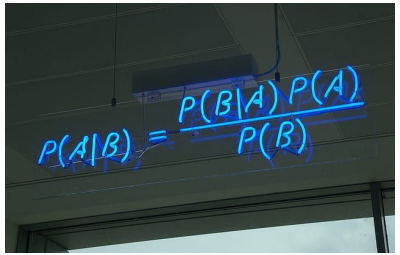
DEPARTAMENTO DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA



Aprendizaje Automático

Clase 4:

Clasificación
Modelos Discriminativos y Generativos

El clasificador óptimo de Bayes

Al construir un clasificador, en general estamos interesados en disminuir el error en "la realidad". Es decir, minimizar

$$Err_{true}(h) = \mathbb{E}_x[error(h(x), y)]$$

En clasificación el error puede definirse como `h(x) != y` (se lo llama "riesgo"). Se puede demostrar que el clasificador que minimiza este error es:

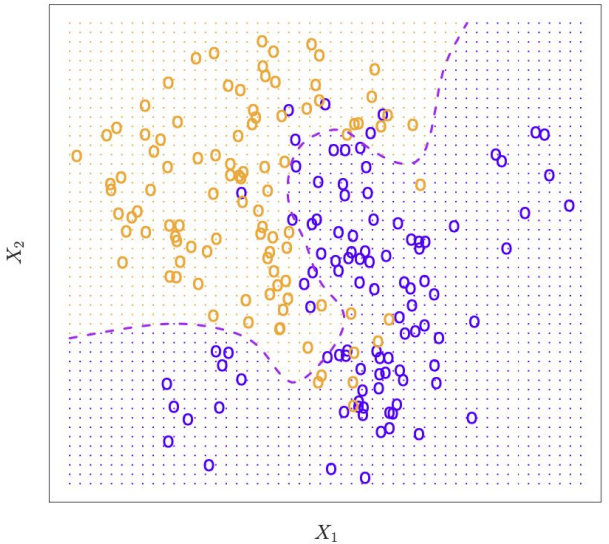
El **clasificador óptimo de Bayes,** un modelo probabilístico que devuelve la predicción más probable para un nuevo ejemplo, dado los valores de sus atributos. Es decir,

$$Pred(x^{(i)}) = \underset{c \in Clases}{\arg\max}\, P(Y = c | X = x^{(i)})$$

En teoría, siempre nos gustaría predecir respuestas cualitativas utilizando el clasificador de Bayes. Pero **para datos reales, no conocemos la distribución condicional de Y dada X**, por lo que calcular el clasificador de Bayes es **imposible**.

Por lo tanto, el clasificador de Bayes sirve como un estándar inalcanzable. En casos específicos (datos simulados), permite comparar diferentes métodos.

Fronteras de decisión según un clasificador de tipo "Bayes Optimal Classifier" para datos simulados .



(**notar** que la frontera hubiera sido la misma sin importar que muestra se visualice aquí)

Modelos Discriminativos

Enfoques (parte 1)

Muchos enfoques intentan **estimar** esta probabilidad a partir de un conjunto de **datos de entrenamiento**. Es decir, modelar (aprender):

$$P(Y = c|X = x^{(i)}) \quad \forall c \in Clases$$

Luego clasificar una observación dada en la clase con la probabilidad estimada más alta:

$$Pred(x^{(i)}) = \arg\max_{c \in Clases} \hat{P}(Y{=}c|X{=}x^{(i)})$$

Estos se conocen como modelos **DISCRIMINATIVOS**

(algunos) ejemplos de modelos discriminativos:

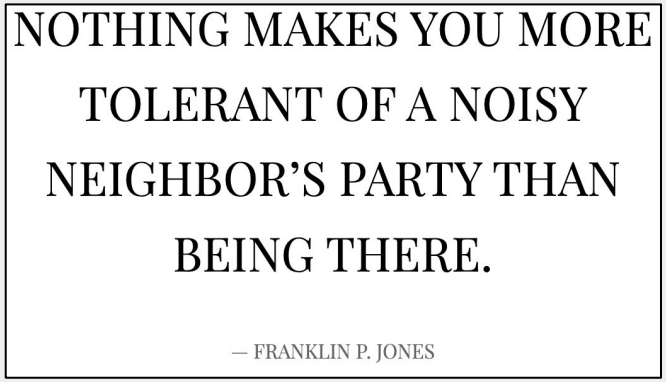
- **Árboles de decisión** (ya lo vimos)
- **K-vecinos más cercanos** (hoy)
- **Support Vector Machines (SVM)** (hoy)
- **Regresión logística** (prox)
- **Random Forest (y otros ensambles)** (prox)
- **Maximum-entropy Markov models**
- **Conditional random fields**
- **Redes neuronales (no todas)** (prox)

(más adelante en la clase veremos que no es el único enfoque).

modelos discriminativos

K Vecinos Más Cercanos

KNN



$$Pred(x^{(i)}) = \arg\max_{c \in Clases} \hat{P}(Y=c|X=x^{(i)})$$

K vecinos más cercanos (KNN)

Método discriminativo

Estima la probabilidad condicional para la clase $\mathbf{c}$ dado $\mathbf{x}$ como: La fracción de puntos en $\mathbf{N_{D,k}(x)}$.
$\mathbf{N_{D,k}(x)}$ = los $\boldsymbol{k}$ vecinos más cercanos a $\mathbf{x}$ en el conjunto $\mathbf{D}$, cuyas etiquetas sean iguales a $\mathbf{c}$.

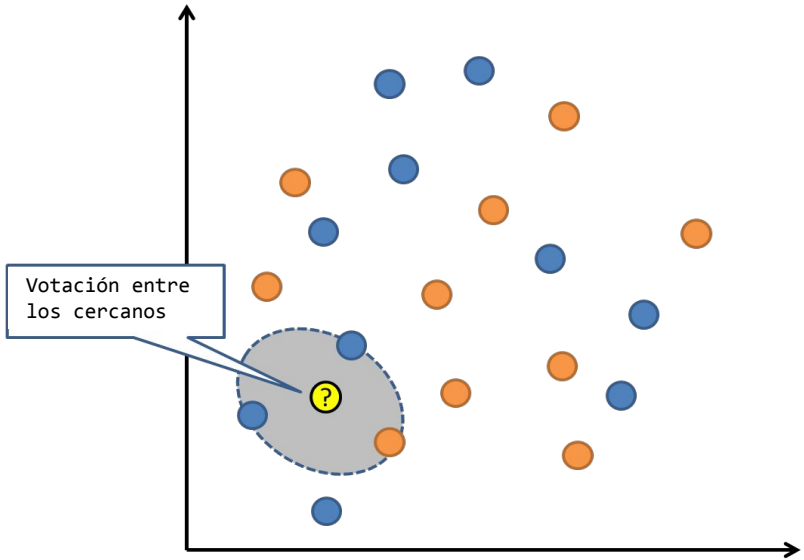
$$Pred(x^{(i)}) = \arg\max_{c \in Clases} P(Y = c|X = x^{(i)})$$

$$\approx \arg\max_{c \in Clases} \sum_{i \in \mathcal{N}_{D,k}(x)} I(y^{(i)} = c)$$

+ sesgo inductivo
+ datos

$$Pred_{proba}(x^{(i)}, c) = P(Y = c|X = x^{(i)})$$

$$\approx \frac{1}{k} \sum_{i \in \mathcal{N}_{D,k}(x)} I(y^{(i)} = c)$$



Algunas versiones ponderan la votación según la distancia

$$Pred(x^{(i)}) = \arg\max_{c \in Clases} \hat{P}(Y=c|X=x^{(i)})$$

K vecinos más cercanos (KNN)

Método discriminativo

¿Cómo se implementa? Algoritmo de asignación a la instancia $x^{(i)}$:

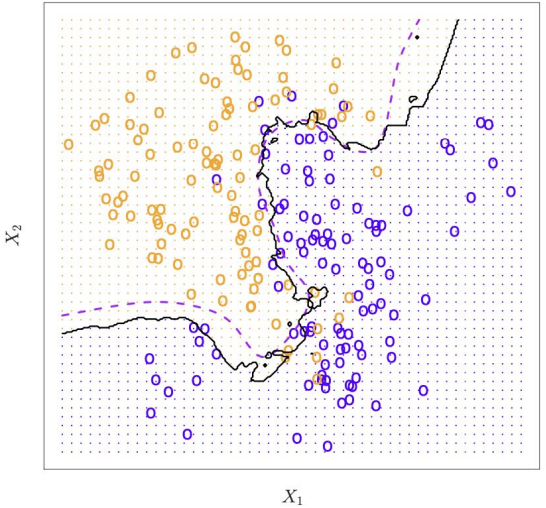
1. Computar la distancia $D(\mathbf{x}^{(i)}, \mathbf{x}^{(j)})$ para todo punto de entrenamiento $\mathbf{x}^{(j)}$.
2. Seleccionar las k instancias más cercanas y sus etiquetas.
3. Devolver la etiqueta más frecuente (o la proporción si es proba).

Pregunta (clásico de entrevistas).
Quiero predecir si una casa tendrá techo verde o no.
¿Qué pasa si los atributos son los siguientes y aplico KNN?

$\mathbf{X_1}$: distancia al obelisco *(medido en metros)*,
$\mathbf{X_2}$: cantidad de personas que viven en la cuadra *(de 1 a 500)*.
$\mathbf{X_3}$: ¿tiene jardín?: *si, no (1 o 0)*.

¿Qué distancia uso? ¿Euclidiana? ¿Hago algo antes? ¿Por qué no pasaba en árboles?

Fronteras de decisión según un clasificador de tipo **KNN** con **K=10.**
(¿cómo dibujarían esto?)
Línea punteada: **Bayes Optimal Classifier**



¿Cómo cambian las fronteras de decisión si cambio el K?

$$Pred(x^{(i)}) = \arg\max_{c \in Clases} \hat{P}(Y=c|X=x^{(i)})$$

K vecinos más cercanos (KNN)

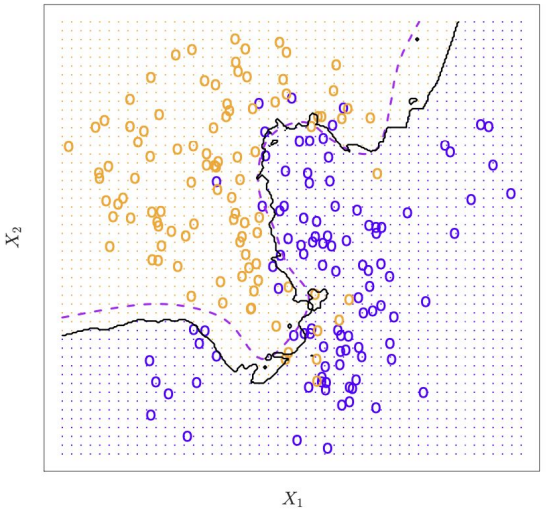
Método discriminativo

¿Cómo se implementa? Algoritmo de asignación a la instancia $x^{(i)}$:

1. Computar la distancia $D(\mathbf{x}^{(i)}, \mathbf{x}^{(j)})$ para todo punto de entrenamiento $\mathbf{x}^{(j)}$.
2. Seleccionar las k instancias más cercanas y sus etiquetas.
3. Devolver la etiqueta más frecuente (o la proporción si es proba).

- ¿Cuánto ocupa en memoria una vez entrenado el modelo ?
- ¿Modelo?
- ¿Cuál es el algoritmo de entrenamiento?
- ¿Entrenamiento?

Fronteras de decisión según un clasificador de tipo **KNN** con **K=10.**
(¿cómo dibujarían esto?)
Línea punteada: **Bayes Optimal Classifier**



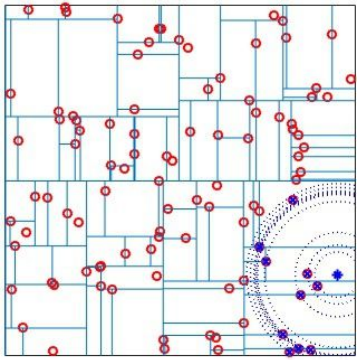
$$Pred(x^{(i)}) = \arg\max_{c \in Clases} \hat{P}(Y=c|X=x^{(i)})$$

K vecinos más cercanos (KNN)

Método discriminativo

¿Cómo se implementa? Algoritmo de asignación a la instancia $x^{(i)}$:

1. Computar la distancia $D(\mathbf{x}^{(i)}, \mathbf{x}^{(j)})$ para todo punto de entrenamiento $\mathbf{x}^{(j)}$.
2. Seleccionar las k instancias más cercanas y sus etiquetas.
3. Devolver la etiqueta más frecuente (o la proporción si es proba).



Fuente: https://www.vlfeat.org/overview/kdtree.html

Costo computacional: $O(n*d + k*n)$ + $O(n)$ memoria
(hay otras opciones dependiendo de decisiones de implementación).
Ver https://stats.stackexchange.com/questions/219655/k-nn-computational-complexity

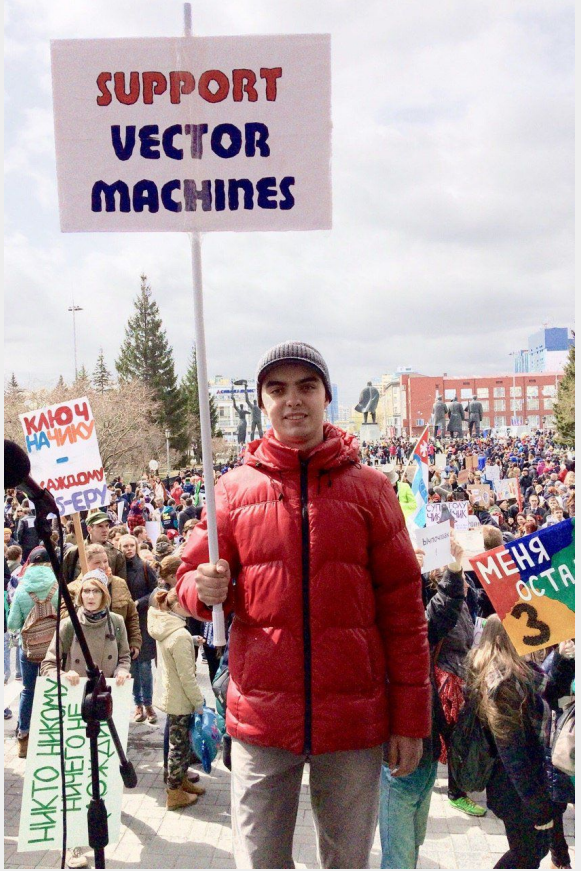
¿Se puede mejorar el costo $O(|D|)$ por query?
Ver estructuras de datos eficientes: "BallTree" y "KDTree".

modelos discriminativos

Support Vector Machines

SVM

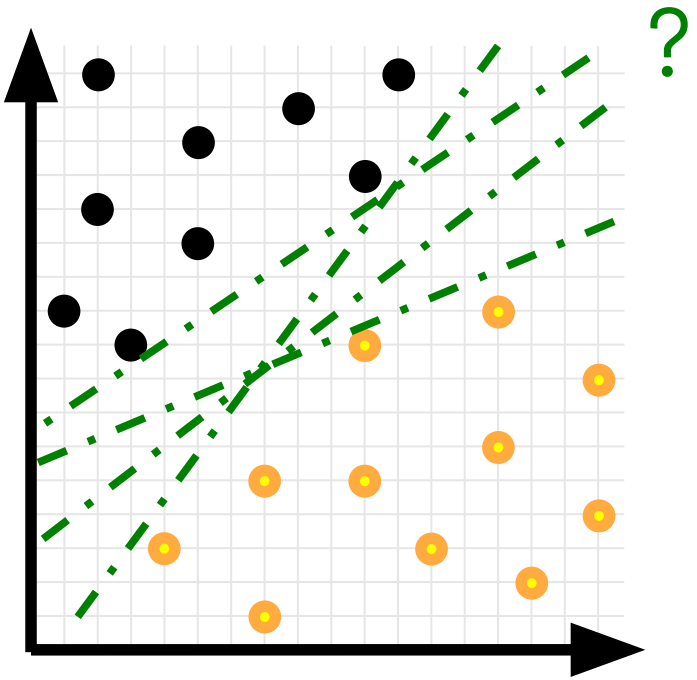
Recomendado [StatQuest - SVM (youtube)]



$$Pred(x^{(i)}) = \arg\max_{c \in Clases} \hat{P}(Y=c|X=x^{(i)})$$

Support Vector Machine (SVM)

Método discriminativo



Idea: buscar una recta que separe las clases lo mejor posible, sin tener que modelar la distribución de los datos de cada clase.

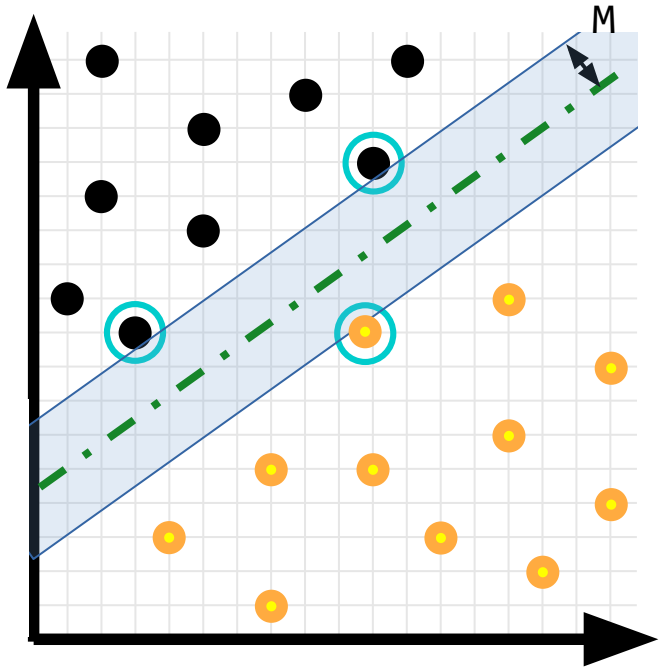
¿Cómo podemos elegir esa recta?
(supongamos por un rato que existe dicha recta)

Buscaremos el **hiperplano de margen máximo** (también conocido como hiperplano de separación óptimo), que es el hiperplano de separación que está más alejado de las observaciones de entrenamiento.

$$Pred(x^{(i)}) = \arg\max_{c \in Clases} \hat{P}(Y=c|X=x^{(i)})$$

Support Vector Machine (SVM)

Método discriminativo



Hiperplano de margen máximo (recta verde en este caso). El hiperplano de dimensión **(d-1)** tal que la distancia a las instancias más cercanas es máxima y todas las instancias quedan del lado correcto del plano. Recordar que las instancias viven en el espacio de atributos de dim **d.**

Margen M:
Distancia de las instancias más cercanas a la recta (hiperplano) de decisión.

"Support Vectors" (vectores de soporte, en este caso con cículos celestes)**:** Instancias más cercanas al hiperplano de decisión

El algoritmo buscará maximizar M.

- Problema de optimización, de la pinta "Minimizar tal expresión con tal y tal restricción".
- Existe una solución eficiente (mundo programación cuadrática). No la veremos.

$$Pred(x^{(i)}) = \arg\max_{c \in Clases} \hat{P}(Y=c|X=x^{(i)})$$

Support Vector Machine (SVM)

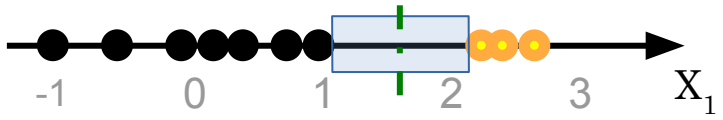
Método discriminativo

Notación
(instancia / atributos)

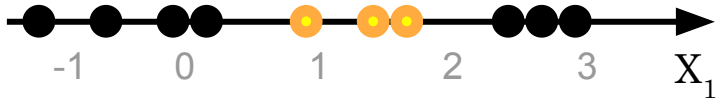
$x^{(i)} = [x^{(i)}_1, x^{(i)}_2, \ldots, x^{(i)}_p]$

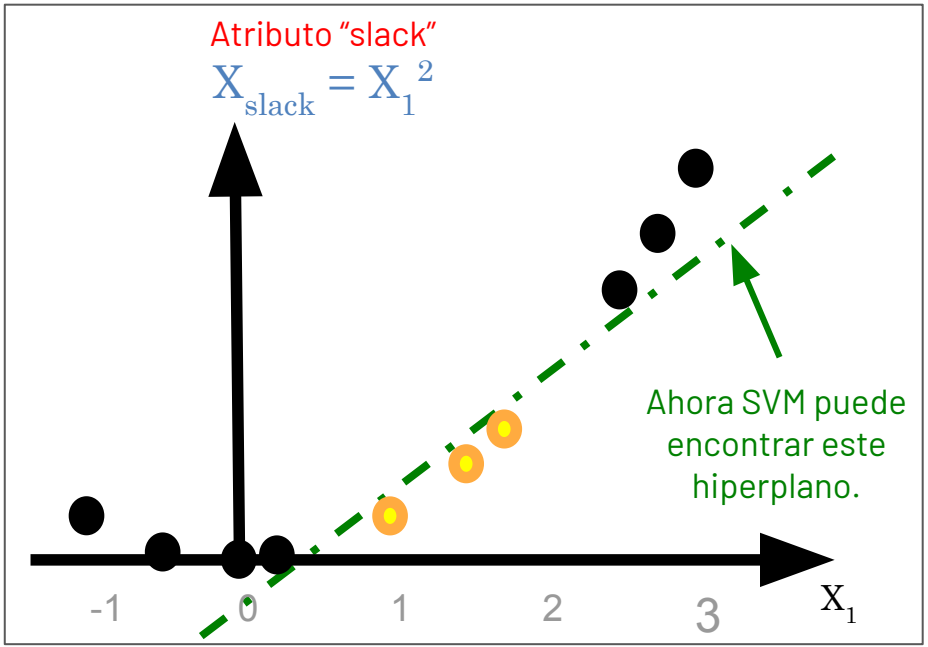
Supongamos para estos ejemplos $\mathbf{x^{(i)} = [x^{(i)}_1]}$

En este caso, es sencillo encontrar un punto que separe bien



¿Qué sucede en un caso como este? Es un dataset linealmente separable?





$$Pred(x^{(i)}) = \arg\max_{c \in Clases} \hat{P}(Y=c|X=x^{(i)})$$

Support Vector Machine (SVM)

Método discriminativo

Notación
(instancia / atributos)
$x^{(i)} = [x^{(i)}_1, x^{(i)}_2, \ldots, x^{(i)}_p]$

Transformación de vectores de atributos.
Ej: $\mathbf{\Phi}([x^{(i)}_1, x^{(i)}_2]) = [x^{(i)}_1, x^{(i)}_2, x^{(i)}_1{}^2, (x^{(i)}_1 * x^{(i)}_2)^2, \ldots]$

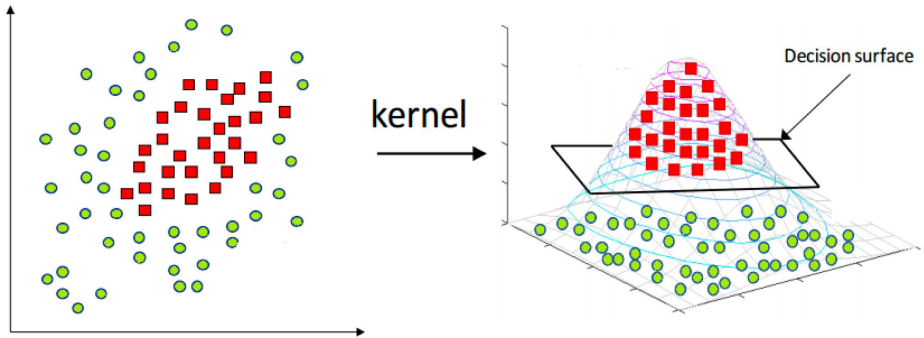
Expandir las transformaciones explícitamente es muy costoso. **Lo evitamos** mediante el **"kernel trick"**.

Kernel: Generalización del producto interno que nos permite operar con nuevos atributos en forma implícita.

$\mathbf{K}(x^{(1)}, x^{(2)}) = \langle \mathbf{\Phi}(x^{(1)}), \mathbf{\Phi}(x^{(2)}) \rangle$ donde $x^{(1)}, x^{(2)}$ son dos instancias.

Si un algoritmo (ej. SVM) puede expresarse en términos de productos internos entre instancias, reemplazamos las apariciones de $\langle \mathbf{\Phi}(x^{(1)}), \mathbf{\Phi}(x^{(2)}) \rangle$ por $\mathbf{K}(\mathbf{x^{(1)}, x^{(2)}})$. Sin nunca tener que computar $\mathbf{\Phi}(x)$.

Así, ejecutamos SVM implícitamente en dimensiones superiores.



[Rizwan, A., Iqbal, N., Ahmad, R., & Kim, D. H. (2021). WR-SVM model based on the margin radius approach for solving the minimum enclosing ball problem in support vector machine classification. *Applied Sciences*.]

$$Pred(x^{(i)}) = \underset{c \in Clases}{\arg\max}\, \hat{P}(Y=c|X=x^{(i)})$$

Support Vector Machine (SVM)

Método discriminativo

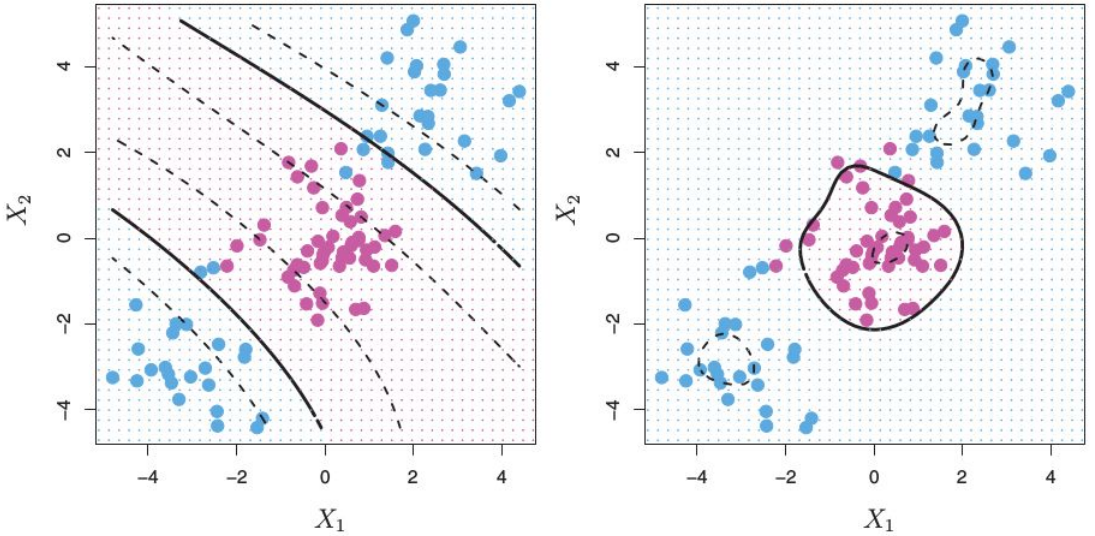


FIGURE 9.9. Left: *An SVM with a polynomial kernel of degree 3 is applied to the non-linear data from Figure 9.8, resulting in a far more appropriate decision rule.* Right: *An SVM with a radial kernel is applied. In this example, either kernel is capable of capturing the decision boundary.*

[ISLR, cap9]

$$Pred(x^{(i)}) = \underset{c \in Clases}{\arg\max}\, \hat{P}(Y=c|X=x^{(i)})$$

Resumen

SVM busca maximizar el margen de separación entre dos clases.

Ejercicios: [ISLR; cap 9]

- Busquen cómo se llama cuando se permiten instancias "incorrectas" dentro de los márgenes.
- Busquen para qué sirve el hiperparámetro ***C***
- Busquen cómo se puede utilizar SVM para **problemas multiclase**.
- Busquen cómo se calcula el **score** de una predicción.
- Kernels: https://scikit-learn.org/stable/modules/svm.html#svm-kernels

Costo computacional:

O(n^3 * d) (pero hay trucos/consideraciones que lo hacen práctico)

Fuente: https://scikit-learn.org/stable/modules/svm.html#complexity

Modelos Generativos

Enfoques (parte 2)

Vimos hasta ahora: Modelos **DISCRIMINATIVOS** clase con la posterior $Pred(x^{(i)}) = \arg\max_{c \in Clases} P(Y = c | X = x^{(i)})$

Otros métodos surgen de aplicar el teorema de bayes:

$$
\begin{aligned}
Pred(x^{(i)}) &= \arg\max_{c \in Clases} P(Y = c | X = x^{(i)}) \\
&= \arg\max_{c \in Clases} \frac{P(Y = c) P(X = x^{(i)} | Y = c)}{P(X = x^{(i)})} \\
&= \arg\max_{c \in Clases} P(Y = c) P(X = x^{(i)} | Y = c) \\
&= \arg\max_{c \in Clases} P(X = x^{(i)}, Y = c)
\end{aligned}
$$

Esto es lo que interesa aproximar ahora

Estos se conocen como modelos **GENERATIVOS**.

¡¡**Conocer** $P(X = x^{(i)} | Y = c)$ **permite generar muestras!!!**

Para pensar: ¿cómo obtener la probabilidad de una predicción en generativos? $\text{Pred}_{\text{proba}}(x^{(i)}, c)$

(algunos) ejemplos de modelos discriminativos:

- **Árboles de decisión** (ya lo vimos)
- **K-vecinos más cercanos** (hoy)
- **Support Vector Machines (SVM)** (hoy)
- **Regresión logística** (prox)
- **Random Forest (y otros ensambles)** (prox)
- **Maximum-entropy Markov models**
- **Conditional random fields**

Teorema de Bayes

$$
\underbrace{P(Y = c \mid X = x^{(i)})}_{\text{posterior}} = \frac{\overbrace{P(X = x^{(i)} \mid Y = c)}^{\text{likelihood (verosimilitud)}} \cdot \overbrace{P(Y = c)}^{\text{prior}}}{\underbrace{P(X = x^{(i)})}_{\text{marginal likelihood}}}
$$

(algunos) ejemplos de modelos generativos:

- **linear discriminant analysis** (LDA) (hoy)
- **gaussian/naive bayes** (hoy)
- **Gaussian mixture model** (GMMs) (prox)
- **Hidden Markov Models** (HMMs)
- **Generative adversarial networks** (GANs),
- **Auto-regressive models** (por ejemplo **GPT3**)
- **Diffusion models**

modelos generativos

Linear y Quadratic Discriminant Analysis

LDA
QDA

Linear / Quadratic Discriminant Analysis

Método generativo

$$
\begin{aligned}
Pred(x^{(i)}) &= \arg\max_{c \in Clases} P(Y{=}c | X{=}x^{(i)}) \\
&= \arg\max_{c \in Clases} \frac{P(Y{=}c)P(X{=}x^{(i)}|Y{=}c)}{P(X{=}x^{(i)})} \\
&= \arg\max_{c \in Clases} P(Y{=}c)P(X{=}x^{(i)}|Y{=}c) \\
&= \arg\max_{c \in Clases} P(Y{=}c)pdf_c(x^{(i)})
\end{aligned}
$$

Para X continua

+ sesgo inductivo
+ datos

$$
\approx \begin{cases}
\arg\max_{c \in Clases} \hat{P}(Y{=}c) f_{norm}(x^{(i)}; \hat{\mu}_c, \hat{\sigma}_c)) & \text{para } x^{(i)} \in R & X|Y{=}c \sim \mathcal{N}(\mu, \sigma) \\
\arg\max_{c \in Clases} \hat{P}(Y{=}c) f^d_{norm}(x^{(i)}; \hat{\boldsymbol{\mu}}_{\boldsymbol{c}}, \hat{\boldsymbol{\Sigma}}_{\boldsymbol{c}})) & \text{para } x^{(i)} \in R^d & X|Y{=}c \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})
\end{cases}
$$

$$
f_{norm}(x; \mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}
$$

$$
f^d_{norm}(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{\sqrt{(2\pi)^k |\boldsymbol{\Sigma}|}} e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x}-\boldsymbol{\mu})}
$$

Linear / Quadratic Discriminant Analysis

Método generativo

(de la diapo anterior, caso general, multidimensional)

$$Pred(x^{(i)}) = \arg\max_{c \in Clases} \hat{P}(Y{=}c) f^{d}_{norm}(x^{(i)}; \hat{\boldsymbol{\mu}}_{\boldsymbol{c}}, \hat{\boldsymbol{\Sigma}}_{\boldsymbol{c}})) \quad \text{para } x^{(i)} \in R^{d}$$

- La suposición de estos métodos es que $\mathbf{X \mid Y{=}c}$ sigue una distribución normal $\mathbf{N}(\boldsymbol{\mu}_{\mathbf{c}}, \boldsymbol{\Sigma}_{\mathbf{c}})$.
  Es decir, los puntos en cada clase fueron generados por distribuciones normales distintas.

- Con esta suposición, para modelar la distribución de instancias de cada clase, $P(X{=}x \mid Y{=}c)$, alcanza con **estimar** $\boldsymbol{\mu}_{\mathbf{c}}$, $\boldsymbol{\Sigma}_{\mathbf{c}}$, $\mathbf{P(Y{=}c)}$ usando los datos de entrenamiento.

- Encontrar la clase $\mathbf{c}$ con probabilidad **máxima a posteriori** sale directo (usando las fórmulas de la diapo anterior).

Si la matriz de covarianza (o los desvíos en univariado) se suponen **iguales para toda clase,** este método se llama: **Linear Discriminant Analysis (LDA)**. Si no: **Quadratic Discriminant Analysis (QDA)**

- $\boldsymbol{\mu}_{\mathbf{c}}$ y $\boldsymbol{\Sigma}_{\mathbf{c}}$ pueden ser estimados mediante el método de máxima verosimilitud (con la corrección de Bessel):

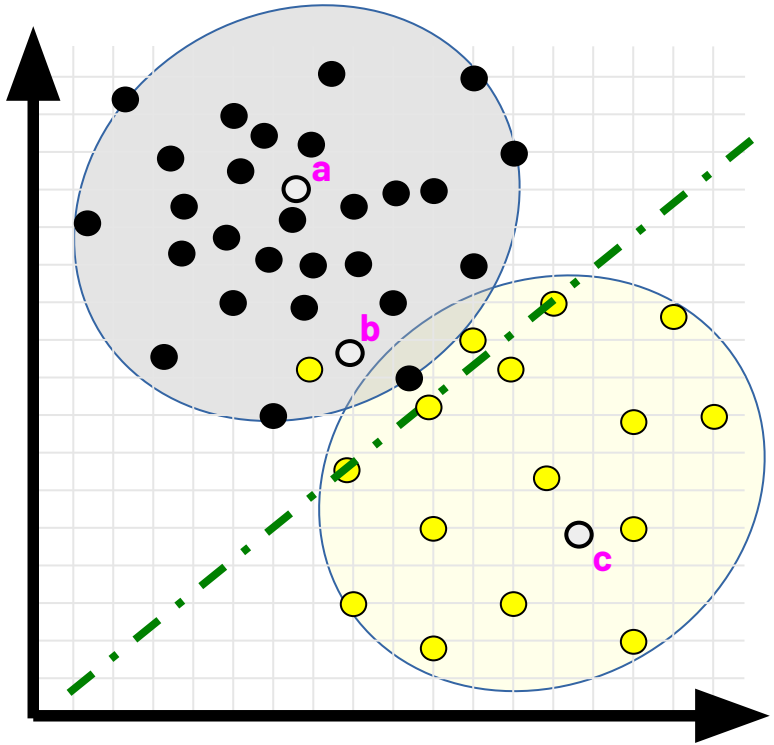
$$\hat{\mu}_c = \frac{1}{n_c} \sum_{x^{(i)} \in \{D \mid y^{(i)} = c\}} x^{(i)}$$

$$\hat{\boldsymbol{\Sigma}}_{\boldsymbol{c}} = \frac{1}{n_c - 1} \sum_{x^{(i)} \in \{D \mid y^{(i)} = c\}} (\mathbf{x^{(i)}} - \hat{\boldsymbol{\mu}}_{\boldsymbol{c}})(\mathbf{x^{(i)}} - \hat{\boldsymbol{\mu}}_{\boldsymbol{c}})^{\top}$$

$$\hat{P}(Y{=}c) = \frac{n_c}{n}$$

Linear / Quadratic Discriminant Analysis

Método generativo



Esquematización (LDA)

¿De qué clase serán **a, b** y **c**?

1) Modelar la distribución (normal) de puntos de cada clase;

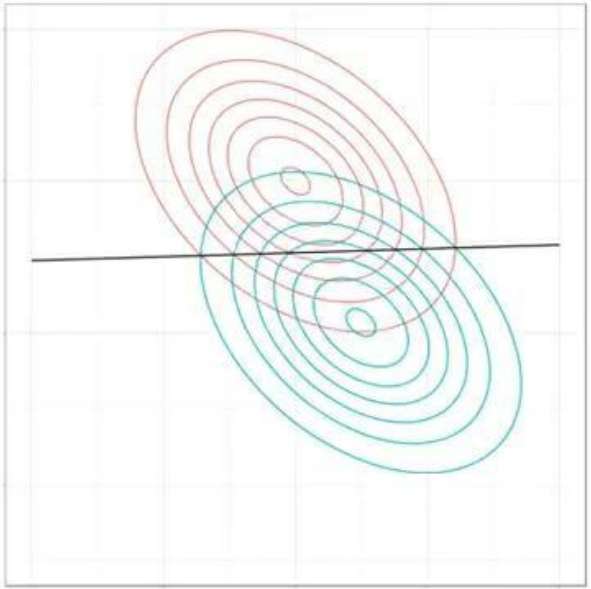
2) Asignar nuevas instancias a la clase que tiene probabilidad máxima a posteriori (MAP).

3) La recta verde es la frontera de decisión entre las 2 clases (en este caso, es una recta, es decir LDA). **¿Por qué no está justo al medio?**

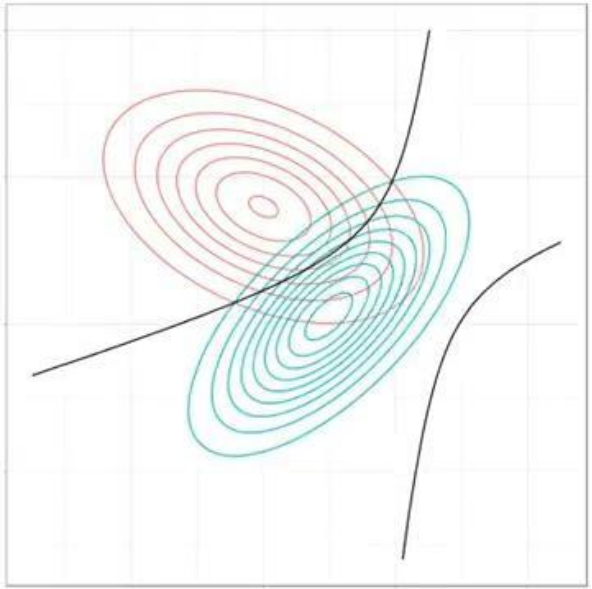
Linear / Quadratic Discriminant Analysis

Método generativo

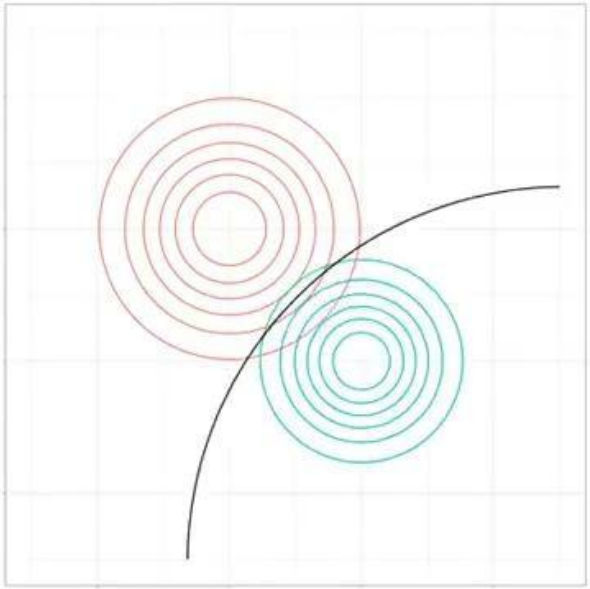
LDA variando la media de una clase



QDA variando la media de una clase



QDA variando la varianza de una clase

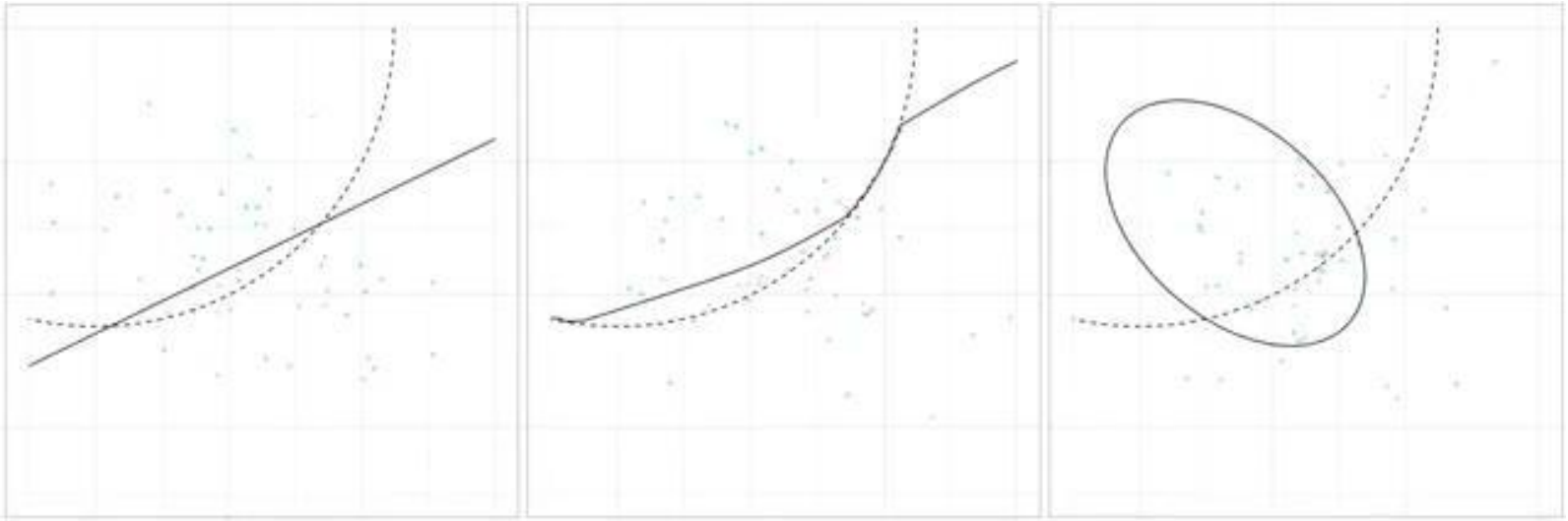


Fuente: https://mathformachines.com/posts/discriminant-analysis/

Linear / Quadratic Discriminant Analysis

Método generativo

LDA | QDA (con mat. covarianza diagonal) | QDA general



La línea punteada representa la frontera de decisión óptima (Bayes Optimal Classifier)

Fuente: https://mathformachines.com/posts/discriminant-analysis/

VER hilo de twitter (para más animaciones como esta): Animated-machine-learning-classifiers

modelos generativos

Naive Bayes

Naive Bayes (versión Gaussiana)

Método generativo

$$Pred(x^{(i)}) = \arg\max_{c \in Clases} P(Y{=}c | X{=}x^{(i)})$$

$$= \arg\max_{c \in Clases} \frac{P(Y{=}c)P(X{=}x^{(i)}|Y{=}c)}{P(X{=}x^{(i)})}$$

$$= \arg\max_{c \in Clases} P(Y{=}c)P(X{=}x^{(i)}|Y{=}c)$$

$$= \arg\max_{c \in Clases} P(Y{=}c)P(X_1 = x_1^{(i)} \wedge \cdots \wedge X_p = x_p^{(i)}|Y{=}c)$$

$$\approx \arg\max_{c \in Clases} P(Y{=}c) \prod_{j=1}^{p} P(X_i = x_j^{(i)}|Y{=}c)$$

$$= \arg\max_{c \in Clases} P(Y{=}c) \prod_{j=1}^{p} pdf_c(x_j^{(i)})$$

$$\approx \arg\max_{c \in Clases} \hat{P}(Y{=}c) \prod_{j=1}^{p} f_{norm}(x_j^{(i)}; \hat{\mu_{c,j}}, \hat{\sigma_j}))$$

+ sesgo inductivo **(suposición "Naive")** ¿Cuál es?

Para X continua

+ sesgo inductivo **Gaussian** Naive Bayes

Una media por clase, por atributo.

Un desvío por atributo (en general compartido entre clases)

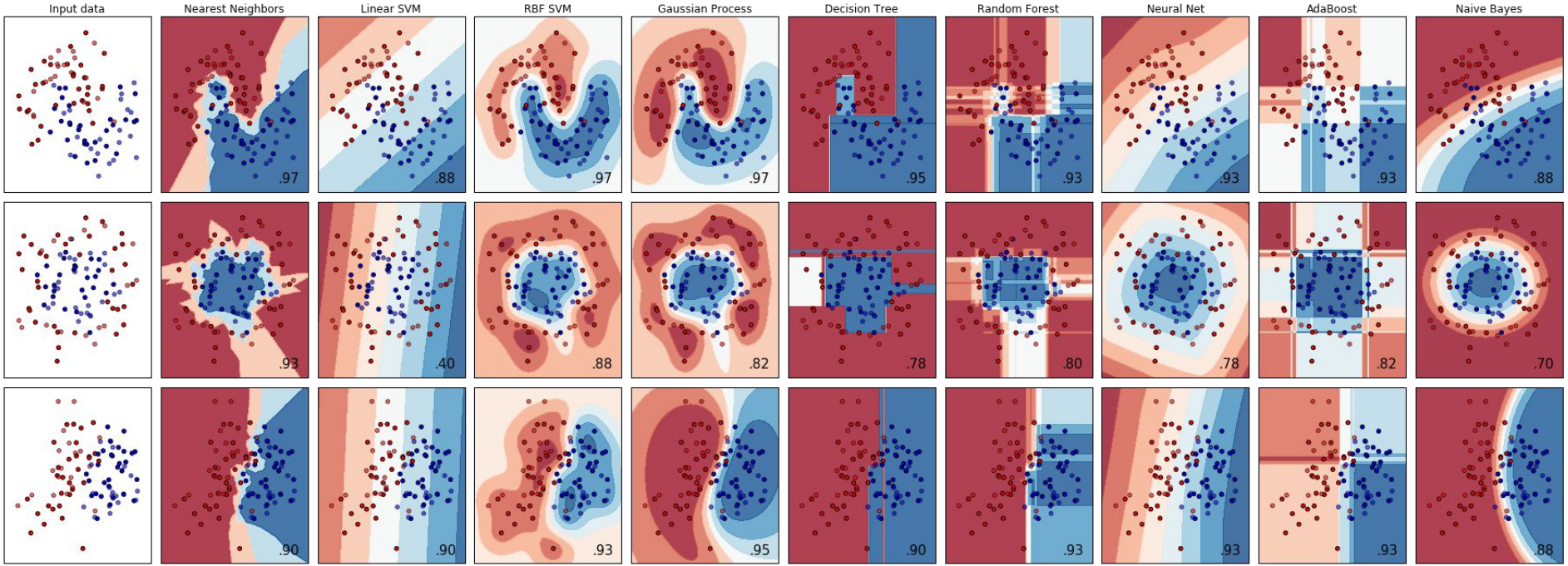
Nota: Modelos paramétricos vs no paramétricos

(Murphy pp.16)

Tiene el modelo una **cantidad fija de parámetros (modelos paramétricos)** o esta cantidad **crece con la cantidad de datos** de entrenamiento **(modelos no paramétricos)**?

¿Cuáles suelen tener “más” sesgo inductivo?

Siempre recordar...



Fuente: https://scikit-learn.org/stable/auto_examples/classification/plot_classifier_comparison.html

Resumen

- Bayes Optimal Classifier
- Modelos discriminativos vs generativos
- K Vecinos más Cercanos (KNN)
- Support Vector Machines (SVM)
    - (vimos muy poco, es todo un mundo este tema: K clases? Probas? etc)
- Linear Discriminant Analysis (LDA) / Quadratic Discriminant Analysis (QDA)
- Naive Bayes Classifier / Gaussian Naive Bayes

Próximos temas: Sesgo y varianza de algoritmos; Ensambles de modelos.

(algunos) ejemplos de modelos discriminativos:

- **Árboles de decisión** (ya lo vimos)
- **K-vecinos más cercanos** (hoy)
- **Support Vector Machines (SVM)** (hoy)
- **Regresión logística** (prox)
- **Random Forest (y otros ensambles)** (prox)
- **Maximum-entropy Markov models**
- **Conditional random fields**

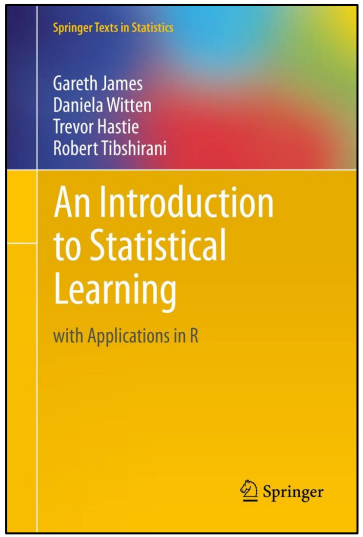
(algunos) ejemplos de modelos generativos:

- **linear discriminant analysis** (LDA) (hoy)
- **gaussian/naive bayes** (hoy)
- **Gaussian mixture model** (GMMs) (prox)
- **Hidden Markov Models** (HMMs)
- **Generative adversarial networks** (GANs),
- **Auto-regressive models** (por ejemplo **GPT3**)
- **Diffusion models**

Tarea

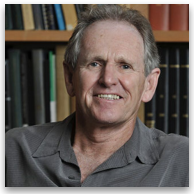
- Obligatorio: Leer la Sec. 4.4 **Generative Models for Classification** del ISLR.
- Obligatorio: Cap 9 hasta 9.4. **Support Vector Machines** del ISLR.
- Recomendado: Leer la Sec. 4.5 A **Comparison of Classification Methods** del ISLR.

ISLR




Gareth James


Daniela Witten


Trevor Hastie


Robert Tibshirani

Download:
https://hastie.su.domains/ISLR2/ISLRv2_website.pdf