# Machine Learning

## Introducción al problema

Se va a estudiar la concesión de hipotecas entre 1997 y 1998 en Boston de acuerdo a una serie de variables. Los datos provienen de la reserva federal de Boston, y pueden ser encontrados en https://vincentarelbundock.github.io/Rdatasets/doc/Ecdat/Hdma.html

La variable objetivo es “deny”: es una variable categórica binaria, Si el valor es “Yes” se rechaza la hipoteca, en cambio, cuando el valor es “no” se concede la hipoteca. EL objetivo del problema es construir un modelo clasificador binario de la variable deny en función de unas variables independientes.

Hay 12 variables independientes, de ellas 6 son continuas y 6 son categóricas.

Las variables continuas son las siguientes:

* Dir: el ratio entre deuda y el salario
* hir: ratio de los gastos de la casa y el salario
* lvr: ratio entre tamaño de la deuda y el valor de la propiedad
* ccs: puntuación de credito del consumidor del 1 al 6 (cuanto menor sea la puntuación mejor puntuación es)
* mcs: puntuación de crédito de la hipoteca del 1 al 4 (cuanto menor sea la puntuación mejor puntuación es)
* uria: la tasa de desempleo en el sector del solicitante en Massachusetts de 1989

Las variables categóricas son las siguientes:

* Pbcr: ¿ El solicitante tiene malos registros crediticios públicos? Las categorías de esta variable son “Yes” o “no”.
* dmi: ¿ Se le ha denegado el seguro de hipoteca al solicitante? Las categorías de esta variable son “Yes” o “no”.
* self: ¿Es el solicitante autónomo? Las categorías de esta variable son “Yes” o “no”.
* single: ¿Es el solicitante soltero? Las categorías de esta variable son “Yes” o “no”.
* condominium: ¿La casa es un condominio? Las categorías de esta variable son “Yes” o “no”.
* black ¿El solicitante es negro? Las categorías de esta variable son “Yes” o “no”.

## Exploración de datos

* Missings?

## Desbalance de categorias

La representación de las categorias están desbalanceados

# no yes

# 2096 285

#Desbalance 7.33

Esto, unido a que los algoritmos tienden a entrenar los modelos hacia una tasa de aciertos máxima provoca que intenten maximizar la categoría dominante a costa de la minoritaria.

En este problema esto es especialmente relevante porque un falso negativo significa que estamos concediendo una hipoteca a alguien que puede que no llegue a pagarla, incurriendo en coste para la entidad.

Para solventar este problema se va a utilizar un método de generación de dato sintético. Esta técnica genera observaciones de la clase mayoritaria usando la distribución de la clase minoritaria. Vamos a usar el paquete ROSE

En las fases siguientes será necesario usar validación cruzada. Sin embargo, no podemos usar simplemente los datos originales. Así que he construido funciones que separan el dataset en k grupos, de estos k-1 son transformados con el paquete ROSE y usados para entrenar el modelo correspondiente. El testeo de las predicción se hace con la parte orignal. Este proceso se repite k-1 veces.

He construido una función para cada algoritmo que uso, y son las siguientes:

* RangerDesbalance: usa una librería llamada ranger para hacer random forest
* GlmDesbalance: regresión logística.

## Selección de variables

En la selección de variables he utilizado dos aproximaciones: lineal y no-lineal. En cada una de las dos aproximaciones obtendremos conjuntos de variables tentativos por distintos métodos y después los compararemos con validación cruzada. Las variables finales se obtendrán de comparar los mejores conjuntos de variables de cada una de las dos aproximaciones usando validación cruzada.

Las métricas para comparar modelos son accuracy, sensibilidad y ROC. En general, en clasificación binaria son importantes accuracy y ROC. En un dataset desbalanceado normalmente los modelos tienden a tener sesgo a favor de la clase dominante, en este caso los no evento (aceptación de la hipoteca), esto se traduce en sensbilidad muy baja porque gran parte de los eventos se predicen como falsos negativos.

En este dataset la sensibilidad es significativa porque este parámetro mide como bien el modelo es capaz de predecir los eventos (denegación de hipoteca, en este caso).

Entonces, la estrategia será intentar hallar una sensibilidad lo mayor posible manteniendo los valores de accuracy y ROC lo más altos posibles.

### Aproximación lineal

Se han utilizado el método stepwise con remuestreo en los datos con ajustes de bondad AIC y BIC. Se ha realizado en cada uno el remuestreo 10 veces (cada una de las veces con distinta semilla) y con un porcentaje de datos de 0,8. Además se ha modificado levelmente la función steprepetidobinaria para que los datos training se balanceen con el paquete ROSE.

Con el ajuste de bondad AIC estos son los conjuntos de variables más frecuentes:

#c("ccs", "dir", "lvr", "mcs", "uria")

#c("ccs", "dir", "lvr", "mcs", "uria","hir")

Con el ajuste de bondad AIC estos son los conjuntos de variables más frecuentes:

#c("ccs", "dir", "lvr", "mcs", "uria")

#c("ccs", "dir", "lvr", "mcs", "uria","hir")

Con el ajuste de bondad BIC este es el conjunto de variables más frecuentes:

#c("ccs", "lvr", "dir", "mcs", "uria")

Este conjunto de variable tiene las misma variables que el primero con AIC.

Además, también se ha utilizado el método stepwise con todos las muestras. El conjunto de variable obtenido es

c( "dmi", "ccs", "dir", "pbcr", "black", "lvr", "self", "single", "uria", "mcs")

Comparamos estos tres conjuntos de variables mediante la función GlmDesbalance, como se ha comentado en la sección anterior, esta función realiza validación cruzada usando en el training datos balanceados con la librería ROSE .

|  |  |  |  |
| --- | --- | --- | --- |
| Modelo | Conjunto variables | Método | Numero variables |
| Modelo 1 | c("ccs", "dir", "lvr", "mcs", "uria") | AIC y BIC con remuestreo | 5 |
| Modelo 2 | c("ccs", "dir", "lvr", "mcs", "uria","hir") | AIC con remuestreo | 6 |
| Modelo 3 | c( "dmi", "ccs", "dir", "pbcr", "black", "lvr","self", "single", "uria", "mcs") | #AIC entero | 10 |

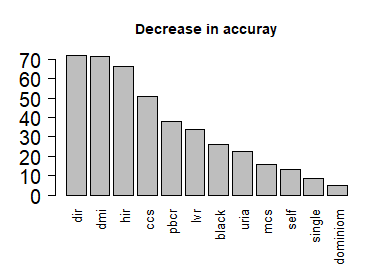
|  |  |
| --- | --- |
| D:\Master_UCM\Machine Learning\Boston\Figuras\V06_seleccion_variables_lineal_Acc.png | D:\Master_UCM\Machine Learning\Boston\Figuras\V06_seleccion_variables_lineal_Sens.png |
| D:\Master_UCM\Machine Learning\Boston\Figuras\V06_seleccion_variables_lineal_ROC.png |  |

En primer lugar descartamos el modelo 3, el aumento en la sensibilidad de alrededor de 0.03 que obtenemos con este conjunto de variables no justifica añadir tantas variables al modelo.

Elijo el modelo 1 frente al modelo 2, porque la variable extra no aporta casi nada.

### Aproximación no-lineal

En primer lugar se ha utilizado un random forest para saber la importancia de las variables sobre el dataset balanceado con la librería ROSE.



Diferentes conjuntos de variables se comparan mediante validación cruzada con la función RangerDesbalance. La aproximación será iterativa, se irán añadiendo variables poco a poco, comparando modelos.

Se empezará con un modelo con las tres variables más importantes c("dir", "dmi", "hir").Además, se ha incluido un modelo con todas las posibles variables para hacernos una idea de cuanto puede mejorar con todas las variables

|  |  |  |
| --- | --- | --- |
| Modelo | Conjunto variables | Numero variables |
| Modelo 1 | c("dir", "dmi", "hir", "ccs", "pbcr", "lvr", "black", "uria",  "mcs", "self", "single", "comdominiom") | 12 |
| Modelo 2 | c("dir", "dmi", "hir") | 3 |
| Modelo 3 | c("dir", "dmi", "hir","ccs") | 4 |
| Modelo 4 | c("dir", "dmi", "hir","pbcr" ) | 4 |
| Modelo 5 | c("dir", "dmi", "hir", "lvr") | 4 |
| Modelo 6 | c("dir", "dmi", "hir","black") | 4 |
| Modelo 7 | c("dir", "dmi", "hir","uria" ) |  |
| Modelo 8 | c("dir", "dmi", "hir","mcs") | 4 |
| Modelo 9 | c("dir", "dmi", "hir","self") | 4 |
| Modelo 10 | c("dir", "dmi", "hir","single" ) | 4 |
| Modelo 11 | c("dir", "dmi", "hir","comdominiom") | 4 |

Desde los modelos 3 al 11 son iguales al modelo 2 salvo que se ha añadido una variable. Entonces, el modelo 2 se usará como referencia para comparar.

|  |  |
| --- | --- |
| C:\Users\Adrián\AppData\Local\Microsoft\Windows\INetCache\Content.Word\V06_seleccion_variables_Nolineal_Rf_Accu(1-11).png | D:\Master_UCM\Machine Learning\Boston\Figuras\V06_seleccion_variables_Nolineal_Rf_Sens(1-11).png |
| C:\Users\Adrián\AppData\Local\Microsoft\Windows\INetCache\Content.Word\V06_seleccion_variables_Nolineal_Rf_ROC(1-11).png |  |

-

Los modelos 3,4,5,6 producen mejoras en las 3 métricas. Especialmente destaca el modelo 3, que es el que más mejora y además reduce la varianza en las métricas, la variable asociada a este modelo es “css”. Además, como las variables asociadas a los modelos del 7 al 11 no dan resultados claros las descartamos. El modelo 1, que contiene todas las variables, tiene muy más sensibilidad respecto al 2, parece aconsejable seguir añadiendo variables.

En la siguiente ronda de comparación se añaden dos variables al modelo con respecto al modelo2: combinaciones de a dos de las variables asociadas a los modelos 3,4,5,6; que corresponden a “css”, “pbcr", "lvr", “black” respectivamente.

Ahora el modelo de referencia será el modelo 11, que esta formado por las mismas variables que el anterior modelo 3. Así podremos decir sin compensa seguir añadiendo variables o, si por el contrario, es mejor tener un modelo más parsimonioso.

|  |  |  |
| --- | --- | --- |
| Modelo | Conjunto variables | Numero variables |
| Modelo 11 | c("dir", "dmi", "hir","ccs") | 4 |
| Modelo 12 | c("dir", "dmi", "hir","ccs","pbcr") | 5 |
| Modelo 13 | c("dir", "dmi", "hir","ccs","lvr") | 5 |
| Modelo 14 | c("dir", "dmi", "hir","ccs","black") | 5 |
| Modelo 15 | c("dir", "dmi", "hir","pbcr","lvr" ) | 5 |
| Modelo 16 | c("dir", "dmi", "hir","pbcr","black" ) | 5 |
| Modelo 17 | c("dir", "dmi", "hir", "lvr","black") | 5 |

|  |  |
| --- | --- |
| D:\Master_UCM\Machine Learning\Boston\Figuras\V06_seleccion_variables_Nolineal_Rf_AC(11-17).png | D:\Master_UCM\Machine Learning\Boston\Figuras\V06_seleccion_variables_Nolineal_Rf_Sens(11-17).png |
|  |  |

Destacan los modelos 12, 13, 14. Todos ellos tienen en común la variable “css”.

Vamos a realizar una nueva iteración, esta vez teniendo como referencia el modelo 18, que es igual al 11. Esta vez se van a añadir a este modelo las variables “pbcr”,”lvr”,”black”, y sus distintas combinaciones.

|  |  |  |
| --- | --- | --- |
| Modelo | Conjunto variables | Numero variables |
| Modelo 18 | c("dir", "dmi", "hir","ccs") | 4 |
| Modelo 19 | c("dir", "dmi", "hir","ccs","pbcr") | 5 |
| Modelo 20 | c("dir", "dmi", "hir","ccs","lvr") | 5 |
| Modelo 21 | c("dir", "dmi", "hir","ccs", "black") | 5 |
| Modelo 22 | c("dir", "dmi", "hir","ccs","pbcr","lvr") | 6 |
| Modelo 23 | c("dir", "dmi", "hir","ccs","pbcr","black") | 6 |
| Modelo 24 | c("dir", "dmi", "hir","ccs","lvr","black") | 6 |
| Modelo 25 | c("dir", "dmi", "hir","ccs","pbcr","lvr","black") | 7 |

|  |  |
| --- | --- |
| C:\Users\Adrián\AppData\Local\Microsoft\Windows\INetCache\Content.Word\V06_seleccion_variables_Nolineal_Rf_Ac(18-25).png | C:\Users\Adrián\AppData\Local\Microsoft\Windows\INetCache\Content.Word\V06_seleccion_variables_Nolineal_Rf_Sens(18-25).png |
| C:\Users\Adrián\AppData\Local\Microsoft\Windows\INetCache\Content.Word\V06_seleccion_variables_Nolineal_Rf_ROC(18-25).png |  |

Las figuras muestran que cuanto más variables más mejoran lasa métricas de ROC y sensibilidad. Como modelos tentativos he escogido los modelos 23 y 25. El modelo 25 supera la barrera 0.6 en sensibilidad y es modelo roc mas alto. El modelo 24 ha sido descartado por su accuracy. Se ha escogido el 23 frente al 22 porque tiene algo más de sensibilidad y valor ROC.

### Selección entre conjuntos de variables

Los modelos escogidos en las secciones anteriores se van comparar vía validación cruzada usando el mismo algoritmo(random forest). Uso este modo de proceder porque lo que busco es un conjunto de variables, no un modelo ya entrenado. Esa parte se dará en fases posteriores.

|  |  |  |
| --- | --- | --- |
| Modelo | Variables | número variables |
| Lineal | c("ccs", "dir", "lvr", "mcs", "uria") | 5 |
| No lineal 1 (antiguo modelo23) | c("dir", "dmi", "hir","ccs","pbcr","black") | 6 |
| No lineal 2  (antiguo modelo25) | c("dir", "dmi", "hir","ccs","pbcr","lvr","black") | 7 |

|  |  |
| --- | --- |
| C:\Users\Adrián\AppData\Local\Microsoft\Windows\INetCache\Content.Word\V06_Comparacion_modelos-Acc.png | C:\Users\Adrián\AppData\Local\Microsoft\Windows\INetCache\Content.Word\V06_Comparacion_modelos-Sens.png |
| C:\Users\Adrián\AppData\Local\Microsoft\Windows\INetCache\Content.Word\V06_Comparacion_modelos-ROC.PNG |  |

El modelo escogido a través de la aproximación lineal es claramente inferior a los otros dos. Escojo el modelo no lineal 2 porque tiene valores ROC y de sensibilidad superiores a modelo no lineal 1.

## Tuneado de parámetros

### Redes

|  |
| --- |
| size Accuracy  1 5 0.5968648  2 10 0.6213575  3 15 0.5621925  4 20 0.5363577  > aggregate(Sensitivity~size,total1,mean)  size Sensitivity  1 5 0.8119832  2 10 0.7101767  3 15 0.8135232  4 20 0.8069477  > aggregate(auc~size,total1,mean)  size auc  1 5 0.7893173  2 10 0.7280241  3 15 0.7495380  4 20 0.7303063 |
|  |
| |  | | --- | | > | |

Con 10 nodos mejora la sensibilidad a costa de un sesgo en los positivos.

La precisión mejora entre 5 y 15 nodos

Sensivitividad empeora entre 5 y 15, en los demas estable

Valor mayor con 5 nodos y decae hasata 10 y se mantiene constante

Conclusion: Explorar entre 5 y 10 nodos para tener un buen auc, y un compromiso entre accuracy y sensitividad

aggregate(Accuracy~decay,total1,mean)

decay Accuracy

1 0.001 0.5679940

2 0.010 0.5673493

3 0.100 0.6022360

> aggregate(Sensitivity~decay,total1,mean)

decay Sensitivity

1 0.001 0.7955208

2 0.010 0.7906989

3 0.100 0.7707534

> aggregate(auc~decay,total1,mean)

decay auc

1 0.001 0.7554989

2 0.010 0.7421072

3 0.100 0.7502832

AC: Learning rate is igual hasta 0.01, y sube

Sensivity: baja desde 0.01 hasta 0.1

Auc: se mantine igual

Explorar:

aggregate(Accuracy~size,total12,mean)

size Accuracy

1 5 0.5968648

2 6 0.5911067

3 7 0.6119669

4 8 0.6041577

5 9 0.6264182

6 10 0.6213575

> aggregate(Sensitivity~size,total12,mean)

size Sensitivity

1 5 0.8119832

2 6 0.7832161

3 7 0.7955410

4 8 0.7753245

5 9 0.7579766

6 10 0.7101767

> aggregate(auc~size,total12,mean)

size auc

1 5 0.7893173

2 6 0.7666261

3 7 0.7738772

4 8 0.7591536

5 9 0.7608727

6 10 0.7280241

|  |
| --- |
| size decay Accuracy Sensitivity auc parametros  1 5 0.100 0.6192987 0.8121554 0.7898657 71  2 10 0.100 0.6546008 0.6427838 0.7105935 141  5 5 0.010 0.6088692 0.8073867 0.7870120 71  6 10 0.010 0.6110874 0.7345839 0.7288580 141  9 5 0.001 0.5624265 0.8164075 0.7910741 71  10 10 0.001 0.5983843 0.7531624 0.7446207 141  13 6 0.100 0.5859715 0.7831596 0.7714039 85  21 7 0.100 0.6072426 0.7901893 0.7745250 99  3 8 0.100 0.6199337 0.7548665 0.7427267 113  4 9 0.100 0.6692717 0.7336836 0.7718264 127  51 6 0.010 0.5770352 0.7874902 0.7558352 85  61 7 0.010 0.6052140 0.8070735 0.7670796 99  7 8 0.010 0.6078169 0.7690090 0.7614983 113  8 9 0.010 0.6098962 0.7367632 0.7346841 127  91 6 0.001 0.6103134 0.7789984 0.7726392 85  101 7 0.001 0.6234441 0.7893601 0.7800271 99  11 8 0.001 0.5847226 0.8020981 0.7732360 113  12 9 0.001 0.6000868 0.8034829 0.7761076 127 |
|  |
| |  | | --- | | > | |

Elegimos size 5 decay 0.1 parametros 71

Tiene la segunda sensitividad, pero lo hace mejor en accuracy bajando poca sensitividad.

Hay otra combinación con mayor accuracy pero baja mucho la sensitividad nodos 9, decay 0.1