CI0131 Diseño de experimentos - I Ciclo 2024

Laboratorio 6: Regresión Lineal Simple

Integrantes:

- Brandon Mora Umaña - C15179
- A. Badilla Olivas - B80874

Primera parte

Pregunta 1

Para mostrar la relación entre la variable `runs` y otra variable numérica, se utilizaría un **diagrama de dispersión**.

```
download.file("http://www.openintro.org/stat/data/mlb11.RData", destfile = "mlb11.RData")
load("mlb11.RData")
mlb11
```

```
##                       team runs at_bats hits homeruns bat_avg strikeouts
## 1            Texas Rangers  855    5659 1599      210   0.283        930
## 2          Boston Red Sox  875    5710 1600      203   0.280       1108
## 3          Detroit Tigers  787    5563 1540      169   0.277       1143
## 4      Kansas City Royals  730    5672 1560      129   0.275       1006
## 5     St. Louis Cardinals  762    5532 1513      162   0.273        978
## 6            New York Mets  718    5600 1477      108   0.264       1085
## 7        New York Yankees  867    5518 1452      222   0.263       1138
## 8       Milwaukee Brewers  721    5447 1422      185   0.261       1083
## 9        Colorado Rockies  735    5544 1429      163   0.258       1201
## 10         Houston Astros  615    5598 1442       95   0.258       1164
## 11     Baltimore Orioles  708    5585 1434      191   0.257       1120
## 12   Los Angeles Dodgers  644    5436 1395      117   0.257       1087
## 13           Chicago Cubs  654    5549 1423      148   0.256       1202
## 14       Cincinnati Reds  735    5612 1438      183   0.256       1250
## 15    Los Angeles Angels  667    5513 1394      155   0.253       1086
## 16 Philadelphia Phillies  713    5579 1409      153   0.253       1024
## 17     Chicago White Sox  654    5502 1387      154   0.252        989
## 18     Cleveland Indians  704    5509 1380      154   0.250       1269
## 19  Arizona Diamondbacks  731    5421 1357      172   0.250       1249
## 20     Toronto Blue Jays  743    5559 1384      186   0.249       1184
## 21       Minnesota Twins  619    5487 1357      103   0.247       1048
## 22       Florida Marlins  625    5508 1358      149   0.247       1244
## 23    Pittsburgh Pirates  610    5421 1325      107   0.244       1308
## 24     Oakland Athletics  645    5452 1330      114   0.244       1094
## 25         Tampa Bay Rays  707    5436 1324      172   0.244       1193
## 26        Atlanta Braves  641    5528 1345      173   0.243       1260
## 27  Washington Nationals  624    5441 1319      154   0.242       1323
## 28  San Francisco Giants  570    5486 1327      121   0.242       1122
## 29      San Diego Padres  593    5417 1284       91   0.237       1320
## 30      Seattle Mariners  556    5421 1263      109   0.233       1280
##    stolen_bases wins new_onbase new_slug new_obs
## 1           143   96      0.340    0.460   0.800
## 2           102   90      0.349    0.461   0.810
## 3            49   95      0.340    0.434   0.773
## 4           153   71      0.329    0.415   0.744
## 5            57   90      0.341    0.425   0.766
```

```
## 6           130   77      0.335    0.391   0.725
## 7           147   97      0.343    0.444   0.788
## 8            94   96      0.325    0.425   0.750
## 9           118   73      0.329    0.410   0.739
## 10          118   56      0.311    0.374   0.684
## 11           81   69      0.316    0.413   0.729
## 12          126   82      0.322    0.375   0.697
## 13           69   71      0.314    0.401   0.715
## 14           97   79      0.326    0.408   0.734
## 15          135   86      0.313    0.402   0.714
## 16           96  102      0.323    0.395   0.717
## 17           81   79      0.319    0.388   0.706
## 18           89   80      0.317    0.396   0.714
## 19          133   94      0.322    0.413   0.736
## 20          131   81      0.317    0.413   0.730
## 21           92   63      0.306    0.360   0.666
## 22           95   72      0.318    0.388   0.706
## 23          108   72      0.309    0.368   0.676
## 24          117   74      0.311    0.369   0.680
## 25          155   91      0.322    0.402   0.724
## 26           77   89      0.308    0.387   0.695
## 27          106   80      0.309    0.383   0.691
## 28           85   86      0.303    0.368   0.671
## 29          170   71      0.305    0.349   0.653
## 30          125   67      0.292    0.348   0.640
```

```r
plot(mlb11$at_bats, mlb11$runs, xlab = "Turnos al Bate (at_bats)", ylab = "Carreras Anotadas (runs)")
```

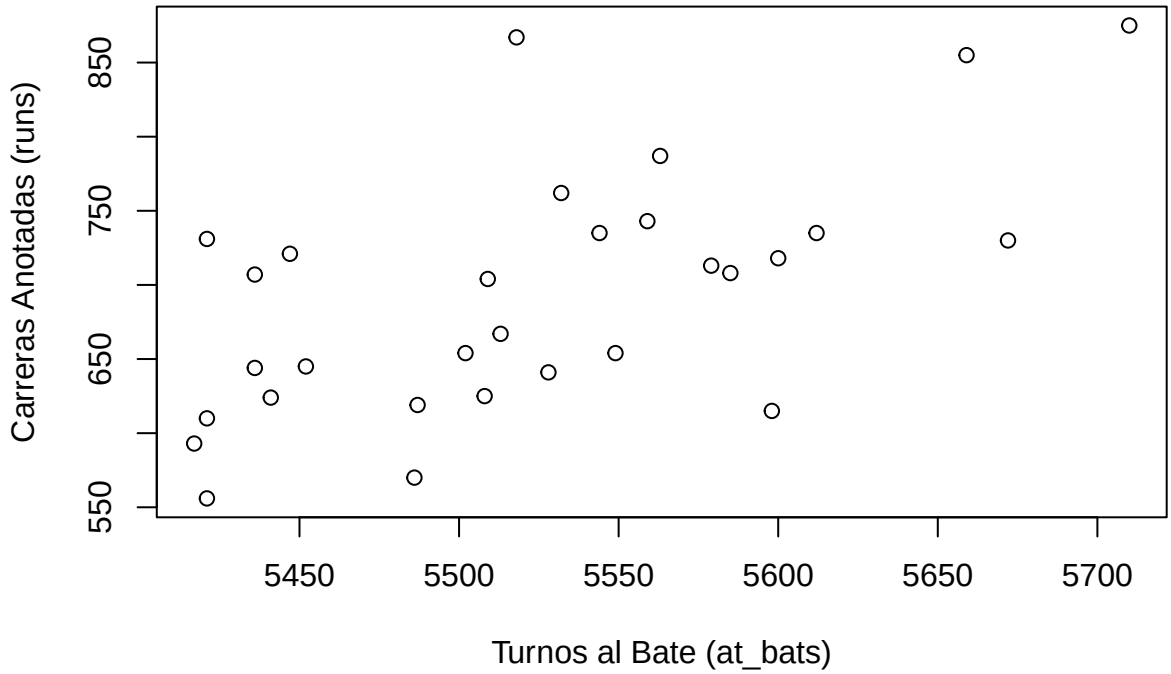

Figure 1: Relación entre Carreras Anotadas (runs) y Turnos al Bate (at_bats)

La relación entre `runs` y `at_bats` parece ser **lineal y positiva**, aunque con cierta dispersión. A medida que aumentan los turnos al bate, tienden a aumentar las carreras anotadas.

El coeficiente de correlación entre `runs` y `at_bats` es:

```
cor(mlb11$runs, mlb11$at_bats)
```

```
## [1] 0.610627
```

Esto confirma una **correlación positiva moderadamente fuerte** entre las variables.

Pregunta 2

Escribimos las siguientes funciones para gráficar además de la linea los residuales de manera más bonita.

```
# Function to plot data points and optionally show squares of residuals
plot_ss <- function(x, y, showSquares = FALSE, predefined=FALSE) {
  # Plot the initial scatter plot
  plot(x, y, xlab = "", ylab = "")
  cat("Click twice on the graph to draw a line.\n")
  # Get two points from the user to define the line or use predefined points
  if (predefined)
    user_points <- data.frame(x=c(min(x), max(x)), y=c(min(y), max(y)))
  else
    user_points <- getUserDefinedLine()
  # Calculate slope and intercept based on user points
  line_params <- calculateLineParams(user_points)

  # Draw the line on the plot
  drawLine(line_params$intercept, line_params$slope)

  # Optionally, show squares of residuals
  if(showSquares) {
    showResidualSquares(x, y, line_params$slope, line_params$intercept)
  }

  # Calculate and display the sum of squares of residuals
  ss_res <- calculateSSRes(x, y, line_params$slope, line_params$intercept)
  cat("Sum of squares of residuals:", ss_res, "\n")
  cat("Intercept:", line_params$intercept, "Slope:", line_params$slope, "\n")
}
# Function to get two points from the user to define the line
getUserDefinedLine <- function() {
  pts <- locator(n = 2)
  return(pts)
}
# Function to calculate slope and intercept from two points
calculateLineParams <- function(pts) {
  slope <- (pts$y[2] - pts$y[1]) / (pts$x[2] - pts$x[1])
  intercept <- pts$y[1] - slope * pts$x[1]
  return(list(slope = slope, intercept = intercept))
}
# Function to draw a line given slope and intercept
drawLine <- function(intercept, slope) {
  abline(a = intercept, b = slope)
}
# Function to show squares of residuals
showResidualSquares <- function(x, y, slope, intercept) {
  # Here we plot the actual data
```

```
  points(x = x, y = y, pch = 16, cex = 1.2)
  # here we plot the points sample frome the line we created
  points(x = x, y = intercept + slope * x, col = "red", pch = 16, cex = 1.2)
  # here we plot line segments from the line  and the actual data.
  segments(x0 = x, y0 = y, x1 = x, y1 = intercept + slope * x, col = "blue", lty = 2)
}
# Function to calculate the sum of squares of residuals
calculateSSRes <- function(x, y, slope, intercept) {
  y_pred <- intercept + slope * x
  ss_res <- sum((y - y_pred)^2)
  return(ss_res)
}

plot_ss(x = mlb11$at_bats, y = mlb11$runs, showSquares = TRUE, predefined=TRUE)
```

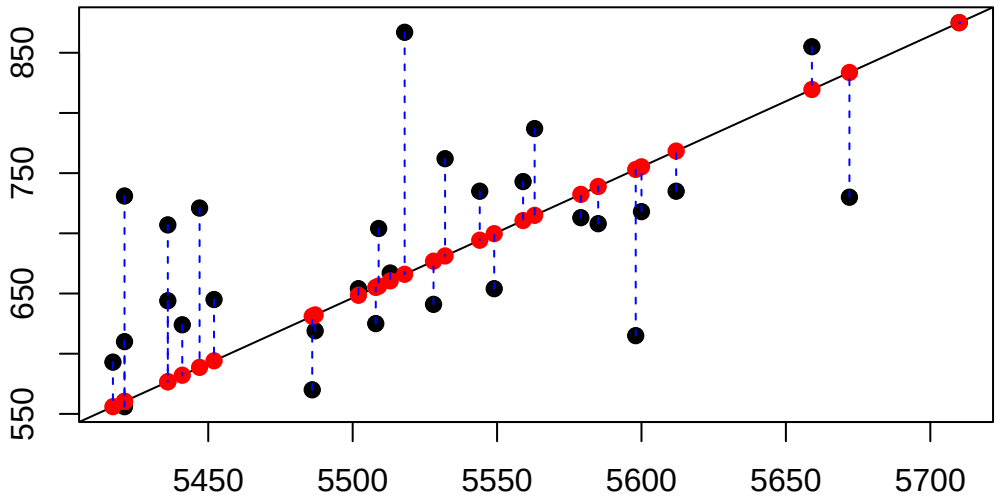

Figure 2: Relación entre Carreras Anotadas (runs) y Turnos al Bate (at_bats) con Residuos Cuadrados

```
## Click twice on the graph to draw a line.
## Sum of squares of residuals: 176623.4
## Intercept: -5341.689 Slope: 1.088737
```

La menor suma de cuadrados que se logró obtener fue **144425.8**. Como se mencionó, es difícil igualar la suma de cuadrados del modelo lineal (123721.9) de forma manual.

Pregunta 3

```
m1 <- lm(runs ~ at_bats, data = mlb11)
summary(m1)
```

```
##
## Call:
## lm(formula = runs ~ at_bats, data = mlb11)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -125.58  -47.05  -16.59   54.40  176.87
##
## Coefficients:
```

```
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2789.2429   853.6957  -3.267 0.002871 **
## at_bats         0.6305     0.1545   4.080 0.000339 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 66.47 on 28 degrees of freedom
## Multiple R-squared:  0.3729, Adjusted R-squared:  0.3505
## F-statistic: 16.65 on 1 and 28 DF,  p-value: 0.0003388
```

Pregunta 4

La fórmula del modelo lineal obtenido es:

$\hat{y}$ = -2789.2429 + 0.6305 * at_bats

Pregunta 5

```r
m2 <- lm(runs ~ homeruns, data = mlb11)
summary(m2)
```

```
##
## Call:
## lm(formula = runs ~ homeruns, data = mlb11)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -91.615 -33.410   3.231  24.292 104.631
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 415.2389    41.6779   9.963 1.04e-10 ***
## homeruns      1.8345     0.2677   6.854 1.90e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 51.29 on 28 degrees of freedom
## Multiple R-squared:  0.6266, Adjusted R-squared:  0.6132
## F-statistic: 46.98 on 1 and 28 DF,  p-value: 1.9e-07
```

La ecuación del modelo lineal para `runs` en función de `homeruns` es:

$\hat{y}$ = 415.2389 + 1.8345 * homeruns

- **Pendiente:** La pendiente de 1.8345 indica que, por cada homerun adicional que un equipo realiza, se espera que anote 1.83 carreras más, manteniendo constantes las demás variables.
- **R-cuadrado:** El R-cuadrado de este modelo (0.6266) es mayor que el del modelo con `at_bats` (0.3729). Esto significa que el 62.69% de la variabilidad en las carreras anotadas se explica por la cantidad de homeruns, lo que indica un mejor ajuste que el modelo anterior.

Pregunta 6

```r
plot(mlb11$runs ~ mlb11$at_bats, xlab = "Turnos al Bate (at_bats)", ylab = "Carreras Anotadas (runs)")
abline(m1)
```

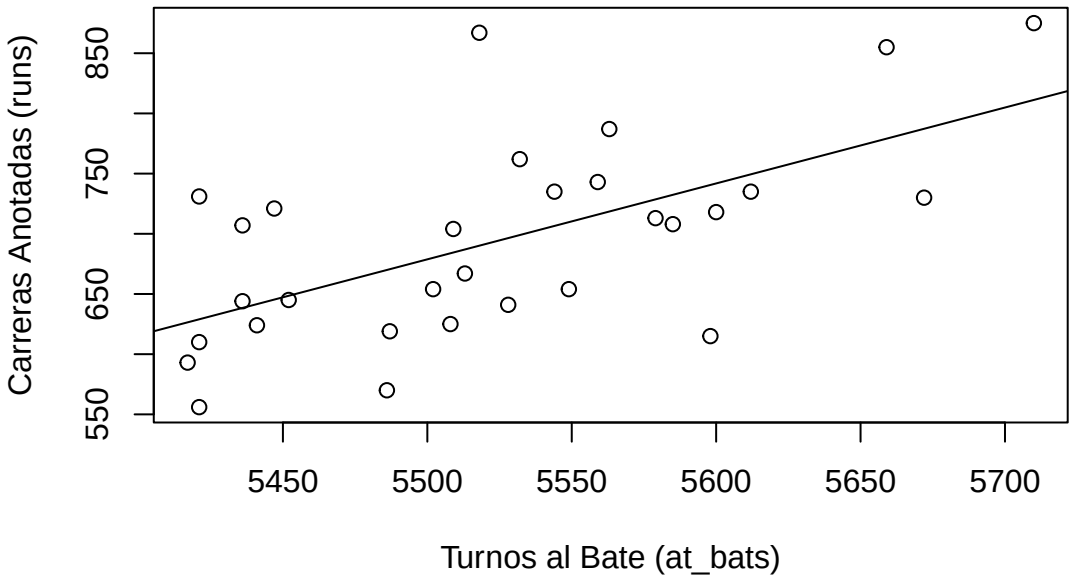

Figure 3: Diagrama de Dispersión con Línea de Mínimos Cuadrados

Pregunta 7

```
plot(m1$residuals ~ mlb11$at_bats, xlab = "Turnos al Bate (at_bats)", ylab = "Residuos")
abline(h = 0, lty = 3)
```

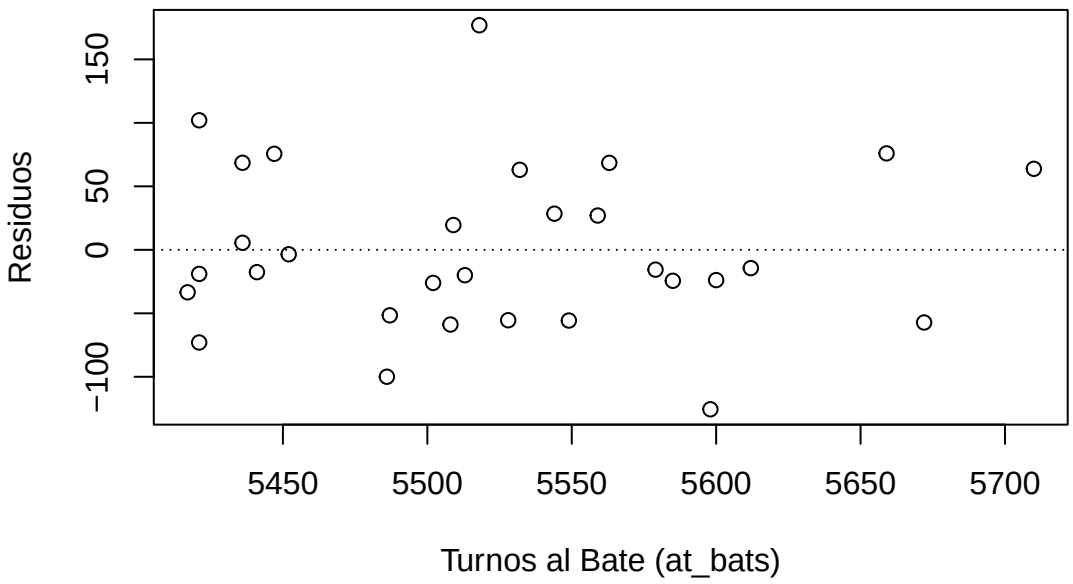

Figure 4: Gráfico de Residuos vs. Turnos al Bate

No parece haber un patrón aparente en el gráfico de residuos. Los residuos parecen estar distribuidos aleatoriamente alrededor de cero, lo que sugiere que **la condición de independencia de residuos se cumple**.

Pregunta 8

```
hist(m1$residuals)
```

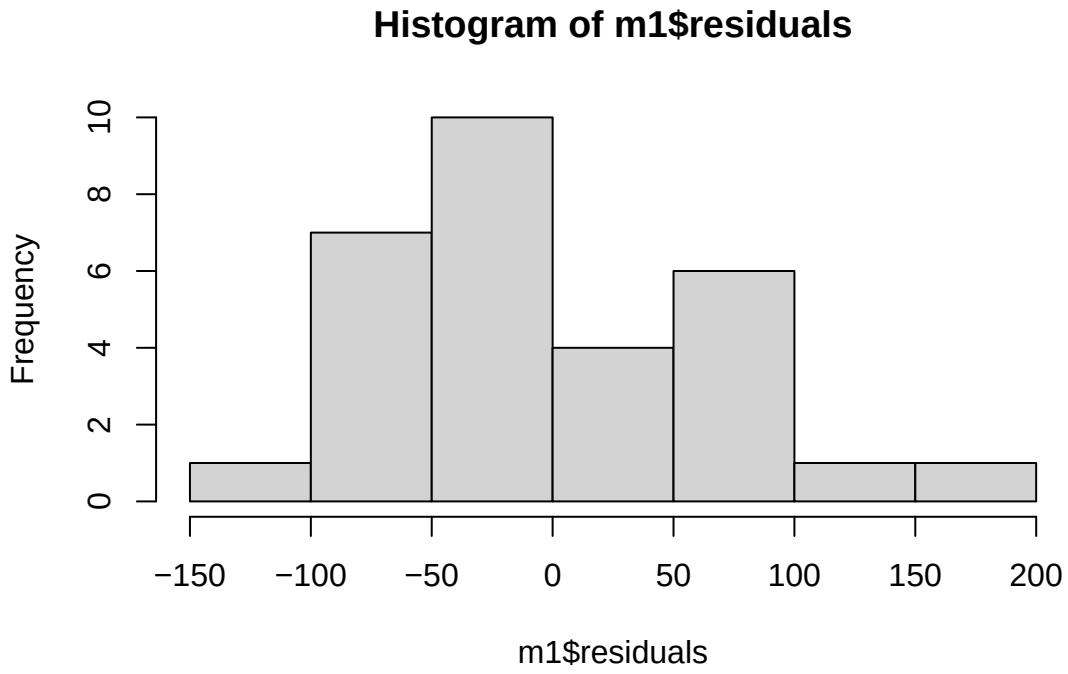

Figure 5: Histograma de Residuos

```
qqnorm(m1$residuals)
qqline(m1$residuals)
```

Tanto el histograma como la gráfica de probabilidad normal sugieren que **la condición de residuos normales se cumple razonablemente**. El histograma muestra una forma aproximadamente simétrica y acampanada, aunque una de las barras en el centro parace desviarse. También, la mayoría de los puntos en la gráfica de probabilidad normal se encuentran cerca de la línea diagonal.

Pregunta 9

```
plot(m1$residuals ~ m1$fitted.values, xlab = "Valores Ajustados", ylab = "Residuos")
abline(h = 0, lty = 3)
```

El gráfico de residuos vs. valores ajustados **no muestra un patrón evidente**, lo que sugiere que **la condición de varianza homogénea de los residuos se cumple**.

Segunda parte

Pregunta 10

```
# Crear una tabla vacía para almacenar los resultados
tabla_resultados <- data.frame(
  Variable = character(),
  Correlacion = numeric(),
  Valor_p = numeric(),
  R_cuadrado = numeric(),
  stringsAsFactors = FALSE
```

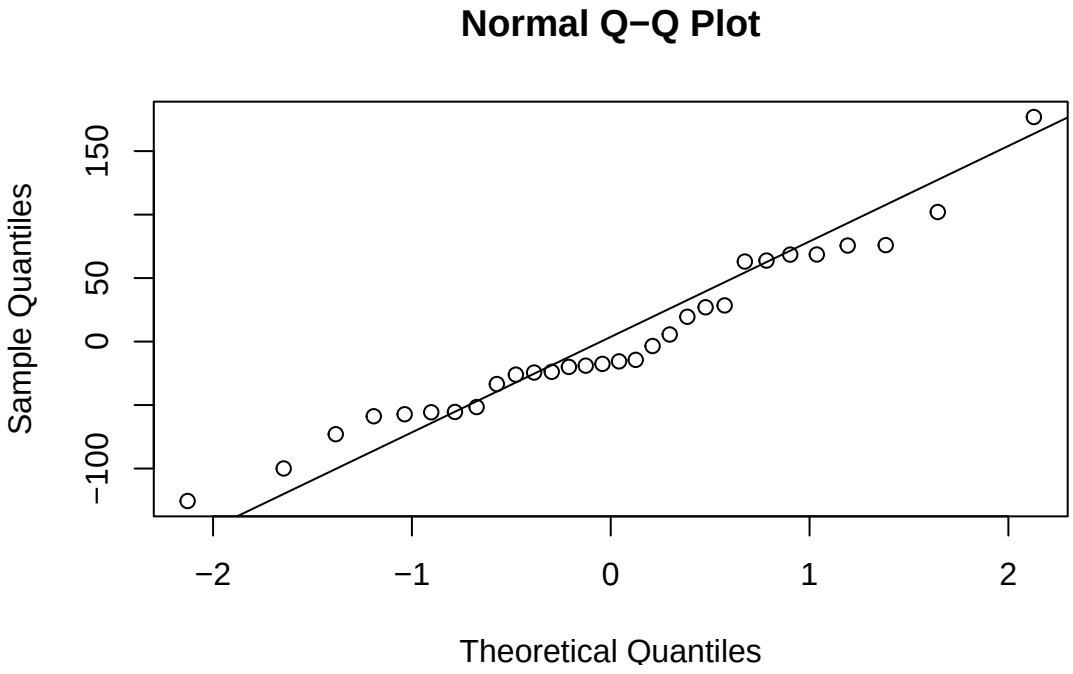

Figure 6: Gráfica de Probabilidad Normal de Residuos

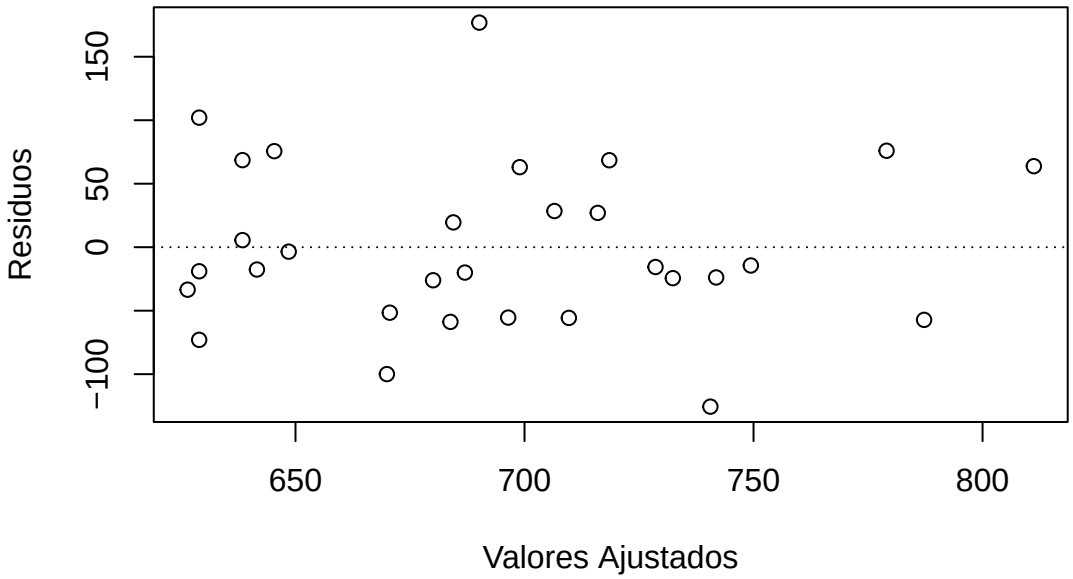

Figure 7: Gráfico de Residuos vs. Valores Ajustados

```
)

# Iterar sobre las variables tradicionales
for (variable in c("at_bats", "hits", "homeruns", "bat_avg", "strikeouts", "stolen_bases", "wins")) {
  # Ajustar el modelo lineal
  formula <- as.formula(paste("runs ~", variable))
  modelo <- lm(formula, data = mlb11)

  # Extraer los resultados
  correlacion <- cor(mlb11$runs, mlb11[[variable]])
  valor_p <- summary(modelo)$coefficients[2, 4]
  r_cuadrado <- summary(modelo)$r.squared

  # Agregar los resultados a la tabla
  tabla_resultados <- rbind(tabla_resultados, data.frame(
    Variable = variable,
    Correlacion = correlacion,
    Valor_p = valor_p,
    R_cuadrado = r_cuadrado
  ))
}

# Mostrar la tabla
tabla_resultados
```

```
##       Variable Correlacion      Valor_p  R_cuadrado
## 1      at_bats  0.61062705 3.388351e-04 0.372865390
## 2         hits  0.80121081 1.043247e-07 0.641938767
## 3     homeruns  0.79155769 1.900086e-07 0.626563570
## 4      bat_avg  0.80998589 5.877038e-08 0.656077135
## 5   strikeouts -0.41153120 2.385596e-02 0.169357932
## 6 stolen_bases  0.05398141 7.769381e-01 0.002913993
## 7         wins  0.60080877 4.469271e-04 0.360971179
```

De acuerdo con los resultados de la tabla, la variable **`bat_avg` (promedio de bateo) parece ser el mejor predictor de `runs`** entre las variables tradicionales. Esto se debe a que presenta el **mayor valor de R-cuadrado (0.6561)**, lo que indica que explica la mayor proporción de la variabilidad en las carreras anotadas. Además, tiene un **valor p muy pequeño**, lo que indica una relación estadísticamente significativa. Por último, también es la que posee la mayor correlación con la varaiable **`runs`**.

Pregunta 11

```
# Crear una tabla vacía para almacenar los resultados
tabla_resultados_nuevas <- data.frame(
  Variable = character(),
  Correlacion = numeric(),
  Valor_p = numeric(),
  R_cuadrado = numeric(),
  stringsAsFactors = FALSE
)

# Iterar sobre las nuevas variables
for (variable in c("new_onbase", "new_slug", "new_obs")) {
  # Ajustar el modelo lineal
```

```
  formula <- as.formula(paste("runs ~", variable))
  modelo <- lm(formula, data = mlb11)

  # Extraer los resultados
  correlacion <- cor(mlb11$runs, mlb11[[variable]])
  valor_p <- summary(modelo)$coefficients[2, 4]
  r_cuadrado <- summary(modelo)$r.squared

  # Agregar los resultados a la tabla
  tabla_resultados_nuevas <- rbind(tabla_resultados_nuevas, data.frame(
    Variable = variable,
    Correlacion = correlacion,
    Valor_p = valor_p,
    R_cuadrado = r_cuadrado
  ))
}

# Mostrar la tabla
tabla_resultados_nuevas
```

```
##     Variable Correlacion      Valor_p R_cuadrado
## 1 new_onbase   0.9214691 5.115676e-13  0.8491053
## 2   new_slug   0.9470324 2.420125e-15  0.8968704
## 3    new_obs   0.9669163 3.764375e-18  0.9349271
```

En general, las **nuevas variables (`new_onbase`, `new_slug`, `new_obs`) son más efectivas para predecir `runs`** que las variables tradicionales. Esto se evidencia en los **valores de R-cuadrado más altos** para las tres nuevas variables en comparación con las variables tradicionales.

El **mejor predictor de `runs` parece ser `new_obs` (suma de porcentaje de embasamiento y porcentaje de slugging)**. Esta variable presenta el **R-cuadrado más alto (0.9349)**, lo que significa que explica una gran proporción de la variabilidad en las carreras anotadas. New obs esta definida como la suma de porcentaje de embasamiento y porcentaje de slugging, ambos con porcentajes de correlación muy altos indicando que sean variables muy significativas para el modelo.

Estos resultados apoyan la premisa de Moneyball, que sugiere que estadísticas como el porcentaje de embasamiento y el porcentaje de slugging son mejores predictores del éxito de un equipo que las estadísticas tradicionales.