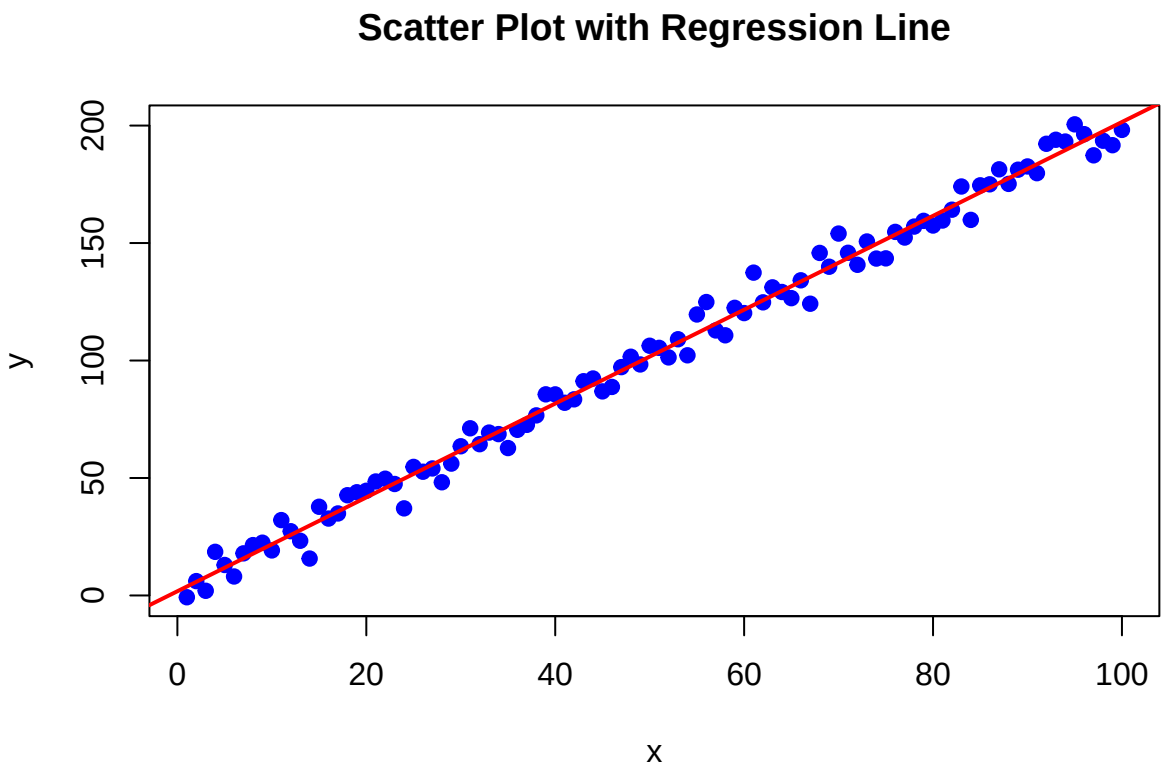
STAT3355(HW-8)

Tulasi Janjanam

2024-11-29

```r
# Step 1: Simulate the data
set.seed(1)
n <- 100 # Sample size
x <- seq(1, 100, by = 1)
beta0 <- 1 # Intercept
beta1 <- 2 # Slope
sigma <- 6 # Standard deviation of the error term
epsilon <- rnorm(n, mean = 0, sd = sigma) # Simulate errors
y <- beta0 + beta1 * x + epsilon

# Step 2: Create a scatter plot and fit a regression line
plot(x, y, main = "Scatter Plot with Regression Line",
     xlab = "x", ylab = "y", pch = 19, col = "blue")
model <- lm(y ~ x) # Fit the linear model
abline(model, col = "red", lwd = 2) # Add regression line
```




```r
# Step 3: Extract regression coefficients
summary(model)
```

```
##
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14.0403  -3.6350   0.0931   3.5109  13.7848
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.78999    1.09138    1.64    0.104
## x            1.99729    0.01876  106.45   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.416 on 98 degrees of freedom
## Multiple R-squared:  0.9914, Adjusted R-squared:  0.9913
## F-statistic: 1.133e+04 on 1 and 98 DF,  p-value: < 2.2e-16
```

```r
beta1_hat <- coef(model)["x"] # Estimated  1
se_beta1 <- summary(model)$coefficients["x", "Std. Error"] # Standard error of  1
t_stat <- (beta1_hat - 2) / se_beta1 # Test statistic
```

```r
p_value <- 2 * pt(-abs(t_stat), df = n - 2) # Two-sided p-value

# Step 5: Decision
significance_level <- 0.05
if (p_value < significance_level) {
  decision <- "Reject H0"
} else {
  decision <- "Fail to reject H0"
}

# Output Results
list(
  beta1_hat = beta1_hat,
  se_beta1 = se_beta1,
  t_stat = t_stat,
  p_value = p_value,
  decision = decision
)
```

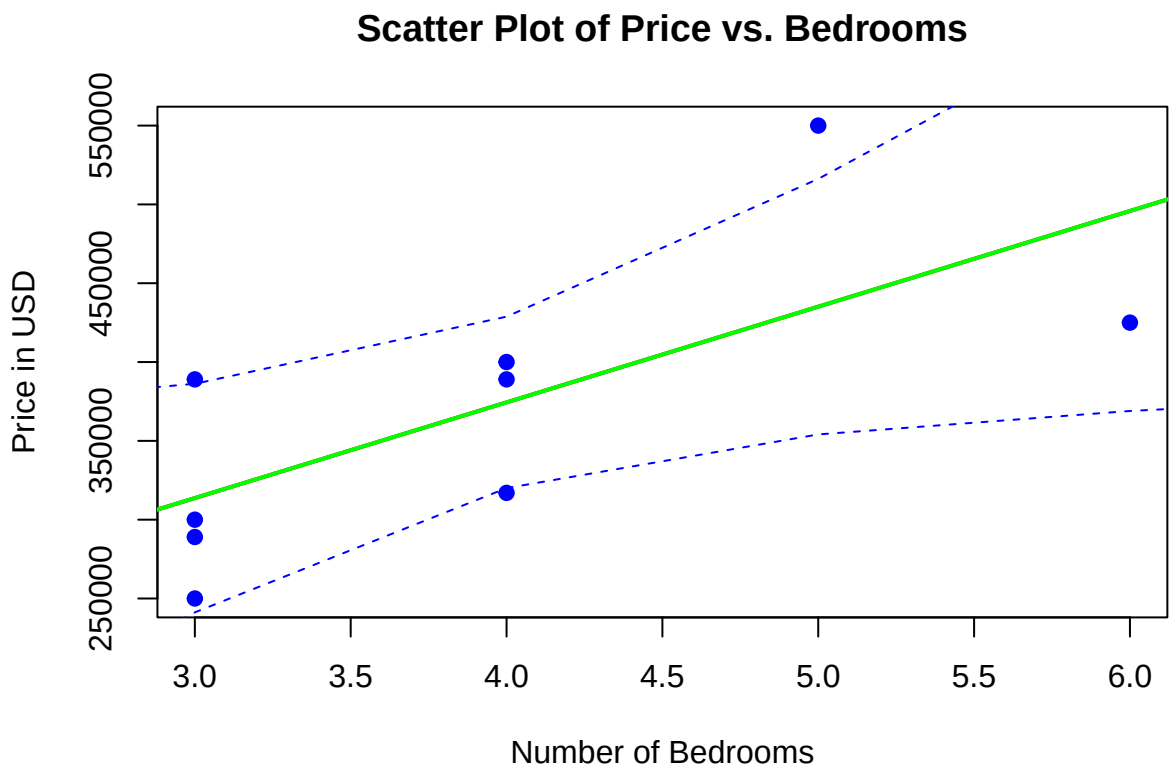
```
## $beta1_hat
##        x
## 1.997294
##
## $se_beta1
## [1] 0.01876268
##
## $t_stat
##          x
## -0.1442406
##
## $p_value
##         x
## 0.8856067
##
## $decision
## [1] "Fail to reject H0"
```

```r
# Step 1: Input the data
prices <- c(300000, 250000, 400000, 550000, 317000, 389000, 425000, 289000, 389000)
bedrooms <- c(3, 3, 4, 5, 4, 3, 6, 3, 4)

# Step 2: Create a scatter plot
plot(bedrooms, prices,
     main = "Scatter Plot of Price vs. Bedrooms",
     xlab = "Number of Bedrooms",
     ylab = "Price in USD",
     pch = 19, col = "blue")

# Step 3: Fit the data with a regression line
model <- lm(prices ~ bedrooms)
abline(model, col = "red", lwd = 2) # Add regression line

# Step 4: Confidence intervals for 2 to 8-bedroom houses
```

```r
library(dplyr) # For pipe operator
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
new_bedrooms <- data.frame(bedrooms = 2:8) # Generate range of bedrooms
conf_intervals <- predict(model, new_bedrooms, interval = "confidence") # Confidence intervals


lines(new_bedrooms$bedrooms, conf_intervals[, "fit"], col = "green", lwd = 2)
lines(new_bedrooms$bedrooms, conf_intervals[, "lwr"], col = "blue", lty = 2)
lines(new_bedrooms$bedrooms, conf_intervals[, "upr"], col = "blue", lty = 2)
```




```r
list(
  regression_summary = summary(model),
  confidence_intervals = data.frame(new_bedrooms, conf_intervals)
)
```

```
## $regression_summary
##
## Call:
## lm(formula = prices ~ bedrooms)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -70838 -57412 -13700  25588 114875
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   131562      92302   1.425   0.1971
## bedrooms       60712      22996   2.640   0.0334 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 68560 on 7 degrees of freedom
## Multiple R-squared:  0.4989, Adjusted R-squared:  0.4274
## F-statistic:  6.97 on 1 and 7 DF,  p-value: 0.03342
##
##
## $confidence_intervals
##   bedrooms      fit      lwr      upr
## 1        2 252987.5 136927.9 369047.1
## 2        3 313700.0 241198.2 386201.8
## 3        4 374412.5 320036.1 428788.9
## 4        5 435125.0 354065.5 516184.5
## 5        6 495837.5 368959.3 622715.7
## 6        7 556550.0 378957.5 734142.5
## 7        8 617262.5 387276.2 847248.8
```

```
# Step 1: Load the dataset
library(UsingR)
```

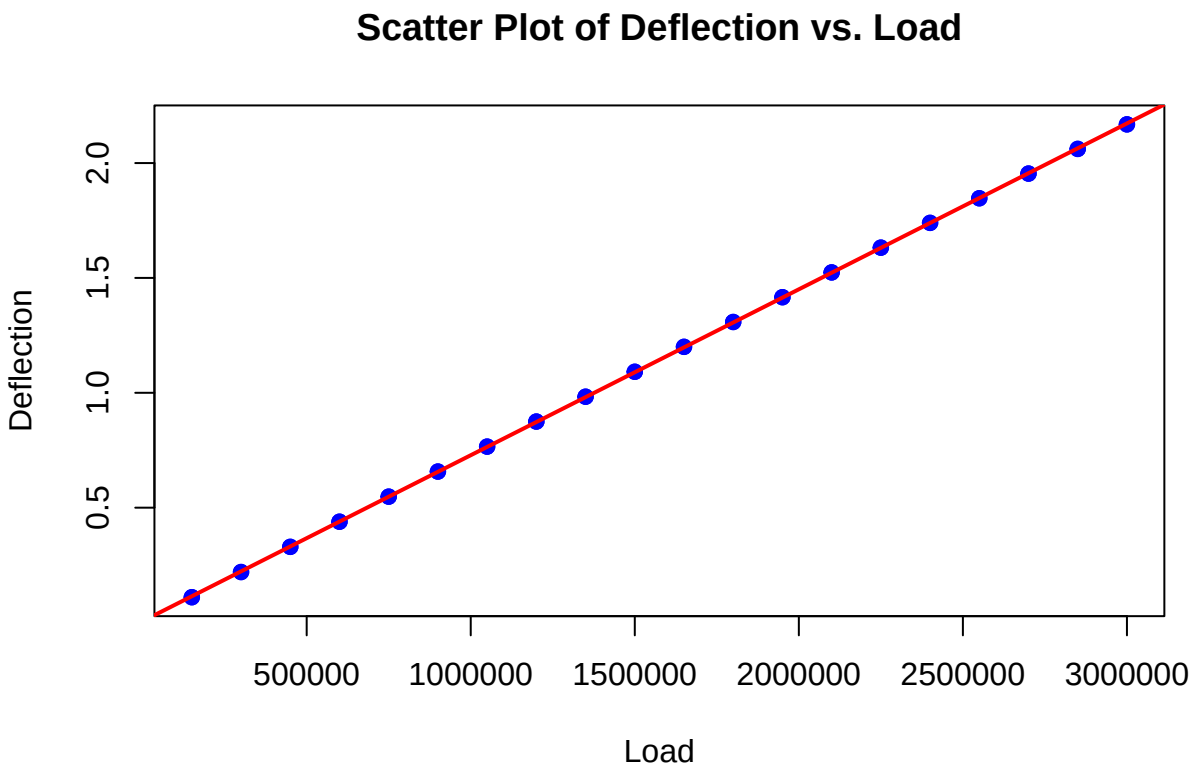
```
## Loading required package: MASS

##
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':
##
##     select

## Loading required package: HistData

## Loading required package: Hmisc

##
## Attaching package: 'Hmisc'

## The following objects are masked from 'package:dplyr':
##
##     src, summarize
```

```
## The following objects are masked from 'package:base':
##
##     format.pval, units
```

```r
data(deflection)

# Step 2: Inspect the data
head(deflection)
```

```
##   Deflection   Load
## 1    0.11019 150000
## 2    0.21956 300000
## 3    0.32949 450000
## 4    0.43899 600000
## 5    0.54803 750000
## 6    0.65694 900000
```

```r
# The dataset has two variables: Load and Deflection

# Step 3: Create a scatter plot
plot(deflection$Load, deflection$Deflection,
     main = "Scatter Plot of Deflection vs. Load",
     xlab = "Load", ylab = "Deflection",
     pch = 19, col = "blue")

# Step 4: Fit a linear model
model <- lm(Deflection ~ Load, data = deflection)
abline(model, col = "red", lwd = 2)
```




```r
# Step 5: Summary of the model
summary(model)
```

```
##
## Call:
## lm(formula = Deflection ~ Load, data = deflection)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -0.0042751 -0.0016308  0.0005818  0.0018932  0.0024211
##
## Coefficients:
##              Estimate Std. Error  t value Pr(>|t|)
## (Intercept) 6.150e-03  7.132e-04    8.623 1.77e-10 ***
## Load        7.221e-07  3.969e-10 1819.289  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.002171 on 38 degrees of freedom
## Multiple R-squared:      1,  Adjusted R-squared:      1
## F-statistic: 3.31e+06 on 1 and 38 DF,  p-value: < 2.2e-16
```

```r
# Step 6: Calculate 95% confidence intervals for 0 and 1
conf_intervals <- confint(model, level = 0.95)
```

```r
# Step 7: Display results
list(
  regression_summary = summary(model),
  confidence_intervals = conf_intervals
)
```

```
## $regression_summary
##
## Call:
## lm(formula = Deflection ~ Load, data = deflection)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -0.0042751 -0.0016308  0.0005818  0.0018932  0.0024211
##
## Coefficients:
##              Estimate Std. Error  t value Pr(>|t|)
## (Intercept) 6.150e-03  7.132e-04    8.623 1.77e-10 ***
## Load        7.221e-07  3.969e-10 1819.289  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.002171 on 38 degrees of freedom
## Multiple R-squared:      1,  Adjusted R-squared:      1
## F-statistic: 3.31e+06 on 1 and 38 DF,  p-value: < 2.2e-16
##
##
## $confidence_intervals
##                     2.5 %       97.5 %
## (Intercept) 4.705876e-03 7.593493e-03
## Load        7.212991e-07 7.229061e-07
```