 Class 0 and 19 Class 1 datapoints. The test set for this report had 11 Class 0 and 19 Class 1 datapoints. In a more extensive study, the data could have been randomized for a large number of iterations, conducting training for the 4 predictors for every iteration to produce an average of results.

# Section IV: Discussion

Though the accuracy value for both the KNN with and without feature selection is similar, the quality of predictor varies due to the biases associated with predictors. Feature selection attempts to minimize the bias between classes due to the classification split of the datapoints. Given that approximately 60% of the datapoints are classified into Class 1, a predictor that is 100% biased to Class 1 will automatically get 60% accuracy. Therefore, a more accurate way to evaluate predictors is using Fisher’s exact probability, which considers the correct values, and whether the errors are Class 0 or Class 1. This filters out biased predictors, because the calculated Matthew’s Correlation Coefficient favors evenly distributed marginal totals over skewed ones.

## K-Nearest Neighbor Classifier

A KNN algorithm was developed using the Skikit-learn python library and implementing Leave-One-Out cross validation to determine the best k value. It was trained on the randomized set representing the training data as described above. Two versions of the KNN algorithm are being evaluated in this section, one that performed feature selection, and one that uses all 7129 features from the original data.

**Fisher Exact Probability Test**

Developed KNN with feature selection

Table : 2x2 Contingency Table of True Values Predicted by developed KNN with feature selection

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | Contingency Table | | *Predicted Classes* | | *Marginal Totals* | | *Class 0* | *Class 1* | | **True Classes** | *Class 0* | **7** | **4** | 11 | | *Class 1* | **4** | **15** | 19 | | Marginal Totals | | 11 | 19 | **30** | | **Calculating Matthew’s Correlation Coefficient:**  **Calculating Fisher’s Probability:** |

Table : Contingency table keeping marginal totals and changing values to maximize Matthew’s Correlation Coefficient for KNN

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | Contingency Table | | *Predicted Classes* | | *Marginal Totals* | | *Class 0* | *Class 1* | | **True Classes** | *Class 0* | **11** | **0** | 11 | | *Class 1* | **0** | **19** | 19 | | Marginal Totals | | 11 | 19 | **30** | | **Calculating Matthew’s Correlation Coefficient:**  **Calculating Fisher’s Probability:** |

Table : Contingency table keeping marginal and changing values to minimize Matthew’s Correlation Coefficient for KNN

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | Contingency Table | | *Predicted Classes* | | *Marginal Totals* | | *Class 0* | *Class 1* | | **True Classes** | *Class 0* | **0** | **11** | 11 | | *Class 1* | **11** | **8** | 19 | | Marginal Totals | | 11 | 19 | **30** | | **Calculating Matthew’s Correlation Coefficient:**  **Calculating Fisher’s Probability:** |

Table 10: Contingency tables and Fisher’s Probability (FP) for all other possibilities of Matthew’s Correlation Coefficient for these marginal totals

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  | | --- | --- | --- | --- | |  | 0 | 1 |  | | **0** | 11 | 0 | 11 | | **1** | 0 | 19 | 19 | |  | 11 | 19 |  |   **FP:** 1.8031\*10^ (-8) **(Max MCC)** | |  |  |  |  | | --- | --- | --- | --- | |  | 0 | 1 |  | | **0** | 10 | 1 | 11 | | **1** | 1 | 18 | 19 | |  | 11 | 19 |  |   **FP:** 3.8259\*10^ (-6) | |  |  |  |  | | --- | --- | --- | --- | |  | 0 | 1 |  | | **0** | 9 | 2 | 11 | | **1** | 2 | 17 | 19 | |  | 11 | 19 |  |   **FP:** 1.7217\*10^ (-4) | |  |  |  |  | | --- | --- | --- | --- | |  | 0 | 1 |  | | **0** | 8 | 3 | 11 | | **1** | 3 | 16 | 19 | |  | 11 | 19 |  |   **FP:** 0.0029268 |
| |  |  |  |  | | --- | --- | --- | --- | |  | 0 | 1 |  | | **0** | 7 | 4 | 11 | | **1** | 4 | 15 | 19 | |  | 11 | 19 |  |   **FP:** 0.0234 **(Actual Values)** | |  |  |  |  | | --- | --- | --- | --- | |  | 0 | 1 |  | | **0** | 6 | 5 | 11 | | **1** | 5 | 14 | 19 | |  | 11 | 19 |  |   **FP:** 0.09834 (not included) | |  |  |  |  | | --- | --- | --- | --- | |  | 0 | 1 |  | | **0** | 5 | 6 | 11 | | **1** | 6 | 13 | 19 | |  | 11 | 19 |  |   **FP:** 0.2295 (not included) | |  |  |  |  | | --- | --- | --- | --- | |  | 0 | 1 |  | | **0** | 4 | 7 | 11 | | **1** | 7 | 12 | 19 | |  | 11 | 19 |  |   **FP:** 0.3044 (not included) |
| |  |  |  |  | | --- | --- | --- | --- | |  | 0 | 1 |  | | **0** | 3 | 8 | 11 | | **1** | 8 | 11 | 19 | |  | 11 | 19 |  |   **FP:** 0.2283 (not included) | |  |  |  |  | | --- | --- | --- | --- | |  | 0 | 1 |  | | **0** | 2 | 9 | 11 | | **1** | 9 | 10 | 19 | |  | 11 | 19 |  |   **FP:** 0.09300 (not included) | |  |  |  |  | | --- | --- | --- | --- | |  | 0 | 1 |  | | **0** | 1 | 10 | 11 | | **1** | 10 | 9 | 19 | |  | 11 | 19 |  |   **FP:** 0.0186 | |  |  |  |  | | --- | --- | --- | --- | |  | 0 | 1 |  | | **0** | 0 | 11 | 11 | | **1** | 11 | 8 | 19 | |  | 11 | 19 |  |   **FP:** 0.001384 **(Min MCC)** |

Developed KNN without feature selection

Table : 2x2 Contingency Table of True Values Predicted by developed KNN without feature selection

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | Contingency Table | | *Predicted Classes* | | *Marginal Totals* | | *Class 0* | *Class 1* | | **True Classes** | *Class 0* | **3** | **8** | 11 | | *Class 1* | **2** | **17** | 19 | | Marginal Totals | | 5 | 25 | **30** | | **Calculating Matthew’s Correlation Coefficient:**  **Calculating Fisher’s Probability:** |

Table : Contingency table keeping marginal totals and changing values to maximize Matthew’s Correlation Coefficient for KNN

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | Contingency Table | | *Predicted Classes* | | *Marginal Totals* | | *Class 0* | *Class 1* | | **True Classes** | *Class 0* | **5** | **6** | 11 | | *Class 1* | **0** | **19** | 19 | | Marginal Totals | | 5 | 25 | **30** | | **Calculating Matthew’s Correlation Coefficient:**  **Calculating Fisher’s Probability:** |

Table : Contingency table keeping marginal and changing values to minimize Matthew’s Correlation Coefficient for KNN

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | Contingency Table | | *Predicted Classes* | | *Marginal Totals* | | *Class 0* | *Class 1* | | **True Classes** | *Class 0* | **0** | **11** | 11 | | *Class 1* | **5** | **14** | 19 | | Marginal Totals | | 5 | 25 | **30** | | **Calculating Matthew’s Correlation Coefficient:**  **Calculating Fisher’s Probability:** |

Table : Contingency tables and Fisher’s Probability (FP) for all other possibilities of Matthew’s Correlation Coefficient for these marginal totals

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  | | --- | --- | --- | --- | |  | 0 | 1 |  | | **0** | 5 | 6 | 11 | | **1** | 0 | 19 | 19 | |  | 5 | 25 |  |   **FP:** 0.003241 **(Max MCC)** | |  |  |  |  | | --- | --- | --- | --- | |  | 0 | 1 |  | | **0** | 4 | 7 | 11 | | **1** | 1 | 18 | 19 | |  | 5 | 25 |  |   **FP:** 0.0440 |
| |  |  |  |  | | --- | --- | --- | --- | |  | 0 | 1 |  | | **0** | 3 | 8 | 11 | | **1** | 2 | 17 | 19 | |  | 5 | 25 |  |   **FP:** 0.19799 **(Actual Values)** | |  |  |  |  | | --- | --- | --- | --- | |  | 0 | 1 |  | | **0** | 2 | 9 | 11 | | **1** | 3 | 16 | 19 | |  | 5 | 25 |  |   **FP:** 0.3740 (not included) |
| |  |  |  |  | | --- | --- | --- | --- | |  | 0 | 1 |  | | **0** | 1 | 10 | 11 | | **1** | 4 | 15 | 19 | |  | 5 | 25 |  |   **FP:** 0.2992 (not included) | |  |  |  |  | | --- | --- | --- | --- | |  | 0 | 1 |  | | **0** | 0 | 11 | 11 | | **1** | 5 | 14 | 19 | |  | 5 | 25 |  |   **FP:** 0.08160 |

## Neural Networks

Using the available MATLAB software to train neural networks with the same randomized training sets used for the KNN algorithm testing.

**Fisher Exact Probability Test**

Bilayer Neural Network without Feature Selection

Table : 2x2 Contingency Table of True Values Predicted by MATLAB Bilayered Neural Network

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | Contingency Table | | *Predicted Classes* | | *Marginal Totals* | | *Class 0* | *Class 1* | | **True Classes** | *Class 0* | **5** | **6** | 11 | | *Class 1* | **3** | **16** | 19 | | Marginal Totals | | 8 | 22 | **30** | | **Calculating Matthew’s Correlation Coefficient:**  **Calculating Fisher’s Probability:** |

Table : Contingency table keeping marginal totals and changing values to maximize Matthew’s Correlation Coefficient for Bilayered Neural Network

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | Contingency Table | | *Predicted Classes* | | *Marginal Totals* | | *Class 0* | *Class 1* | | **True Classes** | *Class 0* | **8** | **3** | 11 | | *Class 1* | **0** | **19** | 19 | | Marginal Totals | | 8 | 22 | **30** | | **Calculating Matthew’s Correlation Coefficient:**  **Calculating Fisher’s Probability:** |

Table 17: Contingency table keeping marginal totals and changing values to minimize Matthew’s Correlation Coefficient for Bilayered Neural Network

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | Contingency Table | | *Predicted Classes* | | *Marginal Totals* | | *Class 0* | *Class 1* | | **True Classes** | *Class 0* | **0** | **11** | 11 | | *Class 1* | **8** | **11** | 19 | | Marginal Totals | | 8 | 22 | **30** | | **Calculating Matthew’s Correlation Coefficient:**  **Calculating Fisher’s Probability:** |

Table : Contingency tables and Fisher’s Probability (FP) for all other possibilities of Matthew’s Correlation Coefficient for these marginal totals for Bilayered Neural Network

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  | | --- | --- | --- | --- | |  | 0 | 1 |  | | **0** | 1 | 10 | 11 | | **1** | 7 | 12 | 19 | |  | 8 | 22 |  |   **FP:** 0.0947 (not included) | |  |  |  |  | | --- | --- | --- | --- | |  | 0 | 1 |  | | **0** | 2 | 9 | 11 | | **1** | 6 | 13 | 19 | |  | 8 | 22 |  |   **FP:** 0.2550 (not included) | |  |  |  |  | | --- | --- | --- | --- | |  | 0 | 1 |  | | **0** | 3 | 8 | 11 | | **1** | 5 | 14 | 19 | |  | 8 | 22 |  |   **FP:** 0.3278 (not included) |
| |  |  |  |  | | --- | --- | --- | --- | |  | 0 | 1 |  | | **0** | 4 | 7 | 11 | | **1** | 4 | 15 | 19 | |  | 8 | 22 |  |   **FP:** 0.2185 (not included) | |  |  |  |  | | --- | --- | --- | --- | |  | 0 | 1 |  | | **0** | 5 | 6 | 11 | | **1** | 3 | 16 | 19 | |  | 8 | 22 |  |   **FP:** 0.0765 **(True MCC)** | |  |  |  |  | | --- | --- | --- | --- | |  | 0 | 1 |  | | **0** | 6 | 5 | 11 | | **1** | 2 | 17 | 19 | |  | 8 | 22 |  |   **FP:** 0.0135 |
| |  |  |  |  | | --- | --- | --- | --- | |  | 0 | 1 |  | | **0** | 7 | 4 | 11 | | **1** | 1 | 18 | 19 | |  | 8 | 22 |  |   **FP:** 0.001073 | |  |  |  |  | | --- | --- | --- | --- | |  | 0 | 1 |  | | **0** | 8 | 3 | 11 | | **1** | 0 | 19 | 19 | |  | 8 | 22 |  |   **FP:** 0.00002819 **(Biggest MCC)** | |  |  |  |  | | --- | --- | --- | --- | |  | 0 | 1 |  | | **0** | 0 | 11 | 11 | | **1** | 8 | 11 | 19 | |  | 8 | 22 |  |   **FP:** 0.0129 **(Smallest MCC)** |

### Narrow Neural Network with Feature Selection

A neural network was trained using only the feature selected by the feature selection process. It is therefore a direct comparison to the KNN predictor trained on the same set of datapoints and features.

Table : 2x2 Contingency Table of True Values Predicted by MATLAB Narrow Neural Network

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | Contingency Table | | *Predicted Classes* | | *Marginal Totals* | | *Class 0* | *Class 1* | | **True Classes** | *Class 0* | **4** | **7** | 11 | | *Class 1* | **3** | **16** | 19 | | Marginal Totals | | 7 | 23 | **30** | | **Calculating Matthew’s Correlation Coefficient:**  **Calculating Fisher’s Probability:** |

Table : Summary of Accuracy and Fisher's Two Tailed Probability for all classifiers

|  |  |
| --- | --- |
| **Overall Results Achieved with 4 Classifiers** | |
| Results Achieved with KNN and Feature Selection  Accuracy over All Data Points: 73.3%  Fisher’s Exact Two Tailed Probability: 0.0465 | **Results Achieved with KNN, without Feature Selection**  Accuracy over All Data Points: 66.7%  Fisher’s Exact Two Tailed Probability: 0.3268 |
| Results Achieved on Narrow Neural Network with Feature Selection  Accuracy over All Data Points: 66.7%  Fisher’s Exact Two Tailed Probability: 0.3717 | **Results Achieved on Bilayered Neural Network without Feature Selection**  Accuracy over All Data Points: 70.0%  Fisher’s Exact Two Tailed Probability: 0.104 |

Figure : Comparison of the Class 1 and Class 0 Prediction Numbers of Each Predictor Compared to the True Predictions

Though accuracy tended above 50% in the four classifiers, the results aren’t as promising as they seem. This is because both the training set and the test set had a strong bias towards Class 1 (66% of the datapoints in each set were Class 1, whereas only 33% percent were Class 0. Though the predictor training resulted in a strong Class 1 bias, this was not reflected in the accuracy scores because much of that bias was interpreted as correct predictions. If a predictor had predicted 100% Class 1 datapoints, showing complete Class 1 bias, the test set accuracy score would have still resulted in a 66.6%. However, the negative impacts of this bias on the predictors are represented in the Fisher’s exact probabilities. Fisher’s probability is a measure of independence for a predictor’s predictions by looking at the ratio of predictions made for each class. The independence of predictions can be defined as the absence of reliance on the total number of predictions per class. The more biased a predictor is, the more predictions it will make for the favored class, minimizing the predictions made in the unfavored class. This makes for uneven predicted marginal totals, differing significantly from the true marginal totals which severely effects the success of Fisher’s exact probability. The predictor with the highest accuracy score – the KNN algorithm using feature selection, showed the lowest two tailed probability at only 0.0465.

Analyzing the results found from all tested predictors, none of the four were able to classify the patients with reliable accuracy while being unbiased. However, bringing attention to the number of total Class 0 datapoints predicted by each prediction, the KNN predictor that used the results from feature selection predicted by far the largest number of Class 0 datapoints. Furthermore, the KNN predictor using feature selection had an accuracy of 70% predicting the Class 0 datapoints, significantly more then any of the other predictors tested. Because the feature selection focused solely on features that were strongly correlated to Class 0 in the training set, it follows that the predictors using it would be more inclined to predict Class 0. This could mean that the reason behind the inaccuracy and bias of the predictors tested in this report are due to the inequalities in the dataset, not in the inability of classifiers or neural networks to find patterns in the feature data.

# Conclusions and Recommendations

This report does not clearly answer the question of whether machine learning or deep learning is a better method to determine the likelihood of whether a medulloblastoma will have a positive outcome for their treatment. However, there is enough evidence that, given a larger, more evenly distributed dataset, either method would be successful at predicting patients’ outcomes. Given more time, additional feature selection algorithms would be beneficial, to learn information about which genes can be infallible markers for each class. The specific genes, and their meaning in the contribution to predicting each class is the most meaningful data to be extracted from this type of analysis. Ranking genes and associating them with the implications of cancer and cancer treatment could be instrumental in the development of an early detection system in the future.

This analysis shows that further than the additional presence of Class 1 datapoints in the data, the gene expression values correlate significantly more with Class 1 then Class 0. Therefore, finding genes that are strongly correlated with Class 0 and classing all other datapoints into Class 1 was the best approach to attempt as unbiased a predictor as possible.

# References

Golub, T., Slonim, D., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J., . . . Lander, E. (1999). Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science, 286*(5439), 531-537. doi:https://doi.org/10.1126/science.286.5439.531

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet Classificaiton with Deep Convolutional Nerual Networks. *Advances in Neural Information Processing Systems, 25*.

Lowry, R. (1998-2021). *2x2 Contingency Table*. Retrieved from Vassar Stats: http://vassarstats.net/tab2x2.html

Pedregosa, F., Varoquaux, G., Gramfort, A., Thirion, B., Grisel, O., Blondel, M., . . . Duchesnay, E. (2011). Skikit-learn Machine Learning in Python. *Journal of Machine Learning Research, 12*, 2825-2830. Retrieved from Ski-Kit Learn.

Pomeroy, S. L., Tamayo, P., Gaasenbeek, M., Sturla, L., Angelo, M., Mclaughlin, M. E., . . . Golub, T. (2001). Prediction of central nervous system embryonal tumour outcome based on gene expression. *Nature*(415), 436-442. doi:https://doi.org/10.1038/415436a

Thompson, C., & Shure, L. (1995-2021). *Machine Learning App - Mathworks*. (Mathworks, Producer) Retrieved 2021, from MATLAB R2021a.

# Appendix A: MATLAB Code for Data Randomization

Text, application

Description automatically generated

Text

Description automatically generated

Graphical user interface, application

Description automatically generated

Text, application

Description automatically generated

# Appendix B: Python Source Code

All other source code discussed, written, and included in trial runs for these classifiers can be found on my GitHub – [Link Here.](https://github.com/hnissenbaum/BioAnalytics)