MATH 70076 Data Science

2023/24 - Assessment 2

02405885

2024/04/20

1 Fake-data simulation and regression:

To simulate 100 data points from the linear model y=3+4x+error, we'll first generate 20 random x values from a uniform distribution ranging between 0 and 100. Next, we'll create error terms by drawing from a normal distribution with a mean of 0 and a standard deviation of 2. Finally, we'll compute each y value using the formula y=3+4x+error. This approach integrates randomness in x and incorporates typical measurement errors through the normally distributed error term, reflecting realistic data generation conditions.

1.1

Fit a regression line to these data and display the output.

```
set.seed(123)
x_values <- runif(20, 0, 100)
intercept <- 3
slope <- 4
errors <- rnorm(20, mean = 0, sd = 2)
y_values <- intercept + slope * x_values + errors

data <- data.frame(x_values, y_values)

linear_model <- lm(y_values ~ x_values, data = data)
model_coefficients <- coef(linear_model)
slope_coeff <- model_coefficients[2]
intercept_coeff <- model_coefficients[1]

# Display the equation of the fitted regression line
cat("Regression line: y =", slope_coeff, "x +", intercept_coeff, "\n")
```

```
## Regression line: y = 3.973702 x + 4.232564
```

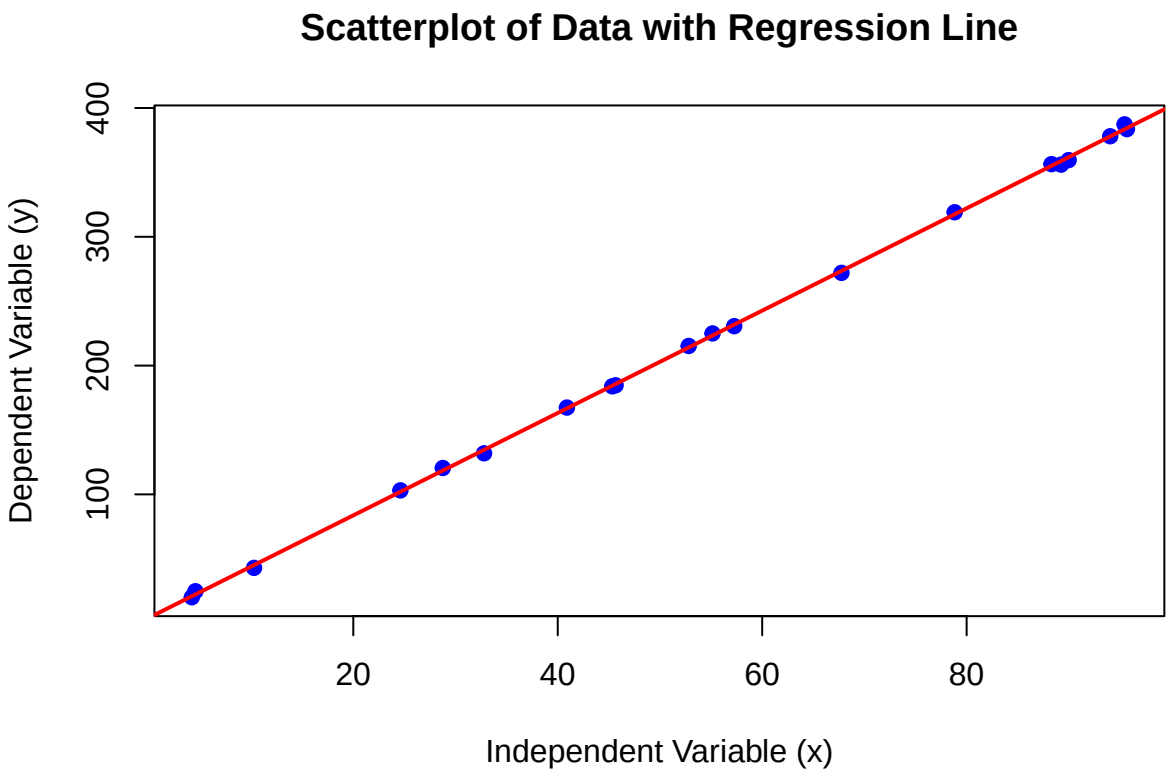
1.2

Graph a scatterplot of the data and the regression line.

```
# Fit the linear model to the data
fitted_model <- lm(y_values ~ x_values, data=data)

# Generate a scatterplot of the data points
plot(x_values, y_values, main="Scatterplot of Data with Regression Line", xlab="Independent Variable (x)

# Overlay the regression line on the scatterplot
abline(fitted_model, col="red", lwd=2)
```



1.3

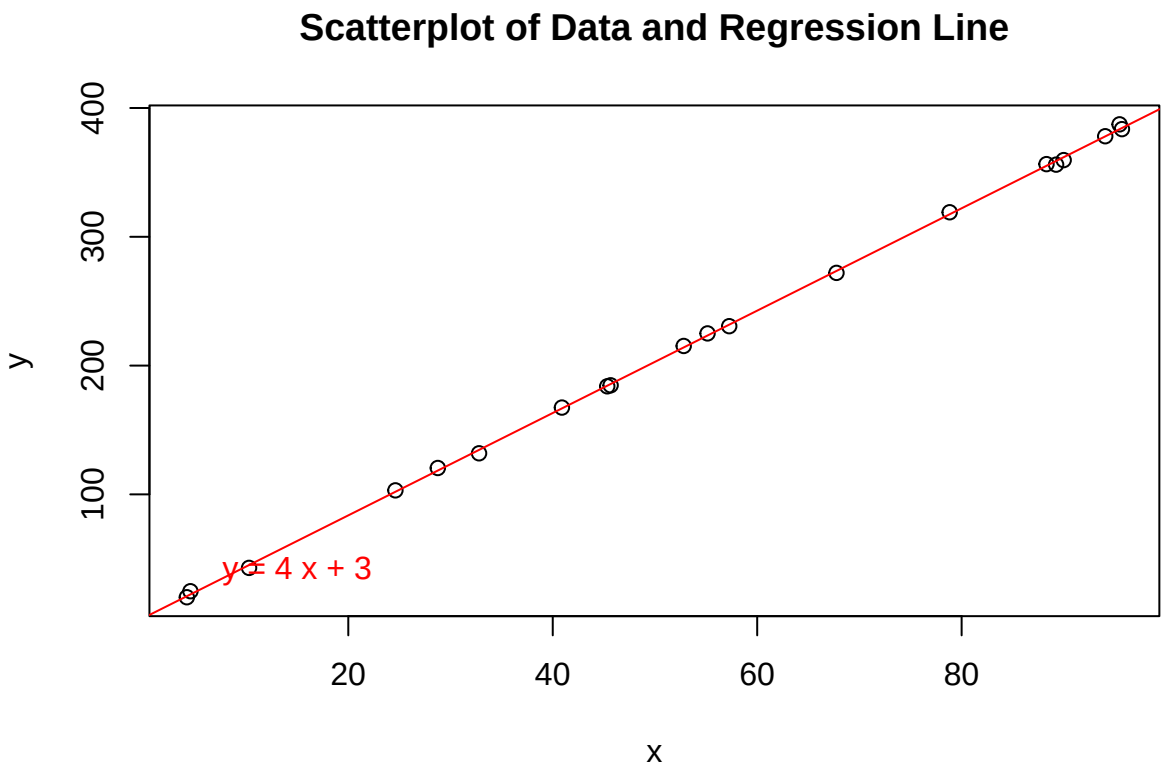
Use the `text` function in R to add the formula of the fitted line to the graph.

```
# Create a scatterplot of the data
plot(x_values, y_values, main="Scatterplot of Data and Regression Line", xlab="x", ylab="y")

# Add the regression line to the plot
abline(fitted_model, col="red")

# Create a text string for the regression line formula
line_formula <- paste("y =", slope, "x +", intercept)

# Use the text function to display the regression line's formula on the plot
text(15, 40, line_formula, col="red")
```



2 Fake-data simulation and fitting the wrong model:

To create 20 data points for a nonlinear model y=a+bx+cx^2 +error, start by defining coefficients a, b, and c that will visually emphasize the nonlinearity in a scatterplot. Next, generate x values uniformly distributed between 0 and 100. Then, produce error terms from a normal distribution with a mean of 0 and a standard deviation of 2. Finally, compute each y value by combining the linear, quadratic components, and the error term to reflect a realistic scatter of data points with a clear quadratic relationship.

2.1

Fit a regression line `stan_glm(y ~ x)` to these data and display the output.

```
set.seed(123)  # Ensure reproducibility

# Define the number of data points
num_points <- 20
x <- runif(num_points, 0, 100)
x_squared <- x^2  # Quadratic term
intercept <- 5
linear_coef <- 2
quadratic_coef <- 0.2
errors <- rnorm(num_points, 0, 3)  # Normal distributed errors
y <- intercept + linear_coef * x + quadratic_coef * x_squared + errors  # Construct y values
dataset <- data.frame(x, y)

# Fit a linear model to the quadratic data using Bayesian regression
fitted_model <- stan_glm(y ~ x + I(x^2), data=dataset, refresh=0)
```

```r
# Print the model output
print(fitted_model)
```

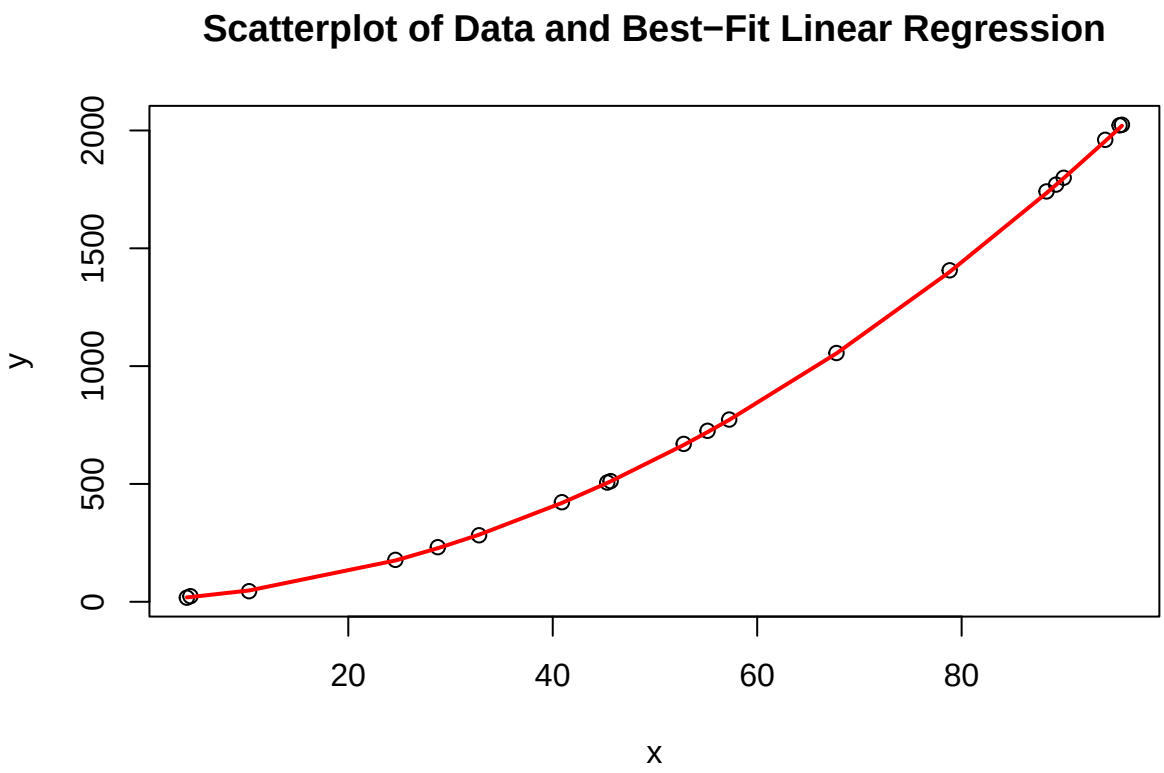
2.2

Graph a scatterplot of the data and the regression line. This is the best-fit linear regression. What does "best-fit" mean in this context?

```r
intercept <- 7.2
slope_x <- 1.9
slope_x2 <- 0.2

pred <- intercept + slope_x * x+ slope_x2 * x_squared

plot(x, y, main="Scatterplot of Data and Best-Fit Linear Regression", xlab="x",
     ylab="y")
ix <- sort(x,index.return=T)$ix
lines(x[ix], pred[ix], col='red', lwd=2)
```



The "best-fit" line represents the linear relationship that minimizes the sum of the squared differences between the observed data points and the predicted values along the line.