```
######################## LAB 4.6.5 ########################################
# Descrizione: Regressione logistica
# Dataset: Smarket della libreria ISRL
# Matricola:
# CdL:
###########################################################################


#########################################################
######## 4.6.6 K-Nearest Neighbors             #########
#########################################################

####### 0 - Copia del dataset #######
### Carico la libreria
library(class)
library(ISLR)
library(ggplot2)

### Copiamo il dataset
mydf = Caravan
### Funzione di riferimento: lda(y~x)
### La funzione lda e' una generalizzata di lm(), effettua la Linear Discriminant Analysis
# contiene 85 predittori
# la variabile di risposta ? Purchase
# le osservazioni sono 5.822
# Ogni record conteine 86 variabili:
#      - sociodemographic data (variables 1-43)
#      - and product ownership (variables 44-86)
dim(mydf)
summary(mydf$Purchase)
# solo il 6% ha comprato una assicurazione per un Caravan
nrow(mydf[mydf$Purchase=="Yes",])/nrow(mydf)

####### 1 - Modello con knn() #######
### Il modello fa parte della libreria di base, questa funzione fa la predizione in un solo step
### questa funzione richiede quattro input:
###      - una matrice contenente i predittori associati al data training chiamata train.X
###      - una matrice che contiene i predittori che utilizziamo per fare le predizioni chiamata test.X
###      - un vettore contenente le classi etichettate per le osservazioni del training chiamata train.Direction
###      - Un valore per K, il numero del vicini che il classificatore deve usare
### Dato che KNN lavora con le distanze, la scala utilizzata per la variabile ha una sua importanza,
### infatti le variabili che sono su una scala pi? grande hanno maggiori effetti sulle distanze tra le osservazioni
### rispetto a quelle che si trovano su una scala pi? piccola. Esempio di due variabili salary (dollari) and age
(anni), in tal senso age potrebbe non avere effetti

#SCALA - STANDARDIZZAZIONE
# un modo per gestire il problema ? Standardizzare le variabili
standardized.X =  scale(mydf[,sapply(mydf, is.numeric)]) # eslcude le non numeriche # Deviazione standard di 1 e
media uguale a 0

# Varianze differenti
var(mydf[,1])
var(mydf[,2])

# Varianze uguali
var(standardized.X[,1])
var(standardized.X[,2])


# Validation SET APPROACH
set.seed(1)
nperc = 0.8283
index_train = sample(1:nrow(mydf), nperc * nrow(mydf), replace = F)

length(mydf$Purchase[index_train])
length(mydf$Purchase[-index_train])

train.X = standardized.X[index_train,]
test.X = standardized.X[-index_train,]

train.Y = mydf$Purchase[index_train]
test.Y = mydf$Purchase[-index_train]
knn.pred = knn(train.X, test.X, train.Y, k = 1)
# Verifichiamo l'errore del knn rispetto alla realta', appena dell'11%
# in realta', potremmo abbassare l'errore al 6% se banalmente predicessimo sempre un No!
mean(test.Y != knn.pred)
# conferma che il test e' in linea con la MEDIA GENERALE, solo il 6% si comprano un caravan
mean(test.Y == "Yes")


####### 5 - Matrice di confusione #######
### La matrice di confusione ci indica quante osservazioni sono state correttamente assegnate rispetto alla
predizione
### Gli elementi nella diagolane della matrice di confusione ci indicano le predizoni corrette
# ci ha azzeccato solo 11%, rispetto a quello che potremo fare se azzeccassimo in modo casuale
```

```
#test.Y
#knn.pred  No Yes
#     No  879  57
#     Yes  59   5
table(knn.pred, test.Y)

#Limitiamo il caso su potenziali chienti (YES-YES o YES-NO) ovvero 59+5, sarebbero quelli che verosimilmente
#si potrebbero trasformare in clineti reali
table(knn.pred, test.Y)[2,2]/(table(knn.pred, test.Y)[2,1]+table(knn.pred, test.Y)[2,2])

# Verifichiamo al variare del k se miglioriamo il tasso
# maxk =(length(train.Y)-1)
maxk = 5
client.prospect = rep(NA, maxk)

for(i in 1: maxk){
  knn.pred = knn(train.X, test.X, train.Y, k = i)
  # Verifica accuratezza del modello
  client.prospect[i] = mean(test.Y == knn.pred)
  #table(knn.pred, test.Y)[2,2]/(table(knn.pred, test.Y)[2,1]+table(knn.pred, test.Y)[2,2])
#  print("--------")
  print(i)
#  print(table(knn.pred, test.Y))
#  print("--------")
  if((table(knn.pred, test.Y)[2,1]+table(knn.pred, test.Y)[2,2])==0)
  {
    print("fine")
    break
  }
}
clip = as.data.frame(client.prospect)
plot(clip$client.prospect)
# il K da scegliere per massimizzare l'accuratezza di Carvan ? = 7
points(which.max(clip$client.prospect), clip$client.prospect[which.max(clip$client.prospect)],
col="red",cex=2,pch=20)




# Verifica con una Binomiale
glm.fit = glm(mydf$Purchase~., data = mydf, family="binomial", subset = index_train)
glm.probs = predict(glm.fit, Caravan[-index_train,], type="response")

glm.pred = rep("No", nrow(Caravan[-index_train,]))
soglia = .20
glm.pred[glm.probs > soglia] = "Yes"
table(glm.pred, test.Y)
#       test.Y
# glm.pred  No Yes
#     No  920  58
#     Yes  15   7
# molto meglio del KNN
table(glm.pred, test.Y)[2,2]/(table(glm.pred, test.Y)[2,1]+table(glm.pred, test.Y)[2,2])
```