Universidad de Buenos Aires
Maestría en Data Mining
DM EyF
ciclo 2019 "the Red Pill year"

Tarea para el Hogar CPE 1704 TKS

1. Tareas de housekeeping
   - Se han hecho cambios a la carpeta R del dropbox, bajarla nuevamente a la PC local (casi a diario se hacen cambios en la carpeta R)

2. Correr `R/rpart/rpart_tune_MBO_meses.r`

   Hasta ahora la mejor ganancia que tenemos con `rpart` sobre `201904_dias.txt` es de alrededor $ 6.100.000 . Los parámetros optimos se encontraron entrenando en los datos de `201902_dias.txt` haciendo 5-fold Montecarlo.

   Pues bien, antes de abandonar el algoritmo `rpart` para pasar a algoritmos de ensembles, vamos a intentar sacarle todo el jugo posible. Para ello entrenando en varios meses del pasado vamos lograr obtener una ganancia en 201904_dias.txt de aproximadamente $ 7.400.000

   Dados unos < ventana_meses, maxdepth, minsplit, minbucket, cp > primero vamos en generar un modelo en n meses, tomados del 201812 hacia el pasado, y medir la ganancia en 201902 , esa la vamos a llamar metrica1_actual
   Luego usando esos mismos parámetros entrenamos en 201902 y n meses tomados hacia el pasado, y aplicamos el modelo a 201904, a esa ganancia la llamamos metrica1_futuro

   Por ejemplo, si n=8 entrenamos en la unión de los meses de {201805, 201806, 201807, 201808, 201809, 201810, 201811, 201812} y a ese modelo lo aplicamos a 201902
   Como segundo paso, con esos mismos parámetros entrenamos en {201807, 201808, 201809, 201810, 201811, 201812, 201901, 201902} y a ese modelo lo aplicamos a 201904

Es la Optimización Bayesiana que resuelve toda la optimización al mismo tiempo

```
obj.fun <- makeSingleObjectiveFunction(
        name = "prueba",
        fn   = funcion_optimizar,
        par.set = makeParamSet(
            makeIntegerParam("pmaxdepth"         ,  lower=6L    , upper=  30L),
            makeNumericParam("pminbucket"        ,  lower=0.05  , upper=   0.5),
            makeIntegerParam("pminsplit"         ,  lower=1L    , upper= 200L),
            makeNumericParam("pcp"               ,  lower=0.0   , upper=   0.001),
            makeIntegerParam("pventana"          ,  lower=1L    , upper=  12L)
        ),
        has.simple.signature = FALSE,
        global.opt.value = -1
        )
```

El alumno que considera perfectibles los lower/upper elegidos para la optimización bayesiana es invitado a probar sus propios valores. Solamente tener en cuenta que en el código, la linea `dataset_grande <- dataset_grande[ foto_mes>=env$data$mes_primero & foto_mes<=env$data$mes_ultimo , ]` con la finalidad de usar menos memoria esta reduciendo el dataset, eliminar esa linea si se quiere probar una ventana de mas de 12 meses.

Debe quedar claro que se está utilizando la eficiente Optimización Bayesiana para determinar los al mismo tiempo la combinación óptima de parámetros :

- ventana (cantidad de meses donde se entrena)
- arbol rpart
  - maxdepth
  - minsplit
  - minbucket
  - cp

Para el mlrMBO esto es optimizar cinco hiperparámetros al mismo tiempo; mlrMBO no tiene idea que un parámetro regula la cantidad de meses del dataset de training y los otros cuatro el algoritmo rpart.

Plataforma de corrida : Google Cloud

Prerrequisitos

- En el bucket de Google Cloud debe existir la carpeta `cloud1` que es donde estan todos los archivos
- Dentro de la carpeta `cloud1` debe estar las carpetas
  - R

    - datasetsOri
    - datasets
    - work

- En el bucket de Google Cloud debe existir los archivos
    - /cloud1/datasets/paquete_premium_dias.txt

- Bajar del dropbox de la materia el archivo R/rpart/rpart_tune_MBO_meses.r a la PC local
- Subir el archivo recien bajado al bucket de Google Cloud a la carpeta cloud1/R/rpart

Plataforma de ejecución: Cloud, Virtual Machine mortal preemptive , 8 vcpu, 32 GB RAM
Para crear la máquina virtual seguir los pasos del documento ProcesamientoCloud.pdf página 30, dicho documento se encuentra en el dropbox en la carpeta cloud y es el que se utilizó para configurar Google Cloud .

Este proceso demora *bastante más* de 24 horas, cuando la máquina virtual preemptive se apague, crear una nueva en algun lugar de mundo que sea de noche y queden aun varias horas sin luz solar, y volver a lanzar el proceso, que retoma desde donde quedó.

La salida de esta corrida quedará en el archivo work/hiperparámetro_1301.txt

3. Correr `R/rpart/rpart_tune_MBO_meses_sample_02.r`

El proceso anterior logró su cometido y luego de algo más de 48 horas y USD 4.50 (cuatro dólares y medio) en máquinas virtuales logramos incrementar nuestra granancia en $ arg 1.300.000 (un millón trescientos mil pesos argentinos).

Desde el punto de vista económico deberíamos estar satisfechos, pero los *daemons de la optimización* siempre nos piden más, y nos preguntamos : ¿Es posible encontrar en significativamente menos tiempo de procesamiento que la cantidad optima de meses donde entrenar es 8 y que se puede llegar a una ganancia de $ 7.400.000 ?

Para hacerlo vamos ha hacer un sampling inteligente del dataset union de n meses donde entrenamos :

- todos los BAJA+2
- solo el 10% de los {BAJA+1, CONTINUA}

En los meses que testeamos la ganancia no hacemos ningun tipo de sampling.

El 10% de sampling no es una constante universal, sinó que se probó primero con un 1% , no resultó y luego se pasó al 10%

Gran atención a este detalle, el undersampling de los negativos que estamos haciendo hace que 0.025 ya no es el umbral de probabilidad de BAJA+2 , dado que ahora los BAJA+2 estan sobre representados en el dataset, su proporción no es la original.
Este umbral ahora se calcula con

```
problema$prob_corte           <-
-problema$ganancia_noacierto*(1/env$undersampling)/
( problema$ganancia_acierto -
problema$ganancia_noacierto*(1/env$undersampling) )
```

Esta fórmula, quizás sea explicada en la próxima clase, o quizas se pida a los alumnos que la deriven por ellos mismos.

Plataforma de corrida : Google Cloud

Prerrequisitos
En el bucket de Google Cloud debe existir la carpeta `cloud1` que es donde estan todos los archivos
Dentro de la carpeta `cloud1` debe estar las carpetas

- `R`
- `datasetsOri`
- `datasets`
- `work`

En el bucket de Google Cloud debe existir los archivos

- `/cloud1/datasets/paquete_premium_dias.txt`

Bajar del dropbox de la materia el archivo `R/rpart/rpart_tune_MBO_meses_sample_02.r` a la PC local
Subir el archivo recien bajado al bucket de Google Cloud a la carpeta `cloud1/R/rpart`

Plataforma de ejecución: Cloud, Virtual Machine mortal preemptive, 8 vcpu, 32 GB RAM

La salida de esta corrida quedará en el archivo `work/hiperparámetro_1303.txt`

4. Correr `R/rpart/rpart_canaritos_meses.r`

¿Podrán los frágiles canaritos con la unión de múltiples datasets ?
¿Cómo se comportará el pruning de canaritos en el caso de undersampling ?

En la tarea 3. , haciendo undersampling pasamos de 50 horas a menos de 5 horas de procesamiento, con una ganancia en 201904 similar.

Que tal si ahora combinamos esto

- Undersampling
  - {BAJA+2} todos
  - {BAJA+1, COUNTINUA } sampling del 10%
- algoritmo arbol : Canary Attributes ™

dado que solamente vamos a variar el tamaño de la ventana, no recurrimos a la optimización bayesiana sino que simplemente usamos un loop

```
#Aqui busco por todas las ventanas posibles
for( ventana in 1:14 )
{
  gan_futuro <-  modelo_rpart_canarito_directo( ventana )
}
```

Prerrequisitos
En el bucket de Google Cloud debe existir la carpeta `cloud1` que es donde estan todos los archivos
Dentro de la carpeta `cloud1` debe estar las carpetas

- `R`
- `datasetsOri`
- `datasets`
- `work`

En el bucket de Google Cloud debe existir los archivos

- `/cloud1/datasets/paquete_premium_dias.txt`

Bajar del dropbox de la materia el archivo `R/rpart/rpart_canaritos_meses.r` a la PC local
Subir el archivo recien bajado al bucket de Google Cloud a la carpeta `cloud1/R/rpart`

Plataforma de ejecución: Cloud, Virtual Machine mortal preemptive, 4 vcpu, 20 GB RAM

La salida de esta corrida quedará en el archivo `work/hiperparámetro_1366.txt`