Rapport Projet 2

Lucas Chabeau - Matthieu François - Etienne Hamard

22 avril 2019

Contents


```
load(".RData")
require(class)
```

```
## Loading required package: class
```

```
## Warning: package 'class' was built under R version 3.5.3
```

```
require(ggplot2)
```

```
## Loading required package: ggplot2
```

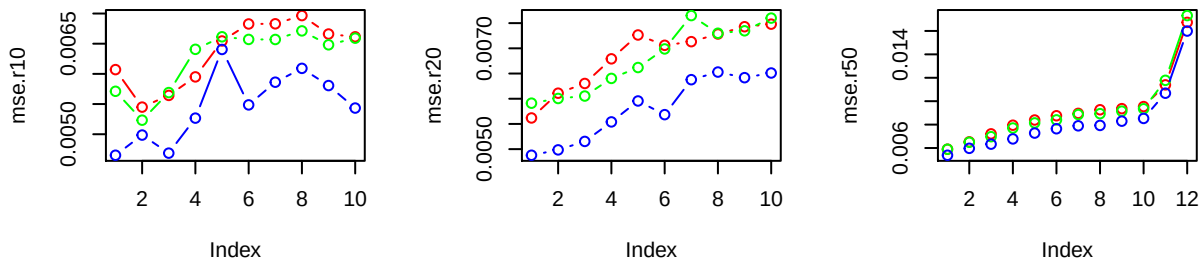
Introduction

Dans le cadre du cours d'Analyse de données 2, nous nous sommes penchés sur la réalisation d'un projet concernant la restauration d'image. Cela avec la méthode des k plus proches voisins (knn). L'étude met à disposition une image déclinés en 4 versions. Une version complète, une version avec 10% de valeur manquante, une avec 20% et enfin une avec 50%. Les images sont composés de 3 jeux données, le pourcentage de vert, de bleu et de rouge. Afin de réaliser l'algorithme des knn nous avons choisis de modifier un peu ces jeux de données. Par image nous avons réalisé trois knn, un par couleur, afin d'avoir un résultats plus précis. Il a donc extraire des trois images, les valeurs R, G et B pour les transofrmer en trois jeux de données différents, comportant la coordonnée x, et y associée.

Impact du choix du nombre de voisins K

Le choix le plus important dans l'algorithme des k plus proches voisins est le nombre de voisins K sur lesquels s'appuyer pour prédir la classe des individus concernés choisir. La présence de l'image originale nous permet donc de calculer la précision que peut avoir l'algorithme pour chaque K. Pour cela nous avons utilisé le Mean Squared Error (MSE).

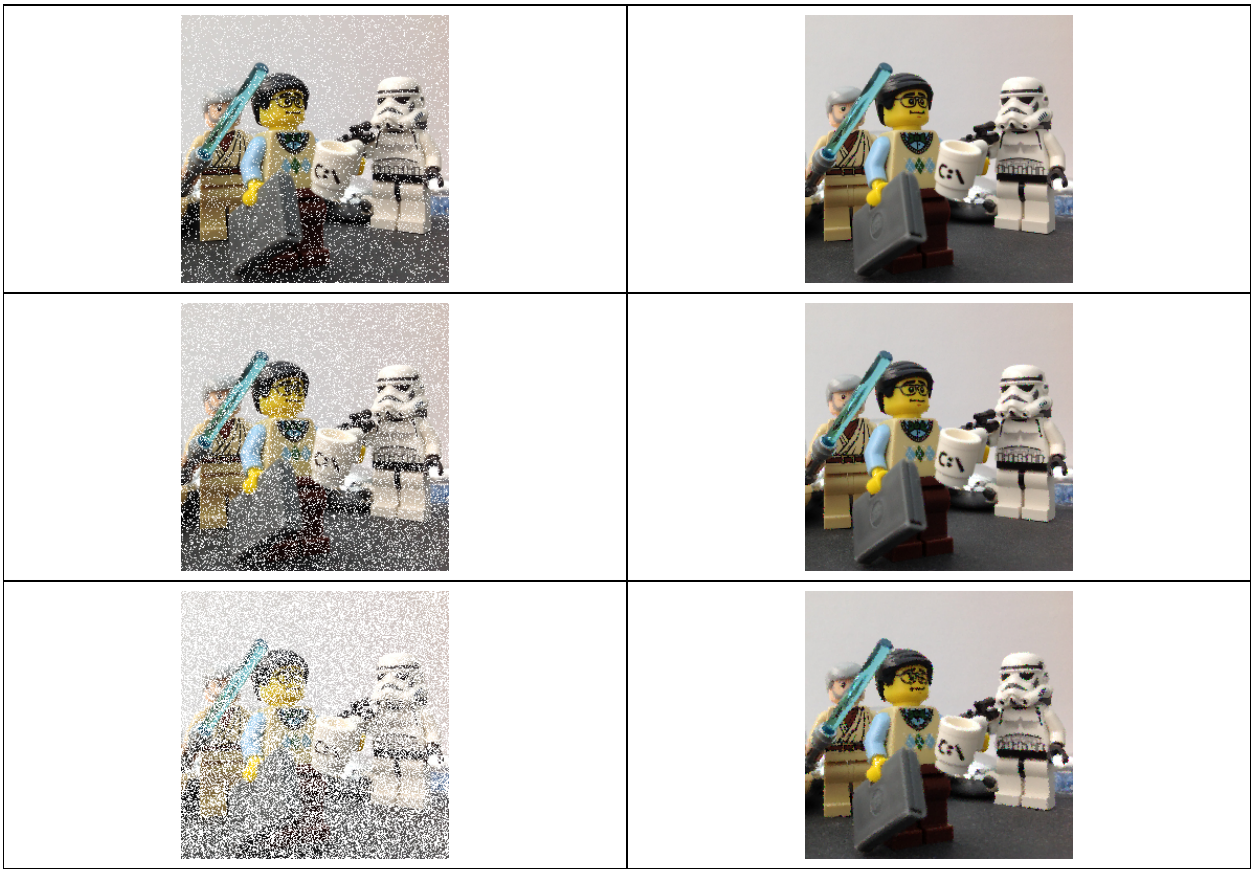
```r
par(mfrow=c(1,3))
#10%
plot(mse.r10,col="red",type="b",ylim = c(min(c(mse.r10,mse.b10,mse.g10)),max(c(mse.r10,mse.b10,mse.g10))
lines(mse.g10,col="green",type="b")
lines(mse.b10,col="blue",type="b")

##20%
plot(mse.r20,col="red",type="b",ylim = c(min(c(mse.r20,mse.b20,mse.g20)),max(c(mse.r20,mse.b20,mse.g20))
lines(mse.g20,col="green",type="b")
lines(mse.b20,col="blue",type="b")

#50%
plot(mse.r50,col="red",type="b",ylim = c(min(c(mse.r50,mse.b50,mse.g50)),max(c(mse.r50,mse.b50,mse.g50))
lines(mse.g50,col="green",type="b")
lines(mse.b50,col="blue",type="b")
```



Ces trois graphes permettent de montrer l'évolution du MSE en fonction du nombre de K voisins choisis. On peut voir nettement que plus l'image a un bruit important, plus le nombre de K voisins choisis pour l'algorithme des knn doit être faible ou le MSE n'en sera que plus grand. On peut voir que l'évolution du MSE de l'image avec 20% de bruit n'a pas la même évolution quasi-linéaire que l'image avec 50% de bruit, mais elle n'en est pas loin. Alors que pour l'image avec 10% de bruit, le choix d'utiliser qu'un seul voisin comme K n'est pas toujours le meilleur.

Nous avons donc choisis d'utiliser K = 1 pour les 3 images à reconstruire, étant donné qu'il s'agit du plus favorable aux 3 images réunis.

```r
par(mfrow=c(3,2))
display(img10)
display(img10new)
display(img20)
display(img20new)
display(img50)
display(img50new)
```



Impact du bruit sur l'efficacité de K

Enfin d'étudier l'impact du bruit sur le MSE en fonction de K fixé, nous avons choisis comme précédemment K = 1 étant donné qu'il s'agit du K entrainant le MSE minimal dans le plus de cas possible (2/3).

```
ggplot(df.mse,aes(x = image,y = mse,fill=couleur))+
  geom_bar(position = "dodge", stat = "identity") +
  scale_fill_manual(values=c("#3F93E8","#FF4F4A","#B8FF4F"))
```

Le graphe précédent permet de voir que pour K=1 les MSE des trois images sont très proches malgré le bruit allant de 10 à 50%. De plus pour on aurait pu s'attendre à un MSE croissant, suivant le pourcentage de bruit. Pourtant l'image avec 10% de bruit, à un MSE concernant la couleur verte qui est supérieur au deux autres images. Cela montre que certaines couleur sont plus ou moins affectées pas le bruit. Le choix de K=1 n'étant pas optimal pour l'image à 10% justifie aussi cette différence.

```
# require(knitr)
by(df.mse$mse,df.mse$image,sum)
```

```
## df.mse$image: image10
## [1] 0.0174686
## ------------------------------------------------------------
## df.mse$image: image20
## [1] 0.0168907
```

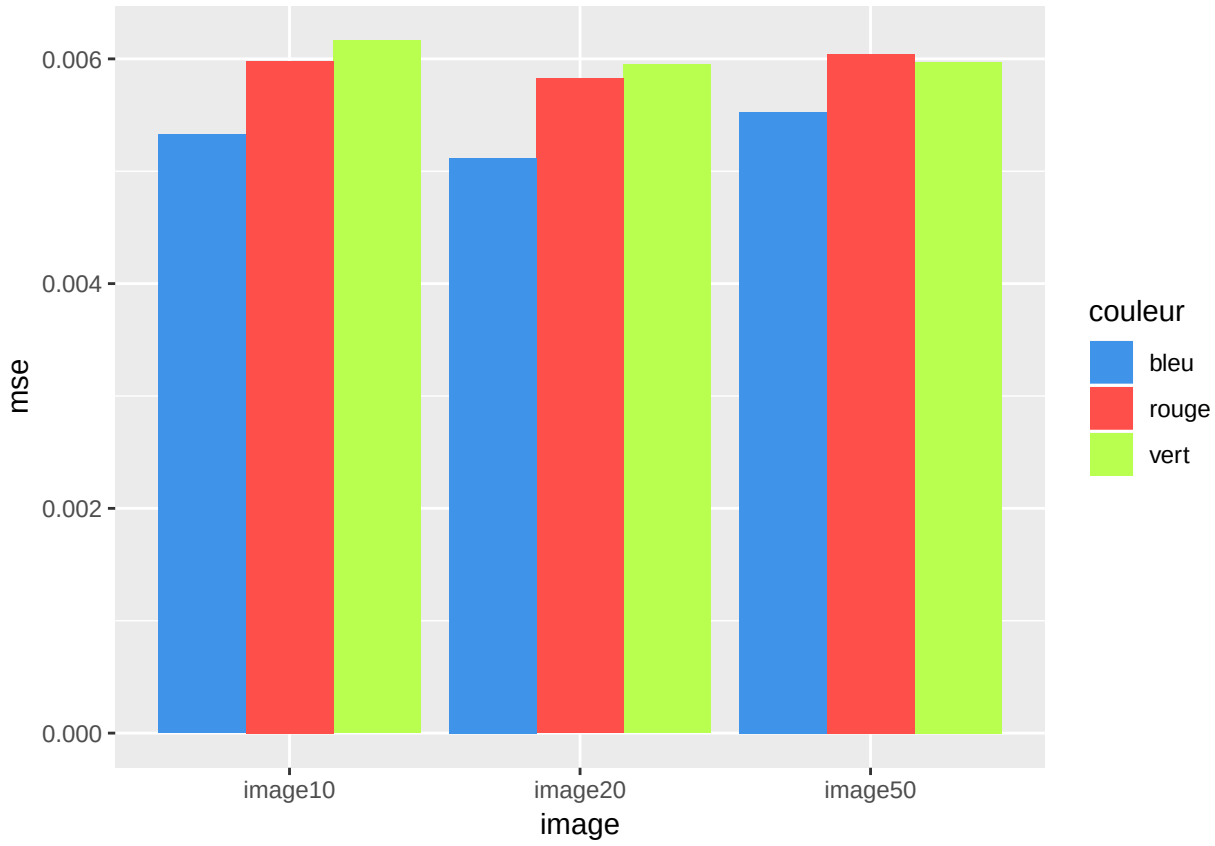


Figure 1: MSE par couleur et par image pour K=1

```
## ------------------------------------------------------
## df.mse$image: image50
## [1] 0.0175416
```

En sommant les erreurs de chaque image on voit que l'image avec 50% d'erreur arrive quand même en tête, toutefois l'image avec 20% est loin derrière.

Conclusion