Recommender Systems

1. Suppose you run a bookstore, and have ratings (1 to 5 stars) of books. Your collaborative filtering algorithm has learned a parameter vector $\theta^{(j)}$ for user j, and a feature vector $x^{(i)}$ for each book. You would like to compute the "training error", meaning the average squared error of your system's predictions on all the ratings that you have gotten from your users.

Which of these are correct ways of doing so (check all that apply)? For this problem, let m be the total number of ratings you have gotten from your users. (Another way of saying this is that $m = \sum_{i=1}^{n_m} \sum_{j=1}^{n_u} r(i,j)$) [Hint: Two of the four options below are correct.]

> $\frac{1}{m} \sum_{(i,j):r(i,j)=1} ((\theta^{(j)})^T x^{(i)} - y^{(i,j)})^2$

> $\frac{1}{m} \sum_{i=1}^{n_m} \sum_{j:r(i,j)=1} (\sum_{k=1}^{n} (\theta^{(j)})_k x_k^{(i)} - y^{(i,j)})^2$

- $\frac{1}{m} \sum_{(i,j):r(i,j)=1} ((\theta^{(j)})^T x^{(i)} - r(i,j))^2$
- $\frac{1}{m} \sum_{j=1}^{n_u} \sum_{i:r(i,j)=1} (\sum_{k=1}^{n} (\theta^{(k)})_j x_i^{(k)} - y^{(i,j)})^2$

2. In which of the following situations will a collaborative filtering system be the most appropriate learning algorithm (compared to linear or logistic regression)?

- You manage an online bookstore and you have the book ratings from many users. You want to learn to predict the expected sales volume (number of books sold) as a function of the average rating of a book.

> You own a clothing store that sells many styles and brands of jeans. You have collected reviews of the different styles and brands from frequent shoppers, and you want to use these reviews to offer those shoppers discounts on the jeans you think they are most likely to purchase

- You're an artist and hand-paint portraits for your clients. Each client gets a different portrait (of themselves) and gives you 1-5 star rating feedback, and each client purchases at most 1 portrait. You'd like to predict what rating your next customer will give you.

> You run an online bookstore and collect the ratings of many users. You want to use this to identify what books are "similar" to each other (i.e., if one user likes a certain book, what are other books that she might also like?)

3. You run a movie empire, and want to build a movie recommendation system based on collaborative filtering. There were three popular review websites (which we'll call A, B and C) which users to go to rate movies, and you have just acquired all three companies that run these websites.

You'd like to merge the three companies' datasets together to build a single/unified system. On website A, users rank a movie as having 1 through 5 stars. On website B, users rank on a scale of 1 - 10, and decimal values (e.g., 7.5) are allowed. On website C, the ratings are from 1 to 100. You also have enough information to identify users/movies on one website with users/movies on a different website. Which of the following statements is true?

- You can combine all three training sets into one as long as your perform mean normalization and feature scaling **after** you merge the data.

> You can merge the three datasets into one, but you should first normalize each dataset separately by subtracting the mean and then dividing by (max - min) where the max and min (5-1) or (10-1) or (100-1) for the three websites respectively.

- It is not possible to combine these websites' data. You must build three separate recommendation systems.
- You can combine all three training sets into one without any modification and expect high performance from a recommendation system.

4. Which of the following are true of collaborative filtering systems? Check all that apply.

- When using gradient descent to train a collaborative filtering system, it is okay to initialize all the parameters ($x^{(i)}$ and $\theta^{(j)}$ to zero.

> If you have a dataset of users ratings' on some products, you can use these to predict one user's preferences on products he has not rated.

> Recall that the cost function for the content-based recommendation system is $J(\theta) = \frac{1}{2}\sum_{j=1}^{n_u}\sum_{i:r(i,j)=1}\left((\theta^{(j)})^T x^{(i)} - y^{(i,j)}\right)^2 + \frac{\lambda}{2}\sum_{j=1}^{n_u}\sum_{k=1}^{n}(\theta_k^{(j)})^2$. Suppose there is only one user and he has rated every movie in the training set. This implies that n_u = 1 and $r(i,j)$ = 1 for every i, j. In this case, the cost function $J(\theta)$ is equivalent to the one used for regularized linear regression.

- To use collaborative filtering, you need to manually design a feature vector for every item (e.g., movie) in your dataset, that describes that item's most important properties.

5. Suppose you have two matrices A and B, where A is 5x3 and B is 3x5. Their product is $C = AB$, a 5x5 matrix. Furthermore, you have a 5x5 matrix R where every entry is 0 or 1. You want to find the sum of all elements $C(i,j)$ for which the corresponding $R(i,j)$ is 1, and ignore all elements $C(i,j)$ where $R(i,j)$ = 0. One way to do so is the following code:

```
C = A * B;
total = 0;
for i = 1:5
  for j = 1:5
    if (R(i,j) == 1)
      total = total + C(i,j);
    end
  end
end
```

Which of the following pieces of Octave code will also correctly compute this total? Check all that apply. Assume all options are in code.

> total = sum(sum((A * B) .* R))

> C = (A * B) .* R; total = sum(C(:));

- total = sum(sum((A * B) * R));
- C = (A * B) * R; total = sum(C(:));