Comparing Classifiers Based on Caruana and Niculescu-Mizil Using the UCI ML Repository

Jimmy Lozano
UCSD COGS 118A
Fall 2018
jnlozano@ucsd.edu

Abstract

With the increased popularity in machine learning and related fields, there has been an accompanied influx in statistical approaches to data. Because of this, it is difficult to maintain a complete and updated log recommending a certain statistical approach for a type of dataset. This project aims to replicate the findings of "*An Empirical Comparison of Supervised Learning Algorithms*" (Caruana and Niculescu-Mizil) by using similar data and approaches. While the original aims at a completed log of all known approaches, this project will focus primarily on KNN, SVM, XGBoost, Random Forests, ANN, and Decision Tree. Some data used in this project will overlap, such as adult and letter-recognition, but three will be different: credit approval, car, and bridges to provide additional insight on these methods.

1. Introduction

Rich Caruana and Alexandra Niculescu-Mizil aimed to provide additional research to the comprehensive STATLOG. This was done due to the continued emergence of new machine learning algorithms, obviously not acknowledged in STATLOG, that may either update certain algorithms in or even render theme obsolete. This same problem will continue to arise and is therefore an appropriate and even mandatory task for any aspiring student in machine learning.

The main objective of this project is to understand and test various algorithms. While the number of models that were tested in the original project were extensive, about 30, this project will focus on the more common models of those 30. Furthermore, these models will be tested on 5 different datasets, of which 3 are unique. Replicating Caruana's and Niculescu-Mizil's research will provide a strong reference to ensure the tested models are functional. It will also provide insight into real-world applications of models and utilizing data that isn't the easiest to work with. This will be done through data-cleansing, cross validation, usage of separate libraries, and hyper-parameter searching.

2. Methodology

2.1 Platform/Libraries

All models were tested on a consumer-grade laptop in Python through the Anaconda client. Libraries included, but were not limited to: xgboost, import_ipynb, scikit-learn, pandas, matplotlib, scipy, time, and seaborn.

2.2 Learning Algorithms

I attempted to research some of the most common models executed by Caruana and Niculescu-Mizil. This section will briefly describe and summarize the parameters utilized by the models that I did use. As a note, many of these models were primarily tested using relevant libraries from Python while a few were implemented without libraries. Also, many of the parameters will not exactly replicate those in the original research and will be noted.

KNN (K Nearest Neighbors): I used 12 values of K ranging from 1 through 25. The method was 'every other value' in the list of 24. This is comparable to the 26 values of 1 through |*trainset*| in the original research.

Random Forests (RF): I used the feature set at each split of values 1,2,4,6,8,12,16,20. This

matches the values in the original research. However, the original has 1024 trees while mine only has 200.

SVMs: I used two kernels, linear and RBF. The original uses linear, polynomial degree 2 & 3, and radial (rbf similar). The gamma parameter ranged from [500000.0, 20000.0, 5000.0, 200, 50, 2.0, 0.5, 0.125].

Boosted Decision Trees (BST-DT): I use the adaBoostClassifier from the scikit library with parameters ranging from the values: [2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048]. The original research has the same range of values but utilizes more trees.

XGBoost (Boosted family): Similar to adaBoostClassifier, it provides additional insight. Same parameters: [2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048]. An additional decision tree model, though the original research utilized much more tree models.

Neural Nets (NN): The momentum used here ranges from [10e-4, 10e-3, 10e-2], while the original ranges from 0 and 1. Furthermore, only one hidden layer is used here while the original research utilizes many hidden units.

2.3 Data Sets

Each of the above mentioned models were ran on 5 different datasets provided by UCI's Machine Learning Repo. As discussed earlier, 2 of these datasets overlap with the data used in the original research: 'adult.data' and 'letter-recognition.data'. The 'adult' dataset stems from the "Census Income Data Set" and is extracted from the 1994 Census database. This data is used to predict whether a person will make above $50,000 income per year. The dataset contains 14 attributes of which the predictive variables include gender, education, nationality, race, and other relevant information. The second overlapping dataset, 'letter-recognition', correlates to letter.p1. The preparation of this data was done the same as well in the sense that the letter 'O' was considered as a positive and the other letters a negative. The dataset contained 20,000 instances and contained 20 different fonts.

The three unique data chosen were credit approval data, car evaluation data, and bridges data. The credit approval data contains 15 attributes with 690 instances, providing for a relatively small dataset when compared to the others. This data is used to predict whether a person credit will get approved or not based on encrypted attributes to protect confidentiality. The car dataset contains 6 attributes and 1728 instances. This data classifies different types of car based on attributes such as price, number of doors, estimated safety, etc. The final dataset, Pittsburgh bridges, contains 13 attributes and 108 instances. There are two versions of this dataset of which I used the first version. Rather than a class domain, this dataset utilizes a design domain. This means that 5 properties must be predicted based on 7 of the attributes. Attributes include location, erection date, material used, etc.

3. Analysis

Each of the datasets were split 3 times in an 80/20, 50/50, 20/80 train/test ratio. The first figures to be shown will provide a general overview of the 80/20 split in the form of heatmaps. The rest of the models will pertain to a more detailed report on the splits.

3.1 Heatmaps

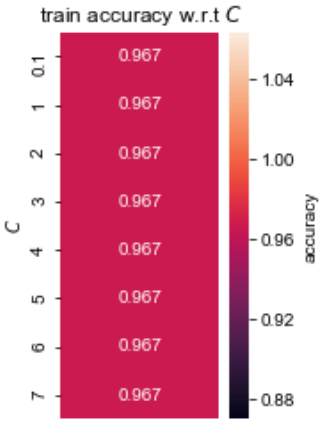

Figure 2: Training Accuracy of letter-recognition data

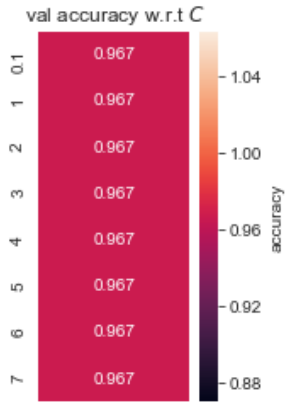

Figure 1: Validation Accuracy of letter-recognition data

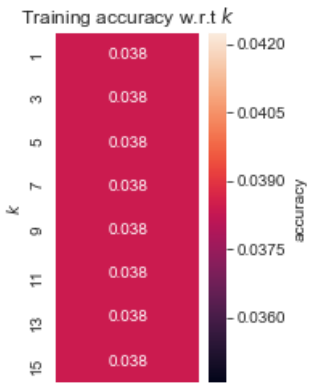

Figure 4: Training Accuracy for KNN(letter-recognition)

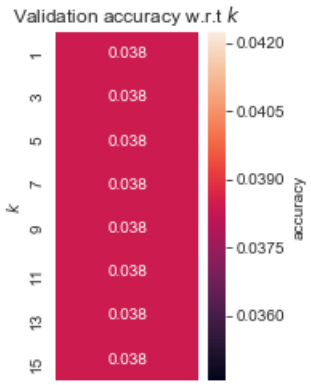

Figure 3: Validation Accuracy for KNN(letter-recognition)

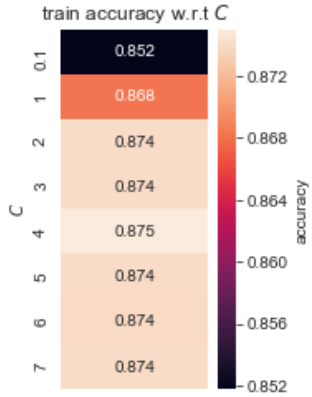

Figure 5: Training Accuracy, linear SVM (Adult)

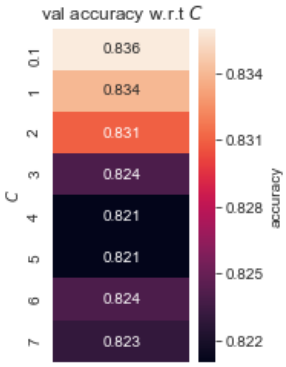

Figure 6: Validation Accuracy, linear SVM(Adult)

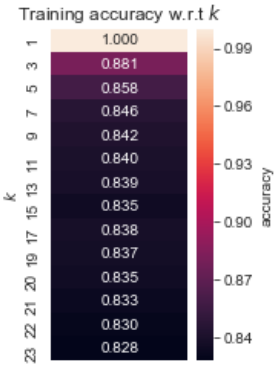

Figure 7: Training Accuracy for KNN(Adult)

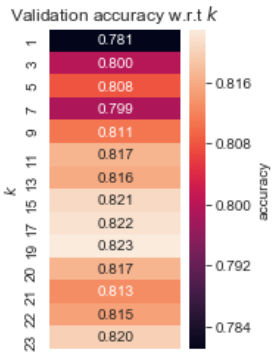

Figure 8:Validation Accuracy for KNN(Adult)

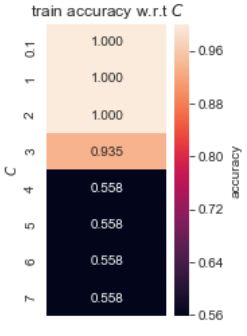

Figure 10: Training Accuracy, linearSVM(Credit)

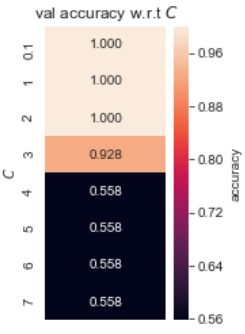

Figure 9: Validation Accuracy, linearSVM(Credit)

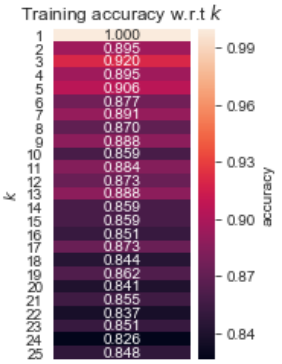

Figure 11: Training Accuracy for KNN(Credit)

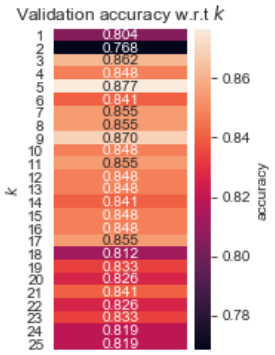

Figure 12: Validation Accuracy for KNN(Credit)

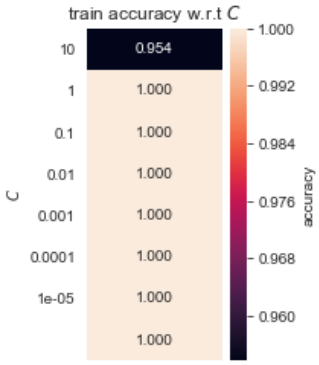

Figure 13: Training Accuracy, linearSVM(Car)

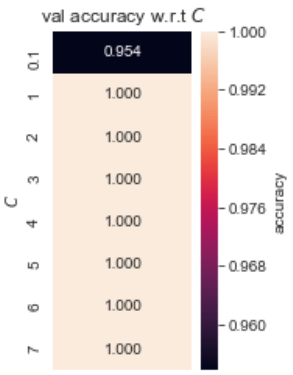

Figure 14: Validation Accuracy, linearSVM(Car)

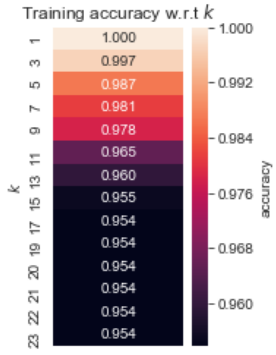

Figure 16: Training Accuracy for KNN(Car)

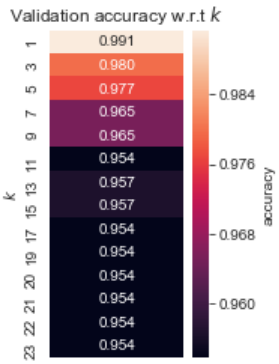

Figure 15: Validation Accuracy for KNN(Car)

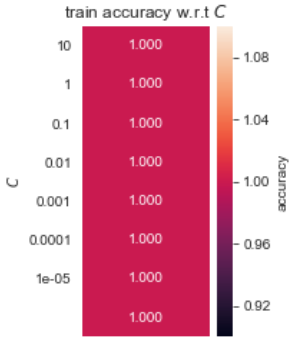

Figure 18: Training Accuracy, linearSVM(Bridges)

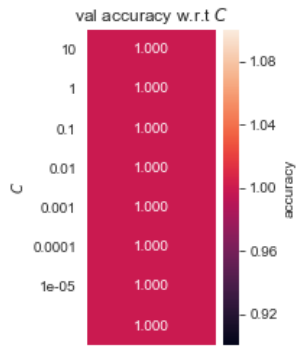

Figure 17: Validation Accuracy, linearSVM(Bridges)

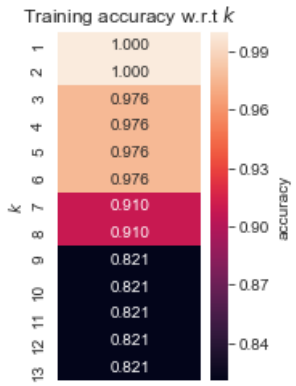

Figure 20: Training Accuracy for KNN(Bridges)

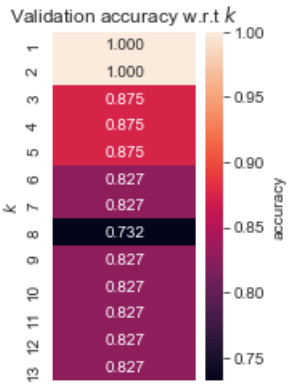

Figure 19: Validation Accuracy for KNN(Bridges)

3.2 Heatmap Analysis

These first heatmaps were done using linear SVM kernel with varied parameters for C, depending on the data. Similarly, the second heatmaps were done using simple KNN method with a K list that varied depending on the dataset in question. In general, both types of heatmaps were done separately of the analysis of the classifiers to reduce error and to further validate the KNN and SVM methods I used.

These heatmaps are primarily targeted as a general overview of the 80/20 datasets. Each dataset shows relatively good results that reflect the research done in the original paper. Though, it is clear that some of the datasets did not have a good list of parameters. Such is the case for the bridges dataset, as all values of k bring about the same accuracy. I did not do an extensive list of K due to the lacking capabilities of my computer. The heatmaps shown took a very long time to load, even with the relatively small parameter constraints. Even so, the majority of the heatmaps reflect results of 80% or above, which coincides with the results in the original paper. A further detailed analysis of the classifiers will ensue.

3.3 Models

For convenience sake, all models will be shown on the next page. All models were tested for accuracy, training time, best hyperparameter, f1 accuracy, precision and recall. While each classifier was tested through multiple ways, the accuracies for the datasets are recorded in the tables. This is because this is the most important value to compare against the original paper's research. The other tests were simply done to validate the accuracy and functionality of the models. KNN was done separately on bridges since this dataset focused on a design prediction, which included predicting 5 attributes from the given set. Still, the KNN heatmap reflect that classifier, it just doesn't fit with the classifier in the bulk of the classifiers and I wanted to be consistent.

3.4 Model Analysis

	ALL DATA
XGBOOST	0.967
BDT	0.9638
RF	0.9588
SVMRBF	0.952933333
SVMLIN	0.9466
KNN	0.929416667
NN	0.923066667

Table 1: Average accuracy across all data. Ranked.

The above table depicts the average accuracies across all the data. The table is also ranked in order. My results in terms of ranking are similar to that of Caruana and Niculescu-Mizil. The boosted decision tree and random forests seem to perform better on average while KNN and support vector machines fall a little behind. The main difference in my results is that NN comes in last for me while a middle rank in the original paper. This could be due to the lack of similar data as mentioned earlier. The bridge data differed from the others due to its lack of a normal attribute to predict. Rather than yes or no, the data allowed for 5 attributes to be predicted.

20/80	LETTER RECOGNITION	CENSUS	CAR EVALUATION	CREDIT APPROVAL	BRIDGES	AVERAGE
KNN	0.993	0.819	1	0.913	-	0.93125
RANDOMFOREST	0.991	0.845	1	1	1	0.9672
SVMLIN	0.962	0.841	1	1	0.909	0.9424
SVMRBF	0.995	0.837	1	0.992	0.954	0.9556
DECISIONTREE	0.983	0.832	1	1	1	0.963
NEURALNET	0.987	0.826	0.997	1	0.772	0.9164
XGBOOST	0.992	0.844	1	1	1	0.9672

Table2: 20/80 split of the data. Accuracies for all classifiers

80/20	LETTER RECOGNITION	CENSUS	CAR EVALUATION	CREDIT APPROVAL	BRIDGES	AVERAGE
KNN	0.991	0.816	0.993	0.902	-	0.9255
RANDOMFOREST	0.989	0.835	1	1	0.941	0.953
SVMLIN	0.961	0.832	1	1	0.965	0.9516
SVMRBF	0.986	0.838	1	0.98	0.941	0.949
DECISIONTREE	0.982	0.835	1	1	1	0.9634
NEURALNET	0.962	0.839	0.963	0.804	0.977	0.909
XGBOOST	0.992	0.844	1	1	0.988	0.9648

Table 3: 80/20 split of the data. Accuracies for all classifiers

50/50	LETTER RECOGNITION	CENSUS	CAR EVALUATION	CREDIT APPROVAL	BRIDGES	AVERAGE
KNN	0.994	0.825	1	0.907	-	0.9315
RANDOMFOREST	0.992	0.845	1	1	0.944	0.9562
SVMLIN	0.962	0.841	1	1	0.926	0.9458
SVMRBF	0.993	0.846	1	0.988	0.944	0.9542
DECISIONTREE	0.981	0.844	1	1	1	0.965
NEURALNET	0.988	0.843	0.971	0.991	0.926	0.9438
XGBOOST	0.995	0.85	1	1	1	0.969

Table 4: 50/50 split of the data. Accuracies for all classifiers

4. Conclusion / Additional Analysis

A few of the models produced higher results than usual, which may have corrupted the average accuracies and overall ranking. This is especially evident in the Car Evaluation data. Though, I believe this was due to the lack of difficulty for the data. The attributes associated with the classification were very relative and intuitive. Also, XGBoost was not directly tested in the original paper, but provided a nice additional model to compare the models to.

Despite this, the models provided similar ranking and results to the original paper. This is especially evident in the datasets that overlapped with the original research. Letter recognition yielded very similar results to that of the "LTR.P2" dataset indicating that the models functioned adequately. The Census data results also correlated with the results "ADULT" in the original by staying within the 80%-90% range of accuracies. Though the random forest in the original performed significantly better with a ~95% accuracy while mine yielded around an 85%. Still my random forest model performed better on average, resulting in similar conclusions.

Any of the difference are likely related to the weakness in my local machine and the choice of datasets. The latter has much more of an effect on the models. There were also variations of parameters in a few of my models which may have also lead to different results. I tried to keep as many of the parameters the same as in the paper while also noting the limitations of my local machine. Aside from the car evaluation dataset, there was a widespread variety of accuracies between datasets and splits yielded similar conclusions to that of the original research.

5. Bonus Points

I used 7 classifiers and 5 different datasets, of which 3 were completely unique. I also computed many heatmaps and went above the word count for a single person.

References

[1] Caruana, Rich, and Alexandru Niculescu-Mizil. "An Empirical Comparison of Supervised Learning Algorithms." *Proceedings of the 23rd International Conference on Machine Learning - ICML 06*, 2006, doi:10.1145/1143844.1143865.

Word Count: 2071 (doesn't include 5. And onware)