Zhifan Sang
Assignment #1 CS4641

Supervised Learning Report

Dataset

Wine Quality. This data is taken from a wine certification and quality assessment study at the Minho (northwest) region of Portugal. There are two datasets in this profile.
Two datasets were created, using red and white wine samples. The inputs include objective tests (e.g. PH values) and the output is based on sensory data (median of at least 3 evaluations made by wine experts). Each expert graded the wine quality between 0 (very bad) and 10 (very excellent) The two datasets are related to red and white variants of the Portuguese "Vinho Verde" wine. Due to privacy and logistic issues, only physicochemical (inputs) and sensory (the output) variables are available (e.g. there is no data about grape types, wine brand, wine selling price, etc.).

The classes are ordered and not balanced (e.g. there are much more normal wines than excellent or poor ones). Outlier detection algorithms could be used to detect the few excellent or poor wines. Also, we are not sure if all input variables are relevant. So it could be interesting to test feature selection methods.

Since the red and white tastes are quite different, the analysis will be performed together, thus two datasets built with 1599 red and 4898 white examples were combined. We will combine two datasets and conduct analysis, and compare the results.

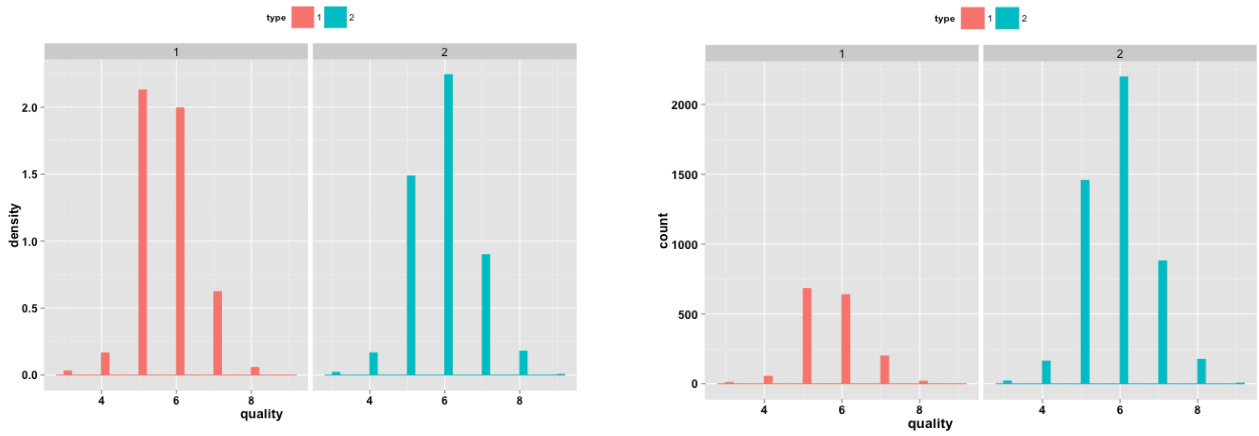


Figure 1 Histgraph of Wine Quality among different wine types(1=red wine, 2=white wine)

Some additional experiments were run on a set of the wine data with the wine type removed. These results are not reported on the paper although they are available in the experiment.

Why are these interesting datasets?

The dataset were interesting for its practical value of supervised machine learning. Wine quality is a critical attributes of red wine and white wine. The quality determine the value and price of the wine, which are consider the most important among dozens of wine's attributes. The biochemistry experimental results and wine type information of the Vinho Verde wine were considered as input in this study. Here white wine and red wine have to be tasted by at last three wine experts to assess the quality attributes. The process is time-consuming, expensive and subjective which could not be affordable for the mill to have all their wine product tested. However, if the training and testing methods in this supervised study yield desirable results, we could use the biochemistry experimental results to predict and validate the results of wine quality assessment.

In addition, this dataset is an ideal example for supervised machine learning. This

dataset consists of 13 attributes and 6497 instances, which is of reasonable scale of training and testing. Several attributes could be treated as target attributes to assess and predict. We may be interested in building a model to predict the quality of wine tested; we could clustered the samples in several ways to look it through; we could compare or predict wine types to see if there is a significant differences between red wine and white wine.

Decision Trees

```
> test #white wine quality decesion tree prediction
        quality.test
tree.pred   3   4   5   6   7   8   9
       3    0   0   0   0   0   0   0
       4    0   0   0   0   0   0   0
       5    1  22 180 140  19   4   0
       6    2   9  79 264 154  24   0
       7    0   0   0   0   0   0   0
       8    0   0   0   0   0   0   0
       9    0   0   0   0   0   0   0
> accuracy=sum(test[1,1]+test[2,2],test[3,3]+test[4,4],test[5,5],test[6,6],test[7,7])/sum(test)
> accuracy
[1] 0.4944320713
```

```
> test2##white wine quality pruned decesion tree prediction
        quality.test
tree.pred   3   4   5   6   7   8   9
       3    0   0   0   0   0   0   0
       4    0   0   0   0   0   0   0
       5    1  22 180 140  19   4   0
       6    2   9  79 264 154  24   0
       7    0   0   0   0   0   0   0
       8    0   0   0   0   0   0   0
       9    0   0   0   0   0   0   0
> accuracy2=sum(test2[1,1]+test2[2,2],test2[3,3]+test2[4,4],test2[5,5],test2[6,6],test2[7,7])/sum(test
> accuracy2
[1] 0.4944320713
```

```
> test #red wine quality decesion tree prediction
        quality.test
tree.pred   3   4   5   6   7   8
       3    0   0   0   0   0   0
       4    0   0   0   0   0   0
       5    1  10 159  74   5   0
       6    0  13  98 141  40   2
       7    0   0   1  19  31   5
       8    0   0   0   0   0   0
> accuracy=sum(test[1,1]+test[2,2],test[3,3]+test[4,4],test[5,5],test[6,6])/sum(tes
> accuracy #  0.5592654424
[1] 0.5525876461
```

```
> test2##red wine quality pruned decesion tree prediction
        quality.test
tree.pred   3   4   5   6   7   8
       3    0   0   0   0   0   0
       4    0   0   0   0   0   0
       5    1   8 145  52   0   0
       6    2  11 106 189  74  11
       7    0   0   0   0   0   0
       8    0   0   0   0   0   0
> accuracy2=sum(test2[1,1]+test2[2,2],test2[3,3]+test2[4,4],test2[5,5],test2[6,6])/sum(test2)
> accuracy2
[1] 0.5575959933
```

```
> test #red and white wine combined quality decesion tree prediction
        quality.test
tree.pred   3   4   5   6   7   8   9
       3    0   0   0   0   0   0   0
       4    0   0   0   0   0   0   0
       5    1  19 239  91   4   0   0
       6    6  29 252 564 242  48   2
       7    0   0   0   0   0   0   0
       8    0   0   0   0   0   0   0
       9    0   0   0   0   0   0   0
>
> accuracy=sum(test[1,1]+test[2,2],test[3,3]+test[4,4],test[5,5],test[6,6],test[7,7])/sum(tes
> accuracy
[1] 0.5364061456
```

```
        quality.test
tree.pred   3   4   5   6   7   8   9
       3    0   0   0   0   0   0   0
       4    0   0   0   0   0   0   0
       5    0  35 311 208  19   1   0
       6    4  17 180 450 231  39   2
       7    0   0   0   0   0   0   0
       8    0   0   0   0   0   0   0
       9    0   0   0   0   0   0   0
> accuracy2=sum(test2[1,1]+test2[2,2],test2[3,3]+test2[4,4],test2[5,5],test2[6,6],test2[7,7])/sum(te
> accuracy2
[1] 0.5083500334
```

Figure 2 The decision tree classification results

There

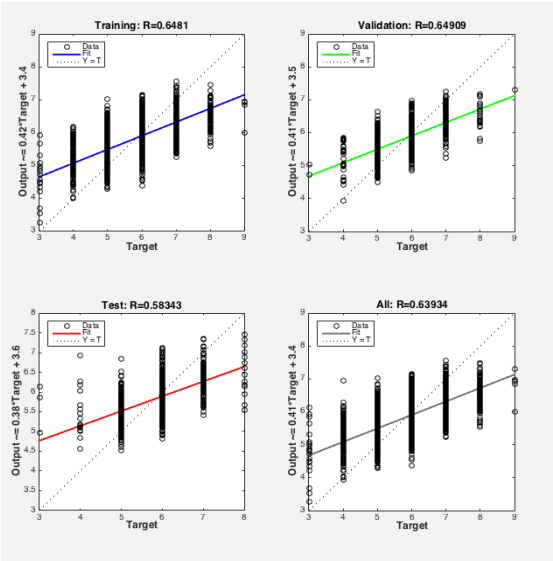
are several interesting features concerning the results of decision trees showed above. First of all, the pruning improved the accuracy only a little bit, but the improvement could be seen in all white wine, red wine, combined wine datasets. Among all these three datasets, the red wine dataset has the best decision tree classification results; the white wine dataset has the worst decision tree classification result; and the combined wine dataset has the median decision tree classification result as expected. In addition, we observed an interesting fact that the combined wine decision tree ignores more attributes than the other decision trees. As a result, the combined wine decision tree fails to identify extreme cases of wine quality. For instance, the samples whose quality greater than 6 could not be assessed in the overall decision tree. Pruned tree also ignored higher quality wine instances in the final results.

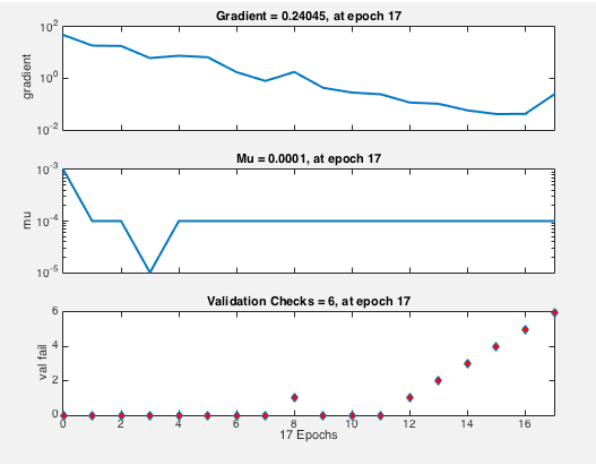
Low success rate of white wine could be the result of inappropriate input attributes to classify and predict the wine quality. As suggested in the related paper, the sources and ingredient of the wine is not available to researchers because of these attributes are considered as confidential information.

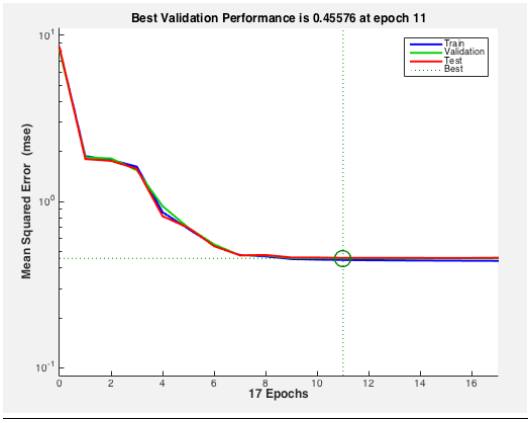
Note regarding the confusion matrix: The correct guesses fall on the diagonal. Incorrect guesses are placed off the diagonal.

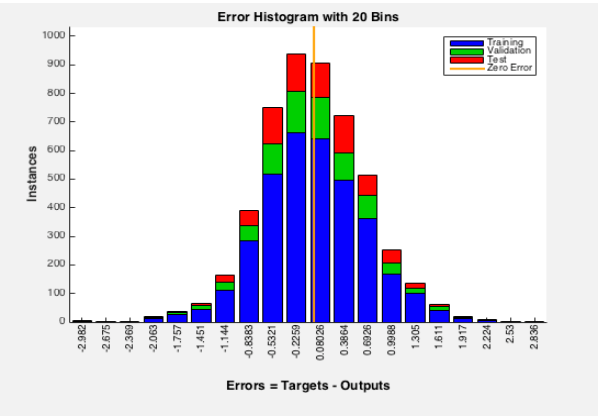
Neural Networks

The experiments that I ran regarding neural networks were quite simple and intuitive. In the experiment, I ran the datasets with hided layer 10, and trained the data through iterations for several times. The results you can see in the the accuracy rate is not high as expected. This is probably the result of lack of training and inappropriate training set-up. This algorithm is more suitable for complex situation where large-scale input and output exist. The neural Network result generated by Matlab in the experiment is shown below.










```
>> t_pred=myNN(x2);
[C,order] = confusionmat(round(t_pred),t2)
accuracy=(C(1,1)+C(2,2)+C(3,3)+C(4,4)+C(5,5)+C(6,6)+C(7,7))/sum(sum(C))

C =

     0     1     0     0     0     0     0
     0     1     3     0     0     0     0
     2    38   298   135     3     0     0
     2    13   204   433   141    25     0
     1     0     8    69    92    27     0
     0     0     0     0     0     0     0
     0     0     0     1     0     0     0


order =

     3
     4
     5
     6
     7
     8
     9


accuracy =

    0.5504
```

Figure3 Neural Network Training Results

Boosting

Boosting was performed using the similar decision tree code that was used in the Decision Tree section of this report. Instead of using accuracy only, here we present the results of mean of square error (MSE) to assess the performance of the decision tree with boosting algorithm as well. The comparison among the performances of boosting decision tree and decision tree without boosting is presented below.

```
> test## white wine combined quality boosting decesion tree prediction
   quality.test
      3   4   5   6   7   8   9
  5   5  35 292 128   6   0   0
  6   3  13 180 477 179  19   0
  7   0   0   2  61  75  20   2
> accuracy3=sum(test3[1,3]+test3[2,4],test3[3,5])/sum(test3)
> accuracy3
[1] 0.5434298441
> mean((yhat.boost-quality.test)^2)
[1] 0.5246746418
```

```
> test## red wine combined quality boosting decesion tree prediction
   quality.test
      3   4   5   6   7   8   9
  5   5  35 292 128   6   0   0
  6   3  13 180 477 179  19   0
  7   0   0   2  61  75  20   2
> accuracy3=sum(test3[1,3]+test3[2,4],test3[3,5])/sum(test3)
> accuracy3
[1] 0.5876460768
> mean((yhat.boost-quality.test)^2)
[1] 0.4218918258
```

```
> test##red and white wine combined quality boosting decesion tree prediction
   quality.test
      3   4   5   6   7   8   9
  5   5  35 292 128   6   0   0
  6   3  13 180 477 179  19   0
  7   0   0   2  61  75  20   2
> accuracy3=sum(test3[1,3]+test3[2,4],test3[3,5])/sum(test3)
> accuracy3
[1] 0.5637942552
> mean((yhat.boost-quality.test)^2)
[1] 0.4820682668
```

Figure 4 MSE and Accuracy of the Boosting Tree Method

It is interesting that the white wine quality dataset, red wine quality dataset and combined wine dataset actually showed slight increase in performance under boosting. This improvement is the result of boosting algorithm, which take advantage of modeling on

multiple copies of the fitted original training data set (tree).

Boosting Parameter Importance

	var	rel.inf
alcohol	alcohol	56.429719838
volatile.acidity	volatile.acidity	23.156336883
free.sulfur.dioxide	free.sulfur.dioxide	4.399475889
sulphates	sulphates	3.209837107
residual.sugar	residual.sugar	2.346053965
chlorides	chlorides	2.291950211
total.sulfur.dioxide	total.sulfur.dioxide	1.994785283
density	density	1.795003788
citric.acid	citric.acid	1.669991095
pH	pH	1.172275968
fixed.acidity	fixed.acidity	1.171459346
type	type	0.363110628

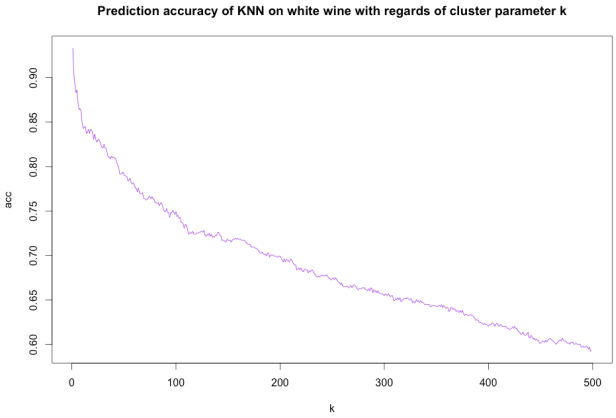
Figure 5 Importance of Input Variables in Boosting

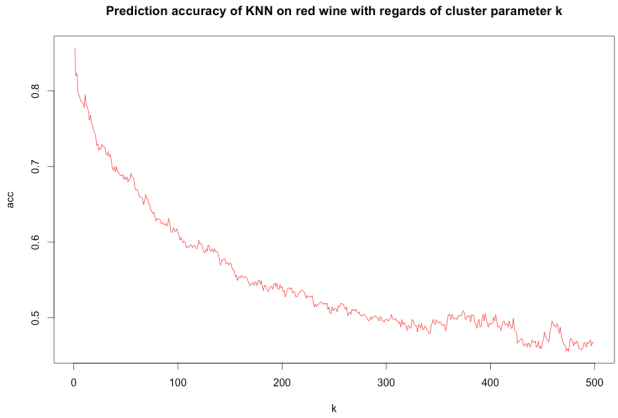
KNN

kNN algorithm works well on the combined wine dataset, and does a good job comparing to other supervised machine learning methods. It shows about an over 50% performance improvement over the other techniques. Why could this happen? The reason that I would guess that this works is largely due to the data normalization of this method. Since

different attributes, like PH value, acid value, have totally different scales. With this normalization, it is likely to reflect the actual situation of the wine quality evaluation. Further, the distribution of the outlying quality function should be continuous, thus it is intuitively reasonable to predict the quality attributes of test instance based on closest training instance.

Besides, the extreme instances in training dataset are considered as outliers, which are ignored in the clustering model.






```
> test#red and white wine combined quality KNN prediction
           quality.test
knn_predict   3   4   5   6   7   8
          3   0   0   0   0   0   0
          4   0   0   0   0   0   0
          5   2  22 306 158   8   0
          6   2  20 170 458 171  30
          7   0   0   3  50  76  21
          8   0   0   0   0   0   0
          9   0   0   0   0   0   0
> accuracy3=sum(test[1,1]+test[2,2],test[3,3]+test[4,4],test[5,5],test[6,6])/sum(test)
> accuracy3
[1] 0.5611222445
```

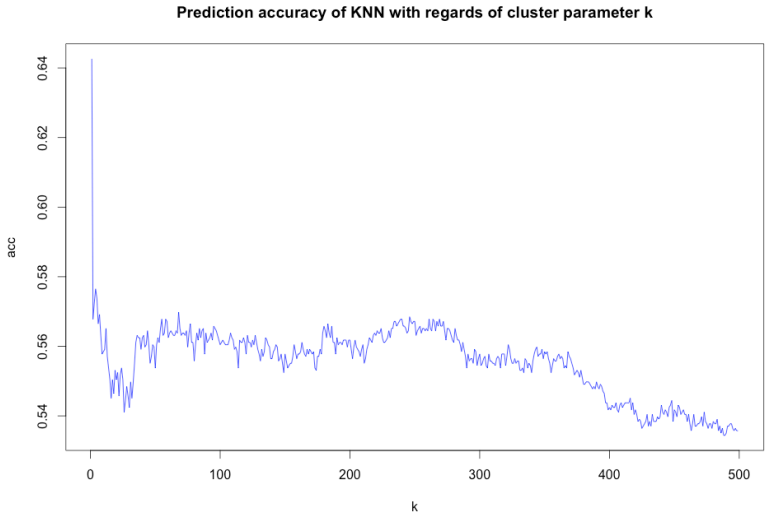


Figure 6 Prediction Accuracy of KNN with regards of Cluster Parameter k

Support Vector Machines(SVM)

For these experiments I used three kind of kernel: linear kernel, polynomial kernel and sigmoid kernel. The training process was relatively slow compared to other supervised machine learning algorithms.

The linear function support vector functions yields better results than the others. This makes sense as the underlying distribution of the sample come from linear models. Nevertheless, SVM should not be considered as the best or suitable method to model and predict wine quality because it has low converging speed, and its prediction accuracy is as low as the decision tree algorithms.

```
> test##red and white wine combined quality SVM prediction
       truth
predict   3   4   5   6   7   8   9
      4   0   2   1   1   0   0   0
      5   2  32 271 133  12   2   0
      6   2  14 224 467 163  33   1
      7   0   1   1  63  55  15   1
      8   0   0   1   0   0   0   0
> accuracy5=sum(test[1,2]+test[2,3],test[3,4]+test[4,5],test[5,6])/sum(test)
> accuracy5
[1] 0.5310621242
```

The linear kernel

```
> test##red and white wine combined quality SVM prediction with polynomial kernel
       truth
predict   3   4   5   6   7   8   9
      3   1   0   0   0   0   0   0
      4   0   1   1   1   0   0   0
      5   2  19 153  65   3   0   0
      6   1  27 342 566 189  37   2
      7   0   1   2  31  37  10   0
      8   0   1   0   1   1   3   0
> accuracy5=sum(test[1,1]+test[2,2],test[3,3],test[4,4],test[5,5],test[6,6])/sum(test)
> accuracy5
[1] 0.5083500334
```

The polynomial kernel

```
> test##red and white wine combined quality SVM prediction with sigmoid kernel
       truth
predict   3   4   5   6   7   8   9
      4   0   1   3   0   0   0   0
      5   2  25 219 121  18   3   0
      6   2  22 275 522 195  42   2
      7   0   1   1  21  17   5   0
> accuracy5=sum(test[1,2]+test[2,3],test[3,4]+test[4,5])/sum(test)
> accuracy5
[1] 0.5070140281
```

The sigmoid kernel

Figure 7 Prediction Accuracy of SVM among Different Kernels