# Purchase Prediction on Amazon Luxury Beauty Products

Will Powers

***Abstract:***
*For this study, Amazon review data was manipulated to replicate purchase history data and used to devise predictive analytical methods for surmising whether a user would purchase a given item or not. Logistics regression models were compared that had features utilizing similarity metrics to compare the given item with items in the user's purchase history. Among them, the cosine similarity model had the best performance. However, the most accurate model was an ensemble model of many of the previously analyzed models.*

**Motivation:**
Amazon is a competitive business that relies on sales to boost revenue. Its competitors for Luxury Beauty products, namely Sephora, rely on in-store experiences where sales representatives and experts can tailor match shopper's preferences to find the right products for them. To compete in an online landscape, Amazon will need to find similar ways that scale personalized recommendations by using user and item data from online interactions.

**Exploratory Analysis:**

The dataset under observation is a collection of product reviews and associated data from Amazon.com product pages. Specifically, the luxury beauty products were under observation and a 5-core subset of the data was used, such that users and items that remain have 5 reviews each. Further, I will keep only the user and item ids as these will be only necessary to my study and I will label all the user/item data points as "purchased" (coded as 1). Then for each user/item pair I will randomly sample an item that the user has not reviewed and add this new pairing to the dataset and label it as "not purchased" (coded as 0).

The data original included 34,278 data points, after appending the data points for non-purchased items, there are 68,556 data points under observation. The data points were then randomly shuffled before being split into training, validation, and test sets. 80% of the data was used for training and the remaining 20% was split equally between the validation and test sets.

Since the dataset under observations is somewhat simplistic, a robust exploratory analysis will not be needed or especially possible. Instead, it will be interesting to see how well the training set resembles the 5-core data structure of the original data and how equally split it is between purchased and non-purchased data. The training data seemed to still be equally split between purchased and non-purchased data with 50.08934432207717% being purchased data.

Surprisingly, the 5-core aspect of the data did not hold up as expected. In my training dataset, It was revealed the average user had approx. 13 reviews, with user A2V5R832QCSOMX having the maximum of 149 and user A10H63KUGJEPAJ having only 3. Likewise, the average item had approx. 31.8 reviews and product B0058TE4WI had a max of 388 and product B0012V7XU6 had a minimum of 9. The data set is not likely to have remained "5-core" after the inclusion of non-purchased data points, however, a minimum of 3 seems likely too low and a maximum of 388 seems widely large for this to have come from a 5-core dataset. So, what is necessary next was to confirm that an error had not been due to my processing and to do the experiment again on the original dataset.

In the original dataset, the average user had approx. 7.3 reviews, with a maximum of 90 for user A2V5R832QCSOMX and a minimum of 1

for user A2P6QV3P7ENWPR. Likewise, the average item had approx. 17.8 reviews with a maximum of 440 for item B0006PLMFQ and a minimum of 1 for item B0002Y5JEG. Thus, it seems the data's 5-core guarantee does not live up to its hype. That being said, everybody makes mistakes, and regardless, the data is still useful and dense enough for our analysis.

**Predictive Task:**

The Predictive task that can be studied on this dataset that has been constructed is when given a user-item pair, to guess whether or not the user will purchase the item. To evaluate the model on this predictive task, classification accuracy will be used, which is the sum of true positives and true negatives divided by the total number of predictions. For a baseline, we will consider the top items that comprise the $50^{th}$ percentile of purchases, for a user-item pair, if the item is one of those top items, then the baseline will predict purchase (coded as 1), otherwise, it will predict not purchased (coded as 0). Using the training set to construct the set of top items and testing it using the test set, the categorization accuracy achieved is 0.6782380396732789. Thus, the model we construct will have to do better than this on the test set. Further, an even lower baseline can be used in which all pairs will be predicted as "not purchased", which achieves an accuracy of 0.5070011668611435.

To assess the validity of the model's predictions we will compare the predictions to the original purchased or non-purchased status of the data and use the categorization accuracy to get a summary of the overall validity, comparing it to the baseline and other iterations of the model (using validation data) to gauge the robustness of the final model. Since the output variable (purchased or not) is a Boolean variable, using a classifier seemed to be the best choice of model for this predictive task. Specifically, logistic regression would be used due to its Boolean classification system and the Sklearn package's implementation will be the one utilized for this study.

Since the data was dense, it had a lot of items that had been reviewed by multiple users and a lot of users that had reviewed multiple items. Therefore, it felt appropriate to feed similarity metrics into the model as features for each user/item pair. The similarity metrics used were Jaccard similarity, cosine similarity, and Pearson similarity, and they were processed in the following manner: for each user/item pair, the similarity metric was taken between the item and every other item that the user had purchased. Then the maximum similarity metric between them was used as a feature.

Additionally, the popularity of an item was considered as a feature in the following manner: for all items, the counts of how many times each were purchased were collected; For a given user-item pair its count was divided by the max count of all the items and that quotient was used as a feature for popularity. Other data from reviews could have potentially been used however the methods for analyzing them were too complex and time-consuming of a task for the researchers on this study.

**Model Descriptions.**

In this section, it's required to explain and justify my decision to use the model I proposed, much has already been said to that effect. However, I can go into more depth since I did use a more complex model than the one proposed in the previous section. Using all the aforementioned features for one classifier ran into the problem of overfitting on the training set because so much data was available the model could hack at the data to get the correct result (William of Ockham would be ashamed). So, to optimize it an ensemble model would be used. A logistic regression model would be trained individually for each of the aforementioned features and a bias term and then the predictions for each would be used as features for a logistic regression model that ensembles the previous models.

Further things to note was that the Pearson similarity model would not be utilized in the ensembled models as it performed well below both baselines and would have an adverse effect on the ensembled model. Also, interesting that

most models seemed to perform equally well on training, validation, and test sets, however, the cosine similarity model did overfit the training set significantly by 10%. Changing the regularization parameter on the model did not have any effect and regardless, the cosine model still performed much better than the other individual models, pre-ensemble.

I did not run into any scalability issues. The dataset was sufficiently small (68,556) that all steps ran in a relatively quick amount of time on a moderately built MacBook Air M1 chip with 8GB of memory.

The other models I considered for comparison were the 2 aforementioned baselines as well as the individual models that were used to feed into the ensemble model

I did not have any unsuccessful attempts along the way. I had frustratedly attempted ensembling on assignment 1 so many times that it was quite easy this time. My approach was different this time in that instead of trying to devise a systematized way of doing it that had a lot of kinks and did not save its place in the Jupyter Notebook but rather had to be restarted over and over again; this time I individually constructed each model and validated and then constructed the ensemble from the results separately.

Additionally, one thing that had to be changed was that I originally was using the raw counts for the popularity scores, but I got much better results when normalizing the counts by dividing each by the maximum count.

The strength of the popularity model is that it is the fastest by far to implement and it has relatively decent accuracy, although not the best and it performs below one of the baselines. The strengths of the Jaccard model are similar, it is somewhat slower than the popularity model, but it has somewhat better accuracy. The Pearson model is by far the worst-performing and the slowest of the individual models to implement so it has mostly downsides since it performs worse than just guessing 1 or 0 for all user/item pairs. The Cosine model is slower to implement than all other individual models however it performed by far better than all of them on the task. The ensemble model performed better still, although only slightly better than the cosine model, yet it is definitely the slowest model to implement as it requires first the implementation of several other models.

If the speed of implementation was a concern over the accuracy, one may prefer the cosine model to the ensemble one and if speed was even more of concern perhaps it would be better to go with the Jaccard model which is slightly faster to implement.

**Related Literature to Problem**

The dataset used for this study was provided by Jianmo Ni, Jiacheng Li, and Julian McAuley and was constructed during their study "*Justifying recommendations using distantly-labeled reviews and fined-grained aspects*" which was published in *Empirical Methods in Natural Language Processing (EMNLP)* in 2019.

The dataset was originally used for item recommendation systems to use natural language processing and generation to justify the item recommendation from information from user reviews about the item. For this task, the dataset had been modified and stripped of all NLP data to observe item-to-item similarity.

Looking at the literature for purchase prediction it seems much of the studies supplement their datasets with clickstream data [2]. Of the remaining few that use transaction data alone utilize very different methods were used than the current study. Mathias Kraus and Stefan Feuerriegel in their study [3] utilize temporal data to track users' purchase history over time and create a personalized recommendation of purchases based on sequences. Specifically, they use Wasserstein-Based Sequence Matching.

Other papers use deep learning approaches to create purchase predictions. In a paper by Chao Huang, et al. [4] temporal data is also utilized, and Graph Multi-Scale Pyramid Networks are used as well as Convolutional Neural Networks.

A further study by Abdul-Saboor et al. [5] uses deep learning approaches where collaborative prediction models fail due to sparse data. A benefit of the dataset used in this study was that it was relatively dense although that might not model real-world conditions accurately.

The studies perform relatively well although measure their success using F1 scores and AUC scores. Although even if this study was translated to their metrics, it still would likely not be comparable due to the fact of the unusual density of the dataset used for this study. Likewise, their conclusions relate to predictive methods wholly different from the ones used for this study, so they are not very comparable to the findings in this analysis.

**Results and conclusions.**

| Model | Accuracy Score |
|---|---|
| Pearson Similarity | 0.4369894982497083 |
| Baseline (Always 0) | 0.5070011668611435 |
| Popularity | 0.661026837806301 |
| Baseline (Popularity) | 0.6782380396732789 |
| Jaccard Similarity | 0.6843640606767795 |
| Cosine Similarity | 0.7297257876312718 |
| Ensemble Model | 0.7342473745624271 |

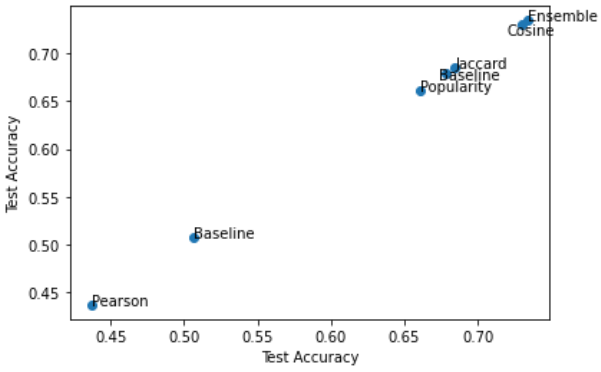


**Table 1 & Figure 1: Categorization Accuracies of different models**

The Ensemble model outperformed the baselines and the alternative models that were used for ensembling. It can be said that the ensembled model outperforms most others by a wide margin except for the cosine similarity model. It only barely beat the predictive power that the cosine similarity logistic regression model had to offer. Therefore, the ensemble model may not be significantly better to justify using it over the cosine model however it did beat the accuracy of the cosine model with training, validation, and test datasets.

The feature representation that did not work well was the Pearson similarity feature representation. The rest performed reasonably well except for cosine similarity which was noticeably better than the rest on the accuracy score.

Turning to the coefficients of the ensemble model we have the bias term with the coefficient 7.05309591e-05, which means the model very slightly favors guessing an item will be purchased.

The coefficient for the Jaccard prediction features is -1.83647304e-03, which initially is strange because it disfavors Jaccard predictions and assumes they will contrarily not be purchased. I believe this has more to do with the fact that much of the accurate predictions of the Jaccard model will also be predicted by the popularity and cosine models so the model will weight those models' data high and any leftover predictions that have been guessed as purchased by only the Jaccard model will likely be incorrect. This seems to show that the Jaccard model could have been taken out of the ensemble and the accuracy would likely not have been affected very much.

The popularity prediction features had a coefficient of 2.18849751e+00 which meant it was strongly correlated with successful predictions. The cosine prediction features had a coefficient term of 3.10299367e+00 which suggests it was by far the best predictor of an item being purchased by a given user.

The ensemble model succeeded and beat all other models mostly because it incorporated all other model's predictions into its own analysis, weighting them by how useful they are at predicting user/item purchases in comparison to other models. In this way, the model could have

done at least as well as the best model by weighting that model 1 and weighting all the rest as 0 (via the coefficients of logistic regression). So, it's no surprise that the ensemble model did better than all others in this way although it only barely edged out the predictive power of the cosine model alone and it seems the other models had very few improvements to make upon the predictions that the cosine model offered.

**Citations**


**[1] Justifying recommendations using distantly-labeled reviews and fined-grained aspects**
Jianmo Ni, Jiacheng Li, Julian McAuley
*Empirical Methods in Natural Language Processing (EMNLP)*, 2019

**[2] literature-review-purchase-prediction**
Douglas Cirqueira
*https://github.com/dougcirqueira/literature-review-purchase-prediction , 2019*

[3] **Personalized Purchase Prediction of Market Baskets with Wasserstein-Based Sequence Matching**
Mathias Kraus, Stefan Feuerriegel
*https://arxiv.org/abs/1905.13131, 2019*

**[4] Online Purchase Prediction via Multi-Scale Modeling of Behavior Dynamics**
Chao Huang, Xian Wu, Xuchao Zhang, Chuxu Zhang, Jiashu Zhao, Dawei Yin, Nitesh V. Chawla
*KDD '19: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019*

**[5] A Deep Learning System for Predicting Size and Fit in Fashion E-Commerce**
Abdul-Saboor Sheikh, Romain Guigoures, Evgenii Koriagin, Yuen King Ho, Reza Shirvany, Roland Vollgraf, Urs Bergmann
*Thirteenth ACM Conference on Recommender Systems, 2019*