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-Neighbours classification algorithm on a given set of datapoints. 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 datapoints. Before creating both the training and test sets, we shuffled the two matrices in order to create groups of random datapoints. **The training set consists of 75% of class A's datapoints appended to 75% of class B's datapoints, while the test set contains the remaining 25% of both classes**. Figure 1 presents the plot results of the training set. **Datapoints of different classes are depicted with different symbols and colors**. Figure 2 presents the plot results of the test set.

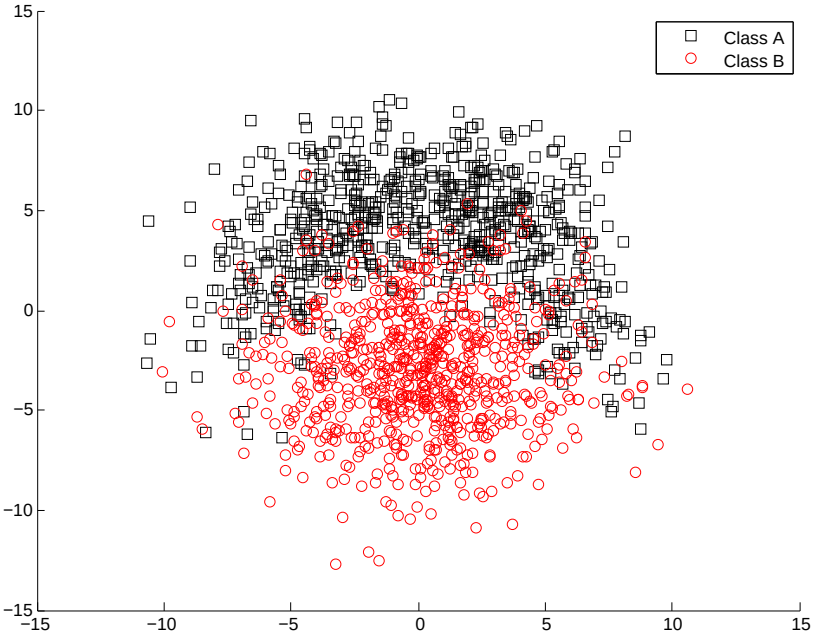

Figure 1: Training Set.

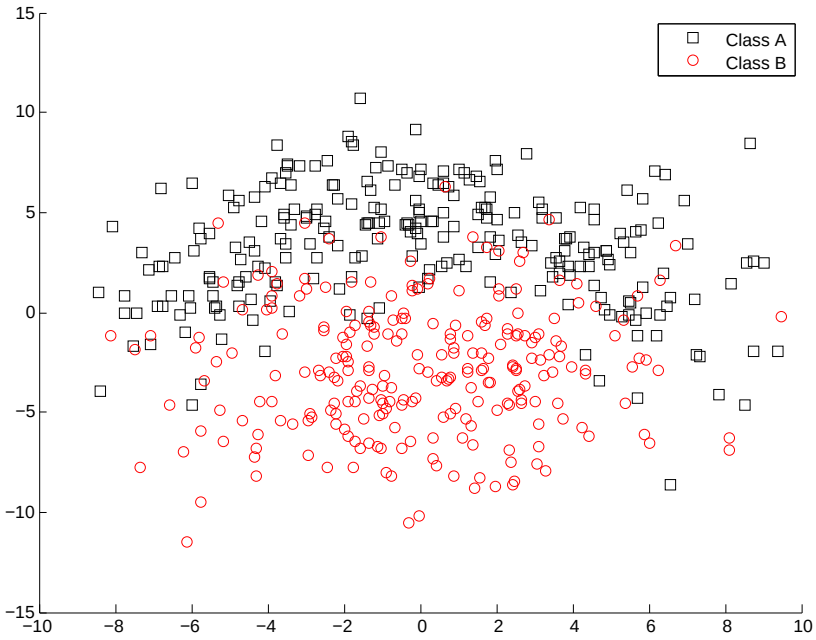

Figure 2: Training Set.

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 neighbour (k=1) is 17%. Then we tried several odd values for k (k=1, 3, 5, . . . , 29), computing the error for each case as shown in Figure 3.

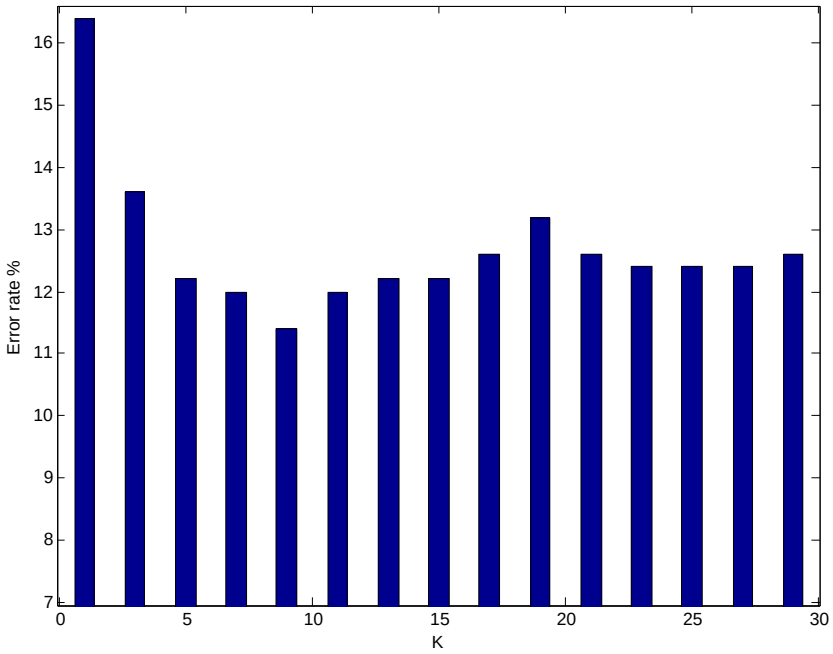

Figure 3: Classification Results.

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 incrementation of the misclassification rate is detected as we increase the value of k. However, we are not able to definately 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 increments aswell. 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

As described before, an **overfitting** problem has occurred while increasing the value of k. This might 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 overcome overfitting while **expanding the data used for training purposes**. Each time we run the algorithm, a new training and test set is being used, thus giving us more **accurate predictions** when it comes to validation. Another algorithm we could apply in order to increase accuracy, is **bootstrapping**, which would at least lead us to a critical conclusion regarding the variance of the data we were given, informing us about the possible outcome of the whole simulation. We applied cross-validation not onty to expand our training and test data, but also to reach a more **reliable conclusion** about the value of k.

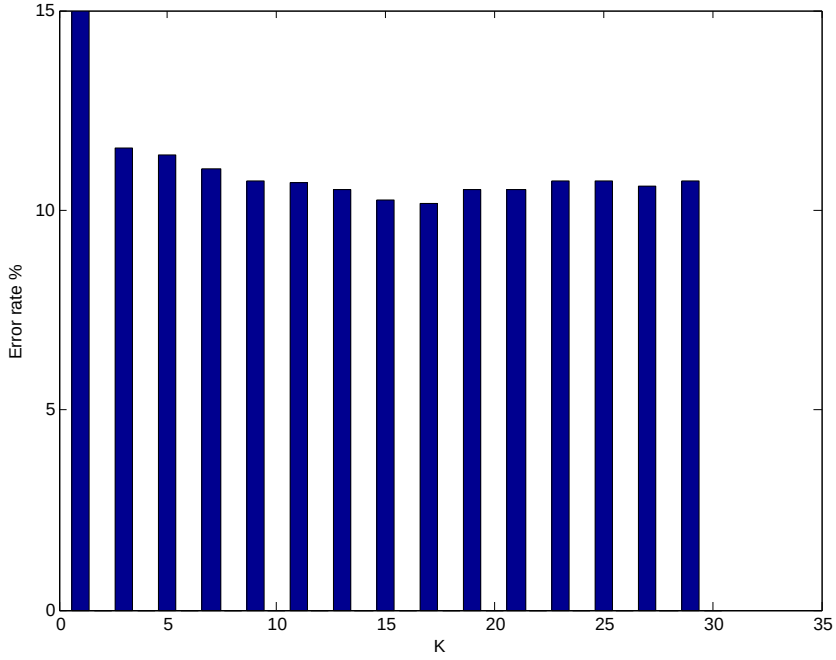

Figure 4: Results of Cross-Validation

As shown in Figure 4, the misclassification error of the cross-validated data reaches its minimum value for k=9, enabling us to state in **certainty** that k=9 is the optimal value for k. This value is the one we would 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 oversmooth our classificator and make our algorithm **computationally expensive**. To compute the **test error**, we calculated the **average** 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 estimate of the misclassification error value. The use of a **validation set** is to define whether the classificator's predictions are accurate or not.

Conclusion

In conclusion, this project has clearly 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.