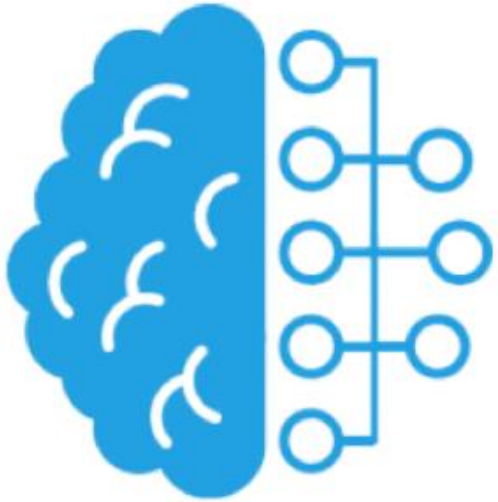
Aprendizaje de Máquina

Clase 3

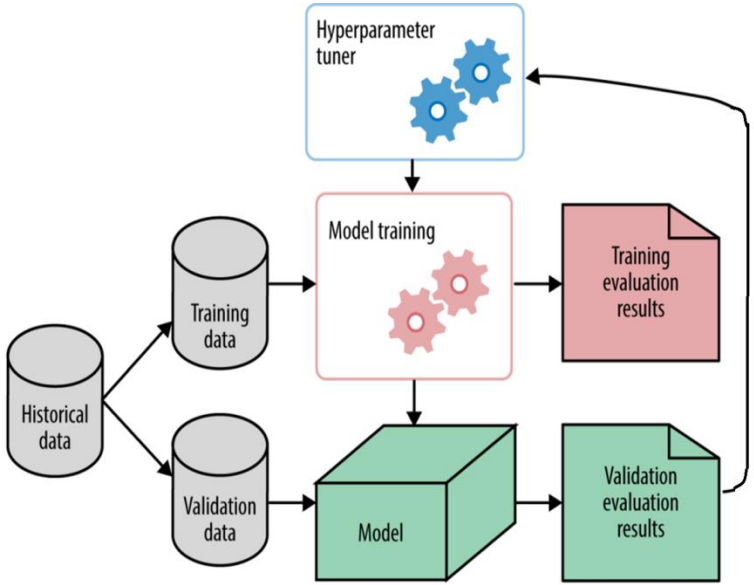
Claudio Delrieux – DIEC - UNS
cad@uns.edu.ar



Clasificadores no paramétricos

Si bien vamos a regresar a la clasificación basada estadística paramétrica, primero vamos a plantear un modelo más general, en el cual no se hace ninguna suposición respecto de los datos.

Tenemos tres tiempos, **entrenamiento** y **validación** (con datos conocidos y en el contexto de aprendizaje) y **testeo** (con datos desconocidos, y en el contexto de aplicación).



No free lunch

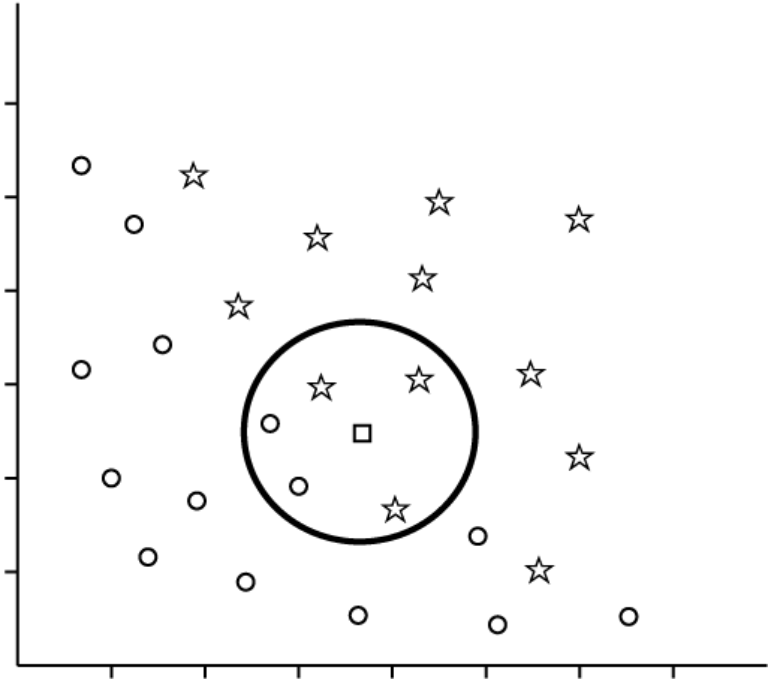
El teorema *no free lunch* es un resultado matemático que establece que para ciertas clases de problemas, el costo computacional de encontrar una solución general “promediada” sobre todos los problemas de la clase, es el mismo para cada modelo o método. Determinar entonces un “atajo” es tan costoso como computar todas las soluciones requeridas.

Este teorema aplica si el espacio de búsqueda es una función de densidad probabilística. Si este espacio tiene estructura subyacente entonces ésta puede ser aprovechada y obtener modelos más eficientes.

K vecinos más cercanos

Se conoce también como “aprendizaje perezoso” porque no hay un proceso de entrenamiento (aunque sí hay N datos del dataset original que se usan para entrenamiento).

Dado un dato de entrada, se buscan los k que estén a menor distancia en el espacio de atributos, y se clasifica por mayoría.



K vecinos más cercanos

En https://github.com/manlio99/Materia-de-aprendizaje/blob/master/5_DataMining/2_classification.ipynb colocamos inicialmente un ejemplo de dataset, queremos organizar las diferentes flores por forma y tamaño de sus pétalos:

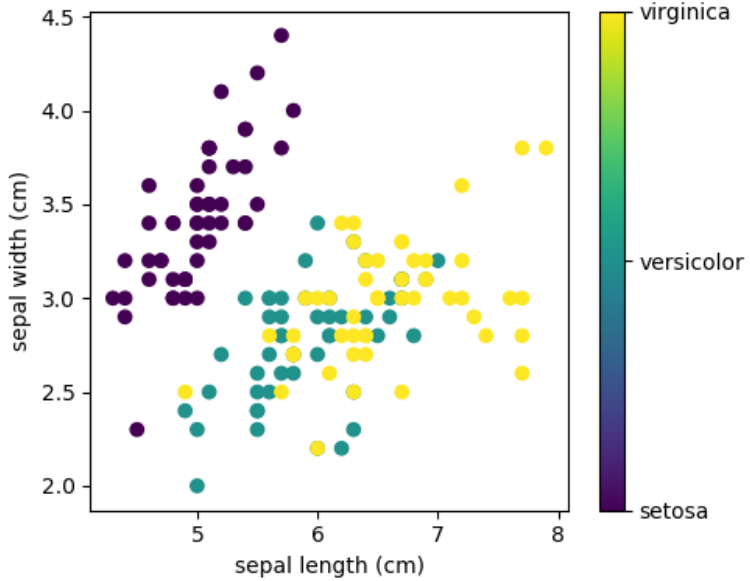
Setosa Iris

Versicolor Iris

Virginica Iris

K vecinos más cercanos

```
from sklearn.datasets import load_iris
iris = load_iris()
>>> print(iris.data.shape)
(150,4)
>>> print(iris.data[0])
[5.1  3.5  1.4  0.2]
plt.figure(figsize=(5, 4))
plt.scatter(iris.data[:,x_index],
            iris.data[:, y_index],
            c=iris.target)
```

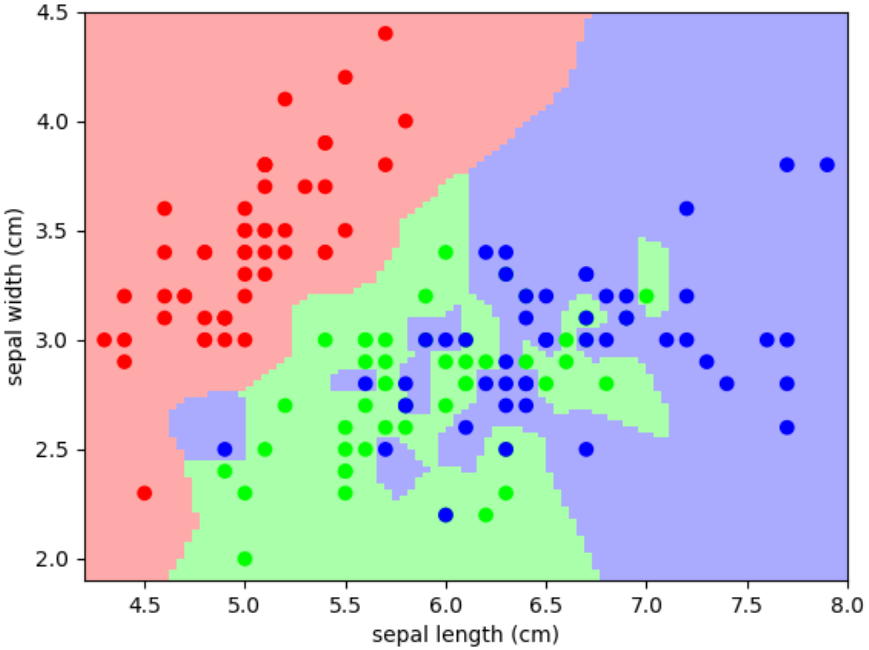


K vecinos más cercanos

```
from sklearn import neighbors
X, y = iris.data, iris.target
knn = neighbors.
    KNeighborsClassifier
    (n_neighbors=1)
knn.fit(X, y)
```

Probarlo con n_neighbors=2, 3…

`knn.fit()`,`knn.predict()` y `knn.score()` son algunos de los métodos que se pueden aplicar.



K vecinos más cercanos

Si bien el modelo es muy sencillo, hay varios aspectos a considerar y que nos permiten anticipar y entender mejor lo que ocurrirá con modelos más complejos:

- El valor de k es el único parámetro del modelo. K no determina la complejidad del cómputo de cada nuevo caso, dado que hay que computar las N distancias.
- Valores de k muy grandes (N/2 o mayor) determinan modelos con alto **sesgo** (tendencia del modelo a elegir la clase mayoritaria). A este fenómeno lo llamamos también subajuste (underfitting): el modelo generalizó por demás.

K vecinos más cercanos

- A la inversa, valores de k muy pequeños (3 o menor) determinan modelos con alta **variancia** (tendencia del modelo a elegir en forma muy inestable). En este caso hablamos de sobreajuste (overfitting): el modelo fue incapaz de generalizar.
- El sesgo del modelo se evalúa con la proporción entre casos positivo y negativo, respecto de la proporción entre predicciones positiva y negativa: (TP+FN)/(TN+FP) : (TP+FP)/(TN+FN).
- La variancia del modelo se puede evaluar con la FDR (false discovery rate) FP/(TP+FP) o con la media armónica de FDR y FOR.

K vecinos más cercanos

- El equilibrio entre sesgo y variancia depende del dataset. Si los datos de cada categoría están muy bien agrupados un k bajo tendrá buen comportamiento. Una regla heurística es usar raíz de N.
- La elección de los N datos de entrenamiento también tienen efecto en el comportamiento. N más grande genera un modelo más robusto pero más costoso de computar, y a la inversa.
- Una selección aleatoria insesgada puede garantizar representatividad, pero puede no ser la elección más eficiente.

K vecinos más cercanos

Ejercicio 3.1: (Recordar el ej. 2.3). Supongamos que tenemos un dataset mezcla de dos grupos aleatorios bidimensionales (cada uno con 100 datos, su centroide y covariancia).
Tip: reutilizar el código de la NB que vimos para outliers.

Encontrar un clasificador por K-NN y evaluar exactitud, precisión, f-measure. Tomar inicialmente un dataset de entrenamiento de N=100 (50+50), k=10, y validar con el resto del dataset. Hacer algunos experimentos con otros N y k, y también cambiando centroide y covariancia de las clases.

Optativo: Cambia mucho si el dataset tiene más dimensiones?

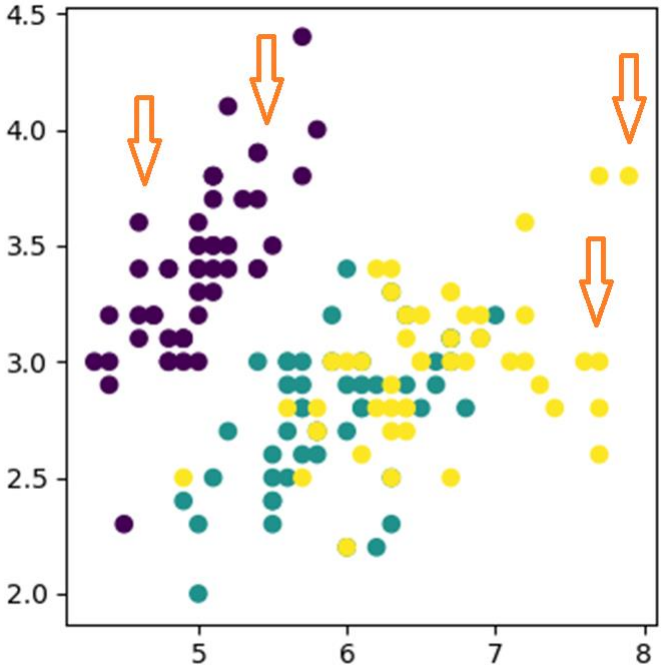
Etiq.	X	Y
A	12	1
A	13	1
B	17	1
B	22	2
A	24	2
A	29	2
B	30	2
B	31	3
A	33	3
B	35	3
A	36	3
A	36	4
B	36	4
B	37	4
A	39	4
...	...	...

K vecinos más cercanos

La elección del dataset de entrenamiento puede tener un efecto dramático en el modelo.

Aquí marcamos algunas regiones del dataset original que claramente no aportan a la calidad del método. Cómo elegir los más adecuados?

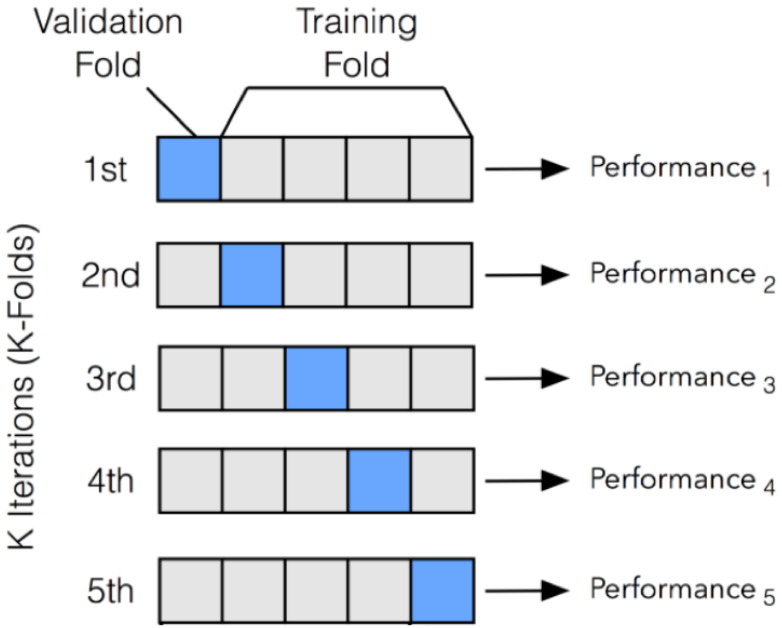
- Cross-fold validation.
- Topología (ver las homologías).



Validación cruzada

La validación cruzada es un método estadístico utilizado para validar modelos, garantizar que el muestreo no tiene influencia, detectar overfitting, etc. https://en.wikipedia.org/wiki/Cross-validation_(statistics).

Particionamos al dataset en K grupos (no confundir con el k de K-NN) y realizamos permutaciones de entrenamiento y validación.



Validación cruzada

Existen varias técnicas de separación (split): estratificado, por grupos, por mezcla, etc., para realizarlo en diversos contextos (datasets sesgados, por ejemplo). Están implementados en los métodos de *model selection* https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.KFold.html

Para un modelo tan sencillo como K-NN tenemos que probar con diferentes N, diferentes k, elegir los atributos, hacer validación cruzada, etc. (esos son los “hiperparámetros” del modelo).

Grid search

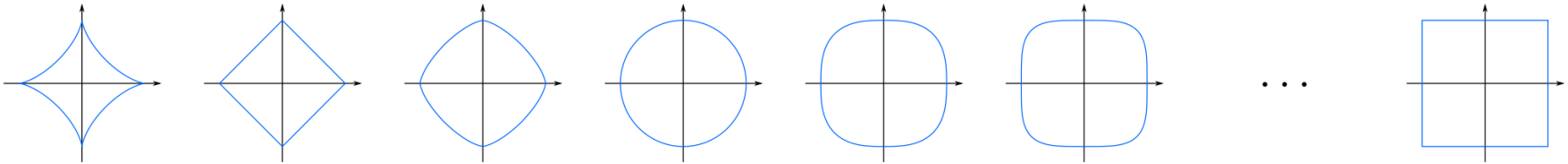
Afortunadamente, *model selection* incluye también métodos para explorar sistemáticamente el espacio de hiperparámetros y recolectar los resultados de cada combinación. https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html

Ejercicio 3.2 (optativo): Hacer alguna experiencia con el dataset anterior utilizando validáción cruzada y (o) grid search.

Métricas en el espacio de atributos

Volviendo a nuestro modelo, existen aún más elementos a tener en cuenta. Estamos utilizando distancia Euclidea por defecto (no tenemos parámetros estadísticos: no hay Mahalanobis). Es el caso particular de la distancia D de Minkowski para p=2:

$$D(X,Y) = \left(\sum_{i=1}^{n} |x_i - y_i|^p \right)^{\frac{1}{p}}$$



$p = 2^{-0.5}$ $= 0.707$ | $p = 2^{0}$ $= 1$ | $p = 2^{0.5}$ $= 1.414$ | $p = 2^{1}$ $= 2$ | $p = 2^{1.5}$ $= 2.828$ | $p = 2^{2}$ $= 4$ | $\ldots$ | $p = 2^{\infty}$ $= \infty$

Métricas en el espacio de atributos

El valor absoluto de todas las variables tiene la misma importancia. Eso hace que el modelo pueda ser sensible a la escala arbitraria de los datos. Por eso tenemos que conocer cómo están distribuidos los atributos individuales (por columna). Algunas técnicas son:

- Normalizar los valores (representar de 0 a 1, cuando no siguen ninguna distribución reconocible).
- Estandarizar (llevar a valor-z).
- Ponderar cada variable de acuerdo a alguna importancia conocida.
- Cuantificar los valores nominales/ordinales.

Selección de atributos

Un capítulo aparte es la selección de variables o atributos. No todos los atributos de un dataset pueden ser informativos para el modelo, incluso algunos pueden ser perjudiciales. Además, datasets muy “anchos” generan modelos muy costosos.

La selección de atributos se puede hacer por varios caminos:

- Agregar incrementalmente
- Quitar selectivamente
- Utilizar “informatividad”
- Reducción de dimensionalidad (p.ej, componentes principales)

Más aspectos de la curación del dataset

En K-NN y otros métodos basados en similitud, también hay que tener en cuenta otros factores:

- Filtrado de datos (registros erróneos), dataset ancho vs. alto.
- Agregación de datos.
- “Binning” (recipientización).
- Imputaciones.
- Etc. etc.

K vecinos más cercanos

Ejercicio 3.3: En el notebook https://github.com/manlio99/Materia-de-aprendizaje/blob/master/3_MidtermProjects/musica.ipynb hay un dataset con 2000+ canciones de Spotify de una usuaria, donde algunas fueron marcadas como gustadas y otras no. Cada canción tiene a su vez 16 atributos (nombre, artista, duración, bailable, etc.). Desarrollar un clasificador que prediga si una canción dada va a ser gustada o no. Aclarar y justificar los pasos, analizar y explicar los resultados.