Analysis of Summer Clothes in E-commerce Wish

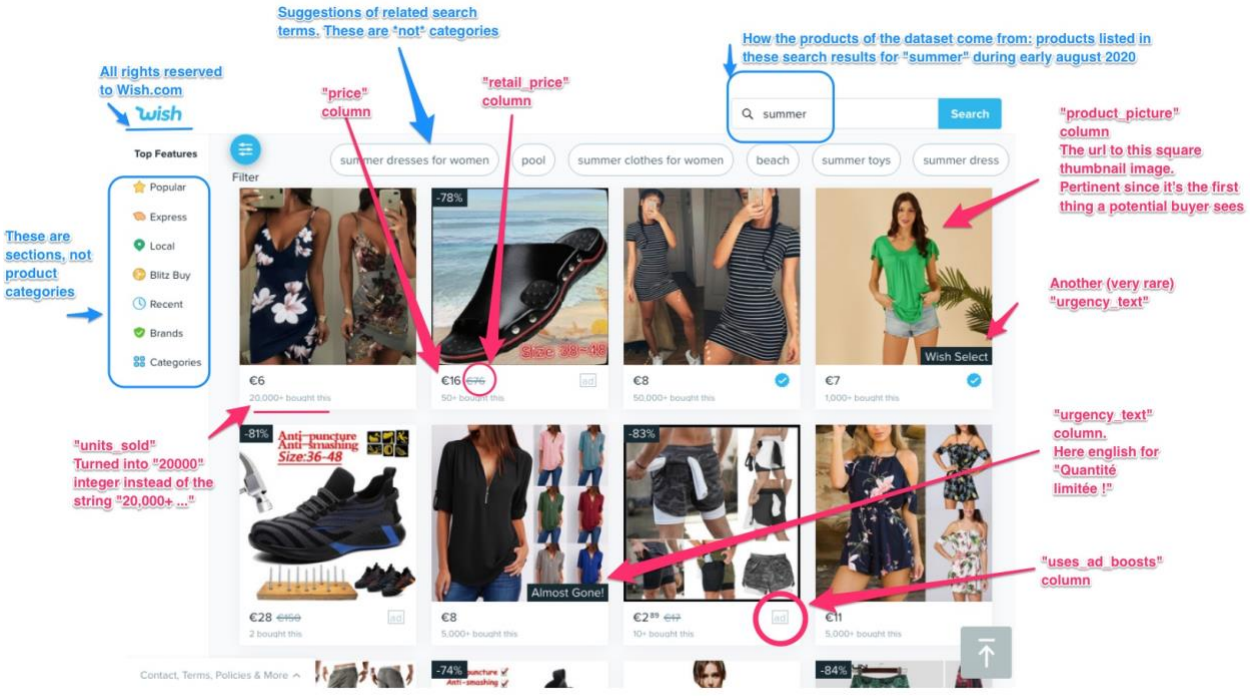
Yibo Feng

Overview

The dataset collects top products information including ratings and sale performance from E-commerce platform “Wish”. At the beginning of data mining, we start to work on product segmentation. The goal is to help the platform improve their customer base, work on target areas, and segment customers based on purchase history and interests. Besides, product segmentation will help the platform find out the common points and different types of product, then they will be able to set target sale policy. We are also interested in what factors contribute to high rates of rating with five count which means the percentage of rating 5 in total rating count is higher than 50%. This will contribute to improving customer preference. We also want to analyze potential factors that help increase the sale of products. For example, it is interesting whether the merchant rating will influence the sale of products. By analyzing these factors, the platform will be able to predict sales status of each product.

Data Summary

We downloaded the data from Kaggle, https://www.kaggle.com/jmmvutu/summer-products-and-sales-in-ecommerce-wish. The following picture from the author perfectly interpreted features captured from the platform.



The dataset includes 1573 observations and 43 attributes. Some variables such as merchant_name and merchant_rating show the feature of merchant. Varaibles like uses_ad_boosts and badges_count demonstrate the detail and characteristics of products.

Data Preprocessing

Before the data analysis tasks, we found most of variables in the data set were irrelevant. For example, ‘title’ wouldn’t contribute to our future analysis nor made sense in the analysis. Therefore, we decided to remove some variables. Then, we used summary() and found that some variables had missing values. We used means to fill the missing values of numeric variables and delete missing values of category variables.

Then, we created a new column named rating_level, which was the percentage of rating with 5. We calculated it with the number of rating with 5 and the total number of all ratings. For those observations with 0 rating with 5, we used 0 to substitute. Then, if the value was below 0.5, we set it as ‘low’, otherwise, it was ‘high’.

Besides, we created a new column named ‘units_sold_level’ using ‘units_sold’. According to the median, we split ‘units_sold’ at 1000, below which was ‘low’ while above was ‘high’.

Finally, before we started experiments, there were 19 variables with 1499 observations.

Analysis

1. Product Segmentation

1.1 Specific Preprocessing

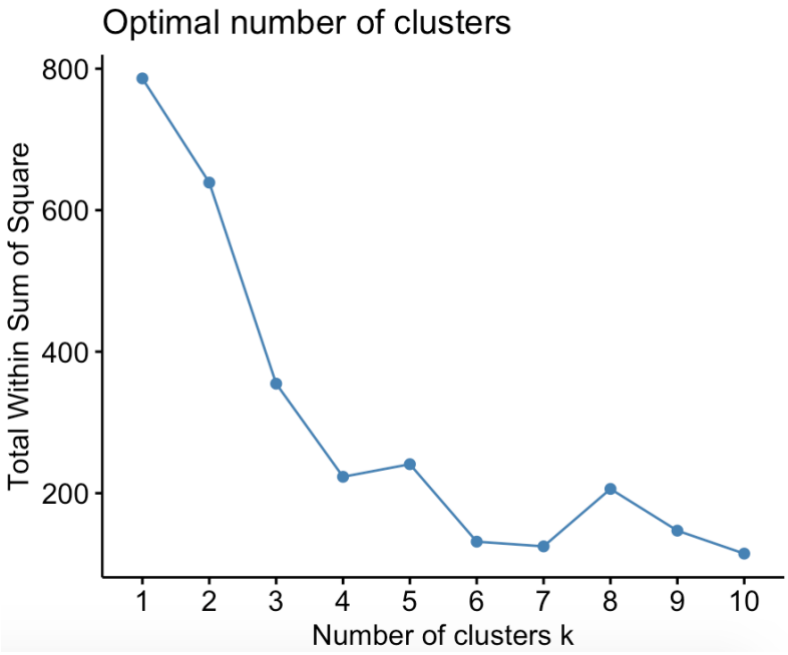
Since we planned to try Kmeans and HAC algorithms to segment products. We need to do additional preprocessing on prepared data. Since we wanted to find different clusters of products, we should remove features irrelevant to products, such as “merchant_name” and “units_sold_level”. Then, we reformatted category variables into numeric ones to meet the requirements of algorithms. The data frame used in the model included 9 variables.

```
'data.frame':   1499 obs. of  9 variables:
 $ price                : num  16 8 8 8 2.72 3.92 7 12 11 5.78 ...
 $ retail_price         : int  14 22 43 8 3 9 6 11 84 22 ...
 $ uses_ad_boosts       : int  0 1 0 1 1 0 0 0 1 0 ...
 $ rating               : num  3.76 3.45 3.57 4.03 3.1 5 3.84 3.76 3.47 3.6 ...
 $ badges_count         : int  0 0 0 0 0 0 0 0 0 0 ...
 $ product_color        : num  89 36 44 8 100 59 89 15 8 7 ...
 $ shipping_option_price: int  4 2 3 2 1 1 2 3 2 2 ...
 $ shipping_is_express  : int  0 0 0 0 0 0 0 0 0 0 ...
 $ origin_country       : num  2 2 2 2 2 2 2 2 2 2 ...
```

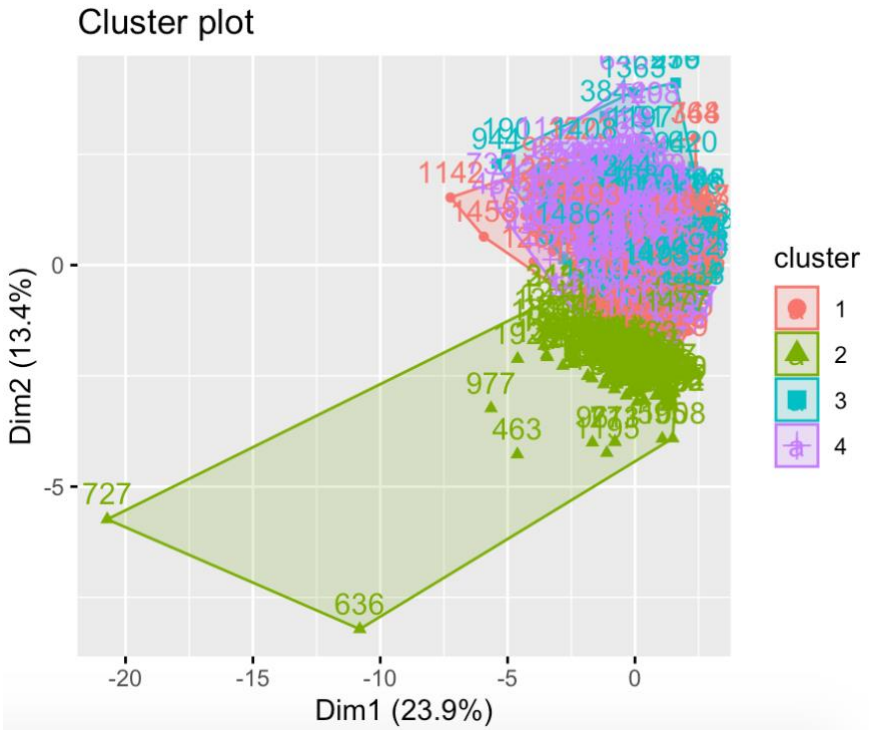
1.2 Clustering Models

(1) K-means

First, we defined a function to normalize numeric variables. Then, we used within-cluster sum of square (WSS) to measure the best k value in the model. According to the elbow diagram, from 1 to 4 clusters, the total WSS decreases rapidly. And as numbers after 4 increase, the variation of WSS was slow and mild. Therefore, the elbow plot displayed the best k value, which was 4.



Then, we inputted 4 as the parameter into Kmeans model and got the cluster plot. As we could see from the plot, the clusters were not clear and there were a lot of outliers. These might come from the variables which were transformed from category variables. Therefore, we tried another clustering model, Hierarchical Cluster.



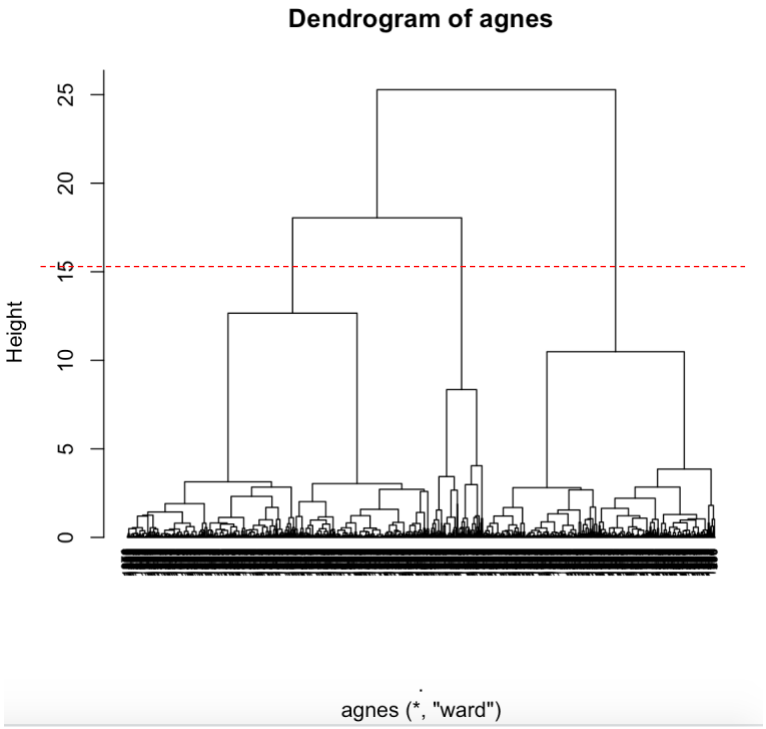
(2) Hierarchical Cluster

We used Euclidean distance to build the distance matrix. Then, we tried the following four methods and got the accuracy of them. According to the accuracy, we decided to use Ward's method to identify clustering.

Method	Accuracy
Average linkage clustering	0.9735894
Weighted linkage clustering	0.9718974
Single linkage clustering	0.9655218
Ward's minimum variance method	0.9976868

1.3 Result Analysis

According to the dendrogram, the higher the height of the fusion, the less similar the observations are. Therefore, we decided to cut trees at six, where the red line was.



According to the result of clustering, we split them into 3 clusters and found some interesting insights.

For the first cluster, the price was under 23. Besides, all products in this cluster didn't use ad boost while the count of badges was 0. The shipping of them was not express.

For the second cluster, all products used ad boost while the count of badges was 0. The shipping of them was not express. Most of them came from China.

For the third cluster, all of products got badges from 1 to 2. Besides, some products in this cluster had express shipping.

2. Factors for High Rating Rates

2.1 Specific Preprocessing

To prepare for the discretization, we removed variables that could not represent the preference of customers. We reformatted category variables as factors and converted numeric variables into category based on their 1st and 3rd quartiles. Therefore, 'price' and 'retail_price' were divided into three types, and 'shipping_option_price' was divided into two types. Finally, there were 12 variables left. We transformed the data frame into transaction to prepare for the model.

```
'data.frame':   1499 obs. of  12 variables:
 $ price               : Factor w/ 3 levels "<6","6-11",">11": 3 2 2 2 1 1 2 3 2 1 ...
 $ retail_price        : Factor w/ 3 levels "<7","7-26",">26": 2 2 3 2 1 2 1 2 3 2 ...
 $ uses_ad_boosts      : Factor w/ 2 levels "0","1": 1 2 1 2 2 1 1 1 2 1 ...
 $ badge_local_product  : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
 $ badge_product_quality: Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
 $ badge_fast_shipping  : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
 $ product_color       : Factor w/ 100 levels "applegreen","apricot",..: 89 36 44 8 100 !
9 89 15 8 7 ...
 $ shipping_option_price: Factor w/ 2 levels "<3",">=3": 2 1 1 1 1 1 1 1 1 1 ...
 $ shipping_is_express  : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
 $ origin_country      : Factor w/ 6 levels "AT","CN","GB",..: 2 2 2 2 2 2 2 2 2 2 ...
 $ merchant_rating     : Factor w/ 901 levels "2.333333333",..: 594 195 335 392 355 19 5(
5 40 94 349 ...
 $ rating_level        : Factor w/ 2 levels "high","low": 2 2 2 1 2 1 2 2 2 2 ...
```

2.2 Association Rule

Since the goal is to find out factors contributed to the high rating rates, we set 'rating_level = high' as the right-hand side of the rules. At first, we set the parameters as 'support = 0.01, confidence = 0.8, minimum length = 2'. Then, we got 604 rules, which was too large to get useful information. Therefore, we fixed confidence as 0.8 and adjust the support as 0.1. This would screen out rules with higher support. However, the result of this was 0 rule. Therefore, we changed support to 0.05 and improved confidence to 0.9 while changing minimum length to 4. Then, we got 26 rules. After summarizing these rules, we found all of their confidence were 1 while lift was 2.653. Thus, we inspected these rules and sorted them by support. Then we chose several interesting rules to analyze.

lhs	rhs	support	confidence	lift
{badge_product_quality=1, badge_fast_shipping=0, shipping_is_express=0}	{rating_level=high}	0.07138092	1	2.653097
{badge_local_product=0, badge_product_quality=1, badge_fast_shipping=0, shipping_is_express=0}	{rating_level=high}	0.06871247	1	2.653097
{badge_local_product=0, badge_product_quality=1, badge_fast_shipping=0, shipping_is_express=0, origin_country=CN}	{rating_level=high}	0.06604403	1	2.653097

{badge_product_quality=1, badge_fast_shipping=0, shipping_option_price=<3}	{rating_level=high}	0.06204136	1	2.653097

2.3 Results Analysis

For the first and second rules, they had the two highest support. We could found products with badges of product quality would contribute to gaining high rate of 5 ratings. Badges of fast shipping and local product wouldn't influence the high percentages of the rating 5.

For the third and fourth rules, we found that origin country of China would contribute to high level of rating. Besides, shipping price under 3 would also make contributions.

3. Sales Status Prediction

3.1 Specific Preprocessing

We selected variables that might contribute to classifying high sales, and we set high 'units_sold_level' as 1 while the others as 0. And for the preparation for SVM and ANN, we transformed 'product_color', 'product_variation_size_id' and 'origin_country' into numeric. Then we split the data frame into training and testing sets with 8:2. We tried 10 folds cross validation and default repeated times, which were set as Controls to apply in the models.

```
'data.frame':   1499 obs. of  16 variables:
 $ price                     : num  16 8 8 8 2.72 3.92 7 12 11 5.78 ...
 $ retail_price              : int  14 22 43 8 3 9 6 11 84 22 ...
 $ uses_ad_boosts            : Factor w/ 2 levels "0","1": 1 2 1 2 2 1 1 1 2 1 ...
 $ rating                    : num  3.76 3.45 3.57 4.03 3.1 5 3.84 3.76 3.47 3.6 ...
 $ badge_local_product       : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
 $ badge_product_quality     : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
 $ badge_fast_shipping       : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
 $ product_color             : Factor w/ 100 levels "applegreen","apricot",..: 89 36 44 8 100 59 89 15 8 7
 ...
 $ product_variation_size_id  : Factor w/ 91 levels "04-3XL","1","1 PC - XL",..: 44 84 84 44 50 70 84 45 44
0 ...
 $ product_variation_inventory: int  50 50 1 50 1 1 50 50 50 50 ...
 $ shipping_option_price     : int  4 2 3 2 1 1 2 3 2 2 ...
 $ shipping_is_express       : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
 $ inventory_total           : int  50 50 50 50 50 50 50 50 50 50 ...
 $ origin_country            : Factor w/ 6 levels "AT","CN","GB",..: 2 2 2 2 2 2 2 2 2 2 ...
 $ merchant_rating           : num  4.13 3.9 3.99 4.02 4 ...
 $ units_sold_level          : Factor w/ 2 levels "0","1": 1 2 1 2 1 1 2 1 1 2 ...
```

3.2 KNN

Before using KNN, we created a function named "norm_min_max" to normalize numeric variables. We used "knn" method to call KNN algorithm. Then, we set tuneLength as 40, which means there were 40 different k values. According to the details of the model, the best parameter is k = 53 with the training accuracy of 0.6974428.

3.3 Random Forest

We used 'ranger' to call Random Forest algorithm. Then, we set grid parameters, mtry = c(8,10,12), minimum size of nodes were c(1,2,4), and fixed splitrule as "gini". According to the details of the model, the best parameter set is mtry = 8, splitrule = gini and min.node.size = 4 with the training accuracy of 0.7958761.

3.4 SVM

Before we constructed SVM model, we tried to reformat categorical variables into dummy ones. However, variables like “product_color”, “product_variation_size_id”, “origin_country” had too many levels. So, we transformed them into numeric variables and set factors before modeling. We used “svmPoly” to call Polynomial kernel for SVM model. We set polynomial degree as c(2,3,5), scale=0.1, and C=c(0.1,0.5). According to the details of the model, the best parameter set is degree=1, scale=0.1, C=0.05 with the training accuracy of 0.6691738.

3.5 ANN

We used the same processed data as SVM and used “nnet” to call ANN algorithm. We set grid parameters, size = c(1,2,3), decay = c(0.05,0.1). According to the details of the model, the best parameter set is size=3, decay=0.1 with the training accuracy of 0.6791207.

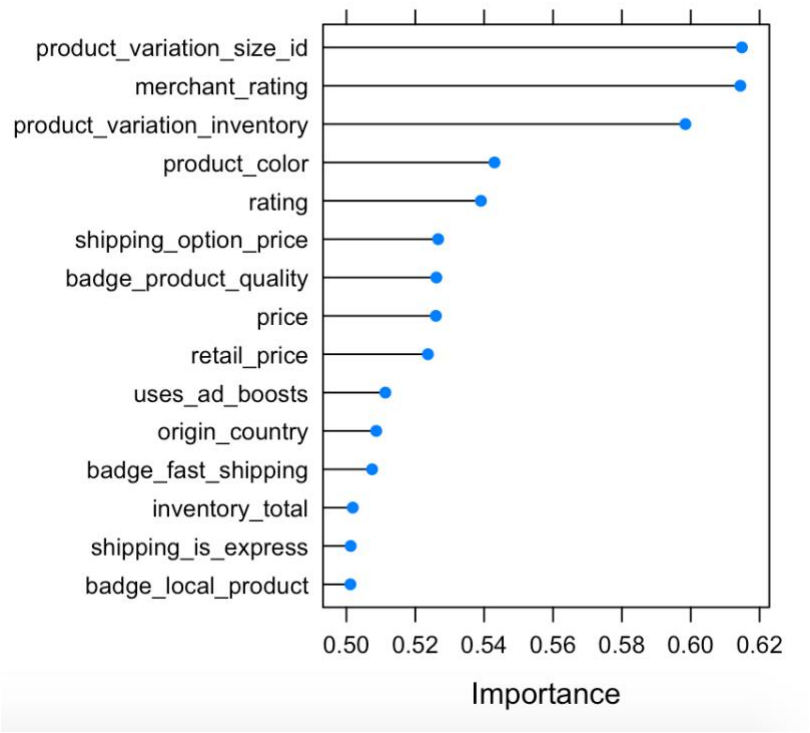
3.6 Results Analysis

From the results of each prediction, we found sensitivity was the same value with recall, and Pos Pred Value was precision. F-Measure can be calculated by these two figures.

Model	Training Accuracy	Prediction Accuracy	Precision	Recall	F-measure
KNN	0.6974428	0.6756	0.6973	0.9100	0.78957631
Random Forest	0.7958761	0.7023	0.9744	0.5700	0.71925408
SVM	0.6691738	0.6689	0.6689	1	0.80160585
ANN	0.6791207	0.6622	0.7071	0.8450	0.76992397

Since the training accuracy and prediction accuracy of Random Forest differs a lot, there may exist overfitting in the model. Besides, we found the positive value is 0, and what met our goals was to classify unit_sold_level=1. Therefore, recall and F-measure could be better to evaluate the models. According to this, we decided to choose SVM.

Then, we displayed the variable importance based on calculating area under ROC curve. As we could see from the diagram, the top three important variables are product_variation_size_id, merchant_rating, and product_variation_inventory. These factors could influence the unit sold level.


product_variation_size_id
merchant_rating
product_variation_inventory
product_color
rating
shipping_option_price
badge_product_quality
price
retail_price
uses_ad_boosts
origin_country
badge_fast_shipping
inventory_total
shipping_is_express
badge_local_product
0.50
0.52
0.54
0.56
0.58
0.60
0.62
Importance