An Empirical Comparison of Supervised Learning Algorithms

Manuel Cortez m6cortez@ucsd.edu

University of California, San Diego

Abstract

To further enrich my understanding of supervised learning algorithms I have learned in my supervised learning class, I will attempt to implement common supervised learning algorithms to three datasets. I will implement Support Vector Machines (SVM), K-Nearest Neighbors (KNN), and Decision Trees. Based on the performances of these algorithms to my datasets of study, I will evaluate each learning method. Supervised learning algorithms became prominent in the 1990s and since then, we continue to utilize these algorithms to understand big data. Since then, we have developed more tools and algorithms that ease the process.

1. Introduction

My empirical comparison of Supervised Learning Algorithms was inspired from Rick Caruana and Alexandru Niculescu-Mizil in their research paper, "An Empirical Comparison of Supervised Learning Algorithms" in which they compare six different classifiers on several datasets. Like the two researchers, my goal is to learn from using the algorithms and see in what condition a supervised machine learning algorithm thrives and when it does not. I plan on carrying out their experiment on a much smaller scale. Instead of performing multiple supervised algorithms, I will focus on three Supervised Learning algorithms and test their performance on three different datasets.

The three classifier that I will implement on my experiment are SVM (Linear), KNN, and decision trees. To star off, SVM is a classification and algorithm used to find the best decision boundary to distinguish two classes. Similarly, KNN is a classification algorithm that measures the distances between data points. We initialize K and calculate the distances between points. Usually, KNN functions with the idea in mind that similarly classified data points are near one another. Lastly, decision trees are another supervised algorithm that classifies our data into a 'tree-like' structure with nodes. We place the best attribute of the data at the root of the tree and then split the training set into subsets making up the branches of the tree. It is important to note that in most cases data consists of multi-class labels. When that is the case, usually we can try to merge the labels into two classes. KNN, SVM, and decision trees are algorithms in which data must be adjusted to a two-class setting. In two of my selected datasets, I attempted to merge labels into two classes.

2. Methods & Datasets

The datasets that I decided to implement the supervised algorithms are the iris dataset, a Pokémon dataset, and the abalone dataset. The iris dataset is a common dataset used when first learning supervised algorithms. This dataset contains 3 different classes of plants and has several observations containing the length and width of the sepal and petal component of a plant. The small number of features and observations in the iris dataset makes the dataset convenient for first-time learners because it is not computationally demanding, the dataset is easy to interpret, and there is a clear distinction between the three classes. Next, I decided to use a Pokémon dataset to attract more attention to my experiments. Pokémon is widely known for their video games. My Pokémon dataset contains 800 Pokémon (up to generation 6). Several attributes are measured, such as attack, defense, and speed. Lastly, the abalone (shellfish, a genus of gastropods) dataset predicts the age of abalone by performing examinations when the abalone is cut. Counting the number of rings, we can determine the age of the abalone. However, this task can be tedious, and the makers of the dataset were curious to find out if other attributes such as the length, diameter, and weight of the abalone is indicative of its age.

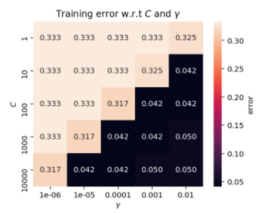
To implement KNN, SVM, and decision tress I had to do some work on my datasets beforehand. First off, I decided that I did not wanted to work with multi-class labels as that will add on to the computational capacity and make it more difficult to implement the algorithms. I decided to merge my labels into two classes. In the Iris dataset, my goal is to correctly predict (Setosa, Virginica, or Versicolor) each prediction. Instead, my goal is to predict whether an observation is Virginica or Setosa/Versicolor. Similarly, in the Pokémon dataset my goal is to predict the type of the Pokémon (e.g. Fire, Water, Grass). However, I found that there were more than 10 different types of Pokémon. Thus, I made two class groups. Each class had an equal amount of differentiating types of Pokémon. Lastly, the abalone dataset aims to predict the age of an abalone based on attributes. We are given the age, via the number of rings, to see if we can use those attributes to correctly predict the age of an abalone. This dataset is also multiclass. It contains abalones from a wide-ranging number of rings. To merge this dataset into two classes I made one class to consist of abalones with more than 10 rings and the other class to consist of abalones with less than 10 rings.

3. Experiments

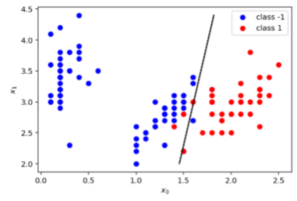
Now that I've established the datasets I will be using and the three supervised algorithms, I will start explaining how I will carry out my experiment and display the results obtained. First off, I will implement all three supervised algorithms on all three datasets. I will perform two types of splits on my dataset and perform three trials for each classifier with each split. More specifically, the two types of splits I will be performing are an 80/20 and a 20/80 split. For each dataset, I will use 80% of my observations for my training data and 20% for my testing data. Likewise, then I will use 20% of my observations for my training data set and 80% for my testing dataset. The goal by doing these two different splits is to show that accuracy results are much higher when our testing data is much greater than our training data.

Iris:

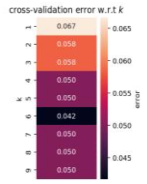
	Model	**Split**	**Train**	**Test**	**Validation**
Trial 1	SVM	80/20 <u>20/80</u>	0.05 <u>0.0</u>	0.06 0.09	N/A
Trial 2	KNN	80/20 <u>20/80</u>	0.0416 <u>0.033</u>	0.133 <u>0.075</u>	0.02 <u>0.06</u>
Trial 3	Decision Tree	80/20 <u>20/80</u>	0.033 <u>0.0</u>	0.066 <u>0.050</u>	N/A



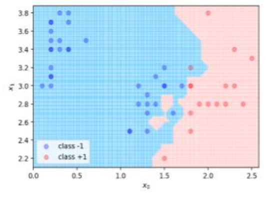
```
Best parameter C*=10, best parameter gamma*=0.01
Test error: 0.06666666666666665
Train error: 0.05000000000000044
Decision boundary: 268.796x0+-40.663x1+-310.000=0
```



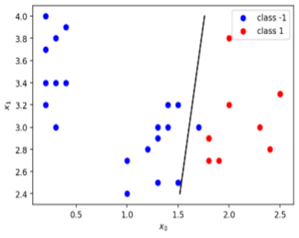
```
Training error: 0.05
```



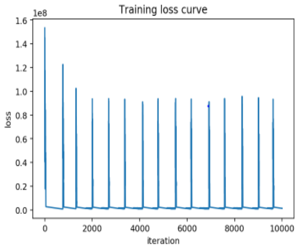
```
Best number of nearest neighbors (k): 6
k=1
```

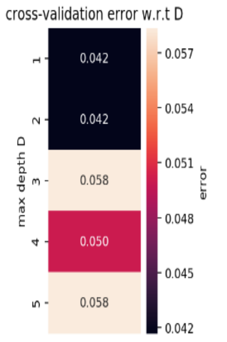


```
Validation error: 0.02
```



```
Test error: 0.03333333333333333
```



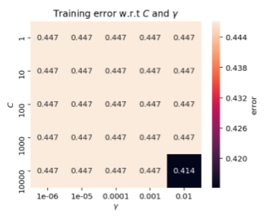


```
Best max depth D: 1
Train error: 0.033333333333333326
Test error: 0.06666666666666665
```

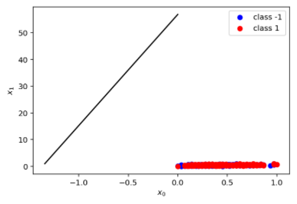
Figure 1: In this figure each heatmap represents the start of a new model. We start off with SVN, then KNN, lastly decision tree. SVM and KNN are accompanied by plots showing the distribution of the Iris dataset. Dataset has 150 observations and 5 columns.

Pokémon:

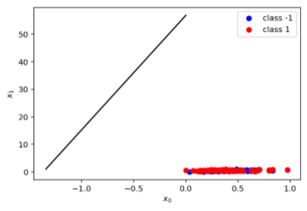
	Model	Split		Train		Test		Validation	
Trial 1	SVM	80/20	20/80	0.412	0.396	0.368	0.354	N/A	
Trial 2	KNN	80/20	20/80	0.13	0.041	0.472	0.462	0.138	0.041
Trial 3	Decision Tree	80/20	20/80	0.374	0.284	0.486	0.422	N/A	



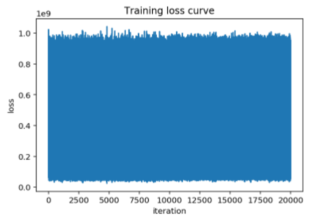
```
Best parameter C*=10000, best parameter gamma*=0.001
Test error: 0.36805555555555556
Train error: 0.41421143847486996
Decision boundary: -209.962x0+5.048x1+-287.000=0
```

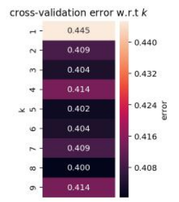


```
Training error: 0.44714038128249567
```

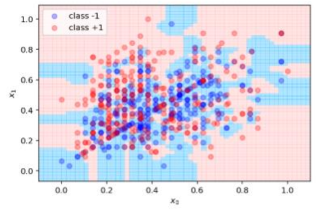


```
Test error: 0.4444444444444444
```

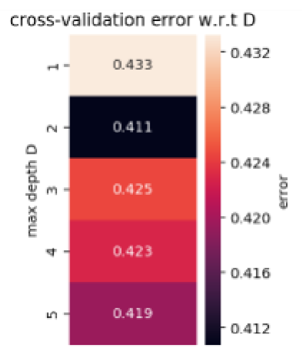




```
Best number of nearest neighbors (k): 8
k=1
```



```
Validation error: 0.1386481802426343
```

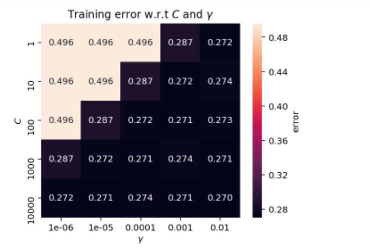


```
Best max depth D: 2
Train error: 0.3743500866551126
Test error: 0.48611111111111116
```

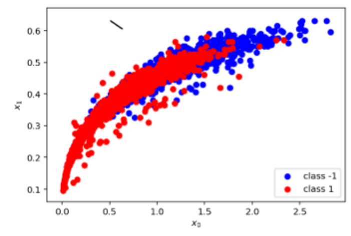
Figure 2: In this figure each heatmap represents the start of a new model. We start off with SVN, then KNN, lastly decision tree. SVM and KNN are accompanied by plots showing the distribution of the Pokémon dataset. Dataset has 721 observations and 7 features.

Abalone:

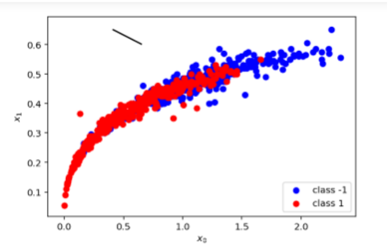
	Model	**Split**		**Train**		**Test**		**Validation**	
Trial 1	SVM	80/20	20/80	0.270	0.264	0.258	0.269	N/A	
Trial 2	KNN	80/20	20/80	0.012	0.0047	0.365	0.347	0.012	0.0047
Trial 3	Decision Tree	80/20	20/80	0.262	0.253	0.275	0.271	N/A	



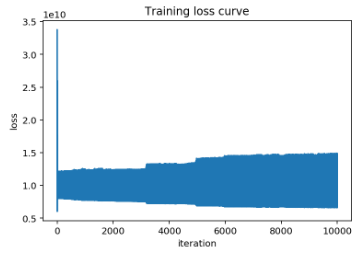
```
Best parameter C*=10000, best parameter gamma*=0.01
Test error: 0.2586826347305389
Train error: 0.27027835977252324
Decision boundary: -855.520x0+-4175.147x1+3068.000=0
```

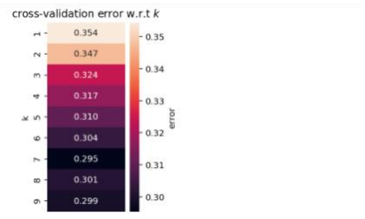


```
Training error: 0.3436096976953008
```

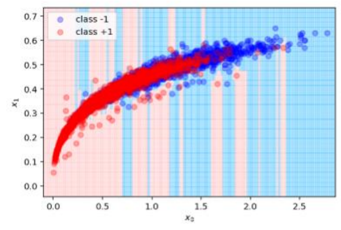


```
Test error: 0.3520958083832335
```

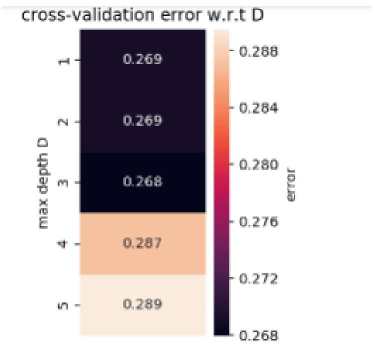




```
Best number of nearest neighbors (k): 7
k=1
```



```
Validation error: 0.012271774917689314
```



```
Best max depth D: 3
Train error: 0.2535885167464115
Test error: 0.27125748502994007
```

Figure 3: In this figure each heatmap represents the start of a new model. We start off with SVN, then KNN, lastly decision tree. SVM and KNN are accompanied by plots showing the distribution of the Pokémon dataset. Dataset has 4176 observations and 8 columns.

4. Discussion and Conclusion

Now that we have analyzed our dataset using all three algorithm models, we can begin to compare the performances of each model. Overall, KNN seems to have performed the best. This makes sense. The way that KNN works is that it checks the distances between all points. The larger K is, the more accurate the results will be as KNN will continue to be updated with the best parameters. Up next, we have linear SVM. Linear SVM performed decently well. This model attempts to use each observation to find a decision boundary to separate the two types of classes. Lastly, we have decision tree. Our decision tree results were fair, but not great. This could be because even though we separated our data into two classes, our observations varied a lot from one another. Overall, linear SVM and KNN performed the best as we can see from the figures above that it did the best job classifying each observation. In the report, "An Empirical Comparison of Supervised Learning Algorithms" by rich Caruana and Alexandru Niculescu-Mizil the results they obtained are similar. They used more models and settings when using their classifiers on the dataset than I, but SVM and KNN were one of their best performing supervised algorithms.

Things we could have done to better our results are choosing a better way to merge our multi-class labels in the Pokémon and the abalone data set into two classes. As we can see from both the Pokémon and the abalone dataset, we had too many classes that merging them into two classes would not be very effective as our observations differentiate quite a bit. For example, our plots from the Iris dataset only had three classes and merging them into two did not really affect our results. As we can see, we were able to obtain high classification results. Lastly, the computer I was working on lacked the hardware to better process the data. I was able to quickly obtain results for the Iris dataset, but not the other two. Overall, I believe that our empirical comparison was a success as we were able to compare the performance of each model we used. From doing comparison experiments, we learn how to implement these algorithms successfully and most importantly, we learn how to interpret our results.

To see more information and plots on my results click this Github link:

https://github.com/m6cortez/Project1

5. References

Abalone. (2019, October 8). Retrieved from https://simple.wikipedia.org/wiki/Abalone

Caruana, R., & Niculescu-Mizil , A. (n.d.). An Empirical Comparison of Supervised Learning Algorithms. Retrieved from https://www.cs.cornell.edu/~caruana/ctp/ct.papers/caruana.icml06.pdf

Dua, D. and Graff, C. (2019). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

Irisflowerdataset.(2020,February13).Retrievedfromhttps://en.wikipedia.org/wiki/Iris_flower_data_set

Pokémon. (2020, March 1). Retrieved from https://en.wikipedia.org/wiki/Pokémon#Concept