AN EXAMPLE OF PREDICTIVE ANALYTICS: BUILDING A RECOMMENDATION ENGINE USING PYTHON

A little about me

- Data Scientist at Texas Advanced Computing Center (TACC)
- My Contact: atrivedi@tacc.utexas.edu ; @anurive
- Independent research center at UT Austin
- One of the largest supercomputer center, HIPAA compliant
- ~250 faculty, researchers, students and staff
- We work on providing support to large scale computing problems

MOTIVATION

Talk Outline

- Predictive Analytics Vs Recommender Systems
- Introduction to recommender systems
- Types of recommender systems
- Recommender Systems for Pubmed document
- Workflow of TACC's Recommender Systems
- Live Demo

Predictive Analytics
Vs
Recommender Systems

Data Analytics

- Mainly three types of Data Analytics:

Descriptive	Predictive	Prescriptive
1. The simplest class of analytics 2. Allows you to condense big data into smaller and more useful chunks of information 3. Use it when you need to understand things at an aggregate level 4. E,g: Vast majority of the statistics that we use (sums, averages, percent changes etc).	1. It utilizes a variety of statistical modeling, data mining, and machine learning techniques to study recent and historical data 2. It allows analysts to make predictions about the future. 3. Use Predictive analytics any time you need to know something about the future or fill in the information that you do not have. 4. E.g: Sentiment Analysis	1. Prescriptive analytics is a type of predictive analytics 2. Prescribe or recommend an action to end users 3. Helps informed decision based on data 4. E.g: Recommendation System

Introduction: Recommender Systems

What are Recommendation Systems?

- Recommender System helps match users with item
- Implicit or explicit user feedback or item suggestion
- Different Recommender System designs
  - Based on the availability of data or content/context of the data
- Our Recommendation system:
  - We try to build a model which recommends Pubmed documents to users

Why **Recommendations** ?

The world is an over-crowded place

Types of Recommender System

Types of Recommender System

Types	Pros	Cons
Knowledge-based (i.e, search)	Deterministic recommendations, assured quality, no cold- start	Knowledge engineering effort to bootstrap, basically static
Content-based	No community required, comparison between items possible	Content descriptions necessary, cold start for new users
Collaborative	No knowledge-engineering effort, serendipity of results	Requires some form of rating feedback, cold start for new users and new items

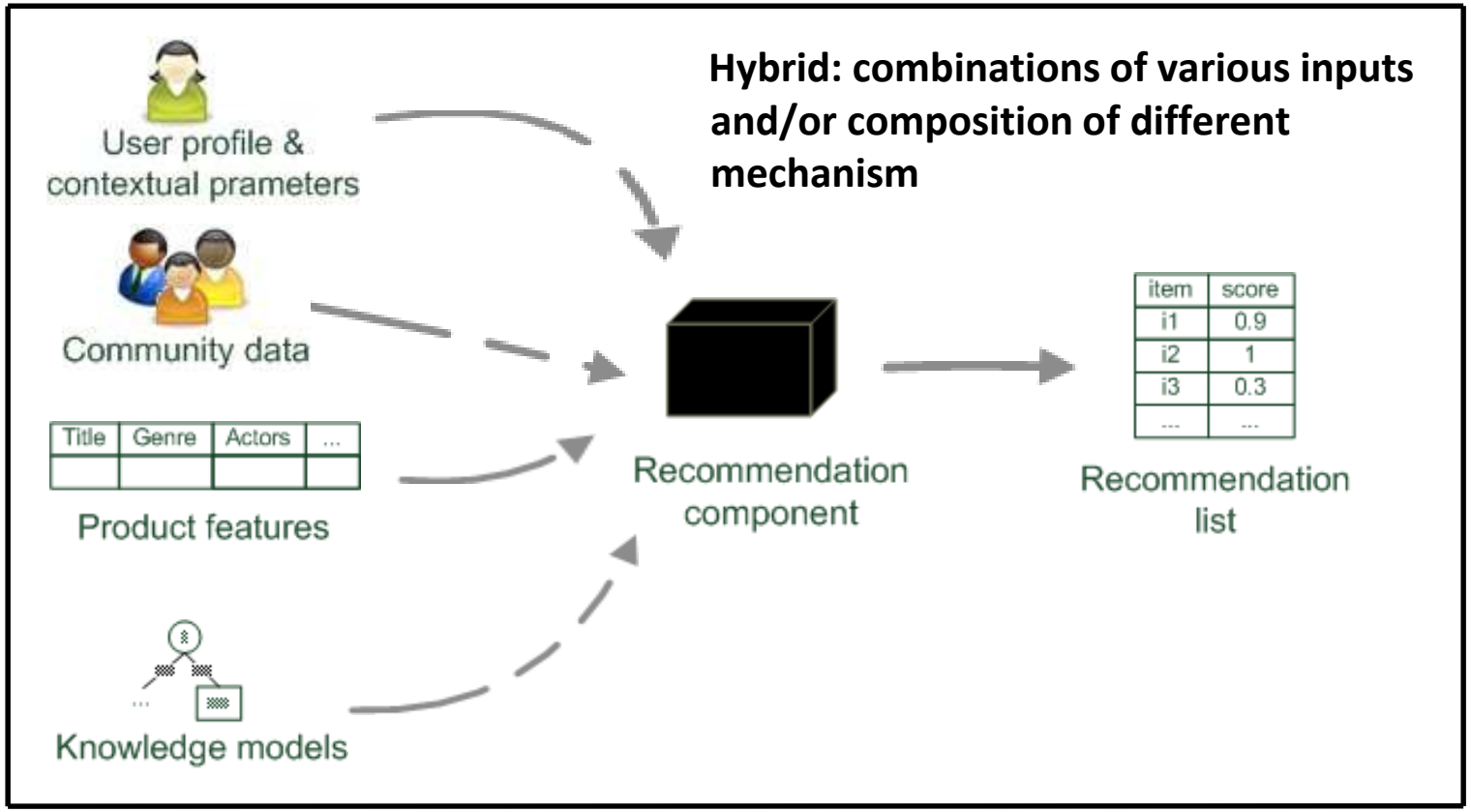
Commonly used Example: User-based CF

- The recommendation system recommends books to customer C

	Book 1	Book 2	Book 3	Book 4	Book 5	Book 6
Customer A	X			X		
Customer B		X	X		X	
Customer C		X	X			
Customer D		X				X
Customer E	X				X	

- Customer B is <u>very close</u> to C (s/he has bought all the books C has bought). Book 5 is highly recommended for customer C
- Customer D is <u>somewhat close</u>. Book 6 is recommended to a lower extent
- Customers A and E are not similar at all

Paradigms of a Hybrid Recommender System



Why need Hybrid Model for recommending Pubmed Documents

- The problem that we are trying to solve
  - Users search documents in Pubmed
  - Based on their search, create a user profile with relevant documents
  - Recommend documents in context with the search

- The steps needed to solve:
  - Search
  - Information Retrieval for users based on Content and Context
  - Recommend documents in context with the search

Our Model: Hybrid Recommendation Engine for Pubmed

Step 1: Content-based filtering on Pubmed documents

- We search documents using query term
- We tokenize each document into a combination of unique terms
- We compare the pubmed documents to a sparse matrix of documents and terms

Step 2: We weight the query term in the sparse matrix and rank documents

We apply Vector Space Model (VSM) to combine Step 1 and Step 2

Our Model: Hybrid Recommendation Engine

Step 3:

- We filter the weighted/ranked documents using collaborative filtering/recommendation
- We use a python library python-recsys (by Oscar Celma - Director of Reseach at Pandora) for recommending Pubmed documents.
- We used Pandas for preprocessing data
- We use Scikit-learn to evaluate our model
- We do two types of recommendation -
  - ➢ Item-based Recommendation
  - ➢ User-based Recommendation
- Several data pre-processing steps are involved in Pubmed document recommendation.

Vector Space Model (VSM)

- Given:
  - A set of Pubmed documents
  - N features (terms) describing the documents in the set
- VSM builds an N-dimensional Vector Space.
- Each itemdocument is represented as a point in the Vector Space
- Information Retrieval based on search
  - Query: A point in the Vector Space
  - We apply TFIDF to the tokenized documents to weight the documents and convert the documents to vectors
  - We compute cosine similarity between the tokenized documents and the query term
  - We select top 3 documents matching our query

Available Recommendation Libraries

Motivation for using Python-Recsys library

Python-Recsys library

- A **python** library for building recommendation engines
- Works with **Scikit-learn** for collaborative, content and hybrid filtering
- Underlying model: K-SVD (good for sparse matrix)
- Open Source: https://github.com/ocelma/python-recsys

Code Execution: Demo

We try to show that our Hybrid model (using VSM and python-recsys combined) works better that using python-recsys library alone

- First, we explain the collaborative filtering technique, using the python-recsys library on a simple movie data set
- Next, we explain our hybrid model for pubmed recommendation
- We run comparative evaluations on the model (RMSE & MAE). Our model performs better – as it utilizes the content/context of items

Data Used for Demo

- Movie dataset format:

UserID	MovieID	Rating
1	1	5.0
2	1	3.5
3	2	4.0
4	3	5.0
5	3	2.5
6	4	4.5
7	4	4.0

- Pubmed dataset format:

ID	User Name	User ID	Query Term	Doc ID	Doc Rank	Doc Name
44	paul	11	genetics	7713442	0.341534906	…

Each_Doc_Token	Combined_Docs_Tokens
….	….

Conclusion

- Pros:
  - Python-Recsys is very accurate recommendation library
  - The Hybrid model increase the prediction/recommendation accuracy
  - Great community support

- Cons:
  - Parallelization is not that great
  - For Big Data, need to use different functions instead of SVD.

Acknowledgement

1. Oscar Celma: Python-Recsys - https://github.com/ocelma/python-recsys

2. Alan Said: Comparative Recommender System Evaluation - Benchmarking Recommendation Frameworks

3. Pasquale Lops: Semantics-aware Content-based Recommender

THANK YOU !

Thanks for your attention.

Questions?