Universidad de Buenos Aires
Maestría en Data Mining
DM EyF ciclo 2019

Tareas para el Hogar

Nuestro problema consiste en hacer una camapaña proactiva de retención de clientes, tenemos datos hasta junio-2019 y queremos predecir quienes se van a dar de baja durante 201908

En nuestro problema en particular el negocio nos definió una métrica llamada ganancia, que se mide en pesos argentinos, y se nos solicita maximizarla. Todo lo que vemos en la materia sería aplicable si la métrica que nos piden maximizar es el accuracy o el AUC de la curva ROC.
Nuestro problema es de clasificación, si la variable que nos propusieran predecir fuera continua, entonces nuestro problema se llamaría de regresión y no sería tan directa la aplicación de lo que vemos en clase.

Hasta el momento hemos visto la libreria `rpart` de árboles de decisión. Sabemos que los árboles son débiles, pero los estudiamos ya son muy fáciles de entender y son la base de los algoritmos XGBoost y LightGBM .
Nos hemos percatado que para este dataset es mejor pasar las variables de fecha de formato absoluto a formato relativo, por lo que empezamos a trabajar con los archivos de texto con del tipo `201902_dias.txt`
Para estimar correctamente la ganancia hacemos 5 – Montecarlo Estimation, dividir 5 veces el dataset en training/testing, generar 5 modelos entrenando en cada uno de los training, calcular 5 ganancias, y entregamos el promedio como la mejor estimación de la ganancia que va a tener ese arbol en nuevos datos. Cada uno de los alumnos utiliza sus propias 5 semillas.
Surgió el tema que la ganancia del arbol depende en gran medida de los hiperparámetros con los que constuirmos el arbol, situación que se da para todos los algoritmos predictivos, también para XGBoost y LightGBM.
El gran objetivo es de las tareas 2 y 3 del presente documento son : dado este dataset con estos atributos y clase, cuales son los parámetros óptimos de rpart para obtener la mejor ganancia ?
Para ello se pide a los alumnos correr a ciegas dos formas encontrar los parámetros óptimos, Grid Search y Bayesian Optimization. Estas corridas llevarán decenas de horas en generar el resultado final, el que será analizado la próxima clase. También la próxima clase se verá el código en cierto detalle y principalmente se explicará conceptualmente que se está haciendo.

La Optimización Bayesiana será algo fundamental que el alumno se pondrá en su mochila, y le será de gran utilidad en su práctica profesional.
Es totalmente cierto que Optmización Bayesiana es un tema que deberia ser enseñado en forma teórica en otra materia, posiblemente a comienzos del segundo año, y los alumnos recién ver una aplicación práctica en Laboratorio de Implementación II , pero la verdad es que en el mundo profesional hoy en dia no se puede vivir sin Optimización Bayesiana, con lo cual se dan las bases en este laboratorio.

Nuestra tarea como mineros siempre será obtener la mayor ganancia posible, o en le caso general, optimizar la métrica que nos soliciten, siempre optimizar.

En este momento estamos entrenando en el mes 201902 y ya estamos midiendo la ganancia en 201904, o sea en un mes del futuro. Estamos muy bien encaminados, ya que suele ser bastante común en ciencia de datos tener que predecir el futuro y no solo el mismo mes.

Obviamente, habrán más actividades que estaremos haciendo en la materia para mejorar más aun la ganancia :

- utilizar algoritmos más poderosos que los árboles de decisión.
- Feature Engineering

Spoiler Alert : algo muy interesante que vamos a ver muypronto, es que si entrenamos en la unión de varios meses (del pasado), para este problema, para este dataset, obtenemos un modelo mucho mas robusto, que nos permite obtener una ganancia mayor.

Para el futuro, vale la pena leer esto, en donde hay un comentario del genial creador de XGBoost: https://www.quora.com/What-makes-xgboost-run-much-faster-than-many-other-implementations-of-gradient-boosting

También esto otro https://towardsdatascience.com/https-medium-com-vishalmorde-xgboost-algorithm-long-she-may-rein-edd9f99be63d

1. Tareas de housekeeping

- Volver a bajar *toda* la carpeta `datasets` del dropbox, notar que ahora hay una carpeta `dias` y más archivos.
- Cada nuevo dia que decida trabajar con esta tarea para el hogar, antes de comenzar, volver a bajar la carpeta R\ del dropbox a la PC, hay varios scripts nuevos que se van modificando a partir de emails que van llegando con inquietudes, necesidad de aclaraciones, etc

2. Los canaritos cobran vida

Correr `R\elementary\canaritos_dibujo_01.r` en donde se agregan al dataset 17 canaritos (el 10%) y luego mirar el dibujo que queda del arbol en la carpeta work `canaritos_01.jpg`
Cuántos canaritos aparecen ?
Cual es el menor nivel en el que aparece un canarito ?
Hay algun corte por canarito que sus hijos se corte por otra variable ?

Correr `R\elementary\canaritos_dibujo_02.r` en donde se agregan al dataset 170 canaritos , es decir un canarito por cada variable de dataset, y luego mirar el dibujo que queda del arbol en la carpeta work `canaritos_02.jpg`
Cuántos canaritos aparecen ?
Cual es el menor nivel en el que aparece un canarito ?
Hay algun corte por canarito que sus hijos se corte por otra variable ?

3. Corrida de optimización de hiperparámetros, Grid Search

Hemos visto en clase que la ganancia de un arbol de decisión depende fuertemente de los parámetros con el que se lo invoca, profundidad, complexity, etc
El objetivo de esta tarea es poder tener disponible al inicio de la proxima clase la salida de la búsqueda de los parámetros optimos de un arbol de decisión en nuestro dataset de 201902, de forma de poder analizar extensamente en clase el programa y acto seguido los resultados. Esta corrida demora 36 horas .

Correr en la PC local `R\rpart\rpart_tune_gridsearch_01.r`

Si llega a cortarse la corrida, volver a ingresar al Rstudio y ponerlo a correr. El programa continuará corriendo desde donde quedó, ha sido programado para esa contingencia.

En la PC local modificar el script `R\rpart\rpart_tune_gridsearch_01.r` reemplazando las semillas por las propias del alumno .
Esto debe hacerse en la linea que posee el siguiente codigo:

```
montecarlo$semilla_azar         <-  c(102191, 200177, 410551, 552581, 892237)
```

Verificar que en la **PC local** existan estos archivos

- R\rpart\rpart_tune_gridsearch_01.r (recién modificado)
- datasets\dias\201902_dias.txt
- datasets\dias\201904_dias.txt

- Verificar en la PC local que exista la carpeta work
- Verificar en el R de la PC local que esten instaladas las librerias
    - rpart
    - data.table

En caso de utilizar una notebook para correr el script, verificar que la notebook no entrará en suspensión ni hibernación cuando se la deje corriendo durante toda la noche.

Finalmente desde Rstudio poner a correr \R\rpart\rpart_tune_gridsearch_01.r

Se puede ir monitoreando el avance abriendo periodicamente el archivo \work\hiperparametro_GLOBAL.txt y viendo la cantidad de lineas que tiene el archivo.
Analizando los loops for del código, calcule cuantas lineas van a generarse en la salida, de forma de saber si efectivamente terminó la corrida.

Este programa realiza una Grid Search de los parámetros del arbol de decision.

4. Corrida de optimización de hiperparámetros, Bayesian Optimization

Esta corrida demora 10 horas y debe hacerse LUEGO de la corrida de Grid Search
El objetivo es siempre el mismo, encontrar los hyperparámetros óptimos de rpart para este dataset. En este caso con el método de Optimización Bayesiana, que es un método superior al Grid Search. Pedagógicamente ambos deben ser mostrados.
En la próxima clase se verá el código en gran detalle, se explicarán los conceptos básicos de Bayesian Optimization así como se analizará la salida generada.

Correr en la PC local `R\rpart\rpart_tune_MBO_01.r`

Si llega a cortarse la corrida, volver a ingresar al Rstudio y ponerlo a correr. El programa continuará corriendo desde donde quedó, ha sido programado para esa contingencia.

En la PC local modificar el script `R\rpart\rpart_tune_MBO_01.r` reemplazando las semillas por las propias del alumno .
Esto debe hacerse en la linea que posee el siguiente codigo:

```
montecarlo$semilla_azar         <-  c(102191, 200177, 410551, 552581, 892237)
```

Verificar que en la **PC local** existan estos archivos

- R\rpart\`rpart_tune_MBO_01.r` (recién modificado)
- datasets\dias\201902_dias.txt
- datasets\dias\201904_dias.txt
- Verificar en la PC local que exista la carpeta `work`
- Verificar en el R de la PC local que esten instaladas las librerias
  - rpart
  - data.table
  - DiceKriging
  - mlrMBO

Finalmente desde Rstudio poner a correr `\R\rpart\rpart_tune_MBO_01.r`

Se puede ir monitoreando el avance abriendo periodicamente el archivo `\work\hiperparametro_1200.txt` y viendo la cantidad de lineas que tiene el archivo.

5. Generación de código simple

Se utilizará el script `\R\agile\mejorar_ganancia.r`
Con 16GB de memoria RAM funciona bien.

Esta tarea está pensada para alumnos que no tienen formación previa en programación.
Se utilizará un script básico en donde se van desarrollando ciertas actividades, algunas están ya resueltas y otras se pide al alumno que escriba el código en R de las mismas.

La simpleza de programación y en particular del código R esta dada por :

- Es un solo script en R, no se incluyen otros archivos
- No se utiliza la estructura de control `if`
- No se utilizan estructuras de control del tipo `for, while`
- No se utilizan `apply, lapply, mapply`
- No se definen funciones, no se utilizan listas
- No se utilizan constantes, nada está parametrizado
- No se generaliza
- No se hacen optimizaciones, no se busca la performance sino la simpleza
- No se hacen tareas de housekeeping, como borrar objetos que no se van a utizar más y su correspondiente garbage collection

Lo simple (y restrictivo) es la forma de programar y en particular de escribir el código en R, s*in embargo se estará trabajando con conceptos complejos, algunos que aun ni siquiera fueron vistos en clase.*

Los alumnos SI pueden usar todas las instrucciones del lenguaje R que deseen, si pueden crear funciones, usar constantes, etc

Solamente se utiliza el dataset `/datasets/paquete_premium_dias.txt`
Las salidas se escriben en la carpeta `/work`