Final Project
Due on Thursday, December 8, 2016 at 11:55 pm

How to Submit

Create one zip file (.zip) of your project and submit it via `ilearn.ucr.edu`. This zip file should have the following:

- Your code to run your project
- A `README.txt` file explaining how to run your code, with a clear example of how to run it on new data and report the accuracy.
- A file titled `project.pdf` that is your 3-page (no more!) write-up of your project

Your `README.txt` and `project.pdf` should contain

- Your name
- Your UCR student ID number
- The date
- The course (CS 229)

Handwritting recognition

For your final project, you will build a handwriting recognizer. The supplied file `handwriting.data` has one handwritten character per line. Each line has, in order, the following fields (separated by a single space):

- Letter (that is, the true label), encoded as a number from 0 to 25.
- Whether this letter is at the end of the word
- 128 binary features, corresponding to the scanned pixels of the image as a 16 (high) and 8 (wide) binary image.

The letters are in word order and you can expect your classifier to be used similarly (that is, it would be given a sequence of characters that represent a word). The first letter of the word has been omitted (it was a capital letter and would have made it harder).

Task

You need to

- Build a classifier
- Understand how to make it better
- Improve the classifier
- Provide code that can check its performance on other data (in the same file format)
- Write a 3-page (no more, no less) report describing your method and why you used this method

I expect the report to take the following format:

1. Approach to the Problem
2. Initial Results *and Analysis*
3. Description of Improvement
4. Final Results *and Analysis*

We will supply your code with a file in the same format at `handwritting.data`, but containing *testing* examples. Your README needs to explain how to easily run your method on this new data. While the README should also explain how to retrain the classifier (from the training data), we should *not* have to perform learning again in order to test your classifier's accuracy.

Grading
You will be graded according to the following criteria

5 points Performance on data that has been held out
5 points Clarity of your report and proper citations
5 points Initial approach
10 points Analysis of initial approach
5 points Improvement based on initial analysis
5 points Final approach
5 points Final analysis

Most critical to your report is that you do not just report results. Rather you need to carefully consider what experiments to run to make a constructive analysis of *why* your results are as they are and *how* your initial performance could be improved. This should be followed by such an improvement and a similar analysis at the end. The final analysis should make it clear what steps should be taken next (were the project to continue).

A poor classifier with a report that clearly analyzes why it is not doing well will receive a far better grade than a good classifier that has a poor analysis.

Report Format
While the content (see above) is most important, here are formatting instructions you need to follow

- Absolutely no more than 3 pages (no fewer either)
- 11 or 12 pt font with at least 1 inch margins
- pdf file format

You should *not* explain machine learning methods. I already know them. You *must* explain your approach, which includes not only your machine learning algorithm, but also your testing method.

You do not need to cite methods we covered in class. However, if you used any external software packages or research papers, cite them briefly.

Programming
You may use external implementations of methods. However, I need to be able to run your code on a Linux box. I would recommend using `Matlab`, `R`, or `Python`. If you have questions about this, ask me, and I am sure we can work something out.

I would recommend strongly *against* using Weka. The package is well known but has many problems.

If you are using `Matlab`, the following code will load the data

```
load -ascii handwriting.data;
Y = handwriting(:,1);
X = handwriting(:,2:end);
```

To look at the image associated with data point i, the following will work

```
imagesc(reshape(X(i,2:end),[8 16])');
colormap (1.0 - gray);
axis equal;
```