Comparing kNN and Logistic Regression algorithms for Wine classification

T-61.3050 Term Project, final report

Fábio Pinheiro 472735 & João Duro 472191

Abstract

The idea of the project and this report is to see how different kNN and Logistic Regression are. Our main problem divides itself in two sub problems of supervised classification, with one being a binary supervised classification and the other a 7-class supervised classification, and we will see how both algorithms behave in each case.

I. Introduction

The data we're given consists of a set of measures from 6000 different wines. The attributes of the wines are "fixedAcidity", "volatileAcidity"," citricAcid", "residualSugar", "chlorides", freeSulfurDioxide, "totalSulfurDioxide", "density", "pH", "sulphates", "alcohol", "quality" and "type". Our first goal is try and predict the type of the wine, which is classified as "White" or "Red, and later, try and predict based on the same attributes the quality of the wine which is a integer variable ranging from 1 to 7, 1 being really good, and 7 being really poor. Based on the results and our understanding of how the algorithms work, we will try and see how different it is to predict a binary class or a 7-value class.

So, which one of our methods is better to classify the wine type or wine quality?

II. Methods

k-Nearest Neighbors algorithm (kNN)

K-nearest neighbors algorithm also known as kNN is a non-parametric method used for classification and regression. The kNN algorithm starts from the principle that similar data are closed one to another.

The algorithm works in a very simple way: given a point 'Y' we find the 'k' nearest points from the training data Then choose the most frequent class among them, and in case of a tie, pick one at random.

The pseudo-code for kNN can be seen in Algorithm 1 [2]

Euclidean Distance

For n dimensions Euclidean Distance is give by following formula:

$$d(p,q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + ... + (p_n - q_n)^2}$$

(Dimensions in here is the number of feature/parameters of the data)

Logistic Regression

Logistic regression is similar to others regressions. It models the relationship between a dependent and one or more independent variables, and allows us to look at the fit of the model as well as at the significance of the relationships (between dependent and independent variables) that we are modeling.
The method estimates the probability of an event occurring. What we want to predict from a knowledge of relevant independent variables

Algorithm 1 kNN

1: Define the distance measure or similarity between two objects
2: Find k;
3: Compute the distance between the new object and all objects in the training set: $d(xi, x0)i = 1, 2, ..., n$;
4: Sort the distances in increasing numerical order and pick the first k elements (neighbors), let be $Vk(x0) \subseteq D$, the set of those neighbors;
5: Save the classifications of all neighbors;
6: Assign new object to the class based on majority vote of its neighbors 2, i.e.

$$\mathcal{Y}_0 = \overset{arg\ max}{C_j} \sum_{(x_i, y_i) \subset V_(x_0)} I(y_i = C_j)$$

is not a precise numerical value of a dependent variable, but rather the probability that it is 1 rather than 0.

$$P = \frac{e^{\alpha+\beta x}}{1 + e^{\alpha+\beta x}}$$

Where P is the probability of a The value of α yields P when x is zero, and β indicates how the probability of a 1 changes.

III. Experiments

The first thing we need to do is to separate our data consisting of 6000 row elements in training and validation set, and a test set, which will be used exclusively to predict the error of our methods. We spited the data so we have 1000 elements in the test set, and the remaining data will be used to train our model or to make assessments about the data, this will not be used to predict the error of our models.

A preliminary analysis shows us the data is not normalized, so when applying both algorithms this will play a huge role, but we decided to test it as it is, and then normalize the data and see how much it improves, if it improves, and we believe it will.

The procedures we took to predict the quality and type of the wine were the same, except when predicting the quality we used the values predicted for wine type to see if would improve it somehow.

Our first task is to predict the wine type since it will be more accurate being just a binary problem. When applying the kNN we first had to predict what k (the number of neighbors) to use. We did this separating the training set (consisting of the 5000 elements) into 3750 training elements, and 1250 validation elements. We ran the kNN algorithm with k going from 1 to 20, recorded the errors and took conclusions from that. We then did the same for the standardized data.

To apply the logistic regression we just used the whole training set as one, and didn't split it. Predicted the type with non normalized data, it was time to use a the normalized approach, and the procedure was the same.

The logistic regression was basically the same, we normalized the variables using the 6000 elements, and made a prediction. With the kNN, we first normalized the 5000 elements of the training set (without the quality and type) and tried to get the best K for the problem. After that, we joined both the training set and test set, without the quality and type variables, and normalized it, and we made the prediction for the type.

As said before, when predicting the quality type using either algorithm, we tried using the wine type to see if the results improved in some way.

IV. Results

Data:

- Number of parameters: 11
- Number of samples in training set: 5000
- #Red: 906
- #White: 4094
- #Quality1: 5
- #Quality2: 161
- #Quality3: 854
- #Quality4: 2197
- #Quality5: 1604
- #Quality6: 159
- #Quality7: 20
- Number of samples in test set: 1000

kNN:

Table 1: *Result summary for kNN*

Experiment	Accuracy
Type (k=4)	95.3%
Type (scaled & K=1)	99.4%
Quality (k=12)	42.8%
Quality (scaled & k=1)	63.4%

Table 2: *Confusion matrix for type using kNN with k=1 and data scaled*

Type	White	Red
White	798	6
Red	0	196

In the confusion matrices the columns represent the prediction values and the line are real values. For example: In table 2, six white wines have been predicted to be red.

Table 3: *Confusion matrix for quality using kNN with k=1 and data scaled*

Quality	1	2	3	4	5	6	7
1	0	0	0	0	0	0	0
2	0	13	7	5	1	0	0
3	0	9	114	49	9	0	0
4	0	6	44	272	78	8	0
5	0	0	11	99	225	9	0
6	0	0	3	11	12	10	0
7	0	0	0	1	3	1	0

Logistic Regression:

Table 4: *Result summary for Logistic Regression*

Experiment	Accuracy
Type	99.6%
Quality	51.3%
Quality using type	51.8%

Table 5: *Confusion matrix for type using Logistic Regression*

Type	White	Red
White	802	2
Red	2	194

Table 6: *Confusion matrix for quality using Logistic Regression*

Quality	1	2	3	4	5	6	7
1	0	0	0	0	0	0	0
2	0	0	13	13	0	0	0
3	0	0	42	129	10	0	0
4	0	0	19	312	77	0	0
5	0	0	3	177	164	0	0
6	0	0	0	12	24	0	0
7	0	0	0	2	2	1	0

V. Discussion

Regarding the use of the kNN algorithm we can confirm our initial assumption that if we normalized the data we would have better results. As seen in table 1, we can say that there's

a huge improvement in the accuracy both for wine prediction and quality prediction. We can see this happen in part because the algorithm uses euclidean distance, which is very sensible to the scales of the variables we're using. The results we have for the binary problem are much better than for the 7 class problem as we also predicted, but that's most always like that, since separating data into more classes is harder than if we have to choose between just two classes. We used Figure 1 to choose the best k to use in the kNN algorithm. One thing to notice using kNN in a binary classification problem, is that if we choose an odd number of neighbors the accuracy will always be the same, because the predictions will always be the same. That doesn't happen if we choose an even number for the number of neighbours because when finding the k nearest neighbors, we can (and most likely will sometimes) find an equal number of different labels, and as the algorithm states, we have to choose one at random. This will make the predictions change each time we run the algorithm, and with it, the accuracy. But in general, with data as big as we have, the results tend to stable and not vary that much.

Comparing the results from the two algorithms , we see that for predicting the wine type, the logistic regression is more suitable, as we have better accuracy than the kNN algorithm even when using scaled data on the last one. On the other hand, the logistic regression algorithm doesn't seem to work as well when the number of predicting classes increases, and in this case, using kNN with scaled that gives us way better accuracy than the logistic regression algorithm. Overall, we get really good results. Of course the binary problem has in all cases better accuracy, since its a simpler problem. Looking at both confusion matrices for the wine quality we noticed a very interesting fact. Both algorithms, even when falling to classify the exact class, don't classify it very far away from the real class. Since the quality of the wine is an ordered scale, predicting a wine has having quality "2" when in fact has quality "1" or "3" in not has bad has classifying it has having quality "7". So, even when failing the prediction, our algorithms don't fail by that much. One last point on the poor results of the wine quality is that their labels are very subjective, since they're based on human taste and opinion, so it is not very precise, it depends on the people grading them, so that might have an affect on the results, and we believe it has , at least to some extend.

For computational resources: In terms of space, the kNN algorithm have to keep all training data to be able to predict new labels, as opposed to the Logistic Regression that after the pre-processing and training of the algorithm, we don't need to keep the training data. In terms of processing the Logistic Regression have all the work in the pre-processing, the kNN does not have any pre-processing but for every prediction have a high cult try to find the k closer values.

For future analysis we could try different distances for the kNN algorithm, and try to normalize the data for the Logistic Regression.

References

[1] Alpaydin, Ethem - Introduction to machine learning, MIT press, 2004.

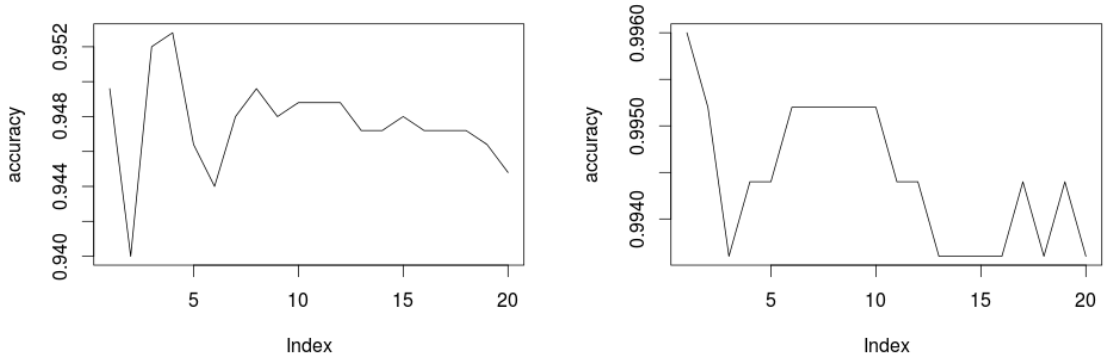
[2] Conceição, Amado - Lecture Slides: Statistical Methods in Data Mining (2014). Instituto Superior Técnico, Portugal.

[3] Duro, João & Pinheiro, Fábio - Project code of Machine Learning Basic Principles, `https://github.com/FabioPinheiro/MachineLearningBasicPrinciples-T-61.3050`

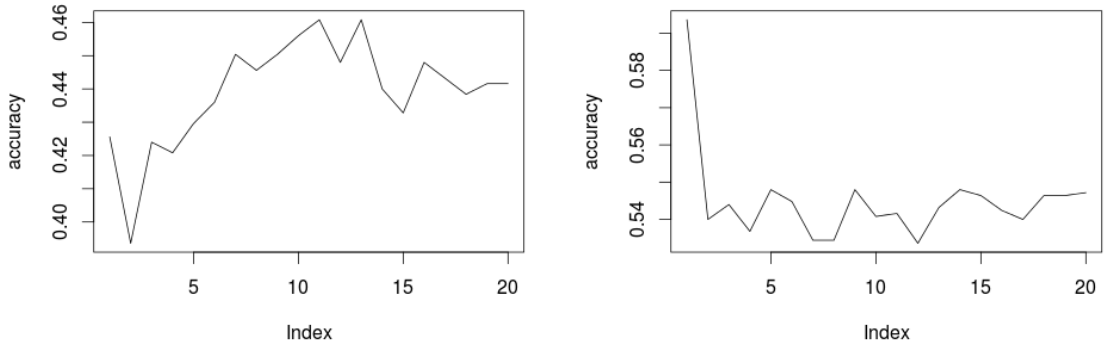
[4] Howe, Bill - k Nearest Neighbors, `https://class.coursera.org/datasci-001/lecture/161`

[5] Ng, Andrew - Machine Learning. Multiclass Classifier: One-vs-All, `https://class.coursera.org/ml-005/lecture/38`.

Appendix



(a) *kNN - Type (With out and with the data scaled)*



(b) *kNN - Quality (With out and with the data scaled)*

Figure 1: *Find k for the kNN*