Systems with Machine Learning

Task 3: Training

Topic: Wine Quality Analysis

1. Problem

Considering the demands of wine safety testing and the complexity involved in the numerous laboratory analysis and residue monitoring, we are keen at providing real-world solutions to the problem of wine quality classification and prediction using ten wines classes.

For the given problem we identified two methods for solving the problem, regression or classification. Both methods fall within the category of supervised ML. Regression helps to predict or explain a particular numerical value based on a set of our prior data, Classification methods predict or explain a class value. Regression algorithms attempt to estimate the mapping function from the input variables to numerical or continuous output variables. On the other hand, Classification attempts to estimate the mapping function from the input variables to discrete or categorical output variables.

Our dataset consists of integer quality grades for each wine so we decided to solve this problem as a classification problem. Each grade from 0-10 corresponds to one glass of wine. In such a way we mitigate the problem of getting a result of wine quality that is a floating-point number. E.g. Quality = 5.1423

For calculating a loss value we are going to use a categorical cross-entropy function. Cross-entropy loss, or log loss, measures the performance of a classification model whose output is a probability value between 0 and 1. Cross-entropy loss increases as the predicted probability diverges from the actual label. So predicting a probability of .012 when the actual observation label is 1 would be bad and result in a high loss value. A perfect model would have a log loss of 0.

In binary classification, where the number of classes M equals 2, cross-entropy can be calculated as:

$$-(log() + (1-)log(1-))$$

If M>2 (i.e. multiclass classification), we calculate a separate loss for each class label per observation and sum the result.

$$-\sum_{c=1}^{M} y_{o,c}\, log(p_o, c)$$

M − *number of classes* (dog, cat, $fish$)

log − *the natural log*

y − *binary indicator* (0 *or* 1) *if class label is the correct classification for observation*

p − *predicted probability observation is of class*

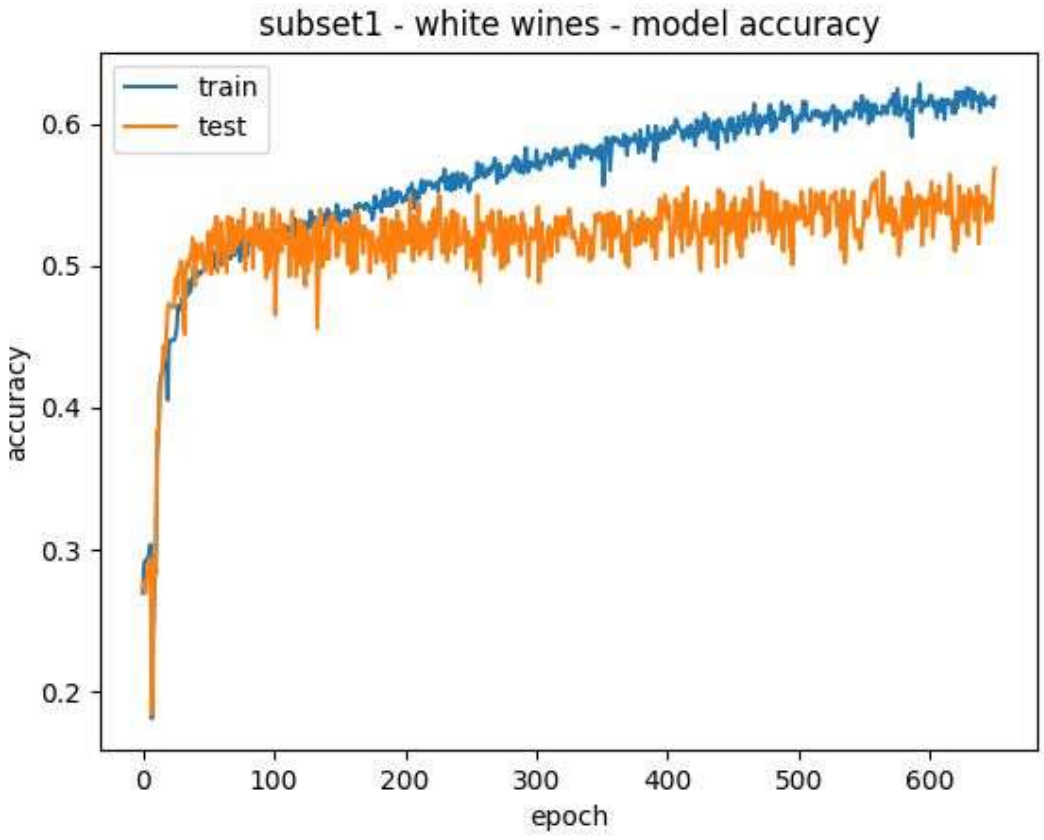
2. Training of SPLIT1

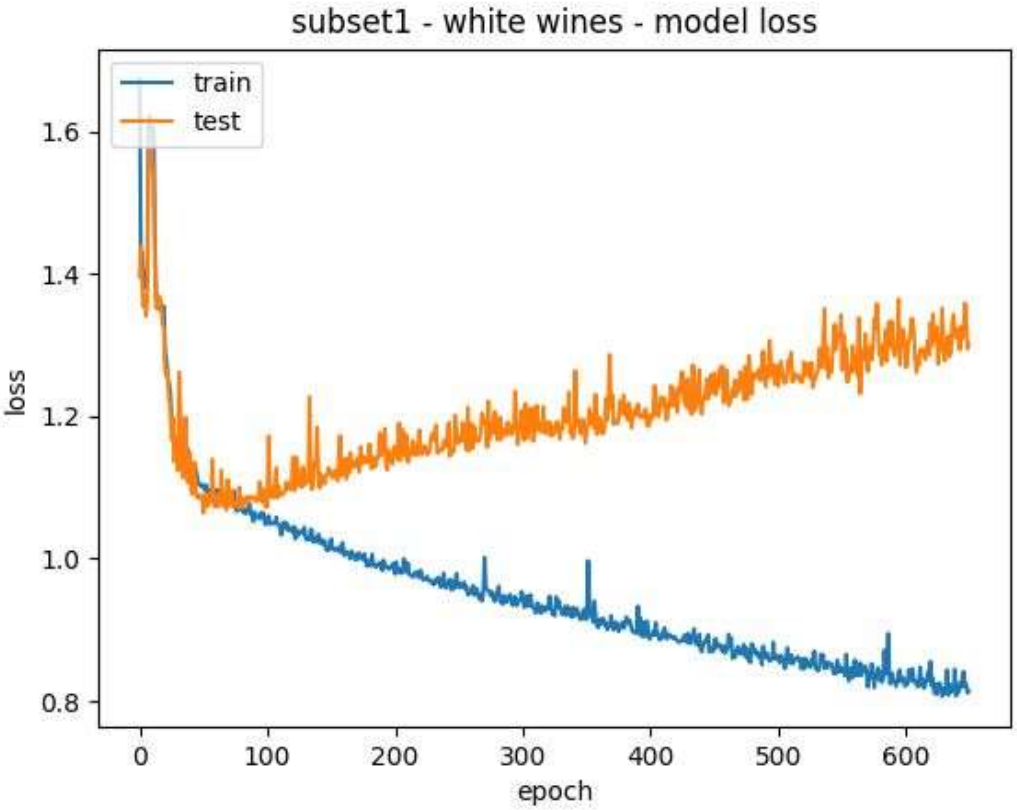
Split 1 consists of three disjunctive subsets TRAIN, VAL, TEST in the ratio of 70%, 15%, 15%. The data is NOT normalized.

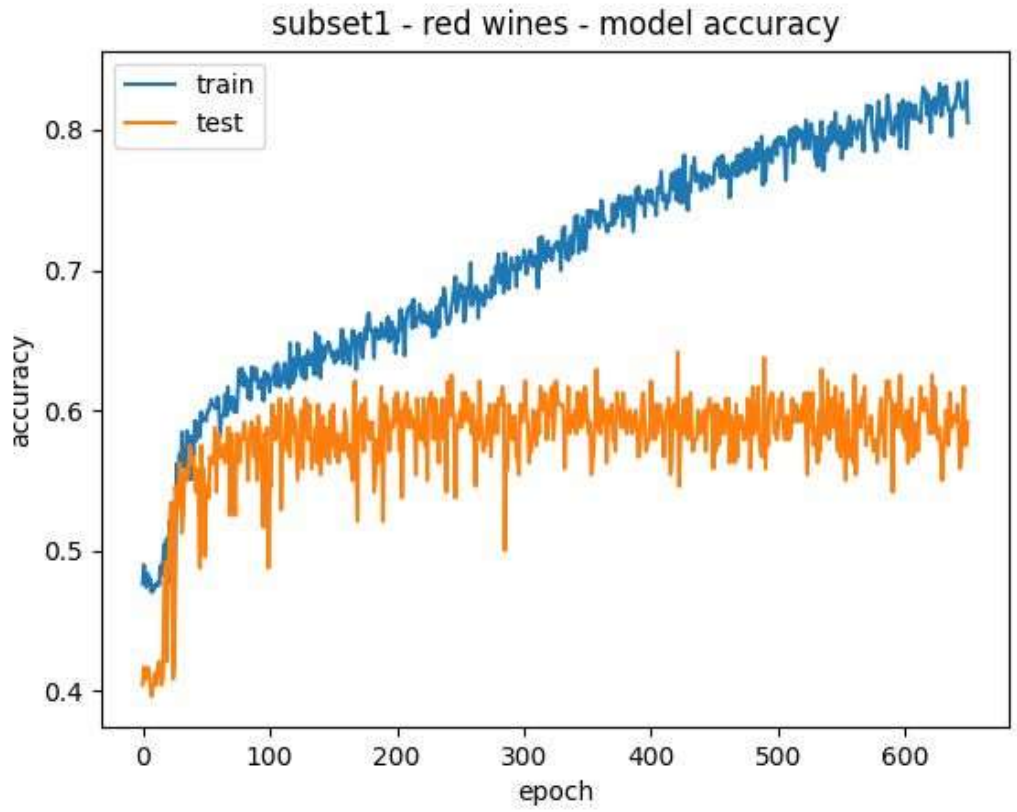
The training didn't bring any good results as we expected. The accuracy of the training set hits a maximum of 60% for white wines and 80% for red wines. The loss value for the training set decreases down to 0.8 and 0.4 for white and red wines respectively. Surprisingly the loss value for the validation set rises during the training.

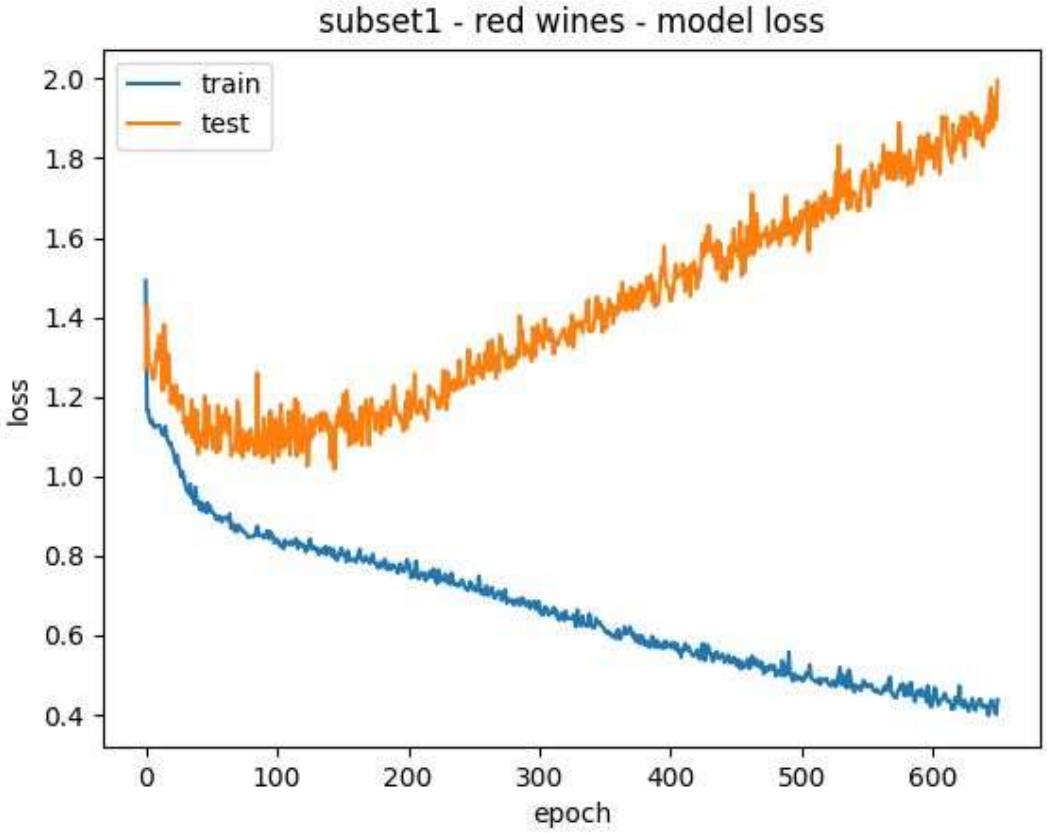
The learning plots for red and white wines are depicted below:

P.S. Please disregard the wrong label on the plots. The orange line is the characteristic of the VAL subset rather than TEST.




subset1 - white wines - model loss
train
test
loss
epoch
1.6
1.4
1.2
1.0
0.8
0
100
200
300
400
500
600


subset1 - red wines - model accuracy
train
test
accuracy
epoch
0.8
0.7
0.6
0.5
0.4
0
100
200
300
400
500
600

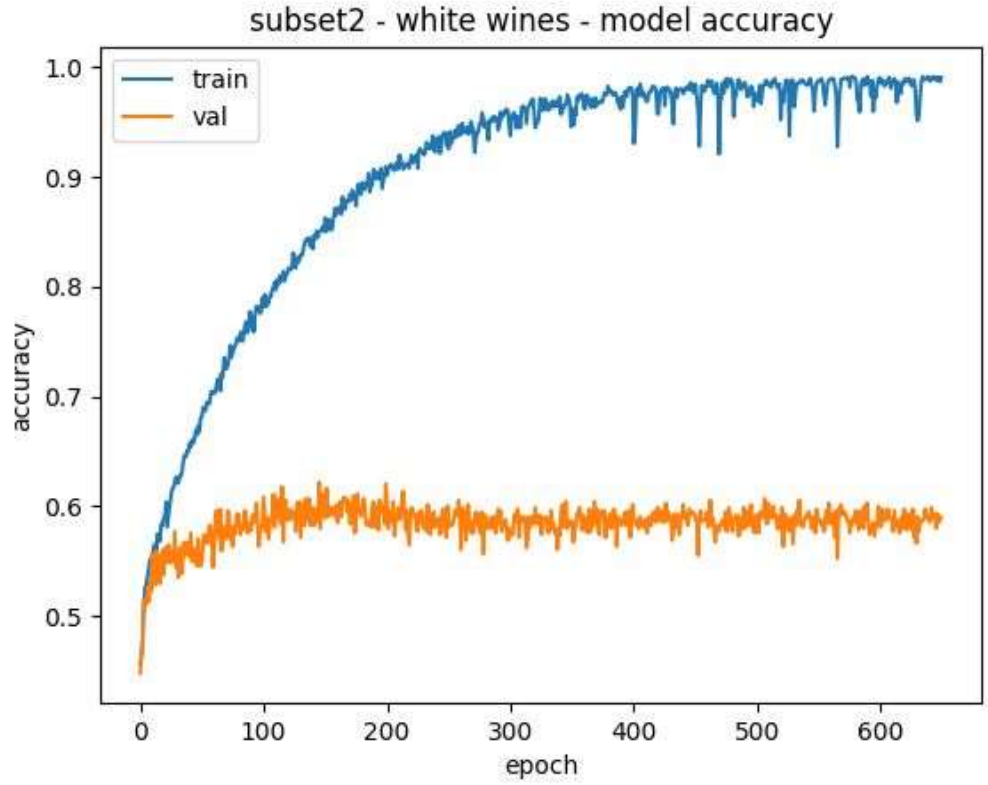


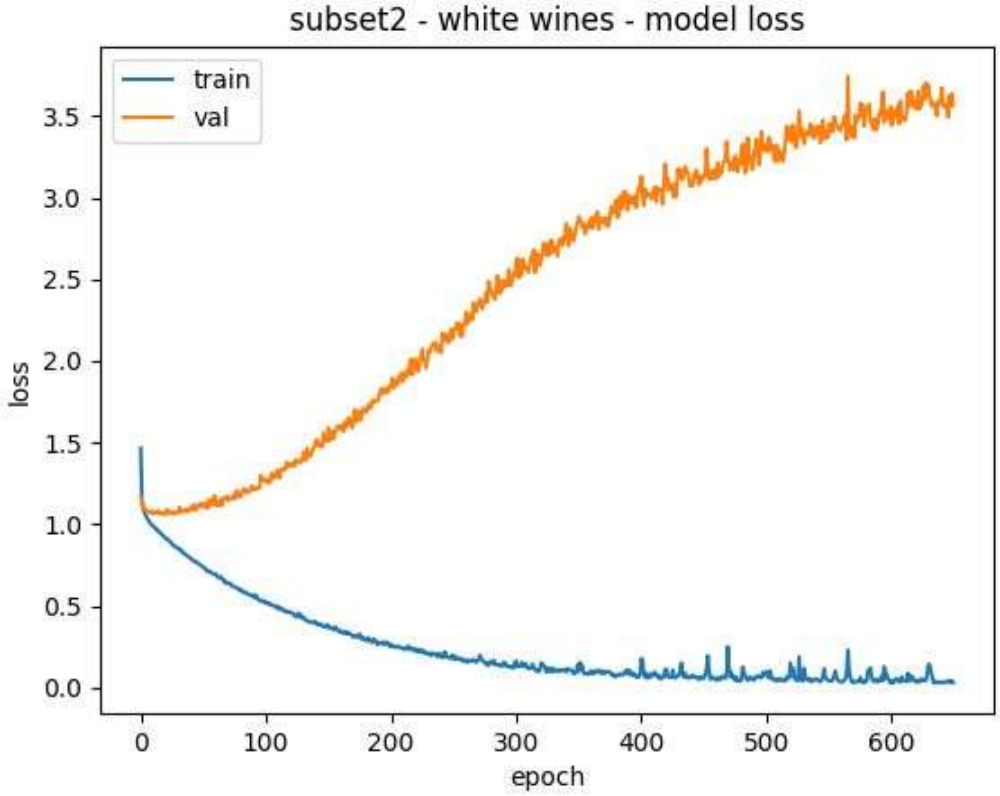
3. Training on SPLIT2

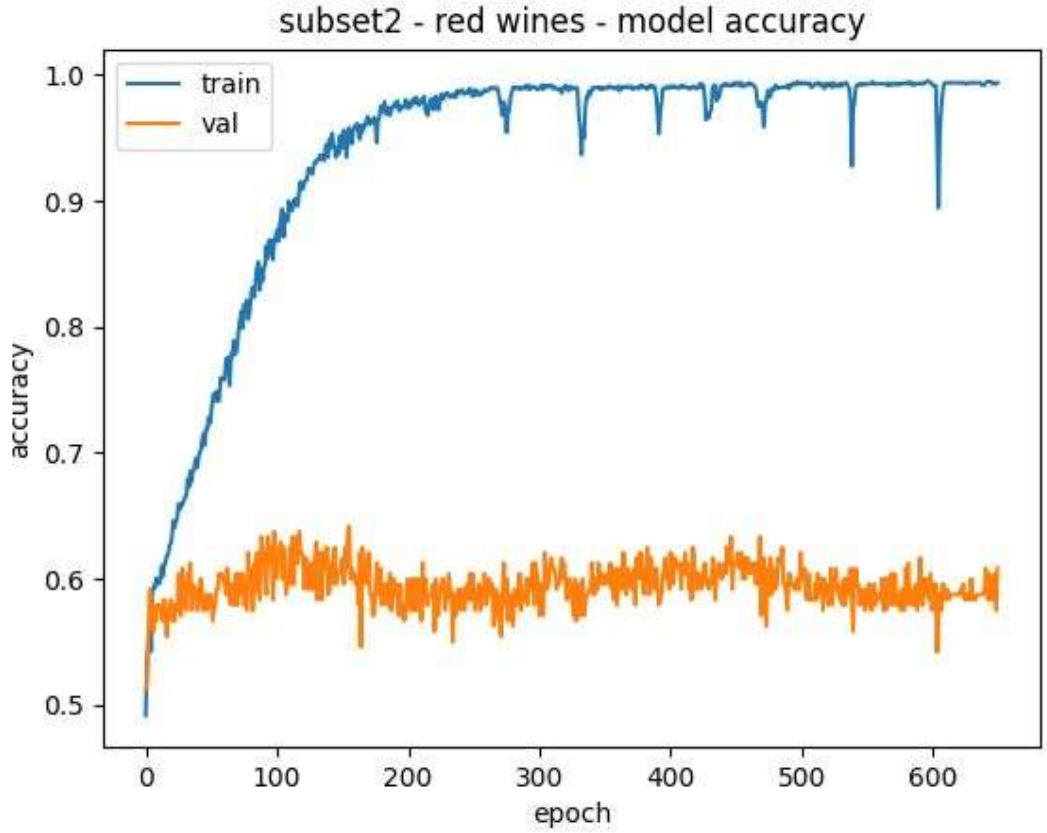
On the contrary to the split 1, split 2 consisted of normalized data. The split ratio for TRAIN, VAL, TEST stayed the same.

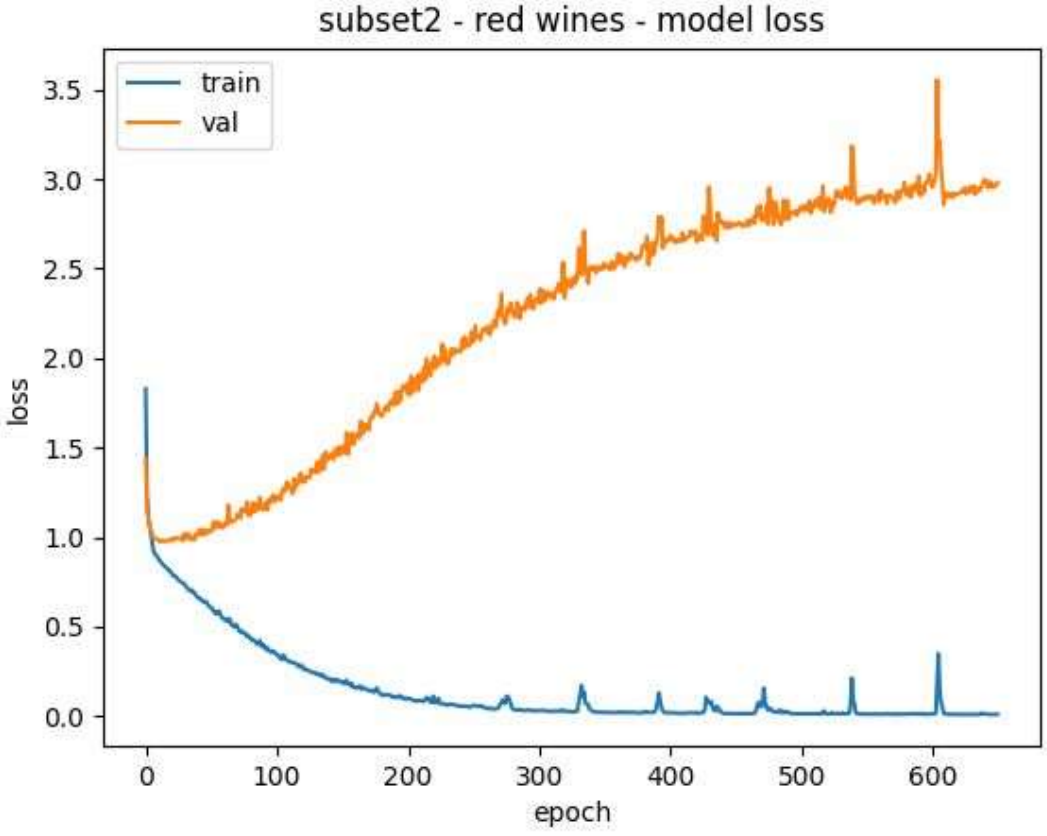
After the normalization, the neural network fitted very well hitting accuracy of nearly 100% for white and red wines. Unfortunately, accuracy for validation sets only rose up to 60% and fluctuated around this value.

The training could have ended after 250 epochs, but it was prolonged to 600 epochs just to see how the VAL subset would behave. The observed trend is that the loss function values go up during the training - opposite to expected behavior.


subset2 - white wines - model accuracy
train
val
accuracy
epoch
1.0
0.9
0.8
0.7
0.6
0.5
0
100
200
300
400
500
600

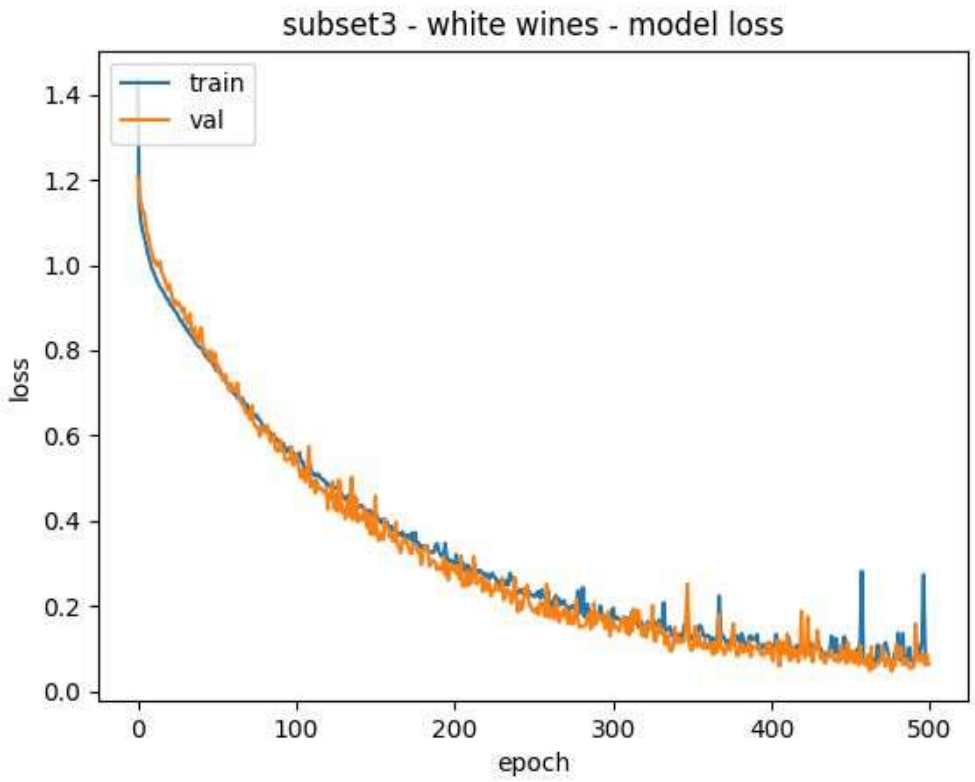

subset2 - white wines - model loss
train
val
loss
epoch
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0
100
200
300
400
500
600

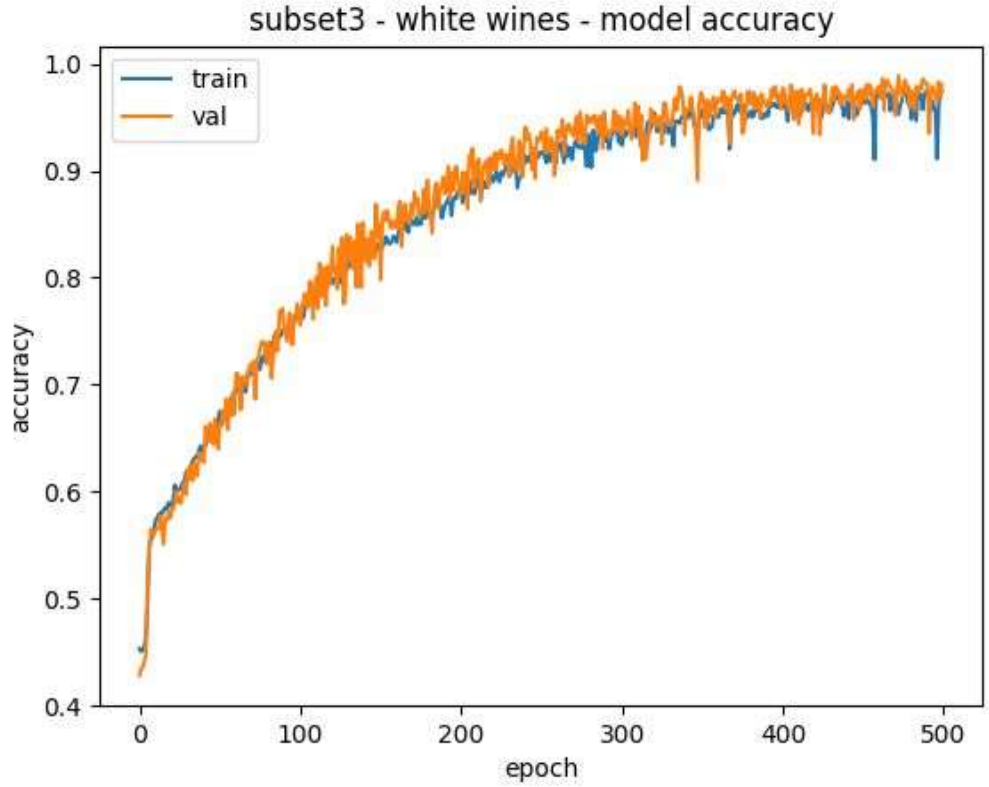

subset2 - red wines - model accuracy
train
val
accuracy
epoch
1.0
0.9
0.8
0.7
0.6
0.5
0
100
200
300
400
500
600

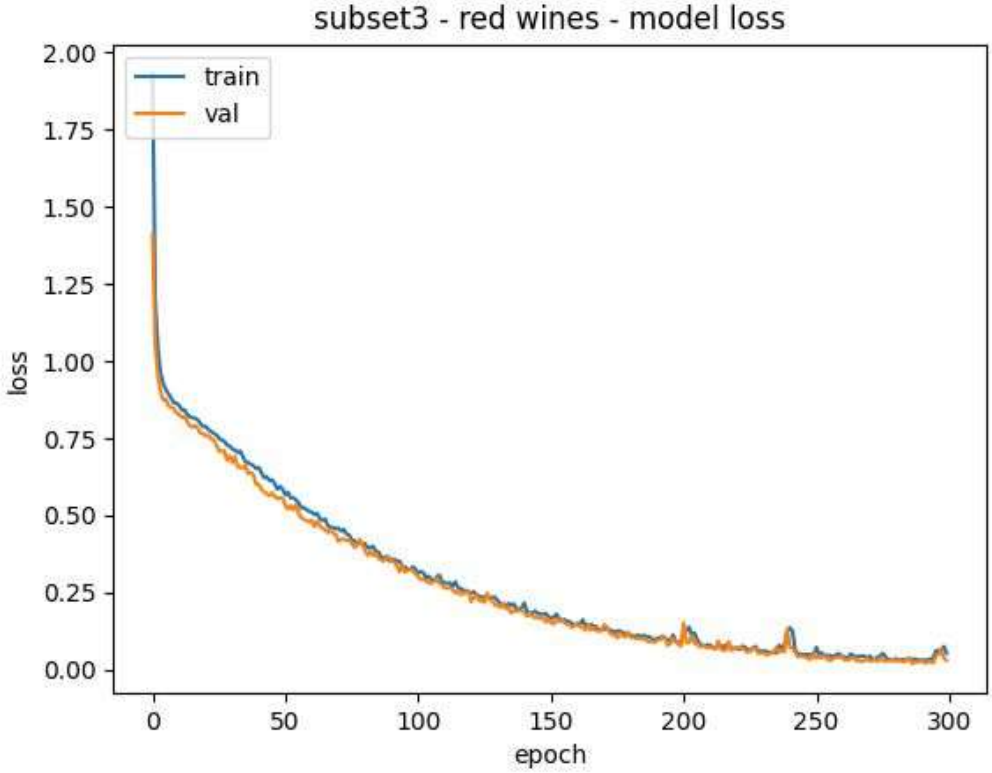

subset2 - red wines - model loss
train
val
loss
epoch
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0
100
200
300
400
500
600

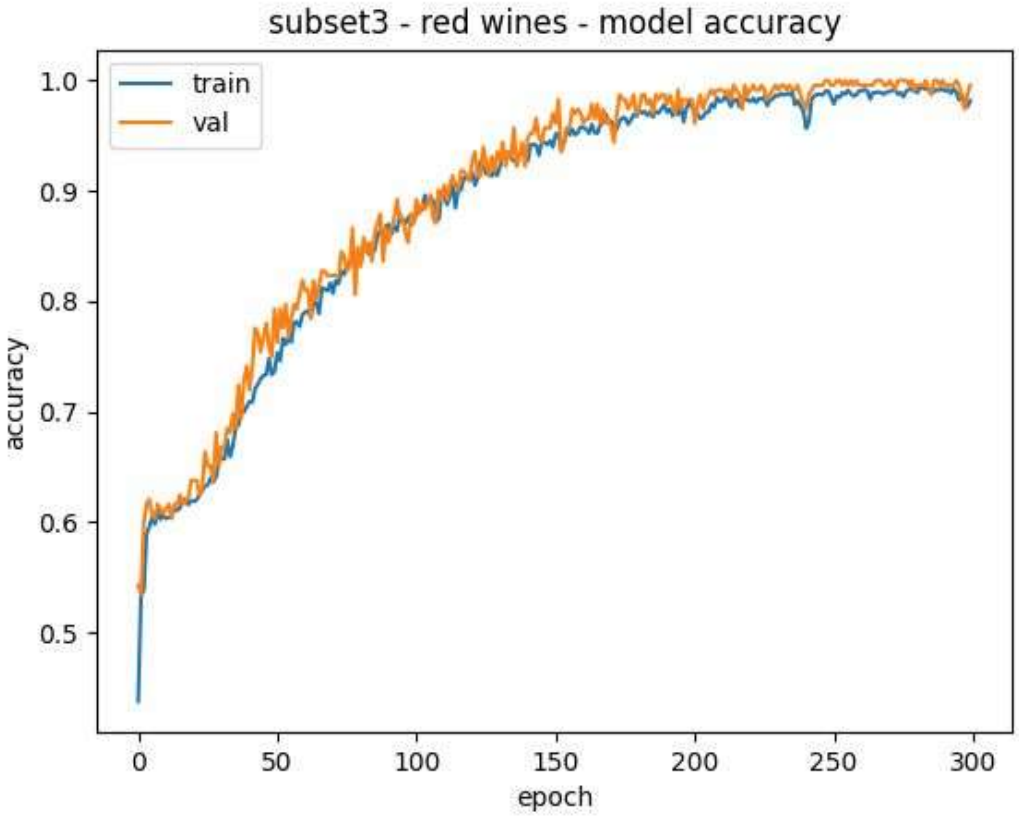
4. Training on SPLIT3

The split number 2 also contained normalized data but in this case, the VAL subset was also a subset of the TRAIN subset. This scenario provided satisfying results. The neural network behaved as we expected. The VAL and TRAIN set acted in the same manner. The accuracy approached the value of 100% and the loss went down to around 0.




subset3 - white wines - model accuracy
train
val
accuracy
epoch
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0
100
200
300
400
500


subset3 - red wines - model loss
train
val
loss
epoch
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
0
50
100
150
200
250
300



5. Conclusion

After the training and briefly examining the results of the TEST subset we can come to the conclusion that the wine quality classification may not be necessarily a problem for neural networks. As we can observe in the training of the SPLIT2 the network fits very well with the training data but it struggles to get a proper result for anything beyond the TRAIN subset. It may be caused by the subjective nature of the wine qualifying. The dataset consists of around 6500 ratings, which simply could not have been assessed by one person. Maybe in the dataset, there are some conflicting records that give totally different ratings for similar wines. Nevertheless, further analysis of the network efficiency will be conducted as a task 4 of this project.

From a technical point of view, we found out that we could combine red and white wines together by adding a binary attribute that would indicate a color, e.g. 0 - white, 1 - red. Ultimately, we decided to keep them separate as the network had a problem with classifying a single color of the wine.