Machine Learning Assignment 3

March 23, 2015

Introduction

This assignment has 2 parts. cs725asgn3.zip has everything you need to start working.

Extract it to a directory of your choice. cs725asgn3 contains 2 folders, one pertaining to each of the parts.

This assignment will require the use of MATLAB and C programming.

We will be running your submissions through a plagiarism detector. Please do not copy or discuss your solutions among yourselves.

1 Visualizing different kernels

Change your working directory to the folder vikern [1].

1.1 Definitions

The Gaussian RBF kernel is defined as follows:

$K(x^{(i)}, x^{(j)}) = \phi(x^{(i)})^T \phi(x^{(j)})$

$= exp(-\gamma||x(i) - x(j)||), \gamma > 0$

1.2 Linear kernel vs RBF kernel

The first part involves playing with differnet kernels and observing the changes the decision boundary in different setups.

We will use libsvm for this demo.

First load the training dataset:

```
>> [train_labels_nonlin, train_features_nonlin] = libsvmread('ex8a.txt');
```

[1] This part is based on `http://openclassroom.stanford.edu/MainFolder/DocumentPage.php?course=MachineLearning&doc=exercises/ex8/ex8.html`

and then plot it:

```
>> plottrainingdata(train_labels_nonlin, train_features_nonlin)
```

As you should note, the dataset is highly non linear and thus cannot be separated by a linear decision boundary.

For comparison, let us first train the model using the linear kernel.

```
>> linear_model = svmtrain(train_labels_nonlin, train_features_nonlin, '-t 1');
```

Plot the boundary using:

```
>> plotboundary(train_labels_nonlin, train_features_nonlin, linear_model);
```

You should see that the decision boundary is far from satisfactory. Next, train the model using an rbf (gaussian) kernel with gamma set to 100

```
>> gaussian_model = svmtrain(train_labels_nonlin, train_features_nonlin, '-t 2 -g 100
```

And plot the decision boundary:

```
>> plotboundary(train_labels_nonlin, train_features_nonlin, gaussian_model);
```

1.3 Affect of γ on the decision boundary

In this part, we will observe how changing γ as defined in the definition of the rbf kernel above changes the decision boundary.

As usual, we start by loading the dataset.

```
>> [train_labels_lin, train_features_lin] = libsvmread('ex8b.txt');
```

We next plot the training data, note that this dataset is actually separable.

```
>> plottrainingdata(train_labels_lin, train_features_lin)
```

Try training the dataset with different values of γ using the rbf kernel. You should note that as γ increases, the accuracy tends to 100%. Generate 6 plots, for $\gamma \in \{1, 4, 16, 64, 256, 1024\}$. Call them gamma_val.png for different vals.

Save the 6 plots in the directory vikern.

2 Struct SVM vs. One vs all

The folder you will use for this part is **ssvm**.

The struct SVM formulation was discussed in the class. In this part, you will complete an implementation of struct svm for multiclass classification. You will then compare your completed implementation with libsvm as discussed in part 1.

2.1 Struct svm

2.1.1 Implementation

The code file that you need to complete is *svm_struct_api.c*. Details about the different data structures used can be located in svm_struct_api_types.h and svm_light/svm_commons.h. The snippets that you have to complete are labeled with a **TODO** comment. The only file you have to change is svm_struct_api.c. All the other files are complete.

Once you are done completing the snippet, compile the codebase by typing *make*.

Next train a model using the implementation that you have just completed for the example training data:

```
./svm_multiclass_learn -c 5000 example4/train.dat example4/model
```

Finally, test the model using:

```
./svm_multiclass_classify example4/test.dat example4/model
```

2.2 Co-ordinate descent Algorithm

2.2.1 Implementation

Code up the co-ordinate descent algorithm outlined in lecture for SVM (hinge loss). The exact algorithm is given in 'algorithm 1' in `http://www.csie.ntu.edu.tw/~cjlin/papers/cddual.pdf`. Don't forget to add appropriate documentation.

2.2.2 Verification

- Compare the objective values and accuracy obtained with liblinear (`http://www.csie.ntu.edu.tw/~cjlin/liblinear/`) and your implementation code for the binary classification dataset at `http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary/diabetes`,

with C = 5000 and $\gamma = 100$. Compare the accuracy and objective values in both the cases, add the results to a file called ver_bin.txt.

- Using your code, train a one-vs-rest multiclass classifier on the train files in example4 folder and compare it with liblinear one-vs-rest variant (*0 – L2-regularized logistic regression (primal)*)

  – liblinear multiclass variant

  – struct svm implementation completed by you

  Add the accuracy (on example4/test) and objective values in ver_struct_one_all.txt.

- Compare the struct svm implementation completed by you above with liblinear multiclass variant 4 (*4 – support vector classification by Crammer and Singer*) variant. Add the accuracy (on example4/test) and objective values in ver_struct_multiclass.txt.

Unless otherwise specified, use the default values for all the (hyper) parameters. Add ver_struct_one_all.txt, ver_struct_multiclass.txt and ver_bin.txt to the folder,rename it to your_roll_number_ssvm and zip. **This zip forms one of the parts of your submission**.

3 Submission

Rename the folder to cs725asgn3_your_rollnumber, zip and submit. Here is a checklist of the artifacts a complete submission will add to the original folder:

- 6 plots for 1.3
- Relevant code and documentation for 2.2.1
- ver_bin.txt for 2.2.2
- ver_struct_one_all.txt for 2.2.2
- ver_struct_multiclass.txt for 2.2.2