Beer Reviews Regression Problem

Mustafa Omer Mohammed Elamin Elshaigi
Politecnico di Torino
Student id: s289815
s289815@studenti.polito.it

***Abstract*—In this report, we present a possible solution for a Beer Reviews problem utilizing regression techniques. The proposed approach embeds feature selection and feature engineering phases to better characterize the data along with Prepossessing Techniques to handle missing data, we examined the effect of using different imputation techniques in the final regression result. We compared two versions of possible solutions, one in which Text analysis techniques including TF-IDF were adopted and resulted in significant improvement in the final score. The proposed solution compares different regression models using R^2 as an evaluation metric.**

I. Problem overview

The proposed task is a regression problem on a dataset containing 100,000 entries, each of which corresponds to a review expressed by a user on a website for beer benchmarks. Each review is described by a set of attributes. The objective of this study is to predict the overall numerical evaluation $between(0, 5)$ with half scores allowed $\{i.e.1.5, 2.5, ...\}$, assigned by users to beer using the provided information. The dataset is split into two portions:

- $dev.tsv$: development set composed by 70,000 rows reviews for which the assigned overall/review score is known;
- $eval.tsv$: evaluation set composed of 30,000 rows without the target variable.

The attributes available in the dataset are 14; to perform feature engineering and preprocessing on the dependent variables we need to divide them into groups:

- $Numerical\_Features$: contain numerical scores given by the users, which is correlated to the target variable, includes[review_aroma ,review_appearance ,review_palate , review_taste , beer_ABV] .
- $Categorical\_features$: all these features are nominal features where we have no order between the categories of the feature. It includes [beer_name ,beer_style,'user_gender','user_profile_Name'], the first 2 give descriptions about the beer and the second two contain the user's information.
- $Textual\_features$: ['review-text'] contain the main text of the reviews written by users to describe the beer, we will carry sentiment analysis on this feature as in Section II-A
- $Use\_age\_features$: Contain the user age information. 79.08% of the values are missing in these features. Hence, we applied feature transformation combined with imputation techniques and examined the effectiveness of this approach. This group includes[user_ageInSeconds, user_birthdayRaw, user_birthdayUnix]

Variable	Cardinality
beerName	17652
beerstyle	104
gender	2
profilename	12305

TABLE I: Categorical Variables Cardinality

Carrying explanatory analysis, we found some considerations regard the dataset. By studying the Target variable ("review/overall"), we found the distribution is not normally distributed, but highly skewed to the left with ($Median = 4$, $\mu = 3.82$, $\sigma = 0.72$) as shown in fig1, the regression models suffer from this highly imbalanced distribution. Another consideration is the cardinality of the $Categorical\_features$ as shown in Table I could represent a problem if not properly managed, this will be discussed further in Section II-A.

Regards the missing values, some columns contain a non-negligible number of missing values, we adopted several approaches to tackle this issue as will be discussed in the following section . Below is reported the percentage of missing values per variable, Only the variables with missing values are reported.

Variable	Missing values (%)
abv	4.392
text	0.022
age	79.083
user/birthdayRaw	79.083
user/birthdayUnix	79.083
gender	59.689
profilename	0.017

TABLE II: Percentage of missing values per variable

II. Proposed approach

A. Data Preprocessing

We have seen the issues regards the data set. After grouping the features as mentioned in section I, proper preprocessing steps were applied to each group, 2 different approaches were taken we denote as $\{V_1, V_2\}$, The approaches are described as follows :

- **Version-1** (V_1): in this version, for the alcohol per volume feature(ABV), we used simple imputations based on replacing the missing with the mean. For variables related to user birthday and gender, we decided to drop these

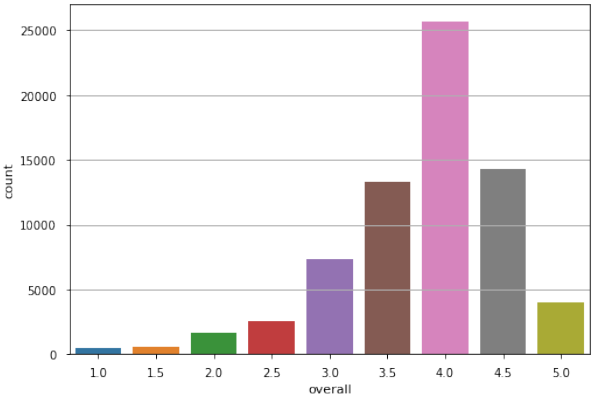

Fig. 1: The Target variable distribution

variables from the analysis since they contain more than 79% missing values. For "profile_name" we replaced the missing values with the most frequent value.

The next phase of preprocessing consists of feature transformation and reduction, we applied a removal of the less commonly found values to categorical features with high cardinality, we defined thresholds in which categories below should be set to "others" value, this also helped in reducing the features before applying the feature transformation step. For feature transformation, the categorical features are "Nominal" with no order between the categories of each feature, therefore, we used One_Hot_Encoding. In this version, we haven't considered the "Review_Text" feature.

- **Version-2** (V_2): In our second proposed solution, a more accurate imputation approach was taken, we first normalized all numerical variables using Min_Max_Scaler, we used the normalized variables to predict missing values in (ABV) variable using k-Nearest Neighbor Imputer (KNN) since it is a distance base classifier that requires normalized features. Before applying imputation, we decided to consider gender and birthday as we assumed these features can be useful, regardless of the significant amount of missing data but birthday can be very crucial in defining the reviewer experience. we first defined a new feature "age" as a numerical feature extracted from the "user_birthday". We defined another numerical feature encoded from "gender".

  to handle categorical features, we defined groups based on the target variable, then we replaced the missing value in each group by the most frequent Profile_Name in that group. Our intuition is replacing the missing values with the most frequent category in the whole feature may cause imbalanced feature distribution, we assumed this process may reduce the imbalanced effect. For all categorical features, we applied the same approach for removing the less commonly found values as in V_1 with different thresholds before applying One_Hot_Encoding.

  We can investigate how to extract relevant information from the Text provided with each review applying sentiment analysis. First, we started by investigating the relation between the length of the review-text and the assigned score, we found the length has almost the same distribution for each assigned score value as shown in Figure2, therefore, we decided to discard this feature. We proposed solution based on Term frequency-inverse document frequency $\{Tf-Idf\}$ representation. After cleaning the text using lemmatization approach, we adopted an iterative process to extract the frequent words that would not be relevant from an informative point of view, we used these to update the stop words. We limited the use of the extracted features to the first N_{freq} words after sorting them in descending order according to the $\{Tf - Idf\}$ values, we examined different values of N_{freq} as discussed in Section II-B. The meaning of adjectives and adverbs often depends on the context and it can have an impact on the feeling expressed by the review. To handle this variability, not only single words were considered but also n_grams (i.e. group of n words) of an adequate dimension. Figure3 shows the most important words extracted by$\{Tf - Idf\}$.

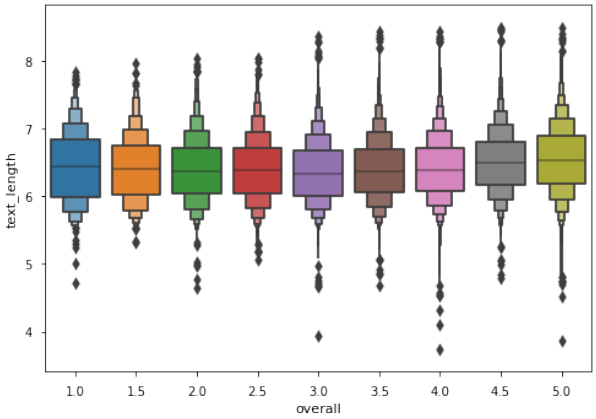

Fig. 2: Text Length Distribution Among Target Variable

Fig. 3: Most relevant words extracted by $Tf - Idf$

B. Model selection

The following algorithms have been tested with (75/25)% of train/validation split on the development set:

- $Random\ forest$: This algorithm belongs to the category of ensemble techniques because it combines multiple decision trees (trained on different portions of the training set and subsets of features) to make predictions. This is generally done to enhance the performance and to avoid overfitting. In regression tasks, it is even less interpretable than it is in classification problems but it can provide an indication of feature importance. We chose to use random forests as they have been shown to perform well on large data sets.
- $LASSO$: the algorithm based on building Linear Model trained with L_1 prior as regularizer. By setting some coefficients to zero, some of the features are completely neglected for the evaluation of output. So Lasso regression not only helps in reducing over-fitting but applies feature selection. The feature selection is controlled by the regularization parameter α.
- $Ridge$: solves a regression model where the loss function is the linear least-squares function and regularization is given by the euclidean norm. Coefficients are shrunk by the same factor (so none are eliminated). L_2 regularization will not result in sparse models. Like is LASSO, the regularization is controlled by λ.

For all regression models, due to the high number of features and hyperparameters, the best-working configuration of hyperparameters has been identified through a Randomize Grid Search, as explained in the following section.

C. Hyperparameters tuning

The set of hyperparameters can be divided in two groups, each corresponding to a different phase:

- $preprocessing$: N_{freq} , n_gram range;
- $final\ predictions$: regression models hyperparameters.

We first examined the first group to find the best values, then we apply Randomize grid search on the second group to maximize the models' performance. we used a Randomized grid search with cross-validation using 10 splits on the various regression models except for the random forest regressor we used 3 folds. Table III shows details of the tuned hyperparameters .

Model	Parameter	Values
Preprocessing	N_{freq}	{ 100 , 250 , 500 , 750 , 1000 }
	n_gram range	{ (1,1) , (1,2) , (1,3) , (2,2) }
LASSO	alpha α	{1e-3 , 1e-4 , 1e-5 , 1e-6 , 1e-7}
	Selection	['random' , 'cyclic']
Ridge	fit_intercept	[True, False]
	alpha α	0.1 $\rightarrow$ 6, step 0.332
	tolerance	[0.001,0.01 , 0.005]
	Solver	{'auto', 'svd', 'cholesky', 'lsqr',.... ... $'sparse_cg', 'sag', 'saga'$}
Random Forest	max_features	{'auto', 'sqrt', 'log2'}
	criterion	{'mse', 'MAE'}
	n_estimators	200 $\rightarrow$ 1200, step 50
	min_samples_split	{20, 15, 10}
	bootstrap	{True, False }

TABLE III: Hyperparameters considered

III. Results

In the first version (V_1) as discussed in section II-A, we discarded the features with high percentage of missing values and the "Review_Text" from the analysis. Therefor, we have no preprocessing parameters to tune in (V_1). The best scores after applying hyper parameter tuning for each regressor is as follows :

- $Random\ forest$: best configuration is {n_estimators: 950, min_samples_split: 20, max_features: sqrt, criterion: mse, bootstrap: True} ($R^2 \approx 0.675$)
- $Ridge$: best configuration is {solver: saga, fit_intercept: True, alpha: 3.0} ($R^2 \approx 0.683$
- $LASSO$: best configuration is {selection: random, alpha: 1e-05} ($R^2 \approx 0.681$)

In (V_2) after preprocessing, we applied different values forN_{freq}, we trained the models with default parameters and reported the results in Table IV, we fixed the best value for N_{freq} as 750 and we tuned the n_grams, we found the best range of (1,2). we used only the training data to fit the models and the validation set to evaluate the performance in this phase. Having set these values, we ran the randomize grid search for the regressors and we identified the following best configurations:

- $Random\ forest$: best configuration is {n_estimators: 1050, min_samples_split: 20, max_features: sqrt, criterion: mse, bootstrap: True} ($R^2 \approx 0.689$)
- $Ridge$: best configuration is {solver: cholesky, fit_intercept: True, alpha: 3.0} ($R^2 \approx 0.699$
- $LASSO$: best configuration is {selection: random, alpha: 1e-05} ($R^2 \approx 0.699$)

We noticed the models' performance enhanced slightly, we can reasoning this due to 2 reasons: first, randomize grid search is faster but less accurate than grid search CV, the second reason is, using a different combination of train_valid in each iteration, leads to better generalization and gives more realistic results. We used the whole development set data to train the regression models with their best configurations. Then they have been applied to the evaluation set to predict the quality scores. The achieved public scores are as follows: 0.692 for the random forest, 0.704 for Ridge, 0.705 for Lasso. The private scores should almost reflect the public ones because there should not be overfitting. The best-reported Score was achieved by Lasso ($R^2 \approx 0.705$) in public.

N	LASSO	Ridge	Random Forest
100	0.6944	0.695	0.6824
250	0.6996	0.7002	0.6873
500	0.7007	0.7012	0.6877
750	**0.7011**	**0.7014**	**0.689**
1000	0.7007	0.7009	0.6887

TABLE IV: N_{freq} effect in model performance R^2

IV. Discussion

Online reviews prediction task is one of the most difficult tasks in the sentiment analysis field, the difficulty of the task

may be indicated by several reasons, including the scores and reviews text were made by normal users with different experiences, tastes, and expressing abilities, this variability makes the task of extracting patterns and useful information from the text very difficult, another consideration is the significant amount of missing values in the data, more specifically one of the most crucial variables (user_birthday) contain over 79% of missing values, this variable could be very useful as it is usually determined the level of experience of the user and could be correlated to the target variable. The proposed approach obtained acceptable results compared to the difficulty of the task. the data preprocessing and feature engineering processes we adopted enhanced the results. Our results are comparable concerning other solutions listed in the leaderboard with a small margin.

To make the model more interpretable, it will be important to reduce the number of features, which is the major aspect to improve. Possible improvements can be probably found in a better fine-tuning of the models and try different models like LinearSVR, RNN based model. These models achieve state of the art in sentiment analysis but it suffers from the lack of interpretability and requires very large datasets. another approach that proven to work very well in sentiment analysis is advanced text mining techniques like pre-trained models (e.g., word embeddings).