CS4786/5786: Machine Learning for Data Science, Spring 2015
2/13/2015: Assignment 1: PCA, CCA and Random Projections

Updates

The updated version of this homework is maintained at http://www.cs.cornell.edu/Courses/cs4786/2015sp/assignments/hw1.pdf. Announcements of changes will be made on the course webpage. Announcements of any *crucial* updates will be made via email, using the CMS system.

The changes mentioned below are marked in orange in the following pages as well.

- Feb 18, 6:39pm:
  - Q1, part 3, definition of rotation operator: rewritten to clarify general definition from specific application to this problem.
  - Q1, part 4, specification of `cubist_email.csv`: "column" should be "row".
- Feb 19, 12:09am: Q1, part 4, clarification of the contents of `X_smilie.csv`: the data vectors for *your* non-cubist smileys, one row per picture.
- Feb 26, 9:24pm:
  - Q1, part 2: the (i, j) entries of Y_I and Y_{II} should have the same absolute values, but we've corrected the text to state that they may have different signs.
  - Q1, part 3: you may, for simplicity, assume that the $\mathbf{x}_t$s are centered.
- Feb 27, 6:36pm: Q2, part 2, definition of error rate: the absolute-value bars were incorrectly placed outside the summation. If you solved the problem with using that previous version of the error metric, you don't have to do the problem over; just state in your writeup that you used the previous version.

CS4786/5786: Machine Learning for Data Science, Spring 2015
2/13/2015: Assignment 1: PCA, CCA and Random Projections

Instructions Due at 11:59pm Tuesday March 3 on CMS. Submit what you have at least once by an hour before that deadline, even if you haven't quite added all the finishing touches — CMS allows resubmissions up to, but not after, the deadline. If there is an emergency such that you need an extension, contact the professors.

You may work in groups of one up to four. Each group of two or more people must create a group on CMS well before the deadline (there is both an invitation step and an accept process; make sure both sides of the handshake occur), and submits 1 submission per group. You may choose different groups for different assignments. The choice of the number "four" is intended to reflect the idea of allowing collaboration, but requiring that all group members be able to fit "all together at the whiteboard", and thus all be participating equally at all times. (Admittedly, it will be a tight squeeze around a laptop, but please try.) Please ensure that each member of the group can individually defend or explain your group's submission equally well.

You will submit both a writeup and some datafiles you create. The writeup can be handwritten or typeset, but please make sure it is easily readable either way.

Keep an eye on the course webpage for any announcements or updates.

Academic integrity policy We distinguish between "merely" violating the rules for a given assignment and violating *academic integrity*. To violate the latter is to commit *fraud* by claiming credit for someone else's work. For this assignment, an example of the former would be getting an answer from person X who is not in your CMS-declared group but stating in your homework that X was the source of that particular answer. You would cross the line into fraud if you did not mention X. The worst-case outcome for the former is a grade penalty; the worst-case scenario in the latter is academic-integrity hearing procedures.

The way to avoid violating academic integrity is to always document any portions of work you submit that are due to or influenced by other sources, even if those sources weren't permitted by the rules.[1]

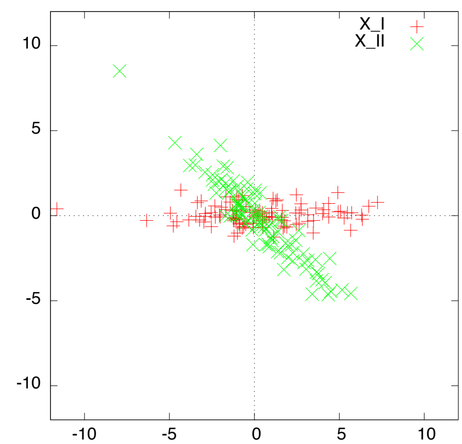
---

*Q1 (***Rotational Invariance of PCA).** In this problem, we shall see that PCA is *rotationally invariant*, and how we can use this property to do some cool stuff, such as deal with seemingly arbitrary reorderings of features (sample real-life setting: when you get data measurements from different sources that didn't check with each other ahead of time). We shall build up to our finale by walking you through various steps.

1. (warmup to make sure you have the right tools available; nothing to turn in) We have provided you with two sets of data points in `2d-gaussian.csv` and `2d-gaussian-rotated.csv`.

[1] We make an exception for sources that can be taken for granted in the instructional setting, namely, the course materials. To minimize documentation effort, we also do not expect you to credit the course staff for ideas you get from them, although it's nice to do so anyway.

Using your favorite programming language or tool, load these files into two data matrices, which we'll refer to as X_I and X_{II} (we're using Roman numerals here in an attempt to avoid notational clashes), and scatter-plot these two set of points on the two- dimensional plane.[2] You should get the following (up to choice of graphical elements like color, style of points, and so on). Notice how the second set of points is simply a rotation of the first set; in that sense, they are two differently-but-relatedly-featurized versions of the same data.



2. Run PCA on X_I and X_{II} to get two corresponding projection matrices W_I and W_{II}. (Refer back to the code posted on the course lectures page for examples of how to do this in various languages.) Recall that the columns of these two matrices are the principal directions, i.e., the eigenvectors of the corresponding covariance matrices. Then, apply the projections W_I and W_{II} to their corresponding data matrices X_I and X_{II} to yield Y_I and Y_{II}.

   Examine W_I and W_{II} and verify that they are different 2×2 matrices. Similarly, examine the two covariance matrices and verify that they are different 2×2 matrices. (Check your understanding by verifying that you understand why the shapes are $2\times$ 2. We can help you with this in office hours.) But then, examine Y_I and Y_{II} (you may wish to use scatterplots to do this, but you don't have to). You should see that ~~they~~ the absolute values of corresponding entries are . . . the same!

   Finally, given the *non*-rotation matrix[3]

$$A = \begin{bmatrix} .7071068 & .7071068 \\ .56 & .1 \end{bmatrix},$$

   run PCA on $X_I A$ to get a third projection matrix W_{III} and corresponding projection Y_{III}. Check to see that Y_{III} is *not* the same as Y_I (you may wish to use scatterplots to do this, but you don't have to).

[2] We plan to provide example scripts during the Thursday Feb 19th lecture.

[3] That is, the transpose of A is *not* A's inverse.

Explain in at most 10 sentences why all the projection matrices and covariance matrices you get are different, and yet $Y_I = Y_{II}$, whereas $Y_I \neq Y_{III}$. Make reference to the concepts illustrated in Figure 2 of http://www.cs.cornell.edu/Courses/cs4786/2015sp/lectures/lecnotes2.pdf. You may *not* just say, "this happens because PCA is rotation invariant", since (a) we haven't really defined what that means yet, and (b) since we haven't given a definition, that statement as it stands doesn't seemingly make sense yet in light of the fact that the Ws differ.

Hint: We cannot imagine a correct answer that doesn't make use of the special meanings that the $\mathbf{w}_t$s produced by PCA have.

3. Let's generalize what we've just observed. A rotation matrix is a square matrix whose transpose is its inverse. (Intuition: a clockwise rotation can be undone via counterclockwise rotation by the same angular amount.) As usual, let our n d-dimensional data vectors be denoted by $\mathbf{x}_1, \ldots, \mathbf{x}_n$ (example: the rows of your X_I matrix above) and let R be a ~~$n \times n$~~ $d \times d$ rotation matrix. For simplicity, you may assume that the $\mathbf{x}_t$s have been centered. Let $\mathbf{x}'_t = R\mathbf{x}_t$ for each of the n $\mathbf{x}_t$s (example: the rows in your X_{II} matrix above), forming a second dataset.

   Now, for any K we pick, let us use PCA on each of the two data sets to obtain K-dimensional projections $\mathbf{y}_1, \ldots, \mathbf{y}_n$ and $\mathbf{y}'_1, \ldots, \mathbf{y}'_n$, respectively.

   (a) Write down a relationship between the two PCA projection matrices W and W' in terms of the rotation matrix R, and explain mathematically how you arrived at this answer.

   (b) Expand on your explanation from the previous subpart (Q1(2)) where you dealt with the two-dimensional matrices X_I and X_{II} and the fact you just proved (Q1(3a)) to explain why, for any $t \in \{1, \ldots, n\}$, the entries of $\mathbf{y}_t$ and $\mathbf{y}'_t$ are the same up to sign.

   Hint: to explain why the signs in corresponding entries might be flipped, consider the following question: if a vector $\mathbf{b}$ is an eigenvector of matrix C, must it follow that -$\mathbf{b}$ is?

4. Now lets move on to the magic trick!

   The story:
   You have an artist friend Picasso living in Spain who is into cubism (or, rather, squarism — we didn't know how to make cubistic smileys :)). When you first met, both of you drew a picture of the same smiley face, except that you drew a regular smiley face, whereas he drew a "cubistic" version of it (these are the files `smilie1.jpeg` and `cubistsmilie1.jpeg`). ~~which correspond to the first data points (rows) in `Xsmilie.csv`~~. Since then, you have sketched many a smiley face (`smilie2.jpeg` to `smilie28.jpeg`) and so has Picasso in his cubistic style (`cubistsmilie2.jpeg` to `cubistsmilie28.jpeg`). But of course, these are portraits of different faces, since he lives in Spain and you don't. You've stored the data for your non-cubist smileys in `X_smilie.csv`: each row i, i ranging from 1 to 28, is the (105×105)-dimensional data vector for your corresponding smiley face smilie$i$.jpeg.

Picasso decides to send you his latest portraits, but, data rates being what they are, and he being the suspicious type to boot[4], would like to economize on how much data he sends. Having heard about this dimension-reduction stuff, he performs PCA on the set of images that he has, obtains a 20-dimensional representation for each of his cubistic smileys, and then emails them to you. Specifically, he emails *only* his $\tilde{\mathbf{y}}$'s — we will, in what is unfortunately not an exact visual analogy, use tildes to indicate cubism — and *nothing else*. We have stored these projections for you in file `cubist_email.csv`, wherein row t is $\tilde{\mathbf{y}}_t$, whose 20 entries are stored as 20 floats separated by commas. The first ~~column~~ row of this file is the 20 dimensional projection of the image you and Picasso sketched together.

Upon receipt, you perform PCA on the images *you* sketched to obtain projection matrix W and the mean of your $\mathbf{x}$'s, which we will refer to $\boldsymbol{\mu}$. Now you simply use the 20-dimensional vectors Picasso sent over and reconstruct these images based on *your* W and $\boldsymbol{\mu}$ calculated from your non-cubistic set of sketches.

What do you think you would see? Try it out on the data we have provided!

How to try out the above:
Remember that any two low-dimensional projections produced via PCA of the *same* data will have the same absolute values for the entries of the $\mathbf{y}$'s but might have their signs flipped on coordinates. This is annoying, but can be fixed because, luckily, there exists that first image of *the same smiley face that you and Picasso each drew* in your own style. Take your projection $\mathbf{y}_1$ and Picasso's projection $\tilde{\mathbf{y}}_1$ corresponding to this face. Now, if on any coordinate, the sign of $\tilde{\mathbf{y}}_1$ and $\mathbf{y}_1$ are different, then for this coordinate flip the sign of *all* the $\tilde{\mathbf{y}}_t$'s. Finally, take the *new* $\tilde{\mathbf{y}}_t$'s and perform image reconstruction based on *your* PCA:

$$\hat{\mathbf{x}}_t = W\tilde{\mathbf{y}}_t + \boldsymbol{\mu}$$

for each of the cubistic image (and then transform them back into images).

Techie version of the story:
Your goal is to understand why you see what you see when you do this reconstruction. Of course, we need to explain how the cubistic smileys are generated. We start with a regular smiley face, which is a 105×105 pixel image. We block the image into 21×21 size patches. Since $105/21 = 5$, we have $5 \times 5 = 25$ total patches. We pick a fixed reordering of these 25 patches and simply reorder the patches in the regular smiley faces to create a "cubistic" version of the image.

Question: Given the way the cubistic smiley faces are generated, and based on the previous subparts this question, explain the quality of the images you reconstructed based on the email Picasso sent you.
Hint: ask yourself, can reordering the pixels be written as a rotation of $\mathbf{x}$'s? Even if so, how can this help, given that you don't know what that specific rotation was?

[4] http://www.bbc.com/news/entertainment-arts-31435906

Q2 (**When do the methods fail/succeed).** The goal of this question is to understand when PCA, CCA and random projection methods are good (relative to each other) and when they aren't.

1. Generate a 100-dimensional dataset consisting of 1000 points where the distribution of each coordinate individually (this is known as the feature's *marginal distribution*) has mean 0 and variance 1. The data points should be such that:

   - When we take view 1 to be the first 50 coordinates and view 2 to be the remaining 50 coordinates, and perform CCA with $K = 1$, then when we plot the points on a line (in each of the views), the first 500 points and last 500 points are well-separated. See the 11th slide of the slides for lecture 5 for a schematic of what we mean by "plotting the points on a line in each view".
   - In contrast, when we take view 1 to be the 50-dimensional vector obtained by taking all the odd coordinates and view 2 as the 50-dimensional vector consisting of all the even coordinates, and perform CCA with $K = 1$, then for either of the one-dimensional projections corresponding to either of the two views, there is no clear separation between the set of 1000 points (i.e., one can't separate/distinguish the first 500 from the next 500 set of points).
   - When we perform PCA on the data set with $K = 2$, there is no clear separation between the first 500 and the last 500 set of points.

   Submit your dataset as a csv file `CcaPca.csv` in the same format we've used in the files we supplied you (so, it should a plain-text file with 1000 lines, each with 100 comma-separated numbers in it). Also, in your assignment writeup, explain how you generated the two data sets (example kind of response: "I made all 100 features be drawn independently from the standard normal distribution") and the rationale behind this choice. Your rationale should explain how you used the properties of what CCA produces to guide your thinking.

2. Generate two data sets consisting of 100 1000-dimensional points. For each point $\mathbf{x}_t$ in the two data sets, ensure that their norm (distance to $\mathbf{0}$) is exactly 1. We shall perform PCA and random projections on both the data sets. To evaluate our projections we shall use the following metric on how well the projections preserve distances:

$$\mathrm{Err}(\mathbf{y}_1, \ldots, \mathbf{y}_n) = \frac{1}{n}\sum_{t=1}^{n} \left| \|\mathbf{y}_t\|_2 - 1 \right|$$

   (The absolute value bars were previously placed outside the summation.)

   You shall pick $K = 20$ and perform PCA and random projections on both the data sets. Your task in this problem is to create the data sets such that

   - On the first data set, Err of PCA is much smaller compared to that of random projection.
   - On the second data set, the Err of Random Projection is much smaller compared to that of PCA.

Submit your two datasets as csv files `PcaBeatsRp.csv` and `RpBeatsPCA.csv` in the same format we've used in the files we supplied you (so, they should be plain-text files with 100 lines, each with 1000 comma-separated numbers in it). Also, in your assignment writeup, explain how you generated the two data sets and the rationale behind this choice. Your rationale should explain how you used the properties of what RP and PCA produce to guide your thinking.