Abstract

In this research we re-evaluate and reproduce the results that Caruana and Niculescu-Mizil produced in their 2006 paper about the general effectiveness and efficiency of modern supervised learning methods algorithms by testing them against different sample sets using different parameters and proportions of data splits. The several supervised learning methods we re-evaluate are: Support Vector Machine (SVM), K-Nearest Neighbors (KNN), logistic regression, random forests and gradient boosting.

By trying to reproduce their results, we gain an idea of not only which algorithm is more effective in general but also how and where specific algorithms outperform the others for certain problems.

1 Introduction

During the invention and usage of machine learning algorithms, we saw the peak and drop in interests of specific types of algorithms based on their usage in a particular decade which changes due to inventions of other ns.types of better algorithms (eg Deep learning). Even so, no one machine learning algorithm would be the best if we can aggregate and bag the classifiers in an most optimal way. A good example in many industries is to use semi-supervised learning, where unsupervised learning is used to split huge data set concerning populations would labels and then by using supervised learning to identify the patterns and create labels for the patterns. Experimentations and scientific quantification allows us to find optimal(but not necessary the best way) to find solutions to our problems. These algorithms may or may not be obsolete in the near future due to many other algorithms invented perhaps on top of these pioneers but understanding them would allow us to find simple solutions to complex problems.

Similar to the research created by Caruana and Niculescu-Mizil, we found out that there were several algorithms that were superior on most problems on average – they scaled better than others and/or performed better (or worse) in high-dimensional spaces. Likewise, some algorithms may also be enough to fulfill the needs of a simple and clean data set with few attributes and we would not need to spend as much time on algorithms that are more computationally intensive.

2 Methodology

2.1 implementation of algorithms

We find the most optimal version of the algorithms and hence the best algorithm overall by averaging the results of the implementation of each algorithms three times by using 20:80, 40:60, 50:50 ratios to train and test data respectively. Besides doing that, we ensure results had a lower discrepancy throughout data set by scrambling them and then doing a 5-fold cross-validation on the data sets to be trained to ensure that results were not biased to a specific portion of the data set.

We used hyperparameters based on the amount of attributes we have or implemented algorithms like random forests directly on datasets that do not require one-hot encoding.

Below are additional information on the ways and reasons on why we implemented the algorithms the way we did:

1) Logistic Regression: We used a LIBLINEAR implementation for binary classification tasks and use

only L1 penalty term while we use CG-NEWTON for multi-label classification which only has L2 penalty and was thus used.

2) Support Vector Machine (SVM): We used an SVM with a polynomial kernel, whose powers vary most likely on how many attributes are contained in the dataset. We tune the C based on estimation on what values the algorithm was pushing towards in the direction and re-test the code.

2.2 Preprocessing of Data sets

Due to time constraints, we only managed to do 2 data sets which presents different problems to us. For the Skins data set, the data set only has 3 attributes so for certain algorithms like Random Forests, we could only allow max features to be at most 3; not much preprocessing is needed because the dataset consists mainly of numbers.

ADULT dataset is more of a challenge because we have not only numerical but also nominal data that we would have to use one-hot encoding so that none of the titles of the nominal has any order and can be evaluated as stand-alone categories. We also need to filter out data rows that has missing data values and also columns that has continuous data values that do not necessary help us in classifying the data with its labels; these data values are furthermore computationally expensive (e.g capital gains, hours per week, age).

The ADULT data set looks into the social-economic attributes of people and predicts whether people earned more or less than $50k annually. Important attributes of the data set includes: age, workclass, education-number, marital-status, occupation, relationship, race, sex, native-country and the labels >50K and <=50K; these attributes are then one hot encoded for all the classification algorithms except for random forests, which as decision trees deal well with the combination of qualitative and quantitative data.

The SKINS data set has attributes of B,G,R values and responses 1 or 2. The attributes are collected from a population that is being treated as homogeneous.

3 Results

From the results we got and averaging of the data before comparing them amongst themselves, we see that they perform pretty closely with Logistic Regression performing the worst while Gradient Boosting and Random Forest are the best performers. Support Vector Machine is an average performer as compared to the rest.

4 Discussion

The results of this paper should be taken lightly due to the computation power of my laptop and also of the usage of 5 classifiers on only 2 data sets. We also only think about accuracy and there might still be improvements in preprocessing the dataset by splitting attributes in ADULTS like age into subsets which might use a slightly statistical approach.

5 Conclusion

Given the encouraging results regardless of the classifier employ, we should decide that the features and processing of the features are more important at accurately recognizing and predicting patterns from data sets since there are many methods that can significantly affect the performance of any classifiers: experimental choices, design constraints, size of train/test split, CV scheme, hyperparameter choices and ranges. And it is seen that it is the simple classifiers that lack the tools to optimize that performs the worse but that does not contradict the fact that classifiers who lack hyper-parameter optimization will not perform as worse as the worse classifiers.