Laboratorio de datos, clase 9

Bienvenid@s al machine learning

Prof. Enzo Tagliazucchi

tagliazucchi.enzo@googlemail.com
www.cocucolab.org

Programa de hoy

17:00 a 17:45 diapositivas
17:45 a 18:30 primer notebook (desbalance)
18:30 a 18:45 intervalo
18:45 a 19:00 segundo notebook (normalización)
19:00 a 19:15 puesta en común
19:15 a 19:45 tercer notebook (hiperparámetros)
19:45 a 20:00 puesta en común

¿Qué vimos hasta ahora?

Problema de regresión o clasificación

$$(x_{1i}, ..., x_{ni}, y_i)$$

n variables independienes (*features*)

k ejemplos (*samples*)

la variable dependiente (*target*) puede ser 0 o 1 (clasificación) o tomar valores reales (regresión)

¿Qué vimos hasta ahora?

Nuestro *modelo* viene dado por una función

$$f(x_1, ..., x_n; \beta_1, ..., \beta_m)$$

cuyo objetivo es aproximar a los *targets*

Para eso, primero obtengo los *targets* estimados

$$\widetilde{y_i} = f(x_{1i}, ..., x_{ni}; \beta_1, ..., \beta_m)$$

¿Qué vimos hasta ahora?

Luego, mediante alguna función de distancia calculo la diferencia entre *targets* y *targets estimados* (función de *costo* o *pérdida*)

$$d\left((y_1, ..., y_k), (\widetilde{y}_1, ..., \widetilde{y}_k)\right) = L(\beta_1, ..., \beta_m)$$

(si *d* es la distancia euclídea tengo cuadrados mínimos). Los *targets estimados* dependen de los parámetros. Busco los que minimicen la *L*:

$$(\widetilde{\beta}_1, ..., \widetilde{\beta}_m) = \text{argmin}\left(L(\beta_1, ..., \beta_m)\right)$$ Entrenamiento

¿Qué vimos hasta ahora?

Si *m* (número de parámetros es grande), puedo hacer que la función de costo sea chica sobreajustando (*overfitteando*) los datos.

Vimos dos soluciones posibles: regularización *ridge* y regularización *lasso*.

$$(\widetilde{\beta}_1, \ldots, \widetilde{\beta}_m) = \operatorname{argmin} \left(L(\beta_1, \ldots, \beta_m) + C \sum_{j=1}^{m} \beta_j^2\right)$$ **Ridge**

$$(\widetilde{\beta}_1, \ldots, \widetilde{\beta}_m) = \operatorname{argmin} \left(L(\beta_1, \ldots, \beta_m) + C \sum_{j=1}^{m} |\beta_j|\right)$$ **Lasso**

Y esto es el *aprendizaje supervisado*

$$f(x_1, ..., x_n; \beta_1, ..., \beta_m)$$

aunque aún hay cosas para elegir

$$\widetilde{y}_i = f(x_{1i}, ..., x_{ni}; \beta_1, ..., \beta_m)$$

$$d\left((y_1, ..., y_k), (\widetilde{y}_1, ..., \widetilde{y}_k)\right) = L(\beta_1, ..., \beta_m)$$

$$(\widetilde{\beta}_1, ..., \widetilde{\beta}_m) = \operatorname{argmin}\left(L(\beta_1, ..., \beta_m) + C \sum_{j=1}^{m} \beta_j^2\right)$$

$$(\widetilde{\beta}_1, ..., \widetilde{\beta}_m) = \operatorname{argmin}\left(L(\beta_1, ..., \beta_m) + C \sum_{j=1}^{m} |\beta_j|\right)$$

Cosas aún por elegir

1. ¿Cómo elijo el modelo *f*?
2. ¿Cómo elijo la función de pérdida *L*?

Clase de hoy

3. ¿Cómo determino *C*?
4. ¿Cómo determino la performance del modelo?
5. ¿Cómo preparo los features para el modelo?

¿Cómo determino la performance del modelo?

$$\mathrm{Acc} = \frac{1}{k}\sum_{i=1}^{k} |\widetilde{y}_i - y_i|$$

Accuracy: cantidad de veces que el *label* estimado es correcto sobre cantidad total de *samples*

¿Qué es un valor aceptable de *accuracy*?

Acc = 1 , todos los *labels* estimados correctamente

Acc = 0 , ningún *label* es estimado correctamente

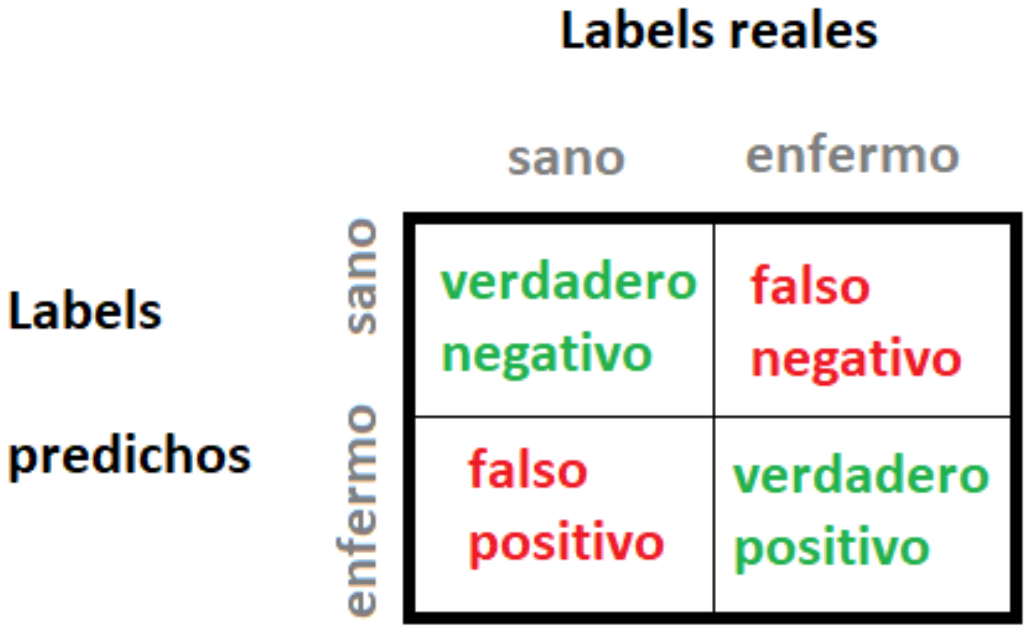
Acc = 0.5, valor que obtengo al azar si las clases **son balanceadas**

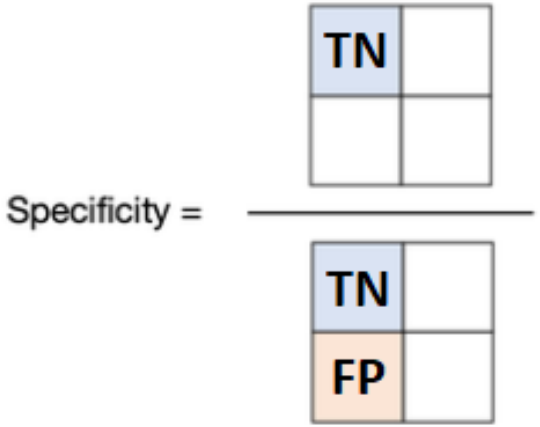
Problema: detectar una enfermedad genética muy rara (1 en un millón)

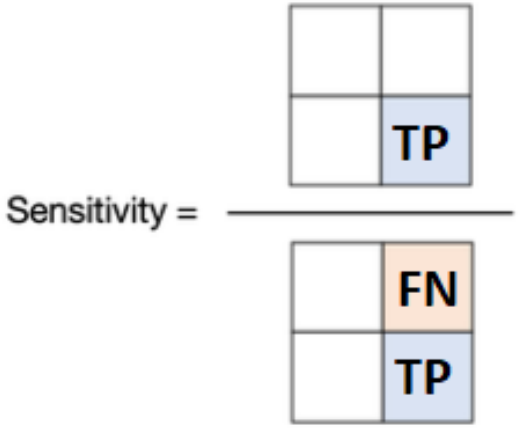
Test de entrenamiento: 999.999 negativos, 1 positivo

Acc = 0.999999

El modelo predice que
nadie tiene la enfermedad





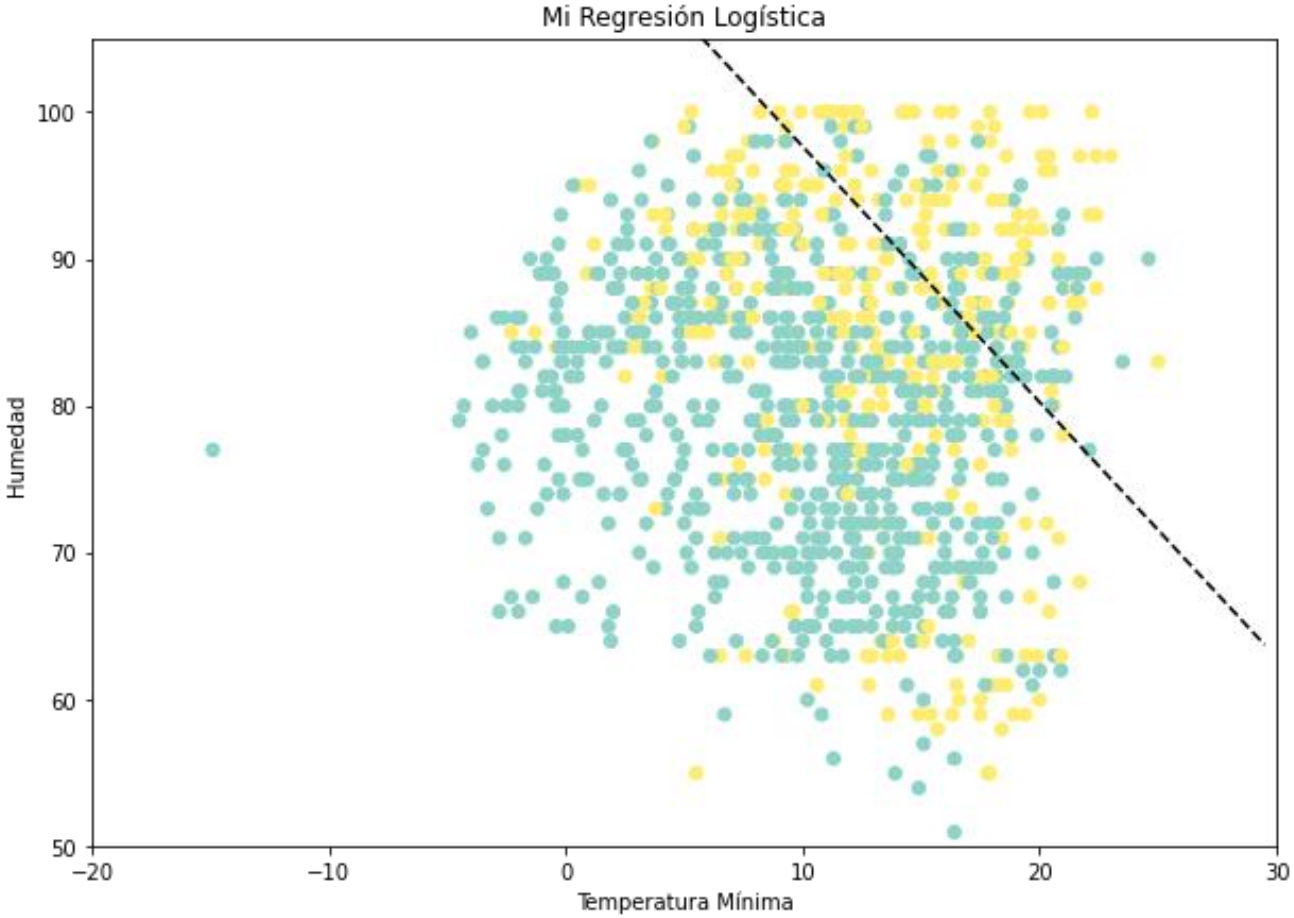


El modelo de la diapo anterior tiene sensitividad muy baja y especificidad muy alta

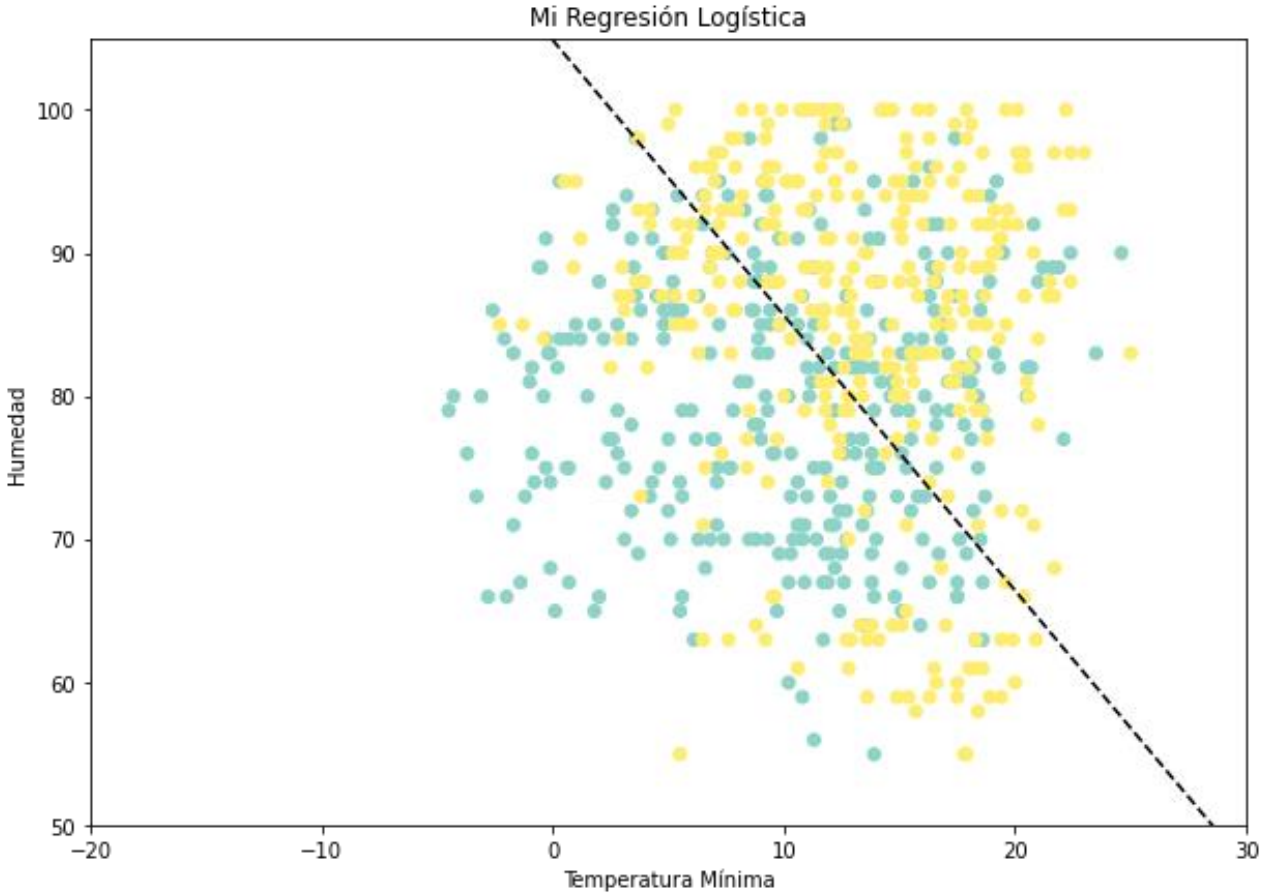
$$\text{Balanced accuracy} = \frac{\text{Sensitivity} + \text{Specificity}}{2}.$$

¿Qué paso si entreno una regresión logística con datos desbalanceados?

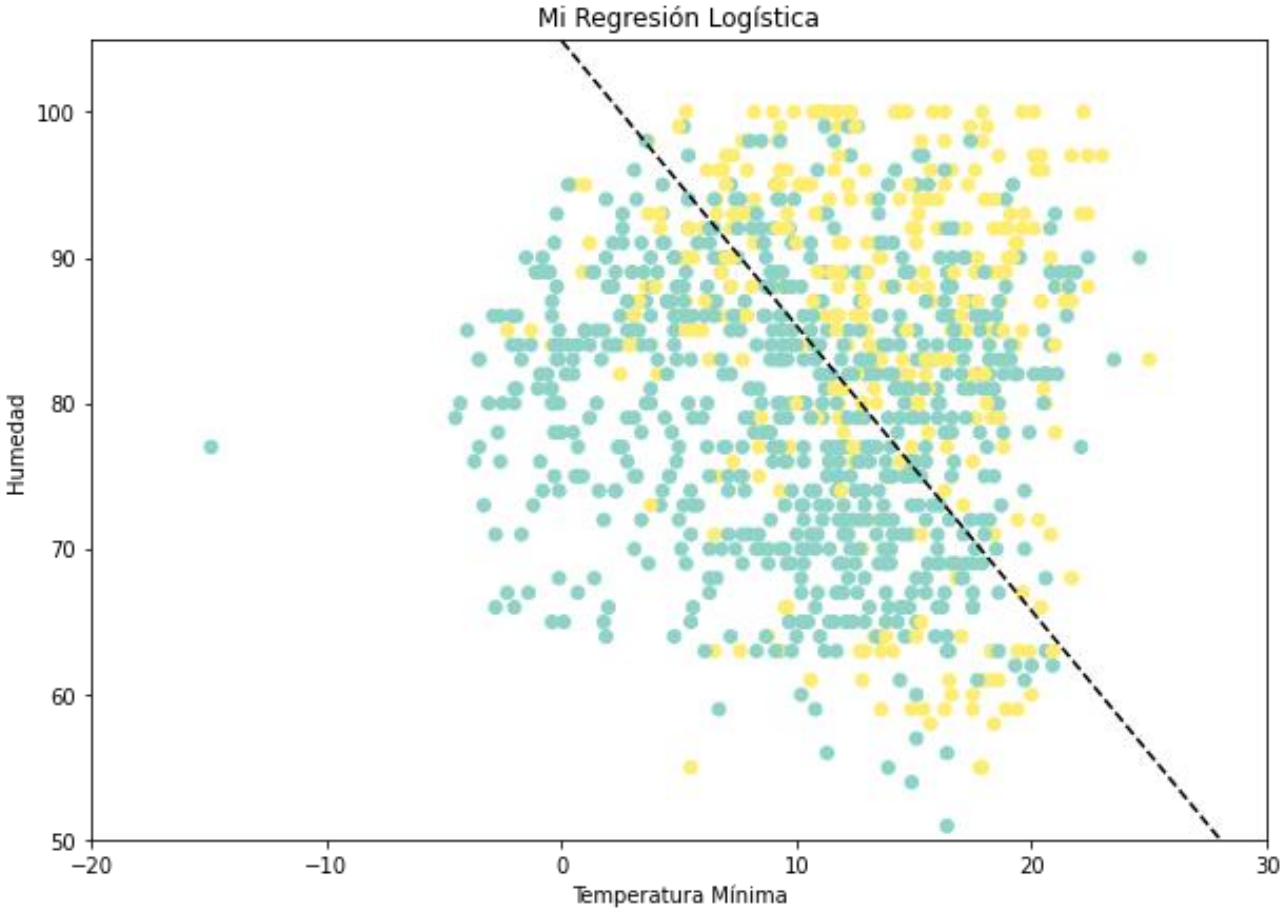
Conviene asegurarse que los días sin lluvia estén bien clasificados, porque son más: **baja sensitividad**



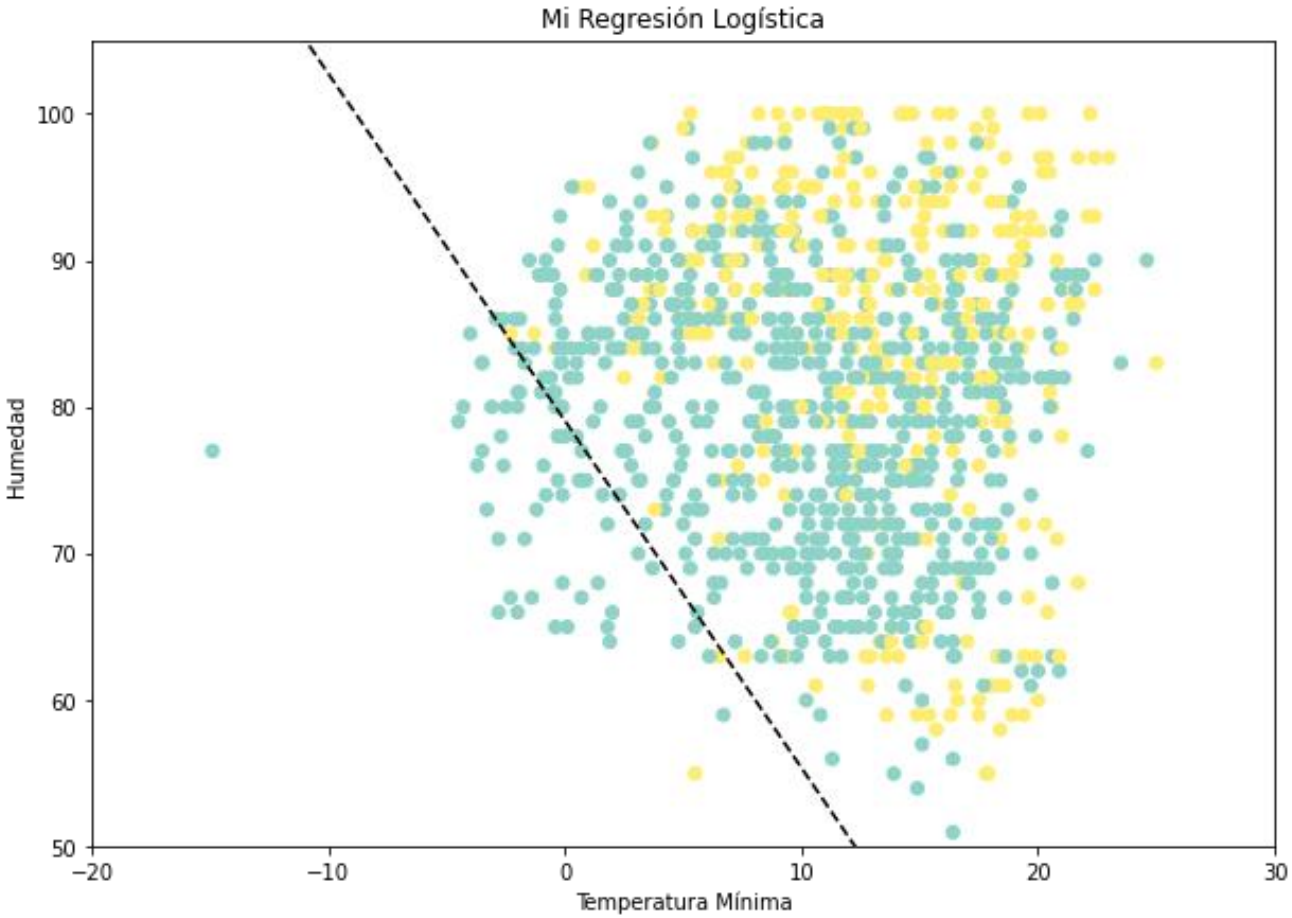
¿Qué pasa si downsampleo los datos para tener la misma cantidad en cada clase?



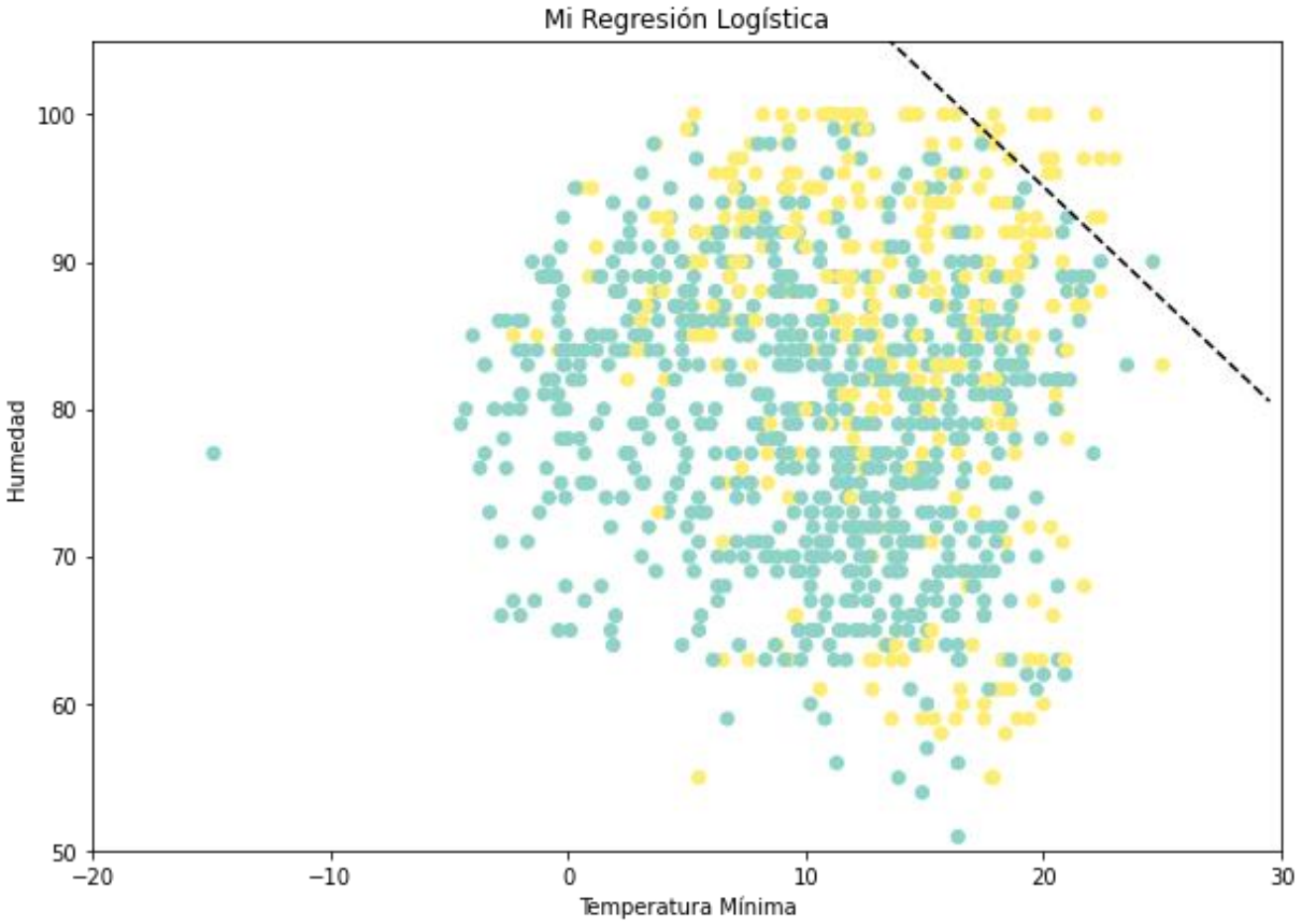
¿Qué pasa si le doy menos peso a los errores de la clase más representada?



¿Qué pasa si le doy menos mucho peso a clasificar mal los días de lluvia?



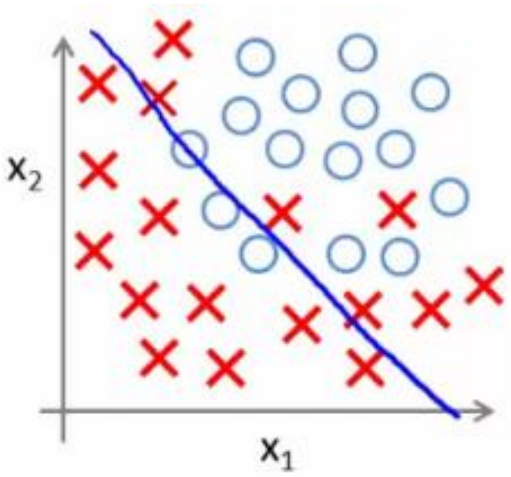
¿Qué pasa si le doy menos mucho peso a clasificar mal los días de NO lluvia?



¿Cómo preparo los *features* para el modelo?

Entreno mi modelo y me da performance baja. No puedo ir a recolectar más datos. ¿Qué hago?

Respuesta: **Sumar features**. Vamos sumando features nuevos basados en features anteriores, por ejemplo, cuadrado, cubo, etc,de esos features o productos entre pares de ellos

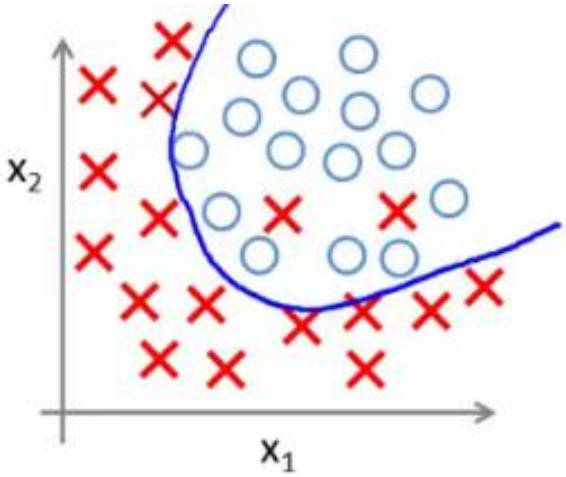


$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$

(g = sigmoid function)

Baja performance en set de entrenamiento

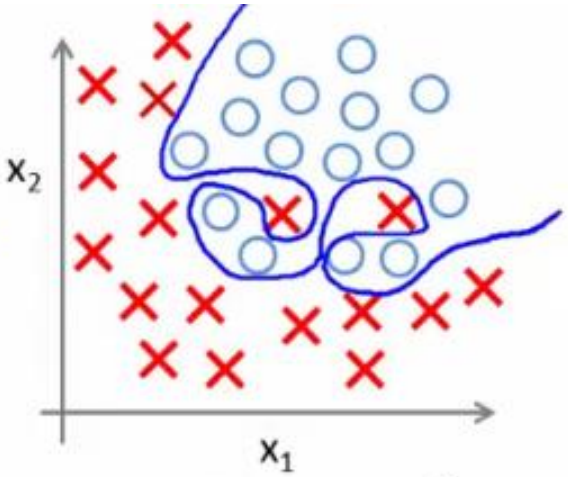
"underfitting"



$g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2 + \theta_5 x_1 x_2)$

Performance media en set de entrenamiento

Buena generalización a otros datos



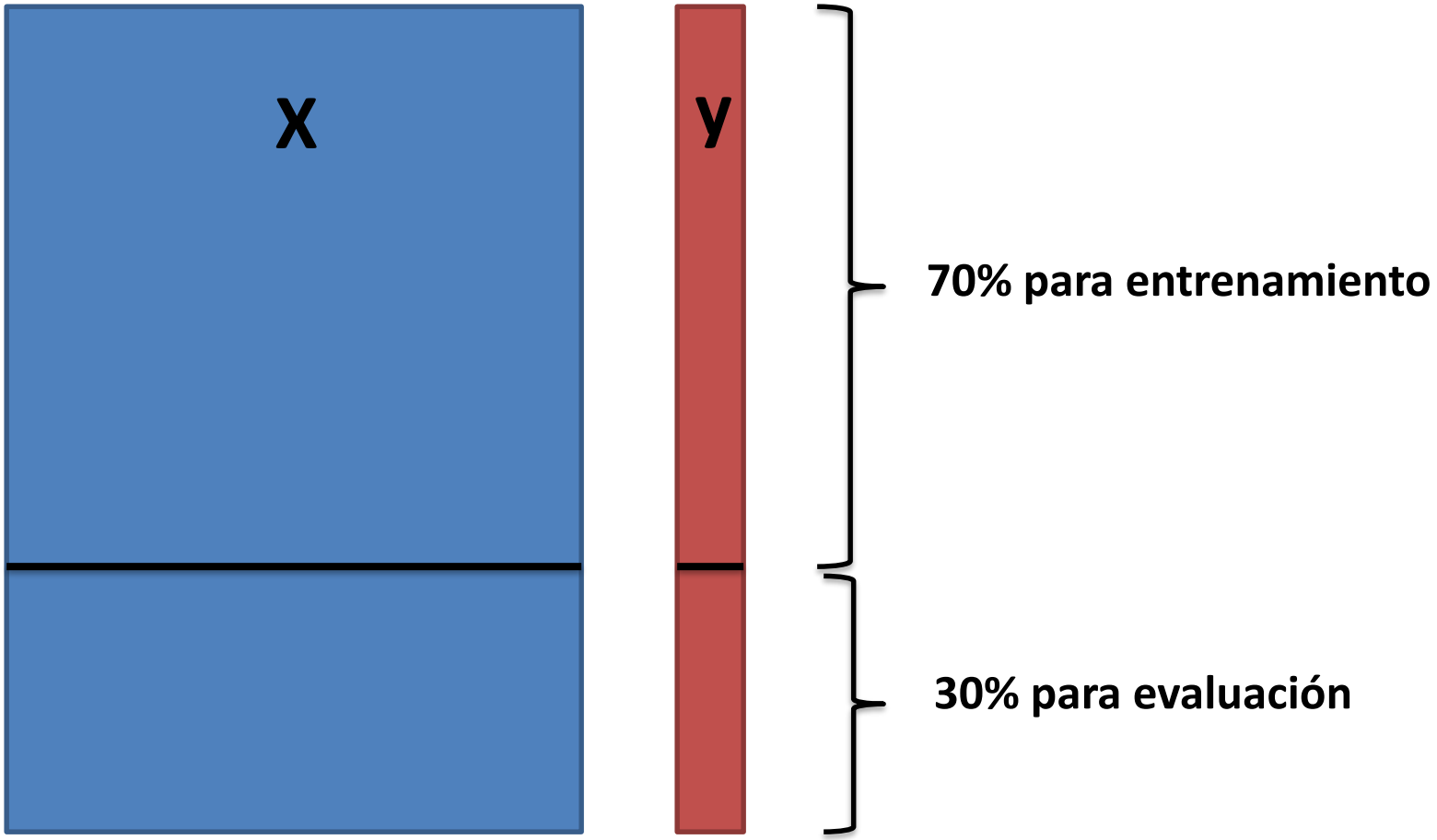
$g(\theta_0 + \theta_1 x_1 + \theta_2 x_1^2 + \theta_3 x_1^2 x_2 + \theta_4 x_1^2 x_2^2 + \theta_5 x_1^2 x_2^3 + \theta_6 x_1^3 x_2 + \ldots)$

Performance muy alta en set de entrenamiento

Mala generalización a otros datos

"overfitting"

Solución: ***train-test split*** (aleatorio)



Alternativa: usar regularización

$$(\widetilde{\beta}_1, \ldots, \widetilde{\beta}_m) = \operatorname{argmin} \left(L(\beta_1, \ldots, \beta_m) + C \sum_{j=1}^{m} \beta_j^2\right)$$

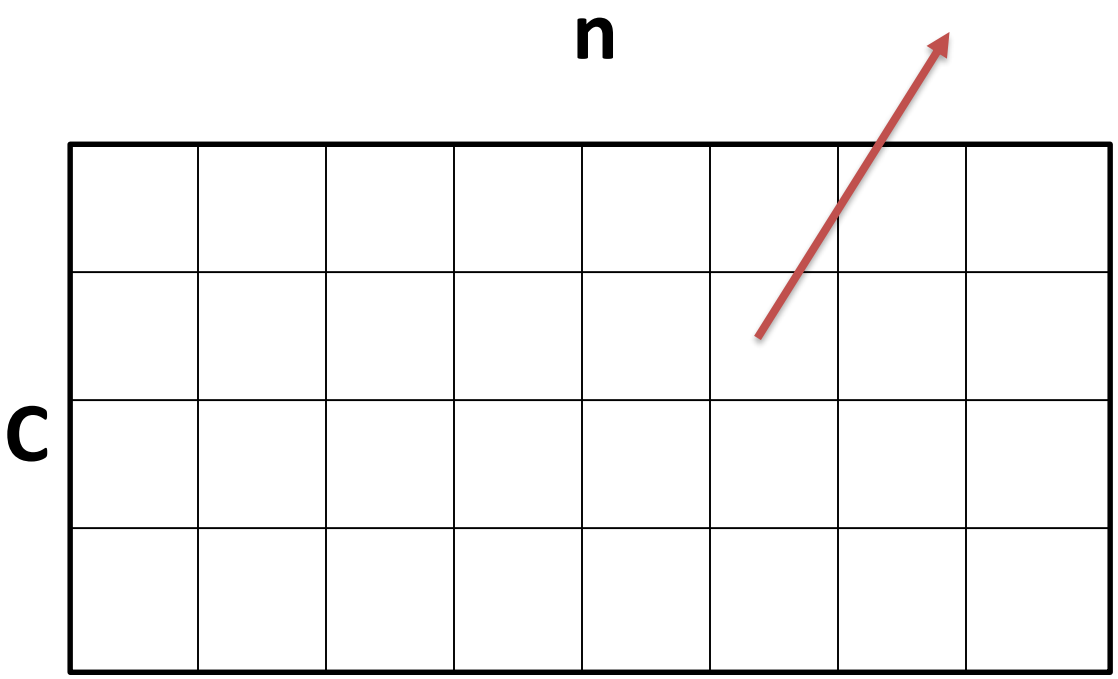
$$(\widetilde{\beta}_1, \ldots, \widetilde{\beta}_m) = \operatorname{argmin} \left(L(\beta_1, \ldots, \beta_m) + C \sum_{j=1}^{m} |\beta_j|\right)$$

Pero... ¿cómo elijo C?

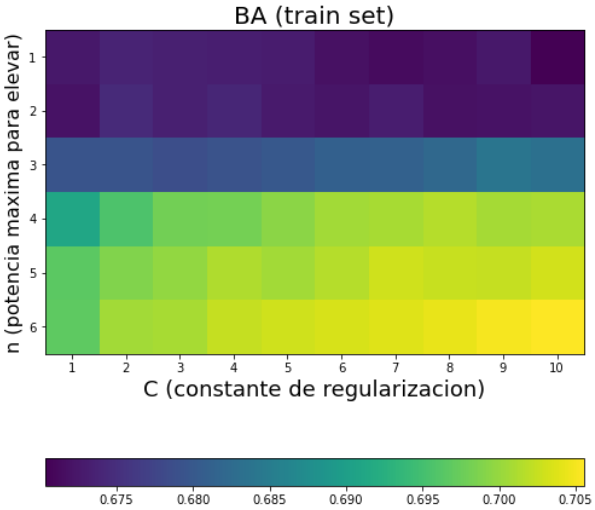
...y ya que estamos... ¿cómo elijo cuantas *features* agregar? por ejemplo, si agrego potencias de features que ya tenía, ¿cuál es la máxima potencia que quiero incluir? (n)

Optimización de hiperparámetros (C, n)

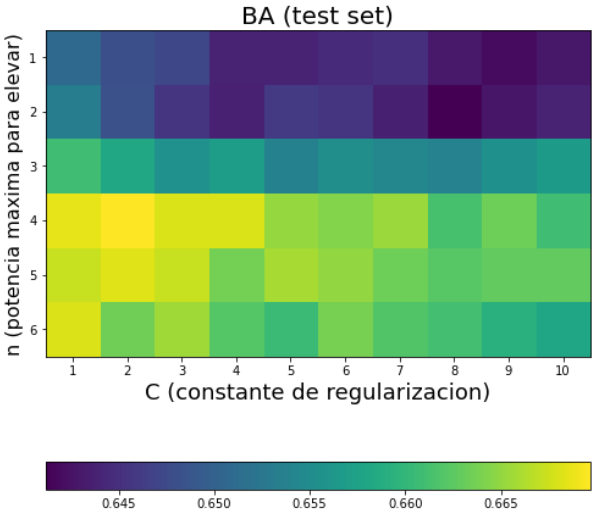
Para cada par (C,n) hago muchas iteraciones dividiendo train-test al azar, calculo una medida de performance promedio para ambos conjuntos (train, test) y los voy poniendo en la grilla.



Caso de predicción de lluvia



En conjunto de entrenamiento aumenta la performance a medida que agrego más features



En cambio, en el conjunto de evaluación necesito además un valor específico de la constante de regularización (para evitar overfitting)

Dos observaciones muy muy muy importantes

1. Cuando divido entre conjuntos de entrenamiento y evaluación, toda transformación que decida hacer tengo que hacerla por separado en cada uno.

De lo contrario, podría transferir información desde el set de entrenamiento al de evaluación sin querer y aumentar artificialmente la performance.

Por ejemplo, si computo z-scores sobre todos los features, al restar la media y dividir por el desvío estándar de todo, meto info de test en train set.

---

2. Cuando hago optimización de hiperparámetros, encuentro la performance máxima de la grilla.

Pero esta performance es demasiado optimista, precisamente porque ya surge de un proceso de optimización.

Para obtener un estimativo que no esté inflado, necesito seleccionar los hiperparámetros que me dan esa performance máxima, entrenar el modelo con ellos, y evaluar la performance en otro datset distinto que no haya sido utilizado para la optimización

Clase de hoy: tres notebooks

Primero: el problema de clases desbalanceadas

Segundo: el problema de la normalización

Tercero: el problema de los hiperparámetros

Objetivo de próximas clases

Tratar de obtener la mejor performance posible en un problema de clasificación

Aprender a estimar correctamente la performance de un modelo

Empezar a explorar otros modelos de clasificación más allá de la regresión logística

Próximo TP (adelanto)

Un dataset de clasificación y el objetivo es usar todo lo que vimos en la materia para obtener la mejor performance posible