Project Github link

Recommendation system for fashion products (H&M)

Zhe Wang (UNI: zw2888) Ayushi Sharma (UNI: as6608) Pooja Srinivasan (UNI: ps3312)
Hrishikesh Arvind Talasila (UNI: ht2566) Tom Liu (UNI: tl3201)

Problem statement

Build a recommendation system that gives customers product suggestions for purchase over the next seven days. This is based on previous purchases made by them and the (disclosable) details collected and shared by the company (H&M).

Introduction

This project's goal was to create a service for product recommendation to customers. This is based on their prior purchases as well as the (disclosable) information that the business has gathered and disseminated (H&M). In this particular field, creating anything akin to a recommendation system is advantageous and has significant effects on customer happiness and product growth, both of which are crucial in the market. The E-commerce domain is an ever growing industry. We chose this problem statement because of the abundance and organization of data that enabled us to concentrate on feature engineering, modeling, and experimenting with various ML methods. Additionally, the learnings are easily transferable to any domain.

With about **1,250,000** client entries and **1,362,281** unique transactions, there are around **45,000** unique products. The dataset is broken up into files and includes information about the products, the customers, location estimates for the customers, and product photos, among other things. The data's combination of numerical, category, and visual data enables us to study and interact with a variety of data types.

Pre-processing

EDA: We used only a few important features from each dataset which we thought were important for the task of recommendation. Data diagram at 2.
Some important pre-processing that was performed is as follows:
Null value handling, junk data cleaning, correlation based cleaning and categorical variable handling. More info at 3.

Insights from data exploration

Articles.csv 4

1. In articles, there are 0.1 million unique articles. Ladieswear unquestionably contains the most indexing among the 10 categories of items in the csv.
2. Menswear makes up the smallest percentage of all the categories of clothing. Sportswear makes up the smallest percentage, which is understandable given that H&M is a clothing company. The majority of Jersey Fancy's sales in the garment category come from ladieswear and infant/child clothing.
3. Menswear is the least portion in all garment groups except shirts. Appearance_solid, color_black, color_blue are the top three most popular features.

Transactions.csv 5

1. The transactions.csv file contains 31.7M transactions and contains transactions from September 2018 to September 2020.

2. Compared to other seasons, summer has more transactions. There are 1.36 million distinct clients and about 0.5 million consumers have one to five transactions, 0.21 million have six to ten, 0.12 million have eleven to fifteen, and the remaining 0.53 million have more than fifteen.

3. The median transaction price is roughly 0.025, with a wide range of prices (MIN=0.001, MAX=0.59, IQR 0.015-0.035). Online sales account for 4% of transactions, while in-store sales account for 29.6%.

Customers.csv 6

1. Customers has the following 5 columns: ['customer id', FN, Active, Club Member Status, Fashion News Frequency, Age, and Postal Code]. Most clients fall under FN's category 0, which suggests that they do not subscribe to the fashion newsletter.

2. The bulk of clients are classified as having an Active score of 0, which suggests that they are not engaged in conversation. The majority of clients fall under the club member status category Active, indicating that they are part of the H&M customer club.

3. Those in the 20–30 age range do the majority of their shopping.

Sampling

Because of our limited computing resources, we extracted 1% of customers (13,720 entries) from the raw dataset. Then, we pulled all transactions (316,443), and articles (51,894) bought related to this 1% of customers.
After the initial sampling, we filtered out customers with only one transaction. The reason is that our methods require at least one transaction in both the testing and training dataset. After removing customers with one transaction, there are 12,306 records of customers and 315,112 records of transactions.
We split training/testing datasets into 80%/20% distribution. For customers with two transactions, our splitting script ensures one entry in the training set and one in the testing set.

Machine Learning techniques used

We explore multiple ways to solve this task of recommendation:

1. **Content Based filtering**
   Content-based filtering uses item features to recommend other items similar to what the user likes, based on their previous actions or explicit feedback. Our approach also used user matrix to calculate the interest score, which in between Content Based Filtering and Collaborative Filtering. Following is the process:
   a) Take important features from the transaction dataset and one hot encode them. Users were selected who had at least 2 items and a dataframe was built for the transactions.
   b) Create and normalize a feature_score matrix which contains sum of # of transaction for each feature grouped by each customer.
   c) Create item_feature matrix, which is basically one-hot encoded for all articles with their corresponding features.
   d) Calculate customer interest score by doing dot multiplication with item_feature matrix and feature_score matrix and get top 20 recommendations.
   e) Evaluate on the test dataset.

   In the evaluation process, products that are being recommended are tested, and a count is created for each consumer to keep track of the actual products that were purchased based on the forecasts. It is a good outcome if the model suggests the same article. The score for accuracy is 0.00991. There are too many articles that are quite similar, which is why accuracy is so low. This accuracy could be enhanced with smoothing, as performed in collaborative filtering.

2. **Collaborative filtering**
   The collaborative filtering method gives customers recommendations based on similar customer purchase histories.
   a) We used singular vector decomposition (SVD) as the collaborative filtering technique. We create a pivot matrix (12,306 * 51,894) where each row represents a customer, and each column represents an article with a value (0 or 1) of if the customer (row) purchased the article (column).

b) The SVD algorithm takes this pivot matrix as the input and decomposes it into three matrixes: left singular vector U, singular values sigma, and right singular vector $V\hat{t}$. U represents the relationship between users and the latent factor, sigma describes the strength of each latent factor, and $V\hat{t}$ indicates the similarity between items and latent factors. We use the first latent factor in sigma and compose U and $V\hat{t}$ to form a recommendation matrix, where each row represents a customer, and each column represents an article. Cells in the recommendation matrix are the weights that the customer buys articles. A higher weight represents the customer is more likely to buy the article and vice versa.

c) To give a customer top n recommendations, it retrieves the row with the customer id from the recommendation matrix and reports top n weights with their article ids (column names). Then, it looks up these article ids from the article's metadata and reports these recommendations to the customer.

d) The accuracy score is 0.00965. We revisit the reason that caused such a low score - too many articles are very similar. For example, if we recommend a customer with item A, but the customer purchased B, given item A and B has identical product name, type, make, etc. With reasoning, it should be adequate to say recommending A or B are both true positive predictions. Therefore, we use a smoothing technique. With the smoothing method, the accuracy score goes up to 0.3304.

3. **Methods based on Neural Networks**
Our corpus consists of images of articles. We use this information to predict purchases of customers based on their previous purchases by finding article-article similarity using images. We tried 2 transfer learning and clustering approaches to find image similarity and use it for recommendations:

Approach 1:

Here we use a pre-trained MobilenetV2 model as a standalone feature extractor to pre-process images and extract relevant features. It was selected based on its applications in this domain and its small size which is useful for fast training. The feature extractor was run for our corpus of over 100,000 images on GPU NVIDIA V100 for nearly 1 hour. Note that there was no dimensionality reduction done as we wanted to avoid any information loss.

Once the features were extracted, we used a clustering algorithm by Spotify to calculate similarity score. **Annoy (Approximate Nearest Neighbor Oh Yeah)**, is an open-sourced library for approximate nearest neighbor implementation. We use it to find the image feature vectors in a given set that are closest to a feature vector. For each image, we generate 20 closest neighbors based on their cosine distance. ANNOY creates an index which is a forest of many trees whose nodes are individual feature vectors. The index creation and subsequent forest traversal to get similar nodes for each node takes a whopping 8 hours on GPU! At the end of this we get similar articles for each article. The similarity results seem pretty good. PFA visual results of similarity solution here. 7

We use this as a tell for future purchases for each customer. Future product purchase is not exactly dependent on product similarity, but is a common indicator as the task of recommendation is not straightforward. We use the article similarity to recommend users more products based on their past purchases. This model gets a decent score of 0.013% recommendation accuracy which is decent in this field. Based on our learnings from this experiment, we tried out other ideas which could improve this further.

Approach 2:

Using a pretrained VGG16 model, we were able to extract features from images, perform Principal Component Analysis(PCA) to reduce dimensions of the dataset and then perform k-means clustering to cluster images and provide recommendations based on which cluster has the item previously purchased by the customer.

To extract features from images, the output layer of the original VGG16 architecture was dropped. Hence the output was now a vector of length 4096 which served as the features for any image that was provided as input to the neural network. This time we tried dimensionality reduction to remove noise and improve performance. Principal component analysis was performed to retain 90% variance in data. This reduced the number of features from 4096 to 1322. Please see 8 for more info.

K-means clustering was performed using a varied number of clusters and the respective silhouette scores were recorded to determine the best number of clusters to be 950. The distribution of number of samples

in each of the 950 clusters is shown below. Please see 9 for more info.

Once clustering was complete the recommendations were made. Mean Average Precision @k for this method was 0.101 which is quite high for NN-based techniques. To put into perspective, the Kaggle competition winners achieved Mean Precision 0.0348 for the entire dataset. Ours is comparable to that given the scale. 10

Conclusion

In conclusion, we tried different suggested methods in the field of recommendation. It was a rich experience and great learning. We were also able to implement many things we learnt in the class.
Overall, Collaborative filtering performed the best. Image similarity with k-means clustering and neural network also looks promising. As an improvement, we can also use the text description of the articles as a feature. We can use NLP techniques in that like attention-based models. We can also operate product category-wise and have an ensemble model on top of each category model. This way we can leverage common patterns and features of products in a category to improve accuracy.

APPENDIX

1. Types of files in the given dataset:

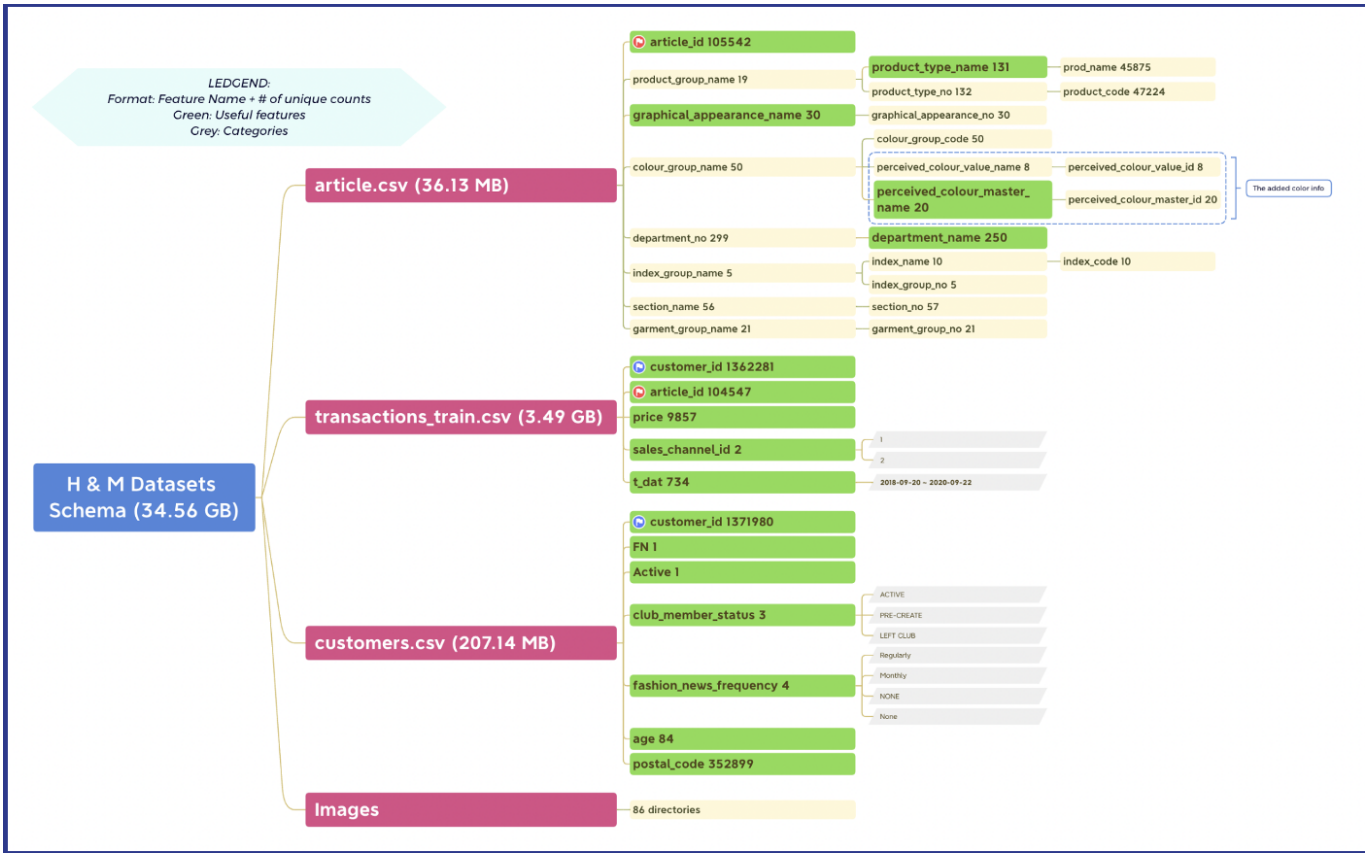
   a) *Images:* This is a folder containing images for articles corresponding to the last three digits of article_id.

   b) *articles.csv:* This file has data about the articles with 25 possible features.

   c) *customers.csv:* This file has data about the articles with seven possible features like id, age, membership, zip code, etc.

   d) *transactions_train.csv:* This file has the training data for the final model that includes purchase details for each customer with data, price of purchase, and additional information.

2. Subset of features selected:

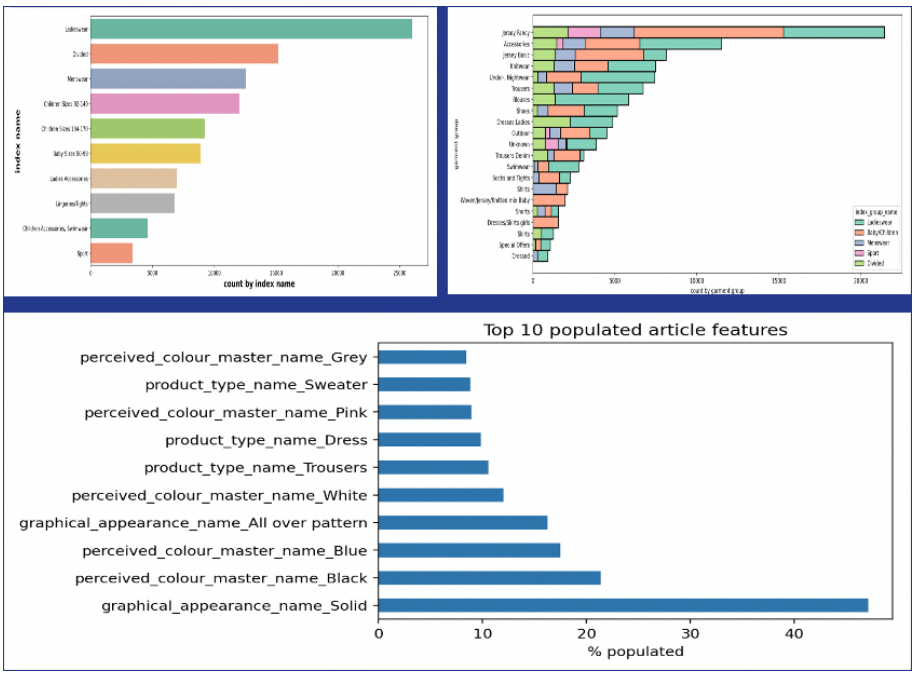


3. Cleaning and sampling:

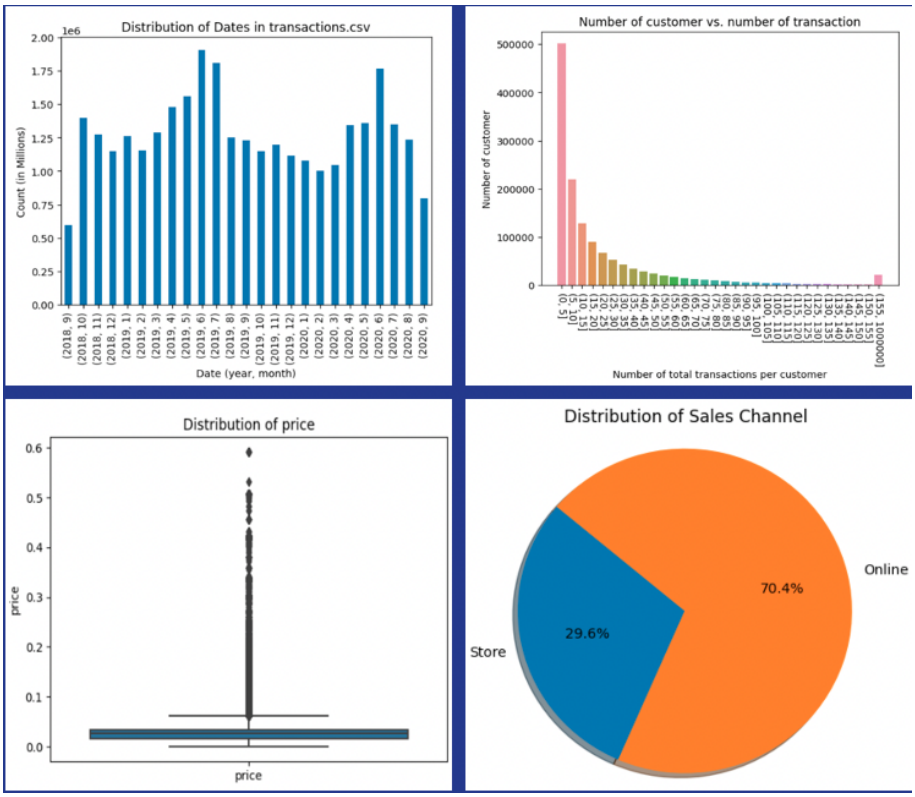
DATA CLEANING

Cleaning and sampling steps	Articles.csv	Transactions	Customers.csv
Null handling	No null data in Articles.csv	No null data in Transactions.csv	All columns having null values were imputed with appropriate values based on the categories of that column.
Junk data handling	For article description, stop words removal using NLTK, cleansing.	No junk data found	Members with None, none, nothing or <empty> club member status were categorised as “NEVER A MEMBER”.
Correlated features	Based on similarity using correlation matrix, we selected a subset of features.	No high correlation between features	FN and fashion news frequency features provide similar information so we dropped fashion news frequency.
Categorical variable handling	One-hot Encoding	Convert date to dtype:datetime64	Since the features have few categories we have decided to to one hot encoding for them

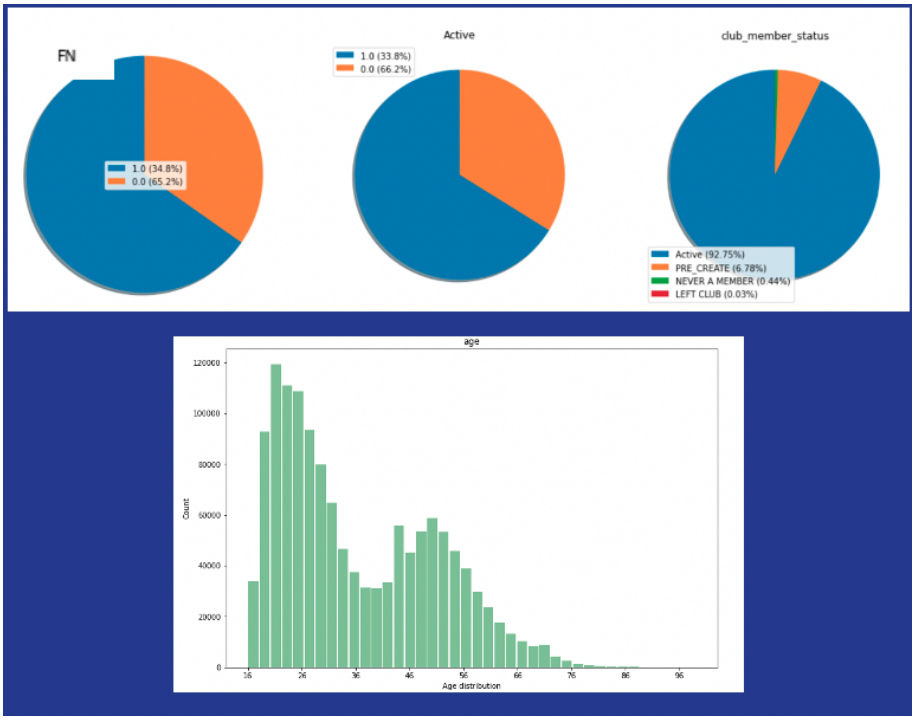
4. Articles EDA:



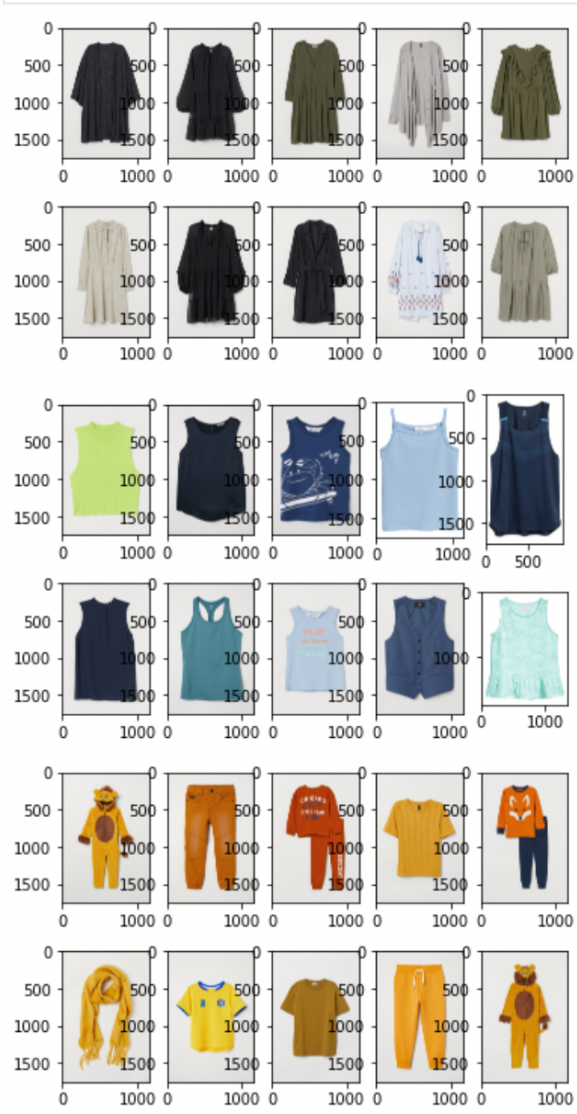
5. Transactions EDA:



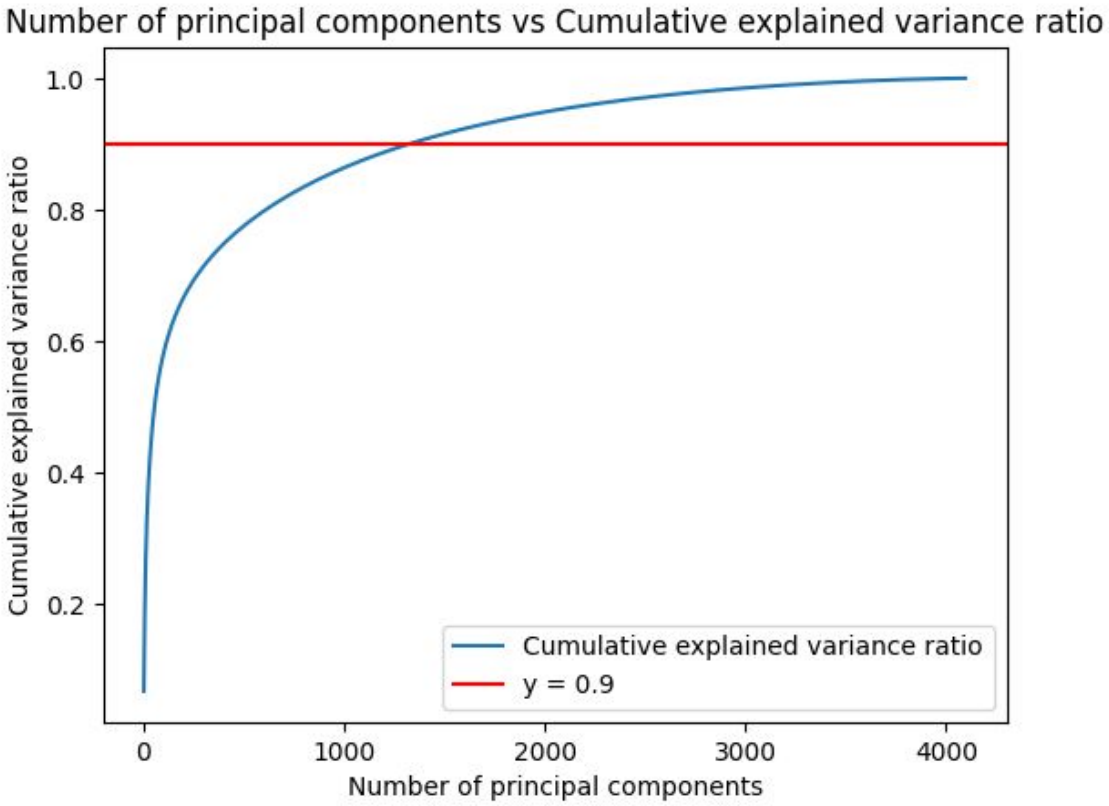
6. Customer EDA:



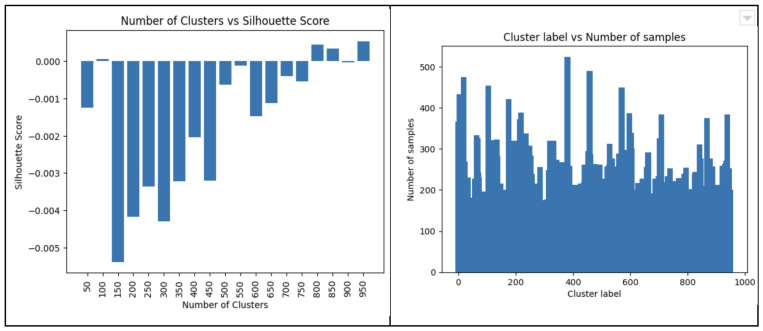
7. Results from Approach 1 of neural networks:



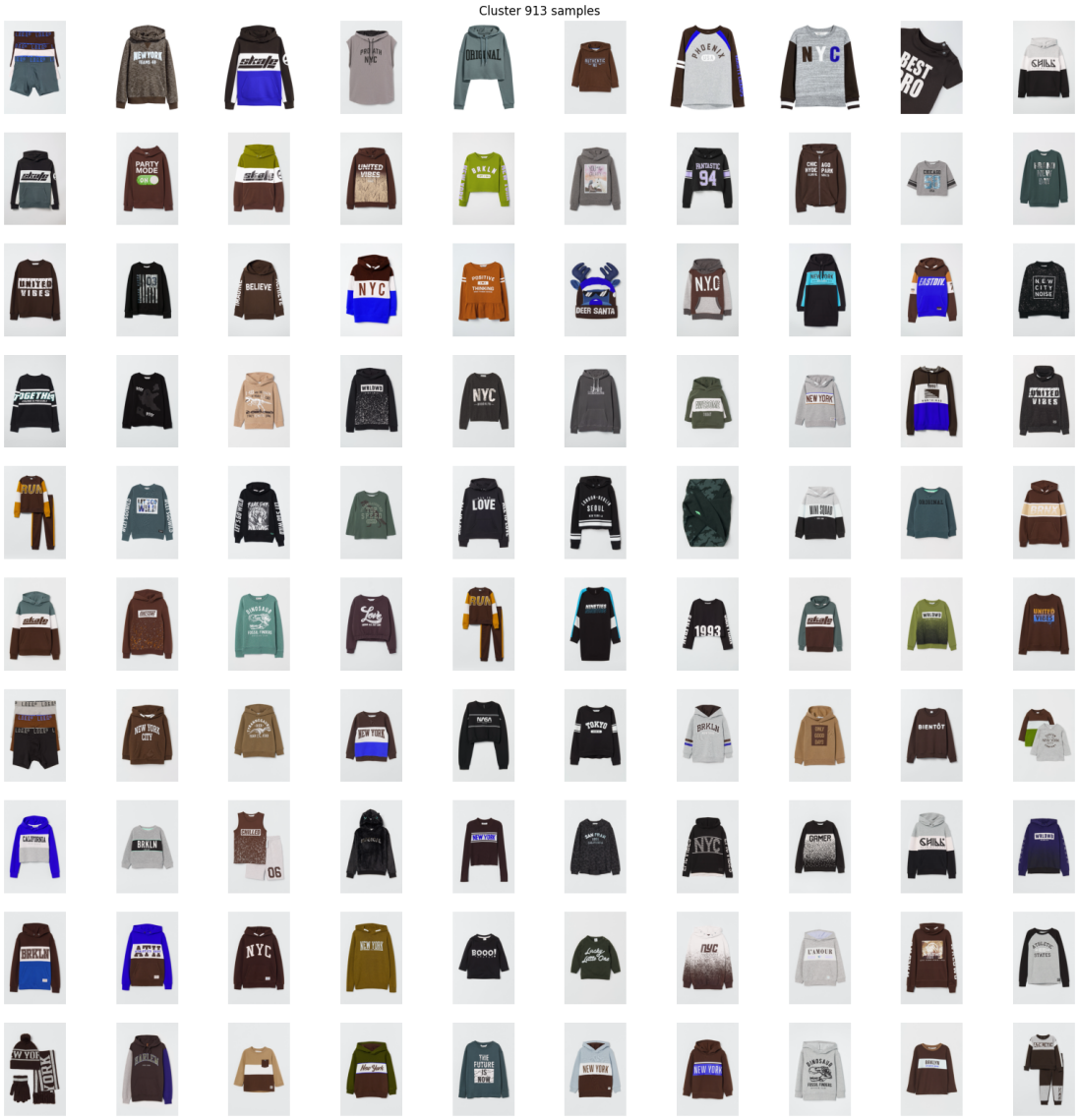
8. From Approach 2 in neural networks: Reduced no. of features after PCA:

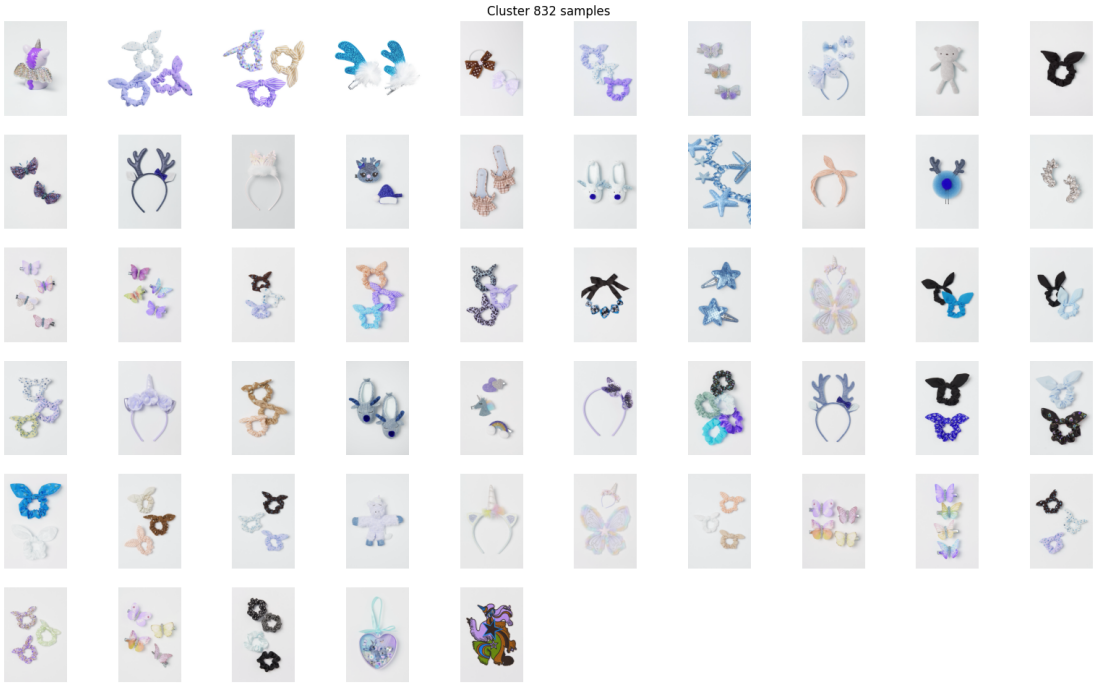


9. From Approach 2 in neural networks: The distribution of number of samples in each of the 950 clusters:



10. Results from Approach 2 of neural networks:


Cluster 913 samples



11. References

- He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep residual learning for image recognition." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770-778. 2016.
- S. Liu, J. Feng, Z. Song, T. Zhang, H. Lu, C. Xu, and S. Yan. "Hi, magic closet, tell me what to wear!". In ACM Multimedia, 2012.
- https://lvngd.com/blog/image-similarity-nearest-neighbors/
- https://medium.com/web-mining-is688-spring-2021/e-commerce-recommendation-engine-with-collaborative-filtering-cb19cd542c18
- https://towardsdatascience.com/a-content-based-recommender-for-e-commerce-web-store-7554b5b73eac