COGS 118A Final Project

Junyi Guo A13815226

Abstract

Many supervised machine learning algorithms have been evaluated in the paper written by Caruana and Niculescu-Mizil. In this paper I choose three of the classifiers: Decision Tree, Random Forest, and K-Nearest Neighbors and test their performances on three different datasets obtained from UCI repository: Electrical Grid Stability Simulated Data Data Set, Default of Credit Card Clients Data Set, and MAGIC Gamma Telescope Data Set. For each classifier and each dataset, I perform three different kinds of partition as well as three times of training and generate the average performances for each classifier, each dataset and each types of partition. After finishing all the experiments, I observed from the performances of each training that Random Forest Classifier usually have the best performances and K-Nearest Neighbor usually have the worst. Also, the dataset size can affect the performances of each classifier. The greater the dataset is, the better the classifiers will perform. Such a result accord with the conclusion obtained in the paper witten by Caruana and Niculescu-Mizil.

1 Introduction

In the paper written by Caruana and Niculescu-Mizil, ten supervised learning algorithms were tested using eight performances criteria (Caruana and Niculescu-Mizil). In this paper, I choose three of the algorithms: Decision Tree, Random Forest, and K-Nearest Neighbors to evaluate their performances on three different datasets and compare my result with the result obtained from paper written by Caruana and Niculescu-Mizil. The three algorithms are performed on three different datasets obtained from UCI repository: Electrical Grid Stability Simulated Data Data Set, Default of Credit Card Clients Data Set, and MAGIC Gamma Telescope Data Set. The training accuracies, validation accuracies and testing accuracies are generated in each training process. This paper mainly focuses on the influences of different classifiers on the training

accuracy and the influences of the different sizes of the training set towards the accuracy of training process.

2 Method

2.1. Dataset

I used three datasets obtained from UCI repository: Electrical Grid Stability Simulated Data Data Set, Default of Credit Card Clients Data Set, and MAGIC Gamma Telescope Data Set. The datasets were originally downloaded as .csv format and were get converted into numpy array later. The Electrical Grid Stability Simulated Data Data Set contains 10,000 data and 14 features. The Default of Credit Card Clients Data Set contains 30,000 data and 24 features. The MAGIC Gamma Telescope Data Set contains 19,019 data and 11 features. All the data from each dataset and all the features of each dataset were used in training processes. For each dataset, the data is shuffled at first and got divided into three different partitions: 80% training and 20% testing, 50% training and 50% testing, and 20% training and 80% testing. Then each algorithms are trained based on the different partitions of the datasets and the average accuracy were generated.

2.2. Learning Algorithms

I include the training of three different machine learning algorithms: Decision Tree, Random Forest, and K-Nearest Neighbors.

Decision Tree: I implement the Decision Tree classifier using the functions from scikit-learn. The parameters are set to default setting except for the hyper-parameter I choose for this classifier, which is the maximum depth for each decision tree. Five different hyper-parameters are tested, which are 11, 12, 13, 14, and 15.

Random Forest: I implement the Random Forest classifier using the functions from scikit-learn. The parameters are set to default setting except for the hyper-parameter I choose for this classifier, which is the maximum depth for each decision tree in the Random Forest. Five different hyper-parameters are tested, which are 11, 12, 13, 14, and 15.

K-Nearest Neighbors: I implement the K-Nearest Neighbors classifier using the functions from scikit-learn. The parameters are set to default setting except for the hyper-parameter I choose for this classifier, which is the number of neighbors. Five different hyper-parameters are tested, which are 1, 4, 8, 12, and 16.

2.3. Performances

The performances are described using the average accuracy obtained in three different trails for each training. I implement the GridSearchCV from scikit-learn to search for the optimal hyper-parameter and use 3-fold cross-validation to get the training accuracy and the validation accuracy. After finding the optimal hyper-parameter I use the parameter to train the classifier again and test the classifier on the testing data and generate the testing accuracy. The training accuracy, validation accuracy and the testing accuracy are averaged among the three trails for each training and are used to evaluate the performances for each classifier on each dataset.

3 Experiment

The following tables shows the average training accuracy, average validation accuracy and the average testing accuracy for each classifier on each dataset. There are total 81 groups of accuracy obtained in the experiment.

		Partition 1(80% training and 20% testing)	**Partition 2(50% training and 50% testing)**	**Partition 3(20% training and 80% testing)**
Decision Tree (Hyper-parameters: depth_list = [11, 12, 13, 14, 15])	Average training accuracy	[[1.] [1.] [1.] [1.] [1.]]	[[1.] [1.] [1.] [1.] [1.]]	[[1.] [1.] [1.] [1.] [1.]]
	Average validation accuracy	[[0.999875] [0.999875] [0.999875] [0.999875] [0.999875]]	[[0.9998] [0.9998] [0.9998] [0.9998] [0.9998]]	[[0.999] [0.999] [0.999] [0.999] [0.999]]
	Average testing accuracy	1.0	1.0	0.99975
Random Forest	Average training accuracy	[[1.] [1.] [1.]	[[1.] [1.] [1.]	[[1.] [1.] [1.]

(Hyper-parameters: rf_depth_list = [11, 12, 13, 14, 15])		[1.] [1.]]	[1.] [1.]]	[1.] [1.]]
	Average validation accuracy	[[1.] [1.] [0.999875] [0.999875] [0.999875]]	[[0.9996] [0.9998] [0.9996] [0.9996] [0.9996]]	[[0.9995] [0.9995] [0.9995] [0.9995] [0.9995]]
	Average testing accuracy	1.0	1.0	0.999625
K-Nearest Neighbors (Hyper-parameters: k_list = [1, 4, 8, 12, 16])	Average training accuracy	[[1.] [0.86068765] [0.83662498] [0.82662499] [0.81824991]]	[[1.] [0.85610081] [0.83179895] [0.82289941] [0.81799895]]	[[1.] [0.8474978] [0.82674391] [0.8119949] [0.81249766]]
	Average validation accuracy	[[0.746125] [0.7565] [0.78225] [0.790875] [0.79025]]	[[0.7374] [0.7534] [0.782] [0.7858] [0.7888]]	[[0.73] [0.755] [0.7705] [0.778] [0.7835]]
	Average testing accuracy	0.786	0.7878	0.7825000000000001

Table 1: Average training, validation, and testing accuracies of each classifier on Electrical Grid Stability Simulated Data Data Set

From Table 1, when using the Electrical Grid Stability Simulated Data Data Set, which contains 10,000 data, and when I use 80% of the data as training set and the other 20% of the data as testing set, the average testing accuracy for both the Decision Tree classifier and the Random Forest classifier are relatively high, which are around 1. The average testing accuracy for the K-Nearest Neighbors classifier is relatively low, which is around 0.786. When I use 50% of the data as training set and the other 50% of the data as testing set, the average testing accuracies for both the Decision Tree classifier and the Random Forest classifier are still relatively high, which are around 1. The average testing accuracy for the K-Nearest Neighbors classifier is relatively low and similar to the result from the first partition, which is around 0.788. When I use 20% of the data as training set and the other 80% of the data as testing set, the average testing accuracies for both the Decision Tree classifier and the Random Forest classifier are lower than the result obtained in the first and second partitions but are still relatively high, which are around 0.999. The average testing accuracy for the K-Nearest Neighbors classifier is lower than the result obtained in the first and second partitions, which is around 0.782.

		Partition 1(80% training and 20% testing)	Partition 2(50% training and 50% testing)	Partition 3(20% training and 80% testing)
Decision Tree (Hyper-parameters: depth_list = [11, 12, 13, 14, 15])	Average training accuracy	[[0.86117361] [0.87127083] [0.882125] [0.89314583] [0.90360417]]	[[0.86994444] [0.88082222] [0.89181111] [0.902] [0.9131]]	[[0.89544444] [0.90988889] [0.92369444] [0.93480556] [0.94758333]]
	Average validation accuracy	[[0.801625] [0.79669444] [0.79108333] [0.7875] [0.77934722]]	[[0.79635556] [0.79217778] [0.78686667] [0.78155556] [0.77568889]]	[[0.76805556] [0.764] [0.75988889] [0.74994444] [0.74405556]]
	Average testing accuracy	0.810888888888889	0.801	0.7915972222222222
Random Forest (Hyper-parameters: rf_depth_list = [11, 12, 13, 14, 15])	Average training accuracy	[[0.87539583] [0.88583333] [0.89427083] [0.90264583] [0.91054167]]	[[0.88576667] [0.89513333] [0.9039] [0.91236667] [0.9231]]	[[0.90408333] [0.91216667] [0.9245] [0.93758333] [0.95158333]]
	Average validation accuracy	[[0.820125] [0.81875] [0.81991667] [0.818875] [0.818125]]	[[0.81686667] [0.8176] [0.81693333] [0.8166] [0.81466667]]	[[0.81333333] [0.8125] [0.81116667] [0.812] [0.81116667]]
	Average testing accuracy	0.8178333333333333	0.8191333333333334	0.8180416666666667
K-Nearest Neighbors (Hyper-parameters: k_list = [1, 4, 8, 12, 16])	Average training accuracy	[[0.999625] [0.81535417] [0.796875] [0.79177083] [0.78804167]]	[[0.9998] [0.815] [0.79773333] [0.7919] [0.78776667]]	[[0.99975] [0.80658333] [0.78916667] [0.78133333] [0.77758333]]
	Average validation accuracy	[[0.6945] [0.76625] [0.77] [0.772375] [0.77345833]]	[[0.69606667] [0.76453333] [0.76873333] [0.77306667] [0.776]]	[[0.67766667] [0.75866667] [0.76116667] [0.76433333] [0.7655]]
	Average testing accuracy	0.7796666666666666	0.7760666666666666	0.7729583333333333

Table 2: Average training, validation, and testing accuracies of each classifier on Default of Credit Card Clients Data Set

From Table 2, when using the Default of Credit Card Clients Data Set, which contains 30,000 data, and when I use 80% of the data as training set and the other 20% of the data as testing set, the average testing accuracy for the Random Forest classifier is the highest, which is around 0.818. The testing accuracy for the Decision Tree classifier is the second highest one, which is around 0.811. The average testing accuracy for the K-Nearest Neighbors classifier is lowest one, which is around 0.780. When I use 50% of the data as training set and the other 50% of the data as testing set, the average testing accuracy for the Random Forest classifier is still the highest and similar to the result from the first partition, which is around 0.819. The testing accuracy for the Decision Tree classifier is the second highest one and slightly lower than the first partition, which is around 0.801. The average testing accuracy for the K-Nearest Neighbors classifier is lowest one and slightly lower than the result from the first partition, which is around 0.776. When I use 20% of the data as training set and the other 80% of the data as testing set, the average testing accuracy for the Random Forest classifier is the highest and similar to the result from the first and second partition, which is around 0.818. The testing accuracy for the Decision Tree classifier is the second highest one and lower than the the result from the first and second partition, which is around 0.792. The average testing accuracy for the K-Nearest Neighbors classifier is lowest one and slightly lower than the result from the first partition, which is around 0.772.

		Partition 1(80% training and 20% testing)	**Partition 2(50% training and 50% testing)**	**Partition 3(20% training and 80% testing)**
Decision Tree (Hyper-parameters: depth_list = [11, 12, 13, 14, 15])	Average training accuracy	[[0.92521634] [0.93840504] [0.95181289] [0.96279987] [0.97165081]]	[[0.94107347] [0.95374577] [0.96596214] [0.97537422] [0.98206966]]	[[0.95897951] [0.96923384] [0.9777798] [0.98382786] [0.98939374]]
	Average validation accuracy	[[0.83288422] [0.82845876] [0.82725381] [0.82249973] [0.82039654]]	[[0.83012585] [0.83065166] [0.82662039] [0.8208364] [0.82293967]]	[[0.81532124] [0.81312999] [0.81041283] [0.80865983] [0.80883513]]
	Average testing accuracy	0.8559411146161935	0.8344199088678583	0.8240886785839466

Random Forest (Hyper-parameters: rf_depth_list = [11, 12, 13, 14, 15])	Average training accuracy	[[0.92651987] [0.9388104] [0.94857047] [0.957279] [0.96500163]]	[[0.93632358] [0.94820695] [0.95772422] [0.96576928] [0.97434015]]	[[0.95950509] [0.96778762] [0.97633447] [0.98422288] [0.99171665]]
	Average validation accuracy	[[0.86644758] [0.86960237] [0.86960237] [0.87163983] [0.87098258]]	[[0.8682301] [0.86949206] [0.87117468] [0.87338311] [0.87590704]]	[[0.86011044] [0.86195109] [0.86247699] [0.86195109] [0.86195109]]
	Average testing accuracy	0.8748685594111462	0.8687346652646338	0.8610672975814931
K-Nearest Neighbors (Hyper-parameters: k_list = [1, 4, 8, 12, 16])	Average training accuracy	[[1.] [0.844857] [0.82997042] [0.82178764] [0.8170884]]	[[1.] [0.8447789] [0.82742673] [0.81864561] [0.81412365]]	[[1.] [0.83250174] [0.80712707] [0.79831889] [0.7930596]]
	Average validation accuracy	[[0.77075255] [0.79487348] [0.80269471] [0.80308906] [0.80335196]]	[[0.76653696] [0.78672836] [0.79713955] [0.79934799] [0.79756021]]	[[0.75729687] [0.77964765] [0.7793847] [0.78122535] [0.78069945]]
	Average testing accuracy	0.818086225026288	0.8017875920084122	0.7908123028391167

Table 3: Average training, validation, and testing accuracies of each classifier on MAGIC Gamma Telescope Data Set

From Table3, when using the MAGIC Gamma Telescope Data Set, which contains 19,019 data, and when I use 80% of the data as training set and the other 20% of the data as testing set, the average testing accuracy for the Random Forest classifier is the highest, which is around 0.875. The testing accuracy for the Decision Tree classifier is the second highest one, which is around 0.856. The average testing accuracy for the K-Nearest Neighbors classifier is lowest one, which is around 0.818. When I use 50% of the data as training set and the other 50% of the data as testing set, the average testing accuracy for the Random Forest classifier is the highest and is lower than the result from the first partition, which is around 0.869. The testing accuracy for the Decision Tree classifier is the second highest one and is lower than the first partition, which is around 0.834. The average testing accuracy for the K-Nearest Neighbors classifier is lowest one and is lower than the result from the first partition, which is around 0.802. When I use 20% of the data as training set and the other 80% of the data as testing set, the average testing accuracy

for the Random Forest classifier is the highest and is lower than the result from the first and second partition, which is around 0.861. The testing accuracy for the Decision Tree classifier is the second highest one and is lower than the first and second partition, which is around 0.824. The average testing accuracy for the K-Nearest Neighbors classifier is lowest one and is lower than the result from the first and second partition, which is around 0.790.

4 Conclusion

From the above performance I obtain, it is reasonable to generate the result that on the average, the Random Forest classifier perform the best, which is accord with the result obtained by Caruana and Niculescu-Mizil. However, different to the result obtained by Caruana and Niculescu-Mizil, the Decision Tree classifier perform the second best and perform better than the K-Nearest Neighbors. The K-Nearest Neighbors classifier perform the worst among the three classifiers trained in the experiment. However, the accuracy for the K-Nearest Neighbors classifier is still high enough, which is around 80% correct generally.

The partition of data in the datasets also contribute to the performances of the classifiers in general. The larger the training set is, the better the classifiers will perform. Different dataset can also have different training result. The Electrical Grid Stability Simulated Data Data Set always have the highest accuracies in training on all classifiers on average. The MAGIC Gamma Telescope Data Set clearly shows the difference in training result among three different classifiers and three different type of partitions.

References

[1] Caruana, R., & Niculescu-Mizil, A. (2006). An empirical comparison of supervised learning algorithms. Proceedings of the 23rd International Conference on Machine Learning - ICML 06. doi:10.1145/1143844.1143865

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

[3] Dua, D. and Karra Taniskidou, E. (2017). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.