Amazon Recommender System

Advisors: Ilkay Altintas, Julian McAuley

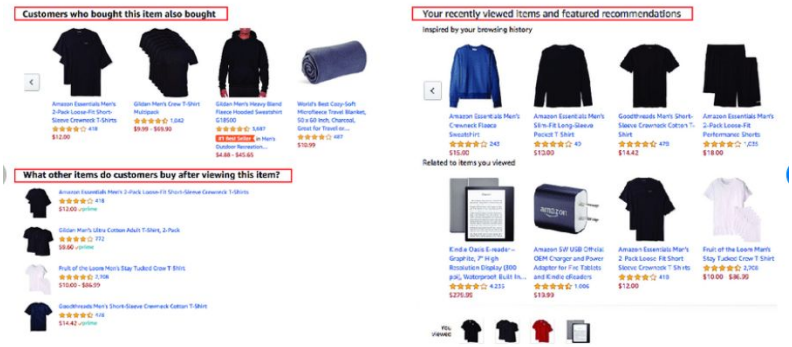
Team Members : JH (Janghyun) Baek, John Tsai, Justin Shamoun, Muriel Marable, Ying Cui

Abstract

Recommender systems are algorithms that suggest relevant items to users based on data. They generate large revenue for the modern e-commerce industry. 35% of Amazon web sales were generated through their recommended items [source: McKinsey]. This study aims to construct an apparel recommender system for Amazon users through user-rating history, product images and product title text. Multiple deep learning models were built on both readily-available and engineered datasets resulting in a multi-step recommender system. Tableau and a web app are used to display results, along with evaluation measurements.

Introduction

Recommender systems are used by E-commerce sites to suggest products to their customers and to provide consumers with information to help them decide which products to purchase. The products can be recommended based on the top overall sellers on a site, on the demographics of the consumer, or on an analysis of the past buying behavior of the consumer as a prediction for future buying behavior. Amazon currently uses item-item collaborative filtering, which scales to massive datasets and produces high quality recommendation systems in real time. This system is a kind of an information filtering system which seeks to predict the "rating" or preferences which user is interested in.



An example of Amazon recommender system.

Product recommendations tailored to a user are more likely to lead to higher conversion. Recommended products account for 35% of Amazon revenue (MacKenzie) . Furthermore, users want recommendations of similar items to help discover new products, or compare items.

Amazon went into the apparel business in 2002. It acquired Shopbop in 2006 and Zappos in 2009, an online shoe retailer. At first, their apparel business faced the challenge of people not trusting purchasing apparel online since they would like to try on an item first. Another challenge was Amazon was perceptively not a trendy clothing brand. But as of 2019, Amazon became the nation's top fashion retailer by 2018, beating out Walmart and Target. Its share of fashion shoppers is 61% in 2018 (Danziger). An advantage Amazon has over other retailers is that it is set up as a data company. It is a leader in collecting, storing, processing, and analyzing personal information from customers as a means of determining how they spend their money. Having this capability has made Amazon the leader in apparel retail.

Challenge

The challenge for our group is to create an apparel-specific recommender system that is personalized to an Amazon user which aims to enhance customer experience. Personalized recommendation based on user preference has a higher likelihood of conversion than general recommendations. In order to meet this challenge, we first have to figure out which machine learning algorithms to use in order to create an apparel recommender system for specific Amazon users. We would need to figure out which features to use for this task.

Additionally, we would like our recommender system to recommend similar items relative to the item that a user is currently viewing. This task would be based on product features similarity. This would require a separate modeling task from the first task.

Lastly, we would need to create a customer-facing product which will provide recommendations to a given user. We would need to assess dashboard tools for this task.

Questions

Some of the questions that we formulated in order to address our challenges are:
Which is the best algorithm to find similarity between users and cluster them and label them?
How do we generate intention values (intention to purchase a product)?
How do we find similarity based on clothing style and how do we measure similarity value?
Could price be used in the model?
What is the best type of database or tool for information retrieval in order to process the recommender system in real time?

Hypotheses

The specific hypotheses in our study were:

Hypothesis 1: We can mimic customers' behavior of purchasing products based on product ratings from the dataset.

Hypothesis 2: We can identify products that users are more likely to purchase when we match the style of the users' current selection.

For hypothesis 1, the consumer review rating provides a powerful source of information about a user's preferences. We postulate that we can utilize this to create predictions on users' preferences.

For hypothesis 2, we postulate that a online apparel shopper may not know exactly what they are looking for when they go to the Amazon site. They may have some general preferences like color or signature style which they may look for using keywords on the search bar. From the results, they will gravitate towards specific selections. Using this behavior, our recommender system will recommend products that are similar in style with the customer's current selection. Similarity scores will be generated between products in order to measure style similarity.

Related Works

Our work builds upon previous work done in recommender systems. Amazon, a leader in data systems already has one of the most successful recommender systems in place. The system uses item-based collaborative filtering which gives recommendations based on items that the user has purchased or has rated, which is then paired with similar items. Item-based collaborative filtering is different from user-based collaborative filtering which predicts user preferences by utilizing user-generated signals, like explicit item ratings, that were gathered from other users (Linden et al.) .

Another popular approach is a content-based recommender which tries to recommend items similar to those a given user has liked in the past. For apparels, content-based suggestions intuitively make sense since an online shopper would like to 'browse' and discover similar items before buying (Kumar).

In more recent years, Deep learning has been demonstrating its effectiveness on recommender systems. The paper Deep Learning Based Recommender System by Zhang, et al surveys different Deep Learning techniques for this type of application, such as Multilayer Perceptron (MLP), Autoencoder (AE), Convolutional Neural Network (CNN), Recurrent Neural Network (RNN) to name a few. They argue that Deep neural networks are effective because they are capable of modeling the nonlinearity in data with nonlinear activations like sigmoid, relu, etc. This makes it possible to capture complex user-item interaction patterns. Deep neural networks also enable automatic feature learning from raw data, lessening the need for labor-intensive feature engineering (Zhang et al.).

The paper Deep Matrix Factorization Models for Recommender Systems talks about using Deep Matrix Factorization Models (DMF) for recommender systems. DMF is a matrix factorization model with a neural architecture. Users and items are projected into low-dimensional vectors in a latent space through the neural network architecture (Xue et al.).

Graph-based recommender system is a different type of recommender system that does not make use of collaborative filtering or content-based approach. It naturally combines the two approaches, resulting in a hybrid mode without having to use a top-level classifier, ranker or regression model (Huang et al.).

Team Roles and Responsibilities

JH Baek
Project Manager, Modeling - Deep Learning
A natural-born leader, JH took on the role of Project Manager of the group. He took care of planning and logistics, led the team meetings and stand-ups. JH's leadership was essential to the success of the project.

John Tsai
Storyteller, Visualizations, Data Exploration
As the Tableau expert in the group, John is the visualization lead. He understands Tableau's inner workings. With this, paired with his natural ability to infer information from sometimes limited data, he is the logical choice to become the group's main story teller.

Justin Shamoun
Data Architect, Budget Manager, Modeling - Neo4j
Proficient in AWS, Justin is the natural choice to become the group's data architect and budget manager. He is able to map out complex data processes and workflows and break it down into simpler structures.

Muriel Marable
Modeling - NLP, Web Scraper
As a tenacious, inquisitive learner, Muriel took on some of the more arduous tasks like web scraping and Natural Language Processing, which is always labor-intensive. She is the group's integrator, the one who brings people and information together as a whole.

Ying Cui
Modeling - Image Processing, Tensorflow
Ying is the jack-of-all trades of the group. She is a quick learner who can do anything that others can and also what others can't. If there's anything that we need to get done quickly, Ying steps up to the plate.

Data Acquisition

Most of the data used in this project is precomputed and collected from other processes. The data is available in the recommender systems data set repository which contains additional sources. For this project we are focused on the Amazon reviews dataset and even more specifically the “Clothing, Shoes, and Jewelry” Category. The dataset is publicly available and can be downloaded here:

Amazon Reviews Dataset: https://nijianmo.github.io/amazon/index.html

This dataset has been computed over various years from techniques such as web scraping. It is used in conjunction with research done in Professor Julian McAuley Lab and most recent versions contain the following details.

Dataset Statistics

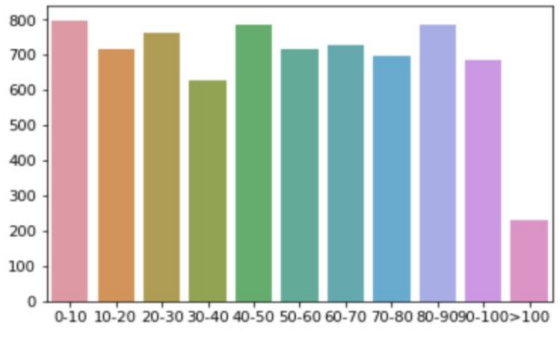
The dataset is large in nature and can be used for a variety of applications. Because of the size, best practices have shown us to start with a small sample size and iterate at each step. For our particular solution, we begin by breaking the dataset down by category, as well as sub category. Version one of our solution was built using only the “Shoes” section in the “Clothing, Shoes, and Jewelry” Category and then scale to the “Clothing and Jewelry” Categories later on.

Ratings	82.83 Million
Users	20.98 Million
Items	9.35 Million
Timespan	May 1996 - July 2014

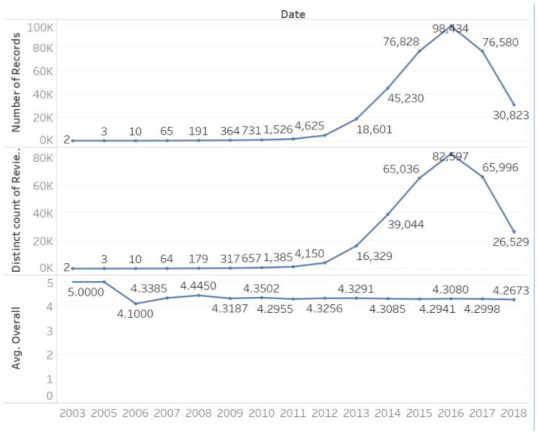
Per-category data - the review and product metadata for each category.

Amazon Fashion	reviews (883,636 reviews)	metadata (186,637 products)
All Beauty	reviews (371,345 reviews)	metadata (32,992 products)
Appliances	reviews (602,777 reviews)	metadata (30,459 products)
Arts, Crafts and Sewing	reviews (2,875,917 reviews)	metadata (303,426 products)
Automotive	reviews (7,990,166 reviews)	metadata (932,019 products)
Books	reviews (51,311,621 reviews)	metadata (2,935,525 products)
CDs and Vinyl	reviews (4,543,369 reviews)	metadata (544,442 products)
Cell Phones and Accessories	reviews (10,063,255 reviews)	metadata (590,269 products)
Clothing Shoes and Jewelry	reviews (32,292,099 reviews)	metadata (2,685,059 products)
Digital Music	reviews (1,584,082 reviews)	metadata (465,392 products)
Electronics	reviews (20,994,353 reviews)	metadata (786,868 products)
Gift Cards	reviews (147,194 reviews)	metadata (1,548 products)
Grocery and Gourmet Food	reviews (5,074,160 reviews)	metadata (287,209 products)
Home and Kitchen	reviews (21,928,568 reviews)	metadata (1,301,225 products)
Industrial and Scientific	reviews (1,758,333 reviews)	metadata (167,524 products)
Kindle Store	reviews (5,722,988 reviews)	metadata (493,859 products)
Luxury Beauty	reviews (574,628 reviews)	metadata (12,308 products)
Magazine Subscriptions	reviews (89,689 reviews)	metadata (3,493 products)
Movies and TV	reviews (8,765,568 reviews)	metadata (203,970 products)
Musical Instruments	reviews (1,512,530 reviews)	metadata (120,400 products)
Office Products	reviews (5,581,313 reviews)	metadata (315,644 products)
Patio, Lawn and Garden	reviews (5,236,058 reviews)	metadata (279,697 products)
Pet Supplies	reviews (6,542,483 reviews)	metadata (206,141 products)
Prime Pantry	reviews (471,614 reviews)	metadata (10,815 products)

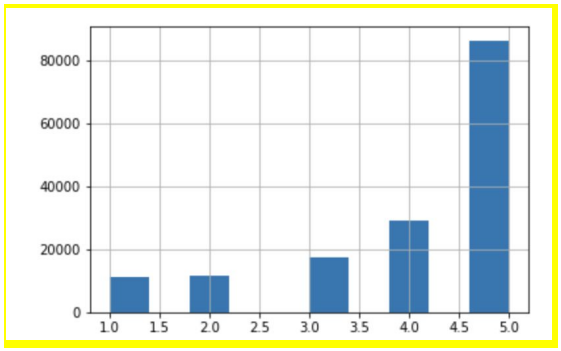
- Distribution of review length



- Trends over time



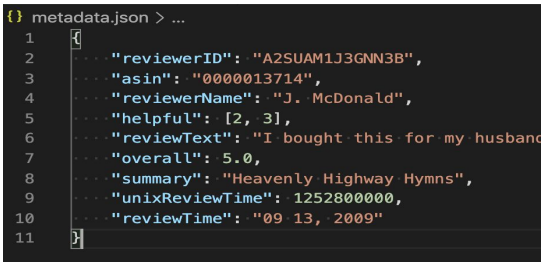
The two charts shown above illustrate some general statistics on the review length (left) and trends (right) over time. The review length was similar to a uniform distribution. A spike in review counts and users were observed from 2014 to 2016. The charts shown below indicated that review ratings were skewed to the lower rating, meaning that the average review rating would be on the higher end of the spectrum. In fact, the majority of the reviews were 5 stars.



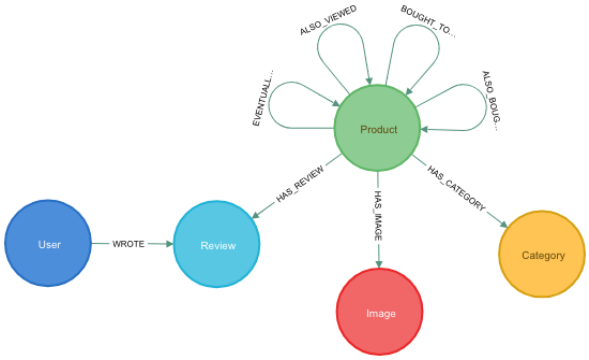
The following metadata can be used to summarize datasets features:

Reviews and Ratings
Item to Item relationships
Timestamps
Helpfulness votes
Product Images and CNN Features
Price
Category
Sales Rank

Amazon Reviews Data Example



The data is formatted as one review per line as in the above example.



Additional Data Sources

Metadata

The metadata is also provided from the same repository and it includes product description, price, sales, rank, brand, and title. The data source is used as an input for our NLP model, where we extract similarity features based on the products title text.

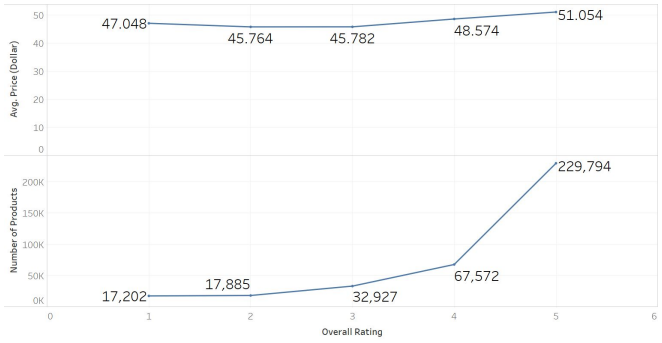
Amazon Web Site

The amazon website is also a data source for the project and specifically feeds into the Image-based model pipeline. As per the required data that is outputted from the first layer in our network, the images for those products are scraped and stored on the platform.

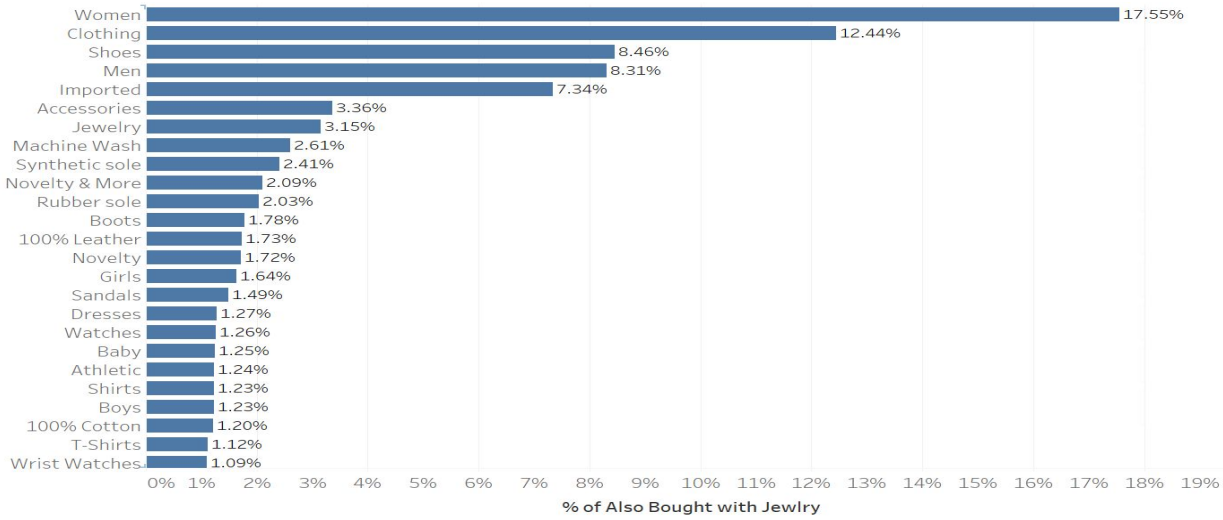
By combining data from these three different sources, it allows us to tailor our recommendations to both visual and relational components. Specifically, we are able to train our model for style based similarity features by combining it with our review ratings relational model. The specific techniques used to collect the data include access through APIs and Web Scraping.

Evaluate statistical inference of observed trends

After augmenting the various datasets, we explored some numeric variables to see if there's any high level statistical inference that could be identified. For simplicity purposes, this part of EDA focused on a small subset of data inside the jewelry category.

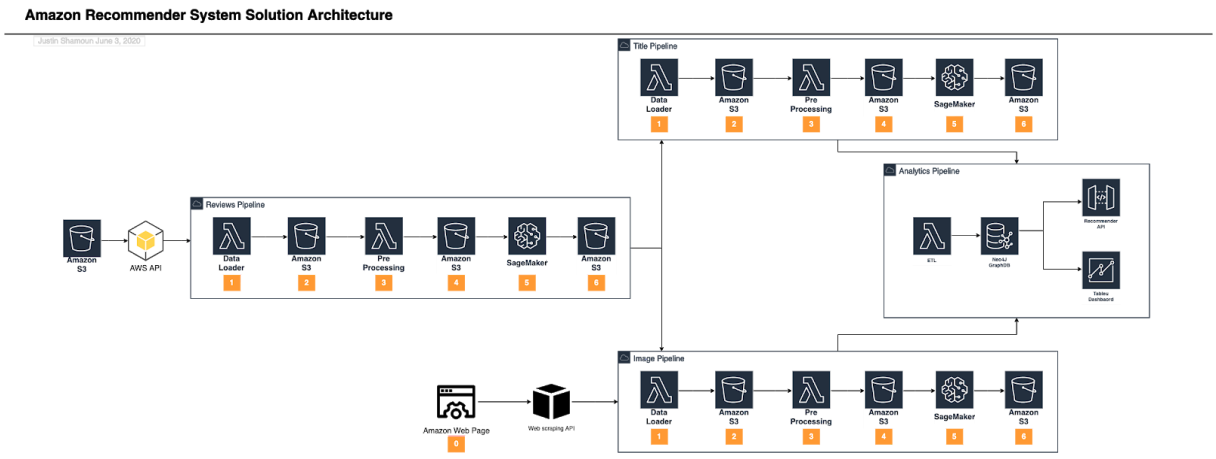


The first thing that caught our attention was that the majority of review ratings were 5 stars. We also see some positive correlation between rating and price in the jewelry category. We also looked into which other categories were bought most frequently with jewelry. The result is shown on the graph below:



From the graph we observed some trends: jewelry products were often bought with Women/Clothing products. While this information seems intuitive, it still provides a valuable message that the correlation exists, and could be used for our recommender system.

Amazon Recommender System Solution Architecture



Data Processing

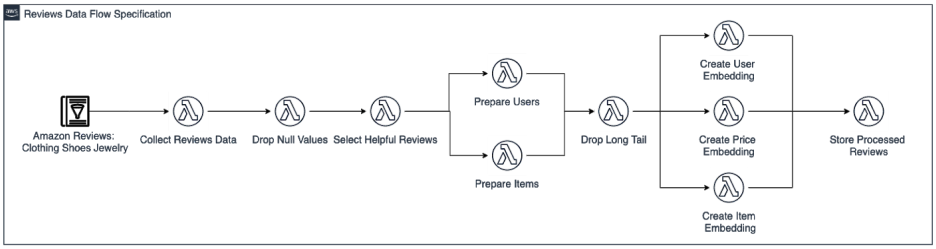
For data processing, we used AWS Glue for processing source data and preparing data sets for exploration. This job is triggered after the schema of the source data is identified. The following tables are constructed in the spark job:

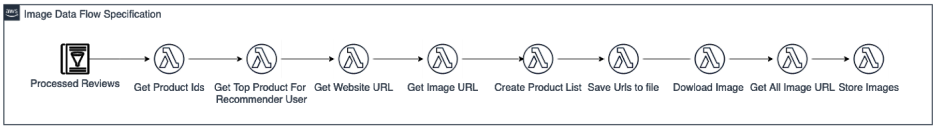
reviews → Users
reviews → Reviews
ratings → Ratings
5-core → 5-Core
metaData Products
Images → Images
metaData → Categories
metaData → related
metaData → feature

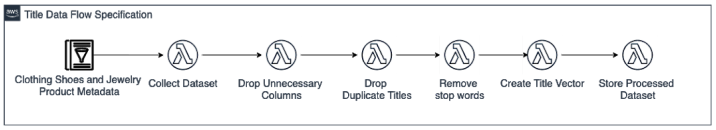
The data is then stored in the s3 standardization zone in parquet format and a Glue Crawler is configured to monitor the schema, as shown in the diagram below.

Amazon Recommender System Data Flow

Justin Shamoun June 3, 2020







Feature Engineering and Data Modeling

ReviewData

Columns	Description	Example
asin	product unique ID	B00EK1QJZK
image	URL to image (if exist)	['https://images-na.ssl-images-amazon.com/images/I/71ogJS7lG+L._SY88.jpg']
overall	overall rating	5
reviewText	review text	I did not handsome, but I put on these shoes, I instantly handsome
reviewerID	reviewer's unique ID	A1VZNQ2ITQIR2G
style	product style	{'Size:': ' 8', 'Color:': ' Blue Depths'}
summary	review summary	Very good!
Date	review posted date	4/24/2017
verified	review verified or not	TRUE
vote	# of vote on review	9
reviewLength*	length of review	100

**feature engineered*

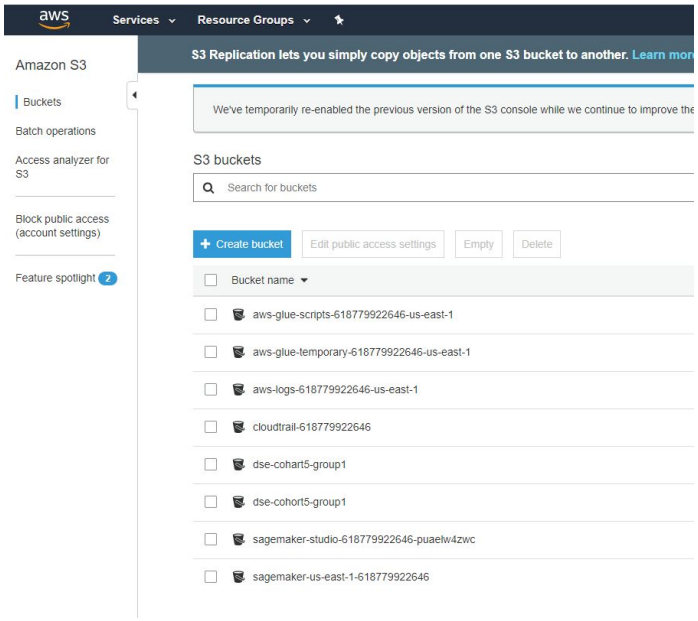
MetaData

Columns	Description	Example
Category	product category	['Women', 'Shoes', 'Athletic']
Description	product description	['This super nice t-shirt is made of 100% cotton
Title	product title	The Goozler Best. Dad. Ever. Funny Father's Day Unisex T-Shirt
brand	product brand	The Goozler
feature	product features	['100% Cotton Preshrunk Jersey Knit tees]
rank	product rank	14,925inClothing,ShoesJewelry(
asin	product ID	B007RGD304
image	product image (if exist)	['https://images-na.ssl-images-amazon.com/images/I/41Y2skHOlsL._US40_.jpg']
also_view	other products that were viewed	['B016IL4A3W', 'B075ZL8CPF']
price	product price (in dollar)	$11.55 - $21.24
fit	product fit (if exist)	class="a-normal a-align-center a-spacing-small">
also_buy	other products that were bought	['B016IL4A3W', 'B075ZL8CPF']
main_cat	product main category (if exist)	Baby
price-average*	computed average product price	16.4

**feature engineered*

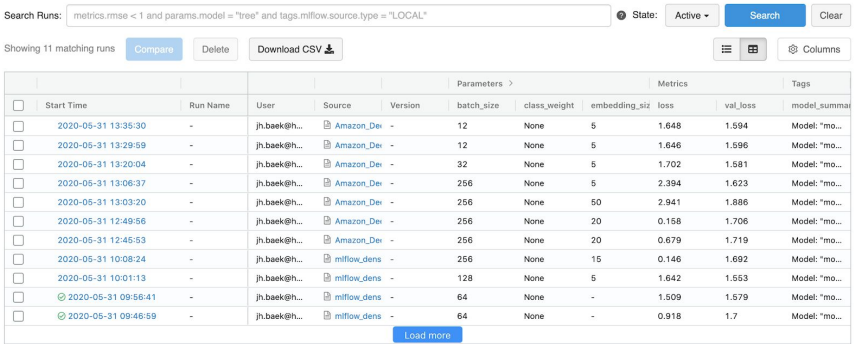
Data Storage & Environment

Raw data sets are stored without transformation in the data lake landing zone. A data source crawler is triggered on object *put events*, which is the event of new data being placed in the s3 bucket. It determines the schema of the source data and maintains it in the data catalog. The standardization zone is where all the processed data sets are located, and the analytics sandbox is where model feature sets are stored and used for analysis. Data sets are partitioned by year-month-date to allow for batch processing. Raw or processed data sets can be accessed programmatically using s3 signed url or Athena/flat file download. The final processed data set is stored in Neo4J graph db for analysis and real time recommendations.

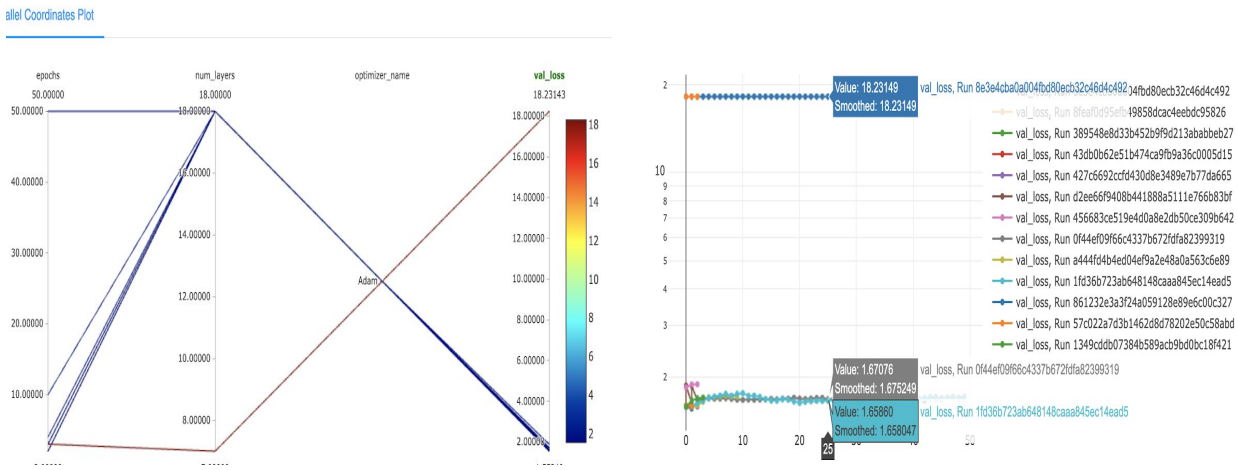


Measurement

Model performances were measured in databricks where multiple models were rated and compared. Similar testing was also done in a few Amazon Sagemaker instances.

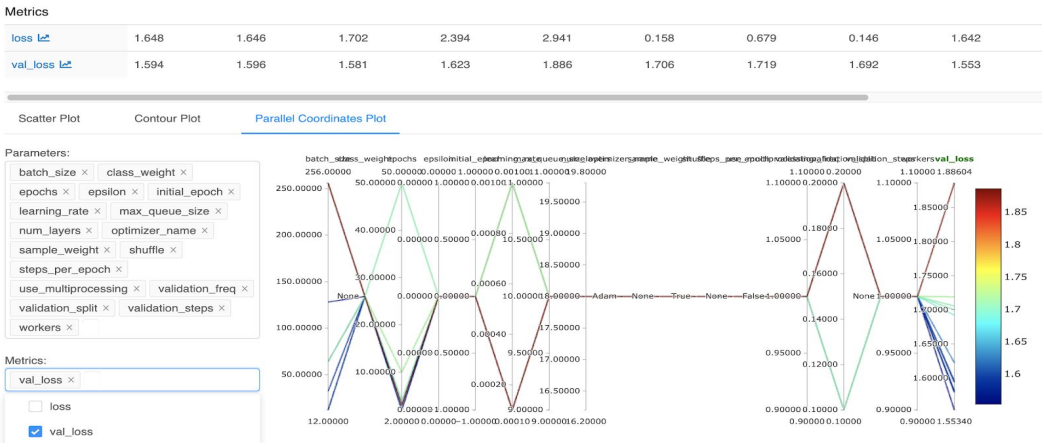


Evaluation



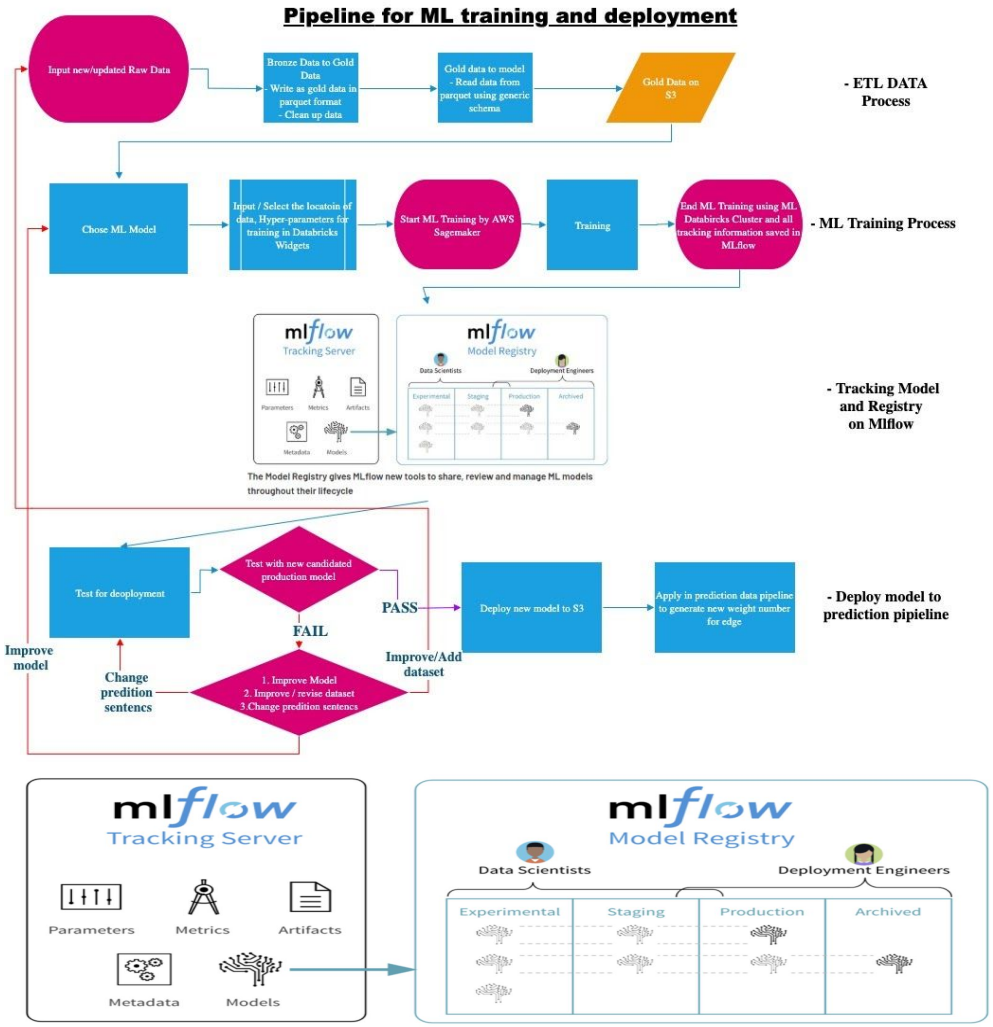
Model evaluation aims to estimate the generalized accuracy of a model on future (unseen/out-of-sample) data. Methods for evaluating a model's performance are divided into 2 categories: *holdout* and *cross-validation*. Both methods use a test set (i.e data not seen by the model) to evaluate model performance.

Optimization



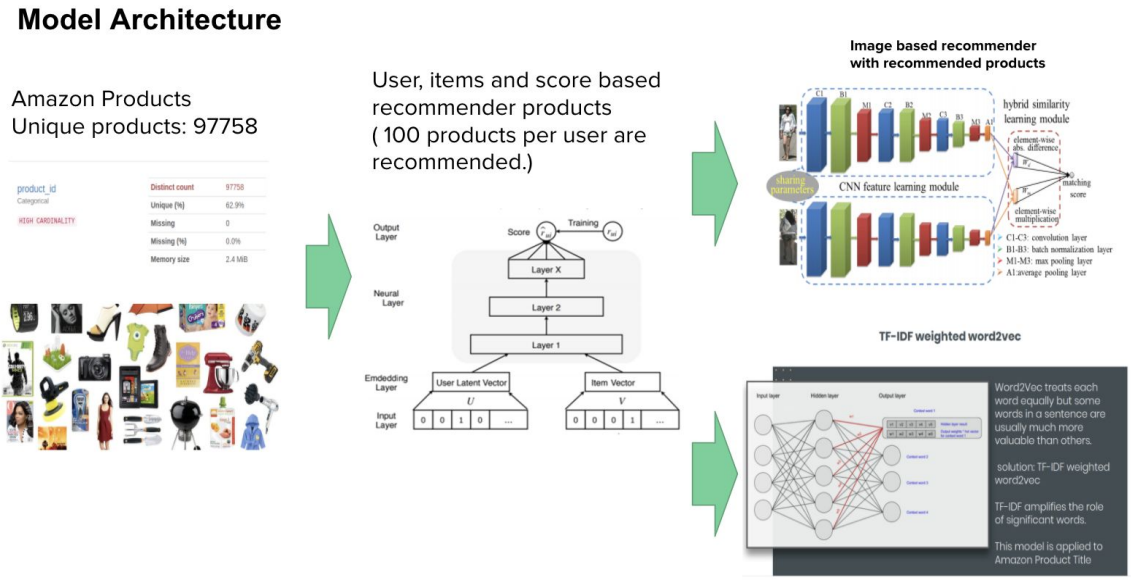
Data Optimization is a process that prepares the logical schema from the data view schema. It is the counterpart of data de-optimization. Data optimization is an important aspect in database management and in data warehouse management. Data optimization is most commonly known to be a non-specific technique used by several applications in fetching data from a data source so that the data could be used in data viewing tools and applications such as those used in statistical reporting.

Workflow and Deployment



A Workflow is a sequence of tasks that processes a set of data. Workflows occur across every kind of business and industry. Anytime data is passed between humans and/or systems, a workflow is created. Workflows are the paths that describe how something goes from being undone to done, or raw to processed. An MLflow workflow is a format for packaging data science code in a reusable and reproducible way, based primarily on conventions. In addition, the Projects component includes an API and command-line tools for running projects, making it possible to chain together projects into workflows.

Analysis Methods



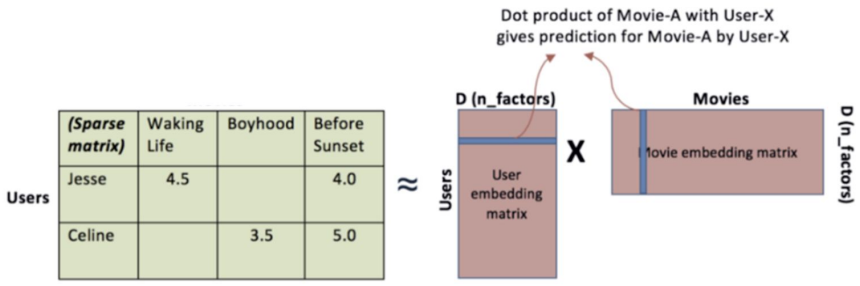
Explicit Feedback Model

Created and compared 2 explicit recommendation engines for predicting user's ratings based on 2 machine learning architecture:

Matrix Factorization: Perform a dot product between the respective user and item embeddings.

Deep neural network: Merge user and item embeddings by concatenation or multiplication, and then use them as features for the neural network.

1. Matrix factorization approach



Matrix Factorisation works on the principle that we can learn the user and the item embeddings, and then predict the rating for each user-item by performing a dot (or scalar) product between the respective user and item embedding.

2. Deep Recommender

Instead of taking a dot product of the user and the item embedding, concatenate or multiply them and use them as features for a neural network.

Thus, we are not constrained to the dot product way of combining the embeddings, and can learn complex non-linear relationships.

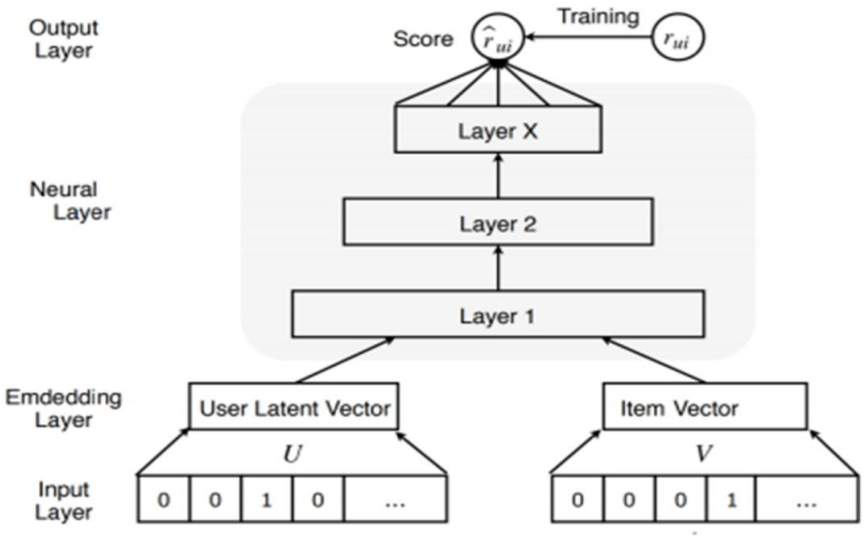


Image-based Model

Feature extraction

This model is based on visual similarity. The model that we used is a pre-trained Deep Learning Convolutional Neural Network. Specifically, we used the VGG16 architecture with 5 convolutional layers followed by 3 fully-connected layers (Fig 1). VGG16 is a convolutional neural network model proposed by K. Simonyan and A. Zisserman from the University of Oxford in the paper “Very Deep Convolutional Networks for Large-Scale Image Recognition”. The model achieves 92.7% top-5 test accuracy in ImageNet, which is a dataset of over 14 million images belonging to 1000 classes (ul Hasan). This implementation was extracted from Keras (Python) using a TensorFlow backend. Each image was resized to 224 x 224 x 3 pixels so it could be placed thru the VGG16 architecture. If we take the whole model, we will get an output containing probabilities to belong to certain classes which is the softmax layer. Although we want to retrieve all the information that the model was able to get in the images. In order to do so, we removed the last layers of the CNN which are only used for class predictions and we extracted 4,096 features from the last fully connected layer (before the Softmax layer).

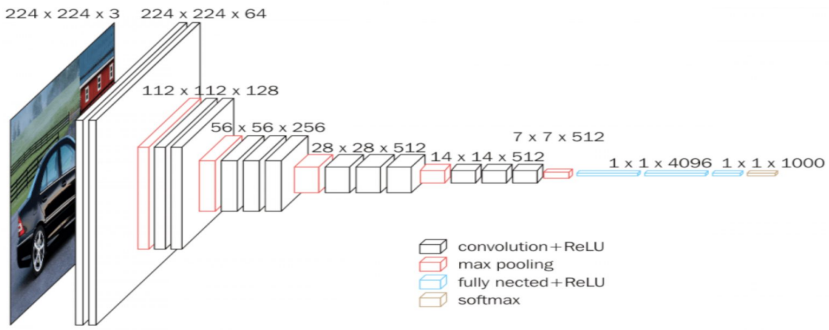


Fig 1: Pre-Trained VGG16 Architecture with ImageNet

PCA

In order to have a better understanding of the features we extracted, we visualized some samples of the shoes in 2 dimensions. As we can see from the bottom figure, sunglasses drift gradually toward bags and slippers and sandals drift smoothly towards sports shoes.

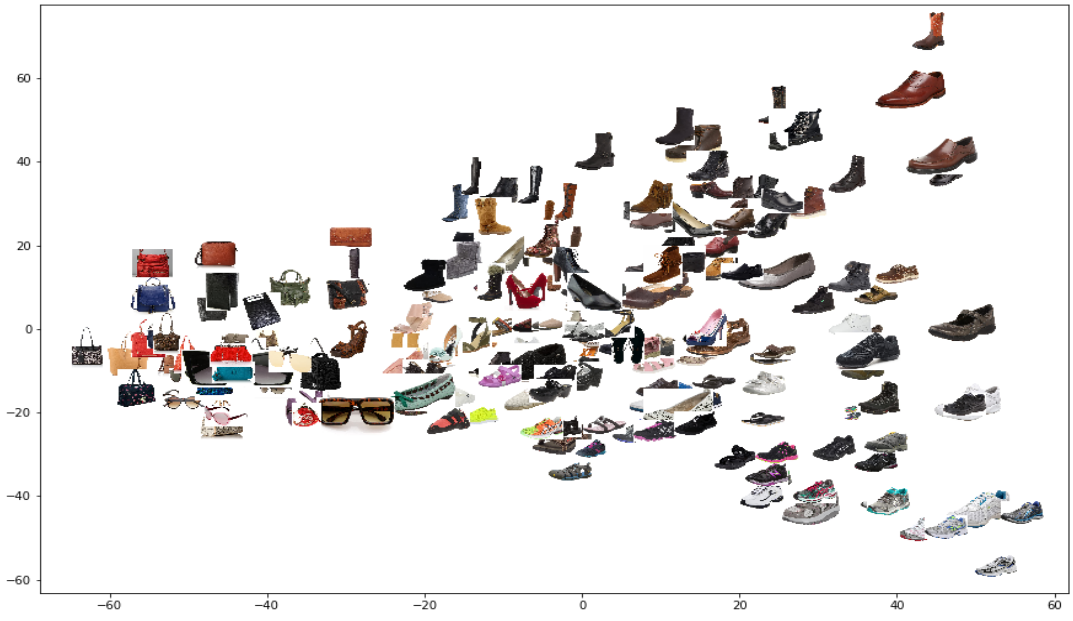


Fig 2: two-dimensional embedding of small sample of products dataset

Clustering

We used a K-Means algorithm to determine the number of possible clusters in our data set. We analyzed the inertia of the model up to 50 clusters. The results are shown in Figure 3 below. Although it is challenging to determine the location where the elbow occurs, we settled

on 15 clusters. The products from each cluster are highlighted in the two dimensional projection plot on the bottom of Figure 4.

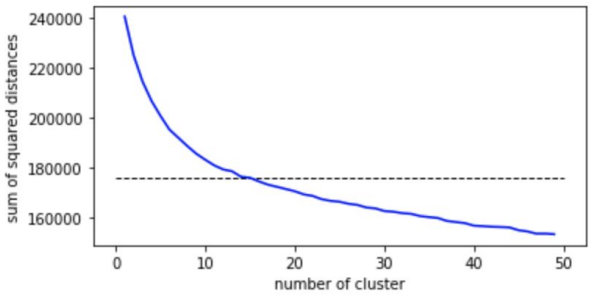


Fig 3: Inertia of the k-means algorithm up to 50 clusters.

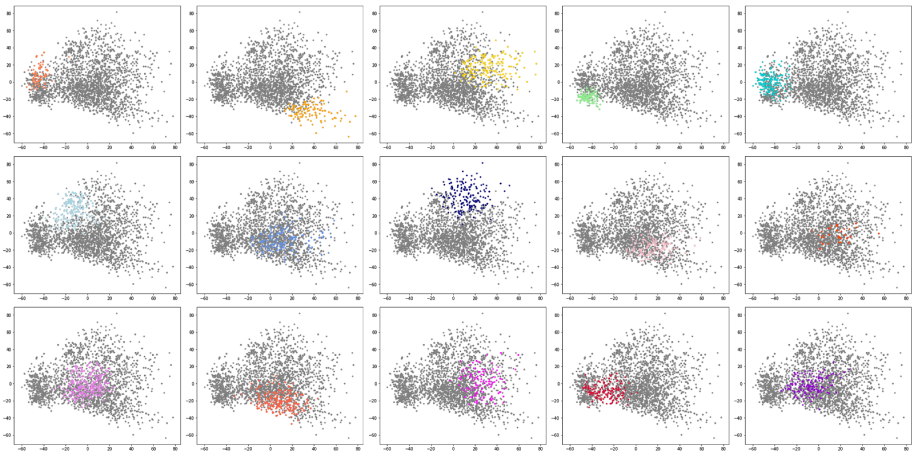


Fig 4: two-dimensional projection highlighting the products that belong to each of the 15 clusters.

Cosine similarity

Cosine similarity is a measure of similarity between two non-zero vectors of an inner product space (wikipedia). Mathematically, it measures the cosine of the angle between two vectors projected in a multi-dimensional space. The cosine similarity is advantageous because even if the two similar products are far apart by the Euclidean distance but they could still have a smaller angle between them. Smaller the angle, higher the similarity.

$$Cos\theta = \frac{\vec{a} \cdot \vec{b}}{\|\vec{a}\| \ \|\vec{b}\|} = \frac{\sum_1^n a_i b_i}{\sqrt{\sum_1^n a_i^2} \sqrt{\sum_1^n b_i^2}}$$

where, $\vec{a} \cdot \vec{b} = \sum_1^n a_i b_i = a_1 b_1 + a_2 b_2 + \cdots + a_n b_n$ is the dot product of the two vectors.

For the given products, because of some products sold by different sellers, there's some products that have different productIds, but the actual images look similar that have the same similarity score. So, when we generating recommendations., we removed the duplicated products which have exactly the same similarity score, and filtered out the similarity score less

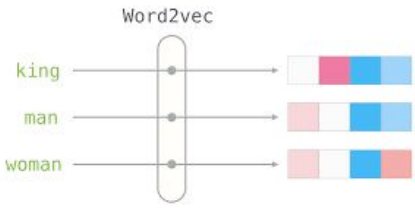
than 0.95, and finally recommended top 5 items for a given product. The picture below shows some samples of the results from this model.

NLP Model

This model is based on text features, particularly product title text. Product titles contain more information on a smaller scale compared to other text features such as product description and reviews. For this model, we used TF-IDF weighted word2vec.
John Rupert Firth famously said, “You shall know a word by the company it keeps.” This is the main principle behind word2vec.

Word2vec is a shallow, two-layer neural net that learns word embeddings. It turns words into vectors by looking for the likelihood that words will co-occur. Word2vec groups the vectors similar words together in vector space. These vectors are distributed numerical representations of word features. Words that we know to be synonyms tend to have similar vectors, and words that are antonyms have dissimilar vectors.

The output is a vocabulary where each word has a vector attached to it.



Source: http://jalammar.github.io/images/word2vec/word2vec.png

Word2vec is neither supervised or unsupervised learning, but rather a *self-supervised* technique. It uses a neural network which back-propagates error. It reads the text and generates labeled data from it.

Word2vec treats each word equally, but some words in a sentence are usually much more valuable than others. To solve this, TF-IDF is used as a weighting statistic for word2vec.

TF, or Term Frequency, measures how frequently a word appears in a document. In contrast, IDF, or Inverse Document Frequency, measures the importance of the words based on how frequently they appear across multiple documents. The value of TF-IDF increases proportionally with the number of times a word appears in a document, or in this case, a product title. It decreases proportionally with the total number of documents in the entire corpus that contains that word. The value reflects how important a word is to a document in a corpus.

Used as a weighting factor, it downplays words that appear frequently across the entire vocabulary and gives more emphasis on words that appear frequently only in a specific product title. This enables a better representation of semantics.

We use Euclidean distance to measure similarities between vectors. The smaller the distance value, the more similar two vectors are. The Euclidean distance of two vectors x = [x1, x2, …, xn] and y = [y1, y2, …, yn] is the 2-norm of their difference x - y. We can compute Euclidean distance between x and y by:

$$d(x, y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

It is important to note that there is no generic way to assess token-vector quality without having to create a ground truth set of words as a benchmark, which would be very labor intensive. Hence, only distance between vectors is used to discern similarity.

Euclidean distance was chosen instead of

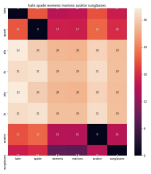
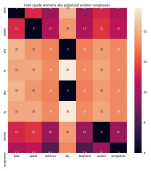
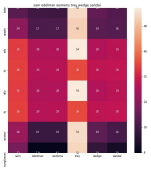
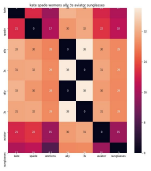
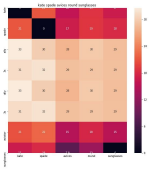
To visualize the model's decision, we plot a heatmap where each cell represents the Euclidean distance between two words. We set the x-axis labels as the featured apparel title and the y-axis as the recommended apparel title.The items are sorted based on distance where the ones with the smaller value have more similarity to the featured product. The top 5 recommended products are the ones that will be featured on the dashboard.

Featured Product

Product Title (pre-processed)	Product Image
kate spade ally 3s ally 3s aviator sunglasses	

Recommended Products

Product Title (pre-processed)	Product Image	Similarity Correlation Matrix

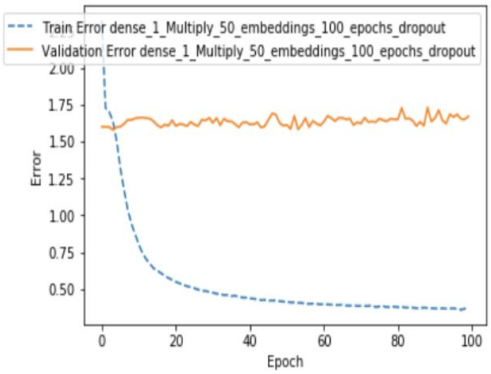
kate spade womens marions aviator sunglasses		
Kate spade womens ally polarized aviator sunglasses		
Sam edelman womens trey wedge sandal		
kate spade womens ally 3s aviator sunglasses		
kate spade avices round sunglasses		

Our findings

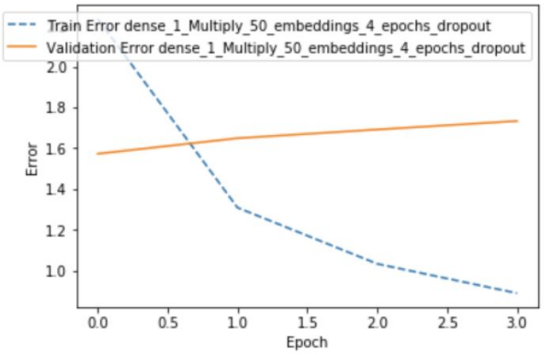
The model we built had the intention to recommend products that best fit the customers' interest. Recommending the right product not only sparks customers' interest and loyalty to the website, but also generates revenue as they are more likely to purchase items that they give high ratings to. In our case, we have experimented numerous different models on multiple combinations of input variables. This brought out another important finding: scalability and workflow pipeline design was just as important as the model itself. Our scalable pipeline enabled us to run experiments efficiently.

While the datasets were very large with extensive attributes, we came to a realization that our recommendation model performs better when it's built in a custom way such that a filtered dataset was used for training. Through our experiments we discovered that the data was

‘noisy’ in the sense that there’s quite a handful of data that was considered ‘meaningless’ from a model perspective, providing no real insights about users’ behaviors. We also found that making predictions within a given product category improves the accuracy of the model (i.e one model per product category). Another benefit of using the more selective dataset was, to our surprise, less overfitting effect. The main take-away insight here is that data engineering is more important than tuning hyper-parameters and model building. At the end, the deep neural network model built using only 5-core dataset within a given category had the best performance out of all models we had experimented.



Error rate using all data



Error rate using selected data

Evaluation

Two common approaches to evaluating recommender systems are:

1. Offline evaluation in the academic world
2. Online evaluation in the business world.

Due to limitations, we first went with the first approach. We have computed prediction errors (such as RMSE & MAE). While RMSE provided valuable insights, it is difficult to tie the result to the products we are recommending. In order to visualize the outcome of our model in a more interpretable way, we developed a Tableau dashboard that dynamically visualizes the recommended products that our model proposed. This dashboard provided a view of how model output would appear in the eyes of end-customers, making the story-telling process more relatable when we propose our models to stakeholders. We also utilize this view to differentiate cases where the model was making reasonable versus unexpected predictions. Since all data and models were executed via amazon cloud space, this pipeline would allow us to visualize how the model output would look to our viewers whenever we make any tuning or adjustment in real time.

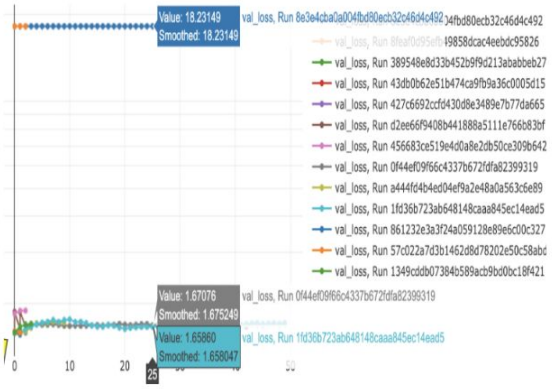


Figure 1: Value Loss Chart of Model

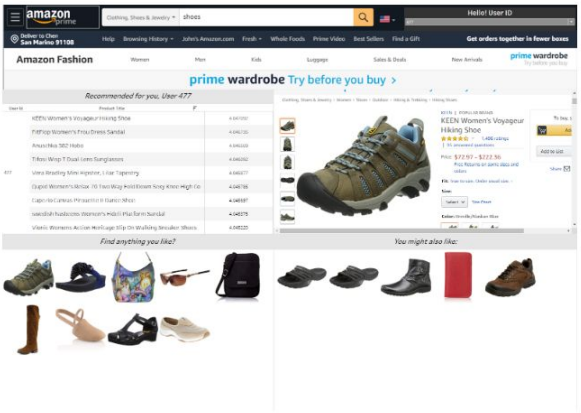


Figure 2: Tableau Dashboard

The other approach is Online Evaluation in the business world. This type of evaluation looks for high Customer Lifetime Values (CLV), going through A/B testing, ROI and QA. These metrics are real-high measurements of the success of a recommendation system. Unfortunately, our group does not have the data nor the capability to acquire these metrics to be used for this project, so we have to settle with offline evaluation.

We can notice the following points from the above:

- Performance got way better when using neural networks compared to using matrix factorization.
- When using a neural network, converging to the best model very quickly, sometimes after 2 epochs and after that the model starts overfitting or at least the validation error does not seem to go down anymore. Matrix factorization does not converge at all.
- Adding epochs lead to overfitting
- Adding layers (over 3) does not help much and actually leads to overfitting
- Changing the number of hidden units does not help.
- Simplifying the model by reducing embedding size does not help either.
- Choosing large values of embedding has made a small improvement in the results.
- Multiply or concatenate user and item embeddings does not seem to matter, but concatenate seems to give little better results
- Training with Dropout seem to prevent some overfitting
- Adding dense layers on top of the embeddings before the merge helps a bit.
- Adding some metadata leads to some improvement in the results.
- Running on a larger dataset does not help either, because the data in both datasets is very skewed.

Conclusion

The main goal of this project is to create an apparel-specific recommender system for Amazon users using machine learning techniques. We ended up with a multi-step recommender system that first recommends items based on explicit feedback. This model uses deep learning that aims to predict a user's rating on products and suggests the one that will likely have a high rating. From the output of that model, our system then looks for product similarities based on two different approaches: image-based processing and Natural Language Processing. These two methods look at the products' features to determine the top 5 most similar items to the featured item.

From creating this system we have discovered that when it comes to optimization, data engineering on selective data is more effective than tuning parameters on various models, which often takes a lot more computing power and resources because of the scale of this dataset. We also discovered that with deep learning models, appropriate drop-outs implemented had the best performance in terms of accuracy. Due to the size of the dataset, the use of AWS was essential to the project. From this we learned that the usage of cloud-computing tools such as Amazon SageMaker and Databricks enable quick workflow to provide effective model building and testing.

Another goal of the project was to build a customer-facing data product which will provide recommendations to a given user. For this, Tableau provided us with an easily buildable and deployable dashboard that is intuitive for customer use.

We would like to note that the product is not ready for deployment as some recommendations were not ideal after reviewing the output in the dashboard. For example, some items recommended based on image similarity were from different product categories. Also, to deploy this product, we need to create a better design for scalability as much as possible to enable iteration and proficiency.

One of the main challenges that we faced was how to measure the performance of our product. There are two approaches to evaluating recommender systems: *offline evaluation in the academic world* (i.e. MAE, Recall) and *Online evaluation in the business world* (i.e. A/B testing, ROI). Even though we would like to measure the success of our recommender system through the means of *online evaluation*, our group does not have the data nor the capability and resources to acquire these metrics for this project. For future work, we recommend online evaluations as part of creating a recommender system product in order to evaluate its real-world performance.

Reference

MacKenzie, Ia. “How Retailers Can Keep up with Consumers.” *McKinsey & Company*, www.mckinsey.com/industries/retail/our-insights/how-retailers-can-keep-up-with-consumers.

Danziger, Pamela N. “Amazon, Already The Nation's Top Fashion Retailer, Is Positioned To Grab Even More Market Share.” *Forbes*, Forbes Magazine, 29 Jan. 2020, www.forbes.com/sites/pamdanziger/2020/01/28/amazon-is-readying-major-disruption-for-the-fashion-industry/.

Linden, Greg, et al. “Amazon.com Recommendations: Item-to-Item Collaborative Filtering.” *A Congestion Control Framework for BISDN Using Distributed Source Control - IEEE Conference Publication*, 23 Jan. 2003, ieeexplore.ieee.org/document/116734.

Kumar, Illa, et al. “Content based Apparel Recommendation System for Fashion Industry.” *International Journal of Engineering and Advanced Technology (IJEAT),* 6 Aug. 2019, https://www.ijeat.org/wp-content/uploads/papers/v8i6/F7880088619.pdf.

Zhang, Shuai, et al. “Deep Learning Based Recommender System: A Survey and New Perspectives.” *ArXiv.org*, 10 July 2019, arxiv.org/abs/1707.07435.

Xue, Hong-Jian, et al. “[PDF] Deep Matrix Factorization Models for Recommender Systems: Semantic Scholar.” *Undefined*, 1 Jan. 1970, www.semanticscholar.org/paper/Deep-Matrix-Factorization-Models-for-Recommender-Xue-Dai/49bef668aff3cc3d470339479dd3cee0b4c9cf4f.

Huang, Zan, et al. “A Graph-Based Recommender System for Digital Library.” *University of Arizona*, 1 Jan. 2002, arizona.pure.elsevier.com/en/publications/a-graph-based-recommender-system-for-digital-library.

Wills, Jennifer. “7 Ways Amazon Uses Big Data to Stalk You.” *Investopedia*, Investopedia, 22 Apr. 2020, www.investopedia.com/articles/insights/090716/7-ways-amazon-uses-big-data-stalk-you-amzn.asp.

McAuley, Julian. “Web Mining and Recommender Systems.” *CSE 258*, cseweb.ucsd.edu/classes/fa19/cse258-a/.

“Basic Data Processing and Visualization.” *Coursera*, www.coursera.org/learn/basic-data-processing-visualization-python.

ul Hasan, Muneeb. “VGG16 - Convolutional Network for Classification and Detection.” *VGG16 - Convolutional Network for Classification and Detection*, 21 Nov. 2018, neurohive.io/en/popular-networks/vgg16/.

“Cosine Similarity.” *Wikipedia*, Wikimedia Foundation, 17 May 2020, en.wikipedia.org/wiki/Cosine_similarity