Predicting Users’ Makeup Rating on Ulta Products

DATASET

For this assignment, we used the *Luxxify: Ulta Makeup Reviews* (linked) dataset from Kaggle, which contains information about different products available on the Ulta website for purchase and about individual reviews submitted to the site, each in different files.

The file with information about products includes factors such as the product’s category (what type of complexion product it is, i.e. foundation, concealer, blush, etc.), the brand, price, average rating, and the number of times it was reviewed. It also includes a breakdown of the number of ratings given for each star (ranging from 1 to 5). The number of times the product was recommended by reviews was also included. There were 1375 products, with 321658 reviews made on them in total.

The file with information about individual reviews tells us the star rating a user gave a product and also includes the comments and review tagline they assigned. Additionally, to ensure that reviews could be trusted, whether the review was made by a verified reviewer or buyer was also noted. For our purposes, we ignored all entries in this data file that gave direct information about the review itself, to ensure that our predictions were not based on the reviews themselves and instead of the relationship between the users and the products in our database.

Looking at the products included in the dataset, a majority of them are face primers, closely followed by blush and setting products (Figure 1). However, if combining foundation, tinted moisturizer, and BB/CC creams into a single category, we see that these base complexion products have almost as many products.

When looking at the price distribution of the products (Figure 2), we see that there are many products under \$20 (likely the large collection of drugstore brands that Ulta sells in-store and online), and then a more normally distributed array of products centered around \$40, with the most expensive products being above \$80.

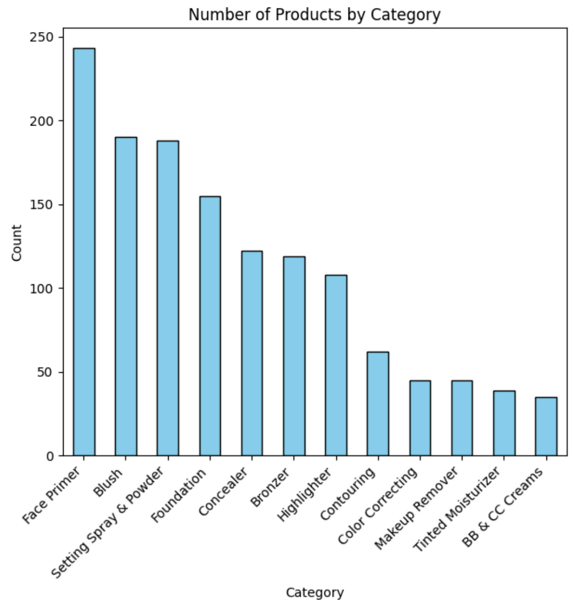

Figure 1: Category distribution of all products

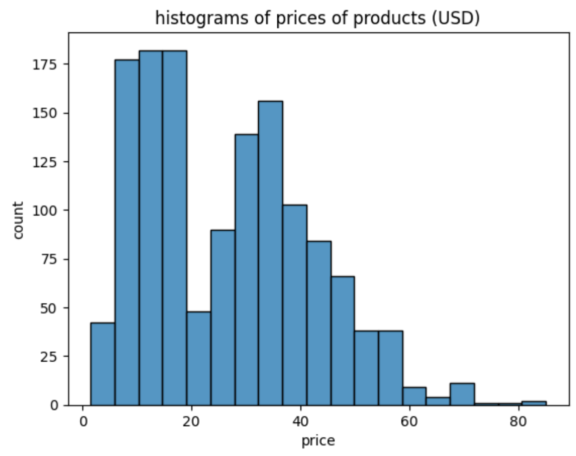

Figure 2: Histogram of price distribution of products

When looking at the distribution of product ratings, we can see that most products are rated rather highly, with few products receiving less than 3 stars. This was surprising, as we were expecting a slightly more even range of ratings. However, upon further review, this makes sense, as most of the makeup products at Ulta are not terrible quality and have some form of a customer base.

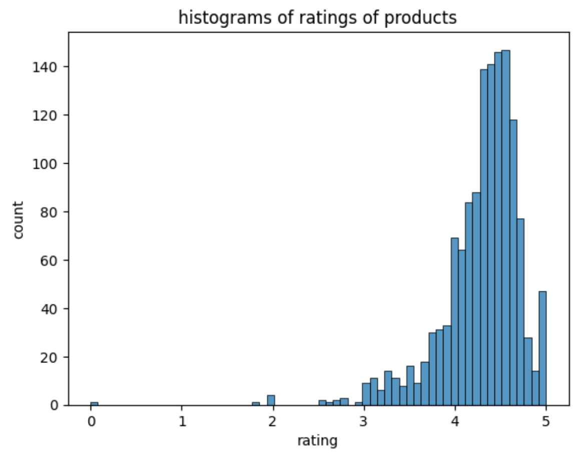

Figure 3: Histogram of product ratings

Interestingly, none of the reviews in our dataset were from Ulta staff members. Additionally, very few reviews were made by either verified buyers or verified reviewers. This suggested to us that a majority of the people who rated products likely bought them in-store or perhaps from other retailers or websites. For our purposes, this does not cause any issues. However, this does raise a bit of concern for the average consumer, since so many of these reviews cannot be verified. This could also be the reason why the ratings are relatively higher than expected.

PREDICTIVE TASK

For this assignment, we implemented a series of models to predict how a user would rate a product. We used a mix of product information already available in the raw dataset, combined with a few features we computed separately. From the original dataset, we used information about the product's category, brand, price, amount of times each star rating was given (e.g. How many people rated it 1, 2, 3, 4, or 5 stars), and the ratio of users that recommended the product to other customers.

In addition to these features, we added a few more. First, we added the total number of upvotes that a user has received over all their reviews. Our rationale here was that reviews that are more helpful likely have either a very high ranking or a very low ranking, and possibly include some helpful comments about the product itself. Thus, how useful a person has been in their reviews could be a useful metric for our predictive task. Additionally, we calculated the Jaccard between the users we are predicting and other users that have rated the products. This Jaccard similarity was based on the agreement between the products each of the users rating.

As a baseline for our predictions, we will be predicting that the rating given to the product was the median rating present in the dataset. We felt this was a valid baseline, as most of the reviews centred around 4-5 stars, meaning our median prediction has a high chance of accuracy. We evaluated this baseline and all the models we used by examining the root mean squared error (RMSE) and the mean absolute error (MAE).

MODELS

In total, we tested 7 different models.

<u>Model 1: Baseline</u> For this model, we predicted that any given user would rate a product at the median rating of all entries in the training set. This model uses the median of the ratings in the training set as the predicted rating for all test instances.

<u>Model 2: Singular Value Decomposition (SVD)</u> Model from class lectures. This model factorizes the user-item interaction matrix into latent factors for users and items. We used the implementation of this model in the surprise

library. SVDs are very useful for rating prediction tasks like we are doing here.
Model 3: Linear Regression Model from class lectures. This model finds a line of best fit through our data points that minimizes the error.
Model 4: Random Forest This model is a machine-learning method that builds decision trees during training and outputs the average of their predictions. Random forests work well for large datasets that are very large, with a mix of features, which our dataset generally falls into. For our tests, we trained a random forest model with 100 estimators or trial trees. The model implicitly chooses the best-performing tree and uses it for all downstream testing.
Model 5: FastFM Model from class lectures. Generally, FastFM performs very well for recommendation systems and recommendation-based tasks. We tried a number of different factors ranging from 5-10 to optimize the model to run efficiently and set both L2 weight penalties to 0.01.
Model 6: AutoRegression Model from class lectures. As an experiment, we decided to try to treat our dataset as time series data and fit the ARIMA model, just to see how a less-than-ideal model would perform on our dataset. We set the autoregressive order p to 1, and the differencing order q and moving average order q to 0.
Model 7: Neural Network We used a simple neural network generally designed for regression tasks. We created a multi-layer model with 3 layers of sizes 64, 32, and 1 respectively. We used ReLU as the activation function and ran it for a range of epochs, finally settling on 25 as the optimal number for the model.

In the process of training these models, we ran into a few issues, mainly with successfully training a FastFM model. For the other models we trained, we were able to achieve a relatively low RMSE and MAE. However, our RMSE for FastFM was extremely high. We attempted to solve this problem by scaling the values in each column to be normally distributed, but this did not solve the problem. We are unsure about what the problem is and are surprised this model did not work for our dataset, as FastFM is generally good for recommendation-based tasks such as the one we identified for this assignment. However, since we are doing more prediction rather than actual product recommendation, perhaps this model was ill-suited to our dataset.

For comparison, it would be interesting to see whether these features could be used to identify whether a person has tried a product before, rather than whether they enjoyed the product itself. In the case of FastFM, identifying whether a person tried out a product might be more in line with the intention of the model, rather than rating prediction. It would be interesting to see how the other models we tested in this assignment work for this task as well.

LITERATURE

The dataset we used was obtained from Kaggle, and it contains makeup review data used to train the luxxify makeup recommender. While searching for datasets we came across a number of data sets for other types of recommendation systems and predictive models, including but not limited to: Amazon food review, IMDB movie reviews, book reviews and recommendations, Amazon product recommendations, pizza topping recommendations and anime recommendations. Many of these datasets followed a similar format to ours; however, most of these datasets were specifically designed to recommend products rather than allowing for flexibility for the type of predictive modelling task we wanted to perform.

There are a number of state of general art methods used to solve recommendations and predictively model ratings, listed below:
Probabilistic Matrix Factorization (PMF): Similar to SVD, this method identifies latent factors present in user, item pairs. This model

typically uses Stochastic Gradient Descent (SGD) or Alternating Least Squares (ALS) to minimize the objective function. This model is good for sparse and large datasets. Since it uses a probabilistic model it is extendable to work on many datasets (Hu et al., 2009; Koren et al., 2009).

Singular Value Decomposition (SVD): SVDs find patterns in data by inferring the relationships between users and different items in latent spaces. This model performs well for recommendation systems as seen in the previous assignment. There are several recent hybrid approaches which combine SVD with neural networks, making them more attractive to use for recommendation systems. These datasets are great at handling large, sparse data, and identifying the patterns (Koren et al., 2008). Below are some examples of these neural network-based models

RNN: These models are used to capture sequential information and are useful for session-based recommendations. Session-based recommendations are recommendations made for users based on their behaviour in other sessions on the website we are collecting data from. RNNS are also useful when there is no easily identifiable long-term behavior, and when the behavior we are interested in predicting only has consistency in the short-term, given the limited attention of the model.

DeepFM: DeepFM combines factorization models (FM) and neural networks. Low-order interactions are captured by the FM and the higher order are captured by deep neural networks. This helps us capture non-linear interactions. This method is used in applications like music, movies and digital advertising (Guo et al., 2017).

Graph Convolution Networks (GCN): This method is extremely useful for making predictions on graph-based data. A great example of this is social media data and other forms of heterogeneous data that have interactions between different measures. Users and items are denoted with nodes and the edges capture interactions. The model updates nodes to aggregate information from other neighbour nodes over all the layers. This model is state-of-the-art in offering personalized recommendations to users (Kipf & Welling, 2016).

In general, existing work concurs with our findings in that SVD-based methods and neural network-based methods perform well for rating and recommendation classification. However, our results for FastFM did not align with the literature, as this model performed surprisingly poorly (see results).

RESULTS

As stated above, we evaluated our models using the RMSE and MAE. Our baseline model, which predicted item ratings as the median of all training items, gave us an RMSE of 1.14 and an MAE of 0.53. Our code is available at this link.

Out of all the models we trained, 6 out of the 7 performed better than the baseline model. Our SVD model performed better than the baseline model, with an RMSE of 0.4323 and an MAE of 0.1954. Linear regression also performed better than the baseline model we greeted, with an RMSE of 0.9443 and an MAE of 0.68814. Our random forest model had an RMSE of 0.59223 and an MAE of 0.30065. Our autoregression model had an RMSE of 1.3591 and an MAE of 0.9430, performing the closest to baseline out of all the models we tested for this assignment. Our final model that performed better or equal to the baseline, the neural network, had an RMSE of 0.9027 and an MAE of 0.65184.

The FastFM model was the only one we tested that performed poorly on our dataset. Surprisingly, even though we expected this model to be well-suited to the task, the RMSE was 160073.22 and the MAE was 93222. These results did not improve when we attempted to scale the dataset and account for any irregularities in our features.

Overall, All models except for FastFM beat the RMSE of the baseline model. Surprisingly, a number of models did not beat the MAE of the test dataset, with linear regression, autoregression, FastFM, and our neural network returning MAE values higher than the 0.53 achieved by the baseline model.

Based on the results of these models, we conclude that the SVD model we trained performed the best on the dataset, with both the lowest RMSE and the lowest MAE out of all the models we tested (Figure 4). Aside from the FastFM model, the auto-regression model we trained performed the second worst, with an RMSE extremely close to the baseline model and an MAE that was actually higher than our baseline.

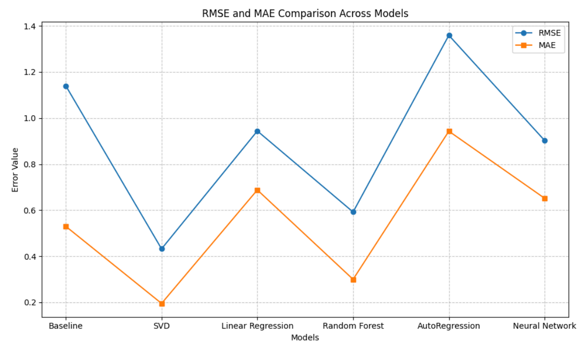

Figure 4: Line plot comparing RMSE and MAE between all models tested

CONCLUSIONS

In conclusion, the results of our experiments on this makeup data suggest that for some tasks, it may be better to not engineer a series of features that all roughly correlate or correspond to each other. Instead, in some instances, it may be better to form connections between users, products, and ratings without any outside influence. However, if we were to change our goal from rating prediction instead to an actual recommendation system, then this additional information would be necessary in order to formulate a more informed opinion.

However, with even more carefully informed features based potentially on some kind of demographic information, we could better enhance our predictions. For example, if we know a person's skin tone from other products they have bought and we know that the new product they bought and are going to review does not match their skin tone, we can predict that it may be slightly more likely for that person to rate the product lower.

We could also incorporate other product information - if a person usually tends to prefer more powdery blushes instead of cream blushes, they may be more likely to rate a cream blush lower depending on whether the formula is something we can predict if they would enjoy. By incorporating this type of information into our predictive models and recommender systems, we can make more informed recommendations for a target customer base and understand why people might enjoy specific products over others.

CODE

Code for this assignment is available at https://colab.research.google.com/drive/1sy-h-UR2pPvm076vSh0DmuQ-M6lM4msc

CITATIONS

Guo, H., Tang, R., Ye, Y., Li, Z., & He, X. (2017). *DeepFM: A Factorization-Machine based Neural Network for CTR Prediction*. http://arxiv.org/abs/1703.04247

Hu, Y., Koren, Y., & Volinsky, C. (n.d.). *Collaborative Filtering for Implicit Feedback Datasets*. Retrieved December 3, 2024, from http://dx.doi.org/10.1109/ICDM.2008.22

Kipf, T. N., & Welling, M. (2016). *Semi-Supervised Classification with Graph Convolutional Networks*. http://arxiv.org/abs/1609.02907

Koren, Y., Bell, R., & Volinsky, C. (n.d.). *Matrix Factorization Techniques for Recommender Systems*. Retrieved December 3, 2024, from https://doi.org/10.1109/MC.2009.263