Wine: Chemical Analysis

Full-Out Linear Regression

* Model
* Diagnostics
* Transformation
* Random Effect Model

**I**

I fit a linear model with quality as a response and the rest of variables as predictors, but I noticed that response is a discrete variable which can be thought of as being treated a continuous variable in this regression, and the problem naturally arises later in residual vs fit plot.

> summary(lmod)

Call:

lm(formula = quality ~ facidity + vacidity + citric + rsugar +

chlorides + fso2 + tso2 + density + pH + so4 + alcohol)

Residuals:

Min 1Q Median 3Q Max

-2.68911 -0.36652 -0.04699 0.45202 2.02498

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) 2.197e+01 2.119e+01 1.036 0.3002

facidity 2.499e-02 2.595e-02 0.963 0.3357

vacidity -1.084e+00 1.211e-01 -8.948 < 2e-16 \*\*\*

citric -1.826e-01 1.472e-01 -1.240 0.2150

rsugar 1.633e-02 1.500e-02 1.089 0.2765

chlorides -1.874e+00 4.193e-01 -4.470 8.37e-06 \*\*\*

fso2 4.361e-03 2.171e-03 2.009 0.0447 \*

tso2 -3.265e-03 7.287e-04 -4.480 8.00e-06 \*\*\*

density -1.788e+01 2.163e+01 -0.827 0.4086

pH -4.137e-01 1.916e-01 -2.159 0.0310 \*

so4 9.163e-01 1.143e-01 8.014 2.13e-15 \*\*\*

alcohol 2.762e-01 2.648e-02 10.429 < 2e-16 \*\*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.648 on 1587 degrees of freedom

Multiple R-squared: 0.3606, Adjusted R-squared: 0.3561

F-statistic: 81.35 on 11 and 1587 DF, p-value: < 2.2e-16

I perform model selection by backward selection based on elimination of variable with the largest p-value.

The largest p-value is with the variate density

> lmod1 = update(lmod, .~., -density)

> summary(lmod1)

Call:

lm(formula = quality ~ facidity + vacidity + citric + rsugar +

chlorides + fso2 + tso2 + pH + so4 + alcohol)

Residuals:

Min 1Q Median 3Q Max

-2.67204 -0.36527 -0.04523 0.45628 2.03894

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) 4.4538341 0.6125783 7.271 5.59e-13 \*\*\*

facidity 0.0081441 0.0160586 0.507 0.61212

vacidity -1.0964449 0.1200866 -9.130 < 2e-16 \*\*\*

citric -0.1836098 0.1471561 -1.248 0.21232

rsugar 0.0089507 0.0120542 0.743 0.45787

chlorides -1.9067341 0.4173928 -4.568 5.30e-06 \*\*\*

fso2 0.0045147 0.0021631 2.087 0.03704 \*

tso2 -0.0033120 0.0007264 -4.560 5.52e-06 \*\*\*

pH -0.5042762 0.1571117 -3.210 0.00136 \*\*

so4 0.8928974 0.1107548 8.062 1.46e-15 \*\*\*

alcohol 0.2927427 0.0173394 16.883 < 2e-16 \*\*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.6479 on 1588 degrees of freedom

Multiple R-squared: 0.3603, Adjusted R-squared: 0.3562

F-statistic: 89.43 on 10 and 1588 DF, p-value: < 2.2e-16

The largest p-value is now with the variate rsugar

> lmod2 <- lm(quality ~ facidity + vacidity + citric + chlorides + fso2 + tso2 + pH + so4 + alcohol)

> summary(lmod2)

Call:

lm(formula = quality ~ facidity + vacidity + citric + chlorides +

fso2 + tso2 + pH + so4 + alcohol)

Residuals:

Min 1Q Median 3Q Max

-2.68601 -0.36723 -0.04516 0.45629 2.02723

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) 4.4410923 0.6122514 7.254 6.31e-13 \*\*\*

facidity 0.0090201 0.0160129 0.563 0.57331

vacidity -1.0905804 0.1198096 -9.103 < 2e-16 \*\*\*

citric -0.1756500 0.1467444 -1.197 0.23149

chlorides -1.8893071 0.4166737 -4.534 6.21e-06 \*\*\*

fso2 0.0046664 0.0021532 2.167 0.03036 \*

tso2 -0.0032560 0.0007224 -4.507 7.04e-06 \*\*\*

pH -0.5022333 0.1570654 -3.198 0.00141 \*\*

so4 0.8872849 0.1104810 8.031 1.86e-15 \*\*\*

alcohol 0.2940206 0.0172514 17.043 < 2e-16 \*\*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.6479 on 1589 degrees of freedom

Multiple R-squared: 0.3601, Adjusted R-squared: 0.3564

F-statistic: 99.34 on 9 and 1589 DF, p-value: < 2.2e-16

> lmod3 <- lm(quality ~ vacidity + citric + chlorides + fso2 + tso2 + pH + so4 + alcohol)

> summary(lmod3)

Call:

lm(formula = quality ~ vacidity + citric + chlorides + fso2 +

tso2 + pH + so4 + alcohol)

Residuals:

Min 1Q Median 3Q Max

-2.66890 -0.37044 -0.04474 0.45697 2.02363

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) 4.6680876 0.4608410 10.129 < 2e-16 \*\*\*

vacidity -1.0736123 0.1159362 -9.260 < 2e-16 \*\*\*

citric -0.1295444 0.1217717 -1.064 0.2876

chlorides -1.9494185 0.4026906 -4.841 1.42e-06 \*\*\*

fso2 0.0047601 0.0021463 2.218 0.0267 \*

tso2 -0.0033658 0.0006954 -4.840 1.42e-06 \*\*\*

pH -0.5491501 0.1331350 -4.125 3.90e-05 \*\*\*

so4 0.8914283 0.1102122 8.088 1.19e-15 \*\*\*

alcohol 0.2928780 0.0171280 17.099 < 2e-16 \*\*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.6477 on 1590 degrees of freedom

Multiple R-squared: 0.3599, Adjusted R-squared: 0.3567

F-statistic: 111.8 on 8 and 1590 DF, p-value: < 2.2e-16

> lmod4 <- lm(quality ~ vacidity + chlorides + fso2 + tso2 + pH + so4 + alcohol)

> summary(lmod4)

Call:

lm(formula = quality ~ vacidity + chlorides + fso2 + tso2 + pH +

so4 + alcohol)

Residuals:

Min 1Q Median 3Q Max

-2.68918 -0.36757 -0.04653 0.46081 2.02954

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) 4.4300987 0.4029168 10.995 < 2e-16 \*\*\*

vacidity -1.0127527 0.1008429 -10.043 < 2e-16 \*\*\*

chlorides -2.0178138 0.3975417 -5.076 4.31e-07 \*\*\*

fso2 0.0050774 0.0021255 2.389 0.017 \*

tso2 -0.0034822 0.0006868 -5.070 4.43e-07 \*\*\*

pH -0.4826614 0.1175581 -4.106 4.23e-05 \*\*\*

so4 0.8826651 0.1099084 8.031 1.86e-15 \*\*\*

alcohol 0.2893028 0.0167958 17.225 < 2e-16 \*\*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.6477 on 1591 degrees of freedom

Multiple R-squared: 0.3595, Adjusted R-squared: 0.3567

F-statistic: 127.6 on 7 and 1591 DF, p-value: < 2.2e-16

We have eliminated the following variables in order: density, residual sugar, fixed acidity, and citric acid.

No other variables have high enough p-value to be eliminated anymore.

Using backward elimination based on AIC value also reaches the same model.

> step(lmod)

Start: AIC=-1375.49

quality ~ facidity + vacidity + citric + rsugar + chlorides +

fso2 + tso2 + density + pH + so4 + alcohol

Df Sum of Sq RSS AIC

- density 1 0.287 666.70 -1376.8

- facidity 1 0.389 666.80 -1376.5

- rsugar 1 0.498 666.91 -1376.3

- citric 1 0.646 667.06 -1375.9

<none> 666.41 -1375.5

- fso2 1 1.694 668.10 -1373.4

- pH 1 1.957 668.37 -1372.8

- chlorides 1 8.391 674.80 -1357.5

- tso2 1 8.427 674.84 -1357.4

- so4 1 26.971 693.38 -1314.0

- vacidity 1 33.620 700.03 -1298.8

- alcohol 1 45.672 712.08 -1271.5

Step: AIC=-1376.8

quality ~ facidity + vacidity + citric + rsugar + chlorides +

fso2 + tso2 + pH + so4 + alcohol

Df Sum of Sq RSS AIC

- facidity 1 0.108 666.81 -1378.5

- rsugar 1 0.231 666.93 -1378.2

- citric 1 0.654 667.35 -1377.2

<none> 666.70 -1376.8

- fso2 1 1.829 668.53 -1374.4

- pH 1 4.325 671.02 -1368.5

- tso2 1 8.728 675.43 -1358.0

- chlorides 1 8.761 675.46 -1357.9

- so4 1 27.287 693.98 -1314.7

- vacidity 1 35.000 701.70 -1297.0

- alcohol 1 119.669 786.37 -1114.8

Step: AIC=-1378.54

quality ~ vacidity + citric + rsugar + chlorides + fso2 + tso2 +

pH + so4 + alcohol

Df Sum of Sq RSS AIC

- rsugar 1 0.257 667.06 -1379.9

- citric 1 0.565 667.37 -1379.2

<none> 666.81 -1378.5

- fso2 1 1.901 668.71 -1376.0

- pH 1 7.065 673.87 -1363.7

- chlorides 1 9.940 676.75 -1356.9

- tso2 1 10.031 676.84 -1356.7

- so4 1 27.673 694.48 -1315.5

- vacidity 1 36.234 703.04 -1295.9

- alcohol 1 120.633 787.44 -1114.7

Step: AIC=-1379.93

quality ~ vacidity + citric + chlorides + fso2 + tso2 + pH +

so4 + alcohol

Df Sum of Sq RSS AIC

- citric 1 0.475 667.54 -1380.8

<none> 667.06 -1379.9

- fso2 1 2.064 669.13 -1377.0

- pH 1 7.138 674.20 -1364.9

- tso2 1 9.828 676.89 -1358.5

- chlorides 1 9.832 676.89 -1358.5

- so4 1 27.446 694.51 -1317.5

- vacidity 1 35.977 703.04 -1297.9

- alcohol 1 122.667 789.73 -1112.0

Step: AIC=-1380.79

quality ~ vacidity + chlorides + fso2 + tso2 + pH + so4 + alcohol

Df Sum of Sq RSS AIC

<none> 667.54 -1380.8

- fso2 1 2.394 669.93 -1377.1

- pH 1 7.073 674.61 -1365.9

- tso2 1 10.787 678.32 -1357.2

- chlorides 1 10.809 678.35 -1357.1

- so4 1 27.060 694.60 -1319.2

- vacidity 1 42.318 709.85 -1284.5

- alcohol 1 124.483 792.02 -1109.4

Call:

lm(formula = quality ~ vacidity + chlorides + fso2 + tso2 + pH +

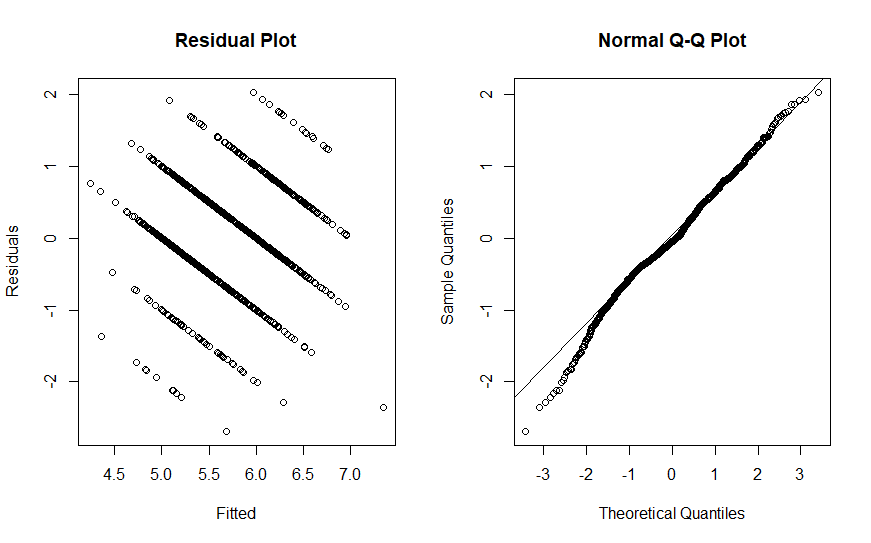
so4 + alcohol)

Coefficients:

(Intercept) vacidity chlorides fso2 tso2 pH so4 alcohol

4.430099 -1.012753 -2.017814 0.005077 -0.003482 -0.482661 0.882665 0.289303

I check the model assumptions. The two diagnostic plots are presented:



It is difficult to interpret this residual plot. The diagonal streaks arise from discrete values of wine quality.

The qqplot suggests that the distribution of residuals is slightly light-tailed on the left tail of distribution.

> shapiro.test(residuals(lmod))

Shapiro-Wilk normality test

data: residuals(lmod)

W = 0.99137, p-value = 4.321e-08

Shapiro test punishes hard for mild deviation from normality and as such gives a verdict that the distribution is clearly not normal.

Recall that for short-tailed distribution the large sample size allows us to make estimation of coefficients, so we have little reason to abandon linear regression. However, it does not allow inference such as confidence intervals.

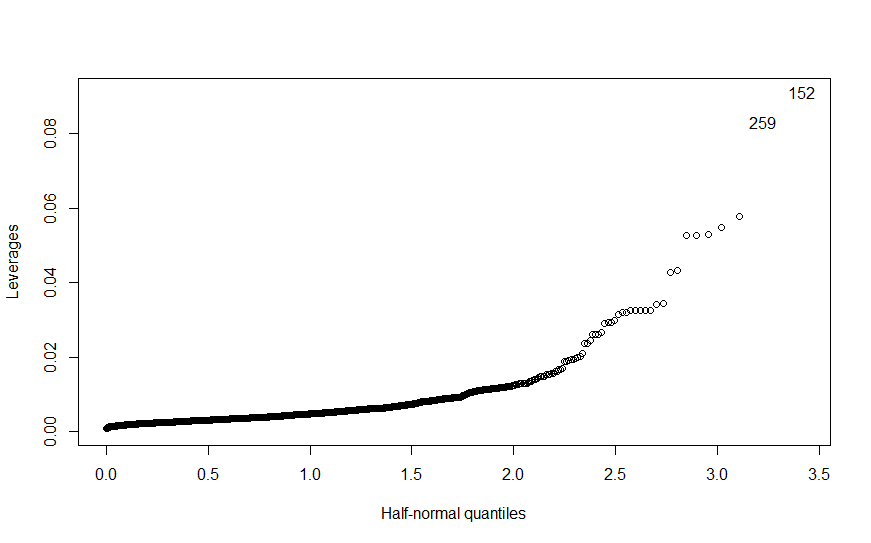
The model also fails Durbin Watson test implying errors are heavily correlated.

> dwtest(quality ~ vacidity + chlorides + fso2 + tso2 + pH + so4 + alcohol)  
  
 Durbin-Watson test  
  
data: quality ~ vacidity + chlorides + fso2 + tso2 + pH + so4 + alcohol  
DW = 1.75, p-value = 2.345e-07  
alternative hypothesis: true autocorrelation is greater than 0

As such I use bootstrap confidence intervals for coefficients.

One solution is to build covariance structure into the model via GLS, but it is not done here.

Now checking to see if we need to remove any data points, I first make a half-normal plot of leverage. There are two data points (152 and 259) that have high leverages and diverge substantially from the rest of the data.



Next, I compute leave-one-out residuals

> tail(jack[order(abs(jack))])

460 900 1506 1277 653 833

-3.284724 -3.349168 -3.434795 -3.550248 -3.678457 -4.185391

> qt(.05/(1599\*2),1599-8)

[1] -4.176048

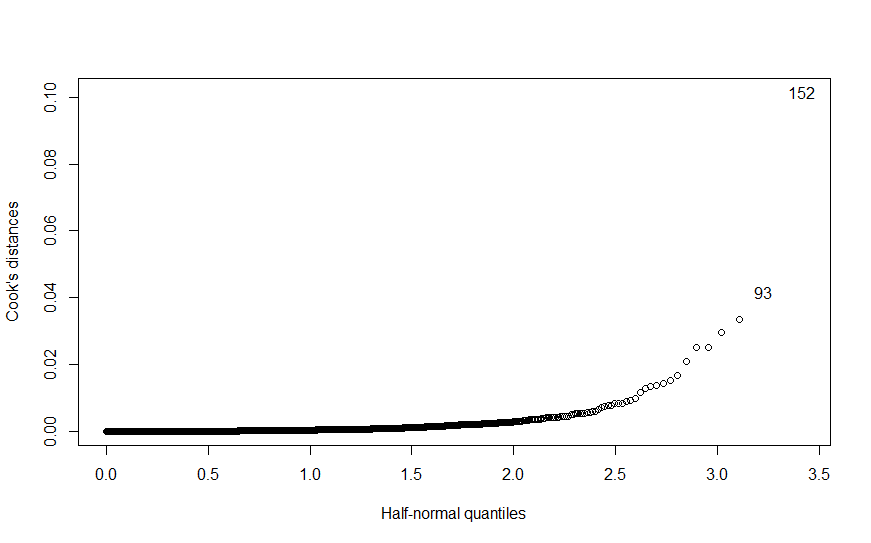
> jack[152]

152

-2.851774

Considering I used Bonferronni correction which is conservative in finding fewer outliers than the nominal level of confidence would dictate, the data point 833 is way beyond the confidence level.

Next we find the most influential points:



We try excluding this particular point 152, which also had the highest leverage. Recall that 152 had t-score of -2.85 which is quite high.

> summary(lm(quality ~ vacidity + chlorides + fso2 + tso2 + pH + so4 + alcohol, subset=(cook<max(cook))))

Call:

lm(formula = quality ~ vacidity + chlorides + fso2 + tso2 + pH +

so4 + alcohol, subset = (cook < max(cook)))

Residuals:

Min 1Q Median 3Q Max

-2.71375 -0.36843 -0.04987 0.46154 2.03385

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) 4.4002381 0.4021530 10.942 < 2e-16 \*\*\*

vacidity -1.0051906 0.1006526 -9.987 < 2e-16 \*\*\*

chlorides -1.7757498 0.4056341 -4.378 1.28e-05 \*\*\*

fso2 0.0053602 0.0021231 2.525 0.0117 \*

tso2 -0.0035385 0.0006855 -5.162 2.76e-07 \*\*\*

pH -0.4912605 0.1173342 -4.187 2.98e-05 \*\*\*

so4 0.9136403 0.1101995 8.291 2.37e-16 \*\*\*

alcohol 0.2904650 0.0167632 17.328 < 2e-16 \*\*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

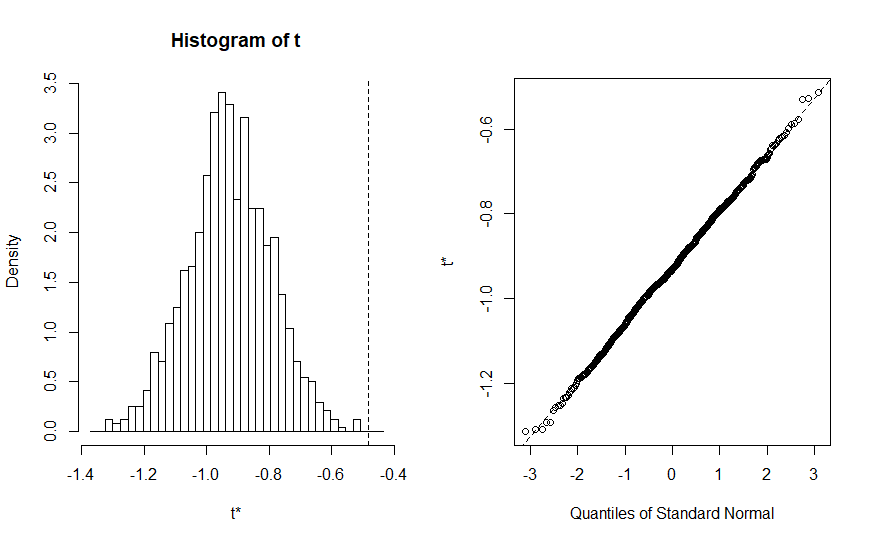
Residual standard error: 0.6463 on 1590 degrees of freedom

Multiple R-squared: 0.3611, Adjusted R-squared: 0.3583

F-statistic: 128.4 on 7 and 1590 DF, p-value: < 2.2e-16

I observed that there is no significant change in the model. P-values are roughly the same for predictors and the estimates do not change. As a final model I decided not to exclude any data point for the reason that they do not change the model substantially and removing outlier automatically without understanding physical context of data can be dangerous.

In order to estimate confidence interval, I use bootstrap method because it does not require any distributional assumptions and provide more accurate inferences when the data are not well behaved.



ORDINARY NONPARAMETRIC BOOTSTRAP

Call:

boot(data = wine, statistic = bs, R = 1000, formula = quality ~

vacidity + chlorides + fso2 + tso2 + pH + so4 + alcohol)

Bootstrap Statistics :

original bias std. error

t1\* 4.430098698 0.433583029 0.4640263454

t2\* -1.012752700 1.005001221 0.1023640512

t3\* -2.017813817 0.912322429 0.4125870793

t4\* 0.005077370 -0.004975567 0.0022538235

t5\* -0.003482245 0.003449912 0.0007027508

**t6\* -0.482661444 -0.444583503 0.1329182029**

t7\* 0.882665133 -0.877056724 0.1056710125

t8\* 0.289302753 0.088698475 0.0192353832

t6 represents the coefficient for pH.

The confidence interval is obtained via:

> lmod = lm(quality ~ vacidity + chlorides + fso2 + tso2 + pH + so4 + alcohol)

> preds = fitted(lmod)

> resids = residuals(lmod)

> vec=numeric(10000)

> for(i in 1:10000) {

+ ynew = preds + sample(resids, rep=TRUE)

+ vec[i]=summary(lm(ynew ~ vacidity + chlorides + fso2 + tso2 + pH + so4 + alcohol))$coef[6]

+ }

> par(mfrow=c(1,1))

> a = -0.48266

> length(vec[vec>a])/10000

[1] 0.4976

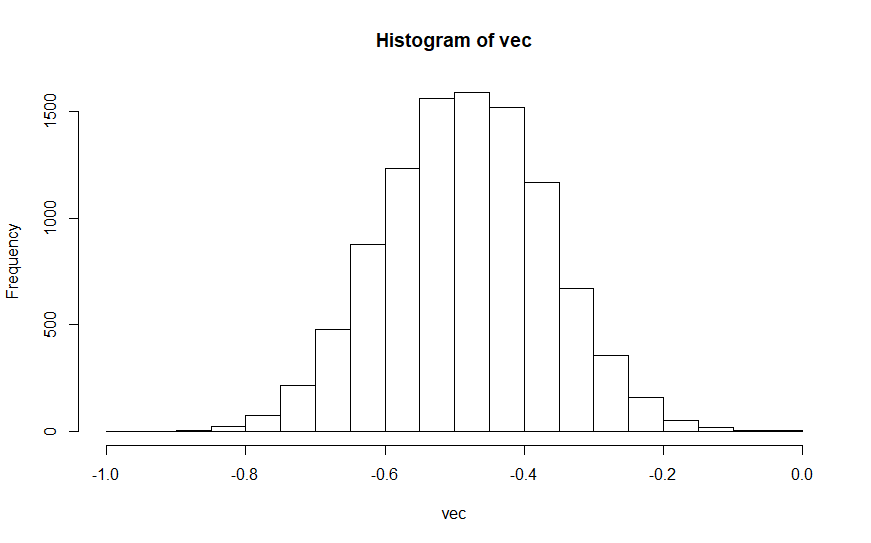
> sort(vec)[250]

**[1] -0.7126148**

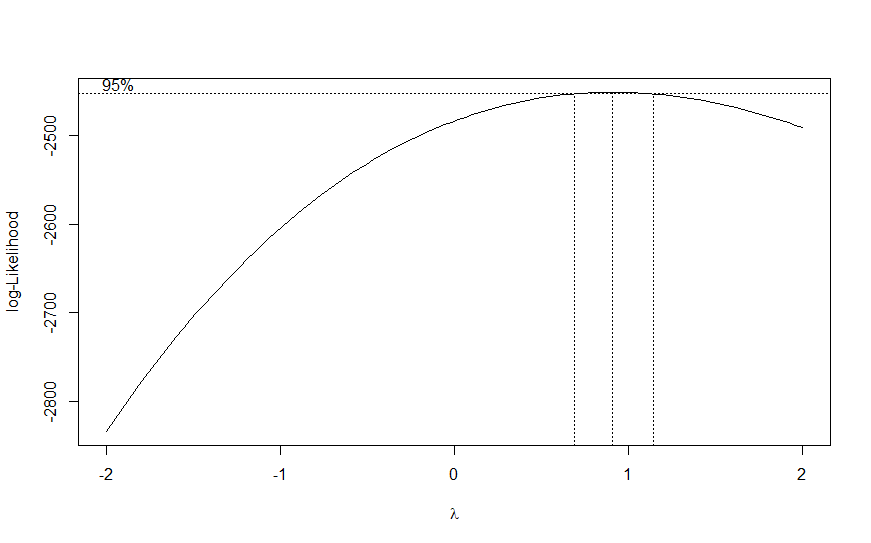
> sort(vec)[9750]

**[1] -0.2534445**

The confidence interval (95%) for the parameter is (-0.7126,-0.2534)



I look to find a suitable box-cox transformation that can change the model. Following is the log-likelihood of the box-coefficient parameter lambda.



The plot suggests that the box-cox parameter is nearly 1. To be more precise,

> head(cbind(boxcox(lmod)$x, boxcox(lmod)$y)[order(-boxcox(lmod)$y),])

[,1] [,2]

[1,] 0.9090909 -2451.054

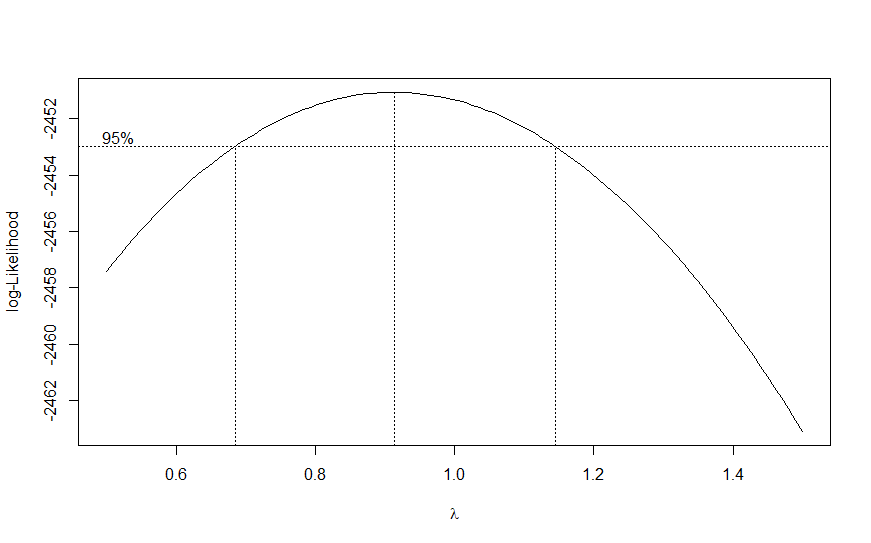
[2,] 0.9494949 -2451.103

[3,] 0.8686869 -2451.125

[4,] 0.9898990 -2451.269

[5,] 0.8282828 -2451.315

[6,] 1.0303030 -2451.552

The parameter is approximately 0.91. 

The 95% confidence interval runs from 0.7 to 1.15. There is no good reason to transform.

However, it might be a good idea to compare the transformed model with the original. With the techniques I learned in class, R-squared and AIC/BIC cannot be used to compare the models that are not nested or use the same set of data. Once the model is transformed by power transform or logarithm, response or predictor variable has different values, and R-squared cannot be compared between a model with untransformed Y and one with transformed Y. Not only that, we know the original model is flawed in that residuals are correlated and slightly non-normal, sowe cannot quite compute R-squared.

We are interested in formulating a model that has predictive power. As such, we compare how much less error a model. We can root-mean-squared/ cross-validation approach, where we compare the predictive power of each model.

Here is the box-cox transformed model with the power =0.91.

Call:

lm(formula = quality\_ ~ vacidity + chlorides + fso2 + tso2 +

pH + so4 + alcohol)

Residuals:

Min 1Q Median 3Q Max

-2.41485 -0.31422 -0.03772 0.40452 1.73805

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) 3.395289 0.351431 9.661 < 2e-16 \*\*\*

vacidity -0.888548 0.087957 -10.102 < 2e-16 \*\*\*

chlorides -1.758634 0.346743 -5.072 4.40e-07 \*\*\*

fso2 0.004448 0.001854 2.399 0.0165 \*

tso2 -0.003014 0.000599 -5.032 5.40e-07 \*\*\*

pH -0.418804 0.102536 -4.084 4.64e-05 \*\*\*

so4 0.766301 0.095864 7.994 2.50e-15 \*\*\*

alcohol 0.250546 0.014650 17.103 < 2e-16 \*\*\*

---

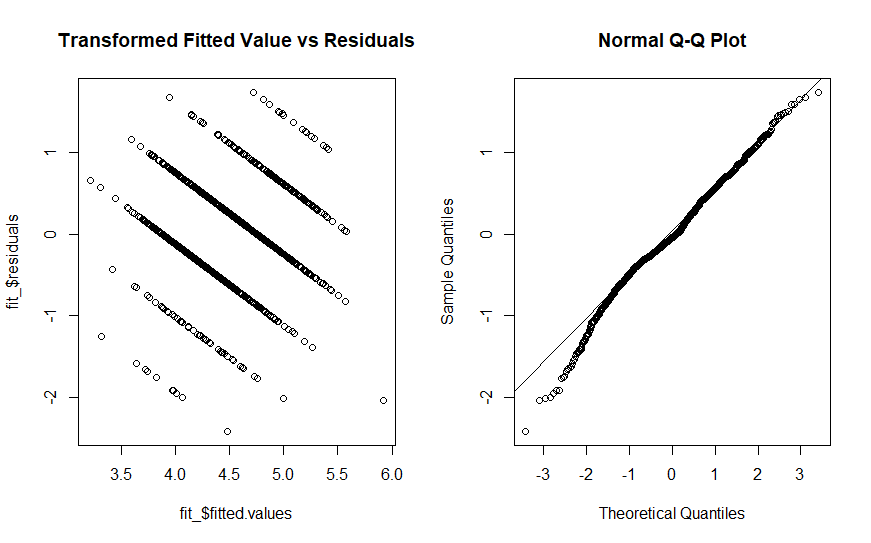
Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.565 on 1591 degrees of freedom

Multiple R-squared: 0.3579, Adjusted R-squared: 0.3551

F-statistic: 126.7 on 7 and 1591 DF, p-value: < 2.2e-16

Looking at the diagnostics, the transformation doesn’t significantly improve the assumptions.



> trainwine <- wine[1:1400,]

> testwine <- wine[1401:1599,]

>

> mod1 <- lm(quality ~ volatile.acidity + chlorides + free.sulfur.dioxide + total.sulfur.dioxide + pH + sulphates + alcohol, data=trainwine)

> mod2 <- lm(((quality -1)^0.91)/0.91 ~ volatile.acidity + chlorides + free.sulfur.dioxide + total.sulfur.dioxide + pH + sulphates + alcohol, data=trainwine)

>

> rmse(predict(mod1,testwine), testwine$quality)

[1] **0.6994192**

> rmse(predict(mod2,testwine), testwine$quality)

[1] **1.304175**

I noticed that the RMSE of the original model is smaller, which means it incurs smaller prediction error. I divided the data into 8 training/testing sets, computed rmse for each, and averaged them. Then I have 0.658for the first model and 1.367.

> (0.699 + 0.631+0.623 + 0.692 + 0.613 + 0.679 +0.666 + 0.662)/8

[1] 0.658125

> (1.304 + 1.269 + 1.495 + 1.412 + 1.296 + 1.425 + 1.53 + 1.207)/8

[1] 1.36725

There is a caution to this result however. The root mean squared contains error sigma-squared, and if they are substantially different, it becomes difficult to compare the root mean squared. Looking at the estimated residual standard error, we see that they are slightly different: 0.64 vs 0.56. However, the difference appears small enough that we can compare the root mean squared (data-driven comparison measure) to see which model is better in terms of prediction error given the data.

**II.**

To reiterate the problem and identify its components, we have four processes of production from A to D - labeled as “treat”,and the five types of blend are the fixed effect (which will be changed to random effect later)

We run regression of yield on the sum of dummy variables for treatment and blend, which R sets up automatically. Intercept is the reference level (or treat A), and the others are its relative magnitude.

> lmod <- lm(yield ~ treat + blend, penicillin)

> summary(lmod)

Call:

lm(formula = yield ~ treat + blend, data = penicillin)

Residuals:

Min 1Q Median 3Q Max

-5.00 -2.25 -0.50 2.25 6.00

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) 90.000 2.745 32.791 4.1e-13 \*\*\*

treatB 1.000 2.745 0.364 0.72194

treatC 5.000 2.745 1.822 0.09351 .

treatD 2.000 2.745 0.729 0.48018

blendBlend2 -9.000 3.069 -2.933 0.01254 \*

blendBlend3 -7.000 3.069 -2.281 0.04159 \*

blendBlend4 -4.000 3.069 -1.304 0.21686

blendBlend5 -10.000 3.069 -3.259 0.00684 \*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 4.34 on 12 degrees of freedom

Multiple R-squared: 0.5964, Adjusted R-squared: 0.361

F-statistic: 2.534 on 7 and 12 DF, p-value: 0.07535

> lmod\_ <- lm(yield ~ blend, penicillin)

> anova(lmod,lmod\_)

Analysis of Variance Table

Model 1: yield ~ treat + blend

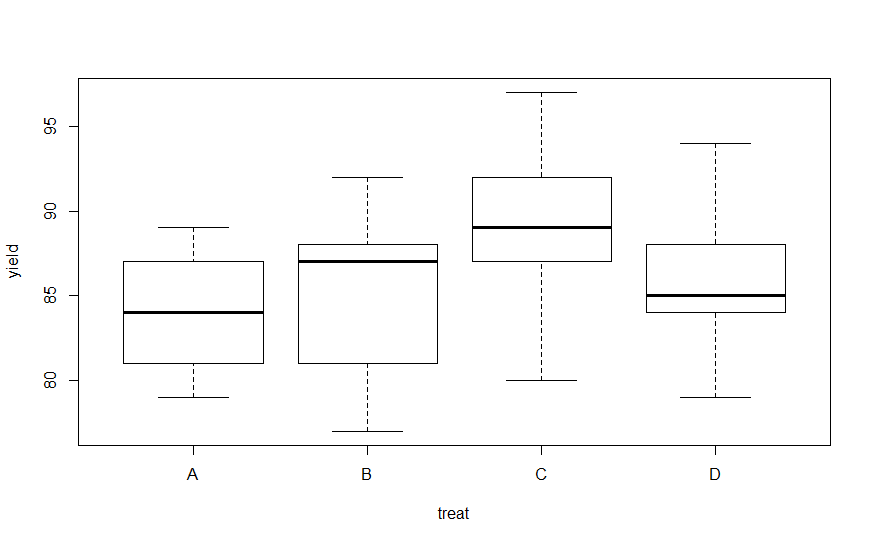
Model 2: yield ~ blend

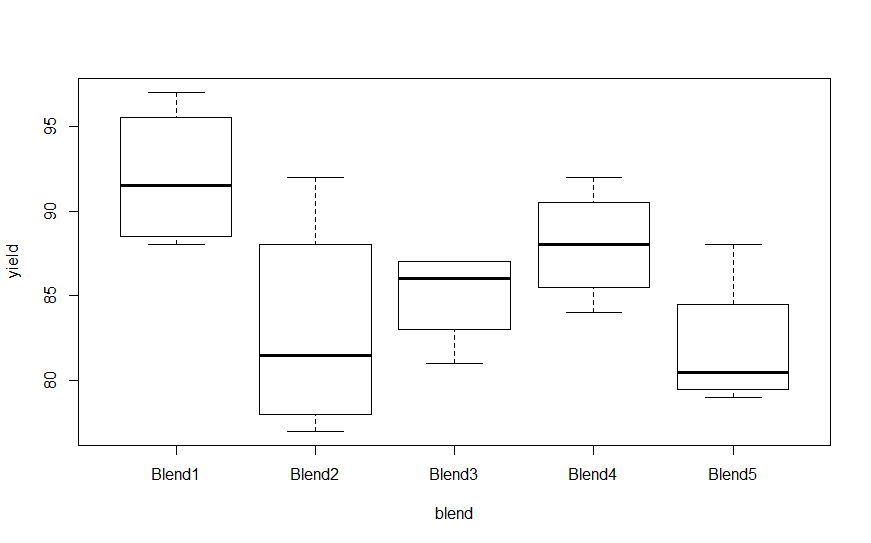
Res.Df RSS Df Sum of Sq F Pr(>F)

1 12 226

2 15 296 -3 -70 1.2389 0.3387

According to the summary of regression, none of the treatments is significant, which means none of the treatments is significantly different from A. When we look at the boxplots of each process with yield, the means are not very different from one another except C, whose p-value is low. On the other hand, there seems to be no block effect except blend 2, 3, and 5, which are significant. As seen in the boxplot, there seem to be significant difference in means among each blend. If we look at the anova test comparing the model without treatment and with treatment, the F-statistic p-value is high, so we can conclude the treatments are not significant.





Using blends increases efficiency as it reduces the variance of the model, where lmod is the blocked model whereas the unblocked model is lmod2 whose variance is underlined. The efficiency is 30.625/18.833 = 1.62 > 1,which implies it is easier to detect treatment effect under RCBD.

> anova(lmod)

Analysis of Variance Table

Response: yield

Df Sum Sq Mean Sq F value Pr(>F)

treat 3 70 23.333 1.2389 0.33866

blend 4 264 66.000 3.5044 0.04075 \*

Residuals 12 226 18.833

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

> anova(lmod2)

Analysis of Variance Table

Response: yield

Df Sum Sq Mean Sq F value Pr(>F)

treat 3 70 23.333 0.7619 0.5318

Residuals 16 490 30.625

Now I consider blend as the random effect whose noise is correlated with each block, so I am interested in the variance of this random effect.Using gls function simply as follows means we are considering both treatment and block as random variables, and is wrong.

> gls(yield ~ treat + blend, penicillin)

Generalized least squares fit by REML

Model: yield ~ treat + blend

Data: penicillin

Log-restricted-likelihood: -39.82778

Coefficients:

(Intercept) treatB treatC treatD blendBlend2 blendBlend3 blendBlend4 blendBlend5

90 1 5 2 -9 -7 -4 -10

Degrees of freedom: 20 total; 12 residual

Residual standard error: 4.339739

If we consider RCBD with one random effect and the variance of random effect, we can regress yield on treat with gls and treat its residual error as the sum of sigma-squared(variance of OLS) and v-squared (variance of random effect). Thus, we compute the residual variance of GLS and subtract by sigma-squared (variance of OLS), which is v-squared (variance of random effect). We have v^2 = 5.53^2 - 4.34^2 = 12.

> glmod <- gls(yield ~ treat, penicillin)

> summary(glmod) #5.533 = 30.61

Generalized least squares fit by REML

Model: yield ~ treat

Data: penicillin

AIC BIC logLik

116.5929 120.4558 -53.29643

Coefficients:

Value Std.Error t-value p-value

(Intercept) 84 2.474874 33.94113 0.0000

treatB 1 3.500000 0.28571 0.7788

treatC 5 3.500000 1.42857 0.1724

treatD 2 3.500000 0.57143 0.5756

Correlation:

(Intr) treatB treatC

treatB -0.707

treatC -0.707 0.500

treatD -0.707 0.500 0.500

Standardized residuals:

Min Q1 Med Q3 Max

-1.6263142 -0.5872801 0.0000000 0.5421047 1.4456126

Residual standard error: **5.533986**

Degrees of freedom: 20 total; 16 residual

> lmod <- lm(yield ~ treat + blend, penicillin)

> summary(lmod) #4.34 = 18.83

Call:

lm(formula = yield ~ treat + blend, data = penicillin)

Residuals:

Min 1Q Median 3Q Max

-5.00 -2.25 -0.50 2.25 6.00

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) 90.000 2.745 32.791 4.1e-13 \*\*\*

treatB 1.000 2.745 0.364 0.72194

treatC 5.000 2.745 1.822 0.09351 .

treatD 2.000 2.745 0.729 0.48018

blendBlend2 -9.000 3.069 -2.933 0.01254 \*

blendBlend3 -7.000 3.069 -2.281 0.04159 \*

blendBlend4 -4.000 3.069 -1.304 0.21686

blendBlend5 -10.000 3.069 -3.259 0.00684 \*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: **4.34** on 12 degrees of freedom

Multiple R-squared: 0.5964, Adjusted R-squared: 0.361

F-statistic: 2.534 on 7 and 12 DF, p-value: 0.07535

**III.**

The idea of transformation is attached. I implement the idea where I re-defined y\_ as y-x2 and x\_ as x1-x2, and obtain the coefficients as follows:

Call:

lm(formula = y\_ ~ x\_)

Coefficients:

(Intercept) x\_

**3.1956** **0.3668**

Call:

lm(formula = y\_ ~ x\_)

Residuals:

Min 1Q Median 3Q Max

-2.93143 -0.78183 0.01373 0.86331 2.49095

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) 3.19563 0.14641 21.827 < 2e-16 \*\*\*

x\_ 0.36676 0.09119 4.022 0.000114 \*\*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.1 on 98 degrees of freedom

Multiple R-squared: 0.1417, Adjusted R-squared: 0.1329

F-statistic: 16.18 on 1 and 98 DF, p-value: 0.0001136

**IV**

**R Code**

# 1

load("C:/Users/jihun/Downloads/winequality.RData")

quality <- wine$quality

facidity <- wine$fixed.acidity

vacidity <- wine$volatile.acidity

citric <- wine$citric.acid

rsugar <- wine$residual.sugar

chlorides <- wine$chlorides

fso2 <- wine$free.sulfur.dioxide

tso2 <- wine$total.sulfur.dioxide

density <- wine$density

pH <- wine$pH

so4 <- wine$sulphates

alcohol <- wine$alcohol

# linear model

lmod <- lm(quality ~ facidity + vacidity + citric + rsugar + chlorides + fso2 + tso2 + density + pH + so4 + alcohol)

summary(lmod)

# model selection

step(lmod)

lmod4 <- lm(quality ~ vacidity + chlorides + fso2 + tso2 + pH + so4 + alcohol)

summary(lmod4)

lmod=lmod4

# diagnostics

# homoscedasticity

par(mfrow=c(1,2))

plot(fitted(lmod),residuals(lmod),xlab="Fitted",ylab="Residuals",main="Residual Plot")

qqnorm(residuals(lmod))

qqline(residuals(lmod))

# normality

shapiro.test(residuals(lmod))

# independence

library(lmtest)

dwtest(quality ~ vacidity + chlorides + fso2 + tso2 + pH + so4 + alcohol)

# outliers

# check leverages

halfnorm(lm.influence(lmod)$hat,ylab="Leverages")

# outlier

jack <- rstudent(lmod)

jack[which.max(abs(jack))]

tail(jack[order(abs(jack))])

qt(.05/(1599\*2),1599-8)

jack[152]

# influence

cook <- cooks.distance(lmod)

halfnorm(cook, ylab="Cook's distances")

summary(lm(quality ~ vacidity + chlorides + fso2 + tso2 + pH + so4 + alcohol, subset=(cook<max(cook))))

# bootstrap

library(boot)

# function to obtain regression weights

bs <- function(formula, data, indices) {

d <- data[indices,] # allows boot to select sample

fit <- lm(formula, data=d)

return(coef(fit))

}

# bootstrapping with 1000 replications

results <- boot(data=wine, statistic=bs,

R=1000, formula=quality ~ vacidity + chlorides + fso2 + tso2 + pH + so4 + alcohol)

# view results

results

plot(results, index=6) # pH

lmod = lm(quality ~ vacidity + chlorides + fso2 + tso2 + pH + so4 + alcohol)

preds = fitted(lmod)

resids = residuals(lmod)

vec=numeric(10000)

for(i in 1:10000) {

ynew = preds + sample(resids, rep=TRUE)

vec[i]=summary(lm(ynew ~ vacidity + chlorides + fso2 + tso2 + pH + so4 + alcohol))$coef[6]

}

par(mfrow=c(1,1))

hist(vec)

a = -0.48266

length(vec[vec>a])/10000

sort(vec)[250]

sort(vec)[9750]

# Box-cox transformation

library(MASS)

boxcox(lmod, plotit=T)

head(cbind(boxcox(lmod)$x, boxcox(lmod)$y)[order(-boxcox(lmod)$y),])

boxcox(lmod,plotit=T,lambda=seq(0.5,1.5,by=0.1))

# data-driven approach

quality\_ = ((quality -1)^0.91)/0.91

fit\_ = lm(quality\_ ~ vacidity + chlorides + fso2 + tso2 + pH + so4 + alcohol)

summary(fit\_)

par(mfrow=c(1,2))

plot(fit\_$fitted.values,fit\_$residuals,main="Transformed Fitted Value vs Residuals")

qqnorm(fit\_$residuals)

qqline(residuals(fit\_))

library(Metrics)

trainwine <- wine[1:1400,]

testwine <- wine[1401:1599,]

mod1 <- lm(quality ~ volatile.acidity + chlorides + free.sulfur.dioxide + total.sulfur.dioxide + pH + sulphates + alcohol, data=trainwine)

mod2 <- lm(((quality -1)^0.91)/0.91 ~ volatile.acidity + chlorides + free.sulfur.dioxide + total.sulfur.dioxide + pH + sulphates + alcohol, data=trainwine)

rmse(predict(mod1,testwine), testwine$quality) # 0.6524137

rmse(predict(mod2,testwine), testwine$quality) # 1.283869

trainwine2 <- wine[-c(1201:1400),]

testwine2 <- wine[c(1201:1400),]

mod21 <- lm(quality ~ volatile.acidity + chlorides + free.sulfur.dioxide + total.sulfur.dioxide + pH + sulphates + alcohol, data=trainwine2)

mod22 <- lm(((quality -1)^0.91)/0.91 ~ volatile.acidity + chlorides + free.sulfur.dioxide + total.sulfur.dioxide + pH + sulphates + alcohol, data=trainwine2)

rmse(predict(mod21,testwine2), testwine2$quality) # 0.631

rmse(predict(mod22,testwine2), testwine2$quality) # 1.269

trainwine3 <- wine[-c(1001:1200),]

testwine3 <- wine[c(1001:1200),]

mod31 <- lm(quality ~ volatile.acidity + chlorides + free.sulfur.dioxide + total.sulfur.dioxide + pH + sulphates + alcohol, data=trainwine3)

mod32 <- lm(((quality -1)^0.91)/0.91 ~ volatile.acidity + chlorides + free.sulfur.dioxide + total.sulfur.dioxide + pH + sulphates + alcohol, data=trainwine3)

rmse(predict(mod31,testwine3), testwine3$quality) # 0.623

rmse(predict(mod32,testwine3), testwine3$quality) # 1.495

trainwine4 <- wine[-c(801:1000),]

testwine4 <- wine[c(801:1000),]

mod41 <- lm(quality ~ volatile.acidity + chlorides + free.sulfur.dioxide + total.sulfur.dioxide + pH + sulphates + alcohol, data=trainwine4)

mod42 <- lm(((quality -1)^0.91)/0.91 ~ volatile.acidity + chlorides + free.sulfur.dioxide + total.sulfur.dioxide + pH + sulphates + alcohol, data=trainwine4)

rmse(predict(mod41,testwine4), testwine4$quality) # 0.692

rmse(predict(mod42,testwine4), testwine4$quality) # 1.412

trainwine5 <- wine[-c(601:800),]

testwine5 <- wine[c(601:800),]

mod51 <- lm(quality ~ volatile.acidity + chlorides + free.sulfur.dioxide + total.sulfur.dioxide + pH + sulphates + alcohol, data=trainwine5)

mod52 <- lm(((quality -1)^0.91)/0.91 ~ volatile.acidity + chlorides + free.sulfur.dioxide + total.sulfur.dioxide + pH + sulphates + alcohol, data=trainwine5)

rmse(predict(mod51,testwine5), testwine5$quality) # 0.613

rmse(predict(mod52,testwine5), testwine5$quality) #1.296

trainwine6 <- wine[-c(401:600),]

testwine6 <- wine[c(401:600),]

mod61 <- lm(quality ~ volatile.acidity + chlorides + free.sulfur.dioxide + total.sulfur.dioxide + pH + sulphates + alcohol, data=trainwine6)

mod62 <- lm(((quality -1)^0.91)/0.91 ~ volatile.acidity + chlorides + free.sulfur.dioxide + total.sulfur.dioxide + pH + sulphates + alcohol, data=trainwine6)

rmse(predict(mod61,testwine6), testwine6$quality) # 0.679

rmse(predict(mod62,testwine6), testwine6$quality) # 1.425

trainwine7 <- wine[-c(201:400),]

testwine7 <- wine[c(201:400),]

mod71 <- lm(quality ~ volatile.acidity + chlorides + free.sulfur.dioxide + total.sulfur.dioxide + pH + sulphates + alcohol, data=trainwine7)

mod72 <- lm(((quality -1)^0.91)/0.91 ~ volatile.acidity + chlorides + free.sulfur.dioxide + total.sulfur.dioxide + pH + sulphates + alcohol, data=trainwine7)

rmse(predict(mod71,testwine7), testwine7$quality) # 0.666

rmse(predict(mod72,testwine7), testwine7$quality) # 1.53

trainwine8 <- wine[-c(1:200),]

testwine8 <- wine[c(1:200),]

mod81 <- lm(quality ~ volatile.acidity + chlorides + free.sulfur.dioxide + total.sulfur.dioxide + pH + sulphates + alcohol, data=trainwine8)

mod82 <- lm(((quality -1)^0.91)/0.91 ~ volatile.acidity + chlorides + free.sulfur.dioxide + total.sulfur.dioxide + pH + sulphates + alcohol, data=trainwine8)

rmse(predict(mod81,testwine8), testwine8$quality) # 0.662

rmse(predict(mod82,testwine8), testwine8$quality) # 1.207

library(analogue)

pcrmod <- pcr(quality ~ volatile.acidity + chlorides +

free.sulfur.dioxide + total.sulfur.dioxide +

pH + sulphates + alcohol, data=trainwine,

validation="CV",ncomp=50)

rmse(predict(pcrmod, testwine$quality))

# 2

library(faraway)

data(penicillin)

penicillin

plot(yield ~ treat, penicillin,pch=unclass(blend))

plot(yield ~ blend, penicillin,pch=unclass(treat))

# RCBD

lmod <- lm(yield ~ treat + blend, penicillin)

summary(lmod)

lmod\_ <- lm(yield ~ blend, penicillin)

anova(lmod,lmod\_)

# CRD

lmod2 <- lm(yield ~ treat, penicillin)

summary(lmod2)

anova(lmod2)

# random effect

library(nlme)

gls(yield ~ treat + blend, penicillin)

glmod <- gls(yield ~ treat, penicillin)

summary(glmod) #5.533 = 30.61

lmod <- lm(yield ~ treat + blend, penicillin)

summary(lmod) #4.34 = 18.83

# 3

load("C:/Users/jihun/Downloads/Constrained.RData")

y\_ = y - x2

x\_ = x1 - x2

summary(lm(y\_ ~ x\_))