Assignment 3: CMTH642

Christopher Graham

November 27, 2015

Overview NEED TO REWRITE

This assignment develops a model to predict a subjective quality rating on Portuguese red vinho verdes. The data is downloaded from

It provides 11 chemical-ish measurement of the wine, and one human-taster derived quality score.

Our goal is to develop a model that will predict the quality rating of a wine based on these chemical measurements. Numbered sections of this paper correspond to the numbered instructions in the assignment sheet.

1. Import Data

```
# load all libraries required in analysis
require(caret)
require(corrplot)

red <- read.csv('winequality-red.csv', sep = ';')
# Treating as classification problem so convert quality to factor variable
red$quality <- as.factor(red$quality)

# Divide off a testing set for final validation
# All work will be done only based on the training set
set.seed(25678)
train_idx <- createDataPartition(y=red$quality, p=0.8, list = FALSE)
training <- red[train_idx,]
testing <- red[-train_idx,]
```

2. Check Data Characteristics

Completeness

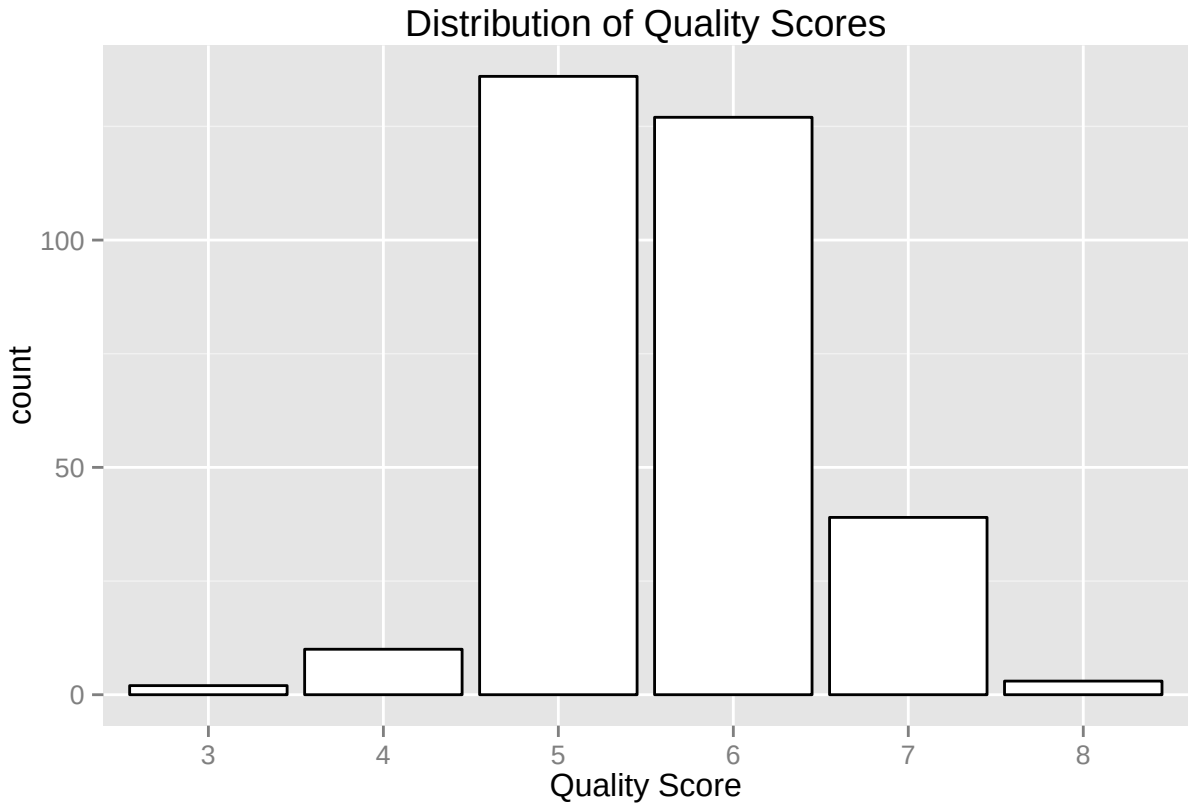
```
if (sum(complete.cases(testing)) == nrow(testing)) {
    print('All complete cases')
}
```

```
## [1] "All complete cases"
```

The data is complete in all variables, and as such there is no need to impute values.

Class Balance

But the data isn't perfect. The biggest problem is with the distribution of quality scores. Specifically, over 80% of the wines have average ratings (5 or 6), and very few wines get ratings at the more extreme ends of the rating spectrum. (3 and 8 are the extreme values awarded, even though wines were rated on a 10-point scale). Class imbalance has been shown to be a significant problem in building an effective Machine Learning model. We provide a strategy for dealing with this below, but first will look at some other characteristics of the data.

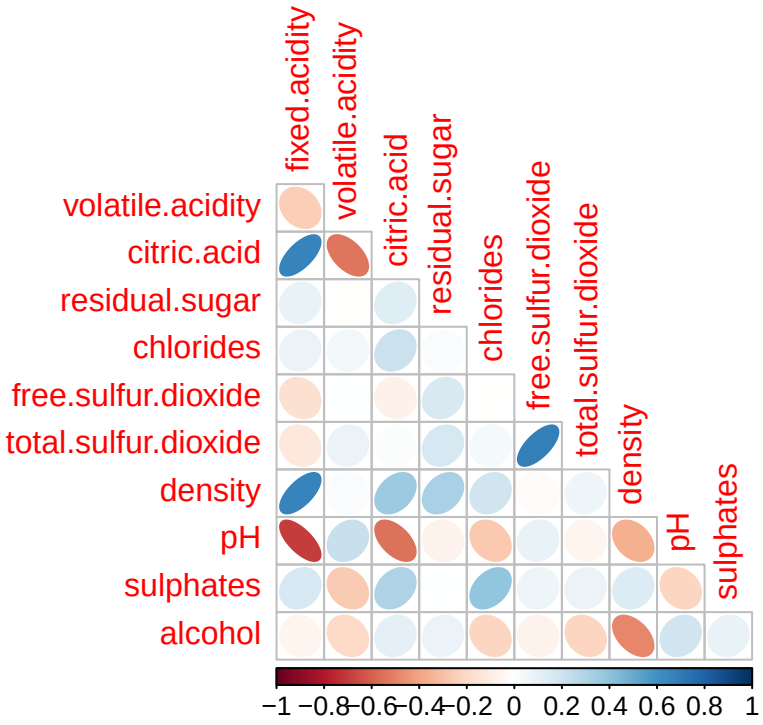


Variable Relationships

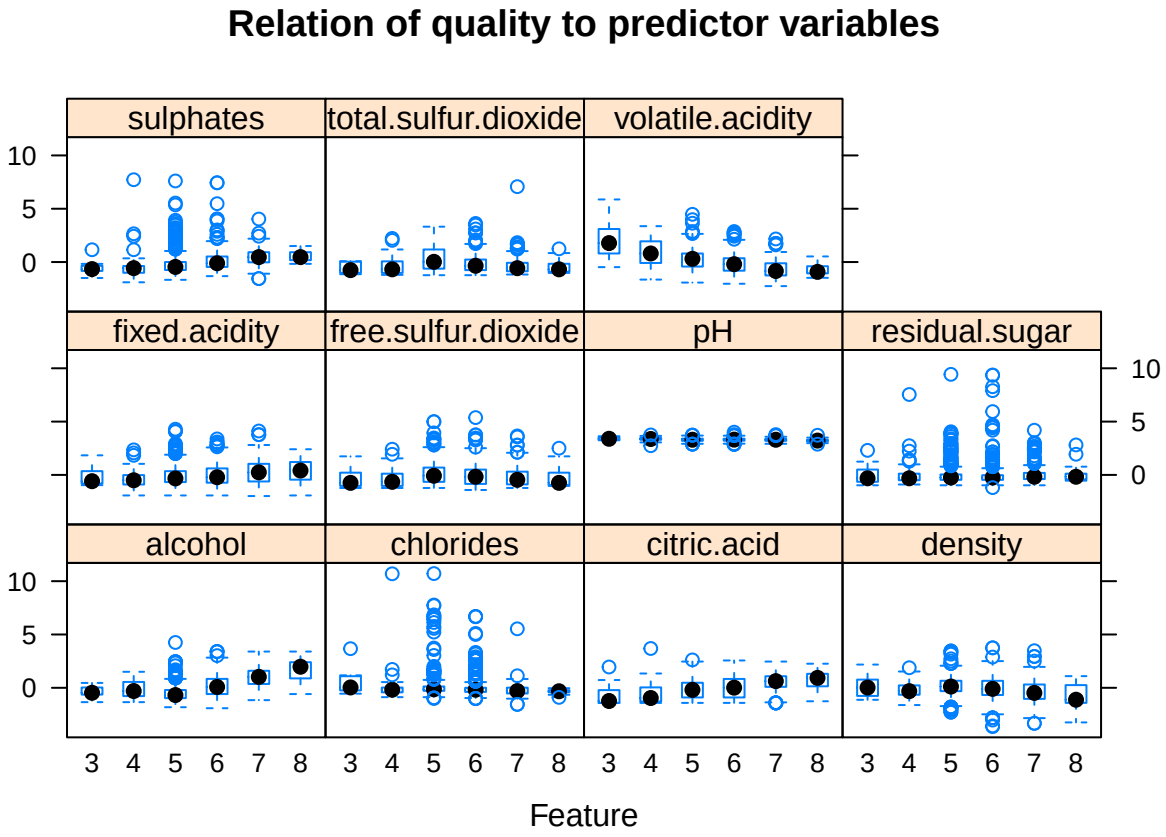
```
# Check for variables with near zero variability
nearZeroVar(training, saveMetrics = TRUE)
```

```
##                      freqRatio percentUnique zeroVar   nzv
## fixed.acidity         1.086957     7.2542902   FALSE FALSE
## volatile.acidity      1.055556    10.6864275   FALSE FALSE
## citric.acid           1.824561     6.2402496   FALSE FALSE
## residual.sugar        1.070796     6.6302652   FALSE FALSE
## chlorides             1.255814    10.9984399   FALSE FALSE
## free.sulfur.dioxide   1.337209     4.6021841   FALSE FALSE
## total.sulfur.dioxide  1.166667    10.9984399   FALSE FALSE
## density               1.033333    30.9672387   FALSE FALSE
## pH                    1.119048     6.7082683   FALSE FALSE
## sulphates             1.000000     7.3322933   FALSE FALSE
## alcohol               1.425000     4.6801872   FALSE FALSE
## quality               1.066536     0.4680187   FALSE FALSE
```

```
# look at correlation between predictor variables
corrplot(cor(training[-12]), method = 'ellipse', type = 'lower',
         title = 'Correlation of predictor variables', diag=FALSE,
         mar=c(0,0,1,0))
```




```
# Compare relationsip of quality to predictor variables
# first normalize the variables so we can compare in one graph
preObj <- preProcess(training[,-9], method=c('center', 'scale'))
train_norm <- predict(preObj, training)
featurePlot(x=train_norm[,-12], y=train_norm$quality, plot='box',
            main = 'Relation of quality to predictor variables')
```



What this preliminary exploration tells us is:

- None of the variables show near zero variability, meaning there is no *prima facie* reason to exclude any at the start of our analysis
- For the most part, there does not seem to be a huge amount of collinearity between variables. There are some instances of collinearity, where it might be expected (citric.acid, fixed.acidity and pH), (free.sulfur.dioxide and total.sulfur.dioxide). So there may be some room to reduce dimensionality.
- There are some clear relationships between some of the predictor variables and wine quality. volatile.acidity, alcohol and density seem particularly important.

Strategies for dealing with Class Imbalance in Provided Data

Since one of the goals for a prediction system would be to help vintners identify wines that are likely to be either very good or very bad, the lack of wine specimens with extreme scores could be problematic for our model development.

Dealing with class imbalance is a common problem in machine learning. The standard approach in this situation is a combination of under-sampling the majority class and over-sampling the minority class to create a distribution that will help to develop an adequate predictive model. However, there is a persistent concern about the extent to which this under & oversampling should occur, and the specific techniques to implement.

While there are many different ways of approaching this problem, many of them are designed around a binary response variable. Specifically, the R package 'unbalanced' provides a number of pre-made approaches to dealing with an unbalanced data set. But, unfortunately, it requires a binary response variable.

In this case, we are approaching this as a multiple-level classification, relying on the classes provided in the original data. In part, this is to ensure comparability with the original paper produced on this data set.

I have identified two different approaches to effectively determining an over-/ under-sampling approach for multi-class response variables. The first is the wrapper approach proposed by Chawla et al[1], and the resampling ensemble algorithm proposed by Qian et al[2].

Both of these approaches rely on a combination of SMOTE oversampling and random undersampling. However, the Chawla paper offers a more coherent (and KDD-Cup proven!) approach to determining sample levels, and so we will use that approach here.

Note, however that the algorithms used in Chawla et al require us to identify a classifier algorithm that will be used to determine stop points for sampling.

So, we need to turn briefly to model selection.

3. Model Selection

Regression vs. Classification

As noted on the source page for this data set, the data lends itself to either a regression or classification approach. The original paper to use this data set adopted a regression appraoch to the data, arguing that regression allows us to better evaluate "near miss" predictions (e.g. if the true value is 3, we can up-score the model if it predicts 4, rather than 7). The paper used both Neural Network (NN) and Support Vector Machine (SVM) models, and obtained an accuracy rate of 64% with a 0.5 error tolerance in quality prediction, and 89% accuracy with a 1.0 error tolerance.[3]

In order to make things interesting, we're going to address this as a classification problem. This means the fairest comparison for our results is with the 0.5 tolerance leve in the original paper, but also that we can hope to beat the best results from their SVM model.

Model Selection

Given the relatively weak predictive accuracy (at T=0.5) for the models in the first paper, it appears that a single model may not be all that great at classification. In these instances, one of the more interesting approaches is *ensemble learning* - essentially combining predictions from multiple models into one super-model.[4]

In terms of specific ensemble learning implementation, we are going to use the approach suggested by Jeff Leek of John's Hopkins in the Coursera course "Practial Machine Learning"[5].

In order to build an ensemble model, we first need to build a number of individual models to combine into the final approach. In an attempt to bring together all we've done in the course (plus a little bit more!), we're going to use:

- multinomial logistic regression
- Naïve Bayes

[1]Chawla NV, Cieslak DA, Hall LO, Joshi A. Automatically countering imbalance and its empirical relationship to cost. *Data Mining and Knowledge Discovery.* 2008;17(2):225-52.

[2]Y. Qian, et al., A resampling ensemble algorithm for classification of imbalance problems, *Neurocomputing* (2014), http://dx.doi.org/10.1016/j.neucom.2014.06.021i

[3]Cortez P, Cerdeira A, Almeida F, Matos T, Reis J. Modeling wine preferences by data mining from physicochemical properties. Decision Support Systems. 2009;47(4):547-53.

[4]Numerous articles have shown this, but see specifically: Rokach L. Ensemble-based classifiers. Artificial Intelligence Review. 2010;33(1):1-39.

[5]Course notes available at: http://sux13.github.io/DataScienceSpCourseNotes/8_PREDMACHLEARN/Practical_Machine_Learning_Course_Notes.pdf

```
print('this is python 2.7.10')
import sys
print(sys.version)
```

```
## this is python 2.7.10
## 2.7.10 (default, Aug 22 2015, 20:33:39)
## [GCC 4.2.1 Compatible Apple LLVM 7.0.0 (clang-700.0.59.1)]
```

3. (30 points) Propose a model for the prediction. Give a few reasons for your selection briefly. You may choose to model the problem as classification or regression. Define the task, experience and performance criteria.
4. (30 points) Do 10 fold cross validation in the experiment.
5. (20 points) Report your results.
6. (Bonus 20 points) Other researchers have built predictive models using this data. Do a brief comparison of your results with the result in the following paper: http://repositorium.sdum.uminho.pt/bitstream/1822/10029/1/wine5.pdf

Citation example

$1.3 million[6]

[6]source: http://www.millersamuel.com/files/2013/04/Manhattan_1Q_2013.pdf