Cloth Fit Recommendation using Latent Factor Models

Sreekrishna Ramaswamy, Yongze Yao
UC San Diego
La Jolla, CA, USA
sramaswa@eng.ucsd.edu, yoyao@ucsd.edu

ABSTRACT

The problem of not having trial rooms to check whether a piece of cloth fits is aggravated in an online setting. This often results in a lot of product returns and that causes loss to the online aggregators and sellers on the myriad of online platforms. To solve cloth fit recommendation problem, we have tried out a few commonly known variants of the latent factor model and estimated why a particular model is/isn't suitable for this problem. Prediction of fit **{Small, Fit, Large}** for a sparsely populated dataset has been presented in this work. The latent factors for customer, product pairs that correspond to their physical true sizes are learnt from the past product purchase data. The outcome for a customer, product pair is predicted based on the difference between customer and product sizes and efficient algorithms have been implemented for evaluating the performance on the ModCloth dataset.

CCS CONCEPTS

• Information Systems → Collaborative Filtering; Recommender systems;

KEYWORDS

Personalization, Hinge Loss, Latent Factors, Deep Learning, Ordinal loss

1 INTRODUCTION

With the advent of web-based technologies, we have seen a huge increase in businesses trying to attract more consumers to increase their presence in the marketplace. Online shopping is one such area where users could purchase items based on their preference through just a few touches on their smartphones. In the case of garment purchase, the biggest drawback in such a setting is the lack of awareness of fit between product size and customer size. To alleviate this problem, recommender systems have been developed in the past [1-3] to provide size suggestions to users purchasing a particular product based on the feedback of previous purchases. Personalization of the clothing fit is a real problem that has a lot of similarities to the problem proposed in Netflix prize. There are a lot of sources of variance within size of a cloth or a shoe amongst different brands. There are buyers who purchase items for their spouses, children, relatives and even friends. So, there are multiple persona who have purchased products under the same username. These datasets tend to be sparsely populated as almost all users buy very few set of items and most of the users don't buy more than 5 items in the entire dataset. There is a lot of variance between products of two different brands at the same size and this creates a lot of confusion to the user whether the product would be a good fit or not. To aid the users in making better purchases and reduce the number of returned goods, this work aims to predict the size of the customer and the size of the product that would best fit the customer. Section 1 talks about the dataset and the exploratory analysis performed on it. Section 2 talks about the prediction task that we have attempted to study using the ModCloth dataset. Section 3 describes the models attempted, loss functions and the algorithms developed to test the model. Section 4 discusses the existing literature that's attempted to solve this problem and compares the strengths and weaknesses of our model with those models. Section 5 discusses the results of our model and concludes the work.

1.1 DATASET

The dataset that has been used for this project has been extracted from https://www.modcloth.com/. It's a popular website for women's clothing. The lineups vary from tops, dresses, skirts, bottoms, lingerie, etc. This business sells their product online and collect

feedback from the users. Their feedback usually includes the metrics mentioned in Table 1. Table 1 describes the statistical measures of the main features of the dataset both pre and post processing of the dataset. The original dataset has 82790 samples with some of the features such as waist, bust size and shoe_size being sparsely populated. Table 2 captures the basic statistical measures such as mean, standard deviation, inter quantile range points, minimum and maximum values of the features. Figure 1 plots the boxplots used to capture the outliers in different features of the input data. Some of the observations include an outlier in the shoe size with one of the data samples having a value of 38 whereas the other values were in within [5, 13]. As bust, waist, shoe_size and shoe width were sparsely populated, the columns were removed from the dataset.

1.2 FEATURE SELECTION

In addition to removal of bust, waist, shoe_size and shoe width features, we have tried to reduce the number of features by identifying a significant correlation between hips and waist values. To avoid redundancy, waist feature has been removed from the dataset. Height was a string object in the original dataset, it was converted to metric scale post-processing the data. Table 1 also provides details of the 12 most prominent features namely height, length, fit, bra_size, hips, waist, bust, cup size, length, quality, user_id, shoe_size and shoe width. The other columns in the original dataset include review_summary and review_text. Figure 2 plots the scatter plot between bra size and size of the product. This was done to check for correlation between bra size and size of the product. Figure 2 informs us that there's no such correlation that exists between bra size and size of the product. Figure 3 describes categories of products in the dataset. Similarly, the labels for fit has been described in figure 4. From the figure, it could be noticed that there's data imbalance present within the dataset. The "fit" labels constitute to almost 68% of the total data samples, whereas the "small" and "large" data labels together contribute to almost 32% of the samples. Scatter plots have been described in figure 5 to measure the correlation between height and shoe_size. Boxplots showed outliers for the bra_size feature, but when the scatter plot for size vs bra_size is plot (shown in figure 6), we could see that neither could be purged from the dataset. After processing the dataset, the following features were used to build the baseline mode – fit, length, quality, size, item_id, user_id, height, bra_size, hips and category. Table 1 captures the number of samples and datatypes for each feature both pre and post processing of the dataset. As our modeling approaches don't use the review text or the summary, those features are also purged from the dataset. Before we proceed with processing the data, we had to convert "NaN" values with "Unknown" for category type features, 0 for int or float type features.

Column	**Prior** Dtype	**Prior** Non-null Count	**Post** Dtype	**Post** Non-null Count
item_id	int64	82790	int64	81594
waist	float64	2882	NA	NA
size	int64	82790	int64	81594
hips	float64	56064	category	81594
bra_size	float64	76772	category	81594
quality	float64	82722	float64	81594
height	string	81683	loat64	81594
length	category	82755	category	81594
bust	category	11854	NA	NA
user_id	int64	82790	int64	81594
shoe_size	float64	27915	NA	NA
shoe width	category	18607	NA	NA

Table 1: Data processing results. Prior Non-null count refers to number of non-null samples in the original dataset, Post Non-null count refers to the number of non-null samples in the post-processed dataset. The columns mentioned with entries as *NA* have been purged from the dataset either due to the lack of samples or irrelevance to the prediction task.

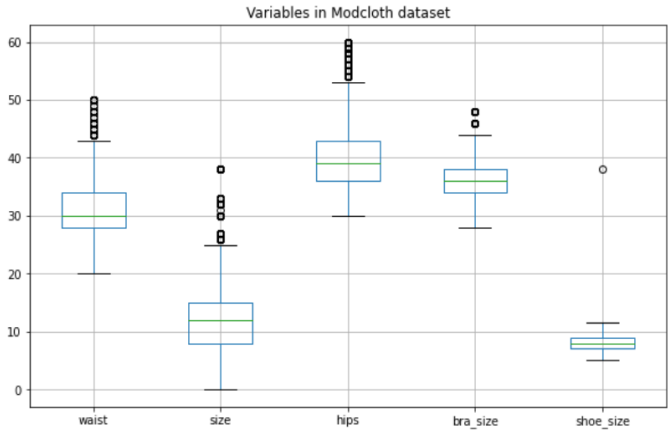

Figure 1: Boxplots of features in the original dataset

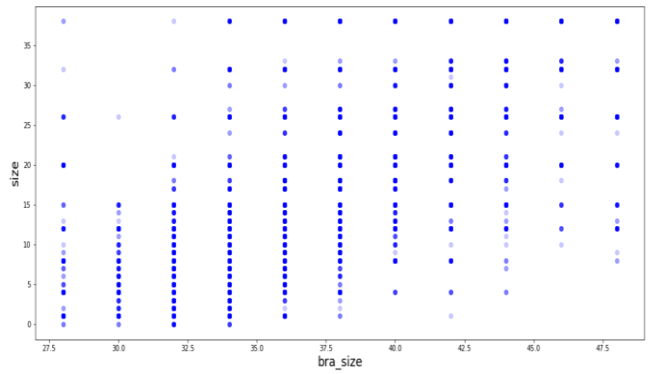

Figure 2: Scatter plot between size of product and bra_size of customer

METRIC	WAIST	PRODUCT SIZE	QUALITY
MEAN	31.39	12.66	3.949
STD	5.3028	8.2719	0.9927
MIN	20	0	1
25%	28	8	3
50%	30	12	4
75%	34	15	5
MAX	50	38	5

Table 2(a): Statistics of waist, size and quality

Amongst these values, waist, cup size and shoe size columns were dropped. Samples where there was no value for length or quality were also dropped. The second model extracted true sizes of the product and customer using a single variable latent factor model which was later passed through a logistic regression classifier to estimate the performance on the test set.

METRIC	HIPS	BRA SIZE	SHOE SIZE
MEAN	40.358501	35.972125	8.1458
STD	5.827166	3.224907	1.3361
MIN	30	28	5
25%	36	34	7
50%	39	36	8
75%	43	38	9
MAX	60	48	38

Table 2(b): Statistics for hips, bra and shoe size

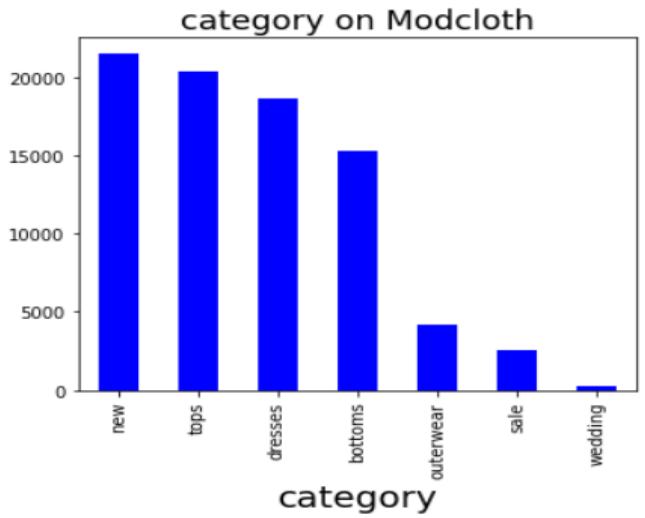

Figure 3: Category bar plot for the dataset

The third model that was developed had used product features such as height, length, size to provide more information to the classifier which the second model couldn't capture. The fourth model used K latent factors behind each customer and product's size. It also tried to bring samples of same class closer to each other using metric learning. The latent features learnt were used to pass through a logistic regression classifier and the performance of the same has been reported in the results section.

The data split used for train, validation and test sets were 80%, 10% and 10% respectively. The model's validity is evaluated through a 3-fold cross validation of a randomly sampled dataset as the test and validation sets have only 10% of all samples. To

reduce the bias on the prediction task, the hyperparameter for the logistic-regression based classifier was set as C = 0.01 and it was achieved after performing a sweep through values for C from [10^{-3}, 10^{3}] on the validation set. All the classifiers in this work have used a logistic loss and L2-norm based regularization. The baseline model uses the features obtained from the dataset for classification. But, the second model has adopted a different formulation of the problem for attaining better classification results. As proposed in [1], the second model has used the same hinge loss function as the objective function for gradient descent. For the second approach, the products have been divided into a hierarchy of representations namely, parent product and a child product. The parent product would be represented using the item identifier whereas the child product uses the item identifier and the size of the product. Even though this problem seems like it only needs to implement a classifier to separate the fit classes from each other, it would be better if the model could learn how to cluster the ones belonging to the same class. With that intuition, metric learning as proposed in [3] has been used. To avoid label imbalance issues, a technique similar to the prototyping technique in [2] has been used. The algorithms used to solve the problem have been mentioned in [1]. We use a similar formulation of the problem and the algorithm also is similar in the basic sense.

2 PREDICTION TASK

For this project, we could identify a number of predictive tasks. The goal initially was to identify the issues that we might face with training time, validation framework, dataset sampling and uniformity of evaluation metric across prior work

Some of the predictive tasks that were identified are –

1. Prediction of customer fit based on the natural language understanding of prior customer feedback.
2. Prediction of customer rating based on text related features such as TF-IDF, bigram, unigram frequencies.
3. Prediction of customer fit based on features directly extracted from the dataset.
4. Prediction of best fit based on similarity to other users defined through Jaccard or cosine similarities.

Amongst these prediction tasks, we discarded the ones based on review lengths and review texts primarily due to the presence of good data samples of reviews for evaluation. Almost 10000 (12.5%) of all the samples in the dataset didn't have any review and amongst the ones that had reviews, enough information couldn't be extracted from them.

Thus, we focused on predicting fit based on the different latent factor models we had learnt in class. So, the problem which we wish to predict is as follows –

Given any customer and their corresponding inputs, which size of a particular product would best fit the customer is the task this problem attempts to predict. So, for any user u, given their features in the form ordered tuples of size n (height, category, bra size, hips), which size of a particular parent product p would best fit the user u? As this dataset has three main categories of the product sizes, we would treat this problem as a multi-label classification problem. Essentially, this problem is also similar to the binary classification problem as we need to predict one class and the others would all be treated as negative classes. There are multi-class wrappers that are available on the traditional binary classifiers. We have used one such classifier for this work. We have extracted features and tried to fit an "One vs Rest" version of the Logistic Regression classifier. We have evaluated our work using the average area under the curve

As an attempt to solve this problem, we started with implementing a very basic logistic regression classifier on the (height, category, bra size, hips, size). To evaluate a multi-class classifier, we have used ROC plots and the area under receiver operating characteristic (ROC) curve. For this problem, the structure of the confusion matrix would be as depicted in figure 4.

		ACTUAL		
		SMALL	FIT	LARGE
PREDICTED	SMALL	SS	SF	SL
	FIT	FS	FF	FL
	LARGE	LS	LF	LL

Figure 4: Confusion matrix for fit prediction

The main reason for using area under curve for evaluating the classifier is because the value indicates how effective the classifier is at discriminating the positive samples from the negative ones. In figure 4, the accuracy is defined as mentioned in the following equation

$$Accuracy = \frac{Sum\ of\ squares\ highlighted\ in\ green}{sum\ of\ all\ squares} \quad (1)$$

Area under curve (AUC) measures the probability that the model ranks a random positive example higher than a random negative sample. The value of AUC lies between 0 and 1. It is computed as the area under the ROC curve. The ROC curve plots True Positive Rate vs False Positive Rate. Thus, if the area under the curve is higher, the classifier is better at discriminating the true positive from false positive samples in the dataset.

3 MODELS

The baseline version of the model derives features from the original dataset and attempts to classify based on the "One vs rest" version of the logistic regression classifier. In [1], the authors had used a single variable latent factor model to compute the true sizes of each for each customer and product. The authors in [1] had modeled it on their custom dataset and we have used the same model for modeling the results on the Modcloth dataset. In [1], the authors had adopted a customer, product pair as their input for each review, with the output being the size of the product which would be the best fit for the customer.

Given an item user pair *(i, j),* the intended aim of the model was to produce y_{ij} where each y_{ij} corresponds to one of {"small", "fit", "large"} labels. The human way of segregating "small" from "fit" and "large" labels is through a comparison between the item's size and the buyer's size. This mathematically translates to what's shown in figure 5 and equation 1. As per the figure, we get the output as large when the customer size < product size, we also obtain the output as small when the customer size > product size. If the product size is within a reasonable range of the user's size, it would be considered a fit. Mathematically representing it provides us the following formulation

$$f(s_i, t_j) = \begin{cases} s_i - t_j > b_1, & y_{ij} = large \\ b_1 \geq s_i - t_j \geq b_2, & y_{ij} = fit \\ s_i - t_j < b_2, & y_{ij} = small \end{cases} \quad (2)$$

In the above equation s_i and t_j refer to the actual sizes of the customer *i* and item *j* respectively. The goal of this model is to compute the value of s_i, the true customer size for all customers in the dataset and t_j, the true product size for all child products in the dataset. The thresholds b_1 and b_2 are the separation boundaries for the *fit-large* regions and *small-fit* regions respectively. There's an ordinal relationship between b_1 and b_2. For this model to work, the constraint $b_1 < b_2$ should always be true. To achieve the best parameters, we have used the same loss function as the authors of [1] had used on their privately collected dataset. Equation 3 defines L in terms of another loss function L^{bin} with binary outcomes in {-1, 1}. For L^{bin}, this work uses the hinge loss function defined as follows

$$L^{bin} = L^{hinge}(y, y') = \max\{0, (1 - y) * y'\} \quad (3)$$

y' is the prediction arising from equation 5. After computing the loss, we compute the gradients of *w*, b_1, and b_2 for each user and item in the entire training dataset. As the hinge loss is a piecewise linear function, there are three different derivatives for the loss function in each of the three intervals of the function *f(w)*. We have used batch wise gradient descent algorithm to arrive at the most optimal parameters in *w*, b_1 *and* b_2. The shape of hinge loss is

plotted against true customer size s_i in figure 7. Due to the nature of the hinge loss, we reach a globally optimal value as long as the learning rate isn't too high. To avoid overfitting the model, it was set to the value explained in equation (4)

$$New\ learning\ rate = \frac{current\ learning\ rate}{sqrt(iteration+1)} \quad (4)$$

With the initial value of the learning rate set to 5e-6, we achieved a converging value for the total loss for this model. To evaluate the entire model, we compute the average AUC of each of the three individual classes. This metric has been reported for all the three models for comparison in section 5.

For this model, the prediction is computed as per the following equation.

$$f(w) = w * (s - t) \quad (5)$$

Here, s and t correspond to the true physical sizes for the customer size and product size that has been learnt so far. This is specific to each customer and each product. The loss functions have been modeled as per equations 3 and 6.

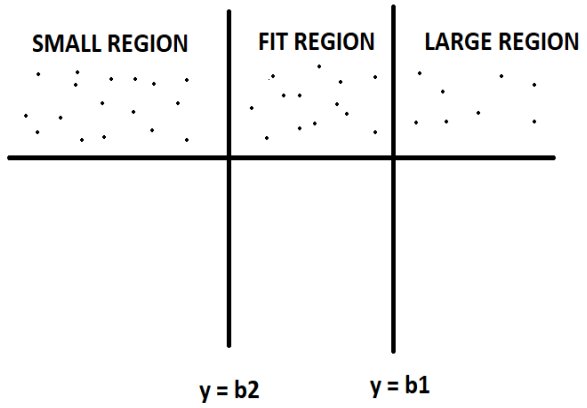

Figure 5: Graphical view of the model

$$L = \begin{cases} L^{bin}(+1, f_w(s_i, t_j) - b_2) & if\ y_{ij} = Small \\ (L^{bin}(-1, f_w(s_i, t_j) - b_2) + \\ L^{bin}(+1, f_w(s_i, t_j) + b1)) & if\ yij = Fit \\ L^{bin}(-1, f_w(s_i, t_j) - b1) & if\ y_{ij} = Large \end{cases} \quad (6)$$

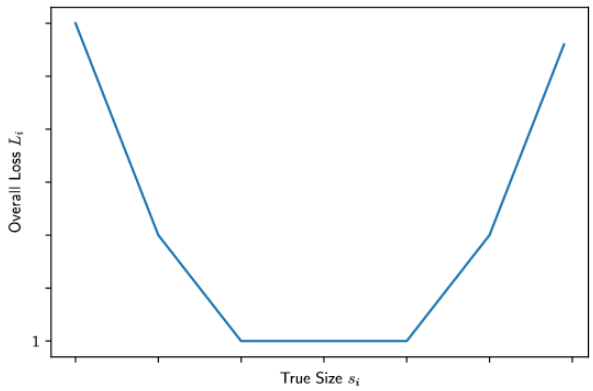

Figure 6: Overall hinge loss function vs True size

Figure 6 captures the piecewise linearity and convex nature of the overall loss. So, with a random initialization of w = 1, b_1 = -1 and b_2 = +1, we run the batch gradient descent algorithm to find the optimal values of w, b_1 and b_2 to yield features for a more accurate classifier.

Algorithm 1: Estimate true customer and product sizes

basicLatentFactorModel(C, P, D, {c_j})
1 For each product j, $t_j = c_j$
2 w = 1; b_1 = -1; b_2 = 1;
3 while (!converged) and (numIterations < maxIter) do
4 for each customer i, s_i = arg min $_{si}$ ($L \mid \{t_j\}, w, b_1, b_2$)
5 for each product j, t_j = arg min $_{tj}$ ($L \mid \{s_i\}, w, b_1, b_2$)
6 w, b_1, b_2 = arg min $_{w, b1, b2}$ ($L \mid \{s_i\}, \{t_j\}$)
7 end while
8 return {{s_i}, {t_j}, w, b_1, b_2}

Algorithm mentioned above has enabled us to obtain the values for w, b_1, b_2, true customer size and true product sizes which minimize the loss function mentioned in equation (6). Just like the model we studied in our class, a single variable latent factor model for each customer and product's true size doesn't consider the interactions between the product and user while computing the prediction. Instead, when we tried to implement the following model, we achieved a better performance than the model

proposed as the simple single variable latent factor model

$$f(w) = w * (s - t) + w_x * x_{ij} \quad (7)$$

This model uses a linear combination of the difference in sizes and also utilizes the features of the product x_{ij}. The loss function for hinge loss has been extended to learn the weight parameters w_x. Even this model doesn't perform significantly better than the other model. The main reason for the failure of this model is that multiple personas belonging to the same userID hasn't yet been modeled in any of the two variants of the model. If we observe the dataset, there are only a few unique users but sizes for each user varies significantly. Thus, the feature of not modeling different personas has impacted the performance of this model.

The third model that we have implemented is identical to the one that was developed by Rishabh Misra et. al in [2]. This was the model that we wanted to replicate for Modcloth dataset, but during the course of implementation for this project, we realized that the model proposed in [1] isn't better for the ModCloth dataset even though [2] had reported results only on another dataset.

Instead of using a single variable as a representation of the latent factor model, they have introduced K variables to model each customer's true size and each parent product's true size. They have also modeled the interactions between users and items that they purchase to extract better features. To overcome the failures of the second model, they have also used a selection criterion to model different persona feedback. To avoid overfitting, they use a very small and adaptive learning rate just like the one that was used in model 2. Authors in [2] have used metric learning very effectively. The advantage of using metric learning based feedback is that it moves the samples which are labeled having the same fit feedback closer and the ones which are labeled different to be moved farther apart from each other. Along with the hinge loss, we would be adding metric learning loss to the overall loss function. Metric learning loss could be explained through equation.

$$f_w(s,t) = < w, (\alpha \ddagger \beta_u \ddagger \beta_{pp} \ddagger (\gamma_u \odot \gamma_{pp}) > \quad (8)$$

Equation 8 describes the traditional latent factor model that we studied in class. We have also implemented that along with the addition of metric learning loss function and thus, for the fourth model, the overall loss function looks as mentioned in equation 9.

$$L_{overall} = L_{1-LV-LR} + L_{LMNN} \quad (9)$$

Where $L_{1\text{-}LV\text{-}LR}$ corresponds to the loss function used in equation 6. L_{LMNN} corresponds to the loss explained by the [8]

Thus, we have developed three different models as mentioned below and 4 variants for evaluating the task of predicting size recommendations for the users for a particular product.

3.1 Comparison Methods

- **Baseline**: Every sample is converted to a feature vector of (height (in cms),
- **1-LV-LR** : Single variable latent factor model for each customer and product. Uses $f_w(s,t) = w * (s_i - t_j)$ for predicting the fit. We use gradient descent to estimate the best values for decision boundaries (b_1, b_2) and w.
- **1-LV-PF-LR :** Single variable latent factor model for each customer and product. Uses $f_w(s,t) = w * (s_i - t_j) + w_x * x_{ij}$ to model prediction, x_{ij} are the product features. Same loss functions and optimizers as **1-LV-LR** has been used.
- **K-LV-ML-LR :** K dimensional vectors are used to model the true interaction between the customer *i* and the product *j*. Metric learning loss has been incorporated on top of hinge loss to produce the final classification results.

3.2 Implementation Challenges

As we were implementing the model, we faced issues related to the choices of hyperparameters for each of the models. To reduce bias, we had decided to train

the model using 3-fold cross validation. Initially we started with a grid search for hyperparameters, but as we moved to implementing **1-LV-PF-LR,** we realized it won't be feasible to complete the project in time. Thus, we shifted to random search across all the parameters. We used sensitivity analysis and go with the winner approach to select the hyperparameters. We ran into problems of overfitting on all the models**.** Because of the convex nature of the loss function for 1-LV-LR and 1-LV-PF-LR, we fixed the overfitting problem by simply changing the learning rates. But, to fix the problems related to K-LV-ML-LR, we had to struggle a bit and eventually used random search to finalize the parameters for the learning rates. The hyperparameter values for each of the models have been reported below –

Model	Initial Learning rate	Regularization Coefficient
Baseline	NA	C = 0.01
1-LV-LR	$5 * 10^{-6}$	w = 1, User and product values were all zeros
1-LV-PF-LR	$5 * 10^{-6}$	w = 1, w_x = 1, User and product values were all zeros
K-LV-LR	$5 * 10^{-6}$	Random initialization for K factors; w = 1; b_1 = 5; b_2 = -5

Table 3: Hyperparameter values for the models

This work compares the average AUC for each of the models in the results section as the evaluation criterion for this model. To compare with more recent models, we have just extracted values from those papers which had reported results on the ModCloth dataset to compare the performance of our model with the more recent state of the art models.

4 LITERATURE SURVEY

This dataset that has been studied in our work has been studied by other authors in [6-7]. The authors in [6] and [7] for addressing market bias and have used a version with images as part of the dataset. The papers relevant to the prediction task we are attempting to solve have used the same dataset without images [1-5]. Performance of [1] and [2] have been tested by deploying it in real world and results of A/B tests have been reported. There are other similar datasets such as RentTheRunway which could be obtained from https://www.renttherunway.com/. We haven't used that dataset for our evaluation.

In [1], the authors have proposed a single variant latent factor model that enables one to compute the true customer and product sizes based on the values estimated using gradient descent. They had tested their model on their custom dataset and reported results on the same. We have implemented that model, tested its performance, suggested a variant and reported the performance on the variant of that model.

In [2], Rishabh et al have used metric learning, handling personas on top of the work done in [1] to build a more accurate model for cloth fit prediction on the ModCloth dataset. This is the model that we have used for comparison as state of the art.

In [3], Sembium et al have proposed a latent factor model to infer true sizes of the customer and products using a Bayesian model using Bayesian logit and probit regression models with ordinal categories to model the size fit data. In this model, the authors have developed learning schemes for learning the posterior distributions of the true customer and product sizes. They have also leveraged them to estimate the fit size probability.

In [4] and [5], the authors have adopted a deep-learning based approach to solve the problem of fit prediction on Modcloth dataset. In this work, the authors have proposed a size and fit network (SFNet). They have approached the problem via likelihood maximization and to that end, they have formulated and optimized the parameters of an instance of a probabilistic model that maximizes the probability of outcomes of observed customer product interactions in the training data. In [5], the author has attempted to achieve semantic understanding of the user's feedback and has used different natural language processing (NLP) techniques such as ULMFit, Bag of words, and BERT to test the performance of his model. The metric used for reporting the results is Micro F1 score on the test set for each of the models that the author had developed.

5 RESULTS

For this work, we have compared the average AUC score with state-of-the-art models mentioned in [2 – 5]. The average AUC scores for each of the three classes {"small", "fit", "large"} have been mentioned in table 4. The baseline model provides an AUC of 0.5806 at a regularization parameter = 0.01. This is primarily because of the fact that the features directly obtained from the dataset don't always mean that it's the most accurate size for the customer. This is exactly why the baseline model didn't perform as well as the 1-LV-LR model. Here, we estimate the true customer size for every user and true product size for every product. The sizes for the child products don't always match the sizes of the parent products as they don't always mean the same.

Model	Average AUC Score
Baseline	0.5806
1-LV-LR	0.6103
1-LV-PF-LR	0.6164
K-LF-ML-LR	0.6518
SFNet	0.693

Table 4: Average AUC score for different models

Table 4 describes the results that we have achieved on the various models that we implemented for this project. The best model as per the results that we have obtained is the model proposed in [2]. The main reason for 1-LV-LR and 1-LV-PF-LR not being able to perform better than K-LF-ML-LR are that they haven't been able to model multiple persona corresponding to each user and that they haven't been able to model interactions between each pair of user and products. In other words, the gamma terms in the following equation have been modeled for K-LF-ML-LR. There have been more recent works using natural language processing methods and advances in deep learning that have attempted to solve this problem. Their results have also been mentioned in table 4. The values reported for SFNet and UMFit haven't been tested. They have been reported from [4] and [5] for comparison purposes.

6 CONCLUSION

Cloth fit recommendation is a hard problem primarily due to the sparsity of data, size variance amongst brands, variance of persona amongst users with the same user_id and the lack of a standardized size that fits everyone. In this work, we attempted to implement a basic model that was in sync with what was taught in the class. We also referred to more advanced work implemented on other datasets and tried to implement (1-LV-LR and 1-LV-PF-LR) them for ModCloth dataset. We have used average AUC score as the metric of evaluation due to its scale and threshold invariability properties. We have also referred to the state-of-the-art models and understood how to model the second order effects particular to this problem such as label imbalance and multiple persona within the same user_id to achieve a better AUC score. To tackle the problems of overfitting, we have adopted the criterion of early stopping for K-LF-ML-LR model, fine tuning of learning rates for 1-LV-LR and 1-LV-PF-LR models. We have referred to more recent works and have reported the values of average AUC score for the deep learning based SFNet architecture. There's been even more progress by [5] using ULMFit and BERT models through which they have improved the state-of-the-art performance in cloth fit recommendation.

REFERENCES

[1] V. Sembium, R. Rastogi, A. Saroop and S. Merugu, 2017. Recommending Product Sizes to Customers. *RecSys' 17*, August 27-31 2017

[2] R. Misra, M. Wan, and J. McAuley 2018. Decomposing fit semantics for product size recommendation in metric spaces. In Proceedings of the 12th ACM Conference on Recommender Systems. ACM, 422–426.

[3] V. Sembium, R. Rastogi, A. Saroop and S. Merugu. Bayesian models for product size recommendations. Proceedings of the 25th International Conference on World Wide Web.

[4] Abdul-Saboor Sheikh, Romain Guigoures, Evgenii Koriagin, Yuen King Ho, Reza Shirvany, Roland Vollgraf, and Urs Bergmann. A Deep Learning System for Predicting Size and Fit in Fashion E-Commerce. In Thirteenth ACM Conference on Recommender Systems (RecSys '19), September16–20, 2019, Copenhagen, Denmark

[5] S Baier, Analyzing Customer Feedback for

Product Fit Prediction. http://arxiv.org/abs/1908.10896

[6] M Kiapour, X. Han, S. Labeznik, A. C. Berg, T.L. Berg. Where to Buy It: Matching Street Clothing Photos in Online Shops. International Conference on Computer Vision, 2015, Santiago, Chile.

[7] Mengting Wan, Jianmo Ni, Rishabh Misra, and Julian McAuley. 2020. Addressing Marketing Bias in Product Recommendations. In the Thirteenth ACM International Conference on Web Search and Data Mining (WSDM '20), February 3–7, 2020, Houston, TX, US

[8] Kilian Q Weinberger and Lawrence K Saul. 2009. Distance metric learning for large margin nearest neighbour classification. JMLR (2009).