Simple Linear Regression - Exam

Katlyn H. Degamo

2023-06-29

1. The fractional Distillation Data

```
x_hydrocarbon <- c(1.02, 1.11, 1.43, 1.11, 1.01, 0.95, 1.11, 0.87, 1.43, 1.02,
                  1.46, 1.55, 1.55, 1.55, 1.40, 1.15, 1.01, 0.99, 0.95, 0.98)
y_purity <- c(86.91, 89.85, 90.28, 86.34, 92.58, 87.33, 86.29, 91.86, 95.61, 89.86,

                     96.73, 99.42, 98.66, 96.07, 93.65, 87.31, 95.00, 96.85, 85.20, 90.56)
x = x_hydrocarbon
y = y_purity
```

a. create a scatter diagram for the table

```
library(ggplot2)
```

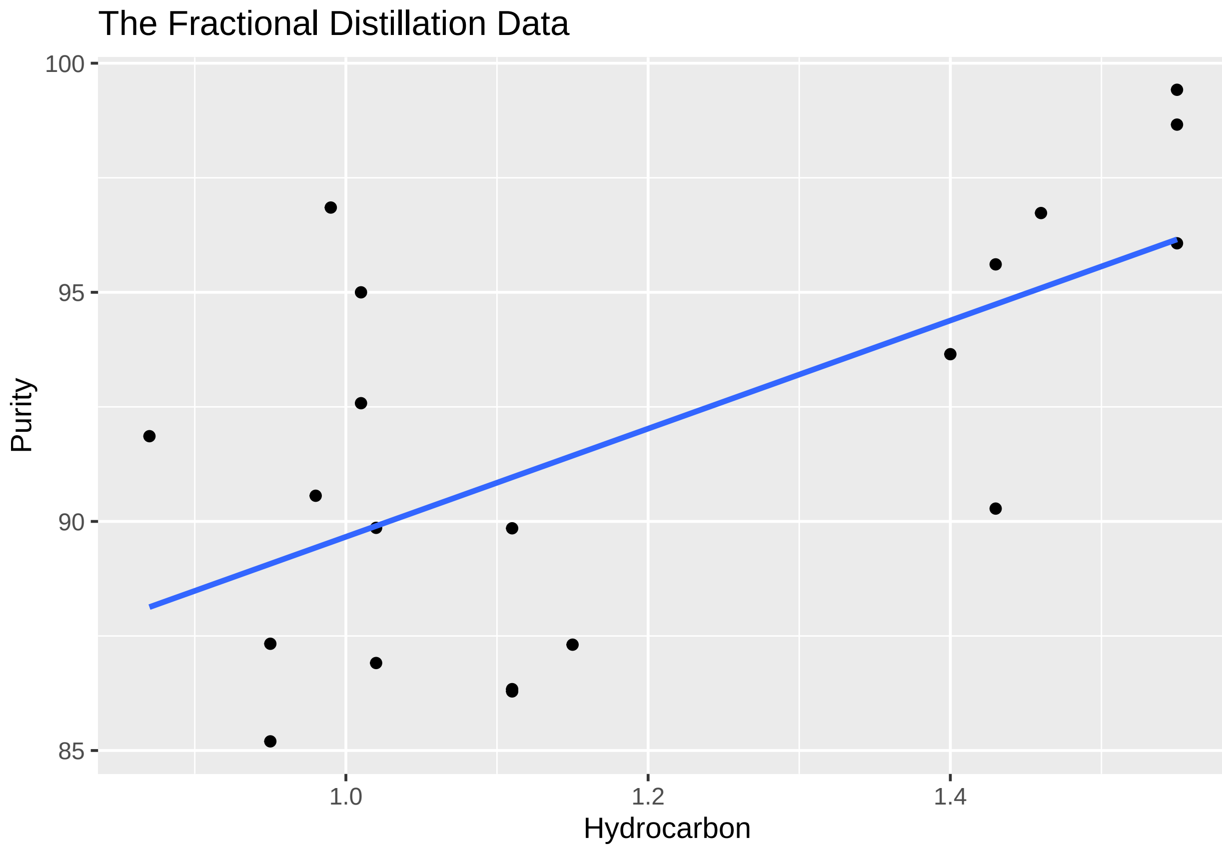
```
## Warning: package 'ggplot2' was built under R version 4.2.3
```

```
# Display Scatter Diagram

scatter_diagram <- ggplot(data = NULL,
                          aes(x = x_hydrocarbon, y= y_purity)) +
                          geom_point() + geom_smooth(method = "lm",
                          se = FALSE) +
                          labs(title = "The Fractional Distillation Data",
                          x = " Hydrocarbon", y = "Purity")

print(scatter_diagram)
```

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



b. The least-square fit is

```
# Creating the linear regression model
model <- lm(y ~ x)

# Getting the coefficients
(coefficients <- model$coefficients)
```

```
## (Intercept)           x
##    77.86328    11.80103
```

```
# Getting the R-squared value
(r_squared <- summary(model)$r.squared)
```

```
## [1] 0.3891224
```

```
# Getting the p-values for the coefficients
p_values <- summary(model)$coefficients[, "Pr(>|t|)"]
print(p_values)
```

```
##  (Intercept)            x
## 3.537382e-13 3.291122e-03
```

```
model_summary <- summary(model)
model_summary
```

```
##
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.6724 -3.2113 -0.0626  2.5783  7.3037
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   77.863      4.199  18.544 3.54e-13 ***
## x             11.801      3.485   3.386  0.00329 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.597 on 18 degrees of freedom
## Multiple R-squared:  0.3891, Adjusted R-squared:  0.3552
## F-statistic: 11.47 on 1 and 18 DF,  p-value: 0.003291
```

```
# Function for simple linear regression
simple_linear_regression <- function(x, y) {
  # Creating the linear regression model
  model <- lm(y ~ x)

  # Extracting coefficients
  coefficients <- model$coefficients

  # Extracting R-squared value
  r_squared <- summary(model)$r.squared

  # Returning coefficients and R-squared value
  return(list(coefficients = coefficients, r_squared = r_squared))
}

result <- simple_linear_regression(x, y)
print(result$coefficients)
```

```
## (Intercept)           x
##    77.86328    11.80103
```

```
print(result$r_squared)
```

```
## [1] 0.3891224
```

```
#Creating Linear Model
model_names <- names(model)

# Inspecting the names of objects in the model
print(model_names)
```

```
##  [1] "coefficients"  "residuals"     "effects"       "rank"
##  [5] "fitted.values" "assign"        "qr"            "df.residual"
##  [9] "xlevels"       "call"          "terms"         "model"
```

```
# Function to get the least-squares fit coefficients
leastsquaresfit <- function(model){
  leastsfit <- model$coefficients
  return(leastsfit)
}

leastfits <- leastsquaresfit(model)

output <- paste("The least-squares fit coefficients:",
                paste("Intercept =", leastfits[1]),
                paste("PropellantAge =", leastfits[2]), sep = "\n")
message(output)
```

```
## The least-squares fit coefficients:
## Intercept = 77.8632841616
## PropellantAge = 11.8010281931501
```

```
least_fits <- lsfit(x, y)$coefficients
print(least_fits)
```

```
## Intercept         X
##  77.86328  11.80103
```

c. The estimate of sigma squared is

```
getsigmasquared <- function(model){
  modelsummary <- summary(model)
  sigmasquared <- modelsummary$sigma^2
  return(sigmasquared)
}

estimatesigmasquared <- getsigmasquared(model)

output <- paste("The Estimate of Sigma Squared is equal to:", estimatesigmasquared)

message(output)
```

```
## The Estimate of Sigma Squared is equal to: 12.9352421041182
```

d. Test for significance of regression in the regression model.

```
# Another way to find the p value
p_value <- summary(model)$coefficients[2, 4]

# Print the p-value
print(p_value)
```

```
## [1] 0.003291122
```

```r
# Get the coefficient and standard error for a specific variable, e.g., "x"
coefficient <- coef(model)["x"]
standard_error <- summary(model)$coefficients["x", "Std. Error"]

# Calculate the t-value
t_value <- coefficient / standard_error

# Print the t-value
print(t_value)
```

```
##        x
## 3.386119
```

```r
# Set the significance level
significance_level <- 0.05

# Set the degrees of freedom
degrees_of_freedom <- 18

# Calculate the critical value
critical_value <- qt(1 - (significance_level / 2), df = degrees_of_freedom)

# Print the critical value
print(critical_value)
```

```
## [1] 2.100922
```

```r
# Perform the significance test for regression
if (abs(t_value) > critical_value) {
    null_hypothesis_status <- "The null hypothesis is rejected."
  } else {
    null_hypothesis_status <- "The null hypothesis is not rejected."
  }

output <- paste("Significance test for regression:",
                paste("T Value =", t_value),
                paste("Critical Value =", critical_value),
                paste(null_hypothesis_status), sep = "\n")

message(output)
```

```
## Significance test for regression:
## T Value = 3.3861194436304
## Critical Value = 2.10092204024104
## The null hypothesis is rejected.
```

e. Use an analysis-of-variance approach to test significance of regression.

```r
# Set the significance level
significance_level <- 0.01

# Specify the degrees of freedom for the numerator and denominator
df_numerator <- 1  # Degrees of freedom for the numerator
df_denominator <- 18  # Degrees of freedom for the denominator

# Calculate the critical value
critical_value <- qf(1 - significance_level, df_numerator, df_denominator)

# Print the critical value
print(critical_value)
```

```
## [1] 8.28542
```

```r
# Perform the analysis-of-variance approach to test significance of regression
if (model_summary$fstatistic["value"] > critical_value) {
  null_hypothesis_status <- "The null hypothesis is rejected."
}else {
  null_hypothesis_status <- "The null hypothesis is not rejected."
}

output = paste("The Analysis Of Variance:",
              paste("F Statistics:", model_summary$fstatistic["value"] ),
              paste("Critical Value:", critical_value),
              paste(null_hypothesis_status), sep = "\n")

message(output)
```

```
## The Analysis Of Variance:
## F Statistics: 11.4658048865319
## Critical Value: 8.28541955509965
## The null hypothesis is rejected.
```

f. Find a 95%CI on the slope

```r
# Calculate the confidence interval
conf_interval <- confint(model, level = 0.95)

conf_interval
```

```
##                 2.5 %   97.5 %
## (Intercept) 69.041747 86.68482
## x            4.479066 19.12299
```

g. Find a 95%CI on the mean purity when the hydrocarbon percentage is 1.00

```r
x_pred <- 1.00
y_pred <- predict(model, newdata = data.frame(x =x_pred), interval= "confidence", level = 0.95)
y_ci <- y_pred[, c("lwr", "upr")]
y_ci
```

```
##      lwr      upr
## 87.51017 91.81845
```

2. The Steam Consumption Data

```r
x_tempearture <- c(21, 24, 32, 47, 50, 59, 68, 74, 62, 50, 41, 30)
y_usage <- c(185.79, 214.47, 288.03, 424.84, 454.68, 539.03, 621.55, 675.06, 562.03, 452.93, 369.95, 27
x = x_tempearture
y = y_usage
```

a. create a scatter diagram for the table

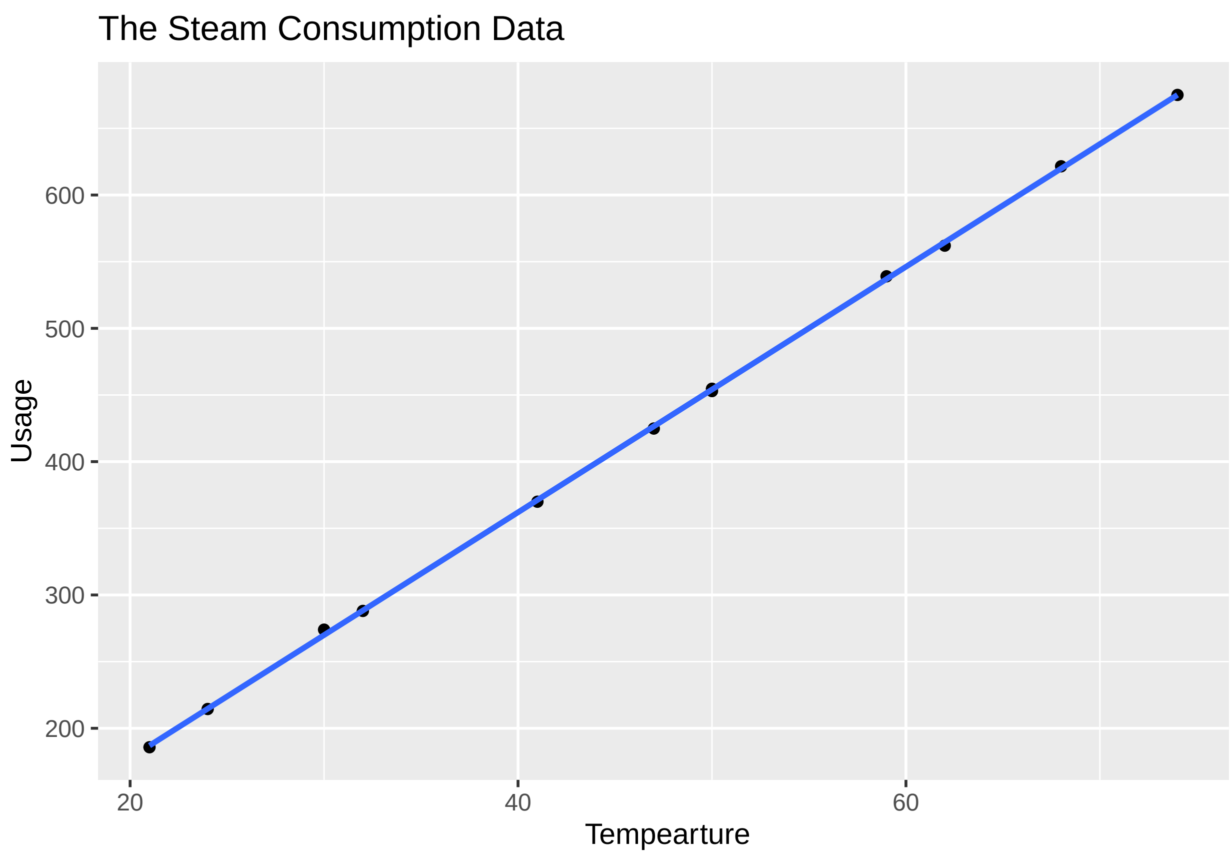
```r
library(ggplot2)

# Display Scatter Diagram

scatter_diagram <- ggplot(data = NULL,
                          aes(x = x_tempearture, y= y_usage)) +
                          geom_point() + geom_smooth(method = "lm",
                          se = FALSE) +
                          labs(title = "The Steam Consumption Data",
                          x = " Tempearture", y = "Usage")

print(scatter_diagram)
```

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



b. The least-square fit is

```r
# Creating the linear regression model
model <- lm(y ~ x)

# Getting the coefficients
(coefficients <- model$coefficients)
```

```
## (Intercept)           x
##   -6.332087    9.208468
```

```r
# Getting the R-squared value
(r_squared <- summary(model)$r.squared)
```

```
## [1] 0.9998651
```

```r
# Getting the p-values for the coefficients
p_values <- summary(model)$coefficients[, "Pr(>|t|)"]
print(p_values)
```

```
##  (Intercept)            x
## 3.534310e-03 1.099192e-20
```

```r
model_summary <- summary(model)
model_summary
```

```
##
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.5629 -1.2581 -0.2550  0.8681  4.0581
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -6.33209    1.67005  -3.792  0.00353 **
## x            9.20847    0.03382 272.255  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.946 on 10 degrees of freedom
## Multiple R-squared:  0.9999, Adjusted R-squared:  0.9999
## F-statistic: 7.412e+04 on 1 and 10 DF,  p-value: < 2.2e-16
```

```r
# Function for simple linear regression
simple_linear_regression <- function(x, y) {
  # Creating the linear regression model
  model <- lm(y ~ x)

  # Extracting coefficients
  coefficients <- model$coefficients

  # Extracting R-squared value
  r_squared <- summary(model)$r.squared

  # Returning coefficients and R-squared value
  return(list(coefficients = coefficients, r_squared = r_squared))
}

result <- simple_linear_regression(x, y)
print(result$coefficients)
```

```
## (Intercept)           x
##   -6.332087    9.208468
```

```r
print(result$r_squared)
```

```
## [1] 0.9998651
```

```r
#Creating Linear Model
model_names <- names(model)

# Inspecting the names of objects in the model
print(model_names)
```

```
##  [1] "coefficients"  "residuals"     "effects"       "rank"
##  [5] "fitted.values" "assign"        "qr"            "df.residual"
##  [9] "xlevels"       "call"          "terms"         "model"
```

```
# Function to get the least-squares fit coefficients
leastsquaresfit <- function(model){
  leastsfit <- model$coefficients
  return(leastsfit)
}

leastfits <- leastsquaresfit(model)

output <- paste("The least-squares fit coefficients:",
                paste("Intercept =", leastfits[1]),
                paste("PropellantAge =", leastfits[2]), sep = "\n")
message(output)
```

```
## The least-squares fit coefficients:
## Intercept = -6.33208673315187
## PropellantAge = 9.20846781504986
```

```
least_fits <- lsfit(x, y)$coefficients
print(least_fits)
```

```
## Intercept         X
## -6.332087  9.208468
```

c. The estimate of sigma squared is

```
getsigmasquared <- function(model){
  modelsummary <- summary(model)
  sigmasquared <- modelsummary$sigma^2
  return(sigmasquared)
}

estimatesigmasquared <- getsigmasquared(model)

output <- paste("The Estimate of Sigma Squared is equal to:", estimatesigmasquared)

message(output)
```

```
## The Estimate of Sigma Squared is equal to: 3.78546984889691
```

d. Test for significance of regression in the regression model.

```
# Another way to find the p value
p_value <- summary(model)$coefficients[2, 4]

# Print the p-value
print(p_value)
```

```
## [1] 1.099192e-20
```

```r
# Get the coefficient and standard error for a specific variable, e.g., "x"
coefficient <- coef(model)["x"]
standard_error <- summary(model)$coefficients["x", "Std. Error"]

# Calculate the t-value
t_value <- coefficient / standard_error

# Print the t-value
print(t_value)
```

```
##       x
## 272.255
```

```r
# Set the significance level
significance_level <- 0.05

# Set the degrees of freedom
degrees_of_freedom <- 10

# Calculate the critical value
critical_value <- qt(1 - (significance_level / 2), df = degrees_of_freedom)

# Print the critical value
print(critical_value)
```

```
## [1] 2.228139
```

```r
# Perform the significance test for regression
if (abs(t_value) > critical_value) {
    null_hypothesis_status <- "The null hypothesis is rejected."
  } else {
    null_hypothesis_status <- "The null hypothesis is not rejected."
  }

output <- paste("Significance test for regression:",
                paste("T Value =", t_value),
                paste("Critical Value =", critical_value),
                paste(null_hypothesis_status), sep = "\n")

message(output)
```

```
## Significance test for regression:
## T Value = 272.254998757911
## Critical Value = 2.22813885198627
## The null hypothesis is rejected.
```

e. Use an analysis-of-variance approach to test significance of regression.

```r
# Set the significance level
significance_level <- 0.01

# Specify the degrees of freedom for the numerator and denominator
df_numerator <- 1  # Degrees of freedom for the numerator
df_denominator <- 10  # Degrees of freedom for the denominator

# Calculate the critical value
critical_value <- qf(1 - significance_level, df_numerator, df_denominator)

# Print the critical value
print(critical_value)
```

```
## [1] 10.04429
```

f. Find a 95%CI on the slope

```r
# Calculate the confidence interval
conf_interval <- confint(model, level = 0.95)

conf_interval
```

```
##                  2.5 %    97.5 %
## (Intercept) -10.053181 -2.610993
## x             9.133106  9.283830
```

g. Find a 95%CI on the mean purity when the hydrocarbon percentage is 1.00

```r
x_pred <- 1.00
y_pred <- predict(model, newdata = data.frame(x =x_pred), interval= "confidence", level = 0.95)
y_ci <- y_pred[, c("lwr", "upr")]
y_ci
```

```
##        lwr        upr
## -0.7738282  6.5265904
```