STATS 3001 Statistical Modelling III
Practical 1
Week 1, Semester 1, 2018

The following exercises are intended to review the basic regression calculation and matrix manipulation functions in R. For the purpose of this exercise, we will use built-in data set `Rubber` that is provided with the `MASS` library. The data set comprises three variables recorded on thirty samples of rubber that were being tested for durability:

`loss`: The abrasion loss in grams per hour;

`hard`: The hardness in Shore units; and

`tens`: The tensile strength in kg per square metre.

(1) Start RStudio, an IDE[1] for R.

(2) Click on `File` ⇒ `New File` ⇒ `R Script`. This will create a new blank R script file (a text file containing a sequence R commands). We will use this file to be a record of the commands used in this practical.

(3) Save the R script as `prac1.R` into a `stat-modelling` folder on your `U:` drive.

(4) Load the package `MASS` by using the command:

```
library(MASS)
```

Write (or copy) this command into your R script. To send commmands to the console (where commands are evaluated by R) simply have the cursor on the relevant line in the R script and press `Ctrl+Enter`.

(5) Also load the `Rubber` data with the command:

```
data(Rubber)
```

Now `Rubber` is an object that R has stored in memory to call upon. If you use the command `objects()`, the console will list all objects visible to R. RStudio handily lists available objects in the `Environment` pane.

(6) You can look at the data by typing `Rubber`, or in the case of big datasets use the `head()` function to give the first 6 rows. You may prefer to use the `View()` command in RStudio.

```
head(Rubber) # or View(Rubber)

##   loss hard tens
## 1  372   45  162
## 2  206   55  233
## 3  175   61  232
## 4  154   66  231
## 5  136   71  231
## 6  112   71  237
```

[1] https://en.wikipedia.org/wiki/Integrated_development_environment

(7) Use the command `pairs(Rubber)` to obtain a scatter plot matrix for the data. What are the apparent patterns of association between each pair of variables? Are these what you would expect?

(8) The `lm()` function is used in R to fit linear models. More information about this function can be obtained by typing `help(lm)` or `?lm`.

(a) Taking `loss` as the outcome variable and `hard` and `tens` as predictors, the model

$$E(\texttt{loss}_i) = \beta_0 + \beta_1 \times \texttt{hard}_i + \beta_2 \times \texttt{tens}_i,$$

is fit using the command:

```
lm(loss~hard+tens,data=Rubber)
```

(b) A more informative output is produced by applying the `summary()` function to the object produced by `lm()`. This can be done by saving the object first and then applying the `summary()` to the object:

```
rubber.lm <- lm(loss~hard+tens,data=Rubber)
summary(rubber.lm)
```

or by passing the result of `lm()` directly to `summary()`:

```
summary(lm(loss~hard+tens,data=Rubber))
```

With the output produced, identify the regression coefficients and their standard errors, the residual standard error $s_e = 36.49$ and the F-statistic (71.0) for testing $H_0 : \beta_1 = \beta_2 = 0$.

(9) Residuals are important for model checking. The residuals and fitted values can be extracted from the result of `lm()` using the `residuals()` and `fitted()` functions, and plotted in the usual ways.

```
par(mfrow=c(2,2)) # make the plot window a 2x2 lattice of plots
rubber.resid <- residuals(rubber.lm)
rubber.fits <- fitted(rubber.lm)
plot(rubber.fits,rubber.resid)
plot(Rubber$hard,rubber.resid)
plot(Rubber$tens,rubber.resid)
qqnorm(rubber.resid)
```

Obtain the plots as described above and decide whether the regression model is appropriate for these data. Give reasons for your answer.

(10) Predictions can be calculated from an `lm` object using the `predict()` function. It is necessary first to create a `data.frame` containing the x-values for which we want to predict. Suppose for example, we want to predict loss for two samples, one with $\texttt{hard} = \texttt{50}, \texttt{tens} = \texttt{200}$ and the other with $\texttt{hard} = \texttt{65}, \texttt{tens} = \texttt{190}$.

(a) The data frame can be constructed as shown below. Note, it is **essential** that the names used in the data frame are identical to those used in the `lm` fit.

```
rubber.new <- data.frame(hard=c(50,65),tens=c(200,190))
rubber.new

##   hard tens
## 1   50  200
## 2   65  190
```

(b) The basic command for prediction produces only point predictions.

```
predict(rubber.lm,newdata=rubber.new)

##        1        2
## 281.7573 196.9379
```

(c) More useful output can be generated by providing optional arguments to the `predict()` function (see `?predict`).

```
# Give the standard errors for the point predictions.
predict(rubber.lm,newdata=rubber.new,se.fit=TRUE)
# Calculate 95% confidence intervals
predict(rubber.lm,newdata=rubber.new,interval="confidence")
# Calculate 95% prediction intervals
predict(rubber.lm,newdata=rubber.new,interval="prediction")
```

(d) Compare the resultant confidence intervals and prediction intervals and comment. Use the output produced when using the `se.fit=TRUE` optional argument to construct the confidence intervals by hand, and check that they agree with those obtained when using the `interval="confidence"` optional argument. Hint, to find $t_{27}(0.025)$ in R, use the command `qt(0.025,df=27)`.

(11) This question is concerned with looking at the calculations happening behind the scenes with the built-in R functions used above.

(a) The design matrix X can be extracted from the lm object using the `model.matrix` function.

```
X <- model.matrix(rubber.lm)
```

Obtain the design matrix X and compare it to the values in the original `Rubber` data frame.

Extract the response variable $\boldsymbol{y}$ from the original data using:

```
y <- Rubber$loss
```

(b) In class, we saw that the least squares estimate is $\hat{\boldsymbol{\beta}} = (X^T X)^{-1} X^T \boldsymbol{y}$. The same quantity can be calculated in R using its matrix functions and compared to the result of `lm()`.
Note: In R the following operators and functions are available:

Operator	Meaning
`%*%`	matrix multiplication
`*`	scalar multiplication (elementwise)
`solve()`	matrix inversion
`t()`	transpose

(c) Calculate the fitted values, $\hat{\boldsymbol{\eta}} = X\hat{\boldsymbol{\beta}}$ using matrix operations in R.

(d) Calculate the residual variance, s_e^2 directly from the observed and fitted values. Compare the result to the `residual standard error` produced by `lm()`. (Remember to take the square-root).

(e) Calculate the estimated variance matrix for $\hat{\boldsymbol{\beta}}$ from the formula $V = s_e^2 (X^T X)^{-1}$. Compare this to the result of the built-in calculation `vcov(rubber.lm)`.

(f) Check that the square roots of the diagonal elements of `V` agree with the standard errors for the regression coefficients obtained from `lm()`. Hint, you can use `sqrt(diag(V))`.

(g) Type in a suitable matrix X_0 of x-values and use matrix multiplication to verify the predicted values and their standard errors from question (10)c. Hints: The X_0 will need to include a column of `1`s for the intercept coefficient. The variance matrix for the predicted values is obtained using the formula:

$$\mathrm{Var}\left(X_0\hat{\boldsymbol{\beta}}\right) = X_0\mathrm{Var}\left(\hat{\boldsymbol{\beta}}\right)X_0^T = X_0 V X_0^T.$$