1 Abstract

In this assignment, we analyzed the performance of two machine learning methods – K-Nearest Neighbors and Decision Tree – on two separate benchmark data sets. We found both models to perform very similarly in regards to both datasets, with both K-Nearest Neighbors and Decision Tree models being significantly more accurate at predicting death in hepatitis patients compared to detecting signs of diabetic retinopathy in the Messidor image set. To get the best possible performance metrics we compared the accuracy of different hyper parameters (number of neighbors and maximum tree depth) and compared the performance of a few different cost/distance functions. Using both KNN and decision tree, we also extracted several key features of both datasets to plot decision boundaries.

2 Introduction

The main tasks given in this assignment was to i) acquire, preprocess and analyze the Hepatitis dataset and the Diabetic Retinopathy Debrecen datasets, ii) implement K-Nearest Neighbors and Decision Tree machine learning models from scratch iii) determine hyperparameters with validation sets and iv) compare both the performance of the models and the importance of features of the datasets through experimentation. The experimentation for KNN included testing different k hyperparameter values (quantity of neighbors), test Euclidean and Manhattan distance functions, and experimenting with different optimizations with the intention of increasing accuracy performance (standardizing features, weighted KNN implementation, etc.). The experimentation for decision tree included testing different maximum depth hyperparameter values and test multiple cost functions (Gini index, misclassification cost, and entropy). We also found key features using random forest classifier feature importance metrics (mean impurity decrease) and using these features to create a decision boundary plots. We then compare which machine learning method performed best on both datasets, and what factors played a role in optimizing the accuracy of both KNN and DT.

3 Methods

The simple idea behind the K-Nearest Neighbors machine learning method is to train the model by storing value of each feature as vectors, each of which have a corresponding class label. Then, with the use of a distance function (e.g., Euclidean, Manhattan, Hamming, etc.), the testing phase consists of finding the k closest samples to the test vector and classifying this vector based on the most frequent classification of k neighbors.

A decision tree is a binary tree with nodes that have a threshold and a feature to compare to the given threshold. To train a tree, we calculate the cost of a

split along a given threshold and feature. The cost is determined by our given cost function (misclassification, entropy, Gini index), and a split is chosen when a minimum is found. Splitting stops when a stop condition is reached such as maximum depth, minimum leaf instances and cost reduction. These hyper-parameters are determined using a validation set, as deciding using the training set leads to over-fitting and losing generality.

Then, starting from the root of a trained tree, unknown data can be classified by either going to the left or right child depending on the given threshold until a leaf is reached.

4 Datasets

The data sets were processed at the beginning by removing any malformed data such as removing all data sets with missing features. All the values of each feature was processed and stored as a float value in both our models.

The distribution of the Class for the Hepatitis data set showed that 67 patients lived and 13 died. The distribution of the Class for the Diabetic Retinopathy Debrecen dataset showed that 611 patients contained signs of DR and 540 patients had no signs of DR. These class distributions were obtained by using the valuecounts function on the CLASS column of the respective data frames.

For the Hepatatis data set the age distribution of the CLASS was plotted. It can be seen that there is much more variance in age for the patients that died and the patients that lived are concentrated between age 40-60.

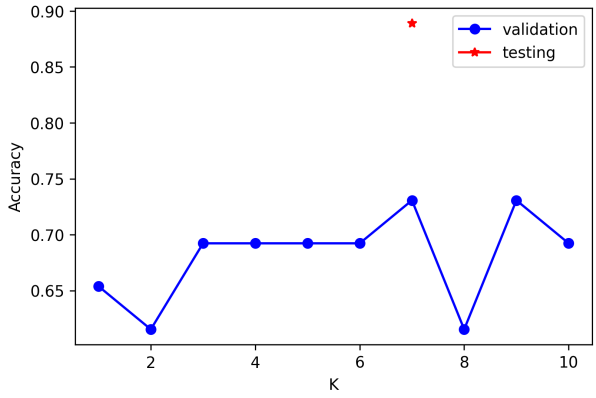
For the Messidor data set the distributions of several of the numerical features(MA-2,MA-4,EXUDATES-8,Euclidean Distance,DIAMETER OF OPTIC DISC) based on the CLASS were plotted. It can be seen that there is very little difference between the two distribution plots of EXUDATES-8, Euclidean Distance, DIAMETER OF OPTIC DISC which may mean that there importance is not as high as other features. The greatest difference between the 2 distribution plots of the CLASS was seen when plotting the MA-2 and MA-4 distribution. The is a clear variance in the data betweent the 2 classes. This may suggest that MA plays a much bigger role in identifying DR signs in patients.

The basic statistics of both data sets were shown with the .describe() function.

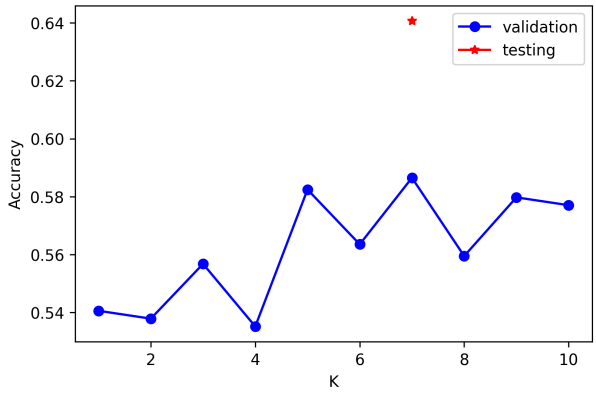
5 Results

With the hyper-parameter of $K = 1$ and euclidean distance function, the KNN model recorded a test accuracy of 81.48% on the hepatitis dataset and a test accuracy of 63.28% on the Messidor dataset. We tried the Manhattan distance function to see if the test accuracy would improve, but the KNN model performed slightly worse with a test accuracy of 77.78% on the hepatitis dataset and 62.24% on the Messidor dataset. We then decided to further split the training data in half to create a validation set to choose the best value for our K

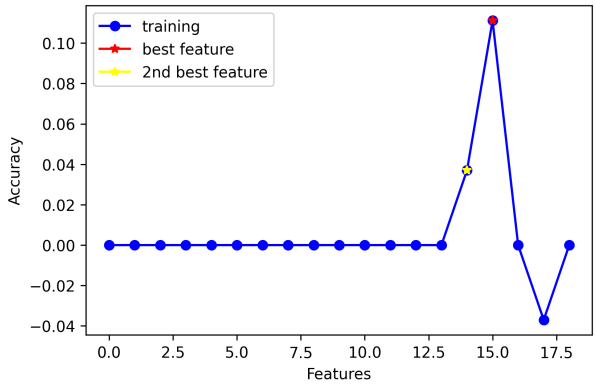
hyper-parameter (from a a range of 1 to 10). For the hepatitis dataset we found that $K = 7$ performed best on the validation set and produced a better test accuracy of 88.89%:



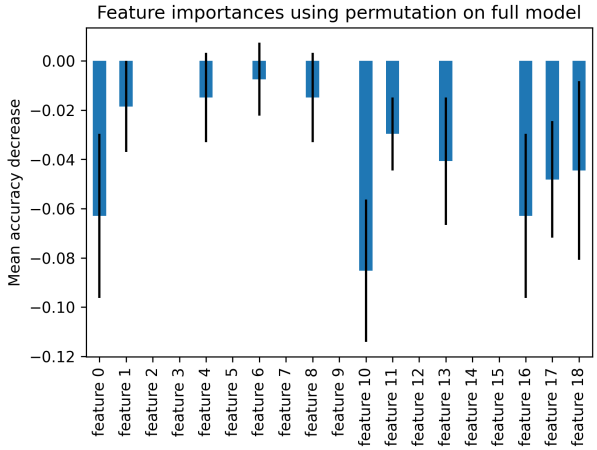
For the Messidor dataset we also found that $K = 7$ performed best on the validation set and produced a better test accuracy of 64.06%:



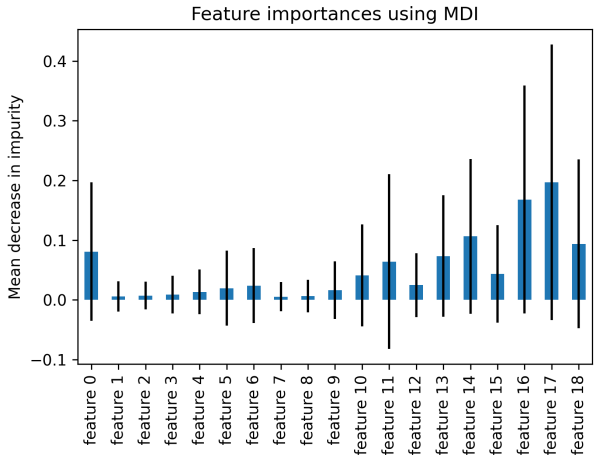
To plot the KNN decision boundaries, we first needed a method to narrow down the two most important features in each dataset since KNN doesn't support any native feature selection. We tried 3 methods on the hepatitis dataset of generating important features. The first was implementing a permutation feature importance algorithm from scratch. This involved permuting the values for each feature separately and comparing which feature permutation caused the highest accuracy decrease. From this first method we obtained that Alk Phosphate (column 14) and SGOT (column 15) were the two most important features:



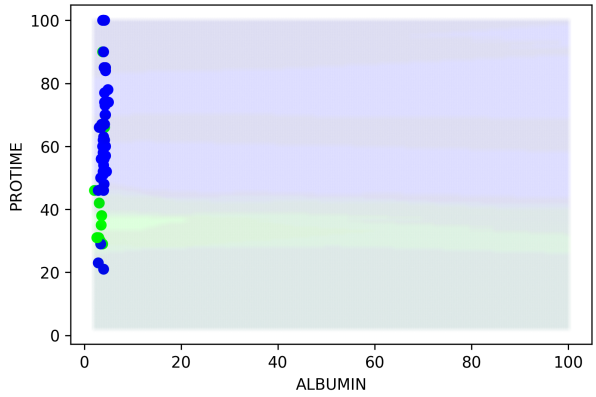
The second method was using sklearn RandomForestClassifier and permutation importance and measuring the mean accuracy decrease for each feature:



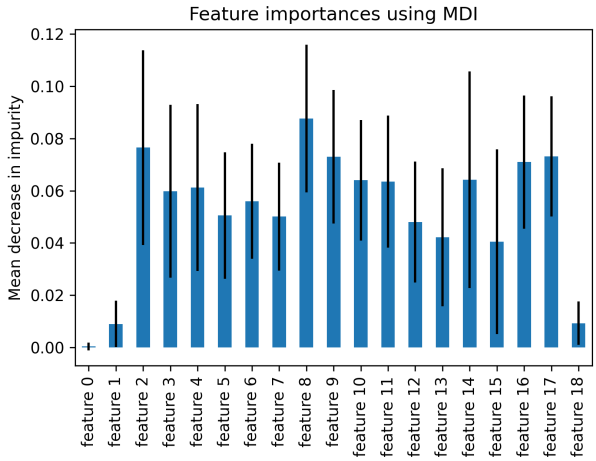
The third and last method was using sklearn RandomForestClassifier and measuring the mean decrease in impurity for every feature:

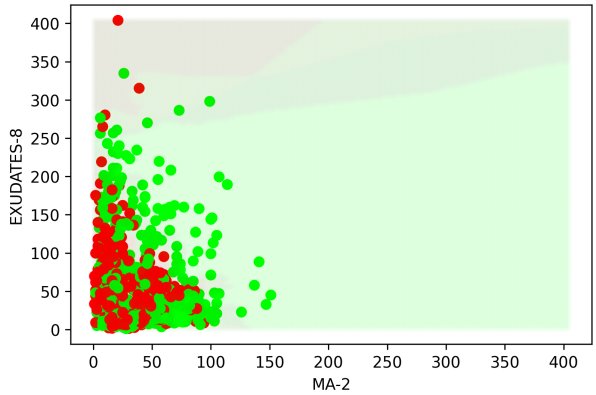


We decided to base our decision boundary on the important features found from the mean decrease in impurity (protime and albumin):



We decided to use mean decrease in impurity again for the Messidor dataset as well, where we found Exudates-8 and MA-2 to be the most important features:





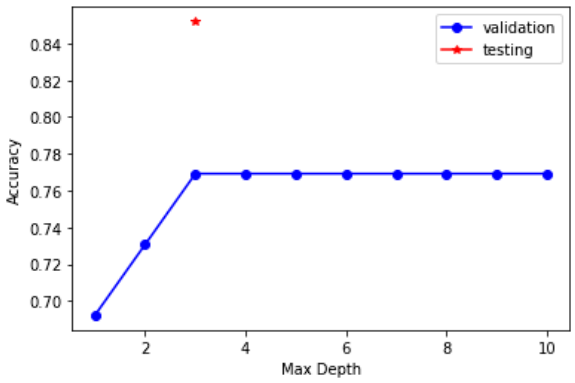
Lastly, we decided to experiment by both standardizing the features and implementing weighted KNN to see if our test accuracy's would increase. However, even after finding the best k value, the test accuracy for the hepatitis dataset remained 88.89% and the test accuracy for the Messidor dataset dropped to 60.68% (compared to 64.06% with regular KNN).

For the decision tree, with max-depth 3 and minimum leaf instances 1 and ignoring cost decrease as an end condition, we found the following accuracies with the noted cost functions:

	Messidor	Hepatitis
Misclassification cost	0.643	0.852
Gini Index	0.641	0.852
Entropy Cost	0.632	0.851

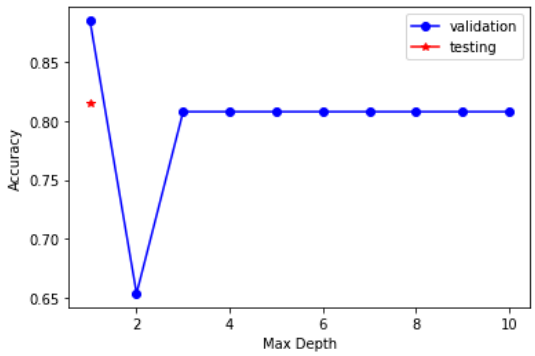
So, the trees that are being constructed are quite similar in accuracy, as the hyper-parameters are restricted in similar ways, which likely affect the tree much more than the cost function.

To determine which hyper-parameters for the decision tree, we tested a range of possible values on a validation set to determine the best one. We did this by setting the other hyper-parameters to constants and by using a consistent cost function. We started the process with the hepatitis data set using the Gini Index as a cost function.



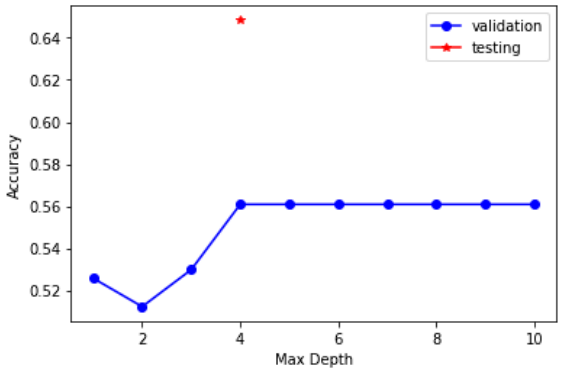
During hyper-parameter testing, the tree never surpasses depth three, which is the reason validation accuracy remains constant. This is because we reach end conditions during the splitting phase in the training other than the tree depth. In the experiment above, the splitting stopped because some leaf nodes had reached the minimum number of instances (1) while the others were pure, so there was no further need to split. So, with Gini Index on the hepatitis dataset, we found that the best maximum depth to be three with a testing accuracy of 85%. Using entropy cost, the results were similar to the test using Gini Index, with depth three having the highest validation testing accuracy.

Then we conducted the test with misclassification cost, where we found that tree depth one leads to the best results on the validation set.

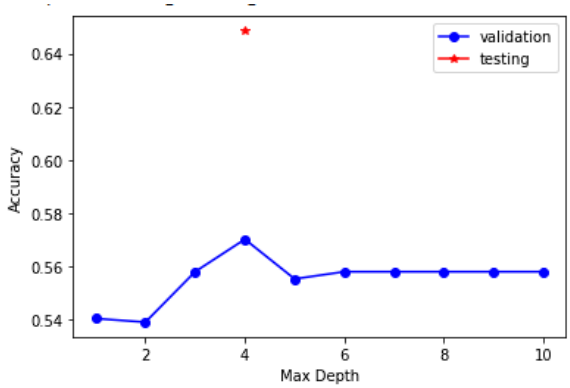


This leads to a lower testing accuracy of 81.4% compared to the Gini Index

To find the tree depth on the Messidor dataset, we did the same process. Starting with the Gini Index with minimum leaf instances set to one and with cost reduction disabled, we get the following results.

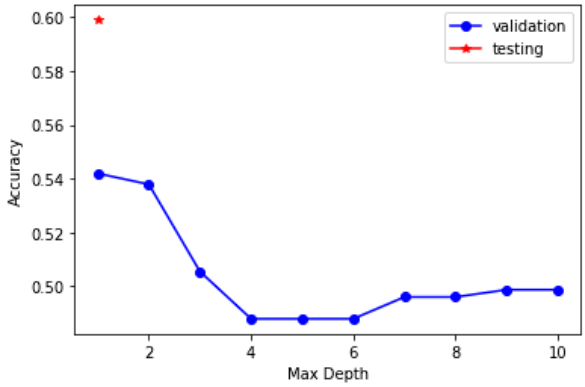


We found that a depth of four achieved the best results on the validation set and has a testing accuracy of 64.8%. Just like the Hepatitis dataset, we found similar results using the entropy cost function, just with a slightly different graph, shown below.



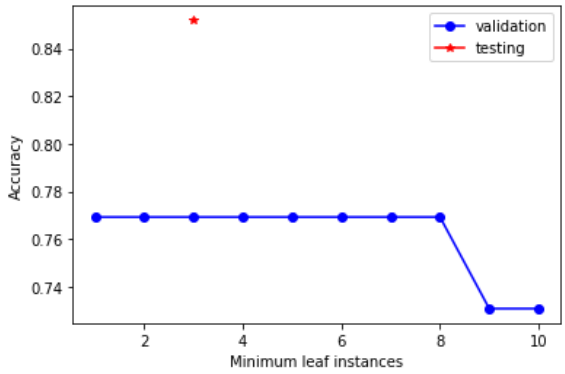
At depth four the tree achieves the highest accuracy on the validation set and has an accuracy of 64.8% on the testing set.

With misclassification cost, the tree depth experiments had a different result.



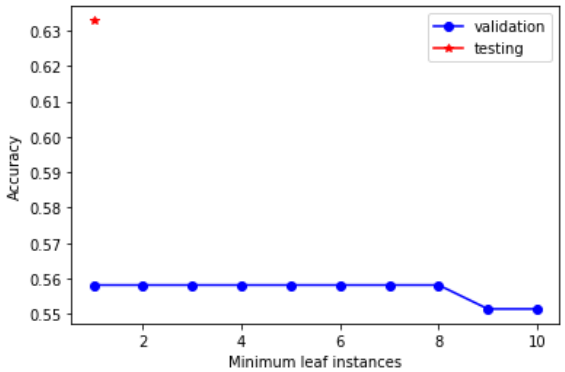
At depth one, the tree performed best on the validation set, and rapidly decreased in accuracy at greater depth. We had a testing accuracy of 59.9% which is much lower than the other cost functions, but this is to be expected with misclassification cost.

To determine the best minimum leaf instances, we fixed the maximum tree depth to three and tested the performance of different minimum leaf parameters with the same cost function on a validation set to select the best value. With the hepatitis dataset, we set used the Gini index

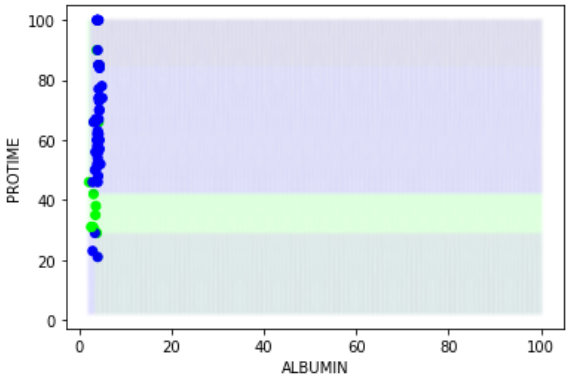


The findings suggest that the minimum leaf instances has minimum effect on the accuracy, but as the minimum leaf instances increase, the accuracy decreases on the validation set.

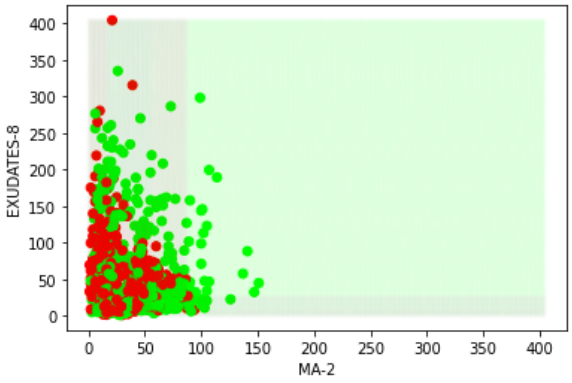
With the Messidor set, we found similar results, with the accuracy on the validation set decreasing when the minimum leaf instances increases.



The decision boundary for the hepatitis set is made by choosing two of the most important features and constructing a tree using only them. The two most important features were found earlier in the KNN section using the random forest approach.



We followed a similar procedure with the Messidor set to create a decision boundary as well.



6 Discussion and Conclusion

From our experiments, we found that the two strategies have comparable accuracies, with KNN having a negligible advantage over the decision tree in the hepatitis dataset (88.8% vs 85.2%). KNN perfomed the best using Euclidean distance function and finding the optimal hyperparameter value K. We tried standardizing the features and implementing weighted KNN in an attempt to increase performance, but the result was relatively the same. For the Messidor datasets, we also have very similar accuracies. So it depends on the situation that they are employed. For example, as the number of features increase, the more time it will take to compute KNN, whereas the time complexity of decision tree predictions scale only with maximum tree depth.

It is hard to make a conclusion that one is strictly better in either datasets as the size of both are relatively small, and the accuracy advantages could be the cause of statistical features and not because of the performance of the model.

7 Statement of Contributors

- Willie Habimana, 260987793:
  - Acquiring and cleaning datasets
  - KNN implmentation(s) and experiments
  - KNN report portion
- William Huang, 260972252
  - Implement greedy node, cost function and decision tree
  - Run experiments on decision tree
  - Write part of the report