Hong Kong University of Science and Technology
COMP 5212: Machine Learning
Fall 2013

Project 2
Due: 28/11/2013, Thursday, 11:59pm

1 Objectives

The objectives of this project are two-fold:

1. To acquire a better understanding of classification methods by implementing two methods and using a public-domain software package for support vector machines (SVM).
2. To compare the performance of several classification methods by conducting empirical comparative study on five real-world data sets.

2 Major Tasks

The project consists of the following tasks:

1. To implement a logistic regression (a.k.a. logistic discrimination) model for binary classification.
2. To implement a k-nearest neighbor classifier for binary classification.
3. To learn to use a public-domain SVM software package.
4. To conduct empirical study to compare several classification methods.
5. To write up a project report.

Each of these tasks will be elaborated in the following subsections.

2.1 Logistic Regression and k-Nearest Neighbor Classifier

The logistic regression algorithm discussed in class uses gradient descent to minimize the cross-entropy error function. Your implementation may be based on this gradient-descent algorithm which requires that the step size parameter η be specified. Try out a few values (<1) and choose one that can lead to stable convergence. You may terminate the learning procedure if the improvement between iterations is not larger than a small threshold or if the number of iterations has reached a prespecified maximum number. Since the solution found may depend on the initial weight values chosen randomly, you may repeat each setting multiple times and report the average classification accuracy. Alternatively, if you wish, you may learn to use the more powerful `fminunc` function which, among other things, does not require you to specify the step size parameter.

The k-nearest neighbor classifier makes classification decision based on the majority class among the k nearest neighbors. In case there is a tie (e.g., two classes have the same largest number of votes from the k nearest neighbors), you may randomly choose one class among those with the largest number of votes. If you wish, you may also implement an enhanced scheme by using the distances of the nearest neighbors from the test case to break the tie in a more informed way. However, this extension is optional.

While the k-nearest neighbor classifier is easy to implement, straightforward implementation may be inefficient especially when the sample size is large. The bottleneck is the computation involved in identifying the nearest neighbors. For simplicity, however, a straightforward implementation (with $O(N)$ time complexity for classifying new test data, where N is the number of training examples) is acceptable for the purpose of this project.

The model parameter k should be determined using cross validation. The generalization performance of the classifier is estimated for each candidate value of $k \in \{1, \ldots, 10\}$. This is done by randomly sampling 80% of the training examples to train a classifier and then testing it on the remaining 20%. Ten such random data splits are performed and the average over these 10 trials is used to estimate the generalization performance. The value of k that gives the best performance can then be found. Subsequently, a classifier with the best k value is trained using all the training examples.

Besides training a k-nearest neighbor classifier on the original training data, we also train another k-nearest neighbor classifier on the training data after performing principal component analysis (PCA) to reduce the data dimensionality for efficiency. You should determine the number of principal components to use such that the proportion of variance to keep is at least 90%. This can be done by modifying your PCA code used for Project 1. For simplicity, the best value of k determined above will be used here directly without the need for performing cross validation again.

For both logistic regression and the k-nearest neighbor classifier, you are expected to do the implementation all by yourself so you can gain a better understanding of these two methods. Octave/MATLAB is the preferred language choice which can allow you to do fast prototyping possibly at the expense of run-time efficiency. You may also use some other programming languages such as C++ and Java if you insist, but this is not recommended because you then cannot take advantage of the powerful and convenient matrix manipulation capabilities and built-in functions provided by Octave/MATLAB.

2.2 SVM Software

You may use any SVM implementation, but the recommended choice is SVM^{light} due to its simplicity:[1]

`http://svmlight.joachims.org/`

Files for the executable code are available for the common operating systems.

[1]LIBSVM (`http://www.csie.ntu.edu.tw/~cjlin/libsvm/`) is a more powerful SVM implementation but its installation may be slightly more involved.

2.3 Empirical Study

You will use five binary classification data sets which can be downloaded as a ZIP file from the project web page. The following table shows the website, number of features, number of training examples, and number of test examples for each data set.

Data set	Website	#features	#train	#test
Ionosphere	LIBSVM	34	250	101
ISOLET	ZJU	617	120	120
Liver	LIBSVM	6	200	145
MNIST	ZJU	784	400	400
Mushroom	LIBSVM	112	4062	4062

The URLs of the websites LIBSVM and ZJU are:

`http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html`
`http://www.cad.zju.edu.cn/home/dengcai/Data/MLData.html`

Note that the two data sets from ZJU originally contain more than two classes each. Subsets of the data have been extracted to form the two data sets used here.

When you load each binary data file into Octave/MATLAB, you will find the variables X and Y. Each row of X stores the features of one example and the corresponding row of Y stores its class label (0 or 1). As is always the case, the class label files for the test sets should not be used for classifier training but only for measuring the classification accuracy on the test data.

SVM^{light} uses a different data file format. For your convenience, a simple Octave/MATLAB program is available on the project web page for generating data files that can be used by SVM^{light}.

For each of the five data sets, the following methods will be compared with respect to the classification accuracy on the training set and the test set separately:

- Logistic regression
- k-nearest neighbor classifier (best value of $k \in \{1, \ldots, 10\}$ determined by cross validation)
- k-nearest neighbor classifier after performing PCA (number of principal components determined by preserving at least 90% of the data variance)
- SVM with linear kernel (default value of regularization parameter C)
- SVM with RBF kernel (default value of regularization parameter C; kernel parameter determined by trying out a few values without having to perform cross validation)[2]

For the first three methods (i.e., logistic regression, k-nearest neighbor classifier, and k-nearest neighbor classifier after PCA), you are expected to also report the time required by each of the methods to complete the task, excluding the time needed for loading the data files. This may be done using the `time` function. The best value of k for the k-nearest neighbor classifier and

[2]Cross validation is not required but you may do it if you wish.

the number of principal components used by PCA for dimensionality reduction should also be reported.

Your programs should be written in such a way that the TA can run them easily to verify the results reported by you.

2.4 Report Writing

In your report, you need to present the parameter settings and the experimental results. Besides reporting the classification accuracy (for both training and test data) in numbers, graphical aids should also be used to compare the performance of different methods visually. For the CPU time information, you may just report it in numbers.

3 Some Programming Tips

As is always the case, good programming practices should be applied when coding your program. Below are some common ones but they are by no means complete:

- Using functions to structure program clearly
- Using meaningful variable and function names to improve readability
- Using indentation
- Using consistent styles
- Including concise but informative comments

For Octave/MATLAB in particular, you are highly recommended to take full advantage of the built-in functions. Also, using loops to index individual elements in matrices and arrays should be avoided as much as possible. Instead, block indexing without explicitly using loops is much more efficient. Proper use of these implementation tricks often leads to speedup by orders of magnitude.

4 Project Submission

Project submission should be done electronically using the Course Assignment Submission System (CASS):

`http://cssystem.cse.ust.hk/UGuides/cass/student.html`

There should be two files in your submission:

1. **Project report** (with filename `report`): preferably in PDF format.
2. **Source code and a README file** (with filename `code`): all necessary code for running your program as well as a brief user guide for the TA to run the programs easily to

verify your results, all compressed into a single ZIP or RAR file. The data should not be submitted to keep the file size small.

When multiple versions with the same filename are submitted, only the latest version according to the timestamp will be used for grading.

5 Grading Scheme

This project will be counted towards 10% of your final course grade. The weights for different tasks are as follows:

- Implementation of logistic regression [2%]
- Implementation of k-nearest neighbor classifier optionally with PCA applied [3%]
- Empirical study [3%]
- Project report [2%]

6 Academic Integrity

Please read carefully the HKUST Academic Honor Code on the course website.

While you may discuss with your fellow classmates on general ideas about the project, your submission should be based on your own independent effort. In case you seek help from any person or reference (from the Web or other sources), you should state it very clearly in your submission. Failure to do so is considered plagiarism which will be handled with appropriate disciplinary actions.