Assignment1: Supervised Learning Report

Liang Xu
College of Computing
Georgia Institute of Technology

***Abstract*—This is the assignment 1: Supervised Learning Algorithm report for CS-7641 Machine Learning class in Georgia Tech. The purpose of this report is to develop some understanding of techniques in supervised learning. In this report, five different supervised learning algorithms have been applied to two different datasets. The algorithm includes decision tree with some of pruning, neural networks, boosting, support vector machines, and k-nearest neighbors.There will be various circumstances condition to test the algorithm performance. The programming language used is Python3 and all the algorithms are implemented by using scikit-learn machine learning library.**

I. Introduction

Supervised learning is the machine learning task of inferring a function from labeled training data[1]. The training data consist of pair input objectives and output classes. The supervised learning algorithm, for example, decision tree and SVM, analyzes the training data and "Learn" a meta-model to represent the relationship between input parameters and output classes. This model will be used to map for mapping new or even brand new instances.

In this report, there will be two data sets used to make a comparison between five different supervised learning algorithms. The comparison between the algorithms will be in detail. Training and testing error rates will be obtained, learning graph and model complexity will be plotted. A detailed result analysis will be the main component of this report.

This report is organized as follows: section 2 describe the two data sets. Section 3 presents different algorithm's results. Section 4 discusses the characteristic of each result and detailed comparison

II. Data sets

Both of the data set are come from UCI Machine Learning Repository. One is mushroom data set[2], and another is bank marking data set[3]. Table I is a basic summary of those two data sets. Both of the data sets can be used as binary classification. For mushroom data, we classify is poisonous oe edible, for bank data we predict the client will be subscribe a term deposit. Both of the data sets are from real world, not generated data for particular machine learning study. Mush room data is small and old, bank data is recent data.

TABLE I: Data Sets Summary

Data Set	Instances	Attributes	Result Class	Date	Tasks
Mushroom	8124	22	2(E/P)	1987	Classify
Bank	45211	17	2(Y/N)	2012	Classify

A. Mushroom Data Set

Mushroom records are drawn from The Audubon Society Field Guide to North American Mushrooms (1981). This dataset includes descriptions of hypothetical samples corresponding to 23 species of gilled mushrooms in the Agaricus. Each species is identified as definitely edible, definitely poisonous. The dataset consists of 2 classes, edible and poisonous. The number of edible 4208(51.8%), and the number of poisonous have 3916(48.2%) total has 8124 instances. So we can assume the two datasets have an equal number of samples, so random guess will achieve 50% accuracy.
The reason for me to choose this dataset because one of the articles to use this data to teach decision tree[4]. I found this data is quite good for beginners. The result is evenly distributed and data size is not too big or too small, also the result quite good.

TABLE II: Mushroom Dataset Summary

	class	cap-shape	bruises	veil-type	odor	cap-surf
count	8124	8124	8124	8124	8124	8124
unique	2	6	2	1	9	4
top	e	x	f	p	n	y
freq	4208	3656	4748	8124	3528	3244

Table II is a summary about mushroom data set. 6 features from 23 features have been selected to show the summary. Some of the features are quite interesting for classification. For example, 'bruises' feature has 2 classes, and the top frequency is 4748, which is quite similar to 'Class' feature. However, from Figure 3 (right side) the distribution of 'bruises' v.s 'Class', we found out this feature does not have a high confidence with what we expect, we can say 't' class have a high probability can be classified as 'e' . For feature 'veil-type', it only has one class 'p', this feature will not give us any information to determine the mushroom is an 'e' or 'p'. Let check the 'odor' feature, this feature have 9 classes, from figure 3 (left), we can easily determine class 'e' is on the left side and 'p' on the right side.So, even though the feature have the same number of classes with the final classification result, it does not mean it a good feature to determine the class.
Figure 1 is the summary of the mushroom data set. From this summary graph we can easily found some of the features are quite easy to determine the class of a particular mush room. Figure 2 is a feature importance for each feature. In order to make our machine learning problem become harder I removed some of the killing features 'habitat','population','gill-

color',’spore-print-color’,’odor’,’gill-size’,’stalk-root’,’gill-spacing’,’cap-color’. So in the following report, only 12 features have been used.

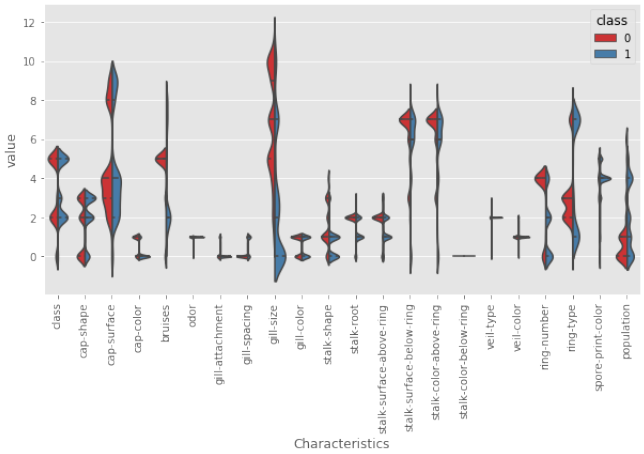

Fig. 1: Mushroom Data Feature Distribution Summary

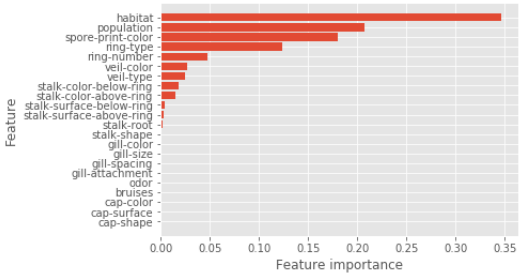

Fig. 2: Mushroom Data Feature Importance for Decision Tree

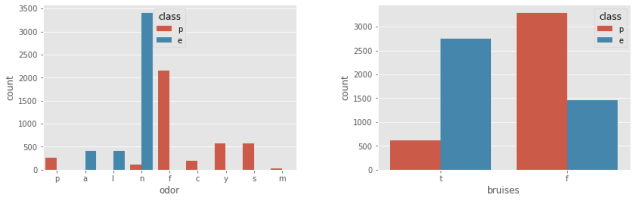

Fig. 3: Odor and Bruises distributions

B. *Bank Marketing Data Set*

The data is related with direct marketing campaigns of a Portuguese banking institution. The marketing campaigns were based on phone calls. Often, more than one contact to the same client was required, in order to access if the product (bank term deposit) would be (’yes’) or not (’no’) subscribed [3]. The bank dataset contains both type variables "Categorical" and "Numerical". for example, age and balance are both ’Numerical’, and job and marital both "Categorical". Figure 5 is the summary for different features distribution. From this summary, we can see the data distribution is hard

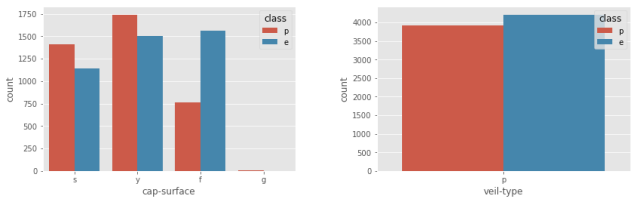

Fig. 4: Cap-surface and Veil-type distributions

to make a direct guess the result, which means the problem is a little bit hard for human.

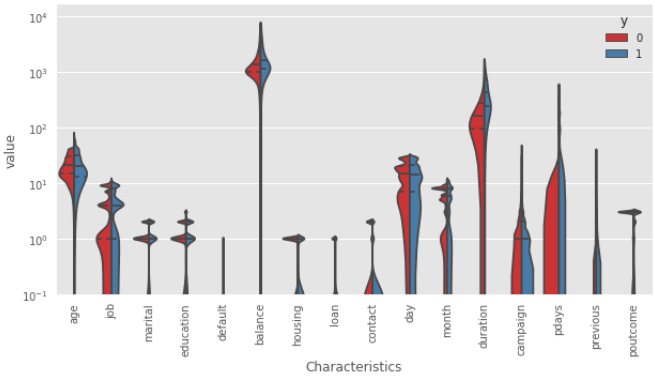

Fig. 5: Bank Data Feature Distribution Summary

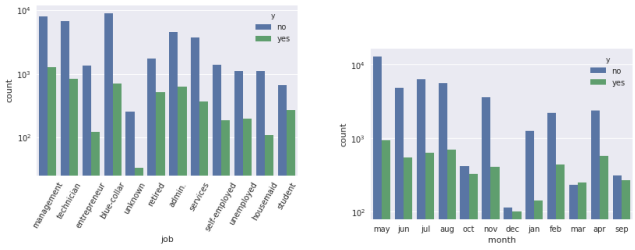

Fig. 6: Categorical distribution example

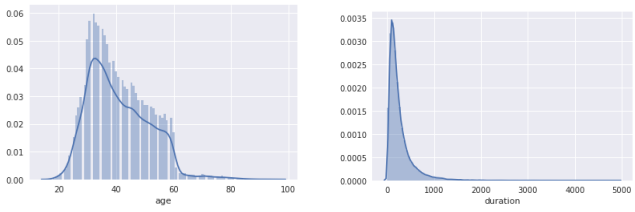

Fig. 7: Numerical data example

Above figure are the display of different category and numerical data. For job and mouthes distribution, the y-axis uses a log scale. Because, in this data set, the number of ’yes’ class only have 5289(11.7%), and ’no’ have 39922(88.3%), so the dataset is highly unbalanced. The reason I chose this data is that of its combination with the Numerical and

Categorical feature, which make the algorithm a little bit different compared with the first one. Also, the data set is unbalanced in terms of the success rate. Only 11.7% of the data classified as "yes". I want to see how machine learning algorithm handles this.

III. Supervised Learning Algorithm Result

In this section, five different machine learning technique will be implemented to the two datasets mentioned in the above paragraph. All the algorithms are directly from the scikit-learn: machine learning library in Python. There is python note book contains all the plots and code. To evaluate the five classification problem, the dataset has been separated to 80/20 for training and testing. For comparison, five different algorithm will be using the same tranning dataset and same testing set.

A. Decision Tree

Decision Trees (DTs) are a non-parametric supervised learning method used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. In this part of the report, I will show some of the accuracy result and some analysis between the two data set.

1) Decision Tree for Mushroom Dataset: The result for mushroom data is very good, even though not all the features have been used for classification. The best testing score we can get is 96.18%, by using 13 layer depth and 13 features decision tree.

Fist let us find out how important each feature used in the decision tree. Figure 2 is the feature importance distribution for the best-scored decision tree. Figure 9 is the tree for this model, for simplicity, I just cut off this tree to 2 level. From the importance and tree figure, we can easily find out the 'ring-type' is on the top of the tree, because it has the largest 'gini' index.

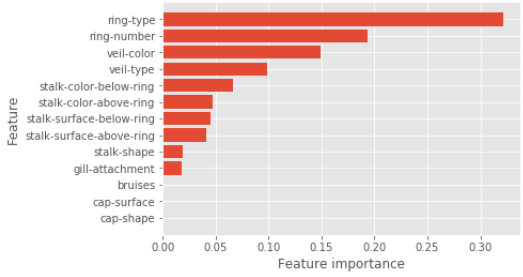

Fig. 8: Feature Importance for Mushroom Dataset

Let us check the learning rate of this algorithm. Figure 10 is the testing set accuracy compare with the number of samples. For this figure, we can easily assume the accuracy of the model is increasing dramatically from 1 to 100, then from 100 to 300 the accuracy increase rate become slow. Finally, the accuracy increasing rate become very slow. This means the heavy learning in the begin of the process, as the amount data increased, the learning become slow.

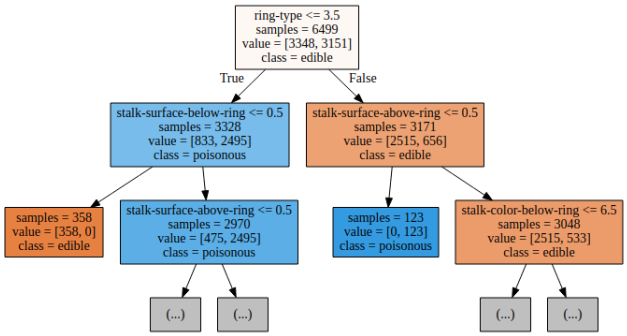

Fig. 9: Best Decision Tree for Mushroom Dataset

Form this figure, we can also found out, with the training data increased the testing accuracy is not decrease, which means the model is over fitting with too mush data. I will use cross validation method to test what I have proposed here.

In the Figure 10, there are two additional curves bedsides Accuracy, there are Precision and Recall. Because this data set is balanced, so the accuracy curve is in the middle of precision and recall, that means the true positive is very high also.

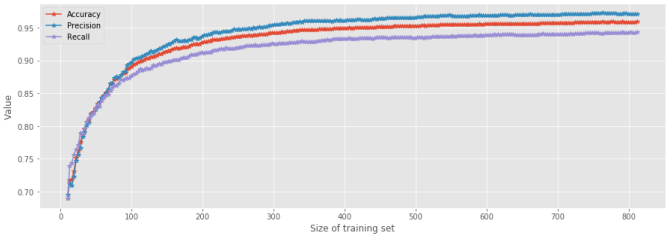

Fig. 10: Learning with Different size Data for Mushroom

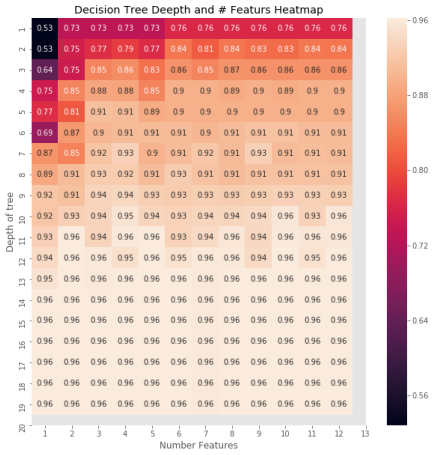

Fig. 11: Model Accuracy Heatmap for Mushroom Data

Figure 11 is the accuracy heat map, the x axis is the number of features, and the y axis is the depth of the tree. According to this graph, the accuracy stop increase even after 13 lairs. However, the accuracy is not decreased also. Thant means the model is not overfitting even we push 20 lairs depth of

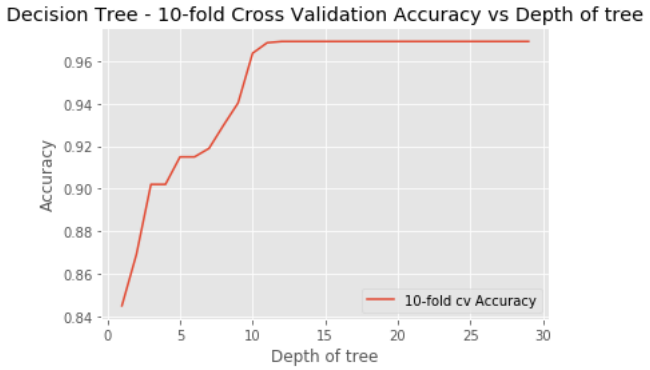

Fig. 12: Cross Validation Curve for Mushroom Dataset

tree node. This has been verified by cross-validation figure 12 also. the accuracy stop increase after around 13. After 13, the accuracy becomes a straight line.

Analysis summary: Decision tree for this data set performance very good, the tree not overfitting by using too much data set, too deep, and too many features. This is because the feature to determine the 'class' is very obvious. In this case, too believe to the model is ok for predicting the class.

2) Decision Tree for Bank Dataset: The bank behaves differently compared with the mushroom dataset. The best score I can get to predict the success by using the decision tree is 90.00$ by using max deoth 7 and max features 11(best result come form the GridSearchCV method). However, 11.7% of the bank data classified as 'yes', our decision tree method only better than just guessing 'No' a little bit. Still, the decision tree is better than random guessing 'Yes' and 'No'. Let discuss what performance graph is for this data set.

First, let us find out what feature is the most important for

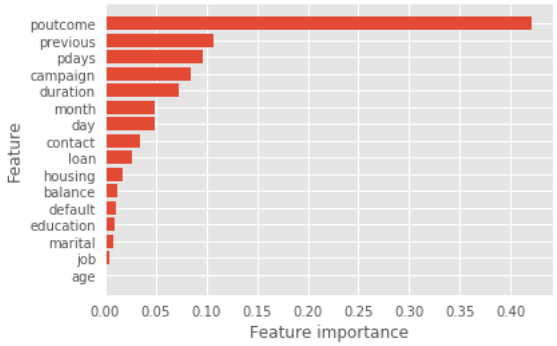

Fig. 13: Feature Importance for Bank Dataset

bank data. The feature 'poutcome' become the most important feature using in the classification, which has 'gini' index more than 0.43. Figure 14 is a smaller version of the decision tree. We can check the top node is not the 'poutcome', this feature probability used in the down level. Figure 15 is the learning curve by using different data size. The accuracy increase dramatically in the beginning. One of the reason is the data set is not balanced. For example, just guess 'No' will also get a high accuracy. So the high learning rate is because the

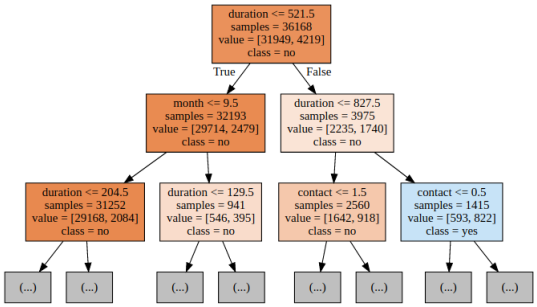

Fig. 14: Best Score Decision Tree for Bank Dataset

unbalanced data.

Other thing is the recall and precision curve is very low compare with the accuracy curve. This is also because the unbalanced data, which means the model did not predict the false positive very will.

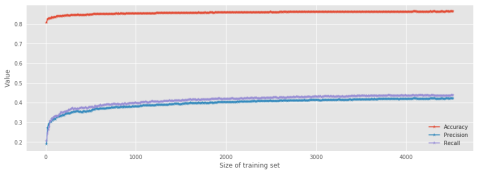

Fig. 15: Bank data learning curve

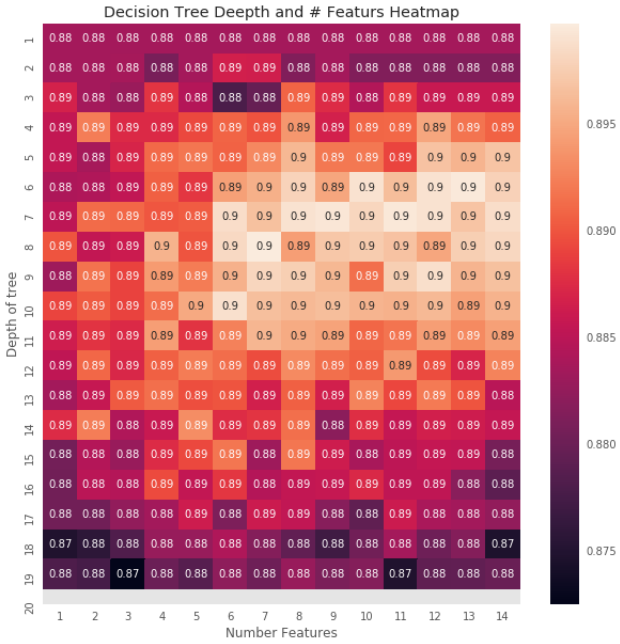

Fig. 16: DT Model Accuracy Heat Map for Bank Dataset

Figure 16 gives us the accuracy heat map for a different number of features and depth of the tree. From this figure, we can find out the high accuracy is not at the bottom right as mushroom data. The high accuracy is in the middle of the graph. The reason is overfitting. By using too complicated

data, the mode becomes too believe the training data set and become too complicated.

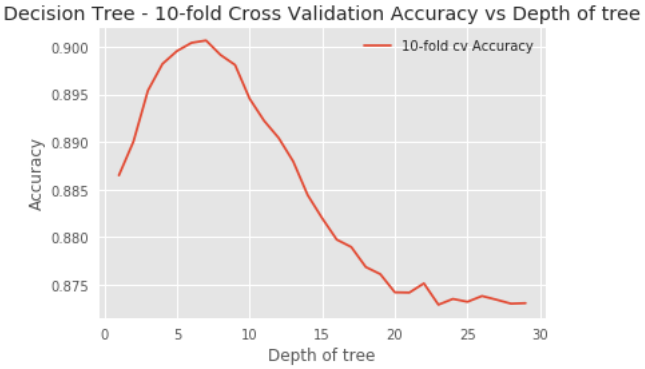

Fig. 17: Cross Validation Curve for Bank Dataset

Figure 17 shows the cross-validation curve for depth of the tree. From this curve, we can easily find out the tree overfitting the after the depth larger than 7. The accuracy decrease by using more complex model. We got the same result by using the heat map Figure 16.

Analysis summary:Decision tree for this data set performance is not very good, however, the performance is much better than random guessing. The tree is nor overfitting with too much data set, however, the tree overfitted by using a large number of features and depth also. form this graph we can find out the power of cross-validation.

B. Boosting

The booing method will use AdaBoost method from Scikit learn library.An AdaBoost classifier is a meta-estimator that begins by fitting a classifier on the original dataset and then fits additional copies of the classifier on the same dataset but where the weights of incorrectly classified instances are adjusted such that subsequent classifiers focus more on difficult cases[5].

1) Boosing for Mushroom Dataset: Using the Boosting method for the Mushroom data performs pretty good which give the same accuracy score compare with the decision tree method, the accuracy is 96.18%. Boosting method is based on the decision tree method, in this implementation, I have set the limitation for the depth to 3 and max_feature to 5.

Most of the performance is quite same with the decision tree. For example the important feature according to Figure 18. For this data set, I did not find too much interesting performance compare boosting result with the decision tree method. However, the boosting take much longer to train and predict. Figure 19 is the testing accuracy compare with different training data size. Boosting method performance is similar with the decision tree. Figure 11 is the testing accuracy heat map, from this heat map, we can easily found out the model perform very well even we use small number of estimator and low learning rate. This means the AdaBoosting method perform very even without the cross validation. So for this question, no need to talk about the cross validation.

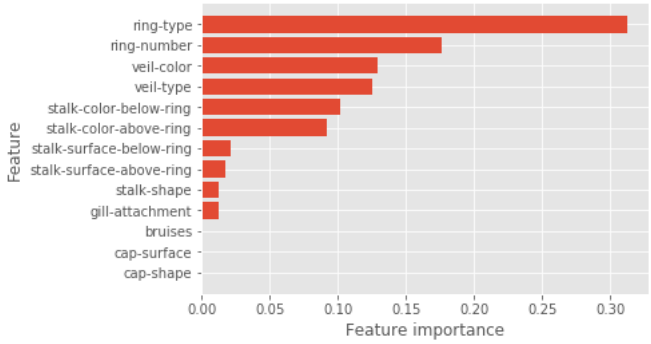

Fig. 18: AdaBoost Feature Importance Graph for Mushroom

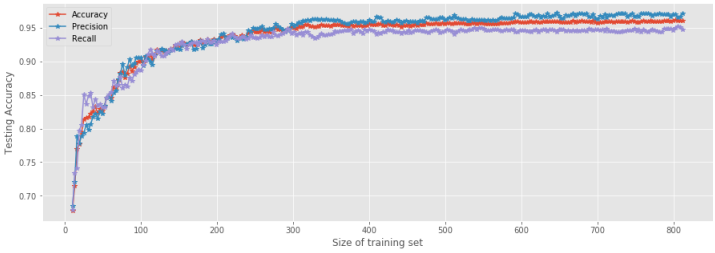

Fig. 19: AdaBoost Learning with different training data set for Mushroom

Analysis summary: The AdaBoost method preform very well, even without too much parameter tunning. This is also one of the advantage of Boosting method. Is this true for every data set or only for this particular dataset. Let us find out in the bank data set in next section.

2) Boosing for Bank Dataset: The boosting method for bank dataset perform better than the decision tree method, The best accuracy score I can get for this dataset is 90.48%, but the training and testing time longer than the decision tree method. From figure 21 importance feature map, is a little bit different than the Figure 13. For the decision tree, the 'poutcome' feature dominate the classification, however, for the boosting method, almost all the feature contribute to the classification.

The learning curve performs a little bit different compared with the decision tree which is Figure 15. Especially for the precision curve, it keep increase pass 0.52

Also the heat map figure 23, also indicate the mode is not easy to overfitting. with number increased, the testing data did not drop. Compare with the decision tree method, which easily gets over fitted by using the deeper decision tree. The best value get from GridSearchCV() is with 80 estimators, and learning rate is 0.3.

For the cross-validation test, I have designed two cases. One is keep the learning rate the same and get 10 folder validation accuracy for estimator [2,100], which is the figure 24. For this figure, we can understand the number of the estimator did not overfit the model. With the number of estimator increase, the accuracy increase. However, the accuracy increasing rate become much slower at 80.

Figure 25 is the boosting learning rate from [0.1 to 1.0].

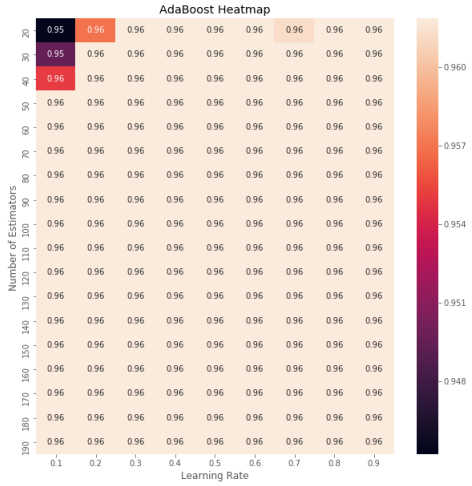

Fig. 20: AdaBoost Estimator and Learning rate Heat Map for Mushroom

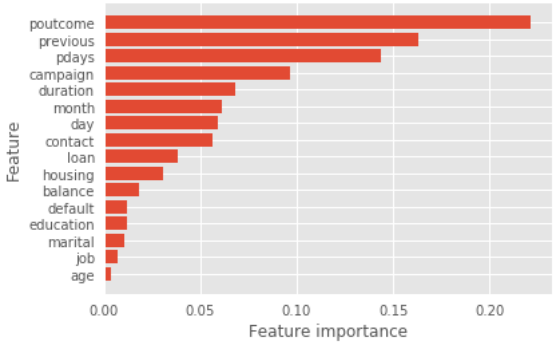

Fig. 21: AdaBoost Estimator and Learning rate Heat Map for Bank

However, from this figure, we can easily find out the model do over with learning rate larger than 0.3. This is a very interesting graph to show that, fine tune the learning rate is very important for some of the machine learning algorithms.

Analysis summary: The AdaBoost method performs much better than the pure decision tree method. Using the same training and testing set, the accuracy increased almost 1.5% which is pretty good for large problems. And the overfitting problem is very well handled by using this method. We only need to set a good learning rate. However, this method is very slow, for detailed comparison, please check the last part of this report. As a method based on the decision tree, boosting method out preform decision tree in terms of the testing accuracy.

C. Neural Network

Advantages of neural networks include their high tolerance to noisy data, as well as their ability to classify patterns on which they have not been trained. The most popular neural network algorithm is the back-propagation algorithm[6]. Neural Network is quite popular in recent years, due to the deep neural network structure improved the machine problem a lot, especial for the large unstructured dataset. For this algorithm,

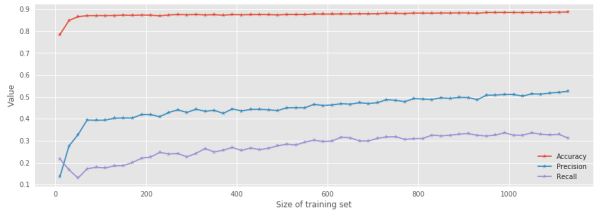

Fig. 22: AdaBoost Learning with different training data set for Bank

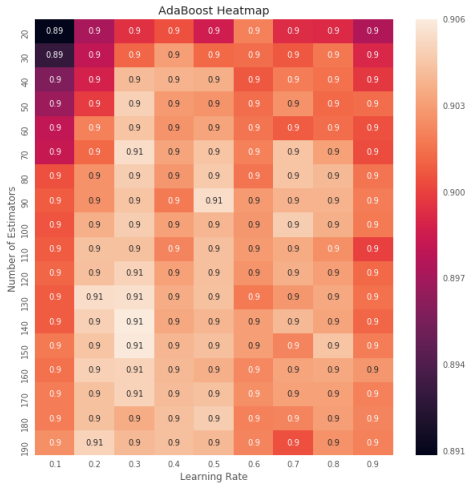

Fig. 23: Estimator and Learning rate Heat Map for Bank

there are quite a lot parameters can be tuned. For example, number of hidden layers, number of the neuron for each hidden layer, different activation function, different solver functions, and different learning rate. For this section of this report, I will keep some of the parameters the same, 2 hidden layers and each of the layer will using the same number of nodes.

1) Neural Network for Mushroom Dataset: The performance of the Neural Network for the mushroom dataset is not that great compared with the previous two methods. The best testing score I can get is 94.89%. I think one of the reason is the neural network is very good for unstructured data for example voice, image. For this simple classification problem, it does not a suitable model.

The analysis is from the previous section. The first graph will be the number of samples used and testing accuracy. Figure 26 is the figure for different size of training set. The data is based on two 30 nodes hidden layer and use 'real'. The accuracy both of training and testing become periodic. Every 200 samples, the accuracy decreased a lot. This probability according to the max interaction settled to 200. From this picture, we find out the neural network model is sensitive to the parameter. After the neural network has been build, during the training the structure can not change any more.

Figure 27 is the 10 folder cross validation testing for learning rate. The learning rate is from 0.001 to 0,05. From this graph, we can easily discover the large number of learning rate decrease the testing accuracy. The solver in this case is 'adam'.

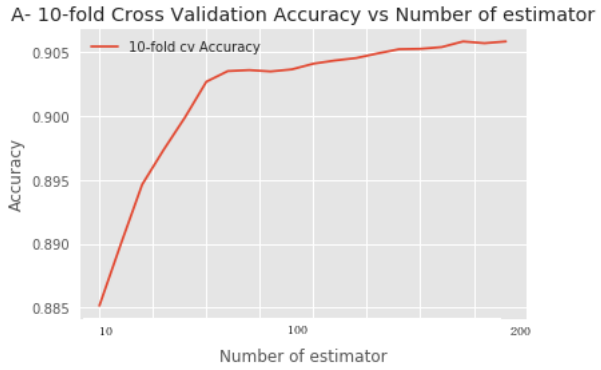

Fig. 24: Cross Validation Score for Number of Estimator for Bank

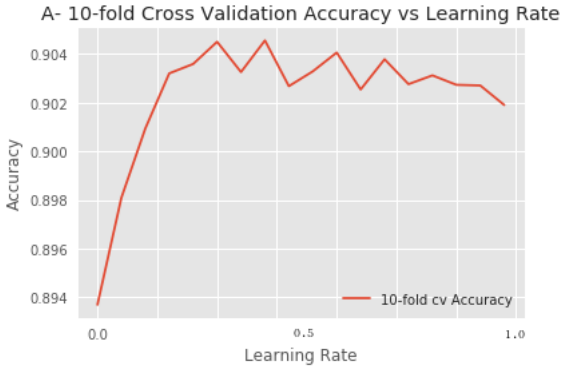

Fig. 25: Cross Validation Score for Learning Rate for Bank

For the model complexity analysis, I have select tow parameter for cross-validation analysis. On is the max iteration and an other is the size of two hidden layer. Figure 28 is the max iteration curve. The range of the iteration is from 20 to 300. From this curve, the accuracy after 120 stop changing. I think the reason is because the model converge after 120. Even tough, the max iteration has been set to large number, the accuracy won't change.

Figure 29 is the size of each hidden layer and cross-validation accuracy. From this curve, the accuracy increase in the begin, which means 2*2 is not complex enough to handle this problem. After 10 by 10, the model is good enough for this classification problem because the accuracy stop increase after size of 10.

2) Neural Network for Bank Dataset: For the bank data set, the neural network did not preform very well as the previous problem. The best training score I got for this problem is 89.10%. Let discover some of the performance of this model. Figure 30 is compare with different training size and testing/ training accuracy. This is quite different from the previous result. The recall accuracy drop to 0, this means the algorithm just predict 'No' all the time.

Figure 31 is the relation between the learning rate and the cross-validation accuracy. From this curve, the model accuracy stop around 88.80%. This is the same accuracy you can get by only predict 'No'. In this case, the Neural Network in this

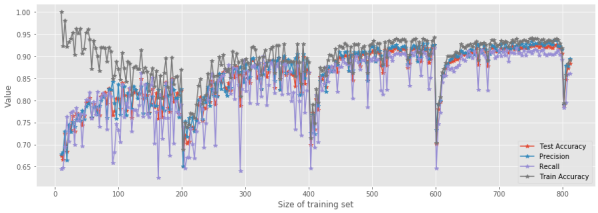

Fig. 26: Neural Network with Different Size Training Set for Mushroom

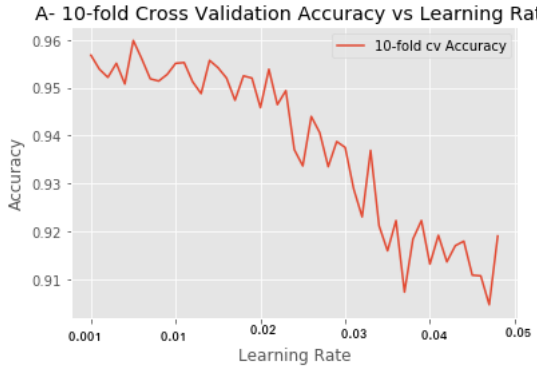

Fig. 27: Cross Validation for Different Learning Rate for Mushroom

particular problem is same will just saying 'No' to every one. This probability because, the neural network is not good to handle the unbalanced datasets.

Figure 32 is the hidden layer size and accuracy, from this curve, we got a similar result with other assumption. The size of hidden is not affecting the accuracy too much, and the model still not working very well.

Analysis summary: In this section, neural network method has been used for this two classification problem. However, no matter what parameter I have change neural network refuse to perform well. In conclusion, neural network probability is not good for this kind of classification problem.

D. Support Vector Machine(SVM)

Support vector machines (SVMs) are a set of supervised learning methods used for classification, regression and outliers detection. The advantages of support vector machines are Effective in high dimensional spaces, versatile: different Kernel functions.

The SVM is also a very complex machine learning model. However, compare with the deep neural network, still, have much less learning parameter. For example, different kernel, degree, max_iter. In this assignment, I will use the mushroom and bank data to test the performance some of these tuning parameters. Let us start with the mushroom first.

1) SVM for Mushroom Dataset: One of the things I need to mention when I use the SVM is slow. The algorithm performs very slow. Both training and testing. The best testing accuracy I can get from this machine learning method is 96.18%. Two kernel functions have been used for testing the learning curve, which 'rbf' and 'ploy' with degree 3.

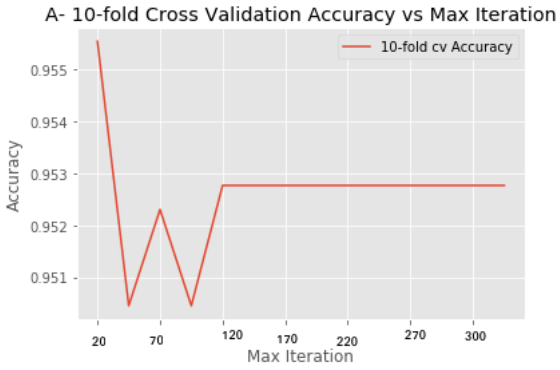

Fig. 28: Cross Validation for Different Learning Rate for Mushroom

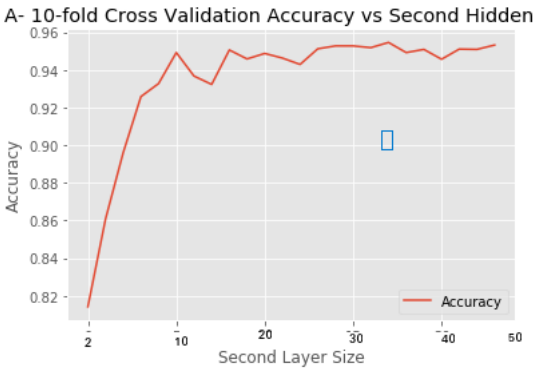

Fig. 29: Cross Validation for Different Learning Rate for Mushroom

Figure 33 is the learning curve with different input size data for 'rbf' kernel. Figure 34 is the learning curve for polynomial kernel with degree 3. From this two graph, we can get some knowledge about these two kernels. The polynomial function with degree 3 is much simpler than rbf kernel. that is probably better for low dimension problem.

For the model complexity problem, the experiment I have designed is the order or polynomial order. Figure 35 is the different polynomial order and accuracy comparison. For this curve, the testing and training accuracy stop increasing after 3 order, which means the oder 3 is good enough for handle this problem.

2) SVM for Bank Dataset: The SVM algorithm for this problem is very slow. I have to reduce the training data set in order to plot some of the graph. The best testing score I can get from SVM is 88.4. The algorithm is quite slow especially for this large dataset. I can not massively run the model to generate such a learning graph using this Banks data. For analysis, I believe the performance of this model is the same with the mushroom dataset.

Analysis summary: the SVM is a very computational expensive algorithm. In order to plot some of the graph, I have to reduce the training data size.

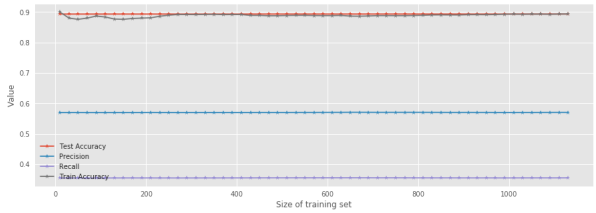

Fig. 30: Neural Network with Different Size Training Set for Bank Data

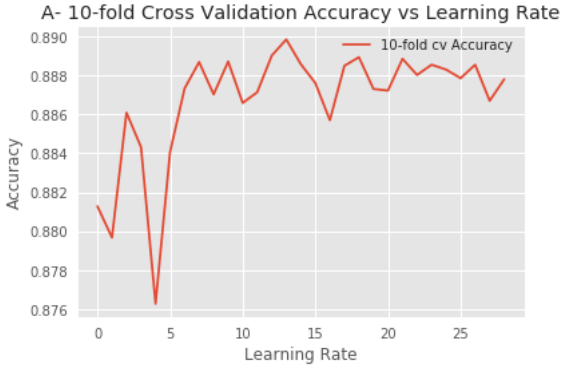

Fig. 31: Cross Validation for Different Learning Rate for Bank Data

E. K Nearest Neighbors(KNN)

Nearest neighbors provide functionality for unsupervised and supervised neighbors-based learning methods. The principle behind nearest neighbor methods is to find a predefined number of training samples closest in distance to the new point, and predict the label from this [7].

The KNN classification is a type of instance-based learning: it does not attempt to construct a general internal model, but simply stores instances of the training data. Classification is computed from a simple majority vote from K nearest neighbors. The algorithm run very fast, as fast as decision tree method. One interesting thing about this method is the running time between the training and predicting. Because this is a slow learner, the predicting take some additional work.

Compare with other machine learning method, KNN is quite simple. It has only a few parameters, for example, number of neighbors, the weight of each sample. The KNN algorithm has been applied to the two datasets. Let us start with the mushroom data.

1) KNN for Mushroom Dataset: The result of the KNN algorithm for this data set is very good, the best score I can get is 96.06% for the testing dataset. Let us check out the relation between training sample size and training/testing data.

Figure 36 is the learning curve. From this curves, it is easy to find out, the training curve is very high and the testing is very low at first.This is normal for most of the machine learning algorithm. However, for KNN the reason is little bit difference Because, for the small number of sample points, the algorithm to believe the data, and there are only a few of them. So at the beginning, the training accuracy decrease,

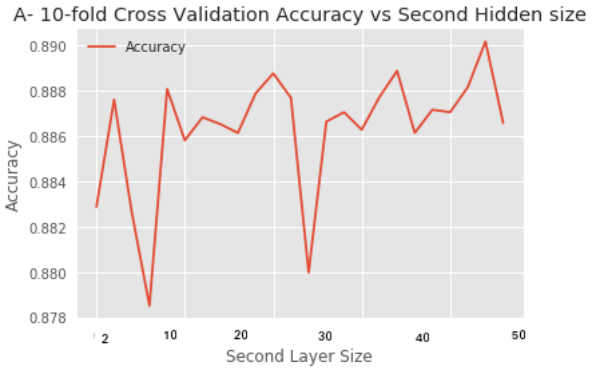

Fig. 32: Cross Validation for Different Learning Rate for Bank Data

Fig. 33: SVM rbf Kernel Learning Curve vs Number of Samples for Mushroom

and testing accuracy increase dramatically. After 500 sample point, the accuracy increase becomes very slow means, this model becomes mature. More samples will not contribute to the model too much.

For the KNN mode, I have only designed one test case to analyze the model complexity. In terms of the model complexity for KNN is the number of neighbors. So in this part, the cross-validation score for different K score shown in figure 37. In this figure, there are two curves, one for uniform sample weight, another weight is based on the distance. From these tow performance curves, we can assume the uniform weight over-fit when K increase over 10 neighbors. When three are lots of neighbors, even some of the neighbors from long distance, we can assume the irrelevance sample point involved for the final result .if every one contributes the same weight the model easy to over-fit. However, for the weight based on the distance, it makes sense when the point for away involved in the final result.

2) KNN for Bank Dataset: This section will discuss the KNN for Bank Dataset. The kNN give us an ok result, the testing accuracy I got is 88.18%. It is common for most of the algorithm. The learning curve is in figure 38, the training curve is always higher than the testing accuracy. The recall accuracy keeps running low, which means the prediction was not great on the "Yes" class. We got a similar result compared with a neural network. However, for the neural network, the training and testing accuracy stays the same.

Figure 39 is the cross-validation accuracy and K. Compare with the mushroom data, the model accuracy did not decrease. One of the reason is that this is an unbalanced dataset. If we

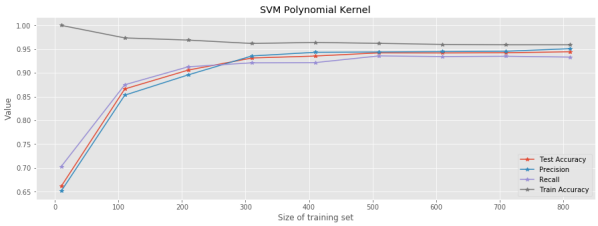

Fig. 34: SVM Polynomial Kernel Learning Curve vs Number of Samples for Mushroom

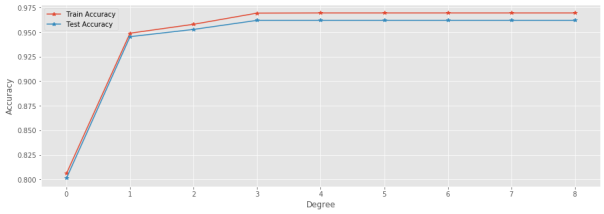

Fig. 35: SVM Polynomial Order and Accuracy for Mushroom

increase the K to the number of training samples, the accuracy would become the majority class all the time. Because the majority of the sample is one class. This probability happed to this dataset. The accuracy stay the portion of "No" class for the training sets.

Analysis summary: the algorithm run very quick, and the result pretty good also. For kNN, the model is not complex, and there are no optimization method involved during the training and testing. Some advanced data structures have been involved, for example KD tree to reduce the complexity.

IV. Result Analysis and Comparison

In this section, I will compare with different machine learning algorithm. There are two comparison, one is testing accuracy, another is the training time. The testing and training data set is the same for comparison. 80% for training and 20% for testing.

1) Testing Dataset Accuracy: Table III is the accuracy comparison for these five different algorithm. For some computational expensive method I did not use the GridSearchCV method to find the best parameter. There are some difference between different algorithm. For mushroom data set, three algorithms give the same answer. For the bank dataset, the boosting method got the best testing accuracy.
Boosting is an improved version of decision tree. So the result better than decision tree is quire normal. However, the computation is mush expensive than decision tree, which talk in detail in next section.

TABLE III: Testing Dataset Accuracy

Data Set	DT	Boost	Neural Net	SVM	KNN
Mushroom	96.18%	96.18%	94.40%	96.18%	96.06%
Bank	90.00%	90.47%	88.85%	88.45%	88.76%

2) Training and Testing Time: In this section, I will talk about the training and testing time for those five different algorithms. Table IV is the training and testing time for five

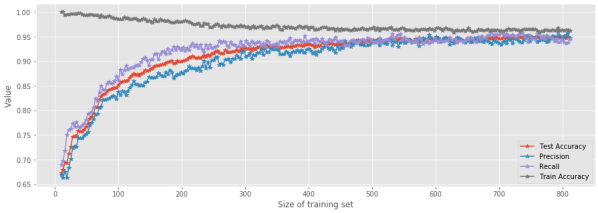

Fig. 36: KNN Learning Curve vs Number of Samples for Mushroom

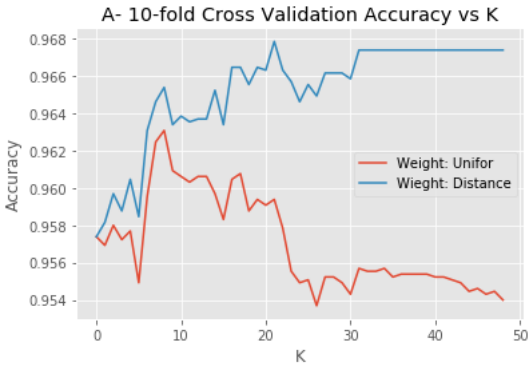

Fig. 37: Cross Validation for Different K for Mushroom

different machine learning algorithm.

In terms of training time, a decision tree is the best, and SVM is the worst algorithm. Decision tree and KNN did not involve complex optimization algorithm, so that is one of the reasons it performs very well.

In terms of the testing time. the decision tree also the best algorithm, KNN become the worst one. I just mentioned in the KNN section. KNN is a slow learner, which means the KNN put the computing during the predicting period. KNN is also the only algorithm which the testing time is more than the training time.

TABLE IV: Mushroom Data Testing and Training Time (second)

Mushroom	DT	Boost	Neural Net	SVM	KNN
Training	0.0052	0.5027	3.0978	7.3608	0.0115
Testing	0.0008	0.0275	0.0017	0.0681	0.1192

V. Conclusion

In this report, I have discussed five supervised learning method and results from two datasets in detail. Some of the algorithms are very simple, for example, decision tree and kNN, some of the algorithms involved lots of optimization and calculation, for example, neural network. They all have their characteristic also. Without knowing the fundamental knowledge, you can not understand the result of a particular algorithm, or how to improve it.

This is the end of this report, I would like to thank for this open end assignment. Typically you can spend the infinite time

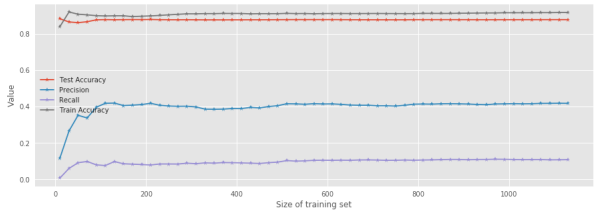

Fig. 38: KNN Learning Curve vs Number of Samples for Bank

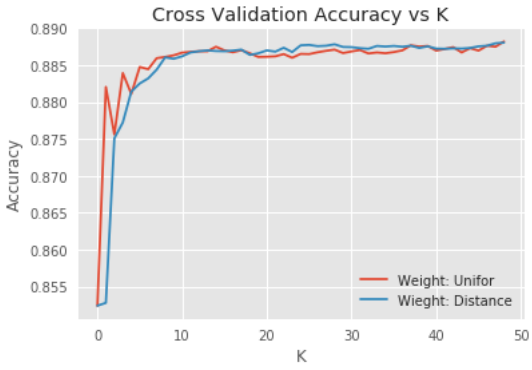

Fig. 39: Cross Validation for Different K for Bank

to work on this. For the time constraint, this would be all I have. I am glad I did this.

References

[1] M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of machine learning*. MIT press, 2012.

[2] "Mushroom data set," https://archive.ics.uci.edu/ml/datasets/Mushroom, accessed: 2018-01-30.

[3] "Bank marketing data set," https://archive.ics.uci.edu/ml/datasets/Bank+Marketing, accessed: 2018-01-30.

[4] "Mushrooms and decision trees," http://drorata.github.io/posts/2017/Mar/31/mushrooms-and-decision-trees/index.html, accessed: 2018-01-30.

[5] "adaboost," http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.AdaBoostClassifier.html, accessed: 2018-01-30.

[6] "Neural network," https://www.solver.com/xlminer/help/neural-networks-classification-intro, accessed: 2018-01-30.

[7] "Nearest neighbors," http://scikit-learn.org/stable/modules/neighbors.html, accessed: 2018-01-30.