# 6.867: Homework 3

For the third homework, you will submit a single PDF containing your solutions to the following problems. There are 2 parts to the homework: a theoretical part and a hands-on part. We impose a strict 6-page limit on your writeup, to ensure that you focus on the important parts of your implementation and analysis. We make a few notes about the collaboration policy, submission procedure below, grading policy and other procedural information below.

## Collaboration Policy

- We *strongly* encourage you to work on the problems in groups. Group size can be arbitrary but we recommend keeping it to no larger than 5. However, all write-ups must be done individually, comprised of your own work, figures, and solutions.
- If partners submit identical PDFs, all will receive no credit.
- You should never use results from other students, the staff (from this year or from previous years), or other *online resources* in preparing your solutions to homework problems. In addition, students should never share their solutions (or staff solutions) with other students, including through public code repositories such as Github.

## Submission: CMT & Dates

CMT is an online conference management software that you will use to submit and peer-grade your submissions. Instructions on how to use CMT should have been posted to Piazza. A few notes on deadlines.

- Submit your paper via the CMT site by 11:59 PM on November 13th.
- After submission on CMT, you will be assigned 3 papers to review.
- Reviews must be entered on the CMT site by 11:59 PM on November 15th.
- All reviews will be made visible shortly thereafter, and students will have a two day period in which to enter rebuttals of their reviews into CMT if they wish.

## Procedure, grading, etc.

You will find a zip file with some useful code and data in the Stellar Materials page. You can do these assignments in any computational system that you are used to. We recommend Matlab or the Pylab/Numpy/Scipy/Matplotlib cluster of packages and we'll try to provide help for those two systems. If you use anything else, you're on your own...

You will be turning in a single, readable "paper" (a single PDF file) with your solutions. In the Stellar Materials page we have provided some sample solutions from a previous year. The content in these examples is not directly relevant, but note that there are coherent explanations and nice illustrations of results in figures and tables. You should write your paper as if it is going to be read by someone who has taken a class in machine learning but has not read this assignment handout.

We will be emulating the process of submitting papers for publication to a conference. We will be using an actual conference review system (CMT) to have these papers peer reviewed (by the other students). We will send out details about CMT soon. This means that your answers have to be readable and understandable to your peers and, where possible, interesting. Note that when explanations are called for, you will need to convince the reviewers that you understand what you're talking about. The course staff will serve as the Program Committee for the conference and make all final decisions.

### Grading rubric

**Your paper must be anonymous (no identifying information should appear in the PDF file). If it is not, it will automatically receive a 20% deduction, and will be graded by a grumpy staff member.**

**The paper must be no more than 6 pages long in a font no smaller than 10 point**. It should include whatever tables, graphs, plots, etc., are necessary to demonstrate your work and conclusions. *It should not include code.*

Each of the **four** parts of the assignment will be graded on a scale from 0 to 5 (where 0 is failing and 5 is an A) on two aspects:

- **Content:** Did the solution answer the questions posed? Were the answers correct? Were the experiments well-designed or examples well chosen?
- **Clarity:** Were the results written up clearly? Were the plots labeled appropriately and described well? Did the plots support the points in the paper? Did the discussion in the paper illuminate the plots?

As a reviewer, you will be asked to provide a score for each section, and at at least two paragraphs of feedback, per review, explaining things that were done well and things that could have been improved upon.

Your overall score for this assignment will be:

- **80%**: The average of all 8 scores on your assignment given by all three reviewers.
- **20%**: A score for the quality of your reviews. This will be full credit, by default. But we will skim reviews and examine some carefully and may reduce this grade for review commentary that is sloppy or wrong.

The course staff will spot-check submissions and reviews, paying careful attention to cases where there were rebuttals. The staff will act as the program committee and determine a final score. Our overall goals in this process are:

- To motivate you to work seriously on the problems and learn something about the machine learning material in the process
- To engage you in thinking critically and learning from other students' solutions to the problems

We will arrange to give full credit to anyone who submits a serious and careful solution to the problems and who gives evidence of having read carefully the solutions they were assigned and who writes thoughtful reviews of them.

The questions are the points that your paper should cover in order to receive full credit. Your presentation should roughly follow the order of these questions so that your reviewers can see what you're doing.

## Introduction to Theoretical Section

The first part of this assignment is 2 theoretical problems that will allow you to apply your more theoretical, proof-oriented problem solving skills. Note that this is a bit different from homeworks of past years, which were focused entirely on more hands-on implementation. For the theoretical problems, the writeup has no page-limit, and can be included with the write-up for the hands-on section. The theoretical problems are stated as follows.

## 1 Kernel PCA and Locally Linear Embedding

Numerous nonlinear dimensionality reduction methods have been proposed in the machine learning literature. One such method that has enjoyed some popularity is the so-called *Locally Linear Embedding (LLE)* method. The purpose of this exercise is to show that actually LLE can be viewed through the lens of the Kernel PCA method by defining a suitable kernel matrix.

Suppose we approximately represent each point of the dataset as a linear combination of its $n$ nearest neighbors. We use $\mathcal{N}(x)$ to denote the set of $n$ nearest neighbours of $x$. Define $W_n \in \mathbb{R}^{m \times m}$

as the optimal solution to the following constrained optimization problem:

$$\begin{aligned} \min_{W_n} \quad & \sum_i \|x_i - \sum_j (W_n)_{i,j} x_j\|^2, \\ \text{subject to} \quad & (W_n)_{ij} = 0 \text{ for } x_j \notin \mathcal{N}_n(x_i) \\ & \sum_j (W_n)_{ij} = 1 \text{ for } i, j \in 1, \ldots, m \end{aligned}$$

The weights $(W_n)_{ij}$ summarize the contribution of $x_j$ to the reconstruction of $x_i$.
The locally linear embedding (LLE) method seeks to represent $x_i$ in a transformed space as $Y_i$, such that with the $W_n$ we have just found, the loss function

$$\Phi(Y) = \sum_i ||Y_i - \sum_j W_{i,j} Y_j||^2$$

is minimized. It can be shown that the quadratic form $\Phi(Y) = \sum_{i,j} M_{i,j} < Y_i, Y_j >$ has $M_{i,j} = (I_n - W_n)^T (I_n - W_n)$. For the problem to be well posed, extra constrains are used: $\frac{1}{n} \sum_i Y_i Y_i^T = I$ and $\sum_i Y_i = 0$.

1. Let $I_m$ be the $m \times m$ identity matrix. Then, show that $k_n(x_i, x_j) := ((I_m - W_n)^T (I_m - W_n))_{ij}$ is a positive semidefinite kernel on the domain $\mathcal{X} = \{x_1, \ldots, x_m\}$.

2. Let $\lambda$ be the largest eigenvalue of $(I_n - W_n)^T (I_n - W_n)$. Prove that the LLE kernel,

$$k_n^{\text{LLE}}(x_i, x_j) := ((\lambda - 1) I_n + W_n^T + W_n - W_n^T W_n)_{ij},$$

   is positive semidefinite on $\{x_1, \ldots, x_m\}$.

3. Prove that kernel PCA using the LLE kernel $k_n^{LLE}$ provides the LLE embedding coefficients (which can be extracted from $k_n$). Note that if the eigenvectors are normalized in $\mathcal{H}$, then dimension $i$ will be scaled by $\lambda_i^{-1/2}$.

4. Discuss the variant of LLE obtained using the centered Gram matrix

$$(I_n - 1_m) \left( (\lambda - 1) I_n + W_n^T + W_n - W_n^T W_n \right) (I_n - 1_m).$$

   where $1_m$ is an all-one square matrix divided by m. Show that in this case, the LLE embedding is provided by $\alpha^2, \ldots, \alpha^{d+1}$, where $\alpha^1, \ldots, \alpha^{d+1}$ are the eigenvectors corresponding to the $d+1$ largest eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_{d+1}$ for the centered Gram matrix.
   **Hint 1**: what is $\lambda_1$?
   **Hint 2**: what is the matrix/vector form of $\Phi(Y)$?

5. Interpret the LLE kernel as a similarity measure based on the similarity of the coefficients required to represent two patterns in terms of $n$ neighboring patterns.

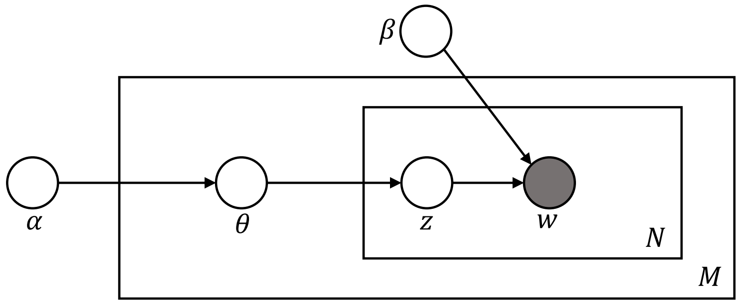

Figure 1: Graphical model representation of LDA.

## 2 Variational Inference for LDA

Latent Dirichlet allocation (LDA) is a probabilistic model for discovering topics in sets of documents. The generative process is defined as follows:

- For each document $\mathbf{w}$ in a corpus of $M$ documents:
    1. Draw topic probabilities $\theta \sim \text{Dirichlet}(\alpha)$
    2. For each of the $N$ words in $\mathbf{w}$:
        (a) Draw a topic $z_n \sim \text{Multinomial}(\theta)$.
        (b) Draw a word $w_n \sim p(w_n|z_n, \beta)$.

Here, $\beta$ is a matrix, with one column per topic. $p(w|z_n, \beta)$ is defined as a Multinomial distribution conditioned on the selected topic $z_n$.

The key inferential problem that we need to solve in order to use LDA is that of computing the posterior distribution of the hidden variables (topics) given a document:

$$p(\theta, \mathbf{z}|\mathbf{w}, \alpha, \beta) = \frac{p(\theta, \mathbf{z}, \mathbf{w}|\alpha, \beta)}{p(\mathbf{w}|\alpha, \beta)}.$$

Unfortunately, this distribution is intractable to compute in general. We instead computes its lower bound by using variational inference. Consider the following variational distribution:

$$q(\theta, \mathbf{z}|\gamma, \phi) = q(\theta|\gamma) \prod_{n=1}^{N} q(z_n|\phi_n),$$

where $q(\theta|\gamma)$ is a Dirichlet distribution parameterized by $\gamma$, and $q(z_n|\phi_n)$ is a Multinomial distribution parametrized by $\phi_n$. By definition, $\sum_{i=1}^{k} \phi_{ni} = 1$, where $k$ is the total number of topics.

1. Show that the log likelihood $\log p(\mathbf{w}|\alpha, \beta)$ is bounded below by the ELBO

$$\mathcal{L}(\gamma, \phi, \mathbf{w}; \alpha, \beta) := \mathbb{E}_q[\log p(\theta, \mathbf{z}, \mathbf{w}|\alpha, \beta] - \mathbb{E}_q[\log q(\theta, \mathbf{z})].$$

2. Expand $\mathcal{L}(\gamma, \phi, \mathbf{w}; \alpha, \beta)$ in terms of the observations $\mathbf{w}$, the model parameters $(\alpha, \beta)$ and the variational parameters $(\gamma, \phi)$. You may want to use the fact that

$$\mathbb{E}_q[\log(\theta_i)|\gamma] = \Psi(\gamma_i) - \Psi(\sum_{j=1}^{k} \gamma_j),$$

where $\Psi$ is the first derivative of the $\log \Gamma$ function[1], which is computable via Taylor approximations.

3. To update the variational parameter $\phi_{ni}$, the probability that the $n$th word is generated by the latent topic $i$, we maximize the ELBO over $\phi_n$. Observe that this is a constrained maximization since $\sum_{i=1}^{k} \phi_{ni} = 1$. Let $\beta_{iv}$ be $p(w_n^v = 1|z^i = 1)$ for the appropriate word $v$ (You need to verify this). Show that the optimum is achieved at

$$\phi_{ni} \propto \beta_{iv} \exp(\Psi(\gamma_i) - \Psi(\sum_{j=1}^{k} \gamma_j)).$$

4. To update the variational parameter $\gamma_i$, the $i$th component of the posterior Dirichlet parameter, we maximize the ELBO over $\gamma_i$. Show that the optimum is achieved at

$$\gamma_i = \alpha_i + \sum_{n=1}^{N} \phi_{ni}.$$

## Introduction to Hands-On Section

The 2nd part of this assignment deals with hands-on implementation questions. Section 3 asks you to perform an unsupervised learning task by using an LDA model for topic modeling on a given corpus. You will examine the topics this approach discovers as the algorithm's parameters are varied.

## 3 Topic Models

In this question, you will be performing an unsupervised learning task on real world data. More specifically, you will be using the latent Dirichlet allocation (LDA) model to perform topic modeling on a corpus of text.

At a high level, LDA views documents as being generated from a mixture of latent topic variables that have an associated distribution over word probabilities. When learning these topics LDA

[1] `http://mathworld.wolfram.com/LogGammaFunction.html`

further assumes a sparse Dirichlet prior for these distributions which models the fact that documents generally cover a small number of topics which in turn are primarily associated with a small number of words.

You are encouraged to read the review article by David Blei on topic models[2] to gain familiarity with this subject. Those who are interested may also read the seminal paper by Blei et al. introducing LDA[3].

In `hw3_resources.zip`, we provide you with the *20Newsgroups* data set[4], which consists of 19,997 articles for 20 categories taken from the Usenet newsgroups collection.

1. **Train an LDA model**: Learn an LDA model over all 19,997 articles (i.e., ignoring the known categories) with 100 topics. Use the implementation of LDA provided by Mallet[5] with the default parameters.

2. **Explore the topics**: Qualitatively describe the topics that your model has discovered. In addition, choose two of the 20 article categories and report the top three most represented topics in each of them, by averaging the topic distributions of the documents in the category. Do they make sense?

3. **Experiment with parameters**: Retrain the model with various numbers of topics. Report how decreasing the number of topics affected the topics that were discovered.

## 4 MovieLens

The MovieLens dataset is a collection of 20 million ratings collected from the movie recommendation website movielens.org. In `hw3_resources.zip`, you are provided with the smaller *MovieLens 1M Dataset*, which consists of 1 million ratings from 6000 users on 4000 movies.

Given this dataset of existing ratings, your task is to predict users' ratings of movies they have not seen. Formally, suppose we have $m$ movies and $n$ users. Let $Y \in \mathbb{R}^{m \times n}$ be our sparse matrix of ratings, where entry $Y_{a,j}$ is the rating from user $a$ on movie $j$. We would like to predict $Y_{a,j}$ for all users $a$ and all movies $j$.

1. **Singular Value Thresholding**: Define $Y' = [Y'_{ij}]$ where $Y'_{ij} = Y_{ij}$ if the entry $Y_{ij}$ exists in $Y$, and $Y'_{ij} = 0$ if the entry is missing. Recall that Singular Value Thresholding computes the Singular Value Decomposition of $Y'$

$$Y' = \sum_{i=1}^{r} \hat{\sigma}_i \hat{u}_i \hat{v}_i^T \tag{1}$$

[2] `http://www.cs.columbia.edu/~blei/papers/Blei2012.pdf`
[3] `http://www.cs.columbia.edu/~blei/papers/BleiNgJordan2003.pdf`
[4] `http://www.ai.mit.edu/people/jrennie/20Newsgroups/`
[5] `http://mallet.cs.umass.edu/`

and creates a new estimation $\hat{A}$ based off of either soft or hard thresholding on $\hat{\sigma}_i$. Implement both types of singular value thresholding. What choice of parameters $\tau_s$ and $\tau_h$ work best for soft and hard thresholding, respectively? Compare the performance of the two types with each other, as well as the other models.

2. **Alternative Least Squares**: Since $Y$ is a sparse matrix, we can approximate $Y$ as a product of low rank matrices, $Y = UV$, where $U \in \mathbb{R}^{m \times k}, V \in \mathbb{R}^{k \times n}$ for some $k \leq n$. We would like to minimize the squared loss,
$$F(U, V) = ||Y - UV||_F^2 \tag{2}$$
taken over entries for which $Y$ is defined. We may "gradient descent" over this loss by fixing $U$, optimizing $V$, and vice versa. Implement the alternating minimization algorithm to solve this problem.

3. **Collaborative Filtering**: For users $a, b$, let $M(a, b)$ be the set of movies with ratings from both users. Recall that the cosine similarity between users is defined as
$$\text{sim}(a, b) = \frac{\langle \hat{Y}_a, \hat{Y}_b \rangle}{||\hat{Y}_a||||\hat{Y}_b||} \tag{3}$$
where $\hat{Y}_a, \hat{Y}_b$ are feature vectors based on the common ratings,
$$(\hat{Y}_a)_j = Y_{a,j} - \frac{1}{|M(a, b)|} \sum_{l \in M(a,b)} Y_{a,l}. \tag{4}$$
Using this information, use the ratings of the top $k$ most similar users (with the metric above) to predict unknown ratings. Note the exact method of aggregation into a single rating remains to be chosen. Discuss both your choice of aggregation method and how varying $k$ changes your results.

4. **Neural Network**: Use a neural network to predict users' ratings of unseen movies. That is, for each user and movie, you should output a numeric score (1-5). You may use existing libraries (e.g. PyTorch, Tensorflow) to implement your neural network. You may also use auxiliary data provided in `user.dat` and `movies.dat`. Discuss which features worked best. Compare your results to the previous methods.