Machine learning y Deep learning con Python: Lectura 3

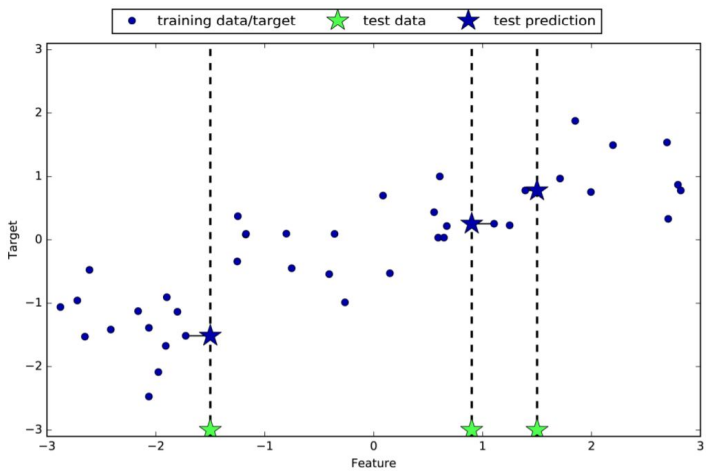
Ing. Pedro Rotta

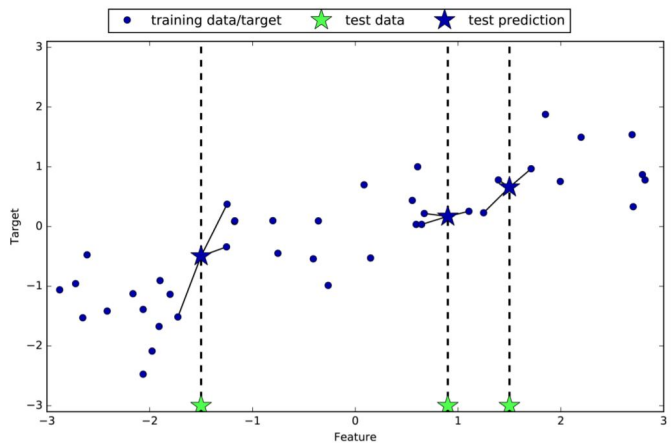
Universidad de Piura - Vida Universitaria

Enero-2022

KNN para Regresión

También se puede usar el algortimo KNN para Regresión. En este caso, si K = 1, por ejemplo, el valor predicho, toma como resultado el target del punto más cercano, usando la distancia euclidiana. En el caso de ser varios K, el resultado se encuentra promediando los resultados de los K puntos.





Métricas para regresión

Cuando entrenamos un modelo para regresión, lo que tenemos que tener en cuenta es el **RSME (Root Mean Square Error)** cuya formula se expresa como:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}}$$

La otra métrica para regresión a tener en cuenta es R^2 que se conoce como el **coeficiente de determinación** que se expresa como:

$$R^2 = 1 - \frac{RSS}{TSS}$$

Donde:

$RSS = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2$ y $TSS = \sum_{i=1}^{n}(y_i - \bar{y}_i)^2$. Donde $\bar{y}_i$ es el valor medio de todos los y_i

Computarización KNN Regresion

Para computarizar el método KNN para regresión importamos el objeto de sklearn:

```
from sklearn.neighbors import KNeighborsRegressor
```

Ver problema 1.

Modelos lineales

Existen 2 modelos lineales, que son un poco más restrictivos que el OLS. Estos son la **regresión ridge** y la **regresión lasso**.

Regresión Ride: También conocida como regularización L2. Agrega un componente α multiplicado por el cuadrado de los pesos a la función de costo reduciendo el sobreajuste en la regresión OLS. El rol del parámetro α es minimizar el efecto de colinealidad cuando se tienen muchos features o sobreajuste de los datos.

Modelos lineales

Existen 2 modelos lineales, que son un poco más restrictivos que el OLS. Estos son la **regresión ridge** y la **regresión lasso**.

Regresión Ride: También conocida como regularización L2. Agrega un componente α multiplicado por el cuadrado de los pesos a la función de costo reduciendo el sobreajuste en la regresión OLS. El rol del parámetro α es minimizar el efecto de colinealidad cuando se tienen muchos features o sobreajuste de los datos.

Para computarizar la regresión ride, se importa desde sklearn:

Modelos lineales

Existen 2 modelos lineales, que son un poco más restrictivos que el OLS. Estos son la **regresión ridge** y la **regresión lasso**.

Regresión Ride: También conocida como regularización L2. Agrega un componente α multiplicado por el cuadrado de los pesos a la función de costo reduciendo el sobreajuste en la regresión OLS. El rol del parámetro α es minimizar el efecto de colinealidad cuando se tienen muchos features o sobreajuste de los datos.

Para computarizar la regresión ride, se importa desde sklearn:

```
from sklearn.linear_model import Ride
```

Modelos Lineales

Regresión Lasso: También conocida como regularización L2. Agrega un componente α multiplicado por el valor absoluto de los pesos a la función de costo cuando se desarrolla el algoritmo reduciendo el sobreajuste en la regresión OLS. El rol del parámetro α es minimizar el efecto de colinealidad cuando se tienen muchos features o sobreajuste de los datos.

Modelos Lineales

Regresión Lasso: También conocida como regularización L2. Agrega un componente α multiplicado por el valor absoluto de los pesos a la función de costo cuando se desarrolla el algoritmo reduciendo el sobreajuste en la regresión OLS. El rol del parámetro α es minimizar el efecto de colinealidad cuando se tienen muchos features o sobreajuste de los datos.

Para computarizar la regresión lasso, se importa desde sklearn:

Modelos Lineales

Regresión Lasso: También conocida como regularización L2. Agrega un componente α multiplicado por el valor absoluto de los pesos a la función de costo cuando se desarrolla el algoritmo reduciendo el sobreajuste en la regresión OLS. El rol del parámetro α es minimizar el efecto de colinealidad cuando se tienen muchos features o sobreajuste de los datos.

Para computarizar la regresión lasso, se importa desde sklearn:

```
from sklearn.linear_model import Lasso
```

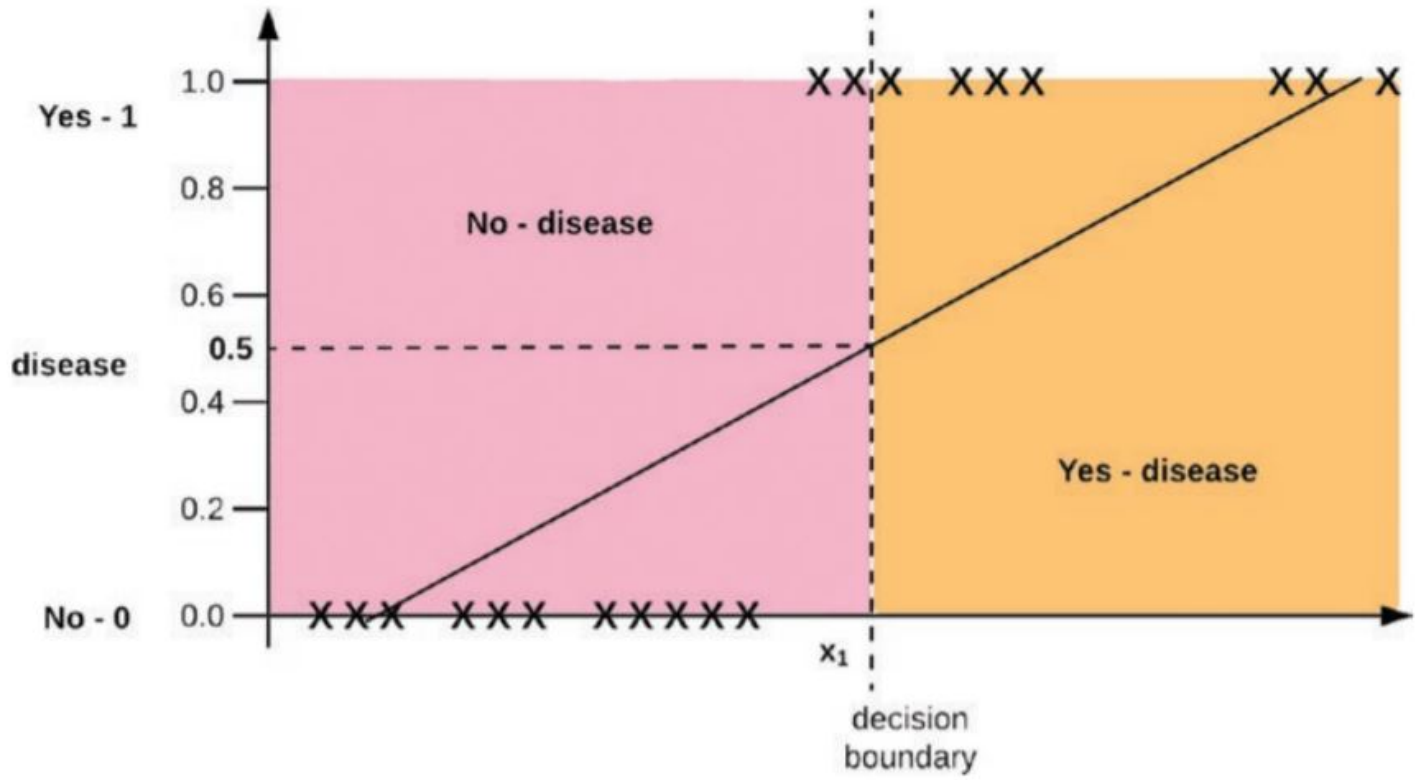
Ver problema 2

Regresión Logística

¿La regresión lineal se podría usar para clasificación?

Regresión Logística

¿La regresión lineal se podría usar para clasificación? En la siguiente imagen podemos ver que no.



Regresión Logística

Para solucionar la falta de precisión que presenta la regresión lineal en clasificación se presenta un cambio de ¿dimensión? añadiendo una **función de activación** luego de la regresión. Aunque hablaremos de funciones de activación más adelante. Se define la función sigmoide como:

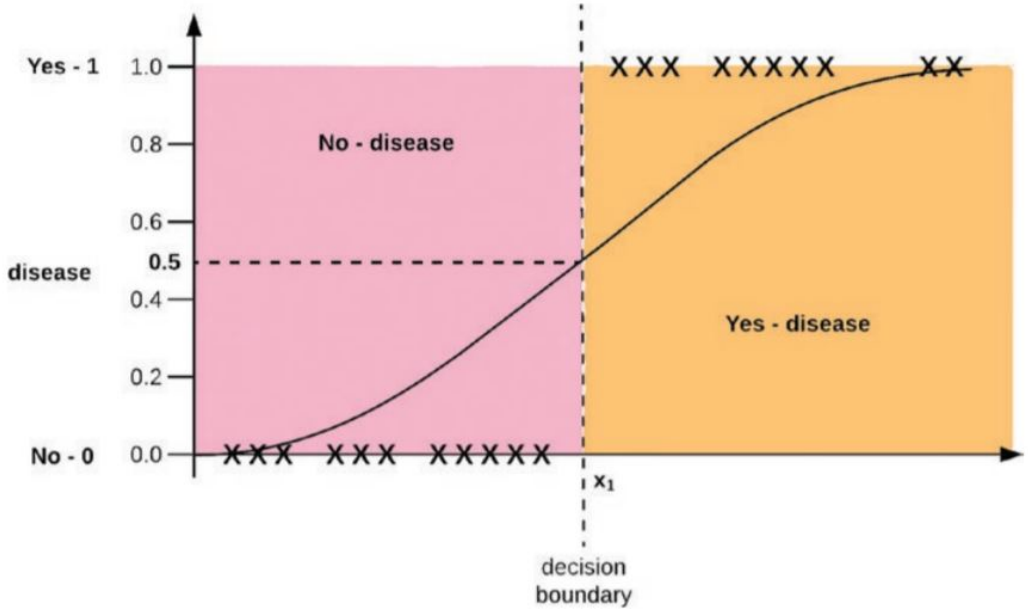
$$h(\hat{y}) = \frac{1}{1 + e^{-\hat{y}}}$$

Donde se cumple que:

$$\hat{y} = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + ...\theta_n x_n$$

Regresión logística

Esto vuelve la regresión lineal, en una función donde sus valores límites son 0 y 1, y donde la probabilidad de que un valor tenga una etiqueta u otra, se comparte en el rango.

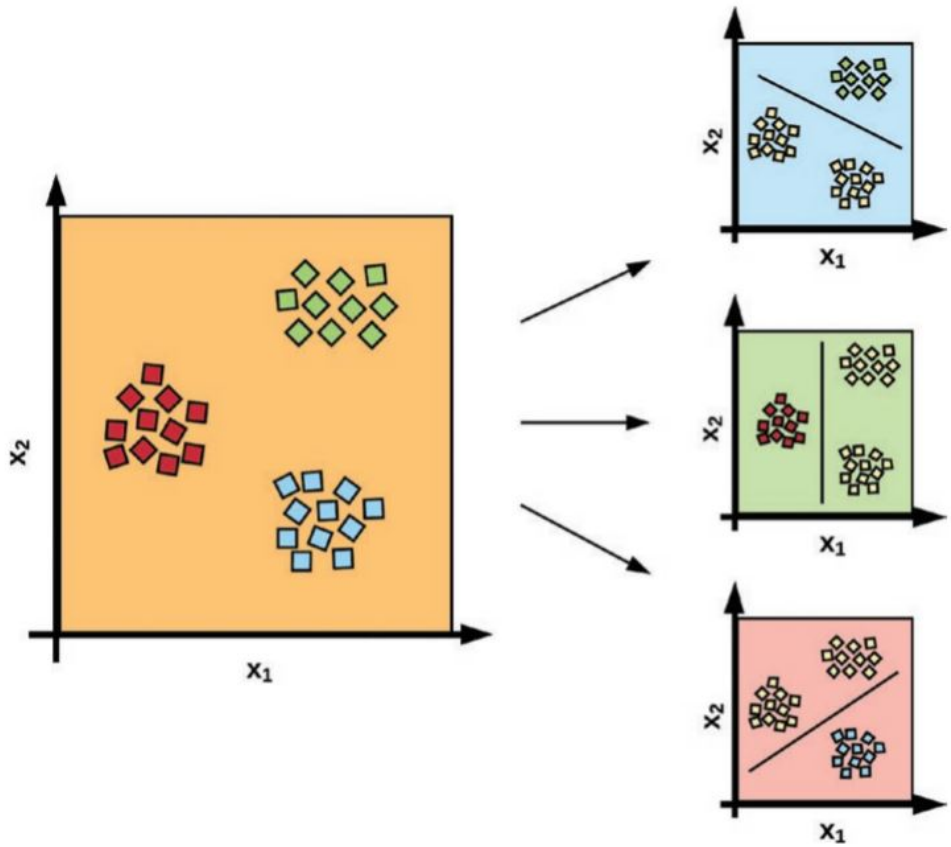


Donde la función de costo viene representado por :

$$J(h(\hat{y}), y) = -\frac{1}{m}\sum_{i}^{m} ylog(h(\hat{y})) + (1-y)log(1-\hat{y})$$

Regresión logística

Para multiclase, se usa el método uno contra todos, y lo que se calcula como predicción es la probabilidad de que esa etiqueta se cumpla para unas características determinada.



Computarización de Regresión logística

Para poder usar la regresión logística importamos de sklearn :

```
from sklearn.linear_model import LogisticRegression
```

Los hiperparámetros que se pueden definir son:

penalty: Regularización que ofrece un cambio en la función de costo para evitar multicolinealidad. Valores : ['l1','l2','elasticnet','none']. Por defecto se encuentra 'l2'.

C: Fuerza de regularización inversa. Debe ser un flotante positivo.

solver: Método de solución del algoritmo de optimización. Valores: ['newton-cg' - ['l2', 'none'], 'lbfgs' - ['l2', 'none'],'liblinear' - ['l1', 'l2'],'sag' - ['l2', 'none'],'saga' - ['elasticnet', 'l1', 'l2', 'none']]

Ver problema 3