Features Extraction & Red-Wine Quality Predictions

Università della Calabria

Dipartimento di Ingegneria Informatica , Modellistica , Elettronica e Sistemistica (DIMES)

Nicola Corea

Abstract. The aim of this work is to present a new solution to the problem of red wine quality prediction. Several articles provides different solutions to the above problem , based on single classifiers or ensemble techniques, whose results in term of accuracy on test data are lower than 70 %. As we will see , after appropriate pre-elaborations of the data , the accuracy of the classifiers can be increased up to 20-30% respect to the current solutions. The application of techniques for dimensionality reduction , like selection and extraction , can be used to reduce the number of features taken in consideration , and loosing at the same time as little as possible on the accuracy. The dimensionality reduction allow us to obtain model simpler than the previous classifiers ,and reduces the computational cost required to train them. We will conclude our discussion by training a multi-layer perceptron (MLP) for classification purposes, and we are going to compare the results with those obtained by simple and standard classifiers.

I. Introduction

The aim of this article is to train different classificators with the purpose to predict the red wine quality on the base of the wine's characteristics. To train our classificators we refer to the dataset available from UCI machine learning repository. The dataset , thanks to the work of P. Cortez , A. Cerdeira , F. Almedia , contains 1599 examples of the red portuguese "Vinho Verde" wine , classified on base of different characteristics , such as : 'fixed acidity' , 'volatile acidity' , 'citric acid' etc… . On the base of these features , the quality of the wine is represented by an integer in the closed interval [3,8]. The value '3' will be refered to a low quality of the wine , instead , the integer '8' will represent an high quality. The performance of the classifiers will be evaluated in terms of accuracy. To this end , with reference to the confusion matrix , represented in **Figure 1.1** , said

$$ERR = \frac{FP + FN}{FP + FN + TP + TN}$$

the classification error , the accuracy of the classifier is given by

$$ACC = 1 - ERR = \frac{TP + TN}{FP + FN + TP + TN}$$

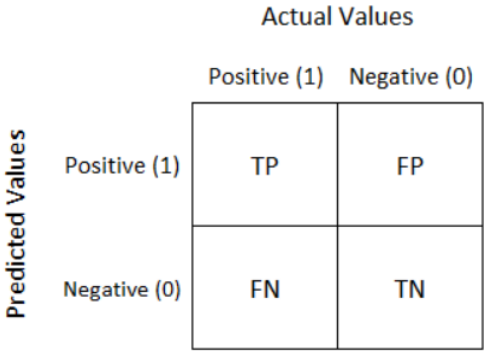

Figure 1.1 Confusion Matrix

We start our discussion , by presenting the results obtained by the classifiers without the modification (pre-processing) of the labels associated to the data. In particular , according to [1] , we will see that the accuracy on the test data , is lower than 70%. The aim of this work , is to find possible solutions with the intention to improve the performance of the classifiers. As we will see , with an appropiate pre-processing phase on the class labels , the accuracy of the predictors can be bring to 95-96% on test data. After doing so , with the purpose to use ensemble techniques such as Boosting , Bagging etc.. , we present different solution to dimensionality reduction , like SBS (Sequential Backward Selection) and PCA (Principal Components Analysis). With the application of these techniques we will obtain simplex classifiers , and of course , we have a reduction on the

computational cost required to train our classifiers. The article end with the training of a multi-layer perceptron (MLP) for classification purposes , and we analyze and compare the results with those obtained in the previous section.

II. Problem Analysis

With the aim of evaluating the performance of the model obtained , before leaving it free to operate in the real world , we split the dataset in a 70% for training and a 30% for test. The next step, of considerable importance , is the standardization (or normalization) of the feauteres. Some algorithms , such as KNN (K-Nearest-Neighbors) or SVM (Support – Vector-Machine) are not invariant respect to the feature scales. For example , remember that in the KNN algorithm , the selection of the k-neighbors is equals to find the k examples in the dataset at minimum distance from the example that we want to classify. If we choose , as metric , the Euclidean Norm (Squared)

$$|x - x_k|_2^2 = \sum_{i=1}^{n} (x_i - x_{ki})^2$$

we can see how the distances are most influenzed by the greatest features. So , before proceding , we have to standardize the features. In **Table 1.1** we report some aspects of the features , like means and variance , and the minimum and maximum values. As we can see , the values of some features , such as , 'Fixed Acidity' , 'Residual Sugar' , are greater than others.
Given means and variance for each characteristic , the examples will be tranformed in according to the following equation

$$x_{std}^{i} = \frac{x^i - \mu_x}{\sigma_x}$$

and they can be interpreted as realization of a normal distribution with 0 mean and 1 deviation. So now , we can train our classifiers , and we can evaluate the accuracy of the single predictors.

	"fixed acidity"	"volatile acidity"	"citric acid"	"residual sugar"
count	1599.000000	1599.000000	1599.000000	1599.000000
mean	8.319637	0.527821	0.270976	2.538806
std	1.741096	0.179060	0.194801	1.409928
min	4.600000	0.120000	0.000000	0.900000
25%	7.100000	0.390000	0.090000	1.900000
50%	7.900000	0.520000	0.260000	2.200000
75%	9.200000	0.640000	0.420000	2.600000
max	15.900000	1.580000	1.000000	15.500000

Table 1.1 Characteristics Aspects

Let's proceed with the training of the following classifiers:

1) KNN (K-Nearest-Neighbors)
2) SVM Kernel (RBF Kernel)
3) DecitionTree Classifier (Entropy)

Remember that we evaluate the performance of our classifiers on base of the accuracy. In the following table we report the results obtained both in training and testing phase

Classifier	Training	Test
SVM	0.94	0.69
KNN	0.66	0.58
DTC	0.67	0.59

Table 1.2 Results

As we can see from the Table 1.2 , and as stated in [1] , the best classifier for this kind of problemi is the SVM algorithm. In particular, with reference to the following criterior

$$J(w) = \frac{1}{2}|w|_2^2 + C\left(\sum \xi_i\right)$$

for the selection of the hyperplane , and with respect to the following kernel function

$$k(x_i, x_j) = \exp(-\gamma|x_i - x_j|)$$

we have choosen the hyperparameters as $C = 10.0$, $\gamma = 0.25$. We can try to improve the accuracy by using ensemble techniques. But in this dataset we have 13 features , so the number of the classifiers in the ensemble has not to be large , related to computational problems.

Ensemble	Training	Test
MV	0.79	0.63
RF	0.99	0.68
AB	0.56	0.54

Table 1.3 Ensemble Results

The **Table 1.3** shows the results obtained by using different ensemble techniques like Boosting , Bagging and HardVoting (Majority Voting). In particular in the majority ensemble we have used the previous trained classifiers.

But as we can see , from the previous results , also by using ensemble techniques , we cannot obtain good performances. In particular , a Random Forest classifier (Bagging with DecisionTree) give us results that are not greater than the accuracy obtained with a single SVM classifier.

A first reason to these lower results could be found in a not uniform distribution of the examples over the classes. Infact , as Figure 1.2 shows , the example are major distributed over the classes [5,6]. But , analyzing more in details the examples in the dataset , the problem to these lower result could be due to the fact that examples belong to different classes share same values of the most significant feauteres.

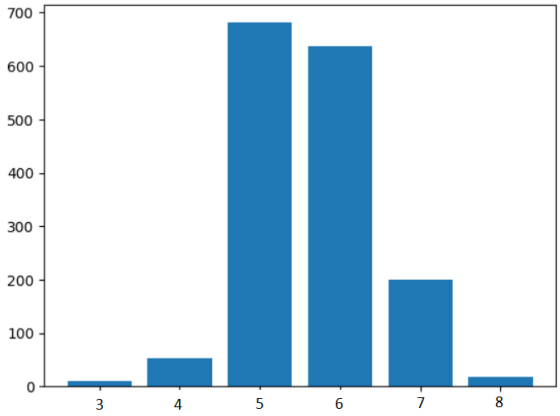

Figure 1.2 Examples for class

We will see in the next section how we can improve the accuracy of our classifiers. The idea behind the proposed solution , consists in a second pre-elaboration phase , where we are going to modify the labels associated to the examples in the dataset.

III. Problem Solution

As said in the previous section , the idea behind the solution , consists in a redefinition of the class labels associated to the examples. In particular , from the interval [3,4,5,6,7,8] of the current labels , we pass to the classes [0,1,2], that correspond to three different levels of quality of the red "Vinho Verde" wine. Between class labels and wine quality , we have the following relations

Label	Wine Quality
0	Low
1	Medium
2	High

Table 1.4 Labels to Quality Association

Obviously, remains to establish how to redefine the class label for each example of the dataset. The transformation of the label for each example is done according to the strategy presented in **Figure 1.3**

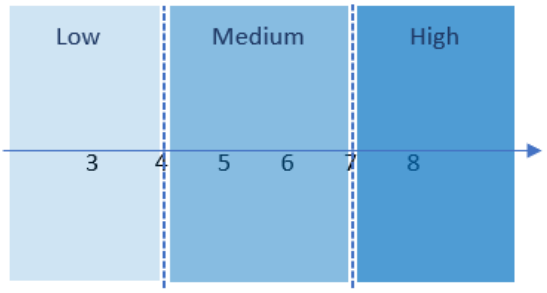

Figure 1.3 Labels redefinition

So , up to 4 , the quality of the red wine will be lower. From 5 to 7 , the quality will be reputed medium , and for labels greater than 7 the quality of the corresponding wine will be evalutated as high.

After done this label modification , we can procede to train our classifiers , and we can compare the results with the previous obtained without label redefinition.

Predictor	Training	Test
SVM	0.99	0.95
DTC	0.96	0.94
KNN	0.95	0.95

Table 1.5 Classifiers Accuracy.

As we can see from **Table 1.5** , after this modification , the accuracy of the single classifiers increased by 20-30% respect to the results reported in **Table 1.2** . In particular , as in the previous case , the SVM classifiers are the ones that give the best results both in training phase and on the validation data.

The previous classifiers have been obtained considering all the eleven features of the original dataset. What we want to do now is to reduce the number of feauters taken in consideration , without loosing a lot on the accuracy of the corresponding classifiers , with the aim to obtain simpler models , and to reduce as much as possible the computational time required to train our predictors. To this end we will use techniques like PCA (Principal Components Analysis) and SBS (Sequential Backward Selection) for dimensionality reduction. Let's start with PCA extraction algorithm.

The PCA , is a techniques to features extraction , with the aim to reduce the complexity of the models. We present briefly the algorithm

PCA STRATEGY

Input : X , k
Output: W
1: Find the Features Covariance Matrix C
2: Find the Eigenvalues & Eigenvectors of C
3: Select the first k Eigenvectors corresponding to the first greatest k Eigenvalues.
4: Build W by using the Eigenvectors obtained at point 3

But , before using the PCA Strategy , we have to choose the number k of components that we want to use in the next step. To this end , we plot the explained variance ratios , defined as

$$\frac{\lambda_i}{\sum_{i=1}^{n} \lambda_i}$$

where λ_i refers to the i-th eigenvalue of the covariance matrix C , and the corresponding cumulative sums , as depicted in **Figure 1.4**

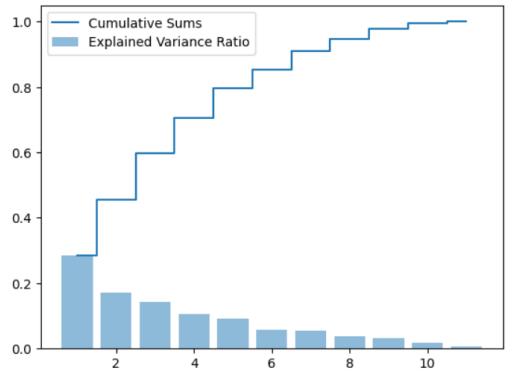

Figure 1.4 PCA Analysis

As we can see from **Figure 1.4** , considering the first 4 components (relative to the greatest eigenvalues) we have over 70% of the information brought by the data. So we procede by choosing k = 4 , and so we pass from 11 features to a space of dimension 4. **Table 1.6** report the result obtained before and after PCA , for the SVM classifier.

Classifier	Training	Validation
SVM	PCA	0.96	0.95
SVM		0.99	0.95

Table 1.6 Results PCA analysis