Instituto Tecnológico de Buenos Aires
Maestría en Ciencia de Datos
Data Mining , 2023 B

# Tarea para el Hogar DOS

Hágase amigo de los scripts, ejecútelos línea a línea y desde RStudio vaya siguiendo que va quedando en cada variable, entienda qué es lo que hacen, haga una copia y pruebe cambiar partes y observe como cambia el resultado, experimente, juegue !

Recuerde que la Sección Pasado está orientada a quienes no han podido asistir a algunas de las clases.

Varios de los ejercicios de esta Tarea para el Hogar pueden hacerse en menos de 2 minutos, ya que solo demandan renombrar scripts y cambiar la semilla.

Es uno de los objetivos de esta Tarea para el Hogar que usted adquiera destreza en el manejo de Google Cloud, modificar scripts, correrlos, analizarlos los resultados. Esto lo entrenará para cuando deba realizar su Experimento Colaborativo.

En esta tarea es fundamental que se de de alta en el documento de Google Slides de la presentación de Experimentos Compartidos, y que luego de hablar con sus compañeros más cercanos, elija un experimento anotando su nombre en la portada.
Los Grupo B pueden ser solamente de una persona y esta persona debe poseer computer literacy al menos intermedio y/o experiencia en ciencia de datos.
Para tener en cuenta más adelante, cuando usted escriba su diseño experimentan en el Google Slides deberá solicitar por Zulip aprobación del mismo al profesor.

# Sección  Pasado

Usted ya debería haber hecho esto en clase/durante la semana; esta sección está pensada para quienes faltaron a alguna clase.

## 1. *Videos Workflow  primera parte*

Prerequisito: *NA*

- Etapas Generales
  - Pres  Etapas Generales
  - Video Etapas Generales
- Catastrophe Analysis
  - Pres  Catastrophe Analysis
  - Video  Catastrophe Analysis

( tiempo estimado 12 minutos totales a 1.3x )

## 2. *Análisis Variables Rotas*

Analizará un problema que el sector de DataWarehouseing ha tenido en la generación de los datos.

Cree una máquina virtual  *fantasmita*  8 vCPU  64 GB de memoria RAM  y nómbrela `z505`
Una vez que se encendió la virtual machine, ingrese a VSCode  tal cual lo ha hecho antes
Ya en VSCode abra el script
`~/dm2023b/src/CatastropheAnalysis/z505_graficar_zero_rate.r`

Ponga a correr el script
Una vez que ha corrido el proceso, la máquina virtual se auto-suicida y su poder de computo regresa al datacenter, para reencarnarse en los siguientes segundos en otro ser.

La salida del script queda en su bucket, en  la carpeta  `~/buckets/b1/exp/CA5050` , analice el archivo identificando a partir de los gráficos las variables que tuvieron problema para algún mes, que el zeroes_ratio vale 1.0
¿En qué mes estan rotas más de 20 columnas del dataset?

tiempo humano estimado : **15 minutos**
dificultad : **baja**
creatividad requerida :  1**0%**
tiempo computacional: 1**0 minutos**

## 3. Análisis Data Drifting

Analizará el Data Drifting en las variables

Cree una máquina virtual *fantasmita* 8 vCPU 64 GB de memoria RAM y nómbrela `z361`
Una vez que se encendió la virtual machine, ingrese a VSCode tal cual lo ha hecho antes
Ya en VSCode abra el script `~/dm2023b/src/drifting/z361_graficar_densidades.r`

Ponga a correr el script, el proceso demorará alrededor de 10 minutos
Una vez que ha corrido el proceso, la máquina virtual se auto-suicida.

La salida del script queda en su bucket, en la carpeta `~/buckets/b1/exp/DR3610` , usted puede abrir los archivos desde la virtual machine `desktop` que vive en Sao Paulo o bajarlos a su PC.
Analice cuales son las variables que presentan más data drifting.

tiempo humano estimado : **15 minutos**
dificultad : **baja**
creatividad requerida : 1**0%**
tiempo computacional: 1**0 minutos**

## 4. Optimización Bayesiana rpart

Alternativamente al método de optimización de hiperparámetros Grid Search que utiliza la fuerza bruta, usted correrá un superador método llamado Optimización Bayesiana.

Cree una máquina virtual *fantasmita* 8 vCPU 64 GB de memoria RAM y nómbrela `z321`
Una vez que se encendió la virtual machine, ingrese a VSCode tal cual lo ha hecho antes
Ya en VSCode abra el script `~/dm2023b/src/rpart/z321_rpart_BO.r`
Cambie la semilla por SU primer semilla.
Ponga a correr el script, el proceso demorará alrededor de 60 minutos
Una vez que ha corrido el proceso, la máquina virtual se auto-suicida.

La salida del script queda en su bucket, en el archivo `~/buckets/b1/exp/HT3210/HT321.txt`
Copie esos hiperparámetros al script z101_PrimerModelo.R y suba el archivo generado a Kaggle.
¿Cómo compara esa ganancia con la que había obtenido con lo mejor de Grid Search?

tiempo humano estimado : **10 minutos**
dificultad : **baja**
creatividad requerida : 1**%**
tiempo computacional: 6**0 minutos**

## 5. Repensando el Overfitting

Qué efecto tiene agregar variables “canarito” (ruido blanco) al dataset y luego ejecutar el algoritmo rpart para crear un árbol de decisión?

Ingrese a la virtual machine `desktop` de Sao Paulo
Una vez que se encendió la virtual machine, ingrese a VSCode tal cual lo ha hecho antes
Ya en VSCode abra el script
`~/dm2023b/src/RepensandoOverfitting/z481_rpart_canaritos_prp.r`
Cambie la semilla por SU primer semilla en la línea 19
De las líneas 32 a 35 cambie los hiperparámetros del rpart por los que mejor le resultaron de su corrida de Grid Search (o si lo prefiere de su corrida de Optimización Bayesiana de rpart)

Ponga a correr el script, el proceso demorará alrededor de 10 minutos

La salida del script queda en su bucket, en el archivo
`~/buckets/b1/exp/EA4810/arbol_canaritos.pdf`
¿Aparecen canaritos en su árbol de decisión?¿Cuántos?¿A qué profundidades?
¿Qué pasa si en la línea 23 aumenta la cantidad de canaritos de 30 a 200 y vuelve a correr?
¿Qué pasa si prueba otros hiperparámetros del rpart y vuelve a correr?

tiempo humano estimado : **10 minutos**
dificultad : **baja**
creatividad requerida : **1%**
tiempo computacional: 10 **minutos**

## 6. Primera corrida LightGBM

Simplemente para probar el poder de LightGBM, el estado del arte en datos tabulares, ejecutará un script en donde la cátedra ha seleccionado en forma manual los hiperparámetros del algoritmo (más adelante usted deberá correr su optimización bayesiana y traer a este script los hiperparámetros que encuentre como óptimos)

Ingrese a la virtual machine `desktop` de Sao Paulo
Una vez que se encendió la virtual machine, ingrese a VSCode tal cual lo ha hecho antes
Ya en VSCode abra el script `~/dm2023b/src/lightgbm/z424_lightgbm_final.r`
Cambie la semilla por SU primer semilla en la línea 23

Ponga a correr el script, el proceso demorará alrededor de 10 minutos

Las archivos de salida quedan en su bucket, en la carpeta `~/buckets/b1/exp/KA4240/`
Suba todos esos archivos a Kaggle

¿Cómo compara la ganancia con la de arboles azarosos?

¿Qué sucede si hace una nueva corrida aumentando al doble el hiperparámetro `num_iterations`? ¿Es como árboles azarosos que a mayor cantidad de arbolitos mejora la ganancia?

tiempo humano estimado : **10 minutos**
dificultad : **baja**
creatividad requerida : 1**%**
tiempo computacional: 20 **minutos**

## Sección Deseable

## 7. Optimización Bayesiana LightGBM en Google Cloud

Prerequisito: el video *Corriendo un script de RStudio en Google Cloud*

Correrá una Optimización Bayesiana para el algortimo LightGBM para este < dataset, clase> con el objetivo de encontrar los hiperparámetros óptimos del LightGBM.
Esta es una tarea que un científico de datos hace casi a diario en el ejercicio de su profesión.

Cree una máquina virtual con 8 vCPU 16 GB de memoria RAM y nómbrela `z423`
Una vez que se encendió la virtual machine, ingrese a RStudio tal cual se explica en el instructivo, recordado que debe especificar una conexión no segura `http://`
Ya en RStudio abra el script `~/dm2023b/src/lightgbm/z423_lightgbm_binaria_BO.r`
Cambie en la línea 45 por su primer semilla
Cambie en la línea 53 por su segunda semilla

De las decenas de hiperpárámetros posibles de la función LightGBM solamente cuatro participan de la Optimización Bayesiana :

`{ learning_rate, feature_fraction, num_leaves, min_data_in_leaf }`

Es lo que en el código aparece como

```
#Aqui se cargan los bordes de los hiperparametros
PARAM$hyperparametertuning$hs  <- makeParamSet(
         makeNumericParam("learning_rate",    lower=    0.01, upper=    0.3),
         makeNumericParam("feature_fraction", lower=    0.2 , upper=    1.0),
         makeIntegerParam("min_data_in_leaf", lower=    1L  , upper=  8000L),
         makeIntegerParam("num_leaves",       lower=   16L  , upper=  1024L),
         makeIntegerParam("envios",           lower= 5000L  , upper= 15000L)
        )
```

El resto de los hiperparámetros está fijo en la siguiente parte del script

```
 param_basicos  <- list( objective= "binary",
                         metric= "custom",
                         first_metric_only= TRUE,
                         boost_from_average= TRUE,
                         feature_pre_filter= FALSE,
                         verbosity= -100,
                         max_depth=  -1,         # -1 significa no limitar
                         min_gain_to_split= 0.0, #por ahora, lo dejo fijo
                         lambda_l1= 0.0,         #por ahora, lo dejo fijo
                         lambda_l2= 0.0,         #por ahora, lo dejo fijo
                         max_bin= 31,            #por ahora, lo dejo fijo
                         num_iterations= 9999,   #un numero muy grande, ...
                         force_row_wise= TRUE,
                         seed= PARAM$hyperparametertuning$semilla_azar

                        )
```

Es muy llamativo que no se esté intentando optimizar { `lambda_l1,lambda_l2, min_gain_to_split }` hiperparámetros que procuran la tan ansiada regularización, idea esencial introducida por XGBoost/LightGBM en el marco de GBDT .

Operativamente en este script, si usted quisiera agregar un hiperparámetro a la Optimización Bayesiana debe agregarlo en `makeParamSet()` con la función `makeNumericParam()` ya que `lambda_l1, lambda_l2 y min_gain_to_split` son números reales, debe quitarlos de `param_basicos`

Revise bibliografía de los hiperparámetros de LightGBM que se suelen optimizar, modifique el script.

Además consúltele a su gran amigo ChatGPT por los rangos en los que le conviene hacer variar `lambda_l1` , `lambda_l y min_gain_to_split . ¿`Le dará ChatGPT una respuesta correcta? La optimización bayesiana se lo dirá !

La cátedra, le deja estos populares artículos, pero no se hace responsable de la pertinencia de los mismos. Busque usted los artículos de su agrado, y no crea en todo el humo que le venden.

- https://lightgbm.readthedocs.io/en/latest/Parameters-Tuning.html
- https://towardsdatascience.com/kagglers-guide-to-lightgbm-hyperparameter-tuning-with-optuna-in-2021-ed048d9838b5
- https://neptune.ai/blog/lightgbm-parameters-guide
- https://medium.com/analytics-vidhya/hyperparameters-optimization-for-lightgbm-catboost-and-xgboost-regressors-using-bayesian-6e7c495947a9
- https://gist.github.com/vacarezzad/6e9f1c34513417c81f6546dcd81a7303

Una vez que hizo las modificaciones, ponga a correr el script
El proceso demorará alrededor de 6 horas
Monitoree el avance del proceso, a partir de los 20 minutos van apareciendo mensajes por la pantalla del tipo `[mbo] 1:` , `[mbo] 2:` , hasta `[mbo] 100`:
Si se le apagara la máquina virtual, debe primero ELIMINARLA en caso que aún siga existiendo, volver a crear una nueva de cero, y volver a correr el script, el que retomará desde donde quedó.

Al finalizar le saldrá el mensaje `“La Optimizacion Bayesiana ha terminado”`
La salida del script queda en su bucket, en la carpeta , el archivo se llama. Ingresar al bucket desde https://console.cloud.google.com/storage/browser/
Baje el archivo que quedó en el bucket en `exp/HT4230/HT4230.txt` a una carpeta homónima en su laptop, el archivo deberá tener una primer linea de títulos y luego 120 lineas.

Consulte en Zulip a sus compañeros todo lo que haga falta.

tiempo humano estimado : **30 minutos**
dificultad : **muy alta**, por ser la primera vez
creatividad requerida : **0%**
tiempo computacional: **6 horas**

## 8. Generación de Modelo Final (Google Cloud)

Prerrequisito: el punto anterior.

En el punto anterior luego de varias horas de corrida, una insoportable Optimización Bayesiana ha generado los hiperparámetros óptimos del LightGBM

Cargue en un Excel el archivo salida de la optimizacion bayesiana, el `HT4230.txt` que usted ya bajó en el punto anterior a su laptop
Ordene toda la planilla por la penúltima columna, `ganancia` , en forma descente, le quedará en la primer linea los hiperparámetros óptimos del LightGBM para ese <dataset, clase> , toda la corrida anterior fue unicamente para obtener esa primer línea !

Cree una máquina virtual con 8vCPU 16GB de memoria RAM y nómbrela `z424`
Corra el script `~/dm2023b/src/lightgbm/z424_lightgbm_final.r`

Cambie en la linea 23 por su primer semilla
Reemplace los hiperparámetros de las lineas 25 a la 29 por los que están en la primera línea de la planilla, los que generan la mejor ganancia

Corra el script
En caso que le apareciera el mensaje `[LightGBM] [Warning] No further splits with positive gain, best gain: -inf` , no se asuste, está todo bien.
Al finalizar le saldrá el mensaje "`La generacion de los archivos para Kaggle ha terminado`"

La salida queda en la carpeta `~/buckets/b1/exp/KA4240/` se llaman `KA4240_8000.csv`, `KA4240_8500.csv` hasta `KA4240_12000.csv`
Suba esos archivos a Kaggle

Cargue los resultados obtenidos en la hoja `LightGBM` de la planilla colaborativa Google Sheets

- ganancia_BO es la ganancia que le mostró la Bayesian Optimization
- mejor_archivo es el nombre del archivo que le dio mayor ganancia en el Public Leaderboard.
- La ultima columna, Public Leaderboard, es la mayor ganancia que obtuvo en el Public Leaderboard.

tiempo humano estimado : **15 minutos**
dificultad : **baja**
creatividad requerida : **0%**
tiempo computacional: **5 minutos**

## 9. Videos Workflow  segunda parte

Prerequisito: *NA*

- Data Drifting
  - Pres  Data Drifting
  - Video  Data Drifting
- Feature Engineering IntraMes
  - Pres  Feature Engineering IntraMes
  - Video  FeatureEngineering IntraMes
- Feature Engineering Histórico
  - Pres  Feature Engineering Histórico
  - Pres  Feature Engineering Histórico
- Training Strategy
  - Pres  Training Strategy
  - Video  Training Strategy
- Hyperparameter Tuning
  - Pres  Hyperparameter Tuning
  - Video  Hyperparameter Tuning
- Etapas Finales
  - Pres  Etapas Finales
  - Video  Etapas Finales

usted debe mandatoriamente ver estos videos antes de continuar con el resto de la Tarea para el Hogar.

( tiempo estimado 40 minutos totales a 1.5x,  dificultad media)

## 10. Registrándose en Experimentos Colaborativos

Estando conectado en su browser a una cuenta de Google, ingrese link restringido de Google Slides https://docs.google.com/presentation/d/1pIAN9bAHLdASOuDBZrIvpOB1Y-e_PsUow0ev1Z8O-Zk/edit?usp=sharing , le aparecerá una pantalla con un botón azul "Solicitar Accesso", presiónelo, y luego de algunas horas recibirá un email (a su cuenta de gmail desde la cual solicitó la autorización) donde se le notificará que ya tiene acceso (se lo otorgó un profesor de la cátedra).

## 11. Modificaciones a scripts para hacer su corrida

Los siguientes pasos son extremadamente sencillos y mecánicos, solo requieren prestar atención. Usted modificará los seis scripts enla máquina virtual `desktop` para luego poder correrlos un una potente máquina virtual del tipo *fantasmita* .

Cuando se le solicite que haga una copia, por ejemplo del script
`~/dm2023b/src/workflow-inicial/z611_CA_reparar_dataset.r`

usted deberá cambiarlo a por ejemplo
`611_CA_reparar_dataset_01.r`

en donde:

- le ha quitado la "z" inicial del nombre
- le agregó un `_01` al final del nombre

recuerde que hemos adoptado la convención de NO modifcar los scripts que comienzan con la letra "z", prefijo que reservamos para los scripts originales provistos por la cátedra.

## 12. Modificaciones al script de CA Catastrophe Analysis

Corregirá el problema de las variables rotas ( pisadas en cero )

Hay dos métodos disponibles.

- El llamado “EstadisticaClasica” al valor que fue pisado en cero para ese mes, lo imputa como el promedio del mes anterior y siguiente.
- El llamado “MachineLearning” al valor que fue pisado en cero para ese mes, lo imputa con un NA, ha leído bien, somos tan diabólicos que agregamos nulos al dataset.

Quizas usted piense que esta última estrategia es imposible que funcione, pero recuerde la máxima de esta asignatura:

Un experimento no se le niega a nadie.

Habrá un equipo que en Experimentos Colaborativos comparará ambos métodos.
Siempre podrá usted modificar el código, agregar nuevas formas de corregir las variables que están rotas.

En la máquina vitual `desktop` haga su copia de trabajo del script
`~/dm2023b/src/workflow-inicial/z611_CA_reparar_dataset.r`
Si su nombre pertenece a este conjunto
{Axel Ilicic}

*esta primera vez* cambie en la actual donde dice

1. `PARAM$metodo <- "MachineLearning"`

   por

1. `PARAM$metodo <- “EstadisticaClasica”`

`( el resto de los alumnos debe elegir o “MachineLearning” o “Ninguno”)`

Agregue el nuevo archivo al repositorio, haga el commit, y sincronícelo con su repositorio en GitHub
Cargue en la planilla colaborativa sus parámetros elegidos

`tiempo humano estimado : 5 minutos`

## 13.Modificaciones al script de DR Data Drifting

Intentará corregir el data drifting existente en el dataset.
Habrá un equipo que en Experimentos Colaborativos comparará en detalle como funcionan los métodos disponibles aplicados a este dataset.
Siempre podrá usted modificar el código, agregar nuevas formas de corregir el drifting.

Haga su copia de
`~/dm2023b/src/workflow-inicial/z621_DR_corregir_drifting.r`

Si usted posee una carrera de grado que pertenece a
`{ Contador, Administración de Empresas, Economia, Ingeniería Industrial }`

cambie donde dice:

```
1. PARAM$metodo  <- "rank_cero_fijo"
```

por

```
1. PARAM$metodo  <- "deflacion"
```

en caso contrario, elija alguno de los métodos disponibles

Agregue el nuevo archivo al repositorio, haga el commit, y sincronícelo con su repositorio en GitHub
Cargue en la planilla colaborativa sus parámetros elegidos

`tiempo humano estimado : **5 minutos**`

## 14.Modificaciones al script FE Feature Engineering

Agregará nuevas variables históricas al dataset.
Habrá varios equipos que en Experimentos Colaborativos compararán los métodos disponibles.

Se usted posee conocimientos previos, se lo invita a que modifique el código, a agregar nuevas variables históricas, por ejemplo, la derivada segunda.

Haga su copia de `~/dm2023b/src/workflow-inicial/z631_FE_historia.r`

Parámetros del script que puede cambiar a gusto

- `PARAM$lag1` agrega para cada variable el valor del mes anterior
- `PARAM$lag2` agrega para cada variable el valor de dos meses antes
- `PARAM$lag3` agrega para cada variable el valor de tres meses antes
- `PARAM$Tendencias1` agrega para cada variable la pendiente que ajusta por cuadrados minimos el valor de ese mes y los cinco meses anteriores
- `PARAM$RandomForest` agrega variables nuevas a partir de los arboles de un Random Forest de baja profundidad.
- `PARAM$CanaritosAsesinos` elimina variables que son menos importantes que los canaritos.

Agregue el nuevo archivo al repositorio, haga el commit, y sincronícelo con su repositorio en GitHub
Cargue en la planilla colaborativa sus parámetros elegidos

`tiempo humano estimado : 5 minutos`

## 15.Modificaciones al script TS Training Strategy

Decidirá en que meses se < entrena, valida, testea> y en que meses se realiza el <train_final>
Si coinciden los meses de entrenamiento y validacion aguas abajo la Optimización de Hiperparámetros se realizará utilizando cross validation.

Dados los largos tiempos de procesamiento de la Optimización de Hiperparámetros posterior que depende de la cantidad de meses donde se entrena, habrá varios equipos que en Experimentos Colaborativos realicen experimentos para determinar que es lo que funciona mejor.

Haga su copia del script
`~/dm2023b/src/workflow-inicial/z641_TS_training_strategy.r`

Cambie por *su* semilla el parámetro correspondiente

Generalmente entrenar en más meses genera un mejor modelo predictivo, en la medida que no se incluyan los meses más duros de la pandemia que en Argentina significaron muy estrictas restricciones a la circulación.

Agregue el nuevo archivo al repositorio, haga el commit, y sincronícelo con su repositorio en GitHub
Cargue en la planilla colaborativa sus párametros elegidos

`tiempo humano estimado : 5 minutos`

## 16. Modificaciones al script HT Hyperparameter Tuning

Haga su copia del script

`~/dm2023b/src/workflow-inicial/z651_HT_lightgbm.r`

Cambie el parámetro de la semilla por su semilla

Agregue el nuevo archivo al repositorio, haga el commit, y sincronícelo con su repositorio en GitHub

Cargue en la planilla colaborativa sus parametros elegidos

`tiempo humano estimado : 5 minutos`

## 17.Modificaciones al script ZZ Pasos Finales

Haga su copia del script
`~/dm2023b/src/workflow-inicial/z661_ZZ_final.r`

Cambie en `PARAM$semillas` por sus cinco semillas
No debe hacer ningun otro cambio

Agregue el nuevo archivo al repositorio, haga el commit, y sincronícelo con su repositorio en GitHub
Cargue en la planilla colaborativa sus parametros elegidos

Las salidas quedan en el bucket `./exp/ZZ6610`

- Archivos con los modelos de LightGBM en formato binario, **FM** Final Models
  - `modelo_01_xxx.model` y `modelo_02_yyy.model`
- Archivos con la importancia de variables
  - `impo_01_xxx.txt` y `impo_02_yyy.txt`
- Archivos con las probabilidades **SC** Scoring
  - `pred_01_xxx.csv` y `impo_02_yyy.txt`
- Archivos generados para **KA** Kaggle
  - `ZZ6610_01_xxx_09500.csv` al `ZZ5910_01_xxx_11500.csv`
  - `ZZ5910_02_yyy_09500.csv` al `ZZ5910_02_yyy_11500.csv`
  - El `01` y `02` indican que son el mejor y el segundo mejor modelo encontrados en la optimización bayesiana
  - `xxx` e `yyy` son el numero de iteración de esos modelo en la optimización bayesiana
  - `09500`, `10000`, 10500, `11000`,`11500` son la cantidad de estímulos a enviar

*tiempo humano estimado :* **5 minutos**

## 18.Modificaciones al script `correr_workflow`

Haga su copia del script
`~/dm2023b/src/workflow-inicial/z601_RUN_correr_workflow.r`
con el nombre
`601_RUN_correr_workflow_01.r`

Haga los cambios en las lineas `source()` para que se reflejen sus nombres de scripts, generalmente alcanzará con que elimine la letra z
por ejemplo

```
source( "~/dm2023b/src/workflow-inicial/z611_CA_reparar_dataset.r")
```

quitando la “z” de “z611” pasa a :

```
source( "~/dm2023b/src/workflow-inicial/611_CA_reparar_dataset_01.r")
```

Agregue el nuevo archivo al repositorio, haga el commit, y sincronícelo con su repositorio en GitHub

*tiempo humano estimado :* **8 minutos**

## 19.Finalmente Corrida del Workflow

Deberá seguir los pasos que están el el documento de Instalación de Google Cloud capítulo *3.4 Crear una máquina virtual desde template*
Cree una virtual machine con 256 GB de memoria RAM y 8 vCPU a partir del correspondiente template.
Asigne el nombre `workflow-inicial-01` elija la región `us-west4 (Las Vegas)`

Una vez que se creó la máquina virtual deberá seguir los pasos que están en el capítulo *3.5 RStudio en forma remota* para correr el RStudio en forma remota en la máquina virtual .
Alternativamente también puede ingresar por la terminal remota a la máquina virtual fantasmita y desde el VSCode correr el script

Una vez que ingresó al RStudio busque su script creado en el paso anterior
`~/dm2023b/src/workflow-inicial/601_RUN_correr_workflow_01.r`
y córralo linea a linea hasta `TS_training_strategy.r` inclusive

finalmente, ponga a correr de ahí en adelante

Los resultados se irán generando dentro de `~/buckets/b1/exp` en las siguientes carpetas
`CA6110`
`DR6210`
`FE6310`
`TS6410`
`HT6510`
`ZZ6610`

En cada carpeta queda un archivo llamado `output.yml`
Analice que queda en cada una de las carpetas.

Una vez que finalizó todo, desde la máquina virtual de Sao Paulo `desktop` suba a Kaggle los archivos generados que quedan en el ~/buckets/b1/exp/ZZ6610/ , archivos de la forma `ZZ6610_*.csv`

`tiempo humano estimado :` ***`20 minutos`***
`tiempo computacional :` **`12 horas`**