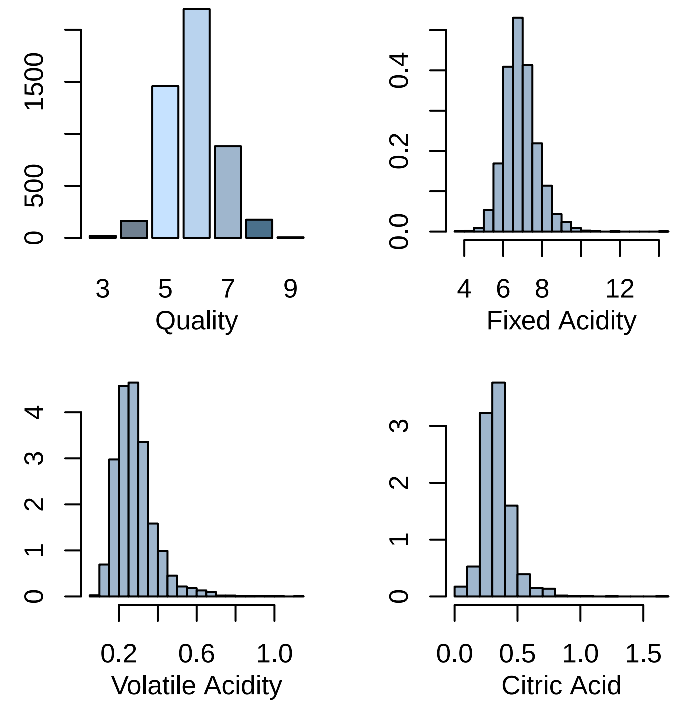
Wesley Pryor

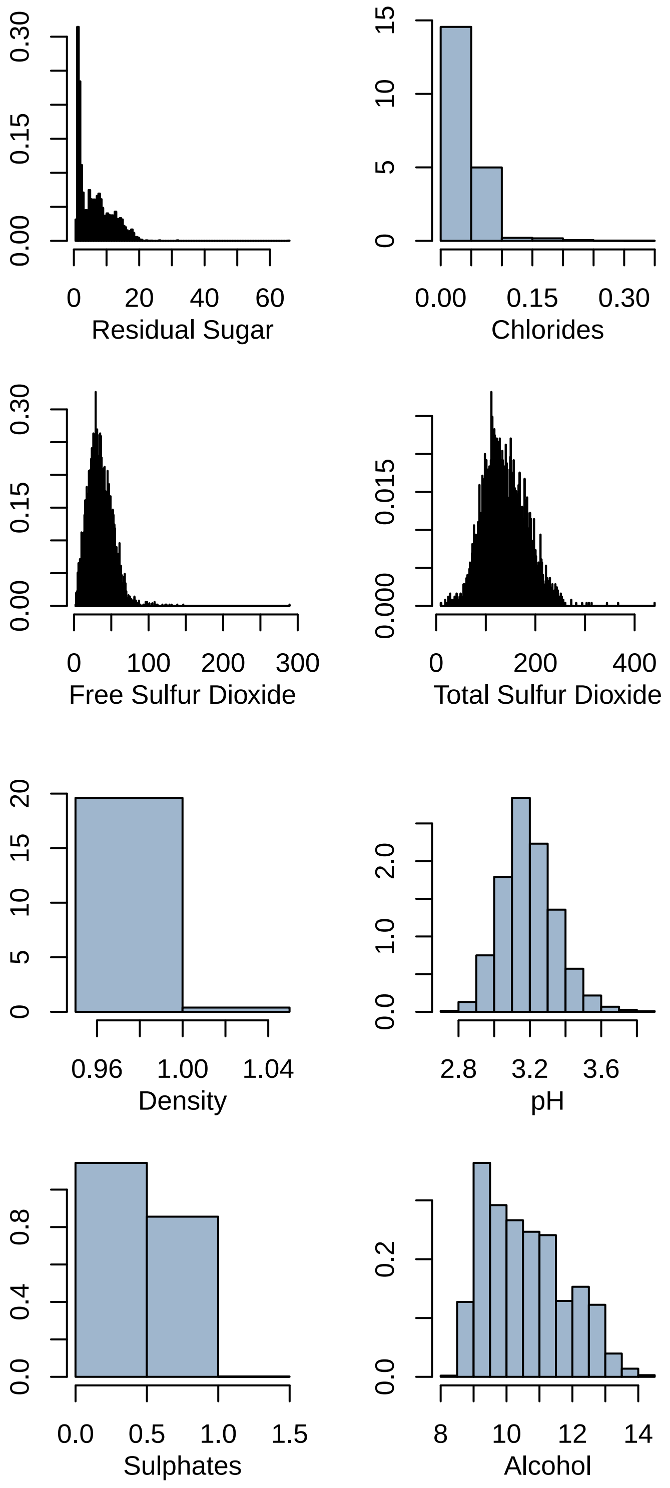
Final Project

M5305–Statistical Models Fall 2016 Tarleton State Univ Dr. Crawford Due 2016-12-12

Data Introduction

For this rpoject, I have chosen the white wine set of data from the Wine Quality dataset. This set of data comes from the UCI Machine Learning Repository. The wines come from Portugal and are variants of the *Vinho Verde* wine. The wines in the data were tested for certain factors of each wine. The set has twelve attributes. The attributes are fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, pH, sulphates, and alcohol. The final attribute is quality. Quality was assesed by wien experts and calculated based on a median value of scores from 0 (very poor) to 10 (extremely excellent). In the data most of the values of quality are between 5 and 7. There are a few 3 values and fewer 9 values. The quality attribute is the value that will be predicted in the following models. The practical application of these proposed models is to predict how well a wine will taste and be received by consumers based upon the chemical makeup of the wine. If a winery can rpedict the quality of a wine, they will know if it is a higher quality, the company should prodcue more wine.I first tested the data to see how it behaves naturally and I ran histograms on all 12 attributes. The histograms were printed as follows:





Linear Models

With further testing using boxplots. There were some extreme outliers in citric acid, fixed acidity, and volatile acidity categories. I first ran a linear model with all attributes in the data to predict quality. This was just an inital test to see how the data behaved and if any changes were needed to enhance the model to predict quality. The plots of the residuals and predicted Y values showed correlation with the predictions. The first Root Mean Square Error produced a value of 0.77. This was a good RMSE, however, the model needed to be changed based on thegraphs that I saw. The graphs showed obvious correlation. I used the box cox transformation to change the values by making the exponents of the quality vector to be 0.68. The adjusted values reduced the RMSE to 0.29. I aslo tested a step model to try to reduce the variables. This model included alcohol, volatile acidity, residual sugar, free sulfur dioxide, sulphates, chlorides, and pH. However, after running and F-test, the p-value was extremely small. Therefore, we could say that the step model was not better than the model with all the attributes.

Decision Trees

I also wanted to incorporate more models. THerefore I decided to run decision trees with the data. The decision tree that was calculated hap a depth of 4 and was initially calculated to have an RMSE of 0.02344. The model was then cross-validated using ten fold cross validation. This method of cross validation resulted in an average RMSE of 0.02011. The low RMSE values show the method of R Part decision trees is a very good method. This shows that decision trees predict the quality of white wines with all attributes very well.

K nearest enighbors

I also decided to incorporate more data mining techniques and run K nearest neighbors. The method was done with 3 nearest neighbors. I thought that this would be a good method. I had to change the value of the residuals from factor to integers to calculate the RMSE. After calculating the RMSE, the value was 2.0374. This was extremely high compared to the other models where I was getting an RMSE close to zero. Once I saw this, I decided to stop trying to fix this model because I had other models that performed exponentially better than this method.

Neural Networks

I also wanted to try one more technique that I have learned from data mining to find the best model. However, a naive run of this neural network was even higher than a test of K nearest neighbors. Therefore, I decided to not continue with this model.

Conclusion

I believe that the ordinary least quares linear model and the decision trees were the best models that can predict the quality of wine. I believe that the decision trees was the most practical model. It is a model that wine developers can follow to determine how well the wine will perform. Iff wine developers want to create successful wines repeatedly, then they cna use the decision trees to see where a wine will be predicted for quality with confidence in the accuracy of the model. With cross-validation of the model we saw that decision were definiely the best model and became a reliable model. Other models found were reliable as well.

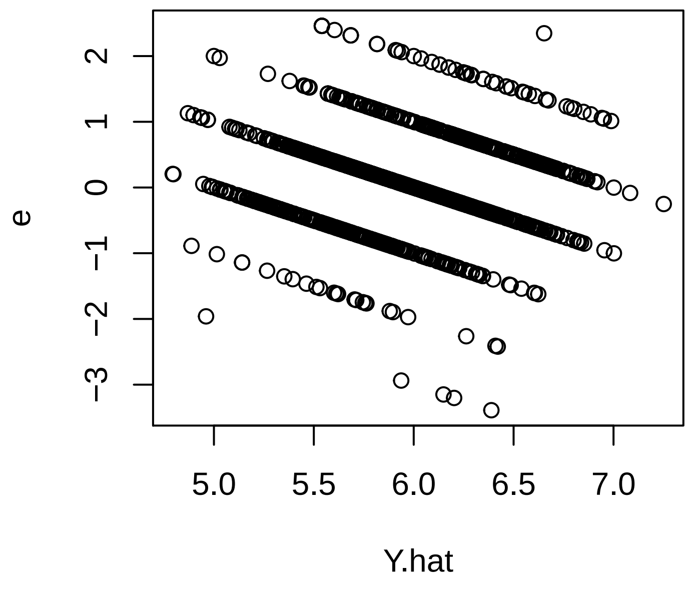
Code Appendix

```
# Preparing the Data for outliers
limout ← rep(0, 11)
for (i in 1:11) {
    t1 ← quantile(WhiteData[, i], 0.75)
    t2 ← IQR(WhiteData[, i], 0.75)
    limout[i] ← t1 + 1.5 * t2
}
WhiteWineIndex ← matrix(0, 4898, 11)
for (i in 1:4898) for (j in 1:11) {
    if (WhiteData[i, j] > limout[j])
        WhiteWineIndex[i, j] ← 1
}
WWInd ← apply(WhiteWineIndex, 1, sum)
WhiteWineTemp ← cbind(WWInd, WhiteData)
Indexes ← rep(0, 208)
j ← 1
for (i in 1:4898) {
    if (WWInd[i] > 0) {
        Indexes[j] ← i
        j ← j + 1
    } else j ← j
}
WhiteWineLib ← WhiteData[-Indexes, ]

X.1 ← WhiteWineLib[, 1]
X.2 ← WhiteWineLib[, 2]
X.3 ← WhiteWineLib[, 3]
X.4 ← WhiteWineLib[, 4]
X.5 ← WhiteWineLib[, 5]
X.6 ← WhiteWineLib[, 6]
X.7 ← WhiteWineLib[, 7]
X.8 ← WhiteWineLib[, 8]
X.9 ← WhiteWineLib[, 9]
X.10 ← WhiteWineLib[, 10]
X.11 ← WhiteWineLib[, 11]

X ← cbind(X.1, X.2, X.3, X.4, X.5, X.6, X.7, X.8, X.9, X.10, X.11)
Y ← WhiteWineLib[, 12]

# Split of Data into training and testing data
set.seed(1506)
index ← sample(1:nrow(WhiteWineLib), size = round(0.7 * nrow(WhiteWineLib),
    0))
wine.train ← WhiteWineLib[index, ]
wine.test ← WhiteWineLib[-index, ]

# Linear Model Initial Application
linmodel ← lm(quality ~ ., data = wine.train)
summary(linmodel)
```

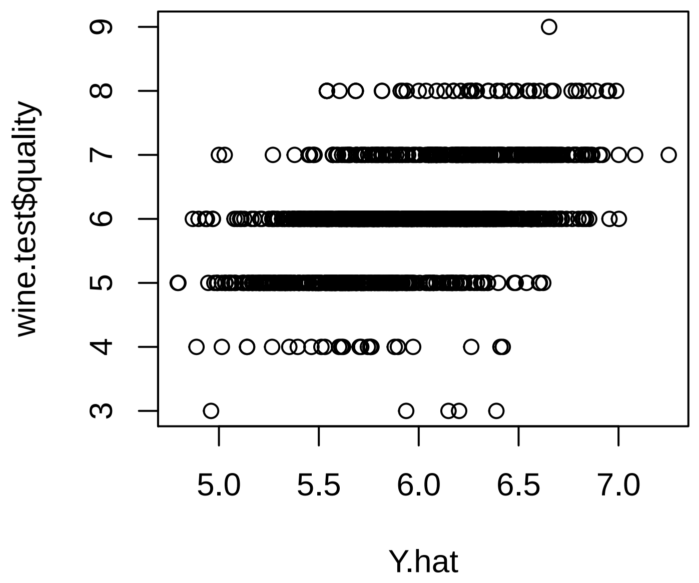
```

Call:
lm(formula = quality ~ ., data = wine.train)

Residuals:
    Min      1Q  Median      3Q     Max
-3.3867 -0.5068 -0.0458  0.4596  2.8007

Coefficients:
                       Estimate Std. Error t value Pr(>|t|)
(Intercept)           2.256e+02  3.309e+01   6.818 1.13e-11 ***
fixed.acidity         1.590e-01  3.256e-02   4.884 1.10e-06 ***
volatile.acidity     -1.914e+00  1.887e-01 -10.141  < 2e-16 ***
citric.acid           1.413e-02  1.603e-01   0.088  0.92979
residual.sugar        1.054e-01  1.234e-02   8.544  < 2e-16 ***
chlorides            -2.848e+00  1.719e+00  -1.657  0.09766 .
free.sulfur.dioxide   4.926e-03  1.235e-03   3.989 6.81e-05 ***
total.sulfur.dioxide  3.516e-04  5.165e-04   0.681  0.49610
density              -2.269e+02  3.353e+01  -6.766 1.60e-11 ***
pH                    9.359e-01  1.545e-01   6.058 1.56e-09 ***
```

```
sulphates              6.903e-01  1.493e-01   4.625 3.91e-06 ***
alcohol                1.062e-01  4.113e-02   2.582  0.00986 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.733 on 2840 degrees of freedom
Multiple R^2:  0.2694,   Adjusted R^2:  0.2665
F-statistic: 95.19 on 11 and 2840 DF,  p-value: < 2.2e-16
```

```
Y.hat <- predict(linmodel, newdata = wine.test)
e <- wine.test$quality - Y.hat
plot(Y.hat, e)
```



```
plot(Y.hat, wine.test$quality)
```



```
RMSE ← sqrt(mean(e^2))

# Step application to determine the best predictors for quality
stepmod ← step(lm(quality ∼ 1, wine.train), scope = list(lower = ∼1, upper = ∼fixed.acidity +
    volatile.acidity + citric.acid + residual.sugar + chlorides + free.sulfur.dioxide +
    total.sulfur.dioxide + pH + sulphates + alcohol), direction = "forward",
    trace = FALSE)
step.model ← lm(quality ∼ alcohol + volatile.acidity + residual.sugar + free.sulfur.dioxide +
    sulphates + chlorides + pH, data = wine.train)

# F.test for normal and step model
f.test(step.model, linmodel)
```

```
$f.stat
[1] 12.00224

$p.value
[1] 1.130184e-09
```

```
# Box-Cox Transformation
lambdarang ← seq(−3, 3, 0.1)
box.cox(Y, X, lambdarang)
```

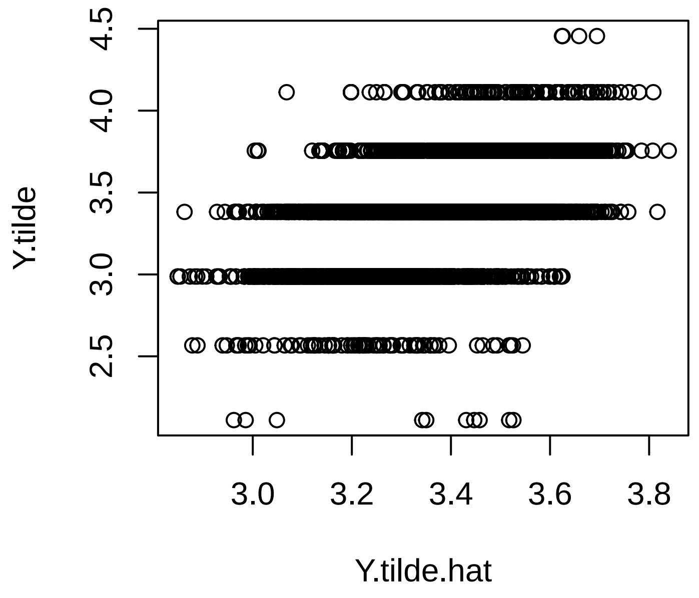
```
[1] 0.7
```

```
lambdarange ← seq(0.5, 1.5, 0.01)
box.cox(Y, X, lambdarange)
```

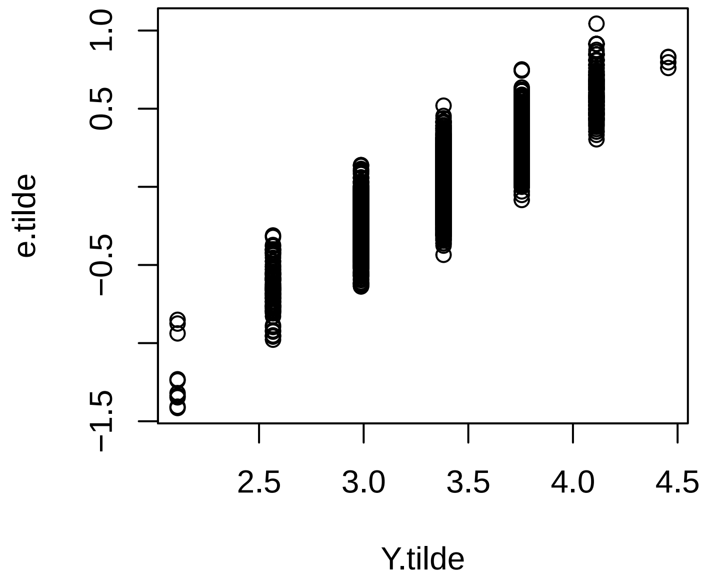
```
[1] 0.68
```

```
Y.tilde ← Y^0.68
adj.model ← lm(Y.tilde ∼ X.1 + X.2 + X.3 + X.4 + X.5 + X.6 + X.7 + X.8 + X.9 +
    X.10 + X.11)
Y.tilde.hat ← predict(adj.model)
```

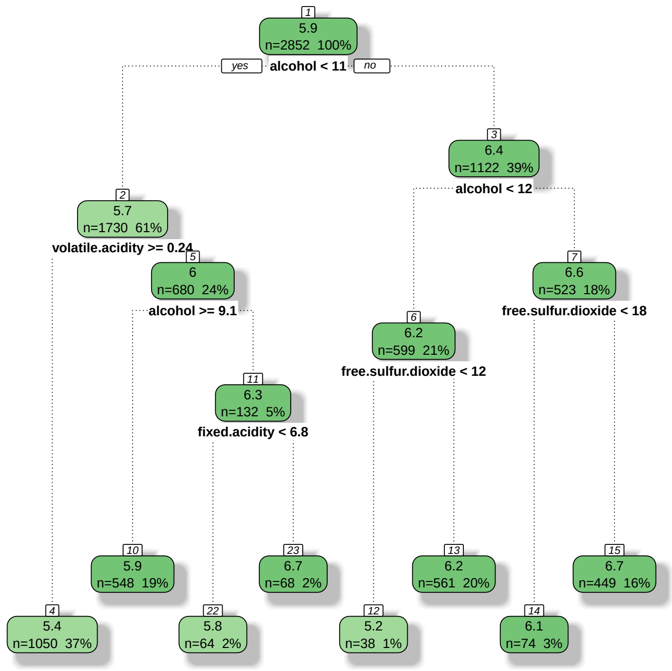
```
e.tilde ← Y.tilde − Y.tilde.hat
plot(Y.tilde.hat , Y.tilde)
```



```
plot(Y.tilde , e.tilde)
```



```
adj.RMSE ← sqrt(mean(e.tilde^2))

# Decision Trees
wine.tree ← rpart(quality ∼ ., data = wine.train)
fancyRpartPlot(wine.tree)
```



Rattle 2016−Dec−11 18:34:39 wdpryor1994

```
wine.pred ← predict(wine.tree, newdata = wine.test)
tree.RMSE ← sqrt(mean(wine.test$quality − wine.pred)^2)

# Cross Validation
createfolds = function(n, K) {
    reps = ceiling(n/K)
    non_rand_folds = rep(1:K, reps)
    non_rand_folds = non_rand_folds[1:n]
    folds = sample(non_rand_folds)
    return(folds[1:n])
}
kflval ← function(k, data) {
    folds = createfolds(nrow(data), k)
    accvector = 1:k
    for (k in 1:k) {
        temptrain = data[folds != k, ]
        temptest = data[folds == k, ]
        temptree = rpart(quality ~ ., data = temptrain)
        temppred = predict(temptree, newdata = temptest)
        analysis = sqrt(mean(temptest$quality − temppred)^2)
        accvector[k] = analysis
    }
    return(mean(accvector))
}

kflval(10, WhiteData)
```

```
[1] 0.03680077
```

```
# K nearest neighbors
x = WhiteData[, 1:11]
xbar = apply(x, 2, mean)
xbarMat = cbind(rep(1, nrow(WhiteData))) %*% xbar
s = apply(x, 2, sd)
```

```
sMat = cbind(rep(1, nrow(WhiteData))) %*% s
z = (x - xbarMat)/sMat
Y = WhiteData$quality
z = cbind(z, Y)

z.split <- sample(nrow(z), round(nrow(z) * 0.7, 0))
z.train <- z[z.split, ]
z.test <- z[-z.split, ]

wine.knn <- knn(train = z.train, test = z.test, k = 3, cl = Y[z.split])
wine.knn <- as.integer(wine.knn)
residual.knn <- Y[-z.split] - wine.knn
knn.RMSE <- sqrt(mean(residual.knn^2))

# Neural Networks
wine.nnet <- nnet(quality ~ ., data = wine.train, size = 10, linout = FALSE,
    maxit = 500)
```

```
# weights:  131
initial  value 86258.628824
final  value 71491.000000
converged
```

```
pred.nnet <- predict(wine.nnet, newdata = wine.test)
nnet.residuals <- wine.test$quality - pred.nnet
nnet.RMSE <- sqrt(mean(nnet.residuals^2))
```