

# Índice

1	Sandbox .....	1
2	Notación .....	1
3	Preliminares .....	2
3.1	El problema de clasificación .....	2
3.2	Estimación de Densidad por Núcleos .....	3
3.3	Clasificación en variedades .....	6
3.4	Aprendizaje de distancias .....	9
3.5	Todo junto: .....	11
4	Resultados .....	15
4.1	Propuesta Original .....	46
4.2	Evaluación .....	47
5	Análisis de Resultados .....	58
6	Comentarios finales .....	58
7	Referencias .....	58
8	Código Fuente .....	58
	Bibliografía .....	59

## 1 Sandbox

```
print("hello world")
print("iolii mundo")
```

Listado 1: Un cacho e' código

Definición 1.1 (La mar en coche): A natural number is called a **prime number** if it is greater than 1 and cannot be written as the product of two smaller natural numbers.

*Observación:* O sea, sería tremendo hundirse en medio de la mar  
Como se explica en [Definición 1.1](#), es muy arriesgado cruzar el océano en un auto. O sea, no.

## 2 Notación

$\mathbb{R}$  los números reales

$d_x$

$\mathbb{R}^{d_x}$

$[k]$  el conjunto de los  $k$  números enteros,  $\{1, \dots, k\}$

$\mathcal{M}$

$\mathbf{H}$

$\|\cdot\|$

$\{\mathbf{X}\}$

$X_{i,j}$

$\mathbb{1}(\cdot)$  la función indicadora

$\Pr(x \in A)$

$\mathbb{P}$

$\mathbb{1}\left(\frac{x}{y}\right)$

## 3 Preliminares

### 3.1 El problema de clasificación

#### 3.1.1 Definición y vocabulario

[ESL §2.2]

- El *aprendizaje estadístico supervisado* busca estimar (aprender) una variable *respuesta* a partir de cierta(s) variable(s) *predictora(s)*.
- Cuando la *respuesta* es una variable *cualitativa*, el problema de asignar cada observación  $x$  a una clase  $G \in \mathcal{G} = \{g^1, \dots, g^K\}$  se denomina *de clasificación*.
- Un *clasificador* es una función  $\hat{G}(x)$  que para cada observación  $x$ , intenta aproximar su verdadera clase  $g$  por  $\hat{g}$  («ge sombrero»).
- Para construir  $\hat{G}$ , contamos con un *conjunto de entrenamiento* de pares  $(x_i, g_i), i \in \{1, \dots, N\}$  conocidos. Típicamente, las clases serán MECE, y las observaciones  $X \in \mathbb{R}^p$ .

#### 3.1.2 Clasificador de Bayes

Una posible estrategia de clasificación consiste en asignarle a cada observación  $x_0$ , la clase más probable en ese punto, dada la información disponible.

$$\hat{G}(x) = \arg \max_{g \in \mathcal{G}} \Pr(G = g | X = x) \quad (1)$$

Esta razonable solución es conocida como el *clasificador de Bayes*, y se puede reescribir usando la regla homónima como

$$\begin{aligned} \hat{G}(x) = g_i &\Leftrightarrow \Pr(g_i | X = x) = \max_{g \in \mathcal{G}} \Pr(G = g | X = x) \\ &= \max_{g \in \mathcal{G}} \Pr(X = x | G = g) \times \Pr(G = g) \end{aligned} \quad (2)$$

#### 3.1.3 Clasificadores «suaves» y «duros»

- Un clasificador que responda «*¿qué clase* es la que más probablemente contenga esta observación» es un clasificador «duro».

- Un clasificador que además puede responder «*¿cuán probable* es que esta observación pertenezca a cada clase  $g_j$ ?» es un clasificador «suave».
- La regla de Bayes para clasificación nos puede dar un clasificador duro al maximizar la probabilidad; más aún, también puede construir un clasificador suave:

$$\begin{aligned}\widehat{\Pr}(G = g_i | X = x) &= \frac{\widehat{\Pr}(x|G = g_i) \times \widehat{\Pr}(G = g_i)}{\widehat{\Pr}(X = x)} \\ &= \frac{\widehat{\Pr}(x|G = g_i) \times \widehat{\Pr}(G = g_i)}{\sum_{k \in [K]} \widehat{\Pr}(X = x, G = g_k)}\end{aligned}\tag{3}$$

## 3.2 Estimación de Densidad por Núcleos

### 3.2.1 Clasificador de Bayes empírico

- Si el conjunto de entrenamiento  $\{(x_1, g_1), \dots, (x_N, g_N)\}$  proviene de un muestreo aleatorio uniforme, las probabilidades de clase  $\pi_i = \Pr(G = g^{(i)})$  se pueden aproximar razonablemente por las proporciones muestrales  $\hat{\pi}_i = \#\{g_j : g_j = g^{(i)}\}/N$
- Resta hallar una aproximación  $\Pr(x|G = g)$  para cada clase, ya sea a través de una función de densidad, de distribución, u otra manera.

### 3.2.2 Estimación unidimensional

[ESL §6.6, Parzen 1962]

Para fijar ideas, asumamos que  $X \in \mathbb{R}$  y consideremos la estimación de densidad en una única clase para la que contamos con  $N$  ejemplos  $\{x_1, \dots, x_N\}$ . Una aproximación  $\hat{f}$  directa sería (1)

$$\hat{f}(x_0) = \frac{\#\{x_i \in \mathcal{N}(x_0)\}}{N \times h}\tag{4}$$

donde  $\mathcal{N}$  es un vecindario métrico de  $x_0$  de diámetro  $h$ . Esta estimación es irregular, con saltos discretos en el numerador, por lo que se prefiere el estimador suavizado por núcleos de Parzen-Rosenblatt

$$\hat{f}(x_0) = \frac{1}{N} \sum_{i=1}^N K(x_0, x_i)\tag{5}$$

### 3.2.3 Función núcleo o «kernel»

Se dice que  $K(x) : \mathbb{R} \rightarrow \mathbb{R}$  es una *función núcleo* si

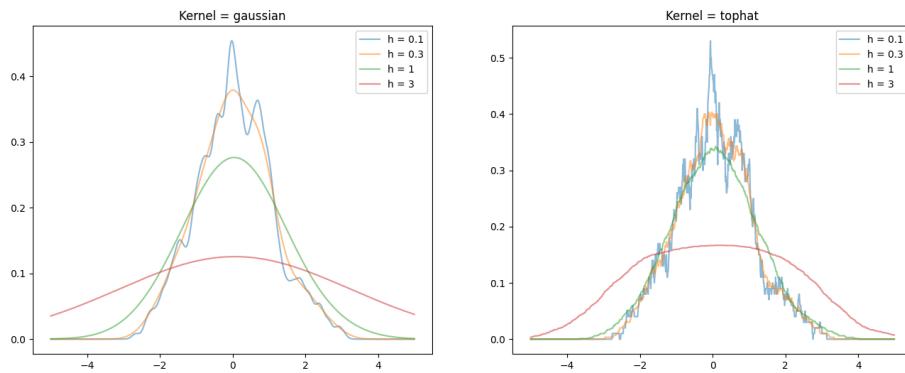
- toma valores reales no negativos:  $K(u) \geq 0 \forall u \in \text{sop}K$ ,
- está normalizada:  $\int_{-\infty}^{+\infty} K(u)du = 1$ ,
- es simétrica:  $K(u) = K(-u)$  y
- alcanza su máximo en el centro:  $\max_u K(u) = K(0)$

Observación 1: Todas las funciones de densidad simétricas centradas en 0 son núcleos; en particular, la densidad «normal estándar»  $\phi(x) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right)$  lo es.

Observación 2: Si  $K(u)$  es un núcleo, entonces  $K_h(u) = \frac{1}{h} K(\frac{u}{h})$  también lo es.

Observación 3: Si  $\mathbb{1}(\cdot)$  es la función indicadora, resulta que  $U_h(x) = \frac{1}{h} \mathbb{1}\left(-\frac{h}{2} < x < \frac{h}{2}\right)$  es un núcleo válido, y el estimador de Ecuación 5 con núcleo  $U_h$  devuelve el estimador Ecuación 4

### 3.2.4 Núcleo uniforme



### 3.2.5 Clasificador de densidad por núcleos

[ESL §6.6.2]

Si  $\hat{f}_k, k \in 1, \dots, K$  son estimadores de densidad por núcleos<sup>1</sup> según Ecuación 5, la regla de Bayes nos provee un clasificador suave

$$\begin{aligned} \widehat{\Pr}(G = g_i | X = x) &= \frac{\widehat{\Pr}(x|G = g_i) \times \widehat{\Pr}(G = g_i)}{\widehat{\Pr}(X = x)} \\ &= \frac{\hat{\pi}_i \hat{f}_i(x)}{\sum_{k=1}^K \hat{\pi}_k \hat{f}_k(x)} \end{aligned} \tag{6}$$

### 3.2.6 Interludio: Naive Bayes

[ESL §6.6.3]

¿Y si las  $X$  son multivariadas ( $X \in \mathbb{R}^d, d \geq 2$ )? ¿Se puede adaptar el clasificador?

Sí, pero es complejo. Un camino sencillo: asumir que condicional a cada clase  $G = j$ , los predictores  $X_1, X_2, \dots, X_p$  se distribuyen independientemente entre sí.

---

<sup>1</sup>KDEs ó *Kernel Density Estimators*, por sus siglas en inglés

$$f_j(X) = \prod_{i=1}^p f_{j,i}(X_i) \quad (7)$$

Cada densidad marginal  $f_{j,i}$  condicional a la clase se puede estimar usando KDE univariado, y hasta se puede aplicar - usando histogramas - cuando algunas componentes  $X_i$  son discretas.

A este procedimiento, se lo conoce como «Naive Bayes».

### 3.2.7 KDE multivariado

[Wand & Jones 1995 §4]

En su forma más general, estimador de densidad por núcleos  $d$ -variado es

$$\hat{f}(x; \mathbf{H}) = N^{-1} \sum_{i=1}^N K_{\mathbf{H}}(x - x_i) \quad (8)$$

donde

- $\mathbf{H} \in \mathbb{R}^{d \times d}$  es una matriz simétrica def. pos. análoga a la ventana  $h \in \mathbb{R}$  para  $d = 1$ ,
- $K_{\mathbf{H}}(t) = |\det \mathbf{H}|^{-\frac{1}{2}} K(\mathbf{H}^{-\frac{1}{2}} t)$
- $K$  es una función núcleo  $d$ -variada tal que  $\int K(\mathbf{x}) d\mathbf{x} = 1$

Típicamente,  $K$  es la densidad normal multivariada

$$\Phi(x) : \mathbb{R}^d \rightarrow \mathbb{R} = (2\pi)^{-\frac{d}{2}} \exp\left(-\frac{\|x\|^2}{2}\right) \quad (9)$$

### 3.2.8 Dificultades: elección de $\mathbf{H}$

Sean las clases de matrices pertenecientes a  $\mathbb{R}^{d \times d}$  ...

- $\mathcal{F}$ , de matrices simétricas definidas positivas,
- $\mathcal{D}$ , de matrices diagonales definidas positivas ( $\mathcal{D} \subseteq \mathcal{F}$ ) y
- $\mathcal{S}$ , de múltiplos escalares de la identidad:  $\mathcal{S} = \{h^2 \mathbf{I} : h > 0\} \subseteq \mathcal{D}$

Aún tomando una única  $\mathbf{H}$  para *toda* la muestra,  $\mathbf{H} \in \dots$

- $\mathcal{F}$ , requiere definir  $\binom{d}{2} = \frac{d(d-1)}{2}$  parámetros de ventana,
- $\mathcal{D}$  requiere  $d$  parámetros, y
- $\mathcal{S}$  tiene un único parámetro  $h$ .

A priori no es posible saber qué parametrización conviene, pero en general  $\mathbf{H} \in \mathcal{D}$  parece un compromiso razonable: no se pierde demasiado contra  $\mathcal{F}$ , pero tampoco se padece la «rigidez» de  $\mathbf{H} \in \mathcal{S}$ .

### 3.2.9 Dificultades: La maldición de la dimensionalidad

[ESL §2.5, Wand & Jones 1995 §4.9 ej 4.1]

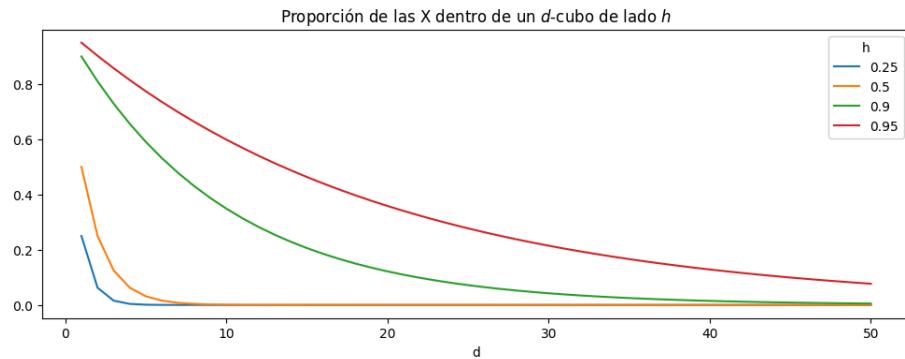
Sean  $X_i \stackrel{\text{iid}}{\sim} \text{Uniforme}([-1, 1]^d)$ ,  $i \in \{1, \dots, N\}$ , y consideremos la estimación de la densidad en el origen,  $f(\mathbf{0})$ . Suponga que el núcleo  $K_{\mathbf{H}}$  es un «núcleo producto» basado en la distribución univariada Uniforme(-1, 1),

y  $\mathbf{H} = h^2 \mathbf{I}$ . Derive una expresión para la proporción esperada de puntos incluidos dentro del soporte del núcleo  $K_{\mathbf{H}}$  para  $h, d$ . arbitrarios.

(... interludio de pizarrón ...)

$$\begin{aligned}\Pr(X \in [-h, h]^d) &= \Pr(\cap_{i=1}^d |X_i| \leq h) \\ \Pr(X \in [-0.95, 0.95]^{50}) &\approx 0.0077\end{aligned}\tag{10}$$

### 3.2.10 Dificultades: La maldición de la dimensionalidad



Para  $h \leq 0.5$ ,  $\Pr(\cdot) < 1 \times 10^{-15}$ . Aún para  $h = 0.95$ ,  $\Pr(\cdot) \approx 0.0077$  😱

## 3.3 Clasificación en variedades

### 3.3.1 La hipótesis de la variedad («manifold hypothesis»)

[Bengio Repr learning] [Bengio en Reddit]

La hipótesis de la variedad postula que los datos  $X \in \mathbb{R}^{d_X}$  muestreados soportados en un espacio de alta dimensionalidad<sup>2</sup>. tenderán a concentrarse sobre una *variedad*  $\mathcal{M}$ , potencialmente de mucha menor dimensión  $d_{\mathcal{M}} \ll d_X$ , embebida en el espacio original  $\mathcal{M} \subseteq \mathbb{R}^{d_X}$ .

- Well suited for AI tasks such as those involving images, sounds or text, for which most uniformly sampled input configurations are unlike natural stimuli.
- archetypal manifold modeling algorithm is, not surprisingly, also the archetypal low dimensional representation learning algorithm: Principal Component Analysis, which models a linear manifold.
- Data manifold for complex real world domains are however expected to be strongly nonlinear.

### 3.3.2 IRL




---

<sup>2</sup>E.g.: imágenes, audio, video, secuencias de nucleótidos



Pero: ¿en qué variedad vive un dígito, o su trazo, o una canción? 🎵

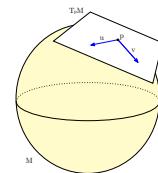
### 3.3.3 Interludio: Variedades de Riemann [Wikipedia]

Una variedad  $d$ -dimensional  $\mathcal{M}$  es un espacio *topológico* tal que cada punto  $p \in \mathcal{M}$  tiene un vecindario  $U$  que resulta *homeomórfico* a un conjunto abierto en  $\mathbb{R}^d$

- topológico: se puede definir cercanía (pero no necesariamente distancia), permite definir funciones continuas y límites
- homeomórfico a  $\mathbb{R}^d$ : para cada punto  $p \in \mathcal{M}$ , existe un mapa *biyectivo* y *suave* entre el vecindario de  $p$  y  $\mathbb{R}^d$ . El conjunto de tales mapas se denomina *atlas*.

Sea  $T_p\mathcal{M}$  el *espacio tangente* a un punto  $p \in \mathcal{M}$ , y  $g_p : T_p\mathcal{M} \times T_p\mathcal{M} \rightarrow \mathbb{R}$  una forma *bilinear pos. def.* para cada  $p$  que induce una *norma*  $\|v\|_p = \sqrt{g_p(v,v)}$ .

Decimos entonces que  $g_p$  es una métrica Riemanniana y el par  $(\mathcal{M}, g)$  es una variedad de Riemann, donde las nociones de *distancia*, *ángulo* y *geodésica* están bien definidas.



### 3.3.4 KDE en variedades de Riemann [Pelletier 2005]

- Sea  $(\mathcal{M}, g)$  una variedad de Riemann compacta y sin frontera de dimensión  $d$ , y usemos  $d_g$  para denotar la distancia de Riemann.
- Sea  $K$  un *núcleo isotrópico en  $\mathcal{M}$  soportado en la bola unitaria* (cf. cond. (i)-(v))
- Sean  $p, q \in \mathcal{M}$ , y  $\theta_p(q)$  la *función de densidad de volumen en  $\mathcal{M}^3$*   
Luego, el estimador de densidad para  $X_i \stackrel{\text{iid}}{\sim} f$  es  $f_{N,K} : \mathcal{M} \rightarrow \mathbb{R}$  que a cada  $p \in \mathcal{M}$  le asocia el valor

---

<sup>3</sup>Ardua definición! Algo así como el cociente entre las medida de volumen en  $\mathcal{M}$ , y su transformación via el mapa local a  $\mathbb{R}^d$

$$f_{N,K}(p) = N^{-1} \sum_{i=1}^N K_h(p, X_i) = N^{-1} \sum_{i=1}^N \frac{1}{h^d} \frac{1}{\theta_{X_i}(p)} K\left(\frac{d_g(p, X_i)}{h}\right) \quad (11)$$

con la restricción de que la ventana  $h \leq h_0 \leq \text{inj}(\mathcal{M})$ , el *radio de inyección* de  $\mathcal{M}$ <sup>4</sup>

### 3.3.5 Interludio: densidad de volumen en la esfera [Henry y Rodríguez, 2009]

En «*Kernel Density Estimation on Riemannian Manifolds: Asymptotic Results*» (2009), Guillermo Henry y Daniela Rodriguez estudian algunas propiedades asintóticas de este estimador, y las ejemplifican con datos de sitios volcánicos en la superficie terrestre. En particular, calculan la densidad de volumen  $\theta_{p(q)}$

$$\begin{aligned} \theta_p(q) &= R \frac{|\sin(d_g(p, q)/R)|}{d_g(p, q)} \quad \text{if } q \neq p, -p \quad \text{and} \\ \theta_p(p) &= 1. \end{aligned}$$

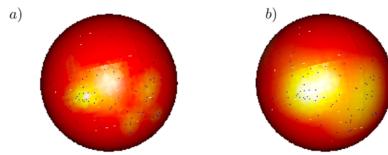


Fig. 1 The nonparametric density estimator using different bandwidth, **a**  $h = 1500$  and **b**  $h = 3000$

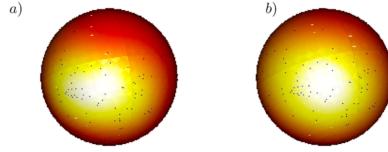


Fig. 2 The nonparametric density estimator using different bandwidth, **a**  $h = 5000$  and **b**  $h = 7000$

### 3.3.6 Clasificación en variedades [Loubes y Pelletier 2008]

¡Clasificador de Bayes + KDE en Variedades = Clasificación (suave o dura) en variedades!

Plantean una regla de clasificación  $\hat{G}$  para 2 clases adaptable a K clases de forma directa. Sea  $p \in \mathcal{M}$  una variedad riemanniana como antes, y  $\{(x_1, g_1), \dots, (x_N, g_N)\}$  nuestras observaciones y sus clases. Luego,

$$\hat{G}(p) = \arg \max_{g \in \mathcal{G}} \sum_{i=1}^N \mathbb{1}(g_i = g) K_h(p, X_i) \quad (12)$$

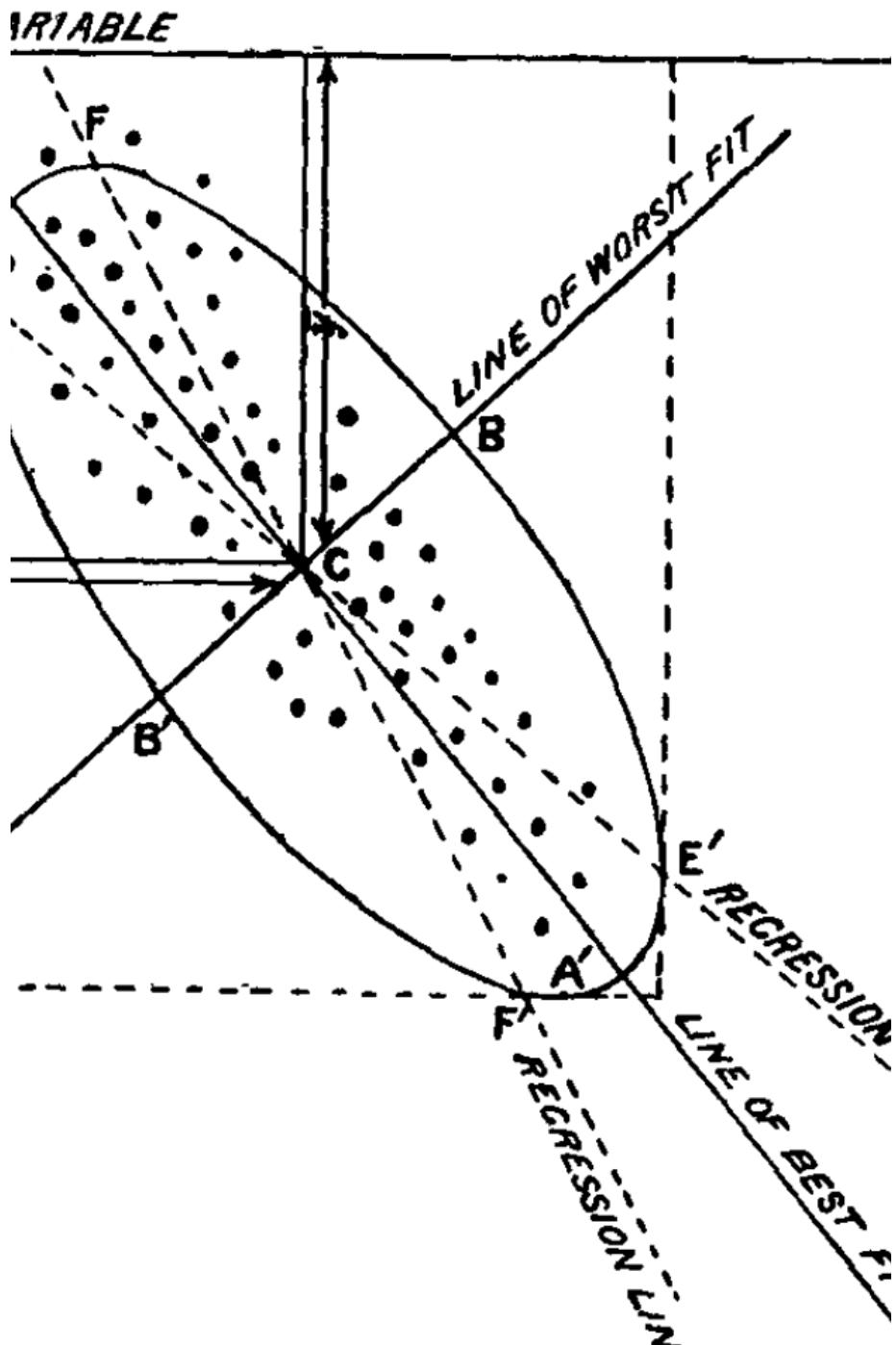
Pero... ¿y si la variedad es desconocida?

---

<sup>4</sup>el ínfimo entre el supremo del radio de una bola en cada  $p$  tal que su mapa es un difeomorfismo

### 3.4 Aprendizaje de distancias

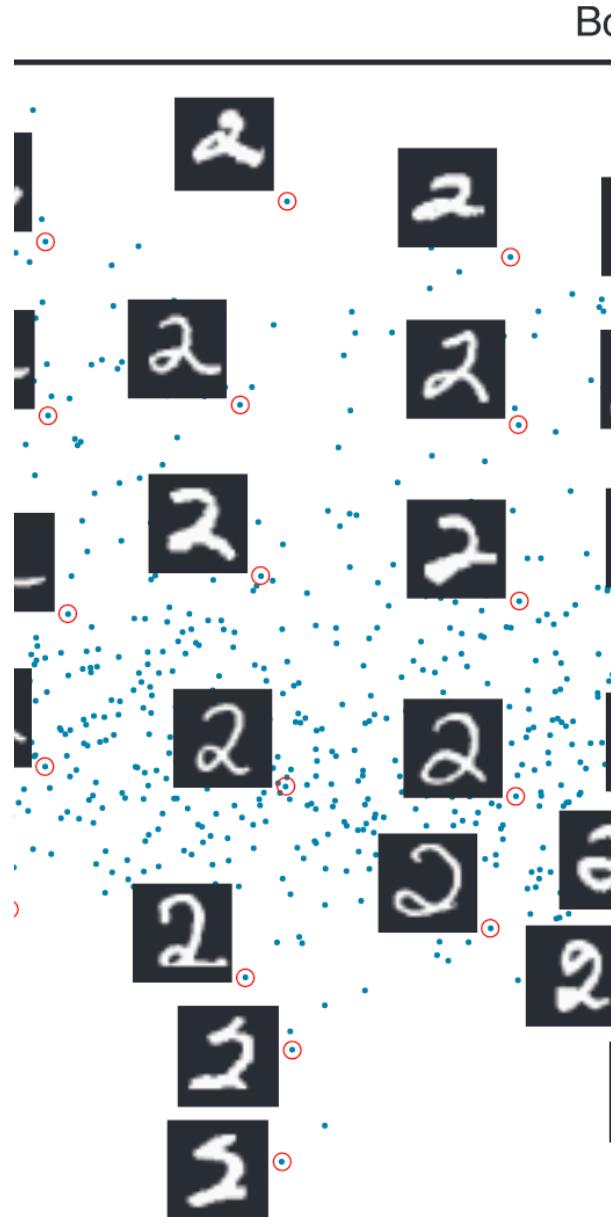
#### 3.4.1 El ejemplo canónica: Análisis de Componentes Principales (PCA)



Karl Pearson (1901), «*LIII. On lines and planes of closest fit to systems of points in space.*»

### 3.4.2 El algoritmo más *cool*: Isomap

1. Construya el grafo de  $k, \varepsilon$ -vecinos,  $\text{NN} = (\mathbf{X}, E)$
2. Compute los caminos mínimos - las geodésicas entre observaciones,  $d_{\text{NN}}(x, y)$ .
3. Construya una representación («*embedding*»)  $d^*$ -dimensional que minimice la discrepancia («*stress*») entre  $d_{\text{NN}}$  y la distancia euclídea en  $\mathbb{R}^{d^*}$



[Tenenbaum et al (2000), «*A Global Geometric Framework for Nonlinear Dimensionality Reduction*»]

### 3.4.3 Distancia de Fermat [Groisman, Jonckheere, Sapienza (2019); Little et al (2021)]

*We tackle the problem of learning a distance between points, able to capture both the geometry of the manifold and the underlying density. We define such a sample distance and prove the convergence, as the sample size goes to infinity, to a macroscopic one that we call Fermat distance as it minimizes a path functional, resembling Fermat principle in optics.*

— P. Groisman et al (2019)

Sea  $f$  una función continua y positiva,  $\beta \geq 0$  y  $x, y \in S \subseteq \mathbb{R}^d$ . Definimos la *Distancia de Fermat*  $\mathcal{D}_{f,\beta}(x, y)$  como:

$$\mathcal{T}_{f,\beta}(\gamma) = \int_{\gamma} f^{-\beta}, \quad \mathcal{D}_{f,\beta}(x, y) = \inf_{\gamma \in \Gamma} \mathcal{T}_{f,\beta}(\gamma) \quad \text{!(?)} \quad (13)$$

... donde el ínfimo se toma sobre el conjunto  $\Gamma$  de todos los caminos rectificables entre  $x$  e  $y$  contenidos en  $\overline{S}$ , la clausura de  $S$ , y la integral es entendida con respecto a la longitud de arco dada por la distancia euclídea.

### 3.4.4 Distancia de Fermat muestral

Para  $\alpha \geq 1$  y  $x, y \in \mathbb{R}^d$ , la *Distancia Muestral de Fermat* se define como

$$D_{\mathbf{X}, \alpha} = \inf \left\{ \sum_{j=1}^{K-1} \|q_{j+1} - q_j\|^{\alpha} : (q_1, \dots, q_K) \text{ es un camino de } x \text{ a } y, K(\underline{M}) \right\}$$

donde los  $q_j$  son elementos de la muestra  $\mathbf{X}$ . Nótese que  $D_{\mathbf{X}, \alpha}$  satisface la desigualdad triangular, define una métrica sobre  $\mathbf{X}$  y una pseudométrica sobre  $\mathbb{R}^d$ .

En su paper, Groisman et al. muestran que

$$\lim_{N \rightarrow \infty} n^{\beta} D_{\mathbf{X}_n, \alpha}(x, y) = \mu \mathcal{D}_{f, \beta}(x, y) \quad (15)$$

donde  $\beta = (a - 1)/d$ ,  $n \geq n_0$  y  $\mu$  es una constante adecuada.

¡Esta sí la podemos aprender de los datos! 💪

## 3.5 Todo junto:

Clasificación en variedades desconocidas por estimación de densidad por núcleos con Distancia de Fermat Muestral

### 3.5.1 Algunas dudas

- Entrenar el clasificador por validación cruzada está OK: como  $\mathbf{X}_{\text{train}} \subseteq \mathbf{X}$  y  $\mathbf{X}_{\text{test}} \subseteq \mathbf{X}$ , se sigue que  $\forall (a, b) \in \{\mathbf{X}_{\text{train}} \times \mathbf{X}_{\text{test}}\} \subseteq \{\mathbf{X} \times \mathbf{X}\}$  y  $D_{\mathbf{X}, \alpha}(a, b)$  está bien definida. ¿Cómo sé la distancia muestral de una nueva observación  $x_0$ , a los elementos de cada clase?

Para cada una de las  $g_i \in \mathcal{G}$  clases, definimos el conjunto

$$Q_i = \{x_0\} \cup \{x_j : x_j \in \mathbf{X}, g_j = g_i, j \in \{1, \dots, N\}\} \quad (16)$$

y calculamos  $D_{Q_i, \alpha}(x_0, \cdot)$

### 3.5.2 Algunas dudas

- El clasificador de Loubes & Pelletier asume que todas las clases están soportadas en la misma variedad  $\mathcal{M}$ . ¿Quién dice que ello vale para las diferentes clases?

¡Nadie! Pero

1. No hace falta dicho supuesto, y en el peor de los casos, podemos asumir que la unión de las clases está soportada en *cierta* variedad de Riemann, que resulta de (¿la clausura de?) la unión de sus soportes individuales.
2. Sí es cierto que si las variedades (y las densidades que soportan) difieren, tanto el  $\alpha_i^*$  como el  $h_i$  \* «óptimos» para los estimadores de densidad individuales no tienen por qué coincidir.
3. Aunque las densidades individuales  $f_i$  estén bien estimadas, el clasificador resultante puede ser mal(ard)o si no diferencia bien «en las fronteras». Por simplicidad, además, decidimos parametrizar el clasificador con dos únicos hiperparámetros globales:  $\alpha, h$ .

### 3.5.3 Diseño experimental

1. Desarrollamos un clasificador compatible con el *framework* de [scikit-learn](#) según los lineamientos de Loubes & Pelleteir, que apodamos KDC.
2. Implementamos el estimador de la distancia muestral de Fermat, y combinándolo con KDC, obtenemos la titular «Clasificación por KDE con Distancia de Fermat», FKDC.
3. Evaluamos el *pseudo-R<sup>2</sup>* y la *exactitud* («accuracy») de los clasificadores propuestos en diferentes *datasets*, relativa a técnicas bien establecidas:
  - regresión logística (LR)
  - k-vecinos-más-cercanos (KN)
  - clasificador de soporte vectorial (SVC)<sup>5</sup>
  - Naive Bayes Gaussiano (GNB)
  - gradient boosting trees (GBT)

---

<sup>5</sup>sólo se consideró su exactitud, ya que no es un clasificador suave

- La implementación de `KNeighbors` de referencia acepta distancias pre-computadas, así que incluimos una versión con distancia de Fermat, que apodamos `F(ermat)KN`.
- Para ser «justos», se reservó una porción de los datos para la evaluación comparada, y del resto, cada algoritmo fue entrenado repetidas veces por validación cruzada de 5 pliegos, en una extensísima grilla de hiperparametrizaciones. Este procedimiento se repitió 25 veces en cada dataset.
- La función de score elegida fue `neg_log_loss` ( $= \ell$ ) para los clasificadores suaves, y `accuracy` para los duros.

- Para tener una idea «sistémica» de la performance de los algoritmos, evaluaremos su performance con *datasets* que varíen en el tamaño muestral  $N$ , la dimensión  $p$  de las  $X_i$ , el nro. de clases  $k$  y su origen («real» o «sintético»).
- Cuando creamos datos sintéticos en variedades con dimensión intrínseca menor a la ambiente, (casi) cualquier clasificador competente alcanza exactitud perfecta; para complejizar la tarea, agregamos un poco de «ruido» a las observaciones, y también analizamos sus efectos.

### 3.5.4 Regla de Parsimonia

- ¿Qué parametrización elegir cuando «en test da todo igual»?  
    └ de Occam: la más «sencilla» (TBD)
- ¿Qué parametrización elegir cuando «en test da casi todo igual»?  
    Regla de  $1\sigma$ : De las que estén a  $1\sigma$  de la mejor, la más sencilla.  
    ¿Sabemos cuánto vale  $\sigma$ ?

### 3.5.5 $R^2$ de McFadden

Sea  $\mathcal{C}_0$  el clasificador «base», que asigna a cada observación y posible clase, la frecuencia empírica de clase encontrada en la muestra  $\mathbf{X}$ . Para todo clasificador suave  $\mathcal{C}$ , definimos el  $R^2$  de McFadden como

$$R^2(\mathcal{C} | \mathbf{X}) = 1 - \frac{\ell(\mathcal{C})}{\ell(\mathcal{C}_0)} \quad (17)$$

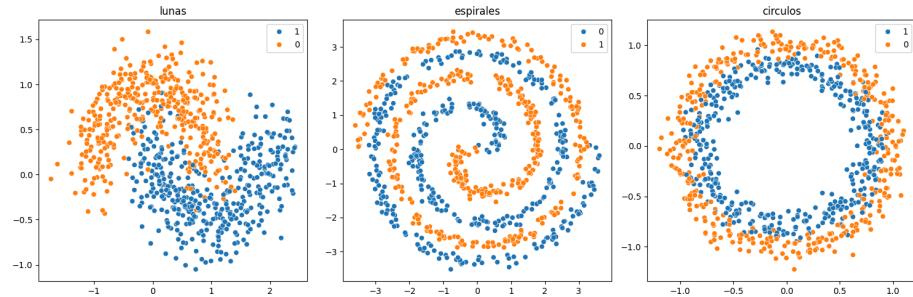
donde  $\ell(\cdot)$  es la log-verosimilitud clásica. Nótese que  $R^2(\mathcal{C}_0) = 0$ . A su vez, para un clasificador perfecto  $\mathcal{C}^*$  que otorgue toda la masa de probabilidad a la clase correcta, tendrá  $L(\mathcal{C}^*) = 1$  y log-verosimilitud igual a 0, de manera que  $R^2(\mathcal{C}^*) = 1 - 0 = 1$ .

Sin embargo, un clasificador *peor* que  $\mathcal{C}_0$  en tanto asigne bajas probabilidades ( $\approx 0$ ) a las clases correctas, puede tener un  $R^2$  infinitamente negativo.

## 4 Resultados

4.0.1 2D, 2 clases: excelente  $R^2$  con exactitud competitiva

4.0.2 Con Bajo Ruido



lunas\_lo (acc: 93.66%)

	delta_acc	r2
clf		
fkdc	-0.29	75.58
fkn	-0.04	74.43
kn	0.00	74.30
kdc	-0.33	73.42
gbt	-0.67	70.14
lr	-8.71	49.90
slr	-7.93	49.62
gnb	-10.12	48.83
base	-45.12	0.00
svc	-0.98	NaN

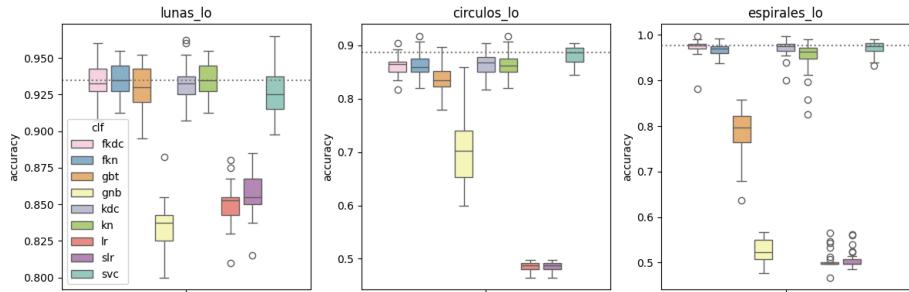
circulos\_lo (acc: 88.15%)

	delta_acc	r2
clf		
fkdc	-1.91	49.25
kdc	-1.71	48.58
fkn	-1.82	45.13
kn	-1.82	44.92
gbt	-4.64	43.42
gnb	-17.43	5.13
base	-39.61	0.00
lr	-39.61	-0.00
slr	-39.61	-0.00
svc	0.00	NaN

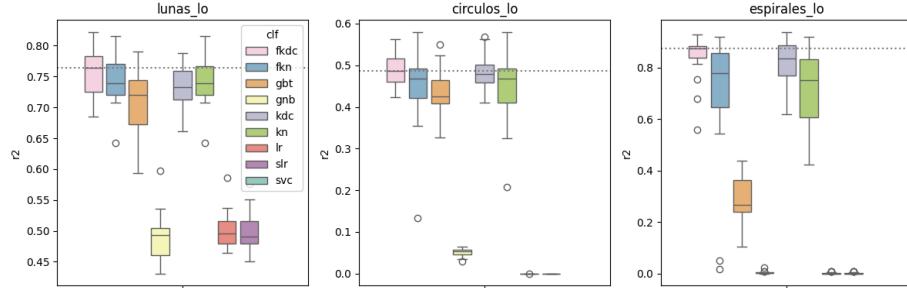
espirales\_lo (acc: 97.27%)

	delta_acc	r2
clf		
fkdc	0.00	84.66
kdc	-0.14	82.36
kn	-2.29	71.47
fkn	-0.50	70.64
gbt	-18.68	28.82
gnb	-44.59	0.35
lr	-46.74	0.17
slr	-46.56	0.17
base	-47.52	0.00
svc	-0.18	NaN

#### 4.0.3 Boxplot Accuracy



#### 4.0.4 Boxplot $R^2$



#### 4.0.5 Superposición de parámetros: $\alpha$ y $h$

- El uso de la distancia de Fermat muestral no hiere la performance, pero las mejoras son nulas o marginales. ¿Por qué?

Si recordamos  $\hat{f}_{K,N}$  según Loubes & Pelletier, al núcleo  $K$  se lo evalúa sobre

$$\frac{d(x_0, X_i)}{h}, \quad d = D_{Q_i, \alpha} \quad (18)$$

Lo que  $\alpha$  afecta a  $\hat{f}$  vía  $d$ , también se puede conseguir vía  $h$ .

Si  $D_{Q_i, \alpha} \propto \|\cdot\|$  (la distancia de fermat es proporcional a la euclídea), los efectos de  $\alpha$  y  $h$  se «solapan»

... y sabemos que localmente, eso es cierto 😊

#### 4.0.6 Parámetros óptimos para (F)KDC en `espirales_lo`

**bandwidth) (kdc,**

0.1000

0.1778

0.1778

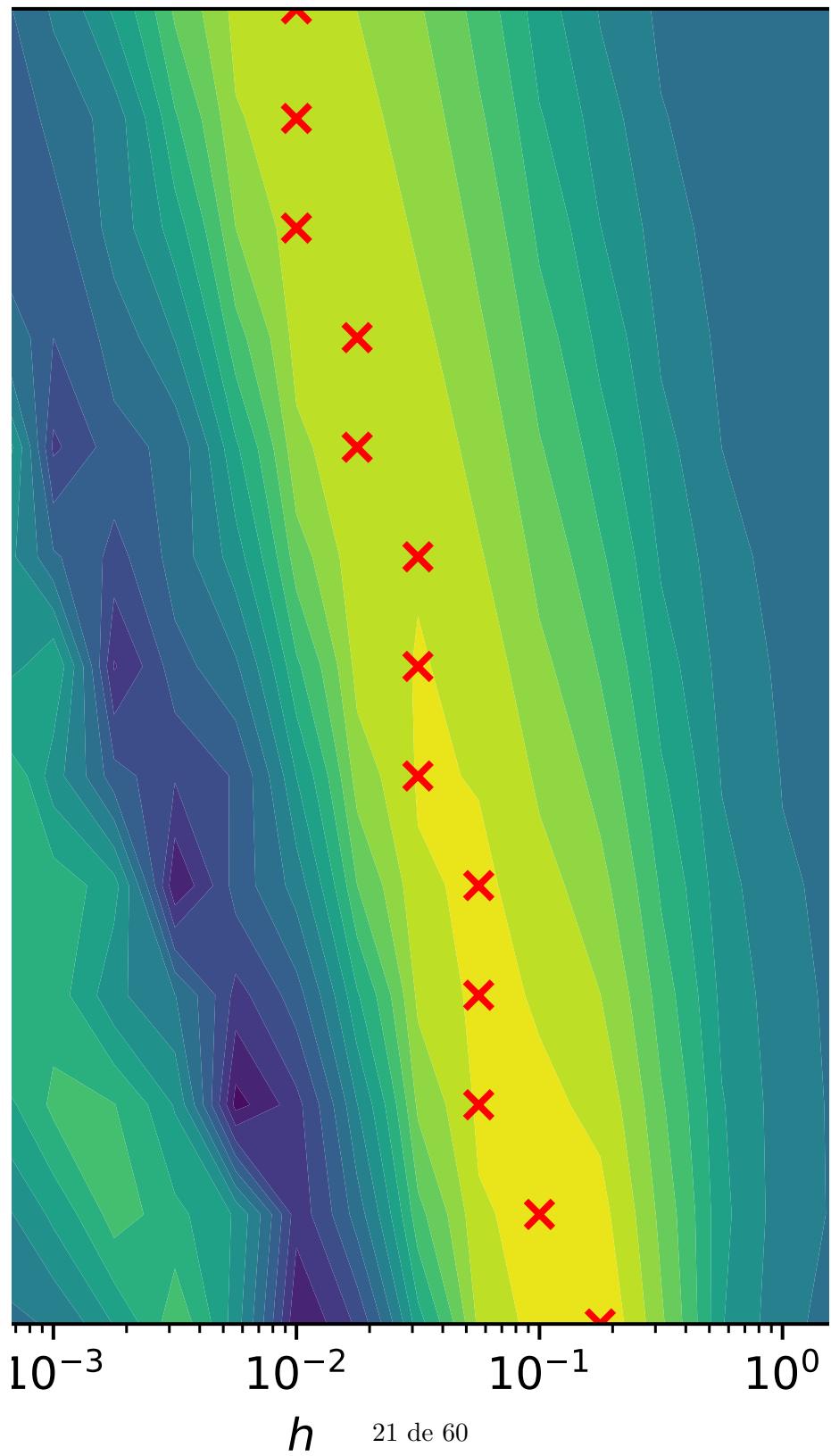
0.1778

0.3162

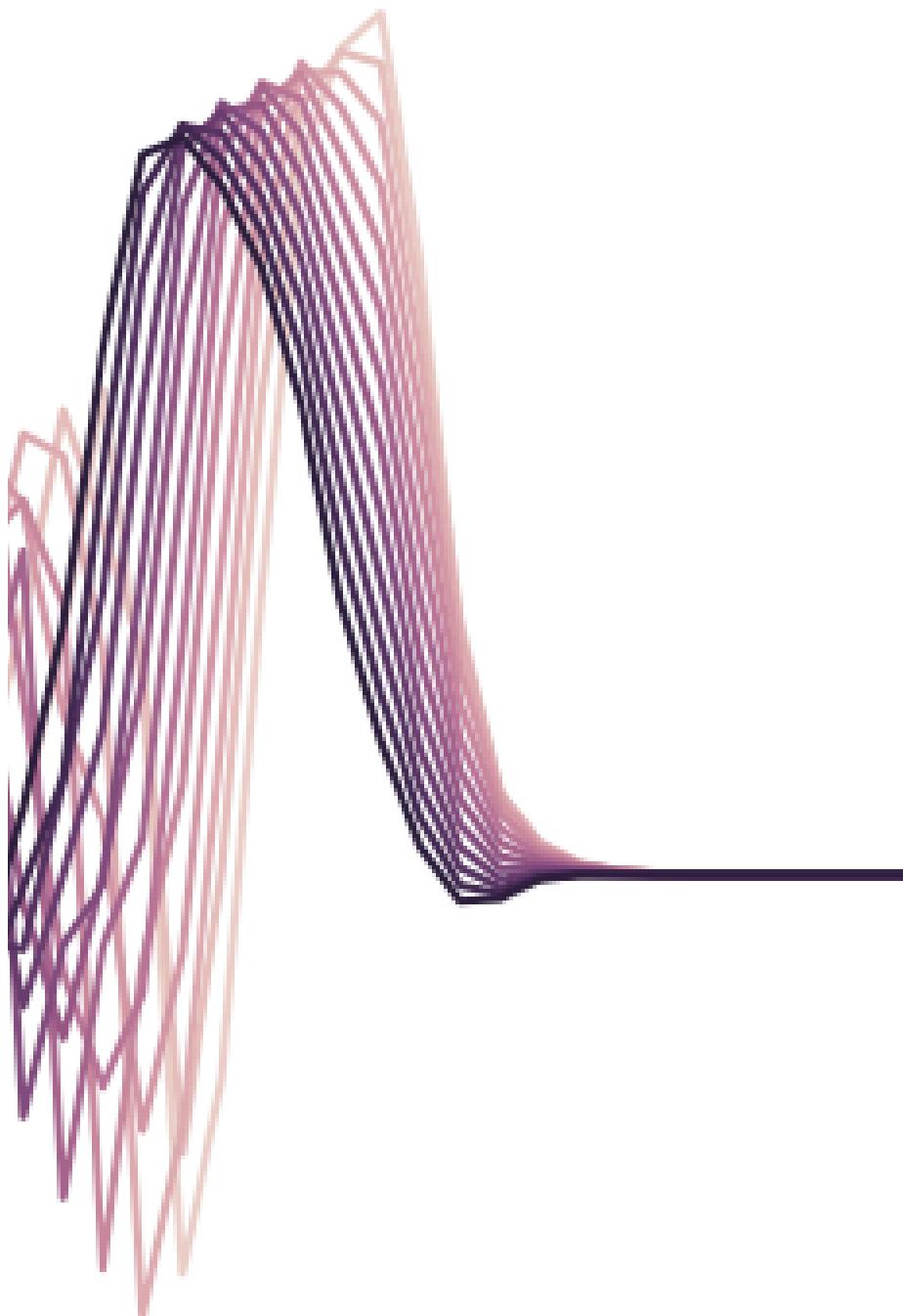
0.0032

#### 4.0.7 Superficies (o paisajes) de *score* para (espirales\_lo, 1434)

## Score para $\alpha$ y $h$



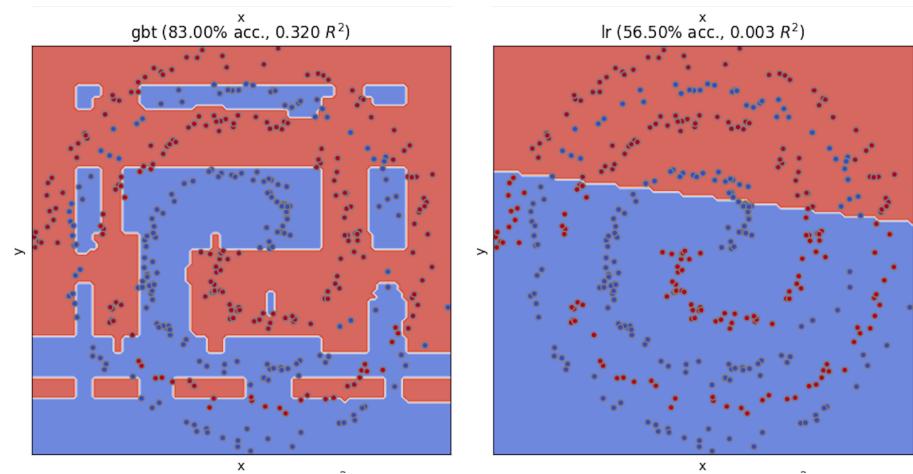
#### 4.0.8 Alt-viz: Perfiles de pérdida para (espirales\_lo, 1434)

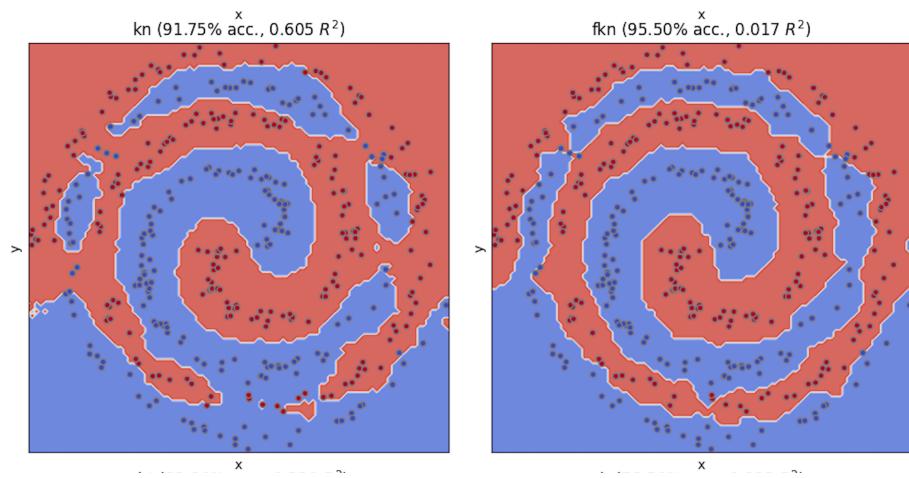


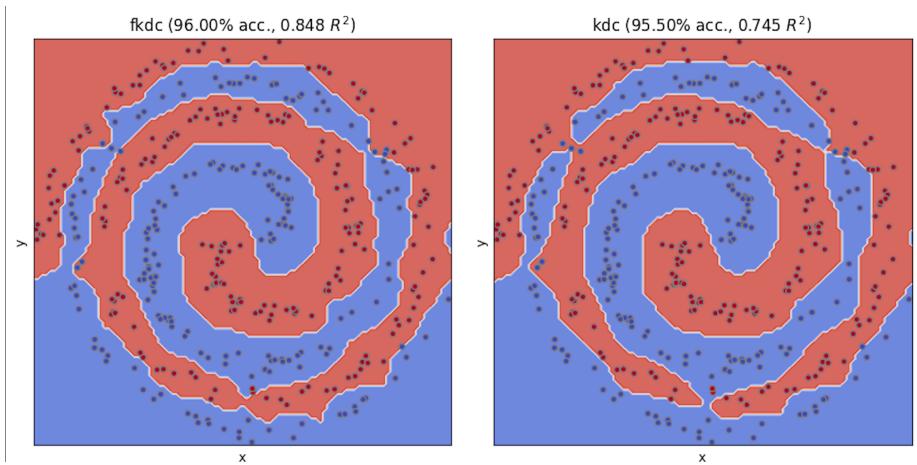
$10^{-2} \quad 10^0 \quad 10^2$

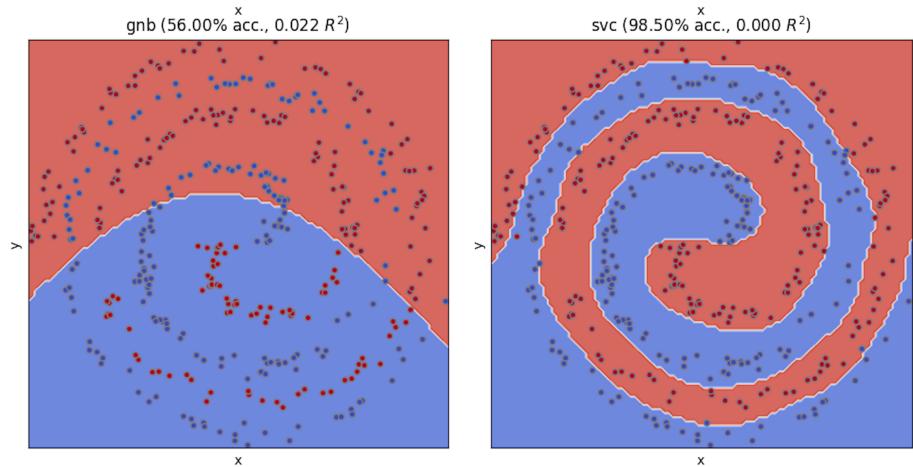
param\_bandwidth

#### 4.0.9 Fronteras de decisión para (espirales\_lo, 1434)

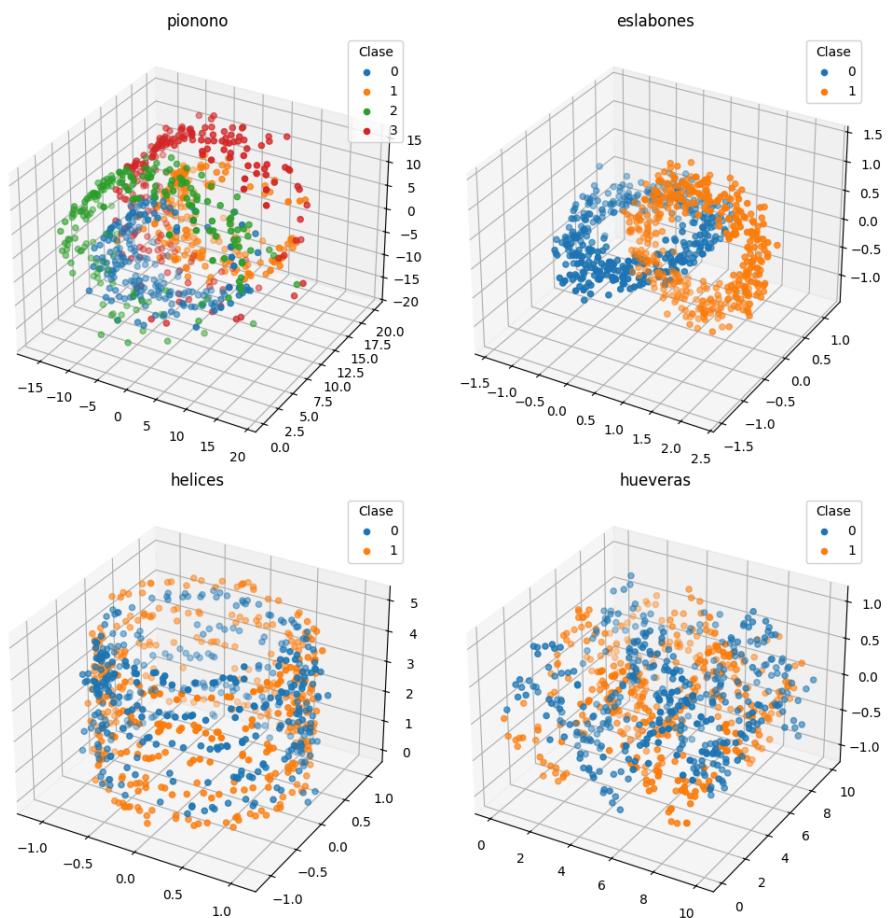


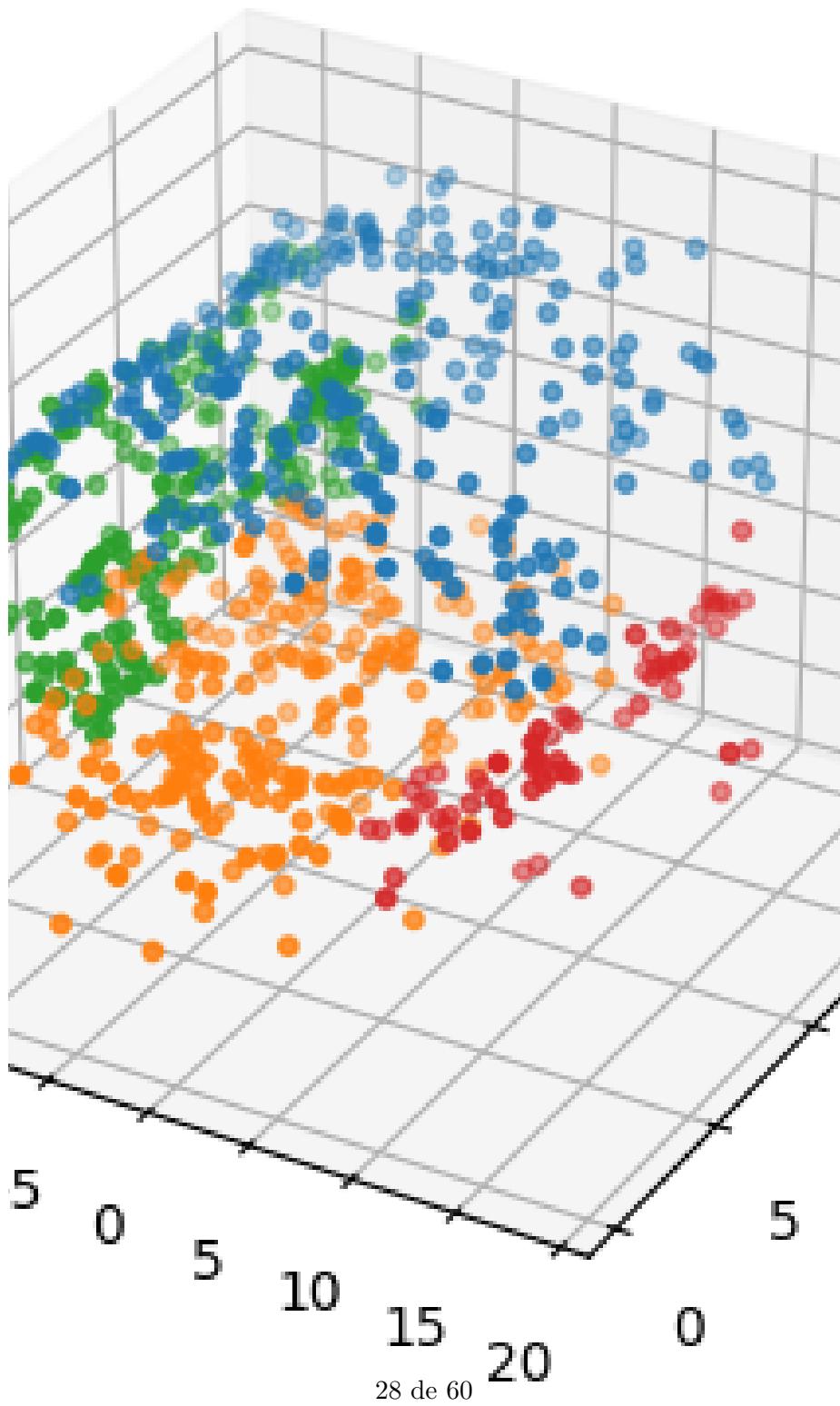






#### 4.0.10 3D, 2 clases + piononos





pionono\_0 (acc: 94.19%)

	clf	delta_acc	r2
fkdc	-1.06	81.04	
gbt	-2.37	81.02	
kdc	-0.83	79.89	
fkn	-2.65	72.55	
kn	-3.57	70.62	
gnb	-20.40	58.65	
sir	-29.69	47.45	
lr	-29.72	47.44	
base	-71.44	0.00	
svc	0.00	NaN	

eslabones\_0 (acc: 99.9%)

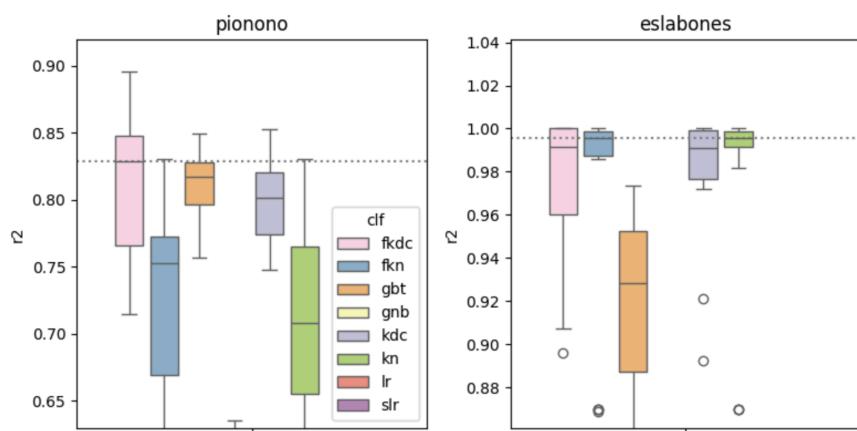
	clf	delta_acc	r2
fkdc	-0.10	97.75	
kn	-0.02	96.51	
fkdc	-0.06	96.44	
fkn	-0.02	96.04	
gbt	-1.14	91.97	
gnb	-10.68	75.16	
lr	-33.32	22.04	
sir	-33.30	21.87	
base	-50.15	0.00	
svc	0.00	NaN	

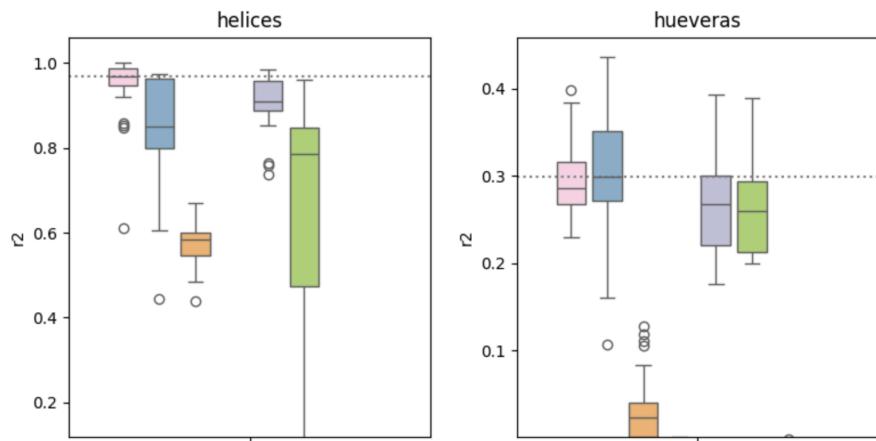
helices\_0 (acc: 99.16%)

	clf	delta_acc	r2
fkdc	0.00	94.41	
kdc	-0.06	86.10	
fkn	-0.76	83.45	
kn	-4.90	62.81	
gbt	-9.48	57.56	
gnb	-49.29	0.01	
base	-49.41	0.00	
lr	-49.41	0.00	
sir	-49.41	0.00	
svc	-24.03	NaN	

hueveras\_0 (acc: 77.88%)

	clf	delta_acc	r2
fkn	-1.60	30.20	
fkdc	-2.57	29.40	
kdc	-3.36	26.62	
kn	-2.71	26.43	
gbt	-19.90	3.44	
gnb	-27.43	0.01	
base	-28.13	0.00	
sir	-28.13	0.00	
lr	-28.15	-0.01	
svc	0.00	NaN	





#### **4.0.11 Parámetros óptimos para (F)KDC en helices\_0**

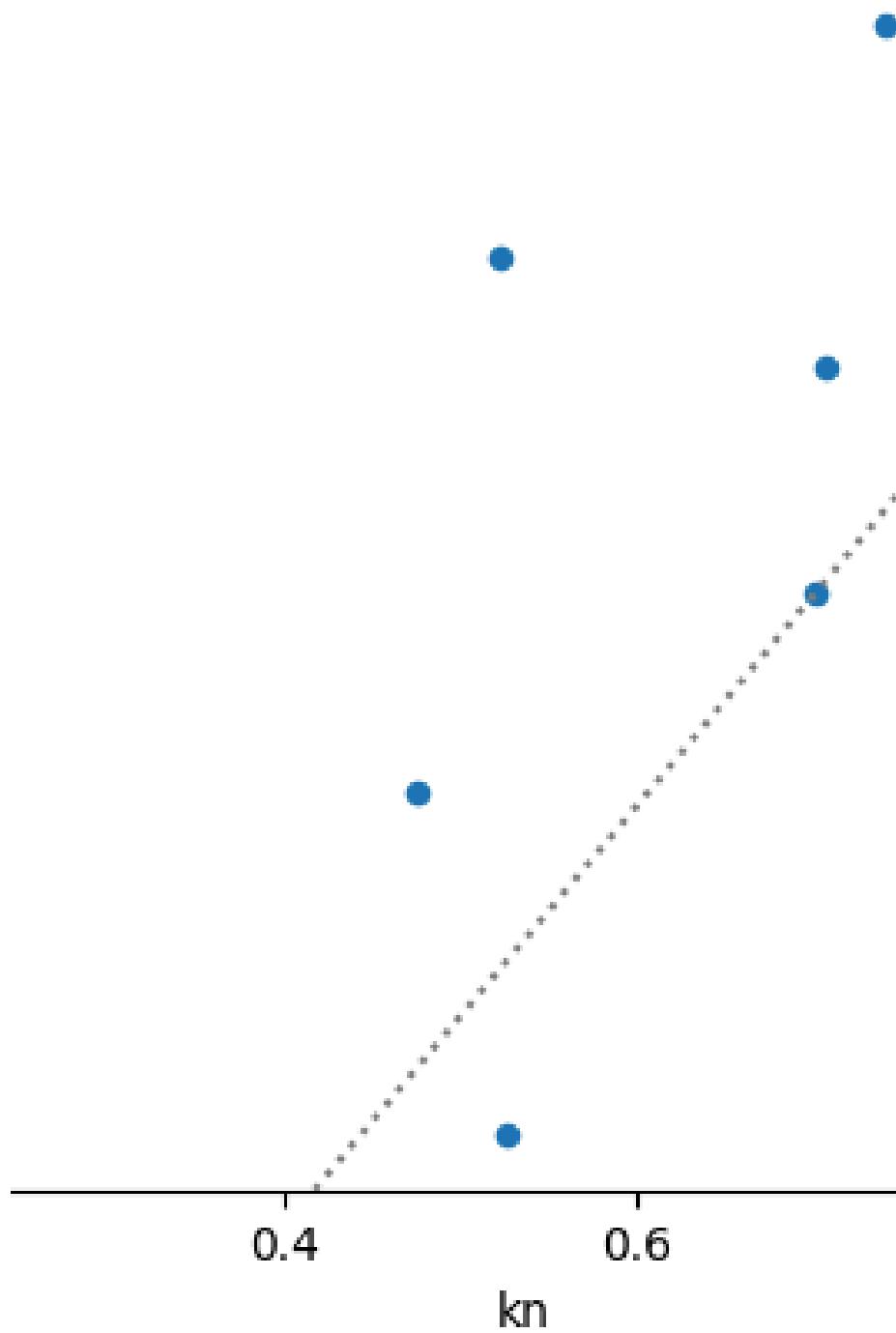
<b>(fkdc, bandwidth)</b>	<b>(kdc, bandwidth)</b>
0.1000	C
0.0100	C
0.0056	0
0.0100	0
0.1000	C
0.0056	C
0.0056	0
0.0100	C
0.0056	0
0.0032	C
0.0032	0
0.0032	C
0.0032	0
0.0010	0
0.0018	0

#### 4.0.12 Microindiferencia, macrodiferencia

- En zonas con muchas observaciones (por tener alta  $f$  o alto  $N$ ) sampleadas, la distancia de Fermat y la euclídea coinciden.
- «Localmente», siempre van a coincidir, aunque sea en un vecindario muy pequeño.
- Si el algoritmo de clasificación sólo depende de ese vecindario local para clasificar, no hay ganancia en la distancia de Fermat.
- ¡Pero tampoco hay pérdida si se elige mal `n_neighbors`! 🦸

#### **4.0.13 $R^2$ por semilla para (F)KN en helices\_0**

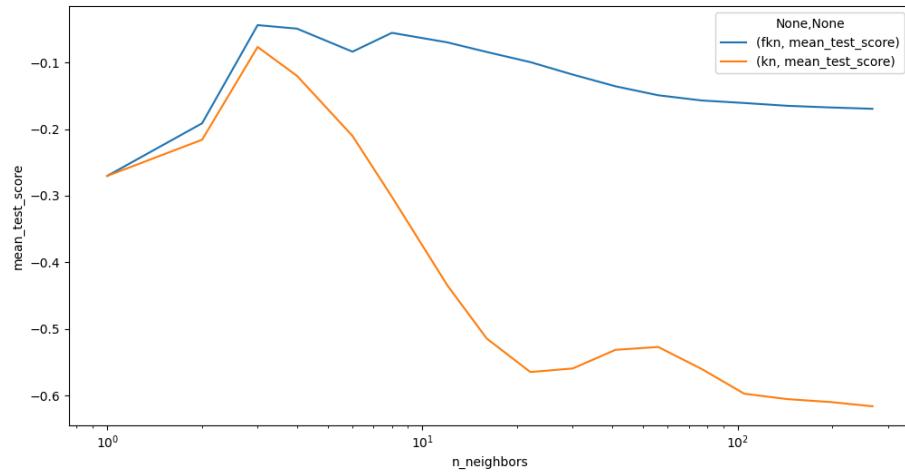
por semilla para FKN y KN en 'h'



4.0.14  $R^2$  y  $\alpha^*$  para (F)KN en helices\_0, n\_neighbors seleccionados

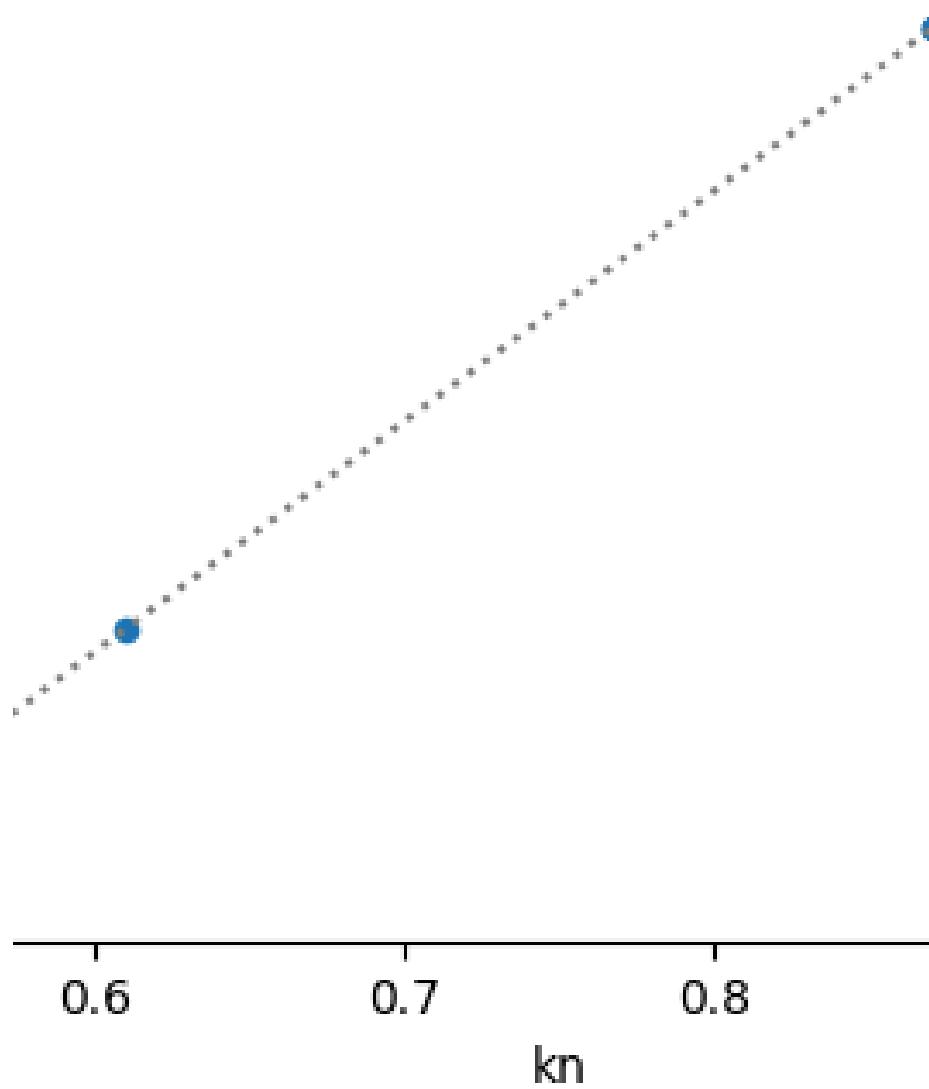
n	ram_alpha	mean_test_s
3.50		-0.27
1.75		-0.04
4.00		-0.06
4.00		-0.13
4.00		-0.16

#### 4.0.15 Mejor $R^2$ para (F)KN en helices\_0, en función de n\_neighbors



#### **4.0.16 $R^2$ por semilla para (F)KN en eslabones\_0**

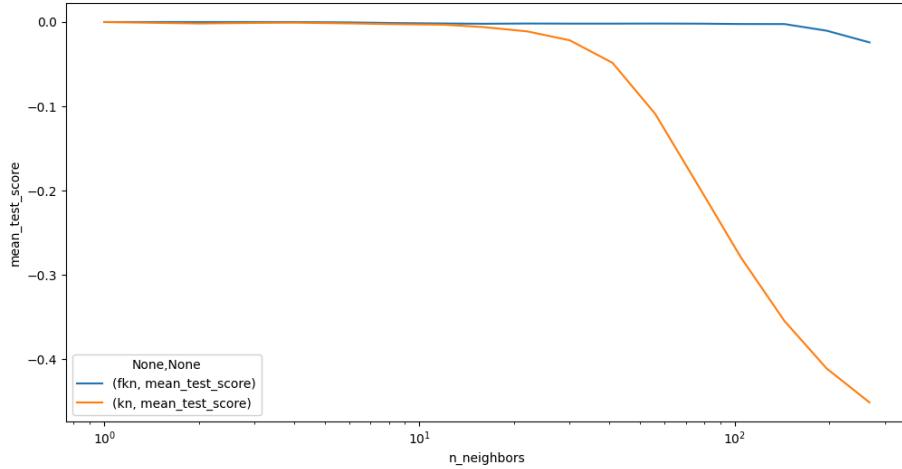
or semilla para FKN y KN en `es|



4.0.17  $R^2$  y  $\alpha^*$  para (F)KN en `eslabones_0`, `n_neighbors` seleccionados

<code>n_neighbors</code>	<code>ram_alpha</code>	<code>mean_test_r2</code>	<code>std_test_r2</code>
1	1.00	-0.000	0.000
2	3.75	-0.000	0.000
3	3.75	-0.000	0.000
4	3.00	-0.000	0.000
5	4.00	-0.000	0.000

#### 4.0.18 Mejor $R^2$ para (F)KN en eslabones\_0, en función de n\_neighbors



#### 4.0.19 Otros datasets: 2D mucho ruido

lunas\_hi (acc: 81.44%)

	clf	delta_acc	r2
gbt	gbt	-0.21	38.14
fkdc	fkdc	-0.33	37.46
fkn	fkn	0.00	36.57
kn	kn	-0.02	36.50
kdc	kdc	-0.47	35.67
slr	slr	-1.21	33.75
lr	lr	-1.73	33.70
gnb	gnb	-1.98	33.34
base	base	-32.90	0.00
svc	svc	-1.18	NaN

circulos\_hi (acc: 65.93%)

	clf	delta_acc	r2
gbt	gbt	-0.23	9.49
fkdc	fkdc	-3.82	6.98
kdc	kdc	-4.78	5.54
kn	kn	-4.87	5.14
fkn	fkn	-5.41	5.12
gnb	gnb	-5.34	3.33
base	base	-17.39	0.00
lr	lr	-17.39	-0.00
slr	slr	-17.39	-0.00
svc	svc	0.00	NaN

espirales\_hi (acc: 85.54%)

	clf	delta_acc	r2
fkdc	fkdc	-1.86	44.41
kdc	kdc	-1.98	43.45
fkn	fkn	-2.57	40.36
kn	kn	-3.06	39.93
gbt	gbt	-14.68	15.22
gnb	gnb	-32.64	0.33
lr	lr	-34.95	0.17
slr	slr	-34.97	0.17
base	base	-35.79	0.00
svc	svc	0.00	NaN

#### 4.0.20 Otros datasets: 15D

pionono\_12 (acc: 91.07%)

	clf	delta_acc	r2
gbt	gbt	0.00	78.96
gnb	gnb	-20.78	54.30
slr	slr	-28.42	42.42
lr	lr	-28.56	42.22
fkdc	fkdc	-46.30	10.18
kdc	kdc	-41.83	10.05
fkn	fkn	-44.47	9.99
kn	kn	-44.15	9.95
base	base	-68.32	0.00
svc	svc	-29.07	NaN

eslabones\_12 (acc: 98.48%)

	clf	delta_acc	r2
gbt	gbt	0.00	91.66
gnb	gnb	-8.67	75.24
slr	slr	-22.66	24.58
lr	lr	-21.98	24.43
fkdc	fkdc	-22.40	24.32
kdc	kdc	-22.52	24.11
kn	kn	-31.16	21.01
fkn	fkn	-30.98	20.69
base	base	-48.73	0.00
svc	svc	-19.20	NaN

helices\_12 (acc: 53.15%)

	clf	delta_acc	r2
kn	kn	-0.99	0.20
fkn	fkn	-0.99	0.20
gnb	gnb	-1.01	0.13
base	base	-3.40	0.00
lr	lr	-3.40	0.00
slr	slr	-3.40	0.00
gbt	gbt	0.00	-0.37
fkdc	fkdc	-3.41	-2.69
kdc	kdc	-3.45	-6.01
kn	kn	-2.92	NaN

hueveras\_12 (acc: 53.55%)

	clf	delta_acc	r2
kn	kn	-0.87	0.20
fkn	fkn	-0.99	0.17
gnb	gnb	-1.30	0.14
lr	lr	-3.81	0.02
base	base	-3.80	0.00
slr	slr	-3.80	0.00
gbt	gbt	0.00	-0.62
fkdc	fkdc	-3.74	-10.75
kdc	kdc	-3.97	-15.08
kn	kn	-3.69	NaN

#### 4.0.21 Otros datasets: multiclase

iris (acc: 96.27%)

	clf	delta_acc	r2
	lr	0.00	88.64
	sir	-0.53	87.99
	gbt	-1.49	85.88
	gnb	-4.05	80.93
	fkdc	-0.96	79.30
	kdc	-1.17	78.02
	kn	-0.48	74.39
	fkn	-0.64	73.21
	base	-65.87	0.00
	svc	-1.65	NaN

vino (acc: 97.3%)

	clf	delta_acc	r2
	gbt	-0.85	89.80
	sir	0.00	89.72
	gnb	-2.70	84.59
	lr	-10.92	67.06
	fkn	-27.55	45.22
	kn	-27.55	45.22
	fkdc	-30.56	42.44
	kdc	-30.92	39.42
	base	-58.83	0.00
	svc	-8.94	NaN

pinguinos (acc: 99.13%)

	clf	delta_acc	r2
	sir	0.00	96.22
	lr	-0.77	95.51
	gbt	-1.80	91.85
	gnb	-4.68	84.81
	fkn	-22.67	49.42
	kn	-22.67	49.36
	fkdc	-26.55	41.60
	kdc	-26.50	40.49
	base	-56.16	0.00
	svc	-3.67	NaN

anteojos (acc: 97.68%)

	clf	delta_acc	r2
	fkdc	-0.08	92.89
	kdc	0.00	91.84
	kn	-0.16	89.32
	fkn	-0.17	87.08
	gbt	-1.36	86.12
	gnb	-0.62	85.17
	lr	-55.51	27.80
	sir	-56.00	27.60
	base	-48.34	0.00
	svc	-0.26	NaN

#### 4.0.22 Otros datasets: digitos y mnist

digitos (acc: 98.41%)

	clf	delta_acc	r2
	fkdc	0.00	97.67
	kdc	-0.10	96.80
	gbt	-2.43	94.17
	lr	-2.24	93.38
	sir	-2.33	93.10
	fkn	-1.98	92.22
	kn	-2.91	90.62
	gnb	-8.42	85.67
	base	-89.43	0.00
	svc	-0.16	NaN

mnist (acc: 87.07%)

	clf	delta_acc	r2
	kdc	-4.04	76.38
	lr	-4.32	76.10
	fkdc	-4.38	73.38
	gbt	-10.11	66.57
	sir	-12.33	61.43
	gnb	-19.10	56.91
	fkn	-16.28	54.02
	kn	-19.10	50.40
	base	-76.57	0.00
	svc	0.00	NaN

#### 4.0.23 El problema de clasificación

Definición 4.0.23.1 (problema de clasificación):

Definición 4.0.23.2 (clasificador vecino más cercano):

#### 4.0.24 Definición del problema unidimensional

[muestra aleatoria] Sea

Consideremos el problema de clasificación: [problema de clasificación] Sea  $X = \{X_{\{1\}}, \dots, X_{\{N\}}\}, X_{\{i\}} \in R^{\{\text{dim}_x\}}$   $\forall i \in [N]$  una muestra de  $N$  observaciones aleatorias  $d_x$ -dimensionales, repartidas en  $M$  clases  $C_{\{1\}}, \dots, C_{\{M\}}$  mutuamente excluyentes y conjuntamente exhaustivas<sup>footnote{es decir,  $\forall i \in [N] \equiv \{1, \dots, N\}, X_{\{i\}} \in C_{\{j\}} \Leftrightarrow X_{\{i\}} \not\in C_{\{k\}}, k \in \mu, k \neq j$ }</sup>. Asumamos además que la muestra está compuesta de observaciones independientes entre sí, y las observaciones de cada clase están idénticamente distribuidas según su propia ley: si  $|C_{\{j\}}| = N_{\{j\}}$  y  $X_{\{i\}}^{(j)}$  representa la  $i$ -ésima observación de la clase  $j$ , resulta que  $X_{\{i\}}^{(j)} \in L_{\{j\}}(X) \forall j \in \mu, i \in [N_{\{j\}}]$ .

Dada una nueva observación  $x_{\{0\}}$  cuya

#### 4.0.25 Clasificadores «duros» y «suaves»

Definición 4.0.25.1 (clasificación dura): ¿a qué clase deberíamos asignarla?

Definición 4.0.25.2 (clasificación suave): ¿qué probabilidad tiene de pertenecer a cada clase  $C_{\{j\}}, j \in [M]$ ?

Cualquier método o algoritmo que pretenda responder el problema de clasificación, prescribe un modo u otro de combinar toda la información muestral disponible, ponderando las  $N$  observaciones relativamente a su cercanía o similitud con  $x_{\{0\}}$ . Por caso,  $k$ -vecinos más cercanos ( $k$ -NN) asignará la nueva observación  $x_{\{0\}}$  a la clase modal - la más frecuente - entre las  $k$  observaciones de entrenamiento más cercanas<sup>emph{ }en distancia euclídea  $\text{norm}\{x_{\{0\}} - \cdot\}$ .</sup>  $k$ -NN no menciona explícitamente las leyes de clase L}, lo cual lo mantiene sencillo a costa de ignorar la estructura del problema.

Definición 4.0.25.3 (k-NN): k-Nearest Neighbors

#### 4.0.26 Clasificador de Bayes

#### 4.0.27 KDE: Estimación de la densidad por núcleos

Definición 4.0.27.1 (KDE):

#### 4.0.28 La maldición de la (alta) dimensionalidad

#### 4.0.29 NB: El clasificador «ingenuo» de Bayes

Definición 4.0.29.1 (Naïve Bayes):

##### 4.0.29.1 Una muestra adversa

#### 4.0.30 KDC Multivariado

Definición 4.0.30.1 (KDE Multivariado):

Definición 4.0.30.2 (KDC):

##### 4.0.30.1 El caso 2-dimensional

##### 4.0.30.2 Relación entre $H$ y la distancia de Mahalanobis

#### 4.0.31 La hipótesis de la variedad

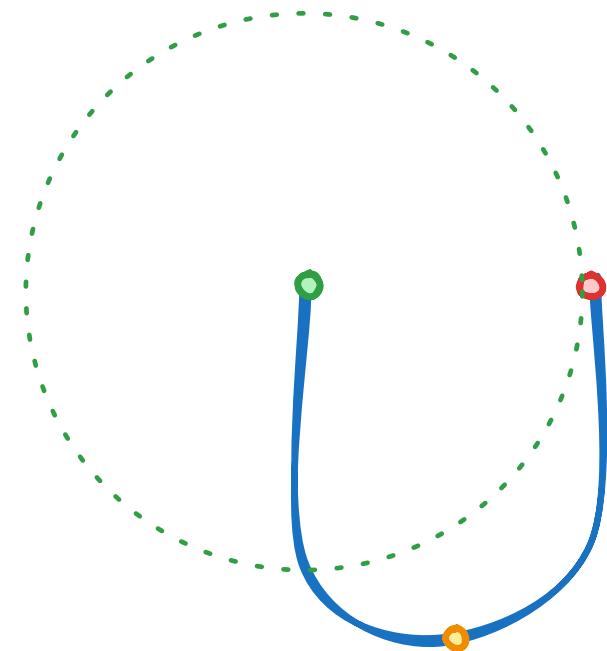


Figura 1: La variedad  $\mathcal{U}$  con  $\dim(\mathcal{U}) = 1$  embebida en  $\mathbb{R}^2$ . Nótese que en el espacio ambiente, el punto rojo está más cerca del verde, mientras que a través de  $\mathcal{U}$ , el punto amarillo está más próximo que el rojo

#### 4.0.32 KDE en variedades de Riemann

#### 4.0.33 Variedades desconocidas

#### 4.0.34 Aprendizaje de distancias

#### 4.0.35 Isomap

#### 4.0.36 Distancias basadas en densidad

#### 4.0.37 Distancia de Fermat

- Groisman & Jonckheere (Groisman, Jonckheere y Sapienza, 2019)
- Little & Mackenzie (Little, McKenzie y Murphy, 2021)
- Bijral (Bijral, Ratliff y Srebro, 2012)
- Vincent & Bengio (Vincent y Bengio, 2003)

Definición 4.0.37.1 (Distancia Muestral de Fermat):

### 4.1 Propuesta Original

Habiendo andado este sendero teórico, la pregunta natural que asoma es: ¿es posible mejorar un algoritmo de clasificación reemplazando la distancia euclídea por una aprendida de los datos, como la de Fermat? Para investigar la cuestión, nos propusimos:

1. Implementar un clasificador basado en estimación de densidad por núcleos ([Definición 4.0.27.1](#)) según (Loubes y Pelletier, 2008), que llamaremos «KDC». Además,
2. Implementar un estimador de densidad por núcleos basado en la distancia de Fermat, a fines de poder comparar la *performance* de KDC con distancia euclídea y de Fermat.

Nótese que el clasificador enunciado al inicio (k-NN, [Definición 4.0.25.3](#)), tiene un pariente cercano,  $\varepsilon - \text{NN}$

Definición 4.1.1 ( $\varepsilon - \text{NN}$ ):

[Definición 4.1.1](#) es esencialmente equivalente a KDC con un núcleo «rectangular»,  $k(t) = \frac{\mathbb{1}(d(x,t) < \varepsilon)}{\varepsilon}$ , pero su definición es considerablemente más sencilla. Luego, propondremos también

3. Implementar un clasificador cual [Definición 4.0.25.3](#), pero con distancia muestral de Fermat en lugar de euclídea.

#### 4.1.1 KDC con Distancia de Fermat Muestral

#### 4.1.2 f-KNN

## 4.2 Evaluación

Nos interesa conocer en qué circunstancias, si es que hay alguna, la distancia muestral de Fermat provee ventajas a la hora de clasificar por sobre la distancia euclídea. Además, en caso de existir, quisiéramos en la medida de lo posible comprender por qué (o por qué no) es que tal ventaja existe. A nuestro entender resulta imposible hacer declaraciones demasiado generales al respecto de la capacidad del clasificador: la cantidad de *datasets* posibles, junto con sus *configuraciones de evaluación* es tan densamente infinita como lo permita la imaginación del evaluador. Con un ánimo exploratorio, nos proponemos explorar la *performance* de nuestros clasificadores basados en distancia muestral de Fermat en algunas *tareas* puntuales.

### 4.2.1 Métricas de *performance*

En tareas de clasificación, la métrica más habitual es la *exactitud*<sup>6</sup>

Definición 4.2.1.1 (exactitud): Sean  $(\mathbf{X}, \mathbf{y}) \in \mathbb{R}^{n \times p} \times \mathbb{R}^n$  una matriz de  $n$  observaciones de  $p$  atributos y sus clases asociadas. Sea además  $\hat{\mathbf{y}} = R(\mathbf{X})$  las predicciones de clase resultado de una regla de clasificación  $R$ . La *exactitud* (exac) de  $R$  en  $\mathbf{X}$  se define como la proporción de coincidencias con las clases verdaderas  $\mathbf{y}$ :

$$\text{exac}(R | \mathbf{X}) = n^{-1} \sum_{i=1}^n \mathbb{1}(\hat{y}_i = y_i) \quad (19)$$

La exactitud está bien definida para cualquier clasificador que provea una regla *dura* de clasificación, según [Definición 4.0.25.1](#). Ahora bien, cuando un clasificador provee una regla *suave* ([Definición 4.0.25.2](#)), la exactitud como métrica « pierde información »: dos clasificadores binarios que asignen respectivamente 0.51 y 1.0 de probabilidad de pertenecer a la clase correcta a todas las observaciones tendrán la misma exactitud, 100%, aunque el segundo es a las claras mejor. A la inversa, cuando un clasificador erra al asignar la clase: ¿lo hace con absoluta confianza, asignando una alta probabilidad a la clase equivocada, o con cierta incertidumbre, repartiendo la masa de probabilidad entre varias clases que considera factibles?

Una métrica natural para evaluar una regla de clasificación suave, es la *verosimilitud* (y su logaritmo) de las predicciones.

---

<sup>6</sup>Más conocida por su nombre en inglés, *accuracy*.

Definición 4.2.1.2 (verosimilitud): Sean  $(\mathbf{X}, \mathbf{y}) \in \mathbb{R}^{n \times p} \times \mathbb{R}^n$  una matriz de  $n$  observaciones de  $p$  atributos y sus clases asociadas. Sea además  $\hat{\mathbf{Y}} = \mathcal{R}(\mathbf{X}) \in \mathbb{R}^{n \times k}$  la matriz de probabilidades de clase resultado de una regla suave de clasificación  $\mathcal{R}$ . La *verosimilitud* (vero) de  $\mathcal{R}$  en  $\mathbf{X}$  se define como la probabilidad conjunta que asigna  $\mathcal{R}$  a las clases verdaderas  $\mathbf{y}$ :

$$L(\mathcal{R}) = \text{vero}(\mathcal{R} | \mathbf{X}) = \Pr(\hat{\mathbf{y}} = \mathbf{y}) = \prod_{i=1}^n \Pr(\hat{y}_i = y_i) = \prod_{i=1}^n \hat{\mathbf{Y}}_{(i,y_i)}^{(20)}$$

Por conveniencia, se suele considerar la *log-verosimilitud promedio*,

$$\ell(\mathcal{R}) = n^{-1} \log(L(\mathcal{R})) = n^{-1} \sum_{i=1}^n \log(\hat{\mathbf{Y}}_{(i,y_i)}) \quad (21)$$

Definición 4.2.1.3 ( $R^2$  de McFadden): Sea  $\mathcal{R}_0$  el clasificador «nulo», que asigna a cada observación y posible clase, la frecuencia empírica de clase encontrada en la muestra de entrenamiento  $\mathbf{X}_{\text{train}}$ . Para todo clasificador suave  $\mathcal{R}$ , definimos el  $R^2$  de McFadden como

$$R^2(\mathcal{R} | \mathbf{X}) = 1 - \frac{\ell(\mathcal{R})}{\ell(\mathcal{R}_0)} \quad (22)$$

*Observación:*  $R^2(\mathcal{R}_0) = 0$ . A su vez, para un clasificador perfecto  $\mathcal{R}^*$  que otorgue toda la masa de probabilidad a la clase correcta, tendrá  $L(\mathcal{R}^*) = 1$  y log-verosimilitud igual a 0, de manera que  $R^2(\mathcal{R}^*) = 1 - 0 = 1$ .

Sin embargo, un clasificador *peor* que  $\mathcal{R}_0$  en tanto asigne bajas probabilidades a las clases correctas, puede tener un  $R^2$  infinitamente negativo.

Visto y considerando que tanto  $\mathcal{F}$ -KDC como  $\mathcal{F}$ -kNN son clasificadores suaves, evaluaremos su comportamiento en comparación con ambas métricas, la exactitud y el  $R^2$  de McFadden<sup>7</sup>

#### 4.2.2 Algoritmos de referencia

Además de medir qué (des)ventajas otorga el uso de una distancia aprendida de los datos en la tarea de clasificación, quisiéramos entender (a) por qué sucede, y (b) si tal (des)ventaja es significativa en el amplio abanico de algoritmos disponibles. Pírrica victoria sería mejorar con la distancia de Fermat la *performance* de cierto algoritmo, para encontrar que aún con la mejora, el algoritmo no es competitivo en la tarea de referencia.

Consideraremos a modo de referencia los siguientes algoritmos:

---

<sup>7</sup>de aquí en más,  $R^2$  para abbreviar

- Naive Bayes Gaussiano ([Definición 4.0.29.1](#), GNB),
- Regresión Logistica (LR) y
- Clasificador de Soporte Vectorial (svc)

Esta elección no pretende ser exhaustiva, sino que responde a un «capricho informado» del investigador. GNB es una elección natural, ya que es la simplificación que surge de asumir independencia en las dimensiones de  $\{\mathbf{X}\}$  para KDE multivariado ([Definición 4.0.30.1](#)), y se puede computar para grandes conjuntos de datos en muy poco tiempo. LR es «el» método para clasificación binaria, y su extensión a múltiples clases no es particularmente compleja: para que sea mínimamente valioso un nuevo algoritmo, necesita ser al menos tan bueno como LR, que tiene ya más de 65 años en el campo (TODO REF bliss1935, cox1958). Por último, fue nuestro deseo incorporar algún método más cercano al estado del arte. A tal fin, consideramos incorporar alguna red neuronal (TODO REF), un método de *boosting* (TODO REF) y el antedicho clasificador de soporte vectorial, svc. Finalmente, por la sencillez de su implementación dentro del marco elegido<sup>8</sup> y por la calidad de los resultados obtenidos, decidimos incorporar svc, en dos variantes: con núcleos (*kernels*) lineales y RBF.

#### 4.2.3 Uno complejo: SVC

Definición 4.2.3.1 (clasificador por soporte vectorial):

#### 4.2.4 Uno conocido: LR - tal vez?

Definición 4.2.4.1 (regresión logística multinomial):

#### 4.2.5 Metodología

La unidad de evaluación de los algoritmos a considerar es una Tarea, que se compone de:

- un *diccionario de algoritmos* a evaluar en condiciones idénticas, definidas por
- un *dataset* con el conjunto de  $n$  observaciones en  $d_x$  dimensiones repartidas en  $k$  clases,  $\{\mathbf{X}\}_n \in R^{n \times d_x}, \{\mathbf{y}\}_n \in [k]^n$ ,
- un *split de evaluación*  $r \in (0, 1)$ , que determina las proporciones de los datos a usar durante el entrenamiento ( $1 - r$ ) y la evaluación ( $r$ ), junto con
- una *semilla*  $s \in [2^{32}]$  que alimenta el generador de números aleatorios y define determinísticamente cómo realizar la división antedicha.

---

<sup>8</sup>Utilizamos *scikit-learn*, un poderoso y extensible paquete para tareas de aprendizaje automático en Python

#### 4.2.6 Entrenamiento de los algoritmos

La especificación completa de un clasificador, requiere, además de la elección del algoritmo, la especificación de sus *hiperparámetros*, de manera tal de optimizar su rendimiento bajo ciertas condiciones de evaluación. Para ello, se definió de antemano para cada clasificador una *grilla* de hiperparámetros: durante el proceso de entrenamiento, la elección de los «mejores» hiperparámetros se efectuó maximizando la exactitud ([Definición 4.2.1.1](#)) con una búsqueda exhaustiva por convalidación cruzada de 5 pliegos<sup>9</sup> sobre la grilla entera.

#### 4.2.7 Estimación de la variabilidad en la *performance* reportada

En última instancia, cualquier métrica evaluada, no es otra cosa que un *estadístico* que representa la «calidad» del clasificador en la Tarea a mano. A fines de conocer no sólo su estimación puntual sino también darnos una idea de la variabilidad de su performance, para cada dataset y colección de algoritmos, se entrenaron y evaluaron 16 Tareas idénticas salvo por la semilla  $s$ , que luego se usaron para estimar la varianza y el desvío estándar en la exactitud ([Definición 4.2.1.1](#)) y el pseudo- $R^2$  ([Definición 4.2.1.3](#)).

Cuando el conjunto de datos proviene del mundo real y por lo tanto *preexiste a nuestro trabajo*, las 16 semillas  $s_1, \dots, s_{16}$  fueron utilizadas para definir el split de entrenamiento/evaluación. Por el contrario, cuando el conjunto de datos fue generado sintéticamente, las semillas se utilizaron para generar 16 versiones distintas pero perfectamente replicables del dataset, y en todas se utilizó una misma semilla maestra  $s^*$  para definir el split de evaluación.

#### 4.2.8 Resultados

##### 4.2.9 Chequeo de sanidad: `blobs`

Antes de considerar ningún tipo de sofisticación, comenzamos asegurándonos que en condiciones benignas, nuestros clasificadores funcionan correctamente. La

---

<sup>9</sup>Conocida en inglés como *Grid Search 5-fold Cross-Validation*

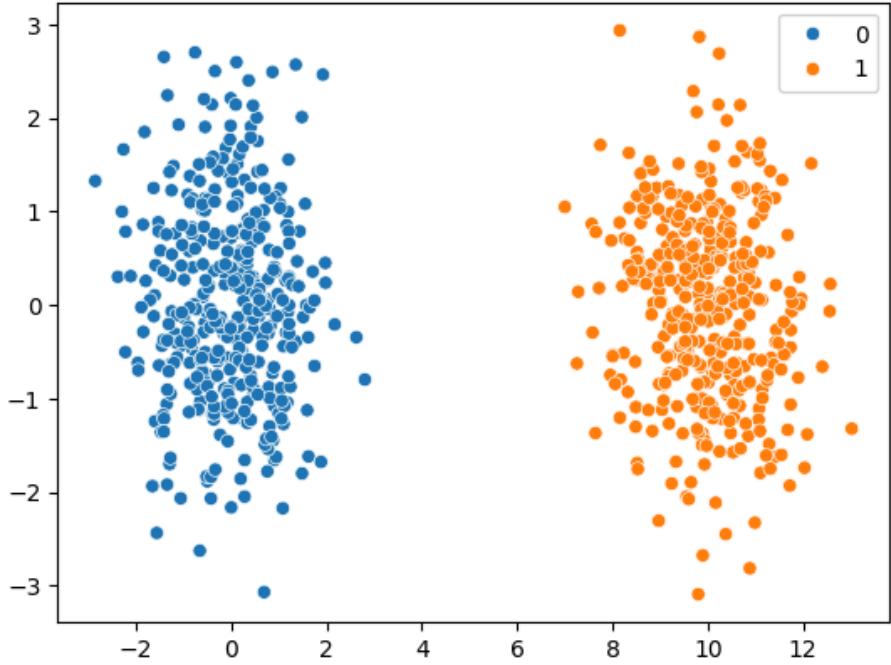


Figura 2: `make_blobs(n_features=2, centers=((0, 0), (10, 0)), random_state=1984)`

En este ejemplo,  $d_{\mathcal{M}} = d_x = 2$ ;  $k = 2$ ;  $n_1 = n_2 = 400$  tenemos dos clases perfectamente separables, con lo cual cualquier clasificador razonable debería alcanzar  $\text{exac} \approx 1$ ,  $\ell \approx 0$ ,  $R^2 \approx 1$ . La evaluación de nuestros clasificadores resulta ser:

	$\ell$	$R^2$	exac
$\mathcal{F}$ -KDC	-0.0	1.0	1.0
KDC	-0.0	1.0	1.0
GNB	-0.0	1.0	1.0
$k$ -NN	-0.0	1.0	1.0
$\mathcal{F}$ -kNN	-0.0	1.0	1.0
LR	-4.5678	0.9589	1.0
SVC			1.0
LSVC			1.0
base	-111.1567	0.0	0.4625

Tabla 1: Resultados de entrenamiento en Figura 2

¡Excelentes noticias! Todos los clasificadores bajo estudio tienen exactitud perfecta, y salvo por una ligeramente negativa  $\ell$  para LR, el resto

da exactamente 0. Pasemos entonces a algunos dataset mínimamente más complejos.

#### 4.2.10 Datasets sintéticos baja dimensión

Consideremos ahora algunas curvas unidimensionales embebidas en  $\mathbb{R}^2$ :

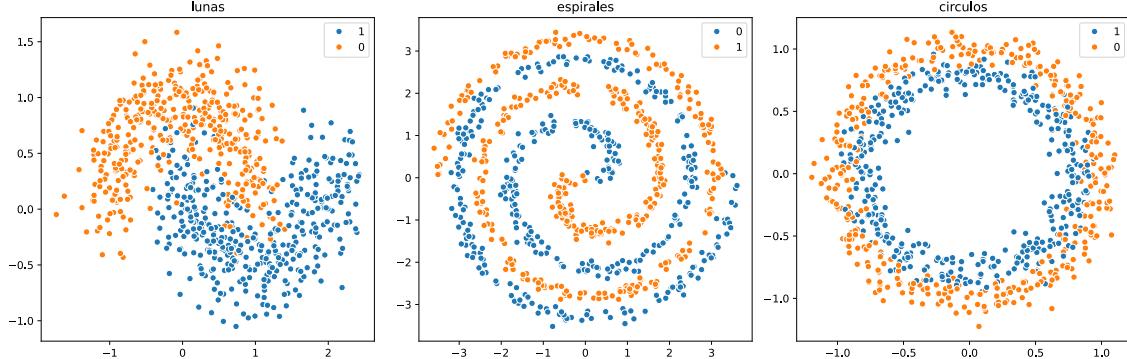


Figura 3: «Lunas», «Círculos» y «Espirales», con  $d_x = 2$ ,  $d_{\mathcal{M}} = 1$  y  $s = 4107$

Resultará obvio al lector que los conjuntos de datos expuestos en Figura 3 no son exactamente variedades «1D» embebidas en «2D», sino que tienen un poco de «ruido blanco» agregado para incrementar la dificultad de la tarea.

**Definición 4.2.10.1 (ruido blanco):** Sea  $X = (X_1, \dots, X_d) \in \mathbb{R}^d$  una variable aleatoria tal que  $E(X_i) = 0$ ,  $\text{Var}(X_i) = \sigma \forall i \in [d]$ . Llamaremos «ruido blanco con escala  $\sigma$ » a toda realización de  $X$ .

Veamos entonces cómo les fue a los contendientes, considerando primero la exactitud. Recordemos que para cada experimento se realizaron 16 repeticiones: en cada celda reportaremos la exactitud *promedio*, y a su lado entre paréntesis el error estándar cpte.:

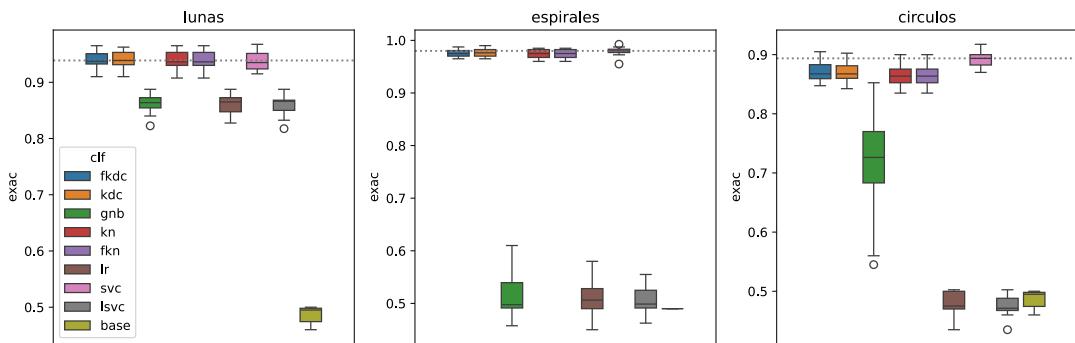


Figura 4: Boxplots con la distribución de dxactitud en las 16 repeticiones de cada experimento de Figura 3

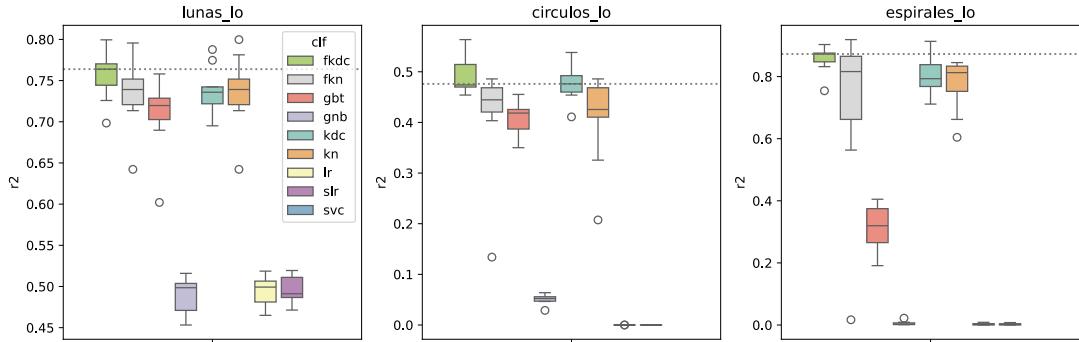


Figura 5: Boxplots con la distribución de dxactitud en las 16 repeticiones de cada experimento de Figura 3

dataset	circulos	circulos	espirales	espirales	lunas	lunas
	mean	std	mean	std	mean	std
clf						
base	48.61	1.42	49.0	0.0	48.61	1.42
fkdc	65.31	2.79	84.88	2.18	81.7	2.22
fkn	64.69	2.46	84.97	2.57	81.86	2.29
gnb	63.98	3.37	51.41	4.0	80.5	2.1
kdc	65.38	2.98	85.09	2.21	81.8	2.44
kn	64.69	2.46	84.97	2.57	81.86	2.29
lr	48.12	1.74	51.88	3.01	80.25	2.17
lsvc	47.86	1.79	50.55	2.68	80.52	1.99
svc	67.8	2.6	86.8	1.82	82.02	2.44

Tabla 2: «mi caption, bo».

KDC (en sus dos variantes), KNN y SVC (con kernel RBF) parecieran ser los métodos más competitivos, con mínimas diferencias de performance entre sí: sólo en «círculos» se observa un ligero ordenamiento de los métodos,  $svc > KDC > k - NN$ , aunque la performance mediana de svc está dentro de «los bigotes» de todos los métodos antedichos. La tarea «lunas» pareciera ser la más fácil de todas, en la que hasta una regresión logística sin modelado alguno es adecuada. Para «espirales» y «círculos», GNB, LR y LSVR no logran performar significativamente mejor que el clasificador base.

Definición 4.2.10.2 (clasificador base):

¿Cómo se comparan los métodos en términos de la log-verosimilitud y el  $R^2$ ?

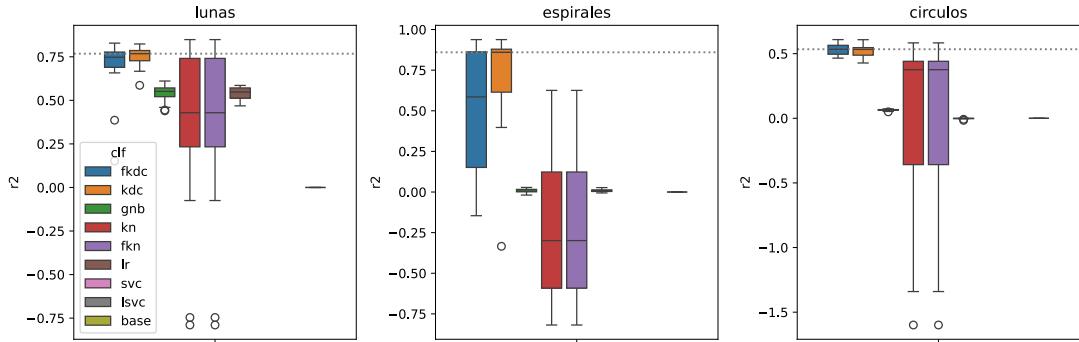


Figura 6: Boxplots con la distribución de  $R^2$  en las 16 repeticiones de cada experimento.

dataset	circulos	circulos	espirales	espirales	lunas	lunas
	mean	std	mean	std	mean	std
clf						
base	0.0	0.0	0.0	0.0	0.0	0.0
fkdc	0.077	0.061	-0.067	1.596	0.347	0.133
fkn	-0.082	0.465	-3.081	3.123	0.299	0.153
gnb	0.05	0.009	0.007	0.012	0.384	0.045
kdc	0.094	0.026	-0.556	2.374	0.374	0.06
kn	-0.082	0.465	-3.081	3.123	0.299	0.153
lr	-0.002	0.004	0.009	0.009	0.364	0.044

Tabla 3: «mi caption, bo-bo».

Como los métodos basados en máquinas de soporte vectorial resultan en clasificadores *duros* ([Definición 4.0.25.1](#)), no es posible analizar la log-verosimilitud u otras métricas derivadas. De entre los dos métodos con exactitud similar a esos, es notoriamente mejor el  $R^2$  que alcanzan ambos KDC. A primera vista, se ve que la dispersión de la métrica es considerable, pues las «cajas» del rango intercuartil son bastante amplias, y aún así se observan *outliers*. En las tres tareas, los clasificadores de estimación de densidad por núcleos tienen las cajas más angostas y los bigotes más cortos, con KDC mostrando una dispersión menor o igual que  $\mathcal{F}$ -KDC. En la Figura 4, observamos que la exactitud de los métodos de k vecinos más cercanos era muy similar a la de KDC y svc, sin embargo en términos de  $R^2$ ,

- en el dataset de «espirales» el  $R^2$  promedio y mediano son *negativos*, y
- en el de «círculos», aunque la locación<sup>10</sup> es positiva, la distribución tiene una pesada cola a izquierda, que entra de lleno en los negativos.

<sup>10</sup>Entendemos tanto al *promedio* o *media* y la *mediana* como *medidas de locación*

En otras palabras, pareciera ser que aunque la exactitud de los métodos basados en vecinos más cercanos es buena, cuando clasifican *mal*, lo hacen con *alta seguridad*, lo que resulta en un pésimo  $R^2$ .

En esta terna inicial de *datasets*, obtenemos unos resultados aceptables:

- Observamos que el clasificador de (Loubes y Pelletier, 2008) es competitivo (es decir que no sólo Loubes y Pelletier propusieron ),
- aunque la distancia de Fermat muestral no parece mejorar significativamente la exactitud de los clasificadores en ella basados.

Que la bondad de los clasificadores *no empeore* con el uso de  $D_{Q,\alpha}$  en lugar de  $\|\cdot\|_2$  es importante. Por una parte, cuando  $\alpha = 1$  y  $n \rightarrow \infty$ ,  $D_{Q,\alpha} \rightarrow \mathcal{D}_{f,\beta} = \|\cdot\|_2$ , con lo cual  $\mathcal{F}$ -KDC debería performar al menos tan bien como KDC cuando la grilla de hiperparámetros en la que lo entrenamos incluye a  $\alpha = 1$ . Sin embargo, el cómputo de  $D_{Q,\alpha}$  es numéricamente bastante complejo, y bien podríamos haber encontrado dificultades computacionales<sup>11</sup>.

#### 4.2.10.1 Comparación entre KDC y $\mathcal{F}$ -KDC para ("circulos", 4479)

Concentrémosnos en un segundo en una corrida específica de un experimento particular. Por caso, tomemos el dataset «circulos», con la semilla 4479. Los parámetros óptimos de  $\mathcal{F}$ -KDC resultaron ser (`alpha: 1.1875, bandwidth: 0.0562`) , mientras que los de KDC fueron (`bandwidth: 0.1202`). Los anchos de banda son diferentes, y el  $\alpha$  óptimo encontrado por  $\mathcal{F}$ -KDC es distinto de 1. Sin embargo, la exactitud de  $\mathcal{F}$ -KDC fue 0.885, y la de KDC, 0.88, prácticamente idénticas<sup>12</sup>. ¿Por qué? ¿Será que los algoritmos no son demasiado sensibles a los hiperparámetros elegidos?

Recordemos que la elección de hiperparámetros se hizo con una búsqueda exhaustiva por convalidación cruzada de 5 pliegos. Por lo tanto, durante el *entrenamiento* se generaron suficientes datos como para graficar la exactitud promedio en los pliegos, en función de  $(\alpha, h)$ . A esta función de los hiperparámetros a una función de pérdida<sup>13</sup> se la suele denominar *superficie de pérdida*.

---

<sup>11</sup>De hecho, hubo montones de ellas, cuya resolución progresiva dio lugar a la pequeña librería que acompaña esta tesis y documentamos en el anexo. A mi entender, ningún error de cálculo persiste en el producto final

<sup>12</sup>Con 400 observaciones para evaluación, dichos porcentajes representan 352 y 354 observaciones correctamente clasificadas, resp.

<sup>13</sup>En realidad, la exactitud es un «score» o puntaje - mientras más alto mejor-, pero el negativo de cualquier puntaje es una pérdida - mientras más bajo, mejor.

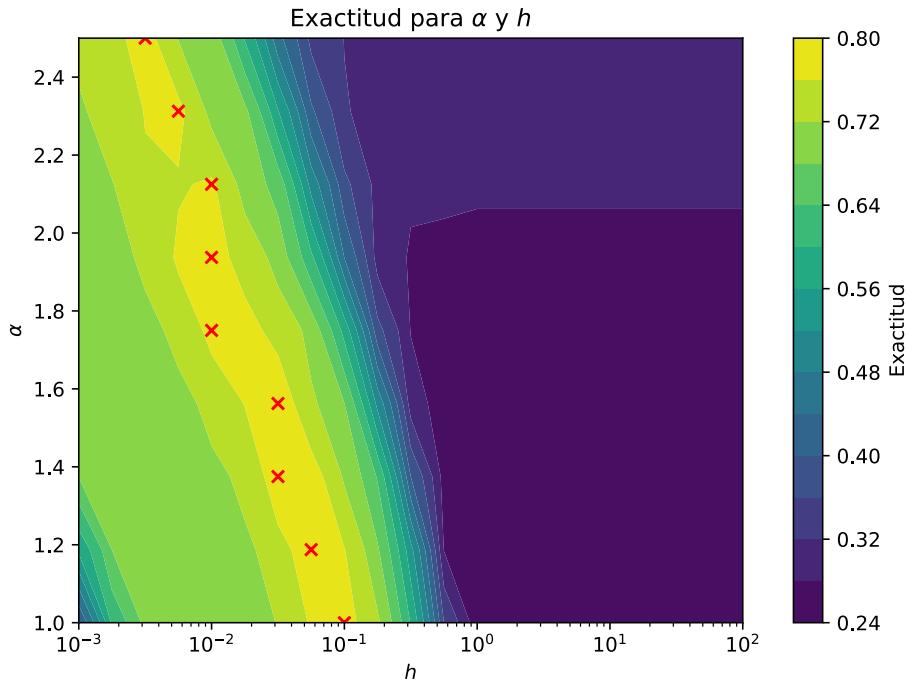


Figura 7: Exactitud promedio en entrenamiento para la corrida ("[círculos](#)", [4479](#)). Las cruces rojas indican la ventana  $h$  óptima para cada  $\alpha$ .

Nótese que la región amarilla, que representa los máximos puntajes durante el entrenamiento, se extiende diagonalmente a través de todos los valores de  $\alpha$ . Es decir, no hay un *par* de hiperparámetros óptimos  $(\alpha^*, h^*)$ , sino que fijando  $\alpha$ , siempre parecería existir un(os)  $h^*(\alpha)$  que alcanza (o aproxima) la máxima exactitud *possible* con el método en el dataset. En este ejemplo en particular, hasta parecería ser que una relación log-lineal captura bastante bien el fenómeno,  $\log(h^*) \propto \alpha$ . En particular, entonces,  $\text{exac}(h^*(1), 1) \approx \text{exac}(h^*, \alpha^*)$ , y se entiende que el algoritmo  $\mathcal{F}$ -KDC, que agrega el hiperparámetro  $\alpha$  a KDC no mejore significativamente su exactitud.

Ahora bien, esto es sólo en *un* dataset, con *una* semilla específica. ¿Se replicará el fenómeno en los otros datasets estudiados? Y si tomásemos datasets con otras características?

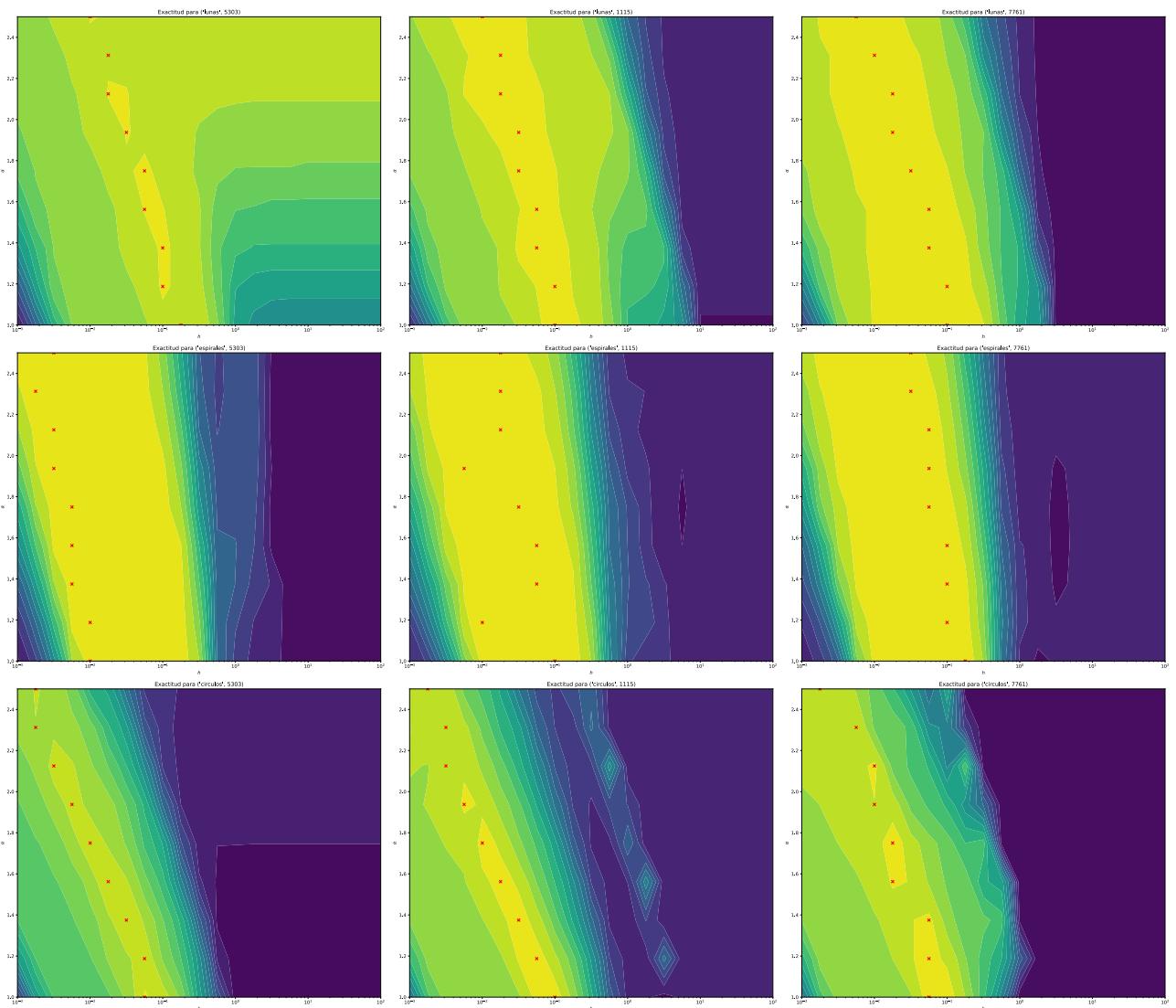


Figura 8: It does replicate

Antes de avanzar hacia el siguiente conjunto de datos, una pregunta más: ¿qué sucede si aumentamos el nivel de ruido? Es decir, mantenemos los dataset hasta aquí considerados, pero subimos  $\sigma$  de [Definición 4.2.10.1](#)?

**4.2.10.2 Efecto del ruido en la performance de clasificación**

**4.2.10.3 Datasets reales en «mediana» dimensión**

**4.2.10.4 Dígitos**

**4.2.10.5 PCA-red MNIST**

## **5 Análisis de Resultados**

**5.0.1 Datasets sintéticos, Baja dimensión**

**5.0.2 Datasets orgánicos, Mediana dimensión**

**5.0.3 Alta dimensión: Dígitos**

**5.0.4 Efecto de dimensiones guillemot ruidosasguillemot**

**5.0.5 fKDC: Interrelación entre  $h, \alpha$**

**5.0.6 fKNN: Comportamiento local-global**

## **6 Comentarios finales**

**6.0.1 Conclusiones**

**6.0.2 Posibles líneas de desarrollo**

**6.0.3 Relación con el estado del arte**

## **7 Referencias**

## **8 Código Fuente**

**8.0.1 sklearn**

**8.0.2 fkdc**

**8.0.3 Datasets**

**8.0.4 Datasets 2d**

Esto digo yo

(Rosenblatt, 1956; Chacón y Duong, 2013; Carpio *et al.*, 2019)

## **Listado de Figuras**

Figura 1 La variedad $\mathcal{U}$ con $\dim(\mathcal{U}) = 1$ embebida en $\mathbb{R}^2$ . Nótese que en el espacio ambiente, el punto rojo está más cerca del verde, mientras que a través de $\mathcal{U}$ , el punto amarillo está más próximo que el rojo .....	45
Figura 2 <code>make_blobs(n_features=2, centers=((0, 0), (10, 0)), random_state=1984)</code> .....	51
Figura 3 «Lunas», «Círculos» y «Espirales», con $d_x = 2$ , $d_{\mathcal{M}} = 1$ y $s = 4107$ .....	52
Figura 4 Boxplots con la distribución de dxactitud en las 16 repeticiones de cada experimento de Figura 3 .....	52
Figura 5 Boxplots con la distribución de dxactitud en las 16 repeticiones de cada experimento de Figura 3 .....	53
Figura 6 Boxplots con la distribución de $R^2$ en las 16 repeticiones de cada experimento. ....	54
Figura 7 Exactitud promedio en entrenamiento para la corrida ( <code>"circulos"</code> , <a href="#">4479</a> ). Las cruces rojas indican la ventana $h$ óptima para cada $\alpha$ . ....	56
Figura 8 It does replicate .....	57

## Listado de Tablas

Tabla 1 Resultados de entrenamiento en Figura 2 .....	51
Tabla 2 «mi caption, bo». ....	53
Tabla 3 «mi caption, bo-bo». ....	54

## Listado de código

Listado 1 Un cacho e' código .....	1
------------------------------------	---

## Bibliografía

- Bijral, A.S., Ratliff, N. y Srebro, N. (2012) «Semi-Supervised Learning with Density Based Distances». arXiv. Disponible en: <https://doi.org/10.48550/arXiv.1202.3702>.
- Carpio, A. *et al.* (2019) *Fingerprints of Cancer by Persistent Homology*. Disponible en: <https://doi.org/10.1101/777169>.
- Chacón, J.E. y Duong, T. (2013) «Data-Driven Density Derivative Estimation, with Applications to Nonparametric Clustering and Bump Hunting», *Electronic Journal of Statistics*, 7(none). Disponible en: <https://doi.org/10.1214/13-EJS781>.
- Groisman, P., Jonckheere, M. y Sapienza, F. (2019) «Nonhomogeneous Euclidean First-Passage Percolation and Distance Learning». arXiv.

- Little, A., McKenzie, D. y Murphy, J. (2021) «Balancing Geometry and Density: Path Distances on High-Dimensional Data». arXiv.
- Loubes, J.-M. y Pelletier, B. (2008) «A Kernel-Based Classifier on a Riemannian Manifold», *Statistics & Decisions*, 26(1), pp. 35-51. Disponible en: <https://doi.org/10.1524/stnd.2008.0911>.
- Rosenblatt, M. (1956) «Remarks on Some Nonparametric Estimates of a Density Function», *The Annals of Mathematical Statistics*, 27(3), pp. 832-837.
- Vincent, P. y Bengio, Y. (2003) «Density Sensitive Metrics and Kernels», en *Proceedings of the Snowbird Workshop*.