



IMT2200 Introducción a la Ciencia de Datos



PONTIFICIA
UNIVERSIDAD
CATÓLICA
DE CHILE

Rodrigo A. Carrasco
Instituto de Ingeniería Matemática y Computacional
Escuela de Ingeniería

-  Rodrigo_carrasco2
 @_rax
 @rodrigo_a_Carrasco
 www.raxlab.science



Avisos Importantes

- **Tarea 4**

- Les recuerdo que la Tarea 4 se debe entregar antes de este domingo a las 23:59

- **Actividad 10**

- El jueves tendremos la Actividad 10, que será un preparativo para la parte de ML de la I2.

- **Interrogación 2**

- El próximo viernes 14 es la I2 a las 17:30 en la sala C201.
 - Incluye todo el material hasta la clase de este jueves 6 (clase 23 en Canvas). El foco estará en las clases de 12 a 23.
 - Al igual que para la I1, habrá un conjunto de preguntas sobre Armas de Destrucción Matemática (capítulos 4 a 6).
 - La parte de conceptos y el libro serán sin apuntes; para la de desarrollo podrán usar todo el material visto en clases.





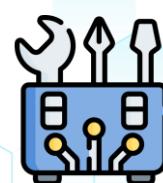
01

Repaso



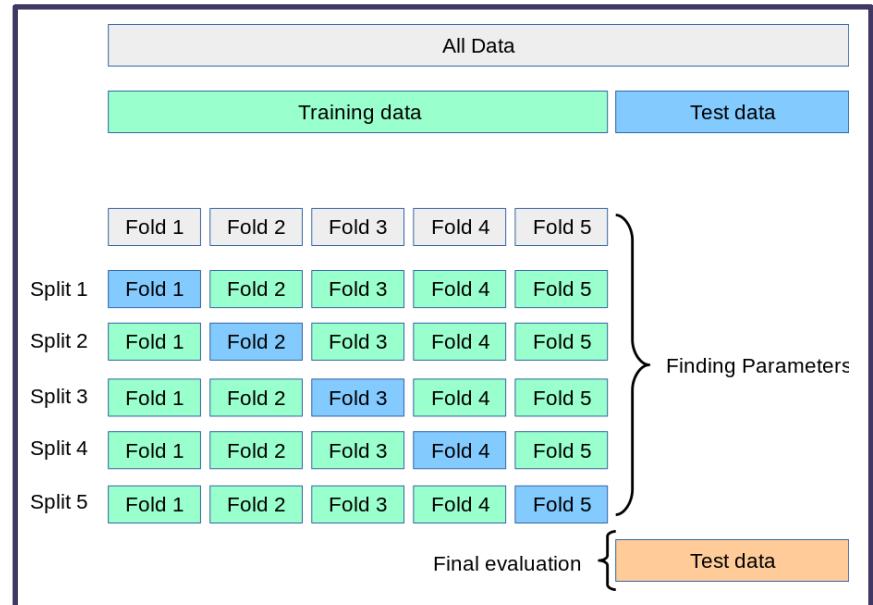
Temas vistos la clase pasada

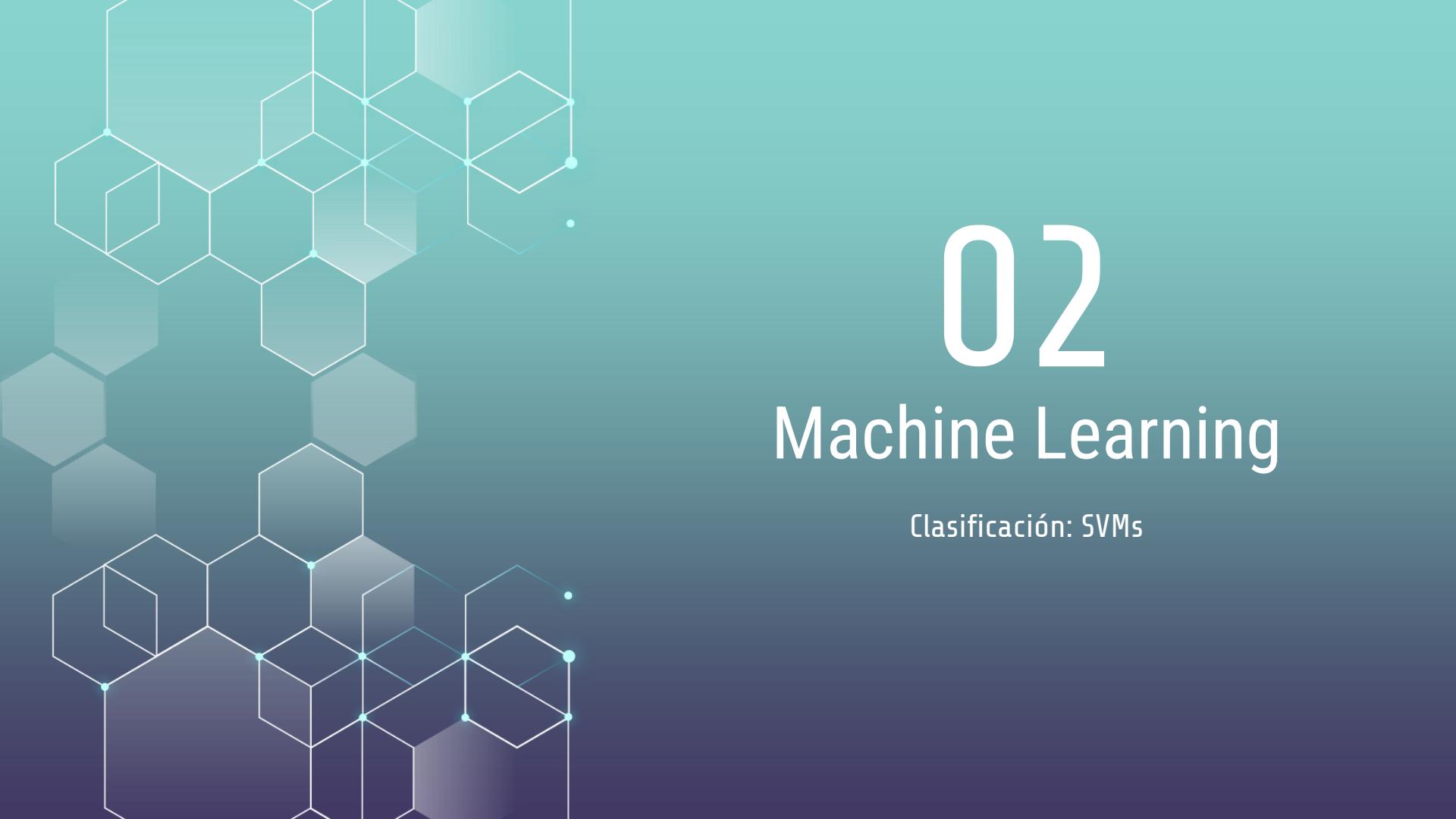
- Estuvimos estudiando un nuevo tema en aprendizaje supervisado: clasificación.
- ¿Cuál es la diferencia entre clasificación y regresión?
- Revisamos diferentes algoritmos para clasificar: kNN y árboles de decisión.
- Aprendizaje clave: los beneficios y complejidades de cada algoritmo.
- Aprendimos, además, cuáles son las métricas relevantes para evaluar un sistema de clasificación.



Cross validation 2.0 (train-validation-test)

- Usar `x_train`, `y_train` para `GridSearchCV`
- Usar `x_test`, `y_test` para reportar rendimiento del modelo





02

Machine Learning

Clasificación: SVMs

Separando con un plano

- Otra forma de clasificar, cuando hay sólo 2 categorías, es buscando una función que separe ambas clases.

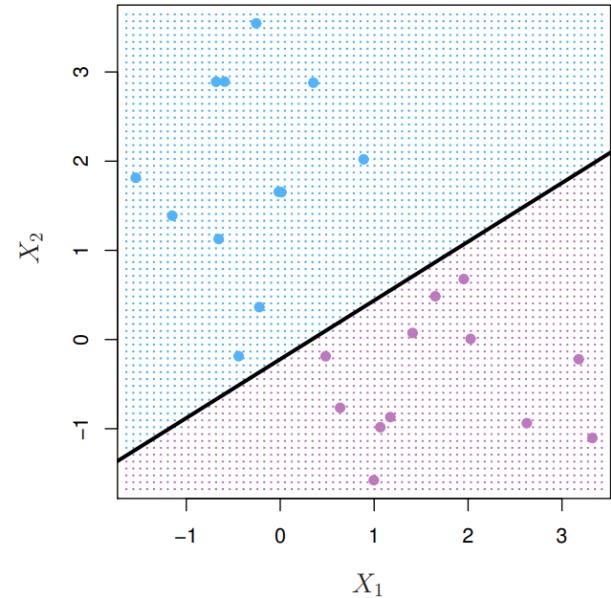
- Por ejemplo, consideremos un hiperplano:

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p = 0$$

- ¿Qué pasa con los puntos a un lado y al otro?

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p > 0$$

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p < 0$$





Clasificación con un hiperplano

- Considere que tiene N muestras de virus que pueden pertenecer a dos tipos de enfermedades.
 - De cada muestra podemos obtener un conjunto de datos (o características) x_{1t}, \dots, x_{nt} y un experto nos permitió clasificar cada muestra identificando con $y_i = 1$ o $y_i = -1$.
 - Queremos crear un clasificador que nos permita identificar a futuro qué tipo de virus será una nueva muestra revisando las características del mismo.
- 



Buscando el hiperplano

- Asociemos las etiquetas de forma de que:

$$\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip} > 0 \text{ if } y_i = 1$$

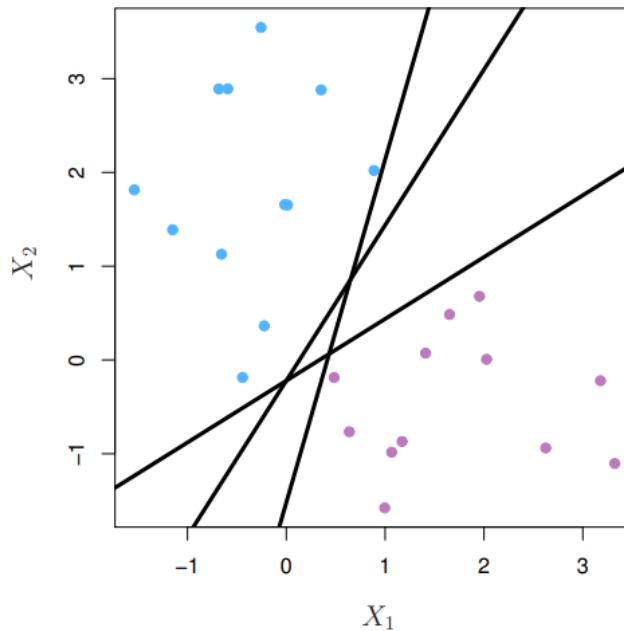
$$\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip} < 0 \text{ if } y_i = -1$$

- Entonces, nuestro hiperplano debe cumplir con:

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip}) > 0$$

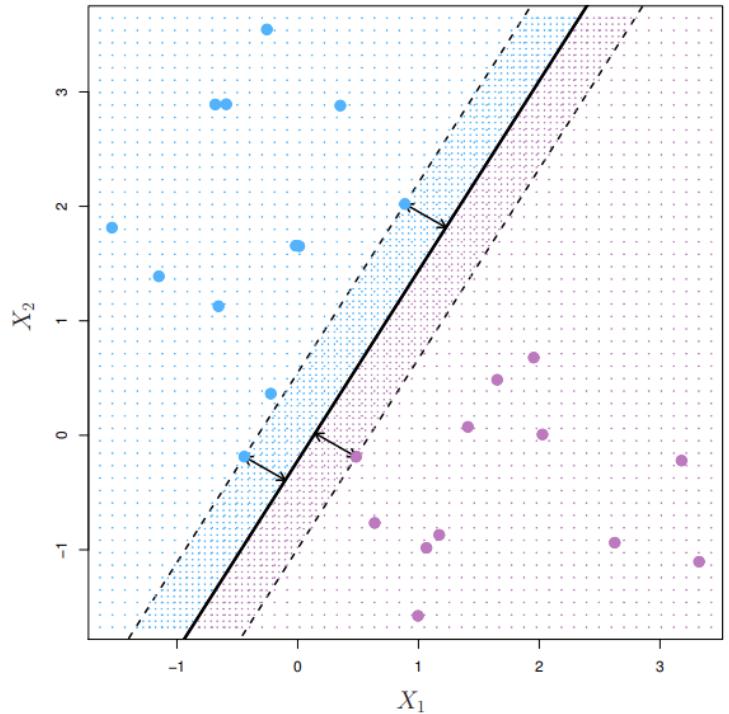

¿Cómo encontramos el mejor hiperplano?

- Si sólo buscamos la condición anterior, tenemos muchos hiperplanos que la pueden cumplir.
- ¿Cómo encontramos el mejor?
- ¿Qué sería el mejor separador?



Clasificador de máximo margen

- Si buscamos maximizar la distancia entre el hiperplano y los puntos de muestra, encontramos el “mejor” hiperplano.
- Los puntos más cercanos al hiperplano definen su dirección: son los vectores de soporte.





Clasificador de máximo margen

- Para encontrar el hiperplano, debemos resolver un problema de optimización:

$$\underset{\beta_0, \beta_1, \dots, \beta_p, M}{\text{maximize}} \quad M$$

subject to $\sum_{j=1}^p \beta_j^2 = 1,$

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip}) \geq M \quad \forall i = 1, \dots, n.$$


En Python

- La librería `sklearn` posee ya implementado una SVM lineal.
- La función permite usar regularización.

`sklearn.svm.LinearSVC`

```
class sklearn.svm.LinearSVC(penalty='l2', loss='squared_hinge', *, dual='warn', tol=0.0001, C=1.0, multi_class='ovr',  
    fit_intercept=True, intercept_scaling=1, class_weight=None, verbose=0, random_state=None, max_iter=1000)
```

[source]

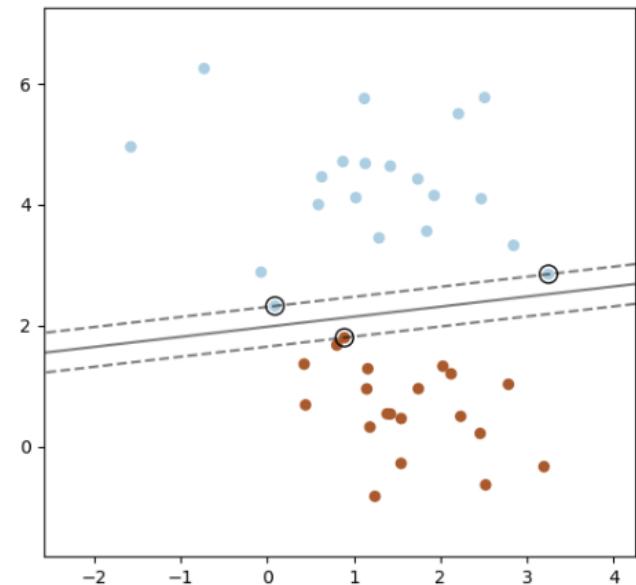
Linear Support Vector Classification.

Similar to SVC with parameter `kernel='linear'`, but implemented in terms of liblinear rather than libsvm, so it has more flexibility in the choice of penalties and loss functions and should scale better to large numbers of samples.

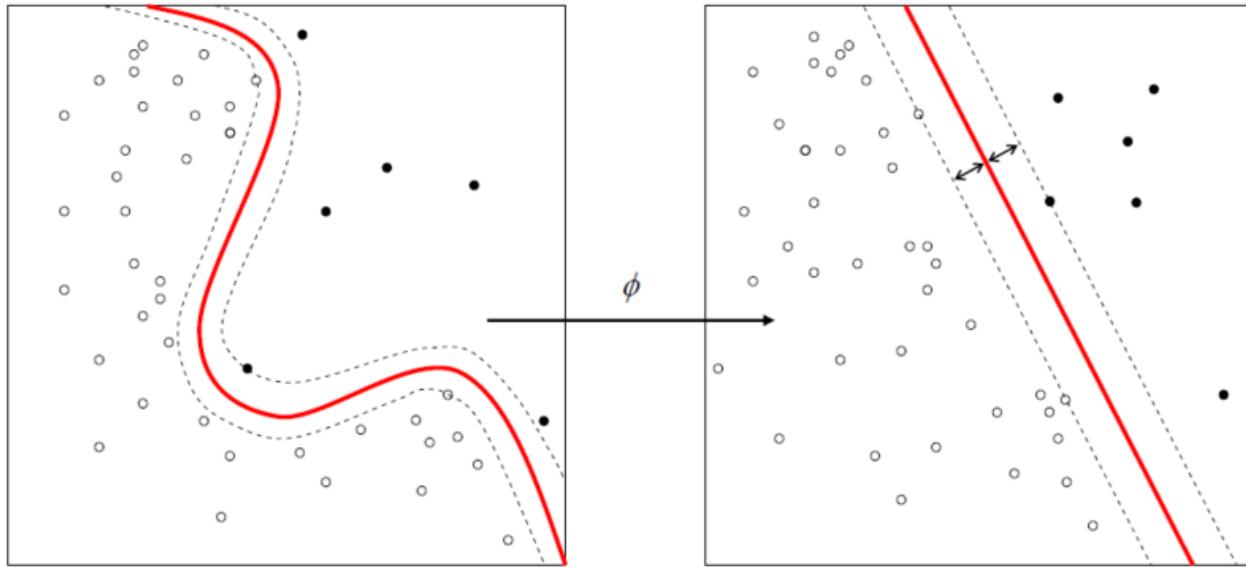
The main differences between `LinearSVC` and `svc` lie in the loss function used by default, and in the handling of intercept regularization between those two implementations.

This class supports both dense and sparse input and the multiclass support is handled according to a one-vs-the-rest scheme.

Read more in the [User Guide](#).



¿Qué pasa si no lo podemos separar?





Clasificador de Soporte Vectorial

- También denominado “Soft Margin Classifier” permite que algunas muestras no estén bien clasificadas.

$$\underset{\beta_0, \beta_1, \dots, \beta_p, \epsilon_1, \dots, \epsilon_n, M}{\text{maximize}} \quad M$$

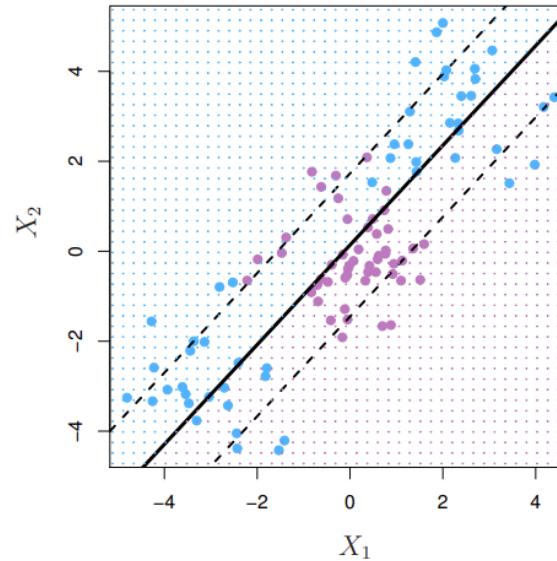
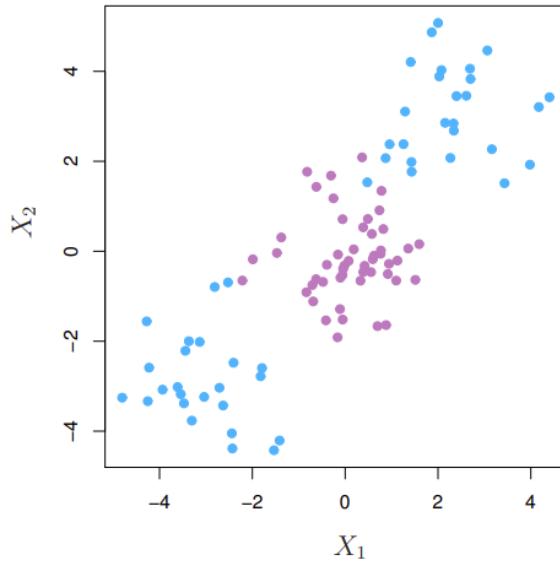
$$\text{subject to} \quad \sum_{j=1}^p \beta_j^2 = 1,$$

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \dots + \beta_p x_{ip}) \geq M(1 - \epsilon_i),$$

$$\epsilon_i \geq 0, \quad \sum_{i=1}^n \epsilon_i \leq C,$$


Separación no lineal

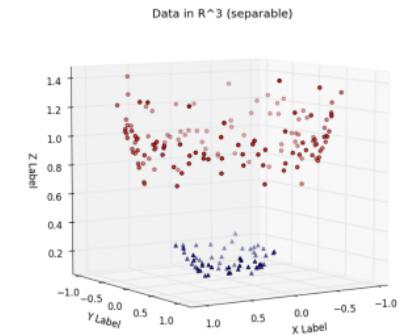
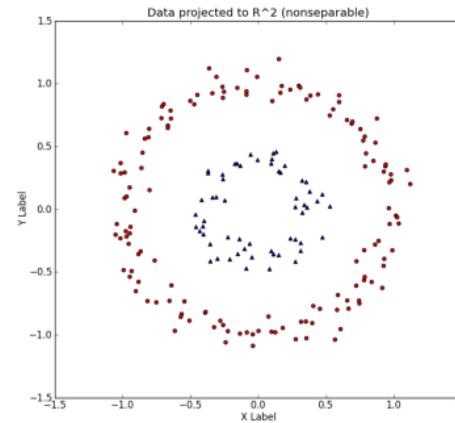
- Pero hay instancias en las cuales incluso el margen flexible no sirve.



Extendiendo el espacio

- La opción en estos casos es usar una frontera no-lineal.
- Esto es equivalente a extender nuestro espacio en dimensiones adicionales.

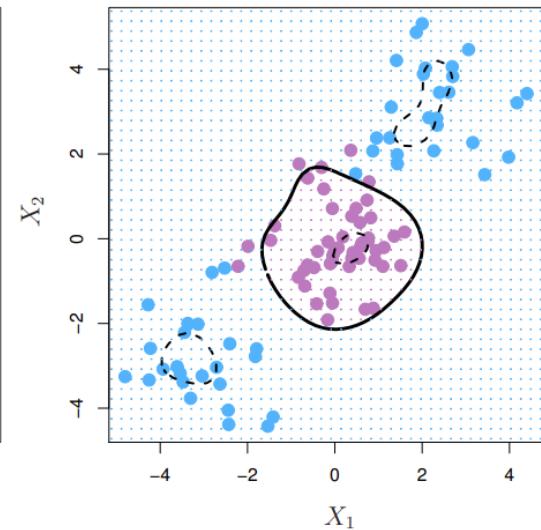
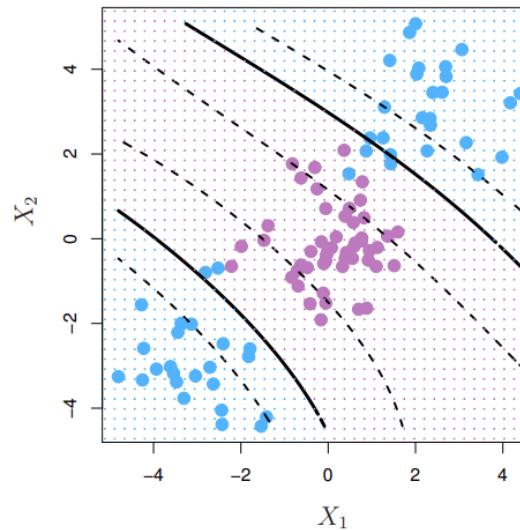
$$y_i \left(\beta_0 + \sum_{j=1}^p \beta_{j1} x_{ij} + \sum_{j=1}^p \beta_{j2} x_{ij}^2 \right) \geq M(1 - \epsilon_i)$$



Máquinas de Sporte Vectorial

- La extensión con múltiples familias de fronteras no lineales se llama SVM.
- La frontera queda definida por un “kernel” que aumenta la dimensión de nuestras muestras.

$$K(x_i, x_{i'}) = \left(1 + \sum_{j=1}^p x_{ij}x_{i'j}\right)^d.$$



En Python

- La librería `sklearn` también tiene la opción de usar otros kernels.

`sklearn.svm.SVC`

```
class sklearn.svm.SVC(*, C=1.0, kernel='rbf', degree=3, gamma='scale', coef0=0.0, shrinking=True, probability=False, tol=0.001, cache_size=200, class_weight=None, verbose=False, max_iter=-1, decision_function_shape='ovr', break_ties=False, random_state=None)
```

[source]

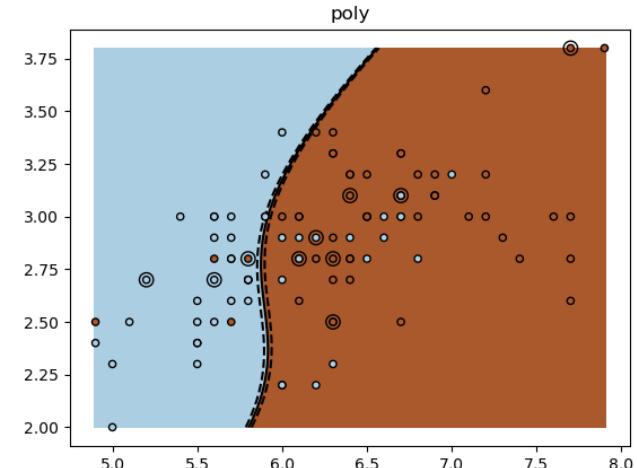
C-Support Vector Classification.

The implementation is based on libsvm. The fit time scales at least quadratically with the number of samples and may be impractical beyond tens of thousands of samples. For large datasets consider using `LinearSVC` or `SGDClassifier` instead, possibly after a `Nystroem` transformer or other `Kernel Approximation`.

The multiclass support is handled according to a one-vs-one scheme.

For details on the precise mathematical formulation of the provided kernel functions and how `gamma`, `coef0` and `degree` affect each other, see the corresponding section in the narrative documentation: [Kernel functions](#).

Read more in the [User Guide](#).



03

Machine Learning

Aprendizaje no supervisado - clustering





Aprendizaje no supervisado

Algoritmos de aprendizaje de máquina donde **no existe etiqueta conocida**, sino que se pide al algoritmo extraer conocimiento a partir de los datos de entrada.

En general, hay **dos grandes clases** de algoritmos de aprendizaje no supervisado:

1. **Clustering**: algoritmos para particionar la data en grupos disímiles, pero con ítems similares.
2. **Transformaciones**; algoritmos que crean una nueva representación de los datos que puede ser más fácil de entender para humano o máquinas en comparación a la representación original.

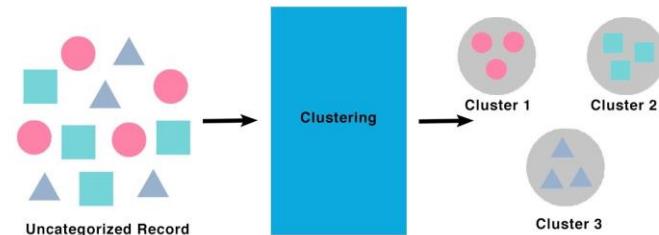
- Reducción de dimensionalidad.
- Extracción de tópicos

Desafío: evaluar si el algoritmo aprendió lo que se buscaba. Muchas veces se requiere inspeccionar o evaluar manualmente los resultados.

- Aprendizaje no supervisado se utiliza muchas veces en una etapa de exploración, o como parte del preprocesamiento para algoritmos supervisados.
- 

Clustering

- Algoritmos para partitionar la data en grupos disímiles, pero con ítems similares.
- Cada observación del dataset es asignada a sólo una categoría según el valor de las variables consideradas para la clasificación.
- Los **clusters** se definen de manera que cada uno agrupe observaciones similares entre sí, pero difieran entre grupos.
- El análisis se simplifica enfocándose sólo en los perfiles de cada cluster, que permiten interpretar la estructura de data multi-variable.

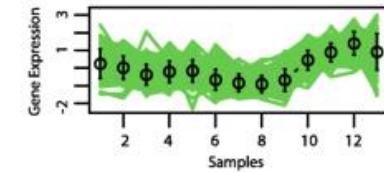
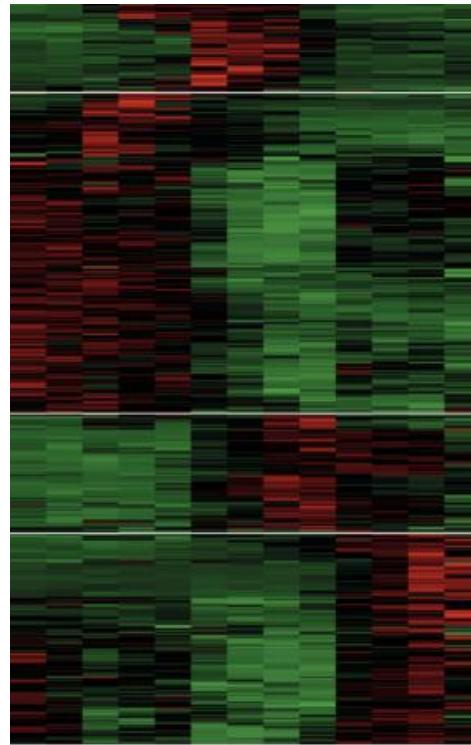


Aplicaciones: patrones genéticos

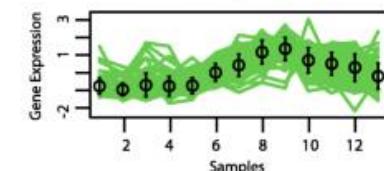
A multi-objective gene clustering algorithm guided by apriori biological knowledge with intensification and diversification strategies

Jorge Parraga-Alava, Marcio Dorn & Mario Inostroza-Ponta 

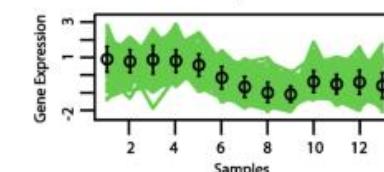
BioData Mining 11, Article number: 16 (2018) | [Cite this article](#)



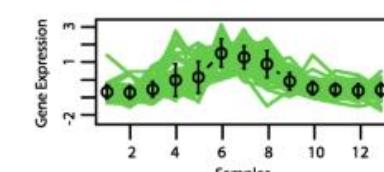
Cluster 1



Cluster 2



Cluster 3



Cluster 4

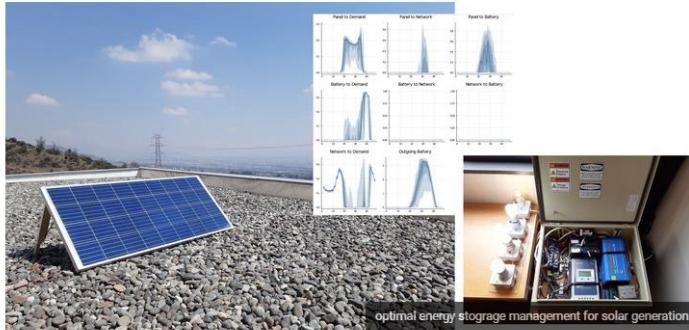
Aplicaciones: generación de escenarios

GEMS: variable-Generation Energy Management with Storage

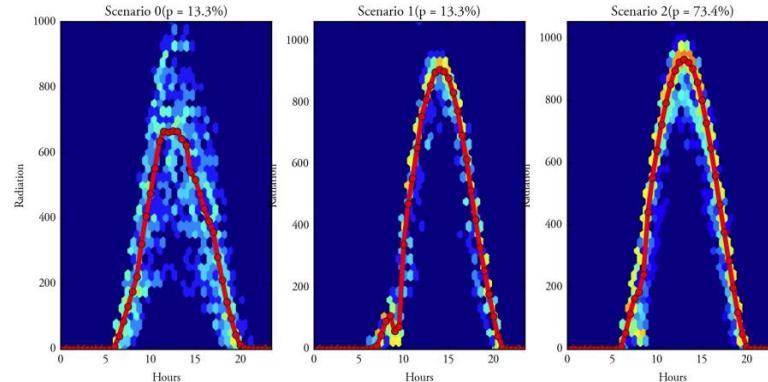
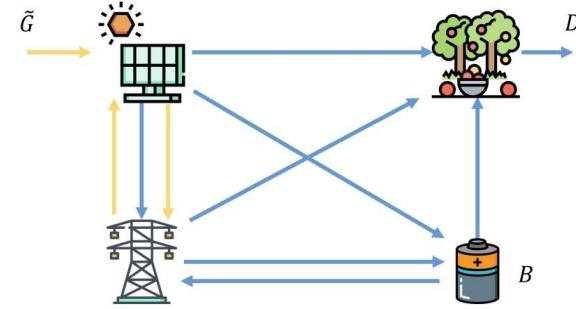
Rodrigo A. Carrasco, Tito Homem De Mello, Francisco Jara, Jocelyn Olivari, Gonzalo Ruz, Carlos Silva,

Benjamín Bastidas, José L. Ortiz, Anthony Cho, Helena Garcia, César Cerdá

May 23, 2020



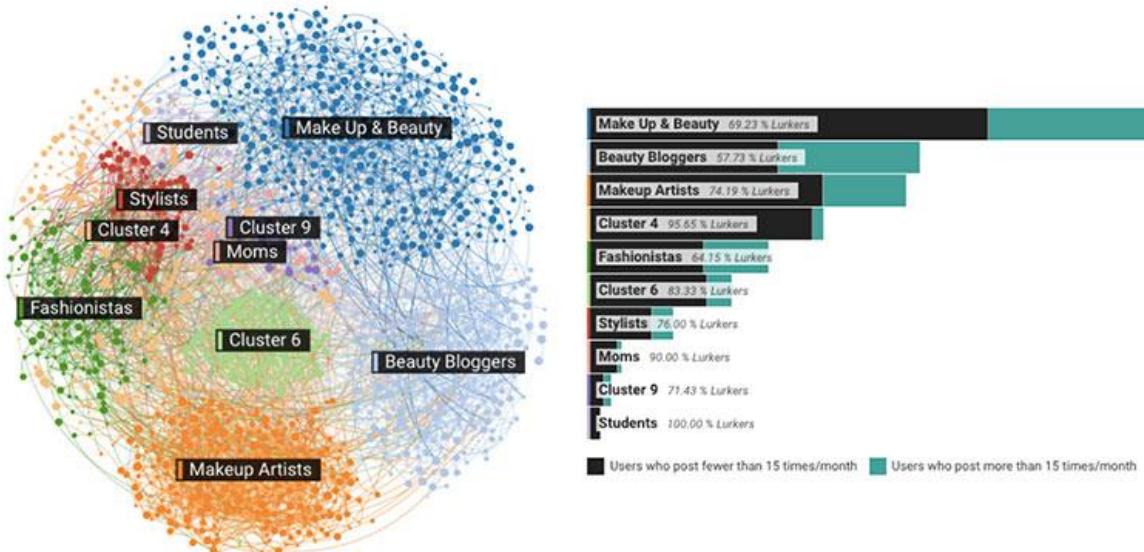
One of the exciting topics we've been working on is algorithms and models related to energy management, where uncertainty is a crucial element in the process. In particular,



Aplicaciones: clasificación de usuarios

Affinio Deep Learning Social Network Analysis Clustering

by Mariya Yao | Mar 5, 2018



Aplicaciones: identificación puntos de conexión

