6.867: Homework 2

This assignment will focus on two different strategies for binary classification, logistic regression and support vector machines. In both cases, we will use data with class labels in $\{-1, +1\}$.

You will find a zip file with some useful code and data in the Resources section of the Piazza course page. You can do these assignments in any computational system that you are used to. We recommend Matlab or the Pylab/Numpy/Scipy/Matplotlib cluster of packages and we'll try to provide help for those two systems. If you use anything else, you're on your own...

You will be turning in a single, readable "paper" (a single PDF file) with your solutions. We will be emulating the process of submitting papers for publication to a conference. We will be using an actual conference review system (Easy Chair) to have these papers peer reviewed (by the other students). This means that your answers have to be readable and understandable to your peers and, where possible, interesting. Note that when explanations are called for, you will need to convince the reviewers that you understand what you're talking about. The course staff will serve as the Program Committee for the conference and make all final decisions. The details of this process will be posted on Piazza.

Grading process and due dates

- We *strongly* encourage you to do this assignment in groups of two students, though you may do it individually if you wish. Post on Piazza or email a TA or instructor if you'd like help finding a partner.
- We *strongly* encourage you to submit your paper via the Easy Chair site well before 11:59 PM on *Tuesday October 20*.
- If that due date will interfere with your ability to study for the exam, then you may turn the assignment in by 11:59 PM on *Tuesday October 27*, but note that it will compress your time available to do the next assignment.
- Each student who was an author or co-author on a submission will be assigned 2 papers to review.
- Reviews must be entered on the Easy Chair site by *11:59PM on Tuesday, November 3*.
- All reviews will be made visible shortly thereafter, and students will have a 2–3 day period in which to enter rebuttals of their reviews into EasyChair if they wish.

Grading rubric

Your paper must be anonymous (no identifying information should appear in the PDF file). If it is not, it will automatically receive a 20% deduction, and will be graded by a grumpy staff member.

The paper must be no more than 6 pages long in a font no smaller than 10 point. It should include whatever tables, graphs, plots, etc., are necessary to demonstrate your work and conclusions. *It should not include code.*

Each of the four parts of the assigment will be graded on a scale from 0 to 5 (where 0 is failing and 5 is an A) on two aspects:

- **Content:** Did the solution answer the questions posed? Were the answers correct? Were the experiments well-designed or examples well chosen?
- **Clarity:** Were the results written up clearly? Were the plots labeled appropriately and described well? Did the plots support the points in the paper? Did the discussion in the paper illuminate the plots?

As a reviewer, you will be asked to provide a score for each section, and at at least two paragraphs of feedback, per review, explaining things that were done well and things that could have been improved upon.

Your overall score for this assignment will be:

- **80%**: The average of all 8 scores on your assignment given by all three reviewers.
- **20%**: A score for the quality of your reviews. This will be full credit, by default. But we will skim reviews and examine some carefully and may reduce this grade for review commentary that is sloppy or wrong.

The course staff will spot-check submissions and reviews, paying careful attention to cases where there were rebuttals. The staff will act as the program committee and determine a final score. Our overall goals in this process are:

- To motivate you to work seriously on the problems and learn something about the machine learning material in the process
- To engage you in thinking critically and learning from other students' solutions to the problems

We will arrange to give full credit to anyone who submits a serious and careful solution to the problems and who gives evidence of having read carefully the solutions they were assigned and who writes thoughtful reviews of them.

The following questions are the points that your paper should cover in order to receive full credit. Your presentation should roughly follow the order of these questions so that your reviewers can see what you're doing.

1 Logistic Regression (LR)

To illustrate the similarities between logistic regression and SVMs, we will formulate the LR objective function as

$$\text{NLL}(w) = \sum_i \log(1 + e^{-y^{(i)}(x^{(i)} \cdot w + w_0)})$$

1. Write code to optimize the logistic regression objective, but with $L2$ regularization on the weight vector. The objective function should be of the form:

   $$E_{LR}(w) = \text{NLL}(w) + \lambda w^T w$$

   Use your gradient descent implementation from HW1 if possible, because it will be more instructive and easy for you to understand what it is doing; however, if your implementation from HW1 does not work well, you may use a professional implementation instead.

   Find optimal values of w and w_0.

2. To test your implementation, we provide a number of datasets in the `data` folder: `data_stdev1`, `data_stdev2`, `data_stdev4` and `data_nonsep`. Set $\lambda = 0$ and explain the results you obtain for the decision boundary and classification error rate on the training and validation set for each dataset (We provide the skeleton code `lr_test.py/m`).

3. Repeat the previous part with varying λ. Report the behavior of the algorithm on the training and validation datasets when λ increases. Pay particular attention to the non-separable dataset.

2 Support Vector Machine (SVM)

1. Implement the dual form of linear SVMs with slack variables. Please do not use the built-in SVM implementation in Matlab or Pylab. Instead, write a program that takes data as input, convert it to the appropriate objective function and constraints, and then call a quadratic programming package to solve it. See the file `optimizers.txt` for installation and usage for matlab/python.

   Show in your report the constraints and objective that you generate for the 2D problem with positive examples (1, 2), (2, 2) and negative examples (0, 0), (-2, 3).

2. Test your implementation on the same 2D datasets from problem 1. Set C=1 and report/explain your decision boundary and classification error rate on the training and validation sets We provide the skeleton code `svm_test.py/m`).

3. The dual form SVM is useful for several reasons, including an ability to handle kernel functions that are hard to express as feature functions in the primal form. Extend your dual form SVM code to operate with kernels. Do the implementation as generally as possible, so that it either takes the kernel function or kernel matrix as input. Test for values of $C = \{0.01, 0.1, 1, 10, 100\}$ and for the Gaussian kernel with some varying bandwidths. Report your result and answer the following questions:

   (a) What happens to the geometric margin $1/||\mathbf{w}||$ as C increases? Will this always happen as we increase C?

   (b) What happens to the number of support vectors as C increases?

   (c) The value of C will typically change the resulting classifier and therefore also affects the accuracy on test examples. Why would maximizing the geometric margin $1/||\mathbf{w}||$ on the training set not be an appropriate criterion for selecting C? Is there an alternative criterion that we could use for this purpose?

Table 1: Titanic Dataset Information

Column Number	Value	Meaning
1-3	One of Many	Passenger class (1 = 1st, 2 = 2nd, 3 = 3rd)
4	Binary	Sex
5	Real	Age
6	Real	Number of Siblings/Spouses Aboard
7	Real	Number of Parents/Children Aboard
8	Real	Passenger Fare
9-11	One of Many	Port of Embarkation
		(9 = Southampton, 10 = Cherbourg, 11 = Queenstown)
12	Binary	Survival (-1 = No, +1 = Yes)

3 Titanic Data

In this section we will now use the binary classifiers from the previous sections to make predictions for a more realistic dataset. The data involves separating passengers on the Titanic, which famously sunk in 1912, into survivors and those who were lost at sea. This data is drawn from Kaggle competition[1], and we have done some preprocessing work for you[2].

The dataset consists of 889 rows and 12 columns. Each row corresponds to one passenger's information, and each column corresponds to different information for passengers. While the last column of data, which is ± 1 valued, indicates whether this passenger has survived or not, other columns specifies information like passenger class, fare, age, etc. Complete description for each columns could be found in Table 1. We have split the data into train, validation and test sets for use in classification.

1. Show the performance of your Logistic Regression classifier on the Titanic data. Feature scaling may help you here.

2. Show the performance of your SVM classifier (with a linear kernel) on the Titanic data.

3. Compare and contrast the results of your classifiers. In particular, what features are the most significant indication of survival for each? Are they the same? (Recall that, even for the dual form of the SVM, the weight vector can be indirectly calculated from the α's.)

[1] https://www.kaggle.com/c/titanic

[2] Though the data set that is given with the competition is already processed, we have gone a step further and made it easier to use. You can also add some other preprocessing (e.g., feature scaling) in order to obtain better performances. In the original dataset, there are a couple of columns (like the passenger name or their ticket code), which we do not expect you to use.