Minería de Datos

UTDT

Master in Management + Analytics

Sesión 4

¿Qué vimos hasta ahora?

- Qué es la minería de datos.
- Distintos tipos de aprendizaje automático.
- Aprendizaje supervisado.
- Naïve bayes.
- K-nearest neighbors.
- Noción de overfitting & underfitting
- Selección de modelos
- Árboles de decisión
- Introducción a Caret
- Ingeniería de atributos (peligro de data leaking)
- Métricas de performance de clasificadores

One-hot-encoding - Matrices ralas

Sample	Category
1	Human
2	Human
3	Penguin
4	Octopus
5	Alien
6	Octopus
7	Alien

Human	Penguin	Octopus	Alien
1	0	0	0
1	0	0	0
0	1	0	0
0	0	1	0
0	0	0	1
0	0	1	0
0	0	0	1

¿Qué opciones tenemos si hay un NA?

¿Tendremos problemas si la variable tiene muchos niveles?

One-hot-encoding - Matrices ralas

Al momento de generar datos para introducir a un modelo predictivo, es común que muchos de los valores sean 0.

Veamos cómo afecta esto al uso de memoria RAM de nuestras computadoras y cómo ésta se puede aprovechar mejor (fuente).

```
library('Matrix')

M1 <- matrix(0, nrow = 1000, ncol = 1000)
M2 <- Matrix(0, nrow = 1000, ncol = 1000, sparse = TRUE)

format(object.size(M1), unit = "Mb")
format(object.size(M2), unit = "Mb")

M1[500, 500] <- 1
M2[500, 500] <- 1

format(object.size(M1), unit = "Mb")
format(object.size(M2), unit = "Mb")
```

One-hot-encoding - Matrices ralas

Usar matrices “esparsas” (ralas) tiene sentido cuando la gran mayoría de los valores son 0, pero puede ser contraproducente en caso contrario.

En este sentido no es una herramienta que convenga utilizar siempre.

```
M1 <- matrix(rnorm(1000000), nrow = 1000, ncol = 1000)
M2 <- Matrix(rnorm(1000000), nrow = 1000, ncol = 1000, sparse = TRUE)

format(object.size(M1), units= "Mb")
format(object.size(M2), units= "Mb")
```

One-hot-encoding - Matrices ralas

Uno puede generar matrices "esparsas" indicando la posición y valor de los elementos no nulos.

```
M1 <- sparseMatrix(i=c(2, 20), j=c(3, 5), x=c(4, 8))
class(M1)
print(M1)

M2 <- sparseMatrix(i=c(2, 20), j=c(3, 5), x=c(4, 8), dims = c(25, 10))
print(M2)

M2 <- sparseMatrix(i=c(2, 20), j=c(3, 5), x=c(4, 8), dims = c(25000, 10000))
print(M2)

format(object.size(M1), units= "Mb")
format(object.size(M2), units= "Mb")

rm(M1, M2)
gc()
```

One-hot-encoding - Matrices ralas

Usando el segundo bloque de código, probemos hacer one-hot-encoding con matrices “esparsas”.

Boosting

Idea:

1. Entreno un árbol de decisión.
2. Veo en qué observaciones de entrenamiento predijo mal.
3. Construyo un segundo árbol de decisión que se enfoque en aquellas observaciones que el primero predijo mal.
4. Tomo como mi predicción final alguna combinación de las predicciones de cada árbol.

¿Tiene sentido?

¿Podría seguir con un tercer árbol?

Boosting

Boosting hace uso de este idea. A los modelos que combinan las predicciones de modelos más pequeños (o modelos base) se los conoce como modelos de ensamble (en Machine Learning van a ver con más detalle esto).

La idea de boosting se puede aplicar a diferentes modelos de clasificación y regresión.

Vamos a enfocarnos en el caso de árboles de decisión aplicados a regresión; donde cada árbol es construido de manera secuencial, usando información de los árboles previos.

Boosting

Algorithm 8.2 *Boosting for Regression Trees*

1. Set $\hat{f}(x) = 0$ and $r_i = y_i$ for all i in the training set.

2. For $b = 1, 2, \ldots, B$, repeat:

   (a) Fit a tree $\hat{f}^b$ with d splits ($d+1$ terminal nodes) to the training data (X, r).

   (b) Update $\hat{f}$ by adding in a shrunken version of the new tree:

   $$\hat{f}(x) \leftarrow \hat{f}(x) + \lambda \hat{f}^b(x). \tag{8.10}$$

   (c) Update the residuals,

   $$r_i \leftarrow r_i - \lambda \hat{f}^b(x_i). \tag{8.11}$$

3. Output the boosted model,

$$\hat{f}(x) = \sum_{b=1}^{B} \lambda \hat{f}^b(x). \tag{8.12}$$

Boosting

La idea es que el algoritmo de boosting aprende lento. Cada modelo predice la parte no predicha hasta ese momento (pero de a poco).

Detalles:

- Cada árbol suele ser pequeño y con pocos cortes.
- El hiperparámetro λ (learning rate) retarda el aprendizaje.
- Cada árbol nuevo adicional que se entrene dependerá de lo que los anteriores predijeron (difícil de paralelizar).

Se puede adaptar para clasificación, pero es matemáticamente más complejo. De hecho existe una generalización que se llama Gradient Boosting Machine (GBM) que permite hacer clasificación, regresión y ranking.

Boosting

xgboost es una mejor formalización y una implementación inteligente de GBM.

Es un algoritmo importantísimo en la academia e industria (otras implementaciones populares son lightGBM y CatBoost).

Muy útil en problemas tradicionales (predecir con datos como los del TP), no tanto en problemas más "raros" (ej: imágenes, audio, texto, etc.).

Boosting

En xgboost se suelen modificar 7 hiperparámetros (link):

- nrounds: número de árboles.
- max_depth: profundidad máxima de los árboles.
- eta: es lo que antes presentamos como λ.
- gamma: mínima reducción del error para generar un corte.
- colsample_bytree: variables a muestrear y considerar en cada árbol.
- min_child_weight: mínima cantidad de observaciones en los hijos para considerar un corte.
- subsample: muestreo de observaciones a considerar en cada árbol.

xgboost puede hacer overfitting! En general los parámetros en rojo, a medida que son mayores, más tienden a hacer overfitting, mientras que los azules mientras más bajos son más tienden a hacer overfitting.

Probémoslo en el tercer bloque de código.

Aprendizaje no supervisado

Hasta ahora vimos lo que se conoce como aprendizaje supervisado. En lo que resta del curso nos enfocaremos en tópicos de aprendizaje no supervisado.

De ahora en adelante sólo tendremos valores conocidos (X) y nuestro objetivo será descubrir patrones interesantes detrás de los datos.

Por ejemplo:

- Existe alguna forma informativa de visualizar los datos.
- Se pueden detectar subgrupos similares de observaciones.
- Se pueden detectar subgrupos de variables relacionadas.

Aprendizaje no supervisado

El aprendizaje no supervisado es más desafiante que el aprendizaje supervisado.

Esto se debe a que es algo más subjetivo, principalmente porque no se dispone de una métrica clara a optimizar.

Muchas veces forma una etapa del análisis exploratorio de datos (o del preprocesamiento de datos).

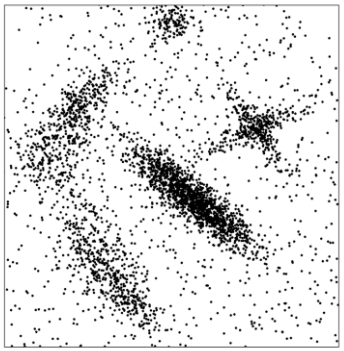
Clustering

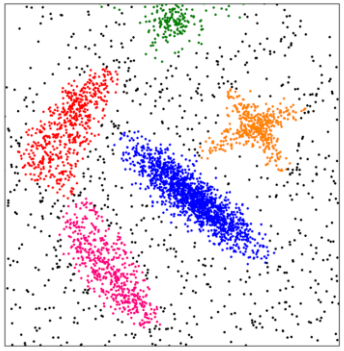
Clustering se refiere a una serie de técnicas cuyo objetivo es encontrar subgrupos o "*clusters*" en un conjunto de datos.

La idea es particionar los datos de manera que:

1. Las observaciones que pertenecen a un grupo sean similares entre ellas.
2. Las observaciones de grupos distintos sean distintas entre ellas.

Esto plantea el "problema" de definir cuando dos observaciones son similares o distintas entre sí (algo que en gran medida puede depender del dominio en donde se esté trabajando.)





Clustering

Existen múltiples familias de algoritmos de clustering:

1. *Partitioning*
2. *Hierarchical*
3. *Density-based*
4. *Grid-based*
5. *Model-based*

Nosotros veremos los dos algoritmos más tradicionales:

- *K-means clustering* (partitioning)

- *Hierarchical*

K-means clustering

Divide al conjunto de datos en K subconjuntos distintos sin solapamiento. Uno debe fijar el valor de K antes de correr el algoritmo de K medias.

Los clusters deben cumplir las siguientes condiciones:

1. $C_1 \cup C_2 \cup \ldots \cup C_K = \{1, \ldots, n\}$. In other words, each observation belongs to at least one of the K clusters.

2. $C_k \cap C_{k'} = \emptyset$ for all $k \neq k'$. In other words, the clusters are non-overlapping: no observation belongs to more than one cluster.

K-means clustering

K-means asume que una buena asignación es aquella que dado un valor de K minimiza lo más posible la variabilidad intra cluster (*within-cluster variation*).

Si $W(C_j)$ es una medida que indica cuánto las observaciones de un cluster *j* difieren entre sí. El problema se puede escribir como:

$$\underset{C_1,\ldots,C_K}{\text{minimize}} \left\{ \sum_{k=1}^{K} W(C_k) \right\}$$

Si uno usa la distancia euclídea (al cuadrado) como medida de disimilaridad $W(C_j)$ puede escribirse de la siguiente manera:

$$W(C_k) = \frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2$$

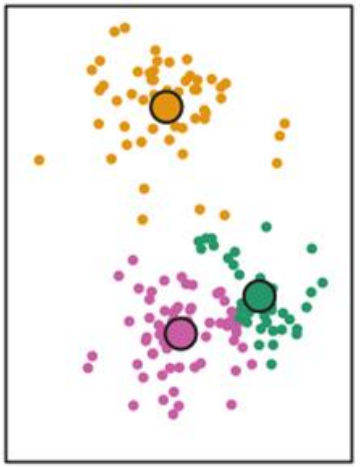
K-means clustering

De esta forma el problema se puede reescribir como:

$$\underset{C_1,\ldots,C_K}{\text{minimize}} \left\{ \sum_{k=1}^{K} \frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2 \right\}$$

En donde *W(C_j)* se puede expresar como la distancia a un centroide:

$$\frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2 = 2 \sum_{i \in C_k} \sum_{j=1}^{p} (x_{ij} - \bar{x}_{kj})^2$$



K-means clustering

Algoritmo de K-medias:

Algorithm 10.1 *K-Means Clustering*

1. Randomly assign a number, from 1 to K, to each of the observations. These serve as initial cluster assignments for the observations.

2. Iterate until the cluster assignments stop changing:

   (a) For each of the K clusters, compute the cluster *centroid*. The kth cluster centroid is the vector of the p feature means for the observations in the kth cluster.

   (b) Assign each observation to the cluster whose centroid is closest (where *closest* is defined using Euclidean distance).

K-means clustering

Algoritmo de K-medias:

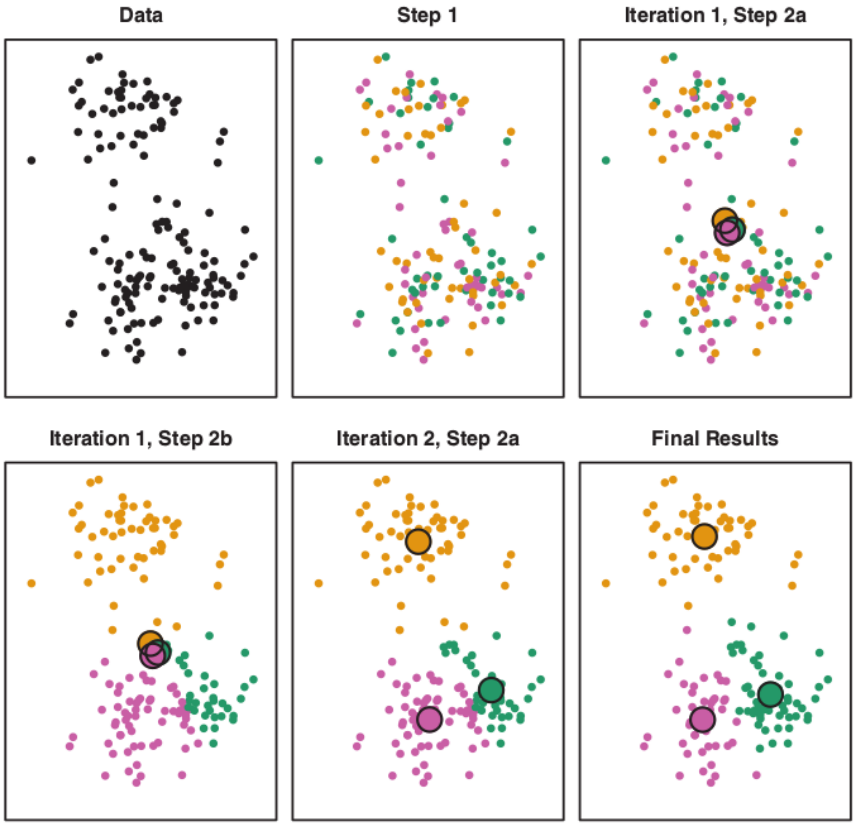
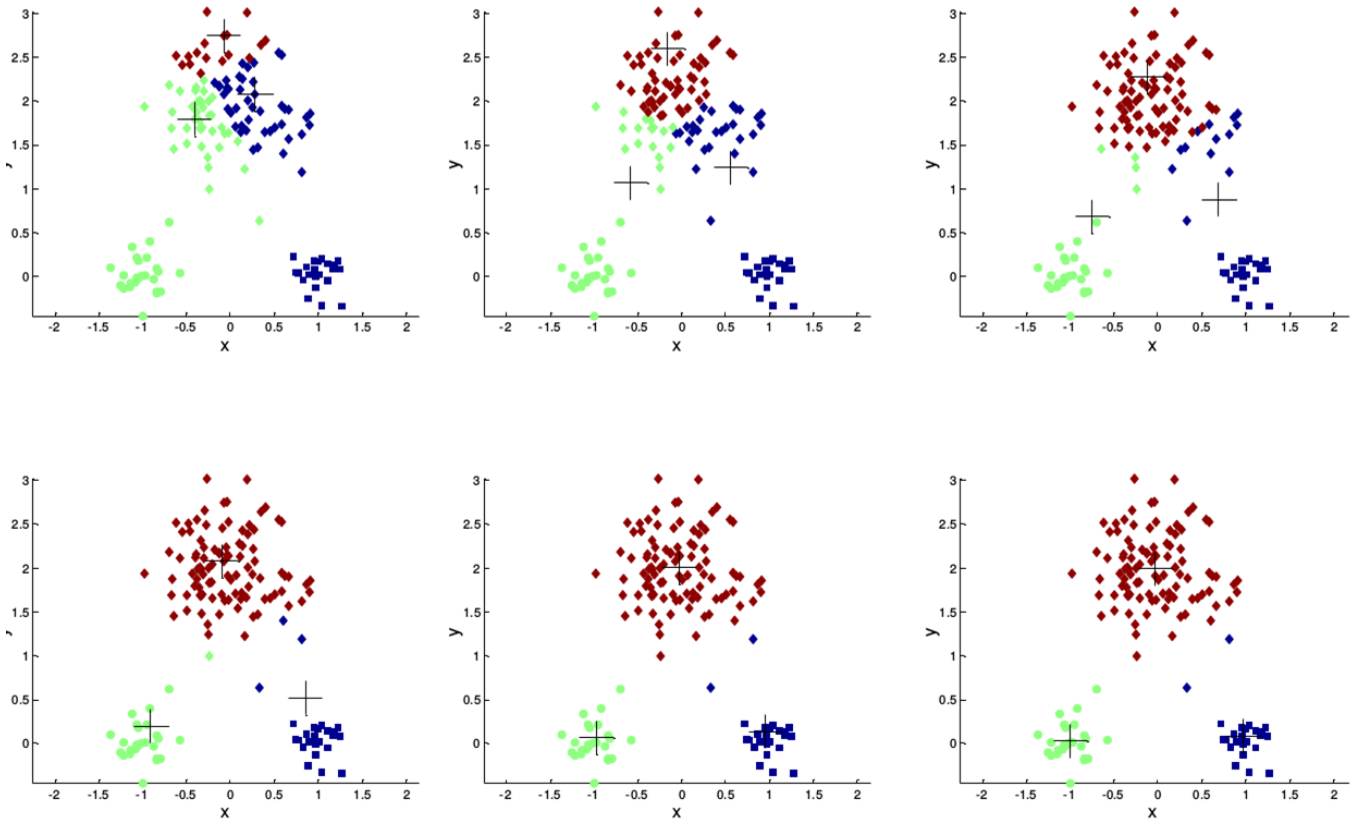


FIGURE 10.6. *The progress of the K-means algorithm on the example of Fig-ure 10.5 with K=3.* Top left: *the observations are shown.* Top center: *in Step 1 of the algorithm, each observation is randomly assigned to a cluster.* Top right: *in Step 2(a), the cluster centroids are computed. These are shown as large col-ored disks. Initially the centroids are almost completely overlapping because the initial cluster assignments were chosen at random.* Bottom left: *in Step 2(b), each observation is assigned to the nearest centroid.* Bottom center: *Step 2(a) is once again performed, leading to new cluster centroids.* Bottom right: *the results obtained after ten iterations.*

K-means clustering

Algoritmo de K-medias (otro ejemplo visual):



K-means clustering

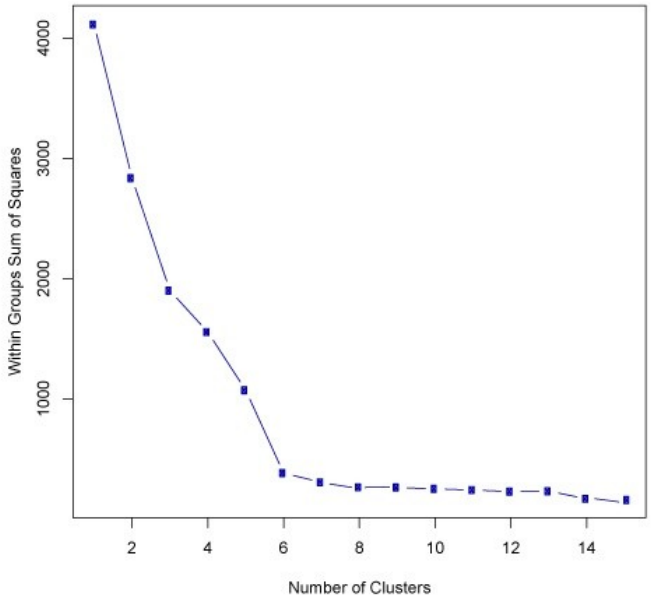
La solución obtenida por K-means depende en gran medida de los valores iniciales de asignación a clusters.

Por este motivo se suele correr el algoritmo muchas veces y quedarse con la mejor solución.

K-means clustering

El valor óptimo a elegir de K es algo que no está claro cómo debe ser elegido.

Algo que se suele hacer es correr el algoritmo con valores crecientes de K y evaluar cómo varía la función a minimizar, eligiendo como valor de K aquel en donde aparezca un “codo”.



K-means clustering

¿Se modifican los resultados si se escalan las variables?

¿Qué se capta en cada caso?

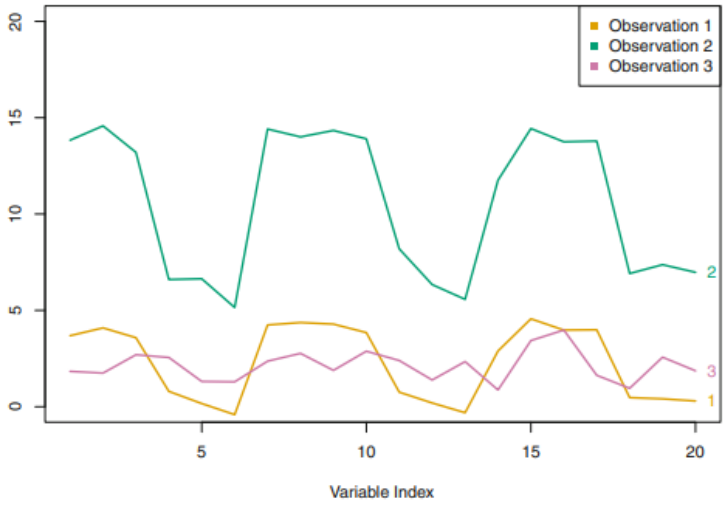


FIGURE 10.13. *Three observations with measurements on 20 variables are shown. Observations 1 and 3 have similar values for each variable and so there is a small Euclidean distance between them. But they are very weakly correlated, so they have a large correlation-based distance. On the other hand, observations 1 and 2 have quite different values for each variable, and so there is a large Euclidean distance between them. But they are highly correlated, so there is a small correlation-based distance between them.*

K-means en R

Veamos en el cuarto bloque de código un ejemplo de k-means.

Lecturas recomendadas para los temas vistos hoy

Boosting:

- ISLR (Sección 8.2, para el final pueden ignorar lo referido a Random Forest)

Xgboost (No obligatoria, pero muy recomendada. No se va a evaluar la teoría):

- https://xgboost.readthedocs.io/en/latest/tutorials/model.html

K-means:

- ISLR (Cap 10, salvo lo referido a componentes principales)

Práctica de laboratorio

Para hacer:

Cargue los datos "*train_3.csv*" del trabajo práctico. Construya la variable *Label* (como vimos en clases) y, además de *Label*, conserve únicamente las variables cuyo nombre termina en "*_sum_dsi3*" y no tienen ningún NA.

Se pide:

- Pase todas las variables menos *Label* a z-scores (i.e., normalizar restando la media y dividiendo por la desviación estándar).
- Utilizando k-means e ignorando la variable *Label* haga un análisis de clustering de las observaciones con k = 5.
- Para los valores de k que van de 1 a 20 pruebe si aparece un codo en la reducción del within sum of squares.
- Tomando como base el análisis anterior, elija un valor de k que le parezca razonable y vea si los distintos clusters encontrados tienen asociados distintos churn rates.

Práctica teórica

¿Es cierto que si los datos de entrenamiento no tienen variables categóricas, no tiene sentido usar matrices ralas? Justifique su respuesta

¿Por qué el algoritmo de boosting puede dar lugar a overfitting si se aumenta la cantidad de árboles indefinidamente? Justifique su respuesta

¿Es cierto que cuando uno lleva adelante un análisis de k medias es indispensable tener un conjunto de validación? Justifique su respuesta

¿En un dataset típico, es cierto que aumentar el valor de k en k medias implica una reduccion del within error? Justifique su respuesta