

RECOMENDACIÓN DE CLASIFICADORES PARA EL SECTOR CREDITICIO

MINERÍA DE DATOS EN NEGOCIOS

Ágatha del Olmo Tirado | 2ºBIA | 09/10/2023



VNIVERSITAT
DE VALÈNCIA

INTELIGENCIA Y ANALÍTICA DE NEGOCIOS

ÍNDICE

1. Introducción	1
2. Tareas previas en el explorer de Weka	2-4
2.1. El clasificador ZeroR y su significado	2
2.2. Relevancia de la variable “capacidad de ahorro” con OneR	2-3
2.3. Efecto de una clasificación sensible al coste con ZeroR y OneR	3
2.4. Evaluación de Clasificadores Estadísticos: Regresión Logística Y Análisis Lineal Discriminante (LDA)	4
3. Análisis de los diferentes clasificadores	5-16
3.1. Análisis de los clasificadores Naivebayes, logística, J48, JRip Y Hoeffding Tree en el experimenter de weka	5-9
3.2. Análisis del clasificador LDA en RStudio	9-11
3.3. Análisis del clasificador RPart en RStudio	11-14
3.4. Análisis del clasificador de red neuronal en RStudio	14-16
4. Conclusiones y recomendaciones	16-17
5. Bibliografía	18

INTRODUCCIÓN

Este informe aborda una investigación centrada en la tarea de clasificación de clientes bancarios sobre la base de datos "creditscore". El objetivo principal de este estudio es desarrollar el mejor clasificador posible para distinguir entre clientes que tendrán un "estatus bueno" en relación a su préstamo y aquellos que tendrán un "estatus malo" o que no lo devolverán adecuadamente.

Para llevar a cabo esta investigación, se utilizarán diferentes métodos y herramientas informáticas, incluyendo el uso de clasificadores como JRip, Logística o Naïve-Bayes y otras técnicas disponibles en plataformas como R y Weka. La base de datos "creditscore" se compone de 15 variables y contiene información de más de 3845 clientes, lo que proporciona un conjunto de datos robusto para realizar los análisis.

En este informe, se detallará el proceso de investigación, desde la selección de clasificadores hasta la evaluación de diferentes métricas y la consideración de los costos asociados a las decisiones de clasificación incorrectas.

TAREAS PREVIAS EN EL EXPLORER DE WEKA

Empezaremos usando Weka, entorno gráfico que nos permite crear y analizar experimentos sobre tareas de clasificación, y en concreto su interfaz Explorer. Durante todo el proceso estaremos usando la semilla “o”.

EL CLASIFICADOR ZEROR Y SU SIGNIFICADO

En el proceso de investigación, uno de los primeros enfoques ha sido la utilización del clasificador ZeroR. Este clasificador arroja resultados sorprendentemente altos en términos de precisión global. Sin embargo, para comprender plenamente el motivo detrás de este desempeño aparentemente fantástico, es esencial considerar el contexto en el que se encuentra la base de datos "creditscore" y este clasificador.

Por un lado, la precisión es una métrica que representa la fracción de elementos clasificados correctamente como positivos de entre todos los que el modelo ha clasificado como positivos, a: $VP / (VP + FP)$. Por otro lado, ZeroR es un clasificador muy simple que se basa en catalogar todos los individuos como la clase mayoritaria de un conjunto de datos. Esto quiere decir que, como sería de esperar, la mayoría de los clientes paga sus préstamos, pues todos los falsos positivos son los del grupo “bueno” en el que hemos metido a todos los “malos”, y si la precisión da muy alta quiere decir que hay una gran diferencia en número entre los grupos. Esta observación tiene grandes implicaciones en el contexto de la tarea de clasificación, podemos esperar que la clasificación del modelo sea desbalanceada y tenga más clientes con buen estatus que con malo, la desproporción deberá aproximarse.

RELEVANCIA DE LA VARIABLE “CAPACIDAD DE AHORRO” CON ONER

El uso del clasificador OneR ha arrojado resultados excelentes en términos de acierto global, lo cual puede hacerlo parecer un clasificador muy interesante. Sin embargo, es fundamental entender cómo funciona este.

El clasificador OneR busca de entre todas las variables cuál, al usarla para clasificar, da el menor error total. La variable discriminante que este usa es la “capacahorro”, es decir, la capacidad de ahorro de cada cliente. Aunque puede parecer lógico utilizar esta variable dada su relevancia en el contexto financiero, al examinar las reglas propuestas para la clasificación, surgen cuestiones sobre su idoneidad.

Cuando OneR utiliza una variable para predecir sobre otra, se espera que la variable seleccionada sea la más discriminante de todas. Esto significa que debe ser capaz de dividir los datos en grupos con la mayor heterogeneidad intergrupal posible. Sin embargo, en este caso, la variable “capacidad de ahorro” parece no cumplir completamente con esta expectativa.

El principal problema radica en que, en lugar de establecer divisiones claras y efectivas, el clasificador OneR crea clasificaciones basadas en capas o franjas de valores de capacidad de ahorro. Como vemos en la imagen, la interpretación carece de sentido: por ejemplo, la franja de -57 a -34 es clasificado como "bueno", mientras que la franja de -34 a -28.5 es clasificada como "malo".

```
capacahorro:  
< -57.0 -> malo  
< -34.0 -> bueno  
< -28.5 -> malo  
< -13.5 -> bueno  
< -10.74      -> malo  
>= -10.74      -> bueno  
(3619/3845 instances correct)
```

Si su forma de clasificar se basara en “capacidad mala vs capacidad buena”, sería un clasificador, como mínimo, con mucho más sentido de interpretación, pero al usar franjas, se permite que, en algunas instancias, el clasificador cometa errores al clasificar una mala capacidad de ahorro como buena o viceversa.

EFECTO DE UNA CLASIFICACIÓN SENSIBLE AL COSTE EN ZEROR Y ONER

Dada la condición de esta base de datos, el error de clasificar una persona incapaz de devolver el préstamo como capaz de hacerlo conlleva un gran coste que representaremos con un 3 en la matriz de costes, ya que el coste monetario que este error tiene es más directo. Para estudiar su efecto podemos ver cómo se modifica al usar los clasificadores ZeroR y OneR.

El clasificador ZeroR, como ya hemos comentado, mete a la totalidad de individuos dentro del grupo con mayor frecuencia, que, en este caso el de “estatus bueno”, es decir, la mayoría es capaz de devolver el préstamo. Por lo tanto, nos encontramos con un error de los del grupo b o “estatus malo” completo, pues ninguno está bien clasificado como b. Hay 281 instancias “missclassified”, que es la totalidad de las b, que, aunque no representen una grandísima parte de la base de datos, al darle un costo triplicado, se obtiene un costo total considerable de 843. En conclusión, nos encontramos con que este clasificador no nos interesa en absoluto porque no se adapta a la naturaleza de los datos.

Con el clasificador OneR se obtiene un costo total menor, de 610. Este costo surge de la suma de los clientes “buenos” clasificados incorrectamente como “malos”, con un costo de 1 cada uno y un total de instancias mal clasificadas de 58, dando 58, y los “malos” clasificados como “buenos”, con un costo de 3 cada uno y 184 instancias, que da 549.

La diferencia clave radica en cómo ZeroR, dado que clasifica a todos los clientes como “buenos”, se enfrenta inevitablemente a un alto costo cuando se cometen errores; mientras que OneR usa un criterio algo más “sofisticado” al basarse en la “capacidad de ahorro” como variable principal, y aunque comete errores en la clasificación, reduce el costo total en comparación con el otro clasificador.

EVALUACIÓN DE CLASIFICADORES ESTADÍSTICOS: REGRESIÓN LOGÍSTICA Y ANÁLISIS LINEAL DISCRIMINANTE (LDA)

En esta etapa del análisis, hemos empleado los clasificadores funcionales, Regresión Logística y Análisis Lineal Discriminante (LDA) y el de árbol de decisión Naïve Bayes.

En principio, el LDA no puede utilizarse en esta base de datos debido a una restricción fundamental: requiere que todas las variables de entrada sean numéricas. Sin embargo, en la base de datos "creditscore" existen atributos nominales que no son numéricos, y esto plantea un desafío.

Ante esta limitación, se ha optado por binarizar las variables nominales, para que sean compatibles con el LDA. Esto se logra utilizando el filtro "NominalToBinary", que crea variables binarias para cada nivel del factor presente en el atributo nominal. Sin embargo, la binarización puede aumentar significativamente la dimensionalidad de los datos y puede tener efectos negativos en algunos clasificadores, por ejemplo, se ha observado que, en particular, Naïve Bayes muestra una mayor sensibilidad a la binarización, mientras que la Regresión Logística apenas se ve afectada. Esto se debe a que Naïve asume una independencia condicional, la cual se ve muy afectada, pues creamos variables que pueden parecer independientes mientras que no lo son, y en cambio, la Regresión Logística no hace unas suposiciones tan fuertes y es más capaz de manejar grandes cantidades de datos.

Después de la binarización, persiste un problema significativo en la base de datos: el elevado número de instancias con valores perdidos. Para abordar este problema, se ha utilizado el filtro "ReplaceMissingValues", que sustituye los valores perdidos por la media o la moda de sus respectivos atributos.

Con estos preprocesamientos, se ha podido aplicar tanto el Análisis Lineal Discriminante (LDA) como la Regresión Logística y el Naïve Bayes a un conjunto de datos ya coherente y comparable a través del clasificador de la interfaz Explorer de Weka. Los resultados son los siguientes:

	Logística	LDA	Naïve Bayes
Coste total	81	229	195
Porcentaje de aciertos	98.9857%	97.6333%	96.697%
Precisión en la clase "buenos"	0,994	0,981	0,99
Precisión en la clase "malos"	0,932	0,906	0,726
Media ponderada de la precisión	0,99	0,975	0,971

Los mejores resultados los hemos indicado en verde, y hay un claro ganador: Logística. A pesar de que LDA y Naïve Bayes, han demostrado un buen desempeño en la clasificación de los clientes en función de su estatus, logística ha tenido el mejor de todos los rendimientos con creces.

ANÁLISIS DE LOS DIFERENTES CLASIFICADORES

ANÁLISIS DE LOS CLASIFICADORES NAÏVEBAYES, LOGÍSTICA, J48, JRIP Y HOEFFDING TREE EN EL EXPERIMENTER DE WEKA

Ahora usamos la interfaz Experimenter de Weka. Esta interfaz nos interesa más que el Explorer porque nos muestra comparación de hipótesis para determinar si el mejor o peor rendimiento de un clasificador u otro es realmente significativo o no. Usamos la base de datos binarizada y la nominal, con los valores perdidos sustituidos previamente. Para llevar a cabo un análisis lo más completo posible vamos a comparar 5 clasificadores distintos: Naïvebayes, logística, J48, JRip y Hoeffding Tree. Las características a tener en cuenta de estos clasificadores para entenderlos mejor son las siguientes:

- Naïvebayes: este clasificador se basa en la obtención de probabilidades condicionadas y su salida es una probabilidad de pertenencia al grupo x. A pesar de que sea rápido y fácil de interpretar, su capacidad se ve limitada al “duplicar” las variables tras binarizarlas ya que las supondrá independientes cuando son claramente dependientes. En cambio, los resultados suelen ser muy acertados porque aún así el orden de las probabilidades no suele alterarse mucho.
- Logística: este clasificador se basa en una función logística y se usa en problemas de clasificación precisamente binaria, donde el resultado solo puede ser de naturaleza dicotómica. Este es sensible a la correlación y existe el riesgo de sobreajuste, pero al igual que en el Naïvebayes, los resultados siguen siendo los mismos a pesar de que se aleje algo la estimación a la que hubiera sido.
- J48: este clasificador es una versión del clásico C4.5 pero más acotado, construye un árbol de decisión a partir del nodo raíz creando subconjuntos y minimizando la variabilidad. En este se suelen implementar métodos pre-poda y post-poda para reducir el error.
- JRip: este clasificador basado en reglas busca patrones y los aplica a los nuevos datos a través del algoritmo de poda incremental repetida para la reducción de errores (RIPPER) tanto en la poda como en su crecimiento.
- Hoeffding Tree: este clasificador va “creciendo” un árbol de decisión, donde un nodo se expande cuando hay suficiente evidencia estadística de que hay una división óptima futura. Es simple de implementar e interpretar, pero en cuanto crea un nodo no lo puede cambiar, no puede corregirse.

Para analizarlos nos fijaremos en diversos indicadores como Percent_correct, Kappa_statistic, TruePositiveRate, FalsePositiveRate, IR_Precision y F_Mesure.

Primero estudiamos qué significa cada uno:

- Percent Correct: porcentaje de individuos correctamente clasificados sobre el total.
- Kappa_statistic: Compara la acuracidad con la que se obtendría al azar $\kappa = \frac{\Pr(a) - \Pr(e)}{1 - \Pr(e)}$,
- TruePositiveRate: también llamado sensibilidad, ratio de verdaderos positivos sobre el total de positivos del dataset. $TPR = \frac{TP}{Actual\ Positive} = \frac{TP}{TP + FN}$

- FalsePositiveRate: también llamado especificidad, ratio de falsos positivos sobre el total de negativos del dataset.
- IR_Precision: Calcula el número de individuos correctamente clasificados respecto al total de positivos,
- F_Mesure: una media ponderada harmónica entre precisión y exhaustividad que provee de un buen balance entre estos.
- Area_under_ROC: área que queda bajo la curva ROC en un gráfico que muestra en el eje y el TPR y en el eje x el FPR, mide en general el desempeño del modelo.

$$FPR = \frac{FP}{Actual\ Negative} = \frac{FP}{TN + FP}$$

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$F1\ Score = \frac{2}{\frac{1}{\text{Precision}} + \frac{1}{\text{Recall}}}$$

Cada clasificador se representará con un código entre paréntesis, siendo (1) J48, (2)JRip, (3)Hoeffding Tree, (4)NaïveBayes y (5)Logística. El contraste de hipótesis se realiza en todo momento con un 5% de nivel de significación.

1. ANÁLISIS DEL PERCENT CORRECT

Dataset	(1) trees.J4 (2) rules (3) trees (4) bayes (5) funct
---------	--

'WekaExcel-weka.filters.u(100)	97.17	97.17	93.21 *	96.79	99.01 v
--------------------------------	-------	-------	---------	-------	---------

'WekaExcel-weka.filters.u(100)	97.13	97.23	92.93 *	97.30	99.01 v
--------------------------------	-------	-------	---------	-------	---------

(v/ /*) (o/1/o) (o/o/1) (o/1/o) (1/o/o)

Logística ha obtenido la mayor precisión, con un 99.01%, esto puede sugerir que, en este conjunto de datos específico, las relaciones entre las características y la variable objetivo son predominantemente lineales. En cambio, el Hoeffding Tree ha obtenido una precisión del 93.21% en la primera base y un 92.93% en la segunda, valores significativamente inferiores a los demás, los cuales pueden deberse a su incapacidad de aprender de sus errores y retroceder en la construcción del árbol.

2. ANÁLISIS DEL KAPPA STATISTIC

Dataset	(1) trees.J (2) rule (3) tree (4) baye (5) func
---------	---

'WekaExcel-weka.filters.u(100)	0.79	0.79	0.36 *	0.78	0.93 v
--------------------------------	------	------	--------	------	--------

'WekaExcel-weka.filters.u(100)	0.78	0.80	0.35 *	0.80	0.93 v
--------------------------------	------	------	--------	------	--------

(v/ /*) (o/1/o) (o/o/1) (o/1/o) (1/o/o)

Logística ha obtenido un Kappa Statistic de 0.93, lo que indica una concordancia muy alta entre sus predicciones y los valores reales, y es significativamente efectiva en este conjunto de datos. Por otro lado, el Hoeffding Tree ha obtenido un Kappa Statistic de 0.36 y 0.35 en cada base de datos, lo que indica una concordancia significativamente baja en comparación con los demás.

3. ANÁLISIS DEL TRUE POSITIVE RATE

Dataset	(1) trees.J (2) rule (3) tree (4) baye (5) func
---------	---

'WekaExcel-weka.filters.u(100)	0.99 0.98 0.98 0.97 *	1.00 v
--------------------------------	-------------------------	--------

'WekaExcel-weka.filters.u(100)	0.99 0.98 0.98 0.99	1.00 v
--------------------------------	-----------------------	--------

(v/ /*) (o/1/o) (o/1/o) (o/o/1) (1/o/o)

Logística ha alcanzado el True Positive Rate más alto, con un valor de 1.00. Esto indica que la Regresión Logística es significativamente efectiva en la identificación de instancias positivas en el conjunto de datos, logrando una detección considerada perfecta. En cambio, Naïve Bayes ha obtenido un True Positive Rate significativamente inferior, con un valor de 0.97.

4. ANÁLISIS DEL FALSE POSITIVE RATE

Dataset	(1) trees.J (2) rule (3) tree (4) baye (5) func
---------	---

'WekaExcel-weka.filters.u(100)	0.21 0.19 0.66 v 0.12 *	0.07 *
--------------------------------	---------------------------	--------

'WekaExcel-weka.filters.u(100)	0.22 0.17 0.67 v 0.20	0.07 *
--------------------------------	-------------------------	--------

(v/ /*) (o/1/o) (1/o/o) (o/o/1) (o/o/1)

En este caso medimos a la inversa, buscamos el clasificador con el valor más bajo. Naïve Bayes y Regresión Logística han obtenido un False Positive Rate bastante bajo, con un valor de 0.12 y 0.07 respectivamente. Esto sugiere son significativamente las más efectivas para evitar falsos positivos, lo que la convierte en una excelente opción para minimizar este tipo de errores. El Hoeffding Tree tiene, por otro lado, el False Positive Rate más alto, con un valor de 0.66. Esto sugiere que este es significativamente menos efectivo en comparación con los otros clasificadores evaluados.

5. ANÁLISIS DEL IR PRECISION

Dataset (1) trees.J | (2) rule (3) tree (4) baye (5) func

'WekaExcel-weka.filters.u(100) 0.98 | 0.99 0.95 * 0.99 v 0.99 v

'WekaExcel-weka.filters.u(100) 0.98 | 0.99 0.95 * 0.98 0.99 v

(v/ /*) | (o/1/o) (o/o/1) (1/o/o) (1/o/o)

Naïve Bayes y Logística han obtenido un valor de IR Precision de 0,99, lo que es una indicación positiva para ser significativamente (Naïve Bayes es únicamente significativa para la primera base de datos, la binarizada) eficientes en la precisión. Pero el Hoeffding Tree tiene el valor más bajo de IR Precision, con 0.95, y esto sugiere que este logra una precisión significativamente menos eficiente en comparación con los otros clasificadores evaluados.

6. ANÁLISIS DEL F MESURE

Dataset (1) trees.J | (2) rule (3) tree (4) baye (5) func

'WekaExcel-weka.filters.u(100) 0.98 | 0.98 0.96 * 0.98 0.99 v

'WekaExcel-weka.filters.u(100) 0.98 | 0.99 0.96 * 0.99 0.99 v

(v/ /*) | (o/1/o) (o/o/1) (o/1/o) (1/o/o)

La Regresión Logística ha alcanzado el valor más alto de F-Measure, con 0.99. Esto sugiere que la Regresión Logística es capaz de clasificar correctamente instancias positivas y negativas sin comprometer significativamente ninguna de las dos. Por otro lado, el Hoeffding Tree tiene un valor significativamente inferior de F-Measure, con 0.96, así que muestra un equilibrio menos eficiente entre precisión y recall en comparación con los otros clasificadores evaluados.

7. ANÁLISIS DEL AREA UNDER ROC

Dataset	(1) trees.J (2) rule (3) tree (4) baye (5) func
---------	---

'WekaExcel-weka.filters.u(100)	0.92 0.90 0.65 * 0.98 v 1.00 v
--------------------------------	----------------------------------

'WekaExcel-weka.filters.u(100)	0.92 0.91 0.65 * 0.98 v 1.00 v
--------------------------------	----------------------------------

(v/ /*) (o/1/o) (o/o/1) (1/o/o) (1/o/o)

La Regresión Logística ha obtenido el valor más alto, con 1.00. Esto sugiere que la Regresión Logística es perfectamente efectiva en la discriminación entre clases positivas y negativas, lo que la convierte en una opción sobresaliente. Naïve Bayes también ha obtenido un valor alto, con 0.98. Ambas tienen una capacidad significativamente alta en comparación con el resto. Sin embargo, el Hoeffding Tree tiene el valor más bajo, con 0.65. Esto sugiere que es significativamente menos efectivo en comparación con los otros clasificadores evaluados.

En resumen, la Regresión Logística ha demostrado consistentemente y de forma significativa el mejor rendimiento durante la totalidad de la evaluación. Esto sugiere que, en este conjunto de datos específico y solo fijándonos en los resultados del experimento en WEKA, la Regresión Logística es el clasificador preferido para la tarea de clasificación. Algo interesante a comentar es el hecho de que no cambiara el valor en ninguna métrica en ambas bases de datos (binarizada y no binarizada) al usar este clasificador, y es que no se espera que la binarización de las variables tenga impacto en el rendimiento de la regresión logística, pues si no ha sido convertido manualmente como hicimos en la primera base de nominales a binarios, lo hace de forma automática a través de la técnica “one-hot-encoding”, que viene a resultar en lo mismo.

ANÁLISIS DEL CLASIFICADOR LDA EN RSTUDIO

Para continuar con la comparación de los clasificadores tenemos que usar R. Primero analizamos el LDA. Esta función discriminante se basa en el aprendizaje automático, buscando las características que más diferencien los grupos maximizando la distancia entre las medias de las clases y minimizando la varianza interclase. Vamos a necesitar nuestra base de datos binarizada, pero para que funcione correctamente le cambiamos las cabeceras de las duplicadas en Excel a propiedad, alquiler, otras_viviendas, etc.

Tras importar la base de datos y realizar un lda() con el paquete “MASS” hemos obtenido la matriz de confusión, que nos ayuda a ver más claramente el resultado.

	o (bueno)	1 (malo)
o	3543	21
1	67	214

La matriz de confusión muestra la cantidad de instancias que se clasificaron correctamente (diagonal) y las que se clasificaron incorrectamente. El eje X representa los valores reales, y el eje Y representa los valores predichos. En la clase o (buen estatus), 3543 instancias se clasificaron correctamente, y 21 se clasificaron incorrectamente. En la clase 1 (mal estatus), 214 instancias se clasificaron correctamente, pero 67 se clasificaron incorrectamente. Es interesante tener en cuenta que nos preocupa el segundo error más que el otro ya que tiene un impacto financiero directo con la empresa, y este, en este caso es mayor, lo cual si añadimos el hecho de que hay una proporción mucho menor de “malos”, es un error muy alto, y esto lo vemos más claro en el porcentaje de los correctamente clasificados:

Clase	o (bueno)	1 (malo)
% Correctamente Clasificados	99.41%	76.16%

A pesar de tener un porcentaje de total de aciertos que queda en 97.71%, ocurre lo que temíamos: a pesar de tener un rendimiento general sólido, hay bastante más error de clasificación del segundo grupo que en el primero.

Después hemos realizado un análisis de discriminante lineal con la función lda(), que nos muestra las probabilidades previas de pertenecer a las clases "o" y "1", de las cuales es mucho más alta para la clase o.

o (bueno)	1 (malo)
0.92691808	0.07308192

Esto tiene sentido si pensamos en el contexto del problema, pues tal como deberíamos esperar, hay muchas menos personas que no devuelven los préstamos.

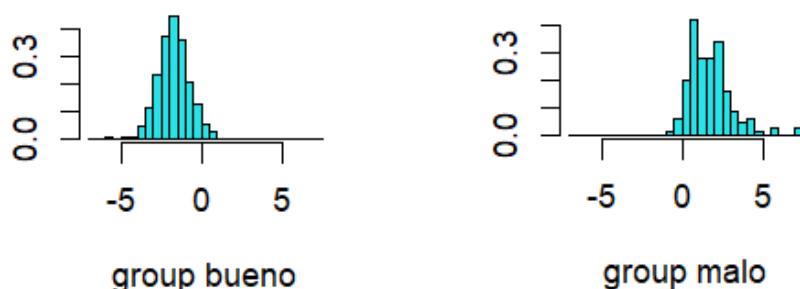
También obtenemos los coeficientes de discriminación. Estos coeficientes indican cómo se combinan las variables predictoras para realizar la clasificación.

Variables	Coeficientes de discriminación
antiguedad	-1.091369e-02
propiedad	-1.690713e-01
alquiler	-1.006806e-01
otras_viviendas	6.157888e-01
tiempoPres	-1.457970e-02
edad	-4.875063e-03
casado	2.524644e-03
viudo	3.735027e-01
soltero	-1.148735e-01
separado	3.596654e-01
divorciado	4.095982e-01
fijo	-5.839712e-01

temporal	-3.474206e-01
freelance	6.603696e-01
otros_trabajos	1.219505e+00
gastos	3.673844e-03
ingresos	1.383552e-02
activos	-2.086010e-05
deudas	5.432841e-05
importe	2.398743e-03
precio	-2.584014e-04
financiado	-8.614060e-03
capacahorro	-2.155696e-02

Las variables con un coeficiente positivo (otras_viviendas, casado, viudo, separado, divorciado, freelance, otros_trabajos, gastos, ingresos, deudas e importe) indican que su mayor valor está asociado a la clase 1, y las que tienen un coeficiente negativo (antiguedad, propiedad, alquiler, tiempoPres, edad, soltero, fijo, temporal, activos, precio, financiado, capacahorro) indican que su mayor valor se asocia a la pertenencia del grupo 0.

Por último, hemos configurado un gráfico que muestra la clasificación:



En resumen, se observa un alto porcentaje total de aciertos, lo que sugiere que el modelo es eficaz en la clasificación. Sin embargo, también se observa que la clase "0" tiene un mayor porcentaje de clasificación correcta en comparación con la clase "1".

En cuanto a la comparación entre este y el de logística, podemos estudiar la precisión, la Regresión Logística tuvo una precisión del 99.01%, mientras que el LDA tuvo una precisión del 97.71%. En términos de precisión, la Regresión Logística supera al LDA. Dado que la Regresión Logística tiene un porcentaje correcto más alto y proporciona métricas adicionales, parece ser el clasificador preferido en este contexto específico.

ANÁLISIS DEL CLASIFICADOR RPART EN RSTUDIO

Ahora analizaremos el clasificador RPart para compararlo con los demás, de nuevo con RStudio. El RPart se centra en dividir el conjunto de datos en subconjuntos más eficientes a partir de un árbol C4.5.

Para que podamos usar la función rpart() pasamos a numéricas todas las variables en RStudio.

El output de rpart() es meramente predictivo en este caso, y se compone del número del nodo, el criterio de bifurcación (Split), número de individuos en el grupo (n), devianza (deviance) y el valor predicho (yval). Y los nodos terminales se expresan con un “*” al final. Algunos ejemplos son:

- 1) root 3784 257.5761000 0.073467230: Este es el nodo raíz, que incluye todas las observaciones (3,784) y tiene una devianza de 257.58 y una proporción de observaciones de EstatusPrest igual a 7.35%.
- 2) capacahorro>=-28.5 3471 99.0025900 0.029386340: El primer nodo hijo se basa en una división de la variable 'capacahorro'. Si 'capacahorro' es mayor o igual a -28.5, las observaciones van a este nodo. Hay 3,471 observaciones en este nodo (313 menos que su nodo padre), con una devianza de 99 y una proporción de observaciones de EstatusPrest igual a 2.94%.
- 4) importe< 2262.5 3417 63.7635400 0.019022530: Este nodo representa una segunda división basada en la variable 'importe'. Si 'importe' es menor que 2262.5, las observaciones caen en este nodo. Hay 3,417 (54 menos que en el anterior) individuos en este nodo, con una devianza de 63.76 y una proporción de observaciones de EstatusPrest igual a 1.9%.

El árbol continúa dividiendo las observaciones en nodos hijos sucesivos basados en diferentes variables predictoras y valores de corte hasta que hay un nodo final (*) y en vez de crear un nodo hijo crea más bien un “nodo hermano”, es decir, un nodo que surge del “nodo padre” del que acaba de terminarse, y sigue el mismo patrón.

Este árbol de decisión es un modelo de clasificación que utiliza las variables predictoras para predecir la variable de salida EstatusPrest (clase 0 o 1). El árbol se ajusta a los datos y divide las observaciones en función de los valores de las variables predictoras para clasificarlas en diferentes categorías de EstatusPrest.

Ahora convertimos el árbol en uno de clasificación y no regresivo, y para eso convertimos la variable EstatusPrest en tipo factor. Nos vuelve a salir otro árbol, pero ahora además de la información que nos daba antes, nos deja de dar la devianza, nos da loss (número de individuos mal clasificados, y yprob pasa a ser una especie de vector con las probabilidades de pertenecer a cada clase. Para que se entienda mejor, las tres primeras líneas son así:

- 1) root 3845 281 bueno (0.926918075 0.073081925)
- 2) capacahorro>=-10.74 3474 89 bueno (0.974381117 0.025618883)
- 4) importe< 2262.5 3420 52 bueno (0.984795322 0.015204678)

El nodo 1) incluye todos los individuos y la probabilidad original de pertenecer a cada clase, luego se ramifica con capacidad de ahorro mayor o igual a 10.74, los individuos disminuyen y la probabilidad de pertenecer a la primera clase crece, y lo mismo ocurre en el siguiente nodo que surge con importe menor que 226.2 y consigue ya un 98.47% de probabilidad de pertenecer al grupo de “bueno”.

En el gráfico del árbol se ven claramente las particiones. Los nodos finales con mayor probabilidad de pertenencia a la clase “bueno” son:

- 10) capacahorro ≥ 155.0825 13 o bueno (1.00 0.00) *
- 14) activos ≥ 25650 8 o bueno (1.00 0.00) *

Los podemos ver rodeados en azul indicando n/loss, es decir, número de individuos en el grupo/individuos mal clasificados.

A continuación podemos obtener la tabla de reducción de error, que nos indica lo siguiente:

-CP: parámetro de complejidad que mide cuánto se penaliza overfitting del modelo. Es ese valor va disminuyendo desde 0.23 a 0.01 (eje x inferior en el gráfico).

-nsplit: Número de "splits" en el árbol. En este caso fueron 14 divisiones (eje x superior en el gráfico).

-rel error: error en el conjunto de entrenamiento después de realizar una división. Este valor parte de 1 y a medida que se divide el árbol baja hasta 0.28.

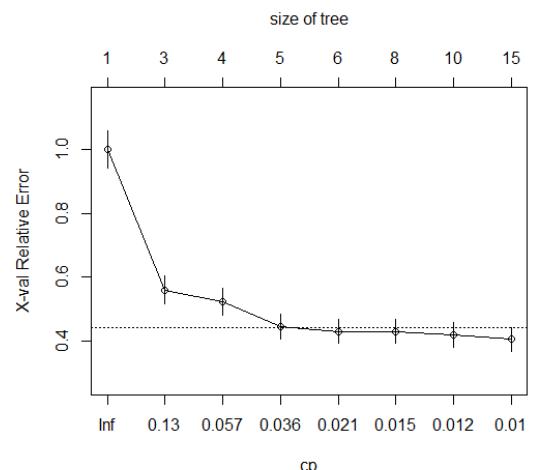
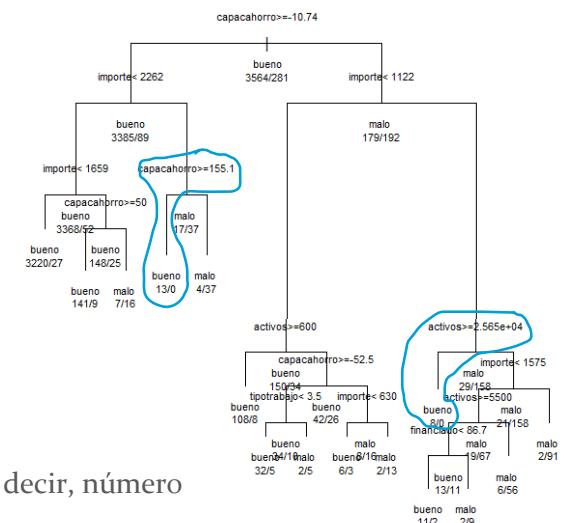
-xerror: Este es el error cruzado (cross-validation error) del modelo. Representa el error en un conjunto de datos de validación. Este también parte de 1 y acaba disminuyendo exponencialmente a 0.41 (eje y en el gráfico).

-xstd: La desviación estándar del error cruzado. También disminuye, pero más tímidamente, de 0.06 a 0.04 aproximadamente.

Para acabar, obtenemos la matriz de confusión, la proporción de verdaderos positivos, y la acuracidad:

	bueno	malo
bueno	3539	25
malo	54	227

ÁRBOL DE DECISIÓN



Este clasificador podría ser preferible al anterior ya que el error de clasificar uno malo como bueno ha disminuido algo, de 67 a 54. Esto también se refleja en la proporción de clasificados correctamente como malos, que pasa de 0.07308192 a 0.8078292.

bueno	malo
0.9929854	0.8078292

La acuracidad global se resume en 97.95%.

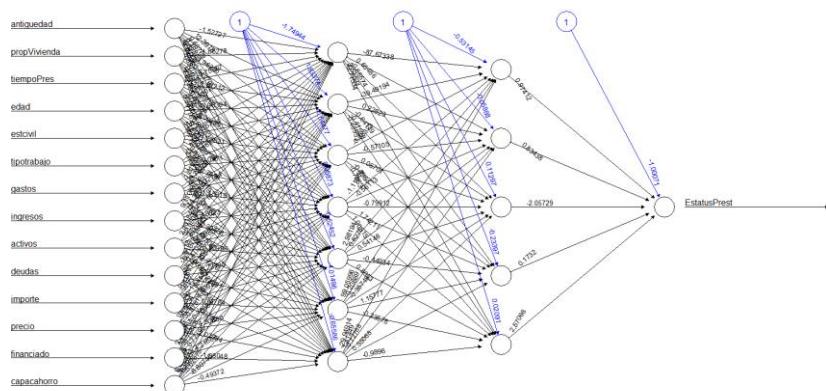
Ahora, comparando estos resultados con los clasificadores anteriores, la Regresión Logística tuvo una precisión del 99.01% y LDA del 97.71%. Comparando esto con la del RPart, 97.95%, Logística parece tener un rendimiento ligeramente superior, y LDA muy ligeramente inferior. Además, como hemos visto, el LDA evita con peor eficiencia los errores de clasificación de “malo” a “bueno”, el cual nos interesa minimizar. Dicho esto, ante Logística, Rpart queda en la sombra, pero la precisión entre LDA y RPart es tan similar que aún siendo algo más baja la segunda seguimos prefiriéndola ya que minimiza el error mencionado.

ANÁLISIS DEL CLASIFICADOR DE RED NEURONAL EN RSTUDIO

Para el preprocesamiento de los datos, algunas variables se han tenido que convertir a numéricas y se han sustituído los NA por las medianas de sus variables para evitar futuros problemas en el modelo.

Después, normalizamos las variables y dividimos los datos en el conjunto de entrenamiento ('train'), que se compone del 70% de los individuos, y de prueba ('test'), compuesta por el resto de individuos.

Ya definidos los datos de entrenamiento y de prueba, creamos una fórmula para simplificar el trabajo y no tener que escribir siempre el nombre de las variables. Estudiamos su relación con un modelo de red neuronal de algoritmo de propagación mejorada y con, en este caso, dos capas ocultas de 7 y 5 neuronas donde las iteraciones se detendrán cuando el cambio del error sea menor a 5% entre una iteracion y otra. Ofrecer el `summary()` del modelo le cuesta mucho, va muy lento con las pocas observaciones que hay en comparación con las que podría haber. El árbol del modelo se ve de la siguiente forma:



Tras crear el modelo, se realiza la predicción, el output son unos valores que representan los valores de salida del modelo para observaciones del conjunto de prueba. Cada número indica la probabilidad de que la observación pertenezca a la clase "bueno" según la red neuronal. Por ejemplo, en la primera fila, obtenemos esta salida:

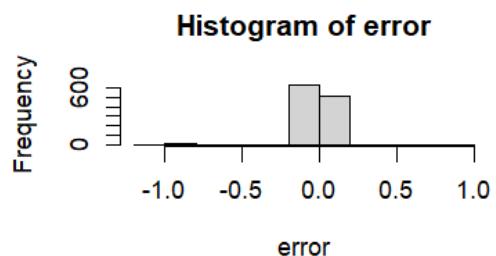
8 9.823324e-01

Así pues, la probabilidad de ser "bueno" es aproximadamente 0.98 (98.23%). Y así sucesivamente.

Ya realizada las predicciones, las desnormalizamos junto a los datos reales aplicando una escala a los datos. De esta forma podemos comparar los datos reales y los datos predichos en su forma original y obtener así el error a partir de la diferencia. Las características más importantes del resumen del error son las siguientes:

Min	Mean	Max
-1.016936	-0.002478	0.996167

Que el mínimo sea aproximadamente -1 significa que ha habido al menos una predicción hasta un punto subestimada. La media del error es aprox -0.002, lo que indica que de media las predicciones han sido más bajas 0.002 puntos, se tiende a subestimar. El máximo es de casi 1 punto también, así que prácticamente se han sobreestimado los valores la misma cantidad que se han subestimado, con una diferencia muy leve (0.002 puntos como vemos en la media).



Para analizar la clasificación primero creamos una función que clasifique como 0 y 1 a los individuos según si se acercan más a estos valores sus predicciones. Al aplicar la función nos encontramos con una lista de 0 y 1 que indican el grupo al que pertenece cada individuo. Si sumamos cuántos 0 hay, o cuántos individuos van al grupo "bueno", nos dan 1069, y si calculamos los del otro grupo nos dan 85 individuos. Para ver qué cantidad fue mal clasificada y cuál no, podemos analizar la matriz de confusión:

	bueno	malo
bueno	1069	6
malo	6	79

Este clasificador reduce considerablemente los dos errores, pero habría que estudiar la acuracidad ya que los individuos son técnicamente menos al no usar una base de datos binarizada.

bueno	malo
0.9943873	0.9294118

En la clasificación de los buenos tiene una eficiencia similar a los anteriores pero en la clasificación de los malos funciona bastante mejor, pasa de 0.8078292 a 0.9294118.

La acuracidad global se resume en 98.96%.

Comparando con el resto de clasificadores interesantes, Logística obtuvo una acuracidad del 99.01%, LDA del 97.71% y RPart del 97.95%, queda claro que a pesar de que Logística siga teniendo la mejor eficiencia como clasificador, la red neuronal queda como segunda en el ranking, ya que no solo aporta una acuracidad bastante sólida sino que además, nos aporta otras métricas como puede ser el error y tiene una eficiencia alta en la precisión del grupo “malo” que nos interesa tanto. Pero tenemos que tener en cuenta que tuvo un rendimiento más bajo en cuanto a rapidez de la creación del modelo.

CONCLUSIONES Y RECOMENDACIONES

En conclusión, el clasificador que escogeríamos para este estudio concreto sería el de Regresión Logística. Estas son las razones principales:

1. Interpretabilidad: La regresión logística es un modelo altamente interpretable. Puedes entender y comunicar fácilmente cómo cada variable afecta la probabilidad de pertenecer a un grupo u otro.
2. Efectivo para problemas de clasificación binaria: Es una excelente opción cuando estás trabajando en problemas que admiten o exigen clasificación binaria, como es este el caso.
3. No requiere asunciones sobre la distribución de los datos: A diferencia de otros como Naïve Bayes, que asumen una distribución particular para los datos, la regresión logística no requiere tales presupuestos.
4. Eficiente y rápido: La regresión logística es altamente y significativamente eficiente como hemos visto a lo largo del problema y no deja de funcionar bien con grandes conjuntos de datos.
5. Disponibilidad de diferentes métricas: A diferencia de otros clasificadores con los que es difícil obtener muchas métricas y rápido, con logística y usando el Experimenter de WEKA es muy sencillo.
6. Gran reducción del segundo error: es el mejor clasificador al momento de evitar errores de clasificación de “malo” a “bueno”.

De todas formas no tiene por qué ser perfecto, todos tienen algunas desventajas, algunas son las siguientes:

1. Asunción de linealidad: A pesar de no presuponer otras formas, sí asume una relación lineal entre las variables independientes y la probabilidad del resultado. No es adecuada para problemas altamente no lineales o complejos.

2. Puede ser sensible a valores atípicos: Valores atípicos en los datos pueden afectar negativamente el rendimiento del modelo y conllevaría un preproceso aún más largo.
3. No maneja bien variables altamente correlacionadas: La alta correlación entre variables puede distorsionar los resultados, pero en la realidad el orden de las probabilidades no dejaría de ser el mismo y se vería poco afectado.
5. Necesita una cantidad suficiente de datos: Para evitar el sobreajuste, se necesita una cantidad suficiente de datos para entrenar el modelo de manera efectiva, lo cual puede ser muy costoso en la vida real, pero no tan relevante en nuestro caso ya que con la historia de datos de la empresa durante unos años sería suficiente.

En resumen, la regresión logística es un modelo útil y eficaz, especialmente en problemas de clasificación binaria con datos que siguen una relación lineal. Sin embargo, en situaciones donde la relación entre las variables y el resultado es altamente no lineal, otros modelos pueden ser más apropiados.

Si se presentara como gran problema alguno de estos en el futuro, el otro clasificador que recomendaríamos es la red neuronal, ya que puede enfrentarse mejor a datos complejos y no lineales. Además, puede aprender automáticamente a partir de los datos, pero por eso mismo también requieren una cantidad significativa de datos y tiempo de entrenamiento, por lo que serían una elección sólida en situaciones donde la complejidad del problema justifica estos recursos.

BIBLIOGRAFÍA

- Raj, A. (2021, December 16). Perfect recipe for classification using logistic regression. *Medium*. <https://towardsdatascience.com/the-perfect-recipe-for-classification-using-logistic-regression-f8648e267592>
- Leonardosrocco, V. a. P. B. (2012, June 19). Árboles de clasificación, algoritmo J48. Data, Mining, BI & Social Stuff. <https://leonardosrocco.wordpress.com/2012/06/18/arboles-de-clasificacion-el-caso-del-algoritmo-j48-un-metodo-inexplorado-de-clasificacion-prediccion-y-generador-de-hipotesis-en-el-campo-de-la-sociologia/>
- Gonzalez, L. (2022). Regresión logística – teoría.  Aprende IA. <https://aprendeia.com/algoritmo-regresion-logistica-machine-learning-teoria/>
- Weka 3 - Data Mining with Open Source Machine Learning Software in Java. (n.d.). https://www.cs.waikato.ac.nz/~ml/weka/gui_explorer.html
- One hot encoding / Interactive Chaos. (n.d.). <https://interactivechaos.com/es/manual/tutorial-de-machine-learning/one-hot-encoding>