Introducción a la Ciencia de Datos

ENTRENAR Y TESTEAR

1

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test= train_test_split(X,y,test_size = 0.15,random_state=1)
```

2

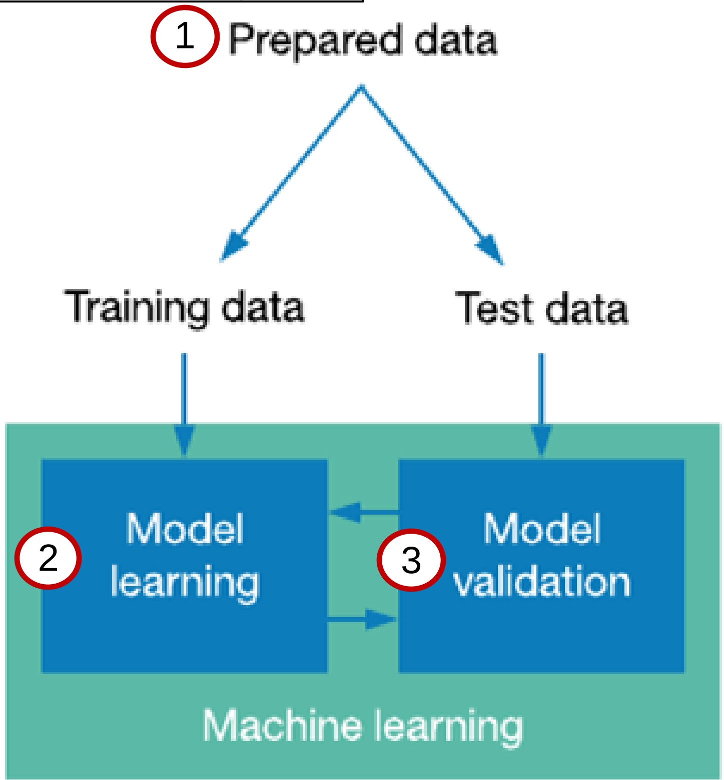
```
logreg.fit(X_train, y_train)
```

3

```
logreg_pred = logreg.predict(X_test)

from sklearn.metrics import accuracy_score
accuracy_score(y_test,logreg_pred)
```

Recuerda que los datos del **entrenamiento** deben tener las **mismas características** que los datos del **test:** número de variables predictoras, tipo de dato, etc.

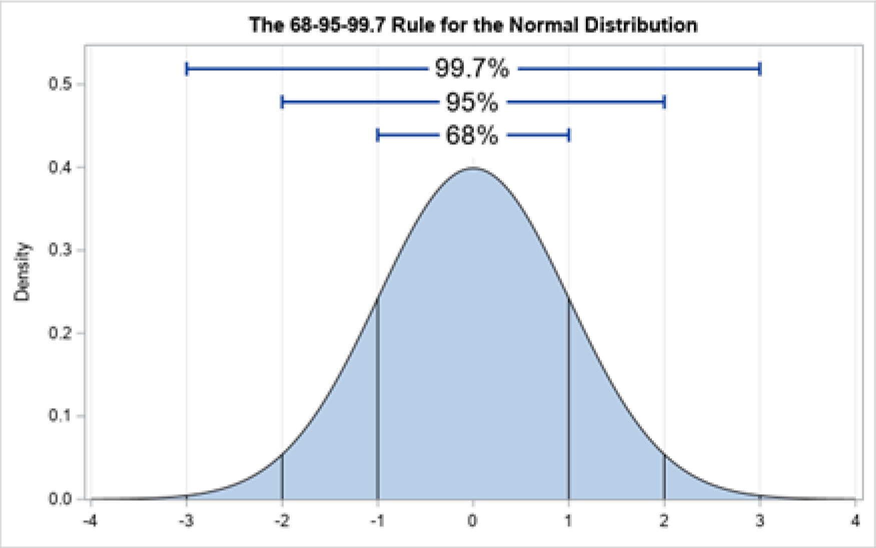


DISTRIBUCION NORMAL

También llamada distribución Gaussiana.

Nos permite **modelar y trabajar bajo supuestos** de que nuestros datos **siempre van a tender a la media**.

Varios modelos como los de **regresión** asumen que los datos tienen una distribución normal, y es nuestro deber transformar los datos para ello.

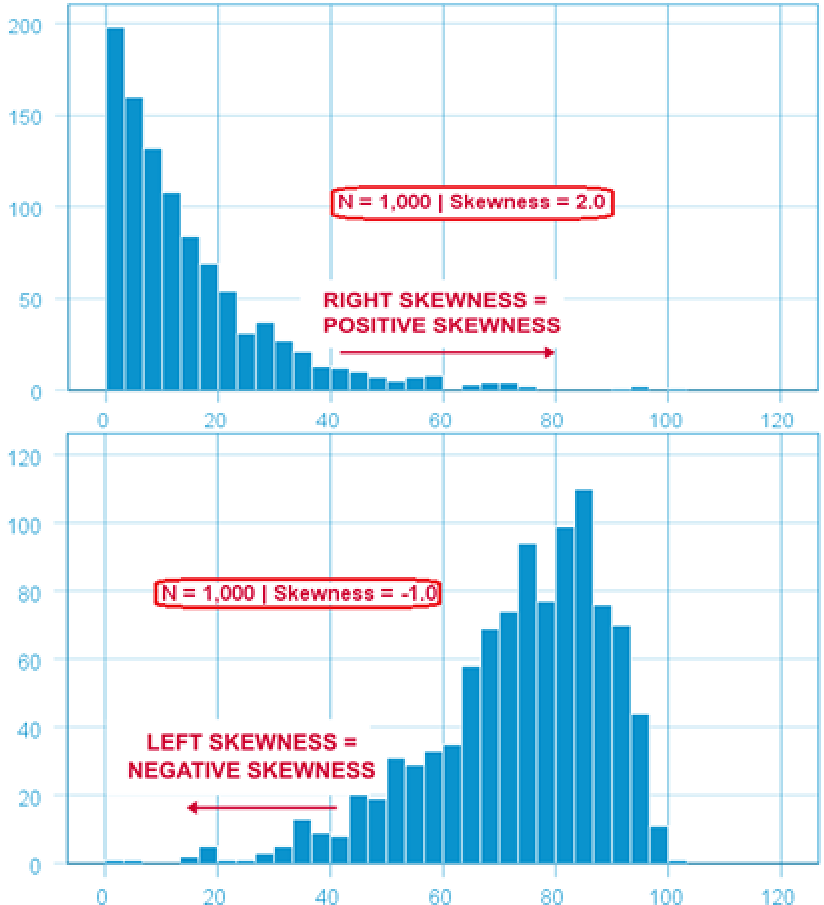


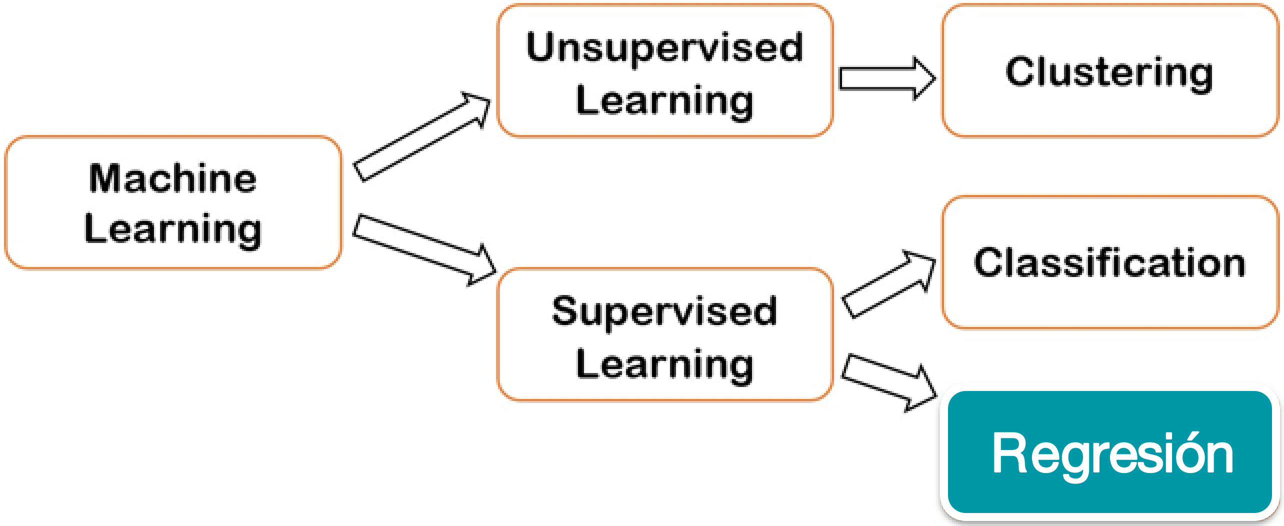
SKEWNESS

Skewness es una variable matemática que cuantifica una **asimetría** en la distribución gráfica de los datos.

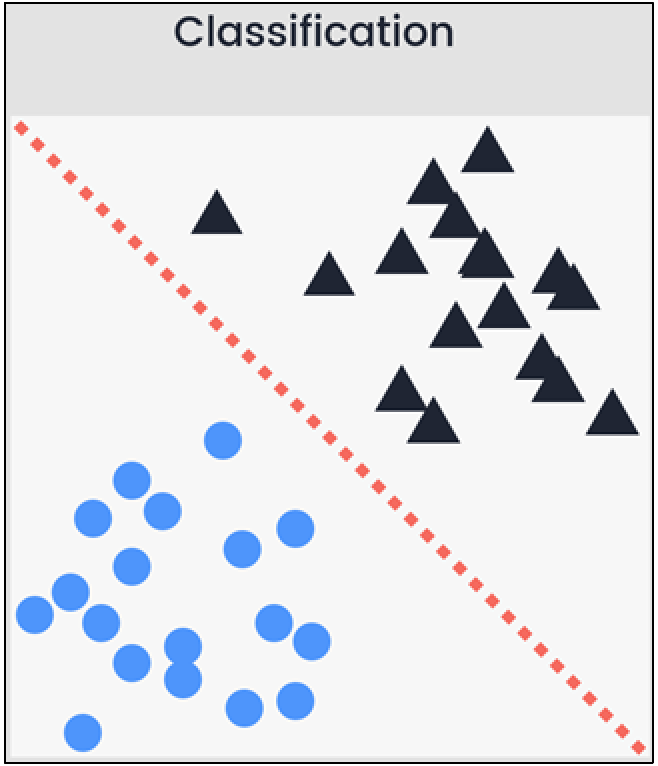
Esta medida **evidencia la presencia de outliers** debido a que **siempre ocasiona una asimetría** a la derecha (positiva) o a la izquierda (negativa).

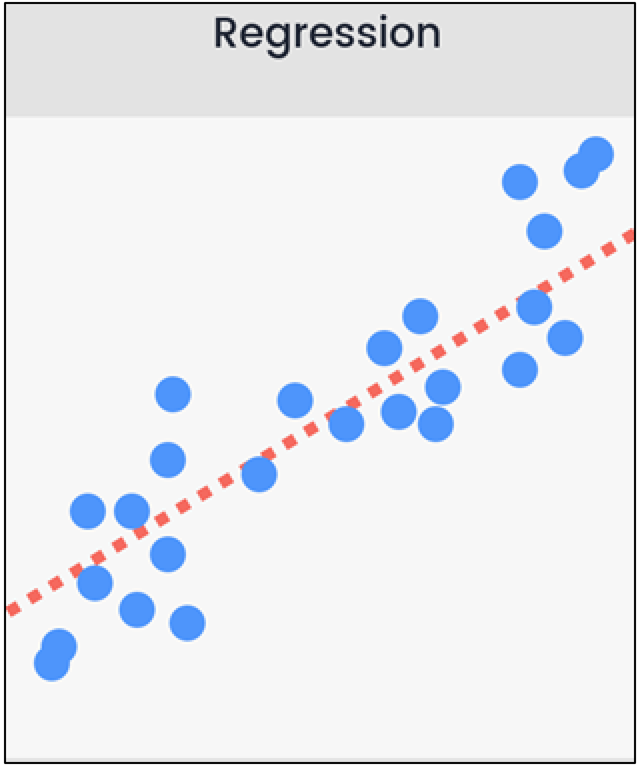
Una medida para **corregir** esta asimetría es **aplicar un logaritmo** sobre toda variable que presente un **skewness mayor a 1** (valor absoluto).




Machine Learning
Unsupervised Learning
Supervised Learning
Clustering
Classification
Regresión

MODELOS DE APRENDIZAJE SUPERVISADO





Aprendizaje Supervisado

Modelos de Regresión

CASO PRÁCTICO

Supongamos que deseo vender un diamante porque mi abuela me dejó como herencia un anillo engarzado con un **diamante de 1,35 quilates**, y quiero tener una idea de cuánto me pagarán.

Tomo un lápiz y un cuaderno, voy a una joyería y escribo el **precio de todos los diamantes de la vitrina y cuántos quilates tienen**.

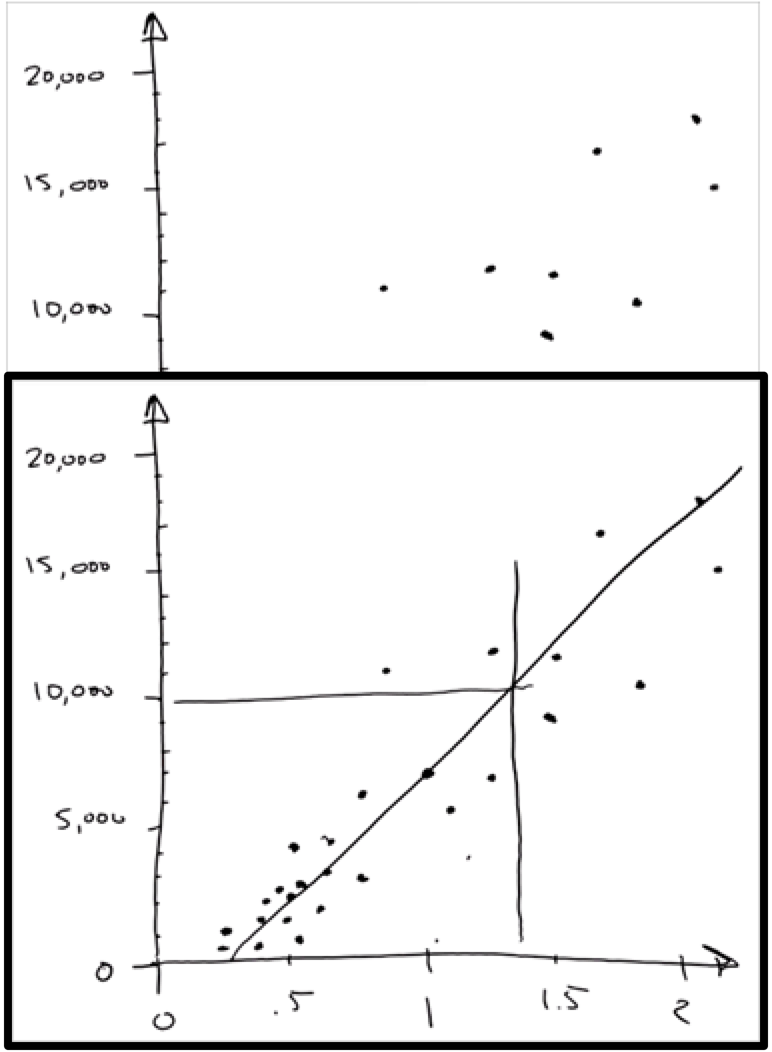
Carats	price
1.01	7,366
.49	985
.31	544
1.51	9,140
.37	493
.73	3,011
1.53	11,413
.56	1,814
.41	876
.74	2 690
.63	1, 190
.6	4, 172
2.06	11,764
1.1	4,682
1.31	6,171

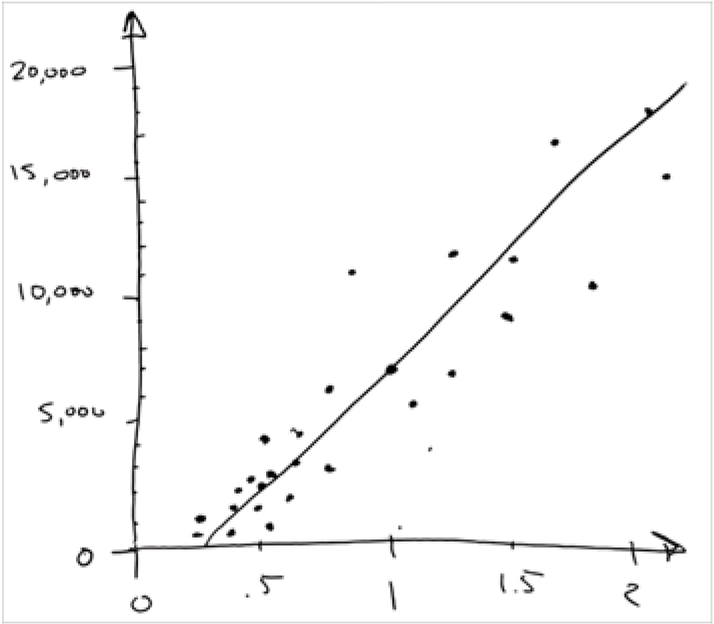
CASO PRÁCTICO

Ahora platearemos nuestra pregunta de forma directa: ¿cuánto costará comprar un diamante 1,35 quilates?

Nuestra lista **no contiene ningún diamante de 1,35 quilates**, pero podemos utilizar el **resto de nuestros datos para obtener una respuesta.**

Así que procedemos a dibujar un plano con dos variables: quilates en el eje X y el precio en el eje Y. Luego **ubicamos nuestros datos** sobre el plano según los valores x, y que les correspondan. Finalmente **trazamos una línea recta**



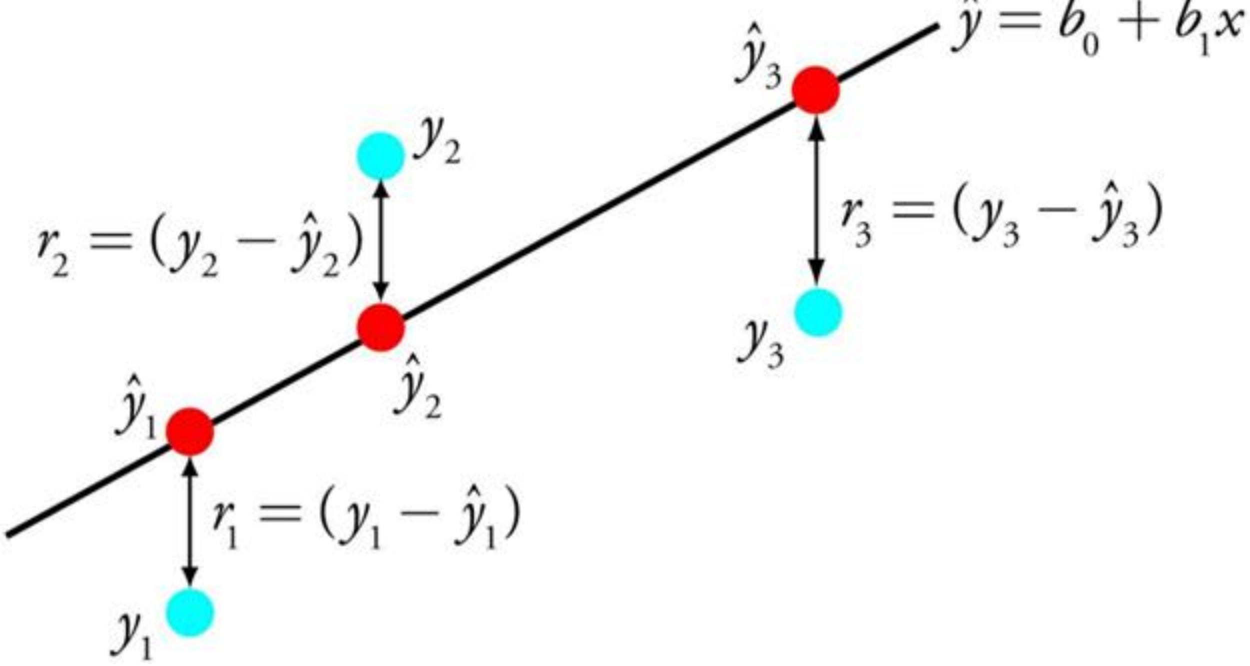


Podemos concluir por la gráfica que nos pagarán por el diamante de 1.35 quilates como máximo 10,000 dólares

REGRESIÓN LÍNEAL

- Es óptimo para problemas de regresión **con datos de distribución normal**

- Busca **minimizar el error** (función de coste)



COEFICIENTES DE REGRESIÓN

$$\underbrace{\boxed{\hat{y}}}_{\text{Valor predecido}} = \underbrace{\beta_0 + \beta_1}_{\text{Coeficientes}} \underbrace{\boxed{x}}_{\text{Entradas}}$$

Función de coste:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y})^2$$

Regresión lineal: múltiples variables

$$\boxed{\hat{y}} = \underbrace{\beta_0 + \beta_1}\boxed{x_1} + \dots + \underbrace{\beta_p}\boxed{x_p}$$

FUNCIONES DE COSTE

$$L_{OLS}(\hat{\beta}) = \sum_{i-1}^{n}(y_i - x_i'\hat{\beta})^2 = \left\|y - X\hat{\beta}\right\|^2$$

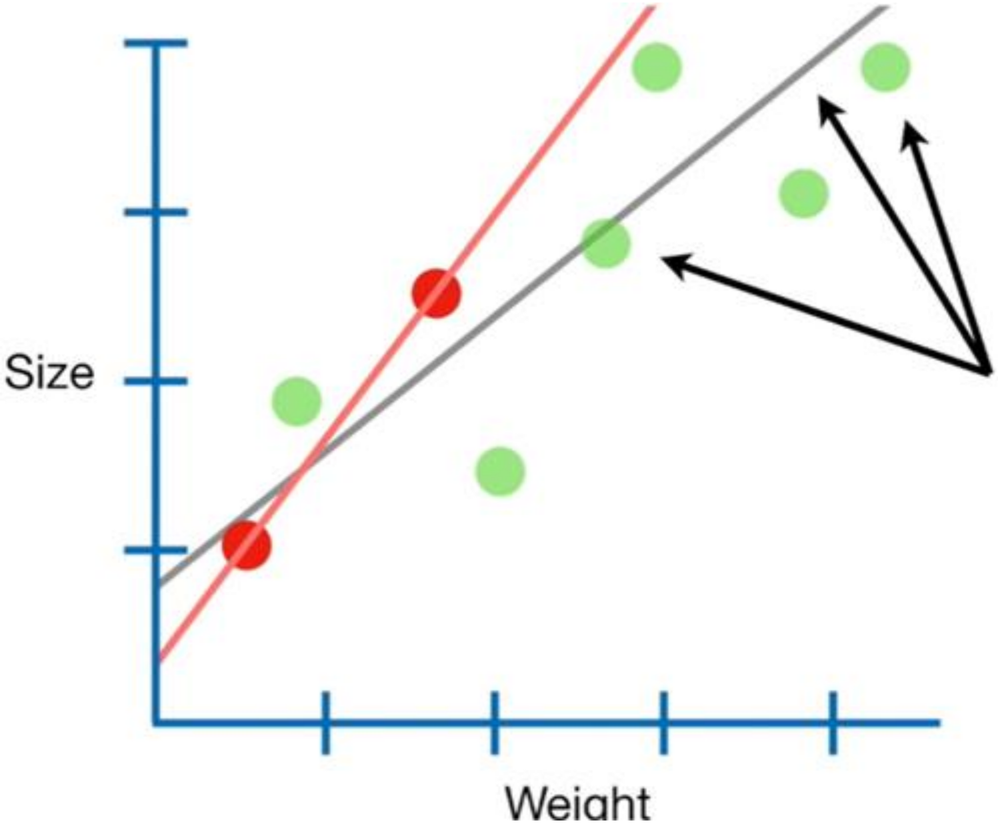
$$L_{ridge}(\hat{\beta}) = \sum_{i-1}^{n}(y_i - x_i'\hat{\beta})^2 + \lambda\sum_{j-1}^{m}\hat{\beta}_j^2 = \left\|y - X\hat{\beta}\right\|^2 + \lambda\left\|\hat{\beta}\right\|^2$$

$$L_{lasso}(\hat{\beta}) = \sum_{i-1}^{n}(y_i - x_i'\hat{\beta})^2 + \lambda\sum_{j-1}^{m}\left|\hat{\beta}_j\right|$$

- Mayor información en la siguiente ruta URL:

https://www.analyticsvidhya.com/blog/2016/01/ridge-lasso-regression-python-complete-tutorial/

REGRESIÓN RIDGE



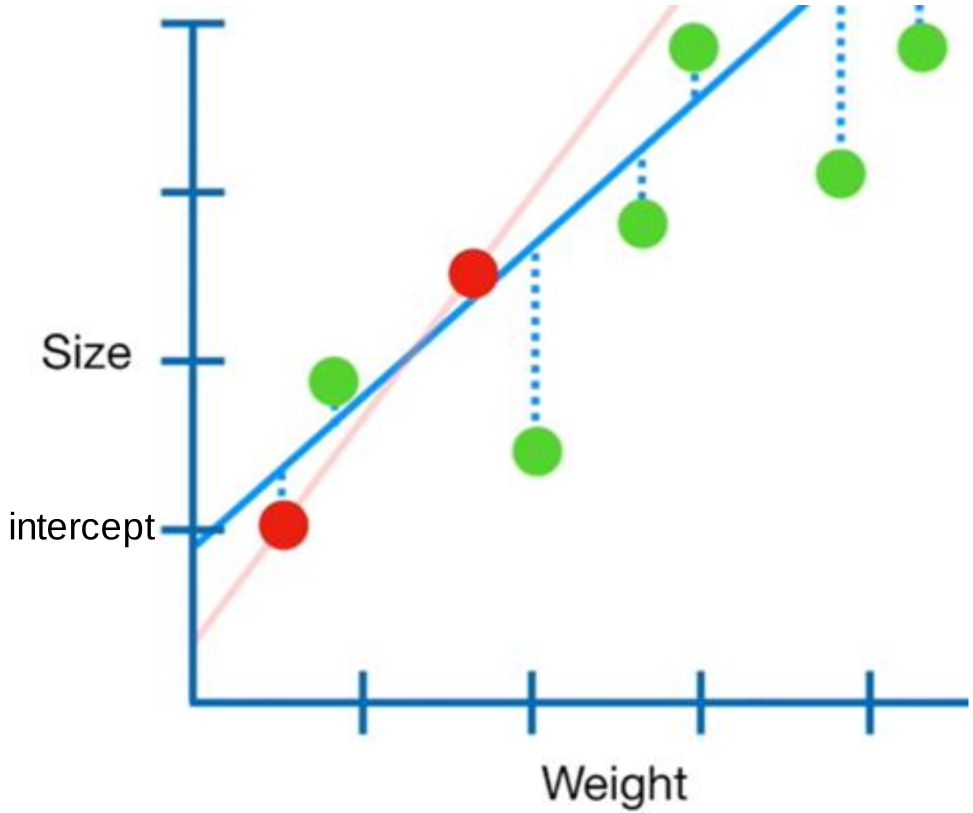
La primera aproximación son dos puntos aleatorios

De esta manera obtenemos un error respecto a estos dos puntos = 0

Sin embargo el error acumulado respecto al resto de los puntos es muy grande.

Entonces el objetivo principal de la Regresión Ridge es encontrar una **línea recta diferente y más cercano al total de los puntos**

REGRESIÓN RIDGE



En la Regresión Ridge se busca **minimizar el error total** considerando también el lamba y el coeficiente^2

Size = y-axis intercept + slope x **Weight**

...it minimizes...

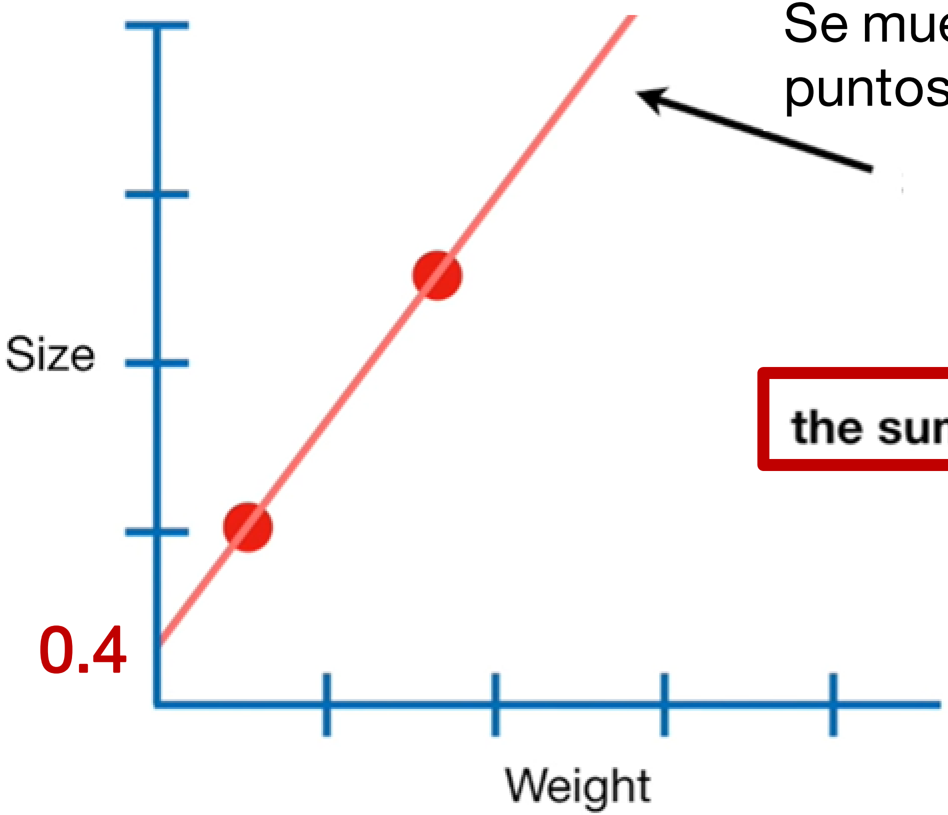
the sum of the squared residuals

+

λ x **the slope²**

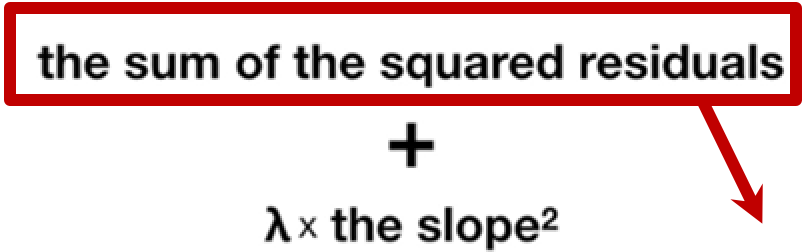
"**slope**" es el **coeficiente** que acompaña **a cada variable (X) de la ecuación**

REGRESIÓN RIDGE



Se muestra una recta ajustada solo a dos puntos con la siguiente ecuación:

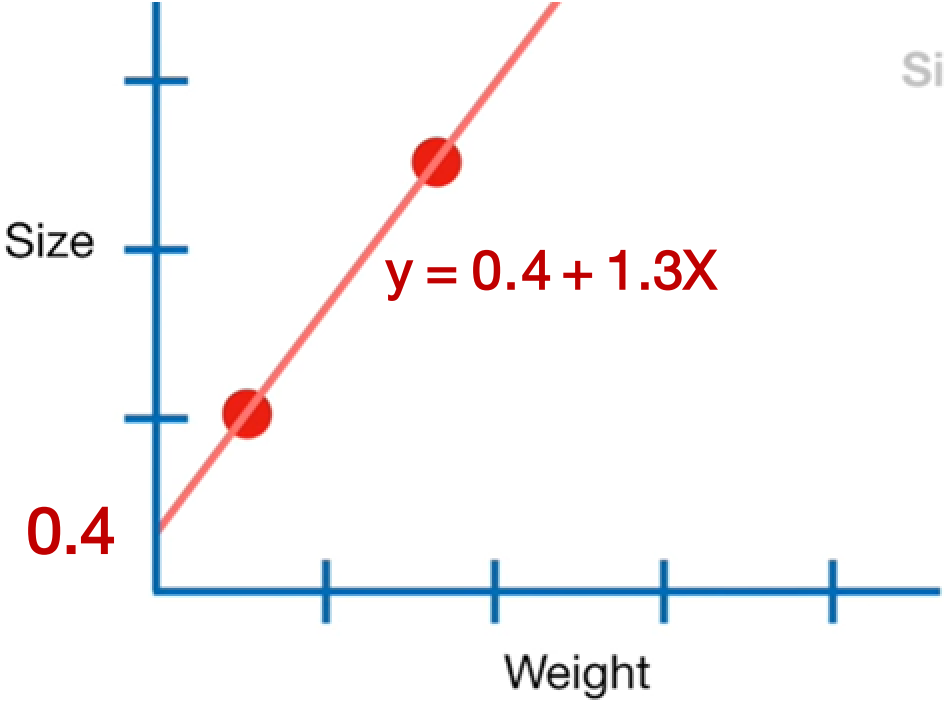
$$y = 0.4 + 1.3X$$



Para esta ecuación inicial el error de los residuos es igual a cero

REGRESIÓN RIDGE

el coeficiente de la variable **Weight** es igual a **1.3**



$Size = 0.4 + \boxed{1.3} \times Weight$

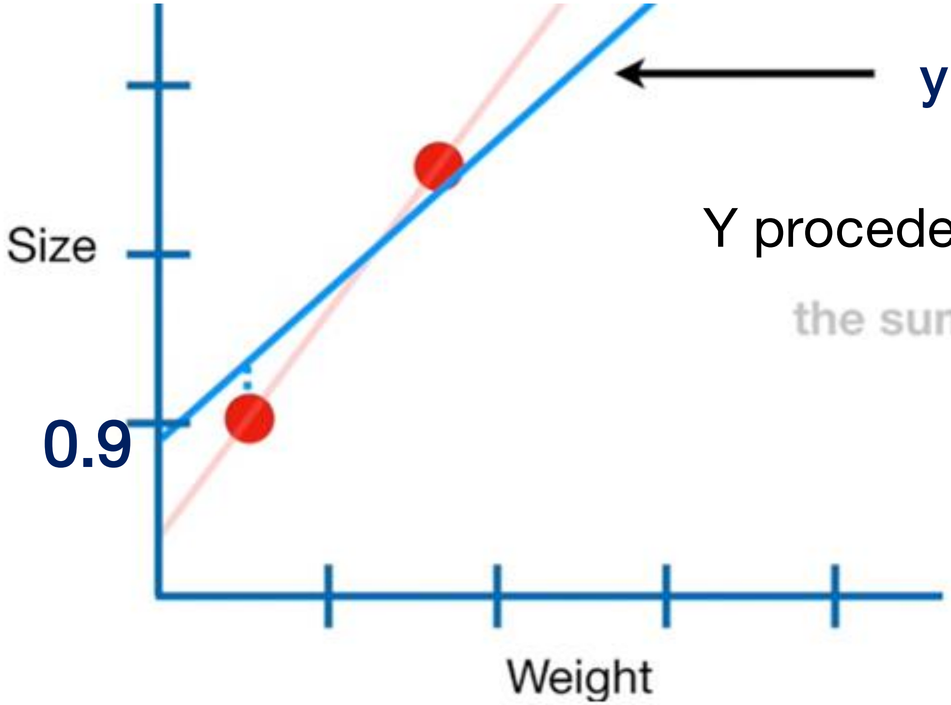
$0 + \lambda \times \boxed{1.3^2}$

$0 + 1 \times 1.3^2 = 0 + 1.69 = \boxed{1.69}$

para este ejemplo **lambda** es igual a 1 $\longrightarrow$ 1×1.3^2

REGRESIÓN RIDGE

Ahora creamos una nueva recta **con nuevo intercepto y un nuevo coeficiente**



$y = 0.9 + 0.8X$

Y procedemos a calcular un nuevo error total:

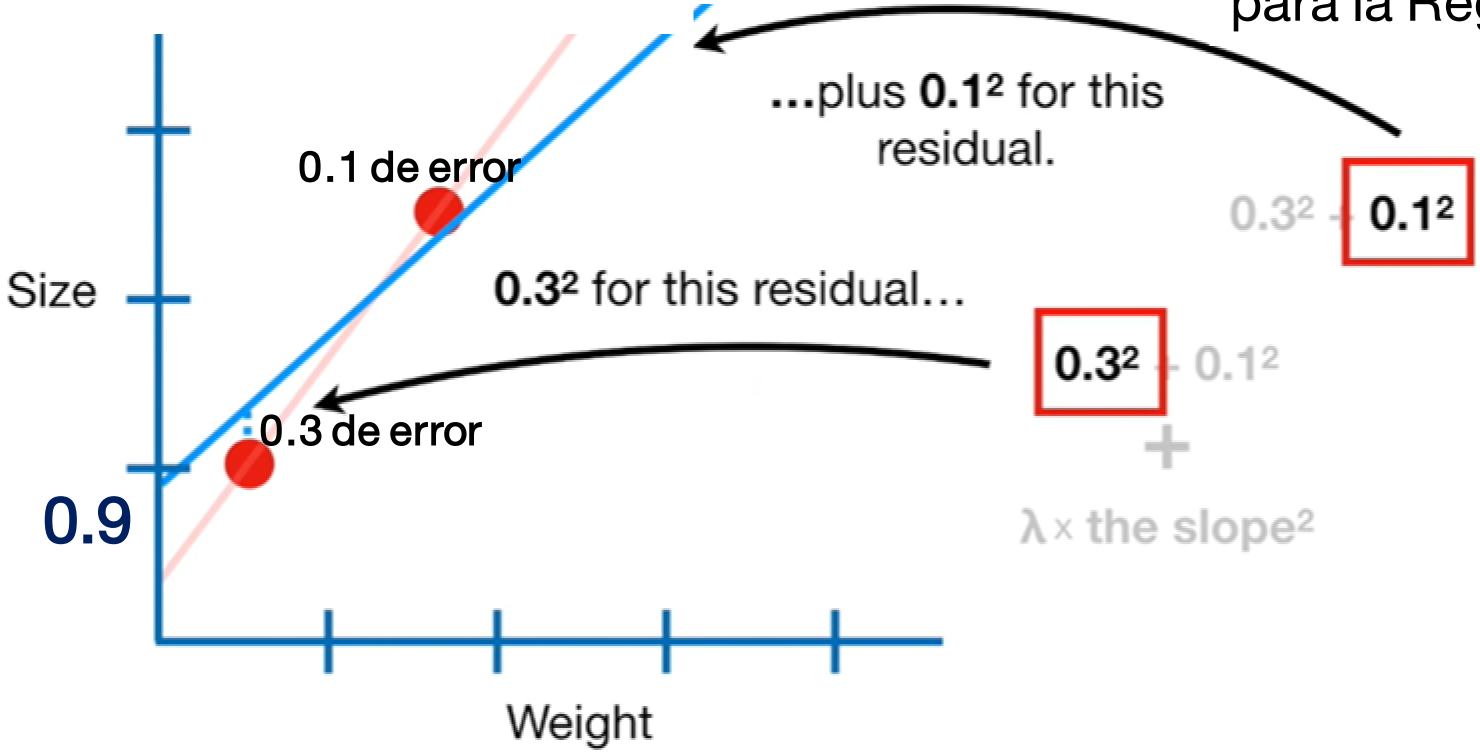
the sum of the squared residuals

$+$

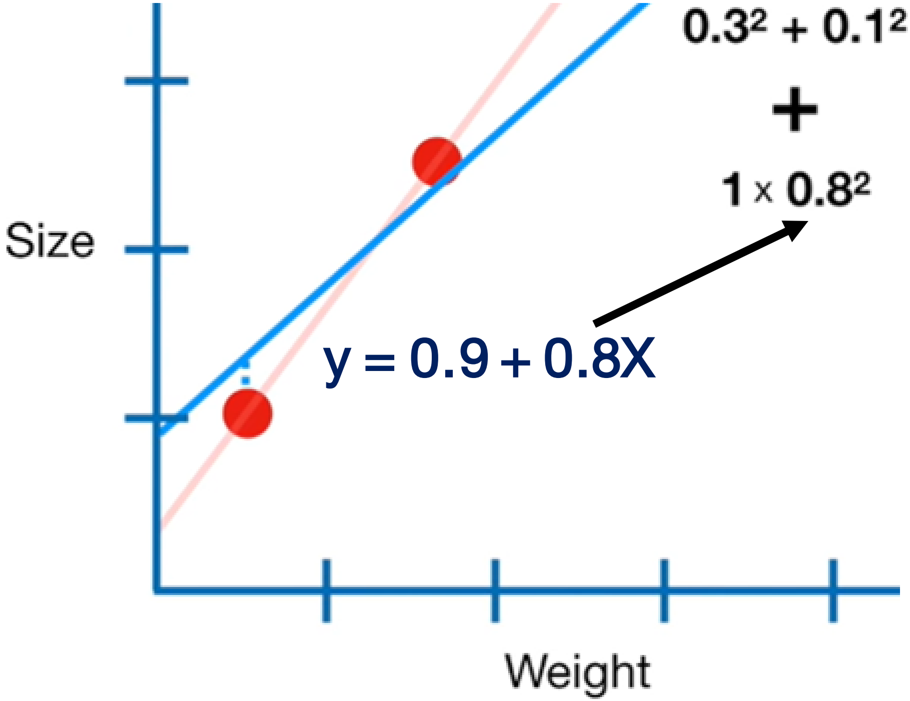
$\lambda \times$ the slope^2

REGRESIÓN RIDGE

Calculamos **nuevos errores** para la Regresión Ridge



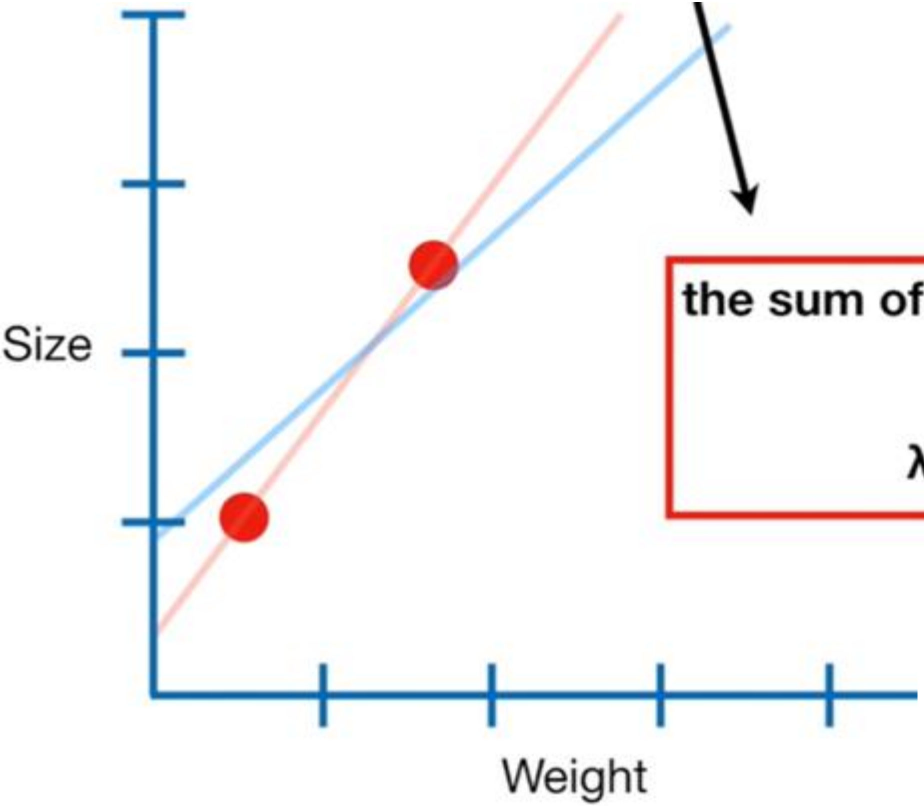
REGRESIÓN RIDGE



$$0.3^2 + 0.1^2 + 1 \times 0.8^2 = 0.09 + 0.01 + 0.64 \boxed{= 0.74}$$

Esta nueva recta obtuvo un **menor error total.** De esta manera podemos seguir creando nuevas rectas hasta encontrar el **mínimo** error total.

REGRESIÓN LASSO



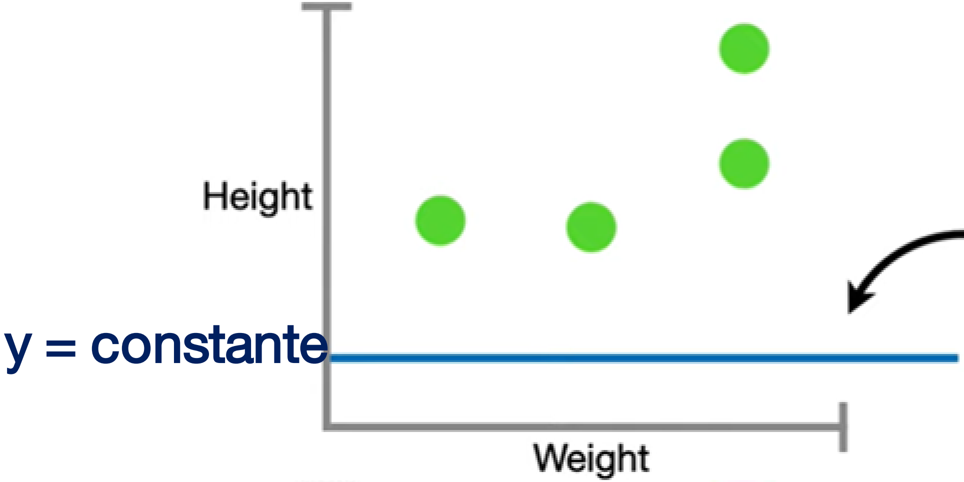
the sum of the squared residuals

+

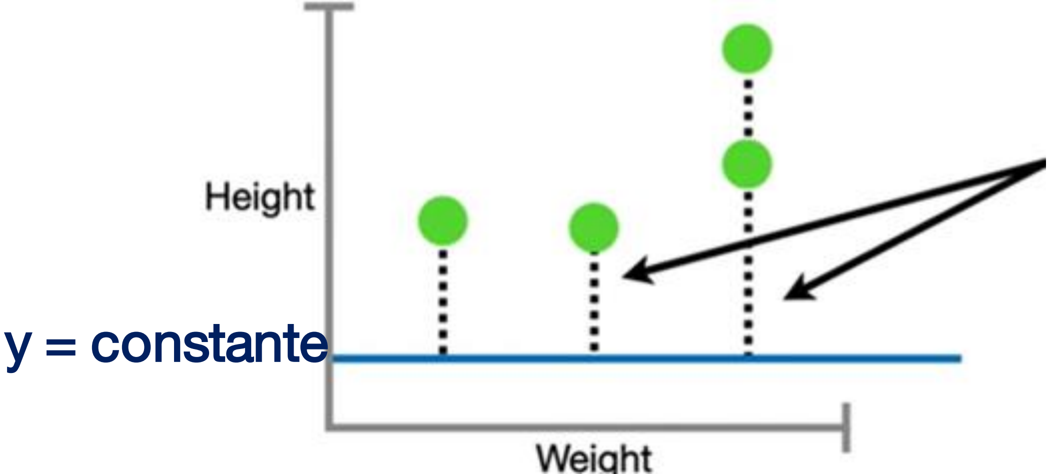
$\lambda \times |\text{the slope}|$

La diferencia entre Ridge y Lasso es la **penalización** (exponente de **slope** o coeficiente de cada variable). Lasso **solamente saca el valor absoluto,** es decir **penaliza menos**

OPTIMIZACION DEL COEFICIENTE (SLOPE)

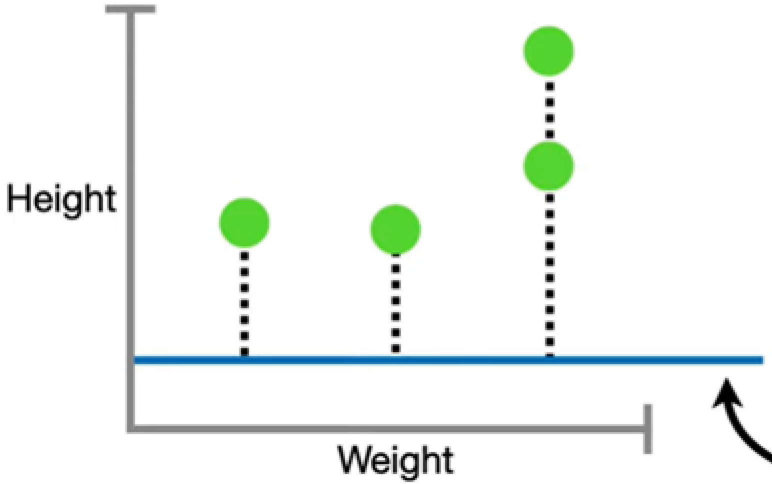


Comenzamos con una línea horizontal como primera aproximación a los datos

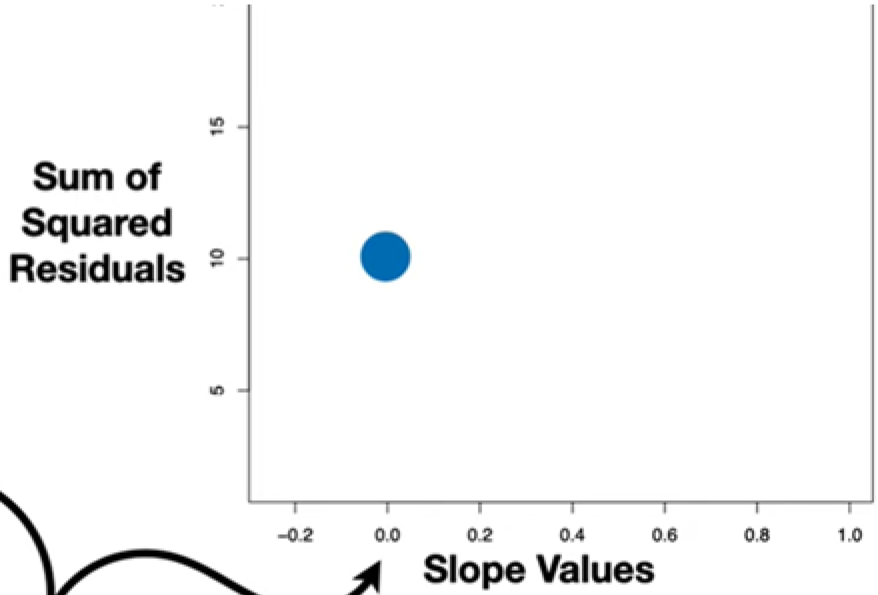


Podemos observar que el **error total de los residuales** es muy grande

OPTIMIZACION DEL COEFICIENTE (SLOPE)

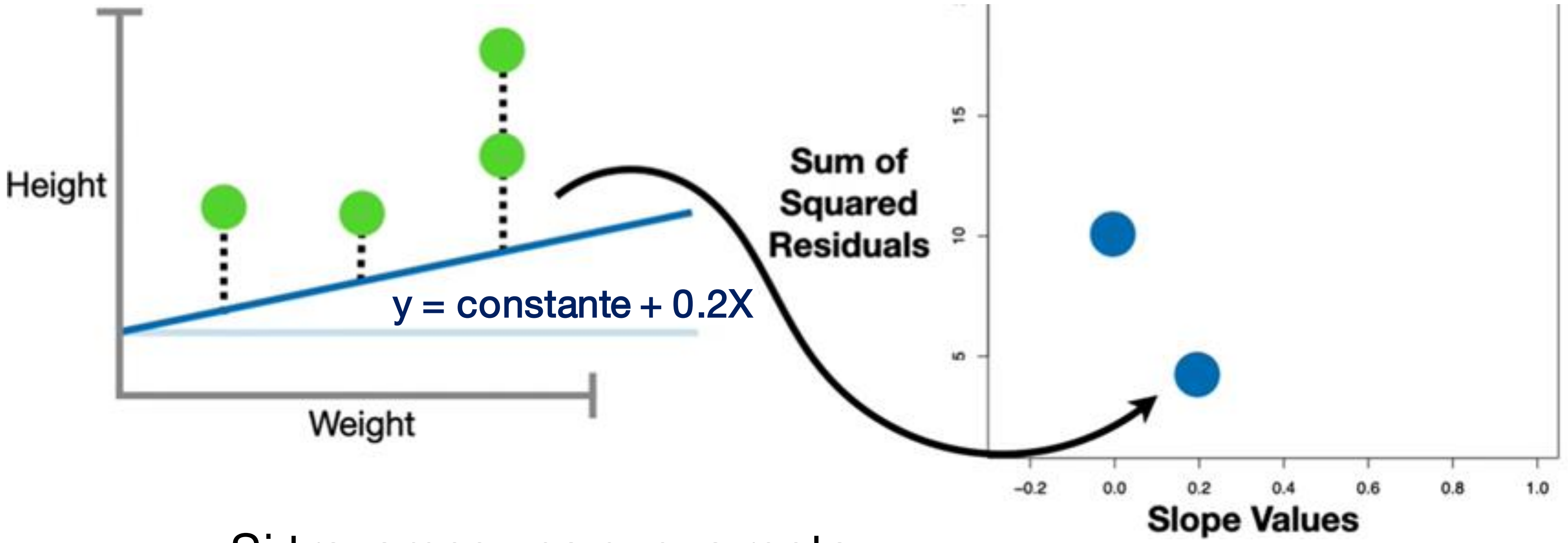


$y = \text{constante} + 0X$



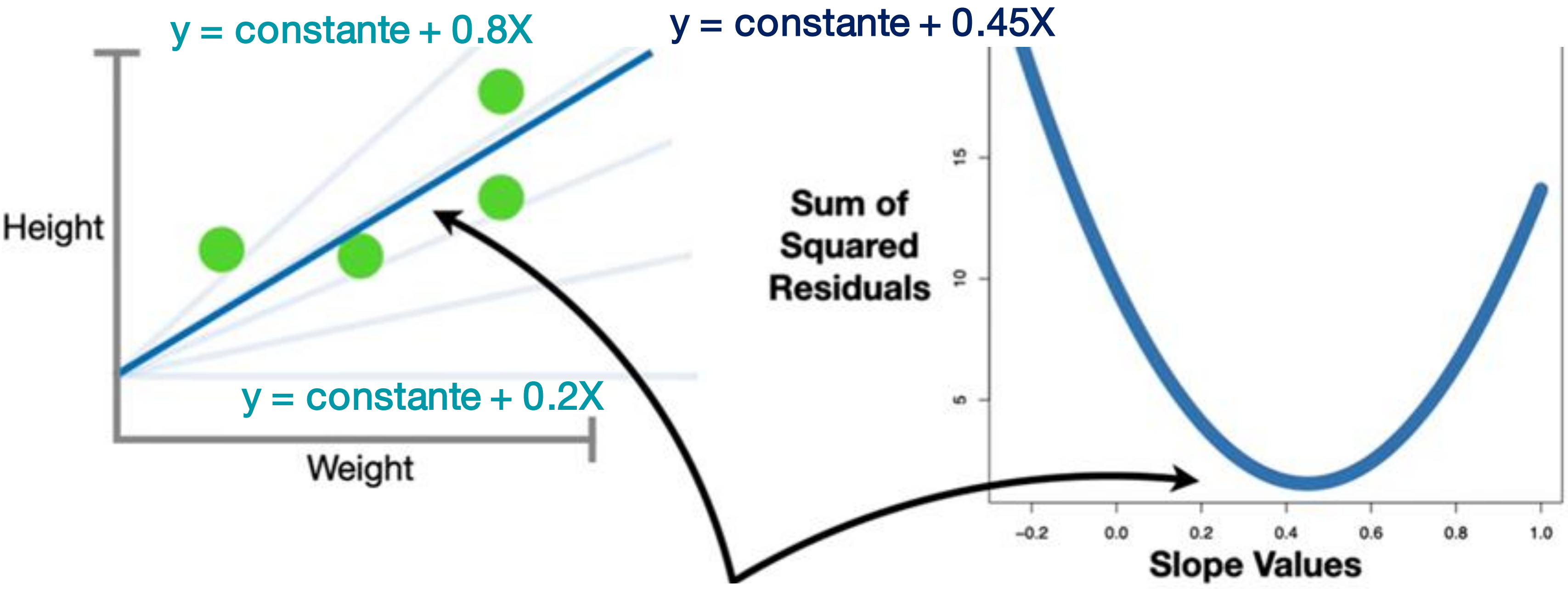
Cuando se traza una línea recta el **coeficiente que acompaña a Weight = 0**

OPTIMIZACION DEL COEFICIENTE (SLOPE)



Si trazamos una nueva recta con un **coeficiente diferente de 0 (0.2)** generamos un **menor error total de residuales**

OPTIMIZACION DEL COEFICIENTE (SLOPE)



El punto óptimo con el **mínimo error total de los residuales** ocurre cuando el **coeficiente es 0.45**

$$\textbf{Size} = \text{y-intercept} + \text{slope}_1 \times \textbf{Weight} + \text{diet difference} \times \textbf{High Fat Diet} + \text{slope}_2 \times \textbf{Age} + \text{slope}_3 \times \textbf{Size of Father}$$

Cuando **todas** las variables predictoras **son significativas** es correcto **penalizar los coeficientes de forma significativa también.**
Para esta opción se recomienda la **Regresión Ridge.**

Cuando **no estamos seguros** si todas las variables predictoras son significativas es correcto **penalizar menos los coeficientes**.
Para esta opción se recomienda la **Regresión Lasso.**

$$\textbf{Size} = \text{y-intercept} + \text{slope} \times \textbf{Weight} + \text{diet difference} \times \textbf{High Fat Diet} + \cancel{\text{astrological offset} \times \textbf{Sign}} + \cancel{\text{airspeed scalar} \times \textbf{Airspeed of Swallow}}$$

MÉTRICAS DE REGRESIÓN

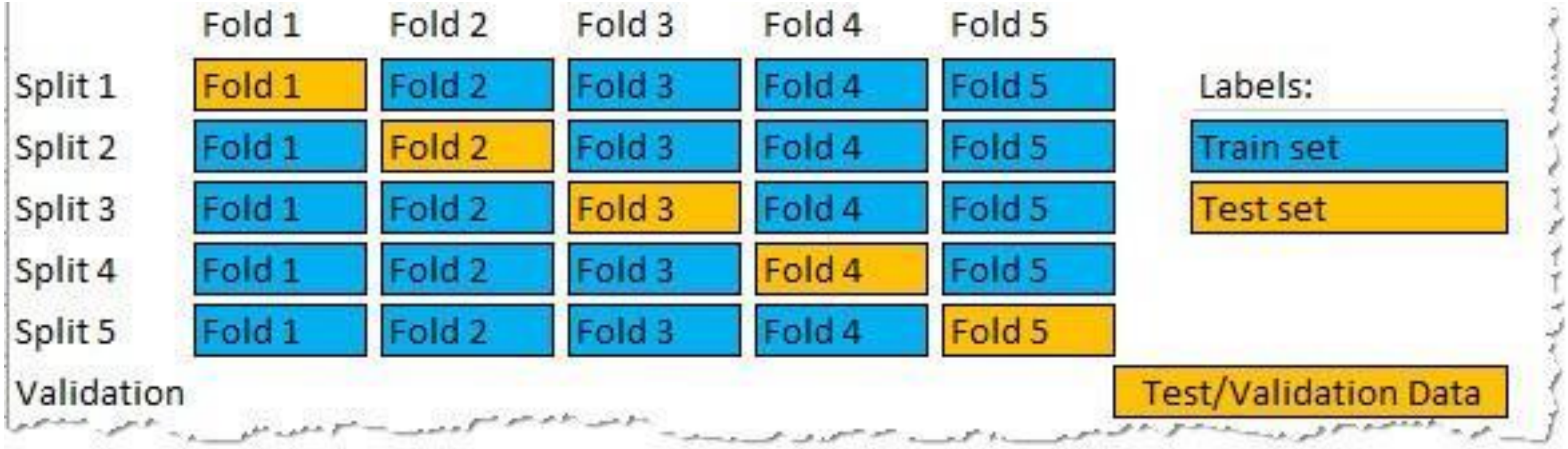
$$MSE = \frac{1}{m}\sum_{i=1}^{m}(y_i - t_i)^2$$

$$RMSE = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(y_i - t_i)^2}$$

$$MAE = \frac{1}{m}\sum_{i=1}^{m}\left|y_i - t_i\right|$$

$$MSLE = \frac{1}{m}\sum_{i=1}^{m}\left(\log(y_i + 1) - \log(t_i + 1)\right)^2$$

CROSS-VALIDATION: K-FOLDS



- Mayor información en la siguiente ruta URL:

https://towardsdatascience.com/complete-guide-to-pythons-cross-validation-with-examples-a9676b5cac12