Práctica 3

Ajuste de datos usando modelos lineales

28 de mayo de 2021

Aprendizaje Automático

FRANCISCO JAVIER SÁEZ MALDONADO

fjaviersaezm@correo.ugr.es

Índice

Introducción

En esta práctica, trataremos de realizar un estudio completo de un problema en el que se nos presenta un conjunto de datos y nuestro objetivo es seleccionar el mejor predictor lineal para este conjunto de datos dado. Concretamente, estudiaremos dos conjuntos de datos extraidos de la web UCI-Machine Learning Repository.

Utilizaremos uno de ellos para tratar de ajustar un modelo lineal a un problema de regresión, y otro conjunto diferente de datos para ajustar otro modelo lineal a un problema de clasificación multiclase. El objetivo será realizar un estudio de los datos, evitando en todo momento el *data snooping*, y argumentar si se utilizan ciertas técnicas de preprocesado de datos antes de escoger el modelo final.

Trataremos primero el problema de regresión y posteriormente el de clasificación.

1. Regresión

1.1. Estudio del conjunto de datos. Identificación de $\mathcal{X}, \mathcal{Y}, f$.

Lo primero que debemos hacer es realizar una buena comprensión de la información que tenemos sobre los datos para comprender un poco más nuestro problema.

Nuestro primer conjunto de datos, [2], contiene características de ciertos elementos superconductores. Junto con estas características, se nos presenta una *temperatura crítica*, que en [1] lo denominan T_c siendo un problema similar al que vamos a abordar, obtenida para un superconductor que posea estas características. También se nos presenta un archivo en el que se nos dan las fórmulas químicas de los superconductores, pero este archivo no será relevante para nosotros.

Lo primero que nos encontramos acerca de nuestros datos es la siguiente tabla:

Características	Multivariable	Número de instancias	21263
Tipo de características	Reales	Número de atributos	81
Tareas asociadas	Regresión	Valores perdidos	$N\backslash A$

Tabla 1: Datos contenidos en el conjunto de datos Superconductivity.

Esta información nos resulta muy útil, pues obtenemos podemos observar que tenemos 81 atributos para cada una de las 21263 instancias. De aquí podemos obtener que el tamaño del conjunto de datos es bastante amplio, por lo que tendremos un buen conjunto de entrenamiento. Las características que obtenemos son reales, es decir, $x_i \in \mathbb{R}^{81}$. Para completar, vemos que no tenemos valores perdidos, por lo que nos ahorraremos en este caso tener que establecer una técnica para reconstruir estos valores.

Con la información proporcionada podemos decir que:

1. Nuestro conjunto de datos de entrenamiento será
$$\mathcal{X} = \{x_i \in \mathbb{R}^{81}, \text{ con } i = 1, \dots, 21263\}.$$
2. Nuestro conjunto de etiquetas, puesto que no se nos indica ninguna restricción sobre las temperaturas, podemos asumir que es:
$$\mathcal{Y} = \{y_i \in \mathbb{R}, \text{ con } i = 1, \dots, 21263\}.$$
3. Por último, nuestra función $f : \mathcal{X} \to \mathcal{Y}$ que asigne a cada vector de características una temperatura crítica.

Hay que anunciar que los siguientes gráficos de visualizado de datos se han realizado posteriormente a realizar la separación en conjuntos de *train* y *test* de nuestro conjunto de datos, para evitar en todo momento el *data snooping*.

Dibujamos ahora un gráfico en el que mostramos el diagrama de caja de los posibles valores que toma la temperatura T_c:

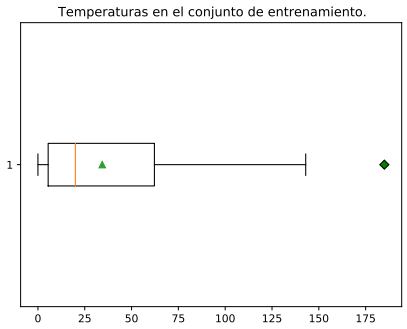

Figura 1: Diagrama de caja de las temperaturas T_c en el conjunto de entrenamiento.

Podemos ver que tenemos una variabilidad razonablemente amplia en los valores que toma f. Sin embargo, se observa que la mayoría de los valores de f están concentrados en el intervalo $[0, 50]$. Debido a que la regresión la estamos haciendo por una función lineal y continua, si los valores de la función f son similares, nos indica que la distancia entre los puntos x_i de nuestro conjunto de entrenamiento es relativamente pequeña, con lo que podemos conseguir un buen ajuste local de nuestro modelo, pero al no tener tantos puntos x_i con valores algo más distantes, podría ocurrir que el ajuste sea peor cuando la distancia entre los nuevos puntos y los puntos del conjunto de entrenamiento. Como podemos ver, tenemos un dato que se aleja mucho de 1.5 por el rango intercuartílico, que es lo que representan los *bigotes* del diagrama de caja. Es por ello que podemos decir que este punto es posiblemente un *outlier*.

En cuanto a los valores de nuestros datos, si tomamos el primer elemento y calculamos la desviación típica σ y el resultado que obtenemos es:

```
Standard deviation of the first element of the dataset 2747.992
```

Por lo que obtenemos que claramente los valores de los diferentes atributos no están en el mismo rango de escala. Es por ello que previamente al entrenamiento realizaremos una estandarización por atributos de nuestro conjunto de datos.

1.2. La clase de funciones $\mathcal{H}$

La clase de funciones a utilizar en este caso viene impuesta por el enunciado del ejercicio. En este caso, utilizaremos la clase de las funciones lineales:

$$\mathcal{H} = \{h(x) = w^t x \ : \ w \in R^{n+1}\}.$$

Tenemos que destacar que, aunque se podría plantear aplicar funciones no lineales a las características dadas (ejemplo $\phi : \mathbb{R}^d \to \mathbb{R}^k$ dada por $\phi(x) = (1, x_1, \cdots, x_n, x_1x_1, x_1x_2, \cdots, x_dx_d)$, no se hace en este caso pues estaríamos añadiendo ua complejidad a la clase de funciones sin saber realmente si esto sería útil de cara a la generalización o no. Es por ello que, al no tener en la información sobre los datos que se nos proporciona ningún motivo para hacerlo, se decide no aplicar ninguna transformación de este estilo a los datos.

1.3. Conjuntos de entrenamiento, validación y test

En este problema, tenemos un conjunto suficientemente grande de datos, que no viene previamente separado en subconjuntos de entrenamiento y test. En concreto, hemos mencionado ya que $N = 21263$ datos. Es por ello que se ha decidido usar un conjunto de entrenamiento con el 70 % de los datos, y dejar el 30 % para el conjunto de test. Para ello nos aprovechamos de la función `train\_test\_split` de `sklearn`.

Para elegir en nuestro conjunto de hipótesis antes de evaluar la función elegida en el conjunto de test, utilizaremos la conocida técnica **K-Fold Cross Validation**.

Esta técnica consiste en, si llamamos X_{train} al conjunto de entrenamiento, realizar los siguientes pasos:

Algorithm 1 K-Fold Cross Validation

1: $Vector\_Eouts = []$
2: **for** i = 1,...,k **do**
3: $\quad Datos_{val} \leftarrow Particion_i$
4: $\quad Datos_{train} \leftarrow X_{train} \backslash Particion_i$
5: $\quad Pesos \leftarrow$ Entrenamiento en $Datos_{train}$
6: $\quad Vector_Eouts \leftarrow Error(Pesos, Datos_{val})$
7: **end for**
8: **return** Average $Vector\_Eouts$

Describiéndolo en pocas palabras, diríamos que partimos el conjunto de entrenamiento en k subconjuntos y en cada iteración entrenamos nuestro modelo con $k-1$ particiones y calculamos el error "fuera de la muestra" usando la partición restante. Hacemos eso con todas las particiones y devolvemos una media de los errores fuera de la muestra que hemos obtenido. Les decimos errores fuera de la muestra porque se calculan usando puntos no usados **en esa iteración del entrenamiento**, aunque formen parte del conjunto de datos.

Obteniendo el error medio en validación usando este tipo de validación cruzada, podemos hacernos una idea de cómo de bueno (en media) será nuestro modelo fuera de la muestra. De hecho, sabemos que:

Teorema.- El error de validación cruzada E_{cv} es un estimador insesgado de la esperanza del error fuera de la muestra en conjuntos de datos de tamaño $N-1$.

Usualmente, $K-$Fold cross validation se utilza para estimar los parámetros con los que se entrenará nuestro modelo final, y una vez que se han estimado, se vuelve a entrenar el modelo usando todos los datos de entrenamiento disponible para tener un modelo entrenado con un conjunto de datos lo mayor posible.

2. Preprocesado de datos

Entramos en una de las fases más importantes de nuestro problema. Recordamos que tenemos 81 varibles para cada dato, y que como hemos visto, no están en la misma escala. Por ello, las normalizamos por columnas.

Referencias

[1] Kam Hamidieh. "A Data-Driven Statistical Model for Predicting the Critical Temperature of a Superconductor". en. En: *arXiv:1803.10260 [stat]* (oct. de 2018). arXiv: 1803.10260. URL: http://arxiv.org/abs/1803.10260 (visitado 25-05-2021).

[2] *Superconductivty Data Data Set*. URL: https://archive.ics.uci.edu/ml/datasets/Superconductivty+Data (visitado 12-10-2018).