# MLDR

## Paquete R para exploración multietiqueta

David Charte Francisco Charte

11 nov 2015 – TAMIDA (Retos) – CAEPIA '15

Soft Computing and Intelligent Information Systems – Universidad de Granada

# ÍNDICE

# INTRODUCCIÓN

# CLASIFICACIÓN DE DATOS

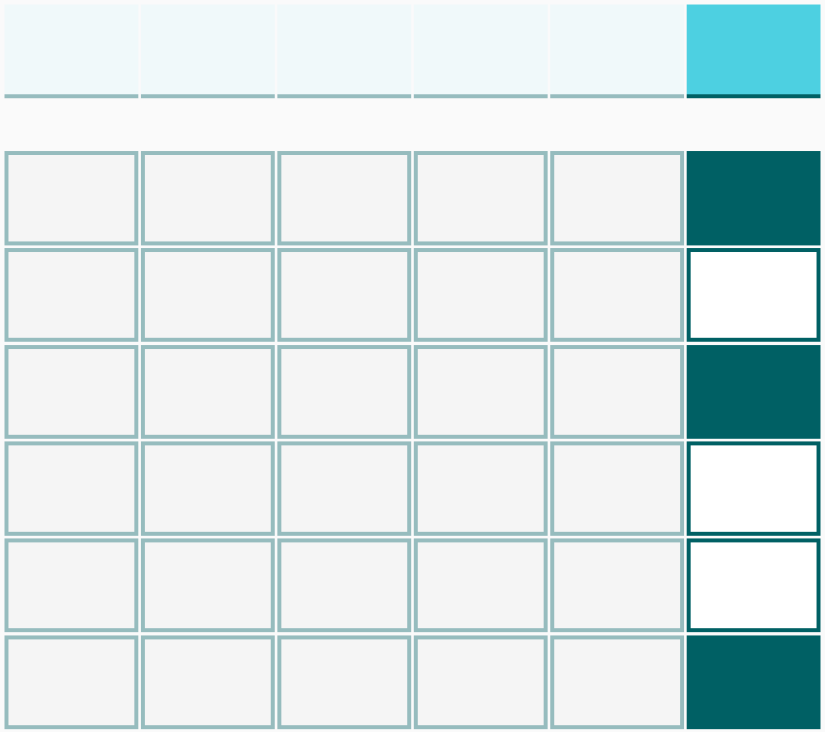
Aplicaciones:

- Detección de spam
- Diagnóstico de enfermedades
- Detección de fraude
- Predicción de riesgos
- ...

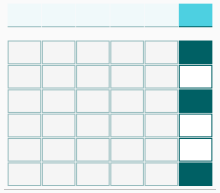

La clasificación de datos es una tarea en la que se aprende de datos clasificados para tratar de predecir cierta información, la información de clase, de nuevos datos. Las técnicas para tratar este tipo de problemas se utilizan en todo tipo de situaciones, algunos ejemplos típicos son la detección del spam en el correo electrónico, el análisis de síntomas de un paciente para asesorar en su diagnóstico médico, o la detección de anomalías en distintos ámbitos, en particular para detectar fraudes.

# CLASIFICACIÓN TRADICIONAL

## Clasificación binaria



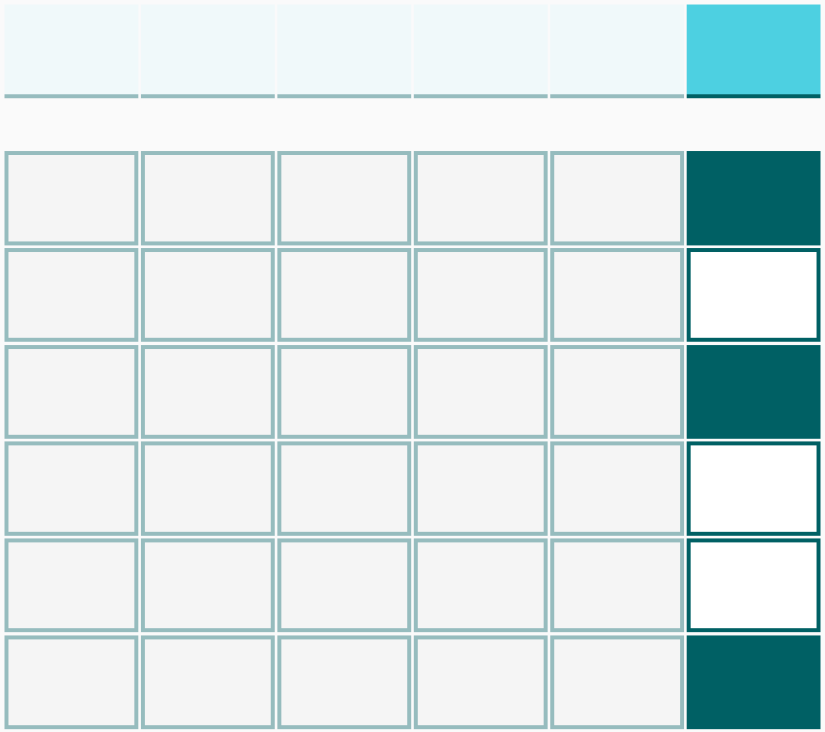
Clasificación binaria



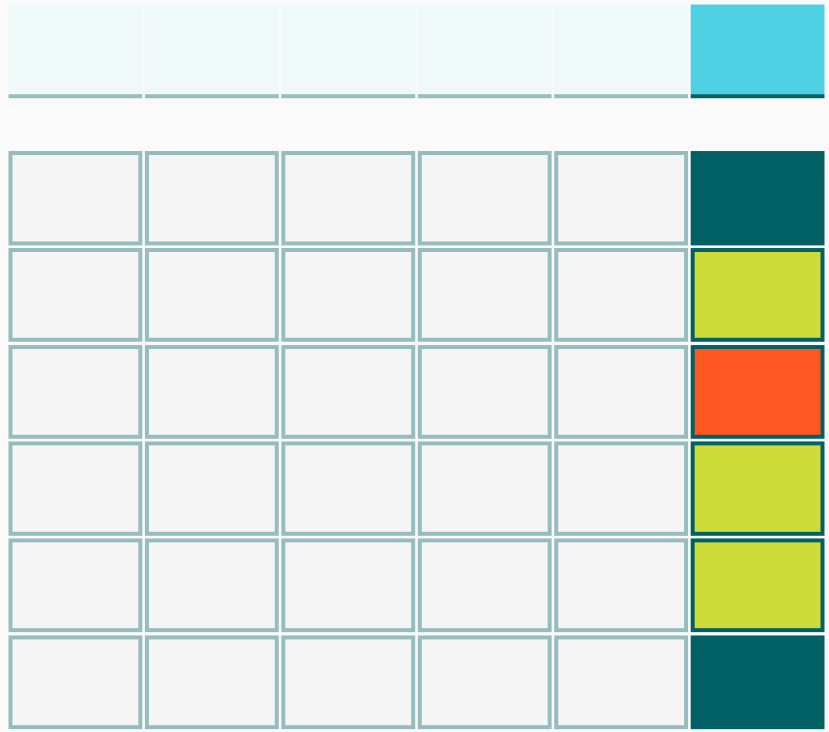
Esta tabla representa un conjunto de datos en el que la información de clase es de tipo binario. Si cada fila representa una instancia, con sus valores para cada atributo, basta con añadir una última columna con valores de cero o uno para representar la información de clase de cada instancia. De esta forma, para cada nueva instancia, hay que predecir una de dos posibles opciones. Si pasamos a un conjunto de datos multiclase, tendremos más de dos clases posibles para cada instancia, y cada instancia pertenecerá a una sola de ellas. De esa forma se puede seguir representando la clase en una sola columna que acepte varios valores. La información que habrá que predecir para una instancia de test será una clase de entre el número de clases disponibles.

# CLASIFICACIÓN TRADICIONAL

## Clasificación binaria



## Clasificación multiclase



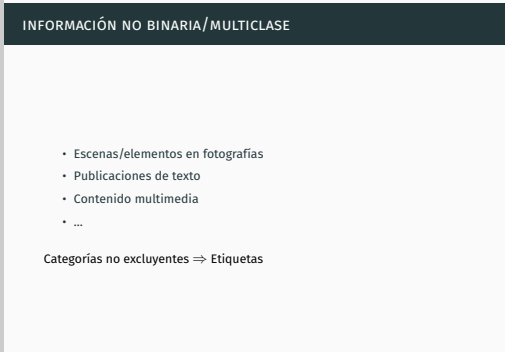

Esta tabla representa un conjunto de datos en el que la información de clase es de tipo binario. Si cada fila representa una instancia, con sus valores para cada atributo, basta con añadir una última columna con valores de cero o uno para representar la información de clase de cada instancia. De esta forma, para cada nueva instancia, hay que predecir una de dos posibles opciones. Si pasamos a un conjunto de datos multiclase, tendremos más de dos clases posibles para cada instancia, y cada instancia pertenecerá a una sola de ellas. De esa forma se puede seguir representando la clase en una sola columna que acepte varios valores. La información que habrá que predecir para una instancia de test será una clase de entre el número de clases disponibles.

# CLASIFICACIÓN MULTIETIQUETA

# INFORMACIÓN NO BINARIA/MULTICLASE

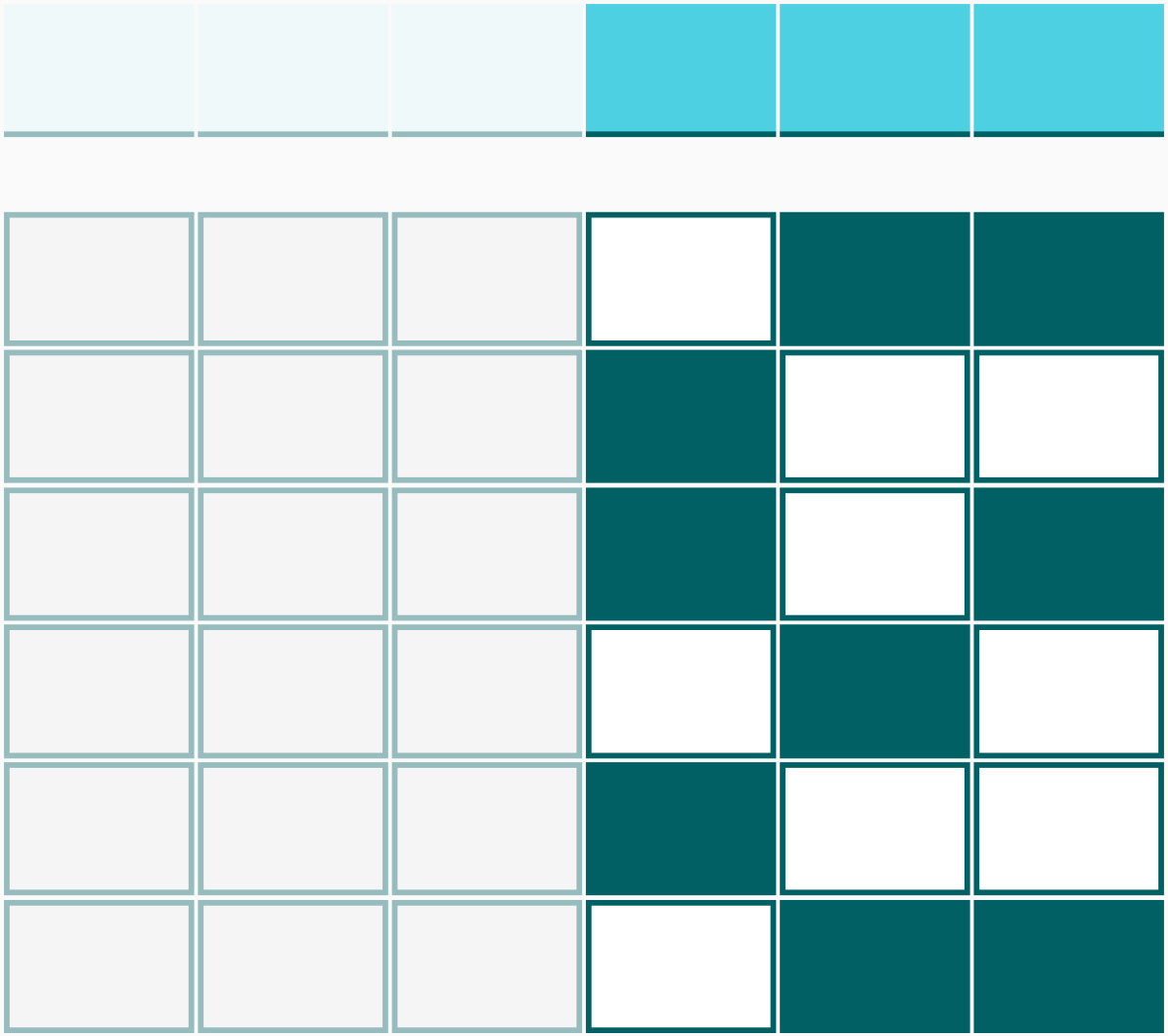
- Escenas/elementos en fotografías
- Publicaciones de texto
- Contenido multimedia
- ...

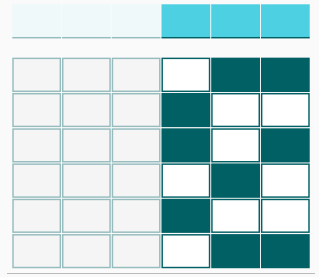
Categorías no excluyentes $\Rightarrow$ Etiquetas




En ocasiones los problemas con los que nos topamos no cumplen las restricciones de estos tipos de clasificación y necesitamos una generalización de ellos, en este caso la clasificación multietiqueta. Algunos ejemplos de esas situaciones pueden ser la identificación de elementos en contenido multimedia, tanto audio como fotografías y vídeo. Por ejemplo, si tenemos un conjunto de fotografías en las que consideramos que pueden aparecer la playa, la montaña, la puesta de sol y personas, en una sola fotografía podrían aparecer varios de esos elementos. Pasa lo mismo cuando analizamos textos, el contenido de uno podría ser político, económico y de opinión a la vez. Cuando tenemos estas categorías no excluyentes, las llamamos etiquetas.

# CLASIFICACIÓN MULTIETIQUETA





Y como no son excluyentes, las representaciones anteriores no nos sirven, necesitamos más de una columna para almacenar toda la información de clase, en concreto una columna por etiqueta. Cada columna, eso sí, sólo aceptará los valores 0 o 1.

# CLASIFICACIÓN MULTIETIQUETA

- Instancia: $(X, Y) \in X^1 \times X^2 \times \cdots \times X^f \times \mathcal{P}(L)$
- Para cada instancia hay $2^{|L|}$ posibles predicciones

Clasificación multietiqueta



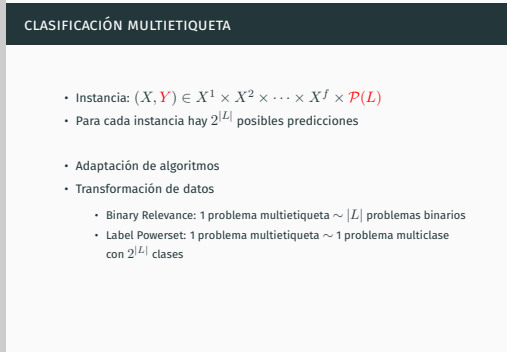
- Instancia: $(X, Y) \in X^1 \times X^2 \times \cdots \times X^I \times \mathcal{P}(L)$
- Para cada instancia hay $2^{|L|}$ posibles predicciones

Cada instancia de tipo multietiqueta se puede ver como un vector del espacio de atributos junto a un subconjunto de la familia de etiquetas que hay disponibles. En los casos anteriores cada instancia toma un único valor de clase, mientras que en este caso estamos tomando un subconjunto. Esto implica que al predecir la información de clase habrá que decidir entre un número de posibilidades de dos elevado al número de etiquetas. Para tratar estas situaciones hay que, o bien adaptar algoritmos existentes al nuevo problema, o bien transformar y separar los datos de forma que los convirtamos en problemas binarios o multietiqueta. En este último caso, el obstáculo que nos podemos encontrar es que se generan varios problemas binarios, o uno multiclase con muchas clases, por cada multietiqueta. Por último, al tener esta nueva situación en la que las etiquetas no son excluyentes, el estudio de cuándo aparecen juntas varias etiquetas puede ser interesante, por ejemplo analizando las interacciones entre etiquetas muy poco frecuentes y las más comunes. Esto implica que necesitamos observar los conjuntos de datos mediante una nueva serie de métricas que nos den más información

# CLASIFICACIÓN MULTIETIQUETA

- Instancia: $(X, Y) \in X^1 \times X^2 \times \cdots \times X^f \times \mathcal{P}(L)$
- Para cada instancia hay $2^{|L|}$ posibles predicciones

- Adaptación de algoritmos
- Transformación de datos
  - Binary Relevance: 1 problema multietiqueta $\sim |L|$ problemas binarios
  - Label Powerset: 1 problema multietiqueta $\sim$ 1 problema multiclase con $2^{|L|}$ clases




Cada instancia de tipo multietiqueta se puede ver como un vector del espacio de atributos junto a un subconjunto de la familia de etiquetas que hay disponibles. En los casos anteriores cada instancia toma un único valor de clase, mientras que en este caso estamos tomando un subconjunto. Esto implica que al predecir la información de clase habrá que decidir entre un número de posibilidades de dos elevado al número de etiquetas. Para tratar estas situaciones hay que, o bien adaptar algoritmos existentes al nuevo problema, o bien transformar y separar los datos de forma que los convirtamos en problemas binarios o multietiqueta. En este último caso, el obstáculo que nos podemos encontrar es que se generan varios problemas binarios, o uno multiclase con muchas clases, por cada multietiqueta. Por último, al tener esta nueva situación en la que las etiquetas no son excluyentes, el estudio de cuándo aparecen juntas varias etiquetas puede ser interesante, por ejemplo analizando las interacciones entre etiquetas muy poco frecuentes y las más comunes. Esto implica que necesitamos observar los conjuntos de datos mediante una nueva serie de métricas que nos den más información

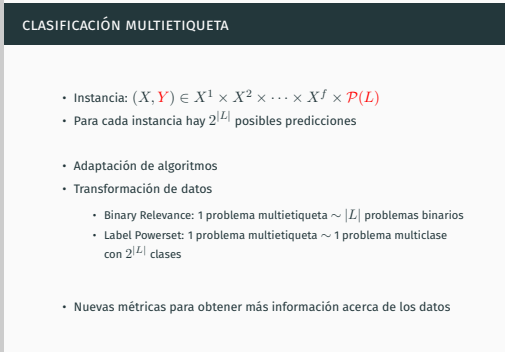
# CLASIFICACIÓN MULTIETIQUETA

- Instancia: $(X, Y) \in X^1 \times X^2 \times \cdots \times X^f \times \mathcal{P}(L)$
- Para cada instancia hay $2^{|L|}$ posibles predicciones

- Adaptación de algoritmos
- Transformación de datos
  - Binary Relevance: 1 problema multietiqueta $\sim |L|$ problemas binarios
  - Label Powerset: 1 problema multietiqueta $\sim$ 1 problema multiclase con $2^{|L|}$ clases

- Nuevas métricas para obtener más información acerca de los datos

└Clasificación multietiqueta




Cada instancia de tipo multietiqueta se puede ver como un vector del espacio de atributos junto a un subconjunto de la familia de etiquetas que hay disponibles. En los casos anteriores cada instancia toma un único valor de clase, mientras que en este caso estamos tomando un subconjunto. Esto implica que al predecir la información de clase habrá que decidir entre un número de posibilidades de dos elevado al número de etiquetas. Para tratar estas situaciones hay que, o bien adaptar algoritmos existentes al nuevo problema, o bien transformar y separar los datos de forma que los convirtamos en problemas binarios o multietiqueta. En este último caso, el obstáculo que nos podemos encontrar es que se generan varios problemas binarios, o uno multiclase con muchas clases, por cada multietiqueta. Por último, al tener esta nueva situación en la que las etiquetas no son excluyentes, el estudio de cuándo aparecen juntas varias etiquetas puede ser interesante, por ejemplo analizando las interacciones entre etiquetas muy poco frecuentes y las más comunes. Esto implica que necesitamos observar los conjuntos de datos mediante una nueva serie de métricas que nos den más información

# EL PAQUETE MLDR

# MOTIVACIÓN

- Necesidad de una herramienta accesible para exploración de datos multietiqueta
- Potencial de R para manejo de datos: estructuras de datos, instrucciones vectorizadas...
- Paquetes de gráficos disponibles para R
- Facilidad de interacción desde la consola interactiva de R

MOTIVACIÓN

- Necesidad de una herramienta accesible para exploración de datos multietiqueta
- Potencial de R para manejo de datos: estructuras de datos, instrucciones vectorizadas...
- Paquetes de gráficos disponibles para R
- Facilidad de interacción desde la consola interactiva de R

mldr es un software para análisis exploratorio que nace de la necesidad de tener una herramienta que agrupe estas métricas específicas para clasificación multietiqueta y las proporcione al usuario de una forma sencilla. Además, elegir R como la plataforma para el desarrollo del paquete vino motivado por las facilidades que aporta para el tratamiento de datos, como las instrucciones vectorizadas o estructuras de datos ya incluídas como el `data.frame`. También son interesantes las funciones disponibles para componer gráficos y otros paquetes que amplían esta funcionalidad. Además, la interacción con el paquete se puede hacer mediante la consola interactiva de R, usando las funciones que se proporcionan, o bien desde una interfaz web que viene incorporada.

# INSTALACIÓN Y CARGA

Disponible en CRAN

```
install.packages("mldr")
library(mldr)
```

INSTALACIÓN Y CARGA

Disponible en CRAN

```
install.packages("mldr")
library(mldr)
```

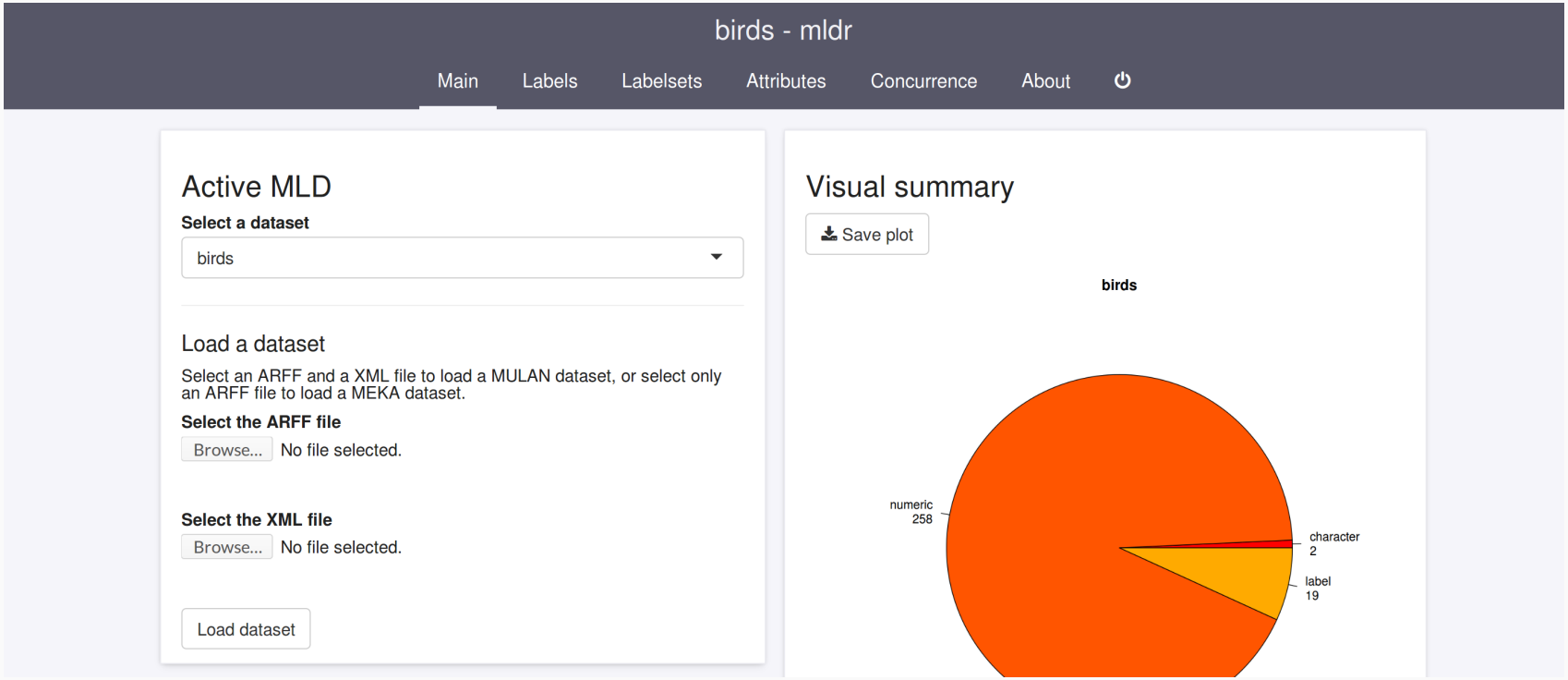
La instalación del paquete es muy simple, basta con usar el comando install.packages y R se encargará de descargar desde CRAN el paquete y sus dependencias e instalarlo todo. Una vez hecho esto, se carga el paquete con la función library. Si se quiere hacer uso de la interfaz gráfica de usuario habrá que llamar a la función mldrGUI, y se abrirá una pestaña de navegador que conectará con una aplicación web incluida en el paquete, desarrollada mediante el uso de otro paquete llamado shiny.

# INSTALACIÓN Y CARGA

## Disponible en CRAN

```
install.packages("mldr")
library(mldr)

mldrGUI()
```

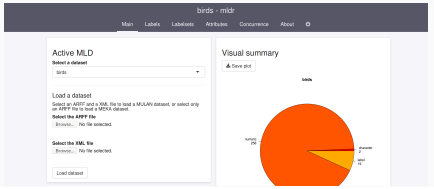


## INSTALACIÓN Y CARGA

Disponible en CRAN

```
install.packages("mldr")
library(mldr)

mldrGUI()
```




La instalación del paquete es muy simple, basta con usar el comando install.packages y R se encargará de descargar desde CRAN el paquete y sus dependencias e instalarlo todo. Una vez hecho esto, se carga el paquete con la función library. Si se quiere hacer uso de la interfaz gráfica de usuario habrá que llamar a la función mldrGUI, y se abrirá una pestaña de navegador que conectará con una aplicación web incluida en el paquete, desarrollada mediante el uso de otro paquete llamado shiny.

# LECTURA Y CREACIÓN DE DATASETS

- Datasets en formato ARFF de Mulan y MEKA:

```
emotions <- mldr("emotions")
enron <- mldr("ENRON-F", use_xml = FALSE)
```

LECTURA Y CREACIÓN DE DATASETS

- Datasets en formato ARFF de Mulan y MEKA:

```
emotions <- mldr("emotions")
enron <- mldr("ENRON-F", use_xml = FALSE)
```

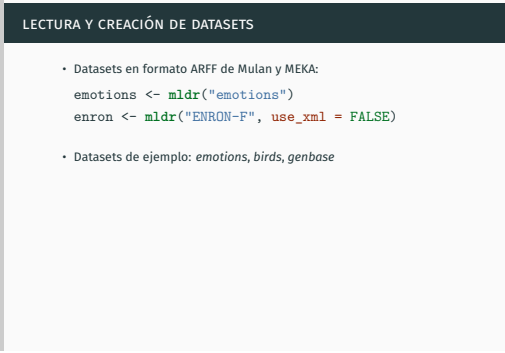
Lo que proporciona mldr, en una visión general, es una clase de objetos con una serie de funciones que se pueden llamar sobre esos objetos. Cada objeto representará un conjunto de datos multietiqueta, y generalmente estos datos vendrán de archivos en formato ARFF de tipo Mulan o MEKA, ambos soportados por mldr. mldr incluye ya 3 datasets de ejemplo, emotions, birds y genbase. Pero además, mldr permite crear nuevos datasets a partir de otras estructuras de datos que estén ya cargadas o se creen en R, simplemente indicándole cuáles de los atributos son de salida, es decir, etiquetas.

# LECTURA Y CREACIÓN DE DATASETS

- Datasets en formato ARFF de Mulan y MEKA:

```
emotions <- mldr("emotions")
enron <- mldr("ENRON-F", use_xml = FALSE)
```

- Datasets de ejemplo: *emotions*, *birds*, *genbase*




Lo que proporciona mldr, en una visión general, es una clase de objetos con una serie de funciones que se pueden llamar sobre esos objetos. Cada objeto representará un conjunto de datos multietiqueta, y generalmente estos datos vendrán de archivos en formato ARFF de tipo Mulan o MEKA, ambos soportados por mldr. mldr incluye ya 3 datasets de ejemplo, emotions, birds y genbase. Pero además, mldr permite crear nuevos datasets a partir de otras estructuras de datos que estén ya cargadas o se creen en R, simplemente indicándole cuáles de los atributos son de salida, es decir, etiquetas.

# LECTURA Y CREACIÓN DE DATASETS

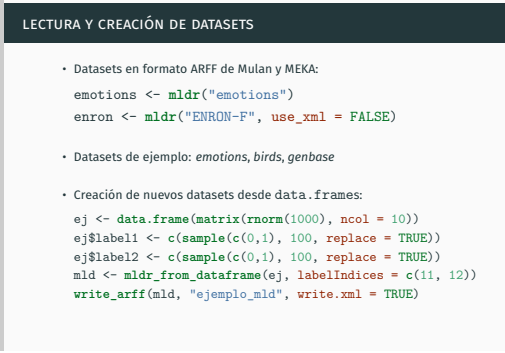
- Datasets en formato ARFF de Mulan y MEKA:

```
emotions <- mldr("emotions")
enron <- mldr("ENRON-F", use_xml = FALSE)
```

- Datasets de ejemplo: *emotions*, *birds*, *genbase*

- Creación de nuevos datasets desde `data.frames`:

```
ej <- data.frame(matrix(rnorm(1000), ncol = 10))
ej$label1 <- c(sample(c(0,1), 100, replace = TRUE))
ej$label2 <- c(sample(c(0,1), 100, replace = TRUE))
mld <- mldr_from_dataframe(ej, labelIndices = c(11, 12))
write_arff(mld, "ejemplo_mld", write.xml = TRUE)
```




Lo que proporciona mldr, en una visión general, es una clase de objetos con una serie de funciones que se pueden llamar sobre esos objetos. Cada objeto representará un conjunto de datos multietiqueta, y generalmente estos datos vendrán de archivos en formato ARFF de tipo Mulan o MEKA, ambos soportados por mldr. mldr incluye ya 3 datasets de ejemplo, emotions, birds y genbase. Pero además, mldr permite crear nuevos datasets a partir de otras estructuras de datos que estén ya cargadas o se creen en R, simplemente indicándole cuáles de los atributos son de salida, es decir, etiquetas.

# LECTURA Y CREACIÓN DE DATASETS

- Datasets en formato ARFF de Mulan y MEKA:

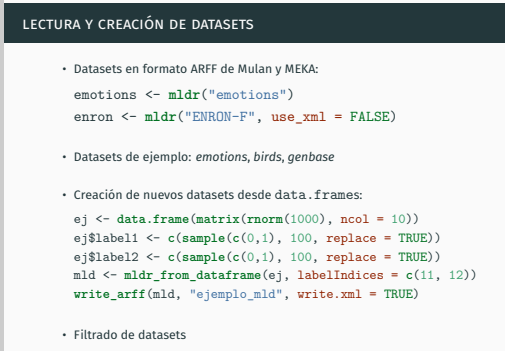
```
emotions <- mldr("emotions")
enron <- mldr("ENRON-F", use_xml = FALSE)
```

- Datasets de ejemplo: *emotions*, *birds*, *genbase*

- Creación de nuevos datasets desde `data.frames`:

```
ej <- data.frame(matrix(rnorm(1000), ncol = 10))
ej$label1 <- c(sample(c(0,1), 100, replace = TRUE))
ej$label2 <- c(sample(c(0,1), 100, replace = TRUE))
mld <- mldr_from_dataframe(ej, labelIndices = c(11, 12))
write_arff(mld, "ejemplo_mld", write.xml = TRUE)
```

- Filtrado de datasets




Lo que proporciona mldr, en una visión general, es una clase de objetos con una serie de funciones que se pueden llamar sobre esos objetos. Cada objeto representará un conjunto de datos multietiqueta, y generalmente estos datos vendrán de archivos en formato ARFF de tipo Mulan o MEKA, ambos soportados por mldr. mldr incluye ya 3 datasets de ejemplo, emotions, birds y genbase. Pero además, mldr permite crear nuevos datasets a partir de otras estructuras de datos que estén ya cargadas o se creen en R, simplemente indicándole cuáles de los atributos son de salida, es decir, etiquetas.

# OBTENCIÓN DE MEDIDAS

```
summary(emotions)
```

```
num.attributes  num.instances  num.labels  num.labelsets
            78            593           6             27
num.single.labelsets  max.frequency  cardinality    density
                   4             81     1.868465  0.3114109
  meanIR     scumble
1.478068  0.01095238
```

OBTENCIÓN DE MEDIDAS

```
summary(emotions)
```

```
num.attributes  num.instances  num.labels  num.labelsets
            78            593           6             27
num.single.labelsets  max.frequency  cardinality    density
                   4             81     1.868465  0.3114109
  meanIR     scumble
1.478068  0.01095238
```

Una vez que tenemos un objeto mldr, muchas medidas se pueden obtener directamente con la función summary, o accediendo a los miembros del objeto que aportan más datos, como labels que incluye cálculos sobre etiquetas.

# OBTENCIÓN DE MEDIDAS

```
summary(emotions)
```

```
num.attributes  num.instances  num.labels  num.labelsets
            78            593           6             27
num.single.labelsets  max.frequency  cardinality    density
                   4             81     1.868465  0.3114109
  meanIR     scumble
1.478068  0.01095238
```

```
emotions$labels
```

| | index | count | freq | IRLbl | SCUMBLE |
|---|---|---|---|---|---|
| amazed-suprised | 73 | 173 | 0.2917369 | 1.526012 | 0.002159173 |
| happy-pleased | 74 | 166 | 0.2799325 | 1.590361 | 0.014332319 |
| relaxing-calm | 75 | 264 | 0.4451939 | 1.000000 | 0.023786461 |
| quiet-still | 76 | 148 | 0.2495784 | 1.783784 | 0.023131538 |
| sad-lonely | 77 | 168 | 0.2833052 | 1.571429 | 0.016133470 |
| angry-aggresive | 78 | 189 | 0.3187184 | 1.396825 | 0.001331189 |

OBTENCIÓN DE MEDIDAS

```
summary(emotions)
```

```
num.attributes  num.instances  num.labels  num.labelsets
            78            593           6             27
num.single.labelsets  max.frequency  cardinality    density
                   4             81     1.868465  0.3114109
  meanIR     scumble
1.478068  0.01095238
```

```
emotions$labels
```

```
                index count      freq    IRLbl     SCUMBLE
amazed-suprised    73   173 0.2917369 1.526012 0.002159173
happy-pleased      74   166 0.2799325 1.590361 0.014332319
relaxing-calm      75   264 0.4451939 1.000000 0.023786461
quiet-still        76   148 0.2495784 1.783784 0.023131538
sad-lonely         77   168 0.2833052 1.571429 0.016133470
angry-aggresive    78   189 0.3187184 1.396825 0.001331189
```

Una vez que tenemos un objeto mldr, muchas medidas se pueden obtener directamente con la función summary, o accediendo a los miembros del objeto que aportan más datos, como labels que incluye cálculos sobre etiquetas.

# OBTENCIÓN DE MEDIDAS

emotions - mldr

Main | Labels | Labelsets | Attributes | Concurrence | About

Show 25 entries Copy Print Save

Search:

| Label | index | count | freq | IRLbl | SCUMBLE |
|---|---|---|---|---|---|
| amazed-suprised | 73 | 173 | 0.2917369 | 1.526012 | 0.002159173 |
| happy-pleased | 74 | 166 | 0.2799325 | 1.590361 | 0.014332319 |
| relaxing-calm | 75 | 264 | 0.4451939 | 1.000000 | 0.023786461 |
| quiet-still | 76 | 148 | 0.2495784 | 1.783784 | 0.023131538 |
| sad-lonely | 77 | 168 | 0.2833052 | 1.571429 | 0.016133470 |
| angry-aggresive | 78 | 189 | 0.3187184 | 1.396825 | 0.001331189 |
| Label | index | count | freq | IRLbl | SCUMBLE |

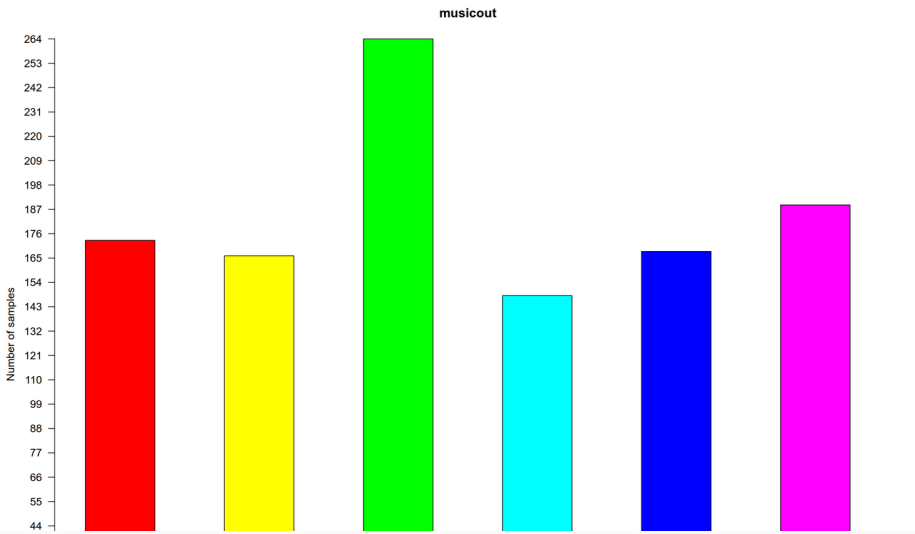
Showing 1 to 6 of 6 entries

Previous 1 Next

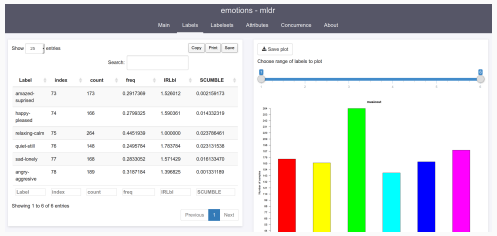
Save plot

Choose range of labels to plot

1 2 3 4 5 6



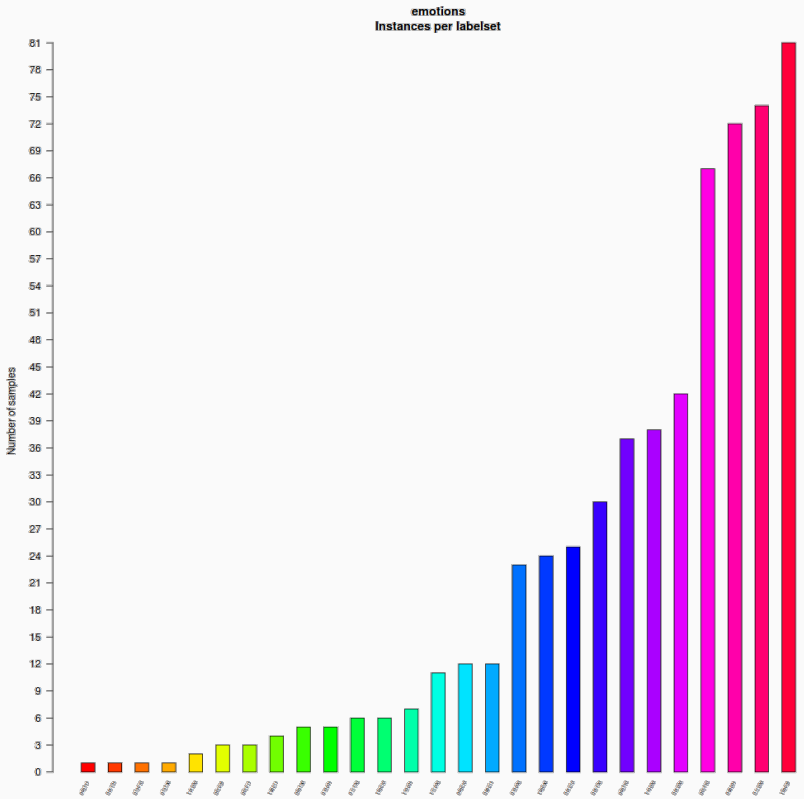
## OBTENCIÓN DE MEDIDAS



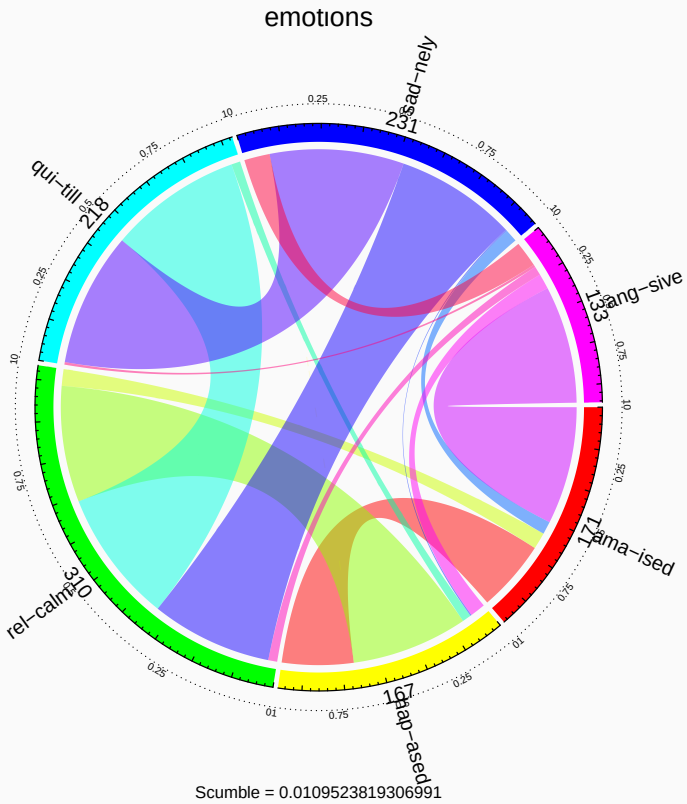

Todas las medidas que se obtienen directamente mediante funciones de mldr se muestran también en la interfaz se usuario, simplemente navegando por cada pestaña se obtendrán tablas y resúmenes de los datos referentes a atributos, etiquetas y combinaciones de etiquetas (labelsets).

# GENERACIÓN DE GRÁFICOS

```
plot(emotions, type = "LSB")
```

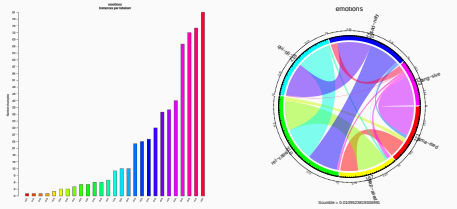



```
plot(emotions, type = "LC")
```



GENERACIÓN DE GRÁFICOS

```
plot(emotions, type = "LSB")  plot(emotions, type = "LC")
```



La función plot está implementada para los objetos de tipo mldr y permite generar 6 tipos de gráfico distintos, entre ellos uno muy interesante que es el gráfico de concurrencia entre etiquetas, a la derecha en la diapositiva, y que muestra cómo se distribuyen las interacciones entre etiquetas. El ejemplo de gráfico de la izquierda muestra las posibles combinaciones de etiquetas y cuántas veces se dan a lo largo de todo el conjunto de datos.

# CLASIFICACIÓN CON MLDR

## Informe de concurrencia: búsqueda de etiquetas difíciles

emotions - mldr

Main Labels Labelsets Attributes Concurrence About

### Concurrence analysis

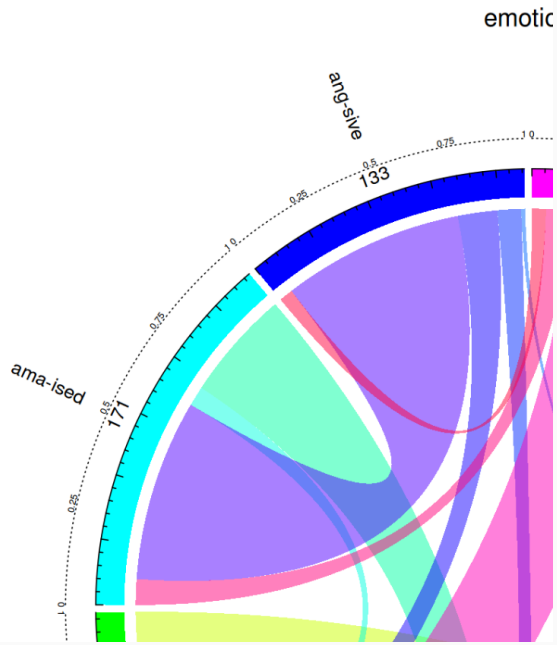
The SCUMBLE level for each label is shown in the table at the left.

In the following table the minority labels most affected by SCUMBLE are shown.

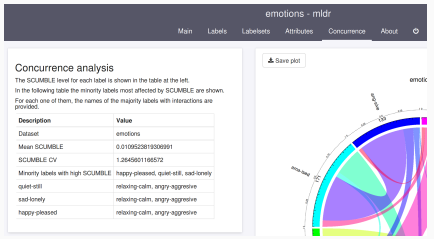
For each one of them, the names of the majority labels with interactions are provided.

| Description | Value |
|---|---|
| Dataset | emotions |
| Mean SCUMBLE | 0.0109523819306991 |
| SCUMBLE CV | 1.2645601166572 |
| Minority labels with high SCUMBLE | happy-pleased, quiet-still, sad-lonely |
| quiet-still | relaxing-calm, angry-aggresive |
| sad-lonely | relaxing-calm, angry-aggresive |
| happy-pleased | relaxing-calm, angry-aggresive |

Save plot



Informe de concurrencia: búsqueda de etiquetas difíciles




mldr puede ir más allá del análisis exploratorio y ayudar a la hora de realizar las tareas de clasificación. Por un lado, es capaz de generar un informe acerca de las etiquetas que podrían resultar difíciles de tratar mediante algoritmos de preprocesamiento, tanto en la interfaz web como en un archivo PDF.

# CLASIFICACIÓN CON MLDR

- Transformaciones Label Powerset y Binary Relevance
- 19 métricas de evaluación de resultados
- (*Pronto*) Interfaz común para implementación de clasificadores

```
mldr_evaluate(emotions, predictions)
```

```
List of 20
 $ Accuracy        : num 0.912
 $ AUC             : num 0.916
 $ AveragePrecision: num 0.669
 $ Coverage        : num 2.72
 $ FMeasure        : num 0.942
 $ HammingLoss     : num 0.0883
 $ MacroAUC        : num 0.919
 $ MacroFMeasure   : num 0.865
 $ MacroPrecision  : num 0.805
 $ MacroRecall     : num 0.936
 $ MicroAUC        : num 0.918
 $ MicroFMeasure   : num 0.868
 $ MicroPrecision  : num 0.811
 $ MicroRecall     : num 0.935
 $ OneError        : num 0.111
 $ Precision       : num 0.927
 $ RankingLoss     : num 0.508
 $ Recall          : num 0.927
 $ SubsetAccuracy  : num 0.831
 $ ROC             :List of 15
```

CLASIFICACIÓN CON MLDR

- Transformaciones Label Powerset y Binary Relevance
- 19 métricas de evaluación de resultados
- (*Pronto*) Interfaz común para implementación de clasificadores

```
mldr_evaluate(emotions, predictions)

List of 20
 $ Accuracy        : num 0.912
 $ AUC             : num 0.916
 $ AveragePrecision: num 0.669
 $ Coverage        : num 2.72
 $ FMeasure        : num 0.942
 $ HammingLoss     : num 0.0883
 $ MacroAUC        : num 0.919
 $ MacroFMeasure   : num 0.865
 $ MacroPrecision  : num 0.805
 $ MacroRecall     : num 0.936
 $ MicroAUC        : num 0.918
 $ MicroFMeasure   : num 0.868
 $ MicroPrecision  : num 0.811
 $ MicroRecall     : num 0.935
 $ OneError        : num 0.111
 $ Precision       : num 0.927
 $ RankingLoss     : num 0.508
 $ Recall          : num 0.927
 $ SubsetAccuracy  : num 0.831
 $ ROC             :List of 15
```

Y por otro lado, ya implementa parte de las tareas adicionales necesarias para la clasificación. Tanto las transformaciones LP y BR para convertir el problema multietiqueta en uno multiclase o varios binarios, como las métricas de evaluación del rendimiento de algoritmos, 19 de ellas. mldr no incorpora algoritmos de clasificación propiamente, pero la próxima funcionalidad que estará disponible muy pronto es una interfaz común para implementar clasificadores externos y que homogeneice la interacción con el usuario.