Lab09

2023-04-03

Introduction

In this lab, we will use linear regression to predict the red wine quality using physicochemical tests scores such as citric acid, pH, etc.

But first, a review of using the `lm()` and `predict()` functions.

```
x1 = rnorm(100)
x2 = rnorm(100)
y= 2*x1 + x2 + rnorm(100)
lm_out = lm(y~x1 + x2)
summary(lm_out)
```

```
##
## Call:
## lm(formula = y ~ x1 + x2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.78948 -0.63856 -0.07625  0.66045  2.97877
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.1766     0.1094   1.614     0.11
## x1            1.9379     0.1175  16.487  < 2e-16 ***
## x2            1.0031     0.1127   8.902 3.14e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.085 on 97 degrees of freedom
## Multiple R-squared:  0.8012, Adjusted R-squared:  0.7971
## F-statistic: 195.5 on 2 and 97 DF,  p-value: < 2.2e-16
```

Calculate prediction for y when x1 is 1 and x2 is 0.5.

```
lm_out$coefficients[1] + lm_out$coefficients[2]* 1 + lm_out$coefficients[3]* 0.5
```

```
## (Intercept)
##    2.616033
```

Another way to do this.

```
predict(lm_out, newdata = data.frame(x1= 1, x2= 0.5))
```

```
##        1
## 2.616033
```

Can do several at once.

```
predict(lm_out, newdata = data.frame(x1= c(1, 2), x2= c(0.5, -1) ))
```

```
##        1        2 
## 2.616033 3.049395
```

A few other notes on regression.

1) If you try to predict one variable and include a perfectly correlated variable in the prediction set, then that variable will be perfectly fit to the outcome to the exclusion of all others.

```
perf_cor = y/4
summary(lm( y ~ perf_cor + x1 + x2 ))
```

```
## 
## Call:
## lm(formula = y ~ perf_cor + x1 + x2)
## 
## Residuals:
##        Min         1Q     Median         3Q        Max 
## -1.573e-15 -4.012e-16 -2.021e-16  3.760e-17  1.905e-14 
## 
## Coefficients:
##               Estimate Std. Error    t value Pr(>|t|)    
## (Intercept) -2.665e-16  2.028e-16 -1.314e+00    0.192    
## perf_cor     4.000e+00  7.428e-16  5.385e+15   <2e-16 ***
## x1          -9.446e-17  4.192e-16 -2.250e-01    0.822    
## x2           2.557e-16  2.778e-16  9.200e-01    0.360    
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 1.984e-15 on 96 degrees of freedom
## Multiple R-squared:      1,  Adjusted R-squared:      1 
## F-statistic: 4.863e+31 on 3 and 96 DF,  p-value: < 2.2e-16
```

2) If there are more variables used for prediction than there are observations, `lm` will only keep the first n-1 variables.

```
x3= rnorm(100)
x4= rnorm(100)
x5= rnorm(100)

all_x = data.frame(x1,x2,x3,x4,x5, y) # 100 x 6 df
lm(y~ . ,data= all_x[1:4,]) # only use the first 4 observations (4<5)
```

```
## 
## Call:
## lm(formula = y ~ ., data = all_x[1:4, ])
## 
## Coefficients:
## (Intercept)           x1           x2           x3           x4           x5  
##       1.146       -3.721        1.408       -1.448           NA           NA
```

```
# Then lm only use the first 4-1=3 variables
```

Wine data

The wine dataset is related to red variants of the Portuguese “Vinho Verde” wine. There are 1599 samples available in the dataset. 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 explanatory variables are all continuous variables based on physicochemical tests:

- **fixed acidity**
- **volatile acidity**
- **citric acid**
- **residual sugar**
- **chlorides**
- **free sulfur dioxide**
- **total sulfur dioxide**
- **density**
- **pH**
- **sulphates**
- **alcohol**

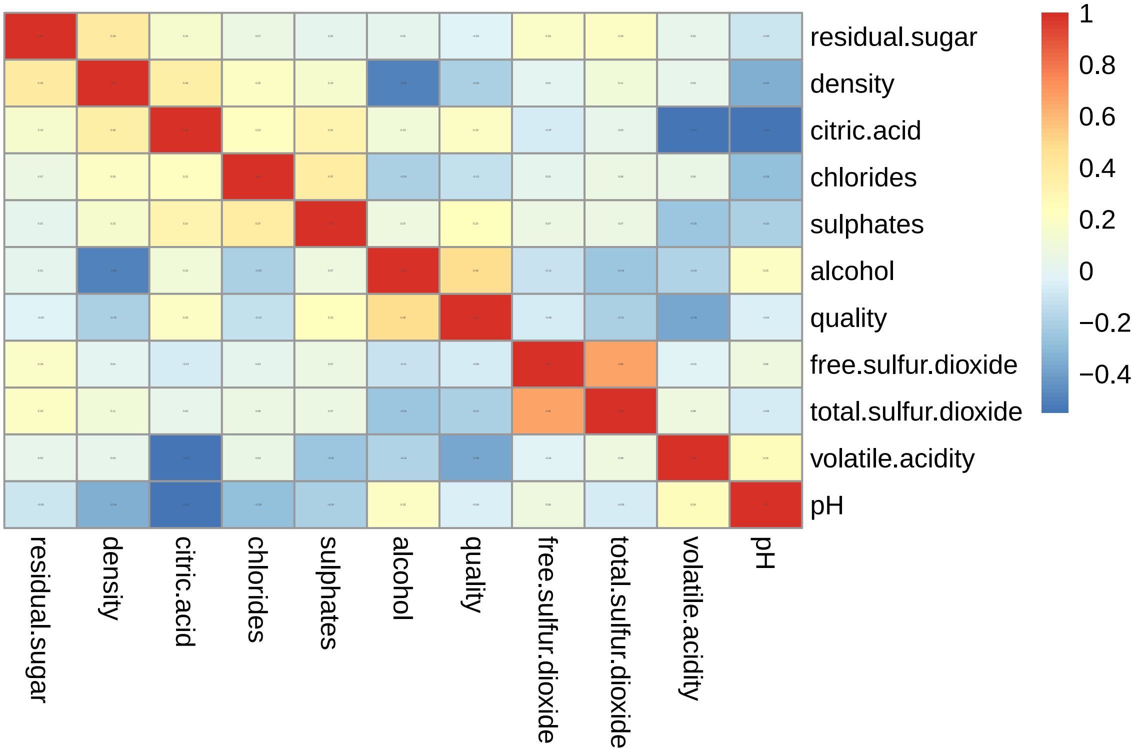
The response variable is the **quality** score between 0 and 10 (based on sensory data).

Read data. We randomly split the data into two parts-the `wine` dataset with 1199 samples and the `wine.test` dataset with 400 samples. Splitting the dataset is a common technique when we want to evaluate the model performance. There are training set, validation set, and test set. The validation set is used for model selection. That is, to estimate the performance of the different model in order to choose the best one. The test set is used for estimating the performance of our final model.

```
## Read data
set.seed("2022")
wine.dataset <- read.csv("winequality-red.csv", sep = ";")
test.samples <- sample(1:nrow(wine.dataset), 400)
wine <- wine.dataset[-test.samples, ]
wine.test <- wine.dataset[test.samples, ]
```

To check the correlation between explanatory variables:

```
library(pheatmap)
corr.wine <- cor(wine[, -1])
pheatmap(corr.wine, treeheight_row = 0, treeheight_col = 0,
         display_numbers = T, fontsize_number = 0.5)
```



We now fit the linear regression using all of the explanatory variables:

```
wine.fit <- lm(quality ~. ,data = na.omit(wine))
summary(wine.fit)
```

```
##
## Call:
## lm(formula = quality ~ ., data = na.omit(wine))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.67557 -0.35919 -0.04682  0.46000  2.04930
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)          12.5643244 24.4761128   0.513   0.6078
## fixed.acidity         0.0045448  0.0299388   0.152   0.8794
## volatile.acidity     -1.0737942  0.1406020  -7.637 4.55e-14 ***
## citric.acid          -0.1394933  0.1661915  -0.839   0.4014
## residual.sugar        0.0007192  0.0175333   0.041   0.9673
## chlorides            -2.0052493  0.4682338  -4.283 2.00e-05 ***
## free.sulfur.dioxide   0.0061072  0.0025030   2.440   0.0148 *
## total.sulfur.dioxide -0.0038532  0.0008402  -4.586 4.99e-06 ***
## density              -8.0141715 24.9850874  -0.321   0.7485
## pH                   -0.4859053  0.2198946  -2.210   0.0273 *
## sulphates             0.8267058  0.1259283   6.565 7.77e-11 ***
## alcohol               0.2822350  0.0302018   9.345  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 0.6496 on 1187 degrees of freedom
## Multiple R-squared:  0.356,  Adjusted R-squared:   0.35
## F-statistic: 59.64 on 11 and 1187 DF,  p-value: < 2.2e-16
```

Multiple regression with diamond price data

This is a very large data set showing various factors of over 50,000 diamonds including price, cut, color, clarity, etc. We are interested in diamond price `price` and how different factors influence it.

Variable	Description
price	price in US dollars ($326–$18,823)
carat	weight of the diamond (0.2–5.01)
cut	quality of the cut (Fair, Good, Very Good, Premium, Ideal)
color	diamond colour, from J (worst) to D (best)
clarity	how clear the diamond is (I1 (worst), SI1, SI2, VS1, VS2, VVS1, VVS2, IF (best))
length.in.mm	length in mm (0–10.74)
width.of.mm	width in mm (0–58.9)
depth.in.mm	depth in mm (0–31.8)
depth	total depth percentage $ = z / mean(x, y) = 2 * z / (x+ y) (43–79)$
table	width of top of diamond relative to widest point (43–95)

```
diamonds <- read.csv("diamonds.csv")
head(diamonds)
```

```
##   carat       cut color clarity depth table price length.in.mm width.of.mm
## 1  0.23     Ideal     E     SI2  61.5    55   326         3.95        3.98
## 2  0.21   Premium     E     SI1  59.8    61   326         3.89        3.84
## 3  0.23      Good     E     VS1  56.9    65   327         4.05        4.07
## 4  0.29   Premium     I     VS2  62.4    58   334         4.20        4.23
## 5  0.31      Good     J     SI2  63.3    58   335         4.34        4.35
## 6  0.24 Very Good     J    VVS2  62.8    57   336         3.94        3.96
##   depth.in.mm
## 1        2.43
## 2        2.31
## 3        2.31
## 4        2.63
## 5        2.75
## 6        2.48
```

Multiple regression with continuous variable

Exercise 1 Fit the model and Calculate the Statistics

(a) Fit a linear model to price with all the continuous variables as explanatory variables. Print the summary of your model.

```
# Insert you code here, save your model as `fit`
```

(b) Calculate the fitted values.

```
# Insert you code here, save your results as `fitted.value`
```

(c) Using the fitted model, we can write the estimated model formula. How do we interpret this equation? *Hint.* Use `summary()` on the model object.

```
# Insert your answer
```

Excercise 2

Let's use the wine data for this exercise

```
head(wine)
```

```
##   fixed.acidity volatile.acidity citric.acid residual.sugar chlorides
## 1           7.4             0.70        0.00            1.9     0.076
## 2           7.8             0.88        0.00            2.6     0.098
## 4          11.2             0.28        0.56            1.9     0.075
## 5           7.4             0.70        0.00            1.9     0.076
## 6           7.4             0.66        0.00            1.8     0.075
## 8           7.3             0.65        0.00            1.2     0.065
##   free.sulfur.dioxide total.sulfur.dioxide density   pH sulphates alcohol
## 1                  11                   34  0.9978 3.51      0.56     9.4
## 2                  25                   67  0.9968 3.20      0.68     9.8
## 4                  17                   60  0.9980 3.16      0.58     9.8
## 5                  11                   34  0.9978 3.51      0.56     9.4
## 6                  13                   40  0.9978 3.51      0.56     9.4
## 8                  15                   21  0.9946 3.39      0.47    10.0
##   quality
## 1       5
## 2       5
## 4       6
## 5       5
## 6       5
## 8       7
```

We now fit the linear regression using all of the explanatory variables:

```
wine.fit <- lm(quality ~. ,data = na.omit(wine))
summary(wine.fit)
```

```
## 
## Call:
## lm(formula = quality ~ ., data = na.omit(wine))
## 
## Residuals:
##      Min       1Q   Median       3Q      Max 
## -2.67557 -0.35919 -0.04682  0.46000  2.04930 
## 
## Coefficients:
##                        Estimate Std. Error t value Pr(>|t|)    
## (Intercept)          12.5643244 24.4761128   0.513   0.6078    
## fixed.acidity         0.0045448  0.0299388   0.152   0.8794    
## volatile.acidity     -1.0737942  0.1406020  -7.637 4.55e-14 ***
## citric.acid          -0.1394933  0.1661915  -0.839   0.4014    
## residual.sugar        0.0007192  0.0175333   0.041   0.9673    
## chlorides            -2.0052493  0.4682338  -4.283 2.00e-05 ***
## free.sulfur.dioxide   0.0061072  0.0025030   2.440   0.0148 *  
## total.sulfur.dioxide -0.0038532  0.0008402  -4.586 4.99e-06 ***
## density              -8.0141715 24.9850874  -0.321   0.7485    
## pH                   -0.4859053  0.2198946  -2.210   0.0273 *  
## sulphates             0.8267058  0.1259283   6.565 7.77e-11 ***
## alcohol               0.2822350  0.0302018   9.345  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6496 on 1187 degrees of freedom
## Multiple R-squared:  0.356,  Adjusted R-squared:   0.35
## F-statistic: 59.64 on 11 and 1187 DF,  p-value: < 2.2e-16
```

Confidence Interval

(a) Calculate the confidence interval for all the coefficients from the regression done above. Which of these factors will positively influence the wine quality?

```
# Insert your code here to calculate the confidence intervals for the regression coefficients.
```

(b) Calculate the confidence intervals for the samples in `wine.test` using the model you just fit. Which confidence interval will you use? Confidence intervals for the average response or the prediction interval?

```
# insert your code here and save your confidence intervals as `wine.confint`
# wine.confint <-
```

(c) What is the percentage that your interval in (b) covers the true **quality** score in `wine.test`? What if you use the other confidence interval? Which one is consistent with your confidence level?

```
# insert your code here and save your percentage as `pct.covered`
# pct.covered <-
# pct.covered
# insert your code here and save your percentage calculated
# using the other confidence interval as `pct.covered.other`
# wine.confint.other <-
# pct.covered.other <-
# pct.covered.other
```

Exercise 3 Bootstrap CI

Scale the columns of the dataset using scale() and then make 95% bootstrap confidence intervals for the coefficients for the predictors. Plot these confidence intervals using the plotCI() function in gplots. Code from professor for making bootstrap CI is included. You can use this or write your own code. Use the wine subset as used above.

```
bootstrapLM <- function(y,x, repetitions, confidence.level=0.95){
    # calculate the observed statistics
  stat.obs <- coef(lm(y~., data=x))
  # calculate the bootstrapped statistics
  bootFun<-function(){
          sampled <- sample(1:length(y), size=length(y),replace = TRUE)
          coef(lm(y[sampled]~.,data=x[sampled,])) #small correction here to make it for a matrix x
  }
  stat.boot<-replicate(repetitions,bootFun())
  # nm <-deparse(substitute(x))
  # row.names(stat.boot)[2]<-nm
  level<-1-confidence.level
  confidence.interval <- apply(stat.boot,1,quantile,probs=c(level/2,1-level/2))
    return(list(confidence.interval = cbind("lower"=confidence.interval[1,],"estimate"=stat.obs,"upper"=
}
```

```
# insert your code here
```