```r

#######################################
# Commandes session 1 : Introduction
#######################################

install.packages("dplyr", DEPENDANCIES=T)
install.packages("ggplot2", DEPENDANCIES=T)
install.packages("plotly", DEPENDANCIES=T)
install.packages("caret", DEPENDANCIES=T)
install.packages("e1071", DEPENDANCIES=T)
install.packages("rpart", DEPENDANCIES=T)
install.packages("randomForest", DEPENDANCIES=T)

library("dplyr")
library("ggplot2")
library("plotly")
library("caret")
library("e1071")
library("rpart")
library("randomForest")

#######################################
# Commandes session 2 : débuter avec R
#######################################

class(3.5)
typeof(3.5)
typeof(3)
typeof(as.integer(3))

class("hello")
typeof("hello")

65 < 60
3>1

# NA

# ma variable mon caractere
mon_caractere="a"

mon_premier_nombre=10
mon_deuxieme_nombre=4

10+4

mon_premier_nombre+mon_deuxieme_nombre
mon_premier_nombre-mon_deuxieme_nombre
mon_premier_nombre/mon_deuxieme_nombre
mon_premier_nombre**mon_deuxieme_nombre
mon_premier_nombre*mon_deuxieme_nombre
mon_premier_nombre%%mon_deuxieme_nombre

c(1,2,3)
mon_vecteur=c(2,3,1)
mon_vecteur
class(mon_vecteur)

vecteur1=seq(from=1, to=10)
vecteur2=rep(10, times=10)

vecteur1 + 1
vecteur1 * 10
vecteur1 * vecteur2

vecteur2=10
vecteur1 / vecteur2

vecteur2=c(2,1,3)
vecteur1 / vecteur2

vecteur2=seq(from=6, to=15)
vecteur1 - vecteur2
```

```r

concatenation=c(vecteur1, vecteur2)

vecteur2[2]
vecteur2[1:3]
vecteur2[c(1,6,2)]

superieur = vecteur2 > 8

mean(vecteur2)

length(vecteur2)
names(vecteur2)=paste("Ma valeur",vecteur2, sep=" ")
sort(vecteur2, decreasing = TRUE)
rank(vecteur2)

sum(vecteur2)
mean(vecteur2)
min(vecteur2)
max(vecteur2)
summary(vecteur2)

names(moyennes_de_la_classe)=c("Pedro", "Baptiste", "Amaury", "Flora", "Kevin",
"Markus", "Rozenn", "Raphael", "Jordan", "Victoire", "Thomas",
"Julia", "Marion", "Goulwen", "Suzon", "Lise", "Quentin", "Boniface", "Emil", "Gabin")

moyennes_de_la_classe=sample(1:20,20)

moyenne_generale=mean(moyennes_de_la_classe)
moins_bonne_note=min(moyennes_de_la_classe)
meilleure_note=max(moyennes_de_la_classe)
eleve_superieur_moyenne_generale=moyennes_de_la_classe[moyennes_de_la_classe>
moyenne_generale]

tableau_recapitulatif=c(moyenne_generale,moins_bonne_note,meilleure_note,length(
eleve_superieur_moyenne_generale))
names(tableau_recapitulatif)=c("Moyenne générale", "Moins bonne note", "Meilleure
note", "Nombre d'élèves avec une note > moyenne générale")

moyennes_de_la_classe[moyennes_de_la_classe==meilleure_note]

sort(moyennes_de_la_classe, decreasing = TRUE)

summary(moyennes_de_la_classe)

####################################
# Commandes session 3 : les matrices
####################################

notes=sample(1:20, 15)
notes[3]=15

matrix(notes, ncol=3, nrow=5)
matrix(notes, ncol=3, nrow=5, byrow=TRUE)

notes_2=c(sample(1:20, 10), "a", "b", "c", "d", "e")
matrix(c(sample(1:20, 10), "a", "b", "c", "d", "e"), ncol=3, nrow=5)

notes_des_eleves=matrix(notes, ncol=3, nrow=5)
colnames(notes_des_eleves)=c("SVT", "Mathématiques", "Fraçais")
rownames(notes_des_eleves)=c("Jean", "Léa", "Thomas", "Julien", "Zoé")

notes_des_eleves[1,]
notes_des_eleves[,1]

notes_des_eleves[1,2]

notes_des_eleves[1,c(2,3)]
notes_des_eleves[1,2:3]

notes_des_eleves["Thomas","Français"]
notes_des_eleves[c("Léa","Thomas"),"Français"]

```

```r
notes_des_eleves[c("Léa","Thomas"),"Français"]=c(15,10)

notes_2=c(rep(0.5, times=5),rep(1, times=5), rep(0.9, times=5))

rowSums(notes_des_eleves)
colSums(notes_des_eleves)

rowMeans(notes_des_eleves)
colMeans(notes_des_eleves)

install.packages("Stat2Data")
library("Stat2Data")
data("HorsePrices")
matrice_prix_cheval=as.matrix(HorsePrices[,-c(1,5)])
rownames(matrice_prix_cheval)=HorsePrices[,1]
colnames(matrice_prix_cheval)=c("Prix", "Age", "Taille")

# je sais que 1 hand fait 0.1016 mètre
matrice_prix_cheval[,3]=matrice_prix_cheval[,3]*0.1016
# 1 dollar c'est 0.86 euros
matrice_prix_cheval[,1]=matrice_prix_cheval[,1]*0.86

dim(matrice_prix_cheval)
summary(matrice_prix_cheval)

matrice_prix_cheval=na.omit(matrice_prix_cheval)
dim(matrice_prix_cheval)

matrice_prix_cheval[matrice_prix_cheval[,1]==946,]
matrice_prix_cheval[matrice_prix_cheval[,1]==51600,]
matrice_prix_cheval[matrice_prix_cheval[,1]==946,]=c(2500,19,1.651)
matrice_prix_cheval[matrice_prix_cheval[,1]==2500,]

matrice_prix_cheval[matrice_prix_cheval[,1]>23082,]
dim(matrice_prix_cheval[matrice_prix_cheval[,1]>23082,])
matrice_prix_cheval[matrice_prix_cheval[,3]>1.6,]
colMeans(matrice_prix_cheval[matrice_prix_cheval[,3]>1.6,])
colMeans(matrice_prix_cheval[matrice_prix_cheval[,3]<1.6,])

colSums(matrice_prix_cheval)

#######################################
# Commandes session 4 : les dataframes
#######################################

mon_dataframe=data.frame(c(18,26,54,78), c(56,84,76,62), c("M", "F", "M", "F"), c(TRUE
,TRUE,TRUE,FALSE))
colnames(mon_dataframe)=c("Age", "Poids", "Sexe", "Ma valeur booléenne")
rownames(mon_dataframe)=c("Jean", "Zoé", "Lucas", "Chloé")

data_iris=read.table("iris.csv", header = TRUE, sep=",")
data_iris=read.table("iris.csv", header = TRUE, sep=",", row.names=1)
data_iris=read.csv("iris.csv", row.names=1)
data("iris")
ls()
write.table(data_iris, file="iris_2.csv", sep=",", row.names=TRUE)
write.csv(data_iris, file="iris_2.csv", row.names=TRUE)
save(data_iris, file="iris.Rdata")
load("iris.Rdata")

data_iris[,1]
data_iris[,c(1:3)]
head(data_iris[,c(1:3)])
head(data_iris[c(1,50,60),c(1:3)])
head(data_iris[c(1,50,60),c("Sepal.Length","Sepal.Width","Petal.Length")])
head(data_iris$Species)

data_iris[data_iris$Species == "setosa",]
dim(data_iris[data_iris$Species == "setosa",])
data_iris$Species == "setosa"
data_iris[which(data_iris$Species == "setosa"),]
which(data_iris$Species == "setosa")

```

```r
data_iris[which(data_iris$Species == "setosa" & data_iris$Petal.Length == 1.4),]
which(data_iris$Species == "setosa" & data_iris$Petal.Length == 1.4)

data_iris[data_iris$Species %in% c("setosa", "versicolor"),]
data_iris$Species %in% c("setosa", "versicolor")

subset(data_iris, Species="setosa" & Petal.Length == 1.4)

subset(data_iris, Species="setosa" & Petal.Length == 1.4, select=c("Petal.Length"))

class(iris$Species)
head(iris$Species)
colnames(iris)=c("Longueur des sépales","Largeur des sépales","Longueur des pétales",
"Largeur des pétales","Espèce")
rownames(iris)=paste("iris_", rownames(iris), sep="")
summary(iris)

min(iris[iris$"Espèce" == "setosa",3])
mean(iris[iris$"Espèce" == "setosa",3])
mean(iris[iris$"Espèce" == "versicolor",3])
mean(iris[iris$"Espèce" == "virginica",3])

data_iris_quantitative=iris[,c(1,2,3,4)]
data_iris_qualitative=iris[,5]
data_iris_qualitative=as.data.frame(iris[,5])
colnames(data_iris_qualitative)=c("Espèce")
rownames(data_iris_qualitative)=paste("iris_", rownames(data_iris_qualitative), sep=""
)
iris_complet=merge(data_iris_quantitative,data_iris_qualitative, by="row.names")
rownames(iris_complet)=iris_complet[,1]
iris_complet=iris_complet[,-1]

head(cbind(data_iris_quantitative,data_iris_qualitative))
head(rbind(iris,iris))

dim(rbind(iris,iris))
dim(iris)

############################################################
# Commandes session 5 : les bases de la programmation en R
############################################################

# opérateur 1 : strictement supérieur
5 > 3
# < strictement inférieur
4 < 6
2 < 1
# >= <=
4 <= 4
4 >= 4
# ==
3 == 3
# !=
3 != 3

# &, &&
2 == 2 & 1 == 1
2 == 2 && 1 == 1

c(2,2) == c(2,2) & c(3,2) == c(2,3)
c(2,2) == c(2,2) && c(3,2) == c(2,3)

c(2,2) == c(2,2) && c(3,2) == c(3,3)
c(2,2) == c(2,2) & c(3,2) == c(3,3)

# ou, |, ||

2 == 2 | 1 == 2
2 == 2 & 1 == 2

# instructions de conditions
#if(condition){
#  action1
```

```r
#} else {
#  action2
#}

if( 5 > 3 & 2 < 3 ){
  print("OK")
} else {
  print("PAS OK")
}

if( 5 == 3 & 2 < 3 ){
  print("OK")
} else {
  print("PAS OK")
}

# instruction de boucles
#for (valeur in vecteur){
#  action1
#}

for(valeur in c(1,2,3,4,5)){
  print(valeur + 1)
}

for(element in c(1,2,3,4,5)){
  print(paste("Mon chiffre :", element))
}

for(element in c(1,2,3,4,5)){
  if(element > 1 & element < 5){
    print(element)
  }
}


# instructions de boucle while : tant que

#while(condition){
#  actions
#}

valeur = 200

while( valeur/5 > 1){
  valeur = valeur/5
  print(valeur)
}

data(iris)

compteur_individu_sepal_superieur_5=0
for( length in iris$Sepal.Length ){
  if( length >= 5 ){
    compteur_individu_sepal_superieur_5=compteur_individu_sepal_superieur_5+1
  }
}

compteur_setosa=0
compteur_versicolor=0
compteur_virginica=0

for(species in iris$Species){
  if(species == "setosa"){
    compteur_setosa=compteur_setosa+1
  } else if( species == "versicolor"){
    compteur_versicolor=compteur_versicolor+1
  } else {
    compteur_virginica=compteur_virginica+1
  }
}

print(paste("Nombre de setosa :", compteur_setosa))
```

```r
print(paste("Nombre de versicolor :", compteur_versicolor))
print(paste("Nombre de virginica :", compteur_virginica))


dim(iris)[1]

nombre_setosa_sepal_sup_5=0
for(ligne in 1:dim(iris)[1]){
  individu=iris[ligne,]
  if( individu$Species == "setosa" & individu$Sepal.Length >=5 ){
    nombre_setosa_sepal_sup_5=nombre_setosa_sepal_sup_5+1
  }
}
print(nombre_setosa_sepal_sup_5)


for(colonne in 1:dim(iris)[2]){
  print(iris[,colonne])
  print("-----------")
}


#nom_de_notre_fonction<-function(argument1,argument2,...){
#  instructions
#    bloc code
# #  return(resultat)
#}


nombre_individus_superieur_5<-function(dataframe){
  compteur_individu_sepal_superieur_5=0
  for( length in dataframe$Sepal.Length ){
    if( length >= 5 ){
      compteur_individu_sepal_superieur_5=compteur_individu_sepal_superieur_5+1
    }
  }
  return(compteur_individu_sepal_superieur_5)
}

nombre_individus_superieur_5(iris)


ma_fonction_qui_calcule_la_moyenne<-function(iris){
  iris_species_setosa=subset(iris, iris$Species=="setosa")
  mean_setosa=colMeans(iris_species_setosa[,-5])
  mean_versicolor=colMeans(subset(iris, iris$Species=="versicolor")[,-5])
  mean_virginica=colMeans(subset(iris, iris$Species=="virginica")[,-5])
  resultat=data.frame(Setosa=mean_setosa, Versicolor=mean_versicolor, Virginica=
  mean_virginica)
  return(resultat)
}

ma_fonction_qui_calcule_la_moyenne(iris)

############################################################
# Commandes session 5 : manipulation avancée des données
############################################################

# apply
# apply(X, MARGIN, FUN)
# X : dataframe, une matrice
# MARGIN : 1 : pour les lignes, 2 : pour les colonnes et colonnes + lignes : c(1,2)
# FUN : fonction (mean, sum, summary, ...)

data(iris)
apply(iris[,-5], 2, mean)
apply(iris[,-5], 1, mean)

apply(iris[,-5], 2, summary)

nombre_valeurs_superieures_5<-function(vecteur){
  length(vecteur[vecteur>5])
}
```

```r

apply(iris[,-5], 2, nombre_valeurs_superieures_5)
apply(iris[,-5], 1, nombre_valeurs_superieures_5)

# by
# by(X, INDICES, FUN)
# X : dataframe
# INDICES : iris$Species
# FUN : fonction

by(iris, iris$Species, summary)
by(iris[,-5], iris$Species, cor)
by(iris[,-5], iris$Species, mean)

# aggregate
# aggregate(X, BY, FUN)
# X : dataframe
# BY : iris$Species
# FUN : fonction

aggregate(iris[,-5], as.data.frame(iris$Species), mean)

library(dplyr)
class(iris)

# tibble

iris_data=as_tibble(iris)
iris_data

# select : selectionner des colonnes
select(iris, Sepal.Length, Petal.Length, Species)
select(iris, Sepal.Length:Petal.Length)
select(iris, -Species)
select(iris, starts_with("Petal"))
select(iris, -starts_with("Sepal"))
select(iris, ends_with("Length"))
select(iris, contains("al"))

# filter : filtrer sur les individus
filter(iris, Sepal.Length >=5, Sepal.Width >=2)
filter(iris, between(Sepal.Length, 4, 7))
filter(iris, Sepal.Length >=4, Sepal.Length <=7)
filter(iris, Species == "setosa")
filter(iris, Species != "setosa")
filter(iris, Species %in% c("setosa", "versicolor"))
filter(iris, (Species == "setosa" | Species =="versicolor"))
filter_all(iris[,-5], any_vars(. > 5))
filter(iris, (Sepal.Length >5 | Sepal.Width >5 | Petal.Length >5 | Petal.Width >5))
filter_all(iris[,-5], all_vars(. > 2))

# %>% :
select(iris, Sepal.Length, Petal.Length, Species)

iris %>%
select(Sepal.Length, Petal.Length, Species)

iris %>%
select(-Species) %>%
filter_all(all_vars(. > 2))

# arrange
iris %>%
arrange(Sepal.Length)

iris %>%
arrange(desc(Sepal.Length))

iris %>%
arrange(Sepal.Length, Sepal.Width)

iris %>%
select(Petal.Length, Petal.Width, Species) %>%
```

```r
  filter(Species == "setosa") %>%
  arrange(Petal.Length, Petal.Width)

  # summarise : résumé statistique d'un vecteur qui retourne une valeur.

  iris %>%
  summarise(moyenne_taille_petal=mean(Petal.Length))

  iris %>%
  summarise(moyenne_taille_petale=mean(Petal.Length),
  minimum_taille_petale=min(Petal.Length),
  maximum_taille_petale=max(Petal.Length),
  total=n())

  iris %>%
  summarise(moyenne_taille_petale=mean(Petal.Length),
  moyenne_taille_sepale=mean(Sepal.Length),
  minimum_taille_petale=min(Petal.Length),
  minimum_taille_sepale=min(Sepal.Length)
  )

  iris %>%
  summarise_each(funs(mean, min), Petal.Length, Sepal.Length)


  # group_by()
  iris %>%
  group_by(Species) %>%
  summarise(moyenne_taille_petale=mean(Petal.Length),
  minimum_taille_petale=min(Petal.Length),
  maximum_taille_petale=max(Petal.Length),
  total=n())

  iris %>%
  group_by(Species) %>%
  filter(Petal.Length > 5) %>%
  summarise(n())

  # mutate : ajouter, supprimer, modifier

  # ajouter une ou plusieurs variables
  iris %>%
  mutate(somme_longueur_largeur_petale=Petal.Length+Petal.Width,
  somme_longueur_largeur_sepale=Sepal.Length+Sepal.Width
  )

  # supprimer une ou des variables
  iris %>%
  mutate(Species=NULL, Sepal.Width=NULL)

  # modifier une variable
  iris %>%
  mutate(Sepal.Length=Sepal.Length*2)

  ######################
  # cas pratique fast food
  ######################

  # lien dropbox
  # https://www.dropbox.com/s/pr288bu5slmk2jk/FastFoodRestaurants.csv?dl=0

  # lecture du fichier fast food aux US
  fast_food=read.csv("FastFoodRestaurants.csv")

  # transformation en tibble
  fast_food_tibble=as_tibble(fast_food)

  # Quelles sont les 5 villes avec le plus de fast-food ?

  fast_food_tibble %>%
  group_by(city) %>%
  summarise(Nombre_de_restaurants=length(city)) %>%
  arrange(desc(Nombre_de_restaurants)) %>%
```

```r
head(n=5)

# Quels sont les fast food les plus présents dans ces 5 villes ?

city_list=fast_food_tibble %>%
group_by(city) %>%
summarise(Nombre_de_restaurants=length(city)) %>%
arrange(desc(Nombre_de_restaurants)) %>%
head(n=5) %>%
pull(city)

fast_food_tibble %>%
filter(city %in% city_list)

fast_food_tibble %>%
filter(city %in% city_list) %>%
group_by(name)  %>%
summarise(nombre_de_fast_food=length(name)) %>%
arrange(desc(nombre_de_fast_food))

# Quels sont les fast food avec le plus de restaurants aux US ?
fast_food_tibble %>%
group_by(name) %>%
summarise(nombre_de_fast_food=length(name), pourcentage_de_fast_food=(length(name)*100/10000)) %>%
arrange(desc(nombre_de_fast_food))

# Dans quelle ville y a-t-il le plus de McDonald's ?
fast_food_tibble %>%
filter(name %in% "McDonald's")  %>%
group_by(city) %>%
summarise(nombre_de_fast_food=length(city)) %>%
arrange(desc(nombre_de_fast_food))


# Où¹ se situe New-York par rapport aux 5 villes avec le plus de fast-foods ?

fast_food_tibble %>%
group_by(city) %>%
filter( city %in% "New York")  %>%
summarise(nombre_de_fast_food=length(city))


# Fast food les plus présents àNBSP New York
fast_food_tibble %>%
filter( city %in% "New York")  %>%
group_by(name) %>%
summarise(nombre_de_fast_food=length(name), pourcentage_de_fast_food=(length(name)*100/10000)) %>%
arrange(desc(nombre_de_fast_food))

# Forbes : Orlando, Cincinnati And The Fast Food Capitals of the US.

############################################################
# Commandes session 7 : visualisation avancée des données
############################################################

data(iris)

# plot()

plot(iris$Sepal.Length, iris$Petal.Width)

plot(iris$Sepal.Length, iris$Sepal.Width, xlab="Longueur", ylab="Largeur", col="red", xlim=c(min(iris$Sepal.Length,iris$Petal.Length),
max(iris$Sepal.Length,iris$Petal.Length)), ylim=c(min(iris$Sepal.Width,iris$Petal.Width),max(iris$Sepal.Width,iris$Petal.Width)))
lines(iris$Petal.Length, iris$Petal.Width, col="slateblue4", type="p", pch=22)
title(main="Longueur en fonction de largeur", col="blue")
legend(1, 4.2, c("Sépales", "Pétales"), col=c("red", "slateblue4"), pch=21:22)

library("ggplot2")
# 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 àNBSP 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+xlab("Longueur des pétales")+ylab("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
```

```r

# modification des noms des axes et ajout d'un titre au graphique
g<-ggplot(iris, aes(x=Petal.Length, y=Petal.Width, color=Petal.Length, shape=Species))+geom_point(size=3)
g<-g+scale_color_gradient(low="blue", high="red")+theme_minimal()+theme(legend.title=element_text(colour="steelblue", size=9, face="bold"))
g<-g+theme(legend.text=(element_text(colour="blue", size=8, face="bold.italic")))
g<-g+xlab("Longueur des pétales")+ylab("Largeur des pétales")+ggtitle("Longueur des pétales en fonction de largeur des pétales")
g<-g+theme(plot.title=element_text(colour="steelblue", size=15, face="bold"))
g<-g+theme(axis.title=element_text(colour="steelblue", size=10, face="bold"))
g<-g+theme(axis.text=element_text(colour="steelblue", size=10, face="bold", angle=45))
#g<-g+theme(axis.line=element_line(colour="steelblue", size=2, linetype="dotted"))
g

# facet_wrap -> combiner des graphiques
g<-ggplot(iris, aes(x=Petal.Length, y=Petal.Width, color=Petal.Length, shape=Species))+geom_point(size=3)
g<-g+scale_color_gradient(low="blue", high="red")+theme_minimal()+theme(legend.title=element_text(colour="steelblue", size=9, face="bold"))
g<-g+theme(legend.text=(element_text(colour="blue", size=8, face="bold.italic")))
g<-g+xlab("Longueur des pétales")+ylab("Largeur des pétales")+ggtitle("Longueur des pétales en fonction de largeur des pétales")
g<-g+theme(plot.title=element_text(colour="steelblue", size=15, face="bold"))
g<-g+theme(axis.title=element_text(colour="steelblue", size=10, face="bold"))
g<-g+theme(axis.text=element_text(colour="steelblue", size=10, face="bold", angle=45))
#g<-g+theme(axis.line=element_line(colour="steelblue", size=2, linetype="dotted"))
g<-g+facet_wrap(~Species)+theme(strip.text=element_text(colour="steelblue", size=10, face="bold"))
g<-g+theme(strip.background=element_rect(colour="steelblue", size=1,linetype="solid"))
g


# modification des noms des axes et ajout d'un titre au graphique
g<-ggplot(iris, aes(x=Petal.Length, y=Petal.Width, color=Petal.Length, shape=Species))+geom_point(size=3)
g<-g+scale_color_gradient(low="blue", high="red")+theme_minimal()+theme(legend.title=element_text(colour="steelblue", size=9, face="bold"))
g<-g+theme(legend.text=(element_text(colour="blue", size=8, face="bold.italic")))
g<-g+xlab("Longueur des pétales")+ylab("Largeur des pétales")+ggtitle("Longueur des pétales en fonction de largeur des pétales")
g<-g+theme(plot.title=element_text(colour="steelblue", size=15, face="bold"))
g<-g+theme(axis.title=element_text(colour="steelblue", size=10, face="bold"))
g<-g+theme(axis.text=element_text(colour="steelblue", size=10, face="bold", angle=45))
#g<-g+theme(axis.line=element_line(colour="steelblue", size=2, linetype="dotted"))
g<-g+annotate("text", x=c(2,4,6), y=0.7, label=c("Setosa","Versicolor","Virginica"), colour="steelblue", size=3, fontface="bold")
g<-g+annotate("rect", xmin=0.5, xmax=2.1, ymin=0, ymax=0.65, alpha=0.2, colour="steelblue", size=2)
g<-g+annotate("segment", x=0.5, xend=4, y=1.5, yend=0, colour="steelblue", size=2, alpha=0.5)
g

# histogramme sur les données iris
png("histogramme_iris.png")
g<-ggplot(iris, aes(x=Petal.Length, fill=Species))+geom_histogram(color="white", binwidth =0.5)
g
dev.off()

pdf("boxplot_iris.pdf")
g<-ggplot(iris, aes(x=Species, y=Petal.Length, fill=Species))+geom_boxplot()
g<-g+ggtitle("Boxplot de la longueur des pétales selon l'espèce")+xlab("Espèce")+ ylab("Longueur des pétales")
g
dev.off()

##############################
# exercice ggplot2
##############################

library("ggplot2")
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()+
    theme_minimal()
789 g<- g + xlab("Les 10 capitales du fast-foods") + ylab("Les 10 restaurants les plus
    implantés")
790 g<- g + ggtitle("Représentation des fast-foods les plus implantés \n dans les 10
    capitales du fast-food")
791 g<- g + theme(plot.title=element_text(hjust=0.5), axis.text=element_text(face="bold",
    size=7, angle=45))
792 g <- g + ylim(0, 100) + theme(legend.title=element_blank()) + scale_fill_brewer(
    palette="Paired")
793 g
794 dev.off()
795
796 install.packages("plotly")
797 library("plotly")
798
799 ggplotly(g)
800
801 g<-ggplot(iris, aes(x=Species, y=Petal.Length, fill=Species))+geom_boxplot()
802 g<-g+ggtitle("Boxplot de la longueur des pétales selon l'espèce")+xlab("Espèce")+ ylab
    ("Longueur des pétales")
803 g
804
805 ggplotly(g)
806
807 g<-ggplot(iris, aes(x=Petal.Length, y=Petal.Width, color=Petal.Length, shape=Species
    ))+geom_point(size=3)
808 g<-g+scale_color_gradient(low="blue", high="red")+theme_minimal()+theme(legend.title=
    element_text(colour="steelblue", size=9, face="bold"))
809 g<-g+theme(legend.text=(element_text(colour="blue", size=8, face="bold.italic")))
810 g<-g+xlab("Longueur des pétales")+ylab("Largeur des pétales")+ggtitle("Longueur des
    pétales en fonction de largeur des pétales")
811 g<-g+theme(plot.title=element_text(colour="steelblue", size=15, face="bold"))
812 g<-g+theme(axis.title=element_text(colour="steelblue", size=10, face="bold"))
813 g<-g+theme(axis.text=element_text(colour="steelblue", size=10, face="bold", angle=45))
814 #g<-g+theme(axis.line=element_line(colour="steelblue", size=2, linetype="dotted"))
815 g<-g+annotate("text", x=c(2,4,6), y=0.7, label=c("Setosa","Versicolor","Virginica"),
    colour="steelblue", size=3, fontface="bold")
816 g<-g+annotate("rect", xmin=0.5, xmax=2.1, ymin=0, ymax=0.65, alpha=0.2, colour=
    "steelblue", size=2)
817 g<-g+annotate("segment", x=0.5, xend=4, y=1.5, yend=0, colour="steelblue", size=2,
    alpha=0.5)
818 g
819
820 ggplotly(g)
821
822
823 ############################################################
```

```r
# Commandes session 8 : cas pratique de data science
############################################################

# lecture du fichier de données
bank_data=read.csv("bank.csv", sep=";")

summary(bank_data)

# visualisation des données
library("ggplot2")

g<-ggplot(bank_data, aes(x=y, y=duration, fill=y))+geom_boxplot()
g

library("plotly")
ggplotly(g)


g<-ggplot(bank_data, aes(x=y, y=age, fill=y))+geom_boxplot()
g
ggplotly(g)

g<-ggplot(bank_data, aes(y, fill=contact))+geom_bar()
g

# Création jeu de test et jeu d'entrainement
library("caret")

dummy_variables=dummyVars(~., data=bank_data)
dummy_variables_data=predict(dummy_variables, newdata=bank_data)
dummy_variables_data=as.data.frame(dummy_variables_data)

dummy_variables_data$"Souscription"=ifelse(dummy_variables_data$"y.no" == 1, "No",
"Yes")
dummy_variables_data$"y.no"=NULL
dummy_variables_data$"y.yes"=NULL

# Création des jeux de données d'entrainement et de test
set.seed(3033)
training_size=floor(0.7*nrow(dummy_variables_data))
indices=sample(seq_len(nrow(dummy_variables_data)), size=training_size)
data_bank.train=dummy_variables_data[indices,]
data_bank.test=dummy_variables_data[-indices,]

dim(data_bank.train)
dim(data_bank.test)

# Normalisation des données

data_preprocess_value=preProcess(data_bank.train, method=c("center","scale"))
data_bank.train.scaled=predict(data_preprocess_value,data_bank.train)
data_bank.test.scaled=predict(data_preprocess_value,data_bank.test)

# Caret - downsample et upsample
table(data_bank.train.scaled[,"Souscription"])

set.seed(3033)
'%ni%' = Negate("%in%")

# downsample
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"))
names(data_bank.train.scaled.downsample)[names(data_bank.train.scaled.downsample) ==
"Class"]="Souscription"
table(data_bank.train.scaled.downsample[,"Souscription"])

# upsample
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"))
names(data_bank.train.scaled.upsample)[names(data_bank.train.scaled.upsample) ==
"Class"]="Souscription"
```

```r
table(data_bank.train.scaled.upsample[,"Souscription"])

# modélisation avec naive bayes
set.seed(3033)
trainControl_data=trainControl(method="repeatedcv", number=10, repeats=3)
naive_bayes_desequilibree=train(Souscription ~., data=data_bank.train.scaled, method=
"nb", preProcess=NULL)

print(naive_bayes_desequilibree)

# prédiction avec notre modèle sur le jeu de données tests
prediction_naive_bayes_desequilibree=predict(naive_bayes_desequilibree, newdata=
data_bank.test.scaled[,-ncol(data_bank.test.scaled)])

# création de la matrice de confusion
confusionMatrix(prediction_naive_bayes_desequilibree, as.factor(data_bank.test.scaled[
,ncol(data_bank.test.scaled)]))

# modélisation avec naive bayes sur les données downsamplé
set.seed(3033)
trainControl_data=trainControl(method="repeatedcv", number=10, repeats=3)
naive_bayes_downsample=train(Souscription ~., data=data_bank.train.scaled.downsample,
method="nb", preProcess=NULL)

print(naive_bayes_downsample)

# prédiction avec notre modèle sur le jeu de données tests
prediction_naive_bayes_downsample=predict(naive_bayes_downsample, newdata=
data_bank.test.scaled[,-ncol(data_bank.test.scaled)])

# création de la matrice de confusion
confusionMatrix(prediction_naive_bayes_downsample, as.factor(data_bank.test.scaled[,
ncol(data_bank.test.scaled)]))

# modélisation avec SVM
set.seed(3033)
trainControl_data=trainControl(method="repeatedcv", number=10, repeats=3)
SVM_desequilibree=train(Souscription ~., data=data_bank.train.scaled, method=
"svmLinear", preProcess=NULL)

print(SVM_desequilibree)

# prédiction avec notre modèle sur le jeu de données tests
prediction_SVM_desequilibree=predict(SVM_desequilibree, newdata=data_bank.test.scaled[
,-ncol(data_bank.test.scaled)])

# création de la matrice de confusion
confusionMatrix(prediction_SVM_desequilibree, as.factor(data_bank.test.scaled[,ncol(
data_bank.test.scaled)]))

# varImp
varImp(naive_bayes_downsample, scale=F)
```