Sobre el script *reservoir\_sampling.R*

Después de realizar algunas sutiles diferencias al código reportamos lo siguiente:

¿Qué hace el código?

Efectivamente el código realiza un muestro secuencial de un archivo, puede definirse como un muestro secuencial de aceptación y rechazo, en general siguiendo las ideas proporcionadas en el seudocódigo de la clase del mismo tema sin embargo tiene algunas diferencias que enumeramos en los comentarios.

El código requiere de tres parámetros, la ruta del archivo a leer, *n\_muestras* que es el número final de filas en la muestra y *n\_filas read* que es el número de filas que se leen secuencialmente del archivo a muestrear en cada iteración.

El código comienza reservando en memoria las primeras *n\_muestras* del archivo original, posteriormente abre una conexión de lectura al archivo que se desea muestrear (lo cual agiliza la lectura del archivo y permite mantener fácilmente el índice del archivo sigue en la lectura).

En cada iteración se leen *n\_filas\_read* y posteriormente se realiza un muestreo sin reemplazo de los índices que ya se han leído del archivo original (para ‘agilizar’ el calculo se vectorizo esta operación con otra función), luego se eligen los índices que son menores a los de nuestra muestra y se sustituyen tales registros de nuestra muestra original por los recientemente leídos.

Finalmente escribimos los resultados en un formato csv para mantener portabilidad entre sistemas operativos. Porque cuando las cosas se hacen en serio no se hacen en Windows.

Comentarios del código.

En estricto sentido este código no realiza el seudocódigo proporcionado en clase. Reserva en un paso intermedio espacio en memoria para un conjunto bastante elevado de registros lo cual tiene una desventaja, para hacer este código similar al seudocódigo proporcionado tendríamos que fijar el parámetro *n\_filas\_read* a el valor de 1, y tardaría 3 días en ejecutarse debido a que la complejidad en tiempo del algoritmo codificado es mientras que su complejidad en espacio es .

La vectorización del muestreo es innecesaria pues la función sample.int del kernel base de R ya lo es, eso solo resta eficiencia al código.

Sobre el script *Estocastico\_JessicaVega.R*

El código tiene como entrada el archivo generado por el código anterior, una muestra aleatoria del archivo original.

Después de leer el archivo y separar la variable a predecir (*y*), el conteo del número de veces que un ítem fue retirado en un mes de un año de la biblioteca de Seattle, se manipulan las otras 10 variables exógenas o que utilizaremos para predecir la matriz *X*.

De las 10 variables restantes tomamos el año y mes de retiro y las normalizamos para que la escala en que se miden no sesgue su aporte en la predicción, además añadimos a la matriz *X*  una columna de unos para representar el intercepto de la regresión lineal múltiple.

En cuanto al conjunto restante de las otras 8 variables, realizar un one hot encoding no es practico en R pues implementarlo en un conjunto de variables con 8 columnas de tipo factor con miles de factores cada una tendría como resultado una matriz con centenas de miles de columnas lo cual origina un *data.frame* del orden de decenas de GB en memoria RAM, para esquivar este problema realizamos el siguiente encoding:

Para cada columna de tipo factor reemplazamos el dato por su nivel tal como se guarda a nivel memoria en el lenguaje, es decir en lugar de guardar su etiqueta trabajamos con su valor numérico, como consecuencia de este encoding ya se cuenta con un *data.frame* con columnas numéricas exclusivamente a las cuales aplicamos la función de normalización para que de igual manera su escala de medición no sesgue los resultaos de predicción, que en este caso se relacionada directamente con el numero de niveles en cada columna.

Finalmente realizamos el ciclo que implementa la multiplicación de matrices y la actualización de los costos, pesos, derivadas y coeficientes .

Mostramos un gráfico de la ejecución que se obtuvo de correr el código con 1,000 epochs donde vemos que el costo parece ser asintótico al valor 33 .

