Taller #9

Business Intelligence

Carlos Reveco
creveco@dcc.uchile.cl

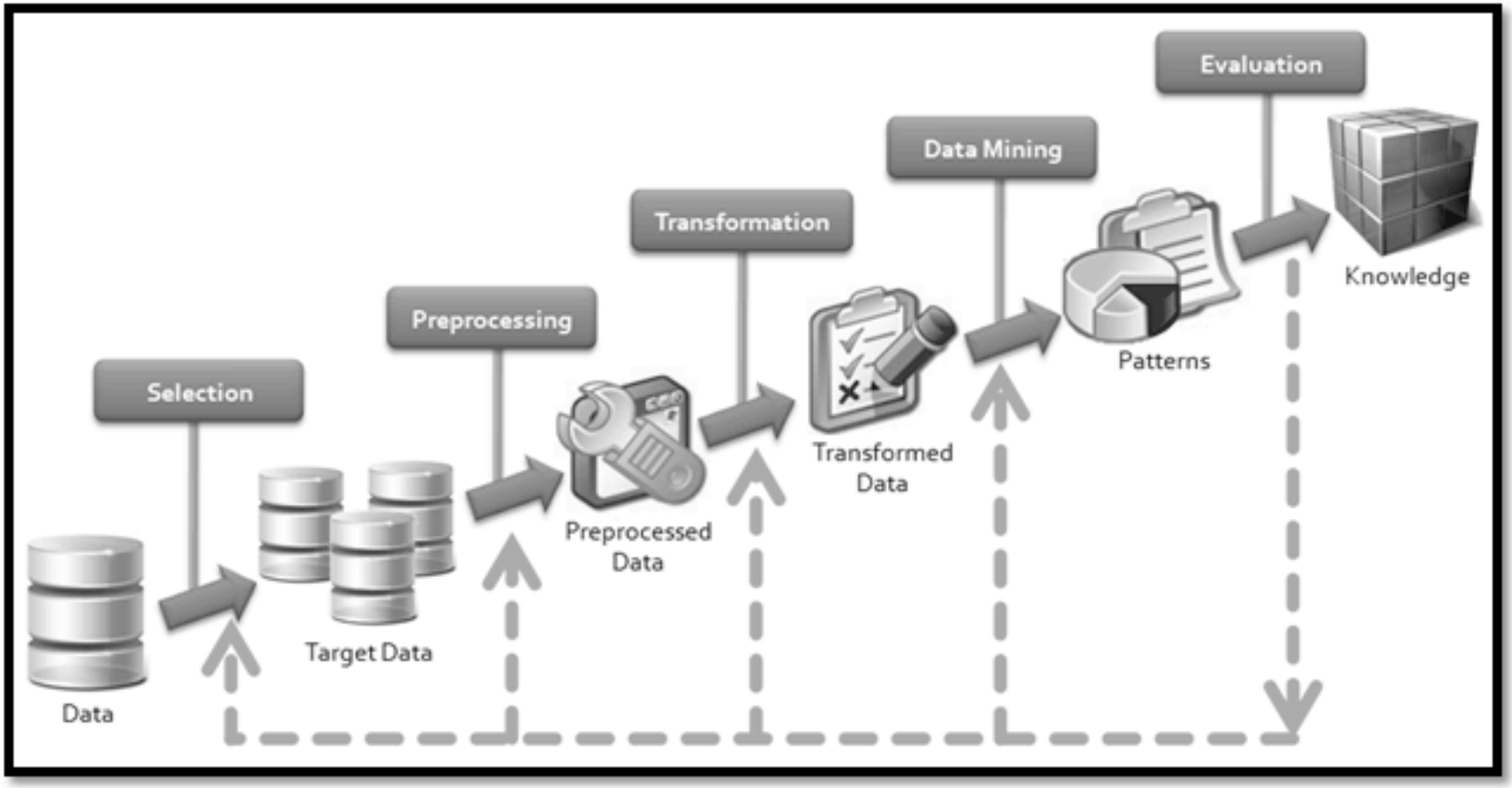
Cinthya Vergara
cvergarasilv@ing.uchile.cl

Agenda

- Taller #9
  - Múltiples Clasificadores
    - Motivación
    - Tipos de Múltiples Clasificadores
    - Múltiples Clasificadores basados en muestreo
  - Bonus Track
    - K vecinos más cercanos
    - Naive Bayes
    - Evaluación de modelos con curvas ROC y área bajo la curva ROC
  - Taller práctico Internet-Ads

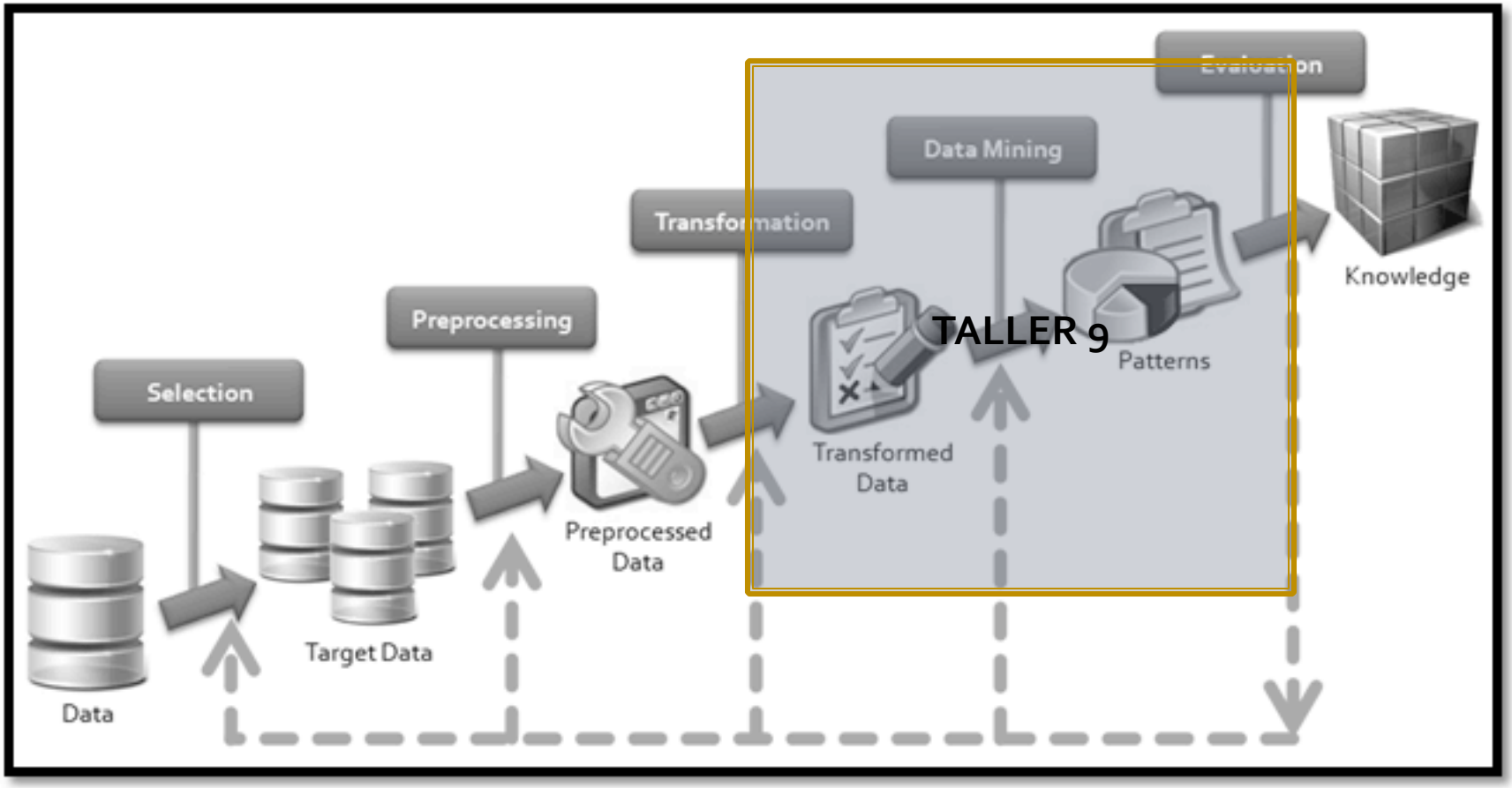
Proceso KDD

Proceso KDD



Knowledge **D**iscovery in **D**atabases → **KDD**

Proceso KDD



Knowledge **D**iscovery in **D**atabases → **KDD**

Múltiples Clasificadores

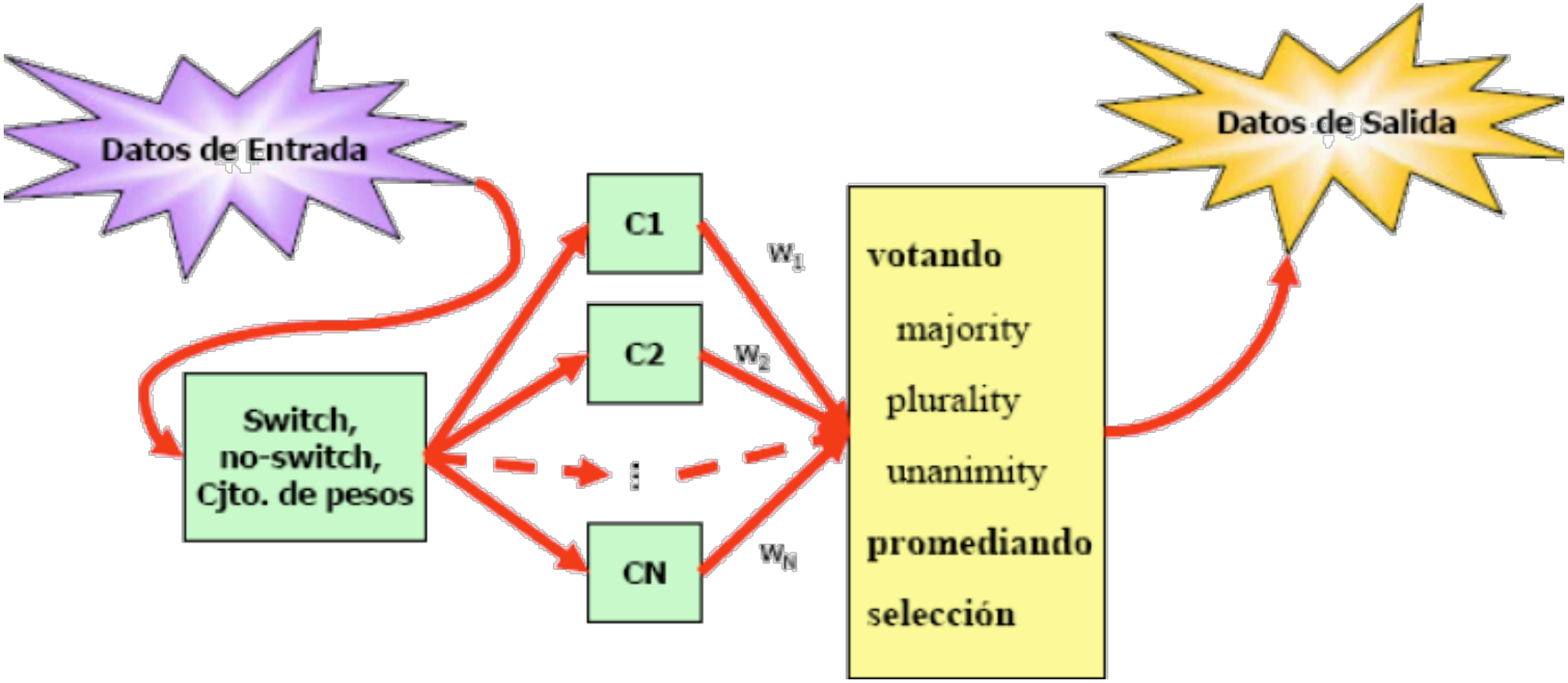
Descripción del problema

- **Problema**: *Existen muchas formas de hacer Clasificaciones y Regresiones y sabemos que cada uno de ellos puede entregarnos información importante para cada problema.*

  - En general la idea es no quedarnos con una sola técnica, aunque sea la **mejor técnica** (que nos entregue los mejores resultados de acuerdo a una instancia del problema), o la **más efectiva** (presentando la oportunidad de tener un modelo robusto ante nuevos elementos a clasificar).
  - Concepto de **Wisdom of the Crouds** o Paneles de expertos
  - **Idea 1:** Explotar las características de cada método de manera independiente para obtener mejores resultados.
  - **Idea 2:** Combinar los resultados de cada método de la manera adecuada.

Descripción del problema

- Problema: Tenemos muchas formas de hacer Clasificaciones y Regresiones.



Modelos para construcción de MC

1. **Subsampling** de las observaciones de entrenamiento mediante técnicas de resampling (boosting, bagging).

2. Manipulación de la **selección de atributos** para entrenar distintos modelos con conjuntos de atributos distintos.

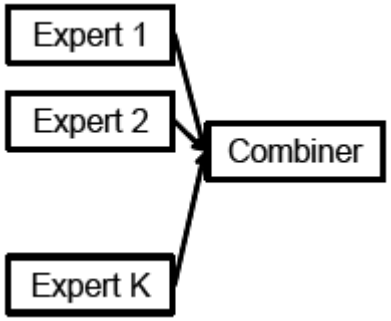
3. **Manipulación de la variable dependiente** de manera de buscar solución a problemas representados por distintos valores objetivos.

4. **Modificación de los parámetros** del clasificador de manera de obtener distintos modelos asociados a un conjunto de entrenamiento dado.

5. **Diversificación de modelos** a utilizar para determinar los valores asociados a la solución del problema objetivo.

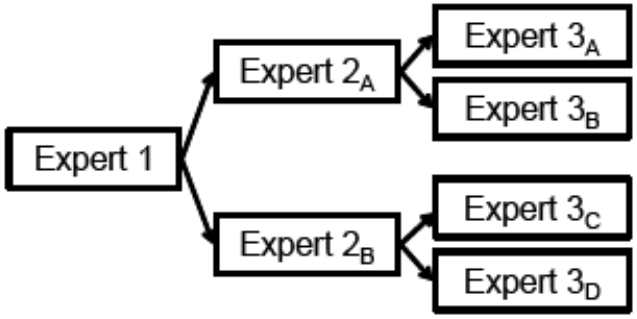
Estructuras de Múltiples Clasificadores [1]

- **Paralelos**



- **En cascada**

- **Jerárquicos**



Estructuras de Múltiples Clasificadores [2]

- **Paralelos:** Todos los clasificadores son invocados independientemente y sus resultados son combinados a través de algún criterio adecuado.
- **En cascada:** Clasificadores son llamados de manera secuencial.
- **Jerárquicos:** Clasificadores son llamados a través de una estructura "de árbol" definida por la jerarquía.

Estrategias de Combinación

- **Estáticos**
  - **No entrenables:** La votación se realiza de manera independiente y no paramétrica el desempeño de cada clasificador. (Majority voting, promedio de resultados, etc.).
  - **Entrenables o Ajustables:** El "combinador" inicia una nueva fase de entrenamiento de manera de mejorar el desempeño general de los modelos generados. (nuevos modelos, MIP, etc.)
- **Adaptativos**
  - La función que define al "combinador" depende de los atributos iniciales considerados para los distintos modelos.

Subsampling el conjunto de entrenamiento [1]

- **Bagging**
  - **(Bootstrap aggregation)** crea un multiclasificador entrenando modelos individuales en conjuntos de muestras que provienen de la técnica de resampling "**bootstrap**" como conjunto de entrenamiento.
  - La idea principal es que dado un conjunto de entrenamiento con N observaciones, Bagging genera m conjuntos Cj, j = {1, …, m},
    donde |Cj| = n <= N. Estos conjuntos los genera utilizando una selección uniforme con reemplazo
  - Con cada uno de los conjunto Cj, se entrena un modelo, obteniendo de esta manera m modelos ajustados con los conjuntos Cj, j={1,…,m}
  - Los resultados considerados para Bagging se pueden promediar (para el caso de las regresiones) o se puede utilizar un Majority Voting (para el caso de clasificación).

Subsampling el conjunto de entrenamiento [2]

- **Boosting**
  - Utiliza una técnica de resampling distinta a bagging, la cual mantiene una probabilidad constante de 1/N para la selección de cada observación.
  - Se va actualizando la probabilidad utilizada en el resampling en el tiempo, basado en el desempeño del modelo.
  - Basado en el concepto de "clasificador débil", donde basta un modelo que entregue resultados un poco mejor que al azar. (mayor a una clasificación de un 50%)
  - Existe una gran variante de técnicas de boosting, uno de los ejemplos más utilizados es el algoritmo Adaboost.

Subsampling el conjunto de entrenamiento [3]

- **AdaBoost**
  - (Adaptive boosting) Permite al experimentador ir agregando nuevas componentes de clasificación al modelo a medida que se va logrando un error más pequeño.
  - Opera de la siguiente manera:
  - 1.- En la iteración n-ésima se provee al "clasificador débil" con una distribución Dn(i) que representa la probabilidad de seleccionar la observación i-ésima.
  - 2.- Se genera el modelo en base a la base de datos de entrenamiento formada por la distribución Dn(i), midiendo la tasa de error en base a un test de hipótesis Hn con respecto a Dn(i)
  - 3.- Se genera Dn+1(i) , bajando la probabilidad de seleccionar aquellas observaciones que fueron bien clasificadas en la iteración n, e incrementando la probabilidad de aquellas mal clasificadas.

Random Forest [1]

- Random Forest: Consiste en un conjunto de árboles de decisión, donde la predicción de la clase corresponde a la moda de la predicción de los árboles de decisión individuales.

- Combina la idea de *bagging* con la selección aleatoria de atributos.

- Ventajas:
  - Buenos resultados en clasificación
  - Puede manejar gran cantidad de atributos de entrada
  - Aprendizaje rápido
  - Robusto frente a valores missing

Random Forest [2]

- Random Forest
  - Cada árbol se construye en base al siguiente proceso:
    1. Sea N el total de observaciones y M el total de atributos, se seleccionan aleatoriamente m (m<M) atributos.
    2. Cada conjunto de entrenamiento se construye seleccionando n muestras con reemplazo de las N posibles observaciones.
    3. Se obtienen los mejores árboles para cada los conjuntos de m variables seleccionadas. Cada árbol es construido sin poda de la forma más simple posible.

Bonus track: K-NN, Naive Bayes y curvas ROC

K vecinos más cercanos (K-NN)

- Características
  - Modelo de tipo "Lazy Learning"
  - Es el modelo más simple para clasificación.

Algorithm 1: k-NN

Data: $\mathcal{T} = \{\mathcal{X}, \mathcal{Y}\}, k$
Result: $\{H_{\mathcal{T}_e} = h, \forall \mathbf{x} \in \mathcal{T}_e\}$
Initialize D = Matriz de distancias de objetos en T_r ;
foreach $(\mathbf{x}, y) \in \mathcal{T}_e$ **do**
    $K \leftarrow \{k$ vecinos más cercanos a $\mathbf{x}\}$;
    h = El "y" más común entre los objetos del conjunto K;
return $\{h\}, \forall \mathbf{x} \in \mathcal{T}_e$;

Naive Bayes

- Características
  - Modelo generativo
  - Modela la probabilidad condicional p(x|y) para posteriormente completar la hipotesis con el teorema de Bayes.

$$P(y|\mathbf{x}) = \frac{P(y) \cdot P(\mathbf{x}|y)}{P(\mathbf{x})} = \frac{P(y)}{P(\mathbf{x})} \cdot \prod_{j=1}^{A} P(\mathbf{x}_j|y) \quad (4)$$

$$ln\left(\frac{P(y=+1|\mathbf{x})}{P(y=-1|\mathbf{x})}\right) = ln\left(\frac{P(y=+1)}{P(y=-1)}\right) + \sum_{j=1}^{A} ln\left(\frac{P(\mathbf{x}_{\tau j}|y=+1)}{P(\mathbf{x}_j|y=-1)}\right) \quad (5)$$

$$h(\mathbf{x}) = \begin{cases} +1 & \text{if } ln\left(\frac{P(y=+1|\mathbf{x})}{P(y=-1|\mathbf{x})}\right) > \theta \\ -1 & Otherwise \end{cases} \quad (6)$$

Curvas ROC [1]

Hypothesized class \ True class	p	n
Y	True Positives	False Positives
N	False Negatives	True Negatives

$$\text{fp rate} = \frac{FP}{N} \qquad \text{tp rate} = \frac{TP}{P}$$

$$\text{precision} = \frac{TP}{TP+FP} \qquad \text{recall} = \frac{TP}{P}$$

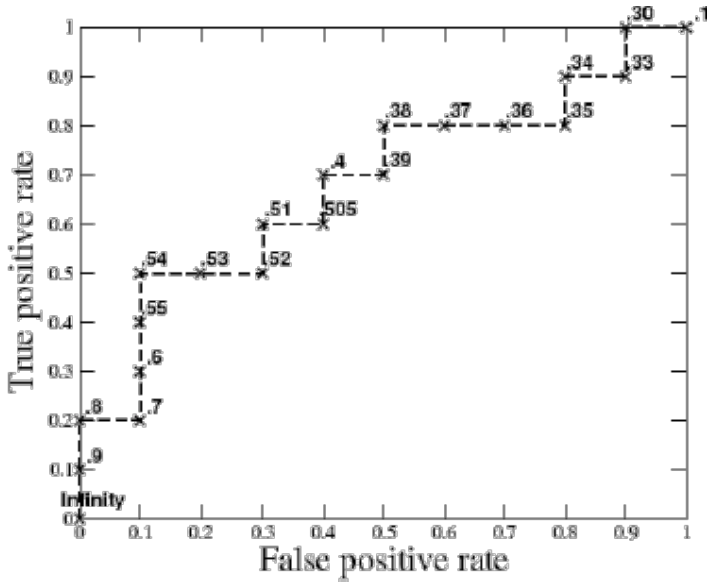
$$\text{accuracy} = \frac{TP+TN}{P+N} \qquad \text{F-measure} = \frac{2}{1/\text{precision}+1/\text{recall}}$$

Curvas ROC [2]

- Curvas ROC (Receiver Operating Characteristic)
  - Curva determinada por la función paramétrica:

$$x = FPrate(t), \quad y = TPrate(t).$$

Inst#	Class	Score	Inst#	Class	Score
1	p	.9	11	p	.4
2	p	.8	12	n	.39
3	n	.7	13	p	.38
4	p	.6	14	p	.37
5	p	.55	15	n	.36
6	p	.54	16	n	.35
7	n	.53	17	p	.34
8	n	.52	18	n	.33
9	p	.51	19	p	.30
10	n	.505	10	n	.1



Curvas ROC [3]

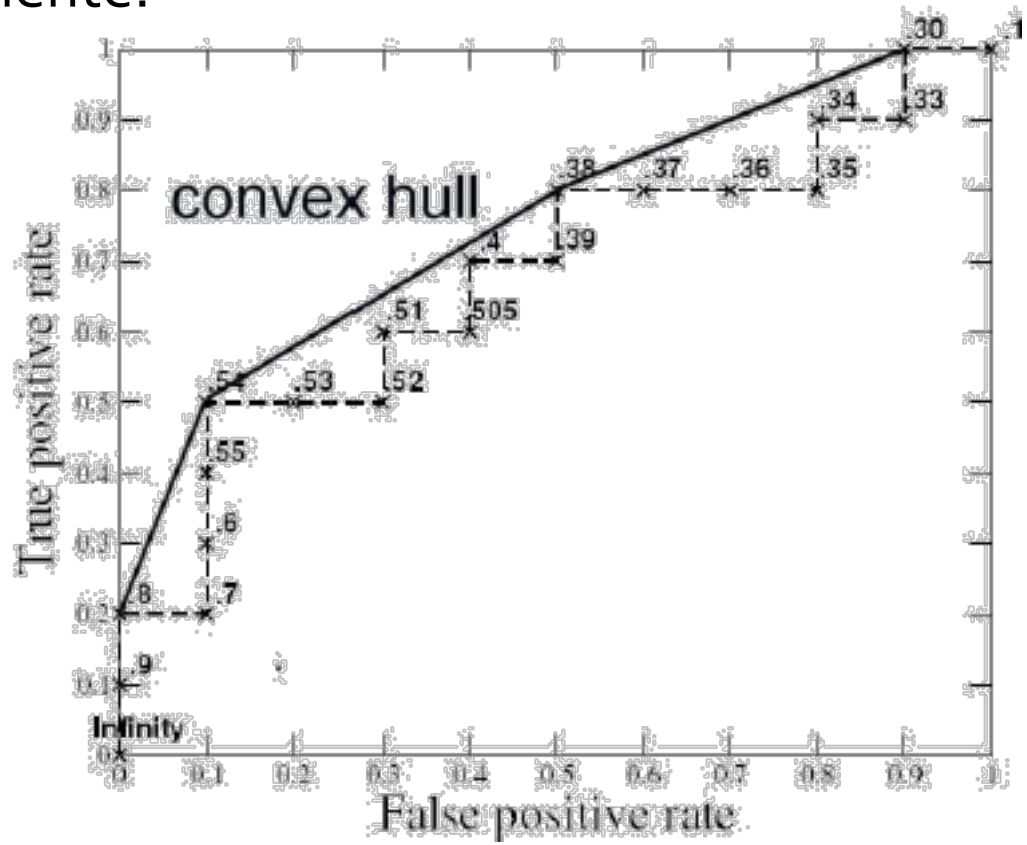
Algorithm 1 Conceptual method for calculating an ROC curve. See algorithm 2 for a practical method.

Inputs: L, the set of test instances; $f(i)$, the probabilistic classifier's estimate that instance i is positive; *min* and *max*, the smallest and largest values returned by f; *increment*, the smallest difference between any two f values.

```
for t = min to max by increment do
    FP ← 0
    TP ← 0
    for i ∈ L do
        if f(i) ≥ t then                    /* This example is over threshold */
            if i is a positive example then
                TP ← TP + 1
            else        /* i is a negative example, so this is a false positive */
                FP ← FP + 1
            end if
        end if
    end for
    Add point (FP/N, TP/P) to ROC curve
end for
end
```

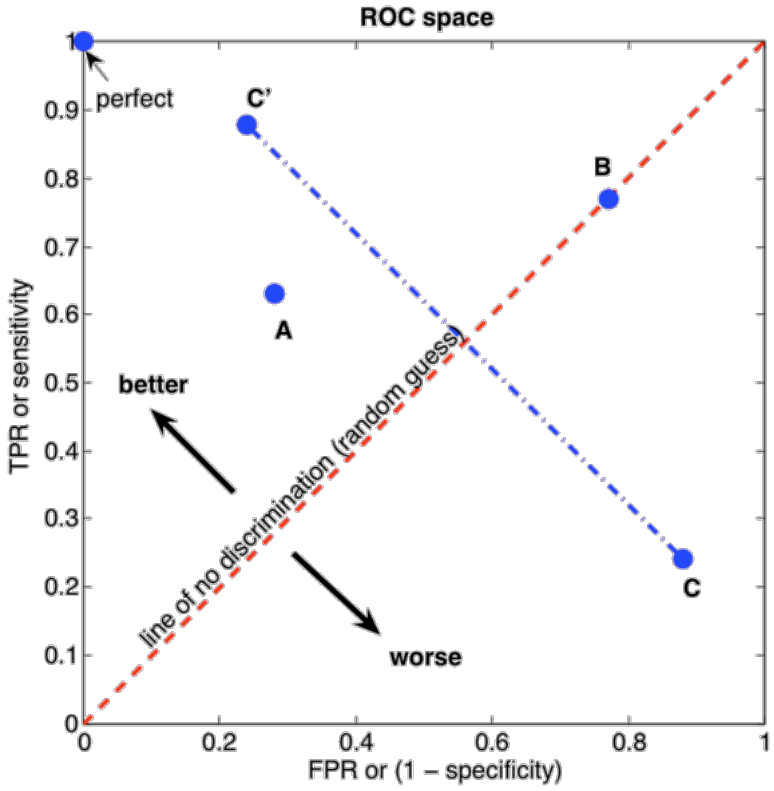
Curvas ROC [4]

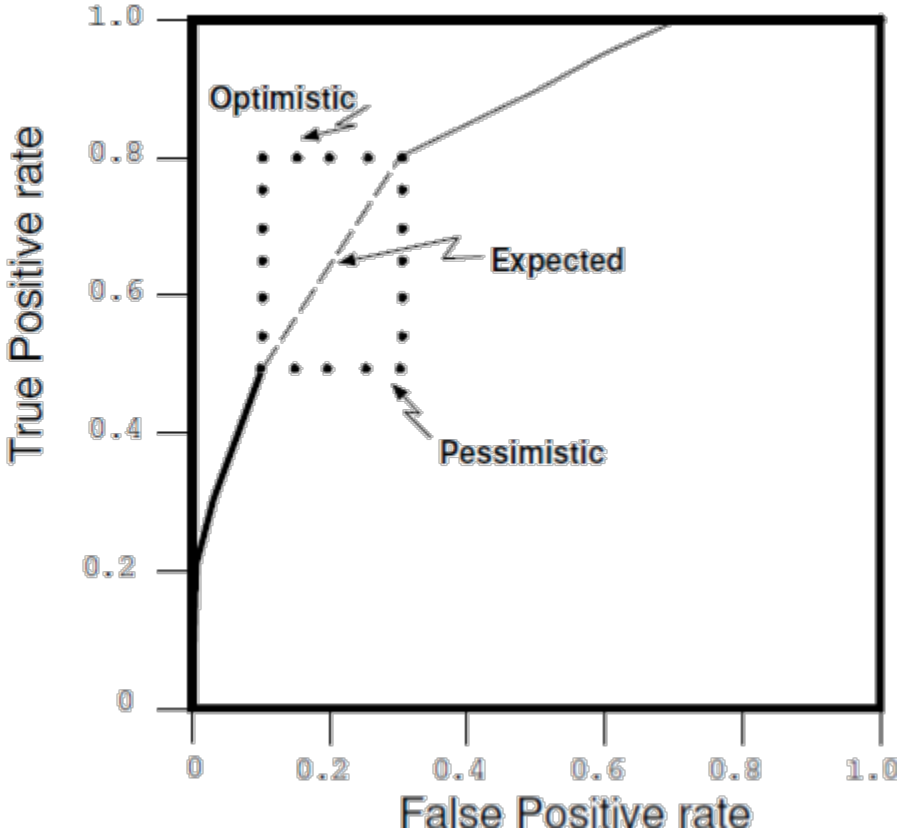
- Se construye la envoltura convexa de los puntos evaluados anteriormente:



Curvas ROC [5]

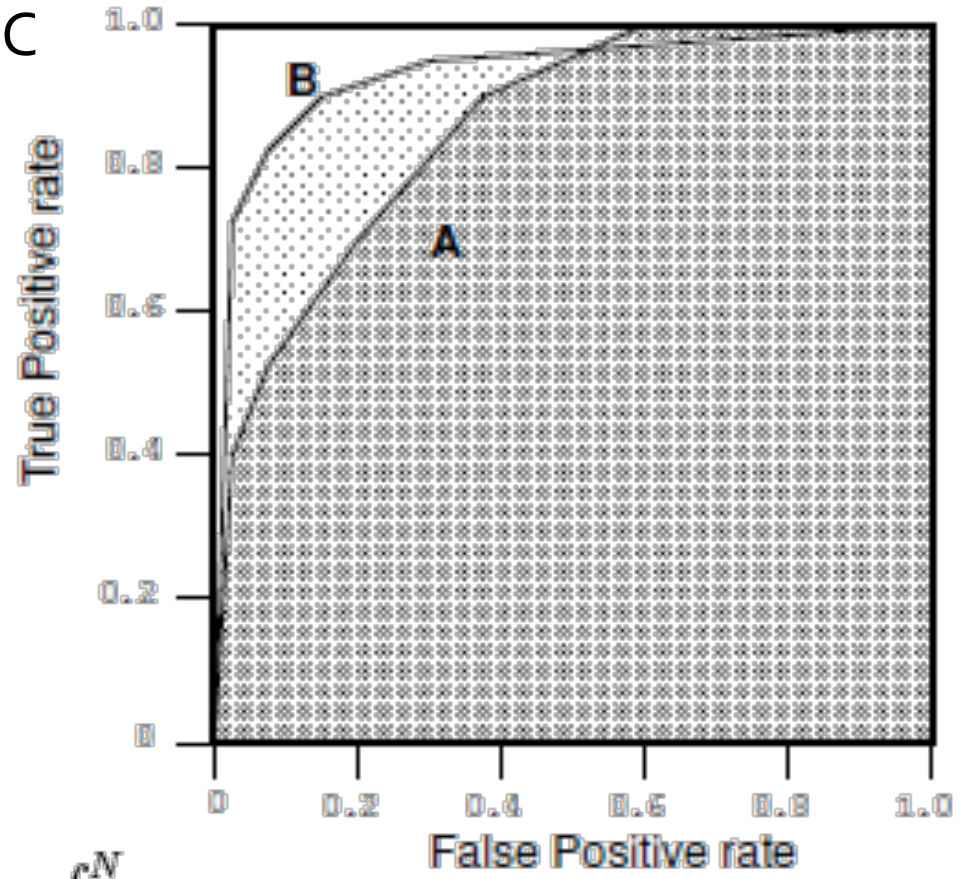
- ¿Como se analizan las Curvas ROC?





Curvas ROC [6]

- Área bajo la curva ROC



$$A_{ROC} = \int_0^1 \frac{TP}{P} \quad d\frac{FP}{N} = \frac{1}{P\,N} \int_0^N TP \;\; dFP$$

Taller #9

Business Intelligence

Carlos Reveco
creveco@dcc.uchile.cl

Cinthya Vergara
cvergarasilv@ing.uchile.cl