Inf2b - Learning & Data

Coursework 2: Q/A session

Heru Praptono

School of Informatics
University of Edinburgh

March, 2018

Outline

Dataset

Dataset: Description

Coursework Tasks

Task 1: k-NN Classification (contains 4 subtasks)

Task 2: Bernoulli naive Bayes classification (contains 3 subtasks)

Task 3: Bayes classification with Gaussian distributions (contains 3 subtasks)

Notes

Q/A

Outline

Dataset Description

- The coursework employs the EMNIST handwritten character data set, see: https://www.nist.gov/itl/iad/image-group/emnist-dataset
- 28-by-28 pixels in grayscale (*range* [0,255]).
- stored as a row vector of 784 elements ($28 \; x \; 28 = 784$)
- see the dataset summary on there, this coursework utilises the subset of it.

Dataset Description

- **This coursework:** English alphabet of 26 letters in either upper case or lower case
- stored in a Matlab file named 'data.mat', see: /afs/inf.ed.ac.uk/group/teaching/inf2b/cwk2/d/UUN/data.mat
- you can even download it through browser by accessing ifile: https://ifile.inf.ed.ac.uk
- Contains 1800 training samples and 300 test samples for *each class*.

Dataset Description

- Contains 1800 training samples and 300 test samples for *each class*.

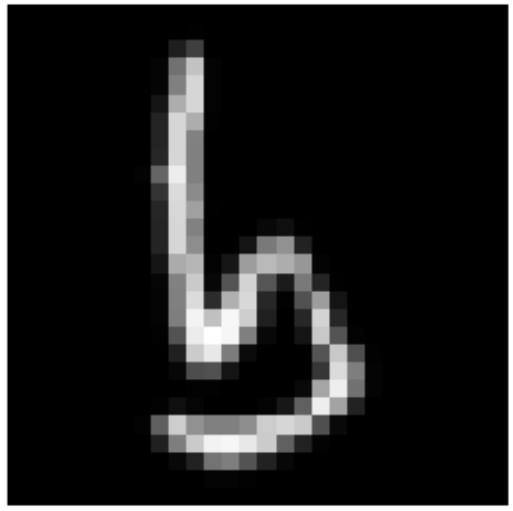
Name	Size (Class)	Description
dataset.train.images	46800x784 (uint8)	training samples
dataset.train.labels	46800x1 (double)	class labels of training samples
dataset.test.images	7800x784 (uint8)	test samples
dataset.test.labels	7800x1 (double)	class labels of test samples

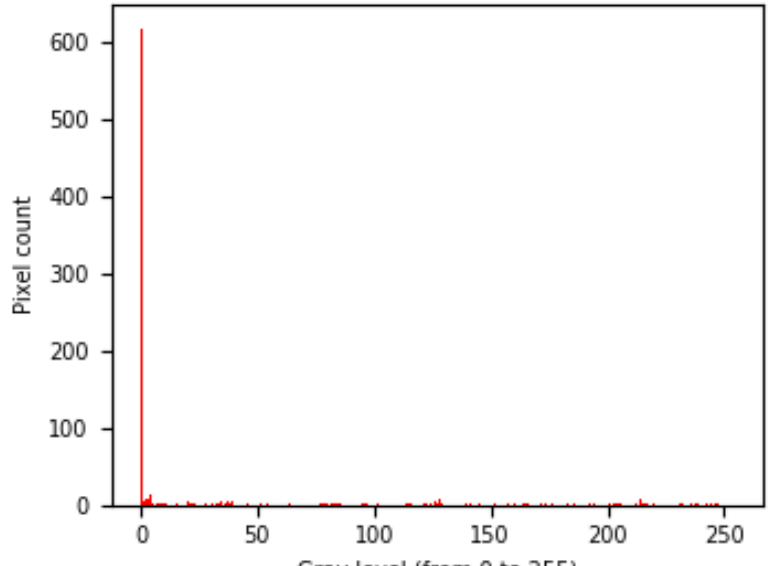
- convert image's data type from unsigned bit integer into double, to enable some sort of calculations on it.
- For task 1 and task 3, sample's number should be converted into [0, 1] range.

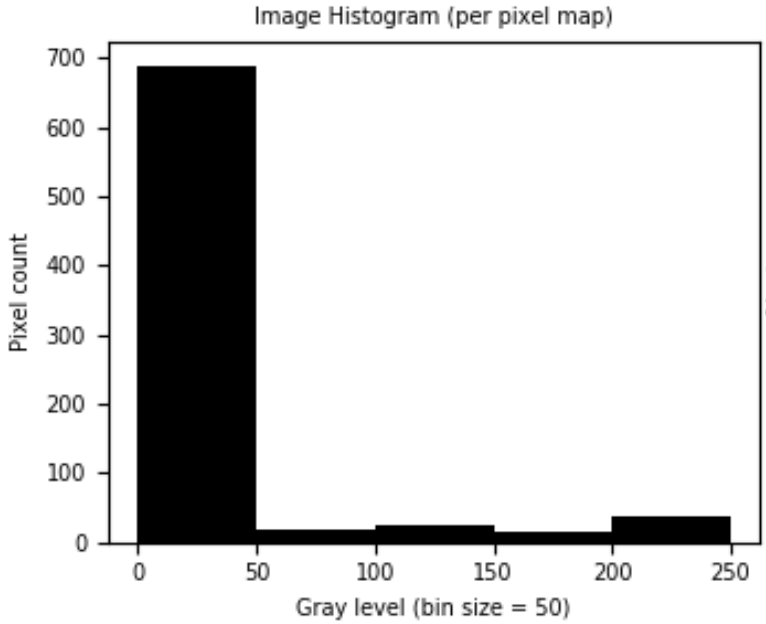
Dataset Description

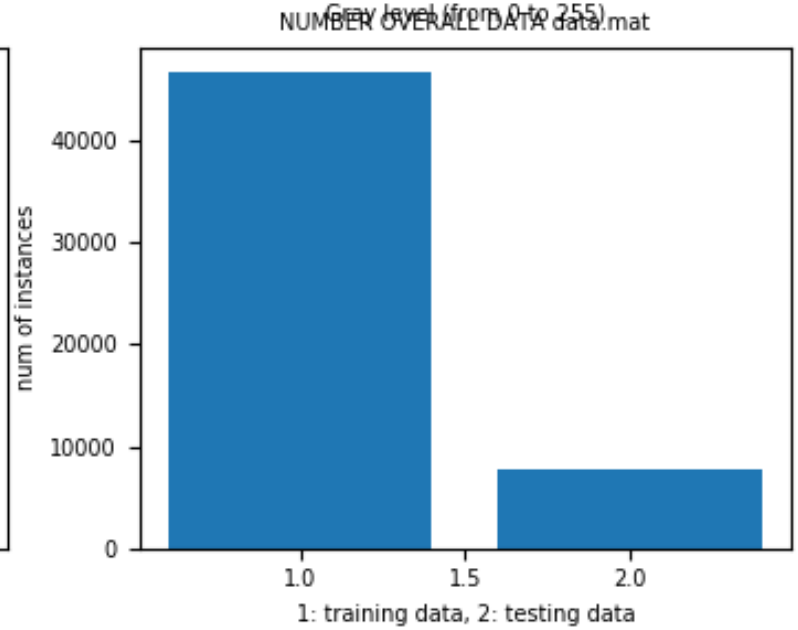
- how does the data look like?
  run: `dispImage('data.mat',1,200)`.

Dataset Description









Outline

Task 1: k-NN Classification

- Implement k-NNs.
- Enables you to familiarise with linear algebra in practice, by converting loop with matrix/vector operation $--> $ vectorisation.

Task 1: k-NN Classification

Task 1.1: Write function for k-NN

- `[Cpreds] = my_knn_classify(Xtrn, Ctrn, Xtst, Ks)`
- to note: `Ks` L-by-1 vector, `Cpreds` N-by-L matrix. We just need to calculate distance once.
- In case of ties (where there is more than one majority group), choose the smallest index (class label).

Task 1: k-NN Classification

Task 1.2: Write function for building the confusion matrix

- This will also be used for subsequent (sub)tasks.
- `[CM, acc] = my_confusion(Ctrues, Cpreds)`
- `acc` falls within range of [0,1]

Task 1: k-NN Classification

Task 1.3: Write function for demonstrating your k-NN

Write `my_knn_system` that enables:

- Loads the data set.
- Run my_knn_classify with ks = [1,3,5,10,20].
- Measures time (in seconds), and displays it, to standard output
- Saves the confusion matrix for each k, to a matrix variable `cm[k].mat`
- displays [k, N, Nerrs, acc] to the standard output

Run the script and report the user time taken and result shown on the display. The experimental result should be shown in a table.

Task 1: k-NN Classification

Task 1.4: Write a report

In your report, explain your implementation of k-NN classification in terms of speeding up, using mathematical expressions if possible. For example, nested loops are required for the algorithm to calculate distance for possible pairs of training samples and test samples, but the loop operation can be avoided or the number of loops can be reduced with vectorisation techniques

Outline

Task 2: Bernoulli naive Bayes classification

Task 2.1: Write function for BnB

- `[Cpreds] = my_bnb_classify(Xtrn, Ctrn, Xtst, threshold)`
- `threshold` is used for binarisation

Task 2: Bernoulli naive Bayes classification

Task 2.2: Write function for demonstrating your BnB

Write `my_bnb_system.m` that enables:

- Loads the data set
- Run `my bnb classify`
- Measures time (in seconds), and displays it, to standard output
- Obtains the confusion matrix using `my_counfusion()`, stores the confusion matrix to a matrix variable cm, and saves it with the file name `'Task2/cm.mat'`.
- displays [N, Nerrs, acc] to the standard output

Run it, and report the result in your report using a table that shows the information of user time taken, N, Nerrs, and acc.

Task 2: Bernoulli naive Bayes classification

Task 2.2: Write report

Write a report, investigating the effect of (different) thresholds on classification accuracy.

Outline

Task 3: Bayes classification with Gaussian distributions

Task 3.1: Write function for classification with a single Gaussian dist. per class

`[Cpreds, Ms, Covs] = my_gaussian_classify(Xtrn, Ctrn, Xtst, epsilon)`

To note, here,

- Ms: D x K
- Covs: D x D x K

Task 3: Bayes classification with Gaussian distributions

Task 3.2: Write function for demonstrating your Gaussian classifier

Write `my_gaussian_system.m` that enables:

- Loads data set
- calls classification function with $\epsilon = 0.001$ (again, remember that this is to avoid zero determinant).
- Measures the user time taken for the classification experiment, and display the time (in seconds) to the standard output.
- Obtains the confusion matrix, stores it to a matrix variable `cm`, and saves it with the file name `'Task3/cm.mat'`.
- Saves the mean vector and covariance matrix for Class 26, i.e, Ms(:,26) and Covs(:,:,26), to files with the file names `'Task3/m26.mat'` and `'Task3/cov26.mat'`, respectively.
- displays N, Nerrs, and acc

do not forget to report it!

Task 3: Bayes classification with Gaussian distributions

Task 3.3: Modify your Gaussian classifier of Task 3.1, and write report of the investigation

.

Write `my_improved_gaussian_system.m` as an improvement, having some exploration on the modification by utilising e.g. (a) k-means clustering to obtain multiple Gaussian, (b) utilising PCA to reduce the dimension.

- Write a classifier filename, filename `[Cpreds] = my improved gaussian classify(Xtrn, Ctrn, Xtst)`.
- Write a file `my_improved_gaussian_system.m` that demonstrates the classifier with the setting is proportional to Task 3.2,but the cm name should be adjusted, becomes `'cm_improved.mat'`.
- Describe the investigation into the report, clarify the methods, and report the results of experiment. Add the discurssion to the remaining problems, and **further improvement**.

Functions which are allowed and not allowed

See the questionsheet for the detail information about which functions are allowed, which functions are not allowed, and the suggestion for your own function name.
My suggestion: always use vectorisation even on your own functions.

Submission

- Submit work electronically via DICE submit command, **by the deadline**
- The marking for each task will be separately done. Thus, prepare report separately (`report_task1.pdf`,`report_task2.pdf`,`report_task3.pdf`).
- Do not forget to place your UUN and task name prominently at the top of each report.
- Your report should be concise and brief (1 or 2 pages long for each task).
- create directory, `LearnCW` and copy the PDF of your reports in it. Create subtasks directory in it, `Task1`, `Task2`, `Task3` containing your code for each task respectively.
- `submit inf2b cw2 LearnCW`.

Q/A

Q/A?

In addition, : always keep in touch with Piazza for any discussion that is not covered here and (possibly) some updates.