Introduction to Data Science

Ryan G

February 12, 2020

Contents

Simple Linear Regression IntroDataSci

Material of Tue 12 2019, week 2

Load Packages

```
# Load Packages
if(require('pacman')){
  library('pacman')
}else{
  install.packages('pacman')
  library('pacman')
}

pacman::p_load(caret, scales, ggplot2, rmarkdown, shiny, ISLR, class, BiocManager,
               corrplot, plotly, tidyverse, latex2exp, stringr, reshape2, cowplot, ggpubr,
               rstudioapi, wesanderson, RColorBrewer, colorspace, gridExtra, grid, car,
               boot, colourpicker, tree, ggtree, mise, rpart, rpart.plot, knitr, MASS,
               magrittr, EnvStats,tidyverse,tidyr,devtools, bookdown, leaps, car, clipr,
               tikzDevice, e1071)



mise()
set.seed(0932)
```

Question 1

(a) Import the Data ATTACH

```
setwd("/home/ryan/Dropbox/Notes/DataSci/IntroDataSci/Org-Babel/")
getwd()
adv <- read.csv(file = "../../data/83/4c42c3-8dd8-4fef-b402-7e341764d5e9/Advertising.csv", header = TRUE, sep = ",")
```

1. Inspect the structure of the Data Set

```
head(adv)
```

```
##      TV Radio Newspaper Sales
## 1 230.1  37.8      69.2  22.1
## 2  44.5  39.3      45.1  10.4
## 3  17.2  45.9      69.3   9.3
## 4 151.5  41.3      58.5  18.5
## 5 180.8  10.8      58.4  12.9
## 6   8.7  48.9      75.0   7.2
outputoutputoutput
 #valuevaluevalue+Bboth boutput
 str(adv)
 #+END_SRC

 #+BEGIN_EXAMPLE
  ## 'data.frame':    200 obs. of  4 variables:
  ##  $ TV       : num  230.1 44.5 17.2 151.5 180.8 ...
  ##  $ Radio    : num  37.8 39.3 45.9 41.3 10.8 48.9 32.8 19.6 2.1 2.6 ...
  ##  $ Newspaper: num  69.2 45.1 69.3 58.5 58.4 75 23.5 11.6 1 21.2 ...
  ##  $ Sales    : num  22.1 10.4 9.3 18.5 12.9 7.2 11.8 13.2 4.8 10.6 ...
```

```
summary(adv)
```

```
##        TV              Radio          Newspaper          Sales
##  Min.   :  0.70   Min.   : 0.000   Min.   :  0.30   Min.   : 1.60
##  1st Qu.: 74.38   1st Qu.: 9.975   1st Qu.: 12.75   1st Qu.:10.38
##  Median :149.75   Median :22.900   Median : 25.75   Median :12.90
##  Mean   :147.04   Mean   :23.264   Mean   : 30.55   Mean   :14.02
##  3rd Qu.:218.82   3rd Qu.:36.525   3rd Qu.: 45.10   3rd Qu.:17.40
##  Max.   :296.40   Max.   :49.600   Max.   :114.00   Max.   :27.00
```

(b) Construct Scatter Plots

So I'd like to do a shiny ggplot here, however it's probably just as easy to use tabset by appending `{.tabset}` to the heading

```
# par(mfrow=c(2,2))
# plot(lm(y~x))
```

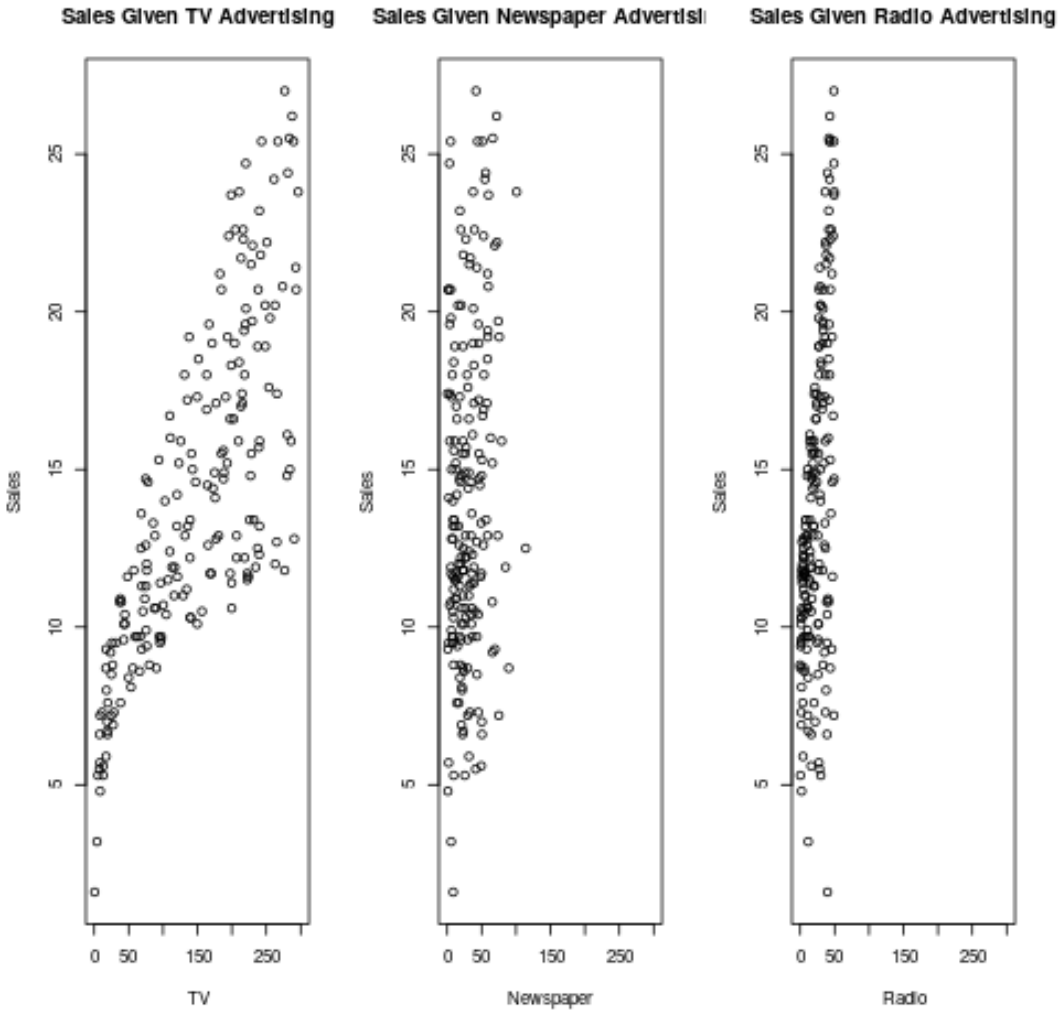
Multiple Plots may be fitted into one output using either the `par()` package or the `layout()` package, I personally prefer the `layout()` package, I think because in the past I had a bad experience with `par()`:

- **In order to use** `par`:
  - `par( mfcol = c(ROW, COLS))`
  - `par (mfcol = c(ROW, COLS))`
    - That's not a typo, `mfrow` and `mfcol` are identical in this case
- **In order to use** `layout`:
  - `layout(MATRIX)`
    - The matrix should be a grid, the plots will be fed to that grid in numerical order so for example:
      - `layout(matrix(1:3, nrow = 1))` will fit the plots to the following matrix in the order specified:

      $$ \begin{bmatrix} 1 & 2 & 3 \end{bmatrix} $$
- **In order to use 'grid.layout():
  - grid.arrange(plot1, plot2, ncol = 2))
    - This is the only one that will work with ggplot2

1. Multi Fit Base Plots R:BasePlot

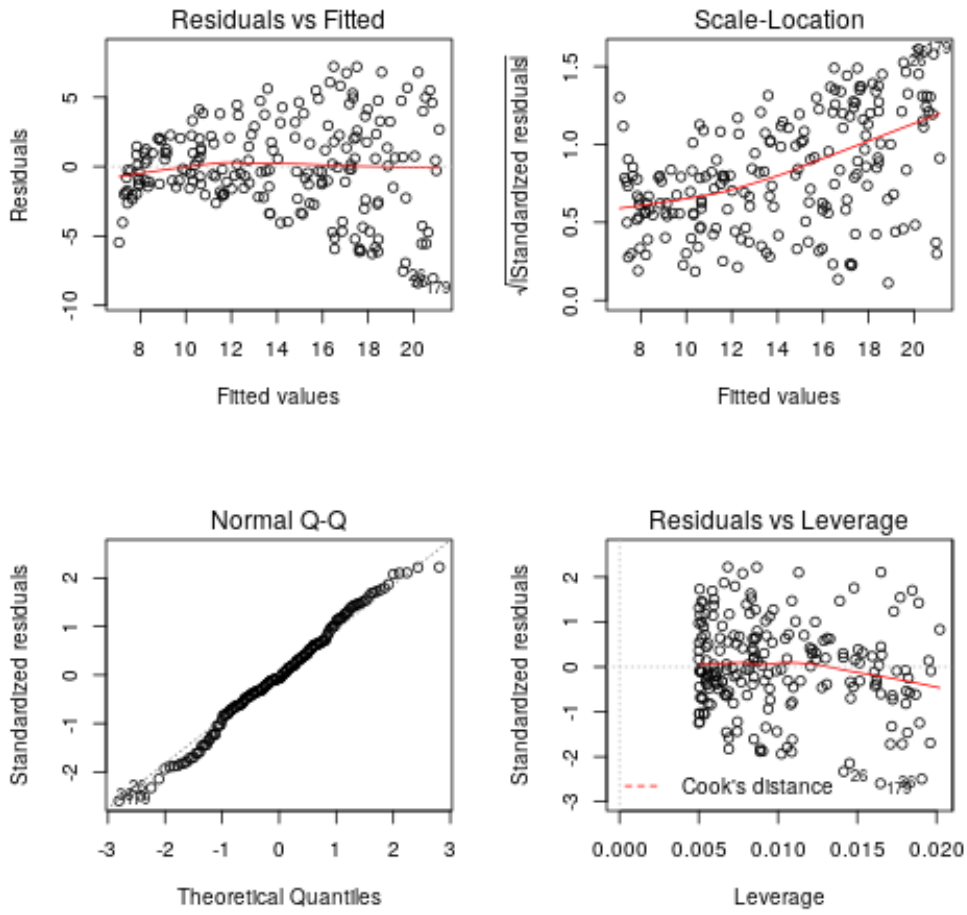
```r
# Set the layout:
  # Using `layout()` command:

    layout(matrix(1:3, nrow =1))

  # using `par()` command:

    #par(mfrow=c(1,3)) # Specify the

# Set the plot Domain
pdom <- c(0, 300) #Plot Domain

#Generate the plots
plot(formula = Sales ~ TV, data = adv, xlim = pdom,
     main = "Sales Given TV Advertising")
plot(formula = Sales ~ Newspaper, data = adv, xlim = pdom,
     main = "Sales Given Newspaper Advertising")
plot(formula = Sales ~ Radio, data = adv, xlim = pdom,
     main = "Sales Given Radio Advertising")
```



2. GGPlot GGPLOT2:LINEAR:REGRESSION

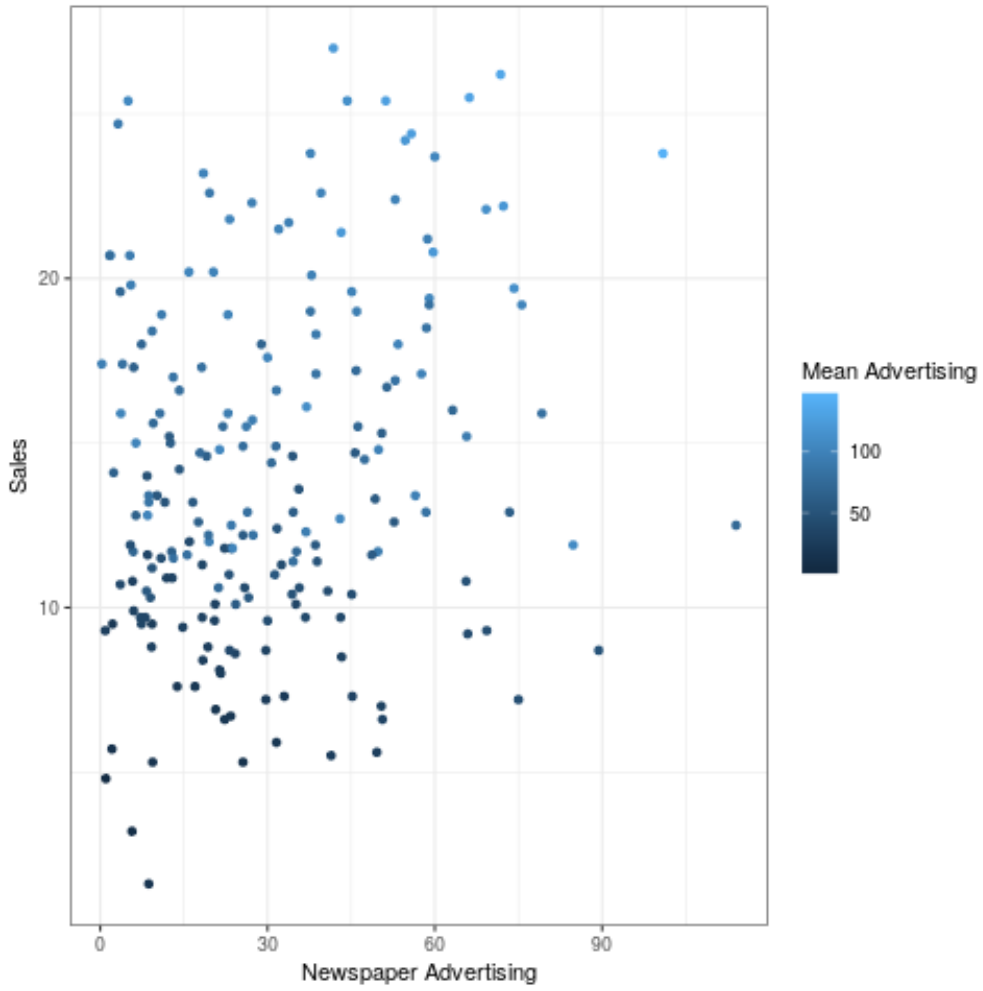
   (a) Television Advertising

```r
  adv$MeanAdvertising <- rowMeans(adv[,c(!(names(adv) ==
  ↪  "Sales"))])

  AdvTVPlot <- ggplot(data = adv, aes(x = TV, y = Sales, col =
  ↪  MeanAdvertising)) +
    geom_point() +
    theme_bw() +
    stat_smooth(method = 'lm', formula = y ~ poly(x, 2, raw =
    ↪  TRUE), se = FALSE) +
   ##stat_smooth(method = 'lm', formula = y ~ log(x), se =
    ↪  FALSE) +
    labs(col = "Mean Advertising", x= "TV Advertising")
print(AdvTVPlot)

  if(knitr::is_html_output()){
    ggplotly(knitr::is_latex_output())
  } else {
    AdvTVPlot
  }
```



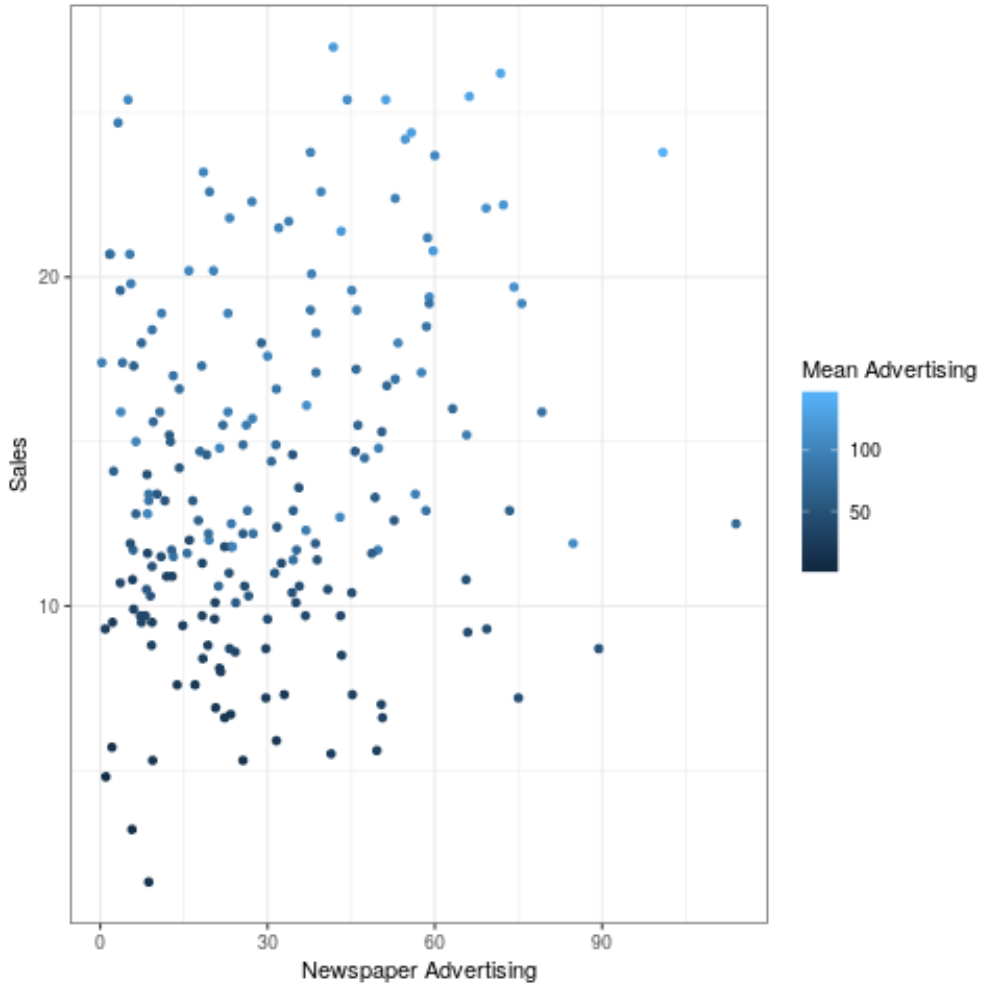
(b) Radio Advertising

```r
AdvRadPlot <- ggplot(data = adv, aes(x = Radio, y = Sales,
 ↪  col = MeanAdvertising)) +
  geom_point() +
  theme_bw() +
  labs(col = "Mean Advertising", x= "Radio Advertising") +
  geom_smooth(method = 'lm')

# padv %>% ggplotly() plotly doesn't work with knitr/LaTeX so
 ↪  test the output and choose accordingly:
#Thise could be combined into an interactive graph by
 ↪  wrapping in ggplotly(padv)

if(knitr::is_html_output()){
AdvRadPlot %>% ggplotly()
 } else {
   AdvRadPlot
 }
```

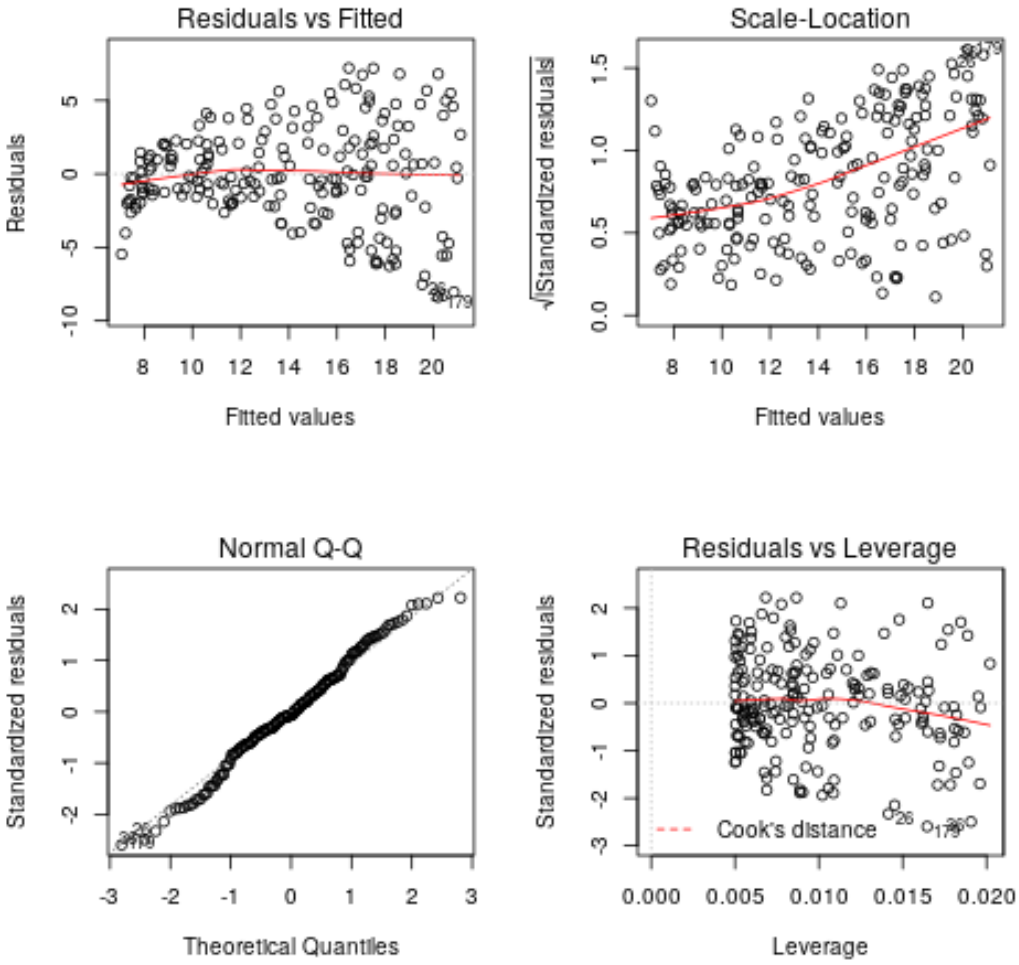


(c) Newspaper Advertising

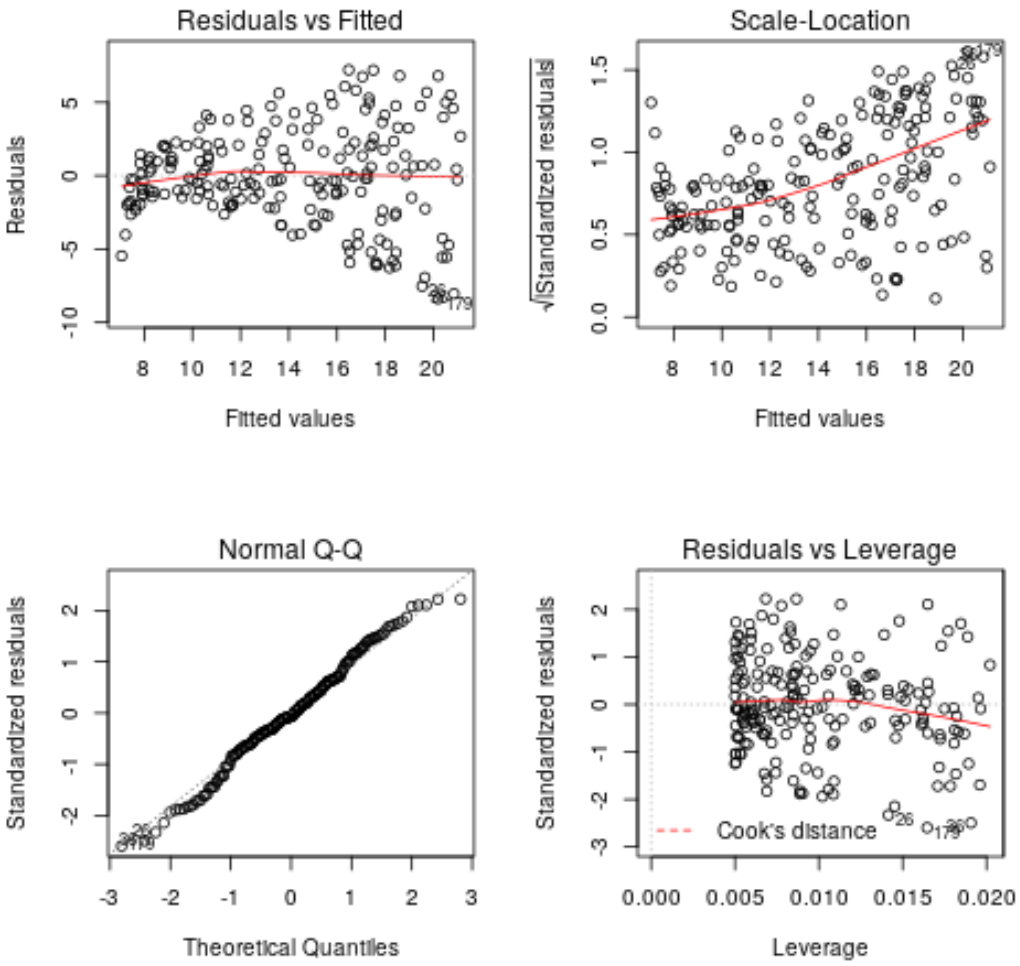
```
AdvNewsPlot <- ggplot(data = adv, aes(x = Newspaper, y =
↪  Sales, col = MeanAdvertising)) +
  geom_point() +
  theme_bw() +
  labs(col = "Mean Advertising", x= "Newspaper Advertising")

# padv %>% ggplotly() plotly doesn't work with knitr/LaTeX so
↪  test the output and choose accordingly:
#Thise could be combined into an interactive graph by wrapping
↪  in ggplotly(padv)

 if(knitr::is_html_output()){
AdvNewsPlot %>% ggplotly()
} else {
AdvNewsPlot
}
```



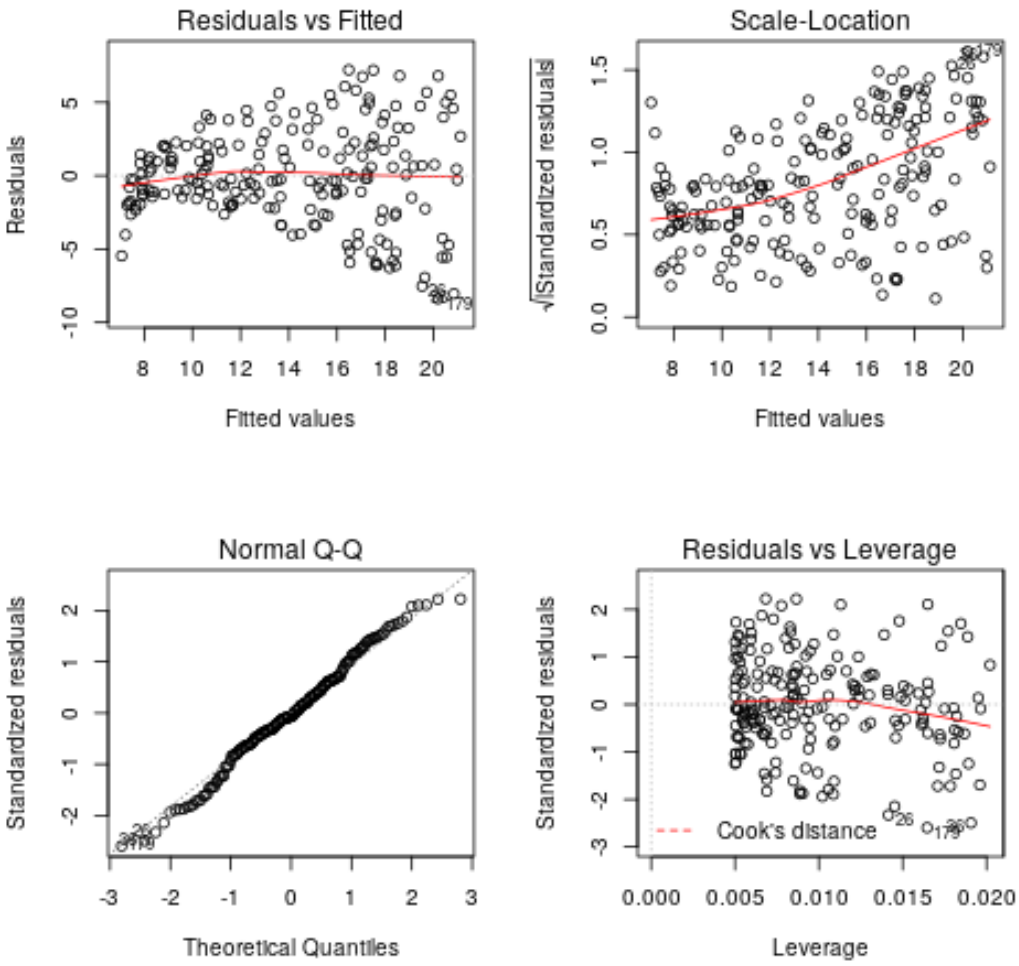
```r
pdom <- c(0, 300) #Plot Domain
plot(formula = Sales ~ TV, data = adv, xlim = pdom,
     main = "Sales Given TV Advertising")
```




```r
plot(formula = Sales ~ Newspaper, data = adv, xlim = pdom,
     main = "Sales Given Newspaper Advertising")
```




```
plot(formula = Sales ~ Radio, data = adv, xlim = pdom,
     main = "Sales Given Radio Advertising")
```



(c) Find the Correlation Coefficient **linear:regression**

The corellation coefficient can be found by using the `cor` function, it is a measurement of the strength of a linear relationship ranging from -1, to 1, wherein a value of 0 would represent no relationship.

The Pearson Correlation Coeffient tends to be used over other models and it's value is determined by:

$$r_{xy} = \frac{\sum_{i=1}^{n} [x_i - \overline{x}] \times (y_i - \overline{y})}{\sqrt{\sum_{i=1}^{n} \left[(x_i - \overline{x})^2\right]} \sqrt{\sum_{i=1}^{n} \left[(y_i - \overline{y})^2\right]}}$$

Some of the assumptions underlying the Correlation Coefficient are: [1]

- Independent Observations

[1] Corellation Coefficient

- Normally distributed observations (i.e.ăfollows a bell curve)
- hmoscedasticity [2]
  - This means equal variance of observations
    - i.e.ăall there is no pattern between the variables and the plot, the points should make a rectangle, not a triangle
- Normally distributed points
- the points must make a straight line not a curve

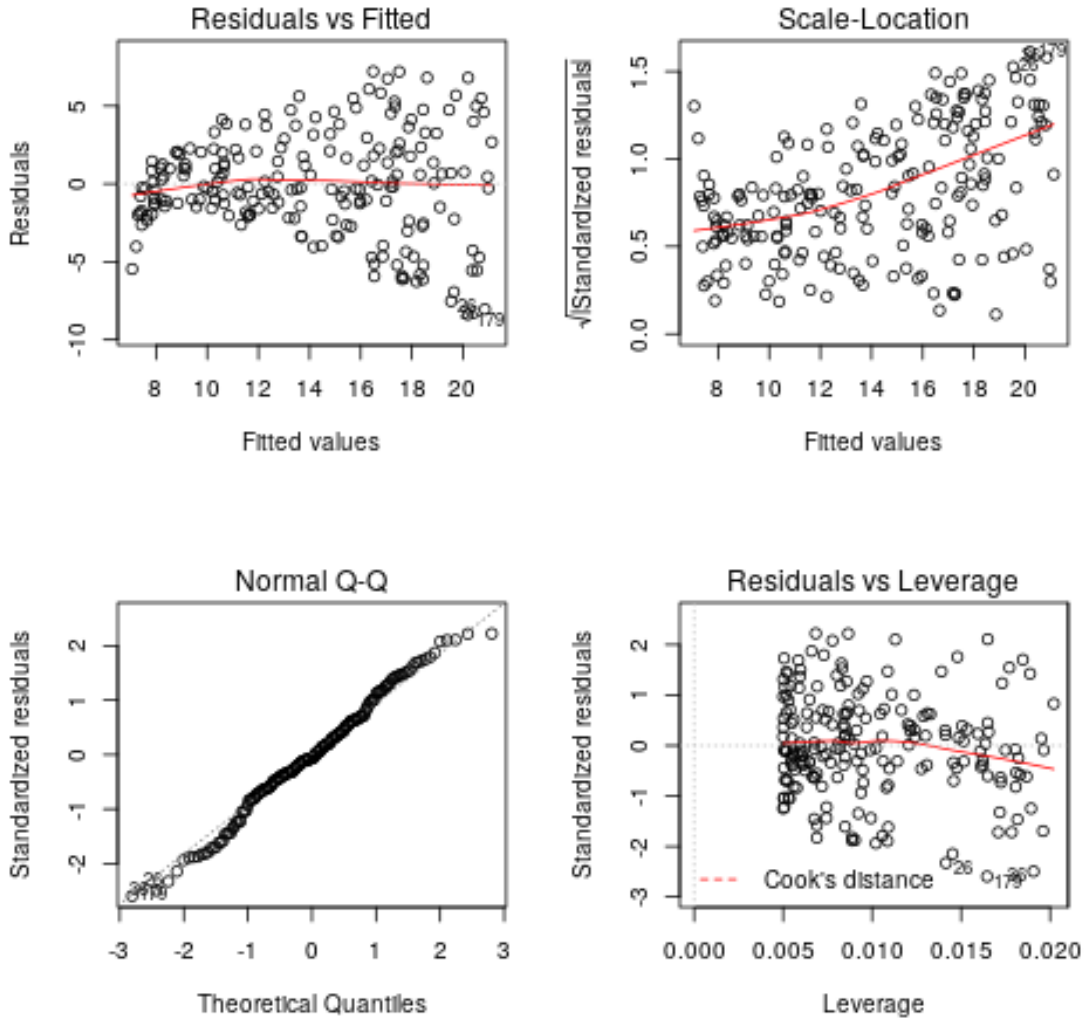
the correlation coefficient in this case can be found by using `cor(x = adv$TV, y = adv$Sales)` and provides that $r \approx 0.78$ This might not be a meaningufl value however because the variance of the sales appears to increase as advertising increases, if that is overlooked however the pearson correlation coefficient provides that the model is a reasonably strong positive linear model.

(d) Assess the accuracy of the parameter estimates **ModelEvaluation**

The parameter estimates may be returned by summarising the model with `summary(lm)`

```
lmMod <- lm(formula = Sales ~ TV, data = adv)
lmSum <- summary(lmMod)
lmSum
```

[2] PennState University




```
##
## Call:
## lm(formula = Sales ~ TV, data = adv)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.3860 -1.9545 -0.1913  2.0671  7.2124
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 7.032594   0.457843   15.36   <2e-16 ***
## TV          0.047537   0.002691   17.67   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.259 on 198 degrees of freedom
```

```
## Multiple R-squared:  0.6119, Adjusted R-squared:  0.6099
## F-statistic: 312.1 on 1 and 198 DF,  p-value: < 2.2e-16
```

```
lmSum$coefficients
```

```
             Estimate  Std. Error  t value    Pr(>|t|)
(Intercept) 7.03259355 0.457842940 15.36028 1.40630e-35
TV          0.04753664 0.002690607 17.66763 1.46739e-42
```

```
##               Estimate  Std. Error  t value    Pr(>|t|)
## (Intercept) 7.03259355 0.457842940 15.36028 1.40630e-35
## TV          0.04753664 0.002690607 17.66763 1.46739e-42
```

```
lmMod2 <- lm(formula = Sales ~ TV, data = adv)
```

In this case we have:

- a slope of $\beta_1 \approx 0.048 \pm 0.0027$
- an Intercept of $\beta_0 \approx 7 \pm 0.46$

The standard deviation of a statistic used as an estimator of a population parameter is often referred to as the **standard error of the estimator (S.E.)**; it is the $\pm$ values specified above:

- Standard Error of Slope Coefficient $\sigma_{\beta_1} = s\sqrt{\frac{1}{n} + \frac{\overline{x}^2}{SS_x}} = 0.00027$
- Standard Errof of Intercept Coefficint $\sigma_{\beta_0} = \frac{s}{\sqrt{SS_x}} = 0.46$

Where:

- s is the sample standard deviation (OF WHAT?)
- $SS_x = \sum_{i=1}^{n} \left[x_i^2\right] - n \cdot (\overline{x})^2$
- s is the sample standard deviation of x
- because the sample standard deiation of x predicts the deviation of y anyway

You may also have the standard deviation of the residuals (the distance along the y-axis of a point from the regression line), this is known as the **Residual Standard Error** and is calculated via the *Ordinary Least Squares Method* [3], it is is given by:

[3] i.e.ăchosing β_0 and β_1 to minimise $\left(\textbf{RSS} = \sum_{i=1}^{n} \left[(y_i - \hat{y}_i)^2\right]\right)$

$$\sigma_\varepsilon = S.E. = \sqrt{\frac{\mathbf{RSS}}{N}} \tag{1}$$

$$\tag{2}$$

$$= \sqrt{\frac{\sum_{i=1}^{n}\left[(y_i - \hat{y}_i)^2\right]}{N}} \tag{3}$$

Which you'll notice is identical to the ***RMSE***.

so by the emperical method $2\times$S.E. would represent a 95% confidence interval (rather than prediction interval) of the expected y-values. Drawing such a confidence interval:

```
paramint <- confint(object = lm(adv$Sales ~ adv$TV), level = 0.95) %>%
  ↪  signif(2)
paramint
```

```
##             2.5 % 97.5 %
## (Intercept) 6.100  7.900
## adv$TV      0.042  0.053
```

So drawing from this we could expect, with only a 5% probability of incorrectly rejecting the null hypothesis that there is no relationship, that in the absence of advertising, the TV sales to fall between 6.1 and 7.9.

With the same degree and type of certainty it could also be oncluded that for every $1000 increase in advertising, the tv sales will increase by between 42 and 53.

1. (f) Test the significance of the slope of the linear model If it is appropriate to fit a linear model to data, then we can test for correlation between the data points by considering whether or not the slope value is non-zero $\beta_1 \neq 0$, this is because a zero coefficient would be such that the model would predict $Y = C + \varepsilon$, this means that X is not a feature/predictor of Y, however Y may still be a function of (or rather response variable of) other values other factors that are 'behind the scenes'.[4]

   So our hypotheses would be:

$$H_0: \ \beta_1 = 0 \qquad (\text{ The null hypothesis is that nothings related}) \tag{4}$$

$$H_1: \ \beta_1 = 1 \tag{5}$$

   So our interest is to determine how far from 0 our expected β_1 value needs to be from 0 for us to conclude

   > The expected value of β_1 is so far from zero we can conclude that it it's not zero at some significance level †"

[4] Refer to page 67 of the text book, section [3.1.2]

#+BEGIN$_{\text{QUOTE}}$ † *at some low probability of incorrectly rejecting the null hypothesis*

#+END$_{\text{QUOTE}}$

The problem is defining how far from zero is far enough, for this we use the expected distance from the regression line, the standard error from above, a value observed observed too many standard deviations to the right of the mean are not very likely too occur.

(a) Choosing a Parametric method A statistical method that relies on an underlying assumption of the statistical distribution of the data is known as a a parametric method, in this case, it is a fundamental assumption of **Ordinary Least Squares** Linear regression that the data is normally distributed.[5]

This is a situation where we use the *Student's t-test* because this is a sample, and the population standard deviation (σ) is not known and hence the confidence interval for the mean must be made broader in order to account for the fact that the sample standard deviation s is being used to estimate σ

because the sampled population is normally distributed, the sampling distribution of $\bar{x}$ will be normally distributed [6] (regardless of sample size) and centred about μ with a a standard deviation of $\frac{\sigma}{\sqrt{n}}$. If the population was non-normal the sampling distribution will be approximately normal for $n \geq 30$.

Because $\frac{\sigma}{n}$ is the standard deviation of the the sample mean $\bar{x}$ it is reffered to as the **Standard Error of the mean** [7], so we could calculate the critical value along the standard normal distribution corresponding to the the sampling distribution in order to determine probabilities, however, σ is unknown and using s instead will not create a normal distribution, the distriution it creates is Gosset's **Student's t-distribution** [8]:

$$t = \frac{\overline{x} - \mu}{\frac{s}{\sqrt{n}}} \tag{6}$$

So in this case our test statistic will be:

$$t = \frac{\hat{\beta}_1 - 0}{\mathsf{SE}\left(\hat{\beta}_1\right)} \tag{7}$$

In order to perform this test in R we can use `qnorm` and `qt` to return critical values, `t.test` will perform a hypothesis test directly from input data but that's not suitable here.

```
tcritval <- qt(p = 0.05,df =nrow(adv)-2 )
tcritval %>% signif(2)
```

[5] Mendenhall, *Introduction to Probability & Statistics* p.ă254 [7.4]
[6] By the Central Limit Theorem
[7] Mendenhall, *Introduction to Probability & Statistics* p.ă254 [7.4]
[8] Mendenhall, *Introduction to Probability & Statistics* p.ă254 [7.4]

```
## [1] -1.7
```

So the critical t-value is -1.7 and from the summary call from before we have that the t-statistic is 17, which far exceeds this, as a matter of fact further over to the right the p-value is reported at $\alpha = 10^{-16}$.

In practice you'd just read off the *p*-values and pick the ones with $*$ to the right of them, the more $*$ the more significance.

Hence we reject the hypothesis that no relationship exists at an extremely low probability of incorrectly doing so (i.e.ălow probability of commiting type 1 error).

2. (g) Plot the straight line within the scatter plot and comment

   (a) Base Plot In order to plot this inside base packages, feed the model object, i.e.ă=lm(Y~X)= inside a call to `abline()` in order to plot the model over the top of the base plot, so all together it might look like: [9]

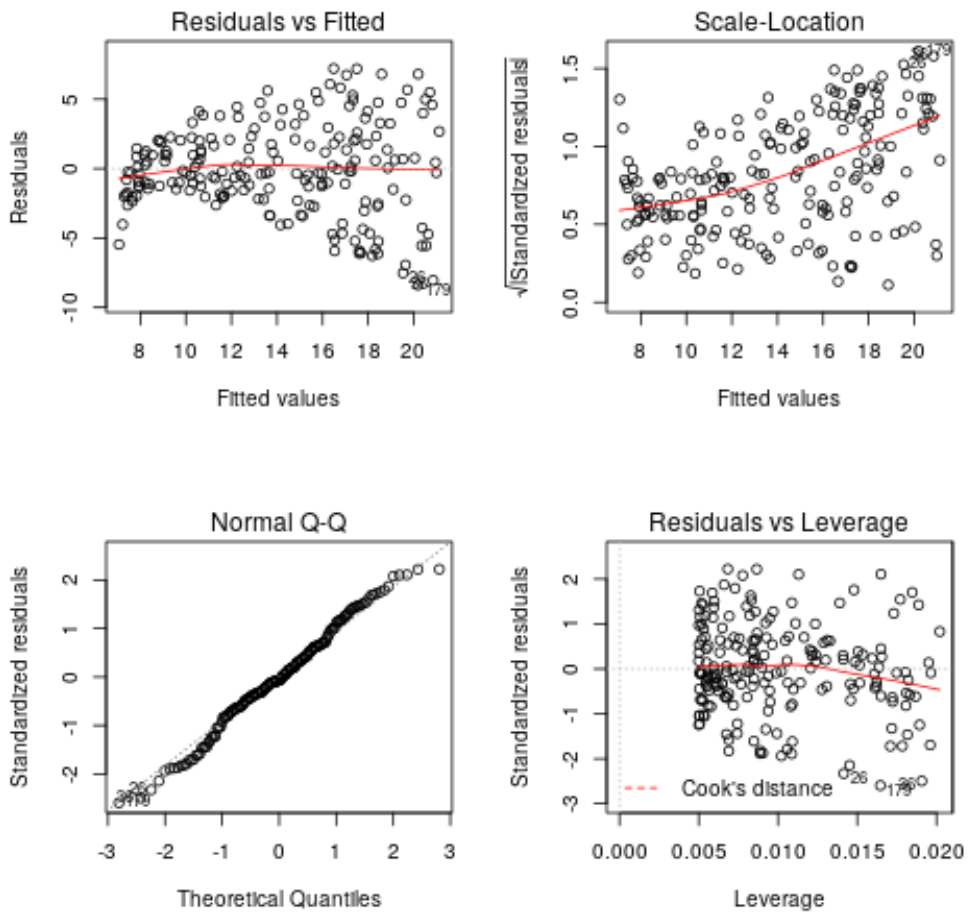
```
Form <- Sales ~ TV
Lmodel <- lm(formula = Form, data = adv, na.action = na.exclude)
plot(Form, data = adv)
abline(Lmodel)
```

Or you could do it like this even, but I think the way above is better syntax because it will behave better with 'predict' function and follows `tidyverse` syntax

```
Lmodel <- lm(adv$Sales ~ adv$TV)
plot(x = adv$TV, y = adv$Sales)
abline(a = Lmodel$coefficients[1], b = Lmodel$coefficients[2])
```

```
1  plot(formula = Sales ~ TV, data = adv, xlim = pdom,
2       main = "Sales Given TV Advertising")
3
4  abline(lmMod)
```

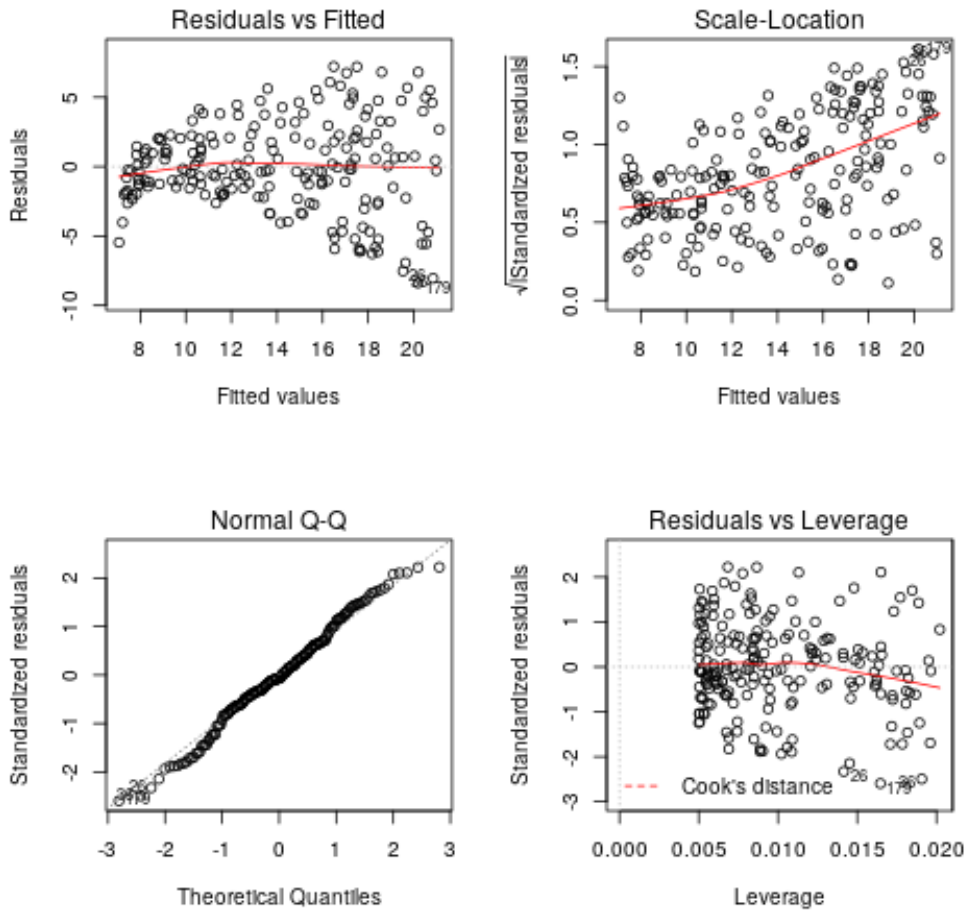
[9] `na.exclude` will pad values extracted so lengths are the same, `na.omit` will not



(b) GGplot

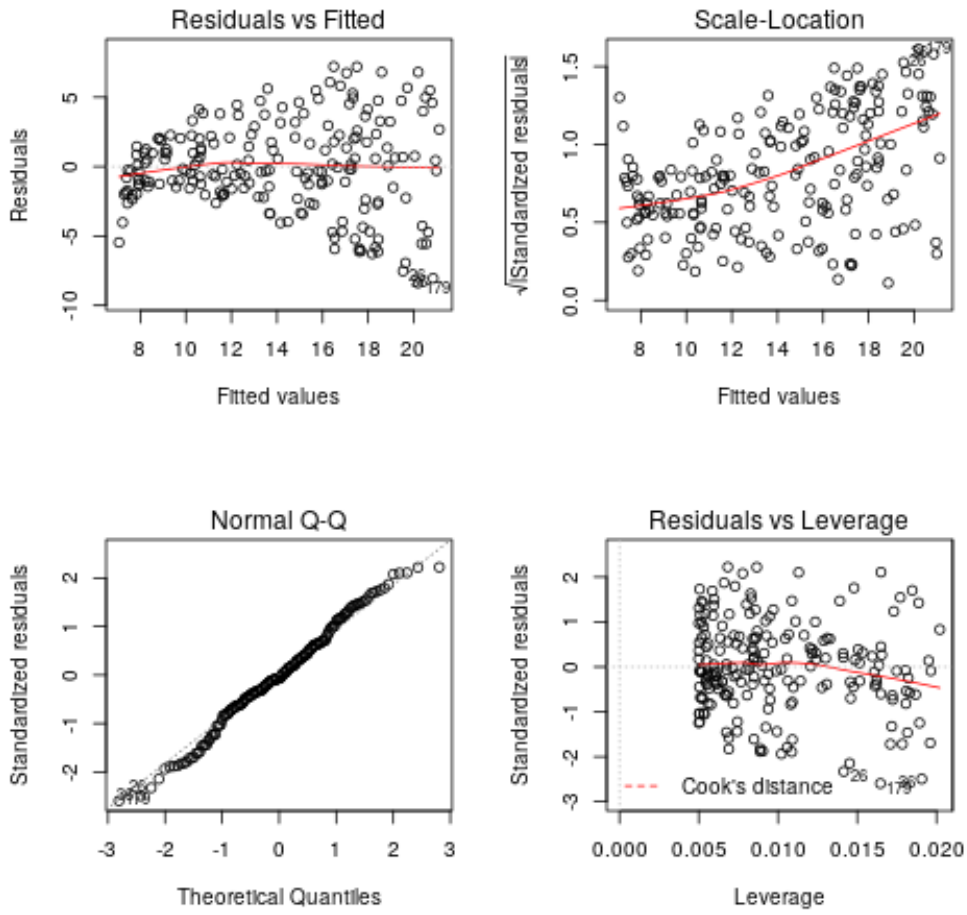
LINEAR:REGRESSION:GGPLOT2

```
AdvTVPlot <- ggplot(data = adv, aes(x = TV, y = Sales, col =
↪  MeanAdvertising)) +
  geom_point() +
  theme_bw() +
  stat_smooth(method = 'lm', formula = y ~ x, se = FALSE)

AdvTVPlot
```



If we needed to feed ggplot a specific model we could do that like this, but it's a whole thing to do and you'd probably rather not do it this way, but if you really really need to

```
AdvTVPlot <- ggplot(data = adv, aes(x = TV, y = Sales, col =
↪  MeanAdvertising)) +
  geom_point() +
  theme_bw() +
  stat_smooth(
      method = "lm",
      mapping = aes( y = predict(lmMod)
                     )
      )

AdvTVPlot
```



3. (h) Assess the overall accuracy of the model The model can be assed by considering the:

   - Coefficient of determination R^2 which is the proportion of variance in the data that is explained by the model
   - Only in the case of simple linear regression is $R^2 = (r)^2$
   - The Residual Standard Error is the standard deviation of the residuals, i.e.ăit is the expected distance between each point to the regression line, taken along the y-axis.

   (a) Terminology The texbook makes, in my opinion, a mistake in that it refers to the the *Root Mean Square Error* (***RMSE***) as the *Residual Standard Error* (***RSE***) [10], this is true, the standard error of the residuals (ε) would be the RMSE, so we would have $\boldsymbol{RMSE} = \sigma_\varepsilon$, that's fine.

[10] Refer to Page 69 of the TB for RMSE definition, the TB divides by DF which is probably more correct that dividing by sample size.

The issue is there is another common term used called the *Relative Squared Error* (***RSE***) is often used [11] and so this is hence ambiguous, hence forth I will:

- Refer to the Standard Error of the residuals ($\sigma_{varepsilon}$) as ***RMSE***:
  - RMSE $= \sqrt{\frac{\sum \varepsilon^2}{n}}$
- Refer to the Relative Standard Error as ***RSE***
  - RSE $= \frac{\sigma_{\hat{\varepsilon}}^2}{\sigma_{\hat{y}}^2} = \frac{\sum (y-\hat{y})^2}{\sum (y-\bar{y})^2}$
    - The advantage to the RSE is that it can be compared between models with different units, whereas the RMSE cannot, just another tool in the belt I suppose.

(b) Root Mean Square Error LossFunction Recall that the model was of the form $Y = \beta_1 X + \beta_0 + \varepsilon$, the ***RMSE*** (*Root Mean Square Error*) is the standard deviation of ε as measured along the Y-axis:

$$\sigma_{\varepsilon} = \sqrt{\frac{\sum_{i=1}^{n} \left[(y_i - \hat{y}_i)^2 \right]}{N}} \tag{8}$$

This value can be returned from R by investigating the anova table:

```
anova(lmMod)
```

```
## Analysis of Variance Table
##
## Response: Sales
##            Df Sum Sq Mean Sq F value    Pr(>F)
## TV          1 3314.6  3314.6  312.14 < 2.2e-16 ***
## Residuals 198 2102.5    10.6
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

From the *ANOVA* table it can be seen that the average squared residual is 10.6

[11] An Introduction to Data Science : Model Evaluation - Regression

$$\text{mean}\left(\varepsilon^2\right) = 10.6 \tag{9}$$

$$\implies \frac{1}{n} \cdot \sum_{i=1}^{n} \left[\varepsilon_i\right] = 10.6 \tag{10}$$

$$\implies \frac{1}{n} \cdot \sum_{i=1}^{n} \left[(\hat{y}_i - y_i)^2\right] = 10.6 \tag{11}$$

$$\implies \sqrt{\frac{1}{n} \cdot \sum_{i=1}^{n} \left[(\hat{y}_i - y_i)^2\right]} = 3.2 \tag{12}$$

$$\tag{13}$$

$$\implies \sigma_\varepsilon = 3.2 \tag{14}$$

Thus we may conclude that we expect the model to predict the sales within $\pm$ 3.2 units, which is quite predictive and hence useful.

(c) Coefficient of Determination The coefficient of determination is the proportion of variation within the model that is explained by the model:

$$R^2 = \frac{TSS - RSS}{TSS} \tag{15}$$

$$= \frac{3315}{3315 + 2103} \tag{16}$$

$$\tag{17}$$

$$= 0.612 \tag{18}$$

In practice we would simply extract the coefficient of determination (R^2) from the model-summary:

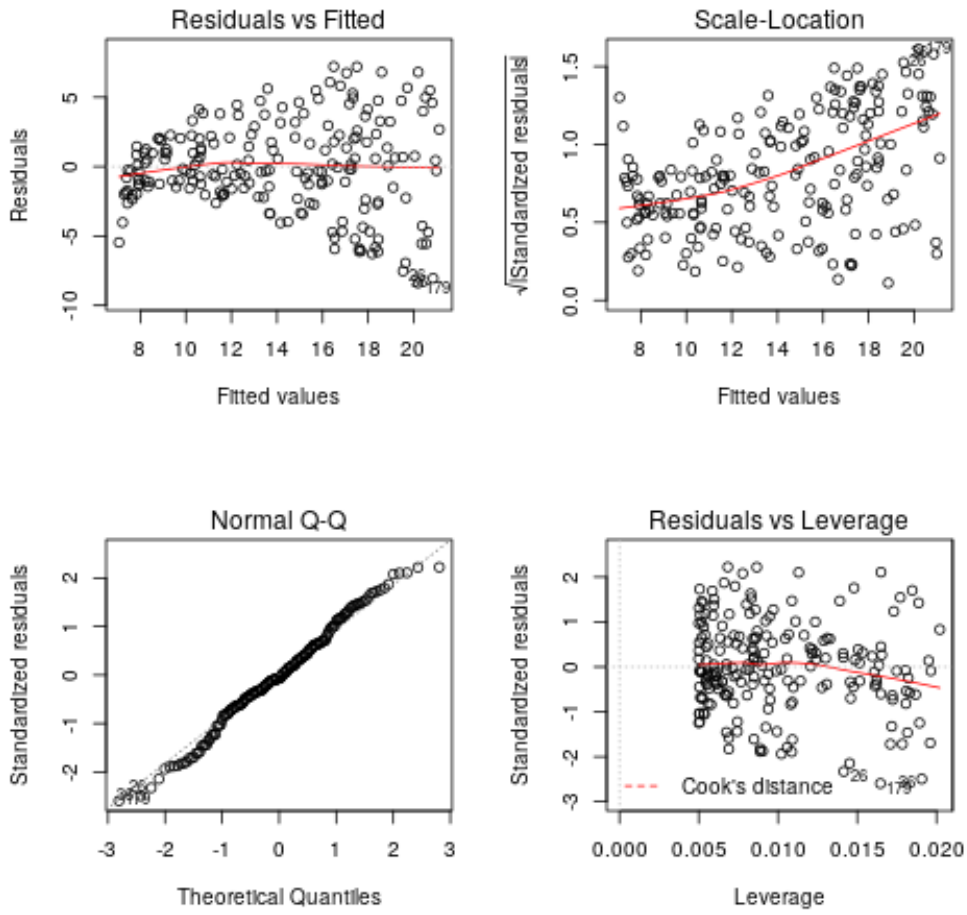
```
lmSum$r.squared %>% round(3) %>% percent()
```

```
## [1] "61.2%"
```

This value suggests that a reasonable amount of the variation is explained by the model, but perhaps a non-linear model could explain more of the variance. (be careful a significant coefficient of determination doesn't necessarily mean that the slope is significantly different from 0)

(d) Residual Analysis

```
layout(matrix(1:4, nrow = 2))
plot(lmMod)
```



- The residual plot does not appear to normally distributed, there is a slight logarithmic trend, this violates assumptions of the linear model undermining the predictive capacity of the model in this case.
- The variance is also non-constant, for a linear model to be used in must be homoscedastic (i.e.ăconstant variance), this is not the case implying that the assumptions of the linear model have been violated and hence this model may not be appropriate [12]
- the standardised residuals should be normally distriuted with a mean of 0 and standard deviation of 1, whilst the standard deviation appears acceptable, the standardised residuals are centred around $\approx 3/4$ with a positive upward slope violating the assumption of normality.
- The normal Q-Q plot is a straight line so actually the data is probably normally distributed, the only issue is the heteroscedasticity of the data.

[12] refer to page 96 of the TB, log or exp transforming may be appropriate here, the data is not homosdcedastic and is hence said to be heteroscedastic. #### How to use predict

- The Cook's Distance plot suggests that there are some points with a high amount of leverage, so perhaps there are some outliers or perhaps the increasing variance is undermining the appropriateness of the model.

4. (i) Use the model to make predictions When making predictions is important to ensure that the names of a data frame are `syntactically correct`, otherwise you will have a bad day trying to get predict to work and ggplot2 to work because specifying the data frame names in a formula will be difficult, make sure that names are always syntactically valid.

   what is important is you create your model with the correct syntax, if you create your model like this:

   ```
   mymodelWRONG <- lm(adv$Sales ~ adv$TV)
   ```

   you won't be able to predict data like this:

   ```
   predict(object = lmMod, newdata = data.frame("TV" = 300))
   ```

   you'll just get an error that says `'newdata' had 1 row but variables found have 200 rows`, you have to give the variables corresponding names so that the model object can save them for later and make the connection, for instance, if inspect the terms from above you will get:

   ```
   mymodelWRONG[["terms"]]
   ```

   which outputs, at the tail end:

   ```
   adv$Sales    adv$TV
   "numeric" "numeric"
   ```

   where as if you create the model like this:

   ```
   lmModCORRECT <- lm(formula = Sales ~ TV, data = adv)
   predict(object = lmModCORRECT, newdata = data.frame("TV" = 300))
   ```

   and inspect the terms with:

   ```
   lmModCORRECT[["terms"]]
   ```

   you will get this as output

   ```
     Sales        TV
   "numeric" "numeric"
   ```

where `Sales` and `TV` are the outputs of `names(adv)` and so I can use that when I use predict. You should not use `attach` it will cause problems later, however, it can be nice to use attach just before a predict call to get auto completed names and then remove attach and re-execute the script .

So always use the `lm(formula = Y~X, data = myDF)` because it works the best; you have to use the same syntax/format when using predict or ggplot anyway so there's no reason not to use the same syntax throughout anyway.

Also the lecturer said to use lists, I reckon use data frames because that way your `newdata` matches the input data one-to-one, moreover:

- It makes it far simpler to assign names, because again, the input/ouptu data will all be the same format
- when creating *Lasso* Regression Models you have to use matrices as input data and it's easier to set your workflow up to go from dataframe to matrix (You have to do this in predictive modelling)

(a) Predict the Data

i. One Point

```
input = 3
output <- predict(object = lmMod, newdata = data.frame("TV"
↪  = 3))
predDatasingle <- data.frame(input, output)
names(predDatasingle) <- names(adv[c(1,4)])

print(predDatasingle)
```

```
##   TV    Sales
## 1  3 7.175203
```

ii. Multiple points

```
input <- seq(from = 100, to = 900, by = 100)
output <- predict(object = lmMod, newdata = data.frame("TV"
↪  = input))

predDF <- data.frame(input, output)
names(predDF) <- names(adv[c(1,4)])
predDF
```

```
##    TV    Sales
## 1 100 11.78626
## 2 200 16.53992
## 3 300 21.29359
## 4 400 26.04725
## 5 500 30.80091
```

```
## 6 600 35.55458
## 7 700 40.30824
## 8 800 45.06191
## 9 900 49.81557
```

Question 02

(a) Upload the Auto Dataset and explore it.

(b) Construct scatter plots to visualize the relationship between

mpg and displacement, weight and accellertion: :CUSTOM$_{\text{ID}}$: b-construct-scatter-plots-to-visualize-the-relationship-between-mpg-and-displacement-weight-and-accellertion

Repeat the analysis in Q1 (c) to (i) using mpg and weight.

Multiple Linear Regression

Material of Tue 19 March2019, week 3

Question 01 - Multiple Linear Regression

Load Packages

```
# Load Packages
if(require('pacman')){
  library('pacman')
}else{
  install.packages('pacman')
  library('pacman')
}

pacman::p_load(caret, scales, ggplot2, rmarkdown, shiny, ISLR, class, BiocManager,
               corrplot, plotly, tidyverse, latex2exp, stringr, reshape2, cowplot, ggpubr,
               rstudioapi, wesanderson, RColorBrewer, colorspace, gridExtra, grid, car,
               boot, colourpicker, tree, ggtree, mise, rpart, rpart.plot, knitr, MASS,
               magrittr, EnvStats,tidyverse,tidyr,devtools, bookdown, leaps, car, clipr,
               tikzDevice, e1071)



mise()
set.seed(0932)
```

(a) Upload the data "Advertising.csv" and explore it. **ATTACH**

First import the data and investigate it:

```
adv <- read.csv(file = "../.././data/83/4c42c3-8dd8-4fef-b402-7e341764d5e9/Advertising.csv", header = TRUE, sep =
  ",")
```

```
head(adv)
```

```
##      TV Radio Newspaper Sales
## 1 230.1  37.8      69.2  22.1
## 2  44.5  39.3      45.1  10.4
```

```
## 3  17.2  45.9      69.3   9.3
## 4 151.5  41.3      58.5  18.5
## 5 180.8  10.8      58.4  12.9
## 6   8.7  48.9      75.0   7.2
```

```
writeLines("\n")
```

```
print("***Dimensions***")
```

```
## [1] "***Dimensions***"
```

```
writeLines("\n")
```

```
dim(adv)
```

```
## [1] 200   4
```

```
writeLines("\n")
```

```
print("***Summary***")
```

```
## [1] "***Summary***"
```

```
writeLines("\n")
```

```
summary(adv)
```

```
##        TV              Radio          Newspaper          Sales
##  Min.   :  0.70   Min.   : 0.000   Min.   :  0.30   Min.   : 1.60
##  1st Qu.: 74.38   1st Qu.: 9.975   1st Qu.: 12.75   1st Qu.:10.38
##  Median :149.75   Median :22.900   Median : 25.75   Median :12.90
##  Mean   :147.04   Mean   :23.264   Mean   : 30.55   Mean   :14.02
##  3rd Qu.:218.82   3rd Qu.:36.525   3rd Qu.: 45.10   3rd Qu.:17.40
##  Max.   :296.40   Max.   :49.600   Max.   :114.00   Max.   :27.00
```

```
writeLines("\n")
```

```
print("***Structure***")
```

```
## [1] "***Structure***"
```

```
writeLines("\n")
```

```
str(adv)
```

```
## 'data.frame':    200 obs. of  4 variables:
##  $ TV       : num  230.1 44.5 17.2 151.5 180.8 ...
##  $ Radio    : num  37.8 39.3 45.9 41.3 10.8 48.9 32.8 19.6 2.1 2.6 ...
##  $ Newspaper: num  69.2 45.1 69.3 58.5 58.4 75 23.5 11.6 1 21.2 ...
##  $ Sales    : num  22.1 10.4 9.3 18.5 12.9 7.2 11.8 13.2 4.8 10.6 ...
```

```
writeLines("\n")
```

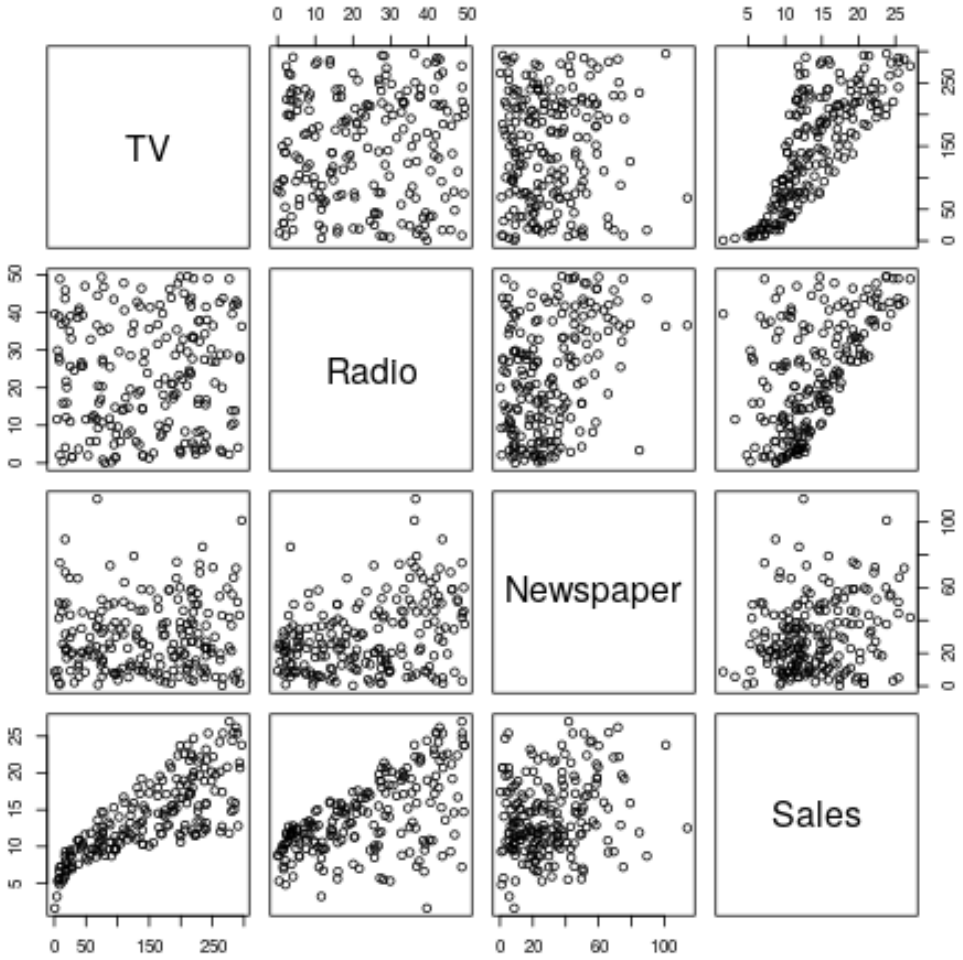
Fom this we can tell that there is one output, with 3 input values and 200 Observations.

(b) Find the Covariance and Correlation Matrix of Sales, TV, Radio

and Newspaper. :CUSTOM$_{ID}$: b-find-the-covariance-and-correlation-matrix-of-sales-tv-radio-and-newspaper. :CLASS: tabset

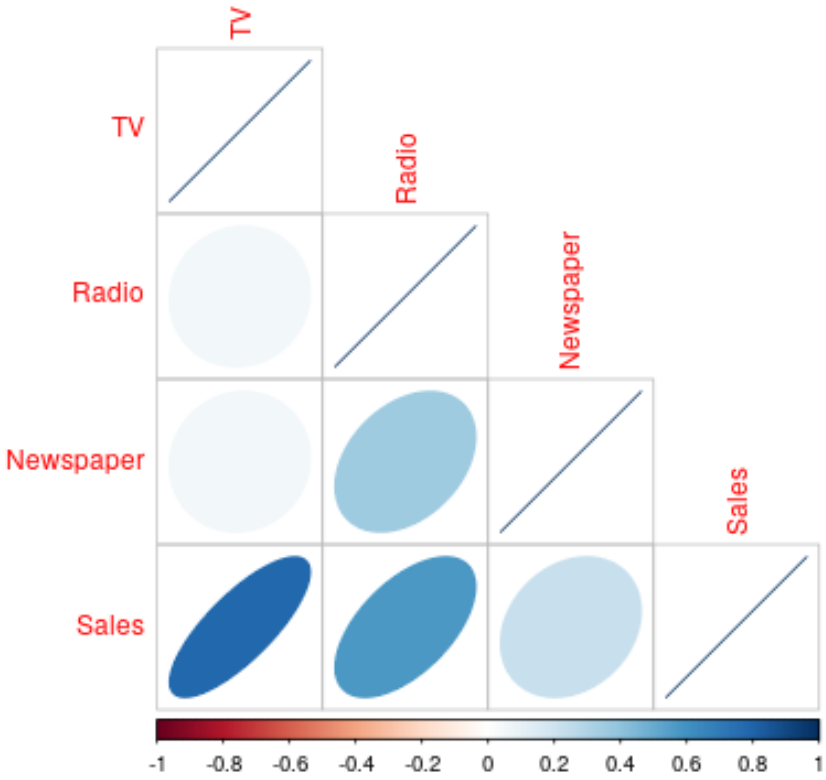
1. Base Packages

```
pairs(x = adv)
```



2. corrplot In order to use `corrplot` first create a correlation matrix using `cor(adv)` then feed that matrix to `corrplot` with the command `corrplot(cor(adv))`

```
# coriris <- cor(iris[,!(names(iris) == "Species")])
# corrplot(method = 'ellipse', type = 'lower', corr = coriris)

corMat <- cor(x = adv)
corrplot(corr = corMat, method = "ellipse", type = "lower")
```



From this we can see that there is a significant amount of correlation between Radio and newspaper (moreso even that newspaper and sales), we should consider this variable interaction when decide upon our model

(c) Construct the multiple linear regression model and find the

least square estimates of the model :CUSTOM$_{ID}$: c-construct-the-multiple-linear-regression-model-and-find-the-least-square-estimates-of-the-model

A multiple linear regression would give the model:

```
advModMult <- lm(formula = Sales ~ TV + Radio + Newspaper, data = adv)
advModMult
```

```
##
## Call:
## lm(formula = Sales ~ TV + Radio + Newspaper, data = adv)
##
## Coefficients:
## (Intercept)           TV        Radio    Newspaper
##    2.938889     0.045765     0.188530    -0.001037
```

This gives that the appropriate linear model is:

$$Y_{Sales} = 0.0458 \times \text{TV} + 0.19 \times \text{Radio} - 0.001 \times \text{Newspaper}$$

the fact that Newspaper has a negative coefficient despite being positively correlated with sales is indicative of the weak effect newspaper advertising has on sales as well as the interaction between newspaper and sales.

from this it can be ### (d) Test the significance of the parameters and find the resulting model to model Sales in terms of advertising modes, TV, Radio and Newspaper. First Summarise the Model:

```
1  advModMult %>% summary()
```

```
##
## Call:
## lm(formula = Sales ~ TV + Radio + Newspaper, data = adv)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.8277 -0.8908  0.2418  1.1893  2.8292
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.938889   0.311908   9.422   <2e-16 ***
## TV           0.045765   0.001395  32.809   <2e-16 ***
## Radio        0.188530   0.008611  21.893   <2e-16 ***
## Newspaper   -0.001037   0.005871  -0.177     0.86
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.686 on 196 degrees of freedom
## Multiple R-squared:  0.8972, Adjusted R-squared:  0.8956
## F-statistic: 570.3 on 3 and 196 DF,  p-value: < 2.2e-16
```

A summary of the model provides that given the hypothesis test:

$$H_0: \ \beta_i = 0 H_a: \ \beta_i \neq 0 \qquad \forall i \in \mathbb{N}$$

There would be an extremely low probability of incorrectly rejecting the null hypothesis that the given a linear model the coefficients would be zero, hence it is accepted that the coefficients are non-zero except for newspaper advertising.

There is a high probability of incorrectly rejecting the null-hypothesis and hence that should not be rejected and the nespaper advertising should not be seen as significant predictor for sales in this linear model.

It is hence approprate to remove, via backwards selection, Newspaper from the model, which gives:

```
## Calculate Power??

# n <-  length(adv)
# sigma <-  sd(adv$Newspaper)
# sem = sigma/sqrt(n)
# alpha <-  0.05
# mu0 <- 0
# q <- qnorm(p = 0.005, mean = mu0, sd = sem);q
# mu <- q # assumed actual mean value
# pnorm(q, mean = mu, sd = sem, lower.tail = FALSE)
```

```
advModMultb1 <- lm(formula = Sales ~ TV + Radio, data = adv)
advModMultb1
```

```
##
## Call:
## lm(formula = Sales ~ TV + Radio, data = adv)
##
## Coefficients:
## (Intercept)           TV        Radio
##     2.92110      0.04575      0.18799
```

```
advModMultb1.sum <- summary(advModMultb1)
```

$$\text{Sales} = 0.045 \times \text{TV} + 0.19 \times \text{Radio} + 2.9211$$

All parameters are highly significant and hence the model is deemed adequate, the model explains 100% of the variation, this can be found by appending `$r.squared` to the model object.

(e) Assess the overall accuracy of the model.

In order to assess the model, consider the anova table and derive the RMSE and R^2 values:

```
advModMultb1.anova <- anova(advModMultb1)
advModMultb1.anova
```

```
## Analysis of Variance Table
##
## Response: Sales
##            Df Sum Sq Mean Sq F value    Pr(>F)
## TV          1 3314.6  3314.6 1172.50 < 2.2e-16 ***
## Radio       1 1545.6  1545.6  546.74 < 2.2e-16 ***
## Residuals 197  556.9     2.8
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

From this we can see that that the F statistic is associated with a very low p-value, these are the exact same p-values from the `summary` call.

1. RMSE The RMSE value is the Root mean square error, it is the standard error of the residuals (recall that the standard error is the standard deviation of a model parameter), so the RMSE is basically the standard deviation of the residuals of the model (as measured along the y-axis).

$$\text{RMSE} = \sqrt{\frac{\sum \varepsilon^2}{n}}$$

```r
rmse <- function(model){
# (sum(model$residuals**2)/length(model$residuals))**0.5
sd(advModMultb1$residuals)
}

rse <- function(model){
  var(model$residuals)/var(model$residuals - model$fitted.values)
}

data.frame("RMSE" = rmse(advModMultb1) , "RSE" = rse(advModMultb1) )
```

```
##       RMSE       RSE
## 1 1.672891 0.1028057
```

So the expected error of the model is $\pm$ 1.7 units of sale, we could use this to create a confidence interval by multiplying by the corresponding *Student's t-statistic* and saying that we would expect an observed value to lie within $1.96 * \text{S.E}$ of the model 95% of the time (is this correct or is it the expected mean or something because of the Central Limit Theorem?).

2. Coefficient of Determination The coefficient of determination is 89.7%, this can be returned by extracting the value from the object by apending `$r.squared` to the model summary (i.e.ăuse `summary(lm( Y ~ X1 + X2 ))$r.squared`).

   The R^2 value will be very nearly the same between the initial model and the backwards selected model, however given that the initial model had non-significant predictors it could be considered as over-parameterized, i.e.ăit violates Occam's Razor.

The coefficient of determination is the ratio of the total variance of the data that is explained by the model, in this case it could be determined by:

$$R^2 = \frac{TSS - SSE}{TSS} = \frac{3314.6 + 1546 + 0.1}{3315 + 1545 + 0.1 + 556.8} = 89\%$$

3. Coefficient of Determination from ANOVA The R^2 value is derived from the ANOVA table thusly:

```
advModMultb1.anova <- anova(advModMultb1)
TSS_Multb1 <- advModMultb1.anova$`Sum Sq` %>% sum()
RSS_Multb1 <- advModMultb1$residuals^2 %>% sum()

((TSS_Multb1- RSS_Multb1)/(TSS_Multb1)) %>% signif(3) %>% percent()
    # Requires `scales` package
```

```
## [1] "89.7%"
```

4. Residual analysis When determining the accuracy or performance of amodel, always turn your mind to residual analysis (i.e.ăhow normally are they distributed), this will be performed below.

(f) Calculate the predicted values and residuals

The residuals and fitted values may be returned by extracting them from the model (you could always derive them from first principles but you should use the tools at your disposal, it is quicker, less error prone and makes more readable code)

```
ResDF <- data.frame("Input" = advModMultb1$fitted.values -
    advModMultb1$residuals, "Output" = advModMultb1$fitted.values,
    "Error" = advModMultb1$residuals)
ResDF %>% head()
```

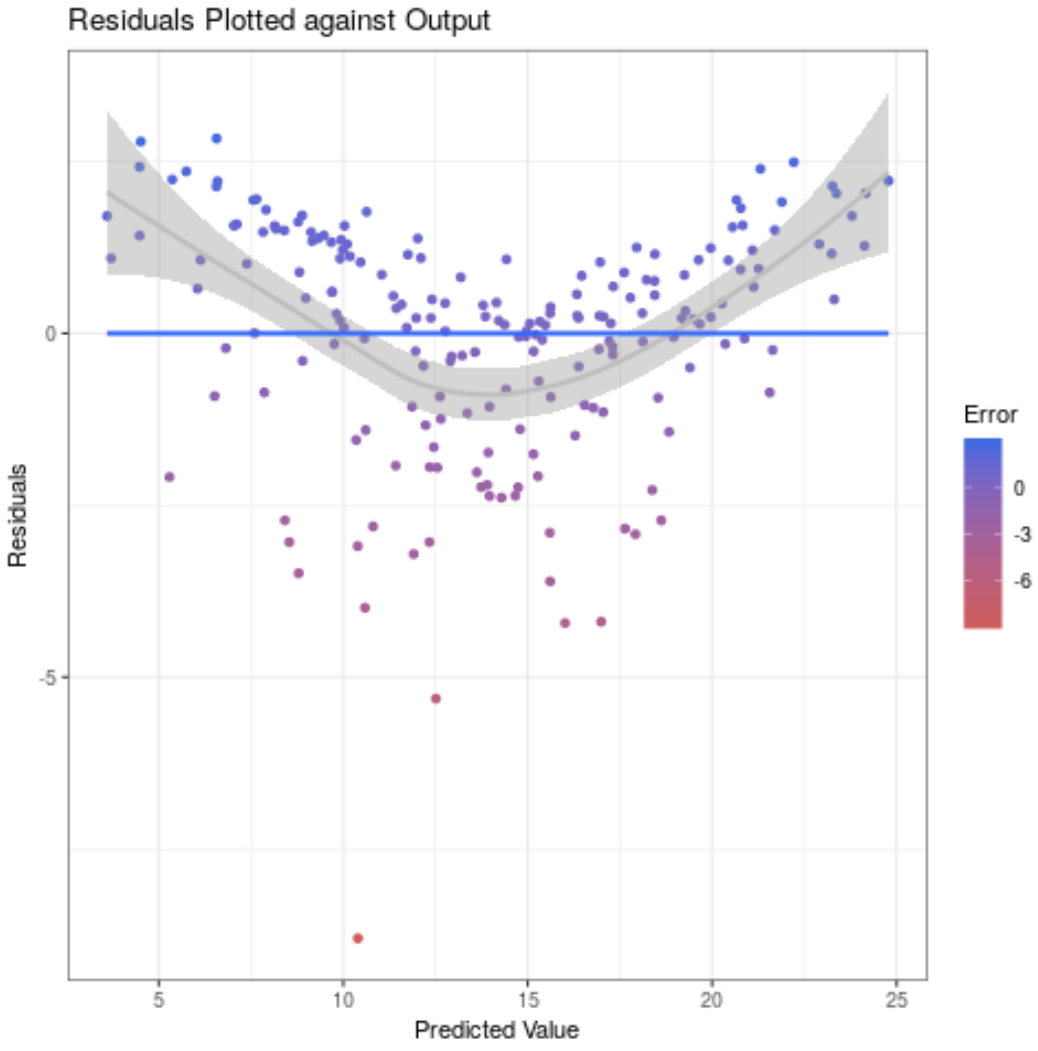
```
##      Input   Output      Error
## 1 19.01093 20.55546  1.5445354
## 2 14.29072 12.34536 -1.9453623
## 3 15.37404 12.33702 -3.0370177
## 4 16.73423 17.61712  0.8828840
## 5 13.54782 13.22391 -0.3239081
## 6 17.82417 12.51208 -5.3120845
```

`Predict` will also return fitted values if no `newdata` is specified.

(g) Plot the residuals against the predicted values

1. ggplot

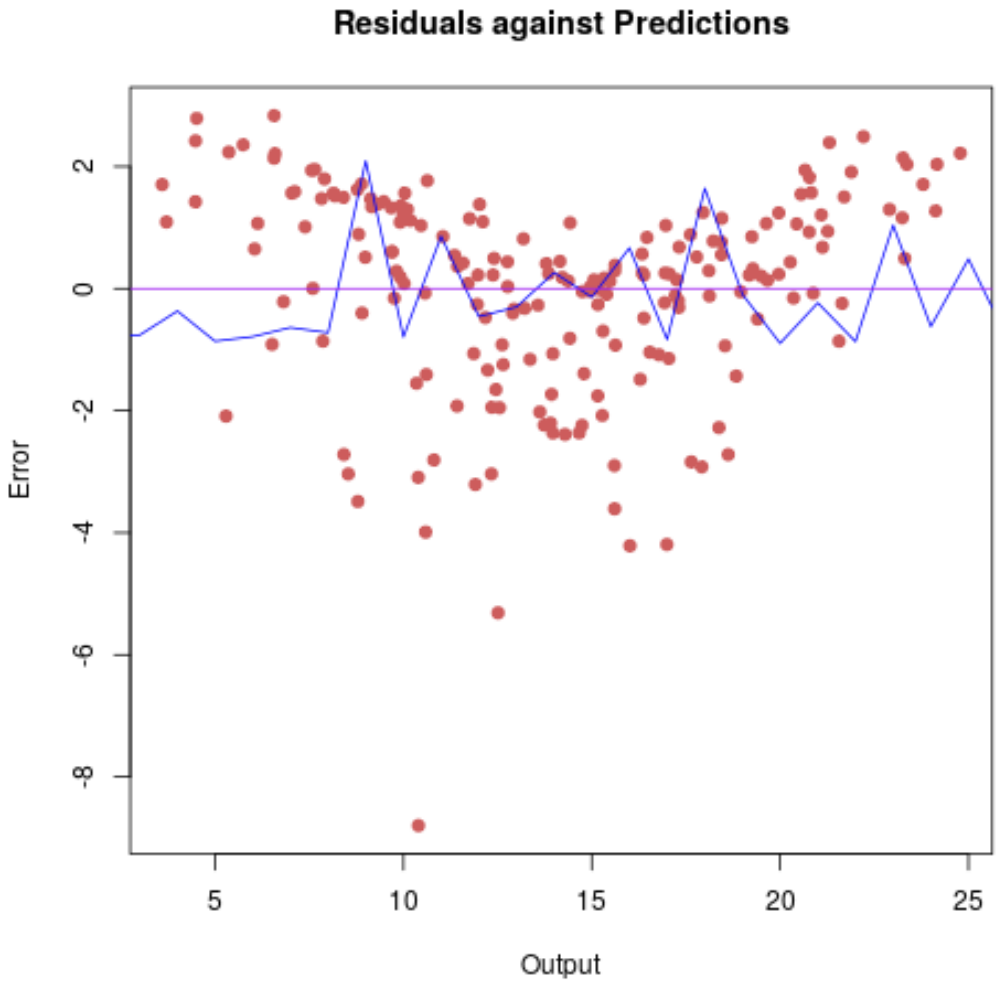
```
ggplot(data = ResDF, aes(x = Output, y = Error, col = Error )) +
  geom_point() +
  theme_bw() +
  stat_smooth(col = "grey")+
  stat_smooth(method = "lm", se = 0, ) +
scale_color_gradient(low = "indianred", high = "royalblue") +
  labs(y = "Residuals", x = "Predicted Value", title = "Residuals
  ↪  Plotted against Output")
```




```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```

2. base

```
plot(Error ~ Output, data = ResDF, pch = 19, col = "IndianRed",
  ↪  main = "Residuals against Predictions")
smooth <- loess(formula = Error ~ Output, data = ResDF, span = 0.8)
predict(smooth) %>% lines(col = "Blue")
abline(lm(Error ~ Output, data = ResDF), col = "Purple")
```



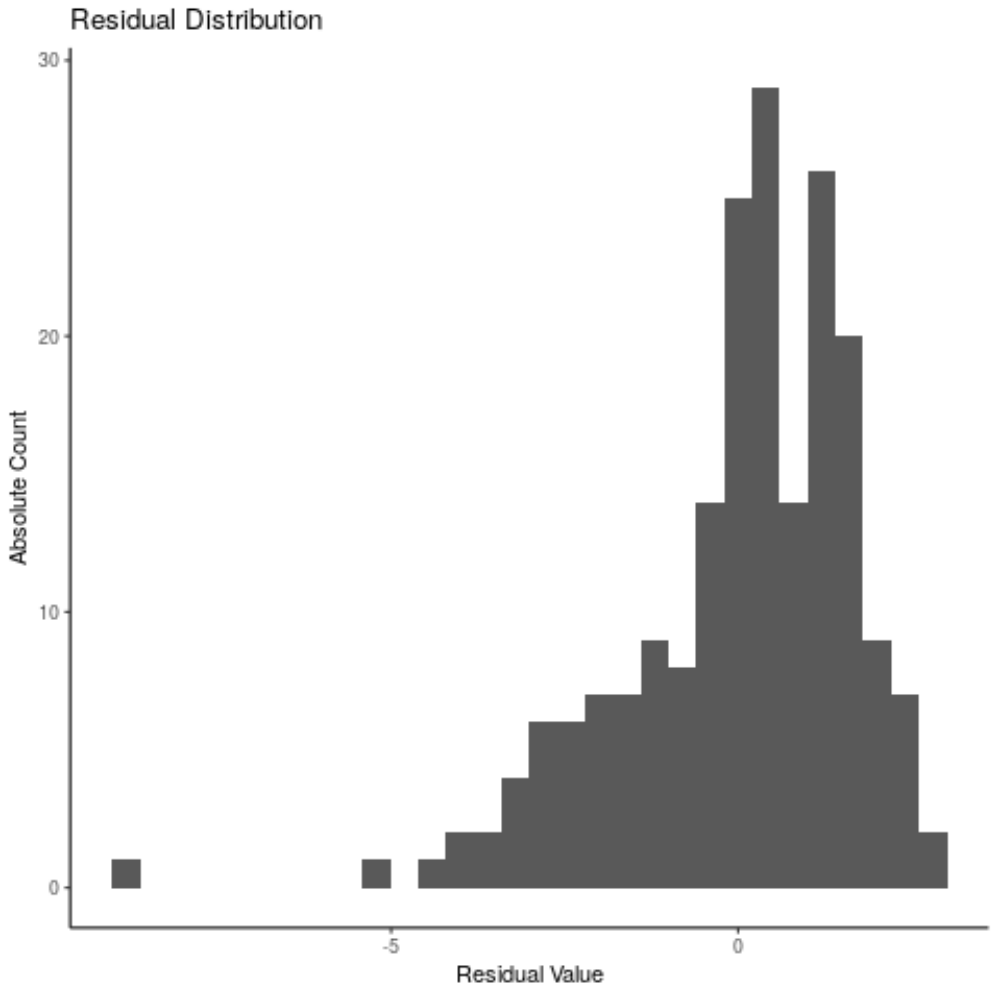
From this it can be seen that the residuals are centred around zero, but they are not normally distributed, the model performance appears to fail normality assumptions for values $\in (10, 20)$

(h) Plot the histogram of the residuals

1. ggplot

   (a) Absolute Count

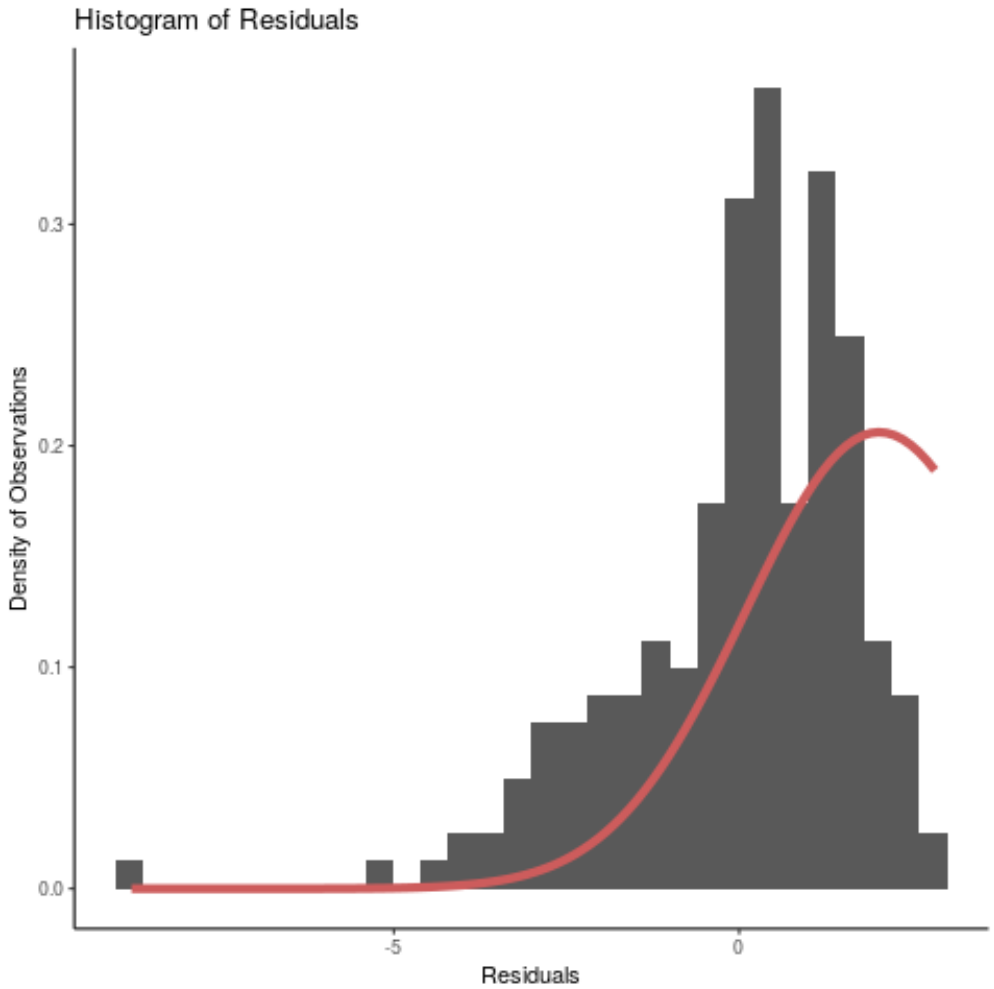
```
ggplot(data = ResDF, aes(x = Error, col = Output)) +
  geom_histogram() +
  theme_classic() +
  labs(y = "Absolute Count", x = "Residual Value", title =
  ↪  "Residual Distribution")
```




```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
(setq org-complete-tags-always-offer-all-agenda-tags t)
(setq org-fast-tag-selection-single-key nil)
```

```r
df <- data.frame(x = rnorm(1000, 2, 2))

# overlay histogram and normal density
ggplot(ResDF, aes(x=Error)) +
  geom_histogram(aes(y = stat(density))) +
  stat_function(
    fun = dnorm,
    args = list(mean = mean(df$x), sd = sd(df$x)),
    lwd = 2,
    col = 'IndianRed'
  ) +
  theme_classic() +
  labs(title = "Histogram of Residuals", y = "Density of
  ↪  Observations", x= "Residuals")
```

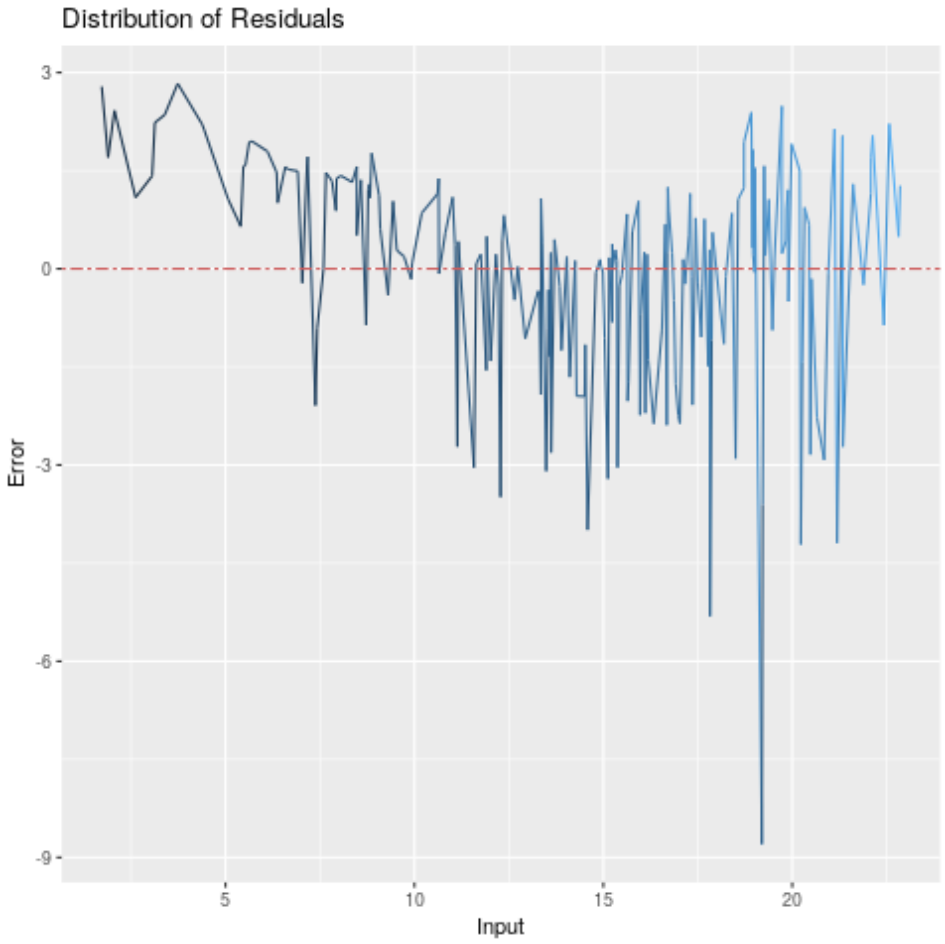



```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

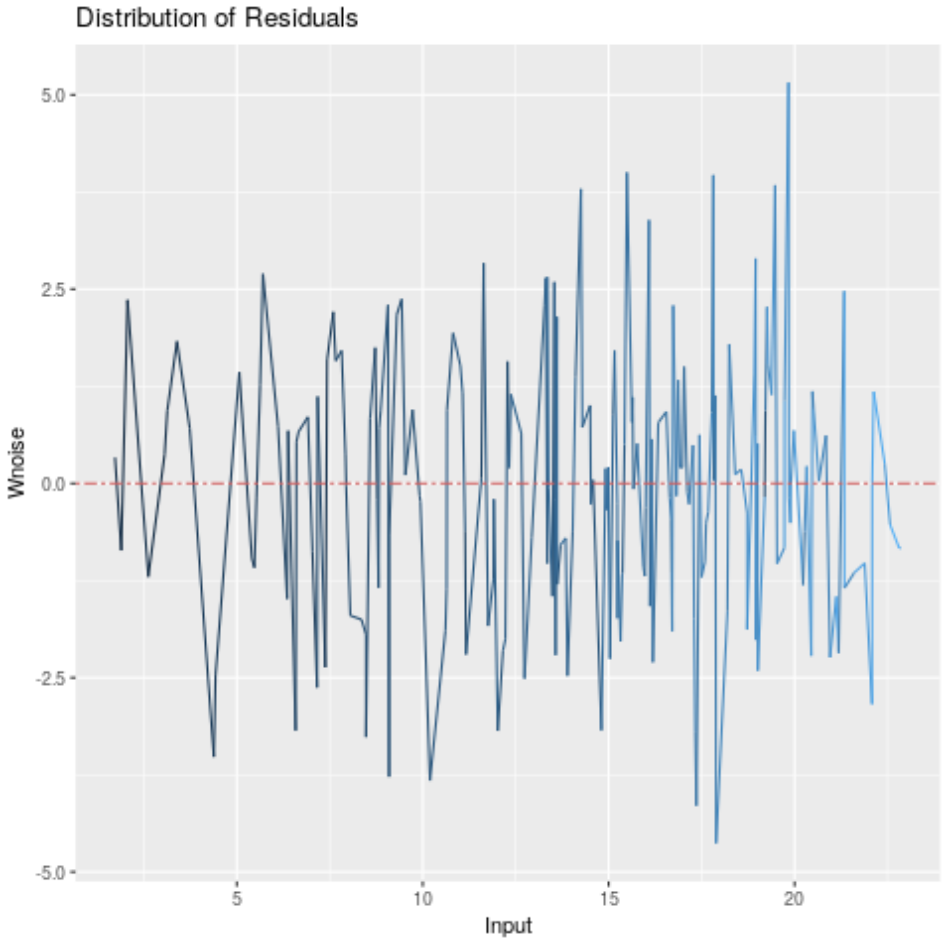
(c) White Noise We can visualise the residuals as white noise:

i. Our Residuals

```r
library(tidyverse)
## Put some White Noise on the ResDF
ResDF <- cbind(ResDF, "Wnoise" = rnorm(nrow(ResDF), mean = 0, sd = sd(ResDF$Error)))
head(ResDF)


## Our Residuals
ggplot(ResDF, aes(x = Input, y = Error, col = Output)) +
  geom_line() +
  geom_abline(slope = 0, intercept = 0, lty = "twodash", col = "IndianRed") +
  theme(legend.position = "none") +
  labs(title = "Distribution of Residuals")
```



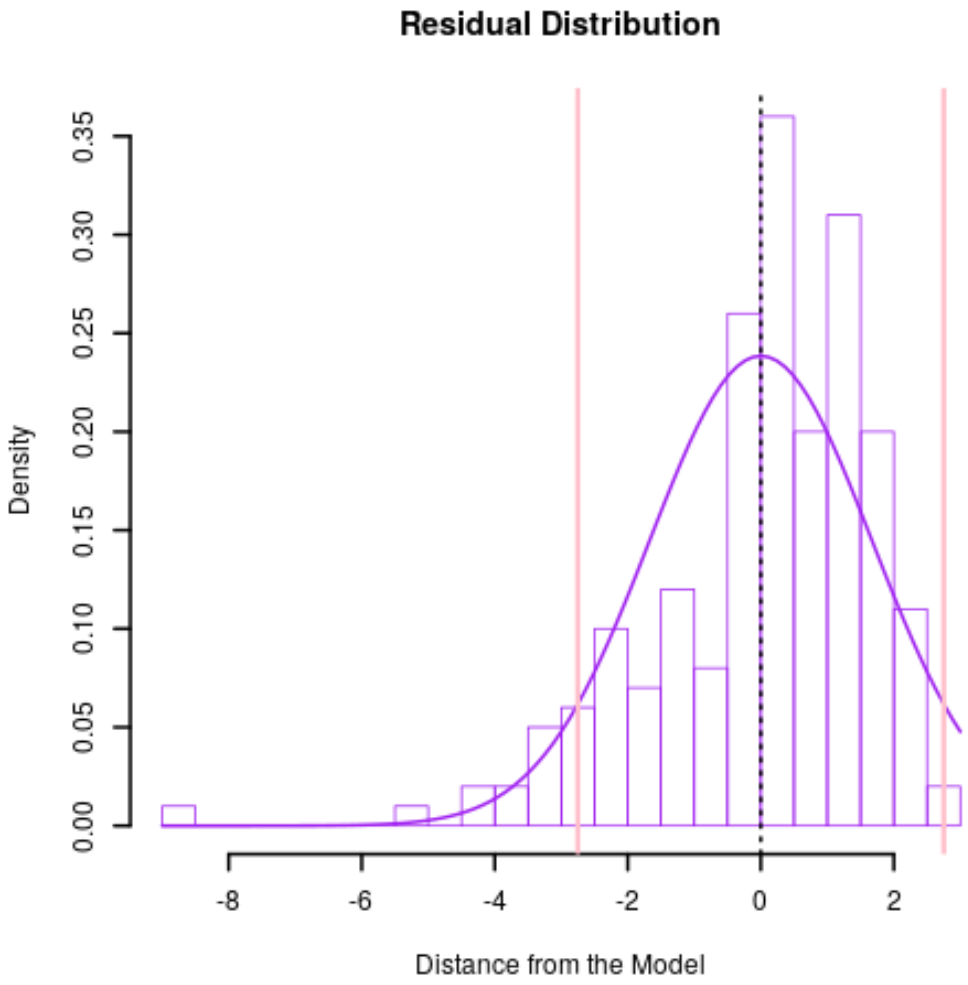
```r
## White Noise

ggplot(ResDF, aes(x = Input, y = Wnoise, col = Output)) +
  geom_line() +
  geom_abline(slope = 0, intercept = 0, lty = "twodash",
  ↪  col = "IndianRed") +
  theme(legend.position = "none") +
  labs(title = "Distribution of Residuals")
```



This clearly shows that the Residuals are non-normal.

2. Base GGPLOT2:HISTOGRAM:BASEPLOT

```r
hist(ResDF$Error
     , breaks = 30,
     prob=TRUE,
     lwd=2,
    main = "Residual Distribution",
    xlab = "Distance from the Model", border = "purple"
    )

  # Overlay the Normal Dist Curve
x <- 1:100    # Stupid base package wants some f(x), this is hacky but easier than stuffing around with lines or defining a function
curve(dnorm(x, mean(ResDF$Error), sd(ResDF$Error)), add=TRUE, col="purple", lwd=2) # Draws the actual density function

# lines(density(possibleyvals_conf), col='purple', lwd=2) # Draws the observed density function
lwr_conf <- qnorm(p = 0.05, mean(ResDF$Error), sd(ResDF$Error))
upr_conf <- qnorm(p = 1-0.05, mean(ResDF$Error), sd(ResDF$Error))
abline(v=upr_conf, col='pink', lwd=3)
abline(v=lwr_conf, col='pink', lwd=3)
abline(v=mean(ResDF$Error), lwd=2, lty='dotted')
```



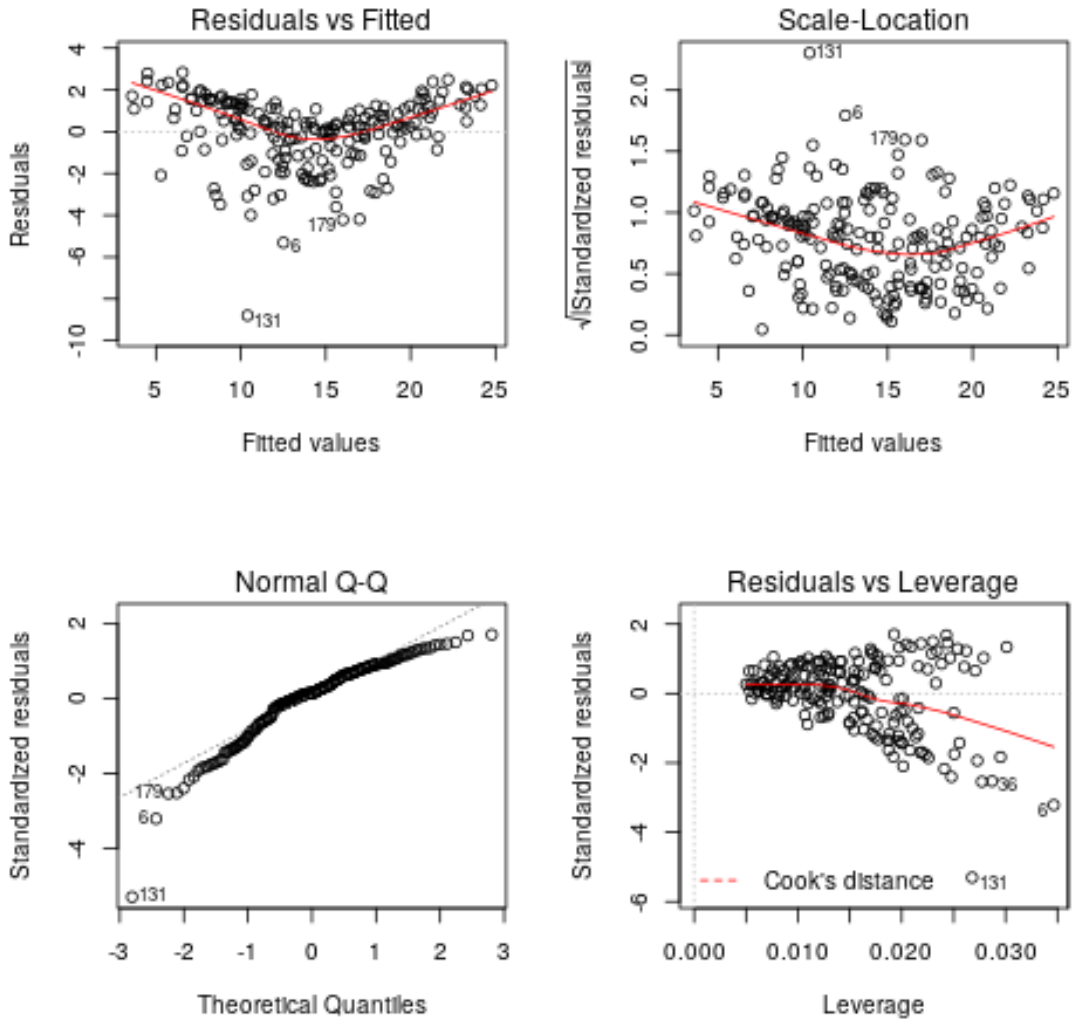

```
lwr_conf <- qnorm(p = 0.05, mean(ResDF$Error), sd(ResDF$Error))
upr_conf <- qnorm(p = 1-0.05, mean(ResDF$Error), sd(ResDF$Error))
```

It hence appears that the Residuals skewed left, which suggests an upper bound on the residual value, the histogram is sufficiently non-normal to reject the assumption that the residuals are normally distributed

(i) Comment on the residual plots

The Residuals can be analised by plotting the model:

```
layout(matrix(data = 1:4, nrow = 2))
plot(advModMultb1)
```



This is an inappropriate model because the residuals are non-normally distributed.

(j) Use the multivariate model for predictions

```r
newdata =  data.frame("TV" = c(3, 1, 2), "Radio" = c(4, 5, 6))
Mod_Adv_Mult_b1 <- predict(object = advModMultb1, newdata)

mypreds <- data.frame("Input" = newdata, "Output" =  Mod_Adv_Mult_b1)
mypreds # Careful with the names, they workout as Input.name in this
  ↪  case.
```

```
##   Input.TV Input.Radio   Output
## 1        3           4 3.810341
## 2        1           5 3.906826
## 3        2           6 4.140575
```

Question 02: Non Linear Models: Use Advertising data set

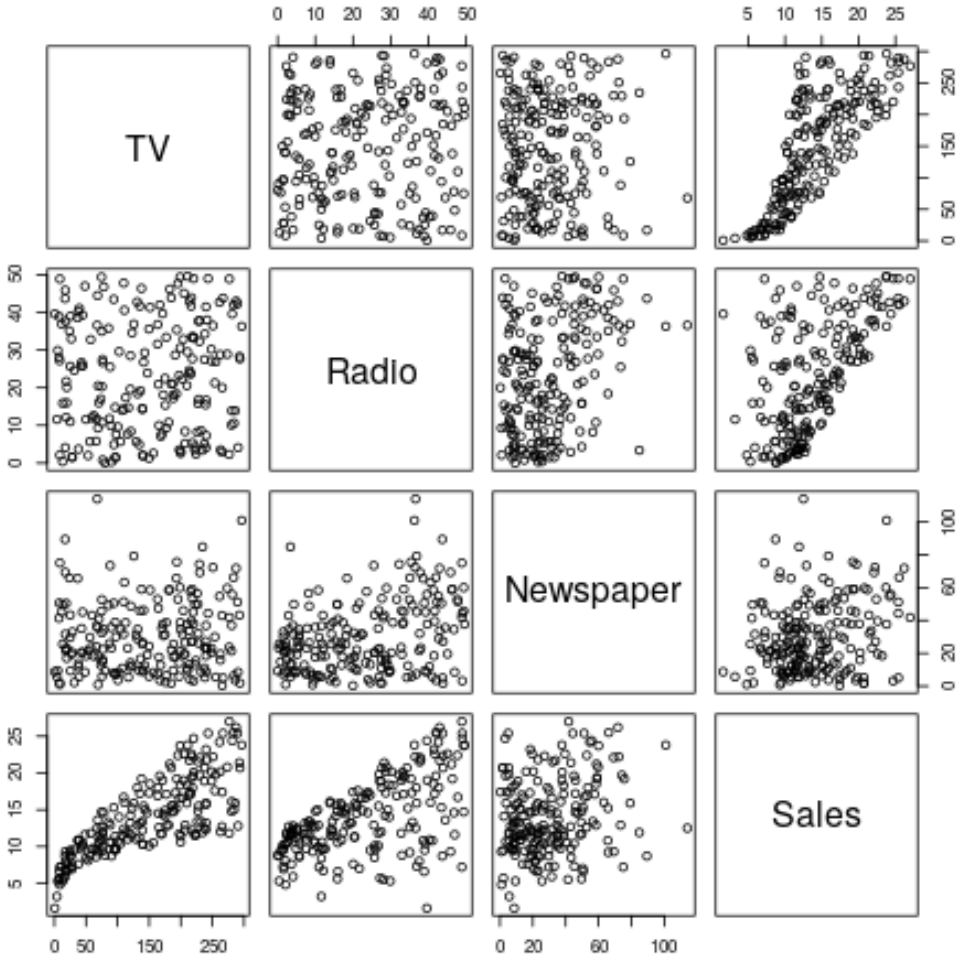
(a) Add the Interaction Term TV*Radio and test the significance of

the interaction term {.tabset} :CUSTOM$_{\text{ID}}$: a-add-the-interaction-term-tvradio-and-test-the-significance-of-the-interaction-term-.tabset

reconsider the correlationplots:

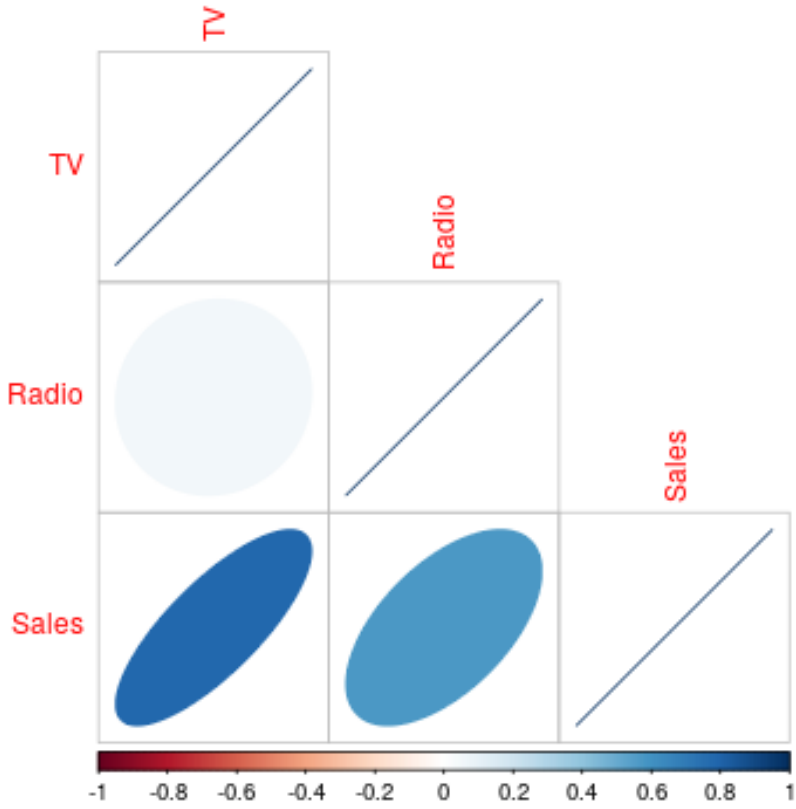
1. Base

```r
pairs(x = adv)
```



corrplot

```
adv[, names(adv)!="Newspaper"] %>% cor() %>% corrplot(method =
↪  "ellipse", type = "lower")
```



From this we can determine that there is no interaction between TV and radio (however previously there was interaction between newspaper and Radio, we should consider that next)

1. Create the MLReg

```
int_mod_adv <- lm(formula = Sales ~ TV * Radio + TV + Radio, data =
    ↪  adv)
int_mod_adv
```

```
##
## Call:
```

```
## lm(formula = Sales ~ TV * Radio + TV + Radio, data = adv)
##
## Coefficients:
## (Intercept)           TV        Radio     TV:Radio
##    6.750220     0.019101     0.028860     0.001086
```

```r
summary(int_mod_adv)
```

```
##
## Call:
## lm(formula = Sales ~ TV * Radio + TV + Radio, data = adv)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.3366 -0.4028  0.1831  0.5948  1.5246
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 6.750e+00  2.479e-01  27.233   <2e-16 ***
## TV          1.910e-02  1.504e-03  12.699   <2e-16 ***
## Radio       2.886e-02  8.905e-03   3.241   0.0014 **
## TV:Radio    1.086e-03  5.242e-05  20.727   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9435 on 196 degrees of freedom
## Multiple R-squared:  0.9678, Adjusted R-squared:  0.9673
## F-statistic:  1963 on 3 and 196 DF,  p-value: < 2.2e-16
```

(b) Give the resulting model after considering this interaction

term. :CUSTOM$_{\text{ID}}$: b-give-the-resulting-model-after-considering-this-interaction-term.

The model will be of the form:

$$\text{Sales} = 0.0019 \times \text{TV} \times + \text{Radio} \times 0.002886 + \text{TV} \times \text{Radio} \times 0.001086$$

Note that all the terms of the model are significant, hence we deem the model as adequate.

(c) Construct the Polynomial Regression Model of order 3 and test

the model significance :CUSTOM$_{\text{ID}}$: c-construct-the-polynomial-regression-model-of-order-3-and-test-the-model-significance

When creating polynomial models there are raw and orthogonal polynomials,

- A raw polynomial will be the the standard System of equations that you get by minimising the RSS

  - The problem with this is that the values will be correlated with each other

- An orthogonal polynomial is a transformed but equivalent polynomial

  - The advantage to this is that the p-value's will be moremeaningful because the coefficients aren't correlated with each other
  - The disadvantage is that the values are not directly related to our data and are not hence meaningful.

```
# Orthogonal (fixes the correlation between the coefficients))

polymodel_Orth <- lm(Sales ~ poly(x = TV, degree = 3, raw = FALSE),
  ↪  data = adv)
summary(polymodel_Orth)
```

```
##
## Call:
## lm(formula = Sales ~ poly(x = TV, degree = 3, raw = FALSE), data = adv)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.9734 -1.8900 -0.0897  2.0189  7.3765
##
## Coefficients:
##                                      Estimate Std. Error t value
## (Intercept)                           14.0225     0.2286  61.353
## poly(x = TV, degree = 3, raw = FALSE)1  57.5727     3.2322  17.812
## poly(x = TV, degree = 3, raw = FALSE)2  -6.2288     3.2322  -1.927
## poly(x = TV, degree = 3, raw = FALSE)3   4.0074     3.2322   1.240
##                                      Pr(>|t|)
## (Intercept)                           <2e-16 ***
## poly(x = TV, degree = 3, raw = FALSE)1   <2e-16 ***
## poly(x = TV, degree = 3, raw = FALSE)2   0.0554 .
## poly(x = TV, degree = 3, raw = FALSE)3   0.2165
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.232 on 196 degrees of freedom
## Multiple R-squared:  0.622,  Adjusted R-squared:  0.6162
## F-statistic: 107.5 on 3 and 196 DF,  p-value: < 2.2e-16
```

```r
# Pure/Raw (has the advantage that you can interpret the coefficients,
↪ but the coefficients will depend on each other and be hence
↪ correlated.)

polymodel <- lm(Sales ~ I(TV*TV*TV) + I(TV*TV) + (TV), data = adv)
summary(polymodel)    # I is used to inhibit the interpretation of * as
↪ relating to the model,
```

```
##
## Call:
## lm(formula = Sales ~ I(TV * TV * TV) + I(TV * TV) + (TV), data = adv)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.9734 -1.8900 -0.0897  2.0189  7.3765
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)      5.420e+00  8.641e-01   6.272 2.23e-09 ***
## I(TV * TV * TV)  5.572e-07  4.494e-07   1.240 0.216519
## I(TV * TV)      -3.152e-04  2.022e-04  -1.559 0.120559
## TV               9.643e-02  2.580e-02   3.738 0.000243 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.232 on 196 degrees of freedom
## Multiple R-squared:  0.622,  Adjusted R-squared:  0.6162
## F-statistic: 107.5 on 3 and 196 DF,  p-value: < 2.2e-16
```

```r
                              # Instead of representing interaction it
                              ↪ represents TV^3

polymodel_Raw<- lm(Sales ~ poly(x = TV, degree = 3, raw = TRUE), data =
↪ adv)
summary(polymodel_Raw)
```

```
##
## Call:
## lm(formula = Sales ~ poly(x = TV, degree = 3, raw = TRUE), data = adv)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.9734 -1.8900 -0.0897  2.0189  7.3765
```

```
##
## Coefficients:
##                                      Estimate Std. Error t value
## (Intercept)                          5.420e+00  8.641e-01   6.272
## poly(x = TV, degree = 3, raw = TRUE)1  9.643e-02  2.580e-02   3.738
## poly(x = TV, degree = 3, raw = TRUE)2 -3.152e-04  2.022e-04  -1.559
## poly(x = TV, degree = 3, raw = TRUE)3  5.572e-07  4.494e-07   1.240
##                                     Pr(>|t|)
## (Intercept)                         2.23e-09 ***
## poly(x = TV, degree = 3, raw = TRUE)1 0.000243 ***
## poly(x = TV, degree = 3, raw = TRUE)2 0.120559
## poly(x = TV, degree = 3, raw = TRUE)3 0.216519
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.232 on 196 degrees of freedom
## Multiple R-squared:  0.622,  Adjusted R-squared:  0.6162
## F-statistic: 107.5 on 3 and 196 DF,  p-value: < 2.2e-16
```

The 3rd degree coefficient is not significant, hence we consider the 2nd degree:

```
quadmod <- lm(formula = Sales ~ I(TV*TV) + TV, data = adv)
summary(quadmod)
```

```
##
## Call:
## lm(formula = Sales ~ I(TV * TV) + TV, data = adv)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.6844 -1.7843 -0.1562  2.0088  7.5097
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  6.114e+00  6.592e-01   9.275  < 2e-16 ***
## I(TV * TV)  -6.847e-05  3.558e-05  -1.924   0.0557 .
## TV           6.727e-02  1.059e-02   6.349 1.46e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.237 on 197 degrees of freedom
## Multiple R-squared:  0.619,  Adjusted R-squared:  0.6152
## F-statistic: 160.1 on 2 and 197 DF,  p-value: < 2.2e-16
```

The quadratic term is not sufficiently significant so the model is rejected

(d) Give the resulting selected model

The model selected is the multiple linear regression with the interaction from before:

$$\text{Sales} = 0.0019 \times \text{TV} \times + \text{Radio} \times 0.002886 + \text{TV} \times \text{Radio} \times 0.001086$$

This was converted from ‘md‘ to ‘org‘ using ‘pandoc -f gfm‘ at time: 2020-02-08T05-00-13