Amazon Beauty Products Recommendation Engine

Varsha Tripathi
Capstone Project
Springboard Data Science
Jan, 2020

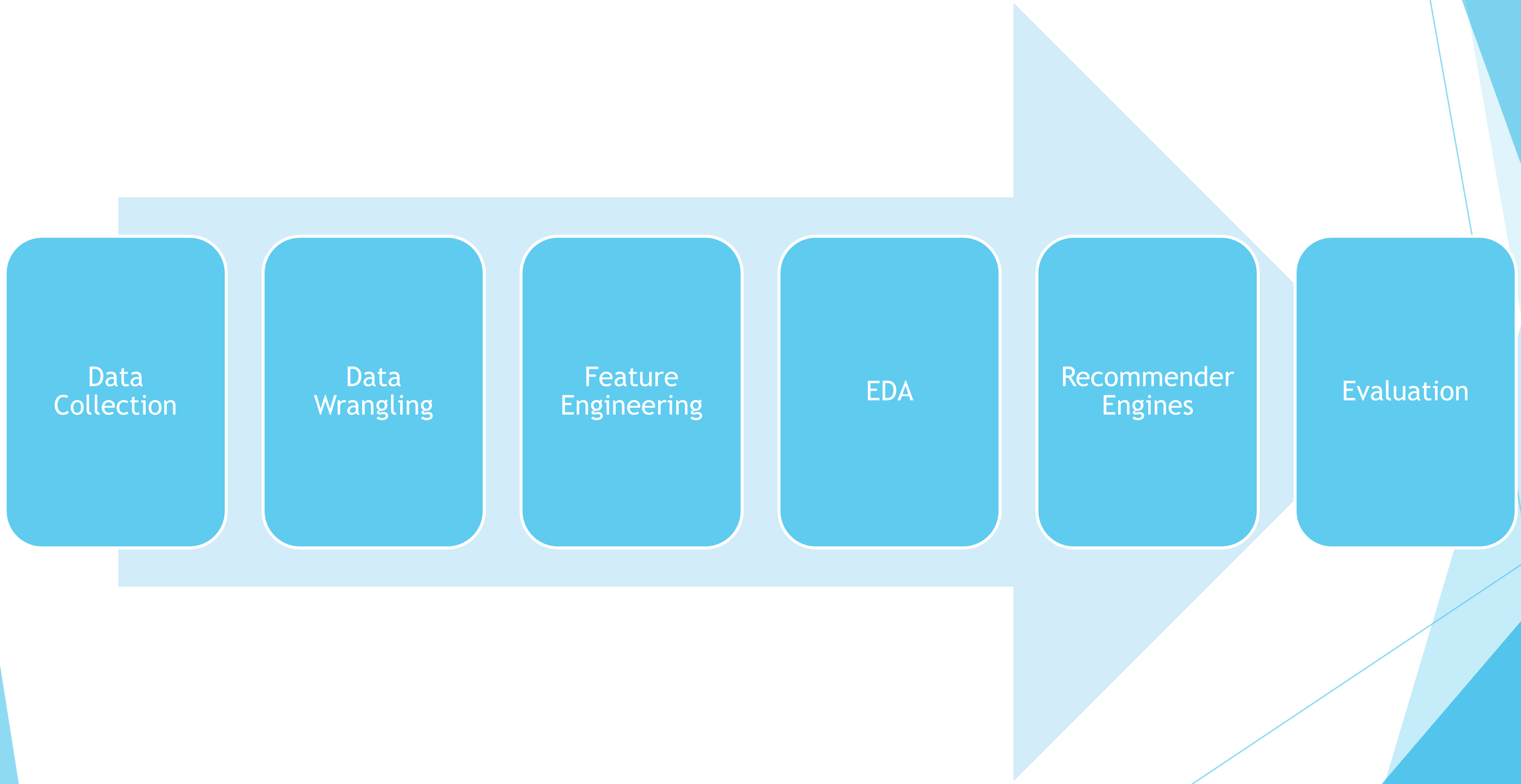
Recommender Systems

- Very commonly used, especially in the digital domain and e-commerce world.
- According to McKinsey, 35% of Amazon.com's revenue is generated by it's recommendation engine.
- Provide relevant suggestions to online users to make better decisions about their purchases.
- Amazon, Netflix, YouTube and Yelp use intelligent recommender systems to help customers take their brand experiences into their hands and make informed decisions about which products to buy or which movies to watch or which restaurants to eat at.
- Strong and smart recommender systems can have positive effects on user experience.
- Result into higher customer satisfaction and retention, and in turn boost revenues.

The Approach

- Data collection
  - json raw data files -> pandas dataframe -> csv files
- Data wrangling
  - Clean the data
  - Fix the missing values and duplicates
- Feature Engineering
  - Extract related count
  - Sentiment analysis of review texts
- EDA
  - Plot and explore the data
  - Check for imbalances and outliers
- Recommendation systems
  1. Popularity-based
  2. Content-based
  3. Collaborative filtering
  4. Hybrid methods
- Evaluation of Model Accuracy and Performance

Process Flowchart



The Client

- The recommendation system will be useful for Amazon.com as well as it’s customers.

  Amazon.com:

  can use this to make personalized recommendations to it’s customers

  using the product attributes and reviews from other customers

  Amazon.com customers:

  will receive personalized recommendations

  based on their ratings for products they have already used and

  based on their similarity with other customers

  smart recommendations will enrich their shopping experience

- The models can be used by any businesses such as Yelp, LinkedIn, Netflix, YouTube, Pandora to create a delightful user experience while driving incremental revenue.

The Dataset

- Publicly provided by Julian McAuley of UCSD
- Link: http://jmcauley.ucsd.edu/data/amazon/links.html
- Reviews timeline: May 1996 – Jul 2014
- Raw Data
  - 2 json files:

    reviews.json and meta.json
  - 2,023,070 reviews
  - 259,204 products
  - File size
    - Reviews file 1111 MB
    - Meta file 320 MB

Data Wrangling

- Loaded the 2 raw json compressed files into pandas dataframes
- Loaded the data from pandas dataframes to csv files

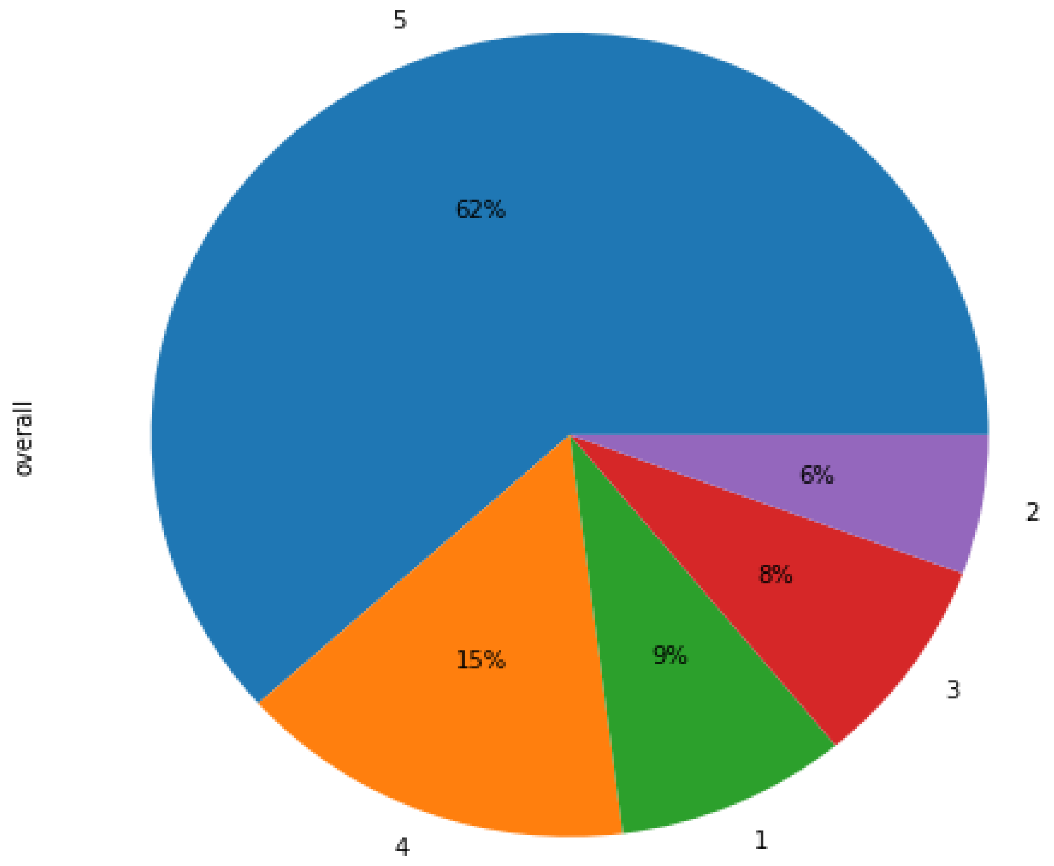
Data Cleaning

- No duplicates were found in meta file
- Dropped imUrl from meta file
- Dropped reviewerName from reviews file

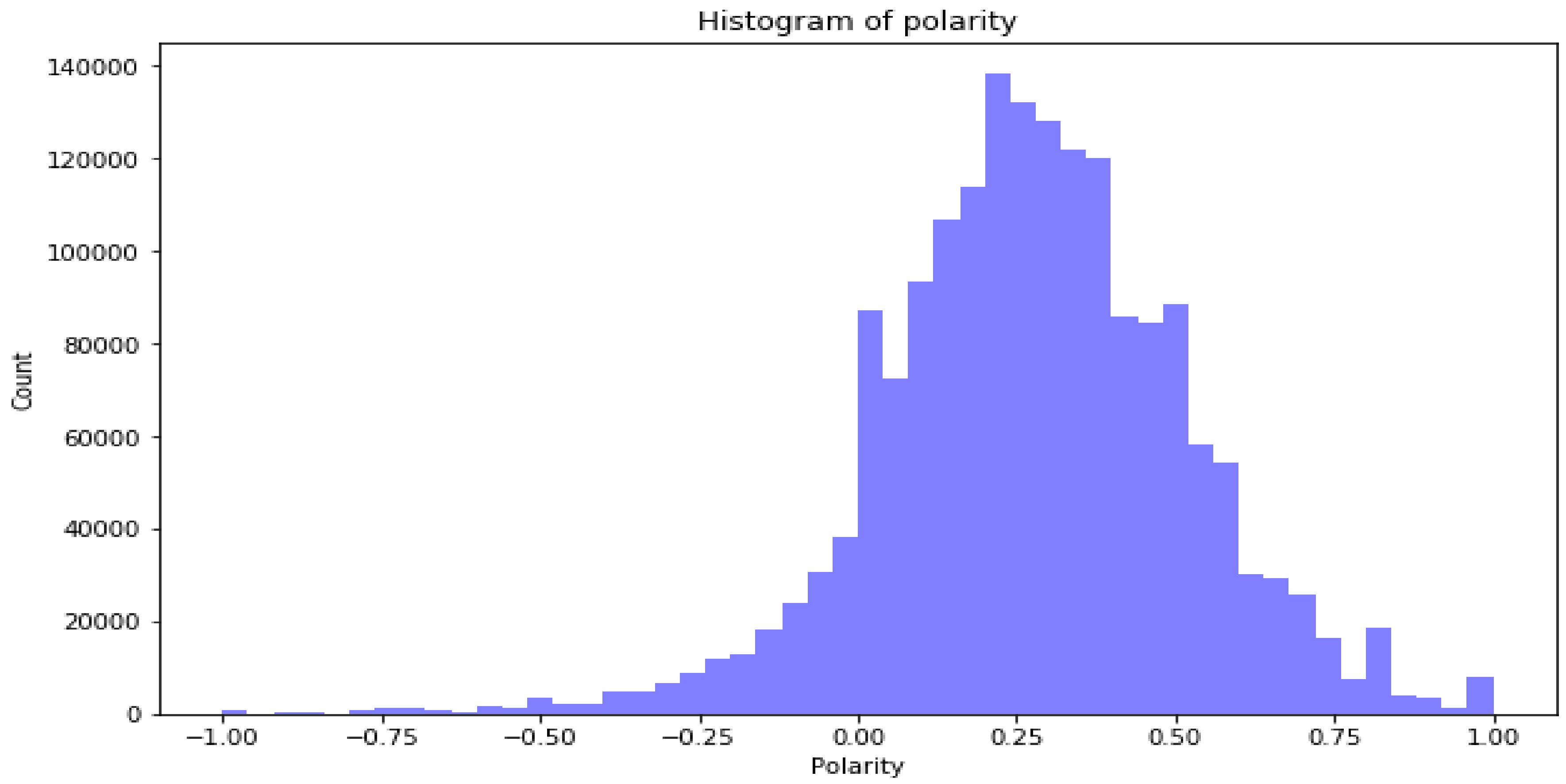
Feature Engineering

- **Meta file:**
  - More than 50% products missing brand information. Merged “brand” and “title” into “brand_title”
  - Extracted “beauty” and “health_personal_care” columns from “salesRank” column (nested dictionary)
  - Extracted “related_count” from “related” column (nested dictionary)
- **Reviews file:**
  - More than 50% products missing brand information. Merged “brand” and “title” into “brand_title”
  - Extracted “beauty” and “health_personal_care” columns from “salesRank” column (nested dictionary)
  - Combined “reviewText” and “summary” into “review” column
  - Extracted “related_count” from “related” column (nested dictionary)
  - Extracted “upvotes” and “downvotes” from “helpful” column (list)
  - Performed sentiment analysis on “review” column and created new feature called “polarity”
  - Extracted “main_cat” and “sub_cat” columns from “categories” (list)

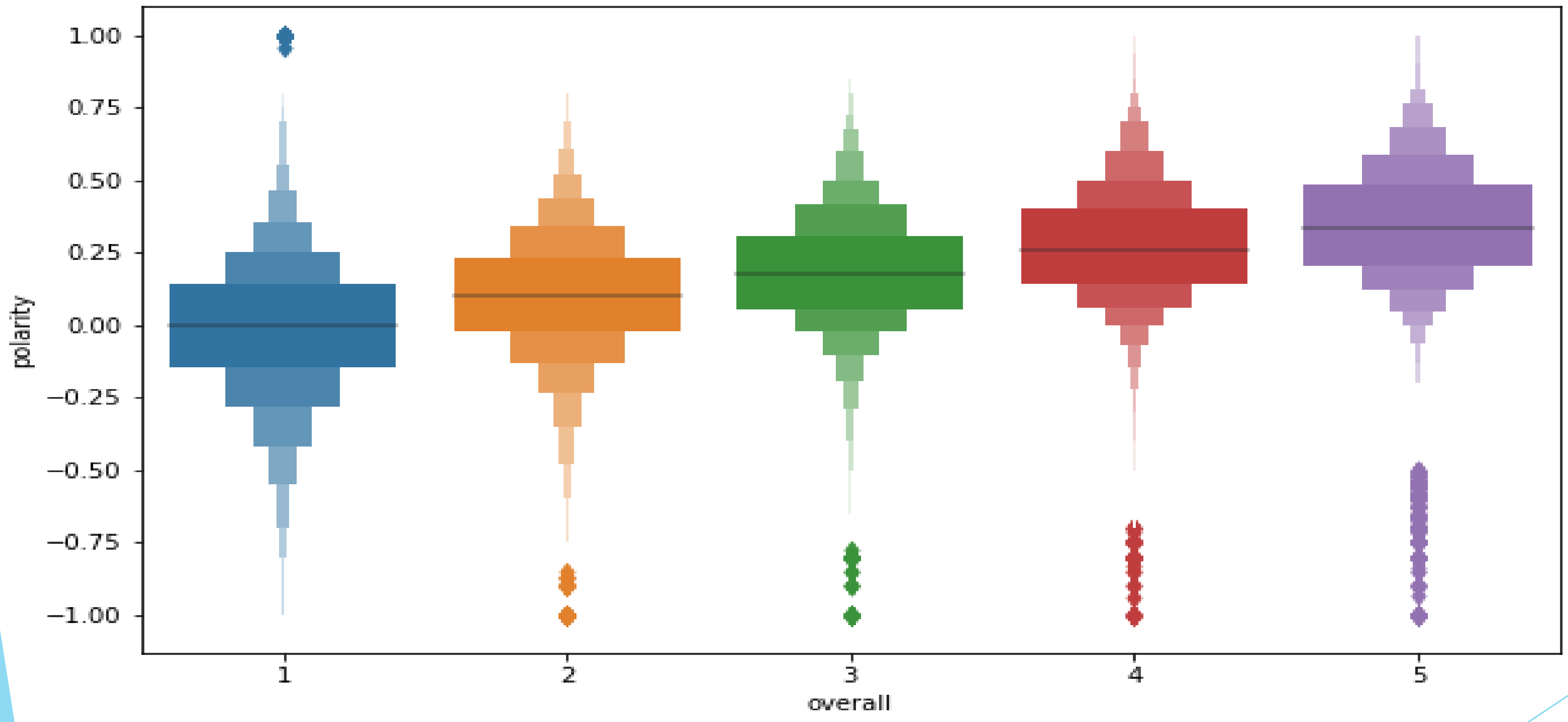
EDA - % of different overall values



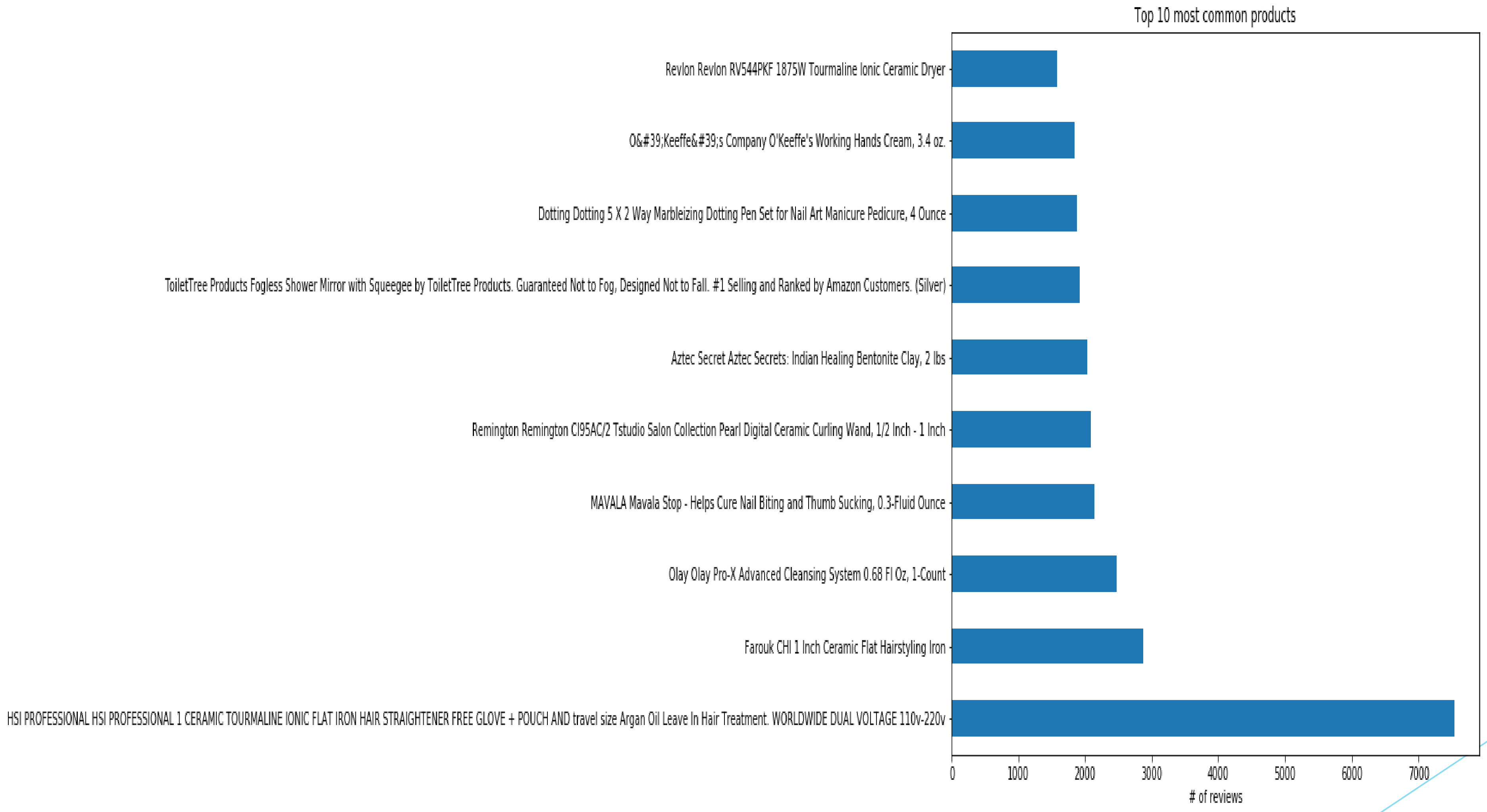
EDA – Histogram of polarity



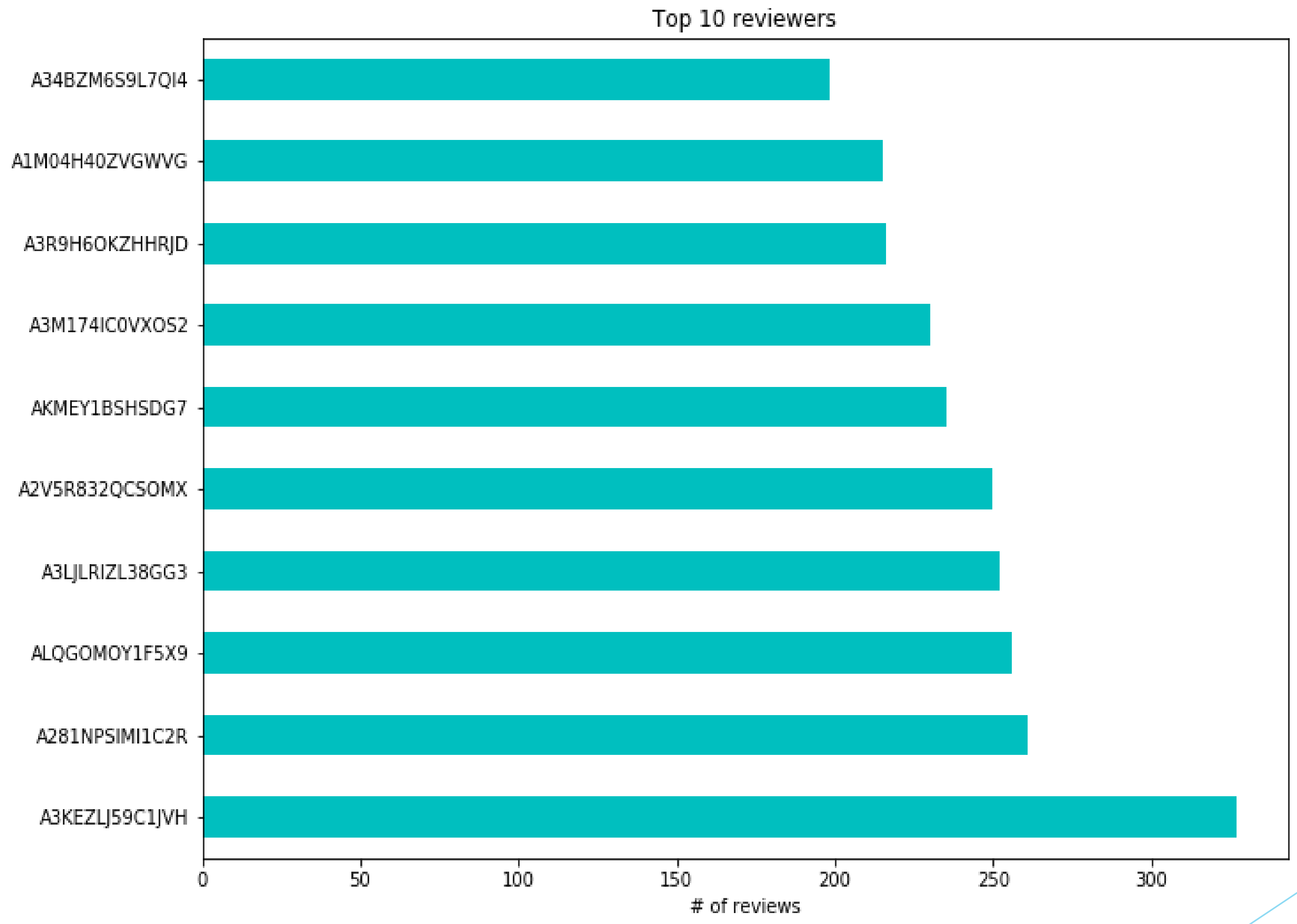
EDA – Polarity by overall score



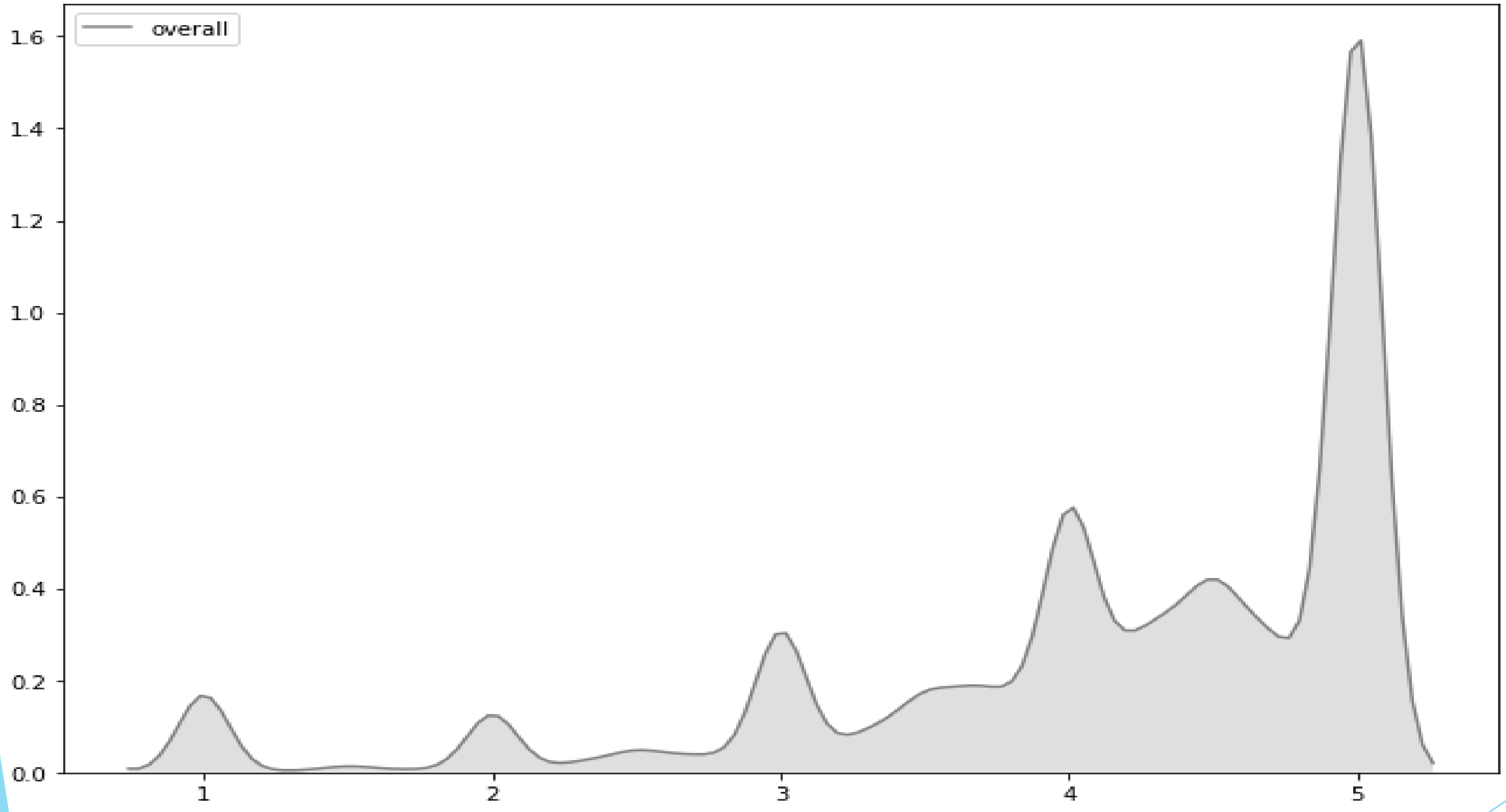
EDA - Top 10 reviewed products



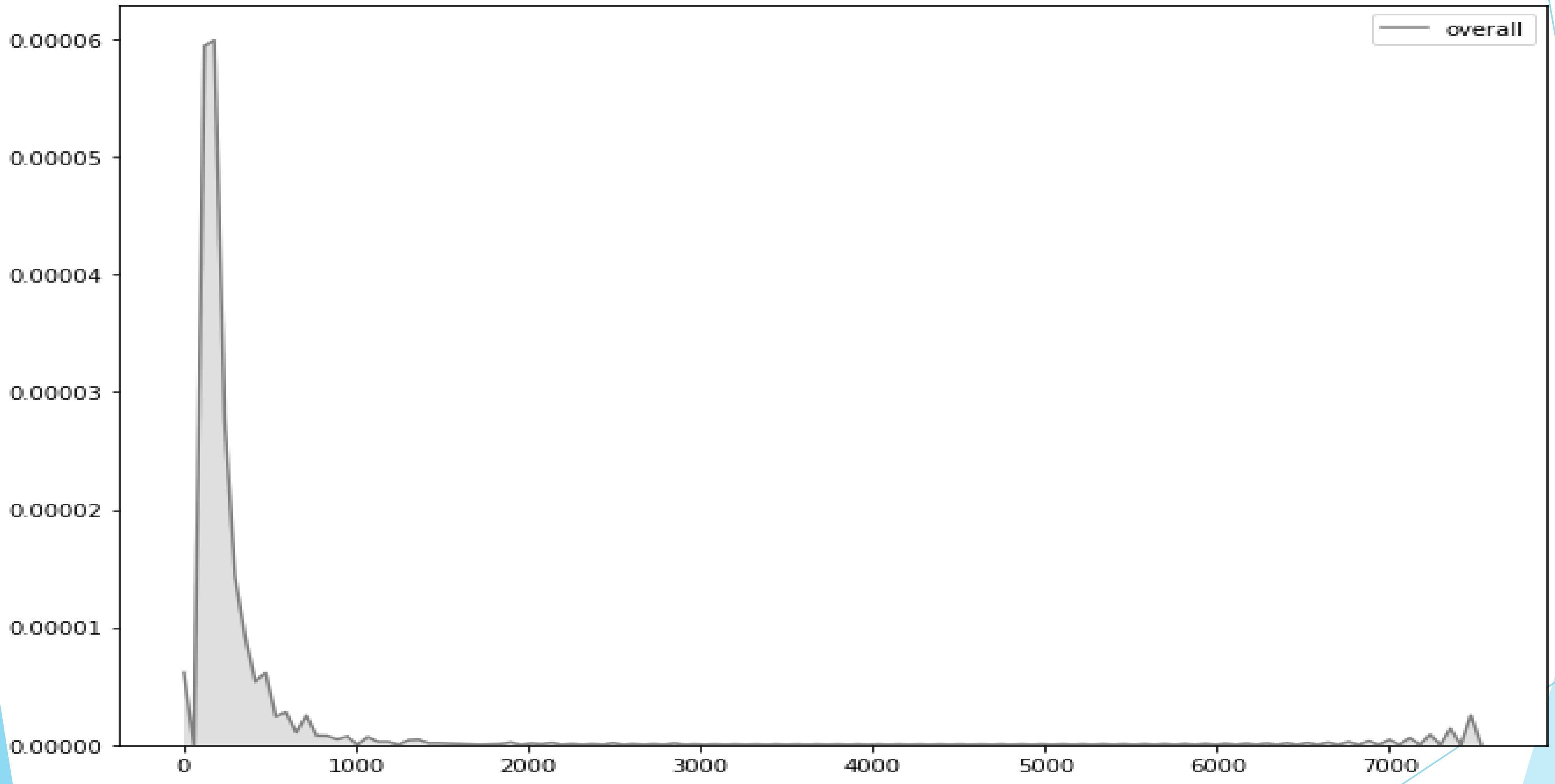
EDA - Top 10 reviewers



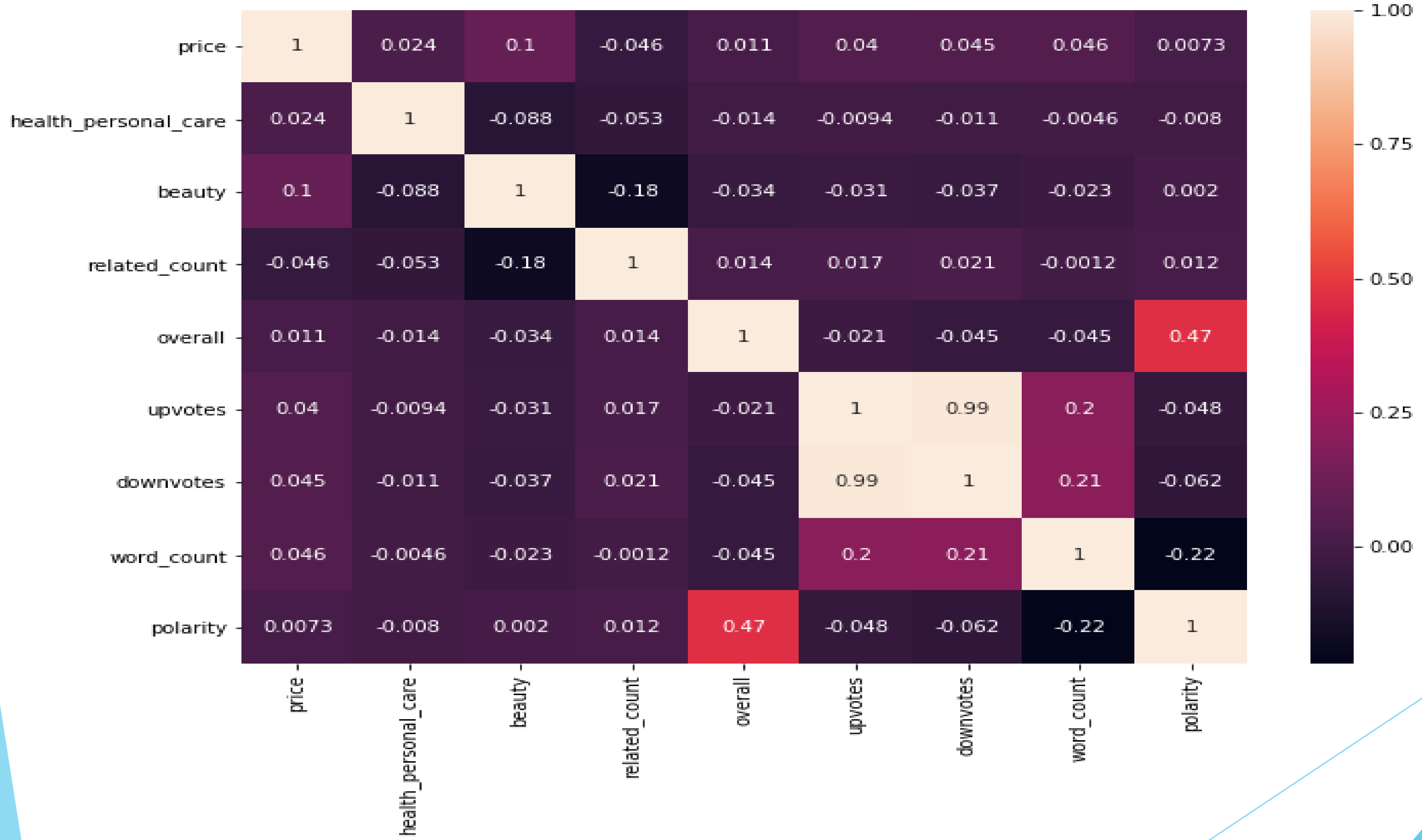
EDA – Mean rating KDE distribution



EDA – No. of ratings per product KDE distribution



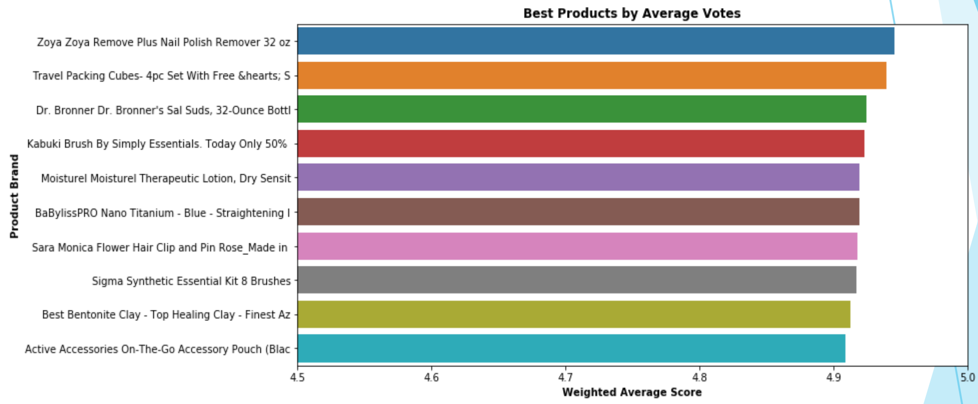
EDA - Correlation Matrix

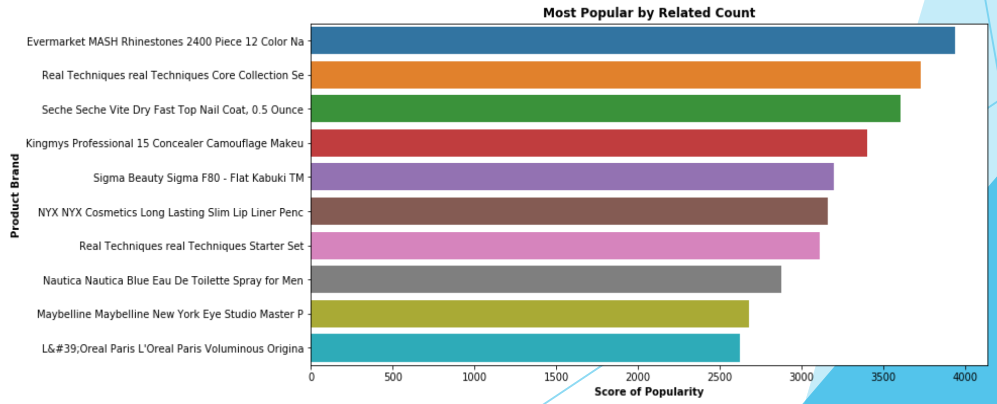


	price	health_personal_care	beauty	related_count	overall	upvotes	downvotes	word_count	polarity
price	1	0.024	0.1	-0.046	0.011	0.04	0.045	0.046	0.0073
health_personal_care	0.024	1	-0.088	-0.053	-0.014	-0.0094	-0.011	-0.0046	-0.008
beauty	0.1	-0.088	1	-0.18	-0.034	-0.031	-0.037	-0.023	0.002
related_count	-0.046	-0.053	-0.18	1	0.014	0.017	0.021	-0.0012	0.012
overall	0.011	-0.014	-0.034	0.014	1	-0.021	-0.045	-0.045	0.47
upvotes	0.04	-0.0094	-0.031	0.017	-0.021	1	0.99	0.2	-0.048
downvotes	0.045	-0.011	-0.037	0.021	-0.045	0.99	1	0.21	-0.062
word_count	0.046	-0.0046	-0.023	-0.0012	-0.045	0.2	0.21	1	-0.22
polarity	0.0073	-0.008	0.002	0.012	0.47	-0.048	-0.062	-0.22	1

Recommender: Popularity-based

- **Non personalized and recommends the same top-N products to everyone**
- **Simplest type of recommender, but not very useful or effective**
- **Popularity can be determined by:**
  - Most rated item (based on rating count)
  - Average overall rating
  - Sales rank based on category
  - Weighted average for each item's average rating
  - Combination of best rated and most popular items





Recommender: Content-based

- Compute item vectors using attributes from the meta dataset
- Vectors created using TfIdf and CountVectorizer
- Recommends an item based on its features and how similar they are to features of other items in the data set
- Similarity of item attributes can be determined using cosine similarity or Euclidean distance or nearest-neighbor algorithms
- Independent of interaction history of other users and captures specific interests of a user
- Tends to over-specialize and will recommend only items similar to those that the user has previously used and rated.
- Evaluation metrics:
  - RMSE
  - R-squared

Recommender: Collaborative

- Ignores user and item attributes and instead focuses on User-Item interactions
- Determines similarities between 2 users or 2 items, using the historical ratings given to items by various users and makes recommendations on the basis of that
- Sparse utility matrix is built using customer rating for items
- Types:
  - Memory-based
    - User-user collaborative filtering
    - Item-item collaborative filtering
  - Model-based: Perform matrix factorization to compress the sparse matrix to improve accuracy and performance
- Cold start problem: New items cannot be rated until they have been rated by someone

Recommender: Hybrid

- Combine the strengths of 2 or more filtering methods
  - Incorporate content-based techniques into a collaborative approach
  - Incorporate collaborative techniques into a content-based approach
- Improves recommender performance and overcomes cold-start problems
- 7 ways to apply hybrid technique:
  - Weighted
  - Feature combination
  - Switching method
  - Mixed
  - Feature Augmentation
  - Cascade
  - Meta-level

Potential Improvements

- Use more features from the 2 datasets to build the personalized recommender
- Apply NLP techniques on item description and review text for a richer and more accurate recommender
- Improve computation using cloud distributed systems and Spark
- Use the entire data set (without dropping any rows)
- Build a user interface screen to input an item or a user, that will output a blend of top-N non-personalized and personalized recommendations on the same screen
- Improve the rating metric by adjusting overall score using review time and age relevance of the rating
- Improvise using Deep Learning methods

Acknowledgement

- Max Sop (mentorship)
- Julian McAuley of UCSD (free dataset)
- Springboard team (curriculum and administrative support)
- Recommender Systems specialization course from Coursera
- Building a Recommendation System with Python Machine Learning & AI
- Machine Learning and AI Foundations: Recommendations

Reference

- Project Final Report: https://github.com/venustrip/Amazon-Recommendation-Engine/blob/master/final_report.ipynb
- Data Wrangling: https://github.com/venustrip/Amazon-Recommendation-Engine/blob/master/clean_meta.ipynb
- EDA: https://github.com/venustrip/Amazon-Recommendation-Engine/blob/master/milestone_report.ipynb
- Popularity-based recommender: https://github.com/venustrip/Amazon-Recommendation-Engine/blob/master/recommender_popular.ipynb
- Personalized content-based recommender: https://github.com/venustrip/Amazon-Recommendation-Engine/blob/master/recommender_content.ipynb
- Personalized collaborative recommender: https://github.com/venustrip/Amazon-Recommendation-Engine/blob/master/recommender_collab.ipynb
- Hybrid recommender: https://github.com/venustrip/Amazon-Recommendation-Engine/blob/master/recommender_hybrid.ipynb