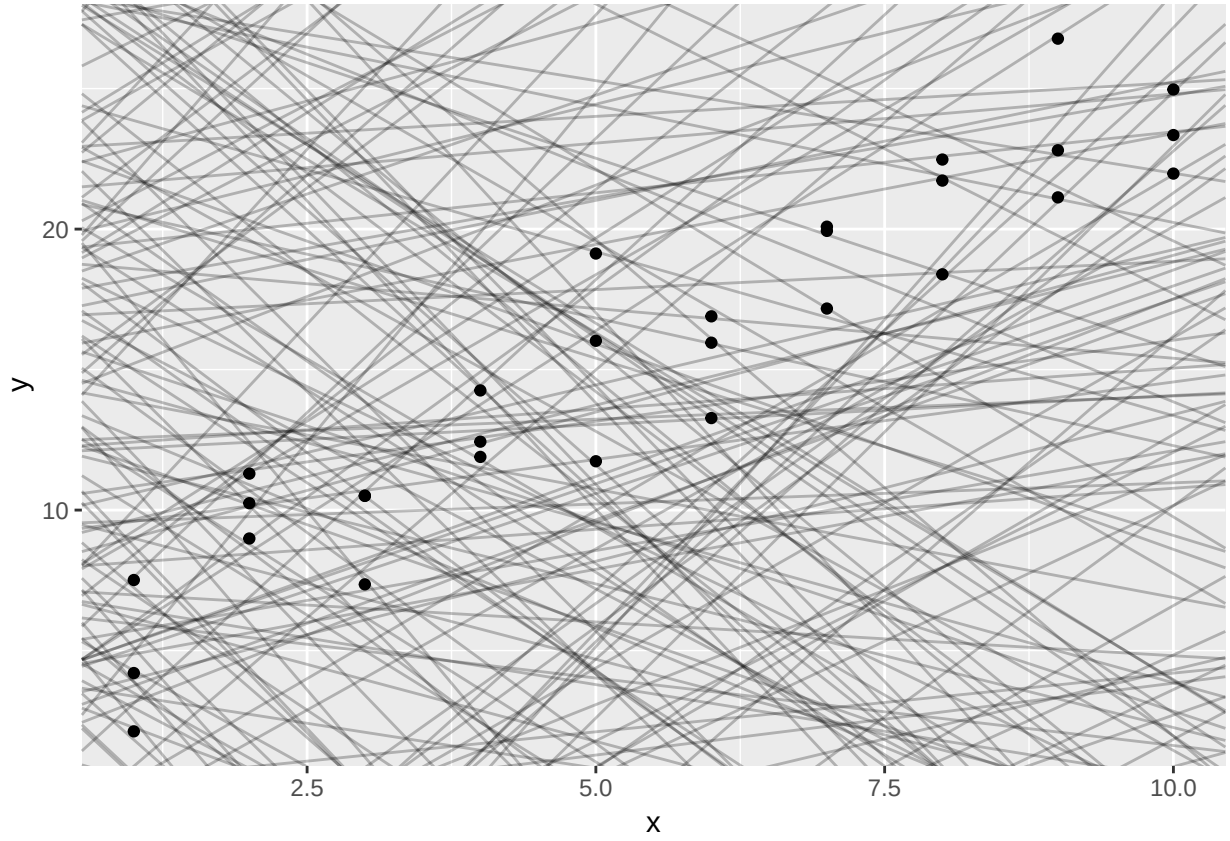
Parte IV: Modelado

ronny hdez-mora

5/26/2019

```
models <- tibble(
  a1 = runif(250, -20, 40),
  a2 = runif(250, -5, 5)
)

ggplot(sim1, aes(x, y)) +
  geom_abline(
    aes(intercept = a1, slope = a2),
    data = models, alpha = 1/4
  ) +
  geom_point()
```



Los 250 modelos anteriores hay unos que son muy malos. Necesitamos uno que esté más cerca de los datos. Podemos ajustar uno encontrando los valores de `a_0` y `a_1` que genere el modelo con las distancias mínimas a estos datos.

Esto se puede hacer con la distancia vertical entre cada punto y el modelo. Esta distancia es la diferencia entre el valor de y dado por el modelo (predicción) y el valor de y real en los datos.

```
model1 <- function(a, data) {
  a[1] + data$x * a[2]
}

model1(c(7, 1.5), sim1)
```

```
##  [1]  8.5  8.5  8.5 10.0 10.0 10.0 11.5 11.5 11.5 13.0 13.0 13.0 14.5 14.5
## [15] 14.5 16.0 16.0 16.0 17.5 17.5 17.5 19.0 19.0 19.0 20.5 20.5 20.5 22.0
## [29] 22.0 22.0
```

... Revisar este capítulo después

Capítulo 19

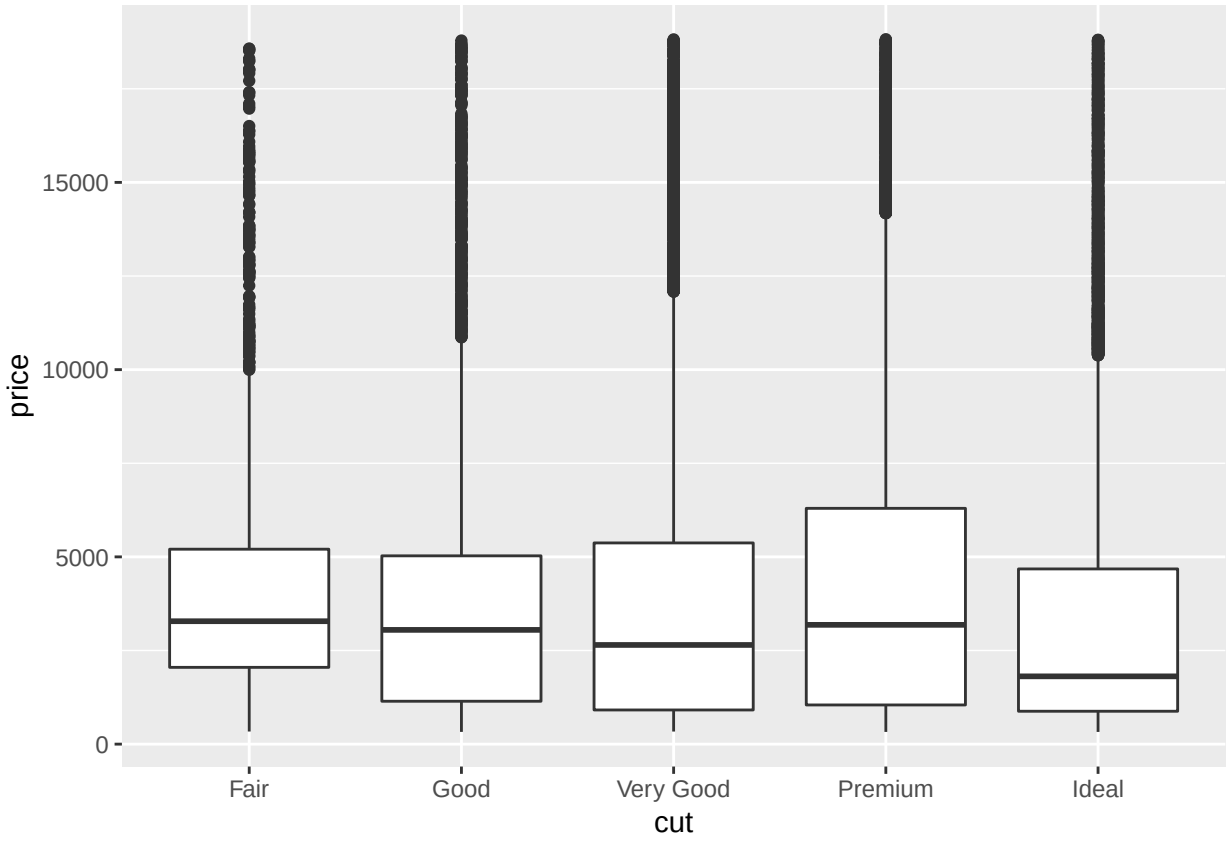
```
library(nycflights13)
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

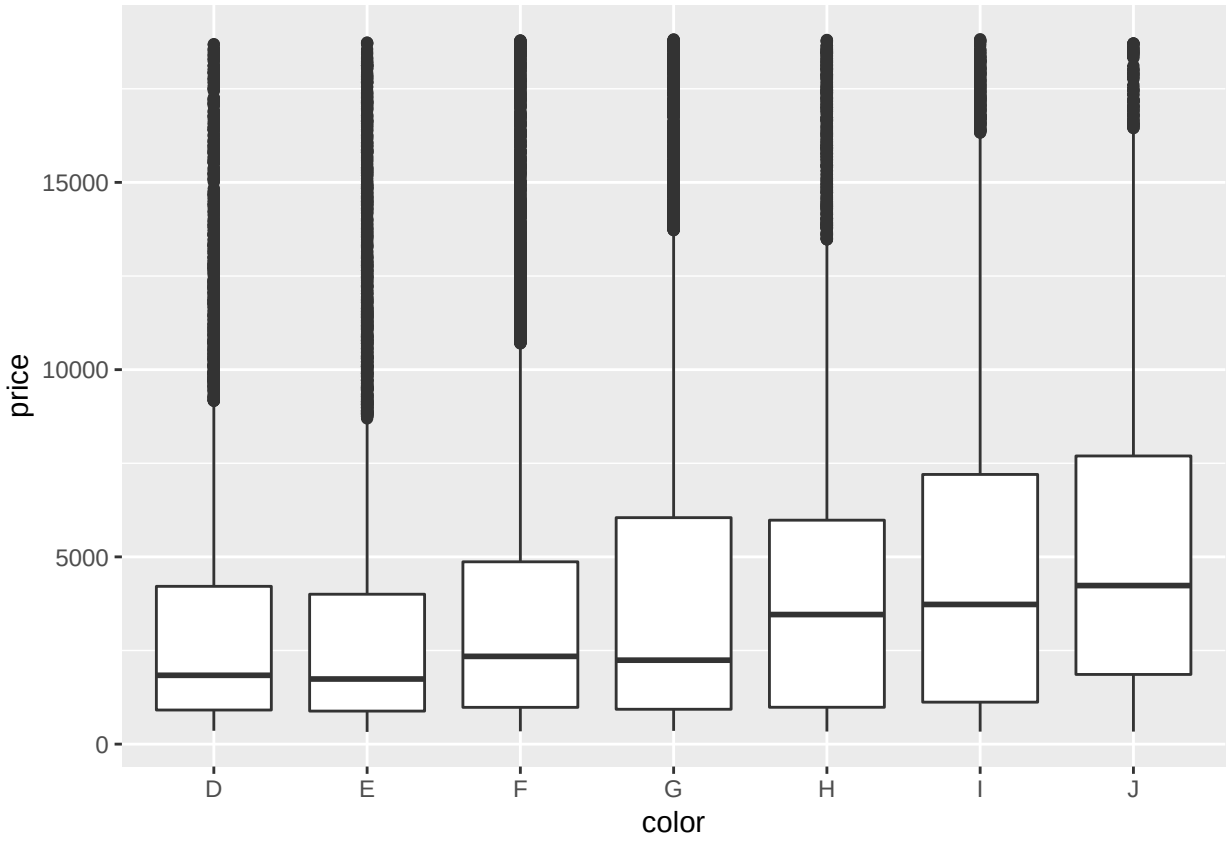
```
## The following object is masked from 'package:base':
##
##     date
```

¿Porqué son los diamantes de baja calidad caros?

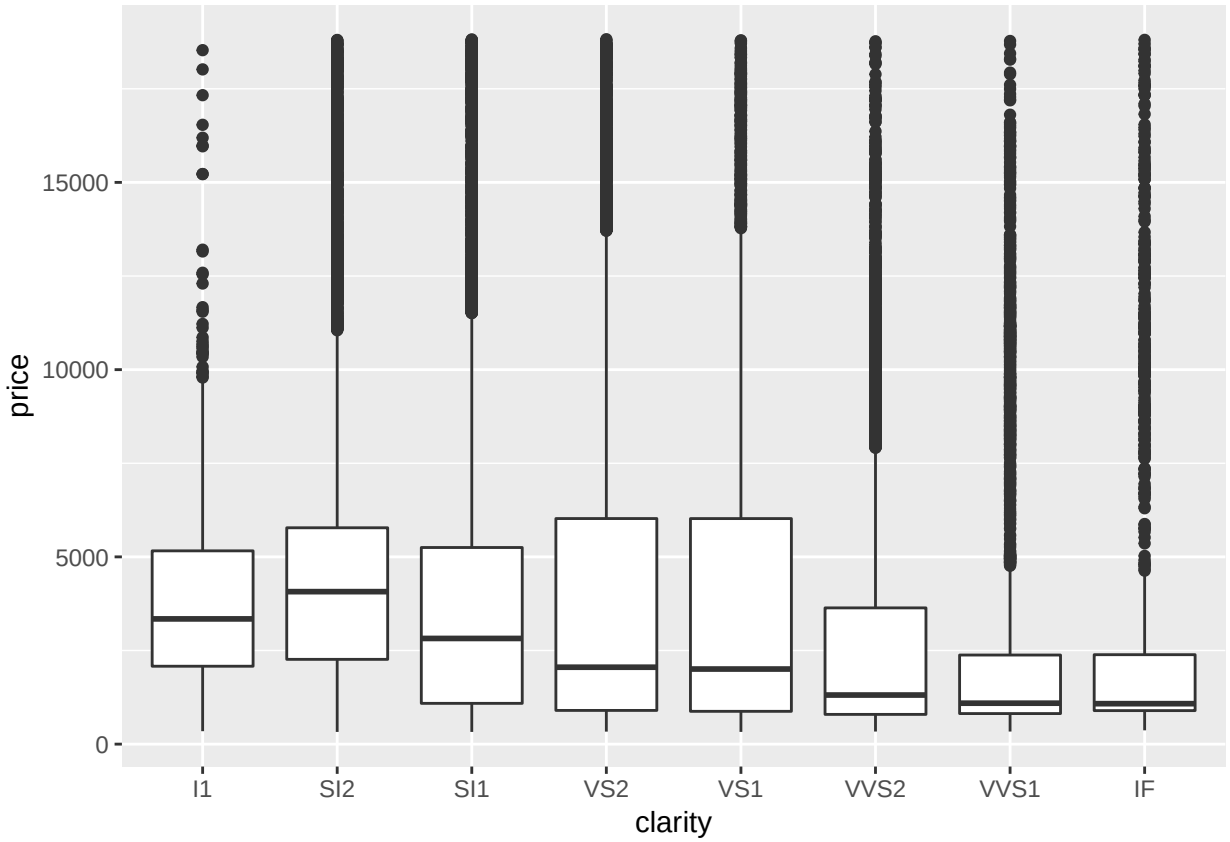
```
ggplot(diamonds, aes(cut, price)) +
  geom_boxplot()
```




```
ggplot(diamonds, aes(color, price)) +
  geom_boxplot()
```

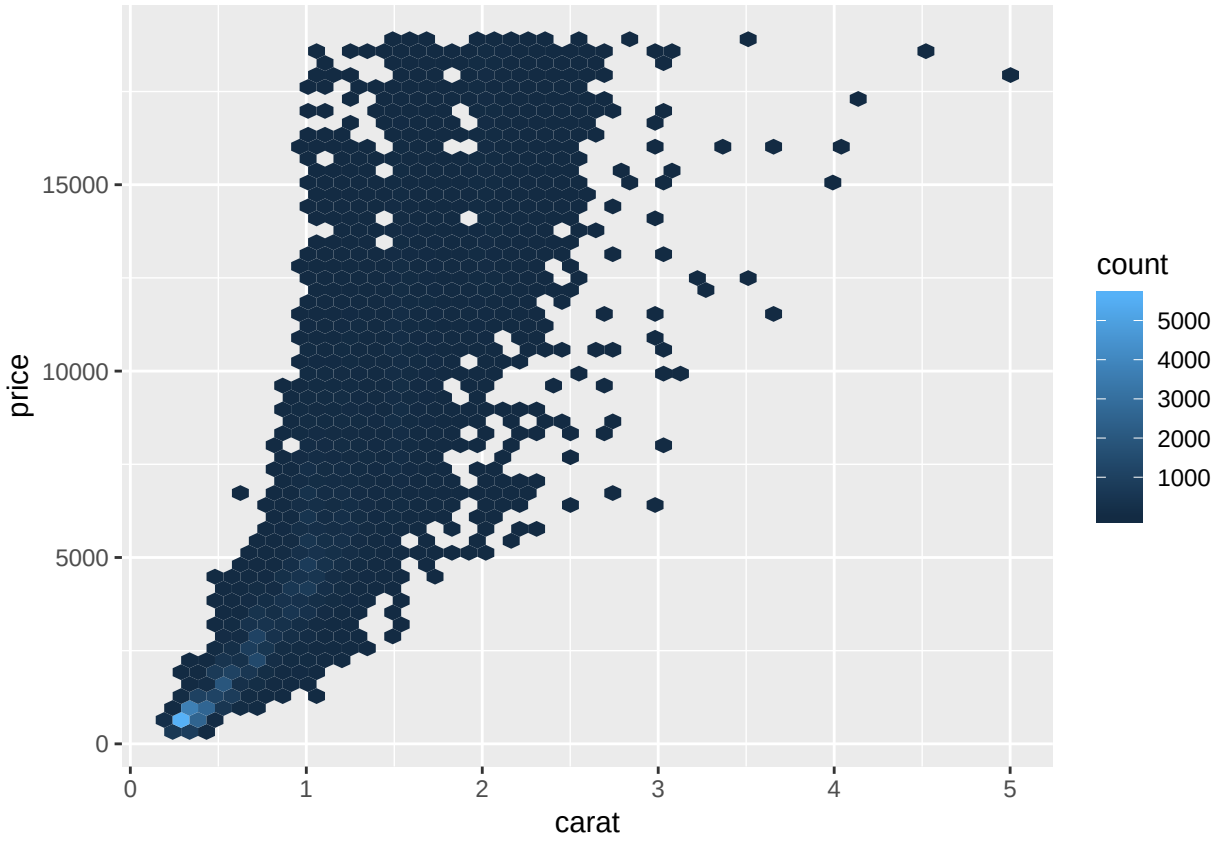



```
ggplot(diamonds, aes(clarity, price)) +
  geom_boxplot()
```



Lo anterior muestra que características malas parecen tener precios más altos. Sin embargo hay uyna característica que está relacionada con el precio y es el peso (carat)

```
ggplot(diamonds, aes(carat, price)) +
  geom_hex(bins = 50)
```



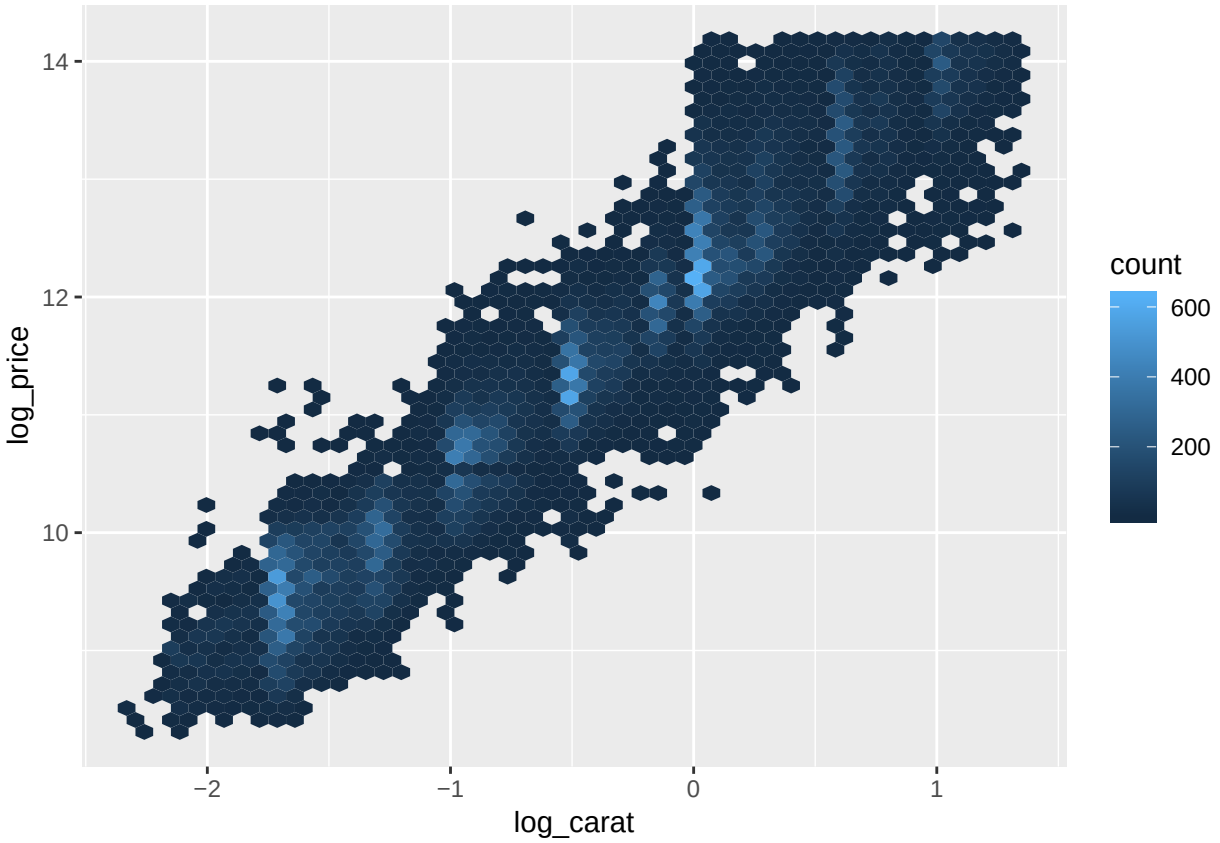
Vamos a limpiar los datos para hacerlos un poco más manipulables:

- Enfocarse en aquellos menores a 2.5 de carat
- Hacer transformación logaritmica de carat y price

```
diamonds2 <- diamonds %>%
  filter(carat <= 2.5) %>%
  mutate(log_price = log2(price),
         log_carat = log2(carat))
```

Con esos cambios la relación será más fácil de verla entre carat y price

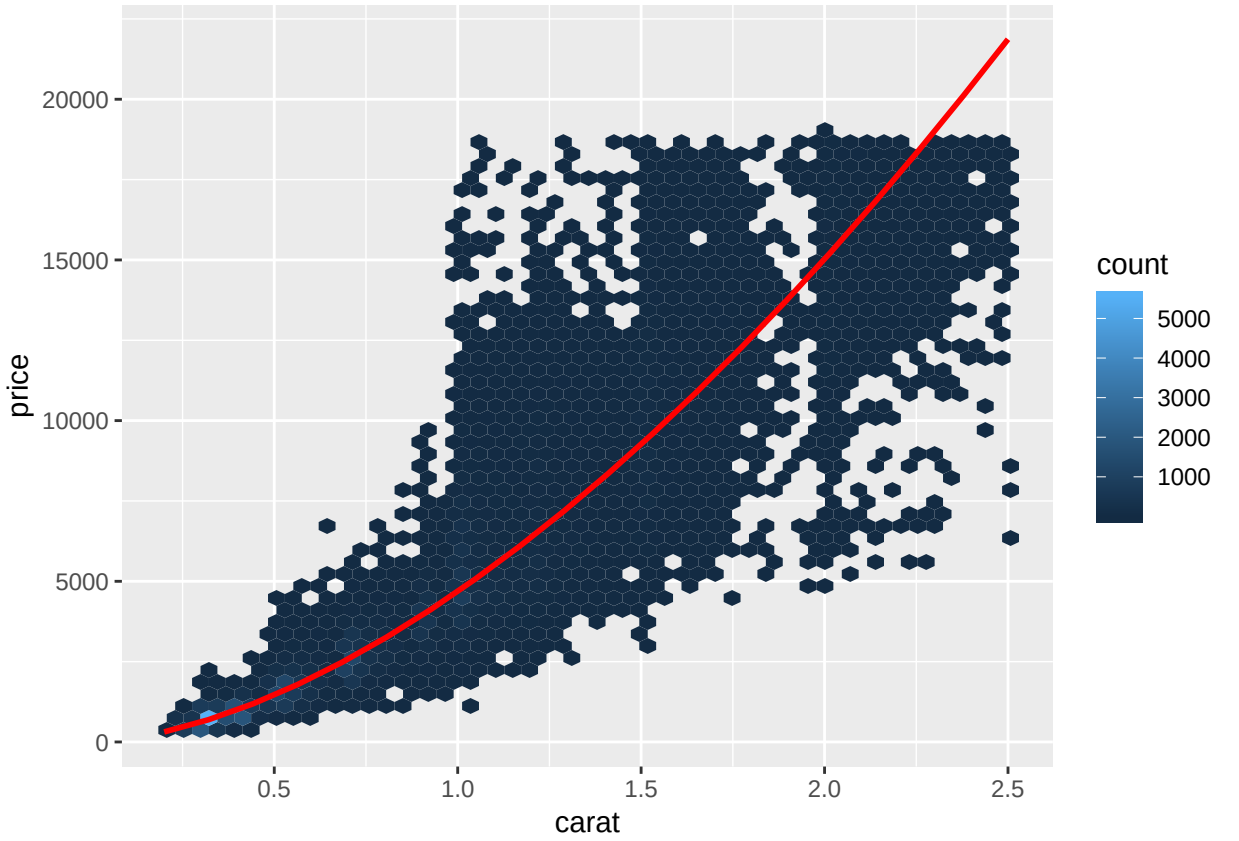
```
ggplot(diamonds2, aes(x = log_carat, y = log_price)) +
         geom_hex(bins = 50)
```




```
mod_diamond <- lm(log_price ~ log_carat, data = diamonds2)
```

Vamos a sobreponer las predicciones del modelo en el gráfico con los valores iniciales

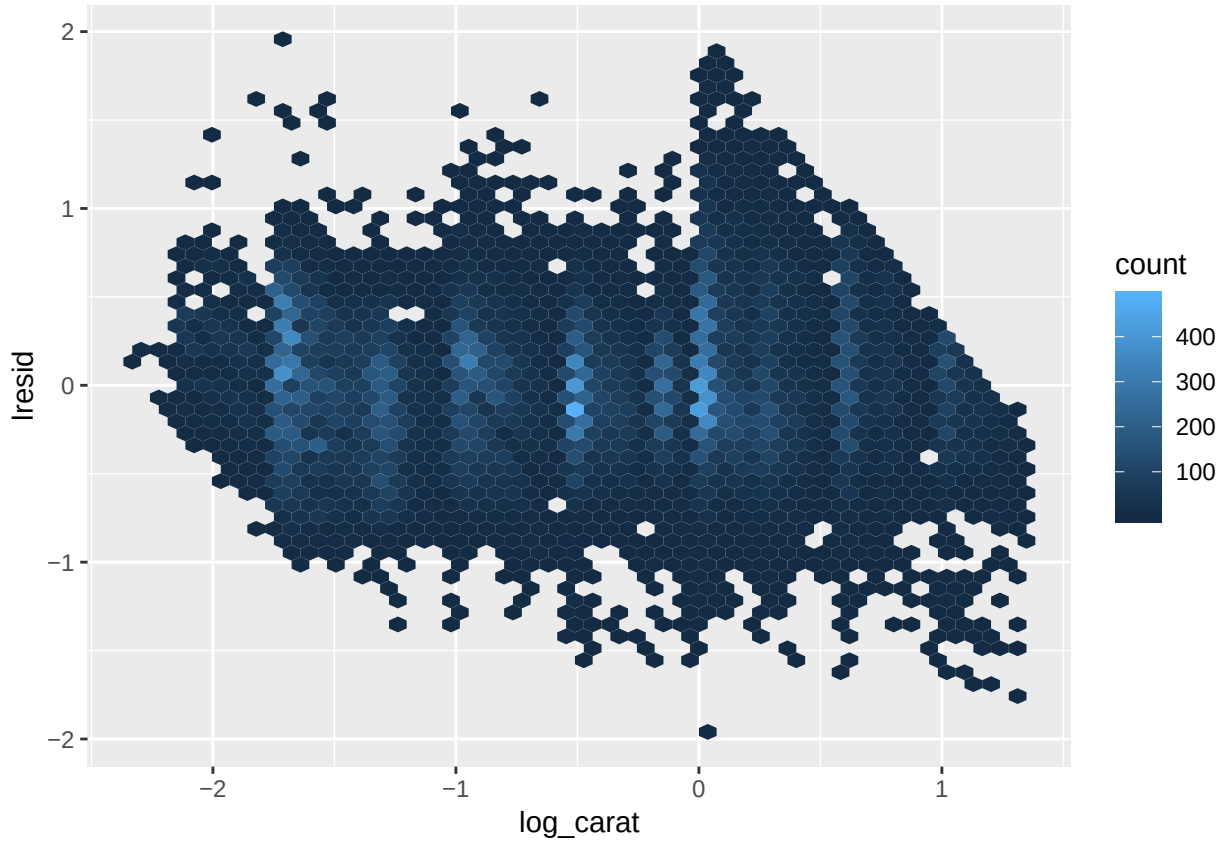
```
grid <- diamonds2 %>%
  data_grid(carat = seq_range(carat, 20)) %>%
  mutate(log_carat = log2(carat)) %>%
  add_predictions(mod_diamond, "log_price") %>%
  mutate(price = 2 ^ log_price)

ggplot(diamonds2, aes(carat, price)) +
  geom_hex(bins = 50) +
  geom_line(data = grid, color = "red", size = 1)
```



Si creemos en nuestro modelo, lo qu vemos es que los diamantes grandes son mucho más baratos de lo esperado.

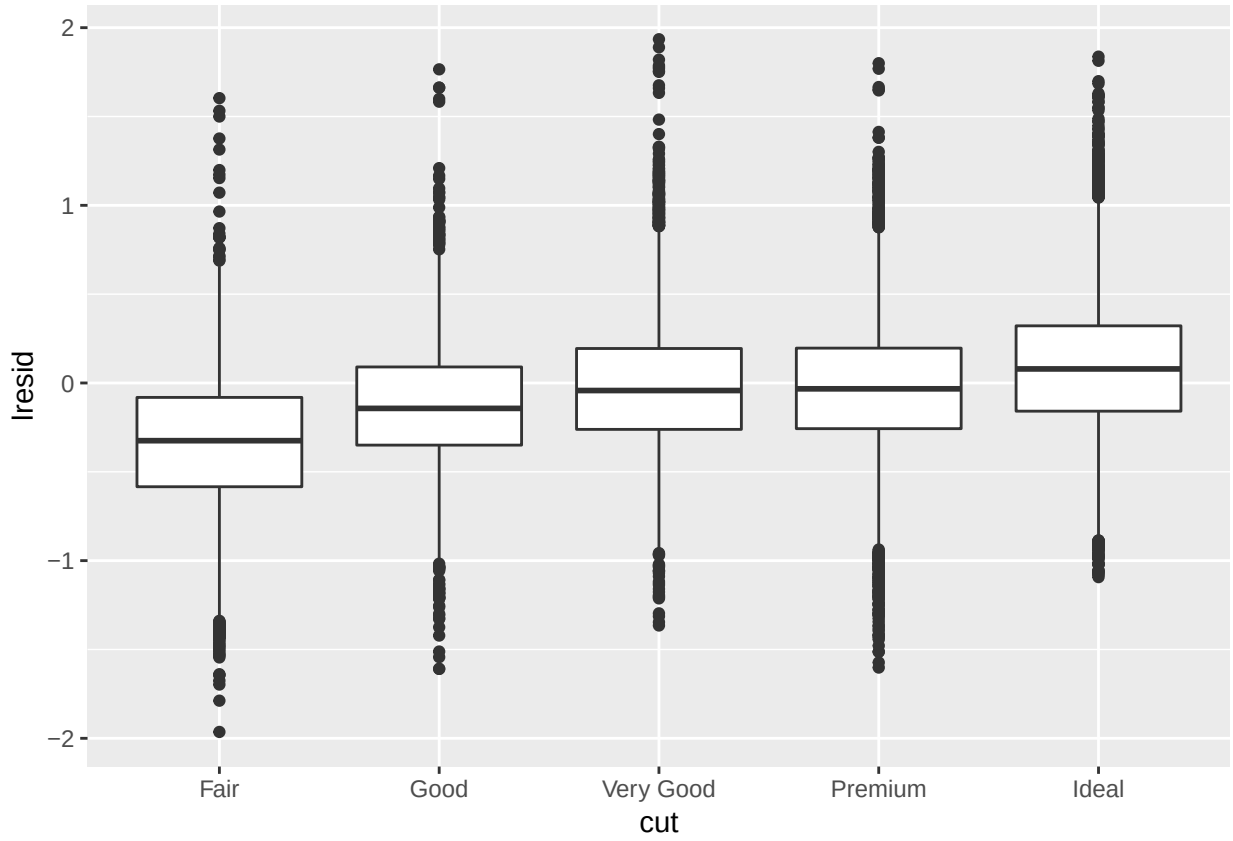
Vamos a revisar los residuales para corroborar que se ha elimiado el patrón lineal

```
diamonds2 <- diamonds2 %>%
  add_residuals(mod_diamond, "lresid")

ggplot(diamonds2, aes(log_carat, lresid)) +
  geom_hex(bins = 50)
```

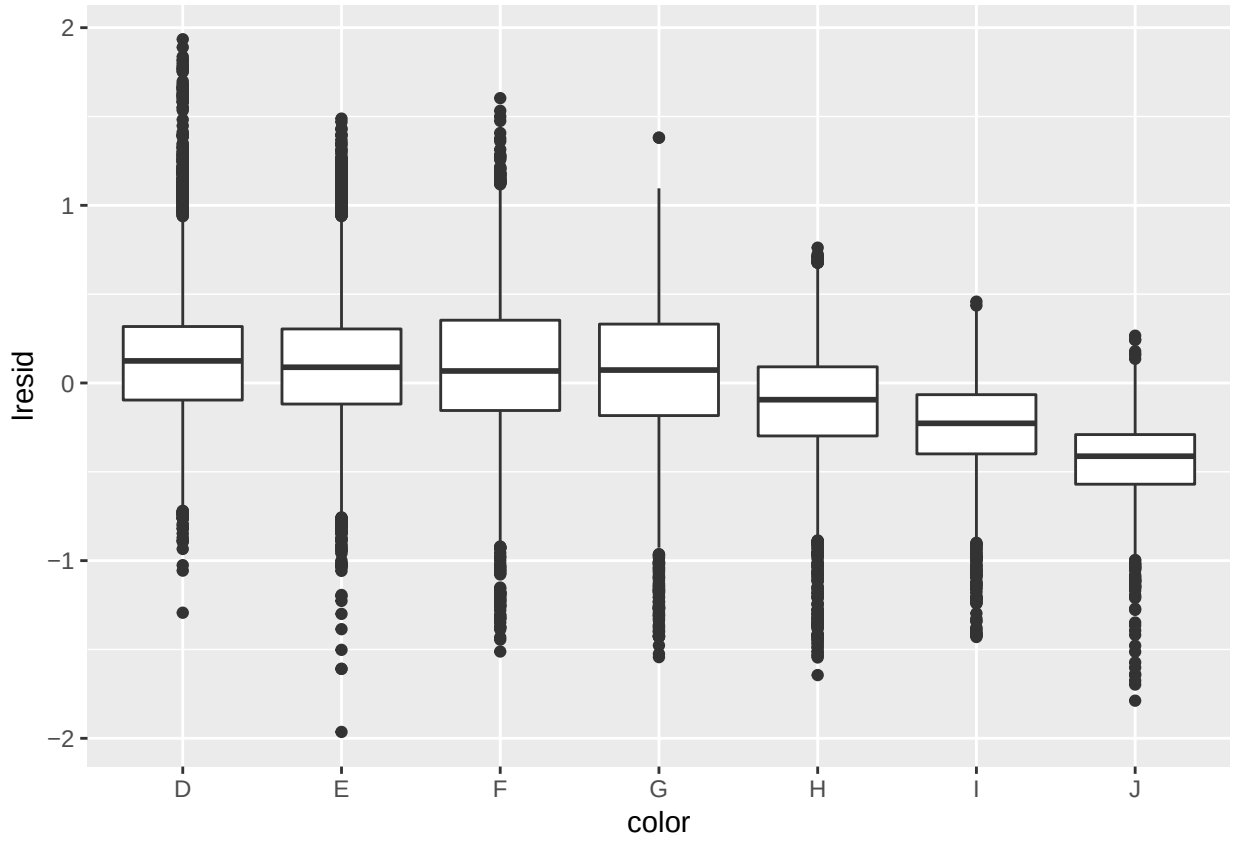


Ahora podemos hacer nuestros gráficos iniciales usando los residuales en lugar del precio

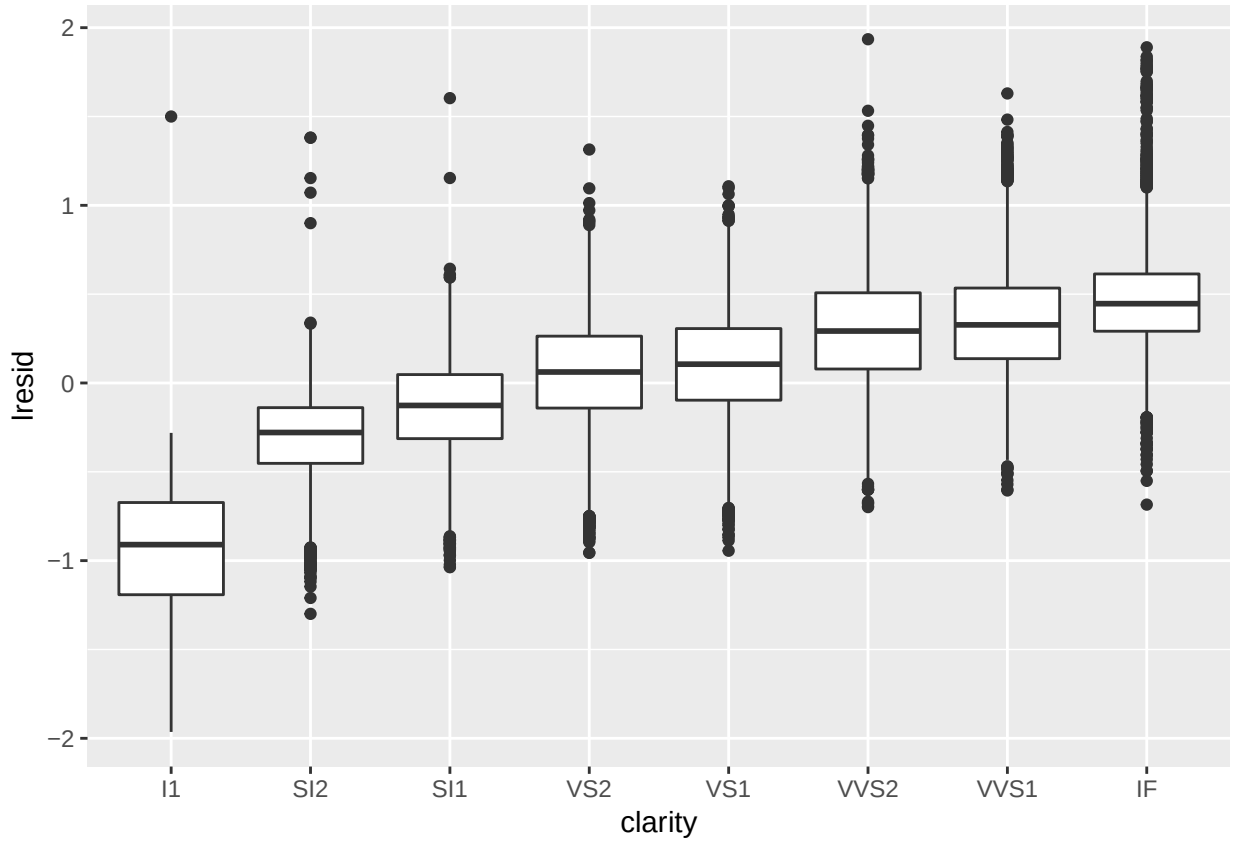
```
ggplot(diamonds2, aes(cut, lresid)) +
  geom_boxplot()
```




```
ggplot(diamonds2, aes(color, lresid)) +
  geom_boxplot()
```




```
ggplot(diamonds2, aes(clarity, lresid)) +
  geom_boxplot()
```



Ahora vemos que a mayor calidad del diamante, mayor el precio.

Un modelo más complicado

Para hacer más explícito lo encontrado se puede incluir color, cut y carat en el modelo y dejar claro el impacto que tienen esas variables categóricas

```
mod_diamond2 <- lm(
  log_price ~ log_carat + color + cut + clarity,
  data = diamonds2
)
```

EL modelo tiene ahora 4 predictores, por lo que es más difícil visualizar. Al ser independientes se pueden graficar por separado.

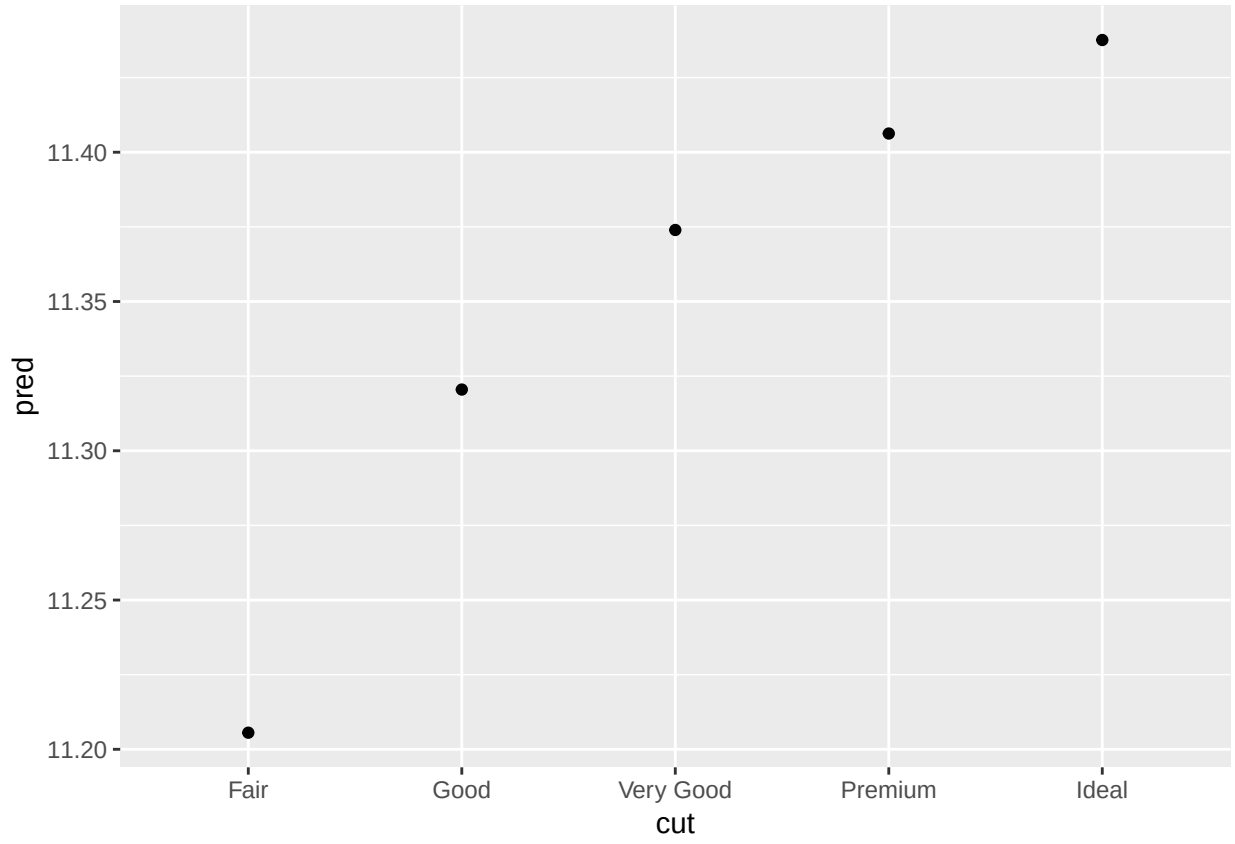
Vamos a usar el argumento `.model` para `data_grid`

```
grid <- diamonds2 %>%
  data_grid(cut, .model = mod_diamond2) %>%
  add_predictions(mod_diamond2)

grid
```

```
## # A tibble: 5 x 5
##   cut       log_carat color clarity  pred
##   <ord>         <dbl> <chr> <chr>   <dbl>
## 1 Fair         -0.515 G     VS2      11.2
```

```
## 2 Good         -0.515 G     VS2      11.3
## 3 Very Good    -0.515 G     VS2      11.4
## 4 Premium      -0.515 G     VS2      11.4
## 5 Ideal        -0.515 G     VS2      11.4
```

```
ggplot(grid, aes(cut, pred)) +
  geom_point()
```



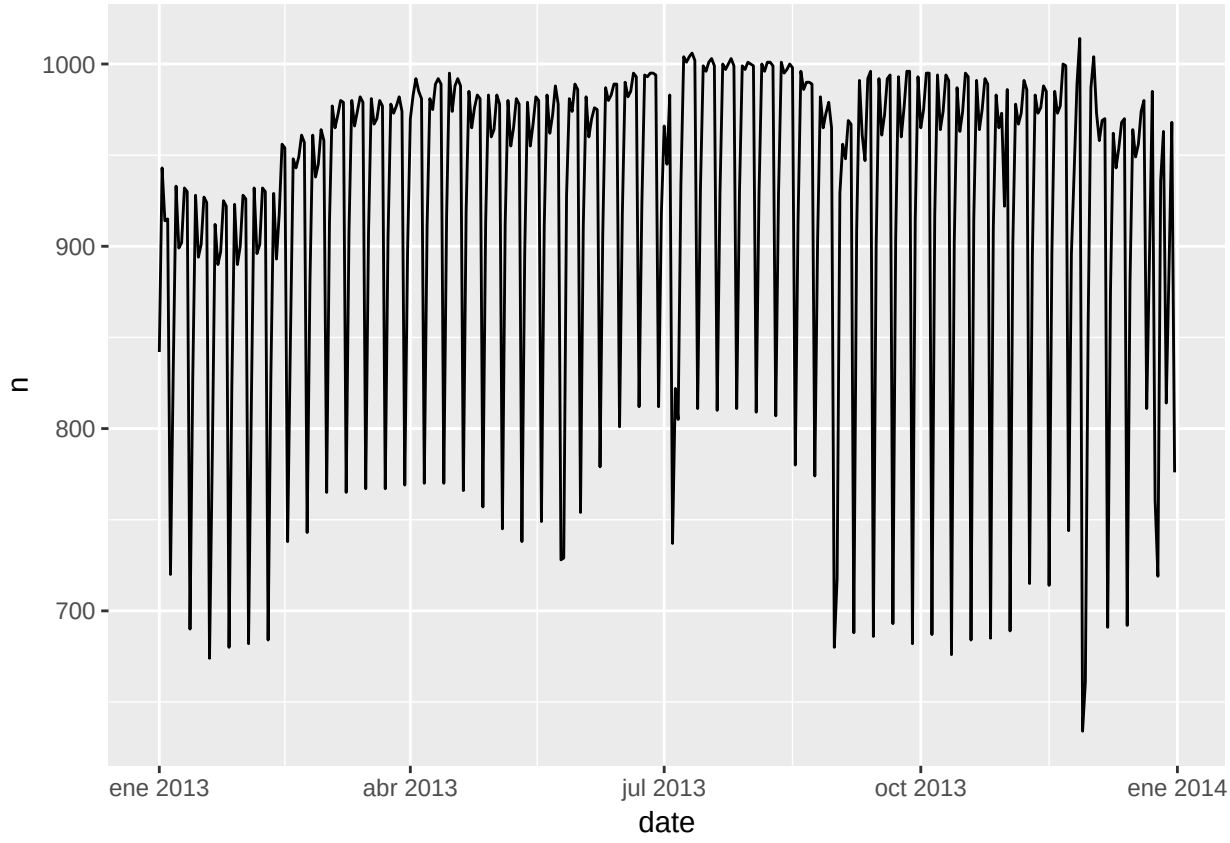
¿Qué afecta el número de vuelos?

Este es un ejemplo para mostras los pasos para entender mejor los datos. Vamos a iniciar contanto la cantidad de vuelos por dia y visualizandolo con ggplot2

```
daily <- flights %>%
  mutate(date = make_date(year, month, day)) %>%
  group_by(date) %>%
  summarize(
    n = n()
    )

glimpse(daily)
```

```
## Observations: 365
## Variables: 2
## $ date <date> 2013-01-01, 2013-01-02, 2013-01-03, 2013-01-04, 2013-01-...
## $ n    <int> 842, 943, 914, 915, 720, 832, 933, 899, 902, 932, 930, 69...
```

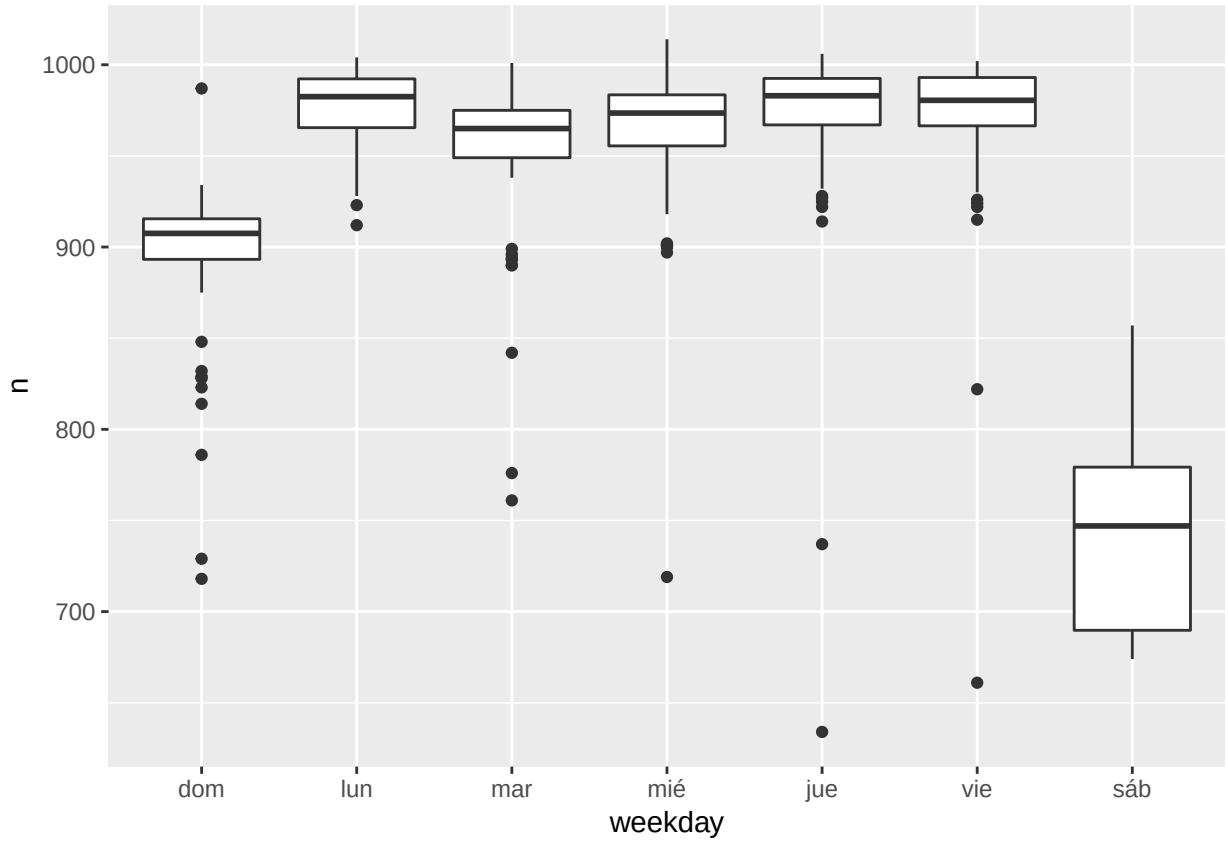
```r
ggplot(daily, aes(x = date, y = n)) +
  geom_line()
```



Día de la semana

Entender el patrón a largo plazo es difícil porque hay un efecto muy fuerte de día de la semana. Vamos a revisar la distribución del número de vuelos por día de la semana:

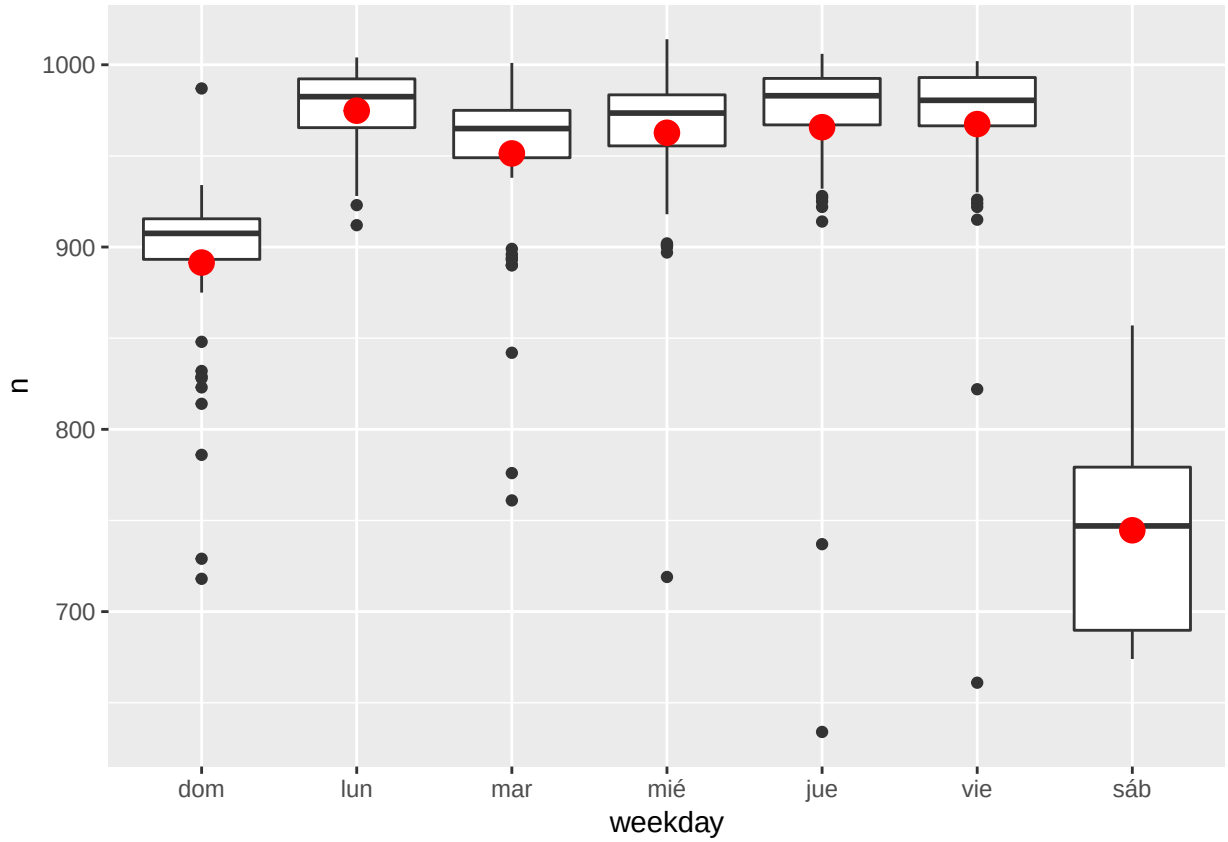
```r
daily <- daily %>%
  mutate(weekday = wday(date, label = TRUE))

ggplot(daily, aes(x = weekday, y = n)) +
  geom_boxplot()
```



Lo que vemos es un patrón donde hay preferencia por viajar entre semana. Esto es más pronunciado los sábados.

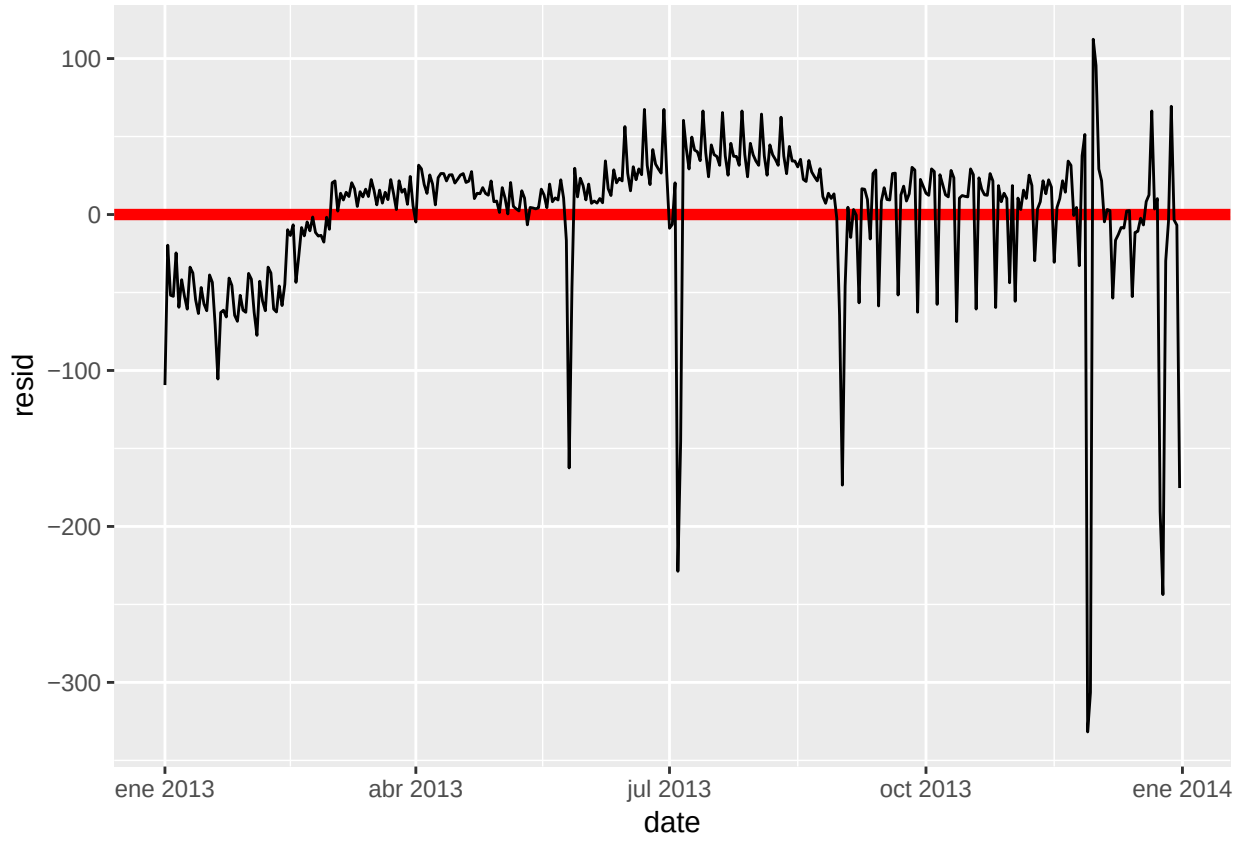
Una manera de remover este efecto es usando un modelo. Primero ajustamos un modelo y desplegamos las predicciones superpuestas en los datos originales.

```
mod <- lm(n ~ weekday, data = daily)

grid <- daily %>%
  data_grid(weekday) %>%
  add_predictions(mod, "n")

ggplot(daily, aes(weekday, n)) +
  geom_boxplot() +
  geom_point(data = grid, color = "red", size = 4)
```



Ahora vamos a computar y plotear los residuales

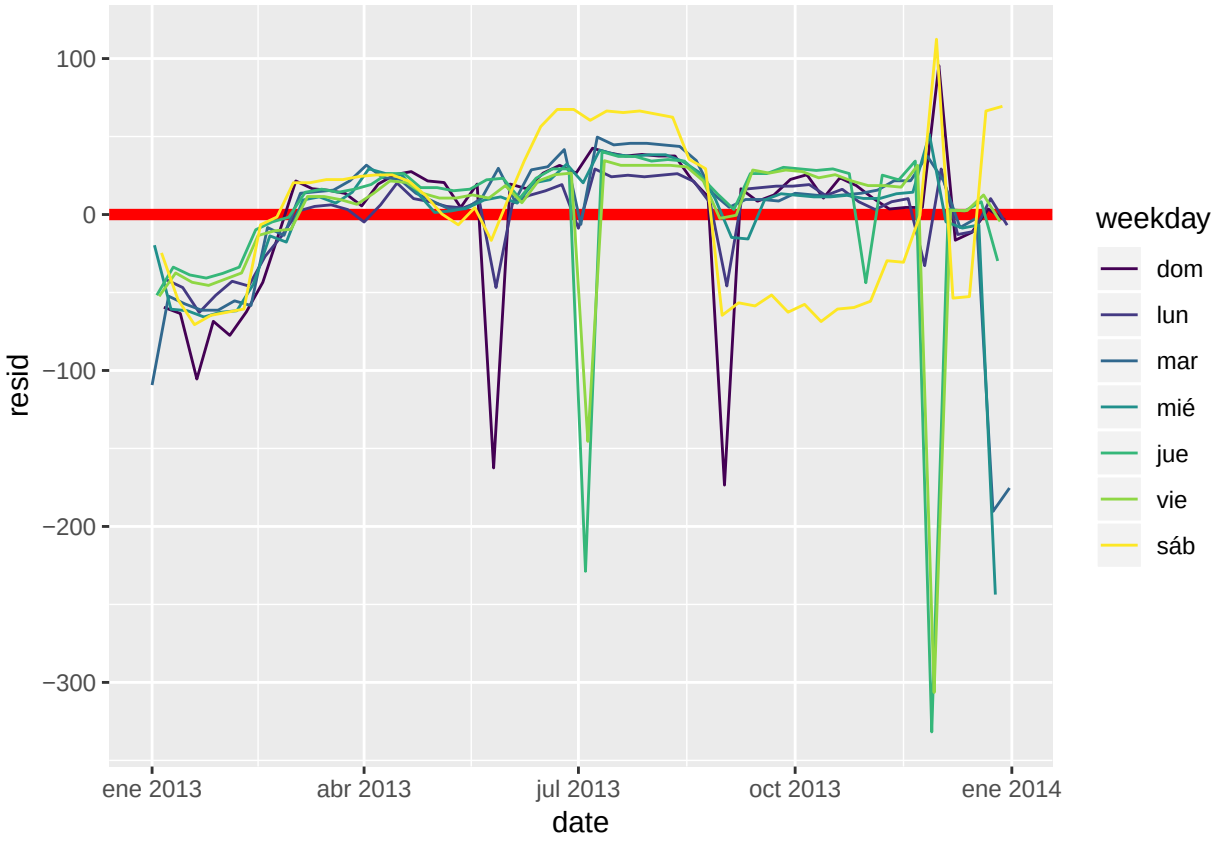
```
daily <- daily %>%
  add_residuals(mod)

daily %>%
  ggplot(aes(x = date, y = resid)) +
  geom_ref_line(h = 0, colour = "red") +
  geom_line()
```



NOtar el cambio en el eje y: ahora vemos la desviación de la cantidad de vuelos esperados dado un día de la semana. Este grafico es util porque hemos removido mucha de la variación del efecto del fin de semana.

Aún así nuestro modelo pare fallar a inicios de junio, vemos un patrón muy fuerte que nuestro modelo no logra capturar. Dibujando una línea por día de la semana nos facilitará visualizarlo

```
ggplot(daily, aes(x = date, y = resid, color = weekday)) +
  geom_ref_line(h = 0, colour = "red") +
  geom_line() +
  scale_fill_viridis_d()
```


100
0
−100
−200
−300
resid
ene 2013
abr 2013
jul 2013
oct 2013
ene 2014
date
weekday
dom
lun
mar
mié
jue
vie
sáb