Unit IV

ECON 3406

Dr. Josh Martin

Introduction to linear regression

$$y = \beta_0 + \beta_1 x + \epsilon$$

- Linear regression is the statistical method for fitting a line to data where the relationship between two variables, x and y, can be modeled by a straight line with some error
  - β_0 = y-intercept
  - β_1 = slope
  - ϵ = error term (a.k.a residual)

Introduction to linear regression

- When we use x to predict y, we usually call:
  - x the explanatory, predictor or independent variable,
  - y the response or dependent variable
- As we move forward in this chapter, we will learn about:
  - criteria for line-fitting
  - the uncertainty associated with estimates of model parameters

Introduction to linear regression

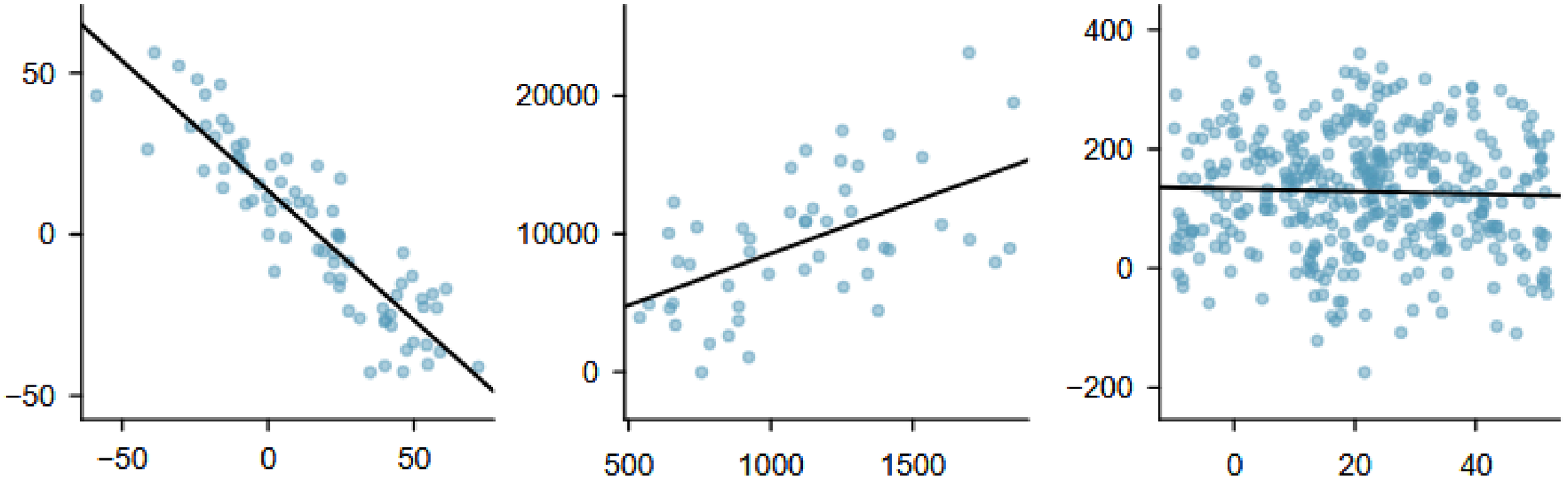


Figure 8.2: Three data sets where a linear model may be useful even though the data do not all fall exactly on the line.

Introduction to linear regression

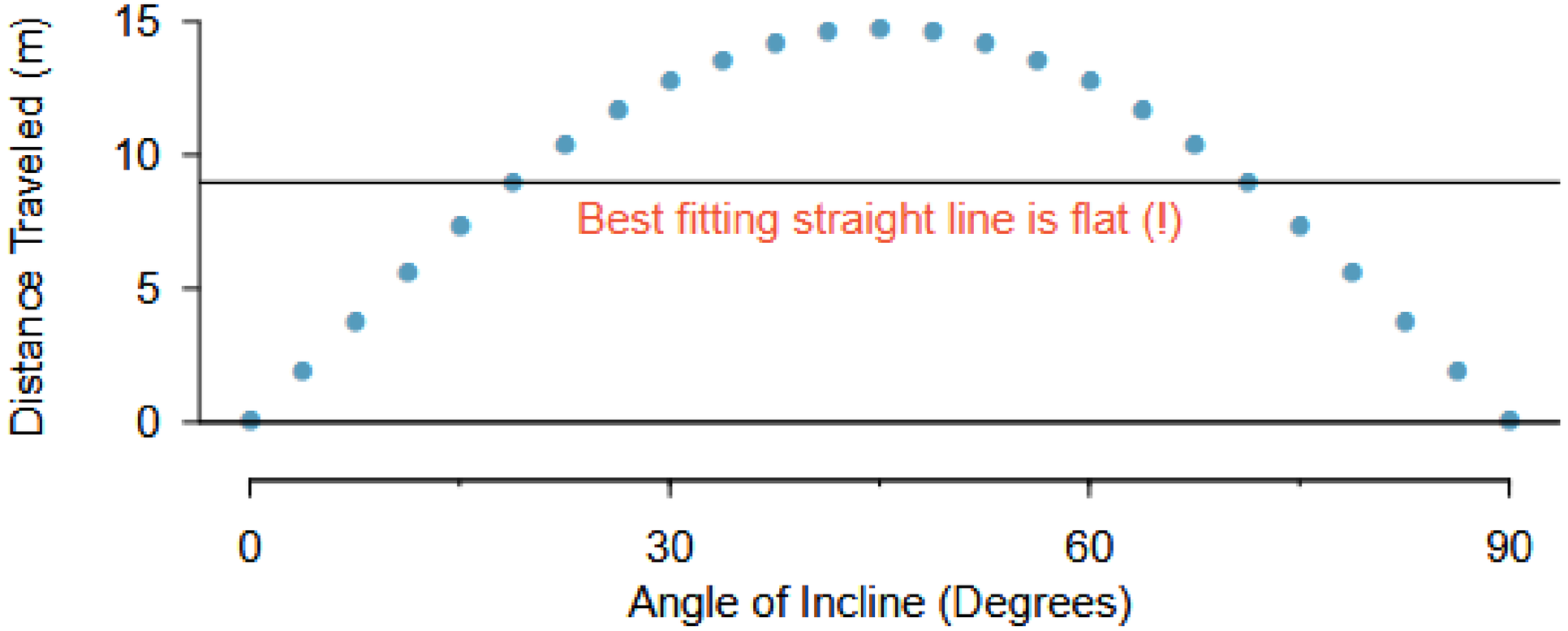


Figure 8.3: A linear model is not useful in this nonlinear case. These data are from an introductory physics experiment.

World's most useless regression

Opossums...?

```
1 x <- read.csv("possum.csv")
2 head(x)
```

```
  site pop sex age head_l skull_w total_l tail_l
1    1 Vic   m   8   94.1    60.4    89.0   36.0
2    1 Vic   f   6   92.5    57.6    91.5   36.5
3    1 Vic   f   6   94.0    60.0    95.5   39.0
4    1 Vic   f   6   93.2    57.1    92.0   38.0
5    1 Vic   f   2   91.5    56.3    85.5   36.0
6    1 Vic   f   1   93.1    54.8    90.5   35.5
```

World's most useless regression

```
lm1 <- lm(head_l ~ total_l, x)
coefficients(lm1)
```

```
(Intercept)     total_l
 42.7097931   0.5729013
```

World's most useless regression

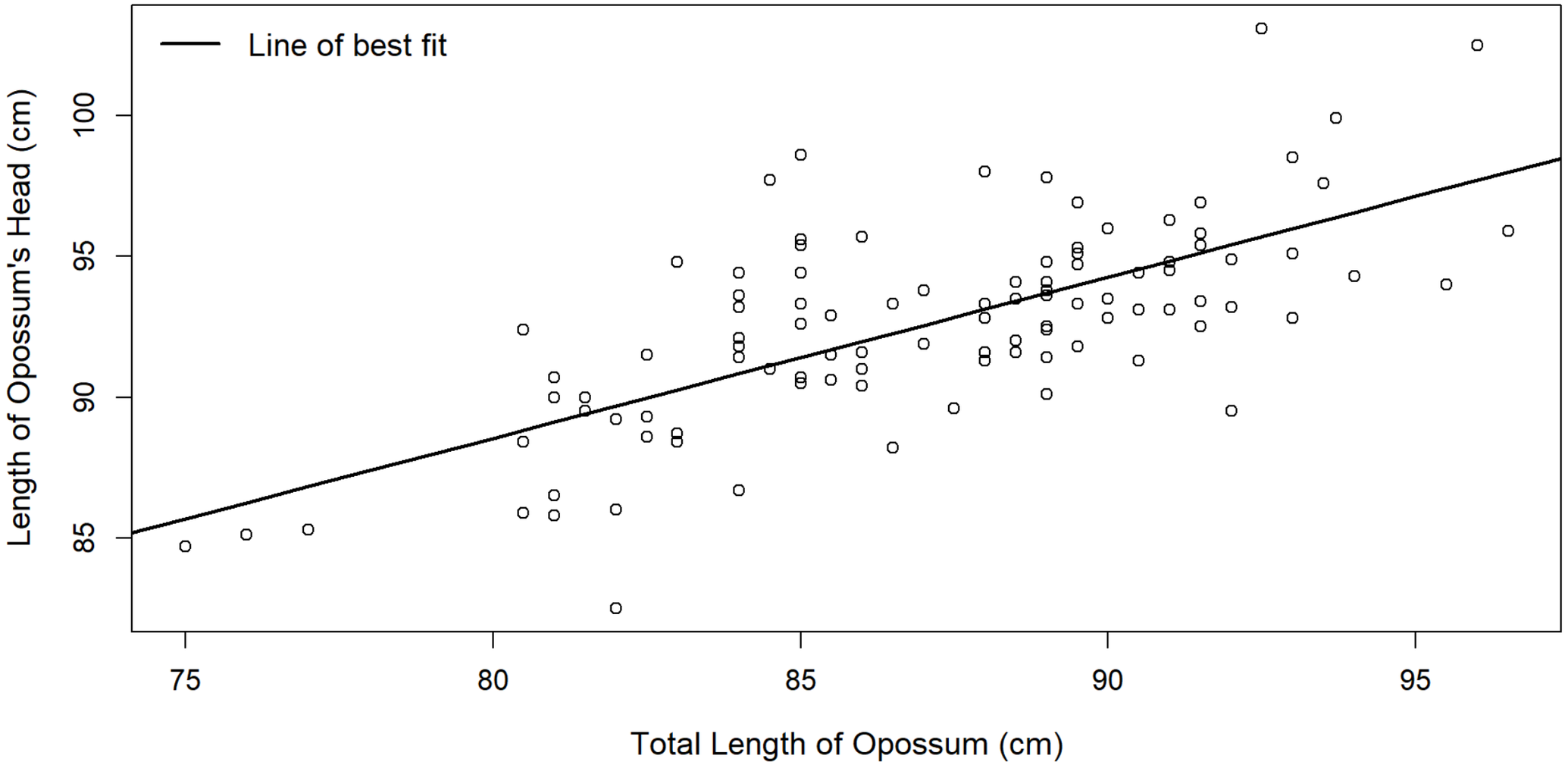
```
summary(x$total_l)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  75.00   84.00   88.00   87.09   90.00   96.50 
```

```
xs <- seq(70, 100, 1)

b0 <- coefficients(lm1)[1]
b1 <- coefficients(lm1)[2]
y_hat <- b0 + b1 * xs

head(
  data.frame(
  x = xs,
  y_hat = round(y_hat, 1)
  )
)
```

```
   x y_hat
1 70  82.8
2 71  83.4
3 72  84.0
4 73  84.5
5 74  85.1
6 75  85.7
```

World's most useless regression



Residuals

$$Data = Fit + Residual$$

- Residuals are the leftover variation after accounting fo the model fit
- The residual for the i^{th} observation (x_i, y_i) is the difference of the observed response (y_i) and the response that we would predict based on the model fit ($\hat{y}_i$)

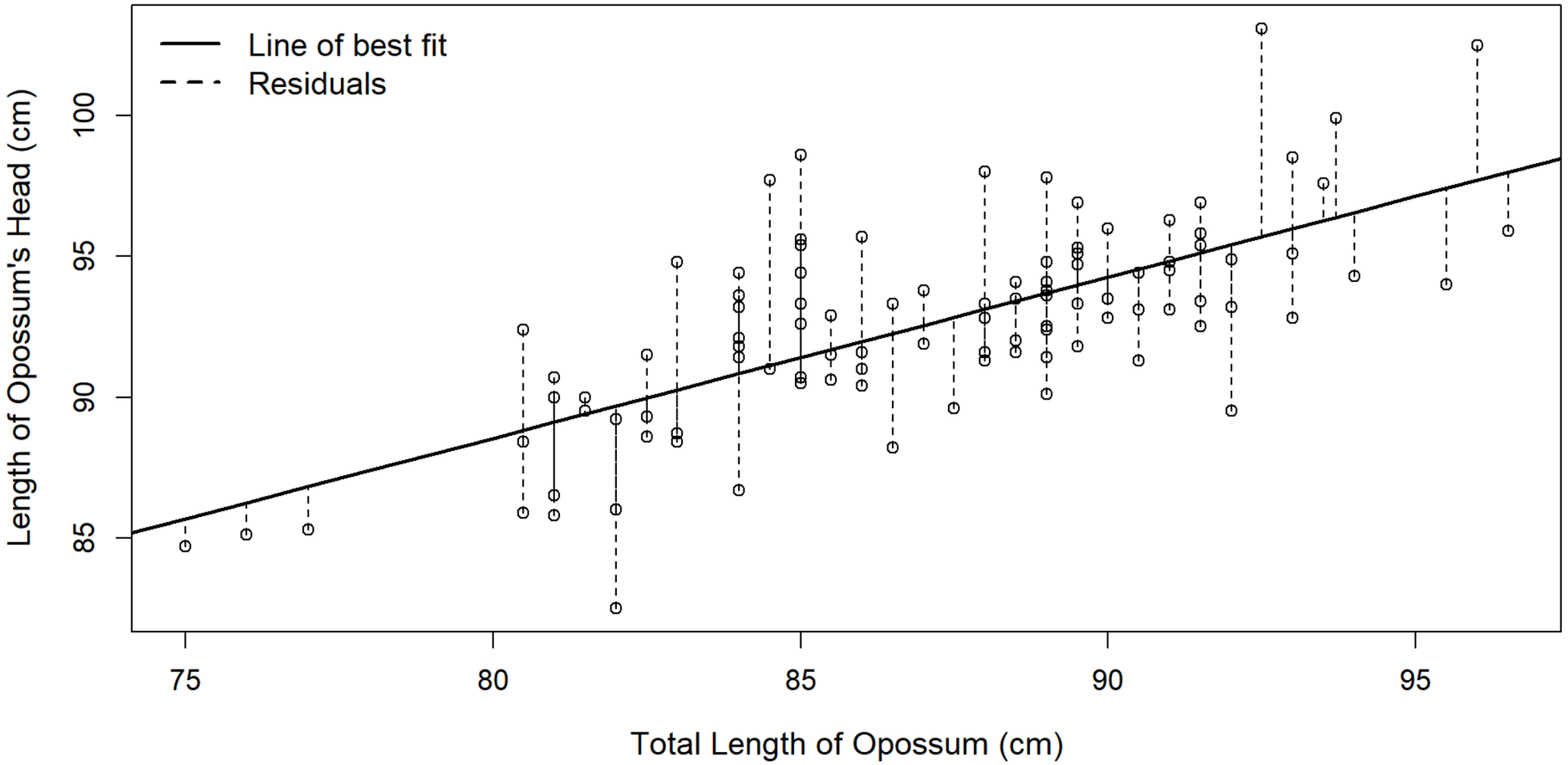
$$\epsilon_i = y_i - \hat{y}_i$$

Residuals

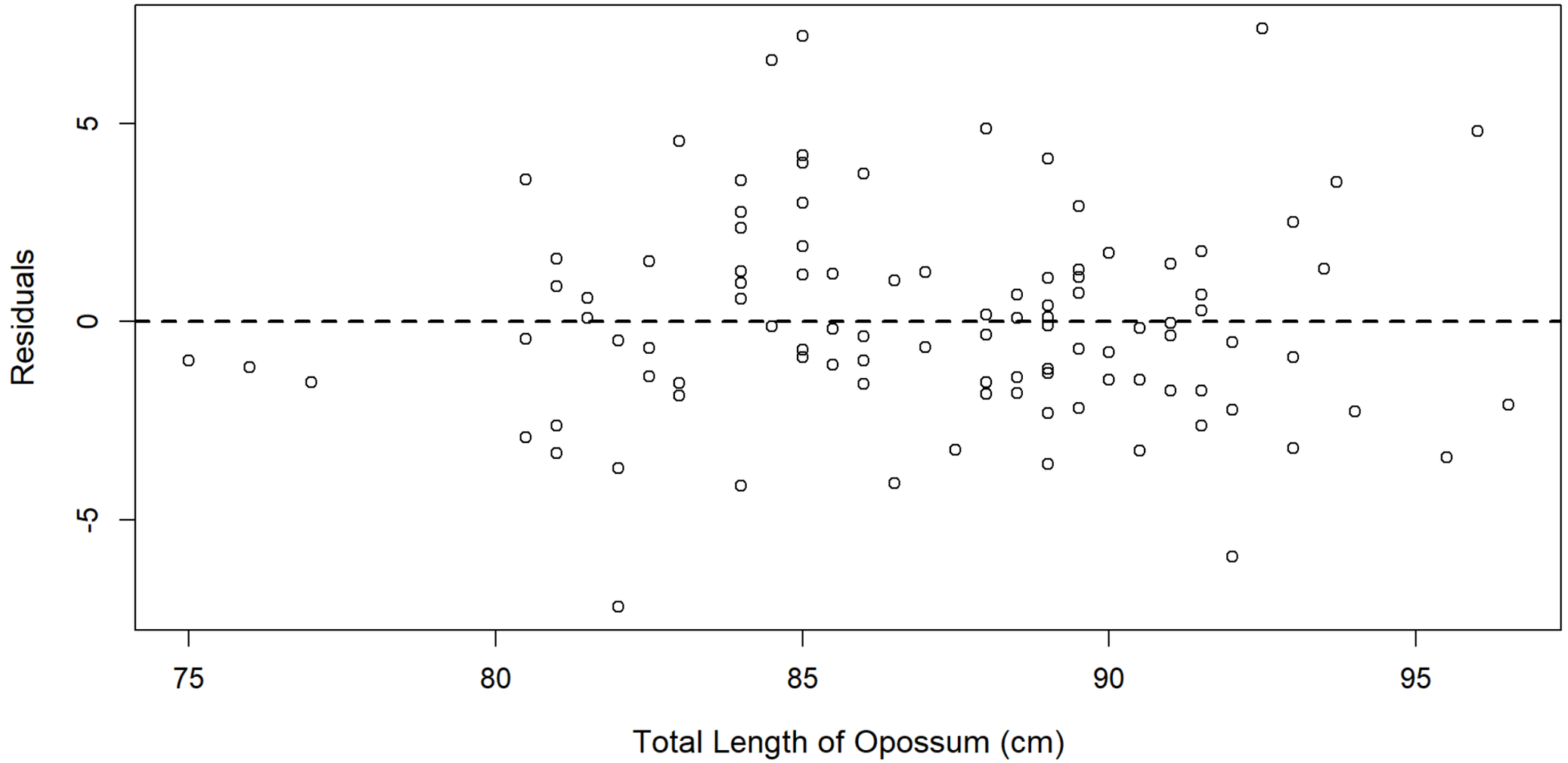
```r
p <- data.frame(
  x = x$total_l,
  y = x$head_l,
  y_hat = predict(lm1)
)
p$residual <- p$y - p$y_hat
head(p)
```

```
     x    y    y_hat   residual
1 89.0 94.1 93.69801  0.4019925
2 91.5 92.5 95.13026 -2.6302607
3 95.5 94.0 97.42187 -3.4218658
4 92.0 93.2 95.41671 -2.2167113
5 85.5 91.5 91.69285 -0.1928530
6 90.5 93.1 94.55736 -1.4573594
```

Residuals


Line of best fit
Residuals
Length of Opossum's Head (cm)
85
90
95
100
75
80
85
90
95
Total Length of Opossum (cm)

Residuals



Residuals

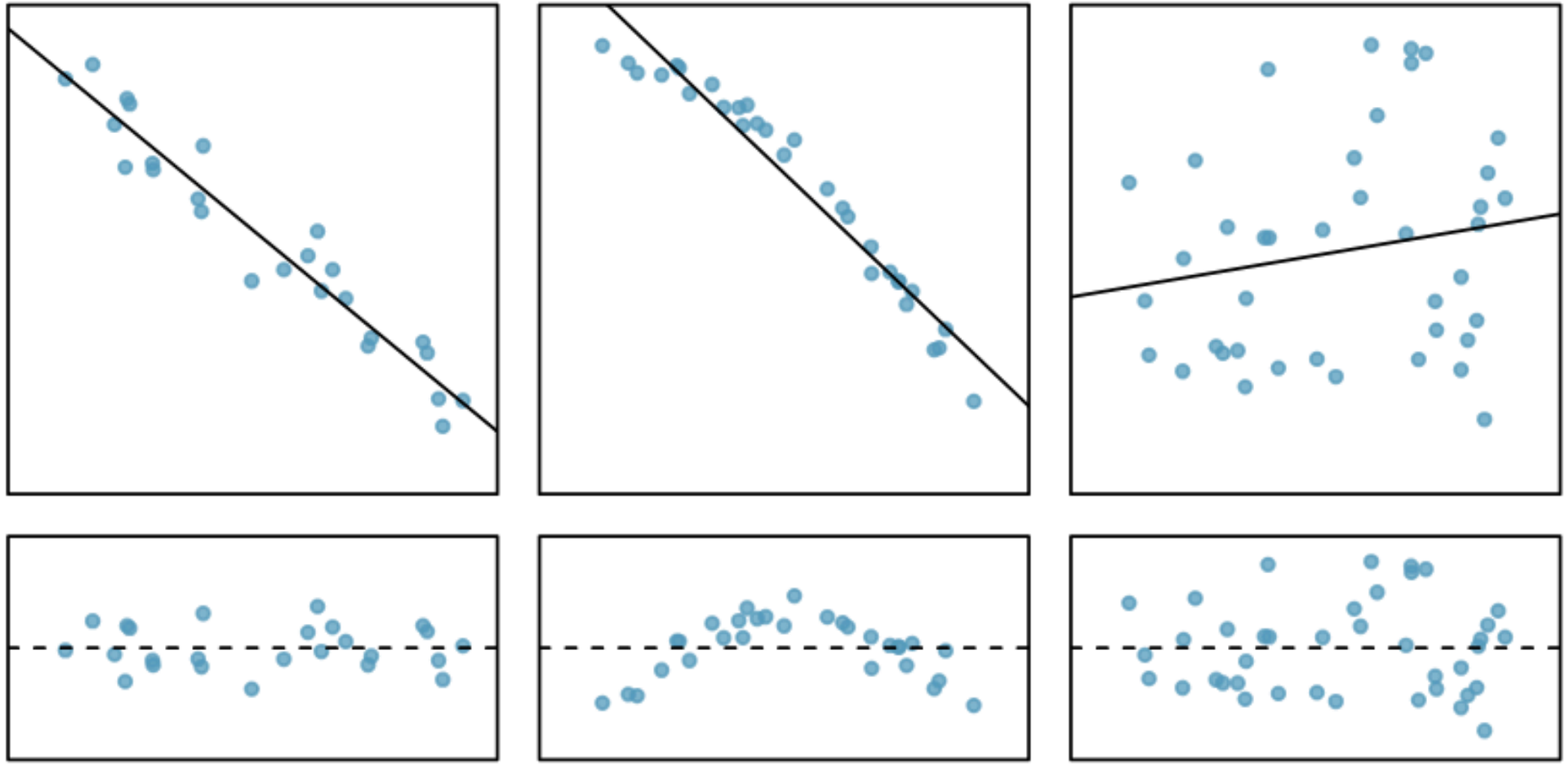


Figure 8.8: Sample data with their best fitting lines (top row) and their corresponding residual plots (bottom row).

Correlation

$$R = \frac{1}{n-1} \sum_{i=1}^{n} \frac{x_i - \bar{x}}{s_x} \frac{y_i - \bar{y}}{s_y}$$

- Correlation describes the strength of the linear relationship between two variables
  - We denote the correlation by
  - Always takes values between -1 and 1

Correlation

```
zx <- (x$total_l - mean(x$total_l)) / sd(x$total_l)
zy <- (x$head_l - mean(x$head_l)) / sd(x$head_l)
sum(zx * zy)/ (nrow(x) - 1)
```

```
[1] 0.6910937
```

```
cor(x$head_l, x$total_l)
```

```
[1] 0.6910937
```

Correlation

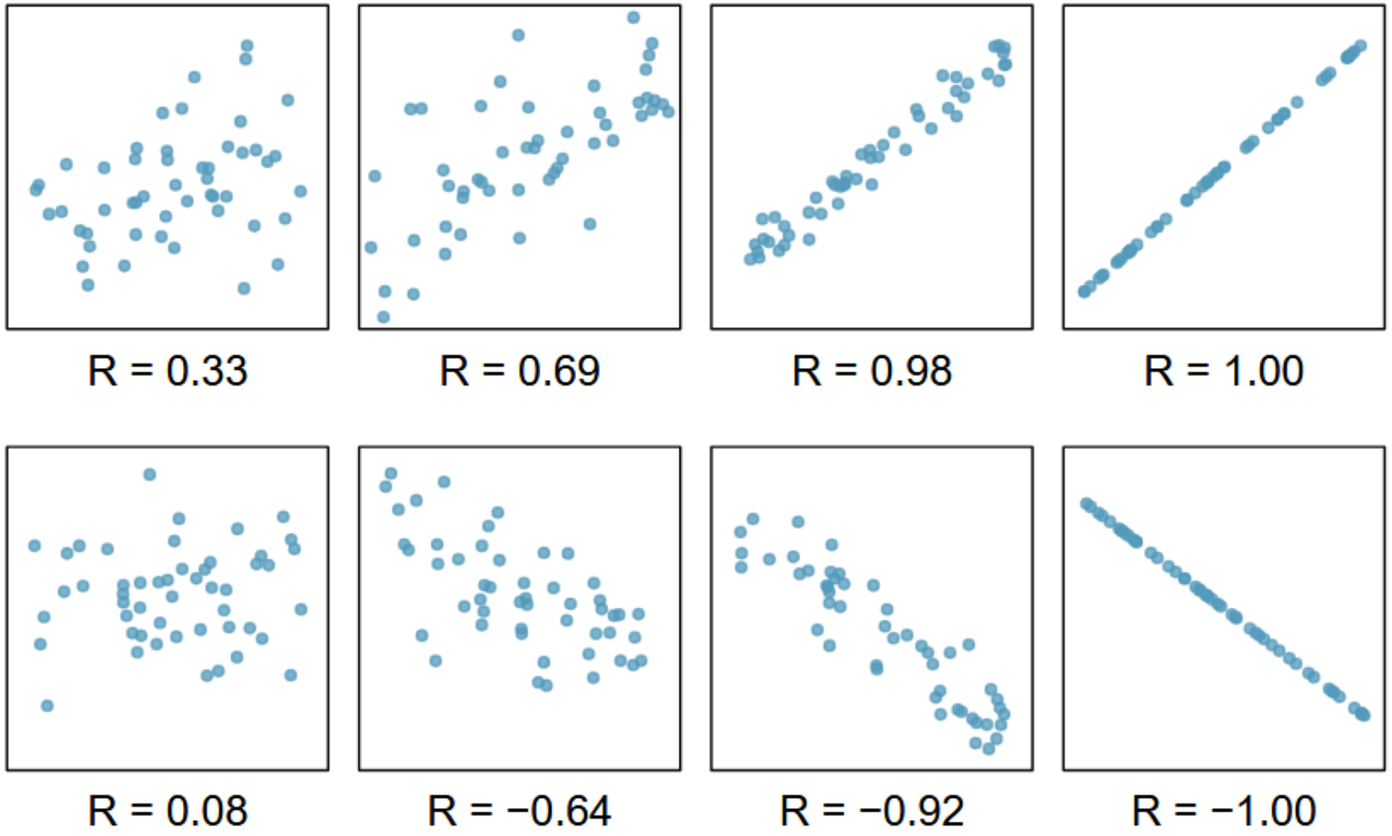


Figure 8.9: Sample scatterplots and their correlations. The first row shows variables with a positive relationship, represented by the trend up and to the right. The second row shows one plot with an approximately neutral trend and three plots with a negative trend.

Correlation

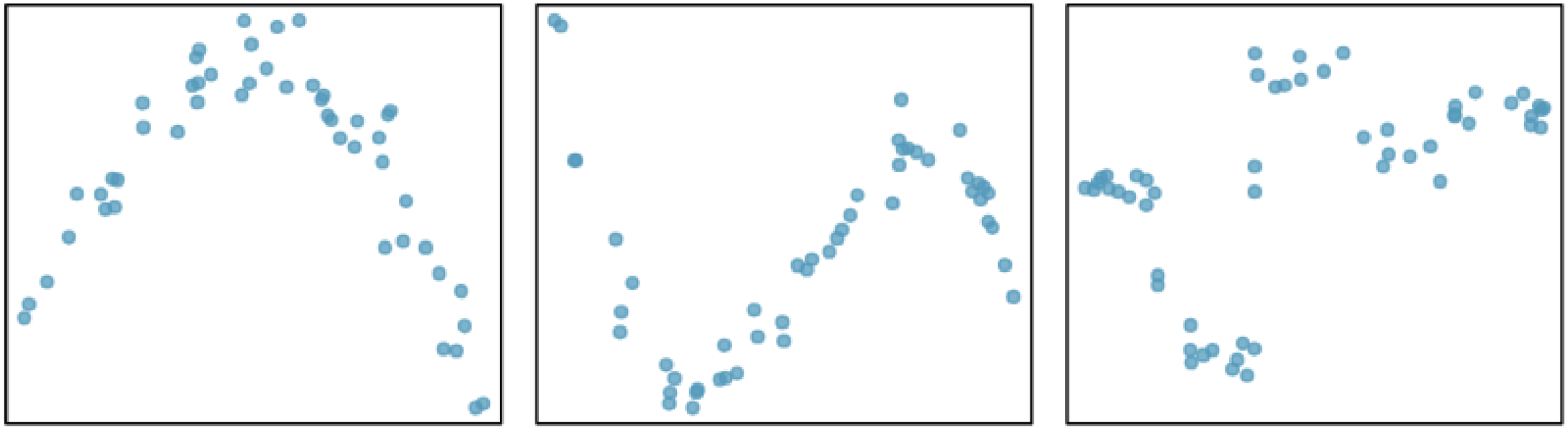


Figure 8.10: Sample scatterplots and their correlations. In each case, there is a strong relationship between the variables. However, because the relationship is nonlinear, the correlation is relatively weak.

Least squares regression

"Line of best fit"

- We begin by thinking about what we mean by "best"
  - Mathematically, we want a line that minimizes the magnitude of residuals
  - Most commonly, this is done by **minimizing the sum of the squared residuals**

Least squares regression

Conditions for the least squares line

- **Linearity**: The data should show a linear trend
- **Normal residuals**: Generally, the residuals must be nearly normal. When this condition is found to be unreasonable, it is usually because of outliers or concerns about influential points
- **Constant variability**: The variability of points around the least squares line remains roughly constant
- **Independent observations**: Be cautious about applying regression to time series data, which are sequential observations in time such as a stock price each day

Least squares regression

Conditions for the least squares line

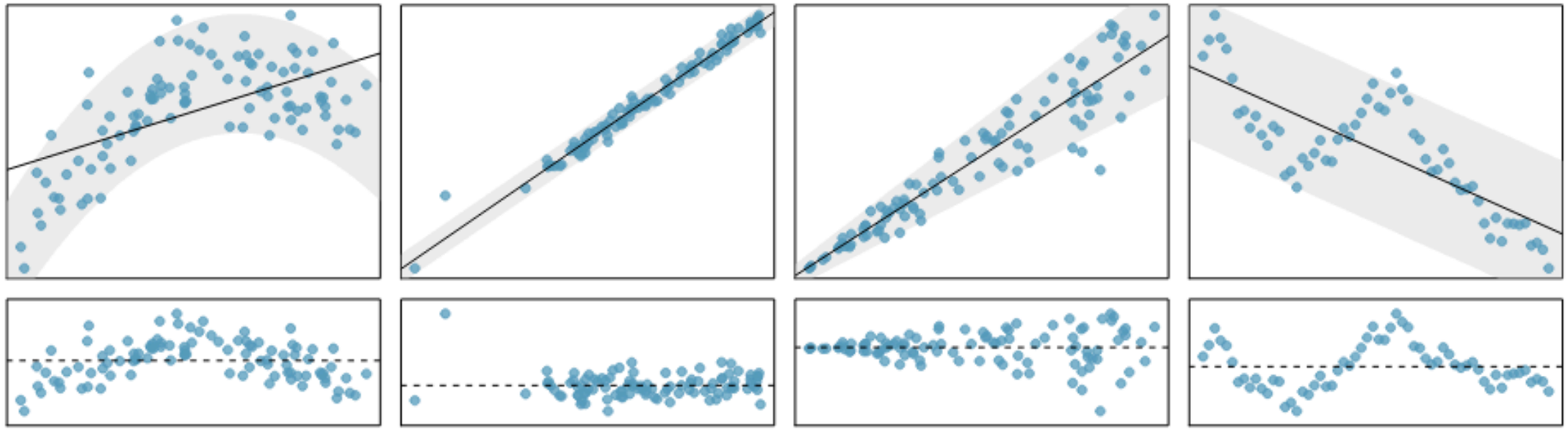


Figure 8.12: Four examples showing when the methods in this chapter are insufficient to apply to the data. First panel: linearity fails. Second panel: there are outliers, most especially one point that is very far away from the line. Third panel: the variability of the errors is related to the value of x. Fourth panel: a time series data set is shown, where successive observations are highly correlated.

Least squares regression

New example: Elmhurst College

```
x <- read.csv("elmhurst.csv")
head(x)
```

```
  family_income gift_aid price_paid
1        92.922    21.72      14.28
2         0.250    27.47       8.53
3        53.092    27.75      14.25
4        50.200    27.22       8.78
5       137.613    18.00      24.00
6        47.957    18.52      23.48
```

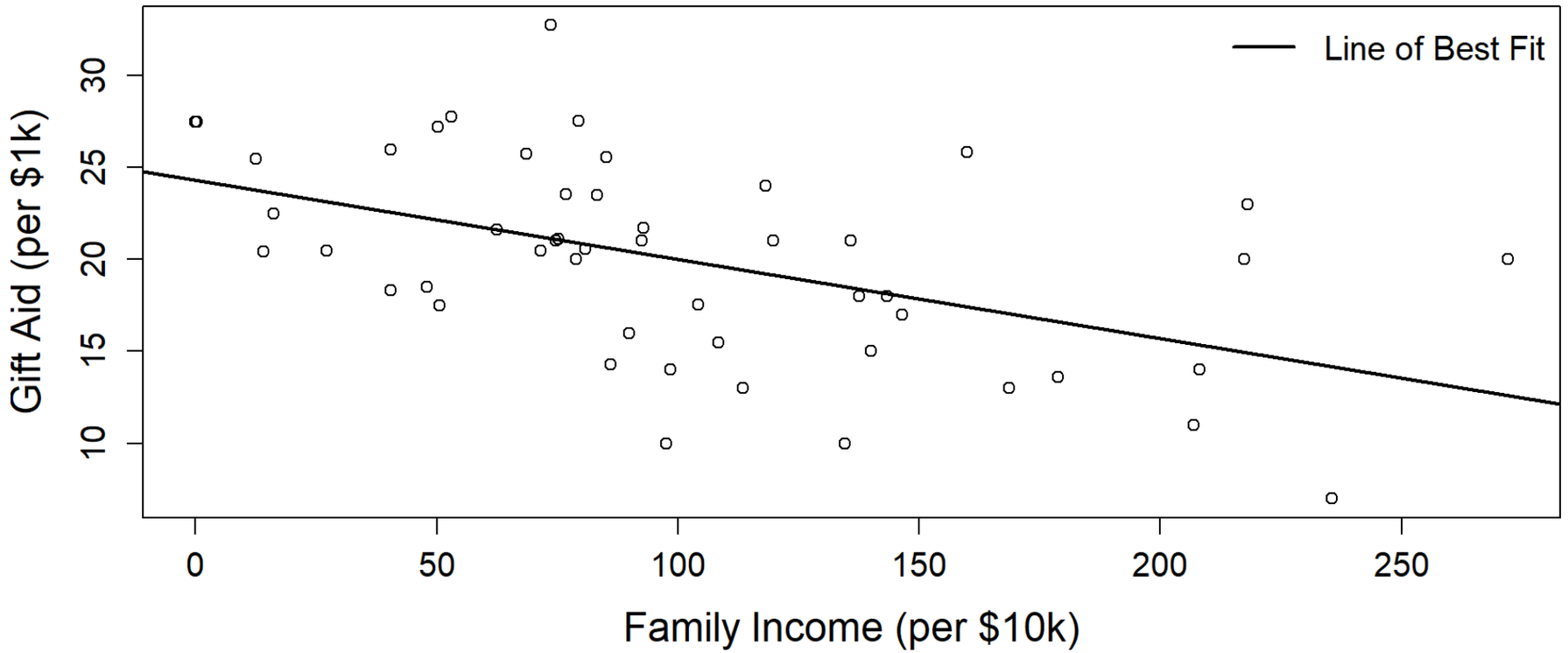
```
lm2 <- lm(gift_aid ~ family_income, x)
coefficients(lm2)
```

```
  (Intercept) family_income
  24.31932901   -0.04307165
```

```
b0 <- coefficients(lm2)[1]
b1 <- coefficients(lm2)[2]
```

Least squares regression

Scatterplot



Least squares regression

Residuals

```r
y <- data.frame(
  x = x$family_income,
  y = x$gift_aid,
  y_hat = predict(lm2)
)
y$residuals <- y$y - y$y_hat
head(y)
```

```
        x     y    y_hat  residuals
1  92.922 21.72 20.31702  1.4029751
2   0.250 27.47 24.30856  3.1614389
3  53.092 27.75 22.03257  5.7174311
4  50.200 27.22 22.15713  5.0628679
5 137.613 18.00 18.39211 -0.3921097
6  47.957 18.52 22.25374 -3.7337418
```

```r
head(y$residuals)
```

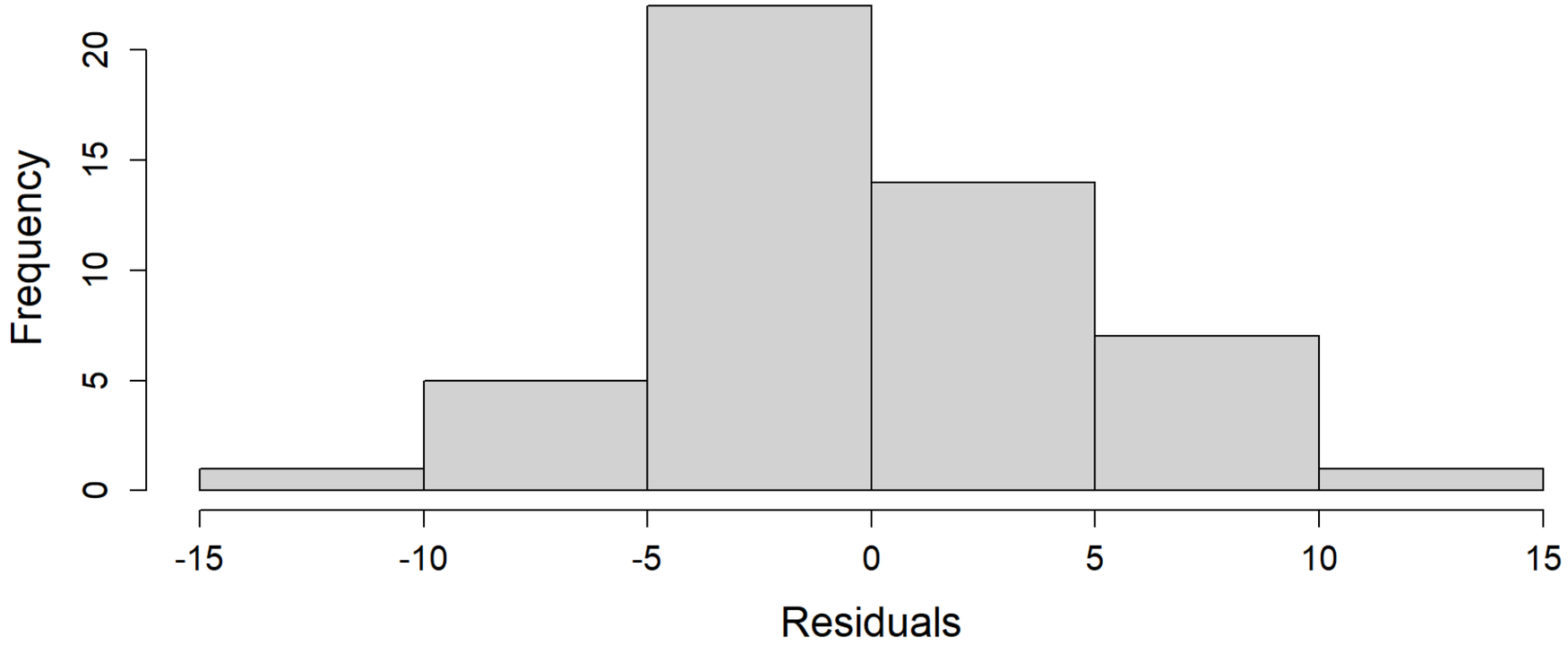
```
[1]  1.4029751  3.1614389  5.7174311  5.0628679 -0.3921097 -3.7337418
```

```r
head(residuals(lm2))
```

```
         1          2          3          4          5          6
 1.4029751  3.1614389  5.7174311  5.0628679 -0.3921097 -3.7337418
```

Least squares regression

Residuals



Least squares regression: Slope

```
mean_y <- round(mean(x$gift_aid)*1000)
sd_y <- round(sd(x$gift_aid)*1000)
mean_x <- round(mean(x$family_income)*1000, 0)
sd_x <- round(sd(x$family_income)*1000, 0)
r <- round(cor(x$gift_aid, x$family_income), 3)
r
```

```
[1] -0.499
```

```
sd_y / sd_x * r
```

```
[1] -0.04311361
```

```
coefficients(lm2)[2]; b1
```

```
family_income
  -0.04307165
```

```
family_income
  -0.04307165
```

Least squares regression

Intercept

- You might recall the point-slope form of a line from math class, which we can use to find the model fit, including the estimate of

- To find the y-intercept, set

Least squares regression

Intercept

```
1  (mean_y - b1*mean_x)/1000
```

```
family_income
     24.31979
```

```
1  coefficients(lm2)[1]; b0
```

```
(Intercept)
   24.31933
```

```
(Intercept)
   24.31933
```

Least squares regression

Interpretation: Slope

- What do these regression coefficients mean?
  - The slope describes the estimated difference in the variable if the explanatory variable was **one unit** larger.
  - For each additional $1,000 of family income, we would expect a student to receive a net difference of $1,000×(−0.0431) = −$43.10 in aid on average, i.e. $43.10 less.

Least squares regression

Interpretation: Intercept

- What do these regression coefficients mean?
  - The intercept describes the average outcome of if and the linear model is valid all the way to , which in many applications is not the case.
  - The estimated intercept describes the average aid if a student's family had no income.
- We must be cautious in this interpretation: while there is a real association, we cannot interpret a **causal** connection between the variables.

Least squares regression

Extrapolation

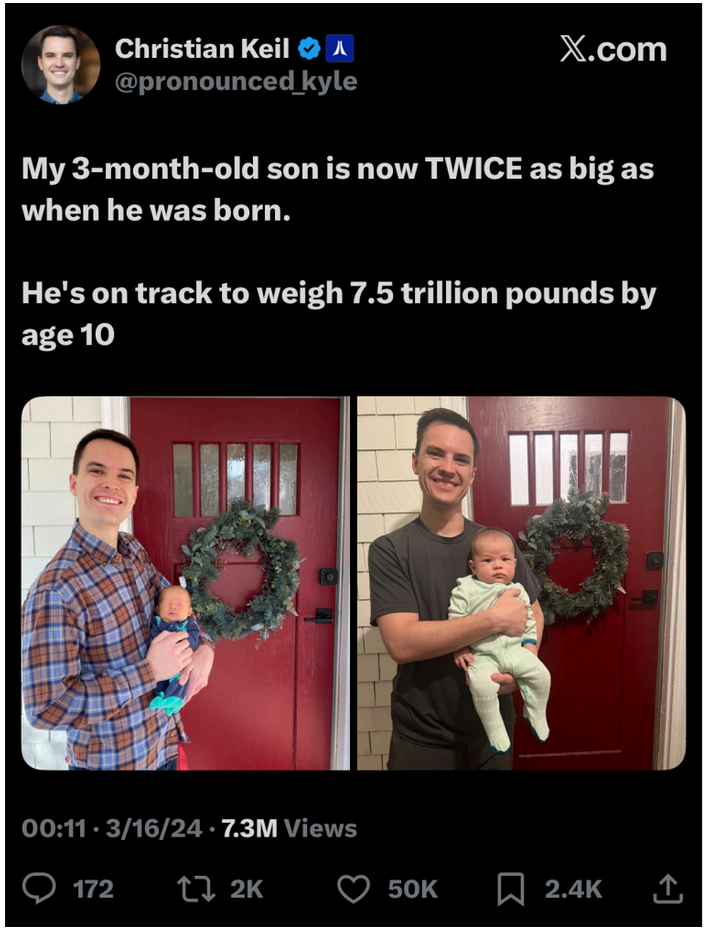
- Applying a model estimate to values outside of the realm of the original data is called **extrapolation**.
  - If we extrapolate, we are making an unreliable bet that the approximate linear relationship will be valid in places where it has not been analyzed.

```
1 as.numeric(b0 + b1*1000)*1000
```

```
[1] -18752.32
```

Least squares regression

Extrapolation



Least squares regression

Strength of a fit: R-squared

- We evaluated the strength of the linear relationship between two variables earlier using the correlation, .
  - However, it is more common to explain the strength of a linear fit using
  - If provided with a linear model, we might like to describe how closely the data cluster around the linear fit.

Least squares regression

Strength of a fit: R-squared

```
1 sy2 <- sd(y$y)^2
2 se2 <- sd(y$residuals)^2
3
4 (sy2 - se2) / sy2
```

```
[1] 0.2485582
```

```
1 summary(lm2)$r.squared
```

```
[1] 0.2485582
```

- About 25% of the variation in gift aid can be explained by differences in family income.

Least squares regression

Example 3: Categorical Predictors

```r
x <- read.csv("mariokart.csv")
y <- data.frame(
  new = ifelse(x$cond == "new", 1, 0),
  price = x$total_pr
)
summary(y)
```

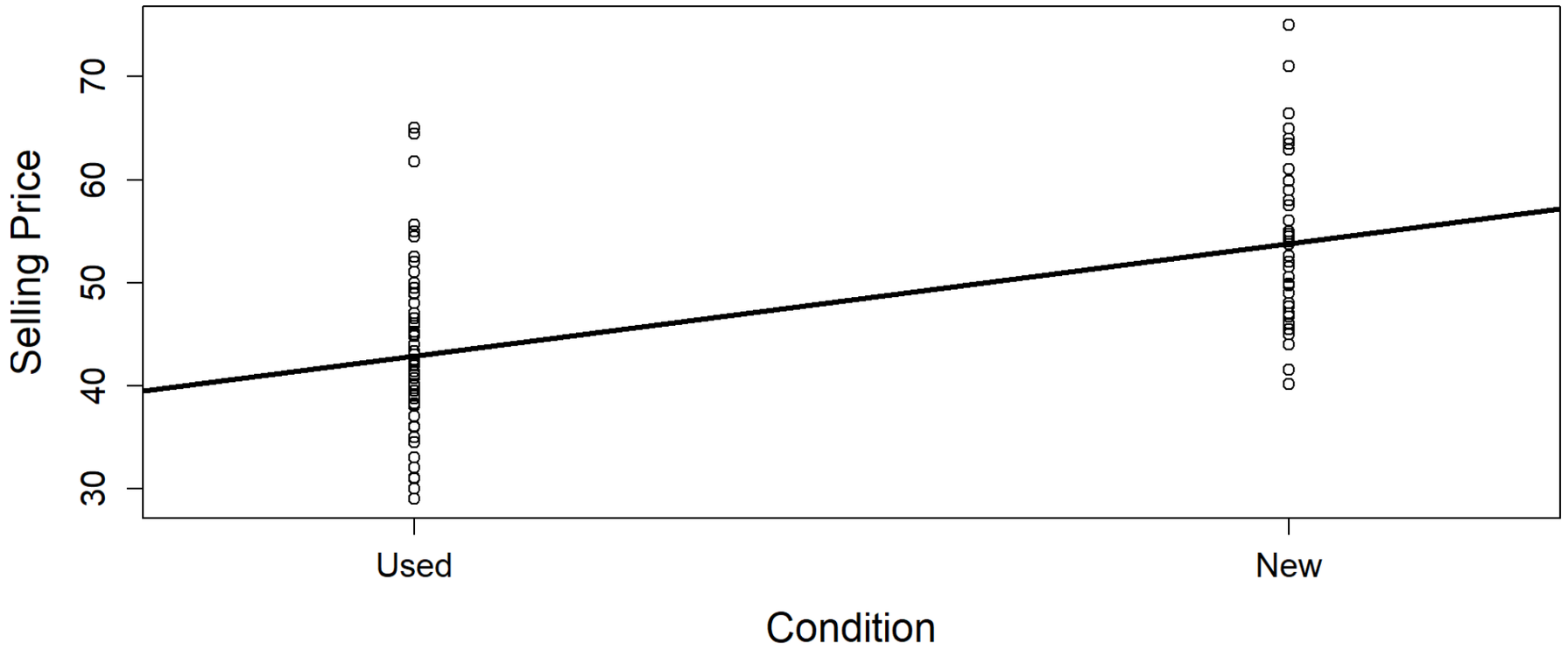
```
      new              price
 Min.   :0.0000   Min.   : 28.98
 1st Qu.:0.0000   1st Qu.: 41.17
 Median :0.0000   Median : 46.50
 Mean   :0.4126   Mean   : 49.88
 3rd Qu.:1.0000   3rd Qu.: 53.99
 Max.   :1.0000   Max.   :326.51
```

```r
z <- (y$price - mean(y$price)) / sd(y$price)
y <- y[z <= 2.576,]

lm3 <- lm(price ~ new, y)
coefficients(lm3)
```

```
(Intercept)         new
   42.87110    10.89958
```

Least squares regression

Example 3: Categorical Predictors

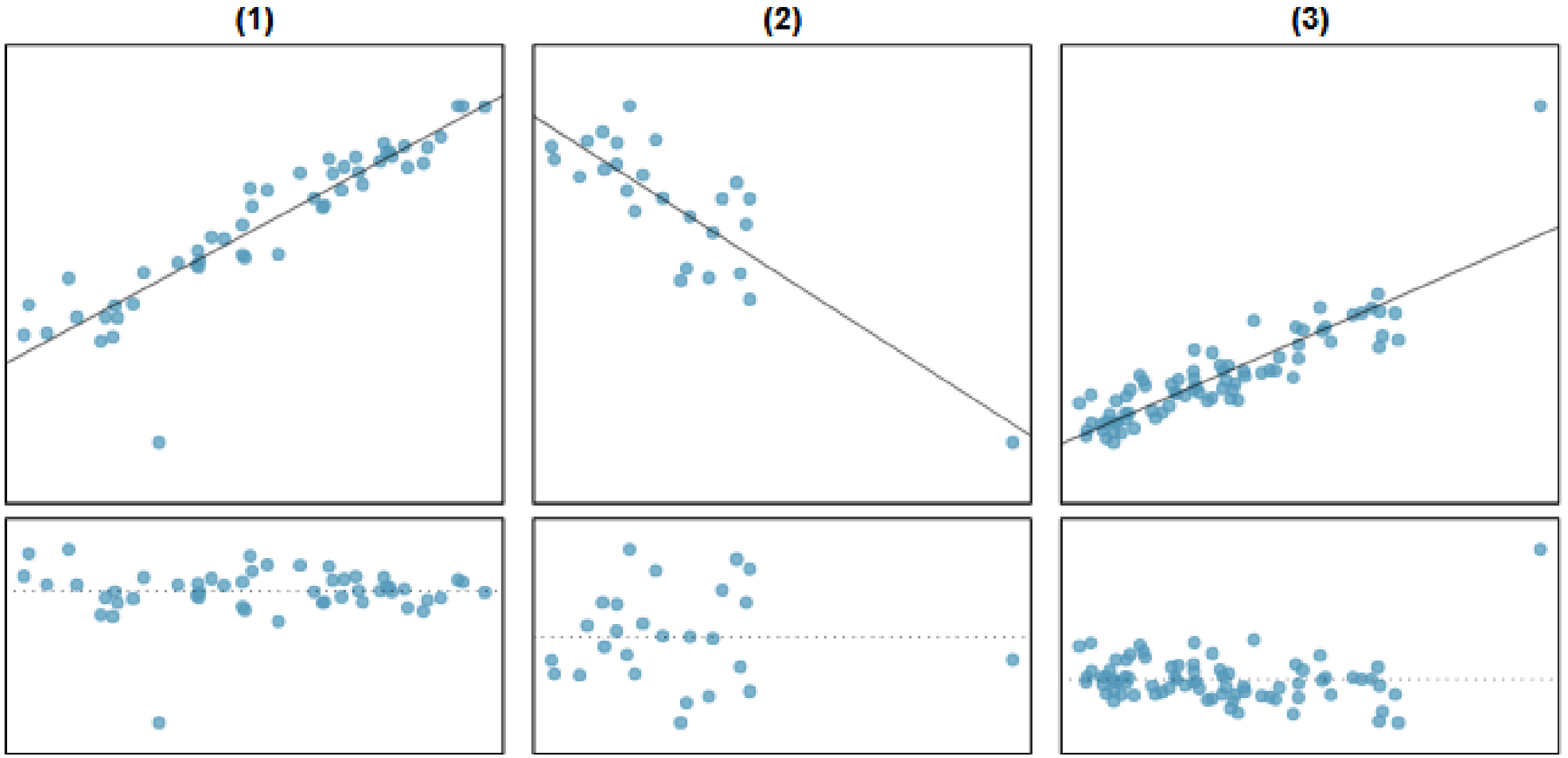


Least squares regression

Example 3: Categorical Predictors

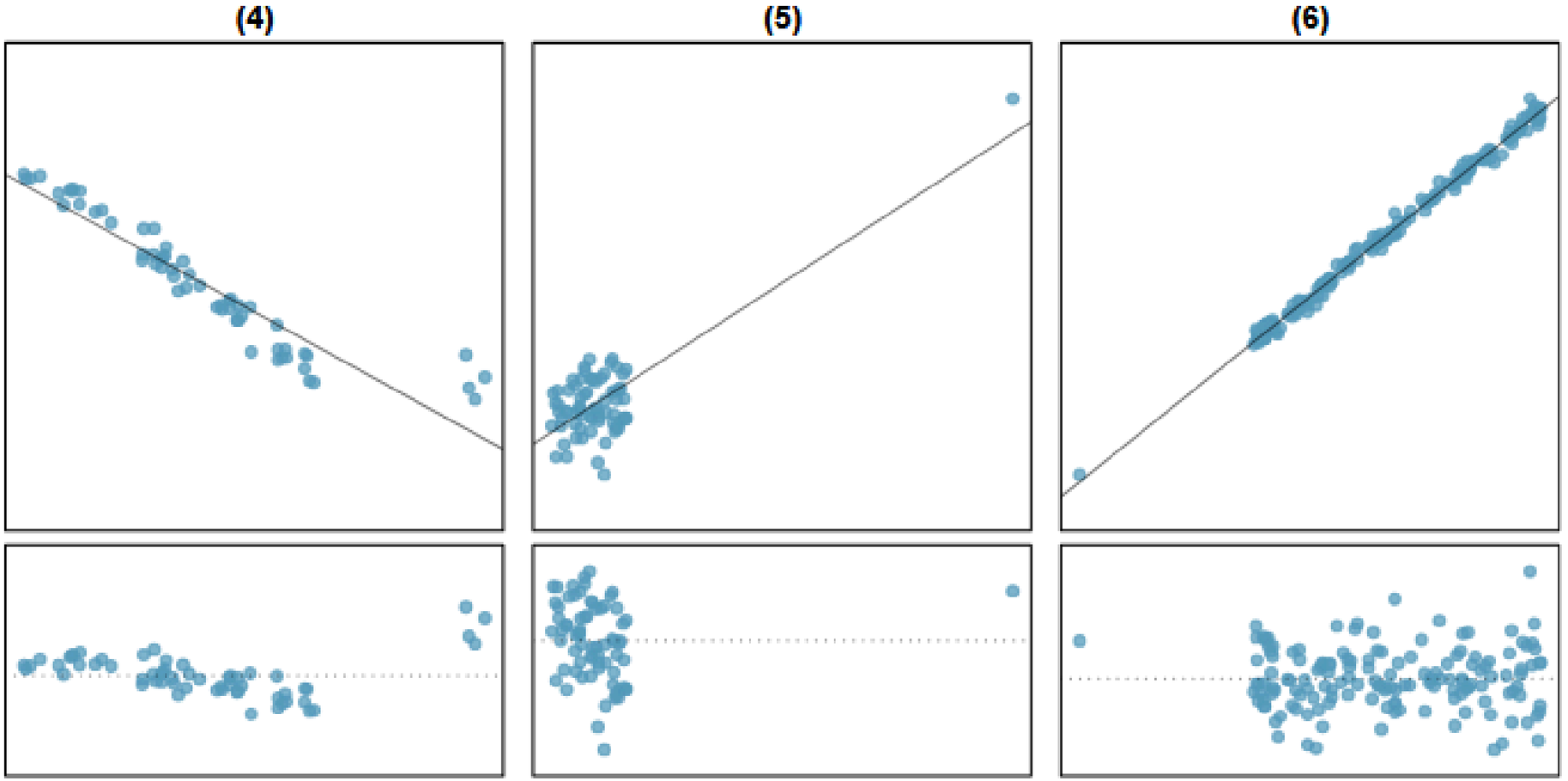
Least squares regression

Highly senstive to outliers



Least squares regression

Highly senstive to outliers



Inference for linear regression

Example: Midterm elections & unemployment

- **Context:** U.S. House elections occur every two years; those held **midway through a presidential term** are called **midterms**.
- **Theory:** When **unemployment is high**, the **President's party performs worse** in midterms.
- **Approach:** Use **historical data (1898–2018)**—excluding **Great Depression** years—to test whether **unemployment predicts midterm losses**.

Inference for linear regression

Example: Midterm elections & unemployment

Inference for linear regression

Example: Midterm elections & unemployment

```
x <- read.csv("midterms_house.csv")
head(x)
```

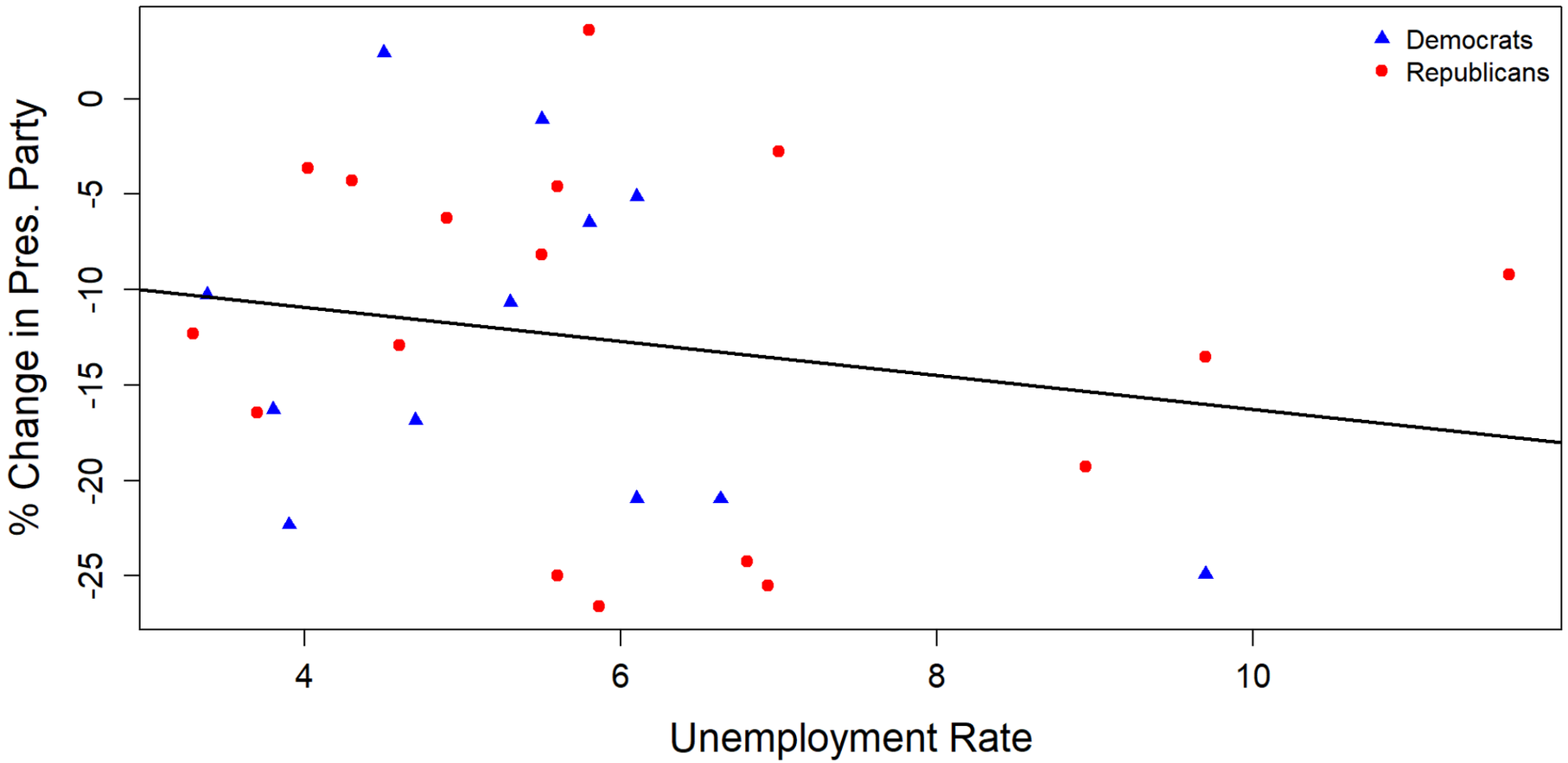
```
  year               potus      party unemp house_change
1 1899    William McKinley Republican 11.62    -9.223301
2 1903  Theodore Roosevelt Republican  4.30    -4.275907
3 1907  Theodore Roosevelt Republican  3.29   -12.291499
4 1911 William Howard Taft Republican  5.86   -26.590640
5 1915      Woodrow Wilson   Democrat  6.63   -20.962199
6 1919      Woodrow Wilson   Democrat  3.38   -10.280374
```

```
z <- (x$unemp - mean(x$unemp)) / sd(x$unemp)
x <- x[z <= 2.576,]

lm4 <- lm(house_change ~ unemp, x)
coefficients(lm4)
```

```
(Intercept)       unemp
 -7.3644063  -0.8897261
```

Inference for linear regression

Example: Midterm elections & unemployment



Inference for linear regression

Example: Midterm elections & unemployment

- We might wonder, is this convincing evidence that the “true” linear model has a negative slope?
  - That is, do the data provide strong evidence that the political theory is accurate, where the unemployment rate is a useful predictor of the midterm election?
-
-

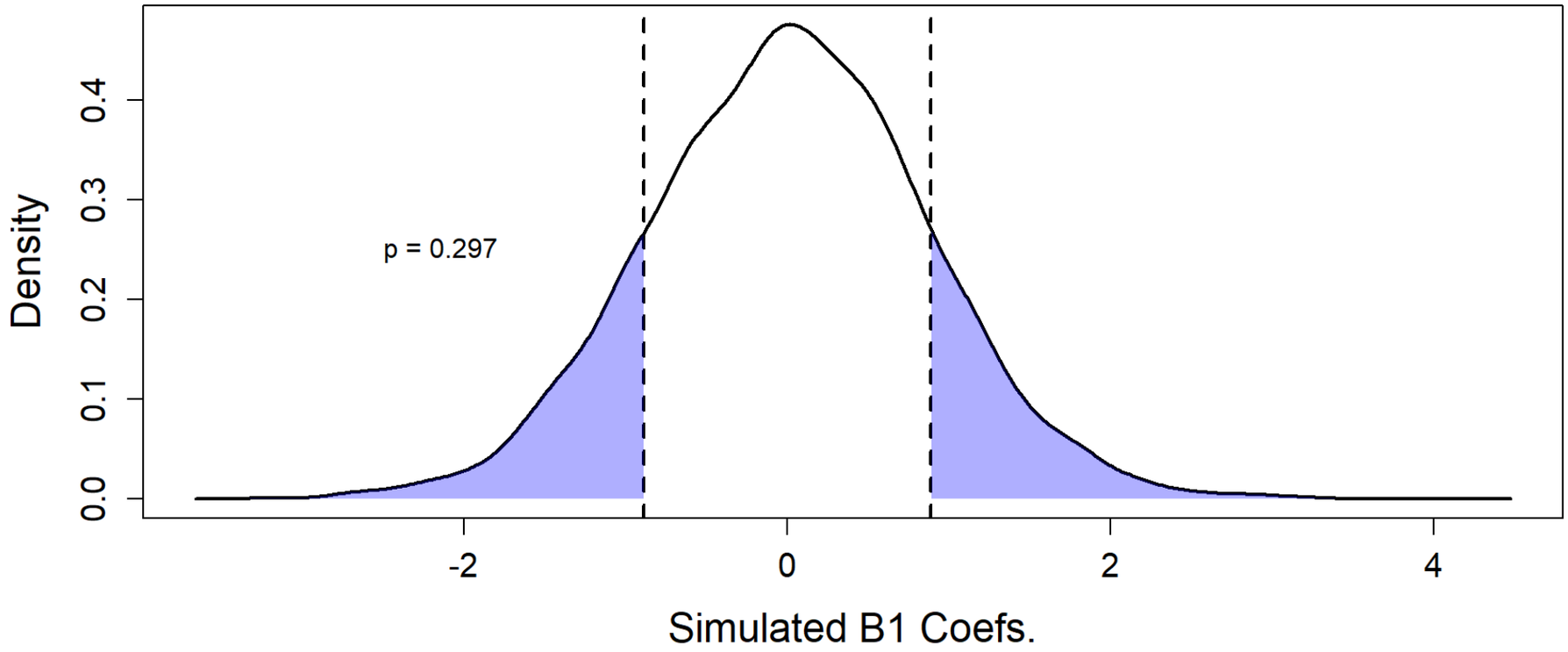
Inference for linear regression

Example: Midterm elections & unemployment

```r
sd_x <- sd(x$unemp)
sd_y <- sd(x$house_change)
n <- nrow(x)
z <- list()
for(i in 1:10000){
  set.seed(i)
  y <- data.frame(
    x = rnorm(n, mean = 0, sd = sd_x),
    y = rnorm(n, mean = 0, sd = sd_y)
  )
  z[[length(z)+1]] <- coefficients(lm(y ~ x, y))[2]
}
z <- unlist(z)
dz <- density(z)
```

Inference for linear regression

Example: Midterm elections & unemployment



Inference for linear regression

Example: Midterm elections & unemployment

```r
b1 <- as.numeric(coefficients(lm4)[2])

mean(ifelse(b1 >= z, 1, 0))*2
```

```
[1] 0.2972
```

```r
summary(lm4)
```

```
Call:
lm(formula = house_change ~ unemp, data = x)

Residuals:
     Min       1Q   Median       3Q      Max
-14.0124  -7.6989   0.0913   7.2974  16.1447

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  -7.3644     5.1553  -1.429    0.165
unemp        -0.8897     0.8350  -1.066    0.296

Residual standard error: 8.913 on 27 degrees of freedom
Multiple R-squared:  0.04035,   Adjusted R-squared:  0.004812
```

Inference for linear regression

- We usually rely on statistical software to identify point estimates, standard errors, test statistics, and p-values in practice.
- However, be aware that software will not generally check whether the method is appropriate, meaning we must still verify conditions are met.

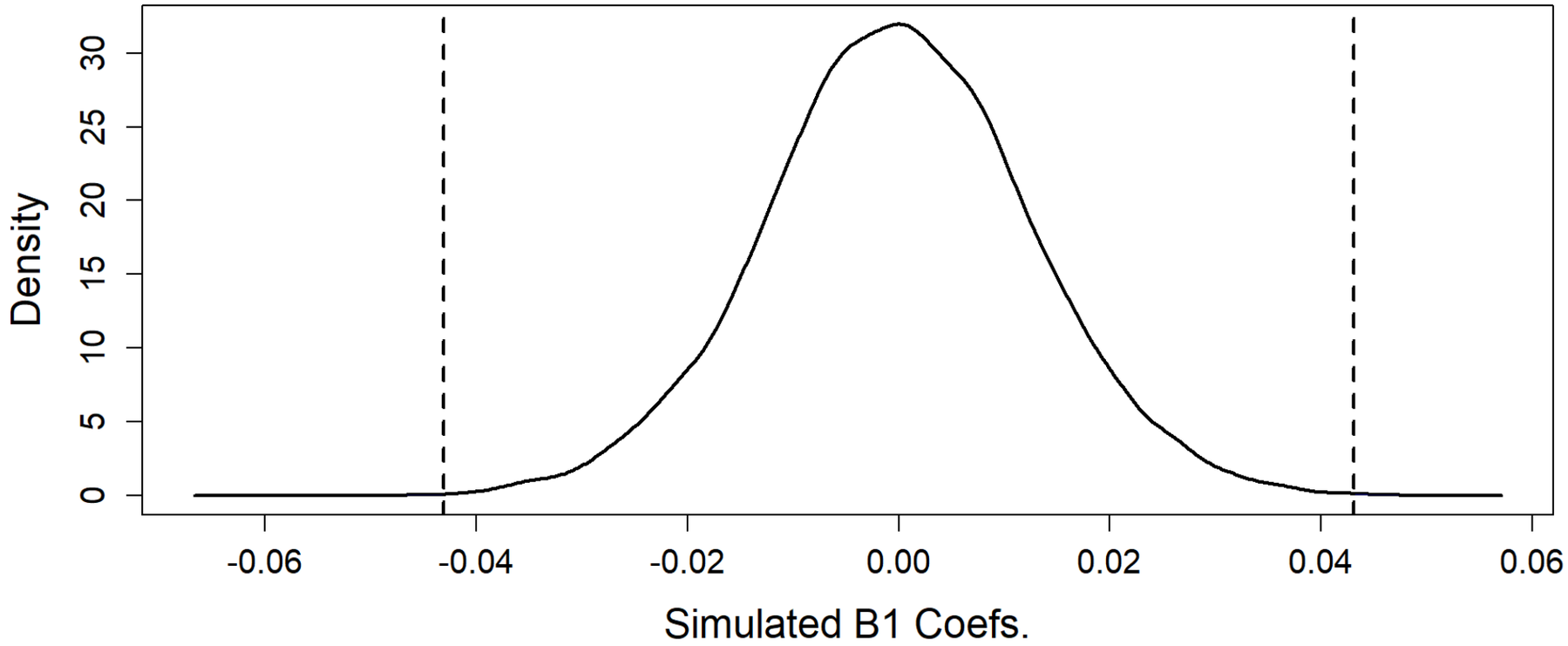
Inference for linear regression

Elmhurst College example

```
x <- read.csv("elmhurst.csv")
sd_x <- sd(x$family_income)
sd_y <- sd(x$gift_aid)
n <- nrow(x)
z <- list()
for(i in 1:10000){
  set.seed(i)
  y <- data.frame(
    x = rnorm(n, mean = 0, sd = sd_x),
    y = rnorm(n, mean = 0, sd = sd_y)
  )
  z[[length(z)+1]] <- coefficients(lm(y ~ x, y))[2]
}
z <- unlist(z)
dz <- density(z)
```

Inference for linear regression

Elmhurst College example



Inference for linear regression

Elmhurst College example

```r
lm2 <- lm(gift_aid ~ family_income, x)
b1 <- as.numeric(coefficients(lm2)[2])

mean(ifelse(b1 >= z, 1, 0))*2
```

```
[1] 0.001
```

```r
summary(lm2)
```

```
Call:
lm(formula = gift_aid ~ family_income, data = x)

Residuals:
     Min       1Q   Median       3Q      Max
-10.1128  -3.6234  -0.2161   3.1587  11.5707

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)   24.31933    1.29145  18.831  < 2e-16 ***
family_income -0.04307    0.01081  -3.985 0.000229 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Introduction to Multiple Regression

- **Multiple regression** extends simple two-variable regression to situations with one response variable and multiple predictors (denoted , , , …).
  - It is motivated by cases where several factors may simultaneously influence an outcome.
- A multiple regression model is a linear model with multiple predictors. In general, we write the model as… where there are predictors.

Introduction to Multiple Regression

Bankruptcy Example

- We will consider data on loans from the peer-to-peer lender, Lending Club.
- The dataset includes both **loan characteristics** and **borrower information**.
  - Our goal is to understand the factors that influence the **interest rate assigned to each loan**.
- Holding all other characteristics constant:
  - Does it matter **how much debt someone already has**?
  - Does it matter **whether their income has been verified**?

Multiple regression: Bankruptcy example

Data

```r
x <- read.csv("loans_full_schema.csv")
dim(x)
```

```
[1] 10000    55
```

```r
y <- data.frame(
  interest_rate = x$interest_rate,
  income_ver = x$verified_income,
  debt_to_income = x$debt_to_income,
  credit_util = x$total_credit_utilized / x$total_credit_limit,
  bankruptcy = ifelse(x$public_record_bankrupt > 0, 1, 0),
  term = x$term,
  issued = x$issue_month,
  credit_checks = x$inquiries_last_12m
)
head(y)
```

```
  interest_rate      income_ver debt_to_income credit_util bankruptcy term
1         14.07        Verified          18.01  0.54759517           0   60
2         12.61    Not Verified           5.04  0.15003472           1   36
3         17.09 Source Verified          21.15  0.66134832           0   36
4          6.72    Not Verified          10.16  0.19673228           0   36
5         14.07        Verified          57.96  0.75490772           0   36
```

Multiple regression: Bankruptcy example

Data descriptions

variable	description
`interest_rate`	Interest rate for the loan.
`income_ver`	Categorical variable describing whether the borrower's income source and amount have been verified, with levels `verified`, `source_only`, and `not`.
`debt_to_income`	Debt-to-income ratio, which is the percentage of total debt of the borrower divided by their total income.
`credit_util`	Of all the credit available to the borrower, what fraction are they utilizing. For example, the credit utilization on a credit card would be the card's balance divided by the card's credit limit.
`bankruptcy`	An indicator variable for whether the borrower has a past bankruptcy in her record. This variable takes a value of `1` if the answer is "yes" and `0` if the answer is "no".
`term`	The length of the loan, in months.
`issued`	The month and year the loan was issued, which for these loans is always during the first quarter of 2018.
`credit_checks`	Number of credit checks in the last 12 months. For example, when filing an application for a credit card, it is common for the company receiving the application to run a credit check.

Figure 9.2: Variables and their descriptions for the `loans` data set.

Multiple regression: Bankruptcy example

Regression analysis

```r
lm1 <- lm(interest_rate ~ bankruptcy, data = y)
summary(lm1)
```

```
Call:
lm(formula = interest_rate ~ bankruptcy, data = y)

Residuals:
    Min      1Q  Median      3Q     Max
-7.7648 -3.6448 -0.4548  2.7120 18.6020

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  12.3380     0.0533 231.490  < 2e-16 ***
bankruptcy    0.7368     0.1529   4.819 1.47e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 4.006 on 9998 degrees of freedom
```

Multiple regression: Bankruptcy example

Regression analysis

```
z <- aggregate(interest_rate ~ bankruptcy, y,
               function(x) c(mean = mean(x),
                             sd = sd(x),
                             n = length(x)))
z <- as.data.frame(do.call(cbind, z))
z$min <- z$mean - z$sd / sqrt(z$n)
z$max <- z$mean + z$sd / sqrt(z$n)
z
```

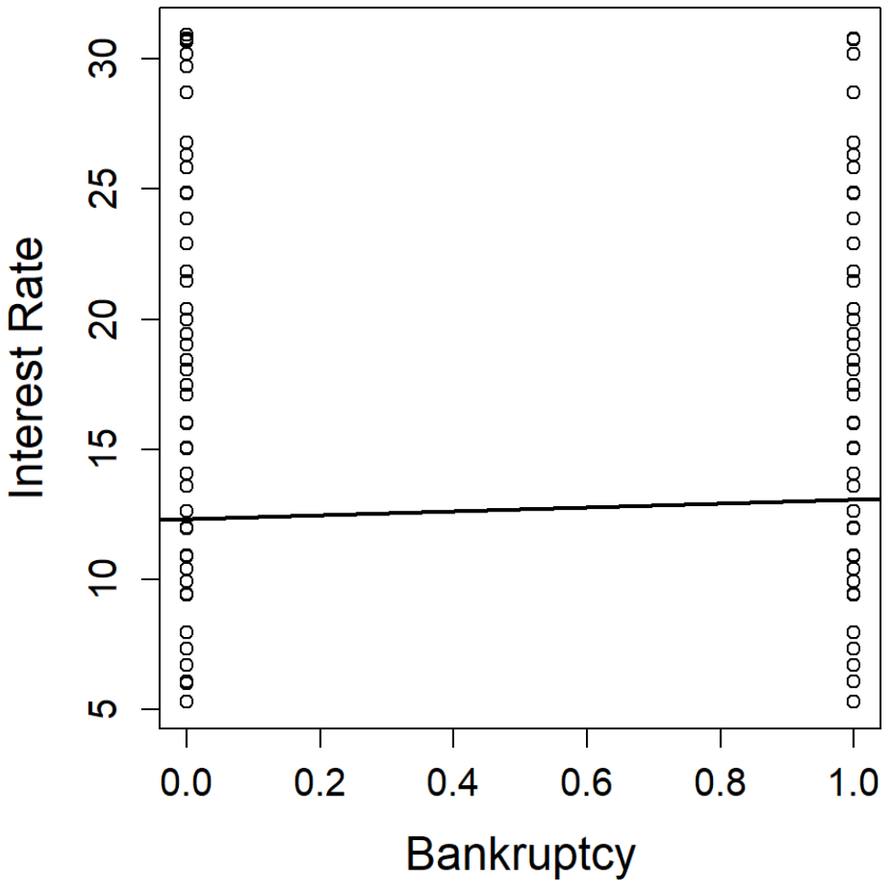
```
  bankruptcy     mean       sd    n      min      max
1          0 12.33800 5.018019 8785 12.28447 12.39154
2          1 13.07479 4.829929 1215 12.93623 13.21335
```

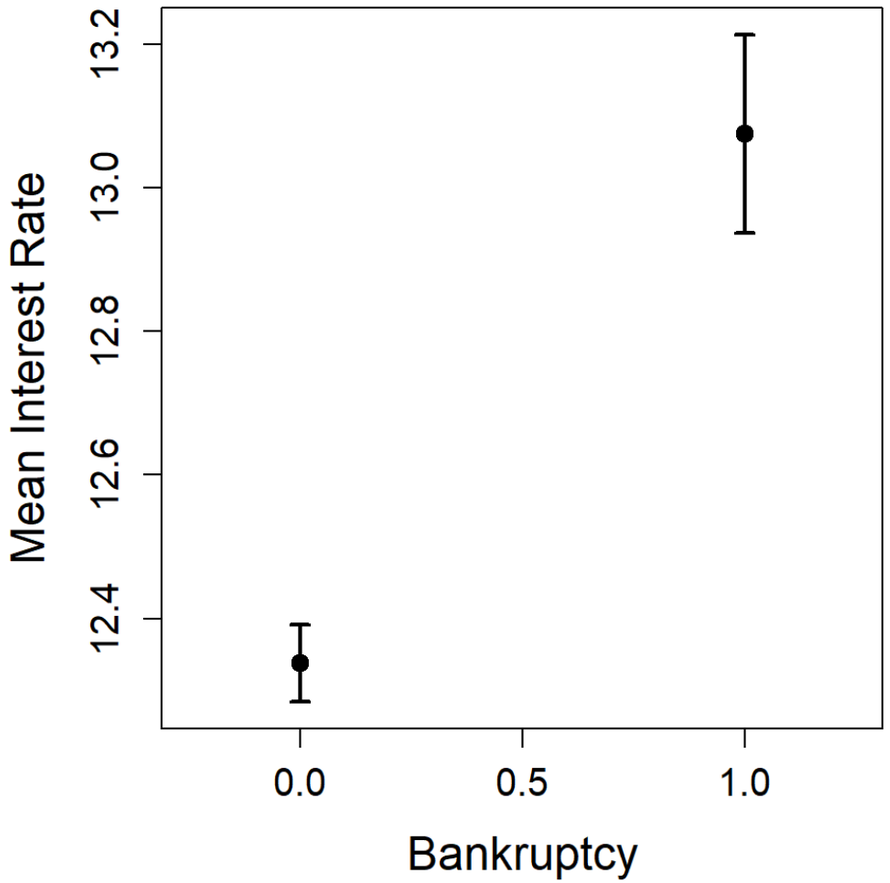
```
z[2,2] - z[1,2]
```

```
[1] 0.7367856
```

Multiple regression: Bankruptcy example

Plotting the regression





Multiple regression: Bankruptcy example

Regression analysis – Reference group

```r
lm2 <- lm(interest_rate ~ income_ver, data = y)
summary(lm2)
```

```
Call:
lm(formula = interest_rate ~ income_ver, data = y)

Residuals:
    Min      1Q  Median      3Q     Max
-9.0437 -3.7495 -0.6795  2.5345 19.6905

Coefficients:
                          Estimate Std. Error t value Pr(>|t|)
(Intercept)                11.09946    0.08091  137.18   <2e-16 ***
income_verSource Verified   1.41602    0.11074   12.79   <2e-16 ***
income_verVerified          3.25429    0.12970   25.09   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Multiple regression: Bankruptcy example

Regression analysis – Reference group

```
z <- aggregate(interest_rate ~ income_ver, y,
               function(x) c(mean = mean(x),
                             sd = sd(x),
                             n = length(x)))
z <- as.data.frame(do.call(cbind, z))
z[,2:4] <- apply(z[,2:4], 2, as.numeric)
z$min <- z$mean - z$sd / sqrt(z$n)
z$max <- z$mean + z$sd / sqrt(z$n)
z
```

```
       income_ver     mean       sd    n      min      max
1    Not Verified 11.09946 4.569159 3594 11.02324 11.17567
2 Source Verified 12.51548 4.735268 4116 12.44167 12.58929
3        Verified 14.35375 5.447896 2290 14.23990 14.46759
```

Multiple Regression: Bankruptcy Example

Regression Analysis — Reference Group

- This regression output provides multiple rows for the **income verification** variable.
  - Each row represents the relative difference for each level of `income_ver`.
- However, one level is missing: *Not Verified*.
  - The missing level is called the **reference group**, representing the default category that all other levels are measured against.

Multiple Regression: Bankruptcy Example

Regression Interpretation

- The higher interest rate for borrowers who have verified their income is surprising.
  - Intuitively, we might expect verified income to make a loan **less risky**.
- However, the situation may be more complex and could involve **confounding variables** that we haven't accounted for.
  - For example, perhaps lenders require borrowers with **poor credit** to verify their income. In that case, income verification could signal *concern about repayment* rather than *reassurance*. This would make the borrower appear higher risk, leading to a higher interest rate.

Multiple Regression: Bankruptcy Example

Omitted Variable Bias

- **Omitted variable bias** occurs when a model leaves out relevant variables, leading to **biased or misleading estimates** of the included predictors.
  - The effects of the missing variables are incorrectly attributed to those remaining in the model, distorting interpretation.

Multiple regression: Bankruptcy example

Multivariate regression analysis

```r
lm3 <- lm(interest_rate ~ income_ver + debt_to_income + credit_util + bankruptcy +
            term + issued + credit_checks, data = y)

summary(lm3)
```

```
Call:
lm(formula = interest_rate ~ income_ver + debt_to_income + credit_util +
    bankruptcy + term + issued + credit_checks, data = y)

Residuals:
     Min       1Q   Median       3Q      Max
-12.1219  -3.0984  -0.7247   2.3318  18.8160

Coefficients:
                          Estimate Std. Error t value Pr(>|t|)
(Intercept)               1.893839   0.210245   9.008  < 2e-16 ***
income_verSource Verified 0.997468   0.099187  10.056  < 2e-16 ***
income_verVerified        2.563168   0.117184  21.873  < 2e-16 ***
debt_to_income            0.021832   0.002937   7.434 1.14e-13 ***
```

Multiple regression: Bankruptcy example

Multivariate regression analysis – Interpretation

- We estimate the parameters that minimize the sum of the squared residuals:

Multiple regression: Bankruptcy example

Multivariate regression analysis – Coefficients interpretation

- Each coefficient represents the **incremental change** in interest rate for that level, relative to the *Not Verified* group, which serves as the **reference level**.
  - For example, a borrower whose income source and amount have been verified is predicted to have a **3.25 percentage point higher** interest rate than a borrower whose income has not been verified.

Multiple regression: Bankruptcy example

Multivariate regression analysis – Intercept interpretation

- The estimated **intercept** is **1.925**, and one might be tempted to interpret this as the model's **predicted interest rate when all predictors equal zero** — i.e., income source not verified, no debt, zero credit utilization, and so on.
  - **term** (the length of the loan in months) never equals zero — a loan with a term of 0 months would have to be repaid immediately… Therefore, in this context, **the intercept has little practical meaning** and should not be overinterpreted.

Multiple Regression: Bankruptcy Example

Collinearity

- Including multiple predictors helps reduce or eliminate **omitted variable bias**, but another challenge can arise: **correlation among predictors**.
  - When two or more predictors are correlated, we say they are **collinear**.
  - This **collinearity** makes it difficult to disentangle each variable's individual contribution to the response, and can complicate model estimation and interpretation.

Multiple Regression: Bankruptcy Example

Collinearity

```
1 round(cor(y[complete.cases(y),-c(2, 7)]), 2)
```

```
                interest_rate debt_to_income credit_util bankruptcy  term
interest_rate            1.00           0.14        0.25       0.05  0.36
debt_to_income           0.14           1.00        0.13       0.01  0.05
credit_util              0.25           0.13        1.00       0.04 -0.04
bankruptcy               0.05           0.01        0.04       1.00  0.00
term                     0.36           0.05       -0.04       0.00  1.00
credit_checks            0.13           0.03       -0.02       0.08  0.03
                credit_checks
interest_rate            0.13
debt_to_income           0.03
credit_util             -0.02
bankruptcy               0.08
term                     0.03
credit_checks            1.00
```

Multiple Regression: Bankruptcy Example

Adjusted R-Squared

- The regular often **overstates** how much variability the model explains, especially for new samples.
  - To obtain a more reliable measure of model fit, we use the **adjusted** , which penalizes unnecessary predictors and better reflects **true explanatory power**.

Multiple Regression: Bankruptcy Example

Adjusted R-Squared

```
y1 <- y[complete.cases(y),]

r2 <- (var(y1$interest_rate) - var(lm3$residuals)) / var(y1$interest_rate)
r2; summary(lm3)$r.squared
```

```
[1] 0.2603605
```

```
[1] 0.2603605
```

```
n <- nrow(y1)
k <- length(coefficients(lm3)) - 1

adj_r2 <- 1 - (1 - r2) * (n - 1) / (n - k - 1)
adj_r2; summary(lm3)$adj.r.squared
```

```
[1] 0.2596924
```

```
[1] 0.2596924
```

Model Selection

- The **best model** is not always the most complicated.
  - Including variables that are not truly important can actually **reduce predictive accuracy**.
- In this section, we discuss **model selection strategies** that help eliminate variables contributing little to the model's explanatory power.
  - Models that have undergone such variable pruning are often called **parsimonious models** — a term that simply means *efficient and no more complex than necessary*.

Model Selection

- Our goal is to identify a **smaller, more interpretable model** that performs just as well—or even better than a "full" model.
  - **Adjusted** measures the strength of a model's fit while penalizing unnecessary predictors.
  - It helps evaluate which variables are truly **adding value** to the model—that is, improving its ability to **predict future outcomes**.
- "All models are wrong, but some are useful"

Model Selection

Bankruptcy example

```
lm3 <- lm(interest_rate ~ income_ver + debt_to_income + credit_util + bankruptcy +
            term + issued + credit_checks, data = y)

lm4 <- lm(interest_rate ~ income_ver + debt_to_income + credit_util + bankruptcy +
            term + credit_checks, data = y)

summary(lm3)$adj.r.squared; summary(lm4)$adj.r.squared
```

```
[1] 0.2596924
```

```
[1] 0.2597878
```

```
(summary(lm4)$adj.r.squared - summary(lm3)$adj.r.squared)*10000
```

```
[1] 0.9532259
```

Model Selection

Common Strategies

- Two common strategies for adding or removing variables in a multiple regression model are **backward elimination** and **forward selection**.
  - These are often called **stepwise selection methods**, since they add or remove one variable at a time while “stepping” through the candidate predictors.
- **Backward elimination** begins with the **full model**, which includes all potential predictors.
  - Variables are removed **one at a time** until no further improvement in **adjusted** is possible.

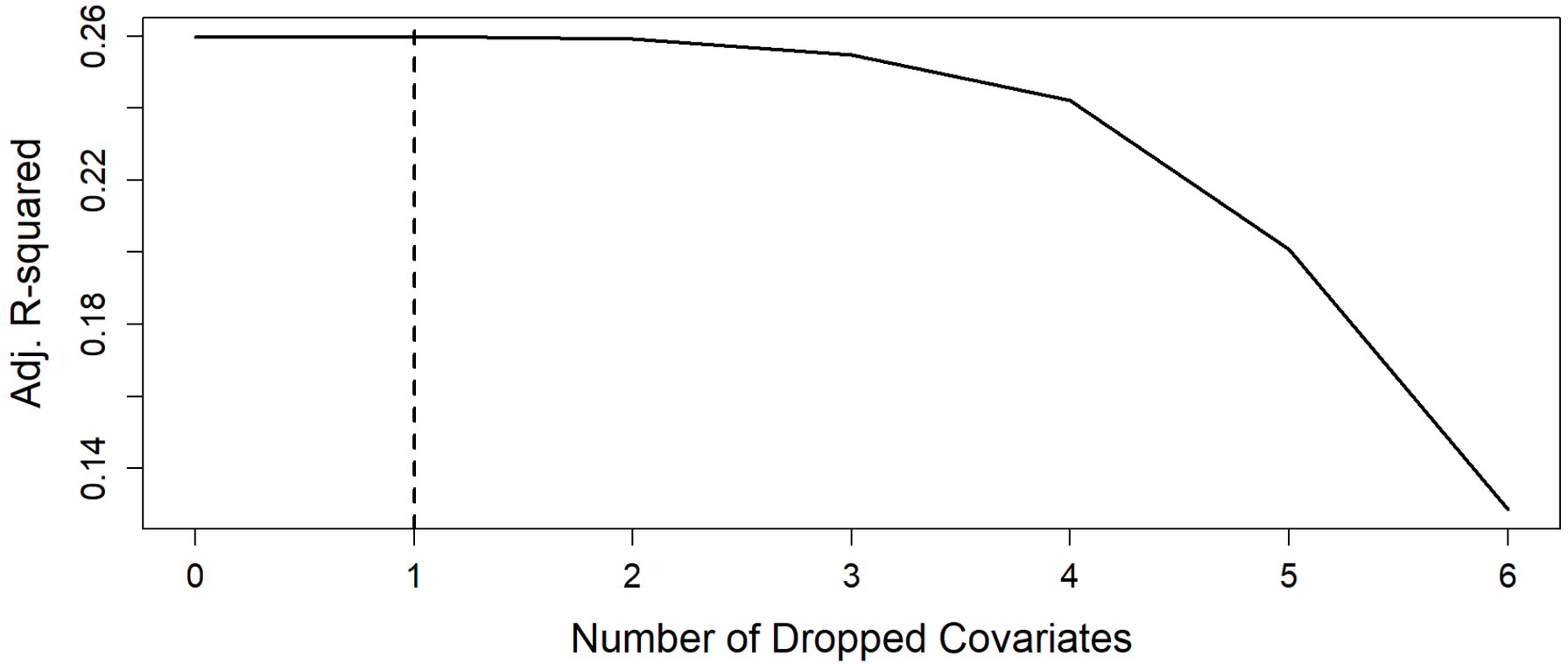
Model Selection

Bankruptcy example

```r
z <- y
lm_full <- lm(interest_rate ~ ., data = z)

res_list <- list()
res_list[[length(res_list) + 1]] <- data.frame(
  drop   = 0,
  adj_r2 = summary(lm_full)$adj.r.squared,
  var    = NA
)

lm_reduced <- lm_full

for(i in 1:6) {
  d1 <- drop1(lm_reduced, test = "F")
  var_to_drop <- rownames(d1)[which.max(d1$`Pr(>F)`)]
  lm_reduced  <- update(lm_reduced, paste(". ~ . -", var_to_drop))

  res_list[[length(res_list) + 1]] <- data.frame(
    drop   = i
```

Model Selection

Bankruptcy example

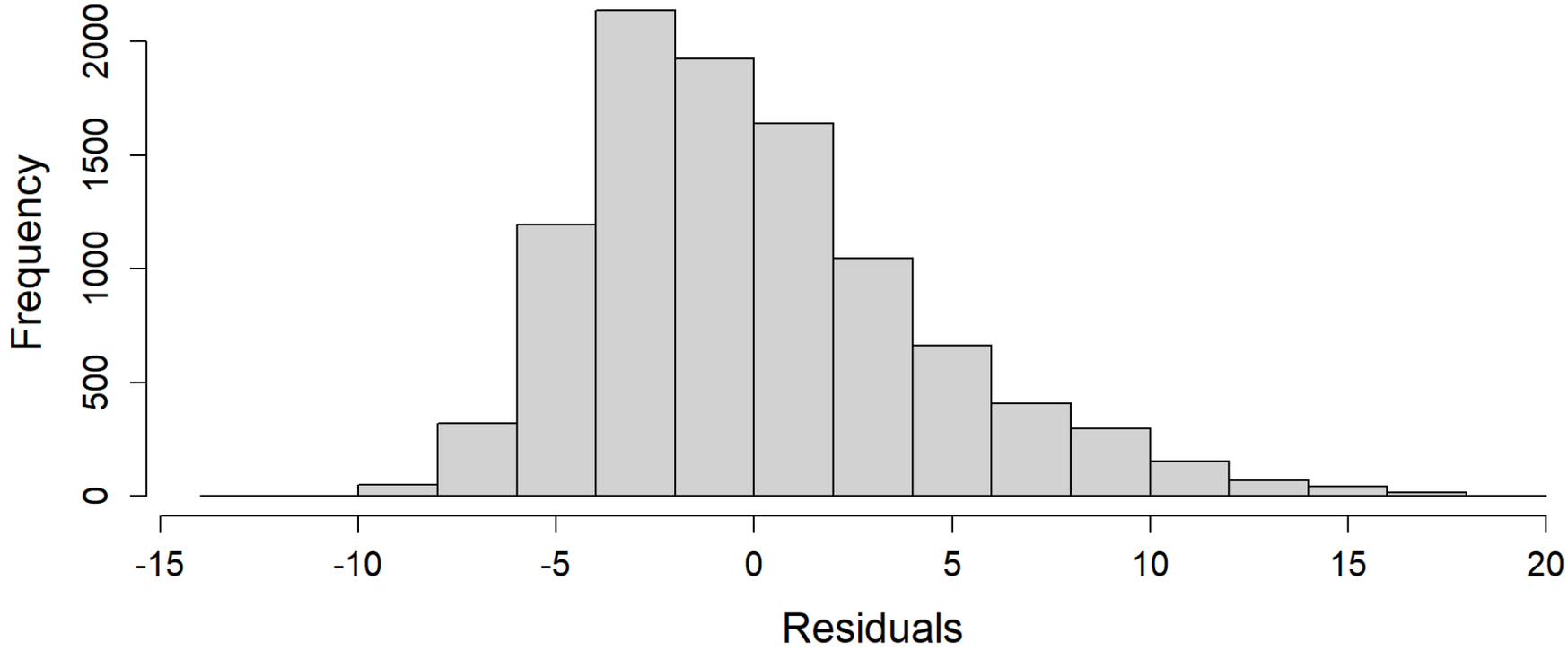


Checking the model

- Multiple regression methods using the model generally depend on the following four conditions:
  - the residuals of the model are nearly normal (less important for larger data sets),
  - the variability of the residuals is nearly constant,
  - the residuals are independent, and
  - each variable is linearly related to the outcome.

Checking the model

Bankruptcy example



Checking the model

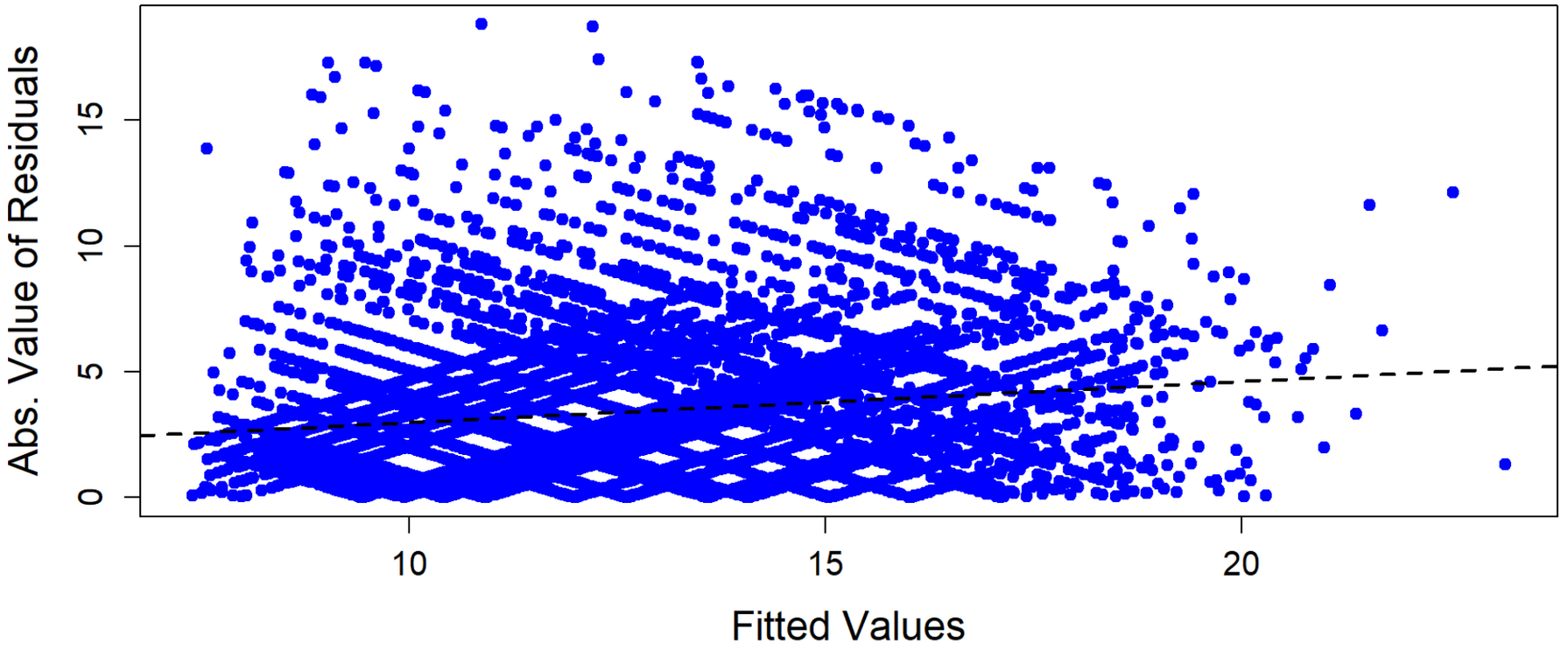
Bankruptcy example

```r
z <- data.frame(
  fitted_values = predict(lm3),
  abs_residuals = abs(lm3$residuals)
)

lm5 <- lm(abs_residuals ~ fitted_values, z)
summary(lm5)
```

```
Call:
lm(formula = abs_residuals ~ fitted_values, data = z)

Residuals:
    Min      1Q  Median      3Q     Max
-4.5888 -1.9387 -0.4386  1.2510 15.6994

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)    1.32720    0.13109   10.12   <2e-16 ***
fitted_values  0.16456    0.01034   15.92   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

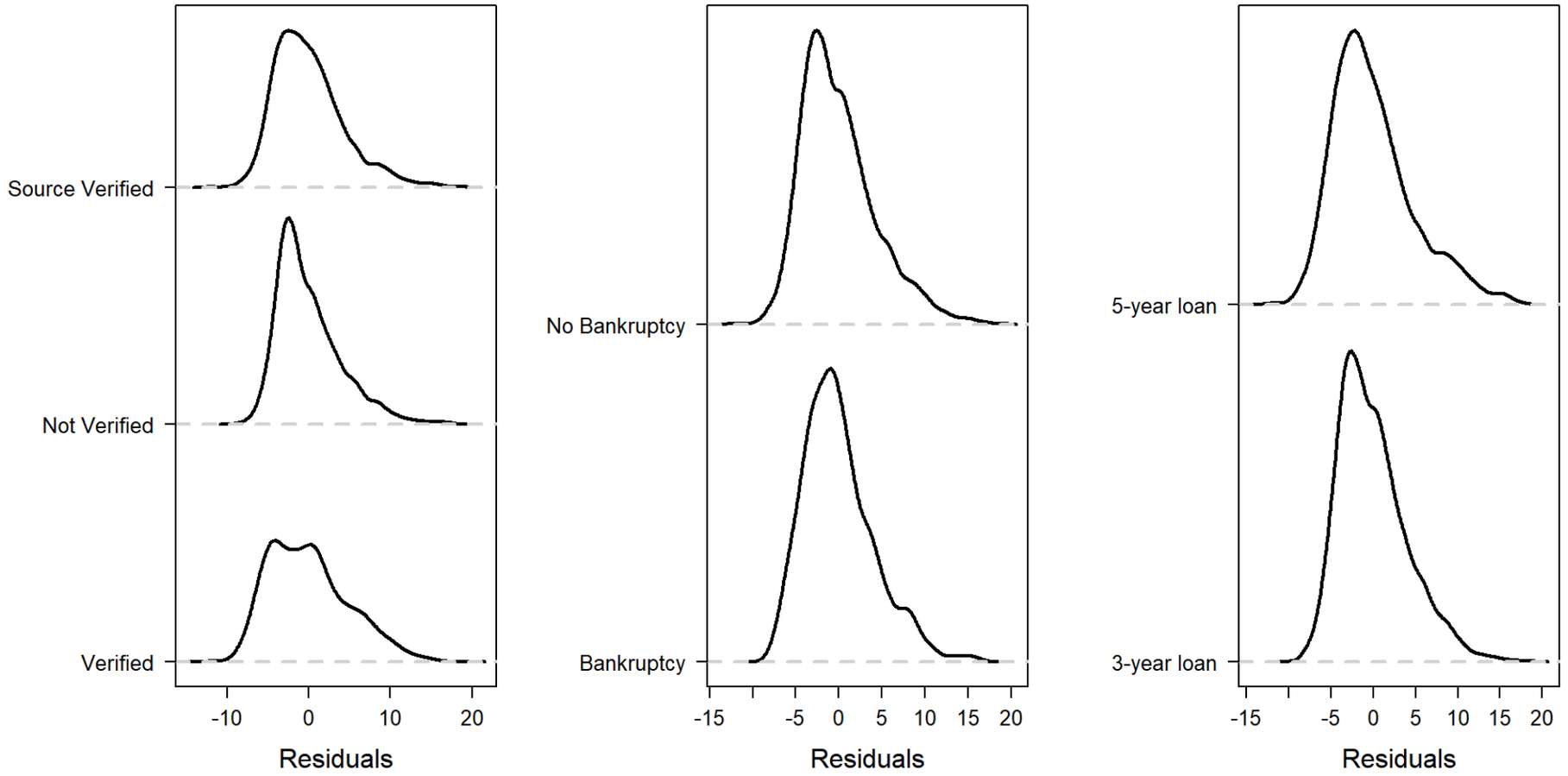
Checking the model

Bankruptcy example



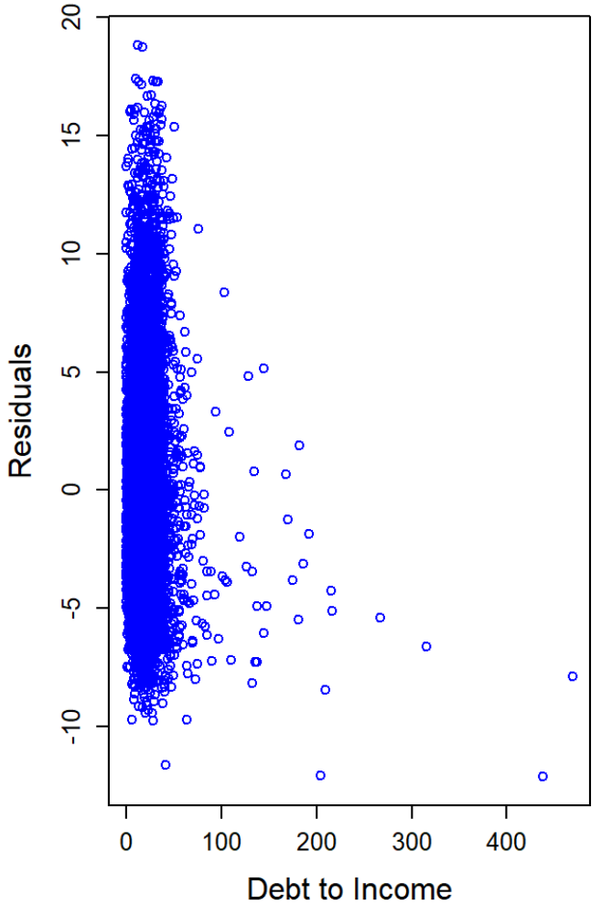
Checking the model

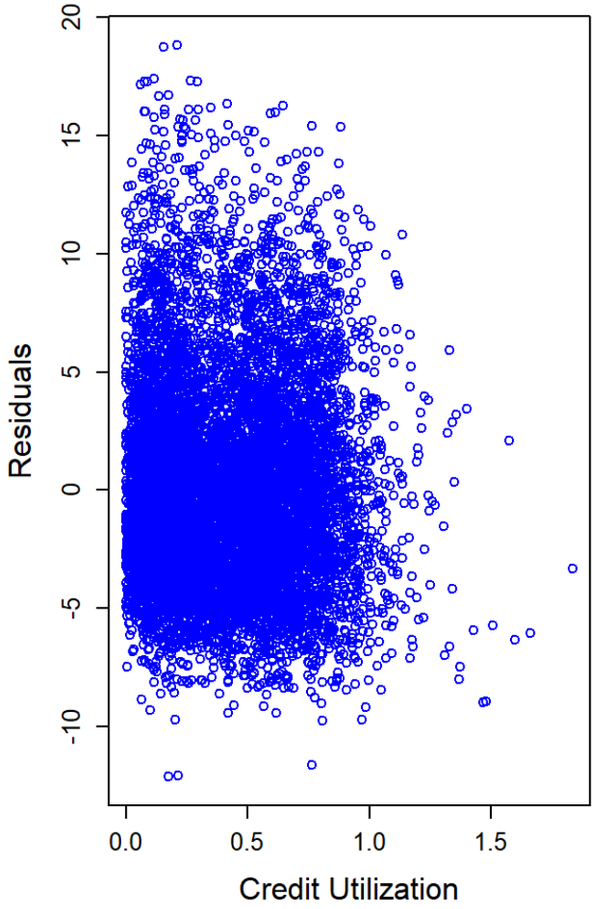
Bankruptcy example

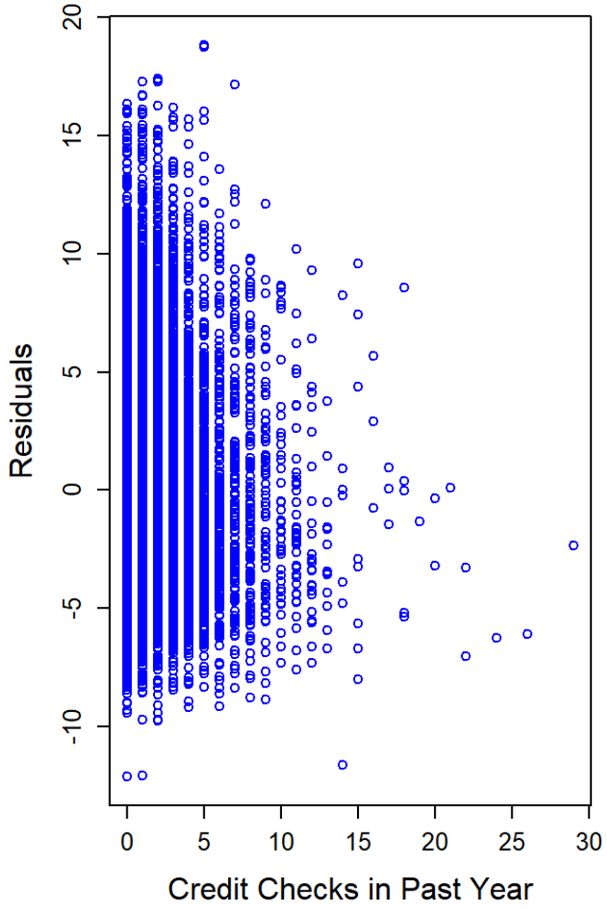


Checking the model

Bankruptcy example

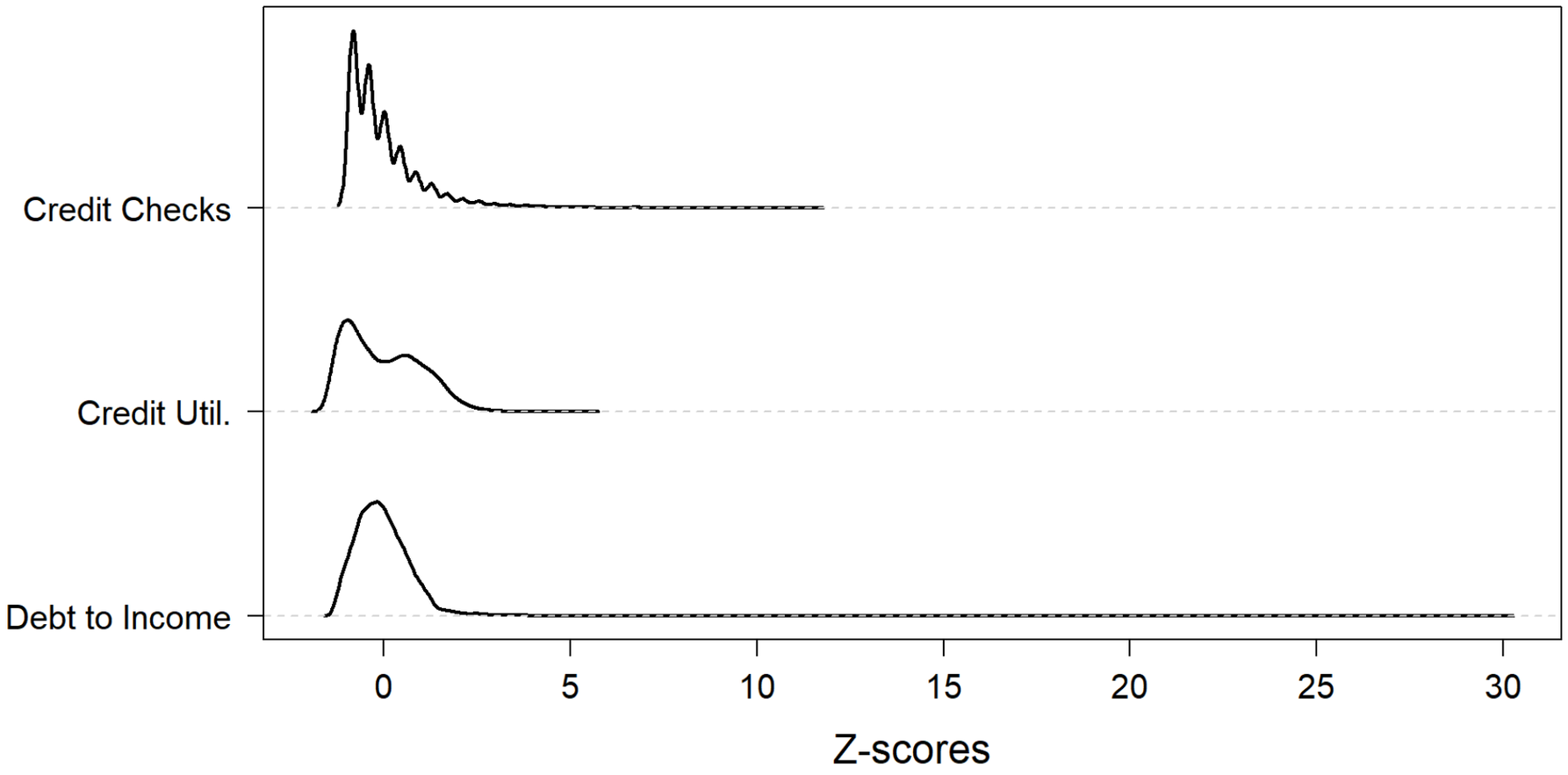






Checking the model

Bankruptcy example

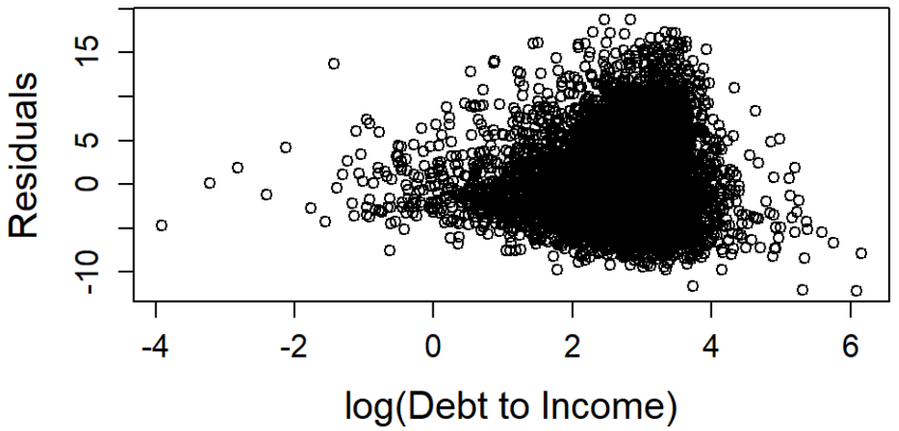


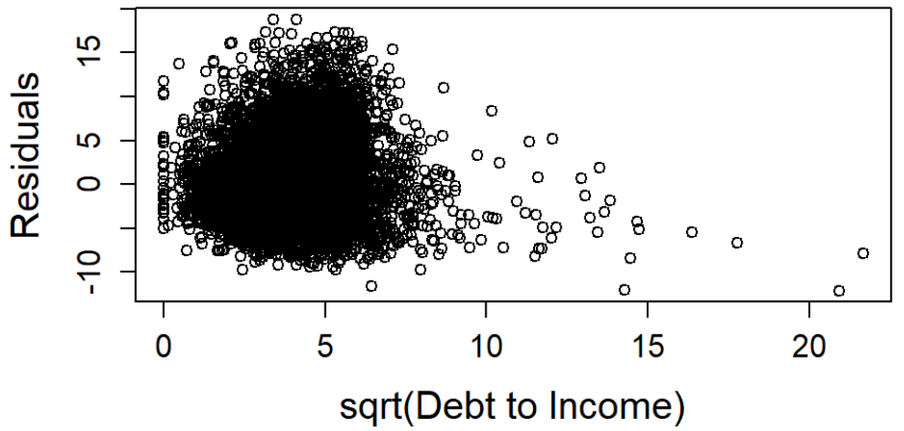
Checking the model

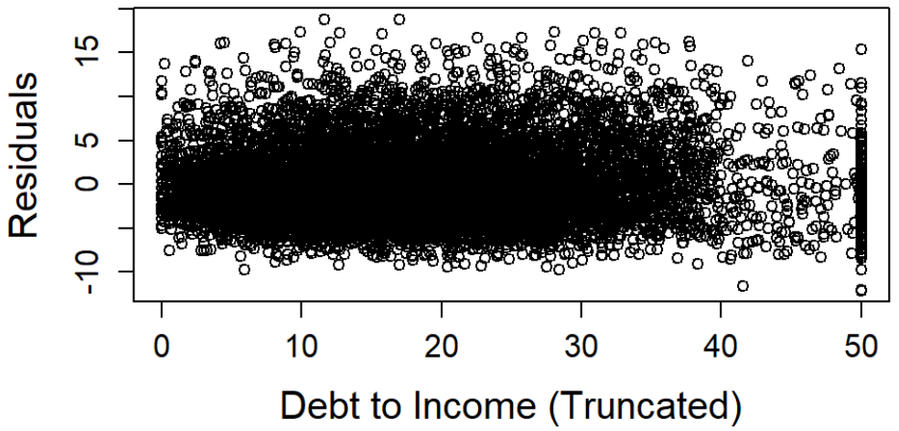
Options for improving the model fit

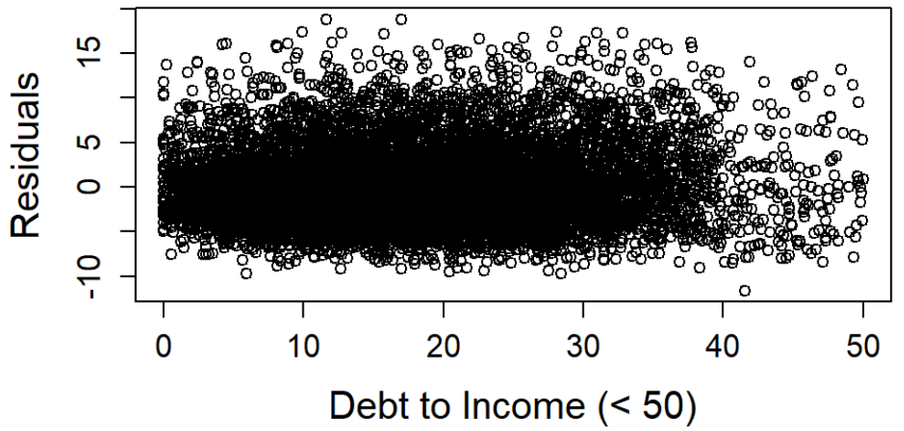
Checking the model

Options for improving the model fit









Checking the model

Options for improving the model fit

```
y2 <- subset(y1, y1$debt_to_income < 50)
lm3_subset <- lm(interest_rate ~ income_ver + debt_to_income + credit_util + bankrup
summary(lm3_subset)
```

```
Call:
lm(formula = interest_rate ~ income_ver + debt_to_income + credit_util +
    bankruptcy + term + issued + credit_checks, data = y2)

Residuals:
     Min       1Q   Median       3Q      Max
-12.2333  -3.0528  -0.7133   2.3294  18.9518

Coefficients:
                           Estimate Std. Error t value Pr(>|t|)
(Intercept)                1.388601   0.215924   6.431 1.33e-10 ***
income_verSource Verified  1.056520   0.099163  10.654  < 2e-16 ***
income_verVerified         2.532965   0.118520  21.372  < 2e-16 ***
debt_to_income             0.059018   0.004660  12.666  < 2e-16 ***
```

Multiple Regression Case Study

Mario Kart Auctions

- We'll examine **eBay auctions** for the video game *Mario Kart* on the **Nintendo Wii**.
  - The **outcome variable** is the **total auction price**, defined as the *winning bid plus shipping cost*.
  - Our goal is to understand how the total price varies with different **auction characteristics**, while **controlling for other factors**.

Multiple Regression Case Study

Mario Kart Auctions

- Holding other characteristics constant:
  - Are **longer auctions** associated with **higher or lower prices**?
  - On average, how much more do buyers pay for **additional Wii wheels** (the plastic steering wheel attachments for Wii controllers)?

Multiple regression

Mario Kart Data

```r
x <- read.csv("mariokart.csv")

x <- data.frame(
  price = x$total_pr,
  cond_new = ifelse(x$cond == "new", 1, 0),
  stock_photo = ifelse(x$stock_photo == "yes", 1, 0),
  duration = x$duration,
  wheels = x$wheels
)

head(x)
```

```
  price cond_new stock_photo duration wheels
1 51.55        1           1        3      1
2 37.04        0           1        7      1
3 45.50        1           0        3      1
4 44.00        1           1        3      1
5 71.00        1           1        1      2
6 45.00        1           1        3      0
```

Multiple regression

Mario Kart data descriptions

variable	description
`price`	Final auction price plus shipping costs, in US dollars.
`cond_new`	Indicator variable for if the game is new (1) or used (0).
`stock_photo`	Indicator variable for if the auction's main photo is a stock photo.
`duration`	The length of the auction, in days, taking values from 1 to 10.
`wheels`	The number of Wii wheels included with the auction. A *Wii wheel* is an optional steering wheel accessory that holds the Wii controller.

Multiple regression

Mario Kart univariate regression

```
1 lm1 <- lm(price ~ cond_new, data = x)
2 summary(lm1)
```

```
Call:
lm(formula = price ~ cond_new, data = x)

Residuals:
    Min      1Q  Median      3Q     Max
-18.168  -7.771  -3.148   1.857 279.362

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   47.148      2.790  16.900   <2e-16 ***
cond_new       6.623      4.343   1.525     0.13
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 25.57 on 141 degrees of freedom
```

Multiple regression

Mario Kart univariate regression

```r
z <- x$price
z <- (z - mean(z)) / sd(z)
x1 <- x[z <= 2.56,]
lm2 <- lm(price ~ cond_new, data = x1)
summary(lm2)
```

```
Call:
lm(formula = price ~ cond_new, data = x1)

Residuals:
     Min       1Q   Median       3Q      Max
-13.8911  -5.8311   0.1289   4.1289  22.1489

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   42.871      0.814  52.668  < 2e-16 ***
cond_new      10.900      1.258   8.662 1.06e-14 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 7.271 on 139 degrees of freedom
```

Multiple regression

Distribution of Residuals

Multiple regression

Multivariate regression

```
1 lm3 <- lm(price ~ cond_new + stock_photo + duration + wheels, data = x1)
2 summary(lm3)
```

```
Call:
lm(formula = price ~ cond_new + stock_photo + duration + wheels,
    data = x1)

Residuals:
     Min       1Q   Median       3Q      Max
-11.3788  -2.9854  -0.9654   2.6915  14.0346

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 36.21097    1.51401  23.917  < 2e-16 ***
cond_new     5.13056    1.05112   4.881 2.91e-06 ***
stock_photo  1.08031    1.05682   1.022    0.308
duration    -0.02681    0.19041  -0.141    0.888
wheels       7.28518    0.55460  13.134  < 2e-16 ***
```

Multiple regression

Distribution of Residuals

Multiple regression

Model Selection

```
z <- list()
for(i in 2:5){
  lm_loop <- lm(price ~ ., x1[,-i])
  z[[length(z)+1]] <- data.frame(
    drop = colnames(x1)[i],
    adj_r2 = summary(lm_loop)$adj.r.squared
  )
}
z <- as.data.frame(do.call(rbind, z))
z$model_pred <- round((z$adj_r2 - summary(lm3)$adj.r.squared)*100, 2)
z <- z[order(-z$adj_r2),]
z
```

```
         drop    adj_r2 model_pred
3    duration 0.7128315       0.21
2 stock_photo 0.7106673      -0.01
1    cond_new 0.6625747      -4.82
4      wheels 0.3486986     -36.21
```

Multiple regression

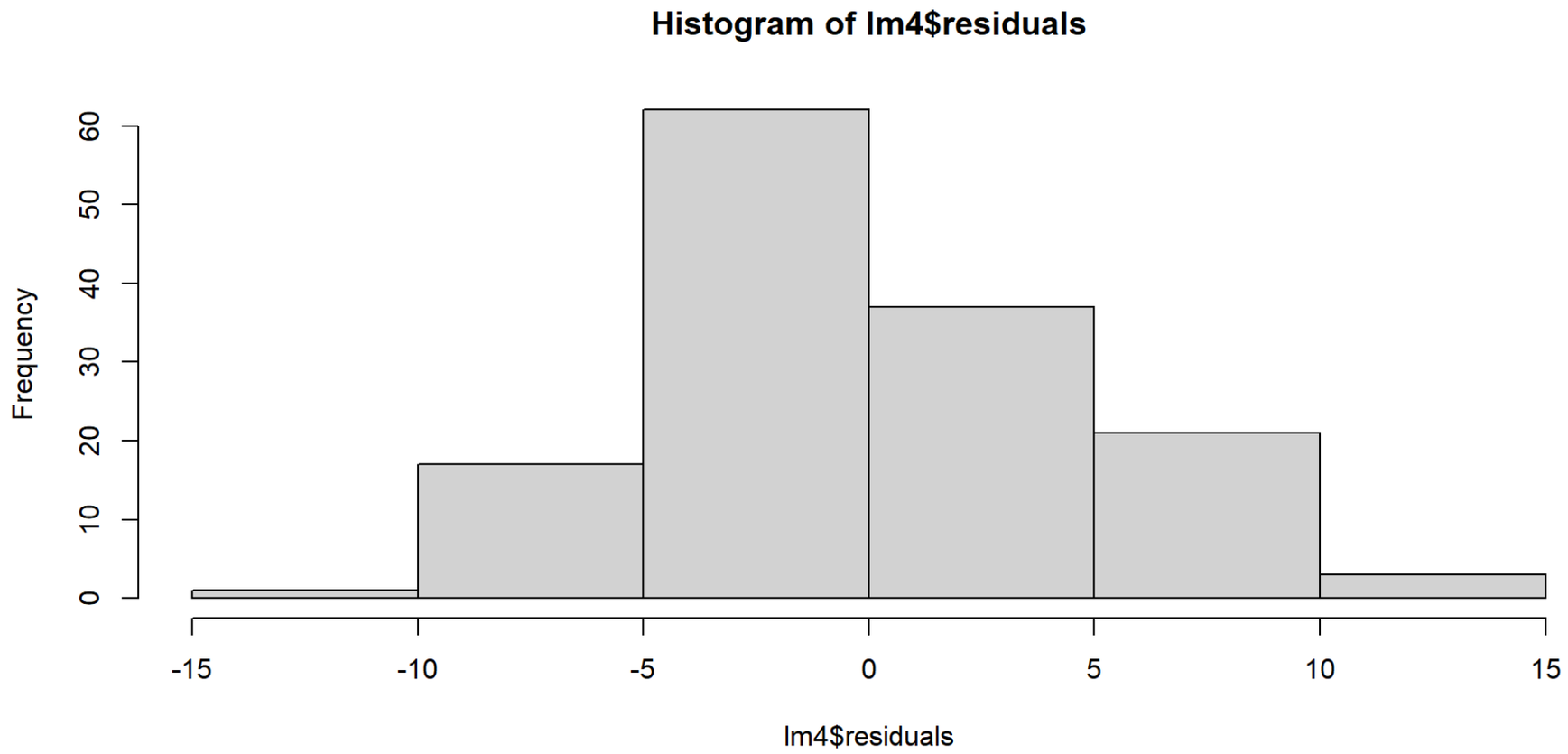
Best Model

```r
lm4 <- lm(price ~ cond_new + stock_photo + wheels, data = x1)
summary(lm4)
```

```
Call:
lm(formula = price ~ cond_new + stock_photo + wheels, data = x1)

Residuals:
    Min      1Q  Median      3Q     Max
-11.454  -2.959  -0.949   2.712  14.061

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  36.0483     0.9745  36.990  < 2e-16 ***
cond_new      5.1763     0.9961   5.196 7.21e-07 ***
stock_photo   1.1177     1.0192   1.097    0.275
wheels        7.2984     0.5448  13.397  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

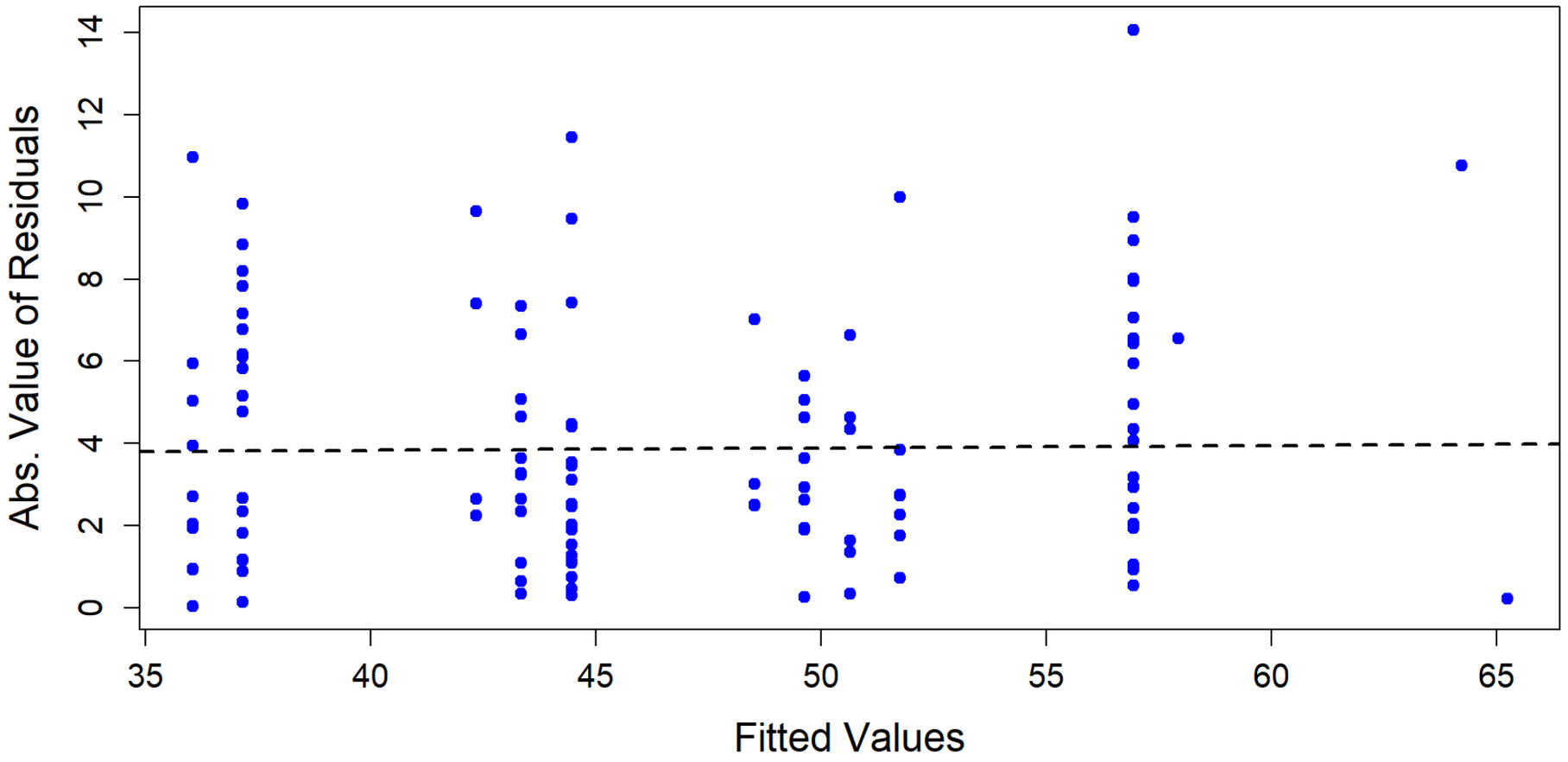
Multiple regression

Distribution of Residuals



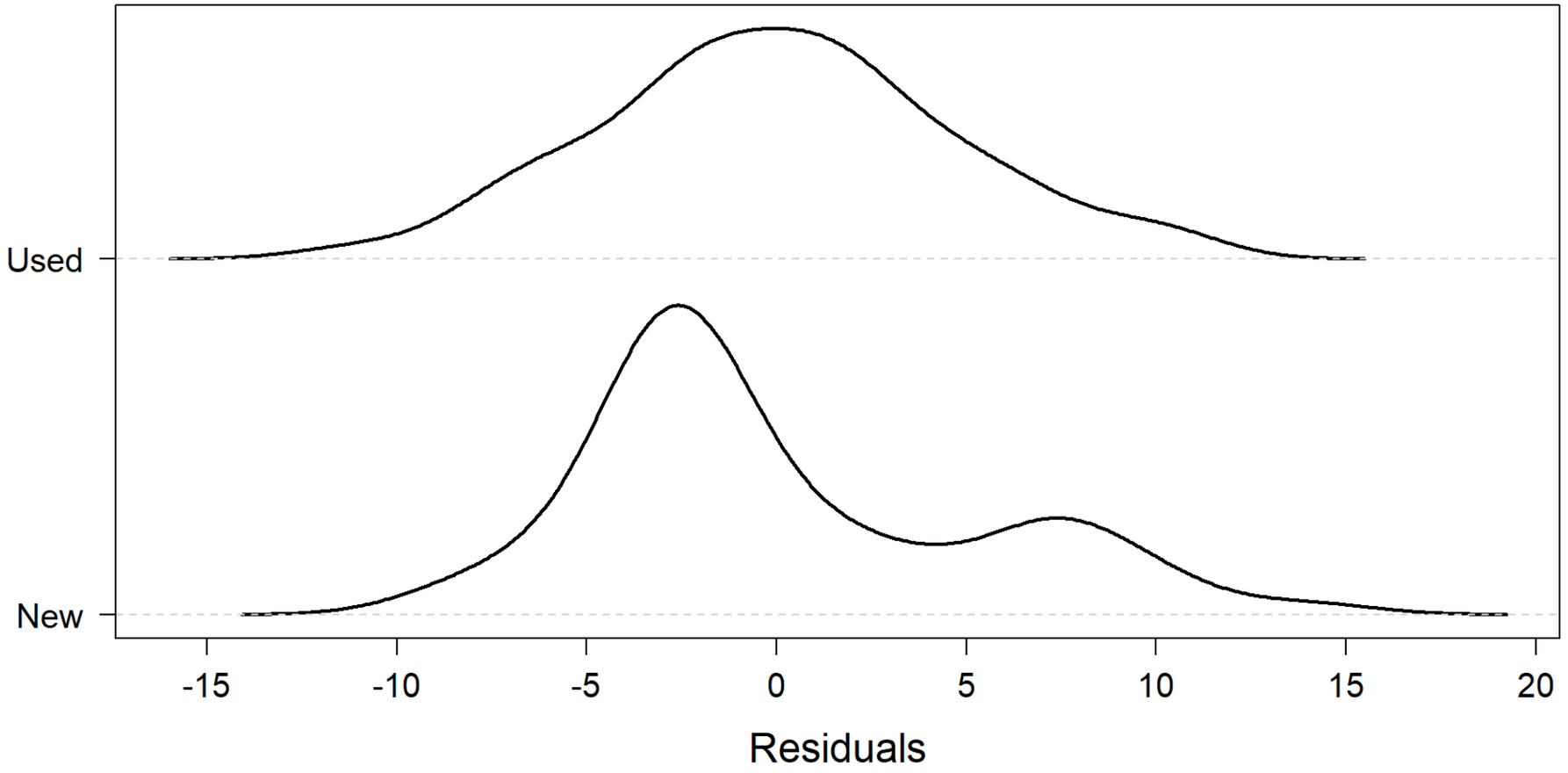
Multiple regression

Model Check



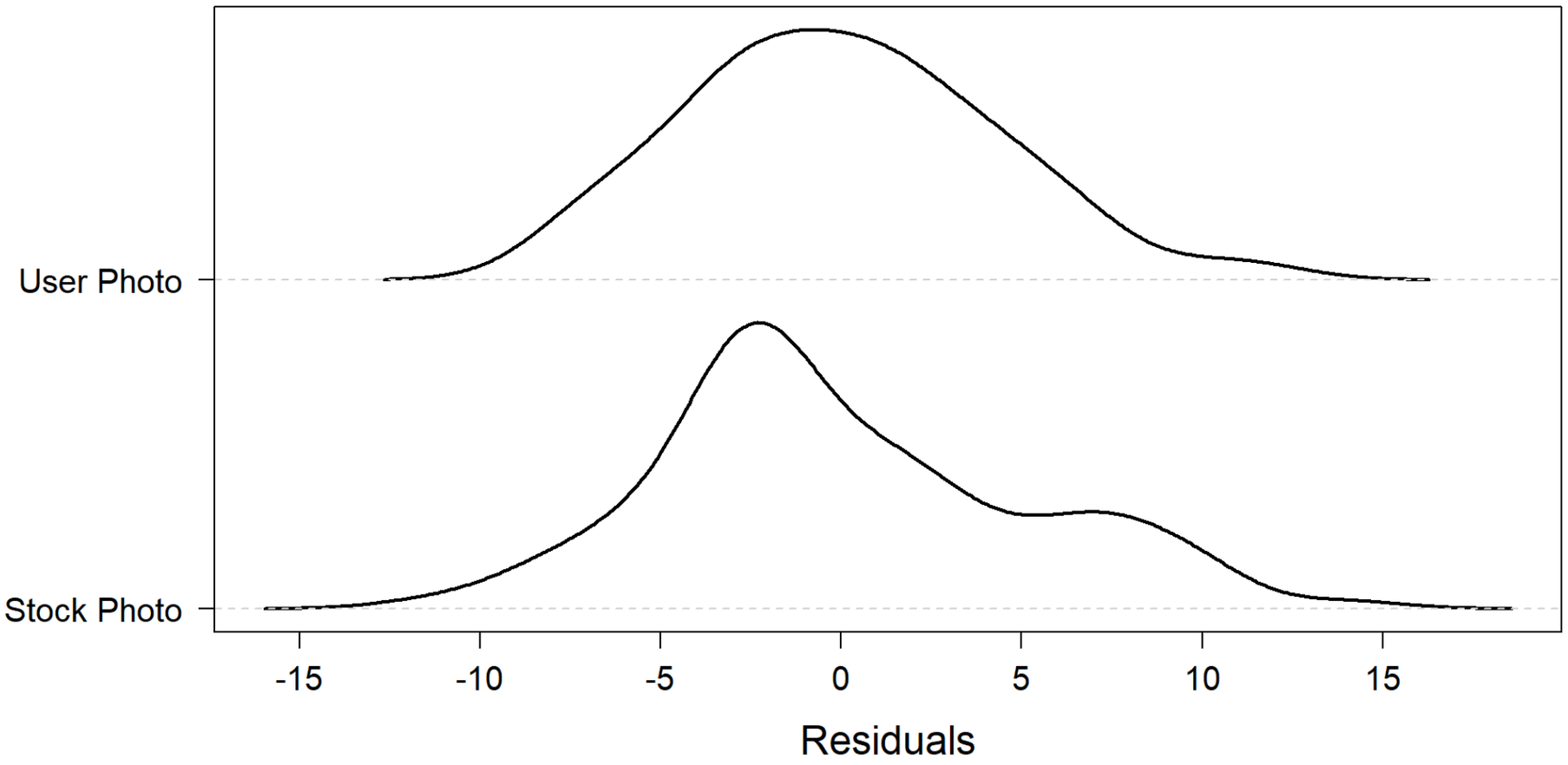
Multiple regression

Model Check



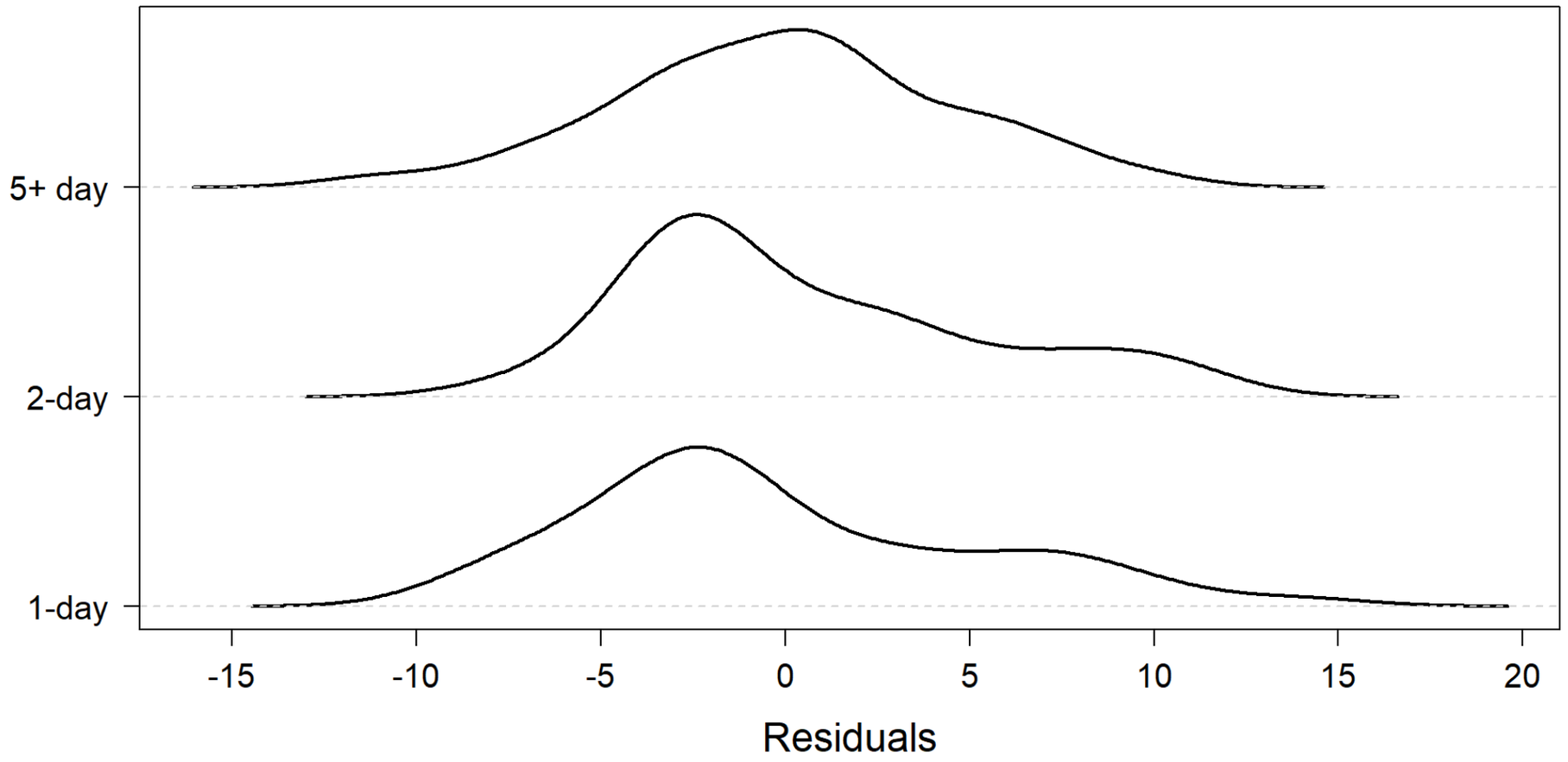
Multiple regression

Model Check



Multiple regression

Model Check



Introduction to Logistic Regression

- In this section, we introduce **logistic regression** as a modeling tool for cases where the **response variable is categorical with two levels**, such as *Yes/No* or *Success/Failure*.
- Logistic regression is a type of **generalized linear model (GLM)** that is well-suited for response variables where **ordinary multiple regression** performs poorly.
  - In these settings, the residuals often **deviate sharply from the normal distribution**, violating key regression assumptions.

Introduction to Logistic Regression

Introduction to Logistic Regression

Model Evaluation

- We will assess model quality using the **Akaike Information Criterion (AIC)** — a measure that balances **model fit** and **model simplicity**.
- Conceptually, AIC plays a similar role here to **adjusted** in multiple regression, helping us compare models and penalize unnecessary complexity.

Logistic Regression

Logistic Regression

Audit Study

Are Emily and Greg More Employable Than Lakisha and Jamal? A Field Experiment on Labor Market Discrimination

Marianne Bertrand

Sendhil Mullainathan

AMERICAN ECONOMIC REVIEW
VOL. 94, NO. 4, SEPTEMBER 2004
(pp. 991–1013)

Download Full Text PDF

Article Information

Abstract

We study race in the labor market by sending fictitious resumes to help-wanted ads in Boston and Chicago newspapers. To manipulate perceived race, resumes are randomly assigned African-American- or White-sounding names. White names receive 50 percent more callbacks for interviews. Callbacks are also more responsive to resume quality for White names than for African-American ones. The racial gap is uniform across occupation, industry, and employer size. We also find little evidence that employers are inferring social class from the names. Differential treatment by race still appears to still be prominent in the U. S. labor market.

Logistic Regression

Audit Study

first_name	race	sex
Aisha	black	female
Allison	white	female
Anne	white	female
Brad	white	male
Brendan	white	male
Brett	white	male
Carrie	white	female
Darnell	black	male
Ebony	black	female
Emily	white	female
Geoffrey	white	male
Greg	white	male

first_name	race	sex
Hakim	black	male
Jamal	black	male
Jay	white	male
Jermaine	black	male
Jill	white	female
Kareem	black	male
Keisha	black	female
Kenya	black	female
Kristen	white	female
Lakisha	black	female
Latonya	black	female
Latoya	black	female

first_name	race	sex
Laurie	white	female
Leroy	black	male
Matthew	white	male
Meredith	white	female
Neil	white	male
Rasheed	black	male
Sarah	white	female
Tamika	black	female
Tanisha	black	female
Todd	white	male
Tremayne	black	male
Tyrone	black	male

Logistic Regression

Audit Study

variable	description
`callback`	Specifies whether the employer called the applicant following submission of the application for the job.
`job_city`	City where the job was located: Boston or Chicago.
`college_degree`	An indicator for whether the resume listed a college degree.
`years_experience`	Number of years of experience listed on the resume.
`honors`	Indicator for the resume listing some sort of honors, e.g. employee of the month.
`military`	Indicator for if the resume listed any military experience.
`email_address`	Indicator for if the resume listed an email address for the applicant.
`race`	Race of the applicant, implied by their first name listed on the resume.
`sex`	Sex of the applicant (limited to only `male` and `female` in this study), implied by the first name listed on the resume.

Audit Study

Data

```r
x <- read.csv("resume.csv")
x <- data.frame(
  callback = x$received_callback,
  chicago = ifelse(x$job_city == "Chicago", 1, 0),
  college_degree = x$college_degree,
  years_experience = x$years_experience,
  military = x$military,
  honors = x$honors,
  email_address = x$has_email_address,
  white = ifelse(x$race == "white", 1, 0),
  male = ifelse(x$gender == "m", 1, 0)
)
```

Audit Study

Data

```
1 head(x)
```

```
  callback chicago college_degree years_experience military honors
1        0       1              1                 6        0       0
2        0       1              0                 6        1       0
3        0       1              1                 6        0       0
4        0       1              0                 6        0       0
5        0       1              0                22        0       0
6        0       1              1                 6        0       1
  email_address white male
1             0     1    0
2             1     1    0
3             0     0    0
4             1     0    0
5             1     1    0
6             0     1    1
```

Audit Study

Correlation matrix

```
1 round(cor(x), 2)
```

```
                  callback chicago college_degree years_experience military
callback              1.00   -0.05          -0.01              0.06    -0.02
chicago              -0.05    1.00          -0.14             -0.19    -0.08
college_degree       -0.01   -0.14           1.00             -0.02     0.02
years_experience      0.06   -0.19          -0.02              1.00    -0.24
military             -0.02   -0.08           0.02             -0.24     1.00
honors                0.07    0.02           0.05              0.13     0.01
email_address         0.03    0.07          -0.06              0.00     0.33
white                 0.06    0.00          -0.01              0.00    -0.02
male                 -0.01   -0.28           0.20             -0.03     0.11
                  honors email_address white  male
callback            0.07          0.03  0.06 -0.01
chicago             0.02          0.07  0.00 -0.28
college_degree      0.05         -0.06 -0.01  0.20
years_experience    0.13          0.00  0.00 -0.03
military            0.01          0.33  0.02  0.11
```

Audit Study

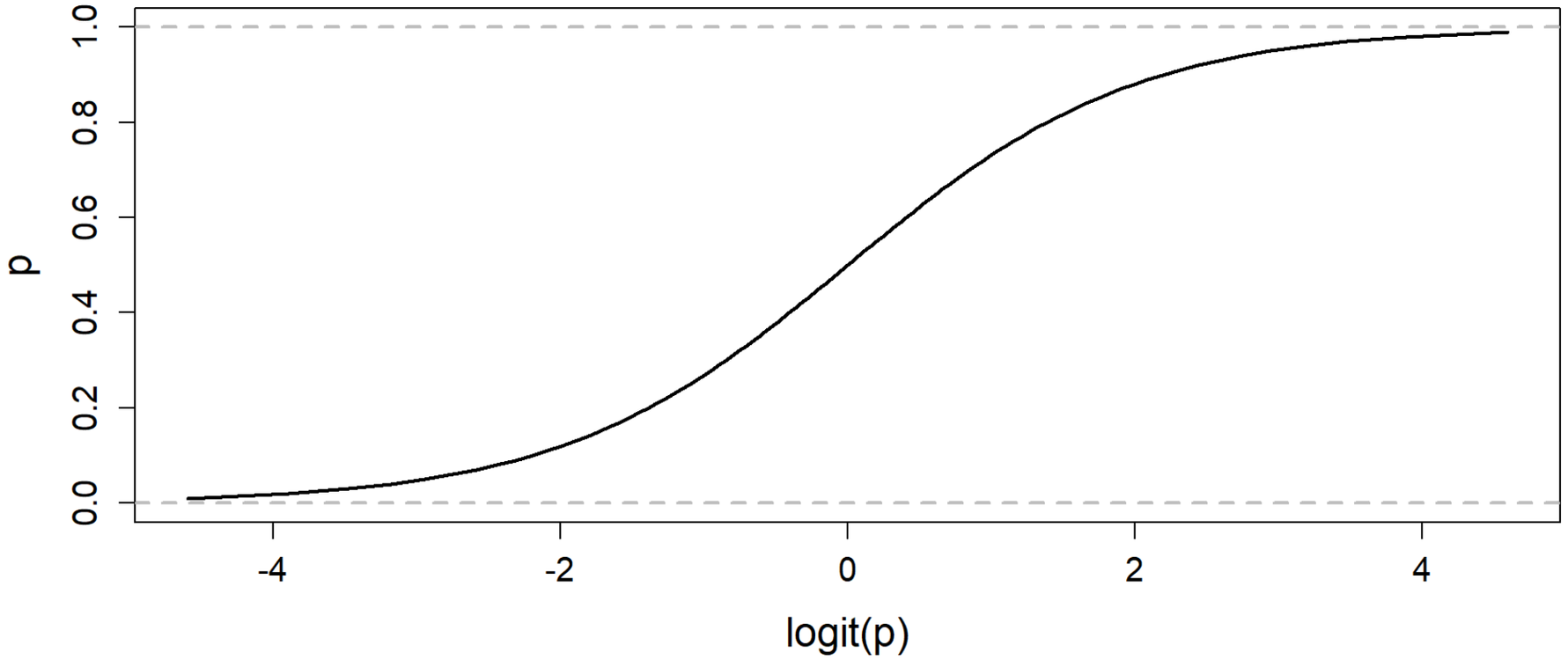
Logit

```r
y <- data.frame(
  p = seq(0, 1, 0.01)
)
y$logit <- log(y$p / (1 - y$p))
head(y)
```

```
     p      logit
1 0.00       -Inf
2 0.01 -4.595120
3 0.02 -3.891820
4 0.03 -3.476099
5 0.04 -3.178054
6 0.05 -2.944439
```

Audit Study

Logit figure



Audit Study

Logit regression

```r
lm1 <- glm(callback ~ honors, data = x, family = binomial(link = "logit"))
summary(lm1)
```

```
Call:
glm(formula = callback ~ honors, family = binomial(link = "logit"),
    data = x)

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -2.4998     0.0556  -44.96  < 2e-16 ***
honors        0.8668     0.1776    4.88 1.06e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2726.9  on 4869  degrees of freedom
Residual deviance: 2706.7  on 4868  degrees of freedom
```

Audit Study

Logit calculation

```
b0 <- as.numeric(coefficients(lm1)[1])
b1 <- as.numeric(coefficients(lm1)[2])
coefs <- b0 + b1

## prob. of callback if no honors
exp(b0) / (1 + exp(b0))
```

```
[1] 0.07587253
```

```
## prob of callback if honors
exp(coefs) / (1 + exp(coefs))
```

```
[1] 0.1634241
```

```
## impact of honors
### this is not equal to b1!
exp(coefs) / (1 + exp(coefs)) - exp(b0) / (1 + exp(b0))
```

```
[1] 0.08755159
```

Audit Study

Logit regression

```r
lm2 <- glm(callback ~ ., data = x, family = binomial(link = "logit"))
summary(lm2)
```

```
Call:
glm(formula = callback ~ ., family = binomial(link = "logit"),
    data = x)

Coefficients:
                  Estimate Std. Error z value Pr(>|z|)
(Intercept)       -2.66318    0.18196 -14.636  < 2e-16 ***
chicago           -0.44027    0.11421  -3.855 0.000116 ***
college_degree    -0.06665    0.12110  -0.550 0.582076
years_experience   0.01998    0.01021   1.957 0.050298 .
military          -0.34217    0.21569  -1.586 0.112657
honors             0.76942    0.18581   4.141 3.46e-05 ***
email_address      0.21826    0.11330   1.926 0.054057 .
white              0.44241    0.10803   4.095 4.22e-05 ***
male               0.19194    0.13757   1.395 0.162960
```

Audit Study

Logit modeling

```r
z <- list()
for(i in 2:8){
  lm_drop <- glm(callback ~ ., data = x[,-i], family = binomial(link = "logit"))
  z[[length(z)+1]] <- data.frame(
    drop = colnames(x)[i],
    aic = summary(lm_drop)$aic
  )

}
z <- as.data.frame(do.call(rbind, z))
z$aic_change <- z$aic - summary(lm2)$aic
z[order(-z$aic),]
```

```
              drop      aic aic_change
7            white 2692.332 15.0855067
5          honors 2690.337 13.0903784
1         chicago 2690.084 12.8376208
3 years_experience 2678.947  1.7000162
6    email_address 2678.944  1.6969088
4         military 2677.908  0.6610767
2   college_degree 2675.548 -1.6990646
```

Audit Study

Surprising result

```
summary(lm(honors ~ college_degree, data = x))
```

```
Call:
lm(formula = honors ~ college_degree, data = x)

Residuals:
     Min       1Q   Median       3Q      Max
-0.05936 -0.05936 -0.05936 -0.03587  0.96413

Coefficients:
               Estimate Std. Error t value Pr(>|t|)
(Intercept)    0.035871   0.006044   5.935 3.14e-09 ***
college_degree 0.023490   0.007125   3.297 0.000985 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.2224 on 4868 degrees of freedom
```

Audit Study

Surprising result

```r
summary(glm(honors ~ college_degree, data = x, family = binomial(link = "logit")))
```

```
Call:
glm(formula = honors ~ college_degree, family = binomial(link = "logit"),
    data = x)

Coefficients:
               Estimate Std. Error z value Pr(>|z|)
(Intercept)     -3.2913     0.1455 -22.622  < 2e-16 ***
college_degree   0.5284     0.1621   3.259  0.00112 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2012.3  on 4869  degrees of freedom
Residual deviance: 2000.6  on 4868  degrees of freedom
```

Audit Study

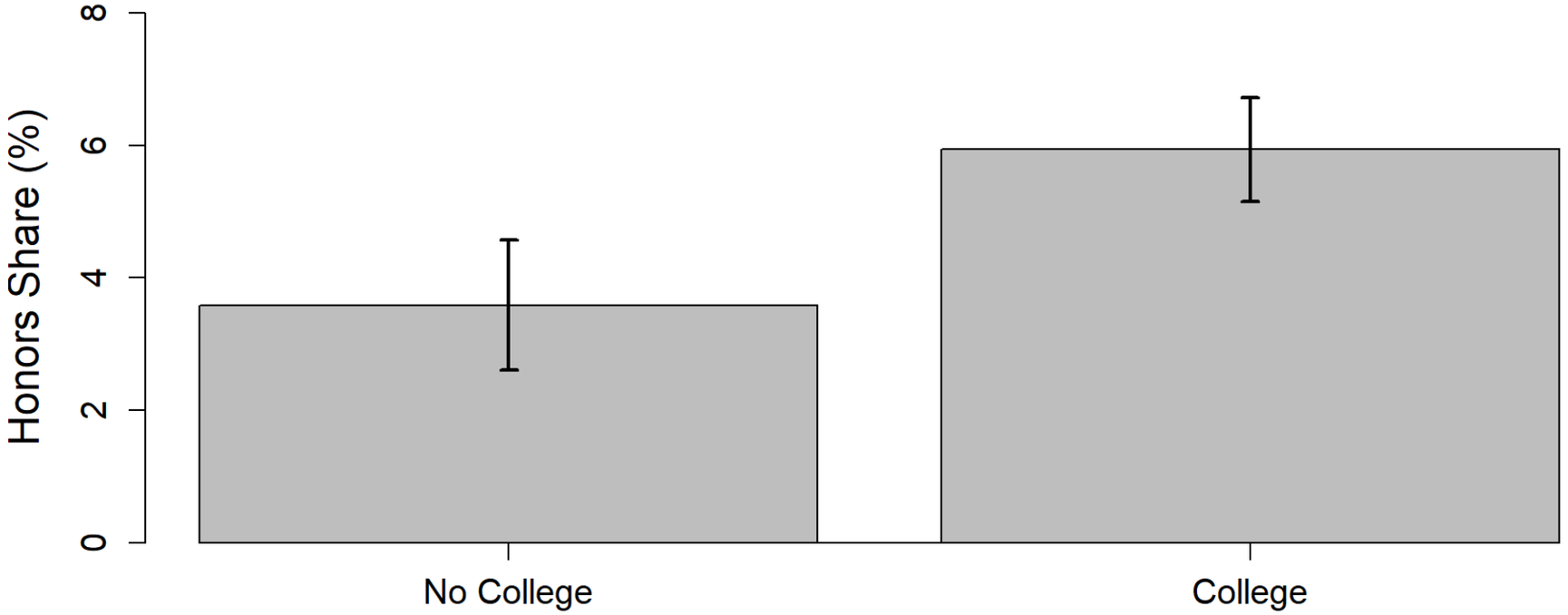
Aggregation

```r
a <- aggregate(honors ~ college_degree, data = x,
               function(x) c(mean = mean(x),
                             sd = sd(x),
                             n = length(x)))
a <- as.data.frame(do.call(cbind, a))
a$min <- a$mean - 1.96 * a$sd / sqrt(a$n)
a$max <- a$mean + 1.96 * a$sd / sqrt(a$n)
a <- a*100
a$college_degree <- a$college_degree / 100
a
```

```
  college_degree     mean       sd      n      min      max
1              0 3.587116 18.60370 136600 2.600541 4.573690
2              1 5.936073 23.63323 350400 5.153550 6.718596
```

Audit Study

Barplot



Audit Study

Best model

```
lm3 <- glm(callback ~ ., data = x[,-3], family = binomial(link = "logit"))
summary(lm3)
```

```
Call:
glm(formula = callback ~ ., family = binomial(link = "logit"),
    data = x[, -3])

Coefficients:
                  Estimate Std. Error z value Pr(>|z|)
(Intercept)       -2.71616    0.15510 -17.513  < 2e-16 ***
chicago           -0.43642    0.11406  -3.826  0.00013 ***
years_experience   0.02055    0.01015   2.024  0.04297 *
military          -0.34426    0.21571  -1.596  0.11050
honors             0.76341    0.18525   4.121 3.77e-05 ***
email_address      0.22208    0.11301   1.965  0.04940 *
white              0.44291    0.10802   4.100 4.13e-05 ***
male              -0.19591    0.13520  -1.449  0.14733
```

Audit Study

Interpret Coefficients

```r
## estimate the probability of receiving a callback for:
### a job in Chicago
### 14 years experience
### no honors, no military experience,
### email address and has a first name that implies they are a White male

z <- as.data.frame(coefficients(lm3))
z <- data.frame(
  var = row.names(z),
  coef = z[,1]
)
z$x <- c(1, 1, 14, 0, 0, 1, 1, 1)
coefs <- sum(z$coef * z$x)
prob_callback <- exp(coefs) / (1 + exp(coefs))
prob_callback
```

```
[1] 0.0834939
```

Audit Study

Interpret Coefficients

```r
## 3.7% more likely than random
(prob_callback / mean(x$callback)) - 1
```

```
[1] 0.03728391
```

```r
## female premium
z$x <- c(1, 1, 14, 0, 0, 1, 1, 0)
coefs <- sum(z$coef * z$x)
prob_callback_f <- exp(coefs) / (1 + exp(coefs))
prob_callback_f - prob_callback
```

```
[1] 0.01626667
```

```r
## black penalty
z$x <- c(1, 1, 14, 0, 0, 1, 0, 1)
coefs <- sum(z$coef * z$x)
prob_callback_b <- exp(coefs) / (1 + exp(coefs))
prob_callback_b - prob_callback
```

```
[1] -0.02822569
```

Audit Study

Predicted probabilities

```r
x$fitted_values <- predict(lm3, family = binomial(link = "logit"))
x$fitted_values <- exp(x$fitted_values) / (1 + exp(x$fitted_values))
lm_check <- lm(fitted_values ~ callback, data = x)
summary(lm_check)
```

```
Call:
lm(formula = fitted_values ~ callback, data = x)

Residuals:
      Min         1Q    Median         3Q        Max
-0.062416 -0.022280 -0.007845  0.011514  0.240854

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 0.0792164  0.0005098 155.381   <2e-16 ***
callback    0.0158576  0.0017970   8.825   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.03412 on 4868 degrees of freedom
```

Audit Study

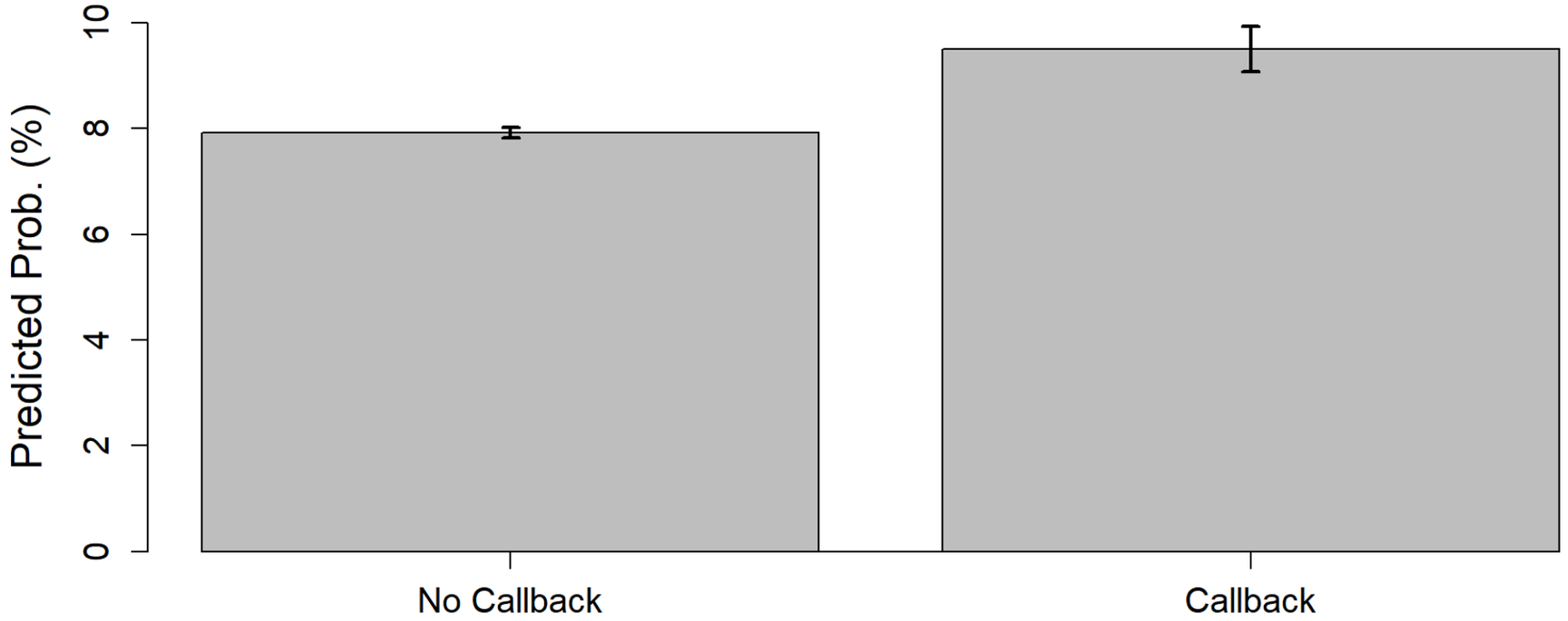
Predicted probabilities

```
a <- aggregate(fitted_values ~ callback, x,
               function(x) c(mean = mean(x),
                             sd = sd(x),
                             n = length(x)))
a <- as.data.frame(do.call(cbind, a))
a$min <- a$mean - 1.96 * a$sd / sqrt(a$n)
a$max <- a$mean + 1.96 * a$sd / sqrt(a$n)
a <- a*100
a[,1] <- a[,1] / 100
a
```

```
  callback     mean       sd      n      min      max
1        0 7.921639 3.318709 447800 7.824435 8.018843
2        1 9.507400 4.335592  39200 9.078199 9.936602
```

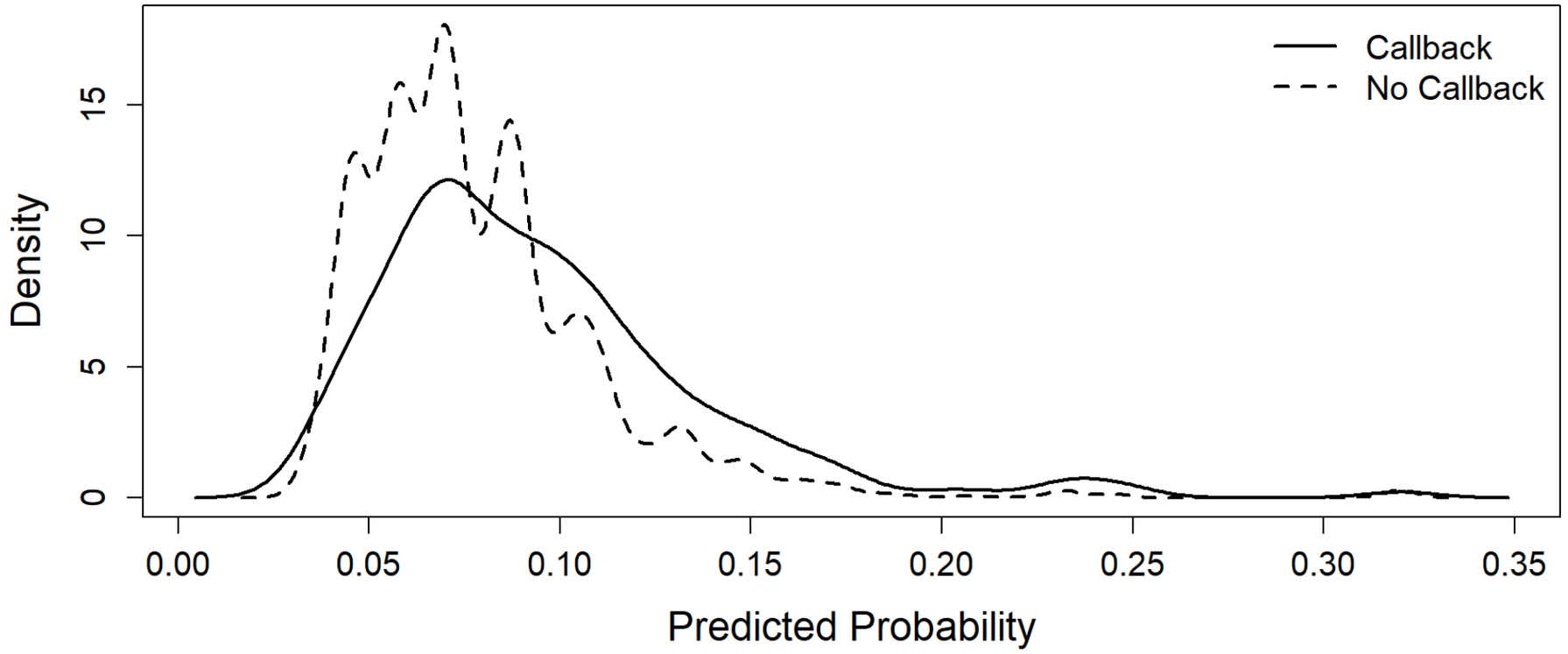
Audit Study

Predicted probabilities: Barplot



Audit Study

Predicted probabilities: histograms



Audit Study

Model check

```r
minz <- min(x$fitted_values)
maxz <- max(x$fitted_values)
loops <- seq(minz, maxz, 0.05)
n_loops <- length(loops) - 1
y <- list()
for(i in 2:n_loops){
  z <- x[x$fitted_values >= loops[i-1] & x$fitted_values < loops[i],]
  y[[length(y)+1]] <- data.frame(
    midpoint = loops[i-1] + (loops[i] - loops[i-1])/2,
    mean = mean(z$callback),
    sd = sd(z$callback),
    n = nrow(z)
  )
}
y <- as.data.frame(do.call(rbind, y))
y$min <- y$mean - 1.96 * y$sd / sqrt(y$n)
y$max <- y$mean + 1.96 * y$sd / sqrt(y$n)
y
```

```
    midpoint       mean         sd    n        min        max
1 0.05409131 0.06005789 0.2376371 2764 0.05119855 0.06891722
```

Audit Study

Model check: scatterplot