

# Aprendizaje semi-supervisado (SSL)

Teoría y casos prácticos de SSL

José Manuel Cuadra Troncoso

# Índice

- 1 Introducción a SSL
- 2 Herramientas para SSL
- 3 Algoritmos para SSL

# Índice

## 1 Introducción a SSL

- Entre el aprendizaje no supervisado y el supervisado
- Datos etiquetados
- Aprendizaje semi-supervisado y transductivo
- ¿Cuándo puede funcionar el SSL?

## 2 Herramientas para SSL

## 3 Algoritmos para SSL

# Índice

## 1 Introducción a SSL

- Entre el aprendizaje no supervisado y el supervisado
- Datos etiquetados
- Aprendizaje semi-supervisado y transductivo
- ¿Cuándo puede funcionar el SSL?

## 2 Herramientas para SSL

## 3 Algoritmos para SSL

# Aprendizaje no supervisado

- El objetivo del aprendizaje no supervisado es encontrar una estructura interesante en un conjunto de ejemplos  $X = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$  extraídas de una población  $\mathcal{X}$ .
  - Más precisamente encontrar una f.d.p que pueda haber generado  $X$ .
  - Este objetivo se extiende a estimación de cuantiles, clustering, detección de outliers y reducción de dimensionalidad.

# Aprendizaje no supervisado

- El objetivo del aprendizaje no supervisado es encontrar una estructura interesante en un conjunto de ejemplos  $X = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$  extraídas de una población  $\mathcal{X}$ .
  - Más precisamente encontrar una f.d.p que pueda haber generado  $X$ .
  - Este objetivo se extiende a estimación de cuantiles, clustering, detección de outliers y reducción de dimensionalidad.

# Aprendizaje no supervisado

- El objetivo del aprendizaje no supervisado es encontrar una estructura interesante en un conjunto de ejemplos  $X = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$  extraídas de una población  $\mathcal{X}$ .
  - Más precisamente encontrar una f.d.p que pueda haber generado  $X$ .
  - Este objetivo se extiende a estimación de cuantiles, clustering, detección de outliers y reducción de dimensionalidad.

# Aprendizaje supervisado

- El objetivo del aprendizaje supervisado es aprender un mapeo  $f$  de  $X$  en  $Y = \{y_1, y_2, \dots, y_n\}$ , donde  $y_i$  son las etiquetas asociadas a los ejemplos  $\mathbf{x}_i$  extraídas de una población  $\mathcal{Y}$ .
  - Los pares  $(\mathbf{x}_i, y_i)$  se supone que se seleccionan de manera i.i.d. en la población  $\mathcal{X} \times \mathcal{Y}$ .
  - Este objetivo se extiende a regresión, detección de outliers y de novedad.



# Aprendizaje supervisado

- El objetivo del aprendizaje supervisado es aprender un mapeo  $f$  de  $X$  en  $Y = \{y_1, y_2, \dots, y_n\}$ , donde  $y_i$  son las etiquetas asociadas a los ejemplos  $\mathbf{x}_i$  extraídas de una población  $\mathcal{Y}$ .
  - Los pares  $(\mathbf{x}_i, y_i)$  se supone que se seleccionan de manera i.i.d. en la población  $\mathcal{X} \times \mathcal{Y}$ .
  - Este objetivo se extiende a regresión, detección de outliers y de novedad.

# Aprendizaje supervisado

- El objetivo del aprendizaje supervisado es aprender un mapeo  $f$  de  $X$  en  $Y = \{y_1, y_2, \dots, y_n\}$ , donde  $y_i$  son las etiquetas asociadas a los ejemplos  $\mathbf{x}_i$  extraídas de una población  $\mathcal{Y}$ .
  - Los pares  $(\mathbf{x}_i, y_i)$  se supone que se seleccionan de manera i.i.d. en la población  $\mathcal{X} \times \mathcal{Y}$ .
  - Este objetivo se extiende a regresión, detección de outliers y de novedad.

# Índice

## 1 Introducción a SSL

- Entre el aprendizaje no supervisado y el supervisado

### ■ Datos etiquetados

- Aprendizaje semi-supervisado y transductivo

- ¿Cuándo puede funcionar el SSL?

## 2 Herramientas para SSL

## 3 Algoritmos para SSL

# Pros y contras de los datos etiquetados

## Pros

- Con relativamente pocos datos se puede entrenar.

# Pros y contras de los datos etiquetados

## Pros

- Con relativamente pocos datos se puede entrenar.

## Contras

- Las etiquetas suelen necesitar expertos anotadores, llevan tiempo y dinero.
- Las etiquetas pueden necesitar dispositivos especializados.

# Pros y contras de los datos etiquetados

## Pros

- Con relativamente pocos datos se puede entrenar.

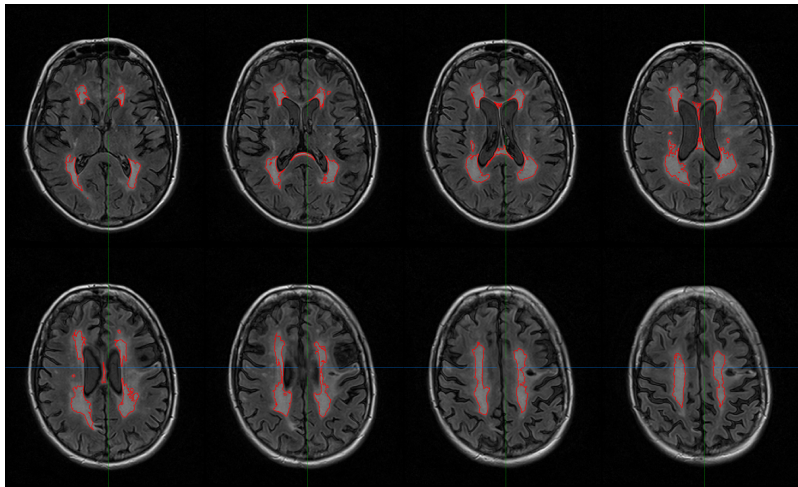
## Contras

- Las etiquetas suelen necesitar expertos anotadores, llevan tiempo y dinero.
- Las etiquetas pueden necesitar dispositivos especializados.

Los datos no etiquetados suelen presentar características opuestas.

# Ejemplo de obtención de etiquetas

Anotación de imágenes médicas



# Uso de juegos para etiquetado

- Juegos de computación basados en humanos o juegos con un propósito (GWAP).
- Para abordar problemas que las computadoras aún no pueden:
  - Etiquetado de imágenes.
  - Anotación de obras de arte, literatura...
  - Genética.
  - Web semántica ...
- [https://en.wikipedia.org/wiki/Human-based\\_computation\\_game](https://en.wikipedia.org/wiki/Human-based_computation_game)



# Índice

## 1 Introducción a SSL

- Entre el aprendizaje no supervisado y el supervisado
- Datos etiquetados
- **Aprendizaje semi-supervisado y transductivo**
- ¿Cuándo puede funcionar el SSL?

## 2 Herramientas para SSL

## 3 Algoritmos para SSL

# Aprendizaje semi-supervisado

## ■ Clasificación semi-supervisada:

- Usar  $l$  datos etiquetados  $\{(\mathbf{x}_i, y_i)\}_{i=1}^l$  y  $u$  no etiquetados  $\{\mathbf{x}_j\}_{j=l+1}^{l+u}$ . Normalmente  $u \gg l$ .
- Objetivo: obtener un mejor clasificador que con datos etiquetados solo. Extensión de SL.
- Hay métodos generativos (probabilísticos) y discriminativos (no probabilísticos).

## ■ Clustering con restricciones:

- Usar  $n$  datos no etiquetados  $\{\mathbf{x}_j\}_{j=1}^n$  y restricciones p. ej. dos puntos están en el mismo cluster (must-links) o no (cannot-links).
- Objetivo: obtener un mejor agrupamiento que con datos no etiquetados solo. Extensión de USL.

# Aprendizaje semi-supervisado

## ■ Clasificación semi-supervisada:

- Usar  $l$  datos etiquetados  $\{(\mathbf{x}_i, y_i)\}_{i=1}^l$  y  $u$  no etiquetados  $\{\mathbf{x}_j\}_{j=l+1}^{l+u}$ . Normalmente  $u \gg l$ .
- Objetivo: obtener un mejor clasificador que con datos etiquetados solo. Extensión de SL.
- Hay métodos generativos (probabilísticos) y discriminativos (no probabilísticos).

## ■ Clustering con restricciones:

- Usar  $n$  datos no etiquetados  $\{\mathbf{x}_j\}_{j=1}^n$  y restricciones p. ej. dos puntos están en el mismo cluster (must-links) o no (cannot-links).
- Objetivo: obtener un mejor agrupamiento que con datos no etiquetados solo. Extensión de USL.

# Aprendizaje semi-supervisado

## ■ Clasificación semi-supervisada:

- Usar  $l$  datos etiquetados  $\{(\mathbf{x}_i, y_i)\}_{i=1}^l$  y  $u$  no etiquetados  $\{\mathbf{x}_j\}_{j=l+1}^{l+u}$ . Normalmente  $u \gg l$ .
- Objetivo: obtener un mejor clasificador que con datos etiquetados solo. Extensión de SL.
- Hay métodos generativos (probabilísticos) y discriminativos (no probabilísticos).

## ■ Clustering con restricciones:

- Usar  $n$  datos no etiquetados  $\{\mathbf{x}_j\}_{j=1}^n$  y restricciones p. ej. dos puntos están en el mismo cluster (must-links) o no (cannot-links).
- Objetivo: obtener un mejor agrupamiento que con datos no etiquetados solo. Extensión de USL.

# SSL vs aprendizaje transductivo

## SSL inductivo

Dados  $\{(\mathbf{x}_i, y_i)\}_{i=1}^l$  y  $\{\mathbf{x}_j\}_{j=l+1}^{l+u}$  aprender  $f : \mathcal{X} \rightarrow \mathcal{Y}$ , se espera que  $f$  realice buenas **predicciones sobre datos futuros**.

## Aprendizaje transductivo

Dados  $\{(\mathbf{x}_i, y_i)\}_{i=1}^l$  y  $\{\mathbf{x}_j\}_{j=l+1}^{l+u}$  aprender  $f : X^{l+u} \rightarrow Y^{l+u}$ , se espera que  $f$  realice buenas **predicciones solo en la muestra de entrenamiento**.

# Índice

## 1 Introducción a SSL

- Entre el aprendizaje no supervisado y el supervisado
- Datos etiquetados
- Aprendizaje semi-supervisado y transductivo
- ¿Cuándo puede funcionar el SSL?

## 2 Herramientas para SSL

## 3 Algoritmos para SSL

# ¿Cuándo puede funcionar el SSL?

## ¿Tiene sentido el SSL?

Precisando: en comparación con un algoritmo supervisado ¿se puede esperar mayor precisión empleando datos sin etiquetar?

- Sí, pero la distribución de ejemplo debe ser relevante para el problema de clasificación planteado.
  - El conocimiento sobre  $p(\mathbf{x})$  que aportan los datos no etiquetados debe tener información útil sobre  $p(\mathbf{y}|\mathbf{x})$ .
- Para esto deben cumplirse ciertas hipótesis: suavidad/continuidad, agrupamiento y de variedad(manifold).

# ¿Cuándo puede funcionar el SSL?

## ¿Tiene sentido el SSL?

Precisando: en comparación con un algoritmo supervisado ¿se puede esperar mayor precisión empleando datos sin etiquetar?

- Sí, pero la distribución de ejemplo debe ser relevante para el problema de clasificación planteado.
  - El conocimiento sobre  $p(\mathbf{x})$  que aportan los datos no etiquetados debe tener información útil sobre  $p(y|\mathbf{x})$ .
- Para esto deben cumplirse ciertas hipótesis: suavidad/continuidad, agrupamiento y de variedad(manifold).



# ¿Cuándo puede funcionar el SSL?

## ¿Tiene sentido el SSL?

Precisando: en comparación con un algoritmo supervisado ¿se puede esperar mayor precisión empleando datos sin etiquetar?

- Sí, pero la distribución de ejemplo debe ser relevante para el problema de clasificación planteado.
  - El conocimiento sobre  $p(\mathbf{x})$  que aportan los datos no etiquetados debe tener información útil sobre  $p(y|\mathbf{x})$ .
- Para esto deben cumplirse ciertas hipótesis: suavidad/continuidad, agrupamiento y de variedad(manifold).

# ¿Cuándo puede funcionar el SSL?

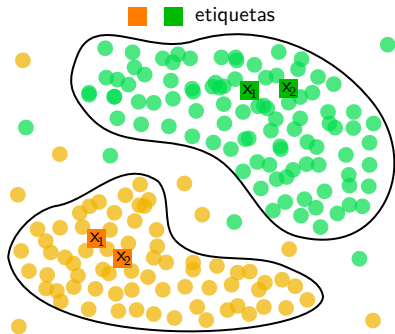
## ¿Tiene sentido el SSL?

Precisando: en comparación con un algoritmo supervisado ¿se puede esperar mayor precisión empleando datos sin etiquetar?

- Sí, pero la distribución de ejemplo debe ser relevante para el problema de clasificación planteado.
  - El conocimiento sobre  $p(\mathbf{x})$  que aportan los datos no etiquetados debe tener información útil sobre  $p(y|\mathbf{x})$ .
- Para esto deben cumplirse ciertas hipótesis: suavidad/continuidad, agrupamiento y de variedad(manifold).

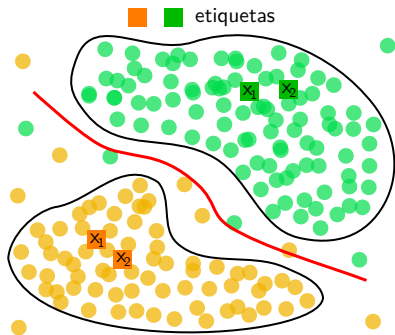
# Hipótesis de suavidad/continuidad semi-supervisada

- Si dos puntos  $\mathbf{x}_1, \mathbf{x}_2$  en una región de alta densidad están cerca, sus etiquetas  $y_1, y_2$  deberían ser iguales.
- La frontera de decisión en región de baja densidad.
- Generalizar a un posible conjunto infinito de casos a partir de un conjunto finito de entrenamiento.



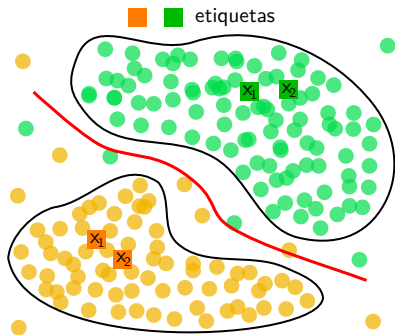
# Hipótesis de suavidad/continuidad semi-supervisada

- Si dos puntos  $\mathbf{x}_1, \mathbf{x}_2$  en una región de alta densidad están cerca, sus etiquetas  $y_1, y_2$  deberían ser iguales.
- La frontera de decisión en región de baja densidad.
- Generalizar a un posible conjunto infinito de casos a partir de un conjunto finito de entrenamiento.



# Hipótesis de suavidad/continuidad semi-supervisada

- Si dos puntos  $\mathbf{x}_1, \mathbf{x}_2$  en una región de alta densidad están cerca, sus etiquetas  $y_1, y_2$  deberían ser iguales.
- La frontera de decisión en región de baja densidad.
- Generalizar a un posible conjunto infinito de casos a partir de un conjunto finito de entrenamiento.

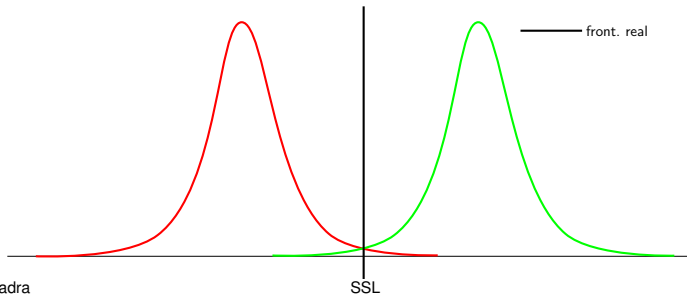


# Hipótesis de agrupamiento

- Si dos puntos  $\mathbf{x}_1$ ,  $\mathbf{x}_2$  pertenecen al mismo cluster, sus etiquetas  $y_1$ ,  $y_2$  deberían ser iguales. Aunque datos con la misma etiqueta pueden pertenecer a distintos clusters. Caso particular de la hipótesis de suavidad.
- Los datos no etiquetados pueden ayudar a precisar las fronteras de los clusters.

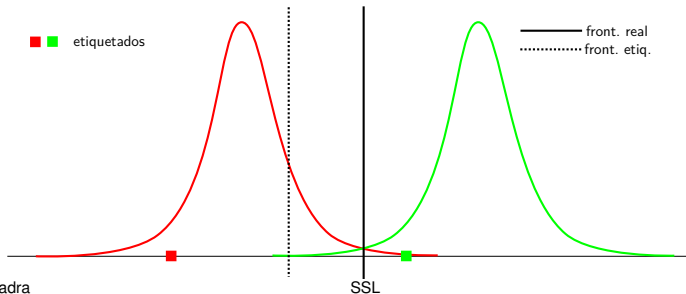
# Hipótesis de agrupamiento

- Si dos puntos  $\mathbf{x}_1, \mathbf{x}_2$  pertenecen al mismo cluster, sus etiquetas  $y_1, y_2$  deberían ser iguales. Aunque datos con la misma etiqueta pueden pertenecer a distintos clusters. Caso particular de la hipótesis de suavidad.
- Los datos no etiquetados pueden ayudar a precisar las fronteras de los clusters.



# Hipótesis de agrupamiento

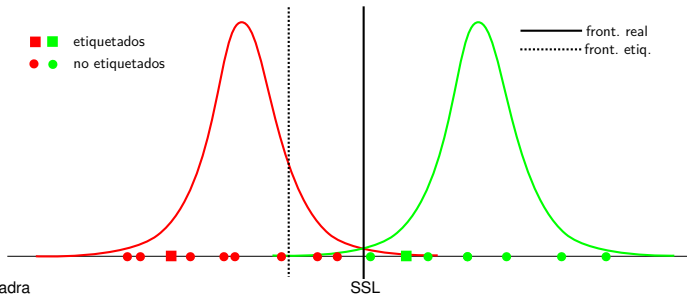
- Si dos puntos  $\mathbf{x}_1, \mathbf{x}_2$  pertenecen al mismo cluster, sus etiquetas  $y_1, y_2$  deberían ser iguales. Aunque datos con la misma etiqueta pueden pertenecer a distintos clusters. Caso particular de la hipótesis de suavidad.
- Los datos no etiquetados pueden ayudar a precisar las fronteras de los clusters.





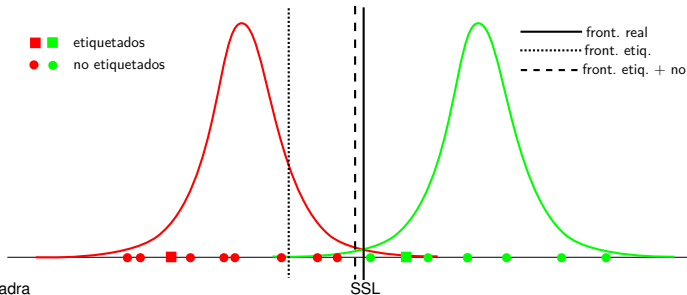
# Hipótesis de agrupamiento

- Si dos puntos  $\mathbf{x}_1, \mathbf{x}_2$  pertenecen al mismo cluster, sus etiquetas  $y_1, y_2$  deberían ser iguales. Aunque datos con la misma etiqueta pueden pertenecer a distintos clusters. Caso particular de la hipótesis de suavidad.
- Los datos no etiquetados pueden ayudar a precisar las fronteras de los clusters.



# Hipótesis de agrupamiento

- Si dos puntos  $\mathbf{x}_1, \mathbf{x}_2$  pertenecen al mismo cluster, sus etiquetas  $y_1, y_2$  deberían ser iguales. Aunque datos con la misma etiqueta pueden pertenecer a distintos clusters. Caso particular de la hipótesis de suavidad.
- Los datos no etiquetados pueden ayudar a precisar las fronteras de los clusters.



# Hipótesis de variedad (manifold)

- Una variedad es un conjunto localmente homeomórfico a un espacio euclídeo.
  - 1D: recta, círculo, pero no un 8. En 2D: plano, esfera, toro.
- Los datos están en una variedad de una dimensión mucho más pequeña que la del espacio de entrada (evitar la maldición de la dimensionalidad).
  - La voz humana se controla con 4 cuerdas vocales, no es necesario modelar el espacio de todas las señales acústicas.
  - La expresión facial se controla con unos pocos músculos, no es necesario modelar el espacio de todas las imágenes.

# Hipótesis de variedad (manifold)

- Una variedad es un conjunto localmente homeomórfico a un espacio euclídeo.
  - 1D: recta, círculo, pero no un 8. En 2D: plano, esfera, toro.
- Los datos están en una variedad de una dimensión mucho más pequeña que la del espacio de entrada (evitar la maldición de la dimensionalidad).
  - La voz humana se controla con 4 cuerdas vocales, no es necesario modelar el espacio de todas las señales acústicas.
  - La expresión facial se controla con unos pocos músculos, no es necesario modelar el espacio de todas las imágenes.

# Hipótesis de variedad (manifold)

- Una variedad es un conjunto localmente homeomórfico a un espacio euclídeo.
  - 1D: recta, círculo, pero no un 8. En 2D: plano, esfera, toro.
- Los datos están en una variedad de una dimensión mucho más pequeña que la del espacio de entrada (evitar la maldición de la dimensionalidad).
  - La voz humana se controla con 4 cuerdas vocales, no es necesario modelar el espacio de todas las señales acústicas.
  - La expresión facial se controla con unos pocos músculos, no es necesario modelar el espacio de todas las imágenes.

# Hipótesis de variedad (manifold)

- Una variedad es un conjunto localmente homeomórfico a un espacio euclídeo.
  - 1D: recta, círculo, pero no un 8. En 2D: plano, esfera, toro.
- Los datos están en una variedad de una dimensión mucho más pequeña que la del espacio de entrada (evitar la maldición de la dimensionalidad).
  - La voz humana se controla con 4 cuerdas vocales, no es necesario modelar el espacio de todas las señales acústicas.
  - La expresión facial se controla con unos pocos músculos, no es necesario modelar el espacio de todas las imágenes.

# Principio de Vapnik

## Definición (Principio de Vapnik)

Cuando se trata de resolver un problema no debería resolverse, como paso intermedio, un problema más difícil.

# Índice

## 1 Introducción a SSL

## 2 Herramientas para SSL

- Introducción
- Python
- R
- Java

## 3 Algoritmos para SSL



# Índice

## 1 Introducción a SSL

## 2 Herramientas para SSL

### ■ Introducción

### ■ Python

### ■ R

### ■ Java

## 3 Algoritmos para SSL

# Introducción

## Aviso

El software para SSL está muy disperso en distintas herramientas normalmente escritas en MatLab, C o C++.  
Consultar la web: [Semi-Supervised Learning Software](#)

- Comentaremos el software que emplearemos en las actividades asociadas a los métodos que describiremos.

# Introducción

## Aviso

El software para SSL está muy disperso en distintas herramientas normalmente escritas en MatLab, C o C++.  
Consultar la web: [Semi-Supervised Learning Software](#)

- Comentaremos el software que emplearemos en las actividades asociadas a los métodos que describiremos.

# Índice

## 1 Introducción a SSL

## 2 Herramientas para SSL

- Introducción

- Python

- R

- Java

## 3 Algoritmos para SSL

# Scikit-learn

- En scikit-learn dispone de métodos de **propagación de etiquetas**.
- Hay también módulos externos para scikit-learn.
  - **Semisup-learn** self-training y S3VM. Para self-training el modelo supervisado tiene que tener el método `predict_proba`.
  - **Sklearn-cotraining**.
  - **pomegranate** modelos generativos.

# Scikit-learn

- En scikit-learn dispone de métodos de [propagación de etiquetas](#).
- Hay también módulos externos para scikit-learn.
  - [Semisup-learn](#) self-training y S3VM. Para self-training el modelo supervisado tiene que tener el método `predict_proba`.
  - [Sklearn-cotraining](#).
  - [pomegranate](#) modelos generativos.

# Índice

## 1 Introducción a SSL

## 2 Herramientas para SSL

- Introducción

- Python

- **R**

- Java

## 3 Algoritmos para SSL

## R

- **RSSL** self-training, S3VM, métodos basados en grafos, ...
- **SSL** cotraining, propagación de etiquetas, self-training (xgboost), métodos basados en grafos, ...



## R

- **RSSL** self-training, S3VM, métodos basados en grafos, ...
- **SSL** cotraining, propagación de etiquetas, self-training (xgboost), métodos basados en grafos, ...

# Índice

## 1 Introducción a SSL

## 2 Herramientas para SSL

- Introducción
- Python
- R
- **Java**

## 3 Algoritmos para SSL

# Java

- El módulo externo para Weka [collective-classification](#) métodos basados en grafos, ...
- [Keel](#) self-training y cotraining. Dispone de interfaz gráfica.

# Java

- El módulo externo para Weka [collective-classification](#) métodos basados en grafos, ...
- [Keel](#) self-training y cotraining. Dispone de interfaz gráfica.

# Índice

1 Introducción a SSL

2 Herramientas para SSL

3 Algoritmos para SSL

- Self-training (-learning)
- Co-training y modelos multivista
- Self-labeled
- Modelos generativos
- Semi-supervised SVM (S3VM)
- Modelos basados en grafos

# Índice

1 Introducción a SSL

2 Herramientas para SSL

3 Algoritmos para SSL

- Self-training (-learning)
- Co-training y modelos multivista
- Self-labeled
- Modelos generativos
- Semi-supervised SVM (S3VM)
- Modelos basados en grafos

# Algoritmo básico

- 1 Entrenar un clasificador supervisado  $f$  con el conjunto de datos etiquetados  $L$ .
- 2 Usar el clasificador para clasificar el conjunto de datos no etiquetados  $U$ .
- 3 Pasar datos de  $U$  a  $L$ .
- 4 Repetir los pasos 1, 2 y 3 hasta que  $U$  quede vacío.

# Algoritmo básico

- 1 Entrenar un clasificador supervisado  $f$  con el conjunto de datos etiquetados  $L$ .
- 2 Usar el clasificador para clasificar el conjunto de datos no etiquetados  $U$ .
- 3 Pasar datos de  $U$  a  $L$ .
- 4 Repetir los pasos 1, 2 y 3 hasta que  $U$  quede vacío.



# Algoritmo básico

- 1 Entrenar un clasificador supervisado  $f$  con el conjunto de datos etiquetados  $L$ .
- 2 Usar el clasificador para clasificar el conjunto de datos no etiquetados  $U$ .
- 3 Pasar datos de  $U$  a  $L$ .
- 4 Repetir los pasos 1, 2 y 3 hasta que  $U$  quede vacío.

# Algoritmo básico

- 1 Entrenar un clasificador supervisado  $f$  con el conjunto de datos etiquetados  $L$ .
- 2 Usar el clasificador para clasificar el conjunto de datos no etiquetados  $U$ .
- 3 Pasar datos de  $U$  a  $L$ .
- 4 Repetir los pasos 1, 2 y 3 hasta que  $U$  quede vacío.

# Características

- Es el método más simple de SSL y es usado a menudo.
- Es un método wrapper: envuelve un método SL.
- La elección del clasificador  $f$  está abierta.
- Los errores cometidos por  $f$  tienden a reforzarse.
  - Como solución se pueden desetiquetar muestras con fiabilidad por debajo de un umbral.
- La convergencia a una solución depende del caso.

# Características

- Es el método más simple de SSL y es usado a menudo.
- Es un método wrapper: envuelve un método SL.
- La elección del clasificador  $f$  está abierta.
- Los errores cometidos por  $f$  tienden a reforzarse.
  - Como solución se pueden desetiquetar muestras con fiabilidad por debajo de un umbral.
- La convergencia a una solución depende del caso.

# Características

- Es el método más simple de SSL y es usado a menudo.
- Es un método wrapper: envuelve un método SL.
- La elección del clasificador  $f$  está abierta.
- Los errores cometidos por  $f$  tienden a reforzarse.
  - Como solución se pueden desetiquetar muestras con fiabilidad por debajo de un umbral.
- La convergencia a una solución depende del caso.

# Características

- Es el método más simple de SSL y es usado a menudo.
- Es un método wrapper: envuelve un método SL.
- La elección del clasificador  $f$  está abierta.
- Los errores cometidos por  $f$  tienden a reforzarse.
  - Como solución se pueden desetiquetar muestras con fiabilidad por debajo de un umbral.
- La convergencia a una solución depende del caso.

# Características

- Es el método más simple de SSL y es usado a menudo.
- Es un método wrapper: envuelve un método SL.
- La elección del clasificador  $f$  está abierta.
- Los errores cometidos por  $f$  tienden a reforzarse.
  - Como solución se pueden desetiquetar muestras con fiabilidad por debajo de un umbral.
- La convergencia a una solución depende del caso.

# Variaciones al método básico

- Añadir a  $L$  solo los datos más fiables en cada iteración.
- Añadir a  $L$  todos los datos, habría una sola iteración.
- Añadir a  $L$  todos los datos ponderando según su fiabilidad, habría una sola iteración.



# Variaciones al método básico

- Añadir a  $L$  solo los datos más fiables en cada iteración.
- Añadir a  $L$  todos los datos, habría una sola iteración.
- Añadir a  $L$  todos los datos ponderando según su fiabilidad, habría una sola iteración.

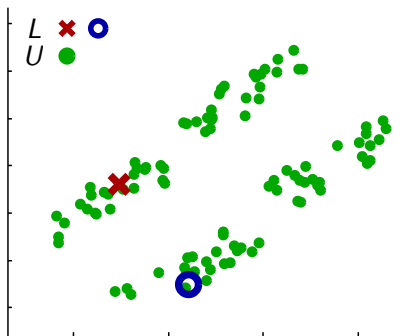
## Variaciones al método básico

- Añadir a  $L$  solo los datos más fiables en cada iteración.
- Añadir a  $L$  todos los datos, habría una sola iteración.
- Añadir a  $L$  todos los datos ponderando según su fiabilidad, habría una sola iteración.

# Ejemplo exitoso con 1-NN

## Tenemos $L$ y $U$ .

- 1 Seleccionamos elemento de  $U$  que esté a la mínima distancia de cualquier elemento de  $L$ .
  - 2 Le damos la etiqueta del elemento de  $L$  más cercano a él y lo pasamos a  $L$ . Los empates se deshacen aleatoriamente.
- Repetimos 1 y 2 hasta vaciar  $U$ .

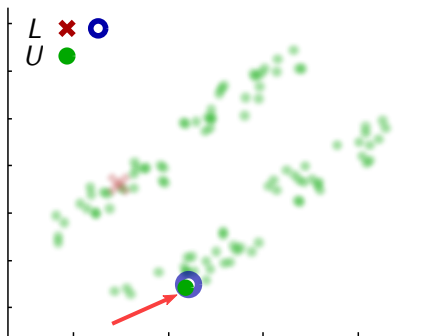


# Ejemplo exitoso con 1-NN

## Tenemos $L$ y $U$ .

- 1 Seleccionamos elemento de  $U$  que esté a la mínima distancia de cualquier elemento de  $L$ .
- 2 Le damos la etiqueta del elemento de  $L$  más cercano a él y lo pasamos a  $L$ . Los empates se deshacen aleatoriamente.

## Repetimos 1 y 2 hasta vaciar $U$ .

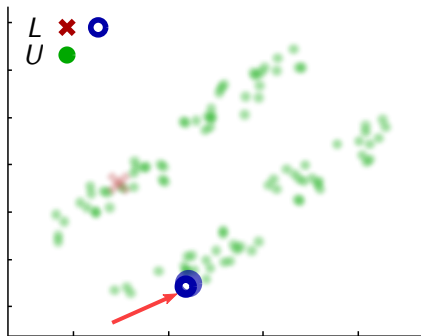


# Ejemplo exitoso con 1-NN

## ■ Tenemos $L$ y $U$ .

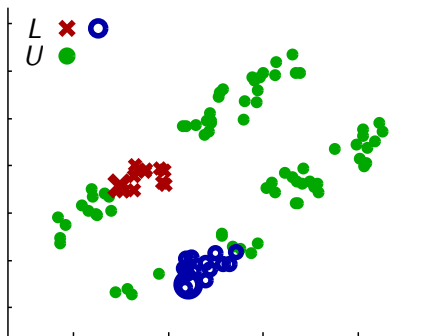
- 1 Seleccionamos elemento de  $U$  que esté a la mínima distancia de cualquier elemento de  $L$ .
- 2 Le damos la etiqueta del elemento de  $L$  más cercano a él y lo pasamos a  $L$ . Los empates se deshacen aleatoriamente.

## ■ Repetimos 1 y 2 hasta vaciar $U$ .



# Ejemplo exitoso con 1-NN

- Tenemos  $L$  y  $U$ .
  - 1 Seleccionamos elemento de  $U$  que esté a la mínima distancia de cualquier elemento de  $L$ .
  - 2 Le damos la etiqueta del elemento de  $L$  más cercano a él y lo pasamos a  $L$ . Los empates se deshacen aleatoriamente.
- Repetimos 1 y 2 hasta vaciar  $U$ .



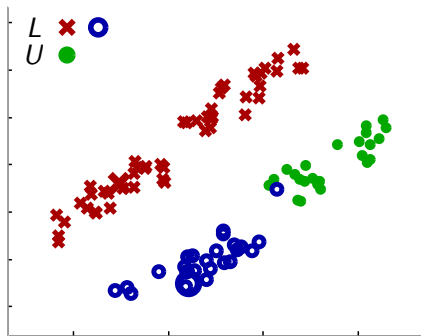
# Ejemplo exitoso con 1-NN

## Tenemos $L$ y $U$ .

- 1 Seleccionamos elemento de  $U$  que esté a la mínima distancia de cualquier elemento de  $L$ .
- 2 Le damos la etiqueta del elemento de  $L$  más cercano a él y lo pasamos a  $L$ . Los empates se deshacen aleatoriamente.

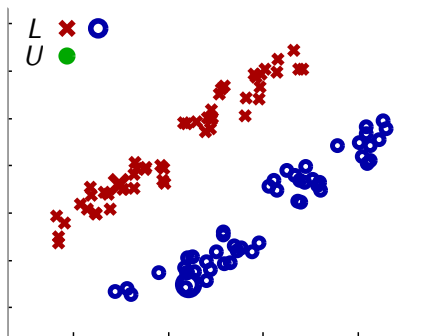
## Tenemos $L$ y $U$ .

Repetimos 1 y 2 hasta vaciar  $U$ .



# Ejemplo exitoso con 1-NN

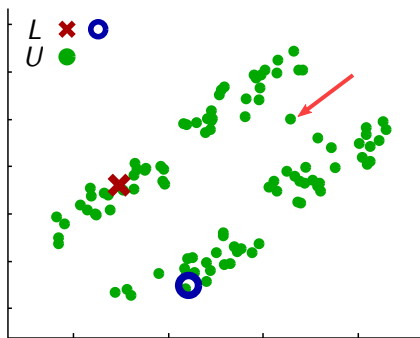
- Tenemos  $L$  y  $U$ .
  - 1 Seleccionamos elemento de  $U$  que esté a la mínima distancia de cualquier elemento de  $L$ .
  - 2 Le damos la etiqueta del elemento de  $L$  más cercano a él y lo pasamos a  $L$ . Los empates se deshacen aleatoriamente.
- Repetimos 1 y 2 hasta vaciar  $U$ .





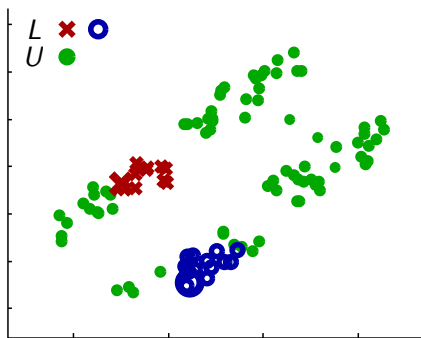
# Ejemplo fallido con 1-NN

- Tenemos  $L$  y  $U$ , pero ahora hay un outlier.
- Procedemos como en el ejemplo anterior.
- Un outlier situado en el sitio oportuno puede dar al traste con la clasificación.
- El outlier propaga su etiqueta.
- La clasificación no es correcta.



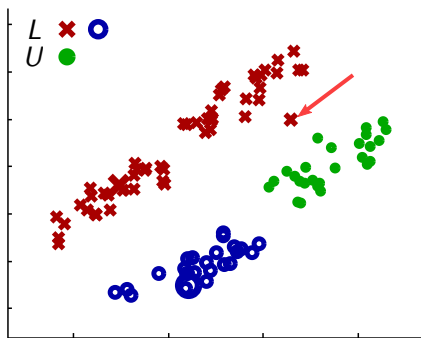
# Ejemplo fallido con 1-NN

- Tenemos  $L$  y  $U$ , pero ahora hay un outlier.
- Procedemos como en el ejemplo anterior.
- Un outlier situado en el sitio oportuno puede dar al traste con la clasificación.
- El outlier propaga su etiqueta.
- La clasificación no es correcta.



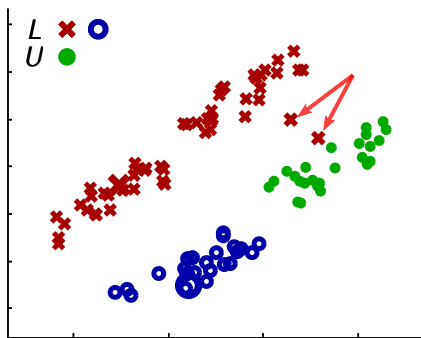
# Ejemplo fallido con 1-NN

- Tenemos  $L$  y  $U$ , pero ahora hay un outlier.
- Procedemos como en el ejemplo anterior.
- Un outlier situado en el sitio oportuno puede dar al traste con la clasificación.
- El outlier propaga su etiqueta.
- La clasificación no es correcta.



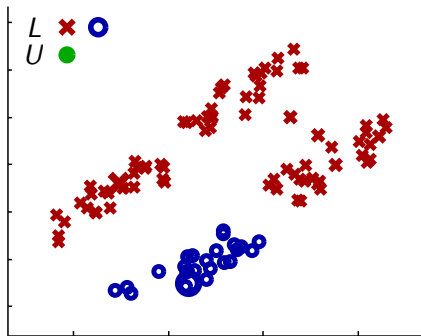
# Ejemplo fallido con 1-NN

- Tenemos  $L$  y  $U$ , pero ahora hay un outlier.
- Procedemos como en el ejemplo anterior.
- Un outlier situado en el sitio oportuno puede dar al traste con la clasificación.
- El outlier propaga su etiqueta.
- La clasificación no es correcta.



# Ejemplo fallido con 1-NN

- Tenemos  $L$  y  $U$ , pero ahora hay un outlier.
- Procedemos como en el ejemplo anterior.
- Un outlier situado en el sitio oportuno puede dar al traste con la clasificación.
- El outlier propaga su etiqueta.
- La clasificación no es correcta.



# Self-training con semisup-learn (Python)

- Para el ejemplo con semisup-learn cargaremos el notebook `self-training-iris.ipynb`.
- Clasificar flores iris usando aprendizaje supervisado en los datos etiquetados y self-training con todos los datos.

# Self-training con semisup-learn (Python)

- Para el ejemplo con semisup-learn cargaremos el notebook `self-training-iris.ipynb`.
- Clasificar flores iris usando aprendizaje supervisado en los datos etiquetados y self-training con todos los datos.

# Self-training con SSL (R)

- Para el ejemplo con SSL cargaremos el notebook `ssl-selftraining-example.ipynb`.
- Clasificar flores iris usando aprendizaje semi-supervisado.



# Self-training con SSL (R)

- Para el ejemplo con SSL cargaremos el notebook `ssl-selftraining-example.ipynb`.
- Clasificar flores iris usando aprendizaje semi-supervisado.

# Self-training con RSSL (R)

- Para el ejemplo con RSSL cargaremos el notebook `example-self-learning-rssl.ipynb`.
- Ejemplo del uso de `LearningCurveSSL` para ver cómo puede fallar un self-learning.

# Self-training con RSSL (R)

- Para el ejemplo con RSSL cargaremos el notebook `example-self-learning-rssl.ipynb`.
- Ejemplo del uso de `LearningCurveSSL` para ver cómo puede fallar un self-learning.

# Actividades self-training

## Actividad

Clasificar mediante self-training los datos de titanic.csv en "Survived" o no usando como características "Pclass", "Sex", "Age". Estudiar los datos antes para ver si hay problemas: datos no disponibles, etc. Dar la matriz de confusión. Usar los tres módulos comentados.

Para cargar csv ver titanic\_start\_python.ipynb y titanic-start\_r.ipynb.

# Índice

1 Introducción a SSL

2 Herramientas para SSL

3 Algoritmos para SSL

- Self-training (-learning)
- **Co-training y modelos multivista**
- Self-labeled
- Modelos generativos
- Semi-supervised SVM (S3VM)
- Modelos basados en grafos

# Introducción

- Queremos clasificar web en categorías.
  - Creamos dos vistas (conjuntos de características  $X = [X_1, X_2]$  de cada web: contenido  $X_1$  y link  $X_2$ .
  - Etiquetamos las vistas de algunas webs según las categorías.
- Entrenamos un clasificador para cada vista y cada uno clasifica algunos datos no etiquetados
- Cada clasificador enseña al otro con los datos que ha etiquetado.
- Como el self-training pero con dos clasificadores que se enseñan mutuamente.

# Introducción

- Queremos clasificar web en categorías.
  - Creamos dos vistas (conjuntos de características  $X = [X_1, X_2]$  de cada web: contenido  $X_1$  y link  $X_2$ .
  - Etiquetamos las vistas de algunas webs según las categorías.
- Entrenamos un clasificador para cada vista y cada uno clasifica algunos datos no etiquetados
- Cada clasificador enseña al otro con los datos que ha etiquetado.
- Como el self-training pero con dos clasificadores que se enseñan mutuamente.

# Introducción

- Queremos clasificar web en categorías.
  - Creamos dos vistas (conjuntos de características  $X = [X_1, X_2]$  de cada web: contenido  $X_1$  y link  $X_2$ .
  - Etiquetamos las vistas de algunas webs según las categorías.
- Entrenamos un clasificador para cada vista y cada uno clasifica algunos datos no etiquetados
- Cada clasificador enseña al otro con los datos que ha etiquetado.
- Como el self-training pero con dos clasificadores que se enseñan mutuamente.



# Introducción

- Queremos clasificar web en categorías.
  - Creamos dos vistas (conjuntos de características  $X = [X_1, X_2]$  de cada web: contenido  $X_1$  y link  $X_2$ .
  - Etiquetamos las vistas de algunas webs según las categorías.
- Entrenamos un clasificador para cada vista y cada uno clasifica algunos datos no etiquetados
- Cada clasificador enseña al otro con los datos que ha etiquetado.
- Como el self-training pero con dos clasificadores que se enseñan mutuamente.

# Hipótesis del co-training

## Hipótesis del co-training

- La división en vistas  $X = [X_1, X_2]$  existe.
- Cada vista es capaz de entrenar un clasificador.
- Las vistas son condicionalmente independientes dada una clase  $C$  ( $X_1$  y  $X_2$  no tiene porqué ser independientes pero si nos dan  $C$  sí (peso, vocabulario y edad).

# Hipótesis del co-training

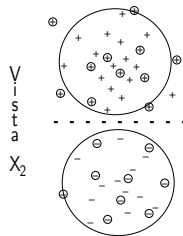
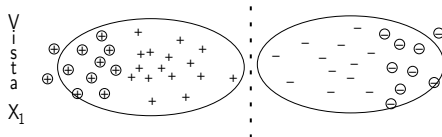
## Hipótesis del co-training

- La división en vistas  $X = [X_1, X_2]$  existe.
- Cada vista es capaz de entrenar un clasificador.
- Las vistas son condicionalmente independientes dada una clase  $C$  ( $X_1$  y  $X_2$  no tiene porqué ser independientes pero si nos dan  $C$  sí (peso, vocabulario y edad).

# Hipótesis del co-training

## Hipótesis del co-training

- La división en vistas  $X = [X_1, X_2]$  existe.
- Cada vista es capaz de entrenar un clasificador.
- Las vistas son condicionalmente independientes dada una clase  $C$  ( $X_1$  y  $X_2$  no tiene porqué ser independientes pero si nos dan  $C$  sí (peso, vocabulario y edad).



# Algoritmo básico del co-training

- 1 Entrenar dos clasificadores  $f_1$  a partir de  $(X_1^l, Y_1^l)$ ,  $f_2$  a partir de  $(X_2^l, Y_2^l)$ .
- 2 Clasificar  $X^u$  usando  $f_1$  y  $f_2$  de forma independiente.
- 3 Co-training a velocidad  $k$ .
  - Añadir los  $k$  datos más fiablemente etiquetados por  $f_1$  a  $(X_2^l, Y_2^l)$ .
  - Añadir los  $k$  datos más fiablemente etiquetados por  $f_2$  a  $(X_1^l, Y_1^l)$ .
- 4 Eliminar estos  $2k$  datos de  $X^u$ .
- 5 Repetir los pasos 2, 3 y 4 hasta acabar con  $X^u$ .

# Algoritmo básico del co-training

- 1 Entrenar dos clasificadores  $f_1$  a partir de  $(X_1^l, Y_1^l)$ ,  $f_2$  a partir de  $(X_2^l, Y_2^l)$ .
- 2 Clasificar  $X^u$  usando  $f_1$  y  $f_2$  de forma independiente.
- 3 Co-training a velocidad  $k$ .
  - Añadir los  $k$  datos más fiablemente etiquetados por  $f_1$  a  $(X_2^l, Y_2^l)$ .
  - Añadir los  $k$  datos más fiablemente etiquetados por  $f_2$  a  $(X_1^l, Y_1^l)$ .
- 4 Eliminar estos  $2k$  datos de  $X^u$ .
- 5 Repetir los pasos 2, 3 y 4 hasta acabar con  $X^u$ .

# Algoritmo básico del co-training

- 1 Entrenar dos clasificadores  $f_1$  a partir de  $(X_1^l, Y_1^l)$ ,  $f_2$  a partir de  $(X_2^l, Y_2^l)$ .
- 2 Clasificar  $X^u$  usando  $f_1$  y  $f_2$  de forma independiente.
- 3 Co-training a velocidad  $k$ .
  - Añadir los  $k$  datos más fiablemente etiquetados por  $f_1$  a  $(X_2^l, Y_2^l)$ .
  - Añadir los  $k$  datos más fiablemente etiquetados por  $f_2$  a  $(X_1^l, Y_1^l)$ .
- 4 Eliminar estos  $2k$  datos de  $X^u$ .
- 5 Repetir los pasos 2, 3 y 4 hasta acabar con  $X^u$ .

# Algoritmo básico del co-training

- 1 Entrenar dos clasificadores  $f_1$  a partir de  $(X_1^l, Y_1^l)$ ,  $f_2$  a partir de  $(X_2^l, Y_2^l)$ .
- 2 Clasificar  $X^u$  usando  $f_1$  y  $f_2$  de forma independiente.
- 3 Co-training a velocidad  $k$ .
  - Añadir los  $k$  datos más fiablemente etiquetados por  $f_1$  a  $(X_2^l, Y_2^l)$ .
  - Añadir los  $k$  datos más fiablemente etiquetados por  $f_2$  a  $(X_1^l, Y_1^l)$ .
- 4 Eliminar estos  $2k$  datos de  $X^u$ .
- 5 Repetir los pasos 2, 3 y 4 hasta acabar con  $X^u$ .



# Algoritmo básico del co-training

- 1 Entrenar dos clasificadores  $f_1$  a partir de  $(X_1^l, Y_1^l)$ ,  $f_2$  a partir de  $(X_2^l, Y_2^l)$ .
- 2 Clasificar  $X^u$  usando  $f_1$  y  $f_2$  de forma independiente.
- 3 Co-training a velocidad  $k$ .
  - Añadir los  $k$  datos más fiablemente etiquetados por  $f_1$  a  $(X_2^l, Y_2^l)$ .
  - Añadir los  $k$  datos más fiablemente etiquetados por  $f_2$  a  $(X_1^l, Y_1^l)$ .
- 4 Eliminar estos  $2k$  datos de  $X^u$ .
- 5 Repetir los pasos 2, 3 y 4 hasta acabar con  $X^u$ .

# Pros y contras del co-training

## Pros

- Es un método wrapper que se puede aplicar a mucho clasificadores supervisados.
- Es menos sensible a errores que el self-training.

# Pros y contras del co-training

## Pros

- Es un método wrapper que se puede aplicar a mucho clasificadores supervisados.
- Es menos sensible a errores que el self-training.

## Contras

- La división de vistas de forma natural puede no existir.
- Modelos usando ambas vistas pueden funcionar mejor.

# Variantes del co-training

- **Co-EM (expectation maximization)**: da etiquetas probabilísticas que pueden cambiar entre iteraciones.
- **Co-regularization**: minimiza una función que depende de la complejidad (normas) de  $f_1$  y  $f_2$ , su acuerdo en los datos no etiquetados y una función de pérdida evaluada en los datos etiquetados usando la media entre  $f_1$  y  $f_2$ .
- **Co-regression**: usa regresores en lugar de clasificadores, la fiabilidad de las nuevas etiquetas se estima por la disminución del MSE.
- **Co-clustering**: funciona bajo la hipótesis de que el agrupamiento real subyacente asignará los puntos correspondientes en cada vista al mismo grupo.

# Variantes del co-training

- **Co-EM (expectation maximization):** da etiquetas probabilísticas que pueden cambiar entre iteraciones.
- **Co-regularization:** minimiza una función que depende de la complejidad (normas) de  $f_1$  y  $f_2$ , su acuerdo en los datos no etiquetados y una función de pérdida evaluada en los datos etiquetados usando la media entre  $f_1$  y  $f_2$ .
- **Co-regression:** usa regresores en lugar de clasificadores, la fiabilidad de las nuevas etiquetas se estima por la disminución del MSE.
- **Co-clustering:** funciona bajo la hipótesis de que el agrupamiento real subyacente asignará los puntos correspondientes en cada vista al mismo grupo.

# Variantes del co-training

- **Co-EM (expectation maximization)**: da etiquetas probabilísticas que pueden cambiar entre iteraciones.
- **Co-regularization**: minimiza una función que depende de la complejidad (normas) de  $f_1$  y  $f_2$ , su acuerdo en los datos no etiquetados y una función de pérdida evaluada en los datos etiquetados usando la media entre  $f_1$  y  $f_2$ .
- **Co-regression**: usa regresores en lugar de clasificadores, la fiabilidad de las nuevas etiquetas se estima por la disminución del MSE.
- **Co-clustering**: funciona bajo la hipótesis de que el agrupamiento real subyacente asignará los puntos correspondientes en cada vista al mismo grupo.

# Variantes del co-training

- **Co-EM (expectation maximization)**: da etiquetas probabilísticas que pueden cambiar entre iteraciones.
- **Co-regularization**: minimiza una función que depende de la complejidad (normas) de  $f_1$  y  $f_2$ , su acuerdo en los datos no etiquetados y una función de pérdida evaluada en los datos etiquetados usando la media entre  $f_1$  y  $f_2$ .
- **Co-regression**: usa regresores en lugar de clasificadores, la fiabilidad de las nuevas etiquetas se estima por la disminución del MSE.
- **Co-clustering**: funciona bajo la hipótesis de que el agrupamiento real subyacente asignará los puntos correspondientes en cada vista al mismo grupo.

# Modelos multivista

- Son una extensión del co-training en la que se usan más de dos clasificadores.
  - Se entrenan clasificadores de distintos tipos.
  - Se clasifican los datos no etiquetados con cada clasificador.
  - Se añade la etiqueta votada por la mayoría.



# Modelos multivista

- Son una extensión del co-training en la que se usan más de dos clasificadores.
  - Se entrenan clasificadores de distintos tipos.
  - Se clasifican los datos no etiquetados con cada clasificador.
  - Se añade la etiqueta votada por la mayoría.

# Cotraining con sklearn-cotraining (Python)

- Para el ejemplo con sklearn-cotraining cargaremos el notebook `sklearn-cotraining-examples.ipynb`.
- Coentrenaremos varios clasificadores para clasificar un conjunto con 25000 datos y 1000 características, comparando la clasificación supervisada con la semi-supervisada.

# Cotraining con sklearn-cotraining (Python)

- Para el ejemplo con sklearn-cotraining cargaremos el notebook `sklearn-cotraining-examples.ipynb`.
- Coentrenaremos varios clasificadores para clasificar un conjunto con 25000 datos y 1000 características, comparando la clasificación supervisada con la semi-supervisada.

# Cotraining con SSL (R)

- Para el ejemplo con SSL cargaremos el notebook `ssl-cotraining-example.ipynb`.
- Clasificar flores iris usando aprendizaje semi-supervisado.

# Cotraining con SSL (R)

- Para el ejemplo con SSL cargaremos el notebook `ssl-cotraining-example.ipynb`.
- Clasificar flores iris usando aprendizaje semi-supervisado.

# Actividades cotraining

## Actividad

Clasificar mediante cotraining los datos de titanic.csv en "Survived" o no usando como características "Pclass", "Sex", "Age". Estudiar los datos antes para ver si hay problemas: datos no disponibles, etc. Dar la matriz de confusión. Usar los dos módulos comentados.

# Índice

1 Introducción a SSL

2 Herramientas para SSL

3 Algoritmos para SSL

- Self-training (-learning)
- Co-training y modelos multivista
- **Self-labeled**
- Modelos generativos
- Semi-supervised SVM (S3VM)
- Modelos basados en grafos

# Introducción a self-labeled

1 Estos métodos engloban al self-training y al co-training/multivista.

2 Características para clasificar:

- Mecanismo de adición:

- Incremental:  $k$  más fiables en cada iteración.

- Por lotes: Se decide si un dato cumple el criterio de adición pero hasta tomar todas las decisiones no se etiquetan realmente (todos) los datos, se puede cambiar de opinión.

- Corrección: se añaden todos lo que cumplen un criterio, los que se etiquetan en una iteración pueden perderla en otra.

- Clasificador simple (p. ej. self-learning) vs. múltiple (p. ej. co.training).

- Usar uno o varios algoritmos de aprendizaje.

- Vista simple o multivista.



# Introducción a self-labeled

- 1 Estos métodos engloban al self-training y al co-training/multivista.
- 2 Características para clasificar:
  - 1 Mecanismo de adición:
    - 1 Incremental:  $k$  más fiables en cada iteración.
    - 2 Por lotes: Se decide si un dato cumple el criterio de adición pero hasta tomar todas las decisiones no se etiquetan realmente (todos) los datos, se puede cambiar de opinión.
    - 3 Corrección: se añaden todos lo que cumplen un criterio, los que se etiquetan en una iteración pueden perderla en otra.
  - 2 Clasificador simple (p. ej. self-learning) vs. múltiple (p. ej. co.training).
  - 3 Usar uno o varios algoritmos de aprendizaje.
  - 4 Vista simple o multivista.

# Introducción a self-labeled

- 1 Estos métodos engloban al self-training y al co-training/multivista.
- 2 Características para clasificar:
  - 1 Mecanismo de adición:
    - 1 Incremental:  $k$  más fiables en cada iteración.
    - 2 Por lotes: Se decide si un dato cumple el criterio de adición pero hasta tomar todas las decisiones no se etiquetan realmente (todos) los datos, se puede cambiar de opinión.
    - 3 Corrección: se añaden todos lo que cumplen un criterio, los que se etiquetan en una iteración pueden perderla en otra.
  - 2 Clasificador simple (p. ej. self-learning) vs. múltiple (p. ej. co.training).
  - 3 Usar uno o varios algoritmos de aprendizaje.
  - 4 Vista simple o multivista.

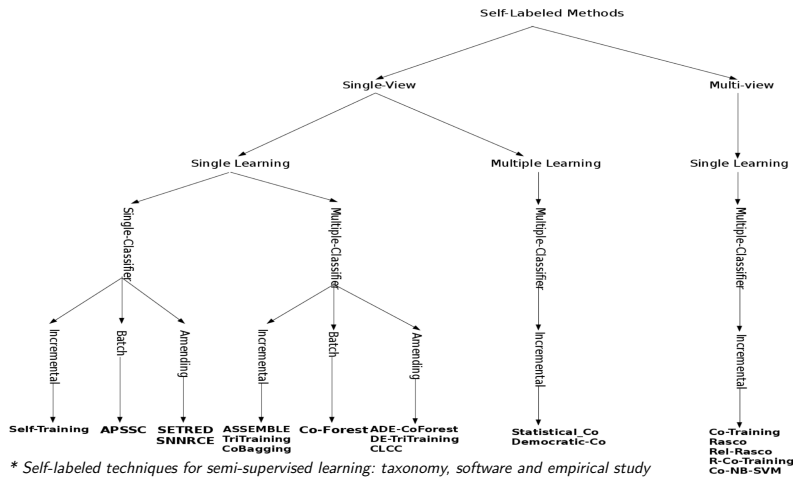
# Introducción a self-labeled

- 1 Estos métodos engloban al self-training y al co-training/multivista.
- 2 Características para clasificar:
  - 1 Mecanismo de adición:
    - 1 Incremental:  $k$  más fiables en cada iteración.
    - 2 Por lotes: Se decide si un dato cumple el criterio de adición pero hasta tomar todas las decisiones no se etiquetan realmente (todos) los datos, se puede cambiar de opinión.
    - 3 Corrección: se añaden todos lo que cumplen un criterio, los que se etiquetan en una iteración pueden perderla en otra.
  - 2 Clasificador simple (p. ej. self-learning) vs. múltiple (p. ej. co.training).
  - 3 Usar uno o varios algoritmos de aprendizaje.
  - 4 Vista simple o multivista.

# Introducción a self-labeled

- 1 Estos métodos engloban al self-training y al co-training/multivista.
- 2 Características para clasificar:
  - 1 Mecanismo de adición:
    - 1 Incremental:  $k$  más fiables en cada iteración.
    - 2 Por lotes: Se decide si un dato cumple el criterio de adición pero hasta tomar todas las decisiones no se etiquetan realmente (todos) los datos, se puede cambiar de opinión.
    - 3 Corrección: se añaden todos lo que cumplen un criterio, los que se etiquetan en una iteración pueden perderla en otra.
  - 2 Clasificador simple (p. ej. self-learning) vs. múltiple (p. ej. co.training).
  - 3 Usar uno o varios algoritmos de aprendizaje.
  - 4 Vista simple o multivista.

# Taxonomía de métodos self-labeled



# Uso de KEEL

- Vamos a ver el uso sencillo de KEEL creando un experimento de clasificación como se explica en el manual sección 7.3 (pg. 172) y sección 2.3.1 (pg. 14).

# Índice

1 Introducción a SSL

2 Herramientas para SSL

3 Algoritmos para SSL

- Self-training (-learning)
- Co-training y modelos multivista
- Self-labeled
- **Modelos generativos**
- Semi-supervised SVM (S3VM)
- Modelos basados en grafos

# Definición de modelo generativo

## Definición (Modelo generativo)

$p(\mathbf{x}, y) = p(y)p(\mathbf{x}|y)$  siendo  $p(\mathbf{x}|y)$  una distribución de probabilidad compuesta (mixture distribution MM) identificable.

Una MM es una distribución de probabilidad en la que los parámetros, o parte de ellos, son variables aleatorias.

Una familia de distribuciones  $\{p_{\theta}\}$  es identificable si

$$\theta_1 \neq \theta_2 \implies p_{\theta_1} \neq p_{\theta_2}.$$

- Con una cantidad suficiente de datos no etiquetados se pueden identificar los componentes de la composición.



# Definición de modelo generativo

## Definición (Modelo generativo)

$p(\mathbf{x}, y) = p(y)p(\mathbf{x}|y)$  siendo  $p(\mathbf{x}|y)$  una distribución de probabilidad compuesta (mixture distribution MM) identificable.

Una MM es una distribución de probabilidad en la que los parámetros, o parte de ellos, son variables aleatorias.

Una familia de distribuciones  $\{p_{\theta}\}$  es identificable si

$$\theta_1 \neq \theta_2 \implies p_{\theta_1} \neq p_{\theta_2}.$$

- Con una cantidad suficiente de datos no etiquetados se pueden identificar los componentes de la composición.

# Modelo generativo en SSL

- Asumimos que conocemos  $p(\mathbf{x}, y)$  con parámetros  $\theta$ .
- La distribución conjunta y la marginal  
$$p(X^l, Y^l, X^u | \theta) = \sum_{Y^u} p(X^l, Y^l, X^u, Y^u | \theta).$$
- Estimar  $\theta$  usando MLE o MAP o por métodos bayesianos.
- Con una cantidad suficiente de datos no etiquetados se pueden identificar los componentes de la suma.

Se puede ver como clustering con información adicional sobre la marginal.

# Modelo generativo en SSL

- Asumimos que conocemos  $p(\mathbf{x}, y)$  con parámetros  $\theta$ .
- La distribución conjunta y la marginal  
$$p(X^l, Y^l, X^u | \theta) = \sum_{Y^u} p(X^l, Y^l, X^u, Y^u | \theta).$$
- Estimar  $\theta$  usando MLE o MAP o por métodos bayesianos.
- Con una cantidad suficiente de datos no etiquetados se pueden identificar los componentes de la suma.

Se puede ver como clustering con información adicional sobre la marginal.

# Modelo generativo en SSL

- Asumimos que conocemos  $p(\mathbf{x}, y)$  con parámetros  $\theta$ .
- La distribución conjunta y la marginal  
$$p(X^l, Y^l, X^u | \theta) = \sum_{Y^u} p(X^l, Y^l, X^u, Y^u | \theta).$$
- Estimar  $\theta$  usando MLE o MAP o por métodos bayesianos.
- Con una cantidad suficiente de datos no etiquetados se pueden identificar los componentes de la suma.

Se puede ver como clustering con información adicional sobre la marginal.

# Modelo generativo en SSL

- Asumimos que conocemos  $p(\mathbf{x}, y)$  con parámetros  $\theta$ .
- La distribución conjunta y la marginal  
$$p(X^l, Y^l, X^u | \theta) = \sum_{Y^u} p(X^l, Y^l, X^u, Y^u | \theta).$$
- Estimar  $\theta$  usando MLE o MAP o por métodos bayesianos.
- Con una cantidad suficiente de datos no etiquetados se pueden identificar los componentes de la suma.

Se puede ver como clustering con información adicional sobre la marginal.

# Ejemplos de modelos generativos

- **Mixtura de gaussianas (GMM):**
  - Clasificación de imágenes usando MLE.
- Mixtura de multinomiales (Naïve Bayes):
  - Categorización de textos usando MLE.
- Modelos de Markov ocultos (HMM):
  - Reconocimiento de voz usando el algoritmo de Baum y Welch.

# Ejemplos de modelos generativos

- **Mixtura de gaussianas (GMM):**
  - Clasificación de imágenes usando MLE.
- **Mixtura de multinomiales (Naïve Bayes):**
  - Categorización de textos usando MLE.
- **Modelos de Markov ocultos (HMM):**
  - Reconocimiento de voz usando el algoritmo de Baum y Welch.

# Ejemplos de modelos generativos

- **Mixtura de gaussianas (GMM):**
  - Clasificación de imágenes usando MLE.
- **Mixtura de multinomiales (Naïve Bayes):**
  - Categorización de textos usando MLE.
- **Modelos de Markov ocultos (HMM):**
  - Reconocimiento de voz usando el algoritmo de Baum y Welch.



# Clasificación con GMM usando MLE

## La mixtura

- Parámetros del modelo:  $\theta = \{w_1, w_2, \mu_1, \mu_2, \Sigma_1, \Sigma_2\}$   
proporciones de clases, medias y covarianzas.

- La GMM:  $p(\mathbf{x}, y) = \sum_{i=1}^2 w_i \mathcal{N}(\mu_i, \Sigma_i)$ .

- Clasificación:  $p(y|\mathbf{x}, \theta) = \frac{p(\mathbf{x}, y|\theta)}{\sum_{y'} p(\mathbf{x}, y'|\theta)}$ .

# Clasificación con GMM usando MLE

## La mixtura

- Parámetros del modelo:  $\theta = \{w_1, w_2, \mu_1, \mu_2, \Sigma_1, \Sigma_2\}$   
proporciones de clases, medias y covarianzas.
- La GMM:  $p(\mathbf{x}, y) = \sum_{i=1}^2 w_i \mathcal{N}(\mu_i, \Sigma_i)$ .
- Clasificación:  $p(y|\mathbf{x}, \theta) = \frac{p(\mathbf{x}, y|\theta)}{\sum_{y'} p(\mathbf{x}, y'|\theta)}$ .

# Clasificación con GMM usando MLE

## Clasificación binaria

### ■ Usando datos etiquetados

- $\log p(X_l, Y_l | \theta) = \sum_{i=1}^l \log p(y_i | \theta) p(\mathbf{x}_i | y_i, \theta).$

- La MLE es trivial:  $w_i \rightarrow$  frecuencias,  $\mu_i \rightarrow$  medias muestrales  
 $\Sigma_i \rightarrow$  covarianzas muestrales para cada clase.

### ■ Con los datos etiquetados y **no etiquetados**:

- $\log p(X_l, Y_l, X_u | \theta) = \sum_{i=1}^l \log p(y_i | \theta) p(\mathbf{x}_i | y_i, \theta) +$   
 $\sum_{i=l+1}^{l+u} \log \left( \sum_{y=1}^2 p(y | \theta) p(\mathbf{x}_i | y, \theta) \right).$

### ■ La MLE es difícil, hay variables ocultas ( $Y^u$ ).

- Usaremos el algoritmo iterativo EM para obtener un máximo local.

# Clasificación con GMM usando MLE

## Clasificación binaria

### ■ Usando datos etiquetados

- $\log p(X_l, Y_l | \theta) = \sum_{i=1}^l \log p(y_i | \theta) p(\mathbf{x}_i | y_i, \theta).$

- La MLE es trivial:  $w_i \rightarrow$  frecuencias,  $\mu_i \rightarrow$  medias muestrales  
 $\Sigma_i \rightarrow$  covarianzas muestrales para cada clase.

### ■ Con los datos etiquetados y **no etiquetados**:

- $\log p(X_l, Y_l, X_u | \theta) = \sum_{i=1}^l \log p(y_i | \theta) p(\mathbf{x}_i | y_i, \theta) +$   
 $\sum_{i=l+1}^{l+u} \log \left( \sum_{y=1}^2 p(y | \theta) p(\mathbf{x}_i | y, \theta) \right).$

### ■ La MLE es difícil, hay variables ocultas ( $Y^u$ ).

- Usaremos el algoritmo iterativo EM para obtener un máximo local.

# Clasificación con GMM usando MLE

## Clasificación binaria

### ■ Usando datos etiquetados

$$\blacksquare \log p(X_l, Y_l | \theta) = \sum_{i=1}^l \log p(y_i | \theta) p(\mathbf{x}_i | y_i, \theta).$$

- La MLE es trivial:  $w_i \rightarrow$  frecuencias,  $\mu_i \rightarrow$  medias muestrales  
 $\Sigma_i \rightarrow$  covarianzas muestrales para cada clase.

### ■ Con los datos etiquetados y **no etiquetados**:

$$\blacksquare \log p(X_l, Y_l, X_u | \theta) = \sum_{i=1}^l \log p(y_i | \theta) p(\mathbf{x}_i | y_i, \theta) +$$

$$\sum_{i=l+1}^{l+u} \log \left( \sum_{y=1}^2 p(y | \theta) p(\mathbf{x}_i | y, \theta) \right).$$

### ■ La MLE es difícil, hay variables ocultas ( $Y^u$ ).

- Usaremos el algoritmo iterativo EM para obtener un máximo local.

# Clasificación con GMM usando MLE

## Aplicación de EM

- 1 Obtenemos una estimación inicial

$\theta^0 = \{w_1^0, w_2^0, \mu_1^0, \mu_2^0, \Sigma_1^0, \Sigma_2^0\}$  por MLE usando  $(X^I, Y^I)$ .

- 2 Paso E: etiqueta esperada  $p(y|\mathbf{x}, \theta^k) = \frac{p(\mathbf{x}, y|\theta^k)}{\sum_{y'} p(\mathbf{x}, y'|\theta^k)} \quad \forall \mathbf{x} \in X^u$

■  $x$  se etiqueta de clase  $c = 1 : 2$  con proporción  $p(y = c|\mathbf{x}, \theta^k)$ .

- 3 Paso M: MLE de  $\theta$  con los ahora datos etiquetados de  $X^u$ .

■  $w_{1:2} \rightarrow$  suma de las proporciones de cada clase.

■  $\mu_{1:2}, \Sigma_{1:2} \rightarrow$  media y covarianza ponderadas de cada clase.

- 4 Repetimos 2 y 3 hasta alcanzar un máximo local,  $k = 1, 2, \dots$

■ Es una especie de self-training.

# Clasificación con GMM usando MLE

## Aplicación de EM

- 1 Obtenemos una estimación inicial

$\theta^0 = \{w_1^0, w_2^0, \mu_1^0, \mu_2^0, \Sigma_1^0, \Sigma_2^0\}$  por MLE usando  $(X^I, Y^I)$ .

- 2 Paso E: etiqueta esperada  $p(y|\mathbf{x}, \theta^k) = \frac{p(\mathbf{x}, y|\theta^k)}{\sum_{y'} p(\mathbf{x}, y'|\theta^k)} \forall \mathbf{x} \in X^u$

- $x$  se etiqueta de clase  $c = 1 : 2$  con proporción  $p(y = c|\mathbf{x}, \theta^k)$ .

- 3 Paso M: MLE de  $\theta$  con los ahora datos etiquetados de  $X^u$ .

- $w_{1:2} \rightarrow$  suma de las proporciones de cada clase.

- $\mu_{1:2}, \Sigma_{1:2} \rightarrow$  media y covarianza ponderadas de cada clase.

- 4 Repetimos 2 y 3 hasta alcanzar un máximo local,  $k = 1, 2, \dots$

- Es una especie de self-training.

# Clasificación con GMM usando MLE

## Aplicación de EM

- 1 Obtenemos una estimación inicial

$\theta^0 = \{w_1^0, w_2^0, \mu_1^0, \mu_2^0, \Sigma_1^0, \Sigma_2^0\}$  por MLE usando  $(X^I, Y^I)$ .

- 2 Paso E: etiqueta esperada  $p(y|\mathbf{x}, \theta^k) = \frac{p(\mathbf{x}, y|\theta^k)}{\sum_{y'} p(\mathbf{x}, y'|\theta^k)} \forall \mathbf{x} \in X^u$

- $x$  se etiqueta de clase  $c = 1 : 2$  con proporción  $p(y = c|\mathbf{x}, \theta^k)$ .

- 3 Paso M: MLE de  $\theta$  con los ahora datos etiquetados de  $X^u$ .

- $w_{1:2} \rightarrow$  suma de las proporciones de cada clase.

- $\mu_{1:2}, \Sigma_{1:2} \rightarrow$  media y covarianza ponderadas de cada clase.

- 4 Repetimos 2 y 3 hasta alcanzar un máximo local,  $k = 1, 2, \dots$

- Es una especie de self-training.



# Clasificación con GMM usando MLE

## Aplicación de EM

- 1 Obtenemos una estimación inicial

$\theta^0 = \{w_1^0, w_2^0, \mu_1^0, \mu_2^0, \Sigma_1^0, \Sigma_2^0\}$  por MLE usando  $(X^I, Y^I)$ .

- 2 Paso E: etiqueta esperada  $p(y|\mathbf{x}, \theta^k) = \frac{p(\mathbf{x}, y|\theta^k)}{\sum_{y'} p(\mathbf{x}, y'|\theta^k)} \forall \mathbf{x} \in X^u$

- $x$  se etiqueta de clase  $c = 1 : 2$  con proporción  $p(y = c|\mathbf{x}, \theta^k)$ .

- 3 Paso M: MLE de  $\theta$  con los ahora datos etiquetados de  $X^u$ .

- $w_{1:2} \rightarrow$  suma de las proporciones de cada clase.

- $\mu_{1:2}, \Sigma_{1:2} \rightarrow$  media y covarianza ponderadas de cada clase.

- 4 Repetimos 2 y 3 hasta alcanzar un máximo local,  $k = 1, 2, \dots$

■ Es una especie de self-training.

# Clasificación con GMM usando MLE

## Aplicación de EM

- 1 Obtenemos una estimación inicial

$\theta^0 = \{w_1^0, w_2^0, \mu_1^0, \mu_2^0, \Sigma_1^0, \Sigma_2^0\}$  por MLE usando  $(X^I, Y^I)$ .

- 2 Paso E: etiqueta esperada  $p(y|\mathbf{x}, \theta^k) = \frac{p(\mathbf{x}, y|\theta^k)}{\sum_{y'} p(\mathbf{x}, y'|\theta^k)} \forall \mathbf{x} \in X^u$

- $x$  se etiqueta de clase  $c = 1 : 2$  con proporción  $p(y = c|\mathbf{x}, \theta^k)$ .

- 3 Paso M: MLE de  $\theta$  con los ahora datos etiquetados de  $X^u$ .

- $w_{1:2} \rightarrow$  suma de las proporciones de cada clase.

- $\mu_{1:2}, \Sigma_{1:2} \rightarrow$  media y covarianza ponderadas de cada clase.

- 4 Repetimos 2 y 3 hasta alcanzar un máximo local,  $k = 1, 2, \dots$

- Es una especie de self-training.

# Pros y contras de los modelos generativos

## Pros

- Es un marco probabilístico claro y bien estudiado.
- Puede ser muy efectivo si el modelo es cercano al correcto.

# Pros y contras de los modelos generativos

## Pros

- Es un marco probabilístico claro y bien estudiado.
- Puede ser muy efectivo si el modelo es cercano al correcto.

## Contras

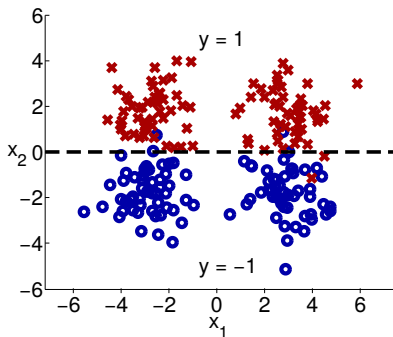
- A menudo es difícil verificar la corrección del modelo.
- La identificabilidad del modelo, no todas las distribuciones lo son (Gauss sí, Bernoulli y uniforme no).
- EM obtiene óptimo locales no necesariamente globales.
- Los datos no etiquetados pueden dañar si el modelo ( $p(y)$  y  $p(\mathbf{x}|y)$ ) no es correcto.

## Ejemplo modelo incorrecto

Clasificación de textos por género y tema, un tema puede estar en varios géneros.

## Ejemplo modelo incorrecto

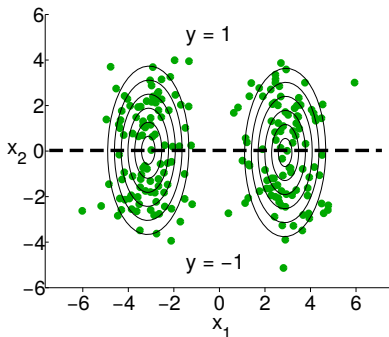
Clasificación de textos por género y tema, un tema puede estar en varios géneros.



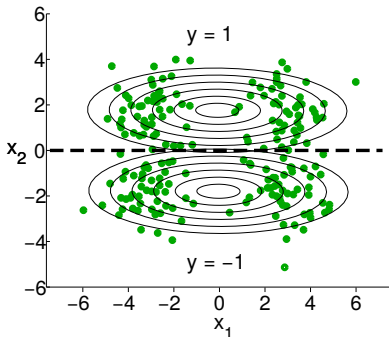
# Ejemplo modelo incorrecto

Clasificación de textos por género y tema, un tema puede estar en varios géneros.

modelo incorrecto, log verosim. -847.93



modelo correcto, log verosim. -921.14



# Ejemplo modelo incorrecto

Clasificación de textos por género y tema, un tema puede estar en varios géneros.

## ■ Posibles soluciones heurísticas:

- Construir cuidadosamente el modelo generativo: p. ej. varias gaussianas por clase en lugar de una.
- Dar menor peso a los datos no etiquetados ( $\lambda < 1$ )

$$\log p(X_l, Y_l, X_u | \theta) = \sum_{i=1}^l \log p(y_i | \theta) p(\mathbf{x}_i | y_i, \theta) +$$

$$\lambda \sum_{i=l+1}^{l+u} \log \left( \sum_{y=1}^2 p(y | \theta) p(\mathbf{x}_i | y, \theta) \right).$$



# Ejemplo modelo incorrecto

Clasificación de textos por género y tema, un tema puede estar en varios géneros.

## ■ Posibles soluciones heurísticas:

- Construir cuidadosamente el modelo generativo: p. ej. varias gaussianas por clase en lugar de una.
- Dar menor peso a los datos no etiquetados ( $\lambda < 1$ )

$$\log p(X_l, Y_l, X_u | \theta) = \sum_{i=1}^l \log p(y_i | \theta) p(\mathbf{x}_i | y_i, \theta) +$$

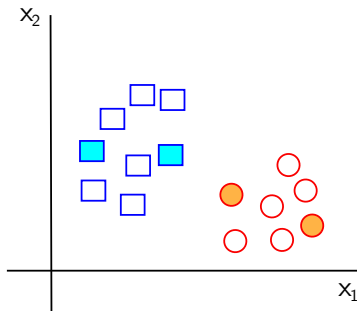
$$\lambda \sum_{i=l+1}^{l+u} \log \left( \sum_{y=1}^2 p(y | \theta) p(\mathbf{x}_i | y, \theta) \right).$$

# Variante: cluster-and-label

- Cluster-and-label es una variante discriminativa, en lugar de usar modelos probabilísticos usa algoritmos de clustering.
- Entradas  $(x_1, y_1), \dots, (x_l, y_l)$ ,  
 $x_{l+1}, \dots, x_{l+u}$ , un alg. de clustering  $\mathcal{A}$   
y un alg. de clasificación SL  $\mathcal{L}$ .
  - Agrupar  $x_1, \dots, x_l, x_{l+1}, \dots, x_{l+u}$  usando  $\mathcal{A}$ .
  - Para cada cluster sea  $S$  sus datos etiquetados.
  - Entrenar un clasificador SL para  $S$ ,  $f_S = \mathcal{L}(S)$ .
  - Aplicar  $f_S$  a no etiquetados de  $S$ , así obtenemos  $y_{l+1}, \dots, y_{l+u}$ .

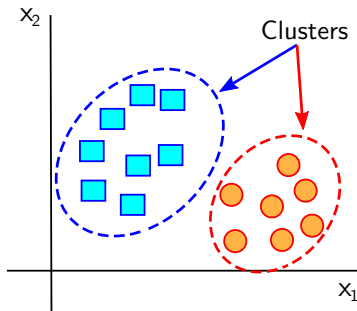
## Variante: cluster-and-label

- Cluster-and-label es una variante discriminativa, en lugar de usar modelos probabilísticos usa algoritmos de clustering.
- Entradas  $(x_1, y_1), \dots, (x_l, y_l)$ ,  $x_{l+1}, \dots, x_{l+u}$ , un alg. de clustering  $\mathcal{A}$  y un alg. de clasificación SL  $\mathcal{L}$ .
  - 1 Agrupar  $x_1, \dots, x_l, x_{l+1}, \dots, x_{l+u}$  usando  $\mathcal{A}$ .
  - 2 Para cada cluster sea  $S$  sus datos etiquetados.
  - 3 Entrenar un clasificador SL para  $S$ ,  $f_S = \mathcal{L}(S)$ .
  - 4 Aplicar  $f_S$  a no etiquetados de  $S$ , así obtenemos  $y_{l+1}, \dots, y_{l+u}$ .



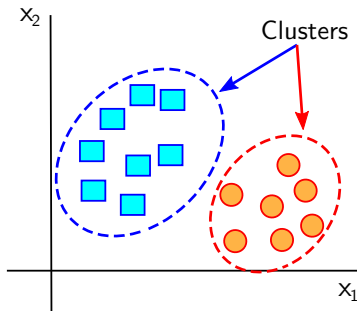
## Variante: cluster-and-label

- Cluster-and-label es una variante discriminativa, en lugar de usar modelos probabilísticos usa algoritmos de clustering.
- Entradas  $(x_1, y_1), \dots, (x_l, y_l)$ ,  $x_{l+1}, \dots, x_{l+u}$ , un alg. de clustering  $\mathcal{A}$  y un alg. de clasificación SL  $\mathcal{L}$ .
  - Agrupar  $x_1, \dots, x_l, x_{l+1}, \dots, x_{l+u}$  usando  $\mathcal{A}$ .
  - Para cada cluster sea  $S$  sus datos etiquetados.
  - Entrenar un clasificador SL para  $S$ ,  $f_S = \mathcal{L}(S)$ .
  - Aplicar  $f_S$  a no etiquetados de  $S$ , así obtenemos  $y_{l+1}, \dots, y_{l+u}$ .



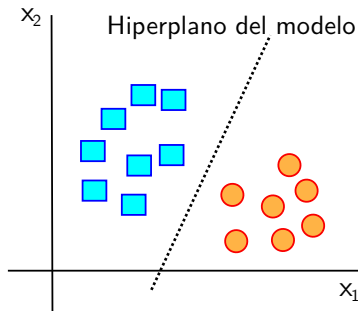
## Variante: cluster-and-label

- Cluster-and-label es una variante discriminativa, en lugar de usar modelos probabilísticos usa algoritmos de clustering.
- Entradas  $(x_1, y_1), \dots, (x_l, y_l)$ ,  $x_{l+1}, \dots, x_{l+u}$ , un alg. de clustering  $\mathcal{A}$  y un alg. de clasificación SL  $\mathcal{L}$ .
  - 1 Agrupar  $x_1, \dots, x_l, x_{l+1}, \dots, x_{l+u}$  usando  $\mathcal{A}$ .
  - 2 Para cada cluster sea  $S$  sus datos etiquetados.
  - 3 Entrenar un clasificador SL para  $S$ ,  $f_S = \mathcal{L}(S)$ .
  - 4 Aplicar  $f_S$  a no etiquetados de  $S$ , así obtenemos  $y_{l+1}, \dots, y_{l+u}$ .



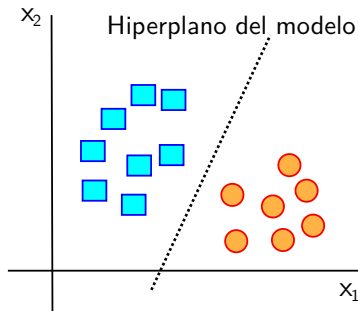
## Variante: cluster-and-label

- Cluster-and-label es una variante discriminativa, en lugar de usar modelos probabilísticos usa algoritmos de clustering.
- Entradas  $(x_1, y_1), \dots, (x_l, y_l)$ ,  $x_{l+1}, \dots, x_{l+u}$ , un alg. de clustering  $\mathcal{A}$  y un alg. de clasificación SL  $\mathcal{L}$ .
  - 1 Agrupar  $x_1, \dots, x_l, x_{l+1}, \dots, x_{l+u}$  usando  $\mathcal{A}$ .
  - 2 Para cada cluster sea  $S$  sus datos etiquetados.
  - 3 Entrenar un clasificador SL para  $S$ ,  $f_S = \mathcal{L}(S)$ .
  - 4 Aplicar  $f_S$  a no etiquetados de  $S$ , así obtenemos  $y_{l+1}, \dots, y_{l+u}$ .



## Variante: cluster-and-label

- Cluster-and-label es una variante discriminativa, en lugar de usar modelos probabilísticos usa algoritmos de clustering.
- Entradas  $(x_1, y_1), \dots, (x_l, y_l)$ ,  $x_{l+1}, \dots, x_{l+u}$ , un alg. de clustering  $\mathcal{A}$  y un alg. de clasificación SL  $\mathcal{L}$ .
  - 1 Agrupar  $x_1, \dots, x_l, x_{l+1}, \dots, x_{l+u}$  usando  $\mathcal{A}$ .
  - 2 Para cada cluster sea  $S$  sus datos etiquetados.
  - 3 Entrenar un clasificador SL para  $S$ ,  $f_S = \mathcal{L}(S)$ .
  - 4 Aplicar  $f_S$  a no etiquetados de  $S$ , así obtenemos  $y_{l+1}, \dots, y_{l+u}$ .



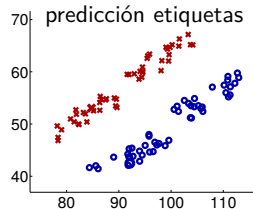
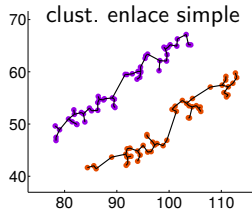
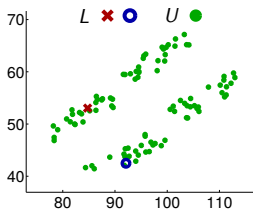
# C-and-L puede funcionar o no

$\mathcal{A}$  clustering jerárquico aglomerativo,  $\mathcal{L}$  voto de mayoría



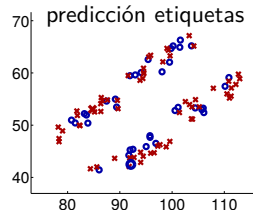
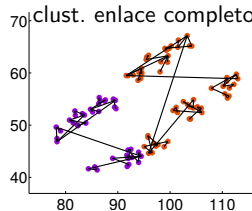
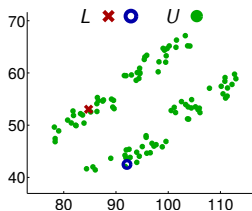
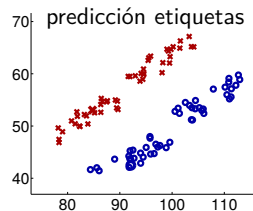
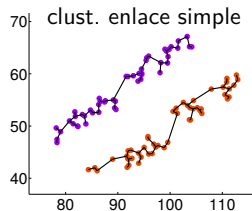
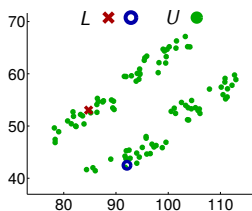
# C-and-L puede funcionar o no

$\mathcal{A}$  clustering jerárquico aglomerativo,  $\mathcal{L}$  voto de mayoría



# C-and-L puede funcionar o no

$\mathcal{A}$  clustering jerárquico aglomerativo,  $\mathcal{L}$  voto de mayoría



# Pros y contras del cluster-and-label

## Pros

- Funciona bastante bien cuando la hipótesis de los clusters se cumple y se elige el algoritmo de clustering apropiado.

# Pros y contras del cluster-and-label

## Pros

- Funciona bastante bien cuando la hipótesis de los clusters se cumple y se elige el algoritmo de clustering apropiado.

## Contras

- Si el algoritmo tiene muchos parámetros puede no ser aplicable en aplicaciones reales.

# Modelos generativos con pomegranate

- Para el ejemplo con pomegranate cargaremos el notebook `pomegranate_example.ipynb`.
- Para un ejemplo más complejo cargaremos el notebook `Tutorial_8_pomegranate_Semisupervised_Learning.ipynb`.

# Modelos generativos con pomegranate

- Para el ejemplo con pomegranate cargaremos el notebook `pomegranate_example.ipynb`.
- Para un ejemplo más complejo cargaremos el notebook `Tutorial_8_pomegranate_Semisupervised_Learning.ipynb`.

# Actividad con pomegranate

## Actividad

Clasificar mediante modelos generativos los datos de titanic.csv en "Survived" o no usando como características "Pclass", "Sex", "Age". Estudiar los datos antes para ver si hay problemas: datos no disponibles, etc. Dar la matriz de confusión.

# Índice

1 Introducción a SSL

2 Herramientas para SSL

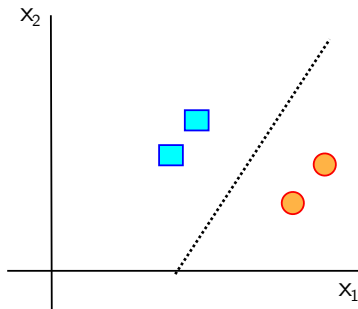
3 Algoritmos para SSL

- Self-training (-learning)
- Co-training y modelos multivista
- Self-labeled
- Modelos generativos
- **Semi-supervised SVM (S3VM)**
- Modelos basados en grafos



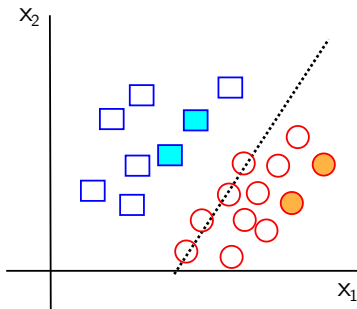
# SVM con datos etiquetados y no etiquetados

- SVM con solo datos etiquetados.
- Añadimos datos no etiquetados.
- S3VM con todos los datos.



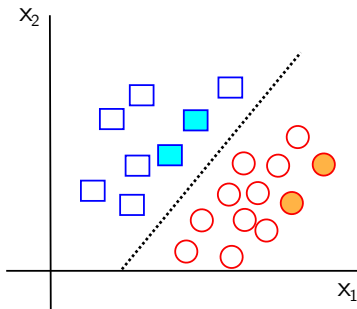
# SVM con datos etiquetados y no etiquetados

- SVM con solo datos etiquetados.
- Añadimos datos no etiquetados.
- S3VM con todos los datos.



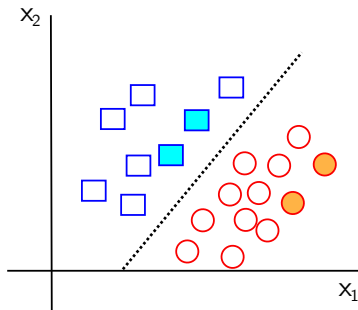
# SVM con datos etiquetados y no etiquetados

- SVM con solo datos etiquetados.
- Añadimos datos no etiquetados.
- S3VM con todos los datos.



# SVM con datos etiquetados y no etiquetados

- SVM con solo datos etiquetados.
- Añadimos datos no etiquetados.
- S3VM con todos los datos.



Hipótesis de separación de baja densidad (de agrupamiento)

Los datos no etiquetados deben estar separados por un margen amplio.

# La idea de S3VM

- Enumerar todos los posibles  $2^u$  etiquetados de  $X^u$ .
- Entrenar un SVM supervisado para cada etiquetado y  $X^l$ .
- Elegir el SMV con margen más amplio.
- Semi-supervised SVMs (S3VMs) = Transductive SVMs (TSVMs)

# La idea de S3VM

- Enumerar todos los posibles  $2^u$  etiquetados de  $X^u$ .
- Entrenar un SVM supervisado para cada etiquetado y  $X^l$ .
- Elegir el SMV con margen más amplio.
- Semi-supervised SVMs (S3VMs) = Transductive SVMs (TSVMs)

# La idea de S3VM

- Enumerar todos los posibles  $2^u$  etiquetados de  $X^u$ .
- Entrenar un SVM supervisado para cada etiquetado y  $X^l$ .
- Elegir el SMV con margen más amplio.
- Semi-supervised SVMs (S3VMs) = Transductive SVMs (TSVMs)

# La idea de S3VM

- Enumerar todos los posibles  $2^u$  etiquetados de  $X^u$ .
- Entrenar un SVM supervisado para cada etiquetado y  $X^l$ .
- Elegir el SMV con margen más amplio.
- Semi-supervised SVMs (S3VMs) = Transductive SVMs (TSVMs)



# SVM supervisado

- $\min_{\mathbf{w}, b, \xi_i} \frac{1}{2} \mathbf{w}^t \mathbf{w} + C \sum \xi_i$ , siendo  $C$  regularización.
- llamando  $f(\mathbf{x}) = \mathbf{w}^t \mathbf{x} + b$ , las restricciones quedan:  
 $y f(\mathbf{x}) \geq 1 - \xi_i$ .
- Las variables de holgura (slack)  $\xi_i \geq 0$ ,  $\xi_i = \max(0, 1 - y_i f(\mathbf{x}_i))$   
hinge loss (función de pérdida)
  - Esta función es convexa, garantiza óptimos globales.
- Clasificamos con  $\text{sign}(f(\mathbf{x}))$ .

# SVM supervisado

- $\min_{\mathbf{w}, b, \xi_i} \frac{1}{2} \mathbf{w}^t \mathbf{w} + C \sum \xi_i$ , siendo  $C$  regularización.
- llamando  $f(\mathbf{x}) = \mathbf{w}^t \mathbf{x} + b$ , las restricciones quedan:  
 $y f(\mathbf{x}) \geq 1 - \xi_i$ .
- Las variables de holgura (slack)  $\xi_i \geq 0$ ,  $\xi_i = \max(0, 1 - y_i f(\mathbf{x}_i))$   
hinge loss (función de pérdida)
  - Esta función es convexa, garantiza óptimos globales.
- Clasificamos con  $\text{sign}(f(\mathbf{x}))$ .

# SVM supervisado

- $\min_{\mathbf{w}, b, \xi_i} \frac{1}{2} \mathbf{w}^t \mathbf{w} + C \sum \xi_i$ , siendo  $C$  regularización.
- llamando  $f(\mathbf{x}) = \mathbf{w}^t \mathbf{x} + b$ , las restricciones quedan:  
 $y f(\mathbf{x}) \geq 1 - \xi_i$ .
- Las variables de holgura (slack)  $\xi_i \geq 0$ ,  $\xi_i = \max(0, 1 - y_i f(\mathbf{x}_i))$   
hinge loss (función de pérdida)
  - Esta función es convexa, garantiza óptimos globales.
- Clasificamos con  $\text{sign}(f(\mathbf{x}))$ .

# SVM supervisado

- $\min_{\mathbf{w}, b, \xi_i} \frac{1}{2} \mathbf{w}^t \mathbf{w} + C \sum \xi_i$ , siendo  $C$  regularización.
- llamando  $f(\mathbf{x}) = \mathbf{w}^t \mathbf{x} + b$ , las restricciones quedan:  
 $y f(\mathbf{x}) \geq 1 - \xi_i$ .
- Las variables de holgura (slack)  $\xi_i \geq 0$ ,  $\xi_i = \max(0, 1 - y_i f(\mathbf{x}_i))$   
hinge loss (función de pérdida)
  - Esta función es convexa, garantiza óptimos globales.
- Clasificamos con  $\text{sign}(f(\mathbf{x}))$ .

# Función objetivo de S3VM

- $\min_{\mathbf{w}, b, \xi_i} \min_{y_u \in \{-1, +1\}^n} \frac{1}{2} \mathbf{w}^t \mathbf{w} + C \sum \xi_i$
- con las restricciones:
  - $y f(\mathbf{x}) \geq 1 - \xi_i \quad i = 1, \dots, l.$
  - $y f(\mathbf{x}) \geq 1 - \xi_i \quad i = l+1, \dots, l+u.$
- Las variables de holgura (slack)
  - $\xi_i \geq 0,$
  - $\xi_i = \max(0, 1 - y_i f(\mathbf{x}_i))$  para  $L$  (hinge loss).
  - $\xi_i = \max(0, 1 - |f(\mathbf{x}_i)|)$  para  $U$  (hat loss).
- Clasificamos con  $\text{sign}(f(\mathbf{x}))$ .

# Función objetivo de S3VM

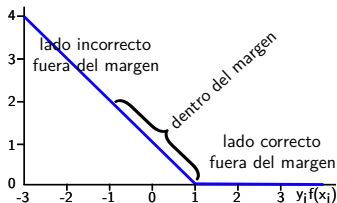
- $\min_{\mathbf{w}, b, \xi_i} \min_{y_u \in \{-1, +1\}^n} \frac{1}{2} \mathbf{w}^\dagger \mathbf{w} + C \sum \xi_i$
- con las restricciones:
  - $yf(\mathbf{x}) \geq 1 - \xi_i \quad i = 1, \dots, l.$
  - $yf(\mathbf{x}) \geq 1 - \xi_i \quad i = l+1, \dots, l+u.$
- Las variables de holgura (slack)
  - $\xi_i \geq 0,$
  - $\xi_i = \max(0, 1 - y_l f(\mathbf{x}_i))$  para  $L$  (hinge loss).
  - $\xi_i = \max(0, 1 - |f(\mathbf{x}_i)|)$  para  $U$  (hat loss).
- Clasificamos con  $\text{sign}(f(\mathbf{x}))$ .

# Función objetivo de S3VM

- $\min_{\mathbf{w}, b, \xi_i} \min_{y_u \in \{-1, +1\}^n} \frac{1}{2} \mathbf{w}^\dagger \mathbf{w} + C \sum \xi_i$
- con las restricciones:
  - $yf(\mathbf{x}) \geq 1 - \xi_i \quad i = 1, \dots, l.$
  - $yf(\mathbf{x}) \geq 1 - \xi_i \quad i = l+1, \dots, l+u.$
- Las variables de holgura (slack)
  - $\xi_i \geq 0,$
  - $\xi_i = \max(0, 1 - y_i f(\mathbf{x}_i))$  para  $L$  (hinge loss).
  - $\xi_i = \max(0, 1 - |f(\mathbf{x}_i)|)$  para  $U$  (hat loss).
- Clasificamos con  $\text{sign}(f(\mathbf{x}))$ .

# Función objetivo de S3VM

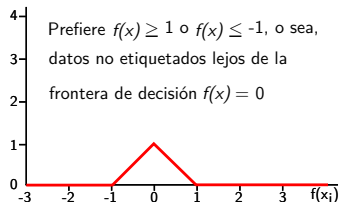
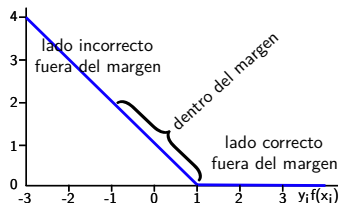
- $\min_{\mathbf{w}, b, \xi_i} \min_{y_u \in \{-1, +1\}^n} \frac{1}{2} \mathbf{w}^t \mathbf{w} + C \sum \xi_i$
- con las restricciones:
  - $y f(\mathbf{x}) \geq 1 - \xi_i \quad i = 1, \dots, l.$
  - $y f(\mathbf{x}) \geq 1 - \xi_i \quad i = l+1, \dots, l+u.$
- Las variables de holgura (slack)
  - $\xi_i \geq 0,$
  - $\xi_i = \max(0, 1 - y_i f(\mathbf{x}_i))$  para  $L$  (hinge loss).
  - $\xi_i = \max(0, 1 - |f(\mathbf{x}_i)|)$  para  $U$  (hat loss).
- Clasificamos con  $\text{sign}(f(\mathbf{x}))$ .





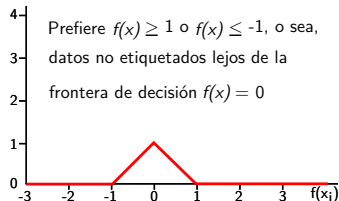
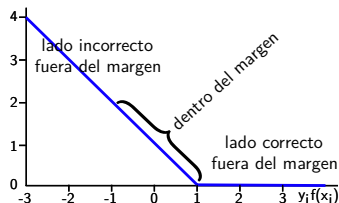
# Función objetivo de S3VM

- $\min_{\mathbf{w}, b, \xi_i} \min_{y_u \in \{-1, +1\}^n} \frac{1}{2} \mathbf{w}^t \mathbf{w} + C \sum \xi_i$
- con las restricciones:
  - $y f(\mathbf{x}) \geq 1 - \xi_i \quad i = 1, \dots, l.$
  - $y f(\mathbf{x}) \geq 1 - \xi_i \quad i = l+1, \dots, l+u.$
- Las variables de holgura (slack)
  - $\xi_i \geq 0,$
  - $\xi_i = \max(0, 1 - y_i f(\mathbf{x}_i))$  para  $L$  (hinge loss).
  - $\xi_i = \max(0, 1 - |f(\mathbf{x}_i)|)$  para  $U$  (hat loss).
- Clasificamos con  $\text{sign}(f(\mathbf{x}))$ .



# Función objetivo de S3VM

- $\min_{\mathbf{w}, b, \xi_i} \min_{y_u \in \{-1, +1\}^n} \frac{1}{2} \mathbf{w}^t \mathbf{w} + C \sum \xi_i$
- con las restricciones:
  - $yf(\mathbf{x}) \geq 1 - \xi_i \quad i = 1, \dots, l.$
  - $yf(\mathbf{x}) \geq 1 - \xi_i \quad i = l+1, \dots, l+u.$
- Las variables de holgura (slack)
  - $\xi_i \geq 0,$
  - $\xi_i = \max(0, 1 - y_i f(\mathbf{x}_i))$  para  $L$  (hinge loss).
  - $\xi_i = \max(0, 1 - |f(\mathbf{x}_i)|)$  para  $U$  (hat loss).
- Clasificamos con  $\text{sign}(f(\mathbf{x}))$ .



# Función objetivo de S3VM escrita sin restricciones

## Función objetivo

$$\min_f \frac{1}{2} \mathbf{w}^t \mathbf{w} + C_l \sum_{i=1}^l \max(0, 1 - y_i f(x_i)) + C_u \sum_{i=l+1}^{l+u} \max(0, 1 - |f(x_i)|)$$

# Función objetivo de S3VM escrita sin restricciones

## Función objetivo

$$\min_f \frac{1}{2} \mathbf{w}^t \mathbf{w} + C_l \sum_{i=1}^l \max(0, 1 - y_i f(x_i)) + C_u \sum_{i=l+1}^{l+u} \max(0, 1 - |f(x_i)|)$$

- La función de pérdida para no etiquetados no es convexa, luego no proporciona óptimos globales.
- Habrá que diseñar métodos de optimización especiales.

# Métodos de optimización para S3VM

- Label-switch-retraining (S3VM<sup>light</sup>).
- Gradient descent ( $\nabla$ S3VM).
- Continuation (cS3VM).
- Concave-convex procedure (CCCP).

*Optimization Techniques for Semi-Supervised Support Vector Machines [Chapelle et al., 2008]*

*An overview on semi-supervised support vector machine [Ding et al., 2015]*

# Restricción de clases equilibradas

- La optimización directa de S3VM suele producir clasificaciones desequilibradas, más datos de los que debería en un clase.
- Solución: introducir una restricción en el problema a optimizar:

- Equilibrio heurístico  $\frac{1}{u} \sum_{i=l+1}^{l+u} y_i = \frac{1}{l} \sum_{i=1}^l y_i.$

- Equilibrio de clase nivelada  $\frac{1}{u} \sum_{i=l+1}^{l+u} f(x_i) = \frac{1}{l} \sum_{i=1}^l y_i$ , preferible porque  $f$  es un número real y es más fácil de optimizar.

# Restricción de clases equilibradas

- La optimización directa de S3VM suele producir clasificaciones desequilibradas, más datos de los que debería en un clase.
- Solución: introducir una restricción en el problema a optimizar:

- Equilibrio heurístico  $\frac{1}{u} \sum_{i=l+1}^{l+u} y_i = \frac{1}{l} \sum_{i=1}^l y_i.$

- Equilibrio de clase nivelada  $\frac{1}{u} \sum_{i=l+1}^{l+u} f(x_i) = \frac{1}{l} \sum_{i=1}^l y_i$ , preferible porque  $f$  es un número real y es más fácil de optimizar.

# S3VM<sup>light</sup>

- Entrenar un SVM con los datos etiquetados.
- Clasificar los datos no etiquetados.
- Bucle externo: empezar con valores pequeños de  $C_u$  e ir aumentando.
  - Obtener nuevo clasificador usando hinge loss para los dos tipos de datos en la función a optimizar.
  - Bucle interno:
    - Los ejemplos son incompatibles si son distintos y el clasificador da valores de función de pérdida diferentes.
    - Incompatibles  $u, v$  si  $u \neq v$  y  $f(u) \neq f(v)$ .
    - Incompatibles  $u, u$  si  $f(u) > 0$ .



# S3VM<sup>light</sup>

- Entrenar un SVM con los datos etiquetados.
- Clasificar los datos no etiquetados.
- Bucle externo: empezar con valores pequeños de  $C_u$  e ir aumentando.
  - Obtener nuevo clasificador usando hinge loss para los dos tipos de datos en la función a optimizar.
  - Bucle interno:
    - Los ejemplos son interconectados si son distintos y el coeficiente  $C_u$  es mayor que el valor de pérdida compartido.
    - Interconectar  $u_i$  y  $u_j$  si  $C_u$  es mayor que  $\frac{1}{2}(\ell_i + \ell_j)$ .

# S3VM<sup>light</sup>

- Entrenar un SVM con los datos etiquetados.
- Clasificar los datos no etiquetados.
- Bucle externo: empezar con valores pequeños de  $C_u$  e ir aumentando.
  - Obtener nuevo clasificador usando hinge loss para los dos tipos de datos en la función a optimizar.
  - Bucle interno:
    - Dos etiquetas son intercambiables si son distintas y al cambiar sus valores la función de pérdida disminuye.
    - Intercambiar etiquetas hasta que no queden etiquetas intercambiables.

# S3VM<sup>light</sup>

- Entrenar un SVM con los datos etiquetados.
- Clasificar los datos no etiquetados.
- Bucle externo: empezar con valores pequeños de  $C_u$  e ir aumentando.
  - Obtener nuevo clasificador usando hinge loss para los dos tipos de datos en la función a optimizar.
  - Bucle interno:
    - Dos etiquetas son intercambiables si son distintas y al cambiar sus valores la función de pérdida disminuye.
    - Intercambiar etiquetas hasta que no queden etiquetas intercambiables.

# S3VM<sup>light</sup>

- Entrenar un SVM con los datos etiquetados.
- Clasificar los datos no etiquetados.
- Bucle externo: empezar con valores pequeños de  $C_u$  e ir aumentando.
  - Obtener nuevo clasificador usando hinge loss para los dos tipos de datos en la función a optimizar.
  - Bucle interno:
    - Dos etiquetas son intercambiables si son distintas y al cambiar sus valores la función de pérdida disminuye.
    - Intercambiar etiquetas hasta que no queden etiquetas intercambiables.

# S3VM<sup>light</sup>

- Entrenar un SVM con los datos etiquetados.
- Clasificar los datos no etiquetados.
- Bucle externo: empezar con valores pequeños de  $C_u$  e ir aumentando.
  - Obtener nuevo clasificador usando hinge loss para los dos tipos de datos en la función a optimizar.
  - Bucle interno:
    - Dos etiquetas son intercambiables si son distintas y al cambiar sus valores la función de pérdida disminuye.
    - Intercambiar etiquetas hasta que no queden etiquetas intercambiables.

# $\nabla$ S3VM

- Similar a S3VM<sup>light</sup> pero solo con el bucle externo.
- Usa  $\exp(-5x^2)$  (es diferenciable) en lugar de hat loss para los datos no etiquetados en la función a optimizar.
- Hace  $b = \frac{1}{I} \sum_{i=1}^I y_i$ , equilibra las clases automáticamente.

# ▽S3VM

- Similar a S3VM<sup>light</sup> pero solo con el bucle externo.
- Usa  $\exp(-5x^2)$  (es diferenciable) en lugar de hat loss para los datos no etiquetados en la función a optimizar.
- Hace  $b = \frac{1}{I} \sum_{i=1}^I y_i$ , equilibra las clases automáticamente.

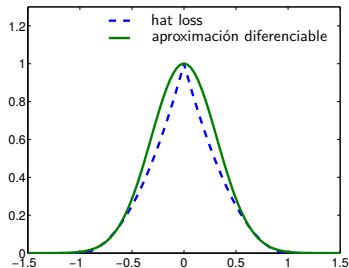
# ▽S3VM

- Similar a S3VM<sup>light</sup> pero solo con el bucle externo.
- Usa  $\exp(-5x^2)$  (es diferenciable) en lugar de hat loss para los datos no etiquetados en la función a optimizar.
- Hace  $b = \frac{1}{l} \sum_{i=1}^l y_i$ , equilibra las clases automáticamente.



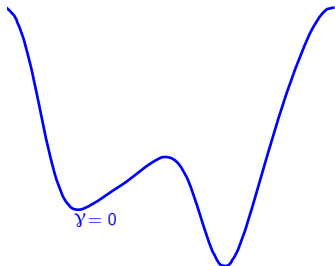
# $\nabla$ S3VM

- Similar a  $S3VM^{light}$  pero solo con el bucle externo.
- Usa  $\exp(-5x^2)$  (es diferenciable) en lugar de hat loss para los datos no etiquetados en la función a optimizar.
- Hace  $b = \frac{1}{I} \sum_{i=1}^I y_i$ , equilibra las clases automáticamente.



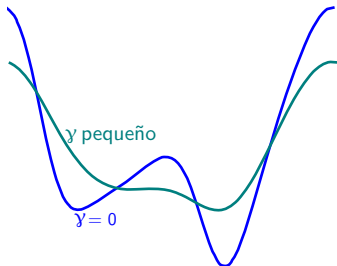
# cS3VM

- Similar a  $\nabla S3VM$ , el bucle externo no varía  $C_u$ , sino suaviza la función objetivo por convolución con una gaussiana de varianza  $\gamma$ .
- Se va aumentando  $\gamma$ .
- La suavización llega hasta una función convexa, ¿ $\gamma$ ?
- A la que se le halla el óptimo global.
- Recorriendo el camino inverso, se llega al óptimo global de la función original.



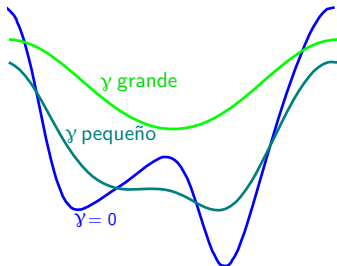
# cS3VM

- Similar a  $\nabla S3VM$ , el bucle externo no varía  $C_u$ , sino suaviza la función objetivo por convolución con una gaussiana de varianza  $\gamma$ .
- Se va aumentando  $\gamma$ .
- La suavización llega hasta una función convexa, ¿ $\gamma$ ?
- A la que se le halla el óptimo global.
- Recorriendo el camino inverso,
- se llega al óptimo global de la función original.



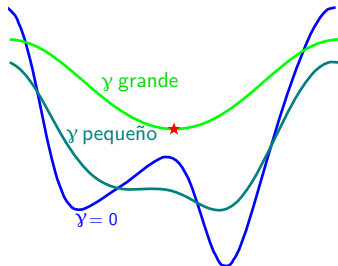
# cS3VM

- Similar a  $\nabla S3VM$ , el bucle externo no varía  $C_u$ , sino suaviza la función objetivo por convolución con una gaussiana de varianza  $\gamma$ .
- Se va aumentando  $\gamma$ .
- La suavización llega hasta una función convexa, ¿ $\gamma$ ?
- A la que se le halla el óptimo global.
- Recorriendo el camino inverso,
- se llega al óptimo global de la función original.



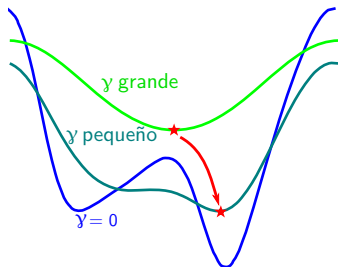
# cS3VM

- Similar a  $\nabla S3VM$ , el bucle externo no varía  $C_U$ , sino suaviza la función objetivo por convolución con una gaussiana de varianza  $\gamma$ .
- Se va aumentando  $\gamma$ .
- La suavización llega hasta una función convexa, ¿ $\gamma$ ?
- A la que se le halla el óptimo global.
- Recorriendo el camino inverso,
- se llega al óptimo global de la función original.



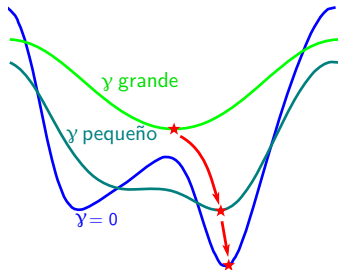
# cS3VM

- Similar a  $\nabla S3VM$ , el bucle externo no varía  $C_u$ , sino suaviza la función objetivo por convolución con una gaussiana de varianza  $\gamma$ .
- Se va aumentando  $\gamma$ .
- La suavización llega hasta una función convexa, ¿ $\gamma$ ?
- A la que se le halla el óptimo global.
- Recorriendo el camino inverso,
- se llega al óptimo global de la función original.



# cS3VM

- Similar a  $\nabla S3VM$ , el bucle externo no varía  $C_u$ , sino suaviza la función objetivo por convolución con una gaussiana de varianza  $\gamma$ .
- Se va aumentando  $\gamma$ .
- La suavización llega hasta una función convexa, ¿ $\gamma$ ?
- A la que se le halla el óptimo global.
- Recorriendo el camino inverso,
- se llega al óptimo global de la función original.



# CCCP

- SVM con los datos etiquetados.
- Se clasifican los datos no etiquetados.
- $\max(0, 1 - |f(x_i)|) =$   
 $\max(0, |f(x_i)| - 1) + (1 - |f(x_i)|)$
- La parte cóncava de la función objetivo se aproxima por una tangente.
- La función objetivo transformada es convexa, se halla su mínimo.
- Se va iterando hasta llegar al mínimo global de la función objetivo original.

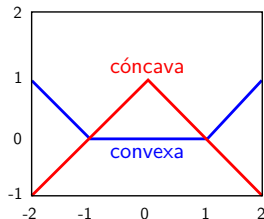
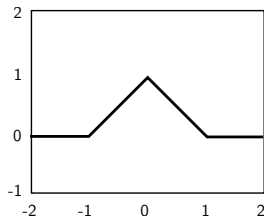


# CCCP

- SVM con los datos etiquetados.
- Se clasifican los datos no etiquetados.
- $\max(0, 1 - |f(x_i)|) =$   
 $\max(0, |f(x_i)| - 1) + (1 - |f(x_i)|)$
- La parte cóncava de la función objetivo se aproxima por una tangente.
- La función objetivo transformada es convexa, se halla su mínimo.
- Se va iterando hasta llegar al mínimo global de la función objetivo original.

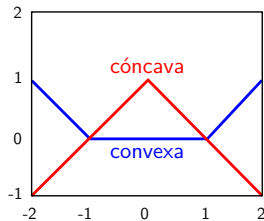
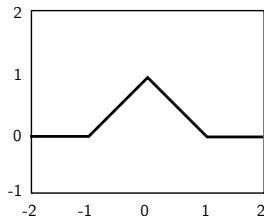
# CCCP

- SVM con los datos etiquetados.
- Se clasifican los datos no etiquetados.
- $\max(0, 1 - |f(x_i)|) =$   
 $\max(0, |f(x_i)| - 1) + (1 - |f(x_i)|)$
- La parte cóncava de la función objetivo se aproxima por una tangente.
- La función objetivo transformada es convexa, se halla su mínimo.
- Se va iterando hasta llegar al mínimo global de la función objetivo original.



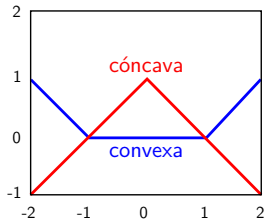
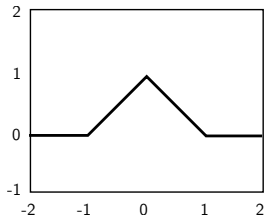
# CCCP

- SVM con los datos etiquetados.
- Se clasifican los datos no etiquetados.
- $\max(0, 1 - |f(x_i)|) =$   
 $\max(0, |f(x_i)| - 1) + (1 - |f(x_i)|)$
- La parte cóncava de la función objetivo se aproxima por una tangente.
- La función objetivo transformada es convexa, se halla su mínimo.
- Se va iterando hasta llegar al mínimo global de la función objetivo original.



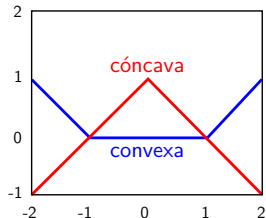
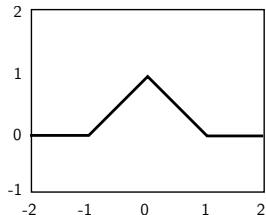
# CCCP

- SVM con los datos etiquetados.
- Se clasifican los datos no etiquetados.
- $\max(0, 1 - |f(x_i)|) =$   
 $\max(0, |f(x_i)| - 1) + (1 - |f(x_i)|)$
- La parte cóncava de la función objetivo se aproxima por una tangente.
- La función objetivo transformada es convexa, se halla su mínimo.
- Se va iterando hasta llegar al mínimo global de la función objetivo original.



# CCCP

- SVM con los datos etiquetados.
- Se clasifican los datos no etiquetados.
- $\max(0, 1 - |f(x_i)|) =$   
 $\max(0, |f(x_i)| - 1) + (1 - |f(x_i)|)$
- La parte cóncava de la función objetivo se aproxima por una tangente.
- La función objetivo transformada es convexa, se halla su mínimo.
- Se va iterando hasta llegar al mínimo global de la función objetivo original.



# Pros y contras de S3VM

## Pros

- Aplicable donde SVM lo es.
- Dispone de un marco matemático claro.

# Pros y contras de S3VM

## Pros

- Aplicable donde SVM lo es.
- Dispone de un marco matemático claro.

## Contras

- Difícil de optimizar.
- Puede quedar atrapado en óptimos locales.
- Resultados potencialmente peores que otros métodos.

# S3VM con semisup-learn

- Para un ejemplo con semisup-learn cargaremos el notebook `compare_linsvm_methods.ipynb`.



# S3VM con RSSL

- Para un ejemplo con RSSL cargaremos el notebook `example-S4VM.Rmd`.
- Para otro ejemplo con RSSL cargaremos el notebook `example-TSVM.Rmd`.

# S3VM con RSSL

- Para un ejemplo con RSSL cargaremos el notebook `example-S4VM.Rmd`.
- Para otro ejemplo con RSSL cargaremos el notebook `example-TSVM.Rmd`.

# Actividades de S3VM

## Actividad

Clasificar mediante S3VM los datos de titanic.csv en "Survived" o no usando como características "Pclass", "Sex", "Age". Estudiar los datos antes para ver si hay problemas: datos no disponibles, etc. Dar la matriz de confusión. Usar varios de los módulos comentados.

# Índice

1 Introducción a SSL

2 Herramientas para SSL

3 Algoritmos para SSL

- Self-training (-learning)
- Co-training y modelos multivista
- Self-labeled
- Modelos generativos
- Semi-supervised SVM (S3VM)
- Modelos basados en grafos

# Introducción

- Construir un grafo con vértices representando a los datos y aristas representando la similitud entre los datos.
- Buscar técnicas para recortar el grafo:
  - Datos etiquetados
  - Heurísticas, p. ej. corte mínimo.

## Hipótesis

Vértices unido por una arista de peso alto pertenecerán a la misma clase.

# Introducción

- Construir un grafo con vértices representando a los datos y aristas representando la similitud entre los datos.
- Buscar técnicas para recortar el grafo:
  - Datos etiquetados
  - Heurísticas, p. ej. corte mínimo.

## Hipótesis

Vértices unido por una arista de peso alto pertenecerán a la misma clase.

# Construcción del grafo

## Construcción del grafo

- $\mathcal{G} = \langle \mathcal{V}, \mathcal{A} \rangle$  donde  $\mathcal{V} = \{\mathbf{x}_i\}_{i=1}^{l+u}$ .
- Conectar los vértices usando una heurística:
  - $\epsilon$ -NN  $\epsilon > 0$ :  $\mathbf{x}_i$  y  $\mathbf{x}_j$  están conectados si  $\text{dist}(\mathbf{x}_i, \mathbf{x}_j) < \epsilon$ .
  - $k$ -NN:  $\mathbf{x}_i$  y  $\mathbf{x}_j$  están conectados si  $\mathbf{x}_j$  es uno de los  $k$ -NN de  $\mathbf{x}_i$ .
- Ponderación del grafo:
  - Ingenua: si  $\mathbf{x}_i$  y  $\mathbf{x}_j$  están conectados, el peso  $w_{ij} = 1$ .
  - Kernel gaussiano si  $\mathbf{x}_i$  y  $\mathbf{x}_j$  están conectados, el peso

$$w_{ij} = \exp\left(-\frac{\text{dist}^2(\mathbf{x}_i, \mathbf{x}_j)}{\sigma^2}\right).$$

# Construcción del grafo

## Construcción del grafo

- $\mathcal{G} = \langle \mathcal{V}, \mathcal{A} \rangle$  donde  $\mathcal{V} = \{\mathbf{x}_i\}_{i=1}^{l+u}$ .
- Conectar los vértices usando una heurística:
  - $\varepsilon$ -NN  $\varepsilon > 0$ :  $\mathbf{x}_i$  y  $\mathbf{x}_j$  están conectados si  $\text{dist}(\mathbf{x}_i, \mathbf{x}_j) < \varepsilon$ .
  - $k$ -NN:  $\mathbf{x}_i$  y  $\mathbf{x}_j$  están conectados si  $\mathbf{x}_j$  es uno de los  $k$ -NN de  $\mathbf{x}_i$ .
- Ponderación del grafo:
  - Ingenua: si  $\mathbf{x}_i$  y  $\mathbf{x}_j$  están conectados, el peso  $w_{ij} = 1$ .
  - Kernel gaussiano si  $\mathbf{x}_i$  y  $\mathbf{x}_j$  están conectados, el peso
 
$$w_{ij} = \exp\left(-\frac{\text{dist}^2(\mathbf{x}_i, \mathbf{x}_j)}{\sigma^2}\right).$$



# Construcción del grafo

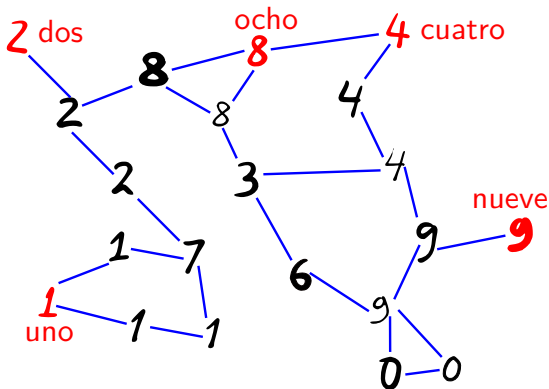
## Construcción del grafo

- $\mathcal{G} = \langle \mathcal{V}, \mathcal{A} \rangle$  donde  $\mathcal{V} = \{\mathbf{x}_i\}_{i=1}^{l+u}$ .
- Conectar los vértices usando una heurística:
  - $\varepsilon$ -NN  $\varepsilon > 0$ :  $\mathbf{x}_i$  y  $\mathbf{x}_j$  están conectados si  $\text{dist}(\mathbf{x}_i, \mathbf{x}_j) < \varepsilon$ .
  - $k$ -NN:  $\mathbf{x}_i$  y  $\mathbf{x}_j$  están conectados si  $\mathbf{x}_j$  es uno de los  $k$ -NN de  $\mathbf{x}_i$ .
- Ponderación del grafo:
  - Ingenua: si  $\mathbf{x}_i$  y  $\mathbf{x}_j$  están conectados, el peso  $w_{ij} = 1$ .
  - Kernel gaussiano si  $\mathbf{x}_i$  y  $\mathbf{x}_j$  están conectados, el peso

$$w_{ij} = \exp\left(-\frac{\text{dist}^2(\mathbf{x}_i, \mathbf{x}_j)}{\sigma^2}\right).$$

# Ejemplo de grafo

Utilizamos la distancia entre píxeles.



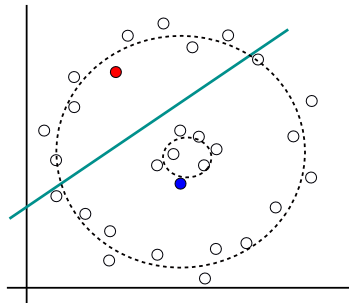
# Algoritmos basados en grafos

- Propagación de etiquetas.
- Partición del grafo
  - Corte mínimo.
  - Función armónica.
  - Regularización de variedades.
  - Consistencia local y global.

*Robust and Scalable Graph-Based Semisupervised Learning [Liu et al. 2012]*

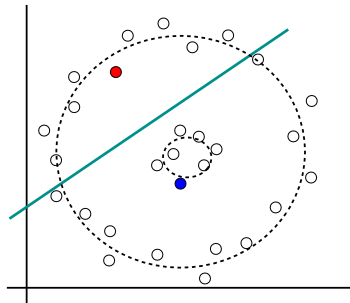
# Propagación de etiquetas

- Entrenamiento SVM: no considerar la distribución de los datos.
- Para incluir los no etiquetados en la predicción de etiquetas:
  - Conectar los puntos cercanos entre sí.
  - Propagar las etiquetas por los nodos conectados...
  - hasta completar el grafo.



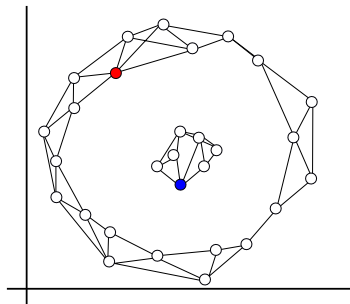
# Propagación de etiquetas

- Entrenamiento SVM: no considerar la distribución de los datos.
- Para incluir los no etiquetados en la predicción de etiquetas:
  - Conectar los puntos cercanos entre sí.
  - Propagar las etiquetas por los nodos conectados...
  - hasta completar el grafo.



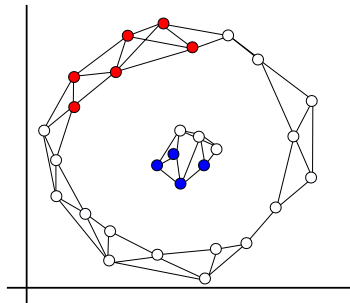
# Propagación de etiquetas

- Entrenamiento SVM: no considerar la distribución de los datos.
- Para incluir los no etiquetados en la predicción de etiquetas:
  - Conectar los puntos cercanos entre sí.
  - Propagar las etiquetas por los nodos conectados...
  - hasta completar el grafo.



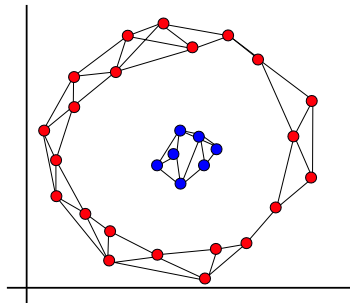
# Propagación de etiquetas

- Entrenamiento SVM: no considerar la distribución de los datos.
- Para incluir los no etiquetados en la predicción de etiquetas:
  - Conectar los puntos cercanos entre sí.
  - Propagar las etiquetas por los nodos conectados...
  - hasta completar el grafo.



# Propagación de etiquetas

- Entrenamiento SVM: no considerar la distribución de los datos.
- Para incluir los no etiquetados en la predicción de etiquetas:
  - Conectar los puntos cercanos entre sí.
  - Propagar las etiquetas por los nodos conectados...
  - hasta completar el grafo.



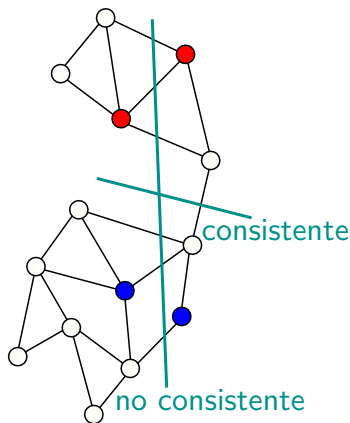


# Partición del grafo

- La idea es que la clasificación es una partición del grafo.
- Buscar una frontera de clasificación:
  - Consistente con los datos etiquetados
  - Particiones el grafo con cortes pequeños.
- Los distintos métodos varían en la forma de definir la función de corte.

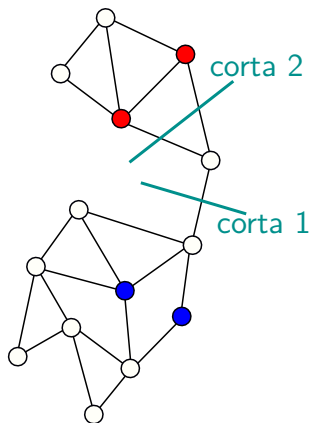
# Partición del grafo

- La idea es que la clasificación es una partición del grafo.
- Buscar una frontera de clasificación:
  - Consistente con los datos etiquetados
  - Particiones el grafo con cortes pequeños.
- Los distintos métodos varían en la forma de definir la función de corte.



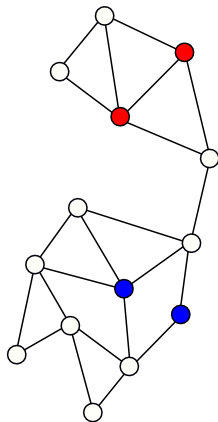
# Partición del grafo

- La idea es que la clasificación es una partición del grafo.
- Buscar una frontera de clasificación:
  - Consistente con los datos etiquetados
  - Particiones el grafo con cortes pequeños.
- Los distintos métodos varían en la forma de definir la función de corte.



# Partición del grafo

- La idea es que la clasificación es una partición del grafo.
- Buscar una frontera de clasificación:
  - Consistente con los datos etiquetados
  - Particiones el grafo con cortes pequeños.
- Los distintos métodos varían en la forma de definir la función de corte.



# Pros y contras de métodos basados en grafos

## Pros

- Dispone de un marco matemático claro.
- El rendimiento es bueno si el grafo se ajusta a la tarea.
- Se puede extender a grafos dirigidos.

# Pros y contras de métodos basados en grafos

## Pros

- Dispone de un marco matemático claro.
- El rendimiento es bueno si el grafo se ajusta a la tarea.
- Se puede extender a grafos dirigidos.

## Contras

- El rendimiento es malo si el grafo lo es.
- Sensible a la estructura del grafo y a los pesos.

# Clasificación mediante grafos con sklearn

- Para un ejemplo con LabelPropagation cargaremos el notebook `plot_label_propagation_structure.ipynb`.
- Para un ejemplo con LabelSpreading cargaremos el notebook `plot_label_propagation_digits.ipynb`.

# Clasificación mediante grafos con sklearn

- Para un ejemplo con LabelPropagation cargaremos el notebook `plot_label_propagation_structure.ipynb`.
- Para un ejemplo con LabelSpreading cargaremos el notebook `plot_label_propagation_digits.ipynb`.



# Actividad con sklearn

## Actividad

Clasificar mediante sklearn los datos de titanic.csv en "Survived" o no usando como características "Pclass", "Sex", "Age". Estudiar los datos antes para ver si hay problemas: datos no disponibles, etc. Dar la matriz de confusión. Usar varios de los módulos comentados.

# Clasificación mediante grafos con Weka

Explicar el uso de Weka en aprendizaje semi-supervisado.

# Actividad listado métodos de SSI y RSSL

## Actividad

Leer los manuales SLL.pdf y RSSL.pdf y clasificar los métodos que aparecen en las categoría que se han explicado.

# Bibliografía I

- Semi-Supervised Learning, Olivier Chapelle; Bernhard Schölkopf; Alexander Zien. MIT-Press
- Semi-Supervised Learning Literature Survey, Xiaojin Zhu
- Hands-on Machine Learning with Scikit-Learn and TensorFlow, Aurélien Géron
- Python: Deeper Insights into Machine Learning, Sebastian Raschka; David Julian; John Hearty
- Scikit-learn : Machine Learning Simplified, Raúl Garreta; Guillermo Moncecchi; Trent Hauck; Gavin Hackeling
- Python: Real World Machine Learning, Prateek Joshi; John Hearty; Bastiaan Sjardin; Luca Massaron; Alberto Boschetti

## Bibliografía II

- Machine Learning: End-to-End guide for Java developers, Richard M. Reese; Jennifer L. Reese; Boštjan Kaluža; Dr. Uday Kamath; Krishna Choppella
- Robust and Scalable Graph-Based Semisupervised Learning, Wei Liu; Jun Wang
- RSSL: Semi-supervised Learning in R, Jesse H. Krijthe
- Self-labeled techniques for semi-supervised learning: taxonomy, software and empirical study, Isaac Triguero; Salvador García; Francisco Herrera

# Thank You



Aprendizaje semi-supervisado (SSL)  
José Manuel Cuadra Troncoso