

Master en Big Data. Fundamentos Matemáticos del Análisis de Datos (FMAD).

Tarea 2

Suero, Jesús

Curso 2021-22. Última actualización: 2021-09-24

Instrucciones preliminares

- Empieza abriendo el proyecto de RStudio correspondiente a tu repositorio personal de la asignatura.
- En todas las tareas tendrás que repetir un proceso como el descrito en la sección *Repite los pasos Creando un fichero Rmarkdown para esta práctica* de la *Práctica00*. Puedes releer la sección *Practicando la entrega de las Tareas* de esa misma práctica para recordar el procedimiento de entrega.

```
library(tidyverse)

## -- Attaching packages ----- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr    0.3.4
## v tibble   3.1.4     v dplyr    1.0.7
## v tidyr    1.1.3     v stringr  1.4.0
## v readr    2.0.1     vforcats 0.5.1

## -- Conflicts ----- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()   masks stats::lag()
```

Ejercicio 1. Simulando variables aleatorias discretas.

Apartado 1: La variable aleatoria discreta X_1 tiene esta tabla de densidad de probabilidad (es la variable que se usa como ejemplo en la Sesión):

valor de X_1	0	1	2	3
Probabilidad de ese valor $P(X = x_i)$	$\frac{64}{125}$	$\frac{48}{125}$	$\frac{12}{125}$	$\frac{1}{125}$

Calcula la media y la varianza teóricas de esta variable.

Respuesta:

```
x1 = c(0:3)
p1 = c(64/125, 48/125, 12/125, 1/125)
(media_1 = sum(x1*p1))
```

```
## [1] 0.6
```

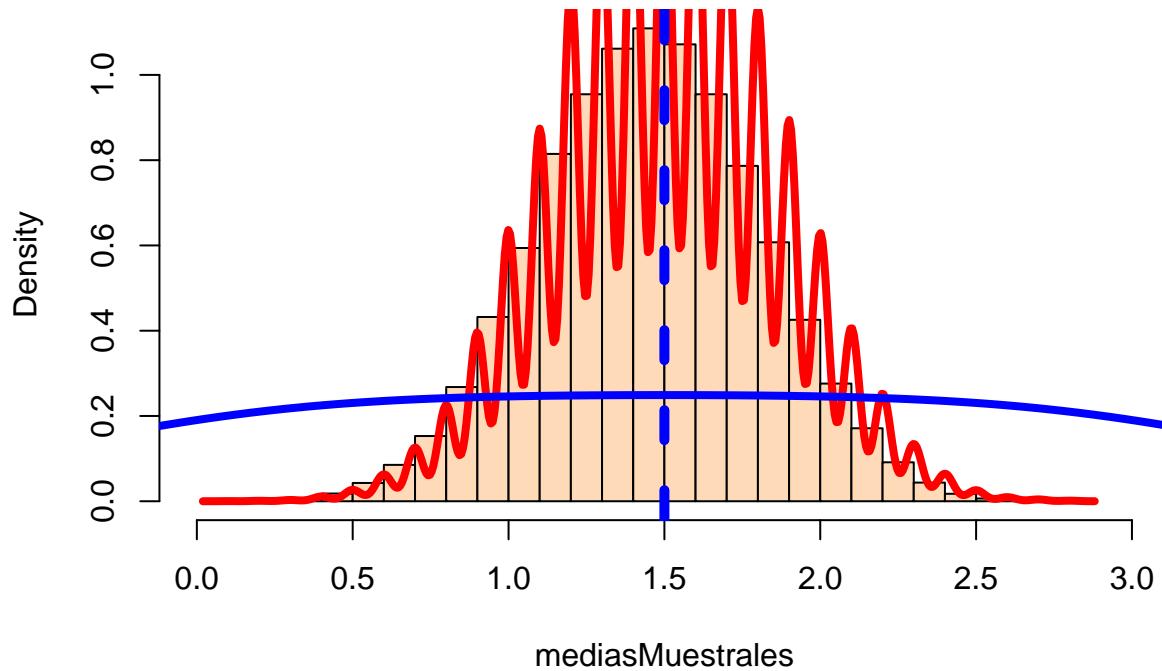
```
(varianza_1 = sum((x1-media_1)^2*p1))
```

```
## [1] 0.48
```

Apartado 2: Combina `sample` con `replicate` para simular cien mil muestras de tamaño 10 de esta variable X_1 . Estudia la distribución de las medias muestrales como hemos hecho en ejemplos previos, ilustrando con gráficas la distribución de esas medias muestrales. Cambia después el tamaño de la muestra a 30 y repite el análisis.

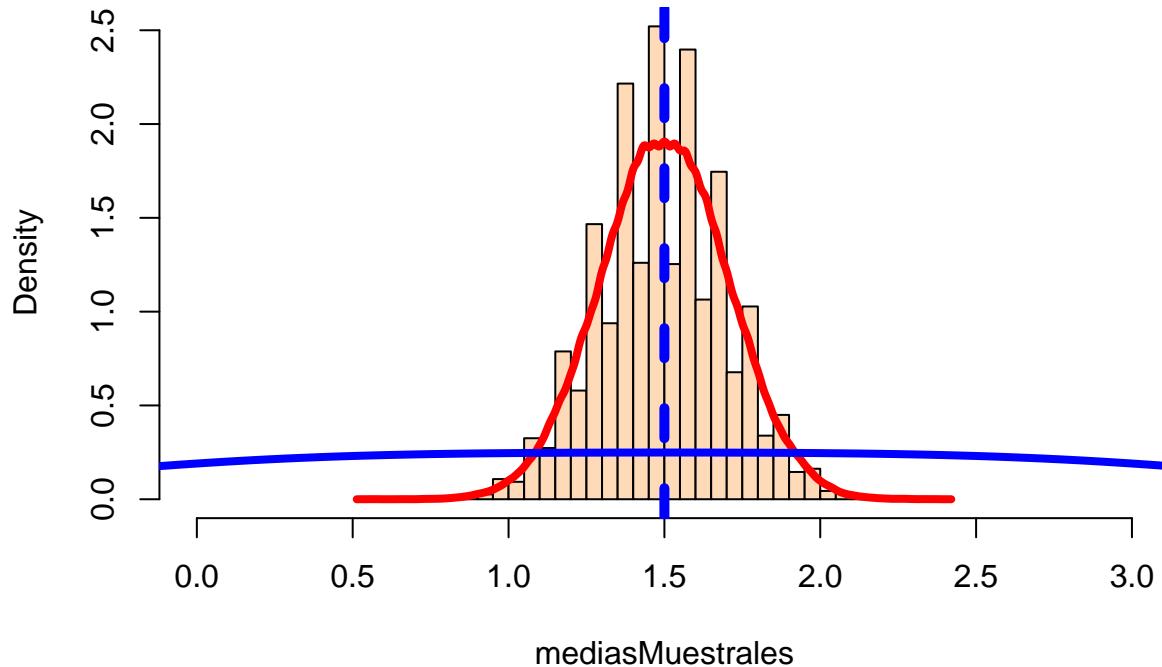
Respuesta: Análisis para tamaño 10:

```
k = 100000
mediasMuestrales = replicate(k, {
  muestra = sample(x1, 10, replace = TRUE)
  mean(muestra)
})
hist(mediasMuestrales, breaks = 20, main="",
  col="peachpuff", probability = TRUE, xlim=range(x1))
lines(density(mediasMuestrales), lwd=4, col="red")
lines(density(x1), lwd=4, col="blue")
abline(v = mean(x1), lty=2, lwd=5, col="blue")
```



Análisis para tamaño 30:

```
k = 100000
mediasMuestrales = replicate(k, {
  muestra = sample(x1, 30, replace = TRUE)
  mean(muestra)
})
hist(mediasMuestrales, breaks = 30, main="",
  col="peachpuff", probability = TRUE, xlim=range(x1))
lines(density(mediasMuestrales), lwd=4, col="red")
lines(density(x1), lwd=4, col="blue")
abline(v = mean(x1), lty=2, lwd=5, col="blue")
```



Apartado 3: La variable aleatoria discreta X_2 tiene esta tabla de densidad de probabilidad:

valor de X_2	0	1	2
Probabilidad de ese valor $P(X = x_i)$	$\frac{1}{2}$	$\frac{1}{4}$	$\frac{1}{4}$

Suponemos que X_1 y X_2 son independientes. ¿Qué valores puede tomar la suma $X_1 + X_2$? ¿Cuál es su tabla de probabilidad?

Respuesta:

```
x2 = c(0:2, each=4)
p2 = c(1/4, 1/4, 1/4, 1/4)
p1_2 = c(64/125, 48/125, 12/125, 1/125)*rep(c(1/2,1/4,1/4), each = 4)
suma = x1 + x2
tabla = data.frame(x1, x2, p1_2, suma)

tabla %>%
  group_by(suma) %>%
  summarise(sumaX1X2=sum(p1_2))
```

```
## # A tibble: 4 x 2
##       sumaX1X2
##   <dbl>     <dbl>
## 1      0     0.512
## 2      2     0.384
```

```
## 3      4     0.096
## 4      7     0.008
```

Apartado 4: Calcula la media teórica de la suma $X_1 + X_2$. Después usa `sample` y `replicate` para simular cien mil *valores* de esta variable suma. Calcula la media de esos valores. *Advertencia:* no es el mismo tipo de análisis que hemos hecho en el segundo apartado.

Respuesta: Para la media de la suma X1 + X2:

```
media_2 = sum(x2*p2)
(media_suma = media_1 + media_2)
```

```
## [1] 2.35
```

Calculamos ahora la media de las 100000 simulaciones de la variable suma:

```
k = 100000
mediasMuestrales = replicate(k, {
  muestra = sample(media_suma, 10, replace = TRUE)
  mean(muestra)
})
mean(mediasMuestrales)
```

```
## [1] 1.500064
```

Ejercicio 2. Datos limpios

- Descarga el fichero de este enlace

<https://gist.github.com/fernandosansegundo/471b4887737cfcec7e9cf28631f2e21e/raw/b3944599d02df494f5903740/testResults.csv>

```
fichero = read_csv("testResults.csv")

## Rows: 200 Columns: 9

## -- Column specification --
## Delimiter: ","
## chr (2): name, gender_age
## dbl (7): id, test_number, week1, week2, week3, week4, week5

##
## 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.
```

- Este fichero contiene las notas de los alumnos de una clase, que hicieron dos tests cada semana durante cinco semanas. La tabla de datos no cumple los principios de *tidy data* que hemos visto en clase. Tu tarea en este ejercicio es explicar por qué no se cumplen y obtener una tabla de datos limpios con la misma información usando *tidyR*.

Indicación: lee la ayuda de la función `separate` de *tidyR*.

Respuesta: Vemos que aspecto tiene la tabla de datos del fichero:

```

fichero %>%
  head()

## # A tibble: 6 x 9
##   name      id gender_age test_number week1 week2 week3 week4 week5
##   <chr>    <dbl> <chr>          <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 Jacob     108 m_20           1     8     5     7     5     6
## 2 Jacob     108 m_20           2     2     2     4     0     3
## 3 Michael   490 m_19           1    10     0     5     4     0
## 4 Michael   490 m_19           2     9    10     8    10     9
## 5 Matthew   424 m_18           1     6     0     0     1    10
## 6 Matthew   424 m_18           2     3     4     2     5     8

```

Los valores week1:week5 hacen que el conjunto de datos no sea limpio, estas columnas no son realmente variables sino valores de una variable. Convertimos los valores de week a una sola columna. Además, la columna gender_age contiene dos variables, la dividimos en dos columnas, cada una con una variable.

```

ficheroTidy = fichero %>%
  pivot_longer(week1:week5, names_to = "week") %>%
  separate(gender_age, into = c("gender", "age"))
ficheroTidy %>%
  head(10)

```

```

## # A tibble: 10 x 7
##   name      id gender age  test_number week  value
##   <chr>    <dbl> <chr> <chr>          <dbl> <chr> <dbl>
## 1 Jacob     108 m     20    1 week1     8
## 2 Jacob     108 m     20    1 week2     5
## 3 Jacob     108 m     20    1 week3     7
## 4 Jacob     108 m     20    1 week4     5
## 5 Jacob     108 m     20    1 week5     6
## 6 Jacob     108 m     20    2 week1     2
## 7 Jacob     108 m     20    2 week2     2
## 8 Jacob     108 m     20    2 week3     4
## 9 Jacob     108 m     20    2 week4     0
## 10 Jacob    108 m     20   2 week5     3

```

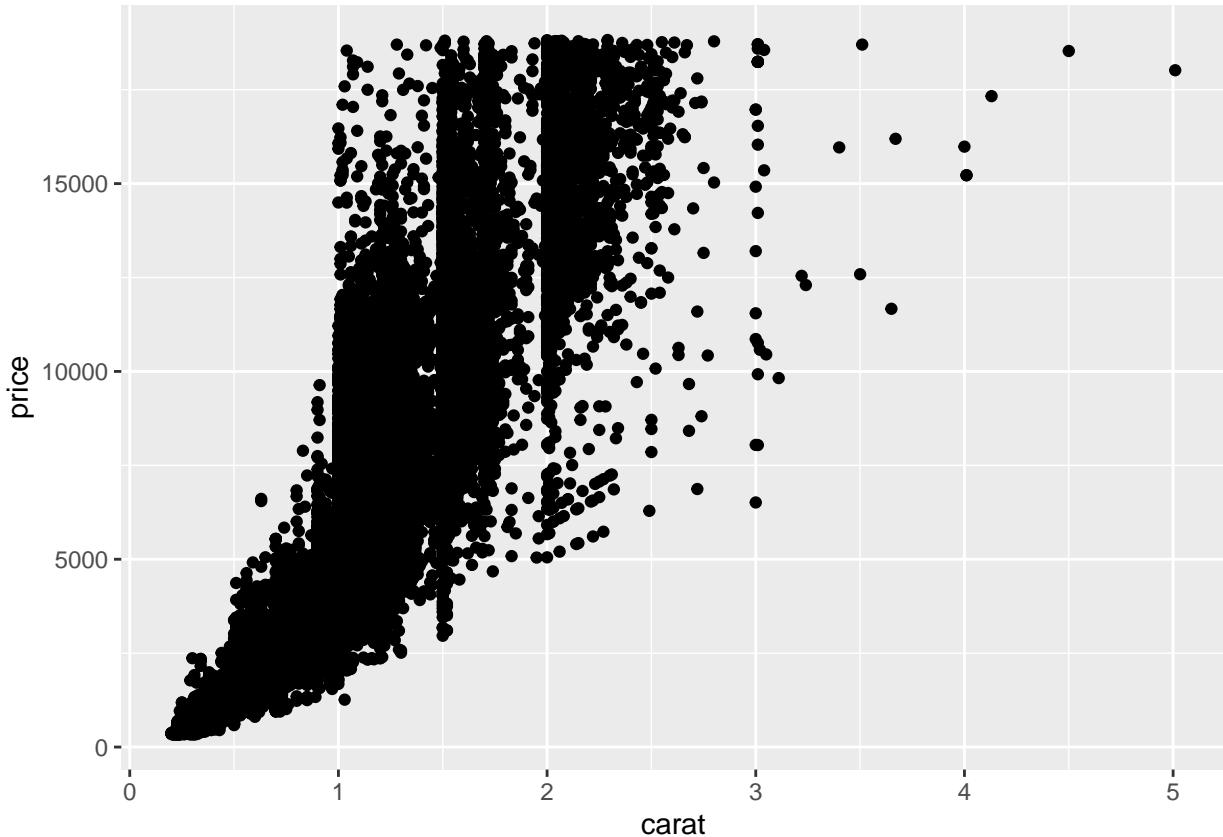
Ejercicio 3. Lectura de R4DS.

Continuando con nuestra *lectura conjunta* de este libro, si revisas el índice verás que hemos cubierto (holgadamente en algún caso) el contenido de los Capítulos 6, 8, 9, 10 y 11. Todos esos Capítulos son relativamente ligeros. Por eso esta semana conviene detenerse un poco en la lectura de los Capítulos 7 y 12, que son los más densos en información. Y como motivación os proponemos un par de ejercicios, uno por cada uno de esos capítulos.

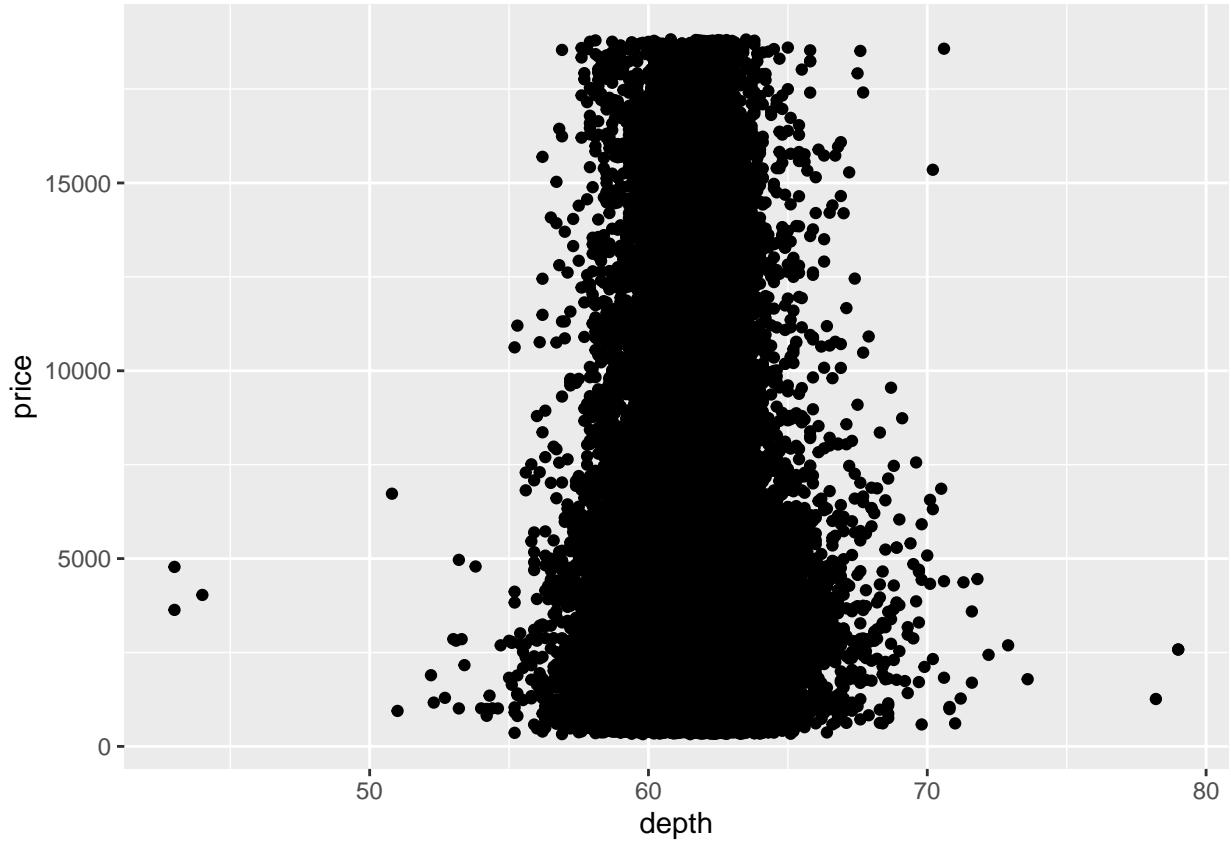
- Haz el ejercicio 2 de la Sección 7.5.1.1 de R4DS. Las ideas de esa sección son importantes para nuestro trabajo de las próximas sesiones.
- What variable in the diamonds dataset is most important for predicting the price of a diamond? How is that variable correlated with cut? Why does the combination of those two relationships lead to lower quality diamonds being more expensive?

Respuesta: Para ver la variable más útil para predecir el precio representamos cada variable respecto al precio. Habrá que fijarse en que cuando esa variable cambie de valor el precio lo haga también.

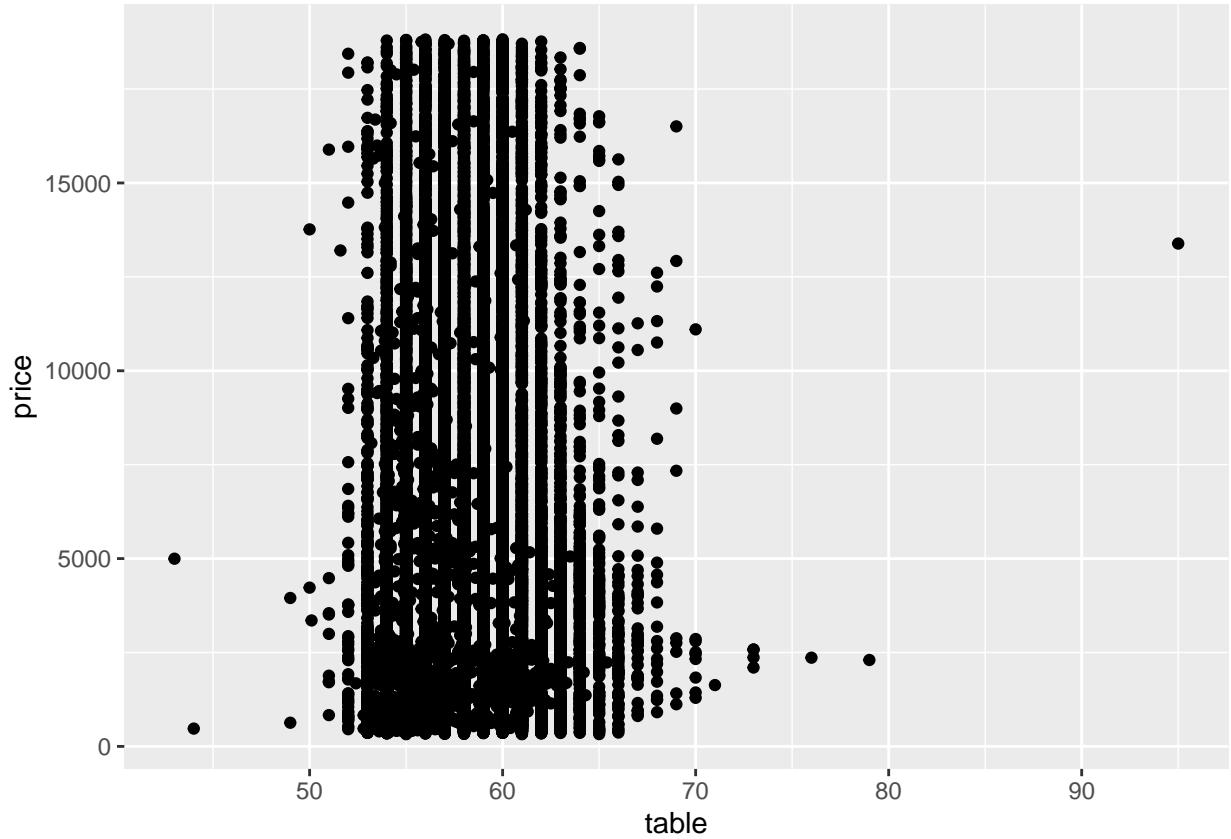
```
ggplot(data = diamonds, mapping = aes(x = carat, y = price)) +  
  geom_point()
```



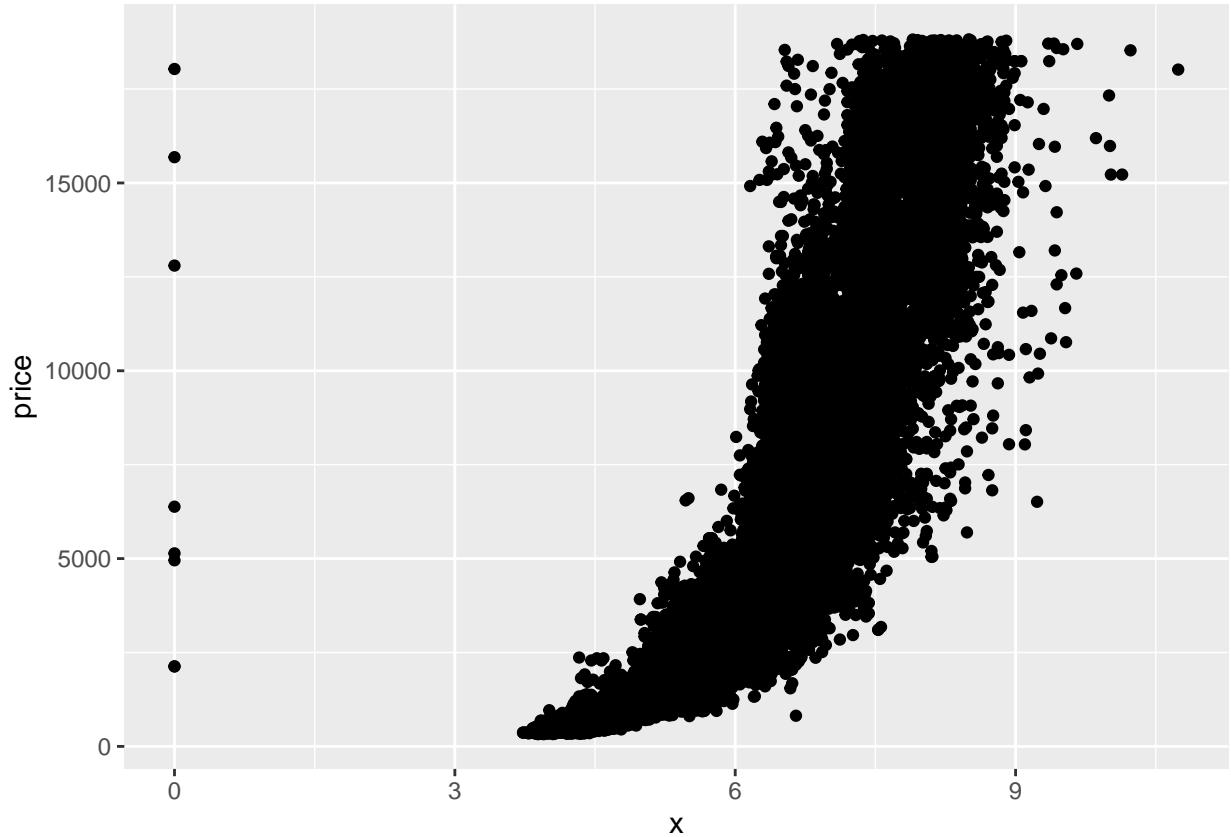
```
ggplot(data = diamonds, mapping = aes(x = depth, y = price)) +  
  geom_point()
```



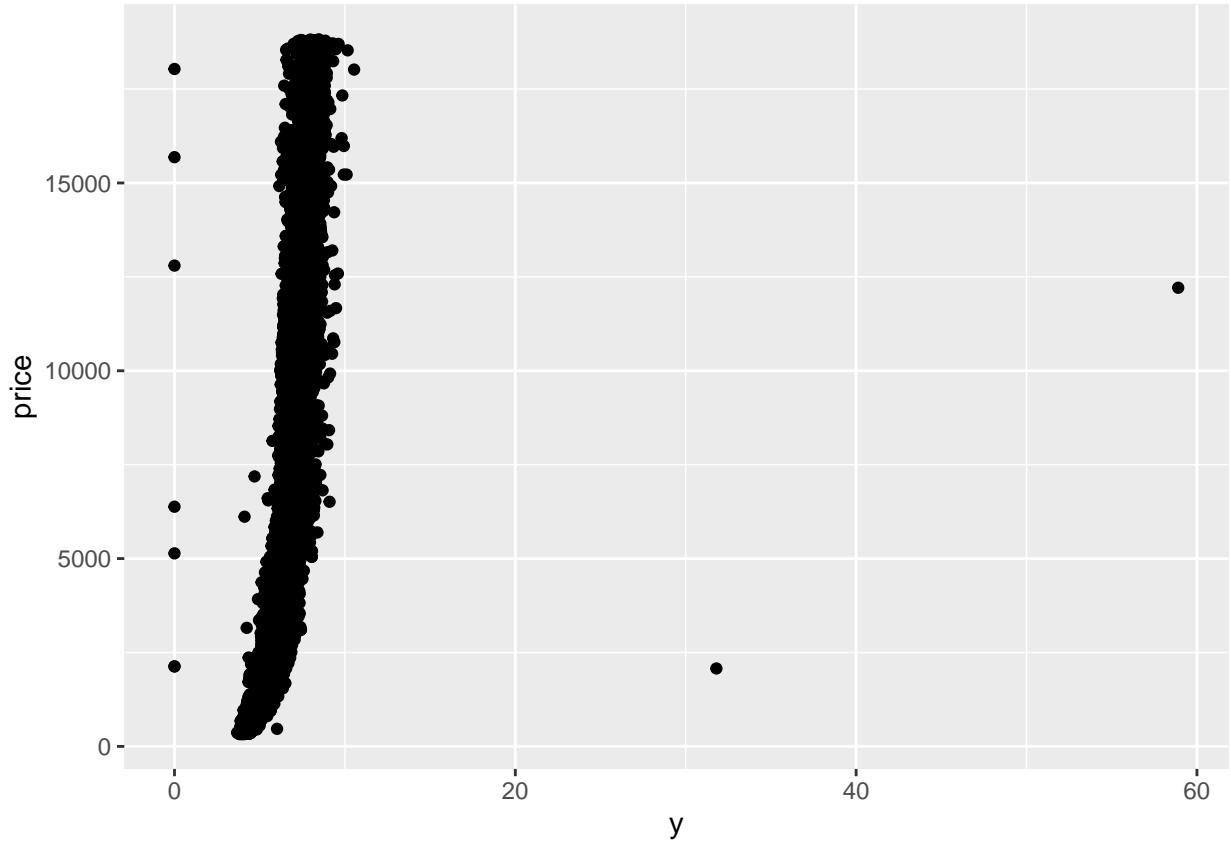
```
ggplot(data = diamonds, mapping = aes(x = table, y = price)) +  
  geom_point()
```



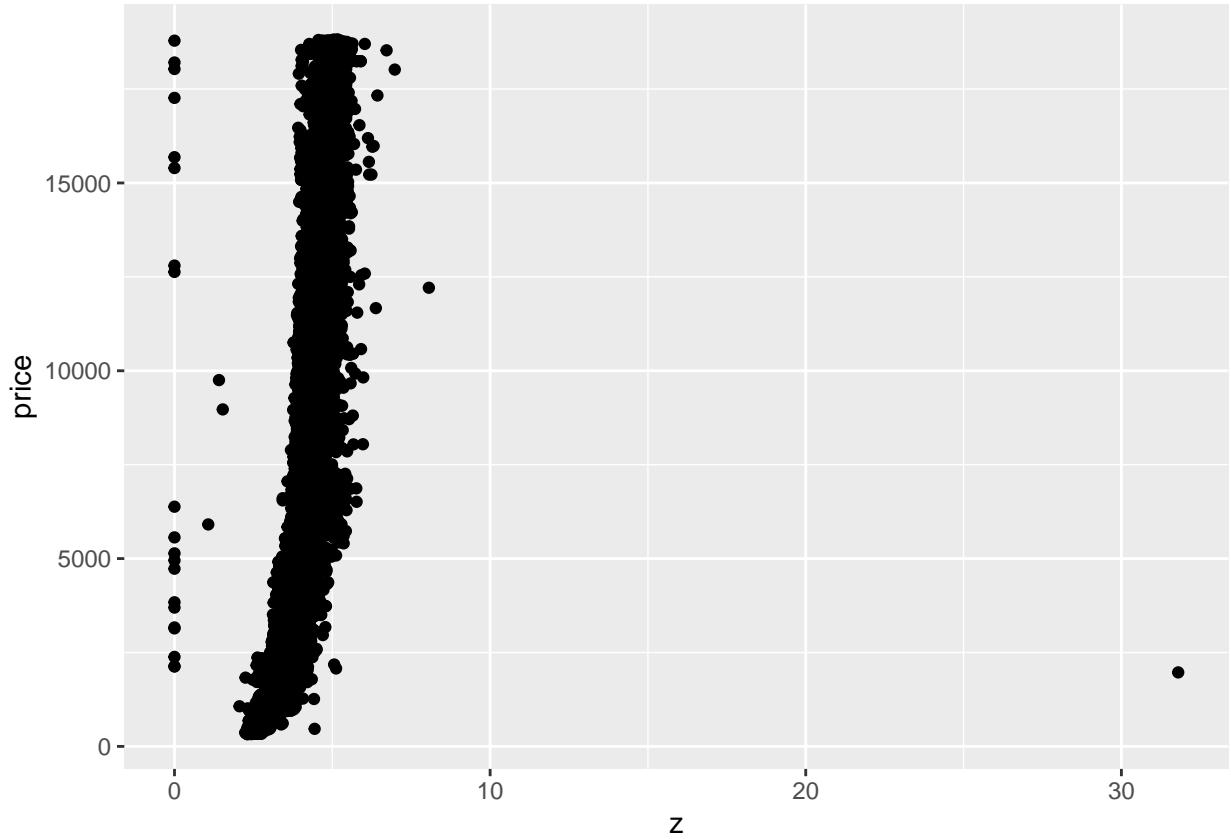
```
ggplot(data = diamonds, mapping = aes(x = x, y = price)) +  
  geom_point()
```



```
ggplot(data = diamonds, mapping = aes(x = y, y = price)) +  
  geom_point()
```



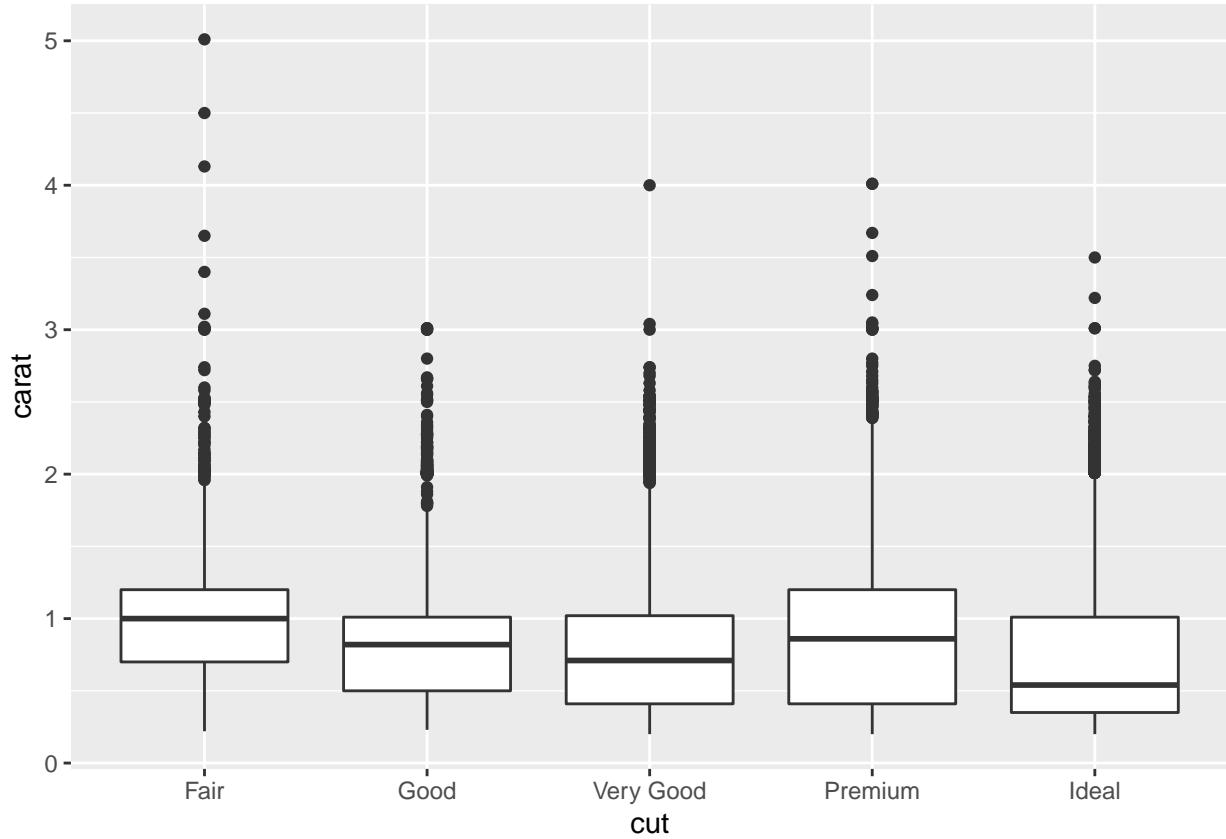
```
ggplot(data = diamonds, mapping = aes(x = z, y = price)) +  
  geom_point()
```



Las variables depth y table no influyen para predecir el precio. Las variables que sí influyen y además de forma parecida son carat (peso/kilates) y las variables x,y,z correspondientes al volumen del diamante. Consideramos que carat es la variable más importante para predecir el precio.

Ahora vemos como carat está correlacionada con cut:

```
ggplot(data = diamonds, mapping = aes(x = cut, y = carat)) +
  geom_boxplot()
```



Observamos que cuanto mejor es el cut más bajo es el carat. Los diamantes con mejor cut son más pequeños y por tanto más baratos. Por eso ocurre que si miramos únicamente la variable cut para predecir el precio vemos que a mejor cut menor precio.

- Haz el ejercicio 4 de la Sección 12.6.1 de R4DS. ¡Aprovecha el código previo de esa sección para trabajar con datos limpios!
- For each country, year, and sex compute the total number of cases of TB. Make an informative visualisation of the data.

Respuesta: Usaremos el conjunto de datos who. Con el siguiente fragmento de código obtenido de la sección obtenemos el conjunto de datos limpio (tidy).

```
whoTidy = who %>%
  pivot_longer(
    cols = new_sp_m014:newrel_f65,
    names_to = "key",
    values_to = "cases",
    values_drop_na = TRUE
  ) %>%
  mutate(
    key = stringr::str_replace(key, "newrel", "new_rel")
  ) %>%
  separate(key, c("new", "var", "sexage")) %>%
  select(-new, -iso2, -iso3) %>%
  separate(sexage, c("sex", "age"), sep = 1)
```

Calculamos los casos totales para cada país y mostramos en diferentes gráficas (una por país) la evolución de los casos a lo largo de los años según el sexo. Mostramos únicamente aquellos países con más de 500000 casos:

```
whoTidy %>%
  group_by(country) %>%
  mutate(country_cases = sum(cases)) %>%
  filter(country_cases > 500000) %>%
  group_by(country, year, sex) %>%
  count(wt = cases) %>%
  ggplot(aes(x = year, y = n, colour = sex)) +
  geom_line() +
  facet_wrap(~country)
```

