Wine Quality Regression Problem

XXXXXX XXXXXX
Politecnico di Torino
Student ID: sXXXXXX
sXXXXXX@studenti.polito.it

***Abstract*—In this report we develop a possible solution for a wine quality problem by means of regression techniques. The proposed approach embeds feature selection and feature engineering phases to better characterize the data. Text analysis techniques were also adopted. The solution compares different regression models achieving promising results.**

I. Problem overview

The proposed task is a regression problem on a dataset containing wine reviews written by experts. Each review is described by a set of attributes. The objective of this study is to predict the overall numerical evaluation assigned by experts to wines using the provided information. The dataset is split into two portions:

- a *development set*, containing 120,744 reviews for which the assigned quality is known;
- a *evaluation set*, containing 30,186 reviews without the target feature.

We will use the development set to build a regression model capable of predicting the quality score that would be assigned to each record of the evaluation set.

The size of the evaluation set is 25% with respect to the development set, which is a correct proportion to train a general model. There are 8 available features and they are all categorical with the exception of the target measure. There are two group of attributes: the first one includes geographical data (e.g., country, province, winery) while the second one contains more "commercial" information (designation, variety). There is also a textual description of the tasted wine.

We need to carry out the data exploration step to derive a first overview of the data to identify both properties we may exploit in further phases and possible problems we will have to take into account. By visualizing a summary description of the two parts of the dataset, we can notice that the columns with the highest percentage of null values are designation and region2, which is a more specific information of the production area. Figure 1 shows the number of occurrences of the top 10 countries in the dataset and, as expected, we can see that records belong mainly to the most famous wine-producing states (e.g., US, Italy, France).

The distribution of the target variable is shown in Figure 2 where we can see it is almost Gaussian, which is in general a desirable property. The mean is 46.27 while the standard deviation is 11.92, which is a slightly large value.

All the categorical features need to be represented in a suitable way to be correctly exploited by the regression model.

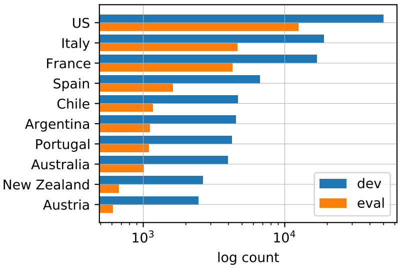

Fig. 1: Occurrences of top 10 countries

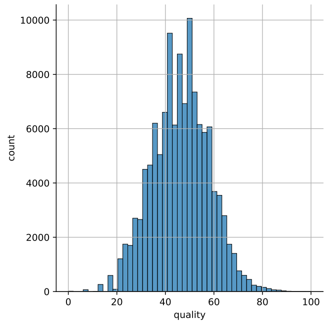

Fig. 2: Distribution of development set quality scores

By checking the number of distinct values for each attribute, we notice that some of them have a high cardinality (e.g., designation, winery) we will have to address in further steps.

II. Proposed approach

A. Data preprocessing

We have seen that the dataset contains a large number of null values we can fill in a proper way or remove. Since the size of the development set would strongly decrease, we chose the first approach. We can assume geographical position has a big influence on determining the quality of wine: for this reason, we tried to fill the missing information.

We manually fixed the samples with no country or province but we did not manage to correct null values for other attributes because of possible inconsistencies that were found. For instance, considering wines produced by a given winery, there were multiple values for province, region1 and region2. We could have chosen the most common value in the given subset but not to modify in a wrong way the knowledge represented by the data, we decided to postpone this phase.

We noticed that the development set contains a significant number (∼35k) of duplicated rows: since there is no reason to keep them and their presence could alter the knowledge inferred by the model, we decided to remove them from the first part of the dataset. The evaluation set also includes duplicates but they do not influence the training of the model and they must be kept to correctly upload the output to the submission platform.

Before encoding categorical features with *one-hot vector* representation, we extracted new features from description, designation and variety fields. In particular, we added two columns containing binary values indicating if the review contains terms like "sparkling" or "vintage". This because wine quality can generally be influenced by these traits. Another feature we attempted to extract was the year, which is known to have a significant impact on the target variable.

As mentioned in section I, we can not use one-hot technique without a proper preprocessing otherwise, if we applied it on all the columns, the resulting dataset would have a huge number of features (∼45k). This would be infeasible from a computational point of view and we would have an extremely sparse matrix that will definitely worsen the performance of the model[1]. As a consequence, we decided to create an artificial value "Other" for each categorical attribute to limit their cardinality. This value was assigned on a statistical basis: we grouped the development set records by the attributes[2] in turn characterizing them by their mean quality value and we computed the aggregate mean μ_{agg} and standard deviation σ_{agg}. Then we selected those values which fell within $3\sigma_{agg}$ from μ_{agg} while records whose value was not in this interval were marked by "Other". The resulting number of columns was 2 orders of magnitude smaller.

We can investigate how to extract relevant information from the textual description provided with each review applying text mining techniques. The proposed solution adopts *tf-idf*[3] representation limited to the first N_1 words. To exclude both too frequent and too rare words that would not be relevant from an informative point of view, appropriate min_df and max_df parameters must be set. Additionally, the meaning of adjectives and adverbs often depends on the context and it can have an impact on the feeling expressed by the review. For example, a *high* tannic wine is different from a *low* tannic one. To handle this variability, not only single words were considered but also *n-grams* (i.e. group of n words) of an adequate dimension.

Figure 3 shows the most relevant terms identified with this representation and we can easily see that some of them belong to domain-specific terminology (e.g., finish, tannins) while others are more common words but often appearing in wine context (e.g., cherry, sweet). Another feature of interest we can extract from the textual description is its length: as we can see from Figure 4, higher quality scores are often associated with longer descriptions while lower ratings with shorter texts. Therefore it could be important for the prediction.

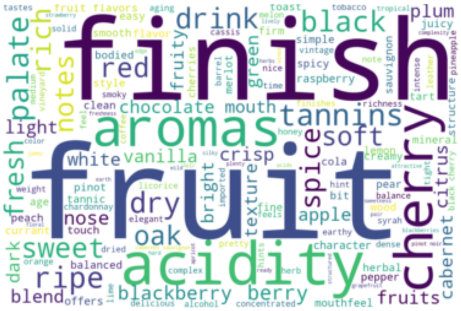

Fig. 3: Most relevant words according to *tf-idf*

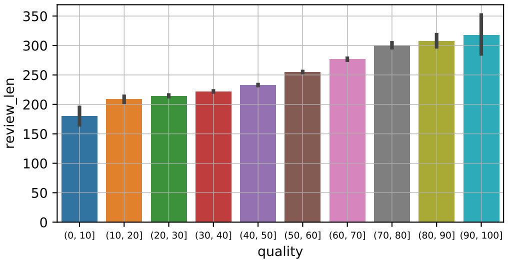

Fig. 4: Distribution of description lengths in quality ranges

We chose to apply[4] the same approach described for the description to a new field containing the merging of designation and variety, considering it as the "brand name" of a wine, limiting the results to the first N_2 words.

Both N_1 and N_2 together with the settings of the two tf-idf applications and the threshold t_{1h} used before one-hot encoding can be considered as hyperparameters of the preprocessing step whose tuning is discussed in II-C. Two dimensionality reduction techniques were tested: SVD and a pipeline of the previous with t-SNE[5] but they resulted computationally expensive without improving the results.

B. Model selection

The following regression models have been tested:

- *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 as it will be discussed in IV. We chose to use this model due to the complexity of the task because it can fit complex data;
- *Linear*: it is the simplest model and it associates to each explanatory variable (i.e. input feature) a coefficient

[1] This problem is known as the *curse of dimensionality*

[2] This was done applying a fixed threshold of occurrences

[3] term frequency-inverse document frequency

[4] This was done before one-hot encoding

[5] Singular Value Decomposition and t-distributed stochastic neighbor embedding

which reflects the contribution of that variable to the prediction. We decided to test it to assess its performance in difficult contexts and to make comparisons;
- *Ridge* and *Lasso*: these techniques perform both variable selection and regularization to improve the prediction accuracy and the interpretability of the model. This is done assigning values close or equal, respectively, to zero to those coefficients associated with features that are not relevant in the regression. We evaluated them to try to reduce the complexity of the model due to the high number of attributes;
- *SVR*: this algorithm applies a transformation to the data which is in general non linear using a kernel function. It tries to find the hyperplane which maximizes the margin of errors between predicted and actual values that are tolerated. We chose to assess it because, in general, it is one of the best performing algorithm but since we have not examined it in detail, its parameter tuning was less accurate.

We excluded the possibility of trying *polynomial* models because they compute new features that are power functions of the input variables. Since the input dimensionality was already high, it would have been prohibitive from a computational perspective.

For all regressors, the best configuration of hyperparameters has been determined using a grid search, as discussed in the following section.

C. Hyperparameters tuning

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

- preprocessing: t_{1h}, N_1, N_2, $\mathit{min\_df}$, $\mathit{max\_df}$, $\mathit{ngram\_range}$;
- final predictions: regression models hyperparameters.

We will first find the values for the first group and then we will maximize the result tuning the second group.

Since the number of different combinations for preprocessing hyperparameters is already high, we can limit it assuming that $N = N_1 = N_2$. To set their values, we can run a grid search on the candidate values with an 80/20 train/test split on the development set. We will train a random forest with its default configuration[6] and evaluate its performance through the R^2 score.

Subsequently, we can run another grid search with cross validation using 10 splits on the various regression models, based on the hyperparameters described in Table I.

III. Results

The tuning of preprocessing hyperparameters was performed considering uniquely random forest because among all the selected models it was the only suitable for fitting complex data, apart from SVR that would have needed a deeper research. The best performing configuration found was: $\{t_{1h}$=100, N=30, $\mathit{min\_df}$=2, $\mathit{max\_df}$=0.4, $\mathit{ngram\_range}$=(1,3)$\}$.

Model	Parameter	Values
Preprocessing	t_{1h}	{50, 100, 200}
	$N = N_1 = N_2$	{10, 30, 50, 70}
	$\mathit{min\_df, max\_df}$	{5, 30}, {0.2, 0.3, 0.4}
	$\mathit{ngram\_range}$	{(1,2), (1,3)}
Random forest	$\mathit{n\_estimators}$	{50, 100, 150, 200}
	$\mathit{criterion}$	mse
	$\mathit{max\_features}$	{sqrt, log2}
	$\mathit{max\_depth}$	{100, None}
Linear	$\mathit{fit\_intercept}$	True
Ridge	alpha	{1, 0.1, 0.05}
	tol	{1e-3, 1e-6}
	solver	{auto, cholesky}
Lasso	alpha	{1, 0.1, 0.01}
	tol	{1e-3, 1e-6}
	$\mathit{max\_iter}$	{1000, 2000}
SVR	kernel	{poly, rbf, sigmoid}
	$\mathit{max\_iter}$	{5000, 10000}
	gamma	scale
	tol	{1e-3, 1e-4}

TABLE I: Hyperparameters considered

Having set these values, we ran the grid search for the regressors and we identified the following best configurations:

- Random forest: $\{\mathit{n\_estimators}$=200, $\mathit{criterion}$=mse, $\mathit{max\_features}$=sqrt, $\mathit{max\_depth}$=None$\}$ $(R^2 \approx 0.58)$;
- Linear: $\{\mathit{fit\_intercept}$=True$\}$ $(R^2 \approx 0.52)$;
- Ridge: $\{\mathit{alpha}$=1, tol=0.001, solver=auto$\}$ $(R^2 \approx 0.53)$;
- Lasso: $\{\mathit{alpha}$=0.01, tol=1e-6, iter=1000$\}$ $(R^2 \approx 0.48)$;
- SVR: $\{\mathit{kernel}$=rbf, $\mathit{max\_iter}$=10000, gamma=scale, tol=1e-4$\}$ $(R^2 \approx 0.55)$.

The five regressors show similar results: we can identify random forest as the best performer followed by SVR, Ridge, Linear and Lasso models. In Figure 5 we can see the improvement in the performance of random forest during the run of its grid search. We decided to modify $\mathit{n\_estimators}$ value setting it to 100 to limit the risk of overfitting.

We assessed the performances of random forest without removing duplicates and we obtained a local score of 0.74. This considerable difference is probably due to the splits made by cross validation: since the high number of duplicates, one record will likely be in the training and its copy in the validation set. As a consequence, the model will correctly predict the value of the record that has been used for training and this leads to a better score.

We used the whole development set data to train the regression algorithms 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: 0.776 for random forest, 0.564 for SVR, 0.556 for Ridge, 0.555 for linear and 0.495 for Lasso. The private scores should almost reflect the public ones because there should not be overfitting. The difference between local and public score can be justified similarly: the evaluation set contains samples already present in the development set which, for this reason, are correctly predicted by the regressor, with a resulting higher score.

[6] Apart from *max_features* set to "sqrt" to speed up the process

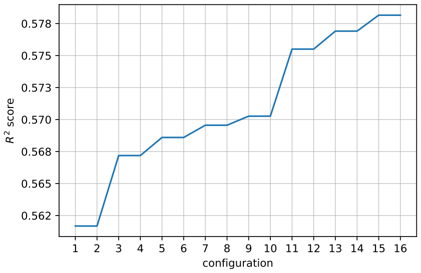

Fig. 5: Random forest performances

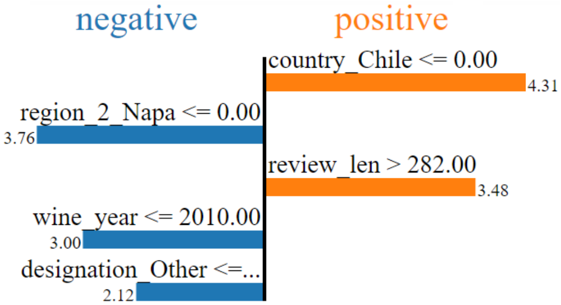

Fig. 6: Sample prediction explanation

To better understand the difficulty of the task, the proposed approach has been compared with two other solutions:

- random guess: generation of the target variable from a normal distribution with mean and standard deviation computed from the development set;
- naïve solution: one-hot encoding of the categorical attributes considering only the best and worst 10 values without any feature extraction or text analysis technique.

For both solutions a default random forest was trained obtaining a public score of -0.985 and 0.327, respectively.

IV. Discussion

The proposed approach achieves better results than random and naïve solutions: this is done through the feature selection and extraction phase which also exploits a text analysis method. With respect to other solutions listed in the leaderboard, at the time of writing we can see that higher scores are concentrated around 0.83 which does not seem too distant.

The examined regressors perform similarly with the exception of random forest which is the only reaching acceptable results. There are various aspects that might be taken into account to improve the obtained results:

- Try to extract technical characteristics from the textual description that can influence the assigned quality (e.g., percentages in case of blended wines, alcohol content, years of aging);
- Better define the threshold used before one-hot encoding or adopt different techniques of encoding categorical features (e.g., *frequency encoding*);
- Investigate more advanced text mining techniques (e.g., *word embeddings*): in [1] it was shown that wine experts share a common vocabulary to describe wines that is used in a consistent way. This allows to automatically predict wine characteristics from a review text;
- Run a more exhaustive grid search both on preprocessing and regressors hyperparameters. Moreover, additional regression models might be considered.

Another important aspect we decided to analyze was the model interpretability which is important both to explain the output predictions and it can be exploited during the analysis, for example during the feature selection phase. In contrast to decision trees, random forest is less interpretable because predictions are made by hundreds of trees and not by a single tree. It can provide global feature importances which allow us to identify the most important attributes for the regression. In this problem, the features that drive most the quality predictions are the description length, wine year, some geographical origins (e.g., California-Napa, Chile) and wine varieties (e.g., Pinot Noir, Chardonnay).

The explainability of a model can be examined in more detail through a Python package called *LIME*[7] presented in [2]. It allows to explain individual predictions visualizing the degree of correlation among the features used by the model and the target variable. In Figure 6 we can see the first five features that drove the prediction, divided into positively and negatively correlated. Since the task is a regression problem, the attributes are not associated with a specific class but they guide the algorithm towards the two different "directions". For example, we can see two geographical information correlated in a different way while the description length has a positive correlation with the quality score, which confirms what has been discussed in II-A.

To make the model more interpretable, it will be important to reduce the number of features, which is the major aspect to improve. Nevertheless, the achieved results are promising and they will only benefit from the proposed improvements.

References

[1] E. Lefever, I. Hendrickx, I. Croijmans, A. Van den Bosch, and A. Majid, "Discovering the language of wine reviews: A text mining account," 05 2018.

[2] M. T. Ribeiro, S. Singh, and C. Guestrin, "Why Should I Trust You?: Explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA*, pp. 1135–1144, 2016.

[7] Local Interpretable Model-Agnostic Explanations