TP 4: Programmer avec 

Exercice I. Tirer jusqu'au 1

Écrire une fonction permettant de savoir combien de fois il faut tirer des variables aléatoires de Bernoulli de paramètre π (argument de la fonction) jusqu'à obtenir le premier 1

Exercice II. Mise en jambe avec la boucle for

Construire une fonction qui prend en argument un jeu de données (de mode « tout numérique ») et qui renvoie en sortie, une matrice tel que :

1. la première ligne de la matrice stocke les valeurs minima de chacune des colonnes de X
2. la deuxième ligne de la matrice stocke les valeurs maximales de chacune des colonnes de X
3. la troisième ligne de la matrice les valeurs moyennes de chacune des colonnes de X
4. la quatrième ligne de la matrice les valeurs médianes de chacune des colonnes de X
5. la cinquième ligne de la matrice, la variance de chacune des colonnes de X

Exercice III. « positif » ou « négatif » ???

Ecrire une fonction qui prend en argument un vecteur numérique et qui renvoie un vecteur de sortie de même longueur composé des valeurs « positif » ou « négatif » en fonction du signe du vecteur d'entrée.

Exercice IV. Standardisation

Construire une fonction qui prend en entrée une matrice numérique et qui renvoie la matrice centrée et réduite associée (i.e. chaque colonne de la matrice et centrée en 0 et de variance unitaire).

Exercice V. Matrice identité

Écrire une fonction permettant de construire une matrice identité de dimension n (où n est fixé par l'utilisateur).

Exercice VI. Stockage de coefficients résultant d'analyses

Considérons le jeu de données auto2004.txt. On souhaite évaluer l'impact des différentes variables sur la variable cylindrée. On décide donc de faire une régression de la variable cylindrée sur chacune des variables et stocker les coefficients de chacune des régressions dans un vecteur. Écrire une fonction prenant 2 arguments (la variable cible et le tableau des variables explicatives) qui retourne le vecteur de coefficients.

Exercice VII. Matrice de confusion

Écrire une fonction retournant une matrice de confusion. Cette fonction prend 2 arguments (Yobservé et Yprédit) et retourne la matrice de confusion.

Correction de l'exercice I

```
jusqua <- function(p) {
        number = 0
        b=0
        while(b != 1) {
                b <- rbinom(1, 1, p)
                number <- number + 1
        }
number
}

jusqua(0.1) #Exemple
```

Correction de l'exercice II.

#Avec la boucle for

```
resume = function(X){
resultat = matrix(0, 5, ncol(X))
   for(j in 1 : ncol(X)){
        resultat[1, j] = min(X[, j])
        resultat[2, j] = max(X[, j])
        resultat[3, j] = mean(X[, j])
        resultat[4, j] = median(X[, j])
        resultat[5, j] = var(X[, j])
   }
return(resultat)
}
```

Avec la fonction apply()

```
resume_bis = function(X){
resultat =  apply(X, 2, min)
resultat = rbind(resultat, apply(X, 2, max))
resultat = rbind(resultat, apply(X, 2, mean))
resultat = rbind(resultat, apply(X, 2, median))
resultat = rbind(resultat, apply(X, 2, var))
rownames(resultat) = c("min","max","mean","median","var")
colnames(resultat) = paste("V", 1:ncol(X), sep ="")
return(resultat)
}
```

Correction de l'exercice III.

```
pos_neg = function(x){
res = rep(0, length(x))
res[which(x>0)]=1
return(res)
}
```

Autre solution

```
pos_neg = function(x)
{
res = ifelse(x > 0, "positif", "negatif")
return(res)
}
```

Correction de l'exercice IV.

```
standardisation = function(X)
{
X = apply(X, 2, function(x) {(x-mean(x))/sd(x)})
return(X)
}
```

Correction de l'exercice V.

```
identite = function(dimension)
{
	X = matrix(0, dimension, dimension)
	for (i in 1:dimension)
	{
		for (j in 1: dimension)
		{
			if (i==j)
			X[i, j] = 1
		}
	}
return(X)
}

# une alternative plus vite :
diag(n)
```

Correction de l'exercice VI.

```
coef_reg = function(X, Y)
{
	b = numeric(dim(X)[2])
	for (i in 1 : dim(X)[2])
	{
		result.lm = lm(Y ~ X[ , i])
		b[i] = result.lm[[1]][2]
	}
return(b)
}

##Applications aux autos
A = read.table("Datasets/auto2004.txt", row.names=1, header = TRUE, sep = "\t")
variable = A[ , 2:6]
cible = A$Cylindree
b = coef_reg(variable, cible)
```

Correction de l'exercice VII.

```
confusion = function(Ypred, Y) {
	M = matrix(0, length(unique(Y)), length(unique(Y)))
	for (i in 1: length(unique(Y))) {
		for (j in 1:length(unique(Y))) {
			sortedY <- sort(unique(Y))
			a <- which(Ypred == sortedY[i] & Y == sortedY[j])
			M[i, j] = length(a)
		}
	}
return(M)
}

# Application
Yobserve = rbinom (10, 2, .5)
Ypredit = rbinom (10, 2, .5)
confusion(Ypredit, Yobserve)
```