Ejercicios de R

Curso: Introducción a la Estadística y Probabilidades CM-274

Lecturas Importantes

1. Una introducción visual al Machine Learning
   `http://www.r2d3.us/visual-intro-to-machine-learning-part-1/`.
2. Artículo de James Le
   `http://www.kdnuggets.com/2016/08/10-algorithms-machine-learning-engineers.html` sobre algunos de los más importantes algoritmos del Machine Learning.

Preguntas

1. Se puede crear un `array` de prueba de 3 dimensiones, de la siguiente manera

```
> p_Array <- array( sample( 1:60, 60, replace=F), dim=c(5,4,3) )
```

La expresión anterior produce un array $5 \times 4 \times 3$, que puede representado matemáticamente como

$$\{x_{i,j,k} : i = 1,2,\ldots,5; j = 1,2,3,4; k = 1,2,3\}$$

Además

```
> apply(p_Array, 3, tmpFn)
```

significa que el índice k es guardado en la respuesta y la función `tmpFn` es aplicado a las 3 matrices $\{x_{i,j,1} : 1 \leq i \leq 5; 1 \leq j \leq 4\}$, $\{x_{i,j,2} : 1 \leq i \leq 5; 1 \leq j \leq 4\}$ y $\{x_{i,j,3} : 1 \leq i \leq 5; 1 \leq j \leq 4\}$.
Similarmente

```
> apply(p_Array, c(3, 1), tmpFn)
```

significa que los índices i y k son guardados en las respuestas y la función `tmpFn` es aplicado a los 15 vectores

$\{x_{i,j,1} : 1 \leq j \leq 4\}$, $\{x_{i,j,2} : 1 \leq j \leq 4\}$, etc.

La expresión anterior, hace la misma operación, pero el formato de la respuesta es diferente: al usar `apply` de esta manera, siempre vale la pena escribir un pequeño ejemplo para comprobar que el formato de la salida de `apply` es como se espera.

(a) Escribe una función `p_Fn` que toma un sólo argumento, que es una array de dimensión 3. Si este array es notado por $\{x_{i,j,k} : i = 1,2,\ldots,d_1; j = 1,2,\ldots,d_2; k = 1,2,\ldots,d_3\}$ entonces a función `tmpFn` retorna una lista de la matriz $\{w_{i,j,k}\}$ de orden $d_1 \times d_2 \times d_3$ y la matriz $\{z_{i,j}\}$ de orden $d_2 \times d_3$, donde

$$w_{i,j,k} = x_{i,j,k} - \min_{i=1}^{d_1} x_{i,j,k} \quad \text{y} \quad z_{j,k} = \min_{i=1}^{d_1} x_{i,j,k} - \max_{i=1}^{d_1} x_{i,j,k}$$

(b) Escribe una función `p_Fn2`, que retorna una matriz $\{z_{j,k}\}$ de orden $d_2 \times d_3$, donde

$$z_{j,k} = \sum_{i=1}^{d_1} x_{i,j,k}^k.$$

2. Un camino aleatorio simétrico empieza en el origen y es definido como sigue: Supongase que $X_1, X_2, \ldots$ son variables aleatorias idénticamente distribuidas independientes con la siguiente distribución

$$\begin{cases} +1 & \text{con probabilidad} \quad 1/2 \\ -1 & \text{con probabilidad} \quad 1/2 \end{cases}$$

Definimos la secuencia $\{S_n\}_{n\geq 0}$ como

$$\begin{aligned} &S_0 = 0 \\ &S_n = S_{n-1} + X_n, \quad \text{para} \quad n = 1, 2, \ldots \end{aligned}$$

Entonces $\{S_n\}_{n\geq 0}$ es un camino aleatorio simétrico empezando en el origen. La posición del camino aleatorio en el tiempo n es la suma de los previos pasos : $S_n = X_1 + \cdots X_n$.

(a) Escribe una función `rcamino(n)` que toma un argumento n y retorna un vector el cuál es una realización de $(S_0, S_1, \ldots, S_n)$ las primeras n posiciones de un camino aleatorio simétrico que empieza en el origen. El código siguiente

```
> sample( c(-1,1), n, replace=TRUE, prob=c(0.5,0.5) )
```

simula n pasos.

(b) Escribe una función `rcaminoPos(n)` que simula el hecho que un camino dura para una longitud de tiempo n y que devuelve la longitud de tiempo del camino que pasa por encima del eje X. Debes observar que un camino con longitud 6 y vértices en $0, 1, 0, -1, 0, 1, 0$ está 4 unidades de tiempo por encima del eje X y 2 unidades de tiempo por debajo del eje X).

3. El conjunto de datos `faithful` contiene la duración (en minutos) `eruptions` y el tiempo de espera hasta otra erupción `waiting`(en minutos) de para un geyser Old Faithful. Estamos interesados en conocer la relación que hay entre las dos variables.

(a) Crea una variable factor `longitud` que es **t_erup1** si la erupción es menor que 3.2 minutos y **t_erup2** en otros casos.

(b) Usa la función `bwplot` en el paquete `lattice`, para construir un gráfico (diagrama de cajas paralelos) de los tiempos de espera para las erupciones **t_erup1** y **t_erup2**.

(c) Usa la función `densityplot` construye un gráfico (de densidades superpuestas) de los tiempos de espera para las erupciones **t_erup1** y **t_erup2**.

En el problema anterior, se compararon los tiempos de espera de los géiseres Old Faithful para las erupciones **t_erup1** y **t_erup2** donde la variable `longitud` en el data frame `faithful` define la duración de la erupción.

(d) Supongamos un data frame `dframe` que contiene una variable numérica `num.var` y un factor `factor.var`. Después de que el paquete `ggplot2` se halla cargado, entonces, los comandos de R

```
> ggplot(dframe, aes(x = num.var, color = factor.var))
> + geom_density()
```

construirán estimaciones de densidades superpuestas de la variable `num.var` para cada valor del factor `factor.var`. Utiliza estos comandos para construir estimaciones de densidades superpuestas de los tiempos de espera de los géiseres con erupciones **t_erup1** y **t_erup2**.

(e) Con un data frame `dframe` que contiene una variable numérica `num.var` y un factor `factor.var`, la sintaxis de `ggplot2`

```
> ggplot(dframe, aes(x = num.var, color = factor.var))
> + geom_boxplot()
```

construirá caja de bloques paralelos de la variable `num.var` para cada valor del factor `factor.var`. Utiliza estos comandos para construir cajas de bloques paralelos de los tiempos de espera de los géiseres con erupciones **t_erup1** y **t_erup2**.

Sugerencia: Revisa el siguiente ejemplo

```
> library(ggplot2)
>
> #datos de muestra
>
> dat <- data.frame(dens = c(rnorm(100), rnorm(100, 10, 5))
+                   , lines = rep(c("a", "b"), each = 100))
> #Plot.
> ggplot(dat, aes(x = dens, fill = lines)) + geom_density(alpha = 0.5)
```

4. Supongamos que se está interesado en mostrar 3 miembros de la familia de curvas beta, donde la densidad con parámetros a y b, denotados por `Beta(a, b)` es dado por

$$f(y) = \frac{1}{B(a,b)} y^{a-1}(1-y)^{b-1}, \ \ 0 < y < 1.$$

Se puede dibujar un sola densidad beta, con paramétros $a = 5$ y $b = 2$, usando la función `curve`:

```
> curve(dbeta(x, 5, 2), from=0, to=1)
```

(a) Usa tres aplicaciones de la función `curve` para mostrar las densidades `Beta(2, 6)`, `Beta(4, 4)` y `Beta(6, 2)` en un sólo gráfico.

(b) Usa el siguiente comando de R, para colocar un título al gráfico de las ecuaciones de densidad beta

```
> title(expression(f(y)==frac(1,B(a,b))*y^{a-1}*(1-y)^{b-1}))
```

(c) Usa la función `text`, para etiquetar cada una de las curvas betas con sus correspondientes valores de los paramétros a y b.

(d) Redibuja el gráfico usando diferentes colores o tipos de líneas para las tres curvas de densidad.

5. Explica el siguiente código y el Teorema del Límite Centrak. La respuesta debe ser correctamente escrita y ordenada

```
> iter = 2000
> avg0 <- avg1 <- avg2 <- avg3 <- rep(0, iter)
> for (i in 1:iter) {
+     S = rexp(90)  # muestra desde la distribucion   exp(1)
+     avg0[i] = S[1]
+     avg1[i] = mean(S[1:3])
+     avg2[i] = mean(S[1:30])
+     avg3[i] = mean(S[1:90])
+ }
> SR = stack(list(`n=1` = avg0, `n=3` = avg1, `n=30` = avg2,
```

```
+     `n=90` = avg3))
> names(SR) = c("promedios", "n")
> ggplot(SR, aes(x = averages, y = ..density..)) + facet_grid(n ~
+     .) + geom_histogram() + scale_x_continuous(limits = c(0,
+     3))
```

6. Escribe una función llamada `listaFN` que toma un único argumento n e implementa el siguiente algoritmo

   (a) Simula n números independientes, denotado por $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ desde la distribución normal estándar.

   (b) Calcula la media $\bar{\mathbf{x}} = \sum_{j=1}^{n} x_j / n$.

   (c) Si $\bar{\mathbf{x}} \geq 0$, entonces simula n números independientes, denotados por $\mathbf{y} = (y_1, y_2, \ldots, y_n)$ desde la densidad exponencial con media $\bar{\mathbf{x}}$.

   (d) Si $\bar{\mathbf{x}} < 0$, entonces simula n números independientes, denotados por $\mathbf{z} = (z_1, z_2, \ldots, z_n)$ desde la densidad exponencial con media $-\bar{\mathbf{x}}$. Se coloca $\mathbf{y} = (y_1, y_2, \ldots, y_n) = $ **-z**.

   (e) Calcula k que es el número j con $|y_j| > |x_j|$.

   (f) Retorna la lista de **x, y** y k con nombres `xVec, yVec` y `count` respectivamente.

   (g) Ejecuta las siguientes líneas y verifica el formato de las respuestas

   ```
   > lapply( rep(10,4), listaFN )
   > sapply( rep(10,4), listaFN )
   ```