Supervised Learning Algorithms: Binary Classification Comparison

Howard Kim HOKIM@UCSD.EDU
Department of Cognitive Science
University of California, San Diego
La Jolla, CA 92093-5004, USA

Editor: Howard Kim

Abstract

This paper describes the replication of a large-scale empirical comparison between supervised machine learning algorithms of Caruana and Niculescu-Mizil (2006) in a smaller scale. For this particular comparison, four different supervised machine learning algorithms are compared and contrasted by their performance on binary classification tasks across four different data sets with five trials each. This process boils down to total of eighty different trials across the following algorithms: logistic regression, k-nearest neighbors, decision trees, and random forests. Each trial will be tuned by five-fold cross-validation on randomly chosen five-thousand data samples to select hyperparameters through a systematic grid searches. The empirical results are in parallel to the findings observed by Caruana and Niculescu-Mizil (2006) with performance measured through ACC, F1, and ROC AUC.

Keywords: binary classification, logistic regression, k-nearest neighbors, decision trees, random forests, hyperparameters, cross-validation

1. Introduction

The precedent of this paper is the widely-known empirical comparison of supervised learning algorithms done by Caruana and Niculescu-Mizil (2006), hereafter referred to as the CNM06. Much like this paper, the CNM06 followed its precedent, STATLOG, which offered a very comprehensive empirical studies comparing the machine learning algorithms done by King et al. (1995). The motive for CNM06 was to update the comprehensive empirical study that was long overdue with newly emerged supervised learning algorithms, such as bagging, boosting, and random forest. However for this paper, it serves the purpose to replicate and compare the results from the CNM06 in a smaller scale for validation in terms of supervised learning algorithms and their performance.

Much focused to the CNM06's motive to update the comprehensive empirical comparison, Caruana and Niculescu-Mizil addressed that different performance metrics are appropriate for each different algorithms, much like how there are different algorithms to be used in different domains. This led them to use many other performance metrics to measure different trade-offs; similarly for this study, three different performance metrics were used to gauge the performance of the different algorithms focused: ACC, F1, and ROC AUC. Also known

as 'Accuracy', 'F1 score', and 'Receiver Operating Characteristics Area Under Curve' respectively. In terms of the mathematics and calculations behind each of these performance metrics, they will be further discussed in 'Performance Metric' section.

While the CNM06 presents a large-scale comparison between ten different learning algorithms, this comparison will be on four supervised learning algorithms. In terms of the algorithm selection, the algorithms were chosen only from the list of algorithms used in the CNM06 to replicate and compare results. Because this comparison is much smaller in scale, it was ensured that the algorithms chosen had varying performances; such as, one algorithm from high performing range (Random Forest), one from mid performing range (K-Nearest Neighbors), one from low performing range (Logistic Regression), and finally one randomly chosen from any of the ranges (Decision Tree). Mostly similar to the CNM06, the comparison was conducted on similar number of problem data sets as the number of learning algorithms.

2. Methodology

2.1 Learning Algorithms

As previously mentioned, this smaller scale empirical comparison focuses on four different supervised learning algorithms: Random Forest, Logistic Regression, Decision Tree, and K-Nearest Neighbors. As seen in the CNM06, these four learning algorithms have varying performances, but they also use varying methods in creating the classification models: Random Forest uses the ensemble learning model, Logistic Regression uses the linear learning model, Decision Tree uses the predictive learning tree model, and K-Nearest Neighbors uses the instance-based learning model. The rest of this section discusses the parameters used with the Scikit-Learn libraries for each of the learning algorithms for replicability.

Random Forest (RF): To replicate the comparison environment as much as possible to the CNM06, similar settings and parameters were used. The Random Forest has a fixed amount of trees (n_estimators) at 1024 trees. The size of the feature (or attribute) set (max_features) had the following array of possible numbers considered at each split: 1, 2, 4, 6, 8, 12, 16, or 20.

Logistic Regression (LR): Also similar to the CNM06, both regularized and unregularized models (penalty) were used. When regularized models were used, both L_1 and L_2 norms were used; when unregularized model was used, the penalty was set to 'none'. For regularized models, the inverse of regularization strength (C) had varying strength with factors of 10 from 10^{-8} and 10^4. Maximum number of iterations taken for the model to converge (max_iter) was set to a fixed value of 5000.

Decision Tree (DT): Differing from the CNM06, the Decision Tree model was left mostly 'as-is' standard from the Scikit-Learn library. The only parameter that was changed was the maximum depth of the tree (max_depth) to use the array of depth from 1 to 5 in steps of one. The reasoning behind the low number in depth (shallowness of the tree) was because increasing the tree can make it more prone to over-fitting the data. The shallower tree was

preferred because it allows less complexity and more easier to generalize.

K-Nearest Neighbor (KNN): Also differing from the CNM06, the values of K selected for this comparison were different from the K values used in the CNM06. For the CNM06, 26 different values of K were used that ranged from 1 to $|trainset|$ (5000). In this particular case, the model degenerates to classifying the label to the class which is the most common in the dataset every time. This is especially important in the case for data sets that are highly imbalanced, which applies to the most of the data sets used in this comparison (refer to Table 1). Hence, 26 values of K were used that ranged for 1 to 105 (steps of 4).

2.2 Performance Metrics

The empirical comparison between four learning algorithms are based on three performance metrics as mentioned previously. This was to address the differently performing algorithms and using the performance metrics that are appropriate for their domain. With the importance of data, all trial performances were scored by 'Accuracy' (ACC), 'F_1 Score' (F1), and 'Receiver Operating Characteristics - Area Under Curve' or Area under ROC curve (ROC AUC). All three performance scoring metrics have score values ranging between [0, 1] that were calculated using the Sci-kit Learn's Classification 'metrics' library. However, the underlying calculation can be achieved by using the following equations:

$$\textbf{Accuracy (ACC)} = \frac{\sum Positive_{true} + \sum Negative_{true}}{\sum Population_{total}}$$

$$\textbf{F}_\textbf{1}\textbf{ Score (F1)} = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

where Precision and Recall can be calculated by

$$\textbf{Precision} = \frac{\sum Positive_{true}}{\sum Positive_{true} + \sum Positive_{false}} \quad \textbf{Recall} = \frac{\sum Positive_{true}}{\sum Positive_{true} + \sum Negative_{false}}$$

As previously mentioned for ROC AUC, we are finding the area under the ROC Curve, and ROC Curve is defined by TPR (True Positive Rate), also known as Recall, and FPR (False Positive Rate). While TPR (or Recall) can be found by the equation above, the FPR can be found by the following equation:

$$\textbf{False Positive Rate (FPR)} = \frac{\sum Positive_{false}}{\sum Positive_{false} + \sum Negative_{true}}$$

Then, the AUC (Area Under Curve) can be calculated by integrating the ROC Curve.

$$\textbf{ROC AUC} = \int ROC\ Curve = \int TPR(FPR^{-1}(x))dx$$

2.3 Data Sets

This empirical comparison of four different learning algorithms will be conducted on four different binary classification task based data sets. These data sets were retrieved from the University of California, Irvine Machine Learning Repository and can be referenced from links provided in the 'References' section. When selecting problem data sets for this comparison, it was ensured to have the following criterias: binary classification (label/class), sample size of at least 10000, features (or attributes) that are reasonable to One-Hot encode (i.e time data and name data would fare as unreasonable attributes for encoding), and scalability by standardization.

The following data sets has been used for this particular empirical comparison:

- **(ADULT):** Data set containing demographic information on adults, which are used to predict whether their income is less than or equal to 50K or over 50K. Also known as 'Census Income' dataset. This data set was encoded using One-Hot encoding for some of its categorical features, which changed the number of features (attributes) to 104 from 14.
- **(GRID):** Data set containing electricity power specification data (such as power produced and consumed) for electrical grids, which are used to predict whether the 4-node star electrical grid system is stable or unstable.
- **(HTRU2):** High Time Resolution Universe Survey (HTRU) data set containing samples of 'Pulsars' (a rare type of Neutron star that produce radio emission detectable on Earth) and its specification, which are used to predict whether the star sample is a read 'Pulsar'.
- **(OCCUPANCY):** Data set containing room conditions temperature, humidity, light, and carbon dioxide levels, which are used to predict the room's occupancy. This data set includes date and time data, which has been removed for this comparison due to its unsuitable characteristics for encoding/scaling.

Table 1: Problem Data Set Summary

PROBLEM	#ATTR	TRAIN SIZE	TEST SIZE	%POS
ADULT	14/104	5000	30162	24%
GRID	13	5000	10000	36%
HTRU2	8	5000	17898	9%
OCCUPANCY	5	5000	20560	23%

3. Experiment & Results

For each data set, five trials were performed. For each trial, a hyperparameter search to find the best settings (parameters) to use for the algorithm was done for each algorithm. Similar to the CNM06, 5000 samples were randomly chosen with replacement from the data set to be used as the training set for each trial. Using Stratified K-Fold, 5-fold cross-validation were done on the training set to select the hyperparameters via a systematic grid-search of the parameter space. For this comparison, the 'Stratified K-Fold' was used over typical 'K-Fold' because it returns stratified folds that is shuffled by 'random_state' ensuring the data sets to not overlap, which is preferred when dealing with classification task with imbalanced data sets (particularly this comparison data sets).

For the systematic grid-search of the parameter space, the classification algorithms were put into a pipleline, which allows me to assemble several steps that can be cross-validated together while setting different parameters. This was especially necessary for this particular case to condense the code as much as possible when dealing with arrays of different parameters to be used and searched inside grid-search. The pipeline allowed the grid-search process to be condensed for the following parameters: array of C-values for Logistic Regression, array of K-values to be used for K-Nearest Neighbors, array of max number of features for Random Forest, array of max number of depth for Decision Tree, and array of regularizations to be used for Logistic Regression. In addition to the parameters, the 'StandardScaler' was added to the pipeline, which allows us to scale the data by standardization.

After creating the pipeline and grid-search for each of the learning algorithms, it loops into each of the classifier grid-searches. For each classifier, it fits the model using the training data, and the best performing parameters for each performance metric (ACC, F1, and ROC AUC) is retrieved from the grid-search. With the best performing parameters tailored for each scoring metric retrieved, those parameters are set back into the pipeline to be fitted again with the training data for each set of parameters. Now with three different pipelines, the binary classification predictions are made using both the training data set and the testing data set. Using the Sci-kit Learn's metric library, the scores for our performance metrics are retrieved and saved into our data structure.

The scores were stored into hierarchical set of dictionaries and arrays; more details to composition of this data structure can be found in the Appendix section (code cell 7). The scores were then retrieved to be cleaned and organized by training data score, testing data score, trial, algorithm, and data set used. Score data cleaning and organization can also be found in the Appendix section.

3.1 Performances by Metric

Table 2 shows the normalized scores for each algorithm on each of the three performance metrics. This contains the mean testing set performance across trials for each algorithm and dataset combination, with the respect to each of the performance metric. Each entry in the table averages the scores from five trials and dataset for that particular performance metric. The last column, MEAN, is the mean normalized score over the three performance

metrics , four problem data sets, and five trials. The learning algorithms are sorted by descending order by the mean normalized scores of this MEAN last column.

In the table entries, the algorithm with the best performance on each performance metric (each column) is **boldfaced**. Aside from the algorithm with the best performance, other algorithm's performance that were not significantly different from the algorithm with the best performance by paired t-test at p = 0.05 has an asterisk (*) on the entry. Values without any sort of **boldface** or asterisk (*) signifies that the algorithm performed significantly worse than the best performing algorithm. The p-values from the paired t-tests performed will be included as secondary results, and this can be found in the Appendix section as Table 6. Null hypothesis statistical testing is not going to be well performed at merely 5 trials, but it was included with the consideration to the importance of data.

Table 2: Normalized scores for each learning algorithm by metric

MODEL	ACC	F1	ROC AUC	MEAN
RF	**0.979237**	**0.879064**	**0.914069**	**0.924123**
LR	0.979051*	0.877122*	0.904680	0.920284
DT	0.977237	0.868157*	0.907909*	0.917768
KNN	0.977717*	0.868224*	0.874881*	0.906941

Just at a glance of Table 2, the Random Forest algorithm performed the best across the board in all performance metrics and its normalized scores. In terms of ACC performance metric, Logistic Regression and K-Nearest Neighbors had no significant difference from the best performing algorithm (Random Forest). However despite its normalized ACC score, the Decision Tree algorithm performed significantly lower than the best performing algorithm. In terms of F1 performance metric, all classification algorithms performed relatively similar to each other with no significant difference. In terms of ROC AUC performance metric, Logistic Regression performed significantly lower than the Random Forest, while Decision Tree and K-Nearest Neighbors performed relatively similar.

3.2 Performances by Problem Data Sets

Table 3 shows the normalized scores for each algorithm on each of the four problem data sets. This contains the mean testing set performance across trials for each algorithm and performance metric combination, with the respect to each of the data sets. Each entry in the table averages the scores from five trials and performance metrics for that particular problem data set. The last column, MEAN, is the mean normalized score over the three performance metrics, four problem data sets, and five trials. The learning algorithms are sorted by descending order by the mean normalized scores of this MEAN last column.

Similar to 'Performances by Metrics', paired t-tests were done using the raw score values for each entry in the table against the raw score values of the best performing algorithm

in the column to gauge its significance at p = 0.05. The best performing algorithm in each column is **boldfaced** while the algorithms with no significant difference to the best performing algorithm are marked with an asterisk (*). The p-values from the paired t-tests performed will be included as secondary results, and this can be found in the Appendix section as Table 7.

Table 3: Normalized scores for each learning algorithm by problem dataset

MODEL	ADULT	GRID	HTRU2	OCCUPANCY	MEAN
RF	**0.924123**	**0.924123**	**0.924123**	**0.924123**	**0.924123**
LR	0.920284*	0.920284*	0.920284*	0.920284*	0.920284*
DT	0.917768*	0.917768*	0.917768*	0.917768*	0.917768*
KNN	0.906941*	0.906941*	0.906941*	0.906941*	0.906941*

While the learning algorithms' scores are different from each other, the anomaly that can be seen at a glance of Table 3 is the uniform scores throughout all the problem datasets. Immediately doubting the resulting data, the experiment pipeline and data processing was triple checked to ensure that no errors were made that led to the uniform scores. Even the raw scores were individually checked to make sure that no errors were made when on score retrieval at each trial; however, no particular issues were found. Throughout the problem data sets, Random Forest also performed the best out of all the other algorithms as expected from the CNM06.

3.3 Performance by Training Sets

Because the training set scores were also recorded at the time of the trials, the performance of the algorithms with the training set were also measured similar to previous measurements done using the testing set. The performance by metric for the training set can be seen in the Appendix section as Table 4, while the performance by problem data sets for the training set can be seen in the same section under Table 5.

While no paired t-tests were performed to gauge the significance of performance difference across algorithms, the best performing algorithm in each column are **boldfaced**. The learning algorithms performed as expected and the results were in parallel to the findings observed in the CNM06; the Random Forest and K-Nearest Neighbors performed the best at solid 1.0 score across the board, while Logistic Regression and Decision Tree fell behind slightly. The scores in general were higher than the scores observed in the testing set, and this can be deduced by the fact that the classifiers were trained using the training set.

4. Conclusion

Based on the results and the scoring data observed from this comparison, there are no hesitations to say that this study was successful in terms of replicating the comprehensive comparison conducted by Caruana and Niculescu-Mizil. While there were some anomalies, the supervised learning algorithms all performed within expectations. Random Forest have consistently shown to perform the best across the four binary classification problem data sets and three performance metrics. Also within expectation, Decision Trees performed consistently on the lower end of the learning algorithms.

The anomalies to this study were Logistic Regression and K-Nearest Neighbors. Logistic Regression unexpectedly performed better on certain performance metrics and throughout the problem data sets. Meanwhile, K-Nearest Neighbors performed worse than expected on testing set, but remained true to the expectations on the training set scoring results. In terms of other comparison specifications, F1 consistently had lower scores compared to the other performance metrics, which is also reflected in the CNM06. It is also notable that the binary classification problem data sets used in this comparison study were mostly imbalanced, which can attribute to the volatile performance of the K-Nearest Neighbors. Nonetheless, the overall results of this comparison were reasonably in parallel to the findings observed in the CNM06 performed by Caruana and Niculescu-Mizil (2006).

5. Bonus

The requirement of this comparison study was to perform binary classification model comparison with 3 supervised learning algorithms, 4 different problem data sets, and 5 trials. This turns out to 60 total trials, but 4th algorithm was included in the comparison study. Therefore the comparison study was performed on 4 supervised learning algorithms, 4 different problem data sets, and 5 trials, which turns to 80 total trials performed. This also means that additional analysis was done between algorithms, metrics, and secondary analysis. The algorithms were carefully selected to ensure that they covered all the performance ranges.

Acknowledgments

I personally would like to acknowledge the support for this study from the instructional staff of COGS 118A at UCSD Department of Cognitive Science for their generous time commitment to answer all Piazza posted questions ASAP. Additionally, I want to thank Professor Fleischer for his endless support by offering additional office hours outside of working hours, as well as providing guidance with detailed answers to questions and providing the framework for this study via Lecture 19 workbook.

Appendix

Table 4: Normalized scores for each learning algorithm by metric (Training set)

MODEL	ACC	F1	ROC AUC	MEAN
RF	**1.000000**	**1.000000**	**1.000000**	**1.000000**
KNN	**1.000000**	**1.000000**	**1.000000**	**1.000000**
DT	0.98188	0.897972	0.927506	0.935786
LogReg	0.97988	0.885514	0.910729	0.925374

Table 5: Normalized scores for each learning algorithm by problem dataset (Training set)

MODEL	ADULT	GRID	HTRU2	OCCUPANCY	MEAN
RF	**1.000000**	**1.000000**	**1.000000**	**1.000000**	**1.000000**
KNN	**1.000000**	**1.000000**	**1.000000**	**1.000000**	**1.000000**
DT	0.935786	0.935786	0.935786	0.935786	0.935786
LogReg	0.925374	0.925374	0.925374	0.925374	0.925374

Table 6: P-Value for Performances by Metric (Table 2)

MODEL	ACC	F1	ROC AUC	MEAN
RF	NaN	NaN	NaN	NaN
LR	1.468651e-01	3.145009e-02	2.616912e-06	0.306631
DT	3.443900e-11	1.393094e-10	9.339767e-02	0.132168
KNN	2.114964e-11	4.557926e-11	5.161585e-15	0.268552

Table 7: P-Value for Performances by Problem Data Sets (Table 3)

MODEL	ADULT	GRID	HTRU2	OCCUPANCY	MEAN
RF	NaN	NaN	NaN	NaN	NaN
LR	0.025984	0.025984	0.025984	0.025984	0.0
DT	0.028778	0.028778	0.028778	0.028778	0.0
KNN	0.001815	0.001815	0.001815	0.001815	0.0

Because the raw scoring data was too large, no separate table was made; however, the raw test set scores data can be seen in code cell 29.

Because the CODE and the output from the CODE was too large, it is included after the References section.

References

L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler, R. Layton, J. VanderPlas, A. Joly, B. Holt, and G. Varoquaux. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pages 108–122, 2013.

R. Caruana and A. Niculescu-Mizil. An empirical comparison of supervised learning algorithms. *Proceedings of the 23rd international conference on Machine learning*, ICML '06: 161–168, 2006.

D. Dua and C. Graff. UCI Machine Learning Repository, 2017. URL `http://archive.ics.uci.edu/ml`.

J. G. Fleischer. Lecture 19 Model Selection, 2021. URL `https://github.com/jasongfleischer/UCSD_COGS118A/blob/main/Notebooks/Lecture_19_model_selection.ipynb`.

R. D. King, C. Feng, and A. Sutherland. STATLOG: Comparison of Classification Algorithms on Large Real-world Problems. *Applied Artificial Intelligence*, 9(3):289–333, 1995.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.