

TICS-411 Minería de Datos

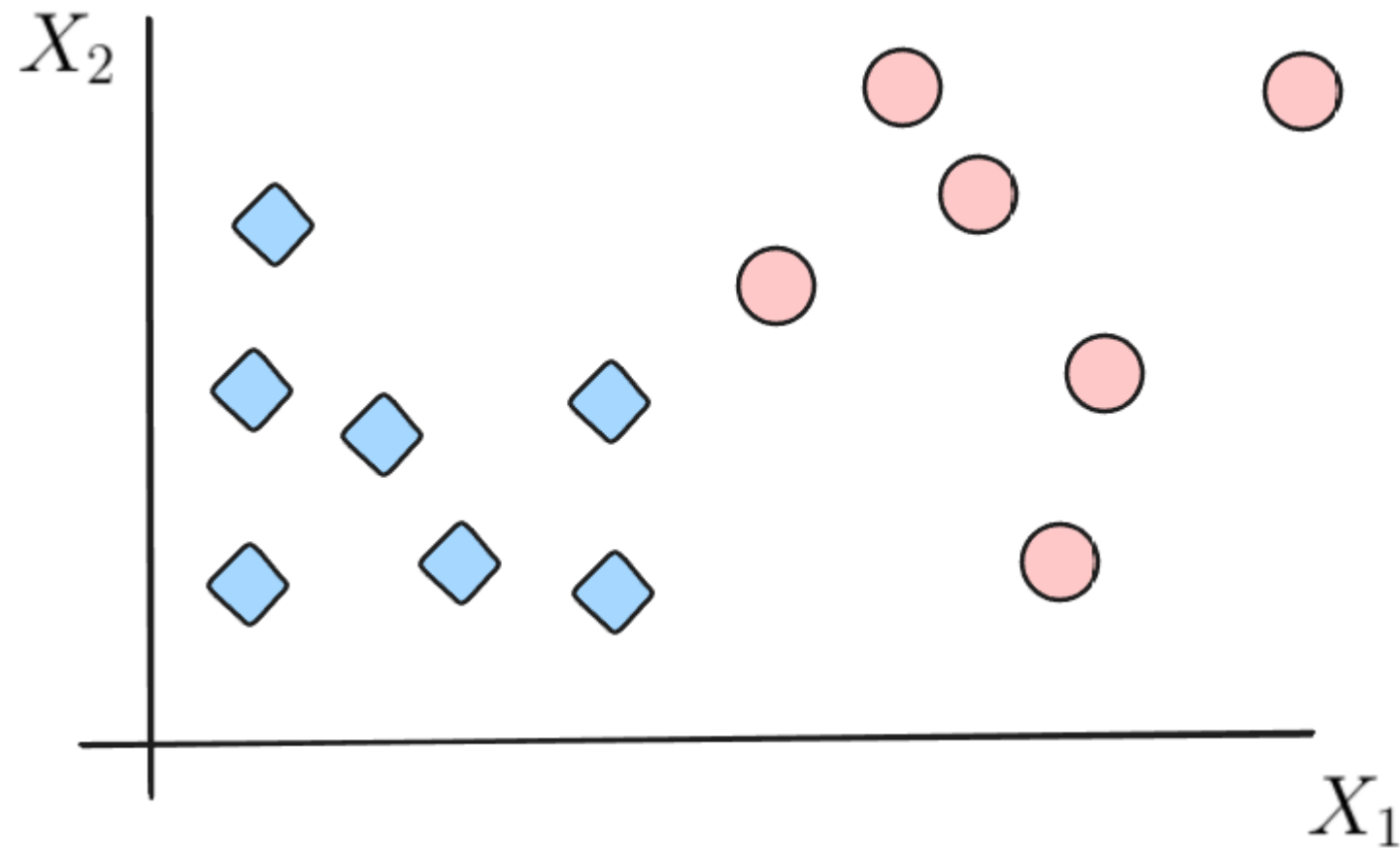
Clase 12: Regresión Logística

Alfonso Tobar-Arancibia

alfonso.tobar.a@edu.uai.cl

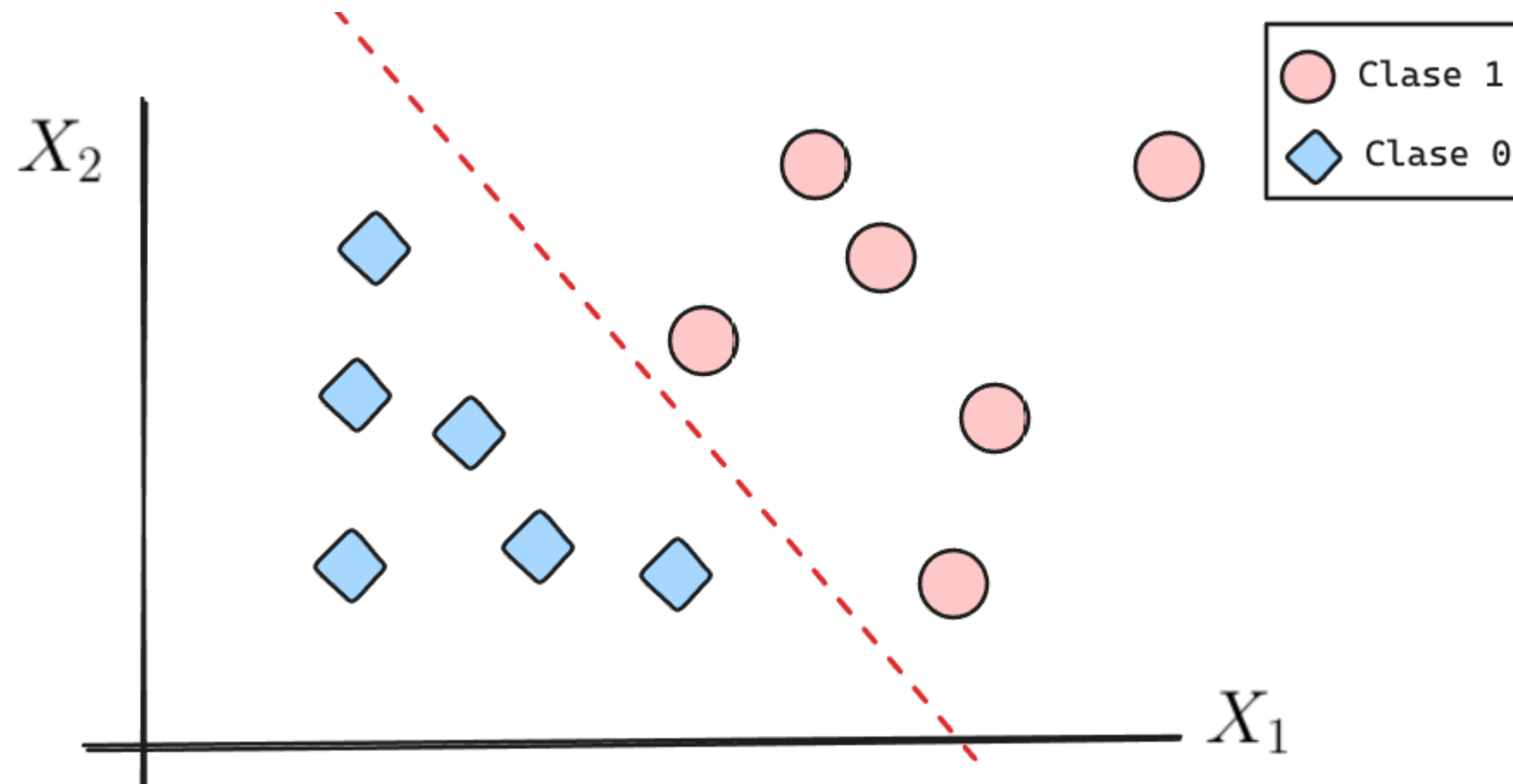
Intuición

Supongamos el siguiente dataset:

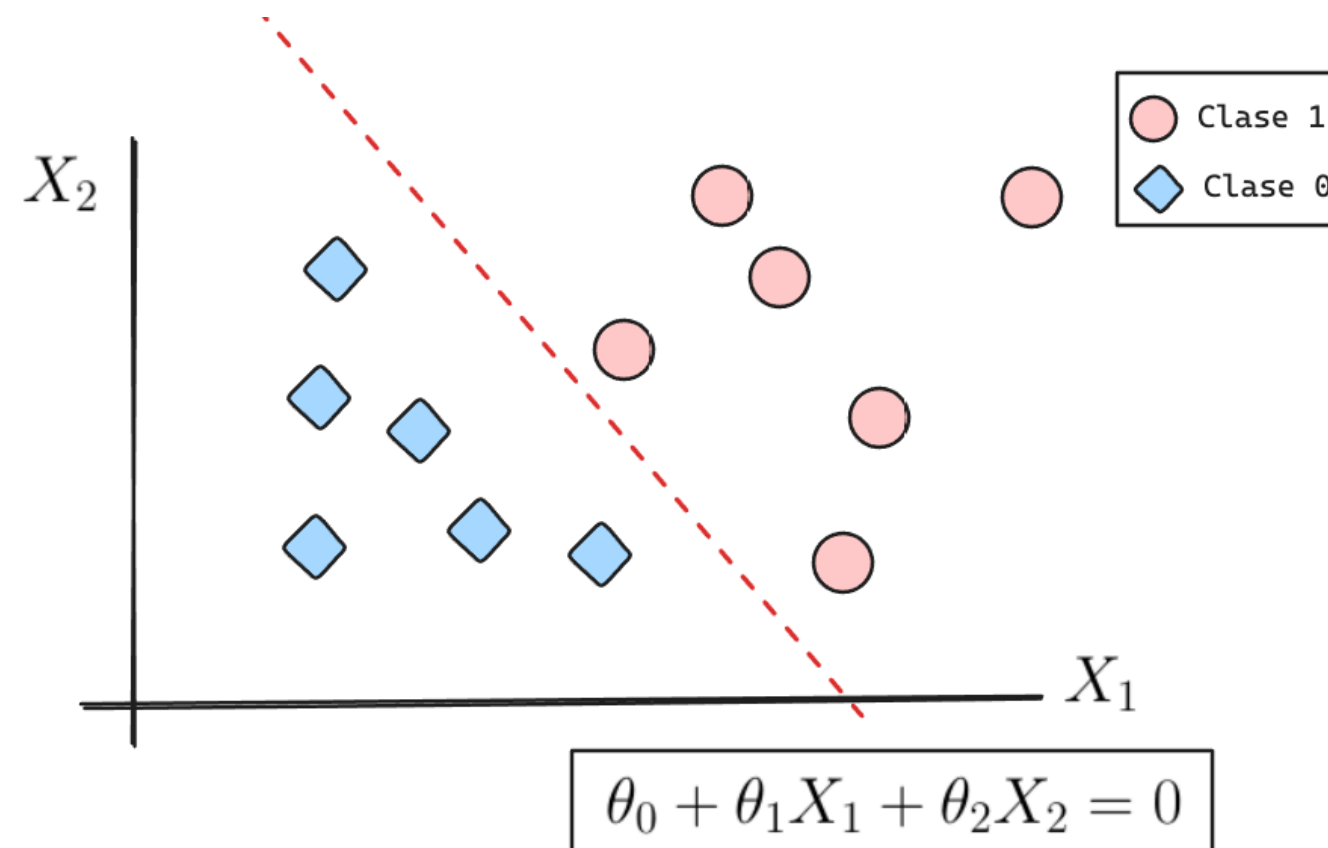


⚠ ¿Cómo puedo separar ambas clases?

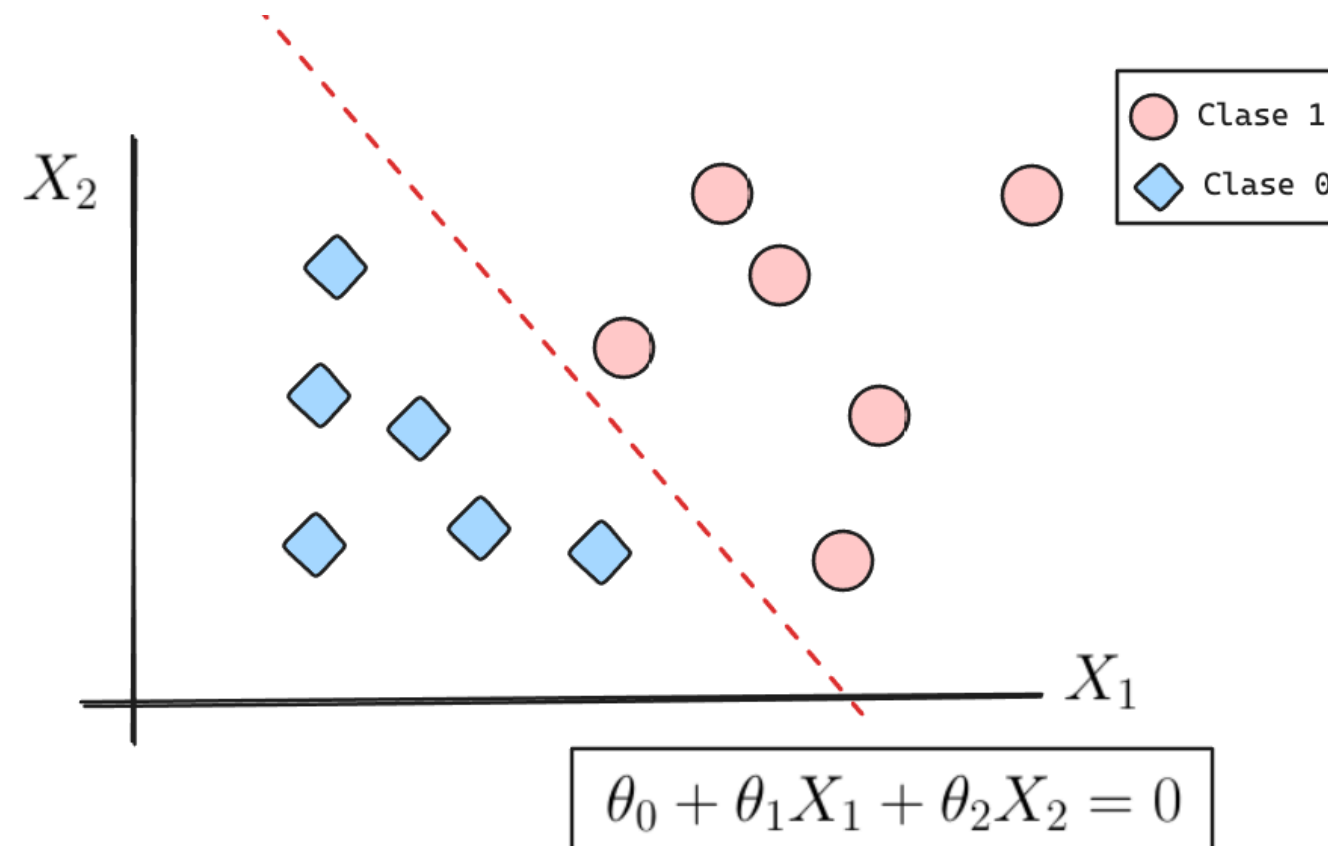
Intuición



Intuición

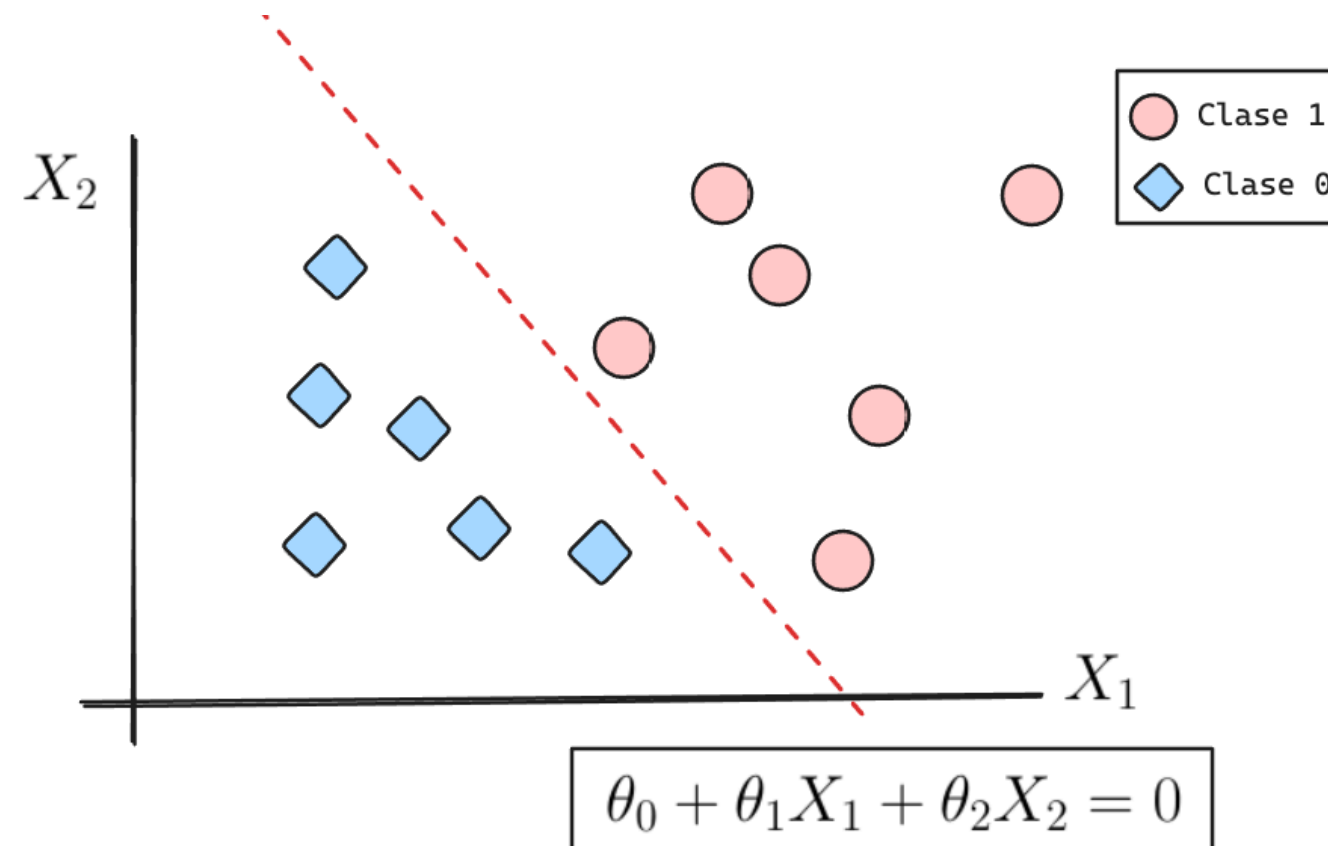


Intuición



i La **frontera de decisión** se puede caracterizar como la **ecuación de una recta** (en forma general).

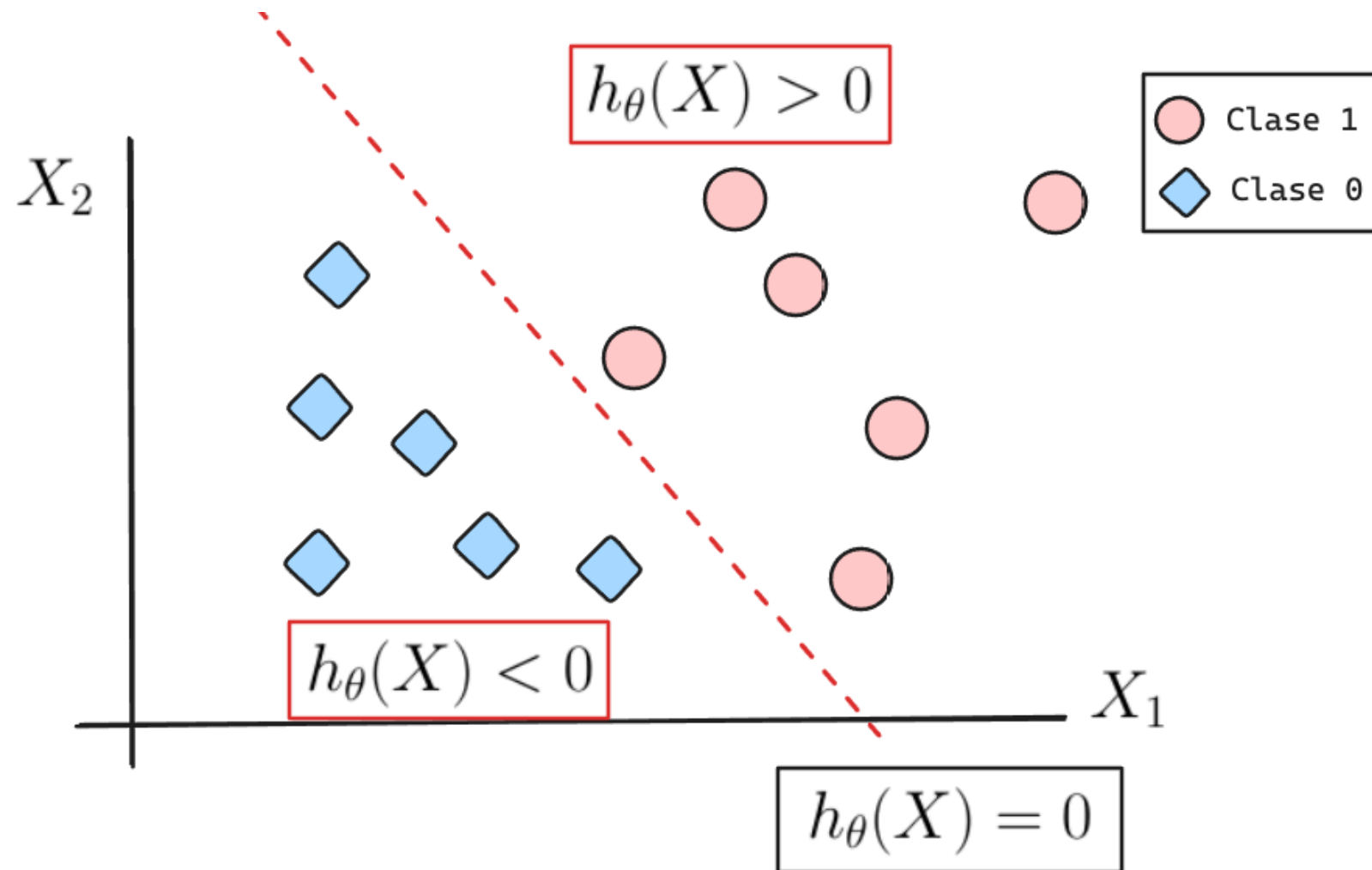
Intuición



❗ La **frontera de decisión** se puede caracterizar como la **ecuación de una recta** (en forma general).

💡 Además definiremos $h_{\theta}(X) = \theta_0 + \theta_1 X_1 + \theta_2 X_2$.

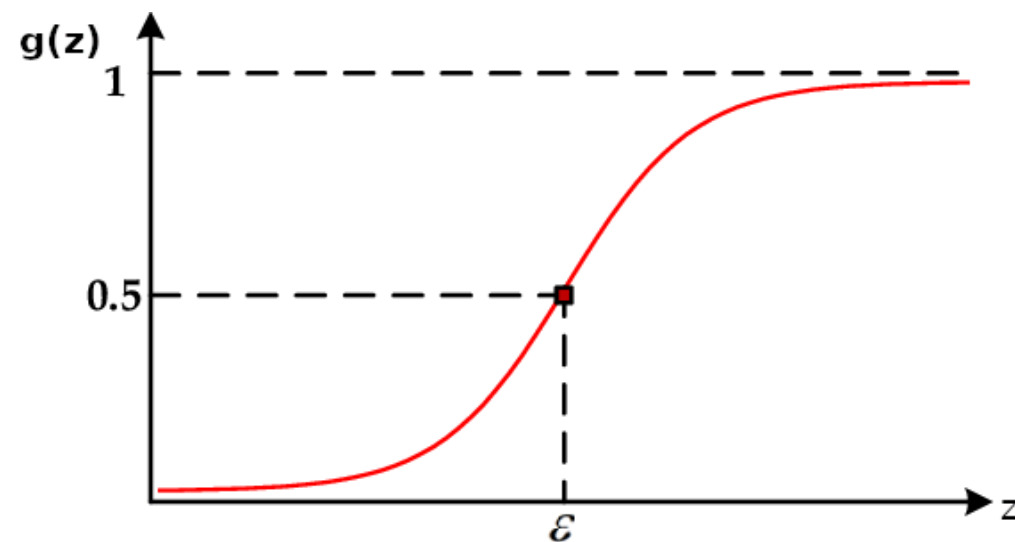
Intuición



❗ Podríamos pensar que si $h_\theta(X)$ es positivo entonces pertenece a la clase 1 y si $h_\theta(X)$ es negativo pertenece a la clase 0.

La Función Sigmoide o Logística

$$g(z) = \frac{1}{1 + e^{-z}}$$

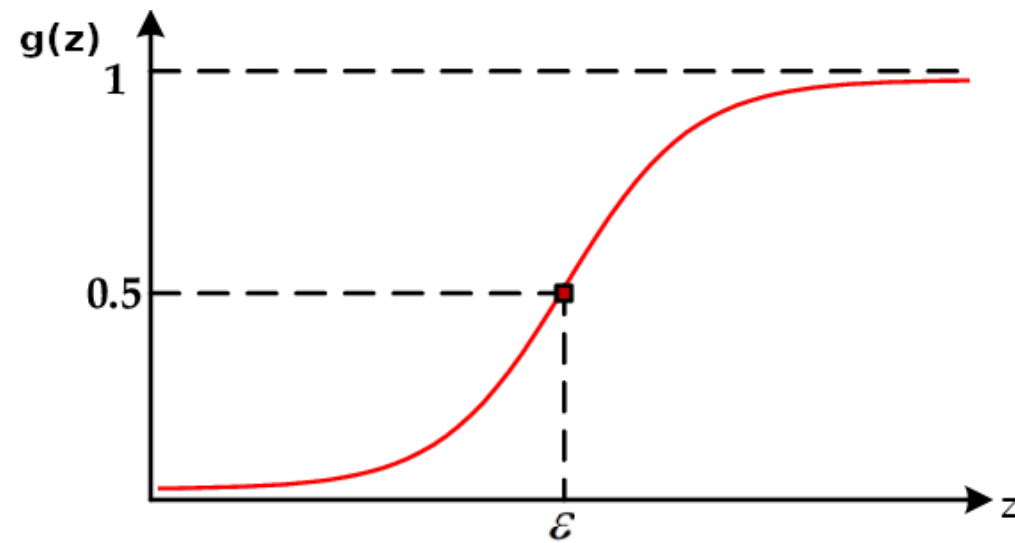


- Función no lineal.
- Función acotada entre 0 y 1.
- $g(\varepsilon) = 0.5, \varepsilon = 0$

i ¿Qué pasaría si ahora decimos que $z = \beta_0 + \beta_1 X_1 + \beta_2 X_2$?

La Función Sigmoide o Logística

$$g(z) = \frac{1}{1 + e^{-z}}$$



- Función no lineal.
- Función acotada entre 0 y 1.
- $g(\varepsilon) = 0.5, \varepsilon = 0$



De acá sale la noción del umbral 0.5 que hemos visto en clases anteriores.

i ¿Qué pasaría si ahora decimos que $z = \beta_0 + \beta_1 X_1 + \beta_2 X_2$?

La Regresión Logística

$$P[y = 1|X, \theta] = g(\theta_0 + \theta_1 X_1 + \theta_2 X_2) = \frac{1}{1 + e^{-(\theta_0 + \theta_1 X_1 + \theta_2 X_2)}}$$

La Regresión Logística

$$P[y = 1|X, \theta] = g(\theta_0 + \theta_1 X_1 + \theta_2 X_2) = \frac{1}{1 + e^{-(\theta_0 + \theta_1 X_1 + \theta_2 X_2)}}$$

i Regla de Decisión:

- Si $g(z) \geq 0.5 \implies \text{Clase 1.}$
- Si $g(z) < 0.5 \implies \text{Clase 0.}$

La Regresión Logística

$$P[y = 1|X, \theta] = g(\theta_0 + \theta_1 X_1 + \theta_2 X_2) = \frac{1}{1 + e^{-(\theta_0 + \theta_1 X_1 + \theta_2 X_2)}}$$

i Regla de Decisión:

- Si $g(z) \geq 0.5 \implies \text{Clase 1.}$
- Si $g(z) < 0.5 \implies \text{Clase 0.}$

! $g(z)$ se puede interpretar como una **probabilidad de pertenecer a la Clase 1.**

La Regresión Logística

$$P[y = 1 | X, \theta] = g(\theta_0 + \theta_1 X_1 + \theta_2 X_2) = \frac{1}{1 + e^{-(\theta_0 + \theta_1 X_1 + \theta_2 X_2)}}$$

i Regla de Decisión:

- Si $g(z) \geq 0.5 \implies \text{Clase 1}$.
- Si $g(z) < 0.5 \implies \text{Clase 0}$.

! $g(z)$ se puede interpretar como una **probabilidad de pertenecer a la Clase 1**.

! $1 - g(z)$ se puede interpretar como una **probabilidad de NO pertenecer a la Clase 1**, es decir, **pertenecer a la Clase 0**.

Aprendizaje del Modelo

Supongamos lo siguiente:

$$P(y = 1|X, \theta) = h_{\theta}(X)$$

$$P(y = 0|X, \theta) = 1 - h_{\theta}(x)$$

Ambas ecuaciones pueden comprimirse en una sola de la siguiente manera:

$$P(y|X, \theta) = h_{\theta}(X)^y (1 - h_{\theta}(X))^{1-y}$$

⚠ Para encontrar los parámetros θ podemos utilizar una técnica llamada **Maximum Likelihood Estimation**.

Maximum Likelihood Estimation

$$\mathcal{L}(\theta) = \prod_{i=1}^n P(y^{(i)} | x^{(i)}, \theta)$$

$$\underset{\theta}{\operatorname{argmin}} -l(\theta)$$

$$l(\theta) = \log(\mathcal{L}(\theta)) = \sum_{i=1}^n y^{(i)} \cdot \log(h_{\theta}(x^{(i)})) + (1 - y^{(i)}) \cdot \log(1 - h_{\theta}(x^{(i)}))$$

💡 Esta ecuación se conoce como **Entropía Cruzada** o como **Negative Log Loss (NLL)** y tiene la gracia de que es una curva convexa lo que *garantiza un valor único de los parámetros θ* .

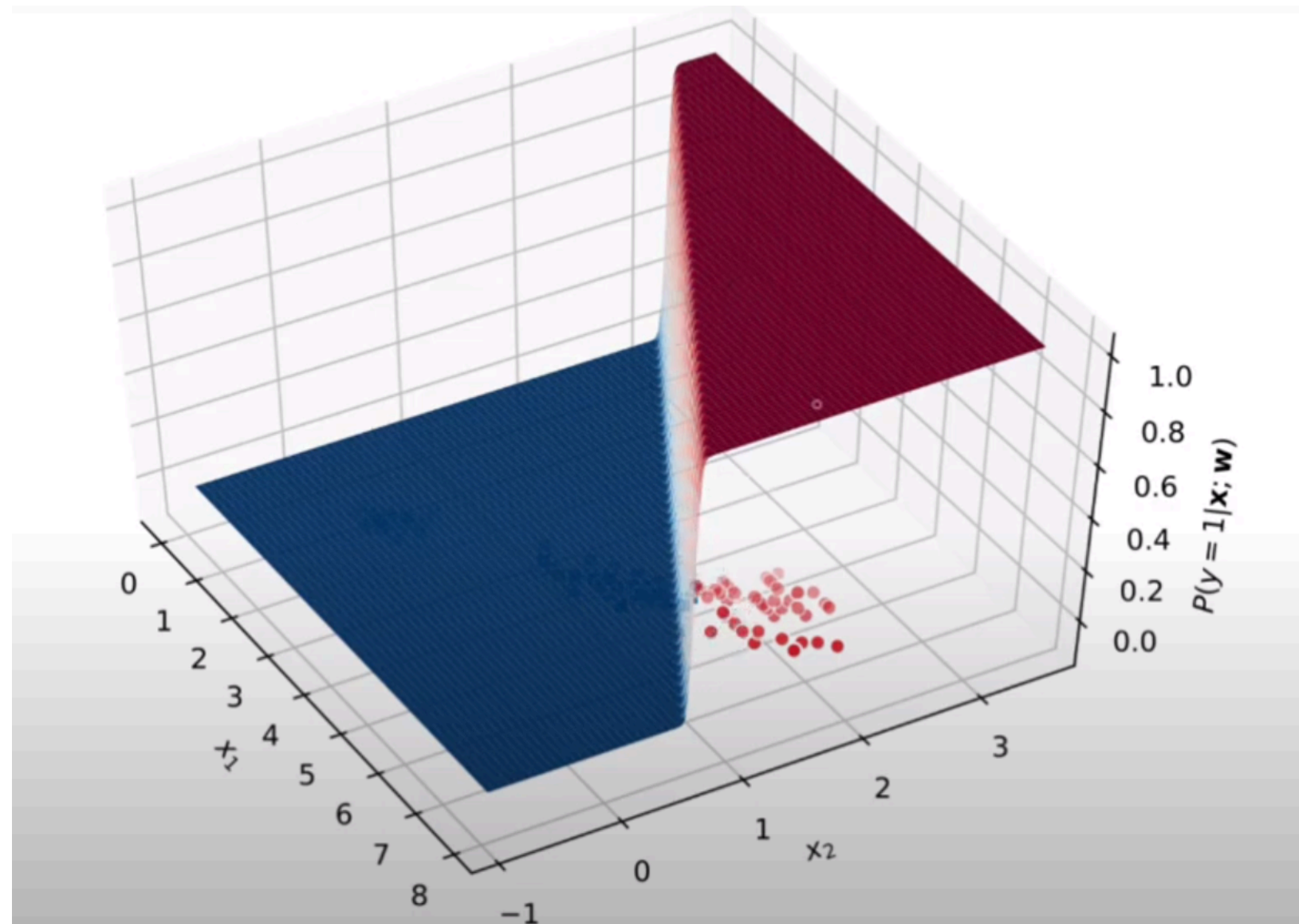
Cálculo de Coeficientes

⚠ La técnica más famosa para minimizar este tipo de problemas se conoce como **Stochastic Gradient Descent**. Lo que genera la siguiente solución:

$$\theta_j \leftarrow \theta_j - \alpha \frac{1}{m} \sum_{i=1} \left(h_{\theta}(x^{(i)}) - y^{(i)} \right) x_j^{(i)}$$

⚠ A pesar de lo complicado que se ve la ecuación, implementarla en código es bastante sencillo.

Frontera de Decisión

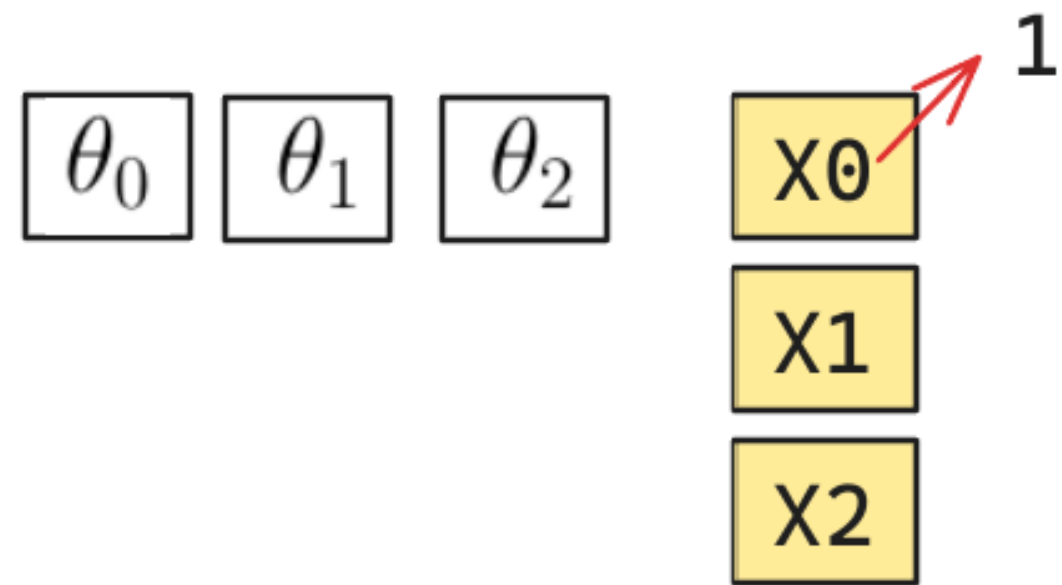


Inference Time

En este caso se calcula:

$$h_{\theta}(x^{(i)}) = \textit{sigmoid}(\theta^t x^{(i)})$$

- θ : Corresponde a un vector con todos los parámetros calculados.
- $x^{(i)}$: Corresponde a una instancia de m variables la cual generará una probabilidad.
 - $\theta^t x^{(i)}$ corresponde al producto punto de dos vectores, que es equivalente a una “**suma producto**”.
- $h_{\theta}(x^{(i)})$: Generará un valor entre 0 y 1 al cuál se le aplica la Regla de Decisión.



Implementación en Python

```
1 from sklearn.linear_model import LogisticRegression
2
3 lr = LogisticRegression(C=1, penalty="l2", random_state = 42)
4 lr.fit(X_train, y_train)
5
6 y_pred = lr.predict(X_test)
7 y_proba = lr.predict_proba(X_test)
8
9 ## Visualizacion de los Parámetros
10 lr.coef_
11 lr.intercept_
```

- **C**: Corresponde a un parámetro de Regularización. Valores más pequeños implica mayor regularización. Por defecto 1.
- **penalty**: Corresponde al tipo de regularización. Por defecto "l2".
 - **"l1"**: Corresponde a la regularización Lasso. Genera que hayan parámetros cero, ayudando en la selección de variables.
 - **"l2"**: Corresponde a la regularización Ridge. Genera que todos los parámetros sean pequeños, entregando estabilidad y buena interpretabilidad.
 - **"elasticnet"**: Corresponde a la combinación de "l1" y "l2".
 - **None**: No hay regularización.

❗ Para cambiar la regularización, consultar la documentación de Scikit-Learn.

Implementación en Python

```
1 from sklearn.linear_model import LogisticRegression
2
3 lr = LogisticRegression(C=1, penalty="l2", random_state = 42)
4 lr.fit(X_train, y_train)
5
6 y_pred = lr.predict(X_test)
7 y_proba = lr.predict_proba(X_test)
8
9 ## Visualizacion de los Parámetros
10 lr.coef_
11 lr.intercept_
```

- **C**: Corresponde a un parámetro de Regularización. Valores más pequeños implica mayor regularización. Por defecto 1.
- **penalty**: Corresponde al tipo de regularización. Por defecto "l2".
 - **"l1"**: Corresponde a la regularización Lasso. Genera que hayan parámetros cero, ayudando en la selección de variables.
 - **"l2"**: Corresponde a la regularización Ridge. Genera que todos los parámetros sean pequeños, entregando estabilidad y buena interpretabilidad.
 - **"elasticnet"**: Corresponde a la combinación de "l1" y "l2".
 - **None**: No hay regularización.

❗ Para cambiar la regularización, consultar la documentación de Scikit-Learn.

Implementación en Python

```
1 from sklearn.linear_model import LogisticRegression
2
3 lr = LogisticRegression(C=1, penalty="l2", random_state = 42)
4 lr.fit(X_train, y_train)
5
6 y_pred = lr.predict(X_test)
7 y_proba = lr.predict_proba(X_test)
8
9 ## Visualizacion de los Parámetros
10 lr.coef_
11 lr.intercept_
```

- **C**: Corresponde a un parámetro de Regularización. Valores más pequeños implica mayor regularización. Por defecto 1.
- **penalty**: Corresponde al tipo de regularización. Por defecto "l2".
 - **"l1"**: Corresponde a la regularización Lasso. Genera que hayan parámetros cero, ayudando en la selección de variables.
 - **"l2"**: Corresponde a la regularización Ridge. Genera que todos los parámetros sean pequeños, entregando estabilidad y buena interpretabilidad.
 - **"elasticnet"**: Corresponde a la combinación de "l1" y "l2".
 - **None**: No hay regularización.

❗ Para cambiar la regularización, consultar la documentación de Scikit-Learn.

Implementación en Python

```
1 from sklearn.linear_model import LogisticRegression
2
3 lr = LogisticRegression(C=1, penalty="l2", random_state = 42)
4 lr.fit(X_train, y_train)
5
6 y_pred = lr.predict(X_test)
7 y_proba = lr.predict_proba(X_test)
8
9 ## Visualizacion de los Parámetros
10 lr.coef_
11 lr.intercept_
```

- **C**: Corresponde a un parámetro de Regularización. Valores más pequeños implica mayor regularización. Por defecto 1.
- **penalty**: Corresponde al tipo de regularización. Por defecto "l2".
 - **"l1"**: Corresponde a la regularización Lasso. Genera que hayan parámetros cero, ayudando en la selección de variables.
 - **"l2"**: Corresponde a la regularización Ridge. Genera que todos los parámetros sean pequeños, entregando estabilidad y buena interpretabilidad.
 - **"elasticnet"**: Corresponde a la combinación de "l1" y "l2".
 - **None**: No hay regularización.

❗ Para cambiar la regularización, consultar la documentación de Scikit-Learn.

Implementación en Python

```
1 from sklearn.linear_model import LogisticRegression
2
3 lr = LogisticRegression(C=1, penalty="l2", random_state = 42)
4 lr.fit(X_train, y_train)
5
6 y_pred = lr.predict(X_test)
7 y_proba = lr.predict_proba(X_test)
8
9 ## Visualizacion de los Parámetros
10 lr.coef_
11 lr.intercept_
```

- **C**: Corresponde a un parámetro de Regularización. Valores más pequeños implica mayor regularización. Por defecto 1.
- **penalty**: Corresponde al tipo de regularización. Por defecto "l2".
 - **"l1"**: Corresponde a la regularización Lasso. Genera que hayan parámetros cero, ayudando en la selección de variables.
 - **"l2"**: Corresponde a la regularización Ridge. Genera que todos los parámetros sean pequeños, entregando estabilidad y buena interpretabilidad.
 - **"elasticnet"**: Corresponde a la combinación de "l1" y "l2".
 - **None**: No hay regularización.

❗ Para cambiar la regularización, consultar la documentación de Scikit-Learn.

Implementación en Python

```
1 from sklearn.linear_model import LogisticRegression
2
3 lr = LogisticRegression(C=1, penalty="l2", random_state = 42)
4 lr.fit(X_train, y_train)
5
6 y_pred = lr.predict(X_test)
7 y_proba = lr.predict_proba(X_test)
8
9 ## Visualizacion de los Parámetros
10 lr.coef_
11 lr.intercept_
```

- **C**: Corresponde a un parámetro de Regularización. Valores más pequeños implica mayor regularización. Por defecto 1.
- **penalty**: Corresponde al tipo de regularización. Por defecto "l2".
 - **"l1"**: Corresponde a la regularización Lasso. Genera que hayan parámetros cero, ayudando en la selección de variables.
 - **"l2"**: Corresponde a la regularización Ridge. Genera que todos los parámetros sean pequeños, entregando estabilidad y buena interpretabilidad.
 - **"elasticnet"**: Corresponde a la combinación de "l1" y "l2".
 - **None**: No hay regularización.

❗ Para cambiar la regularización, consultar la documentación de Scikit-Learn.

Interpretabilidad

Una de las grandes ventajas que tiene la Regresión Logística es que sus predicciones son interpretables.

- Tenemos un dataset de 2 variables:
 - W : Corresponde al peso del Vehículo.
 - $qsec$: Corresponde al tiempo en Segundos que lo toma en recorrer un cuarto de milla.
- Queremos predecir si el vehículo es Económico o no (en términos de consumo de Bencina).

$$h_{\theta}(x) = 0.5 - 3.5 \cdot W + 1.5 \cdot qsec$$

Interpretabilidad

Una de las grandes ventajas que tiene la Regresión Logística es que sus predicciones son interpretables.

- Tenemos un dataset de 2 variables:
 - W : Corresponde al peso del Vehículo.
 - $qsec$: Corresponde al tiempo en Segundos que lo toma en recorrer un cuarto de milla.
- Queremos predecir si el vehículo es Económico o no (en términos de consumo de Bencina).

$$h_{\theta}(x) = 0.5 - 3.5 \cdot W + 1.5 \cdot qsec$$



Interpretabilidad

Una de las grandes ventajas que tiene la Regresión Logística es que sus predicciones son interpretables.

- Tenemos un dataset de 2 variables:
 - W : Corresponde al peso del Vehículo.
 - $qsec$: Corresponde al tiempo en Segundos que lo toma en recorrer un cuarto de milla.
- Queremos predecir si el vehículo es Económico o no (en términos de consumo de Bencina).

$$h_{\theta}(x) = 0.5 - 3.5 \cdot W + 1.5 \cdot qsec$$



- Si el vehículo se demora más en el cuarto de milla ($qsec$ aumenta) entonces el vehículo es más económico.

Interpretabilidad

Una de las grandes ventajas que tiene la Regresión Logística es que sus predicciones son interpretables.

- Tenemos un dataset de 2 variables:
 - W : Corresponde al peso del Vehículo.
 - $qsec$: Corresponde al tiempo en Segundos que lo toma en recorrer un cuarto de milla.
- Queremos predecir si el vehículo es Económico o no (en términos de consumo de Bencina).

$$h_{\theta}(x) = 0.5 - 3.5 \cdot W + 1.5 \cdot qsec$$



- Si el vehículo se demora más en el cuarto de milla ($qsec$ aumenta) entonces el vehículo es más económico.
 - Tiene menos potencia.

Interpretabilidad

Una de las grandes ventajas que tiene la Regresión Logística es que sus predicciones son interpretables.

- Tenemos un dataset de 2 variables:
 - W : Corresponde al peso del Vehículo.
 - $qsec$: Corresponde al tiempo en Segundos que lo toma en recorrer un cuarto de milla.
- Queremos predecir si el vehículo es Económico o no (en términos de consumo de Bencina).

$$h_{\theta}(x) = 0.5 - 3.5 \cdot W + 1.5 \cdot qsec$$



- Si el vehículo se demora más en el cuarto de milla ($qsec$ aumenta) entonces el vehículo es más económico.
 - Tiene menos potencia.



Interpretabilidad

Una de las grandes ventajas que tiene la Regresión Logística es que sus predicciones son interpretables.

- Tenemos un dataset de 2 variables:
 - W : Corresponde al peso del Vehículo.
 - $qsec$: Corresponde al tiempo en Segundos que lo toma en recorrer un cuarto de milla.
- Queremos predecir si el vehículo es Económico o no (en términos de consumo de Bencina).

$$h_{\theta}(x) = 0.5 - 3.5 \cdot W + 1.5 \cdot qsec$$



- Si el vehículo se demora más en el cuarto de milla ($qsec$ aumenta) entonces el vehículo es más económico.
 - Tiene menos potencia.



- Si el vehículo es más pesado (W aumenta), entonces es menos económico.

Interpretabilidad

Una de las grandes ventajas que tiene la Regresión Logística es que sus predicciones son interpretables.

- Tenemos un dataset de 2 variables:
 - W : Corresponde al peso del Vehículo.
 - $qsec$: Corresponde al tiempo en Segundos que lo toma en recorrer un cuarto de milla.
- Queremos predecir si el vehículo es Económico o no (en términos de consumo de Bencina).

$$h_{\theta}(x) = 0.5 - 3.5 \cdot W + 1.5 \cdot qsec$$



- Si el vehículo se demora más en el cuarto de milla ($qsec$ aumenta) entonces el vehículo es más económico.
 - Tiene menos potencia.



- Si el vehículo es más pesado (W aumenta), entonces es menos económico.
 - Requiere probablemente más combustible para mover dicho peso.

Interpretabilidad

Una de las grandes ventajas que tiene la Regresión Logística es que sus predicciones son interpretables.

- Tenemos un dataset de 2 variables:
 - W : Corresponde al peso del Vehículo.
 - $qsec$: Corresponde al tiempo en Segundos que lo toma en recorrer un cuarto de milla.
- Queremos predecir si el vehículo es Económico o no (en términos de consumo de Bencina).

$$h_{\theta}(x) = 0.5 - 3.5 \cdot W + 1.5 \cdot qsec$$



- Si el vehículo se demora más en el cuarto de milla ($qsec$ aumenta) entonces el vehículo es más económico.
 - Tiene menos potencia.



- Si el vehículo es más pesado (W aumenta), entonces es menos económico.
 - Requiere probablemente más combustible para mover dicho peso.



El valor del parámetro representa también la magnitud de la contribución.

Sugerencias

- 💡 • **Estandarización/Normalización de datos:** Permite que la escala de los datos no afecte en la interpretabilidad.
- **One Hot Encoder:** En general tiende a dar mejores resultados que el Ordinal.
- **Interacciones:** Combinación de variables.
- **Variables no Lineales:** Permite que la frontera de Decisión no sea necesariamente lineal (Regresión Polinómica).

мы сделали