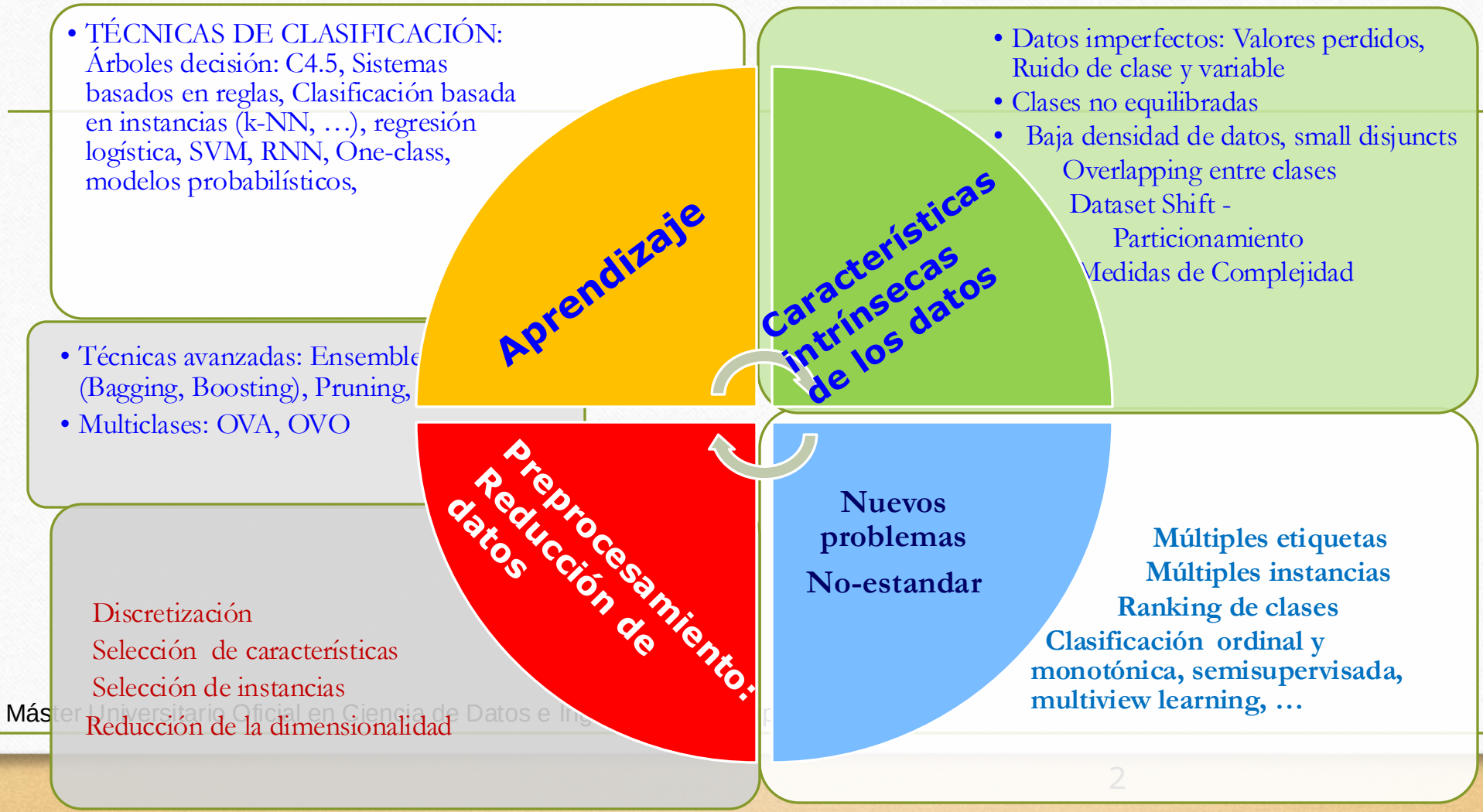
APRENDIZAJE MULTI-INSTANCIA Y MULTI-ETIQUETA

Minería de Datos: Aspectos Avanzados

Salvador García (salvagl@decsai.ugr.es)

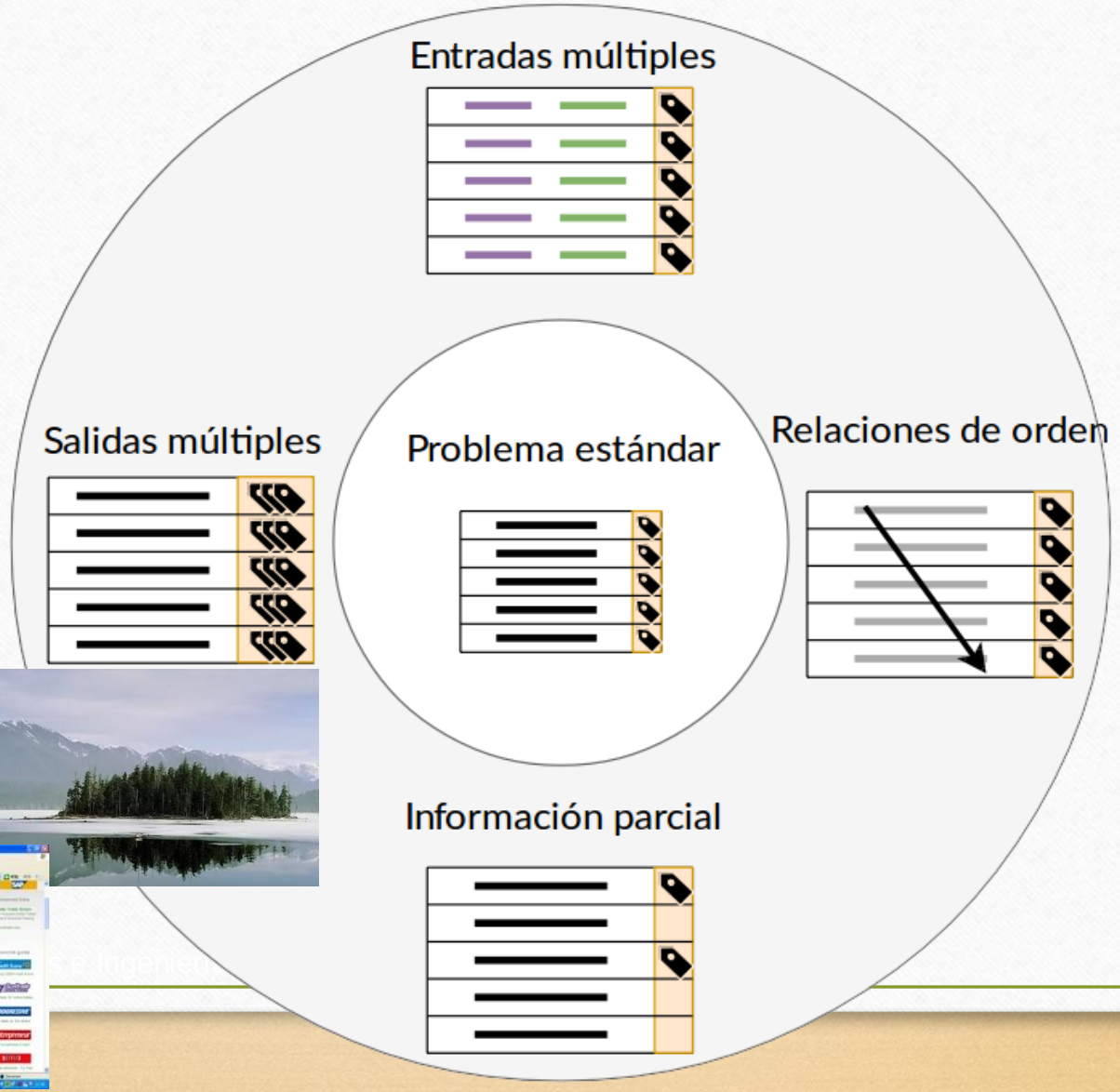
Alberto Fernández (alberto@decsai.ugr.es)

PROBLEMAS NO ESTÁNDAR DE PREDICCIÓN

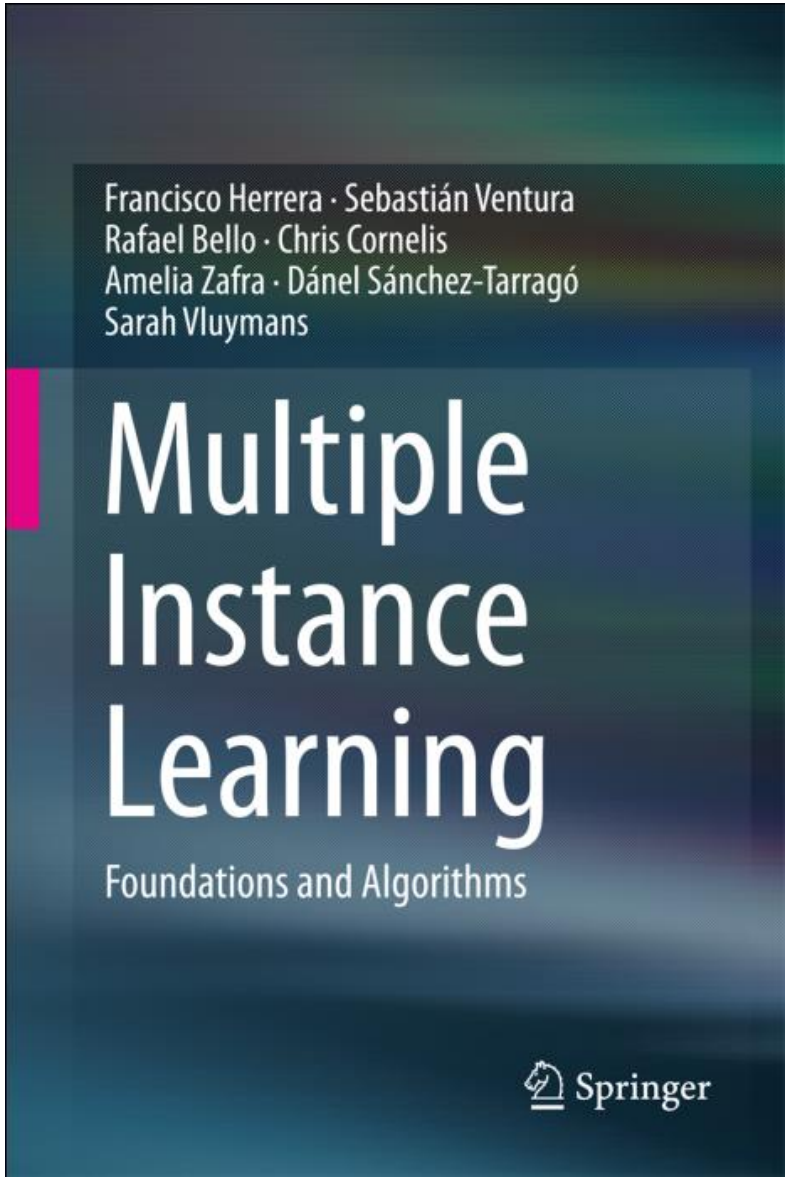


Aprendizaje supervisado no estándar

Surge por variaciones en las estructuras de entrada y salida que no se ajustan al problema estándar.



- **MIL: Multi-instance learning**
- ML: Multi-label classification



Desde ML a MIL: Aprendizaje Multi-Inastancia

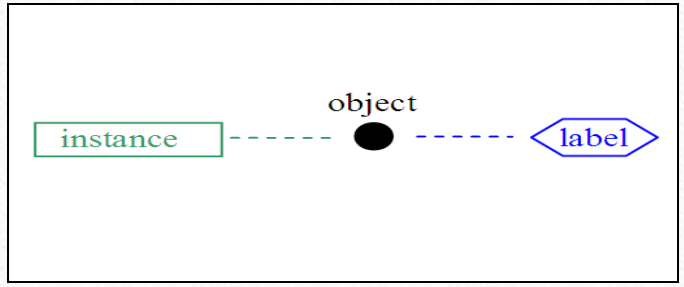
Generaliza el Machine Learning convencional

Cada instancia consiste en un conjunto (*bolsa*) de instancias

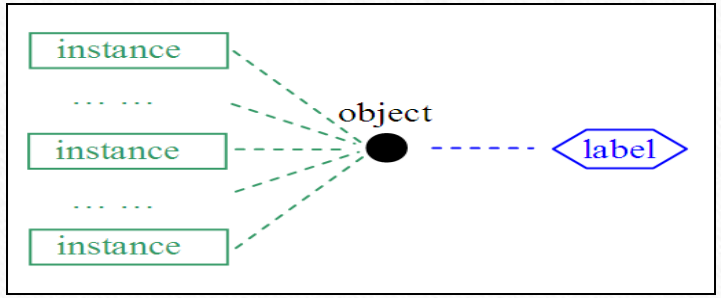
La etiqueta única de una bolsa complete es una función de las etiquetas individuales de las instancias

Multiple Instance Learning

- In MIL, identifying positive instances is an important problem
- Understanding the relation between the bag and input patterns.



Traditional supervised learning



Multi-instance learning

From ML to MIL: Multi-Instance Learning

- Originated from the research on drug activity prediction [Dietterich et al. AIJ97]
- Drugs are small molecules working by binding to the target area
  - For molecules qualified to make the drug, one of its shapes could tightly bind to the target area
  - A molecule may have many alternative shapes

- **The difficulty:**
  - **Biochemists know that whether a molecule is qualified or not, but do not know which shape responses for the qualification**

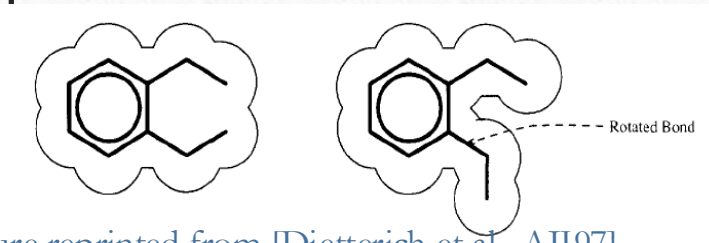
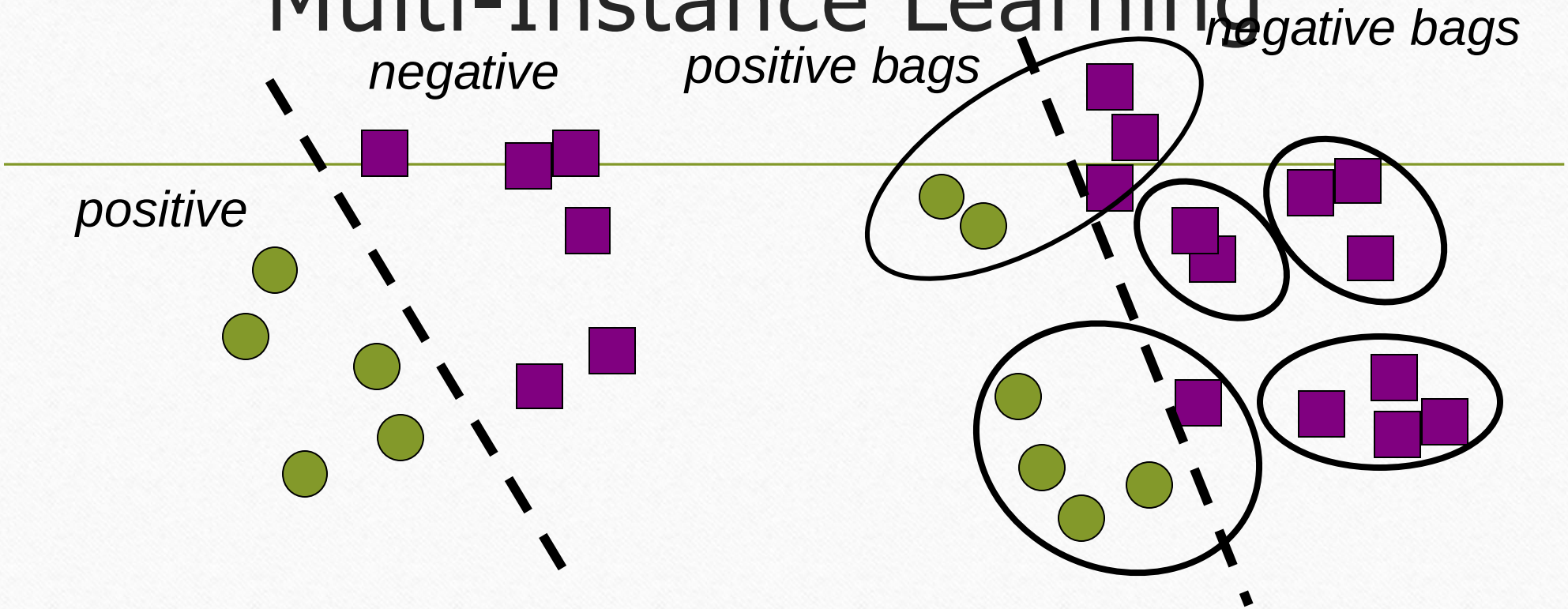


Figure reprinted from [Dietterich et al., AIJ97]

[Dietterich *et al.*, 1997] T. G. Dietterich, R. H. Lathrop, T. Lozano-Perez. Solving the Multiple-Instance Problem with Axis-Parallel Rectangles. *Artificial Intelligence Journal,* 89, 1997.

From ML to MIL: Multi-Instance Learning

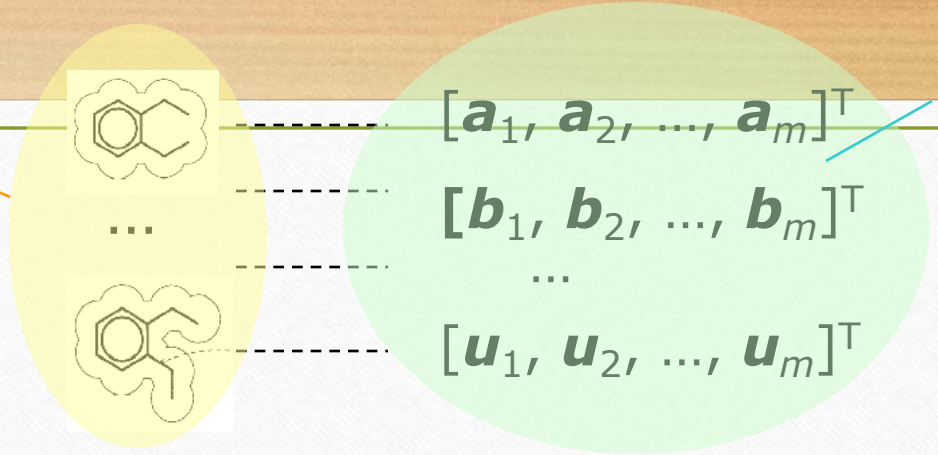


Traditional supervised learning

Multiple-instance learning

[Dietterich et al. 1997]

From ML to MIL: Multi-Instance Learning



- Each shape can be represented by a feature vector, i.e., an instance
- Thus, a molecule is a **bag** of instances
  - A bag is positive if it contains at *least one positive instance*; otherwise, it is negative
  - The labels of the training bags are known
  - The labels of the instances in the training bags **are unknown**

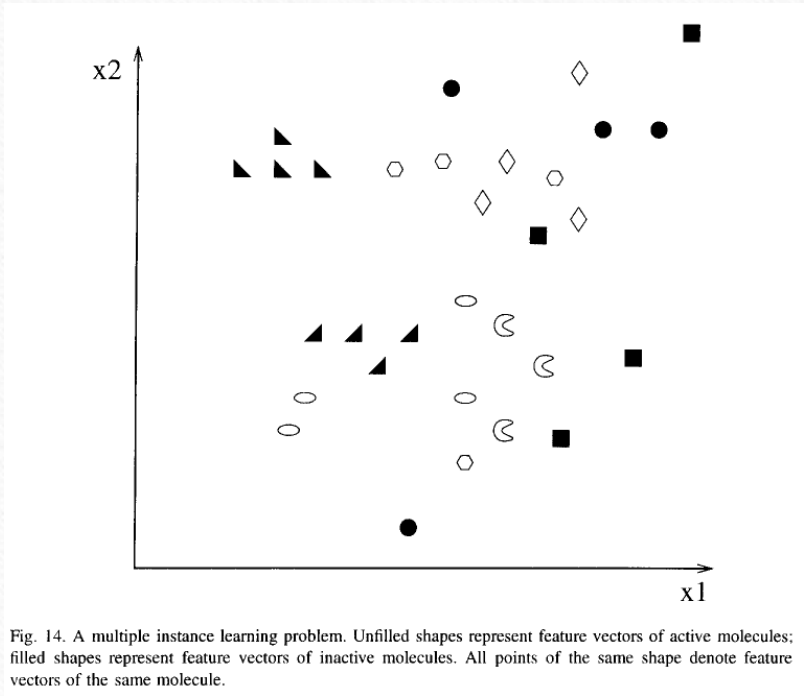


Fig. 14. A multiple instance learning problem. Unfilled shapes represent feature vectors of active molecules; filled shapes represent feature vectors of inactive molecules. All points of the same shape denote feature vectors of the same molecule.

Figure reprinted from [Zhi-Hua Zhou et al., icml09]

From ML to MIL: Summary & Example

- Bags of points
  - Labels: +1/-1 for each *bag*
- Example:
  - results of repeated medical test generate: sick/healthy bag (bag = person)
- An unseen bag is *positive* if *at least one* point in the bag is on the positive side of the decision surface
- An unseen bag is *negative* if *all* points in the bag are on the negative side of the decision surface

From ML to MIL: Multi-Instance Learning

- Multiple-instances/ Single table
- Examples as sets
- Each instance is a person
- Each set describes a family

Examples, e.g.

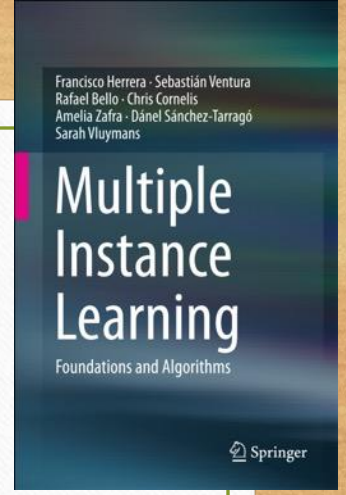
class(neg) :- person(aa,aa,aa,AA),
person(aa,aa,aa,aa).

or

{person(aa,aa,aa,AA),
person(aa,aa,aa,aa) }

Table 3.2. A multi-instance example.

Gene1	Gene2	Gene3	Gene4	Class
aa	aa	aa	AA	negative
aa	aa	aa	aa	
AA	aa	aa	AA	positive
aA	AA	aa	AA	
aA	aA	AA	AA	
aA	aA	AA	aa	
AA	aA	AA	aa	negative
aa	AA	aa	AA	
aa	aA	AA	AA	
aA	AA	AA	AA	positive
aa	AA	AA	aa	
AA	AA	aa	aa	
AA	aa	AA	AA	



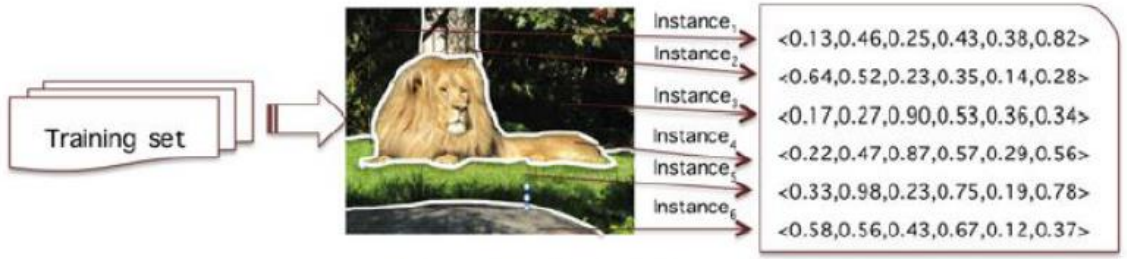
Multiple-Instance Learning. Applications

http://link.springer.com/book/10.1007%2F978-3-319-47759-6

- Predicción de actividad de fármacos
- Clasificación de imágenes por segmentos
- Predicción de bancarrotas

From ML to MIL: Multi-Instance Learning

(a) Single-instance classification



(b) Multiple instance classification

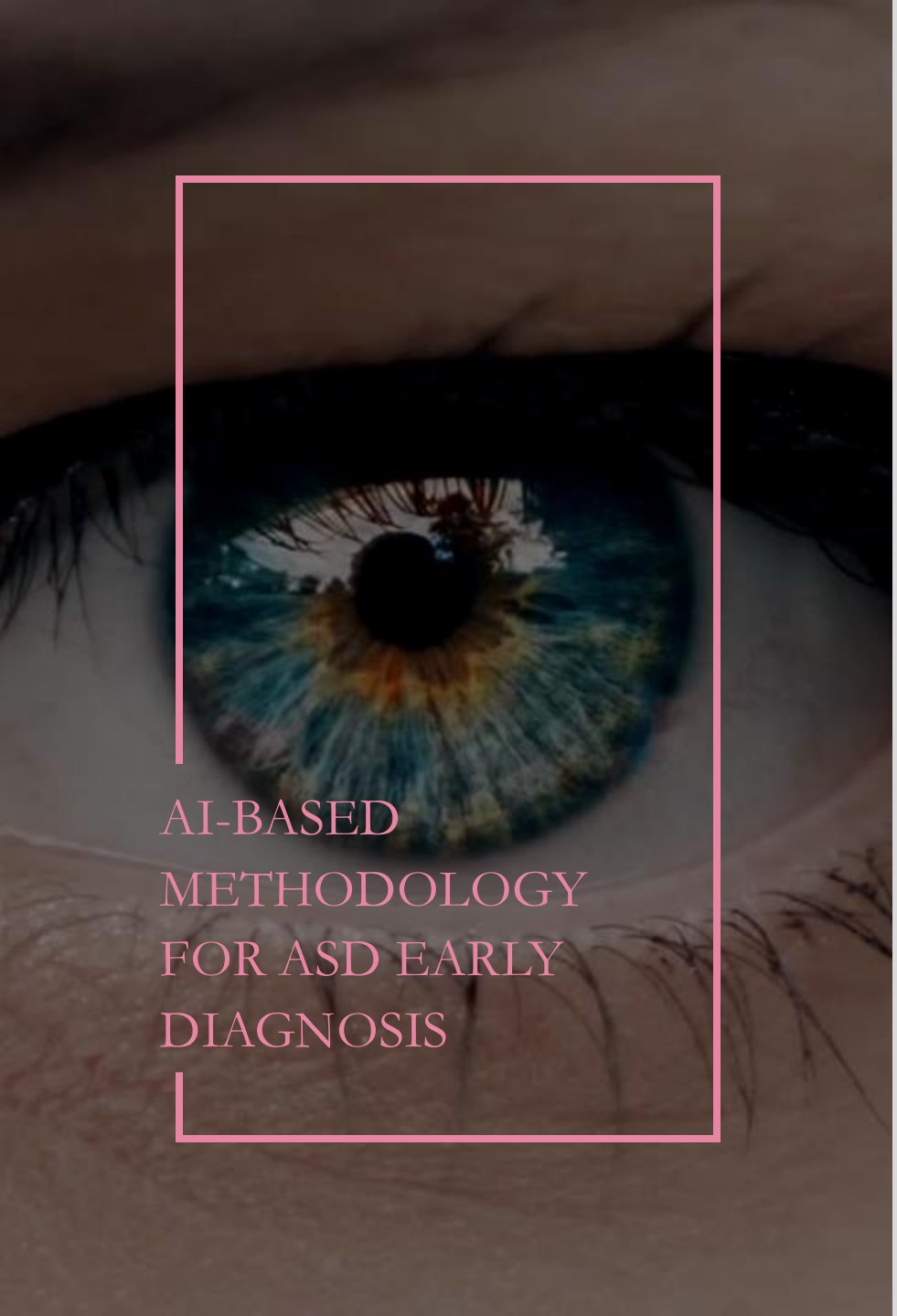
Fig. 3.2 Training data set for classification task

Multi-Instance Classifiers: Taxonomy

Paradigma del espacio de instancias: se construye un clasificador a nivel de instancia para discriminar las instancias en bolsas. El clasificador final a nivel de bolsa se obtiene agregando puntuaciones a nivel de instancia. Se considera las características de las instancias individuales, no las características de toda la bolsa (ej. miSVM).

Paradigma del espacio de bolsas: la clasificación de una nueva bolsa se basa en la información proporcionada para toda la bolsa, no para las instancias individuales (ej. Citation kNN)

Paradigma del espacio incrustado: En este paradigma se realiza un mapeo del espacio de bolsas a un único espacio vectorial. A continuación, se entrena un clasificador de instancia única tradicional (ej. SimpleMI)



AI-BASED METHODOLOGY FOR ASD EARLY DIAGNOSIS

Image processing

Feature extraction

Aggregation

Classificatio

Multi-Instance Aggregation Strategy

- *Purpose*: Combines multiple instances of eye-tracking data for a single diagnostic decision, crucial due to varied behavior in different images from same individual.
- *Aggregation Mechanism*: A sophisticated algorithm weighs individual predictions and their confidence levels to ensure the aggregated prediction accurately reflects the subject's overall data.

Aggregation mechanism

- Calculate the mean of the predictions and set a certain positive threshold.
- It based on two parameters:
  - The threshold (Δ): from which the individual would be assigned the positive class
  - The aggregation operator (g_{mi} : $[0,1]^m$): mean (arithmetic, geometric or harmonic), median or range.
- Example based on the arithmetic mean:

$$F(A_i) = g_{m_i}\begin{pmatrix} f(x_{i_1}) \\ \vdots \\ f(x_{i_{m_i}}) \end{pmatrix} = g_{m_i}\begin{pmatrix} p_{i_1} \\ \vdots \\ p_{i_{m_i}} \end{pmatrix} = \begin{cases} Control, & \text{si } \frac{\sum_{j=1}^{m_i} p_{i_j}}{m_i} <= \Delta \\ TEA, & \text{si } \frac{\sum_{j=1}^{m_i} p_{i_j}}{m_i} > \Delta \end{cases}$$

Multiple Instance Learning Methods: Bag Space

- Citation kNN
- Support Vector Machine for multi-instance learning
- Multiple-decision tree
- … …

Citation K-NN

The popular k Nearest Neighbor (k-NN) approach can be adapted for MIL problems if the distance between bags is defined.

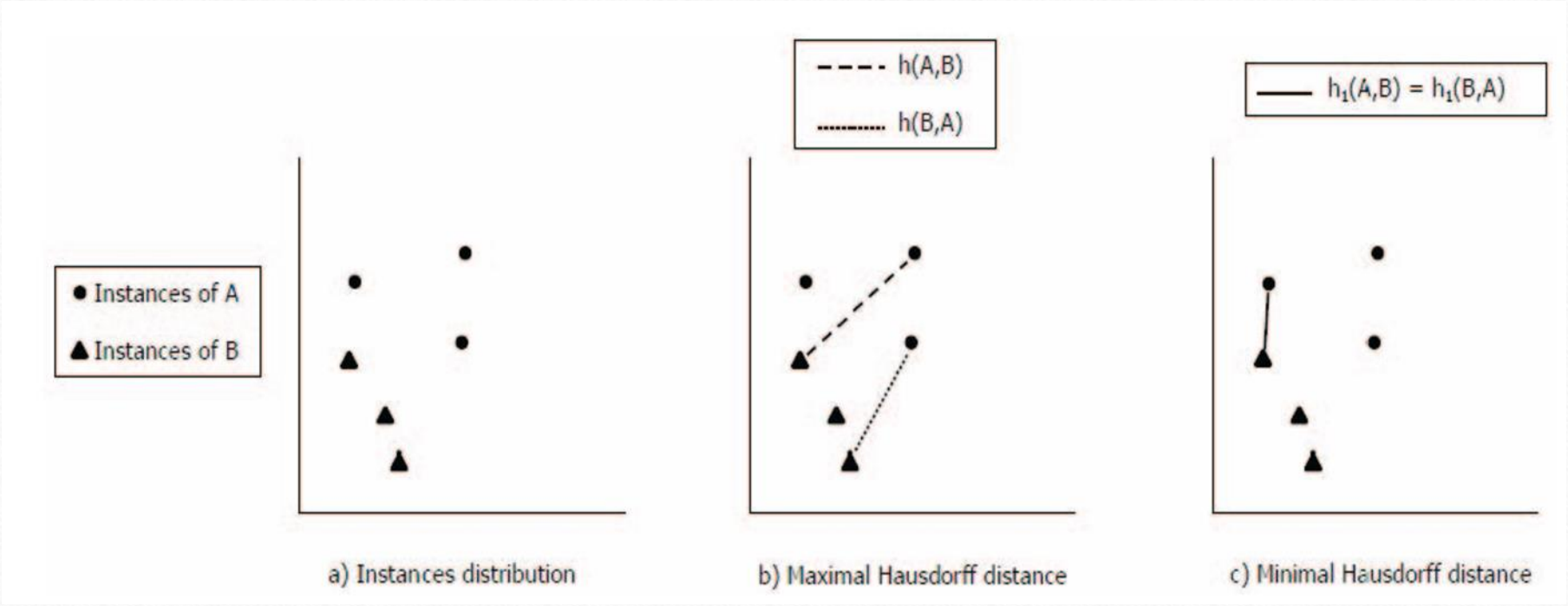
The *minimum Hausdorff distance* was used as the bag-level distance metric, defined as the shortest distance between any two instances from each bag, where A and B denote two bags, and a_i and b_j are instances from each bag.

Using this bag-level distance, we can predict the label of an unlabeled bag using the k-NN algorithm.

Distancia Hausdorff

- Mide la "discrepancia" máxima entre dos conjuntos de puntos, A y B . Calcula la mayor distancia mínima entre un punto de un conjunto y el punto más cercano del otro conjunto:
- $d_H(A,B) = \max\left(\max_{a\in A}\min_{b\in B}|a-b|, \max_{b\in B}\min_{a\in A}|b-a|\right)$, donde:
  - | a - b | es la distancia entre los puntos a y b .
  - $\max_{a\in A}\min_{b\in B}|a-b|$, busca la dist. mínima de cada punto en A al punto más cercano de B .
  - $\max_{b\in B}\min_{a\in A}|b-a|$, idem pero para cada punto de B con respecto a A .
- **Dirección del cálculo** en la distancia Hausdorff: Si un conjunto tiene una distribución de puntos más densa que el otro, la distancia puede dar diferentes resultados en algunos casos.

Hausdorff distance in a two-dimensional instance space



Citation K-NN

La etiqueta de la mayoría de los K vecinos más cercanos de una bolsa sin etiquetar puede no ser la verdadera etiqueta de esa bolsa, ya que la votación por mayoría, puede ser fácilmente confundido por las instancias positivas falsas en las bolsas positivas.

El enfoque de la citación considera no sólo a las bolsas como los vecinos más cercanos (referencias) de una bolsa B, sino también a las bolsas que cuentan con B como sus vecinos (citadores) basándose en la distancia mínima de Hausdorff. Esta diferencia se hace considerando la **dirección en el cálculo de H**.

Predice la etiqueta de una bolsa basándose en las etiquetas tanto de las referencias como de los citadores de esa bolsa. Otra alternativa es el método bayesiano, que calcula las probabilidades posteriores de la etiqueta de una bolsa desconocida basándose en las etiquetas de sus vecinos.

Decision process in Citation kNN

- Let Ne(b) represent the set of K -nearest neighbors of b and Ci(b) represent the set of C -nearest citers of b. Four values are calculated to derive the class label of a new unseen bag b:
  - Kp : Number of positive bags in Ne(b)
  - Kn : Number of negative bags in Ne(b)
  - Cp : Number of positive bags in Ci(b)
  - Cn : Number of negative bags in Ci(b)
- Once these values are calculated, the classification rule is as follows:

```
if  (Kp + Cp > Kn + Cn)  then
    class = positive,
else
    class = negative.
```

- Obviously, Kp + Kn = K , but the total number of citers (Cp + Cn) is not known a priori.
- Therefore, whether K is odd or even and whether C is odd or even, the sum Kp + Cp + Kn + Cn can be an even number.
- Thus, a tie between the number of positive bags and negative bags is possible. In the original Citation-KNN algorithm, the tie is always solved by assigning the negative class to the bag.

Citation kNN Example

- Based on the given information, when K = 3, the K-nearest neighbors of b_3 are {b_4, b_6, b_1}.
- Conversely, if C = 2, the C-nearest citers of b_4 constitute the set {b_1, b_3, b_5, b_6 }, those of b_2 form the set {b_1 }, and those of b_6 compose the set { b_3, b_5}.
- Additionally, there are no { C }-nearest citers for b_5
- In the original description of Citation-KNN, the value of C was empirically set to K + 2, reflecting the observation that citers seem to be more important than neighbors

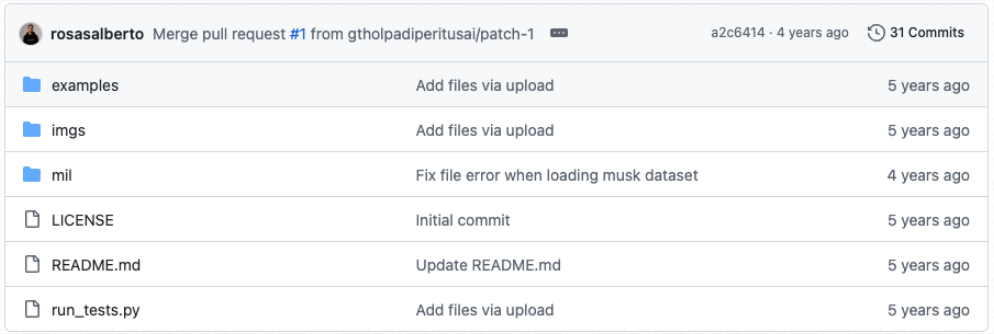
TABLE I
NEAREST NEIGHBORS OF SIX BAGS

	$K=1$	$K=2$	$K=3$	$K=4$	$K=5$
b_1	b_4	b_2	b_3	b_6	b_5
b_2	b_1	b_3	b_4	b_6	b_5
b_3	b_4	b_6	b_1	b_2	b_5
b_4	b_1	b_3	b_6	b_2	b_5
b_5	b_6	b_4	b_1	b_3	b_2
b_6	b_4	b_3	b_5	b_1	b_2

Multiple-Instance Learning: Software

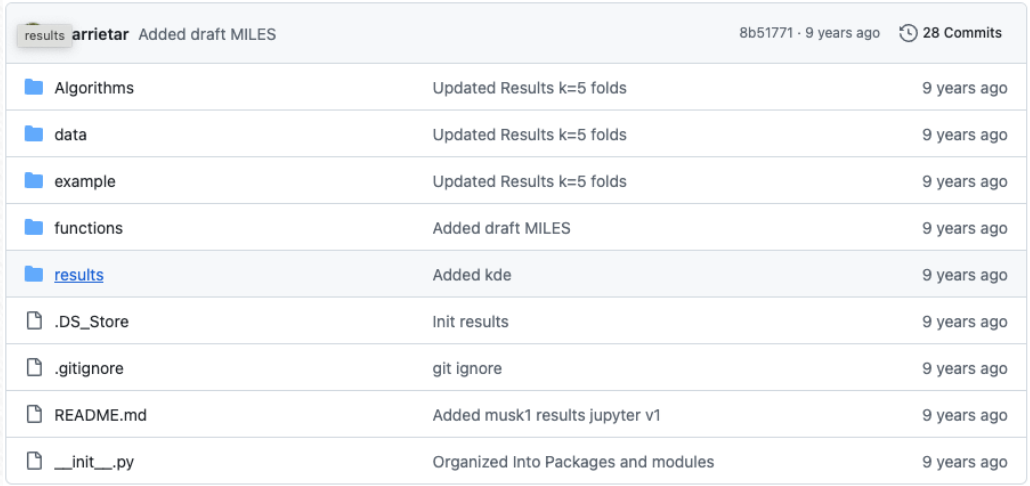
mil: multiple instance learning library for Python.

- https://github.com/rosasalberto/mil



MILpy: Multiple-Instance Learning Python Toolbox:

- https://github.com/jmarrietar/MILpy



mil: multiple instance learning library for Python

Features: The overall implementation tries to be as much user-friendly as possible. That's why most of it is constructed on top of sklearn and tensorflow.keras.

- mil.data
- mil.bag_representation
- mil.dimensionality_reduction
- mil.metrics
- mil.models
- mil.preprocessing
- mil.utils
- mil.validators
- mil.trainer

Models: It contains all the end-to-end models. All the models implement a sklearn-like structure with fit, predict, and sometimes get_positive_instances when the method allows it.

- MILES
- APR
- AttentionDeepPoolingMil

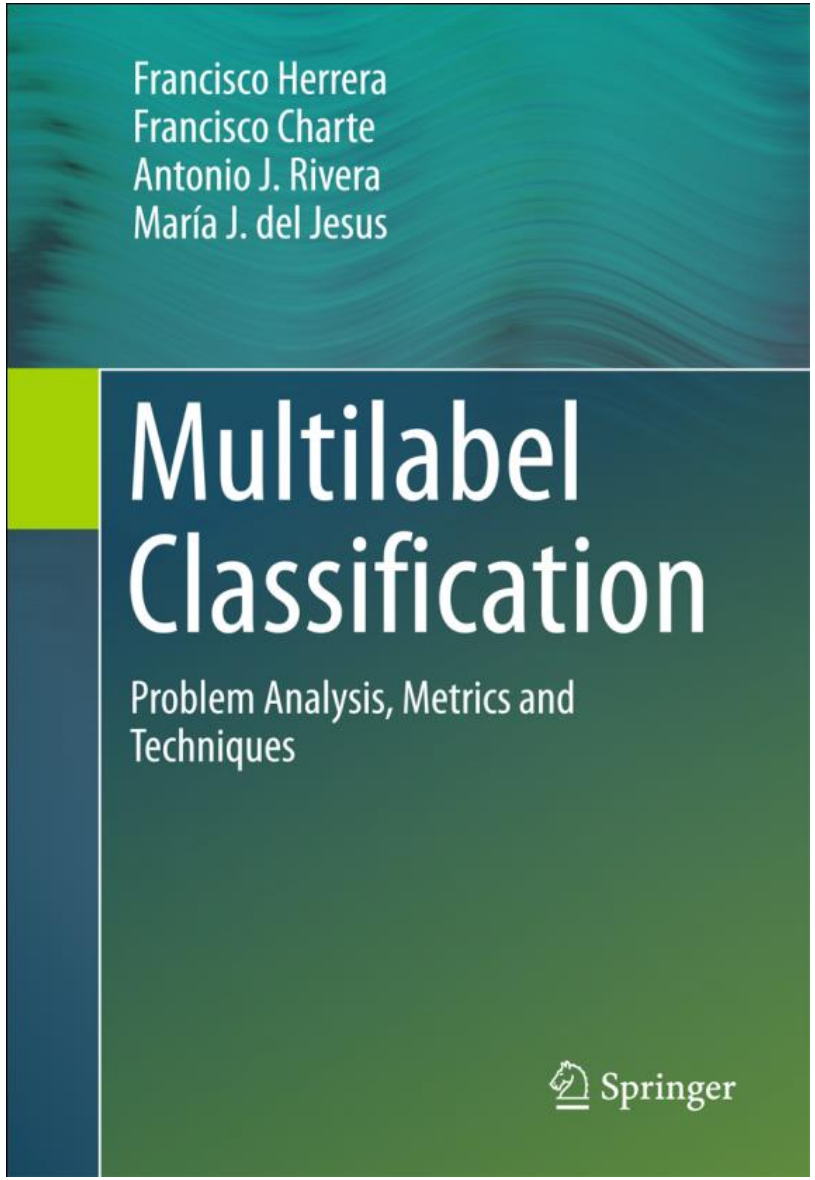
Access: https://github.com/rosasalberto/mil

A Tutorial on Multilabel Learning

EVA GIBAJA and SEBASTIÁN VENTURA, Department of Computer Science and Numerical Analysis, University of Córdoba, Spain

ACM Computing Surveys, Vol. 47, No. 3, Article 52, Publication date: April 2015.

- MIL: Multi-instance learning
- **ML: Multi-label classification**
  - Etiquetado de textos y documentos
  - Etiquetado de imágenes y escenas
  - Asignación de tags a preguntas en foros
  - Clasificación de proteínas



Variaciones no estándares

Salidas múltiples: cada instancia está asociada a varias etiquetas

Clasificación multi-etiqueta

Label distribution learning

Regresión multi-salida

Label ranking

Clasificación multi-dimensional

Motivation: Multi-label objects

- Text classification is everywhere
  - Web search
  - News classification
  - Email classificatio
  - ……
- Many text data are multi-labeled

Business

Politics

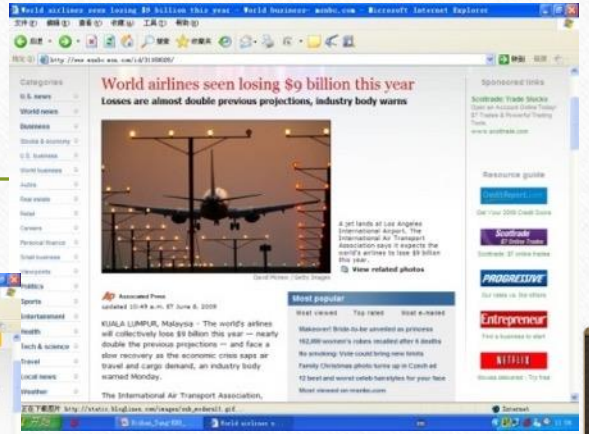
Travel

World news

Entertainment

Local news

… …



Motivation: Multi-Label Objects

e.g. natural scene image

Lake

Trees

Mountains

Multi-label learning

Documents, Web pages, Molecules.....

Multi-label classification: Example to consider

X	Y_1	Y_2	Y_3	Y_4
$x^{(1)}$	0	1	1	0
$x^{(2)}$	1	0	0	0
$x^{(3)}$	0	1	0	0
$x^{(4)}$	1	0	0	1
$x^{(5)}$	0	0	0	1

Multi-label classification

- Input $\mathcal{X} = \mathbb{R}^D$, Labelset $\mathcal{Y} = \{\lambda_1, \ldots, \lambda_L\}$, label assignment $Y \subseteq \mathcal{Y}$.
- We have set of training examples $\mathcal{D} = \{(\mathbf{x}^{(i)}, Y^{(i)})\}_{i=1}^{N} =$

$$\underbrace{\begin{bmatrix} x_1^{(1)} & x_2^{(1)} & \cdots & x_D^{(1)} \\ x_1^{(2)} & x_2^{(2)} & \cdots & x_D^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ x_1^{(N)} & x_2^{(N)} & \cdots & x_D^{(N)} \end{bmatrix}}_{\mathbf{X} \in \mathcal{X}^N} \underbrace{\begin{bmatrix} Y^{(1)} \\ Y^{(2)} \\ \vdots \\ Y^{(N)} \end{bmatrix}}_{\mathbf{Y} \in \mathcal{Y}^N}$$

where

- $\mathbf{x}^{(i)} = [x_1, \ldots, x_D] \in \mathcal{X}$ is the representation of a *data instance*
- $Y^{(i)} \subset \mathcal{Y}$ is some *label set*, where for example, $Y^{(1)} = \{\lambda_1, \lambda_4, \lambda_8\}$ are the labels relevant to $\mathbf{x}^{(1)}$.

Multi-label classification

- Input $\mathcal{X} = \mathbb{R}^D$, Output $\mathcal{Y} = \{0, 1\}^L$
- We have set of training examples $\mathcal{D} = \{(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})\}_{i=1}^{N} =$

$$\underbrace{\begin{bmatrix} x_1^{(1)} & x_2^{(1)} & \cdots & x_D^{(1)} \\ x_1^{(2)} & x_2^{(2)} & \cdots & x_D^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ x_1^{(N)} & x_2^{(N)} & \cdots & x_D^{(N)} \end{bmatrix}}_{\mathbf{X} \in \mathcal{X}^N} \underbrace{\begin{bmatrix} y_1^{(1)} & y_2^{(1)} & \cdots & y_L^{(1)} \\ y_1^{(2)} & y_2^{(2)} & \cdots & y_L^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ y_1^{(N)} & y_2^{(N)} & \cdots & y_L^{(N)} \end{bmatrix}}_{\mathbf{Y} \in \mathcal{Y}^N}$$

where

- $\mathbf{x}^{(i)} = [x_1, \ldots, x_D] \in \mathcal{X}$ is the representation of a *data instance*
- $\mathbf{y}^{(i)} = [y_1, \ldots, y_L] \in \mathcal{Y}$ is some *label vector*, where

$$y_j = \begin{cases} 1, & \text{if label } j \text{ is relevant to this instance} \\ 0, & \text{otherwise} \end{cases}$$

Equivalent notation (for $L = 10$):

$$Y^{(i)} = \{\lambda_1, \lambda_4, \lambda_8\} \Leftrightarrow \mathbf{v}^{(i)} = [1.0.0.1.0.0.0.1.0.0]$$

Multi-label classification

- L number of labels
- N number of examples
- D number of input feature attributes
- Label Cardinality (LC) $\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{L} y_j^{(i)}$ (Average number of labels per example)
- Label Density $\frac{\mathrm{LC}}{L}$ (LC divided by the number of labels)
- Diversity: $\mathrm{LC} \cdot N$
- Distinct labelsets: proportion of labelsets that are distinct
- Most frequent labelset: proportion of instances that have most frequent labelset

Multi-label classification: Methods

Problem Transformation Methods

- Transforms the multi-label problem into single-label problem(s)
- Use any off-the-shelf single-label classifier to suit requirements
- i.e., **Adapt your data to the algorithm**

Algorithm Adaptation Methods

- Adapt a single-label algorithm to produce multi-label outputs
- Benefit from specific classifier advantages (e.g., efficiency)
- i.e., **Adapt your algorithm to the data**

Many methods involve a mix of both approaches.

Multi-label classification

For example,

- Binary Relevance: L binary problems (one vs. all)
- Label Powerset: one multi-class problem of 2^L class-values
- Pairwise: $\frac{L(L-1)}{2}$ binary problems (all vs. all)
- Copy-Weight: one multi-class problem of L class values

At training time, with $\mathcal{D}$:

1. Transform the multi-label training data to single-label data
2. Learn from the single-label transformed data

At testing time, for $\tilde{\mathbf{x}}$:

1. Make single-label predictions
2. Translate these into multi-label predictions

Multi-label classification: Binary Relevance

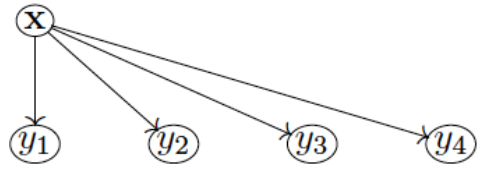
$\mathbf{X}$	Y_1
$\mathbf{x}^{(1)}$	0
$\mathbf{x}^{(2)}$	1
$\mathbf{x}^{(3)}$	0
$\mathbf{x}^{(4)}$	1
$\mathbf{x}^{(5)}$	0

$\mathbf{X}$	Y_2
$\mathbf{x}^{(1)}$	1
$\mathbf{x}^{(2)}$	0
$\mathbf{x}^{(3)}$	1
$\mathbf{x}^{(4)}$	0
$\mathbf{x}^{(5)}$	0

$\mathbf{X}$	Y_3
$\mathbf{x}^{(1)}$	1
$\mathbf{x}^{(2)}$	0
$\mathbf{x}^{(3)}$	0
$\mathbf{x}^{(4)}$	0
$\mathbf{x}^{(5)}$	0

$\mathbf{X}$	Y_4
$\mathbf{x}^{(1)}$	0
$\mathbf{x}^{(2)}$	0
$\mathbf{x}^{(3)}$	0
$\mathbf{x}^{(4)}$	1
$\mathbf{x}^{(5)}$	1

Prediction: $\hat{\mathbf{y}} = [h_1(\tilde{\mathbf{x}}), \ldots, h_L(\tilde{\mathbf{x}})]$



Disadvantages:

- Does not model label dependency, {adult, family} possible
- Class imbalance, e.g., $P(\neg\texttt{family}) \gg P(\texttt{family})$

Multi-label classification: Label Powerset

$\mathbf{X}$	$Y \in 2^L$
$\mathbf{x}^{(1)}$	0110
$\mathbf{x}^{(2)}$	1000
$\mathbf{x}^{(3)}$	0110
$\mathbf{x}^{(4)}$	1001
$\mathbf{x}^{(5)}$	0001

- complexity: many class labels
- imbalance: not many examples per class label
- overfitting: how to predict new value?

Multi-label classification: Pairwise Binary

X	Y_{1v2}
$\mathbf{x}^{(1)}$	0
$\mathbf{x}^{(2)}$	1
$\mathbf{x}^{(3)}$	0
$\mathbf{x}^{(4)}$	1

X	Y_{1v3}
$\mathbf{x}^{(1)}$	0
$\mathbf{x}^{(2)}$	1
$\mathbf{x}^{(4)}$	1

X	Y_{1v4}
$\mathbf{x}^{(2)}$	1
$\mathbf{x}^{(5)}$	0

X	Y_{2v3}
$\mathbf{x}^{(3)}$	1

X	Y_{2v4}
$\mathbf{x}^{(1)}$	1
$\mathbf{x}^{(3)}$	1
$\mathbf{x}^{(4)}$	0
$\mathbf{x}^{(5)}$	0

X	Y_{3v4}
$\mathbf{x}^{(1)}$	1
$\mathbf{x}^{(4)}$	0
$\mathbf{x}^{(5)}$	0

Predict $y_{j,k} = \mathbf{h}_{j,k}(\tilde{\mathbf{x}})$ *for all* $1 \leq j < k \leq L$

$$y_{j,k} = \begin{cases} 0, & \lambda_j \succ \lambda_k \\ 1, & \lambda_k \succ \lambda_j \end{cases}$$

Issues:

- this produces pairwise rankings, how to get a labelset?
- how much sense does it make to find a decision boundary between overlapping labels?
- can be expensive in terms of numbers of classifiers ($\frac{L(L-1)}{2}$)

Multi-label classification: Copy-Weight

$\mathbf{X}$	Y_1	Y_2	Y_3	Y_4
$\mathbf{x}^{(1)}$	0	1	1	0
$\mathbf{x}^{(2)}$	1	0	0	0
$\mathbf{x}^{(3)}$	0	1	0	0
$\mathbf{x}^{(4)}$	1	0	0	1
$\mathbf{x}^{(5)}$	0	0	0	1

. . . make a single multi-class problem with L possible class values:

$\mathbf{X}$	$Y \in \{1, \ldots, L\}$	w
$\mathbf{x}^{(1)}$	2	0.5
$\mathbf{x}^{(1)}$	3	0.5
$\mathbf{x}^{(2)}$	1	1.0
$\mathbf{x}^{(3)}$	2	1.0
$\mathbf{x}^{(4)}$	1	0.5
$\mathbf{x}^{(4)}$	4	0.5
$\mathbf{x}^{(5)}$	4	1.0

each example duplicated $|Y^{(i)}|$ times, weighted as $\frac{1}{|Y^{(i)}|}$.

Multi-label classification

Classifier Chains

X	y1
x1	0
x2	1
x3	0

Classifier 1

X	y1	y2
x1	0	1
x2	1	0
x3	0	1

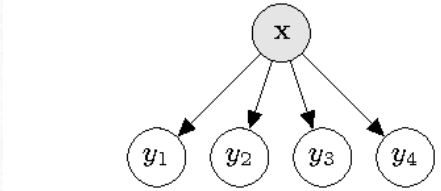
Classifier 2

X	y1	y2	y3
x1	0	1	1
x2	1	0	0
x3	0	1	0

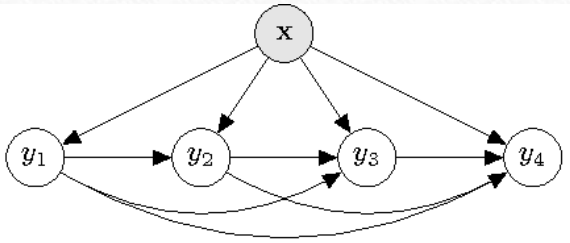
Classifier 3

X	y1	y2	y3	y4
x1	0	1	1	0
x2	1	0	0	0
x3	0	1	0	0

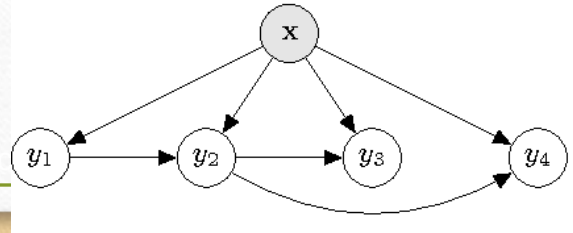
Classifier 4



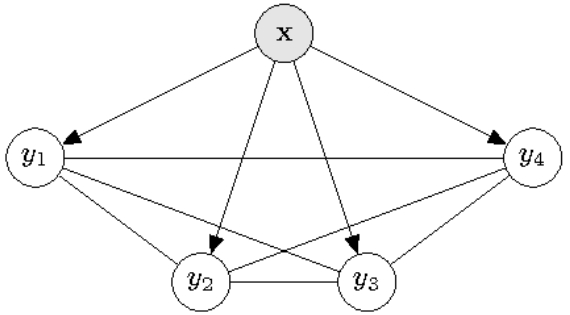
(a) Independent Classifiers (IC)



(b) Classifier Chain (CC)



(c) Bayesian Classifier Chain (BCC)



(d) Conditional Dependency Network (CDN)

Classifier Chains

TRAINING($D = \{(x_1, S_1), \cdots, (x_n, S_n)\}$)

1 **for** $j \in 1 \cdots |L|$
2 **do** ▷ single-label transformation and training
3 $D' \leftarrow \{\}$
4 **for** $(x, S) \in D$
5 **do** $D' \leftarrow D' \cup ((x, l_1, \cdots, l_{j-1}), l_j)$
6 ▷ train C_j to predict binary relevance of l_j
7 $C_j : D' \rightarrow l_j \in \{0, 1\}$

CLASSIFY(x)

1 $Y \leftarrow \{\}$
2 **for** $j \leftarrow 1$ **to** $|L|$
3 **do** $Y \leftarrow Y \cup (l_j \leftarrow C_j : (x, l_1, \cdots, l_{j-1}))$
4 **return** (x, Y) ▷ the classified example

Ensemble of Classifier Chains

An ensemble of chains performs better because it not only captures relationship between labels but also does not make strong assumptions about their correct order.

ECC trains m CC classifiers C1,C2,⋯ ,Cm. Each Ck is trained with:

- A random chain ordering (of L); and
- A random subset of D.

These predictions are summed by label so that each label receives a number of votes.

A threshold is used to select the most popular labels which form the final predicted multi-label set.

Ensemble of LP: RAkEL

In the same way as EBR and ECC it uses bagging to construct an ensemble of classifiers.

Random k-labELsets (RAkEL) induces a partition over the label space and then trains an LP classifier over each partition to later concatenate the outputs of each classifier.

RAkEL presents the advantage that through dividing the label space, each LP task is much simpler since they only consider a small portion of the labels. Moreover, the distribution of the labelsets are more balanced in this manner than by considering the full dataset.

RAkEL presents two variants, RAkELo and RAkELd, whose difference is whether these labelsets can overlap with each other or not, respectively.

Multi-label classification: Ensemble-based Voting

Ensemble methods (e.g., RAkEL, EPS) make prediction via a voting scheme. For some test instance $\tilde{\mathbf{x}}$:

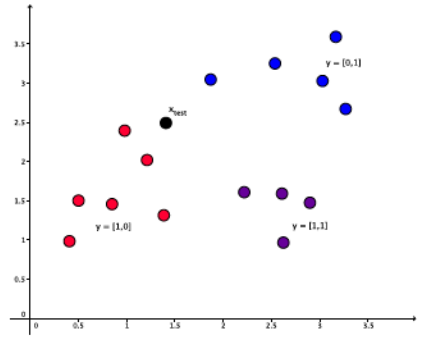
	$\hat{y}_1$	$\hat{y}_2$	$\hat{y}_3$	$\hat{y}_4$
$\mathbf{h}^1(\tilde{\mathbf{x}})$	1	0	1	
$\mathbf{h}^2(\tilde{\mathbf{x}})$		1	1	0
$\mathbf{h}^3(\tilde{\mathbf{x}})$	1		1	0
$\mathbf{h}^4(\tilde{\mathbf{x}})$	1	0		0
$\mathbf{h}(\tilde{\mathbf{x}})$	3	1	3	0
$\hat{\mathbf{y}}$	1	0	1	0

(majority vote; can also use weighted vote, *threshold*)

- more predictive power (ensemble effect)
- can predict new label combinations

Multi-label classification: k-NN

- *k*NN assigns to $\tilde{\mathbf{x}}$ the majority class of the k 'nearest neighbours'
- ML*k*NN [Zhang and Zhou, 2007] assigns to $\tilde{\mathbf{x}}$ the most common *labels* of the k nearest neighbours neighbours



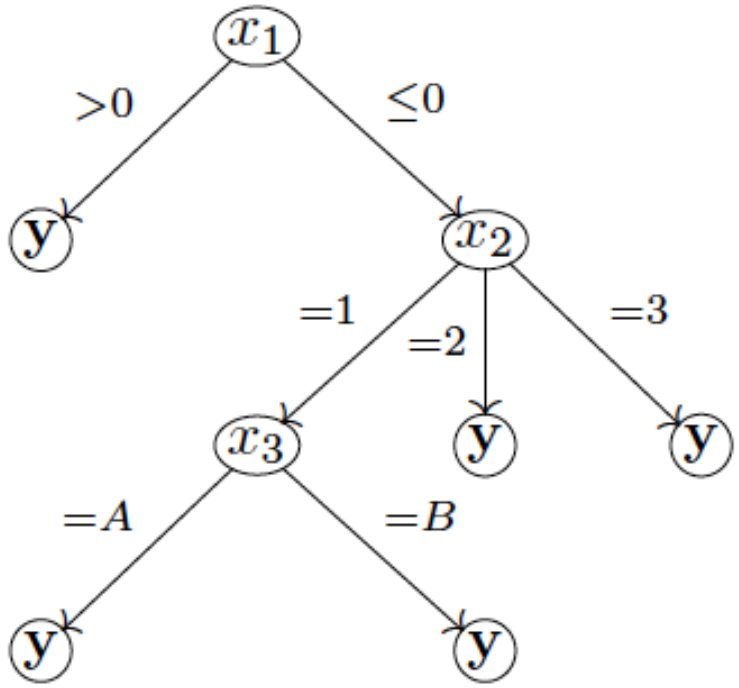
- . . . combined with Bayesian inference (MAP principle):

MLkNN provides a probabilistic framework for assigning multiple labels to instances.

Model Training:	During the training phase, MLkNN builds a model by storing the feature vectors and corresponding labels of the training instances.
Label Space Transformation	MLkNN transforms the multi-label problem into multiple binary classification problems. Each label in the dataset is treated as a separate binary classification task.
k-Nearest Neighbors Search:	When a new instance is presented, MLkNN identifies its k-nearest neighbors from the training set based on some distance metric (e.g., Euclidean distance).
Label Assignment:	For each label, MLkNN assigns a label based on the labels of its k-nearest neighbors.
Probability Estimation:	It calculates the probability of each label by considering the frequency of the label among the labels of the k-nearest neighbors.
Threshold Determination:	If the probability of a label exceeds a threshold, it is assigned to the instance; otherwise, it is not.
Prediction:	Labels for the new instance based on the probabilities calculated for each label. It assigns the labels with probabilities exceeding the threshold.

- Multi-label C4.5 [Clare and King, 2001]: Extension of the popular C4.5 decision tree algorithm; with multi-label entropy:

$$H_{\mathrm{ML}}(S) = \sum_{j=1}^{L} P(y_j)\log(P(y_j)) + (1 - P(y_j))\log(1 - P(y_j))$$



- constructed just like C4.5
- allows multiple labels at the leaves

Multi-label classification: Métricas

HAMMING LOSS

$$= \frac{1}{NL} \sum_{i=1}^{N} \sum_{i=1}^{L} \mathcal{I}[\hat{y}_j^{(i)} \neq y_j^{(i)}]$$
$$= 0.20$$

0/1 LOSS

$$= \frac{1}{N} \sum_{i=1}^{N} \mathcal{I}(\hat{\mathbf{y}}^{(i)} \neq \mathbf{y}^{(i)})$$
$$= 0.60$$

ACCURACY

$$= \frac{1}{N} \sum_{i=1}^{N} \frac{|\hat{\mathbf{y}}^{(i)} \wedge \mathbf{y}^{(i)}|}{|\hat{\mathbf{y}}^{(i)} \vee \mathbf{y}^{(i)}|}$$
$$= \frac{1}{5}(\frac{1}{3} + 1 + 1 + \frac{1}{2} + \frac{1}{2})$$
$$= 0.67$$

Calcula la fracción de etiquetas incorrectamente predichas en comparación con todas las etiquetas. Es útil para evaluar la precisión de las predicciones en todas las etiquetas.

	$\mathbf{y}^{(i)}$	$\hat{\mathbf{y}}^{(i)}$
$\tilde{\mathbf{x}}^{(1)}$	[1 0 1 0]	[1 0 0 1]
$\tilde{\mathbf{x}}^{(2)}$	[0 1 0 1]	[0 1 0 1]
$\tilde{\mathbf{x}}^{(3)}$	[1 0 0 1]	[1 0 0 1]
$\tilde{\mathbf{x}}^{(4)}$	[0 1 1 0]	[0 1 0 0]
$\tilde{\mathbf{x}}^{(5)}$	[1 0 0 0]	[1 0 0 1]

Multi-label classification: Métricas

Precisión (Precision):

- Proporción de instancias positivas correctas entre todas las instancias clasificadas positivas. Es útil para evaluar la relevancia de las etiquetas predichas.

Recall:

- Proporción de instancias positivas correctas entre todas las instancias positivas. Útil para evaluar la capacidad del modelo en capturar las instancias relevantes.

F1-score:

- Es la media armónica de precisión y recuperación. Proporciona una medida única que tiene en cuenta tanto la precisión como la recuperación.

Subconjunto exacto (Exact Match Ratio):

- Mide la proporción de instancias clasificadas correctamente para todas las etiquetas. Solo considera una predicción correcta si todas las etiquetas están clasificadas correctamente.

Jaccard Score (Índice de Jaccard):

- Calcula la similitud entre dos conjuntos dividiendo el tamaño de la intersección entre el tamaño de la unión. En el contexto de la clasificación multi-etiqueta, mide la similitud entre las etiquetas predichas y las etiquetas verdaderas.

Software: Mulan: An Open Source Library for Multi-Label Learning

Machine Learning & Knowledge Discovery Group

Learning from Multi-Label Data

Java library for **Mu**lti-**la**bel lear**n**ing, called **Mulan**

Mulan is hosted at SourceForge, so you can grab latest releases from there, as well as the latest development source code from the project's public SVN repository.

There is a collection of several multilabel datasets, properly formatted for use with Mulan.

… …

http://mlkd.csd.auth.gr/multilabel.html

- Learning approaches, applications, software:

mldr: Exploratory Data Analysis and Manipulation of Multi-Label Data Sets

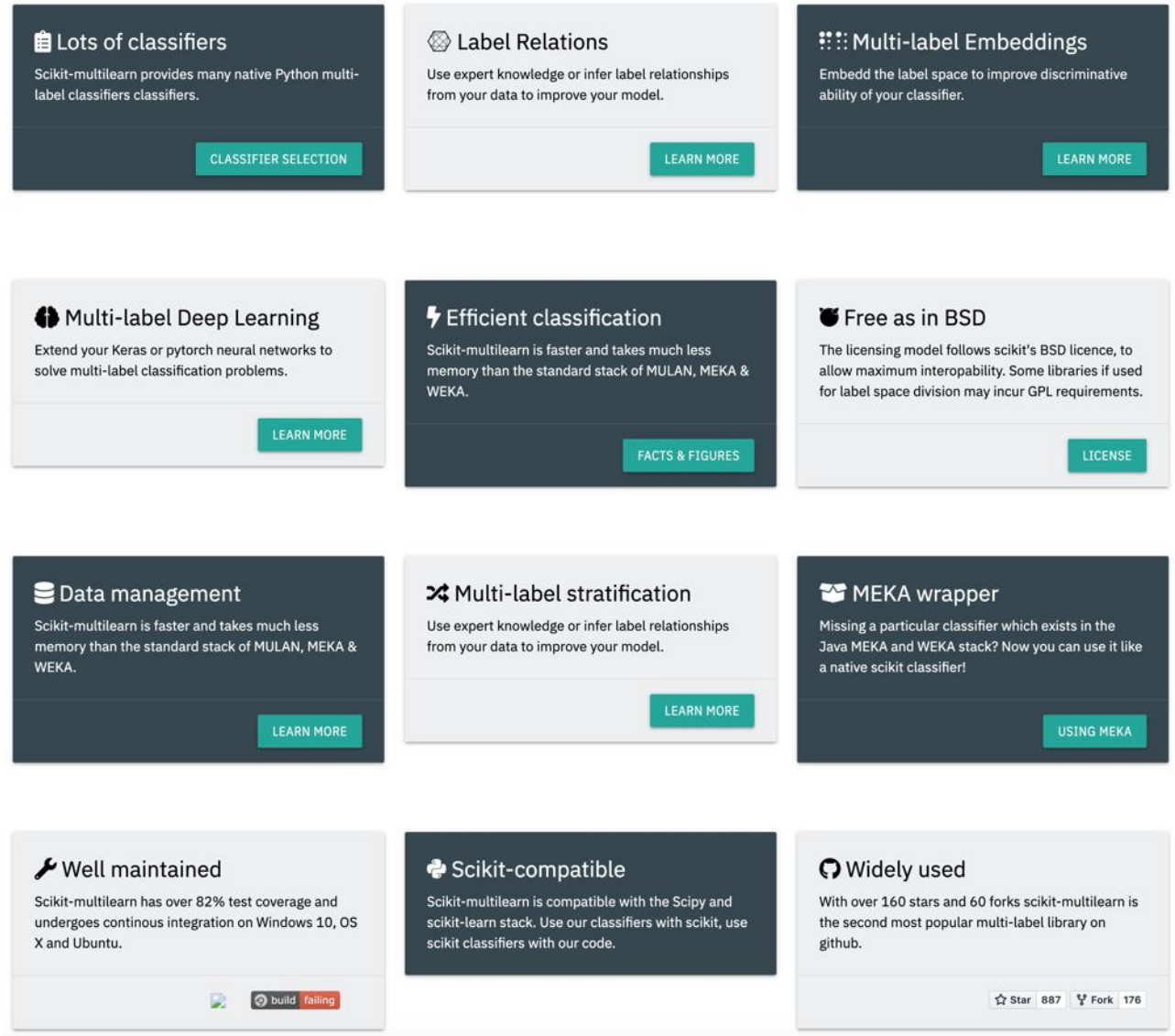
mldr: Exploratory Data Analysis and Manipulation of Multi-Label Data Sets

Exploratory data analysis and manipulation functions for multi- label data sets along with an interactive Shiny application to ease their use.

Version: 0.4.3
Depends: R (≥ 3.0.0)
Imports: shiny (≥ 0.11), XML, circlize, graphics, grDevices, stats, methods
Suggests: pROC, knitr, mldr.datasets, testthat
Published: 2019-12-19
Author: David Charte ORCID iD [cre], Francisco Charte ORCID iD [aut], Antonio J. Rivera [aut]
Maintainer: David Charte <fdavidcl at ugr.es>
License: LGPL (≥ 3) | file LICENSE
URL: https://github.com/fcharte/mldr
NeedsCompilation: no
Citation: mldr citation info
Materials: README
CRAN checks: mldr results
Documentation:
Reference manual: mldr.pdf
Vignettes: Working with Multilabel Datasets in R: The mldr Package
Downloads:
Package source: mldr_0.4.3.tar.gz
Windows binaries: r-devel: mldr_0.4.3.zip, r-release: mldr_0.4.3.zip, r-oldrel: mldr_0.4.3.zip
macOS binaries: r-release (arm64): mldr_0.4.3.tgz, r-oldrel (arm64): mldr_0.4.3.tgz, r-release (x86_64): mldr_0.4.3.tgz
Old sources: mldr archive
Reverse dependencies:
Reverse depends: utiml
Reverse imports: mldr.resampling
Reverse suggests: mldr.datasets, mlr
Linking:
Please use the canonical form https://CRAN.R-project.org/package=mldr to link to this page

Scikit-Multilearn

Available at
http://scikit.ml/



Otros problemas:

- multi-dimensional / multi-objective learning; $y_j \in \{1, \ldots, K\}$

X_1	X_2	X_3	X_4	X_5	sex	cat.	type
x_1	x_2	x_3	x_4	x_5	F	4	A
x_1	x_2	x_3	x_4	x_5	M	2	B
x_1	x_2	x_3	x_4	x_5	F	3	C

- multi-target regression; $y_j \in \mathbb{R}$

X_1	X_2	X_3	X_4	X_5	price	age	percent
x_1	x_2	x_3	x_4	x_5	37.00	25	0.88
x_1	x_2	x_3	x_4	x_5	22.88	22	0.22
x_1	x_2	x_3	x_4	x_5	88.23	11	0.77

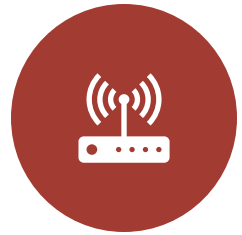
- multi-task; data may come from different sources, e.g., different text corpora
- label ranking; interested in label preferences e.g., $\lambda_3 \succ \lambda_1 \succ \lambda_4 \succ \ldots \succ \lambda_2$

- Multi-Instance Multi-Label Learning

Multi-vista

CATEGORIZACIÓN DE TEXTO MULTILINGÜE

DETECCIÓN DE CARAS CON VARIAS POSES



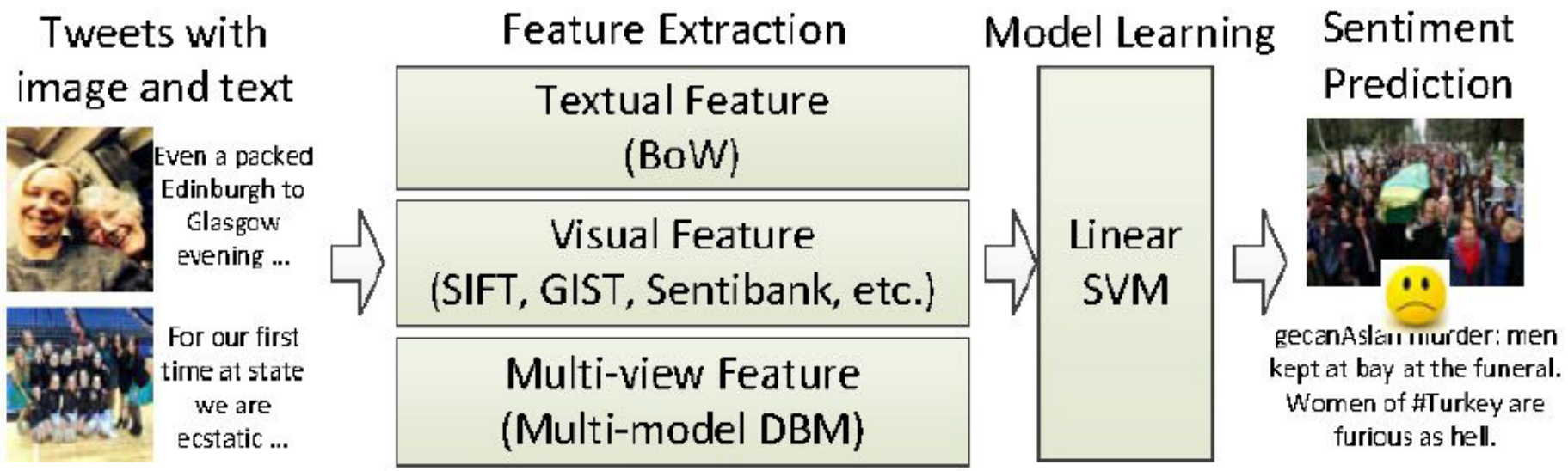
LOCALIZACIÓN DE USUARIOS DE REDES WIFI

CLASIFICACIÓN DE ANUNCIOS (IMÁGENES+TEXTO)

CLASIFICACIÓN DE IMÁGENES CON VARIAS VISTAS BASADAS EN COLORES Y TEXTURAS

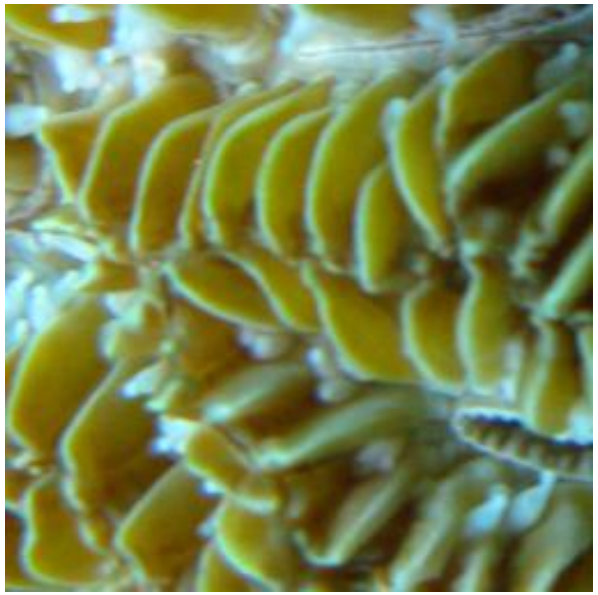
Multi-view learning is concerned with the problem of machine learning from data represented by multiple distinct feature sets.

Example: http://www.mcrlab.net/wp-content/uploads/2015/08/framework.jpg
multi-view sentiment analysis.



Multi-vista: Extracciones sobre una imagen

Multi-vista: Corales

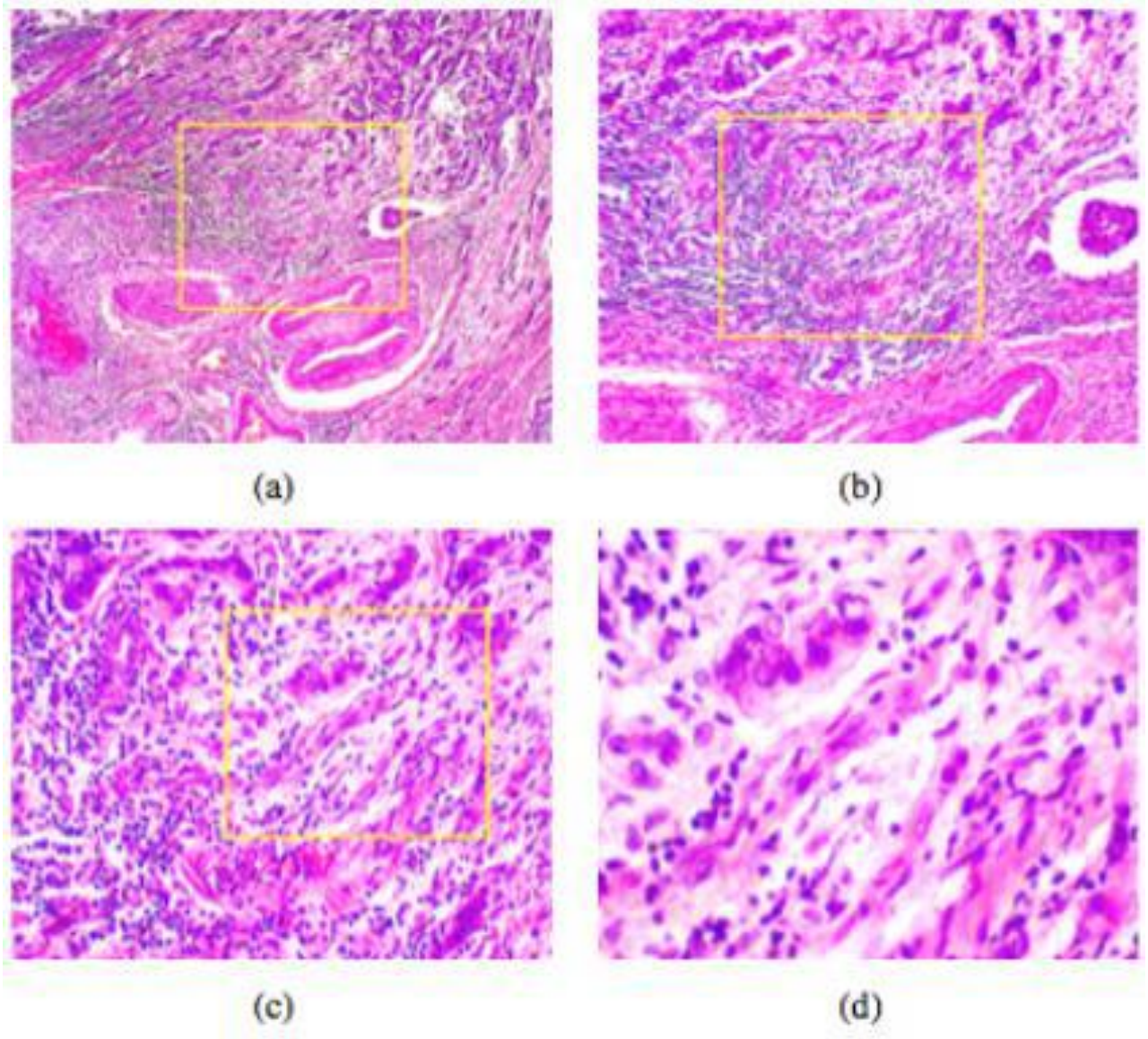


Multi-vista: Corales

Multi-vista: Imágenes de cáncer

- Muestras de tejido correspondientes a cáncer de mama maligno:

- a)Magnificación 40x
- b)Magnificación 100x
- c)Magnificación 200x
- d)Magnificación 400x



Regresión multi-salida: modelado de ecosistemas

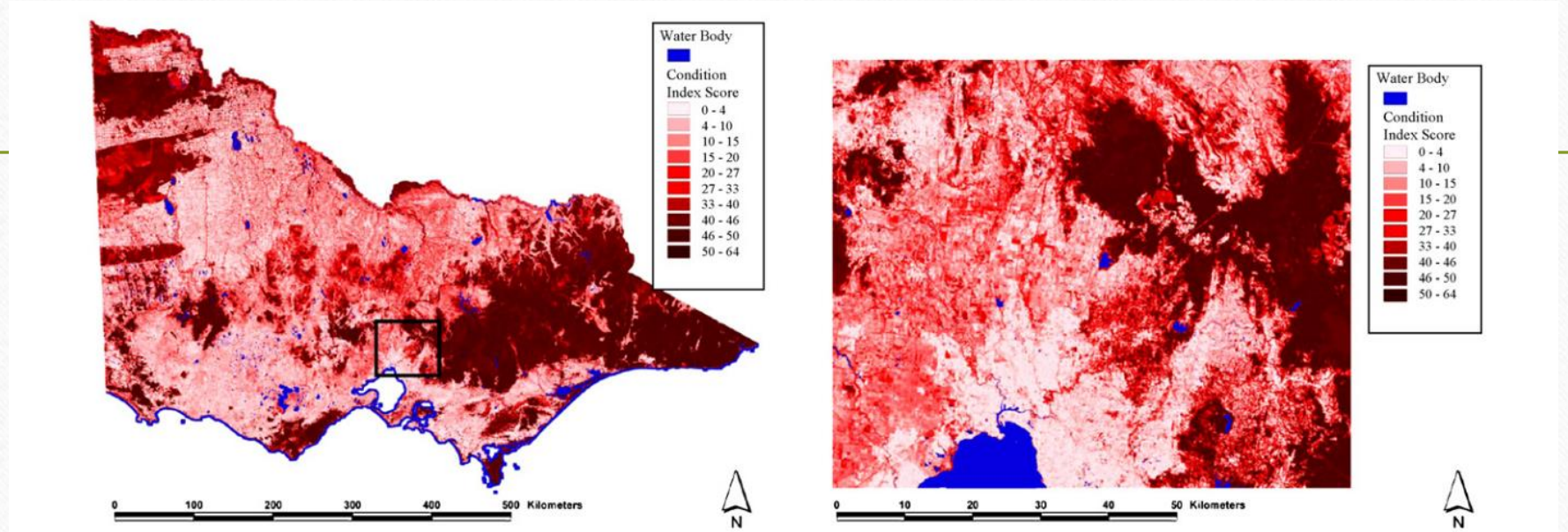
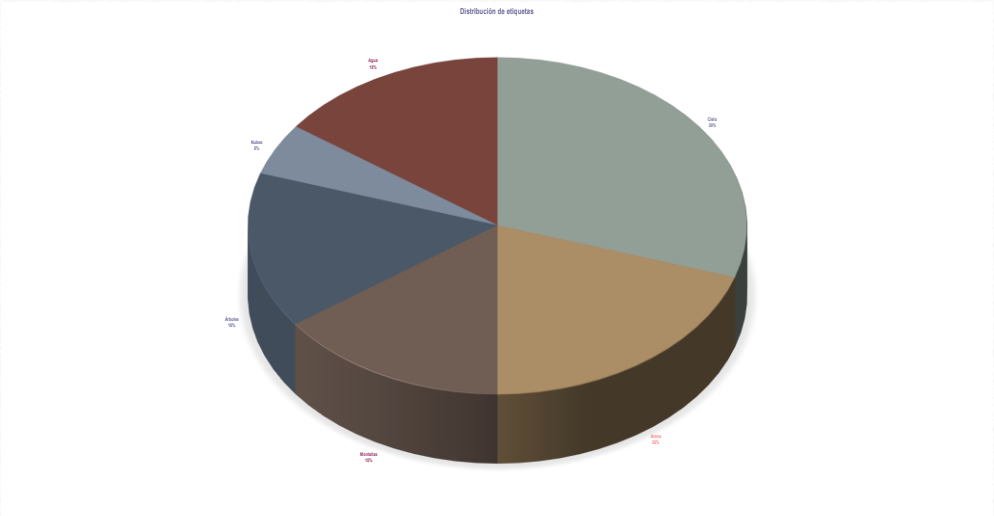


Fig. 7. Map of the condition of indigenous remnant vegetation in Victoria derived from the application of the random forests of MTRTs (left-hand side figure). The dark bordered rectangular inset refers to the area represented at higher resolution at the right-hand side figure.

La *Condition Index Score* es una suma de las 7 variables objetivo: *Weeds, Tree Canopy, Understorey, Recruitment, Logs, Litter* y *Large Tree*.

Kocev, D., Džeroski, S., White, M. D., Newell, G. R., & Griffioen, P. (2009). Using single-and multi-target regression trees and ensembles to model a compound index of vegetation condition. Ecological Modelling, 220(8), 1159-1168.

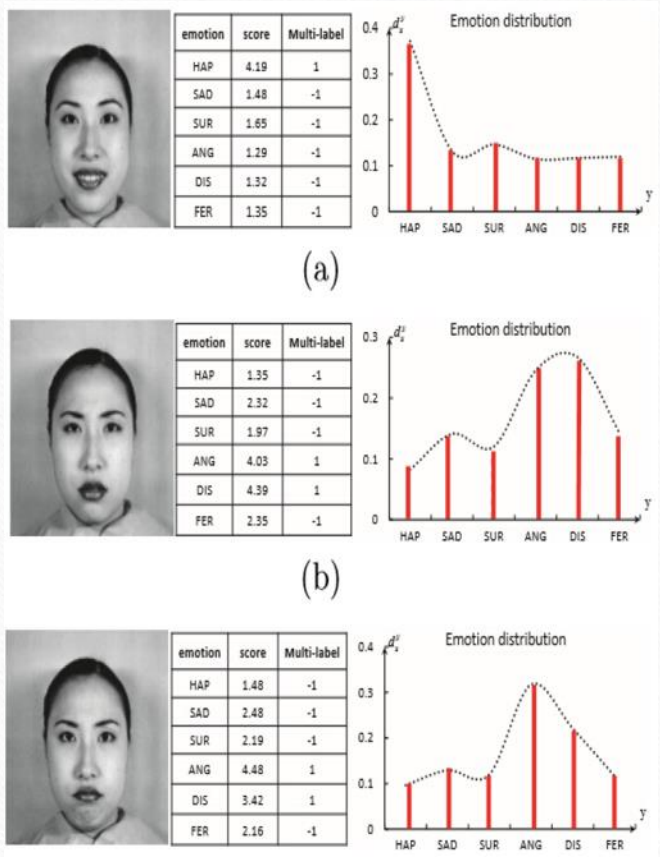
Multi-Label vs. Label distribution:
Una generalización del problema de clasificación



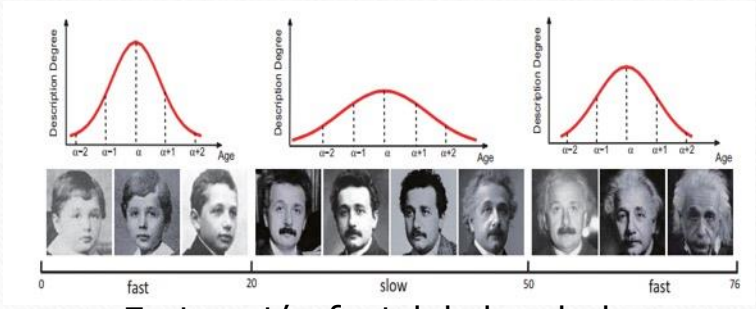
- Single Label learning (c salidas):
  - Playa
- Multi-Label Learning (2^c – 1 salidas)
  - Arena,
  - Cielo,
  - Agua…
- Label Distribution Learning (inf salidas)
  - Mayormente cielo
  - Bastante área
  - Bastante Agua…

LDL: Un gran número de aplicaciones reales

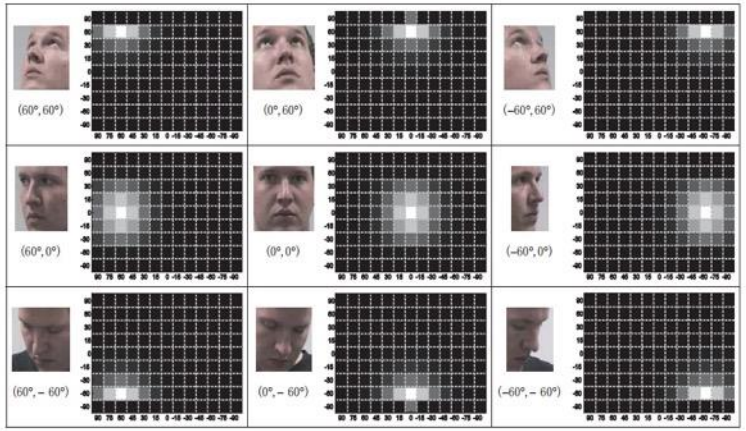
LDL presenta un paradigma que se ajusta natural a la complejidad real de los problemas
En estos casos existe una "ambigüedad" de etiquetas



Detección de emociones en la expresión facial.



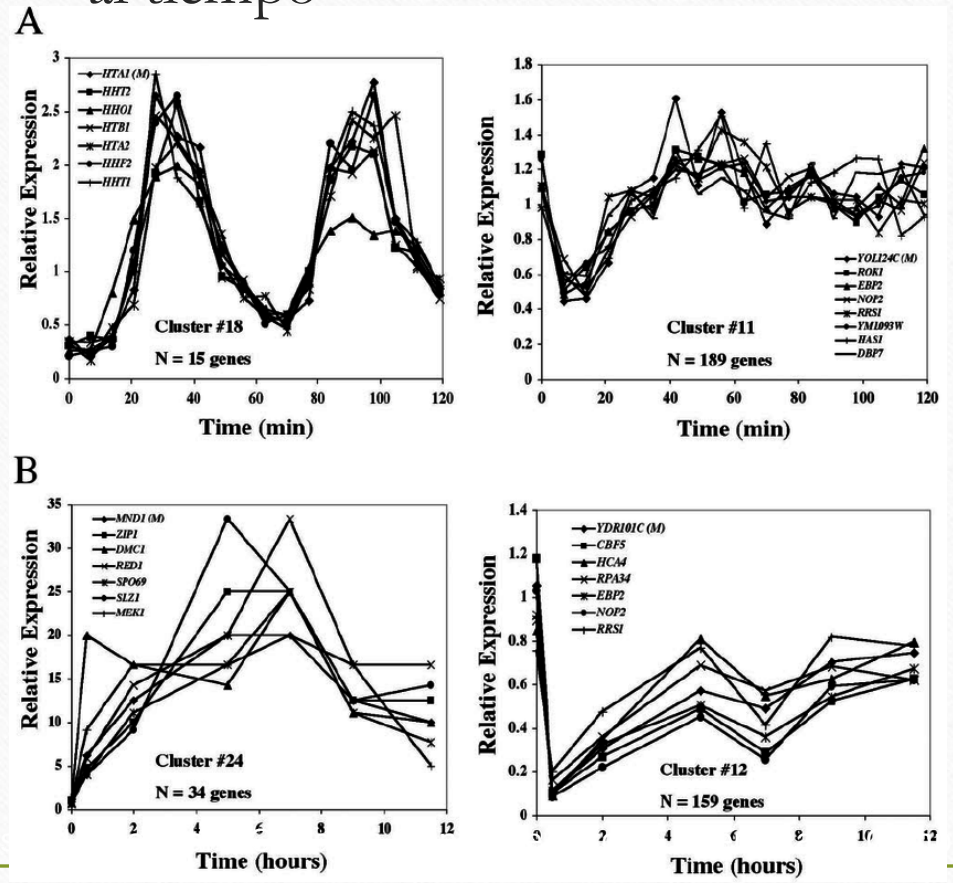
Estimación facial de la edad.



Detección de la orientación de la cara.

Aplicaciones reales: distribuciones muli-modales

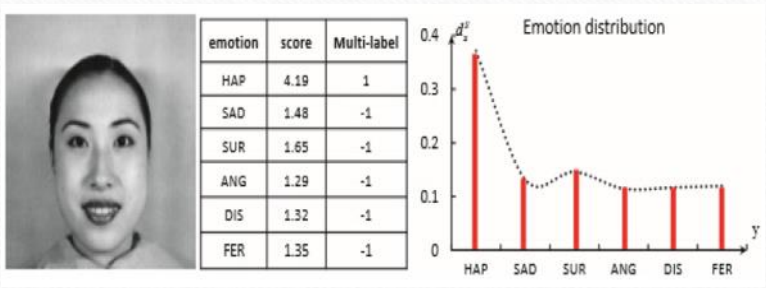
- Expresión genética de acuerdo al tiempo



- Ranking películas

Fundamentos de label distribution learning

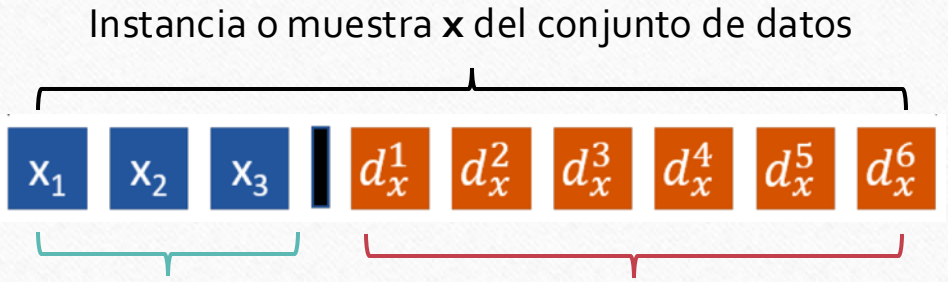
- LDL es un paradigma de aprendizaje capaz de abordar naturalmente problemas de múltiples salidas y explotar la *relación* entre dichas salidas.
- La pregunta es: CUÁNTO describe cada etiqueta al ejemplo vs. CUÁL(ES) etiqueta describe
- La salida no es un valor de confianza o probabilidad si no una "proporción" sobre la descripción completa



emotion	score	Multi-label
HAP	4.19	1
SAD	1.48	-1
SUR	1.65	-1
ANG	1.29	-1
DIS	1.32	-1
FER	1.35	-1

Ejemplo de distribución de etiquetas

Instancia o muestra **x** del conjunto de datos



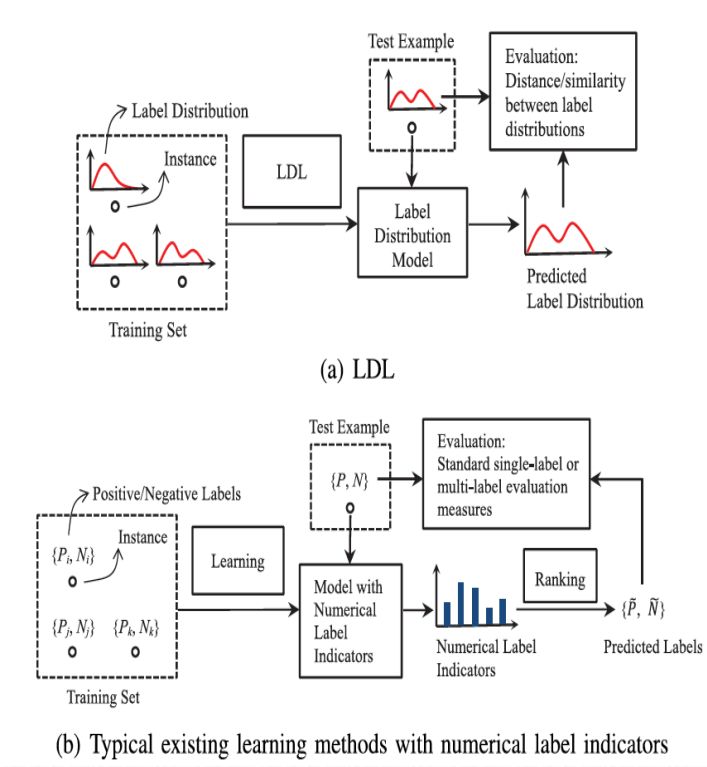
Todo problema Multi-Target Learning es LDL si se cumplen dos conceptos importantes:

- **Grado de descripción**
- **Distribución de etiquetas**

Propiedades de LDL:

1. Cada grado de descripción d_x^y debe tener un valor real entre [0,1].
2. La suma de la distribución de etiquetas debe ser la unidad.

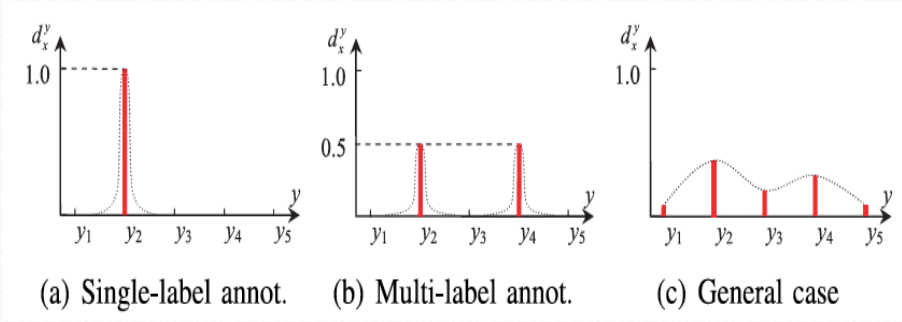
Label Distribution Learning vs. Estándar Machine learning



La distribución de etiquetas es una parte natural del problema, no se hace a posteriori.

Para LDL, el valor de cada salida es importante. En aprendizaje estándar se usa un ranking y una función de umbral (ej. arg-max)

Las métricas de evaluación del rendimiento deben adaptarse a la nueva definición de salida.



Métricas de evaluación

- Partimos de un vector de salida numérica:
  - Chequear proximidad del vector de salida predicho vs. vector original
  - Hay una cierta similitud regresión multi-salida (MTR)
    - Una primera aproximación sería utilizar el error cuadrático medio
    - Necesitamos agregar el error cometido en cada salida "individual"
  - Una forma más apropiada de medir las diferencias predicción vs. real
    - Cálculo de la distancia
    - Cálculo de la similitud
  - Además, debemos considerar si se cumple fielmente la restricción de suma 1 (en caso de modelos "laxos").
  - Finalmente, puede ser interesante analizar en qué salidas existe mayor diferencia con el original

Métricas de evaluación

¿Cómo comprobar la calidad de los modelos aprendidos?

Measure	Formula
Chebyshev ↓	$Dis_1(D,\hat{D}) = \max_j \lvert d_j - \hat{d}_j \rvert$
Clark ↓	$Dis_2(D,\hat{D}) = \sqrt{\sum_{j=1}^{c} \frac{(d_j - \hat{d}_j)^2}{(d_j + \hat{d}_j)^2}}$
Canberra ↓	$Dis_3(D,\hat{D}) = \sum_{j=1}^{c} \frac{\lvert d_j - \hat{d}_j \rvert}{d_j + \hat{d}_j}$
Kullback-Leibler ↓	$Dis_4(D,\hat{D}) = \sum_{j=1}^{c} d_j \ln \frac{d_j}{\hat{d}_j}$
Cosine ↑	$Sim_1(D,\hat{D}) = \frac{\sum_{j=1}^{c} d_j \hat{d}_j}{\sqrt{\sum_{j=1}^{c} {d_j}^2} \sqrt{\sum_{j=1}^{c} {\hat{d}_j}^2}}$
Intersection ↑	$Sim_2(D,\hat{D}) = \sum_{j=1}^{c} \min(d_j, \hat{d}_j)$

Métricas de evaluación especializadas.

$$\text{aRMSE} = \frac{1}{d}\sum_{i=1}^{d} \text{RMSE}$$

$$= \frac{1}{d}\sum_{i=1}^{d} \sqrt{\frac{\sum_{l=1}^{N_{test}} \left(y_i^{(l)} - \hat{y}_i^{(l)}\right)^2}{N_{test}}}$$

Métrica aRMSE, estándar de MTR.

Metodologías para resolver LDL

En la literatura especializada existen tres formas principales de abordar el problema LDL.

Transformación del problema	Adaptación de algoritmos	Algoritmos especializados
n instancias con ***c*** grados de descripción de LDL	Modelo para SLL	Algoritmos especializados y diseñados para el problema
n * c instancias para abordar el problema como uno de SLL	"Elimina" la restricción de una salida única (*función arg-max*)	Algoritmos que minimizan una función objetivo del problema LDL

Label Distribution Learning

Experimental Setup

Evaluation Measures

Name	Formula
Chebyshev(Cheby)↓	$Dis(D,\hat{D}) = max_j \lvert d_j - \hat{d}_j \rvert$
Clark↓	$Dis(D,\hat{D}) = \sqrt{\sum_{j=1}^{c} \frac{(d_j-\hat{d}_j)^2}{(d_j+\hat{d}_j)^2}}$
Canberra(Can)↓	$Dis(D,\hat{D}) = \sum_{j=1}^{c} \frac{\lvert d_j-\hat{d}_j \rvert^2}{d_j+\hat{d}_j}$
Kullback-Leibler(KL)↓	$Dis(D,\hat{D}) = \sum_{j=1}^{c} d_j \ln \frac{d_j}{\hat{d}_j}$
Cosine(Cos)↑	$Sim(D,\hat{D}) = \frac{\sum_{j=1}^{c} d_j \hat{d}_j}{\sqrt{\sum_{j=1}^{c} d_j^2}\sqrt{\sum_{j=1}^{c} \hat{d}_j^2}}$
Intersection(Inter)↑	$Sim(D,\hat{D}) = \sum_{j=1}^{c} min(d_j, \hat{d}_j)$

State-of-the-Art Algorithms

Algorithm	Parameter	Description	Value
k-NN	k	Number of selected neighbors	4
BFGS	ϵ	Convergence criterion: must be less than ϵ before successful termination	10^{-5}
StructRF	trees	Number of trees	50
	sampling	Sampling ratio of data	0.8
	max. depth	Maximum depth of the tree	20
	min. leaf	Minimum size of the leaf	5

Data Sets

No.	Datasets	Examples(n)	Features(q)	Labels(l)
1	Yeast_alpha	2465	24	18
2	Yeast_cdc	2465	24	15
3	Yeast_cold	2465	24	4
4	Yeast_diau	2465	24	7
5	Yeast_dtt	2465	24	4
6	Yeast_elu	2465	24	14
7	Yeast_heat	2465	24	6
8	Yeast_spo	2465	24	6
9	Yeast_spo5	2465	24	3
10	Yeast_spoem	2465	24	2
11	SJAFFE	213	243	6
12	SBU_3DFE	2500	243	6
13	Movie	7755	1869	5
14	Natural_Scene	2000	294	9
15	Human_Gene	30542	36	68

Label Distribution Learning

Proposal: Synthetic Sample Generation SSG-LDL

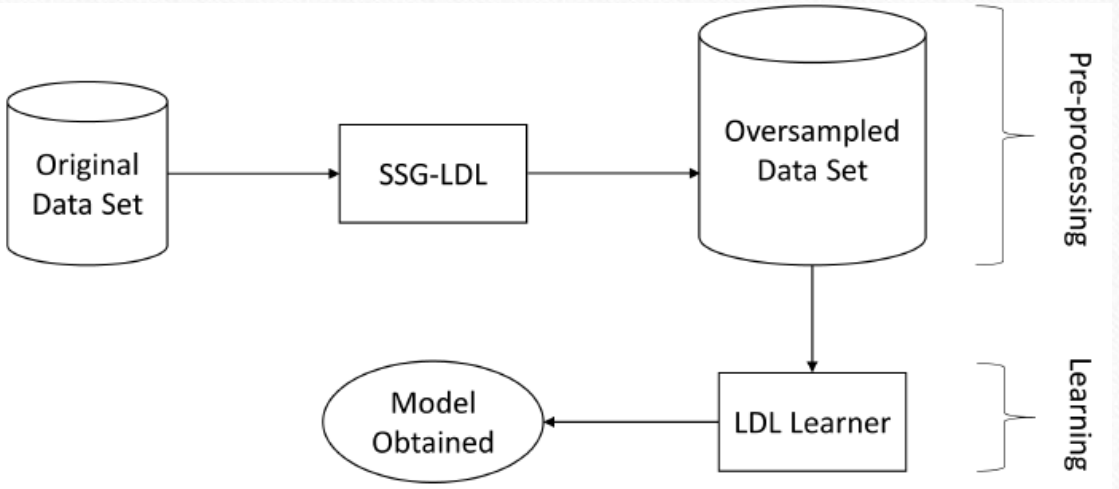
- Improve the performance of the underlying LDL learners without modifying them.

Algorithm 1: SSG-LDL

Function SSG-LDL($S[]$, N, k):

1 **input** : $S[]$ ← original training set ;
  N ← % of oversampling ;
  k ← considered neighbors ;
2 **output:** $S'[]$ ← the oversampled training set
3 $S' = S$;
4 $T = m * N/100$ ← nº synthetic samples to create;
5 **for** i=1 *to* T **do**
6 j = SelectSample(S) ;
7 ss = CreateSyntheticSample(S, j, k) ;
8 $S' \leftarrow ss$;
9 $i = i + 1$;
10 **end**

End Function



$$\overline{DIST}[x_i] = f_x \frac{\sum_{l=1}^{m} \text{euclidean}(x_i, x_l)}{m} + f_y \frac{\sum_{l=1}^{m} \text{euclidean}(D_i, D_l)}{m}.$$

Diverge Cumulative Selection

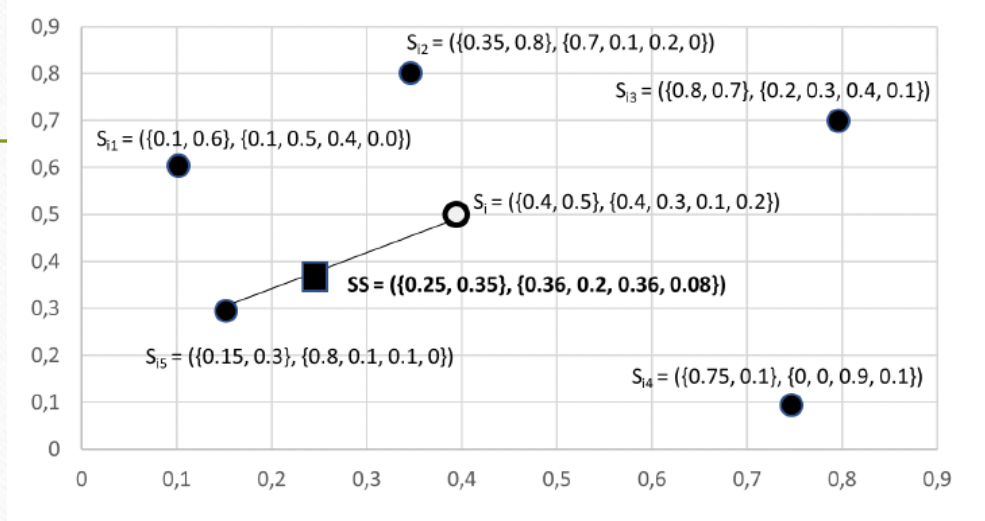
$$P_{x_i} = \frac{\overline{DIST}[x_i]}{\sum_{j=1}^{m} \overline{DIST}[x_j]}.$$

Synthetic features generation: Inputs → Interpolation, Outputs → mean of the k nearest neighbors.

Label Distribution Learning

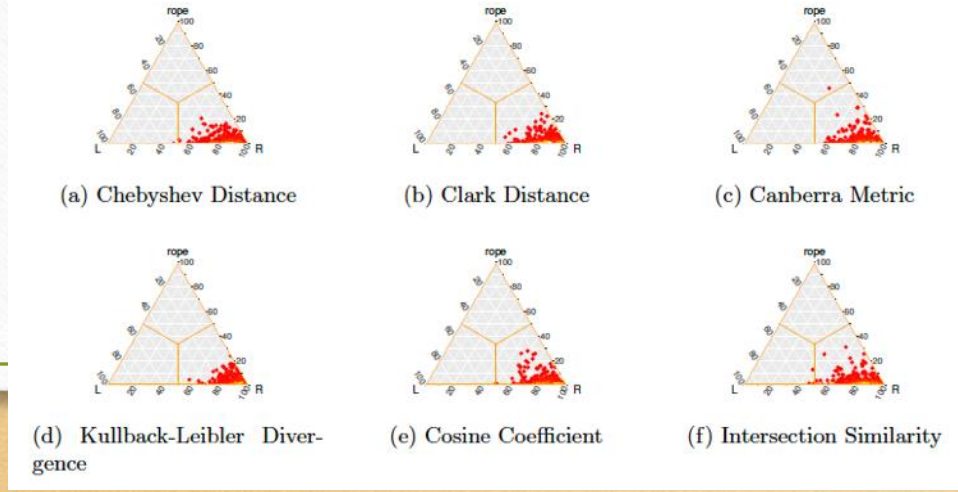
Proposal: Synthetic Sample Generation SSG-LDL

- Example of creation of a synthetic data point

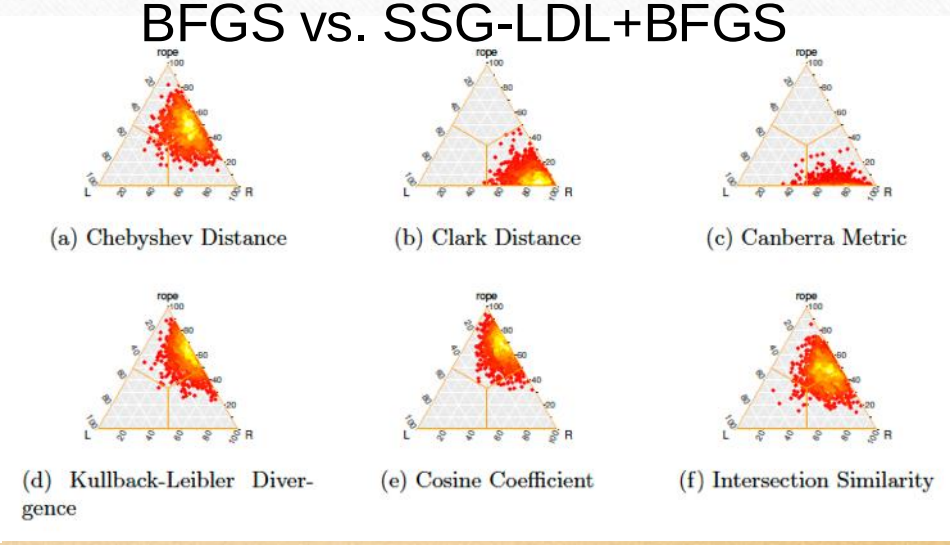


- Results (Bayesian Wilcoxon test)

kNN vs. SSG-LDL+kNN



BFGS vs. SSG-LDL+BFGS



APRENDIZAJE MULTI-INSTANCIA Y MULTI-ETIQUETA

Minería de Datos: Aspectos Avanzados

Salvador García (salvagl@decsai.ugr.es)

Alberto Fernández (alberto@decsai.ugr.es)