Machine Learning & Data Mining
Homework 4

Caltech CS/CNS/EE 155
February 24th, 2015

Overview

- This homework is due on Moodle at 2:00 pm on March 10, 2015.
- This homework explores the relationship between PCA and SVD, goes through the derivation of matrix factorization with missing values, and looks at a recent application of Markov embeddings.
- Students should be able to complete this homework in 8–10 hours.

1 SVD and PCA [Vineet Augustine, 30 points]

Question A [10 points]: Let X be an $N \times d$ matrix and $Y = X^T$. If the SVD of $Y = U\Sigma V^T$, then show that the columns of V are the PCA of X.

Question B: Prove that the SVD is the

(1) [10 points] Best rank-k approximation of a matrix in the Frobenius norm - i.e. $A = U\Sigma V^T$ satisfies

$$\|A - A_k\|_F = \min_{\text{rank[B] = k}} \|A - B\|_F$$

(2) [10 points] The complexity equivalent to trace norm

$$\|Y\|_* = \min_{Y=UV^T} \frac{1}{2}\left(\|U\|_F^2 - \|B\|_F^2\right)$$

Refer to Slides 16 and 17 of Lecture 13. Half the solution is there.

2 Matrix Factorization [Minfa Wang, 30 points]

In the class, we learnt that in the setting of collaborative filtering, the way we derive the coefficients of matrix $U_{K\times M}$ and $V_{K\times N}$ is to minimize the regularized squared error:

$$\underset{U,V}{\operatorname{argmin}} \frac{\lambda}{2}(\|U\|_{Fro}^2 + \|V\|_{Fro}^2) + \sum_{ij}(y_{ij} - u_i^T v_j)^2$$

In the above equation, u_i and v_j are the i^{th} and j^{th} column of U and V respectively.

Question A [16 points]: Specify ∂_{u_i} and ∂_{v_j} for stochastic gradient descent formula below where η is the learning rate:

$$u_i = u_i - \eta\partial_{u_i}$$
$$v_j = v_j - \eta\partial_{v_j}$$

Question B [14 points]: Another method to minimize the regularized squared error is alternating least squares (ALS). ALS solves the problem by fixing one of the U and V, and solving the optimal condition for the other. Then keep rotating this process until it converges. Please derive the closed form of optimal conditions of u_i and v_j.

3 Markov Embeddings [Bryan He, 60 points]

This problem explores the embedding model set up in Playlist Prediction via Metric Embedding. The model from this paper is designed to estimate the probability that a coherent playlist will transition from one song to another and also creates an automated method for playlist generation. This problem will focus on the dual-point model, which represents each songs s as two d-dimensional vectors $U(s)$, the "entry vector", and $V(s)$, the "exit vector", and makes it more likely for one song to transition to another if the ending of the first song is similar to the beginning of the second song. The probability of transitioning from song s_1 to song s_2 only depends on $\|U(s_2) - V(s_1)\|_2$, which is the Euclidean distance between the "exit vector" of the first song and the "entry vector" of the second song.

More formally, there is a collection of songs $\mathcal{S} = \{s_1, \ldots, s_{|S|}\}$, along with a training sample of playlists $D = (p_1, \ldots, p_n)$, where p_i is an ordered list of k_{p_i} elements denoted $p_i^{[1]}, p_i^{[2]}, \ldots, p_i^{[k_{p_i}]}$. In the transition probabilities, $p_i^{[0]}$ is a special start symbol. In the dual-point model, the transition probability from song $p^{[i-1]}$ to song $p^{[i]}$ is

$$P(p^{[i]}|p^{[i-1]}) = \frac{e^{-\|U(p^{[i]}) - V(p^{[i-1]})\|_2^2}}{Z(p^{[i-1]})}.$$

Next, the total probability of a playlist is

$$P(p) = \prod_{i=1}^{k_p} P(p^{[i]}|p^{[i-1]}) = \prod_{i=1}^{k_p} \frac{e^{-\|U(p^{[i]}) - V(p^{[i-1]})\|_2^2}}{Z(p^{[i-1]})},$$

where Z is a function that normalizes the probability. Finally, the total probability of the dataset is

$$P(D) = \prod_{p \in D} P(p) = \prod_{p \in D} \prod_{i=1}^{k_p} \frac{e^{-\|U(p^{[i]}) - V(p^{[i-1]})\|_2^2}}{Z(p^{[i-1]})}.$$

Question A: First, we will look at the motivation for using embedding rather than a more direct representation.

(1) [6 points] One way of modeling the transition probability $P(p^{[i]}|p^{[i-1]})$ is to have one value for each set of choices for $p^{[i]}$ and $p^{[i-1]}$. How many features will be used in this representation? Equivalently, how many parameters are in a full probability table?

(2) [6 points] If the representation of songs are d-dimensional, what is the number of features used in the dual-point model?

(3) [6 points] Based on the previous two parts, why is embedding used for playlist prediction, rather than modeling all of the transitions independently?

Question B: The feature representation of the songs can be found using gradient descent. For this problem, the gradients for the dual-point model will be found.

(1) [6 points] Rewrite the optimization problem (maximizing the data likelihood) as a optimization problem for the log-likelihood. This version of the problem is simpler to work with.

(2) [6 points] What is the normalization factor $Z(p^{[i-1]})$?

(3) [12 points] Compute the gradient of the log-likelihood for one of the U vectors for the dual-point model.

(4) [12 points] Compute the gradient of the log-likelihood for one of the V vectors for the dual-point model.

Question C [6 points]: The paper randomly initializes the feature vectors when running gradient descent for the dual-point model. In particular, the feature vectors cannot be initialized to just zeros. What is the problem in this case?