NYU | COURANT

MATH-GA.2047-001 Data Science in Quantitative Finance

Homework 7

Profs. Ivailo Dimov and Petter Kolm
Due: October 27, 2020

Instruction

This homework is to be done individually. No collaboration and/or code sharing permitted.

Objective

In this assignment, you will:

- Build an LSA-based recommender and understand its pros and cons.

Methodology and Deliverables

1. LSA-based Recommender.
   In this assignment, you can use and modify the LSA notebook that we covered in class.

   Start by downloading the Reuters 10K article corpus `raw_text_dataset.pickle` from https://github.com/chrisjmccormick/LSA_Classification.

   (a) Create a `doc2vec(doc, tfidf_vectorizer)` function corresponding to a TFIDF vectorizerer where:
   INPUTS: `doc`, `tfidf_vectorizer`

   - `doc`: any string
   - `tfidf_vectorizer`: a TfidfVectorizer instance

   OUTPUTS: `vec`, `doc_features`, `doc_counts`

   - `vec`: a vector with L_2 norm of 1
   - `doc_features`: the features after tokenization and pre-processing
   - `doc_counts`: the counts of each feature in this document

   Train your `tfidf_vectorizer` on the Reuters 10K article corpus.

   (b) For each of the following doc strings, calculate their corresponding vectors

- `doc1`: "Jabberwocky"
- `doc2`: "buy MSFT sell AAPL hold Brent"
- `doc3`: "bullish stocks"
- `doc4`: "Some random forests produce deterministic losses"

(c) Implement a function `recommend(vec, X_model, X_corpus)` which projects any document vector onto a given `X_model`

- where `X_model = {X_train_tfidf, X_train_lsa}`

and returns `doc_vec, idx_top10, sim_top10, X_top10` where

- `doc_vec`: the (sparse) vector of similarity scores of vec and members of `X_model`. This vector should be size $D \times 1$
- `idx_top10`: the indices of the top-10 similarity scores
- `sim_top10`: the top-10 similarity scores
- `X_top10`: the top-10 corpus articles most similar to the input model

What does your `recommend()` function output for the doc vectors in (b)? Do you see an improvement of the LSA similarity recommendation relative to the TF-IDF similarity recommendation?

(d) *Extra credit:* Repeat the same exercise but instead of the Reuters 10K dataset, use the following corpus of 200K English plaintext jokes: https://github.com/taivop/joke-dataset. Does your recommender system actually find similar jokes? Give examples of good and bad recommendations. Provide a list of suggestions of how one could improve upon this recommender.