Practica 1. Tareas previas.

En esta primera práctica vamos a trabajar con la base de datos creditscore.

Nos plantearemos por lo tanto una tarea de clasificación: A partir de la información de los clientes de un banco tratar de clasificarlos entre los que tendrán un estatus bueno en el préstamo (lo devolverán adecuadamente) , y los que tendrán un estatus malo ( no lo devolverán adecuadamente) . En definitiva, vamos en busca del mejor **clasificador** posible.

Está disponible desde la web en formato sav y xlsx. Con carácter general puede resultar preferible manejar la versión sav cuando se trabaje en R y la versión de Excel cuando se trabaje en weka.

A lo largo de los trabajos a realizar durante la práctica es muy posible que **interese** grabar la base de datos tras alguna(s) modificación(es) en formato arff (para weka).

También puede ser útil contar con versiones modificadas de los datos (tras aplicar filtros, por ejemplo). Puede ser útil para distintos propósitos (datos de test externos, o datos de entrenamiento preferibles en algunos casos)

Realizaremos tareas de clasificación con distintas **herramientas** informáticas (sobre R y sobre weka) y distintas **técnicas** (usando diferentes clasificadores, quizás con varias opciones).

La base de datos cuenta con 15 variables (atributos), y más 3845 clientes.

Mientras en el formato sav las variables **Propiedad de la vivienda**, **estado civil**, **tipo de trabajo** y (la propia clase) **Estatus del préstamo** son variables nominales en la versión de Excel han perdido ese carácter y al procesarlas con weka requerirán ser modificadas utilizando los filtros que permitan convertir en nominales esas variables y asignarles las etiquetas adecuadas (NumericToNominal y RenameNominalValues)

1. NumericToNominal puede hacerse sólo una vez para todos los atributos

(el código que puede editarse y copiarse sería: weka.filters.unsupervised.attribute.NumericToNominal -R 2,5,6,15 )

1. RenameNominalValues (habrá que hacerlo una vez para cada atributo)

weka.filters.unsupervised.attribute.RenameNominalValues

weka.filters.unsupervised.attribute.RenameNominalValues -R 2 -N 1:propiedad,2:alquiler,3:otros

weka.filters.unsupervised.attribute.RenameNominalValues -R 5 -N 0:casado,1:viudo,2:soltero,3:separado,4:divorciado

weka.filters.unsupervised.attribute.RenameNominalValues -R 6 -N 1:fijo,2:temporal,3:freelance,4:otros

weka.filters.unsupervised.attribute.RenameNominalValues -R 15 -N 0:bueno,1:malo

**Una vez hecho esto valorar guardar el archivo ya adecuadamente modificado en formato arff.**

Las razones para guardarlo con este formato tras estas modificaciones son contundentes ¿estás de acuerdo?

Tras esto: [ Se llama la atención al estudiante de que los números que siguen no son indicadores de “cuestiones a realizar y tienen mero carácter enumerativo/organizativo]

1. Estudiaremos para su posterior aplicación a la práctica: Usamos un clasificador cualquiera (pero no trivial, no usaremos ni ZeroR ni OneRl) para ver cuatro cuestiones previas importantes. [logística, p.ej.]
2. Las opciones para la evaluación del clasificador: mostrar predicciones y otros atributos: Se puede pedir que salgan los resultados de clasificación de las instancias de la base de datos en distintos formatos
3. evaluación sensible al coste. Para comprobar su utilidad probaremos dos clasificadores distintos [logística y naïve Bayes,p.ej.]

Si se marca aparece una matriz que asigna en términos proporcionales los distintos costes (Obviamente el coste de acertar [ en la diagonal ] es habitualmente 0) Probar la opción que aparece en el punto 4

1. cambiar los indicadores de calidad de la evaluación.
2. cambiar la semilla aleatoria (si hay selección aleatoria en la evaluación como ocurre en cross validation y en split)



Además de comprender estas posibilidades se espera del alumno que sea capaz de llevar a cabo distintas modificaciones y anotarlas de cara a los comentarios finales futuros de la práctica.

1. Usar el clasificador zeroR. Da unos resultados de acuracidad global fantásticos ¿por qué?: la inmensa mayoría de los clientes pagan su préstamo.

¿qué cosas indica esto respecto a la información de la basa de datos? ¿qué utilidad puede tener, de cara a la resolución final de la tarea de clasificación?. Plantearse estas y otras cuestiones análogas, de cara a contar con el elementos apropiados para los comentarios al final de la resolución de la tarea de clasificación.

1. Usar el clasificador OneR da excelente acierto global. La variable discriminante seleccionada es la capacidad de ahorro. Eso es muy razonable. Pero si nos fijamos en las reglas propuestas para la clasificación el asunto deja de ser tan razonable.

Entonces, y pensando de nuevo, en contar con elementos de valoración de los futuros comentarios finales de la tarea ¿ hasta qué punto es relevante que la variable “capacidad de ahorro” sea **por sí sola** la más discriminante de todas? ¿Utilidad? ¿sentido?

1. Si hacemos una clasificación sensible al coste con mayores costes para los errores de falsos positivos en la clase “bueno” (3 contra 1, por ejemplo) , ¿cómo clasifican zeroR y oneR respecto al coste en relación los anteriormente utilizados?De nuevo es interesante consideran estos resultados de cara al futuro análisis no tanto por su relevancia clasificatoria como por lo que revelan sobre la naturaleza de los datos.
2. Trabajaremos, para empezar, algo más seriamente, con algunos clasificadores **funcionales,** basados en técnicas de filiación estadística.

**Regresión logística** y análisis lineal discriminante. Analizaremos los resultados y los guardaremos para su posterior análisis. Esto lo haremos también para otros clasificadores.

El análisis discriminante necesita instalar un paquete adicional

Una vez hecho esto:

¿Por qué, en principio no puede usarse el lda (análisis discriminante lineal)?

Necesita que todas las variables de entrada (independientes) sean **númericas**

¿Qué opciones tenemos?

* Prescindir de atributos nominales. Volver a los datos numéricos (archivo de Excel). Binarizar.
* Binarizar : consiste en convertir una variable cualitativa en tantas variables binarias ( 0,1) como niveles del factor haya.

Para binarizar puede usarse el filtro:

weka.filters.supervised.attribute.NominalToBinary

**Ojo: al binarizar se modifica la base de datos: si usamos los datos binarizados para repetir una clasificación ya hecha con los datos originales , la clasificación puede dar resultados diferentes ( con Naïve Bayes, ocurre, con la logística , en cambio, apenas se modifica, por razones de construcción de la propia regresión)**

Con todo, parece que las mejores opciones no funcionan. La razón está en el elevado número de instancias con valores perdidos que existen en la base de datos. Una solución es sustituir los valores perdidos por la media/moda. Con el filtro

weka.filters.unsupervised.attribute.ReplaceMissingValues

Tras la aplicación de este filtro ya va a ser posible aplicar tanto el análisis lineal discriminantes como la logística y el Naïve Bayes . Pero para poder comparar unos y otros resultado, obviamente todos los aplicaremos al mismo conjunto de datos.

Llegados aquí aprenderemos a utilizar el experimenter de weka ,compararemos estos resultados con los de utilizar NaïveBayes, también probaremos con algunos clasificadores árbol y basados en reglas , y nos plantearemos si es razonable o no ensayar una red MLP . Pero esto será en próximas sesiones.

Nos va interesar para todo ello guardar, **al menos** los datos tras la “nominalización” y tras la “binarización+sustitución de missings”.

Cuestión para ir pensando: Se ha comentado que el experimenter nos permitirá comparar el funcionamiento de unos y otros clasificadores pero de hecho eso ya lo podemos hacer con el explorer, algo más debe aportar el experimenter en su forma de hacerlo o alguna ventaja debe darnos o algún posible inconveniente en la comparación directa con explorer debe ser capaz de corregir.( Os avanzo que es esto último lo más relevante,aunque también ofrezca alguna ventaja )