

```

1
2 #####
3 # Commandes session 1 : Introduction
4 #####
5
6 install.packages("dplyr", DEPENDANCIES=T)
7 install.packages("ggplot2", DEPENDANCIES=T)
8 install.packages("plotly", DEPENDANCIES=T)
9 install.packages("caret", DEPENDANCIES=T)
10 install.packages("e1071", DEPENDANCIES=T)
11 install.packages("rpart", DEPENDANCIES=T)
12 install.packages("randomForest", DEPENDANCIES=T)
13
14 library("dplyr")
15 library("ggplot2")
16 library("plotly")
17 library("caret")
18 library("e1071")
19 library("rpart")
20 library("randomForest")
21
22 #####
23 # Commandes session 2 : débiter avec R
24 #####
25
26 class(3.5)
27 typeof(3.5)
28 typeof(3)
29 typeof(as.integer(3))
30
31 class("hello")
32 typeof("hello")
33
34 65 < 60
35 3>1
36
37 # NA
38
39 # ma variable mon caractere
40 mon_caractere="a"
41
42 mon_premier_nombre=10
43 mon_deuxieme_nombre=4
44
45 10+4
46
47 mon_premier_nombre+mon_deuxieme_nombre
48 mon_premier_nombre-mon_deuxieme_nombre
49 mon_premier_nombre/mon_deuxieme_nombre
50 mon_premier_nombre**mon_deuxieme_nombre
51 mon_premier_nombre*mon_deuxieme_nombre
52 mon_premier_nombre%%mon_deuxieme_nombre
53
54 c(1,2,3)
55 mon_vecteur=c(2,3,1)
56 mon_vecteur
57 class(mon_vecteur)
58
59 vecteur1=seq(from=1, to=10)
60 vecteur2=rep(10, times=10)
61
62 vecteur1 + 1
63 vecteur1 * 10
64 vecteur1 * vecteur2
65
66 vecteur2=10
67 vecteur1 / vecteur2
68
69 vecteur2=c(2,1,3)
70 vecteur1 / vecteur2
71
72 vecteur2=seq(from=6, to=15)
73 vecteur1 - vecteur2

```

```

74
75 concatenation=c(vecteur1, vecteur2)
76
77 vecteur2[2]
78 vecteur2[1:3]
79 vecteur2[c(1,6,2)]
80
81 superieur = vecteur2 > 8
82
83 mean(vecteur2)
84
85 length(vecteur2)
86 names(vecteur2)=paste("Ma valeur",vecteur2, sep=" ")
87 sort(vecteur2, decreasing = TRUE)
88 rank(vecteur2)
89
90 sum(vecteur2)
91 mean(vecteur2)
92 min(vecteur2)
93 max(vecteur2)
94 summary(vecteur2)
95
96 names(moyennes_de_la_classe)=c("Pedro", "Baptiste", "Amaury", "Flora", "Kevin",
97 "Markus", "Rozenn", "Raphael", "Jordan", "Victoire", "Thomas",
98 "Julia", "Marion", "Goulwen", "Suzon", "Lise", "Quentin", "Boniface", "Emil", "Gabin")
99 moyennes_de_la_classe=sample(1:20,20)
100
101 moyenne_generale=mean(moyennes_de_la_classe)
102 moins_bonne_note=min(moyennes_de_la_classe)
103 meilleure_note=max(moyennes_de_la_classe)
104 eleve_superieur_moyenne_generale=moyennes_de_la_classe[moyennes_de_la_classe>
moyenne_generale]
105
106 tableau_recapitulatif=c(moyenne_generale,moins_bonne_note,meilleure_note,length(
eleve_superieur_moyenne_generale))
107 names(tableau_recapitulatif)=c("Moyenne générale", "Moins bonne note", "Meilleure
note", "Nombre d'élèves avec une note > moyenne générale")
108
109 moyennes_de_la_classe[moyennes_de_la_classe==meilleure_note]
110
111 sort(moyennes_de_la_classe, decreasing = TRUE)
112
113 summary(moyennes_de_la_classe)
114
115 #####
116 # Commandes session 3 : les matrices
117 #####
118
119 notes=sample(1:20, 15)
120 notes[3]=15
121
122 matrix(notes, ncol=3, nrow=5)
123 matrix(notes, ncol=3, nrow=5, byrow=TRUE)
124
125 notes_2=c(sample(1:20, 10), "a", "b", "c", "d", "e")
126 matrix(c(sample(1:20, 10), "a", "b", "c", "d", "e"), ncol=3, nrow=5)
127
128 notes_des_eleves=matrix(notes, ncol=3, nrow=5)
129 colnames(notes_des_eleves)=c("SVT", "Mathématiques", "Français")
130 rownames(notes_des_eleves)=c("Jean", "Léa", "Thomas", "Julien", "Zoé")
131
132 notes_des_eleves[1,]
133 notes_des_eleves[,1]
134
135 notes_des_eleves[1,2]
136
137 notes_des_eleves[1,c(2,3)]
138 notes_des_eleves[1,2:3]
139
140 notes_des_eleves["Thomas","Français"]
141 notes_des_eleves[c("Léa","Thomas"),"Français"]
142

```

```

143 notes_des_eleves[c("Léa", "Thomas"), "Français"]=c(15, 10)
144
145 notes_2=c(rep(0.5, times=5), rep(1, times=5), rep(0.9, times=5))
146
147 rowSums(notes_des_eleves)
148 colSums(notes_des_eleves)
149
150 rowMeans(notes_des_eleves)
151 colMeans(notes_des_eleves)
152
153 install.packages("Stat2Data")
154 library("Stat2Data")
155 data("HorsePrices")
156 matrice_prix_cheval=as.matrix(HorsePrices[, -c(1, 5)])
157 rownames(matrice_prix_cheval)=HorsePrices[, 1]
158 colnames(matrice_prix_cheval)=c("Prix", "Age", "Taille")
159
160 # je sais que 1 hand fait 0.1016 mètre
161 matrice_prix_cheval[, 3]=matrice_prix_cheval[, 3]*0.1016
162 # 1 dollar c'est 0.86 euros
163 matrice_prix_cheval[, 1]=matrice_prix_cheval[, 1]*0.86
164
165 dim(matrice_prix_cheval)
166 summary(matrice_prix_cheval)
167
168 matrice_prix_cheval=na.omit(matrice_prix_cheval)
169 dim(matrice_prix_cheval)
170
171 matrice_prix_cheval[matrice_prix_cheval[, 1]==946, ]
172 matrice_prix_cheval[matrice_prix_cheval[, 1]==51600, ]
173 matrice_prix_cheval[matrice_prix_cheval[, 1]==946, ]=c(2500, 19, 1.651)
174 matrice_prix_cheval[matrice_prix_cheval[, 1]==2500, ]
175
176 matrice_prix_cheval[matrice_prix_cheval[, 1]>23082, ]
177 dim(matrice_prix_cheval[matrice_prix_cheval[, 1]>23082, ])
178 matrice_prix_cheval[matrice_prix_cheval[, 3]>1.6, ]
179 colMeans(matrice_prix_cheval[matrice_prix_cheval[, 3]>1.6, ])
180 colMeans(matrice_prix_cheval[matrice_prix_cheval[, 3]<1.6, ])
181
182 colSums(matrice_prix_cheval)
183
184 #####
185 # Commandes session 4 : les dataframes
186 #####
187
188 mon_dataframe=data.frame(c(18, 26, 54, 78), c(56, 84, 76, 62), c("M", "F", "M", "F"), c(TRUE
, TRUE, TRUE, FALSE))
189 colnames(mon_dataframe)=c("Age", "Poids", "Sexe", "Ma valeur booléenne")
190 rownames(mon_dataframe)=c("Jean", "Zoé", "Lucas", "Chloé")
191
192 data_iris=read.table("iris.csv", header = TRUE, sep=",")
193 data_iris=read.table("iris.csv", header = TRUE, sep=",", row.names=1)
194 data_iris=read.csv("iris.csv", row.names=1)
195 data("iris")
196 ls()
197 write.table(data_iris, file="iris_2.csv", sep=",", row.names=TRUE)
198 write.csv(data_iris, file="iris_2.csv", row.names=TRUE)
199 save(data_iris, file="iris.Rdata")
200 load("iris.Rdata")
201
202 data_iris[, 1]
203 data_iris[, c(1:3)]
204 head(data_iris[, c(1:3)])
205 head(data_iris[c(1, 50, 60), c(1:3)])
206 head(data_iris[c(1, 50, 60), c("Sepal.Length", "Sepal.Width", "Petal.Length")])
207 head(data_iris$Species)
208
209 data_iris[data_iris$Species == "setosa", ]
210 dim(data_iris[data_iris$Species == "setosa", ])
211 data_iris$Species == "setosa"
212 data_iris[which(data_iris$Species == "setosa"), ]
213 which(data_iris$Species == "setosa")
214

```

```

215 data_iris[which(data_iris$Species == "setosa" & data_iris$Petal.Length == 1.4),]
216 which(data_iris$Species == "setosa" & data_iris$Petal.Length == 1.4)
217
218 data_iris[data_iris$Species %in% c("setosa", "versicolor"),]
219 data_iris$Species %in% c("setosa", "versicolor")
220
221 subset(data_iris, Species="setosa" & Petal.Length == 1.4)
222
223 subset(data_iris, Species="setosa" & Petal.Length == 1.4, select=c("Petal.Length"))
224
225 class(iris$Species)
226 head(iris$Species)
227 colnames(iris)=c("Longueur des sépales", "Largeur des sépales", "Longueur des pétales",
228 "Largeur des pétales", "Espèce")
229 rownames(iris)=paste("iris_", rownames(iris), sep="")
230 summary(iris)
231
232 min(iris[iris$"Espèce" == "setosa",3])
233 mean(iris[iris$"Espèce" == "setosa",3])
234 mean(iris[iris$"Espèce" == "versicolor",3])
235 mean(iris[iris$"Espèce" == "virginica",3])
236
237 data_iris_quantitative=iris[,c(1,2,3,4)]
238 data_iris_qualitative=iris[,5]
239 data_iris_qualitative=as.data.frame(iris[,5])
240 colnames(data_iris_qualitative)=c("Espèce")
241 rownames(data_iris_qualitative)=paste("iris_", rownames(data_iris_qualitative), sep="")
242
243 iris_complet=merge(data_iris_quantitative,data_iris_qualitative, by="row.names")
244 rownames(iris_complet)=iris_complet[,1]
245 iris_complet=iris_complet[,-1]
246
247 head(cbind(data_iris_quantitative,data_iris_qualitative))
248 head(rbind(iris,iris))
249
250 dim(rbind(iris,iris))
251 dim(iris)
252
253 #####
254 # Commandes session 5 : les bases de la programmation en R
255 #####
256
257 # opérateur 1 : strictement supérieur
258 5 > 3
259 # < strictement inférieur
260 4 < 6
261 2 < 1
262 # >= <=
263 4 <= 4
264 4 >= 4
265 # ==
266 3 == 3
267 # !=
268 3 != 3
269
270 # &, &&
271 2 == 2 & 1 == 1
272 2 == 2 && 1 == 1
273
274 c(2,2) == c(2,2) & c(3,2) == c(2,3)
275 c(2,2) == c(2,2) && c(3,2) == c(2,3)
276
277 c(2,2) == c(2,2) && c(3,2) == c(3,3)
278 c(2,2) == c(2,2) & c(3,2) == c(3,3)
279
280 # ou, |, ||
281 2 == 2 | 1 == 2
282 2 == 2 & 1 == 2
283
284 # instructions de conditions
285 #if(condition){
286 # action1

```

```

286 #} else {
287 #   action2
288 #}
289
290 if( 5 > 3 & 2 < 3 ){
291   print("OK")
292 } else {
293   print("PAS OK")
294 }
295
296 if( 5 == 3 & 2 < 3 ){
297   print("OK")
298 } else {
299   print("PAS OK")
300 }
301
302 # instruction de boucles
303 #for (valeur in vecteur){
304 #   action1
305 #}
306
307 for(valeur in c(1,2,3,4,5)){
308   print(valeur + 1)
309 }
310
311 for(element in c(1,2,3,4,5)){
312   print(paste("Mon chiffre :", element))
313 }
314
315 for(element in c(1,2,3,4,5)){
316   if(element > 1 & element < 5){
317     print(element)
318   }
319 }
320
321
322 # instructions de boucle while : tant que
323
324 #while(condition){
325 #   actions
326 #}
327
328 valeur = 200
329
330 while( valeur/5 > 1){
331   valeur = valeur/5
332   print(valeur)
333 }
334
335 data(iris)
336
337 compteur_individu_sepal_superieur_5=0
338 for( length in iris$Sepal.Length ){
339   if( length >= 5 ){
340     compteur_individu_sepal_superieur_5=compteur_individu_sepal_superieur_5+1
341   }
342 }
343
344 compteur_setosa=0
345 compteur_versicolor=0
346 compteur_virginica=0
347
348 for(species in iris$Species){
349   if(species == "setosa"){
350     compteur_setosa=compteur_setosa+1
351   } else if( species == "versicolor"){
352     compteur_versicolor=compteur_versicolor+1
353   } else {
354     compteur_virginica=compteur_virginica+1
355   }
356 }
357
358 print(paste("Nombre de setosa :", compteur_setosa))

```

```

359 print(paste("Nombre de versicolor :", compteur_versicolor))
360 print(paste("Nombre de virginica :", compteur_virginica))
361
362
363 dim(iris)[1]
364
365 nombre_setosa_sepal_sup_5=0
366 for(ligne in 1:dim(iris)[1]){
367     individu=iris[ligne,]
368     if( individu$Species == "setosa" & individu$Sepal.Length >=5 ){
369         nombre_setosa_sepal_sup_5=nombre_setosa_sepal_sup_5+1
370     }
371 }
372 print(nombre_setosa_sepal_sup_5)
373
374
375 for(colonne in 1:dim(iris)[2]){
376     print(iris[,colonne])
377     print("-----")
378 }
379
380
381 #nom_de_notre_fonction<-function(argument1,argument2,...){
382 #   instructions
383 #   bloc code
384 #   return(resultat)
385 #}
386
387
388 nombre_individus_superieur_5<-function(dataframe){
389     compteur_individu_sepal_superieur_5=0
390     for( length in dataframe$Sepal.Length ){
391         if( length >= 5 ){
392             compteur_individu_sepal_superieur_5=compteur_individu_sepal_superieur_5+1
393         }
394     }
395     return(compteur_individu_sepal_superieur_5)
396 }
397
398 nombre_individus_superieur_5(iris)
399
400
401 ma_fonction_qui_calcule_la_moyenne<-function(iris){
402     iris_species_setosa=subset(iris, iris$Species=="setosa")
403     mean_setosa=colMeans(iris_species_setosa[,-5])
404     mean_versicolor=colMeans(subset(iris, iris$Species=="versicolor")[,-5])
405     mean_virginica=colMeans(subset(iris, iris$Species=="virginica")[,-5])
406     resultat=data.frame(Setosa=mean_setosa, Versicolor=mean_versicolor, Virginica=
407     mean_virginica)
408     return(resultat)
409 }
410
411 ma_fonction_qui_calcule_la_moyenne(iris)
412
413 #####
414 # Commandes session 5 : manipulation avancée des données
415 #####
416
417 # apply
418 # apply(X, MARGIN, FUN)
419 # X : dataframe, une matrice
420 # MARGIN : 1 : pour les lignes, 2 : pour les colonnes et colonnes + lignes : c(1,2)
421 # FUN : fonction (mean, sum, summary, ...)
422
423 data(iris)
424 apply(iris[, -5], 2, mean)
425 apply(iris[, -5], 1, mean)
426
427 apply(iris[, -5], 2, summary)
428
429 nombre_valeurs_superieures_5<-function(vecteur){
430     length(vecteur[vecteur>5])
431 }

```

```

431
432 apply(iris[,-5], 2, nombre_valeurs_superieures_5)
433 apply(iris[,-5], 1, nombre_valeurs_superieures_5)
434
435 # by
436 # by(X, INDICES, FUN)
437 # X : dataframe
438 # INDICES : iris$Species
439 # FUN : fonction
440
441 by(iris, iris$Species, summary)
442 by(iris[,-5], iris$Species, cor)
443 by(iris[,-5], iris$Species, mean)
444
445 # aggregate
446 # aggregate(X, BY, FUN)
447 # X : dataframe
448 # BY : iris$Species
449 # FUN : fonction
450
451 aggregate(iris[,-5], as.data.frame(iris$Species), mean)
452
453 library(dplyr)
454 class(iris)
455
456 # tibble
457
458 iris_data=as_tibble(iris)
459 iris_data
460
461 # select : selectionner des colonnes
462 select(iris, Sepal.Length, Petal.Length, Species)
463 select(iris, Sepal.Length:Petal.Length)
464 select(iris, -Species)
465 select(iris, starts_with("Petal"))
466 select(iris, -starts_with("Sepal"))
467 select(iris, ends_with("Length"))
468 select(iris, contains("al"))
469
470 # filter : filtrer sur les individus
471 filter(iris, Sepal.Length >=5, Sepal.Width >=2)
472 filter(iris, between(Sepal.Length, 4, 7))
473 filter(iris, Sepal.Length >=4, Sepal.Length <=7)
474 filter(iris, Species == "setosa")
475 filter(iris, Species != "setosa")
476 filter(iris, Species %in% c("setosa", "versicolor"))
477 filter(iris, (Species == "setosa" | Species == "versicolor"))
478 filter_all(iris[,-5], any_vars(. > 5))
479 filter(iris, (Sepal.Length >5 | Sepal.Width >5 | Petal.Length >5 | Petal.Width >5))
480 filter_all(iris[,-5], all_vars(. > 2))
481
482 # %>% :
483 select(iris, Sepal.Length, Petal.Length, Species)
484
485 iris %>%
486 select(Sepal.Length, Petal.Length, Species)
487
488 iris %>%
489 select(-Species) %>%
490 filter_all(all_vars(. > 2))
491
492 # arrange
493 iris %>%
494 arrange(Sepal.Length)
495
496 iris %>%
497 arrange(desc(Sepal.Length))
498
499 iris %>%
500 arrange(Sepal.Length, Sepal.Width)
501
502 iris %>%
503 select(Petal.Length, Petal.Width, Species) %>%

```

```

504 filter(Species == "setosa") %>%
505 arrange(Petal.Length, Petal.Width)
506
507 # summarise : résumé statistique d'un vecteur qui retourne une valeur.
508
509 iris %>%
510 summarise(moyenne_taille_petal=mean(Petal.Length))
511
512 iris %>%
513 summarise(moyenne_taille_petale=mean(Petal.Length),
514 minimum_taille_petale=min(Petal.Length),
515 maximum_taille_petale=max(Petal.Length),
516 total=n())
517
518 iris %>%
519 summarise(moyenne_taille_petale=mean(Petal.Length),
520 moyenne_taille_sepale=mean(Sepal.Length),
521 minimum_taille_petale=min(Petal.Length),
522 minimum_taille_sepale=min(Sepal.Length)
523 )
524
525 iris %>%
526 summarise_each(funs(mean, min), Petal.Length, Sepal.Length)
527
528
529 # group_by()
530 iris %>%
531 group_by(Species) %>%
532 summarise(moyenne_taille_petale=mean(Petal.Length),
533 minimum_taille_petale=min(Petal.Length),
534 maximum_taille_petale=max(Petal.Length),
535 total=n())
536
537 iris %>%
538 group_by(Species) %>%
539 filter(Petal.Length > 5) %>%
540 summarise(n())
541
542 # mutate : ajouter, supprimer, modifier
543
544 # ajouter une ou plusieurs variables
545 iris %>%
546 mutate(somme_longueur_largeur_petale=Petal.Length+Petal.Width,
547 somme_longueur_largeur_sepale=Sepal.Length+Sepal.Width
548 )
549
550 # supprimer une ou des variables
551 iris %>%
552 mutate(Species=NULL, Sepal.Width=NULL)
553
554 # modifier une variable
555 iris %>%
556 mutate(Sepal.Length=Sepal.Length*2)
557
558 #####
559 # cas pratique fast food
560 #####
561
562 # lien dropbox
563 # https://www.dropbox.com/s/pr288bu5slmk2jk/FastFoodRestaurants.csv?dl=0
564
565 # lecture du fichier fast food aux US
566 fast_food=read.csv("FastFoodRestaurants.csv")
567
568 # transformation en tibble
569 fast_food_tibble=as_tibble(fast_food)
570
571 # Quelles sont les 5 villes avec le plus de fast-food ?
572
573 fast_food_tibble %>%
574 group_by(city) %>%
575 summarise(Nombre_de_restaurants=length(city)) %>%
576 arrange(desc(Nombre_de_restaurants)) %>%

```



```

577 head(n=5)
578
579 # Quels sont les fast food les plus présents dans ces 5 villes ?
580
581 city_list=fast_food_tibble %>%
582 group_by(city) %>%
583 summarise(Nombre_de_restaurants=length(city)) %>%
584 arrange(desc(Nombre_de_restaurants)) %>%
585 head(n=5) %>%
586 pull(city)
587
588 fast_food_tibble %>%
589 filter(city %in% city_list)
590
591 fast_food_tibble %>%
592 filter(city %in% city_list) %>%
593 group_by(name) %>%
594 summarise(nombre_de_fast_food=length(name)) %>%
595 arrange(desc(nombre_de_fast_food))
596
597 # Quels sont les fast food avec le plus de restaurants aux US ?
598 fast_food_tibble %>%
599 group_by(name) %>%
600 summarise(nombre_de_fast_food=length(name), pourcentage_de_fast_food=(length(name)*100
/10000)) %>%
601 arrange(desc(nombre_de_fast_food))
602
603 # Dans quelle ville y a-t-il le plus de McDonald's ?
604 fast_food_tibble %>%
605 filter(name %in% "McDonald's") %>%
606 group_by(city) %>%
607 summarise(nombre_de_fast_food=length(city)) %>%
608 arrange(desc(nombre_de_fast_food))
609
610
611 # Où se situe New-York par rapport aux 5 villes avec le plus de fast-foods ?
612
613 fast_food_tibble %>%
614 group_by(city) %>%
615 filter( city %in% "New York") %>%
616 summarise(nombre_de_fast_food=length(city))
617
618
619 # Fast food les plus présents à New York
620 fast_food_tibble %>%
621 filter( city %in% "New York") %>%
622 group_by(name) %>%
623 summarise(nombre_de_fast_food=length(name), pourcentage_de_fast_food=(length(name)*100
/10000)) %>%
624 arrange(desc(nombre_de_fast_food))
625
626 # Forbes : Orlando, Cincinnati And The Fast Food Capitals of the US.
627
628 #####
629 # Commandes session 7 : visualisation avancée des données
630 #####
631
632 data(iris)
633
634 # plot()
635
636 plot(iris$Sepal.Length, iris$Petal.Width)
637
638 plot(iris$Sepal.Length, iris$Sepal.Width, xlab="Longueur", ylab="Largeur", col="red",
xlim=c(min(iris$Sepal.Length,iris$Petal.Length),
639 max(iris$Sepal.Length,iris$Petal.Length)), ylim=c(min(iris$Sepal.Width,iris$
Petal.Width),max(iris$Sepal.Width,iris$Petal.Width)))
640 lines(iris$Petal.Length, iris$Petal.Width, col="slateblue4", type="p", pch=22)
641 title(main="Longueur en fonction de largeur", col="blue")
642 legend(1, 4.2, c("Sépales", "Pétales"), col=c("red", "slateblue4"), pch=21:22)
643
644 library("ggplot2")
645 # Plot = data + aesthetics + Geometry

```

```

646
647 g<-ggplot(iris, aes(x=Petal.Length, y=Petal.Width))
648 g<-g+geom_point()
649
650 # couleur selon l'espèce
651 g<-ggplot(iris, aes(x=Petal.Length, y=Petal.Width, color=Species))+geom_point()
652
653 # couleurs et formes des points différentes selon l'espèce
654 g<-ggplot(iris, aes(x=Petal.Length, y=Petal.Width, color=Species, shape=Species))+
geom_point()
655
656 # modifier taille des points
657 g<-ggplot(iris, aes(x=Petal.Length, y=Petal.Width, color=Species, shape=Species))+
geom_point(size=3)
658
659 # créer un gradient de couleur
660 g<-ggplot(iris, aes(x=Petal.Length, y=Petal.Width, color=Petal.Length, shape=Species
))+geom_point(size=3)
661 g<-g+scale_color_gradient(low="blue", high="red")
662 g
663
664 # modifier le fond de notre graphique
665 g<-g+theme_minimal()
666
667 # modifier la position de la légende
668 g<-g+theme(legend.position="top")
669 g
670
671
672 g<-ggplot(iris, aes(x=Petal.Length, y=Petal.Width, color=Petal.Length, shape=Species
))+geom_point(size=3)
673 g<-g+scale_color_gradient(low="blue", high="red")+theme_minimal()+theme(legend.title=
element_blank())
674 g
675
676
677 # modifier couleur, texture du titre de la légende
678 g<-ggplot(iris, aes(x=Petal.Length, y=Petal.Width, color=Petal.Length, shape=Species
))+geom_point(size=3)
679 g<-g+scale_color_gradient(low="blue", high="red")+theme_minimal()+theme(legend.title=
element_text(colour="red", size=15, face="bold"))
680 g
681
682 # modifier couleur, texture, taille des labels de ma légende
683 g<-ggplot(iris, aes(x=Petal.Length, y=Petal.Width, color=Petal.Length, shape=Species
))+geom_point(size=3)
684 g<-g+scale_color_gradient(low="blue", high="red")+theme_minimal()+theme(legend.title=
element_text(colour="red", size=15, face="bold"))
685 g<-g+theme(legend.text=(element_text(colour="blue", size=8, face="bold.italic"))))
686 g
687
688 # steelblue
689 # ajouter un cadre à NBSP notre légende
690 g<-ggplot(iris, aes(x=Petal.Length, y=Petal.Width, color=Petal.Length, shape=Species
))+geom_point(size=3)
691 g<-g+scale_color_gradient(low="blue", high="red")+theme_minimal()+theme(legend.title=
element_text(colour="red", size=15, face="bold"))
692 g<-g+theme(legend.text=(element_text(colour="blue", size=8, face="bold.italic"))))
693 g
694
695 # modification des noms des axes et ajout d'un titre au graphique
696 g<-ggplot(iris, aes(x=Petal.Length, y=Petal.Width, color=Petal.Length, shape=Species
))+geom_point(size=3)
697 g<-g+scale_color_gradient(low="blue", high="red")+theme_minimal()+theme(legend.title=
element_text(colour="steelblue", size=9, face="bold"))
698 g<-g+theme(legend.text=(element_text(colour="blue", size=8, face="bold.italic"))))
699 g<-g+xlabs("Longueur des pétales")+ylabs("Largeur des pétales")+ggtitle("Longueur des
pétales en fonction de largeur des pétales")
700 g<-g+theme(plot.title=element_text(colour="steelblue", size=15, face="bold"))
701 g<-g+theme(axis.title=element_text(colour="steelblue", size=10, face="bold"))
702 g<-g+theme(axis.text=element_text(colour="steelblue", size=10, face="bold", angle=45))
703 #g<-g+theme(axis.line=element_line(colour="steelblue", size=2, linetype="dotted"))
704 g

```

```

705
706 # modification des noms des axes et ajout d'un titre au graphique
707 g<-ggplot(iris, aes(x=Petal.Length, y=Petal.Width, color=Petal.Length, shape=Species
)) + geom_point(size=3)
708 g<-g+scale_color_gradient(low="blue", high="red")+theme_minimal()+theme(legend.title=
element_text(colour="steelblue", size=9, face="bold"))
709 g<-g+theme(legend.text=(element_text(colour="blue", size=8, face="bold.italic"))))
710 g<-g+xlab("Longueur des pétales")+ylab("Largeur des pétales")+ggtitle("Longueur des
pétales en fonction de largeur des pétales")
711 g<-g+theme(plot.title=element_text(colour="steelblue", size=15, face="bold"))
712 g<-g+theme(axis.title=element_text(colour="steelblue", size=10, face="bold"))
713 g<-g+theme(axis.text=element_text(colour="steelblue", size=10, face="bold", angle=45))
714 #g<-g+theme(axis.line=element_line(colour="steelblue", size=2, linetype="dotted"))
715 g
716
717 # facet_wrap -> combiner des graphiques
718 g<-ggplot(iris, aes(x=Petal.Length, y=Petal.Width, color=Petal.Length, shape=Species
)) + geom_point(size=3)
719 g<-g+scale_color_gradient(low="blue", high="red")+theme_minimal()+theme(legend.title=
element_text(colour="steelblue", size=9, face="bold"))
720 g<-g+theme(legend.text=(element_text(colour="blue", size=8, face="bold.italic"))))
721 g<-g+xlab("Longueur des pétales")+ylab("Largeur des pétales")+ggtitle("Longueur des
pétales en fonction de largeur des pétales")
722 g<-g+theme(plot.title=element_text(colour="steelblue", size=15, face="bold"))
723 g<-g+theme(axis.title=element_text(colour="steelblue", size=10, face="bold"))
724 g<-g+theme(axis.text=element_text(colour="steelblue", size=10, face="bold", angle=45))
725 #g<-g+theme(axis.line=element_line(colour="steelblue", size=2, linetype="dotted"))
726 g<-g+facet_wrap(~Species)+theme(strip.text=element_text(colour="steelblue", size=10,
face="bold"))
727 g<-g+theme(strip.background=element_rect(colour="steelblue", size=1, linetype="solid"))
728 g
729
730
731 # modification des noms des axes et ajout d'un titre au graphique
732 g<-ggplot(iris, aes(x=Petal.Length, y=Petal.Width, color=Petal.Length, shape=Species
)) + geom_point(size=3)
733 g<-g+scale_color_gradient(low="blue", high="red")+theme_minimal()+theme(legend.title=
element_text(colour="steelblue", size=9, face="bold"))
734 g<-g+theme(legend.text=(element_text(colour="blue", size=8, face="bold.italic"))))
735 g<-g+xlab("Longueur des pétales")+ylab("Largeur des pétales")+ggtitle("Longueur des
pétales en fonction de largeur des pétales")
736 g<-g+theme(plot.title=element_text(colour="steelblue", size=15, face="bold"))
737 g<-g+theme(axis.title=element_text(colour="steelblue", size=10, face="bold"))
738 g<-g+theme(axis.text=element_text(colour="steelblue", size=10, face="bold", angle=45))
739 #g<-g+theme(axis.line=element_line(colour="steelblue", size=2, linetype="dotted"))
740 g<-g+annotate("text", x=c(2,4,6), y=0.7, label=c("Setosa", "Versicolor", "Virginica"),
colour="steelblue", size=3, fontface="bold")
741 g<-g+annotate("rect", xmin=0.5, xmax=2.1, ymin=0, ymax=0.65, alpha=0.2, colour=
"steelblue", size=2)
742 g<-g+annotate("segment", x=0.5, xend=4, y=1.5, yend=0, colour="steelblue", size=2,
alpha=0.5)
743 g
744
745 # histogramme sur les données iris
746 png("histogramme_iris.png")
747 g<-ggplot(iris, aes(x=Petal.Length, fill=Species))+geom_histogram(color="white",
binwidth =0.5)
748 g
749 dev.off()
750
751 pdf("boxplot_iris.pdf")
752 g<-ggplot(iris, aes(x=Species, y=Petal.Length, fill=Species))+geom_boxplot()
753 g<-g+ggtitle("Boxplot de la longueur des pétales selon l'espèce")+xlab("Espèce")+ ylab
("Longueur des pétales")
754 g
755 dev.off()
756
757 #####
758 # exercice ggplot2
759 #####
760
761 library("ggplot2")
762 library("dplyr")

```

```

763
764 fast_food=read.csv("FastFoodRestaurants.csv")
765 fast_food_tibble=as_tibble(fast_food)
766
767 city_list=fast_food_tibble %>%
768 group_by(city) %>%
769 summarise(Nombre_de_restaurants=length(city)) %>%
770 arrange(desc(Nombre_de_restaurants)) %>%
771 head(n=10) %>%
772 pull(city)
773
774 fast_food_tibble_10_villes=fast_food_tibble %>%
775 filter(city %in% city_list)
776
777 list_fast_food=fast_food_tibble_10_villes %>%
778 group_by(name) %>%
779 summarise(Nombre_de_restaurants=length(name)) %>%
780 arrange(desc(Nombre_de_restaurants)) %>%
781 head(n=10) %>%
782 pull(name)
783
784 fast_food_tibble_10_villes_10_restaurants=fast_food_tibble_10_villes %>%
785 filter(name %in% list_fast_food)
786
787 pdf("fast_food.pdf")
788 g<-ggplot(fast_food_tibble_10_villes_10_restaurants, aes(city, fill=name))+geom_bar()+
789 theme_minimal()
790 g<- g + xlab("Les 10 capitales du fast-foods") + ylab("Les 10 restaurants les plus
791 implantés")
792 g<- g + ggtitle("Représentation des fast-foods les plus implantés \n dans les 10
793 capitales du fast-food")
794 g<- g + theme(plot.title=element_text(hjust=0.5), axis.text=element_text(face="bold",
795 size=7, angle=45))
796 g <- g + ylim(0, 100) + theme(legend.title=element_blank()) + scale_fill_brewer(
797 palette="Paired")
798 g
799 dev.off()
800
801 install.packages("plotly")
802 library("plotly")
803
804 ggplotly(g)
805
806 g<-ggplot(iris, aes(x=Species, y=Petal.Length, fill=Species))+geom_boxplot()
807 g<-g+ggtitle("Boxplot de la longueur des pétales selon l'espèce")+xlab("Espèce")+ ylab
808 ("Longueur des pétales")
809 g
810 ggplotly(g)
811
812 g<-ggplot(iris, aes(x=Petal.Length, y=Petal.Width, color=Petal.Length, shape=Species
813 ))+geom_point(size=3)
814 g<-g+scale_color_gradient(low="blue", high="red")+theme_minimal()+theme(legend.title=
815 element_text(colour="steelblue", size=9, face="bold"))
816 g<-g+theme(legend.text=(element_text(colour="blue", size=8, face="bold.italic")))
817 g<-g+xlab("Longueur des pétales")+ylab("Largeur des pétales")+ggtitle("Longueur des
818 pétales en fonction de largeur des pétales")
819 g<-g+theme(plot.title=element_text(colour="steelblue", size=15, face="bold"))
820 g<-g+theme(axis.title=element_text(colour="steelblue", size=10, face="bold"))
821 g<-g+theme(axis.text=element_text(colour="steelblue", size=10, face="bold", angle=45))
822 #g<-g+theme(axis.line=element_line(colour="steelblue", size=2, linetype="dotted"))
823 g<-g+annotate("text", x=c(2,4,6), y=0.7, label=c("Setosa","Versicolor","Virginica"),
824 colour="steelblue", size=3, fontface="bold")
825 g<-g+annotate("rect", xmin=0.5, xmax=2.1, ymin=0, ymax=0.65, alpha=0.2, colour=
826 "steelblue", size=2)
827 g<-g+annotate("segment", x=0.5, xend=4, y=1.5, yend=0, colour="steelblue", size=2,
828 alpha=0.5)
829 g
830 ggplotly(g)
831
832 #####

```

```

824 # Commandes session 8 : cas pratique de data science
825 #####
826
827 # lecture du fichier de données
828 bank_data=read.csv("bank.csv", sep=";")
829
830 summary(bank_data)
831
832 # visualisation des données
833 library("ggplot2")
834
835 g<-ggplot(bank_data, aes(x=y, y=duration, fill=y))+geom_boxplot()
836 g
837
838 library("plotly")
839 ggplotly(g)
840
841
842 g<-ggplot(bank_data, aes(x=y, y=age, fill=y))+geom_boxplot()
843 g
844 ggplotly(g)
845
846 g<-ggplot(bank_data, aes(y, fill=contact))+geom_bar()
847 g
848
849 # Création jeu de test et jeu d'entraînement
850 library("caret")
851
852 dummy_variables=dummyVars(~., data=bank_data)
853 dummy_variables_data=predict(dummy_variables, newdata=bank_data)
854 dummy_variables_data=as.data.frame(dummy_variables_data)
855
856 dummy_variables_data$"Souscription"=ifelse(dummy_variables_data$"y.no" == 1, "No",
"yes")
857 dummy_variables_data$"y.no"=NULL
858 dummy_variables_data$"y.yes"=NULL
859
860 # Création des jeux de données d'entraînement et de test
861 set.seed(3033)
862 training_size=floor(0.7*nrow(dummy_variables_data))
863 indices=sample(seq_len(nrow(dummy_variables_data)), size=training_size)
864 data_bank.train=dummy_variables_data[indices,]
865 data_bank.test=dummy_variables_data[-indices,]
866
867 dim(data_bank.train)
868 dim(data_bank.test)
869
870 # Normalisation des données
871
872 data_preprocess_value=preProcess(data_bank.train, method=c("center","scale"))
873 data_bank.train.scaled=predict(data_preprocess_value,data_bank.train)
874 data_bank.test.scaled=predict(data_preprocess_value,data_bank.test)
875
876 # Caret - downsample et upsample
877 table(data_bank.train.scaled[, "Souscription"])
878
879 set.seed(3033)
880 '%ni%' = Negate('%in%')
881
882 # downsample
883 data_bank.train.scaled.downsample=downSample(x=data_bank.train.scaled[, colnames(
data_bank.train.scaled) %ni% "Souscription"], y=as.factor(data_bank.train.scaled$
"Souscription"))
884 names(data_bank.train.scaled.downsample)[names(data_bank.train.scaled.downsample) ==
"Class"]="Souscription"
885 table(data_bank.train.scaled.downsample[, "Souscription"])
886
887 # upsample
888 data_bank.train.scaled.upsample=upSample(x=data_bank.train.scaled[, colnames(
data_bank.train.scaled) %ni% "Souscription"], y=as.factor(data_bank.train.scaled$
"Souscription"))
889 names(data_bank.train.scaled.upsample)[names(data_bank.train.scaled.upsample) ==
"Class"]="Souscription"

```

```

890 table(data_bank.train.scaled.upsample[, "Souscription"])
891
892 # modélisation avec naive bayes
893 set.seed(3033)
894 trainControl_data=trainControl(method="repeatedcv", number=10, repeats=3)
895 naive_bayes_desequilibree=train(Souscription ~., data=data_bank.train.scaled, method=
"nb", preProcess=NULL)
896
897 print(naive_bayes_desequilibree)
898
899 # prédiction avec notre modèle sur le jeu de données tests
900 prediction_naive_bayes_desequilibree=predict(naive_bayes_desequilibree, newdata=
data_bank.test.scaled[, -ncol(data_bank.test.scaled)])
901
902 # création de la matrice de confusion
903 confusionMatrix(prediction_naive_bayes_desequilibree, as.factor(data_bank.test.scaled[
, ncol(data_bank.test.scaled)]))
904
905 # modélisation avec naive bayes sur les données downsample
906 set.seed(3033)
907 trainControl_data=trainControl(method="repeatedcv", number=10, repeats=3)
908 naive_bayes_downsample=train(Souscription ~., data=data_bank.train.scaled.downsample,
method="nb", preProcess=NULL)
909
910 print(naive_bayes_downsample)
911
912 # prédiction avec notre modèle sur le jeu de données tests
913 prediction_naive_bayes_downsample=predict(naive_bayes_downsample, newdata=
data_bank.test.scaled[, -ncol(data_bank.test.scaled)])
914
915 # création de la matrice de confusion
916 confusionMatrix(prediction_naive_bayes_downsample, as.factor(data_bank.test.scaled[,
ncol(data_bank.test.scaled)]))
917
918 # modélisation avec SVM
919 set.seed(3033)
920 trainControl_data=trainControl(method="repeatedcv", number=10, repeats=3)
921 SVM_desequilibree=train(Souscription ~., data=data_bank.train.scaled, method=
"svmLinear", preProcess=NULL)
922
923 print(SVM_desequilibree)
924
925 # prédiction avec notre modèle sur le jeu de données tests
926 prediction_SVM_desequilibree=predict(SVM_desequilibree, newdata=data_bank.test.scaled[
, -ncol(data_bank.test.scaled)])
927
928 # création de la matrice de confusion
929 confusionMatrix(prediction_SVM_desequilibree, as.factor(data_bank.test.scaled[, ncol(
data_bank.test.scaled)]))
930
931 # varImp
932 varImp(naive_bayes_downsample, scale=F)
933
934

```