Supervised Machine Learning Algorithms Comparison

Abstract

Machine learning has become very common relatively recently, with many new algorithms created over the past several years. While each algorithm has been shown to work, their effectiveness relative to each other has not been completely decided. In this study, I test and discuss the effectiveness of three models: SVM, KNN, and logistic regression, and compare them with each other.

1. Introduction

While there are many different supervised machine learning algorithms today, this paper will focus on and compare three of them: support vector machine, decision tree, and k-nearest neighbors algorithm. It is important to note that the effectiveness of these algorithms is not absolute; that is, they can vary depending on the dataset, parameters, etc. Therefore in order to make this study more comprehensive, I will be using three datasets with to measure their effectiveness on each one. For the purpose of this study, the datasets used will all have numerical attributes and binary classifications. Additionally, I will measure their effectiveness with different partitions of the data. I will also be testing different values for a hyperparameter for each model, and train the model based on the best hyperparameter measured for that iteration.

2. Method

I compared three different algorithms using three different datasets each with three different partitions, shuffling the data each time a model was trained. The partitions used were 20% training/80% testing, 50% training/50% testing, and 80% training/20% testing. Before training and testing, I used a scikit-learn's grid search algorithm to get the best hyperparameters for each model, using 3-fold cross-validation. To measure model performance, I used the test accuracy of each model to compare against one another. Each partition was trained and measured three times (shuffling each time) and the results were averaged together in order to get the accuracy of each model's partition. Additionally, I tested the models with normalization techniques and then compared accuracy with the most effective normalization for each model.

2.1 Learning Algorithms

SVM: I used a SVM from the scikit-learn library called SVC. I opted to use a linear kernel, and the hyperparameter I tuned was the best inverse regularization factor (C-value), with the values tested varying by factors of 10 and ranging from 10^-6 to 10^-1.

Decision Tree: I implemented a decision tree with the DecisionTreeClassifier from the scikit library, using the entropy criterion. The hyperparameter I tuned was the max depth, testing integer values ranging from 1 to 20.

KNN: I implemented a KNN using the KNeighborsClassifier from the scikit library. The hyperparameter tuned was the number of nearest neighbors K, with integer values ranging from 1 to 50.

2.2 Performance Metrics

To measure performance for my trained models I used their F1 score on predictions made against their test set after fitting them to their training set. F1 score is the weighted average of the model's precision and recall in order to account for both false positives and false negatives. Precision is the average of the proportion of true positives predicted to the total number of positives predicted, while recall is the proportion of true positives predicted to the total number of positives in the dataset. The equations are:

TP - True Positives, FP - False Positives, FN - False Negatives

Precision = TP / (TP + FP)

Recall = TP / (TP + FN)

F1 Score = 2 * (Precision * Recall) / (Precision + Recall)

There will be an F1 value for each partition on each dataset for each model, totaling to 27 values. Each of these values is an average of three measurements made with the data being shuffled, hyperparameters being re-calibrated, and the model being refitted before each measurement.

2.3 Datasets

The first dataset I used was data on women's method of contraceptive use. It had 1473 instances and 9 attributes including age, education, and number of children. The dataset had three classifications on contraception methods: none, long-term methods, and short-term methods. To convert it to a binary classification I grouped short-term and long-term contraception together, changing the classifications to either using or not using any contraception.

The next dataset was on the objectivity of sports articles. Sports articles were read and consequently labeled as either objective or subjective. There were 1000 instances with originally 59 attributes including word count and various semantic phenomena. For this dataset I removed two attributes: the textID and the URL of the article. This left me with 57 attributes for the dataset.

The third dataset is one for determining whether a molecule is a musk molecule or not. It originally had a total of 6598 instances and 166 attributes describing the shape of the molecule, and 2 attributes labeling the "symbolic name" of the molecule. I removed those two attributes since they were not to be used for predictions, leaving me with 166 attributes. I also shuffled the data and sampled 2000 instances to keep it more in line with the other two datasets.

Dataset	Number of Instances	Number of Attributes	Classification
Contraceptive Method Choice	1473	9	Using contraception vs. not using contraception
Sports Articles for Objectivity	1000	57	Objective vs. subjective
Musk Molecule	2000	166	Musk vs. non-musk

3. Experiment

In terms of normalization, I surprisingly found that using any normalization technique for SVM's gave worse results than no normalization. Additionally, I found that normalization techniques either had no effect or gave marginally lower results for KNN's and Decision Trees. In the end, I compared them without any normalization. This could be due either the datasets used or possibly the way they were shuffled. After optimizing for normalization, testing the models, and saving their F1 scores, the results are on the table below.

Dataset	Partition (Training/Testing)	Model	F1 Score
Contraceptive	20%/80%	SVM	0.7327
Contraceptive	20%/80%	DT	0.7497
Contraceptive	20%/80%	KNN	0.7265
Contraceptive	50%/50%	SVM	0.7450
Contraceptive	50%/50%	DT	0.7649

Contraceptive	50%/50%	KNN	0.7365
Contraceptive	80%/20%	SVM	0.7411
Contraceptive	80%/20%	DT	0.7817
Contraceptive	80%/20%	KNN	0.7744
Sports	20%/80%	SVM	0.8628
Sports	20%/80%	DT	0.7835
Sports	20%/80%	KNN	0.7929
Sports	50%/50%	SVM	0.8692
Sports	50%/50%	DT	0.8326
Sports	50%/50%	KNN	0.8349
Sports	80%/20%	SVM	0.8751
Sports	80%/20%	DT	0.8538
Sports	80%/20%	KNN	0.8431
Musk	20%/80%	SVM	0.7117
Musk	20%/80%	DT	0.3995
Musk	20%/80%	KNN	0.7085
Musk	50%/50%	SVM	0.7533
Musk	50%/50%	DT	0.6916
Musk	50%/50%	KNN	0.7907
Musk	80%/20%	SVM	0.7916
Musk	80%/20%	DT	0.7199
Musk	80%/20%	KNN	0.8209

I also added the values to an excel sheet and computed various averages as shown in the figure below.

					Averages each dataset	Average datasets 1 & 2
50%/50% & 80%/20% avg				**0.795883**		
		20%/80%	**50%/50%**	**80%/20%**		**0.804316667**
	Dataset 1	0.7327	0.745	0.7411	**0.7396**	
SVM	**Dataset 2**	0.8628	0.8692	0.8751	**0.869033333**	
	Dataset 3	0.7117	0.7533	0.7916	**0.7522**	
partition averages		**0.769067**	**0.789167**	**0.8026**		
50%/50% & 80%/20% avg				**0.774083**		
		20%/80%	**50%/50%**	**80%/20%**		**0.794366667**
	Dataset 1	0.7497	0.7649	0.7817	**0.765433333**	
DT	**Dataset 2**	0.7835	0.8326	0.8538	**0.8233**	
	Dataset 3	0.3995	0.6916	0.7199	**0.603666667**	
partition averages		**0.644233**	**0.763033**	**0.785133**		
50%/50% & 80%/20% avg				**0.800083**		
		20%/80%	**50%/50%**	**80%/20%**		**0.784716667**
	Dataset 1	0.7265	0.7365	0.7744	**0.7458**	
KNN	**Dataset 2**	0.7929	0.8349	0.8431	**0.823633333**	
	Dataset 3	0.7085	0.7907	0.8209	**0.773366667**	
partition averages		**0.742633**	**0.787367**	**0.8128**		

Examining the data, it is clear that the easiest dataset for every algorithm is the sports dataset. This could be because the models are more effective with a feature list of around 60, but more likely because the data is merely more predictable than the other two.

The SVM and KNN performed similarly well, with the SVM having an overall average F1 score of 0.7870 and KNN having a score of 0.7809, with the DT trailing behind with a score of 0.7308. Dissecting their performance further, shows that the KNN has a somewhat better performance for the 80%/20% split, with an average score of .8128, SVM with 0.8026, and DT with 0.7851. It's important to also note the disparity in the DT's total average and its average when omitting the 20%/80% partition, which brings it more in line with the other two models.

The poorest single performance on a dataset belongs to the Decision Tree on the Musk dataset. Considering the Musk data had by far the highest number of features, it seems that a DT needs relatively larger amount of data when fitting to a dataset with a large number of attributes.

In terms of performance per dataset, the best performance for dataset 1 (contraceptives w/ 9 attributes) was the DT. The best performance for dataset 2 (sports w/ 57 attributes) was the SVM. The best performance for dataset 3 (musk w/ 166 attributes) was KNN. This could suggest that DT works best with a small number of attributes, SVM works best with a medium number of attributes, and KNN works best with a larger number of attributes.

4. Conclusion

Machine Learning Algorithms are powerful tools that anyone has access to. The importance of understanding the effectiveness of the various models is significant. In this study we found that SVM's and KNN's having a similar overall effectiveness with both beating the Decision Tree. The Decision Tree seems to do well enough, especially for datasets with fewer attributes, but lags behind when analyzing complex data with a relatively small amount of training. While this study is limited in scope and the results could be due to various factors such as the implementation of each model and the nature of the datasets used, my analysis suggests that the algorithms shine in different areas, with the DT being the better choice for more simple data and SVM's and KNN's working better with more complex data.

5. References

1. *UCI Machine Learning Repository: Contraceptive Method Choice Data Set*, archive.ics.uci.edu/ml/datasets/Contraceptive+Method+Choice.
2. *UCI Machine Learning Repository: Musk (Version 2) Data Set*, archive.ics.uci.edu/ml/datasets/Musk+%28Version+2%29.
3. *UCI Machine Learning Repository: Sports Articles for Objectivity Analysis Data Set*, archive.ics.uci.edu/ml/datasets/Sports+articles+for+objectivity+analysis.
4. "Accuracy, Precision, Recall & F1 Score: Interpretation of Performance Measures." *Exsilio Blog*, 11 Nov. 2016, blog.exsilio.com/all/accuracy-precision-recall-f1-score-interpretation-of-performance-measures/.