## Rapport de Projet

Par :

EL ALAMI Dina

Année universitaire : 2019-2020

# Table des matières

1. [Analyse de données II](#_bookmark0) 2
   1. [Modélisation avec méthodes factorielles et méthodes linéaires](#_bookmark1) 3
      1. [Le modèles linéaires généralisés (GLM)](#_bookmark2) 3
      2. [Colinéarité des variables explicatives](#_bookmark3) 5
      3. [Analyse en Composantes Principales (ACP) :](#_bookmark4) 7
      4. [Modélisation via GLM](#_bookmark5) 10
2. [Machine Learning/Optimisation Stochastique](#_bookmark6) 14
3. [Optimisation Stochastique :](#_bookmark1) 15
   1. [Optimisation Gradient Descente](#_bookmark2) 15
   2. [Optimisation gradient stochastique :](#_bookmark3) 17
4. [Machine Learning :](#_bookmark7) 17
   1. [K-Nearest Neighbors (K-NN) :](#_bookmark8) 17
   2. [Support Vector Machines (SVM) :](#_bookmark9) 18
   3. [Arbre de décision :](#_bookmark10) 19
   4. [Random Forrest](#_bookmark11) 20
   5. [Bagging :](#_bookmark12) 21
   6. [Gradient Boosting :](#_bookmark13) 22
   7. [Régression Logistique](#_bookmark14) 23

## Première partie

**Analyse de données II**

### 1 Modélisation avec méthodes factorielles et méthodes li- néaires

L’analyse linéaire de la régression, appelée aussi plus simplement régression linéaire, est l’une des méthodes statistiques les plus utilisées dans les sciences appliquées. Son objectif est double : il consiste tout d’abord à décrire les relations entre une variable privilégiée, appelée variable expliquée (ou dépendante), et plusieurs variables jouant un même rôle par rapport à la première, appelées variables explicatives (ou indépendantes). Elle per- met aussi d’effectuer des prévisions de la variable expliquée en fonction des variables explicatives.

Les liaisons entre les variables explicatives exercent une influence très importante sur l’ef- ficacité du modèle, parmi les conséquences de la colinéarité statistique entre les variables explicatives sont les suivantes :

* les coefficients de régression estimés peuvent être élevés en valeur absolue
* leurs signes peuvent être contraires à l’intuition
* les variances des estimateurs peuvent être élevées

#### Le modèles linéaires généralisés (GLM)

Les modèles linéaires gaussiens ont longtemps été utilisés pour modéliser des variables aléatoires afin de réaliser une étude . Cependant, ils ne sont pas adaptés à la réalité, puisque la variable à modéliser, c’est-à-dire la variable réponse, n’est pas nécessairement gaussienne. Ainsi, le modèle linéaire généralisé a été créé afin d’étendre le modèle linéaire aux variables non gaussiennes et plus précisément aux variables dont la loi fait partie de la famille exponentielle. L’objectif du modèle linéaire généralisé est de modéliser la relation existante entre une variable réponse et une ou plusieurs variables explicatives. Avant d’exposer le modèle linéaire généralisé, il est primordial de passer par le modèle linéaire gaussien.

##### Le modèle linéaire gaussien

Nous considérons *n* observations indépendantes *y*1*, y*2*, ..., yn* correspondant à des réali- sations de la variable réponse *Yi* . L’équation s’écrit sous la forme suivante :

*p*

Σ

*Yi* = *β*0 + *βjXij* + *si i* = 1*,* 2 *, n*

*j*=1

Où :

* *n* : Nombre d’observations,
* *p* : Nombre de variables explicatives
* *Xi*1*, ..., Xip* :variables explicatives associées à l’individu i
* *β*0*, ......βp* : paramètres inconnus à estimer

*si* : terme d’erreur provenant de la différence entre l’observation et l’estimation de la variable réponse . *si* est supposé de moyenne nulle et de variance constante

−

On dit que le modèle est «gaussien» parce que nous supposons que les erreurs sont dis- tribuées selon une loi normale d’espérance nulle et de variance constante inconnue *σ*2.

Le modèle linéaire généralisé se distingue du modèle linéaire gaussien par les trois com- posantes suivantes :

**la composante aléatoire :** Nous supposons que les observations yi sont indépen- dantes et associées à une loi de probabilité issue d’une structure exponentielle dont les lois disposant d’une fonction de densité pouvant s’écrire sous la forme suivante :

−

*f* (*y, θ, φ*) = exp . *yθ* − *b*(*θ*) − *c*(*y, φ*)Σ

*a*(*φ*)

L’espérance et la variance sont données par les formules suivantes :

**E**(*Y* ) = *db*(*θ*)

*dθ*

*V ar*(*Y* ) = *a*(*φ*)*bjj* (*θ*)

où le paramètre *θ* est appelé le paramètre de position et *φ* le paramètre de disper- sion, les fonctions *a*(*.*), *b*(*.*) et *c*(*., .*) sont des fonctions réelles.

**la composante systématique :** La composante systématique *ηi*, nommée pré- dicteur linéaire, correspond à une combinaison linéaire des variables explicatives. Soit *xij* les observations de la variable explicative *Xij* , nous avons :

−

*ηi* = *xtβ*

*i*

où :

*xt* = .1*, x*

*,* · · · *, , x*

*β*0

Σ *β* = *β*1

*i i*1

*it*  *..* 

*βp*

**la fonction de lien :** La relation entre la composante aléatoire et le prédicteur linéaire est exprimée par la troisième composante appelée fonction de lien g, stric- tement monotone et différentiable.Notons *µi* = **E**(*Yi*), alors

−

*g*(*µi*) = *ηi µi* = *g*−1(*η*) = *g*−1(*xtβ*)

*i*

Contrairement aux modèles linéaires simples et multiples, il s’agit ici de modéliser une transformation de l’espérance de la variable réponse.

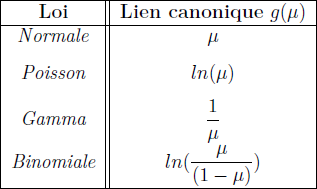


Figure 1 – Les fonctions de lien classiques

##### Modèle de la régression logistique

Il s’agit d’un modèle de régression binomiale avec le lien logit. La loi binomiale est na- turellement choisie pour la modélisation en raison du caractère binaire de la variable réponse. Nous choisissons ensuite la fonction logit comme fonction de lien car il s’agit du lien canonique associé au modèle de régression binomiale.

Le modèle la régression logistique qui se formalise comme suit :

*g*(*µi*) = ln

*µi*

1 − *µi*

. Σ

*η*

*ηi* ⇐⇒ *µi* = 1 + *eηi*

= *e i*

##### Modèle logistique multi-classes

Nous traitons dans ce chapitre le cas où la variable à expliquer *Y* prend plus de deux modalités. Pour simplifier les notations, nous supposerons que *Y* peut prendre *K* valeurs 1*, ..., K*. Nous cherchons donc à expliquer *Y* par le vecteur *X* = (1*, X*1*, ..., Xp*)*j* des variables explicatives (*Xj*) étant qualitatives, quantitatives ou pouvant représenter des interactions.

Tout comme dans le cas binaire, nous cherchons ici à modéliser la loi de *Y* |*X* = *x* . Si on suppose *Y* prend ses valeurs dans {1*, ..., K*}, le problème va consister à mettre une forme paramétrique sur les probabilités *πj* = **P**(*Y* = *j*|*X* = *x*)*, j* = 1*, ..., K.*

#### Colinéarité des variables explicatives

Avant de commencer la modélisation il faut extraire la matrice de corrélation de nos va- riables pour évaluer la dépendance entre plusieurs variables en même temps. Le résultat est une table contenant les coefficients de corrélation entre chaque variable et les autres. Pour cela nous utilisons le coefficient de Spearman, qui est associé à une approche non

paramétrique (sans hypothèse sur la loi de probabilité des variables) permettant de me- surer et tester si deux variables continues sont corrélées.

Le graphe ci-dessous montre la matrice de corrélation de Spearman associée à nos don- nées. Nous utilisons la Classification Hiérarchique Ascendante (CHA) avec la métrique de Ward pour agréger les clusters de variables qui ont des corrélations ensemble.

Les corrélations positives sont affichées en bleu et les corrélations négatives en rouge. L’intensité de la couleur et la taille des cercles sont proportionnelles aux coefficients de corrélation. A droite du corrélogramme, la légende de couleurs montre les coefficients de corrélation et les couleurs correspondantes.

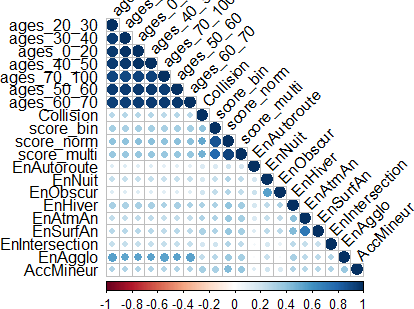


Figure 2 – Analyse de corrélation

Dès le premier coup d’œil, on remarque qu’a un problème avec les variables tranches âges qui ont fortement corrélé et ça se représente par gros triangle bleu de la population, ensuite on peut remarquer aussi la corrélation entre la variable « EnAgglo » qui décrit les accidents qui se produisent dans les villes représentent une grande densité d’habitants et les variables qui représentent la population. On remarque que les variables cibles sont corrélées positivement entre eux et légèrement avec les variables tranches âges et les autres variables. On remarque une corrélation positive entre la variable « EnSurfAn

» qui décrit infrastructures de la route et la variable « EnAtmAn» qui représente les circonstances atmosphériques.

#### Analyse en Composantes Principales (ACP) :

En général, pour remédier le problème de colinéarité des variables explicatives nous utilisons les techniques de l’analyse factorielle qui permet au praticien de réduire le nombre de variables et de rendre l’information moins redondante comme l’analyse en composantes principales (ACP) :

##### Le principe de l’ACP

L’analyse en composantes principales est une technique de statistique multidimension- nelle permettant d’analyser les liaisons entre plusieurs variables quantitatives simulta- nément. L’objectif est de passer d’un espace à *p* dimensions à un espace de dimension inférieur à *p* en perdant le moins possible d’informations du tableau de données initial.

Cette technique consiste à projeter orthogonalement le nuage des individus sur un plan factoriel, plan passant au plus près des individus du nuage. Cela permet de créer de nouvelles variables artificielles, nommées les « axes factoriels ».

Mathématiquement, il s’agit de diagonaliser une matrice de variance-covariance où les vecteurs propres (axes factoriels) et les valeurs propres (variances associées aux axes) sont extraits. De la même façon, les variables sont projetées orthogonalement sur un plan factoriel s’inscrivant dans un cercle de rayon unitaire appelé cercle de corrélation, sur lequel nous pouvons observer les corrélations entre les différentes variables.

##### Implémentation de ACP

Dans notre étude, nous appliquons une ACP sur le jeu d’entraînement, qui va engendrer une transformation des données selon les axes principaux qui expliquent le plus d’infor- mation, puis nous utilisons les axes principaux obtenus pour évaluer une prédiction sur le jeu de test, et ceci pour garder une indépendance entre la base train et la base test et éviter les fuites de données (data leakage).

Dans l’analyse en composantes principales, les variables sont souvent normalisées. Ceci est particulièrement recommandé lorsque les variables sont mesurées dans différentes unités, sinon le résultat de l’ACP obtenue sera fortement affecté, Techniquement, l’ap- proche consiste à transformer les données en soustrayant à chaque valeur une valeur de référence (la moyenne de la variable) et en la divisant par l’écart type. La standardisa- tion des données est une approche beaucoup utilisée dans le contexte de l’analyse des données d’expression de gènes avant les analyses de type PCA et de clustering.

La fonction PCA() dans la librairie « FactoMineR », normalise les données automati- quement pendant l’ACP en mettant « scale.unit = TRUE »

##### Sélection du nombre d’axes

Avant toute interprétation des représentations graphiques, il est nécessaire de choisir le nombre optimal d’axes factoriels, afin d’avoir un résumé précis de l’information du tableau de données initial. Malheureusement, il n’existe pas de méthode objective bien acceptée pour décider du nombre d’axes principaux qui suffisent, Une méthode proposée pour déterminer le nombre de composantes principales est de regarder le graphique des valeurs propres (appelé scree plot). Le nombre d’axes est déterminé par le point, au- delà duquel les valeurs propres restantes sont toutes relativement petites et de tailles comparable.

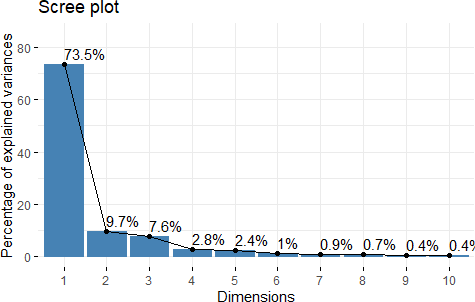


Figure 3 – Poucentage de l’information expliqué

D’après le graphe ci-dessus, nous pourrions vouloir nous arrêter à la cinquième compo- sante principale. 96% des informations contenues dans les données sont conservées par les cinq premières composantes principales.

##### Qualité de représentation des variables

La qualité de représentation des variables sur la carte de l’ACP s’appelle cos2. Un cos2 élevé indique une bonne représentation de la variable sur les axes principaux, par contre Un faible cos2 indique que la variable n’est pas parfaitement représentée par les axes principaux.

Nous pouvons visualiser le cos2 des variables sur toutes les dimensions en utilisant le package « corrplot ».

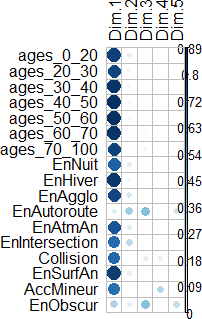


Figure 4 – Qualité de représentation

##### Corrélation des composantes principales

Avant de passer à l’étape de la modélisation, nous vérifions la corrélation entre les 5 les composantes principales :

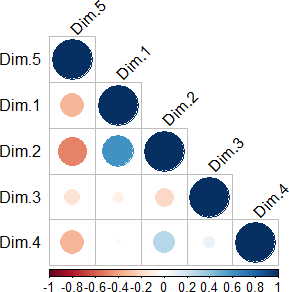


Figure 5 – Corrélation des composantes principales

Comme on peut le voir dans le graphe ci-dessus, les composantes principales sont moins corrélées par rapport aux variables initiales et ça c’est déjà pas mal.

#### Modélisation via GLM

##### Tests d’adéquation

Nous appliquons dans cette partie les modèles linéaires généralisés à la modélisation du variable score-norm. Cette variable étant continue et positive. Les distributions les plus utilisées dans ce cas sont la loi normale, gamma et log-normale. Il s’agit ainsi de choisir le modèle proposant le meilleur ajustement aux données étudiées.

Nous comparons la distribution empirique avec la fonction de densité des trois lois : normale, gamma et log-normale. Les paramètres des différentes distributions sont estimés par maximum de vraisemblance et indiqués sous le graphique ci-dessous :

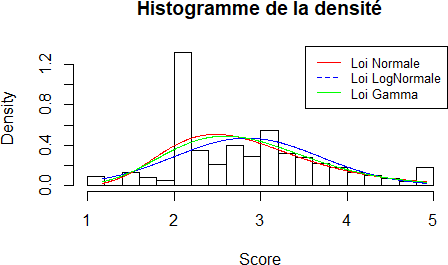


Figure 6 – Histogramme de la densité

L’analyse graphique des histogrammes n’est pas suffisante pour choisir un modèle per- mettant le meilleur ajustement aux données. Elle permet uniquement d’avoir un avis sur le type de distribution qui pourrait être utilisé. Cette analyse doit être complétée par un test d’adéquation.

Le test de *Kolmogorov-Smirnov* est un test qui cherche à vérifier l’adéquation suivante, sous l’hypothèse nulle :

*H*0 : *F* = *Fn*

Où *F* est la fonction de répartition théorique de la loi à tester et *Fn* la fonction de répartition empirique de nos données observées.

Le test s’appuie directement sur l’écart maximal entre la distribution empirique et la distribution de la loi testée. En effet, ce test mesure la valeur suivante :

*Kn* = √*n* sup *Fn*(*x*) *F* (*x*)

| − |

*x*∈R

##### Les critères de choix de modèle et la mesure de performance

Après avoir présenté le cadre théorique des GLM, nous proposons dans cette partie d’étudier les différents critères utilisés dans la sélection et la comparaison de modèles.

##### Déviance (D)

La déviance *D* est définie comme deux fois la différence entre la log-vraisemblance du modèle saturé et la log-vraisemblance du modèle étudié

*L*(*mod*è*le* é*tudi*é)

*D* = 2 log

*L*(*mod*è*le satur*é)

Le modèle avec la déviance la plus faible sera préféré aux autres modèles puisque ce critère indique un écart plus faible entre les log-vraisemblances, et ainsi une distance plus faible entre les valeurs modélisées et les valeurs observées.

##### La règle du pouce

Pour évaluer la pertinence du modèle de régression logistique, on préconise la règle du pouce telle que l’espérance d’une v.a.r suivant une loi du Chi-deux est égale à ses degrés de libertés. Par conséquent, si le modèle est pertinent, on doit avoir *D* proche de son espérance *ν* = *n* − (*p* + 1) , soit encore,

*D*

##### Critère d’Akaike AIC et BIC

*ν* c 1

Outre la déviance, les critères AIC (Akaike Information Criterion) et BIC (Bayesian Information Criterion) permettent également de comparer les modèles entre eux. L’utili- sation de ces critères semble plus appropriée dans la comparaison de modèles construits avec des distributions de variables réponses différentes.

L’AIC est défini par la formule suivante :

*AIC* = 2 log(*L*) + 2*k*

où log(*L*) constitue la log-vraisemblance maximisée et *k* le nombre de paramètres.

Le critère BIC doit être privilégié lorsqu’il s’agit de modèles disposant d’un grand nombre d’observations. En effet, dans la littérature il est précisé que le critère AIC a tendance à choisir les modèles avec de nombreuses variables explicatives dans le cas de grands échantillons. Afin d’écarter ce problème, le nombre de paramètres dans la formule du BIC est multiplié par le logarithme du nombre d’observations log(*n*) et permet ainsi d’appliquer une pénalité plus sévère afin de privilégier l’utilisation de modèles avec moins de variables explicatives :

*R*2 **et** *R*2

*adj*

##### ajusté

*BIC* = 2 log(*L*) + 2 log(*n*)

La justesse de l’estimation peut être mesurée par le coefficient de détermination :

*n*

Σ 2(ˆ − )

*yi yi*

*R*2 = 1 − *i*=1

*n*

Σ 2(*y* − *y*¯)

*i*

*i*=1

*R*2 ∈ [0*,* 1]

où *yi* est la vraie valeur de la variable cible *y* à l’indice *i*, *y*ˆ*i* la prédiction du modèle à l’indice *i*, et *y*¯ la moyenne empirique de la variable cible.

Un *R*2 proche de 1 indique que l’ajustement est de bonne qualité. En effet ce coefficient est le rapport de la somme des carrés expliquée à la somme des carrés totale. Un *R*2 proche de 1 équivaut donc à une somme des carrés expliquée proche de la somme des carrés totale et témoigne ainsi d’une perte minime d’information dans la modélisation.

Cet estimateur a cependant le défaut de tendre systématiquement vers 1 avec l’ajout de variables explicatives, on lui préférera donc le coefficient de détermination ajusté suivant

noté *R*2 , pénalisé par le nombre *p* de variables explicatives du modèle :

*adj*

Σ (*y*ˆ*i* − *yi*)2

*n*

= 1 −

*R*

2

*adj*

*n* − 1

*n* − *p* − 1

*i*=1

*n*

Σ 2( − ¯)

*yi y*

*i*=1

##### Variance Inflation Factor (VIF)

Les FIV mesurent l’accroissement de la variance d’un coefficient de régression estimé s’il existe une corrélation entre vos prédicteurs. Si tous les FIV sont égaux à 1, il n’existe pas de multicolinéarité, mais si certains FIV sont supérieurs à 1, les prédicteurs sont corrélés. Lorsqu’un FIV est supérieur à 5, le coefficient de régression de ce terme n’est pas correctement estimé.

##### La matrice de confusion

La matrice de confusion ou tableau de contingence est un outil permettant de mesurer les performances d’un modèle de Machine Learning en vérifiant notamment à quelle fréquence ses prédictions sont exactes par rapport à la réalité dans des problèmes de classification.

La matrice de confusion est un résumé des résultats de prédictions sur un problème de classification. Les prédictions correctes et incorrectes sont mises en lumière et réparties par classe. Les résultats sont ainsi comparés avec les valeurs réelles.

Pour bien comprendre le fonctionnement d’une matrice de confusion, il convient de bien comprendre les quatre terminologies principales : **TP**, **TN**, **FP** et **FN**. Voici la définition précise de chacun de ces termes :

TP (True Positives) : les cas où la prédiction est positive, et où la valeur réelle est effectivement positive

•

TN (True Negatives) : les cas où la prédiction est négative, et où la valeur réelle est effectivement négative.

•

FP (False Positive) : les cas où la prédiction est positive, mais où la valeur réelle est négative.

•

FN (False Negative) : les cas où la prédiction est négative, mais où la valeur réelle est positive.

•

## Deuxième partie

**Machine Learning/Optimisation Stochastique**

### Optimisation Stochastique :

#### Optimisation Gradient Descente :

Dans son cadre le plus général, l’optimisation d’une fonction permet de rechercher l’ensemble de paramètres permettant d’obtenir le meilleur résultat. Nous avons donc deux approches duales : la maximisation et la minimisation.

La recherche du minimum peut être faite de façon analytique, ou de façon numérique. Dans la grande majorité des cas réels, la fonction à optimiser n’est pas minimisable de façon analytique. On recourt donc à des méthodes numériques, et plus particulièrement l’échantillonnage et la descente de gradient. La première de ces méthodes consiste à éva- luer la valeur de la fonction pour chacun des jeux de paramètres possibles. La seconde se base sur l’amélioration d’une solution approchée.

La minimisation par échantillonnage (balayage) assure que le minimum de la fonc- tion sera trouvé. De plus, la recherche est faite en un temps constant, qui ne dépend que du nombre de jeux de paramètres à tester. La minimisation par descente de gradient permet quant à elle de ne pas tester tous les jeux de paramètres. L’amélioration pro- gressive d’une solution partielle permet en effet de concentrer ses efforts aux environs de la solution recherchée. Néanmoins, cette seconde approche se heurte aux problèmes des minima locaux. En effet, rien ne garantit qu’il n’existe pas d’autre minimum local, ni que sa valeur soit supérieure à celle du minimum trouvé.

Afin de minimiser une fonction à partir d’une solution approchée, le plus simple est de suivre la ligne de plus grande pente. D’un point de vue mathématique, la pente d’une fonction correspond à la dérivée de cette dernière. Si l’on se place dans le cadre d’une fonction ayant plusieurs paramètres, la dérivée devient un vecteur : "Le gradient de la fonction". Chaque élément de ce vecteur correspond alors à la dérivée partielle de la fonction selon l’un de ses paramètres.

##### Limite de l’optimisation Gradient Descente :

L’une des limites majeures de l’optimisation de gradient descente on trouve le pro- blème des minima locaux. En effet, même si l’on considère un cas simple, comme celui de la figure suivante, il arrive fréquemment que le minimum global soit masqué par un maxi- mum, local lui aussi. Dans cet exemple, en partant du point marqué d’une croix, il est impossible d’aller au-delà du point marqué d’un cercle. En effet, toute variation du pa- ramètre implique une augmentation de la valeur, et donc une dégradation de la solution.

La seconde limite de cette méthode tient à la notion de paramètre borné. En effet, il arrive souvent que certains paramètres ne soient valides que dans un certain intervalle. Or la descente de gradient peut amener à dépasser ces limites si l’on n’y prend pas garde. Ainsi, l’image ci-dessus montre une pente importante au niveau de la borne inférieure.

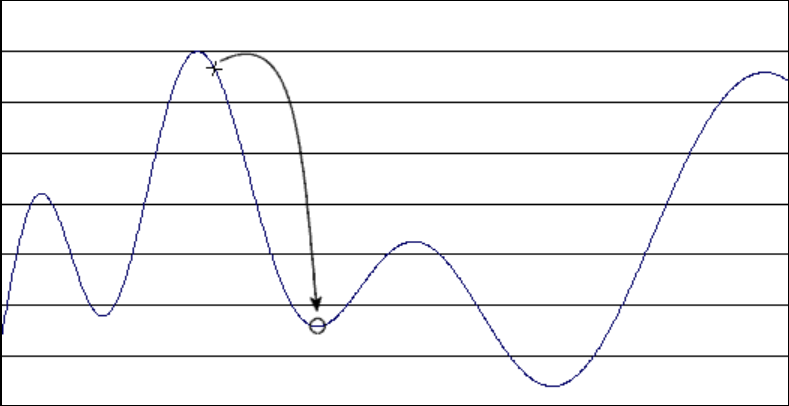


Figure 7 – Limite du Gradient descendant

Elle semble donc indiquer la présence d’un minimum intéressant en dessous de cette borne. Il faut donc ajouter aux algorithmes classiques des gardes permettant de garantir la validité des paramètres.

Finalement, afin de suivre la ligne de plus grande pente, il est nécessaire d’évaluer cette pente. Si la fonction à optimiser est dérivable, il suffit de calculer sa dérivée au niveau de la solution approchée. Cependant, toutes les fonctions ne sont pas dérivables. En particulier, il peut être intéressant d’optimiser des fonctions dont la formule analy- tique est inconnue.

##### Utilisation en Machine Learning :

En Machine Learning, une grande majorité se base sur l’optimisation des fonction de coût, qui est à minimiser. Cette fonction représente la somme du coût de l’erreur. Par exemple, pour une régression linéaire, celle-ci s’écrira de la façon suivante :

*J* (*α*) =

Σ(*yi* − (*α*0 + Σ *αjxij*))2

1

Avec :

*N*

*N i*=1

*k*

*j*=1

* + - *xij* l’observation *i* à la variable *j*.
    - *yi* la variable cible à l’observation *i*.
    - *α* = (*αj*)*j* les paramètres de la régression linéaire.

L’application de la descente du gradient se fait simplement par le calcul des dérivées partielles de la fonction coût, et de leur application itérative pour mettre à jour chacun des paramètres jusqu’à convergence.

#### Optimisation gradient stochastique :

La descente en gradient est une méthode d’optimisation pour trouver le minimum d’une fonction. Elle est couramment utilisée dans les modèles d’apprentissage profond pour mettre à jour les poids d’un réseau neuronal par rétropropagation.

Dans notre algorithme de descente de gradient, nous avons fait les gradients de chaque observation une par une. Dans la descente de gradient stochastique, nous pouvons choisir les observations aléatoires de manière aléatoire. On l’appelle stochastique car les échan- tillons sont sélectionnés de manière aléatoire (ou mélangés) au lieu de les regrouper en un seul groupe (comme dans la descente à gradient standard) ou dans l’ordre dans lequel ils apparaissent dans l’ensemble d’apprentissage.

Bien que l’utilisation de l’ensemble des données soit vraiment utile pour atteindre les minima de manière moins bruyante ou moins aléatoire, le problème se pose lorsque nos ensembles de données deviennent vraiment énormes. Supposons que notre jeu de don- nées contient un million d’observations. Par conséquent, si vous utilisez une technique d’optimisation typique de descente de gradient, vous devrez utiliser la totalité du million d’échantillons pour effectuer une itération lors de l’exécution de la descente de gradient pour chaque itération jusqu’à ce que les minima soient atteints. Par conséquent, cela devient très coûteux en terme de calcul.

Ce problème est résolu par une descente de gradient stochastique. Dans SGD, un seul échantillon est utilisé, c’est-à-dire une taille de lot de un, pour effectuer chaque itération. L’échantillon est mélangé au hasard et sélectionné pour effectuer l’itération.

La mise-à-jour des paramètres se fait comme suit :

*θj* = *θj* − *η*(*predictions*(*i*) − *y*(*i*))*X*(*i*)

*j*

### Machine Learning :

#### K-Nearest Neighbors (K-NN) :

Le K-Nearest Neighbors ou le K-NN, est un algorithme d’apprentissage supervisé. Il sert aussi bien pour la classification que la régression. Le K-NN est un algorithme qui se base sur le jeu de données pour donner un résultat, c’est à dire que le K-NN n’a pas besoin de créer un modèle prédictif.

Le K-NN effectue une prédiction en se basant entièrement sur le jeu de données. Par exemple, pour une observation qui ne fait pas partie de notre jeu de données, on va chercher les K instances du jeu de données les plus proches de notre observation. Ensuite pour ces K voisins, l’algorithme se basera sur leurs variables de sortie (Dans notre cas : score\_bin) pour calculer la valeur de la variable score\_bin de l’observation qu’on souhaite prédire.

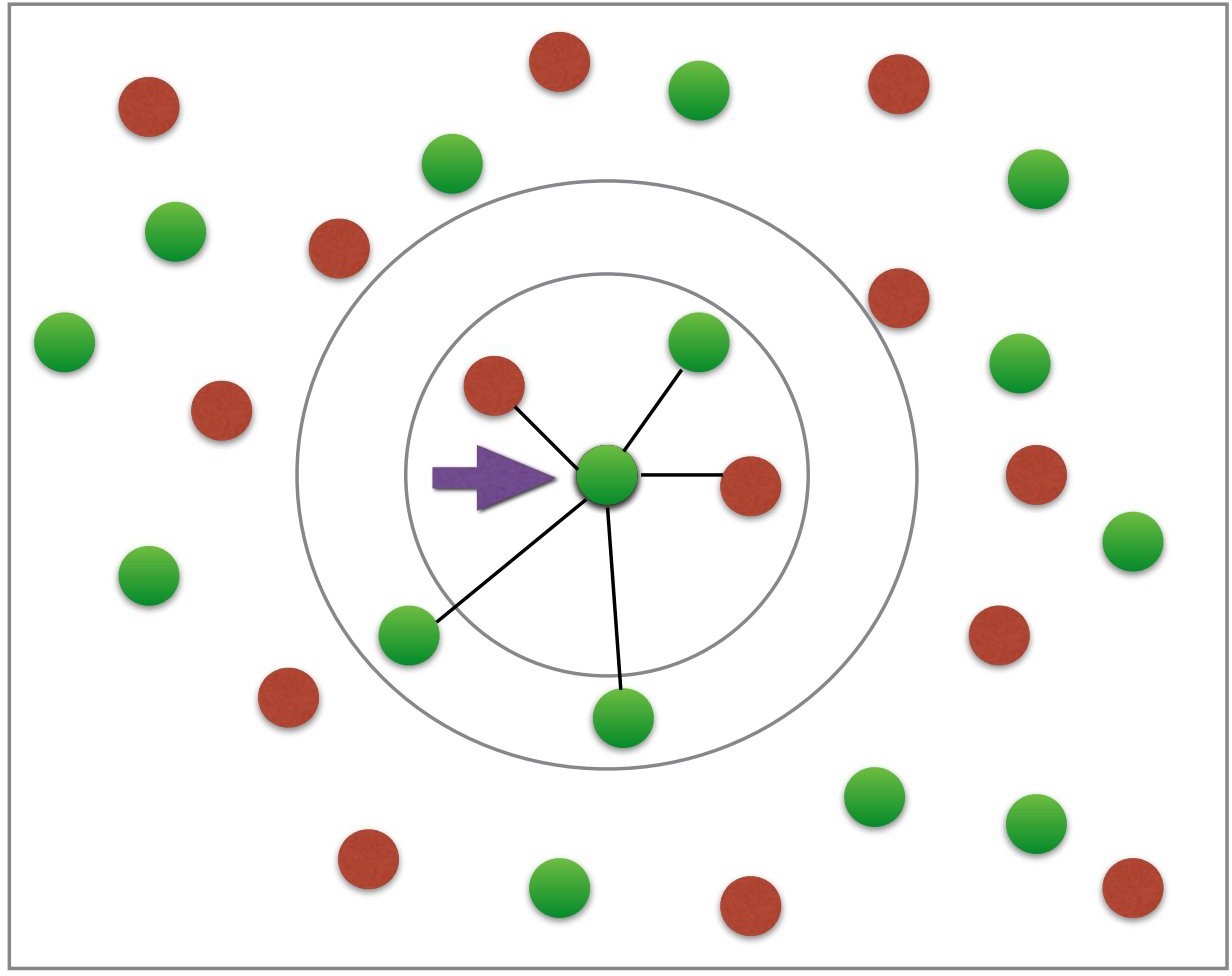


Figure 8 – K-NN : Fonctionnement

Généralement, l’algorithme K-NN s’enchaîne de la même façon. Pour une nouvelle observation dont on veut prédire sa variable de sortie, on calculera toutes les distances de cette observation avec les autres observations du jeu de données. Ensuite, on retient les K observations les plus proches de notre observation à prédire en utilisant la fonction de calcul de distance. Puis, on calculera le mode des variables de sorties des K observations retenues.

Pour procéder à la programmation de l’algorithme du K-NN, plusieurs étapes s’im- posent. Donc après avoir importer les données, on les divise en deux parties, une partie entrainement (train\_set : 75%), et une deuxième partie pour tester (test\_set : 25%). Ensuite on enchaîne à normaliser les données des variables explivatives, et cela en créant une fonction normalizer(X) qui retranche la moyenne de la variable explicative *Xi*, et puis divise sur son écart-type. Ensuite pour avoir une borne maximale du parmètre K, on calcule la racine carré du nombre d’observation du train, qui vaut approximativement *K* = 89 dans notre cas, et retourne une accuracy de 71% pour cette valeur. Après, on effectue une boucle avec un nombre d’itération inférieur à *K* = 89, en prenant comme résultat le numéro d’itération avec la plus grande valeur d’accuracy.

#### Support Vector Machines (SVM) :

Les machines à vecteurs de support ou séparateurs à vaste marge (Support Vector Machine, SVM) sont un ensemble de techniques d’apprentissage supervisé, destinées à résoudre des problèmes de discrimination et de régression.

Contrairement à la majotité des algorithmes, le SVM utilise un hyperplan qui agit comme une frontière de décision entre les différentes classes. Le SVM peut être uti- lisé pour générer plusieurs hyperplans de séparation de telle sorte que les données sont divisées en segments et que chaque segment ne contient qu’un seul type de données.

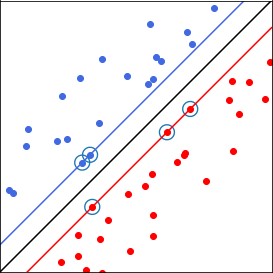
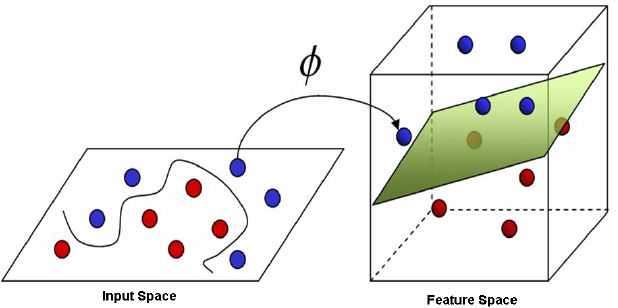


Figure 9 – Hyperplan de séparation SVM

Cette notion de frontière suppose que les données soient linéairement séparables, ce qui est rarement le cas. Pour y pallier, les SVM reposent souvent sur l’utilisation de noyaux. On appelle ce type de SVM : "SVM Radial", et le fonctionnement de ses fonctions mathématiques permet de séparer les données en ajoutant une dimension, c’est à dire en les projetant dans un espace vectoriel de plus grande dimension. La technique de maximisation de marge permet, quant à elle, de garantir une meilleure robustesse face au bruit, et donc un modèle plus généralisable.



#### Arbre de décision :

Figure 10 – Caption

Les arbres de décision font partie de la catégorie des algorithmes supervisés, ils per- mettent de prédire une valeur ou une catégorie. Dans le cas de classification, l’algorithme permet d’affecter chaque individu à une catégorie à l’aide de règles de décision.

Cet outil d’aide à la décision ou d’exploration de données permet de représenter un ensemble de choix sous la forme graphique d’un arbre. C’est une des méthodes d’ap- prentissage supervisé les plus populaires pour les problèmes de classification de données. Pour la classification, les arbres de classification permettent de prédire à quelle classe la variable de sortie appartient.

Les décisions possibles sont situées aux extrémités des branches (les « feuilles » de

l’arbre) et sont atteintes en fonction de décisions prises à chaque étape. Un arbre de décision fonctionne en appliquant de manière itérative des règles logiques très simples, chaque règle étant choisie en fonction du résultat de la règle précédente.Plus l’arbre est profond, et plus il peut apprendre des règles complexes, mais plus il perd en sa capacité à généraliser. Les règles sont construites en choisissant à chaque branche les séparations qui expliquent le plus la variance des données.

Les arbres de décision ont pour avantage d’être simple à interpréter, très rapide à entraîner, d’être non paramétrique, et de nécessiter très peu de pré-traitement des données. Ils peuvent être calculés automatiquement par des algorithmes d’apprentissage supervisé capables de sélectionner automatiquement les variables discriminantes au sein de données non-structurées et potentiellement volumineuses. Ces algorithmes permettent aussi d’extraire des règles logiques qui n’apparaissaient pas dans les données brutes.

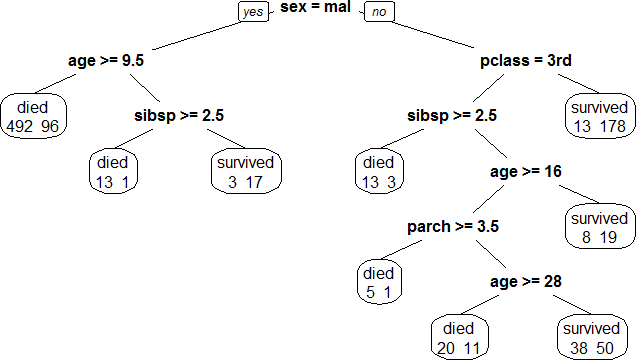


Figure 11 – Exemple d’arbre de décision

Parmi les extensions de la méthode en usage Machine Learning consiste à non seule- ment créer un arbre, mais une forêt d’arbres de décision. Les résultats ainsi obtenus sont remarquables notamment lorsque les arbres de décision sont utilisés en forêts aléatoires Random Forest.

#### Random Forrest :

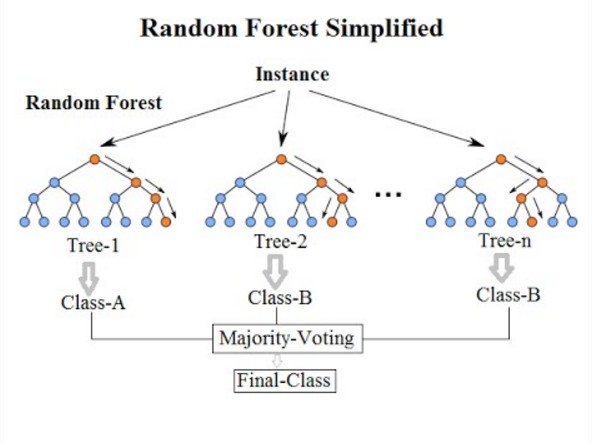
L’algorithme des « forêts aléatoires » (ou Random Forest) est un algorithme de clas- sification qui réduit la variance des prévisions d’un arbre de décision seul, améliorant ainsi leurs performances. Pour cela, il combine de nombreux arbres de décisions dans une approche de type bagging.

Dans sa formule la plus classique, le Random Forest effectue un apprentissage en

parallèle sur de multiples arbres de décision construits aléatoirement et entraînés sur des sous-ensembles de données différents. Toutefois, le nombre idéal des arbres est un paramètre très important, qui peut aller jusqu’à plusieurs centaines voire plus, c’est un paramètre très variable et dépend du problème. Concrètement, chaque arbre de la forêt aléatoire est entraîné sur un sous ensemble aléatoire de données selon le principe du bagging, avec un sous ensemble aléatoire de features, selon le principe des projections aléatoires. L’algorithme des forêts aléatoires est connu pour être un des classifieurs les plus efficaces « out-of-the-box » (c’est-à-dire nécessitant peu de pré-traitement des don- nées).

L’avantage principal du Random Forest est que grâce à un grand nombre d’arbres, la réduction de variance est souvent significative et donc finit par augmenter la perfor- mance du modèle sur le jeu de test.

L’inconvénient principal est dans le cas d’une très grande forêt qui prennent un plus grand sous-ensemble des variables explicatives et des observations, on va augmenter tant le temps d’exécution et le sur-apprentissage.



#### Bagging :

Figure 12 – Caption

Le Bagging est une méthode puissante pour améliorer les performances des mo- dèles simples et réduire le "Overfitting" des modèles plus complexes. Le principe est très simple à comprendre, au lieu d’ajuster le modèle sur un échantillon de la population, plusieurs modèles sont ajustés sur différents échantillons de la population, ces nouveaux échantillons sont sélectionnés par tirage au hasard dans l’ancien échantillon, avec remise.

Ensuite, ces modèles sont agrégés en utilisant leur moyenne, leur moyenne pondérée ou un système de vote dans la cas d’une classification.

#### Gradient Boosting :

Les Gradient Boosting Machines (GBM), sont un algorithme d’apprentissage extrê- mement populaire qui a fait ses preuves dans de nombreux domaines et est l’une des principales méthodes pour gagner des concours Kaggle. Alors que les Random Forest construisent un ensemble d’arbres indépendants profonds, les GBMs construisent un en- semble d’arbres successifs peu profonds et faibles, chaque arbre apprenant et améliorant le précédent. Lorsqu’ils sont combinés, ces nombreux arbres successifs faibles produisent un ensemble puissant qui est souvent difficile à battre avec d’autres algorithmes.

##### Avantages :

* + - Offre souvent une précision prédictive imbattable.

Beaucoup de flexibilité : peut optimiser sur différentes fonctions de perte et offre plusieurs options de réglage des hyperparamètres qui rendent l’ajustement de la fonction très souple.

•

Aucun prétraitement des données requis - fonctionne souvent très bien avec des valeurs catégorielles et numériques telles quelles.

•

* + - Traitement des données manquantes - imputation non nécessaire.

##### Inconvénients :

Les GBM continueront à s’améliorer pour minimiser toutes les erreurs. Cela peut accentuer les valeurs aberrantes et provoquer un sur-ajustement.

•

Coûteux en termes de calcul - les GBM nécessitent souvent de nombreux arbres (>1000) qui peuvent être exhaustifs en termes de temps et de mémoire.

•

La grande flexibilité se traduit par de nombreux paramètres qui interagissent et influencent fortement le comportement de l’approche (nombre d’itérations, pro- fondeur de l’arbre, paramètres de régularisation, etc.) ). Cela nécessite une grande recherche de grille pendant la régularisation.

•

Moins interprétable, bien que ce problème soit facilement résolu avec divers outils (importance des variables, tracés de dépendance partielle, LIME, etc.)

•

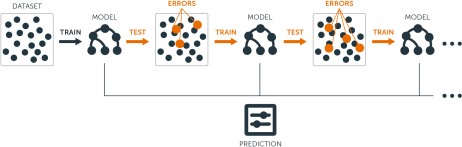


Figure 13 – Approche d’ensemble séquentielle

#### Régression Logistique :

La régression logistique est une manière de faire de la classification automatique par Machine Learning. Il s’agit de déterminer comment un ensemble de propriétés numé- riques (des nombres) peuvent être associées à une des classes considérées. Dans le cas le plus simple, on ne considère que deux classes : vrai ou faux, 1 ou 0, etc. Il existe des extensions de ce cas pour classifier en plus de deux classes.

Contrairement à ce que son nom laisse entendre, une régression logistique est donc bien utilisée en classification (trouver la bonne étiquette) et non en régression (trouver la bonne valeur numérique).

Si on ne considère qu’un seul paramètre réel *x*, la régression logistique consiste à trouver les meilleurs valeurs réelles de *β*0 et *β*1 dans l’équation :

1

*F* (*x*) =

1 + exp−(*β*0+*β*1*x*)

Tel que *F* (*x*) soit proche de 1 lorsque l’objet considéré doit avoir la classe “1”, et que

*F* (*x*) soit proche de 0 lorsque l’objet considéré doit avoir la classe “0”.

Donc, dès l’obtention des valeur de *β*0 et *β*1, il suffit, pour un nouvel objet, de déterminer sa valeur de *x*, d’évaluer *F* (*x*), et d’en déduire s’il doit être classé “0” ou “1”.

# Bibliographie

1. Gradient Boosting Machines - UC Business Analytics R Programming Guide.
2. Gradient Boosting Classification with GBM in R - DataTechNotes.
3. Compute Gradient Descent of a Multivariate Linear Regression Model in R - My DataScience Notebook.
4. Sélection de modèle en régression linéaire - univ Toulouse.
5. Linear Regression Tutorial Using Gradient Descent for Machine Learning - Machine Learning Mastery.