# Report title

Mattia Delleani
*Politecnico di Torino*
Student id: s288854
s288854@studenti.polito.it

***Abstract*—In this report we introduce a possible approach to the Wine Review Dataset regression problem. In particular, the proposed approach consists in extracting knowledge from a set of categorical and text features in order to predict the wine quality. In order to predict these values, data encoding techniques are computed and used as input for three regression models. The proposed approach outperforms a baseline defined for the problem and obtains overall satisfactory results.**

## I. Problem overview

The proposed competition is a prediction problem on the Wine Review Dataset, a collection of wines of different types and origins. The goal of this competition is to correctly predict the quality scores of unlabeled wines. The dataset is divided into two parts:

- a development set, containing 120744 records with the quality values
- an evaluation set, comprised of 30186 records without the quality values

We will need to use the development set to build a regression model to correctly predict the quality scores of the evaluation set. We can make some considerations based on the development set. There are 8 columns, which are:

- description: a quick text review of the wine
- country, province, region1, region12: the geographical origin of the wine
- winery: the name of the winery that produces the current wine
- designation: the nomination of the wine
- variety: the grapes present in the wine

To get a better understanding of the type of data at hand, we can inspect the dataset and its proprieties. First, we can compute some basic statistic on the *quality* columns to better understand how the quality score is represented. The quality goes from 0 to 100, both boundaries are included, and in the development set seems to have almost a normal distribution (Figure 1). Second, the most of the columns have categorical attributes and only the description can be seen as actual text, so all columns must be converted into numbers before using any kind of machine learning algorithm. Figure 2 shows that there are many missing values, especially in *region2* category. Moreover, there many duplicated rows in the dataset that we need to handle.

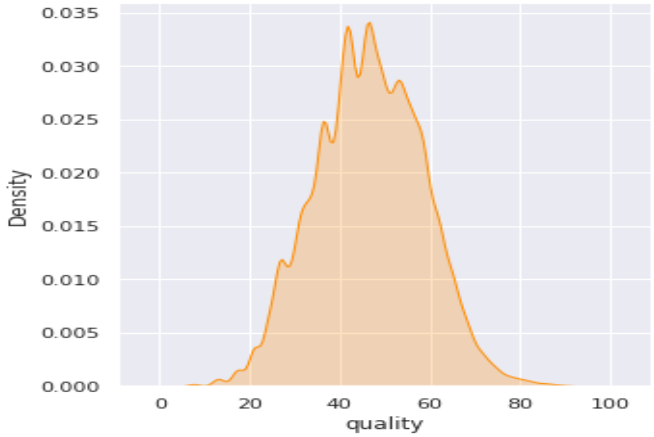


Fig. 1. Quality distribution

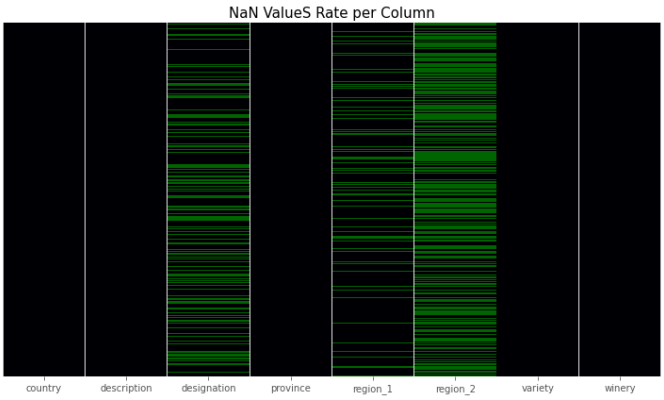


Fig. 2. NaN values per column

## II. Proposed approach

### A. Preprocessing

We have seen that there are many duplicated rows, maybe because our dataset comes from merging different sources. In order to prevent overfitting we drop these rows and this action involves a significant reduction in the size of the dataset, almost fifty percent of the original one. Even if some columns have an high percentage of missing values, we decided to fill the null values with a blank space, instead of drop them, because they may provide useful information. Since we have categorical columns that we need to encode them into numbers, and in order to choose the type of encoding

```
country: 46
description: 56529
designation: 20872
province: 402
region_1: 1128
region_2: 19
variety: 560
winery: 12330
quality: 86
```

Fig. 3. Unique values per column

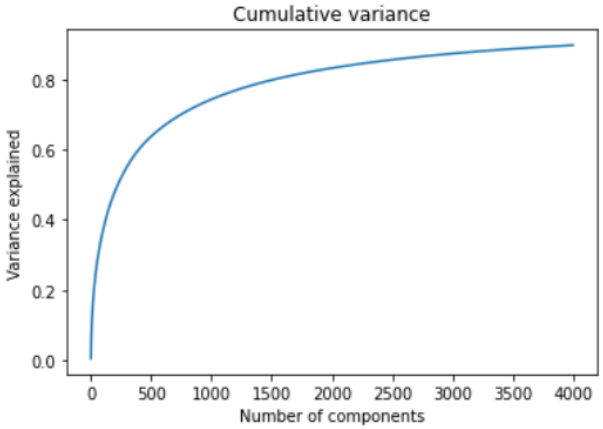


Fig. 4. Relation between variance and *'n_components'*

and problem approach is good to check the unique values by columns. As we can see Figure 3 shows that there are many unique values and data are not ordinal attributes, this means that we cannot infer an order among them. As a result, even if we have a large number of unique values, one hot encoding is applied to the categorical columns after a case normalization, whereas a simple term frequency vectorizer, that check if a word is present or not in the description, is used for handling the description column. The result obtained is a sparse matrix with 56540 rows and 59668 features. Features are quite a lot and curse of dimensionality may occur, so dimensionality reduction is applied using TruncatedSVD. This transformer performs linear dimensionality reduction by means of truncated singular value decomposition (SVD) and can be applied on sparse matrix, because does not center the data before computing SVD. Figure 4 shows the variance explained with respect to the number of components after the SVD is applied, with dimensionality of output data set at 4000 components. To represent 90 percent of the variance at least 1515 elements are needed.

### B. Model selection

The following algorithms have been tested on the full rank matrix:

- Lasso: this regression algorithm aims to produce a model that has high accuracy and only uses a subset of the original features. It performs both variable selection and regularization in order to enhance the prediction accuracy and interpretability of the statistical model it produces. The way it does this is by putting in a constraint where the sum of the absolute values of the coefficients is less than a fixed value, which forces some coefficients to be set to zero. Lasso does not do well when you have a low number of features and also with features that are highly correlated, because some of them may be dropped when they do have an effect on the model when looked at together.
- Ridge: the aim of this regression algorithm is the same as Lasso, but the process is a little bit different. It performs both variable selection and regularization to reduce model complexity and prevents overfitting. It does this by putting a constraint, same as Lasso, but it does not drop any of the coefficients to zero, it simplifies the model by shrinking the size of some coefficients. It performs better in cases where there may be high multi-collinearity, or high correlation between certain features. This is because it reduces variance in exchange for bias.

While, Random Forest Regressor has been tested on the reduced matrix. Random Forest leverages multiple decision trees (trained on various subsets of data and features) to make predictions and merge their predictions together to get a more accurate and stable prediction rather than relying on individual decision trees. This typically avoids the overfitting problems of decision trees and increases robustness but still maintaining some degree of interpretability. Random forests, like decision trees, work on one feature at a time, so no normaliziation is necessary.

For these classifiers, the best-working configuration of hyperparameters has been identified through a grid search, as explained in the following section.

### C. Hyperparameters tuning

There are two main sets of hyperparameters to be tuned:

- the number of dimension in the preprocessing step
- Lasso, Ridge and Random forest parameters

Before running the grid search, that is time and computationally costly especially for the random forest, an 75/25 train/test split on the development set and a relation analysis between the number of dimensions and the r2 score has been performed. The same split has been used to compute the r2 score with the full rank matrix using Lasso and Ridge, with default configurations and assess their performance based on the resulting r2 score. We can run a grid search on the preferred algorithm, based on the hyperparameters defined in Table I.

## III. Results

Figure 5 shows that Random forest tends to have a lower score than Ridge (0.4 vs 0.68) and also has a bigger impact in time cost than Ridge. As we may expected, Lasso does not work well with this problem, because there are correlated features, such as: country, province, region1 and region2, that all provides geographical information but with different detail level. On the contrary, Ridge seems to works well as there is correlation and multi-collinearity. Among this models, we

TABLE I
HYPERPARAMETERS CONSIDERED

| Model | Parameters | Values |
|---|---|---|
| Random forest | $n\_estimators$ | 100, 200,300 |
| | *criterion* | mse, mae |
| | $max\_features$ | "auto", "sqrt" |
| Ridge | *alpha* | 1,0.1,0.01,0.001,0.0001,0.005,0.05 |
| | $fit\_intercept$ | True, False |
| Lasso | *alpha* | 1,0.1,0.01,0.001,0.0001,0.005,0.05 |
| | $fit\_intercept$ | True, False |

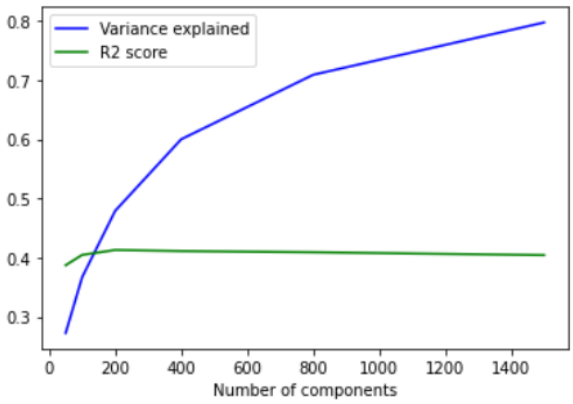


Fig. 5. Random forest score and variance, with respect to $n\_components$

choose the best performer, which is actually Ridge and we performs the hyperparameters tuning using a grid search. The best configuration for Ridge was found for ('*alpha*': 1, '$fit\_intercept$': True), and we trained on all available development data, and locally we obtained an r2=0.688. Then the models have been used to label the evaluation set. The public score obtained is of 0.791. Since the score obtained using the evaluation data is 0.1 higher than what we reached locally, it is reasonable that the model is capable of generalize well and there should be no overfitting. Reasonably, we can assume that similar results can be obtained on the private score. For comparison, the same model was evaluated without removing the duplicated rows from the development set. The results achieved were 0.822 local and 0.84 in public leaderboard. There is still an improvement on the score, maybe because even the eveluation set contains duplicates, but the gain in r2 score is lower, and this means that the second approach does not generalized as well as the previous one.

## IV. DISCUSSION

The proposed approach achieves results that far outperform the naive baseline defined. There are some aspects that might be worth considering to further improve the obtained results:

- Other feature extraction approaches may be considered in order to significantly decrease the number of dimensions that allows us to try other models that cannot works with sparse matrices. This could be done by building a custom description handler that can extract useful information such as the year of the wine that is numerical and a relevant information in the context. Other approach is to use a recursive features elimination, this algorithm applies a backward selection process to find the optimal combination of features in predicting the target variable in a predictive model.
- Run a more complex model, like MLPRegressor an important model of Artificial Neural Network that can be used as regressor for prediction, and choose the best set of hyperparameters thorough grid search process. This approach would have an high computational cost and very high time consuming.

## REFERENCES

[1] Paper 131 (2018), *Regulation Techniques for Multicollinearity: Lasso, Ridge, and Elastic Nets*, Deanna Schreiber-Gregory, Henry M Jackson Foundation

[2]Asian Journal of Economics and Banking (2020), *How to Choose Tuning Parameters in Lasso and Ridge Regression?*, Chon Van Le, International University, Vietnam National University