*PROBLEM SET 2*

*“Las guerras de las naciones se libran para cambiar los mapas. Pero las guerras de la pobreza se libran para mapear el cambio” M. Ali*

***Introducción***

expone brevemente el problema y sus antecedentes, una descripción de los datos y su idoneidad para el problema en cuestión. Contiene una vista previa de los resultados y las conclusiones principales)

El siguiente trabajo está inspirado en una competencia reciente organizada por el banco mundial: Pover-Tests: Predicting Poverty, el cual consistía en predecir la pobreza de forma eficiente. Como afirma la competencia,

“*medir la pobreza es difícil, requiere mucho tiempo y es costoso. Al construir mejores modelos, podemos realizar encuestas con menos preguntas y más específicas que miden de manera rápida y económica la efectividad de las nuevas políticas e intervenciones. Cuanto más precisos sean nuestros modelos, con mayor precisión podremos orientar las intervenciones e iterar las políticas, maximizando el impacto y la rentabilidad de estas estrategia*s”.

El objetivo de este trabajo es predecir la pobreza a nivel de los hogares para Colombia, a través del uso de datos proporcionados a nivel de hogar e individual, con el propósito de que se pueda medir la pobreza de una manera rápida y económica, al construir un modelo que use la menor cantidad de variables para su predicción.

Los datos utilizados para este análisis contienen cuatro conjuntos de datos de la Gran Encuesta Integrada de Hogares (GEIH) para el año 2018. Dicha muestra está dividida entre la muestra de entrenamiento y la de prueba. La descripción de las variables está disponible en los anexos de este trabajo.

El orden de este trabajo será primero esta introducción, seguida de la presentación de los datos utilizados para el ejercicio de predicción y finalizando con la presentación de las especificaciones y modelos utilizados para las tareas predictiva y los resultados.

***Datos***

Trate esta sección como una oportunidad para presentar una narrativa convincente para justificar o defender sus opciones de datos y ayudar al lector a comprender sus datos y su variación. Utilice su conocimiento profesional para agregar valor a esta sección. No la presente como una lista “seca” de ingredientes.

Datos de empleo

Condiciones e indicadores sociales

Ingresos, propiedad, ahorro

***Modelo***

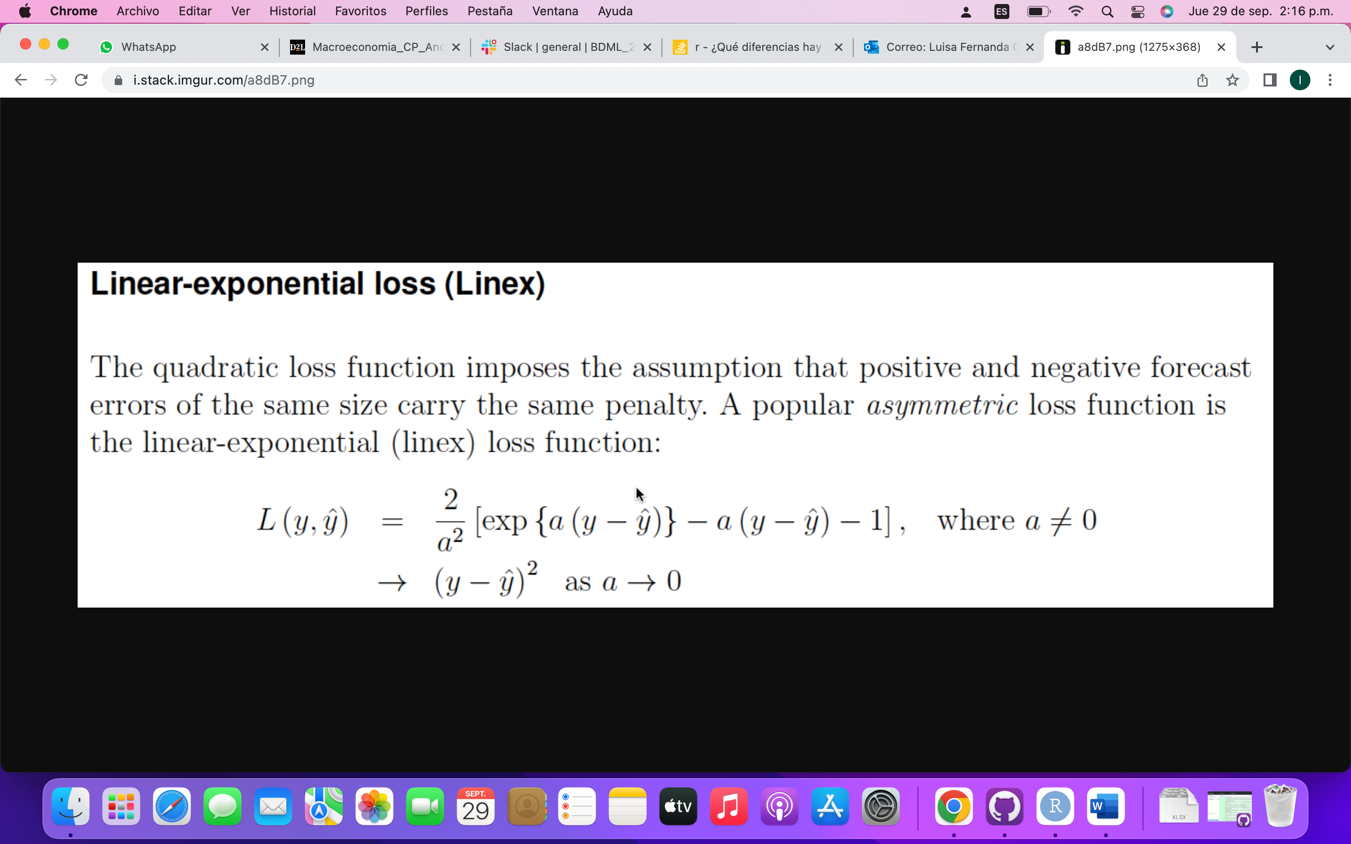
*Esta sección presenta las especificaciones y modelos utilizados para las tareas predictivas.*

Dado que el objetivo principal es construir un modelo predictivo de la pobreza en los hogares colombianos, entonces los hogares serán clasificados como

donde es una función indicadora que toma valores de 1 si el ingreso del hogar está por debajo de cierta línea de pobreza.

Esto sugiere dos formas de predecir la pobreza, en primer lugar, a partir del enfoque como un problema de clasificación: donde se debe asignar ceros para hogares no pobres y unos para hogares pobres. En segundo lugar, como un problema de predicción de ingresos, donde a partir de los ingresos predichos, se pueda obtener la clasificación haciendo uso de la línea de pobreza. En este trabajo se explorarán ambas rutas.

Existen diferentes métodos para llevar a cabo la identificación de la población pobre. Sen (1979) diferencia entre dos métodos generales. El primero de ellos es el método directo. A través de éste se identifica el conjunto de hogares (o personas) que no satisfacen un **grupo específico de necesidades previamente establecidas (condiciones de la vivienda, educación, composición demográfica del hogar, tenencia de activos, etc**.). El segundo se denomina método indirecto. Es este caso la identificación se lleva a cabo mediante el **cálculo de un umbral mínimo, por lo general asociado a un nivel de ingreso o gasto, por debajo del cual se considera que una persona no puede satisfacer sus necesidades básicas**. Mientras que el primero se basa en la satisfacción efectiva de un conjunto de necesidades, el segundo hace referencia a la posibilidad de satisfacerlas (considerando para ello un nivel mínimo de ingresos).



|  |  |
| --- | --- |
| Precision | Precisión |
| Recall | Exhaustividad |
| F1-score | Valor-F |
| Accuracy | Exactitud |
| Confusion Matrix | Matriz de Confusión |
| True Positive | Positivos Verdaderos |
| True Negative | Negativos Verdaderos |
| False Positive | Positivos Falsos |
| False Negative | Negativos Falsos |

* ***Modelos de clasificación***

- Esta subsección describe el enfoque de clasificación, es decir, su intento de predecir directamente ceros (no pobre) y unos (pobre)

- Una comparación con al menos otras 5 especificaciones/modelos en términos de poder predictivo para la tarea en cuestión.

– Una descripción de las variables utilizadas en el modelo y discutir su importancia relativa en la predicción.

– Una descripción de cualquier estrategia de submuestreo utilizada para abordar el desequilibrio de clases

* ***Modelos de predicción de ingresos***

– Una explicación detallada del modelo final elegido. La explicación debe incluir cómo se entrenó el modelo, la selección de hiperparámetros y otra información relevante.

– Una comparación con al menos otras 5 especificaciones/modelos en términos de poder predictivo para la tarea en cuestión.

– Una descripción de las variables utilizadas en el modelo y discutir su importancia relativa en la predicción.

– Una descripción de cualquier estrategia de submuestreo utilizada para abordar el desequilibrio de clases

***Resultados***

Juzgaré las predicciones en función de las tasas de falsos positivos, las tasas de falsos negativos y la escasez del modelo. En la puntuación final tendrán más peso los falsos negativos (75%), es decir, familias pobres clasificadas como no pobres, y cuantas más variables uses, menor será tu puntuación.

Cuantas más variables uses, menor será tu puntuación

Las transformaciones de la misma variable no cuentan como variables adicionales.

Las tablas, figuras y escritos deben ser lo más prolijos posible. Etiquete todas las variables incluidas. Si tiene algo en sus figuras o tablas, espero que se aborden en el texto.

***Conclusiones y recomedaciones***

|  |
| --- |
| #-------PROBLEM SET 2 |
|  | #Regresión |
|  | # Lasso |
|  | # Ridge |
|  | # Elastic Net |
|  | # regresión normal |
|  | # subset selection (backward) |
|  | # medida para mejorar la métrica |
|  |  |
|  | #Clasificación |
|  | # K vecinos cercanos |
|  | # Arbol |
|  | # RF |
|  | # Lasso logit |
|  | # LDA |
|  | # Medida para mejorar la métrica |
|  |  |
|  | #-------Lectura bases |
|  | train\_hogares<-readRDS(here("../data/train\_hogares.Rds")) |
|  | train\_personas<-readRDS(here("../data/train\_personas.Rds")) |
|  |  |
|  | #-------Unión de variables |
|  | install.packages("glmnet") |
|  | require("here") |
|  | require("tidyverse") |
|  | require("glmnet") |
|  | install.packages("tidyverse") |
|  | install.packages("glmnet", repos = "https://cran.us.r-project.org") |
|  | install.packages("caret", dependencies = c("Depends", "Suggests")) |
|  | library(AppliedPredictiveModeling) |
|  | library(caret) |
|  | install.packages("ggplot2") |
|  | library (ggplot2) |
|  | library(glmnet) |
|  | library(pacman) |
|  | p\_load(tidyverse, fastDummies, caret, glmnet, MLmetrics) |
|  | library(modeest) |
|  | library(dplyr) |
|  | library(tidyverse) |
|  | library(glmnet) |
|  | library(pacman) |
|  | p\_load(AER, tidyverse, caret, MLmetrics, tidymodels, themis) |
|  |  |
|  | colnames(train\_hogares) |
|  | for i { |
|  | "id" |
|  | "Clase" |
|  | "Dominio" |
|  | "Lp" |
|  | "Pobre" |
|  | "Npobres" |
|  | "Depto" |
|  | "Hacinamiento= Nper/P5010" |
|  | "P5090= Vivienda propia totalmente pagado, propia la están pagando, |
|  | en arriendo o subarriendo, usufructo, posesión sin título" |
|  | "PAUG=Npersug/Nper= Número de personas en la unidad de gasto" |
|  | "Ingtotugarr=Ingreso total de la unidad de gasto con la imputación del arriendo" |
|  | "Ingpcug=Ingreso per capita de Ingtotugarr" } |
|  |  |
|  | colnames(train\_personas) |
|  | for i ¨{ |
|  | "id" |
|  | "Clase" |
|  | "estrato=Estrato1 (media)" |
|  | "Sexo\_jefe\_hog=P6020 y P6050" |
|  | "edad\_jefe\_hogar=P6020 P6050" |
|  | "rs\_jefe\_hog (P6100 P6050)" |
|  | "educ\_jefe\_hog (P6210)" |
|  | "P6430 |
|  | a: obrero empleado empresa particular |
|  | b. obrero empleado del gob |
|  | c. empleado doméstico |
|  | d. Trabajador cuenta propia |
|  | e. Patrón empleador |
|  | f. Trabajador familiar sin remuneración |
|  | g. Trabajador sin remuneración externo |
|  | h. jornalero peón |
|  | i. otro |
|  |  |
|  | Clasificar jefe de hogar " |
|  |  |
|  | #subsidios (El hogar recibe algún subsidio) |
|  | "P6585s1 alimentación" "P6585s2 transporte" "P6585s3 familiar" |
|  | "P6585s4 educativo" "1=si, 9=NS" |
|  |  |
|  | "P6800 (suma de horas 1er trabajo por hogar)" |
|  | "P6920 (cotiza 1=si, 2=no, 3=pensionado) jefe hogar " |
|  | "P7045 (suma horas 2do trabajo por hogar) " |
|  | "PET+OC trabajo infantil dummy" |
|  | "TD=Des ds/pet x hogar" |
|  |  |
|  | "estrato, sexo\_jefe\_hog,edad\_jefe\_hog, hacinamiento, |
|  | paug,rs\_jefe\_hog, educ\_jefe\_hog, ocupacion\_jefe\_hog, |
|  | sub\_hog, horas\_tra\_hogar, cot\_jefe\_hog,horas\_tra\_hogar\_2, |
|  | sum\_pet, sum\_des" |
|  |  |
|  | 26 variables |
|  | } |
|  |  |
|  | #----------------------------------- C o n s t r u c c i o n d e l a b a s e --------------------------------- |
|  | #Estrato (no está en test) |
|  | estrato<-train\_personas %>% group\_by(id) %>% summarize(Estrato1=mean(Estrato1,na.rm = TRUE)) |
|  | table(estrato, train\_personas$Estrato1) |
|  |  |
|  | #Sexo jefe hogar |
|  | sex\_jefe\_hog<- as.data.frame(ifelse((train\_personas$P6020==1 & train\_personas$P6050==1),1,0)) |
|  | train\_personas<- cbind(train\_personas, sex\_jefe\_hog) |
|  | names(train\_personas)[names(train\_personas)=='ifelse((train\_personas$P6020 == 1 & train\_personas$P6050 == 1), 1, 0)']<-'sexo\_jefe\_hogar' |
|  | sexo\_jefe\_hog<-train\_personas %>% group\_by(id) %>% summarize(sexo\_jefe\_hogar=sum(sexo\_jefe\_hogar,na.rm = TRUE)) |
|  | summary(sexo\_jefe\_hog) |
|  | #Edad jefe hogar |
|  | edad\_jefe\_hogar<- (ifelse((train\_personas$P6050==1),train\_personas$P6040,0)) |
|  | train\_personas<- cbind(train\_personas, edad\_jefe\_hogar) |
|  | edad\_jefe\_hog<-train\_personas %>% group\_by(id) %>% summarize(edad\_jefe\_hogar=sum(edad\_jefe\_hogar,na.rm = TRUE)) |
|  | summary(edad\_jefe\_hog) |
|  |  |
|  | #Hacinamiento |
|  | hacinamiento=train\_hogares$Nper/train\_hogares$P5010 |
|  | train\_hogares<-cbind(train\_hogares, hacinamiento) |
|  |  |
|  | #Personas que aportan a la unidad de gasto (PAUG) |
|  | paug=(train\_hogares$Npersug/train\_hogares$Nper) |
|  | train\_hogares<-cbind(train\_hogares, paug) |
|  | table(paug) |
|  |  |
|  | #Régimen de salud jefe de hogar |
|  | rs\_jefe\_hogar<- (ifelse((train\_personas$P6050==1),train\_personas$P6100,0)) |
|  | train\_personas<- cbind(train\_personas, rs\_jefe\_hogar) |
|  | rs\_jefe\_hog<-train\_personas %>% group\_by(id) %>% summarize(rs\_jefe\_hogar=sum(rs\_jefe\_hogar,na.rm = TRUE)) |
|  | summary(rs\_jefe\_hog) |
|  | table(train\_personas$P6050, train\_personas$P6100) |
|  |  |
|  | #Nivel educativo jefe hogar |
|  | edu\_jefe\_hogar<- (ifelse((train\_personas$P6050==1),train\_personas$P6210,0)) |
|  | train\_personas<- cbind(train\_personas, edu\_jefe\_hogar) |
|  | educ\_jefe\_hog<-train\_personas %>% group\_by(id) %>% summarize(edu\_jefe\_hogar=sum(edu\_jefe\_hogar,na.rm = TRUE)) |
|  | summary(educ\_jefe\_hog) |
|  | table(train\_personas$P6050, train\_personas$P6210) |
|  |  |
|  | # Ocupación jefe hogar |
|  | ocupacion\_jefe\_hogar<- (ifelse((train\_personas$P6050==1),train\_personas$P6430,0)) |
|  | train\_personas<- cbind(train\_personas, ocupacion\_jefe\_hogar) |
|  | ocupacion\_jefe\_hog<-train\_personas %>% group\_by(id) %>% summarize(ocupacion\_jefe\_hogar=sum(ocupacion\_jefe\_hogar,na.rm = TRUE)) |
|  | summary(ocupacion\_jefe\_hog) |
|  | table(train\_personas$P6050, train\_personas$P6430) |
|  |  |
|  | #Subsidios |
|  | train\_hogares<-train\_hogares[,-30] |
|  | train\_personas<-train\_personas[,-141] |
|  |  |
|  | train\_personas$P6585s1 = ifelse((train\_personas$P6585s1==2),0,train\_personas$P6585s1) |
|  | train\_personas$P6585s2 = ifelse((train\_personas$P6585s2==2),0,train\_personas$P6585s2) |
|  | train\_personas$P6585s3 = ifelse((train\_personas$P6585s3==2),0,train\_personas$P6585s3) |
|  | train\_personas$P6585s4 = ifelse((train\_personas$P6585s4==2),0,train\_personas$P6585s4) |
|  | train\_personas$P6585s1 = ifelse((train\_personas$P6585s1==9),0,train\_personas$P6585s1) |
|  | train\_personas$P6585s2 = ifelse((train\_personas$P6585s2==9),0,train\_personas$P6585s2) |
|  | train\_personas$P6585s3 = ifelse((train\_personas$P6585s3==9),0,train\_personas$P6585s3) |
|  | train\_personas$P6585s4 = ifelse((train\_personas$P6585s4==9),0,train\_personas$P6585s4) |
|  | subsidio<-(train\_personas$P6585s1+train\_personas$P6585s2+train\_personas$P6585s3+train\_personas$P6585s4) |
|  | train\_personas<-cbind(train\_personas,subsidio) |
|  | sub\_hog<-train\_personas %>% group\_by(id) %>% summarize(subsidio=sum(subsidio,na.rm = TRUE)) |
|  |  |
|  | #Suma de horas trabajadas por hogar |
|  | horas\_tra\_hogar<-train\_personas %>% group\_by(id) %>% summarize(P6800=mean(P6800,na.rm = TRUE)) |
|  |  |
|  | #Cotiza pensión jefe de hogar |
|  | cot\_jefe\_hogar<- (ifelse((train\_personas$P6050==1),train\_personas$P6920,0)) |
|  | train\_personas<- cbind(train\_personas, cot\_jefe\_hogar) |
|  | cot\_jefe\_hog<-train\_personas %>% group\_by(id) %>% summarize(cot\_jefe\_hogar=sum(cot\_jefe\_hogar,na.rm = TRUE)) |
|  |  |
|  | #Suma de horas trabajadas segundo empleo por hogar |
|  | horas\_tra\_hogar\_2<-train\_personas %>% group\_by(id) %>% summarize(P7045=mean(P7045,na.rm = TRUE)) |
|  |  |
|  | #TD por hogar |
|  | #reemplazar missing de des por 0 |
|  | train\_personas$Des = ifelse(is.na(train\_personas$Des)==T,0,train\_personas$Des) |
|  | #Suma Pet |
|  | sum\_pet<-train\_personas %>% group\_by(id) %>% summarize(Pet=sum(Pet,na.rm = TRUE)) |
|  | #Suma Des |
|  | sum\_des<-train\_personas %>% group\_by(id) %>% summarize(Des=sum(Des,na.rm = TRUE)) |
|  |  |
|  |  |
|  | #-------Merge |
|  | train\_hogares<-left\_join(train\_hogares, sum\_des) |
|  | train\_hogares<-left\_join(train\_hogares, sexo\_jefe\_hog) |
|  | train\_hogares<-left\_join(train\_hogares, edad\_jefe\_hog) |
|  | train\_hogares<-left\_join(train\_hogares, hacinamiento) |
|  | train\_hogares<-left\_join(train\_hogares, paug) |
|  | train\_hogares<-left\_join(train\_hogares, rs\_jefe\_hog) |
|  | train\_hogares<-left\_join(train\_hogares, educ\_jefe\_hog) |
|  | train\_hogares<-left\_join(train\_hogares, ocupacion\_jefe\_hog) |
|  | train\_hogares<-left\_join(train\_hogares, sub\_hog) |
|  | train\_hogares<-left\_join(train\_hogares, horas\_tra\_hogar) |
|  | train\_hogares<-left\_join(train\_hogares, cot\_jefe\_hog) |
|  | train\_hogares<-left\_join(train\_hogares, horas\_tra\_hogar\_2) |
|  | train\_hogares<-left\_join(train\_hogares, sum\_pet) |
|  | train\_hogares<-left\_join(train\_hogares, sum\_des) |
|  | colnames(train\_hogares) |
|  |  |
|  |  |
|  | #-------Renonbrar variables y cambiar missings |
|  | names(train\_hogares)[names(train\_hogares)=='P5000']<-'Ncuartos' |
|  | names(train\_hogares)[names(train\_hogares)=='P5010']<-'Ncuartos\_dormir' |
|  | names(train\_hogares)[names(train\_hogares)=='P5090']<-'Tipo\_vivienda' |
|  | names(train\_hogares)[names(train\_hogares)=='P6800']<-'Horas\_trabajo1' |
|  | names(train\_hogares)[names(train\_hogares)=='P7045']<-'Horas\_trabajo2' |
|  | train\_hogares$P5130 = ifelse(is.na(train\_hogares$P5130)==T,0,train\_hogares$P5130) |
|  | train\_hogares$P5140 = ifelse(is.na(train\_hogares$P5140)==T,0,train\_hogares$P5140) |
|  | train\_hogares$Horas\_trabajo1 = ifelse(is.na(train\_hogares$Horas\_trabajo1)==T,0,train\_hogares$Horas\_trabajo1) |
|  | train\_hogares$Horas\_trabajo2 = ifelse(is.na(train\_hogares$Horas\_trabajo2)==T,0,train\_hogares$Horas\_trabajo2) |
|  |  |
|  | #-------Eliminar y crear variables |
|  | train\_hogares<-train\_hogares[,-2] |
|  | train\_hogares<-train\_hogares[,-6] |
|  | arriendo\_estimado<-train\_hogares$P5130+train\_hogares$P5140 |
|  | train\_hogares<-cbind(train\_hogares,arriendo\_estimado) |
|  | train\_hogares<-train\_hogares[,-6] |
|  | train\_hogares<-train\_hogares[,-6] |
|  | train\_hogares<-train\_hogares[,-8] |
|  | train\_hogares<-train\_hogares[,-10] |
|  | train\_hogares<-train\_hogares[,-12] |
|  | train\_hogares<-train\_hogares[,-13] |
|  | train\_hogares<-train\_hogares[,-13] |
|  | train\_hogares<-train\_hogares[,-14] |
|  |  |
|  | TD<-train\_hogares$Des/train\_hogares$Pet |
|  | train\_hogares<-cbind(train\_hogares,TD) |
|  | train\_hogares<-train\_hogares[,-26] |
|  | train\_hogares<-train\_hogares[,-25] |
|  | train\_hogares$subsidio<-(ifelse((train\_hogares$subsidio>0),1,0)) |
|  | #back up |
|  | train\_hogares2<-train\_hogares |
|  | #Logaritmo del ingreso percapita |
|  | ing\_per = train\_hogares2$Ingpcug |
|  | ing\_per<-ifelse((ing\_per)==0,1,ing\_per) |
|  | log\_ing\_per<- log(ing\_per) |
|  | train\_hogares2<-cbind(train\_hogares2,log\_ing\_per) |
|  |  |
|  |  |
|  | # Variables como factor |
|  | train\_hogares2$Dominio<-as.factor(train\_hogares2$Dominio) |
|  | train\_hogares2$Tipo\_vivienda<-as.factor(train\_hogares2$Tipo\_vivienda) |
|  | train\_hogares2$rs\_jefe\_hogar<-as.factor(train\_hogares2$rs\_jefe\_hogar) |
|  | train\_hogares2$edu\_jefe\_hogar<-as.factor(train\_hogares2$edu\_jefe\_hogar) |
|  | train\_hogares2$ocupacion\_jefe\_hogar<-as.factor(train\_hogares2$ocupacion\_jefe\_hogar) |
|  |  |
|  | #----------------------------------- E s t a d í s t i c a s D e s c r i p t i v a s --------------------------------- |
|  |  |
|  | table (train\_hogares$Pobre) #131936 33024 |
|  |  |
|  | #----------------------------------- D i v i s i ó n d e M u e s t r a --------------------------------- |
|  | set.seed(12345) |
|  | train\_hogares2 <- train\_hogares2 %>% |
|  | mutate(holdout= as.logical(1:nrow(train\_hogares2) %in% |
|  | sample(nrow(train\_hogares2), nrow(train\_hogares2)\*.2)) |
|  | ) |
|  | test<-train\_hogares2[train\_hogares2$holdout==T,] #32.992 |
|  | train<-train\_hogares2[train\_hogares2$holdout==F,] #131.968 |
|  |  |
|  | #----------------------------------- M o d e l o s d e e s t i m a c i ó n --------------------------------- |
|  | #-----------#Modelo 1 y 2 |
|  | modelo1<-lm(log\_ing\_per~ Tipo\_vivienda+ rs\_jefe\_hogar+edu\_jefe\_hogar+ Ncuartos + Ncuartos\_dormir+ |
|  | Nper+Npersug+ hacinamiento+edad\_jefe\_hogar+Horas\_trabajo1+Horas\_trabajo2+ |
|  | arriendo\_estimado+ocupacion\_jefe\_hogar+Dominio+subsidio+sexo\_jefe\_hogar |
|  | +cot\_jefe\_hogar, data=train) |
|  | summary(modelo1) |
|  |  |
|  | modelo2<-lm(log\_ing\_per~ Tipo\_vivienda+ rs\_jefe\_hogar+edu\_jefe\_hogar+ Ncuartos\_dormir+ |
|  | Npersug+ hacinamiento+edad\_jefe\_hogar+Horas\_trabajo1+Horas\_trabajo2+ |
|  | arriendo\_estimado+ocupacion\_jefe\_hogar+Dominio+subsidio, data=train) |
|  | summary(modelo2) |
|  |  |
|  |  |
|  | #-----------#MSE |
|  | test$modelo1<-predict(modelo1,newdata = test) |
|  | with(test,mean((log\_ing\_per-modelo1)^2)) #1.082514 |
|  |  |
|  | test$modelo2<-predict(modelo2,newdata = test) |
|  | with(test,mean((log\_ing\_per-modelo2)^2)) #1.100224 |
|  |  |
|  |  |
|  | #Gráfico de coeficientes modelo 2 |
|  |  |
|  | df\_coeficientes\_reg2 <- modelo2$coefficients %>% |
|  | enframe(name = "predictor", value = "coeficiente") |
|  |  |
|  | df\_coeficientes\_reg2[-1,] %>% |
|  | filter(predictor != "`(Intercept)`") %>% |
|  | ggplot(aes(x = reorder(predictor, abs(coeficiente)), |
|  | y = coeficiente)) + |
|  | geom\_col(fill = "darkblue") + |
|  | coord\_flip() + |
|  | labs(title = "Coeficientes del modelo de regresión", |
|  | x = "Variables", |
|  | y = "Coeficientes") + |
|  | theme\_bw() |
|  |  |
|  | #-----------Evaluamos el modelo de regresión lineal |
|  | y\_hat\_in1 <- predict(modelo2, newdata = train) |
|  | y\_hat\_out1 <- predict(modelo2, newdata = test) |
|  |  |
|  | # Métricas dentro y fuera de muestra. Paquete MLmetrics |
|  | r2\_in1 <- R2\_Score(y\_pred = exp(y\_hat\_in1), y\_true = (train$Ingpcug)) |
|  | rmse\_in1 <- RMSE(y\_pred = exp(y\_hat\_in1), y\_true = (train$Ingpcug)) |
|  |  |
|  | r2\_out1 <- R2\_Score(y\_pred = exp(y\_hat\_out1), y\_true = (test$Ingpcug)) |
|  | rmse\_out1 <- RMSE(y\_pred = exp(y\_hat\_out1), y\_true = (test$Ingpcug)) |
|  |  |
|  | resultados <- data.frame(Modelo = "Regresión lineal", |
|  | Muestra = "Dentro", |
|  | R2\_Score = r2\_in1, RMSE = rmse\_in1) %>% |
|  | rbind(data.frame(Modelo = "Regresión lineal", |
|  | Muestra = "Fuera", |
|  | R2\_Score = r2\_out1, RMSE = rmse\_out1)) |
|  |  |
|  | #------------------------------------------------------ L A S S O ------------------------------------------------------- |
|  | #-----------Estandarizar variables continuas train |
|  | train[,c(4,7,14,18,22,24,26)]<- scale(train[,c(3,4,6,7,8,14,18,22,24,26)],center=T,scale=T) |
|  | train[,c(3,4,6,7,14,18,22,24,26)]<- scale(train[,c(3,4,6,7,8,14,18,22,24,26)],center=T,scale=T) |
|  |  |
|  | #Estandarizar variables continuas test |
|  | test[,c(3,4,6,7,8,14,18,22,24,26)]<- scale(test[,c(3,4,6,7,8,14,18,22,24,26)], center=T,scale=T) |
|  | x\_continuas=scale(train[,c(4,7,14,18,22,24,26)]) |
|  | x\_categoricas=scale(train[,c(2,5,17,19,20,21,25)]) |
|  | # Ahora procedemos a dummyficar la base |
|  | x\_categoricas<- model.matrix(~ ., x\_categoricas) %>% |
|  | as.data.frame #%>% |
|  | cx<- cbind(x\_categoricas,x\_continuas) |
|  |  |
|  | modelo\_lasso <- glmnet( |
|  | x = cx, |
|  | y = train$log\_ing\_per, |
|  | alpha = 1, |
|  | nlambda = 1000, |
|  | standardize = TRUE |
|  | ) |
|  |  |
|  | #Lasso para lambda distinto (establecer grilla) |
|  |  |
|  | # Analicemos cómo cambian los coeficientes para diferentes lambdas |
|  | regularizacion <- modelo\_lasso$beta %>% |
|  | as.matrix() %>% |
|  | t() %>% |
|  | as\_tibble() %>% |
|  | mutate(lambda = modelo\_lasso$lambda) |
|  |  |
|  | regularizacion <- regularizacion %>% |
|  | pivot\_longer( |
|  | cols = !lambda, |
|  | names\_to = "predictor", |
|  | values\_to = "coeficientes" |
|  | ) |
|  |  |
|  | #--------Gráfico de Lasso |
|  | regularizacion %>% |
|  | ggplot(aes(x = lambda, y = coeficientes, color = predictor)) + |
|  | geom\_line() + |
|  | scale\_x\_log10( |
|  | breaks = scales::trans\_breaks("log10", function(x) 10^x), |
|  | labels = scales::trans\_format("log10", |
|  | scales::math\_format(10^.x)) |
|  | ) + |
|  | labs(title = "Coeficientes del modelo en función de la regularización (Lasso)", x = "Lambda", y = "Coeficientes") + |
|  | theme\_bw() + |
|  | theme(legend.position="bottom") |
|  |  |
|  | # ¿Cómo escoger el mejor lambda? |
|  | # Veamos cuál es el mejor prediciendo (fuera de muestra) |
|  | # En este caso vamos a crear la predicción para cada uno de los |
|  | # 5000 lambdas seleccionados |
|  | predicciones\_lasso <- predict(modelo\_lasso, |
|  | newx = as.matrix(cx)) |
|  | lambdas\_lasso <- modelo\_lasso$lambda |
|  |  |
|  | ------------# Cada predicción se va a evaluar |
|  | resultados\_lasso <- data.frame() |
|  | for (i in 1:length(lambdas\_lasso)) { |
|  | l <- lambdas\_lasso[i] |
|  | y\_hat\_out2 <- predicciones\_lasso[, i] |
|  | r22 <- R2\_Score(y\_pred = y\_hat\_out2, y\_true = test$log\_ing\_per) |
|  | rmse2 <- RMSE(y\_pred = y\_hat\_out2, y\_true = test$log\_ing\_per) |
|  | resultado <- data.frame(Modelo = "Lasso", |
|  | Muestra = "Fuera", |
|  | Lambda = l, |
|  | R2\_Score = r22, |
|  | RMSE = rmse2) |
|  | resultados\_lasso <- bind\_rows(resultados\_lasso, resultado) |
|  | } |
|  |  |
|  | #-----------Gráfico RMSE Lasso |
|  | ggplot(resultados\_lasso, aes(x = Lambda, y = RMSE)) + |
|  | geom\_point() + |
|  | geom\_line() + |
|  | theme\_bw() + |
|  | scale\_y\_continuous(labels = scales::comma) |
|  | #-----------Gráfico R2 |
|  | ggplot(resultados\_lasso, aes(x = Lambda, y = R2\_Score)) + |
|  | geom\_point() + |
|  | geom\_line() + |
|  | theme\_bw() + |
|  | scale\_y\_continuous(labels = scales::comma) |
|  |  |
|  |  |
|  | filtro <- resultados\_lasso$RMSE == min(resultados\_lasso$RMSE) |
|  | mejor\_lambda\_lasso <- resultados\_lasso[filtro, "Lambda"] |
|  |  |
|  | # Guardamos el mejor Lasso |
|  | y\_hat\_in2 <- predict.glmnet(modelo\_lasso, |
|  | newx = as.matrix(cx), |
|  | s = mejor\_lambda\_lasso) |
|  | y\_hat\_out2 <- predict.glmnet(modelo\_lasso, |
|  | newx = as.matrix(cx), |
|  | s = mejor\_lambda\_lasso) |
|  |  |
|  | # Métricas dentro y fuera de muestra. Paquete MLmetrics |
|  | r2\_in2 <- R2\_Score(y\_pred = exp(y\_hat\_in2), y\_true = exp(train$log\_ing\_per)) |
|  | rmse\_in2 <- RMSE(y\_pred = exp(y\_hat\_in2), y\_true = exp(train$log\_ing\_per)) |
|  |  |
|  | r2\_out2 <- R2\_Score(y\_pred = exp(y\_hat\_out2), y\_true = exp(test$log\_ing\_per)) |
|  | rmse\_out2 <- RMSE(y\_pred = exp(y\_hat\_out2), y\_true = exp(test$log\_ing\_per)) |
|  |  |
|  | # Guardamos el desempeño |
|  | resultados2 <- data.frame(Modelo = "Lasso", |
|  | Muestra = "Dentro", |
|  | R2\_Score = r2\_in2, RMSE = rmse\_in2) %>% |
|  | rbind(data.frame(Modelo = "Lasso", |
|  | Muestra = "Fuera", |
|  | R2\_Score = r2\_out2, RMSE = rmse\_out2)) |
|  |  |
|  | # Juntamos resultados con regresión lineal |
|  | resultados <- rbind(resultados, resultados2) |
|  |  |
|  | #----------------------------------------------------- R I D G E ------------------------------------------------------ |
|  |  |
|  | #------------------------------------------------------------------------------------------------------------------------------------- |
|  | # M o d e l o s d e C l a s i f i c a c i ó n |
|  | #------------------------------------------------------------------------------------------------------------------------------------ |
|  |  |
|  | #----------------------------------- B a l a n c e o d e l a m u e s t r a --------------------------------- |
|  | #----------------- Gráfico de desbalanceo |
|  | ggplot(train, aes(x = Pobre)) + |
|  | geom\_bar(fill = "darkblue") + |
|  | theme\_bw() + |
|  | labs(title = " ¿El hogar es pobre? |
|  | si=1, no=0", |
|  | x = "", |
|  | y = "Frecuencia") + |
|  | coord\_flip() |
|  | #----------------- Balancear la muestra |
|  | prop.table(table(train$Pobre)) #0.8001031 0.1998969 |
|  |  |
|  | #Dummyficamos ANTES de partir la base en train/test |
|  | train\_s <- data.frame(train\_s) |
|  | test\_s <- data.frame(test\_s) |
|  | train <- data.frame(train) |
|  | test <- data.frame(test) |
|  |  |
|  | train\_s$infielTRUE <- as.numeric(train\_s$infielTRUE) |
|  | modelo1 <- lm(formula = infielTRUE ~ ., data = train\_s) |
|  | probs\_insample1 <- predict(modelo1, train\_s) |
|  | probs\_insample1[probs\_insample1 < 0] <- 0 |
|  | probs\_insample1[probs\_insample1 > 1] <- 1 |
|  | probs\_outsample1 <- predict(modelo1, test\_s) |
|  | probs\_outsample1[probs\_outsample1 < 0] <- 0 |
|  | probs\_outsample1[probs\_outsample1 > 1] <- 1 |
|  |  |
|  | # Convertimos la probabilidad en una predicción |
|  | y\_hat\_insample1 <- as.numeric(probs\_insample1 > 0.5) |
|  | y\_hat\_outsample1 <- as.numeric(probs\_outsample1 > 0.5) |
|  |  |
|  | acc\_insample1 <- Accuracy(y\_pred = y\_hat\_insample1, y\_true = train$infielTRUE) |
|  | acc\_outsample1 <- Accuracy(y\_pred = y\_hat\_outsample1, y\_true = test$infielTRUE) |
|  |  |
|  | pre\_insample1 <- Precision(y\_pred = y\_hat\_insample1, y\_true = train$infielTRUE, positive = 1) |
|  | pre\_outsample1 <- Precision(y\_pred = y\_hat\_outsample1, y\_true = test$infielTRUE, positive = 1) |
|  |  |
|  | rec\_insample1 <- Recall(y\_pred = y\_hat\_insample1, y\_true = train$infielTRUE, positive = 1) |
|  | rec\_outsample1 <- Recall(y\_pred = y\_hat\_outsample1, y\_true = test$infielTRUE, positive = 1) |
|  |  |
|  | f1\_insample1 <- F1\_Score(y\_pred = y\_hat\_insample1, y\_true = train$infielTRUE, positive = 1) |
|  | f1\_outsample1 <- F1\_Score(y\_pred = y\_hat\_outsample1, y\_true = test$infielTRUE, positive = 1) |
|  |  |
|  | metricas\_insample1 <- data.frame(Modelo = "Regresión lineal", |
|  | "Muestreo" = NA, |
|  | "Evaluación" = "Dentro de muestra", |
|  | "Accuracy" = acc\_insample1, |
|  | "Precision" = pre\_insample1, |
|  | "Recall" = rec\_insample1, |
|  | "F1" = f1\_insample1) |
|  |  |
|  | metricas\_outsample1 <- data.frame(Modelo = "Regresión lineal", |
|  | "Muestreo" = NA, |
|  | "Evaluación" = "Fuera de muestra", |
|  | "Accuracy" = acc\_outsample1, |
|  | "Precision" = pre\_outsample1, |
|  | "Recall" = rec\_outsample1, |
|  | "F1" = f1\_outsample1) |
|  |  |
|  | metricas1 <- bind\_rows(metricas\_insample1, metricas\_outsample1) |
|  | metricas1 %>% |
|  | kbl(digits = 2) %>% |
|  | kable\_styling(full\_width = T) |
|  |  |
|  | #---------------------- Oversamplig |
|  | # Implementamos oversampling |
|  | train$PobreTRUE <- factor(train$PobreTRUE) |
|  | train\_s2 <- recipe(ProbreTRUE ~ ., data = x) %>% |
|  | themis::step\_smote(PobreTRUE, over\_ratio = 1) %>% |
|  | prep() %>% |
|  | bake(new\_data = NULL) |
|  |  |
|  | prop.table(table(x$PobreTRUE)) |
|  |  |
|  |  |
|  | #----------------------------------- L a s s o - L o g i t ----------------------------------------- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  | #---------------------------------------- T r e e s --------------------------------- |
|  |  |
|  | #--------------------------------- R a n d o m F o r e s t --------------------------------- |
|  | #--------------- Tunear la grilla |
|  | gbmGrid <- expand.grid(interaction.depth = c(1, 5, 9), |
|  | n.trees = (1:30)\*50, |
|  | shrinkage = 0.1, |
|  | n.minobsinnode = 20) |
|  |  |
|  | nrow(gbmGrid) |
|  |  |
|  | set.seed(825) |
|  | gbmFit2 <- train(Class ~ ., data = training, |
|  | method = "gbm", |
|  | trControl = fitControl, |
|  | verbose = FALSE, |
|  | ## Now specify the exact models |
|  | ## to evaluate: |
|  | tuneGrid = gbmGrid) |
|  | gbmFit2 |
|  |  |
|  |  |
|  |  |
|  | #---------------------------------M E J O R A R M O D E L O G A N A D O R --------------------------------- |
|  | #------------------------------------------------ T U N E A R --------------------------------------- |
|  |  |
|  | #------------Métricas |
|  | The function trainControl generates parameters that further control how models are created, with possible values: |
|  |  |
|  | #---ROC |
|  | set.seed(825) |
|  | gbmFit3 <- train(Class ~ ., data = training, |
|  | method = "gbm", |
|  | trControl = fitControl, |
|  | verbose = FALSE, |
|  | tuneGrid = gbmGrid, |
|  | ## Specify which metric to optimize |
|  | metric = "ROC") |
|  | gbmFit3 |
|  | #-------------- Model tuning: Maximizar la capacidd predictiva del modelo (para logit) |
|  |  |
|  | ctrl def <- trainControl(method = "cv", |
|  | number = 5, |
|  | summaryFunction = defaultSummary, |
|  | classProbs = TRUE, |
|  | verbose=FALSE, |
|  | savePredictions = T) |
|  | #\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Accuracy y Kappa |
|  | set.seed(1410) |
|  | mylogit caret def <- train( |
|  | Default ~amount+installment+age+ historygood + historypoor + purposeusedcar+ purposegoods.repair + purposeedu + foreigngerman + rentTRdata = training, |
|  | method = "glm", #for logit |
|  | trControl = ctrl def, |
|  | family = "binomial", |
|  | preProcess = c("center", "scale") |
|  | ) |
|  | mylogit caret def |
|  |  |
|  |  |
|  | ctrl two <- trainControl(method = "cv", |
|  | number = 5, |
|  | summaryFunction = twoClassSummary, |
|  | classProbs = TRUE, |
|  | verbose=FALSE, |
|  | savePredictions = T) |
|  |  |
|  | #\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ROC, Sensibilidad y especificidad |
|  |  |
|  | set.seed(1410) |
|  | mylogit caret two <- train( |
|  | Default ~amount+installment+age+ historygood + historypoor + purposeusedcar+ purposegoods.repair + purposeedu + foreigngerman + rentTRdata = training, |
|  | method = "glm", #for logit |
|  | trControl = ctrl two, |
|  | family = "binomial", |
|  | preProcess = c("center", "scale") |
|  | ) |
|  |  |
|  |  |
|  | #------------ 5 Métricas juntas |
|  |  |
|  | fiveStats <- function(...) c(twoClassSummary(...), defaultSummary(...)) |
|  | ctrl<- trainControl(method = "cv", |
|  | number = 5, |
|  | summaryFunction = fiveStats, |
|  | classProbs = TRUE, |
|  | verbose=FALSE, |
|  | savePredictions = T) |
|  | #logit |
|  | set.seed(1410) |
|  | mylogit caret <- train( |
|  | Default ~amount+installment+age+ historygood + historypoor + purposeusedcar+ purposegoods.repair + purposeedu + foreigngerman + rentTRdata = training, |
|  | method = "glm", #for logit |
|  | trControl = ctrl, |
|  | family = "binomial", |
|  | preProcess = c("center", "scale") |
|  | ) |
|  |  |
|  | #-------------- Model tuning: Maximizar la capacidd predictiva del modelo (para losso) |
|  | #Model(glmnet) method (glmnet) Type (Classification), Regression libraries(glmnet, Matrix) Tunning parameters (alpha, lambda) |
|  | #-----Lasso |
|  | lambda grid <- 10^seq(-4, 0.01, length = 10) #en la practica se suele usar una grilla de 200 o 300 |
|  | lambda grid |
|  |  |
|  | #------------ 5 Métricas juntas |
|  |  |
|  | set.seed(1410) |
|  | mylogit lasso acc <- train( |
|  | Default ~amount+installment+age+ historygood + historypoor + purposeusedcar+ purposegoods.repair + purposeedu + foreigngerman + rentTRdata = training, |
|  | method = "glmnet", |
|  | trControl = ctrl, |
|  | family = "binomial", |
|  | metric = "Accuracy", |
|  | tuneGrid = expand.grid(alpha = 0,lambda=lambda grid), |
|  | preProcess = c("center", "scale") |
|  | ) |
|  |  |
|  | mylogit lasso acc |
|  |  |
|  | #------ROC |
|  | set.seed(1410) |
|  | mylogit lasso roc <- train( |
|  | Default ~amount+installment+age+ historygood + historypoor + purposeusedcar+ purposegoods.repair + purposeedu + foreigngerman + rentTRdata = training, |
|  | method = "glmnet", |
|  | trControl = ctrl, |
|  | family = "binomial", |
|  | metric = "ROC", |
|  | tuneGrid = expand.grid(alpha = 0,lambda=lambda grid), |
|  | preProcess = c("center", "scale") |
|  | ) |
|  | #------ Sensibilidad |
|  |  |
|  | set.seed(1410) |
|  | mylogit caret sens <- train( |
|  | Default ~amount+installment+age+ historygood + historypoor + purposeusedcar+ purposegoods.repair + purposeedu + foreigngerman + rentTRdata = training, |
|  | method = "glmnet", |
|  | trControl = ctrl, |
|  | family = "binomial", |
|  | metric = "Sens", |
|  | tuneGrid = expand.grid(alpha = 0,lambda=lambda grid), |
|  | preProcess = c("center", "scale") |
|  | ) |
|  |  |
|  | #------ Cutoff óptimo |
|  |  |
|  | evalResults <- data.frame(Default = evaluation$Default) |
|  | evalResults$Roc <- predict(mylogit lasso roc, |
|  | newdata = evaluation, |
|  | type = "prob")[,1] |