EE3-23 INTRODUCTION TO MACHINE LEARNING

Predicting wine ratings

Keidi Kapllani

KK2314 /

1. Abstract

The aim of this coursework is to find and implement a machine learning model to train datasets of red and white variants of the Portuguese "Vinho Verde" wine in order to predict the wine rating as accurately as possible. We are given a dataset containing 11 inputs of objective tests (e.g. PH values) [2] and one target rating, which is based on sensory data(median of at least 3 evaluations made by wine experts). We will begin by analyzing the datasets and then proceed to choose an appropriate loss function, establish a baseline predictor and try to implement more advanced algorithms in order to increase the performance .

2. Introduction

2.1. Preprocessing the dataset

Going through the datasets we notice firstly that the inputs have all vastly different ranges which can, depending on the model we are going to choose, increase the error in prediction. However in the research paper [1] for which this dataset has been originally used, the authors argue that although that is a clear limitation it is more appropriate to construct the baseline predictor without modifying the datasets and then proceed to consider modifications to improve accuracy which is what I will do.

Differently from the research paper though I will consider the two datasets jointly by adding a features which describes if the wine is red or white, namely a features which is 1 for red and 0 for white wines. The reasoning for this assumption is based on the fact that white and red wines differ in taste and as such the same parameters can not yield same ratings for both. Having a new parameter which differentiates the two allows for the models to distinguish between red and white. Nonetheless we will run models for combined and separate datasets.

Quotation marks were remove from parameter names to facilitate reading in MATLAB.

2.2. The Loss Function

In order to choose an appropriate loss function we need to consider the context of the usage of the model we will select. Suppose the model will be used by a company/firm which sells and markets wines. In this context two aspects are important.

The first is the distance of the predicted rating to the target, as it is more disadvantageous for the company to have a much higher/lower predicted wine than the actual target *e.g. target is a 5 rated wine and predicted an 8 rated wine*. The second is whether the model predicted higher or lower than the target output. The reasoning behind this for this choice is the same as for the "credit approval" example discussed in lectures, where a higher prediction then the target is worse then a lower prediction since a costumer buying a 8 rated wine and realizing afterwards it's actually of lower quality is more hurtful to the company/firm then a costumer which bought 5 rated wine and discovered that it taster better then expected.

To implement a loss function with these properties we use a Mean Squared Error(MSE) with an added weight depending on whether $(\hat{\mathbf{y}} - \mathbf{y})$ is positive or negative:

$$\text{LF} = \frac{1}{n}\sum_{i=0}^{n}(\hat{\mathbf{y}}_i - \mathbf{y}_i)^2 \times (\text{sgn}(\hat{\mathbf{y}}_i - \mathbf{y}_i) + 2) \quad (1)$$

where $\hat{\mathbf{y}}$ is the vector of predicted ratings, $\mathbf{y}$ is the vector of target ratings, n is the number of samples and a shifted sgn() function which adds a weight of 3 for higher then target prediction and a weight of 1 for lower then target prediction.

3. Baseline Predictors

3.1. Linear Regression

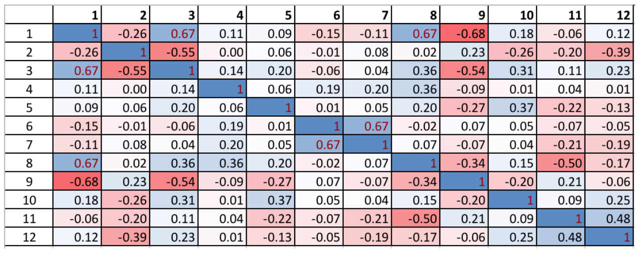
The baseline predictor we are going to adapt is the linear regression model. Although it is highly unlikely that we will produce a reasonable error, the linear algorithm provides a good baseline to compare against. We run the linear algorithm, which is implemented using the `fitlm()` MATLAB function, on the datasets combined and separate and produce a table of the error in each case measured according to our loss function defined in Section 2.2 and the misclassification error which is how many points it predicted incorrectly over the whole number of points. The linear regression model is implement with 20% of randomly selected dataset for testing and 80% for training. The results are shown in Table 1.

	Combined	White wine	Red wine
Error	1.34	1.31	1.20
Misclass.	48%	48%	42%

Table 1. Error table for linear regression

It is clear that the white wine dataset has in general a higher classification error and loss error. This seems to contribute to and increased error when we attempt to train from the combined dataset. In the next sections we will analyze the dataset more extensively and try to improve its quality.

3.2. Analyzing the data



	1	2	3	4	5	6	7	8	9	10	11	12
1	1	-0.26	0.67	0.11	0.09	-0.15	-0.11	0.67	-0.68	0.18	-0.06	0.12
2	-0.26	1	-0.55	0.00	0.06	-0.01	0.08	0.02	0.23	-0.26	-0.20	-0.39
3	0.67	-0.55	1	0.14	0.20	-0.06	0.04	0.36	-0.54	0.31	0.11	0.23
4	0.11	0.00	0.14	1	0.06	0.19	0.20	0.36	-0.09	0.01	0.04	0.01
5	0.09	0.06	0.20	0.06	1	0.01	0.05	0.20	-0.27	0.37	-0.22	-0.13
6	-0.15	-0.01	-0.06	0.19	0.01	1	0.67	-0.02	0.07	0.05	-0.07	-0.05
7	-0.11	0.08	0.04	0.20	0.05	0.67	1	0.07	-0.07	0.04	-0.21	-0.19
8	0.67	0.02	0.36	0.36	0.20	-0.02	0.07	1	-0.34	0.15	-0.50	-0.17
9	-0.68	0.23	-0.54	-0.09	-0.27	0.07	-0.07	-0.34	1	-0.20	0.21	-0.06
10	0.18	-0.26	0.31	0.01	0.37	0.05	0.04	0.15	-0.20	1	0.09	0.25
11	-0.06	-0.20	0.11	0.04	-0.22	-0.07	-0.21	-0.50	0.21	0.09	1	0.48
12	0.12	-0.39	0.23	0.01	-0.13	-0.05	-0.19	-0.17	-0.06	0.25	0.48	1

Figure 1. Correlation coefficients for Red Wine dataset

Having established a baseline we are now going to analyze the datasets in order to be able to choose the appropriate models to improve the error measure and misclassification. Firstly we compute the correlation of both red and wine dataset to spot any features which maybe correlated, which would be a factor in increasing our error. In Figure 1 we see the table of correlation coefficients for the red wine data. We can spot from the color-coding of the table that parameter pairs (1,9), (1,8), (7,6) and (1,3) all have correlation coefficients higher then 0.6. For the white wine dataset[1] the pairs with high correlation correspond to (8,11), (8,4) and (6,7).

Although high multicollinearity among predictor variables does not prevent good, precise predictions of the response it can have an adverse effect on the marginal contribution of any one predictor variable in reducing the error sum of square. Therefore we will try to choose models which are relatively insensitive to predictor multicollinearity.

We also consider the distribution of the target variables, plotted in Figure 2. The histograms present with the shape of a normal distribution, which implies higher frequency of samples in the mid-ranges and lower in the extremities. This class imbalance needs to be taken into account when choosing more complex models as well.

Lastly, by scanning through the data files it is clear that the predictor variables are not in the same range nor do they have the same variance so some method of data normalization should be considered.

3.3. Features Scaling

In order to tackle the difference in range of out raw data we will use features scaling on all out input parameters in order to set them all in a range $[-1, 1]$ while preserving the sign. This should allow for a better generalization error in the linear model. Furthermore we will also standardize the data to mean 0 and standard deviation 1.

4. Ridge Regularization

One way in which we can improve the improve the generalization error and loss error is by implementing ridge regularization to our linear regression. Since ridge attempts alleviates multicollinearity amongst regression predictor variables in a model it is a reasonable idea to test out the effect it would have on our dataset.

We implement the ridge with the MATLAB function `fitrlinear()` with a 5-fold cross-validation to avoid over-fitting and stochastic gradient descent as our solver. The function is iterated for varying number of λ from 10^{-2} to 10^{-9} on a maximum iteration number of 1000. The best model is then selected from the algorithm and used for testing. The test is performed with `kfoldPredict(model)` which predicts the response of the trained model to new features from the dataset not used in training. The resulting vector of predictions is than rounded and error measures are performed on it, whose results are displayed in Table 2.

	Combined	White wine	Red wine
Error	1.24	1.30	0.99
Misclass.	46%	48%	41%

Table 2. Error table for linear regression with ridge regularization

The only significant improvement is on the Loss Error of the red wine dataset. This could mean a that it is performing better in terms of over-predicting, *i.e.* predicting higher then target, although is it still misclassifying the same amount the samples.

5. Support Vector Machines

The results from the baseline predictor and the ridge regularization suggest that the we cannot rely on a linear predictor to have a good generalization. Thus we take into

[1]Correlation coefficient matrix for white wine is located in the appendix

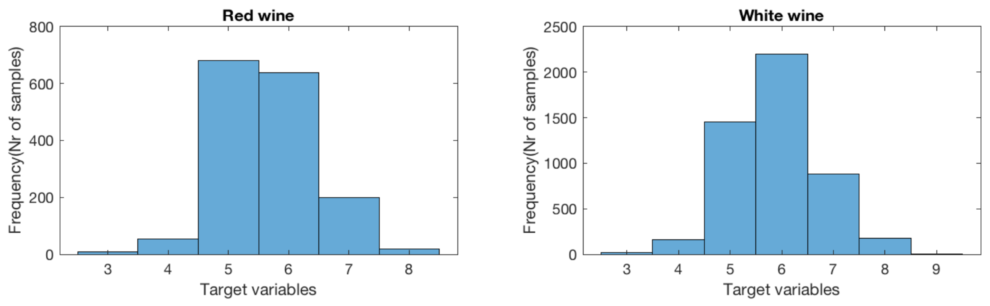


Figure 2. Histogram of red and wine target variables

account the soft-margin Support Vector Machines which efficiently perform non-linear classification by implicitly mapping the input to higher dimensions. Although it is commonly used for binary classification we can adapt it to perform multi-class classification by using ECOC(error-correcting output codes) which reduces the problem of classification with three or more classes to a set of binary classifiers [3]. We implement it using `fitecoc` with 5-fold cross-validation and using parallel computing toolbox on MATLAB to increase computational speed. Also we use Hyper Parameter Optimization in ”all” mode so the algorithm optimizes all eligible Hyper Parameters for SVM. Errors are then computed and displayed in Table 3.

	Combined	White wine	Red wine
Error	1.12	1.16	0.92
Misclass.	46%	47%	41%

Table 3. Error table for SVM with ECOC

Again we get an improvement across all the datasets in term of Loss Error but we are still misclassifying the same amount of points. This is possibly due to the class imbalance present in out target variables which makes it difficult for the models to correctly predict the classes with less samples such as ratings 3,8 and 9.

6. Neural Networks

The last attempt to increase our prediction performance will be by using Neural Networks. Being very powerful tools they are especially useful in this scenario since they have an embedded feature selection algorithms which sets non-useful features to 0.

To implement Neural Network on MATLAB we use `nprtool`, which is a tool that allows you to generate scripts to implement neural networks. With it we generate the script for a single neural network and then proceed to convert it to a MATLAB function in order to test the networks on all out datasets. We set the parameters to use damped least-squares (DLS) method for training, which is commonly used to solve non-linear least squares problems. The validation checks, *i.e.* number of times the validation error does not change before stopping, is set to 6 with a maximum of 1000 iterations. Also the outputs are reformatted so that they are present in 7 columns, representing the 7 output classes, with 1s for each instance when the rating occurs and 0 for the rest. We run the Neural Network models iteratively varying H, number of hidden layers, to iterate from 5 to 25 and then plot the results for varying layers in Figure 3 and Figure 4. It is immediately obvious that

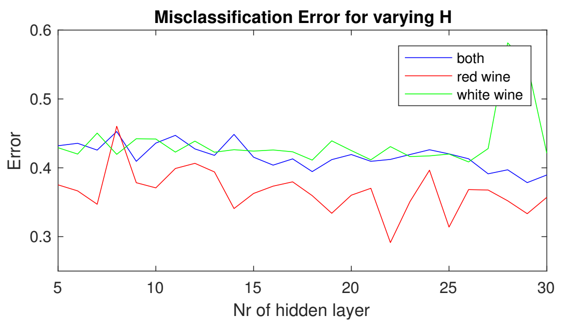


Figure 3. Misclassification Error for varying H

this model performs significantly better then all previously tested models. We also notice that we get the lowest loss error and classification error between 20-25 Hidden layers.

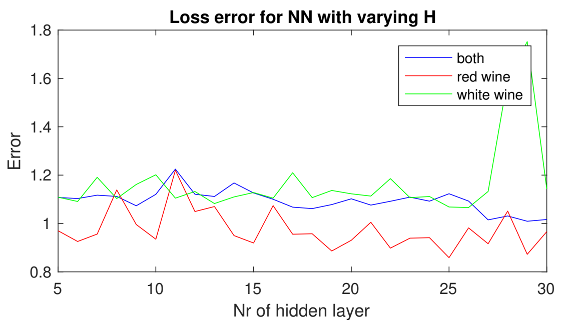


Figure 4. Loss Error for varying H

	Combined	White wine	Red wine
Error	1.00	1.06	0.85
Misclass.	37%	40%	29%

Table 4. Minimum Errors for Neural Networks

When we compare values from Table 3, SVM errors, to Table 4, minimum NN errors, we can see that the there is an improvement of about 0.12 in all errors with the lowest errors still being from the red wine dataset and an improvement of 7-9% in classification error with the smallest still being red wine.

7. Conclusion

In conclusion encouraging results were achieved when we take into account the large class imbalance in our targets and the missing samples samples for very high or very low ratings *i.e*. 9 and 10 or 1 and 2 ratings. The red wine is where out predictors were generally more accurate(29% with NN) suggesting that further curating might need to be done on the white wine data. Furthermore Neural Networks were the most effective model in predicting out datasets, outperforming SVM by a large margin. The combined data performed slightly worse then red wine which might be a results of the problems in the white wine dataset.

A proposed improvement of my approach could be using some form of Feature Selection, since we observed high correlation between some of our predictor variables, like Principal Component Analysis which would reduce the dimensionality of data while retaining the variation present amongst samples.

Note

I, Keidi Kapllani, 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.

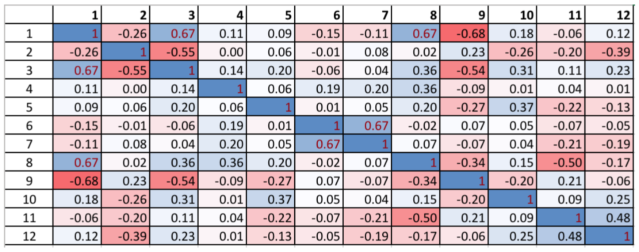
References

[1] C. et al. Modeling wine preferences by data mining. 2009. Elsevier, 47(4):547-553. ISSN: 0167-9236.

[2] https://archive.ics.uci.edu/ml/datasets/spambase. Uci archive. Accessed on 2018-03-11.

[3] https://uk.mathworks.com/help/stats/fitcecoc.htmlbufm33c-. Mathworks. Accessed on 2018-03-19.

Appendix



	1	2	3	4	5	6	7	8	9	10	11	12
1	1	-0.26	0.67	0.11	0.09	-0.15	-0.11	0.67	-0.68	0.18	-0.06	0.12
2	-0.26	1	-0.55	0.00	0.06	-0.01	0.08	0.02	0.23	-0.26	-0.20	-0.39
3	0.67	-0.55	1	0.14	0.20	-0.06	0.04	0.36	-0.54	0.31	0.11	0.23
4	0.11	0.00	0.14	1	0.06	0.19	0.20	0.36	-0.09	0.01	0.04	0.01
5	0.09	0.06	0.20	0.06	1	0.01	0.05	0.20	-0.27	0.37	-0.22	-0.13
6	-0.15	-0.01	-0.06	0.19	0.01	1	0.67	-0.02	0.07	0.05	-0.07	-0.05
7	-0.11	0.08	0.04	0.20	0.05	0.67	1	0.07	-0.07	0.04	-0.21	-0.19
8	0.67	0.02	0.36	0.36	0.20	-0.02	0.07	1	-0.34	0.15	-0.50	-0.17
9	-0.68	0.23	-0.54	-0.09	-0.27	0.07	-0.07	-0.34	1	-0.20	0.21	-0.06
10	0.18	-0.26	0.31	0.01	0.37	0.05	0.04	0.15	-0.20	1	0.09	0.25
11	-0.06	-0.20	0.11	0.04	-0.22	-0.07	-0.21	-0.50	0.21	0.09	1	0.48
12	0.12	-0.39	0.23	0.01	-0.13	-0.05	-0.19	-0.17	-0.06	0.25	0.48	1

Figure 5. Correlation coefficients for Red Wine dataset

Explanation of MATLAB functions

`data2matrix` - reads the data from the .csv files and imports them into MATLAB arrays. It also add the extra parameters to determine white or red

`linear_cross` - performs linear regression with randomized data.

`make_NN_array` - return two arrays, one of features and one of outputs formatted adapt Neural Networks.

`neural_net` - trains and test from the input data.

`rescale_feat` - Performs rescaling of the features it is inputted

`ridge_reg_cross` - performs ridge regularization

`svm` - performs SVM on input data

NOTE: All plots are generated in the kk2314.m file which serves as main and calls all the above functions. Functions are not implemented in the same order as discussed in the report.