FINAL PROJECT: COGS 118A

COGS 118A: Classification Model Comparison

Authors: Chace McGuyer, Oscar Bailon

Abstract

The sole purpose of this comparison project is to fairly extrapolate which of the three algorithms performs better on datasets with the foundational structure and method of the CNM06 research paper. The overall premise of this paper is to perform 3 different algorithms(KNN, DecisionTree,and Support-Vector Machines(SVM)) on 4 datasets to compare the overall success of binary classifications through metrics of performance. Speculative information of performance was drawn from the Caruana Paper as it was stated that SVMs and Trees tend to do astronomically better than memory-based algorithms and Logistic Regression. This will be investigated and analyzed through GridSearch, which will provide us with the optimal hyperparameters to test on the datasets, which will be split as 5000 initial points and the rest as the test set utilized in determining performance. Explicit description and results will evince the discoveries of CNM06, while taking into consideration that the No Free Lunch Theorem specifies that algorithms have a tendency to perform well under specific scenarios/problems; therefore, we cannot generalize the conclusion to all datasets but only to those imported in this paper. We are also taking into account the Parsimony theory, which elaborates on the ideology that different variations of a model/algorithm with nearly identical measures of accuracy, precision, etc are indifferent in results; therefore, we shall proceed with the more parsimonious(simpler) model.

I. Introduction

To commence and initiate the structure of this report, we will first introduce the underlying pinnings of the Caruana and Niculescu paper (CNM06). To summarize the motivations of the study, there was an interest in replicating the process of comparative analysis of modern methods/algorithms to determine the optimal,newer model. This was previously executed in the famous STATLOG study, which motivated Caruana to reassess older algorithms with newer models like SVM variations, boosted and bagged trees, higher order trees such as random forests and more post-STATLOG-study models(King et al., 1995). In contrast to the STATLOG study, Caruana implemented various metrics that were not considered in the STATLOG study due to their specification to a certain field/professional setting. Caruana decided that tampering with this social construct would allow for a significantly higher measure of mean performance per algorithm with datasets that traversed from the medical field to letter datasets. These niched data sets ranged from nominal data, transformed into Boolean, to continuous and binary data types that provided an overall perspective on the power of each algorithm in a specific scenario, as previously stated with the No Free Lunch Theorem.

The conclusive information that Caruana gathered from the study demonstrated how trees, specifically boosted and random forests outperformed the majority of algorithms that were used. Contrary to the trees and forest algorithms, exhaustive search like tree stumps and the sigmoid-recognized algorithm of Logistic Regression were amongst the worst algorithms throughout the datasets and trials. The entire purpose of these research papers are to instigate conversation and reproductivity for further evidence to findings. According to this caveat, we are to check or reproduce work published for the purpose of creating concrete evidence or further advance the study. In this paper, we are going to be performing a miniature replica experiment to learn about the scientific process of algorithmic comparison and provide evidence of replicability for CNM06. The difference in this paper is that we are going to resort to 3 algorithms (SVM, DecisionTrees, and Logisticregression) with four distinct datasets than the 11 datasets and 8 metrics that CNM06 used. At the end of this, this experiment would demonstrate that the results are still conceptually relevant/consistent(as concrete evidence) and are procedurally/systematically reproducible.

II. Methods

2.1 Learning Algorithms

In this comparison study three different algorithms are utilized. We use a memory-based

method: SVM, a tree method: Decisions Trees and K Neighbors. We used different hyperparameters in our grid search depending on the model that we were using. In the following text we detail the hyperparameters included in the grid search tuning, used through 5-fold cross validation.

Support Vector Machine (SVM):
The kernels used in the SVM were linear, radial basis function and sigmoid. For the regularization parameter the interval was c = [1,1E4]. The kernel coefficient for the rbf and sigmoid varied on the interval gamma = [1E-2,1E-6].

Decision Trees (DT):
We compared different criteria for splitting, using both gini and entropy measures. Max depth was compared between [1,30] integer values only. We chose to set the number of classes to 2 because this was a binary classification task. For the criterion we chose to use both gini and entropy to measure the spit. Splitter was set to best and random, while max_features was auto, sqrt or log2.

KNeighbors (KN):
We used 5 different hyper-parameters. For weights we varied between uniform and distance weight functions. Leaf size varied from [1,30] integer values. P was either 1 or 2. The distance metric used was either euclidean or manhattan distance. Number of parallel jobs was in the interval [1,10] discrete.

2.2 Performance Metrics

The primary metrics used to evaluate the models in this study were accuracy, f1, precision, recall and ROC. Accuracy will function as the basis of our measurements across models. However as some of our datasets are imbalanced quite significantly**,** the accuracy score becomes non-representative of the classification model. This causes problems where the accuracy score becomes inflated higher than it would be in a more general context (less skewed data). Because of this we will use f1, precision, recall and ROC on the highly skewed data. Precision and ROC operate despite the distribution of the class measuring the efficiency of classifiers ordering positive cases prior to negative cases. (Caruana 2006). The f1 metric is used to incorporate both precision and recall. Precision is that metric that measures how well the model correctly predicts positive from the group of positive labels. Recall will typically be small if the model leans towards negative classifications, this is in the case of negative imbalance.**Metrics formulations are added at the end of the paper along with the visuals.**
.

2.3 Data sets

The algorithms were implemented on 3 datasets that were gathered through distinct means. The first is a real-estate cold call dataset centered around the Central Valley in California that was obtained from Capitol Real Estate Group. The label being predicted is whether the individual answers the phone or not.

The second dataset was obtained from Kaggle and contains information about people who traveled. The label being predicted in this dataset is whether the traveler purchased travel insurance. The third dataset was obtained from the UCI machine Learning repository and contains information about individuals from the census. The label being predicted is whether the individual made more than 50 thousand dollars or not.

Call Data

Our first dataset (‘corrected_call_data.csv’) was obtained through Capitol Real Estate group. The dataset is a list of calls made to potential customers and the outcome of the call, whether they answered or not. Variables included measured things such as property type, time of day, day of week, number of call attempts etc. Overall there were 69 variables and 14587 observations used in the analysis. This dataset is highly imbalanced with 1263 observations in the positive class. In order to get the data to this point cleaning had to be done, two csv files had to be merged on the phone number variable. Then many columns needed to be one-hot encoded.

Travel Insurance
The second dataset (‘corrected_travel_insurance.csv’) was obtained via Kaggle. The dataset contains a list of customers who are traveling, which agency and the type of agency used as well as some other data. The label was whether or not the customer purchased Travel insurance. This dataset included 202 variables

and 63326 observations. This dataset is also highly imbalanced with only 927 observations in the positive class. Many categorical variables such as the agency name, type and travel destination were one-hot encoded.

Census Income:

The third dataset ('census_income.csv') was a list of individuals who filled out their census, some information about them such as years of education, hours worked per week, whether they were self-employed or obtained a master's degree. The label being predicted was whether or not the individual makes more than 50 thousand dollars a year or not. This dataset has a total of 109 variables with 32561 observations. There are 7841 observations in the positive class making this dataset more balanced than the other two. This dataset also required some pre-processing such as one-hot encoding of categorical variables.

III. Experiment

The procedure taken for this paper was to extract 5000 data instantiations and utilize a 5-fold cross validation to take in the most amount of information into each system(algorithm). This would ensure the best possible learning for equal comparison.

Grid search will be used to find the optimal hyperparameters to utilize on the test set, which is total_number_instantces - 5000. This ensures that the distributive datasets are mostly displaying the results of large testing with determined optimal hyperparameters from the training set. This is done procedurally onto all datasets listed in the 2.3 section, with 5 trials (of 5000 instantiations each) per algorithm and dataset, providing results for each combination.

IV. Results:

Overall the DT algorithm performed the best across all trials and datasets on average. The SVM model came in second place with KNeighbors performing the worst. Due to the fact that two of our datasets were imbalanced, accuracy becomes a less viable score. The metric that we are most concerned with is the precision metric as it represents the number of true positives predicted out of all true positives.

	model	accuracy	precision	f1	recall	roc_auc
0	SVM	0.929831	0.555000	0.296932	0.248490	0.759180
1	DT	0.979415	0.571376	0.410214	0.400068	0.805709
2	KNeighbors	0.927631	0.387390	0.284702	0.241932	0.655876

In the precision metric the DT model still performed the best. However, if you reference table _____ you can see that the DT model over fit on one of the datasets obtaining a perfect score in each trial. This would cause the results to be highly skewed in favor of DT. Because of this we believe that SVM is most likely the more accurate model of the three.

We have included all three results tables in the tables section of this paper.

V. Discussion

Conclusion

The results from this paper corroborate the CNM06 paper's findings as SVM was the optimal model for the these datasets, DecisionTree was the next proximate model had it not been over fit and lastly, KNN was overall the worst, yet follow the No Free Lunch Theorem as they would all excel at different metrics for specifics situations. As expected KNN and clustering algorithms generally do worse than Forest and Tree algorithms all while falling behind the well structured SVM model.

Bonus

Due to extensive work done on the Real Estate Dataset, from data wrangling untidy and unmerged data to setting up meetings with the IT department and owner of the properties for development and idealization of work, we have had to delay our project starting time. Not only this, one-hot encoding the dataset and merging different components from other datasets consumed much of our time and delayed the running of the final script, which was already on track to surpass 30 hours of runtime. We had to cut the script short and provide results only on the first 3 datasets of the 4, which wasn't ideal but very much practical for our

case.Also, we extended the number of metrics from the default 3 to 5 for a more elaborate comparison of model behavior and performance.To further elaborate, we were planning on doing 3 algorithms, on 4-5 datasets, with 5 trials each and 5 performance metrics.

Future Improvements

As the purpose of this experiment was to prove reproducibility, we would like to advise for future scientists, that want to reproduce or further expand on this research, to incorporate modern algorithms, measure different metrics, utilize problem-specific datasets to visualize the effects of the No Free Lunch Theorem and create graphs to show performance over trials(learning curves),time complexity(temporal resolution) of algorithms and progressive results of trails.

VII. Tables:

	model	accuracy	precision	f1	recall	roc_auc
0	SVM	0.929831	0.555000	0.296932	0.248490	0.759180
1	DT	0.979415	0.571376	0.410214	0.400068	0.805709
2	KNeighbors	0.927631	0.387390	0.284702	0.241932	0.655876

Table 1: performance metrics averaged across trials and datasets

	model	dataset	accuracy	precision	f1	recall	roc_auc
0	SVM	corrected_call_data	0.97328	0.932639	0.531323	0.441256	0.853169
1	SVM	corrected_travel_insurance	0.98540	0.150848	0.078765	0.068762	0.725263
2	SVM	census_income	0.76615	0.612479	0.268007	0.224467	0.653894
3	DT	corrected_call_data	0.96108	0.604085	0.362129	0.343795	0.723693
4	DT	corrected_travel_insurance	0.98540	0.281492	0.104427	0.096381	0.771152
5	DT	census_income	1.00000	1.000000	1.000000	1.000000	1.000000
6	KN	corrected_call_data	0.96592	0.619080	0.501502	0.436385	0.745824
7	KN	corrected_travel_insurance	0.98508	0.061333	0.018655	0.011238	0.578087
8	KN	census_income	0.76880	0.548566	0.371013	0.306279	0.633236

Table 2: performance metrics averaged across datasets

	model	dataset	trial	accuracy	precision	f1	recall	roc_au
0	SVM with rbf	corrected_call_data.csv	0	0.9718	0.864286	0.538286	0.441538	0.87642
1	Decision Trees	corrected_call_data.csv	0	0.9610	0.572381	0.384394	0.339231	0.72576
2	KNeighbors	corrected_call_data.csv	0	0.9682	0.678846	0.536496	0.461923	0.75555
3	SVM with rbf	corrected_call_data.csv	1	0.9758	0.918977	0.579090	0.541282	0.84260
4	Decision Trees	corrected_call_data.csv	1	0.9606	0.528571	0.392791	0.375000	0.74539
5	KNeighbors	corrected_call_data.csv	1	0.9654	0.581687	0.520597	0.476667	0.74942
6	SVM with rbf	corrected_call_data.csv	2	0.9732	0.987500	0.524107	0.396282	0.85833
7	Decision Trees	corrected_call_data.csv	2	0.9606	0.419472	0.330067	0.315513	0.69157
8	KNeighbors	corrected_call_data.csv	2	0.9654	0.599419	0.469926	0.391282	0.74841
9	SVM with rbf	corrected_call_data.csv	3	0.9728	0.968485	0.498782	0.425769	0.85322
10	Decision Trees	corrected_call_data.csv	3	0.9610	0.600000	0.346856	0.344103	0.71951
11	KNeighbors	corrected_call_data.csv	3	0.9654	0.592017	0.488731	0.425256	0.72996
12	SVM with rbf	corrected_call_data.csv	4	0.9728	0.923950	0.516350	0.401410	0.83524
13	Decision Trees	corrected_call_data.csv	4	0.9622	0.900000	0.376535	0.345128	0.73621
14	KNeighbors	corrected_call_data.csv	4	0.9652	0.643431	0.491758	0.426795	0.74578
15	SVM with rbf	corrected_travel_insurance.csv	0	0.9854	0.155556	0.083275	0.054286	0.72480
16	Decision Trees	corrected_travel_insurance.csv	0	0.9854	0.266667	0.097391	0.082857	0.79539
17	KNeighbors	corrected_travel_insurance.csv	0	0.9850	0.166667	0.047222	0.027619	0.55600
18	SVM with rbf	corrected_travel_insurance.csv	1	0.9854	0.355556	0.128544	0.108571	0.68656
19	Decision Trees	corrected_travel_insurance.csv	1	0.9854	0.400000	0.133030	0.124762	0.76627
20	KNeighbors	corrected_travel_insurance.csv	1	0.9852	0.000000	0.000000	0.000000	0.60532
21	SVM with rbf	corrected_travel_insurance.csv	2	0.9854	0.046667	0.043423	0.040952	0.72989
22	Decision Trees	corrected_travel_insurance.csv	2	0.9854	0.128889	0.084162	0.080000	0.74999
23	KNeighbors	corrected_travel_insurance.csv	2	0.9854	0.000000	0.000000	0.000000	0.57268
24	SVM with rbf	corrected_travel_insurance.csv	3	0.9854	0.093434	0.087199	0.083810	0.74399
25	Decision Trees	corrected_travel_insurance.csv	3	0.9854	0.200000	0.101270	0.099048	0.76872
26	KNeighbors	corrected_travel_insurance.csv	3	0.9852	0.140000	0.046053	0.028571	0.56387
27	SVM with rbf	corrected_travel_insurance.csv	4	0.9854	0.103030	0.071385	0.056190	0.74104
28	Decision Trees	corrected_travel_insurance.csv	4	0.9854	0.411905	0.106281	0.095238	0.77537
29	KNeighbors	corrected_travel_insurance.csv	4	0.9846	0.000000	0.000000	0.000000	0.59254
30	SVM with rbf	census_income.csv	0	0.7642	0.598110	0.256607	0.214319	0.65104
31	Decision Trees	census_income.csv	0	1.0000	1.000000	1.000000	1.000000	1.00000
32	KNeighbors	census_income.csv	0	0.7656	0.531689	0.356843	0.295733	0.62970
33	SVM with rbf	census_income.csv	1	0.7654	0.592420	0.278105	0.235875	0.65716
34	Decision Trees	census_income.csv	1	1.0000	1.000000	1.000000	1.000000	1.00000
35	KNeighbors	census_income.csv	1	0.7692	0.549558	0.373944	0.307282	0.63654
36	SVM with rbf	census_income.csv	2	0.7648	0.607035	0.274957	0.230080	0.66897
37	Decision Trees	census_income.csv	2	1.0000	1.000000	1.000000	1.000000	1.00000
38	KNeighbors	census_income.csv	2	0.7694	0.552758	0.369552	0.303987	0.64059
39	SVM with rbf	census_income.csv	3	0.7702	0.652351	0.282358	0.217597	0.63839
40	Decision Trees	census_income.csv	3	1.0000	1.000000	1.000000	1.000000	1.00000
41	KNeighbors	census_income.csv	3	0.7710	0.560260	0.383713	0.318115	0.62611

Table 3: performance metrics by tri

VI. References

R. Caruana and A. Niculescu-Mizil. ”An empirical comparison of super
gorithms.”
In Proceedings of the 23rd international conference on Machine learning
,
161-168. 2006.
Available at: https://www.cs.cornell.edu/~caruana/ctp/ct.papers/caruana.

K-Means:

https://scikit-learn.org/stable/modules/clustering.html#k-means
https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans

SVM:
https://scikit-learn.org/stable/modules/generated/sklearn.sv

DecisionTree:
https://scikit-learn.org/stable/modules/generated/sklearn.tr

GridSearchCV:
https://scikit-learn.org/stable/modules/generated/sklearn.m
https://github.com/jasongfleischer/UCSD_COGS118A/blo

Datasets(links):
https://www.kaggle.com/mhdzahier/travel-insurance?select=travel+insurance.csv
https://archive.ics.uci.edu/ml/datasets/Bank+Marketing
https://archive.ics.uci.edu/ml/datasets/Census+Income

Metric Formulations:

$$F_1 = \frac{2}{\text{recall}^{-1} + \text{precision}^{-1}} = 2 \cdot \frac{\text{precisi}}{\text{precisi}}$$

$$\text{Accuracy} = \frac{tp + tn}{tp + tn + fp + fn}$$

$$\text{Precision} = \frac{tp}{tp + fp}$$

$$\text{Recall} = \frac{tp}{tp + fn}$$