Tarea 6: evaluación en problemas de clasificación

En este problema evaluamos un clasificador para predecir qué clientes comprarárán un seguro para caravanas (*campers*). Tenemos cierta información socioeconómica de los clientes, así como información acerca de sus compras y conducta. Este problema es interesante también porque presenta desbalance considerable entre la clase de compra y la de no compra.

```
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------------------
```

```
## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.3     v dplyr   1.0.2
## v tidyr   1.1.1     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0
```

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

```
library(tidymodels)
```

```
## -- Attaching packages ---------------------------------------------------------------------------
```

```
## v broom     0.7.0      v recipes   0.1.13
## v dials     0.0.8      v rsample   0.0.7
## v infer     0.5.3      v tune      0.1.1
## v modeldata 0.0.2      v workflows 0.1.3
## v parsnip   0.1.3      v yardstick 0.0.7
```

```
## -- Conflicts ------------------------------------------------------------------------------------
## x scales::discard() masks purrr::discard()
## x dplyr::filter()   masks stats::filter()
## x recipes::fixed()  masks stringr::fixed()
## x dplyr::lag()      masks stats::lag()
## x yardstick::spec() masks readr::spec()
## x recipes::step()   masks stats::step()
```

Consideremos los siguientes datos (del paquete @ISLR):

The data contains 5,822 real customer records. Each record consists of 86 variables, containing sociodemographic data (variables 1-43) and product ownership (variables 44-86). The sociodemographic data is derived from zip codes. All customers living in areas with the same zip code have the same sociodemographic attributes. Variable 86 (Purchase) indicates whether the customer purchased a caravan insurance policy. Further information on the individual variables can be obtained at http://www.liacs.nl/~putten/library/cc2000/data.html

Aquí puedes ver un resumen de las variables, **aunque por el momento no nos preocupamos mucho por esto**:

Todas las variables son numéricas, excepto MOSTYPE, MOSHOOFD

MOSTYPE: Customer Subtype; see L0 MAANTHUI: Number of houses 1 - 10 MGEMOMV: Avg size household 1 - 6 MGEMLEEF: Avg age; see L1 MOSHOOFD: Customer main type; see L2

MGODRK: Roman catholic MGODPR: Protestant ... MGODOV: Other religion MGODGE: No religion MRELGE: Married MRELSA: Living together MRELOV: Other relation MFALLEEN: Singles MFGEKIND: Household without children MFWEKIND: Household with children MOPLHOOG: High level education MOPLMIDD: Medium level education MOPLLAAG: Lower level education MBERHOOG: High status MBERZELF: Entrepreneur MBERBOER: Farmer MBERMIDD: Middle management MBERARBG: Skilled labourers MBERARBO: Unskilled labourers MSKA: Social class A MSKB1: Social class B1 MSKB2: Social class B2 MSKC: Social class C MSKD: Social class D MHHUUR: Rented house MHKOOP: Home owners MAUT1: 1 car MAUT2: 2 cars MAUT0: No car MZFONDS: National Health Service MZPART: Private health insurance MINKM30: Income < 30.000 MINK3045: Income 30-45.000 MINK4575: Income 45-75.000 MINK7512: Income 75-122.000 MINK123M: Income >123.000 MINKGEM: Average income MKOOPKLA: Purchasing power class

PWAPART: Contribution private third party insurance PWABEDR: Contribution third party insurance (firms) ... PWALAND: Contribution third party insurane (agriculture) PPERSAUT: Contribution car policies PBESAUT: Contribution delivery van policies PMOTSCO: Contribution motorcycle/scooter policies PVRAAUT: Contribution lorry policies PAANHANG: Contribution trailer policies PTRACTOR: Contribution tractor policies PWERKT: Contribution agricultural machines policies PBROM: Contribution moped policies PLEVEN: Contribution life insurances PPERSONG: Contribution private accident insurance policies PGEZONG: Contribution family accidents insurance policies PWAOREG: Contribution disability insurance policies PBRAND: Contribution fire policies PZEILPL: Contribution surfboard policies PPLEZIER: Contribution boat policies PFIETS: Contribution bicycle policies PINBOED: Contribution property insurance policies PBYSTAND: Contribution social security insurance policies AWAPART: Number of private third party insurance 1 - 12 AWABEDR: Number of third party insurance (firms) ... AWALAND: Number of third party insurance (agriculture) APERSAUT: Number of car policies ABESAUT: Number of delivery van policies AMOTSCO: Number of motorcycle/scooter policies AVRAAUT: Number of lorry policies AAANHANG: Number of trailer policies ATRACTOR: Number of tractor policies AWERKT: Number of agricultural machines policies ABROM: Number of moped policies ALEVEN: Number of life insurances APERSONG: Number of private accident insurance policies AGEZONG: Number of family accidents insurance policies AWAOREG: Number of disability insurance policies ABRAND: Number of fire policies AZEILPL: Number of surfboard policies APLEZIER: Number of boat policies AFIETS: Number of bicycle policies AINBOED: Number of property insurance policies ABYSTAND: Number of social security insurance policies CARAVAN: Number of mobile home policies 0 - 1

L0: Customer subtype

1: High Income, expensive child 2: Very Important Provincials 3: High status seniors 4: Affluent senior apartments 5: Mixed seniors 6: Career and childcare 7: Dinki's (double income no kids) 8: Middle class families 9: Modern, complete families 10: Stable family 11: Family starters 12: Affluent young families 13: Young all american family 14: Junior cosmopolitan 15: Senior cosmopolitans 16: Students in apartments 17: Fresh masters in the city 18: Single youth 19: Suburban youth 20: Etnically diverse 21: Young urban have-nots 22: Mixed apartment dwellers 23: Young and rising 24: Young, low educated 25: Young seniors in the city 26: Own home elderly 27: Seniors in apartments 28: Residential elderly 29: Porchless seniors: no front yard 30: Religious elderly singles 31: Low income catholics 32: Mixed seniors 33: Lower class large families 34: Large family, employed child 35: Village families 36: Couples with teens 'Married with children' 37: Mixed small town dwellers 38: Traditional families 39: Large religous families 40: Large family farms 41: Mixed rurals

L2: customer main type keys:

1: Successful hedonists 2: Driven Growers 3: Average Family 4: Career Loners 5: Living well 6: Cruising Seniors 7: Retired and Religeous 8: Family with grown ups 9: Conservative families 10: Farmers

Queremos predecir la variable *Purchase*, que indica si el cliente compró o no el seguro de camper:

```
caravan <- read_csv("caravan.csv") %>%
  mutate(MOSTYPE = factor(MOSTYPE),
         MOSHOOFD = factor(MOSHOOFD))
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   Purchase = col_character()
## )
```

```
## See spec(...) for full column specifications.
```

```
set.seed(823)
# usamos muestreo estratificado para tener el mismo balance
# de Purchase en
caravan_split = initial_split(caravan, strata = Purchase, prop = 0.8)
caravan_split
```

```
## <Analysis/Assess/Total>
## <4658/1164/5822>
```

```
caravan_ent <- training(caravan_split)
```

Y vemos el desbalance de clases:

```
nrow(caravan_ent)
```

```
## [1] 4658
```

```
caravan_ent %>% count(Purchase) %>%
  mutate(pct = 100 * n / sum(n)) %>%
  mutate(pct = round(pct, 2))
```

```
## # A tibble: 2 x 3
##   Purchase     n   pct
##   <chr>    <int> <dbl>
## 1 No        4384 94.1
## 2 Yes        274  5.88
```

Esta es la distribución natural de respuesta que vemos en los datos, y tenemos relativamente pocos datos en la categoría "Yes".

Y usaremos regresión logística (aunque lo mismo aplicará para métodos más avanzados)

```
# preparacion de datos
caravan_receta <- recipe(Purchase ~ ., caravan_ent) %>%
  # convertir a dummy variables nominales, esto lo explicamos
  # con detalle más adelante
  step_dummy(all_nominal(), -Purchase) %>%
  step_normalize(all_predictors()) %>%
  step_relevel(Purchase,  ref_level = "Yes") %>%
  prep()
# modelo - nota: puedes ajustar los parámetros de descenso, pero
# para este ejercicio no es tan importante.
modelo_log <-
  logistic_reg() %>%
  set_engine("keras")  %>%
  set_mode("classification") %>%
  fit(Purchase ~ ., data = caravan_receta %>% juice)
```

Análisis con tasas de correctos (no es apropiado) La matriz de confusión de prueba es:

```
prueba_procesado <- bake(caravan_receta, testing(caravan_split))
predictions_glm <- modelo_log %>%
  predict(new_data = prueba_procesado) %>%
  bind_cols(prueba_procesado %>% select(Purchase))
predictions_glm %>%
  conf_mat(Purchase, .pred_class)
```

```
##           Truth
## Prediction  Yes   No
##        Yes    1    6
##        No    73 1084
```

Y la de entrenamiento:

```
predictions_ent_glm <- modelo_log %>%
  predict(new_data = juice(caravan_receta)) %>%
  bind_cols(juice(caravan_receta) %>% select(Purchase))
predictions_ent_glm %>%
  conf_mat(Purchase, .pred_class)
```

```
##           Truth
## Prediction  Yes   No
##        Yes    3    9
##        No   271 4375
```

Pregunta 1: ¿qué piensas del desempeño de este clasificador? ¿Cómo crees que este paquete decide clasificar en "Yes" o "No" dependiendo del modelo de regresión logística que ajustaste?

Pregunta 2 Calcula especificidad y sensiblidad para este clasificador. ¿Crees que es útil? ¿Existen clasificadores triviales que se desempeñen de manera similar

```
## Metodología 1 para calcular métricas del modelo
esp <- 4371/(4371 + 13)
sprintf("La especificidad del modelo es: %1.4f", esp)
```

```
## [1] "La especificidad del modelo es: 0.9970"
```

```
sens <- 4/(4 + 270)
sprintf("La sensibilidad del modelo es: %1.4f", sens)
```

```
## [1] "La sensibilidad del modelo es: 0.0146"
```

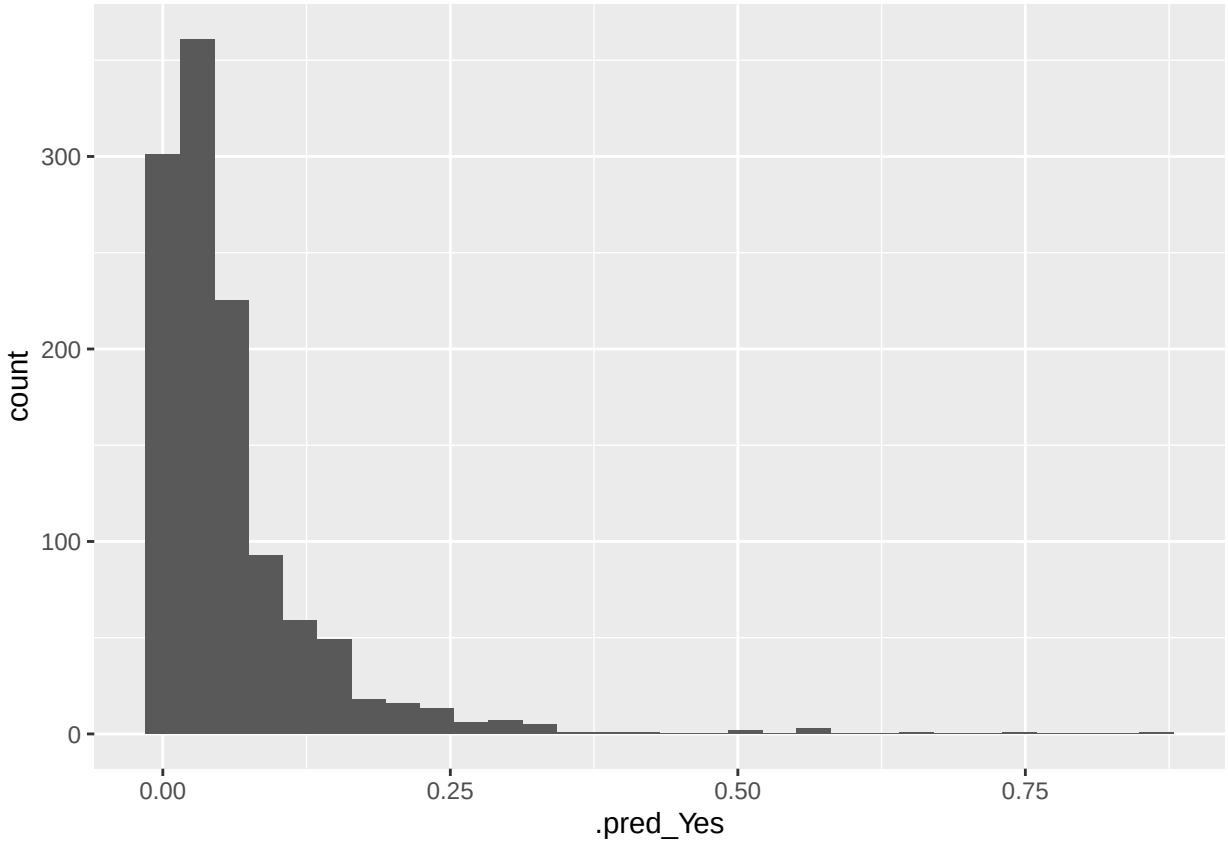
```
## Metodología 2 para calcular métricas del modelo (no está funcionando)
# met_1 <- metric_set(precision, recall)
# predictions_ent_glm %>%
#   met_1(type, estimate=.pred_class)
```

Análisis correcto usando probabilidades Podemos trabajar con probabilidades en lugar de predicciones de clase con punto de corte de 0.5.

Las probabilidades que obtenemos podemos visualizarlas:

```
preds_prob <- modelo_log %>%
  predict(new_data = prueba_procesado, type = "prob") %>%
  bind_cols(prueba_procesado %>% select(Purchase)) %>%
  select(.pred_Yes, .pred_No, Purchase)
ggplot(preds_prob, aes(x = .pred_Yes)) + geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



que en general muestran probabilidades bajas de comprar este producto.

Por ejemplo, podemos visualizar con una curva ROC (o curva lift, precisión recall, o alguna otra similar que tome en cuenta probabilidades):

```
preds_prob <- modelo_log %>%
  predict(new_data = prueba_procesado, type = "prob") %>%
  bind_cols(prueba_procesado %>% select(Purchase)) %>%
  select(.pred_Yes, .pred_No, Purchase)
datos_roc <- roc_curve(preds_prob, Purchase, .pred_Yes)

datos_roc %>%
  filter((sensitivity > 0.79) & (sensitivity < 0.82))
```

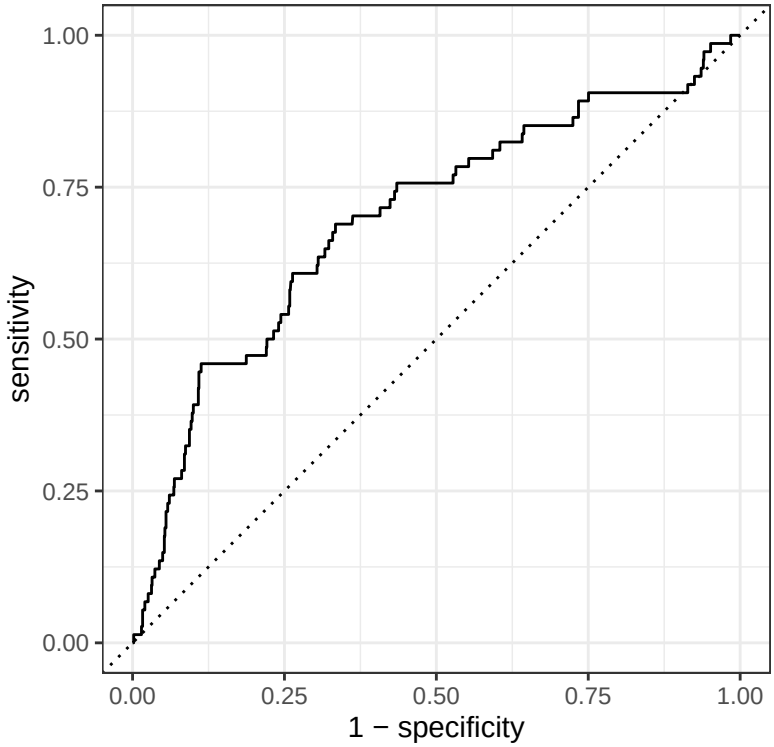
```
## # A tibble: 58 x 3
##    .threshold specificity sensitivity
##         <dbl>       <dbl>       <dbl>
##  1     0.0241       0.395       0.811
##  2     0.0242       0.396       0.811
##  3     0.0242       0.397       0.811
##  4     0.0243       0.398       0.811
##  5     0.0243       0.399       0.811
##  6     0.0244       0.4         0.811
##  7     0.0245       0.401       0.811
##  8     0.0245       0.402       0.811
##  9     0.0245       0.403       0.811
## 10     0.0245       0.404       0.811
```

```
## # ... with 48 more rows
```

```
roc_ant <- datos_roc
roc_ant
```

```
## # A tibble: 1,135 x 3
##      .threshold specificity sensitivity
##           <dbl>       <dbl>       <dbl>
##  1 -Inf            0                   1
##  2    0.0000187    0                   1
##  3    0.0000312    0.000917            1
##  4    0.000357     0.00183             1
##  5    0.000407     0.00275             1
##  6    0.000566     0.00367             1
##  7    0.000621     0.00459             1
##  8    0.000651     0.00550             1
##  9    0.000705     0.00642             1
## 10    0.000712     0.00734             1
## # ... with 1,125 more rows
```

```
autoplot(datos_roc)
```



Donde vemos que podemos alcanzar buenos niveles de sensibilidad si aceptamos alguna degradación en la especificidad, que originalmente es muy alta. Por ejemplo, cortando en 0.05 podemos obtener especificidad y sensibilidad que posiblemente sean adecuadas para el problema:

```
datos_roc %>% filter(abs(.threshold - 0.05) < 1e-4) %>% round(3)
```

```
## # A tibble: 1 x 3
##   .threshold specificity sensitivity
##        <dbl>       <dbl>       <dbl>
## 1       0.05       0.628       0.703
```

Y evaluamos también con devianza y auc:

```
metricas <- metric_set(roc_auc, mn_log_loss)
metricas
```

```
##        metric       class direction
## 1     roc_auc prob_metric  maximize
## 2 mn_log_loss prob_metric  minimize
```

```
preds_prob %>%
  metricas(Purchase, .pred_Yes)
```

```
## # A tibble: 2 x 3
##   .metric     .estimator .estimate
##   <chr>       <chr>          <dbl>
## 1 roc_auc     binary         0.697
## 2 mn_log_loss binary         0.232
```

Pregunta 3 ¿Qué devianza tiene un modelo que da la probabilidad base de 0.5 a todos los casos?

```
-log(0.5)
```

```
## [1] 0.6931472
```

```
n <- 15
(-2/n)*(log(0.5) * n)
```

```
## [1] 1.386294
```

- ¿El modelo anterior es superior o inferior a este modelo base? ¿Qué pasa si usas la probabilidad de compra en la muestra de entrenamiento como tu predicción?

Un modelo más simple

Supongamos que usamos un modelo que sólo incluye unas cuantas variables:

ABRAND: Number of fire policies APERSAUT: Number of car policies

```
caravan_ent %>%
  select(ABRAND, APERSAUT) %>%
  summary
```

```
##      ABRAND          APERSAUT
##  Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.0000   1st Qu.:0.0000
##  Median :1.0000   Median :1.0000
##  Mean   :0.5719   Mean   :0.5577
##  3rd Qu.:1.0000   3rd Qu.:1.0000
##  Max.   :7.0000   Max.   :6.0000
```

```
modelo_log_simple <-
  logistic_reg() %>%
  set_engine("keras")  %>%
  set_mode("classification") %>%
  fit(Purchase ~ MOPLHOOG + ABRAND + APERSAUT, data = caravan_receta %>% juice)
```

```
preds_prob_simple <- modelo_log_simple %>%
  predict(new_data = prueba_procesado, type = "prob") %>%
  bind_cols(prueba_procesado %>% select(Purchase)) %>%
```

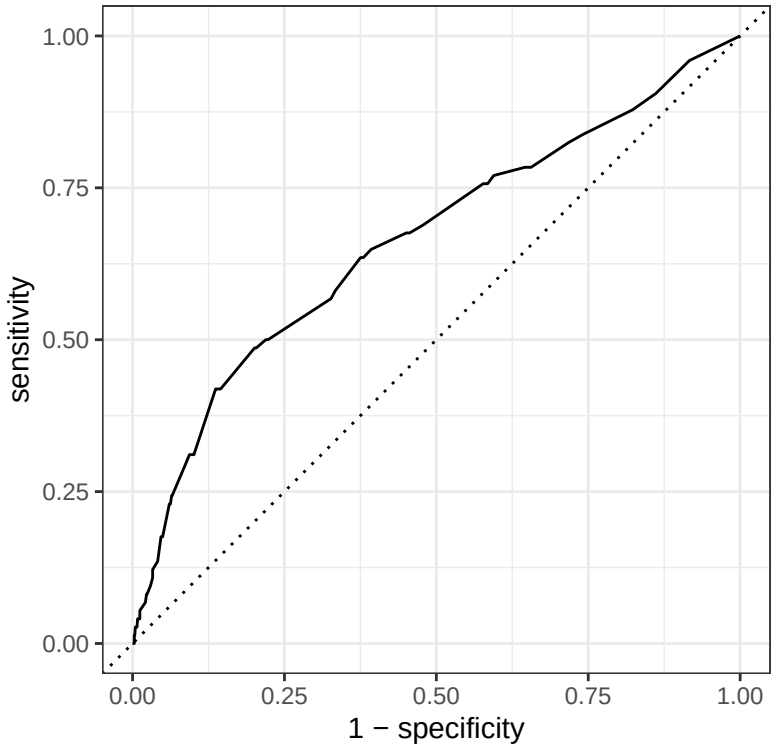
```
  select(.pred_Yes, Purchase)

datos_roc <- roc_curve(preds_prob_simple, Purchase, .pred_Yes)

roc_simp <- datos_roc
roc_simp
```

```
## # A tibble: 62 x 3
##    .threshold specificity sensitivity
##         <dbl>       <dbl>       <dbl>
##  1  -Inf           0           1
##  2     0.0182      0           1
##  3     0.0215      0.0835      0.959
##  4     0.0255      0.139       0.905
##  5     0.0278      0.177       0.878
##  6     0.0301      0.259       0.838
##  7     0.0329      0.283       0.824
##  8     0.0356      0.344       0.784
##  9     0.0389      0.354       0.784
## 10     0.0420      0.406       0.770
## # ... with 52 more rows
```

```
autoplot(datos_roc)
```



Pregunta 4: Corta en una probablilidad que te de alrededor 80% de sensibilidad (probabilidad relativamente alta de identificar a los compradores). ¿Cómo se compara la especificidad de este modelo con la que obtendrías con el modelo anterior, al mismo nivel de sensibilidad? Imagina un posible problema que quisieras resolver con este clasificador. ¿La diferencia que encontraste tiene relevancia para este problema?

```
## Modelo simple: selección de valor relevante
# datos_roc %>%
```

```r
#   filter((sensitivity > 0.7) & (sensitivity < 0.9))
print("Métricas seleccionadas del modelo simple:")
```

```
## [1] "Métricas seleccionadas del modelo simple:"
```

```r
print("    - Sensibilidad: 0.8108108")
```

```
## [1] "    - Sensibilidad: 0.8108108"
```

```r
print("    - Especificidad: 0.2752294")
```

```
## [1] "    - Especificidad: 0.2752294"
```

```r
print("---")
```

```
## [1] "---"
```

```r
## Modelo anterior
print("Métricas seleccionadas del modelo anterior:")
```

```
## [1] "Métricas seleccionadas del modelo anterior:"
```

```r
print("    - Sensibilidad: 0.8108108")
```

```
## [1] "    - Sensibilidad: 0.8108108"
```

```r
print("    - Especificidad: 0.4018349")
```

```
## [1] "    - Especificidad: 0.4018349"
```

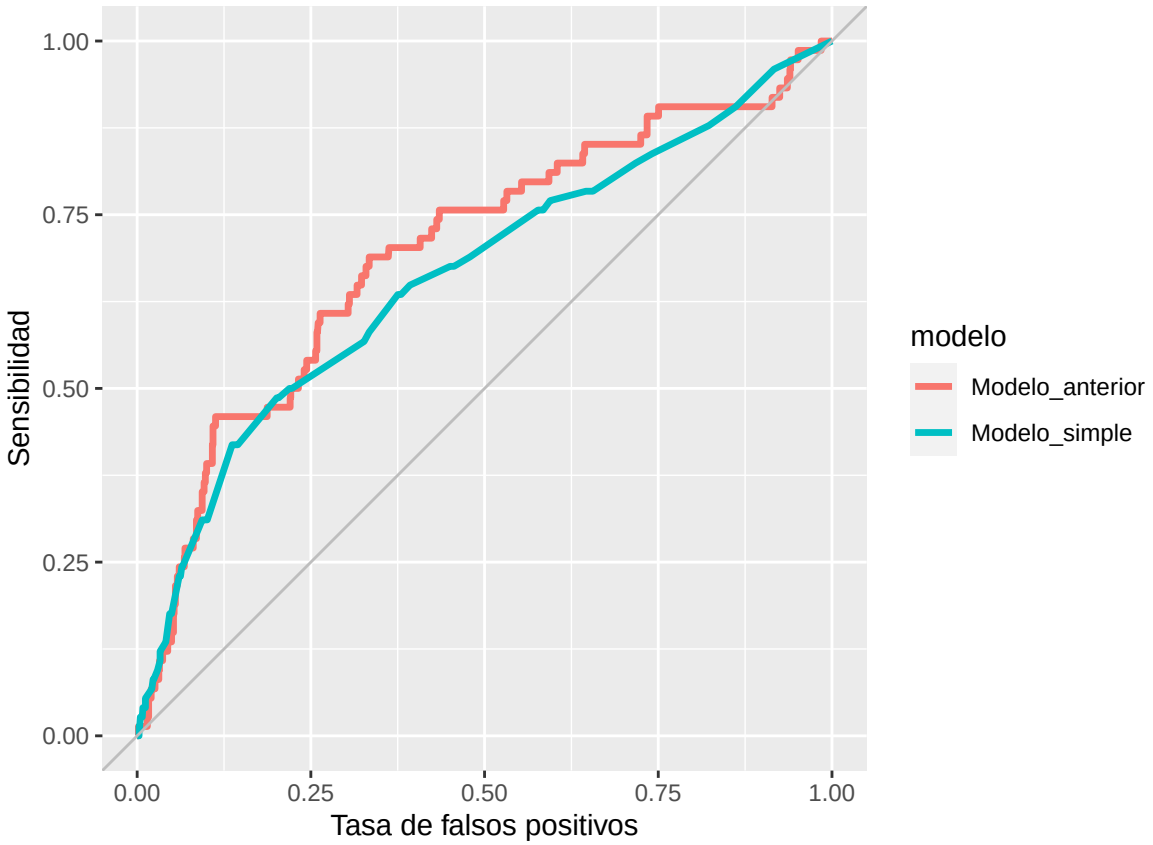
Pregunta 5 Grafica las dos curvas ROC de estos clasificadores y explica cuál funciona mejor.

```r
## Renaming columns related to past model
roc_ant_r <- roc_ant %>%
  mutate(modelo = "Modelo_anterior")

## Renaming columns related to simple model
roc_simp_r <- roc_simp %>%
  mutate(modelo = "Modelo_simple")

roc_tbl <- bind_rows(roc_ant_r, roc_simp_r)

## Plotting ROC curves for both models
ggplot(
  roc_tbl,
  aes(x = 1 - specificity, y = sensitivity)
) +
  geom_path(
    aes(colour = modelo),
    size = 1.2
  ) +
  geom_abline(colour = "gray") +
  coord_equal() +
  xlab("Tasa de falsos positivos") + ylab("Sensibilidad")
```



<!-- - Si tienes muchas variables te comprometes a tenerlas todas en un futuro. -->