Basic Machine Learning Methods

Pierre-Marie CERVERA–LARRICQ Fabien LOPEZ Gaby MAROUN

3 mars 2021

Contents

Introduction

Aujourd'hui le machine learning est de plus en plus à la mode. On le retrouve dans de plus en plus de domaines d'activités du au développement de l'informatique. On le retrouve aussi dans de plus en plus de bouches. A travers ce rapport nous allons d'abord présenter le sens du machine learning. Ensuite, nous allons nous poser une question très simple : "Quel est son but ?" Grâce à elle nous allons pouvoir présenter les différentes méthodes existantes, avec des exemples d'applications. Nous finirons enfin sur un thème aussi récurrent : "Quelle méthode est la plus utilisée aujourd'hui ?"

Qu'est-ce que le machine Learning

Je commencerai mon paragraphe en citant et traduisant la définition de Samuel Muller :

Le Machine Learning est le champ d'étude qui permet à l'ordinateur d'avoir la capacité d'apprendre sans être explicitement programmé.

Cela veut dire que l'ordinateur peut prendre des décisions, non pas grâce à une suite de if et else. Mais d'apprendre grâce à son environnement. Avant de rentrer dans les détails des différents types de machine learning, certains lecteurs peuvent déjà se poser la question sur ce que j'appels l'environnement de l'ordinateur. J'entends par là un Jeu de données fourni à l'ordinateur, il sera capable de prédire certaines informations sur ce jeu grâce à son contenu. Comment-fait on cela et quelles sont ces informations. Eh bien il existe deux grandes familles d'apprentissage et une plus petite. Les deux principales sont **L'apprentissage Supervisé** (ou Supervised Learning) et **L'apprentissage non Supervisé** (ou Unsupervised Learning). La troisième étant **L'apprentissage par renforcement** (ou Reinforcement Learning).

Je vais maintenant vous décrire rapidement leurs différences avant de rentrer dans les details plus techniques de leurs différentes méthodes dans la section suivante.

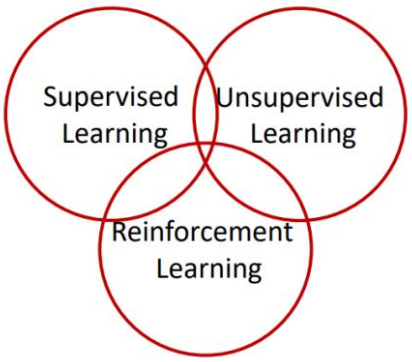


Figure 1: Les grandes familles d'apprentissage

L'apprentissage Supervisé

Les jeux de données des algorithme d'apprentissage supervisé sont en deux parties. Les caractéristiques ou variable prédictive, elles sont souvent notée X. Et les variables à prédire, notée Y. Un algorithme d'apprentissage supervisé va trouver ou utiliser une fonction mathématique afin de créer un lien entre X et Y. Cette fonction est appelée **modèle de prédiction**.

La variable à prédire appartient à une des deux catégories :

- **La classification** : dans le cas où Y est une variable discrète. C'est à dire qu'elle appartient à une classe. Par exemple à partir de X trouver si Y est un Homme ou une Femme.
- **La Regression** : dans le cas où Y est une variable continue. Elle peut prendre n'importe quelle valeur. Trouver la taille Y de l'individu à partir de ses données X.

L'apprentissage non Supervisé

L'apprentissage non supervisé va quant à lui possèder un jeu de données uni. C'est à dire qu'il n'y a pas de données X ou Y. Il y a seulement des données et il va essayer de trouver une structure à l'intérieur pour classer les données. Par exemple regrouper des individus par langue parler et dans la majorité des cas ils seront aussi regroupés par pays.

L'apprentissage par renforcement

L'apprentissage par renforcement quant à lui, moins vulgarisé que les autres, consiste à améliorer l'apprentissage basé sur les retours des précédentes expériences. On aura alors besoin d'un jeu d'apprentissage où les résultats attendus seront déjà connus. C'est le principe de base d'un réseau de neurone.

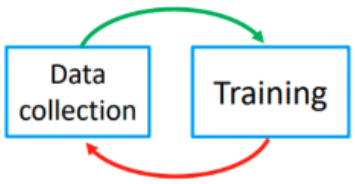


Figure 2: Apprentissage par renforcement

Quel est son but ? Les différentes méthodes

Nous n'allons pas rentrer dans les détails des différentes méthodes, le but n'étant pas de faire un cours dessus. De plus les détails seront expliqués plus en profondeur dans différents chapitres. Le but est plus de résumer un peu l'objectif de chaque méthode avec un petit exemple.

L'apprentissage Supervisé

Le but de l'apprentissage supervisé, comme expliqué précédement, est de prédire une variable en fonction d'autres variables. De plus on connait déjà le résultat pour une partie des données, on va utiliser ce résultat pour lui apprendre. D'où le terme supervisé.

La classification : Dans le cas de la classification, il faut classer les variables dans différentes catégories, dans différentes classes.

Dans le cas suivant, le jeu de données est constitué de 3 variables. 2 variables numériques et 1 de classes à deux niveaux, *traité* et *non traité*. Dans l'exemple suivant nous allons essayer de prédire le résultat traité ou non en fonction des données numériques.

```
data("Puromycin")
puro <- Puromycin

set.seed(12345)
choix <- sample(2, nrow(puro), replace=TRUE, prob= c(0.67, 0.33))

puro['choix'] <- choix
train <- puro[choix == 1, 1:3]
test <- puro[choix == 2, 1:3]
```

On a séparé le jeu en deux parties, une pour l'entrainement et l'autre pour tester notre modèle.

```
model <- glm(state~., family = "binomial", data=train)
```

On entraine le modèle

```
resultat <- predict.glm(model, test, type="response")

resultat <- ifelse(resultat>0.5, "treated", "untreated")
resultat <- as.factor(resultat)
confusionMatrix(resultat,test$state)
```

```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction  treated untreated
##   treated         4         3
##   untreated       2         0
##
##                Accuracy : 0.4444
##                  95% CI : (0.137, 0.788)
##     No Information Rate : 0.6667
##     P-Value [Acc > NIR] : 0.9576
```

```
##
##                  Kappa : -0.3636
##
##  Mcnemar's Test P-Value : 1.0000
##
##             Sensitivity : 0.6667
##             Specificity : 0.0000
##          Pos Pred Value : 0.5714
##          Neg Pred Value : 0.0000
##              Prevalence : 0.6667
##          Detection Rate : 0.4444
##    Detection Prevalence : 0.7778
##       Balanced Accuracy : 0.3333
##
##        'Positive' Class : treated
##
```

Le résultat est plus que moyen. Il peut avoir plusieurs raison, un mauvais modèle par exemple. Ici une des raisons est sans doute la taille du jeu qui ne fait que 23 lignes. De plus beaucoup de probabilitées on été ajustées par défaut à 1 par le modèle.

Mais l'objectif d'un petit exemple clair a été atteint.

La Regression : La regression permet d'affecter une valeur à une donnéee. Par exemple à partir du poids et de l'âge d'une personne, on peut trouver un modèle qui donne sa taille. Et bien en rentrant une nouvelle personne avec son poids et son âge, le modèle sera capable de donner sa taille, on se basant sur les corrélations précédentes.

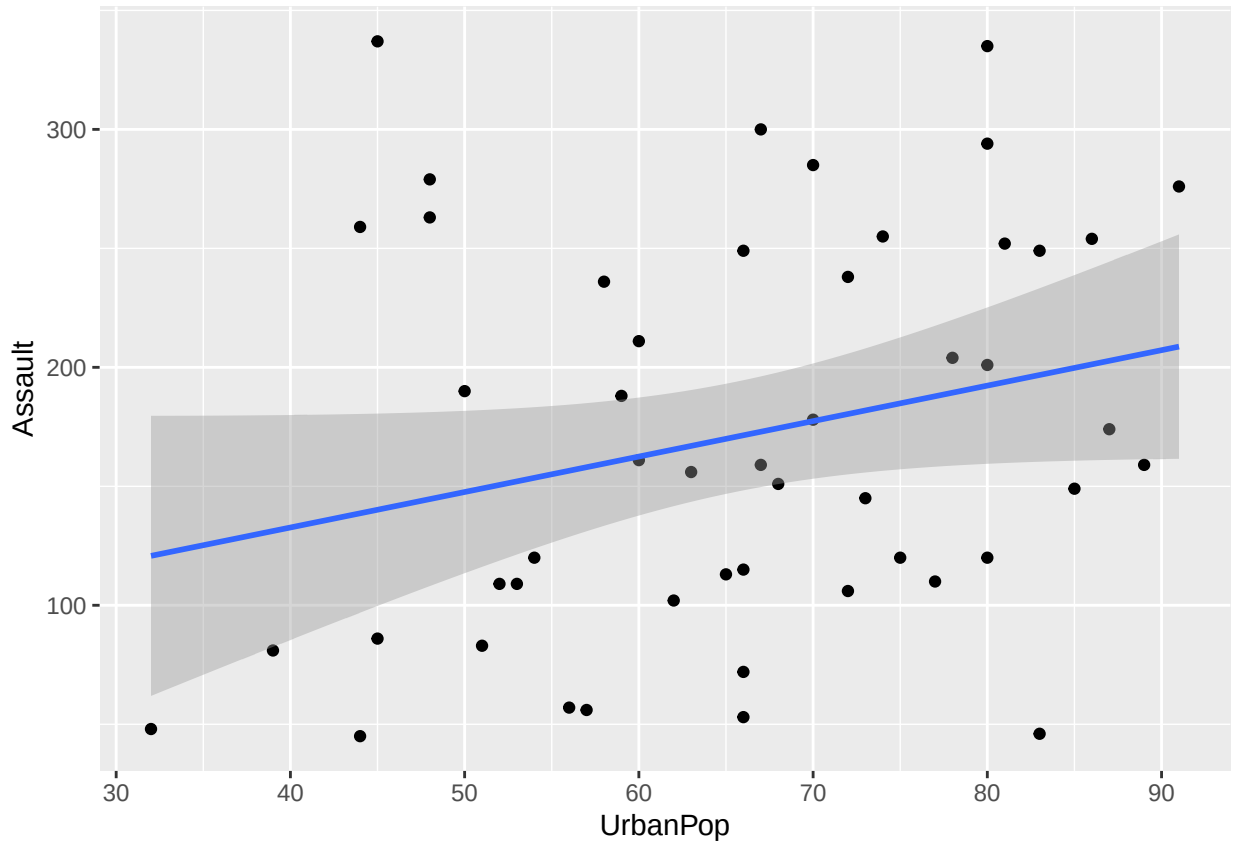
Voici un petit exemple avec les arrestations en fonction de la population.

```
data("USArrests")
USArr <- USArrests
model <- lm(Assault ~ UrbanPop,data = USArr )
model
```

```
##
## Call:
## lm(formula = Assault ~ UrbanPop, data = USArr)
##
## Coefficients:
## (Intercept)     UrbanPop
##       73.08         1.49
```

Le modèle donne automatiquement le nombre d'arrestation en fonction de la population.

```
## `geom_smooth()` using formula 'y ~ x'
```



Evidement on peut voir que le modèle n'est pas du tout précis. Si on avait voulu un modèle précis il aurait fallu que chaque points soient sur la ligne ou au moins dans la zone grise. On optient ce résultat tout simplement parce qu'il est utopiste de prévoir le nombre d'arrestations en fonction du nombre d'habitants. Cela dépends de beaucoup plus de choses, la richesse de la ville, la présence de la police, etc ...

L'apprentissage non Supervisé

Le but est de faire ressortir une structure dans les données. Trouver automatiquement un agencement logique.

Pour cet exemple nous allons voir la méthode la plus utilisée, le **clustering**. Et pour faire ce "clustering" nous allons utiliser l'algortihme des K-moyennes. Le principe est de regrouper les données et donner un centre à ces groupes. Chaque nouvelle donnée devant être classer sera attribuée au groupe ayant le centre le plus proche d'elle.

On va réutiliser l'exemple USArrests pour voir si certaines villes présentent des caractéristiques similaire et si on peut les regrouper par cluster.

```
data("USArrests")
USArr <- scale(x = USArrests)
set.seed(123)
```

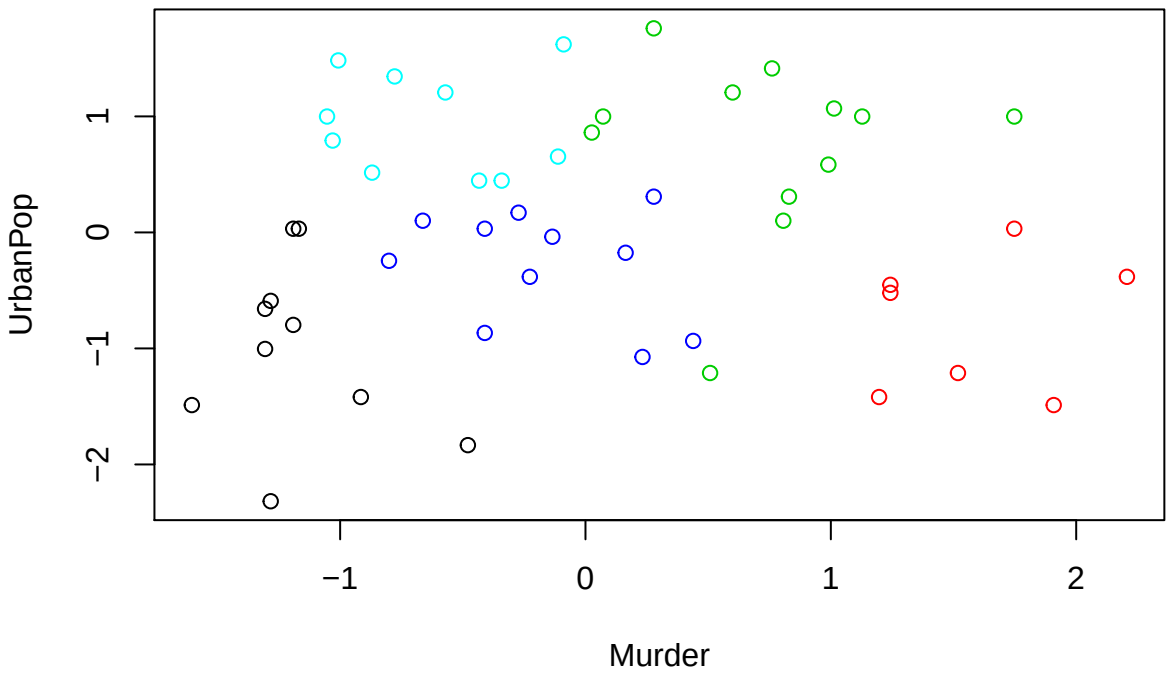
Ici on lancer la fonction de K-Means, avec k le nombre de cluster désiré.

```
k = 5
resultat = kmeans(USArr, k , nstart = 30)
resultat
```

```
## K-means clustering with 5 clusters of sizes 10, 7, 12, 11, 10
##
## Cluster means:
##       Murder    Assault   UrbanPop        Rape
## 1 -1.1727674 -1.2078573 -1.0045069 -1.10202608
## 2  1.5803956  0.9662584 -0.7775109  0.04844071
## 3  0.7298036  1.1188219  0.7571799  1.32135653
## 4 -0.1642225 -0.3658283 -0.2822467 -0.11697538
## 5 -0.6286291 -0.4086988  0.9506200 -0.38883734
##
## Clustering vector:
##        Alabama         Alaska        Arizona       Arkansas     California
##              2              3              3              4              3
##       Colorado    Connecticut       Delaware        Florida        Georgia
##              3              5              5              3              2
##         Hawaii          Idaho       Illinois        Indiana           Iowa
##              5              1              3              4              1
##         Kansas       Kentucky      Louisiana          Maine       Maryland
##              4              4              2              1              3
##  Massachusetts       Michigan      Minnesota    Mississippi       Missouri
##              5              3              1              2              4
##        Montana       Nebraska         Nevada  New Hampshire     New Jersey
##              4              4              3              1              5
##     New Mexico       New York North Carolina   North Dakota           Ohio
##              3              3              2              1              5
##       Oklahoma         Oregon   Pennsylvania   Rhode Island South Carolina
##              4              4              5              5              2
##   South Dakota      Tennessee          Texas           Utah        Vermont
##              1              2              3              5              1
##       Virginia     Washington  West Virginia      Wisconsin        Wyoming
##              4              5              1              1              4
##
## Within cluster sum of squares by cluster:
## [1]  7.443899  6.128432 18.257332  7.788275  9.326266
##  (between_SS / total_SS =  75.0 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

On voit bien dans le résultat nos 5 clusters avec la moyenne de leur centre. On peut aussi voir l'appartenance de chaque ville avec leur cluster.

```
plot(USArr[,c(1,3)], col = resultat$cluster )
```



Si on affiche les villes avec les meurtres en fonction de la population, on peut voir apparaitre des clusters.

Quelle est la méthode la plus utilisée ?

La méthode la plus utilisée serai surement la regressions linéaire. C'est la plus simple d'utilisation et de compréhension. De plus avec un modèle bien ajusté on obtient rapidement des résultats précis.

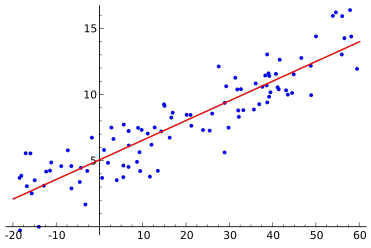


Figure 3: Linear Regression

Conclusion

Voilà ce qui conclue les bases sur le machine learning. J'éspère que vous aurez une bonne vision de départ sur ce qu'est le machine learning aujourd"hui. N'ésitez pas à aller lire les autres topics afin vous familiariser avec les concepts plus en profondeur.

Advisor : Fatim **THIAM** Elle nous a aidé à corriger les fautes d'orthographe. Elle nous a aussi dirigé sur le choix de graphique.