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.

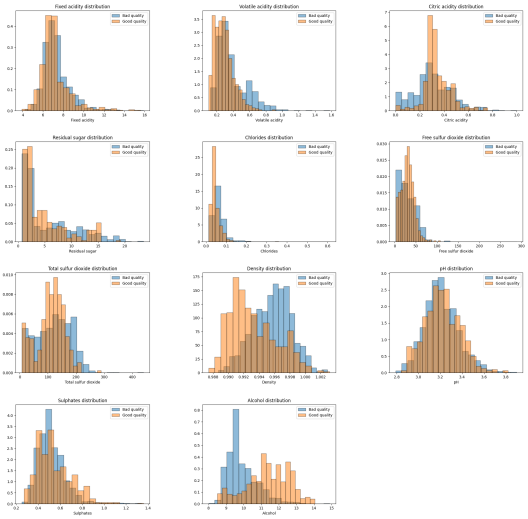

Figure 1: *Raw features distribution of the UCI Wine Quality Dataset*

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

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

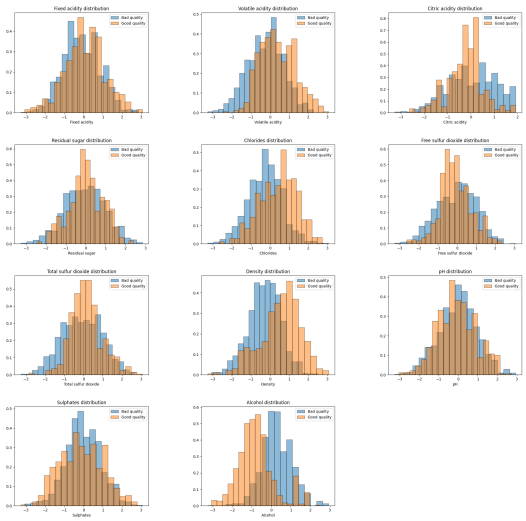

Figure 2: *Gaussianized features distribution of the UCI Wine Quality Dataset*

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.

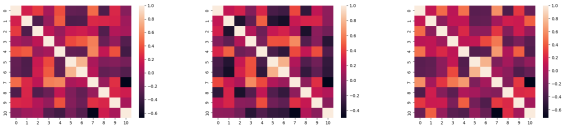

Figure 3: *Heatmaps among Gaussianized features*

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.

II.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 for validation 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 **bias** the validation and 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 $\tilde{\pi}$ the (unbiased, in the specified case) prior, C_{fp}, C_{fn} the costs of the false positive and false negative case, respectively.

II.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 on raw and gaussianized data and applying a principal component analysis.

	Single split $\tilde{\pi} = 0.5$	**5-fold** $\tilde{\pi} = 0.5$
Raw features		
Full-Cov	0.264	0.312
Diag-Cov	0.409	0.420
Tied Full-Cov	0.327	0.333
Tied Diag-Cov	0.391	0.403
Gaussianized features		
Full-Cov	0.255	0.301
Diag-Cov	0.460	0.440
Tied Full-Cov	0.344	0.347
Tied Diag-Cov	0.441	0.441
Gaussianized features, PCA(m=10)		
Full-Cov	0.242	0.289
Diag-Cov	0.372	0.385
Tied Full-Cov	0.359	0.352
Tied Diag-Cov	0.362	0.350
Gaussianized features, PCA(m=9)		
Full-Cov	0.254	0.304
Diag-Cov	0.384	0.392
Tied Full-Cov	0.361	0.351
Tied Diag-Cov	0.361	0.354

Table 1: *min DCF for Gaussian models*

We notice that the full-covariance Multivariate Gaussian classifier yields the better performances, both with and without PCA. In particular, the dimensionality reduction is beneficial when applyied with 10 components and doesn't degrade the performances with 9.

This suggests the dataset is large enough to estimate a covariance matrix for each class and loosening our assumptions (introducing diagonalization or tying) is not needed. This is furtherly suggested by the fact that the diagonal covariance models yield the worst results, as the uncorrelation hypotesis does not hold.

Eventually, it should be noted that, as expected, gaussianization improves the performances with respect to raw data. The results obtained for the single split and 5-fold are consistent to one another, conveying that, in this specific scenario, the size of our training set is appropriate even for a single fold evaluation (i.e. a single split).

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

The most significant results obtained for both the models on a grid search on λ are shown in Table 2, while a complete visualization is shown in Figure **??**.

	Single split $\tilde{\pi} = 0.5$	**5-fold** $\tilde{\pi} = 0.5$
Raw features		
Log-Reg ($\lambda = 10^{-4}$)	0.352	0.363
Quad-LR ($\lambda = 0$)	1.000	1.000
Z-normalized features		
Log-Reg($\lambda = 10^{-2}$)	0.353	0.344
Quad-LR($\lambda = 0$)	0.237	0.274
Gaussianized features		
Log-Reg($\lambda = 0$)	0.335	0.356
Quad-LR($\lambda = 10^{-2}$)	0.256	0.294
Z-normalized features, PCA(m=10)		
Log-Reg($\lambda = 10^{-2}$)	0.353	0.341
Quad-LR($\lambda = 0$)	0.243	0.284

Table 2: *min DCF for Logistic Regression models*

Differently from the previous case, Gaussianization doesn't prove this helpful, however standardizing data improved the performances. On the other hand, PCA does not bring any advantage, though it dosn't prove detrimental.

Overall, among the different values of λ and the two models in the 3 different scenario analysed, the quadratic logistic regression with $\lambda = 0$ yields the best results. In fact, quadratic models are characterized by a "curved" surface, which usually allow to better describe and learn the real distribution of the data, without overfitting.

II.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 (RBF) kernel.

	Single split $\tilde{\pi} = 0.5$	**5-fold** $\tilde{\pi} = 0.5$
Raw features		
SVC($C = 0.1$)	0.816	0.760
SVC($C = 0.1, \pi_T^{emp}$)	0.695	0.920
Z-normalized features		
SVC($C = 1$)	0.334	0.341
SVC($C = 0.1, \pi_T^{emp}$)	0.338	0.348
Gaussianized features		
SVC($C = 0.1$)	0.339	0.350
SVC($C = 0.1, \pi_T^{emp}$)	0.352	0.360

Table 3: *min DCF for Support Vector Classifier models, with and without feature balancing.*

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

$$C_T = C\frac{\pi_T}{\pi_T^{emp}} \qquad C_F = C\frac{1-\pi_T}{1-\pi_T^{emp}}$$

We will first conduct our analysis emplying the linear SVC, even if we already discussed above that linear models don't perform that good for our classification task.

The best results obtained for the linear kernel are reported in Table 3, where we specified π_T^{emp} where we balanced the features using the empirical prior. We observe how standardization improved greatly the results with respect to the raw case, where we obtain high values of minDCF both for the single split and 5-fold cross validation approach. Gaussianization and features balancing don't prove advantageous

A further analysis is, now, conducted using a polynomial kernel of grade 2 (quadratic) and a radial base function (RBF) kernel. We tune again C for both the cases and for the RBF kernel we perform a grid search involving γ as well, to select the best hypermameters. Overall, we try the following values:

$$C = \{10^{-1},\ 1,\ 10\}$$
$$\gamma = \{1/e,\ 1/e^2\}$$
$$\pi_T = \{0.5,\ \pi_T^{emp}\}$$

Place notice that π_T is included above, even if it's not an hyperparameter, only to convey the idea that we're searching the best combination of parameters and, at the same time, assess the benefits of rebalancing features, if any. A visualization of the grid is shown in Figure **??**.

We can notice that, for the RBF SVC, we obtain similar results regardless of features balancing. Therefore, balancing is not beneficial nor detrimental in our specific case. Table 4 shows the results of the 2nd degree polynomial kernel and RBF kernel for the best choices of the hyperparameters, both with and without balancing.

	Single split $\tilde{\pi} = 0.5$	**5-fold** $\tilde{\pi} = 0.5$
Z-normalized features		
Poly-SVC($C = 1$)	0.208	0.275
Poly-SVC($C = 0.1$)	0.234	0.272
RBF-SVC($C = 10, log\gamma = -2$)	0.140	0.225
RBF-SVC($C = 10, log\gamma = -2$)	0.160	0.223

Table 4: *min DCF for Support Vector Classifier models, with and without feature balancing.*

Overall, the RBF kernel yields the best results.

II.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) for a total of 4 models. The best number of components is selected via cross-validation on gaussianized and standardized features.

	5-fold $\tilde{\pi} = 0.5$
Guassianized features	
Tied-GMM (n_comp=4)	0.000
GMM(n_components=8)	0.000
Tied-Diag-GMM (n_comp=16)	0.000
Diag-GMM (n_comp=16)	0.000
Z-normalized features	
Tied-GMM (n_comp=4)	0.000
GMM (n_components=8)	0.000
Tied-Diag-GMM (n_comp=16)	0.000
Diag-GMM (n_comp=16)	0.000

Table 5: *min DCF for Gaussian Mixture Models*

II.VI. Model selection

At the end of our preliminary analysis, we can conclude that the Support Vector Classifier with an RBF kernel having $log\gamma = -2$, applied to standardized features yields the best performances.

However, the Quadratic Logistic Regression and the Gaussian Mixture with 8 components - both applied to standardized features - performed well. Therefore, we will mainly take these into account in the following analysis for the evaluation. To sum up, here's are our best choices:

- GMM(n_components=8)
- Quad-LR ($\lambda = 0$)
- RBF SVC($log\gamma = -2$)

II.VII. Combining the best classifiers

After selecting the most perfoming classifiers using the minimum detection cost function as metric, we aim at assesing their performances using the optimal theoretical threshold $t = -log\frac{\tilde{\pi}}{1-\tilde{\pi}}$. Since we're dealing with non-probabilistic models, we expect a (slight) worsening of the performances wrt to the min DCF.

II.VIII. Discussion

From the above results, we can observe that the fusion of the RBF SVC with the Quadratic Logistic Regression yields similar results of the same models fused with an 8 components full-covariance Gaussian Mixture Model. Our choice is, thereby, the simplest of the two as, given the same performances, the simpler of the two reduces the odds of incurring into overfitting.

III. Evaluation

IV. Discussion and Conclusions