Advanced computational methods - Problem set 4

Hrvoje Stojic

January 29, 2016

1. Develop your own function for kNN. You can use mine from the handout as a starting point if you wish. I expect the following:

   - Function should have the following name: **kNN**.

   - It should be stored in a file **kNN.R**

   - Your function should have at least these four arguments: **features** (dataframe or matrix of feature values), **labels** (vector of labels), **k** and **p** parameters. Please use exactly these names. Of course, you can add more arguments if you need to.

   - Verify that inputs are in correct format (you can use `assertthat` package if you want).

   - Output should be a **named** list with at least these two elements - **predLabels** (vector of predicted labels), **prob** (vector of probabilities for predicted labels). Please use exactly these names.

   - Function should be able to deal with multiple classes - I will check this by testing it on multi-class data.

   - Do your best to come up with an efficient solution (something better than what we used in the class). As an additional incentive, I will time the execution on several different datasets, and the winner gets the bonus points.

   - Finally, select one function for generating datasets that you can find in `genData.R` script in `Box/problemSets` folder. From your submissions for Problem set 1 I selected several datasets that were interesting and created this file, with standardized inputs and outputs. Select a two dimensional dataset. Apply your newly minted kNN function on dataset generated with one of these functions and produce two files: (1) `predictions.csv` file with the original dataset and two additional columns - predLabels and prob, (2) `plot.pdf` with data and decision boundaries illustrated. Note that boundaries are nonlinear, check `?contour` function in base or `?stat_contour` in `ggplot2` package. Put this code in file named **kNN_genData.R**

2. In the `Box/datasets/MNIST` folder you will find two files, one called `MNIST_train.csv` and another called `MNIST_test.csv`. This dataset is a subset of a famous MNIST database of handwritten digits. Training file is a dataset with 5999 observations of handwritten digits. Each observation is 16x16 pixel gray-scale image which is represented in a single row as 256 features. Each of 256 pixels is represented by a floating point number indicating the gray-scale intensity at that location in the image. True label is at a first position in a row, it identifies which digit is represented in a row with an integer from 0 to 9. Test set has 1500 observations and contains only features, not labels. You can use displayDigit.R functions to display the digits. I expect the following:

   - You should use the `MNIST_train.csv` to train the kNN classifier and make predictions for the hold out sample in the `MNIST_test.csv`. You do not need to use your own function, you can use one of the packages if you want to. Be careful how you optimize the parameters k and p.

   - Your code should save a csv file named **MNIST_predictions.csv** with a single column - predicted labels for each observation in the `MNIST_test.csv` file. I will check it against the true labels and let you know the accuracy.

   - YOur code should be saved in a file under the following name: **kNN_MNIST.R**

Important notes

- Deadline is **February 5, at 12h**.

- Recall that you have to keep your code under version control in a repository at Github, I will pull the content of the repository at the designated time. Place all the files for this problem set in a folder in this repo under the name **PS4**.

- Create a simple text file with the name `Readme.md` in the problem set folder, if you want to leave me any instructions or messages regarding the problem set.

- Please, pay attention to the names of the files I instruct you to use. Names facilitate examining the code greatly, there is 30 of you and it takes me far more time if everybody produces different set of files with different names.

- I will give extra points for extra nice solutions!