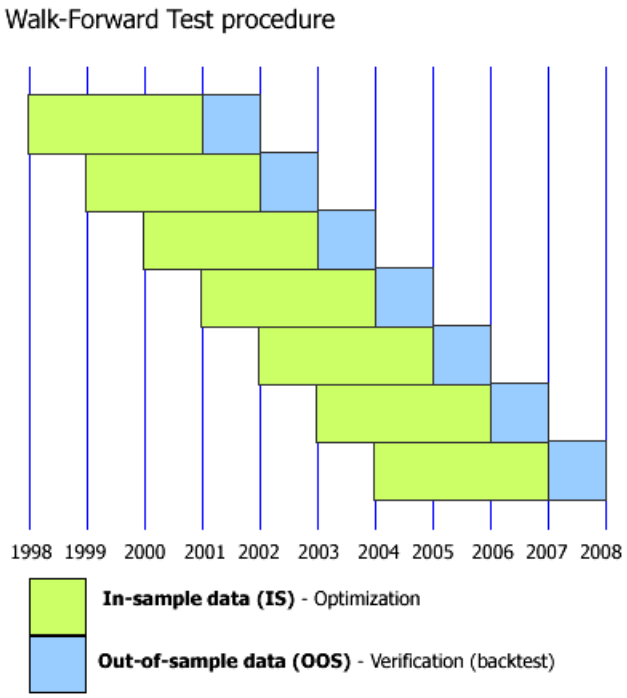
Universidad de Buenos Aires
Maestría en Data Mining
DM EyF
ciclo 2019 "the Red Pill year"

Tarea para el Hogar NUEVE "Linea de Muerte"

En esta tarea para el hogar se presenta la Linea de Muerte de la materia y se intentan unos primeros modelos predictivos para superarla.

La forma en que estamos construyendo los modelos predictivos mes a mes con el método de Walk Forward Validation es la siguiente (en nuestro caso la ventana de meses de color verde es de tamaño variable).



Dispondremos para la materia de estos cuatro tipo de datasets, ordenados por complejidad creciente:

1. `paquete_premium_dias.txt` (sin feature engineering)
2. `paquete_premium_ext.txt` (variables nuevas dentro del mismo mes)
3. `paquete_premium_hist.txt` (solo historia max, min, tendencia)
4. `paquete_premium_exthist.txt` (aparece en esta tarea para el hogar)

FE hace referencia a Feature Engineering

	Sin FE en el mes	Con FE en el mes
Sin FE histórico	`_dias`	`_ext`
Con FE histórico	`_hist`	`_exthist`

1. Tareas de housekeeping

- Se han hecho cambios a la carpeta R del dropbox, se han agregado scripts y se han modificado scripts existentes, bajarla nuevamente a la PC local y luego subir al bucket de Google Cloud

2. Linea de Muerte

Este punto no es una tarea, sino una información.
El script `R/lineademuerte/lineademuerte_UBA.r` es el script que genera los `numero_de_cliente` de la linea de muerte de la materia

Básicamente se genera un dataset con una historia de diez meses, y se llama a XGBoost con estos parámetros :

```
xgb.train(
          data= dgeneracion,
          objective= "binary:logistic",
          tree_method= "hist",
          max_bin= 31,
          base_score= mean( getinfo(dgeneracion, "label") ),
          eta= 0.04,
          nrounds= 300,
          colsample_bytree= 0.6
         )
```

estos parámetros suelen ser estandar en una corrida de XGBoost
`tree_method= "hist"` los atributos del dataset se discretizan
`max_bin= 31` la discretizacion es en 31 bins

`base_score= mean( getinfo(dgeneracion, "label") )` la probabilidad inicial de ser positivo es pos/total_registros del dataset de entrenamiento

Los siguientes parámetros fueron optimizados con Bayesian Optimization para que el modelo fuera bueno en el promedio de los ultimos once meses (201904 a 201806):

`eta= 0.04` el learning rate
`nrounds= 300` la cantidad de arboles
`colsample_bytree= 0.6` antes de la generación de cada arbol se selecciona al azar el 60% de los atributos, y se construye el arbol usando unicamente esos atributos

Hay un parámetro que no figura explicitamente en la llamada a XGBoost que sin embargo es fundamental y es que `data= dgeneracion` fue creado usando una ventana de 10 meses.

```
dgeneracion  <-   xgb.DMatrix(
data  = data.matrix( dataset[ foto_mes>=201807 & foto_mes<=201904 , !
c("numero_de_cliente","clase_ternaria"), with=FALSE]),
label = dataset[ foto_mes>=201807 & foto_mes<=201904,clase_ternaria ])
```

Es necesario hacer especial énfasis en que no se estan utilizando los parámetros más avanzados del XGBoost como lo son : `max_depth, min_child_weight, gamma, alpha, lambda`
Estrictamente hablando, XGBoost están usando los valores por default para los parámetros anteriores.

Simple is beatiful, and hard to beat .

No podemos saber como le irá a la Linea de Muerte en los datos del futuro 201906, sin embargo si estamos en condiciones de calcular la ganancia para el pasado :

Entrenamiento (ventana 10 meses)		**Aplicacion mes**	**Registros**	**Positivos**	**Ganancia Linea de Muerte**
mes_desde	mes_hasta				
201706	201804	201806	180,768	1,294	12,505,500
201708	201805	201807	181,615	1,201	10,298,000
201709	201806	201808	182,603	1,222	11,132,500
201710	201807	201809	183,646	1,197	11,939,500
201711	201808	201810	184,511	1,010	9,620,500
201712	201809	201811	184,888	1,070	10,382,500
201801	201810	201812	185,392	1,103	11,073,000
201802	201811	201901	186,398	985	9,479,000
201803	201812	201902	187,861	1,085	10,528,000
201804	201901	201903	188,834	1,119	10,423,000
201805	201902	201904	189,594	918	9,122,500
201807	201904	201906			aún desconocido

Es justo reconocer cuatro grandes fortalezas de como fue construida esta Linea de Muerte, que hace que sea un gran desafío superarla :

- Esta entrenando con el archivo _hist
- Esta entrenando con todo el dataset, no hace undersampling de los negativos
- Cuando entrena, lo hace con una ventana de 10 meses
- se utiliza XGBoost

Estas son las ganancias que deben ser superadas *comodamente* por el mejor modelo que encuentren para tener mas oportunidades de superar la Linea de Muerte en 201906 y aprobar la materia. Intencionalmente no especifico el significado de la palabra *comodamente*, científicos de datos deben aprender a vivir en las profundidades de la mina son insondables.

La Linea de Muerte es el archivo `R/lineademuerte/lineademuerte_entregar_UBA.txt` , contiene 7312 registros. Para aprobar la materia es necesario tener una ganancia mayor a igual a la que obtendrán esos 7312 registros en el mes 201906 .

La decisición de solamente calcular en el pasado hasta 201806 fue arbitraria; siéntase libre de extender la tabla anterior hasta reducir su angustia.

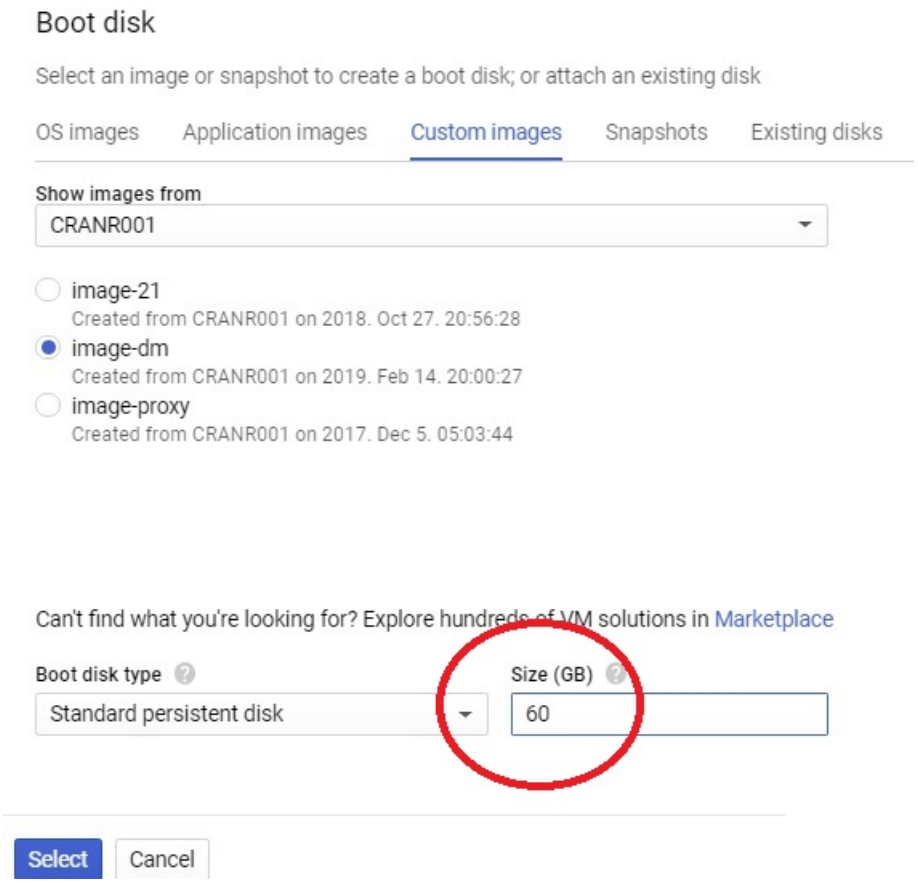
Por favor no confudir :

- El modelo siempre se entrena en una ventana de 10 meses
- Se calculó cuanta ganancia da la Linea de Muerte para once meses del pasado, pero siempre se está entrenando en 10 meses

3. Correr R/FeatureEngineering/fe_todoenuno.r

Antes de correrlo es muy conveniente que se haga feature engineering y se agreguen nuevos campos al dataset, a partir de la linea 224 del script .

Plataforma de ejecución: Cloud, Virtual Machine inmortal pagando el precio fulll, ya que este proceso no está programado para retomar el procesamiento en caso de apagarse la máquina
4 vcpu, 64 GB RAM , al elegir la imagen del sistema operativo asignar 60GB al espacio en disco.



Este proceso demora 50 minutos, por favor recordar apagar la maquina virtual ya que esta es inmortal y **no** se apaga automaticamente a las 24 horas.

La corrida genera :

- /datasets/paquete_premium_exthist.txt.gz
- /datasets/exthist 36 archivos

4. Correr R/LightGBM/lightgbm_directo_wfv_dias.r

Este punto ya está corrido para los alumnos, es uno de los objetivos que comprendan los meses que se entrena y testea y evaluen los resultados de la corrida.
Plataforma de ejecución: Cloud, Virtual Machine mortal preemptive , 8 vcpu, 32 GB RAM

Este script trabaja con el muy pobre dataset paquete_premium_dias.txt , hace once optimizaciones bayesianas, cada una de las cuales predice los meses de { 201904, 201903, 201902, 201901, 201812, 201811, 201810, 201809, 201808,201807, 201806 } Notar que 201806 es la competencia del año pasado.

La metodología que este script utiliza para entrenar es la siguiente :
Supongamos que el mes sin clase fuera 201904 y se desea entregar la predicción de la materia, estamos simulando como si 201904 fuera nuestro 201906 para el que no tenemos la clase, pero en este juego si vamos a poder saber como nos hubiera ido en la competencia.

Si 201904 es nuestro mes sin clase, entonces tenemos que 201902 sería el último mes con clase.
Para la optimización bayesiana, si 201902 es usado como testing, entonces NO puede ser utilizado para training, con lo que el esquema utilizando una ventana de ejemplo de 4 meses quedaria :

- 201904 mes donde se aplica el modelo final, que supuestamente no tiene clase,este mes NO se puede usar para nada más.
- (201903 supuestamente tiene incompleta la clase, por lo que no se usa para nada.)
- { 201902, 201901, 201812, 201811 } meses donde se entrena el modelo final con parámetros optimos aprendidos en una optimización bayesiana en meses anteriores.

- 201902 se usa como testing de la optimización bayesiana
- { 201812, 201811, 201810, 201809 } son los meses que se usan como training en la optimización bayesiana. De la optimización bayesiana salen cuales son los mejores parámetros.

Todo el esquema anterior, se va “corriendo en forma móvil hacia el pasado” a medida que en nuestro juego vamos probando el modelo en meses pasados.

La salidas importantes, que generan en la carpeta work, son :

- hiperparametro_15001.txt
- salida_15001_entregamateria.txt

En `hiperparámetro_15001.txt` quedan solamente los resultados finales de las once optimizaciones bayesianas, los valores de `metrica1_futuro` son los que se deben comparar con los de la tabla de Linea de Muerte de la tarea 2, el campo dataset_futuro_test indica el mes que se está prediciendo.

Linea de Muerte		Experimento 15001	
Aplicacion Mes	Ganancia	dataset_futuro_test	metrica1_futuro
201806	12,505,500	201806	11,988,000
201807	10,298,000	201807	9,937,500
201808	11,132,500	201808	10,083,000
201809	11,939,500	201809	10,803,500
201810	9,620,500	201810	8,591,000
201811	10,382,500	201811	9,711,500
201812	11,073,000	201812	10,436,000
201901	9,479,000	201901	8,998,000
201902	10,528,000	201902	8,781,000
201903	10,423,000	201903	9,441,000
201904	9,122,500	201904	7,997,000

Se ve que el Experimento 15001 NO supera para ninguno de los once meses la linea de muerte , sin la menor duda usar el archivo _hist le aporta al modelo mucha mas información que el archivo _dias .

Notar que NO se estan grabando en `hiperparametro_15001.txt` los valores intermedios de la optimización bayesiana, solamente se graba el mejor de cada optimizacion.

5. Correr R/LightGBM/lightgbm_directo_wfv_hist.r

Plataforma de ejecución: Cloud, Virtual Machine mortal preemptive , 8 vcpu, 64 GB RAM

Es la misma corrida que la anterior, pero ahora utilizando el archivo con Feature Engineering que se agregaron campos históricos, pero no tiene campos nuevos del presente. Este archvio es el mismo que usa la linea de muerte.
Recordar que se está haciendo un subsampling de los negativos del 10% y utilizando LightGBM

Es el mismo que el script de la tarea anterior con estos dos cambios

- cambiado `env$experimento` (linea 38) a `15002`
- cambiado `data$archivo_grande` (linea 52) a `"paquete_premium_hist.txt.gz"`

con los cambios anteriores estamos haciendo que se use el dataset `paquete_premium_hist.txt` el que debería generar una ganancia sensiblemente superior.

Linea de Muerte		Experimento 15002	
Aplicacion Mes	Ganancia	dataset_futuro_test	metrica1_futuro
201806	12,505,500	201806	11,764,500
201807	10,298,000	201807	9,887,500
201808	11,132,500	201808	10,828,500
201809	11,939,500	201809	11,295,000
201810	9,620,500	201810	9,208,000
201811	10,382,500	201811	10,251,500
201812	11,073,000	201812	10,534,000
201901	9,479,000	201901	9,453,000
201902	10,528,000	201902	9,857,000
201903	10,423,000	201903	9,693,500
201904	9,122,500	201904	8,743,000

El experimento 15002 esta por debajo de la Línea de Muerte para diez de los once meses.
Esto es desesperante, ya que estamos utilizando el mismo archivo de entrada que Linea de Muerte, el `paquete_premium_hist` .
Pero, en realidad la Linea de Muerte está generada con el dataset completo, y esta corrida 15002 fue hecha con undersampling del 10% de los negativos.
Otra diferencia es que esta corrida utiliza LightGBM mientras que la Linea de Muerte usa XGBoost

6. Correr R/LightGBM/lightgbm_directo_wfv_hist.r modificado

Plataforma de ejecución: Cloud, Virtual Machine mortal preemptive , 8 vcpu, 128 GB RAM
No temer a utilizar una maquina con 128 GB de RAM y en caso que luego de correr algunas horas falle por falta de memoria, recrear la máquina virtual con incrementos de 32 GB hasta que funcione.

Es la misma corrida que la anterior, usando los datos históricos, pero ahora SIN undersampling, es decir usando el 100% de los datos. Esto hará la corrida muchísimo mas lenta, pero todo vale en el desesperado intento de superar la linea de muerte.
En esta corrida se utiliza LightGBM .

Al script al se le deben hacer estos cambios

- cambiar `env$experimento` (linea 38) a `15003`
- cambiar env$undersampling (linea41) a `1.0`

Linea de Muerte		Experimento 15003	
Aplicacion Mes	Ganancia	dataset_futuro_test	metrica1_futuro
201806	12,505,500	201806	12,203,500
201807	10,298,000	201807	10,181,500
201808	11,132,500	201808	10,733,500
201809	11,939,500	201809	11,620,500
201810	9,620,500	201810	9,086,000
201811	10,382,500	201811	10,161,500
201812	11,073,000	201812	11,001,000
201901	9,479,000	201901	9,319,000
201902	10,528,000	201902	10,319,000
201903	10,423,000	201903	10,473,500
201904	9,122,500	201904	9,012,500

El experimento 15003 tampoco supera la linea de muerte.
Estamos utilizando el mismo dataset de entrada que la Linea de Muerte
`paquete_premium_hist.txt`
Estamos entrenando sobre todos los datos al igual que la Linea de Muerte
La diferencia es que usamos LightGBM y la Linea de Muerte usa XGBoost .

Pero tenemos el *as en la manga* del archivo paquete_premium_exthist en donde toda la nuestra creatividad como seres humanos se aplica a la creación de nuevas variables que luego automaticamente se derivan las históricas.