A Comprehensive Comparison of Supervised Learning Algorithms

Jin Gan

j6gan@ucsd.edu

University of California, San Diego

Abstract

A large variety of classifier algorithms have been newly introduced in the past decade, and it's necessary to have an updated, comprehensive comparison between the performances of the commonly-used classifiers so that more efficient selection of model could be achieved when dealing with large dataset. In this paper, a comparison between SVM, KNN, XGBoost, Random Forest, Logistic Regression and Decision Tree is made on three different datasets and three different partitions of data instances for a total of three trials. The models are then ranked in order of average test accuracies.

1.Introduction

Selecting the best model for a dataset is a computationally-expensive and time-consuming task. Without any prior knowledge of which classifier could do better, trying out a big number of classifiers may significantly hinder the project progress. It's thus necessary to compare the performances of some commonly-used classifiers so that we could significantly shrink the candidate pool and pick one from the models known to have decent performance. In this paper, performance would be represented by the classifier's best training, validation and test accuracies. The final evaluation metrics used is test accuracy score.

2. Method

2.1 Learning Algorithms

Six common classifiers are selected for evaluation. Before actually training the models on datasets, hyperparameters need to be tuned for optimal performance. This section specifies the hyperparameters tested. Due to the limit of computation power, only the most important hyperparameters are explored. Other less important parameters are left optional, which may be tuned in the future with higher computation power.

XGBoost: XGB is a popular ensemble model newly-introduced in the recent years and it almost always produces decent results. Here I vary max depth with a candidate list of {1,3,5,7,10} and gamma chosen from {0.01,0.1,0.5,1,2}. More hyperparameters like regularization parameters, colsample_bytree, learning_rate, max_features, min_child_weight and n_estimators could be tested if computation allows.

Logistic Regression: logistic regression is one of the most basic algorithms used. Here I vary regularization parameter C by {0.01,0.1,1,10,100,1000}.

Decision Tree: decision tree displays potential outcomes under each condition in a tree-like structure. The hyperparameter tuned here is max_depth with a list of {None,4,8,12}. More important hyperparameters like criterion, splitter, max_depth, min_samples_split, min_samples_leaf and max_features could be tested if computation allows.

SVM: SVM uses support vectors to determine its decision boundary. The parameter tuned here is the regularization parameter C from list {0.1,1,10,100,1000}. Kernel may also be tuned with more computation power.

KNN: K-nearest-neighbors classifies a data instance based on its closest neighbors in terms of Euclidian Distance. Here I vary n_neighbors hyperparameter from {1,5,10}. Other parameters to be tuned include weights and algorithm.

Random Forest: random forest is a popular and well-performing ensemble method based on decision tree by default. The hyperparameter tuned here is max_depth by {10,50,100}. More relevant parameters like n_estimators, max_features, min_samples_split and min_samples_leaf may be explored in the future.

2.2 Datasets

I compare the algorithms on four datasets from UCI repository: Adult, Winequality-red, Pendigits and Avila. Most of Adult dataset's features are categorical, thus I used one-hot encoding to transform them into additional 0 or 1 features and expanded feature number to 98. For Winequality, I converted quality 3 to 5 into 0 and the rest into 1. For pendigits, I converted label 0 to 4 into 0 and the rest into 1. As for avila, I converted labels A to F into 0 and the rest into 1. All classifications here are binary since all of them have two kinds of labels after conversion. Other than winequality dataset having only 1599 instances itself, all other datasets have 5000 instances randomly selected for use in consideration of the balance between limited computation power and good data size. Here's a table for summary:

Table 1: Description of dataset

Dataset	Attribute number	Dataset size	%POZ
Adult	14/98	5000	24.4%
Avila	10	5000	25.4%
Pendigits	16	5000	48.7%
Winequality	11	1599	53.5%

3.Experiment

First, import functions from libraries and load in the datasets. A total of three datasets is used to eliminate randomness, and functions like describe, groupby, boxplot, histogram and correlation matrix are used when fit to give a general idea of the data distribution. Below is an example correlation matrix of avila dataset:

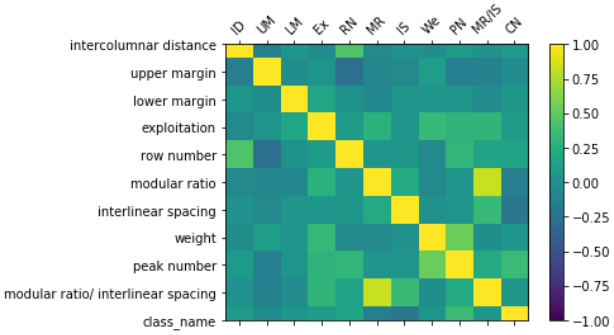


Figure 1: Correlation matrix illustrating the correlations between each feature. It gives a better understanding of the dataset.

Preprocess the datasets so that they have numeric features and binary labels. Then, split the data into training set and test set. Here three different kinds of partition are implemented with test size equal to 20%, 50% and 80% respectively, thereby resulting in three splits--- 80/20, 50/50 and 20/80. Each partition would have three trials to eliminate randomness by letting all points have a chance to be present in both training and testing set.

Next, prepare a list of classifiers and their corresponding hyperparameters that need to be tuned. Use Grid Search method to perform 5-fold cross-validation on the training dataset and obtain the best score and best hyperparameters for each model. Run this through three partitions and three datasets, and run three trials for each to eliminate randomness. Here the evaluation metrics used is accuracy score. The best score represents the validation accuracy for each classifier under the optimal hyperparameter, thus we could store it into a list for later report. Also, heatmaps could be drawn using the mean test scores within cv results, visualizing which hyperparameters correspond to which validation accuracies and how they change together.

Below are some example heatmaps when selecting the best hyperparameter for each model. Due to simplicity concern, only one set in one trial for one partition generated from avila dataset is shown. They indicate how each hyperparameter corresponds to the model's performance in terms of validation accuracy, and the hyperparameter number with the highest score is selected:

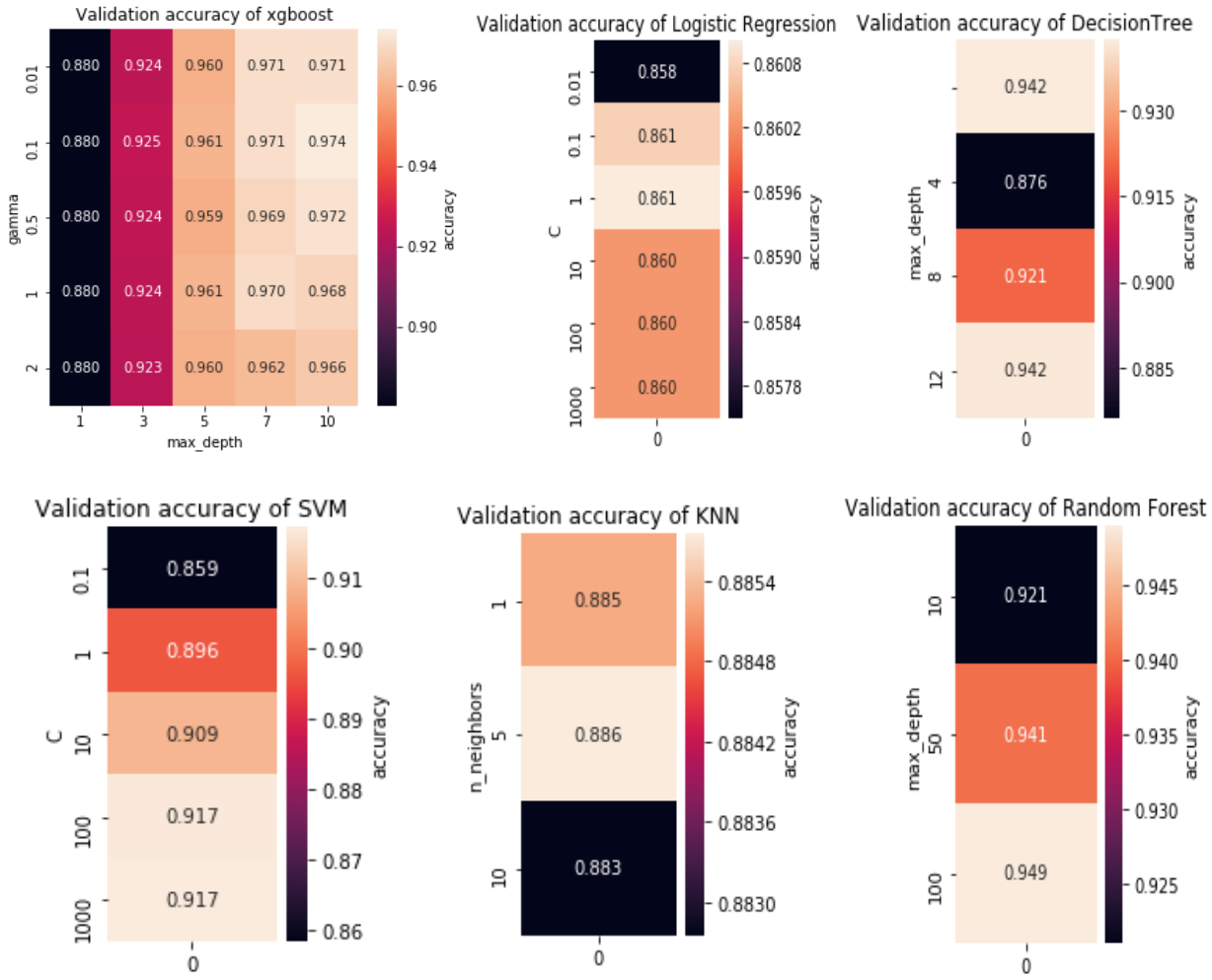


Figure 2: Example heatmaps of validation accuracy scores listed with each hyperparameter value.

Now we have the best hyperparameters figured out. Use them in the models and fit the models on training set to get the trained models. Apply trained models on both training set and test set to get training accuracy and test accuracy. After all three trials are done for each partition, report these accuracies along with validation accuracies stored, ranking them in order of test accuracies averaged across all trials and all datasets. This is the performance rank we need.

4.Conclusion

Table 2: Comparing different classifiers on four datasets with partition of 80/20 (20% data as test set). Accuracy averaged over three repeated trials. ± indicates standard deviation. For simplicity, standard deviation for validation accuracy is not included below because a lot of them are too small.

Classifiers	Dataset A Adult (80/20)			Dataset B Avila (80/20)			Dataset C Winequality (80/20)			Dataset D Pendigits (80/20)			Average
	Training accuracy	Validation Accuracy	Test Accuracy	Training accuracy	Validation Accuracy	Test Accuracy	Training accuracy	Validation Accuracy	Test Accuracy	Training accuracy	Validation Accuracy	Test Accuracy	Test Accuracy
XGBoost	90.3 ± 0.3%	85.8%	86.5 ±1.1 %	100 ±0%	97.4%	97.8 ± 0.2%	99.3 ± 0.8%	80.3%	81.6 ± 1%	100 ± 0%	99.2%	98.8 ± 0.3%	91.2 %
Random Forest	87.4 ± 0.4%	84.0%	85.0 ± 0.6%	99.8 ± 0.1%	94.9%	95.8 ± 0.4%	96.7 ± 1.9%	78.1%	79.2 ± 1.2%	100 ± 0%	99.3%	98.8 ± 0.2%	89.7 %
SVM	86.5 ± 0.3%	83.4%	83.8 ± 1.5%	96.1 ± 0.8%	91.7%	92.4 ± 0.5%	83.5 ± 2.8%	76.1%	76.7 ± 1.5%	99.9 ± 0.1%	99.7%	99.6 ± 0.1%	88.1 %
Decision Tree	86.0 ±1%	83.7%	84.2 ± 1.3%	100 ± 0%	94.2%	95.0 ± 0.9%	98.3 ± 0.1%	73.7%	74.6 ± 1.6%	100 ±0%	97.3%	98.0 ± 0.4%	88.0 %
KNN	84.4 ± 0.3%	80.4%	80.8 ± 1%	93.4 ± 0.2%	88.6%	90.6 ± 1.6%	100 ± 0%	73.9%	75.9 ± 1.4%	99.9 ± 0.1%	100%	99.7 ± 0.1%	86.8 %
Logistic Regression	85.2 ± 0.3%	84.2%	84.5 ± 1.4%	86.3 ± 0.2%	86.1%	86.8 ± 0.8%	74.4 ± 0.5%	74.0%	76.4 ± 1.3%	85.2 ± 0.2%	84.9%	85.5 ± 0.8%	83.3 %

Table 3: With partition of 50/50 (50% data as test set).

Classifiers	Dataset A Adult (50/50)			Dataset B Avila (50/50)			Dataset C Winequality (50/50)			Dataset D Pendigits (50/50)			Average
	Training accuracy	Validation Accuracy	Test Accuracy	Training accuracy	Validation Accuracy	Test Accuracy	Training accuracy	Validation Accuracy	Test Accuracy	Training accuracy	Validation Accuracy	Test Accuracy	Test Accuracy
XGBoost	88.4 ± 0.5%	85.5%	85.4 ± 0.6%	100 ± 0%	95.8%	96.2 ±0.2 %	98.6 ± 1%	77.7%	77.7 ±1.4 %	100 ± 0%	98.6%	98.8 ± 0.1%	89.5 %

Random Forest	88.0 ± 0.3%	83.8%	84.0 ± 1%	99.6 ±0.2 %	93.2%	93.5 ±0.7 %	97.9 ±0.4 %	76.6%	75.8 ±1.6 %	100 ± 0%	98.3%	98.6± 0.3%	88.0 %
SVM	87.0 ± 0.4%	82.3%	83.1 ± 0.3%	94.8 ± 1.8%	91.1%	90.3 ±0.1 %	85.2 ±2.7 %	77.2%	75.7 ±1.5 %	99.9± 0.1%	99.6%	99.5± 0.1%	87.2 %
Decision Tree	84.5 ± 0.5%	84.3%	83.7 ± 0.3%	99.7 ± 0.4%	92.4%	93.1 ±0.9 %	84.4 ± 9.4%	70.8%	73.4 ±0.8 %	100±0 %	96.6%	96.7± 0.4%	86.7 %
KNN	84.3 ± 0.5%	80.1%	80.8 ± 0.3%	97.5 ±3.4 %	88.3%	88.4 ±1.2 %	79.1 ±2.6 %	73.0%	72.2 ±1.6 %	99.9± 0.2%	99.5%	99.6± 0%	85.3 %
Logistic Regression	85.6 ± 0.2%	83.6%	83.9 ± 0.4%	86.0 ± 0.1%	85.6%	86.4 ±0.3 ±	75.3 ±1.5 %	75.3%	73.5 ±1.3 %	85.8± 0.3%	85.3%	84.9± 0.3%	82.2 %

Table 4: With partition of 20/80 (80% of data as test set)

Classifiers	Dataset A Adult (20/80)			Dataset B Avila (80/20)			Dataset C Winequality (80/20)			Dataset D Pendigits (80/20)			Average
	Training accuracy	Validation Accuracy	Test Accuracy	Training accuracy	Validation Accuracy	Test Accuracy	Training accuracy	Validation Accuracy	Test Accuracy	Training accuracy	Validation Accuracy	Test Accuracy	Test Accuracy
XGBoost	89.9 ±0.4%	84.0 %	84.5± 0.2%	100±0 %	93.4 %	92.8± 0.8%	94.6± 1.9%	75.5 %	73.8± 1.0%	100±0 %	97.6 %	97.8± 0%	87.2 %
SVM	87.3± 0.5%	82.4 %	82.1± 0.3%	94.1± 0.5%	89.4 %	89.4± 0.6%	82.5± 2.1%	74.6 %	72.9± 0.7%	99.8± 0.2%	98.7 %	99.3± 0.2%	85.9 %
Random Forest	93.8± 3.5%	83.0 %	83.0± 0.7%	99.0± 0.6%	91.4 %	91.0± 0.8%	98.5± 0.5%	75.2 %	71.2± 1.5%	100±0 %	97.5 %	97.7± 0.6%	85.7 %
KNN	84.7± 1.6%	79.5 %	80.2± 0.9%	91.2± 1.4%	88.3 %	87.0± 0.5%	75.9± 1.7%	70.8 %	71.4± 0.6%	100±0 %	99.0 %	99.4± 0.1%	84.5 %
Decision Tree	84.9± 0.5%	81.9 %	82.7± 0.5%	97.2± 2%	90.2 %	89.8± 0.8%	88.0± 8.5%	65.5 %	66.8% ±2.2	100±0 %	94.2 %	95.1± 0.5%	83.6 %
Logistic Regression	86.4 ±0.4%	83.8 %	83.0± 0.5%	86.9± 0.9%	86.7 %	86.1± 0.5%	76.9± 0.9%	74.0 %	73.9± 0.2%	85.4± 1%	85.5 %	85.0± 0.3%	82%

As you can see in the table, the general ranking of the classifiers in terms of average test accuracy is XGBoost, random forest, SVM, Decision Tree, KNN and Logistic Regression. XGBoost and Random Forest perform relatively well on all datasets, even though there are times that they aren't the best for certain datasets. Logistic Regression typically performs the worst. KNN, SVM and Decision Tree are in the middle with SVM typically being better than the other two. Although there's distinction on average performances, there is still variability across different datasets and models that do poorly in general could do better than models with good general performance in certain cases.

From different partitions, we could also see that the test accuracy decreases as training set becomes smaller. This is also evident by visualizing test accuracies across datasets:

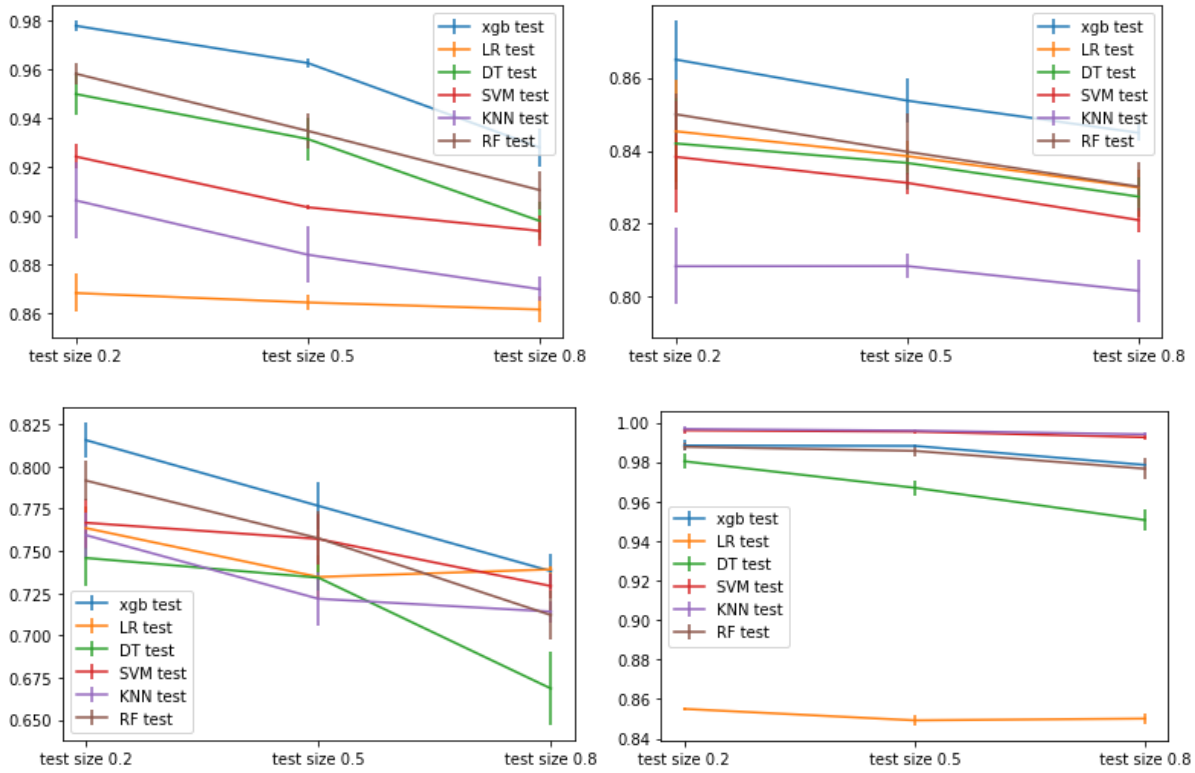


Figure 3: Test accuracies in various partitions for Avila dataset, Adult dataset, Winequality dataset and Pendigits dataset respectively.

As shown above, the general trend shared by all models is the decrease of performance with less training data. This is understandable because a model needs large amount of data to learn features better.

5.References

Caruana, R., Niculescu-Mizil, A. (2006). An Empirical Comparison of Supervised Learning Algorithms. ICML 2006.

C. DeÂ Stefano, M. Maniaci, F. Fontanella, A. ScottoÂ diÂ Freca, Reliable writer identification in medieval manuscripts through page layout features: The 'Avila' Bible case, Engineering Applications of Artificial Intelligence, Volume 72, 2018, pp. 99-110.

Kohavi R., Becker B., (1996). Adult Data Set [https://archive.ics.uci.edu/ml/datasets/Adult]. UCI Machine Learning Repository. Irvine, CA: University of California, School of Information and Computer Science.

P. Cortez, A. Cerdeira, F. Almeida, T. Matos and J. Reis. Modeling wine preferences by data mining from physicochemical properties. In Decision Support Systems, Elsevier, 47(4):547-553, 2009.

E. Alpaydin, Fevzi. Alimoglu. Pen-Based Recognition of Handwritten Digits Data Set [https://archive.ics.uci.edu/ml/datasets/Pen-Based+Recognition+of+Handwritten+Digits]. UCI Machine Learning Repository. Irvine, CA: University of California, School of Information and Computer Science.

Chen T., Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16). Association for Computing Machinery, New York, NY, USA, 785–794. DOI:https://doi.org/10.1145/2939672.2939785

6. Extra Credit

1. I did my report on 6 classifiers and 4 datasets, which satisfies the extra credit requirement.
2. I reported standard deviation along with accuracies, which requires a lot of extra work.
3. Three out of my datasets have 5000 instances. All of them have beyond 10 features---one even has 98. This is well above the requirement, and a lot of time was spent for training models.
4. I also tested eliminating randomness by doing three trials of cross validation but the same split of dataset. For simplicity, only the 80/20 partition is reported here.

Table 5: Accuracies with three trials on cross validation instead of splitting data.

Classifiers	Dataset A Adult (80/20)			Dataset B Avila (80/20)			Dataset C Winequality (80/20)			Dataset D Pendigits (80/20)			Average
	Training accuracy	Validation Accuracy	Test Accuracy	Training accuracy	Validation Accuracy	Test Accuracy	Training accuracy	Validation Accuracy	Test Accuracy	Training accuracy	Validation Accuracy	Test Accuracy	Test Accuracy
XGBoost	0.897	0.862	0.850	1	0.973	0.983	0.959	0.796	0.794	1	0.992	0.992	0.905
Logistic Regression	0.854	0.846	0.837	0.866	0.866	0.851	0.747	0.737	0.747	0.852	0.853	0.862	0.824
Decision Tree	0.841	0.841	0.831	1	0.937	0.948	0.894	0.731	0.750	1	0.978	0.977	0.877
SVM	0.868	0.838	0.835	0.956	0.921	0.917	0.854	0.765	0.728	0.999	0.996	0.997	0.869
KNN	0.842	0.810	0.809	0.933	0.899	0.883	0.818	0.757	0.713	1	0.996	0.996	0.850
Random Forest	0.877	0.845	0.839	0.997	0.951	0.946	0.989	0.780	0.784	0.999	0.991	0.987	0.889

Comparing the average test accuracy here with the table of three trials on splits, we can see that only repeating cross validation without re-splitting data isn't sufficient enough to reduce randomness, for its test accuracies are generally lower than the accuracies obtained from re-splitting data.

Also, either the fluctuation of test accuracies becomes bigger or the general trend of larger training size having better test accuracy becomes unclear in this case:

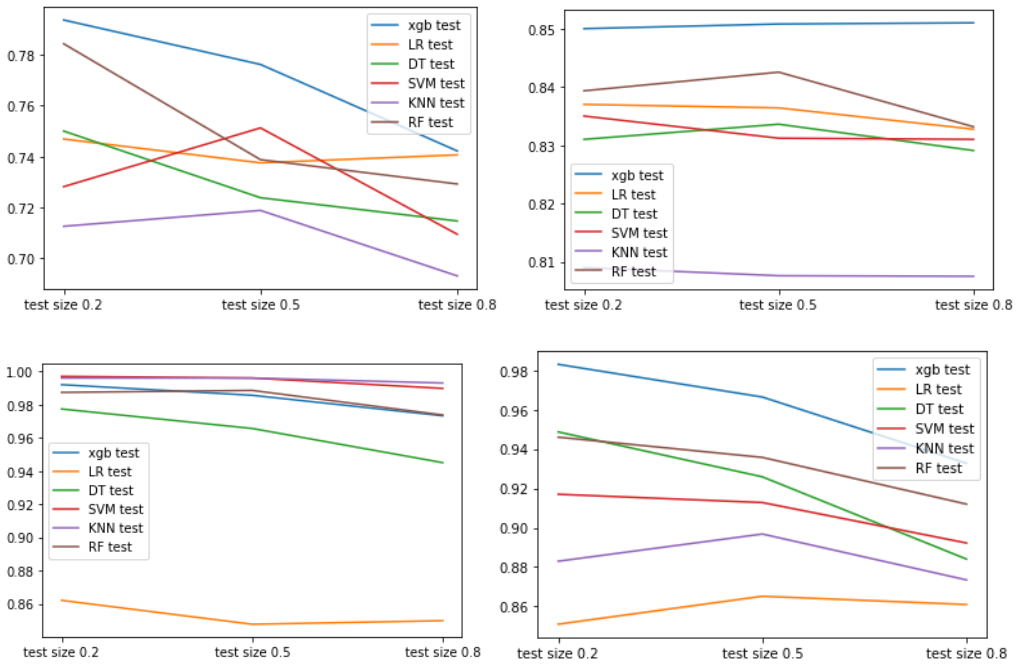
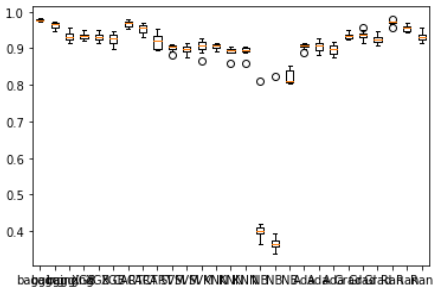
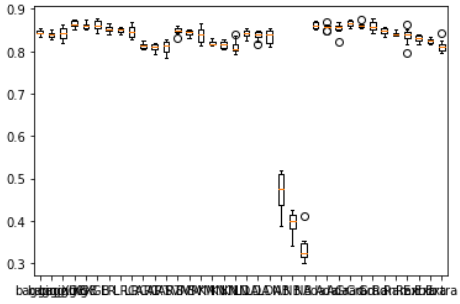


Figure 4: Test accuracies for winequality, Adult, pendigits and avila respectively. The trend of more training data having better test accuracies is less clear compared to re-splitting data for three trials, and more fluctuations are present.

5. I even tried to find the best model for two of the datasets.



Avila



Adult

Figure 5: Test accuracies of various models visualized.

Although the labels aren't clearly shown here, the best model for Avila dataset is bagging classifier with with accuracy of 0.977709, and the best model for Adult dataset is gradient descent boosting with average accuracy of 0.866. Both are in partition 80/20.