Homework Assignment 1

To learn the programming and analytics skills, master students are required to complete Option 1: hands-on exercises. PhD students have two options: (1) hands-on exercises and (2) paper reading.

Option 1: Hands-on exercises

1. Download the news articles dataset named "20news-18828.tar.gz" from an online textual dataset repository: http://qwone.com/%7Ejason/20Newsgroups/. The 20 Newsgroups data set is a collection of approximately 20,000 newsgroup articles, partitioned (nearly) evenly across 20 different newsgroups.

2. Convert them to a term frequency (TF) matrix (each row is an article, each column is a unique term, and each entry of this TF matrix is term frequency). In this step, it is important to preprocess the news articles using proper data preprocessing techniques such as
    a. remove semantically insignificant words, such as prepositions (e.g., on, in, next to, in front of, behind, between, under, through, around), pronouns (e.g., I, me, my, mine, you, your, yours, he, him, his, she, her, hers, it, its, we, us, our, ours, they, their, theirs, them), adverbs (e.g., easily, loudly, quickly, quietly, sadly, silently, slowly, always, frequently, often, once), articles (e.g., a, an, the), etc.
    b. remove rows/columns with many missing values
    c. estimate missing values
    d. normalization
    e. standardization, etc.

    Hints: You can read this post to learn how to process a large textual dataset. https://github.com/chenmiao/Big_Data_Analytics_Web_Text/wiki/Text-Preprocessing-with-R

3. Feature selection: Select top 100 best features from the data matrix. Popular feature selection methods include
    a. Removing features with low variance. This is because features with low variance provide limited information and poor classification power. The implementation of this method is available in Scikit-Learn: http://scikit-learn.org/stable/modules/feature_selection.html.
    b. Random Forest (Tree) based feature selection. This is a feature ranking method, which exploits Random Forest or decision trees to compute the importance of each feature for ranking features. To learn more technical details, please visit this post: https://www.r-bloggers.com/variable-importance-plot-and-variable-selection/. Scikit-Learn provides an implementation of this method: http://scikit-learn.org/stable/modules/feature_selection.html. Also, there is an implementation in R language: https://cran.r-project.org/web/packages/randomForest/randomForest.pdf. Then, you can select Top-K features.
    c. mRMR (maximize Relevance and minimize redundancy). There are a C++ implantation and a Matlab implementation of mRMR: http://home.penglab.com/proj/mRMR/. Also, there is an implantation in R: https://cran.r-project.org/web/packages/mRMRe/index.html.

Here is a post discussing how to use mRMR package in R:
http://stackoverflow.com/questions/36502796/using-mrmre-in-r .
You can also implement mRMR by yourself using a two-step framework. In the Step1, compute the correlations of feature pairs, select highly-correlated feature pairs, and remove one redundant feature from each highly-correlated feature pair by a thresholding method. In the Step2, compute the correlations between features and labels, rank the features in terms of correlations, and select Top-K features.

d. LaplacianScore for feature ranking. You can download the Matlab implantation of LaplacianScore from http://www.cad.zju.edu.cn/home/dengcai/Data/MCFS.html. Then, you can select Top-K features. This approach is published in a top machine learning conference called NIPS:
Xiaofei He, Deng Cai, and Partha Niyogi, "Laplacian Score for Feature Selection," NIPS 2005.

e. Directly select the top-100 most frequent terms as the top 100 best features

You can try one of them or combine multiple of them to conduct feature selection.

4. Compute the similarity between each pair of articles with Euclidean distance (you need to convert the Euclidean distance to similarity), Cosine similarity and Generalized Jaccard similarity. Particularly:
The definition of generalized Jaccard similarity is as follows:

Generalized Jaccard similarity and distance [edit]

If $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ and $\mathbf{y} = (y_1, y_2, \ldots, y_n)$ are two vectors with all real $x_i, y_i \geq 0$, then their Jaccard similarity coefficient is defined as

$$J(\mathbf{x}, \mathbf{y}) = \frac{\sum_i \min(x_i, y_i)}{\sum_i \max(x_i, y_i)},$$

and Jaccard distance

$$d_J(\mathbf{x}, \mathbf{y}) = 1 - J(\mathbf{x}, \mathbf{y}).$$

You can visit https://en.wikipedia.org/wiki/Jaccard_index for more details.
The definition of cosine similarity is as follows:

Given two vectors of attributes, *A* and *B*, the cosine similarity, *cos(θ)*, is represented using a dot product and magnitude as

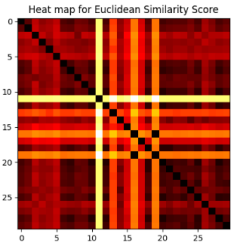
$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$

, where A_i and B_i are components of vector A and B respectively.

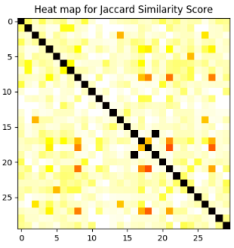
You can visit https://en.wikipedia.org/wiki/Cosine_similarity for more details.
The definition of Euclidean distance can be found via:
https://en.wikipedia.org/wiki/Euclidean_distance

5. Term frequency analysis: Let N denote the number of terms, provide term frequency histograms plot where X-axis (x=[1, 2, 3, 4, …, N]) represents the rankings of term frequencies in a descending order from let to right. For instance, 1 denotes the rank-1 term with the highest frequency; 2 denotes the rank-2 term with the 2-nd highest frequency. Y-axis represents corresponding term frequencies.

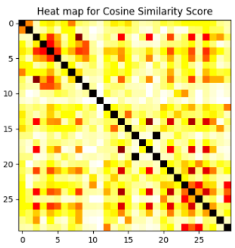
6. Heatmap based similarity analysis of document pairs: Let M denote the number of articles, compute three pairwise similarity matrices of M articles with respect to Euclidean, Cosine, and Jarcard. The size of this matrix is M*M. Then, plot the heatmaps of three similarity matrices. Finally, analyze whether the distances among articles are consistent. A sample heatmap plot is as follows:



(a) Euclidean Similarity Matrix



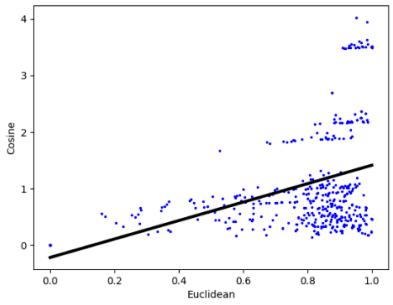
(b) Jaccard Similarity Matrix



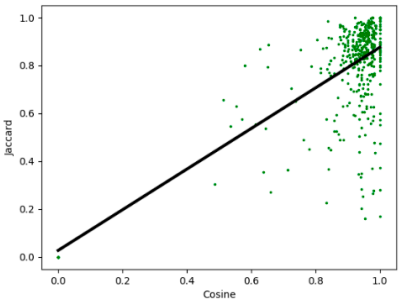
(c) Cosine Similarity Matrix

7. Analyze the correlations (degree of similarity) of Cosine-Euclidean, Euclidean-Jarcard, Jarcard - Cosine: First, compute the Pearson correlation coefficients of each similarity pair and list them in a table. Then, use the linear regression y=ax+b to fit the three similarity pairs
    - Cosine=a*Euclidean+b,
    - Euclidean=a*Jarcard+b, and
    - Jarcard=a*Cosine+b.

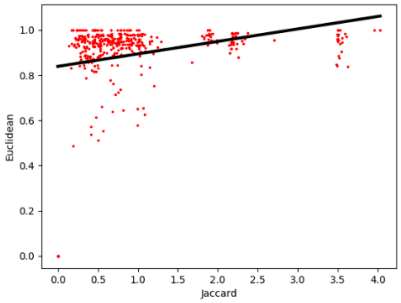
    Finally, plot the scatter plots and the fitted lines of the three similarity pairs. A sample plot is as follows:



(a) Cosine vs. Euclidean

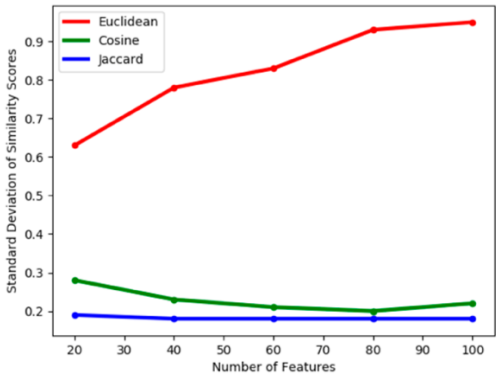


(b) Jaccard vs. Cosine



(c) Euclidean vs. Jaccard

8. Let N denote the number of features (i.e., terms). Analyze how the standard deviation of similarity scores changes/varies when the number of features (i.e., terms) increases from 2 to N with respect to Euclidean, Cosine, and Jarcard. A sample plot is as follows:



9. Rank all the articles pairs in a decreasing order of similarity with respect to each metric (Euclidean, Cosine, and Jarcard) and select top 3 article pair with respect to each metric. Compare the 9 article pairs and discuss which one is more accurate (i.e., close to your own judgment).

Deadline: please submit your results by 02/14 . You may compress your data files, code, and report into a zip file to submit.

Alternative Option for PhD students:

Select a feature-selection paper that has more than 200 citations from Google Scholar, and present in the class on 02/14. You are encouraged to include motivation, problem formulation, methodology overview, technical details, and experiment interpretations in your presentation. The presentation should take no more than 10 minutes.