Clothing Fit Size Recommendation System

Ryan Liao
rsliao@ucsd.edu

William Nathan
wnathan@ucsd.edu

Alexei Davis
a8davis@ucsd.edu

University of California, San Diego
November 29th, 2022

Abstract

One of the biggest disadvantages of online shopping for fashion compared to brick-and-mortar stores has always been finding the perfect fit. The average online shopper is unable to interpret the measurements well enough to be confident that the apparel fits. This paper seeks to mediate this problem by evaluating the performances of various models in predicting a comfortable size for users.

Introduction

A big barrier of entry online fashion retailers have to overcome is the convenience of trying out sizes without returns. Current measurement standards are not detailed enough to ensure proper fit with garments like dresses and the like, but if the measurement system gets any more complicated, users will not be able to keep up. A compromise that many papers have looked into is to extract better measurements from information that we already have through machine learning. Normally, someone who knows they are a size 10 from the t-shirts they wear may not necessarily be able to wear a size 10 dress. A machine learning model can overcome this by also taking into account the type of garment along with other metadata to figure out what size a user should be for all types of garments. This paper proposes using XGBoost to make a recommendation system trained with the Renttherunway dataset. Unlike other implementations, we chose the task of predicting a recommended size instead of predicting whether an apparel will fit.

Literature Review

The academic paper by Misra, which provided us with the Renttherunway dataset we are using, tackled the same problem with a different approach. Their dataset consisted of fit feedback from Renthterunway and ModCloth totalling to 275,334 transactions. The objective was to accurately predict clothing fit based on a variety of factors. With this data, they proposed modeling fit semantics with multiple latent factors. They used a metric learning approach which takes in the user IDs, item IDs, sizes, and categories, and compared the proposed model to 4 other baseline models. This proposed model performs better for cold start prediction at the cost of overall performance. Although, since our solution aims to predict size without using prior user or item history, we have the advantage of not having to deal with cold starts.

Another academic paper attempted to create a clothing size recommendation system using data from Chinese online shopping malls. The results showed a 88% match rate for women and an 80% match rate for men. This was considered quite impressive, considering the high return rates citing wrong sizing as well as a 2017 article which found that 16% of people could not accurately pick a size online. The format of this system was an app that used image processing and accumulated a variety of key factors in determining an optimal size: bust length, waist length, hip length, shoulder width, sleeve length and pants length. The user only had to input their height and weight. The final recommendation was geared towards apps with a specific UI and process in which anyone could take a picture of themselves in order to get size recommendations for whatever they wanted to buy. The solution proposed by this paper shared the same predicting task of recommending clothing size, which they did through SVM and RBF. They reportedly had an accuracy over 98%, though we could not replicate their workflow, since it includes changing the measurement standards and platform.

This last paper we looked at also attempted to create a clothing size recommendation system for online shopping. This time, the chosen method was a skip gram based Word2vec model to learn the latent representation of products. Then, gradient boosting classification was used to predict the preferred size of a user. The classification was binary for whether a product was returned or retained. The data sample was from Myntra Design sales from January 2015 to February 2017, with the training data having 1.4 million samples, the validation data having 0.4 million samples, and the test data having 0.45 million samples. Three different types of classification models were experimented with: observable features, latent features, and combined features. The evaluation used precision, accuracy, and area under curve as performance metrics. The combined features model ended consistently performing the best in every metric. Similar to the Misra paper, this one also used latent variables to predict fit. Though part of their workflow involved modeling for size, which we drew inspiration from.

For our model, we have also decided to implement a gradient descent model. However, we will be using different performance metrics and we will not be incorporating latent representation.

Exploratory Data Analysis

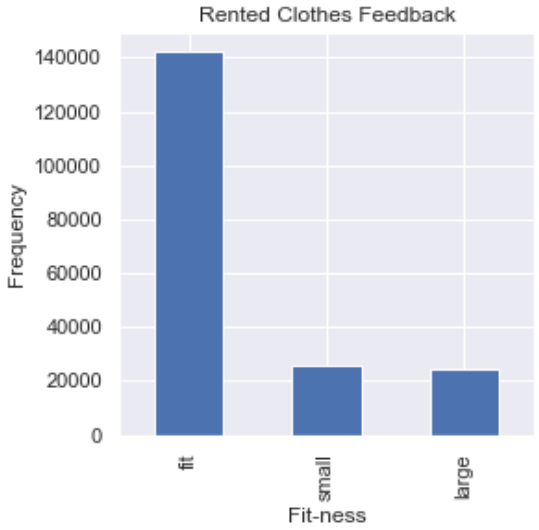

Figure 1

We will be performing our analysis on the dataset from RentTheRunway collected by Misra, a site that provides rental service for designer clothes. The data contains feedback from clients who have rented a

piece of clothing, along with the apparel and user metadata. It is formatted as individual transactions, with up to 15 potential variables describing each transaction. The variables range from describing the user who reviewed the product, information about the product itself, and information about the review. For our purposes, we were only interested in variables containing data on the user and what they were reviewing. In addition, we also calculated these statistics of interest in Table 1.

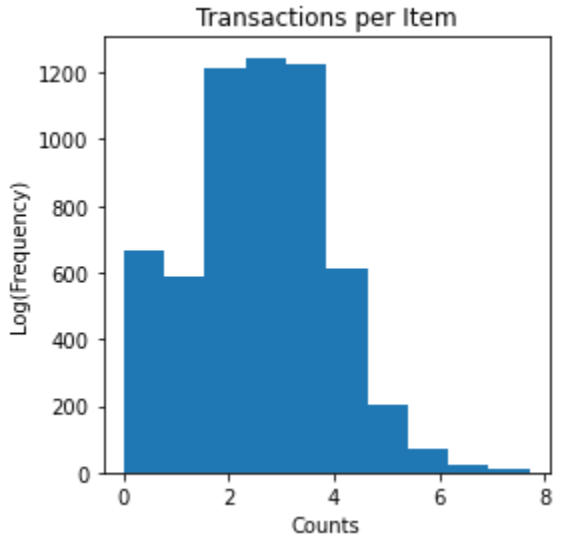

Figure 2

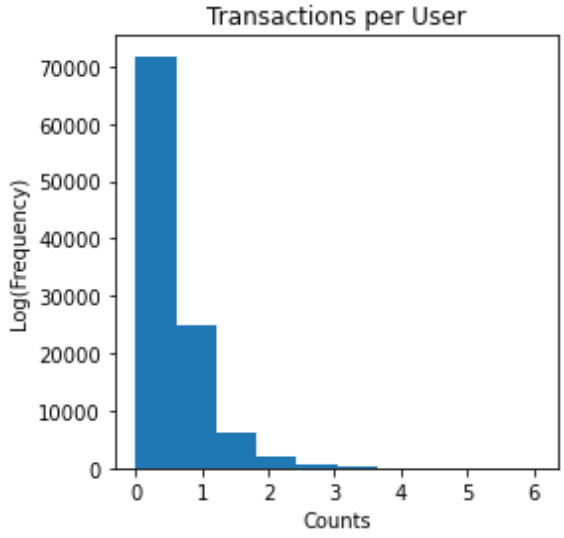

Figure 3

Statistics	Value
Transactions	192,544
Users	105,508
Items	5,850

Table 1

We see that the variable fit is highly imbalanced, with the majority of users reporting that their clothes fit. From figure 2, we can also observe that most users have only done a few transactions, which means latent factor models like the one proposed by Misra will suffer from cold start very often. Knowing this, we opted for a model that does not take the user's or item's history into account. By intuition, the variables we used to predict the ideal size are bust size, weight, body type, category, height, size, and age.

The data itself was relatively raw and messy. This required some cleaning and some parsing in order for these variables to be of any use. First, we need to parse these variables into useful features. We remove transactions that do not fit, since we want to predict sizes that would fit, leaving us with 141,995 data points. There are data entries with cup sizes "dd" and "ddd" in bust size, implying that Renttherunway uses the US cup size standards. We can parse bust size into the band size (numbers in front) and cup size, which can be converted to numbers from the letters. These sizes are separate since we suspect they affect size differently. Although in our table, they are mislabeled as bust instead of band. For weight we simply remove the units attached

to them, which are all standardized to pounds. Body type and category are one-hot encoded. Height is parsed and converted to inches. Now that we have machine-readable features, we check for correlation between our features.

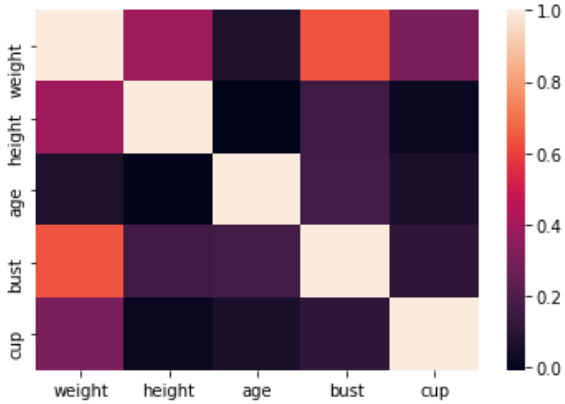

Figure 4

There doesn't seem to be any high correlation between our features, so we are good to go.

Methodology

To evaluate the effectiveness of our proposed solution, we will be comparing the MSE statistic with a baseline linear regression model.

Given that all the columns in the cleaned dataset are numerical, the baseline model we used was a Linear Regression model. For the initial model we split data 50/50 into training and test sets. We then fit a Sklearn Linear Regression model with the training data and used the trained model to predict sizes based on the feature columns in the test set. The initial model produced a MSE of 26.67. We used this MSE as a baseline to measure the accuracy of our models, as we felt that other models were capable of performing better than a Linear Regression model.

While the initial model produces a reasonable MSE, when looking at the data itself there is plenty of evidence that a better model can be used for the prediction task at hand. To begin with, the data still needs to be further cleaned in order to use a Linear Regression model. When looking at the feature columns of the data before being split into training and test sets we can see that the data still has a large quantity of NaN values. Below is a table which shows the number of NaN values in each column (one hot encoded columns are not shown since those columns only have values of 0 or 1).

	NaNs
weight	22305
height	486
size	0
age	705
bust	13661
cup	13661

Table 2

A problem with the Sklearn Linear Regression model is that it doesn't have the capabilities to deal with NaN values. Therefore, prior to fitting the initial model we first had to fill all of the NaN values. We chose to fill all NaNs in a certain column by imputing the mean value for that column. Therefore, the data that the Linear Regression Model was trained on had all of its NaN values filled with mean values, which could lead to inaccuracies in the model as every NaN has a real life value associated with that user, but that value may not be close to the mean, which leads to either a poorly trained model or an

inaccurate prediction for that transaction. Furthermore, a Sklearn Linear Regression model might not be optimal for this situation as not all of the feature data is necessarily linearly related. Seen below is a table of the r correlation coefficients of the same 5 main features (weight, height, age, bust, cup) compared to the target column, “size.”

	r
weight	0.860715
height	0.236283
age	0.158115
bust	0.667974
cup	0.335050

Table 3

As we can see the linear correlation between some of the features is close to 0 which suggests a weak linear relationship between some of the features and the target, which would result in inaccurate results for the initial model. In particular, age was not as significant in linear regression, because intuitively size does not grow linearly with age. As we expected, cup size is not as significant as band size, as it is more flexible. Lastly, we also felt that a Linear Regression Model was not the best choice for the prediction task given that our prediction was going to need a more complex or more efficient model than a Sklearn Regression Model. Therefore, we decided to keep the Sklearn Regression Model and move onto a different model.

We knew that the model we would end up using would be a regression based model, and the one we settled on was an XGBoost Regression Model. This model was better for the prediction task as it uses a gradient descent approach to achieve its objective of minimizing the squared error. Other advantages include its ability to handle NaN values in the training data and it has parameters that allow for more precise model tuning. Furthermore, XGBoost is rather time efficient as it has similar run times as Sklearn Linear Regression, and scalability is not an issue for either models. To begin with, we fit an XGBRegressor model with the training data (NaNs still in the training set this time) and used default parameters other than the objective parameter objective=”squarederror”, which tells the model that we are trying to recommend size values based on minimizing the squared error of the training set. This initial XGBoost model produced a MSE which was significantly better than the baseline model.

This supported the idea that XGBoost was a better approach for our problem compared to the baseline model. However, improvements could still be made to the XGBoost model. The first thing we addressed was the training/test split. With both the Linear Regression Model and XGBoost model the initial split was 50% training and 50% test. However, we decided that the model would perform better and would not be in danger of being overfit if we changed the split to 80% training data and 20% test data. When refitting the XGBoost model with the new training/test split we got a slightly improved MSE.

Lastly, we also were able to improve the accuracy of the model by tuning the parameters of the XGBRegressor. While most of the default parameters were kept, a few parameters were changed that allowed for the model to once again be slightly improved upon, without having parameters

that encourage overfitting. This tuned model is the final model that we used for size recommendation.

Results

The final version of our XGBRegressor model produced a mean squared error of 17.91. This result was a significant improvement on our linear regression model. Running the model with different parameters and different training and test splits ultimately yielded worse results than the ones aforementioned. A 50/50 training test split yielded an 18.49 MSE and an 80/20 split without any parameters yielded a 18.05 MSE. While these differences were very slight, we are confident that our methodology and our thought processes were ultimately backed up by the final numbers.

	importances
weight	0.351128
height	0.012091
age	0.013517
bust	0.124724
cup	0.028381

Table 4

Our XGBoost model had similar feature importances with the linear regression. It was unable to extract as much semantic insight from age and height as we previously expected.

In the final XGBRegressor model we used some of our own hyperparameter values in order to make our model even more accurate. Specifically we set the max_depth of the model equal to 10, colsample_bytree equal to .5, and alpha equal to 1. Max_depth determines the depth that trees are allowed to grow and 10 was the value that produced best results. Colsample_bytree determines the percentage of features used in each tree (from 0 to 1). Higher values for this hyperparameter yielded more accurate results, however, a higher value of colsample_bytree may lead to overfitting so we settled on .5. Lastly, alpha=1 is a regularization hyperparameter and this value yields best results and is also a common value for alpha. All other hyperparameters were left at their default values.

References

1. Abdulla, G. Mohammed, and Sumit Borar. *Size Recommendation System for Fashion E-Commerce*.
2. Misra, Rishabh, et al. “Decomposing Fit Semantics for Product Size Recommendation in Metric Spaces.” *Proceedings of the 12th ACM Conference on Recommender Systems*, 2018, https://doi.org/10.1145/3240323.3240398.
3. Yuan, Ying, et al. “A Proposal for Clothing Size Recommendation System Using Chinese Online Shopping Malls: The New Era of Data.” *Applied Sciences*, vol. 11, no. 23, 2021, p. 11215., https://doi.org/10.3390/app112311215.