Regresión lineal aplicado a ML

1.- Regresión lineal.

2.- Métricas de rendimiento.

3.- Training set y testing set.

Regresión lineal.

Regresión lineal

El escenario más sencillo con el que nos podemos encontrar es el de una variable dependiente y una única variable independiente, que siguen una relación aproximadamente lineal, salvo ruido (típicamente, normalmente distribuido):

$$y \sim \beta_0 + \beta_1 x$$

En general, vamos a desconocer la relación de arriba, pero con un conjunto de datos y mediante cuadrados mínimos podemos estimar los coeficientes:

$$\hat{y} = \widehat{\beta_0} + \widehat{\beta_1} x$$

Regresión lineal

Si tenemos múltiples variables independientes, estamos en un caso de regresión lineal múltiple:

$$y \sim \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_m x_m$$

Ejemplo: precio de un auto en términos de consumo de combustible, año de fabricación, etc.

Nuestros datos disponibles: n filas del data set.

$$\left(x_1, y_1\right), \left(x_2, y_2\right), \ldots, \left(x_i, y_i\right), \ldots, \left(x_n, y_n\right)$$

Sin embargo, este modelo nos impone una relación lineal entre la variable dependiente y sus regresores.

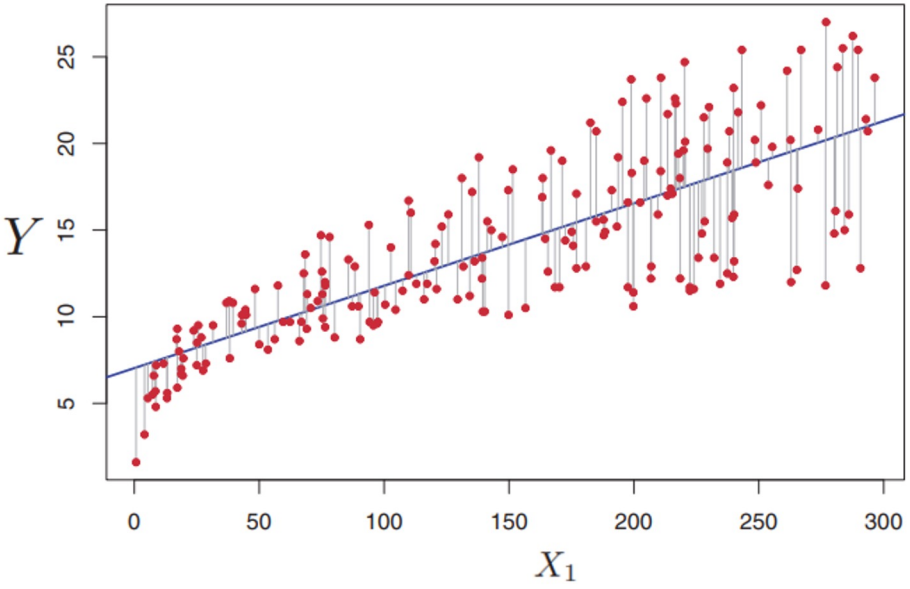
Regresión lineal

Con un modelo como el anterior mostrado:

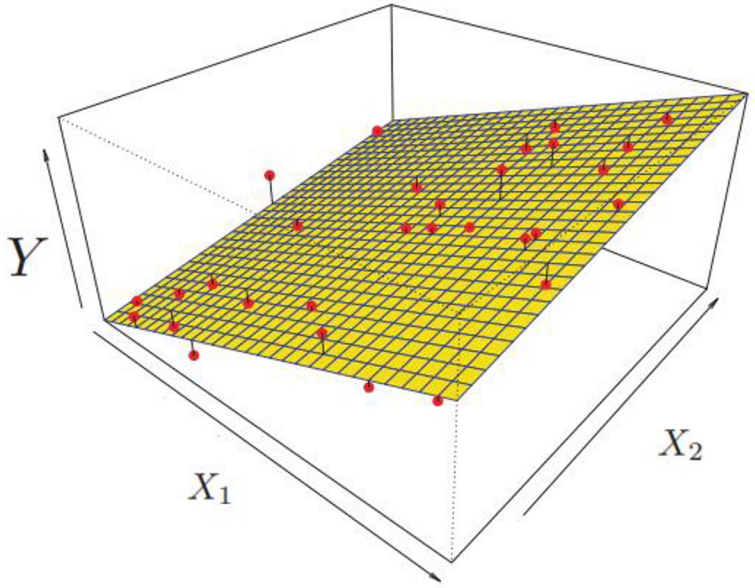
$$y \sim \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_m x_m$$

Tenemos lo siguiente para m=1 y m=2:

m=1



m=2



Para m=1: recta

Para m=2: plano

Para m=3: Hiper-plano

Regresión lineal

Los modelos tienen siempre un error:

$$\mathrm{RSS} = \sum_{i=1}^{n} (y_i - f(x_i))^2$$

Residual Sum of Square, pero tambien llamado:
Sum of Squared estimate of Errors (SSE)

$$\mathrm{RSS} = \sum_{i=1}^{n} (\widehat{\varepsilon}_i)^2 = \sum_{i=1}^{n} (y_i - (\widehat{\alpha} + \widehat{\beta} x_i))^2$$

β_0

El algoritmo de regresión lineal debe estimar los valores de los β. De manera general, se estiman de la siguiente manera:

$$X\hat{\beta} = y \iff$$

$$X^{\mathrm{T}} X\hat{\beta} = X^{\mathrm{T}} y \iff$$

$$\hat{\beta} = (X^{\mathrm{T}} X)^{-1} X^{\mathrm{T}} y.$$

Métricas de rendimiento.

Métricas

Para regresión existen diversas métricas para medir qué tan bueno es nuestro modelo.

Existen diversas métricas y no todas miden exactamente lo mismo. Por ejemplo, no todas miden el resultado final de la estimación.

En esta presentación, revisaremos las métricas más usadas en regresión incluyendo las funciones respectivas de sckikit-learn.

Link a scikit-learn

Métricas

MSE (Mean Square Error):

$$MSE = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \widehat{y_i}\right)^2$$

En scikit-learn: sklearn.metrics.mean_squared_error

MAE (Mean Absolute Error):

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

En scikit-learn: sklearn.metrics.mean_absolute_error

Métricas

R2 (Coeficiente de determinación):

$$R^2 = 1 - \frac{\sum_{i=1}^{N} \left(y_i - \widehat{y}_i \right)^2}{\sum_{i=1}^{N} \left(y_i - \bar{y} \right)^2}$$

En scikit-learn: sklearn.metrics.r2_score

R2 ajustado:

$$R^2_{\alpha} = 1 - \left(\frac{N-1}{N-D-1} \right) \times \left(1 - R^2 \right)$$

La ventaja es que R2 ajustado no aumenta en función al número de variables que incorpora el modelo y R2 si.

Métricas

MAPE (Mean Absolute percentage Error):

$$MAPE = \frac{100\%}{N} \sum_{i=1}^{N} \left| \frac{y_i - \widehat{y_i}}{y_i} \right|$$

MAPE entrega un error relativo ya que divide la diferencia absoluta por el valor real.

Cuidado con los valores muy pequeños de y_i

Training set
y testing set.

Training set y testing set

Los datos originales será nuestro 100%

De este 100%, dividiremos los datos en training set y testing set. ¿En qué porcentaje?:

- El training set varia entre 60% y 80% y el testing set entre 40% y 20% respectivamente.

- El porcentaje se elige según la cantidad de datos que se tienen, el algoritmo a utilizar y cualidades particulares de los datos.

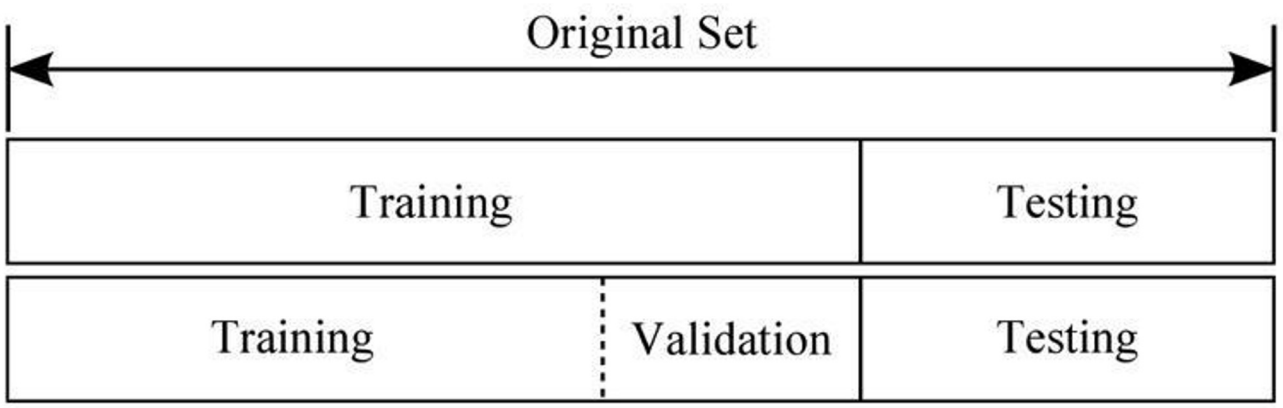
Es importante recalcar que el testing set nunca será ocupado sino hasta en la fase final.

El training set comúnmente se subdivide en: Training set y Validation set.

Training set y testing set

La división de los datos comúnmente se realiza aleatoria (Uniforme).

Si los datos están desbalanceados, la división se realiza intentando mantener los porcentajes de datos de cada clase.



Training set y testing set

La división realizada de los datos originales a veces se le da el nombre de *shuffle.*

Un shuffle entonces es la partición de los dato originales en training set y testing set. De manera aleatoria o pseudo-aleatoria.

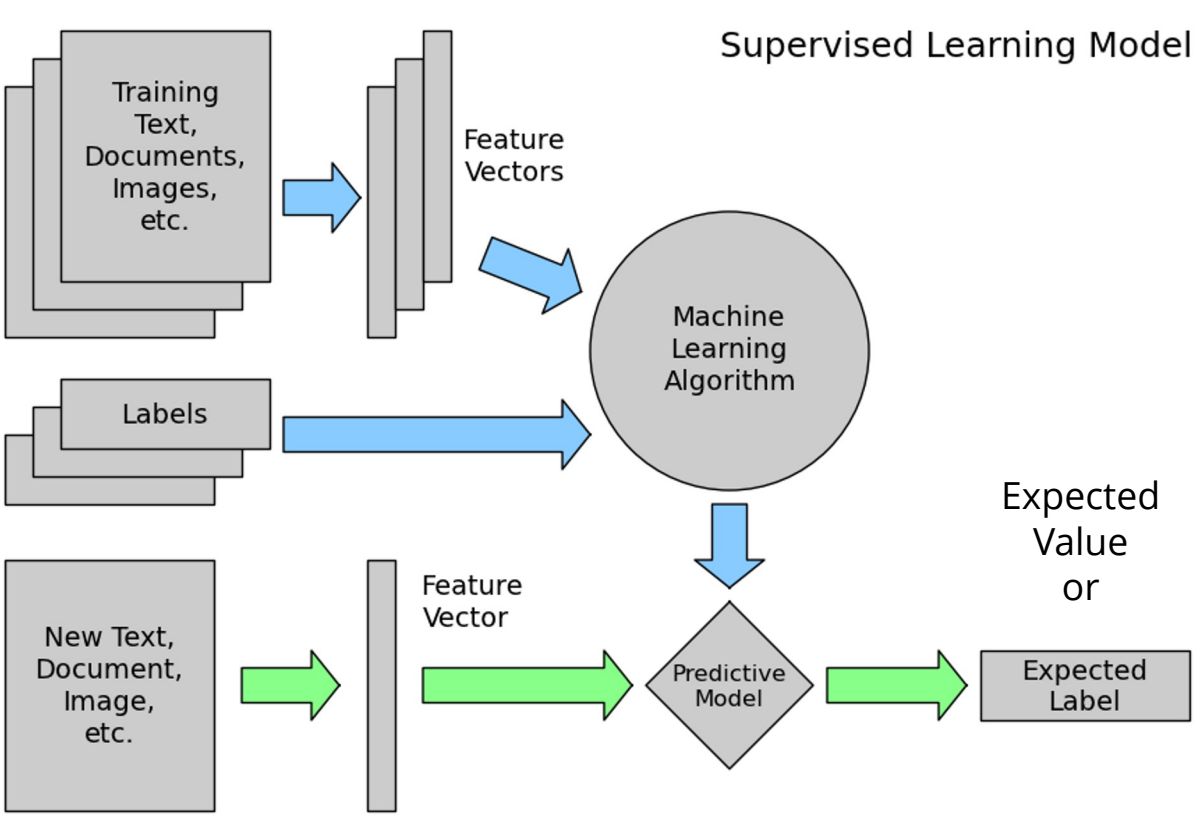
Para obtener resultados más robustos, se acostumbra a generar muchos shuffles con los mismo datos originales.

Con esto, y como es aleatorio, cada shuffle será distinto.

Así, al reportar los resultados, se podrán calcular una media y varianza de éstos. En definitiva, una estadística de los resultados.

Training set y testing set

En resumen, aplicar
Supervised Learning es:



Training set y testing set

¿Se recuerda de la normalización y estandarización de los datos? Bueno, acá se debe utilizar. ¿Cómo?

Se debe realizar al training set de manera separada al testing set.

Una vez realizada la normalización o estandarización al training set, se debe guardar los parámetros según cada modelo:

- **Normalización:** vector de mínimos y máximos.

sklearn.preprocessing.MinMaxScaler

- **Estandarización:** vector de medias y vector de desviaciones estándar.

sklearn.preprocessing.StandardScaler

Estos parámetros se utilizan para realizar el proceso en el testing set.

Training set y testing set

A modo de ejemplo, si quisiéramos escalar los datos entre 0 y 1 (Normalización o MinMaxScaler), tendríamos que hacer lo siguiente:

$$X_{new}^{te} = \frac{X_{old}^{te} - X_{min}^{tr}}{X_{max}^{tr} - X_{min}^{tr}}$$

Donde:

$X_{(\cdot)}^{te}$: son datos de prueba (testing).

$X_{(\cdot)}^{tr}$: son datos de entrenamiento (training).

¿Donde estamos en CRISP-DM?

CRISP-DM

Fuente: https://www.proglobalbusinesssolutions.com/six-steps-in-crisp-dm-the-standard-data-mining-process/