Machine Learning Final Project

Andre Mendes (amd871)

December 7, 2015

1 Introduction

The objective of this project is to use the machine learning methods learned in the classroom to perform a classification using a dataset provided by the Instructor. The data consists of a training set, a test set and labels for the training set. The labels are binary values and the goal is to use the training set and labels to train a classifier to classify the test set. Therefore, this is a supervised learning problem with a binary classification task.

This project uses a popular machine learning tool called Weka [3] that provides many options to perform filtering, classification, clustering and many other useful data mining operations.

The steps in this project are shown in Figure 1 and they are divided as follow: In the first step, the procedure for preparing the data to be used in the Weka are explained. Then the training data is modified in a pre-processing phase to be used in the classification. After that, the classification algorithms and their parameters are selected. It is then performed parameter selection and optimization using grid search in cross validation. The optimized parameters are then used to train the classifier again in cross validation and in the final step, the classifier is used in the test set.

2 Preparing the data

Weka uses a standard file format for input and output data called Attribute-Relation File Format (ARFF). Therefore, in this step the data provided for this project is converted in an arff. The process used to convert was:

1. Define the name of the file using @relation

2. Create a list of 77 attributes, using @attibute, with generic names and define the type numeric for all of them.

3. Include one last attribute for labels and call it class.

4. Define the data relation using @data. Include the labels column provided in trainLabels.csv in the last column of train.csv. Copy the training data with the labels to the arff file under @data.

5. Save the file as .arff

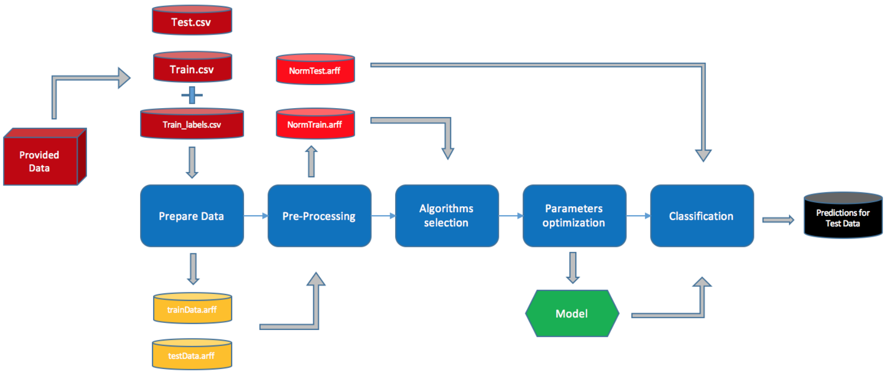

Figure 1: The framework

3 Pre-Processing the Training Data

By analyzing the training data, it is possible to see that the values for different features have different scales. In order to avoid having features with great values dominating features with small values, it is important to perform feature scaling. Feature scaling is also important for the computation process, helping to avoid numerical problems that can happen in some functions and algorithms when dealing with huge numerical values. In this project, the data is normalized using one of the weka filters called Normalized(). This filter performs normalization by using the equation:

$$z_i = \frac{x_i - min(x)}{max(x) - min(x)},$$

where $x = (x_1, ..., x_n)$ and z_i is the i^{th} normalized data. Therefore, all the data in each column representing a feature is normalized to the scale between 0 and 1.

4 Algorithms and function Selection

Even though information about training data was not provided, by analyzing it, it is possible to gather insights about which model to choose. The first important thing to notice is that the training set is reasonably large, with 50.000 samples and 77 features. Another important characteristic is that the number of samples is much larger than the number of features, which means that the method selected should be fast and also be able to identify complex relations between the attributes. Finally, it is used an histogram in the labels to identify how the they are distributed between the binary classes.

In this project, different algorithms with regards to complexity, speed and convergence are explored and evaluated. Algorithms such as logistic regression, random forest and logistic regression with stochastic gradient descent are used due to their speed and simplicity. These simple algorithms are compared with other more complex algorithms such as SVM, Bayes net, J48 and Voted Per-

10 fold Cross Validation							
Classifiers	Parameter	Value	Precision	Recall	Accuracy	F-Measure	Time(sec)
RF	Max Depth	5	0.9650	0.9650	0.9653	**0.9650**	27.95
LR	batch	100	0.9640	0.9640	0.9643	**0.9640**	10.59
SVM	cost/gamma	4/4	0.9630	0.9630	0.9633	**0.9630**	219.01
SGD	learning rate	0.01	0.9630	0.9630	0.9630	0.9630	13.63
J48	confidence	0.25	0.9600	0.9600	0.9605	0.9600	9.85
RT	Max Depth	5	0.9600	0.9610	0.9606	0.9600	0.55
VP	-	-	0.9490	0.9490	0.9493	0.9490	42.13
BN	-	-	0.9430	0.9290	0.9289	0.9320	3.80
ZeroR	-	-	0.6440	0.8020	0.8025	0.7150	0.01

Table 1: Results using 10 fold cross validation in Weka

ceptron that usually achieve higher performance in training data but they are more prune to overfit.

Some algorithms have different functions to implement. A reasonable choice for an SVM for example, is the RBF kernel function that is able to map the nonlinearities of the samples in a higher dimensional space but at the same time, has less hyperparemeters than other functions and for this reason is less prone to overfit the data with very complex models [2]. For the Bayes Net, the function used to optimize the search is hill climber.

5 Parameters optimization

For the both logistic regression algorithms, the parameter to optimize is the learning rate. For the J48, the confidence interval parameter is optimized by using values in the interval 0.1,0.5 with increments of 0.1. For the SVM with RBF Kernel, there are two parameters to optimize, cost and gamma. In this case, it is used grid search optimization, in which for each value of gamma, we test different values for the cost. The values for the cost and gamma are $2^{-5}, 2^{-3}, ..., 2^{3}, 2^{5}$. In Random Forest and Random Trees, the objective is to find the best depth of the tree within the range of 1 and 10.

6 Results in Training Data

To compare these algorithms five parameters are used: Precision, Recall, Accuracy and F-Measure. The basis of the comparison is the ZeroR algorithm that is basically a classifier that always selects the dominant class. Besides, two experiments are performed: One with 10 cross validation and another one with 2/3 split data where 2/3 of the data is used for training and 1/3 is used for testing.

The results for the cross validation experiment are shown in Table 1 and for the split data in Table 2. These Tables show the algorithms used, the optimal values for parameter optimization, the performance in the metrics and the time spent for building the model in Weka. It is clear that all the algorithms performed much better than the basis algorithm ZeroR. In both experiments, the best tree algorithms were Random Forest, Logistic Regression and SVM when considering F-measure metric as the defining measure. Both logistic and Random Forests are less complex algorithms that take less time to build the model. SVM is more complex and for this reason takes much longer to define a

2/3 Split Data							
Classifiers	Parameter	Value	Precision	Recall	Accuracy	F-Measure	Time(sec)
RF	Max Depth	5	0.9650	0.9650	0.9653	**0.9650**	27.50
LR	batch	100	0.9640	0.9640	0.9643	**0.9640**	10.17
SVM	cost/gamma	4/4	0.9630	0.9630	0.9639	**0.9640**	223.92
J48	confidence	0.1	0.9620	0.9620	0.9620	0.9620	9.67
RT	Max Depth	5	0.9600	0.9610	0.9606	0.9600	0.55
SGD	learning rate	0.01	0.9590	0.9590	0.9590	0.9590	11.36
VP	-	-	0.9540	0.9540	0.9540	0.9530	39.86
BN	-	-	0.9440	0.9290	0.9288	0.9320	0.17
ZeroR	-	-	0.6440	0.8020	0.8025	0.7150	0.01

Table 2: Results using 2/3 of the data for training and 1/3 for testing in Weka

classifier for the data. In this work, it is used a fast and robust library for the SVM called LibSVM [1].

7 Conclusion

In this project, it is performed a supervised learning task using the data provided by the instructor and the algorithms learned in the classroom. By using Weka, 9 algorithms were used to train a classifier. To improve the accuracy and avoid overfitting, parameter optimization in cross validation was used. The performance of each of these algorithms were evaluated using 4 metrics: precision, accuracy, recall and f-measure. By considering the best performance in F-measure, the best results were obtained using Random Forest, logistic regression and SVM. Therefore, these algorithms were used to classify the test set.

All the files mentioned in this work and the prediction results are included the zip folder. Also in the zip folder, there is a file called README.txt that describes all the files included.

References

[1] Chang, C.-C., and Lin, C.-J. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology 2* (2011), 27:1–27:27.

[2] Hsu, C.-W., Chang, C.-C., and Lin, C.-J. A practical guide to support vector classification. Tech. rep., Department of Computer Science, National Taiwan University, 2003.

[3] Witten, I. H., Frank, E., Trigg, L., Hall, M., Holmes, G., and Cunningham, S. J. Weka: Practical machine learning tools and techniques with java implementations, 1999.