CS5487 – Assignment 3 – Course Project

Antoni Chan
Department of Computer Science
City University of Hong Kong

Proposal due date: Fri, Week 10
Presentation date: TBA, Week 14
Report due date: Fri, Week 14

1 Course Project

The final assignment is a student-defined course project. The goal of the project is to get some hands-on experience using the course material on your own research problems. If you can't think of a project, then you can do the "default" project, which is digit classification (see Section 2).

1.1 Project topic

The goal of the project is to get some hands-on experience using the course material on your own research problems. Keep in mind that there will only be about 4 weeks to do the project, so the scope should not be too large. Following the major themes of the course, here are some general topics for the project:

- *regression* (supervised learning) – use regression methods (e.g. ridge regression, Gaussian processes) to model data or predict from data.
- *classification* (supervised learning) – use classification methods (e.g., SVM, BDR, Logistic Regression) to learn to distinguish between multiple classes given a feature vector.
- *clustering* (unsupervised learning) – use clustering methods (e.g., K-means, EM, Mean-Shift) to discover the natural groups in data.
- *visualization* (unsupervised learning) – use dimensionality reduction methods (e.g., PCA, kernel-PCA, non-linear embedding) to visualize the structure of high-dimensional data.

You can pick any one of these topics and apply them to your own problem/data. Before actually doing the project, you need to write a *project proposal* so that we can make sure the project is doable within the 3-4 weeks. I can also give you some pointers to relevant methods, if necessary.

- *Can my project be my recently submitted or soon-to-be submitted paper?* If you plan to just turn in the results from your paper, then the answer is **no**. The project cannot be be work that you have already done. However, your course project can be based on extending your work. For example, you can try some models introduced in the course on your data/problem.

1.2 Project details

- **Group project** – *all projects* should have a group of 2 students. To sign up for a group, go to Canvas ⇒ "People" and then join one of the existing "Project Groups". If you cannot find a group, please use the Discussion board.

- **Project Proposal** – For the first part of the project, you need to write a project proposal. The project proposal should be at most one page with the following contents: 1) an introduction that briefly states the problem; 2) a precise description of what you plan to do – e.g., What types of features do you plan to use? What algorithms do you plan to use? What dataset will you use? How will you evaluate your results? How do you define a good outcome for the project? – The goal of the proposal is to work out, in your head, what your project will be. Once the proposal is done, it is just a matter of implementation!

- **Project Poster Presentation** – Project poster presentations will be at the end of the semester, before the project due date (Week 14). More details will be sent out later. The poster presentation is optional. However, if you want to get an "A" on your project, then you *must* give a presentation. Put it another way, if you don't give a presentation, then you will get at most a "B+" on your project.

- **Project Report** – The project report is essentially the project proposal with all the details filled in. The report should have the following contents: 1) introduction – what is the problem? why is important?; 2) methodology – what algorithms did you use and what are the technical details? what are the advantages and disadvantages?; 3) experimental setup – what data did you use? how did you pre-process the data? which algorithms did you run on the data? what is the metric for evaluation?; 4) experimental results – what were the results? what insight do you get from these results? what are some typical success and failure cases? – The project report should be at least 4 pages. There is no upper page limit, but probably it should not be more than 8 pages long. *For group projects, the project report must state the level of contribution from each project member.*

- **What to hand in** – You need to turn in the following things:

  1. Project proposal (**due Friday, Week 10**).
  2. Project report (**due Friday, Week 14**).
  3. Presentation poster (**due Friday, Week 14**).
  4. Source code files (**due Friday, Week 14**).

  Only one group member needs to submit the files on Canvas. *You must submit your course project materials using the Canvas website.* Go to "Assignments" ⇒ "Course Project" ⇒ select the appropriate entry.

- **Third Party Code** – In the course project, you may use 3rd party source code, e.g., libsvm, etc. If you use 3rd party code, you must acknowledge it with an appropriate reference.

- **Grading** – The marks for this project will be distributed as follows:

  – 16.7% – Project proposal.
  – 16.7% – Technical correctness (whether you used the algorithms correctly)

- 16.7% – Experiments. More points for thoroughness and testing interesting cases (e.g., different parameter settings).
- 16.7% – Analysis of the experiments. More points for insightful observations and analysis.
- 16.7% – Quality of the written report (organized, complete descriptions, etc).
- 16.7% – Project poster presentation.

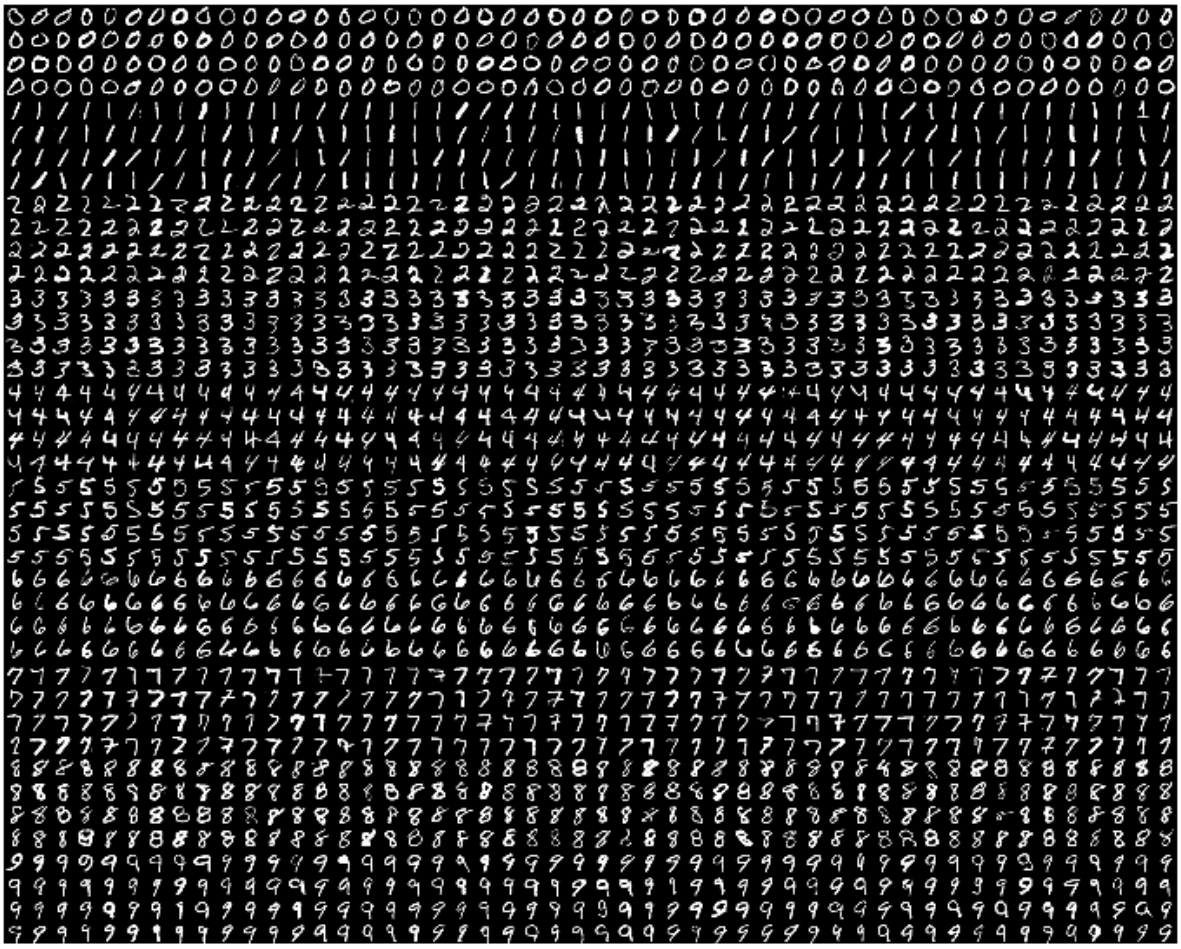
Note: Here 16.7% means 5/30

---

2 Default Course Project – Digit Classification

The default project is handwritten digit classification on a subset of the MNIST digits dataset.

- **Dataset** – The provided dataset is a subset of the MNIST digits. The dataset has 10 classes (digits 0 through 9) with 4000 images (400 images per class). Each feature vector is a vectorized image (784 dimensions), containing grayscale values $[0, 255]$. The original image dimensions are 28×28. Here is an example montage of the digits:



  The MATLAB file `digits4000.mat` (or `digits4000_*.txt` for non-MATLAB users) contains the following data:

  - `digits_vec` – a 784×4000 matrix, where each column is a vectorized image, i.e. the feature vector $x_i \in \mathbb{R}^{784}$.
  - `digits_labels` – a 1×4000 matrix with the corresponding labels $y_i \in \{0, \cdots, 9\}$.
  - `trainset` – a 2×2000 matrix, where each row is a set of indices to be used for training the classifier.
  - `testset` – a 2×2000 matrix, where each row is the corresponding set of indices to be used for testing the classifier.

  The image above was generated with the following MATLAB code:

```
testX = digits_vec(:,testset(1,:));        % get test data (trial 1)
testXimg = reshape(testX, [28 28 1 2000]); % turn into an image sequence
montage(uint8(testXimg), 'size', [40 50]); % view as a montage
```

- **Methodology** – You can use any technique from the course material, e.g., Bayes classifiers, Fisher's Discriminant, SVMs, logistic regression, perceptron, kernel functions, etc. You may also use other classification techniques not learned in class, but you will need to describe them in detail in your report. Two useful libraries for classification are "libsvm" and "liblinear". You can also pre-process the feature vectors, e.g., using PCA or kPCA to reduce the dimension, or apply other processing techniques (e.g., normalization or some image processing).

  Finally, a common trick for doing multi-class classification using only binary classifiers (e.g. SVMs) is to use a set of *1-vs-all* binary classifiers. Each binary classifier is trained to distinguish one digit (+1) vs. the rest of the digits (-1). In this case, there are 10 binary classifiers total. Given a test example, each binary classifier makes a prediction. Hopefully, only one classifier has a positive prediction, which can then be selected as the class. If not, then the classifier that has the most confidence in its prediction is selected. For example, for SVMs the classifier that places the test example furthest from the margin would be selected. For logistic regression, the selection would be based on the calculated class probability.

- **Evaluation** – The classifiers are evaluated over 2 experiment trials. In each trial, 50% of the data has been set aside for training (and cross-validation of parameters), and the remaining 50% is held out for testing only. The indices of the training set and test sets are given in the `trainset` and `testset` matrices. For a given trial, the same writer does not appear in both the training and test sets.

  For each trial, train a classifier using only the training set data (images and labels). You may also use the training set to select the optimal model parameters using cross-validation. After training the classifier, apply the classifier to the test data (images only) to predict the class. Record the accuracy (number correct predictions / total number) for that trial. *Do not tune the parameters to optimize the test accuracy directly!* You can only tune the parameters using the training set.

  As a baseline, a simple nearest-neighbors classifier with Euclidean distance was used on the test data. The resulting classification accuracy for each experiment trial is:

trial	1	2	mean (std)
1-NN	0.9135	0.9185	**0.9160 (0.0035)**

  In your experiments, which classifier does better? What feature pre-processing helps or hurts the performance? How does the performance vary with parameter values?

- **Bonus Challenge** – In the bonus challenge, I will give you a new test set containing my own handwritten digits, and you will try to classify them using your trained classifiers. Whoever gets the best performance wins a prize!