Hong Kong University of Science and Technology
COMP 5212: Machine Learning
Fall 2016

Project 1
Due: 7 October 2016, Friday, 11:59pm

1 Objective

The objective of this project is twofold:

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 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 classification.

2. To implement a single-hidden-layer neural network model for 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 Neural Network Models

As discussed in class, neural network classifiers generalize logistic regression by introducing one or more hidden layers. Although you may implement the two models separately for this project, you are also allowed to implement a single, more general model which can be specialized to the two models needed here by specifying the number of hidden layers to be 0 and 1, respectively.

Learning of both models may use a (batch or stochastic) gradient-descent algorithm by minimizing the cross-entropy loss as discussed in class.[1] It requires that the step size parameter η be specified. Try out a few values (<1) and choose one that leads to stable convergence. You may also decrease η gradually during the learning process to enhance convergence. A common criterion used to terminate the learning procedure is when the improvement between iterations

[1] For simplicity, you are not required to add regularization terms to the error functions though you may do it if you wish.

does not exceed a small threshold or when the number of iterations has reached a prespecified maximum. 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 in MATLAB which, among other things, does not require you to specify the step size parameter.

For the single-hidden-layer neural network model, the number of hidden units H should be determined using cross validation. The generalization performance of the model is estimated for each candidate value of $H \in \{1, 2, \ldots, 10\}$. This is done by randomly sampling 80% of the training instances to train a classifier and then testing it on the remaining 20%. Five such random data splits are performed and the average over these five trials is used to estimate the generalization performance. The value H^* that gives the best performance among the 10 choices of H can then be found. Subsequently, a neural network classifier with H^* hidden units in a single layer is trained from scratch using all the training instances available.

You are expected to do the implementation all by yourself so you will gain a better understanding of the two methods. MATLAB/Octave is the preferred language choice which facilitates fast prototyping, possibly at the expense of run-time efficiency.[2] You may also use some other programming language, such as C++, Java, Python, or R, if you so wish, but then you may not be able to take advantage of the powerful and convenient matrix manipulation capabilities and built-in functions provided by MATLAB.

2.2 SVM Software

You may use any SVM implementation, but the recommended choice is SVM^{light} due to its simplicity:[3]

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

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

2.3 Empirical Study

You will use five binary classification data sets which are available as a ZIP file (`datasets.zip`). The following table shows the number of features, number of training examples, and number of test examples for each data set.

Data set	#features	#train	#test
Australian	14	552	138
Breast cancer	10	547	136
Diabetes	8	615	153
German numer	24	800	200
Heart	13	216	54

When you load each binary data file into MATLAB, you will find the variables X and Y. Each

[2] When we refer to MATLAB hereafter, we mean either MATLAB or Octave.

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

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 MATLAB program (`matlab2svmlight.m`) is provided for generating data files that can be used by SVM^{light}.

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

- Logistic regression
- Neural network with H^* hidden units (H^* determined by cross validation)
- 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)[4]

For the first two methods, you are expected to also report the time required by each method to complete the task, excluding the time needed for loading the data files. This may be done using the `time` function in MATLAB. For the neural network model, you should also report the performance of each value of $H \in \{1, 2, \ldots, 10\}$ in the cross validation procedure for determining the best value H^*.

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 are expected to present the parameter settings and the experiment 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 MATLAB in particular, you are highly recommended to take full advantage of the built-in

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

functions which can keep your program both short and efficient. Note that using loops to index individual elements in matrices and arrays should be avoided in MATLAB 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 only 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 with the following naming convention required:

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. Files not adhering to the naming convention above will be ignored.

5 Grading Scheme

This project will be counted towards 8% of your final course grade. The maximum scores for different tasks are as follows:

- Implementation of logistic regression [20 points]
- Implementation of neural network model [20 points]
- Empirical study [30 points]
- Project report [30 points]

Late submission will be accepted but with penalty. The late penalty is deduction of one point (out of a maximum of 100 points) for every minute late.

6 Academic Integrity

Please read carefully the relevant web pages linked from the course website.

While you may discuss with your 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 source, you should state it clearly in your submission. Failure to do so is considered plagiarism which will lead to appropriate disciplinary actions.