An Empirical Comparison of SVM, KNN, and Decision Trees

Yu Xu

Abstract

This paper presents an empirical comparison between three supervised learning methods: Linear SVM, K-nearest neighbors, and decision tree on datasets with different characteristics.

Introduction

Nowadays, learning algorithms are applied in many distinct domains. The importance of having a basic knowledge and the ability of practical application of learning algorithms can be seen. This paper presents results of an empirical comparison between Linear SVM, K-nearest neighbors, and decision tree, which are three of the most popularly used supervised learning algorithms. I examine the performance of those algorithms using 2 metrics -- accuracy and f1-score -- on three different datasets named in short: adult, NPHA, and car. For each algorithm I perform hyper-parameter tuning and cross validation to ensure the model's performance is consistent across different subsets of the dataset.

Methodology

Learning Algorithms

SVM: linear SVM radial with width {0.01, 0.1, 1, 10}.

K-nearest neighbor (KNN): 10 values of K ranging from K = 1 to 10.

Decision Tree (DT): They are easy to interpret and can handle categorical data effectively.

Data Pre-processing and Cleaning

Before separating the data into features and target, I do basic data pre-processing including checking for missing values, handling categorical data, and scaling the data (for KNN).

Datasets

I chose 3 datasets from the UC Irvine Machine Learning Repository.

National Poll on Healthy Aging (NPHA): all of 14 features are categorically represented by numeric orderings. Number of instances: 714.

Adult: 14 features: a mix of binary, categorical and integer types. Number of instances: 48842.

Car: 6 features: all non-numeric categorical features. Number of instances: 1728.

Performance Metrics

In order to find the best representative performance for an algorithm, I split the dataset into training and testing sets in a 80/20 split before model evaluation.
I use accuracy (ACC) and f1-score (F1), the weighted average of Precision and Recall to compare the performance of the different algorithms.

Algorithm-Dataset	ACC	F1
SVM-NPHA	0.48	0.31
SVM-car	0.70	0.61
SVM-adult	**0.79**	**0.81**
KNN-NPHA	0.50	0.46
KNN-car	**0.94**	**0.93**
KNN-adult	0.83	0.83
DT-NPHA	0.32	0.32
DT-car	**0.97**	**0.97**
DT-adult	0.82	0.82

Table 1: Performance Metrics of Different Algorithms on Various Datasets

Experiment

In order to evaluate the performance of linear SVM, KNN, and Decision Tree separately on the three datasets, I follow the procedures that consist of hyper-parameter tuning and cross validation to ensure the model's performance is consistent across different subsets of the dataset.
1. set up partitions of training and testing sets partitions = [(0.2, 0.8), (0.5, 0.5), (0.8, 0.2)], rounds = 3
2. set up C_values = [0.01, 0.1, 1, 10] in SVM; Range of n_neighbors to test from 1 to 10; max_depth_values = [3, 5, 10] in decision tree.
3. iterate through each train-test partition for n rounds. For each round, train a model on the training set and evaluate it on both the training set (to get the training score) and the testing set (using cross-validation).
4. keeps track of the best training and testing scores for each partition
5. printing results and plotting.

```
Partition 20.0% Train / 80.0% Test
Average Training Scores: [0.5704225352112676, 0.6690140845070424, 0.847
4178403755869]
Average Testing Scores: [0.48948893974065594, 0.4802542588354945, 0.452
88075260615307]
Best Training Score: 0.8474178403755869 at max_depth = 10
Best Testing Score: 0.48948893974065594 at max_depth = 3
```

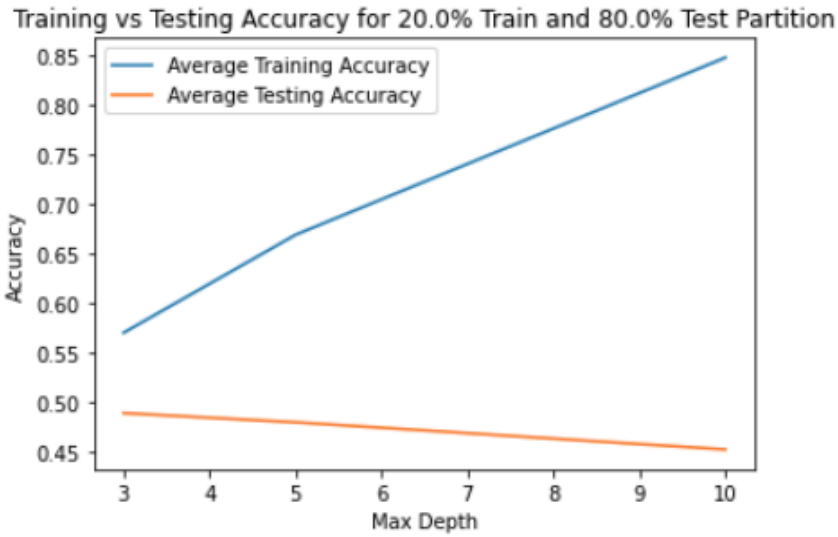

Figure 1: snippet of visualization and best score tracking for decision tree evaluation on NPHA dataset

Through *Figure 1*, we can see that the decision tree has been performed in a 3-fold cross validation manner on the NPHA dataset. As the average training accuracy increases, average testing accuracy decreases. Best training and testing scores and their corresponding hyperparameters are respectively recorded. Similar method to extract training and testing scores is applied on the evaluation for Linear SVM and KNN as well.

The below table shows the training and testing scores for each algorithm on the three different datasets in different partitions. Since the test scores are cross validated, cross validation scores will be the same as test scores.

20/80 split:

	NPHA-train	NPHA-test	car-train	car-test	adult-train	adult-test
Linear SVM	0.54	0.51	0.70	0.73	**0.79**	**0.79**
KNN	0.97	0.48	**1.00**	**0.92**	1.00	0.83
DT	0.84	0.48	**1.00**	**0.95**	0.89	0.85

50/50 split:

	NPHA-train	NPHA-test	car-train	car-test	adult-train	adult-test
Linear SVM	0.55	0.50	0.72	0.73	**0.80**	**0.79**
KNN	0.96	0.48	**1.00**	**0.87**	0.99	0.82
DT	0.85	0.45	**0.99**	**0.93**	0.87	0.84

80/20 split:

	NPHA-train	NPHA-test	car-train	car-test	adult-train	adult-test
Linear SVM	0.53	0.48	0.73	0.68	**0.81**	**0.79**
KNN	0.94	0.47	**1.00**	**0.83**	0.99	0.82
DT	0.74	0.47	**0.99**	**0.92**	0.87	0.85

Table 2: training and testing scores for each algorithm in different partitions

Conclusion

In this study I have practiced using three different supervised learning algorithms on three different datasets provided by UCI Machine Learning Repository in order to examine the characteristics and performance of those algorithms.

Using accuracy and f1 score as the performance metrics for our first brief practice of evaluating linear SVM, K-nearest neighbors, and Decision Tree, I compared the values of accuracy and f1 scores for each algorithm applied on the same dataset. I found that statistically speaking, SVM works best on the adult dataset, with both significantly high scores for accuracy and f1 compared to it does on other datasets. KNN works best on the car dataset, with both accuracy and fl score exceeding 0.9. Lastly, Decision Tree works best on the same dataset as KNN does. Surprisingly, all three algorithms work poorly on the NPHA dataset: both accuracy and f1 score for each algorithm do not exceed 0.5, which means it corrects no more than half of the time (similarly to random guessing).

The following experiment consists of hyper-parameter tuning and cross validation somewhat consolidates the results presented in our first practice. SVM works best on the adult dataset. KNN works best on the car dataset, as well as obtaining high training scores for all datasets. I also found that despite good performance in training tasks, KNN continues to perform poorly in generating to testing tasks. Decision Tree also works best on car dataset, obtaining significantly high training and testing accuracy.

Based on my understanding of characteristics of the three learning algorithms, I guess that one of the reasons why KNN gets significantly high training scores yet inconsistent testing scores is its nonparametric learning method. Because it could be overfitting to training data by failing to make any assumptions about the distribution of data but to memorize it. As well as my explanation for the scenario in which all three algorithms work poorly on the NPHA dataset in testing is based on both the nature of the dataset and of the algorithms. With all features in NPHA being categorical, models can become complex. In addition, KNN might not work well because it relies on distance metrics to find the nearest neighbors, whereas categorical and ordinal features have very limited distance or distribution. There could also be my limitation in data understanding and knowledge in the domain that contributes to the mistakes in preprocessing and modeling decisions that cause the poor testing performance for all the algorithms on the NPHA dataset.

Reference

All the analysis in this report is supported by Final_Project_code.ipynb located in jupyterhub under my UCSD account and imported datasets from UC Irvine Machine Learning Repository. Please read the attached PDF of the complete code in my submission to address my progress of analysis.

Rich Caruana and Alexandru Niculescu-Mizil. 2006. An empirical comparison of supervised learning algorithms. In Proceedings of the 23rd international conference on Machine learning (ICML '06). Association for Computing Machinery, New York, NY, USA, 161–168. https://doi.org/10.1145/1143844.1143865