Assignment 2

The deadline for the second assignment is December 23, before the end of the day. Turn in the code, with instructions on how to run it (if necessary), and your report. The report should include detailed observations on the results. For late submissions the late policy on the page of the course will be applied. Details for the turn-in, and how to write reports are on the Assignments web page of the course. There will be an oral examination of the Assignment. The programming questions of the assignment can be done in teams of two.

Question 1

A power-law distribution is defined as $P(X = x) = (a - 1)x^{-a}$, where a is the exponent of the distribution. You are given a set of observations $Z = \{z_1, \dots, z_n\}$ that are generated from a power-law distribution. Use the Maximum Likelihood Estimation method we described in class to find the exponent of the power-law distribution that best fits the data observations.

Question 2 (Singular Value Decomposition)

For this question you will experiment with the application of Singular Value Decomposition. Download from the dataset "20 Newsgroups" of sklearn the data for topics 'talk.politics.mideast', and 'rec.sport.baseball' (there is a similar example in the tutorial). Use the TfidfVectorizer(stop_words='english',min_df=1) to extract the features, remove the documents with zero features and obtain the final data matrix M.

Compute the SVD decomposition of matrix M using the Numpy library and do the following tasks:

1. Compute the mean tf-idf score of each word (the mean value of the columns of M). Do a scatter plot of this vector with the first right singular vector (the one that corresponds to the highest singular value), and compute the pearson correlation coefficient between the two vectors. What do you observe?
   The first right singular vector is the vector on which we project the documents (the lines of M) to obtain the first let singular vector. What do you conclude for the first left singular vector?
2. Represent the documents as two-dimensional points, using the first two left singular vectors (the ones corresponding to the first two larger singular values), and plots the points.
   Do the same plot giving different color to the points that correspond to each topic. What do you observe? What do you conclude for the values of the second left singular vector?
3. Apply the k-means algorithm on matrix M and create the confusion matrix with the topics. Apply the k-means again on the two dimensional points from the previous question. What do you observe?

Hand in your code and a report with your observations.

Question 3 (Recommendation systems)

The goal of this question is to experiment with recommendation algorithms.

You will use a dataset with recipes from Kaggle. Download the train data. To obtain access to the data, you should create an account. Use your department email since you may need it in the next assignment.

The data is described in the Kaggle page. They are in json format, and you should use the Python json library to load them. They contain lists of ingredients for different recipes, and the food category (ethnicity).

You will work with the category "italian" which is the most popular. Using the data you will produce a binary matrix M, where the lines are recipes and the columns are the ingredients. You can write your won code, or use one of the existing libraries from sklearn (e.g., CountVectorizer). If you opt for the latter you may need to do a preprocessing of the data to remove the punctuation and make each ingredient into a single word.

Pick a random set R of 1000 recipes and remove randomly one of the ingredients from each recipe in R. The goal is to find these ingredients using algorithms for recommendations. For each recipe r in R, and for each ingredient i not already in the recipe r you will compute a score $s(r,i)$ and you will recommend the K ingredients with the highest scores. You will experiment with the following algorithms for computing the score $s(r,i)$:

1. **Most Popular (MP):** Use the popularity of the ingredient as the score (independent of the recipe r).
2. **User-based Collaborative Filtering** (**UCF**): For each recipe r compute the set $B_N(r)$ with the N most similar recipes. For the similarity computation you will use Jaccard Similarity. Let $J(r,r')$ denote the similarity between r and r'. You will compute the score $s(r,i)$ as follows:

$$s(r,i) = \frac{\sum_{r' \in B_N(r)} J(r,r') M[r',i]}{\sum_{r' \in B_N(r)} J(r,r')}$$

3. **Item-based Collaborative Filtering** (**ICF**): For each ingredient i not used by recipe r compute the set $B_N(i)$ with he N most similar ingredients. For the similarity computation you will use the Jaccard Similarity. Let $J(i,i')$ be the similarity between i and i'. You will compute the score $s(r,i)$ as follows:

$$s(r,i) = \frac{\sum_{i' \in B_N(i)} J(i,i') M[r,i']}{\sum_{i' \in B_N(i)} J(i,i')}$$

4. **Singular Value Decomposition** (**SVD**)**:** Apply the Singular Value Decomposition on matrix M, and keep the N first singular vectors to obtain a rank-N matrix M_N. Then you will use the value $s(r,i) = M_N(r,i)$ to select the K ingredients with the highest score to recommend for recipe r.

For the evaluation and the comparison of the algorithms you will use Precision@K, that is, the fraction of the recipes for which the algorithm identifies the removed ingredient within the K recommendations.

To determine the value of the parameter N, for the algorithms that use this parameter, you will set $K = 1$, and for different values of N in [1,100] you will compute the Precision@K. Do a plot with your measurements. Select the value N that gives the best precision. Then for the value of N that you selected for each algorithm, compute Precision@K for $K = 1,2,5,10$. Put all the results together in a table and a plot and compare the different algorithms.

Select at random 100 of the recipes in R and examine manually the results of the algorithms (για $K = 1$). In some cases some of the ingredients are essentially the same (e.g., different types of pasta, diced tomatoes instead of tomatoes, or minced garlic instead of garlic). How does the Precision@1 change if you do these corrections? Hand in an excel file (or a dataframe) with your evaluation consisting of triples of the form: ingredient, recommendation, evaluation (the lest should be filled manually) .

You should hand in the following:

- All the code that you wrote for the data processing, the algorithm implementation, and the evaluation.
- A report which describes what you did, compares the different algorithms, and comments on the results.

Bonus I: Add and extra category in your data and do the same process as above (for $K = 1$). Examine if the results improve or deteriorate.

Bonus II (for the fans of cooking): Select 100 new recipes at random (from which we wave not removed any ingredient). Apply the best of the algorithms you implemented for $K = 1$. Examine the recommended ingredient and evaluate if the recommendation is reasonable. Hand in the evaluation, with the commentaryyou're your evaluation and the Precision@1.

Notes:

- Use python functions to compute the distances, and do matrix computations, since they are significantly faster. The computation of the scores can be done using matrix computations. It will be counted positively if you use this approach.

Question 4 (Clustering)

In this Question you will practice with the application of clustering algorithms. We will use the same data as in Question 3. Select 3 categories with about the same size, and do the same processing as before to obtain a binary matrix. Apply the k-means, and agglomerative clustering algorithms for $k = 3$. (Do not take too large categories so that the Agglomerative algorithm can run.) Create the confusion matrix and compute precision and recall as we described in the tutorial. What do you observe?

It is possible that the documents do not cluster in a way that agrees with the categories. For the dataset you created in the previous step, using the k-means algorithm, and create the silhouette plot to decide the number of clusters. For the number of clusters you selected, examine the clusters output by the k-means algorithm using cluster centers and the categories and try to determine what type of recipes corresponds to each cluster.

Hand in your code and a report with a detailed commentary and analysis of your results.

Question 5 (Bonus)

Combine the work you did for Questions 3 and 4, and use clustering to improve the recommendation algorithm. Propose the algorithm, implement it, and compare it with the corresponding algorithm in Question 3