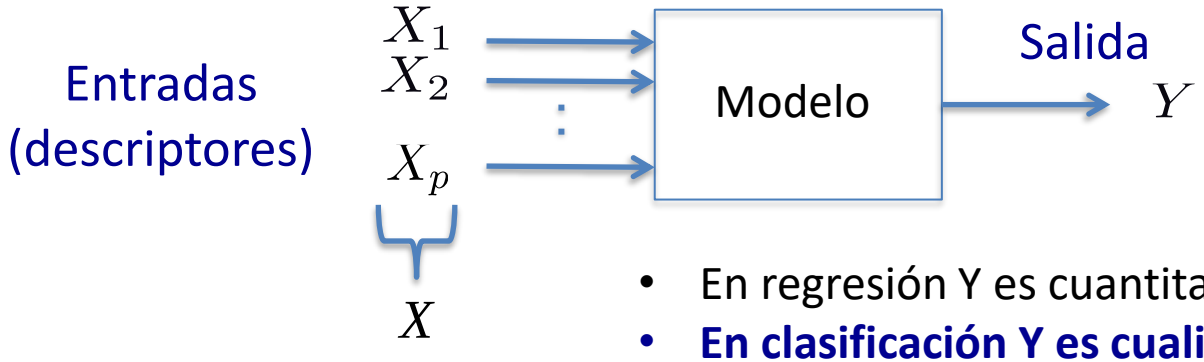
Clasificación y Regresión Logística

Santiago Alférez

edwin.alferez@urosario.edu.co

Diciembre 12 de 2020

Clasificación



- En regresión Y es cuantitativa
- **En clasificación Y es cualitativa (categórica)**

Muchas situaciones:

- Condición de paciente ∈ {sano, enfermo}
- Célula ∈ {*normal*, *drepanocito*, *esferocito*}
- ……

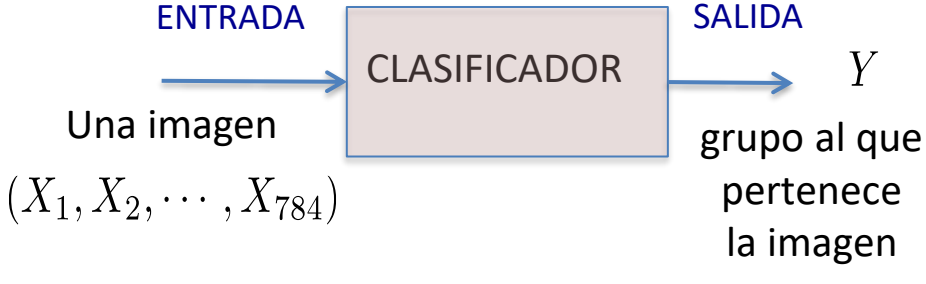
Dado un objeto caracterizado por un vector de descriptores, el **objetivo de un clasificador (modelo)** es predecir a qué clase pertenece el objeto dentro de un conjunto de clases predefinidas.

Clasificación - ejemplo

Base de datos MNIST: 70000 imágenes de los **dígitos (0,1,…., 9)** escritos a mano

Cada imagen contiene un único dígito en escala de grises y está **etiquetada**, es decir se conoce cuál es el valor del dígito.

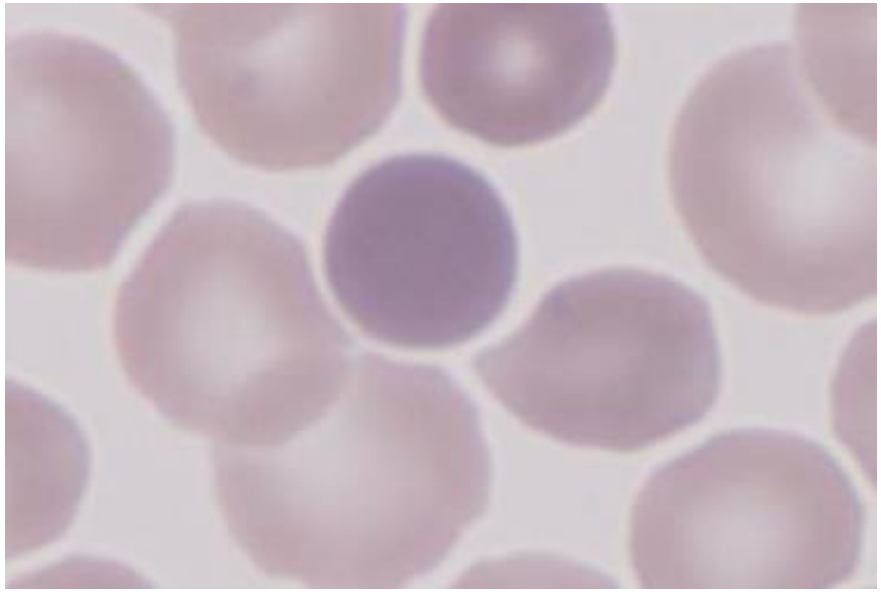
La imagen tiene 28 x 28 píxeles y está representada por **784 descriptores numéricos**, que son los valores de la intensidad de cada uno de los píxeles en un rango entre 0 y 255.



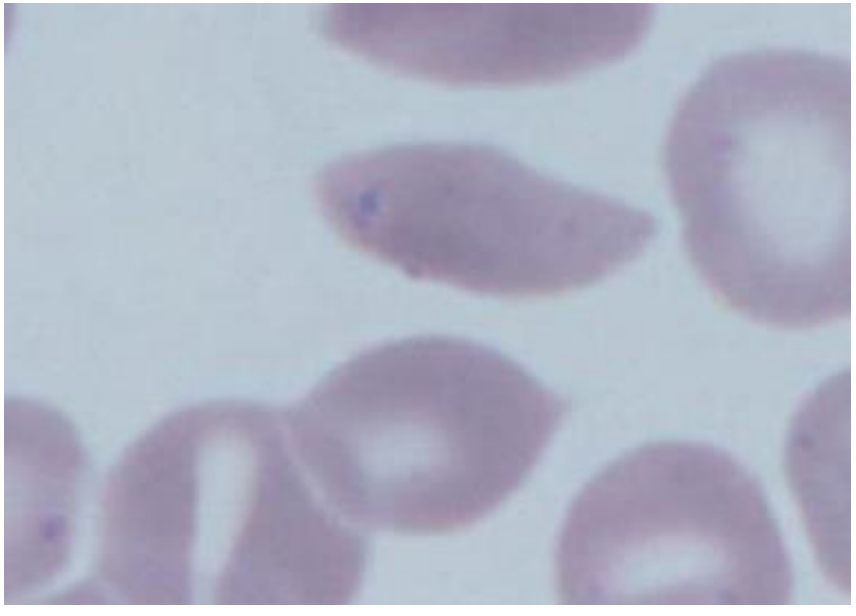
(Cuál es el dígito ?)

Anemias Hemolíticas

Esferocitosis



Drepanocitosis



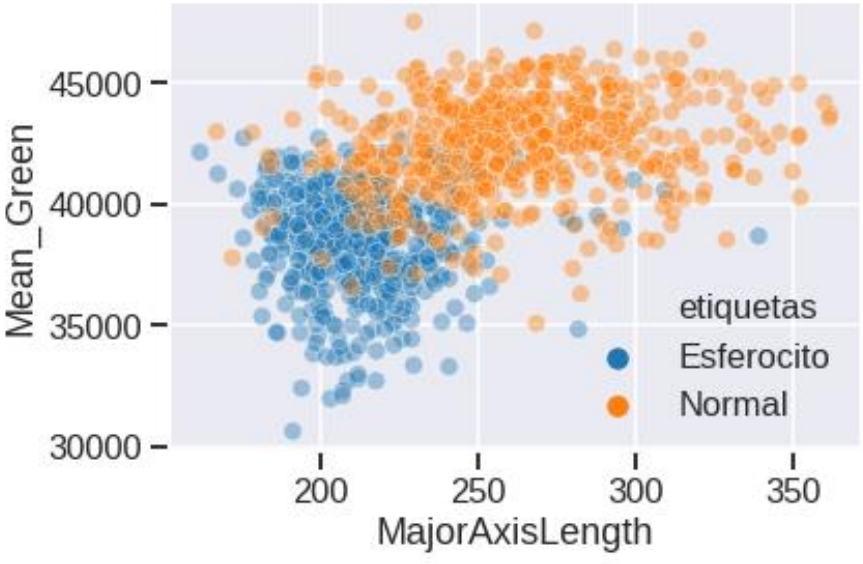
Ejemplo: tipos de glóbulos rojos

	etiquetas	MajorAxisLength	Area_cualitativa	Mean_Green
1565	Normal	305.976345	High	41625.075155
413	Drepanocits	410.030612	High	36756.232305
1428	Normal	285.253194	High	42508.682130
1095	Esferocito	205.203827	Low	38928.427773
856	Esferocito	193.184427	Low	41742.858976
9	Drepanocits	432.632437	High	35829.883802

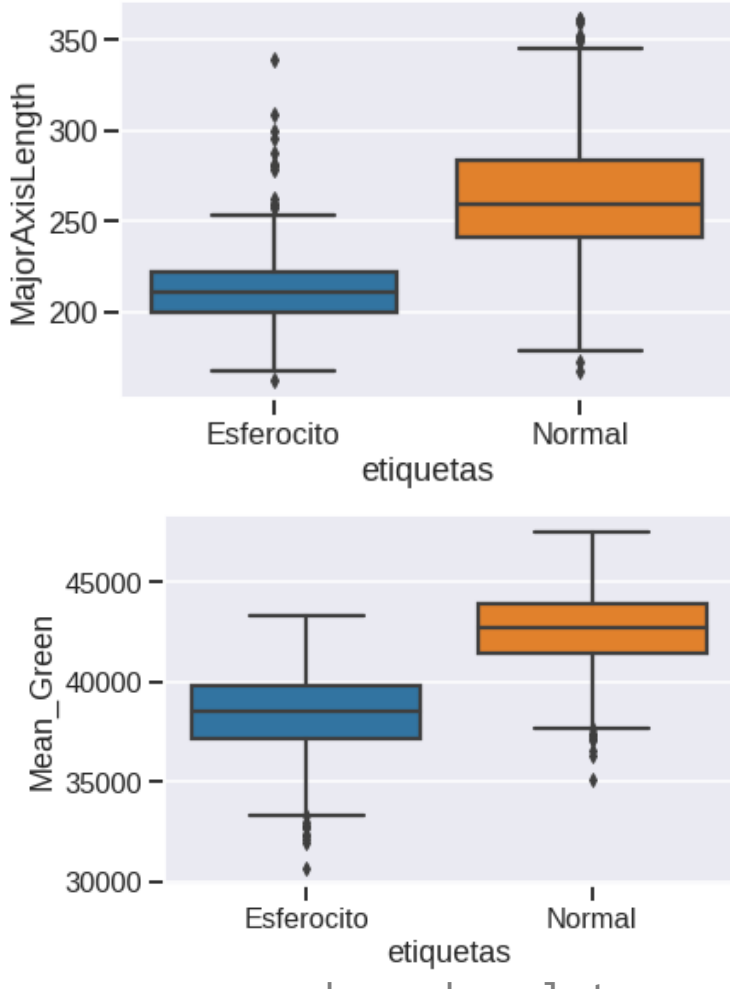
```
1 df['etiquetas'].value_counts()
```

```
Esferocito     552
Normal         552
Drepanocits    552
Name: etiquetas, dtype: int64
```

Ejemplo: tipos de glóbulos rojos



seaborn.scatterplot

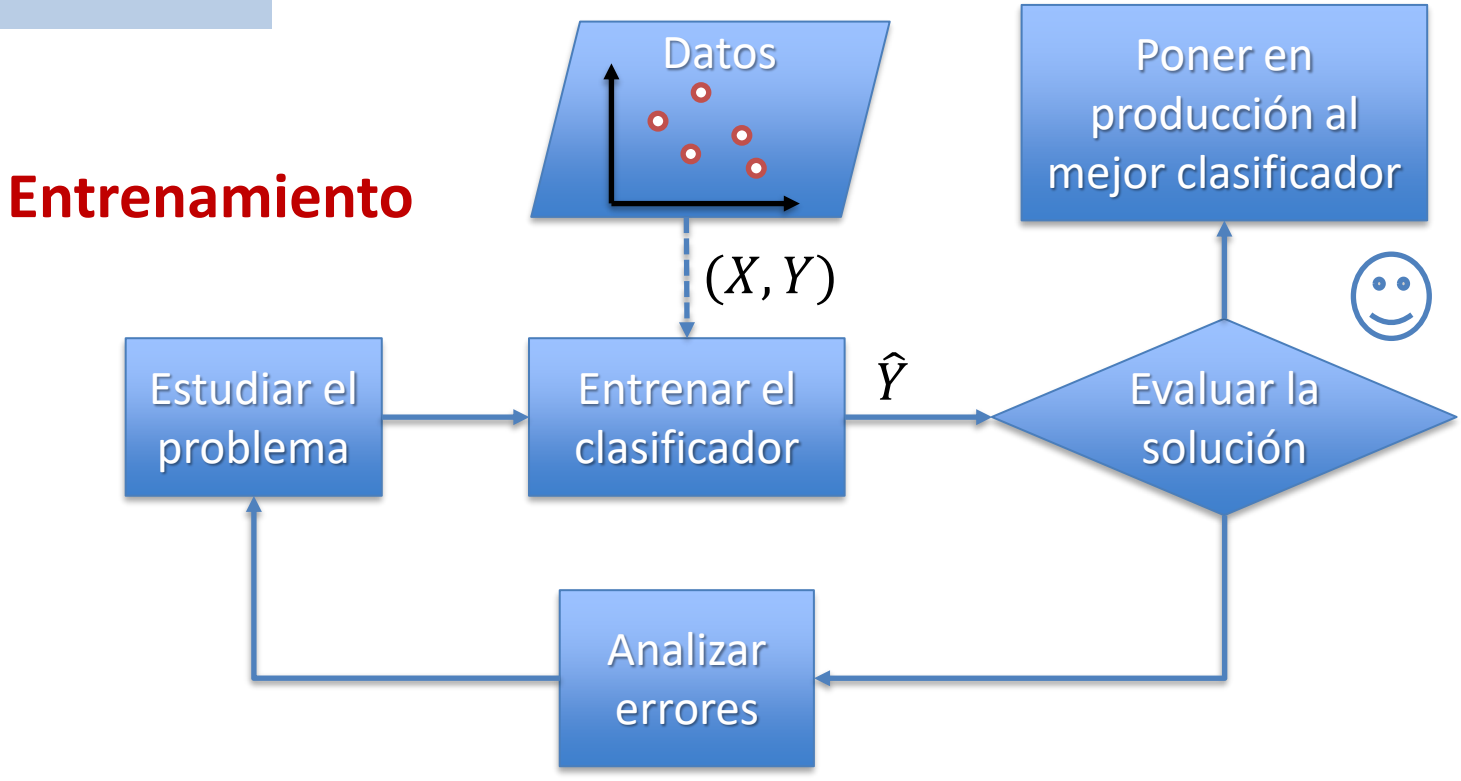


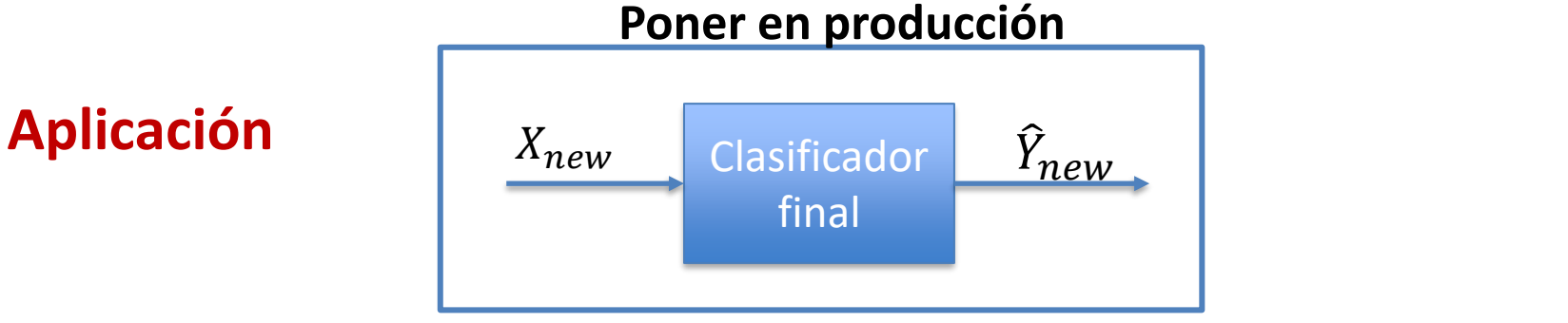
seaborn.boxplot

X1 = Meen_Green.
X2 = MajorAxisLength

Y= tipo de RBC {Normal, Esferocito}

Clasificación





Clasificación

Existen muchos tipos de clasificadores:

- Regresión logística
- Bayes
- Análisis discriminante lineal
- KNN
- Árboles de decisión, arboles aleatorios
- SVM
- Redes neuronales

…….. y otros

Cómo abordar la clasificación

Una forma natural para clasificar un objeto es mediante las **probabilidades** de que, dado un valor de los descriptores $(X = x)$, la salida pertenezca a una u otra de las categorías $(Y = k)$:

$$\Pr(Y = k|X = x)$$

Esto requiere estimar o modelizar la probabilidad condicional de la respuesta *Y* de cada clase dados los descriptores.

¿Cómo se hace?

Clasificador ideal de Bayes

Supongamos que existen K clases $\{label_1, label_2, \dots, label_K\}$, y que se codificadas como $\{1, 2, \dots, K\}$. Sean

$$p_k(x) = \Pr(Y = k|X = x), \qquad k = 1,2 \dots, K.$$

las probabilidades condicionadas en x (particularmente, la probabilidad de que $Y = k$, dado el vector predictor x). Entonces, el clasificador óptimo Bayesiano en x, es definido por:

$$f(X = x) = j \; si \; p_j(x) = \max\{p_1(x), p_2(x), \dots, p_K(x)\}$$

Se elige la clase que tiene mayor probabilidad para el valor del descriptor calculado.

Puede demostrarse que este clasificador minimiza el error medio en los aciertos.

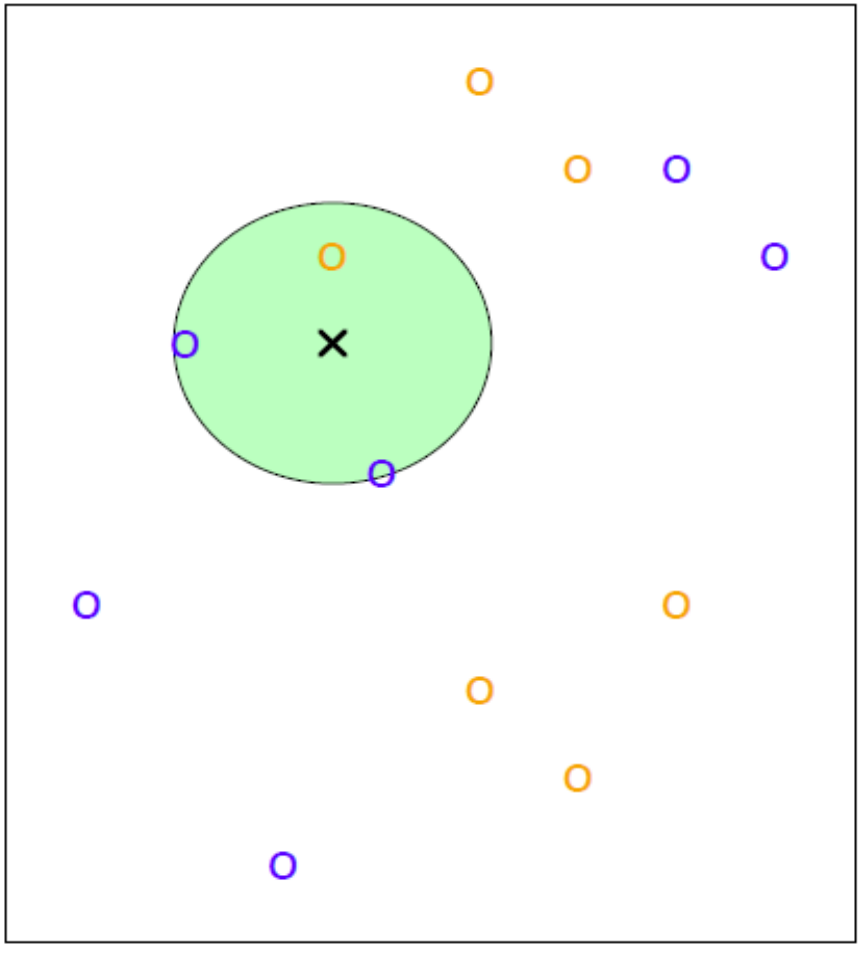
K-Nearest Neighbors (KNN)

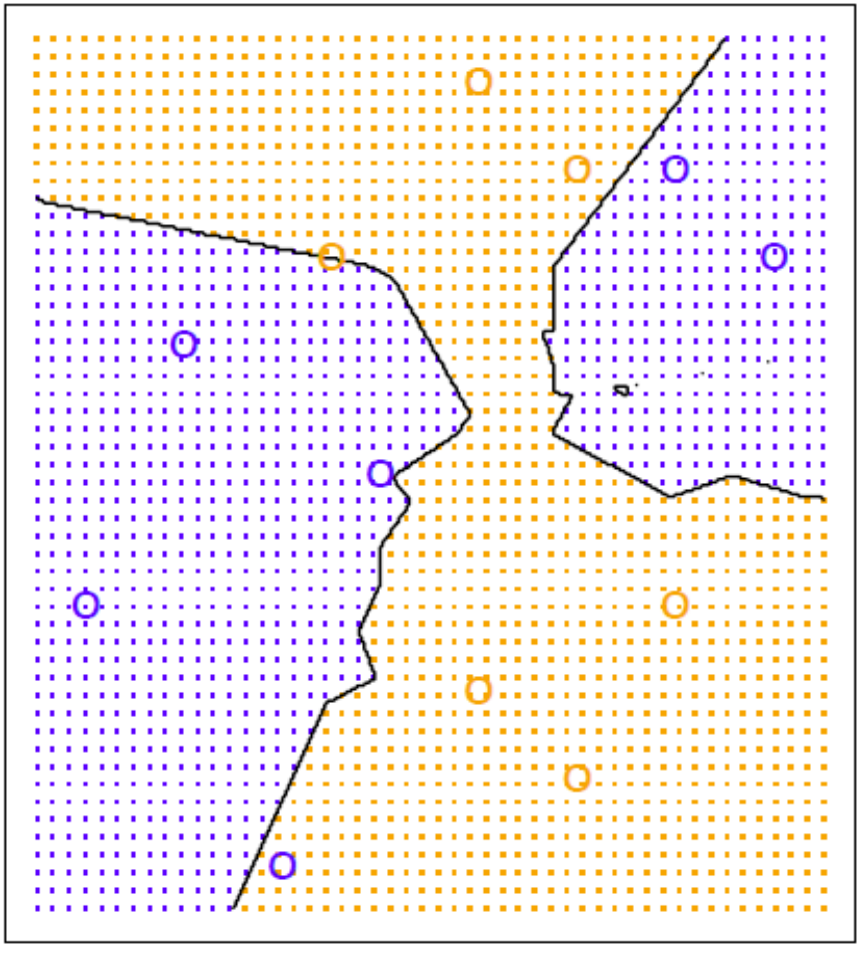
- El clasificador de Bayes es ideal (imposible de calcular) y sirve como un estándar de oro para comparar otros métodos.
- Un método para *estimar* la probabilidad condicional de *Y* dado *X*, es KNN.
- Dados un posible entero *K*, y una observación de *prueba* x_0:

$$\Pr(Y = j | X = x_0) = \frac{1}{K} \sum_{i \in \mathcal{N}_0} I(y_i = j)$$

Estima la probabilidad condicional para la clase *j* como la fracción de puntos en la vecindad $\mathcal{N}_0$ que pertenecen a la clase *j*.

Ejemplo de KNN con k = 3





¿Se puede usar regresión lineal para clasificar?

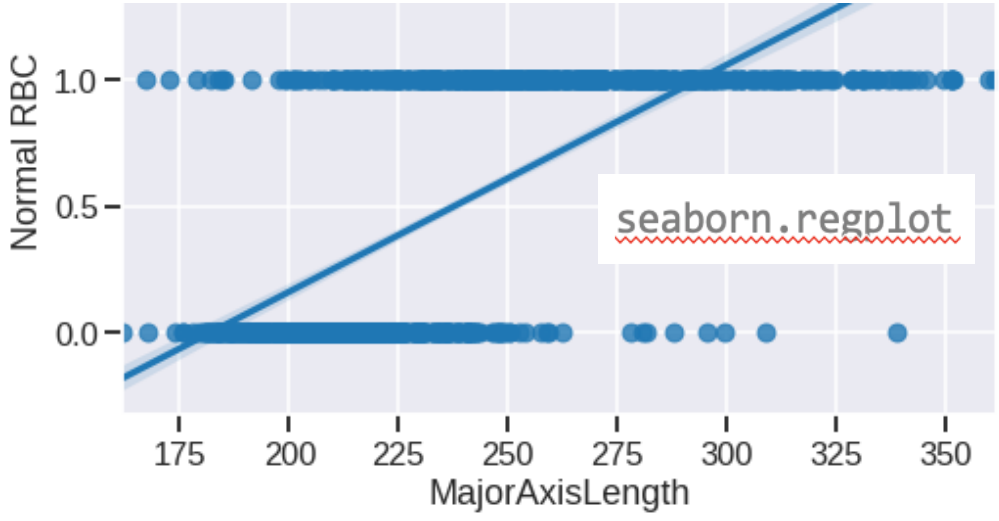
Caso de las anemias

Supongamos que la variable de salida tipo de RBC se codifica como:

$$Y = \begin{cases} 0 \quad si \;\; no\ es\ Normal\ (esferocito) \\ 1 \quad si \;\; Normal \end{cases}$$

Consideremos la probabilidad para la clase Normal: $p(X) = \Pr(Y = 1|X)$

Un modelo lineal sería $p(X) = \theta_0 + \theta_1 X$

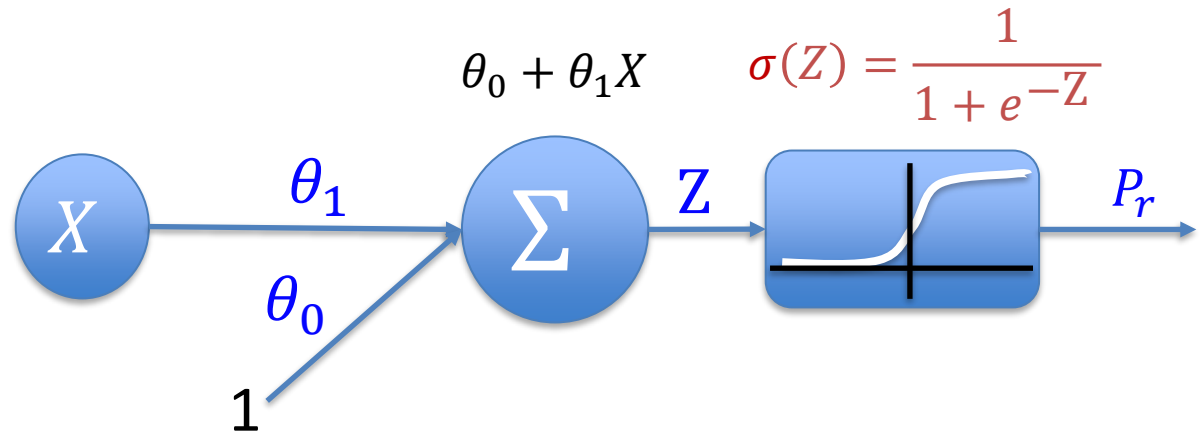


- Probabilidades negativas para longitudes pequeñas
- Probabilidades > 1 para longitudes grandes

Idea inviable

Regresión logística

$$P_r(Y = 1|X) = p(X) = \frac{e^{\theta_0+\theta_1 X}}{1 + e^{\theta_0+\theta_1 X}}$$



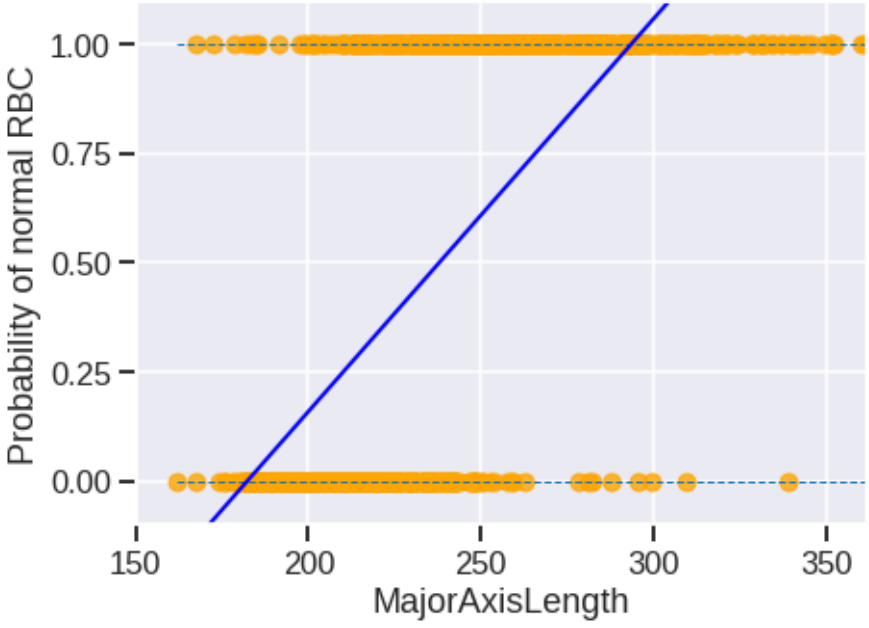
log-odds
logit

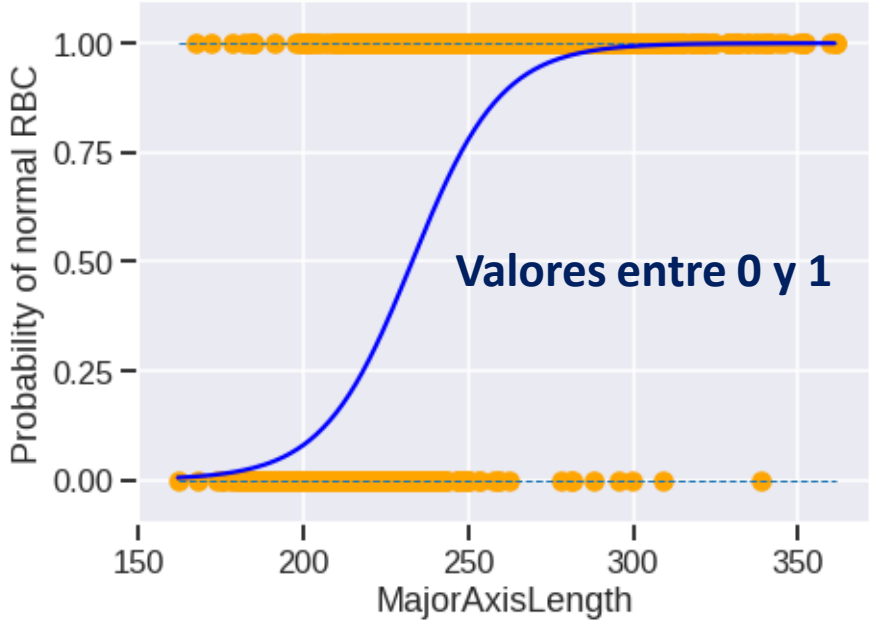
$$log\left(\frac{p(X)}{1-p(X)}\right) = \theta_0 + \theta_1 X$$

odss

Regresión Lineal versus Regresión Logística

$$p(x) = P_r(Y = 1|X)$$





Regresión lineal

$$p(X) = \theta_0 + \theta_1 X$$

Regresión logística

$$p(X) = \frac{e^{\theta_0+\theta_1 X}}{1 + e^{\theta_0+\theta_1 X}}$$

Interpretación de β_1

$$\underbrace{\frac{p(X)}{1-p(X)}}_{\text{Odds}} = e^{\theta_0+\theta_1 X} \qquad \underbrace{log\left(\frac{p(X)}{1-p(X)}\right)}_{\text{Logit}} = \theta_0 + \theta_1 X$$

- No es fácil interpretar lo que significa θ_1 con la regresión logística, porque estamos prediciendo $P(Y)$ y no *Y*.
- Si $\theta_1 = 0$, significa que no hay relación entre *Y* y *X*.
- Si $\theta_1 > 0$, significa que cuando X aumenta, también lo hace la probabilidad de que Y = 1.
- Si $\theta_1 < 0$, significa que cuando X aumenta, la probabilidad de que Y = 1 disminuye.
- Pero, ¿cuánto más grande o más pequeño? La probabilidad depende del valor particular de *X*.

Máxima verosimilitud (*likelihood*) para estimar parámetros

Se trata de **estimar** los parámetros (θ_0, θ_1)

tales que la probabilidad predicha para la clase Y (default) sea lo más próxima a 1 cuando el individuo observado tenga de verdad default y lo más próxima a 0 en caso contrario.

Con esta idea, definimos la **función de verosimilitud**

$$\mathcal{L}(\theta_0, \theta_1) = \prod_{i:y_i=1} p(x_i) \prod_{i:y_i=0} (1 - p(x_i))$$

Esta verosimilitud da la probabilidad de los unos y ceros observados en los datos como sucesos independientes.

Máxima verosimilitud (*likelihood*) para estimar parámetros

$$\mathcal{L}(\theta_0, \theta_1) = \prod_{i:y_i=1} p(x_i) \prod_{i:y_i=0} (1 - p(x_i))$$

- Se puede encontrar θ_0 y θ_1 **maximizando** la verosimilitud de los datos observados.
- Se convierte en un **problema de optimización** de una función objetivo.
- Existen diversos métodos iterativos de optimización, por ejemplo: el método de Newton, o el método del gradiente descendiente.

¿Cómo aplicar el gradiente descendiente?

$$\mathrm{J}(\boldsymbol{\theta}) = -\log L = -\sum_{i:y_i=1} \log p(x_i) - \sum_{i:y_i=1} \log(1 - p(x_i))$$

$$\mathrm{J}(\boldsymbol{\theta}) = -y_i \sum_i \log p(x_i) - (1 - y_i) \sum_i \log(1 - p(x_i))$$

$$\theta_j \leftarrow \theta_j - \eta \frac{\partial}{\partial \theta_j} \mathrm{J}(\boldsymbol{\theta})$$

Implementación de la regresión logística: sklearn

```
1  model = LogisticRegression()
2
3  X_train = df2[['MajorAxisLength']]
4
5  y = df2['etiquetas']
6
7  model.fit(X_train,y)
8
9  print("classes: {}\ncoefficientes: {}\nintercepto: {}".format(
10     model.classes_,model.coef_, model.intercept_))
```

```
classes: ['Esferocito' 'Normal']
coefficientes: [[0.07414429]]
intercepto: [-17.2769656]
```

Predicción con la regresión logística

¿Cuál es la estimación para la probabilidad de normal (un glóbulo rojo normal) para un glóbulo rojo con longitud del eje mayor de 200 pixeles?

$$\hat{p}(X) = \frac{e^{\hat{\theta}_0+\hat{\theta}_1 X}}{1+e^{\hat{\theta}_0+\hat{\theta}_1 X}} = \frac{e^{-17.2773+0.0742\times 200}}{1+e^{-17.2773+0.0742\times 200}} = 0.0796$$

```
1  model.predict_proba([[200]])
```

```
array([[0.92042297, 0.07957703]])
```

Esferocito Normal

Predicción con la regresión logística

¿Y con una longitud del eje mayor de 250 pixeles?

$$\hat{p}(X) = \frac{e^{\hat{\theta}_0+\hat{\theta}_1 X}}{1+e^{\hat{\theta}_0+\hat{\theta}_1 X}} = \frac{e^{-17.2773+0.0742\times 250}}{1+e^{-17.2773+0.0742\times 250}} = 0.7789$$

```
1 model.predict_proba([[250]])
```

```
array([[0.22112274, 0.77887726]])
```

Esferocito **Normal**

Descriptores cualitativos en la regresión logística

	etiquetas	Area_cualitativa	MajorAxisLength	Mean_Green
230	Esferocito	Low	220.269454	38532.624704
539	Esferocito	Low	212.155436	40150.730297
890	Normal	High	279.427778	45520.430587

Descriptor cualitativo = Area (Low, High)

Para ajustar el modelo de regresión logística se introduce la codificación: 0=Low, 1=High

```
1 model = LogisticRegression()
2 model.fit(df2[['Area_cualitativa2']],df2['etiquetas'])
```

	coeficientes
θ_0 **intercepto**	-1.227996
θ_1 **Area_cualitativa2**	3.207253

Descriptores cualitativos en la regresión logística

	coeficientes
intercepto	-1.227996
Area_cualitativa2	3.207253

Coeficiente positivo

Los glóbulos rojos con un área grande tienen mayor probabilidad de ser normales.

Los glóbulos rojos con un área pequeña tienen mayor probabilidad de ser esferocitos.

$$\widehat{Pr}(\text{etiqueta} = \text{Normal}|\text{area} = \text{high}) = \frac{e^{-1.2280+3.2073\times1}}{1+e^{-1.2280+3.2073\times1}}=0.8786$$

$$\widehat{Pr}(\text{etiqueta} = \text{Normal}|\text{area} = \text{Low}) = \frac{e^{-1.2280+3.2073\times0}}{1+e^{-1.2280+3.2073\times0}}=0.2265$$

Regresión logística con múltiples variables

$$\log\left(\frac{p(X)}{1-p(X)}\right) = \theta_0 + \theta_1 X_1 + \cdots + \theta_p X_p = \mathbf{x}^{\mathrm{T}}\boldsymbol{\theta}$$

$$p(X) = \frac{1}{1 + e^{-\mathbf{x}^{\mathrm{T}}\boldsymbol{\theta}}}$$

	coeficientes
intercepto	0.401577
MajorAxisLength	2.051337
Area_cualitativa2	-0.005281
Mean_Green	2.619206

¿Por qué ahora el coeficiente para el área (grande) es negativo, mientras antes fue positivo?

Regresión logística con más de dos clases

En muchos casos, se necesita clasificar en varias (más de dos) clases.

El modelo de regresión logística se puede extender a varias clases, modelizando la probabilidad de cada clase (k) en la forma

Regresión multinomial

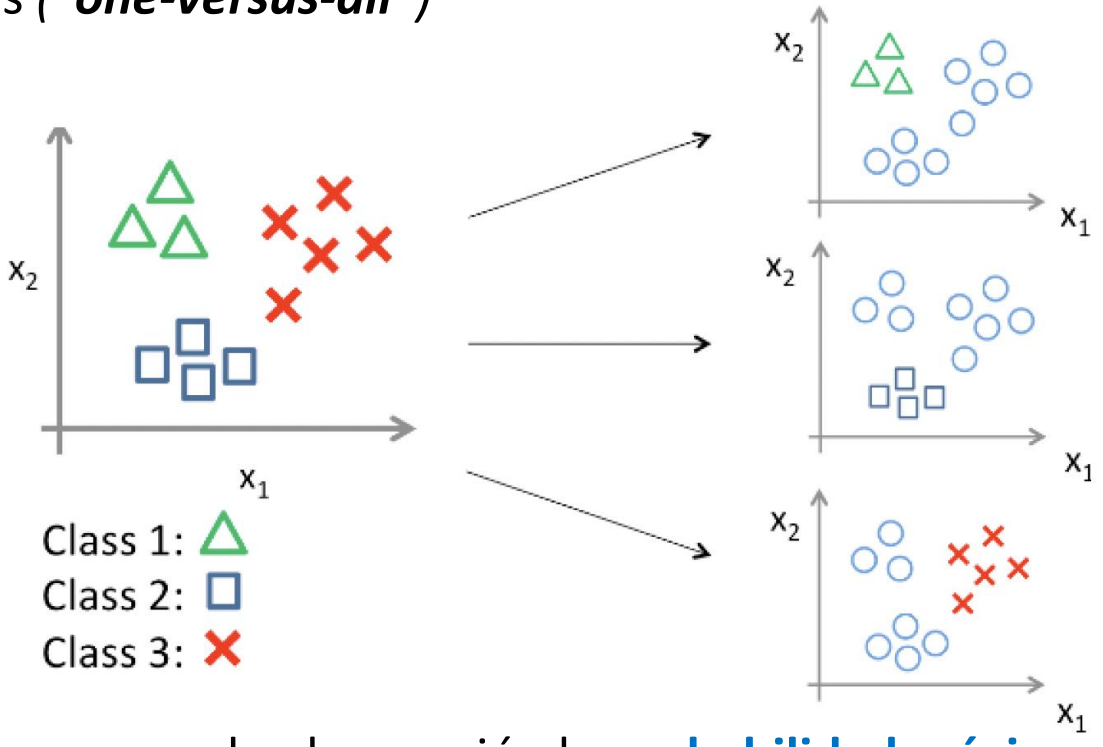
$$\Pr(Y = k|X) = \frac{e^{\beta_{0k}+\beta_{1k}X_1+\cdots+\beta_{pk}X_p}}{\sum_{\ell=1}^{K} e^{\beta_{0\ell}+\beta_{1\ell}X_1+\cdots+\beta_{p\ell}X_p}}$$

Sklearn lo implementa mediante el parámetro:

`multi_class = "multinomial"`

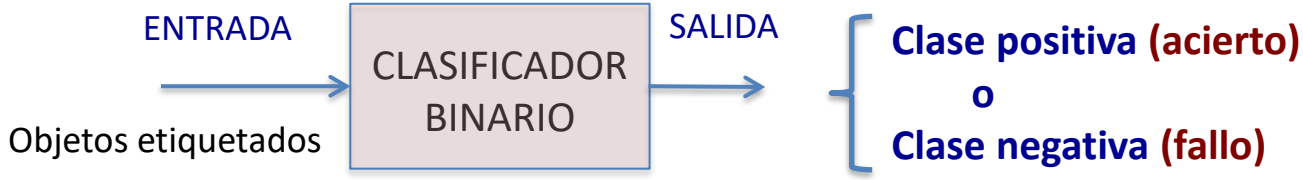
Regresión logística con más de dos clases

Otra posibilidad es plantear una clasificación binaria para cada clase frente al resto de clases ***("one-versus-all")***



- Se escoge para cada observación la **probabilidad máxima**.
- Es fácilmente interpretable.
- Sklearn lo implementa mediante el parámetro:
  - ✓ `multi_class = "ovr"`

Medidas del rendimiento de un clasificador



Matriz de confusión

Ejemplo de clasificación de 60000 imágenes MIST

Clasificación

Clases Verdaderas	CLASIFICACIÓN NEGATIVA	CLASIFICACIÓN POSITIVA
CLASE NEGATIVA	Verdadero Negativo VN = 53272	Falso Positivo FP = 1307
CLASE POSITIVA	Falso Negativo FN = 1077	Verdadero Positivo VP = 4344

Medidas del rendimiento de un clasificador

Exactitud (*accuracy*)

Es la proporción de aciertos:

$$exactitud = \frac{VN + VP}{total\ de\ objetos} = \frac{53272 + 4344}{60000} = 0.96 = 96\%$$

Precisión (valor predictivo positivo)

Es la exactitud de las predicciones positivas. Es decir, la proporción de verdaderos positivos (aciertos) sobre el total de objetos que han sido clasificados como positivos, esto es

$$Precisión = \frac{VP}{VP + FP} = \frac{4344}{4344 + 1307} = 0.77 = 77\%$$

Sensibilidad (Proporción de verdaderos positivos - PVP) o Recall

Es la proporción de objetos positivos que son clasificados correctamente. Es decir, la proporción de aciertos dentro del conjunto total de objetos que son realmente positivos:

$$Sensibilidad = PVP = \frac{VP}{VP + FN} = \frac{4344}{4344 + 1077} = 0.79 = 79\%$$

Medidas del rendimiento de un clasificador

Especificidad (Proporción de verdaderos negativos - PVN)

Es la proporción de objetos negativos que son clasificados correctamente. Es decir, la proporción de aciertos dentro del conjunto total de objetos que son realmente negativos:

$$Especificidad = PVN = \frac{VN}{VN + FP} = \frac{53272}{53272 + 1307} = 0.98 = 98\%$$

Proporción de falsos positivos (PFP)

Es la proporción de objetos negativos que son clasificados incorrectamente, es decir como si fueran positivos. Este valor es complementario con la especificidad y se calcula en la forma

$$PFP = 1 - Especificidad = \frac{FP}{VN + FP} = \frac{1307}{53272 + 1307} = 0.02 = 2\%$$