EE3-23-Machine Learning: Coursework

Cao An Le
Imperial College London
Kensington, London SW7 2AZ
caoan.le15@imperial.ac.uk

1. Project Introduction

1.1. Subject

This document proposes a data mining and machine learning solution aimed at predicting human wine taste preference as a rating on a scale of 0 to 10. The experiment considers a dataset of red and white variants of the "*Vinho Verde*" wine (1599 and 4898 samples respectively), a product from the Minho region of Portugal.

1.2. Set-up

This *supervised learning* problem may be approached as regression or multiclass classification problem; here, the classification procedure is employed.

Results displayed in the form of figures or tables in this document are generated by Python code using the following libraries:

- `numpy`
- `scipy`
- `matplotlib.pyplot`
- `pandas`
- `seaborn`
- `sklearn`

The loss function for this problem is chosen to be the *Mean Squared Error* (MSE) defined as:

$$MSE = \frac{1}{N}\sum_{n=1}^{N}\left[h(\mathbf{x}_n) - f(\mathbf{x}_n)\right]^2 \qquad (1)$$

where N is the number of predicted points, h is the final hypothesis, f is the target function, and $\mathbf{x}_n$ represents the data points. The MSE is chosen since we wish to penalize bigger prediction errors accordingly [1].

[1] e.g. if the target valuet is 5, predicting 7 is worse than predicting 6.

2. Data Preprocessing

2.1. Visualisation

The available dataset consists of 6497 samples of both red and white wines, with 11 physicochemical properties as features. The matrix of features is thus denoted as $\mathbf{X}_{6497\times 11}$ and the output indicating the quality of wine samples is the column vector $\mathbf{y}_{6497\times 1}$. In order to obtain a more meaningful representation and better insight in the nature of the dataset, basic statistical quantities [2] of the individual features are shown in Table 1, and a histogram showing the samples' quality distribution is plotted in Figure 1.

Property	Min	Max	Mean	Std
fixed acidity	3.80	15.90	7.22	1.30
volatile acidity	0.08	1.58	0.34	0.16
citric acid	0.00	1.66	0.320	0.15
residual sugar	0.60	65.80	5.44	4.76
chlorides	0.01	0.61	0.06	0.04
free sulfur dioxide	1.00	289.00	30.53	17.7494
total sulfur dioxide	6.00	440.00	115.74	56.5218
density	0.99	1.04	0.99	0.003
pH	2.72	4.01	3.23	0.16
sulphates	0.22	2.00	0.53	0.15
alcohol	8.00	14.90	10.49	1.19
quality	3.00	9.00	5.82	0.87

Table 1. Dataset Statistics

2.2. Transformation

Before applying any learning model, a few changes have been made to the dataset.

Firstly, an appropriate *train/test split* with a 9:1 ratio has been performed in order to train and test the model independently. This is a crucial step since in real-world applications, the incoming data onto which the supervised learning model is applied is and remains unknown. The train/test

[2] minimum, maximum, mean, standard deviation

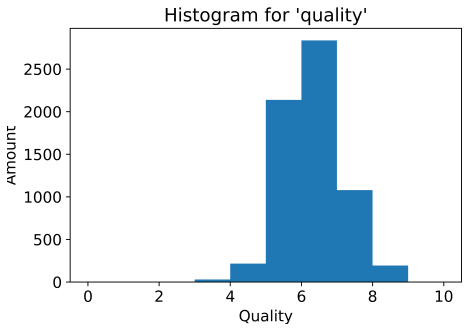

Figure 1. Histogram of the quality of the wine dataset

split therefore constitutes a simple method to simulate this. The choice of keeping an important majority of the dataset for training is supported by the latter use of *K-fold Cross-Validation*, where the training set is again partitionned into K subsets, used as "pseudo-test sets", in order to obtain a more accurate estimate of the model's out-of-sample performance.

Secondly, samples considered as *outliers* have been removed from the dataset. In this experiment, outliers are defined as samples whose feature values deviate by at least 4 times the feature standard deviation from the feature mean. This reduced the number of data points down to 6306.

Lastly, feature values in the training and test sets[3] have been *standardized* such that each feature has zero mean $\mu = 0$ and unit standard deviation $\sigma = 1$ (i.e. they have properties of a standard normal distribution). This technique gets all of the data on the same scale; ensuring standardised feature values implicitly weights all features equally in their representation. This is done through the following formula:

$$z = \frac{x - \bar{\mu}}{\bar{\sigma}} \tag{2}$$

where x is the sample feature value, and $\bar{\mu}$ and $\bar{\sigma}$ are the sample mean and standard deviation of the feature.

3. Learning Models

3.1. Baseline Classifier: the Perceptron

The selected baseline predictor for the wine dataset is the *perceptron*, arguably one of the oldest and simplest algorithm in supervised learning for linear classification problems. This algorithm linearly combines a set of weights $\mathbf{w} \in \mathbb{R}^{11}$ with the columns of $\mathbf{X}$ in order to make a prediction on $\mathbf{y}$, and therefore works perfectly if the data is *linearly separable* [4]. For instance, for binary classication where only two classes are considered this reduces to:

$$h(\mathbf{x}) = \text{sign}\left(\sum_{i=0}^{d} w_i x_i\right) = \text{sign}\left(\mathbf{w}^T \mathbf{x}\right) \tag{3}$$

The perceptron also generalizes to multiclass classification, and hence can be used to solve this problem.

Running the *Perceptron Learning Algorithm* (PLA) gives the results shown in Table 2. The numbers show that less than half of the wine samples' quality in the test set are predicted correctly, indicating the limited performance of PLA for this dataset. This also reveals that the wine dataset is most likely not linearly separable.

	% Error	**MSE**
$\hat{R}_n(h)$	53.45%	0.82
$R(h)$	52.69%	0.81
5-Fold Cross-validation	55.47%	$\times$

Table 2. Perceptron Performance

Two other algorithms, are therefore considered: the *K-Nearest Neighbors* (KNN) and the *Random Forest* algorithms.

3.2. K-Nearest Neighbors

In supervised learning, the KNN algorithm is a non-parametric method used for classification and regression. For classification, a new sample assigned to the class most common among its k-nearest sample neighbors. For example, if k = 1, then the new sample is simply assigned to the class of that single nearest neighbor. A popular choice for the distance metric is the *Euclidian Distance*, defined as:

$$d(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{\sum_{l=1}^{d} \left(x_{il} - x_{jl}\right)^2} \tag{4}$$

where $\mathbf{x}_j$ is a point to be classified, and $\mathbf{x}_i$ is a point among the k-nearest neighbors.

In order to choose the best value of the parameter $\mathbf{k}$, cross-validation has been performed in a loop where, at each iteration, the out-of-sample accuracy of the algorithm was estimated. Figure 2 shows that the model yields the best performance when k is equal to one. Thus, setting $k = 1$, and running the KNN learning algorithm on the dataset, the obtained results are shown in Table 3.

It can be concluded that the KNN algorithm is a much better algorithm than the PLA for the wine quality prediction, as the error measures are considerably lower.

An interesting point to make is that despite being another simple model, the KNN obtains a much better score, which

[3] $\mathbf{X}_{train}$ and $\mathbf{X}_{test}$ resulting from train/test split on $\mathbf{X}$.

[4] some hyperplane(s) can separate all the points of different classes.

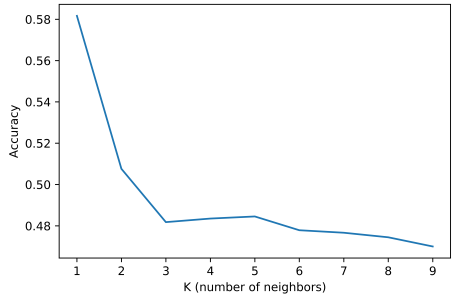

Figure 2. Accuracy of the KNN algorithm VS number of neighbors

	% Error	MSE
$\hat{R}_n(h)$	0.00%	0.00
$R(h)$	36.67%	0.619
5-Fold Cross-validation Error	38.07%	$\times$

Table 3. KNN Performance

proves that the model complexity is not a conclusive factor for a model's performance. In fact, it is probably more likely that the KNN algorithm is better suited to the nature of the data points.

3.3. Random Forest

The Random Forest model is an extention of the *Decision Tree* algorithm. In the Decision Tree algorithm, the feature space is split into homogeneous sets based on most significant differentiators in input variables. A metric considered when performing the splits[5] is the *Entropy*. It is a quantity that measures the uncertainty of having correctly-classified points in the regions split, and it is therefore aimed to be minimized. Mathematically, entropy is defined as:

$$Entropy = -\sum_{i=1}^{c} p_i \log_2 p_i \tag{5}$$

where c is the number of distinct classes in the region and p_i the probability of finding a certain class i in the region.

In the Random Forest algorithm, the trees or splits are built progressively from a successive random selection of K data points, instead of the whole dataset.

In a similar way as for KNN, cross-validation is again employed to estimate the out-of-sample performance of the Random Forest algorithm for different numbers of trees considered (see Figure 3). The optimal number of trees is found to be 58, since this model configuration yields the highest accuracy score according to cross-validation. Table 4 shows the prediction performance of the Random Forest algorithm with the number of trees parameter set to 58.

[5]in orther words, when creating trees.

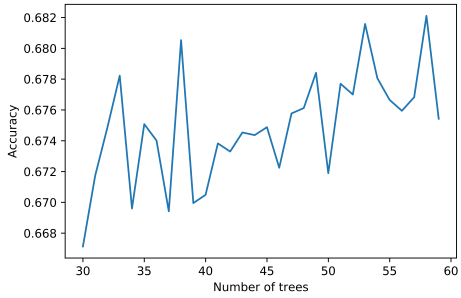

Figure 3. Accuracy of the Random Forest algorithm VS the number of trees

	% Error	MSE
$\hat{R}_n(h)$	0.02%	0.00017
$R(h)$	29.36%	0.397
5-Fold Cross-validation Error	32.55%	$\times$

Table 4. Random Forest Performance

4. Conclusion

Bar charts in Figure 4 summarise all of the algorithms and their performances on the wine dataset.

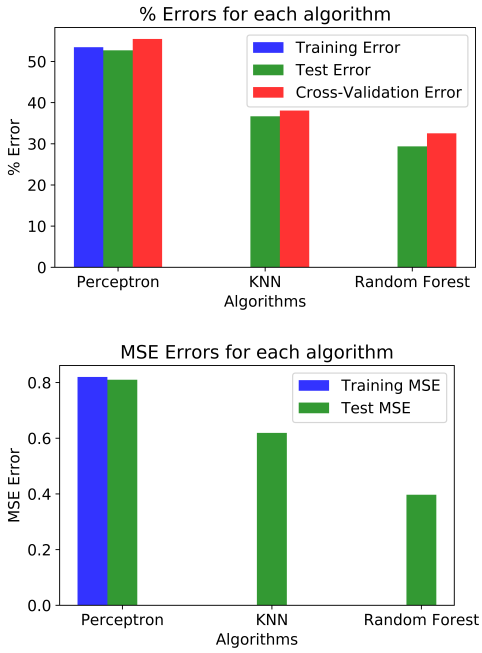

Figure 4. Bar chart sumarizing the performance of the 3 studied algorithms

It is clear that the Random Forest algorithm yields the lowest error measures and it can therefore be concluded that

it delivers the best performance for predicting wine quality of red and white *Vhino Verde* samples.

5. Pledge

I, Cao An Le, pledge that this assignment is completely my own work, and that I did not take, borrow or steal work from any other person, and that I did not allow any other person to use, have, borrow or steal portions of my work. I understand that if I violate this honesty pledge, I am subject to disciplinary action pursuant to the appropriate sections of Imperial College London.