6.867: Homework 2 and 3

Overview

This assignment combines homework assignments 2 and 3. You will get experience with two popular methods for classification: logistic regression (LR) and support vector machines (SVM).

We will be following the same process that we did for Homework 1 (more logistics are in the last pages of this handout).

- This assignment may be done by pairs of students.
- You submit your paper via the Easy Chair site by 11:59 PM on **Thursday November 6**.
- Reviews must be entered on the Easy Chair site by 11:59PM on **Thursday, November 13**.
- 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.

1 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 will take 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` on 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 2D datasets in the `data` folder: `data_bigOverlap`, `data_smallOverlap` `data_ls` and `data_nonsep`. 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. Recall that the dual form SVM can 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 it as generically 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?

2 Logistic Regression (LR)

We are going to be extending logistic regression to work with kernels, so we will be interested in finding sparse set of weights.

To be consistent with SVMs, we will relabel all the data with $y^{(i)} \in \{-1, 1\}$, and use the form of the objective function:

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

1. Derive the kernelized form of the logistic regression objective, with the assumption that $W = X^T \alpha$. You do not need to include a regularizer.

2. Write code to optimize the logistic regression objective, but with L1 regularization on the α values (or using a smoother penalty of $\sqrt{\|\alpha\|^2 + \epsilon}$ for small ϵ). This objective function is not exactly the dual of a regularized version of the primal logistic regression: but it is a sensible optimization criterion if we seek a logistic regression solution in a high-dimensional space with sparsity on the number of "support points" (that is, training examples with α_i not equal to 0).

   Use your gradient descent implementation from HW1 (or a professionally implemented one) to find optimal values of α_i.

3. Test your implementation on the 2D data sets. Vary λ and see how the sparsity is affected.

4. Extend your implementation to use kernels. Report performance as a function of λ and bandwidth of the Gaussian kernel.

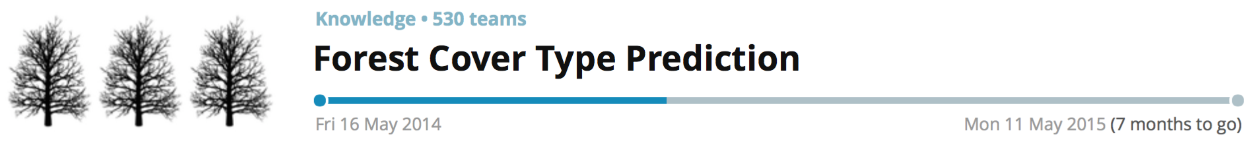
5. Compare the SVM and LR solutions in terms of sparsity. Do they tend to select the same "support vectors"?

3 Multi-label Classification

We will now extend the binary classifiers from the previous sections to make predictions for problems with more than two classes.

1. 1. Multi-label LR:
      Read and implement `read/multi_lr.pdf` (excerpt from Bishop's book).
   2. Multi-label SVM:
      Read and implement the one-versus-the-rest approach in `read/multi_svm.pdf` (excerpt from Bishop's book).

2. With the multi-label LR or SVM written for previous problems, you are now ready to enter the arena to solve real world AI problems! Check out a Kaggle challenge: `https://www.kaggle.com/c/forest-cover-type-prediction`



Knowledge • 530 teams

Forest Cover Type Prediction

Fri 16 May 2014 Mon 11 May 2015 **(7 months to go)**

Given a 54-dimension feature vector from a 30mx30m area, you need to predict the forest cover type out of 7 choices. More details are on the data description page.

1. We provide the training data in `data/kaggle_train.csv`. Make your own training/validation/test set, and compare the performance of your SVM and logistic regression methods on this data.
2. **Optional Competition, for Fun and Glory (but not grades)** Register and download the test data from the competition website. Make a choice for validation and report/explain your result on training/validation/test data. In addition, you are welcome to try out any off-the-shelf packages and report anything you find interesting. To make it a class party, name your team on Kaggle as “6867_x" (replace "x" with your choice).

Grading process (Same as Homework 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 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 10 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 three 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 assigenemnt 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.