Études des méthodes de classification sur la table spam

Par Quentin Guardia, quentin.guardia@etu.u-paris.fr
M1 Cybersécurité FI

Fichier associé : spam.R

Question 1 :

?spam nous indique :

```
    A data set collected at Hewlett-Packard Labs, that classifies 4601
    e-mails as spam or non-spam. In addition to this class label there
    are 57 variables indicating the frequency of certain words and
    characters in the e-mail.
```

Question 2 :

Analyse univariée

summary(spam) donne beaucoup d'informations sur les valeurs de chaque colonne.

Pour ne citer que deux résumés statistiques:

```
> summary(make)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.0000  0.0000  0.0000  0.1046  0.0000  4.5400
> summary(capitalAve)
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
   1.000    1.588    2.276    5.191    3.706 1102.500
```

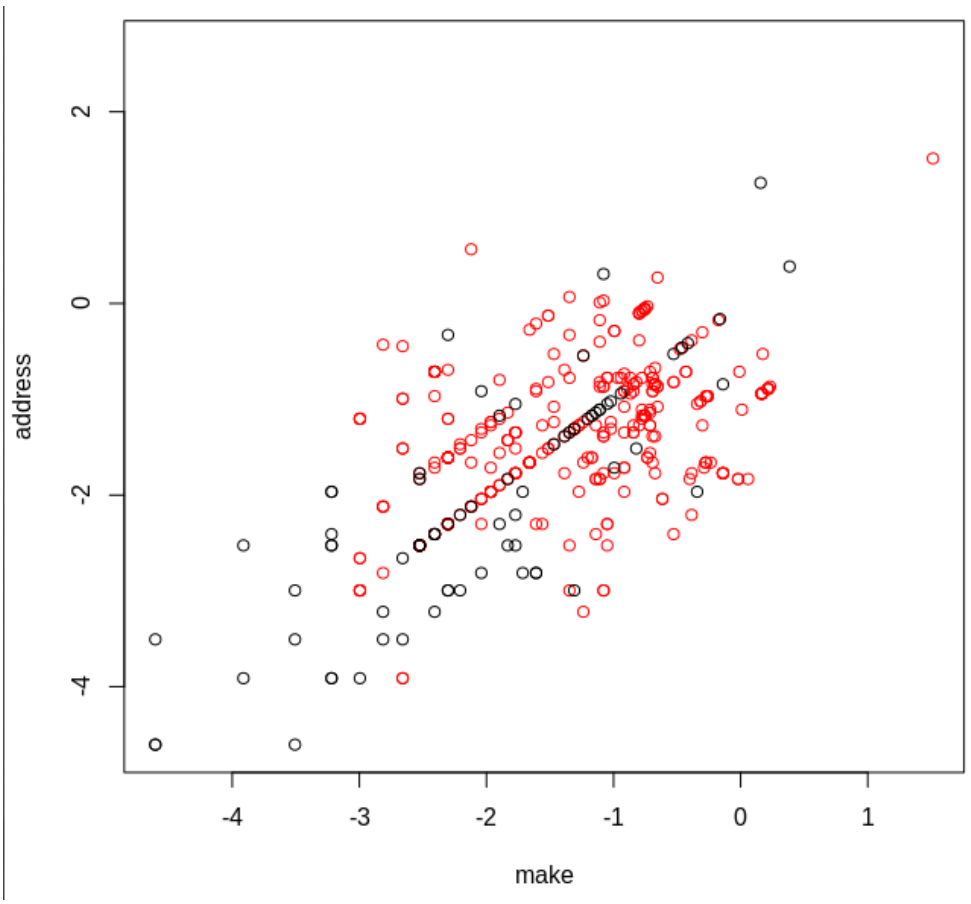
La dernière colonne, type, est plus particulière car elle ne contient pas de fréquence mais l'information spam ou non spam :

```
> prop.table(table(spam$type))
  nonspam      spam
0.6059552 0.3940448
```

Analyse bivariée

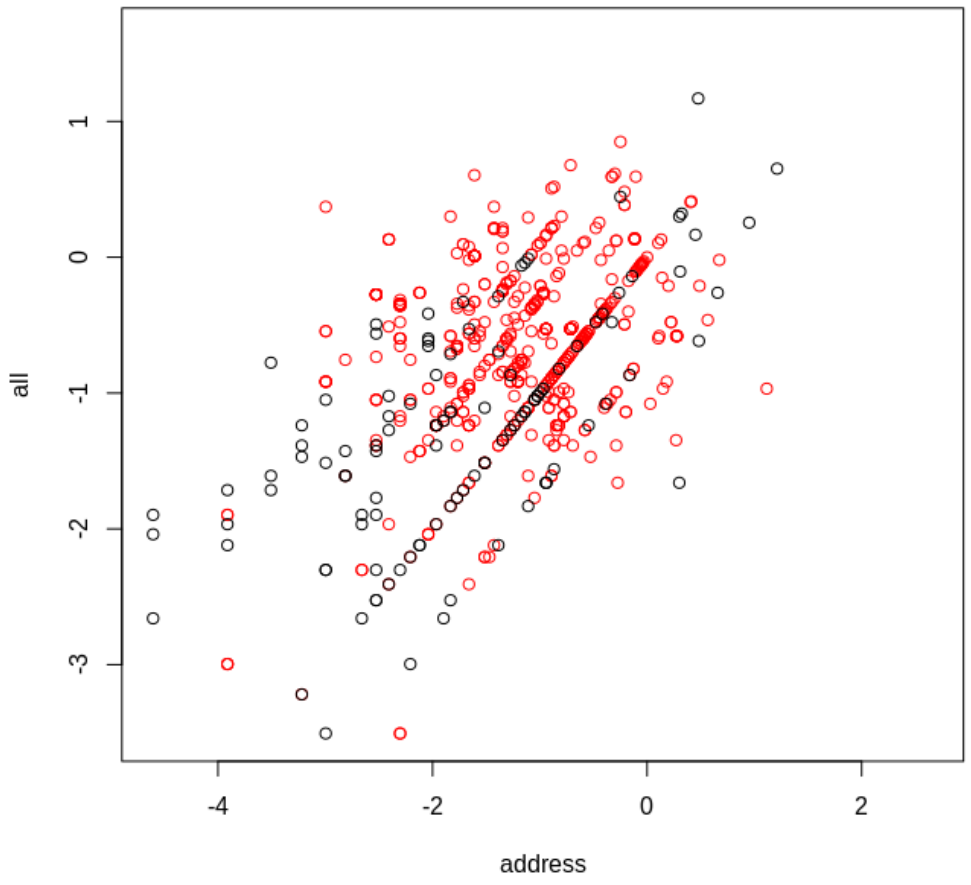
On affiche la première colonne en fonction de la deuxième colonne en coloriant les points selon s'ils sont des spams ou non. En rouge les nonspam et en noir les spam. Avec une échelle logarithmique pour y voir plus clair.

```
plot(log(spam[1:2]), col=type)
```



Même principe mais avec la deuxième et troisième colonne :

```
plot(log(spam[2:3]), col=type)
```



Question 3 :

Régression linéaire :

Après binarisation de la colonne type (spam=1, nonspam=0), j'applique une régression linéaire comme ceci :

```
bin <- ifelse(type=="spam", 1, 0)
lm.fit <- lm(bin~.-type,data=spam)
yhat <- predict(lm.fit)
lm.ghat <- factor(ifelse(yhat > 0.5, "spam", "nonspam"))
```

L'étude de l'erreur ci-dessous :

```
sum(lm.ghat != type)
mean(lm.ghat != type)
table(lm.ghat, type)
```

Nous indique qu'il y a 515 erreurs, soit un taux d'erreur de 0.1119322 en moyenne, avec une cette répartition :

```
 lm.ghat   nonspam spam
  onspam    2665     392
     spam      123    1421
```

Méthode des k plus proches voisins

Nous appliquons maintenant la fonction knn permettant l'usage de la méthode des k plus proches voisins, avec k le plus optimal possible. Pour cela j'ai testé avec plusieurs k possibles. Puis je fais une moyenne des résultats obtenus pour le meilleur k. Dans le code final je m'arrête à 10 itérations par soucis de temps d'exécution mais j'ai testé avec itérations=100, et le résultat est sensiblement le même

```
library(class)
tr <- sample(1:nrow(spam),3500) #environ 0.75*nrow(spam)
train <- spam[tr,]
test <- spam[-tr,]
iterations= 10
erreur <- vector(mode="integer", length=iterations)
moyenne <- 0
for(k in 1:iterations){
        knn.fit<-knn(train=train[,-58], test=test[,-58], cl=train[,58], prob=F , k)
        erreur[k] <- sum(test$type != knn.fit)/length(test$type)
}
k=which.min(erreur)
for(i in 1:iterations){
        knn.fit<-knn(train=train[,-58], test=test[,-58], cl=train[,58], prob=F , k)
        moyenne <- moyenne + mean(test$type != knn.fit)
}
moyenne <- moyenne/iterations
moyenne
```

Le code nous indique qu'avec le meilleur k il y a un taux d'erreur moyen d'environ 0.1882834.

Classifieur bayésien naïf

On applique le naiveBayes de la bibliothèque e1071 sur la colonne type avec les données de spam, sans la colonne type.

```
library(e1071)
m <- naiveBayes(type ~ .-type, data = spam)
p <- predict(m, spam)
```

Cependant,

```
sum(p != type)
mean(p != type)
table(p, type)
```

Nous indique qu'il y a 1318 erreurs soit un taux de 0.2864595. Voici comment les erreurs sont exactement réparties :

```
        type
p           nonspam spam
  nonspam    1564   94
    spam       1224 1719
```

Synthèse

En prenant comme facteurs le nombre d'erreurs et la moyenne d'erreur, on s'aperçoit que la méthode de régression linéaire est plus efficace que celle des k plus proches voisins, qui est elle-même plus efficace que celle du classifieur bayésien naïf.

Question 4 :

Normalisation

Je normalise avec la méthode indiquée dans la consigne, puis j'ajoute la colonne type à la nouvelle table normalisée

```
spamNorm<-spam[1:57]
colonneSomme <- colSums(spamNorm)
ligneSomme <- rowSums(spamNorm)
for(i in 1:length(ligneSomme)){
        for(j in 1:length(colonneSomme)){
                spamNorm[i,j]=spam[i,j]/sqrt(ligneSomme[i]*colonneSomme[j])
        }
}
spamNorm["type"]=spam$type
```

Puis je réapplique chaque méthode, en utilisant spamNorm et non spam.

Comparatif des méthodes

Concernant la régression linéaire, moyenne de taux d'erreur de 0.1388829.
Au sujet de la méthode des k plus proches voisins le taux s'élève à 0.07811081.
À propos du classifieur bayésien naïf, le taux est à 0.3251467.

Question 5 :

On peut classer ces trois méthodes de classification par ordre de précision sur la table spam : régression linéaire, la méthode des k plus proches voisins et classifieur bayésien naïf. La régression linéaire a un taux d'erreur de 11%, la méthode de k plus proches voisins de 19 % et celle du classifieur bayésien naîf de 28 %.

La normalisation permet entre autre de diminuer les écarts entre les valeurs. Après normalisation, on s'aperçoit que la méthode des k plus proches voisins est bien plus précise avec un taux d'erreur inférieur, approximant 8 %. La précision du classifieur bayésien naïf semble un peu moins bonne avec un taux d'erreur de 33 %. Tout comme celle de la régression linéaire, qui a un taux de 14 %.

Tableau représentant les taux d'erreurs :

	Valeurs d'origine	Valeurs normalisées
Régression linéaire	11 %	14 %
K plus proches voisins	19 %	**8 %**
Classifieur de Bayes naïf	28 %	33 %