Applied Linear Models

STAT 331

Fall 2020 (1209)[1]

Cameron Roopnarine[2] Samuel Wong[3]

March 3, 2024

[1]Online Course
[2]LaTeXer
[3]Instructor

Contents

Contents 1

1 Simple Linear Regression 3
- 1.1 Introduction . . . 3
- 1.2 Inference . . . 5
- 1.3 Prediction . . . 8
- 1.4 Application . . . 9
  - 1.4.1 R Demo . . . 10

2 Multiple Linear Regression 14
- 2.1 Random Vectors . . . 14
- 2.2 Multivariate Normal Distribution . . . 17
- 2.3 Inference . . . 17
- 2.4 Application . . . 20
- 2.5 Prediction . . . 20
- 2.6 Categorical Predictors . . . 23
  - 2.6.1 R Demo . . . 23
- 2.7 Analysis of Variance . . . 29
- 2.8 Coefficient of Determination . . . 29
- 2.9 General Linear Hypothesis Tests Based on F Distribution . . . 30
- 2.10 ANOVA F Test . . . 32
  - 2.10.1 R Demo . . . 33
    - 2.10.1.1 F-distribution Examples . . . 34
    - 2.10.1.2 Random numbers for the F-distribution . . . 36
    - 2.10.1.3 Revisit Rocket Example . . . 39
    - 2.10.1.4 Revist Coffee Example (Coffee Quality Institute, 2018) . . . 41
- 2.11 Multicollinearity . . . 45
- 2.12 Detection of Multicollinearity . . . 45
  - 2.12.1 R Demo . . . 46
- 2.13 Model Selection Criteria . . . 51
- 2.14 Model Selection Basic Strategies . . . 54
  - 2.14.1 R Demo . . . 55
- 2.15 Residual Plots for Linear Regression Assumptions . . . 61
  - 2.15.1 R Demo . . . 62
- 2.16 Addressing Problems With Regression Model Assumptions . . . 69
  - 2.16.1 R Demo . . . 71
- 2.17 Effects of Individual Observations . . . 82
  - 2.17.1 R Demo . . . 84
- 2.18 Prediction Error . . . 90
- 2.19 Cross-Validation . . . 91
  - 2.19.1 R Demo . . . 92

2.20 Combining Cross Validation With Model Selection 94
2.21 Iterative Conditional Minimization 95
2.21.1 R Demo 95

3 Generalized Linear Models **99**
3.1 Beyond STAT 331: Generalized Linear Models and Logistic Regression 99
3.2 Last Lecture! 101

Introduction

Lecture 1 | 2020-09-08

DEFINITION 0.0.1: Response variable

A **response** (**dependent**) **variable** is the primary variable of interest, denoted by a capital roman letter Y.

DEFINITION 0.0.2: Explanatory Variable

An **explanatory** (**independent**, **predictor**) **variable** are variables that impact the response, denoted by x_i for $i = 1, \dots, p$.

DEFINITION 0.0.3: Regression Model

A **regression model** deals with modelling the functional relationship between a response variable and one or more explanatory variables.

EXAMPLE 0.0.4: Alligators in Florida

Let Y be the length in metres of an alligator and $x_1 := \{0, 1\}$ (male or female). The mass in an alligators stomach consists of fish (x_2), invertebrates (x_3), reptiles (x_4), birds (x_5), and other ($x_6, \dots, x_p$). We imagine we can explain Y in terms of $(x_1, \dots, x_p)$ using some function such that $Y = f(x_1, \dots, x_p)$.

In this course, we will be looking at linear models.

DEFINITION 0.0.5: Linear model

A general **linear model** is defined as $Y = \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p + \varepsilon$ where Y is the response variable, $(x_1, \dots, x_p)$ are the p explanatory variables, $(\beta_0, \beta_1, \dots, \beta_p)$ are the model parameters, and ε is the random error. We assume that $(x_1, \dots, x_p)$ are fixed constants, β_0 is the intercept of Y, $(\beta_1, \dots, \beta_p)$ all quantify effect on x_j on Y, and $\varepsilon \sim \mathcal{N}(0, \sigma^2)$.

REMARK 0.0.6

In general, the model will not perfectly explain the data.

"All models are wrong, but some are useful."

$Y \sim \mathcal{N}(\beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p, \sigma^2)$ since $\mathbb{E}[Y] = \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p$ and $\mathbb{V}(Y) = \mathbb{V}(\varepsilon) = \sigma^2$.

Chapter 1

Simple Linear Regression

LECTURE 2 | 2020-09-09

1.1 Introduction

DEFINITION 1.1.1: Simple linear regression

A **simple linear regression** is a linear model that uses only one explanatory variable; that is, $Y = \beta_0 + \beta_1 x + \varepsilon$. The **data** in a simple linear regression consists of pairs (x_i, y_i) where $i = 1, \dots, n$.

REMARK 1.1.2

Before fitting any model, we might want to make a scatter plot to visualize if there is a linear relationship between x and y, or calculate the *correlation*.

DEFINITION 1.1.3: Correlation

The **correlation** of random variables X and Y is $\rho_{XY} = \dfrac{\text{Cov}(X,Y)}{\text{Sd}(X)\text{Sd}(Y)}$.

DEFINITION 1.1.4: Sample correlation

The **sample correlation** of all pairs (x_i, y_i) is

$$\begin{aligned} r &= \frac{\frac{1}{n-1}\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\frac{1}{n-1}\sum_{i=1}^n (x_i - \bar{x})^2}\sqrt{\frac{1}{n-1}\sum_{i=1}^n (y_i - \bar{y})^2}} \\ &= \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^n (x_i - \bar{x})^2}\sqrt{\sum_{i=1}^n (y_i - \bar{y})^2}} \\ &= \frac{S_{xy}}{\sqrt{S_{xx}S_{yy}}} \end{aligned}$$

REMARK 1.1.5

The sample correlation measures the strength and direction of the linear relationship between x and y. Note that $-1 \leqslant r \leqslant 1$. If $|r| \approx 1$, then there is a strong linear relationship, and if $|r| \approx 0$ then there is a lack of linear relationship. Also, if $r > 0$, then there is a positive relationship, and if $r < 0$ then there is a negative relationship. It does not tell us how to predict y from x. To do so, we need to estimate β_0 and β_1.

DEFINITION 1.1.6: Simple linear regression model

For data (x_i, y_i) for $i = 1, \dots, n$, the **simple linear regression model** is $Y_i = \beta_0 + \beta_1 x_i + \varepsilon_i$ with the assumption that $\varepsilon_i \overset{\text{iid}}{\sim} \mathcal{N}(0, \sigma^2)$. Therefore, $Y_i \sim \mathcal{N}(\mu_i = \beta_0 + \beta_1 x_i, \sigma^2)$.

DEFINITION 1.1.7: Least squares

The method of estimating β_0 and β_1 by minimizing $S(\beta_0, \beta_1) = \sum_{i=1}^n (y_i - (\beta_0 + \beta_1 x_i))^2$ is referred to as the **method of the least squares**.

REMARK 1.1.8

The least squares is equivalent to maximum likelihood estimate when $\varepsilon_i \overset{\text{iid}}{\sim} \mathcal{N}(0, \sigma^2)$.

THEOREM 1.1.9: Least Square Estimates (LSEs) for SLR

Minimizing $S(\beta_0, \beta_1)$, gives the least square estimates

$$\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x} \quad \textit{and} \quad \hat{\beta}_1 = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2} = \frac{S_{xy}}{S_{xx}}$$

Proof of 1.1.9

$\frac{\partial S}{\partial \beta_0} = 2\sum_{i=1}^n [y_i - (\beta_0 + \beta_1 x_i)](-1)$ and $\frac{\partial S}{\partial \beta_1} = 2\sum_{i=1}^n [y_i - (\beta_0 + \beta_1 x_i)](-x_i)$.
Now,

$$\frac{dS}{d\beta_0} := 0 \iff \sum_{i=1}^n y_i - n\beta_0 - \beta_1 \sum_{i=1}^n x_i = 0 \iff \beta_0 = \bar{y} - \beta_1 \bar{x}$$

$$\begin{aligned}
\frac{dS}{d\beta_1} := 0 &\overset{\text{plug } \beta_0}{\iff} \sum_{i=1}^n [y_i - \bar{y} + \beta_1 \bar{x} - \beta_1 x_i] x_i = 0 \\
&\iff \sum_{i=1}^n x_i (y_i - \bar{y}) - \beta_1 \sum_{i=1}^n x_i (x_i - \bar{x}) = 0 \\
&\iff \beta_1 = \frac{\sum_{i=1}^n x_i (y_i - \bar{y})}{\sum_{i=1}^n x_i (x_i - \bar{x})} = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2} = \frac{S_{xy}}{S_{xx}}
\end{aligned}$$

REMARK 1.1.10

We use a hat on the β's to show that they are estimates.

DEFINITION 1.1.11: Fitted value, Residual

The expression $\hat{\mu}_i = \hat{\beta}_0 + \hat{\beta}_1 x_i$ is called the **fitted value** that corresponds to the i^{th} observation with x_i as the explanatory variable. The difference between y_i and $\hat{\mu}_i$, and $e_i = y_i - \hat{\mu}_i$ is referred to as the **residual**. It is the vertical distance between the observation y_i and the estimated line $\hat{\mu}_i$ evaluated at x_i.

LECTURE 3 | 2020-09-14

1.2 Inference

For $Y_i \sim \mathcal{N}(\beta_0 + \beta_1 x_i, \sigma^2)$, the equation of fitted line is given by $y = \hat{\beta}_0 + \hat{\beta}_1 x$. Our interpretation of the parameters is as follows.

- $\hat{\beta}_0$ is the estimate of the expected response when $x = 0$ (but not always meaningful if outside range of x_i's in data)
- $\hat{\beta}_1$ is the estimate of expected change in response for unit increase in x
- σ^2 is the "variability around the line" where $\sigma^2 = \mathbb{V}(\varepsilon_i) = \mathbb{V}(Y_i)$

Q: How should we estimate σ^2?

$$\varepsilon_i = Y_i - (\beta_0 + \beta_1 x_i) \quad \text{and} \quad e_i = y_i - (\hat{\beta}_0 + \hat{\beta}_1 x_i)$$

Our intuition tells us to use variability in the residuals to estimate σ^2, so we use

$$\hat{\sigma}^2 = \frac{\sum_{i=1}^n (e_i - \bar{e})^2}{n-2} = \frac{\sum_{i=1}^n e_i^2}{n-2}$$

where the first term looks like sample variance of e_i's. The second equality follows since $\bar{e} = \bar{y} - (\hat{\beta}_0 + \hat{\beta}_1 \bar{x}) = 0$ by definition of our β_0 estimate.

DEFINITION 1.2.1: Residual sum of squares

SS(Res) $= \sum_{i=1}^n (y_i - \hat{\mu}_i)^2 = \sum_{i=1}^n e_i^2$, is known as the **residual (error) sum of squares**.

REMARK 1.2.2

The $n-2$ will be looked at in more detail later, but for now it suffices to say that the degrees of freedom is $n-2$ or equivalently, $n -$ number of parameters estimated. It allows $\hat{\sigma}^2$ to be an unbiased estimator for the true value of σ^2; that is, $\mathbb{E}[\hat{\sigma}^2] = \sigma^2$ whenever $\hat{\sigma}^2$ is viewed as a random variable.

THEOREM 1.2.3: Linear Combination of Independent Normal Random Variables

If $Y_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$, $i = 1, \ldots, n$ independently, then

$$\sum_{i=1}^n a_i Y_i \sim N\left(\sum_{i=1}^n a_i \mu_i, \sum_{i=1}^n a_i^2 \sigma_i^2\right)$$

Proof of 1.2.3

The proof is completed in STAT 330 with moment generating functions.

Viewing $\hat{\beta}_1$ as a random variable:

$$\hat{\beta}_1 = \frac{\sum_{i=1}^n (x_i - \bar{x})(Y_i - \bar{Y})}{\sum_{i=1}^n (x_i - \bar{x})^2} = \frac{\sum_{i=1}^n (x_i - \bar{x})Y_i - \bar{Y}\overbrace{\sum_{i=1}^n (x_i - \bar{x})}^{0}}{\sum_{i=1}^n (x_i - \bar{x})x_i - \bar{x}\underbrace{\sum_{i=1}^n (x_i - \bar{x})}_{0}} = \frac{\sum_{i=1}^n (x_i - \bar{x})Y_i}{\sum_{i=1}^n (x_i - \bar{x})x_i} = \sum_{i=1}^n a_i Y_i$$

where $a_i = \dfrac{x_i - \bar{x}}{\sum_{i=1}^n x_i(x_i - \bar{x})}$. Therefore,

$$\mathbb{E}[\hat{\beta}_1] = \sum_{i=1}^n a_i \mathbb{E}[Y_i] = \frac{\sum_{i=1}^n (x_i - \bar{x})(\beta_0 + \beta_1 x_i)}{\sum_{i=1}^n x_i(x_i - \bar{x})} = \frac{\beta_0 \overbrace{\sum_{i=1}^n (x_i - \bar{x})}^{0} + \beta_1 \sum_{i=1}^n x_i(x_i - \bar{x})}{\sum_{i=1}^n x_i(x_i - \bar{x})} = \beta_1$$

Now, we calculate the variance of $\hat{\beta}_1$:

$$\mathbb{V}(\hat{\beta}_1) = \sum_{i=1}^n a_i^2 \mathbb{V}(Y_i) = \sigma^2 \frac{\sum_{i=1}^n (x_i - \bar{x})^2}{[\sum_{i=1}^n x_i(x_i - \bar{x})]^2} = \sigma^2 \frac{\sum_{i=1}^n (x_i - \bar{x})^2}{[\sum_{i=1}^n (x_i - \bar{x})^2]^2} = \frac{\sigma^2}{S_{xx}}$$

Using our calculations from $\hat{\beta}_1$, and viewing $\hat{\beta}_0$ as a random variable:

$$\mathbb{E}[\hat{\beta}_0] = \mathbb{E}[\bar{Y}] - \bar{x}\mathbb{E}[\hat{\beta}_1] = \mathbb{E}\left[\frac{\sum_{i=1}^n Y_i}{n}\right] - \bar{x}\beta_1 = \frac{\sum_{i=1}^n (\beta_0 + \beta_1 x_i)}{n} - \beta_1 \bar{x} = \beta_0 + \beta_1 \bar{x} - \beta_1 \bar{x} = \beta_0$$

Now, we calculate the variance of $\hat{\beta}_0$:

$$\begin{aligned}
\mathbb{V}(\hat{\beta}_0) &= \mathbb{V}(\bar{Y} - \hat{\beta}_1 \bar{x}) \\
&= \mathbb{V}(\bar{Y}) + (-\bar{x})^2 \mathbb{V}(\hat{\beta}_1) + 2(1)(-1)(\bar{x}) \underbrace{\mathsf{Cov}(\bar{Y}, \hat{\beta}_1)}_{0} \\
&= \mathbb{V}\left(\frac{\sum_{i=1}^n Y_i}{n}\right) + \bar{x}^2 \left(\frac{\sigma^2}{S_{xx}}\right) \\
&= \frac{n\sigma^2}{n^2} + \frac{\sigma^2 x^2}{S_{xx}},
\end{aligned}$$

where we used the fact that $\mathsf{Cov}(\,\cdot\,,\,\cdot\,)$ is linear to get

$$\begin{aligned}
\mathsf{Cov}(\bar{Y}, \hat{\beta}_1) &= \mathsf{Cov}\left(\frac{1}{n}\sum_{i=1}^n Y_i, \sum_{i=1}^n a_i Y_i\right) \\
&= \frac{1}{n}\sum_{i=1}^n a_i \underbrace{\mathsf{Cov}(Y_i, Y_i)}_{\mathbb{V}(Y_i)} \\
&= \frac{\sigma^2}{n}\sum_{i=1}^n a_i \\
&= 0.
\end{aligned}$$

Also, since $\hat{\beta}_1$ and $\hat{\beta}_0$ are linear combination of Normal random variables, they follow a Normal distribution. Therefore, we get the following theorem.

THEOREM 1.2.4: Distribution of LSEs

The distribution of the least square estimates are given by

$$\hat{\beta}_1 \sim \mathcal{N}\left(\beta_1, \frac{\sigma^2}{S_{xx}}\right) \quad \textit{and} \quad \hat{\beta}_0 \sim \mathcal{N}\left(\beta_0, \sigma^2\left(\frac{1}{n} + \frac{\bar{x}^2}{S_{xx}}\right)\right)$$

Since $\mathbb{E}[\hat{\beta}_1] = \beta_1$, we say $\hat{\beta}_1$ is an unbiased estimator of β_1. This implies that as $n \to \infty$, the average of the estimates $\hat{\beta}_1$; that is, $\mathbb{E}[\hat{\beta}_1]$ coincides with the true value of β_1. A similar argument can be made for β_0.

Then, $\dfrac{\hat{\beta}_1 - \beta_1}{\sigma/\sqrt{S_{xx}}} \sim \mathcal{N}(0,1)$, but σ is unknown, so need to use $\hat{\sigma}$ to get $\dfrac{\hat{\beta}_1 - \beta_1}{\hat{\sigma}/\sqrt{S_{xx}}} \sim t(n-2)$.

DEFINITION 1.2.5: Standard deviation and standard error of $\hat{\beta}_1$

The **standard deviation** of $\hat{\beta}_1$ is defined as $\mathrm{Sd}(\hat{\beta}_1) = \sigma/\sqrt{S_{xx}}$. The **estimated** standard deviation of $\hat{\beta}_1$ is also referred to as the **standard error** of the estimate $\hat{\beta}_1$, and we write $\mathrm{Se}\left(\hat{\beta}_1\right) = \hat{\sigma}/\sqrt{S_{xx}}$.

DEFINITION 1.2.6: Student t distribution

Suppose $Z \sim \mathcal{N}(0,1)$ and $U \sim \chi^2(\nu)$, with Z and U independent. Then, $T = Z/\sqrt{U/\nu}$ has a **Student t distribution** with ν degrees of freedom.

THEOREM 1.2.7

For a simple linear regression model,

$$\frac{\hat{\sigma}^2(n-2)}{\sigma^2} = \frac{SS(Res)}{\sigma^2} \sim \chi^2(n-2)$$

Proof of 1.2.7

Too hard for sure.

Using the theorem stated, we justify the fact that replacing σ with $\hat{\sigma}$ gives us a $t(n-2)$ distribution.

$$\frac{\hat{\beta}_1 - \beta_1}{\hat{\sigma}/\sqrt{S_{xx}}} = \frac{\dfrac{\hat{\beta}_1 - \beta_1}{\sigma/\sqrt{S_{xx}}}}{\sqrt{\dfrac{\hat{\sigma}^2(n-2)}{\sigma^2}\left(\dfrac{1}{n-2}\right)}} = \frac{Z}{\sqrt{U/\nu}} = T \sim t(n-2)$$

where $\dfrac{\hat{\sigma}^2(n-2)}{\sigma^2} = U$, $\nu = n-2$, and $Z = \dfrac{\hat{\beta}_1 - \beta_1}{\hat{\sigma}/\sqrt{S_{xx}}}$. A $(1-\alpha)$ confidence interval for β_1 is

$$\hat{\beta}_1 \pm c\,\mathrm{Se}\left(\hat{\beta}_1\right)$$

where c is the $1 - \dfrac{\alpha}{2}$ quantile of $t(n-2)$; that is, $P(|T| \leqslant c) = 1-\alpha$ or $P(T \leqslant c) = 1 - \dfrac{\alpha}{2}$ where $T \sim t(n-2)$.

Hypothesis test: H_0: $\beta = 0$ versus H_A: $\beta_1 \neq 0$. If H_0 is true, then $\hat{\beta}_1/\mathrm{Se}\left(\hat{\beta}_1\right) \sim t(n-2)$, so calculate the **t statistic** $t = \hat{\beta}_1/\mathrm{Se}\left(\hat{\beta}_1\right)$, and reject H_0 at level α if $|t| > c$ where c is $1 - \dfrac{\alpha}{2}$ quantile of $t(n-2)$. Therefore, p-value $= P(|T| \geqslant |t|) = 2P(T \geqslant |t|)$.

1.3 Prediction

Suppose we want to predict the response y for a new value of x, say $x = x_0$. Then, SLR model says $Y_0 \sim \mathcal{N}(\beta_0 + \beta_1 x_0, \sigma^2)$ where Y_0 is a random variable for response when $x = x_0$; that is, $\hat{Y}_0 = \hat{\beta}_0 + \hat{\beta}_1 x_0$. The fitted model predicts the *value* of y to be $\hat{y}_0 = \hat{\beta}_0 + \hat{\beta}_1 x_0$.

Also, $\mathbb{E}[\hat{Y}_0] = \mathbb{E}[\hat{\beta}_0] + x_0\mathbb{E}[\hat{\beta}_1] = \beta_0 + \beta_1 x_0 = \mathbb{E}[Y_0]$, since $\hat{\beta}_i$ for $i = 0, 1$ are unbiased. Therefore, we can say that $\hat{Y}_0$ is an unbiased estimate of the random variable for the mean of Y_0. For the variance of $\hat{Y}_0$ we write

$$\begin{aligned}
\hat{Y}_0 &= \hat{\beta}_0 + \hat{\beta}_1 x_0 = \bar{Y} - \hat{\beta}_1 \bar{x} + \hat{\beta}_1 x_0 \\
&= \bar{Y} + \hat{\beta}_1 (x_0 - \bar{x}) \\
&= \sum_{i=1}^{n} \left[\frac{Y_i}{n} + (x_0 - \bar{x}) \left(\frac{(x_i - \bar{x})(Y_i - \bar{Y})}{S_{xx}} \right) \right] \\
&= \sum_{i=1}^{n} \left[\frac{Y_i}{n} + (x_0 - \bar{x}) \left(\frac{(x_i - \bar{x}) Y_i}{S_{xx}} \right) \right] \\
&= \sum_{i=1}^{n} \left[\frac{1}{n} + \frac{(x_0 - \bar{x})(x_i - \bar{x})}{S_{xx}} \right] Y_i \\
&= \sum_{i=1}^{n} a_i Y_i
\end{aligned}$$

where $a_i = \dfrac{1}{n} + \dfrac{(x_0 - \bar{x})(x_i - \bar{x})}{S_{xx}}$. Therefore,

$$\begin{aligned}
\mathbb{V}(Y_0) &= \sum_{i=1}^{n} \left[\frac{1}{n} + \frac{(x_0 - \bar{x})(x_i - \bar{x})}{S_{xx}} \right]^2 \\
&= \sum_{i=1}^{n} \left[\frac{1}{n^2} + \frac{2(x_0 - \bar{x})(x_i - \bar{x})}{n S_{xx}} + \frac{(x_0 - \bar{x})^2 (x_i - \bar{x})^2}{(S_{xx})^2} \right] \\
&= \sum_{i=1}^{n} \left[\frac{1}{n^2} \right] + \frac{2(x_0 - \bar{x})}{n S_{xx}} \sum_{i=1}^{n} (x_i - \bar{x}) + \frac{(x_0 - \bar{x})^2}{(S_{xx})^2} \sum_{i=1}^{n} (x_i - \bar{x})^2 \\
&= \frac{1}{n} + \frac{2(x_0 - \bar{x})}{S_{xx}} (0) + \frac{(x_0 - \bar{x})^2}{(S_{xx})^2} (S_{xx}) \\
&= \frac{1}{n} + \frac{(x_0 - \bar{x})^2}{S_{xx}}
\end{aligned}$$

We proved the following theorem.

THEOREM 1.3.1: Distribution of Prediction

The distribution of the prediction random variable is given by

$$\hat{Y}_0 \sim \mathcal{N}\left(\beta_0 + \beta_1 x_0, \sigma^2 \left(\frac{1}{n} + \frac{(x_0 - \bar{x})^2}{S_{xx}} \right) \right)$$

DEFINITION 1.3.2: Prediction error

The random variable for **prediction error** is defined as $Y_0 - \hat{Y}_0$ where Y_0 and $\hat{Y}_0$ are independent and $\hat{Y}_0$ is a function of $Y_1, \ldots, Y_n$.

$$\mathbb{E}[Y_0 - \hat{Y}_0] = \mathbb{E}[Y_0] - \mathbb{E}[\hat{Y}_0] = 0$$

$$\mathbb{V}(Y_0 - \hat{Y}_0) = \mathbb{V}(Y_0) + (-1)^2\mathbb{V}(\hat{Y}_0) = \sigma^2 + \sigma^2\left(\frac{1}{n} + \frac{(x_0 - \bar{x})^2}{S_{xx}}\right)$$

We proved the following theorem.

THEOREM 1.3.3: Distribution of Prediction Error

The distribution of the prediction error is given by

$$Y_0 - \hat{Y}_0 \sim \mathcal{N}\left(0, \sigma^2\left(1 + \frac{1}{n} + \frac{(x_0 - \bar{x})^2}{S_{xx}}\right)\right)$$

Since σ is unknown, we use $\hat{\sigma}$ and get the following:

$$\frac{Y_0 - \hat{Y}_0}{\hat{\sigma}\sqrt{1 + \frac{1}{n} + \frac{(x_0 - \bar{x})^2}{S_{xx}}}} \sim t(n-2)$$

Intuition for prediction error composed of 2 terms:

- $\mathbb{V}(Y_0)$: random error of new observation
- $\mathbb{V}(\hat{Y}_0)$ (predictor): estimating β_0 and β_1

Those are 2 sources of uncertainty.

REMARK 1.3.4

Be careful that the prediction may not make sense if x_0 is outside the range of the x_i's in the data.

A $(1-\alpha)$ prediction interval for the mean response $y_0 = \beta_0 + \beta_1 x_0$ at x_0 is

$$\hat{y}_0 \pm c\,\hat{\sigma}\sqrt{1 + \frac{1}{n} + \frac{(x_0 - \bar{x})^2}{S_{xx}}}$$

where c is the $1 - \dfrac{\alpha}{2}$ quantile of $t(n-2)$.

1.4 Application

EXAMPLE 1.4.1: Orange production 2018 in FL

We are given the following information.

- x: acres
- y: # boxes of oranges (thousands)
- (x_i, y_i) recorded for each of 25 FL counties
- $r = 0.964$
- $\bar{x} = 16133$
- $\bar{y} = 1798$
- $S_{xx} = 1.245 \times 10^{10}$
- $S_{xy} = 1.453 \times 10^{9}$

Now, $\hat{\beta}_1 = S_{xy}/S_{xx} = 0.1167$ has a positive slope, therefore x and y are positively correlated. The expected number of boxes produced is estimated to be about 117 higher per an additional acre.
Computing $\hat{\beta}_0 = \bar{y} - \hat{\beta}_1\bar{x} = -85.3$, we see that it is not meaningful to interpret, since it is the expected production if there were 0 acres (outside the range of x_i) as no county has $x = 0$.
Now suppose SS(Res) $= 1.31 \times 10^7$ the residuals are the differences between y_i and the fitted regression

line.

- $\hat{\sigma}^2 = \dfrac{\sum_{i=1}^{n} e_i^2}{n-2} = \dfrac{1.31 \times 10^7}{25-2} = 5.7 \times 10^5$
- $\text{Se}\left(\hat{\beta}_1\right) = \dfrac{\hat{\sigma}}{\sqrt{S_{xx}}} = 0.00676$
- To test H_0: $\beta_1 = 0$, calculate $t = (\hat{\beta}_1 - 0)/\text{Se}\left(\hat{\beta}_1\right) = 0.1167/0.00676 \approx 17.3$, then elect the 0.975 quantile (for demonstration purposes) of $t(23)$ which is 2.07.
- Note that 17.3 is very unlikely to see in $t(23)$.

Since $17.3 \gg 2.07$, we reject H_0 at $\alpha = 0.05$ level, and conclude there's a significant linear relationship between acres and oranges produced.
The 95% confidence interval for β_1 is given by $0.1167 \pm 2.07(0.00676)$, which does not contain 0.

$$p\text{-value} = P(|t_{23}| \geqslant 17.3) = 2P(t_{23} \geqslant 17.3) \approx 1.2 \times 10^{-14}$$

Predict the # of boxes in thousands produced if we had 10000 acres to grow oranges.

$$\hat{\beta}_0 + \hat{\beta}_1 x_0 = -85.3 + (0.1167)(10000) \approx 1082$$

The 95% prediction interval is given by

$$1082 \pm 2.07\sqrt{5.69 \times 10^5}\sqrt{1 + \frac{1}{25} + \frac{(6133)^2}{1.245 \times 10^{10}}} = [-512.0407, 2675.595]$$

REMARK 1.4.2

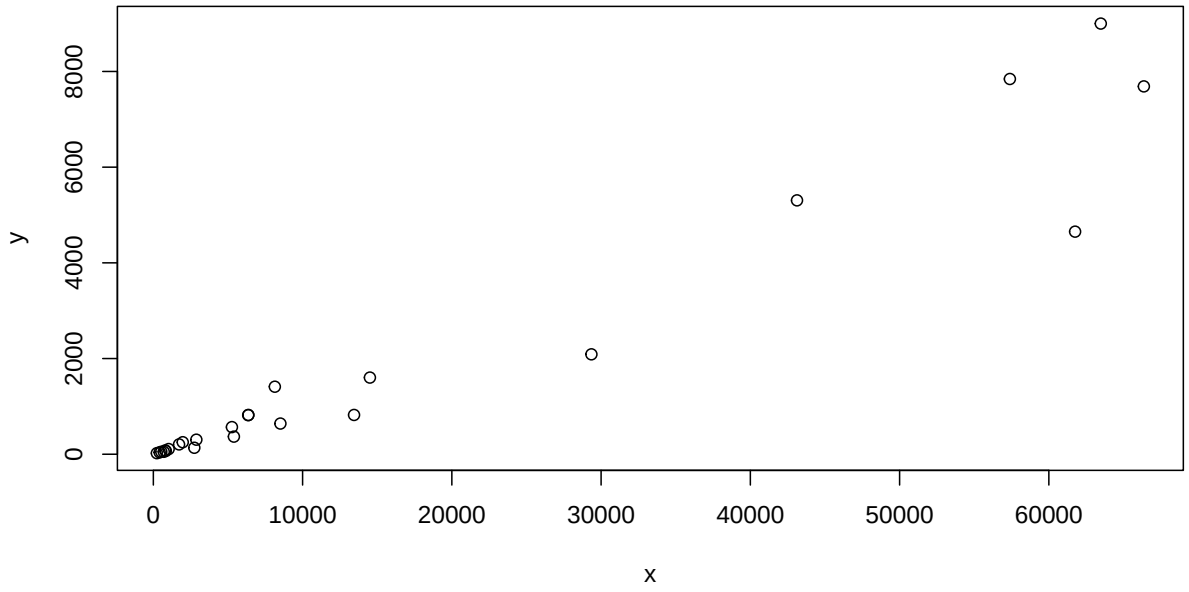
We are **not** trying to establish causation.

1.4.1 R Demo

```
# Read data from florange.csv and input it into the dat
# vector.
dat <- read.csv("csv/florange.csv")
# Done to make the predict function work well.
x <- dat$acres
y <- dat$boxes
# Output the first 6 rows in dat.
head(dat)
```

```
##      county boxes acres
## 1   Brevard    51   696
## 2 Charlotte   821 13447
## 3   Collier  2088 29351
## 4    DeSoto  7688 66365
## 5    Glades   368  5396
## 6    Hardee  5306 43126
```

```
# Draw a scatterplot with x-axis as 'acres' and y-axis as
# 'boxes'.
plot(x, y)
```




```r
# Compute some common variables with common functions.
r <- cor(x, y)
xbar <- mean(x)
ybar <- mean(y)
cat("r:", r, "xbar:", xbar, "ybar:", ybar)
```

```
## r: 0.9635098 xbar: 16132.64 ybar: 1797.56
```

Therefore, $r = 0.9635098$, $\bar{x} = 16132.64$, and $\bar{y} = 1797.56$.

```r
# Compute some common variables manually.
Sxx <- sum((x - xbar)^2)
Sxy <- sum((x - xbar) * (y - ybar))
cat("Sxx: ", Sxx, "Sxy: ", Sxy)
```

```
## Sxx:  12450023404 Sxy:  1453128337
```

Therefore, $S_{xx} = 12450023404 = 1.245 \times 10^{10}$ and $S_{xy} = 1453128337 = 1.453 \times 10^{9}$.

```r
# R's lm function fits linear models
lm.1 <- lm(y ~ x)
summary(lm.1)
```

```
##
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2470.81    -6.17    71.72   106.46  1677.32
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept) -85.391989 186.178031  -0.459    0.651
## x             0.116717   0.006761  17.263 1.16e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 754.4 on 23 degrees of freedom
## Multiple R-squared:  0.9284,Adjusted R-squared:  0.9252
## F-statistic:   298 on 1 and 23 DF,  p-value: 1.164e-14
```

From the summary, we can see that $\hat{\beta}_0 = -85.391989$, $\hat{\beta}_1 = 0.116717$, $\text{Se}\left(\hat{\beta}_1\right) = 0.006761$, $t = 17.263$, p-value $= 1.64 \times 10^{-14}$, and $\hat{\sigma} = 754.4$.

```
# Sum Squared Fitted Values
sum(lm.1$fitted.values^2)
```

```
## [1] 250385207
```

```
# Sum Squared Residuals
sum(lm.1$residuals^2)
```

```
## [1] 13089860
```

Therefore, $\text{SS(Res)} = \sum_{i=1}^{n} e_i^2 = 13089860 = 1.31 \times 10^7$.

```
# Manual calculation of sigma^2 estimate
sum(lm.1$residuals^2)/23
```

```
## [1] 569124.3
```

Therefore, $\hat{\sigma}^2 = 69124.3 = 5.7 \times 10^5$.

```
# Manual calculation of sigma estimate
sqrt(sum(lm.1$residuals^2)/23)
```

```
## [1] 754.4033
```

Therefore, $\hat{\sigma} = 754.4$.

```
# t distribution values
qt(0.975, 23)
```

```
## [1] 2.068658
```

Therefore, $c = 2.07$.

```
# 95% confidence interval
confint(lm.1)
```

```
##                    2.5 %      97.5 %
## (Intercept) -470.5305905 299.7466119
## x              0.1027305   0.1307034
```

```
# 95% prediction interval with predicted boxes if we had
# 10000 acres
predict(lm.1, data.frame(x = 10000), interval = "prediction")
```

```
##        fit       lwr      upr
## 1 1081.777 -512.0407 2675.595
```

Q: Is σ the same for all values of y?

A: It appears to not in the sense that the variance appears to be higher with respect to higher acres. Sigma will be smaller when there's less acres. Later, this will be testing equal variance or homoscedastic assumption.

Later, when we talk about variable transformations we can consider taking the logarithm.

Q: Are the error terms plausibly independent? In other words, does knowing one e_i (residual) help predict e_j (another residual) for a different county?

A: There's diagnostics for checking this. However, intuitively there could be some common factors at play when two counties are geographically close.

Chapter 2

Multiple Linear Regression

LECTURE 5 | 2020-09-21

2.1 Random Vectors

DEFINITION 2.1.1: Multiple linear regression

A **multiple linear regression** (MLR) model is defined as

$$Y = \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p + \varepsilon$$

which links a response variable y to several independent explanatory variables $x_1, x_2, \dots, x_p$.

EXAMPLE 2.1.2: Rocket MLR

- x_1: nozzle area (large or small, 0 or 1)
- x_2: mixture in propellant, ratio oxidized fuel
- Y: thrust

Want to develop linear relationship between response y and x_1, x_2; that is, we want to develop a linear relationship between thrust and both nozzle area and mixture in propellant.

In multiple linear regression, there are n observations, where each consists of p response variables (y_i), and p explanatory variables $(x_{i1}, x_{i2}, \dots, x_{ip})$. Then,

$$Y_i \sim N(\underbrace{\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip}}_{\mathbb{E}[Y_i]=\mu_i}, \sigma^2)$$

or $Y_i = \mu_i + \varepsilon_i$ where $\varepsilon_i \overset{\text{iid}}{\sim} \mathcal{N}(0, \sigma^2)$. We can write in vector/matrix form

$$\begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{bmatrix}_{n\times 1} = \begin{bmatrix} \beta_0 + \beta_1 x_{11} + \cdots + \beta_p x_{1p} \\ \beta_0 + \beta_1 x_{21} + \cdots + \beta_p x_{2p} \\ \vdots \\ \beta_0 + \beta_1 x_{n1} + \cdots + \beta_p x_{np} \end{bmatrix} + \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix}_{n\times 1}$$

Which we can more commonly write as $\boldsymbol{Y} = X\boldsymbol{\beta} + \boldsymbol{\varepsilon}$ where

$$\boldsymbol{Y} = \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{bmatrix}_{n\times 1} \quad X = \begin{bmatrix} 1 & x_{11} & x_{12} & \cdots & x_{1(p-1)} & x_{1p} \\ 1 & x_{21} & x_{22} & \cdots & x_{2(p-1)} & x_{2p} \\ \vdots & & & \ddots & & \vdots \\ 1 & x_{(n-1)1} & x_{(n-1)2} & \cdots & x_{(n-1)(p-1)} & x_{(n-1)p} \\ 1 & x_{n1} & x_{n2} & \cdots & x_{n(p-1)} & x_{np} \end{bmatrix}_{n\times(p+1)} \quad \boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \vdots \\ \beta_p \end{bmatrix}_{(p+1)\times 1} \quad \boldsymbol{\varepsilon} = \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix}_{n\times 1}$$

DEFINITION 2.1.3: Random vector

We call $\boldsymbol{Y} = (Y_1, Y_2, \dots, Y_n)^\top$ a **random vector**.

DEFINITION 2.1.4: Mean vector

The **mean vector** of $\boldsymbol{Y}$ is defined as $\mathbb{E}[\boldsymbol{Y}] = (\mathbb{E}[Y_1], \mathbb{E}[Y_2], \dots, \mathbb{E}[Y_n])^\top$.

DEFINITION 2.1.5: Covariance matrix

The **covariance matrix** (or **variance-covariance matrix**) of $\boldsymbol{Y}$ is defined as

$$\mathbb{V}(\boldsymbol{Y}) = \begin{bmatrix} \mathbb{V}(Y_1) & \mathrm{Cov}(Y_1, Y_2) & \cdots & \mathrm{Cov}(Y_1, Y_{n-1}) & \mathrm{Cov}(Y_1, Y_n) \\ \mathrm{Cov}(Y_2, Y_1) & \mathbb{V}(Y_2) & \cdots & \mathrm{Cov}(Y_2, Y_{n-1}) & \mathrm{Cov}(Y_2, Y_n) \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathrm{Cov}(Y_{n-1}, Y_1) & \mathrm{Cov}(Y_{n-1}, Y_2) & \cdots & \mathbb{V}(Y_{n-1}) & \mathrm{Cov}(Y_{n-1}, Y_n) \\ \mathrm{Cov}(Y_n, Y_1) & \mathrm{Cov}(Y_n, Y_2) & \cdots & \mathrm{Cov}(Y_n, Y_{n-1}) & \mathbb{V}(Y_n) \end{bmatrix}_{n\times n}$$

PROPOSITION 2.1.6: Properties of Covariance Matrix

Let $\boldsymbol{Y}$ be a random vector and $\boldsymbol{a} \in \mathbb{R}^n$, then the covariance matrix has the following properties.

(1) Symmetric since $Cov(Y_i, Y_j) = Cov(Y_j, Y_i)$; that is $\mathbb{V}(\boldsymbol{Y})^\top = \mathbb{V}(\boldsymbol{Y})$.

(2) Positive semi-definite since $\boldsymbol{a}^\top \mathbb{V}(\boldsymbol{Y})\boldsymbol{a} \geqslant 0$ for all $\boldsymbol{a} \in \mathbb{R}^n$.

(3) $\mathbb{V}(\boldsymbol{Y}) = \mathbb{E}\left[(\boldsymbol{Y} - \mathbb{E}[\boldsymbol{Y}])(\boldsymbol{Y} - \mathbb{E}[\boldsymbol{Y}])^\top\right]$

Proof of 2.1.6

Trivial.

PROPOSITION 2.1.7: Properties of Random Vector

Let $\boldsymbol{a}$ be a $1 \times n$ matrix (row vector) of constants and A be an $n \times n$ matrix of constants, then the random vector has the following properties.

(1) $\mathbb{E}[\boldsymbol{a}\boldsymbol{Y}] = \boldsymbol{a}\mathbb{E}[\boldsymbol{Y}]$

(2) $\mathbb{E}[A\boldsymbol{Y}] = A\mathbb{E}[\boldsymbol{Y}]$

(3) $\mathbb{V}(\boldsymbol{a}\boldsymbol{Y}) = \boldsymbol{a}\mathbb{V}(\boldsymbol{Y})\boldsymbol{a}^\top$

(4) $\mathbb{V}(A\boldsymbol{Y}) = A\mathbb{V}(\boldsymbol{Y})A^\top$

Proof of 2.1.7

We prove property (4) only.

$$\begin{aligned}
\mathbb{V}(A\boldsymbol{Y}) &= \mathbb{E}\Big[(A\boldsymbol{Y} - \mathbb{E}[A\boldsymbol{Y}])(A\boldsymbol{Y} - \mathbb{E}[A\boldsymbol{Y}])^\top\Big] \\
&= \mathbb{E}\Big[(A\boldsymbol{Y} - A\mathbb{E}[\boldsymbol{Y}])(A\boldsymbol{Y} - A\mathbb{E}[\boldsymbol{Y}])^\top\Big] \\
&= \mathbb{E}\Big[A(\boldsymbol{Y} - \mathbb{E}[\boldsymbol{Y}])(A(\boldsymbol{Y} - \mathbb{E}[\boldsymbol{Y}]))^\top\Big] \\
&= \mathbb{E}\Big[A(\boldsymbol{Y} - \mathbb{E}[\boldsymbol{Y}])(\boldsymbol{Y} - \mathbb{E}[\boldsymbol{Y}])^\top A^\top\Big] \\
&= A\mathbb{E}\Big[(\boldsymbol{Y} - \mathbb{E}[\boldsymbol{Y}])(\boldsymbol{Y} - \mathbb{E}[\boldsymbol{Y}])^\top\Big]A^\top \\
&= A\mathbb{V}(\boldsymbol{Y})A^\top
\end{aligned}$$

EXAMPLE 2.1.8: Calculations with MLR Varaibles

Let $\boldsymbol{Y} = (Y_1, Y_2, Y_3)^\top$. Suppose $\mathbb{E}[\boldsymbol{Y}] = (3, 1, 2)^\top$. Let $\mathbb{V}(\boldsymbol{Y}) = \begin{bmatrix} 4 & 1/2 & -2 \\ 1/2 & 1 & 0 \\ -2 & 0 & 3 \end{bmatrix}$ and $\boldsymbol{a} = (1, -1, 2)$ and $A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$. Note that $\boldsymbol{a}$ is a 1×3 row vector. Compute the following.

(i) $\mathbb{E}[\boldsymbol{aY}]$
(ii) $\mathbb{V}(\boldsymbol{aY})$
(iii) $\mathbb{E}[A\boldsymbol{Y}]$
(iv) $\mathbb{V}(A\boldsymbol{Y})$

Solution. We do the first two and leave the rest as an exercise.

(i) $\mathbb{E}[\boldsymbol{aY}] = \boldsymbol{a}\mathbb{E}[\boldsymbol{Y}] = \begin{bmatrix} 1 & -1 & 2 \end{bmatrix} \begin{bmatrix} 3 \\ 1 \\ 2 \end{bmatrix} = 1(3) - 1(1) + 2(2) = 6.$

(ii)

$$\begin{aligned}
\mathbb{V}(\boldsymbol{aY}) &= \boldsymbol{a}\mathbb{V}(\boldsymbol{Y})\boldsymbol{a}^\top \\
&= \begin{bmatrix} 1 & -1 & 2 \end{bmatrix} \begin{bmatrix} 4 & 1/2 & -2 \\ 1/2 & 1 & 0 \\ -2 & 0 & 3 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \\ 2 \end{bmatrix} \\
&= \begin{bmatrix} 1 & -1 & 2 \end{bmatrix} \begin{bmatrix} 4(1) + (1/2)(-1) - 2(2) \\ (1/2)(1) + 1(-1) + 0(2) \\ -2(1) + 0(-1) + 3(2) \end{bmatrix} \\
&= \begin{bmatrix} 1 & -1 & 2 \end{bmatrix} \begin{bmatrix} -1/2 \\ -1/2 \\ 4 \end{bmatrix} \\
&= 1(-1/2) - 1(-1/2) + 2(4) \\
&= 8
\end{aligned}$$

2.2 Multivariate Normal Distribution

DEFINITION 2.2.1: Multivariate normal distribution

Let $\boldsymbol{Y} = (Y_1, \dots, Y_n)^\top$ be a random vector. We say that $Y \sim \text{MVN}(\mu, \sigma)$; that is, Y follows a **multivariate normal distribution** (MVN) when

$$f(\boldsymbol{y}; \boldsymbol{\mu}, \Sigma) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} \exp\left\{-\frac{1}{2}(\boldsymbol{y}-\boldsymbol{\mu})^\top \Sigma^{-1}(\boldsymbol{y}-\boldsymbol{\mu})\right\}$$

where $\boldsymbol{\mu}$ is defined as the **mean vector**, and Σ is defined as the **covariance matrix**. Note that Σ^{-1} is the inverse of the covariance matrix and $|\Sigma|$ is the determinant of Σ.

THEOREM 2.2.2: Properties of Multivariate Normal Distribution

Let $\boldsymbol{Y} = (Y_1, Y_2, \dots, Y_n)^\top \sim MVN(\boldsymbol{\mu}, \Sigma)$.

(1) Any subset of $Y_1, Y_2, \dots, Y_n$ also has a multivariate normal distribution and in particular

$$Y_i \sim \mathcal{N}(\mu_i, \Sigma_{ii}) \qquad i = 1, 2, \dots, n$$

(2) Let $\boldsymbol{a} = (a_1, a_2 \dots, a_n)$ be a non-zero row vector ($1 \times n$) of constants, then

$$\boldsymbol{aY} \sim \mathcal{N}(\boldsymbol{a\mu}, \boldsymbol{a}\Sigma\boldsymbol{a}^\top)$$

(3) Let A be an $n \times n$ matrix of rank n, then

$$A\boldsymbol{Y} \sim MVN(A\boldsymbol{\mu}, A\Sigma A^\top)$$

(4) The conditional distribution of any subset of $(Y_1, Y_2, \dots, Y_n)$ given any other coordinates that are not in the subset is a multivariate normal distribution.

(5) Y_i and Y_j are independent random variables if and only if $\Sigma_{ij} = Cov(Y_i, Y_j) = 0$.

LECTURE 6 | 2020-09-23

2.3 Inference

Recall that last lecture, for multiple linear regression, we have $\boldsymbol{Y} = X\boldsymbol{B} + \boldsymbol{\varepsilon}$ with the assumption that $\varepsilon \overset{\text{iid}}{\sim} \mathcal{N}(0, \sigma^2)$. Therefore, for a random vector $\boldsymbol{\varepsilon}$, we have

$$\boldsymbol{\varepsilon} \sim \text{MVN}\left(\begin{bmatrix}0\\0\\\vdots\\0\\0\end{bmatrix}, \begin{bmatrix}\sigma^2 & 0 & \cdots & 0 & 0\\ 0 & \sigma^2 & \cdots & 0 & 0\\ \vdots & \vdots & \ddots & \vdots & \vdots\\ 0 & 0 & \cdots & \sigma^2 & 0\\ 0 & 0 & \cdots & 0 & \sigma^2\end{bmatrix}\right) = \text{MVN}(\boldsymbol{0}_{n\times 1}, \sigma^2 I_{n\times n})$$

since $\text{Cov}(\varepsilon_i, \varepsilon_j) = 0$ for all $i \neq j$ due to independence.

Thus, $\boldsymbol{Y} \sim \text{MVN}(X\boldsymbol{B}, \sigma^2 I)$.

DEFINITION 2.3.1: Least squares for MLR

We define the **least squares for a multiple linear regression model** as

$$S(\beta_0, \beta_1, \dots, \beta_p) = \sum_{i=1}^{n} (y_i - \underbrace{(\beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip})}_{\mathbb{E}[Y_i] = \mu_i})^2$$

THEOREM 2.3.2: Least Square Estimates (LSEs) for MLR

Minimizing $S(\beta_0, \beta_1, \dots, \beta_p)$, gives the least squares estimate $\hat{\boldsymbol{\beta}} = (X^\top X)^{-1} X^\top \boldsymbol{y}$.

Proof of 2.3.2

The first partial is $\frac{\partial S}{\partial \beta_0} = \sum_{i=1}^{n} 2(y_i - \mu_u)(-1)$, and all other partials for $j = 1, \dots, p$ are

$$\frac{\partial S}{\partial \beta_j} = \sum_{i=1}^{n} 2(y_i - \mu_i)(-x_{ij})$$

Set $\dfrac{\partial S}{\partial \beta_0} = 0$ and $\dfrac{\partial S}{\partial \beta_j} = 0$ for $j = 1, \dots, p$ to get

$$\begin{cases} \sum_{i=1}^{n}(y_i - \mu_i) = 0 \iff \mathbf{1}^\top(\boldsymbol{y} - \boldsymbol{\mu}) = 0 \\ \sum_{i=1}^{n}(y_i - \mu_i)x_{ij} = 0 \iff \boldsymbol{x}_j^\top(\boldsymbol{y} - \boldsymbol{\mu}) = 0 \quad j = 1, \dots, p \end{cases}$$

since we recall that

$$X = \begin{bmatrix} 1 & x_{11} & \cdots & x_{1(p-1)} & x_{1p} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & x_{n1} & \cdots & x_{n(p-1)} & x_{np} \end{bmatrix} = \begin{bmatrix} \mathbf{1} & \boldsymbol{x}_1 & \cdots & \boldsymbol{x}_{p-1} & \boldsymbol{x}_p \end{bmatrix}$$

Therefore,

$$X^\top(\boldsymbol{y} - X\boldsymbol{\beta}) = 0 \iff X^\top \boldsymbol{y} - X^\top X\boldsymbol{\beta} = 0 \iff X^\top X\boldsymbol{\beta} = X^\top \boldsymbol{y} \iff \boldsymbol{\beta} = (X^\top X)^{-1} X^\top \boldsymbol{y}$$

assuming $X^\top X$ is invertible; that is, $\text{rank}(X^\top X) = p + 1$. So, the LS solution for $\boldsymbol{\beta}$ is given by $\hat{\boldsymbol{\beta}} = (X^\top X)^{-1} X^\top \boldsymbol{y}$.

DEFINITION 2.3.3: Residuals for MLR

The **residuals** for a multiple linear regression model is defined as

$$e_i = y_i - (\underbrace{\hat{\beta}_0 + \hat{\beta}_1 x_{i1} + \cdots \hat{\beta}_p x_{ip}}_{\text{fitted value } \mu_i})$$

or equivalently, $\hat{\boldsymbol{\mu}} = X\hat{\boldsymbol{\beta}}$ and $\boldsymbol{e} = \boldsymbol{y} - \hat{\boldsymbol{\mu}}$.

The estimate σ^2 based on e_i's is

$$\hat{\sigma}^2 = \frac{\text{SS(Res)}}{n - (p+1)} = \frac{\sum_{i=1}^{n} e_i^2}{n - p - 1} = \frac{\boldsymbol{e}^\top \boldsymbol{e}}{n - p - 1}$$

since d.f. is $n -$ (number of estimated parameters). When viewed as a random variable,

$$\frac{(n - p - 1)\hat{\sigma}^2}{\sigma^2} \sim \chi^2(n - p - 1)$$

Inference for $\hat{\boldsymbol{\beta}} = (\hat{\beta}_0, \dots, \hat{\beta}_p)^\top = (X^\top X)^{-1} X^\top \boldsymbol{Y}$.

Note that $\hat{\boldsymbol{\beta}}$ is a matrix of constants and $\boldsymbol{Y}$ is a random vector, and $\boldsymbol{Y} \sim \text{MVN}(X\boldsymbol{\beta}, \sigma^2 I)$, so

$$\begin{aligned}\mathbb{E}[\hat{\boldsymbol{\beta}}] &= \mathbb{E}[(X^\top X)^{-1} X^\top \boldsymbol{Y}] \\ &= (X^\top X)^{-1} X^\top \mathbb{E}[\boldsymbol{Y}] \\ &= (X^\top X)^{-1}(X^\top X)\boldsymbol{\beta} \\ &= \boldsymbol{\beta}\end{aligned}$$

That is, $\mathbb{E}[\hat{\beta}_0], \dots, \mathbb{E}[\hat{\beta}_p] = \beta_p$ all unbiased.

$$\begin{aligned}\mathbb{V}((X^\top X)^{-1} X^\top \boldsymbol{Y}) &= (X^\top X)^{-1} X^\top \mathbb{V}(\boldsymbol{Y}) \left[(X^\top X)^{-1} X^\top\right]^\top \\ &= (X^\top X)^{-1} X^\top \sigma^2 I (X^\top)^\top \left[(X^\top X)^{-1}\right]^\top && X^\top X \text{ symmetric} \\ &= \sigma^2 (X^\top X)^{-1}(X^\top X)(X^\top X)^{-1}\end{aligned}$$

Since $\hat{\boldsymbol{\beta}}$ is a linear transformation of $\boldsymbol{Y}$ we have $\hat{\boldsymbol{\beta}} \sim \text{MVN}(\boldsymbol{\beta}, \sigma^2 \underbrace{(X^\top X)^{-1}}_{V})$. We proved the following theorem.

THEOREM 2.3.4: Distribution of $\hat{\beta}_j$

The distribution of a given $\hat{\beta}_j$ is

$$\hat{\beta}_j \sim \mathcal{N}(\beta_j, \sigma^2 V_{jj}) \qquad j = 0, 1, \dots, p$$

$$Z = \frac{\hat{\beta}_j - \beta_j}{\sigma\sqrt{V_{jj}}} \sim \mathcal{N}(0,1) \quad \text{and} \quad T = \frac{\hat{\beta}_j - \beta_j}{\hat{\sigma}\sqrt{V_{jj}}} \sim t(n-p-1) \qquad j = 0, 1, \dots, p$$

DEFINITION 2.3.5: Standard error for $\hat{\beta}_j$

We define the **standard error** of $\hat{\beta}_j$ as

$$\text{Se}\big(\hat{\beta}_j\big) = \hat{\sigma}\sqrt{V_{jj}} \qquad j = 0, 1, \dots, p$$

So, a $(1-\alpha)$ confidence interval for β_j is

$$\hat{\beta}_j \pm c\text{Se}\big(\hat{\beta}_j\big)$$

where c is $(1-(\alpha/2))$ quantile of $t(n-p-1)$.

To test H_0: $\beta_j = 0$ vs H_A: $\beta_j \neq 0$, calculate t-statistic $t = \dfrac{\hat{\beta}_j}{\text{Se}\big(\hat{\beta}_j\big)}$ reject at level α if $|t| > c$ and p-value is $2P(T \geqslant |t|)$ where $T \sim t(n-p-1)$.

Interpretation of $\hat{\boldsymbol{\beta}}$: fitted linear regression model says $\widehat{\mathbb{E}[Y]}$ (estimate of the expected response) is

$$\hat{\beta}_0 + \hat{\beta}_1 x_1 + \cdots + \hat{\beta}_p x_p$$

- $\hat{\beta}_0$ is the estimate of expected response when all explanatory variables are equal to 0.
- $\hat{\beta}_j$ is the estimated change in expected response for a unit increase in x_j when holding all other explanatory variables constant.

$$\hat{\beta}_0 + \hat{\beta}_1(x_1+1) + \cdots + \hat{\beta}_p x_p - (\hat{\beta}_0 + \hat{\beta}_1 x_1 + \cdots + \hat{\beta}_p x_p) = \hat{\beta}_1$$

REMARK 2.3.6

When it's written V_{jj}, that means the $j+1^{\text{th}}$ column and $j+1^{\text{th}}$ row since we start from index 0 for these matrices. Some unfortunate events may have happened on the quiz to me due to this.

2.4 Application

EXAMPLE 2.4.1: Rocket MLR

Let $n=12$, $\hat{\boldsymbol{\beta}} = (473.6, 16.7, -1.09)^\top = (\hat{\beta}_0, \hat{\beta}_1, \hat{\beta}_2)^\top$.

- x_1: nozzle area ($1 = L, 0 = S$)
- x_2: propellant ratio
- Y: thrust

$$\hat{\sigma} = \sqrt{\frac{\sum_{i=1}^{12} e_i^2}{12-1-2}} = \sqrt{\frac{\boldsymbol{e}^\top \boldsymbol{e}}{9}} = 2.655$$

Interpretation of $\hat{\boldsymbol{\beta}}$:

- $\hat{\beta}_1$ estimated change in expected thrust is 16.7 when changing small to large nozzle while holding other variables (propellant ratio) constant.
- $\hat{\beta}_2$ estimated thrust to decrease by 1.09 on average for a unit increase in propellant ratio while holding other variables (nozzle area) constant.

Given $\text{Se}\big(\hat{\beta}_2\big) = 0.94$, we compute the t-statistic for H_0: $\beta_2 = 0$ vs H_A: $\beta_2 \neq 0$ which is $t = -1.09/0.94 = -1.16$.

$$p\text{-value} = 2P(T \geqslant 1.16) = 0.275 \text{ from R where } T \sim t(9)$$

Do not reject H_0 (e.g., $\alpha = 0.05$), therefore propellant ratio does not significantly influence thrust.

LECTURE 7 | 2020-09-28

2.5 Prediction

Recall that $\boldsymbol{Y} = X\boldsymbol{\beta} + \boldsymbol{\varepsilon} \sim \text{MVN}(X\boldsymbol{\beta}, \sigma^2 I)$, and

- Estimates: $\hat{\boldsymbol{\beta}} = (X^\top X)^{-1} X^\top \boldsymbol{Y}$
- Fitted values: $\hat{\boldsymbol{\mu}} = X\hat{\boldsymbol{\beta}}$
- Residuals: $\boldsymbol{e} = \boldsymbol{y} - \hat{\boldsymbol{\mu}}$
- Constants: $X = \begin{bmatrix} \mathbf{1} & \boldsymbol{x}_1 & \cdots & \boldsymbol{x}_p \end{bmatrix}_{n \times (p+1)}$
- Values of responses: $\boldsymbol{y} = (y_1, y_2, \dots, y_n)^\top \in \mathbb{R}^n$

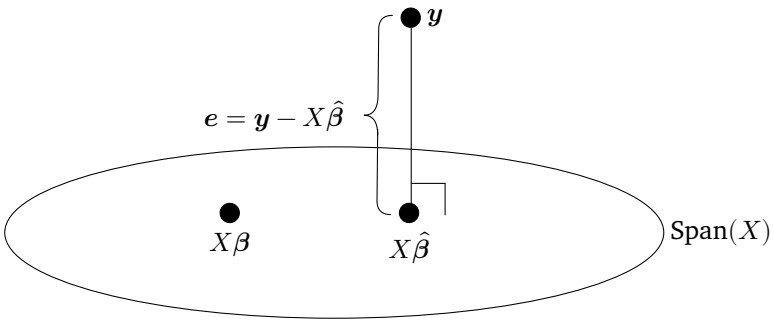
Author's Note: Geometric interpretation of data is omitted in these notes because I'm simply too lazy.

The span of X is $\text{Span}(X) = \{b_0\mathbf{1} + b_1\boldsymbol{x}_1 + \cdots + b_p\boldsymbol{x}_p : b_0, \dots, b_p \in \mathbb{R}\} \subset \mathbb{R}^n$ which is all linear combinations of columns of X which is a subspace of $\mathbb{R}^n$, and by assumption we know $\text{rank}(X) = p+1$.

We can say $\text{Span}(X)$ represents all possible vector values $X\boldsymbol{b}$ where $\boldsymbol{b} = (b_0, b_1, \dots, b_p)^\top$.

Generally, $\boldsymbol{y} \notin \text{Span}(X)$, so since the linear model is an approximation, $\boldsymbol{\varepsilon}$ variability not explained by model.

Intuitively, it makes sense to choose an estimate $\hat{\boldsymbol{\beta}}$ so that $X\hat{\boldsymbol{\beta}}$ is as close to $\boldsymbol{y}$ as possible.



Therefore, $\boldsymbol{e}$ must be orthogonal to Span(X) $\iff$ $\boldsymbol{e}$ is orthogonal to all columns of X.

$$\begin{aligned}\mathbf{1}^\top(\boldsymbol{y} - \hat{\boldsymbol{\mu}}) &= 0\\ \boldsymbol{x}_1^\top(\boldsymbol{y} - \hat{\boldsymbol{\mu}}) &= 0\\ &\vdots\\ \boldsymbol{x}_p^\top(\boldsymbol{y} - \hat{\boldsymbol{\mu}}) &= 0\end{aligned}$$

which is the same as LS estimates. We also know $\hat{\boldsymbol{\mu}} = X\hat{\boldsymbol{\beta}}$ and $\boldsymbol{e} = \boldsymbol{y} - \hat{\boldsymbol{\mu}}$.

DEFINITION 2.5.1: Hat matrix

The **hat matrix** is defined as $H = X(X^\top X)^{-1}X^\top$.

PROPOSITION 2.5.2: Properties of Hat Matrix

Let H be a hat matrix, then H has the following properties.
(1) H is symmetric; that is, $H = H^\top$.
(2) H is idempotent; that is, $H^2 = HH = H$.
(3) $I - H$ is symmetric idempotent; that is, $(I-H)^2 = (I-H)(I-H) = I - H$.

Proof of 2.5.2

We prove all three because it's easy.
(1) $H^\top = [X(X^\top X)^{-1}X^\top]^\top = X(X^\top X)^{-1}X^\top = H$.
(2) $HH = X(X^\top X)^{-1}(X^\top X)(X^\top X)^{-1}X^\top = H$.
(3) $(I-H)(I-H) = I(I-H) - H(I-H) = II - IH - HI + HH = I - 2H + HH = I - 2H + H = I - H$.

Let's view $\hat{\boldsymbol{\mu}}$ and $\boldsymbol{e}$ as random vectors

$$\hat{\boldsymbol{\mu}} = X\hat{\boldsymbol{\beta}} = X(X^\top X)^{-1}X^\top \boldsymbol{Y} = H\boldsymbol{Y}$$

$$\boldsymbol{e} = \boldsymbol{Y} - \hat{\boldsymbol{\mu}} = I\boldsymbol{Y} - H\boldsymbol{Y} = (I-H)\boldsymbol{Y}$$

$$\mathbb{E}[\hat{\boldsymbol{\mu}}] = \mathbb{E}[H\boldsymbol{Y}] = H\mathbb{E}[\boldsymbol{Y}] = X(X^\top X)^{-1}X^\top \underbrace{X\boldsymbol{\beta}}_{\mathbb{E}[\boldsymbol{Y}]} = X\boldsymbol{\beta}$$

$$\mathbb{V}(\hat{\boldsymbol{\mu}}) = \mathbb{V}(H\boldsymbol{Y}) = H\mathbb{V}(\boldsymbol{Y})H^\top = H\sigma^2 I H^\top = \sigma^2(HH^\top) = \sigma^2 H$$

$$\mathbb{E}[\boldsymbol{e}] = \mathbb{E}[(I-H)\boldsymbol{Y}] = \mathbb{E}[\boldsymbol{Y}] - \mathbb{E}[H\boldsymbol{Y}] = X\boldsymbol{\beta} - X\boldsymbol{\beta} = 0$$

$$\mathbb{V}(\boldsymbol{e}) = (I-H)\mathbb{V}(\boldsymbol{Y})(I-H)^\top = \sigma^2(I-H)(I-H)^\top = \sigma^2(I-H)$$

So since $\hat{\boldsymbol{\mu}}$ and $\boldsymbol{e}$ are linear transformations of $\boldsymbol{Y}$ we have proved the following theorem.

THEOREM 2.5.3: Distribution of $\hat{\mu}$ and e

$\hat{\boldsymbol{\mu}}$ and $\hat{\boldsymbol{e}}$ have the following distribution.

$$\hat{\boldsymbol{\mu}} \sim MVN(X\boldsymbol{\beta}, \sigma^2 H)$$

$$\hat{\boldsymbol{e}} \sim MVN(0, \sigma^2(I-H))$$

Suppose we want to predict response for $\boldsymbol{x}_0$ where the first 1 represents the intercept in the row vector.

$$\boldsymbol{x}_0 = \begin{bmatrix} 1 & x_{01} & x_{02} & \cdots & x_{0p} \end{bmatrix}_{1\times(p+1)}$$

Let Y_0 random variable representing the response associated with $\boldsymbol{x}_0$. In multiple linear regression,

$$Y_0 \sim \mathcal{N}(\beta_0 + \beta_1 x_{01} + \cdots + \beta_p x_{0p}, \sigma^2)$$

So we predict the value

$$\hat{y}_0 = \hat{\beta}_0 + \hat{\beta}_1 x_{01} + \cdots + \hat{\beta}_p x_{0p} = \boldsymbol{x}_0 \hat{\boldsymbol{\beta}}$$

which represents the estimated mean response given $x_{01}, x_{02}, \ldots, x_{0p}$. Corresponding distribution has

$$\mathbb{E}[\hat{Y}_0] = \boldsymbol{x}_0 \mathbb{E}[\hat{\boldsymbol{\beta}}] = \boldsymbol{x}_0 \boldsymbol{\beta} = \mathbb{E}[Y_0]$$

$$\mathbb{V}(\hat{Y}_0) = \boldsymbol{x}_0 \mathbb{V}(\hat{\boldsymbol{\beta}}) \boldsymbol{x}_0^\top = \boldsymbol{x}_0 \sigma^2 (X^\top X)^{-1} \boldsymbol{x}_0^\top$$

We have proved the following theorem.

THEOREM 2.5.4: Distribution of Predictor

The distribution of $\hat{Y}_0$ which is a function of $Y_1, \ldots, Y_n$ is

$$\hat{Y}_0 \sim \mathcal{N}(\boldsymbol{x}_0 \boldsymbol{\beta}, \sigma^2 \boldsymbol{x}_0 (X^\top X)^{-1} \boldsymbol{x}_0^\top)$$

$$\frac{\hat{Y}_0 - \boldsymbol{x}_0 \boldsymbol{\beta}}{\sigma \sqrt{\boldsymbol{x}_0 (X^\top X)^{-1} \boldsymbol{x}_0^\top}} \sim \mathcal{N}(0,1)$$

$$\frac{\hat{Y}_0 - \boldsymbol{x}_0 \boldsymbol{\beta}}{\hat{\sigma} \sqrt{\boldsymbol{x}_0 (X^\top X)^{-1} \boldsymbol{x}_0^\top}} \sim t(n-(p+1)) = t(n-p-1)$$

A $(1-\alpha)$ confidence interval for the mean response $y_0 = \boldsymbol{x}_0 \hat{\boldsymbol{\beta}}$ given $\boldsymbol{x}_0$ is

$$\hat{y}_0 \pm c\hat{\sigma} \sqrt{\boldsymbol{x}_0 (X^\top X)^{-1} \boldsymbol{x}_0^\top}$$

where c is the $1-\alpha/2$ quantile of $t(n-p-1)$.

Prediction error: $Y_0 - \hat{Y}_0$ which are independent since Y_0 is a random variable with variance σ^2 and $\hat{Y}_0$ is a function of $Y_1, \ldots, Y_n$. Therefore,

$$\mathbb{E}[Y_0 - \hat{Y}_0] = \boldsymbol{x}_0 \boldsymbol{\beta} - \boldsymbol{x}_0 \boldsymbol{\beta} = 0$$

$$\mathbb{V}(Y_0 - \hat{Y}_0) = \mathbb{V}(Y_0) + (-1)^2 \mathbb{V}(\hat{Y}_0) = \sigma^2 + \sigma^2 (\boldsymbol{x}_0 (X^\top X)^{-1} \boldsymbol{x}_0^\top)$$

We have proved the following theorem.

THEOREM 2.5.5: Distribution of Prediction Error

The distribution of the prediction error is

$$Y_0 - \hat{Y}_0 \sim N(0, \sigma^2(1 + \boldsymbol{x}_0(X^\top X)^{-1}\boldsymbol{x}_0^\top)).$$

A $(1-\alpha)$ prediction interval for the mean response $y_0 = \boldsymbol{x}_0\hat{\boldsymbol{\beta}}$ given $\boldsymbol{x}_0$ is

$$\hat{y}_0 \pm c\hat{\sigma}\sqrt{1 + \boldsymbol{x}_0(X^\top X)^{-1}\boldsymbol{x}_0^\top}$$

where c is the $1-\alpha/2$ quantile of $t(n-p-1)$.

REMARK 2.5.6

Our intuition tells us that the prediction interval is wider than the confidence interval for mean. In other words, estimating an average is "easier" than an individual response.

LECTURE 8 | 2020-09-30

2.6 Categorical Predictors

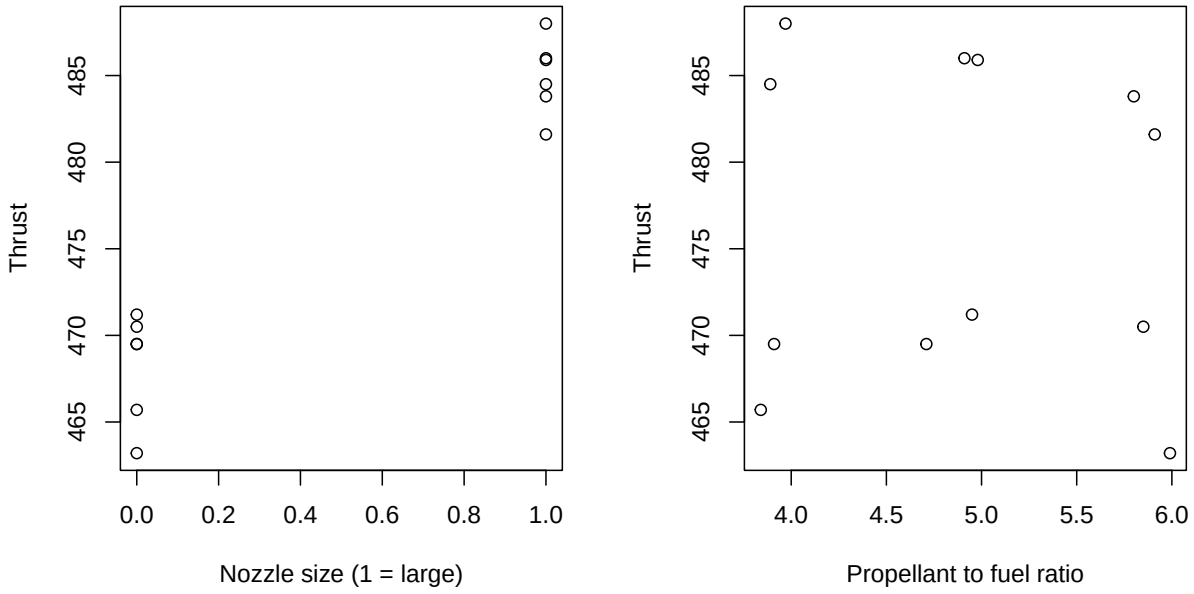
2.6.1 R Demo

```
## NASA rocket data example
## From: R.S. Jankovsky, T.D. Smith, A.J. Pavli (1999).
## 'High-Area-Ratio Rocket Nozzle at High Combustion
## Chamber Pressure-Experimental and Analytical
## Validation'.
# setwd(...) first if your CSV file is somewhere else
rocket <- read.csv("csv/rocket.csv")
# output all data in rocket vector
rocket
```

```
##    thrust nozzle propratio
## 1   488.0      1      3.97
## 2   481.6      1      5.91
## 3   485.9      1      4.98
## 4   486.0      1      4.91
## 5   484.5      1      3.89
## 6   483.8      1      5.80
## 7   463.2      0      5.99
## 8   471.2      0      4.95
## 9   469.5      0      3.91
## 10  470.5      0      5.85
## 11  469.5      0      4.71
## 12  465.7      0      3.84
```

Y (thrust) is the response variable, and there are two explanatory variables x_1, x_2 (nozzle, propratio) where nozzle is coded as 1 if it's large.

```
# Scatter plots where mfrow is used to put multiple plots
# on one image
par(mfrow = c(1, 2))
plot(rocket$nozzle, rocket$thrust, ylab = "Thrust", xlab = "Nozzle size (1 = large)")
plot(rocket$propratio, rocket$thrust, ylab = "Thrust", xlab = "Propellant to fuel ratio")
```



Left is nozzle size vs thrust. Right is propellant relationship vs thrust.

```
# Fit MLR using lm
m1 <- lm(thrust ~ nozzle + propratio, data = rocket)
summary(m1)
```

```
##
## Call:
## lm(formula = thrust ~ nozzle + propratio, data = rocket)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.8459 -1.7555  0.5934  1.2906  3.3008
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 473.6039     4.7158 100.430 4.88e-15 ***
## nozzle       16.7383     1.5329  10.919 1.71e-06 ***
## propratio    -1.0948     0.9414  -1.163    0.275
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.655 on 9 degrees of freedom
## Multiple R-squared:  0.9303,Adjusted R-squared:  0.9148
## F-statistic: 60.05 on 2 and 9 DF,  p-value: 6.238e-06
```

```
m2 <- lm(thrust ~ 0 + nozzle, data = rocket)
summary(m2)
```

```
##
## Call:
## lm(formula = thrust ~ 0 + nozzle, data = rocket)
```

```
##
## Residuals:
##    Min     1Q Median     3Q    Max
##  -3.37   0.58 233.12 469.50 471.20
##
## Coefficients:
##        Estimate Std. Error t value Pr(>|t|)
## nozzle    485.0      141.2   3.435  0.00558 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 345.8 on 11 degrees of freedom
## Multiple R-squared:  0.5175,Adjusted R-squared:  0.4736
## F-statistic:  11.8 on 1 and 11 DF,  p-value: 0.005575
```

```
anova(m1)
```

```
## Analysis of Variance Table
##
## Response: thrust
##           Df Sum Sq Mean Sq  F value    Pr(>F)
## nozzle     1 836.67  836.67 118.7377 1.743e-06 ***
## propratio  1   9.53    9.53   1.3524    0.2748
## Residuals  9  63.42    7.05
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

On the left it's Y (response variable) and on the right it's x_1, x_2 (explanatory variables). From summary, we get the estimate vector $\widehat{\beta} = (473.6039, 16.7383, -1.0948)^\top$.

```
# Manual beta estimates where rep is used to make the
# columns of 1s
X <- cbind(rep(1, 12), rocket$nozzle, rocket$propratio)  # X matrix
y <- matrix(rocket$thrust, ncol = 1)  # response vector
beta_hat <- solve(t(X) %*% X) %*% t(X) %*% y
beta_hat
```

```
##            [,1]
## [1,] 473.603924
## [2,]  16.738319
## [3,]  -1.094822
```

`solve` is used for the inverse. `%*%` is used for matrix-matrix multiplication, and `t(X)` is used for transposing X.

```
# Manual sigma estimate
mu_hat <- X %*% beta_hat  # fitted values
e <- y - mu_hat  # residuals
sigma_hat <- sqrt((t(e) %*% e)/9)  # Note n-p-1 = 12-2-1 = 9
sigma_hat
```

```
##        [,1]
## [1,] 2.6545
```

```
sigma_hat <- sqrt(sum(e^2)/9)  # equivalent
sigma_hat
```

```
## [1] 2.6545
```

- $\hat{\boldsymbol{\mu}} = X\hat{\boldsymbol{\beta}}$
- $\boldsymbol{e} = \boldsymbol{y} - \hat{\boldsymbol{\mu}}$
- $\hat{\sigma} = \sqrt{\left(\sum_{i=1}^{n} e_i^2\right)/9} = 2.6545$, or
- $\hat{\sigma} = \sqrt{(\boldsymbol{e}^\top \boldsymbol{e})/9} = 2.6545$

```
# Covariance matrix of beta_hat
vcov(m1)
```

```
##             (Intercept)      nozzle   propratio
## (Intercept)   22.238325 -1.02316688 -4.32080608
## nozzle        -1.023167  2.34987593 -0.03102117
## propratio     -4.320806 -0.03102117  0.88631920
```

```
sqrt(diag(vcov(m1)))  # SEs of individual betas
```

```
## (Intercept)      nozzle   propratio
##   4.7157528   1.5329305   0.9414453
```

```
# Manual
se_beta <- sigma_hat * sqrt(diag(solve(t(X) %*% X)))
se_beta
```

```
## [1] 4.7157528 1.5329305 0.9414453
```

- $\text{Se}\left(\hat{\boldsymbol{\beta}}\right) = \hat{\sigma}\sqrt{(X^\top X)^{-1}} = (4.71, 1.53, 0.94)^\top$

```
# Estimate the mean response for units with small nozzle
# and propellant ratio 5.5 include a 95% CI
predict(object = m1, newdata = data.frame(nozzle = 0, propratio = 5.5),
  interval = "confidence", level = 0.95)
```

```
##        fit      lwr      upr
## 1 467.5824 464.7929 470.3719
```

Therefore, $\hat{y}_0 = 467.58$. The 95% confidence interval for the mean response given $\boldsymbol{x}_0$ is $[464.7929, 470.3719]$.

```
# Manual calculation
x0 <- matrix(c(1, 0, 5.5), nrow = 1)
y0_hat <- x0 %*% beta_hat
y0_hat
```

```
##          [,1]
## [1,] 467.5824
```

```
# mu0 is also known as \hat{Y}_0
se_mu0 <- sigma_hat * sqrt(x0 %*% solve(t(X) %*% X) %*% t(x0))
se_mu0
```

```
##          [,1]
## [1,] 1.233132
```

```
crit_val <- qt(0.975, 9)
ci_lo <- y0_hat - crit_val * se_mu0
ci_hi <- y0_hat + crit_val * se_mu0
c(y0_hat, ci_lo, ci_hi)
```

```
## [1] 467.5824 464.7929 470.3719
```

- $\boldsymbol{x}_0 = \begin{bmatrix} 1 & 0 & 5.5 \end{bmatrix}$

- $\hat{y}_0 = \boldsymbol{x}_0\hat{\boldsymbol{\beta}} = 467.5824$
- $\text{Se}\left(\hat{Y}_0\right) = \hat{\sigma}\sqrt{\boldsymbol{x}_0(X^\top X)^{-1}\boldsymbol{x}_0^\top} = 1.233132$

Therefore, $\hat{y}_0 = 467.58$. The 95% confidence interval for the mean response given $\boldsymbol{x}_0$ is $[464.7929, 470.3719]$.

```
# Predict the value of the response for a unit with small
# nozzle and propellant ratio 5.5 include a 95% PI
predict(object = m1, newdata = data.frame(nozzle = 0, propratio = 5.5),
  interval = "prediction", level = 0.95)
```

```
##        fit      lwr      upr
## 1 467.5824 460.9612 474.2036
```

Therefore, $y_0 = 467.5824$. The 95% prediction interval for the response (y_0) given $\boldsymbol{x}_0$ is $[460.9612, 474.2036]$.

```
# Manual calculation for an individual
x0 <- matrix(c(1, 0, 5.5), nrow = 1)
y0_hat <- x0 %*% beta_hat
se_y0 <- sigma_hat * sqrt(1 + x0 %*% solve(t(X) %*% X) %*% t(x0))
se_y0
```

```
##          [,1]
## [1,] 2.926941
```

```
crit_val <- qt(0.975, 9)
pi_lo <- y0_hat - crit_val * se_y0
pi_hi <- y0_hat + crit_val * se_y0
c(y0_hat, pi_lo, pi_hi)
```

```
## [1] 467.5824 460.9612 474.2036
```

- $\text{Se}\left(Y_0 - \hat{Y}_0\right) = \hat{\sigma}\sqrt{1 + \boldsymbol{x}_0(X^\top X)^{-1}\boldsymbol{x}_0^\top} = 2.926941$

Handling categorical variables: when there are explanatory variables with values that fall into one of several categories.

- e.g., nozzle large/small, if just binary, code as 1 and 0
- ordered small, medium, large or not red, blue, green

Approach: can convert to indicator variables or treat as numerical if it makes sense to do so.

Example: Coffee Quality Institute (2018)

Extract a few variables:

	Acidity	Method
1	8.7	Washed-wet
2	8.3	Washed-wet
3	8.2	Natural-dry
4	8.4	Semi-washed/pulped

Flavour (response)

How to set up X? For example,

$$x_{i2} = \begin{cases} 0 & \text{dry} \\ 1 & \text{semi} \\ 2 & \text{wet} \end{cases}$$

Not generally appropriate unless we think a response is linear according to this scheme.

More flexible approach: indicator/dummy variables

$$x_{i2} = \begin{cases} 1 & \text{semi} \\ 0 & \text{otherwise} \end{cases}, \quad x_{i3} = \begin{cases} 1 & \text{wet} \\ 0 & \text{otherwise} \end{cases}$$

Therefore,

$$X = \begin{bmatrix} 1 & 8.7 & 0 & 1 \\ 1 & 8.3 & 0 & 1 \\ 1 & 8.2 & 0 & 0 \\ 1 & 8.4 & 1 & 0 \end{bmatrix}$$

Why not $x_{i4} = \begin{cases} 1 & \text{dry} \\ 0 & \text{otherwise} \end{cases}$? If we did that, we would have

$$X = \begin{bmatrix} 1 & 8.7 & 0 & 1 & 0 \\ 1 & 8.3 & 0 & 1 & 0 \\ 1 & 8.2 & 0 & 0 & 1 \\ 1 & 8.4 & 1 & 0 & 0 \end{bmatrix}$$

This has linearly dependent columns since $\boldsymbol{x}_4 = \mathbf{1} - \boldsymbol{x}_2 - \boldsymbol{x}_3$. There is no new information and X would not have full rank.

Model: $Y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \beta_3 x_{i3} + \varepsilon_i$.

Interpretation:

- Mean flavour if acidity $= x_{01}$ and method dry is $\beta_0 + \beta_1 x_{01}$.
- Mean flavour if acidity $= x_{01}$ and method wet is $\beta_0 + \beta_1 x_{01} + \beta_3$.
- Mean flavour if acidity $= x_{01}$ and method semi is $\beta_0 + \beta_1 x_{01} + \beta_2$.
- β_2 is the difference between semi and dry in expected response (holding acidity constant)
- β_3 is the difference between wet and dry in expected response (holding acidity constant)
- $\beta_2 - \beta_3$ is the difference between semi and wet (holding other variables constant)

$\hat{\boldsymbol{\beta}} \sim \text{MVN}(\boldsymbol{\beta}, \sigma^2 V)$ where $V = (X^\top X)^{-1}$.

- We know $\hat{\beta}_j \sim \mathcal{N}(\beta_j, \sigma^2 V_{jj})$ with $\text{Se}\big(\hat{\beta}_j\big) = \hat{\sigma}\sqrt{V_{jj}}$ where $j = 0, \dots, p$.
- What about $\beta_2 - \beta_3$?

  $$\mathbb{V}(\hat{\beta}_2 - \hat{\beta}_3) = \mathbb{V}(\hat{\beta}_2) - \mathbb{V}(\hat{\beta}_3) - 2\text{Cov}(\hat{\beta}_2, \hat{\beta}_3) = \sigma^2 V_{22} + \sigma^2 V_{33} - 2\sigma^2 V_{23}$$

  Therefore,

  $$\text{Se}\big(\hat{\beta}_2 - \hat{\beta}_3\big) = \hat{\sigma}\sqrt{V_{22} + V_{33} - 2V_{23}}$$

  Now, we can construct a CI for $\beta_2 - \beta_3$.

In general, for an explanatory variable with k categories. We need $k - 1$ indicator variables.

LECTURE 9 | 2020-10-05

2.7 Analysis of Variance

Analysis of variance (ANOVA): how well does our regression model fit our response variable?

Variability in response can be measured by "total sum of squares:"

$$\text{SS(Total)} = \sum_{i=1}^{n}(y_i - \bar{y})^2$$

as seen in A1, it's closely related to sample variance of $y_1, \dots, y_n$, which is $\text{SS(Total)}/(n-1)$.

ANOVA decomposes $\text{SS(Total)} = \text{SS(Reg)} + \text{SS(Res)}$ where SS(Reg) is the regression sum of squares and SS(Res) is the residual sum of squares.

The regression sum of squares is variation explained by the model and the residual sum of squares is the variation not explained by the regression model.

Using the fact that

$$y_i - \bar{y} = y_i - \hat{\mu}_i + \hat{\mu}_i - \bar{y}$$

When regression fits data well, the observations y_i tend to be much closer to $\hat{\mu}_i$. Note that $\bar{y}$ is line a regression line with $\beta_1 = 0$.

Mathematically,

$$\underbrace{\sum_{i=1}^{n}(y_i - \bar{y})^2}_{\text{SS(Total)}} = \underbrace{\sum_{i=1}^{n}(\hat{\mu}_i - \bar{y})^2}_{\text{SS(Reg)}} + \underbrace{\sum_{i=1}^{n}(y_i - \hat{\mu}_i)^2}_{\text{SS(Res)}}$$

since we showed that $\sum_{i=1}^{n}(\hat{\mu}_i - \bar{y})\underbrace{(y_i - \hat{\mu}_i)}_{e_i} = 0$ in A1 for SLR. It's also true for multiple linear regression since

$$\sum_{i=1}^{n}(\hat{\mu}_i - \bar{y})e_i = \sum_{i=1}^{n}(e_i\hat{\mu}_i) - \bar{y}\sum_{i=1}^{n}e_i = \hat{\boldsymbol{\mu}}^\top \boldsymbol{e} - \bar{y}\mathbf{1}^\top \boldsymbol{e} = 0$$

Recall: $\mathbf{1}^\top \boldsymbol{e} = 0$ is one of LS equations, and $\hat{\boldsymbol{\mu}} = X\hat{\boldsymbol{\beta}}$ is in $\text{Span}(X)$, so $\boldsymbol{e}$ is orthogonal to $\text{Span}(X)$, so $\hat{\boldsymbol{\mu}}^\top \boldsymbol{e} = 0$.

Table 2.1: ANOVA Table

Source	d.f.	SS	Mean Square	F
Regression	p	SS(Reg)	SS(Reg)$/p$	MS(Reg)/MS(Res)
Residual	$n-p-1$	SS(Res)	SS(Res)$/(n-p-1) = \hat{\sigma}^2$	
Total	$n-1$	SS(Total)		

F is used to test the overall significance of regression (later).

2.8 Coefficient of Determination

DEFINITION 2.8.1: Coefficient of Determination

We define the **coefficient of determination** as

$$R^2 = \frac{\text{SS(Reg)}}{\text{SS(Total)}} = 1 - \frac{\text{SS(Res)}}{\text{SS(Total)}}$$

Note that $0 \leqslant R^2 \leqslant 1$.

R^2 is the proportion of variation (in our response variable) that is explained by the regression model. Larger R^2 means the fitted values are closer to the observations y_i, which means the residuals are small; that is, smaller SS(Res). Note that (A1) in SLR, R^2 is equivalent to the square of the sample correlation between x and y based on $(x_1, y_1), \ldots, (x_n, y_n)$.

Table 2.2: Rocket ANOVA Table

Source	d.f.	SS	Mean Square	F
Regression	2	846.2	423.1	60
Residual	9	63.42	7.05	
Total	11	909.62		

Response thrust $R^2 = 846.2/909.62 \approx 0.93$. R^2 interpretation: regression model with nozzle size and propellant ratio explains 93% of variation in thrust (response).

LECTURE 10 | 2020-10-07

2.9 General Linear Hypothesis Tests Based on F Distribution

So far we've tested H_0: $\beta_j = 0$ vs H_A: $\beta_j \neq 0$ involving individual parameters, using t distribution.

Now consider hypothesis test of the form H_0: $A\boldsymbol{\beta} = \mathbf{0}$ where A is a matrix of constraints specifying linear combinations of parameters.

EXAMPLE 2.9.1: Coffee Continued

The full model is:

$$Y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \beta_3 x_{i3} + \varepsilon_i$$

- Y_i is the flavour
- x_{i1} is acidity
- x_{i2} is 1 if semi, and 0 otherwise.
- x_{i3} is 1 if wet, and 0 otherwise.

Example 1.

- H_0: $\beta_1 = \beta_2 = \beta_3 = 0$ versus
- H_A: at least one of $\beta_1, \beta_2, \beta_3$ not 0.
- If H_0 is true, the model reduces to $Y_i = \beta_0 + \varepsilon_i$.
- This tests overall significance of regression (whether any of predictors impact response)
- $A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$. Note that row i considers the constraint of $\beta_i = 0$ for $i = 1, 2, 3$ in this example.

Example 2.

- H_0: $\beta_2 = \beta_3 = 0$
- If H_0 is true, $Y_i = \beta_0 + \beta_1 x_{i1} + \varepsilon_i$
- Q: Is reduced model with only acidity plausible?
- $A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$. Note that $A\boldsymbol{\beta} = \mathbf{0}_{1\times 2}$

Example 3.

- H_0: $\beta_2 - \beta_3 = 0$
- H_A: $\beta_2 \neq \beta_3$
- $Y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 (x_{i2} + x_{i3}) + \varepsilon_i$ where $(x_{i2} + x_{i3})$ is 1 if semi/wet and 0 if dry.
- Do the wet and semi methods have the same impact on the response (holding acidity constant)?

- $A = \begin{bmatrix} 0 & 0 & 1 & -1 \end{bmatrix}$

In general, with ℓ constraints. A is an $\ell \times (p+1)$ matrix with rank ℓ. Recall that

$$\text{Span}(X) = \{\beta_0 \mathbf{1} + \beta_1 \boldsymbol{x}_1 + \cdots + \beta_p \boldsymbol{x}_p\}$$

Let

$$\text{Span}(X)_A = \{\beta_0 \mathbf{1} + \beta_1 \boldsymbol{x}_1 + \cdots + \beta_p \boldsymbol{x}_p : A\boldsymbol{\beta} = \mathbf{0}\}$$

which is a subspace of $\text{Span}(X)$ since any vector in $\text{Span}(X)_A$ is also in $\text{Span}(X)$. We call $\text{Span}(X)_A$ the $\text{Span}(X)$ with constraint A on $\boldsymbol{\beta}$.

Let $\hat{\boldsymbol{\mu}}_A$ denote the fitted values from fitting the reduced model. The residual if we fit the model with $A\boldsymbol{\beta} = \mathbf{0}$ is $\boldsymbol{e}_A = \boldsymbol{y} - \hat{\boldsymbol{\mu}}_A$.

If H_0: $A\boldsymbol{\beta} = \mathbf{0}$ is true, then $\hat{\boldsymbol{\mu}}$ and $\hat{\boldsymbol{\mu}}_A$ should be close; that is, the model makes similar predictions whether we set $A\boldsymbol{\beta} = \mathbf{0}$ or not when fitting the model.

So to assess whether H_0 is plausible, look at $\|\hat{\boldsymbol{\mu}} - \hat{\boldsymbol{\mu}}_A\|$ where $\|\cdot\|$ is Euclidean or L_2 norm. That is,

$$\|\hat{\boldsymbol{\mu}} - \hat{\boldsymbol{\mu}}_A\| = \sqrt{(\hat{\boldsymbol{\mu}} - \hat{\boldsymbol{\mu}}_A)^\top (\hat{\boldsymbol{\mu}} - \hat{\boldsymbol{\mu}}_A)}$$

If it's "large" or "small" (close to 0) where large gives evidence against H_0 and small gives evidence for H_0.

By Pythagoras,

$$\|\boldsymbol{y} - \hat{\boldsymbol{\mu}}_A\|^2 = \|\boldsymbol{y} - \hat{\boldsymbol{\mu}}\|^2 + \|\hat{\boldsymbol{\mu}} - \hat{\boldsymbol{\mu}}_A\|^2 \quad \text{or} \quad \|\boldsymbol{e}_A\|^2 = \|\boldsymbol{e}\|^2 + \|\hat{\boldsymbol{\mu}} - \hat{\boldsymbol{\mu}}_A\|^2$$

or equivalently $\boldsymbol{e}_A^\top \boldsymbol{e}_A = \boldsymbol{e}^\top \boldsymbol{e} + \|\hat{\boldsymbol{\mu}} - \hat{\boldsymbol{\mu}}_A\|^2$ where $\boldsymbol{e}_A^\top \boldsymbol{e}_A$ is the sum of squares residual in the reduced model and $\boldsymbol{e}^\top \boldsymbol{e}$ is the sum of squares residual in the full model.

We define $\boldsymbol{e}_A^\top \boldsymbol{e}_A = \text{SS(Res)}_A$ and $\boldsymbol{e}^\top \boldsymbol{e} = \text{SS(Res)}$.

Thus, $\|\hat{\boldsymbol{\mu}} - \hat{\boldsymbol{\mu}}_A\|^2 = \text{SS(Res)}_A - \text{SS(Res)} \geqslant 0$ additional sum of squares explained by full model vs reduced one with constraints A.

Practical implications:

- SS(Res) cannot decrease when constraints applied.
- Equivalently, full model always has small (or equal) SS(Res) for a fixed SS(Tot) and thus higher R^2 compared to a reduced model.

Define test statistic:

$$F = \frac{(\text{SS(Res)}_A - \text{SS(Res)})/\ell}{\text{SS(Res)}/(n-p-1)} = \frac{(\text{SS(Res)}_A - \text{SS(Res)})/\ell}{\hat{\sigma}^2}$$

DEFINITION 2.9.2: F distribtion

If $U \sim \chi^2(a)$ and $V \sim \chi^2(b)$ are independent. We say F follows an F **distribution** if

$$F = \frac{U/a}{V/b}$$

and write $F \sim F(a, b)$.

Here, we have these facts when H_0 is true

$$V = \frac{\hat{\sigma}^2(n-p-1)}{\sigma^2} \sim \chi^2(n-p-1)$$

$$U = \frac{\|\hat{\boldsymbol{\mu}} - \hat{\boldsymbol{\mu}}_A\|^2}{\sigma^2} \sim \chi^2(\ell)$$

where U and V are independent. Therefore,

$$F = \frac{\frac{\|\hat{\boldsymbol{\mu}} - \hat{\boldsymbol{\mu}}_A\|^2}{\sigma^2}\left(\frac{1}{\ell}\right)}{\frac{\hat{\sigma}^2(n-p-1)}{\sigma^2}\left(\frac{1}{n-p-1}\right)} \sim F(\ell, n-p-1)$$

when H_0 is true. Reject H_0: $A\boldsymbol{\beta} = \mathbf{0}$ at level α if F is greater than $(1-\alpha)$ quantile of $F(\ell, n-p-1)$ and p-value is $P(Y \geqslant F)$ where $Y \sim F(\ell, n-p-1)$.

Relation to T distribution: Say $Y \sim t(a)$

$$Y = \frac{Z}{\sqrt{U/a}}$$

where $Z \sim \mathcal{N}(0,1)$ and $U \sim \chi^2(a)$ are independent. Squaring everything,

$$Y^2 = \frac{Z^2}{U/a}$$

and we know $Z^2 \sim \chi^2(1)$. Therefore, $Y^2 \sim F(1, a)$ (we divide by 1 in the numerator).

Thus, if our hypothesis test has one constraint, then F test is equal to t test of same hypothesis; for example, H_0: $\beta_1 = 0$ versus H_A: $\beta_1 \neq 0$.

LECTURE 11 | 2020-10-19

2.10 ANOVA F Test

Recall the general linear hypothesis: H_0: $A\boldsymbol{\beta} = \mathbf{0}$ vs H_A: $A\boldsymbol{\beta} \neq \mathbf{0}$ where A gives ℓ constraints.

$$F \text{ statistic} = \frac{(\text{SS(Res)}_A - \text{SS(Res)})/\ell}{\text{SS(Res)}/(n-p-1)} = \frac{(\text{SS(Res)}_A - \text{SS(Res)})/\ell}{\hat{\sigma}^2}$$

compare to $F(\ell, n-p-1)$.

Special case: overall test of significance

"Are any predictors related to response?"

- H_0: $\beta_1 = \beta_2 = \cdots = \beta_p = 0$
- H_A: $\beta_j \neq 0$ for at least one j

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & 0 & \cdots & 0 & 0 \\ \vdots & & & \ddots & & & \\ 0 & 0 & 0 & 0 & \cdots & 1 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 1 \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_{p-1} \\ \beta_p \end{bmatrix}$$

If H_0 is true: $Y_i = \beta_0 + \varepsilon_i$ where $Y_i \sim \mathcal{N}(\beta_0, \sigma^2)$.

Fit reduced model; that is, in this case estimate β_0 using the least squares, minimize $\sum_{i=1}^n (y_i - \beta_0)^2$, which can be shown $\hat{\beta}_0 = \bar{y}$. So,

$$\text{SS(Res)}_A = \sum_{i=1}^n e_i^2 = \sum_{i=1}^n (y_i - \hat{\mu}_i)^2 = \sum_{i=1}^n (y_i - \bar{y})^2 = \text{SS(Total)}$$

Then,

$$F = \frac{(\text{SS(Total)} - \text{SS(Res)})/p}{\text{SS(Res)}/(n-p-1)} = \frac{\text{SS(Reg)}/p}{\text{SS(Res)}/(n-p-1)} = \frac{\text{MS(Reg)}}{\text{MS(Res)}} \leftarrow F \text{ statistic on ANOVA table}$$

2.10.1 R Demo

```
## R demo for Oct 19 Plotting functions and histograms, F
## distribution, ANOVA tables, F tests, MLR with
## categorical variables
```

Evaluate the function at many x values, then plot it.

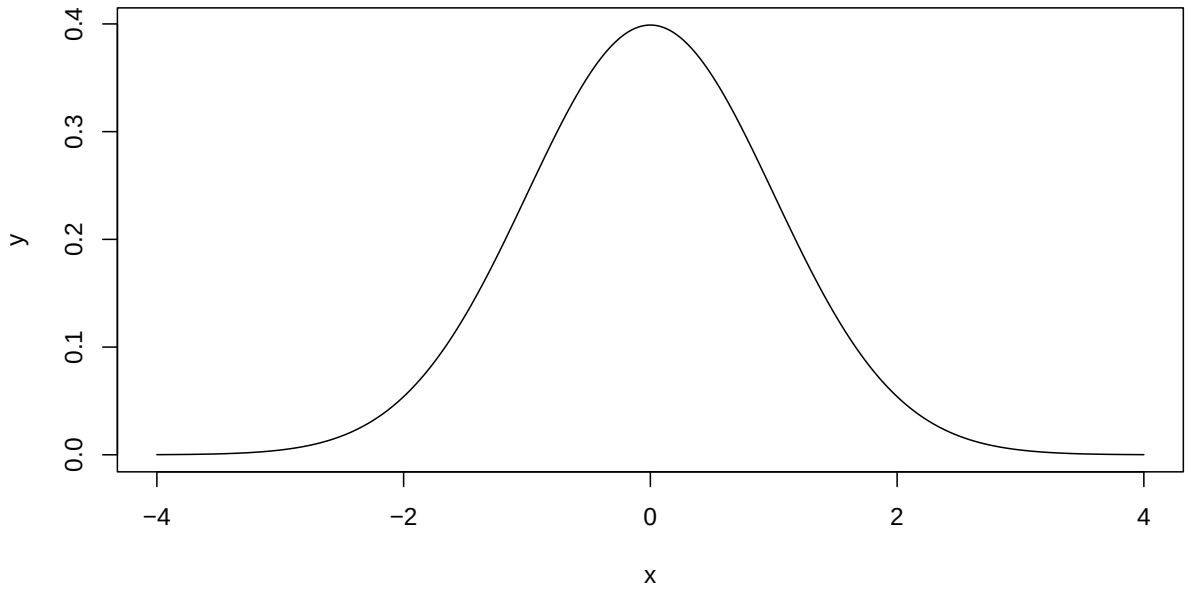
```
# Plotting functions (e.g., probability density functions)
# Create sequence from -4 to 4 increasing 0.01 each time.
x <- seq(-4, 4, 0.01)
head(x)
```

```
## [1] -4.00 -3.99 -3.98 -3.97 -3.96 -3.95
```

```
# Normal probability density function with mean 0, and
# standard deviation 1.
y <- dnorm(x, 0, 1)
```

`dnorm` is for density normal.

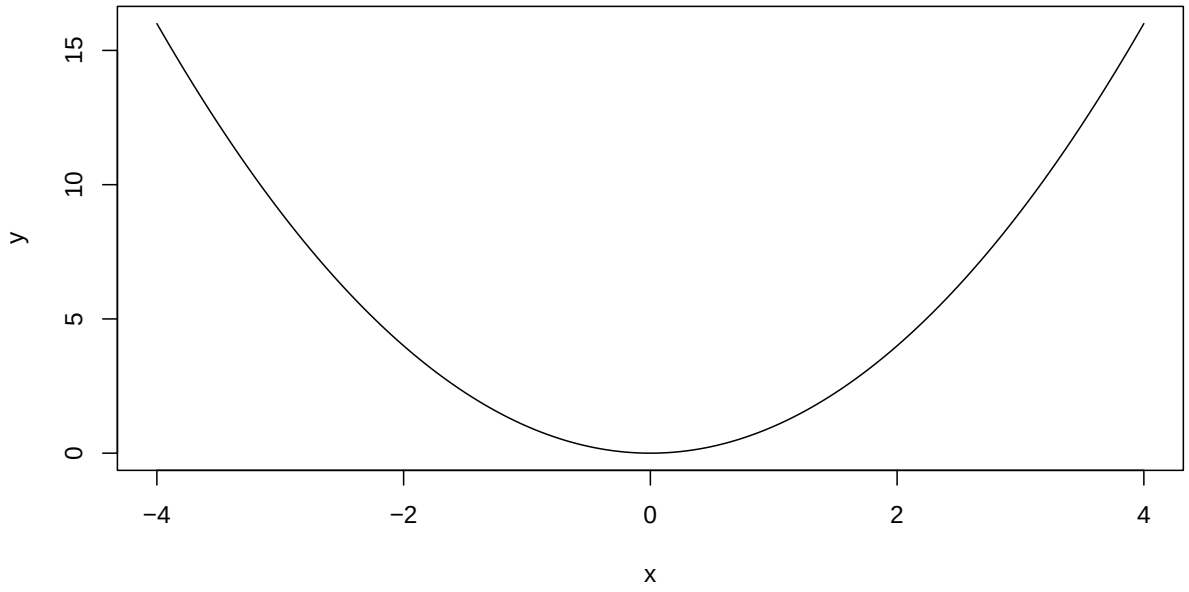
```
plot(x, y, type = "l")
```



`type = "l"` is for a smooth line (instead of dots).

We can also plot $y = x^2$ for example.

```
y <- x^2
plot(x, y, type = "l")
```

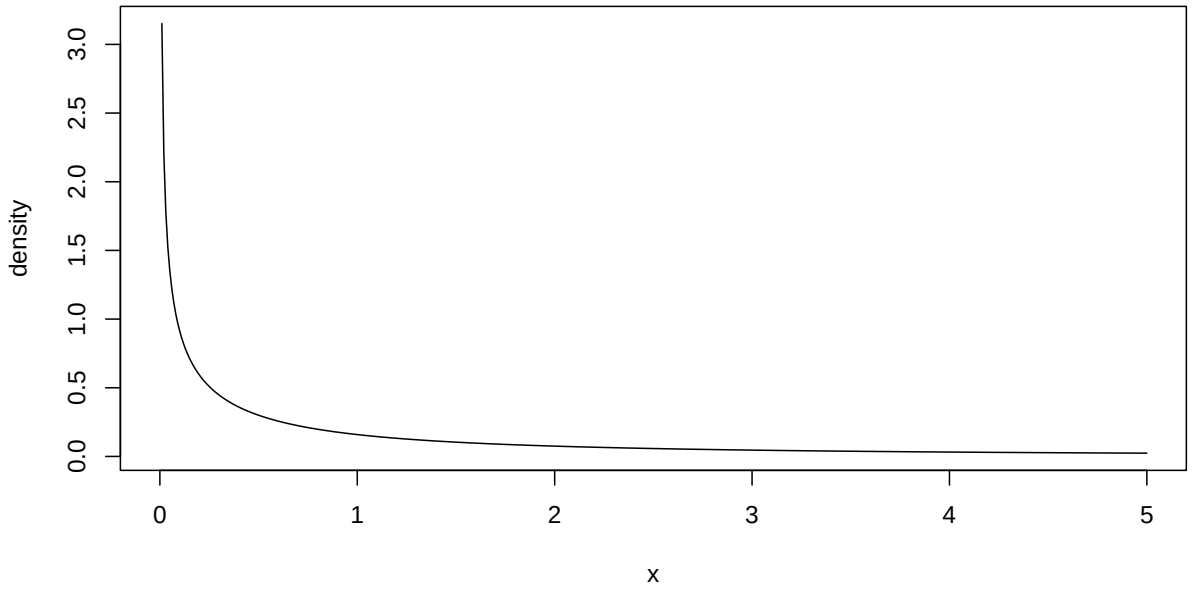


2.10.1.1 F-distribution Examples

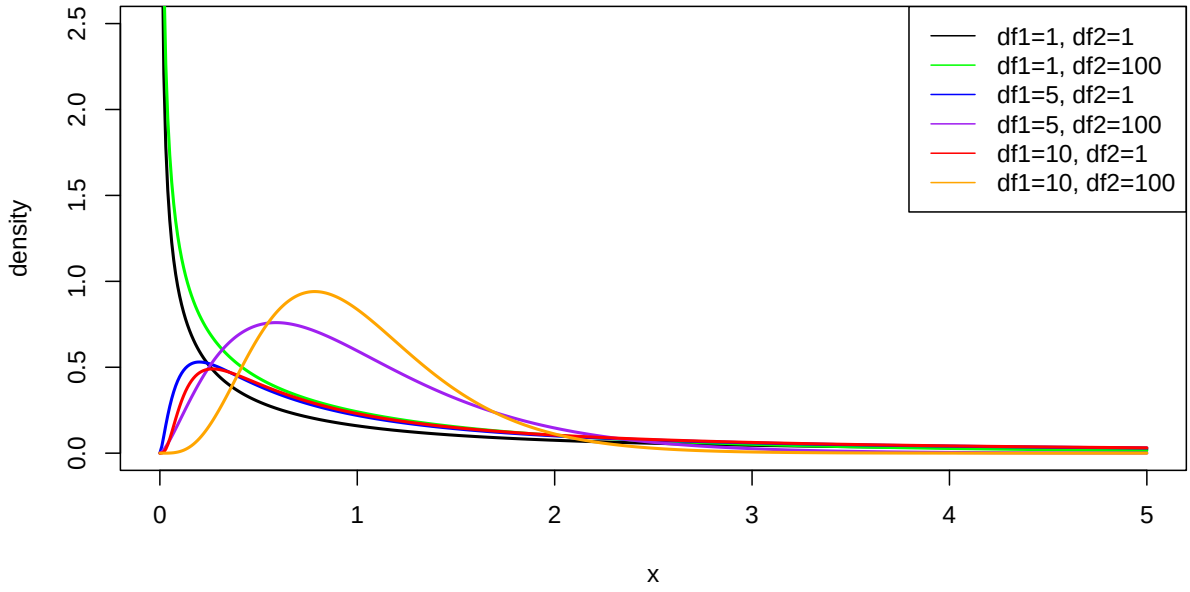
```
x <- seq(0, 5, 0.01)
head(x)
```

```
## [1] 0.00 0.01 0.02 0.03 0.04 0.05
```

```
# df is degrees of freedom.  type = 'l' is for a smooth
# curve
plot(x, y = df(x, df1 = 1, df2 = 1), type = "l", xlab = "x",
  ylab = "density")
```

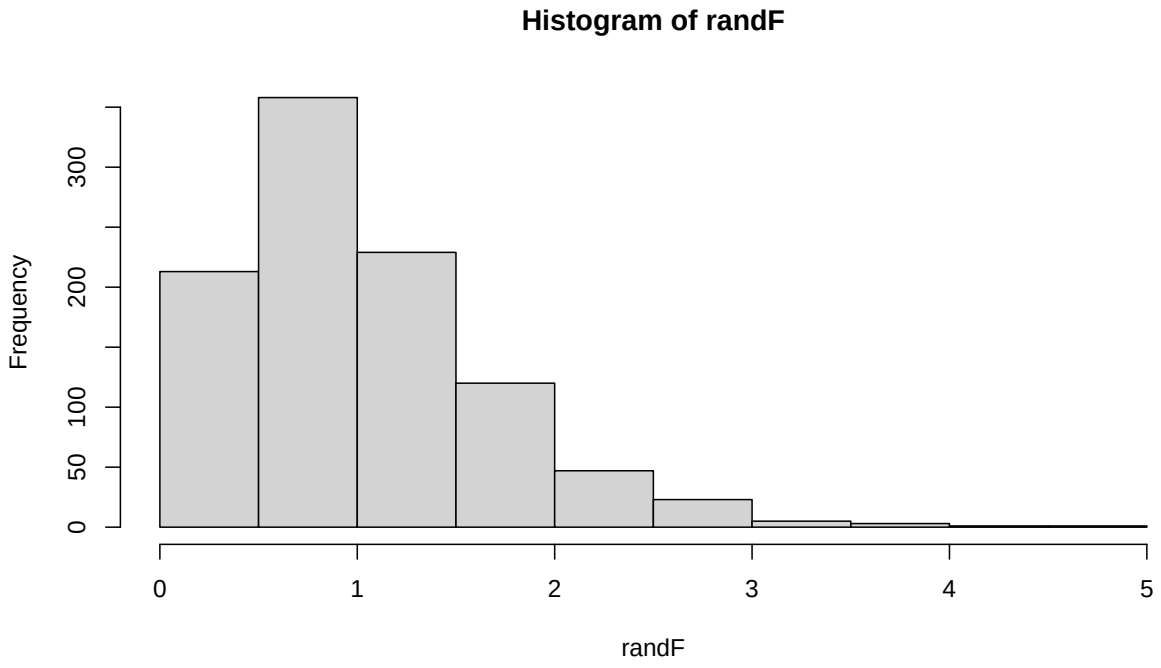



```r
# ylim is for the y-axis limits lwd is for line width
plot(x, y = df(x, df1 = 1, df2 = 1), type = "l", col = "black",
  xlab = "x", ylab = "density", ylim = c(0, 2.5), lwd = 2)
# Add lines to the existing plot.
lines(x, y = df(x, df1 = 1, df2 = 100), type = "l", col = "green",
  lwd = 2)
lines(x, y = df(x, df1 = 5, df2 = 1), type = "l", col = "blue",
  lwd = 2)
lines(x, y = df(x, df1 = 5, df2 = 100), type = "l", col = "purple",
  lwd = 2)
lines(x, y = df(x, df1 = 10, df2 = 1), type = "l", col = "red",
  lwd = 2)
lines(x, y = df(x, df1 = 10, df2 = 100), type = "l", col = "orange",
  lwd = 2)
# Add a legend to the top-right.  lty = 1 is for a straight
# solid line.
legend("topright", legend = c("df1=1, df2=1", "df1=1, df2=100",
  "df1=5, df2=1", "df1=5, df2=100", "df1=10, df2=1", "df1=10, df2=100"),
  lty = 1, col = c("black", "green", "blue", "purple", "red",
    "orange"))
```

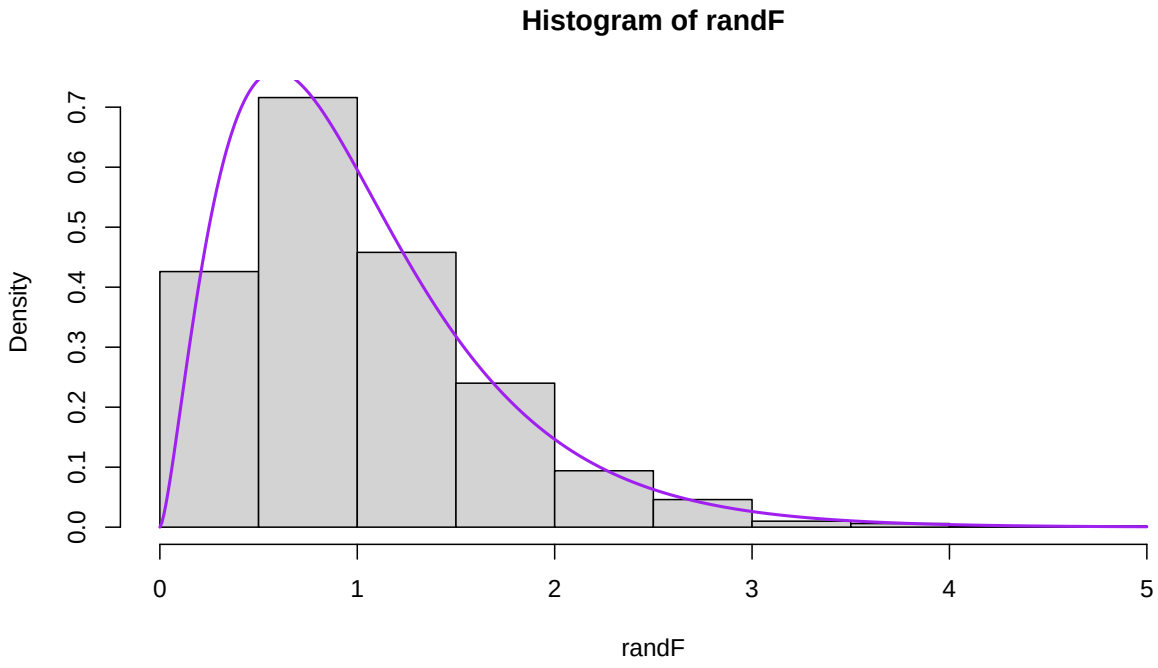


2.10.1.2 Random numbers for the F-distribution

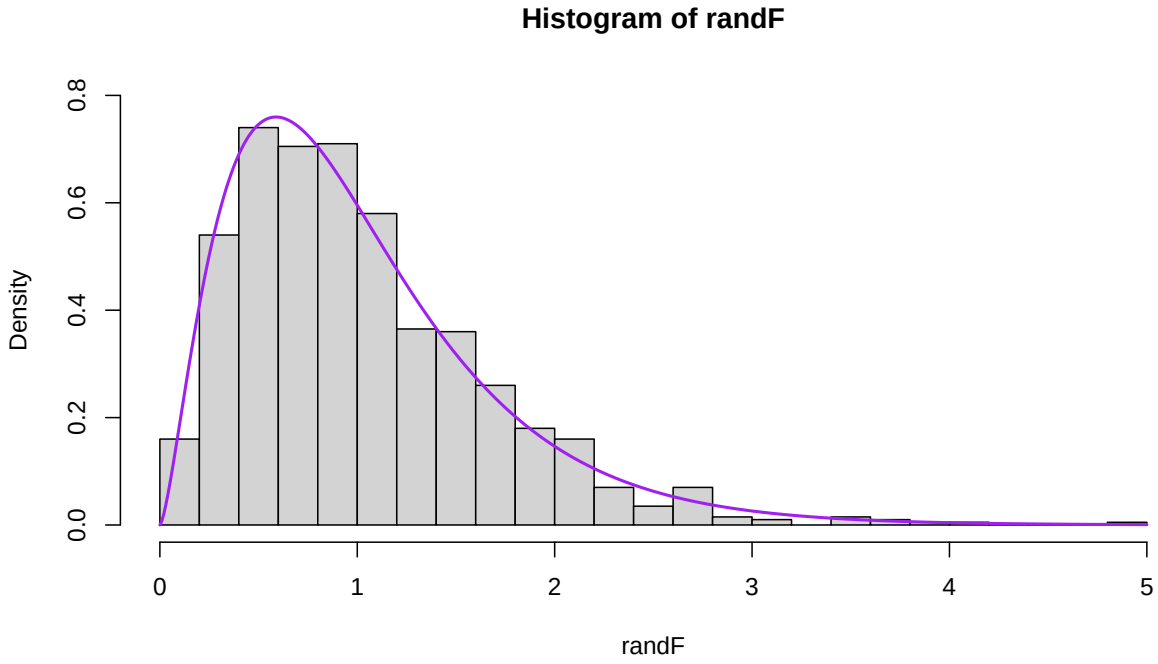
```
# set.seed allows for exact reproduction.
set.seed(12345678)
randF <- rf(1000, 5, 100)
# Generate histogram for the random numbers with exact.
hist(randF)
```



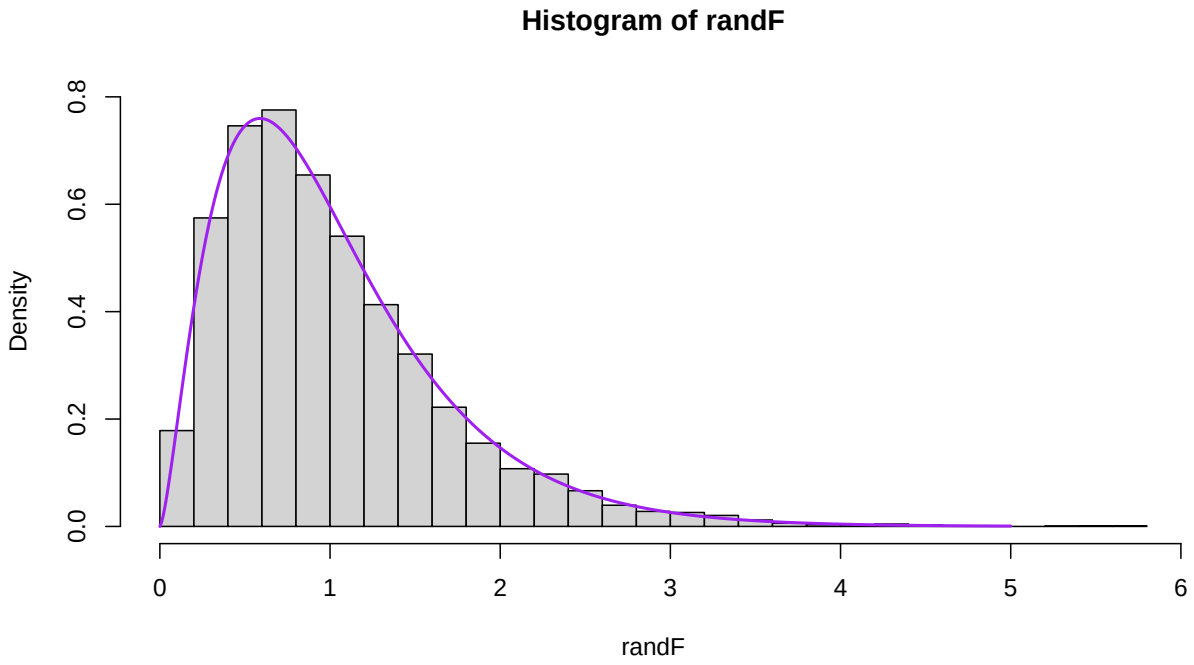

```
# Generate histogram for the random numbers with relative
# frequency.  This is normalized, so we can superimpose an
# F-distribution to it.
hist(randF, freq = FALSE)
# Superimpose an F-distribution on the histogram.
lines(x, y = df(x, df1 = 5, df2 = 100), type = "l", col = "purple",
  lwd = 2)
```




```
# Set y-axis limits and more detailed histogram bins using
# 'breaks = 25'
hist(randF, freq = FALSE, ylim = c(0, 0.8), breaks = 25)
lines(x, y = df(x, df1 = 5, df2 = 100), type = "l", col = "purple",
  lwd = 2)
```



```r
# Generate more random F-distributions to get closer to the
# 'true' density.
randF <- rf(10000, 5, 100)
hist(randF, freq = FALSE, ylim = c(0, 0.8), breaks = 25)
lines(x, y = df(x, df1 = 5, df2 = 100), type = "l", col = "purple",
  lwd = 2)
```



2.10.1.3 Revisit Rocket Example

```r
rocket <- read.csv("csv/rocket.csv")
m1 <- lm(thrust ~ nozzle + propratio, data = rocket)
summary(m1)
```

```
##
## Call:
## lm(formula = thrust ~ nozzle + propratio, data = rocket)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.8459 -1.7555  0.5934  1.2906  3.3008
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 473.6039     4.7158 100.430 4.88e-15 ***
## nozzle       16.7383     1.5329  10.919 1.71e-06 ***
## propratio    -1.0948     0.9414  -1.163    0.275
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.655 on 9 degrees of freedom
```

```
## Multiple R-squared:  0.9303,Adjusted R-squared:  0.9148
## F-statistic: 60.05 on 2 and 9 DF,  p-value: 6.238e-06
```

```
# Compare summary with ANOVA table on board from Oct. 5.
anova(m1)
```

```
## Analysis of Variance Table
##
## Response: thrust
##           Df Sum Sq Mean Sq  F value    Pr(>F)
## nozzle     1 836.67  836.67 118.7377 1.743e-06 ***
## propratio  1   9.53    9.53   1.3524    0.2748
## Residuals  9  63.42    7.05
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
anova(m1)$`Sum Sq`
```

```
## [1] 836.670000   9.529332  63.417335
```

```
sum(anova(m1)$`Sum Sq`[1:2])
```

```
## [1] 846.1993
```

```
SSRes <- anova(m1)$`Sum Sq`[3]
# Test of overall significance.
m_red <- lm(thrust ~ 1, data = rocket)
summary(m_red)
```

```
##
## Call:
## lm(formula = thrust ~ 1, data = rocket)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -13.4167  -7.1167  -0.2167   8.2333  11.3833
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  476.617      2.625   181.6   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.094 on 11 degrees of freedom
```

```
anova(m_red)
```

```
## Analysis of Variance Table
##
## Response: thrust
##           Df Sum Sq Mean Sq F value Pr(>F)
## Residuals 11 909.62  82.692
```

```
SSRes_A <- anova(m_red)$`Sum Sq`[1]
# Manually calculate F-statistic.
l <- 2
n <- nrow(rocket)
p <- 2
Fstat <- ((SSRes_A - SSRes)/l)/(SSRes/(n - p - 1))
```

```
Fstat
```

```
## [1] 60.04505
```

```
pval <- 1 - pf(Fstat, df1 = l, df2 = n - p - 1)
pval
```

```
## [1] 6.238398e-06
```

```
# Automatically calculate F-statistic.
anova(m1, m_red)$F[2]
```

```
## [1] 60.04505
```

2.10.1.4 Revist Coffee Example (Coffee Quality Institute, 2018)

```
coffee <- read.csv("csv/coffee_arabica.csv")
mfull <- lm(Flavor ~ factor(Processing.Method) + Aroma + Aftertaste +
  Body + Acidity + Balance + Sweetness + Uniformity + Moisture,
  dat = coffee)
summary(mfull)
```

```
##
## Call:
## lm(formula = Flavor ~ factor(Processing.Method) + Aroma + Aftertaste +
##     Body + Acidity + Balance + Sweetness + Uniformity + Moisture,
##     data = coffee)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.68587 -0.08465  0.00079  0.08910  0.63633
##
## Coefficients:
##                                                    Estimate Std. Error t value
## (Intercept)                                       -0.728757   0.168516  -4.325
## factor(Processing.Method)Semi-washed / Semi-pulped -0.001396   0.022021  -0.063
## factor(Processing.Method)Washed / Wet              -0.033061   0.011024  -2.999
## Aroma                                               0.220302   0.020447  10.774
## Aftertaste                                          0.468759   0.023912  19.603
## Body                                                0.096140   0.024334   3.951
## Acidity                                             0.216751   0.021194  10.227
## Balance                                             0.046806   0.022558   2.075
## Sweetness                                           0.025507   0.010150   2.513
## Uniformity                                          0.016297   0.009803   1.663
## Moisture                                            0.169012   0.102480   1.649
##                                                   Pr(>|t|)
## (Intercept)                                       1.67e-05 ***
## factor(Processing.Method)Semi-washed / Semi-pulped  0.94947
## factor(Processing.Method)Washed / Wet               0.00277 **
## Aroma                                               < 2e-16 ***
## Aftertaste                                          < 2e-16 ***
## Body                                              8.28e-05 ***
## Acidity                                             < 2e-16 ***
## Balance                                             0.03823 *
## Sweetness                                           0.01211 *
## Uniformity                                          0.09669 .
```

```
## Moisture                                              0.09938 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.148 on 1108 degrees of freedom
## Multiple R-squared:  0.8091,Adjusted R-squared:  0.8073
## F-statistic: 469.5 on 10 and 1108 DF,  p-value: < 2.2e-16
```

```
anova(mfull)
```

```
## Analysis of Variance Table
##
## Response: Flavor
##                             Df Sum Sq Mean Sq   F value    Pr(>F)
## factor(Processing.Method)    2  2.313   1.156   52.8096 < 2.2e-16 ***
## Aroma                        1 67.258  67.258 3071.2889 < 2.2e-16 ***
## Aftertaste                   1 29.097  29.097 1328.6722 < 2.2e-16 ***
## Body                         1  1.129   1.129   51.5460  1.28e-12 ***
## Acidity                      1  2.522   2.522  115.1618 < 2.2e-16 ***
## Balance                      1  0.116   0.116    5.2963 0.0215553 *
## Sweetness                    1  0.251   0.251   11.4392 0.0007442 ***
## Uniformity                   1  0.064   0.064    2.9154 0.0880167 .
## Moisture                     1  0.060   0.060    2.7200 0.0993839 .
## Residuals                 1108 24.264   0.022
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
SSRes <- anova(mfull)$`Sum Sq`[10]
# Reduced model without Uniformity and Moisture
# (beta9=beta10=0):
m_red <- lm(Flavor ~ factor(Processing.Method) + Aroma + Aftertaste +
  Body + Acidity + Balance + Sweetness, dat = coffee)
summary(m_red)
```

```
##
## Call:
## lm(formula = Flavor ~ factor(Processing.Method) + Aroma + Aftertaste +
##     Body + Acidity + Balance + Sweetness, data = coffee)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.67907 -0.08487  0.00054  0.08490  0.64763
##
## Coefficients:
##                                                  Estimate Std. Error t value
## (Intercept)                                     -0.606791   0.159741  -3.799
## factor(Processing.Method)Semi-washed / Semi-pulped  0.002275   0.021969   0.104
## factor(Processing.Method)Washed / Wet           -0.031115   0.011009  -2.826
## Aroma                                            0.221362   0.020472  10.813
## Aftertaste                                       0.470849   0.023858  19.735
## Body                                             0.087671   0.024102   3.637
## Acidity                                          0.219257   0.021182  10.351
## Balance                                          0.047526   0.022283   2.133
## Sweetness                                        0.032406   0.009597   3.377
##                                                 Pr(>|t|)
```

```
## (Intercept)                                          0.000153 ***
## factor(Processing.Method)Semi-washed / Semi-pulped 0.917539
## factor(Processing.Method)Washed / Wet              0.004795 **
## Aroma                                               < 2e-16 ***
## Aftertaste                                          < 2e-16 ***
## Body                                               0.000288 ***
## Acidity                                             < 2e-16 ***
## Balance                                            0.033160 *
## Sweetness                                          0.000759 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1482 on 1110 degrees of freedom
## Multiple R-squared:  0.8081,Adjusted R-squared:  0.8067
## F-statistic: 584.2 on 8 and 1110 DF,  p-value: < 2.2e-16
```

```
anova(m_red)
```

```
## Analysis of Variance Table
##
## Response: Flavor
##                              Df Sum Sq Mean Sq  F value    Pr(>F)
## factor(Processing.Method)     2  2.313   1.156   52.637 < 2.2e-16 ***
## Aroma                         1 67.258  67.258 3061.263 < 2.2e-16 ***
## Aftertaste                    1 29.097  29.097 1324.335 < 2.2e-16 ***
## Body                          1  1.129   1.129   51.378 1.387e-12 ***
## Acidity                       1  2.522   2.522  114.786 < 2.2e-16 ***
## Balance                       1  0.116   0.116    5.279 0.0217690 *
## Sweetness                     1  0.251   0.251   11.402 0.0007591 ***
## Residuals                  1110 24.387   0.022
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
SSRes_A <- anova(m_red)$`Sum Sq`[8]
# Manually calculate F-statistic.
l <- 2
n <- nrow(coffee)
p <- 10
Fstat <- ((SSRes_A - SSRes)/l)/(SSRes/(n - p - 1))
Fstat
```

```
## [1] 2.81769
```

```
pval <- 1 - pf(Fstat, df1 = l, df2 = n - p - 1)
pval
```

```
## [1] 0.06017197
```

```
# Automatically calculate F-statistic.
anova(mfull, m_red)$F[2]
```

```
## [1] 2.81769
```

```
# Reduced model without Uniformity and Moisture and setting
# effect of Dry = Semi (beta1=beta9=beta10=0) 1 = wet, 0
# otherwise
coffee$method2 <- ifelse(coffee$Processing.Method %in% c("Natural / Dry",
  "Semi-washed / Semi-pulped"), 0, 1)
```

```r
# 1 = semi/dry, 0 o.w
coffee$wet <- ifelse(coffee$Processing.Method == "Washed / Wet",
  0, 1)
m_red2 <- lm(Flavor ~ method2 + Aroma + Aftertaste + Body + Acidity +
  Balance + Sweetness, dat = coffee)
summary(m_red2)
```

```
##
## Call:
## lm(formula = Flavor ~ method2 + Aroma + Aftertaste + Body + Acidity +
##     Balance + Sweetness, data = coffee)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.67906 -0.08508  0.00052  0.08490  0.64722
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.606597   0.159659  -3.799 0.000153 ***
## method2     -0.031543   0.010200  -3.092 0.002036 **
## Aroma        0.221408   0.020458  10.823  < 2e-16 ***
## Aftertaste   0.470861   0.023847  19.745  < 2e-16 ***
## Body         0.087561   0.024068   3.638 0.000287 ***
## Acidity      0.219266   0.021173  10.356  < 2e-16 ***
## Balance      0.047527   0.022273   2.134 0.033077 *
## Sweetness    0.032462   0.009577   3.389 0.000725 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1482 on 1111 degrees of freedom
## Multiple R-squared:  0.8081,Adjusted R-squared:  0.8069
## F-statistic: 668.3 on 7 and 1111 DF,  p-value: < 2.2e-16
```

```r
anova(m_red2)
```

```
## Analysis of Variance Table
##
## Response: Flavor
##              Df Sum Sq Mean Sq   F value    Pr(>F)
## method2       1  2.313   2.313  105.3648 < 2.2e-16 ***
## Aroma         1 67.255  67.255 3063.8526 < 2.2e-16 ***
## Aftertaste    1 29.100  29.100 1325.6571 < 2.2e-16 ***
## Body          1  1.126   1.126   51.3088 1.434e-12 ***
## Acidity       1  2.522   2.522  114.9115 < 2.2e-16 ***
## Balance       1  0.116   0.116    5.2882 0.0216552 *
## Sweetness     1  0.252   0.252   11.4883 0.0007249 ***
## Residuals  1111 24.388   0.022
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
SSRes_A <- anova(m_red2)$`Sum Sq`[8]
## Manually calculate F-statistic.
l <- 3
n <- nrow(coffee)
p <- 10
```

```
Fstat <- ((SSRes_A - SSRes)/l)/(SSRes/(n - p - 1))
Fstat
```

```
## [1] 1.882046
```

```
pval <- 1 - pf(Fstat, df1 = l, df2 = n - p - 1)
pval
```

```
## [1] 0.1308207
```

```
# Automatically calculate F-statistic.
anova(mfull, m_red2)$F[2]
```

```
## [1] 1.882046
```

LECTURE 12 | 2020-10-21

2.11 Multicollinearity

Multicollinearity: occurs when some explanatory variables have a **strong linear** relationship amongst themselves. For example, this might occur exactly

$$\boldsymbol{x}_3 = \alpha_0 \mathbf{1} + \alpha_1 \boldsymbol{x}_1 + \alpha_2 \boldsymbol{x}_2$$

in which case the columns of X would be **linearly dependent** and $X^\top X$ does not have an inverse. Practically, there is no new info including $\boldsymbol{x}_3$ when $\boldsymbol{x}_1, \boldsymbol{x}_2$ are in the model. **Approximately**,

$$\boldsymbol{x}_3 \approx \alpha_0 \mathbf{1} + \alpha_1 \boldsymbol{x}_1 + \alpha_2 \boldsymbol{x}_2$$

in which case the columns of X are close to being linearly dependent which cases $\mathbb{V}(\hat{\beta}_j)$ to be **inflated**, in turn leads to inaccurate confidence intervals and conclusions of hypothesis tests for the regression parameters, in practice. $\text{Se}\left(\hat{\beta}_j\right)$ when fitting models can change drastically when adding/removing variables from the model.

EXAMPLE 2.11.1: Hockey (NHL)

In the NHL we have Goals + Assists = Points. Suppose we want to predict a forward's salary. Define

- $x_1 =$ Goals
- $x_2 =$ Assists
- $x_3 =$ Points

$x_3 = x_1 + x_2$, therefore we have exact multicollinearity.

EXAMPLE 2.11.2: Burmese Pythons in Florida (2017)

- $y =$ fat content
- $x_1 =$ mass
- $x_2 =$ overall length
- $x_3 =$ snout-to-vent length

It turns out that x_2 and x_3 are highly correlated. Including all variables in regression lead to inflated $\text{Se}\left(\hat{\beta}_2\right)$ and $\text{Se}\left(\hat{\beta}_3\right)$.

2.12 Detection of Multicollinearity

If two predictors are related

- Scatter plot matrix [all possible pairs of scatter plots b/w $y, x_1, x_2, \ldots, x_p$]
- Correlation matrix (all pairwise correlations)

DEFINITION 2.12.1: Variance inflation factor

For multicollinearity between more than two predictors, we can define the **variance inflation factor** (VIF).

$$\text{VIF}_j = \frac{\mathbb{V}(\hat{\beta}_j)}{\mathbb{V}(\hat{\beta}_j^*)}$$

for $j = 1, \ldots, p$, where $\hat{\beta}_j$ is the estimate of β_j with all predictors in the model, and $\hat{\beta}_j^*$ estimate of β_j based on regression with x_j only.

THEOREM 2.12.2

$$\text{VIF}_j \geqslant 1$$

Fit multiple linear regression model of x_j in terms of other predictors; that is,

$$x_{ij} = \alpha_0 + \alpha_1 x_{i1} + \cdots + \alpha_{j-1} x_{i(j-1)} + \alpha_{j+1} x_{i(j+1)} + \cdots + \alpha_p x_{ip} + \varepsilon_{ij}$$

and compute R^2 for this model, call it R_j^2.

Intuition: if R_j^2 is close to 1, x_j is strongly related linearly to other predictors. It can be shown that

$$\text{VIF}_j = \frac{1}{1 - R_j^2}$$

If $\text{VIF}_j \geq 10$, then there is solid evidence of multicollinearity; that is, $R_j^2 > 0.9$.

Procedure:

- remove predictors with largest VIF, if it exceeds 10. Repeat until no more multicollinearity.

2.12.1 R Demo

```
## Coffee example (Coffee Quality Institute, 2018)
## continued
coffee <- read.csv("csv/coffee_arabica.csv")
# cor(coffee) doesn't work as there's a categorical
# variable.
cor(coffee[, -1])  # e.g., remove first column
```

```
##                 Aroma     Flavor Aftertaste        Body     Acidity    Balance
## Aroma       1.00000000  0.7339782  0.6892744  0.56699932  0.60115765  0.6156508
## Flavor      0.73397820  1.0000000  0.8582783  0.67694834  0.73845546  0.7324530
## Aftertaste  0.68927440  0.8582783  1.0000000  0.67407704  0.69408861  0.7657979
## Body        0.56699932  0.6769483  0.6740770  1.00000000  0.60795391  0.6924568
## Acidity     0.60115765  0.7384555  0.6940886  0.60795391  1.00000000  0.6417994
## Balance     0.61565084  0.7324530  0.7657979  0.69245676  0.64179938  1.0000000
## Sweetness   0.06955938  0.1345364  0.1185760  0.03977892  0.06906093  0.1016718
## Uniformity  0.14785498  0.2132347  0.2143116  0.07195778  0.14876428  0.2180726
## Moisture   -0.11567549 -0.1327342 -0.1745366 -0.21009097 -0.10391684 -0.2161964
##             Sweetness Uniformity    Moisture
## Aroma      0.06955938 0.14785498 -0.11567549
## Flavor     0.13453644 0.21323472 -0.13273418
```

```
## Aftertaste 0.11857600 0.21431157 -0.17453658
## Body       0.03977892 0.07195778 -0.21009097
## Acidity    0.06906093 0.14876428 -0.10391684
## Balance    0.10167183 0.21807265 -0.21619640
## Sweetness  1.00000000 0.34756414  0.08049300
## Uniformity 0.34756414 1.00000000  0.02105693
## Moisture   0.08049300 0.02105693  1.00000000
```

Plot the pairs (disabled due to loading time on PDF).

```
pairs(~Flavor + Aroma + Aftertaste + Body + Acidity + Balance +
  Sweetness + Uniformity + Moisture, data = coffee)
```

```
# Code our own indicators, so that we can more easily
# interpret VIFs.  1 = wet, 0 otherwise
coffee$wet <- ifelse(coffee$Processing.Method == "Washed / Wet",
  1, 0)
# 1 = semi/dry, 0 otherwise
coffee$semi <- ifelse(coffee$Processing.Method == "Semi-washed / Semi-pulped",
  1, 0)
```

Model:

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \beta_3 x_{i3} + \beta_4 x_{i4} + \beta_5 x_{i5} + \beta_6 x_{i6} + \beta_7 x_{i7} + \beta_8 x_{i8} + \beta_9 x_{i9} + \beta_{10} x_{i(10)} + \varepsilon_i$$

where

- $y =$ flavour
- $x_1 = 1$ if wet, 0 otherwise
- $x_2 = 1$ if semi, 0 otherwise
- $x_3 =$ Aroma
- $x_4 =$ Aftertaste
- $x_5 =$ Body
- $x_6 =$ Acidity
- $x_7 =$ Balance
- $x_8 =$ Sweetness
- $x_9 =$ Uniformity
- $x_{10} =$ Moisture

```
# Full MLR with our manually coded indicators.
mfull <- lm(Flavor ~ wet + semi + Aroma + Aftertaste + Body +
  Acidity + Balance + Sweetness + Uniformity + Moisture, dat = coffee)
summary(mfull)
```

```
##
## Call:
## lm(formula = Flavor ~ wet + semi + Aroma + Aftertaste + Body +
##     Acidity + Balance + Sweetness + Uniformity + Moisture, data = coffee)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.68587 -0.08465  0.00079  0.08910  0.63633
```

```
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.728757   0.168516  -4.325 1.67e-05 ***
## wet         -0.033061   0.011024  -2.999  0.00277 **
## semi        -0.001396   0.022021  -0.063  0.94947
## Aroma        0.220302   0.020447  10.774  < 2e-16 ***
## Aftertaste   0.468759   0.023912  19.603  < 2e-16 ***
## Body         0.096140   0.024334   3.951 8.28e-05 ***
## Acidity      0.216751   0.021194  10.227  < 2e-16 ***
## Balance      0.046806   0.022558   2.075  0.03823 *
## Sweetness    0.025507   0.010150   2.513  0.01211 *
## Uniformity   0.016297   0.009803   1.663  0.09669 .
## Moisture     0.169012   0.102480   1.649  0.09938 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.148 on 1108 degrees of freedom
## Multiple R-squared:  0.8091,Adjusted R-squared:  0.8073
## F-statistic: 469.5 on 10 and 1108 DF,  p-value: < 2.2e-16
```

```
# Full MLR alternative, using the factor command.
mfull_alternative <- lm(Flavor ~ factor(Processing.Method) +
  Aroma + Aftertaste + Body + Acidity + Balance + Sweetness +
  Uniformity + Moisture, dat = coffee)
```

Suppose we want to check the VIF for $j = 1$; that is, x_1. Now, we fit:

$$x_{i1} = \alpha_0 + \alpha_2 x_{i2} + \alpha_3 x_{i3} + \alpha_4 x_{i4} + \alpha_5 x_{i5} + \alpha_6 x_{i6} + \alpha_7 x_{i7} + \alpha_8 x_{i8} + \alpha_9 x_{i9} + \alpha_{10} x_{i(10)} + \varepsilon_i$$

```
wet_reg <- lm(wet ~ semi + Aroma + Aftertaste + Body + Acidity +
  Balance + Sweetness + Uniformity + Moisture, dat = coffee)
summary(wet_reg)
```

```
##
## Call:
## lm(formula = wet ~ semi + Aroma + Aftertaste + Body + Acidity +
##     Balance + Sweetness + Uniformity + Moisture, data = coffee)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.0015 -0.0283  0.1770  0.2522  0.7704
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.81748    0.45838   1.783 0.074794 .
## semi        -0.75675    0.05551 -13.632  < 2e-16 ***
## Aroma        0.09690    0.05562   1.742 0.081774 .
## Aftertaste  -0.13169    0.06502  -2.026 0.043054 *
## Body        -0.21885    0.06596  -3.318 0.000936 ***
## Acidity      0.18696    0.05746   3.254 0.001173 **
## Balance     -0.10804    0.06136  -1.761 0.078563 .
## Sweetness    0.08373    0.02753   3.041 0.002413 **
## Uniformity   0.03547    0.02668   1.329 0.184053
## Moisture     0.59486    0.27858   2.135 0.032956 *
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4031 on 1109 degrees of freedom
## Multiple R-squared:  0.1911,Adjusted R-squared:  0.1845
## F-statistic: 29.11 on 9 and 1109 DF,  p-value: < 2.2e-16
```

```
r2_wet <- summary(wet_reg)$r.squared
r2_wet
```

```
## [1] 0.191077
```

R_j: In our case, $R_1 = 0.191077$.

```
VIF_wet <- 1/(1 - r2_wet)
VIF_wet
```

```
## [1] 1.236212
```

VIF_j: $\text{VIF}_1 = 1.236212$. Interpretation: in a regression with all the variables compared to a regression with just this one, the estimated variance has increased by a factor of 1.24, which is not a very large inflation. The variable wet is not very linearly correlated or dependent on the other predictors that we have in the model.

```
Aroma_reg <- lm(Aroma ~ wet + semi + Aftertaste + Body + Acidity +
  Balance + Sweetness + Uniformity + Moisture, dat = coffee)
r2_Aroma <- summary(Aroma_reg)$r.squared
r2_Aroma
```

```
## [1] 0.5204716
```

```
VIF_Aroma <- 1/(1 - r2_Aroma)
VIF_Aroma
```

```
## [1] 2.085382
```

$R_3 = 0.5204716$, $\text{VIF}_3 = 2.085382$.

```
Aftertaste_reg <- lm(Aftertaste ~ wet + semi + Aroma + Body +
  Acidity + Balance + Sweetness + Uniformity + Moisture, dat = coffee)
r2_Aftertaste <- summary(Aftertaste_reg)$r.squared
r2_Aftertaste
```

```
## [1] 0.7101012
```

```
VIF_Aftertaste <- 1/(1 - r2_Aftertaste)
VIF_Aftertaste
```

```
## [1] 3.449479
```

```
# Load car library for automatic VIF calculation using
# vif()
library(car)
vif(mfull)
```

```
##        wet       semi      Aroma Aftertaste       Body    Acidity    Balance
##   1.236212   1.178004   2.085382   3.449479   2.317728   2.232210   3.002813
##  Sweetness Uniformity   Moisture
##   1.159602   1.209901   1.086101
```

No serious signs of inflation, all VIFs are less than 10.

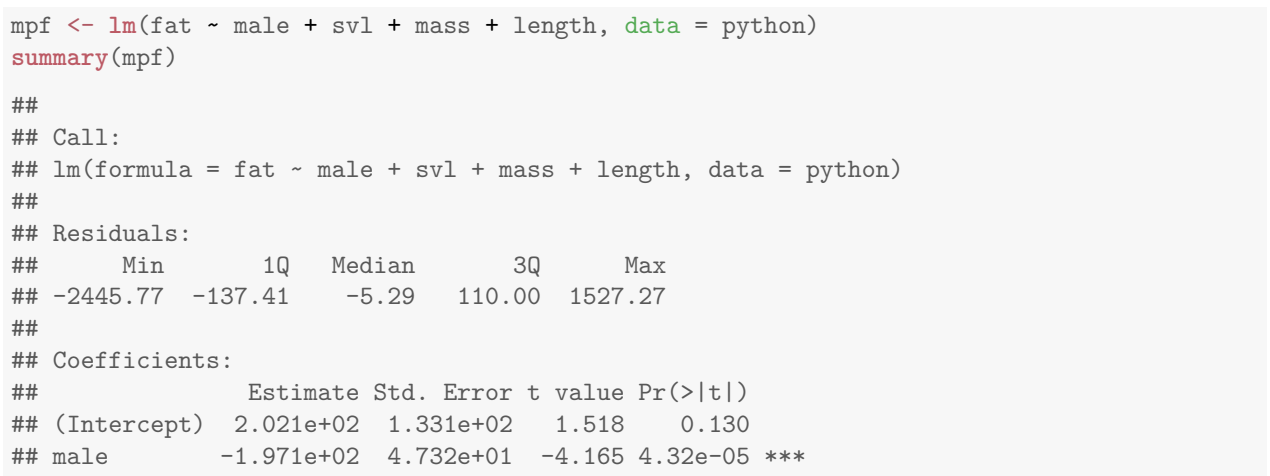
```
## Python in FL everglades example (2017) Sex, length,
## total mass, fat mass, and specimen condition data for
## 248 Burmese pythons (Python bivittatus) collected in the
## Florida Everglades
python <- read.csv("csv/FLpython.csv")
head(python)
```

```
##   sex  svl mass length    fat
## 1   F 70.0  186   77.5  6.000
## 2   M 76.0  310   83.8 11.000
## 3   M 77.0  260   86.1  6.000
## 4   M 78.0  262   87.1  8.000
## 5   M 81.0  306   91.1  4.000
## 6   M 93.5  605  104.6 18.959
```

```
python$male <- ifelse(python$sex == "M", 1, 0)  # 1 = M, 0 =F
cor(python[, -1])
```

```
##               svl       mass     length        fat       male
## svl     1.0000000  0.8843022  0.9994935  0.8098652 -0.1602418
## mass    0.8843022  1.0000000  0.8858256  0.9419114 -0.2190993
## length  0.9994935  0.8858256  1.0000000  0.8114658 -0.1593512
## fat     0.8098652  0.9419114  0.8114658  1.0000000 -0.2933111
## male   -0.1602418 -0.2190993 -0.1593512 -0.2933111  1.0000000
```

```
pairs(python[, -1])
```



```
mpf <- lm(fat ~ male + svl + mass + length, data = python)
summary(mpf)
```

```
##
## Call:
## lm(formula = fat ~ male + svl + mass + length, data = python)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2445.77  -137.41    -5.29   110.00  1527.27
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.021e+02  1.331e+02   1.518    0.130
## male        -1.971e+02  4.732e+01  -4.165 4.32e-05 ***
## svl         -3.370e+00  1.125e+01  -0.300    0.765
## mass         1.178e-01  5.302e-03  22.210  < 2e-16 ***
## length       1.594e+00  1.010e+01   0.158    0.875
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 360.9 on 243 degrees of freedom
## Multiple R-squared:  0.897,Adjusted R-squared:  0.8953
## F-statistic:   529 on 4 and 243 DF,  p-value: < 2.2e-16
```

```
vif(mpf)
```

```
##        male         svl        mass      length
##    1.058699  994.546545    4.813078 1007.484200
```

```
mpf_l <- lm(length ~ male + svl + mass, data = python)
1/(1 - summary(mpf_l)$r.squared)
```

```
## [1] 1007.484
```

Misleading conclusion: `svl` and `length` are both irrelevant (this is not the case). Also, the standard errors are very large.

```
# remove 'length' based on VIF
mpf2 <- lm(fat ~ male + mass + svl, data = python)
summary(mpf2)$adj
```

```
## [1] 0.8957164
```

```
vif(mpf2)
```

```
##     male     mass      svl
## 1.056139 4.720065 4.611903
```

```
anova(mpf2)
```

```
## Analysis of Variance Table
##
## Response: fat
##            Df    Sum Sq   Mean Sq   F value  Pr(>F)
## male        1  26435988  26435988  203.7689 < 2e-16 ***
## mass        1 248624259 248624259 1916.3988 < 2e-16 ***
## svl         1    567377    567377    4.3734 0.03754 *
## Residuals 244  31655372    129735
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
AIC(mpf2)
```

```
## [1] 3629.527
```

`svl` now has a significant t-statistic.

LECTURE 13 | 2020-10-26

2.13 Model Selection Criteria

Given p explanatory variables, find the subset $k \leqslant p$ of explanatory variables ("reduced model") that gives us the "best" model:

- Goodness of fit.
- Interpretability.
- Predictive performance.

Some related concepts:

1. F tests compare between 2 specific models where test adequacy of a "reduced" model (subset, "nested") relative to full model.

   Quiz 4: $\beta_1 = \beta_2$ in part d-f

2. Multicollinearity: can affect interpretability of $\hat{\beta}_j$ usual interpretation "holding other variables constant" doesn't really work when x_j is strongly correlated with other predictors.

3. R^2 is the proportion of variability in the response explained by the regression model. It always increases when adding variables.
4. $\hat{\sigma}^2$ is estimated residual variable, used for prediction, want $\hat{\sigma}^2$ small to give good predictive performance

Two key ingredients:

- Metric (or criterion) for comparing different models with potentially different number of predictors
- Selection/search strategy (which models should we fit and test?)

Examples of metrics for model selection:

DEFINITION 2.13.1: Adjusted R^2

$$R^2_{\text{adj}} = 1 - \frac{\text{SS(Res)}/(n-k-1)}{\text{SS(Total)}/(n-1)} = 1 - \frac{\hat{\sigma}^2}{s^2}$$

for model with k predictors.

Compared to

$$R^2 = 1 - \frac{\text{SS(Res)}}{\text{SS(Total)}} = 1 - \frac{\hat{\sigma}^2(n-k-1)}{s^2(n-1)}$$

- $\text{SS(Res)}/(n-k-1)$ estimated $\hat{\sigma}^2$ for model with k predictors
- $\text{SS(Total)}/(n-1)$ is the sample variance of responses y_i.

$$\begin{aligned} R^2_{\text{adj}} = 1 - \frac{n-1}{n-k-1}(1-R^2) &= 1 - \left(1 + \frac{k}{n-k-1}\right)(1-R^2) \\ &= 1 - \left[1(1-R^2) + \left(\frac{k}{n-k-1}\right)(1-R^2)\right] \\ &= 1 - \left[1 - R^2 + \left(\frac{k}{n-k-1}\right)(1-R^2)\right] \\ &= 1 - 1 + R^2 - \frac{k}{n-k-1}(1-R^2) \\ &= R^2 - (1-R^2)\frac{k}{n-k-1} \end{aligned}$$

Intuition: R^2_{adj} accounts for number variables in model, *penalizes* inclusion of unimportant predictors; that is, SS(Res) has little decrease when adding such variables. Meanwhile, R^2 always increases with more predictors, but R^2_{adj} can decrease if SS(Res) change is small.

While R^2_{adj} loses its usual interpretation of R^2, but can be used as a measure of "goodness of fit" and model selection criterion (e.g., pick subset of predictors that gives the highest R^2_{adj}).

EXAMPLE 2.13.2

Given

- $n = 25$
- $\text{SS(Total)} = 20$
- $p = 6$

Suppose we're considering on a subset of $k = 4$ predictors, and find:

	Reduced	Full
	$k = 4$	$p = 6$
SS(Total)	20	20
SS(Res)	10	9.8
R^2	$10/20 = 0.5$	$9.8/20 = 0.49$
R^2_{adj}	$1 - \frac{10/(25-4-1)}{20/(25-1)} = 0.4$	$1 - \frac{9.8/(25-6-1)}{20/(25-1)} \approx 0.347$
$\hat{\sigma}^2$	$10/(25-4-1) = 0.5$	$9.8/(25-6-1) \approx 0.544$

- $n - k - 1$ d.f. Res in reduced
- $n - p - 1$ d.f. Res in full

Remarks:

- $R^2_{\text{adj}} < R^2$, but as $n \to \infty$, $R^2_{\text{adj}} \to R^2$.
- Model with higher R^2_{adj} has lower $\hat{\sigma}^2$, thus is a reasonable metric for model selection.

DEFINITION 2.13.3: Akaike Information Criterion (AIC)

Let n be sample size and q be the number of estimated parameters.

- MLR: $q = p + 2$ since we have p predictors + 1 intercept (β_0) + 1 (σ^2).

The **Akaike information criterion** (AIC) is defined as

$$\text{AIC} = 2q - 2\ln[L(\hat{\theta})]$$

where $L(\hat{\theta})$ is the likelihood function evaluated at $\hat{\theta}$ (parameter estimates).

REMARK 2.13.4

- The least square estimates of β are equivalent to maximum likelihood estimates under the usual normal assumptions on ε.
- $2q$ is the penalty for including more predictors. With more parameters, $L(\hat{\theta})$ increases, offset by penalty $2q$.
- The model with lower AIC is preferred; that is, differences in AIC matter not the value itself.

REMARK 2.13.5: †

If we want to measure just the difference of AIC, we can do

$$\text{AIC} = n\ln\left[\frac{\text{SS(Res)}}{n}\right] + 2q$$

DEFINITION 2.13.6: Bayesian Information Criterion (BIC)

Let n be sample size and q be the number of estimated parameters.

- MLR: $q = p + 2$ since we have p predictors + 1 intercept (β_0) + 1 (σ^2).

The **Bayesian information criterion** (BIC) is defined as

$$\text{BIC} = q\ln(n) - 2\ln[L(\hat{\theta})]$$

where $L(\hat{\theta})$ is the likelihood function evaluated at $\hat{\theta}$ (parameter estimates).

REMARK 2.13.7

AIC is similar to BIC, but BIC strongly penalizes inclusion of more variables. Note that in BIC, $q\ln(n)$ depends on sample size.

REMARK 2.13.8: †

If we want to measure just the difference of BIC, we can do

$$\text{BIC} = n\ln\left[\frac{\text{SS(Res)}}{n}\right] + q\ln(n)$$

Recap:

- R^2, AIC, BIC are all based on comparing the fitted models. In other words, they look at the explanatory power of the model.
- They all have penalties to try to prevent "overfitting." That is, having too many variables might end up modelling spurious relationships that are actually noise.

Mean Square Prediction Error (MSPE)

Consider predictive performance of model on *new* data; that is, data *not* used in fitting of models. "Is model generalisable to new data?" Overfitted models tend to have high prediction error.

For example, via cross-validation schemes. We've given 4 examples of metrics/criteria for comparing models. Imagine we have p predictors:

$$\begin{matrix}\binom{p}{1} & 1 \text{ predictor} \\ \binom{p}{2} & 2 \text{ predictors} \\ \vdots & \\ \binom{p}{p} & p \text{ predictors}\end{matrix} \implies \sum_{j=0}^{p}\binom{p}{j} = 2^p$$

Occam's Razor: "The simplest explanation is usually the best one."—William Ockham

LECTURE 14 | 2020-10-28

2.14 Model Selection Basic Strategies

- Criteria: R^2_{adj}, AIC, BIC, MSPE, etc. explicitly penalizes unnecessarily complex models.
- Search strategies (use with chosen criterion)

(i) Brute force: fit all possible regressions. With p predictors, we have $\sum_{j=0}^{p}\binom{p}{j} = 2^p$ possible models to fit.

- Finds optimal model that may be computationally intensive (or infeasible) if p is large.

Idea: Find a "good" (useful) model in reasonable computational time (not necessarily optimal). Many strategies focus on adding/removing variables one at a time.

(ii) Forward selection: add one variable at a time to model.

- Start with a model that only has an intercept (β_0).
- Fit p simple linear regression models

$$\boldsymbol{y} = \beta_0\mathbf{1} + \beta_1\boldsymbol{x}_j + \boldsymbol{\varepsilon} \quad j = 1, \ldots, p$$

- Pick the best of p models (with 1 predictor) according to chosen criterion, and add that variable x_j to model.

- Fit $(p-1)$ models containing x_j and one other variable.
  - If none of $(p-1)$ models improves criterion, stop.
  - Pick the best of $(p-1)$ models according to criterion, so now we have 2 variables in the model.

Continue adding 1 variable at a time in this way until we can no more variables improve the criterion. The final model is one with the best criterion after we *stop*; that is, no further improvement is possible.

Note: Much faster than brute force as the maximum number of models to fit is:

$$p + (p-1) + \cdots + 2 + 1 = \sum_{i=1}^{p} i = \frac{p(p+1)}{2}$$

which is $\mathcal{O}(p^2)$ compared to $\mathcal{O}(2^p)$ for all possible regressions.

(iii) Backward direction: remove one variable at a time to model.

- Start with model that has p predictors.
- Fit p models that result from removing one variable from the regression; that is, each one has $(p-1)$ variables.
- Pick the best of p models according to criterion.
  - Eliminate that variable x_j from model.
  - Fit $(p-1)$ models that remove x_j and one other variable from model.
  - Pick best of $(p-1)$ models (2 variables removed).

Continue removing 1 variable at a time in this way until we can no more variables improve the criterion. Same computational complexity as forward selection.

(iv) Forward-backwards (allows individual variables to be both added/removed)

- Start as in forward selection
- If we have k variables in model:
  - Backwards: fit k models with $(k-1)$ variables. If any of these improve criterion, remove the variable.
  - Forwards: fit $(p-k)$ models with $(k+1)$ variables. If any of these improve criterion, add that variable.

- These are the basic "stepwise" selection models to get a "good" (useful) model.
- Many other have sophisticated procedures available. For example, stochastic search, lasso.
- We've assumed that $n > p$ because otherwise $(X^\top X)$ is not invertible. More specialized methods needed if number of predictors is larger than sample size.

2.14.1 R Demo

```
## Coffee example (Coffee Quality Institute, 2018)
## continued.
coffee <- read.csv("csv/coffee_arabica.csv")
mfull <- lm(Flavor ~ factor(Processing.Method) + Aroma + Aftertaste +
  Body + Acidity + Balance + Sweetness + Uniformity + Moisture,
  dat = coffee)
summary(mfull)$adj.r.squared
```

```
## [1] 0.8073297
```

```
AIC(mfull)
```

```
## [1] -1087.524
```

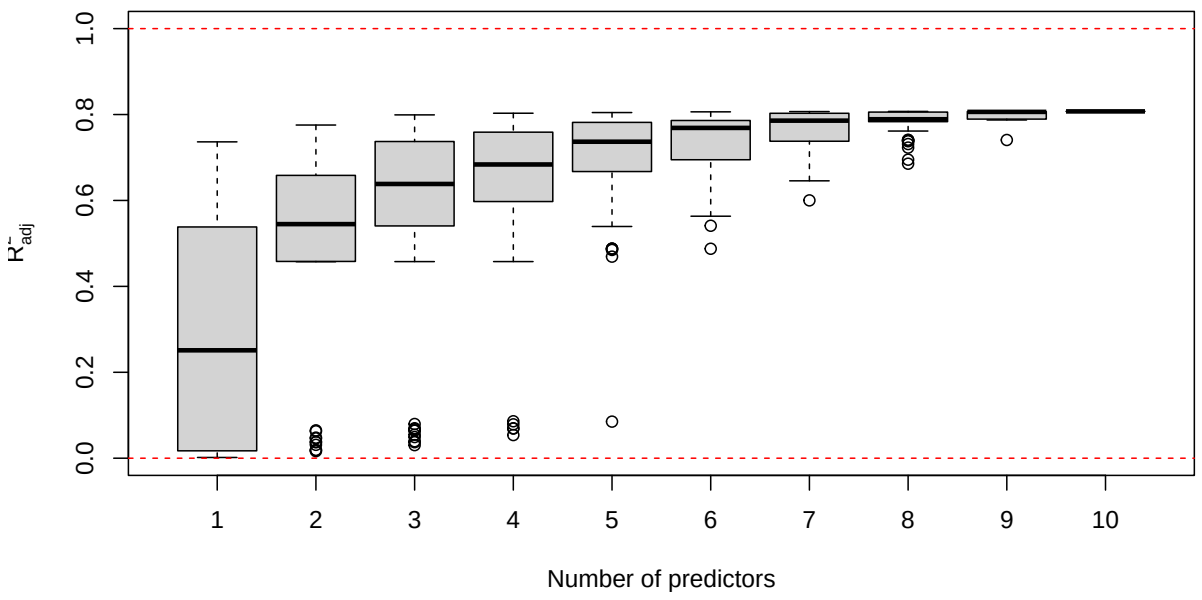
```
BIC(mfull)
```

```
## [1] -1027.282
```

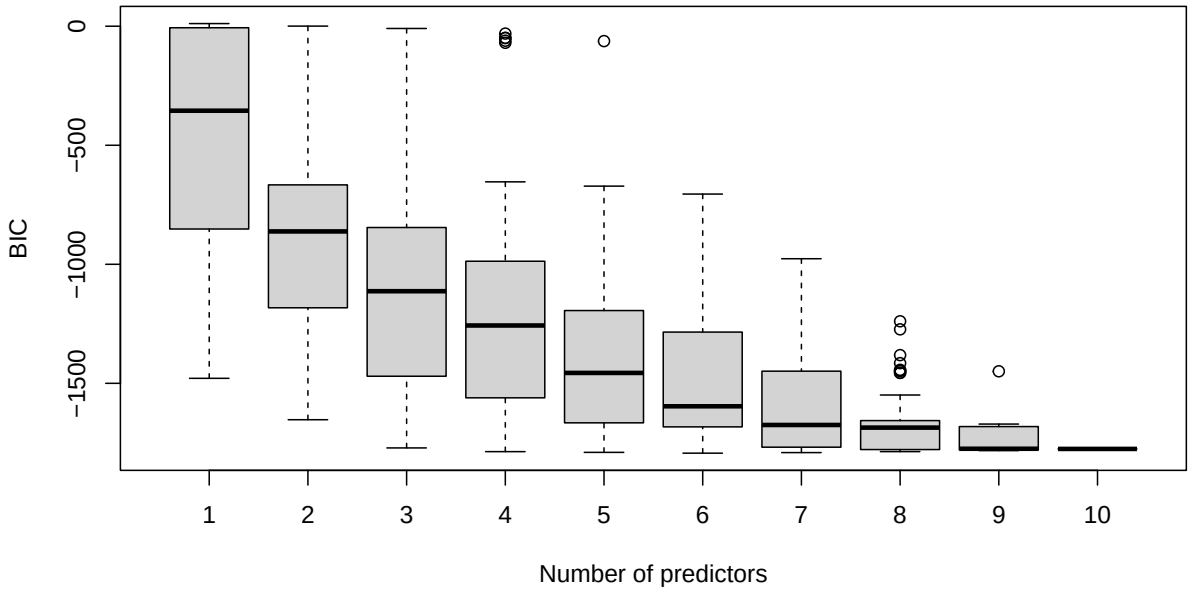
```
library(leaps)
# Exhaustive, brute-force search.
all_regs <- regsubsets(Flavor ~ ., data = coffee, nvmax = 10,
  nbest = 2^10, really.big = TRUE)
all_regs_summ <- summary(all_regs)
```

```
all_regs_summ$which
all_regs_summ$adjr2
all_regs_summ$bic
```

```
# Organize results according to number of variables in
# model.
p <- 10
k <- c(rep(1, choose(p, 1)), rep(2, choose(p, 2)), rep(3, choose(p,
  3)), rep(4, choose(p, 4)), rep(5, choose(p, 5)), rep(6, choose(p,
  6)), rep(7, choose(p, 7)), rep(8, choose(p, 8)), rep(9, choose(p,
  9)), rep(10, choose(p, 10)))
boxplot(all_regs_summ$adjr2 ~ k, xlab = "Number of predictors",
  ylab = expression(R[adj]^2), ylim = c(0, 1))
abline(h = c(0, 1), lty = 2, col = "red")
```




```
boxplot(all_regs_summ$bic ~ k, xlab = "Number of predictors",
  ylab = "BIC")
```




```
max(all_regs_summ$adjr2)
```

```
## [1] 0.8075027
```

```
bestR2adj <- which.max(all_regs_summ$adjr2)
min(all_regs_summ$bic)
```

```
## [1] -1793.389
```

```
bestBIC <- which.min(all_regs_summ$bic)
# Find out which predictors in those models.
all_regs_summ$which[bestR2adj, ]
```

```
##                                (Intercept)
##                                       TRUE
## Processing.MethodSemi-washed / Semi-pulped
##                                      FALSE
##              Processing.MethodWashed / Wet
##                                       TRUE
##                                      Aroma
##                                       TRUE
##                                 Aftertaste
##                                       TRUE
##                                       Body
##                                       TRUE
##                                   Acidity
##                                       TRUE
##                                   Balance
##                                       TRUE
##                                 Sweetness
##                                       TRUE
##                                Uniformity
##                                       TRUE
```

```
##                                   Moisture
##                                       TRUE

all_regs_summ$which[bestBIC, ]

##                                 (Intercept)
##                                        TRUE
## Processing.MethodSemi-washed / Semi-pulped
##                                       FALSE
##              Processing.MethodWashed / Wet
##                                        TRUE
##                                     Aroma
##                                        TRUE
##                                  Aftertaste
##                                        TRUE
##                                      Body
##                                        TRUE
##                                   Acidity
##                                        TRUE
##                                   Balance
##                                     FALSE
##                                 Sweetness
##                                        TRUE
##                                Uniformity
##                                     FALSE
##                                  Moisture
##                                     FALSE

coffee$wet <- ifelse(coffee$Processing.Method == "Washed / Wet",
  1, 0)  # 1 = wet, 0 otherwise
coffee$semi <- ifelse(coffee$Processing.Method == "Semi-washed / Semi-pulped",
  1, 0)  # 1 = semi/dry, 0 otherwise
coffee$Processing.Method <- NULL
m_bestr2adj <- lm(Flavor ~ wet + Aroma + Aftertaste + Body +
  Acidity + Balance + Sweetness + Uniformity + Moisture, dat = coffee)
summary(m_bestr2adj)

##
## Call:
## lm(formula = Flavor ~ wet + Aroma + Aftertaste + Body + Acidity +
##     Balance + Sweetness + Uniformity + Moisture, data = coffee)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.68587 -0.08469  0.00080  0.08923  0.63660
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.728709   0.168439  -4.326 1.65e-05 ***
## wet         -0.032797   0.010197  -3.216  0.00134 **
## Aroma        0.220278   0.020434  10.780  < 2e-16 ***
## Aftertaste   0.468749   0.023901  19.612  < 2e-16 ***
## Body         0.096194   0.024308   3.957 8.06e-05 ***
## Acidity      0.216754   0.021185  10.232  < 2e-16 ***
## Balance      0.046793   0.022547   2.075  0.03819 *
```

```
## Sweetness    0.025480   0.010136   2.514  0.01209 *
## Uniformity   0.016291   0.009798   1.663  0.09665 .
## Moisture     0.168439   0.102033   1.651  0.09906 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1479 on 1109 degrees of freedom
## Multiple R-squared:  0.8091,Adjusted R-squared:  0.8075
## F-statistic: 522.1 on 9 and 1109 DF,  p-value: < 2.2e-16
```

```
AIC(m_bestr2adj)
```

```
## [1] -1089.52
```

```
BIC(m_bestr2adj)
```

```
## [1] -1034.298
```

```
m_bestBIC <- lm(Flavor ~ wet + Aroma + Aftertaste + Body + Acidity +
  Sweetness, dat = coffee)
summary(m_bestBIC)
```

```
##
## Call:
## lm(formula = Flavor ~ wet + Aroma + Aftertaste + Body + Acidity +
##     Sweetness, data = coffee)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.65627 -0.08781  0.00032  0.08529  0.63010
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.609003   0.159910  -3.808 0.000148 ***
## wet         -0.032852   0.010198  -3.221 0.001313 **
## Aroma        0.225969   0.020378  11.089  < 2e-16 ***
## Aftertaste   0.490988   0.021938  22.381  < 2e-16 ***
## Body         0.103438   0.022926   4.512 7.11e-06 ***
## Acidity      0.225638   0.020994  10.748  < 2e-16 ***
## Sweetness    0.033445   0.009582   3.491 0.000501 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1484 on 1112 degrees of freedom
## Multiple R-squared:  0.8073,Adjusted R-squared:  0.8063
## F-statistic: 776.4 on 6 and 1112 DF,  p-value: < 2.2e-16
```

```
AIC(m_bestBIC)
```

```
## [1] -1085.26
```

```
BIC(m_bestBIC)
```

```
## [1] -1045.098
```

```
# Let's also try stepwise methods.
library(MASS)
# Full model and empty model with just intercept.
full <- lm(Flavor ~ ., data = coffee)
```

```r
empty <- lm(Flavor ~ 1, data = coffee)
# Default stepAIC uses AIC criterion.
m_f_AIC <- stepAIC(object = empty, scope = list(upper = full,
  lower = empty), direction = "forward", trace = 0)
# Let's get stepAIC to use BIC by specifying the penalty k
# = log(n).  Add 'trace = 0' to hide the output.  Forward.
m_f <- stepAIC(object = empty, scope = list(upper = full, lower = empty),
  direction = "forward", trace = 0, k = log(nrow(coffee)))
summary(m_f)
```

```
##
## Call:
## lm(formula = Flavor ~ Aftertaste + Acidity + Aroma + Body + Sweetness +
##     wet, data = coffee)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.65627 -0.08781  0.00032  0.08529  0.63010
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.609003   0.159910  -3.808 0.000148 ***
## Aftertaste   0.490988   0.021938  22.381  < 2e-16 ***
## Acidity      0.225638   0.020994  10.748  < 2e-16 ***
## Aroma        0.225969   0.020378  11.089  < 2e-16 ***
## Body         0.103438   0.022926   4.512 7.11e-06 ***
## Sweetness    0.033445   0.009582   3.491 0.000501 ***
## wet         -0.032852   0.010198  -3.221 0.001313 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1484 on 1112 degrees of freedom
## Multiple R-squared:  0.8073,Adjusted R-squared:  0.8063
## F-statistic: 776.4 on 6 and 1112 DF,  p-value: < 2.2e-16
```

```r
# Backward.
m_b <- stepAIC(object = full, scope = list(upper = full, lower = empty),
  direction = "backward", trace = 0, k = log(nrow(coffee)))
summary(m_b)
```

```
##
## Call:
## lm(formula = Flavor ~ Aroma + Aftertaste + Body + Acidity + Sweetness +
##     wet, data = coffee)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.65627 -0.08781  0.00032  0.08529  0.63010
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.609003   0.159910  -3.808 0.000148 ***
## Aroma        0.225969   0.020378  11.089  < 2e-16 ***
## Aftertaste   0.490988   0.021938  22.381  < 2e-16 ***
## Body         0.103438   0.022926   4.512 7.11e-06 ***
```

```
## Acidity      0.225638   0.020994  10.748  < 2e-16 ***
## Sweetness    0.033445   0.009582   3.491 0.000501 ***
## wet         -0.032852   0.010198  -3.221 0.001313 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1484 on 1112 degrees of freedom
## Multiple R-squared:  0.8073,Adjusted R-squared:  0.8063
## F-statistic: 776.4 on 6 and 1112 DF,  p-value: < 2.2e-16
```

```
# Forward-backward.
m_h <- stepAIC(object = empty, scope = list(upper = full, lower = empty),
  direction = "both", trace = 0, k = log(nrow(coffee)))
summary(m_h)
```

```
##
## Call:
## lm(formula = Flavor ~ Aftertaste + Acidity + Aroma + Body + Sweetness +
##     wet, data = coffee)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.65627 -0.08781  0.00032  0.08529  0.63010
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.609003   0.159910  -3.808 0.000148 ***
## Aftertaste   0.490988   0.021938  22.381  < 2e-16 ***
## Acidity      0.225638   0.020994  10.748  < 2e-16 ***
## Aroma        0.225969   0.020378  11.089  < 2e-16 ***
## Body         0.103438   0.022926   4.512 7.11e-06 ***
## Sweetness    0.033445   0.009582   3.491 0.000501 ***
## wet         -0.032852   0.010198  -3.221 0.001313 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1484 on 1112 degrees of freedom
## Multiple R-squared:  0.8073,Adjusted R-squared:  0.8063
## F-statistic: 776.4 on 6 and 1112 DF,  p-value: < 2.2e-16
```

10 variables is still a fairly small problem: in this example all 3 approaches identify the same BIC-based model as the exhaustive search.

LECTURE 15 | 2020-11-02

2.15 Residual Plots for Linear Regression Assumptions

Recall: $\boldsymbol{Y} = X\boldsymbol{\beta} + \boldsymbol{\varepsilon}$ where $\boldsymbol{\varepsilon} \sim \text{MVN}(0, \sigma^2 I_n)$. Practically, this means

$$\varepsilon \overset{\text{iid}}{\sim} \mathcal{N}(0, \sigma^2)$$

- Independence among all error terms
- Normally distributed

- Since $\mathbb{E}[Y_i] = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip}$ is implied by $\mathbb{E}[\varepsilon_i] = 0$ for any $x_{i1}, \ldots, x_{ip}$, the linear model is appropriate; that is, it correctly explains response on average.
- Constant error variance σ^2

We could assess these assumptions via ε_i's, but we can't observe ε_i directly. Rather, we do have an approximation via residuals e_i from the fitted model.

Recall, $\boldsymbol{e} \sim \text{MVN}(0, (I-H)\sigma^2)$. So $\boldsymbol{e}$ and $\boldsymbol{\varepsilon}$ are related:

$$\boldsymbol{e} = \boldsymbol{Y} - X\hat{\boldsymbol{\beta}} = (I-H)\boldsymbol{Y} = (I-H)(X\boldsymbol{\beta} + \boldsymbol{\varepsilon}) = (X\boldsymbol{\beta} - HX\boldsymbol{\beta}) + (I-H)\boldsymbol{\varepsilon} = (I-H)\boldsymbol{\varepsilon}$$

Note: we can't "solve" $\boldsymbol{\varepsilon}$ since $(I-H)$ is not invertible since it is not full rank: recall H and $(I-H)$ are idempotent, so

$$\text{trace}(H) = p + 1 = \text{rank}\, X$$

$$\text{trace}(I-H) = n - (p+1) < n$$

Similarly, this does not imply $\boldsymbol{Y} = \boldsymbol{\varepsilon}$. So, $e_i = \varepsilon_i - \sum_{j=1}^{n} h_{ij}\varepsilon_j$ which means e_i is a good approximation to ε when entries of h_{ij} of H are small (which is "usually" the case, especially when n is large).

$$e_i \sim \mathcal{N}(0, \sigma^2(1-h_{ii})) \iff \frac{e_i - 0}{\sigma\sqrt{1-h_{ii}}} \sim \mathcal{N}(0,1)$$

If we plug in $\hat{\sigma}$, that defines the studentized residuals.

$$d_i \equiv \frac{e_i}{\hat{\sigma}\sqrt{1-h_{ii}}}$$

Common practice is to use $\boldsymbol{e}$ for the following residual plots/diagnostics, (using d_i is also possible) to check model assumptions:

Plot $\boldsymbol{e}$ versus $\hat{\boldsymbol{\mu}}$ which was shown in A2 were mutually independent since they are multivariate normal with covariance 0.

Typical "good" scatter plot will have a random scatter around $y = 0$ (no visible patterns).

Problematic scatter plot is when variance of e_i is not constant (cone) increases with fitted values.

In general, plot of $\boldsymbol{e}$ and $\hat{\boldsymbol{\mu}}$ can show deviations from independence, constant variance if those assumptions are violated.

Plot $\boldsymbol{e}$ versus $\boldsymbol{x}_j$ for each $j = 1, \ldots, p$ in model when not many predictors. This can help detect non-linearity between $\boldsymbol{x}_j$ and $\boldsymbol{y}$, not as practical when p is large.

Typical "good" scatter plot will have a random scatter around $y = 0$ (no visible patterns).

If the observation numbers were collected in some order (time, space, etc.), also plot e_i versus indices i to check for any patterns (again, look for random scatter)

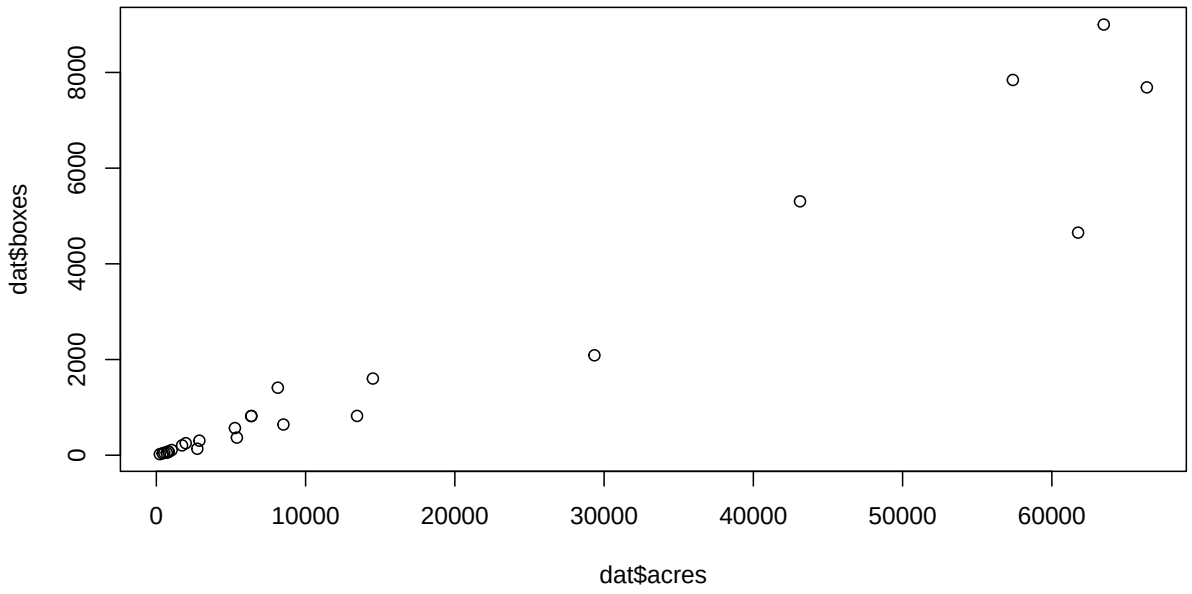
Histogram of $\boldsymbol{e}$: is it bell shaped and symmetric to assess Normal assumption, but the histogram can't easily detect overly fat/thing tails.

Q-Q plot of $\boldsymbol{e}$ more formally assess Normality: scatter plot of ordered quantiles from 2 distributions. In our case: empirical quantiles from residuals (data) versus theoretical quantiles from assumed Normal distribution. If quantiles roughly match, the points will roughly fall on the 45° line through origin.

If these sets of residuals plots show violations, then that could affect the validity of confidence intervals, hypothesis tests, etc.

2.15.1 R Demo

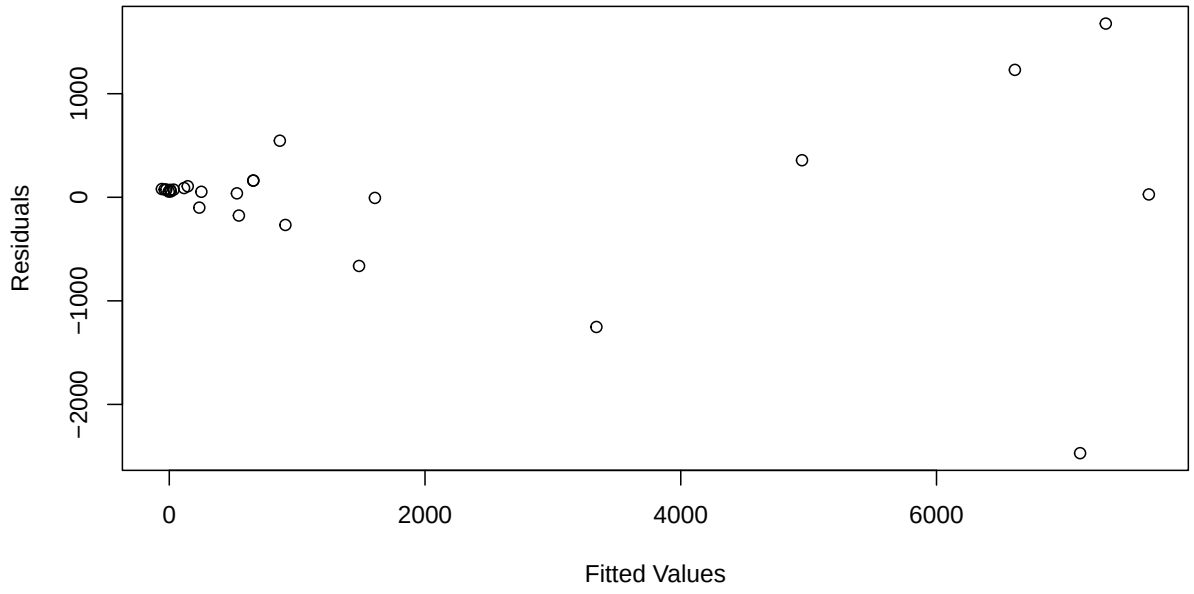
```
### Residual plots/diagnostics demo.
## Florida oranges revisited.
dat <- read.csv("csv/florange.csv")
plot(dat$acres, dat$boxes)
```



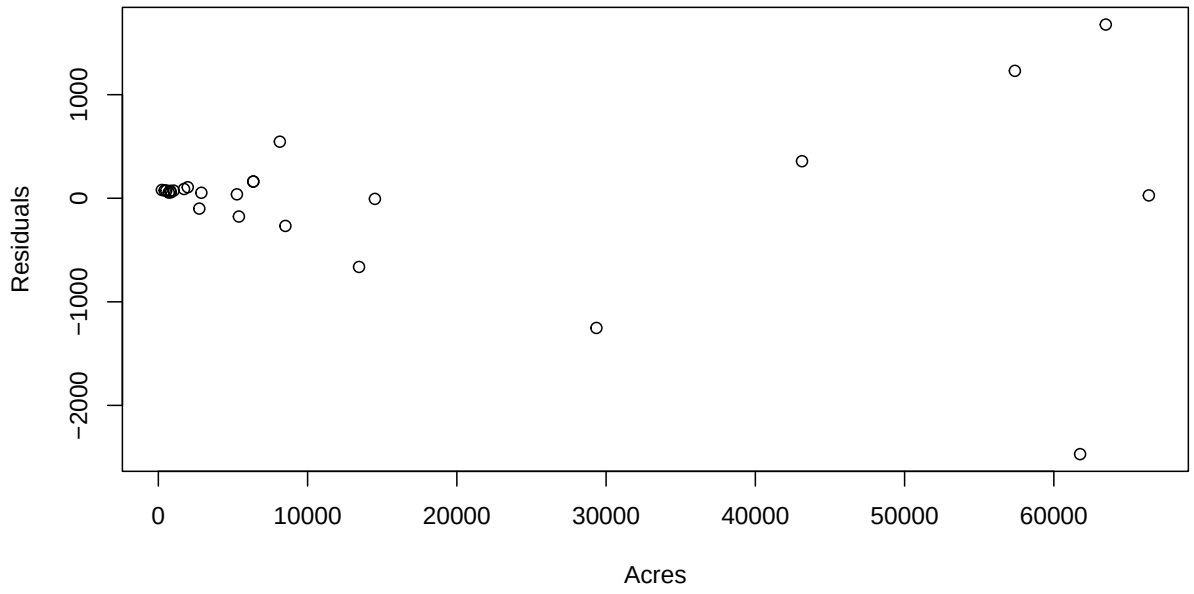

```
lm.1 <- lm(dat$boxes ~ dat$acres)
summary(lm.1)
```

```
##
## Call:
## lm(formula = dat$boxes ~ dat$acres)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2470.81    -6.17    71.72   106.46  1677.32
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -85.391989 186.178031  -0.459    0.651
## dat$acres     0.116717   0.006761  17.263 1.16e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 754.4 on 23 degrees of freedom
## Multiple R-squared:  0.9284,Adjusted R-squared:  0.9252
## F-statistic:   298 on 1 and 23 DF,  p-value: 1.164e-14
```

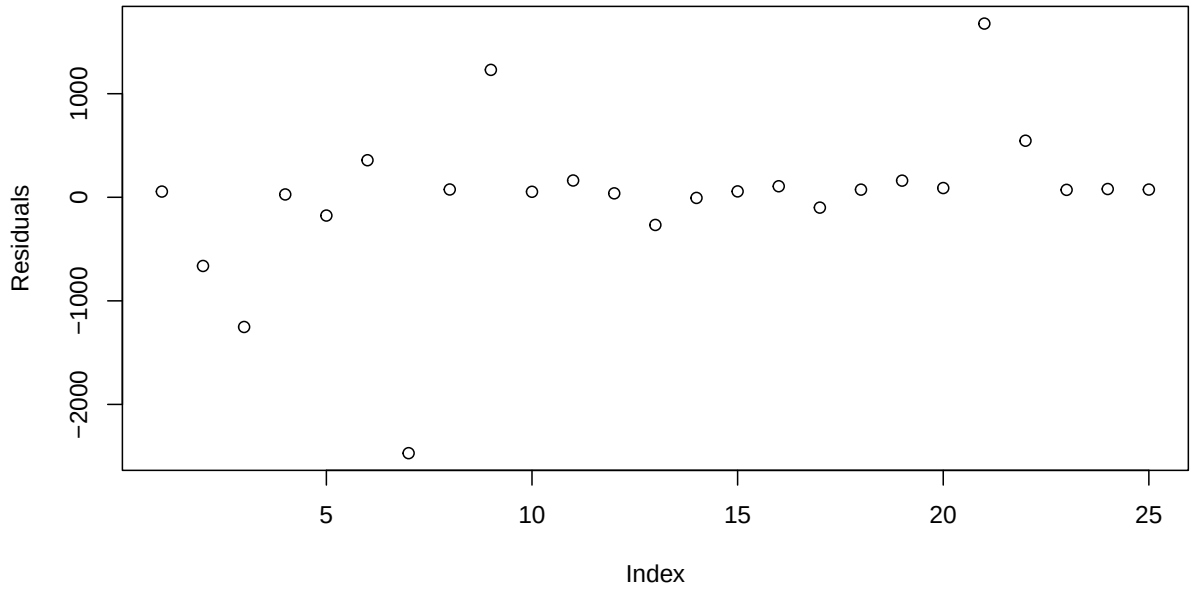
```
# Residual plot: vs fitted values.
plot(lm.1$fitted.values, lm.1$residuals, xlab = "Fitted Values",
  ylab = "Residuals")
```



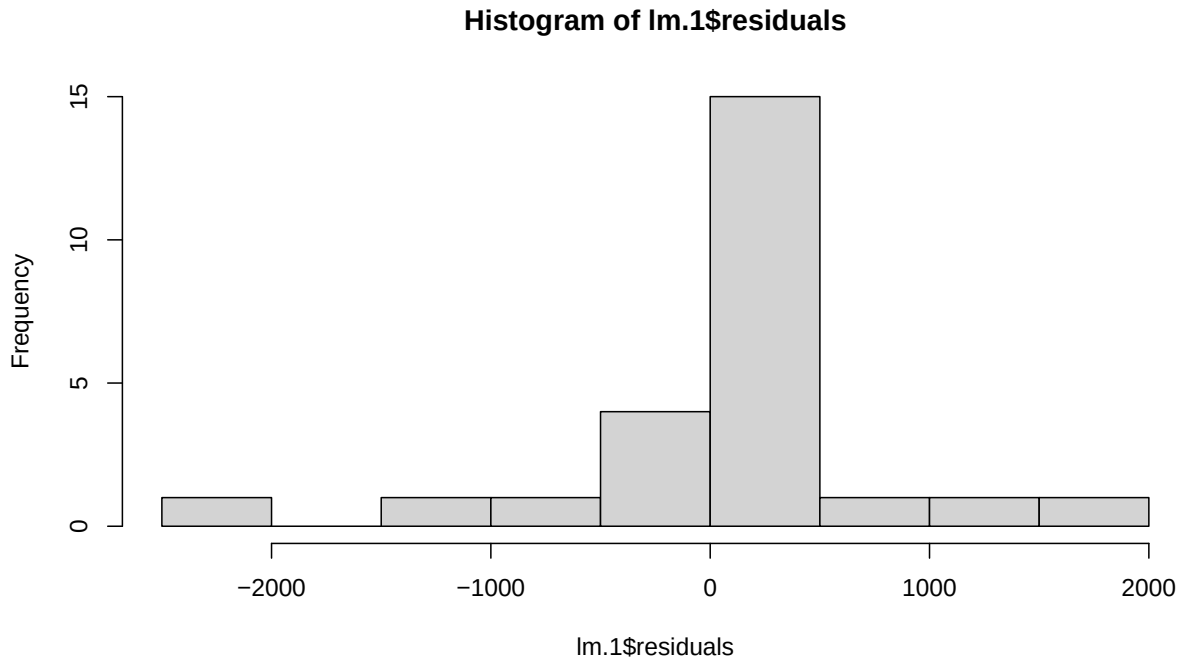

```
# Residual plot: vs predictor (just one in this case).
plot(dat$acres, lm.1$residuals, xlab = "Acres", ylab = "Residuals")
```



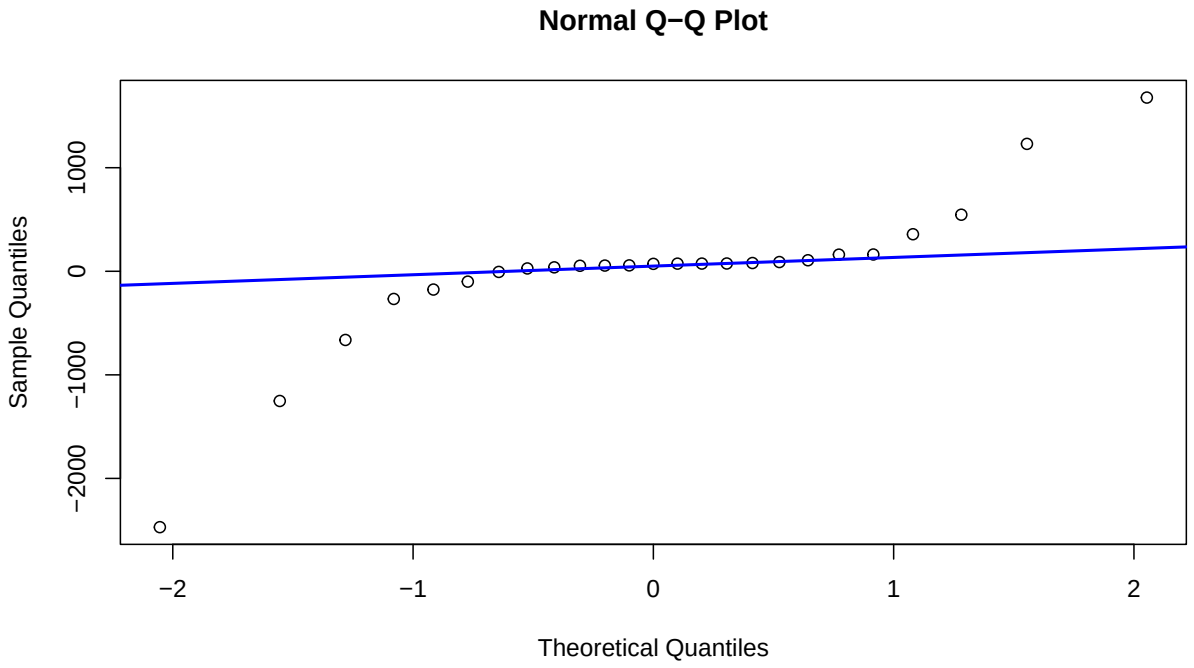

```
# Residual plot: vs i (just to demo plot; no time/space
# ordering here).
plot(1:nrow(dat), lm.1$residuals, xlab = "Index", ylab = "Residuals")
```




```r
# Histogram of residuals.
hist(lm.1$residuals)
```



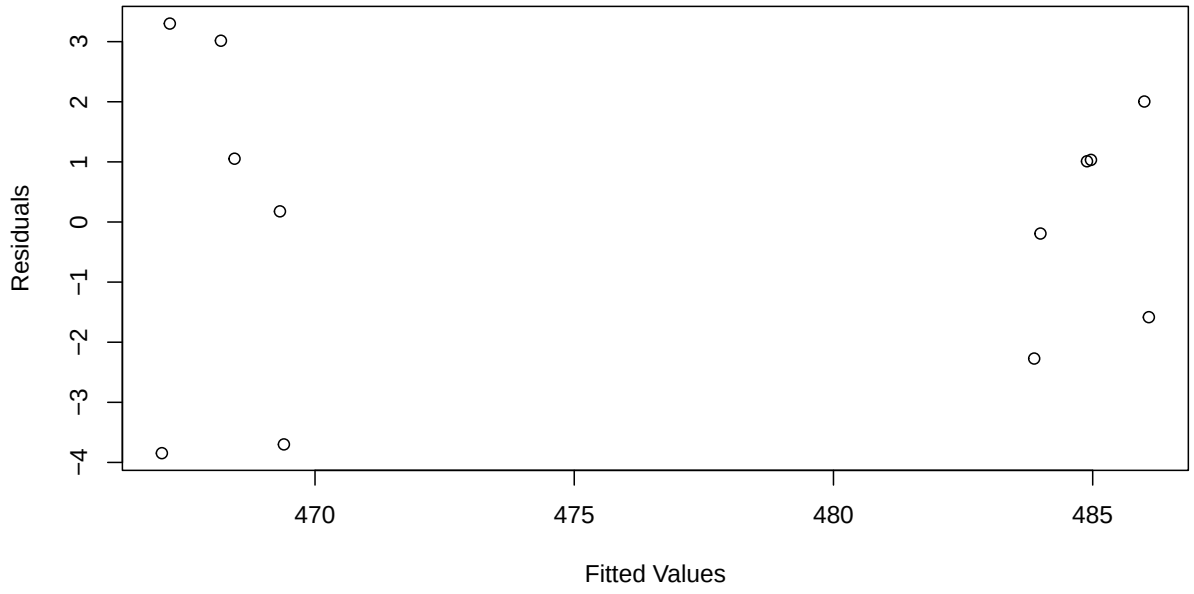

```r
# QQ plot of residuals.
qqnorm(lm.1$residuals)
qqline(lm.1$residuals, col = "blue", lwd = 2)
```



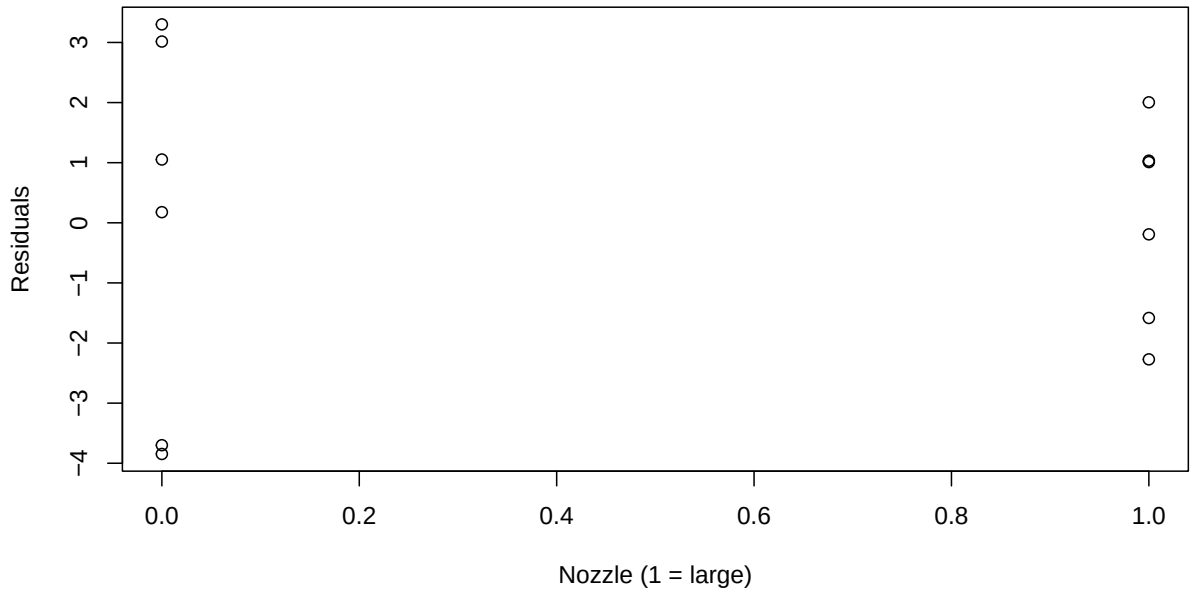

```r
## Rocket data revisited.
rocket <- read.csv("csv/rocket.csv")
mr <- lm(thrust ~ nozzle + propratio, data = rocket)
summary(mr)
```

```
##
## Call:
## lm(formula = thrust ~ nozzle + propratio, data = rocket)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.8459 -1.7555  0.5934  1.2906  3.3008
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 473.6039     4.7158 100.430 4.88e-15 ***
## nozzle       16.7383     1.5329  10.919 1.71e-06 ***
## propratio    -1.0948     0.9414  -1.163    0.275
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.655 on 9 degrees of freedom
## Multiple R-squared:  0.9303,Adjusted R-squared:  0.9148
## F-statistic: 60.05 on 2 and 9 DF,  p-value: 6.238e-06
```

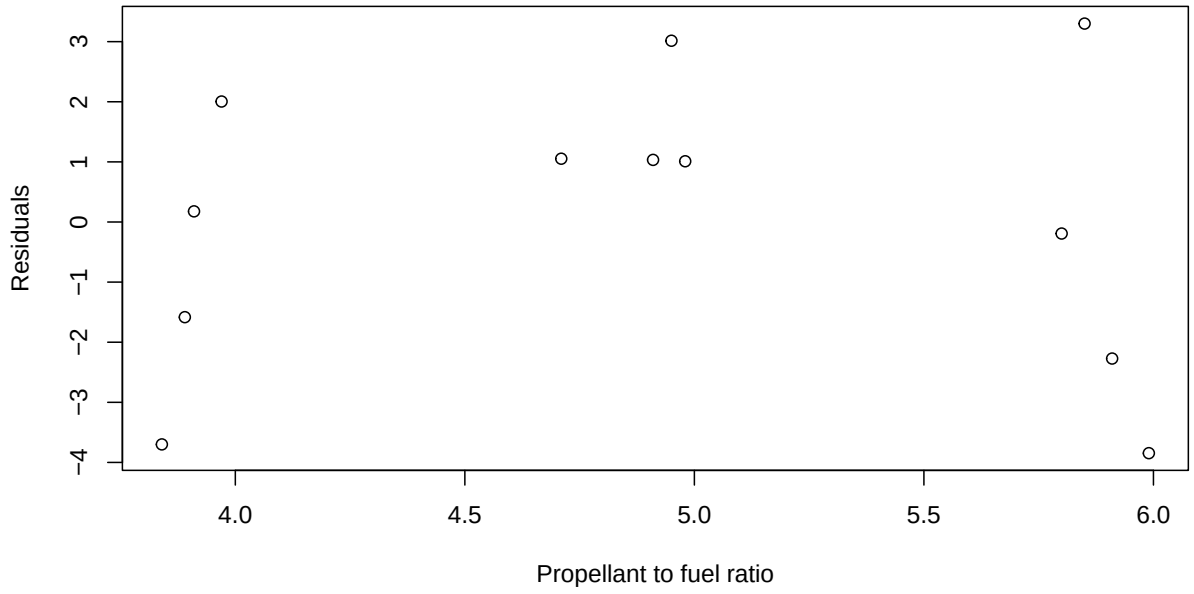
```r
# Residual plot: vs fitted values.
plot(mr$fitted.values, mr$residuals, xlab = "Fitted Values",
  ylab = "Residuals")
```



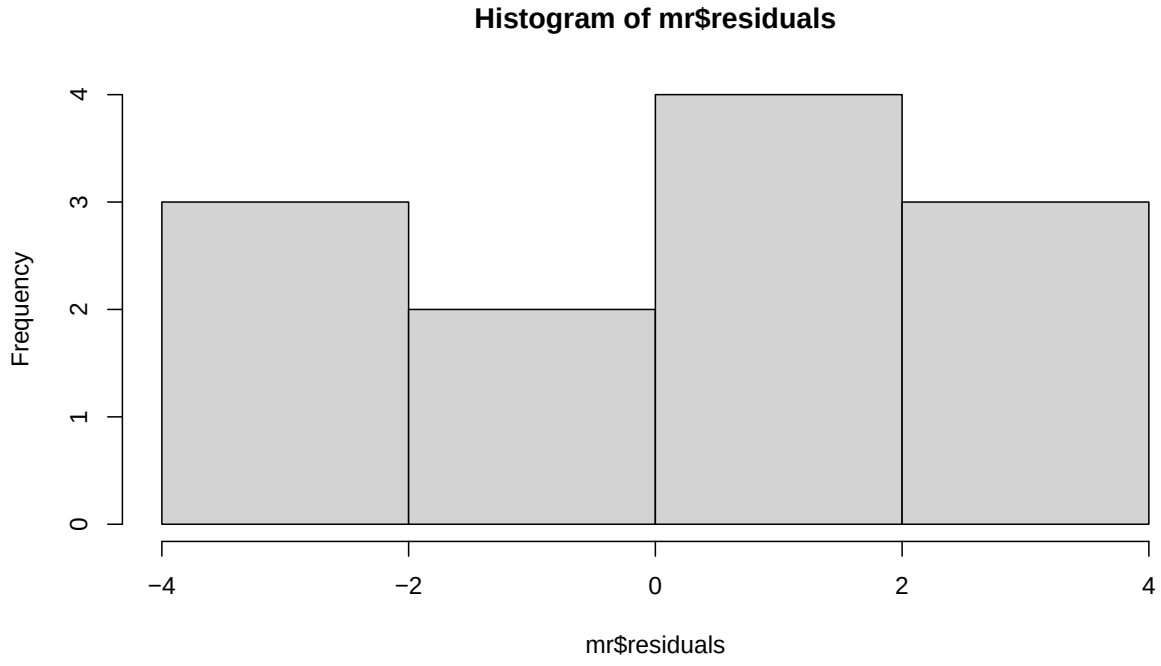

```r
# Residual plot: vs predictors.
plot(rocket$nozzle, mr$residuals, xlab = "Nozzle (1 = large)",
  ylab = "Residuals")
```



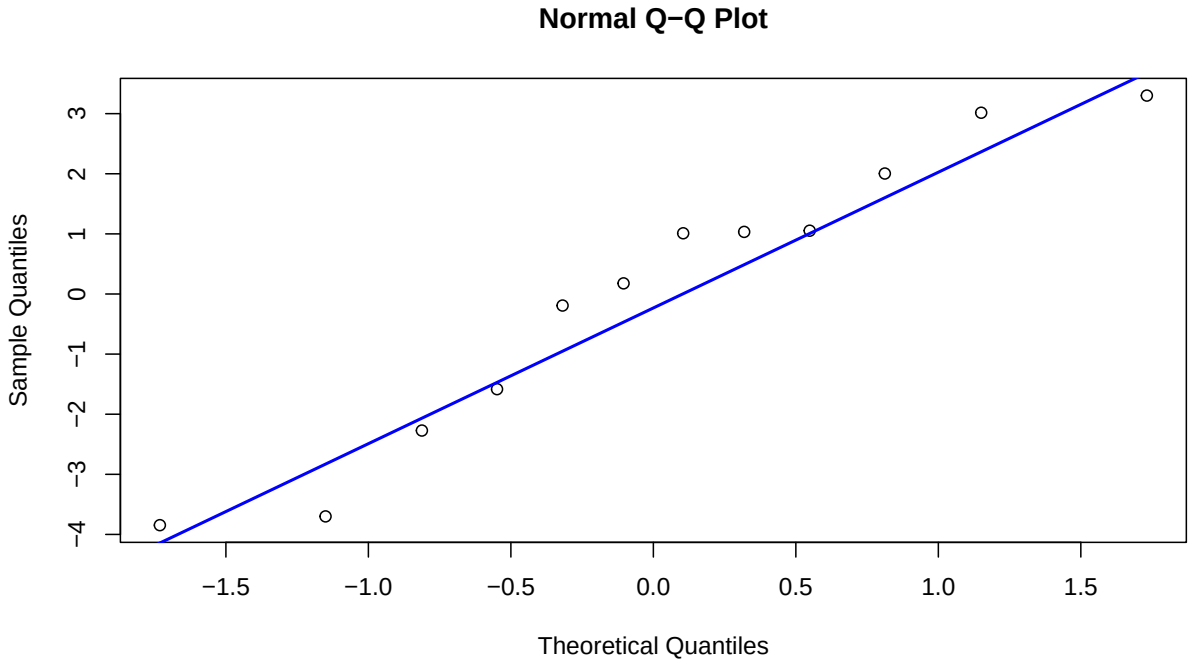

```r
plot(rocket$propratio, mr$residuals, xlab = "Propellant to fuel ratio",
  ylab = "Residuals")
```




```
# Histogram of residuals,
hist(mr$residuals)
```




```
# QQ plot of residuals,
qqnorm(mr$residuals)
qqline(mr$residuals, col = "blue", lwd = 2)
```



LECTURE 16 | 2020-11-04

2.16 Addressing Problems With Regression Model Assumptions

If residual plots reveal problems with assumptions (although plots don't fully check linearity, independence), we might be able to address via transformations, adding variables to model, using different error distribution on ε.

(1) Variance-stabilizing transformations on responses y_i, can help constant variance identified on $\boldsymbol{e}$ vs $\hat{\boldsymbol{\mu}}$ plot.

Idea: apply function $g(\cdot)$ and fit regression on transformed $g(y_i)$; that is,

$$g(Y_i) = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip} + \varepsilon_i$$

Note this can drastically change SS(Res) and $\hat{\sigma}$, so cannot directly use those among different choices $g(\cdot)$.

Rationale: variance of response might be a function of mean $\mu_i = \mathbb{E}[Y_i]$; could be expressed

$$\mathbb{V}(Y_i) = \mathbb{V}(\varepsilon_i) = h(\mu_i)\sigma^2$$

for some $h(\cdot) > 0$. In which case, we want

$$\mathbb{V}(g(Y_i)) \approx \sigma^2$$

1st order Taylor expansion

$$g(Y_i) \approx g(\mu_i) + (Y_i - \mu_i)g'(\mu_i)$$
$$\mathbb{V}(g(Y_i)) \approx [g'(\mu_i)]^2 \mathbb{V}(Y_i)$$

Thus, for $\mathbb{V}(g(Y_i))$ to be constant, we need

$$[g'(\mu_i)]^2 \propto \frac{1}{h(\mu_i)}$$

Examples:

(i) $h(\mu_i) = \mu_i$; that is, $\mathbb{V}(Y_i) = \sigma^2\mu_i \propto \mu_i$. Variance in responses proportional to mean response. We need

$$g'(\mu_i) \propto \frac{1}{\sqrt{h(\mu_i)}} = \frac{1}{\sqrt{\mu_i}}$$

and so $g(\mu_i) = \sqrt{\mu_i}$ works, and we apply $g(y_i) = \sqrt{y_i}$ to obtain approximately constant variance

(ii) $h(\mu_i) = \mu_i^2$; that is, $\mathbb{V}(Y_i) = \sigma^2\mu_i^2 \propto \mu_i^2$ or $\text{Sd}(Y_i) \propto \mu_i$. We need

$$g'(\mu_i) \propto \frac{1}{\mu_i}$$

and so $g(\mu_i) = \ln(\mu_i)$ works, and we apply $g(y_i) = \ln(y_i)$ to stabilize variance.

(iii) Class of power transformations (Box-Cox)

$$g(y_i) = \begin{cases} \dfrac{y_i^\lambda - 1}{\lambda} & \lambda \neq 0 \\ \ln(y_i) & \lambda = 0 \end{cases}$$

$$g'(\mu_i) = \begin{cases} \mu_i^{\lambda-1} & \lambda \neq 0 \\ \dfrac{1}{\mu_i} & \lambda = 0 \end{cases} \iff h(\mu_i) \propto \frac{1}{[g'(\mu_i)]^2} = \mu_i^C \quad C \in \mathbf{R}$$

Box-Cox transformation can help address non-constant variance of the form

$$\mu_i^C\sigma^2 = \mathbb{V}(Y_i)$$

Special cases include:

- $\lambda = \dfrac{1}{2}$ is $\sqrt{\cdot}$
- $\lambda = 0$ is $\ln(\cdot)$
- $\lambda = 1$ is identity
- $\lambda = -1$ is reciprocal

can automatically try a sequence of λ and find the choice that gives the best value of likelihood

Note that interpreting $\hat{\beta}_j$ can be less intuitive as a result of transformation, since now increasing x_j by 1 unit corresponds to an estimated change of $\hat{\beta}_j$ in $g(y_i)$. For $g(y_i) = \ln(y_i)$, $\hat{\beta}_j$ represents estimate of expected change in $\ln(y_i)$ which corresponds to $e^{\hat{\beta}_j}$ being the expected multiplicative change applied to the (original) response. But for an arbitrary λ, the transformation might be less interpretable.

(2) Transforming and/or adding explanatory variables.

- If y (or $g(y)$) has a clear non-linear relationship with some x_j, we can consider transforming $\boldsymbol{x}_j$ (e.g., $\ln(\cdot)$, $\sqrt{\cdot}$, etc.)
- Could add polynomial terms (e.g., $x^2, x^3, ...$). For example,

$$Y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \varepsilon_i$$

suppose we think adding $\boldsymbol{x}_1^2$ is appropriate, then we define $x_{i3} = x_{i1}^2$ and fit

$$Y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \beta_3 x_{i3} + \varepsilon_i$$

which is still <u>linear</u> in β and note that $\boldsymbol{x}_1$ and $\boldsymbol{x}_1^2$ are linearly independent.

- Add interaction terms: if we think the effect of $\boldsymbol{x}_i$ on response depends on the value of $\boldsymbol{x}_j$, e.g., suppose we think $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ interact, then we might fit

$$Y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \beta_3 x_{i3} + \varepsilon_i$$

where $x_{i3} = x_{i1} + x_{i2}$, so that

$$Y_i = \beta_0 + (\beta_1 + \beta_3 x_{i2}) x_{i1} + \beta_2 x_{i2} + \varepsilon_i$$

Note: in general, there's $\binom{p}{2}$ possible interactions, consider whether interactions are conceptually plausible.

(3) Q-Q plot of residuals not normal (even after any appropriate transformations)

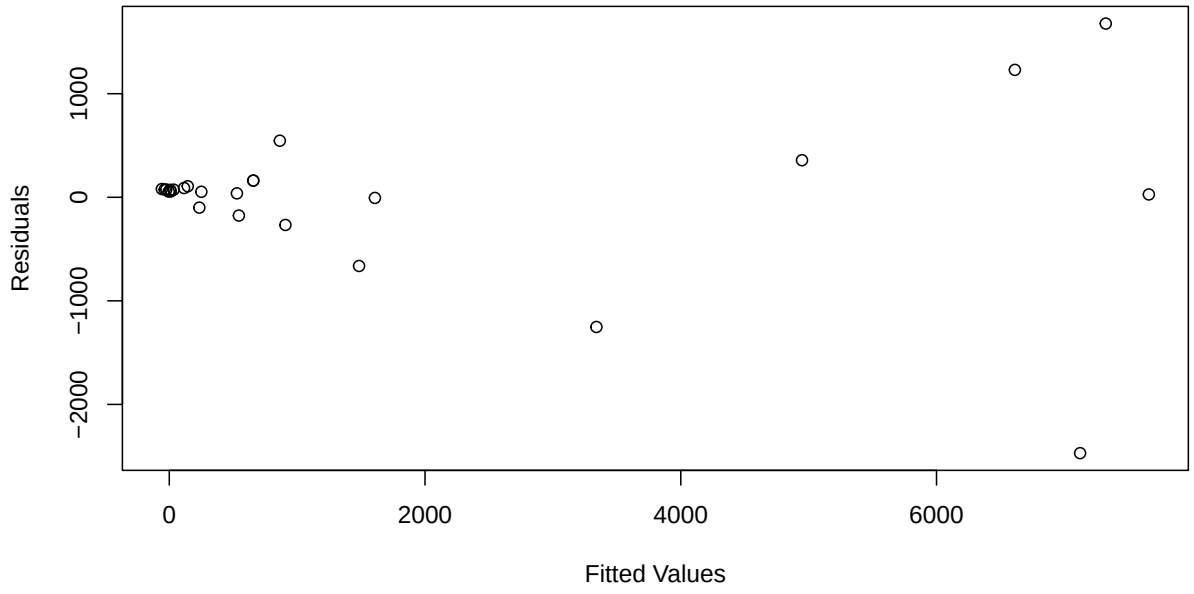
- Consider using different distribution for error term ε (e.g., t, Cauchy, Laplace, etc.)

2.16.1 R Demo

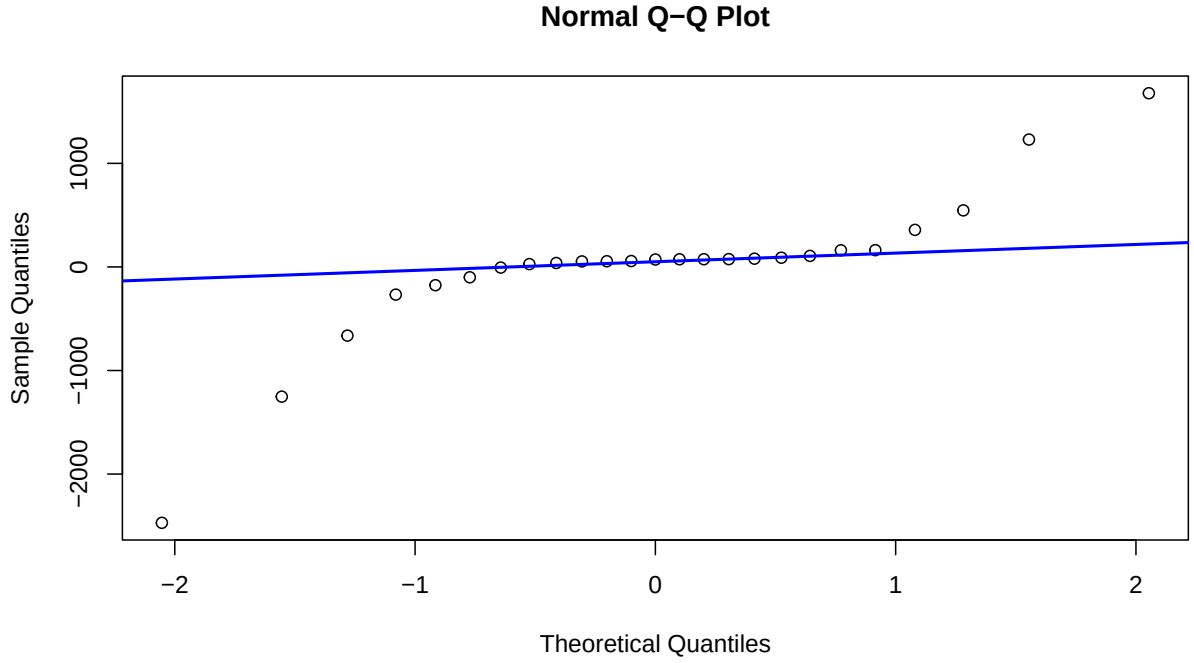
```
library(MASS)
## Demo for transformations and interactions
## Florida oranges revisited
dat <- read.csv("csv/florange.csv")
lm.1 <- lm(dat$boxes ~ dat$acres)
summary(lm.1)
```

```
##
## Call:
## lm(formula = dat$boxes ~ dat$acres)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2470.81    -6.17    71.72   106.46  1677.32
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -85.391989 186.178031  -0.459    0.651
## dat$acres     0.116717   0.006761  17.263 1.16e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 754.4 on 23 degrees of freedom
## Multiple R-squared:  0.9284,Adjusted R-squared:  0.9252
## F-statistic:   298 on 1 and 23 DF,  p-value: 1.164e-14
```

```
# Recall: residuals had non-constant variance (variance
# increases with fitted values)
plot(lm.1$fitted.values, lm.1$residuals, xlab = "Fitted Values",
  ylab = "Residuals")
```



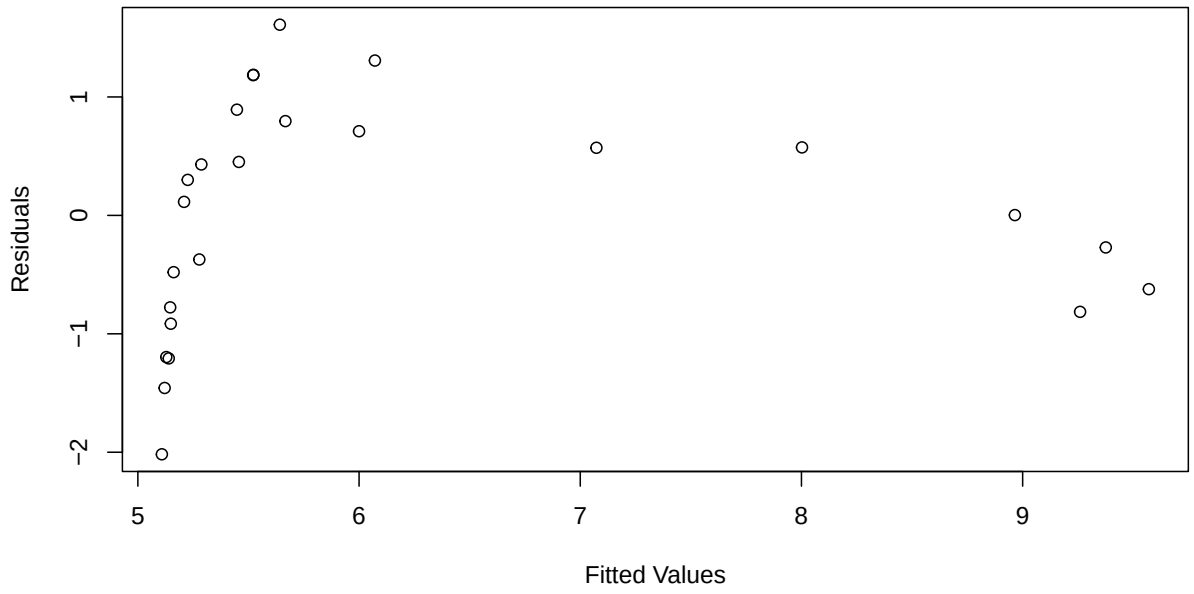

```
qqnorm(lm.1$residuals)
qqline(lm.1$residuals, col = "blue", lwd = 2)
```



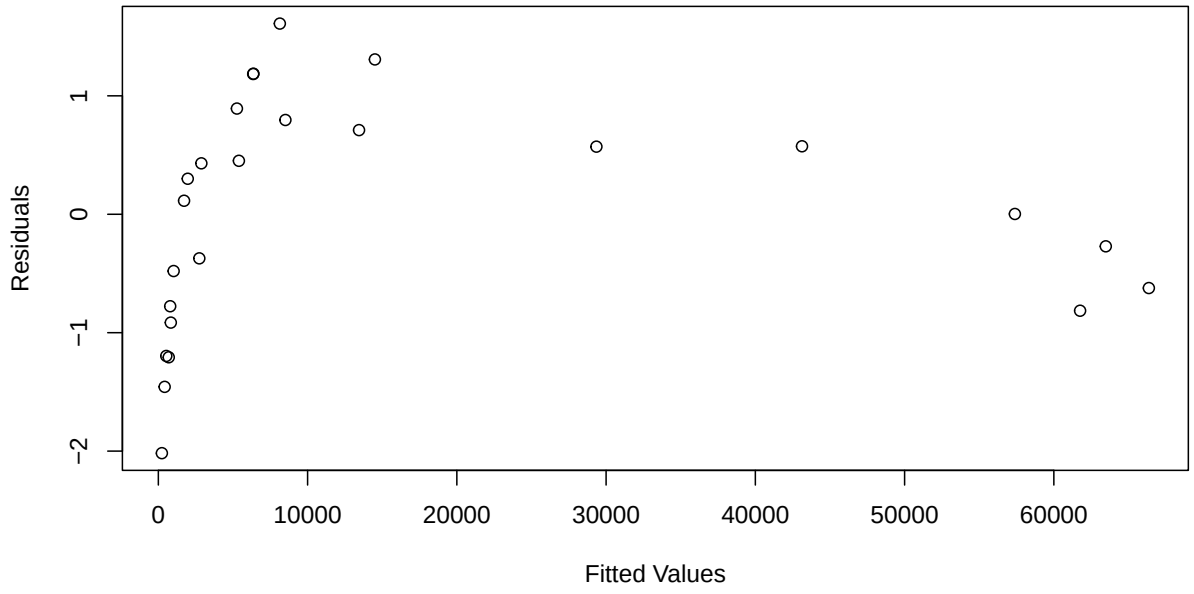

```
# Try log-transforming y
lm.log <- lm(log(dat$boxes) ~ dat$acres)
summary(lm.log)
```

```
##
## Call:
## lm(formula = log(dat$boxes) ~ dat$acres)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.0175 -0.7767  0.1142  0.7106  1.6102
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 5.093e+00  2.425e-01  20.997  < 2e-16 ***
## dat$acres   6.748e-05  8.808e-06   7.661 8.95e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9828 on 23 degrees of freedom
## Multiple R-squared:  0.7184,Adjusted R-squared:  0.7062
## F-statistic: 58.69 on 1 and 23 DF,  p-value: 8.948e-08
```

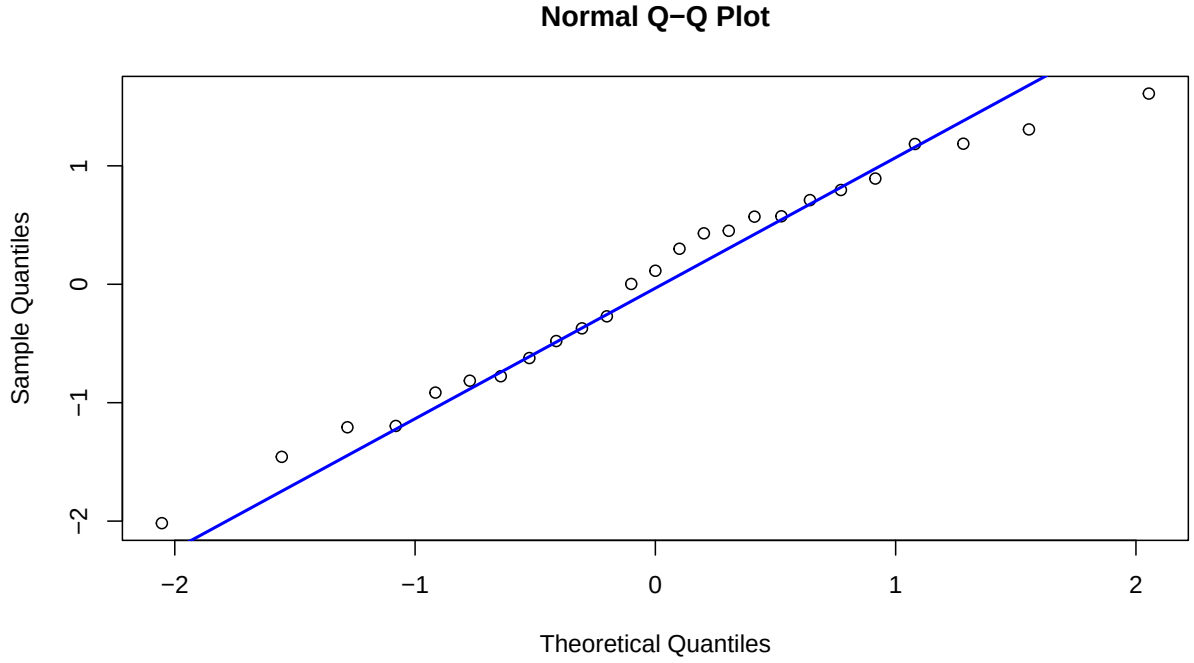
```
plot(lm.log$fitted.values, lm.log$residuals, xlab = "Fitted Values",
  ylab = "Residuals")
```



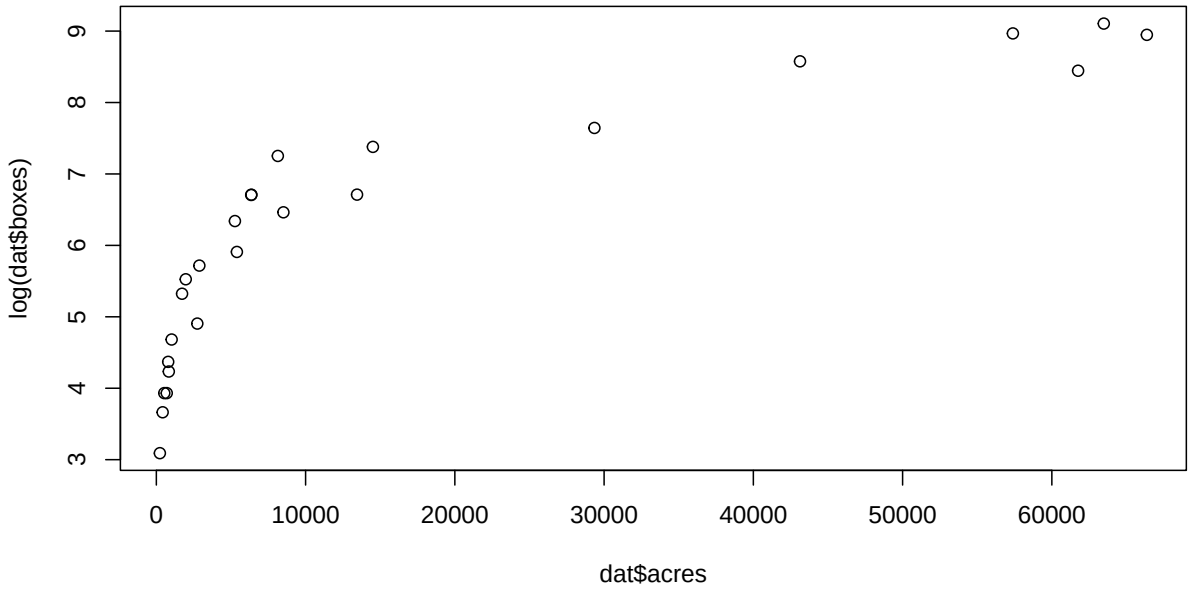

```
plot(dat$acres, lm.log$residuals, xlab = "Fitted Values", ylab = "Residuals")
```



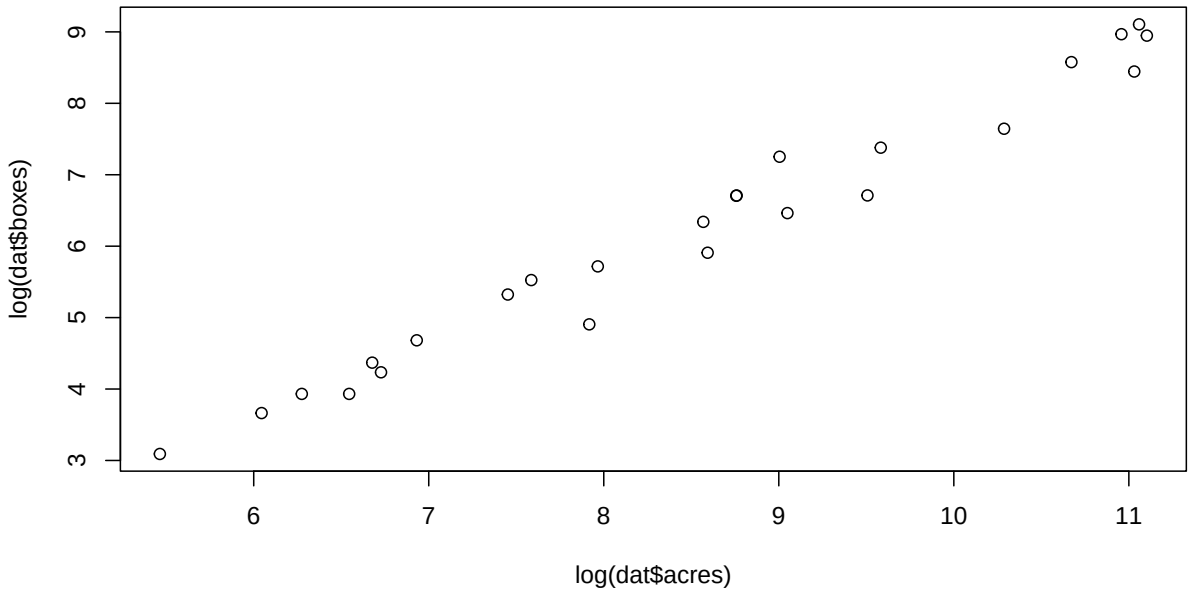

```
qqnorm(lm.log$residuals)
qqline(lm.log$residuals, col = "blue", lwd = 2)
```



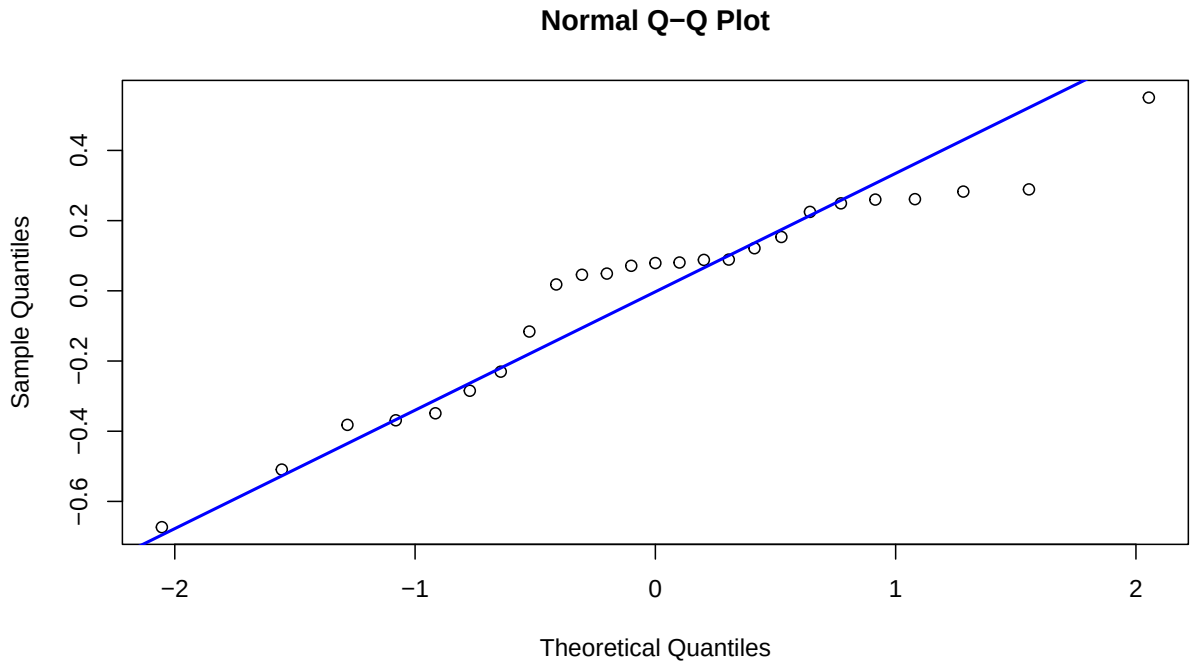

```
# Does the plot of residuals vs x suggest a problem Let's
# take a closer look
plot(dat$acres, log(dat$boxes))  # evidently not linear!
```



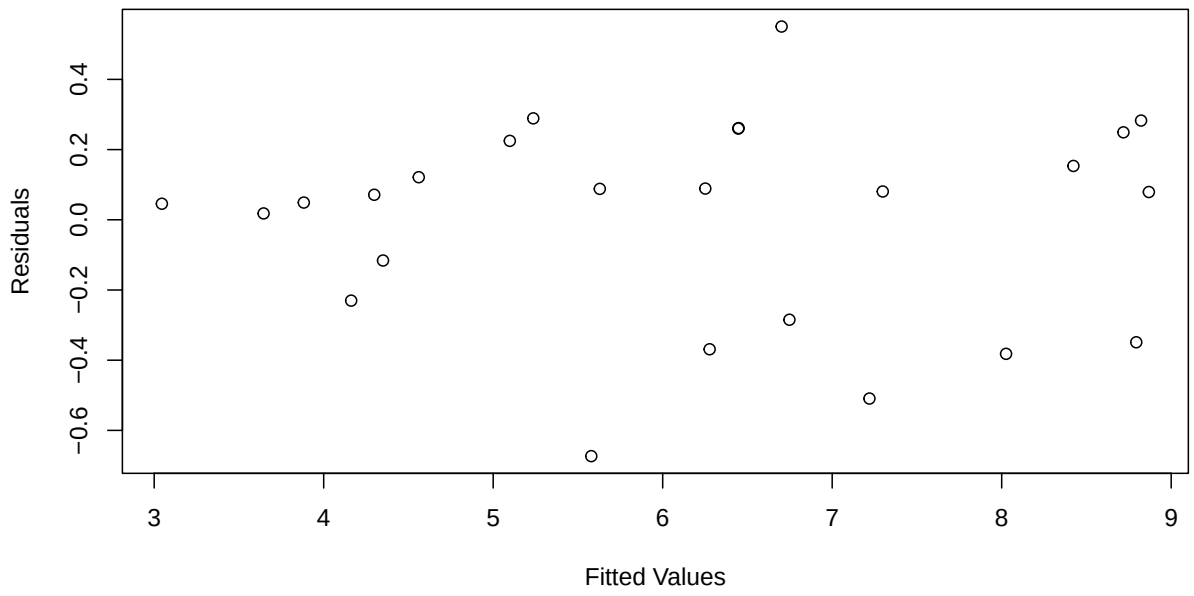

```r
# Log-transform x as well
plot(log(dat$acres), log(dat$boxes))  # looks much more linear!
```



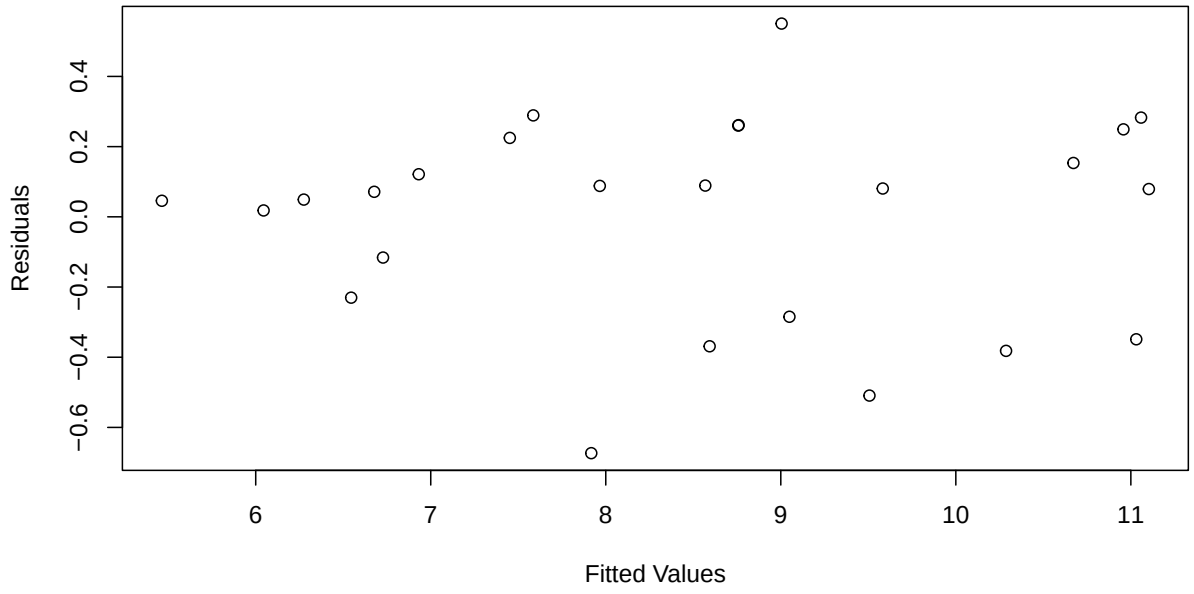

```r
lm.loglog <- lm(log(dat$boxes) ~ log(dat$acres))
qqnorm(lm.loglog$residuals)
qqline(lm.loglog$residuals, col = "blue", lwd = 2)
```




```
plot(lm.loglog$fitted.values, lm.loglog$residuals, xlab = "Fitted Values",
  ylab = "Residuals")
```



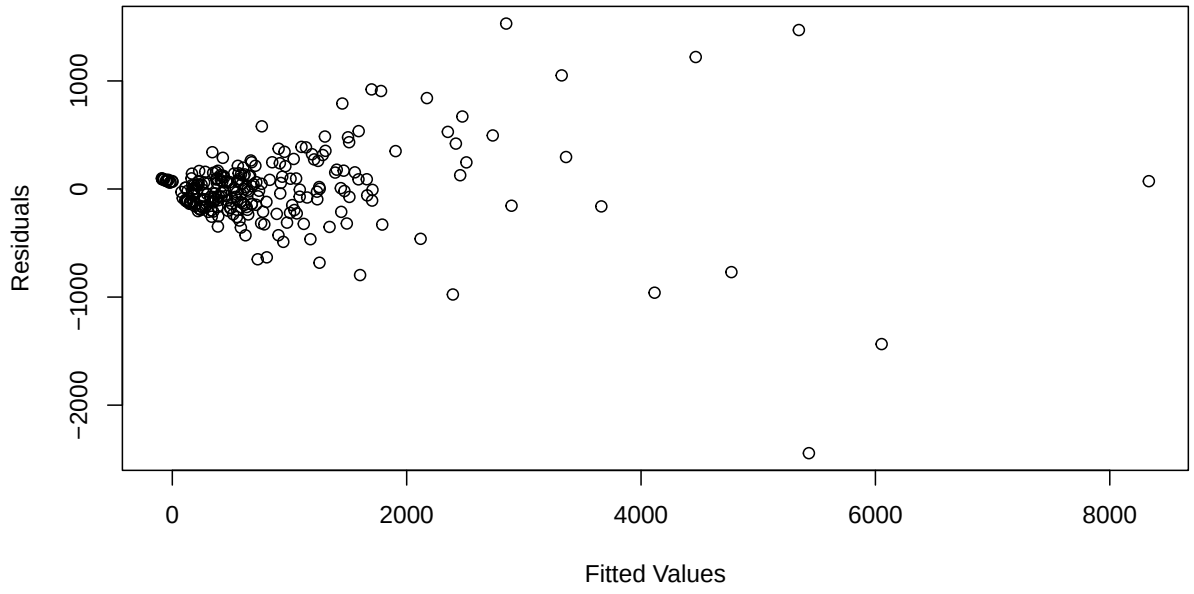

```
plot(log(dat$acres), lm.loglog$residuals, xlab = "Fitted Values",
  ylab = "Residuals")
```



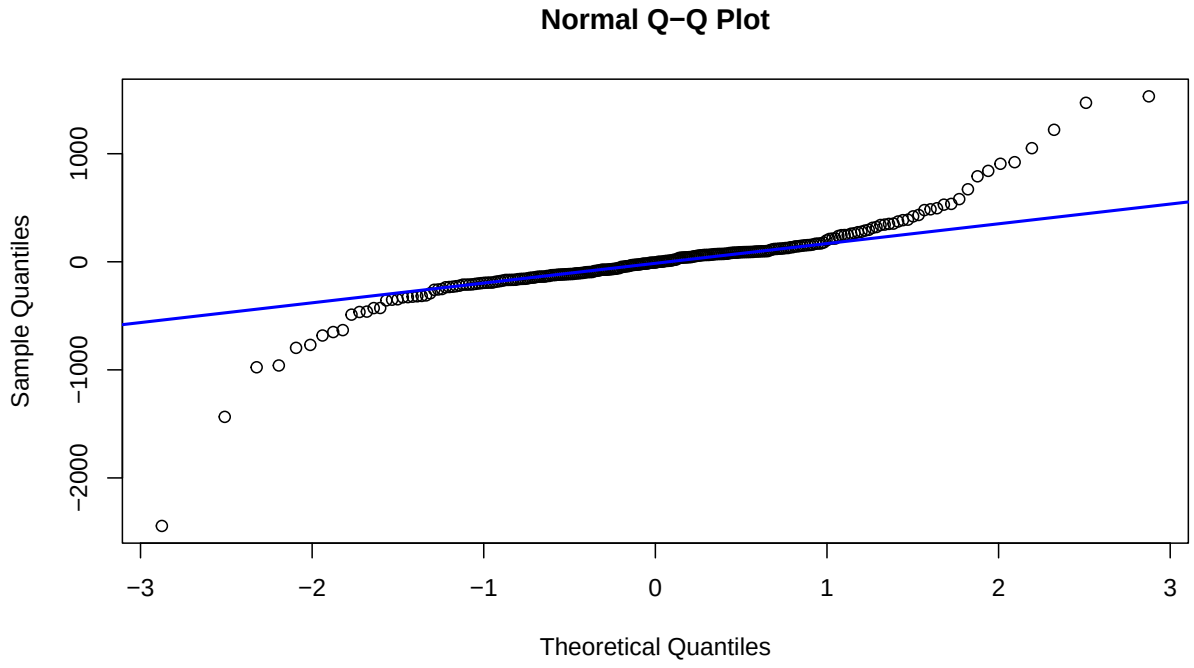

```
## Python data revisited
python <- read.csv("csv/FLpython.csv")
python$male <- ifelse(python$sex == "M", 1, 0)  # 1 = M, 0 =F
mpf2 <- lm(fat ~ male + mass + svl, data = python)
summary(mpf2)
```

```
##
## Call:
## lm(formula = fat ~ male + mass + svl, data = python)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2444.44  -137.38    -6.66   109.22  1530.81
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  204.09840  132.30121   1.543   0.1242
## male        -196.71705   47.16396  -4.171 4.22e-05 ***
## mass           0.11788    0.00524  22.495  < 2e-16 ***
## svl           -1.59841    0.76433  -2.091   0.0375 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 360.2 on 244 degrees of freedom
## Multiple R-squared:  0.897,Adjusted R-squared:  0.8957
## F-statistic: 708.2 on 3 and 244 DF,  p-value: < 2.2e-16
```

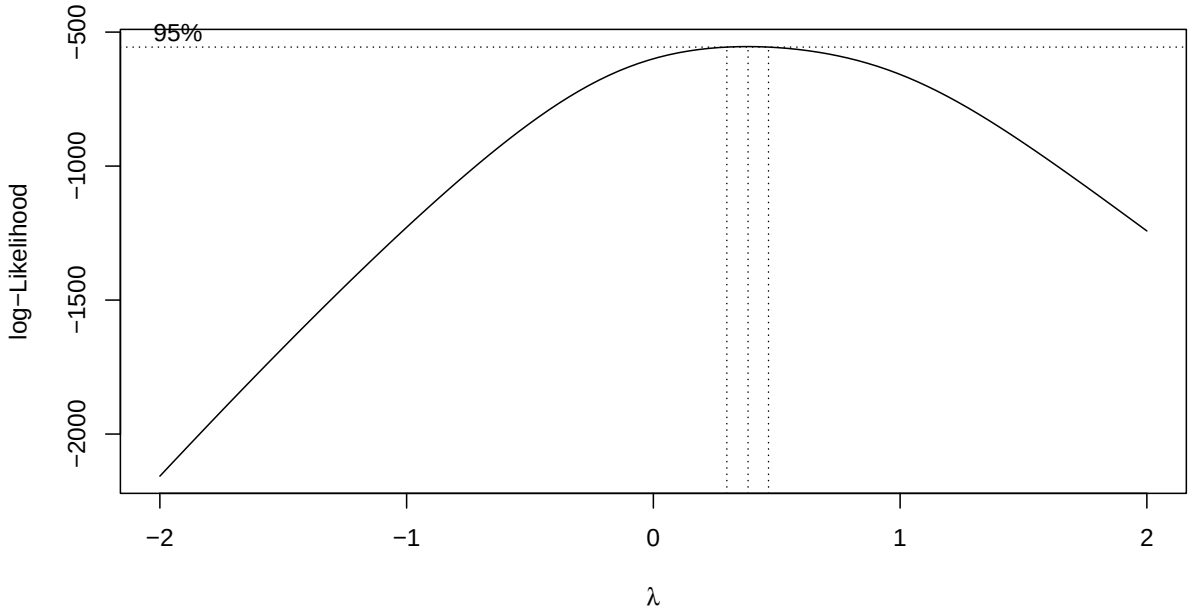
```
# Residual plot: vs fitted values
plot(mpf2$fitted.values, mpf2$residuals, xlab = "Fitted Values",
  ylab = "Residuals")
```




```
## QQ plot of residuals
qqnorm(mpf2$residuals)
qqline(mpf2$residuals, col = "blue", lwd = 2)
```



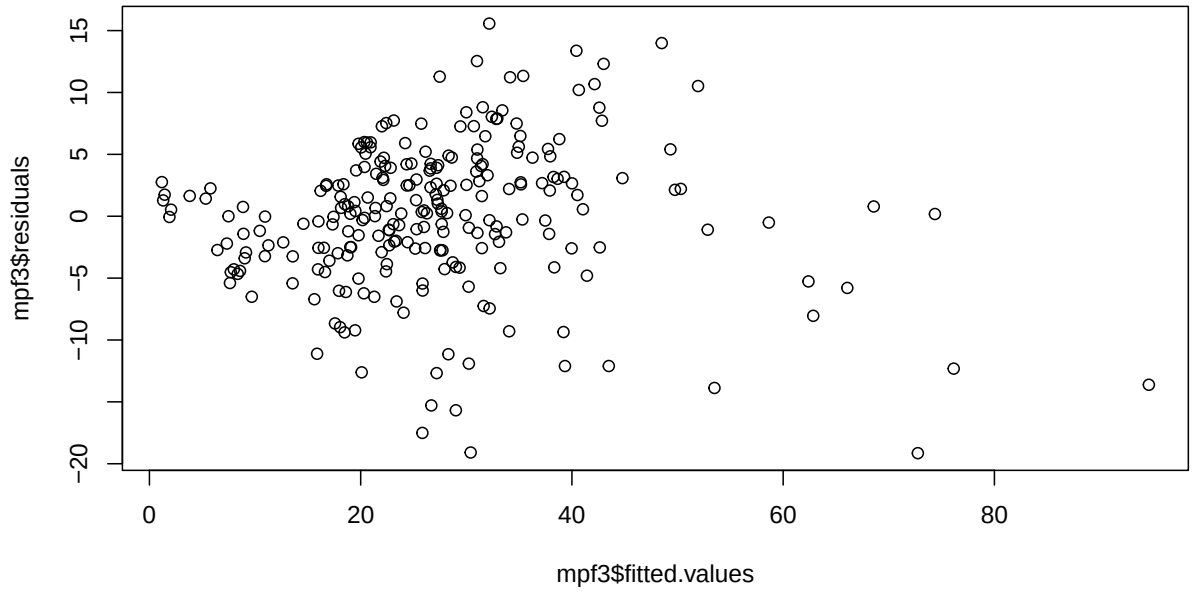

```
# Try a Box-Cox transformation
bc <- boxcox(mpf2)
```



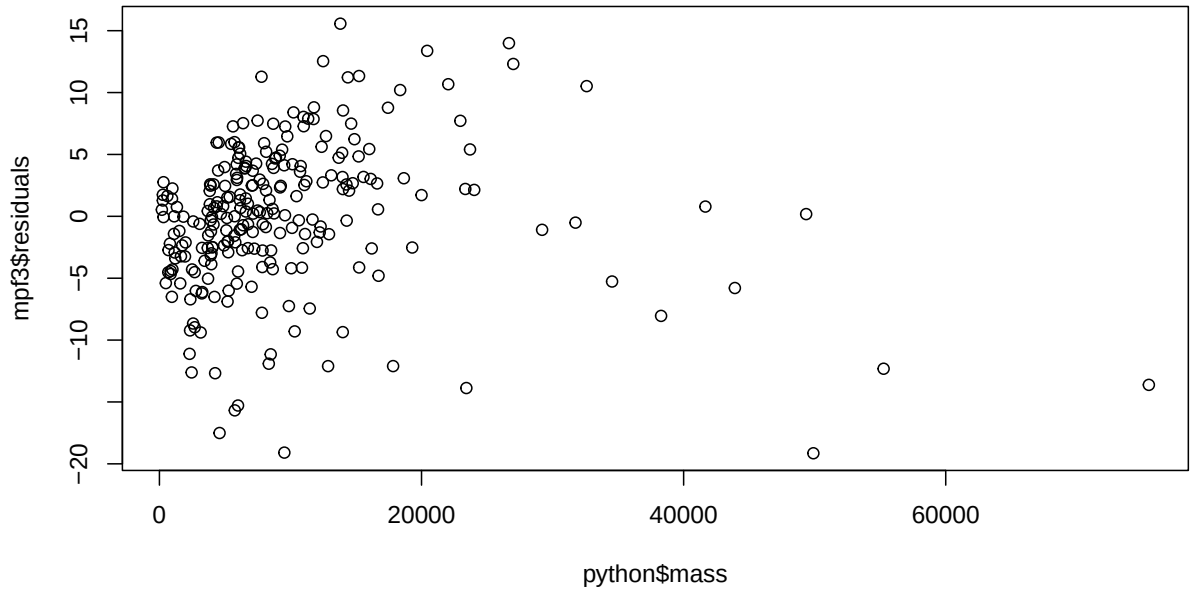

```
lambda <- bc$x[which.max(bc$y)]
mpf3 <- lm((fat^lambda - 1)/lambda ~ male + mass + svl, data = python)
summary(mpf3)
```

```
##
## Call:
## lm(formula = (fat^lambda - 1)/lambda ~ male + mass + svl, data = python)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -19.146  -2.910   0.297   3.688  15.568
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -8.0558134  2.1813183  -3.693 0.000273 ***
## male        -1.7849310  0.7776166  -2.295 0.022560 *
## mass         0.0004461  0.0000864   5.164 5.03e-07 ***
## svl          0.1431492  0.0126019  11.359  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.939 on 244 degrees of freedom
## Multiple R-squared:  0.8356,Adjusted R-squared:  0.8336
## F-statistic: 413.5 on 3 and 244 DF,  p-value: < 2.2e-16
```

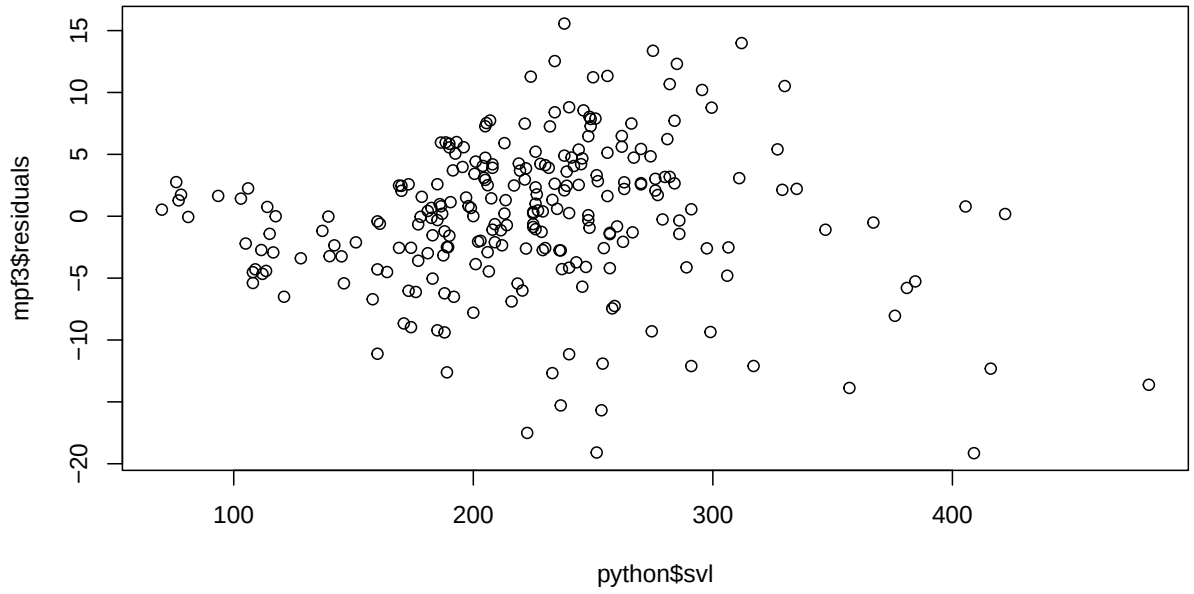
```
plot(mpf3$fitted.values, mpf3$residuals)
```



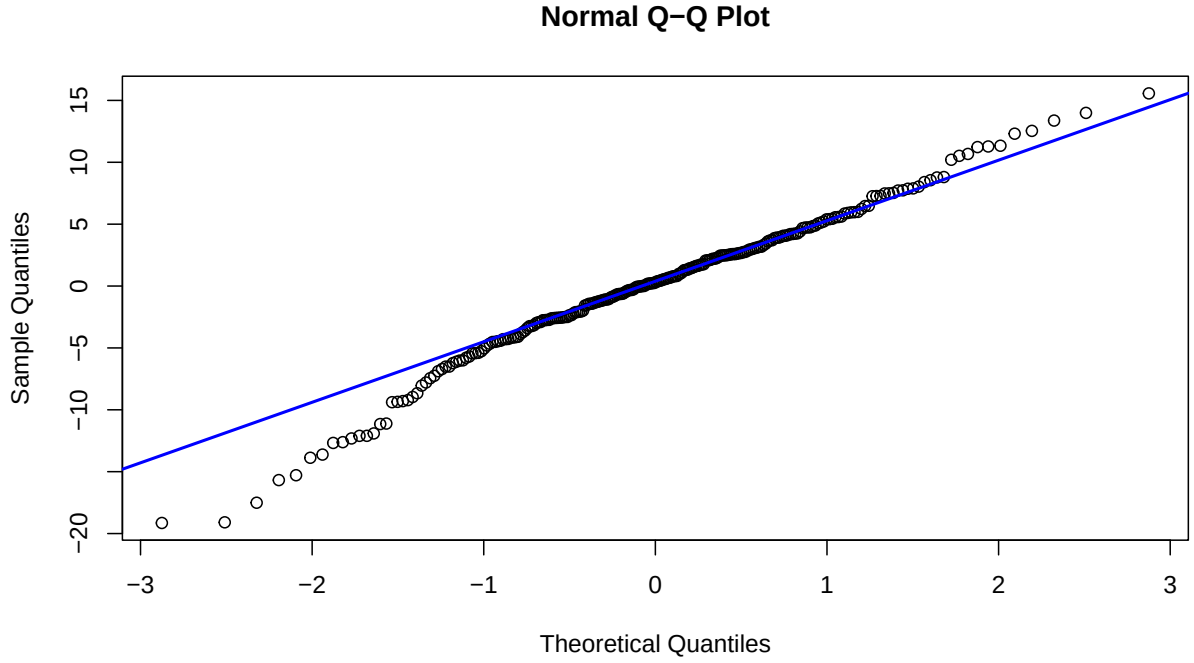

```
plot(python$mass, mpf3$residuals)
```




```
plot(python$svl, mpf3$residuals)
```




```
qqnorm(mpf3$residuals)
qqline(mpf3$residuals, col = "blue", lwd = 2)
```




```
# still some skew, but better!
```

2.17 Effects of Individual Observations

(1) We say an observation $(y_i, x_{i1}, \dots, x_{ip})$ is an outlier if it is substantially different from other observations. This can occur if its response and/or some of its explanatory variables have values that are unusual or extreme compared to the others. Outliers can occur for different reasons. For example, an extraordinary subject, data entry errors. Generally, we don't recommend removing outliers unless we have a strong reason to believe that observation is an error. But it can be useful to investigate what effect it has on our fitted model and quality of fit to the rest of data.

(2) How to detect and characterize outliers: Studentized residual

$$d_i = \frac{e_i}{\hat{\sigma}\sqrt{1-h_{ii}}}$$

are standardized to have approximately variance 1 where

$$e_i \sim \mathcal{N}(0, \sigma^2(1-h_{ii}))$$

So, if $|d_i|$ is large, observation i could be considered an outlier in the sense of having an extreme value of response y_i (e.g., $|d_i| > 3$)

(3) Recall that h_{ii} is the i^{th} diagonal element of hat matrix H. We call h_{ii} the **leverage** of observation i.

$$\hat{\boldsymbol{\mu}} = X\hat{\boldsymbol{\beta}} = H\boldsymbol{Y}$$

so

$$\hat{\mu}_i = [h_{i1} \quad h_{i2} \quad \cdots \quad h_{in}] \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \sum_{j=1}^{n} h_{ij}y_j = h_{ii}y_i + \sum_{j \neq i} h_{ij}y_j$$

so h_{ii} captures contribution of Y_i in determining its corresponding fitted value. If the leverage is large relative to h_{ij}'s then $\hat{\mu}_i$ is mostly determined by Y_i. In A3, you will show the leverage is between $\frac{1}{n}$ and 1; that is,

$$\frac{1}{n} \leq h_{ii} \leq 1$$

If $h_{ii} \approx 1$, then $\mathbb{V}(e_i) = \sigma^2(1-h_{ii}) \approx 0$, which in turn implies $y_i \approx \hat{\mu}_i$; that is, residuals with observations with high leverage tend to be small.

Rule of thumb: an observation with leverage higher than twice the average leverage is considered high.

$$h_{ii} > 2\bar{h}$$

where $\bar{h} = \frac{1}{n}\sum_{i=1}^{n} h_{ii}$.

$$\text{trace}(H) = \text{rank}\, X = p + 1$$

Recall that $H = X(X^\top X)^{-1}X^\top$ only involves predictors. Thus, leverage is useful to help identify outliers n the sense of having explanatory variables with extreme or unusual values. In simple linear regression,

$$h_{ii} = \frac{1}{n} + \frac{(x_i - \bar{x})^2}{S_{xx}}$$

Which means if x_i is far from $\bar{x}$, that point will have high leverage. This generalizes to multiple linear regression: an observation with high leverage is an outlier with extreme values in one or more explanatory variables. However, leverage does not tell us directly whether that observation is also an outlier in response, in fact, $y_i \approx \hat{\mu}_i$ for such observations with high leverage.

(4) An observation i is quite influential if its presence in fitting regression considerably changes estimates compared to when observation i is not used to fit a model.

Start with fitting $\boldsymbol{Y} = X\boldsymbol{\beta} + \boldsymbol{\varepsilon}$ using all observations and call estimates $\hat{\boldsymbol{\beta}}$ as usual in the least squares.

Let $\hat{\boldsymbol{\beta}}^{(i)}$ denote the least squares estimates based on fitting a model with the i^{th} observation removed.

Idea: If $\hat{\boldsymbol{\beta}}^{(i)}$ is quite different from $\hat{\boldsymbol{\beta}}$, then observation i is highly influential. Measure this via **Cook's distance** between $\hat{\boldsymbol{\beta}}$ and $\hat{\boldsymbol{\beta}}^{(i)}$:

$$D_i = \frac{(\hat{\boldsymbol{\beta}}^{(i)} - \hat{\boldsymbol{\beta}})^\top X^\top X(\hat{\boldsymbol{\beta}}^{(i)} - \hat{\boldsymbol{\beta}})}{\hat{\sigma}^2(p+1)}$$

To see this intuition, let $\hat{\boldsymbol{\mu}}^{(i)} = X\hat{\boldsymbol{\beta}}^{(i)}$ fitted values based on removing i^{th} observation and estimating $\boldsymbol{\beta}$. Then,

$$D_i = \frac{(X\hat{\boldsymbol{\beta}}^{(i)} - X\hat{\boldsymbol{\beta}})^\top (X\hat{\boldsymbol{\beta}}^{(i)} - X\hat{\boldsymbol{\beta}})}{\hat{\sigma}^2(p+1)} = \frac{(\hat{\boldsymbol{\mu}}^{(i)} - \hat{\boldsymbol{\mu}})^\top (\hat{\boldsymbol{\mu}}^{(i)} - \hat{\boldsymbol{\mu}})}{\hat{\sigma}^2(p+1)} = \frac{\|\hat{\boldsymbol{\mu}}^{(i)} - \hat{\boldsymbol{\mu}}\|^2}{\hat{\sigma}^2(p+1)}$$

D_i measures the Euclidean distance between fitted values of two regressions that give $\hat{\boldsymbol{\mu}}^{(i)}$ and $\hat{\boldsymbol{\mu}}$, up to a scaling factor. Further, it can be shown

$$D_i = d_i^2 \left(\frac{h_{ii}}{1-h_{ii}}\right)\left(\frac{1}{p+1}\right)$$

where we can see that both d_i and h_{ii} are key quantities to calculate D_i and observation i that are most influential will have larges of $|d_i|$ and h_{ii}.

- High $|d_i| \to$ outlier in response
- High $h_{ii} \to$ outlier in explanatory variable

So high influential observations tend to be outliers in the sense of having extreme values in *both* response and one/more predictors. Check if some D_i's are much larger than others, as they would be the most influential observations.

LECTURE 18 | 2020-11-11

For the i^{th} observation, the values of its predictors are

$$\boldsymbol{v}_i = \begin{bmatrix} 1 & x_{i1} & x_{i2} & \cdots & x_{ip} \end{bmatrix}^\top$$

$$X = \begin{bmatrix} \boldsymbol{v}_1^\top \\ \boldsymbol{v}_2^\top \\ \vdots \\ \boldsymbol{v}_n^\top \end{bmatrix}$$

$$X^\top X = \begin{bmatrix} \boldsymbol{v}_1 & \cdots & \boldsymbol{v}_n \end{bmatrix} \begin{bmatrix} \boldsymbol{v}_1^\top \\ \vdots \\ \boldsymbol{v}_n^\top \end{bmatrix} = \sum_{j=1}^n \boldsymbol{v}_j \boldsymbol{v}_j^\top$$

Recall $H = X(X^\top X)^{-1}X^\top$ so

$$h_{ii} = \boldsymbol{v}_i^\top (X^\top X)^{-1} \boldsymbol{v}_i^\top$$

Let $X^{(i)}$ be X with i^{th} observation deleted, $\boldsymbol{y}^{(i)}$ be $\boldsymbol{y}$ with i^{th} response deleted. Then,

$$(X^{(i)})^\top X^{(i)} = \sum_{j \neq i} \boldsymbol{v}_j \boldsymbol{v}_j^\top \implies X^\top X = (X^{(i)})^\top X^{(i)} + \boldsymbol{v}_i \boldsymbol{v}_i^\top$$

Similarly,

$$X^\top \boldsymbol{y} = \sum_{j=1}^n \boldsymbol{v}_j y_j \implies (X^{(i)})^\top \boldsymbol{y}^{(i)} = \sum_{j \neq i} \boldsymbol{v}_j y_j \implies X^\top \boldsymbol{y} = (X^{(i)})^\top \boldsymbol{y}^{(i)} + \boldsymbol{v}_i y_i$$

Let A be an $n \times n$ invertible matrix

$$(A - \boldsymbol{a}\boldsymbol{a}^\top)^{-1} = A^{-1} + \frac{A^{-1}\boldsymbol{a}\boldsymbol{a}^\top A^{-1}}{1 - \boldsymbol{a}^\top A^{-1}\boldsymbol{a}}$$

Fit the least squares using $X^{(i)}$ and $\boldsymbol{y}^{(i)}$ to obtain

$$\begin{aligned}
\hat{\boldsymbol{\beta}}^{(i)} &= [(X^{(i)})^\top X^{(i)}]^{-1}(X^{(i)})^\top \boldsymbol{y}^{(i)} \\
&= (X^\top X - \boldsymbol{v}_i \boldsymbol{v}_i^\top)^{-1}(X^\top \boldsymbol{y} - \boldsymbol{v}_i y_i) \\
&= \left[(X^\top X)^{-1} + \frac{(X^\top X)^{-1}\boldsymbol{v}_i \boldsymbol{v}_i^\top (X^\top X)^{-1}}{1 - \boldsymbol{v}_i^\top (X^\top X)^{-1}\boldsymbol{v}_i}\right](X^\top \boldsymbol{y} - \boldsymbol{v}_i y_i) \\
&= (X^\top X)^{-1}X^\top \boldsymbol{y} - (X^\top X)^{-1}\boldsymbol{v}_i y_i + \frac{(X^\top X)^{-1}\boldsymbol{v}_i \boldsymbol{v}_i^\top (X^\top X)^{-1}X^\top \boldsymbol{y} - (X^\top X)^{-1}\boldsymbol{v}_i \boldsymbol{v}_i^\top (X^\top X)^{-1}\boldsymbol{v}_i y_i}{1 - h_{ii}} \\
&= \hat{\boldsymbol{\beta}} - (X^\top X)^{-1}\boldsymbol{v}_i \left[y_i - \frac{\boldsymbol{v}_i^\top \hat{\boldsymbol{\beta}} - h_{ii} y_i}{1 - h_{ii}}\right] \\
&= \hat{\boldsymbol{\beta}} - (X^\top X)^{-1}\boldsymbol{v}_i \left[\frac{y_i - h_{ii} y_i - \boldsymbol{v}_i^\top \hat{\boldsymbol{\beta}} + h_{ii} y_i}{1 - h_{ii}}\right] \\
&= \hat{\boldsymbol{\beta}} - (X^\top X)^{-1}\boldsymbol{v}_i \left[\frac{y_i - \boldsymbol{v}_i^\top \hat{\boldsymbol{\beta}}}{1 - h_{ii}}\right] \\
&= \hat{\boldsymbol{\beta}} - (X^\top X)^{-1}\boldsymbol{v}_i \left[\frac{e_i}{1 - h_{ii}}\right]
\end{aligned}$$

Therefore,

$$\hat{\boldsymbol{\beta}}^{(i)} - \hat{\boldsymbol{\beta}} = \frac{-e_i}{1 - h_{ii}}(X^\top X)^{-1}\boldsymbol{v}_i$$

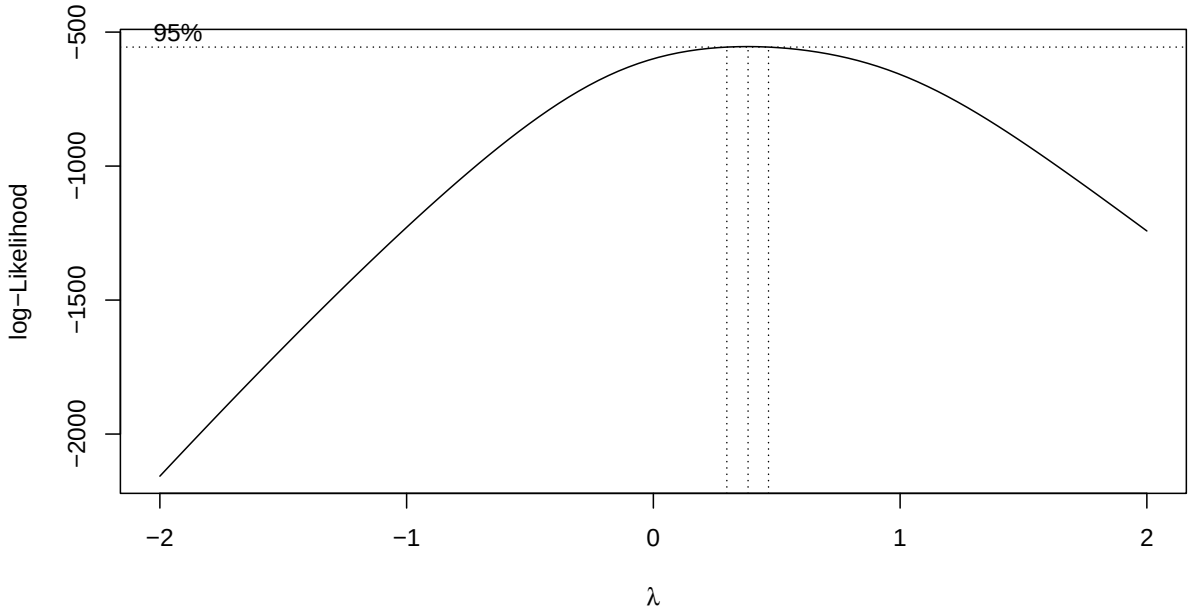
In fact, we can get $\hat{\boldsymbol{\beta}}^{(i)}$ from the regression with all n observations; that is, we don't need to fit a separate model.

$$\begin{aligned}
D_i &= \frac{(\hat{\boldsymbol{\beta}}^{(i)} - \hat{\boldsymbol{\beta}})^\top X^\top X (\hat{\boldsymbol{\beta}}^{(i)} - \hat{\boldsymbol{\beta}})}{\hat{\sigma}^2(p+1)} \\
&= \frac{\left(\frac{-e_i}{1 - h_{ii}}\right)\boldsymbol{v}_i^\top (X^\top X)^{-1}X^\top X \left(\frac{-e_i}{1 - h_{ii}}\right)(X^\top X)^{-1}\boldsymbol{v}_i}{\hat{\sigma}^2(p+1)} \\
&= \frac{e_i^2}{\hat{\sigma}^2(1 - h_{ii})(1 - h_{ii})}\left(\frac{h_{ii}}{p+1}\right) \\
&= d_i^2 \left(\frac{h_{ii}}{1 - h_{ii}}\right)\left(\frac{1}{p+1}\right)
\end{aligned}$$

as claimed, so we can calculate Cook's distance in terms of $|d_i|$ and h_{ii}. Therefore, the most influential observation on estimates of $\boldsymbol{\beta}$ are those with high $|d_i|$ and h_{ii}.

2.17.1 R Demo

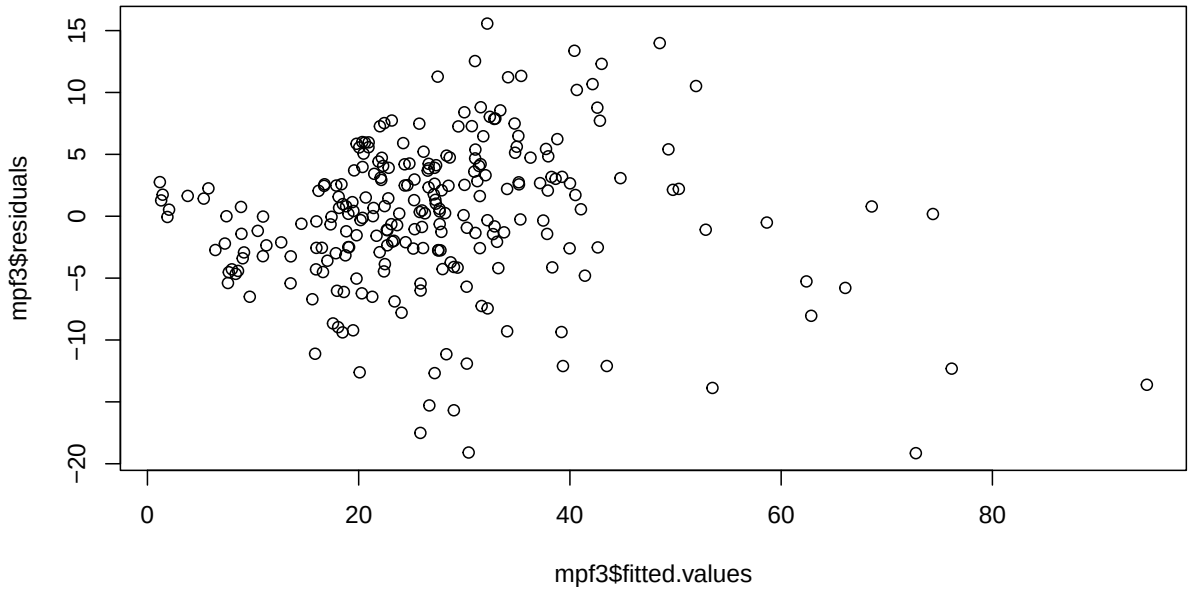
```
## Effect of individual observations
## Python data revisited
python <- read.csv("csv/FLpython.csv")
python$male <- ifelse(python$sex == "M", 1, 0)  # 1 = M, 0 =F
mpf2 <- lm(fat ~ male + mass + svl, data = python)
# Last time we used a Box-Cox transformation
library(MASS)
bc <- boxcox(mpf2)
```



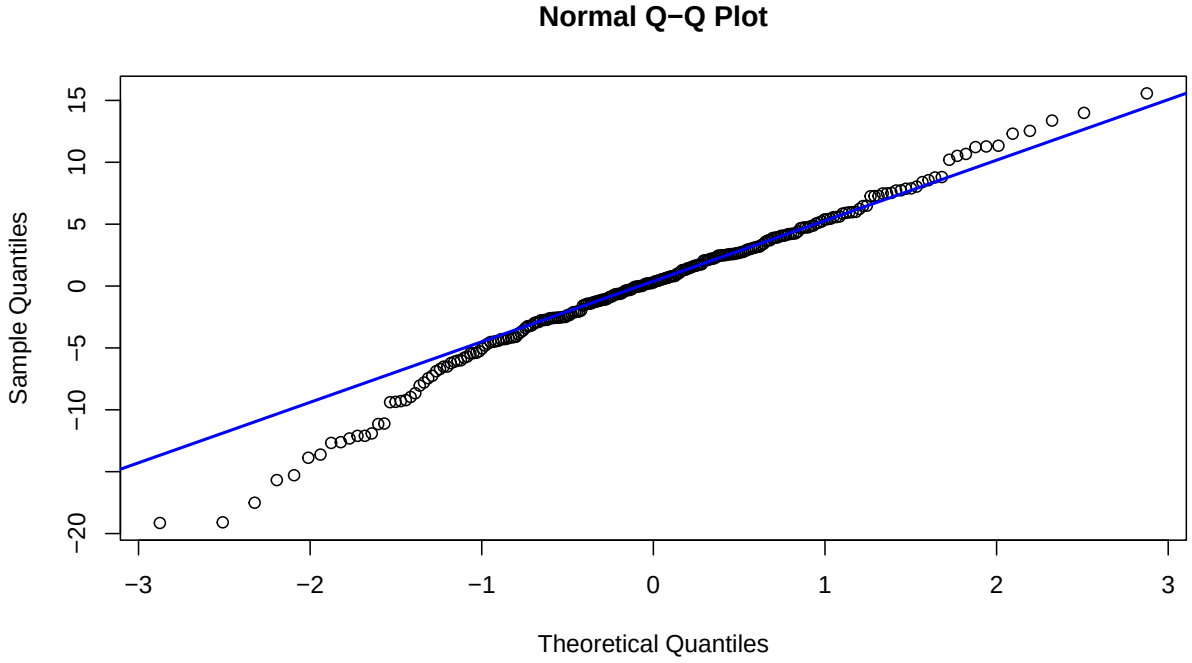

```
lambda <- bc$x[which.max(bc$y)]
mpf3 <- lm((fat^lambda - 1)/lambda ~ male + mass + svl, data = python)
summary(mpf3)
```

```
##
## Call:
## lm(formula = (fat^lambda - 1)/lambda ~ male + mass + svl, data = python)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -19.146  -2.910   0.297   3.688  15.568
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -8.0558134  2.1813183  -3.693 0.000273 ***
## male        -1.7849310  0.7776166  -2.295 0.022560 *
## mass         0.0004461  0.0000864   5.164 5.03e-07 ***
## svl          0.1431492  0.0126019  11.359  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.939 on 244 degrees of freedom
## Multiple R-squared:  0.8356,Adjusted R-squared:  0.8336
## F-statistic: 413.5 on 3 and 244 DF,  p-value: < 2.2e-16
```

```
plot(mpf3$fitted.values, mpf3$residuals)
```

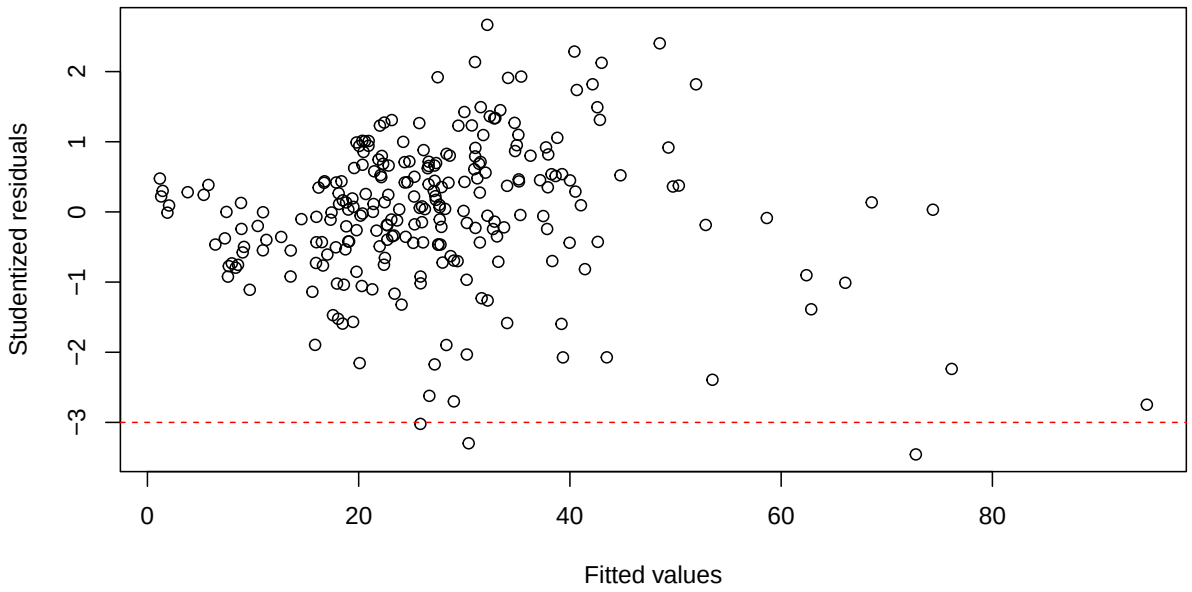



```
qqnorm(mpf3$residuals)
qqline(mpf3$residuals, col = "blue", lwd = 2)
```




```
# Quantities for individual observations
studres(mpf3)  # studentized residuals
hatvalues(mpf3)  # leverage
cooks.distance(mpf3)  # Cook's distance
```

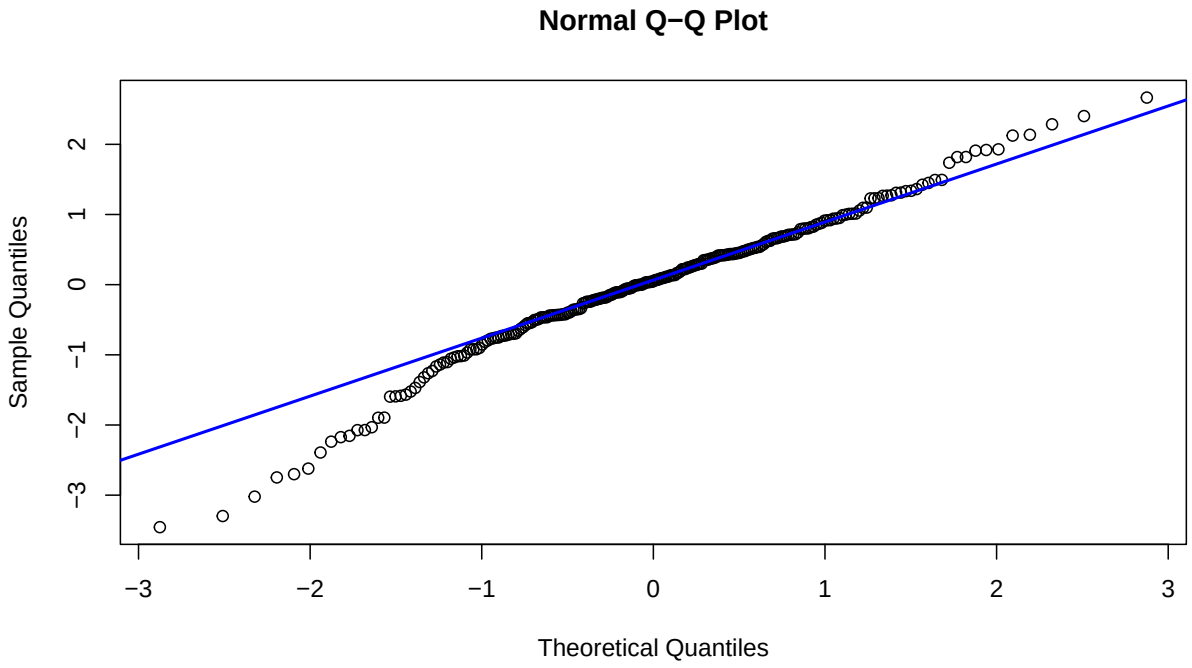
```r
# Residual plots with studentized residuals
plot(mpf3$fitted.values, studres(mpf3), xlab = "Fitted values",
  ylab = "Studentized residuals")
abline(h = c(3, -3), col = "red", lty = 2)
```



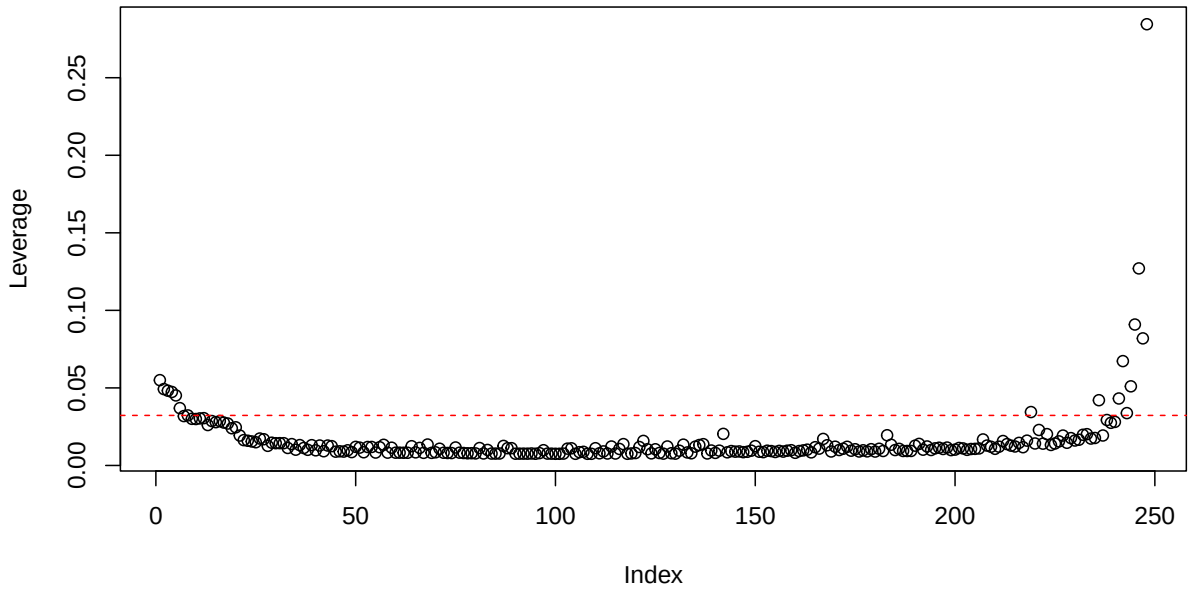

```r
which(abs(studres(mpf3)) > 3)
```

```
## 122 181 245
## 122 181 245
```

```r
qqnorm(studres(mpf3))
qqline(studres(mpf3), col = "blue", lwd = 2)
```




```
# Leverage
plot(hatvalues(mpf3), ylab = "Leverage")
abline(h = 2 * mean(hatvalues(mpf3)), col = "red", lty = 2)
```




```
which(hatvalues(mpf3) > 2 * mean(hatvalues(mpf3)))
```

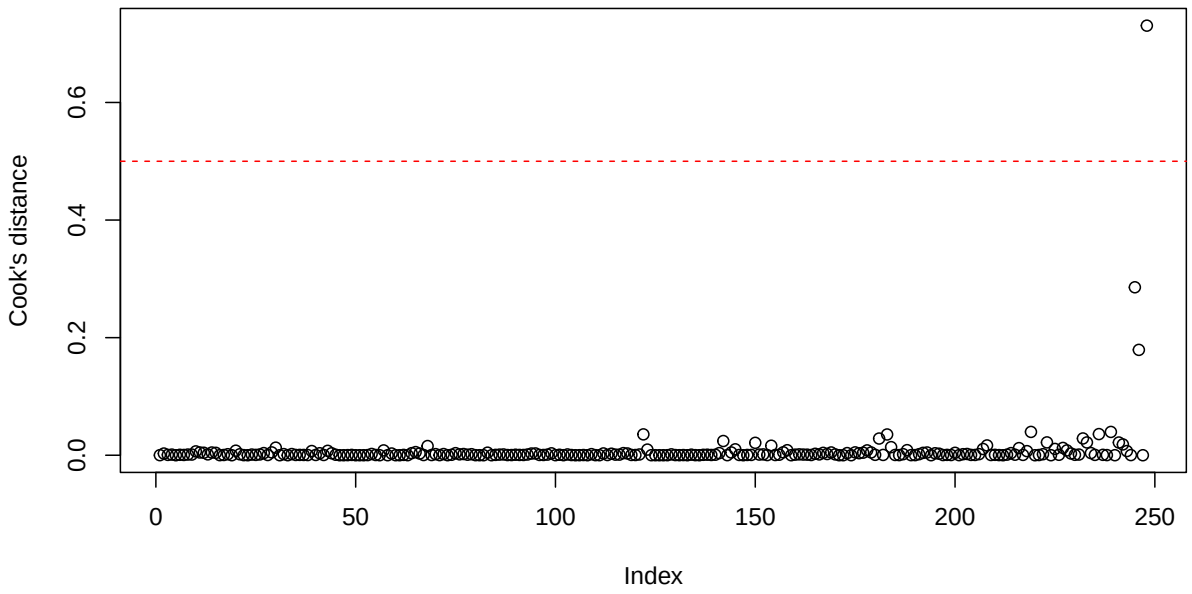
```
##   1   2   3   4   5   6   8 219 236 241 242 243 244 245 246 247 248
```

```
##   1   2   3   4   5   6   8 219 236 241 242 243 244 245 246 247 248
```

```
python[which(hatvalues(mpf3) > 2 * mean(hatvalues(mpf3))), ]
```

```
##     sex   svl  mass length      fat male
## 1     F  70.0   186   77.5    6.000    0
## 2     M  76.0   310   83.8   11.000    1
## 3     M  77.0   260   86.1    6.000    1
## 4     M  78.0   262   87.1    8.000    1
## 5     M  81.0   306   91.1    4.000    1
## 6     M  93.5   605  104.6   18.959    1
## 8     F 105.0   800  117.5   17.000    0
## 219   M 285.0 27000  316.2 3230.000    1
## 236   M 330.0 32600  370.9 4374.000    1
## 241   F 376.0 38280  424.2 3156.000    0
## 242   F 381.0 43910  424.9 4002.000    0
## 243   F 384.5 34540  432.4 3500.000    0
## 244   F 405.5 41660  455.3 5688.000    0
## 245   F 409.0 49900  460.2 2988.000    0
## 246   F 416.0 55260  469.1 4618.000    0
## 247   F 422.0 49350  473.4 6818.000    0
## 248   F 482.0 75500  545.0 8406.000    0
```

```
# Cook's distance
plot(cooks.distance(mpf3), ylab = "Cook's distance")
abline(h = 0.5, col = "red", lty = 2)
```




```
which(cooks.distance(mpf3) > 0.5)
```

```
## 248
## 248
```

```
# Let's look at actual changes in beta estimates
mpf3$coefficients  # with all the data
```

```
##   (Intercept)          male          mass           svl
## -8.0558134354 -1.7849310101  0.0004461197  0.1431491887
```

```
# e.g., fit without obs 248
mpf4 <- lm((fat^lambda - 1)/lambda ~ male + mass + svl, data = python[-248,
  ])
mpf4$coefficients
```

```
##   (Intercept)          male          mass           svl
## -6.6475616056 -1.6605858218  0.0005743312  0.1313793189
```

```
# e.g., fit without obs 50
mpf5 <- lm((fat^lambda - 1)/lambda ~ male + mass + svl, data = python[-50,
  ])
mpf5$coefficients
```

```
##   (Intercept)          male          mass           svl
## -8.0628754675 -1.7805093651  0.0004462354  0.1431573753
```

LECTURE 19 | 2020-11-16

2.18 Prediction Error

Sometimes a key application of a fitted model is to do prediction on new data (and is more important than interpretability).

Previously, we saw model selection criteria that are computed on fitted models (AIC, BIC, Adjusted R^2) which assess the explanatory power of a model on the *data used to fit the model* (or *train*).

While these criteria incorporate penalty terms to try to prevent overfitting, they don't directly assess how well a model would perform in predicting the response on new data given predictors.

We mentioned metrics such as MSPE as measures of predictive accuracy.

To assess accuracy in prediction, we need metrics for measuring prediction error, e.g., evaluated over m observations.

DEFINITION 2.18.1: Mean-squared Error (MSE)

If we have m observations,

$$\text{MSE} = \frac{1}{m}\sum_{i=1}^{m}(y_i - \hat{y}_i)^2$$

where y_i is the actual value, $\hat{y}_i$ is the predicted value (or equivalently, if we apply MSE on the fitted data, this would be the fitted value, $\hat{\mu}_i$; if calculated on training data). Measured on the scale of σ^2.

REMARK 2.18.2

MSE is equivalently called MSPE if applied on new data.

DEFINITION 2.18.3: Root-mean-squared Error (RMSE)

$$\text{RSME} = \sqrt{\text{MSE}}$$

REMARK 2.18.4

RMSE is measured on the scale of σ.

DEFINITION 2.18.5: Mean Absolute Error (MAE)

$$\text{MAE} = \frac{1}{m}\sum_{i=1}^{m}|y_i - \hat{y}_i|$$

Ideally, we have lots of data, conceptualize having three parts.

Train $(y_1, y_2, \ldots, y_n)$	Validation $(y_{n+1}, \ldots, y_{n+v})$	Test $(y_{n+v+1}, \ldots, y_{n+v+t})$
• n observations. • Fit models, as many as we want.	• v observations. • Estimate prediction error for each fitted model.	• t observations. • Used at the very end for final assessment of our selected model.

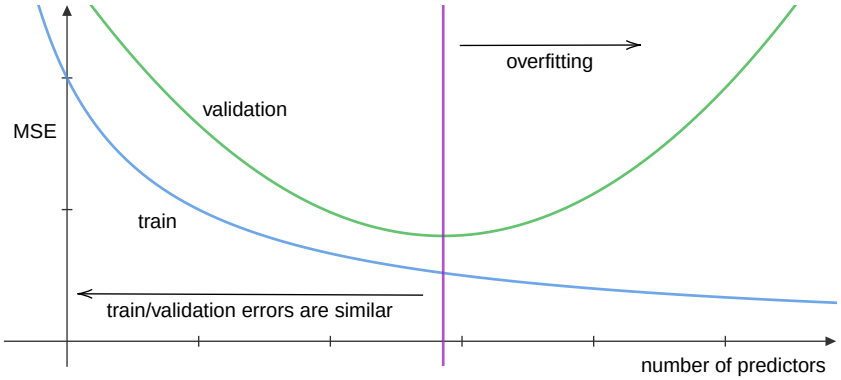
We have access to train and validation. Test data is from the future, we won't have access to this until the data is released (say the actual stock prices were released); that is, assume we don't get to see this.

For example, using MSE as a metric, based on a fitted model we can compute and compare:

MSE Train	MSE Validation	MSE Test
$\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$	$\frac{1}{m}\sum_{i=n+1}^{n+v}(y_i - \hat{y}_i)^2$	$\frac{1}{t}\sum_{i=n+v+1}^{n+v+t}(y_i - \hat{y}_i)^2$

- Observe "MSE Train" is equal to $\dfrac{\text{SS(Res)}}{n}$ and our usual estimate of $\hat{\sigma}^2$ is a scaled version of this quantity to compensate for the number of predictors. Specifically, $\hat{\sigma}^2 = \dfrac{\text{SS(Res)}}{n-p-1}$ so it's unbiased.
- Consider "MSE Validation" as an *estimate* of MSPE on new data.
- "MSE Test" is the actual test of prediction, we call this the *actual* MSPE.

Idea: We hope that "MSE Validation" $\approx$ "MSE Test" since neither sets were used to fit the model.



So, if we're using MSE/RMSE as a metric (as related to $\hat{\sigma}^2$ /$\hat{\sigma}$) and the MSE/RMSE is significantly larger on validation compared to train, then we probably overfitted the model; that is, we can't expect the model to generalize well to new data.

2.19 Cross-Validation

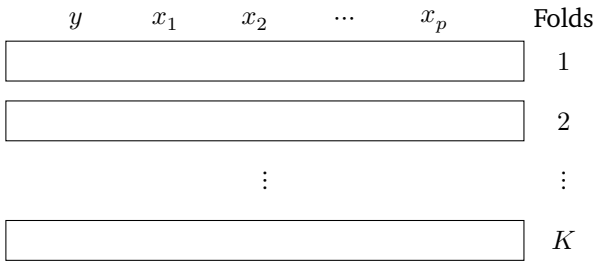
How to use framework in practice:

- Simplest: randomly divide available data between train/validation, say $80\%/20\%$ split.

  Weakness:

  1. Don't use all data for training.
  2. Only get one estimate of prediction error.

- Better: Use cross-validation scheme (CV). How to do CV with K folds:



  - Divide available data for train and validation into K roughly equally sized sets (folds), usually randomly.
  - For CV k, use data in fold k as validation, and train on the rest of data.
  - Thus, to estimate the prediction error *for a given model*, we fit it K times, each time treating the data in folds, $1, 2, \ldots, K$ as validation. Therefore, we get K estimates of prediction error for that particular model.
  - For example, using RMSE, we get

$$\text{RMSE}_1, \text{RMSE}_2, \ldots, \text{RSME}_K$$

    and can take the average

$$\overline{\text{RSME}} = \frac{1}{K}\sum_{k=1}^{K}\text{RSME}_k$$

    as an estimate for RMSPE on new data (test set).

2.19.1 R Demo

```
## Cross-validation
## Coffee example (Coffee Quality Institute, 2018)
## continued
coffee <- read.csv("csv/coffee_arabica.csv")
# 1 = wet, 0 otherwise
coffee$wet <- ifelse(coffee$Processing.Method == "Washed / Wet",
  1, 0)
# 1 = semi/dry, 0 otherwise
coffee$semi <- ifelse(coffee$Processing.Method == "Semi-washed / Semi-pulped",
  1, 0)
coffee$Processing.Method <- NULL
N <- nrow(coffee)
## Train and validation set split
set.seed(12345678)
trainInd <- sample(1:N, round(N * 0.8), replace = F)
trainSet <- coffee[trainInd, ]
validSet <- coffee[-trainInd, ]
```

```r
# Calculate RMSE on three models each with different
# variables included
m1 <- lm(Flavor ~ wet + semi + Aroma + Aftertaste + Body, dat = trainSet)
pred1 <- predict(m1, newdata = validSet)
sqrt(mean((validSet$Flavor - pred1)^2))  # RMSE
```

```
## [1] 0.1577479
```

```r
mean(abs(validSet$Flavor - pred1))  # MAE
```

```
## [1] 0.113643
```

```r
m2 <- lm(Flavor ~ wet + Aroma + Aftertaste + Body + Acidity +
  Balance + Sweetness + Uniformity + Moisture, dat = trainSet)
pred2 <- predict(m2, newdata = validSet)
sqrt(mean((validSet$Flavor - pred2)^2))
```

```
## [1] 0.1426565
```

```r
m3 <- lm(Flavor ~ Aroma + Aftertaste, dat = trainSet)
pred3 <- predict(m3, newdata = validSet)
sqrt(mean((validSet$Flavor - pred3)^2))
```

```
## [1] 0.1615385
```

```r
# K fold cross validation
K <- 5
validSetSplits <- sample((1:N)%%K + 1)
RMSE1 <- c()
RMSE2 <- c()
RMSE3 <- c()
for (k in 1:K) {
  validSet <- coffee[validSetSplits == k, ]
  trainSet <- coffee[validSetSplits != k, ]
  m1 <- lm(Flavor ~ wet + semi + Aroma + Aftertaste + Body,
    dat = trainSet)
  pred1 <- predict(m1, newdata = validSet)
  RMSE1[k] <- sqrt(mean((validSet$Flavor - pred1)^2))
  m2 <- lm(Flavor ~ wet + Aroma + Aftertaste + Body + Acidity +
    Balance + Sweetness + Uniformity + Moisture, dat = trainSet)
  pred2 <- predict(m2, newdata = validSet)
  RMSE2[k] <- sqrt(mean((validSet$Flavor - pred2)^2))
  m3 <- lm(Flavor ~ Aroma + Aftertaste, dat = trainSet)
  pred3 <- predict(m3, newdata = validSet)
  RMSE3[k] <- sqrt(mean((validSet$Flavor - pred3)^2))
}
RMSE1
```

```
## [1] 0.1479415 0.1653329 0.1556385 0.1656876 0.1482716
```

```r
RMSE2
```

```
## [1] 0.1427025 0.1525461 0.1478815 0.1620440 0.1384244
```

```r
RMSE3
```

```
## [1] 0.1513836 0.1667202 0.1616626 0.1675113 0.1532496
```

```r
mean(RMSE1)
```

```
## [1] 0.1565744
```

```
mean(RMSE2)
## [1] 0.1487197
mean(RMSE3)
## [1] 0.1601055
```

LECTURE 20 | 2020-11-18

2.20 Combining Cross Validation With Model Selection

- Given some candidate models (e.g., with different variables included) we can do K-fold cross-validation using each of them, and choose the one with the lowest average prediction error across the folds, e.g., $\frac{1}{K}\sum_{k=1}^{K}\text{RSME}_k$.
- How to choose K? Some common choices are:
  - $K = 5$
  - $K = 10$
  - $K = N$ (number of observations for train/valid). This is commonly known as "leave one out" (LOO).

  The more folds, the higher the computational time, but it can give a better estimate of prediction error.
- What if we are not given a list of models to compare? (and if there are too many predictors to do all possible regressions). Then, we can combine cross-validation with *model selection procedures* (criteria based on training data and search strategy).

To estimate the prediction error for a given *model selection procedure* we apply it K times, each time treating observations in fold $1, 2, \ldots, K$ as validation sample, e.g., use the *average* RMSE and choose the model selection procedure with the lowest estimated prediction error.

Note: actual variables selected in each fold might be different.

Recommendation: apply chosen model selection *procedure* (lowest estimated prediction error) to the entire set of observations available for training and validation to get a final model (for applications to test data).

Recall we have:

- AIC $= 2q - 2\ln[L(\hat{\theta})]$
- BIC $= q\ln(n) - 2\ln[L(\hat{\theta})]$

We can go further (and beyond *PLUS ULTRA!*) and consider the following.

DEFINITION 2.20.1: L_0 Penalized Likelihood

For any $\lambda > 0$,

$$q\lambda - 2\ln[L(\hat{\theta})]$$

Beyond Stepwise Regression

- Stepwise is a deterministic algorithm which is fast, but might not give optimal model selected.
- Could try stochastic algorithm, e.g., iterative conditional minimization (ICM)

2.21 Iterative Conditional Minimization

Start with a model with just intercept. For a given random seed, take predictors $x_1, \ldots, x_p$ and randomly re-order them into $x_{(1)}, \ldots, x_{(p)}$. For $j = 1, \ldots, p$: (let $\mathcal{S}$ be the predictors currently in model, so start at $\mathcal{S} = \emptyset$).

- If $x_{(j)}$ is not in $\mathcal{S}$:
  - Fit model with $\mathcal{S}$. If addition of $x_{(j)}$ improves criterion, then add $x_{(j)}$ to $\mathcal{S}$.
- If $x_{(j)}$ is in $\mathcal{S}$:
  - Fit model removing $x_{(j)}$. If excluding $x_{(j)}$ improves criterion, then remove $x_{(j)}$ from $\mathcal{S}$.

Repeat for loop until no predictors are added/removed from an entire pass through the for loop.

REMARK 2.21.1

Different random orderings could give different sets $\mathcal{S}$ at end of procedure, could pick one that has the best criterion overall.

2.21.1 R Demo

```
## Cross-validation with model selection
# Dataset from paper 'Where does Haydn end and Mozart
# begin?  Composer classification of string quartets'
# (Journal of New Music Research, vol 49, 457-476)
HM <- read.csv("haydn-mozart.csv")
HM[, 1] <- NULL  # first col is just name of quartet, remove it
dim(HM)  # 285 observations, 1116 columns
# Let's treat 'number of notes in violin part' as the
# response That's variable name 'count_pitch_1' and column
# 683 of data matrix So we have 1115 possible predictors
# More model selection, for clarity start with one
# train/validation split
N <- nrow(HM)
set.seed(12345678)
trainInd <- sample(1:N, round(N * 0.8), replace = F)
trainSet <- HM[trainInd, ]
validSet <- HM[-trainInd, ]
library(MASS)
# Full model and empty model with just intercept
full <- lm(count_pitch_1 ~ ., data = trainSet)
empty <- lm(count_pitch_1 ~ 1, data = trainSet)
# Stepwise forward with BIC
stepAIC(object = empty, scope = list(upper = full, lower = empty),
  direction = "forward", k = log(nrow(trainSet)))
m1 <- lm(formula = count_pitch_1 ~ Prop_m3_num_0_8.1 + count_pitch_3 +
  Prop_m3_num_0_8.3 + Prop_m3_mean_8.1 + count_pitch_4 + Dev_count_8_thresh4.393.1 +
  Prop_m3_mean_8.3 + mean_time_3 + Dev_count_t_14_thresh0.216.3 +
  voicepair_int_dist_6_1.2 + Prop_m3_sd_8.3 + Prop_m3_sd_8.1 +
  Prop_m3_num_.6_8.1 + simult_rest_perc + Dev_perc_t_14.1 +
  count_pitch_2 + Prop_m3_num_0_8.2 + Prop_m3_mean_8.2 + Dev_count_8_thresh4.024.2 +
  Dev_count_18_thresh3.899.2 + Expo_t_count_14.thresh0.7.1 +
  Prop_m3_num_0_12.1 + Expo_acc_8.1 + Expo_perc_8.2 + Prop_m3_num_0_12.2 +
  Dev_count_t_8_thresh0.247.1 + Expo_t_count_8.thresh0.7.1 +
  voicepair_int_dist_1_1.3 + Expo_perc_12.3 + Pairwise_voice_int_mean.1.3 +
```

```
  Expo_t_count_18.thresh0.7.3 + Prop_m3_q3_16.2 + Dev_perc_t_14.4 +
  Prop_m3_q3_14.4 + Dev_count_t_14_thresh0.187.3, data = trainSet)
# we can use the AIC function with our own k for the LO
# penalty
AIC(m1, k = log(nrow(trainSet)))
BIC(m1)  # in this case matches BIC as we expect (1977.6)
pred1 <- predict(m1, newdata = validSet)
sqrt(mean((validSet$count_pitch_1 - pred1)^2))  # RMSE on validation
sqrt(mean(m1$residuals^2))  # RMSE on train
# Try ICM to search for a model with a potentially better
# BIC than the one found with stepwise
pen <- log(nrow(trainSet))  #
varlist = c()
varnames = names(trainSet)
n = nrow(trainSet)
varorder <- sample(1:ncol(trainSet))  # random order of variables
minCrit = Inf
noChange = F
while (!noChange) {
  noChange = T
  for (i in varorder) {
    if (i == 683)
      next
    if (i %in% varlist & length(varlist) > 1) {
      index = c(683, varlist[varlist != i])
      trainVars = trainSet[, index]
      fit = lm(count_pitch_1 ~ ., data = trainVars)
      if (AIC(fit, k = pen) < minCrit) {
        minCrit = AIC(fit, k = pen)
        varlist = varlist[varlist != i]
        print(paste0("Criterion: ", round(minCrit, 1),
          ", variables: ", paste0(varnames[varlist],
          collapse = " ")))
        best.model = fit
        noChange = F
      }
    } else if (!i %in% varlist) {
      index = c(683, varlist, i)
      trainVars = trainSet[, index]
      fit = lm(count_pitch_1 ~ ., data = trainVars)
      if (AIC(fit, k = pen) < minCrit) {
        minCrit = AIC(fit, k = pen)
        varlist = c(varlist, i)
        print(paste0("Criterion: ", round(minCrit, 1),
          ", variables: ", paste0(varnames[varlist],
          collapse = " ")))
        best.model = fit
        noChange = F
      }
    }
  }
}
summary(best.model)
```

```
predICM <- predict(best.model, newdata = validSet)
sqrt(mean((validSet$count_pitch_1 - predICM)^2))  # RMSE on validation
sqrt(mean(best.model$residuals^2))  # RMSE on train
# Try stepwise again, with a larger L0 penalty (e.g., twice
# the usual BIC penalty)
stepAIC(object = empty, scope = list(upper = full, lower = empty),
  direction = "forward", k = 2 * log(nrow(trainSet)))
m2 <- lm(formula = count_pitch_1 ~ Prop_m3_num_0_8.1 + count_pitch_3 +
  Prop_m3_num_0_8.3 + Prop_m3_mean_8.1 + count_pitch_4 + Dev_count_8_thresh4.393.1 +
  Prop_m3_mean_8.3 + mean_time_3 + Dev_count_t_14_thresh0.216.3,
  data = trainSet)
AIC(m2, k = 2 * log(nrow(trainSet)))
# calculate the value of criterion based on this larger L0
# penalty
pred2 <- predict(m2, newdata = validSet)
sqrt(mean((validSet$count_pitch_1 - pred2)^2))  # RMSE on validation
sqrt(mean(m2$residuals^2))  # RMSE on train
# Try ICM as well with this penalty
pen <- 2 * log(nrow(trainSet))
varlist = c()
varnames = names(trainSet)
n = nrow(trainSet)
varorder <- sample(1:ncol(trainSet))
minCrit = Inf
noChange = F
while (!noChange) {
  noChange = T
  for (i in varorder) {
    if (i == 683)
      next
    if (i %in% varlist & length(varlist) > 1) {
      index = c(683, varlist[varlist != i])
      trainVars = trainSet[, index]
      fit = lm(count_pitch_1 ~ ., data = trainVars)
      if (AIC(fit, k = pen) < minCrit) {
        minCrit = AIC(fit, k = pen)
        varlist = varlist[varlist != i]
        print(paste0("Criterion: ", round(minCrit, 1),
          ", variables: ", paste0(varnames[varlist],
          collapse = " ")))
        best.model = fit
        noChange = F
      }
    } else if (!i %in% varlist) {
      index = c(683, varlist, i)
      trainVars = trainSet[, index]
      fit = lm(count_pitch_1 ~ ., data = trainVars)
      if (AIC(fit, k = pen) < minCrit) {
        minCrit = AIC(fit, k = pen)
        varlist = c(varlist, i)
        print(paste0("Criterion: ", round(minCrit, 1),
          ", variables: ", paste0(varnames[varlist],
          collapse = " ")))
```

```
        best.model = fit
        noChange = F
      }
    }
  }
}
predICM <- predict(best.model, newdata = validSet)
sqrt(mean((validSet$count_pitch_1 - predICM)^2))  # RMSE on validation
sqrt(mean(best.model$residuals^2))  # RMSE on train
# K fold cross validation to choose model selection method
K <- 5
validSetSplits <- sample((1:N)%%K + 1)
RMSE1 <- c()
RMSE2 <- c()
for (k in 1:K) {
  validSet <- HM[validSetSplits == k, ]
  trainSet <- HM[validSetSplits != k, ]
  full <- lm(count_pitch_1 ~ ., data = trainSet)
  empty <- lm(count_pitch_1 ~ 1, data = trainSet)
  m1 <- stepAIC(object = empty, scope = list(upper = full,
    lower = empty), direction = "forward", k = log(nrow(trainSet)))
  pred1 <- predict(m1, newdata = validSet)
  RMSE1[k] <- sqrt(mean((validSet$count_pitch_1 - pred1)^2))
  m2 <- stepAIC(object = empty, scope = list(upper = full,
    lower = empty), direction = "forward", k = 2 * log(nrow(trainSet)))
  pred2 <- predict(m2, newdata = validSet)
  RMSE2[k] <- sqrt(mean((validSet$count_pitch_1 - pred2)^2))
}
RMSE1
RMSE2
mean(RMSE1)
mean(RMSE2)
# turns out m2 is indeed the better procedure among these
# two based on CV prediction error if we decide on
# procedure m2, we can apply procedure m2 to the entire 285
# observations to get a final model for future prediction
# e.g.,
full <- lm(count_pitch_1 ~ ., data = HM)
empty <- lm(count_pitch_1 ~ 1, data = HM)
mfinal <- stepAIC(object = empty, scope = list(upper = full,
  lower = empty), direction = "forward", k = 2 * log(nrow(trainSet)))
```

Chapter 3

Generalized Linear Models

LECTURE 21 | 2020-11-23

3.1 Beyond STAT 331: Generalized Linear Models and Logistic Regression

We have a vector of responses $\boldsymbol{Y}$ which are independent, and predictors $\boldsymbol{x}_1, \dots, \boldsymbol{x}_p$.

DEFINITION 3.1.1: Generalized Linear Model (GLM)

Three ingredients of a GLM:

1. Random component. Response Y_i is a random variable with a distribution that is a member of the *exponential family*, e.g., Normal, Binomial, Poisson.
2. Systematic component. Typically, a linear predictor based on $x_{i1}, \dots, x_{ip}$ which we denote as
$$\eta_i = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip}$$
3. Link function: A function $g(\cdot)$ that defines a relationship between $\mathbb{E}[Y_i]$ and η_i; that is, $\eta_i = g(\mathbb{E}[Y_i])$.

EXAMPLE 3.1.2: Multiple Linear Regression

MLR: $Y_i = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip} + \varepsilon$ where $\varepsilon_i \overset{\text{iid}}{\sim} \mathcal{N}(0, \sigma^2)$

- Random component: Y_i is Normal and $\mathbb{E}[Y_i] = \eta_i$.
- Link function: $g(\mathbb{E}[Y_i]) = \mathbb{E}[Y_i]$ (identity).

EXAMPLE 3.1.3: Logistic Regression

Logistic regression: can be used when response Y_i is binary; that is,

$$Y_i = \begin{cases} 1 & \text{success/on/yes/goose} \\ 0 & \text{failure/off/no/non-goose} \end{cases}$$

- Random component: $\mathbb{P}(Y_i = 1) = \pi_i$ and $\mathbb{P}(Y_i = 0) = 1 - \pi_i$; that is, $Y_i \sim \text{Binomial}(1, \pi_i)$.
- Link function for logistic regression sets
$$\ln\left(\frac{\pi_i}{1 - \pi_i}\right) = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip} = \eta_i$$

Note: $\mathbb{E}[Y_i] = 1(\pi_i) + 0(1-\pi_i) = \pi_i$ so logistic regression takes $g(\pi_i) = \ln\left(\frac{\pi_i}{1-\pi_i}\right)$. Say A is an event, then the *odds* of event A is defined as

$$\frac{\mathbb{P}(A)}{1-\mathbb{P}(A)}$$

Therefore, $\frac{\pi_i}{1-\pi_i}$ is the odds that $Y_i = 1$ and $\ln\left(\frac{\pi_i}{1-\pi_i}\right)$ is the log-odds that $Y_i = 1$. Since $\eta_i = \ln\left(\frac{\pi_i}{1-\pi_i}\right)$, inverting this function gives

$$e^{\eta_i} = \frac{\pi_i}{1-\pi_i} \iff \pi_i = \frac{e^{\eta_i}}{1+e^{\eta_i}}$$

Plot $y = \frac{e^x}{1+e^x}$.

Note that π_i is bounded by 0 to 1, while $\ln\left(\frac{\pi_i}{1-\pi_i}\right)$ can take any real number.

- Since $\ln\left(\frac{\pi_i}{1-\pi_i}\right) = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip}$ and $\mathbb{P}(Y_i = 1) = \pi_i$, then β_j is the expected *additive change* in the log-odds of the event that $Y_i = 1$ for a unit increase in x_j (holding other variables constant). Also,

  $$\frac{\pi_i}{1-\pi_i} = e^{\beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip}}$$

  would say that e^{β_j} is the expected *multiplicative change* in odds of $Y_i = 1$ or a unit increase in x_j (holding other variables constant).
- Model is fit using MLE (`glm()` function in R).

EXAMPLE 3.1.4: Logistic Regression (Music Analysis)—String Quartet Classification

Define

$$Y_i = \begin{cases} 1 & \text{Haydn} \\ 0 & \text{Mozart} \end{cases}$$

The model (for illustration purposes, the actual model had 7 predictors) is given by:

$$\ln\left(\frac{\pi_i}{1-\pi_i}\right) = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2}$$

From training data (observation with Y_i's known and predictors given) we have

$$\hat{\boldsymbol{\beta}} = (-1.12, -15.47, 16.71)^\top$$

Also,

- x_1 = SD of note duration in 1st violin where $x_1 \in (0.05, 0.35)$.
- x_2 = proportion descending pairwise intervals in 1st violin where $x_2 \in (0.2, 0.5)$.
- Higher values of x_1 are more likely to be Mozart
- Higher values of x_2 are more likely to be Haydn

Suppose we are given $\boldsymbol{x}_1 = (0.05, 0.2, 0.35)^\top$ and $\boldsymbol{x}_2 = (0.4, 0.4, 0.4)^\top$, then the higher estimated probability is Mozart since $\hat{\beta}_1$ is negative.

Estimate log odds of $Y_i = 1$ is

$$-1.12 - 15.47(0.05) + 16.71(0.4) = \hat{\eta}_i$$

Estimate π_i is

$$\frac{e^{\hat{\eta}_i}}{1+e^{\hat{\eta}_i}} = 0.992$$

which is the estimate probability an observation is Haydn for $x_1 = 0.05$ and $x_2 = 0.4$.
- Second probability: 0.992
- Third probability: 0.537

Interpretation for $\widehat{\beta}_1$:
- For unit increase in SD of note duration in 1st violin, estimated expected change in log-odds of the piece being Haydn is -15.47.

LECTURE 22 | 2020-11-25

3.2 Last Lecture!

Project Tips:

- Explore the data
- No one "right" model
- Understand what you're fitting and why
- Present it well (no raw R output in report)
- Data set has many variables (be careful about overfitting)—don't wait until last minute.

Residual Surrealism:

- Want $(\widehat{\mu}_i, e_i)$, $i = 1, \dots, n$ to represent black pixel locations in the image.
- Recall in fitting MLR that $\boldsymbol{e}$ and $\widehat{\boldsymbol{\mu}}$ satisfy
$$\widehat{\boldsymbol{\mu}} = H\boldsymbol{Y}$$
$$\boldsymbol{e} = (I - H)\boldsymbol{Y}$$
which is a linear system of $2n$ equations.
- Reverse engineer what $\boldsymbol{Y}, \boldsymbol{x}_1, \dots, \boldsymbol{x}_p$ could be. There are $n(p+1)$ free variables.
- Fix $\boldsymbol{\beta}$ (choose our regression coefficient).
- Recall $\boldsymbol{Y} = X\boldsymbol{\beta} + \boldsymbol{\varepsilon}$, so the equations can be expressed as
$$\widehat{\boldsymbol{\mu}} = H(X\boldsymbol{\beta} + \boldsymbol{\varepsilon}) = X\boldsymbol{\beta} + H\boldsymbol{\varepsilon}$$
$$\boldsymbol{e} = (I - H)\boldsymbol{\varepsilon} = \boldsymbol{\varepsilon} - H\boldsymbol{\varepsilon}$$
- Residuals also satisfy condition
$$\widehat{\boldsymbol{\mu}}^\top \boldsymbol{e} = 0$$
since $\boldsymbol{e}$ are orthogonal to Span(X) which might not be satisfied by an arbitrary image.

  Solution. Add a few pixels to corners (outliers, influential points) to satisfy orthogonality.
- Least squares solution implies $\mathbf{1}^\top \boldsymbol{e} = 0$ for the intercept, and $\tilde{X}^\top \boldsymbol{e} = \mathbf{0}$ for the predictors.
$$X = \begin{bmatrix} \mathbf{1} & \tilde{X} \end{bmatrix}$$
So if we let
$$\tilde{X} = \left(I_n - \frac{\boldsymbol{e}\boldsymbol{e}^\top}{\boldsymbol{e}^\top \boldsymbol{e}} \right) M_{n \times p}$$
then $\tilde{X}\boldsymbol{e} = 0$ is satisfied, for any matrix M.

- Simulate $\boldsymbol{Z}$ i.i.d. from $\mathcal{N}(0, \tau^2)$ controls R^2 in MLR and let

$$\boldsymbol{\varepsilon} = \boldsymbol{e} + H\boldsymbol{Z}$$

  and substitute in 1st equation to get

$$\hat{\boldsymbol{\mu}} = X\boldsymbol{\beta} + H\boldsymbol{e} + H\boldsymbol{Z} = \underbrace{X\boldsymbol{\beta}}_{\text{function of } M} + \underbrace{H\boldsymbol{Z}}_{\text{function of } M}$$

- Iterative solution to update M until it satisfies the equation.

  The program provided by the author that has the easiest version to use runs on Windows, so I had to boot up a Virtual Box for Windows.—Samuel Wong

REMARK 3.2.1: Author's Note

I use Arch btw.