Soluciones Taller de problemas GRUPO inferencia 2023 MAT3 GIN

Marc Link, Carlos Gálvez, Pau Toni Bibiloni, Jesús Castillo

Contenidos

1 Taller Problemas evaluable 22-23: Estadística Inferencial

Valor 14 puntos. Todos los apartados valen 1 punto.

Se trata de resolver los siguientes problemas y cuestiones en un fichero Rmd y su salida en un informe en html, word o pdf.

1.1 Problema 1: Regresión lineal simple. 7 puntos.

Consideremos los siguientes datos

```
x=c(-2,-1,2,0,1,2)
y=c(-7, -5,  5, -3,  3.0,  4)
```

1. Calcular manualmente haciendo una tabla los coeficiente de la regresión lineal de y sobre x.

Para poder calcular los coeficientes de la regresión lineal necesitaremos el número de elementos, los valores de X, los valores de Y, los valores de $X \cdot Y$ y los valores de X^2. Además de sus correspondientes sumas.

```
# El número de elementos
n = length(x)

# La suma de los valores de x
sum_x = sum(x)

# La suma de los valores de y
sum_y = sum(y)

# La suma de los valores de x*y
sum_xy = sum(x*y)

# La suma de los valores de x^2
sum_x2 = sum(x^2)
```

Una vez obtenidos, podremos calcular los coeficientes b_0 y b_1.

```
b1 = (n * sum_xy - sum_x * sum_y) / (n * sum_x2 - sum_x^2)
b0 = (sum_y - b1 * sum_x) / n

cat("Coeficiente b0:", b0, "\n")
```

```
## Coeficiente b0: -1.525
```

```
cat("Coeficiente b1:", b1, "\n")
```

```
## Coeficiente b1: 3.075
```

2. Calcular los valores $\hat{y}_i = b_0 + b_1 \cdot x_1$ para los valores de la muestra y el error cometido.

Sabemos que $\varepsilon_i = y_i - \hat{y}_i$, por lo que simplemente calculamos los valores para $\hat{y}_i$ y podremos calcular el error ε_i.

```
y_gorro = b1*x + b0

errs = y - y_gorro

cat("Valores y_gorro", y_gorro, "\n")
```

```
## Valores y_gorro -7.675 -4.6 4.625 -1.525 1.55 4.625
```

```
cat("Errores:", errs, "\n")
```

```
## Errores: 0.675 -0.4 0.375 -1.475 1.45 -0.625
```

3. Calcular la estimación de la varianza del error.

Sabemos que la estimación de la varianza del error es $\frac{\sum \varepsilon_i^2}{n-2}$.

```
s2 = sum(errs^2) / (n - 2)

cat("Estimación de la varianza del error (s^2):", s2, "\n")
```

```
## Estimación de la varianza del error (s^2): 1.35625
```

4. Resolver manualmente el contraste $\begin{cases} H_0: & \beta_1 = 0 \\ H_1: & \beta_1 \neq 0 \end{cases}$, calculando el p-valor.

Dadas estas hipótesis, para contrastarlas, primero deberemos calcular el valor del estadístico de contraste t y así obtener el $p - valor$. El estadístico de contraste se calcula de la siguiente manera: $t = \frac{b_1}{\sqrt{\frac{s^2}{\sum_{i=1}^{n}(x_i - \bar{x})^2}}}$.
Una vez calculado el estadístico podemos calcular el $p - valor$ de la siguiente manera: $p = 2 \cdot P(T > t_s)$. Utilizando leyes de t de Student con $n - 2$ grados de libertad.

```r
mean_x = sum_x/n
S = sqrt(sum(errs^2)/(n - 2))
# Calculamos el estadístico
t = b1 / (S/(sd(x) * sqrt(n - 1)))

# Calculamos el p-valor
p = 2 * pt(abs(t), n - 2, lower.tail = FALSE)

cat("El valor del estadístico t es", t, "\n")
```

```
## El valor del estadístico t es 9.6415
```

```r
cat("El p-valor vale", p, "\n")
```

```
## El p-valor vale 0.0006472191
```

Dado que el $p - valor$ ha resultado ser menor que 0.05 tenemos evidencia suficiente como para rechazar la hipótesis nula. Gracias a que rechazamos dicha hipótesis podemos decir que existe una relación entre Y y X y que, cuando la segunda cambia, la primera también lo hace.

5. Calcular SST, SSR y SSE.

Calcularemos dichas variables de la siguiente manera:

```r
media_y = sum_y / n

SST = sum((y - media_y)^2)
SSR = sum((y_gorro - media_y)^2)
SSE = sum(errs^2)

cat("SST =", SST, "\n")
```

```
## SST = 131.5
```

```r
cat("SSR =", SSR, "\n")
```

```
## SSR = 126.075
```

```r
cat("SSE =", SSE, "\n")
```

```
## SSE = 5.425
```

Además podemos comprobar que los resultados son corretos mediante la siguiente fórmula $SS_T = SS_R + SS_E$. Calculando obtenemos que:

```r
SSR + SSE
```

```
## [1] 131.5
```

```
SST
```

```
## [1] 131.5
```

Por lo que podemos afirmar que los cálculos son correctos.

6. Calcular el coeficiente de regresión lineal r_{xy} y el coeficiente de determinación R^2. Interpretad el resultado en términos de la cantidad de varianza explicada por el modelo.

Sabemos que $R^2 = \frac{SS_R}{SS_T}$ y que $R^2 = r_{xy}^2$. Así que calculando obtenemos:

```
R2 = SSR / SST
rxy = sqrt(R2)

cat("El coeficiente de determinación (R^2) es:", R2, "\n")
```

```
## El coeficiente de determinación (R^2) es: 0.9587452
```

```
cat("El coeficiente de regresión lineal (r_xy) es", rxy, "\n")
```

```
## El coeficiente de regresión lineal (r_xy) es 0.9791554
```

Vemos que el valor de R^2 es cercano a 1 por lo que podemos decir que nuestra recta de regresión se ajusta bastante bien a nuestros datos.

7. Comprobar que los resultados son los mismos que los obtenidos con la función `summary(lm(y~x))`.

Dados los resultados obtenidos podemos comprobar su veracidad comparándolos con los resultados que nos ofrece la función de R. Ejecutando la función obtenemos los siguiente:

```
linear = summary(lm(y~x))
linear
```

```
##
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##      1      2      3      4      5      6
##  0.675 -0.400  0.375 -1.475  1.450 -0.625
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.5250     0.4872  -3.130 0.035176 *
## x             3.0750     0.3189   9.642 0.000647 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.165 on 4 degrees of freedom
## Multiple R-squared:  0.9587, Adjusted R-squared:  0.9484
## F-statistic: 92.96 on 1 and 4 DF,  p-value: 0.0006472
```

Cuyos coeficientes y R^2 son idénticos a los obtenidos por nuestros cálculos.

```
cat("Coeficientes:\n\tPunto de corte:", b0, "\n\tPendiente:", b1, "\n")
```

```
## Coeficientes:
##  Punto de corte: -1.525
##  Pendiente: 3.075
```

```
cat("R^2 =", R2)
```

```
## R^2 = 0.9587452
```

1.2 Problema 2: Distribución de los grados de un grafo de contactos. 3 puntos

The marvel chronology project es una web que ha recopilado las apariciones de los personajes Marvel en cada uno de los cómics que se van publicando.

En el artículo Marvel Universe looks almost like a real social network se estudió la red de contactos de los personajes del Universo Marvel de la serie de cómics books. Dos personajes tienen relación si han participado en al menos un mismo cómic; a semejanza del Oracle of Bacon donde se relacionan los actores de las películas de Hollywood que han participado en al menos una película juntos.

Si construimos el grafo de asociado a esas relaciones el grado de cada carácter (personaje) será el número de ortos caracteres (personajes) con los que ha colaborado. Cuando más importante es el personaje más colaboraciones tiene.

Los grados de cada caracteres están en el fichero `datasets/degree_Marvel_characters.cvs`. Según algunos estudios la distribución de los grados de los grafos de contactos sigue una ley potencial frecuencia grado $k = \beta_0 \cdot grado^{\beta} 1$ si eliminamos los 20 más pequeños.

```
data=read_csv("datasets/degree_Marvel_characters.csv")
```

Se pide:

1. Cargad los datos. Calcular las frecuencias de los grados, es decir el número de caracteres que tienen 1, 2 ,3 .... colaboradores para cada grado (número de colaboraciones) observado.

2. Ajustar un modelo lineal, potencial y exponencial a la relación entre $y =$ "frecuencia del grado" y $x = grado$ dibujar las gráficas de ajuste de cada modelo con gráficos semi-log y log-log si es necesario.
3. Para el mejor modelo calcular los coeficientes en las unidades originales y escribir la ecuación del modelos.

1.2.1 Solucion 2

Apartado 1

Cargamos la librería tidyverse y calculamos las frecuencias de los grados en el conjunto de datos.

```
# Cargar librerías
library(tidyverse)

# Calcular las frecuencias de los grados
frecuencias <- table(data$degree_Marvel_characters)
print(frecuencias)
```

```
##
##    2    4    6    8   10   12   14   16   18   20   22   24   26   28   30   32
##   45   60  144  202  311  350  427  442  530  514  524  455  429  441  414  411
##   34   36   38   40   42   44   46   48   50   52   54   56   58   60   62   64
##  380  361  344  367  285  247  231  232  230  193  176  179  189  176  174  149
##   66   68   70   72   74   76   78   80   82   84   86   88   90   92   94   96
##  145  118  115  131  145  109   97   88   85  103   91   76   78   66   65   59
##   98  100  102  104  106  108  110  112  114  116  118  120  122  124  126  128
##   90   55   66   80   49   53   51   57   41   55   49   49   43   38   41   38
##  130  132  134  136  138  140  142  144  146  148  150  152  154  156  158  160
##   57   43   65   54   21   40   41   28   43   45   40   34   33   32   33   24
##  162  164  166  168  170  172  174  176  178  180  182  184  186  188  190  192
##   32   22   28   26   13   38   29   29   22   32   27   21   21   24   21   17
##  194  196  198  200  202  204  206  208  210  212  214  216  218  220  222  224
##   17   20   25   19   16   21   11   39   18   16   18   13   17   26   14   14
##  226  228  230  232  234  236  238  240  242  244  246  248  250  252  254  256
##   10   10   18   20   12   13   22   22   13   26   18   12   10   14   11   15
##  258  260  262  264  266  268  270  272  274  276  278  280  282  284  286  288
##   11   15    9   10    6   13    6    5    8   12    8    7   18   11   11    9
##  290  292  294  296  298  300  302  304  306  308  310  312  314  316  318  320
##    8   16    7   11   13    6   11    7    6    6    7   14   10    7   10   12
##  322  324  326  328  330  332  334  336  338  340  342  344  346  348  350  352
##    7   10   10    5   11    3   10   19    8    8    7   10   11    8    5    4
##  354  356  358  360  362  364  366  368  370  372  374  376  378  380  382  384
##   12    1    8   11    5    8    6    3    7    3    5    6    5    9    6    8
##  386  388  390  392  394  396  398  400  402  404  406  408  410  412  414  416
##    4    7   10    5    6    4    6    5    4    5    5    4    4    1    6    6
##  418  420  422  424  426  428  430  432  434  436  438  440  442  444  446  448
##    3    2    6    4    2    2    2    3    2    2    4    6    8    5    2    7
##  450  452  454  456  458  460  462  464  466  468  470  472  474  476  478  480
##    7    3    2    1    4    3    5    1    6    3    6    6    3    3    2    2
##  482  484  486  488  490  492  494  496  498  500  502  506  508  510  512  514
##    5    5    3    3    2    2    2    6    4    5    3    7    3    3    4    5
##  516  518  520  522  524  526  528  530  532  534  536  538  540  542  544  546
##    3    3    3    1    3    4    3    3    1    2    2    2    3    1    3    5
##  548  550  552  554  556  558  560  562  564  566  568  570  572  576  580  584
##    6    2    3    3    5    5    4    2    1    5    2    2    2    2    3    2
##  586  588  590  592  594  596  598  600  602  604  606  608  610  614  616  618
##    2    3    2    2    4    6    5    2    1    2    6    2    4    4    1    1
##  620  624  626  628  630  632  634  640  642  644  648  650  654  656  658  660
##    3    2    3    3    1    2    1    5    1    2    3    3    3    1    1    1
##  662  670  672  676  678  680  682  684  686  690  692  696  698  702  704  706
##    3    1    5    3    1    3    2    4    1    3    3    1    3    2    1    2
##  708  710  712  714  716  718  720  722  724  726  728  730  732  734  736  738
##    2    2    1    2    2    1    1    3    2    2    1    3    1    4    1    1
##  740  742  746  748  750  752  754  756  758  760  762  764  766  768  772  774
##    1    1    4    2    1    1    2    1    1    1    3    2    1    3    3    1
##  776  784  788  790  796  798  804  806  810  812  814  816  818  820  824  828
##    2    1    1    1    3    1    4    1    1    1    1    3    2    3    2    1
##  830  832  834  836  842  844  846  848  850  854  856  858  860  862  864  866
##    2    3    1    1    1    1    1    2    1    2    3    1    2    2    2    1
##  868  870  872  874  876  878  882  884  886  888  892  896  902  908  910  914
##    1    1    2    1    1    1    2    2    1    1    1    2    1    1    1    1
##  916  920  922  928  930  934  936  942  948  952  954  956  962  964  966  972
```

```
##    2    3    1    1    1    1    2    1    4    1    2    1    3    1    2    1
##  974  976  980  982  984  986  988  990  994  998 1002 1004 1008 1010 1014 1022
##    1    2    1    1    1    1    1    2    2    1    1    1    2    1    1    2
## 1024 1026 1028 1030 1032 1038 1040 1042 1044 1046 1050 1054 1056 1074 1076 1078
##    2    1    1    1    2    2    2    1    1    1    1    1    1    1    1    2
## 1080 1082 1090 1092 1098 1102 1104 1106 1110 1114 1120 1122 1126 1132 1134 1146
##    2    1    1    1    2    2    1    2    1    1    2    2    1    2    1    1
## 1148 1152 1154 1162 1164 1172 1176 1178 1180 1186 1188 1190 1206 1208 1216 1222
##    1    1    3    2    1    1    1    1    1    2    1    1    1    1    1    1
## 1224 1226 1228 1232 1242 1246 1250 1252 1254 1256 1262 1264 1266 1272 1274 1280
##    2    3    1    3    1    1    1    1    1    1    1    1    1    1    1    1
## 1286 1288 1292 1294 1298 1300 1304 1314 1318 1320 1322 1326 1332 1346 1348 1352
##    1    2    1    1    1    1    1    1    2    1    1    1    2    1    1    1
## 1356 1360 1370 1384 1386 1396 1410 1416 1430 1442 1446 1450 1454 1460 1462 1466
##    2    1    1    1    1    1    1    2    2    2    1    1    1    1    1    1
## 1482 1494 1496 1498 1500 1502 1508 1516 1520 1534 1536 1548 1560 1564 1566 1572
##    1    1    1    1    1    1    2    1    1    3    1    1    1    1    1    2
## 1578 1586 1594 1596 1618 1636 1638 1640 1660 1664 1674 1676 1684 1690 1692 1734
##    1    1    1    1    1    1    1    1    2    1    1    1    1    2    1    1
## 1740 1754 1756 1760 1764 1768 1778 1782 1792 1794 1798 1804 1816 1822 1828 1842
##    1    1    1    1    2    1    1    1    1    1    1    1    1    2    1    1
## 1864 1866 1868 1870 1882 1884 1886 1896 1898 1932 1936 1958 1984 2006 2008 2018
##    1    1    1    1    1    1    1    2    1    1    1    1    1    1    1    1
## 2028 2030 2046 2058 2092 2128 2152 2258 2292 2326 2330 2346 2392 2406 2440 2458
##    1    1    2    1    1    1    1    1    1    1    1    2    1    2    1    1
## 2460 2504 2520 2560 2598 2600 2606 2616 2660 2678 2718 2738 2742 2824 2870 2878
##    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1
## 2944 2976 2980 3018 3092 3202 3380 3384 3392 3608 3704 3712 3806 3900 4008 4060
##    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1
## 4066 4108 4174 4282 4382 4384 4402 4432 4506 4532 4678 4798 5286 5378 5616 5678
##    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1
## 5696 6370 6408 7834 7982 8276 8586
##    1    1    1    1    1    1    1
```

Apartado 2

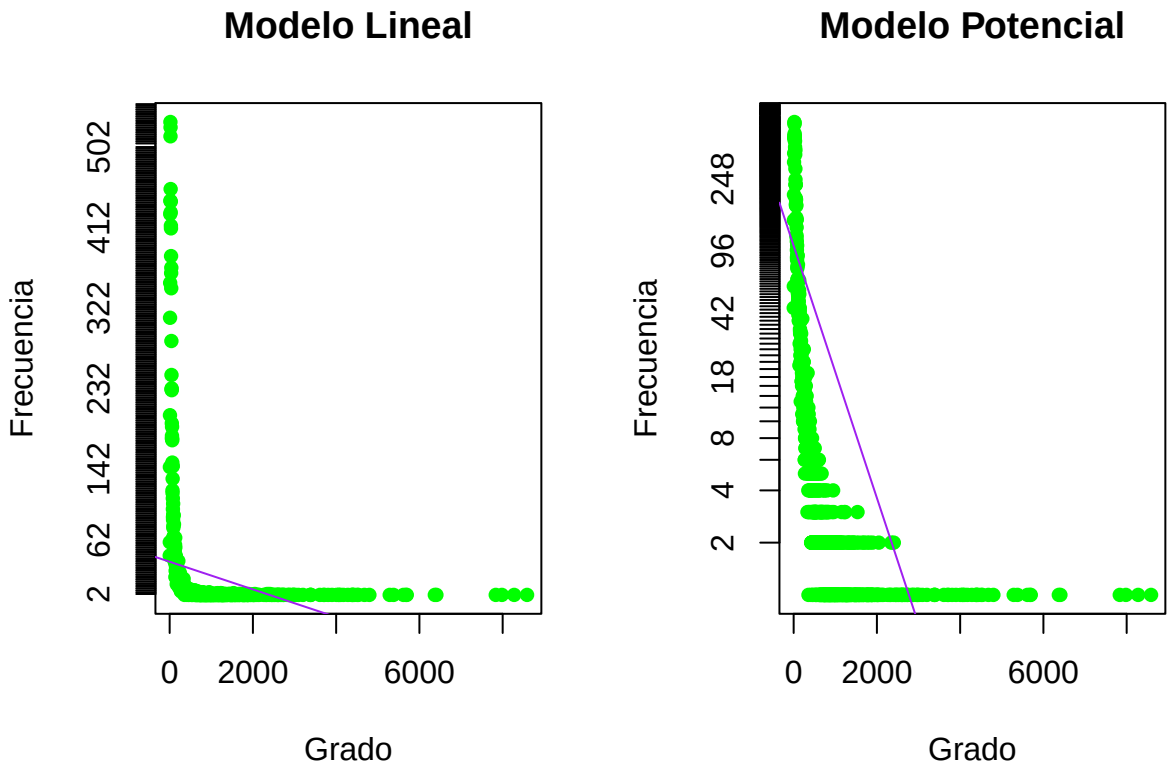
Ajustamos modelos lineales, potenciales y exponenciales utilizando la función lm de R.

```r
# Ajustar modelos lineal, potencial y exponencial
modelo_lineal <- lm(frecuencias ~ as.numeric(names(frecuencias)))
modelo_potencial <- lm(log(frecuencias) ~ as.numeric(names(frecuencias)))
modelo_exponencial <- lm(log(frecuencias) ~ as.numeric(names(frecuencias)))
```

Creamos los gráficos para los modelos lineal y potencial. Se utiliza par(mfrow=c(1,2)) para organizar los gráficos en una fila de dos columnas.

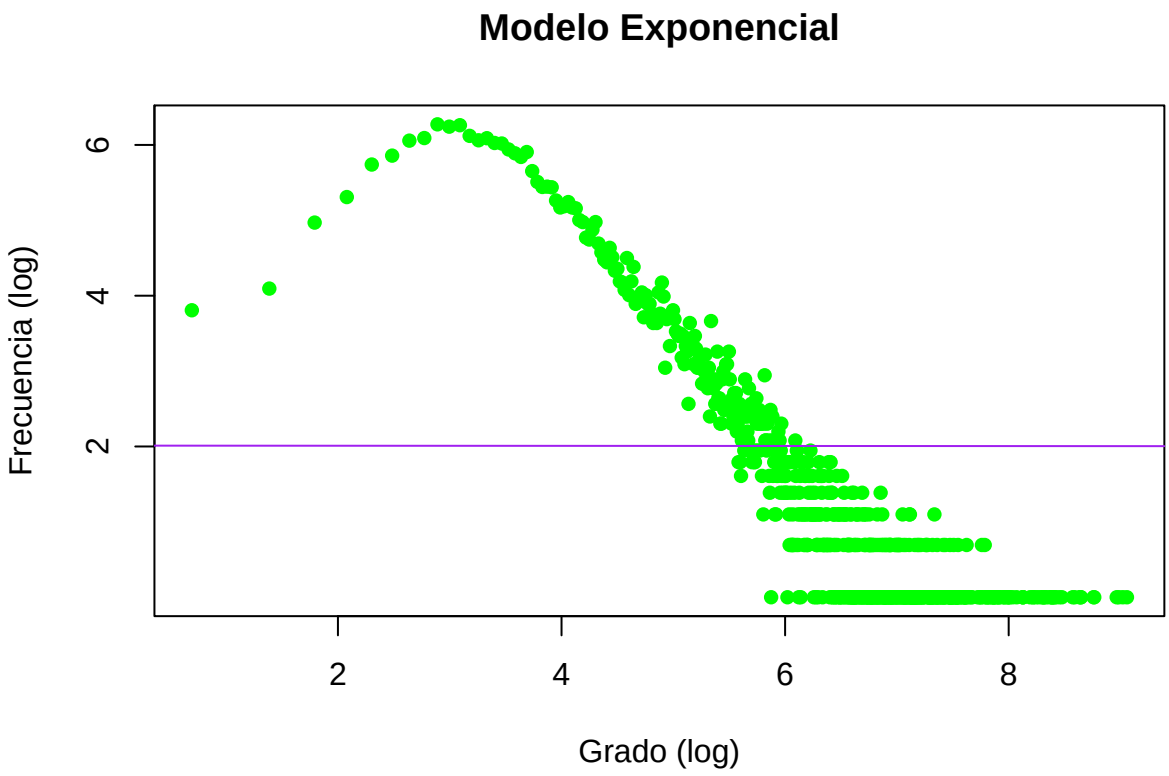
```r
# Dibujar gráficos semi-log y log-log sin leyendas
par(mfrow=c(1,2))

# Gráfico lineal
plot(as.numeric(names(frecuencias)), frecuencias, main="Modelo Lineal", xlab="Grado", ylab="Frecuencia"
abline(modelo_lineal, col="purple")

# Gráfico semi-log
```

```
plot(as.numeric(names(frecuencias)), frecuencias, log = "y", main="Modelo Potencial", xlab="Grado", ylab
abline(modelo_potencial, col="purple")
```



Creamos el gráfico log-log para el modelo exponencial y restauramos el diseño original de la disposición de gráficos.

```
# Gráfico log-log
plot(log(as.numeric(names(frecuencias))), log(frecuencias), main="Modelo Exponencial", xlab="Grado (log)
abline(modelo_exponencial, col="purple")
```




```
# Restaurar el diseño de la disposición
par(mfrow=c(1,1))
```

Apartado 3

Calculamos los errores cuadráticos para cada modelo utilizando la función residuals y la función sum.

```
# Calcular los errores cuadráticos para cada modelo
errores_lineal <- sum(residuals(modelo_lineal)^2)
errores_potencial <- sum(residuals(modelo_potencial)^2)
errores_exponencial <- sum(residuals(modelo_exponencial)^2)
```

Identificamos el mejor modelo seleccionando el que tiene el menor error cuadrático y lo imprimimos.

```
# Identificar el mejor modelo seleccionando el de menor error cuadrático
mejor_modelo <- which.min(c(errores_lineal, errores_potencial, errores_exponencial))

# Imprimir el mejor modelo identificado
cat("Mejor modelo:", c("Lineal", "Potencial", "Exponencial")[mejor_modelo], "\n")
```

```
## Mejor modelo: Potencial
```

Extraemos los coeficientes del mejor modelo y los imprimimos en las unidades originales.

```r
# Extraer los coeficientes del mejor modelo identificado
coeficientes_mejor_modelo <- coef(c(modelo_lineal, modelo_potencial, modelo_exponencial))[[mejor_modelo]]

# Imprimir los coeficientes en las unidades originales
cat("Coeficientes en las unidades originales:", coeficientes_mejor_modelo, "\n")
```

```
## Coeficientes en las unidades originales: -0.01547836
```

Escribimos la ecuación del modelo seleccionado utilizando la función switch.

```r
# Escribir la ecuación del modelo seleccionado
ecuacion_modelo <- switch(mejor_modelo,
  "y = b0 * x + b1",     # Modelo lineal
  "y = b0 * x^b2",       # Modelo Potencial
  "y = b0 * exp(b1 * x)" # Modelo Exponencial
)
cat("Ecuación del modelo:", ecuacion_modelo, "\n")
```

```
## Ecuación del modelo: y = b0 * x^b2
```

1.3 Problema 3: Longitud reviews mallorca AirBnb 2022. 4 puntos

El siguiente código cuenta cuantas palabras hay en un la variable `commnets` del fichero `reviews.csv` de los comentario a cada apartamento de Mallorca extraído de la web Inside AirBnb que recoge datos de los alquileres vacacionales por zonas del mundo de la web de alquiler de apartamentos vacacionales AirBnb. Se puede leer con el siguiente código y contar el número de palabras con la `stringr::str_count`.

```r
read_csv("datasets/reviews.csv")->reviews
```

```
## Rows: 342750 Columns: 6
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (2): reviewer_name, comments
## dbl  (3): listing_id, id, reviewer_id
## date (1): date
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
names(reviews)
```

```
## [1] "listing_id"    "id"            "date"          "reviewer_id"
## [5] "reviewer_name" "comments"
```

```r
library(stringr)
#str_count(str, pattern = "")
str_count(str=reviews$comments[1],pattern ="\\s+")
```

```
## [1] 78
```

Es habitual que la frecuencia de la longitud de los comentarios, es decir cuantos comentarios tienen 5, 6, 7 palabras y sus frecuencias siguen una ley que puede ser: lineal, exponencial o potencial. Como hemos hecho en el tema de regresión lineal calcular se trata de calcular y dibujar los tres modelos y decidir cuál es el más ajustado.

Se pide:

1. Calcular las longitudes de todos los comentarios (utilizar funciones como `mutate`, `arrange`, `filter`....) y las frecuencias de cada longitud y filtrar (con la función `filter`) solo los comentarios con **MÁS de 20 palabras y MENOS de 800** y guardarlos en una tibble con dos columnas N_{words}= número de palabras y $Frec$=frecuencia absoluta de las palabras.

2. Calcular los tres modelos lineal $Freq = \beta_0 + \beta_1 \cdot N_{words}$, potencial $Freq = \beta_0 \cdot (N_{words})^{\beta_1}$ y exponencial $Freq = \beta_0 \cdot \beta_1^{N_{words}}$.

3. Repetir el ajuste anterior pero sustituyendo el la variable N_{words} por el rango u orden de N_{words}.

1.3.1 Solución 3

Apartado 1

Vamos a calcular la longitud de cada comentario y lo añadimos en una columna nueva llamada *Nwords*.

```
# Le sumamos 1 porque estamos contando los espacios y para contar las palabras
# necesitamos contar los espacios + 1
reviews <- reviews %>%
  mutate(Nwords = str_count(comments, "\\s+") + 1)
```

Ahora ya podemos calcular las frecuencias de cada longitud y filtrar los comentarios con más de 20 palabras y menos de 800. Lo metemos en una tibble(*word_freq*) con dos columnas: *Nwords* y *Frec*.

```
# Calclulamos la frecuencia
word_freq <- reviews %>%
  group_by(Nwords) %>%
  summarise(Frec = n(), .groups = "drop")

# Filtramos los comentarios
word_freq <- word_freq %>%
  filter(Nwords > 20, Nwords < 800)

head(word_freq)
```

```
## # A tibble: 6 x 2
##   Nwords  Frec
##    <dbl> <int>
## 1     21  4925
## 2     22  4791
## 3     23  4868
## 4     24  4842
## 5     25  4645
## 6     26  4723
```

Apartado 2

Calculemos el modelo lineal:

```
modelo_lineal <- lm(Frec ~ Nwords, data = word_freq)
```

Calculemos el modelo potencial:

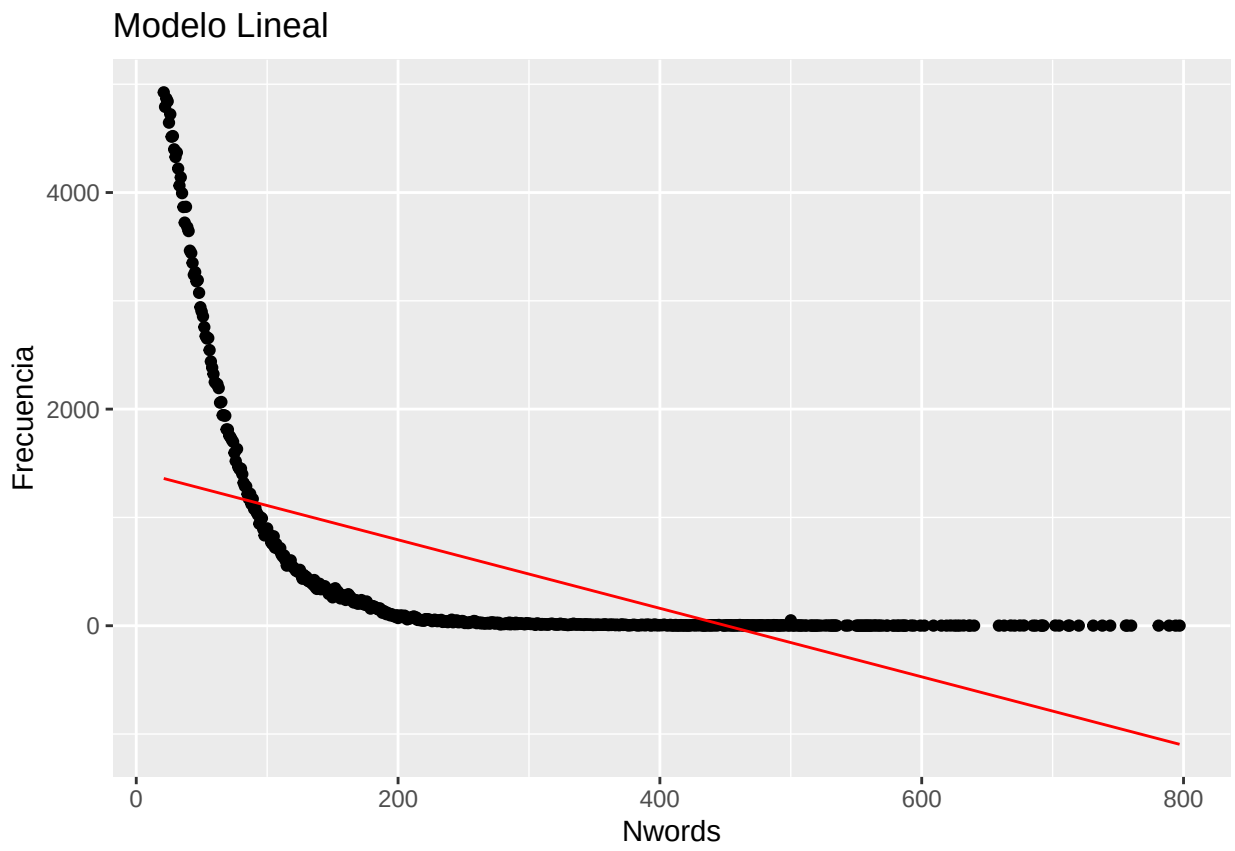
```
word_freq_transformed <- word_freq %>%
  mutate(log_Frec = log(Frec), log_Nwords = log(Nwords))

modelo_potencial <- nls(log_Frec ~ log(B0) + B1 * log_Nwords, data = word_freq_transformed, start = lis
```

Calculemos el modelo exponencial:

```
word_freq_transformed <- word_freq %>%
  mutate(log_Frec = log(Frec))
modelo_exponencial <- nls(log_Frec ~ log(B0) + Nwords * log(B1), data = word_freq_transformed, start = l
```

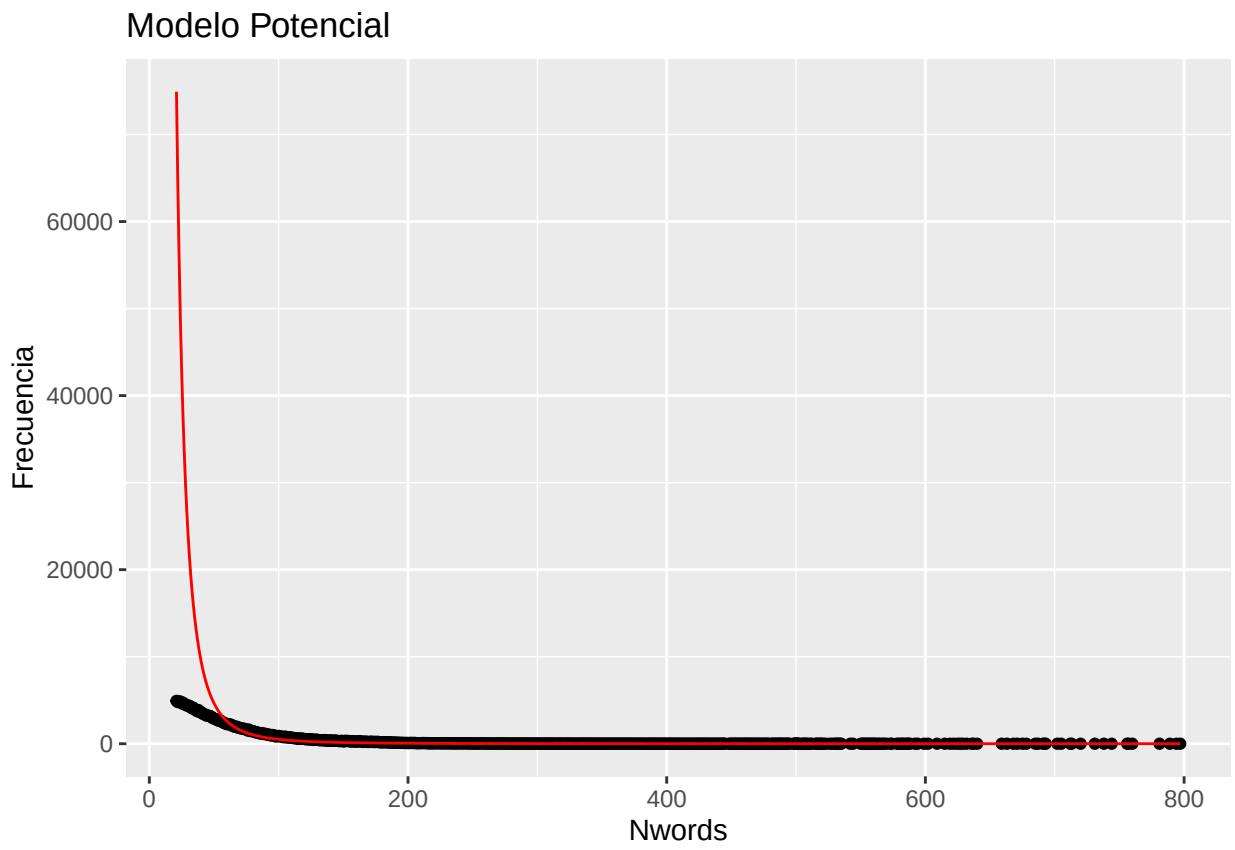
Para completar un poco voy ha hacer los diferentes graficos.

```
word_freq$Pred_Lineal <- predict(modelo_lineal, newdata = word_freq)
ggplot(word_freq, aes(x = Nwords, y = Frec)) +
  geom_point() +
  geom_line(aes(y = Pred_Lineal), color = "red") +
  labs(title = "Modelo Lineal", x = "Nwords", y = "Frecuencia")
```



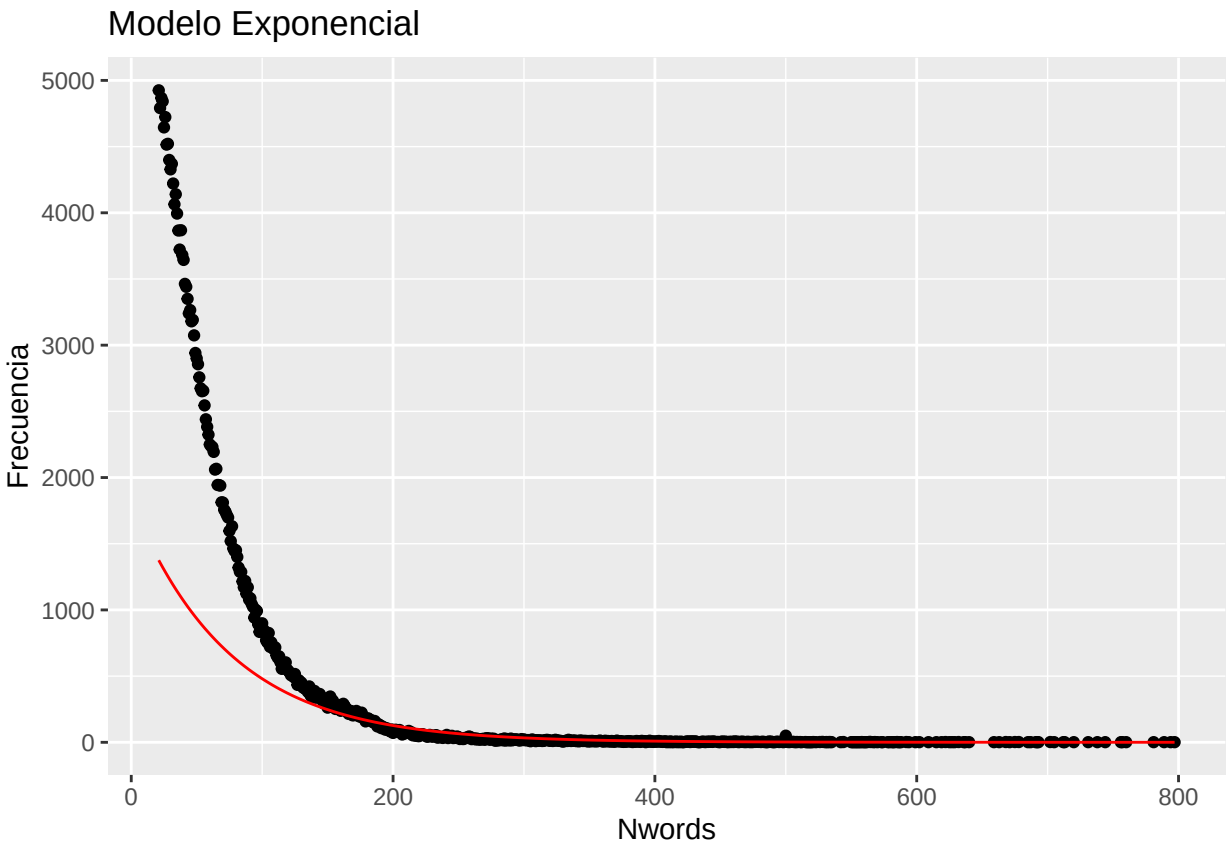
El modelo lineal no se ajusta muy bien a la distribución de los datos.

```
word_freq$Pred_Potencial <- exp(predict(modelo_potencial, newdata = word_freq_transformed))
ggplot(word_freq, aes(x = Nwords, y = Frec)) +
  geom_point() +
  geom_line(aes(y = Pred_Potencial), color = "red") +
  labs(title = "Modelo Potencial", x = "Nwords", y = "Frecuencia")
```



El modelo potencial se empieza a ajustar a la distribución datos cuando Nwords es mayor que 80(aprox)

```
word_freq$Pred_Exponencial <- exp(predict(modelo_exponencial, newdata = word_freq_transformed))
ggplot(word_freq, aes(x = Nwords, y = Frec)) +
  geom_point() +
  geom_line(aes(y = Pred_Exponencial), color = "red") +
  labs(title = "Modelo Exponencial", x = "Nwords", y = "Frecuencia")
```



El modelo exponencial se empieza a ajustar a la distribución datos cuando Nwords es mayor que 175(aprox)

Apartado 3

Vamos a substituir la variable *Nwords* por el rango de *Nwords*.

```
word_freq <- word_freq %>%
  mutate(Rank_Nwords = rank(Nwords))
```

Ahora realizamos los mismos pasos que en el apartado 2.

Calculemos el modelo lineal:

```
modelo_lineal_rank <- lm(Frec ~ Rank_Nwords, data = word_freq)
```

Calculemos el modelo potencial:

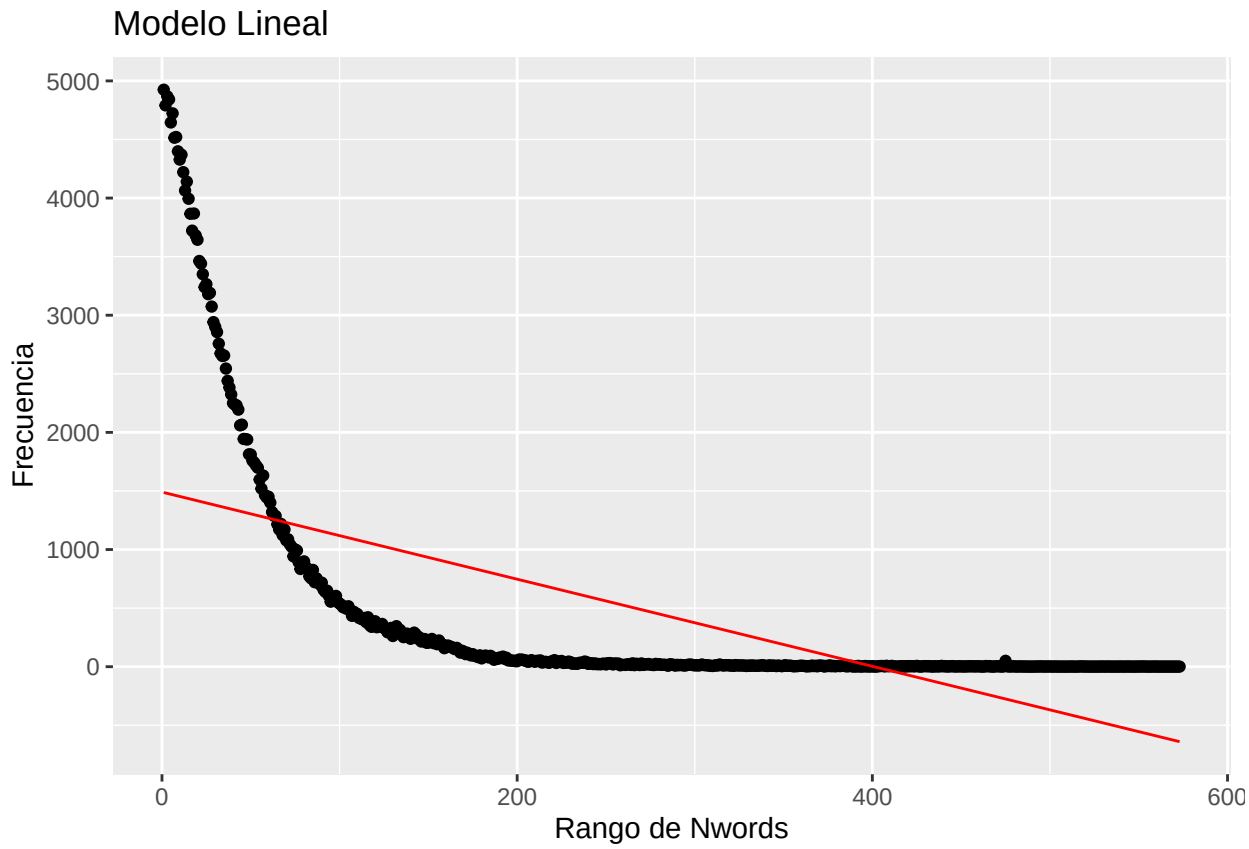
```
word_freq_transformed <- word_freq %>%
  mutate(log_Frec = log(Frec), log_Rank_Nwords = log(Rank_Nwords))

modelo_potencial_rank <- nls(log_Frec ~ log(B0) + B1 * log_Rank_Nwords, data = word_freq_transformed, s
```

Calculemos el modelo exponencial:

```r
word_freq_transformed <- word_freq %>%
  mutate(log_Frec = log(Frec))
modelo_exponencial_rank <- nls(log_Frec ~ log(B0) + Rank_Nwords * log(B1), data = word_freq_transformed
```

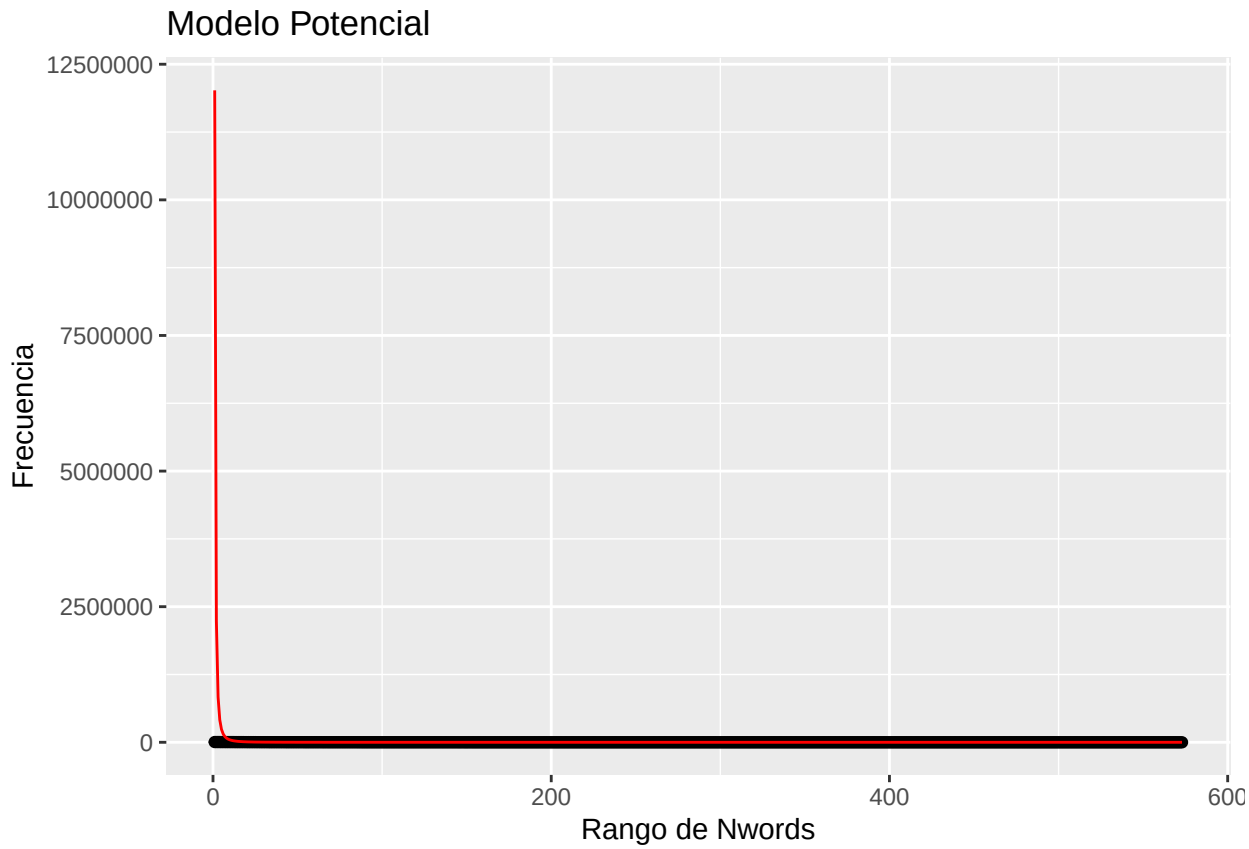
Para completar un poco voy ha hacer los diferentes graficos.

```r
word_freq$Pred_Lineal <- predict(modelo_lineal_rank, newdata = word_freq)
ggplot(word_freq, aes(x = Rank_Nwords, y = Frec)) +
  geom_point() +
  geom_line(aes(y = Pred_Lineal), color = "red") +
  labs(title = "Modelo Lineal", x = "Rango de Nwords", y = "Frecuencia")
```



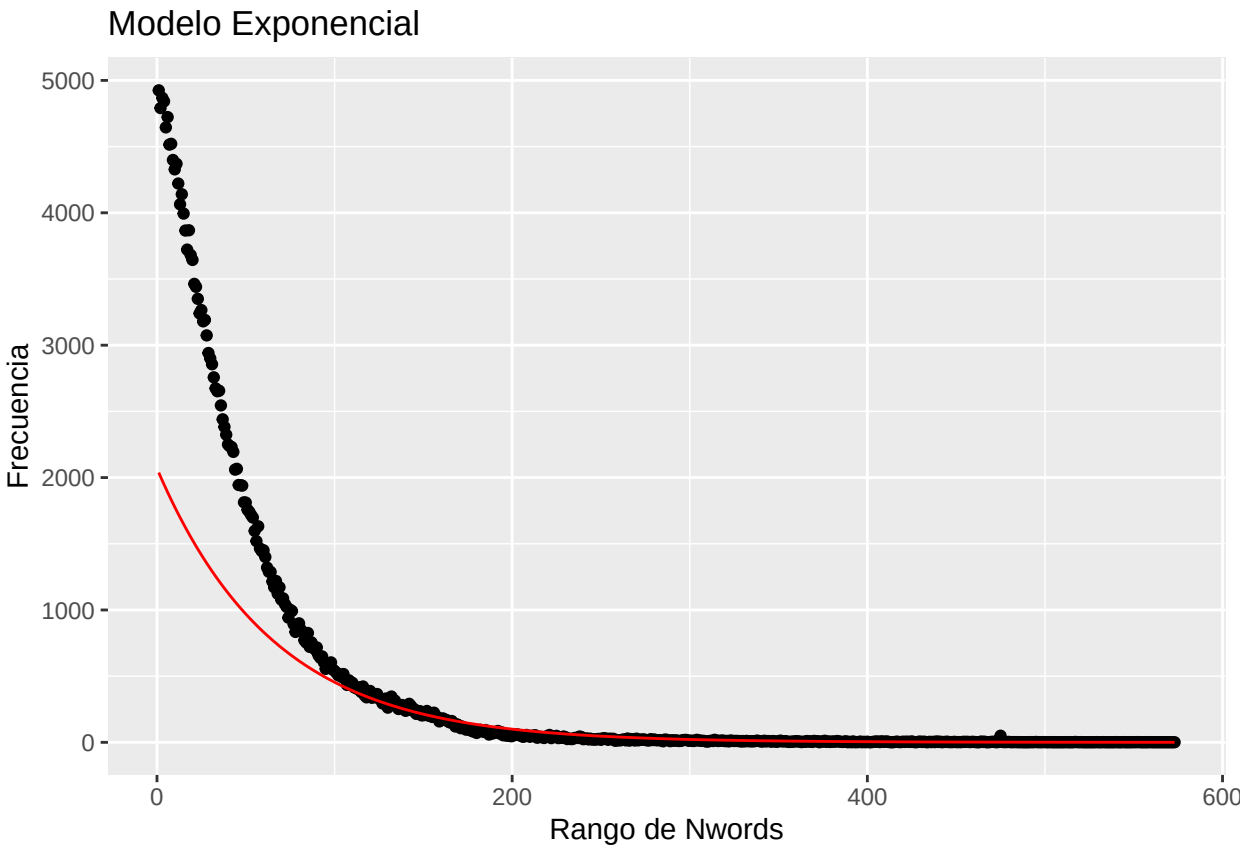
El modelo lineal no se ajusta muy bien a la distribución de los datos.

```r
word_freq$Pred_Potencial <- exp(predict(modelo_potencial_rank, newdata = word_freq_transformed))
ggplot(word_freq, aes(x = Rank_Nwords, y = Frec)) +
  geom_point() +
  geom_line(aes(y = Pred_Potencial), color = "red") +
  labs(title = "Modelo Potencial", x = "Rango de Nwords", y = "Frecuencia")
```



El modelo potencial no se ajusta para nada a la distribución de los datos en los primeros rangos de *Nwords*.

```
word_freq$Pred_Exponencial <- exp(predict(modelo_exponencial_rank, newdata = word_freq_transformed))
ggplot(word_freq, aes(x = Rank_Nwords, y = Frec)) +
  geom_point() +
  geom_line(aes(y = Pred_Exponencial), color = "red") +
  labs(title = "Modelo Exponencial", x = "Rango de Nwords", y = "Frecuencia")
```



El modelo exponencial se empieza a ajustar a la distribución datos cuando Nwords es mayor que 100(aprox).