Wine Quality Prediction on the UCI Wine Dataset

Massimiliano Pronesti

Politecnico di Torino
s287646@studenti.polito.it

Abstract

This report provides an analysis of the effectiveness of different classification approaches applied to the popular wine quality prediction problem on the wine dataset from the UCI repository. Specifically, the goal is to predict whether the wine has a good or a bad quality, given a set of features related to its chemical composition, such as the pH and the percentage of free sulfur dioxide. The original dataset consists of 10 classes, however, for this work, the dataset has been binarized, collecting all wines with low quality (lower than 6) into class 0, and good quality (greater than 6) into class 1, while those with quality 6 have been discarded. In addition, the dataset contains both red and white wines (merged for the sake of this analysis). There are 11 features, that represent physical properties of the wine, with partially balanced classes.

I. Data Analysis

In this section, we are going to conduce an analysis on the main characteristics of the features contained in the training dataset. The training set consists of 1126 bad quality samples and 613 good quality samples, then one class is twice as much present as the other. A visualization of how raw features are distributed is shown in Figure 1.

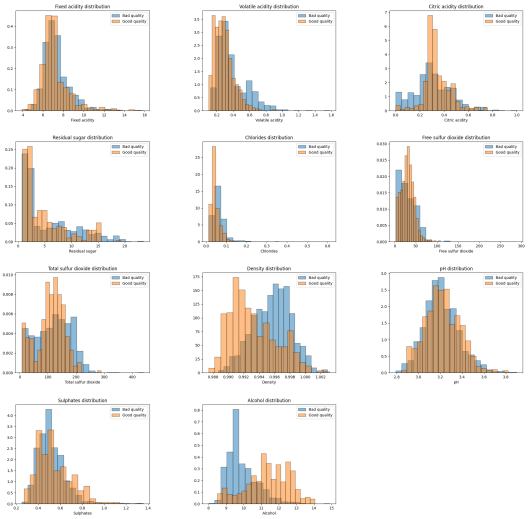

Figure 1: *Raw features distribution of the UCI Wine Quality Dataset*

We can observe that features don't have a zero mean, therefore we might consider standardizing them, i.e. centering data and scaling it by the variance, so that the obtained random variable has zero mean and unitary variance.

In addition, features don't expose a Gaussian trend, with the presence of some outliers.

Therefore, we gaussianize the features, computing the cumulative rank $r(x)$ over the training set

$$r(x) = \frac{\sum_{i=1}^{N} I[x < x_i] + 1}{N + 2}$$

being x_i the value of the considered feature for the i-th sample, and transforming the features computing the inverse of the cumulative distribution function Φ fed with the rank $r(x)$

$$X_{gauss} = \Phi^{-1}(r(x))$$

The distribution of the gaussianized features is shown in Figure 2.

Moreover, we provide an analysis of the correlation between the features, exploiting the Pearson product-moment correlation coefficient, defined as

$$\rho_{X,Y} = \frac{cov(X, Y)}{\sigma_X \sigma_Y}$$

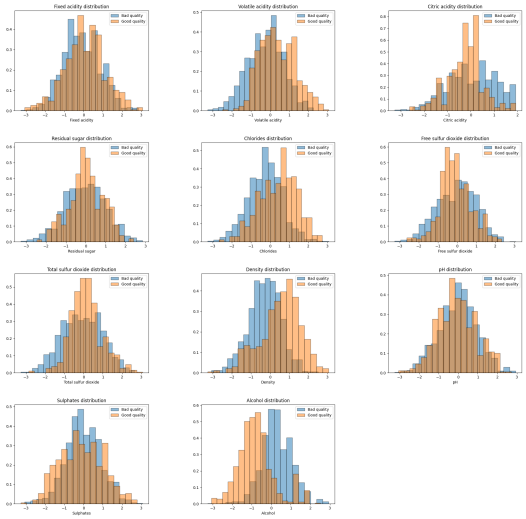

Figure 2: *Gaussianized features distribution of the UCI Wine Quality Dataset*

being $cov(X, Y)$ the covariance matrix of X and Y, expressible as the expectation of the product of X and Y centered using their respective mean.

$$cov(X, Y) = \mathbb{E}[(X - \mu_X)(Y - \mu_Y)]$$

The obtained heatmaps among the gaussianized features are shown in Figure 3.

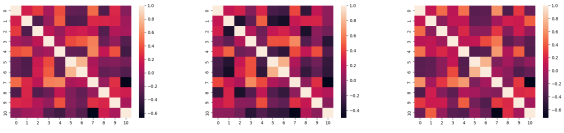

Figure 3: *Heatmaps among Gaussianized features*

where we can observe some features are correlated (darker colors in the heatmap), thus exploring dimensionality reduction techniques such as the PCA, could prove beneficial.

II. Method

In this section we are going to first describe the approach followed towards model selection and model evaluation. Then, we will analyse the explored classification methods and the results they yielded.

I. Approach

In order to perform our analysis, we will adopt two approaches towards model selection:

- single split: the training set is divided into two chunks, where the 80% of the samples are used for fitting the classifer and the remaining 20% for testing it.

- k-fold cross validation: the training set is split into k folds, one of whom is used of testing and the other $k-1$ for fitting the model. The process is repeated k times. This approach usually makes the process of model selection more reliable as, one by one, all the chunks will be used as unseen data. For this specific application, we set $k = 5$, i.e. we used 5 folds.

In both cases, we make sure no transformation is applied on the whole training data before splitting it, not to introduce data leakages.

As regards model evalution, we want to be Bayesian and adopt the minimum of the normalized Bayesian risk as metric, which measures the cost we would pay if we made optimal decisions using the recognizer scores. The application of interest is a uniform prior one

$$(\tilde{\pi}, C_{fp}, C_{fn}) = (0.5, 1, 1)$$

being π the (unbiased, in the specified case) prior, C_{fp}, C_{fn} the cost of the false positive and false negative case, respectively.

II. Gaussian Classifiers

Gaussian classifiers are the first class of methods we take into considerations. In particular we analyse the performances yielded by a Multivariate Gaussian and a Naive Bayes, both with full and tied covariance, for a total of 4 classifiers.

Table 1 shows the results obtained for these models both for the single split and for the k-fold cross validation.

	Single split $\tilde{\pi} = 0.5$	**5-fold** $\tilde{\pi} = 0.5$
Raw features		
Full-Cov	0.00	0.00
Diag-Cov	0.00	0.00
Tied Full-Cov	0.00	0.00
Tied Diag-Cov	0.00	0.00
Gaussianized features		
Full-Cov	0.00	0.00
Diag-Cov	0.00	0.00
Tied Full-Cov	0.00	0.00
Tied Diag-Cov	0.00	0.00
Gaussianized features, PCA(m=10)		
Full-Cov	0.00	0.00
Diag-Cov	0.00	0.00
Tied Full-Cov	0.00	0.00
Tied Diag-Cov	0.00	0.00
Gaussianized features, PCA(m=9)		
Full-Cov	0.00	0.00
Diag-Cov	0.00	0.00
Tied Full-Cov	0.00	0.00
Tied Diag-Cov	0.00	0.00

Table 1: *min DFC for Gaussian models*

III. Logistic Regression

In this section, we assess the performances of a Logistic Regression classifier, both with linear and quadratic kernel.

Being classes unbalanced, their costs are rebalanced using a loss function accounting for the number of samples for each class, used as weights for the two terms deriving from the split of the summation of the original loss

$$J(w,b) = \frac{\lambda}{2}\|w\|^2 + \frac{\pi_T}{n_T}\sum_{i|c_i=1} log\ \sigma(y_i)^{-1} + \frac{1-\pi_T}{n_T}\sum_{i|c_i=0}^{n} log\ \sigma(y_i)^{-1}$$

being σ the sigmoid function, y_i the output of the model $y_i = -z_i(w^T x_i + b)$, z_i a variable equalt to ± 1 depending on the class, w, b the parameters of the model.

IV. Support Vector Classifier

In this section, we test the performances of different flavours of support vector classifiers. In particular, we are going to use a linear kernel, a quadratic polynomial kernel and a radial basis function kernel.

As done for the previous class of models, we will also take into account the possibility of rebalancing the classes introducing an empirical prior $\pi_{T,emp}$ over the training set, thus using two different values of C for the bad and good quality wines

$$C_T = C\frac{\pi_T}{\pi_{T,emp}} \qquad C_F = C\frac{1-\pi_T}{1-\pi_{T,emp}}$$

V. Gaussian Mixture Models

Eventually, Gaussian Mixutres are the last type of classifier employed for this task, for which we expect to yield generally better results than Gaussian models, as GMMs can approximate generic distributions.

We consider both full and diagonal covariance models, with and without covariance tying, where tying takes place at class level (i.e. different classes have different covariance matrices).

III. Evaluation

IV. Discussion and Conclusions