Importo archivos

```
library(bayesrules)
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(rstanarm)
```

```
## Cargando paquete requerido: Rcpp
## This is rstanarm version 2.32.1
## - See https://mc-stan.org/rstanarm/articles/priors for changes to default priors!
## - Default priors may change, so it's safest to specify priors, even if equivalent to the defaults.
## - For execution on a local, multicore CPU with excess RAM we recommend calling
##   options(mc.cores = parallel::detectCores())
```

```
library(broom.mixed)
library(brms)
```

```
## Loading 'brms' package (version 2.22.0). Useful instructions
## can be found by typing help('brms'). A more detailed introduction
## to the package is available through vignette('brms_overview').
##
## Adjuntando el paquete: 'brms'
##
## The following objects are masked from 'package:rstanarm':
##
##     dirichlet, exponential, get_y, lasso, ngrps
##
## The following object is masked from 'package:stats':
##
##     ar
```

Importo nuestro dataset

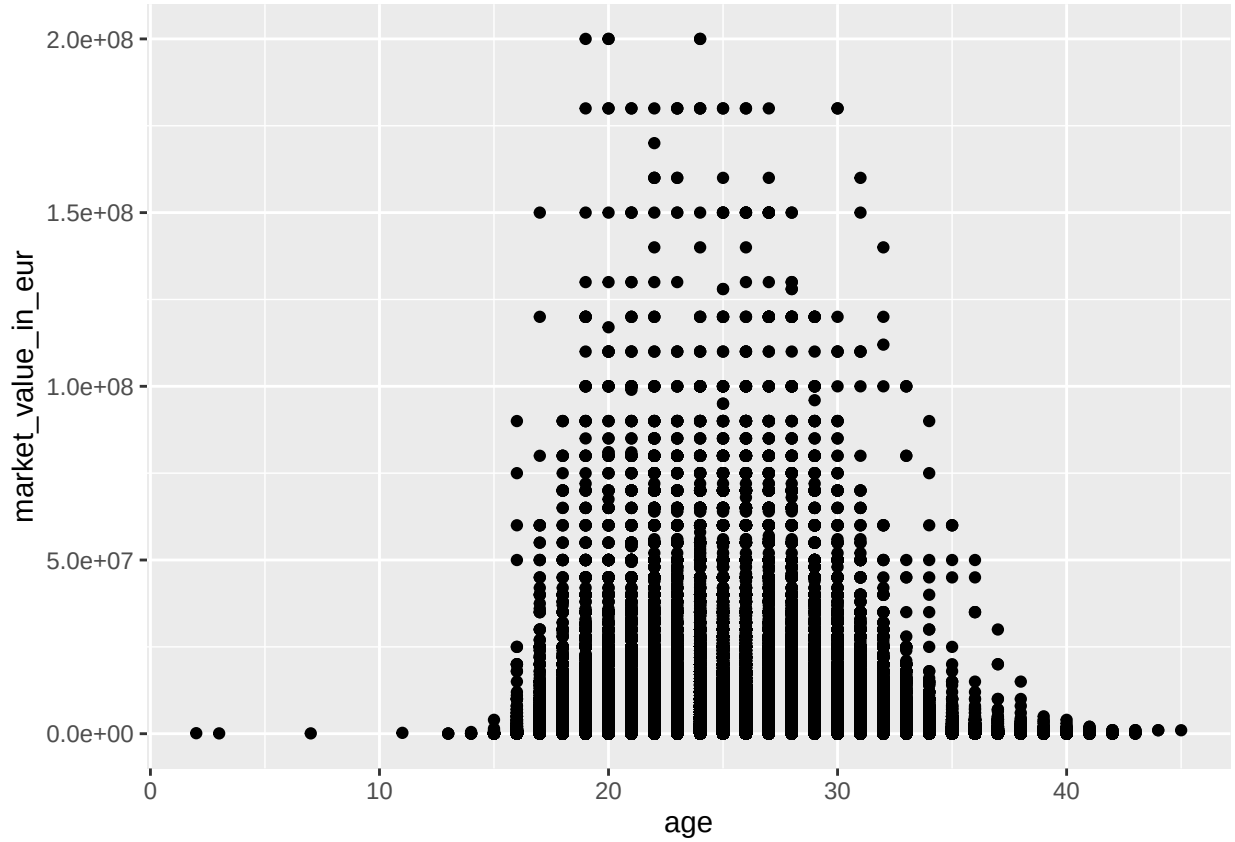
```
#import csv from file called players_age_valuation.csv
players_age_valuation <- read.csv("players_age_valuation.csv")
```

```
# Print header of dataset
head(players_age_valuation)
```

```
##   player_id           name age market_value_in_eur
## 1        10 Miroslav Klose  26              7000000
## 2        10 Miroslav Klose  26              9000000
## 3        10 Miroslav Klose  26             12000000
## 4        10 Miroslav Klose  27             15000000
## 5        10 Miroslav Klose  27             20000000
## 6        10 Miroslav Klose  28             30000000
```

Visualicemos un poco la data con la que trabajaremos.

```
ggplot(players_age_valuation, aes(x = age, y = market_value_in_eur)) +
geom_point()
```




```
# Nuestros priors seran normales con media 0 y desviacion estandar de 2.5
# Usaremos un modelo de regresion lineal con una distribucion gaussiana
# Usaremos el metodo de muestreo de Markov Chain Monte Carlo (MCMC) para estimar los parametros del mod
# Usaremos 4 cadenas y 10000 iteraciones
complete_pooled_model <- stan_glm(
  market_value_in_eur ~ age,      # Comparo el valor de mercado con la edad
  data = players_age_valuation,
  family = gaussian,
  prior_intercept = normal(0, 2.5, autoscale = TRUE),  # Priors para la intersección
  prior = normal(0, 2.5, autoscale = TRUE),           # Priors para la pendiente
  prior_aux = cauchy(0, 1),                           # Priors para sigma (Half-Cauchy)
  chains = 4,
  iter = 1000, seed = 84735)
```

```
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 9.2e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.92 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:   1 / 1000 [  0%]  (Warmup)
## Chain 1: Iteration: 100 / 1000 [ 10%]  (Warmup)
## Chain 1: Iteration: 200 / 1000 [ 20%]  (Warmup)
## Chain 1: Iteration: 300 / 1000 [ 30%]  (Warmup)
## Chain 1: Iteration: 400 / 1000 [ 40%]  (Warmup)
## Chain 1: Iteration: 500 / 1000 [ 50%]  (Warmup)
## Chain 1: Iteration: 501 / 1000 [ 50%]  (Sampling)
## Chain 1: Iteration: 600 / 1000 [ 60%]  (Sampling)
## Chain 1: Iteration: 700 / 1000 [ 70%]  (Sampling)
## Chain 1: Iteration: 800 / 1000 [ 80%]  (Sampling)
## Chain 1: Iteration: 900 / 1000 [ 90%]  (Sampling)
## Chain 1: Iteration: 1000 / 1000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 1.499 seconds (Warm-up)
## Chain 1:                19.23 seconds (Sampling)
## Chain 1:                20.729 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 3.5e-05 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.35 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:   1 / 1000 [  0%]  (Warmup)
## Chain 2: Iteration: 100 / 1000 [ 10%]  (Warmup)
## Chain 2: Iteration: 200 / 1000 [ 20%]  (Warmup)
## Chain 2: Iteration: 300 / 1000 [ 30%]  (Warmup)
## Chain 2: Iteration: 400 / 1000 [ 40%]  (Warmup)
## Chain 2: Iteration: 500 / 1000 [ 50%]  (Warmup)
## Chain 2: Iteration: 501 / 1000 [ 50%]  (Sampling)
## Chain 2: Iteration: 600 / 1000 [ 60%]  (Sampling)
## Chain 2: Iteration: 700 / 1000 [ 70%]  (Sampling)
## Chain 2: Iteration: 800 / 1000 [ 80%]  (Sampling)
## Chain 2: Iteration: 900 / 1000 [ 90%]  (Sampling)
## Chain 2: Iteration: 1000 / 1000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 2.557 seconds (Warm-up)
## Chain 2:                19.691 seconds (Sampling)
## Chain 2:                22.248 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 2.5e-05 seconds
```

```
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.25 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:   1 / 1000 [  0%]  (Warmup)
## Chain 3: Iteration: 100 / 1000 [ 10%]  (Warmup)
## Chain 3: Iteration: 200 / 1000 [ 20%]  (Warmup)
## Chain 3: Iteration: 300 / 1000 [ 30%]  (Warmup)
## Chain 3: Iteration: 400 / 1000 [ 40%]  (Warmup)
## Chain 3: Iteration: 500 / 1000 [ 50%]  (Warmup)
## Chain 3: Iteration: 501 / 1000 [ 50%]  (Sampling)
## Chain 3: Iteration: 600 / 1000 [ 60%]  (Sampling)
## Chain 3: Iteration: 700 / 1000 [ 70%]  (Sampling)
## Chain 3: Iteration: 800 / 1000 [ 80%]  (Sampling)
## Chain 3: Iteration: 900 / 1000 [ 90%]  (Sampling)
## Chain 3: Iteration: 1000 / 1000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 3.024 seconds (Warm-up)
## Chain 3:                19.271 seconds (Sampling)
## Chain 3:                22.295 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 2.6e-05 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.26 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:   1 / 1000 [  0%]  (Warmup)
## Chain 4: Iteration: 100 / 1000 [ 10%]  (Warmup)
## Chain 4: Iteration: 200 / 1000 [ 20%]  (Warmup)
## Chain 4: Iteration: 300 / 1000 [ 30%]  (Warmup)
## Chain 4: Iteration: 400 / 1000 [ 40%]  (Warmup)
## Chain 4: Iteration: 500 / 1000 [ 50%]  (Warmup)
## Chain 4: Iteration: 501 / 1000 [ 50%]  (Sampling)
## Chain 4: Iteration: 600 / 1000 [ 60%]  (Sampling)
## Chain 4: Iteration: 700 / 1000 [ 70%]  (Sampling)
## Chain 4: Iteration: 800 / 1000 [ 80%]  (Sampling)
## Chain 4: Iteration: 900 / 1000 [ 90%]  (Sampling)
## Chain 4: Iteration: 1000 / 1000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 3.199 seconds (Warm-up)
## Chain 4:                18.824 seconds (Sampling)
## Chain 4:                22.023 seconds (Total)
## Chain 4:
```

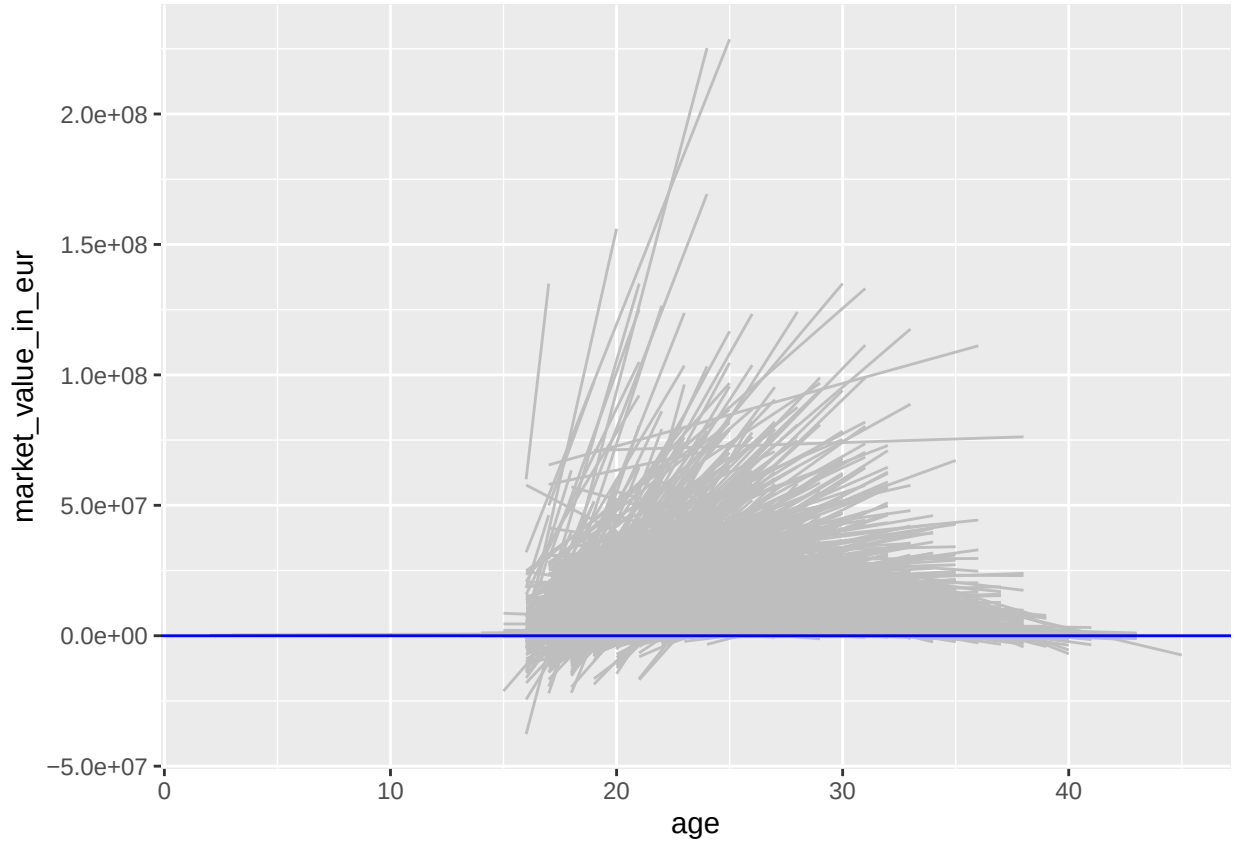
```
# Resumen del modelo
tidy(complete_pooled_model, conf.int = TRUE, conf.level = 0.80)
```

```
## # A tibble: 2 x 5
##   term        estimate std.error conf.low conf.high
##   <chr>          <dbl>     <dbl>    <dbl>     <dbl>
## 1 (Intercept) 1644247.    52704. 1578529.  1714742.
```

```
## 2 age           30613.     2064.   27739.    33120.
```

```r
# Plot of the posterior median model
ggplot(players_age_valuation, aes(x = age, y = market_value_in_eur, group = player_id)) +
  geom_smooth(method = "lm", se = FALSE, color = "gray", linewidth = 0.5) +
  geom_abline(aes(intercept = 75.2, slope = 0.268), color = "blue")
```

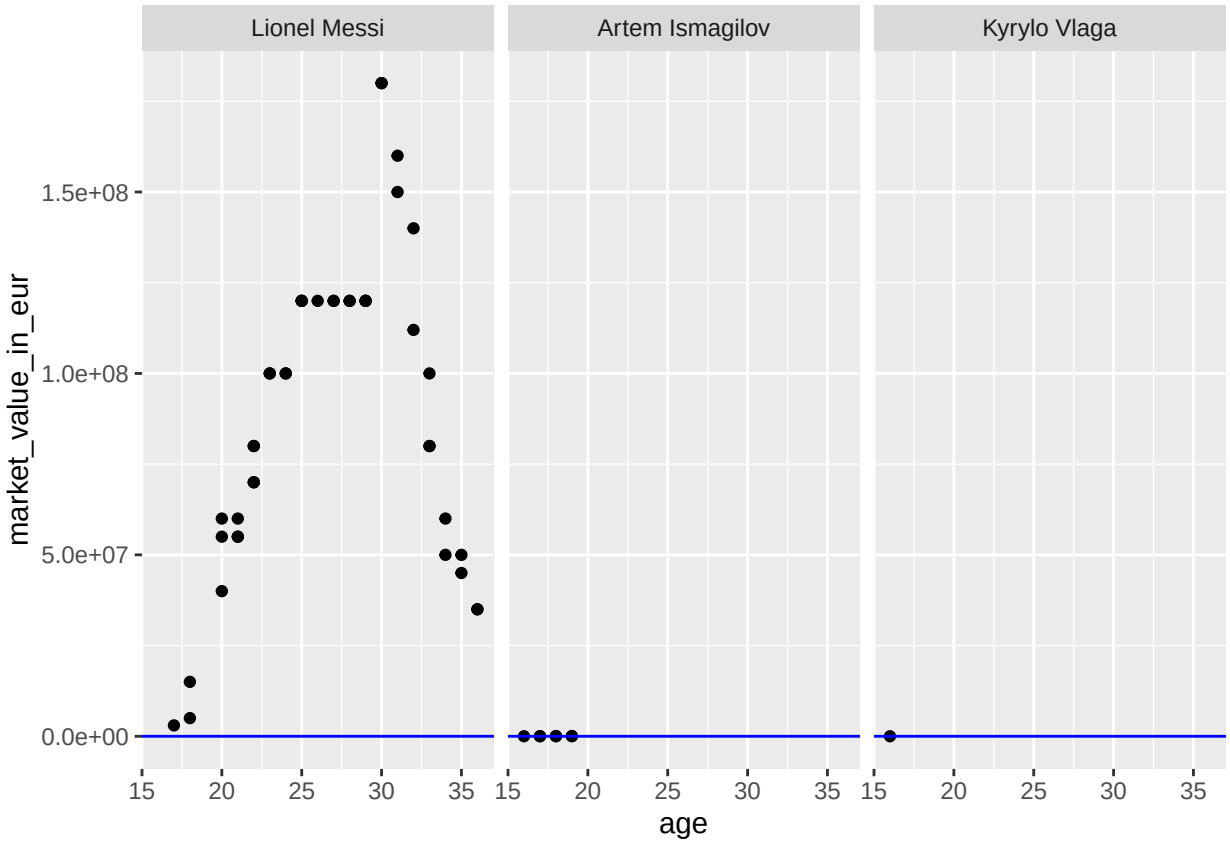
```
## `geom_smooth()` using formula = 'y ~ x'
```




```r
# Selecciono 3 jugadores randoms para comparar
examples <- players_age_valuation %>%
  filter(player_id %in% c("697853", "1120127", "28003")) # Elijo 3 player_ids randoms



ggplot(examples, aes(x = age, y = market_value_in_eur)) +
  geom_point() +
  geom_abline(aes(intercept = 75.2242, slope = 0.2678), color = "blue") +
  facet_wrap(~ player_id, labeller = labeller(player_id = setNames(examples$name, examples$player_id)))
```


Lionel Messi
Artem Ismagilov
Kyrylo Vlaga
market_value_in_eur
1.5e+08
1.0e+08
5.0e+07
0.0e+00
15
20
25
30
35
age