twiml

AI Enterprise Workflow Study Group

Course 5, Week 2

5/9/2020

Agenda

- Check in
- Discussion
- Next steps

Course & Study Group Schedule

AI Enterprise Workflow Study Group		
Session	**Topic**	**Date**
Overview Webinar	Webinar with instructor, Ray Lopez	15-Feb
Course 1 Week 1	Course intro	22-Feb
Course 1 Week 2	Data ingestion, cleaning, parsing, assembly	29-Feb
Course 2 Week 1	Exploratory data analysis & visualization	7-Mar
Course 2 Week 2	Estimation and NHT	14-Mar
Course 3 Week 1	Data transformation and feature engineering	21-Mar
Course 3 Week 2	Pattern recognition and data mining best practices	28-Mar
Course 4 Week 1	Model evaluation and performance metrics	18-Apr
Course 4 Week 2	Building machine learning and deep learning models	25-Apr
Course 5 Week 1	Deploying models	2-May
Course 5 Week 2	Deploying models using Spark	9-May
Course 6 Week 1	Feedback loops and monitoring	16-May
Course 6 Week 2	Hands on with OpenScale and Kubernetes	23-May
Course 6 Week 3	Captsone project week 1	30-May
Course 6 Week 4	Captsone project week 2	6-Jun

Course 5 Week 2 learning objectives

1. Build a data ingestion pipeline using Apache Spark
2. Tune hyperparameters in machine learning models on Apache Spark
3. Explain how collaborative filtering and content-based filtering work
4. Connect Apache Spark Streaming to a recommendation engine
5. Deploy a ML algorithm with both training and prediction endpoints

Course 5 Week 1 learning objectives

1. Build a data ingestion pipeline using Apache Spark
2. Tune hyperparameters in machine learning models on Apache Spark ???
3. Explain how collaborative filtering and content-based filtering work
4. ~~Connect Apache Spark Streaming to a recommendation engine~~
5. ~~Deploy a ML algorithm with both training and prediction endpoints~~

Spark Pipelines

- **DataFrame**. This ML API uses DataFrame from Spark SQL as an ML dataset which can hold a variety of data types. For example, a DataFrame could have a mixture of continuous and categorical data types.
- **Transformer**. A Transformer is an algorithm which can transform one DataFrame into another DataFrame. For example, a ML model is a Transformer which transforms a DataFrame with features into a DataFrame with predictions.
- **Estimator**. An Estimator is an algorithm which can be fit on a DataFrame to produce a Transformer For example, a learning algorithm is an Estimator which trains on a DataFrame and produces a model.
- **Pipeline**. A Pipeline chains multiple Transformers and Estimators together to specify an ML workflow.
- **Parameter**. All Transformers and Estimators now share a common API for specifying parameters.

Exploration

Random Forest Demo

Additional Spark Notes

RDD vs DataFrames - https://spark.apache.org/docs/latest/sql-programming-guide.html

MLlib vs “Spark ML”:

- `import org.apache.spark.mllib.`
- `import org.apache.spark.ml.Pipeline`

Pipelines - https://spark.apache.org/docs/latest/ml-pipeline.html

Tuning - https://spark.apache.org/docs/latest/ml-tuning.html

PySpark Docs - https://spark.apache.org/docs/latest/api/python/pyspark.html

Learning Apache Spark - https://mingchen0919.github.io/learning-apache-spark/index.html

Recommender Systems

Consider when asking Qs like:

- What would a user like?
- What would a user buy?
- What would a user click?

Also use cases like autocompletion.

Supervised learning doesn't perform well for some highly multiclass problems.

Recommender systems can be a solution.

Types of Recommender Systems

Explicit vs Implicit

- Explicit => data comes from ratings
- Implicit => data comes from behaviors like likes, shares, visits or time watched
- Hybrid => combines the two; most common systems can handle both

Collaborative vs Content filtering

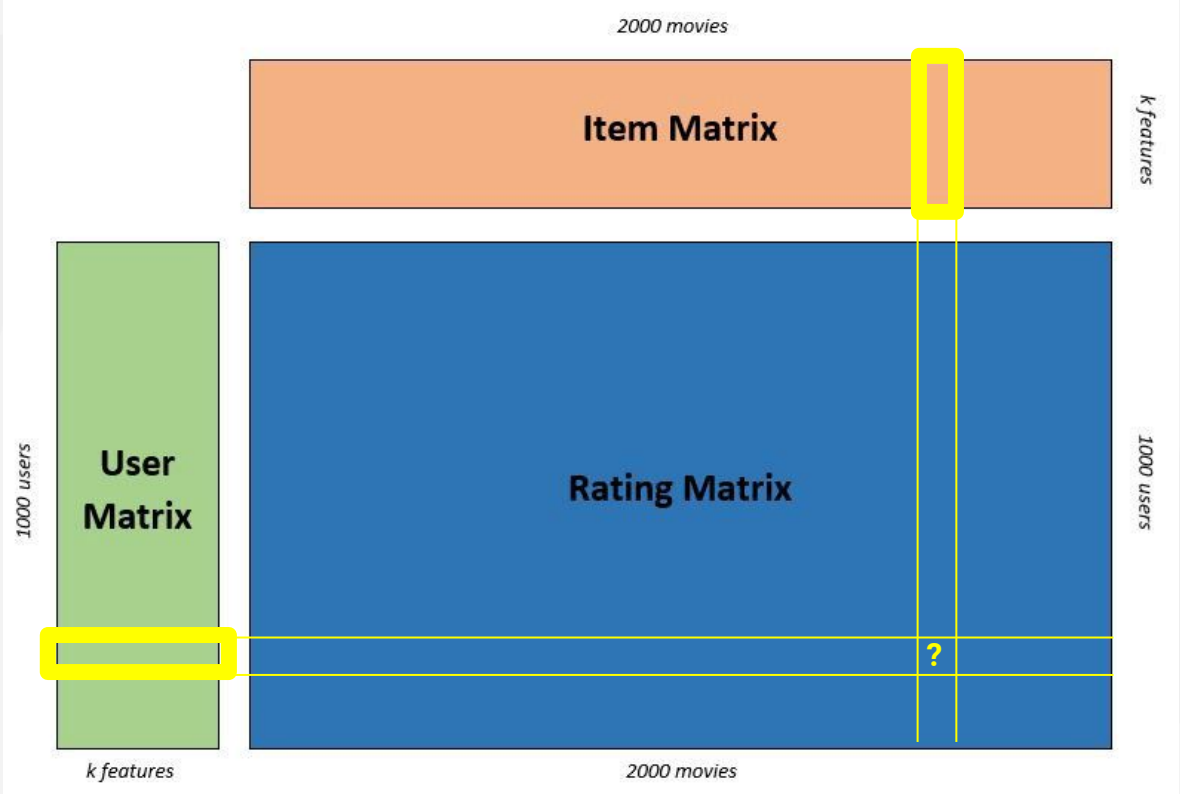
- Collab. filtering => identifies subset of users who have preferences similar to target user. ratings matrix is created. items preferred by similar users combined and filtered to create ranked recommendations
- Content-based filtering - predictions based on the properties and characteristics of an item. user behavior not considered
- Hybrid => combines the two

Matrix Factorization

Matrix Factorization

Matrix Factorization

Matrix Factorization



Matrix Factorization

Utility matrix is very, very large and very sparse

Use SGD or ALS (Alternating Least Squares) to learn latent factors.

RMSE used as loss function.

ALS can be run parallelized and is appropriate for large-scale problems.

Was created by HP Labs researchers as part of Netflix Prize competition:

Paper: Large-scale Parallel Collaborative Filtering for the Netflix Prize

Challenges

Cold Start Problem

- If a user hasn't rated anything, what do we recommend to them? (Some services will ask you to choose content categories on signup.) Can use most popular items; even better if customized using metadata.
- If a new item hasn't been rated, can randomly suggest it during a trial period (airbnb ex) to collect data. Add to special section e.g. apple podcast new and notable. Use metadata to find similar items and infer.

Concurrency another challenge due to complex inference. Can use Spark and distribute workload, or e.g. like celery or bento in python. Use simulation to investigate.

Exploration

Case Study

Additional Discussion

What did you learn?

What stumbling blocks did you run into?

How do these lessons relate to your experience?

What did you learn/find interesting in this week's lesson?

What are you doing as homework?

What interesting resources have you found?

Other?