GroupH_HM2

Simonutti, Younes Pour Langaroudi, Billo, Tavano, Vicig

2024-12-04

Contents

FSDS - Chapter 4

Ex 4.2

For a sequence of observations of a binary random variable, you observe the geometric random variable (Section 2.2.2) outcome of the first success on observation number y = 3. Find and plot the likelihood function.

Solution

The probability mass function (p.m.f) of the **Geometric Distribution** is given by:

$$f_Y(y;p) = (1-p)^{y-1}p, \quad y = 1, 2, 3, \ldots,$$

Substituting $y = 3$ in the formula we obtain:

$$f(3;p) = (1-p)^{3-1}p = (1-p)^2 p$$

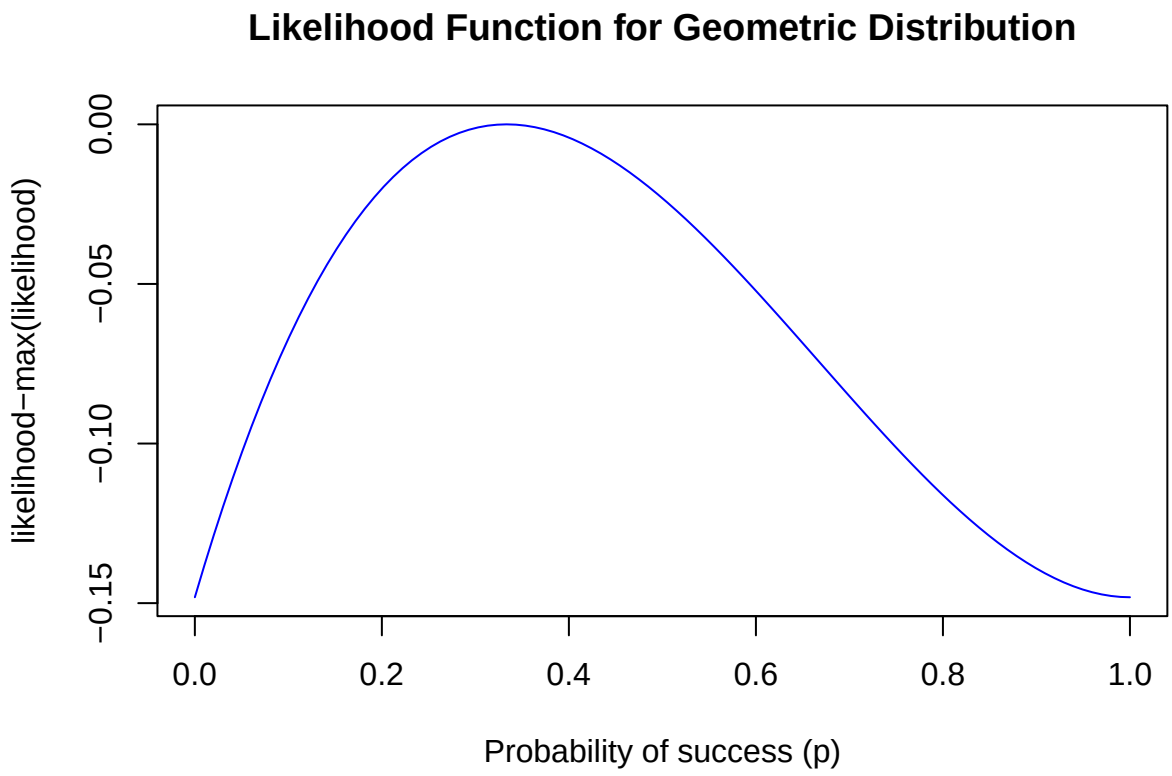
Compute the likelihood function:

$$L(\theta) = L(p) = \prod_{i=1}^{n} f(y_i;p) = \prod_{i=1}^{n}[(1-p)^{y_i-1}p] = (1-p)^{\sum_{i=1}^{n}(y_i-1)}p$$

$$L(p) = p(1-p)^2$$

We can graphically represent the likelihood function:

```r
# Define the likelihood function for y = 3
likelihood_function <- function(p) {
  return((1 - p)^2 * p)
}

# Generate a sequence of p values
p_values <- seq(0, 1, length.out = 100)

# Calculate the likelihood for each p value
likelihood_values <- likelihood_function(p_values)
max_pi <- p_values[which.max(likelihood_values)]

# Plot the likelihood function
plot(p_values, likelihood_values-max(likelihood_values), type = "l", col = "blue",
     main = "Likelihood Function for Geometric Distribution",
     xlab = "Probability of success (p)", ylab = "likelihood-max(likelihood)")
```



The graph shows that with $y = 3$, meaning that it was necessary to have 3 trials before a success, the most likely parameter to have generated this data is $p = 0.3333333$

Ex 4.4

For the Students data file and corresponding population, find the ML estimate of the population proportion believing in life after death. Construct a Wald 95% confidence interval, using its formula (4.8). Interpret.

(4.8)

$$\hat{\pi} = z_{\alpha/2}\sqrt{\frac{\hat{\pi}(1-\hat{\pi})}{n}}$$

Solution

```
url <- "https://stat4ds.rwth-aachen.de/data/Students.dat"

students <- read.table(url, header = TRUE)

head(students)
```

```
##   subject gender age hsgpa cogpa dhome dres tv sport news aids veg affil ideol
## 1       1      0  32   2.2   3.5     0  5.0  3     5    0    0   0     2     6
## 2       2      1  23   2.1   3.5  1200  0.3 15     7    5    6   1     1     2
## 3       3      1  27   3.3   3.0  1300  1.5  0     4    3    0   1     1     2
## 4       4      1  35   3.5   3.2  1500  8.0  5     5    6    3   0     3     4
## 5       5      0  23   3.1   3.5  1600 10.0  6     6    3    0   0     3     1
```

```
## 6       6      0  39   3.5   3.5   350  3.0  4     5     7     0    1     1     2
##   relig abor affirm life
## 1     2    0      0    1
## 2     1    1      1    3
## 3     2    1      1    3
## 4     1    1      1    2
## 5     0    1      0    2
## 6     1    1      1    3
```

```
n = nrow(students)
life_after_death = students$life["life" = 1]

waldInterval = function(x, n, conf.level = 0.95){
   p <- x/n
   sd <- sqrt(p*((1-p)/n))
   z <- qnorm(c( (1 - conf.level)/2, 1 - (1-conf.level)/2))
   #returns the value of thresholds at which conf.level has to be cut at. for 95% CI,
   #this is -1.96 and +1.96
   ci <- p + z*sd
   return(ci)
}
waldInterval(life_after_death, n)
```

```
## [1] -0.01572604  0.04905937
```

The interval is extremely small and includes 0: most probably the portion of students that believe in life after death is not significantly different from 0. However, it must be noted that the interval also contains values smaller than 0, which obviously cannot be taken by the parameter of a proportion. For this reason, other types of tests may be more suitable in this situation.

Ex 4.38

For independent observations $y_1, ..., y_n$ having the geometric distribution (2.1):

(a) Find a sufficient statistic for π. (b) Derive the ML estimator of π.

Solution

a)

The probability mass function (p.m.f) of the **Geometric Distribution** is given by:

$$f_Y(y; p) = (1-p)^{y-1}p, \quad y = 1, 2, 3, \dots,$$

For n independent observations $y_1, \dots, y_n$, the joint likelihood function is:

$$L(p; y_1, \dots, y_n) = \prod_{i=1}^{n} P(Y_i = y_i) = \prod_{i=1}^{n}[(1-p)^{y_i-1}p] = (1-p)^{\sum_{i=1}^{n}(y_i-1)}p^n$$

The sufficient statistic, derived from the formula is given by the exponential part $\sum_{i=1}^{n}(y_i - 1)$. Thus, the likelihood depends on the data only through $\sum_{i=1}^{n} y_i$.

In conclusion, by the Factorization Theorem, $s(y) = \sum_{i=1}^{n} y_i$ is a sufficient statistic for p.

b)

In order to find ML for p analytically, we need to compute the log-likelihood function:

$$l(p) = log(L(p; y_1, ..., y_n)) = \sum_i^n y_i \ log(1-p) + n \log(p)$$

Now, we differentiate $l(p)$ with respect to p and set it to 0:

$$\frac{\partial \ l(p)}{\partial \ p} = \frac{\sum_i^n y_i}{1-p} - \frac{n}{p} = 0$$

$$p \sum_i^n y_i = n(1-p) => p = \frac{n}{\sum_i^n y_i}$$

In conclusion, The Maximum Likelihood Estimate (MLE) for p is:

$$\hat{p} = \frac{n}{\sum_{i=1}^n y_i}$$

Ex 4.44

Refer to the previous two exercises. Consider the selling prices (in thousands of dollars) in the Houses data file mentioned in Exercise 4.31. (a) Fit the normal distribution to the data by finding the ML estimates of μ and σ for that distribution.

(b) Fit the log-normal distribution to the data by finding the ML estimates of its parameters.

(c) Find and compare the ML estimates of the mean and standard deviation of selling price for the two distributions.

(d) Superimpose the fitted normal and log-normal distributions on a histogram of the data. Which distribution seems to be more appropriate for summarizing the selling prices?

Solution

```
url_houses = "https://stat4ds.rwth-aachen.de/data/Houses.dat"

houses <- read.table(url_houses, header = TRUE)

head(houses)
```

```
##   case  price size new taxes bedrooms baths
## 1    1 419.85 2048   0  3104        4     2
## 2    2 219.75  912   0  1173        2     1
## 3    3 356.55 1654   0  3076        4     2
## 4    4 300.00 2068   0  1608        3     2
## 5    5 239.85 1477   0  1454        3     3
## 6    6 749.85 3153   1  2997        3     2
```

a)

```r
likelihood_normal <- function(mu, sigma2, data) {
  n <- length(data)
  constant <- 1 / sqrt(2 * pi * sigma2)
  exponent <- exp(-((data - mu)^2) / (2 * sigma2))
  likelihood <- prod(constant * exponent)
  return(-likelihood)
}

mu_hat = mean(houses$price)
s2_hat = var(houses$price)

ML_sigma = optim(
  par = s2_hat,  # Initial guess for sigma^2
  fn = likelihood_normal,
  mu = mu_hat,
  data = houses$price,
  method = "L-BFGS-B",
  lower = 1e-6  # Ensure sigma^2 > 0
)

ML_mu = optim(
  par = mu_hat,  # Initial guess for sigma^2
  fn = likelihood_normal,
  sigma = s2_hat,
  data = houses$price,
  method = "L-BFGS-B",
  lower = 1e-6  # Ensure sigma^2 > 0
)

comparison <- data.frame(
  Quantity = c("Mean", "Variance"),
  MLE_Function = c(ML_mu$par, ML_sigma$par),
  Built_in_Function = c(mu_hat, s2_hat)
)

comparison
```

```
##   Quantity MLE_Function Built_in_Function
## 1     Mean     232.9965          232.9965
## 2 Variance   22986.5803        23071.5803
```

b)

```r
log_likelihood_normal <- function(mu, sigma2, data) {
  #browser()
  n <- length(data)
  log_constant <- -n / 2 * log(2 * pi * sigma2)  # Logarithm of the constant term
  sum_squared <- sum((data - mu)^2)              # Sum of squared deviations
  log_exponent <- -sum_squared / (2 * sigma2)    # Logarithm of the exponent
  log_likelihood <- log_constant + log_exponent
  return(-log_likelihood)  # Return the negative log-likelihood for optimization
}
```

```r
log_likelihood_normal(mu_hat, s2_hat, houses$price)
```

```
## [1] 643.7117
```

```r
Log_ML_sigma = optim(
  par = s2_hat,  # Initial guess for sigma^2
  fn = log_likelihood_normal,
  mu = mu_hat,
  data = houses$price,
  method = "L-BFGS-B",
  lower = 1e-6  # Ensure sigma^2 > 0
)

Log_ML_mu = optim(
  par = mu_hat,  # Initial guess for sigma^2
  fn = log_likelihood_normal,
  sigma = s2_hat,
  data = houses$price,
  method = "L-BFGS-B",
  lower = 1e-6  # Ensure sigma^2 > 0
)
```

c)

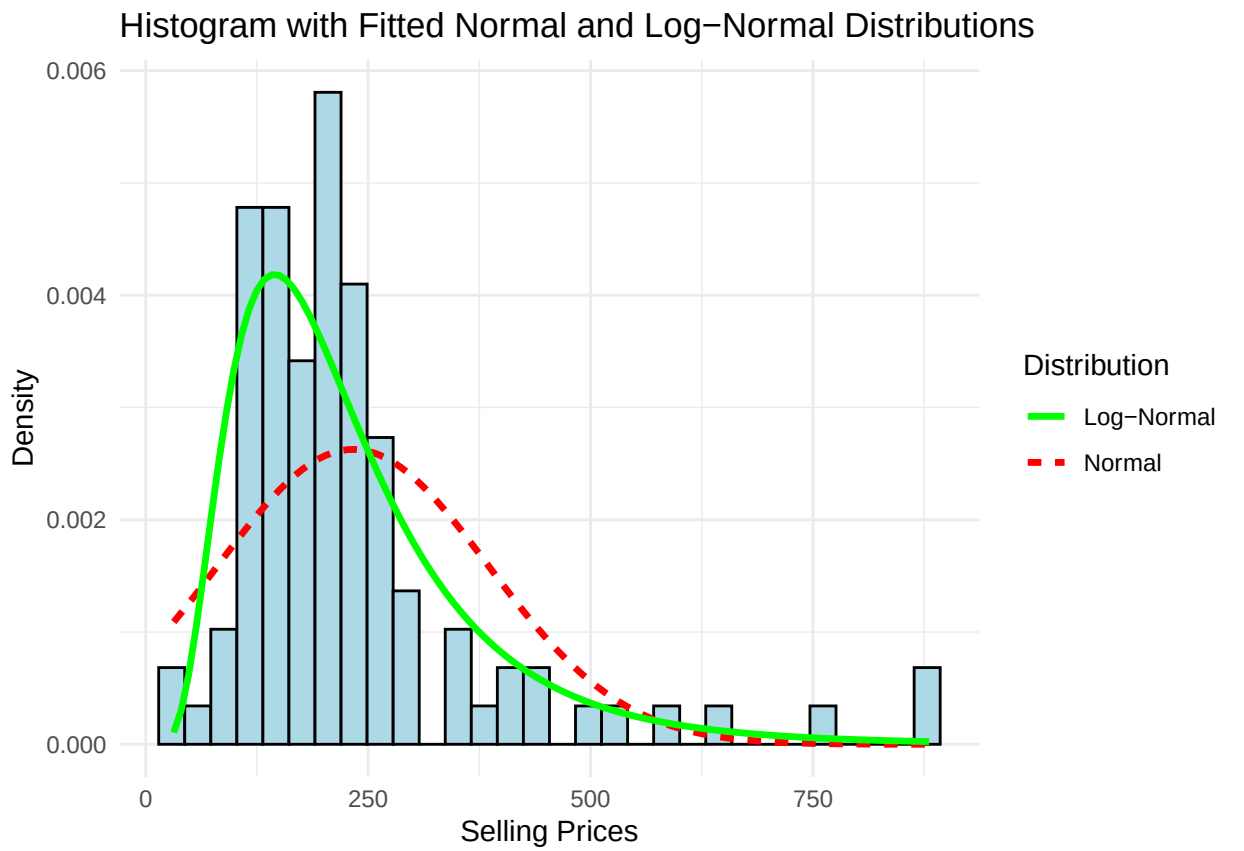
```r
comparison2 <- data.frame(
  Quantity = c("Mean", "Variance"),
  MLE_Function = c(Log_ML_mu$par, Log_ML_sigma$par),
  Built_in_Function = c(mu_hat, s2_hat)
)

comparison2
```

```
##   Quantity MLE_Function Built_in_Function
## 1     Mean     232.9965          232.9965
## 2 Variance   23071.5803        23071.5803
```

```r
# Load required libraries
library(ggplot2)

# Example data: Replace with your data
data <- houses$price

# Fit the Normal distribution
mu_normal <- mean(data)       # Mean
sigma_normal <- sd(data)      # Standard deviation

# Fit the Log-Normal distribution
log_data <- log(data)         # Log-transform the data
mu_log <- mean(log_data)      # Mean of log-transformed data
sigma_log <- sd(log_data)     # Std dev of log-transformed data
```

d)

```
# Histogram of the data
ggplot(data = data.frame(price = data), aes(x = price)) +
  geom_histogram(aes(y = ..density..), bins = 30, fill = "lightblue", color = "black") +

  # Add Normal PDF
  stat_function(
    fun = dnorm,
    args = list(mean = mu_normal, sd = sigma_normal),
    color = "red",
    size = 1.2,
    aes(linetype = "Normal")
  ) +

  # Add Log-Normal PDF
  stat_function(
    fun = function(x) dlnorm(x, meanlog = mu_log, sdlog = sigma_log),
    color = "green",
    size = 1.2,
    aes(linetype = "Log-Normal")
  ) +

  # Labels and Legend
  labs(
    title = "Histogram with Fitted Normal and Log-Normal Distributions",
    x = "Selling Prices",
    y = "Density",
    linetype = "Distribution"
  ) +
  theme_minimal()
```



It appears that the log-normal distribution fits the data better.

Ex 4.54

Consider n independent observations from an exponential pdf $f(y;\lambda) = \lambda e^{-\lambda y}$ for $y \geq 0$, with parameter $\lambda > 0$ for which $E(Y) = \frac{1}{\lambda}$.

(a) Find the sufficient statistic for estimating λ.
(b) Find the maximum likelihood estimator of λ and of $E(Y)$.
(c) One can show that $2\lambda(\sum_i Y_i)$ has a chi-squared distribution with $df = 2n$. Explain why $2\lambda(\sum_i Y_i)$ is a pivotal quantity, and use it to derive a 95% confidence interval for λ.

Solution

a)

To find the sufficient statistic for λ, we use the factorization theorem which states that a statistic T(y) is sufficient for a parameter λ if the joint p.d.f. can be factored in two parts: one part depends on the data y only through T(y) and λ and the other depends on the data y but not on λ.

The exponential p.d.f is:

$$f(y;\lambda) = \lambda e^{-\lambda y}, \quad y \geq 0$$

with $E[Y] = \frac{1}{\lambda}$

For n independent observations $y_1, y_2, \ldots, y_n$ the joint p.d.f is:

$$f(y_1, y_2, ..., y_n; \lambda) = \prod_{i=1}^{n} \lambda e^{-\lambda\ y_i}$$

$$f(y_1, y_2, ..., y_n; \lambda) = \lambda^n e^{-\lambda \sum_{i=1}^{n} y_i}$$

Now we can identify the sufficient statistic:

$$g(T(y), \lambda)\ h(y)$$

We can identify from the previous formula:

$T(y) = \sum_{i=1}^{n} y_i$, $g(T(y), \lambda) = \lambda^n e^{-\lambda \sum_{i=1}^{n} y_i}$ and $h(y) = 1$.

Thus, the sufficient statistic for λ is $\sum_{i=1}^{n} y_i$.

b)

The log-likelihood function is:

$$l(\lambda) = log(L(\lambda)) = log(f(y; \lambda)) = n\ log\ \lambda\ - \lambda \sum_{i=1}^{n} y_i$$

To find the MLE of λ we have to differentiate $l(\lambda)$ w.r.t λ and set it equal to 0.

$$\frac{\partial\ l(\lambda)}{\partial\ \lambda} = \frac{n}{y} - \sum_{i=1}^{n} y_i = 0$$

Solving for λ we obtain $\hat{\lambda} = \frac{n}{\sum_{i=1}^{n} y_i}$. This is our MLE.

As $E[Y] = \frac{1}{\lambda}$, it follows that

$$\hat{E}[Y] = \frac{1}{\hat{\lambda}} = \frac{\sum_{i=1}^{n} y_i}{n}$$

Which corresponds to the sample mean.

c)

A pivotal quantity is a function of the sample data and the parameter of interest that has a known probability distribution, independent of the parameter's actual value. In this case:

$$Q = 2\lambda(\sum_{i} Y_i)$$

It is pivotal because it is a function of the sample data $\sum_i Y_i$ and the parameter λ. Its distribution does not depend on λ and it follows a chi-squared distribution with 2 degrees of freedom.

$$Q \sim X^2_{2n}$$

This independence from λ makes Q a pivotal quantity.

To find the C.I. we set this equality:

$$P(q_{\alpha/2} \leq Q \leq q_{1-\alpha/2}) = 1 - \alpha$$

From the definition of Q:

$$P(q_{\alpha/2} \leq 2\lambda \sum_i Y_i \leq q_{1-\alpha/2}) = 1 - \alpha$$

We can therefore write

$$P(\chi^2_{2n,\frac{\alpha}{2}} < 2\lambda \sum Y_i < \chi^2_{2n,1-\frac{\alpha}{2}})$$

In conclusion, with some simple algebra, the 95% C.I. for λ is:

$$\lambda \in [\frac{\chi^2_{2n,\frac{\alpha}{2}}}{2\sum y_i}; \frac{\chi^2_{2n,1-\frac{\alpha}{2}}}{2\sum y_i}]$$

FSDS - Chapter 5

Ex 5.68

Explain why the confidence interval based on the Wald test of $H_0 : \theta = \theta_0$ is symmetric around $\hat{\theta}$ (i.e., having center exactly equal to $\hat{\theta}$. This is not true for the confidence intervals based on the likelihood-ratio and score tests.) Explain why such symmetry can be problematic when θ and $\hat{\theta}$ are near a boundary, using the example of a population proportion that is very close to 0 or 1 and a sample proportion that may well equal 0 or 1.

Solution

The Wald Confidence Interval is symmetric around $\hat{\theta}$ because it is build from the normal approximation:

$$C.I. = \hat{\theta} \pm z_{\alpha/2} SE(\hat{\theta})$$

The symmetry is supported from the assumption that the sampling distribution of $\hat{\theta}$ is approximately normal and centered in $\hat{\theta}$.

On the other hand, confidence intervals based on the likelihood-ratio test and score test are not necessarily symmetric. These methods are derived from the curvature of the likelihood function or deviations under the null hypothesis, which may lead to asymmetric intervals that depends on the properties of the data.

The symmetry of Wald type intervals becomes problematic when $\hat{\theta}$ is near a boundary (i.e. a proportion close to 0 or 1). If we imagine that the result of a C.I. sets the lower bound below 0, that would make no sense since a proportion cannot be negative. Another problem could arise when the upper bound is greater than 1, because the proportions are included in $[0, 1]$.

These issues arise because the Wald C.I. does not account for the constraints of the parameter space.

FSDS - Chapter 6

Ex 6.12

For the UN data file at the book's website (see Exercise 1.24), construct a multiple regression model predicting Internet using all the other variables. Use the concept of multicollinearity to explain why adjusted R^2 is not dramatically greater than when GDP is the sole predictor. Compare the estimated GDP effect in the bivariate model and the multiple regression model and explain why it is so much weaker in the multiple regression model.

Solution

```r
UN_url <- "https://stat4ds.rwth-aachen.de/data/UN.dat"

UN <- read.table(UN_url, header = TRUE)

# UN$Nation = as.factor(UN$Nation)
# excluding the nation column because of multicollinearity
fit = lm(Internet ~ . - Nation, data = UN)
summary(fit)
```

```
##
## Call:
## lm(formula = Internet ~ . - Nation, data = UN)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -23.819  -5.827  -2.182   7.166  26.354
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  11.158310  38.773097   0.288  0.77526
## GDP           0.440903   0.290680   1.517  0.13856
## HDI          55.851013  46.652218   1.197  0.23952
## GII         -72.428931  25.323061  -2.860  0.00719 **
## Fertility     4.092148   3.065379   1.335  0.19076
## CO2           0.310113   0.654899   0.474  0.63886
## Homicide      0.377324   0.299751   1.259  0.21668
## Prison        0.009091   0.018347   0.495  0.62344
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10.53 on 34 degrees of freedom
## Multiple R-squared:  0.8477, Adjusted R-squared:  0.8164
## F-statistic: 27.04 on 7 and 34 DF,  p-value: 3.947e-12
```

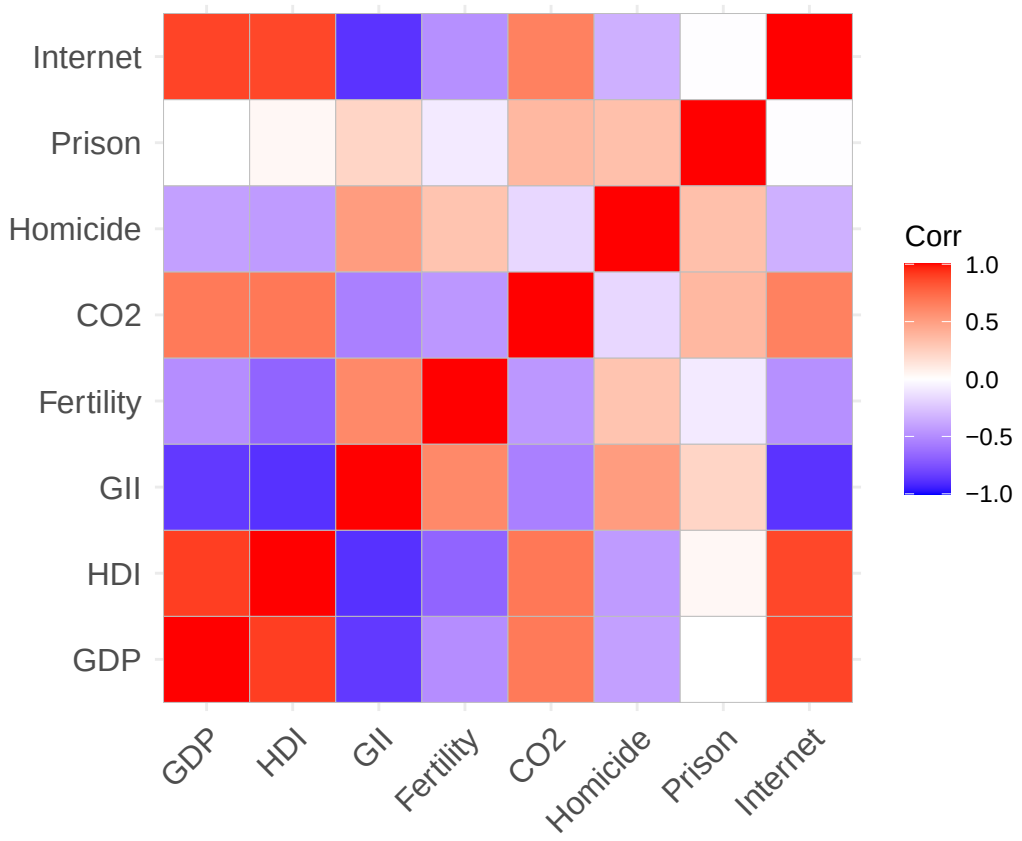
```r
fit_gdp = lm(Internet ~ GDP, data = UN)
summary(fit_gdp)
```

```
##
## Call:
## lm(formula = Internet ~ GDP, data = UN)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -27.130  -5.729   2.124  10.092  20.165
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  26.1341     3.7490   6.971 2.06e-08 ***
## GDP           1.4060     0.1217  11.555 2.55e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 11.95 on 40 degrees of freedom
## Multiple R-squared:  0.7695, Adjusted R-squared:  0.7637
## F-statistic: 133.5 on 1 and 40 DF,  p-value: 2.549e-14
```

Including all the other variables just marginally increases the R squared.

```
library(ggcorrplot)
corr_matrix = cor(UN[, -1])
ggcorrplot(corr_matrix)
```




```
# Find excessive correlations (excluding the diagonal)
excessive_corr <- which(abs(corr_matrix) > 0.70 & upper.tri(corr_matrix), arr.ind = TRUE)

# Print the pairs of predictors with excessive correlation
for (i in 1:nrow(excessive_corr)) {
  row_name <- rownames(corr_matrix)[excessive_corr[i, 1]]
  col_name <- colnames(corr_matrix)[excessive_corr[i, 2]]
  corr_value <- corr_matrix[excessive_corr[i, 1], excessive_corr[i, 2]]

  print(paste(row_name, "and", col_name, "are correlated with value:", round(corr_value, 2)))
}
```

```
## [1] "GDP and HDI are correlated with value: 0.9"
## [1] "GDP and GII are correlated with value: -0.85"
## [1] "HDI and GII are correlated with value: -0.88"
```

```
## [1] "GDP and Internet are correlated with value: 0.88"
## [1] "HDI and Internet are correlated with value: 0.87"
## [1] "GII and Internet are correlated with value: -0.87"
```

```
# how many variables are correlated with gdp and by how much?
corr_matrix["GDP", ]
```

```
##          GDP          HDI          GII    Fertility          CO2     Homicide
##  1.000000000  0.902973484 -0.850669343 -0.486158857  0.674469917 -0.407365069
##       Prison     Internet
## -0.002952338  0.877198714
```

In the complete model the effect of the main variable GDP is dispersed in lots of other less relevant variables. This however does not mean that the simpler model with just GDP is equal or better than the more complex model

```
c(AIC(fit), AIC(fit_gdp))
```

```
## [1] 326.0731 331.4895
```

```
anova(fit, fit_gdp, test = "Chisq")
```

```
## Analysis of Variance Table
##
## Model 1: Internet ~ (Nation + GDP + HDI + GII + Fertility + CO2 + Homicide +
##     Prison) - Nation
## Model 2: Internet ~ GDP
##   Res.Df    RSS Df Sum of Sq Pr(>Chi)
## 1     34 3770.1
## 2     40 5707.5 -6   -1937.4 0.007697 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

In fact, both ANOVA and AIC tests confirm that they are indeed significantly different in their prediction quality, with the simpler model performing worse.

Ex 6.14

The data set Crabs2 at the book's website comes from a study of factors that affect sperm traits of male horseshoe crabs. A response variable, SpermTotal, is the log of the total number of sperm in an ejaculate. It has $\bar{y} = 19.3$ and $s = 2.0$. The two explanatory variables used in the R output are the horseshoe crab's carapacewidth (CW, mean 18.6 cm, standard deviation 3.0 cm), which is a measure of its size, and color (1 = dark, 2 = medium, 3 = light), which is a measure of adult age, darker ones being older.

(a) Using the results shown, write the prediction equation and interpret the parameter estimates. (b) Explain the differences in what is tested with the F statistic (i) for the overall model, (ii) for the factor(Color) effect (iii) for the interaction term. Interpret each

Solution

```r
crabs2_url <- "https://stat4ds.rwth-aachen.de/data/Crabs2.dat"

crabs2 = read.table(crabs2_url, header = TRUE)
head(crabs2)
```

```
##   Location SpermTotal Ejacsize   CW Mass Color  OSR Condition
## 1   Sapelo      21.11     5.52 21.8 1500     2 1.89         8
## 2   Sapelo      22.21     5.82 23.4 1550     1 1.89         4
## 3   Sapelo      21.36     5.61 24.4 1650     2 1.89         7
## 4   Sapelo      20.80     5.41 21.4 1350     2 1.89         8
## 5   Sapelo      21.11     5.50 26.1 1900     2 1.89         5
## 6   Sapelo      17.46     2.86 23.8 1250     2 1.89         9
```

```r
crabs2$Color = as.factor(crabs2$Color)
```

a)

```r
fit_crab = lm(SpermTotal ~ CW + Color, data = crabs2)
summary(fit_crab)
```

```
##
## Call:
## lm(formula = SpermTotal ~ CW + Color, data = crabs2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.9180 -1.0040  0.0444  1.0342  4.2392
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 11.35896    0.63776  17.811  < 2e-16 ***
## CW           0.39115    0.03357  11.654  < 2e-16 ***
## Color2       0.80811    0.24563   3.290  0.00114 **
## Color3       1.14879    0.27105   4.238 3.15e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.585 on 254 degrees of freedom
## Multiple R-squared:  0.3941, Adjusted R-squared:  0.3869
## F-statistic: 55.06 on 3 and 254 DF,  p-value: < 2.2e-16
```

Prediction Equation:

$$SpermTotal = 11.3589620 + 0.3911538 \times CW + 0.8081103 \times Color2 + 1.1487937 \times Color3$$

Interpretation of the estimates:

- 1 extra centimeter of carapace width contributes to approximately $(1 - e^{0.3911538})100\%$ to sperm production.
- A crab having color 2 contributes $(1 - e^{0.8081103})100\%$ to sperm production
- A crab having color 3 contributes $(1 - e^{1.1487937})100\%$ to sperm production

b)

The F-statistic in an analysis of variance (ANOVA) or regression context is used to test different hypotheses depending on what aspect of the model is being examined.

```
anova(lm(SpermTotal ~ CW + factor(Color) + CW:factor(Color), data=crabs2))
```

```
## Analysis of Variance Table
##
## Response: SpermTotal
##                    Df Sum Sq Mean Sq  F value    Pr(>F)
## CW                  1 367.15  367.15 146.7098 < 2.2e-16 ***
## factor(Color)       2  47.93   23.96   9.5757 9.812e-05 ***
## CW:factor(Color)    2   7.61    3.80   1.5198    0.2207
## Residuals         252 630.65    2.50
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

(i) the test on the overall model checks whether it is significantly different from the Null model; more specifically if **at least one** predictor is significantly different from 0 (this is the case, as both CW and colour(s) are statistically significant).

(ii) Regarding the color effect, it checks whether there is a significant difference between different color groups.

(iii) Regarding the interaction effect, it tests whether there is a significant difference between color groups when taking into account the combined effect of Carapace weight and Color. In this case, the P value is > 0.2 so we can't reject H0 (there is no interaction effect).

Ex 6.30

When the values of y are multiplied by a constant c, from their formulas, show that s_y and $\hat{\beta}_1$ in the bivariate linear model are also then multiplied by c. Thus, show that $r = \hat{\beta}_1(\frac{s_x}{s_y})$ does not depend on the units of measurement.

Solution

Given the bivariate linear model $\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_i$ and a constant c s.t. $y_i^* = cy_i$, we have that:

$$s_{y^*} = \sum_{i=1}^{n}(y_i^* - \bar{y}^*) = \sum_{i=1}^{n}(cy_i - c\bar{y}) = c\sum_{i=1}^{n}(y_i - \bar{y}) = cs_y$$

and:

$$\hat{\beta}_1^* = \frac{s_{xy^*}}{s_x^2} = \frac{\sum_{i=1}^{n}(y_i^* - \bar{y}^*)(x_i - \bar{x})}{\sum_{i=1}^{n}(x_i - \bar{x})^2} = \frac{\sum_{i=1}^{n} c(y_i - \bar{y})(x_i - \bar{x})}{\sum_{i=1}^{n}(x_i - \bar{x})^2} = c\hat{\beta}_1$$

Finally, given u_1 and u_2 constants that represent different units of measurement respectively for x and y, s.t. $x^* = u_1 x; \quad y^* = u_2 y$

$$r^* = \hat{\beta}_1^*(s_{x^*}/s_{y^*}) = \frac{s_{x^*y^*}s_{x^*}}{s_{x^*}^2 s_{y^*}} = \frac{\sum_{i=1}^{n}(y_i^* - \bar{y}^*)(x_i^* - \bar{x}^*)\sum_{i=1}^{n}(x_i^* - \bar{x}^*)}{\sum_{i=1}^{n}(x_i^* - \bar{x}^*)^2 \sum_{i=1}^{n}(y_i^* - \bar{y}^*)} =$$
$$= \frac{\sum_{i=1}^{n} u_1 u_2 (y_i - \bar{y})(x_i - \bar{x})\sum_{i=1}^{n} u_1(x_i - \bar{x})}{\sum_{i=1}^{n} u_1^2(x_i - \bar{x})^2 \sum_{i=1}^{n} u_2(y_i - \bar{y})} = \hat{\beta}_1(s_x/s_y) = r$$

Therefore we can conclude that r does not depend on the units of measurement.

Ex 6.42

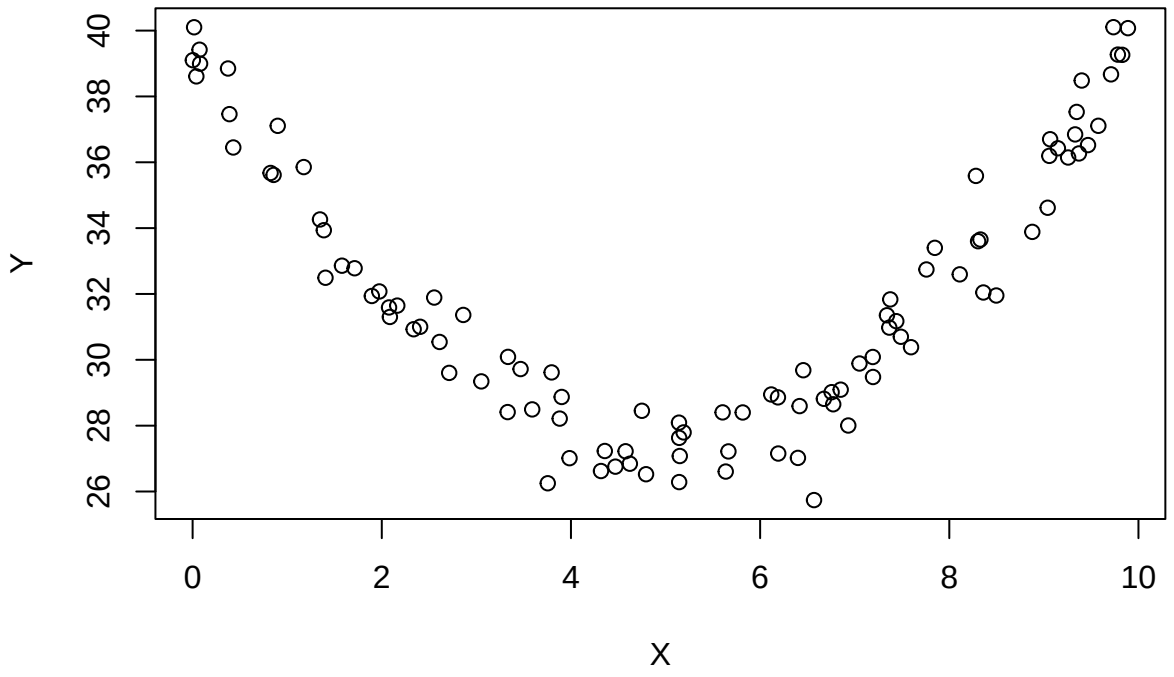
You can fit the quadratic equation $E(Y) = \beta_0 + \beta_1 x + \beta_2 x^2$ *by fitting a multiple regression model with* $x_1 = x$ *and* $x_2 = x^2$.

(a) Simulate 100 independent observations from the model $Y = 40.0 - 5.0x + 0.5x^2 + \epsilon$, *where X has a uniform distribution over [0, 10] and* $\epsilon \sim N(0, 1)$. *Plot the data and fit the quadratic model. Report how the fitted equation compares with the true relationship (b) Find the correlation between x and y and explain why it is so weak even though the plot shows a strong relationship with a large* R^2 *value for the quadratic model.*

Solution

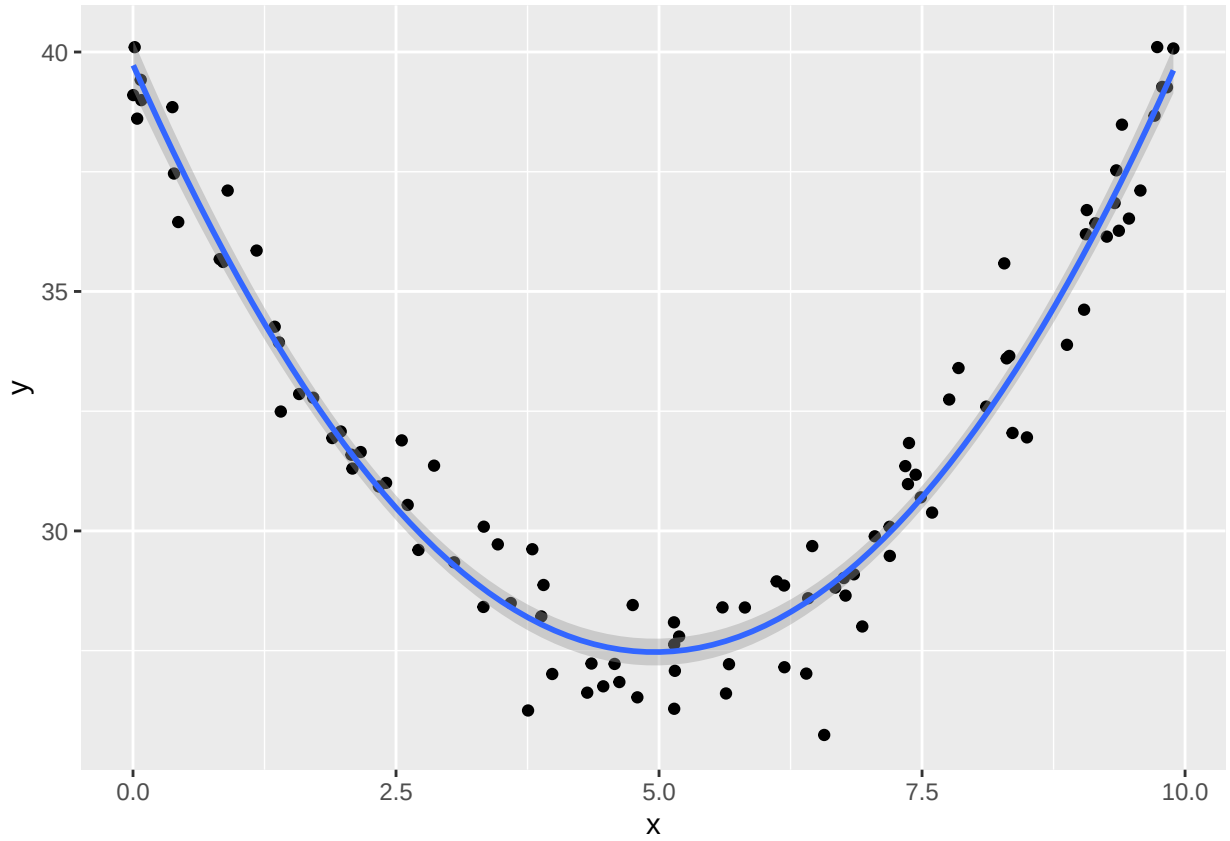
a)

```
set.seed(42) # For reproducibility

n = 100
X = runif(n, min = 0, max = 10)
epsilon = rnorm(n, 0, 1)
Y = 40.0 - 5.0*X+0.5* X^2 + epsilon
plot(X, Y)
```




```
data = data.frame(y = Y, x = X)
fit_quadratic <- lm(y ~ poly(x, 2, raw = TRUE), data = data)

#Install & load ggplot2
```

```
library("ggplot2")

# Create basic ggplot
# and Add regression line
ggp <- ggplot(data, aes(x, y)) +
  geom_point()
ggp = ggp +
  geom_smooth(method = "lm",
              formula = y ~ poly(x, 2, raw = TRUE))
ggp
```



The regression line with the quadratic term fits the data very well even though it ignores the other terms. The quadratic term prevails as X grows, resulting in the model being an overall good fit even if some points are spread around the regression line and not exactly on it.

b)

First of all, we print the summary of the model and the correlation between X and Y:

```
print(summary(fit_quadratic))
```

```
##
## Call:
## lm(formula = y ~ poly(x, 2, raw = TRUE), data = data)
##
```

```
## Residuals:
##      Min       1Q   Median       3Q      Max
## -3.03050 -0.51696  0.05156  0.57232  2.59598
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)              39.73423    0.26512  149.87   <2e-16 ***
## poly(x, 2, raw = TRUE)1 -4.94899    0.12079  -40.97   <2e-16 ***
## poly(x, 2, raw = TRUE)2  0.49926    0.01162   42.96   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9303 on 97 degrees of freedom
## Multiple R-squared:  0.9502, Adjusted R-squared:  0.9492
## F-statistic: 925.1 on 2 and 97 DF,  p-value: < 2.2e-16
```

```
correlation <- cor(X, Y)
print(correlation)
```

```
## [1] 0.04878872
```

The correlation is so low (0.0487887) despite the large R^2 value of the model because Pearson's correlation coefficient can't capture non-linear relationships.

Ex 6.52

F statistics have alternate expressions in terms of R^2 values.

(a) Show that for testing $H_0 : \beta_1 = ... = \beta_p = 0$,

$$F = \frac{(TSS - SSE)/p}{SSE/[n-(p+1)]}$$

is equivalent to:

$$F = \frac{R^2/p}{(1-R^2)/[n-(p+1)]}$$

Explain why larger values of R^2 yield larger values of F.

(b) Show that for comparing nested linear models,

$$F = \frac{(SSE_0 - SSE_1)/(p_1 - p_0)}{SSE_1/[n-(p_1+1)]} \quad = \quad \frac{R_1^2 - R_0^2/(p_1-p_0)}{(1-R_1^2)/[n-(p_1+1)]}$$

Solution

a)

F-statistic for testing $H_0 : \beta_1 = ... = \beta_p = 0$ is given by:

$$F = \frac{Explained\ Variance\ per\ Predictor}{Unexplained\ Variance\ per\ Residual\ Degree\ of\ Freedom}$$

This is expressed as:

$$F = \frac{(TSS - SSE)/p}{SSE/[n-(p+1)]}$$

We can relate R^2 to TSS and SSE:

$$R^2 = \frac{TSS - SSE}{TSS} \quad => \quad TSS - SSE = R^2\ TSS$$

The proportion of unexplained variance is: $1 - R^2 = \frac{SSE}{TSS}$.

And we can express SSE as: $SSE = (1 - R^2)\ TSS$.

We can now substitute this results into F-statistics formula:

$$F = \frac{(R^2\ TSS)\ /p}{[(1-R^2)\ TSS]\ /[n-(p+1)]}$$

Simplifying the TSS term in the numerator and denominator, we obtain:

$$F = \frac{R^2\ /p}{(1-R^2)\ \ /[n-(p+1)]}$$

Rearranging:

$$F = \frac{R^2}{p}\ \frac{n-(p+1)}{1-R^2}$$

The result we have obtained is equal to:

$$F = \frac{(TSS - SSE)/p}{SSE/[n-(p+1)]}$$

A larger proportion of explained variance (higher R^2) results in a more significant F-value, indicating stronger evidence against H_0.

b)

From the previous point:

$$R^2 = 1 - \frac{SSE}{TSS}$$

For the reduced (0 index) and the full model (1 index):

$$R_0^2 = 1 - \frac{SSE_0}{TSS} \quad and \quad R_1^2 = 1 - \frac{SSE_1}{TSS}$$

The reduction in SSE when moving from the reduced model to the full model can be expressed in terms of R^2:

$$SSE_0 - SSE_1 = (R_1^2 - R_0^2)\ TSS$$

We can substitute into the numerator of the F-statistics formula:

$$\frac{SSE_0 - SSE_1}{p_1 - p_0} = \frac{(R_1^2 - R_0^2)\ TSS}{p_1 - p_0}$$

Where p_1 and p_0 are the number of predictors in the full model and reduced models, respectively.

For the full model, the denominator becomes:

$$\frac{SSE_1}{n-(p_1+1)} = \frac{(1-R_1^2)\ TSS}{n-(p_1+1)}$$

We can combine numerator and denominator and obtain:

$$F = \frac{(SSE_0 - SSE_1)/(p_1 - p_0)}{SSE_1/[n-(p_1+1)]} \quad = \quad \frac{R_1^2 - R_0^2/(p_1-p_0)}{(1-R_1^2)/[n-(p_1+1)]}$$

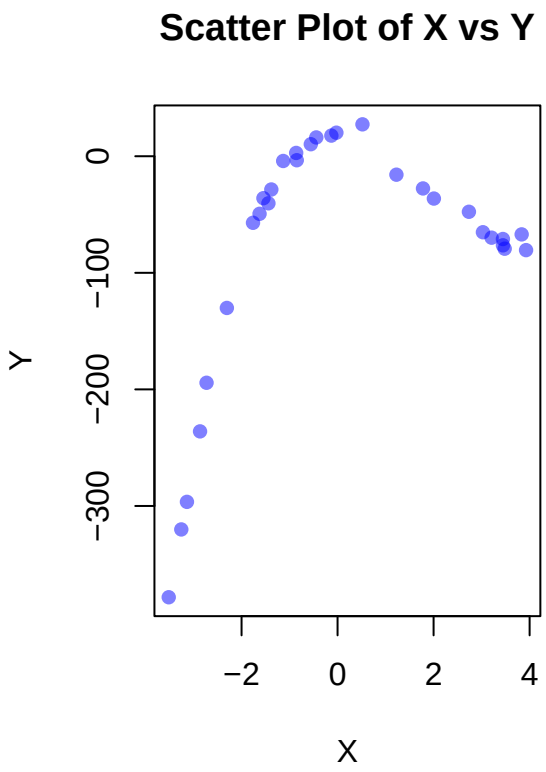
FSDS - Chapter 7

Ex 7.4

Analogously to the previous exercise, randomly sample 30 X observations from a uniform in the interval (-4,4) and conditional on $X = x$, 30 normal observations with $E(Y) = 3.5x^3 - 20x^2 + 0.5x + 20$ and $\sigma = 30$. Fit polynomial normal GLMs of lower and higher order than that of the true relationship. Which model would you suggest? Repeat the same task for $E(Y) = 0.5x^3 - 20x^2 + 0.5x + 20$ (same σ) several times. What do you observe? Which model would you suggest now?

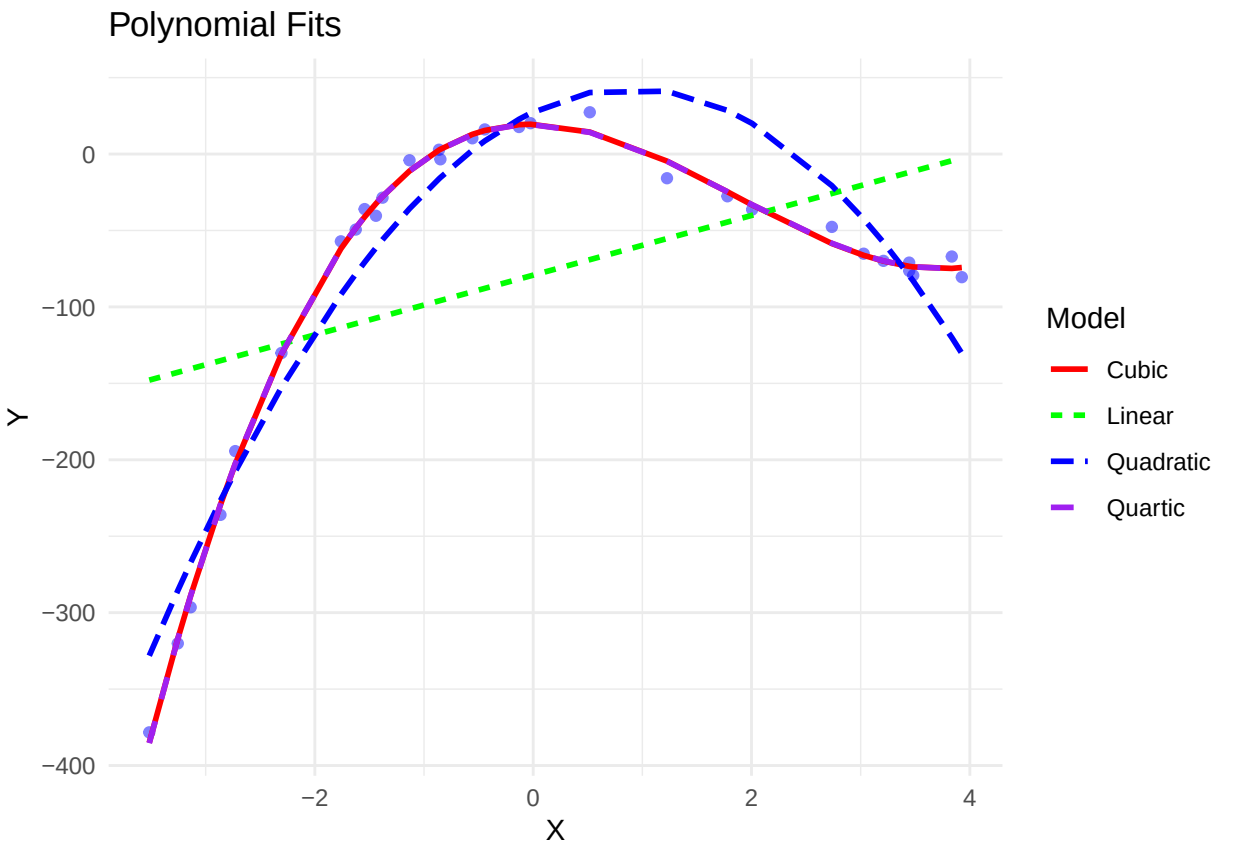
Solution

```
n = 30
sigma = 30
X = runif(n, min = -4, max = 4)
mu_y = 3.5*X^3 - 20*X^2 + 0.5*X + 20

E_Y = sapply(mu_y, function(mu) rnorm(n, mean = mu, sd = sigma))
Y = colMeans(E_Y)

par(mfrow = c(1, 2))

plot(X, Y, pch = 16, col = rgb(0, 0, 1, 0.5),
     xlab = "X", ylab = "Y", main = "Scatter Plot of X vs Y")

# Fit polynomial models
glm1 = glm(Y ~ X, family = gaussian())                # Linear
glm2 = glm(Y ~ poly(X, 2), family = gaussian())       # Quadratic
glm3 = glm(Y ~ poly(X, 3), family = gaussian())       # Cubic
glm4 = glm(Y ~ poly(X, 4), family = gaussian())       # Quartic
```




```r
# Predictions for each model
pred1 = predict(glm1, newdata = data.frame(X))
pred2 = predict(glm2, newdata = data.frame(X))
pred3 = predict(glm3, newdata = data.frame(X))
pred4 = predict(glm4, newdata = data.frame(X))

# Create a data frame for predictions
pred_data <- data.frame(
  X = X,
  Y = c(pred1, pred2, pred3, pred4),
  Model = factor(rep(c("Linear", "Quadratic", "Cubic", "Quartic"), each = length(X)))
)

data = data.frame(X, Y)
# Step 2: Create the base plot
ggplot(data, aes(x = X, y = Y)) +
  geom_point(color = "blue", alpha = 0.5) +  # Scatter plot of X vs Y
  geom_line(data = pred_data, aes(x = X, y = Y, color = Model, linetype = Model), size = 1) +
  scale_color_manual(values = c("red", "green", "blue", "purple")) +  # Manual color mapping
  labs(
    title = "Polynomial Fits",
    x = "X",
    y = "Y",
    color = "Model",
    linetype = "Model"
  ) +
```

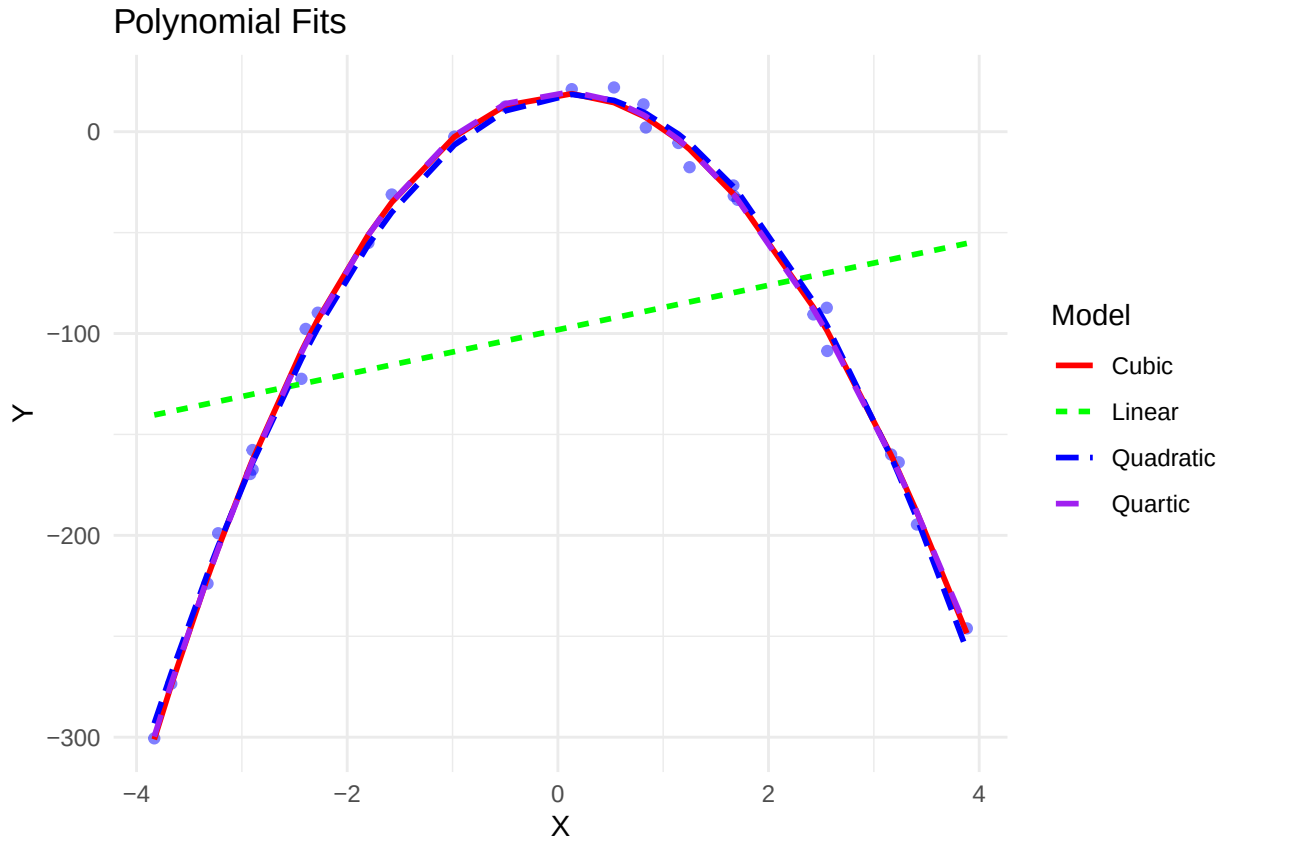
```
  theme_minimal()  # Use a clean theme
```



As we can see, the quartic model (purple line) seems to fit the data best, whereas the quadratic and linear models appear incapable of capturing the complexity of the relationship. Choosing a model such as the quartic one, or of higher degree, would just introduce unnecessary complexity in explaining the relationship, and significantly imbalance the bias-variance tradeoff.

```
n = 30
sigma = 30
X = runif(n, min = -4, max = 4)
mu_y = 0.5*X^3 - 20*X^2 + 0.5*X + 20

E_Y = sapply(mu_y, function(mu) rnorm(n, mean = mu, sd = sigma))
Y = colMeans(E_Y)

# Fit polynomial models
glm1 = glm(Y ~ X, family = gaussian())                # Linear
glm2 = glm(Y ~ poly(X, 2), family = gaussian())       # Quadratic
glm3 = glm(Y ~ poly(X, 3), family = gaussian())       # Cubic
glm4 = glm(Y ~ poly(X, 4), family = gaussian())       # Quartic

# Predictions for each model
pred1 = predict(glm1, newdata = data.frame(X))
pred2 = predict(glm2, newdata = data.frame(X))
pred3 = predict(glm3, newdata = data.frame(X))
pred4 = predict(glm4, newdata = data.frame(X))
```

```r
# Create a data frame for predictions
pred_data <- data.frame(
  X = X,
  Y = c(pred1, pred2, pred3, pred4),
  Model = factor(rep(c("Linear", "Quadratic", "Cubic", "Quartic"), each = length(X)))
)

data = data.frame(X, Y)
# Step 2: Create the base plot
ggplot(data, aes(x = X, y = Y)) +
  geom_point(color = "blue", alpha = 0.5) +  # Scatter plot of X vs Y
  geom_line(data = pred_data, aes(x = X, y = Y, color = Model, linetype = Model), size = 1) +
  scale_color_manual(values = c("red", "green", "blue", "purple")) +  # Manual color mapping
  labs(
    title = "Polynomial Fits",
    x = "X",
    y = "Y",
    color = "Model",
    linetype = "Model"
  ) +
  theme_minimal()  # Use a clean theme
```



Now all models but the linear one fit the data almost perfectly: the low coefficient on the cubic term has made it almost irrelevant in the interval, compared to the quadratic coefficient.

Ex 7.20

In the Crabs data file introduced in Section 7.4.2, the variable y indicates whether a female horseshoe crab has at least one satellite (1 = yes, 0 = no).

(a) Fit a main-effects (no interaction terms) logistic model using weight and categorical color as explanatory variables. Conduct a significance test for the color effect, and construct a 95% confidence interval for the weight effect. (b) Fit the model that permits interaction between color as a factor and weight in their effects, showing the estimated effect of weight for each color. Test whether this model provides a significantly better fit. (c) Use AIC to determine which models seem most sensible among the models with (i) interaction, (ii) main effects, (iii) weight as the sole predictor, (iv) color as the sole predictor, and (v) the null model.

Solution

```
crabs_url <- "https://stat4ds.rwth-aachen.de/data/Crabs.dat"

crabs = read.table(crabs_url, header = TRUE)

head(crabs)
```

```
##   crab sat y weight width color spine
## 1    1   8 1   3.05  28.3     2     3
## 2    2   0 0   1.55  22.5     3     3
## 3    3   9 1   2.30  26.0     1     1
## 4    4   0 0   2.10  24.8     3     3
## 5    5   4 1   2.60  26.0     3     3
## 6    6   0 0   2.10  23.8     2     3
```

a)

Using the glm function, we build a logistic regression model and we obtain a summary.

```
fit = glm(y ~ weight + color, family = binomial(link = "logit"), data = crabs)
summary(fit)
```

```
##
## Call:
## glm(formula = y ~ weight + color, family = binomial(link = "logit"),
##     data = crabs)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -2.0316     1.1161  -1.820   0.0687 .
## weight        1.6531     0.3825   4.322 1.55e-05 ***
## color        -0.5142     0.2234  -2.302   0.0213 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 225.76  on 172  degrees of freedom
## Residual deviance: 190.27  on 170  degrees of freedom
## AIC: 196.27
##
## Number of Fisher Scoring iterations: 4
```

Significance test for the color effect:

```
anova_model <- anova(fit, test = "Chisq")
cat("Significativity test for color:\n")
```

```
## Significativity test for color:
```

```
print(anova_model)
```

```
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: y
##
## Terms added sequentially (first to last)
##
##
##        Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
## NULL                     172     225.76
## weight  1  30.0214       171     195.74 4.273e-08 ***
## color   1   5.4684       170     190.27   0.01936 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Although the effect is less strong than in weight, the color effect is still significant ($p < 0.05$). This suggests that color contributes to improved model fit and explains some of the variability in response.

Building the confidence interval:

```
CI_weight = fit$coefficients[2] + c(-1,1)*qnorm(0.975)*(summary(fit)$coefficients[, "Std. Error"][2])
CI_weight
```

```
## [1] 0.903438 2.402665
```

```
# check
CI_weight_function <- confint(fit, param="weight")
```

```
## Waiting for profiling to be done...
```

```
CI_weight_function
```

```
##                  2.5 %      97.5 %
## (Intercept) -4.2812920  0.11668655
## weight       0.9381057  2.44452879
## color       -0.9625119 -0.08254975
```

```
# the intervals match!
```

b)

Fitting the model that permits interaction between color as a factor and weight in their effects.

```r
fit1 = glm(y ~ weight + factor(color) + weight:factor(color), family = binomial(link = "logit"), data =
summary(fit1)
```

```
##
## Call:
## glm(formula = y ~ weight + factor(color) + weight:factor(color),
##     family = binomial(link = "logit"), data = crabs)
##
## Coefficients:
##                        Estimate Std. Error z value Pr(>|z|)
## (Intercept)             -1.6203     4.8909  -0.331    0.740
## weight                   1.0483     1.8929   0.554    0.580
## factor(color)2          -0.8320     5.0311  -0.165    0.869
## factor(color)3          -6.2964     5.5165  -1.141    0.254
## factor(color)4           0.4335     5.4046   0.080    0.936
## weight:factor(color)2    0.3613     1.9559   0.185    0.853
## weight:factor(color)3    2.7065     2.2284   1.215    0.225
## weight:factor(color)4   -0.8536     2.1551  -0.396    0.692
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 225.76  on 172  degrees of freedom
## Residual deviance: 181.66  on 165  degrees of freedom
## AIC: 197.66
##
## Number of Fisher Scoring iterations: 5
```

```r
c(AIC(fit), AIC(fit1))
```

```
## [1] 196.2688 197.6563
```

```r
anova(fit1, fit, test = "LRT")
```

```
## Analysis of Deviance Table
##
## Model 1: y ~ weight + factor(color) + weight:factor(color)
## Model 2: y ~ weight + color
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1       165     181.66
## 2       170     190.27 -5  -8.6125   0.1256
```

The extended model scores a slightly lower deviance, but it doesn't significantly improve with the AIC. Also with LRT, the difference between the 2 models is not significant even at the 10% level. We thus prefer the smaller and simpler model.

c)

We tested various models that seem the most sensible. In particular, we tested the model with interaction, the main model (fit), the model with weights as the only predictor, color as the only predictor and the null model.

```r
# Fit the models
model_interaction <- glm(y ~ weight * color, family = binomial, data = crabs)
model_main <- glm(y ~ weight + color, family = binomial, data = crabs)
model_weight <- glm(y ~ weight, family = binomial, data = crabs)
model_color <- glm(y ~ color, family = binomial, data = crabs)
model_null <- glm(y ~ 1, family = binomial, data = crabs)
```

Comparing the AIC

```r
aic_values <- AIC(model_interaction, model_main, model_weight, model_color, model_null)
print(aic_values)
```

```
##                   df      AIC
## model_interaction  4 198.1897
## model_main         3 196.2688
## model_weight       2 199.7371
## model_color        2 217.2979
## model_null         1 227.7585
```

The main effects models scores the lowest AIC criteria, and is therefore the best model.

Anova

```r
anova(model_null, model_color, model_weight, model_main, model_interaction, test = "Chisq")
```

```
## Analysis of Deviance Table
##
## Model 1: y ~ 1
## Model 2: y ~ color
## Model 3: y ~ weight
## Model 4: y ~ weight + color
## Model 5: y ~ weight * color
##   Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
## 1       172     225.76
## 2       171     213.30  1  12.4607 0.0004156 ***
## 3       171     195.74  0  17.5607
## 4       170     190.27  1   5.4684 0.0193637 *
## 5       169     190.19  1   0.0791 0.7785110
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The best model is the main effects one (4), as it represents a good balance between simplicity and explanatory power. The addition of both predictors significantly reduces residual deviance compared with the simpler models, indicating that both weight and color help to explain variability in the response variable. However, the inclusion of an interaction term between the two predictors does not make a significant improvement, suggesting that their effects are mainly additive. Therefore, the model with only main effects effectively balances complexity and statistical significance, respecting the principle of parsimony.

Ex 7.26

A headline in The Gainesville Sun (Feb. 17, 2014) proclaimed a worrisome spike in shark attacks in the previous two years. The reported total number of shark attacks in Florida per year from 2001 to 2013 were

33, 29, 29, 12, 17, 21, 31, 28, 19, 14, 11, 26, 23. Are these counts consistent with a null Poisson model? Explain, and compare aspects of the Poisson model and negative binomial model fits.

Solution

Poisson model

```
shark_attacks <- c(33, 29, 29, 12, 17, 21, 31, 28, 19, 14, 11, 26, 23)

poisson_model <- glm(shark_attacks ~ 1, family = poisson(link = "log"))
summary(poisson_model)
```

```
##
## Call:
## glm(formula = shark_attacks ~ 1, family = poisson(link = "log"))
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  3.11522    0.05842   53.32   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 31.392  on 12  degrees of freedom
## Residual deviance: 31.392  on 12  degrees of freedom
## AIC: 97.129
##
## Number of Fisher Scoring iterations: 4
```

```
#dispersion
dispersion <- sum(residuals(poisson_model, type = "deviance")^2) / df.residual(poisson_model)
dispersion
```

```
## [1] 2.616028
```

```
residual_deviance <- poisson_model$deviance
residual_df <- poisson_model$df.residual
p_value <- pchisq(residual_deviance, residual_df, lower.tail = FALSE)

cat("Residual Deviance:", residual_deviance, "\n")
```

```
## Residual Deviance: 31.39234
```

```
cat("Residual DF:", residual_df, "\n")
```

```
## Residual DF: 12
```

```
cat("P-value :", p_value, "\n")
```

```
## P-value : 0.001715944
```

The residual deviance is significantly different from the expected deviance. Poisson's model may not be suitable. Moreover, the value obtained from pchisq is very low, indicating that the data at hand cannot come from a Null Poisson model.

Negative Binomial

```
library(MASS)
```

```
##
## Caricamento pacchetto: 'MASS'

## Il seguente oggetto è mascherato _per_ '.GlobalEnv':
##
##     crabs
```

```
model_nb <- glm.nb(shark_attacks ~ 1)

summary(model_nb)
```

```
##
## Call:
## glm.nb(formula = shark_attacks ~ 1, init.theta = 15.49441181,
##     link = log)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  3.11522    0.09153   34.03   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Negative Binomial(15.4944) family taken to be 1)
##
##     Null deviance: 13.363  on 12  degrees of freedom
## Residual deviance: 13.363  on 12  degrees of freedom
## AIC: 92.608
##
## Number of Fisher Scoring iterations: 1
##
##
##               Theta:  15.5
##           Std. Err.:  10.5
##
##  2 x log-likelihood:  -88.608
```

```
pchisq(model_nb$deviance, poisson_model$df.residual, lower.tail = F)
```

```
## [1] 0.3432158
```

In the case of the negative binomial, from the observation of the p-value, we cannot reject the null hypothesis that the counts at hand originate from such a distribution.

```r
#aic comparison
aic_comparison <- AIC(poisson_model, model_nb)
cat("Comparison AIC:\n")
```

```
## Comparison AIC:
```

```r
print(aic_comparison)
```

```
##                df      AIC
## poisson_model   1 97.12863
## model_nb        2 92.60803
```

AIC of the negative binomial is lower than the Poisson, thus indicating a better fit.