XAI 3

2025-05-12

Paso 1: Cargar paquetes necesarios

```
library(randomForest)

## randomForest 4.7-1.2

## Type rfNews() to see new features/changes/bug fixes.

library(pdp)
library(ggplot2)

##
## Adjuntando el paquete: 'ggplot2'

## The following object is masked from 'package:randomForest':
##
##     margin

library(dplyr)

##
## Adjuntando el paquete: 'dplyr'

## The following object is masked from 'package:randomForest':
##
##     combine

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

library(readr)
```

Paso 2: Cargar el dataset de bike rentals

```
bike_data <- read_csv("C:/Users/Elena Chirivella/OneDrive -
UPV/CARRERA/4o/edm/EDM ELEN/practicas/Practica 5 Sin sol/day.csv")

## Rows: 731 Columns: 16
## ── Column specification
────────────────────────────────────────────────────────
## Delimiter: ","
## dbl  (15): instant, season, yr, mnth, holiday, weekday, workingday,
weathers...
```

```
## date  (1): dteday
## 
## 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.
```

Paso 3: Entrenar modelo Random Forest para predecir cnt

```r
# Seleccionar variables relevantes
bike_model <- randomForest(cnt ~ instant + temp + hum + windspeed,
                           data = bike_data,
                           ntree = 300,
                           importance = TRUE)

# Ver importancia de las variables
importance(bike_model)
```

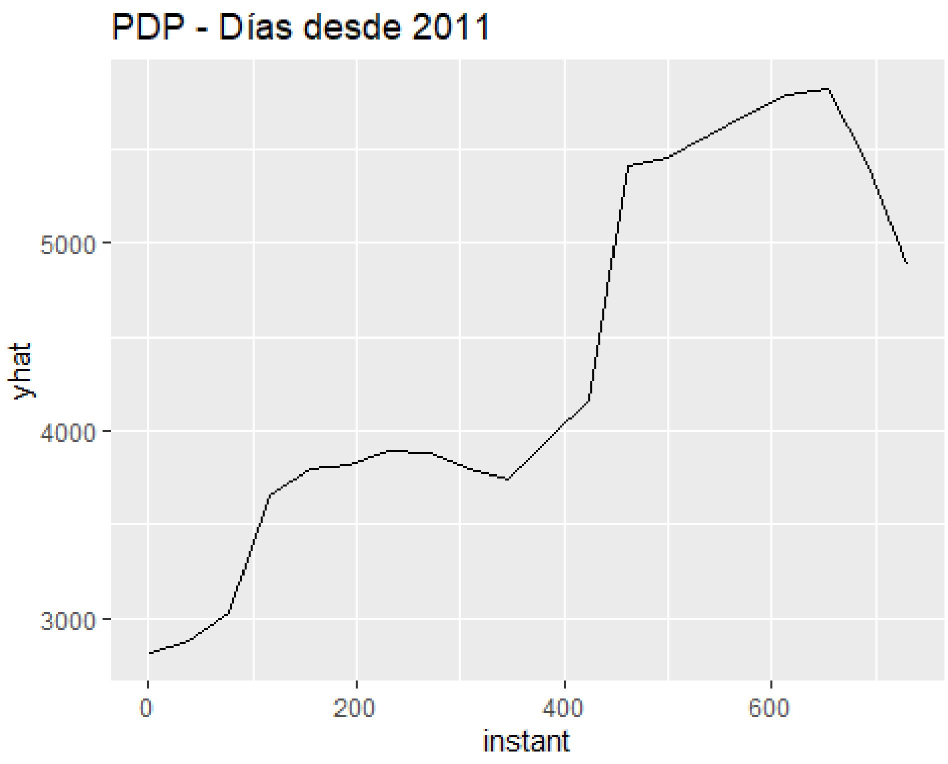
```
##            %IncMSE IncNodePurity
## instant   58.72774    1200245732
## temp      45.72647     904776047
## hum       35.54732     294644730
## windspeed 16.81891     258952414
```

Paso 4: Calcular y representar los PDP unidimensionales

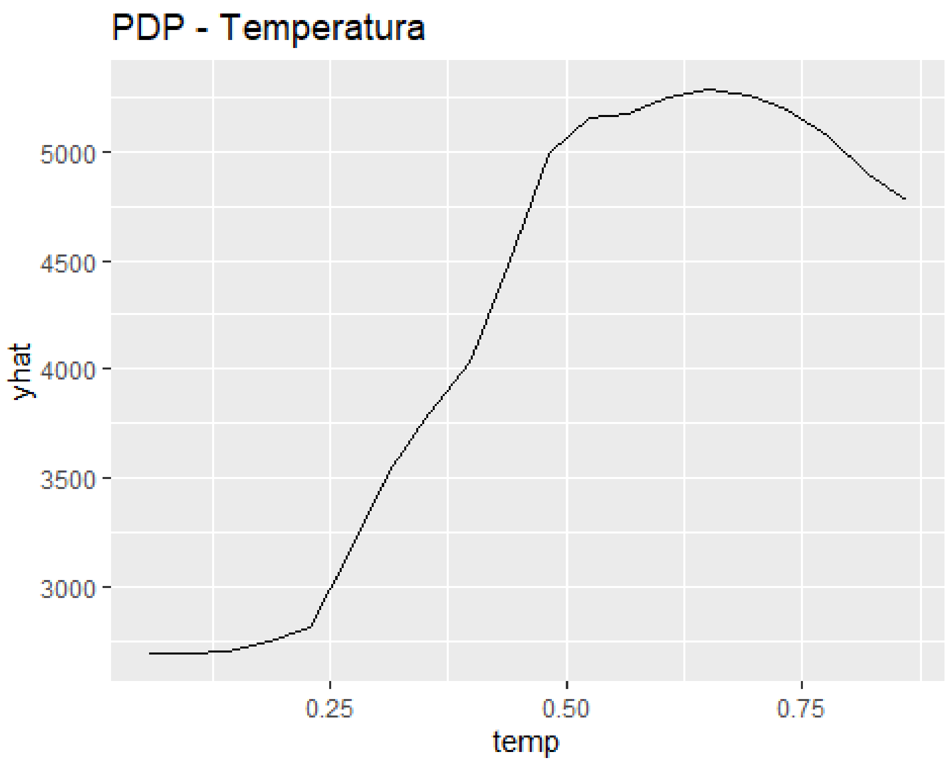
```r
# PDP para instant (días desde 2011)
pdp_instant <- partial(bike_model, pred.var = "instant", grid.resolution = 20)
plot_instant <- autoplot(pdp_instant) + ggtitle("PDP - Días desde 2011")

# PDP para temperatura
pdp_temp <- partial(bike_model, pred.var = "temp", grid.resolution = 20)
plot_temp <- autoplot(pdp_temp) + ggtitle("PDP - Temperatura")

# PDP para humedad
pdp_hum <- partial(bike_model, pred.var = "hum", grid.resolution = 20)
plot_hum <- autoplot(pdp_hum) + ggtitle("PDP - Humedad")

# PDP para velocidad del viento
pdp_wind <- partial(bike_model, pred.var = "windspeed", grid.resolution = 20)
plot_wind <- autoplot(pdp_wind) + ggtitle("PDP - Velocidad del viento")
```

Paso 5: Mostrar los gráficos

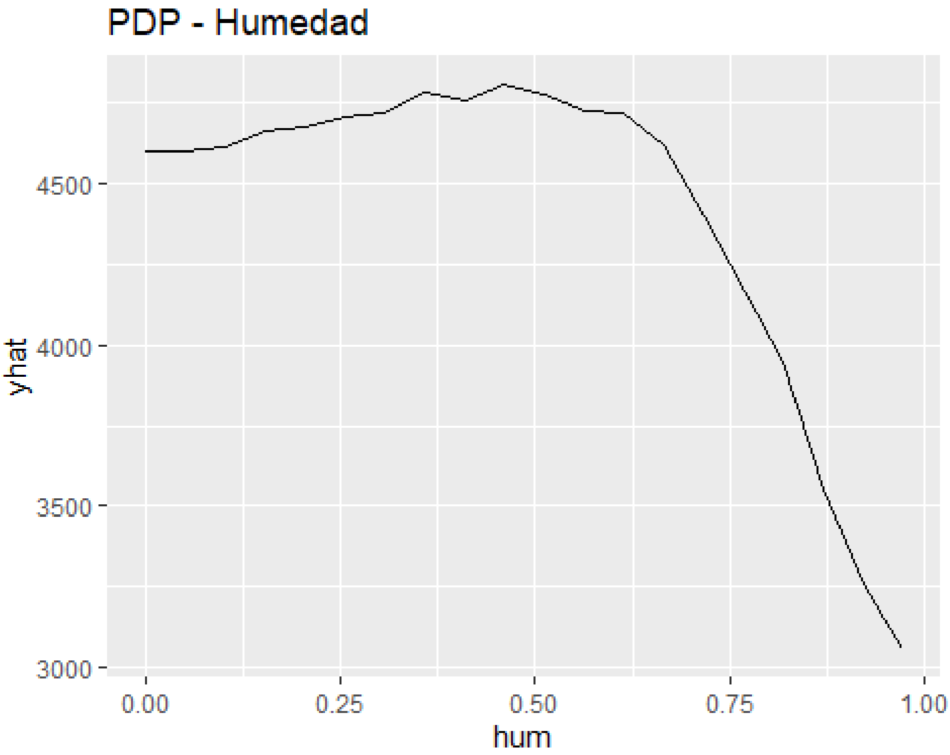
```r
# Mostrar todos los PDPs
print(plot_instant)
```



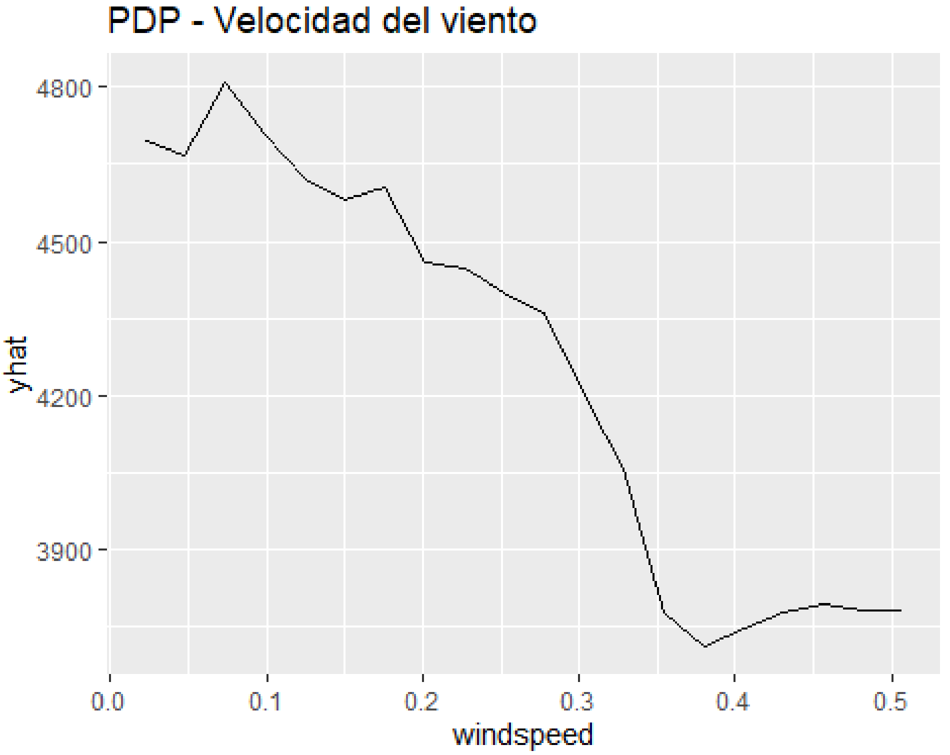
```
print(plot_temp)
```



```
print(plot_hum)
```

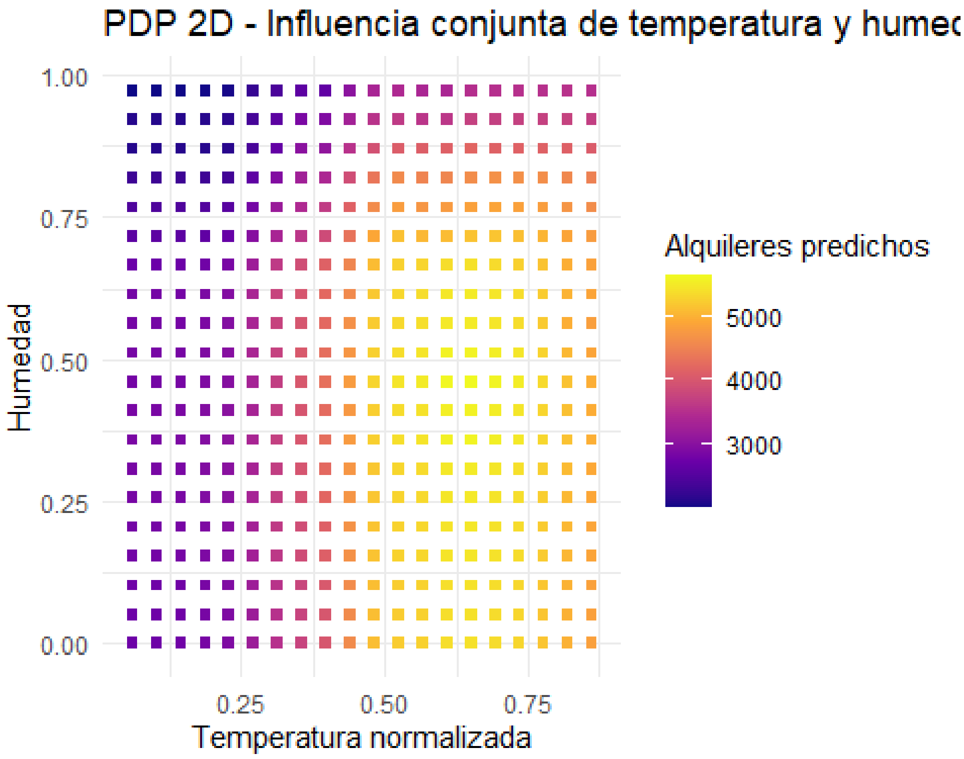


```
print(plot_wind)
```



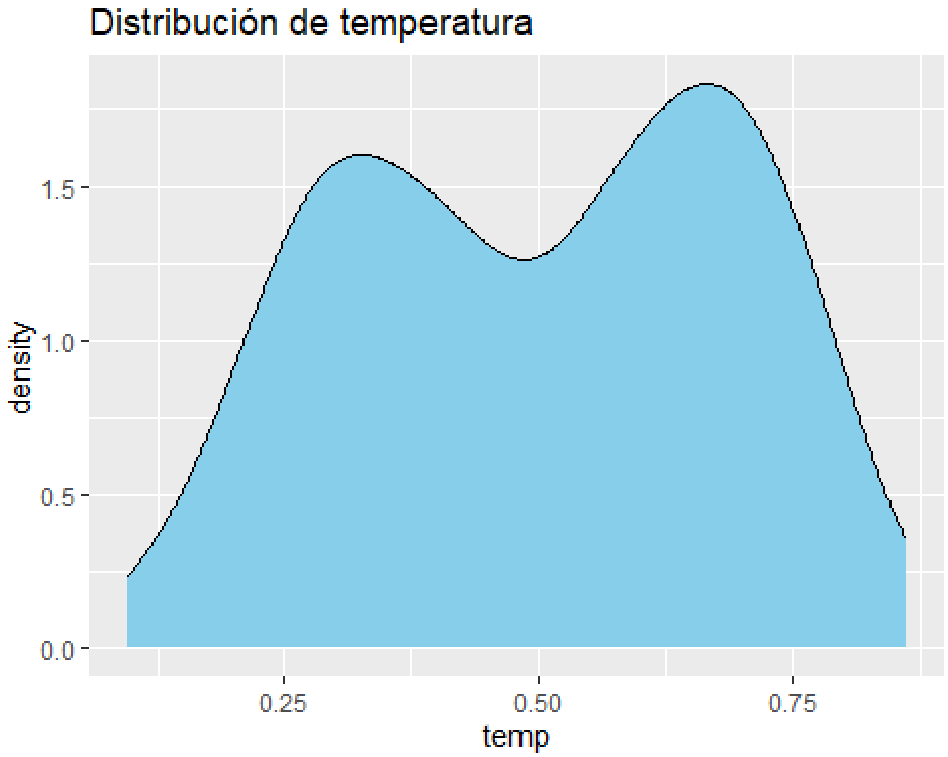
Paso 6 : Calcular PDP bidimensional

```
# PDP 2D con temperatura y humedad
pdp_2d <- partial(bike_model,
                  pred.var = c("temp", "hum"),
                  grid.resolution = 20,
                  progress = "text")
# Visualizar como mapa de calor
pdp_plot <- ggplot(pdp_2d, aes(x = temp, y = hum, fill = yhat)) +
  geom_tile(width = 0.02, height = 0.02) +  # ancho/alto para que no haya
huecos
  scale_fill_viridis_c(option = "plasma") +
  labs(title = "PDP 2D - Influencia conjunta de temperatura y humedad",
       x = "Temperatura normalizada",
       y = "Humedad",
       fill = "Alquileres predichos") +
  theme_minimal()

print(pdp_plot)
```

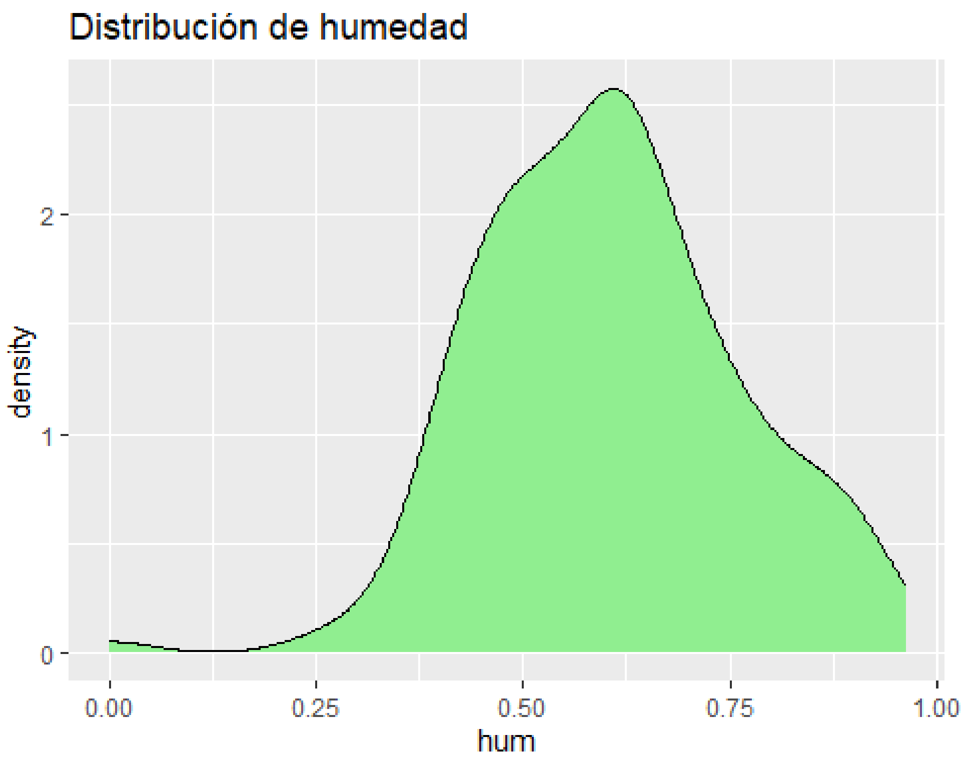


Mostrar densidades de temperatura y humedad

```
# Tomar una muestra aleatoria para que sea más rápido
set.seed(123)  # para reproducibilidad
bike_sample <- sample_n(bike_data, 150)
# Densidad de temperatura
ggplot(bike_sample, aes(x = temp)) +
  geom_density(fill = "skyblue") +
  labs(title = "Distribución de temperatura")
```



```r
# Densidad de humedad
ggplot(bike_sample, aes(x = hum)) +
  geom_density(fill = "lightgreen") +
  labs(title = "Distribución de humedad")
```

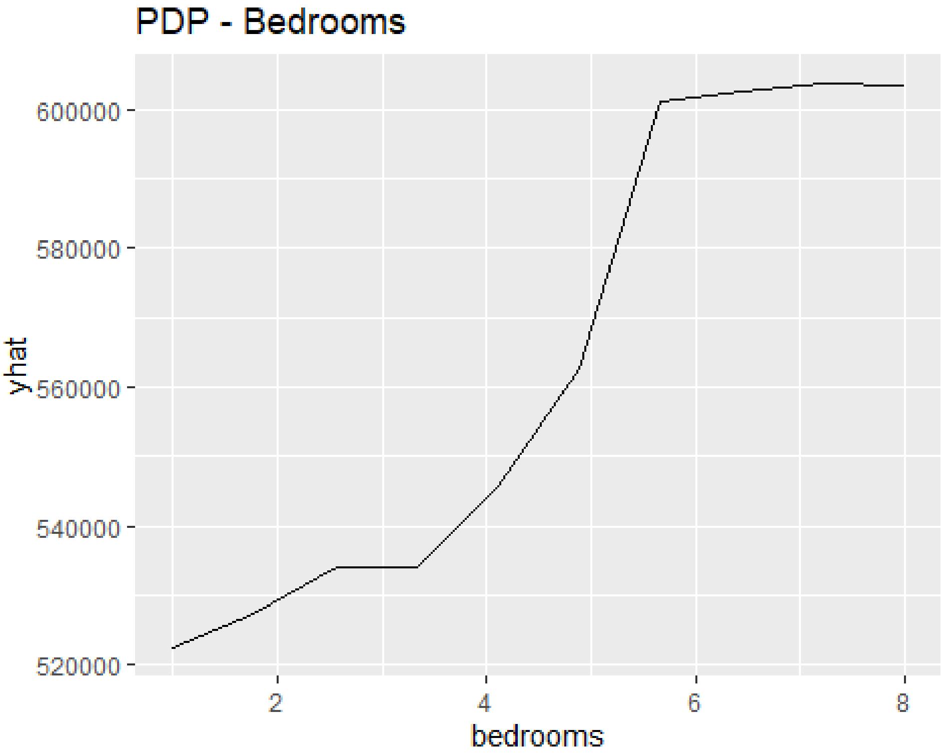


ANALISIS DE COMO INFLUYEN DIFERENTES VARIABLES A LA VIVIENDA

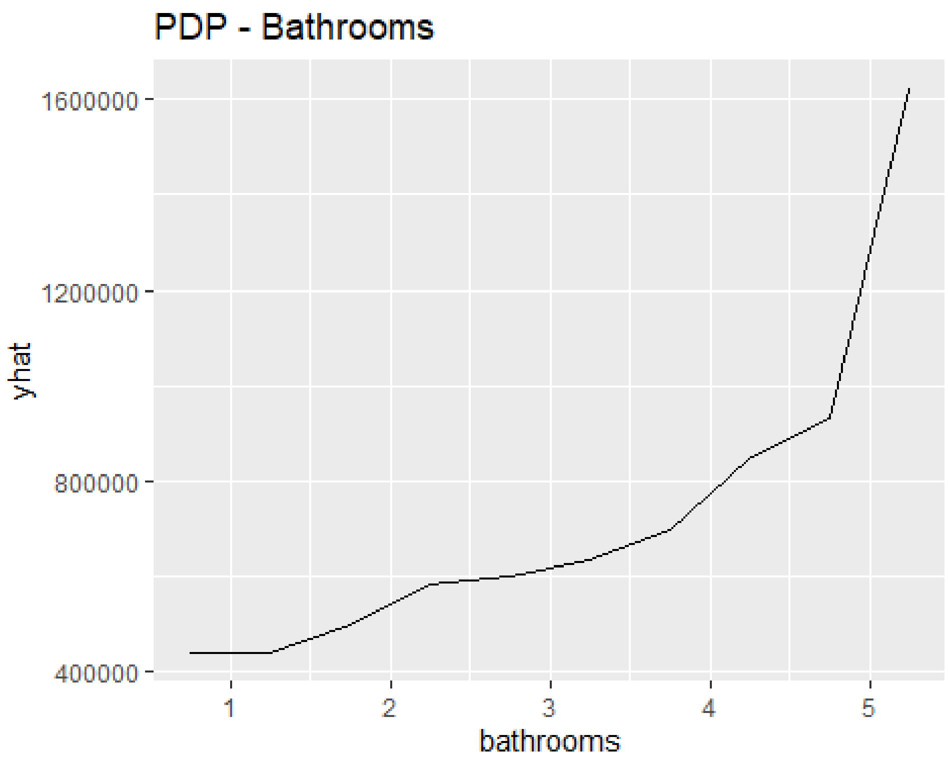
```
house_data <- read_csv("C:/Users/Elena Chirivella/OneDrive -
UPV/CARRERA/4o/edm/EDM ELEN/practicas/Practica 5 Sin
sol/kc_house_data.csv")

## Rows: 21613 Columns: 21
## ── Column specification
────────────────────────────────────────────────────────
## Delimiter: ","
## chr   (1): id
## dbl  (19): price, bedrooms, bathrooms, sqft_living, sqft_lot, floors,
waterf...
## dttm  (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.

set.seed(123)
house_sample <- sample_n(house_data, 300)
```

Entrenar el modelo Random Forest

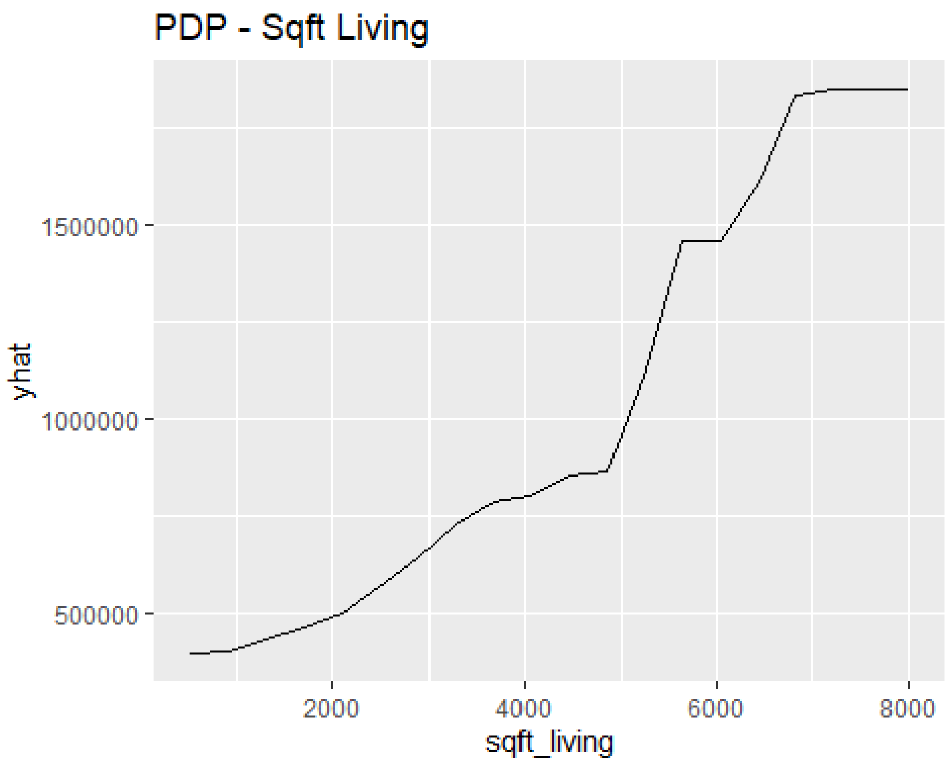
```
house_model <- randomForest(price ~ bedrooms + bathrooms + sqft_living +
sqft_lot + floors + yr_built,
                            data = house_sample,
                            ntree = 300)

# PDP para bedrooms
pdp_bed <- partial(house_model, pred.var = "bedrooms", grid.resolution =
10)
plot_bed <- autoplot(pdp_bed) + ggtitle("PDP - Bedrooms")

# PDP para bathrooms
pdp_bath <- partial(house_model, pred.var = "bathrooms", grid.resolution
= 10)
plot_bath <- autoplot(pdp_bath) + ggtitle("PDP - Bathrooms")

# PDP para sqft_living
pdp_sqft <- partial(house_model, pred.var = "sqft_living",
grid.resolution = 20)
plot_sqft <- autoplot(pdp_sqft) + ggtitle("PDP - Sqft Living")

# PDP para floors
pdp_floors <- partial(house_model, pred.var = "floors", grid.resolution =
10)
plot_floors <- autoplot(pdp_floors) + ggtitle("PDP - Floors")

print(plot_bed)
```



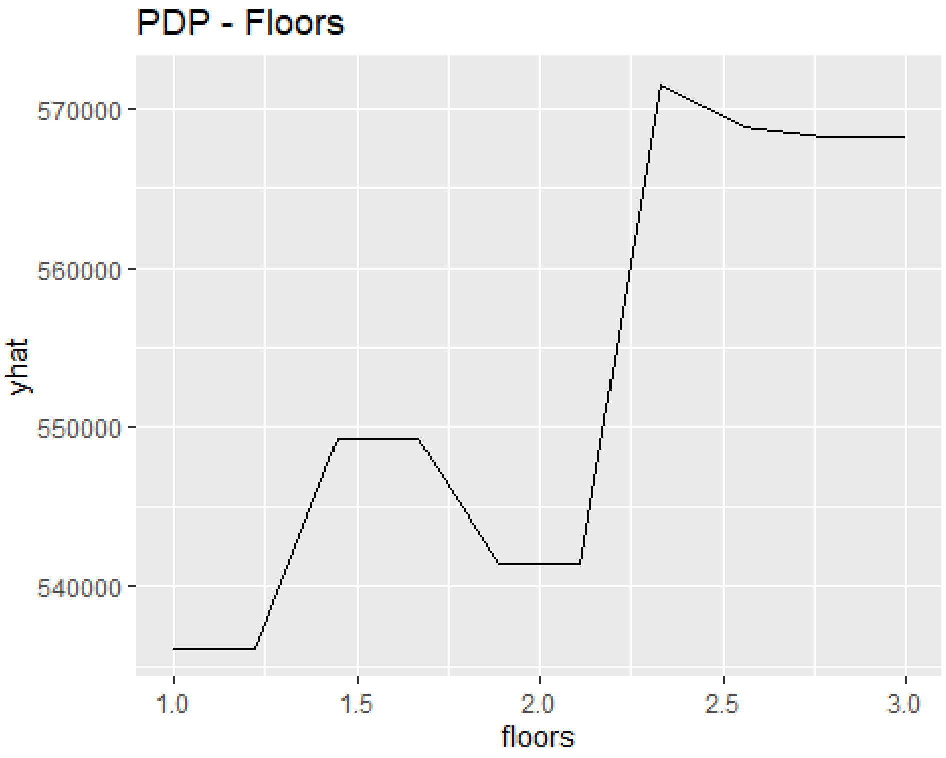
```
print(plot_bath)
```



```r
print(plot_sqft)
```



```r
print(plot_floors)
```



PDP - Floors