Uso de R en ambiente productivos.

R como lenguaje es cada día más usado en el ámbito corporativo y no solo académico, sin embargo dentro de las corporaciones muchas veces es importante que los modelos desarrollados queden en ambientes productivos teniendo que cuidar entonces temas como: el performance (cuando se trata de un sistema que debe calificar en tiempo real), Tiempos de procesamiento y posibles overflow de memoria (calificar toda la población de una entidad bancaria). Es por eso que a continuación se explicará como abordar estos problemas sin tener que reescribir los desarrollos R en otro lenguajes como Python, Java, etc.

En este caso se expondrá el uso de herramientas complementarias a R como son h2o y Docker. H2o es un Framework de Inteligencia Artificial, que posee muchos algoritmos tradicionales y no tan tradicionales, con la ventaja de que su procesamiento se hace de manera paralelizada e incluso se puede ejecutar en un cluster Spark, este framework para los conocedores de Caret les puede ser un simil. Por otro lado Docker es un proyecto de código abierto que automatiza el despliegue de aplicaciones dentro de contenedores de software, proporcionando una capa adicional de abstracción y automatización de virtualización de aplicaciones en múltiples sistemas operativos

Para tal motivo se presentarán dos posibilidades, donde claramente se pueden combinar las mismas para hacer aún más eficiente R.

1. R + h2o: Para este escenario se usará R más una versión de h2o (Framework de AI, https://www.h2o.ai/). La idea es procesar toda la data mediante cualquier esquema, realizar cualquier análisis previo a un modelo de ML y posteriormente realizar el modelo bajo h2o desde R. Ya con un modelo generado sobre h2o, el mismo puede ser extraído como objeto y ser cargado y consumido desde otro R, sin embargo, siempre tendremos las limitantes de versiones de R y h2o (caso que veremos más adelante), más los tiempos de respuesta. Por eso se recomienda que el modelo generado sea descargado como un objeto Java (POJO o MOJO). De esta manera se puede compilar sobre cualquier arquitectura y así poner en producción de manera muy rápida y sobre todo eficiente. Bajo este esquema se genera una clase Java que prácticamente no depende de nada salvo un ambiente Java que como bien sabemos hoy día todos los sistemas operativos poseen una versión open u no open.

2. R + h2o + Docker: En este escenario vamos a pensar que no se trata solo del modelo ML, si no que se necesita que el flujo de predicción conlleve algo de tratamiento previo de data, sobre esto y pensando en el escenario anterior se pudiese pensar en trasladar todo el código a Java, sin embargo, ya se pierde

el sentido de usar y programar en R. Para esto la posible solución sería crear un contenedor (Docker) con R y el modelo en h2o no como objeto Java si no como objeto de R. De esta manera toda la aplicación, los códigos, y en fin todo el desarrollo cuidando librerías y dependencias externas a R se pueden mantener sin ningún problema. Ese contenedor se puede extraer del computador del creador y ser puesto en otro ambiente sin ningún problema siempre y cuando ambos sistemas cuenten con el respectivo Docker. Ni siquiera hace falta que ambos tengan R, h2o o cualquier otra librerías o dependencia. Claramente con este esquema se debe tener cuidado extra si el tiempo de respuesta debe ser acorde a una transacción en línea (milesimas de segundos). Allí se debería pensar en usar esquemas de APIS con plumber o temas que no trataremos en esta ocasión.

Hoy día en empresas de Buró como es Experian estás metodologias se están adoptando con fuerza, de está manera los desarrollos analíticos pueden llegar a implementarse sin problemas. Ya sea en arquitecturas On Premise u arquitecturas Cloud.