Bases de Datos 2
Práctica 5: Hadoop y HBase

Jordi Bernad , Sergio Ilarri

1. Objetivos de la práctica

1. Familiarizarse con el paradigma paralelo MapReduce y su implementación Hadoop.
2. Familiarizarse con la base de datos NoSQL HBase.
3. Aprender nociones básicas del problema de clasificación en minería de datos.
4. Realizar una pequeña aplicación de ejemplo.

2. Introducción

El fichero *data.csv*, adjunto al material de esta práctica, contiene la información de 1600 características de 10.000 usuarios. Según estas características los usuarios se clasifican en dos grupos diferentes, A o B. El fichero *data.csv* es un fichero en formato csv (comma separated values) donde cada línea contiene la información de un usuario con los datos: código único del usuario formado por letras mayúsculas y números; clasificación del usuario en A o B; 1600 valores que se corresponden con las 1600 características del usuario donde aparecerá un 0.0 si el usuario no tiene esa característica, y 1.0 si la posee. Cada usuario tiene alrededor de 200 características positivas, esto es, en cada línea del fichero, entre los 1600 valores que se corresponden con las características del usuario hay unos 200 valores 1.0 y el resto son 0.0.

En base a la información que tenemos, deseamos crear un modelo predictivo para poder clasificar automáticamente nuevos posibles usuarios en el grupo A o B. Las acciones que se realizarán serán las siguientes:

- Preparar y almacenar la información de todos los usuarios en una base de datos HBase (Sección 3).
- Utilizar map/reduce para generar un modelo predictivo que nos permita clasificar automáticamente nuevos usuarios (Sección 4).
- Comprobar la bondad del modelo contando el número de usuarios bien clasificados para los datos guardados en el fichero *dataTest.csv* adjunto al material de la práctica (Sección 5).

En las siguientes secciones se explicarán los pasos necesarios para preparar el entorno, y qué es y cómo obtener el modelo predictivo.

3. PREPARACIÓN DEL ENTORNO

Antes de empezar a crear el modelo predictivo es necesario instalar y configurar los siguientes elementos:

1. Instalar Hadoop en modo pseudo-distribuido (versión hadoop-3.2.1).
2. Instalar HBase en modo pseudo-distribuido (versión hbase-1.4.13).
3. Opcionalmente, si se trabaja con el IDE Eclipse, instalar el plugin HDT (Hadoop Development Tools) en Eclipse.

El siguiente paso será crear, con el shell de HBase, la base de datos vacía para almacenar la información de todos los usuarios. Los datos de los usuarios se almacenarán en una tabla llamada `Clientes`. Esta tabla tendrá dos familias de columnas llamadas `cli` y `car`.

Una vez creada la base de datos, se implementará en el fichero `Importacion.java`, adjunto al material de este enunciado, una clase para cargar los datos del fichero *data.csv* a la tabla `Clientes` utilizando map/reduce.

Se va a trabajar en modo pseudodistribuido por sencillez. Todos los programas que se desarrollan en esta práctica tendrían un comportamiento más eficiente en un entorno completamente distribuido en un clúster.

TRABAJO PARA EL ALUMNADO. Completar el fichero `Importacion.java` adjunto al enunciado.

4. MODELO PREDICTIVO

El modelo predictivo que vamos a utilizar está basado en una técnica de minería de datos llamada *Regresión Logística*. Aunque está fuera del alcance del curso de Bases de Datos 2 explicar con detalle la regresión logística, sí os daremos las claves necesarias para que podáis programar el modelo.

La regresión logística es un método de clasificación supervisado, paramétrico, y basado en técnicas estadísticas. Expliquemos con un poco de detalle qué significado tienen las palabras modelo, paramétrico y supervisado. Pero antes veamos la notación que utilizaremos para explicar éstos y otros conceptos.

En nuestra base de datos disponemos de n datos (cada uno de ellos representando un usuario) de los que conocemos su clasificación (en nuestro caso, $n \approx 10.000$). Como sabemos, cada uno de estos datos tendrá 1600 características, desde la característica 1 a la 1600. De esta forma, cada dato lo podemos ver como un vector de 1600 componentes, una componente por cada característica. Para simplificar la notación de ciertas operaciones que más adelante aparecerán, supondremos que los datos tienen una característica ficticia más (la característica 0) que siempre tendrá un valor igual a 1. Por tanto, el dato almacenado en la i-ésima posición de la base de datos lo veremos como un vector de 1601 componentes donde la pri-

mera característica es igual a 1:

$$\mathbf{x}^i = (1, x_1^i, x_2^i, \ldots, x_{1600}^i)$$

A cada dato $\mathbf{x}^i$ de la base de datos le asociaremos un valor y^i que será igual a cero si $\mathbf{x}^i$ está clasificado en el grupo A, e igual a 1 si esta clasificado como B. Así, podemos asumir que nuestra base de datos en notación vectorial es un conjunto de n datos:

$$(\mathbf{x}^1, y^1), (\mathbf{x}^2, y^2), \ldots, (\mathbf{x}^n, y^n)$$

¿Qué significa modelo? Un modelo es la representación de una realidad compleja mediante alguna formulación matemática. En el caso de la regresión logística se realiza la siguiente suposición: ¿podemos encontrar, por una de esas casualidades que tiene la vida, 1601 valores reales, $\theta_0, \ldots \theta_{1600}$, de forma que dado un dato cualquiera $\mathbf{x} = (1, x_1, x_2, \ldots, x_{1600})$ (recordad que a las 1600 características propias del dato, $x_1, \ldots, x_{1600}$, le añadimos un 1 delante) podemos clasificar el dato según la siguiente función, $f(\mathbf{x})$?

$$f(\mathbf{x}) = \begin{cases} A, & \text{si } \dfrac{1}{1+e^{-\theta_0 * 1 - \theta_1 * x_1 - \cdots \theta_{1600} * x_{1600}}} < 0.5 \\ B, & \text{si } \dfrac{1}{1+e^{-\theta_0 * 1 - \theta_1 * x_1 - \cdots \theta_{1600} * x_{1600}}} \geq 0.5 \end{cases} \tag{4.1}$$

En otras palabras: denotemos por $g(z) = 1/(1+e^{-z})$ y por $\boldsymbol{\theta} = (\theta_0, \ldots \theta_{1600})$. Notemos que $\theta_0 * 1 + \theta_1 * x_1 + \cdots \theta_{1600} * x_{1600}$ no es más que el producto escalar de los vectores $\boldsymbol{\theta}$ y $\mathbf{x}$. Nuestro modelo nos dice que si $g(\boldsymbol{\theta}\mathbf{x})$ es menor que 0.5, $\mathbf{x}$ lo clasificamos como A; y si es mayor, lo clasificamos como B. Observad que la función $g(z)$ siempre nos devuelve un valor en el intervalo $(0,1)$; $g(z)$ es la llamada *función logística*, de ahí el nombre de regresión logística. De alguna forma, al calcular $g(\boldsymbol{\theta}\mathbf{x})$ estamos hallando la probabilidad de pertenencia al grupo B. Por ejemplo, si al hallar $g(\boldsymbol{\theta}\mathbf{x})$ obtenemos 0.7, significará que hay un 70 % de probabilidades de que el dato $\mathbf{x}$ pertenezca al grupo B, y por tanto, un 30 % de probabilidades de que pertenezca al grupo A, en cuyo caso, clasificaremos el dato $\mathbf{x}$ en el grupo B. La función $f(\mathbf{x})$ de la Ecuación 4.1 es nuestro modelo.

¿Qué significa paramétrico? El modelo es paramétrico porque depende de unos parámetros θ_i desconocidos. Si conseguimos hallar el valor de esos parámetros, nos permitirá clasificar nuevos datos. En nuestro modelo tendremos que ajustar el valor de 1601 parámetros, uno más que el número de características que tengamos. Es decir, buscamos los 1601 valores del vector:

$$\boldsymbol{\theta} = (\theta_0, \theta_1, \ldots, \theta_{1600})$$

La búsqueda de estos valores empezará suponiendo que $\boldsymbol{\theta}$ es el vector nulo, esto es, $\theta_j = 0$ para todo $0 \leq j \leq 1600$, y en base a un algoritmo que explicamos más adelante, encontraremos nuevos valores de $\boldsymbol{\theta}$ que mejoren el modelo.

¿Qué significa supervisado? La pregunta ahora es cómo hallar los valores $\boldsymbol{\theta}$. Nuestro método es supervisado porque utilizaremos un conjunto de datos que sabemos bien clasificados (en nuestro problema, los datos almacenados en HBase del fichero *data.csv*) para hallar los valores de los parámetros $\boldsymbol{\theta}$, con la esperanza de que si el modelo clasifica bien los datos conocidos será capaz de clasificar también bien nuevos datos. Nuestro objetivo será buscar valores para los parámetros $\boldsymbol{\theta}$ de tal forma que cada dato $\mathbf{x}^i$ del que tenemos conocida su clasificación cumpla la siguiente propiedad: *si* $\mathbf{x}^i$ *está clasificado como A, entonces* $g(\boldsymbol{\theta}\mathbf{x}^i) \approx 0$,

y si está clasificado como B, $g(\boldsymbol{\theta}\mathbf{x}^i) \approx 1$. Es obvio que si conseguimos valores $\boldsymbol{\theta}$ cumpliendo la propiedad anterior, al menos nuestro modelo será capaz de clasificar bien los datos conocidos. Esta propiedad la podemos expresar buscando valores para $\boldsymbol{\theta}$ que maximicen la función:

$$L(\boldsymbol{\theta}) = \prod_{i=1}^{n} g(\boldsymbol{\theta}\mathbf{x}^i)^{y^i} (1 - g(\boldsymbol{\theta}\mathbf{x}^i))^{1-y^i}$$

o, aplicando el logaritmo, los valores $\boldsymbol{\theta}$ que maximicen la función:

$$J(\boldsymbol{\theta}) = \log L(\boldsymbol{\theta}) = \sum_{y^i=1} y^i \log(g(\boldsymbol{\theta}\mathbf{x}^i)) + \sum_{y^i=0} (1 - y^i) \log(1 - g(\boldsymbol{\theta}\mathbf{x}^i)) \tag{4.2}$$

Para buscar los valores $\boldsymbol{\theta}$ que maximicen la función utilizaréis el algoritmo de gradiende descendente[1]:

```
1 θ_act = (0,0,...,0)
2 while (no se cumpla criterio de convergencia) {
3     //vector de 1601 posiciones, grad_j es la posición j de grad
4     grad = (0,0,...,0)
5     for j ← 0 to 1600 {
6         grad_j = Σ_{i=1}^{n} (y^i − g(θ_act · x^i)) * x_j^i
7     }
8     θ_act = θ_act + α * grad
9 }
```

Empezamos definiendo un vector $\boldsymbol{\theta}_{act}$ de 1601 ceros (línea 1). Entramos en un bucle donde se hallarán nuevos valores de $\boldsymbol{\theta}_{act}$ (línea 2). Definimos el vector **grad** de 1601 posiciones (línea 4) para guardar el valor del gradiente de la función $J(\boldsymbol{\theta})$ (Ecuación 4.2) en el punto $\boldsymbol{\theta}_{act}$. Hallamos cada una de las 1601 componentes de **grad** (líneas 5 y 6). Actualizamos $\boldsymbol{\theta}_{act}$ con los valores de $\boldsymbol{\theta}_{act} + \alpha * \mathbf{grad}$ (línea 8), donde α es un número positivo real pequeño. Para nosotros el criterio de convergencia será simple: haremos 20 veces el bucle; y el valor de α será 0.01.

Observad que los datos conocidos solo se utilizan para hallar $grad_j$, el valor de la componente j-ésima del vector **grad**. El valor $grad_j$ es la suma para todos los datos y clasificación conocidas, $(\mathbf{x}^i, y^i)$, $1 \leq i \leq n$:

$$grad_j = \sum_{i=1}^{n} (y^i - g(\boldsymbol{\theta}_{act} \cdot \mathbf{x}^i)) * x_j^i \tag{4.3}$$

donde $g(\boldsymbol{\theta}_{act}\mathbf{x}^i)$ es el valor de aplicar la función $g(z)$ al producto escalar $\boldsymbol{\theta}_{act}\mathbf{x}^i$; y^i es igual a 0 o 1, dependiendo de si el dato $\mathbf{x}^i$ está clasificado como A o B, respectivamente; y x_j^i es el valor de la componente j-ésima del dato $\mathbf{x}^i$.

Como se dice al inicio del enunciado de la práctica, se debe utilizar map/reduce para ajustar el modelo predictivo. Ajustar el modelo en la regresión logística es hallar los valores $\boldsymbol{\theta}$. Si analizáis el algoritmo que calcula $\boldsymbol{\theta}$, notaréis que el único sitio donde merece la pena aplicar map/reduce es en el cálculo del sumatorio que aparece en la Ecuación 4.3. Se trata de sumar n cantidades, donde n puede ser un número enorme. Aunque en nuestro caso n sea un número modesto, en un escenario más realista se pueden estar sumando miles de millones de valores en bases de datos con terabytes o petabytes de datos.

[1] Para consultar más detalles del algoritmo de gradiente descente: https://en.wikipedia.org/wiki/Gradient_descent

TRABAJO PARA EL ALUMNADO. Completar el fichero `Logistica.java` adjunto al enunciado.

5. VALIDAR EL MODELO

Finalmente, se ha de implementar código para comprobar la capacidad de clasificación del modelo con respecto a datos que no han sido utilizados para hallar los valores $\boldsymbol{\theta}$. Estos datos se encuentran en el fichero *dataTest.csv* que contiene la información de 500 usuarios. Se mostrará por pantalla el porcentaje de usuarios correctamente clasificados. Se deberían obtener resultados de al menos el 90 % de usuarios bien clasificados. Además se mostrará también la llamada *matriz de confusión*: es una matriz donde las filas representan la clasificación real y las columnas la clasificación predicha por el modelo; en la posición (f,c) de la matriz se almacena el número de usuarios que sabemos que son f (= A o B) y el modelo ha predicho como c (= A o B). Por ejemplo, la matriz de confusión:

```
    A    B
A   240  10
B   12   238
```

nos dice que hay 240 usuarios que son A y el modelo predice A; 10 usuarios que son A y el modelo predice B; 12 usuarios que son B y se predicen como A; y 238 usuarios que son B y el modelo predice B.

TRABAJO PARA EL ALUMNADO. Completar el fichero `Validacion.java` adjunto.

6. ENTREGA DE LA PRÁCTICA

Se debe entregar un fichero zip denominado *p5-<nia>.zip* (donde *<nia>* representa el NIA del coordinador/responsable del grupo) con el siguiente contenido:

1. Fichero *autores.txt*: contendrá el nombre y apellidos de los autores de la práctica y sus NIAs.
2. Directorio *fuentes*: contendrá todo el código desarrollado para la práctica:
   a) Fichero *Importacion.java* con el código necesario para importar los datos del fichero *data.csv* a una base de datos HBase.
   b) Fichero *Logistica.java* con el código que genere los valores de los parámetros $\boldsymbol{\theta}$ del modelo.
   c) Fichero *Validacion.java* con el código que muestre por pantalla el porcentaje de usuarios correctamente clasificados del fichero *dataTest.csv* y la matriz de confusión.
3. Fichero *memoria-p5.pdf*, conteniendo la memoria de la práctica, donde se explique detalladamente el trabajo realizado:

a) Instalación de Hadoop y HBase, problemas encontrados y cómo se han solucionado.

b) Pseudo-código de las funciones map y reduce utilizadas.

c) Todos aquellos detalles e instrucciones necesarias para resolver la práctica.

d) Conclusiones personales sobre el uso de Hadoop/HBase para resolver el problema clasificación.

Instrucciones de envío. El fichero anterior se someterá a través de Moodle (se habilitará una opción a tal efecto) utilizando la cuenta del coordinador/responsable del grupo.

Fecha límite de entrega. La fecha límite de entrega será el **10 de junio de 2020** (incluido).