ASI Assessment : Bonus Question

Introduction

The paper *Modeling wine preferences by data mining from physicochemical properties* exploits exactly the same database we used during our work. Thus, it is a great opportunity to see the real usage of such a data in real research : in this report we will try to enlighten the knowledge we got from the course and our understanding through our work during the assessment.

Machine Learning methods

Three algorithms have mainly been used during this study (This paper uses a lot of other similar studies as reference). They mainly compare neural network and support vector machine.

K-Nearest Neighbours

These one is not the most represented in the paper. However, we've been asked to implement it in the assessment. Actually, a 3-nearest neighbour algorithm is used to tune a Gaussian kernel SVM algorithm. The predictions are used to define heuristically a variance of reference.

Neural Networks (NN)

They are using a one hidden layer perceptron. One of the big question is : how many nodes in my hidden layer ? For this, we train a NN for each value of H in a finite set (from 0 to H_{max}), and we measure the performance through the *generalization estimate*, we will try to maximize over our set of value (it's cross-validation).

By the way, using one layer could appear relevant : it already gives nice predictions and limits the computation due to the backpropagation on a neural network with more hidden layers.

SVM

It's the second big method used in this paper after neural networks. The Gaussian kernel has been used. The paper let guess one of the main reason of this choice : it is because of the popularity of the kernel. There are three parameters to fix : γ, ε, and C. ε is defined thanks to the 3-nearest neighbour algorithm we discussed above. C has been set as C=3. The interesting part is to set γ. Indeed, we try to find the value of γ which maximizes the *predictive estimate* by cross validation. In conclusion of the paper, this algorithm finally outperforms all the other methods.

Validation

In order to quantify the performance of their algorithms, they use the *mean absolute deviation*. In the assessment, we used the *mean squared error*. My preference goes to this second solution which highly penalizes the big mistakes.

The predictions are made using an *absolute error tolerance* T. If the absolute error between a predicted value and the expected value is smaller than this treshold : the prediction is correct and the point is given the predicted value as a class. Else, we look for the closest class to our prediction.

This allows to build the *regression error characteristic* curve (0/1 Loss given T) and the *confusion matrix* given T. In the assessment, we also used such tools : 0/1 Loss and confusion matrix. They allowed us to have a critical sight about the performances of our algorithms.

Feature Selection

It's the part which is the most missing in our assessment : we are given a dataset with 11 features. But who knows if the « free sulfur dioxide » feature is really relevant and discriminant when we will train our algorithm ? We could need to select the good features, we also could need to know the weight of each feature which will be relevant in the algorithm.

Backward Selection

Iteratively, we compute the relative importance of each feature dimension and we remove the less important. We iterate until we reach the wanted dimension of our feature space. This wanted dimension is defined by k-fold cross validation.

Conclusion & Criticism

This paper gives an overview on the methods applied to a concrete subject : remodeling of the feature space, choice of the hyperparameters, training of the algorithm, and validation. However, « It works » or « It is common to do this way » seem to be the main justifications to the most of the choice made in this paper : I find it a bit superficial.

In term of result, the paper give between 60 et 70 % of accuracy in the case of our dataset, which is really good because we got similar results during our assessment. We have not been so wrong !