Project 1

Let's build a recommender system.

Overall deliverable guidelines

The main deliverable is a GitHub repository with the following:

- A README file outlining the repository contents
- A requirements file with all software/package requirements to run your code
- Your code:
    - All supporting methods you write to solve the problems
    - Any initial analysis you conduct, if you do so
    - The final notebook or markdown with results. This should be clearly marked as the final product.

For each person in your group, please include your full name, email, and uni.

Projects should be completed in Python (preferred) or R.

Data Scientists are more than statisticians and more than engineers. We solve real-life business problems by leveraging data and are usually brought in to brainstorm and frame a problem from day one. You will probably be expected to interface directly with other departments in the business, where your trust will be gained by sharing your insights into the core business problem at hand, and by translating these problems into a technical solution. This also means that you will be required to *communicate* your solution to key stake holders that are committed to solving the problem, but may not have your technical background.

Imagine a future employer looking at this repository. They would evaluate your project based on the solution's accuracy and your technical prowess, but they would also look for coherence and creativity. An employer would look for thoughtfulness and thoroughness; for example, how does the solution scale, were the hyper parameters tuned, will this solution discover novel recommendations, etc. Are the major contributions of your work clear? Did you call out important caveats?

These are the same standards on which you will be graded for this project.

Instructions for groups

You are organized into groups. Each team member will bring their own strengths and have their own opportunity areas for development. It's ok for teams to divide and conquer for divisible tasks, but *every team member should have a complete understanding of the entire solution, end to end*. I also recommend that before any piece solution is implemented by an individual, the entire team should brainstorm possible approaches, and should agree on the framework, what to try, and what the deliverable will look like.

Data set

- Please use the “20Ml” data set for education and development (20M ratings; 190 MB):
    - https://grouplens.org/datasets/movielens/

Instructions

Due November 6th

1. Objective

You will build two very simple collaborative filtering models. You may use published packages or methods (e.g. the ALS method in Spark ML) - the goal of this exercise is to gain a practical intuition for how these types of common models work, and to develop methods to test and explore them.

Treat this as a case study and think about it from a business perspective. Imagine that you work for a digital media company that needs to recommend movies. What is your objective? What are you trying to optimizing and what are you willing to sacrifice? This framework and these decisions should be clear in your report.

2. Data and methods

Start developing with a small data set (e.g. < 10,000 users / < 1,000 items). Be very thoughtful about how you sample your data. For example, do you care about choosing popular items or unpopular items? Do you care about how many items a user has rated? If you just sample the raw data in your downloaded file, will you be missing relevant data for samples users or items?

Please use Spark where applicable (https://spark.apache.org/downloads.html). Use can use Spark ML methods from a Python or R environment.

3. Report

- As data scientists of a digital media company, state your objectives in building a recommendation system. For example, what metrics do you care about, who is this system built to serve (users or your boss?), and what business rules may you care to introduce?
- Build two brute-force collaborative filtering algorithms to recommend items to users:
    1. Neighborhood-based (either item *or* user)
    2. Model-based (you can pick one, but vanilla Matrix Factorization is a good default)
- Develop evaluation methods:
    1. Cross-validation setup
    2. Accuracy on training and test data (choose a primary accuracy metric and a secondary accuracy metric)

    3. Coverage on training and test data
- Use your small development data set. Systematically try a range of hyper parameters for your models (e.g. neighborhood size or number of latent dimensions). Record *and explain* in your markdown how your evaluation metrics change as a function of these parameters. *Include plots!*
- After seeing these results, what other design choices might you consider in order to have a more accurate or more useful model?
- How do the evaluation metrics change as a function of your model size? Systematically change your data set size by sampling your data from a small size to a large size
    1. Does overall accuracy change?
    2. What about the distribution of accuracy over users or items?
    3. How does run-time scale with data size?
- How does your recommendation system meet your hypothetical objectives? Would you feel comfortable putting these solutions into production at a real company? What would be the potential watch outs?

4. **Evaluation**

[10%] Case study framework
- How well was this envisioned as an actual business problem? Is it clear what is being solved, and what acceptance criteria would be?

[30%] Technical correctness
- Is the code complete and did it run?
- Were appropriate methods applied in the right place? Were hyper-parameters used in a reasonable manner?

[10%] Evaluation methods
- Correct implementation

[25%] Model exploration
- How well was each model explored?
    - Hyper-parameters
    - Data sample size
    - (Optional) Anything else?
- Are plots clearly legible, and do they make sense?

[25%] Write up
- If I were your manager at work, or a key stake holder, and I needed to understand what you did and why you did it in order to make a business decision (e.g. “go / no-go”), is this write up enough for me to go on? Would I have any lingering questions that were not already pointed out as “potential watch outs”?
- Clarity matters
- Connection to the business problem matters
- Demonstrating an understanding of why the algorithms are performing or not, or *when* then do or do not, matters