Report Machine Learning, Homework 4

Domenico Alfano

The report describes the results obtained during the development of Homework 4. The main goal of the assignment has been to evaluate the capabilities and functionalities of Support Vector Machine and Cross-Validation.

1 Introduction

In machine learning, support vector machines are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier. An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall. In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces.

2 Linear SVM

For this part of homework, two classes of the dataset were selected in order to perform a binary classification using Linear SVM. The new dataset was saved in a Matrix X, and the labels the images were located in an array y. The dataset was split into the training, validation and testing set as 50%, 20%, 30% by using the function train test split(). In order to obtain a high classification accuracy, its necessary to chose the correct value of C. Several training and validating processes were performed to tune C. The value of C was changed from 0.001 to 1000 each decade. Figure 1 shows the obtained accuracies by different C. Figure 2 shows the changes of decision boundaries for each C value.

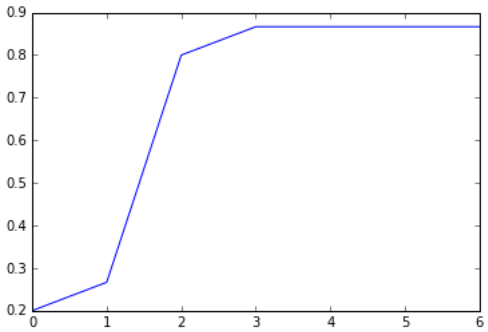

Figure1

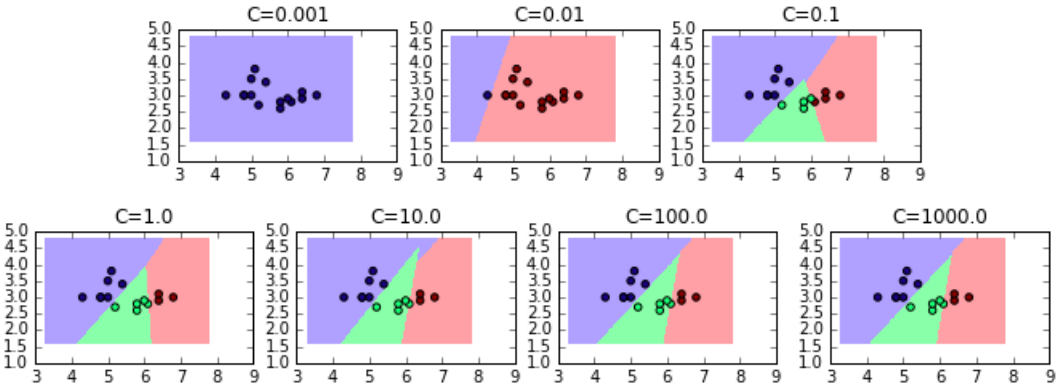

Figure 2

The C parameter tells the SVM optimization how much you want to avoid misclassifying each training example. For large values of C, the optimization will choose a smaller-margin hyperplane if that hyperplane does a better job of getting all the training points classified correctly. Conversely, a very small value of C will cause the optimizer to look for a larger-margin separating hyperplane, even if that hyperplane misclassifies more points. For very tiny values of C, you should get misclassified examples, often even if your training data is linearly separable. From the plot, the value of C which gives the highest accuracy was selected as the best solution, basically because I'm selecting the C which will lie to the lowest classification error. Using a C=10 an accuracy of 0.866 was obtained. Moreover, testing phase was performed in order to evaluate the accuracy of the trained SVM using the tunned C. The accuracy obtained was 0.844, which is not better than the result on the validation phase. One explanation about the difference among the values is given by the fact that the testing set is bigger than the validation set, increasing the size of information to classify. However, the fact that the accuracy results are quite similar shows the good performance of the trained SVM. Figure 3 shows the best value of C.

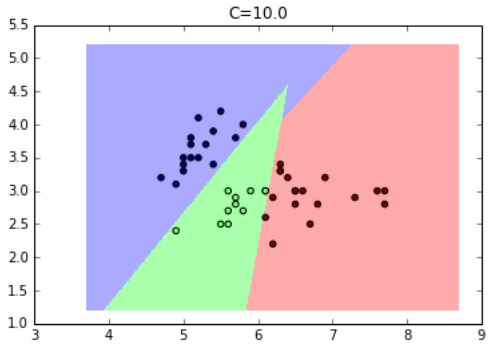

Figure3

3 Non-linear SVM

In machine learning, the (Gaussian) radial basis function kernel, or RBF kernel, is a popular kernel function used in various kernelized learning algorithms. The RBF kernel on two samples x and x', represented as feature vectors in some input space, is defined as

$$K(x, x') = exp\left(-\frac{\|x-x'\|^2}{2\sigma^2}\right)$$

where $\|x - x'\|^2$ may be recognized as the squared Euclidean distance between the two feature vectors and σ is a free parameter. An equivalent, but simpler, definition involves a parameter $\gamma = \frac{1}{2\sigma^2}$:

$$K(x, x') = exp\left(-\gamma\|x - x'\|^2\right)$$

As in the previous section, the training, validation and testing sets were generated as 50%, 20%, 30% but for this part was used the RBF Kernel. In Figure 4 is shown the best value of C. In this case we notice that the boundaries are different because the decision boundary for a linear support vector machine is an hyperplane. For non-linear kernel support vector machines, the decision boundary of the support vector machine is not an hyperplane in the original feature space but a non-linear hypersurface whose shape depends on the type of kernel.

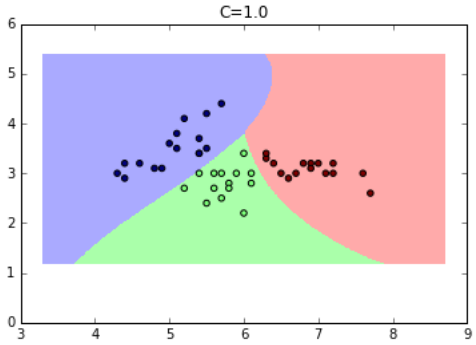

Figure 4

Using a C=1 an accuracy of 0.933 was obtained. Also in this case testing phase was performed in order to evaluate the accuracy of the trained SVM using the tunned C. The accuracy obtained was 0.822, which is not better than the result on the validation phase. Successively, the training of the Non-Linear SVM was performed by using the function SVC(), where the value of C, γ and the type of kernel (RBF) can be chosen. In this case, the value of C and γ should be tuned to guarantee the best accuracy of the SVM. To perform the tunning process a grid search was implemented, where the values of C and γ were changed in a specified range. As we can see in Figure 5, γ and C were selected choosing the values that return the highest accuracy of the SVM. In this case, using C=1 and a γ=1 the result accuracy was 0.933 on the validation phase. Once again, the tuned values was used in the testing set giving an accuracy of 0.822. As in the previous section, the result in the validation set is better than in the training case.

C / γ	0,01	0,1	1	10	100
0,001	0,4	0,4	0,4	0,4	0,4
0,01	0,4	0,4	0,4	0,4	0,4
0,1	0,4	0,466	0,866	0,599	0,4
1	0,599	0,933	0,933	0,866	0,599
10	0,933	0,933	0,866	0,933	0,666
100	0,933	0,933	0,866	1.0	0,666
1000	0,933	0,866	0,866	0,933	0,666

Figure 5

Figure 6 shows the best value of C and γ.

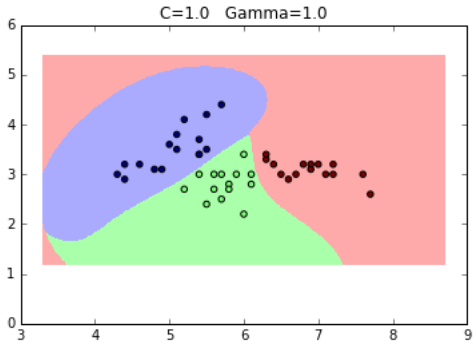

Figure 6

4 K-Fold

Cross-Validation is a model validation technique for assessing how the results of a statistical analysis will generalize to an independent data set. It is mainly used in settings where the goal is prediction, and one wants to estimate how accurately a predictive model will perform in practice. In a prediction problem, a model is usually given a dataset of known data on which training is run, and a dataset of unknown data against which the model is tested. The goal of cross-validation is to define a dataset to "test" the model in the training phase, in order to limit problems like overfitting, give an insight on how the model will generalize to an independent dataset. In k-fold cross-validation, the original sample is randomly partitioned into k equal sized subsamples. Of the k subsamples, a single subsample is retained as the validation data for testing the model, and the remaining $k-1$ subsamples are used as training data. The cross-validation process is then repeated k times, with each of the k subsamples used exactly once as the validation data. The k results from the folds can then be averaged to produce a single estimation. The advantage of this method over repeated random sub-sampling is that all observations are used for both training and validation, and each observation is used for validation exactly once. In this last part of the homework, the dataset was split into the training and test set as 70%,30% by using the function train test split() and as requested was chosen K=5. As we can see in Figure 7, was implemented a new grid search for γ and C but this time perform 5-fold validation.

C / γ	0,01	0,1	1	10	100
0,001	0,699	0,699	0,699	0,699	0,5
0,01	0,699	0,699	0,699	0,699	0,5
0,1	0,699	0,699	0,699	0,699	0,5
1	0,699	0,849	0,75	0,699	0,599
10	0,9	0,8	0,8	0,65	0,55
100	0,75	0,8	0,699	0,699	0,55
1000	0,75	0,8	0,699	0,65	0,55

Figure 7

From the Figure 7, the values of C and γ which gives the highest accuracy was selected as the best solutions. Using a C=10 and γ=0,01 an accuracy of 0.9 was obtained. Moreover, testing phase was performed in order to evaluate the accuracy of the trained SVM using the tunned C and γ.
Figure 8 shows the best value of C and γ.

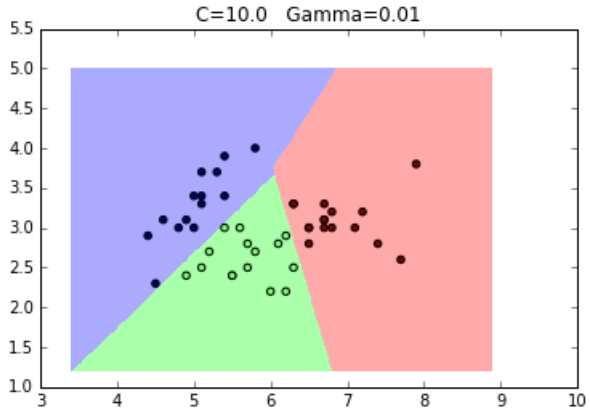

Figure 8

The accuracy obtained was 0.866, which is not better than the result on the validation phase. It's clear that the results respect to the previous section are different. Probably beacuse this approach is more intensive from computational point of wiew if $K << n$ where n is the dimensionality of data set. Moreover, K-Fold cross-validation tends to have less variability.