BS1821: Advanced Machine Learning Spring 2018 Imperial College Business School
Martin Haugh

Assignment 3

Due: 9.00pm Tuesday 13^{th} March 2018

Rules

1. This is a group assignment. (There are approximately 3 people per group and by now you should know your assigned group.)
2. While `R` is the default package / programming language for this course you are free to use `R` or `Python` for the programming components of this assignment.
3. Within each group **I strongly encourage each person to attempt each question by his / herself first** before discussing it with other members of the group.
4. Students should **not** consult students in other groups when working on their assignments.
5. Late assignments will **not** be accepted and all assignments must be submitted through the Hub with one assignment submission per group. Your submission should include a PDF report with your answers to each question as well as any relevant code. Make sure your PDF clearly identifies each member of the group by CID and name.

Question 1. Eigen-Faces (40 marks)

Open up the *Eigen_Faces_Fragment* `R` Notebook (posted on the Hub) and familiarise yourself with the Olivetti faces data-set. (The `Matlab` .mat file containing the data is also available on the Hub and the `R` Notebook shows you how to read in the data.) Then use PCA to construct k-dimensional approximations to the data-points. (To be clear, each data-point is a vector $\mathbf{x} \in \mathbb{R}^{4,096}$ corresponding to a particular face.) Some other points you may wish to consider:

- By default the *prcomp* function in `R` first de-means the data. In addition, setting `scale=TRUE` as an argument to *prcomp* ensures that each component of the data has standard deviation 1. (There are 4,096 components or variables in this data-set.) In general it is a good idea to do this!
- The means and standard deviations of the original data are some of the outputs of *prcomp*. You will need to use these when constructing your k-dimensional approximations because the PCA is applied to the de-meaned and standardized data (assuming you set `scale=TRUE`).

Your answer to this question should show plots of the original 4 faces that are currently displayed in the R Notebook in addition to k-dimensional reconstructions of them for $k = 3, 10, 25$ and 50. You should also provide a short mathematical expression for how you construct the approximation. (This will be identical to the approximation provided in the slides except you also need to account appropriately for the mean and standard deviation terms.)

Question 2. Clustering & PCA (Q10.10 from *ISLR*) (60 marks)

In this problem, you will generate simulated data, and then perform PCA and K-means clustering on the data.

(a) Generate a simulated data set with 20 observations in each of three classes (i.e. 60 observations total), and 50 variables. (10 marks)

Hint: There are a number of functions in R that you can use to generate data. One example is the rnorm() function; runif() is another option. Be sure to add a mean shift to the observations in each class so that there are three distinct classes.

(b) Perform PCA on the 60 observations and plot the first two principal component score vectors. Use a different color to indicate the observations in each of the three classes. If the three classes appear separated in this plot, then continue on to part (c). If not, then return to part (a) and modify the simulation so that there is greater separation between the three classes. Do not continue to part (c) until the three classes show at least some separation in the first two principal component score vectors. (10 marks)

(c) Perform K-means clustering of the observations with $K = 3$. How well do the clusters that you obtained in K-means clustering compare to the true class labels? (10 marks)

Hint: You can use the table() function in R to compare the true class labels to the class labels obtained by clustering. Be careful how you interpret the results: K-means clustering will arbitrarily number the clusters, so you cannot simply check whether the true class labels and clustering labels are the same.

(d) Perform K-means clustering with $K = 2$. Describe your results. (5 marks)

(e) Now perform K-means clustering with $K = 4$, and describe your results. (5 marks)

(f) Now perform K-means clustering with $K = 3$ on the first two principal component score vectors, rather than on the raw data. That is, perform K-means clustering on the 60×2 matrix of which the first column is the first principal component score vector, and the second column is the second principal component score vector. Comment on the results. (10 marks)

(g) Using the *scale()* function, perform K-means clustering with $K = 3$ on the data after scaling each variable to have standard deviation one. How do these results compare to those obtained in (b)? Explain. (10 marks)

Question 3. Collaborative Filtering – ENTIRELY OPTIONAL

You do not need to do this question and will not receive any question if you do. I only include it here as the `MovieLens` data-set is a famous machine-learning data-set and you now have the tools to tackle this question! (Maybe you'll try this question after the exams or over the summer when you have nothing better to do :-))

Download the `Excel` workbook *Assignment_MovieData.xlsx*. The workbook contains 100k movie ratings from the `MovieLens` data-set. The data consists of ratings from 1 to 5 from a total of 943 users on 1682 movies. These ratings are split into "train" and "test" work-sheets, respectively. The "test" worksheet contains 9,430 observations with exactly 10 ratings from each user.

(a) Construct the baseline estimator where we use the average rating (across all ratings in the training data), $\bar{\mathbf{x}}$, as our estimator. What is the test error for this estimator? (Here and in the other parts below we mean RMSE when we refer to (test) error.)

(b) Now construct biases for each movie and user according to

$$b_i := \frac{\sum_u x_{ui}}{M_i} - \bar{x} \tag{1}$$

$$b_u := \frac{\sum_i x_{ui}}{M_u} - \bar{x} \tag{2}$$

where $M_i = \#$ users that rated movie i and $M_u = \#$ movies rated by user u. The new baseline estimator is

$$\hat{x}_{ui} = \bar{\mathbf{x}} + b_u + b_i.$$

What is the test error of this estimator?

(c) Repeat part (b) but now use regularization and validation on the test set to choose the biases. That is, solve

$$\min_{b_i, b_u} \sum_{(u,i)} (x_{ui} - \hat{x}_{ui})^2 + \lambda \left(\sum_i b_i^2 + \sum_u b_u^2 \right). \tag{3}$$

where the sum is over observations (u, i) in the training data and choose $\lambda \geq 0$ to be that value which gives the best performance on the test set. What is the test error here? (Note that we are really using the test set as a validation set here and in part (e) below.)

Hint: Note that (3) is an unconstrained concave optimization problem and the first order conditions will be sufficient to find the global optimum. You can check that these first order conditions (for user u and movie i) are:

$$b_u = \frac{\sum_{i:i \text{ rated by } u} (x_{ui} - b_i) - M_u \bar{x}}{\lambda + M_u} \tag{4}$$

$$b_i = \frac{\sum_{u:u \text{ rated } i} (x_{ui} - b_u) - M_i \bar{x}}{\lambda + M_i} \tag{5}$$

Note that (4) and (5) is a system of $M+I$ linear equations in $M+I$ unknowns (where $M = \#$ of movies and $I = \#$ of users) and will have a unique solution for any $\lambda > 0$. You can either solve this system directly or by using (4) and (5) to construct an iterative scheme.

(d) Use your best estimator from parts (a), (b) and (c) to construct the residual matrix, $\tilde{\mathbf{X}}$.

(e) Now use a neighborhood method (as described in the slides) applied to the residual matrix to construct a new estimator of the form

$$\hat{x}_{ui}^{N} = \bar{\mathbf{x}} + b_u + b_i + \frac{\sum_{j \in \mathcal{L}_i} d_{ij}\, \tilde{x}_{uj}}{\sum_{j \in \mathcal{L}_i} |d_{ij}|}$$

where $\mathcal{L}_i$ denotes the neighborhood of movie i and the d_{ij}'s are as defined in the slides. You can choose L, the size of a neighborhood, via validation on the test set. What is error on the test set now?

Remark: You could also try playing around with some of the matrix factorization methods (instead of the neighborhood method) for part (e). (They both yield similar results.)