metarecommendr

Predicting your next Game, Movie, TV Show

By Yvonne, Stefan, Daniel

Presentation Outline

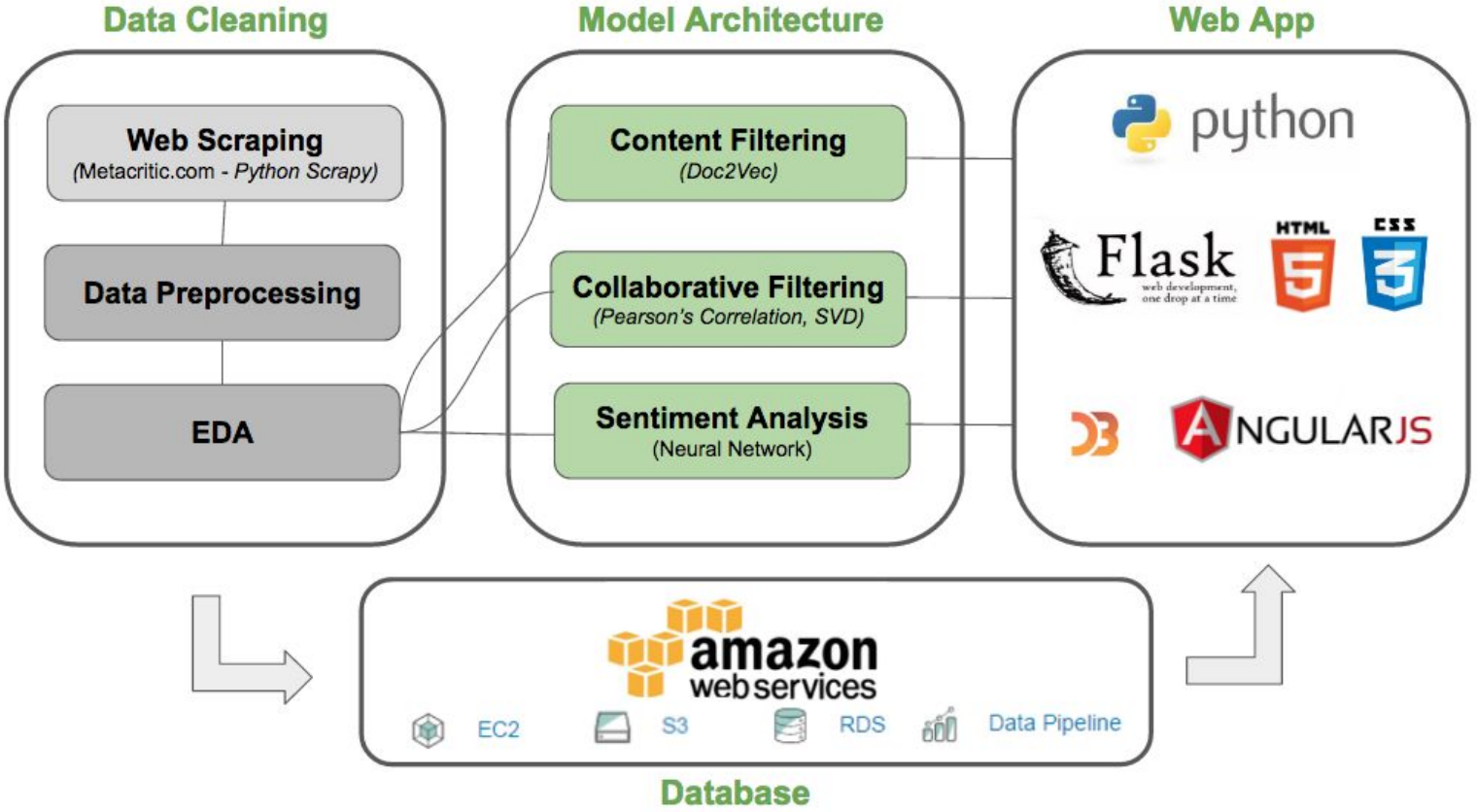
1. Introduction
2. Web Scraping
3. EDA
4. Model Architecture
    a. Content Filtering
    b. Collaborative Filtering
    c. Sentiment Analysis
5. App Demo

metarecommendr

1. Introduction: *Metacritic*

- Metacritic offers a centralized platform aggregating reviews for different media products *(Video games, Movies, Music, TV Shows)*
  - Aggregated Critic reviews from 1-100, all in English
  - User reviews from 1-10, not restricted to English
- Problem:
  - Product suggestions is usually within the same category
    - Need to sift through lots of text to find non-obvious recommendations
  - User Interface can be too crowded with information
  - Review Sentiment sometimes does not correspond to score
- Solution:
  - Metarecommendr
    - Optimized recommendations using collaborative and content filtering
    - Clean Interface to find products
    - Review Grader to check sentiment of user reviews

1. metarecommendr - *Project Workflow*



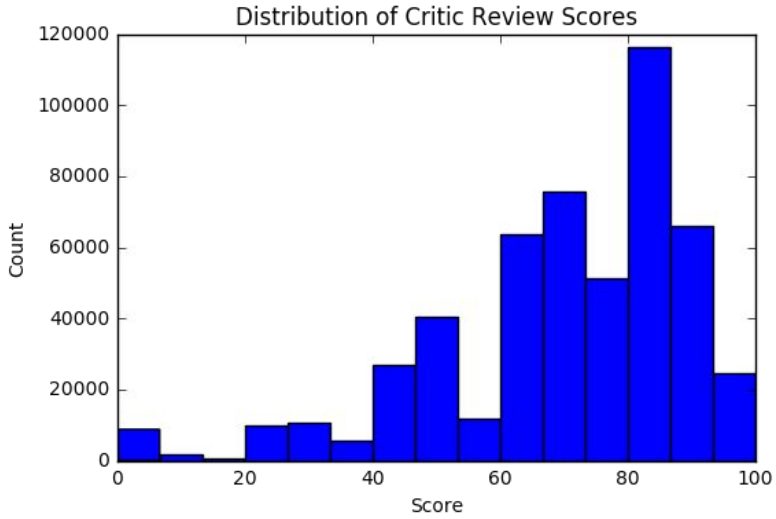
2. Web scraping: *Python Scrapy*

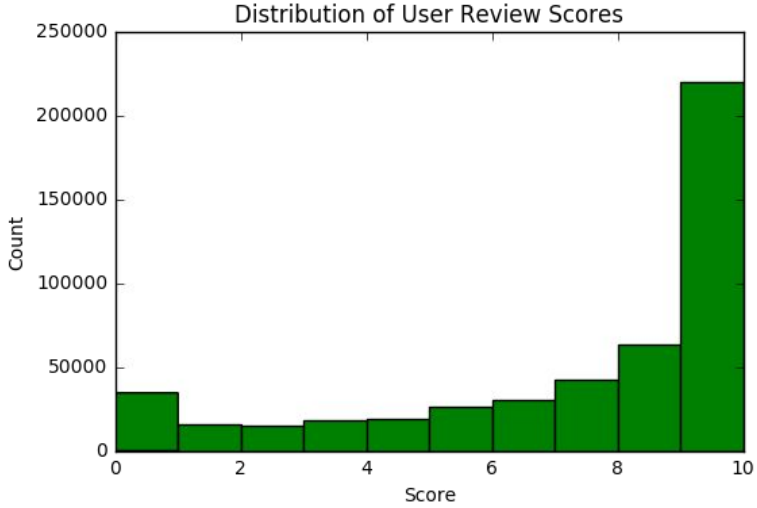
- Website: www.metacritic.com
- ~1 Million reviews scraped in 10 days via Python Scrapy
  - Data scope:
    - Movies: Last 10 years
    - TV Shows: All TV shows
    - Games: All games
- Since metacritic is more popular among gamers, there are more game reviews

Games, TV, Movies	**Reviews Field**
Name	Publication
Link	Author
Image Link	Date
Developer	ReviewType *(Critic or User)*
Rating	Review Content
Release Date	Score
Summary	
System	

3. EDA: *Overall Statistics*

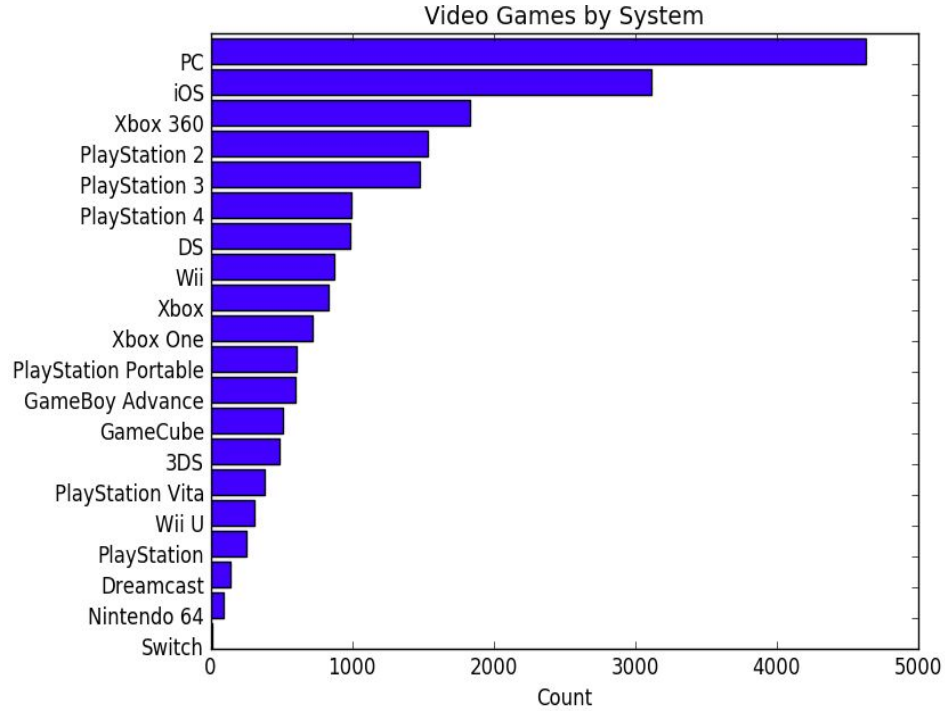
- # Items
  - Games: 20416
  - Movies: 5470
  - TV Shows: 1978
- Critic Reviews:
  - 514,304 Critic Reviews
    - Mean: 68.96
    - Median: 74.0
    - Standard Deviation: 19.98
- User Reviews:
  - 484,278 User Reviews
    - Mean: 7.03
    - Median: 8.0
    - Standard Deviation: 3.16

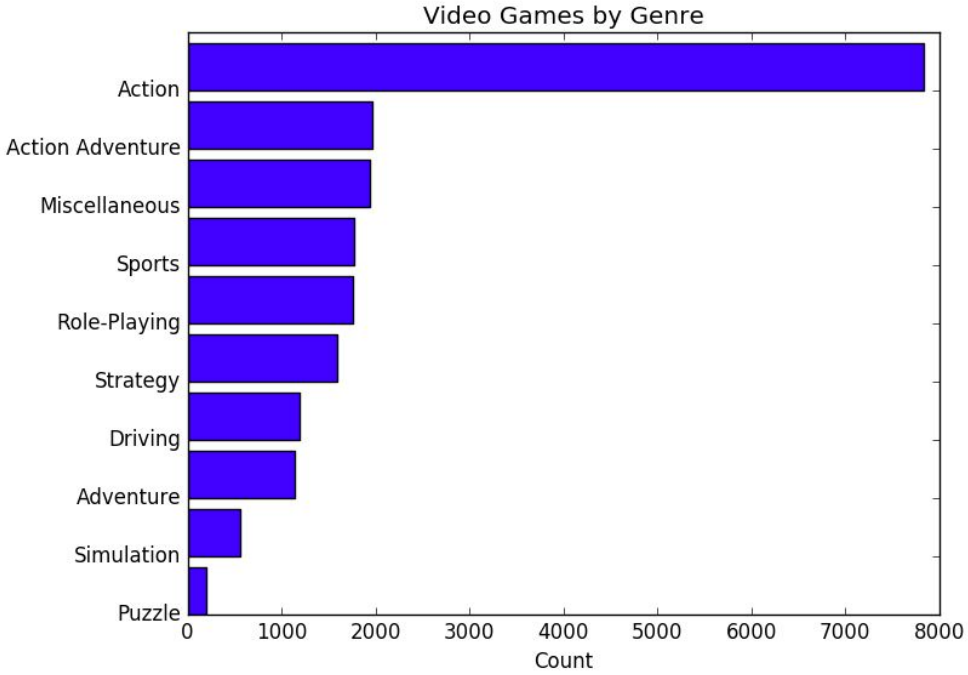




3. EDA: Video *Games*

- Action games and PC games are the most common video games



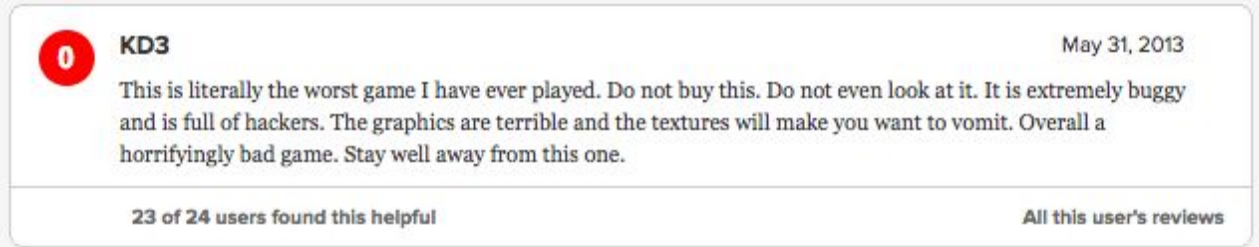


3. EDA: *Games - Interesting findings*

- A bad video game may cause people to leave more reviews

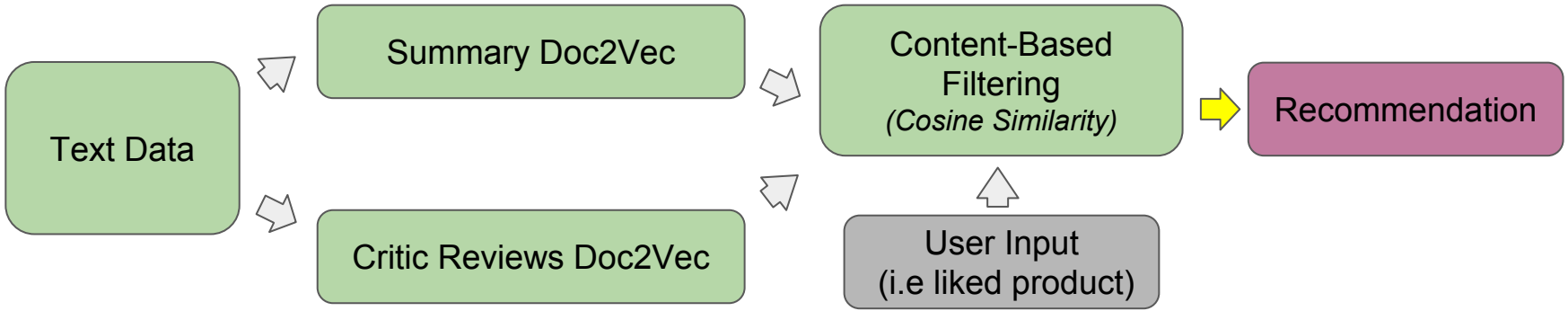
Top 5 Most Reviewed Games*(Count on Critics+Users)*

	uniqueID	count	name	AvgCritRat	AvgUserRat
0	7638	2266	Call of Duty: Modern Warfare 3	88.33	3.60
1	13089	2227	Diablo III	87.71	3.87
2	1389	1847	The Last of Us	94.83	9.24
3	13210	1761	The Elder Scrolls V: Skyrim	93.78	7.81
4	253	1619	Infestation: Survivor Stories (The War Z)	21.31	1.85



4. Content-Based Filtering: *unstructured text data is powerful!*

- Content-based filtering uses the item's metadata to make a recommendation
- Transformed text reviews into a vector representation using *Doc2Vec* algorithm in Python's Gensim library
  - Doc2Vec for item summaries and critic reviews
    - Over 500,000 reviews of 30,000 items aggregated into one corpus
- Computed cosine similarity to compare user input with database of items, returning items with the highest cosine similarity



4.b) Collaborative Filtering: *Pearson's Correlation*

Pearson Similarity Using Critic and User Ratings

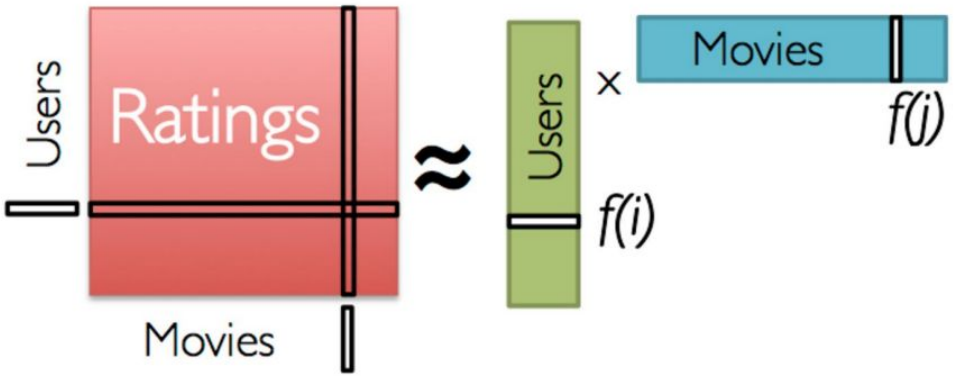
- Item to Item Matrix

- Restricted to items with more than 100 reviews to reduce the size of the matrix

	3	25	51	73	95	171	182
3	1	0.443913	0.308666	0.0381164	0.365199	0.184312	0.078321
25	0.443913	1	-0.00955871	-0.235418	-0.036264	-0.206055	0.187101
51	0.308666	-0.00955871	1	0.340238	0.262468	0.257613	-0.0237654
73	0.0381164	-0.235418	0.340238	1	0.326446	0.369903	0.366808
95	0.365199	-0.036264	0.262468	0.326446	1	0.305	0.152899
171	0.184312	-0.206055	0.257613	0.369903	0.305	1	0.282427
182	0.078321	0.187101	-0.0237654	0.366808	0.152899	0.282427	1
253	0.717126	0.862439	0.609682	0.204226	0.593271	-0.30922	0.35344
313	0.847922	0	0.255282	0.467417	0.929063	0.627763	0.403403
337	0	0.00663741	0	0	0	0.526075	0.221095

4.b) Collaborative Filtering: *SVD*

- Recommendation based on:
  - User's past ratings (explicit information)
  - History of like-minded users
- Problem: User-item matrix is very sparse!
  - Solution: SVD - latent matrix factorization of user-item matrix to find latent features
  - Used SVD's implementation in Scipy's library



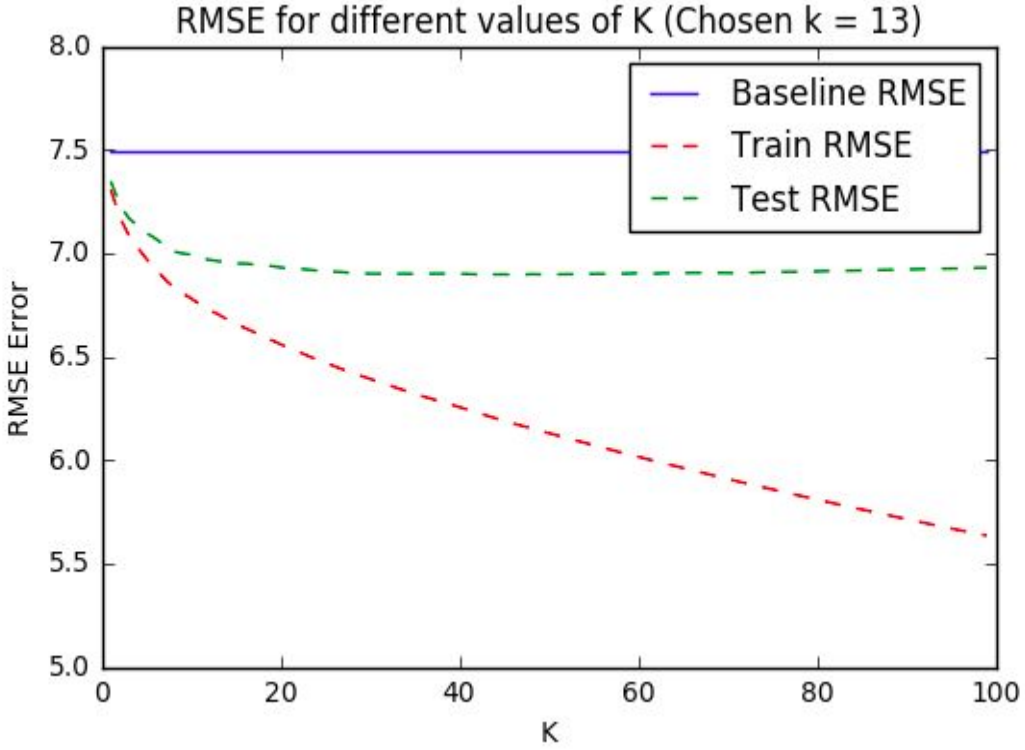
4.b) Collaborative Filtering: *SVD - Predicting User Rating*

- Predicted User Rating (formula)

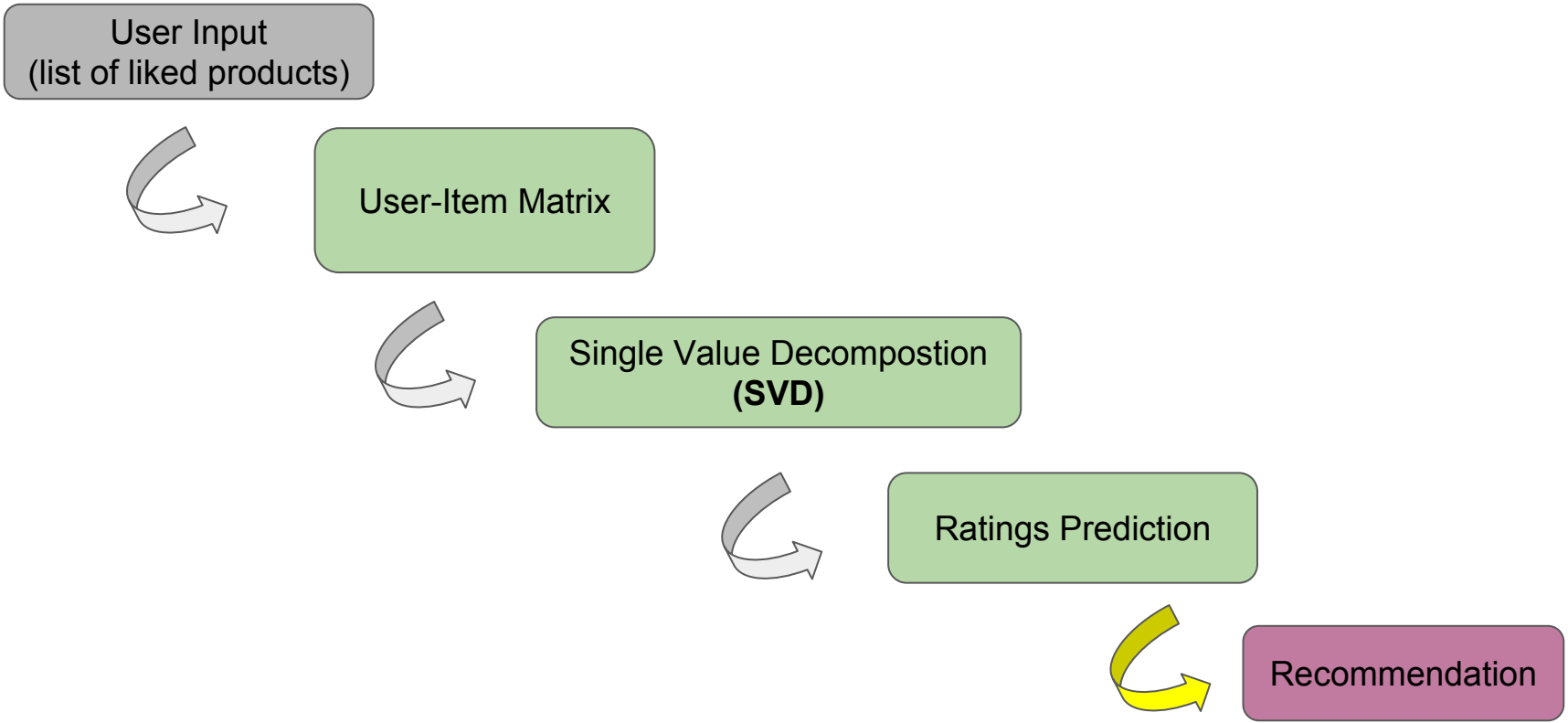
$$\mathbf{A}_{nxp} = \mathbf{U}_{nxn}\,\mathbf{S}_{nxp}\,\mathbf{V}^{\mathbf{T}}_{pxp}$$

- Best Model
  - K =13 (from scree plot)
  - Entire dataset RMSE: 6.07
  - Baseline RMSE: 7.5
  - Improvement: ~ 19%

$$\mathrm{RMSE} = \left[\left(\sum_{i=1}^{n}(y_i - \hat{y}_i)\right)\Big/ n\right]^{1/2}$$



4.b) Collaborative Filtering: *SVD - Recommendations*

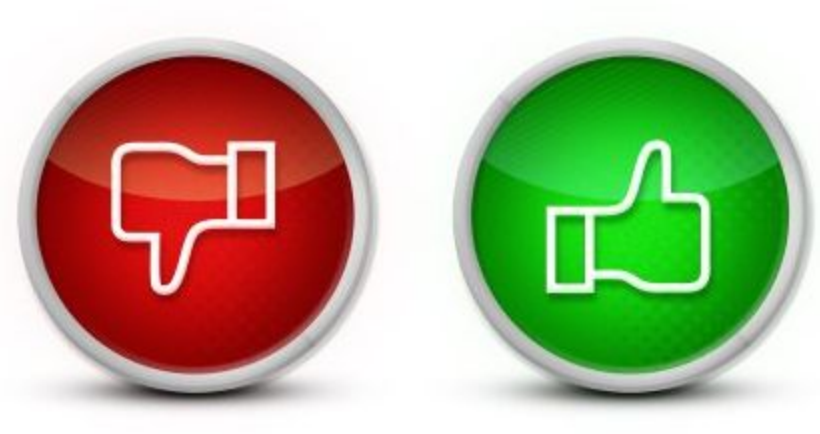


Note: Current version of app restricts user-item matrix to users with > 50 reviews

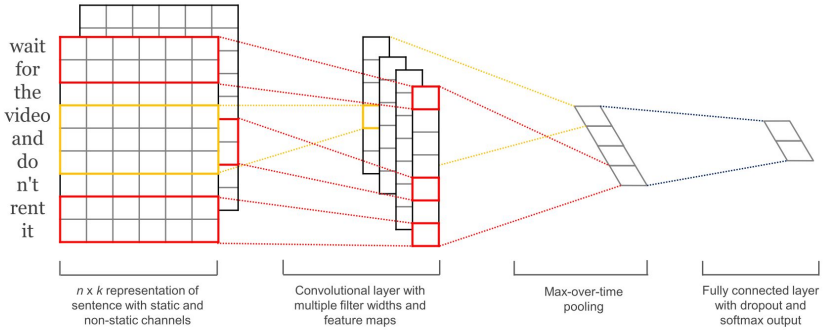
4.c) Sentiment Analysis: *Tested Machine Learning Models*

- Logistic Regression and SVM
  - Around 75% test accuracy
  - Doc2Vec to create features

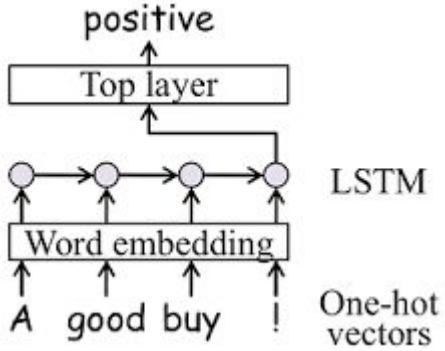
- Neural Network
  - 90% validation accuracy
  - Tokenized corpus
  - 10,000 word dictionary



4.c) Sentiment Analysis: *Neural Network Architecture*



+



- Tried Recurrent NN alone - poor results
- Tried Convolution NN alone - better results
- Found best results when combined
  - 2 Convolution and pool layers
  - 2 recurrent layers - LSTM (Long Short Term Memory)
  - 3 dense, fully connected layers
  - 3 epochs

4.c) Sentiment Analysis: *Best NN Results*

```
Train on 121233 samples, validate on 80822 samples
Epoch 1/3
121233/121233 [==============================] - 677s - loss: 0.3316 - acc: 0.8579 - val_loss
: 0.2735 - val_acc: 0.8834
Epoch 2/3
121233/121233 [==============================] - 685s - loss: 0.2412 - acc: 0.9041 - val_loss
: 0.2583 - val_acc: 0.8961
Epoch 3/3
121233/121233 [==============================] - 682s - loss: 0.2036 - acc: 0.9215 - val_loss
: 0.2423 - val_acc: 0.9010
Accuracy: 90.10%
```

5) Flask Web Application

- Web application built using Flask in Python
  - AngularJS: great framework for building dynamic web apps
  - Database: MySQL on AWS RDS
- Live demo

[Extra Slide] Future Improvements

- Hybrid Recommendation system
- Neural network to summarize text
- Improve score prediction
- Improve scaling efficiency
- A/B testing
  - Content vs collaborative
  - User onboarding
  - Click through rate