Machine Learning: Pattern Recognition

Report Lab 1

11 September, 2012

By:
Paris Mavromoustakos
Georgios Methenitis
Marios Tzakris

Introduction

The purpose of this lab assignment was to apply a k-Nearest-Neighbors classification algorithm on a given set of data-points. To achieve this we had to become familiar with the Matlab environment, including the Netlab package. The Netlab package is a complete suit of Matlab functions implementing machine learning techniques.

Exercise 1

First of all, we loaded a given data file (`twoclass.mat`) which contained two classes of two-dimensional data points. Before creating both the training and test sets, we shuffled the two matrices in order to create groups of random data points. **The training set consists of 75% of data points from class A appended to 75% of data points from class B, while the test set contains the remaining 25% of both classes A and B**. Figure 1 presents the plot results of the training set. **Data points of different classes are depicted with different symbols and colors**. Figure 2 presents the plot results of the test set.

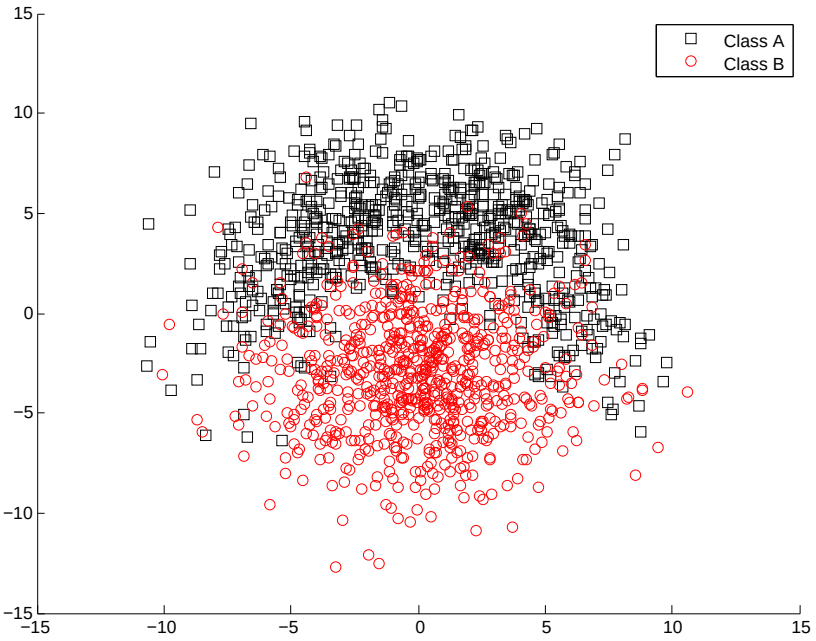

Figure 1: Data points of the Training Set.

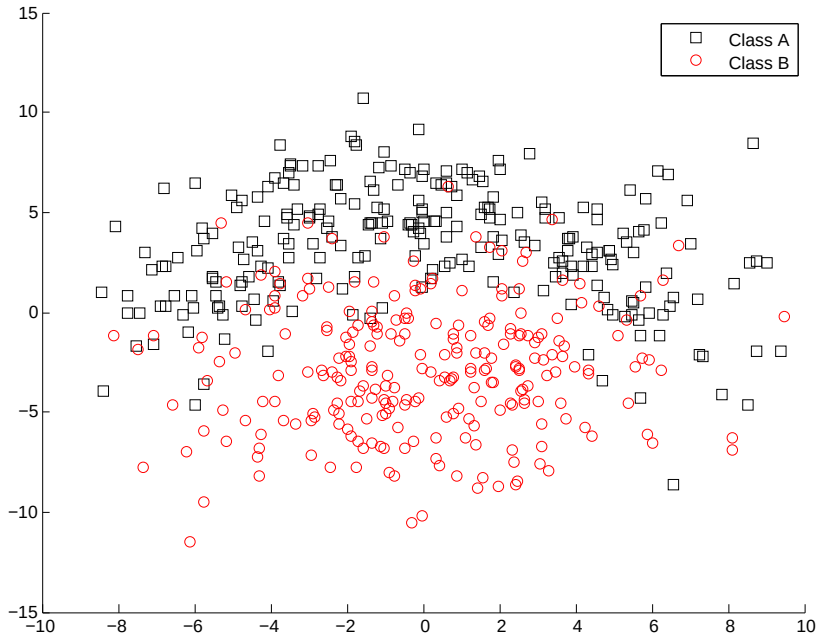

Figure 2: Data points of the Test Set.

Exercise 2

We trained our knn classifier using the training data, evaluating its performance on the test data and computing the misclassification rate. The misclassification rate taking into consideration only one neighbor (k=1) is 17%. Then we tried several odd values for k ($k = 1, 3, 5, \ldots, 29$), computing the error for each case as shown in Figure 3.

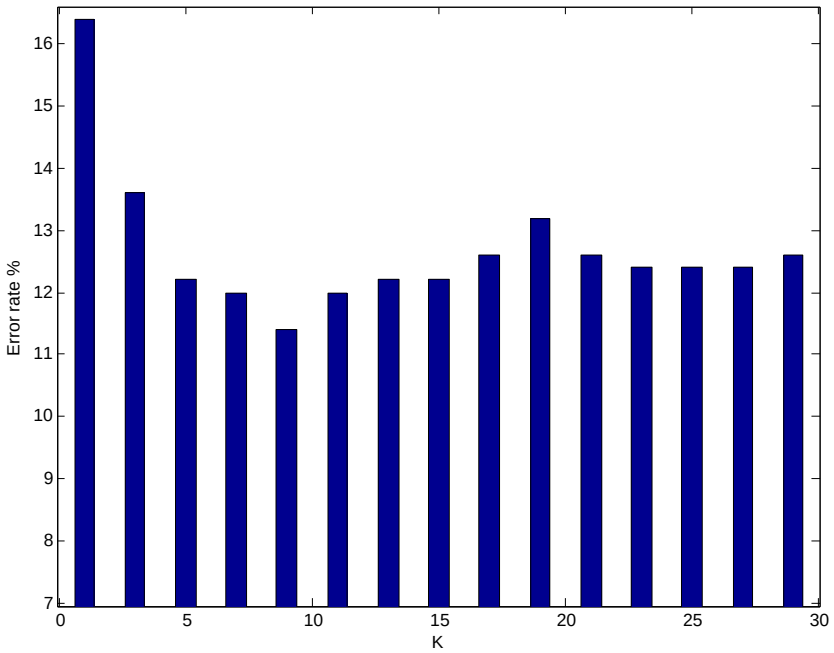

Figure 3: Classification Results.

The value of k that we would choose is $k = 9$, because as the figure states, for $k = 9$ the **test error** reaches its **minimum** value, however, for $k > 9$, an increment of the misclassification rate is detected as we increase the value of k. However, we are not able to definitely state that $k = 9$ is the optimal value for k. As mentioned above, as the value of k increases, the test error is expected to decrease. This behavior does not refer to the whole range of k's tested values, but as k becomes significantly high the test error calculated increases as well. We interpreted this fact as a typical case of **overfitting**, which is **expected** to occur if the value of k becomes high enough (in this case, $k > 9$).

Exercise 3

3.1

As described before, the problem of **overfitting** occurs while we are increasing the value of k. This may happen because either we have a model which is too complex, or the data used for training is too little. In our case, the model is relatively simple so, we should find a way to increase our training data. Here comes the solution of **cross-Validation**. Cross-Validation helps us to overcome the problem of overfitting while **it expands the data used for training purposes**. In each iteration of the algorithm, a new training and test set are used, offers us more **accurate predictions** when it comes to validation. Another algorithm that could be applied, in order to increase accuracy, is **Bootstrapping** algorithm, which can lead us at least to a critical conclusion regarding the variance of the training data we are given, informing us about the possible outcome of the whole simulation. We applied Cross-Validation algorithm not only to expand our training and test data, but also to reach to a more **reliable and accurate conclusion** about the optimal value of k.

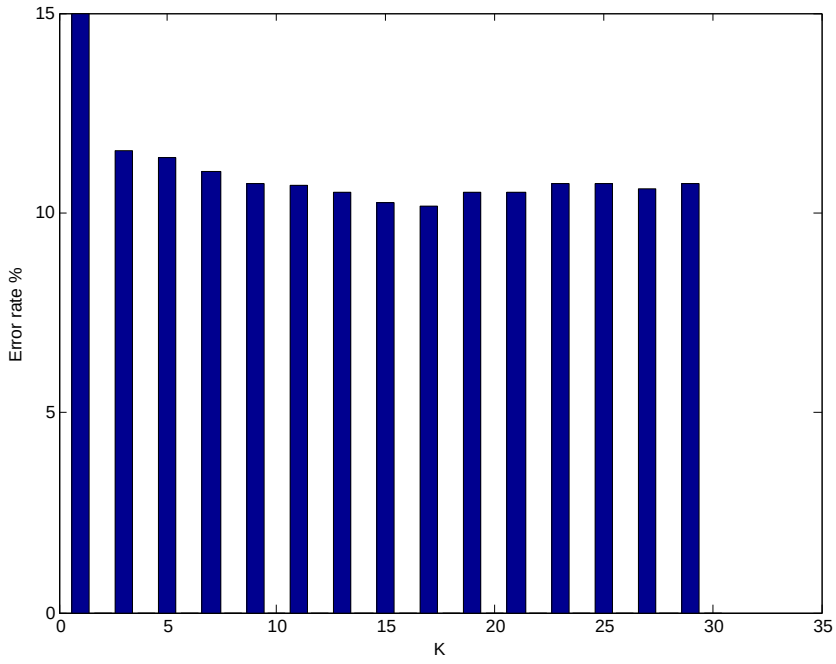

Figure 4: Results of Cross-Validation

3.2

In the second part of exercise three we implemented Cross-Validation algorithm in order to derive more accurately an optimal value for k. 10-Fold Cross Validation splits data into 10 equal subsets. In each iteration of the algorithm we use one of these subsets as the validation (test) data and the other nine subsets to train our classifier. The outcome of this implementation is presented in Figure 4. The misclassification error of the cross-validated data reaches its minimum value for $k = 9$, enabling us to choose more accurately an optimal value for k variable. We chose $k = 9$ as a good estimation.

3.3

This value ($k = 9$) is the one that we can use in our **final classifier**, although the misclassification rate becomes minimum when $k = 17$. We have chosen $k = 9$ because we prefer to "sacrifice" a very small difference in the algorithm's **accuracy** in order to implement a more **simple** classificator. We estimated that a value of $k = 17$ would over-smooth our classificator and make our algorithm **computationally expensive**. Cross-validation is a way to predict the accuracy of a model to a hypothetical test set when a definite test set is not available. Cross-validation algorithm is an applicable way to predict the performance of a model on a test set using your data in a way that more accurate estimation can be derived.

3.4

To compute the **test error**, we calculate the **average** misclassification rate over the 10 iterations of the 10-fold Cross-Validation algorithm. The test error computed is different in each iteration, so the **average** value gives us a good estimation of the misclassification rate. The classification error which we computed is an estimation over several experiments with different training and validation data. The use of a **validation set** is to define whether the predictions of the classification are accurate or not.

Further Experimentation

To experiment further with the k-NN algorithm we applied the Cross-Validation method over a huge amount of values for the k variable. We used a set of values $k = \{1, 3, 5, \ldots, 1000\}$. In Figure 5 presented the results of our experiment. We can distinguish two regions. The red colored region is the results of the k-NN algorithm when we apply it with a big value for its k variable. We can realize that giving such a huge value for the k variable results to the increase of the misclassification rate as the desicion boundaries are becoming really smooth. Although we can indicate an optimal error rate at the value of k=93, we doubt whether it would be applicable in real time constraints. A value so high would create a very "expensive" algorithm, which is unnecessary for a simple 2-class classificator. However, if our goal was to minimize the misclassification rate regardless the algorithm's complexity, k=93 would possibly be the optimal value.

Conclusion

In conclusion, this project really helped us understand the nature of the k-NN algorithm. Moreover, we were able to point out its advantages while facing the expected drawbacks, like the overfitting phenomenon. On the other hand we studied and implemented methods of dealing with overfitting, and thus reaching more accurate and reliable conclusions.

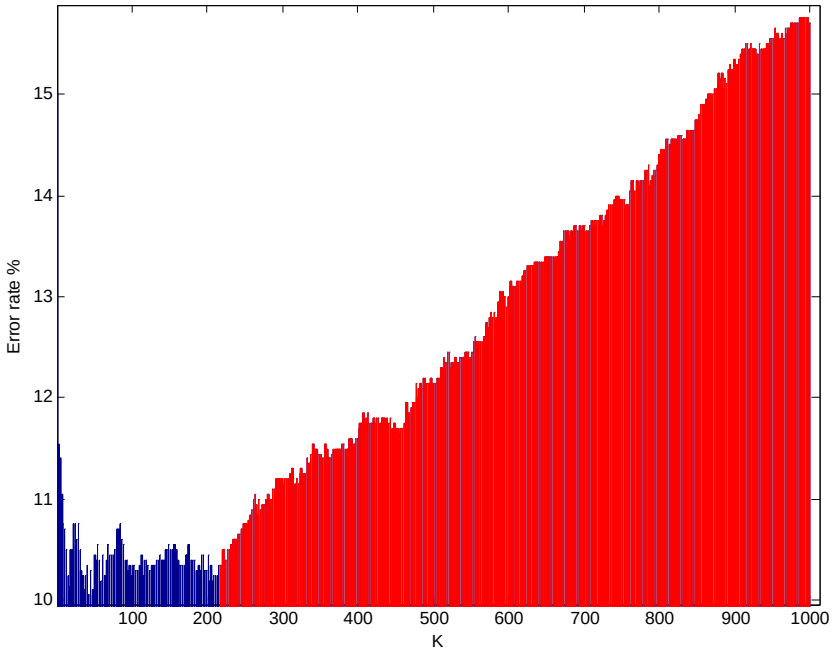

Figure 5: Further Experimentation