Rotman

INTRO TO R PROGRAMMING

R Tutorial (RSM358) – Session 4

October 1, 2024 Prepared by Jay Cao / TDMDAL

Website: https://tdmdal.github.io/r-intro-2024-fall/

Rotman School of Management
UNIVERSITY OF TORONTO

A2: Scatter Plot & Regression Line Plot

```
# simple linear regression
my_lm <- lm(formula = y ~ x, data = my_data)

# scatter plot of y against x
plot(my_data$x, my_data$y)    # note that x-coord first
plot(my_data$y ~ my_data$x)    # alternatively, use formula

# draw regression line
abline(my_lm)
```

A2: CI and PI - Setup

- True model, true Y, and true coefficients β_i,

$$Y = f(X) + \epsilon = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \epsilon, \text{ where } \mathbb{E}(\epsilon) = 0.$$

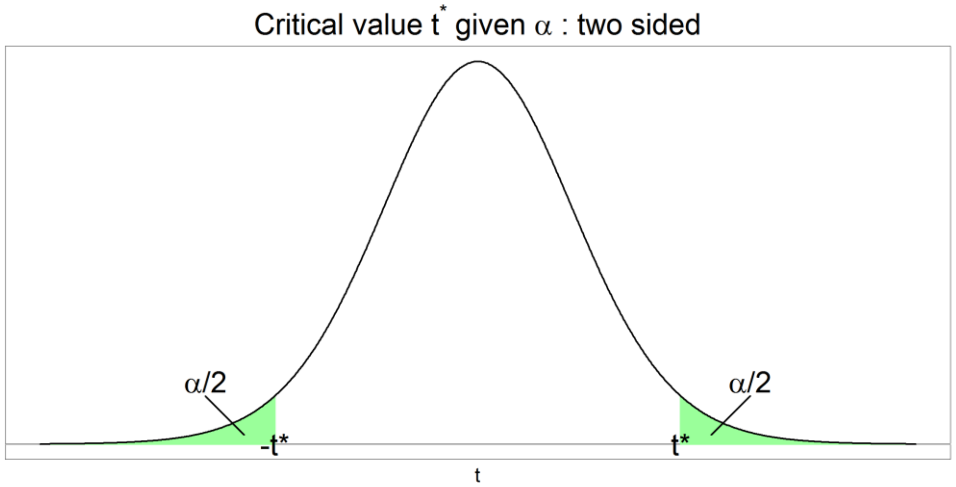
- Estimated model, predicted $\hat{Y}$, and estimated coefficients $\hat{\beta}_i$,

$$\hat{Y} = \hat{f}(X) = \hat{\beta}_0 + \hat{\beta}_1 X_1 + \hat{\beta}_2 X_2$$

- R code

```
my_lm <- lm(formula = y ~ x1 + x2, data = my_df)
```

A2: CI & PI – CI of β_i



- Under "usual/standard" assumptions,

$$\frac{\hat{\beta}_i - \beta_i}{\widehat{SE}(\hat{\beta}_i)} \sim t_{n-p},$$

So,

$$\mathbb{P}\left(-t_{n-p,\frac{\alpha}{2}} < \frac{\hat{\beta}_i - \beta_i}{\widehat{SE}(\hat{\beta}_i)} < t_{n-p,\frac{\alpha}{2}}\right) = 1 - \alpha,$$

Now, we get CI

$$[\hat{\beta}_i - \widehat{SE}(\hat{\beta}_i) t_{n-p,\frac{\alpha}{2}}, \hat{\beta}_i + \widehat{SE}(\hat{\beta}_i) t_{n-p,\frac{\alpha}{2}}]$$

- Interpretation (e.g., $\alpha = 95\%$)?
  - "If we take repeated samples and construct the confidence interval for each sample, 95% of the intervals will contain the true unknown value of the parameter." – from you textbook

- R code: `confint(my_lm)`

Notations: t_{n-p} is t-dist w/ df $n-p$, n is # of obs., and p is # of parameters; α is significance level and $t_{n-p,\frac{\alpha}{2}}$ is critical value (the t^* in the graph) such that the prob of the t_{n-p} distribution to the right of it is $\frac{\alpha}{2}$; $\widehat{SE}(\hat{\beta}_i)$ is $SE(\hat{\beta}_i)$ in your textbook.

A2: CI & PI – CI of $\mathbb{E}(Y) = f(X)$

- α level CI of $\mathbb{E}(Y) = f(X)$ at $X = x_0$ can be derived similarly

- Interpretation (say, $\alpha = 95\%$): "95% of intervals of this form will contain the true value of $f(X)$" – from your textbook
  - To be precise, $f(X = x_0)$
  - What does it mean "of this form"?
    - If we take repeated samples and construct the confidence interval for each sample, …

- R code
  - `predict(my_lm, new_data, interval = "confidence")`

A2: CI & PI – PI of $Y = f(X) + \epsilon$

- α level PI of $Y = f(X) + \epsilon$ at $X = x_0$
- Wider than the corresponding CI
  - because it accounts for the irreducible error
- Interpretation?
- R code
  - `predict(my_lm, new_data, interval = "prediction")`

A3 - Q14 Data Simulation, Any Questions?

- Q14 in Section 3.7: data simulation

```
# simulation in Q14
set.seed(1)
x1 <- runif(100)
x2 <- 0.5 * x1 + rnorm(100) / 10
y <- 2 + 2 * x1 + 0.3 * x2 + rnorm(100)

# additional observation
x1 <- c(x1 , 0.1)
x2 <- c(x2 , 0.8)
y <- c(y, 6)
```

Logistic Regression - Lab 4.7

- `my_model <- glm(formula = …, data = …, family=binomial)`
- `summary(my_model)`

- `predict(my_model, newdata = …, type = "response")`
  - Set the argument `type = "response"` to get predicted probabilities, i.e., $P(Y = 1|X)$
  - Otherwise, `predict(my_model)` gives log odds (logit)
  - If the `newdata` argument is not supplied, the prediction is applied on the training data set
  - Use `contrast()` to find out which y category is set to 1.

- Construct confusing matrix
  - Convert probability prediction to binary prediction (cutoff prob.)
  - `table()`

A3 - Q14/a Prepare Data, Any Questions?

- Q14 in Section 4.8: load and prepare the binary y

```
# Q14/a load the data
Auto <- read.csv("Auto.csv", na.strings = "?")
Auto$origin <- as.factor(Auto$origin)

# prepare the binary variable y
Auto$mpg01 = ifelse(Auto$mpg > median(Auto$mpg), 1, 0)
```

Training & Test Set - Lab 4.7

- Training and test set split
  - For time series data, need to respect the time when splitting the data
    - That is, train on early data, test on late data
  - Otherwise, randomly split data to train and test
- A time series training & test set split from lab 4.7
  - `Year` and `Direction` are columns in the `Smarket` dataset
    - the `Smarket` data is “attached”

```
> train <- (Year < 2005)
> Smarket.2005 <- Smarket[!train, ]
> dim(Smarket.2005)
[1] 252   9
> Direction.2005 <- Direction[!train]
```

Training & Test Set – A3 Q14/c

- Training and test set split
  - For time series data, need to respect the time when splitting the data
    - That is, train on early data, test on late data
  - Otherwise, randomly split data to train and test

```
# Q14/c randomly split Auto dataset into training and test set
num_rows <- nrow(Auto)
train_fraction <- 0.7
train_idx = sample(1:num_rows, size = round(num_rows * train_fraction))
train_data <- Auto[train_idx, ]
test_data <- Auto[-train_idx, ]
```

Confusion Matrix and Error Rate - Lab 4.7

```
> glm.fits <- glm(
    Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume,
    data = Smarket, family = binomial, subset = train
  )
> glm.probs <- predict(glm.fits, Smarket.2005,
    type = "response")
```

```
> glm.pred <- rep("Down", 252)
> glm.pred[glm.probs > .5] <- "Up"
> table(glm.pred, Direction.2005)
        Direction.2005
glm.pred Down Up
    Down   77 97
    Up     34 44
> mean(glm.pred == Direction.2005)
[1] 0.48
> mean(glm.pred != Direction.2005)
[1] 0.52
```