COMP-551 Project 1: Getting Started with Machine Learning

Ding Ma (260871301) Wenxin Shen (260910591) Yiyi Yang (260923855)

Abstract

In this project, we investigated the performance of our self-implemented version K-Nearest Neighbour (KNN) and Decision Trees classifier over the breast cancer dataset and the hepatitis dataset provided by UCI Machine Learning Repository [1][2]. To train our models, we used 10-Fold cross-validation and averaged the accuracy.

For the KNN, we first fined tuned for the number of neighbors then explored the various distance functions. On the cancer dataset, we used $K = 5$ and achieved an accuracy of 96.93%. The hepatitis dataset achieved an accuracy of 89.87% with $K = 12$. This result is based on the mean of four different distance functions. For the Decision Tree classifier, the tree depth was calculated greedily. The cancer dataset had a performance of 96% with a tree depth of 4 whereas hepatitis had an accuracy of 83% with a depth of 2.

We have also compared our classifiers against Scikit-Learn's *DecisionTreeClassifier* and *KNeighborsClassifier*. There was no significant difference in terms of accuracy which indicates that our implementations are correct. However, Scikit-Learn's model is highly optimized and trained a lot quicker than ours.

Keywords: KNN, Decision Trees

1 Introduction

K-Nearest Neighbour (KNN) and Decision Trees have been studied in detail both in the area of pattern recognition and in the area of machine learning. In the vast literature concerning KNN and Decision Trees (also known as classification trees or hierarchical classifiers), we would like to mention two seminal works that bring us inspiration: the one by DB Skalak [3] and those by Esposito et al [4]. The first one presents in depth the combination of nearest neighbor classifiers, while the latter one gives a comparative study of multiple well-known pruning methods on Decision Trees. In this paper, we have used the same data-sets mentioned in the two works above, which are the breast cancer and the hepatitis data-sets provided by UCI Machine Learning Repository [1][2], to investigate the performance of our self-implemented K-Nearest Neighbour (KNN) and Decision Trees classifier. The results are then compared to the ones obtained by using Scikit-Learn's library. Our KNN classifier achieved an accuracy of 96.93% with $k = 5$ for the cancer data-set and accuracy of 89.87% with $k = 12$ for the hepatitis data-set (this result is based on the mean of four different distance functions). In the case of the decision tree, the cancer data-set had a performance of 95.61% with a tree depth of 8 whereas the hepatitis data-set had an accuracy of 83% with a depth of 2.

Several conclusions are made based on our results, for which some examples are listed in the following: first, both data set have followed the same pattern of underfitting, reached their maximum accuracy, and then became overfitted as k-value/tree depth increases; second, the KNN method performs generally better than the decision tree method in the testing phase; third, hyperparameter-tuning may be hard in our case as differences in the resulting accuracies are small and those small oscillations may be caused by other factors.

2 Datasets

Both data sets are described in detail in their respective *.names* file. Since the KNN is sensitive to feature scaling, we decided to normalized all the data points that were not represented by Boolean variables. In the

data, True was represented by 2, and False was represented by 1. We decided to map True as 1 and False as 0 to follow conventional Boolean algebra. We do not see any privacy concerns about those data points as they cannot be linked to any patient. However, there might be ethical concers as the hepatitis data set is heavily skewed towards males which may lead to wrong predictions when female uses the model. It is also unclear to us how the data-sets are collected, and whether consents have been given from the patients who provided those data-sets as this data is open-sourced.

2.1 Cancer

In the Cancer data set, every feature's data point is discreet. Most of the features have a clear definition in their distribution. For instance, the *uniformity of the cell* feature has the majority of its data points labeled as 1. This goes for most of the features other than *bland chromatin* and *clump thickness* where the data points are more spread out. Since this data set does not have a lot of missing values, we simply opted to drop the rows where the are missing entries which represented 2% of our entries.

2.2 Hepatitis

The hepatitis data set contains both categorical and continuous features. The continuous features had to be round to the nearest integer when performing KNN computations. This allows us to use the data points for any distance calculations but the accuracy of the classifier may be downgraded since we have less precision. In this data set, the *protime* was missing 69 times, which is approximately 40% of all the rows. If the dropped the rows where *protime* is missing, there wouldn't be many data points to train on. We tackled this problem in two ways. First, we computed the average of that column and filled the missing data points with the mean. Second, we ran our model by dropping the entire column. This methodology was applied for *Alk Phosphate* too as it was missing a lot of data points. With our experiments, we found out that dropping those two columns lead to the best results. After these operations, 26 data points were removed out of 155. This data set is very small and might lead to overfitting or inaccurate results when testing with unseen data.

3 Results

For our experiments, we used 10 fold cross-validation from Scikit-Lean and averaged out the results from all the folds. For the KNN model, we first fine-tuned for the number of neighbors ranging from 1 to 20 with the Euclidian function. After finding the best K, we retrained the model again over 5 different cost functions. For the decision tree, we simply trained by adjusting the *max depth* of the tree ranging from 1 to 20. During our training, we noticed Scikit-Learn library will train around 600x faster than our models. For a 10 fold cross validation, our implementation runs for around 4 minutes whereas Scikit-Learn's training would finish in under 500ms. This is because our implementations are basic compared to the library where they have a lot of optimizations.

3.1 Accuracy of KNN and Decision Tree

Both data set with different hyper-parameters and cost functions have followed the same pattern of underfitting, reached its maximum accuracy and then became overfitted. For Cancer data set, the highest accuracy we have obtained is 0.969309 with KNN and is 0.957587 with Decision Tree. However, KNN has shown better stability in its results with less fluctuation. Overall, using the same data set, we got better train accuracy with Decision Tree and better test accuracy with KNN.

TABLE 1: Result of Train and Test Accuracy with Different K Values

K value	Cancer		K value	Hepatitis		K value	Cancer		K value	Hepatitis	
	Test	Train		Test	Train		Test	Train		Test	Train
1	0.957651	1.000000	1	0.797436	1.000000	11	0.966411	0.974622	11	0.890385	0.894039
2	0.945951	0.975925	2	0.781410	0.900066	12	0.964940	0.971042	12	0.898718	0.897487
3	0.964898	0.979666	3	0.835897	0.906123	13	0.963470	0.973320	13	0.890385	0.894916
4	0.966368	0.973483	4	0.835897	0.897510	14	0.963491	0.969254	14	0.890385	0.895778
5	0.969309	0.978853	5	0.851923	0.893199	15	0.963491	0.971368	15	0.875000	0.895785
6	0.963491	0.978038	6	0.844231	0.880261	16	0.962042	0.969091	16	0.882692	0.897510
7	0.966390	0.979990	7	0.866667	0.880261	17	0.964962	0.970555	17	0.866667	0.891482
8	0.964940	0.976249	8	0.882692	0.891453	18	0.962042	0.969253	18	0.867308	0.894931
9	0.961999	0.977550	9	0.867308	0.893184	19	0.963491	0.970066	19	0.842949	0.878559
10	0.964962	0.972507	10	0.898077	0.894039	20	0.962063	0.969253	20	0.842949	0.880276

3.2 Different K values for train and test accuracy

The results are based on the average of 10-fold cross-validation and 4 distance functions mentioned below. The test accuracies for the cancer data-set remains approximately at the same level once $k > 2$, while the best range for k appears to be from 10 to 14 in the case of hepatitis. On the other hand, the train accuracies for both data-sets deteriorate slightly as k increases. In reality, it is hard to tell if there is a general trend or not since the number of data is relatively small and the resulting accuracies only differ slightly from each other. In the rest of the experiment, we used $k = 12$ for stable performance.

3.3 Tree depth against Train and Test Accuracy

As the maximum tree depth increases, the training accuracy increases (as shown in Table 2), and we may obtain a result very close or equal to 1.00. However, the accuracy of test data does not follow the same pattern. We can see that for the Hepatitis data set, the test accuracy increases from depth 1 to 2, reaches its maximum at depth 2, and decreases steadily. This observation confirms the fact that larger decision trees can easily overfit the data which results in outputting low training error and high test error.

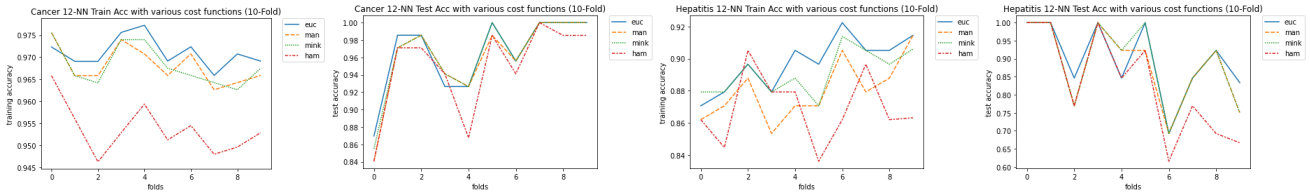
TABLE 2: Result of Train and Test Accuracy with Different Maximum Tree Depth

Depth	Cancer		Depth	Hepatitis		Depth	Cancer		Depth	Hepatitis	
	Test	Train		Test	Train		Test	Train		Test	Train
1	0.909335	0.931025	1	0.781410	0.875088	11	0.953176	0.993657	11	0.796154	0.993966
2	0.945887	0.958193	2	0.836538	0.903522	12	0.953154	0.994633	12	0.796154	0.994828
3	0.950234	0.967140	3	0.820513	0.917308	13	0.951705	0.995446	13	0.796154	0.998276
4	0.951705	0.971694	4	0.811538	0.921611	14	0.950234	0.996260	14	0.796154	0.998276
5	0.957587	0.974948	5	0.811538	0.934542	15	0.948764	0.997398	15	0.796154	0.998276
6	0.953197	0.979177	6	0.811538	0.947465	16	0.948764	0.998049	16	0.796154	0.998276
7	0.953197	0.983733	7	0.796154	0.963823	17	0.948764	0.998699	17	0.788462	0.999138
8	0.956117	0.988289	8	0.796154	0.975877	18	0.948764	0.999187	18	0.788462	0.999138
9	0.953176	0.990403	9	0.796154	0.982766	19	0.947293	0.999512	19	0.788462	0.999138
10	0.953176	0.992355	10	0.796154	0.989655	20	0.947293	1.000000	20	0.788462	0.999138

3.4 Various Distance Functions for KNN

We have attempted Euclidean, Manhattan, Minkowski, and Hamming distance functions for the KNN method, where Minkowski has a p value equals 1.54. This value is inspired by the article written by Shalid et al.[9], in which they completed a thorough comparison of distance measures in the context of health service planning. It is worth noticing that we have a different model than the one they have been using, thus $p = 1.54$ may not result in the best performance. During the trials, we also discover that when p = 1, Minkowski is equivalent to the

Manhattan distance, and in the case where p = 2, it is equivalent to the Euclidean distance. Finally, we could say that the Euclidean function results in the best performance while Hamming function results in the worst.

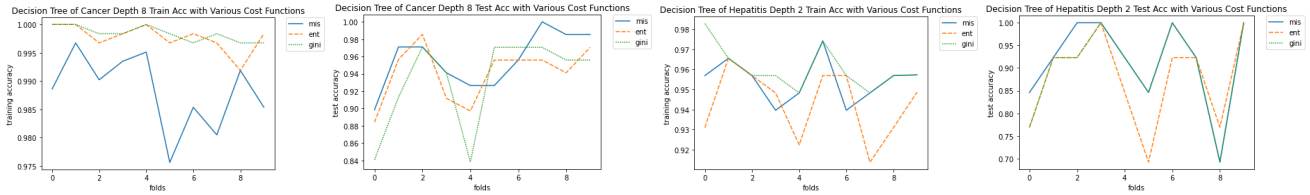


(A) (B) (C) (D)

FIGURE 1: Accuracy Comparison Various Distance Functions for KNN

3.5 Performance of Various Distance Functions for Decision Trees

We have chosen depth that gives the best results to plot the graph. In Figure 2 below we can observe that for Cancer, misclassification cost function performs better on test data set whereas for Hepatitis, using misclassification cost functions gives the best accuracy on both. It is worth mentioning that due to the limitation of greedy heuristics, accuracy obtained in the experiment may not be the best results.

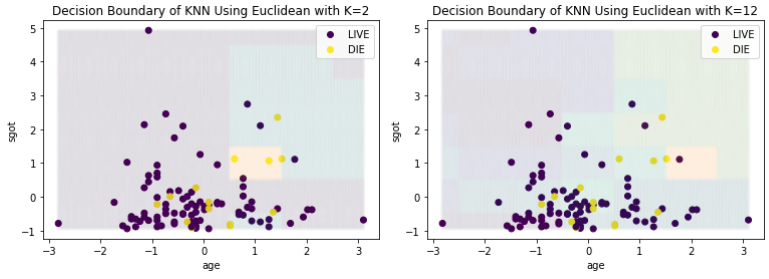


(A) (B) (C) (D)

FIGURE 2: Accuracy Comparison of Various Cost Functions for Decision Tree

3.6 Decision Boundaries for KNN and Decision Trees

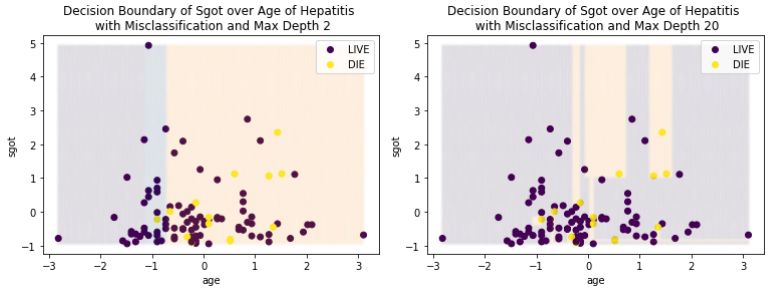
With Figure 3, it is more clear how different K values affect the result. In (A), as we are restraining each point in the data set to only "see" two nearest neighbors, the data set has very little information to decide which label is most frequently appeared in a certain set, which results in lesser different colored regions in the graph compare to (B). When we make $K = 12$, the model gives the most accurate predictions among all the tested values.



(A) (B)

FIGURE 3: Decision Boundaries of KNN Using Euclidean on Hepatitis Data Set with Different K Values

Figure 4 shows the decision boundaries of the same data set with different maximum tree depth. In (A), we can see that with a very small tree depth, the decision regions are roughly divided and a large number of data points are classified in the wrong category. But in (B), the regions are more precisely carved with few data points left misclassified.



(A) (B)

FIGURE 4: Decision Boundaries of Decision Tree Using Misclassification with Different Max Tree Depth

4 Discussion and Conclusion

Through this project, we have built a deeper understanding on the basic concepts of machine learning and explored useful techniques, such as data cleaning, result validation and data visualization. In particular, through the implementation of KNN classifier and decision tree model, we have combined the theoretical knowledge learnt in class with practical experience. We also realize that each step and decision we take in the process of a machine learning project can change the final result.

As beginners, we have encountered many difficulties in the process. For example, we noticed that there are missing features in the hepatitis data set. We then tried both removing the feature from the data set and replacing them with mean value and took the solution that gives the best result. We were also unable to understand the difference between the train accuracy and the test accuracy in the primary stage of the project, which is then solved under the help of TA and the Internet.

For future research, we may try efficient implementations using KD-tree for KNN as suggested in the class note, and add a step of pruning for our decision tree. We may also consider to apply some adaptive methods. For instance, J. Basak has proposed a new Online Adaptive Decision Trees[10], which performs better than widely used models and we believe that it may be implementable in our case. We have also read from an article by S.Sun and R.Huang[11] that an adaptive k-nearest neighbor algorithm (AdaNN) is brought forward to overcome the limitation of the traditional k-nearest neighbor algorithm which usually identifies the same number of nearest neighbors for each test example.

5 Statement of Contributions

Ding cleaned the datasets and made the plots. Wenxin and Yiyi implemented both classifiers. All three of us contributed to the experiments and the report. We would also like to thank Dr.Reihaneh Rabbany and her TA team for their guidance and their python code template[5][6], which help enormously in our project.

References

[1] "UCI Machine Learning Repository: Hepatitis Data Set", Archive.ics.uci.edu. 2021. UCI Machine Learning Repository: Hepatitis Data Set. [online] Available at: http://archive.ics.uci.edu/ml/datasets/Hepatitis. [Accessed 31 January 2021].

[2] "UCI Machine Learning Repository: Breast Cancer Wisconsin (Diagnostic) Data Set", Archive.ics.uci.edu, 2021. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic). [Accessed: 31- Jan- 2021].

[3] D. Skalak, Prototype selection for composite nearest neighbor classifiers.

[4] Esposito, F., Malerba, D., Semeraro, G. and Kay, J., 1997. A comparative analysis of methods for pruning decision trees. IEEE Transactions on Pattern Analysis and Machine Intelligence, 19(5), pp.476-493.

[5] "Google Colaboratory", Colab.research.google.com, 2021. [Online]. Available: https://colab.research.google.com/github/mravanba/comp551-notebooks/blob/master/KNN.ipynb#scrollTo=SX0Oul88hndj. [Accessed: 31- Jan- 2021].

[6] "Google Colaboratory", Colab.research.google.com, 2021. [Online]. Available: https://colab.research.google.com/github/mravanba/comp551-notebooks/blob/master/DecisionTree.ipynb. [Accessed: 31- Jan- 2021].

[7] Harris, C.R., Millman, K.J., van der Walt, S.J. et al., 2020. Array programming with NumPy. Nature, 585, pp.357–362.

[8] Pedregosa et al., 2011. Scikit-learn: Machine Learning in Python, JMLR, 12, pp.2825-2830.

[9] R. Shahid, S. Bertazzon, M. Knudtson and W. Ghali, "Comparison of distance measures in spatial analytical modeling for health service planning", BMC Health Services Research, vol. 9, no. 1, 2009. Available: 10.1186/1472-6963-9-200.

[10] J. Basak, "Online Adaptive Decision Trees: Pattern Classification and Function Approximation", Neural Computation, vol. 18, no. 9, pp. 2062-2101, 2006. Available: 10.1162/neco.2006.18.9.2062.

[11] S. Sun and R. Huang, "An adaptive k-nearest neighbor algorithm," 2010 Seventh International Conference on Fuzzy Systems and Knowledge Discovery, Yantai, 2010, pp. 91-94, doi: 10.1109/FSKD.2010.5569740.