COMP551 Assignment 1 Group 16

Rohan Agarwal, Eric Chao, Meilin Lyu

September 2022

1 Abstract

In this paper we investigate the performance of two machine learning models on two benchmark datasets. We use the K-Nearest Neighbors (KNN) classifier and Decision Tree models to perform binary classification and test prediction accuracy. Each model was validated using different hyper-parameters to generate the most accurate results. Various cost functions were evaluated for both models. Decision Tree proved to be a better model with a balanced, direct-representation feature dataset and KNN performed better with a smaller, imbalanced dataset.

2 Introduction

In this assignment we were given the task to run two different classification models on two biomedical datasets.

Dataset 1 is the Diabetic Retinopathy Debrecen (DRD) dataset, which contains features extracted from the Messidor image set that represent either a detected lesion, a descriptive anatomical feature or an image-level descriptor. Previous work by Antal and Hajdu [1] have used this data set to build an ensemble classifier system that achieved 90% accuracy in DR screening.

Dataset 2 is the Hepatitis dataset, which contains biomedical information encoded in both binary and continuous features from patients with Hepatitis along with an outcome label (live or die)[3]. Previous work by Fern and Brodley using Boosted Lazy Decision trees with this data have achieved a peak accuracy of 85.87% [2].

The machine learning models examined were KNN and Decision Tree. We compared the quality of the two on each data set, measured by the performance metric known as area under the receiving operating characteristic (ROC) curve. We ran validation procedures to find the hyper-parameters that produced the greatest accuracy for each model and each data set. In the case of KNN, we tested different values of K closest neighbors, and different cost functions among Euclidean distance, Manhattan distance and Cosine distance. In the case of Decision Tree, we experimented with different combinations of two parameters: the maximum depth of the generated tree and the minimum number of examples in a node. We further evaluated different cost functions among Gini impurity, misclassification rate and entropy. To visualize the behavior of the models, we plotted accuracy as a function of the values of their hyper-parameters.

3 Methods

We used two different models in this assignment:

1. KNN - There is no training involved with this method. Every data point is stored into the model which can be later referenced by the prediction method. While predicting, the classifier calculates the distances between the test sample and all the training points using either the euclidean, cosine or Manhattan distance function. The K closest distances are then extracted and stored. Then, the probability distribution across two classes using the probabilities of the training samples at K closest distances is calculated. Accordingly, the predicted class will be the one with the highest frequency in the distribution.

2. Decision Tree - Decision Trees recursively learns the rules to split the training data points at every node in such a way that minimizes the cost function. Since finding the global optimal split is a NP Hard problem, we use greedy algorithm instead to find the local optimal. During prediction, input data are split based on the splitting rules at every node, until the data reaches the leaf node, in which case the predicted label is the label of the highest occurrence among all labels in the respective leaf.

4 Datasets

For both datasets, we removed samples with missing values, plotted distributions for all features, and transformed all data types to numerical data type. From our distribution plots we gained insights about distribution of variables. Specifically in DRD dataset, we omitted the feature "quality" with constant values throughout all samples because it being constant does not help train the model. Then we one-hot encoded categorical features and standardized numerical features to take values between zero and one. We did not perform feature selection, since it has been shown that it generally does not improve accuracy [4].

5 Results

5.1 KNN on Hepatitis Dataset

5.1.1 Hyperparameter Selection

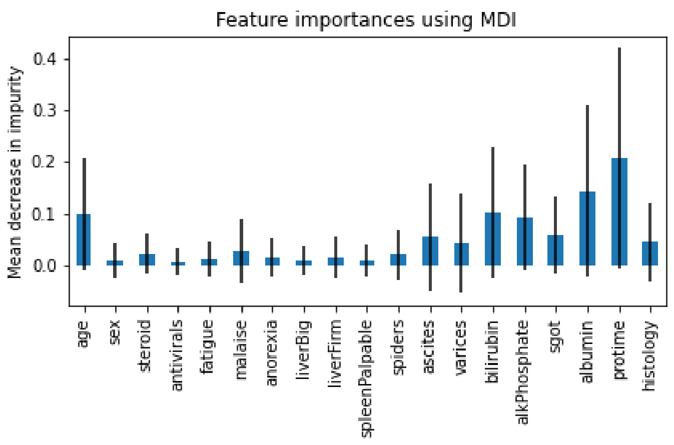
The KNN model was trained on the hepatitis dataset with an 80-20 train-test split. Given the small size of the sample (N=64) we ran 5-Fold cross validation across three different distance functions. This experiment returned the following information:

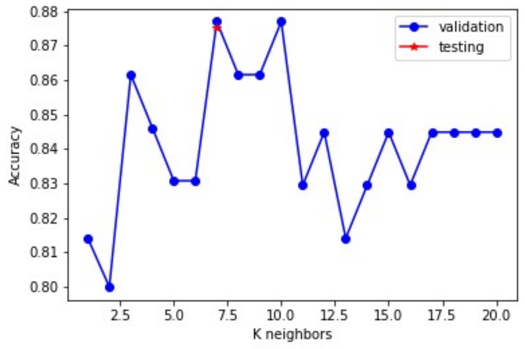
Distance Function	Mean Accuracy	K Value
Euclidean	0.8615384615384615	7
Cosine	0.8602564102564102	8
Manhattan	0.876923076923077	7

From this we come to the conclusion that using the Manhattan distance function returns the best accuracy rate with 87.7%.

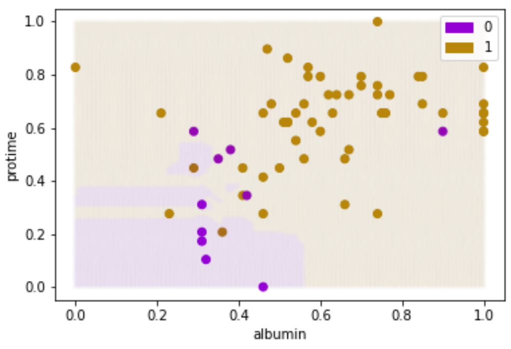
5.1.2 KNN Decision Boundary Graph

To visualize the model's decision boundary we need to select the 2 most relevant features from the dataset using Random Forest Feature importance. The graph below shows that *albumin* and *protime* are the key features we can use to plot the boundary in a 2d graph.





(a) Accuracy over K



(b) Decision Boundary

As seen above in figure 5.1.2(b), we can see that albumin and protime were identified as the highest importance features with age being a close third. The decision boundary graph shows albumin on the x-axis and protime on the y-axis.

5.2 Decision Tree on Hepatitis Dataset

The Decision Tree model was run on the hepatitis dataset using two parameters: *max depth*, which controls the maximum depth of the generated tree, and *min num instances*, which defines the minimum number of examples present in a leaf node. These are both considered stopping conditions when recursively building the tree.

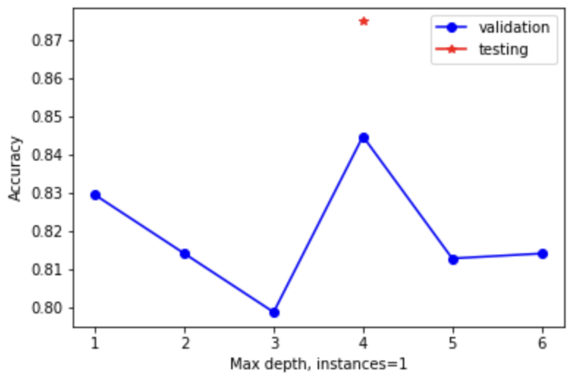
Since we have 2 different parameters we had to test different combinations of values. We performed 5-fold cross-validation with 6 values of depth and 10 values for instance, totaling 60 unique models. We would then pick the model with the best average performance among the folds. This process was done for all three cost functions.

Distance Function	Max depth	Min size	Accuracy Rate
Entropy	2	1	0.84487179
Gini impurity	3	9	0.84358974
Misclassification	4	1	0.84487179

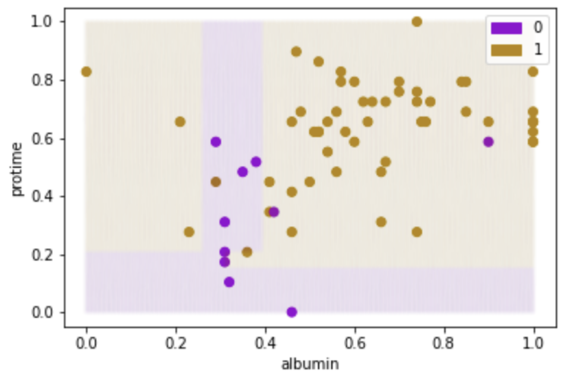
Though Entropy and Misclassification have the same performance over the 5 folds, we kept the latter model, since it has mode leaf nodes and hence is less prone to underfitting.

5.2.1 Decision Tree Accuracy and Decision Boundary Graph

As observed in figure 5.2.1(a), among all different tree depths, depth=4 gives the best accuracy.



(a) Accuracy over different tree depths



(b) Decision Boundary

5.3 KNN on Diabetes Dataset

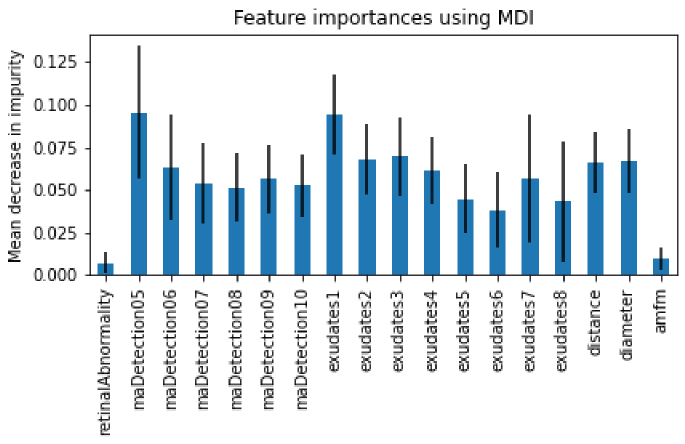
5.3.1 Hyperparameter selection

We first did an 80-20 training-testing split on the dataset and then another 80-20 training-validation split on the training set which we used to fit the model. Moving onward we take KNN with K=13 and Euclidean distance to be our benchmark model.

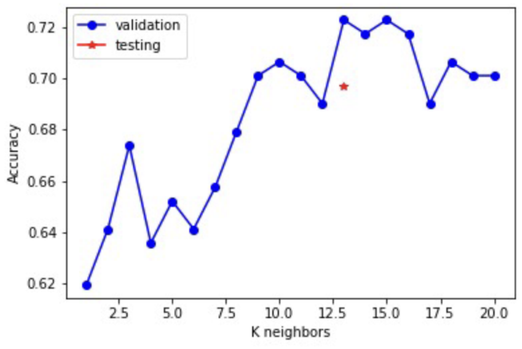
Distance Function	Accuracy Rate	K Value
Euclidean	0.7228260869565217	13
Cosine	0.7282608695652174	10
Manhattan	0.7228260869565217	4

5.3.2 KNN Decision Boundary Graph

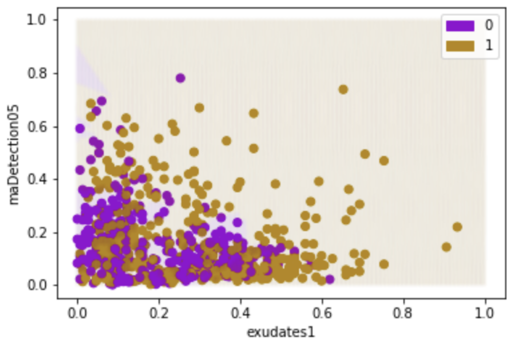
In a similar fashion as above, we generate a random forest tree to identify the two most important features for the boundary graph.



We can identify *maDetection05* and *exudates1* as the two most important features. The following decision boundary graph 5.3.2(b) was generated, showing datapoints closer to origin are more likely to be negative due to their high positive correlation of these features with the target.



(a) Accuracy over K

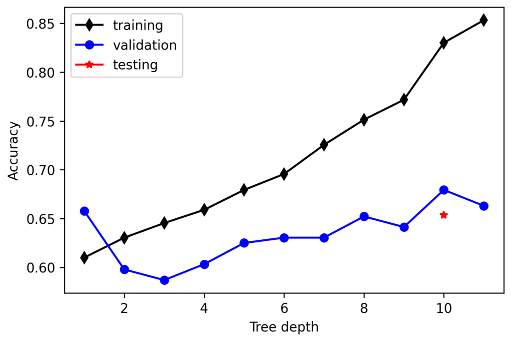


(b) Decision Boundary

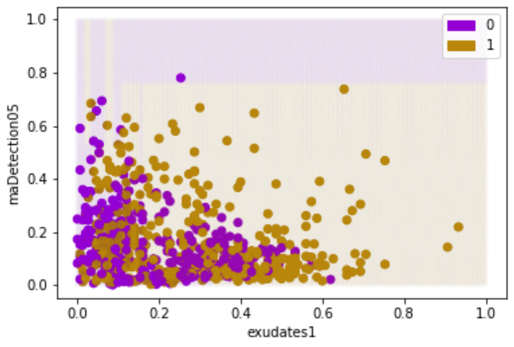
5.4 Decision Tree on Diabetes Dataset

We fitted DT on our training data and tested the accuracy on validation data with 12 possible values for depth and 3 values for minimum leaf instance, a total of 36 possible depth-instance combinations. This process was done for all three cost functions.

Distance Function	Max depth	Min size	Accuracy Rate
Entropy	4	1	0.668
Gini impurity	4	1	0.668
Misclassification	10	1	0.679



(a) Accuracy over different tree depths



(b) Decision Boundary

As observed in figure 5.4(a), we can see that using the Misclassification cost function with depth = 10 and minimum leaf instance = 1 returns the best accuracy. Hence, it was used to create the decision boundary graph using the exudates1 and maDetection05.

6 Discussion and Conclusion

6.0.1 Comparing KNN & DT on the Hepatitis Dataset

The KNN model using the Manhattan distance function with $K = 7$ returns an AUROC score of 0.667. Whereas, the DT model using the Misclassification cost function returns an AUROC score of 0.795.

6.0.2 Comparing KNN & DT on the Diabetes Dataset

The KNN model using the Euclidean distance function with $K = 13$ returns an AUROC score of 0.697. Whereas, the DT model using the Misclassification cost function returns an AUROC score of 0.654.

6.0.3 Future Research

Our results on model comparisons show that DT is an fitter model on the Hepatitis dataset, while KNN performs slightly better on the DRD dataset. However, the performance on the Diabetes data is considerably lower for both models, despite the bigger sample size. The properties of both data sets could account for this difference. Because the DRD data are image-level descriptors and thus not direct measurements of the person as opposed to Hepatitis data, there could be a greater chance that noise is introduced. This amount of noise may not be suitable for single models like KNN and DT. Going forward, research comparing performance of KNN vs. Decision Tree should investigate how different properties of features determine which model gives a better performance.

7 Statement of Contributions

1. Meilin - Decision Tree model implementation, accuracy plots and write up.
2. Eric - Code for visualizations, cross validation and utility functions, notebook sections.
3. Rohan - K-Fold and KNN code, utility functions and write up

Bibliography

[1] Balint Antal and Andras Hajdu. "An ensemble-based system for automatic screening of diabetic retinopathy". In: *Knowledge-Based Systems* 60 (Apr. 2014), pp. 20–27.

[2] Xiaoli Zhang Fern and Carla Brodley. "Boosting Lazy Decision Trees". In: 47907 (July 2003).

[3] Gail Gong. *UCI Machine Learning Repository*. 1988. URL: `http://archive.ics.uci.edu/ml/datasets/Hepatitis`.

[4] Martijn Post, Peter van der Putten, and Jan van Rijn. "Does Feature Selection Improve Classification? A Large Scale Experiment in OpenML". In: (Sept. 2016).