Comparative Analysis of Regression and Classification Approaches for Predicting Discrete Ratings

Abstract*—Predicting user ratings is vital for enhancing user experience in the context of recommendation systems. This study uses the Rent the Runway dataset to evaluate two distinct approaches for user rating prediction: Regression and Classification. Unlike traditional rating prediction tasks where ratings are continuous, this dataset presents a unique challenge with ratings taking on discrete values of 2, 4, 6, 8, and 10. Given the discrete nature of the ratings in the dataset, the primary goal of this research is to determine the most effective technique for rating prediction. We do this by comparing the performance of both regression and classification models using a variety of evaluation metrics. The analysis will include a comprehensive evaluation of both methods, considering measures like accuracy and F1 score for classification models, and Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and Mean Absolute Error (MAE) for regression models. We will also investigate how model selection affects interpretability and effectiveness when it comes to rating prediction. The results of this investigation will clarify whether regression and classification methods are appropriate for forecasting discrete ratings in situations where conventional continuous ratings are not relevant. The approach that has been considered superior will give users major insights into which method to pick for similar rating prediction problems involving discrete values.

I. Introduction

In the ever-evolving landscape of e-commerce and online retail, understanding customer preferences and providing personalized experiences is paramount. A critical aspect of this understanding is the ability to predict and anticipate user satisfaction with specific products. Predicting user ratings is one of the recommendation approaches, the significance of which lies in its potential to enhance user experience, guide purchasing decisions, and optimize inventory management for e-commerce platforms. By accurately forecasting how users perceive and rate different items, businesses can tailor their product recommendations, recommending customers with options that align with their preferences. Moreover, such predictions can aid in identifying areas for improvement in product design, quality, and overall customer satisfaction.

In this context, our task revolves around predicting user ratings for clothing items using user and item features, defining the problem into a classification task with five distinct rating categories. For our task, we pre-processed and extracted new features from the dataset available at [1] and implemented multiple models, including Random Forest, FSM, Shallow Neural Network, and Collaborative Filtering using Cosine similarity. As the dataset consists of features for users and items, we also implemented regression-based approaches and performed a comparative study of the model performances.

II. Literature Review

In the field of recommendation systems, the task of rating prediction has garnered substantial attention from researchers and practitioners alike. A plethora of studies have delved into the complexities of understanding user preferences and behaviors in the context of e-commerce platforms. For our rating prediction task the Rent the Runway dataset [1], was used. The study in [1] focuses on developing a predictive framework for the product size recommendation problem by factorizing the semantics of customers' fit feedback and employing a metric learning technique to resolve label imbalance issues. In addition to this, there are a few studies [3],[5] carried out on the dataset. [3] utilizes a deep learning-based content-collaborative methodology to address the sparsity challenge in customer-article orders and optimizes global parameters to learn population-level abstractions and employs customer and article-specific embedding's to provide personalized size and fit recommendations. The proposed work differs from these, as it focuses on predicting the item ratings by taking into consideration the user-item interactions and combining them with the polarity scores of review text. [5] uses a similar approach which focuses on predicting item ratings and creating a content-based item recommendation engine. However, [5] varies from the proposed work in which only the clothing category: dress was considered and the Singular Value Decomposition model was implemented for rating prediction. In contrast, the proposed work makes use of the entire dataset with feature engineering. Another study [4], proposes a framework for review rating prediction that integrates both content-based methods focusing on textual content and collaborative filtering methods leveraging reviewer-item rating matrix experimented on movie review dataset.

III. DATASET

To investigate the best technique for rating prediction we have considered the Rent the Runway dataset [1] which contains features regarding the clothing fit of female apparel. The dataset has 192462 interactions where 105508 users, review 5850 items containing 68 different categories of apparel. The dataset can be broadly categorized into two subsets where features like user_id, age, height, weight, review text, bust size, body type, rented for, and review summary correspond to user attributes while item_id, fit, rating and size and review date correspond to item attributes.

As a standard procedure entire data is split into train, validation, and test subsets which are in the ratio of 8:1:1. The next step is preprocessing which involves

1) Imputation of missing values - The ideal procedure is to remove all the observations that contain missing values if they account for less than 5% of the entire data, while if they exceed the 5% threshold impute the values with the corresponding median or mode of the data.

As many of the features are categorical variables that exceed the 5% threshold, the missing values of these features are imputed with the mode of the specific feature, and the 'age' is imputed with the median value.

2) Class imbalance – The dataset is skewed towards the higher end of the ratings i.e. 10 and there are a lot more instances of 10 than any other rating. If it were a classification approach this has to be tackled by resampling the data to account for class imbalance using weighted sampling or any other approach of either under or over-sampling. However, as we are trying to compare different techniques that best suit the need, making it a class-balanced dataset with sampling will affect the regression model.

IV. EXPLORATORY DATA ANALYSIS

A. Analysis of features using the Chi-Squared Test

The chi-squared test is employed to evaluate the independence of a categorical characteristic and establish its relevance with the target for classification tasks. It aids in determining the relationship between two categorical variables. In addition to that, it allows us to rank the features in terms of their relevance. Analysis showed that the 'fit' feature was more relevant as it had the highest score shown in Fig 1.

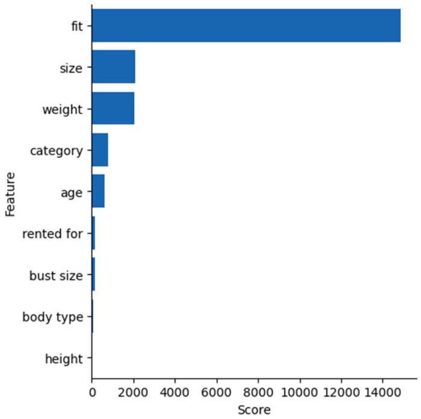

Fig. 1. Feature Relevance using Chi-Squared Test

B. Distribution of Fit Feature

The distribution of the fit categories is shown in Fig 2. It can be observed that the number of clothing items in the category fit is approximately 5 times the other two categories small and large.

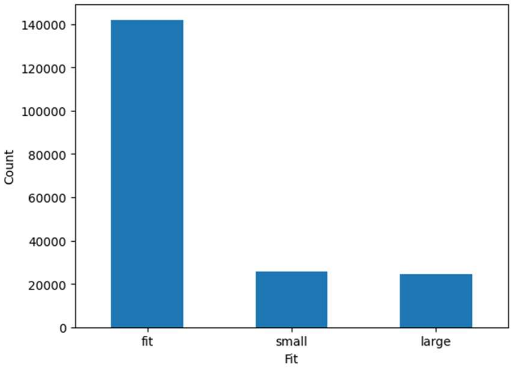

Fig. 2. Fit Feature Category Counts

C. Distribution of Ratings

The distribution of ratings is shown in Fig 3. It can be observed that it is skewed towards the rating of 10. It is a clear indication of class imbalance (when classification task is considered).

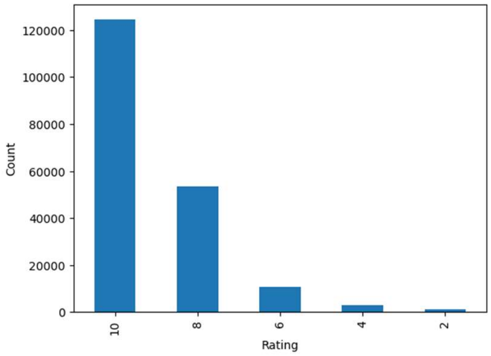

Fig. 3. Ratings Category Counts

D. Analysis of Features against Rating

1. Fit

The distribution nature of ratings across the fit categories in Fig 4 looks almost identical. In other words, irrespective of the fit category, the user tends to give low to high ratings.

However, it is also observed that the user tends to give higher ratings if the item is a perfect fit as the absolute count of ratings '8' and '10' are higher for the 'fit' category. Hence, we infer that the feature fit would play a useful role in rating prediction. But, in practice, many instances determine the rating like the quality of the item, appearance, etc. which may have led the user to give lower ratings even to the perfectly fitting items.

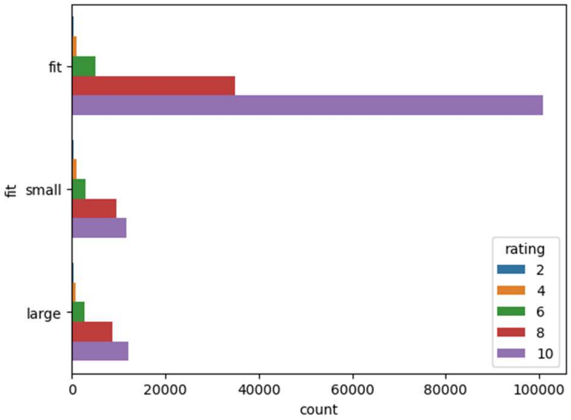

Fig. 4. Fit vs Rating

2. Age

Inspection of the scatter plot in Fig 5 reveals that rating is spread across all the ages and much concentrated between ages 20 and 60. This conveys that age may not be a valuable feature on which rating depends.

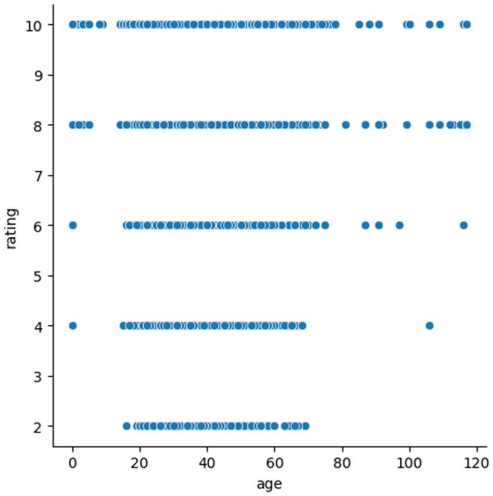

Fig. 5. Age vs Rating

3. Rented for

Fig 6 looks identical to Fig 4 which reveals no specific inference on how the feature 'rented for' is affecting rating. Another observation that can be made from this plot is that people who rent clothes for 'weddings' are higher in number and tend to give higher ratings followed by 'formal affair' and 'party'.

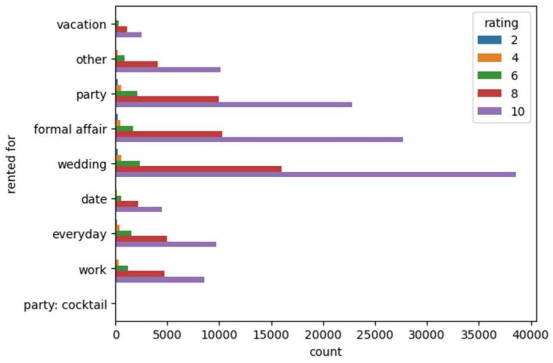

Fig. 6. Rented for vs Rating

4. *Category*

It was observed that there are very few instances for most of the clothing categories making it difficult to infer a relationship with rating across the categories. This led us to not use it as a feature for the predictive task.

5. *Review Length*

The plot of average review length against ratings in Fig 7 depicts an interesting behavior where the average review length increases with an increase in the ratings up to 8 and then decreases for the ratings of 10. This can be associated with the fact that people tend to write less descriptive reviews and make use of rare words for describing such as: marvelous dress.

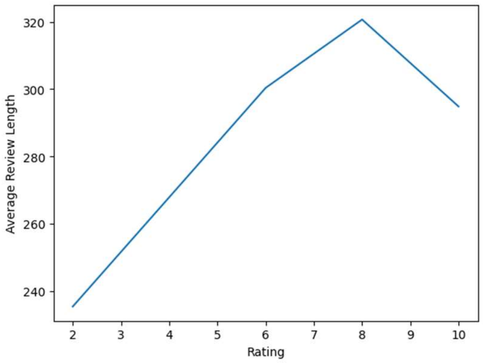

Fig. 7. Review Length vs Rating

V. FEATURE ENGINEERING

Feature engineering is a crucial step in building effective recommendation systems, and it involves transforming raw data into features that enhance the performance of machine learning models. In this context, it is useful or appropriate to extract features such as:

A. *Body Mass Index (BMI)*

Body Mass Index (BMI) is used instead of raw height and weight as features, calculating BMI allows the model to capture more meaningful information about the user reducing the sparsity from one hot encoding of weight and height.

$$BMI = 703 * \frac{weight}{height^2}$$

where, weight in ‘lbs. and height in ‘inches

B. *Review Text Polarity Score*

Reviews provide valuable information about user preferences and opinions. However, raw text data needs to be converted into a numerical format for machine learning algorithms to process. Sentiment analysis or review polarity scoring is a form of feature engineering that translates textual reviews into a numerical value representing the sentiment (positive, negative, or neutral). This feature helps the recommendation system understand user sentiments towards products, enabling it to recommend items that align with user preferences. The Sentiment analyzer from ‘nltk’ is used to perform this task and the compound score of the polarity is used as the review text polarity score.

C. *Temporal Information (Year and Month) from the review date*

User preferences and trends can change over time. Incorporating temporal information allows the model to capture seasonality, trends, and evolving user preferences. Extracting the year and month from the review date enables the recommendation system to consider the temporal aspect of user behavior.

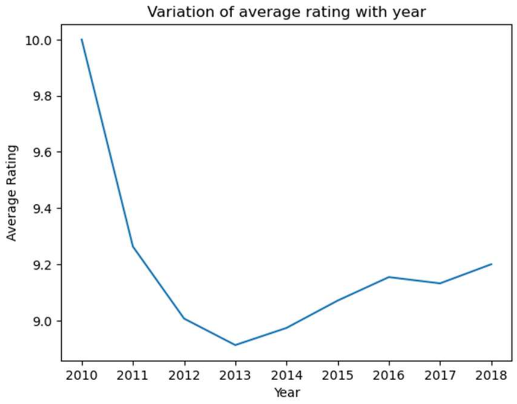

Fig. 8. Average Rating vs Year

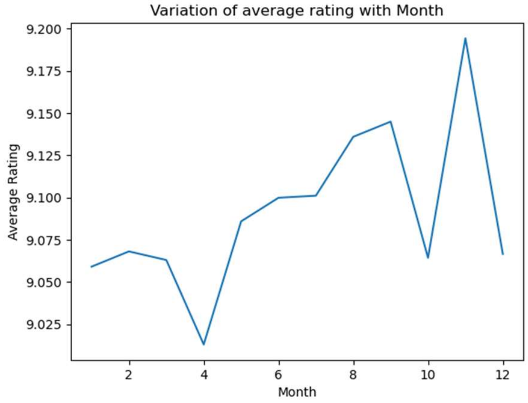

Fig. 9. Average Rating vs Month

Decomposing the review date into year and month helps us to understand how the rating varies over a period. From the above graphs, it looks like the average rating has a steep descent between 2010 and 2013 and gradually starts to pick up from there. In addition to that people tend to rate higher during October and November than any other months.

After pre-processing and extraction of a few user features, we finally end up with the required features which we will be using to evaluate our model. Certain features like ‘review text’ , and ‘review summary’ are dropped as they are redundant when the polarity of the sentence is captured. Also ‘review date’ is dropped after extracting year and month. The final feature set which is used across the models for comparison is { 'fit', 'user_id', 'bust size', 'item_id', 'weight', 'rating', 'rented for', 'body type', 'category', 'height', 'size', 'age', 'review_polarity_score', 'BMI', 'year', month' }

It is to be noted that BMI is added to the feature space in addition to weight and height so that certain models that require sparse representation can use weight, and height and drop BMI further and vice versa.

VI. EXPERIMENTAL EVALUATION

A. Models and Methodology

1. Heuristic Cosine Similarity (Baseline)

A heuristic approach based on Cosine Similarity is considered which uses features extracted from the data available to build a feature vector that cosine similarity uses to generate the similarity score. Apart from the mean user and item rating, this specific model is designed to deal with new observations and users. The model deals with the problem of cold start (for instance a new user) by finding the users who have a similar BMI to the given user under consideration and comparing the 'fit' attribute of those users, which means the comparison of how other users with similar body aspects find the product deemed appropriate fit for them. Similarity scores of these extracted users are weighted with the difference of the rating of individual users with an average rating so that users with rigid rating behavior don't get poor predictions. The BMI for comparison is a choice that depends on the dataset, as the dataset is of people purchasing or renting clothes, it is better to consider people with similar sizes as similar users. The model also accounts for new user or item features in the test set by comparing the new attribute with the closest attribute that the model has seen thereby accurate representation of the user is obtained. For instance, if a new 'bust size' attribute of '42i' is present in the test set, then we impute this value with the most similar size to '42i' which may be '42a' from the training set instead of imputing with median or mode values of the data. This choice of imputation allows for capturing better information about the user without relying on the statistics of the dataset. The heuristic model is designed in a way keeping in mind the ratings ideally fall between 1 and 10 (aka Regression Problem).

$$\text{Cosine Similarity} = \frac{A.B}{||A||||B||}$$

where || || represents the magnitude of vector

To make a common ground for evaluation of choice of statistical technique for the given problem at hand, in the heuristic approach we have employed round the predicted ratings to its nearest even ratings so that it becomes compatible with the true ratings of the test set.

2. Factorization Machines

Factorization machines (FM) are novel machine learning [2] models designed for handling high-dimensional sparse data, making them particularly useful in collaborative filtering and recommendation systems. It is an extension to linear models which capture the interactions between features effectively. The general formula of second-order factorization machines is given by:

$$y^{\wedge}(x) = w_0 + \sum_{i=1}^{n} w_i\, x_i + \sum_{i=1}^{n} \sum_{j=1+1}^{n} < v_i, v_j > x_i\, x_j$$

where,

• w_0 is the global bias, and w_i represents the weights for individual features.

• v_i are the latent vectors (or factors) associated with feature x_i.

• $\langle v_i, v_j \rangle$ denotes the inner product between the latent vectors v^i and v_j capturing interactions between features x^i and x_j.

As FM utilizes the sparse representation of the data it is necessary for the data type of the column to be either string or object, which is then used by the module DictVectorizer to convert into sparse representation. The sparse representation of the features is the key in FM, through investigation it is observed that any numeric values present in the sparse matrix negatively affect the MSE, meaning a numeric value of range 1-100 increases the MSE when it is treated as an integer but MSE goes down slightly when it is treated as a one-hot representation. The higher the range of integers or floats in the sparse matrix, the higher the deviation of prediction from the actual value. FM produce better results when every column of the sparse matrix is either 0 or 1.

3. Shallow Neural Network

We use a multi-dense layer model that can calibrate the interactions between different features. The input layer i.e the feature layer of size 4 is mapped to a hidden layer of size 100 which is then mapped to output. We employed 'relu' activation for this purpose. – Strength: It learns non-linear models well and the complexity of our model is pretty high. It gives good classification accuracy on the training data, test data, valid data. – Weakness: Neural Networks don't work well for class imbalance even though the classification accuracy is high, the F1 and recall scores are not that good this signifies that the Neural Network is good at classifying classes with good amount of Data but fails to classify classes with a small amount of Data.

The last layer of the Neural Network is a softmax layer. This layer decides the class of a sample. It assigns the class which has the highest probability corresponding to it. The optimizer used is Adam and

Tabel 1 - Evaluation metrics of different models (Regression)

Model		Heuristic Cosine Similarity	Factorization Machines	Shallow Neural network	Random Forest	Decision tree	XGBoost
	Metric						
Validation	MSE	2.18	2.225	2.51	2.6	3.7	2.8
	MAE	1.08	1.06	0.85	0.8	1.2	0.9
	RMSE	1.38	1.491	1.58	1.6	1.9	1.7
Test	MSE	2.19	2.66	2.8	2.5	3.2	2.7
	MAE	1.09	0.935	0.9	0.8	1.0	0.8
	RMSE	1.45	1.59	1.68	1.6	1.8	1.6

Tabel 2 - Evaluation metrics of different models (Classification)

Model		Heuristic Cosine Similarity	Factorization Machines	Shallow Neural network	Random Forest	Decision tree	XGBoost
	Metric						
Validation	Accuracy	55.89	60.2	65.45	61.56	57.98	52.89
	F1 Score	0.556	0.582	0.554	0.563	0.516	0.546
Test	Accuracy	56.6	59.9	64.97	60.98	56.98	50.88
	F1 Score	0.560	0.581	0.569	0.523	0.501	0.551

the loss function used is ‘Categorical loss’ the formula for the categorical loss is

$$\text{Cross Entropy} = \sum_{I}^{c} ti\,.\log{(S1)}$$

The formula for cross-entropy is the formula for Categorical loss. The reason behind using a shallow Neural Network is that a Neural Network is good at learning the interaction between the features.

4. Random Forest

A Random Forest is an ensemble technique which uses decision trees as base estimator and introduces further randomization at each node by sampling the set of features instead of all the features in contrast to Decision Trees. It incorporates several techniques like bagging, boosting to improve the information gain at each node based on cross entropy or gini impurity. In addition to that being an ensemble technique it combines the advantage of using multiple decision trees to predict the output. GridSearchCV is used to find the best parameters that best explains the data and the best parameters found is used to train the model.

5. Decision Tree

A Decision Tree is an approach that captures nonlinearity among the features and can scale well. In a decision tree each node is grown recursively until it reaches the leaf node or specific criteria of stopping condition is met. At each node all the input data is sampled and a split point is chosen in chosen in such a way that it maximizes the amount of information gain at each node. The biggest advantage of decision trees over linear algorithms like Linear Regression is the features are not expected to be in same scale. In this we have used Decision Tree Regressor and Classifier from sklearn tuning the hyper parameters along with cross validation. The results are tabulated in Table1

6. XGBoost

In the given context, the adoption of the XGBoost algorithm becomes crucial to address certain challenges and leverage its unique strengths for rating prediction in the imbalanced dataset. XGBoost, an advanced implementation of gradient boosting, stands out as a powerful ensemble learning technique known for its efficiency and predictive accuracy. Unlike decision trees or Random Forests, XGBoost employs a boosting framework, which involves iteratively adding weak learners to the model, each correcting the errors of the previous ones. This iterative refinement process enhances the model's predictive performance and generalization capabilities. One notable advantage of XGBoost over decision trees is its regularization techniques, such as shrinkage and pruning, which mitigate overfitting. This is particularly valuable in scenarios like ours, where imbalanced datasets may lead to suboptimal performance. XGBoost's ability to handle imbalanced data is further strengthened by its incorporation of weighting mechanisms, allowing it to assign higher importance to minority classes during the training process. This is pivotal for our rating prediction task, where balancing the representation of different rating categories is crucial for accurate model training. Despite its strengths, it is essential to acknowledge that XGBoost, like any other algorithm, has limitations. Training time and computational resources required can be higher compared to simpler models, and parameter tuning may be necessary for optimal performance. Nonetheless, the enhanced predictive accuracy and robustness of XGBoost make it a valuable choice for addressing the complexities associated with imbalanced datasets and discrete rating predictions.

The evaluation metrics of all models are tabulated in Table 1, Table 2.

B. Choice of Metrics

MSE is clearly ruled out of the scope because we are predicting the rating and as the rating values predicted by the model are continuous while actual values being discrete even ratings, the square of difference between actual values and predicted ratings is high. The cause for a high MSE value is the distribution of the ratings in the dataset, as the dataset has only discrete even ratings 2,4,6,8 and 10 , any other rating like 9,7,5… predicted by the model negatively affects the MSE and raises a question about is the choice of MSE as an evaluation metric better in this scenario ? Ideally it makes sense to use MAE or RMSE in this case, because the difference squared is amplified much by MSE.

$$\text{MSE} = \frac{\sum(y_i - y_p)^2}{n}$$

1) MAE is more robust metric that is used in case of extremely high dimensional sparse matrices which is more immune to outliers. As outliers are handled by the proposed model choice of MAE is trivial.

$$\text{MAE} = \frac{|(y_i - y_p)|}{n}$$

2) RMSE is better suited in this scenario as it tends to minimize large errors due to rounding off predictions.

$$\text{RMSE} = \sqrt{\frac{\sum(y_i - y_p)^2}{n}}$$

where,
y_i = Actual value,
y_p = Predicted value,
n = Number of observations

3) Accuracy

Accuracy is better suited for classification tasks involving balanced dataset

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

4) F1 Score

F1 score is a more reliable metric in case of imbalanced classification task.

$$F1 = 2 \cdot \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

where,
TP = True Positives,
FP = False Positives
TN = True Negatives,
FN = False Negatives.

VII. Results

Comparison of metrics from Table1 and Table 2 , it can be observed that not all models perform equally well. In regression approach, Factorization Machines tend to perform better than Tree based models and shallow neural network due to the fact that it takes into consideration both the user and item features along with interactions.

In case of classification approach, Shallow Neural Network gives accuracy better than others but fails to get a decent F1 score. This could be attributed to the class imbalance of the data.

VIII. Conclusion

As class imbalance plays a huge role in determining the performance of classification models, F1 Score is considered a better metric to evaluate the performance of a classifier and none of the models are better at predicting the rating accurately. Hence, the task of rating prediction is handled well by a regression-based approach and RMSE is chosen to be the metric that best describes the model performance.

After testing multiple models mentioned in Tables 1,2 we propose Factorization Machines over Baseline (Heuristic Cosine Similarity) though it has higher RMSE. Because the heuristic approach is not realizable as it depends on thresholds limits that are set manually and ignores the latent factors which are essential components for capturing user-item interactions.

IX. Future Scope

Class imbalance is a hurdle for getting a model that can predict the rating for an item. Further investigation is needed to address the issue of class imbalance, to improve model performance. To prevent our current models from overlearning the data that favors a particular class regularization can be added to specific models, for instance, we can use drop-out regularization in our shallow Neural Network this might decrease the accuracy of shallow neural networks favoring the F1 score which is a reliable metric in case of classification.

REFERENCES

[1] Rishabh Misra, Mengting Wan, and Julian 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.

[2] fastFM: A Library for Factorization Machines , Immanuel Bayer arXiv:1505.00641[cs.LG],https://doi.org/10.48550/arXiv.1505.00641

[3] Abdul-Saboor Sheikh, Romain Guigoures, Evgenii Koriagin, Yuen King Ho, Reza Shirvany, Roland Vollgraf, Urs Bergmann, A Deep Learning System for Predicting Size and Fit in Fashion E-Commerce, Thirteenth ACM Conference on Recommender Systems (RecSys '19), September 16--20, 2019, Copenhagen, Denmark, https://paperswithcode.com/paper/a-deep-learning-system-for-predicting-size/review/

[4] Bingkun Wang, Yongfeng Huang, Xing Li, "Combining Review Text Content and Reviewer-Item Rating Matrix to Predict Review Rating", Computational Intelligence and Neuroscience, vol. 2016, Article ID 5968705, 11 pages, 2016. https://doi.org/10.1155/2016/5968705

[5] Medium 2020, Brittany Fowle: Creating a Recommendation Engine using Rent The Runway Data https://medium.com/@befowle/creating-a-recommendation-engine-using-rent-the-runway-data-c4c7867ad9c .