EE379K
Spring 2016
Dr. Sriram Vishwanath

Lab 3 – SVMs and Boosting on Scikit-Learn

Assigned: 3/10/16 **Due: 3/24/16 by 11:59pm**

This lab will get you familiarized with Scikit-Learn, which is an extremely powerful Python machine learning library.

Setup

Go into your Git directory and do a "git pull" to get the new materials for this assignment. You should see a "lab3" directory if the pull was successful. If you do not see it, contact me.

For this lab you will be using Scikit-Learn, which is a machine learning library for Python. To install it, open a terminal and type in these three commands in succession:

```
sudo apt-get install python-numpy

sudo apt-get install python-scipy

sudo pip install -U scikit-learn
```

Classification on the Census Income Dataset

In the lab3 folder, there is a folder called "dataset" that contains the census income dataset. The census income dataset is a labeled dataset which tells you whether or not a person has a salary over $50,000. There are three files in the "dataset" folder:

- **dataset_descriptor** – this file will tell you how to parse the dataset.
- **train.data** – this is the training dataset.
- **test.data** – this is the testing dataset.

Your job: Train classifiers on the training dataset to minimize error on the testing dataset. That is, given a person's attributes, your classifier must be able to predict whether or not that person makes over $50,000 a year.

Your first step will be to clean the dataset. That is, you will need to read in the data, parse it, and deal with the missing attributes (some people don't have all the attributes. These are marked as "?" in the dataset).

In your report: explain how you dealt with the missing attributes.

Part 1: SVMs

Now, you split your training dataset into a train and validate set. **Note that the test set is NOT the validation set!** The validation set is simply a subset of the training data. The test set is only for your final testing, when you are confident that your model is correct.

You may use any method to split your training data into a train and validation set. You may use something as simple as the **train_test_split** function provided by scikit-learn, or you may use more advanced methods. The sizing of the split is also up to you (try 40/60 as a baseline). Look here for information on how to use these cross-validation techniques: http://scikit-learn.org/stable/modules/cross_validation.html

Now, train an SVM on the training dataset. Look here for documentation: http://scikit-learn.org/stable/modules/svm.html Please stick to the SVC implementation (do not use NuSVC or LinearSVC). Try tweaking the parameters of the SVM and see how they affect your error rates on the validation set. Again, do NOT run your model on the test set yet!

In your report, answer the following questions:

- How did you split your data? If you did a standard percentage split, which percentages did you try (you should try at least three), and which split gave you the best accuracy on the validation set?
- Try using three different kernels (linear, polynomial, RBF). What is the accuracy for each?
- There are many more parameters that can be tweaked. Pick **three** others besides the kernel and for each parameter, describe what that parameter does and show how changing that parameter affects accuracy. This can be done by either:
  - Plotting a graph of the parameters value vs accuracy **OR**
  - If the parameter is a Boolean, as a simple table.

Now, keep tweaking the parameters until your validation accuracy is as high as you think it will get. Keep in mind that you can also remove features from your dataset! In many cases, too many features can degrade the performance of your classifier. Then, run your model on the testing dataset.

In your report, answer the following questions:

- What were the final parameters you used for your model?
- Did you remove any features? If so, which ones and why? If not, why not?
- With these parameters, what was the accuracy on the validation set?
- With the same parameters, what was the accuracy on the test set?

Part 2: Boosting

Now, try using the AdaBoost classifier to fit your training data. The documentation is here: http://scikit-learn.org/stable/modules/ensemble.html#adaboost

In your report, answer the following questions:

- Plot the validation set accuracy as the number of estimators changes. Plot accuracy for the following number of estimators: 10, 20, 30, 40, 50, 100, 200, 400)
- Plot the validation set accuracy as the learning rate changes. Try at least 10 different rates.

As in part 1, tweak parameters until your validation set accuracy is as high as you think it will get. Then, run your classifier on the test set.

In your report, answer the following questions:

- What were the final parameters you used for your model?
- With these parameters, what was the accuracy on the validation set?
- With the same parameters, what was the accuracy on the test set?