Small-Scale Recreation of CNM06

Melvin Chang MECHANG@UCSD.EDU
COGS 118A: Supervised Machine Learning
Department of Cognitive Science
University of California, San Diego

Editor: N/A

Abstract

This project aims to replicate the methods and results of the 2006 Caruana and Niculescu-Mizil paper (CNM06) which compares the performance of several machine learning algorithms on several data sets for binary classification. My project compares the performance of three supervised machine learning algorithms across four different data sets. The algorithms that were used are logistic regression, random forest, and k-nearest neighbors. This was conducted by using a grid search to find the optimal hyper-parameters of a given algorithm based on three metrics (accuracy, roc auc, & f1). The procedures from the CNM06 paper were carefully followed in order to reproduce results which support the findings of Caruana and Niculescu-Mizil.

Keywords: Supervised Machine Learning, Logistic Regression, Random Forest, k-Nearest Neighbors

1. Introduction

With the evolution of machine learning algorithms and the introduction of new learning algorithms, Caruana and Niculescu-Mizil sought to evaluate the performance of older learning algorithms compared to the performance of the newer learning algorithms (e.g., SVMs, random forest, bagging). In the CNM06 paper, algorithms were evaluated on a multitude of performance metrics where a classifier may perform well on one metric, yet perform poorly on other metrics.

Ultimately, the CNM06 paper reports results where bagged trees, random forests, and neural nets produce the best performances over eight metrics and eleven different data sets, whereas memory-based learning algorithms like logistic regression, boosted stumps, and naive bayes perform among the worst. However, they do acknowledge that boosted stumps and logistic regression have shown to be the best models for some metrics on two of their data sets (Caruana, 2006).

In this project, I attempt to recreate CNM06's machine learning algorithm performance findings by using a subset of algorithms (logistic regression, random forest, & k-nearest neighbors) and data sets (ADULT, LETTER.AM, LETTER.O, & COVTYPE) as specified in the paper. The performance of these algorithms were measured based on three scoring metrics: accuracy, roc auc, and f1 score.

My small-scale replication of CNM06 seem to achieve results which support the claims of CNM06. Although my replication of CNM06 only uses 3 models (logisitic regression, random forest, and k-nearest neighbors), my findings corroborate with the claims in the CNM06 paper. Random forest had produced the best performance across my three metrics, whereas logistic regression performed the worst (see Table 2). My results also show that k-nearest neighbors produced scores which are not significantly different than random forest by metric across all four data sets.

2. Methodology

2.1 Learning Algorithms

Three algorithms were used in the conduction of this experiment. They are logistic regression, random forest, and k-nearest neighbors. Parameters for each of these algorithms follow close to those used on the CNM06 paper, but may differ slightly. The exact specifications for each algorithm are as follows:

Logistic Regression (LOGREG): Both saga and LBFGS solvers were included in the space of parameters. The penalties that were used were *l1, l2*, and *none*, and finally the regularization parameter ranged from 10^{-4} and 10^{4}.

Random Forest (RF): Most parameters remain default; however, there are 1024 trees in the forest. Like the parameters stated in the CNM06 paper, the maximum number of features per tree split is: 1, 2, 4, 6, 8, 12, 16, or 20.

k-Nearest Neighbors (KNN): Both *uniform* and *distance* weights are considered, and uses Euclidean distance as its distance metric. Values of K range from the set: 1, 5, 9, ..., 101, 105. The set consists of 26 values which increment by 4 starting from 1 until 105 neighbors.

2.2 Performance Metrics

The performance metrics used for this analysis are accuracy (ACC), ROC AUC (ROC), and F1 (FSC). All of which have values which range from [0,1]. Best models aim to have a value close to 1.

The accuracy classification score computes the sum of true positives and the sum of true negatives over the entire sample population. This can be represented with the following equation:

$$Accuracy = \frac{\sum True_{pos} + \sum True_{neg}}{TotalPopulation_{sample}} \tag{1}$$

The ROC AUC score computes the area under the receiver operating characteristic curve. This metric is a performance measurement for the classifier at various threshold settings. The ROC curve is a probability curve where the true positive rate is plotted against the false positive rate.

The F1 score computes the weighted average of the precision and recall. This can be represented with the following equation:

$$FSC = \frac{2 * (Precision * Recall)}{Precision + Recall} \tag{2}$$

2.3 Data Sets

Table 1: Description of Problems

	# ATTR	TRAIN SIZE	TEST SIZE	% POS
ADULT	105	5000	27561	24%
LETTER.AM	16	5000	15000	49.7%
LETTER.O	16	5000	15000	3.77%
COV_TYPE	54	5000	576012	48.8%

The three data sets used in the experiment are all made available to us through the UCI Machine Learning Repository. These data sets are: ADULT, LETTER, AND COVTYPE (see Table 1). For each data set, continuous features were scaled, and categorical features were one-hot encoded.

The ADULT data set consists of 14 features where 8 features are categorical features. One continuous feature, *education-num* is dropped as it is already represented as a categorical feature through the *education* feature. Thus, there are a total of 105 attributes for this data set. The target variable, *income*, is treated as a binary classification problem by converting those who make more than 50k as the positive class and those who make less than or equal to 50k as the negative class. Those who are in the positive class are assigned a 1 and those in the negative class are assigned a -1. This makes for a partially imbalanced data set where only 24% are represented in the positive class.

The LETTER data set can be represented as two different data sets. First, the LETTER data set can be converted to a binary problem by treating the letters A-M as the positive class and the letters N-Z as the negative class. This leads to a quite balanced data set where 49.7% are represented as the positive class. Second, like the CNM06 paper, this data set also treats the letter "O" as the positive class, and every other remaining letter as the negative class. This results in quite an imbalanced data set where only 3.77% of the data set is represented in the positive class. This data set consists of 16 continuous features for a total of 16 attributes.

The COVTYPE data set is converted to a binary classification problem by treating the largest number of cover type in the data set as the positive class, and every other cover type number as the negative class. Thus, those in the *cover type* column containing a 2 are assigned a 1, and the rest are assigned a -1. The positive class makes up for 48.8% of the data set. Out of the 55 features, only 10 features need to be scaled. The remaining 44 features are already represented as binary columns.

3. Experiment & Results

For this experiment, five trials of a cross-validation grid search is conducted on three algorithms across four separate data sets which allow for the tuning of the optimal hyperparameters per algorithm. For a single trial, 5000 data samples are randomly sampled and run through a 5-fold cross-validation grid search, training on every specified classifier. From this output, we are able to obtain the best parameters of an algorithm for each metric (ACC, ROC-AUC, & F1).

With these optimal hyper-parameters, we have three different models for a single algorithm which scored best in ACC, ROC, and FSC. These models are then retrained on the same training data, and used to predict on the test set. The following tables, Table 2 and Table 3 represent the mean results for each algorithm over four problems and over three metrics, respectively. In these two tables, the best performing algorithm(s) is/are marked in **bold**, and scores that are not significantly different are marked with an asterisk (*), meaning that the algorithm performs about as well as the bold-faced algorithm.

Table 2: Test Set Performance by Metric (Average over 4 Problems)

	ACC	ROC	F1	MEAN
LOGREG	.822	.850	.532	.735
RF	**.902**	**.948**	**.815**	.888
KNN	.891*	.933*	.806*	.877

In Table 2, we see the test set performance for each algorithm-metric combination averaged over each trial and each of the four data sets. Consistent with the results in the CNM06 paper, the mean score performances for random forest performed the best compared to logistic regression and k-nearest neighbors. Although the performances for random forest are the best among these three algorithms, we can also see that the k-nearest neighbors algorithm performed just as well as the random forest algorithm (see Table 6 in Appendix for p-values).

Table 3: Test Set Performance by Problem (Average over 3 Metrics)

	ADULT	Letter.AM	Letter.O	COV	MEAN
LOGREG	.801*	.753	.607	.778	.735
RF	**.806**	.961*	.938*	**.849**	.889
KNN	.780*	**.965**	**.952**	.810	.877

In Table 3, we see the test set performance for each algorithm-data set combinations averaged over each trial and each of the three scoring metrics. From this table, we can see how random forest tops the performances under two data sets, whereas k-nearest neighbors tops the performances for both of the LETTER data sets.

In the ADULT data set, we have that the random forest algorithm performed the best; however, t-tests show insignificant differences between random forest and logistic regression, and random forest and k-nearest neighbors (see Table 7 in Appendix 7 for p-values).

In the LETTER.AM data set, we have that the k-nearest neighbors algorithm held the best performance among the three algorithms. However, there is an insignificant difference when compared to the random forest algorithm (see Table 7 in the Appendix). We can also see that logistic regression performed significantly worse despite being a fairly balanced data set.

In the LETTER.O data set, we have that the k-nearest neighbors algorithm continued to perform the best among the three algorithms. However, once again, there is an insignificant difference when compared to the random forest algorithm (see Table 7 in the Appendix).

Additionally, we have that the logistic regression algorithm produces incredibly poor results given an incredibly imbalanced data set.

Lastly, in the COV data set, we have that the random forest algorithm performed the best among the three algorithms. Furthermore, t-tests from Table 7 of the Appendix appear to show that there are no insignificant differences when the random forest algorithm is compared to the two other learning algorithms.

Generally speaking, random forests seem to perform well across all data sets and metrics. Additionally, k-nearest neighbors seem to perform almost as well as the random forests, while the memory-based learning algorithm, logistic regression, performs among the worst. These results corroborate with the claims made by Caruana and Niculescu-Mizil.

4. Conclusion

I believe my findings were able to validate the findings in the CNM06 paper given my usage of less machine learning algorithms, data sets, and scoring metrics. The random forest algorithm was shown to perform consistently well across the metrics I had used as well as across each of the four data sets that were used. On the other hand, the memory-based learning algorithms, logistic regression and k-nearest neighbors, performed as we would expect from the CNM06 paper. Despite inconsistencies across data sets, k-nearest neighbors would still, on average, perform better than logistic regression. Thus, given these three algorithms, we have that random forest performs the best, whereas logistic regression performs the worst. Although my testing was not as extensive as the CNM06 paper, we can conclude that my findings are somewhat representative of what was claimed in the CNM06 paper.

Acknowledgments

I would like to acknowledge the COGS 118A staff as well as the students for their support with this project. Active contribution by both the staff and students have helped me significantly with both the implementation and understanding of this project. I would also like to acknowledge Professor Fleischer for providing numerous resources of help ranging from posted office hour recordings, access to jupyter notebooks including but not limited to feature representation and model selection, and general help through responses Piazza.

Appendix

Table 4: Training Set Performance by Metric (Average over 4 Problems)

	ACC	ROC	F1	MEAN
LOGREG	.827	.856	.541	.741
RF	1.00	1.00	1.00	1.00
KNN	.977	1.00	.966	.981

Unsurprisingly, we have high scores for the random forest algorithm. Given the most optimal hyper-parameters, we would expect this algorithm to yield high scores on the training set. The k-nearest neighbors algorithm seems to produce inconsistent scores across data sets (which can be seen in Table 3) which may be problematic when wanting to generalize. On the other hand, we have logistic regression yielding low scores compared to both random forest and k-nearest neighbors. Compared to the test set performance (see Table 2), we should expect these scores to be, on average, higher than those found in Table 2 as they were trained on the same training data as the ones used in the cross-validation grid search.

Table 5.1: Raw Test Set Score Performance on Accuracy

	ADULT				
	TRIAL	TRIAL 2	TRIAL 3	TRIAL 4	TRIAL 5
LOGREG	0.850042	0.84659	0.848119	0.850913	0.847466
RF	0.849897	0.851638	0.851021	0.853706	0.850767
KNN	0.833134	0.838649	0.836835	0.837814	0.832626
	LETTER.AM				
	TRIAL 1	TRIAL 2	TRIAL 3	TRIAL 4	TRIAL 5
LOGREG	0.720400	0.73000	0.724267	0.720200	0.722933
RF	0.946867	0.945800	0.951000	0.944800	0.941467
KNN	0.952933	0.953533	0.953600	0.952333	0.954600
	LETTER.O				
	TRIAL 1	TRIAL 2	TRIAL 3	TRIAL 4	TRIAL 5
LOGREG	0.961267	0.962467	0.962867	0.961933	0.961867
RF	0.985400	0.988667	0.990200	0.988467	0.989400
KNN	0.991467	0.990600	0.989000	0.991067	0.989733
	COVER TYPE				
	TRIAL 1	TRIAL 2	TRIAL 3	TRIAL 4	TRIAL 5
LOGREG	0.754182	0.754182	0.759224	0.753682	0.752227
RF	0.823861	0.826809	0.826330	0.820420	0.820247
KNN	0.781579	0.783232	0.783083	0.782916	0.785096

Table 5.2: Raw Test Set Score Performance on ROC AUC

	ADULT				
	TRIAL	TRIAL 2	TRIAL 3	TRIAL 4	TRIAL 5
LOGREG	0.904451	0.903426	0.905164	0.904053	0.902377
RF	0.900433	0.903124	0.900143	0.905685	0.903196
KNN	0.883103	0.887285	0.886553	0.883853	0.884475
	LETTER.AM				
	TRIAL 1	TRIAL 2	TRIAL 3	TRIAL 4	TRIAL 5
LOGREG	0.811108	0.812500	0.812471	0.810595	0.811377
RF	0.990355	0.991099	0.991418	0.990945	0.989088
KNN	0.988659	0.990085	0.989743	0.989118	0.990496
	LETTER.O				
	TRIAL 1	TRIAL 2	TRIAL 3	TRIAL 4	TRIAL 5
LOGREG	0.859269	0.854011	0.863343	0.860515	0.854739
RF	0.996582	0.996991	0.997103	0.997916	0.996722
KNN	0.986908	0.992086	0.994596	0.994232	0.995394
	COVER TYPE				
	TRIAL 1	TRIAL 2	TRIAL 3	TRIAL 4	TRIAL 5
LOGREG	0.824484	0.824716	0.823864	0.824396	0.824663
RF	0.901971	0.904860	0.903450	0.903520	0.899825
KNN	0.863367	0.863243	0.865375	0.865189	0.862883

Table 5.3: Raw Test Set Score Performance on F1

	ADULT				
	TRIAL	TRIAL 2	TRIAL 3	TRIAL 4	TRIAL 5
LOGREG	0.660087	0.645955	0.639449	0.658182	0.642760
RF	0.658973	0.659222	0.661221	0.667765	0.669187
KNN	0.600954	0.620531	0.622640	0.634206	0.612581
	LETTER.AM				
	TRIAL 1	TRIAL 2	TRIAL 3	TRIAL 4	TRIAL 5
LOGREG	0.721069	0.733693	0.721631	0.721777	0.726363
RF	0.946297	0.945056	0.950964	0.944593	0.940812
KNN	0.952896	0.953300	0.953207	0.951984	0.954512
	LETTER.O				
	TRIAL 1	TRIAL 2	TRIAL 3	TRIAL 4	TRIAL 5
LOGREG	0.000000	0.000000	0.000000	0.000000	0.000000
RF	0.781219	0.837476	0.855172	0.824721	0.846079
KNN	0.885510	0.874667	0.856895	0.880143	0.864912
	COVER TYPE				
	TRIAL 1	TRIAL 2	TRIAL 3	TRIAL 4	TRIAL 5
LOGREG	0.753143	0.748730	0.758968	0.755063	0.752054
RF	0.822566	0.826916	0.823174	0.817224	0.820058
KNN	0.775175	0.782389	0.785174	0.781706	0.785343

Table 8: P-Vals for Table 2 (RF compared with LOGREG and KNN)

	P-Val - LOGREG	P-Val KNN
RF - ACC	.004	.641
RF - ROC	6.87e-09	.373
RF - FSC	.001	.818

Table 9: P-Vals for Table 3 (RF compared with LOGREG and KNN over Data Sets)

	RF
ADULT (LOGREG)	.899
ADULT (KNN)	.535
LETTER.AM (LOGREG)	4.44e-16
LETTER.AM (KNN)	.526
LETTER.O (LOGREG)	.009
LETTER.O (KNN)	.602
COV (LOGREG)	1.09e-5
COV (KNN)	.010

References

[1] Caruana and A. Niculescu-Mizil. "An empirical comparison of supervised learning algorithms" *In Proceedings of the 23rd international conference on Machine learning*, 161-168, 2006.

[2] Logistic Regression, scikit-learn.org,
`https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html#`

[3] K-Nearest Neighbors, scikit-learn.org,
`https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html`

[4] Metrics, scikit-learn.org,
`https://scikit-learn.org/stable/modules/model_evaluation.html`

[5] Random Forest, scikit-learn.org,
`https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html`