

Data Mining en Economía y Finanzas



UBA

Maestría en Explotación de datos y Descubrimiento del Conocimiento
2019

Indice de la presentación

- 1) Perspectiva de la materia
- 2) Experimento reducción variables
- 3) Como ganarle a la línea de muerte sin ~~terminar~~
~~escupiendo sangre~~ embarrassarse

Parte 1

Perspectiva
de materia

Objetivo de la materia

Se guiará a los alumnos a desarrollar análisis de data mining que sea competitivo con los de nivel profesional existentes en el mercado argentino.

Metodología Pedagógica

Dar teoría y scripts soporte a los alumnos de forma elevarlos lentamente hasta una altitud segura que les permita, una vez solos, maniobrar sin estrellarse. Mencionarles dosificadamente alternativas superadoras.

Librarlos a su suerte, para que vivan una experiencia emotiva con la ciencia de datos, y en un autodiagnóstico posterior determinen el *signo* y la magnitud de dichas emociones.



Photographer: Richard Lockett
Air-and-Space.com







Ganancia 201906	
1°	\$ 9,138,000
2°	\$ 9,110,500
mediana	\$ 8,892,000
Linea Muerte original	\$ 8,744,000

Linea de Muerte

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

Evolución de la ganancia

período evaluación 201904

implica

entrenamiento <= 201902

línea de muerte en 201904 \$ 9,122,500

Los campos de tipo fecha
originalmente estan
en formato de fecha absoluta

Se los pasa a
cantidad de dias
relativa a la fecha de la foto

Primer algoritmo

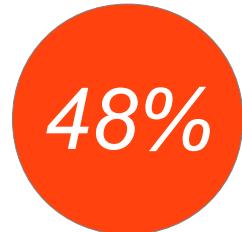
Decision Trees
Probability Trees

Rpart

Lo más básico de Rpart

Modelo	tiempo hs	Ganancia
Rpart parámetros default entrenado sobre periodo 201902	0.02	4,393,500

paquete_premium_dias.txt



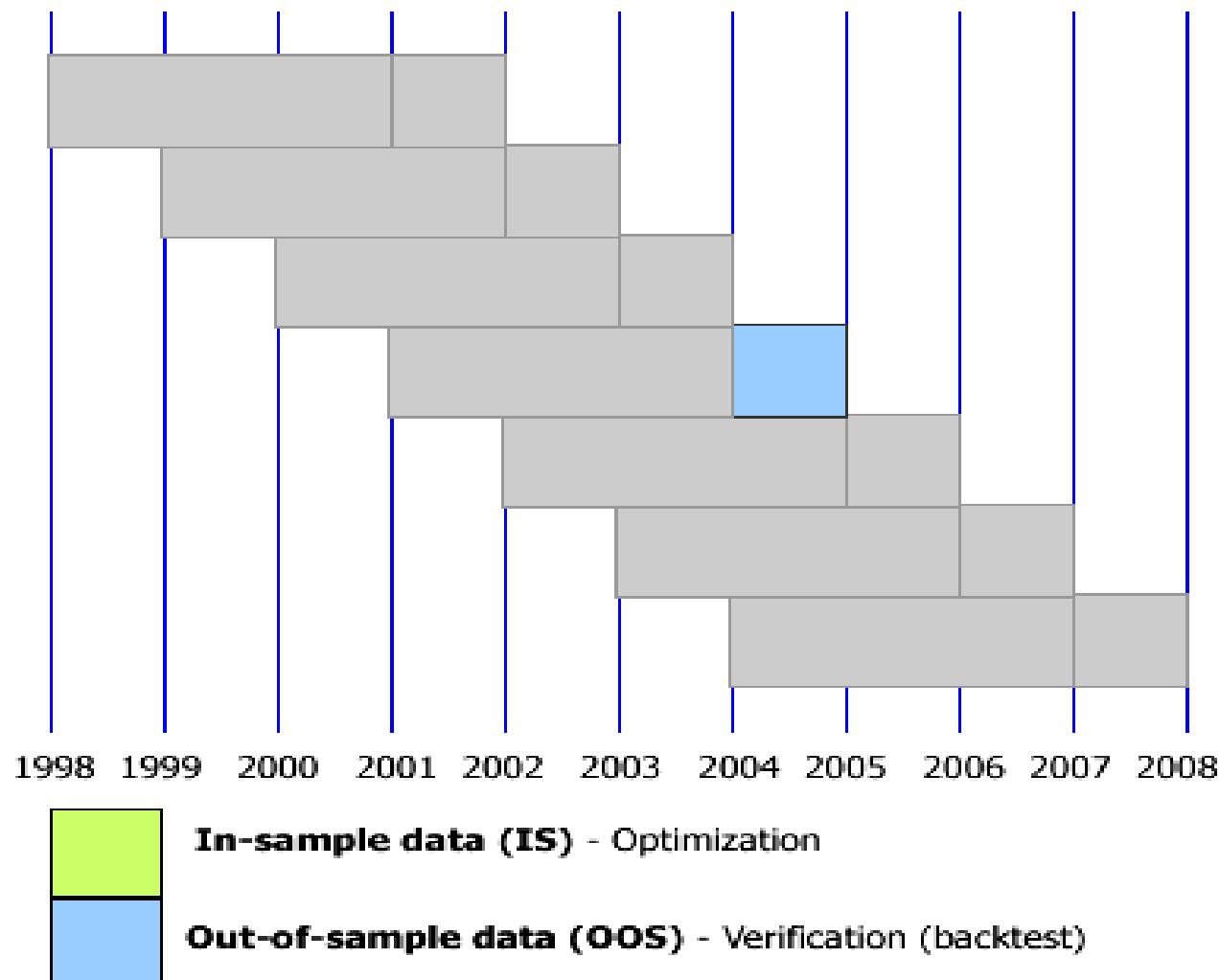
9,122,500

Se ve la necesidad de optimizar los
hiperparámetros

Grid Search
Bayesian Optimization

Walk Forward Validation

STEP 8. Verification (backtest)
over 4th section of OUT-OF-SAMPLE data



Walk Forward Validation

Determino parámetros óptimos mediante
Bayesian Optimization en
Train=[201803, 201812] Validate=[201802]

Con esos parámetros óptimos genero
el modelo final en Train=[201805, 201902]

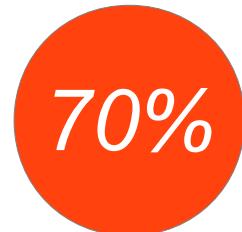
Aplico el modelo final a
[201904]

Hyperparameter Optimization

Rpart entrenado en 201902

Tipo Optimización	tiempo	Ganancia
Grid Search	30.0	5,971,500
Bayesian Optimization	1.5	6,109,500
Pruning Canaritos	0.02	6,392,500

paquete_premium_dias.txt



9,122,500

Aquí se produce el gran salto

entrenar en una ventana
de n-meses

Entrenar en Ventana

Rpart entrena en 201805 – 201902 (10 meses)

Modelo Rpart	tiempo	Ganancia
Bayesian Optimization	17	7,179,500
Pruning Canaritos	0.3	7,195,000
paquete_premium_dias.txt	79%	9,122,500



Segundo algoritmo

Random Forest

ranger

ranger
corre tan lento
(comparado con XGBoost)

que no vale la pena

(además XGBoost puede emularlo)

Tercer algoritmo

Gradient Boosting
of Decision Trees

XGBoost LightGBM
CatBoost

Gradient Boosting inicial

Entrena en 201902 (1 mes)

Modelo	tiempo	Ganancia
XGBoost parametros default	0.006	6,481,500
LightGBM parametros default	0.0005	6,649,500

paquete_premium_dias.txt



9,122,500

Parámetros por default
NUNCA
dan buenas ganancias

hay que hacer
Hyperparameter Tuning

Gradient Boosting Bayesian Optimization

Entrena en 201902 (1 mes)

Modelo con Bayesian Optim	tiempo	Ganancia
XGBoost	1.4	7,897,500
LightGBM	0.3	7,860,500

paquete_premium_dias.txt



9,122,500

Gradient Boosting

Bayesian Optimization

Entrena en 201805 – 201902 (10 meses)

Modelo con Bayesian Optim	tiempo	Ganancia
XGBoost	7.9	8,691,000
LightGBM	0.4	8,491,000

paquete_premium_dias.txt



9,122,500

Llegué al estado del arte
en cuanto a poder del algoritmo

¿Qué más puedo hacer ?



Datasets

	Variables Originales	Variables Nuevas
Solo ese mes	_dias	_ext
Historia 6 meses	_hist	_exhist

Variables historia

historia 6 meses (mes actual y 5 previos)

Variables

- Mínimo
- Máximo
- Tendencia (pendiente regresión)
- Promedio

el dataset tiene una superficie x4

En los siguientes experimentos
Se trabaja con todos los registros
de cada mes

(no se hace undersampling)

Gradient Boosting inicial

Entrena en 201805 – 201902 (10 meses)

Modelo	tiempo	Ganancia
XGBoost parametros default	0.02	5,604,000
LightGBM parametros default	0.006	7,917,500

paquete_premium_dias.txt



9,122,500

dataset `_hist` parámetros default

Entrena en 201805 – 201902 (10 meses)

Modelo	tiempo	Ganancia
XGBoost parametros default	0.02	6,545,000
LightGBM parametros default	0.02	8,009,000

paquete_premium_hist.txt



9,122,500

dataset _hist

Entrena en 201902 (1 mes)

Modelo con Bayesian Optim	tiempo	Ganancia
XGBoost	2.4	8,152,500
LightGBM	1.5	8,343,500

paquete_premium_hist.txt



9,122,500

dataset _hist

Entrena en 201805 – 201902 (10 meses)

Modelo con Bayesian Optim	tiempo	Ganancia
XGBoost	13	9,272,500
LightGBM	5	9,148,500
paquete_premium_hist.txt	101.6%	9,122,500

101.6%

101.6%

en 201904

¿ya aprobaba?

XGBoost sobre el dataset _hist
y una ventana de 10 meses de entrenamiento
logra una ganancia en 201904 de 9,272,500
La que es superior a 9,122,500 de la linea de muerte
de ese mes.

¿Qué hubiera pasado si entregaba ese modelo?

XGBoost

```
data= dgeneracion,  
objective= "binary:logistic",  
tree_method= "hist",  
max_bin= 31,  
base_score= mean( getinfo(dgeneracion, "label") ),  
eta= 0.0869930919170685,  
nrounds= 261,  
colsample_bytree= 0.713037027046375,  
max_depth= 17,  
min_child_weight= 47.2673118770532,  
alpha= 10.6864050620908,  
lambda= 10.6518704269984,  
gamma= 5.00773516416978
```

101.6%

en 201904

aplicado a 201906

BAJA+2= 626

BAJA+1= 282

CONTINUA= 6417



Ganancia en 201906 = 8,857,500

(linea de muerte = 8,744,000)

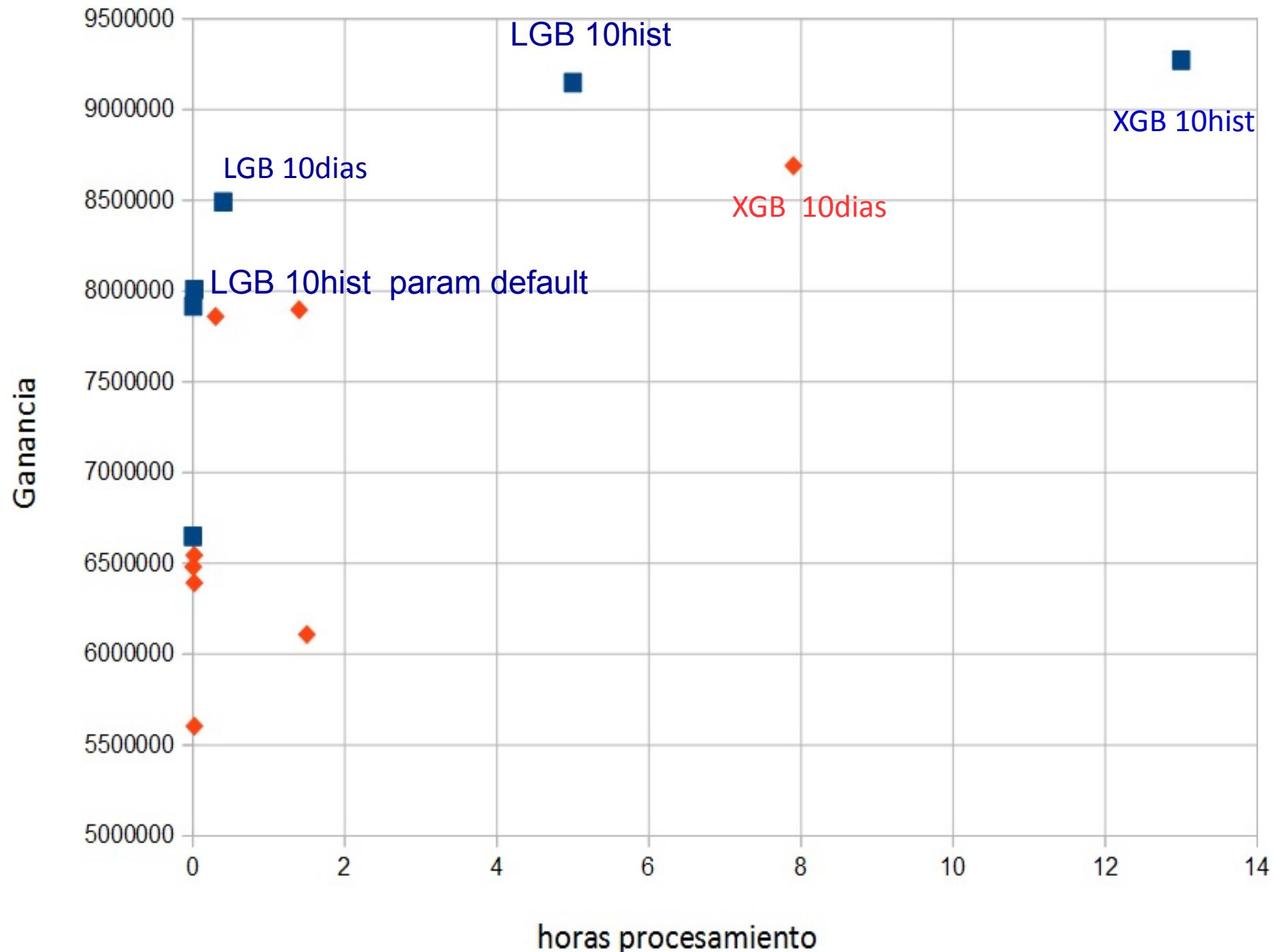
101.3%

en 201906

Pespectiva Final

Tiempo	Ganancia	Algoritmo	Params	Dataset	Meses
0.0005	6,649,500	LightGBM	--	_dias	1
0.006	7,917,500	LightGBM	--	_dias	10
0.02	8,009,000	LightGBM	--	_hist	10
0.4	8,491,000	LightGBM	BO	_dias	10
5.0	9,148,500	LightGBM	BO	_hist	10
13	9,272,500	XGBoost	BO	_hist	10

9,122,500



Conclusión 1

Entrenar / Validar / Testear

Walk Forward Validation

validando en un mes distinto
a donde se entrena

Conclusión 2

Hyperparameter Optimization
con
Bayesian Optimization

Conclusión 3

El mayor salto individual
en la ganancia, lo da

utilizar Gradient Boosting
of Decision Trees

Conclusión 4

Ya utilizando Gradient Boosting
of Decision Trees
el mayor salto individual
en la ganancia, lo da

entrenar en una ventana
de n (~ 10) meses

Conclusión 5

El último gran empuje
lo da utilizar variables históricas

Este tipo de feature engineering
provee 7 puntos, muy por encima
de lo alcanzable con feature engineering
dentro del mismo mes



The important thing is not triumph,
but the struggle .

Pierre de Coubertin
founder of the International Olympic Committee

Struggle.verb /'strʌg.əl/
to experience difficulty
and make a very great effort
in order to do something



Parte 2

Experimento
reducción
de
variables
(feature selection)

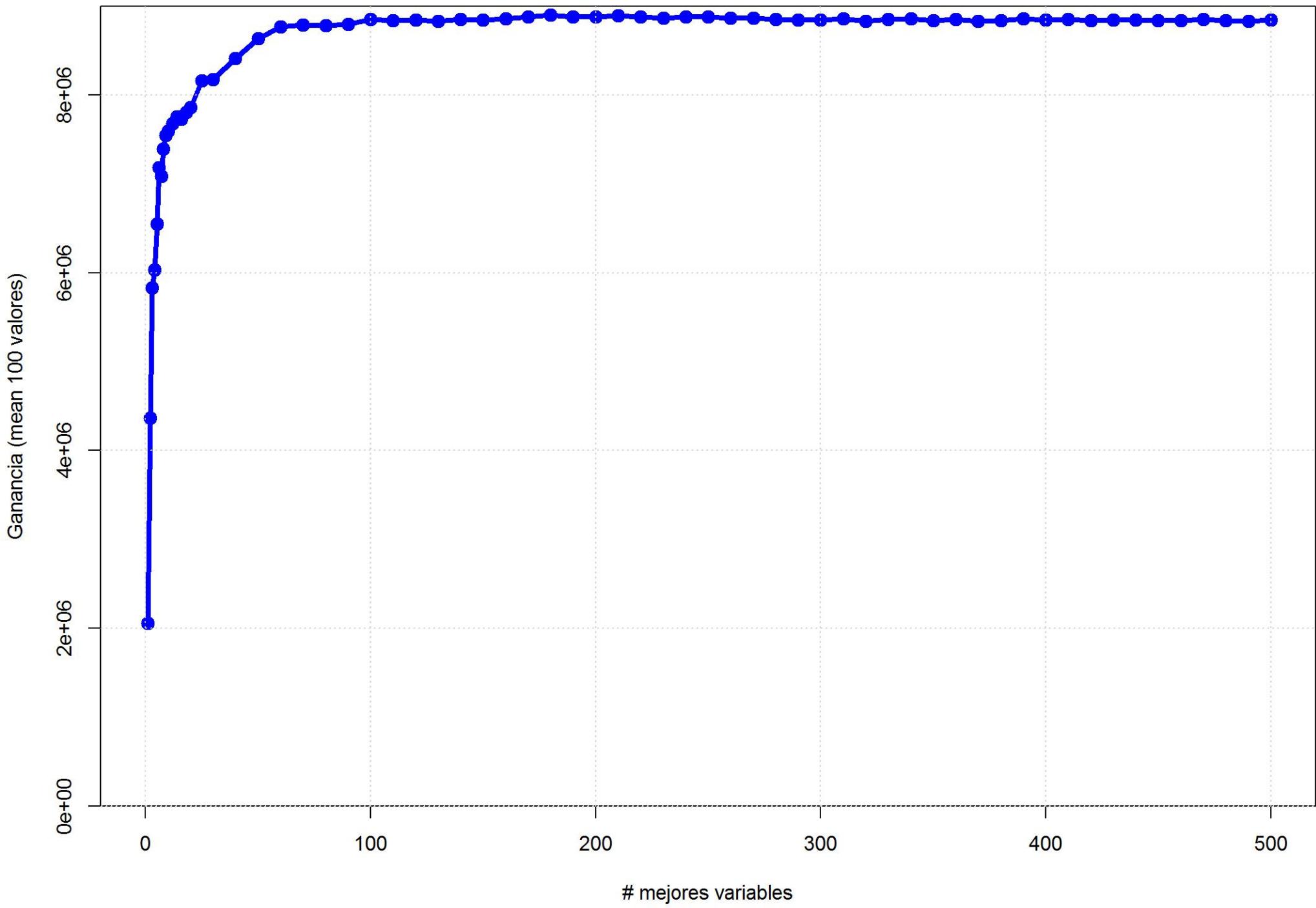
Experimento

Hiperparámetros obtenidos por Optim Bayes en
Train=[201803, 201812] Validate=[201902]
feature_fraction=1.0

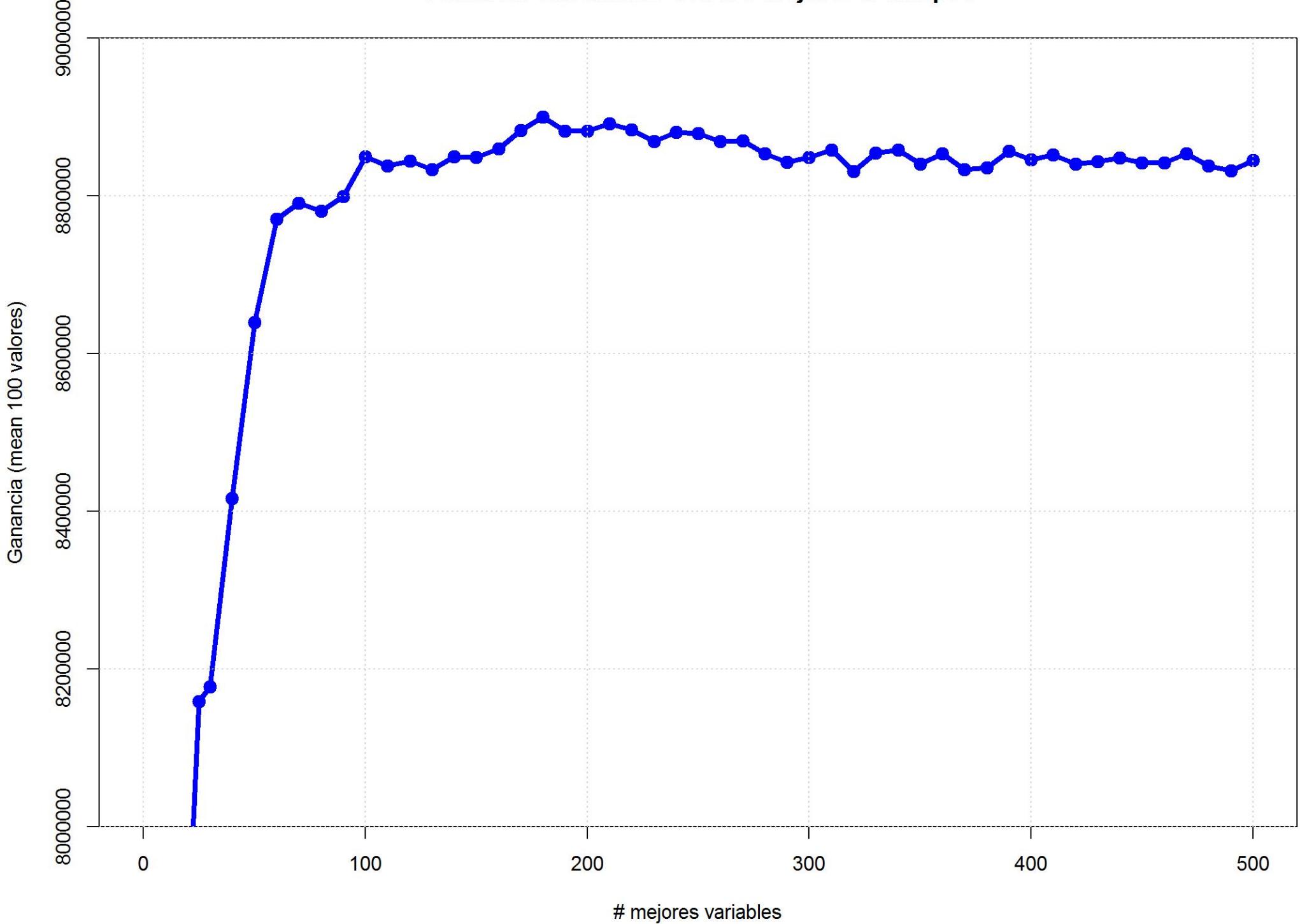
Se obtiene la lista ordenada de Feature Importance

Con esos hiperparametros,
tomando cada vez los
 n in $c(1,2,3,\dots,9, 10,20, \dots ,490,500)$ mejores features
se genera nuevos modelo en [201805,201902]
y aplica a [201904]
(se repite 100 veces para cada n)

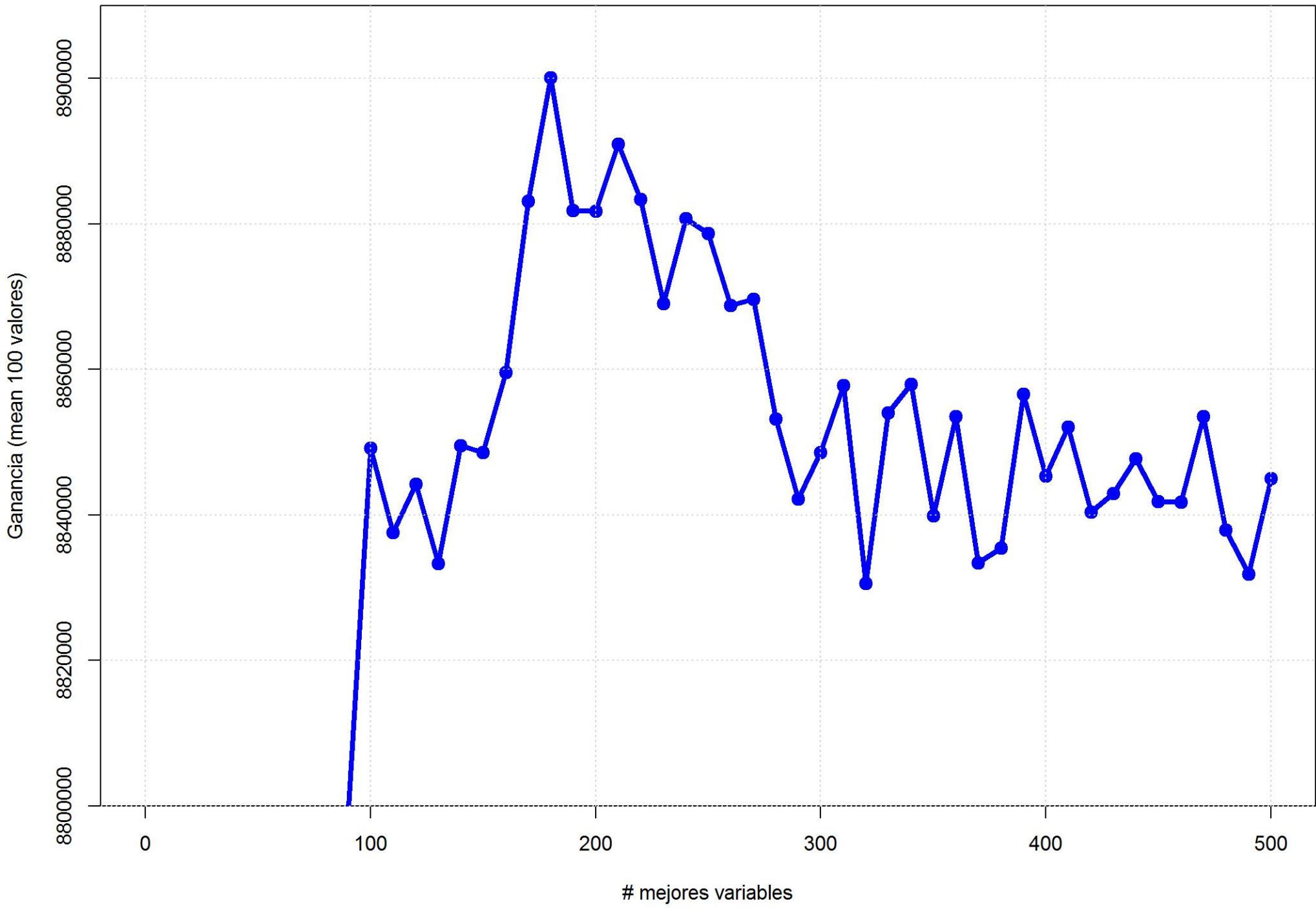
Ganancia entrenando con los mejores n campos



Ganancia entrenando con los mejores n campos



Ganancia entrenando con los mejores n campos



Resultado

La cantidad “apenas optima”
está alrededor
de entre los primeros 180 y 220
features mas importantes

Parte 3

Cómo ganarle
a la linea de muerte

sin embarrassarse

\$ 8,744,000 en 201906

hasta ahora ... WFV final

Determino parámetros óptimos mediante
100 rounds Bayesian Optimization en
Train=[201805, 201902] Validate=[201904]

Con esos parámetros óptimos entreno
el modelo final en 10 meses Train=[201807, 201904]

Aplico el modelo final a
[201906]

Empujando los límites

¿Cómo puedo manipular el walk forward validation tradicional?

Por un lado, esto es inalterable :

- El último mes con la clase completa es [201904]
- Dos modelos deben compararse (para elegir el mejor) en datos que no vieron al momento de generarse

Extendiendo WFV, parte 1

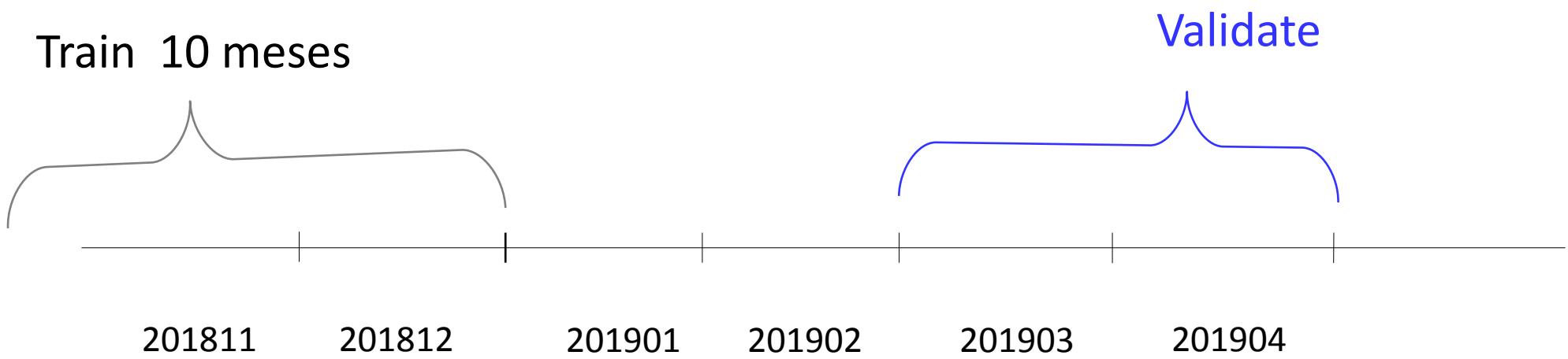
Determino parámetros óptimos mediante
Bayesian Optimization en
Train=[201804, 201901]
Validate=[201903, 201904]

Con esos parámetros óptimos entreno
el modelo final en 10 meses Train=[201807, 201904]

Aplico el modelo final a
[201906]

Bayesian Optimization

Train 10 meses



Validate
Ahora valido en DOS meses

una vuelta de rosca más

“poco miedo, mucha reflexión”

Extendiendo WFV, parte 2

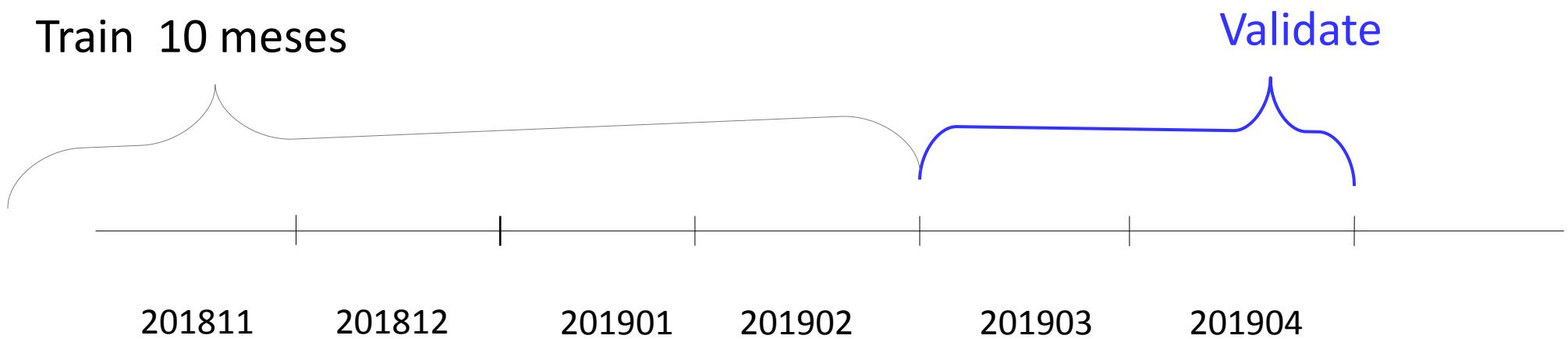
Determino parámetros óptimos mediante
Bayesian Optimization en
Train=[201805, 201902]
Validate=[201903, 201904]

Con esos parámetros óptimos entreno
el modelo final en 10 meses Train=[201807, 201904]

Aplico el modelo final a
[201906]

Ahora entreno lo mas cerca posible

Train 10 meses



Train=[201805, 201902]

Validate=[201903, 201904]

lightgbm

```
objective= "binary",
max_bin= 31,
num_leaves= 2048,
boost_from_average= TRUE,
learning_rate= 0.0292390925187225,
num_iterations= 469,
max_depth= 15,
feature_fraction= 0.523532764803938,
min_data_in_leaf= 172,
min_gain_to_split= 0.513112443440314,
lambda_l1= 2.88465299535605,
lambda_l2= 51.4982545289962
```

Ganancia en 201906

9,030,500

103.3%

\$ 8,744,000

FIN

1 O tiempo utilizando muy distintos y con las experiencias mejorando el rendimiento de los corredores.
Todos corrieron mundo business (la máquina no importó),
20 CPU, 500 GB memoria). El tiempo de la mejor máquina
LightGBM ~~no~~ más ~~de~~ mientras que la peor con el
LG Boost a los 4 AV terminó 9 contados ~~que~~ no más terminó
muy mal del corredor.