Date	14.06.2017
Auteur	Didier LEPICAUT
Email	Didier.le_picaut@insa_rouen.fr
Période	Période 4
Unité Enseignement	THESE PROFESSIONNELLE – INSA - Bruno PORTIER
Objet	Contrat de Professionnalisation à la MATMUT/SCD - Valérie BRIVET
Version	2.0
Date Version	13.10.2017

Problème : le rôle des Machines à Vecteur de Support pour prédire la transformation d'un devis commercial en contrat d'assurance automobile ?

DOCUMENT STRICTEMENT CONFIDENTIEL

LORS DE LA REALISATION DE CETTE THESE PROFESSIONNELLE,
LA PARTIE APPLICATION NUMERIQUE A ETE REALISEE EN TANT QUE SALARIE DE LA MATMUT,
DURANT MON CONTRAT DE PROFESSIONNALISATION, DU 07 NOVEMBRE 2016 AU 27 OCTOBRE 2017,
ET A PARTIR DE DONNEES DONT LA PROPRIETE ET L'UTILISATION SONT STRICTEMENT ENCADREES
PAR LES OBLIGATIONS LIEES A MON CONTRAT DE TRAVAIL.

EN CONSEQUENCE, IL EST DEMANDE PAR LA MATMUT LA PLUS STRICTE CONFIDENTIALITE
SUR L'UTILISATION DE CES DONNEES ET L'EXPLOITATION DES RESULTATS PRODUITS, A DIDIER LE PICAUT,
A TOUTE PERSONNE ET PARTENAIRE DE L'INSA ROUEN QUI PEUVENT AVOIR ACCES A CE DOCUMENT.

© TOUTE REPRODUCTION, MEME PARTIELLE,
TOUT TRANSFERT A UN TIERS SOUS QUELQUE FORME QUE CE SOIT,
SONT STRICTEMENT INTERDITS SANS AUTORISATION EXPRESSE ET ECRITE DE LA MATMUT.

« THÈSE PROFESSIONNELLE POUR L'OBTENTION DU MASTÈRE SPECIALISE EXPERT EN SCIENCE DES DONNEES »

Auteur : Didier LE PICAUT

Encadré par : Bruno PORTIER

En qualité de :
. Professeur 26ème Section Spécialité Statistiques,
. Responsable du Mastère Spécialisé en alternance Expert en Sciences des données.

****

SOMMAIRE

1. Remerciements 5

2. Mode d'emploi du document 6

3. Résumé 7

4. Introduction 8

5. Le contexte et le problème 9
5.1. La Matmut : le contexte d'affaires 9
5.2. Le cadre de la thèse professionnelle 9
5.2.1. Le problème de la transformation 9
5.2.2. La formalisation du problème mathématique 11
5.2.3. Etat des lieux de modèles prédictifs utilisés à la MATMUT 13
5.2.4. Présentation de la méthode et traitements des calculs numériques 14

6. La méthode des SVM : « les Machines à Vecteur de Support ou Séparateurs à vaste Marge » 15
6.1. Les Fondements Mathématiques 15
6.1.1. Présentation résumée de Yann LeCun 15
6.1.2. Démonstration mathématique 16
6.1.2.1. Le SVM linéaire : 16
6.1.2.1.1. Principe général : 16
Classement binaire : discrimination linéaire et hyperplan séparateur 16
6.1.2.2. Recherche de l'hyperplan optimal 17
6.1.2.2.1. Maximisation de la Marge : formulation primale 17
Définition de la Marge Maximale : 17
Comment maximiser la marge de mon classifieur : 20
6.1.2.2.2. Maximisation de la Marge : formulation duale et expression du noyau 21
6.1.2.2.3. Les conséquences 22
6.1.2.3. Dans le cas non séparable 23
6.1.2.3.1. Principe du Kernel trick 23
Présentation de la façon de convertir un algorithme linéaire en algorithme non linéaire: Cas n° 1, problème de classification binaire supervisée 23
Présentation de la façon de convertir un algorithme linéaire en algorithme non linéaire : Cas n° 2, problème de régression 24
Astuce du noyau 25
6.1.2.3.2. Choisir son noyau 27
6.1.3. Extensions 28
6.1.3.1. Le cas de la séparation imparfaite : Marge Souple « Soft Margin » 28
6.1.3.2. Cas de problème de classification multi-classe 29
6.1.3.3. Relation entre le classifieur SVM et régression logistique 30
6.2. Les méthodes SVM dans la pratique 31
6.2.1. Les 12 questions pour réussir son SVM (Lorette Noiret) 31
6.2.2. Les aspects logiciels et matériels 32
6.2.3. Paramétrer un SVM performant en apprentissage inductif 33
6.2.4. Comparer plusieurs de méthodes à noyaux 37
6.2.5. Comparer le SVM à d'autres classifieurs 37

7. Les données 38
7.1. Périmètre des données 38
7.2. Présentations des données explicatives 38
7.3. Utilisation d'un échantillon S.S.P. 39

8. Application numérique : le modèle SVM, comparaison avec les autres méthodes 42
8.1. Période 1 : résultats des 6 modèles (ACP, AFD, LDA, RL, RF, XGboost) 42
8.2. Période 2 : résultats du modèle RF avec grid search 51
8.2.1. Analyse descriptive du « dataset optimisé » 51
8.2.2. Performance du modèle des forêts aléatoires 57
8.3. Période 2 : résultats du modèle SVM avec grid search 63
8.3.1. Performance du modèle SVM 63
8.4. Période 2 : analyse comparée résultats SVM vs autres classifieurs 79

9. Conclusion 85
9.1. Bilan méthodologique et résultats 85
9.2. Prolongements 85

10. Annexes : 87
10.1. Note de cadrage de la thèse professionnelle 87
10.2. Eléments descriptifs du Groupe MATMUT 89
10.2.1. L'organisation du Groupe MATMUT 89
10.2.2. Le réseau commercial MATMUT 90
10.3. Benchmark des organisations Data Sciences 91
10.4. Synoptiques Stéphane TUFFERY 93
10.4.1. Tableau des méthodes descriptives 93
10.4.2. Zoom sur les méthodes prédictives 94
10.4.3. Machine Learning summarized in one picture 94
10.5. SVM 95
10.5.1. Théorème de Tom COVER : 95
10.5.2. Résumé article : A Practical Guide to Support Vector Classification 96
10.5.3. Définition des paramètres internes du modèle SVM 97
10.6. Calculs numériques - Modélisations Période 1 98
10.6.1. Code 98
10.6.2. Analyse 175
10.7. Calculs numériques - Modélisations Période 2 213
10.7.1. Formalisation du « dataset » de travail 213
10.7.2. Le classifieur forêts aléatoires « Random Forest » 280
10.7.3. Le classifieur Séparateur à Vaste Marge « SVM ». 305

11. Bibliographie 344

12. Glossaire des principaux termes et concepts 345

13. Certificat Mooc - Méthodes de sondage et d'enquête proposé par ULB 346

****

****

LISTE DES TABLEAUX

Tableau 1 - synoptique de la matrice de confusion 12
Tableau 2 - synthese de la performance des classifieurs periode de calcul 1 42
Tableau 3 - synthese de la performance du classifieur Random Forest periode de calcul 2 57
Tableau 4 - synthese de la performance du classifieur SVM Forest periode de calcul 2 63
Tableau 5 - fiche de lecture - A Practical Guide to SVM Classification 65
Tableau 6 - synthese de la performance des 2 classifieurs retenus 80
Tableau 7 - analyse ROC – « detecter les avions allemands sans detecter tous les oiseaux » 83
Tableau 8 - synthese des resultats et recommandations 85

****

****

LISTE DES FIGURES

Figure 1 : « la ruse du noyau = kernel trick » : augmenter la dimension des vecteurs Xi de « 1 » 15
Figure 2 : discrimination lineaire et hyperplan separateur de 2 regions 16
Figure 3 : distance entre un point x et ma surface de decision..... 17
Figure 4 : hyperplan & marge d'un SVM 18
Figure 5 : forme du vecteur de poids et hyperplan affine 19
Figure 6 : maximiser la marge de mon classifieur 20
Figure 7 : representation graphique de vecteurs supports. 21
Figure 8 : transformer un classifieur lineaire en classifieur non lineaire avec des 2 fonctions de bases gaussiennes $\emptyset j$ 23
Figure 9 : l'importance de la matrice de Gram * 24
Figure 10 : Extrait theoreme de Mercer, valeurs & vecteurs propres 25
Figure 11 : fonctions noyaux derivees valides (10 exemples) 27
Figure 12 : Max voting' strategy is not always conclusive 29
Figure 13 : SVM algorithme de regression 30
Figure 14 : fonction perte de la reglog version lisse d'un SVM 30
Figure 15 : degre de complexite optimale d'un modele en fonction des erreurs d'apprentissage et de prevision 33
Figure 16 : graphe de la dimension VC 33
Figure 17 : effet de la regularisation et du parametre de noyaux pour la discrimination a l'aide d'un SVM. 36
Figure 18 : « Pipeline» construire un dataset pour realiser une prediction avec un modele SVM 38
Figure 19 : rappel methodologique LDA vs QDA..... 86

****

1. Remerciements

. Je tiens à remercier Valérie Brivet et Rémi Coeugniet, salariés MATMUT, pour m'avoir accueilli et permis de réaliser ces travaux lors de mon année en contrat de professionnalisation au sein du MATMUT Datalab.

. Je tiens à Remercier l'équipe Pédagogique de l'INSA Rouen, Michel Bobbia, Jean-Michel Poggi, Jean-Marc Lasgouttes, Arnaud Knippel pour les réponses et conseils prodigués durant la rédaction de ce livrable, et en particulièrement auprès de mon Directeur de Thèse professionnelle, Bruno Portier, pour sa disponibilité et la qualité de ses enseignements.

2. Mode d'emploi du document

. Le présent livrable est organisé en deux parties :
- une partie de présentation des fondements mathématiques du SVM,
- une partie d'application numérique sur les données MRSQ.

Note à l'attention des lecteurs :

. Je préconise aux lecteurs de faire les allers-retours nécessaires entre les deux parties, théorie et application, afin de rendre lisible les résultats obtenus et de matérialiser les enseignements sur la problématique de transformation des devis en contrats.

. Concernant la partie des fondements mathématiques, les différents aspects théoriques du classifieur SVM à connaître sont détaillés au travers de fiches indépendantes et didactiques présentées sous format d'un tableau à 3 colonnes défini comme suit :

- colonne 1 [à gauche] : expression littérale (i.e. formule mathématique et présentation des théorèmes)
- colonne 2 [au milieu] : figure (i.e. illustration des problèmes sous forme de graphiques)
- colonne 3 [à droite] : commentaires (i.e. mise en relation avec notre objectif de construire un modèle de prédiction)

Ce dispositif complété par la liste des principaux tableaux et figures a pour objectif de permettre aux lecteurs, non avancés dans la mobilisation des méthodes d'apprentissage statistique, de comprendre les avantages et les intérêts des classifieurs SVM pour les futurs travaux de MATMUT DATALAB.

Remarque : présence d'une signalétique MATMUT

. Dans ce document, une signalétique a été mise en place à destination des permanents MATMUT DATALAB définie comme suit :

⚠ **Bonnes Pratiques DATALAB :**

Cette signalétique fait suite à un échange avec Madame Valérie Brivet, ma tutrice entreprise, qui souhaite que je puisse prendre du recul lors de la rédaction de cette thèse professionnelle afin dégager et de partager des bonnes pratiques « en data sciences » auprès des salariés permanents du MATMUT DATALAB.

3. Résumé

- résumé -

Objectifs	. Etudier le problème de transformation de devis commercial en contrat sociétaire au travers d'un classifieur capable de prédire une classe sur un problème de classification binaire supervisée avec des classes déséquilibrées *.
Méthode	. Lors d'une première période de calcul, l'étude de 6 modèles statistiques m'a permis de retenir le classifieur Random Forest avec 6 descripteurs continues dont 2 considérés comme variables réellement discriminantes. Ce travail a pointé 3 hypothèses sous-jacentes à notre modèle des données formulées telles que : (i) l'absence de relation linéaire entre les Xi et le Y, (ii) le rôle structurant des classes déséquilibrées de notre Y et (iii) le défaut d'avoir identifié toutes les Xi discriminantes nécessaires à la modélisation de notre problème. Dans un deuxième temps, après avoir formé un échantillon par Sondage Stratifié Proportionnel (n = 6150), en se basant sur une procédure de « Grid Search, avec une validation croisée (k=10) et un mode de calcul parallèle », j'ai réussi à isoler 2 classifieurs optimisés définis tels que : (a) RF "gridsearch - ROC", AUC = 0.61 et F-mesure = 0.74, (b) SVM Gaussien C = 1000 et $\gamma = 0.15$, AUC = 0.56 et F-mesure = 0.74.
Résultats	. Dans une hypothèse naïve, je cherchais à expliquer en quoi la flexibilité de l'approche SVM permettait de fiabiliser les prédictions de classification face aux performances connues du modèle des Forêts Aléatoires (RF). Sur ce modèle des données, cette approche n'est pas vérifiée. En se basant sur la métrique de performance adaptée à notre problème, le ROC-AUC, le meilleur classifieur est le modèle Random Forest spécifié tel que : RF "gridsearch - ROC", AUC = 0.61 et F-mesure = 0.74. Le modèle concurrent SVM Gaussien n'est pas retenu car il n'a pas su détecter la classe des « OUI ». Au final, on enregistre les résultats insuffisants du RF retenu qui détecte seulement 12.27% de « OUI ».
Logiciel / fonction	. Pour le traitement numérique, j'ai codé l'application de l'algorithme SVM en langage R [version 3.4.0 (2017-04-21)] avec son IDE R studio [version 1.0.153 pour Mac OS X 10_12_6 / MacBook Air 13 – Ram = 4 GO - DDR3 - 1600 MHz] et accompagné des packages : caret, plyr, ROCR, ggplot2, kernlab, parallel, doMC, iterators, foreach.

* déséquilibre des classes de la variable d'intérêt : trsf_dev [N ; O]

- abstract -

Objectives	. To study the subject of sales quotations conversion into corporate contracts through a classifier able to predict a class on a supervised binary classification problem with imbalanced classes *.
Calculus Framework	. In a first calculation period, the study of six statistical models allowed me to retain the Random Forest classifier with six continuous descriptors, two of which are considered to be really discriminating variables. This work has pointed out three underlying assumptions of our data model such as: (i) the lack of a linear relationship between Xi and Y, (ii) the structuring role of the unbalanced classes of our Y, and (iii) the failure to have identified all the discriminating Xi necessary for the modeling of our problem. In a second step, after building a sample by Proportional Laminate Survey (n = 6150), based on a Grid Search procedure, cross validation (k = 10) and a parallel calculation method, I have been able to discover 2 defined optimized classifiers such as : (a) RF "gridsearch - ROC", AUC = 0.61 and F-measure = 0.74, (b) Gaussian Kernel with C = 1000 and $\gamma = 0.15$, AUC = 0.56 and F-measure = 0.74.
Key Results	. In a naive hypothesis, I tried to explain how the flexibility of the SVM approach made it possible to make the classification predictions more reliable compared to the well-known performances of the Random Forest model (RF). On this data model, this approach is not verified. Based on the performance metric adapted to our problem, the ROC-AUC metric, the best classifier is the Random Forest model specified as : RF "gridsearch - ROC", AUC = 0.61 and F-measure = 0.74. The RBF kernel competitor model is not retained because it failed to detect the class of "YES". In the end, the insufficient results of the selected RF are registered, which detects only 12.27% of "YES".
Integrated Dev Environment	. For the calculation processing, I coded the application of the SVM algorithm in R language [version 3.4.0 (2017-04-21)] with its IDE R studio [version 1.0.153 for Mac OS X 10_12_6 / MacBook Air13 - Ram = 4 GB - DDR3 - 1600 MHz] and based on these [R] packages : caret, plyr, ROCR, ggplot2, kernlab, parallel, doMC, iterators, foreach.

* Imbalanced classes of dependent variable : trsf_dev [N ; O]

4. Introduction

Dans le cadre de l'obtention du diplôme de l'Education Nationale, Master Spécialisé (bac +6) « Expert en sciences des donnée » réalisé au sein de l'Ecole INSA de Rouen, j'ai souhaité étudier le classifieur Séparateur à Vaste Marge (appelé SVM[1]) et son application en entreprise pour le formaliser comme sujet d'application de ma thèse professionnelle.

Au cours de mon contrat de professionnalisation, du 07 novembre 2016 au 27 octobre 2017, au sein de l'Employeur MATMUT, j'ai produit une série de travaux en « data sciences » sur le marché de l'assurance automobiles IARD. Dans le champ des techniques de fouilles de données sur un problème de classification binaire supervisée, j'ai cherché à expliquer et à prédire la transformation de devis prospects en contrats sociétaires automobiles ?

Ce chantier s'est réalisé en 2 phases définies comme suit :
. période 1 - trim4/2016 : appréhender le jeu de données MATMUT et prédire des premiers résultats de classification,
. période 2 - trim3/2017 : mobiliser le classifieur Séparateur à Vaste Marge « appelé SVM » et vérifier si les résultats de prédiction sont améliorés ?

Le plan de ce livrable se défini comme il suit :

. Après une présentation formelle de la théorie de SVM pour expliquer en quoi le classifieur SVM peut être un classifieur alternatif aux modèles usuels,

. Au moment la présentation des résultats de l'application numérique, d'expliquer en quoi la flexibilité de l'approche SVM permet de fiabiliser les prédictions de classification face aux performances connues du modèle des Forêts Aléatoires (RF). Ce dernier étant qualifié de classifieur « boîte noire où tout est aléatoire ».

. Lors de la rédaction de ce travail, j'attacherai une attention particulière à traduire les méthodes détaillées et les résultats obtenus en outils actionnables pour une future mise en oeuvre au sein du MATMUT Data Lab.

[1] SVM = Séparateur à Vaste Marge, Machine à Vecteurs de Supports, et RF = « Random Forest » = Les Forêts Aléatoires,

5. Le contexte et le problème

5.1. La Matmut : le contexte d'affaires

. Le Groupe MATMUT, est un assureur Mutualiste ou encore Mutuelle d'assurance qui assure 3.2 millions de sociétaires et 6.8 millions de contrats au 31 décembre 2016.[2] Cet acteur du marché français affiche un Chiffre d'Affaires de 2.003 Milliards d'Euros en 2016. Son cœur de métier historique repose sur le marché de l'assurance IARD[3], c'est à dire principalement sur deux types de contrats : assurance automobile et assurance habitation.

Depuis l'année 2015, sur le marché de l'assurance automobile qui est un marché historiquement très concurrentiel, la MATMUT, enregistre un fléchissement du nombre de contrats automobiles malgré l'établissement d'une politique tarifaire qualifiée par ses dirigeants « de commercialement agressive ». Deux phénomènes ont été observés parallèlement à cette baisse du nombre de sociétaires automobiles. D'une part une accélération du taux de réalisation suite à l'entrée en vigueur de la Loi HAMON[4], d'autre part un ralentissement du taux d'affaires nouvelles en automobile stabilisé à 3% pour l'année 2016.

5.2. Le cadre de la thèse professionnelle

5.2.1. Le problème de la transformation

. Dans ce contexte de marché difficile, la MATMUT a souhaité confié à la nouvelle initiative DATALAB hébergée au sein du service SCD, ayant en charge l'activité historique de l'infocentre, de travailler la problématique « business » définie comme suit :

Comment optimiser la transformation des devis commerciaux en "OUI" contrats sociétaires ?

. Cette question de transformation commerciale de devis en contrat prend un « sens marketing particulier » pour un opérateur mutualiste comme la MATMUT et doit être expliquée pour mieux comprendre la spécification des modèles de classifieurs mise en œuvre in fine au sein de ce travail.

➔ Sur le marché de l'assurance automobile (IARD) – comment les opérateurs d'assurance gèrent le risque contrat selon la nature de ce risque ?

- pour un Bancassureur, le raisonnement est le suivant : « j'apprécie le risque, je tarifie et je prends les risques clients qui m'intéresse (i.e. problème classification traité par régression - scoring) ».

- pour un assureur Mutualiste, le raisonnement est le suivant : « je prends ou ne prends pas le risque d'assurer un contrat auto (i.e. problème Classification binaire supervisée du type Oui/Non) ».

[2] Données issues - Rapport SFCR 2017 – Matmut Groupe.
[3] IARD = contrat d'assurance couvrant les risques incendie, accidents et risques divers.
[4] Depuis le 01/01/15, la loi Hamon donne aux consommateurs le pouvoir de résilier leur contrat d'assurance auto, moto et habitation à la date de leur choix, passé un an de contrat.

. A la MATMUT : les principes suivant régissent la politique de gestion de risque et flèchent la politique tarifaire IARD Autos.

- **Principe 1** : toute la politique de souscription des risques est expliquée par la politique tarifaire suivante. Le « modèle pricing – actuaire » implique une relation linéaire forte entre les variables (i.e. les paramètres internes) qui structurent le modèle tarifaire et qui fixe au final le prix de la cotisation annuelle payé par le sociétaire et le montant du devis affiché aux prospects pour un contrat d'assurance automobile.

- **Principe 2** : Une politique de risque adverse qui cherche à recruter en « portefeuille risque auto » des clients du type « bon père de famille ». Cette politique se traduit dans les faits, lors de la production de devis commerciaux, par une prime annuelle élevée pour les porteurs de risques qui n'entrent pas dans cette politique de gestion de portefeuille.

Cette politique de gestion se matérialise sur notre jeu de données par la répartition « NON » et « OUI » suivante :
Pour un effectif de 614048 lignes devis :

```
# comparaison avec les valeurs de la Population Totale
# POURCENTAGE : poids de l'etiquette NON / OUI
PoidsY=(table(dv17$trsf_dev)/nrow(dv17)*100)
PoidsY

#   N        O
#   73.31072 26.68928
#   dataset avec classes déséquilibrées (les NON majoritaires)

Ratio_OUI_NON = PoidsY[[2]]/PoidsY[[1]]*100
Ratio_OUI_NON # [1] 36.40
```

Résultats : le nombre Devis NON = 73.31% > nombre Devis OUI = 26.68% ➔ ratio OUI/NON = 36.40%

. et un nombre de devis moyen supérieur à 2 (= 2.145)

```
> summary(dv18)
   ID_CLIENT           NUM_CTC         NUM_DEV          C_AN_TTC          COEFF          PUIS_FISC        AGE_CLI          NB_ENF        trsf_dev      ANC_VEH
 Min.   :211978166   Min.   : 1.00   Min.   :  1.000   Min.   :   65.0   Min.   : 35.00   Min.   : 0.000   Min.   : 13.00   Min.   : 0.0000   N:450163   Min.   :-1.000
 1st Qu.:212234208   1st Qu.: 1.00   1st Qu.:  1.000   1st Qu.:  434.3   1st Qu.: 40.00   1st Qu.: 4.000   1st Qu.: 23.00   1st Qu.: 0.0000   O:163885   1st Qu.: 5.000
 Median :212379122   Median : 1.00   Median :  2.000   Median :  635.3   Median : 65.00   Median : 5.000   Median : 32.00   Median : 0.0000              Median : 9.000
 Mean   :212395481   Mean   : 1.61   Mean   :  2.145   Mean   :  729.5   Mean   : 66.27   Mean   : 5.577   Mean   : 36.88   Mean   : 0.8838              Mean   : 9.495
 3rd Qu.:212567554   3rd Qu.: 2.00   3rd Qu.:  2.000   3rd Qu.:  915.3   3rd Qu.: 95.00   3rd Qu.: 6.000   3rd Qu.: 49.00   3rd Qu.: 2.0000              3rd Qu.:14.000
 Max.   :212760888   Max.   :17.00   Max.   :318.000   Max.   :13335.3   Max.   :256.00   Max.   :68.000   Max.   :168.00   Max.   :20.0000              Max.   :60.000
```

(attention sur données brutes, sans la phase de preprocessing)

à retenir pour la suite de la thèse professionnelle :

. La relation commerciale Prospect-Matmut dans le cadre de l'établissement d'un devis, produit plusieurs devis au travers de plusieurs contacts (les variables NUM_CTC et NUM_DEVIS).

. Pour matérialiser cette situation : vue d'un sociétaire qui du 03 au 07 janvier 2015 a obtenu 6 devis pour le même véhicule.

ReglogOrdinal.R × | binary_taxinc_analysis.R × | binary.taxinc_quali_16mai17.R × | classe_taxinc2_analysis.R × | 00_prepadata.R × | dv × | MRSQ_svm_1508.R × | RandomForest.R ×

Filter

	ID_CLIENT	REF_SOC	STE_SOC	TYP_CLI	NUM_CTC	NUM_DEV	STE_CIBLE	DAT_REAL	SIT_REAL	FLE_MRSQ	CD_GAM_MRSQ	AN_FAB	APP_COM	C_AN_TTC	CD_BFAM	CD_FRAN	CD_REMIS	CLAS_DEV	COEFF	COM_MRSQ
1	211978166	980001139026	M: MATMUT	P: PROPOSANT	3	1	M	03/01/2015	75L1	2: EQUILIBRE	1 : GAMME REFERENCE	6	1.4 TONIC	1 630,30	N	D1	N	A	100	CLICHY SOUS BOIS
2	211978166	980001139026	M: MATMUT	P: PROPOSANT	3	2	M	03/01/2015	75L1	5: TIERS VOL INCENDIE	1 : GAMME REFERENCE	6	1.4 TONIC	1 076,30	N	D1	N	A	100	CLICHY SOUS BOIS
3	211978166	980001139026	M: MATMUT	P: PROPOSANT	3	3	M	03/01/2015	75L1	5: TIERS VOL INCENDIE	1 : GAMME REFERENCE	6	1.4 TONIC	1 076,30	N	D1	N	A	100	CLICHY SOUS BOIS
4	211978166	980001139026	M: MATMUT	P: PROPOSANT	3	4	M	03/01/2015	75L1	4: URBAIN	1 : GAMME REFERENCE	6	1.4 TONIC	1 245,30	N	D1	N	A	100	CLICHY SOUS BOIS
5	211978166	980001139026	M: MATMUT	P: PROPOSANT	3	5	M	03/01/2015	75L1	2: EQUILIBRE	1 : GAMME REFERENCE	6	1.4 TONIC	1 630,30	N	D1	N	A	100	CLICHY SOUS BOIS
6	211978166	980001139026	M: MATMUT	P: PROPOSANT	6	1	M	07/01/2015	7546	2: EQUILIBRE	1 : GAMME REFERENCE	6	1.4 TONIC	1 471,30	N	D1	N	A	100	CLICHY SOUS BOIS

. Concernant la production de ces devis prospects, la majorité de ces derniers sont réalisés par le réseau physique des 525 agences implantées sur le territoire national[5].

[5] Pour plus de détails se rendre en Annexes, paragraphe 8.2.2.

5.2.2. La formalisation du problème mathématique

. Le point de départ du raisonnement pour formaliser le problème mathématique » est de répondre aux questions posées par les métiers, c'est à dire par les réseaux commerciaux et le service marketing :

INPUT	OUTPUT
	QUESTIONS BUSINESS
. le réseau commercial adresse des devis assurance automobile à des prospects	. le réseau commercial veut des réponses **automatiquement** aux questions : - quels sont les devis qui risquent de se transformer ? [étiquette : OUI] - quels sont les devis qui risquent de ne pas se transformer ? [étiquette : NON] Livrable : . une liste de tous les devis prospects classés selon les 2 modalités qualitatives [OUI] / [NON] L'objectif opérationnel business : . avoir un système d'aide à la décision qui produit automatiquement une étiquette de classification [OUI] / [NON] pour chaque devis produit afin de : - investir du temps commercial pour transformer les prospects [OUI] ou potentiel en sociétaire - ne pas investir du temps commercial sur les prospects [NON].
	PROBLEME MATHEMATIQUE
. chaque ligne devis avec un vecteur de descripteurs (Xi1, … , Xin)	. Production d'une étiquette de classe Yi avec i ∈ [OUI, NON] pour chaque ligne devis CONCLUSION : **. Le problème de transformation de devis prospect en contrat sociétaire se formalise mathématiquement comme un problème de classification binaire.** . reste à confirmer : la classification binaire est-elle supervisée, semi-supervisée ou non supervisée ?

Nota : mise en œuvre opérationnelle sur des données devis en mode « BATCH[6] » pour des prospects.

INPUT	OUTPUT
	QUESTIONS BUSINESS
. J'ai des données qui caractérisent mes sociétaires porteurs de contrat automobiles (base d'apprentissage, test, validation du classifieur le plus performant)	Je veux répondre **automatiquement** aux questions suivantes : . Parmi tous les devis prospects que le réseau commercial a généré, - quels sont les devis vraiment transformés ? ***[Vrai Positif] = VP*** - quels sont les devis vraiment non transformés ? ***[Vrai Négatif] = VN*** - parmi les devis qui ont des chances ou non de se transformer, comment identifier ceux où je dois investir du temps pour les transformer ? ***[Faux Négatif] = FN*** - parmi les devis qui ont des chances ou non de se transformer, comment identifier ceux où je ne dois pas investir du temps car ils ne risquent de ne pas se transformer ? ***[Faux positif] = FP*** ***RESULTATS ATTENDUS : classer automatiquement des devis générés selon les 4 catégories (ou appelées Etiquette i) pour optimiser le temps de productivité commercial*** ***E1 = [VP] = le réseau commercial consacre le temps nécessaire pour ce devis gagné,*** *E2 = [VN] = le réseau commercial n'investit pas de temps pour ce devis perdu,* ***E3 = [FN] = le réseau commercial investit du temps pour un devis qui peut être gagné,*** *E4 = [FP] = le réseau commercial n'investit pas de temps pour ce devis qui risque d'être perdu,* Résultats voulus : construire une machine qui produit automatiquement un Ensemble étiquettes E = {E1, E2, E3, E4} lorsqu'on lui présente un devis commercial.
	PROBLEME MATHEMATIQUE
. temps 1 : sur une base de devis sociétaires, j'ai des couples (Xi, Yi) (base découpée en échantillons apprentissage, test, et validation) . temps 2 : sur une nouvelle base devis prospect (Xi_new) seulement	. temps 1 : avec (Xi, Yi) entraîner un classifieur à produire des Yi_chapeau, avec paramétrisation du modèle optimal et choix optimal du classifieur selon les critères de performance de classification : matrice de confusion, Accuracy (basé sur le taux de bien classés) et AUC (ROC) . temps 2 : je peux produire des Yi_chapeau_new, prédire l'appartenance de Xi_new à l'étiquette de classe Yi avec i ∈ [OUI, NON] CONCLUSION : **. Le problème de transformation de devis prospect en contrat sociétaire se formalise mathématiquement comme un problème de classification binaire supervisée qui entre dans le cadre l'apprentissage statistique supervisé.**

[6] Ce service d'aide à la décision reste à construire dans le SI Matmut pour les conseillers du réseau commercial.

Synoptique de la matrice de confusion : (organisation à partir des sorties du langage R)

		REFERENCE = réelle	
		NON	**OUI**
PREDICTION	**NON**	VN	FN
	OUI	FP	VP

Signalétique

effort commercial pour un objectif de croissance numérique du portefeuille contrat automobile	
VN	(- - -)
FP	
VP	(+)
FN	(+++)

Tableau 1 - synoptique de la matrice de confusion

(à construire sur la base apprentissage, test)

⚠ **Bonnes Pratiques DATALAB :**
Plusieurs remarques méthodologiques à prendre en compte pour adapter la spécification du modèle à la problématique métier :

Remarque 1 : prendre en compte le poids des modalités (i.e. des classes) de notre variable d'intérêts
(Poids des Devis NON = 73.31% > Poids des Devis OUI = 26.68%)

On qualifie alors notre problème de classification comme **un problème de classification binaire supervisée avec classes déséquilibrées,** dans notre cas « moyennement déséquilibré ». Dans notre cas, ce déséquilibre de classes concernant l'étiquette sera pris en compte comme un paramètre à spécifier lors de la phase d'apprentissage du classifieur. Sous l'environnement de développement R, avec l'IDE R studio, les principaux packages permettant de traiter du modèle SVM intègre ce déséquilibre de classe comme un paramètre à spécifier.

Dans le cas, où le DATALAB serait confronté à des modèle d'apprentissage statistique avec classes très déséquilibrées (99% de NON pour 1% de OUI), d'autres techniques plus avancées seraient à mobiliser dans la phase de préparation des données (méthode de simulation d'échantillon, redresser et augmenter la classe minorée, redresser et diminuer la classe sur représentée, les 2 en même temps, fixer un poids a priori pour les deux classes à 0.5). Pour plus de détails, se référer à la [7]référence bibliographique "Learning from Imbalanced Data".[8]

Remarque 2 : d'un point de vue « business », il serait facilitant d'étudier seulement « prédire le DEVIS-OUI », mais d'un point vue robustesse des résultats, il est judicieux de contrôler la cohérence des valeurs obtenues pour tous les cas identifiés dans la matrice de confusion.

Remarque 3 : dans le cas où l'étude souhaiterait se concentrer sur l'événement « DEVIS-OUI », avec une fréquence d'apparition du « OUI » très faible (i.e. remarque 2), il existe des modèles qui traitent de cette problématique de détection d'événement rare.

[7] Analyse discriminante, LDA vs. QDA, Mathieu Emily, Agrocampus ouest,
[8] Possibilité d'utiliser le côté pratique du package R - https://journal.r-project.org/archive/2014/RJ-2014-008/RJ-2014-008.pdf

5.2.3. Etat des lieux de modèles prédictifs utilisés à la MATMUT

. Dans le cadre de son cœur de métier, le cadre standard de la tarification en assurance IARD (automobile et habitation) se base sur :

(1) la définition du contrat d'assurance définit comme il suit :

L'assurance est un contrat par lequel, moyennant le versement d'une prime dont le montant est fixé a priori (en début de période de couverture), l'assureur s'engage à indemniser l'assuré pendant toute la période de couverture (disons un an). Cette prime est censée refléter le risque associé au contrat (on peut renvoyer à Denuit & Charpentier (2004) sur la théorie du calcul des primes, et à Denuit & Charpentier (2005) pour les considérations économiques). Pour chaque police d'assurance, la prime est fonction de variables dites de tarification (permettant de segmenter la population en fonction de son risque). Généralement, on considère

- des informations sur l'assuré, comme l'âge ou le sexe pour un particulier, ou le secteur d'activité et le nombre de salariés pour une entreprise,
- des informations sur le bien assuré, comme l'âge du véhicule, la puissance ou la marque en assurance auto, la surface du logement en multirisque habitation, le chiffre d'affaire de l'entreprise en perte d'exploitation,
- des informations géograhiques comme le revenu moyen dans la commune ou le département, la densité de médecins, la densité de population, etc.

La *fréquence* (annualisée) est le nombre de sinistres divisé par l'exposition (correspondant au nombre d'années police) pour une police d'assurance, ou un groupe de polices d'assurance. La plupart des contrats étant annuels, on ramènera toujours le nombre de sinistres à une exposition annuelle lors du calcul de la prime, et on notera N la variable aléatoire associée. Durant la période d'exposition, on notera Y_i les coûts des sinistres, c'est à dire les indemnités versées par l'assureur à l'assuré (ou une tierce personne). La charge totale par police est alors $S = 0$ s'il n'y a pas eu de sinistres, ou sinon :

$$S = Y_1 + \cdots + Y_N = \sum_{i=1}^{N} Y_i.$$

Source : L'actuariat avec R, la tarification a priori, p.37, Arthur Charpentier, Christophe Dutang, décembre 2012.

(2) L'analyse de la prime d'assurance payée par l'assuré se basant sur le modèle fréquentiste [fréquence du sinistre x coût du sinistre] où l'impact des variables explicatives sur la valeur du risque est historiquement modélisé par le Modèle Linéaire Généralisé (GLM) de régression paramétrique.

Dans ce cadre standard de la tarification de l'assurance IARD, ces modèles de régression GLM sont utilisés par principalement par deux départements à la MATMUT: Le département de la tarification qui peut travailler, par exemple, sur des calculs d'élasticité prix/demande et positionnement tarifaire concurrentiel, et le département de l'actuariat qui souhaite modéliser la sinistralité des contrats proposés et déterminer le niveau de provisions techniques pour être conforme aux réglementations en vigueur.

Chapitre 3

Les provisions pour sinistres à payer

Dans ce chapitre, nous allons étudier les méthodes pour calculer le montant des *provisions pour sinistres à payer*, et plus particulièrement, des méthodes permettant de quantifier la marge d'erreur associée. Comme les définit Simonet (1998), "*les provisions techniques sont les provisions destinées à permettre le règlement intégral des engagements pris envers les assurés et bénéfi-ciaires de contrats. Elles sont liées à la technique même de l'assurance, et imposées par la règlementation*". D'un point de vue plus formel, à la date t, la compagnie d'assurance est tenue de constituer une provision pour les sinistres survenus avant la date t qu'elle sera tenu d'indemniser. Elle doit donc estimer le coût des sinistres survenus, et retrancher les montants déjà versés. Il s'agit donc fondamentalement d'un problème de prévision.

Source : L'actuariat avec R, la tarification a priori, p.91, Arthur Charpentier, Christophe Dutang, décembre 2012.

Mais l'exploitation des nouvelles sources de données en assurance IARD (i.e. : prise OBD sur les véhicules, pour remonter des données de comportement de conduite de l'assuré), couplées avec l'augmentation des performances des systèmes d'information entreprise permettrait à la MATMUT, dans le cadre du DATALAB d'opérer des modélisations non paramétriques ou semi paramétriques. Cette nouvelle approche dans la construction de modèles statistiques assurancielle se matérialise par le courant d'apprentissage statistique « appelé machine learning » qui permet de desserrer les contraintes du formaliste théorique de la régression paramétrique GLM.[9]

[9] Pour plus de détails, voir les synoptiques zoom sur les méthodes prédictives, tableau des méthodes descriptives - Stéphane TUFFERY – présentées en Annexes.

5.2.4. Présentation de la méthode et traitements des calculs numériques

. La mise en production des calculs numériques s'organise selon le détail de la méthode suivante :

Période 1 : trim4/2016	Les livrables	Eléments de Méthode
. appréhender le jeu de données MATMUT et prédire des premiers résultats de classification	. analyser les données (728674 lignes x 112 col.) . sélectionner les variables discriminantes . tester 6 modèles : ACP, AFD, LDA, RL, RF, XGboost . sélectionner le classifieur de référence	. analyser et matrice des corrélations des Xi . sélectionner les Xi quantitatives discriminantes (expert métier et compréhension tarification matmut) . vérifier la pertinence d'intégrer les Xi qualitatives . vérifier la relation linéaire ou non entre Xi quanti et Yi . identifier le modèle le plus parcimonieux Yi~Xi . retenir un classifieur avec critères (Accuracy, AUC)
Procédure de « stress tests » pour benchmarker les résultats du classifieur Random Forest		. consolider résultats RF D. Lepicaut vs résultats RF N MOKHEDDES (autre analyste) . organiser un test de Mc Nemar[10] . conclure sur les résultats du Random Forest[11]
Période 2 : trim3/2017	Les livrables	Eléments de Méthode
. mobiliser le classifieur SVM et vérifier si les résultats de prédiction sont améliorés ?	. dataset optimisé . produire résultats modèle RF grid search . produire résultats modèle SVM grid search . comparer résultats SVM vs autres classifieurs	. traiter les data : NA, centré-réduit, outlier, normalisé . construire un échantillon optimisé de 6150 lignes x 7 colonnes par la méthode de sondage stratifié proportionnel (réduire temps et faisabilité des calculs en SVM) . construire des valeurs optimisées Accuracy, AUC avec le modèle RF grid search (classsifieur de référence Période 1) ➔ construire un « point de référence » des critères de performances ACCURACY, AUC d'un classifieur optimisé sur notre dataset échantillon. . construire des valeurs optimisées Accuracy, AUC avec le modèle SVM grid search . TEST statistiques : comparer et expliquer résultats SVM vs autres classifieurs ➔ identifier le classifieur le plus performant en prédiction pour garantir les réponses à notre problématique business.

En résumé :
- Construire et optimiser un classifieur « Séparateur à Vaste Marge » comme un modèle alternatif au Random Forest.
- avec le protocole de traitement suivant :

1. Analyse exploratoire des données,
2. Traitement et transformation des données,
3. Construction d'un modèle de référence en utilisant le Random Forest optimisé « grid search »,
4. Construction d'un modèle SVM optimisé « grid search »,
5. Interprétation des résultats,
6. Etude d'impacts pour la Matmut.

⚠ **Bonnes Pratiques DATALAB :**

Pour auditer la cohérence et la performance d'un classifieur en apprentissage statistique, il y a 3 précautions à prendre :
(1) Privilégier une procédure de validation croisée sur plusieurs plis (i.e. k-folds cross validation, avec k=10),
(2) « Stress tester » les résultats d'un classifieur entre plusieurs analystes du DATALAB,
(3) Tester le classifieur retenu sur un échantillon test « caché » du dataset intial (méthode Kaggle.com[12]).

[10] Test qui permet de vérifier comment 2 classifieurs classent correctement ou non les exemples d'un échantillon test (cf. références bibliographiques sur le sujet).
[11] Validation des résultats du classifieur RF sur une procédure simple échantillons apprentissage et test.
[12] Pour plus de détails se rendre à la conclusion (9.) du benchmark des organisations Data Sciences en annexes.

6. La méthode des SVM : « les Machines à Vecteur de Support ou Séparateurs à vaste Marge »

6.1. Les Fondements Mathématiques

6.1.1. Présentation résumée de Yann LeCun

. Dans le champ des techniques d'apprentissage statistique pour résoudre les problèmes de classification binaire supervisée, la pertinence du classifieur « Séparateur Vaste Marge » repose sur les deux principes suivant :

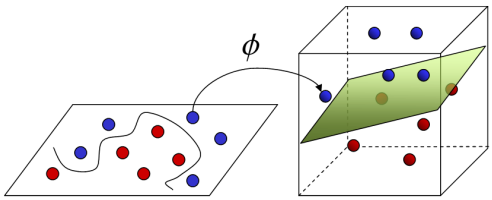
(1) La mise en application de règles de décision linéaires appelées « hyper plan séparateur » (plan affine),
(2) Lorsque les points data ne sont pas linéairement séparables, en appliquant le noyau « K = Kernel » adéquat, pour transformer - projeter les points-données dans un espace de plus grande dimension (que la dimension de l'espace d'origine) et pour trouver une solution qui rend le problème linéairement séparable.

C'est une application directe du théorème de Tom Cover (1965) : selon Yann leCun[13] « … La probabilité d'un étiquetage particulier d'un ensemble de p-points en n-dimension soit linéairement séparable, est très proche de « 1 » quand p < n, très proche de « 0 » quand p > n … cela appelle ».

Remarque 1 : beaucoup de données naturelles sont linéairement séparables,
Remarque 2 : en revanche de nombreux problèmes ne sont pas séparables avec un classifieur linéaire.

Comment rendre un problème linéairement séparable ?
. Piste 1 : extraire des caractéristiques appropriées au problème étudié pour faciliter le travail du classifieur (= trouver les Xi discriminants),
. Piste 2 : « la ruse du noyau = **kernel trick** » : augmenter la dimension des vecteurs Xi de « 1 », c'est à dire trouver une transformation non linéaire pour augmenter la dimension de l'espace initial des points-données, c'est à dire pour projeter les points-données de l'espace de départ vers un espace d'arrivée permettant de rendre les points-données linéairement séparables.

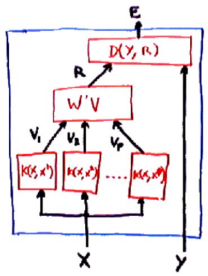
Vocabulaire : Kernel base methode = méthode à base de noyaux et $\emptyset = \text{"}Phi\text{"} = fonction\ de\ transformation$



Input Space — **espace départ : dim = 2 axes (x,y)**
problème non linéairement séparable

Feature Space — **espace d'arrivée : dim = 3 axes (x,y,z)**
Problème linéairement séparable

Figure 1 : « la ruse du noyau = kernel trick » : augmenter la dimension des vecteurs Xi de « 1 »



Sample-Centered Basis Functions (Kernels)

Place the center of a basis function around each training sample. That way, we only spend resources on regions of the space where we actually have training samples.

- Discriminant function:

$$f(X,W) = \sum_{k=1}^{k=P} W_k K(X, X^k)$$

- $K(X,X')$ often takes the form of a *radial basis function*: $K(X,X') = \exp(b||X - X'||^2)$ or a polynomial $K(X,X') = (X.X' + 1)^m$
- This is a very common architecture, which can be used with a number of energy functions.
- In particular, this is the architecture of the so-called Support Vector Machine (SVM), but the energy function of the SVM is a bit special. We will study it later in the course.

> le fonctionnement d'un SVM ?
. architecture à 2 niveaux en termes de blocs fonctionnels
. 1er niveau : compare le vecteur d'entrée à des sous-ensembles d'apprentissage
. chacune des boîtes sort un score de ressemblance (V1, V2 … Vk) calculé avec la fonction K = extraction de caractéristiques à partir de la fonction K,
. puis en 2nd niveau une combinaison linéaire des scores (W'V)
. il faut trouver la fonction « K = Kernel ad hoc » pour chaque problème,

> pourquoi les SVM sont performants en apprentissage automatique ?
. les machines à base de noyaux sont sur-paramétrisées, le vecteur de caractéristiques (Xi) a autant de dimensions qu'il y a d'exemples, cela apporte de la flexibilité pour classifier l'ensemble d'apprentissage.
. cad, rendre le modèle SVM suffisamment flexible, de dimension suffisamment élevée pour apprendre = classifier l'ensemble d'apprentissage,
. ensuite : régulariser, en mettant une pénalité sur les poids Wk de manière à ce que le système donne de bonnes performances sur un autre dataset et pour minimiser la taille du modèle.
. régularisation type : mettre une fonction coût qui permet d'avoir un petit nombre coefficients $W_k \neq 0$ (non nul) ce qui permet d'avoir une garantie meilleure sur la généralisation. (théorème de Vapnik = l'erreur en test sera d'autant plus petite que le nombre de coefficients $W_k \neq 0$ est petit)
. les X_k pour lesquels on a $W_k \neq 0$ s'appellent les vecteurs de supports.
. ils existent différents types de noyaux : linéaire, gaussien, radial, polynomial…

[13] https://www.college-de-france.fr/site/yann-lecun/course-2016-02-12-14h30.htm
slides disponibles : https://drive.google.com/drive/folders/0BxKBnD5y2M8NclFWSXNxa0JlZTg

6.1.2. Démonstration mathématique

6.1.2.1. Le SVM linéaire :

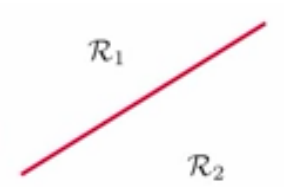
6.1.2.1.1. Principe général :

Classement binaire : discrimination linéaire et hyperplan séparateur

MOTIVATION : problème de classification binaire supervisée, famille des modèles à noyaux.
. Dans les algorithmes de classification, le modèle SVM est l'algorithme le plus utilisé, donc il est important à connaître.
. On part de la version du SVM linéaire (sans noyau), puis on utilise la formalisation du « problème dual » pour le mettre en noyau.

. **Intérêt : on va faire apprendre cet algorithme SVM sur un Dataset d'entraînement, puis après on pourra l'appliquer sur de nouvelles données avec la propriété de s'exécuter en ignorant notre sous-ensemble d'entraînement, donc au final il sera plus efficace.**

La façon de procéder du SVM, le calcul du noyau va être multiplié par zéro pour certains éléments de mon Dataset d'apprentissage. Ce qui justifie qu'on peut ignorer mon dataset d'apprentissage, il ne va pas influencer mon calcul de la prédiction.

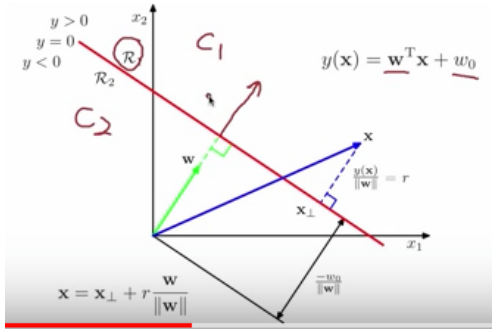
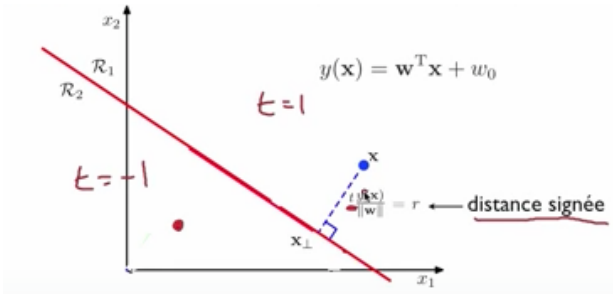
EXPRESSION LITTERALE	FIGURE	COMMENTAIRES
Sujet : classification binaire = séparabilité linéaire de 2 régions R1 / R2 associées à des Classes C1 et C2 La Classe C1, on la représente par une cible t = -1 La Classe C2, on la représente par une cible t = 1 . Définition : la surface de décision entre chaque paire de décision est linéaire, c'est à dire un hyperplan (i.e. une droite rouge, pour dim = 2). . Définition : on dit qu'un problème est linéairement séparable si une surface linéaire permet de le classifier parfaitement.	 *Figure 2 : discrimination linéaire et hyperplan séparateur de 2 régions*	. On commence par le code de la classification linéaire, puis on utilise la représentation duale de la définition du problème, pour utiliser l'astuce du noyau. Au final, ce qui permet d'obtenir un algorithme de classification non linéaire.

6.1.2.2. Recherche de l'hyperplan optimal

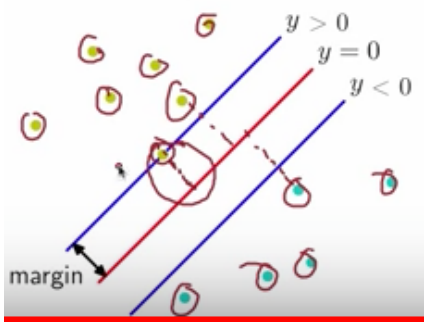
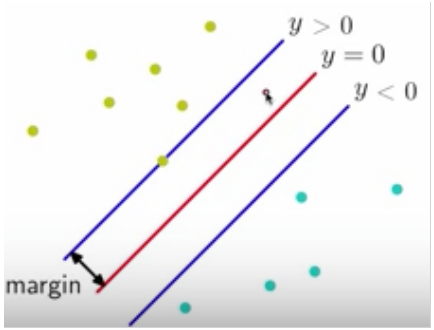
6.1.2.2.1. Maximisation de la Marge : formulation primale

Définition de la Marge Maximale :

. Dans l'algorithme SVM, la marge est au centre du critère que l'on souhaite optimiser quand on va entraîner un algorithme SVM.

EXPRESSION LITTERALE	FIGURE
Fonction discriminante : . il existe une relation entre une fonction discriminante $Y(x) = W^T x + wo$ (avec wo = biais) et un classifieur linéaire. . application d'un seuil >0, associé à la classe C1 région R1, sinon C2 région R2 (<0). . la surface de décision associée à cette fonction discriminante est un hyperplan (i.e. droite rouge pour Dim = 2) . le vecteur W est perpendiculaire à la surface de décision, . entraîner W c'est déterminer l'orientation de l'hyperplan qui sépare les 2 classes C1, C2. . le biais wo détermine la position dans la direction de W. Définition : la distance entre un point x et ma surface de décision = $Y(x)$ / $\|W\|$. cette distance est positive, si la cible de x est t = 1, -> si il est bien classifié y(x) > 0 . cette distance est négative, si la cible de x est t = -1, -> si il est bien classifié y(x) < 0 Définition : distance signée = t * $Y(x)$ / $\|W\|$ C1 avec R1 : si mon point est bien classifié, t * $\frac{Y(x)}{\|W\|} > 0$ avec t = 1 C2 avec R2 : si mon point est bien classifié, t * $\frac{Y(x)}{\|W\|} > 0$ avec t = - 1 Conclusion : si mon point est bien classifié, alors la distance signée à l'hyperplan est toujours > 0	  *Figure 3 : distance entre un point x et ma surface de décision*

(suite définition de la marge maximale : interprétation graphique)

EXPRESSION LITTERALE	FIGURE	COMMENTAIRES
<u>Définition</u> : la marge est la plus petite distance signée entre la surface de décision et les entrées de l'ensemble d'apprentissage. <u>Définition</u> : le meilleur Classifieur SVM est celui qui maximise la marge, c'est à dire en partant de hyperplan séparateur (droite linéaire rouge), élargir la marge (la bande aux bordures bleues) jusqu'au moment où j'arrive à la frontière de toucher les points .	 *Figure 4 : hyperplan & marge d'un SVM*	

EXPRESSION LITTERALE

<u>Notations</u> : modèle d'apprentissage, espace vectoriel et hyperplan affine

. Il est important de préciser une notation importante au formalisme mathématique du SVM, en particulier avec l'angle d'attaque de construire un modèle d'apprentissage pour faire de la prédiction. Cette notation sera mobilisée à plusieurs reprises lors de la mise en place des démonstrations mathématiques.

(i) Précisions sur les hyperplans et espace vectoriel :

Dans un espace vectoriel de dimension finie n, un hyperplan est un sous-espace vectoriel de dimension $n-1$. Ainsi, dans un espace de dimension 2 un hyperplan sera une droite, dans un espace de dimension 3 un hyperplan sera un plan, etc.

Soit l'espace vectoriel E de dimension n, doté de la base $(e_1, \dots, e_n)$. L'équation caractéristique d'un hyperplan est de la forme $w_1 e_1 + w_2 e_2 + \cdots + w_n e_n = 0$. Par exemple, en dimension 2, munie la base usuelle (x, y), $ax + by = 0$ est bien l'équation caractéristique d'une droite vectorielle (qui passe par l'origine).

Si l'on regarde les choses « avec les mains », un hyperplan sépare complètement l'espace vectoriel en deux parties distinctes. Ainsi, une ligne droite sépare le plan en deux régions distinctes ; et en dimension 3, c'est le plan qui joue le rôle de ce séparateur : il y a un demi-espace « d'un côté » du plan, et un autre demi-espace « de l'autre côté » du plan. Avec un hyperplan, il est donc possible de diviser notre espace vectoriel en deux catégories distinctes : tout pile ce qu'il nous faut pour notre problème de classification !

(ii) W = vecteur de poids et b = le scalaire « appelé le biais » :

Comme vous pouvez le constater, un hyperplan vectoriel passe toujours par 0. C'est pour cette raison qu'on utilisera un **hyperplan affine**, qui n'a pas quant à lui obligation de passer par l'origine.

Ainsi, si l'on se place dans $\mathbb{R}^n$ muni de la base $(e_1, \dots, e_n)$, pendant son entraînement le SVM calculera un hyperplan vectoriel d'équation $w_1 \cdot e_1 + w_2 \cdot e_2 + \cdots + w_n \cdot e_n = 0$, ainsi qu'un scalaire (un nombre réel) b. C'est ce scalaire b qui va nous permettre de travailler avec un hyperplan affine, comme nous allons le voir.

Le vecteur $w = \begin{pmatrix} w_1 \\ \vdots \\ w_n \end{pmatrix}$ est appelé **vecteur de poids**, le scalaire b est appelé **biais**.

Une fois l'entraînement terminé, pour classer une nouvelle entrée $x = \begin{pmatrix} a_1 \\ \vdots \\ a_n \end{pmatrix} \in \mathbb{R}^n$, le SVM regardera le signe de :

$$h(x) = w_1 a_1 + \cdots + w_n a_n + b = \sum_{i=1}^{n} w_i \cdot a_i + b = w^\top \cdot x + b$$

Si $h(x)$ est positif ou nul, alors x est d'un côté de l'hyperplan affine et appartient à la première catégorie, sinon x est de l'autre côté de l'hyperplan, et donc appartient à la seconde catégorie.

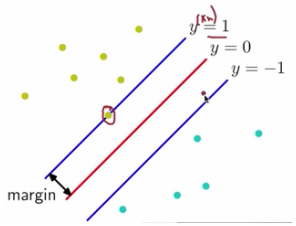
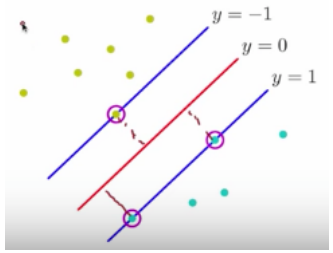
En résumé, on souhaite savoir, pour un point x, s'il se trouve d'un côté ou de l'autre de l'hyperplan. La fonction h nous permet de répondre à cette question, grâce à la classification suivante :

$$\begin{cases} h(x) \geq 0 \implies x \in \text{catégorie 1} \\ h(x) < 0 \implies x \in \text{catégorie 2} \end{cases}$$

Figure 5 : forme du vecteur de poids et hyperplan affine

(source : https://zestedesavoir.com/tutoriels/1760/un-peu-de-machine-learning-avec-les-svm/#3-tet)

Comment maximiser la marge de mon classifieur :

EXPRESSION LITTERALE	FIGURE	COMMENTAIRES
. la distance signée pour 1 exemple (Xn, tn) s'écrit : $\mathbf{tn} * \frac{Y(Xn)}{\lVert W \rVert} = \mathbf{tn} * \frac{(W^T * \phi(Xn) + b)}{\lVert W \rVert}$ **avec (b= biais Wo)** But du SVM : maximiser la marge, cad rechercher la distance signée la plus petite à travers tous les points dans mon ensemble d'apprentissage. $argmax\ (w,b) \left\{ \min n [\mathrm{tn} * \frac{(W^T * \phi(Xn) + b)}{\lVert W \rVert}] \right\}$ donc entraîner un SVM = c'est maximiser la quantité $\{avec\ un\ min\}\ par\ rapport\ à\ mes\ paramètres\ W\ et\ mon\ biais\ b$ $argmax\ (w,b) \left\{ \frac{1}{\lVert W \rVert} \min n\ [\mathrm{tn} * (W^T * \phi(Xn) + b)\ ] \right\}$ **avec Y(Xn) =** $(W^T * \phi(Xn) + b)$ $argmax\ (w,b) \left\{ \frac{1}{\lVert W \rVert} \min n [\mathrm{tn} * (Y(Xn)\ ] \right\}$ = quand j'entraîne mon SVM, je fais une optimisation, i.e. trouver le paramètres (w, b) tel que la marge est la plus grande possible. Nota : on normalise avec une constante a, la marge est la même si on multiplie w et b par une constante (a) tq : $[tn * \frac{(aW^T * \phi(Xn) + ab)}{a \lVert W \rVert}]$. On va donc contraindre la solution pour que $\mathbf{tn} * (W^T * \phi(Xn) + b) = 1$ pour le point (Xn, tn), qui définit ma marge, est le plus proche de la surface de décision. . donc pour le point soit le plus proche de ma surface de décision, **tn*Y(Xn) = 1,** c'est trouver les valeur (w,b) qui satisfait cette équation.	 Graphiquement : . le point (Xn) qui correspond, au point qui est sur ma marge, définissant ma marge qui est le plus proche de ma surface de décision, cela veut dire que pour ce point, **son y = 1 ie Y(Xn) = 1,** Conclusion : je contraints mon problème d'optimisation en respectant cette égalité Y(Xn) = 1. La solution graphique qui maximise la marge de notre figure de départ : ici notre classifieur à marge maximale avec 3 points sur les marges avec une distance = 1 de notre surface de décision (pour les points vecteurs supports)  *Figure 6 : maximiser la marge de mon classifieur*	. contexte de notre problème d'optimisation : paramétrique et linéaire - j'entraîne directement mon entrée, i.e. mon vecteur W ⚠ **pas encore de notion de noyau** - hypothèse 1 : la distance du point le plus proche de mon hyperplan = 1 donc je contraints mes (w,b) pour respecter cette condition. - hypothèse 2 : les points de mon ensemble sont linéairement séparables, cela veut dire que je peux construire une surface de décision (i.e. un hyperplan linéaire) qui sépare distinctement tous les points de la Classe C1 d'un côté et tous les points de la Classe C2 de l'autre côté. => **tous les points sont bien classifiés donc :** **ECRITURE DE MON PRIMALE** $argmax\ (w,b) \left\{ \frac{1}{\lVert W \rVert} \min n [\mathrm{tn} * (Y(Xn)\ ] \right\}$ **avec tn*Y(Xn) = 1** **devient** $=> argmax\ (w,b) \left\{ \frac{1}{\lVert W \rVert} \right\}$ donc maximiser $\frac{1}{\lVert W \rVert}$ c'est la même chose que $argmin\ (w,b) \left\{ \frac{1}{2} * \lVert W \rVert^2 \right\}$ *S.C.* $tn * (W^T * \phi(Xn) + b) >= 1$ *pour n = 1, ..., N (i.e. N contraintes avec N exemples)* **donc mon problème de maximisation devient un problème de minimisation sous contrainte que la distance signée de mon ensemble d'apprentissage est toujours positive et le point le plus proche va avoir une distance = 1, et tous les autres sont moins proches donc de distance >= 1** ***Donc notre problème d'optimisation devient un programme quadratique en complexité $O\ (D^3)$.***

6.1.2.2.2. Maximisation de la Marge : formulation duale et expression du noyau

EXPRESSION LITTERALE	EXPRESSION LITERALE	FIGURE

. en repartant de la formule primale de notre problème **de SVM linéaire**

$$argmin\ (w,b)\left\{\frac{1}{2} * \|W\|^2\right\}$$

S.C.

$$tn * (W^T * \phi(Xn) + b) >= 1\ (N\ contraintes)$$

Solution : je cherche le W avec la norme ||W|| minimale, parmi les classifieurs qui classent correctement avec une marge.

ECRITURE DU PRIMAL AVEC LAGRANGIEN

. On transforme ces contraintes d'inégalité en contrainte d'égalité en passant par un Lagrangien :
- on introduit l'écriture du Lagrangien pour résoudre ce Problème d'optimisation
- on introduit 1 nouvelle variable a_n (= multiplicateur de Lagrange)

$$L\ (w,b,a) = \frac{1}{2} * \|W\|^2 - \sum_{n=1}^{N} an\ \{tn * (W^T * \phi(Xn) + b) - 1\}$$

où les N multiplicateurs de Lagrange sont $a_n >= 0$
(cf. des contraintes d'inégalités)

mais alors avec la disparition des contraintes sur les variables qui me permettent de faire l'optimisation de mon problème (w,b).
Les a_n sont les coefficients du vecteur a [… a_n …] de la représentation duale.

➔ Résolution de mon Problème d'optimisation Lagrangien avec annulation des gradients (W,b) :

$$\nabla L(w,b,a)\ /\ dw = 0 \Leftrightarrow w = \sum_{n=1}^{N} an * \{tn * (\phi(Xn)\}$$

➔ cad W est d'une somme pondérée de mes entrées, i.e. de mes points de mon ensemble d'apprentissage. (ie une combinaison linéaire de mes entrées)

$$\nabla L(w,b,a)\ /\ db = 0 \Leftrightarrow \sum_{n=1}^{N} an * \{tn\} = 0$$

- réécriture de mon Lagrangien $L\ (w,b,a)\ problème\ d'optimisation$ en introduisant la nouvelle forme de w :

ECRITURE DUAL DU SVM

$$L(a) = \sum_{n=1}^{N} an - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} an\ am\{tntm\}k(Xn, Xm)$$

$$avec\ l'astuce\ du\ noyau\ \ k(Xn, Xm) = [\ (\phi(Xn)^T) * \phi(Xm)\ ]$$

$$SC\ \sum_{n=1}^{N} an * \{tn\} = 0$$
$$SC\ an >= 0$$

Pour résoudre mon Pb d'optimisation, c'est toujours de Min $L(a)$ *à (a)*
➔ Donc notre problème d'optimisation devient un programme quadratique en complexité O (N^3) avec N = nombre d'éléments dans mon ensemble d'entraînement… donc résoudre mon SVM va être coûteux à calculer selon mon N
(N = nombre de d'exemples d'apprentissage)

SOLUTION DU DUAL

(la résolution du problème d'optimisation sous contrainte avec le théorème de KKT, **permet d'avoir une solution optimale**)

. les conditions de Karush – Kuhn –Tucker (KKT)
. les vecteurs de support

. la solution de notre problème d'optimisation doit satisfaire les 3 conditions KKT :

(1) $a_n >= 0$
(2) $tn*Y(Xn) - 1 >= 0$
(3) $an\ \{tn * Y(Xn) - 1\} = 0$

avec (3) cela implique que pour chaque n :
. $a_n = 0$ ou $tn * Y(Xn) = 1$

Ce qui implique les résultats suivants :
- cela est vrai uniquement pour le point qui est le plus proche de ma surface de décision,
- pour tous les autres points qui seront bien classifiés cette quantité > 1,
- pour les points qui ne sont pas proches de ma surface de décision, alors nécessairement les a_n = 0. Cela veut dire que dans ma solution ces points ne contribuent pas à ma solution.
Ils ne vont pas influencer la valeur de W (car leurs a_n, le poids dans la somme pondérées des $\phi(Xn)$ qui donnent W, le poids de ces points = 0).

Alors que les points, Xn tels que $a_n > 0$ sont les vecteurs de support.

Notation : **les 3 *Xn* entourés violets tels que a_n > 0 sont appelés les vecteurs de support.** Les autres points entourés rouges n'ont aucune influence sur la solution (**$Y(X) = 0$**)

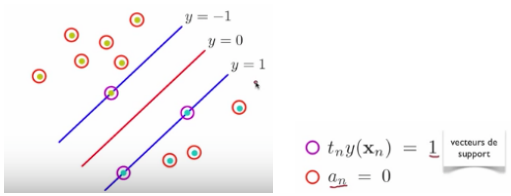


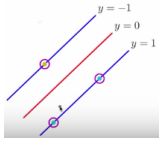
Figure 7 : représentation graphique de vecteurs supports.

****

COMMENTAIRES

Méthode : après avoir étudié l'algorithme pour entraîner un SVM linéaire dans sa version primale, on va étudier la représentation duale pour introduire la notion de noyau. On obtient la version à noyau du SVM et donc *on pourra mobiliser des versions non linéaires du SVM pour résoudre une plus grande classe de problèmes.*

****



→ Cela prouve que si j'avais fait mon SVM seulement avec ces 3 points vecteurs de support, cela aurait donné la même solution (c'est comme si les autres points n'existaient pas). C'est le gros avantage des SVM. Par exemple, pour faire une phase de test en validation croisée, j'aurais juste à entraîner mon classifieur SVM avec ces 3 points vecteurs de supports.

PREDICTION DANS LA REPRESENTATION DU DUAL

. si je fais une prédiction, **l'hyperplan séparateur en fonction de la fonction noyau :**

$$Y(X) = (W^T * \phi(X) + b)$$

[avec X de mon dataset test + b = mon biais]

$$= \left(\sum_{n=1}^{N} an * \{tn * (\phi(Xn)\}\right)^T * \phi(X) + b$$

$$= \left(\sum_{n=1}^{N} an * tn * \{\ k(Xn,\ X)\}\ \right) + b$$ **(ie solution = fonction affine)**

[les a_n > 0, pour les vecteurs supports]
[les cibles tn, sont -1/1, ie l'information de la classe pour chacun des Xn]

Notre SVM à noyaux, c'est comme un algorithme qui fait voter les différents exemples d'apprentissage :
. ceux qui votent, sont les exemples qui sont représentatifs,
. ceux qui sont dans R1/C1 votent positivement,
. ceux qui sont dans R2/C2 votent négativement,

= seulement les vecteurs de support vont voter !!

. Avantage : les SVM linéaire à large bande assure une certaine confiance à bien classifier les futurs points candidats. En 1995, Vladimir Vapnik a prouvé que cette maximisation produit un hyperplan optimal, c'est-à-dire qu'il produit le moins d'erreurs de classification. C'est la ***capacité de généralisation maximale.***

6.1.2.2.3. Les conséquences

Il y a trois conséquences importantes à formaliser par l'application de la 3éme <u>*conditions de Karush- Kuhn-Tucker*</u> *définie comme suit selon* $\boldsymbol{a_n * \{tn * Y(Xn) - 1\} = 0}$ ***:***

- *1ère conséquence, les seuls points pour lesquels les contraintes du lagrangien sont actives sont donc les points, tels que* $\boldsymbol{tn * Y(Xn) = 1}$*, qui sont les points situés sur les hyperplans de marges maximales. En d'autres termes, seuls les vecteurs supports participent à la définition de l'hyperplan optimal.*

- *La seconde conséquence découle de la première. Seul un sous-ensemble restreint de points est nécessaire pour le calcul de la solution optimale du problème d'optimisation quadratique, les autres points de l'apprentissage ne participant pas du tout à sa définition. Ceci est donc efficace au niveau de la* <u>*complexité*</u>*. D'autre part, le changement ou l'agrandissement de l'ensemble d'apprentissage a moins d'influence que dans un* <u>*modèle de mélanges gaussiens*</u> *(ie EM), par exemple, où tous les points participent à la solution. En particulier, le fait d'ajouter des points à l'ensemble d'apprentissage qui ne sont pas des vecteurs supports n'a aucune influence sur la solution finale.*

- *La 3ème conséquence est que l'hyperplan solution optimale ne dépend que* <u>*du produit scalaire*</u> *entre le vecteur d'entrée* $\boldsymbol{(X)}$ *et les vecteurs supports* $\boldsymbol{Y(X) = 0}$*. Cette remarque est l'origine de la deuxième innovation majeure des SVM : le passage par un espace de redescription grâce à une fonction noyau. L'utilisation d'un noyau défini* ***comme le produit scalaire des transformations de nos données****, va faciliter les calculs puisque nous n'avons pas besoin d'évaluer le produit scalaire dans l'espace de départ.*

LA SOLUTION DE L'HYPERPLAN OPTIMAL AVEC LA VERSION A NOYAU
DANS LA PRESENTATION DUALE DU PROBLEME D'OPTIMISATION QUADRATIQUE SVM

$$\boldsymbol{Y(X) = \left(W^T * \phi(X) + b\right) = \left(\sum_{n=1}^{N} an * \{tn * (\phi(Xn)\})^T * \phi(X) + b = \left(\sum_{n=1}^{N} an * tn * \{k(Xn,\ X)\}\right) + b\right)}$$

$$\boldsymbol{avec\ l'astuce\ du\ noyau\ \ k(Xn, Xm) = [\left(\phi(Xn)^T\right) * \phi(Xm)]}$$

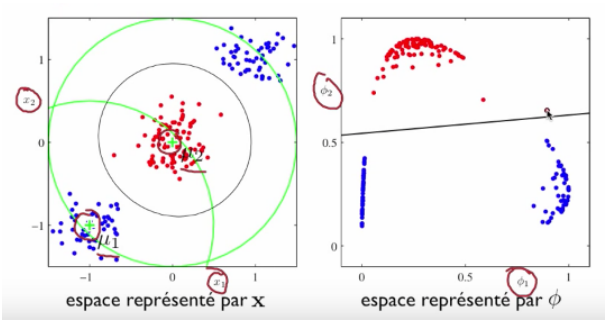
(source : https://fr.wikipedia.org/wiki/Machine_%C3%A0_vecteurs_de_support#Cons.C3.A9quences)

6.1.2.3. Dans le cas non séparable

6.1.2.3.1. Principe du Kernel trick

Présentation de la façon de convertir un algorithme linéaire en algorithme non linéaire: Cas n° 1, problème de classification binaire supervisée

(ou convertir un modèle non linéaire à l'aide de la méthode à noyaux)

EXPRESSION LITTERALE	FIGURE	COMMENTAIRES
MOTIVATION des Méthodes à noyaux : . comment transformer un problème non linéaire en problème linéaire de plus grande dimension ? . comment convertir des algorithmes linéaires en algorithmes non linéaires avec des transformations simples ? Hypothèse : partons d'un problème de classification non linéaire défini comme suit. Soit un problème de classification bi-classes mais qui n'est pas résolvable par un classifieur linéaire en 2 dimensions (cf. figure). - 2 classes identifiées : les points rouges et les points bleus. - vue la configuration en cercle des points, avec les points rouges au centre du cercle, il n'existe pas d'hyperplan linéaire capable de bien séparer les points en 2 classes distinctes. => la solution : on tranforme le classifieur en utilisant 2 fonctions de bases $en\ gaussiennes\ \emptyset j$ telles que, $\emptyset j = \exp\left\{-\frac{(x-uj)^2}{2s^2}\right\}$ Sous forme vectorielle, $\boldsymbol{exp}\ \{-\|\boldsymbol{X}-\boldsymbol{Uj}\|^2\ /\ (\boldsymbol{2s^2})\}$ = la distance euclidienne au carré entre le vecteur X de points et un vecteur de moyennes des précédents Points. Puis, je fixe sur ma figure u1 sur la classe des points bleus, et u2 sur la classe des points rouges. Alors, j'obtiens 1 second système de représentation Ø1 et Ø2 où il possible de passer une droite qui sépare correctement les 2 classes.	 *Figure 8 : transformer un classifieur linéaire en classifieur non linéaire avec des 2 fonctions de bases gaussiennes Øj* Conclusion : avec 2 fonctions de bases, on réussit à transformer un problème de classification non linéaire en problème de classification linéaire. **L'objectif des méthodes à noyaux est d'incorporer et d'utiliser plus facilement ces fonctions de bases non-linéaires dans un modèle linéaire.**	En conséquence : . les fonctions de bases vont être définies implicitement. Cela veut dire de représenter explicitement en mémoire $\emptyset(\mathrm{x})$. . cela veut dire que la matrice $\emptyset[..]$ qu'on aurait à construire avec chacun des vecteur $\emptyset(x)$ pour chacune de mes entrées de mon ensemble de points X, on ne va jamais avoir à la construire explicitement. Elle va être implicitement utilisée. C'est à dire les représentations des vecteurs $\emptyset(x)$ vont être implicites. . il suffit simplement de calculer une comparaison K(X, X') entre les entrées X et X', cela veut dire qu'on construit **une distance de similarité** entre mes 2 entrées X et X'. . alors dans ce cadre formel, **la fonction K(X, X') est appelée un noyau / ou fonction à noyau** et **les algorithmes utilisant un noyau sont appelés méthodes à noyaux.** **Alors ces algorithmes linéaires sont convertis en forme non linéaire.**

Présentation de la façon de convertir un algorithme linéaire en algorithme non linéaire : Cas n° 2, problème de régression

EXPRESSION LITTERALE

MOTIVATION :
. on souhaite construire la version à noyaux de la régression.
. en particulier en utilisant la représentation duale de la version à noyaux de la régression.

ECRITURE PRIMALE DU PROBLEME DE REGRESSION

Expression du modèle linéaire de la régression :
. ma prédiction se définit (1) Y(X) = $\boldsymbol{W^T} * \boldsymbol{\phi(Xn)}$
. puis je cherche à minimiser la somme des écarts au carré de ma prédiction à la cible connue $\boldsymbol{tn}$ plus un terme de normalisation (i.e. avoir un certain contrôle sur la capacité de mon modèle). On parle alors de régression régularisée définie telle que, c'est à dire sous forme d'un problème d'apprentissage :

(2) **J(w) = ½ *** $\sum_{n=1}^{N}\{\ \boldsymbol{W^T} * \boldsymbol{\phi(Xn)} - \boldsymbol{tn}\}$^2 **+** $\frac{\lambda}{2} * \boldsymbol{W^T} * \boldsymbol{W}$

On cherche alors w qui va minimiser ce problème d'apprentissage :
∇J(w, tn) / dw *= 0 (annulation du gradient en W)*
$\frac{\nabla J(w,tn)}{dw} = \sum_{n=1}^{N}\{\ W^T * \phi(Xn) - tn\} + \lambda W = 0$

alors la solution optimale
w = - 1/ λ * $\sum_{n=1}^{N}\{\boldsymbol{W^T} * \boldsymbol{\phi(Xn)} - \boldsymbol{tn}\} * \boldsymbol{\phi(Xn)}$

w = $\sum_{n=1}^{N} \boldsymbol{an} * \boldsymbol{\phi(Xn)}$

c.à.d. w la solution de mon problème d'apprentissage est égale à une combinaison linéaire de mes entrées $\boldsymbol{Xn}$ *de mon ensemble d'apprentissage représentées selon les fonctions de bases (c'est prendre chacune de mes entrées, la multiplier par un scalaire qui détermine à quel poids cette entrée contribue à ma solution finale avec une accumulation pondérée via les* $\boldsymbol{an}$)

w = $\boldsymbol{\phi^T} * \boldsymbol{a}$ avec a = $(a1, \ldots, aN)^T$ avec longueur de vecteur a = N
où chaque an est la contribution de $\boldsymbol{\phi(Xn)}$ **à la solution .**
où N = le nombre de points de mon ensemble d'apprentissage.

ECRITURE DUALE DU PROBLEME DE REGRESSION

Postulat : cherchons a comme solution optimale au problème d'optimisation, alors (2) peut s'écrire sous la forme duale tq
(3) **J(a) = 1/2**$(\boldsymbol{a^T\phi\phi^T\phi\phi^Ta}) - (\boldsymbol{a^T\phi\phi^Tt}) + (\frac{1}{2}\boldsymbol{t^Tt}) + \frac{\lambda}{2}(\boldsymbol{a^T\phi\phi^Ta})$
Avec J(a) représentation duale de ma fonction primale J(w)

FIGURE

. Définition de la matrice de Gram : **ou matrice noyau symétrique**
dans l'expression duale du problème de régression je remplace l'expression produit de 2 matrices selon $\boldsymbol{\phi\phi^T} = \boldsymbol{K}$

avec K matrice de Gram contient tous les **Knm = k(Xn, Xm) de mon ensemble d'apprentissage.**

cad : la matrice de Gram contient chacun des produits scalaires entre les vecteurs $\boldsymbol{\phi}$ pour mes différents points d'apprentissage avec la notation suivante

$$\boldsymbol{\phi(Xn)^T} * \boldsymbol{\phi(Xm)} = \mathbf{k(Xn, Xm)} = \mathbf{Knm}$$

et avec $\mathbf{k(Xn, Xm)} = \mathbf{fonction\ noyau}$

donc (3) devient : **J(a) = 1/2**$(\boldsymbol{a^TKKa}) - (\boldsymbol{a^TKt}) + (\frac{1}{2}\boldsymbol{t^Tt}) + \frac{\lambda}{2}(\boldsymbol{a^TKa})$

objectif : on cherche alors a qui va minimiser cette fonction de coût J(a), avec calcul des gradients par rapport (a) tel que,

∇J(a,t) / da = 0 ⇔ a = (K + λIN)^-1 *t
avec taille matrice a = (N x N)

on retrouve la forme habituelle de la solution régression linéaire :
$\hat{\beta} = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{Y}.$ *(source : wikipedia.org)*

avec :
. K matrice de Gram (matrice noyau),
. λ = paramètre de régularisation,
. I matrice Identité de taille (N x N),
. t = vecteur qui contient toutes les cibles connues de mon ensemble d'apprentissage.

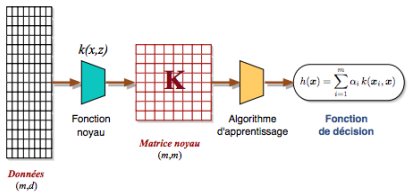


Figure 9 : l'importance de la matrice de Gram **

COMMENTAIRES

PREDICTION DANS LA REPRESENTATION DU DUAL

Donc (1) peut s'écrire sous la forme

Y(X) = $\boldsymbol{W^T} * \boldsymbol{\phi(Xn)} = (\boldsymbol{\phi^T} * \boldsymbol{a})^T * \boldsymbol{\phi(Xn)} = \boldsymbol{a^T}\ [\boldsymbol{\phi}] * \boldsymbol{\phi(Xn)}$
où $[\phi] = matrice;\ \phi(Xn) = vecteur\ de\ mes\ entrées$
où $[\boldsymbol{\phi}] * \boldsymbol{\phi(Xn)}$ = ***K(Xn)*** ***(Xn exprimé en fonction de base)***

(4) **Y(X) =** $[\boldsymbol{K(Xn)}]^T * (\boldsymbol{K} + \lambda \boldsymbol{I_N})$**^-1 *t**

où $\boldsymbol{K(Xn)} = (\boldsymbol{K(X1, X)}, \ldots, \boldsymbol{K(XN, X)})$

et où dans (4)
. s'exprime uniquement avec le noyau K, et $\boldsymbol{\phi(Xn)}$ n'apparaît plus.
. je suppose que ma fonction noyau K fait bien les produits scalaires des représentations des vecteurs $\boldsymbol{\phi(Xn)}$à partir de mes fonction de bases $\boldsymbol{\phi}(..)$

=> on appelle **Y(X) =** $[\boldsymbol{K(Xn)}]^T * (\boldsymbol{K} + \lambda \boldsymbol{I_N})$**^-1 *t**
la représentation duale à noyaux de la régression,
une forme générique à fonction noyau, où je peux changer facilement la forme de mon noyau.

<u>En résumé, la régression à noyau se définit comme suit :</u>

(i) sur mes entrées d'apprentissage, je calcule le vecteur (a) selon l'expression algèbre linéaire : ***a = (K + λIN)^-1 *t***

(ii) pour faire ma prédiction, je calcule : ***Y(X) = K(X)*** $^T * \boldsymbol{a}$
où le vecteur K(X) = où j'applique ma fonction noyau entre mon entrée X et chacune de mes entrées de mon ensemble d'apprentissage.

Avec l'astuce du noyau K(Xn, Xm) = au produit scalaire entre mes fonctions de bases qui sont en grande dimension.

Remarque : pour calculer numériquement K(X) je dois garder en mémoire tous mes données de mon ensemble d'apprentissage. (attention à la taille des données sur des problèmes de data sciences).

** source : Antoine Cornuéjols / cours Master ISI / CH14 : méthodes à noyaux

Astuce du noyau

EXPRESSION LITTERALE

. L'astuce du noyau est vraiment au centre des méthodes à noyau et qui permet d'utiliser le SVM à noyau.

L'idée : certains noyaux, implicitement, avec leur représentation en fonction de base va être en haute dimension, très hautement non linéaire, mais le calcul direct du noyau va être plus efficace que les calculs du produit scalaire des représentations en fonction de base.

. L'astuce du noyau, permet de définir des noyaux tels que : calculer K(Xn, Xm) est plus efficace que de calculer (1) les vecteurs $\boldsymbol{\phi(Xn)}$ ***et*** $\boldsymbol{\phi(Xm)}$ et ensuite faire (2) de multiplier les vecteurs $\boldsymbol{\phi(Xn)^T * \phi(Xm)}$ **en produit scalaire ?**
. on n'a pas besoin de $\boldsymbol{\phi}(..)$ explicitement,
. on a besoin des faire des produits scalaires entre les vecteurs $\boldsymbol{\phi}(..)$

. Démonstration : pour (dim = 2)

Postulat : sans passer par les représentations en fonction de base, j'utilise mon noyau et je fais simplement le produit scalaire de mes vecteur X[1,2] x Z[2,1] comme suit : **k(x, z) =** $\boldsymbol{(x^T * z)^2 = (x1z1 + x2z2)^2}$
[*coût : 3 mult. + 1 addition*]

→ je développe le $(..)^2$ puis je l'exprime au travers des représentations en fonction de base $\phi(X)$

= $x1^2\ z1^2$ + 2x1z1x2z2 + $x2^2\ z2^2$ [*somme de produits scalaires*]
= $(x1^2, \sqrt{2x1x2},\ x2^2)\ (z1^2, \sqrt{2z1z2},\ z2^2)^T$[*produits scalaires 2 vecteurs*]

= $\boldsymbol{\phi(X)^T * \phi(Z)}$ [*produits 2 vecteurs en fonction de base* $\boldsymbol{\phi(X)}$]

[*coût : 2x 4 multiplications,*
+ construire $\boldsymbol{\phi(X)}$ ***et*** $\boldsymbol{\phi(Z)}$
+ suivi de 3 multiplications et 2 additions (produit scalaire)]

Commentaires :
. *le calcule en fonction explicite :* $\phi(X)^T * \phi(Z)$
il coûte 11 multiplications + 2 additions

. *le calcule en passant par le noyau* : k(x, z) = $(x1z1 + x2z2)^2$
il coûte 3 multiplications + 1 addition

Conclusion : pour résumé l'astuce du noyau
. il y a des noyaux tels que le produit scalaire 2 vecteurs X,Z est moins coûteux que de construire la représentation explicite en $\boldsymbol{\phi(X)}$,
. à l'inverse il y a des noyaux, implicitement, c'est le produit des $\phi(X)$, je peux l'utiliser dans la formulation duale d'une régression, cad la formulation duale d'une méthode noyau.

FIGURE

. Dans notre exemple, nous avons pris un noyau polynomial de degré = 2 écrit sous la forme générique **K(x,z) =** $\boldsymbol{< x, z >^2}$**, il est défini positif.**

. Définition : un noyau k est dit positif, si pour tout entier n fini et pour toute suite de n observations, pour {xi = 1, .., n}, la matrice noyau associée au noyau k est une matrice symétrique défini positive.

. Propriété : des noyaux définis positifs
- ces noyaux mobilisent des produis scalaires,
- ils facilitent les calculs puisque le produit scalaire n'est pas réalisé dans l'espace de départ.

En résumé : une fonction noyau est associée à un ou plusieurs espaces de redescription muni(s) d'un produit scalaire. Une condition nécessaire et suffisante pour qu'une fonction K(x, x') : X × X → $\mathbb{R}$ soit une fonction noyau est qu'elle soit semi-définie positive ou qu'elle vérifie la condition de Mercer.

Les fonctions noyaux ont la propriété que toute matrice de Gram associée de dimension finie est semi-définie positive. Le théorème de Mercer est une formulation équivalente de cette propriété pour les espaces vectoriels.

. Le théorème de Mercer (14.2) fournit en plus une description explicite de l'espace de redescription par une base de fonctions analysantes orthogonales. Ce n'est cependant pas la seule base de fonctions analysantes possibles. Une autre base (qui peut parfois coïncider avec la base de Mercer) est donnée par les noyaux reproduisants.

***

2. Philosophie des méthodes à noyaux

2.1 Les méthodes linéaires remises au goût du jour

La section 1.2 a fourni une première illustration de l'utilisation de fonctions noyau comme moyen de transformer la recherche de régularités non-linéaires en recherche de régularités linéaires grâce au passage par un espace de redescription virtuel F (voir la figure 14.1). **Quatre grandes idées** sous-tendent l'approche générale :

1. Les données décrites dans l'espace d'entrée $\mathcal{X}$ sont projetées dans un espace vectoriel de redescription F
2. Des régularités linéaires sont cherchées dans cet espace F
3. Les algorithmes de recherche n'ont pas besoin de connaître les coordonnées des projections des données dans F, mais seulement leurs produits scalaires
4. Ces produits scalaires peuvent être calculés efficacement grâce à l'utilisation de fonctions noyau.

(Source : page 440 - Antoine Cornuéjols / cours Master ISI / CH14 : méthodes à noyaux)

Figure 10 : Extrait théorème de Mercer, valeurs & vecteurs propres

COMMENTAIRES

Théorème 14.2 (Théorème de Mercer)

Si $\kappa(\cdot,\cdot)$ est une fonction noyau continue symétrique d'un opérateur intégral

$$g(\mathbf{y}) \;=\; A\,f(\mathbf{y}) \;=\; \int_a^b \kappa(x,y)\,f(y)\,dy + h(x)$$

vérifiant :

$$\int_{\mathcal{X}\times\mathcal{X}} \kappa(\mathbf{x},\mathbf{x}')\,f(\mathbf{x})\,f(\mathbf{x}')\,d\mathbf{x}d\mathbf{x}' \;\geq 0$$

pour toute fonction $f \in L_2(\mathcal{X})$ (de carré sommable) ($\mathcal{X}$ étant un sous-espace compact de $\mathbb{R}^d$), alors la fonction $\kappa(\cdot,\cdot)$ peut être développée en une série uniformément convergente en fonction des valeurs propres positives λ_i et des fonctions propres ψ_i :

$$\kappa(\mathbf{x},\mathbf{x}') \;=\; \sum_{j=1}^{N} \lambda_j\,\psi_j(\mathbf{x})\,\psi_j(\mathbf{x}') \qquad (14.10)$$

où N est le nombre de valeurs propres positives (nombre éventuellement infini).

On peut alors décrire la fonction Φ de redescription des entrées comme :

$$\Phi(\mathbf{x}) \;=\; (\sqrt{\lambda_1}\,\psi_1(\mathbf{x}),\ \sqrt{\lambda_2}\,\psi_2(\mathbf{x}),\ldots) \qquad (14.11)$$

Cette décomposition est l'analogue en dimension infinie de la diagonalisation d'une matrice Hermitienne. Le taux de décroissance des valeurs propres λ_i fournit une information sur la régularité du noyau. Plus cette décroissance est rapide, plus régulière est la fonction noyau.

(Source : page 442 - Antoine Cornuéjols / cours Master ISI / CH14 : méthodes à noyaux)

***

FORME GENERALE D'UN NOYAU POLYNOMIAL

. Il existe une forme générale pour notre noyau exemple **k(x, z) =** $\boldsymbol{(x^T * z)^2}$, c'est le Φ **noyau polynomial** défini tq : **k(x, x') =** $\boldsymbol{(x^T * x' + c)^M}$**,**
- (1) je fais le produit scalaire de mes 2 vecteurs,
- (2) je rajoute une constance c > 0,
- (3) je mets à la puissance M la somme (1) + (2)

. où **M** est un hyper-paramètre, détermine le degré des produits que j'ai dans mon vecteur,
. on peut démontrer implicitement que l'expression de mon noyau $\boldsymbol{(x^T * x' + c)^M}$ **est le produit scalaire des représentations** $\boldsymbol{\phi(X)}$ **qui contient tous les produits possibles entre les éléments de X.** Cela me construit une base polynomiale, pour par exemple faire de la régression polynomiale.

$\boldsymbol{\phi(X)}$ implicite contient tous les produits possibles entre au plus M éléments de X : (avec D = dimension de mon espace)

$\boldsymbol{\phi(X) = (c0, c1x1,\ ..,cDxD, c11\ x1^2, c12x1x2, ..,}$
$\boldsymbol{...\ pour\ l'ordre\ 3,\ ..jusqu'à\ l'ordre\ M}$) cela revient à faire un produire scalaire de vecteur de grande dimension (> M).

Nota : si c = 0 et M = 1 (degré) = noyau linéaire

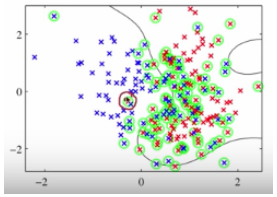
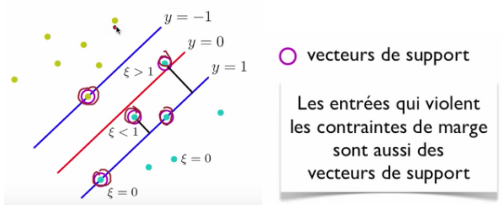
EXPRESSION LITTERALE	FIGURE
Avantage de passer par la méthode à noyaux pour calculer la solution d'une régression polynomiale (non linéaire) . Illustration de l'avantage à passer par la méthode noyau pour déterminer la solution d'une régression polynomiale : . **soit un modèle de régression avec un degré = 3 défini tq :** y(x, w) = w0 + $\sum_{i=1}^{D} wixi$ + $\sum_{i=1}^{D} \sum_{j=1}^{D} \quad wij\ xi\ xj$ + $\sum_{i=1}^{D} \sum_{j=1}^{D} \sum_{k=1}^{D} \quad wijk\ xi\ xj\ xk$. pour M = 3, on a 1 + D + D^2 + D^3 paramètres à apprendre (pour seulement une régression d'ordre 3) . en généralisant : on augmente le nombre de paramètres selon la complexité **O($\boldsymbol{D^M}$)** . . pour D = 100, M = 3 > 1 millions de paramètres … . **cela veut dire que le nombre de paramètre w à apprendre dans un problème d'apprentissage est exponentiel dans le degré de ma fonction polynomiale que je vais apprendre**. . en passant par la méthode des noyaux, je n'ai plus de paramètre w à apprendre explicitement, donc je n'ai plus ce problème de dimensionnalité des paramètres à apprendre. Ma fonction à noyau contient implicitement tous les produits polynomiaux que j'ai besoin pour construire la solution de ma régression polynomiale. **AVANTAGE : Ma fonction à noyau me permet de résoudre un problème d'apprentissage, avec $\phi(X)$ grand (car associé à un problème de départ en grande dimension) sans manipuler des vecteurs de paramètre à apprendre de grande dimension.**	**MALEDICTION DE LA DIMENSIONNALITE** . **<u>MAIS attention</u>** à la dimensionnalité de mon espace d'arrivée avec un problème au départ de grand dimension (quelque soit, implicitement en noyau ou explicitement avec en fonction de base $\boldsymbol{\phi}(\mathbf{X})$) . c'est à dire, je dois avoir des exemples d'apprentissage proches de toutes les valeurs de $\boldsymbol{\phi}(\mathbf{X})$ que je pourrais observer dans mon ensemble d'apprentissage. C'est à dire si j'ai un noyau associé à problème de départ de grande dimension, plus la dimensionnalité est élevée, plus je peux avoir des problèmes de sur-apprentissage (mon modèle, pas défaut de données, colle trop aux données de mon dataset d'apprentissage et mon modèle va mal généraliser en test). C'est à dire, la dimension (D) de mes entrées de mon ensemble d'apprentissage peut empêcher à bien généraliser les résultats de mon modèle obtenu sur l'ensemble d'apprentissage, En d'autres termes, la quantité de données nécessaires pour bien généraliser n'est pas fournie par une taille insuffisante des données de l'échantillon d'apprentissage. ➔ <u>Conclusion</u> : je risque d'avoir des problèmes de sur-apprentissage lorsque la dimensionnalité de $\boldsymbol{\phi}(\boldsymbol{X})$ **augmente.**

6.1.2.3.2. Choisir son noyau

EXPRESSION LITTERALE	FIGURE	COMMENTAIRES
L'objectif : comment construire une fonction noyau pour s'adapter aux différents problèmes d'apprentissage classification ou régression, c'est à dire exploiter la connaissance a priori du problème pour construire des noyaux plus adaptés. L'astuce du noyau : c'est calculé le produit scalaire des représentations des fonctions de bases $\boldsymbol{\phi(X)}$ que je n'instancie pas numériquement, elles sont implicites. Preuve : quand on utilise le noyau polynomial (***modèle non linéaire***) **k(x, x') =** $(\boldsymbol{x}^T * \boldsymbol{x}' + \boldsymbol{c})^M$ on vérifie qu'il est valide car on a démontré que implicitement à ce noyau, il existe bien derrière ce noyau un produit scalaire de représentations de fonctions de bases $\boldsymbol{\phi(X)}^T * \boldsymbol{\phi(X')}$ Conséquence : Pour travailler avec une méthode à noyau, il faut s'assurer que le noyau est valide, c'est à dire qu'il y a bien derrière un produit implicite de vecteurs $\boldsymbol{\phi(X)}^T * \boldsymbol{\phi(X')}$ **. Cela veut dire trouver une fonction qu'on pourrait écrire sous la forme d'un produit en 2 vecteurs** $\boldsymbol{\phi(X)}$**.** Ce qui est difficile à faire « from scratch ». Il est plus facile de partir de méthodes à noyaux existantes et d'y associer des règles de constructions de noyaux. [ces règles sont définies telles qu'elles permettent de maintenir la validité de ma fonction noyau de base] **Règles de compositions de noyaux à partir de noyaux définis valides : DERIVER DES NOYAUX (Forme Fonctionnelle)** Où c>0, f(x) est une fonction, q(a) un polynôme de coefficients positifs, A une matrice définie positive, x = (xa, xb) avec k1, k2, k3, Ka et Kb noyaux vérifiés valides. 1. **k(x,x') =** $\boldsymbol{c * k1(x, x')}$ noyau valide x cste → noyau valide 2. **k(x,x') =** $\boldsymbol{f(x)\ k1(x, x')\ \ f(x')}$ noyau valide x à gauche par 1 fonction quelconque (entrée X) puis à droite par 1 fonction quelconque (X') → noyau valide 3. **k(x,x') = q(k1(x,x'))** je prends un noyau valide et sa sortie K1 que je passe dans 1 fonction q polynomiale à coefficients > 0 → noyau valide	4. **k(x,x') = exp(k1(x,x'))** je prends un noyau valide et sa sortie K1 que je passe dans 1 fonction exponentielle → noyau valide 5. **k(x,x') = k1(x,x')) + k2(x,x'))** la somme de 2 noyaux valides différents → noyau valide 6. **k(x,x') = k1(x,x')) k2(x,x'))** la produit de 2 noyaux valides différents → noyau valide 7. **k(x,x') = k3(**$\boldsymbol{\phi(X)}$ **,** $\boldsymbol{\phi(X')}$**)** je passe à un noyau valide 2 fonctions de représentation de bases $\boldsymbol{\phi(X)}$ ***et*** $\boldsymbol{\phi(X')}$**)** → noyau valide *application : j'ai des entrées X auxquelles j'applique un prétraitement, je passe cela dans un noyau valide, alors cela donne un noyau valide.* 8. **k(x,x') =** $\boldsymbol{(X)^T A(X')}$ pas de noyau valide, mais cette expression algébrique → noyau valide [par exemple, $\boldsymbol{A = matrice\ identité\ I}$**, alors k(x,x') =** $\boldsymbol{(X)^T(X')}$ = noyau valide, c'est comme avoir $\boldsymbol{\phi(X) = X}$ 9. **k(x,x') = ka(xa,xa')) + kb(xb,xb'))** dans le cas x = (xa, xb), x séparé en 2 parties, les 1er éléments dans xa et les derniers éléments dans xb, si j'ai 2 noyaux appliqués sous chaque sous-partie, puis la somme des 2 noyaux → noyau valide *application : dans le cas où x est composé de données hétérogènes.* 10. **k(x,x') = ka(xa,xa'))kb(xb,xb'))** idem que 9. La somme est remplacée par le produit des noyaux appliqués à des 2 sous-parties des données de x. *** démonstration *** k(x,x') = $c * k1(x, x')$ avec $k1(x, x') = \phi^1(X)^T * \phi^1$ (X') $c * k1(x, x') = c\ \phi^1(X)^T * \phi^1$ (X') = $(\sqrt{c}\ \phi^1(X))^T\ (\sqrt{c}\phi^1(X'))^T$ = $\phi(X)^T * \phi(X')$ → $\boldsymbol{c * k1(x, x')}$ ***est bien un noyau valide*** *Figure 11 : fonctions noyaux dérivées valides (10 exemples)*	• **② Noyau Gaussien (RBF = Radial Basis function)** est un noyau valide : **k(x,x') =** $\boldsymbol{exp}\ \{-\|\boldsymbol{X} - \boldsymbol{X}'\|^2 / (2\sigma^2)\}$ ie : expontielle de la distance euclidienne entre les 2 vecteurs X et X', divisée par l'hyper-paramètre σ qui détermine la sensibilité de mon noyau $\|\boldsymbol{X} - \boldsymbol{X}'\|^2 = (\boldsymbol{X} - \boldsymbol{X}')^T * (\boldsymbol{X} - \boldsymbol{X}') = \boldsymbol{X}^T\boldsymbol{X} + (\boldsymbol{X}')^T\boldsymbol{X}' - 2\boldsymbol{X}^T\boldsymbol{X}'$ **k(x,x') =** $\boldsymbol{exp}\ \{\ -\boldsymbol{X}^T\boldsymbol{X}/(2\sigma^2)\}$ k(x,x') = $\boldsymbol{f(x)\ k1(x, x')^M f(x')}$ $* \boldsymbol{exp}\ \{\ \boldsymbol{X}^T\boldsymbol{X}'/(2\sigma^2)\}$ → noyau valide $* \boldsymbol{exp}\ \{\ -(\boldsymbol{X}')^T\boldsymbol{X}'/(2\sigma^2)\}$ k(x,x') = $\boldsymbol{f(x)\ k1(x, x')^M f(x')}$ **Grâce aux règles de compositions 1. , 2 et 4. ,** le Noyau Gaussien (RBF = Radial Basis function) est un noyau valide. Avec : $\boldsymbol{\phi(X)}$ ***vecteur de grande dimension.*** **DERIVER DES NOYAUX : autres façons de dériver des noyaux** **(i) On peut également construire des noyaux qui ne sont pas des vecteurs x de taille fixe.** - des chaînes de caractères (en bioinformatique) - des ensembles de vecteurs, **(ii) On peut mobiliser le noyau de Fisher :** idée : dériver de nouveaux noyaux à partir de modèles probabilistes génératifs. **5.3.8 Les noyaux génératifs** Il arrive que l'on dispose d'information sur le processus générant les données. Il peut alors être intéressant d'en tirer parti pour définir des fonctions noyau mieux adaptées à la tâche considérée. Plusieurs approches ont été proposées. Nous décrivons brièvement ici les fonctions noyau dérivées de la matrice d'information de Fisher[12]. L'idée sous-jacente dans cette approche est de mesurer l'effort d'adaptation nécessaire du modèle existant pour accommoder la donnée considérée. En particulier, il est nécessaire de disposer d'un modèle de génération des données qui soit paramétré continûment de telle manière que les dérivées par rapport à ces paramètres puissent être calculées. **Définition 14.10 (Noyau de Fisher)** *Le* noyau de Fisher *par rapport au modèle génératif de vecteur de paramètre* θ^0 *est :* $\kappa(\mathbf{x}, \mathbf{x}') = \mathbf{g}(\theta^0, \mathbf{x})^\top \mathbf{I}^{-1} \mathbf{g}(\theta^0, \mathbf{x}')$ (Source : ingénierie des fonctions noyaux- page 479 - Antoine Cornuéjols / cours Master ISI / CH14 : méthodes à noyaux) - Noyau à connaître : **③ Noyau sigmoïde = k(x,x') =** $\boldsymbol{tanh}(\gamma * < \boldsymbol{x}, \boldsymbol{x}' > + \boldsymbol{c})$

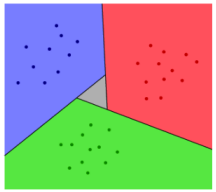
6.1.3. Extensions

6.1.3.1. Le cas de la séparation imparfaite : Marge Souple « Soft Margin »

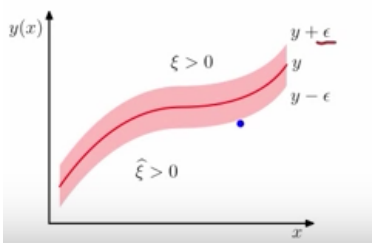
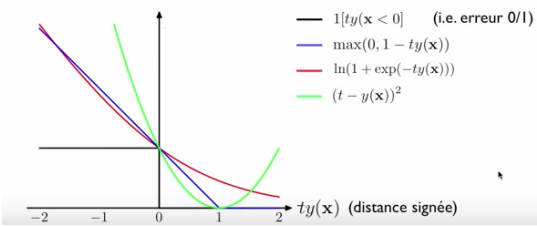
EXPRESSION LITTERALE	FIGURE	COMMENTAIRES
Hypothèse : on relâche l'hypothèse forte que notre problème de classification est linéairement séparable. . on ne peut classifier parfaitement l'ensemble d'apprentissage, . on vérifie des chevauchements entre les 2 classes, on se permet des erreurs sur l'ensemble d'apprentissage. Point de départ : formalisation du cas séparable, [ECRITURE DE MON PRIMALE] On maximise la marge { … } tq (1) $argmax\,(w,b)\left\{\frac{1}{\|W\|}\min n\,[\mathrm{tn} * (W^T * \phi(Xn)+b)\,]\right\}$ avec $\min n\,[\mathrm{tn} * (W^T * \phi(Xn)+b)\,] = 1$ **DEVIENT** (2) $argmin\,(w,b)\left\{\frac{1}{2} * \|W\|^2\right\}$ *S.C.* $tn * (W^T * \phi(Xn)+b) >= 1$ *ET pour n = 1, … , N* Avec : . (1) la cible = tn . (2) la sortie de ma fonction discriminante = $(W^T * \phi(Xn)+b)$. donc (1) & (2) sont du même signe, . donc tous les points de mon ensemble d'apprentissage sont classifiés dans la bonne classe, et je trouve mon w avec sa norme la plus petite, parmi tous les classifieurs qui classifient correctement avec une marge > =1. QUE FAIRE : dans 2 cas souvent rencontrés en « machine learning » . des erreurs dans l'ensemble d'apprentissage, . des exemples exceptionnellement difficiles à classifier. Solution : on va permettre au SVM sur certains points apprentissage de ne pas respecter la contrainte de marge. **DEVIENT** (3) $argmin\,(w,b,\xi n)\left\{\frac{1}{2} * \|W\|^2 + C\sum_{n=1}^{N}\xi n\right\}$ *S.C.* $tn * (W^T * \phi(Xn)+b) >= 1 - \xi n$ $\xi n \geq 0$ *pour n = 1, … , N* on introduit des ξn (Xi-n) sont des **variables de ressort** (slack variable), elles correspondent aux violations des contraintes de marge. (1 var ξn pour chaque exemple de mon dataset d'apprentissage). Je pénalise le fait de violer les contraintes de valeur de marge >=1, cad je pénalise et minimise la somme de ces violations de contraintes (+ une pondération C dans ma minimisation = hyper-paramètre) pour que SVM classe bien.	Comment permettre à notre SVM de faire des erreurs ?  présence d'un point x rouge parmi les points bleus = erreur ? ** **Formulé ainsi le classifieur SVM arbitre le meilleur compromis possible entre maximisation de la marge et minimisation de l'erreur d'apprentissage.** . si $\xi n \in [0,1]$ alors le point exemple de mon apprentissages est bien classé (il se trouve du bon côté de la surface de décision) . si $\xi n > 1$, alors le point exemple de mon apprentissages est mal classé (il se trouve du mauvais côté de la surface de décision) = cela veut dire que j'ai accepté de faire une erreur sur ce point de mon ensemble d'apprentissage. . la constance C > 0 est un hyper-paramètre qui multiplie la somme des erreurs, **Elle pénalise les termes d'erreur.** > plus C est petit, plus la capacité diminue, plus je me permets de faire des erreurs, ➔ **C trop faible, risque de sous-apprentissage.** > lorsque C = ∞, on revient au problème original (je pénalise très fortement les violations des contraintes de marges), cad je ne viole aucune de mes contraintes de marge ➔ **C trop grand, risque de sur-apprentissage.** Exemple graphique :  	Commentaires : . un modèle SVM est entraîné pour un C = donné, . j'autorise des erreurs, . présence de vecteurs supports, pour Y(X) = 1 ou -1, je respecte mes contraintes de marges pour ces points, le ξ = 0, . mais aussi le ξ = 0, pour tous les points verts et turquoises bien classés, . mais pour le point qui pénètre dans ma marge **y = 1**, le $\xi < 1$, ξ est dans la marge et du bon côté de la surface de décision, c'est à dire en dessous de la droite séparatrice rouge, c'est à dire du bon côté des points turquoises. . pour le point qui pénètre dans ma marge **y = -1**, le $\xi > 1$, car le (Xi) ξ n'est dans la marge et du bon côté de la surface de décision, c'est à dire en au-dessus de la droite séparatrice rouge, c'est à dire du bon côté des points verts. . en revanche, par rapport au cas linéairement séparable, ici je dois mettre à jours mes vecteurs de supports, c'est à dire, en plus de mes points sur les marges qui sont vecteur de supports, je rajoute les points pour lesquels j'ai une violation de mes contraintes de marge, ces points font partie de ma solution optimale. C'est à dire, une fois l'algorithme SVM entrainé sur l'ensemble d'apprentissage, je devrais garder ces points qui violent mes contraintes de marges, ils font partie de la somme pondérée des entrées qui font me vecteur W. [ECRITURE DUAL de mon problème d'optimisation avec intégration des var. ressort ξn] $L(a) = \sum_{n=1}^{N} an - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} an\,am\{tntm\}k(Xn,Xm)$ *avec* $k(Xn,Xm) = (\phi(Xn)^T) * \phi(Xm)$ *… astuce du noyau* *Sous Contrainte* $\sum_{n=1}^{N} an * \{tn\} = 0$ *Sous Contrainte* $C >= an >= 0$ *[cas de séparation imparfaite]* . on retrouve la même formulation du précèdent dual, modulo, que l'on met à jour mes contraintes sur les an. Ont dit que les an *sont mis en « Box Constraint »* (tels que Simple bounds of the form $L \leq x \leq U$). . Au final : on doit résoudre un problème de programmation quadratique avec des contraintes mises à jour. **Rappel - la Capacité d'un classifieur SVM ?:** En 1995, Vladimir Vapnik a prouvé que cette maximisation produit un hyperplan optimal, c'est-à-dire qu'il produit le moins d'erreurs de classification. C'est la ***capacité de généralisation maximale.***

6.1.3.2. Cas de problème de classification multi-classe

(Source : https://zestedesavoir.com/tutoriels/1760/un-peu-de-machine-learning-avec-les-svm/#3-tet)

EXPRESSION LITTERALE		FIGURE / COMMENTAIRES
. Le classifieur SVM initialement développé pour les problèmes de classification binaire supervisée peut être adapté au problème de classification à plusieurs classes où (k le nombre de classes) k > 3. . En pratique, l'adaptation du SVM bi-classes au cas multi-classes se fait selon 2 stratégies identifiées. Le choix de l'une ou l'autre se fera selon la taille des données du problème à traiter. **(1) Stratégie du « one versus all » (un contre tous)** - l'adaptation consiste à transformer un problème de classification de k classes en en k classifieurs binaires de taille (n), c'est à dire à entraîner itérativement un SVM bi-classes en utilisant les points d'une classe contre une autre classe qui regroupe tous les points restants des autres classes. - critère de décision : le classement est donné par le classifieur qui obtient le meilleur taux de classification. - modèle : . construire itérativement k classifieurs avec Yk pour Y ∈ $\{Y1,..,Yk\}$, → ce qui produit K fonctions de décision Fk(X) . en classement, retenir la classe au score le plus important selon la règle $\boldsymbol{Y^{\wedge} = argmax_k\ Fk}$**(X)** - avantage : facilité de mise en œuvre - inconvénients : . lors de la construction du modèle bi-classes, on peut introduire un biais de classes déséquilibrées, donc obtenir des scores biaisés et donc ne pouvoir mettre en œuvre équitablement la règle $Y^{\wedge} = argmax_k\ Fk$(X). . on manipule « de grosse » matrice de points **(2) Stratégie du « one versus one - pairwise» (un contre un)** - l'adaptation consiste à transformer un problème de classification de k classes en (k(k-1)) / 2 classifieurs binaires. Chaque classe i étant en duel avec chaque classe i'. C'est à dire entraîner (k(k-1)) / 2 SVM sur chacun des couples de classes, puis de décider la classe gagnante à l'aide d'une matrice de vote pour chaque duel et par un vote majoritaire (la classe qui remporte le plus de duel) - critère de décision : • **matrice de voting : « winner takes all »** . règle du vote majoritaire	• **sortie en probabilité pour décider sur les zones in conclusives** . organiser un post-traitement des résultats de voting grâce à l'estimation des probabilités à priori. - modèle : itérativement, on produit k(k-1)/2 modèles discriminants de paire de classes, → ce qui produit k(k-1)/2 fonctions de décisions Fk,j(X) . en classement, la classe qui affiche le plus grand nombre de victoire telle que **Dk(X) =** $\sum_{j=1,j\#k}^{K} \boldsymbol{sign}[\boldsymbol{Fk,j}\ (\boldsymbol{X})]$ → va fournir le vote et selon la règle $\boldsymbol{Y^{\wedge} = argmax_k}$ **Dk(X)** - avantage : le problème de classe déséquilibrée est réduit, - inconvénient : temps de calcul s'accroît en fonction de la taille des données et du nombre de classe k. - codage du problème : ne pas oublier d'indexer les lignes pour réduire les temps de calcul. **Conclusion : La gestion de l'erreur en apprentissage, la robustesse à la généralisation sur données test et les conditions optimisées de calcul en grande dimension, font du SVM un classifieur à succès sur les problèmes de classification multi-classes. D'un point calculatoire, restera à identifier le noyau (dans une mesure de similarité) adapté aux données.** ⚠ **Bonnes Pratiques DATALAB :** ********** ***COMMENTAIRES*** Malgré que notre problème business (MATMUT) soit un problème de classification binaire supervisée, il est important de bien maîtriser le classifieur SVM multi-classes, car dans de nombreux cas on peut être amené à la demande des métiers de transformer le problème bi-classes en multi-classes par une discrétisation en « déciles » de la variable d'intérêt du problème étudié. **********	**(3) Une troisième stratégie nommée « approche globale »** a été formalisée comme suit par Philippe Besse (INSA Toulouse) 3. L'approche *globale* consiste à traiter le problème en une seule fois. Cela peut se faire en posant formellement le problème, par exemple si l'on note $f_\ell(\boldsymbol{x}) - b_\ell$ la fonction de discrimination associée à la classe ℓ : $$\begin{cases} \min\limits_{f\in\mathcal{H},b,\xi\in\mathbb{R}^n} & \frac{1}{2}\sum\limits_{\ell=1}^{c}\|f_\ell\|^2_{\mathcal{H}} + \frac{C}{d}\sum\limits_{i=1}^{n}\sum\limits_{\ell=1,\ell\neq y_i}^{c}\xi^d_{i\ell} \\ \text{avec} & y_i\big(f_{y_i}(\boldsymbol{x}_i)+b_{y_i}-f_\ell(\boldsymbol{x}_i)-b_\ell\big)\geq 1-\xi_{i\ell} \\ & \xi_{i\ell}\geq 0 \text{ pour } i=1,\dots,n;\ \ \ell=1,\dots,c;\ \ \ell\neq y_i \end{cases}$$ Le dual de ce problème est un programme quadratique de même nature que celui des SVM et de taille $n \times c$. L'estimateur associé est non consistant mais donne de bons résultats en pratique. Il existe là aussi de nombreuses autres formulation (dont certaines consistantes). [source : Machines à vecteurs supports - https://www.math.univ-toulouse.fr/~besse/Wikistat/pdf/st-m-app-svm.pdf] **** FIGURE **** **source : cours SVM – Gilles Gasso – Lorette Noiret** **Multi Class SVM** zone grise : svm multi-classes – stratégie 1 vs 1 - in conclusive  Figure 12 : Max voting' strategy is not always conclusive

6.1.3.3. Relation entre le classifieur SVM et régression logistique

EXPRESSION LITTERALE		FIGURE
. 2 extensions possibles entre le SVM et les problèmes de régression définies telles que : (1) **<u>Mobiliser le SVM pour faire de la régression</u>** : . il est possible de prendre un SVM pour en faire un algorithme de régression, c'est à dire prédire une cible qui appartient au réel. <u>Principe</u> : l'approche du SVM régression dans une logique de modèle d'apprentissage statistique est opposé au SVM classification. En régression, on souhaite que les points de mon ensemble d'apprentissage se trouve dans la marge autour de mon hyper-plan Y(X). C'est à dire, sur le graphique, les points ne doivent pas dépasser la distance telle que Y(X) ma prédiction complétée de l'intervalle [- ε ; ε] où ε *est appelée la marge.* <u>Noyaux utilisés</u> : Noyaux de splines, d'ondelettes ou Dirichlet. (2) **<u>SVM vs régression logistique ~ 2 algorithmes de classification</u>** ? . Après une transformation d'écriture du problème d'optimisation du SVM, c'est à dire d'exprimer le problème sans contrainte, alors on peut dire que la régression logistique est version « lisse » d'un SVM. <u>Démonstration</u> : conversion sous une forme sans contrainte $(3)\ argmin\ (w, b, \xi n) \left\{\frac{1}{2} * \|W\|^2 + C\ \sum_{n=1}^{N} \xi n\right\}$ *S.C.* $tn * (Y(Xn))$ *>= 1-* ξn $\xi n \geq 0$ *pour n = 1, … , N* **DEVIENT** $(4)\ argmin\ (w, b, \xi n) \left\{\frac{1}{2} * \|W\|^2 + C\ \sum_{n=1}^{N} \xi n\right\}$ *S.C.* ξn *>= 1-* $tn * (Y(Xn))$ $\xi n \geq 0$ *pour n = 1, … , N* **EQUIVAUT** $(5)\ argmin\ (w, b,) \sum_{n=1}^{N} [1 - tn * (Yn)](+) + \lambda * \|W\|^2$ *avec* $\lambda = 1/(2C)$	(5) **DEVIENT** $(6)\ argmin\ (w, b,) \sum_{n=1}^{N} max\{0, 1 - tn * (Y(Xn))\} + \lambda * \|W\|^2$ *avec* $\lambda = 1/(2C)$ **appelée : fonction de perte hinge loss ou fonction à perte charnière** <u>En résumé</u> : quand on entraîne un SVM, on optimise la somme des hinge loss régularisée par [lambda] x [la norme de mes paramètres]. (5) **DEVIENT** $(7)\ argmin\ (w, b,) \sum_{n=1}^{N} ln\{1 + exp(-tn * Y(Xn))\} + \lambda * \|W\|^2$ *avec* $\lambda = 1/(2C)$ **appelée : forme équivalente à la régression logistique** **<u>Conclusion</u> :** en exprimant les problèmes d'optimisation sous forme de fonction perte de la régression logistique et fonction de perte du SVM, puis après vérification graphique, il est démontré **que la régression logistique est la version lisse du SVM**, ou encore qu'il y ait vraisemblance entre ces 2 algorithmes de classification.	 *Figure 13 : SVM algorithme de régression*  *Figure 14 : fonction perte de la reglog version lisse d'un SVM* *. l'erreur de classification (courbe noire)* → $\{1[ty(X) < 0]\}$ **- je regarde si y(x) est du même signe que t ~ ie est-ce que ty(X) est positif ? si positif, pas d'erreur, j'ai placé mon X dans la bonne classe, c'est à dire comme une fonction identité qui retourne 1 si la fonction [ty(X)<0] est vrai –** *. la perte associée à la hinge loss (courbe violette)* → $max\ \{0, 1 - ty(X)\}$ **➔ la courbe violette est la plus proche de ma courbe en noire** *. la perte associée à la reglog (courbe rouge)* → $ln\ \{1 + exp\ (-ty(X))\}$ **➔ la courbe rouge est une fonction plus lisse de la hinge loss** *. la perte associée à la régression MCO (courbe verte)* ***➔ mauvaise approximation de mon erreur de de classification. [le MCO est un mauvais classifieur, à cause du carré la cf verte remonte]***

6.2. Les méthodes SVM dans la pratique

6.2.1. Les 12 questions pour réussir son SVM (Lorette Noiret)

➔ D'un point vue opérationnel, une checklist de 11 questions à se poser ?

Q1 : le (package, langage de code) pour faire le SVM, utilise-t-il le noyau SVM Libsvm développé par Chih-Chung Chang and Chih-Jen Lin ? Sinon comment travaille-t-il ? forme de noyaux ? paramètres d'optimisation ? valeur de C ? paramètres internes ?

Q2 : faut-il transformer les Xi, pour resserrer les intervalles de valeurs [-1,1], [0,1] (centrage, réduction, centrage-réduction) ? Sinon il y a un risque de grosse matrice de données et un temps de calcul rallongé,

Q3 : faut-il normaliser les Xi, appliquer un changement de variable pour normaliser les Xi (log(x), $\sqrt{x}, \exp(x)$) ?

Q4 : la var Yi est-elle continue (quanti) ou nominale plusieurs modalités ➔ est-ce un problème de régression ou classification ?

Q5 : selon la connaissance du problème, notre problème est-il linéairement séparable ou non ?
Quel noyau SVM faut-il appliquer pour résoudre le problème ?

Q6 : si c'est un problème de classification, est-il bi-classes (binaire, k=2) ou multi-classes (k>2) ? quelle la stratégie de résolution ?

Q7 : selon le dimensionnement du dataset [nb ligne x nb colonnes], selon si c'est une classification binaire ou multi-classes, quelle stratégie de traitement : one vs one / ou / one versus all pour éviter les zones d'indécisions ?
Faut-il employer la matrice de « voting », avec combat de classe et fonction de décisions : « winner takes all » ?

Q8 : pour identifier de quel côté de la frontière de décision se situe le point candidat à la classification,
Doit-on utiliser l'Argmax de la fonction de décision (distance Max) ou une sortie en probabilité après application d'une régression logistique ? (avec utilisation d'un dataset de test et validation croisée)

Q9 : en problème de classification, comment corriger le poids déséquilibré des classes (modalités de Yi) ?

Q10 : quand on a un paquet de nouveaux points à classifier, comment faire la prédiction et quelle est la matrice de confusion ?
On met dans la même matrice le vecteur candidat avec Yi = (0, .. ,0) + vecteur Yi_chapeau (vecteur prédictions)

Q11 : pour trouver les valeurs optimales (C=Cost, G=Gamma) du noyau, quelle est la stratégie de gridsearch performante ?
« Coarse grid first » + grille affinage ou grille d'affinage « first » (➔ faire une « ontology du problème avant pour décider »)
+ mise en œuvre d'une procédure de validation croisée ?

Q12 : propre noyau vs noyau packagé ? si propre noyau, 1. Appliquer un polynôme (filtre non linéaire) et 2. Calculer les valeurs de la matrice de Gram ?

6.2.2. Les aspects logiciels et matériels

(i) L'architecture logicielle :

. L'utilisation d'un Séparateur à Vaste Marge pour traiter un problème de discrimination trouve une solution optimale par la résolution d'un problème d'optimisation convexe quadratique. Mais cette résolution nécessite un traitement mathématique spécifique avec l'utilisation de solveur en programmation quadratique.

. Par exemple, Stéphane CANU[14] propose d'utiliser le logiciel Matlab pour coder le SVM, accompagné du solveur quadratique avec méthodes de contraintes actives MonQP.

. Une « library » (brique logiciel) importante à connaître (LIBSVM[15]) couvre la majorité des problèmes pouvant être traités par le SVM (classification, régression et estimation de densité). Cette brique permet de traiter des problèmes en grande dimension et met en œuvre toutes les étapes du traitement numérique imposé par l'algorithme SVM (de la normalisation des données à la transformation de matrice de voting en sortie probabilité, avec du multi-classes et avec une imparfaite séparabilité des données).

. Pour le traitement numérique du chantier MRSQ – transformation des devis, j'ai codé l'application de l'algorithme SVM en langage R [version 3.4.0 (2017-04-21)] avec son IDE R studio [version 1.0.153 pour Mac OS X 10_12_6] et accompagné des packages : **caret, plyr, ROCR, ggplot2, kernlab, parallel, doMC, iterators, foreach.**

. **Le package Caret** propose en « mode natif » une implémentation de la « library » LIBSVM avec la possibilité d'une procédure de calcul parallèle activée avec les « libraries » (parallel + doMC). L'utilisation couplée de ces packages sous R permet de traiter des échantillons de taille moyenne et des traitements statistiques telles que l'indice AUC de la courbe de ROC (package ROCR).

⚠ **Bonnes Pratiques DATALAB :**

. Pour les futurs développements, il existe 2 ressources importantes pour suivre les « libraries » spécifiques SVM mises à jour et disponibles dans les différents environnements de développements : R, Python, Java, scala …

- http://www.kernel-machines.org/
- http://www.support-vector-machines.org/

. Sur les travaux réalisés lors de la 1ère période, j'ai utilisé le logiciel tableau.com pour présenter la pertinence d'un outil de datavizualisation lors de la phase d'analyse descriptive du modèle des données Y ~ Xi.

(ii) L'architecture matérielle :

Pour faire ce chantier, j'ai utilisé l'ordinateur suivant :

	Sous contraintes des spécifications du chantier : . la configuration matérielle proposant une puissance calculatoire limitée, . l'environnement R utilisant des procédures de chargement des données et de calcul «all In-memory », . avec une taille des données = 728674 observations x 112 variables = 1 fichier.csv de 370 Mo . l'algorithme SVM demandant des calculs coûteux (grid search),

J'ai dû suivre 2 axes nécessaires pour la réalisation des calculs :

. Axe 1 : une procédure de calcul SVM en mode parallèle,

. Axe 2 : réduire la taille des données, en prélevant un échantillon de données n = 6150 lignes
(*i.e. la méthode d'échantillonnage sera décrite ci-après*)

[14] Enseignant chercheur, spécialiste du SVM, Insa Rouen – [http://asi.insa-rouen.fr/enseignants/~scanu/SVM/]
[15] https://www.csie.ntu.edu.tw/~cjlin/libsvm/ -- des auteurs : Chih-Chung Chang and Chih-Jen Lin.

6.2.3. Paramétrer un SVM performant en apprentissage inductif

Formellement :
. Paramétrer un SVM performant en apprentissage statistique revient à déterminer un hyperplan dont la distance est minimale aux points de mon ensemble d'apprentissage, c'est à dire celui qui passe au milieu des 2 classes et dont la distance au point les plus proche est maximale.
. Mettre en œuvre un apprentissage inductif, c'est arriver à classer des nouvelles données grâce aux connaissances apprises à l'aide d'un modèle sur les données d'un ensemble d'apprentissage. L'objectif est de bien spécifier le modèle sur les données d'apprentissage (erreur empirique petite, mais pas trop) et de s'assurer que le modèle généralise bien sur les données test (erreur réelle petite), pour ensuite l'utiliser (ie le modèle spécifié) pour classer de nouvelles données.

(1) Définition de la capacité d'un modèle

. Définition : la capacité d'un modèle est son aptitude à apprendre « par cœur » les données d'apprentissage. Plus la capacité est grande, plus la différence entre l'erreur d'entraînement (empirique/modélisation) et l'erreur de test (réelle/prévision) augmente. Idéalement, on souhaite que le modèle construit en apprentissage ne soit pas en sur-apprentissage (il a appris à partir de données Xi + il a appris le bruit aussi). C'est à dire que l'écart entre l'erreur empirique et l'erreur réelle ne soit pas trop grand en test. Itérativement sur les Xi test, les 2 courbes d'erreur convergent en forme. Notre objectif est de bien fixer la valeur de notre hyper-paramètre de régularisation pour contrôler la Capacité (Vladimir Vapnik, 1995, SVM : a prouvé que cette maximisation produit un hyperplan optimal, c'est-à-dire qu'il produit le moins d'erreurs de classification. C'est la ***capacité de généralisation maximale***)

. Illustration de la capacité d'un modèle : Le compromis biais – variance : équilibrer la complexité d'un modèle vs capacité à se généraliser.

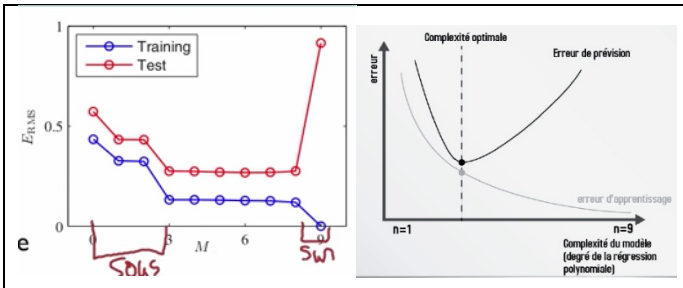


Figure 15 : degré de complexité optimale d'un modèle en fonction des erreurs d'apprentissage et de prévision

[source : data science – p.68 – Eric Biernat / Michel Lutz]

Ex : régression polynomiale (graphique à droite)
. erreur de modélisation (↘) quand complexité (↗) [ie degré (↗)]
. erreur réelle (↘) tant que (↗) de la complexité → (↗) les résultats en prévision
[hors : la complexité (↗) quand le modèle commence à sur-apprendre]

→ modèle optimal : modèle qui minimise les 2 erreurs **= compromis biais-variance**

Verticale de la complexité optimale : comportement d'un classifieur ?
à gauche : biais (+) & variance (-), prévision (=) → sous-apprentissage
➔ amélioration du modèle avec (+) complexité

à droite : biais (-) & variance (+), prévision (KO) → sur-apprentissage
➔ amélioration du modèle = simplifier le modèle

. Généralisation vs. quantité de données : plus la quantité de données en apprentissage augmente, plus le modèle entrainé va bien généraliser. Sauf, pour le classifieur SVM qui détermine la solution optimale de l'hyper-plan à partir seulement des vecteurs supports (« on dit alors que les autres points ne participent pas à la solution optimale… on peut jeter les données d'apprentissage).

. Sélection de modèle : pour déterminer la capacité de généralisation maximale, on utilise une procédure de sélection de modèle par validation croisée afin de déterminer la valeur optimale des paramètres internes de mon modèle et la valeur de de mon hyper-paramètre C de régularisation.

. Définition : **la dimension VC (Vapnik-Chervonenkis)**
La dimension de Vapnik-Chervonenskis d'un ensemble de fonctions {f (x, α)} est le plus grand nombre de points pouvant être séparé (ou pulvérisé) par {f (x, α)}. Dans l'exemple suivant, nous remarquons qu'il existe bien un hyperplan séparateur permettant de séparer les points de classes différentes. Le dernier cas (i.e. à droite) n'est pas faisable, nous ne pouvons séparer parfaitement les quatres points. Notons également que, de manière générale, un ensemble de fonctions de dimensions p, auront une dimension de VC de (p+1)
(correspondant aussi au nombre de "free parameter" ou encore appelé degré de liberté).

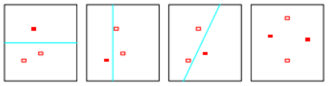


FIGURE 7.6. *The first three panels show that the class of lines in the plane can shatter three points. The last panel shows that this class cannot shatter four points, as no line will put the hollow points on one side and the solid points on the other. Hence the VC dimension of the class of straight lines in the plane is three. Note that a class of nonlinear curves could shatter four points, and hence has VC dimension greater than three.*

Figure 16 : graphe de la dimension VC

Cette dimension VC est très importante car directement reliée au classifieur SVM : le principe fondamental du SVM est d'inclure le contrôle des paramètres (correspondant à la notion de VC dimension) qui sera associé aux vecteurs supports.

Interprétation : la dimension VC est un indicateur de mesure de complexité du modèle.
- modèle avec dimension VC (↗) = erreur de modélisation (↘) **mais risque de sur-apprentissage,**
- modèle avec dimension VC (↘) = erreur de modélisation (↗) **mais risque de sous-apprentissage.**

[source : rapport PFE / insa 76/ p. 10 – Boukhari]

(2) SVM : lien entre capacité du noyau et hyper-paramètre du noyau.

Avec la méthode à noyau SVM, en plus de l'action de la valeur de C = constante de régularisation de la capacité de mon modèle, le SVM possède des paramètres internes qui agissent aussi sur la Capacité de mon modèle (i.e. c'est la raison pour laquelle on dit que le modèle SVM est sur-paramétrisé). Selon la forme fonctionnelle du noyau sélectionnée pour répondre à la nature du problème, la valeur de ces paramètres agissent sur la capacité de mon modèle (cf. les tableaux de noyaux ci-dessous).

(3) Problème de classification linéaire / non linéaire : choix noyaux et choix des hyper-paramètres

Méthode : quelle que soit la nature du problème de classification (linéaire / non linéaire), le nombre de classes de la variable d'intérêt (bi-classe / multi-classes), la méthode pour spécifier un classifieur SVM performant se décompose en 2 tâches :

- tâche 1 : le choix du noyau,
- tâche 2 : le paramétrage, par validation croisée, des paramètres internes du noyau,

pour le cas de classes imparfaitement séparables, il faut rajouter une tâche complémentaire, le choix de l'hyper-paramètre C (constante de pénalisation des erreurs).

➔ La détermination de la valeur optimale de ces paramètres revient à sélectionner le modèle le plus approprié pour résoudre le problème lié au jeu de données.

(i) Choix du noyau : [reprise de : comment choisir un noyau - proposé par Antoine Cornuéjols]

« … Il est clair que les fonctions noyau encodent des connaissances essentielles sur le problème d'induction étudié, en particulier :
- Une mesure de similarité sur les données,
- La forme fonctionnelle des fonctions de décisions possibles,
- Le type de contrôle réalisé sur les hypothèses : par exemple les fonctions noyau Gaussiennes pénalisent les dérivées de tous les ordres et favorisent donc les solutions « régulières »,
- Le type de covariance dans l'espace des entrées stipulant comment des points en divers endroits de l'espace sont liés les uns aux autres. On peut ainsi chercher des fonctions noyau invariantes pour certaines transformations de l'espace d'entrée, comme par exemple la rotation. Il faut donc les choisir avec soin en essayant de traduire grâce à elles le maximum de connaissances préalables dont on dispose sur le problème et sur les données … » .

[source : Page 20. / Une nouvelle méthode d'apprentissage : Les SVM - Antoine Cornuéjols / afia – numéro 51 – juin 2002]

⚠ **Bonnes Pratiques DATALAB :**
Pour choisir un noyau adapté au problème à traiter (classification / régression), il est utile de rechercher les précédentes études réalisées sur le problème observé (i.e. Ontology). Cela permettra d'épargner du temps de mise en œuvre et d'expérimentation pour obtenir les paramètres optimaux, mais aussi cela permettra d'acquérir une base de comparaison en termes de performance du noyau sélectionné sur les résultats d'apprentissage / de tests.

(ii) Choix des paramètres internes du noyau : le Gamma = γ

. Paramètre par défaut : si non spécifié par l'analyste, les outils (librairies / package) le fixent par défaut.
Exemple = pour le noyau Gaussien : par défaut, $\gamma = \frac{1}{p}$ (où P = le nombre de variables du modèle)

. Valeur fixée par l'utilisateur : après une procédure de validation croisée en « Grid search = grille de recherche, à la main ou automatique (proposée par certains outils) ».

(iii) Choix de la valeur C (cost) du problème d'optimisation :
Signalétique : (↗) = grand ; (↘) = petit

Contexte : dans le cas de problème de classification type séparation imparfaite.

Rappel : C, coût de la pénalisation du terme d'erreur / régularisation « hyper – paramètre » est la constance de rééquilibrage contrôlant le compromis entre la fidélité aux données et la tolérance aux erreurs du classifieur SVM.

Réglage de C : méthode classique et coûteuse, réalisée par une procédure de validation croisée sur une grille prédéfinie de valeurs de C.
En pratique, on fait varier la valeur de C sur l'intervalle [0, ∞ [et on sélectionne la valeur pour laquelle l'erreur estimée par validation croisée est minimale.

Valeur de C apprentissage	Type de marge	Biais	Variance	Taille de la marge	Type hyper-plan	Valeur ξn (Xi)	Pouvoir de classification	Taille I.C	Risque apprentissage	Généralisation
(↘) [Petit] : C ~ 0	soft margin	(↗)	(↘)	(↗)	linéaire « smooth »	ξ grande	(-)	(↗)	sous	(+)
(↗) [Grand] : C ~ ∞	hard margin	(↘)	(↗)	(↘)	forme colle aux data	ξ petite	(++)	(↘)	sur	(-)

3.1. : cas linéairement séparable – connaissance sur les noyaux :

Noyau (forme fonctionnelle)	Paramètre du noyau	Biais	Variance	Problème type	Analyse Capacité (Ca)
Linéaire **k(x, x') =** $(\mathbf{x}^{\mathbf{T}} * \mathbf{x}')$				Séparation linéaire entre 2 classes (marge douce) - pour des données en grande dimension, un noyau linéaire est suffisant	Classification type « les plus proches voisins »
Polynomial → Linéaire **k(x, x') =** $(\mathbf{x}^{\mathbf{T}} * \mathbf{x}' + \mathbf{c})^{\mathbf{M}}$	c = 0 et M = 1 (degré)				

3.2. : cas non linéairement séparable – connaissance sur les noyaux :

Noyau (forme)	Paramètre du noyau	Biais	Variance	Problème type	Analyse Capacité
Polynomial **k(x, x') =** $(x^T * x' + c)^M$	c et M = le degré (ordre) du polynôme (apriori fixé / utilisateur)			Régression polynomiale / classification avec M = 2 ou 3 -- traitement automatique du langage naturel (NLP) - pour des données en grande dimension et creuses	Capacité à bien mesurer la complexité d'un modèle
	M (↘)				Ca (↘)
	M (↗)				Ca (↗)
Gaussien (RBF = Radial Basis Funct.) **k(x,x') =** $\exp\{-\|\mathbf{X}-\mathbf{X}'\|^2 / (2\boldsymbol{\sigma}^2)\}$ $\gamma = \mathbf{Gamma} = \mathbf{ddl} = \mathbf{1}/(2\boldsymbol{\sigma}^2)\}$	σ^2 = variance, commun à tous les noyaux (apriori fixé utilisateur) -- σ = largeur de bande			Noyau passe-partout mais peut aussi ne pas convenir au problème traité (rôle de l'Ontology)	Capacité à bien mesurer la similarité entre les couples de points
	Par défaut : $\gamma = \frac{1}{p}$ (p = Nb xi)				
	[Petit] : γ ~ 0 [$\boldsymbol{\sigma}^2$(↗)]	(↘)	(↗)	Frontière linéaire Sous-apprentissage	Ca (↘)
	[Grand] : γ ~ ∞ [$\boldsymbol{\sigma}^2$(↘)]	(↗)	(↘)	Frontière quadratique Sur-apprentissage	Ca (↗)
Sigmoïde **k(x,x') =** $\tanh(\gamma * < \mathbf{x}, \mathbf{x}' > \ + \mathbf{c})$	γ = gamma (le théorème de Mercer spécifié que pour certaines valeurs de γ)			Problèmes où le nombre de dimensions est élevé ou une séparation non linéaire en 2 dimensions – noyau alternatif au Gaussien, mais moins performant que le Gaussien	

<u>Conclusion</u> : pour la mise en pratique des SVM

→ 4 noyaux de bases Linéaire, Polynomial, Gaussien et Sigmoïde sont mobilisables

<u>Règle 1</u> : pour choisir son noyau, regarder ce qui a été fait avant sur le sujet,

<u>Règle 2</u> : prendre le temps de bien déterminer les valeurs optimales des paramètres du noyau et de l'hyper-paramètre C.

<u>Remarque</u> : si la règle 2 n'aboutit pas à l'obtention de valeurs optimales, malgré avoir sélectionné un noyau adapté au problème à traiter, la performance du modèle spécifié sera dégradée en apprentissage comme en test.

⚠ **Bonnes Pratiques DATALAB :**

- Pour bien comprendre l'importance de régler conjointement les paramètres internes d'un noyau et ainsi que la valeur de la régularisation C, je propose de tester la procédure de simulation numérique et dynamique autour du paramétrage d'un noyau Gaussien définie comme suit (lien URL disponible sur la page personnelle de l'enseignant chercheur Rémi flamary)

https://remi.flamary.com/demos/svmreg.fr.html

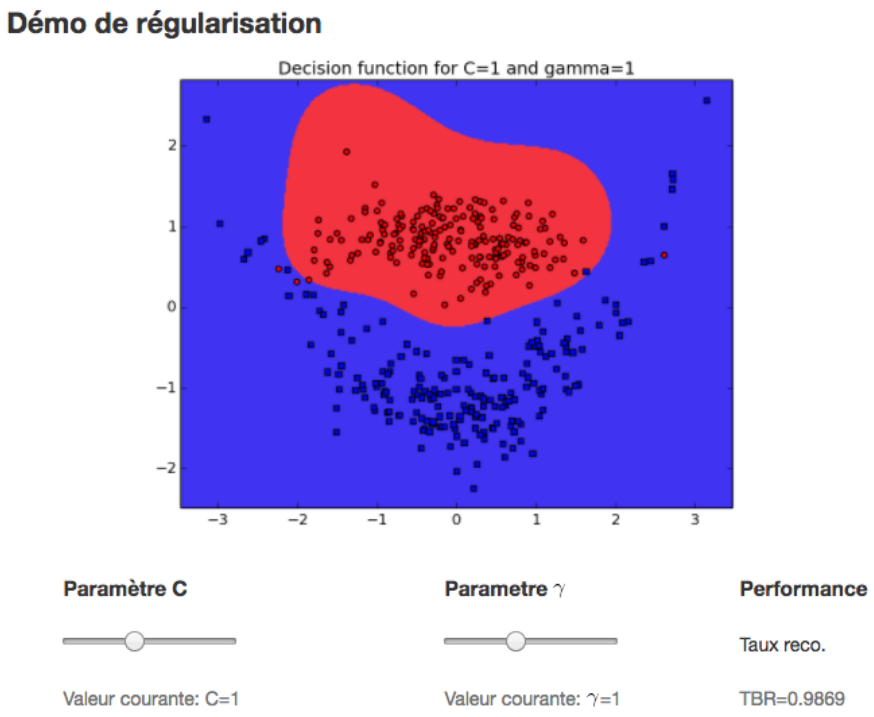


Figure 17 : effet de la régularisation et du paramètre de noyaux pour la discrimination à l'aide d'un SVM.

Enseignements :

. Le processus de détermination des paramètres optimaux d'un noyau Gaussien, même sur un problème bi-classes à complexité limitée n'est pas aussi simple qu'il n'y paraît (comme décrit dans ce cas « jouet » discrimination 2D – données clown).

. Ce processus, comme toute activité de « machine learning » nécessite du temps pour mener une activité de tests essais-erreurs avant de trouver la solution optimale et de sélectionner la spécification du modèle performant sur le problème considéré.

6.2.4. Comparer plusieurs de méthodes à noyaux

Méthodes (M[i]) : 2 méthodes connues,
. M1 = empiriquement : par validation croisée,
. M2 = heuristiquement,
- nombre de points vecteurs supports → moins il y en a, mieux c'est (… mais théoriquement pas de preuve)
- caractéristique de la matrice de Gram → si pas de structure (forme) dans G, pas de régularité identifiée.

Par exemple :
. si les termes non diagonaux sont petits alors risque de sur-apprentissage,
. si la matrice semble uniforme (tous les points sont prédits sur la même classe) alors risque de sous-apprentissage,

En Pratique :
(i) Le critère d'accuracy = le taux de bien classé calculé à partir de la matrice de confusion,
(ii) Le critère AUC / courbe de ROC,
(iii) Nombre de vecteurs supports.

6.2.5. Comparer le SVM à d'autres classifieurs

- Dans le cadre de notre application numérique, pour comparer la performance du classifieur SVM aux performances d'autres classifieur (ie RF), dans un souci de d'utiliser des critères simples et lisibles, nous utilisons :

(iv) Le critère d'accuracy = le taux de bien classé calculé à partir de la matrice de confusion,
(v) Le critère AUC / courbe de ROC (accompagné des critères associés : Spécificité & Sensibilité),
(vi) F-mesure et Kappa.

Remarque :
. Un audit des valeurs empiriques de ces mesures doit être réalisé avant de conclure sur la performance ou non d'un classifieur. C'est à dire, il est raisonnable de contextualiser les résultats à la nature du problème de classification.

7. Les données

7.1. Périmètre des données

- le périmètre est fixé pour les 2 périodes de calcul du chantier :

SPECIFICATION DU MODELE DES DONNEES	
date	13.fevrier.2017
version	version 3.1
source des données	données MRSQ - année 2015
nombre de variables	112 « features » - variables
nombre d'enregistrements	728674 lignes
périmètre devis concerné	Devis auto (hors 2 roues) et particulier (hors usage professionnel)
problème	classification binaire supervisée **avec CLASSES DESEQUILIBREES**
type d'analyse	modèle de discrimination avec sélection de variables **quantitatives**
Note destinée au lecteur	la variable nombre de jour qui sépare le 1er devis (delai_1_ctc) non prise en compte pour cette version du rapport

Analyse : Comparé aux autres travaux du Matmut / Datalab sur la transformation Devis MRSQ, le périmètre de mon travail a été concentré sur le segment de marché des 4 roues pour un usage particulier. Après un audit des données par Benjamin Lamarre[16], expert Gouvernance des données Matmut/SCD, afin de bien analyser le processus de transformation du « cœur de marché historique de la Matmut », il paraissait judicieux de fixer cette spécification des données.

7.2. Présentations des données explicatives

- Résumé des éléments de méthodes présentés au paragraphe 5.2.4

Lors de la Période 1 :
- sur la base de l'expertise Métier de Benjamin Lamarre, sélectionner les variables discriminantes parmi les 112 Xi,
- construire une base de résultats de classification à partir de 6 modèles : ACP, AFD, LDA, RL, RF, XGboost,
- retenir un classifieur de référence.

Lors de la Période 2 :
- construire un dataset optimisé pour la mise en œuvre des SVM [*notamment avec procédure échantillonnage*]
- produire les résultats optimisés (grid search) pour les classifieurs RF, SVM,
- comparer et expliquer les résultats SVM vs les autres classifieurs.

➔ « Pipeline[17] » pour construire un dataset optimisé pour réaliser une prédiction avec un modèle SVM

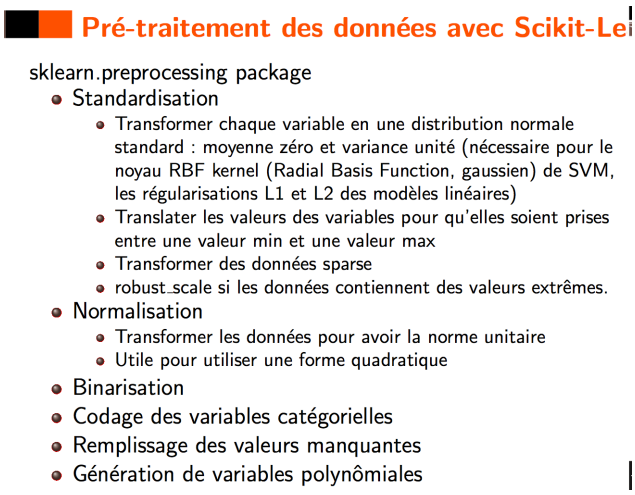


Figure 18 : « Pipeline» construire un dataset pour réaliser une prédiction avec un modèle SVM

Source : data sc., introduction au machine learning, p10, Ons Jelassi, Telecom Evolution.

[16] Permanent Matmut/SCD, statisticien, expérimenté sur les questions d'actuariat et de tarification iard MATMUT (notamment contrat MRSQ).
[17] « a pipeline has inputs go through a number of processing steps chained together in some way to produce some sort of output ".

7.3. Utilisation d'un échantillon S.S.P.

Pourquoi un échantillon obtenu par sondage stratifié proportionnel (S.S.P) ?

Pour les raisons évoquées de limitations de traitement calcul liées aux architectures logiciels et matériels mobilisées, la construction d'un échantillon de données vise à ne pas travailler sur le dataset total (> 600 000 Lignes) mais à prélever un échantillon par une méthode de sondage.

Rappel de l'objectif (vu au paragraphe 5.2.4) : construire un échantillon optimisé de 6150 lignes x 7 colonnes [Yi ~ X1 à X6] par la méthode de sondage stratifié proportionnel.

Concrètement : obtenir des calculs R qui tiennent en RAM ordinateur et un temps de traitement réduits pour les SVM.

2 questions se posent :
. Quelle est la taille du prélèvement au sein du « dataset » total ?
. Quelle est la méthode d'échantillonnage à employer pour ne pas introduire de biais de sélection dans notre problématique de prévision par méthode d'apprentissage statistique ?

Les réponses ont été apportées par l'enseignant-chercheur Jean-Michel Poggi lors de la séance de travaux pratiques sur les classifieurs Arbres Cart et Random Forest, et par la suite complétées par les précisions méthodologiques fournies par le document arbres CART et Forêts aléatoires de référence hal0138765v2.

→ Réponse : construire un sondage stratifié proportionnel sur la variable d'intérêt (Yi)

[*Dans notre cas on prélève 1% de l'effectif total 614929 lignes, c'est à dire un effectif n = 6150 lignes*]

6.3 Echantillonnage et BLB

L'une des limites importantes de MR-RF est le biais d'échantillonnage possible provenant du découpage des données en plusieurs morceaux pour les travaux Map exécutés en parallèle. Cela pourrait être pris en compte en améliorant le plan de partition, en utilisant au moins une partition au hasard des données ou, probablement plus efficacement, grâce à une stratification selon la variable Y, ou encore de sous-échantillonner plus soigneusement en exploitant le fait que l'ensemble des données n'est probablement pas nécessaire pour obtenir des estimations précises.

[source : P.37 Arbres CART et Forêts Aléatoires, Importance et sélection de variables – R Genuer & Jean-Michel POGGI - hal0138765v2]

→ Réponse : Que pensez du poids de 1% de l'effectif prélevé par rapport à rapport à l'effectif de l'échantillon total
(i.e. effectif du « dataset total » période 2 = 614929 lignes)

En théorie des sondages et selon les pratiques observées lors des élections présidentielles en France, l'application du Théorème Centrale Limite indique que la marge d'erreur des sondages d'intention de vote pour les candidats est de + / – 1% des votes pour un intervalle de confiance de la mesure à 95%. Cela veut dire en pratique que la marge d'erreur est garantie pour un échantillon de 1000 personnes par rapport à la taille de la population des votants (47 millions de votants). La taille optimale de l'échantillon n'a rien avoir avec la taille de la population totale. Par exemple, dans le cas de l'élection présidentielle, pour réduire la marge d'erreur à 0.5%, il faudrait un échantillon de 4000 personnes sondées.[18]

Enseignement pour notre dossier : la taille prélevée de n = 6150 lignes n'est ni bonne, ni mauvaise. Simplement, par empirisme, il a été testé que les calculs du code SVM / RF sur cette taille tiennent en « Ram » sans générer des temps de calculs trop longs.

→ Réponse : Méthode sondage stratifié optimal ou sondage stratifié proportionnel
En revanche, le plus important est de ne pas introduire un biais de sélection suite à un prélèvement de lignes. Pour cela, j'ai fouillé les outils proposés lors de la réalisation du MOOC Méthodes de sondage et d'enquête proposé par Université Libre de Bruxelles (ULB). L'outil de sondage stratifié en utilisant la variable intérêt Yi [O/N] (trsf_dev) comme variable de stratification m'a semblé être adapté aux recommandations de JM Poggi. La méthode sondage stratifié optimal est non nécessaire ici, car les variables explicatives Xi sont centrées-réduites (μ = 0, $\sigma = 1$), cependant il faut s'assurer du prérequis suivant : ***vérifier que les « metrics » moyenne et écart-type sont bien respectées au sein des sous populations Non / Oui de l'échantillon prélevé !***

[18] https://www.franceculture.fr/emissions/la-methode-scientifique/peut-calculer-le-hasard

Définition : l'erreur d'échantillonnage

SCRIPT

L'erreur d'échantillonnage

Module 8 – Sources d'erreur dans une enquête par sondage
Section 2 – L'erreur d'échantillonnage

Le premier type d'erreur intervenant dans l'erreur totale associée à l'estimation d'un paramètre-population est l'erreur d'échantillonnage. Nous avons introduit cette notion dès le troisième module de ce MOOC.

Rappelez-vous… Cette erreur est due au fait que l'on estime un paramètre de la POPULATION à partir d'un ECHANTILLON prélevé dans cette dernière ; l'échantillon n'est jamais qu'un sous-ensemble de la population et il n'est donc pas étonnant qu'il ne puisse nous fournir qu'une approximation de la valeur exacte du paramètre.

L'erreur d'échantillonnage est donc intrinsèquement liée à la procédure même de l'estimation. Elle est inévitable, mais on peut en évaluer l'importance de diverses manières, du moins dans le cadre d'un sondage aléatoire : comme je vous l'ai expliqué dans le module 5, on peut se baser sur la valeur estimée de la variance de la distribution d'échantillonnage de l'estimateur du paramètre ; mieux encore, on peut faire appel à l'incertitude absolue ou à l'incertitude relative associées à l'intervalle de confiance que l'on a pu calculer pour le paramètre.

Par ailleurs, le choix d'un plan de sondage bien adapté à la structure particulière de la population et à la façon dont la dispersion de la variable d'intérêt s'y construit, permet de contrôler ou de limiter l'ampleur de l'erreur d'échantillonnage. Pensez, par exemple, au gain en précision que permet, pour une taille fixée de l'échantillon, le sondage stratifié proportionnel ou, mieux encore, le sondage stratifié optimal, par rapport au sondage aléatoire simple.

Définition : sondage stratifié

Conclusion

Module 6 – Sondage stratifié
Section 7 – Conclusion

Que retenir de ce module ?

La stratification consiste tout d'abord à prédéfinir une partition de la population en différentes catégories ou strates. Cette partition se fait avant l'échantillonnage, sur la base des informations fournies sur les unités de la population par la base de sondage par exemple. Il nous faut en réalité savoir à quelle strate appartient chaque unité de la population. Cette connaissance fait que l'on connaît également la taille de chaque strate.

Une fois la stratification de la population bien définie, on tire un échantillon PESR dans chacune des strates de la population, indépendamment d'une strate à l'autre. L'échantillon stratifié final, de taille n, est obtenu en regroupant tous les sous-échantillons aléatoires simples que l'on a prélevés.

La manière la plus simple de répartir les n prélèvements à effectuer entre les différentes strates de la population est d'appliquer le même taux de sondage f, égal à n/N, dans chacune des strates. On parle alors de sondage stratifié *proportionnel*, car chaque strate a le même poids dans la population et dans l'échantillon.

Le sondage stratifié proportionnel s'avère toujours plus efficace – plus précis donc – que le sondage aléatoire simple de même taille. Son efficacité est d'autant plus grande que les strates sont bien homogènes – en intra – et de niveaux moyens très contrastés les uns par rapport aux autres.

Dans certains cas, lorsqu'on dispose de l'information selon laquelle certaines strates sont beaucoup plus hétérogènes que les autres, on gagne en efficacité par rapport au sondage stratifié proportionnel en appliquant des taux de sondage plus élevés dans les strates plus hétérogènes et des taux de sondage plus faibles dans les strates fort homogènes.

S'il nous est possible d'estimer les écarts-types de la variable d'intérêt dans les différentes strates, on peut faire appel à l'*allocation optimale* pour déterminer les tailles des sous-échantillons à prélever dans les différentes strates ; cette allocation donne lieu à la plus grande précision qu'il est possible d'atteindre avec la stratification considérée, pour une taille n d'échantillon.

Enfin, il existe d'autres méthodes de sondage aléatoires que le sondage aléatoire simple ou le sondage stratifié : je pense notamment au sondage *à probabilités inégales*, au sondage *en grappes*, au sondage *à deux ou plusieurs degrés*. Elles sont cependant un peu plus complexes à étudier, raison pour laquelle j'ai choisi de ne pas vous les présenter dans le cadre de ce MOOC.

Retenez toutefois que toutes ces méthodes de sondage poursuivent le même objectif : faciliter la mise en œuvre du sondage tout en assurant une bonne précision pour la procédure d'estimation, et cela en tirant le meilleur parti possible des caractéristiques structurelles de la population à sonder et de la connaissance que l'on a d'informations auxiliaires relatives aux unités statistiques de cette population.

***Metrics* : moyenne et écart-type sont bien respectées au sein des sous populations Non / Oui de l'échantillon prélevé ?**

(1) La variable d'intérêt :

```
dq2norm_y=dq2norm_y[sample(1:nrow(dq2norm_y),6150),]

# sanity check

# 10.2 sur la variable d interet

# en nombre etiquette NON / OUI
summary(dq2norm_y$trsf_dev)
```

```
##    N    O
## 4491 1659
```

```
# N    O
# 4533 1617

# en POURCENTAGE : poids de l'etiquette NON / OUI
PoidsY6150=(table(dq2norm_y$trsf_dev)/nrow(dq2norm_y)*100)
PoidsY6150
```

```
##
##        N        O
## 73.02439 26.97561
```

```
# N        O
# 72.95935 27.04065
Ratio_OUI_NON_6150 = PoidsY6150[[2]]/PoidsY6150[[1]]*100
Ratio_OUI_NON_6150 # [1] 37.06
```

```
## [1] 36.94055
```

```
# comparaison avec les valeurs de la Population Totale
# POURCENTAGE : poids de l'etiquette NON / OUI
PoidsY=(table(dv17$trsf_dev)/nrow(dv17)*100)
PoidsY
```

```
##
##        N        O
## 73.31072 26.68928
```

```
#   N        O
#   73.31072 26.68928
#   dataset avec classes déséquilibrées (les NON majoritaires)

Ratio_OUI_NON = PoidsY[[2]]/PoidsY[[1]]*100
Ratio_OUI_NON # [1] 36.40
```

```
## [1] 36.4057
```

```
delta=(Ratio_OUI_NON-Ratio_OUI_NON_6150)
delta # [1] 0.73%
```

```
## [1] -0.5348458
```

```
# conclusions sur Yi :
# les conditions de proportionnalité de la var Yi sont respectées lors de la réduction
# dataset total à dataset sondage.
```

(1) Les variables Xi : par exemple, la variable COTISATION

```
# 10.3 sur les variables explicatives
# COTISATION
tapply(dq2norm_y[,"LogC_AN_TTC"], dq2norm_y[,"trsf_dev"], mean, na.rm=TRUE)
```

```
##           N           O
##  0.06002123 -0.10353042
```

```
tapply(dq2norm_y[,"LogC_AN_TTC"], dq2norm_y[,"trsf_dev"], sd, na.rm=TRUE)
```

```
##         N         O
## 1.0295601 0.8942901
```

```
tapply(dq2norm_y[,"LogC_AN_TTC"], dq2norm_y[,"trsf_dev"], var, na.rm=TRUE)
```

```
##         N         O
## 1.0599940 0.7997548
```

Moyenne ~ 0 et Ecart-type ~ 1

```
# conclusion sur les Xi :
# le passage dataset total à dataset sondage, garantit le centrage réduction des Xi.


# CONCLUSION
# au regard des résultats vérifiés au paragraphe 10.1, 10.2 et 10.3
# on obtient le dataset sondage,
dim(dq2norm_y) # [1] 6150 lignes x 7 colonnes
```

Pour le détail des autres Xi se rendre à l'annexe :
Calculs numériques - Modélisations Période 2 > 10.7.1 Formalisation du « dataset » de travail

8. Application numérique : le modèle SVM, comparaison avec les autres méthodes

8.1. Période 1 : résultats des 6 modèles (ACP, AFD, LDA, RL, RF, XGboost)

*** RESULTATS NUMERIQUES ***

PERIODE 1

SYNTHESE PERFORMANCE DES CLASSIFIEURS

hypo : relation linéaire Yi ~ Xi	a priori non
Nb total d'observation	614048
Echantillonnage	non

Etiquette Yi	trsf_dev						
Poids des Yi[i]	[-1; +1] == [75.16 ; 24.84]						
Nb Xi	6						
Detail Xi	C_AN_TTC + PUIS_FISC + COEF + AGE_CLI + NB_ENF + ANC_VEH						
normalisation Xi	non						
Langage Prog	R	R	R	R	R	R	R

METRICS	Xgboost	RF	Lda	Reglog	AFD	ACM	ACP
apprentissage	0,7	0,7	0,7				
test	0,3	0,3	0,3				
k (cv- k folds)	1	1	1				
type : classif / reg.	classif.	classif.	classif.				
Correction Poids des Yi[i] - (prior)			[0.5; 0.5]				
ntree		30					
mtry		6					
Tx de biens classés (Accuracy)	0,7625	**0,7811**	0,7222				
Tx d'Erreur	0,2375	0,2189	0,2778				
AUC		**0,6900**					
Kappa		0,4165	0,00				
F1 (F-mesure)		0,8544	0,8337				
Contrôle Graphique		auc / lift					
importance des descripteurs 1	AGE_CLI	C_AN_TTC	C_AN_TTC				
importance des descripteurs 2	COEF	AGE_CLI	AGE_CLI				

modèle retenu

Tableau 2 - synthèse de la performance des classifieurs période de calcul 1

*** Synthèse des résultats ***

But : avant de faire de l'apprentissage statistique (Machine Learning), il faut réaliser une phase d'analyse exploratoire des données (i.e. feature engineering) dont les objectifs sont : parmi différents modèles statistiques utilisables, trouver et paramétrer le modèle statistique « le meilleur » pour la phase de machine learning sur notre jeu des données. C'est-à-dire, vérifier les prérequis définis comme il suit :

. Prérequis 1 : identifier les véritables variables discriminantes,
. Prérequis 2 : qualifier le type de relation linéaire ou non entre les variables explicatives et la variable à expliquer (étiquette Oui / Non),
. Prérequis 3 : sur le critère de la parcimonie (Data Sc.), trouver le modèle le plus simple possible qualifiant le mieux la relation entre Y et Xi,
. Prérequis 4 : sur le critère de performance du classifieur, mettre en œuvre la procédure de validation croisée pour généraliser le modèle.

Résultats : sur ce jeu de données

. **Utiliser les outils de la datavizualisation** pour comprendre les relations entre les variables : entre les Xi, mais aussi entre les Xi et Y.
. En analyse descriptive : ACP fonctionne, AF Discriminante Géométrique et l'ACM ne fonctionnent pas,
. En inférence statistique :
- Analyse discriminante linéaire et Régression logistique ne fonctionnent pas,
- Sur le critère de Performance de taux de biens classés, le modèle Gradient Boosting (XGboost) à 6 variables est moins performant que le modèle de Forêts Aléatoires (Random Forest) avec les paramètres d'algorithme par défaut.

→ Résultat 1 : pas de relation linéaire entre les 6 Xi continues sélectionnés et le Y,
→ Résultat 2 : le rôle des classes déséquilibrées du Y (72% de Non I 28% de Oui) → « fera apprendre du Non en Machine Learning »,
→ Résultat 3 : les Xi discriminantes ne semblent pas toutes identifiées,
→ Résultat 4 : sur le taux de bien classés (Accuracy), en test, le RF affiche le meilleur résultat = 78.11 % et avec une AUC = 69%.

Pour tous ces modèles :
. 2 variables explicatives AGE_CLI et C_AN_TTC sont « discriminantes et plus petit dénominateur commun à tous les modèles », ces résultats sont nominaux et alignés avec l'objectif du « modèle de pricing » de la Matmut de cibler « les bons pères de famille en terme risque iard ».
. Convergence « en variable » avec les résultats de l'algorithme de méthode gradient boosting et RF utilisés en Machine Learning.

⚠ Bonnes Pratiques DATALAB :

Pour les futurs sujets avec phase de Machine Learning (i.e. transformation)
. Passer du temps sur la phase d'analyse exploratoire (« feature engineering »),
. Mobiliser des modèles statistiques différents et variés (« Algo ») pour s'assurer que les 4 prérequis sont vérifiés,
. Mettre en œuvre les tests statistiques adaptés pour vérifier que le modèle retenu « est valide ».

*** Analyse ***

→ Résultat 1 : pas de relation linéaire entre les Xi et le Y
. pas de relation « directe » entre les variables explicatives et la variable à expliquer (étiquette Oui / Non) ➔ la complexité du modèle est trop simple (hypothèse de présence : effet additif et/ ou multiplicatif des Xi, mais aussi d'une relation quadratique Y ~ Xi ?).

→ Résultat 2 : rôle des classes déséquilibrées du Y,
. les différents modèles statistiques de classification recherchent à minimiser l'erreur de classement, ils sont implémentés « c'est à dire codés » pour détecter en priorité la classe majoritaire. En conséquence, pour obtenir un classifieur performant en termes de taux de bien classés (i.e. matrice de confusion) le modèle statistique doit prendre en compte et corriger ce déséquilibre de classe de la variable à prédire (i.e. paramètres de l'algorithme ou autre technique de preprocessing pour redresser).

→ Résultat 3 : les Xi discriminantes pas toutes identifiées ?
. défaut des véritables variables discriminantes … (déf. : une variable discriminante = variable qui aide à séparer les « Oui » et les « Non »),
. le modèle sélectionné : transformation-devis = fct (cotisation, puissance fiscale, coefficient B/M, âge client, nombre enfant, ancienneté VL),
. les Xi sélectionnées sont continues, résultat nominal et aligné avec la qualité des données du SI MATMUT (seule variable bien renseignée),
. les Xi qualitatives « semblent constituer du bruit » sur ce jeu de données (preuve : ACM non significative).

→ Résultat 4 : le modèle Random Forest est le meilleur classifieur sur ce jeu de données ?
. résultats obtenus avec une simple procédure de validation croisée apprentissage / test,
. robustesse de ces résultats à confirmer (en moy.de tx de biens classés) avec la mise en œuvre d'une validation croisée k-folds.

⚠ Bonnes Pratiques DATALAB :

. En parallèle de la mise en œuvre de l'algorithme issus du machine learning XGBOOST / RF pour prédire une classification « O/N » avec un % de taux de « biens classés », il faut s'assurer que le modèle statistique utilisé par l'algorithme mobilise bien les bonnes variables discriminantes et que la prédiction de « biens classés » est fiable [i.e. la robustesse du modèle].
. Construire une grille de modèles statistiques pour déterminer un socle stable et convergent de variables discriminantes, puis identifier un modèle de classification dont la performance de « biens classés » se généralise en phase Machine Learning.

*** les éléments chiffrés / visuels à retenir ***

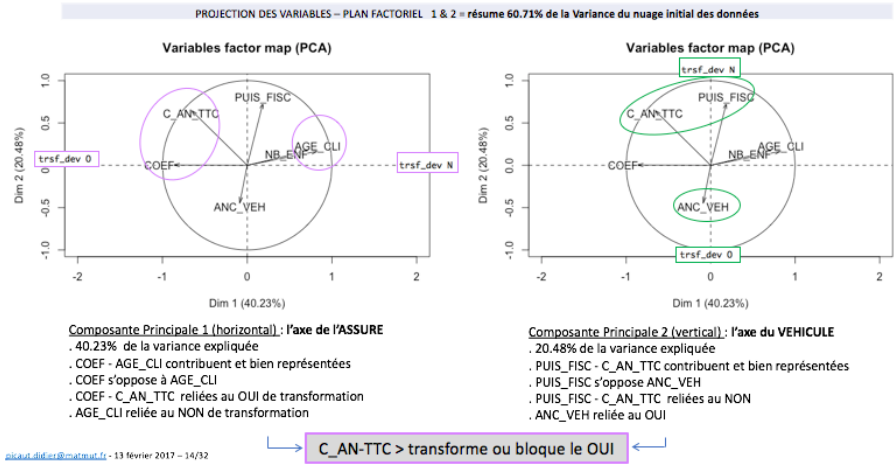
Partie 1. : analyse méthodes descriptives

partie 1.1. : 1er éléments de Datavizualisation

Modèle.0 : Datavizualisation -- pour le détail, voir les éléments annexe 10.6.2

partie 1.2. : modèles géométriques

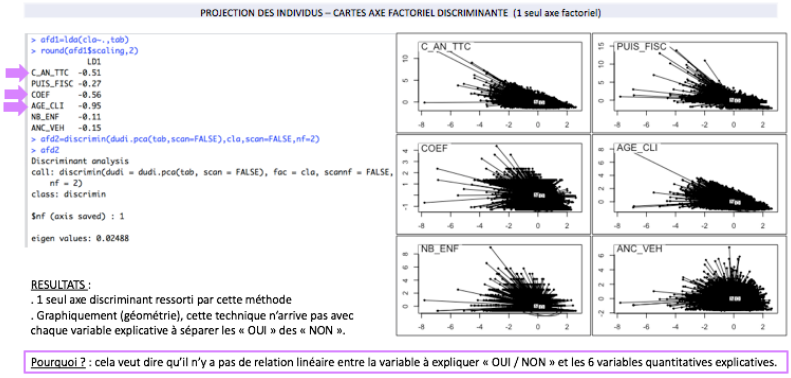
Modèle.1 : ACP (Analyse en Composantes Principales)



A retenir :
- Plan factoriel 1,2 cumule 60.71% de l'inertie du nuage initiale,
- Composante Principale 1 = l'axe de l'assuré ; Composante Principale 2 = l'axe du véhicule
- Axe 1 = Cotisation et Coefficient s'opposent à Age client,
- Axe 2 = Cotisation et Puissance Fiscale s'opposent à Ancienneté du Véhicule,
- la variable discriminante : cotisation (C_AN-TTC)

⚠ **Bonnes Pratiques DATALAB :**
- Pour mieux spécifier le modèle, « augmenter sa complexité au juste niveau » pour bien qualifier la relation Y [O/N] ~ Xi, l'ACP montre qu'il serait intéressant de positionner l'élément de tarification Matmut (Cotisation) avec des éléments de tarification concurrentes. Idéalement, il serait intéressant d'acheter des fichiers externes de simulations de devis tarifaires auprès des opérateurs les www.lesfurets.com / www.lelynx.fr (i.e. plateforme internet, multi-comparateur, assureurs automobiles et simulation de devis).

Modèle.2 : AFD (Analyse Factorielle Discriminante)



A retenir :
. Cette méthode factorielle de réduction de dimension, reposant sur la distance géométrique de Mahalanobis $\Delta_p^2 = (\mu_i - \mu_j)'\Sigma^{-1}(\mu_i - \mu_j)$. Dans le cas de 2 classes, le principe est de rechercher parmi la famille de toutes les ACP possibles, celle qui discrimine au mieux ces 2 classes (i.e. l'ACP qui classifie le mieux nos deux Groupes « les Oui / les Non » par rapport aux Xi. Pas de corrélations entre les Xi et l'axe factoriel discriminant).
➔ **Enseignement** : les 6 Xi retenues dans le modèle selectionné ne s'associent pas en combinaison linéaire pour former un axe discriminant (i.e. une frontière de décision linéaire) qui permettrait de vérifier les résultats d'une classification en 2 groupes des « Oui » et des « Non » (i.e. sur le critère de la variance : maximiser le variance inter-groupe et minimiser la variance intra-groupe).
Conclusion : 2 raisons possibles, défaut Xi discriminantes et la frontière de décision sur ces données serait quadratique (QDA ?).

Partie 2. : inférence statistique

partie 2.1. : modèle d'analyse discriminante (lda)

Modèle.3 : Lda (Analyse Linéaire Discriminante)

rappel 1 – conditions de mise en eouvre du lda:
. Hypothèse de multi-normalité des variables explicatives
. Hypothèse d'homoscédasticité entre la variable à expliquer et les variables explicatives (i.e. relation linéaire)

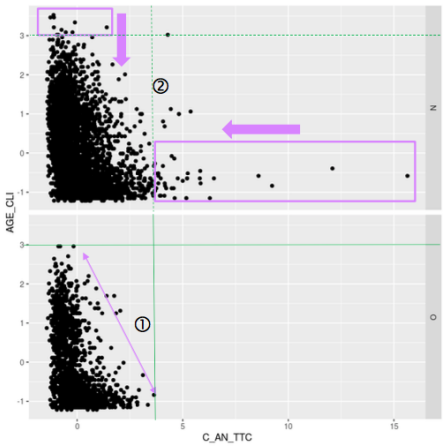
rappel 2 – pourquoi ce modèle lda:
. Lors d'une phase exploratoire d'enquête, s'assurer que le futur modèle d'apprentissage supervisé garantit que :
(1) le modèle soit bien spécifié (linéaire vs. quadratique),
(2) le modèle spécifié contienne bien les variables explicatives discriminantes de la variable d'intérêt.

résultats du modèle 2 Xi discriminantes : **Avec « prior » optimal (0.5;0.5)**	**. taux mal classés = 27.78% (important car le taux > à 10%)** **. taux AUC = 50% (cf. courbes ROC / Lift)** (cf. annexe : 10.6.2)

`Call:`
`lda(dq2norm_y$trsf_dev ~ C_AN_TTC + AGE_CLI, data = dq2norm_y,`
`prior = c(0.5, 0.5))`
 	
`Prior probabilities of groups:`
`N O`
`0.5 0.5`
 	
`Group means:`
`C_AN_TTC AGE_CLI`
`N 0.06174682 0.03479996`
`O -0.16048557 -0.09044824`
 	
`Coefficients of linear discriminants:`
`LD1`
`C_AN_TTC -1.0095862`
`AGE_CLI -0.7891436`
	`> confusionMatrix(data=prev, reference=dq2norm_y$trsf_dev,mode="prec_recall")`
	`Confusion Matrix and Statistics`
	
	`Reference`
	`Prediction N O`
	`N 1820577 700467`
	`O 0 0`
	
	`Accuracy : 0.7222`
	`95% CI : (0.7216, 0.7227)`
	`No Information Rate : 0.7222`
	`P-Value [Acc > NIR] : 0.5003`
	
	`Kappa : 0`
	`Mcnemar's Test P-Value : <2e-16`
	
	`Precision : 0.7222`
	`Recall : 1.0000`
	`F1 : 0.8387`
	`Prevalence : 0.7222`
	`Detection Rate : 0.7222`
	`Detection Prevalence : 1.0000`
	`Balanced Accuracy : 0.5000`
	
	`'Positive' Class : N`

Sur le critère de la PARCIMONIE, le modèle d'analyse discriminante (lda) à 2 variables est suffisant pour la phase « Machine Learning ».	Sur le critère de Performance de taux de biens classés, le modèle d'analyse discriminante (lda) à 2 variables n'est pas retenu pour la phase « Machine Learning ». Cause (probable) : des classes déséquilibrées de la variable à prédire → action : corriger le déséquilibre dans la spécification du modèle + hypothèse de mise en œuvre non respectée (?)
En période 2 : lors de la définition du dataset optimisé, le rejet de normalité des 6 Xi au test shapiro-wilks remet en cause l'Hypothèse de multi-normalité des variables explicatives Xi nécessaire au Lda → nous confirmons que ce classifieur ne peut être retenu sur ce modèle des données.	

En revanche : sur le critère pouvoir descriptif, les 2 Xi sélectionnées **Cotisation** (C_AN_TTC) et **Age Client** (AGE_CLI), mobilisées en tris croisé pour visualiser les groupes des individus « Oui » et « Non », ont permis d'apporter de la connaissance nouvelle au métier du marketing tarifaire de la Matmut :

VISUALISATION DES GRAPPES D'INDIVIDUS (cluster) avec modèle LDA sélectionné 2 variables explicatives	
	**Analyse en diagonale des « devis Non » et des « devis Oui »**: . résultat : [âge = vieux] en relation avec [cotisation = faible] alors étiquette = Non . résultat : [âge = jeune] en relation avec [cotisation = élevée] alors étiquette = Non

A retenir :
. la confirmation des résultats initialisés en Analyse Factorielle discriminante,
. la non normalité des Xi.

⚠ **Bonnes Pratiques DATALAB :**
. Avant d'industrialiser sur la plateforme Saagie, un modèle de « machine learning », il y a un travail incompressible en amont pour bien spécifier le modèle d'apprentissage : Vérifier les conditions de mise en œuvre du modèle, avec « les métiers » sélectionner uniquement les véritables variables discriminantes et visualiser graphiquement si le modèle a du sens dans « la vraie vie de l'entreprise ».

partie 2.2. : modèle de régression logistique (RegLog)

Modèle.4 : RegLog (Régression Logistique)

rappel 1 – conditions de mise en eouvre du modèle de Régression Logistique :
. Hypothèse de multi-normalité des variables explicatives
. Hypothèse d'homoscédasticité entre la variable à expliquer et les variables explicatives (i.e. relation linéaire)

L'hypothèse sous jacente au modèle logistique, est que

$$\log\left(\frac{f_1(\mathbf{x}^*)}{f_0(\mathbf{x}^*)}\right) = \tilde{\beta}_0 + \boldsymbol{\beta}^{\mathrm{T}}\mathbf{x}^*, \tag{8}$$

avec $\tilde{\beta}_0 \in \mathbb{R}$, $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_d)^{\mathrm{T}} \in \mathbb{R}^d$. En effet, par la formule de Bayes, on a $\pi_{\mathbf{x}^*} = \pi f_1(\mathbf{x}^*)/[\pi f_1(\mathbf{x}^*) + (1-\pi) f_0(\mathbf{x}^*)]$ et par suite,

$$\log\left(\frac{f_1(\mathbf{x}^*)}{f_0(\mathbf{x}^*)}\right) = \log\left(\frac{(1-\pi)\pi_{\mathbf{x}^*}}{\pi(1-\pi_{\mathbf{x}^*})}\right) = \log\left(\frac{1-\pi}{\pi}\right) + \beta_0 + \boldsymbol{\beta}^{\mathrm{T}}\mathbf{x}^*. \tag{9}$$

Ainsi, le modèle logistique vaut pour les situations où le rapport des lois marginales f_1 et f_0 peut être considéré à logarithme linéaire, comme cela est le cas par exemple lorsque f_1 et f_0 sont des distributions gaussiennes homoscédatiques. Plus généralement, cela est vérifié pour une large famille de distributions multivariées continues et discrètes (Anderson 1982).

[source : http://math.univ-lille1.fr/~biernack/index_files/Beninel-Biernacki.pdf - Relaxations de la régression logistique – P3/12]

Résultats du modèle retenu avec 6Xi discriminantes :

```
Call:  glm(formula = trsf_dev ~ C_AN_TTC + PUIS_FISC + COEF + AGE_CLI +
    NB_ENF + ANC_VEH, family = binomial, data = dq2norm_y)

Coefficients:
(Intercept)    C_AN_TTC   PUIS_FISC        COEF     AGE_CLI      NB_ENF     ANC_VEH
   -0.98966    -0.21440    -0.15202     -0.16488    -0.23867    -0.07483     0.10822

Degrees of Freedom: 2521043 Total (i.e. Null);  2521037 Residual
Null Deviance:      2979000
Residual Deviance: 2912000      AIC: 2912000
```

CONCLUSION : sur le critère AIC min, le modèle régression logistique à 2 variables n'est pas le meilleur modèle → on garde le modèle à 6 variables Xi.
Résultat : ce modèle RegLog n'est pas bien spécifié pour ce modèle des données ➔ pas de garantie d'avoir sourcé toutes les variables explicatives réellement discriminantes et interrogation sur la nature linéaire entre les variables explicatives et la variable à expliquer (var. delai_1_ctc de Cyrille et mobiliser d'autres modèles tels que Logit / t-SNE « représentation graphique » pour croiser les résultats des différents modèles, et organiser une confrontation des résultats pour dégager un modèle « main stream adapté à notre problème ») – **on retrouve « la violation des hypothèses » de mise en œuvre du modèle régression logistique (i.e. normalité de variables explicatives / scénario identique au modèle lda)**

En période 2 :
lors de la définition dataset optimisé, le rejet de normalité des 6 Xi au test shapiro-wilks remet en cause l'Hypothèse de multi-normalité des variables explicatives Xi nécessaire au RegLog → nous confirmons que ce classifieur ne peut être retenu sur ce modèle des données.

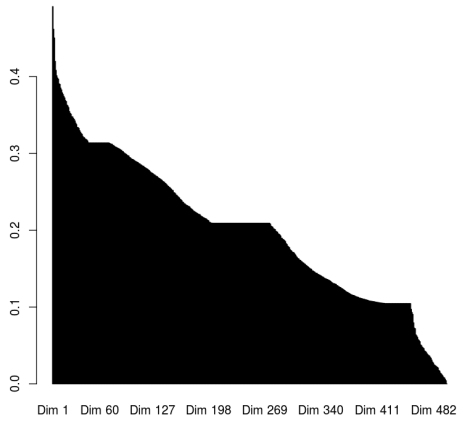
Modèle. autres : ACM sur variables qualitatives (Analyse en Composantes Multiples)

. Ce modèle ACM fait partie des modèles d'analyse factorielle descriptive, mais couplé avec un modèle CAH (classification hiérarchique ascendante sur les axes factorielles issus de l'ACM et vus comme de variables numériques), il permet de faire de l'inférence statistique.

. La mobilisation de l'ACM a été réalisée pour confirmer un a priori initial pointé lors de la spécification du modèle d'apprentissage avec l'expert métier Benjamin Lamarre : historiquement, les variables dites « qualitatives » qui renseignent le sociétaire sont mal documentées dans le Système d'information Matmut. Par expérience, les enquêtes Matmut avec ce type de variable sont difficilement exploitables.

A noter : chronologiquement, ce modèle a été réalisé en premier pour identifier le type de variable explicative à mobiliser sur ce modèle des données MRSQ => **Conclusion : retenir les Xi quantitativement continues qui sont historiquement mieux renseignées à la Matmut.**

Résultats du modèle ACM



eigenvalue	percentage of variance	cumulative percentage of variance	
dim 1	0.34	0.49	0.49
dim 2	0.32	0.46	0.95
dim 3	0.31	0.45	1.40
dim 4	0.29	0.42	1.82
dim 5	0.28	0.41	2.23

ANALYSE : choix du nombre d'axe selon la méthode du coude ?
. la faible valeur d'inertie récupérée par les 5 premiers axes factoriels, pas de possibilité de construire des cartes factorielles.
. graphiquement, pas de coude identifié.

CONCLUSION :
. on se trouve dans l'impossibilité de choisir le nombre d'axe factoriel et donc de produire une analyse factorielle pour construire des relations entre la variable qualitative trsf_dev et les autres variables.
=> Sur le jeu de données MRSQ, il est difficile d'expliquer clairement le rôle des variables qualitatives sur la variable transformation des devis.

⚠ Bonnes Pratiques DATALAB :

. Avant d'industrialiser sur la plateforme Saagie, un modèle de « machine learning », il y a un travail incompressible en amont pour bien spécifier le modèle d'apprentissage : Vérifier les conditions de mise en œuvre du modèle, avec « les métiers » sélectionner uniquement les véritables variables discriminantes et visualiser si le modèle a du sens dans « la vraie vie de l'entreprise ».

. Dans le lot des tâches du travail incompressible en amont, il est important au regard de la qualité des données de conserver dans le modèle d'apprentissage uniquement les variables bien renseignées (les avantages : dataset moins volumineux, temps de calcul moins coûteux, robustesse et lisibilité de résultats).

partie 2.3. : les autres modèles (Xgboost et RF)

modèle.5 : RF (Random Forest : Les Forêts Aléatoires)

Rappel méthodologique : classifieur important qui appartient à la famille des algorithmes d'apprentissage automatique. Les forêts aléatoires proposés par Leo Breiman (2001) constituent un algorithme pertinent pour détecter les liens entre les variables explicatives et la variable d'intérêts. C'est un algorithme où « tout est aléatoire et qui fonctionne en boîte noire », mais dont le principe est d'effectuer un apprentissage statistique sur des multiples arbres entrainés sur des sous-ensembles (fraction) de données qui différent à chaque fois.

En pratique : l'algorithme RF génère une multitude d'arbres de classification sur une fraction aléatoire des données puis il les fait voter.
De ce vote, on récupère :
. un classement par ordre d'importance des variables explicatives,
. un taux d'erreur interne de prédiction appelé **« erreur OOB = Out of Bag »**,
. le but de ce classifieur est de minimiser cette erreur OOB en jouant sur les 2 paramètres du modèles :
- le nombre d'arbres construits par RF (**appelé = ntree**),
- le nombre de variables explicatives à tester à chaque division (**appelé = mtry**).

. L'optimisation de ce « pur » algorithme de machine learning se fait selon une procédure classique de validation croisé apprentissage / test, en maximisant les critères de taux de bien classés (Accuracy) et de Kappa (comme l'Accuracy mais qui prend en compte le déséquilibre des classes) ➔ **la performance du modèle se mesure** : plus ces 2 critères affichent des valeurs importantes, meilleure sera la prédiction du modèle spécifié (i.e. on dit « tuned » le modèle)

Pseudo – code de l'algorithme :
« … La base du calcul repose sur l'apprentissage par arbre de décision. La proposition de Breiman vise à corriger plusieurs inconvénients connus de la méthode initiale, comme la sensibilité des arbres uniques à l'ordre des prédicteurs, en calculant un ensemble **B** arbres partiellement indépendants ... »

Une présentation rapide de la proposition[3] peut s'exprimer comme suit :

1. Créer B nouveaux ensembles d'apprentissage par un double processus d'échantillonnage :
    1. sur les observations, en utilisant un tirage avec remise d'un nombre N d'observations identique à celui des données d'origine (technique connue sous le nom de bootstrap),
    2. et sur les p prédicteurs, en n'en retenant qu'un échantillon de cardinal $m < \sqrt{p}$ (la limite n'est qu'indicative).
2. Sur chaque échantillon, on entraîne un arbre de décision selon une des techniques connues, en limitant sa croissance par validation croisée.
3. On stocke les B prédictions de la variable d'intérêt pour chaque observation d'origine.
4. La prédiction de la forêt aléatoire est alors un simple vote majoritaire *(Ensemble learning)*.

Le principal revers de cette méthode est que l'on perd l'aspect visuel des arbres de décision uniques.

[source : https://fr.wikipedia.org/wiki/For%C3%AAt_d%27arbres_d%C3%A9cisionnels]

Avantage :
. Coupler ce classifieur (pour la sélection des Xi discriminantes) avec des méthodes factorielles pour décrire graphiquement la relation des Xi avec la variable Y (i.e. expliquer intuitivement).
. Dans les compétitions de data sciences Kaggle.com, ce classifieur en configuration de problème de classification binaire supervisée affiche de bonnes performances.

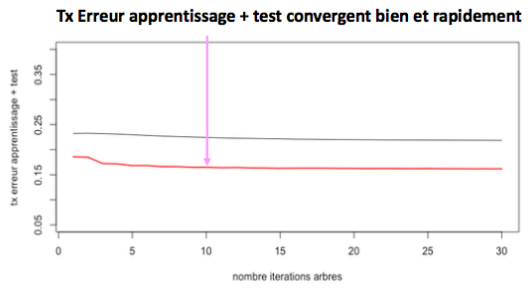
Inconvénient : contrairement à un modèle SVM « où l'analyste supervise les paramètres du modèle », le fonctionnement « boîte noire » de RF, limite l'action de l'analyste « qui applique » un package sur son jeu de données.

(test) – erreur OOB = 16.18%

Pour : ntree = 30, mtry = 6, avec 6 Xi dans le modèle RF

```
        OOB estimate of  error rate: 21.85%
Confusion matrix:
        N      O class.error
N 1457868 180776   0.1103205
O  314882 315411   0.4995804
                Test set error rate: 16.18%
Confusion matrix:
        N     O class.error
N 169509 12767  0.07004213
O  28024 41804  0.40132898
```

83.82 % = (100 -16.18) %

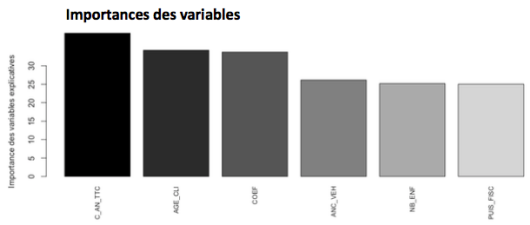


A RETENIR :

. randomForest Modèle prédictif le moins « overfitted » et qui « généralise bien ».

PREUVE :

. Delta Erreur OOB (Apprentissage - Test)
= 21.85% - 16.18% = 5.67 % (proche 5%)
. Gain (Tx bien classés - Tx Distribué [NON])
= 83.82% - 73.31% = 10.51% (sup à 10%)



Var. discriminantes = idem autres modèles stats.

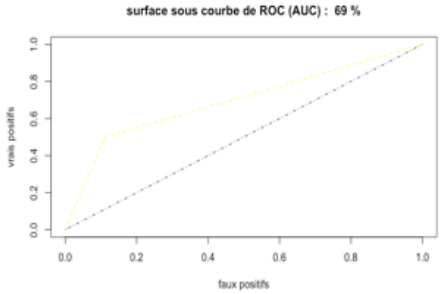
Analyse :
. le modèle RF n'a pas besoin de beaucoup d'arbres pour produire une erreur OOB en test de = 16.18%
. le RF à 6 Xi généralise bien (les courbe d'erreur en test ne remonte pas)
. les 2 Xi les plus importantes sont : Cotisation et Age Client.

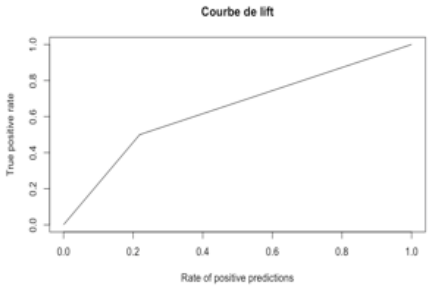
Pour : ntree = 30, mtry = 6, avec 6 Xi dans le modèle RF – metrics de performance
Procédure simple de validation croisée : poids échantillon apprentissage = 0.7, poids échantillon test = 0.3

sur données test

```
               Accuracy : 0.7811
                 95% CI : (0.7806, 0.7817)
    No Information Rate : 0.7222
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.4165
 Mcnemar's Test P-Value : < 2.2e-16

            Precision : 0.8222
               Recall : 0.8893
                   F1 : 0.8544
           Prevalence : 0.7222
       Detection Rate : 0.6423
 Detection Prevalence : 0.7812
    Balanced Accuracy : 0.6946
```

A RETENIR :
. Performance Moyenne, Courbes ROC - LIFT





Taux de biens classés : 0.7811 = 78.81%
Kappa : 41.65%
F1_mesure : 85.44% (parmi les biens classés, le modèle détecte 85.44% les Vrais Positifs)
AUC : 69%

Analyse : les valeurs affichées font de ce modèle RF avec 6 xi continues le classifieur de référence.

modèle.6 : Xgboost (Gradient Boosting)

Présentation de l'algorithme : dans la famille des modèles adaptatifs, le boosting.

Selon Philippe Besse (Insa 31) – Agrégation de modèles

Le boosting adopte le même principe général que le bagging : construction d'une famille de modèles qui sont ensuite agrégés par une moyenne pondérée des estimations ou un vote. Il diffère nettement sur la façon de construire la famille qui est dans ce cas récurrente : chaque modèle est une version adaptative du précédent en donnant plus de poids, lors de l'estimation suivante aux observations mal ajustées ou mal prédites. Intuitivement, cet algorithme concentre donc ses efforts sur les observations les plus difficiles à ajuster tandis que l'agrégation de l'ensemble des modèles réduit le risque de sur-ajustement.

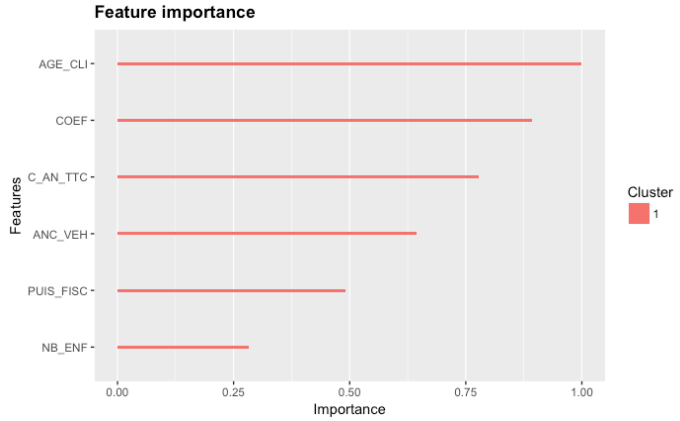
Les algorithmes de boosting proposés diffèrent par différentes caractéristiques :
• la façon de pondérer c'est-à-dire de renforcer l'importance des observations mal estimées lors de l'itération précédente,
• leur objectif selon le type de la variable à prédire Y : binaire, qualitative à k classes, réelles;
• la fonction perte, qui peut être choisie plus ou moins robuste aux valeurs atypiques, pour mesurer l'erreur d'ajustement;
• la façon d'agréger, ou plutôt pondérer, les modèles de base successifs.

La littérature sur le sujet présente donc de très nombreuses versions de cet algorithme **dont le dernier avatar, proposé dans une librairie XGBoost (extrem gradient boosting) connaît beaucoup de succès dans les concours de prévision Kaggle…**

[source : Philippe BESSE - Agrégation de modèles - https://www.math.univ-toulouse.fr/~besse/Wikistat/pdf/st-m-app-agreg.pdf]

XGBOOST = MODELE COMPLET 6 VARIABLES QUANTITATIVES > Taux BIEN CLASSES = 76.25%

```
niter: 10
evaluation_log:
    iter train_merror test_merror
       1     0.238359    0.238548
       2     0.237093    0.237128
---
       9     0.237444    0.237581
      10     0.237447    0.237581
```



ANALYSE :
. sur le critère de Performance de taux de biens classés, le modèle Gradient Boosting (packag XGboost) à 6 variables est moins performant que le modèle de forêt aléatoire (Random Forest) avec les paramètres d'algorithme par défaut.
. sur le critère d'importance des « features », le socle de variables discriminantes du modèle ne converge par vers le socle convergent des autres modèles.

➔ ACTION : Algorithme long à paramétrer, arbitrage temps de paramétrage du modèle optimal XGboost vs. gain de taux de bien classés obtenu ?

CONCLUSION :
. ce modèle XGboost n'est pas retenu.

*** CONCLUSION GENERALE DE CETTE PREMIERE PERIODE DE CALCUL ***

. Sur notre modèle de données MRSQ, avec classes déséquilibrées, **le classifieur Random Forest avec 6 variables explicatives Xi continues est le classifieur le plus performant sur 3 « metrics » :**

- Accuracy (le taux de biens classés calculé à partir de la matrice de confusion),
- l'AUC, (la qualité de la prévision, est-ce que la prévision de RF est-elle supérieure à la performance du hasard)
- La F-mesure, (capacité du RF parmi les individus bien classés à détecter les Vrai Positifs)

➔ **Ce classifieur des forêts aléatoires constitue notre modèle de référence auprès duquel nous testerons la performance du modèle SVM.**

8.2. Période 2 : résultats du modèle RF avec grid search

8.2.1. Analyse descriptive du « dataset optimisé »

Pourquoi un dataset optimisé : L'objectif de préparer un « dataset optimisé » est de vérifier que tous les prérequis concernant les données sont réalisés avant de tester et de comparer la performance des classifieurs Random Forest et SVM en paramétrage « optimisé ».

En particulier, un travail attentif sera réalisé pour vérifier la question de normalité ou non des 6 variables explicatives quantitatives et continues qui ont participé à la construction de notre modèle de référence « Random Forest » à la période n°1 de calculs numériques.

(1) Taux de valeurs manquantes (nettoyé)

```
# [1] 728674 lignes x 10 colonnes

# 3. étude des valeurs manquantes
Pourcentage_NA = colMeans(is.na(dv16))
Pourcentage_NA = Pourcentage_NA * 100
Pourcentage_NA
```

```
##  ID_CLIENT    NUM_CTC    NUM_DEV   C_AN_TTC      COEFF  PUIS_FISC
##  0.0000000  0.0000000  0.0000000  0.0000000  0.6631223  0.6491243
##    AGE_CLI     NB_ENF   trsf_dev    ANC_VEH
##  3.9983587 13.7212251  0.0000000  0.0000000
```

```
# Analyse : taux de valeurs manquantes (%)
# COEFF      =  0.66%
# PUIS_FISC  = 0.64%
# AGE_CLI    = 3.99%
# NB_ENF     = 13.72%
# sans surprise les var Xi concernant les données personnelles de l'assuré,
# affichent les taux de Na les plus élevés (AGE CLIENT et NOMBRE D ENFANT)

summary(Pourcentage_NA)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0000  0.0000  0.0000  1.9032  0.6596 13.7212
```

```
mean(Pourcentage_NA)
```

```
## [1] 1.903183
```

(2) Vue générale

- les variables explicatives

```
##    ID_CLIENT            NUM_CTC          NUM_DEV            C_AN_TTC
##  Min.   :211978166   Min.   : 1.00   Min.   :  1.000   906,30 :  1466
##  1st Qu.:212234208   1st Qu.: 1.00   1st Qu.:  1.000   821,30 :  1419
##  Median :212379122   Median : 1.00   Median :  2.000   442,30 :  1394
##  Mean   :212395481   Mean   : 1.61   Mean   :  2.145   811,30 :  1370
##  3rd Qu.:212567554   3rd Qu.: 2.00   3rd Qu.:  2.000   735,30 :  1227
##  Max.   :212760888   Max.   :17.00   Max.   :318.000   783,30 :  1204
##                                                        (Other):605968
##      COEFF          PUIS_FISC          AGE_CLI           NB_ENF
##  Min.   : 35.00   Min.   : 0.000   Min.   : 13.00   Min.   : 0.0000
##  1st Qu.: 40.00   1st Qu.: 4.000   1st Qu.: 23.00   1st Qu.: 0.0000
##  Median : 65.00   Median : 5.000   Median : 32.00   Median : 0.0000
##  Mean   : 66.27   Mean   : 5.577   Mean   : 36.88   Mean   : 0.8838
##  3rd Qu.: 95.00   3rd Qu.: 6.000   3rd Qu.: 49.00   3rd Qu.: 2.0000
##  Max.   :256.00   Max.   :68.000   Max.   :168.00   Max.   :20.0000
##
##  trsf_dev      ANC_VEH
##  N:450163   Min.   :-1.000
##  O:163885   1st Qu.: 5.000
##             Median : 9.000
##             Mean   : 9.495
##             3rd Qu.:14.000
##             Max.   :60.000
```

- la variable d'intérêt (fréquence Oui / Non)

```
#   N         O
#   73.31072 26.68928
#   dataset avec classes déséquilibrées (les NON majoritaires)
```

(3) Relation Yi ~ Xi

```
# analyse :
# . la variable d'intérêt transformation devis est en relation avec tous les Xi quantitatives explicatives.
# . la var Xi C_AN_TTC a une relation plus important avec trsf_dev, en comparaison avec les autres Xi
```

Pour plus de détails, voir les diagramme en annexes : 10.7. Calculs numériques - Modélisations Période 2

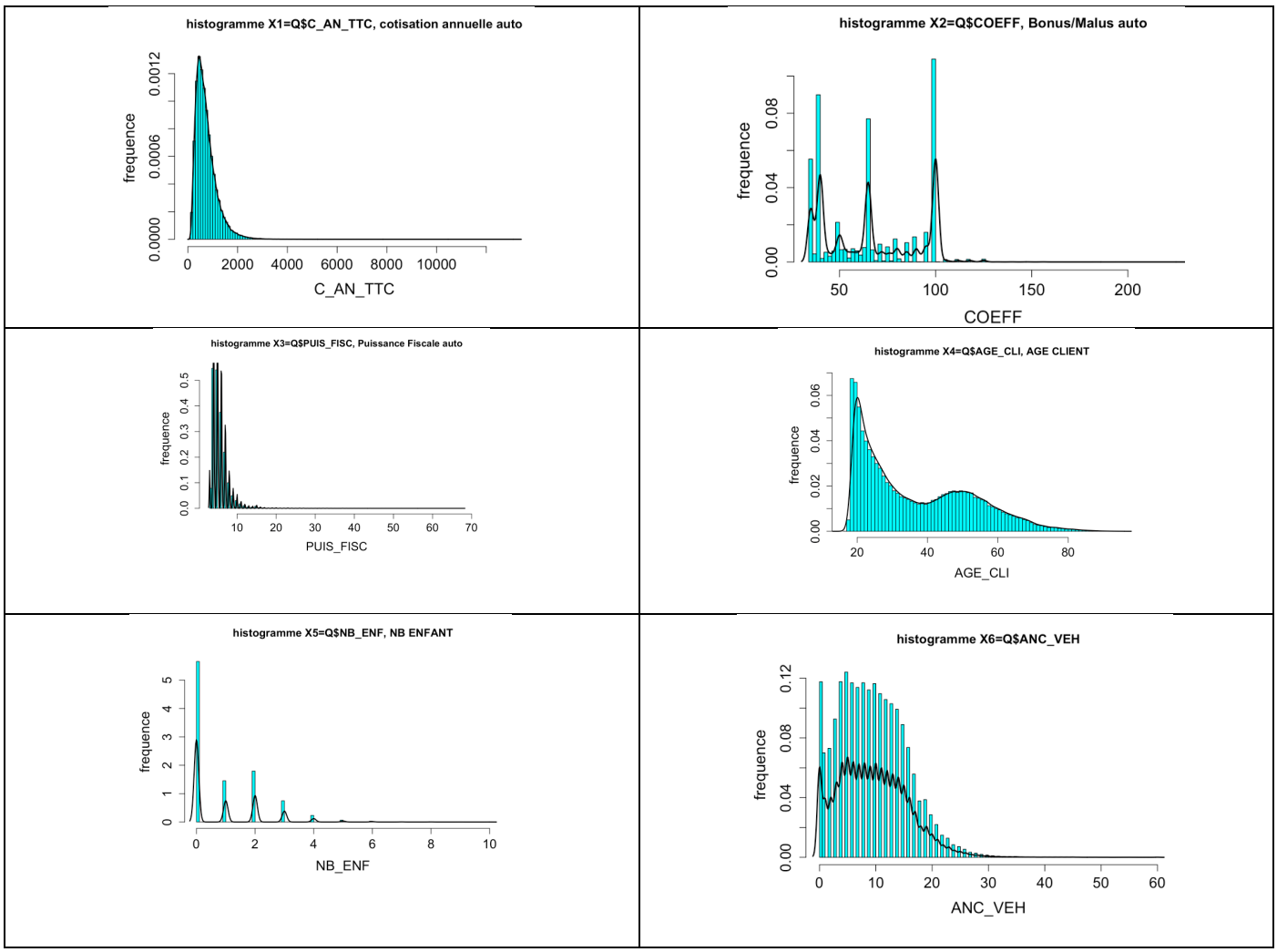
(4) Indicateur de dispersion / répartition

- matrice de variance / covariance

```
##              C_AN_TTC      COEFF   PUIS_FISC      AGE_CLI       NB_ENF
## C_AN_TTC  179468.7379 6109.84306 205.6420799 -2451.297652 -121.3654449
## COEFF       6109.8431  613.70555 -11.0109672  -235.350965  -12.0094598
## PUIS_FISC    205.6421  -11.01097   5.2168244     6.913562    0.4504491
## AGE_CLI    -2451.2977 -235.35097   6.9135624   246.338006    9.9166648
## NB_ENF      -121.3654  -12.00946   0.4504491     9.916665    1.4893831
## ANC_VEH     -466.3874   23.82561   0.5895995   -11.069806   -0.2369642
##                ANC_VEH
## C_AN_TTC  -466.3874120
## COEFF       23.8256108
## PUIS_FISC    0.5895995
## AGE_CLI    -11.0698060
## NB_ENF      -0.2369642
## ANC_VEH     38.3683223
```

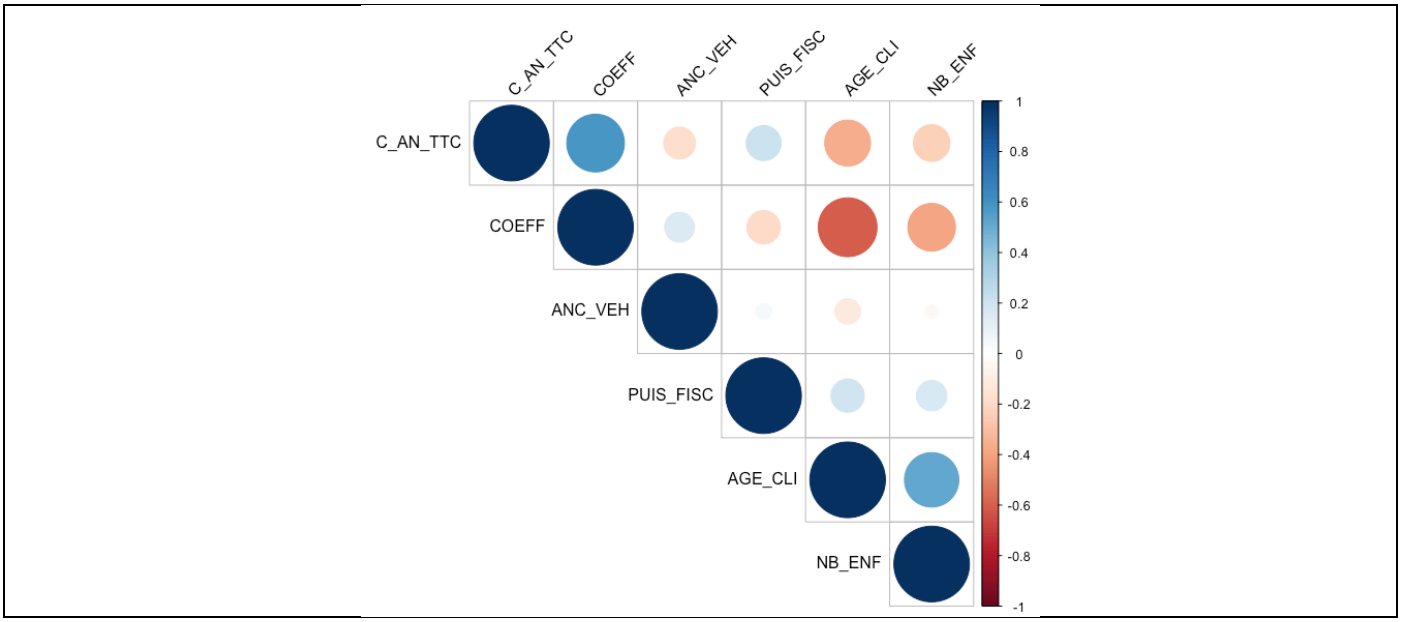
A noter : ***la var C_AN_TTC a une très grosse variance par rapport aux autres Xi***

- Histogramme



(5) Indicateurs Corrélations et associations & Graphiques de statistiques bivariées

- matrice des corrélations entre les Xi :




```
# ANALYSE :
# . la Cotisation Annuelle est corrélée avec le Coefficient B/M,
# . le Coefficient est corrélé avec Age Client.
# .  Age Client avec NB d'enfant.
# COMMENTAIRE :
# . ces éléments sont illustratifs du modèle tarifaire MATMUT pour un contrat Auto.
```

- comparer graphiquement les Oui / Non sur chaque Xi

- estimer les statistiques de bases par sous-populations

[Pour plus de détails voir les diagramme en annexes : 10.7. Calculs numériques - Modélisations Période 2]

(6) Tests statistiques

→ Tests d'adéquation a une loi donnée pour variable quantitative : test de normalité (Shapiro-Wilks)

- pour les 6 Xi, pour les 2 sous-populations « OUI » et « NON », on rejette la normalité des 6 Xi,
- **conclusion : il faut normaliser les 6 Xi avec une transformation de variable.**

→ Test de comparaison de populations :

- test d'égalité de variance
- pour les 6 Xi, la variance est significativement différente entre les groupes des "Oui" et des "Non" avec un intervalle de confiance à 95%.

- test d'égalité de moyenne
- les 6 Xi discriminent bien la variable à expliquer taux de transformation (Oui / Non).

(7) Détection des points aberrants (Outliers)

- Le test du Chi-Deux pour les valeurs extrêmes : La fonction réalise un test simple pour détecter la présence d'une valeur extrême. Le test est basé sur une distribution du chi-deux des carrés des différences entre les données et la moyenne de l'échantillon. La variance de la population est supposée connue.
- pour les 6 Xi, on rejette la présence « d'outliers ».

(8) Normalisation des Xi

---

transformation ie changement de var pour rendre certaines distributions plus symétriques et plus gaussiennes.

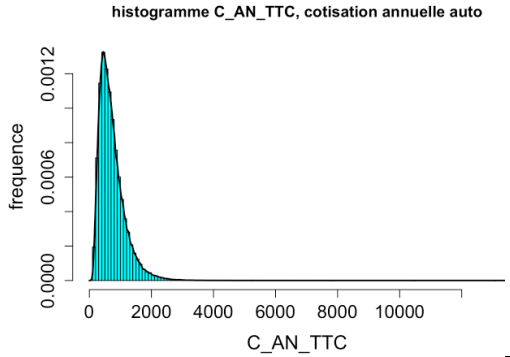
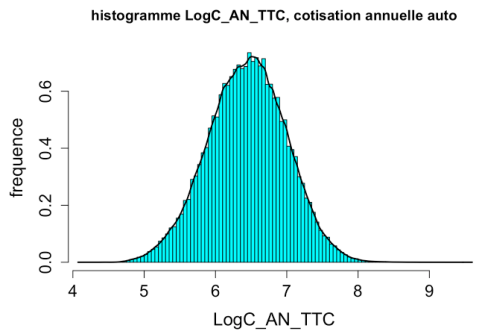
Sknewness

Une valeur nulle du Sknewness indique une distribution symétrique des données autour de leur moyenne alors qu' un Skewness positif (respectivement négatif) indique une queue distribution plus étalée vers la droite (resp. vers la gauche).

Kurtosis

Plus le Kurtosis est élevé, plus il y a de valeurs éloignées de la moyenne, les queues de distributions sont plus épaisses (que celles de la gaussienne, dont le kurtosis est 3).

Pour les 6 (Xi) quantitatives et continues → les transformations envisagées : log(Xi), $(Xi)^2$, $e^{(Xi)}$, $\sqrt{(Xi)}$, $\frac{1}{(Xi)}$

Variable cotisation (C_AN_TTC) : LogC_AN_TTC (passage en logarithme décimal)	
Avant	Après
	
Test Shapiro-Wilks	
# test de normalité des "Oui" avec LogC_AN_TTC # p-value = 0.14 # analyse : valeur p-value > 5%, on accepte la normalité de la var **LogC_AN_TTC pour les "Oui".**	# test de normalité des "Non" avec LogC_AN_TTC # p-value = 0.23 # analyse : valeur p-value > 5%, on accepte la normalité de la var **LogC_AN_TTC pour les "Non".**
Transformation testée	**Résultat obtenu**
COEFF BONUS MALUS > carré (SqCOEFF)	conclusion : skewness SqCOEFF réduit, on conserve COEFF **Normalité rejetée**
PUISSANCE FISCAL > carré (SqPUIS_FISC)	conclusion : skewness SqPUIS_FISC réduit, on conserve PUIS_FISC **Normalité rejetée**
AGE CLIENT > carré (SqAGE_CLI)	conclusion : skewness SqAGE_CLI réduit, on conserve AGE_CLI **Normalité rejetée**
NOMBRE D ENFANT > carré (SqNB_ENF)	conclusion : skewness SqNB_ENF réduit, on conserve NB_ENF **Normalité rejetée**
ANCIENNETE VEHICULE > carré (SqANC_VEH)	conclusion : on conserve ANC_VEH **Normalité rejetée**

Conclusion sur la NORMALISATION des 6 Xi :

- on accepte la variable transformée Log(C_AN_TTC) comme une variable normalisée,
- on conserve les 5 autres variables : COEFF, PUIS_FISC, AGE_CLI, NB_ENF, ANC_VEH comme des variables non normalisées.

Remarque : cette conclusion que seulement 1/6 Xi est normalisée est importante pour l'optimisation du classifieur SVM.

[Pour plus de détails voir les diagramme en annexes : 10.7. Calculs numériques - Modélisations Période 2]

(9) Centrage et réduction des Xi quantitatives

```
##   LogC_AN_TTC          COEFF            PUIS_FISC          AGE_CLI
##  Min.   :-4.15148   Min.   :-1.26197   Min.   :-1.1648   Min.   :-1.3301
##  1st Qu.:-0.67751   1st Qu.:-1.06014   1st Qu.:-0.7270   1st Qu.:-0.8841
##  Median : 0.01818   Median :-0.05098   Median :-0.2892   Median :-0.3107
##  Mean   : 0.00000   Mean   : 0.00000   Mean   : 0.0000   Mean   : 0.0000
##  3rd Qu.: 0.68606   3rd Qu.: 1.16001   3rd Qu.: 0.1487   3rd Qu.: 0.7725
##  Max.   : 5.58588   Max.   : 6.40764   Max.   :27.2936   Max.   : 3.7033
##      NB_ENF            ANC_VEH
##  Min.   :-0.7243   Min.   :-1.53296
##  1st Qu.:-0.7243   1st Qu.:-0.72575
##  Median :-0.7243   Median :-0.07999
##  Mean   : 0.0000   Mean   : 0.00000
##  3rd Qu.: 0.9145   3rd Qu.: 0.72722
##  Max.   : 7.4697   Max.   : 8.15350
```

```
sd(dq2normCR$LogC_AN_TTC) # [1] 1
```

```
## [1] 1
```

```
sd(dq2normCR$COEFF)       # [1] 1
```

```
## [1] 1
```

```
sd(dq2normCR$PUIS_FISC)   # [1] 1
```

```
## [1] 1
```

```
sd(dq2normCR$AGE_CLI)     # [1] 1
```

```
## [1] 1
```

```
sd(dq2normCR$NB_ENF)      # [1] 1
```

```
## [1] 1
```

```
sd(dq2normCR$ANC_VEH)     # [1] 1
```

```
## [1] 1
```

⚠ **<u>Bonnes Pratiques DATALAB</u> :**

- Pour notre travail sur le classifieur SVM, cette transformation de centrage-réduction de chaque Xi est importante pour réduire le temps de calcul (i.e. appelée également standardisation).

(10) Formatage d'un échantillon par Sondage Stratifié Proportionnel

Effectif total = 614928 lignes,
Effectif SSP = 6150 lignes,
Taux de prélèvement **= 1.0001 %**

Taille du dataset = 6150 lignes x 7 variables

<u>Contrôle de l'échantillon sondage obtenu</u> :

Yi : distribution de la variable d'intérêt trsf_dev

En poids	*En fréquence*
`##  N  O` `## 4491 1659`	`##  N  O` `## 73.02439 26.97561`

Xi : sur chaque Xi, audit des statistiques descriptives (mean, sd, var, quantile, skewness, kurtosis, density)

Pour chaque Xi de l'échantillon SSP, le centrage-réduction moyenne est vérifié (xi) ~ 0 et écart-type ~ 1

[Pour plus de détails voir les diagramme en annexes : 10.7. Calculs numériques - Modélisations Période 2]

(11) Vérification des matrices variance et corrélation, chez les « OUI » et chez les « NON » après normalisation :

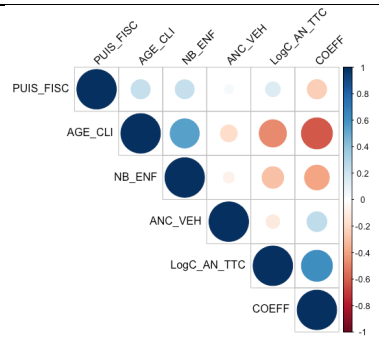
Matrice de variance –covariance chez les « Oui »

	LogC_AN_TTC	COEFF	PUIS_FISC	AGE_CLI	NB_ENF	ANC_VEH
LogC_AN_TTC	0.79756677	0.5512906	0.09762519	-0.4156359	-0.26535431	-0.10285899
COEFF	0.55129056	0.9976945	-0.18634862	-0.6031884	-0.39231639	0.24416417
PUIS_FISC	0.09762519	-0.1863486	0.60407534	0.1793368	0.17880981	0.03703679
AGE_CLI	-0.41563590	-0.6031884	0.17933682	0.9537194	0.52928915	-0.17267949
NB_ENF	-0.26535431	-0.3923164	0.17880981	0.5292891	0.99467055	-0.07383266
ANC_VEH	-0.10285899	0.2441642	0.03703679	-0.1726795	-0.07383266	0.93542646

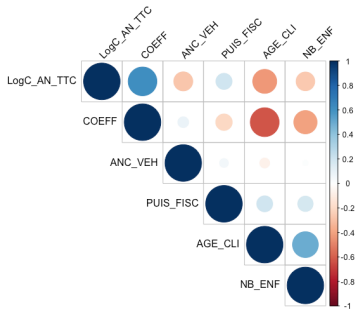
Matrice de variance –covariance chez les « Non »

	LogC_AN_TTC	COEFF	PUIS_FISC	AGE_CLI	NB_ENF	ANC_VEH
LogC_AN_TTC	1.0293175	0.62685408	0.20358283	-0.43553260	-0.27690087	-0.27579527
COEFF	0.6268541	1.00477734	-0.21099902	-0.62126485	-0.42356123	0.08703313
PUIS_FISC	0.2035828	-0.21099902	1.07168064	0.19637382	0.19068945	0.05976262
AGE_CLI	-0.4355326	-0.62126485	0.19637382	0.99645489	0.51872845	-0.07325241
NB_ENF	-0.2769009	-0.42356123	0.19068945	0.51872845	1.08725156	0.01894466
ANC_VEH	-0.2757953	0.08703313	0.05976262	-0.07325241	0.01894466	1.00823367

Matrice des corrélations chez les « Oui »



Matrice de corrélations chez les « Non »



Analyse :

. La situation est nominale, on retrouve les conclusions des tests d'égalité de moyenne / variance des Xi entre les OUI/NON de notre variable d'intérêt,

. Ces résultats plaident pour une frontière de séparation quadratique entre les 2 sous-populations étudiées.

CONCLUSION :

. Nous disposons d'un « dataset réduit et vérifié » pour tester la performance des classifieurs Random Forest et SVM.

➔ Vue sous R – extrait du dataset [6150 x 7] :

Filter

	LogC_AN_TTC	COEFF	PUIS_FISC	AGE_CLI	NB_ENF	ANC_VEH	trsf_dev
535414	-1.200465453	-1.26197174	-0.2891568	1.473327080	0.09510035	1.21154008	N
34174	-1.091659617	-0.57574301	-0.7269781	-0.501804271	-0.72430122	-0.56430992	N
458574	-1.091659617	-1.06013976	1.0243071	1.600754910	1.73390350	0.56577645	N
569322	0.429022087	-0.13171266	0.1486645	0.007907046	0.09510035	-0.56430992	N
38324	-1.524655874	-0.65647580	-0.7269781	-0.119520784	-0.72430122	1.21154008	O
304468	0.831953564	1.36184399	-0.2891568	-0.947801673	0.09510035	0.88865827	O
300381	0.641561077	1.07927922	-0.2891568	-0.119520784	-0.72430122	-0.56430992	N
434669	0.999338216	1.36184399	-0.2891568	-1.138943417	-0.72430122	0.24289463	N
446978	-0.547479517	-0.65647580	-0.7269781	2.174180141	2.55330508	-1.53295538	N
466455	0.507794198	-0.37391103	0.5864858	-0.947801673	-0.72430122	-1.37151447	O

8.2.2. Performance du modèle des forêts aléatoires

*** RESULTATS NUMERIQUES ***

PERIODE 2

SYNTHESE PERFORMANCE DES CLASSIFIEURS RF

hypo : relation linéaire Yi ~ Xi	5/6 des Xi non
Nb total d'observation	6150
Echantillonnage	1%

Etiquette Yi	trsf_dev		
Poids des Yi[i]	[N;O] = [72.95 ; 27.04]		
Nb Xi	6		
Détail Xi	LogC_AN_TTC + PUIS_FISC + COEF + AGE_CLI + NB_ENF + ANC_VEH		
standardisation Xi	centrage-réduction de Xi ~ [m ~ 1; var ~ 0]		
normalisation Xi	C_AN_TTC -> LogC_AN_TTC		
calcul parallèle	3-cpu	3-cpu	3-cpu
Langage Prog - package	R-caret	R-caret	R-caret

METRICS	RF "random search"	RF "grid search - accuracy"	RF "grid search - ROC"
apprentissage	0.7	0.7	0.7
test	0.3	0.3	0.3
Correction Poids des Yi[i]	wts=100/table(training$trsf_dev)	wts=100/table(training$trsf_dev)	wts=100/table(training$trsf_dev)
k (cv- k folds)	10	10	10
répétition	10	10	10
méthode opti paramètre	Random + parallel	Grid + parallel	Grid + parallel
critère de performance	accuracy	accuracy	roc
type : classif / reg.	classif	classif	classif
ntree (nb d'arbre)	~ 350	~ 250	~ 350
mtry (nb var / nœud)	mtry = 2	mtry = 1	mtry = 2
temps de calcul	11,41 mins	5,32 mins	5,43 mins
Tx de biens classés (Accuracy)	0,7223	0,7397	0,7131
Tx d'Erreur	0,2777	0,2603	0,2869
AUC	0,6140	0,5799	0,6143
Kappa	0,0856	0,0148	0,0560
F1 (F-mesure)	0,7534	0,7411	0,7485
Contrôle Graphique	oui	oui	oui
importance des descripteurs 1	LogC_AN_TTC	LogC_AN_TTC	**LogC_AN_TTC**
valeur	100,00	100,00	100,00
importance des descripteurs 2	AGE_CLI	AGE_CLI	**AGE_CLI**
valeur	58,45	72,01	57,11

valeur maximum sur le critère

modèle retenu

Tableau 3 - synthèse de la performance du classifieur Random Forest période de calcul 2

*** Synthèse des résultats ***

But : construire un classifieur RF optimisé comme modèle de référence pour un objectif de précision de classification, afin de le comparer au modèle SVM optimisé sur notre jeu de données.

Résultats : sur ce jeu de données, le modèle retenu est **le RF "grid search - ROC" .**

→ Résultat 1 : le classifieur retenu est celui dont le paramétrage d'optimisation correspond le mieux à notre problème de classification,
→ Résultat 2 : les valeurs empiriques AUC (0.61/P2 vs 0.69/P1) et F-mesure (0.74/P2 vs 0.85/P1) sont en écarts contenus,
→ Résultat 3 : ces valeurs GridSearch **AUC = 0.6143 et F-mesure = 0.7485** semblent être les valeurs maximales inter-classifieurs avec notre dataset,
→ Résultat 4 : la valeur **AUC = 0.6143 associée** avec la valeur **Kappa = 0.0560** sont « moyennes » pour un RF sur un problème bi-classes,

(+) la valeur optimale des paramètres recherchés est : mtry = 2 variables et ntree ~ 350 arbres.

Pour tous ces modèles :
. en termes d'importance de variable explicative, les 3 modèles sélectionnent **Cotisation Annuelle et Age Client**.
. ce résultat est aligné avec le classement d'importance du Random Forest de la période de calcul 1.

*** Analyse ***

2 enseignements à retenir :

. le modèle Random Forest est consistant à la fois en performance de classification et en importance de variables entre les versions de validation croisée simple (apprentissage / test – période 1) et optimisée (GridSearch et k-cv=10 – période 2).

. les valeurs sous-performantes des mesures AUC, Kappa et même F-mesure sont à rapprocher du problème de manque de variable discriminante présente dans notre modèle et identifié avec l'expert Benjamin Lamarre en période 1 (i.e. données tarifaires de la concurrence).

*** les éléments de méthodes d'optimisation de la précision du Random Forest ***

Méthode :
Après avoir sélectionné l'algorithme d'apprentissage « Random Forest » adapté à notre problème de transformation MRSQ (période 1), il s'agit de mettre en place une procédure de réglage des paramètres internes du classifieur RF (mtry et ntree), sur un dataset optimisé, afin d'obtenir les meilleures valeurs sur les 3 critères de performances définis comme il suit : *[Critère_i]*

. C1 : le taux de biens classés (« Accuracy »)
- le taux de biens classés est calculé à partir de la matrice de confusion, il correspond au nombre de lignes classées correctement,
- critère facile à lire mais souvent non exploitable au regard de la complexité du problème de classification.

. C2 : AUC (« area under Roc Curve »)
- estimer la qualité de la prévision : la prévision de RF est-elle supérieure à la performance du hasard ?
- critère à mobiliser dans les problèmes de classification binaire supervisée et en présence de classes déséquilibrées pour la variable d'intérêt.

. C3 : F1 ("F-mesure")
- capacité du modèle RF parmi les individus bien classés à détecter les Vrai Positifs (pour notre cas, les « NON », la classe majoritaire),
- la valeur empirique de cette mesure est une « bonne évaluation de la qualité de prédiction du classifieur mais ne peut s'utiliser seule.

On regarde aussi le « kappa de Cohen »,
. Kappa :
- c'est le taux de biens classés (accuracy) qui prend en compte la problématique de classes déséquilibrées,
- on dit aussi que c'est un critère qui mesure la capacité d'un modèle à prédire « mieux qu'au hasard ».

Au regard de la spécification de notre problème de classification binaire supervisée avec classes déséquilibrées, on cherche à maximiser notre critère d'AUC complété du critère F-mesure, puis en association on contrôlera la cohérence de l'Accuracy et du Kappa.

➔ Pour la mise en eouvre, les 3 modèles suivants sont utilisés et le travail de réglage porte sur les 2 principaux paramètres qui impactent la précision du classifieur RF :

Mtry : nombre de variables échantillonnées au hasard et candidates à chaque division,
Ntree : le nombre d'arbre à générer.

Nota : avec le package R-Caret, seul le paramètre mtry est « tunable ».

Modèle 1 : RF « random search »
C'est un Random Forest avec une recherche aléatoire dans un intervalle borné [1,5] pour le paramètre mtry.
[spécification R-Caret tunelength = 5] , sinon par défaut R-Caret fixe les valeurs telles que mtry = sqrt (ncol (x)) et ntree = 500.

Dans notre cas, en mode boîte noire, RF dans son processus d'optimisation de mtry fixe lui-même la valeur ntree nécessaire à son processus.
Dans le jargon, ce modèle est appelé (« baseline »).

Modèle 2 : RF « grid search - accuracy », ou grid search avec le critère de performance = accuracy
Un autre mode de recherche consiste à définir une grille de valeurs de paramètres à tester.
Chaque axe de la grille est un paramètre d'algorithme. Les points dans la grille sont des combinaisons spécifiques de paramètres.
Parce que nous ne réglons un seul paramètre, la recherche de grille est une recherche linéaire à travers un vecteur de valeurs candidates.
[tunegrid<-expand.grid(.mtry=c(1 :3)] + [metric = "Accuracy"]

Un autre point important, notre critère de performance est « l'accuracy ». Cette « métrique » constitue la métrique « en génréral » pour les problèmes de classification. Elle est utilisée en multi-classes, sans prise en compte de déséquilibre de classes.

Modèle 3 : RF « grid search - roc », ou grid search avec le critère de performance = roc
Principe identique au modèle 2,
[tunegrid<-expand.grid(.mtry=c(1 :3)] + [summaryFunction = twoClassSummary] + [metric = "ROC"]

La mesure "ROC" est adaptée à notre problème de la transformation, c'est à dire un problème de classification binaire supervisée avec déséquilibre des classes de la variable d'intérêts.

A noter :
(i) pour éviter les phénomènes de sur-apprentissage du classifieur, notre procédure de validation croisée se fera, avec un nombre de plis = 10 et avec le processus de validation croisée répété = 10 fois.

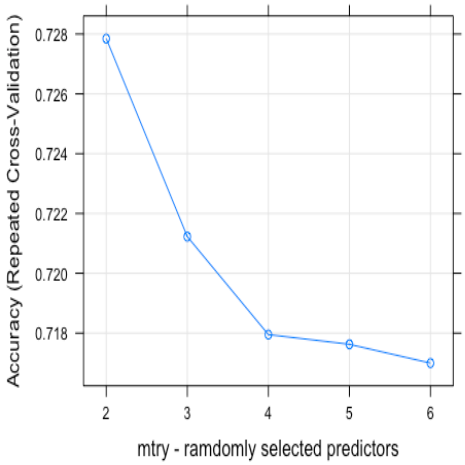
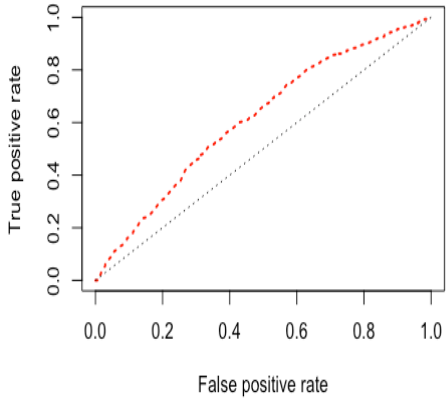
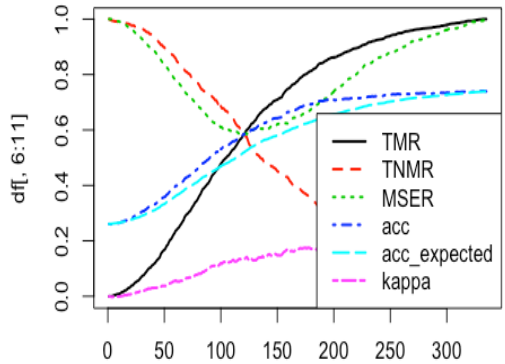
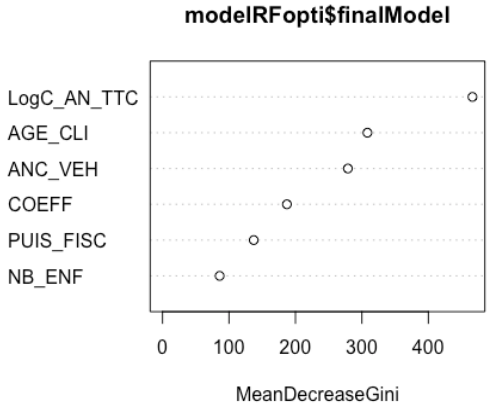
(ii) en conséquence, les valeurs obtenues sur les critères de performance sont des « valeurs moyennes », donc elles seront plus basses que celles obtenues lors de la période 1 de calcul. En revanche, leurs robustesses (i.e. fiabilité) sont plus élevées.

(iii) pour obtenir des temps de calcul optimisés, les 3 modèles ont été exécutés en procédure de calcul parallèle.

*** les éléments chiffrés / visuels à retenir ***

LA PERFORMANCE DES 4 CADRANS

<u>Modèle 1</u> : RF « random search »

Accuracy = fonction (Mtry)	Courbe de ROC
	
Courbes des critères d'erreurs	**Importance des variables explicatives**
	

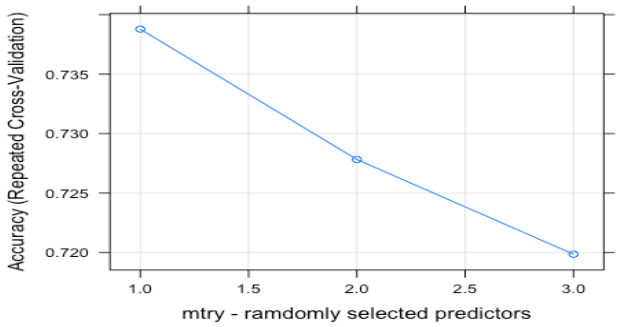
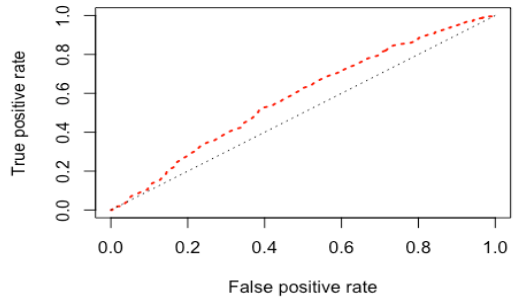
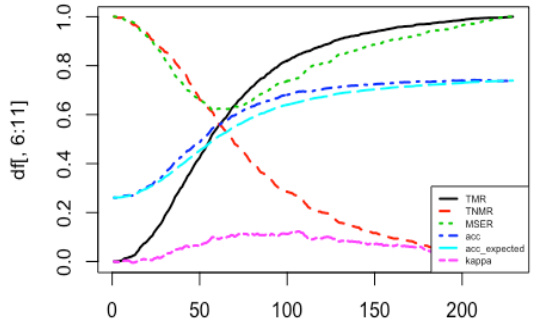
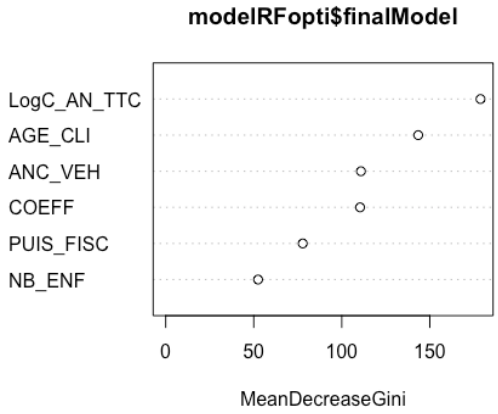
<u>Signalétique</u>:

. ROC curve points of equal error rate (EER), . maximum accuracy, . maximum kappa, . minimum squared error rate (MSER = square of minimum euclidean distance between ROC curve and top left corner for TMR = TNMR = 1))	. The true match rate (TMR) indicates the rate of how much test data set samples were correctly identified as being of the positive class (in our case: les NON). . The false match rate (FMR, counterpart to true non match rate TNMR) indicates the rate of how much test data set samples were incorrectly identified as being of the positive class.

[source : https://geekoverdose.wordpress.com/2014/07/25/svm-classification-example-with-performance-measures-using-r-caret/]

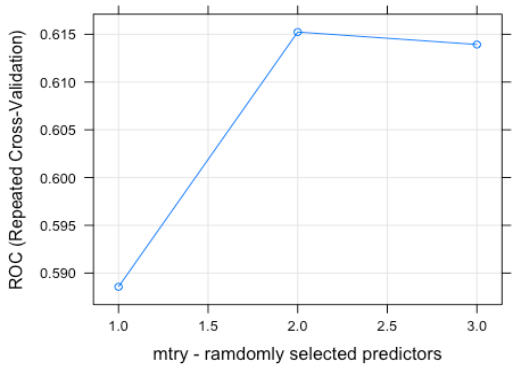
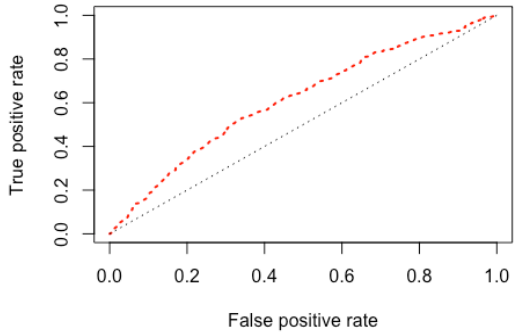
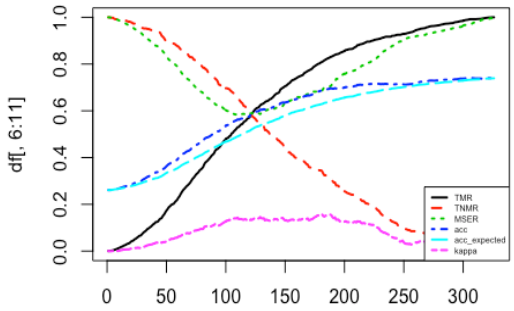
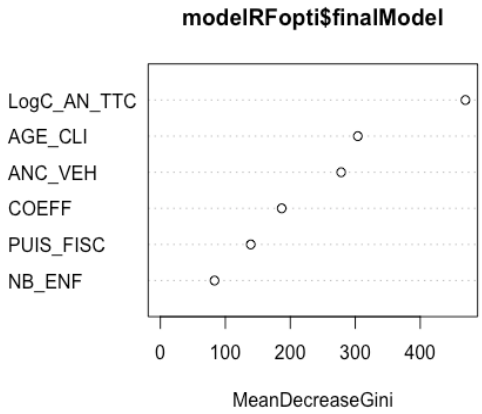
<u>Analyse</u> :
. la bonne performance des courbes de ROC et Kappa,
. la proximité des 2 courbes d'accuracy.

<u>Modèle 2</u> : RF « grid search - accuracy »

Accuracy = fonction (Mtry)	Courbe de ROC
	
Courbes des critères d'erreurs	**Importance des variables explicatives**
	

<u>Analyse</u> :
. la courbe ROC reste « collée » à la bissectrice de la prévision du hasard,
. la sous-performance de la courbe Kappa.

<u>Modèle 3</u> : RF « grid search - roc »

ROC = fonction (Mtry)	Courbe de ROC
	
Courbes des critères d'erreurs	**Importance des variables explicatives**
	

<u>Analyse</u> :
. la mesure ROC = fonction (Mtry) varie en sens inverse, Accuracy = fonction (Mtry),
. l'écart maximal à la bissectrice pour la courbe de ROC (par rapport aux autres courbes de ROC),
. le max de Kappa au 181e arbres.

8.3. Période 2 : résultats du modèle SVM avec grid search

8.3.1. Performance du modèle SVM

*** RESULTATS NUMERIQUES ***

PERIODE 2

SYNTHESE PERFORMANCE DES CLASSIFIEURS SVM

hypo : relation linéaire Yi ~ Xi	5/6 non
Nb total d'observation	6150
Echantillonnage	1%

Etiquette Yi	trsf_dev					
Poids des Yi[i]	[N;O] = [73.31 ; 26.69]					
Nb Xi	6					
Détail Xi	LogC_AN_TTC + PUIS_FISC + COEF + AGE_CLI + NB_ENF + ANC_VEH					
standardisation Xi	centrage-réduction de Xi ~ [m ~ 1; var ~ 0]					
normalisation Xi	C_AN_TTC -> LogC_AN_TTC					
calcul parallèle	1-cpu	1-cpu	1-cpu	1-cpu	3-cpu	3-cpu
Langage Prog	R-e1071	R-e1071	R-e1071	R-e1071	R-Caret/kernlab	R-Caret/kernlab
itération	initialisation	différents noyaux	GridSearch - CV (3e itération a)	GridSearch - CV (3e itération b)	GridSearch - CV - parallel (4e itération a)	GridSearch - CV - parallel (4e itération b)

METRICS	RADIAL (RBF)	RADIAL (RBF)	RADIAL (RBF)	RADIAL (RBF)	RADIAL (RBF)	RADIAL (RBF)
stratégie SVM	1 vs 1	1 vs 1	1 vs 1	1 vs 1	1 vs 1	1 vs 1
apprentissage	0,7	0,7	0,7	0,7	0,7	0,7
test	0,3	0,3	0,3	0,3	0,3	0,3
Correction Poids des Yi[i]	weights	weights	weights	weights	weights	weights
k (cv- k folds)			oui	oui	10	10
répétition					5	5
méthode opti paramètre			grid	grid	Grid + parallel	Grid + parallel
critère de performance	accuracy	ROC/AUC	accuracy	accuracy	accuracy	accuracy
type : classif / reg.	C-classification	C-classification	C-classification	C-classification	C-classification	C-classification
Probabilité	oui	oui	oui	oui	oui	oui
Gamma	1 000 000	**0.17 (1/p)**	**0.53**	**0.2**	**0.5**	**0.15**
Fct Coût - pénalisation [C]	1000	1000	**0.32**	**1000**	**0.55**	**1000**
Degré polynôme						
Temps Calcul (seconde)	27.48 secs	52.61 secs	34.04 mins	5.79 mins	14.91 mins	21.91 mins
Tx de biens classés (Accuracy) %	74.03		**52.97**	**63.24**	**73.11**	**74.52**
Tx d'Erreur %	25.96		47.02	36.75	26.89	25.48
AUC	**0.4920**	**0.6319**	**0.50**	**0.56**	**0.5675**	**0.5685**
Nombre de vecteurs supports			**410**	**248**		
Kappa					0.0089	0.00
F1 (F-mesure)					**0.7404**	**0.7452**
Contrôle Graphique	oui	oui	oui	oui	oui	oui

valeur maximum sur le critère

modèle retenu

Tableau 4 - synthèse de la performance du classifieur SVM Forest période de calcul 2

*** Synthèse des résultats ***

But : construire un classifieur SVM optimisé comme modèle concurrent pour un objectif de précision de classification, afin de le comparer au modèle de référence RF optimisé sur notre jeu de données.

Résultats : sur ce jeu de données, le modèle retenu est le **SVM Gaussien C = 1000 et $\gamma = 0.15$**
(RBF = Radial Basis Function = Gaussien)

→ Résultat 1 : le classifieur retenu est celui dans la famille SVM qui peut classer nos données non séparables linéairement,
→ Résultat 2 : ce classifieur a été obtenu après 6 itérations, avec **1 Grid Search calcul parallèle** sur le total des 6150 lignes du dataset SSP,
→ Résultat 3 : le paramétrage du noyau correspond à une forte pénalisation du SVM (C = 1000) et au minimum du Gamma ($\boldsymbol{\gamma = 0.15}$)
→ Résultat 4 : ce classifieur maximise les résultats sur les 3 critères : **AUC, F-mesure et taux de biens classés (56.86%, 74.52% et 74.52%).**

(+) la valeur optimale des paramètres recherchés est : $\boldsymbol{\gamma = 0.15}$ (en dessous du réglage préconisé par les auteurs de LibSVM = 1/p)

Pour tous ces modèles :

. Ces résultats sont nominaux (P.35) par rapport formalisme du noyau **RBF : (↗) C et (↘) $\boldsymbol{\gamma}$ ➔ et (↘) $\mathbf{Biais = Accuracy}$ (↗)** et le modèle sélectionné semble **bien « généraliser »** avec une valeur empirique de la F-mesure plus élevée (c.à.d., le classifieur détecte bien et a une capacité à reconnaître les classes de la variable d'intérêt – **A VERIFIER dans le détail, avec l'audit des différents critères de performance**).

*** Analyse ***

2 enseignements à retenir :

. Les valeurs empiriques (AUC, F-mesure, Accuracy) obtenues avec ce modèle SVM Gaussien C = 1000 et γ = 0.15 semblent être les valeurs maximales de la famille des classifieurs SVM pour notre Dataset.

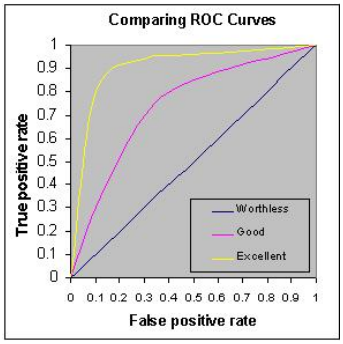
. Pour autant une valeur AUC = 0.5685 pour un problème de classification binaire supervisée avec classes déséquilibrées est considérée comme **« moyenne »** au regard de la littérature abondante sur les mesures de performance en machine learning.

TABLE VII. EXPERIMENTAL RESULTS OF GNOME DATASET

Approaches	Classifiers					
	NN		*SVM*		*FSVM*	
	G-mean	*AUC*	*G-mean*	*AUC*	*G-mean*	*AUC*
Original	0.0000	0.5000	0.0000	0.5000	0.0000	0.5000
CMT1	0.0000	0.5000	0.0000	0.5000	0.0000	0.5000
CMT2	0.0000	0.5000	0.0000	0.5000	0.0000	0.5000
SMOTE	0.6569	0.7598	0.9470	0.9482	0.9588	0.9596
CMTSMT1	0.5633	0.7075	0.9368	0.9381	**0.9589**	**0.9598**
CMTSMT2	0.5950	0.7128	0.9470	0.9482	**0.9589**	**0.9598**
SMTCMT1	0.6859	0.7462	0.9577	0.9586	0.9576	0.9585
SMTCMT2	0.7866	0.7921	0.9579	0.9588	0.9577	0.9586

Source : Imbalanced Data Classification using Complementary Fuzzy Support Vector Machine Techniques and SMOTE[19] - 2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC) Banff Center, Banff, Canada, **October 5-8, 2017**

A noter : le modèle hybride qui combine le modèle Fuzzy SVM avec le modèle de correction de classes avec données déséquilibrées.



Source : http://gim.unmc.edu/dxtests/roc3.htm

[19] Learning from Imbalanced Data : http://ieeexplore.ieee.org/document/5128907/?reload=true

*** les éléments de méthodes d'optimisation de la précision du SVM ***

Méthode :

. Pour obtenir ses résultats, j'ai suivi « le Pipe de réglage » préconisé par l'état de l'art de la mise en oeuvre du classifieur SVM,

(i) Utiliser la connaissance du domaine / du problème pour repérer une famille de noyaux utile et démarrer avec,
(ii) Tester plusieurs noyaux et sélectionner du noyau le plus adapté au problème,
(iii) Faire des essais / erreurs et tester des procédures de calcul parallèle,
(iv) Sur la base de plusieurs critères de performance de précision de classification, sélectionner le meilleur modèle (et ses réglages associés).

. Je me suis inspiré des préconisations émises par les auteurs (Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin) de la « library » LIBSVM que j'ai résumé dans la fiche de lecture suivante :

A Practical Guide to Support Vector Classification		
https://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf		
page	objet	detail
1	introduction	SVM utile pour la classification de données
1	introduction	dataset - découpé training set (avec label) et test set (sans label)
2	introduction	C > 0 : parametre du terme d'erreur
2	introduction	K(xi, xj) : fonction noyau > 4 noyaux base = lineaire, polynomial, gaussien RBF, sigmoid
2	introduction	4 noyaux base = lineaire, polynomial, gaussien RBF, logistique sigmoid
3	best pratice	1.2: Proposed Procedure - pas de transfo des datas, utiliser des noyaux non lineaires
4	best pratice	1.2: dataset training - Cross Validation (CV) pour trouver parametres C et Gamma puis Test
3	Data Preprocessing	2.1: variable catégorielle (quali) - transformer en var Indicatrice (0,1)
4	Scaling	2.2: centrage reduction - reduction avant d'appliquer 1 noyau SVM
4	Scaling	2.2: centrage reduction - identique pours datasets apprentissage et test
4	sélection model - rbf	3.1 : commencer par le noyau gaussien (RBF Kernel) avec 2 parametres : (C,Gamma)
5	sélection model - rbf	3.1 : qd bcp de variables, préférer un noyau lineaire (linear kernel)
5	sm - CV and Grid-search	3.2: faire 1 CV pour éviter l'over-fitting, utiliser 1 première grille (coarse grid first) + grille affinage
8	discussion	4. : avec bcp de features, faire de la selection de variable avant d'appliquer SVM
9	discussion	4. : LIBSVM - procédure auto de centrage et de sélection de modèle (code par défaut)
12	scaling training - test ?	B : Using the same scaling factors for training and testing sets, we obtain much better accuracy.
		accuracy performance se mesure uniquement sur dataset TEST (et non TRAINING)
13	linear vs RBF kernel	C1: Nb of instances ≪ nb of features => **CV accuracy linear kernel = RBF Kernel**
14	linear vs RBF kernel	C2: instances & features - larges = LIBLINEAR-svm (avec RegLog), plus rapide que LIBSVM
14	linear vs RBF kernel	C3: Nb of instances ≫ nb of features => linear kernel
		ou LIBLINEAR with the option -s 2
		Nb features = petit --> LIBLINEAR with the option -s1
		Nb instances très supérieur ≫ nb features = LIBLINEAR with the option -s 1 // et option -s 2
		https://en.wikipedia.org/wiki/LIBSVM
		http://www.csie.ntu.edu.tw/~cjlin/liblinear/
		https://www.csie.ntu.edu.tw/~cjlin/libsvm/
		définition : instance = ligne ; feature = colonne
		définition : coarse grid = "grille grossière = grille large"

Tableau 5 - fiche de lecture - A Practical Guide to SVM Classification

*** les éléments chiffrés / visuels à retenir ***

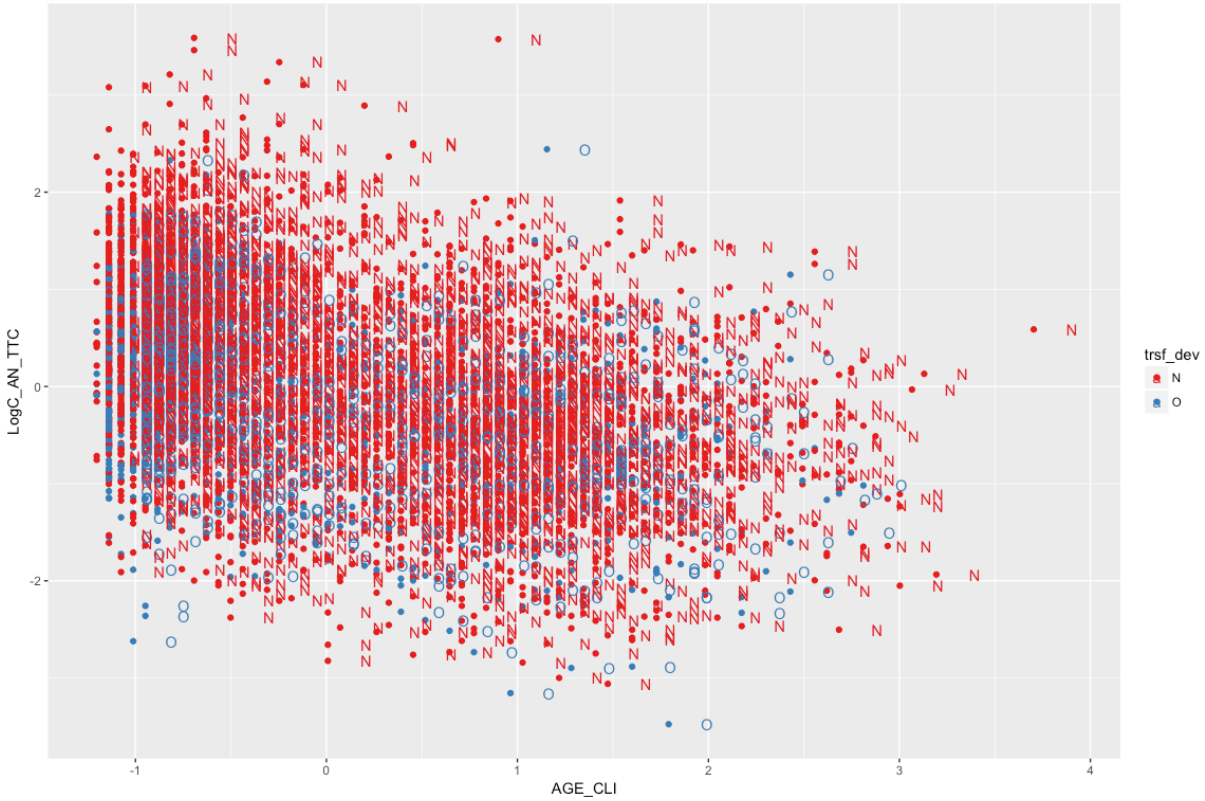
① SVM et données d'apprentissage : le jeu de données d'apprentissage et le classifieur SVM ?
(Effectif 4305 obs. * 7 variables, dont 3156 Non et 1149 Oui)

2 questions :
. Peut-on faire apprendre un SVM sur ces données d'apprentissage ?
. Le problème de classification est-il exploitable avec le SVM ?

ELEMENTS	ANALYSE
Table fréquence : trsf_dev et LogC_AN_TTC (sur l'apprentissage 0.7) 	. Figure difficilement exploitable, . Légère différence entre les « non » et les « oui ». . On peut dire qu'il se passe quelque chose, une relation entre la variable d'intérêt (trsf_dev) et la variable explicative la plus importante LogC_AN_TTC

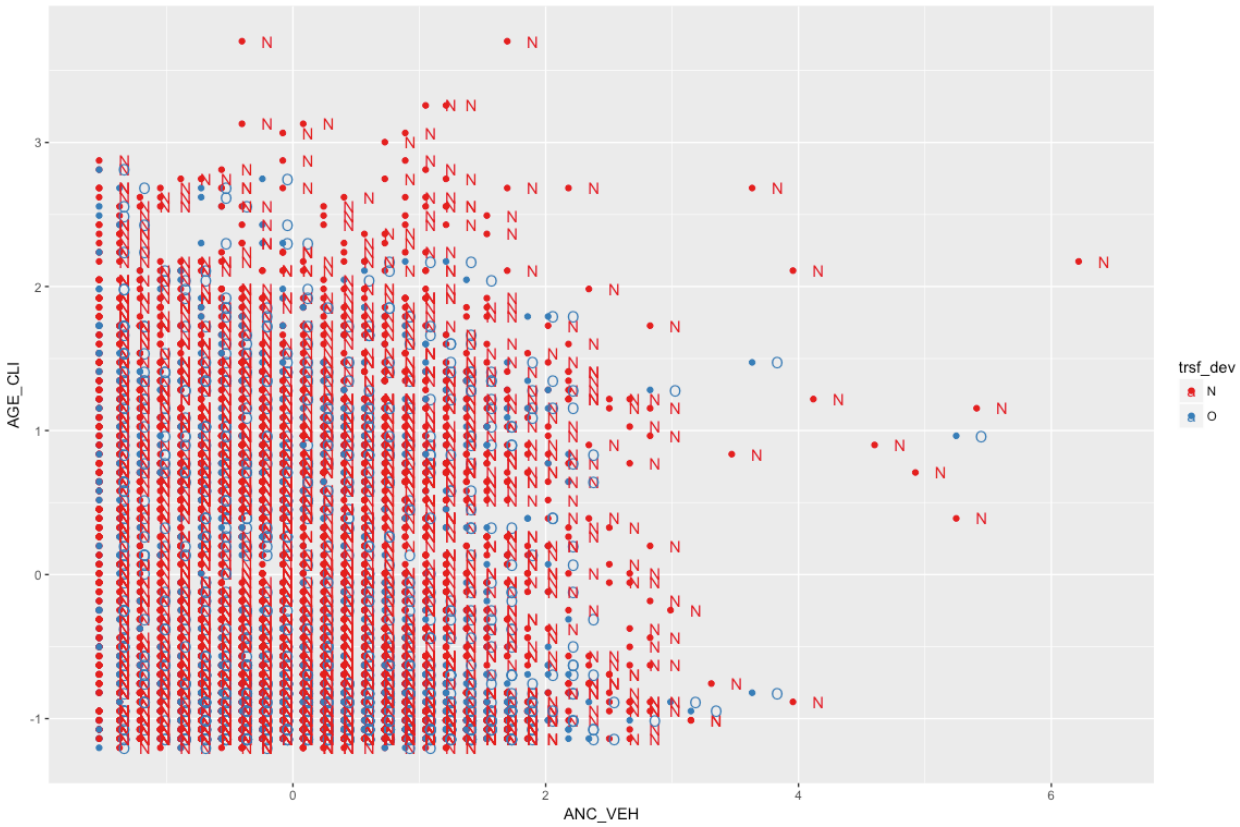
. A partir des variables explicatives importantes sélectionnées par le RF Gridsearch ROC, produisons des « datavizualisations tris croisés » exprimés en fonction des modalités de la variable d'intérêt (N, O)

Cotisation ~ Age Client :



Analyse :
. avec 4305 points, la lecture n'est pas facilitée,
. le graphique est saturé dans une masse de Non rouges,
. la modalité « Oui » est noyée « sporadiquement » dans la masse de Non rouges.

Age Client ~ Ancienneté du Véhicule :



Analyse :
. avec 4305 points, la lecture n'est pas facilitée,
. le graphique est saturé dans une masse de Non rouges,
. la modalité « Oui » est noyée dans la masse de Non rouges.

Conclusion :
. A la lecture des résultats de ces 2 graphiques, les classes étiquetées [N] et [O] sont mélangées, le problème de classification binaire supervisée de la transformation devis n'est pas linéairement séparable. Le classifieur SVM avec noyau linéaire ne permettra pas de séparer nos données d'apprentissage.

. Cela veut dire avant de lancer un SVM optimisé en GridSearch, nous devons travailler les questions suivantes :

⚠ Bonnes Pratiques DATALAB : « la check list SVM »

```
#==========================================================================================
#-- TODO : « SVM Grid Search » - les questions à se poser
#- q1: problème de classification : bi-classes / multi-classes ?
#- q2: séparateur linéaire ?
#- q3: séparateur non linéaire ?
#- q4 : le problème est-il séparable ou imparfaitement séparable ?
#--------
#- q5 : comment faire du grid search (environnement de développement, package, réglage des paramètres ?)
#- q6 : comment choisir le meilleur coût de régularisation (cost) ? vecteur de valeurs coût ? pourquoi ?
#- q7 : comment choisir le meilleur noyaux, choix du noyau ?
#- q8 : comment choisir la valeur du gamma du noyau ?
#- q9 : noyau ? polynôme : comment choisir le degré du polynôme, si noyau polynômial ?
#==========================================================================================
```

② 1ère itération calcul SVM : initialisation du modèle SVM avec noyau radial ?

Paramétrage : A partir de la connaissance et des résultats obtenus lors des étapes précédentes, nous pouvons en déduire,
(1) nous avons un problème classification binaire non linéairement séparable => on retire kernel linear,
(2) lors du modèle régression logistique : le modèle n'est pas bien spécifié avec ce dataset, on retire le noyau sigmoid,
(3) on retient le kernel gaussien vs kernel polynomial ==> pour démarrer le kernel gaussien (radial-RBF), "spectre d'utilisation large pour de nombreux problèmes",
(4) avec ce kernel RBF : les auteurs[20] proposent les valeurs de paramètres : cost = 1000, gamma = 1/p = 0.1666
(5) modélisation sans optimisation du type GridSearch, sans calcul parallèle.

PERIODE 2 *initialisation modèle SVM (1ère itération)*

hypo : relation linéaire Yi ~ Xi	5/6 non
Nb total d'observation	6150
Echantillonnage	1%

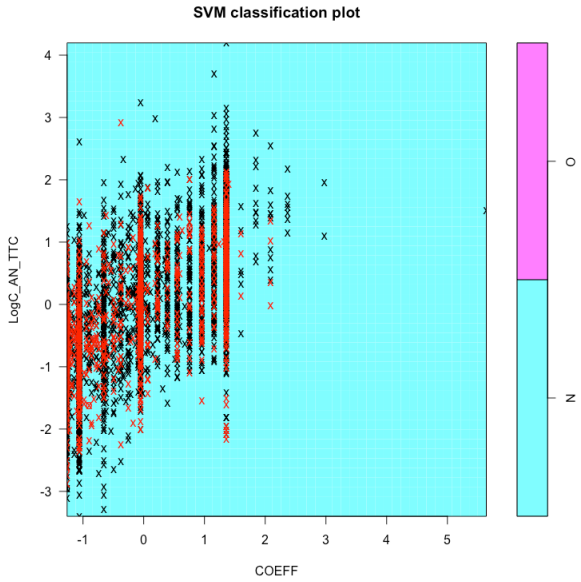
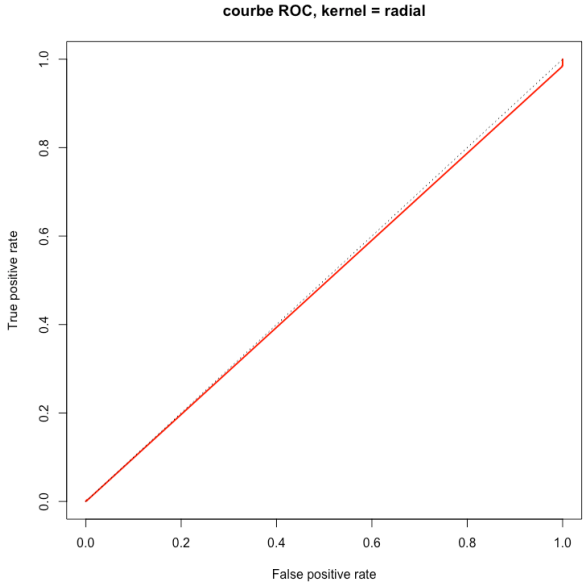
Etiquette Yi	trsf_dev					
Poids des Yi[i]	[N;O] = [73.31 ; 26.69]					
Nb Xi	6					
Detail Xi	LogC_AN_TTC + PUIS_FISC + COEF + AGE_CLI + NB_ENF + ANC_VEH					
standardisation Xi	centrage-réduction de Xi ~ [m ~ 1; var ~ 0]					
normalisation Xi	C_AN_TTC -> LogC_AN_TTC					
calcul parallèle	1-cpu	1-cpu	1-cpu	1-cpu	1-cpu	1-cpu
Langage Prog	R-e1071	R-e1071	R-e1071	R-e1071	R-e1071	R-e1071

METRICS	RADIAL (RBF)	RADIAL (RBF)	RADIAL (RBF)	RADIAL (RBF)	RADIAL (RBF)	**RADIAL (RBF)**
stratégie SVM	1 vs 1	1 vs 1	1 vs 1	1 vs 1	1 vs 1	1 vs 1
apprentissage	0,7	0,7	0,7	0,7	0,7	0,7
test	0,3	0,3	0,3	0,3	0,3	0,3
Correction Poids des Yi[i]	weights	weights	weights	weights	weights	weights
k (cv- k folds)						
répétition						
méthode opti paramètre						
critère de performance	accuracy	accuracy	accuracy	accuracy	accuracy	accuracy
type : classif / reg.	C-classification	C-classification	C-classification	C-classification	C-classification	C-classification
Probabilité	oui	oui	oui	oui	oui	oui
Gamma	0,166 (1/p)	1	10	100	1000	1 000 000
Fct Coût - pénalisation [C]	1000	1000	1000	1000	1000	1000
Degré polynôme						
Temps Calcul (seconde)	27.86 secs	30.56 secs	29.18 secs	30.58 secs	26.56 secs	27.48 secs
Tx de biens classés (Accuracy) %	56.04	60.27	69.43	73.00	73.55	**74.03**
Tx d'Erreur %	43.95	39.72	30.56	26.99	26.44	25.96
AUC	0.3652	0.4019	0.45	0.4610	0.49	**0.4920**
Kappa						
F1 (F-mesure)						
Contrôle Graphique	oui	oui	oui	oui	oui	oui

Analyse : les résultats sont alignés avec le formalisme mathématique, l'accroissement du paramètre ***γ* (*Gamma*)** → **sur-apprentissage** du modèle qui se traduit en valeur empirique par une élévation du taux de biens classés.
(→ indicateur d'une frontière décision quadratique… indice que notre modèle de transformation n'est pas linéaire)

Gaussien (RBF = Radial Basis Funct.) $k(x,x') = \exp\{-\|X - X'\|^2 / (2\sigma^2)\}$ $\gamma = \text{Gamma} = \text{ddl} = 1/(2\sigma^2)\}$	σ^2 = variance, commun à tous les noyaux (apriori fixé utilisateur) -- σ = largeur de bande			Noyau passe-partout mais peut aussi ne pas convenir au problème traité (rôle de l'Ontology)	Capacité à bien mesurer la similarité entre les couples de points
	Par dédaut : $\gamma = \frac{1}{p}$ (p = Nb xi)				
	[Petit] : $\gamma \sim 0$ [σ^2 (↗)]	(↘)	(↗)	Frontière linéaire Sous-apprentissage	Ca (↘)
	[Grand] : $\gamma \sim \infty$ [σ^2 (↘)]	(↗)	(↘)	Frontière quadratique Sur-apprentissage	Ca (↗)

[20] Chih-Chung Chang and Chih-Jen Lin (auteurs de la « library » libsvm) - https://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf

ELEMENTS	ANALYSE
Représentation graphique : modèle retenu SVM radial (c = 1000 ; $\gamma = 1\ 000\ 000$) Dataset = apprentissage LogC_AN_TTC ~ COEFF = fonction (NON, OUI)  	. Analyse : malgré le paramétrage optimisé « à la main » du SVM radial, le résultat est sans surprise (une prévision médiocre), le classifieur ne sait détecter que du « NON » la classe dominante.
 	Analyse : la qualité de la prédiction de ce classifieur est alignée sur la prédiction du hasard.

③ 2e itération : différents noyaux comparés > trouver l'AUC max ?

Paramétrage :
(1) on teste les 4 noyaux « usuels »,
(2) cost = 1000, gamma = 1/p = 0.1666 et degré polynôme = 2,
(3) on compare les 4 modèles sur le critère AUC, on sélectionne le modèle avec AUC max,
(5) sans optimisation du type GridSearch, sans calcul parallèle.

PERIODE 2 *2.3 différents noyaux - argmax = AUC (2e itération)*

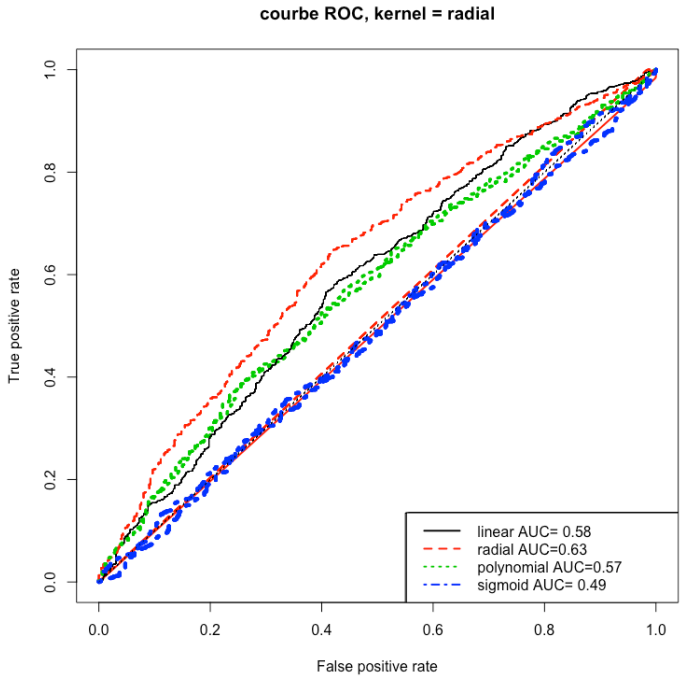
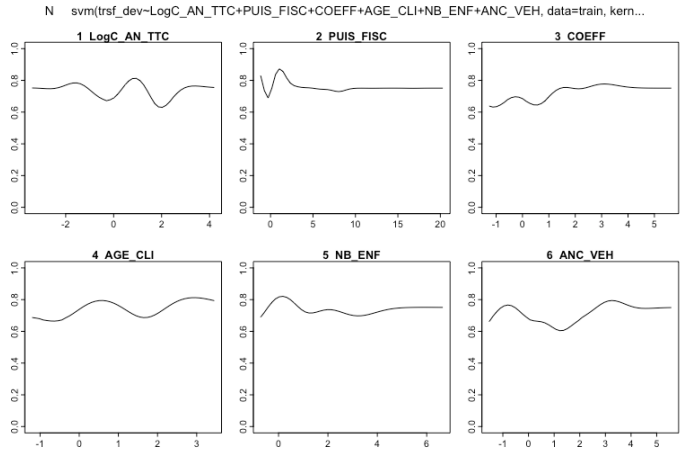
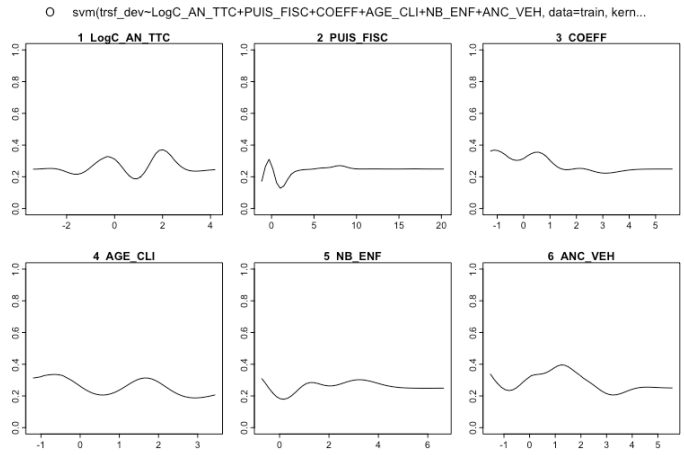
hypo : relation linéaire Yi ~ Xi	5/6 non
Nb total d'observation	6150
Echantillonnage	1%

Etiquette Yi	trsf_dev			
Poids des Yi[i]	[N;O] = [73.31 ; 26.69]			
Nb Xi	6			
Detail Xi	LogC_AN_TTC + PUIS_FISC + COEF + AGE_CLI + NB_ENF + ANC_VEH			
standardisation Xi	centrage-réduction de Xi ~ [m ~ 1; var ~ 0]			
normalisation Xi	C_AN_TTC -> LogC_AN_TTC			
calcul parallèle	1-cpu	1-cpu	1-cpu	1-cpu
Langage Prog	R-e1071	R-e1071	R-e1071	R-e1071

METRICS	LINEAR	**RADIAL (RBF)**	POLYNOMIAL	SIGMOID
stratégie SVM	1 vs 1	1 vs 1	1 vs 1	1 vs 1
apprentissage	0,7	0,7	0,7	0,7
test	0,3	0,3	0,3	0,3
Correction Poids des Yi[i]	weights	weights	weights	weights
k (cv- k folds)				
répétition				
méthode opti paramètre				
critère de performance		ROC/AUC		
type : classif / reg.	C-classification	C-classification	C-classification	C-classification
Probabilité	oui	oui	oui	oui
Gamma		0.17 (1/p)	0.17	0.17
Fct Coût - pénalisation [C]	1000	1000	1000	1000
Degré polynôme			2	
Temps Calcul (seconde)	52.61 secs	52.61 secs	52.61 secs	52.61 secs
Tx de biens classés (Accuracy) %				
Tx d'Erreur %				
AUC	0.5892	**0.6319**	0.5780	0.4957
Kappa				
F1 (F-mesure)				
Contrôle Graphique	oui	oui	oui	oui

Analyse :
Sur le critère de l'AUC, critère le plus adapté pour sélectionner un classifieur adapté à notre problème, le classifieur radial avec **un AUC = 0.6319** semble être le meilleur classifieur. En revanche, résultat à modérer car les métriques taux de bien classés et AUC varient en sens inverse. Ces résultats peuvent être modifiés par une taille plus importante du dataset.

Au final : ce résultat dans le choix du noyau SVM est à confirmer avec la procédure de grid search tune(..) du package caret.

ELEMENTS	ANALYSE
Graphique : analyse comparée des courbes de ROC des 4 noyaux (linéaire, radial, polynomial et sigmoïde)  	Graphique : le comportement des Xi dans la probabilité de prédire la classe [N ; O] par le classifieur RBF (sélectionné à l'étape précédente)   Analyse : . tous les Xi jouent un rôle, . les probabilités Xi (Non) > probabilités Xi (Oui) **Résultat nominal : Le « Non » classe majoritaire en fréquence dans notre dataset, l'est aussi en probabilité de prédiction pour le modèle SVM RBF.** (en probabilité, l'argument majoritaire = probabilité plus élevée)

④ **3e itération : grid_search "à la main" grid_search "avec libsvm et cross-validation k-folds" trouver les paramètres du best modèle?**

Paramétrage :
(1) on teste les 4 noyaux « usuels »,
(2) **en mode GridSearch**, à la main, vecteur de valeurs et en Cross-Validation (Tuning de l'algorithme package e1071),
(3) sélection du noyau et de ses paramètres optimisés sur le critère de valeur minimale de vecteurs supports,
(5) sans calcul parallèle,
(6) limitation de l'échantillon à n = 614 lignes (par rapport à 6150 lignes) pour réussir à exécuter le calcul en « Ram » ordinateur.

PERIODE 2 *2.4 GridSearch - CV (3e itération a)*

hypo : relation linéaire Yi ~ Xi	5/6 non
Nb total d'observation	6150
Echantillonnage	nombre de ligne maximal (en Ram) = **614**

Etiquette Yi	trsf_dev			
Poids des Yi[i]	[N;O] = [73.31 ; 26.69]			
Nb Xi	6			
Détail Xi	LogC_AN_TTC + PUIS_FISC + COEF + AGE_CLI + NB_ENF + ANC_VEH			
standardisation Xi	centrage-réduction de Xi ~ [m ~ 1; var ~ 0]			
normalisation Xi	C_AN_TTC -> LogC_AN_TTC			
calcul parallèle	1-cpu	1-cpu	1-cpu	1-cpu
Langage Prog	R-e1071	R-e1071	R-e1071	R-e1071

METRICS	LINEAR	**RADIAL (RBF)**	POLYNOMIAL	SIGMOID
stratégie SVM	1 vs 1	1 vs 1	1 vs 1	1 vs 1
apprentissage	0,7	0,7	0,7	0,7
test	0,3	0,3	0,3	0,3
Correction Poids des Yi[i]	weights	weights	weights	weights
k (cv- k folds)	oui	oui	oui	oui
répétition				
méthode opti paramètre		grid		
critère de performance		accuracy		
type : classif / reg.	C-classification	C-classification	C-classification	C-classification
Probabilité		oui		
Gamma		**0.53**		
Fct Coût - pénalisation [C]		**0.32**		
Degré polynôme				
Temps Calcul (seconde)		34.04 mins		
Tx de biens classés (Accuracy) %		**52.97**		
Tx d'Erreur %		47.02		
AUC		**0.50**		
Nombre de vecteurs supports		**410**		
Kappa				
F1 (F-mesure)				
Contrôle Graphique	oui	oui	oui	oui

Gamma	c(0.1,0.5,1.0,2.0,10)
	c(0.001 , 0.01 , 0.1, 1 ,5 ,10 ,100,1000),
	c(0.001, 0.1, 0.5, 1 ,10 ,100),
	seq(0.3,0.6,by=0.01)

Fct Coût - pénalisation [C]	2^(-10:0)
	seq(0.01,1,by=0.01)
	c(0.05, 0.25, 0.5, 0.75, 1)
	seq(0.3,0.6,by=0.01)

Analyse : (4 séquences de paramètres ont été testées, la dernière séquence a produit les valeurs du tableau de synthèse)
. sur n = 614, le noyau sélectionné est le **Radial** avec un nombre élevé de vecteurs supports (410 vs / 614 lignes),
. les Taux de biens classés et AUC sont dégradées

Conclusion : Il faut tester le noyau Radial avec valeurs de paramètres $C = 1000$ (2e itération) et avec recherche de γ.

(Résultats de : **Itérer le Radial avec valeurs $C = 1000$ (2e itération) et recherche de γ.**)

PERIODE 2

2.4 GridSearch - CV (3e itération b)

hypo : relation linéaire Yi ~ Xi	5/6 non
Nb total d'observation	6150
Echantillonnage	nombre de ligne maximal (en Ram) = **614**

Etiquette Yi	trsf_dev			
Poids des Yi[i]	[N;O] = [73.31 ; 26.69]			
Nb Xi	6			
Détail Xi	LogC_AN_TTC + PUIS_FISC + COEF + AGE_CLI + NB_ENF + ANC_VEH			
standardisation Xi	centrage-réduction de Xi ~ [m ~ 1; var ~ 0]			
normalisation Xi	C_AN_TTC -> LogC_AN_TTC			
calcul parallèle	1-cpu	1-cpu	1-cpu	1-cpu
Langage Prog	R-e1071	R-e1071	R-e1071	R-e1071

METRICS	LINEAR	**RADIAL (RBF)**	POLYNOMIAL	SIGMOID
stratégie SVM	1 vs 1	1 vs 1	1 vs 1	1 vs 1
apprentissage	0,7	0,7	0,7	0,7
test	0,3	0,3	0,3	0,3
Correction Poids des Yi[i]	weights	weights	weights	weights
k (cv- k folds)	oui	oui	oui	oui
répétition				
méthode opti paramètre		grid		
critère de performance		accuracy		
type : classif / reg.	C-classification	C-classification	C-classification	C-classification
Probabilité		oui		
Gamma		**0.2**		
Fct Coût - pénalisation [C]		**1000**		
Degré polynôme				
Temps Calcul (seconde)		5.79 mins		
Tx de biens classés (Accuracy) %		**63.24**		
Tx d'Erreur %		36.75		
AUC		**0.56**		
Nombre de vecteurs supports		**248**		
Kappa				
F1 (F-mesure)				
Contrôle Graphique	oui	oui	oui	oui

Gamma	c(0.1,0.5,1.0,2.0,10)
	c(0.001 , 0.01 , 0.1, 1 ,5 ,10 ,100,1000),
	c(0.001, 0.1, 0.5, 1 ,10 ,100),
	seq(0.3,0.6,by=0.01)
	seq(0.15,0.20,by=0.01)

Fct Coût - pénalisation [C]	2^(-10:0)
	seq(0.01,1,by=0.01)
	c(0.05, 0.25, 0.5, 0.75, 1)
	seq(0.3,0.6,by=0.01)
	1000

Analyse :

. en se guidant des valeurs $C = 1000$ *et* $\gamma = 0.17$ utilisées en 2e itération pour le Radial (effectif n = 6150 et AUC = 0.6319),

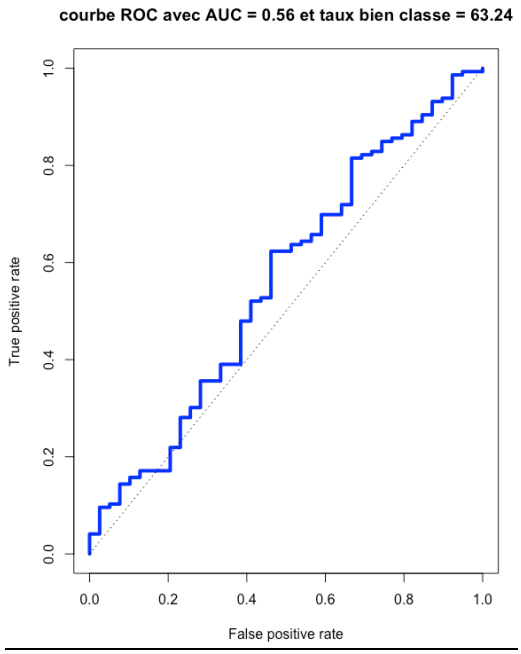
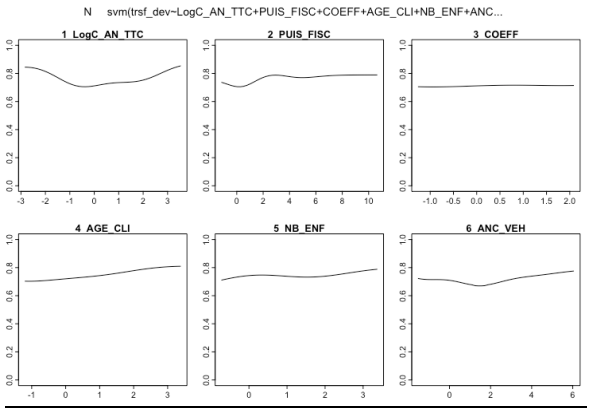
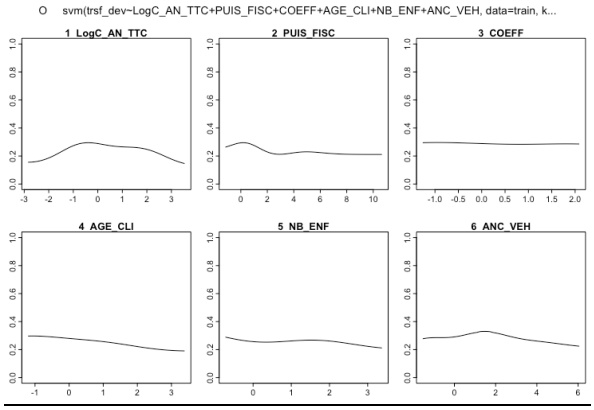
. on sélectionne **le SVM Radial** $C = 1000, \gamma = 0.20$ *et le nombre de vecteurs supports* $= \mathbf{248}$,

. la pénalisation augmentée du SVM a permis d'améliorer la solution SVM par réduction du nombre de vecteurs supports, mais aussi la **performance de classification est accrue** par rapport au modèle précédent car la valeur de l'AUC passe **de 0.50 à 0.56**.

. le taux de biens classés s'étant amélioré passant de **52.97% à 63.24%** ➔ ces résultats sont nominaux (P.35) par rapport formalisme du noyau **RBF : (↗) C et (↘) γ ➔ et (↘) Biais = Accuracy (↗)** mais un modèle sélectionné qui risque « *de mal généraliser* ».

. Attention : ces résultats sont à confirmer au regard de la petite taille d'échantillon imposée par la contrainte de la « Ram ordinateur ».

Conclusion : la nécessité de poursuivre notre optimisation GridSearch du SVM Radial en mobilisant une procédure de calcul parallèle pour travailler l'effectif total de notre échantillon SSP (n = 6150).

ELEMENTS	ELEMENTS
Graphique : Courbe ROC du classifieur **SVM Radial** $C = 1000, \gamma = 0.20$ $et\ \ nombre\ de\ vecteurs\ supports = 248,$ pour une taille d'échantillon n = 614	 Analyse : les résultats de ce classifieur SVM Radial sont à confirmer sur un échantillon notre échantillon SSP de taille n = 6150.
Graphique : le comportement des Xi dans la probabilité de prédire la classe par le classifieur RBF $C = 1000, \gamma = 0.20$ **Les « NON »**  Analyse : On retrouve les conclusions du même graphique itération 2 (i.e. pour les 4 différents noyaux)	Graphique : le comportement des Xi dans la probabilité de prédire la classe par le classifieur RBF $C = 1000, \gamma = 0.20$ **Les « OUI »** 

⑤ **4e itération : grid_search "à la main" grid_search "avec libsvm et cross-validation k-folds" trouver les paramètres du best modèle?**

Paramétrage :
(1) on teste seulement le noyau Radial (RBF),
(2) en mode GridSearch, à la main, vecteur de valeurs et en Cross-Validation (Tuning de l'algorithme package Kernlab),
(3) sélection du noyau et de ses paramètres optimisés sur le critère de l'Accuracy (i.e. Accuracy, Kappa, AccuracySD, KappaSD)
(5) avec une procédure de calcul parallèle (3/4 cpu utilisés),
(6) l'échantillon à n = 6150 lignes.

PERIODE 2 *2.5 GridSearch - CV - parallel (4e itération a)*

hypo : relation linéaire Yi ~ Xi	5/6 non
Nb total d'observation	6150
Echantillonnage	1%

Etiquette Yi	trsf_dev			
Poids des Yi[i]	[N;O] = [73.31 ; 26.69]			
Nb Xi	6			
Détail Xi	LogC_AN_TTC + PUIS_FISC + COEF + AGE_CLI + NB_ENF + ANC_VEH			
standardisation Xi	centrage-réduction de Xi ~ [m ~ 1; var ~ 0]			
normalisation Xi	C_AN_TTC -> LogC_AN_TTC			
calcul parallèle	3-cpu	3-cpu	3-cpu	3-cpu
Langage Prog	R-Caret/kernlab	R-Caret/kernlab	R-Caret/kernlab	R-Caret/kernlab

METRICS	LINEAR	**RADIAL (RBF)**	POLYNOMIAL	SIGMOID
stratégie SVM	1 vs 1	1 vs 1	1 vs 1	1 vs 1
apprentissage	0,7	0,7	0,7	0,7
test	0,3	0,3	0,3	0,3
Correction Poids des Yi[i]	weights	weights	weights	weights
k (cv- k folds)		10		
répétition		5		
méthode opti paramètre		Grid + parallel		
critère de performance		accuracy		
type : classif / reg.	C-classification	C-classification	C-classification	C-classification
Probabilité		oui		
Gamma		**0.5**		
Fct Coût - pénalisation [C]		**0.55**		
Degré polynôme				
Temps Calcul (seconde)		14.91 mins		
Tx de biens classés (Accuracy) %		**0,7311**		
Tx d'Erreur %		0,2689		
AUC		**0.5675**		
Nombre de vecteurs supports				
Kappa		0.0089		
F1 (F-mesure)		**0.7404**		
Contrôle Graphique	oui	oui	oui	oui

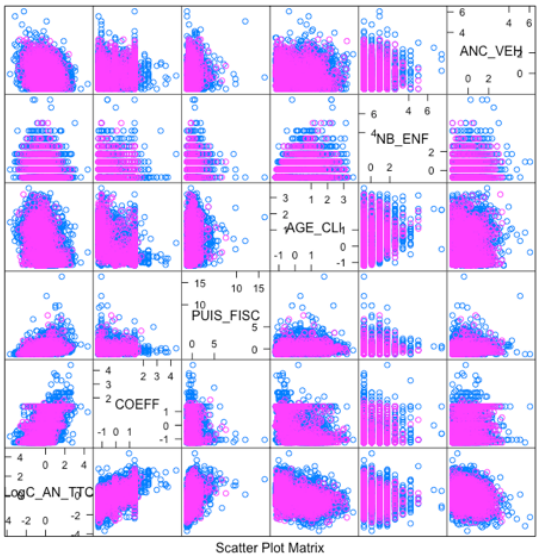
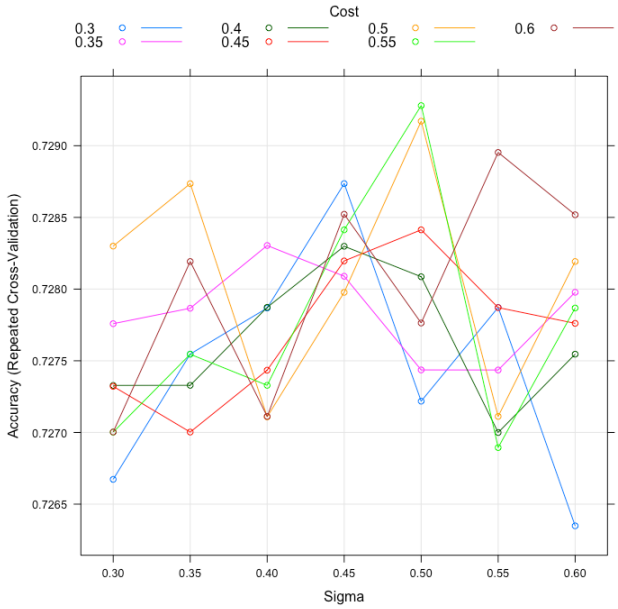
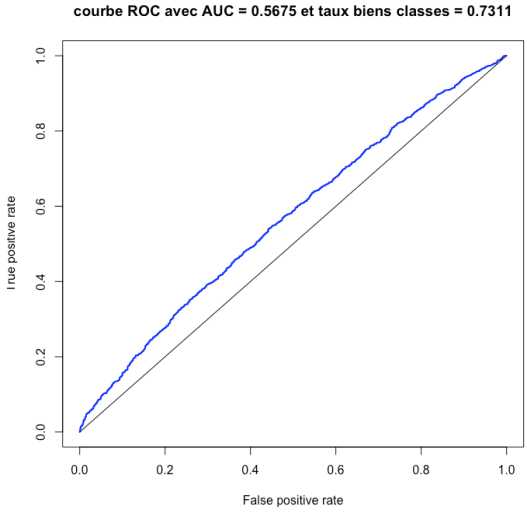
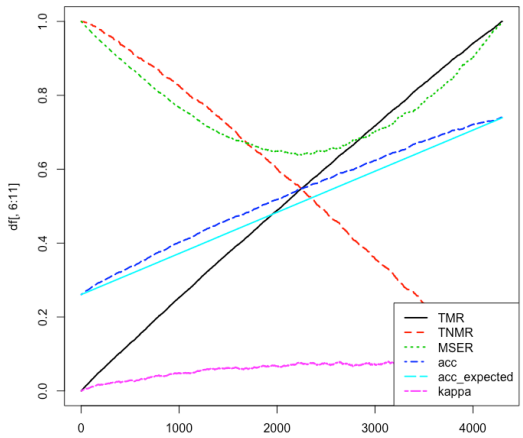
Gamma	c(0.1,0.5,1.0,2.0,10)
	c(0.001 , 0.01 , 0.1, 1 ,5 ,10 ,100,1000),
	c(0.001, 0.1, 0.5, 1 ,10 ,100),
	seq(0.3,0.6,by=0.05)

Fct Coût - pénalisation [C]	2^(-10:0)
	seq(0.01,1,by=0.01)
	c(0.05, 0.25, 0.5, 0.75, 1)
	seq(0.3,0.6,by=0.05)

Analyse : (attention : défaut du nombre de vecteurs supports avec ce R-package)
. 1 seule séquence de paramètres a été testée et a produit les valeurs du tableau de synthèse (modification du pas = 0.05),
. le travail sur les 6150 lignes permet d'améliorer les Taux de biens classés **(52.97 à 73.11%) et AUC (0.50 à 0.5675),**
. à noter la valeur de la F-mesure = **0.7404** (bonne qualité),
En conséquence → **il est intéressant de tester le noyau Radial avec valeurs $C = 1000$ (2e itération) et avec la recherche de γ.**

LA PERFORMANCE DES 4 CADRANS

Modèle sélectionné : noyau Radial C = 0.55 et $\gamma = 0.5$

Croisement des Xi	Accuracy = fonction (γ)
	
Courbe de ROC	**Courbes des critères d'erreurs**
	

Signalétique:

. ROC curve points of equal error rate (EER), . maximum accuracy, . maximum kappa, . minimum squared error rate (MSER = square of minimum euclidean distance between ROC curve and top left corner for TMR = TNMR = 1))	. The true match rate (TMR) indicates the rate of how much test data set samples were correctly identified as being of the positive class (in our case: les NON). . The false match rate (FMR, counterpart to true non match rate TNMR) indicates the rate of how much test data set samples were incorrectly identified as being of the positive class.

[source : https://geekoverdose.wordpress.com/2014/07/25/svm-classification-example-with-performance-measures-using-r-caret/]

Analyse :

. la performance « **moyenne** » de la courbe ROC,
. la proximité « quasi-linéaire » des 2 courbes d'accuracy (lié au tuning du modèle, l'accuracy est le critère de performance),
. la courbe kappa est en légère croissance,
. le graphe croisement des Xi (i.e. scatterplot) : la variable Cotisation (LogC_AN_TTC) « travaille » avec toutes les autres explicatives.

(Résultats de : **d'itérer le Radial avec valeurs $C = 1000$ (2e itération) et recherche de γ.**)

PERIODE 2 *2.5 GridSearch - CV - parallel (4e itération b)*

hypo : relation linéaire Yi ~ Xi	5/6 non
Nb total d'observation	6150
Echantillonnage	1%

Etiquette Yi	trsf_dev			
Poids des Yi[i]	[N;O] = [73.31 ; 26.69]			
Nb Xi	6			
Detail Xi	LogC_AN_TTC + PUIS_FISC + COEF + AGE_CLI + NB_ENF + ANC_VEH			
standardisation Xi	centrage-réduction de Xi ~ [m ~ 1; var ~ 0]			
normalisation Xi	C_AN_TTC -> LogC_AN_TTC			
calcul parallèle	3-cpu	3-cpu	3-cpu	3-cpu
Langage Prog	R-Caret/kernlab	R-Caret/kernlab	R-Caret/kernlab	R-Caret/kernlab

METRICS	LINEAR	**RADIAL (RBF)**	POLYNOMIAL	SIGMOID
stratégie SVM	1 vs 1	1 vs 1	1 vs 1	1 vs 1
apprentissage	0,7	0,7	0,7	0,7
test	0,3	0,3	0,3	0,3
Correction Poids des Yi[i]	weights	weights	weights	weights
k (cv- k folds)		10		
répétition		5		
méthode opti paramètre		Grid + parallel		
critère de performance		accuracy		
type : classif / reg.	C-classification	C-classification	C-classification	C-classification
Probabilité		oui		
Gamma		**0.15**		
Fct Coût - pénalisation [C]		**1000**		
Degré polynôme				
Temps Calcul (seconde)		21.91 mins		
Tx de biens classés (Accuracy) %		**0,7452**		
Tx d'Erreur %		0,2548		
AUC		**0.5685**		
Nombre de vecteurs supports				
Kappa		0.00		
F1 (F-mesure)		**0.7452**		
Contrôle Graphique	oui	oui	oui	oui

Gamma	**seq(0.15,0.20,by=0.01)**

Fct Coût - pénalisation [C]	**1000**

Analyse :
. on finalise **le SVM Radial** $C = 1000, \gamma = 0.15$ (i.e. la valeur minimale obtenue pour ce paramètre de réglage),
. le calcul avec la totalité de l'échantillon SSP + la pénalisation augmentée du SVM a permis d'augmenter la **performance de classification** par rapport au modèle précédent *(4e itération a)* car les valeurs empiriques augmentent sur nos 3 critères de performance :
. le taux de biens classés passe de **73.11% à 74.52%,** l'AUC passe de **0.5675 à 0.5685** et la F-mesure passe de **0.7404 à 0.7452.**

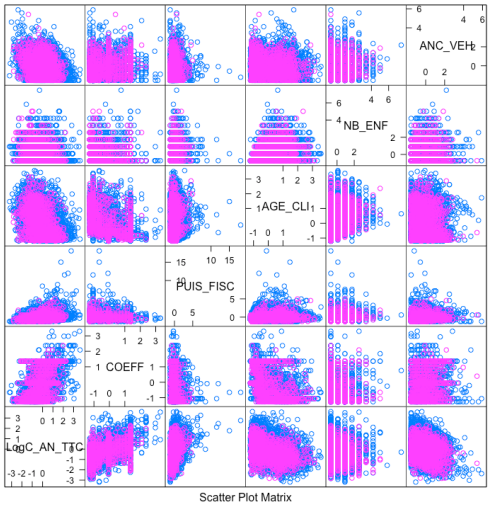
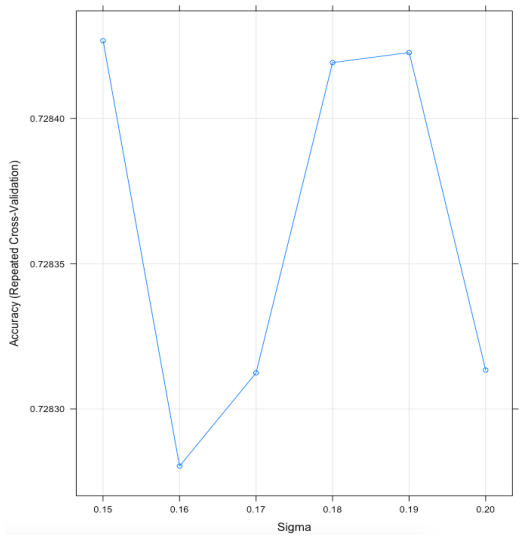
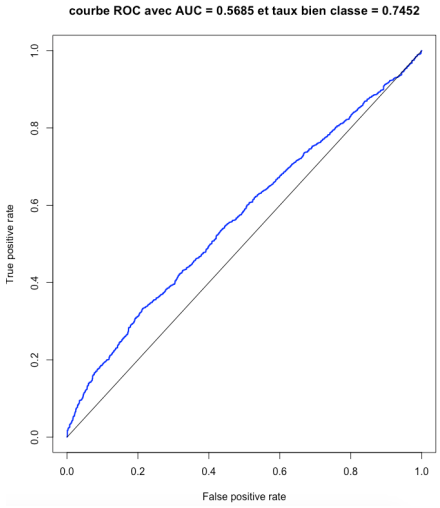
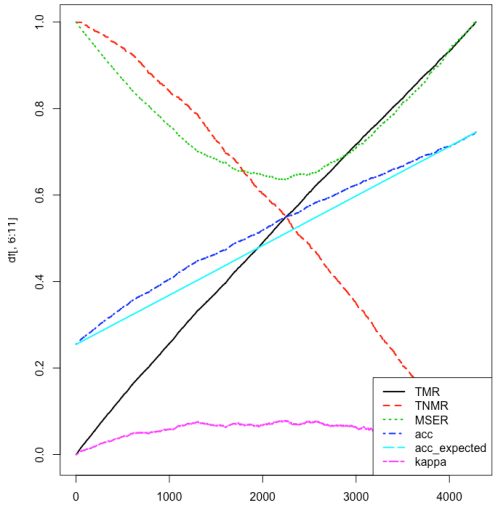
➔ Ces résultats sont nominaux (P.35) par rapport formalisme du noyau **RBF : (↗) C et (↘) γ ➔ et (↘) Biais = Accuracy (↗)** et le modèle sélectionné semble **bien « généraliser »** avec une valeur empirique de la F-mesure plus élevée (cad, le classifieur détecte bien et a une capacité à reconnaître les classes de la variable d'intérêt - **A VERIFIER dans le détail, avec l'audit des différents critères de performance**).

Conclusion : l'optimisation GridSearch du SVM Radial, fortement pénalisé, en mobilisant une procédure de calcul parallèle pour traiter l'effectif total de notre échantillon SSP nous permet de former le classifieur SVM le plus performant sur les 3 critères : AUC, F-mesure et taux de biens classés.

Reste à faire : Comparer sur ces « métriques de performance » notre SVM optimisé avec celles du RF optimisé.

LA PERFORMANCE DES 4 CADRANS

Modèle sélectionné : noyau Radial C = 1000 et $\gamma = 0.15$

Croisement des Xi	Accuracy = fonction (γ)
	
Courbe de ROC	**Courbes des critères d'erreurs**
	

Signalétique:

. ROC curve points of equal error rate (EER), . maximum accuracy, . maximum kappa, . minimum squared error rate (MSER = square of minimum euclidean distance between ROC curve and top left corner for TMR = TNMR = 1))	. The true match rate (TMR) indicates the rate of how much test data set samples were correctly identified as being of the positive class (in our case: les NON). . The false match rate (FMR, counterpart to true non match rate TNMR) indicates the rate of how much test data set samples were incorrectly identified as being of the positive class.

[source : https://geekoverdose.wordpress.com/2014/07/25/svm-classification-example-with-performance-measures-using-r-caret/]

Analyse :

. la performance « **moyenne** » de la courbe ROC,

. la proximité « quasi-linéaire » des 2 courbes d'accuracy (lié au tuning du modèle, l'accuracy est le critère de performance) ,

. la courbe kappa qui s'améliore par rapport au modèle dernier modèle *(4e itération a)*

. le graphe croisement des Xi (i.e. scatterplot) : la variable Cotisation (LogC_AN_TTC) « travaille » avec toutes les autres explicatives.

8.4. Période 2 : analyse comparée résultats SVM vs autres classifieurs

Afin de comparer des « objets comparables », nous allons utiliser les 2 classifieurs concurrents et les plus performants sur notre dataset optimisé de la période 2. C'est à dire les 2 modèles de classification optimisés selon une procédure GridSearch et calcul parallèle définis comme suit : **le RF "grid search - ROC" et le SVM Gaussien C = 1000 et $\gamma = 0.15$.**

Rappel :
- Le modèle des Forêts Aléatoires "grid search - ROC" a été sélectionné sur le critère de performance « ROC/AUC », tandis que,
- Le modèle SVM Gaussien C = 1000 et $\gamma = 0.15$ a été sélectionné sur le critère de performance « Accuracy = taux de biens classés ».

L'objectif du travail : en confrontant la performance dans la précision de classification des 2 classifieurs retenus, nous allons construire une série de mesure et de tests statistiques permettant de vérifier l'hypothèse alternative,

H0 : absence de différence de performance, la performance des 2 classifieurs est identique,
H1 : présence de différence de performance, la performance des 2 classifieurs est différente,

Enfin, essayer de conclure en dégageant une analyse qualitative des résultats observés.

(i) Tableau d'analyse comparée :

PERIODE 2	Tableau 6 - synthèse de la performance des 2 classifieurs retenus		
hypo : relation linéaire Yi ~ Xi	5/6 non		
Nb total d'observation	6150		
Echantillonnage	1%		
Etiquette Yi	trsf_dev		
Poids des Yi[i]	[N;O] = [72.95 ; 27.04]		[N;O] = [73.31 ; 26.69]
Nb Xi	6		
Détail Xi	LogC_AN_TTC + PUIS_FISC + COEF + AGE_CLI + NB_ENF + ANC_VEH		
standardisation Xi	centrage-réduction de Xi ~ [m ~ 1; var ~ 0]		
normalisation Xi	C_AN_TTC -> LogC_AN_TTC		
calcul parallèle	3-cpu		3-cpu
Langage Prog - pkg	R-caret		R-Caret/kernlab

METRICS	**RF "grid search - ROC"**	METRICS	**SVM Grid Gaussien**
apprentissage	0.7	apprentissage	0,7
test	0.3	test	0,3
Correction Poids des Yi[i]	wts	Correction Poids des Yi[i]	weights
k (cv- k folds)	10	k (cv- k folds)	10
répétition	10	répétition	5
méthode opti paramètre	grid	méthode opti paramètre	grid
critère de performance	roc	critère de performance	accuracy
type : classif / reg.	classif	type : classif / reg.	C-classification
		stratégie SVM	1 vs 1
ntree (nb d'arbre)	350		
mtry (nb var / noeud)	mtry = 2		
		Probabilité	oui
		Gamma [γ]	0.15
		Fct Coût - pénalisation [C]	1000
		Nombre de vecteurs supports	NaN

valeur maximum sur le critère			
temps de calcul	5,43 mins	Temps Calcul (seconde)	21.91 mins
Tx de biens classés (Accuracy)	0,7131	**Tx de biens classés (Accuracy) %**	0,7452
Tx d'Erreur	0,2869	Tx d'Erreur %	0,2548
AUC	0,6143	**AUC**	0.5685
Kappa	0,0560	**Kappa**	0.00
F1 (F-mesure)	0,7485	**F1 (F-mesure)**	0.7452
Contrôle Graphique	oui	Contrôle Graphique	oui
importance des descripteurs 1	LogC_AN_TTC		
valeur	100,00		
importance des descripteurs 2	AGE_CLI		
valeur	57,11	*(autres métriques – sorties [R])*	

Accuracy	0,7131	**Accuracy**	0,7452
95% CI	(0,6919, 0,7337)	**95% CI**	(0,7319, 0,7582)
No Information Rate	0,7392	**No Information Rate**	0,7452
P-Value [Acc > NIR]	0,9946	**P-Value [Acc > NIR]**	0,5081
Kappa	0,056	**Kappa**	0
Mcnemar's Test P-Value	<2e-16	**Mcnemar's Test P-Value**	<2e-16
Sensitivity	0,9215	**Sensitivity**	1
Specificity	0,1227	**Specificity**	0
Pos Pred Value	0,7485	**Pos Pred Value**	0,7452
Neg Pred Value	0,3554	**Neg Pred Value**	NaN
Prevalence	0,7392	**Prevalence**	0,7452
Detection Rate	0,6811	**Detection Rate**	0,7452
Detection Prevalence	0,91	**Detection Prevalence**	1
Balanced Accuracy	0,5221	**Balanced Accuracy**	0,5
Positive' Class	N	**Positive' Class**	N

Sensitivity = sensibilité capacité à identifier les VP (N)

Specificity = spécificité capacité à identifier les VN (O)

(ii) Mesure et tests statistiques pour comparer nos 2 classifieurs :

ii.1 : Intervalle de confiance pour l'erreur estimée :

-> Formalisme de la Mesure :

Les classifieurs sont évalués sur l'ensemble test : l'erreur empirique du classifieur calculé sur l'échantillon test fournit une estimation de l'*erreur réelle* du classifieur, le seul paramètre qui nous intéresse. Il est important que l'échantillon test soit indépendant de l'échantillon d'apprentissage pour que l'estimation de l'erreur soit *non biaisée*. Mais même non biaisée, on aura d'autant plus *confiance* dans cette estimation que l'effectif de l'échantillon test est important. On peut démontrer en statistiques que :

— si l'échantillon test contient N exemples,
— si l'erreur estimée du classifieur calculée sur l'échantillon test est e,

alors, avec une confiance supérieure à 95%, l'erreur réelle du classifieur est comprise dans l'intervalle

$$[e - 1.96\sqrt{\frac{e(1-e)}{N}}, e + 1.96\sqrt{\frac{e(1-e)}{N}}].$$

[source : http://pageperso.lif.univ-mrs.fr/~francois.denis/IAAM1/tp2M12016.pdf / comparaison de classifieurs]

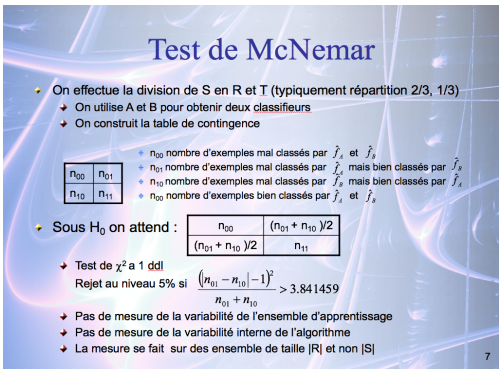
-> Statistique et conclusion :

METRICS	RF "grid search - ROC"	METRICS	SVM Grid Gaussien
Tx d'Erreur	**0,2869**	Tx d'Erreur %	**0,2548**
95% CI = (1- erreur réelle)	(0.6919, 0.7337)	95% CI	(0.7319, 0 ;7582)
IC à 95% de l'erreur réelle *(calculé)*	(0.2592, 0.3146)	IC à 95% de l'erreur réelle *(calculé)*	(0.2418, 0.2678)
Amplitude IC à 95% de l'erreur réelle	0,0555	Amplitude IC à 95% de l'erreur réelle	0,0260

-> Analyse : avec le calcul à la main de l'intervalle de confiance de l'erreur réelle pour les 2 classifieurs, **le modèle SVM Gaussien est plus précis** car l'amplitude son IC est plus resserrée et la borne supérieure est plus **basse = 0.2678**. (Rappel : IC à 95% = 95% de erreurs réelles empiriques sont situées dans l'IC)

ii.2 : Test de Mc Nemar :

-> Formalisme du test :



[source : Thomas G. Dietterich Approximate Statistical Tests for Comparing, 1998]

-> Statistique et conclusion :

Matrice de Confusion

	(a)	(t)
(t)	0,7	0,3
6150	4305	1845

RF "grid search - ROC"	reference N	reference O
Prediction N	1256	422
Prediction 0	107	59

SVM Grid Gaussien	reference N	reference O
Prediction N	1374	470
Prediction 0	0	0

Mc Nemar	***(à 5%)***			
n00	999			
n11	2689			
n01	951,5			
n10	892,5			
	3364	stats test	à 5%	sur le critère accuracy
	1844	1,82	3,84	(inf.) on accepte H0

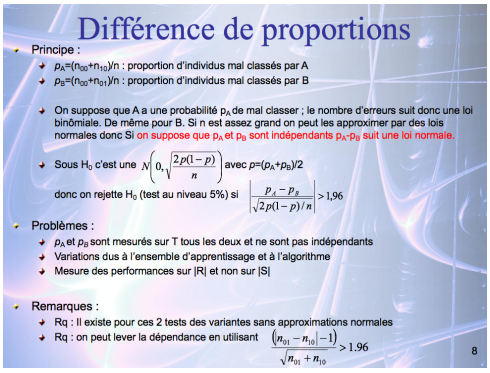
H0 : égalité des classifieurs
H1 : les classifieurs sont différents

conclusion: SVM = RF

-> Analyse : en se basant sur les effectifs de la matrice de confusion, le test de Mc Nemar travaille sur le taux de biens classés (accuracy) pour comparer les 2 classifieurs. **A 5%,** on accepte H0 : absence de différence de performance**, la performance des 2 classifieurs est identique.**

ii.3 : test de différence de proportions

-> Formalisme du test :



[source : Thomas G. Dietterich Approximate Statistical Tests for Comparing, 1998]

-> Statistique et conclusion :

Matrice de Confusion

	(a)	(t)
(t)	0,7	0,3
6150	4305	1845

RF "grid search - ROC"	reference N	reference O
Prediction N	1256	422
Prediction 0	107	59

SVM Grid Gaussien	reference N	reference O
Prediction N	1374	470
Prediction 0	0	0

Proportions	***(à 5%)***			
pA	0,30756098			
pB	0,31715447			
pA-pB	0,0095935			
p	0,31235772			
2p(1-p)/n	6,9851E-05	stats test	à 5%	sur le critère accuracy
Rc[2p(1-p)/n]	0,00835766	1,15	1,96	(inf.) on accepte H0
H0 : égalité des classifieurs				
H1 : les classifieurs sont différents				
		conclusion:		**SVM = RF**

(calculé)

-> <u>Analyse</u> :

. En se basant sur les effectifs de la matrice de confusion, le test de Différence de proportions travaille sur le taux de biens classés (accuracy) pour comparer les 2 classifieurs. **A 5%,** on accepte H0 : absence de différence de performance, **la performance des 2 classifieurs est identique.**

. Le résultat de ce test est identique à celui Mc Nemar, les résultats sont convergents et consistants puisqu'ils se basent sur une métrique Accuracy calculée sur l'échantillon test et avec un apprentissage en validation croisée K = 10 plis. Ces résultats sont « sans surprise » car obtenus dans des conditions expérimentales de mises en œuvre relativement proches. La différence entre les 2 mesures « d'Accuracy » est resserrée (**0,032** = 0,7452 - 0,7131). De plus, les différents papiers de recherche sur la performance des classifieurs indiquent que le RF Grid search est légèrement plus performant que le SVM Grid search selon la nature de problème de classification traité et selon le jeu de données associé.

Abstract

We evaluate **179 classifiers** arising from **17 families** (discriminant analysis, Bayesian, neural networks, support vector machines, decision trees, rule-based classifiers, boosting, bagging, stacking, random forests and other ensembles, generalized linear models, nearest-neighbors, partial least squares and principal component regression, logistic and multinomial regression, multiple adaptive regression splines and other methods), implemented in Weka, R (with and without the caret package), C and Matlab, including all the relevant classifiers available today. We use **121 data sets**, which represent **the whole UCI** data base (excluding the large-scale problems) and other own real problems, in order to achieve significant conclusions about the classifier behavior, not dependent on the data set collection. **The classifiers most likely to be the bests are the random forest** (RF) versions, the best of which (implemented in R and accessed via caret) achieves 94.1% of the maximum accuracy overcoming 90% in the 84.3% of the data sets. However, the difference is not statistically significant with the second best, the SVM with Gaussian kernel implemented in C using LibSVM, which achieves 92.3% of the maximum accuracy. A few models are clearly better than the remaining ones: random forest, SVM with Gaussian and polynomial kernels, extreme learning machine with Gaussian kernel, C5.0 and avNNet (a committee of multi-layer perceptrons implemented in R with the caret package). The random forest is clearly the best family of classifiers (3 out of 5 bests classifiers are RF), followed by SVM (4 classifiers in the top-10), neural networks and boosting ensembles (5 and 3 members in the top-20, respectively).

[source : Fernández-Delgado, M., Cernadas, E., Barro, S. et Amorim, D. [2014]. Do we need hundreds of classifiers to solve real world classification problems. J. Mach. Learn. Res , 15 (1), 3133–3181]

. En revanche, l'alignement de ces principaux résultats pour ce jeu de données avec les résultats **« main stream »** d'autres études empiriques doivent être « stress tested » par le fait que ces deux classifieurs ont été sélectionnés sur des ces critères de performances différents, le ROC-AUC pour le RF et l'Accuracy pour le SVM. **Mais surtout par le fait que la seule véritable métrique de performance de classification sur un problème de classification binaire supervisée avec déséquilibre de classes est le ROC-AUC.** Nous devons donc continuer notre comparaison des deux classifieurs avec le critère AUC.

(iii) Analyse ROC – AUC :

METRICS	RF "grid search - ROC"	METRICS	SVM Grid Gaussien
Tx de biens classés (Accuracy)	0,7131	**Tx de biens classés (Accuracy) %**	0,7452
Tx d'Erreur	0,2869	Tx d'Erreur %	0,2548
AUC	0,6143	**AUC**	0.5685
Kappa	0,0560	**Kappa**	0.00
F1 (F-mesure)	0,7485	**F1 (F-mesure)**	0.7452
Accuracy	0,7131	**Accuracy**	0,7452
95% CI	(0,6919, 0,7337)	**95% CI**	(0,7319, 0,7582)
No Information Rate	0,7392	**No Information Rate**	0,7452
P-Value [Acc > NIR]	0,9946	**P-Value [Acc > NIR]**	0,5081
Kappa	0,056	**Kappa**	0
Mcnemar's Test P-Value	<2e-16	**Mcnemar's Test P-Value**	<2e-16
Sensitivity	0,9215	**Sensitivity**	1
Specificity	0,1227	**Specificity**	0
Pos Pred Value	0,7485	**Pos Pred Value**	0,7452
Neg Pred Value	0,3554	**Neg Pred Value**	NaN
Prevalence	0,7392	**Prevalence**	0,7452
Detection Rate	0,6811	**Detection Rate**	0,7452
Detection Prevalence	0,91	**Detection Prevalence**	1
Balanced Accuracy	0,5221	**Balanced Accuracy**	0,5
Positive' Class	N	**Positive' Class**	N

Sensitivity = sensibilité capacité à identifier les VP (N)
Specificity = spécificité capacité à identifier les VN (O)

Tableau 7 - analyse ROC – « détecter les avions allemands sans détecter tous les oiseaux »

1er résultat : sur le critère de l'AUC, le RF est plus performant que le SVM.

2e résultat : sur les critères AUC complété du Kappa et de la F-mesure, le RF est plus performant que le SVM.

3e résultat :
. Sur les critères **de Sensibilité et Spécificité**, on note le résultat suivant : Le classifieur SVM arrive à 100% à détecter les étiquettes N (i.e. la classe majoritaire sur notre dataset) **et à 0% les étiquettes O (les « Oui »),**
. A contrario, le RF sait détecter à 12.27 % les événements « Oui » et à 92.15% les étiquettes « Non » sur notre dataset.

Analyse :
. Sur le critère de l'AUC et des critères associés Sensibilité & Spécificité, le classifieur SVM n'est pas assez performant sur ce jeu de données, il ne sait « apprendre que du NON ».
. Le RF « fait légèrement mieux » que le SVM, mais la détection à 12.27 % des évènements « OUI » qui nous intéressent n'est pas un résultat satisfaisant pour répondre aux « enjeux business de la Matmut sur la transformation des devis commerciaux en contrats sociétaires ».

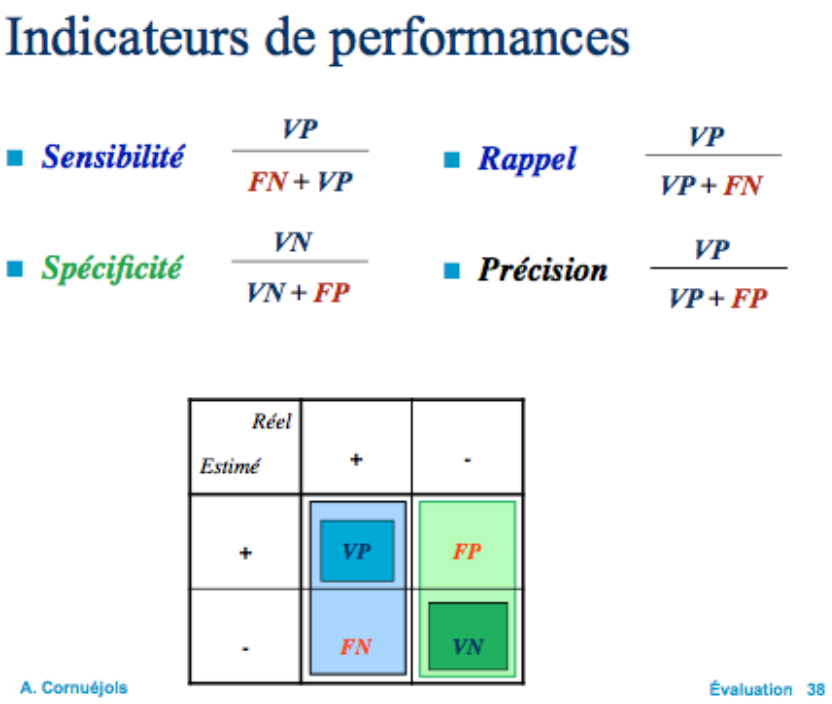
Résultats finaux:

. Le classifieur « Random Forest Grid Search – Roc » est meilleur que le « SVM Gaussien Grid Search » sur notre jeu de données et avec un problème de classification binaire supervisée comportant des classes moyennement déséquilibrées
(« répartition à grosse maille », Non ~ 70%, Oui ~30%),

. Sur notre dataset : le vrai critère de performance de mesure d'un classifieur est le critère d'ROC-AUC et non le critère d'Accuracy (le taux de biens classés),

. Le plus important : c'est l'analyse comparée de l'ensemble des résultats des mesures de performance des deux classifieurs qui nous a permis conclure et non la lecture empirique d'une seule mesure.

Pour rappel : courbe ROC = courbe Receiver Operating Curve (radar UK – 2nd Guerre Mondiale – rôle des critères Sensibilité & Spécificité)



[source : Évaluation de l'apprentissage : méthodes A. Cornuéjols LR]

⚠ **Bonnes Pratiques DATALAB :**

Sur un problème de classification bi-classes ou multi-classes supervisée avec des données déséquilibrées :

1. Tester plusieurs classifieurs, dont au moins un modèle « tuned accuracy » et un autre modèle « tuned AUC »,
2. Organiser la confrontation des résultats et réaliser des tests statistiques pour auditer les résultats empiriques,
3. Travailler la famille des modèles pour détecter les événements rares,
4. Précéder les modèles de Machine Learning par des modèles d'analyse descriptives des données (i.e. analyses factorielles).

*****

Remarque : afin d'être exhaustif, il existe un indicateur de mesure de performance de classifieur utilisé par la communauté scientifique anglo-saxonne, le G-mean, dans le cas de problème de classification avec classes déséquilibrées :

Définition :

$$G - mean = \sqrt{\frac{TP}{TP+FN} \times \frac{TN}{TN+FP}} \qquad (7)$$

[source : Imbalanced Data Classification using Complementary Fuzzy Support Vector Machine Techniques and SMOTE[21] - **October 5-8, 2017**]

Statistiques :

METRICS	RF "grid search - ROC"	METRICS	SVM « grid RBF accuracy »
G-mean	**0.5158**	G-mean	**NaN**

Conclusion : défaut de calcul du G-mean pour le SVM Gaussien, le RF est meilleur.

[21] Learning from Imbalanced Data : http://ieeexplore.ieee.org/document/5128907/?reload=true

9. Conclusion

9.1. Bilan méthodologique et résultats

- Suite à ce travail de spécification du « meilleur classifieur adapté à notre problème », il faut retenir que :

Spécifications	**Résultats observés**
Résultats – période1	. 6 variables explicatives continues* sélectionnées (vs. 112 Xi du fichier de départ)
Le classifieur retenu	. le classifieur RF est le modèle retenu : AUC = 0.69 et F-Mesure = 0.85
Variables importantes	. 2 variables discriminantes : Cotisation Annuelle et Age Client [*alignées avec le pricing Matmut*]
Spécification du modèle	. pas de relation linéaire entre les Xi et le Y
	. les Xi discriminantes ne sont pas toutes identifiées (tarifs concurrents)
	. le rôle important des classes déséquilibrées du Y
Résultats – période2	. un échantillon sondage stratifié proportionnel (n = 6150) est suffisant pour le calcul
Méthodologie	. une procédure de gridsearch, CV**, calcul parallèle permet le calcul de classifieurs optimisés
Famille des Forêts Aléatoires	. le meilleur modèle : RF "gridsearch - ROC", AUC = 0.61 et F-mesure = 0.74
Variables importantes	. Cotisation Annuelle et Age Client
Famille des SVM	. le meilleur modèle : SVM Gaussien C = 1000 et $\gamma = 0.15$, AUC = 0.56 et F-mesure = 0.74
Le critère de performance retenu	. le critère ROC-AUC (et ses critères Sensibilité et Spécificité) sélectionne le meilleur classifieur
Le classifieur retenu	. le Random Forest "grid search - ROC"
Le classifieur non retenu	. le SVM Gaussien qui ne sait apprendre que du « NON »
La réponse Business	. les résultats insuffisants du RF qui détecte seulement 12.27% de « OUI »
Avantages & Inconvénients du RF	**Analyse**
Inconvénients	. modèle de machine learning « boîte noire » *(… on ne maîtrise pas ce qu'il fait …)*
Avantages	. le ratio performance / coût de mise en œuvre en fait un classifieur très utilisé
Avantages & Inconvénients du SVM	**Analyse**
Inconvénients	. déploiement méthodologique exigeant (i.e. statisticien) et calculs numériques coûteux
Avantages	. l'apport des réponses aux questions qui qualifient la relation Yi ~ Xi (i.e. les données)

* Xi quantitatives – continues, résultat aligné avec la qualité des données du Système d'Information MATMUT.
** CV = validation croisée

9.2. Prolongements

Type de recommandation	**Détail**
Recommandation technique « modèles statistiques » - piste 1	**. itérer les 2 classifieurs finaux en apportant des compléments de spécifications :** - apporter des Xi supplémentaires : intégrer les données de tarif concurrent, - tester les modèles hybrides : classifieur avec un algorithme de redressement des classes déséquilibrées (SMOTE : Synthetic minority oversampling technique)
Recommandation technique « modèles statistiques » - piste 2	. développer les modèles de détection d'événement rare (ex. Fraude & modèle QDA) . tester les réseaux de neurones sur environnement technique GPU

Tableau 8 - synthèse des résultats et recommandations

⚠ **Bonnes Pratiques DATALAB :**

Sur un problème de classification bi-classes ou multi-classes supervisée avec des données déséquilibrées :

1. Formaliser la rédaction de note de cadrage méthodologique des sujets « machine learning (ML) »,
2. Exiger une préparation des données pour garantir les résultats de ML (audit des données et dataset optimisé),
3. Tester plusieurs classifieurs, dont au moins un modèle « tuned accuracy » et un autre modèle « tuned AUC »,
4. « Stress tester » les résultats avec des tests statistiques pour vérifier les résultats annoncés,
5. Travailler les modèles pour détecter les événements rares,
6. Précéder les modèles de Machine Learning par des modèles d'analyse descriptives des données (i.e. analyses factorielles),
7. Valider les spécifications du classifieur retenu selon **la méthode kaggle.com : « en mode blind test », tester la performance réelle du modèle en validation croisée avec 1 échantillon test, prélevé au départ du chantier, du Dataset initial.**

**** repère méthodologique LDA / DQA ****

Modèle LDA : analyse linéaire discriminante :

Data Mining prédictif | Classement

Analyse linéaire discriminante (LDA)

- Principe : les données sont issues d'un mélange de lois normales et chaque classe possède un vecteur moyen distinct. Formellement, étant donné une classe C_k on suppose que $\mathbf{x}|C_k \sim \mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma})$ où $\boldsymbol{\mu}$ est le vecteur moyen et $\boldsymbol{\Sigma}$ la matrice de variance-covariance. On cherche alors à estimer $\boldsymbol{\Sigma}$ et pour chaque classe C_k, le vecteur $\boldsymbol{\mu}_k$. Puis, on donne un score et une prédiction à un point $\mathbf{x}$ quelconque à l'aide de $P(C_k|\mathbf{x})$ que l'on calcule grâce à la règle de Bayes.
- Remarques importantes :
  - Le cas linéaire considère que toute classe C_k possède la même matrice de variance-covariance $\boldsymbol{\Sigma}$: hypothèse d'homoscédasticité.
  - On aboutit alors à une fonction de score qui est linéaire en $\mathbf{x}$ appelée également **score de Fisher**.
  - La méthode peut être vue telle une technique de réduction de dimension où l'on cherche simultanément à maximiser l'inertie inter-classe et minimiser l'inertie intra-classe.
- Commande `lda [MASS]`.

J. Ah-Pine (Univ-Lyon 2) | Data Mining | L3 CESTAT - 2016/2017 | 82 / 94

Modèle QDA : analyse quadratique discriminante :

Data Mining prédictif | Classement

Analyse quadratique discriminante (QDA)

- Principe : c'est le même cadre formel que précédemment càd pour chaque classe C_k on suppose que $\mathbf{x}|C_k \sim \mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$. Mais ici toute classe a un vecteur moyen et une matrice de variance-covariance distincte $\boldsymbol{\Sigma}_k$. L'abandon de l'hypothèse d'homoscédasticité conduit à une fonction de score qui est quadratique en $\mathbf{x}$.
- Remarques :
  - Pour LDA, la fonction de score est linéaire en $\mathbf{x}$ ce qui veut dire que dans l'espace de description, on peut tracer des frontières linéaires (hyperplans) séparant les classes C_k entre elles.
  - Pour QDA, la fonction de score est quadratique en $\mathbf{x}$ ce qui veut dire que les frontières séparant les classes dans l'espace de description sont des courbes.
  - QDA est plus flexible que LDA mais demande plus de calculs car il y a q matrices $\boldsymbol{\Sigma}_k$ à estimer.
- Commande `qda [MASS]`.

J. Ah-Pine (Univ-Lyon 2) | Data Mining | L3 CESTAT - 2016/2017 | 83 / 94

[source : https://eric.univ-lyon2.fr/~jahpine/cours/cm_dm-l3cestat_1617.pdf]

Figure 19 : rappel méthodologique LDA vs QDA

10. Annexes :

10.1. Note de cadrage de la thèse professionnelle

date	28-févr-17
nom	le picaut
prénom	didier
email	lepicaut@gmail.com
téléphone	06 42 45 10 76
objet	note cadrage thèse professionnelle

date de début	09-mars-17
date de fin	31-oct-17
remise du manuscrit	19-oct-17
soutenance orale	27-oct-17
format du livrable	document Ms Word + annexes techniques de code
nombre de page	50 pages maximum avec les éléments de code R à auditer

dataset	
sources des données	données MRSQ - année 2015
nombre de variables	112 « features » - variables
nombre d'enregistrements	728674 lignes (données brutes - i.e. extraction infocentre)
nombre de variables discriminantes	6 variables quantitatives explicatives (travaux préliminaires)
périmètre des devis concerné	Devis auto (hors 2 roues) et particulier (hors usage professionnel)

objectifs	
objectifs poursuivis, qualitatifs :	**identifier le meilleur classifieur pour faire du Machine Learning sur un problème de classification binaire supervisée avec CLASSES DESEQUILIBREES ?**
objectifs poursuivis, quantitatifs :	package CARET > analyse comparée de la performance des classifieurs (comparaison en grille d'analyse)
indicateurs de résultats :	AUC ROC / critère de l'ACCURACY avec test de Mc Nemar sur la matrice de confusion

ressources	
moyens nécessaires humains :	1 ETP producteur
répartition des tâches :	soir, week-end, jour de congés payés
répartition des compétences :	du preprocessing à la datavizualisation des résultats des modèles

architecture technique	
machine :	MacBook Air, core I5, 4 GO DDR3 , Intel HD Graphics 5000 1536 Mo
calcul :	Mac OS Sierra, 10.12.6
logiciels :	Rstudio Version 1.0.153, R version 3.4.0

la question managériale	marché de l'assurance automobile > comment transformer des devis prospects en contrats sociétaires ?
titre de la thèse professionnelle	**modèles statistiques : les modernes contre les anciens, quel est le meilleur classifieur ?**
résultat connu	**à la MATMUT : toute la politique de souscription des risques est expliquée par la politique tarifaire (modèle pricing - actuaire = relation linéaire fort entre les variables)**
état de l'art dans le domaine afférent	marché de l'assurance automobile (IARD) : pour un assureur mutualiste > je prends ou ne prends pas le risque d'assurer un contrat auto (Classification Oui/Non) versus un Bancassureur > j'apprécie, je tarifie et le prends les risques clients qui m'intéresse (classification par régression - scoring) ?

plan détaillé de la thèse	
1. introduction	
2. le contexte et les enjeux du projet	**exercice 2015 : 1ère année où l'opérateur mutualiste Matmut stagne sur son cœur de métier historique "le marché du contrat d'assurance automobile" > optimiser la transformation des devis en "OUI"**
période 1. 1ère utilisation de classifieurs	**modèle de discrimination avec sélection de variables quantitatives**
. modèle géométrique	ACP
	Analyse Factorielle discriminante
. modèle inférence (fréquentiste)	LDA : Analyse Linéaire Discriminante
	Régression Logistique
	les Forêts Alatoires : RF
	Gradient boosting : Xgboost
3. La méthode des SVM	formalisme de la théorie du SVM : le principe, SVM linéaire, les variantes, ...)
4. les données	
5. Application des SVM	**Application des SVM, comparaison avec les autres méthodes**
période2. application numérique SVM	**analyse comparée des résultats des classifieurs RF vs SVM**
. Formatage dataset	préparation d'un dataset nettoyé commun à tous les classifieurs, avec échantillon stratifié proportionnel de 6150 lignes
. Randomforest	application du modèle RF avec grid search
. SVM	application du modèle SVM avec grid search
. Analyse comparée	comparaison des résultats des classifieurs RF vs SVM
6. Conclusion	
7. Annexes	
8. Bibliographie	

parties prenantes	
producteur	Didier Le picaut - alternant
matmut - tuteur entreprise	Valérie Brivet - demande autorisation d'utilisation des données pour réaliser une thèse professionnelle en vue de l'obtention de la certification Master Spécialisée Insa Rouen
insa directeur Master ESD	Bruno Portier - validation de la note de cadrage thèse professionnelle

10.2. Eléments descriptifs du Groupe MATMUT

10.2.1. L'organisation du Groupe MATMUT

1 Groupe MATMUT

Acteur majeur sur le marché français, le Groupe Matmut assure aujourd'hui près de 3,2 millions de sociétaires et plus de 6,8 millions de contrats. Créée en 1961. La société fonctionne démocratiquement sous le contrôle de ses sociétaires.

Le Groupe Matmut se compose de 7 entités :

- Matmut
- La mutuelle Matmut
- Sa « mutuelle sœur » Matmut Mutualité
- AMF Assurances.
- Inter Mutuelles Entreprises
- Matmut Vie.
- Matmut Protection Juridique.

1.1 Organisation MATMUT

La **Mutuelle Assurance des Travailleurs Mutualistes** (Matmut) est en charge de la stratégie du Groupe et des fonctions centrales telles que finances, ressources humaines, informatique…

Elle possède toutes les filiales opérationnelles d'assurance : Matmut Vie, Matmut Protection Juridique, Inter Mutuelles Entreprises, AMF *Assurances*.

Elle apporte également aux assurés des prestations en matière de Protection Juridique vie privée et d'assistance à domicile.

Elle conçoit, gère et distribue des contrats d'assurance "dommages" pour les particuliers :

- Auto/Moto
- Habitation/Responsabilité civile
- Accidents corporels
- Assurance scolaire
- Bateau
- Chasse

Et aussi pour les professionnels :

- Autoentrepreneur
- Commerçant, Artisan
- Entreprise
- Profession libérale
- Profession de santé
- Association

source : extrait - thèse professionnelle de Me Naima MOKHEDDES – 2016.

10.2.2. Le réseau commercial MATMUT

1.2 Les réseaux commerciaux

Données sur les opérations commerciales au 31-12-2015

LES ACTEURS		
525 AGENCES…	2 101 collaborateurs dont :	• 1 339 Conseillers en Assurance sédentaires • 21 Conseillers Particuliers Professionnels itinérants • 575 Responsables d'Agence et Adjoints au Responsable d'Agence
5 PLATES-FORMES TÉLÉPHONIQUES	249 collaborateurs dont :	• 208 Conseillers Téléphoniques en Assurance • 24 Responsables d'Équipe Téléphonique
L'ACTIVITÉ		
Indicateurs	Délégation agences	Délégations pletes-formes téléphoniques
Fréquentation	4 522 254	
Appels traités	4 730 327	1 050 124
Nombre d'opérations réalisées lors des fréquentations	10 697 048	1 295 599
NATURES DES OPÉRATIONS		
Nombre de devis	4706406	486 618
Réalisations commerciales	1 156 664	128 394
Nombre d'opérations sinistres	228 649	

source : extrait - thèse professionnelle de Me Naima MOKHEDDES – 2016.

10.3. Benchmark des organisations Data Sciences

Date	12.06.2017
Auteur	Didier LEPICAUT
Email	Didier.le_picaut@insa_rouen.fr
Période	Contrat de pro – période 4
Unité Enseignement	MATMUT / SCD
Objet	Benchmark des organisations Data Sciences secteur de la Bancassurance
Version	2.1
Date Version	19.06.2017

Problème : cartographie des bonnes pratiques Data Sciences et des leviers d'action pour MATMUT/SCD « DataLab »

****

PLAN D'ETUDE

1. Résumé 1
2. Présentation de la méthode 1
3. Quelles compétences mobilisées ? [QUI] 2
4. Quel environnement de développement et de production ? [QUOI] 2
5. Où localiser les Data Scientists ? [OU] 2
6. Quel calendrier de déploiement ? [QUAND] 2
7. Quel déploiement ? [COMMENT] 2
8. Quels objectifs ? [POURQUOI] 2
9. Conclusion : pour accélérer et garantir le succès d'un projet Data Sciences à MATMUT/SCD 3
10. Annexes 4

****

1. Résumé

Les objectifs	. identifier les leviers d'action à déployer immédiatement
Les méthodes	. observer et identifier des pratiques d'excellence opérationnelle du secteur bancassurance
Logiciel / fonction	. Not Applicated (NA)
Code source / data	. 4 bancassureurs + 1 opérateur télécoms
Les résultats obtenus	. liste des impacts MATMUT identifiés

2. Présentation de la méthode

Une brève discussion des méthodes utilisées pour analyser le problème managérial :	
Nota :	. grille de lecture 5W I H : Who I What I where I When I Why & How

Introduction :

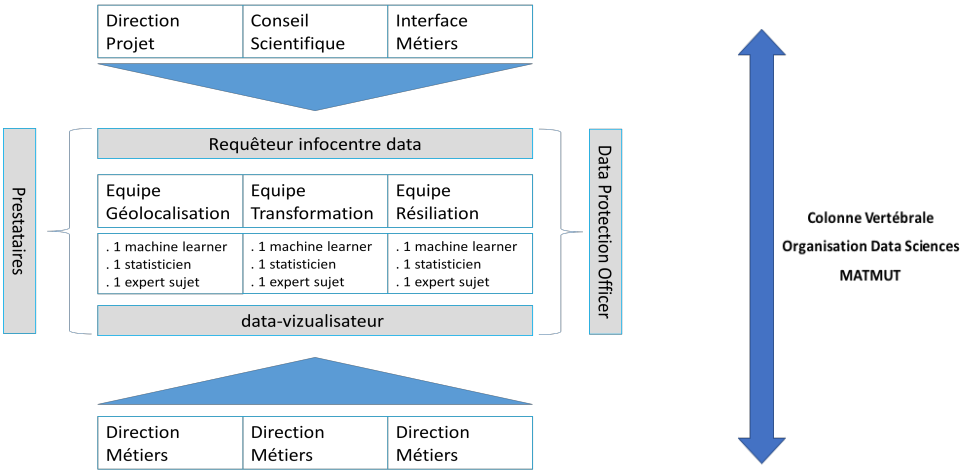
. L'objectif de cette note technique est de réaliser un benchmark des organisations pérennes en Data Sciences auprès d'opérateurs significatifs du secteur de la Bancassurance. Les axes architecture technique, solutions logiciels, organisation, méthode de déploiement, gestion des compétences et facteurs des clefs de réussite seront passés en revue.
. Les meilleures pratiques seront ciblées et traduites en leviers opérationnels pour accélérer le déploiement du DataLab au sein de la MATMUT.

9. Conclusion : pour accélérer et garantir le succès d'un projet Data Sciences à MATMUT/SCD

(i) Immédiatement mettre en place les fonctions et activités suivantes :

Direction Projet	**Conseil Scientifique**	**Interface Métiers**
. coordonne tous les producteurs et toutes les parties prenantes	. enseignant chercheur en Machine Learning / mathématiques décisionnelles	. « commercial grand compte » qui gère les Directions métiers comme des clients internes (i.e. Generali France)
. assure la coordination des chantiers avec les Directions des métiers	. rôle de conseil auprès des Managers SCD sur la mise en production des chantiers du DataLab	. contractualise la relation entre SCD et les métiers avec une proposition technique et financière (i.e. Groupe AXA)
. rédige la roadmap stratégique des chantiers à produire	. rédige les notes de cadrage méthodologique des sujets	. pilote la production des contrats et produit les indicateurs de suivi
. coordonne les partenariats avec les labos & académiques	. supporte la Direction SCD à prononcer les Go / No Go des sujets Data Sciences	. pilote la communication projet avec toutes les parties prenantes
. gère en mode « agence intérim » le recrutement et la mise à disposition de stagiaires écoles /alternants pour garantir le niveau d'activité des chantiers DataLab	. dans le cadre des revues d'avancement chantier (sem.) pour garantir la livraison des jalons, valide les solutions techniques et statistiques **=> avec la méthode kaggle.com, il teste la performance de chaque modèle en validation croisée sur 1 échantillon test du Dataset initial : un « Go » de validation pourra être prononcé.**	
	. supporte Direction SCD et Owners métiers : audite les livraisons de chaque Sprint et assure le suivi des méthodes correctives nécessaires l'avancement des jalons de chaque chantier DataLab	

(ii) Initialiser le schéma organisationnel en phase « DataLab » du déploiement projet « Data Sciences » :



Data Protection Officer : au sein de SCD, le DPO doit décliner « les droits et devoirs de l'employeur Matmut » en termes d'application des textes de loi français et européens sur la protection des données personnelles auprès de toutes « la parties prenantes » au DataLab (permanents Matmut, stagiaires écoles et prestataires de services). Il pourra décliner opérationnellement l'activité de « record management » en organisant le **« clean out day »** après chaque fin Sprint.

10.4. Synoptiques Stéphane TUFFERY

10.4.1. Tableau des méthodes descriptives

Tableau des méthodes descriptives

type	famille	sous-famille	méthode
méthodes descriptives	modèles géométriques	analyse factorielle (projection sur un espace de dimension inférieure)	analyse en composantes principales ACP (variables continues)
			analyse factorielle des correspondances AFC (2 variables qualitatives) analyse des correspondances multiples ACM (+ de 2 var. qualitatives)
		analyse typologique (regroupement en classes homogènes)	méthodes de partitionnement (centres mobiles, *k*-means, nuées dynamiques)
			méthodes hiérarchiques (ascendantes, descendantes)
		analyse typologique + réduction dimens.	classification neuronale (cartes de Kohonen)
	modèles combinatoires		classification relationnelle (variables qualitatives)
	modèles à base de règles logiques	détection de liens	détection d'associations

En grisé : méthodes « classiques »

Tableau des méthodes prédictives

type	famille	sous-famille	méthode
méthodes prédictives	modèles à base de règles logiques	arbres de décision	arbres de décision (variable à expliquer continue ou qualitative)
	modèles à base de fonctions mathématiques	réseaux de neurones	réseaux à apprentissage supervisé : perceptron multicouches, réseau à fonction radiale de base
		modèles paramétriques ou semi-paramétriques	régression linéaire, ANOVA, MANOVA, ANCOVA, MANCOVA, modèle linéaire général GLM, régression PLS, ridge, lasso, SVR (variable à expliquer continue)
			analyse discriminante linéaire, régression logistique, régression logistique PLS, SVM (variable à expliquer qualitative)
			modèle log-linéaire, régression de Poisson (variable à expliquer discrète = comptage)
			modèle linéaire généralisé, modèle additif généralisé (variable à expliquer continue, discrète ou qualitative)
	prédiction sans modèle		*k*-plus proches voisins (*k*-NN)

En grisé : méthodes « classiques »

Source : Data Mining et Big Data, Stephani TUFFERY, 11 décembre 2015, pages 10 & 11

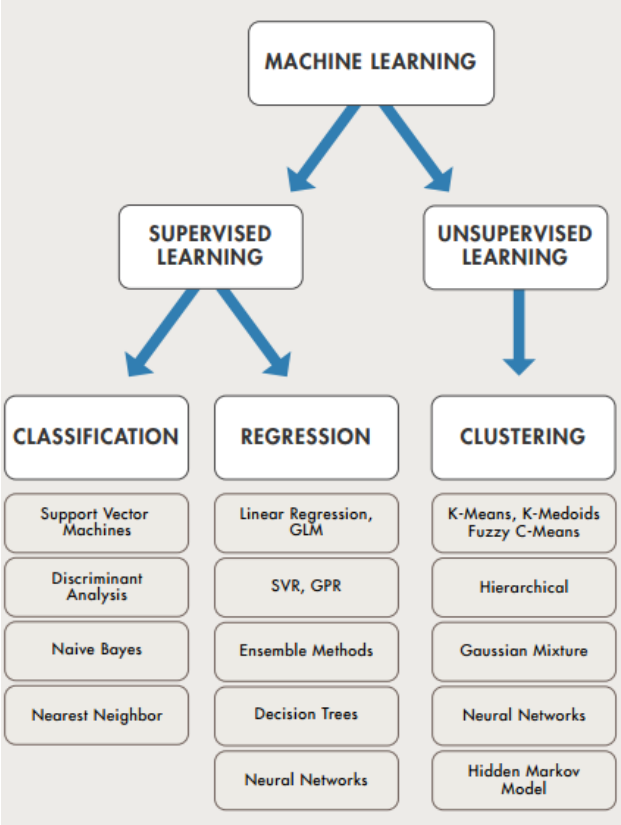
10.4.2. Zoom sur les méthodes prédictives

Zoom sur les méthodes prédictives

explicatives ➔ **🡻 à expliquer**	**1 quantitative (covariable)**	**n quantitatives (covariables)**	**1 qualitative (facteur)**	**n qualitatives (facteurs)**	**mélange**
1 quantitative	rég. linéaire simple, régression robuste, arbres de décision	rég. linéaire multiple, rég. robuste, PLS, arbres, réseaux de neurones	ANOVA, arbres de décision	ANOVA, arbres de décision, réseaux de neurones	ANCOVA, arbres de décision, réseaux de neurones
***n* quantitatives (représentent des quantités ⇗)**	régression PLS2	régression PLS2, réseaux de neurones	MANOVA	MANOVA, réseaux de neurones	MANCOVA, réseaux de neurones
1 qualitative nominale ou binaire	ADL, régression logistique, arbres de décision	ADL, rég. logistique, reg. logistique PLS, arbres, réseaux de neurones, SVM	régression logistique, DISQUAL, arbres	régression logistique, DISQUAL, arbres, réseaux de neurones	régression logistique, arbres, réseaux de neurones
1 discrète (comptage)	modèle linéaire généralisé (régression de Poisson, modèle log-linéaire)				
1 quantitative asymétrique	modèle linéaire généralisé (régressions gamma et log-normale)				
1 qualitative ordinale	régression logistique ordinale (au moins 3 niveaux)				
***n* quantitatives ou qualitatives**	modèle à mesures répétées (les n variables représentent des mesures répétées d'une même quantité)				

Source : Data Mining et Big Data, Stephani TUFFERY, 11 décembre 2015, pages 10 & 11

10.4.3. Machine Learning summarized in one picture



source : http://www.datasciencecentral.com/profiles/blogs/machine-learning-summarized-in-one-picture

10.5. SVM

10.5.1. Théorème de Tom COVER :

Cover's theorem

From Wikipedia, the free encyclopedia

Cover's Theorem is a statement in computational learning theory and is one of the primary theoretical motivations for the use of non-linear kernel methods in machine learning applications. The theorem states that given a set of training data that is not linearly separable, one can with high probability transform it into a training set that is linearly separable by projecting it into a higher-dimensional space via some non-linear transformation. The theorem is named after the information theorist Thomas M. Cover who stated it in 1965.

The proof is easy. A deterministic mapping may be used. Indeed, suppose there are n samples. Lift them onto the vertices of the simplex in the $n-1$ dimensional real space. Every partition of the samples into two sets is separable by a linear separator. QED.

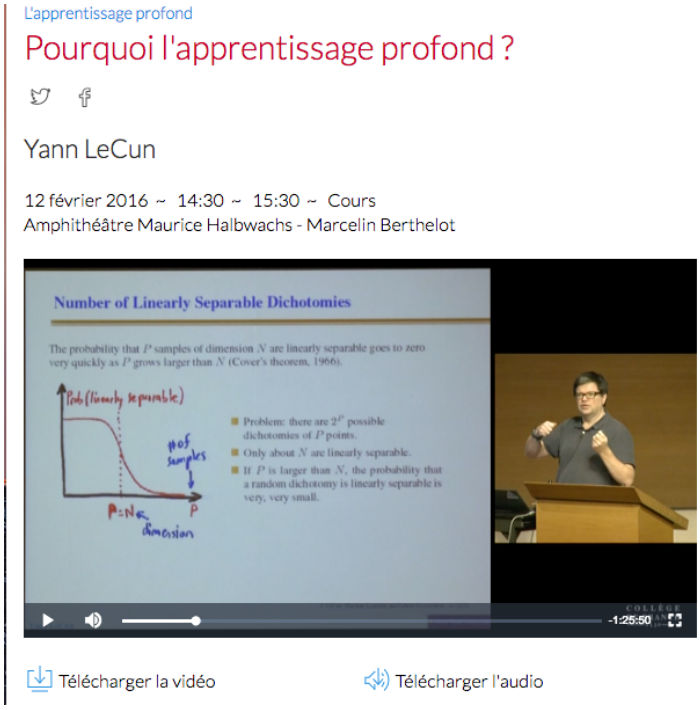
> A complex pattern-classification problem, cast in a high-dimensional space nonlinearly, is more likely to be linearly separable than in a low-dimensional space, provided that the space is not densely populated.
>
> — Cover, T.M., Geometrical and Statistical properties of systems of linear inequalities with applications in pattern recognition, 1965

References [edit]

- Haykin, Simon (2009). *Neural Networks and Learning Machines* (Third ed.). Upper Saddle River, New Jersey: Pearson Education Inc. pp. 232–236. ISBN 978-0-13-147139-9.
- Cover, T.M. (1965). "Geometrical and Statistical properties of systems of linear inequalities with applications in pattern recognition". *IEEE Transactions on Electronic Computers*. EC-14: 326–334. doi:10.1109/pgec.1965.264137.
- Mehrotra, K.; Mohan, C. K.; Ranka, S. (1997). *Elements of artificial neural networks* (2nd ed.). MIT Press. ISBN 0-262-13328-8. (Section 3.5)

source : https://en.wikipedia.org/wiki/Cover%27s_theorem

➔ Explications du théorème « comment rendre un problème linéairement séparable : minute 15e :



source : https://www.college-de-france.fr/site/yann-lecun/course-2016-02-12-14h30.htm

10.5.2. Résumé article : A Practical Guide to Support Vector Classification

A Practical Guide to Support Vector Classification		
https://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf		
page	objet	detail
1	introduction	SVM utile pour la classification de données
1	introduction	dataset - découpé training set (avec label) et test set (sans label)
2	introduction	C > 0 : parametre du terme d'erreur
2	introduction	K(xi, xj) : fonction noyau > 4 noyaux base = lineaire, polynomial, gaussien RBF, sigmoid
2	introduction	4 noyaux base = lineaire, polynomial, gaussien RBF, logistique sigmoid
3	best pratice	1.2: Proposed Procedure - pas de transfo des datas, utiliser des noyaux non lineaires
4	best pratice	1.2: dataset training - Cross Validation (CV) pour trouver parametres C et Gamma puis Test
3	Data Preprocessing	2.1: variable catégorielle (quali) - transformer en var Indicatrice (0,1)
4	Scaling	2.2: centrage reduction - reduction avant d'appliquer 1 noyau SVM
4	Scaling	2.2: centrage reduction - identique pours datasets apprentissage et test
4	sélection model - rbf	3.1 : commencer par le noyau gaussien (RBF Kernel) avec 2 parametres : (C,Gamma)
5	sélection model - rbf	3.1 : qd bcp de variables, préférer un noyau lineaire (linear kernel)
5	sm - CV and Grid-search	3.2: faire 1 CV pour éviter l'over-fitting, utiliser 1 première grille (coarse grid first) + grille affinage
8	discussion	4. : avec bcp de features, faire de la selection de variable avant d'appliquer SVM
9	discussion	4. : LIBSVM - procédure auto de centrage et de sélection de modèle (code par défaut)
12	scaling training - test ?	B : Using the same scaling factors for training and testing sets, we obtain much better accuracy.
		accuracy performance se mesure uniquement sur dataset TEST (et non TRAINING)
13	linear vs RBF kernel	C1: Nb of instances ≪ nb of features => **CV accuracy linear kernel = RBF Kernel**
14	linear vs RBF kernel	C2: instances & features - larges = LIBLINEAR-svm (avec RegLog), plus rapide que LIBSVM
14	linear vs RBF kernel	C3: Nb of instances ≫ nb of features => linear kernel
		ou LIBLINEAR with the option -s 2
		Nb features = petit --> LIBLINEAR with the option -s1
		Nb instances très supérieur ≫ nb features = LIBLINEAR with the option -s 1 // et option -s 2
		https://en.wikipedia.org/wiki/LIBSVM
		http://www.csie.ntu.edu.tw/~cjlin/liblinear/
		https://www.csie.ntu.edu.tw/~cjlin/libsvm/
		définition : instance = ligne ; feature = colonne
		définition : coarse grid = "grille grossière = grille large"

source : http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf -

10.5.3. Définition des paramètres internes du modèle SVM

Exemple : avec le package R – e1071

SVM

* SVM : Support Vector Machines. Ce sont des technologies qui permettent un apprentissage en limitant au maximum les risques d'over-fitting (sur des bases théoriques). On utilisera ici la librairie e1071 (faire `library(e1071)`, la librairie devant être installée avant) qui s'appuie sur libsvm.

* Phase d'apprentissage :

- soit avec une forme traditionnelle, en fournissant les variables prédictives, puis la variable à prédire : `model <- svm(subset(iris, select = c("Sepal.Length", "Sepal.Width", "Petal.Length", "Petal.Width")), iris$Species)` (cf le data frame "iris" qui peut être obtenu en faisant `data(iris)`.
- soit avec une formule : `model <- svm(Species ~ ., iris)` (le '.' indique que toutes les variables autres que Species doivent être utilisées).

Par défaut, le SVM utilisé est de type à noyau radial et les valeurs sont normalisées avant de faire tourner le SVM (pour avoir une moyenne 0 et un écart-type 1). L'objet retourné est de classe svm.

* Paramètres du SVM :
`model <- svm(Species ~ ., iris, type = "C-classification", kernel = "polynomial", degree = 2, gamma = 2, cost = 0.5, coef0 = 0)`

- type : C-classification si on fait de la classification simple (défaut si la variable prédite est de type factor), eps-regression si on fait de la régression (défaut si la variable prédite est de type numérique).
- kernel : noyau à utiliser : radial (radial basis de type exp(-gamma*|u-v|^2)), linear (u'*v), polynomial: (gamma*u'*v + coef0)^degree, sigmoid: tanh(gamma*u'*v + coef0).
- degree : degré du polynôme (pour kernel polynomial).
- gamma : facteur gamma (inutile pour kernel linéaire).
- coef0 : coefficient pour kernel polynomial ou sigmoïde.
- cost : constante de coût pour le lagrangien.

* Attributs d'un objet de la classe svm :

- model$SV : matrice des vecteurs de support.
- model$nSV : nombre de vecteurs de support.

10.6. Calculs numériques - Modélisations Période 1

10.6.1. Code

CODE R	RESULTATS
Modèle Analyse Discriminante Linéaire	
Modèle Régression Logistique	

quanti_select_dlp_v170208.R

lepicaut

Sun Aug 27 21:52:57 2017

```
############
# compile rapport : quanti_select_dlp_v170208.R - en date du 27 aout17
############

# 0. nettoyer historique
rm(list = ls(all = TRUE)) # make sure previous work is clear
ls()

## character(0)

tic <- Sys.time()

# 1. importing and exploring data

path<-"/Users/lepicaut/Desktop/sos/MATMUT/0_0_matmut/fevrier_2017/00000_code"
#path <- "/Users/lepicaut/Desktop/00000_code"

# 2. set working directory
setwd(path)
dv<- read.csv("devis_trsf_2015.csv", header=TRUE, sep=";")


# chargement des librairies R
library(plyr)
library(MASS)
library(corrplot)
library(lattice)
library(caret)

## Loading required package: ggplot2

library(ROCR)

## Loading required package: gplots

##
## Attaching package: 'gplots'

## The following object is masked from 'package:stats':
##
##     lowess

library(ggplot2)
library(FactoMineR)
library(ade4)

## Warning: package 'ade4' was built under R version 3.4.1

##
## Attaching package: 'ade4'
```

```
## The following object is masked from 'package:FactoMineR':
##
##     reconst

library(colorspace)
library(e1071)
library(tidyr)

## Warning: package 'tidyr' was built under R version 3.4.1

library(rmarkdown)

################################
# PARTIE 1. PREPROCESSING DATA
################################


str(dv)

## 'data.frame':    728674 obs. of  112 variables:
##  $ ID_CLIENT              : int  211978166 211978166 211978166 211978166 211978166 211978166 211978182
211978204 211978204 211978217 ...
##  $ REF_SOC                : num  9.8e+11 9.8e+11 9.8e+11 9.8e+11 9.8e+11 ...
##  $ STE_SOC                : Factor w/ 2 levels "F: AMF ASSURANCES",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ TYP_CLI                : Factor w/ 1 level "P: PROPOSANT": 1 1 1 1 1 1 1 1 1 1 ...
##  $ NUM_CTC                : int  3 3 3 3 3 6 2 2 3 3 ...
##  $ NUM_DEV                : int  1 2 3 4 5 1 1 1 1 1 ...
##  $ STE_CIBLE              : Factor w/ 2 levels "F","M": 2 2 2 2 2 2 2 2 2 2 ...
##  $ DAT_REAL               : Factor w/ 305 levels "01/04/2015","01/06/2015",..: 18 18 18 18 18 60 131 8
8 50 ...
##  $ SIT_REAL               : Factor w/ 600 levels "0101","0103",..: 426 426 426 426 426 416 65 598 598
296 ...
##  $ FLE_MRSQ               : Factor w/ 6 levels "1: PERFORMANCE",..: 2 4 4 3 2 2 2 4 4 1 ...
##  $ CD_GAM_MRSQ            : Factor w/ 2 levels "1 : GAMME REFERENCE",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ AN_FAB                 : int  6 6 6 6 6 6 11 9 9 13 ...
##  $ APP_COM                : Factor w/ 27428 levels "","-","/","//",..: 3471 3471 3471 3471 3471 3471 1
045 4130 4130 4268 ...
##  $ C_AN_TTC               : Factor w/ 5312 levels "0,00","1 000,00",..: 1090 151 151 474 1090 864 4126
171 171 4969 ...
##  $ CD_BFAM                : Factor w/ 4 levels "","B","N","O": 3 3 3 3 3 3 3 3 3 3 ...
##  $ CD_FRAN                : Factor w/ 20 levels "00","C1","D1",..: 3 3 3 3 3 3 3 3 3 3 ...
##  $ CD_REMIS               : Factor w/ 7 levels "","0","2","4",..: 5 5 5 5 5 5 5 5 5 5 ...
##  $ CLAS_DEV               : Factor w/ 19 levels "","A","B","C",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ COEFF                  : int  100 100 100 100 100 100 40 100 100 85 ...
##  $ COM_MRSQ               : Factor w/ 21992 levels ".MONTBELIARD",..: 4592 4592 4592 4592 4592 4592 11
935 16442 16442 5039 ...
##  $ CONT                   : logi  NA NA NA NA NA NA ...
##  $ DAT_EFF                : Factor w/ 402 levels "01/01/2015","01/01/2016",..: 36 36 36 36 36 90 179 2
2 22 77 ...
##  $ D_ECH_S                : Factor w/ 6 levels "01/01/2015","01/01/2016",..: 5 5 5 5 5 5 5 5 5 5 ...
##  $ GAR_COND               : Factor w/ 5 levels "","G1","G2","G8",..: 2 2 2 2 2 2 2 3 3 3 ...
##  $ GAR_DOM                : Factor w/ 5 levels "","DE","DEC1",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ GAR_KN                 : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ GAR_RN                 : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ GAR_SP                 : logi  NA NA NA NA NA NA ...
##  $ GNR_MRSQ               : Factor w/ 13 levels "1: CYCLOMOTEUR",..: 3 3 3 3 3 3 3 3 3 3 ...
##  $ GRP_DEV                : int  61 61 61 61 61 61 61 61 61 61 ...
##  $ LEASING                : Factor w/ 7 levels ": NON RENSEIGN",..: 7 7 7 7 7 7 2 7 7 4 ...
##  $ MCN                    : Factor w/ 10 levels "","0 %","100 %",..: 9 9 9 9 9 9 2 9 9 2 ...
##  $ MT_SMAC                : Factor w/ 3 levels "","22,20","22,52": 1 1 1 1 1 1 1 1 1 1 ...
##  $ NUM_CAT                : Factor w/ 40227 levels "","A053120","A053122",..: 22123 22123 22123 22123
22123 22123 27902 24363 24363 35135 ...
##  $ OPT_F3                 : logi  NA NA NA NA NA NA ...
##  $ POND_FR                : int  170 NA NA 170 170 170 NA NA NA NA ...
##  $ POND_GP                : Factor w/ 28 levels "","000","063",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ PROFIL                 : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ PTT_MRSQ               : int  93390 93390 93390 93390 93390 93390 13013 95200 95200 57530 ...
##  $ REG_FISC               : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ RTS                    : Factor w/ 7 levels "","2: HIVER 2 MOIS",..: 5 5 5 5 5 5 5 5 5 5 ...
##  $ SEGMENT                : Factor w/ 84 levels "","09","11","A",..: 49 49 49 49 49 49 49 49 49 49 ...
##  $ TYP_PERS               : Factor w/ 2 levels "M","P": 2 2 2 2 2 2 2 2 2 2 ...
##  $ USAGE                  : Factor w/ 6 levels "AM: AMBULANCE",..: 3 3 3 3 3 5 3 3 3 3 ...
```

```
##  $ ZON_MRSQ               : Factor w/ 9 levels "0","4","5","6",..: 7 7 7 7 7 7 1 7 7 8 ...
##  $ TYP_CONT               : Factor w/ 4 levels "2R  :CONTRAT MRSQ 2 ROUES",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ ZON_MRSQ_REDU_PTIT_ROUL: int  NA NA NA NA NA NA NA NA NA NA ...
##  $ BL_ORI_NET             : Factor w/ 1 level "N": 1 1 1 1 1 1 1 1 1 1 ...
##  $ CD_LOC                 : logi  NA NA NA NA NA NA ...
##  $ BL_OPT_0KM             : logi  NA NA NA NA NA NA ...
##  $ C_AN_TTC_SANS_OPT      : logi  NA NA NA NA NA NA ...
##  $ MARQ_VEH               : Factor w/ 1245 levels "","?","..",".FENDT",..: 250 250 250 250 250 250 118
3 912 912 912 ...
##  $ MODL_VEH               : Factor w/ 4932 levels "","- 500 KG",..: 1364 1364 1364 1364 1364 1364 3410
1613 1613 1617 ...
##  $ PUIS_FISC              : int  5 5 5 5 5 5 4 4 4 4 ...
##  $ NRJ                    : Factor w/ 8 levels "","D","E","F",..: 3 3 3 3 3 3 3 2 2 2 ...
##  $ GAR_GN                 : Factor w/ 4 levels "","G1","G2","G8": 2 2 2 2 2 2 2 3 3 3 ...
##  $ COEF_REF               : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ TRSF_MRSQ              : Factor w/ 2 levels "N","O": 2 2 2 2 2 2 2 2 2 2 ...
##  $ COEF                   : int  100 100 100 100 100 100 40 100 100 85 ...
##  $ NB_DEVI_ARCH_MRSQ      : int  5 5 5 5 5 1 1 2 2 1 ...
##  $ NB_DEVI_ARCH_RP        : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ NB_DEVI_ARCH_RS        : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ NB_DEVI_ARCH_PNO       : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ NB_DEVI_ARCH_TBNH      : logi  NA NA NA NA NA NA ...
##  $ NB_DEVI_ARCH_HEB       : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ NB_DEVI_ARCH_JEUN      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ NB_DEVI_ARCH_ETUE      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ NB_DEVI_ARCH_ETUD      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ NB_DEVI_ARCH_NDEP      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ NB_DEVI_ARCH_CHAS      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ NB_DEVI_ARCH_SNTI      : logi  NA NA NA NA NA NA ...
##  $ NB_DEVI_ARCH_MAV       : int  0 0 0 0 0 1 1 0 0 0 ...
##  $ NB_DEVI_ARCH_FC        : int  0 0 0 0 0 0 1 2 2 0 ...
##  $ NB_DEVI_ARCH_SCOL      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ NB_DEVI_ARCH_VIE       : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ NB_DEVI_ARCH_VIEE      : logi  NA NA NA NA NA NA ...
##  $ NB_DEVI_ARCH_MVE       : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ NB_DEVI_ARCH_MVG       : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ NB_DEVI_ARCH_DC        : logi  NA NA NA NA NA NA ...
##  $ NB_DEVI_ARCH_PREV      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ NB_DEVI_ARCH_OBSQ      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ NB_DEVI_ARCH_FORF      : logi  NA NA NA NA NA NA ...
##  $ NB_DEVI_ARCH_ASSO      : logi  NA NA NA NA NA NA ...
##  $ NB_DEVI_ARCH_MPRO      : logi  NA NA NA NA NA NA ...
##  $ NB_DEVI_ARCH_PNS       : logi  NA NA NA NA NA NA ...
##  $ NB_DEVI_ARCH_MLEC      : logi  NA NA NA NA NA NA ...
##  $ NB_DEVI_ARCH_CAD       : logi  NA NA NA NA NA NA ...
##  $ NB_DEVI_ARCH_GAP       : logi  NA NA NA NA NA NA ...
##  $ NB_DEVI_ARCH_ETUM      : logi  NA NA NA NA NA NA ...
##  $ NB_DEVI_ARCH_MPAP      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ DAT_1ER_FRQ            : Factor w/ 305 levels "01/04/2015","01/06/2015",..: 18 18 18 18 18 18 131 8
8 40 ...
##  $ DAT_ADH                : Factor w/ 554 levels "","01/02/2016",..: 108 108 108 108 108 108 236 17 17
90 ...
##  $ SEXE                   : Factor w/ 4 levels "","F","M","P": 2 2 2 2 2 2 3 2 2 2 ...
##  $ IDENT                  : Factor w/ 13 levels "","1: CONJOINT",..: 4 4 4 4 4 4 4 4 4 4 ...
##  $ AGE_CLI                : int  25 25 25 25 25 25 47 24 24 24 ...
##  $ CSP                    : Factor w/ 23 levels "","10: AGRICULTEUR",..: 15 15 15 15 15 15 15 15 15 15
...
##  $ PROF                   : Factor w/ 39513 levels "","-","--","?",..: 2274 2274 2274 2274 2274 2274 3
5917 18264 18264 38096 ...
##  $ CD_FONC                : Factor w/ 2 levels "N","O": 1 1 1 1 1 1 1 1 1 1 ...
##  $ CD_ADMIN               : Factor w/ 12 levels "","01: TERRITORIALE",..: 1 1 1 1 1 1 1 1 1 1 ...
##   [list output truncated]
```

```r
names(dv)
```

```
##   [1] "ID_CLIENT"               "REF_SOC"
##   [3] "STE_SOC"                 "TYP_CLI"
##   [5] "NUM_CTC"                 "NUM_DEV"
##   [7] "STE_CIBLE"               "DAT_REAL"
##   [9] "SIT_REAL"                "FLE_MRSQ"
##  [11] "CD_GAM_MRSQ"             "AN_FAB"
##  [13] "APP_COM"                 "C_AN_TTC"
```

```
##  [15] "CD_BFAM"                  "CD_FRAN"
##  [17] "CD_REMIS"                 "CLAS_DEV"
##  [19] "COEFF"                    "COM_MRSQ"
##  [21] "CONT"                     "DAT_EFF"
##  [23] "D_ECH_S"                  "GAR_COND"
##  [25] "GAR_DOM"                  "GAR_KN"
##  [27] "GAR_RN"                   "GAR_SP"
##  [29] "GNR_MRSQ"                 "GRP_DEV"
##  [31] "LEASING"                  "MCN"
##  [33] "MT_SMAC"                  "NUM_CAT"
##  [35] "OPT_F3"                   "POND_FR"
##  [37] "POND_GP"                  "PROFIL"
##  [39] "PTT_MRSQ"                 "REG_FISC"
##  [41] "RTS"                      "SEGMENT"
##  [43] "TYP_PERS"                 "USAGE"
##  [45] "ZON_MRSQ"                 "TYP_CONT"
##  [47] "ZON_MRSQ_REDU_PTIT_ROUL" "BL_ORI_NET"
##  [49] "CD_LOC"                   "BL_OPT_0KM"
##  [51] "C_AN_TTC_SANS_OPT"        "MARQ_VEH"
##  [53] "MODL_VEH"                 "PUIS_FISC"
##  [55] "NRJ"                      "GAR_GN"
##  [57] "COEF_REF"                 "TRSF_MRSQ"
##  [59] "COEF"                     "NB_DEVI_ARCH_MRSQ"
##  [61] "NB_DEVI_ARCH_RP"          "NB_DEVI_ARCH_RS"
##  [63] "NB_DEVI_ARCH_PNO"         "NB_DEVI_ARCH_TBNH"
##  [65] "NB_DEVI_ARCH_HEB"         "NB_DEVI_ARCH_JEUN"
##  [67] "NB_DEVI_ARCH_ETUE"        "NB_DEVI_ARCH_ETUD"
##  [69] "NB_DEVI_ARCH_NDEP"        "NB_DEVI_ARCH_CHAS"
##  [71] "NB_DEVI_ARCH_SNTI"        "NB_DEVI_ARCH_MAV"
##  [73] "NB_DEVI_ARCH_FC"          "NB_DEVI_ARCH_SCOL"
##  [75] "NB_DEVI_ARCH_VIE"         "NB_DEVI_ARCH_VIEE"
##  [77] "NB_DEVI_ARCH_MVE"         "NB_DEVI_ARCH_MVG"
##  [79] "NB_DEVI_ARCH_DC"          "NB_DEVI_ARCH_PREV"
##  [81] "NB_DEVI_ARCH_OBSQ"        "NB_DEVI_ARCH_FORF"
##  [83] "NB_DEVI_ARCH_ASSO"        "NB_DEVI_ARCH_MPRO"
##  [85] "NB_DEVI_ARCH_PNS"         "NB_DEVI_ARCH_MLEC"
##  [87] "NB_DEVI_ARCH_CAD"         "NB_DEVI_ARCH_GAP"
##  [89] "NB_DEVI_ARCH_ETUM"        "NB_DEVI_ARCH_MPAP"
##  [91] "DAT_1ER_FRQ"              "DAT_ADH"
##  [93] "SEXE"                     "IDENT"
##  [95] "AGE_CLI"                  "CSP"
##  [97] "PROF"                     "CD_FONC"
##  [99] "CD_ADMIN"                 "SIT_FAMD"
## [101] "NB_ENF"                   "AGE_E1"
## [103] "AGE_E2"                   "AGE_E3"
## [105] "AGE_E4"                   "AGE_E5"
## [107] "LIB_MOTIV"                "ACT_REAL"
## [109] "SIT_REAL1"                "TYP_REAL"
## [111] "trsf_dev"                 "ANC_VEH"

dv1<-dv[,-3:-5]

dv2<-dv1[,-2]

dv3<-dv2[,-3:-5]

dv4<-dv3[,-4:-5]

dv5<-dv4[,-4]

dv6<-dv5[,-5:-8]

dv7<-dv6[,-7:-14]

dv8<-dv7[,-7:-19]

dv9<-dv8[,-7:-9]

dv10<-dv9[,-8:-9]

dv11<-dv10[,-9:-11]
```

```r
dv12<-dv11[,-14:-15]

names(dv12)

##  [1] "ID_CLIENT"         "NUM_DEV"           "FLE_MRSQ"
##  [4] "C_AN_TTC"          "COEFF"             "COM_MRSQ"
##  [7] "ZON_MRSQ"          "BL_ORI_NET"        "MARQ_VEH"
## [10] "MODL_VEH"          "PUIS_FISC"         "NRJ"
## [13] "GAR_GN"            "COEF"              "NB_DEVI_ARCH_MRSQ"
## [16] "NB_DEVI_ARCH_RP"   "NB_DEVI_ARCH_RS"   "NB_DEVI_ARCH_PNO"
## [19] "NB_DEVI_ARCH_TBNH" "NB_DEVI_ARCH_HEB"  "NB_DEVI_ARCH_JEUN"
## [22] "NB_DEVI_ARCH_ETUE" "NB_DEVI_ARCH_ETUD" "NB_DEVI_ARCH_NDEP"
## [25] "NB_DEVI_ARCH_CHAS" "NB_DEVI_ARCH_SNTI" "NB_DEVI_ARCH_MAV"
## [28] "NB_DEVI_ARCH_FC"   "NB_DEVI_ARCH_SCOL" "NB_DEVI_ARCH_VIE"
## [31] "NB_DEVI_ARCH_VIEE" "NB_DEVI_ARCH_MVE"  "NB_DEVI_ARCH_MVG"
## [34] "NB_DEVI_ARCH_DC"   "NB_DEVI_ARCH_PREV" "NB_DEVI_ARCH_OBSQ"
## [37] "NB_DEVI_ARCH_FORF" "NB_DEVI_ARCH_ASSO" "NB_DEVI_ARCH_MPRO"
## [40] "NB_DEVI_ARCH_PNS"  "NB_DEVI_ARCH_MLEC" "NB_DEVI_ARCH_CAD"
## [43] "NB_DEVI_ARCH_GAP"  "NB_DEVI_ARCH_ETUM" "NB_DEVI_ARCH_MPAP"
## [46] "DAT_1ER_FRQ"       "DAT_ADH"           "SEXE"
## [49] "IDENT"             "AGE_CLI"           "CSP"
## [52] "PROF"              "CD_FONC"           "CD_ADMIN"
## [55] "SIT_FAMD"          "NB_ENF"            "AGE_E1"
## [58] "AGE_E2"            "AGE_E3"            "AGE_E4"
## [61] "AGE_E5"            "LIB_MOTIV"         "ACT_REAL"
## [64] "SIT_REAL1"         "TYP_REAL"          "trsf_dev"
## [67] "ANC_VEH"

which(names(dv12) %in% c('NB_DEVI_ARCH_MRSQ','DAT_ADH'))

## [1] 15 47

dv13<-dv12[,-15:-47]

names(dv13)

##  [1] "ID_CLIENT"  "NUM_DEV"    "FLE_MRSQ"   "C_AN_TTC"   "COEFF"
##  [6] "COM_MRSQ"   "ZON_MRSQ"   "BL_ORI_NET" "MARQ_VEH"   "MODL_VEH"
## [11] "PUIS_FISC"  "NRJ"        "GAR_GN"     "COEF"       "SEXE"
## [16] "IDENT"      "AGE_CLI"    "CSP"        "PROF"       "CD_FONC"
## [21] "CD_ADMIN"   "SIT_FAMD"   "NB_ENF"     "AGE_E1"     "AGE_E2"
## [26] "AGE_E3"     "AGE_E4"     "AGE_E5"     "LIB_MOTIV"  "ACT_REAL"
## [31] "SIT_REAL1"  "TYP_REAL"   "trsf_dev"   "ANC_VEH"

which(names(dv13) %in% c('AGE_E1','TYP_REAL'))

## [1] 24 32

dv14<-dv13[,-24:-32]

which(names(dv14) %in% c('CD_ADMIN'))

## [1] 21

dv15<-dv14[,-21]

which(names(dv15) %in% c('MARQ_VEH'))

## [1] 9

dv16<-dv15[,-9]

summary(dv16)

##    ID_CLIENT            NUM_DEV                          FLE_MRSQ
##  Min.   :211978166   Min.   :  1.000   1: PERFORMANCE        : 58225
##  1st Qu.:212233606   1st Qu.:  1.000   2: EQUILIBRE          :302757
##  Median :212378405   Median :  1.000   4: URBAIN             : 65985
##  Mean   :212395235   Mean   :  2.123   5: TIERS VOL INCENDIE :136241
##  3rd Qu.:212568373   3rd Qu.:  2.000   6: TIERS BRIS DE GLACE: 79589
##  Max.   :212760888   Max.   :318.000   7: TIERS              : 85877
##
##     C_AN_TTC          COEFF               COM_MRSQ         ZON_MRSQ
```

```
##  906,30 :  1654   Min.   : 35.00   MARSEILLE  : 18462   6      :187980
##  442,30 :  1628   1st Qu.: 40.00   PARIS      : 11120   B      :141497
##  821,30 :  1626   Median : 65.00   TOULOUSE   :  6635   5      :141386
##  811,30 :  1554   Mean   : 66.11   NICE       :  6226   C      :122417
##  735,30 :  1393   3rd Qu.: 95.00   MONTPELLIER:  6060   7      : 43090
##  783,30 :  1393   Max.   :288.00   LYON       :  5227   4      : 34267
##  (Other):719426   NA's   :4832     (Other)    :674944   (Other): 58037
##  BL_ORI_NET    MODL_VEH        PUIS_FISC           NRJ         GAR_GN
##  N:728674   CLIO   : 70858   Min.   : 0.000   D      :445711     :  4895
##             206    : 35985   1st Qu.: 4.000   E      :273391   G1:219671
##             TWINGO : 25226   Median : 5.000          :  4191   G2:476918
##             207    : 17272   Mean   : 5.655   H      :  2548   G8: 27190
##             MEGANE : 15593   3rd Qu.: 6.000   G      :  1915
##             FIESTA : 13841   Max.   :85.000   Z      :   707
##             (Other):549899   NA's   :4730     (Other):   211
##       COEF        SEXE                   IDENT           AGE_CLI
##  Min.   : 35.00    :  8161   3: TIERS       :456008   Min.   :  6.00
##  1st Qu.: 40.00   F:303492   2: ENFANT      :163566   1st Qu.: 23.00
##  Median : 65.00   M:406185                  : 62492   Median : 33.00
##  Mean   : 66.11   P: 10836   1: CONJOINT    : 35419   Mean   : 37.51
##  3rd Qu.: 95.00              9: STE BUT LUCR: 10007   3rd Qu.: 50.00
##  Max.   :288.00              7: ASS LOI 1901:   500   Max.   :168.00
##  NA's   :4832                (Other)        :   682   NA's   :29135
##                CSP                PROF        CD_FONC
##  54: EMPLOYE PRIV:213534             : 80339   N:544886
##  51: EMPLOYE PUBL:139916   ETUDIANT  : 24702   O:183788
##  83: ETUDIANT    :117461   RETRAITE  : 23010
##  81: SANS EMPLOI : 49020   SANS      : 21364
##  70: RETRAITE    : 42956   ETUDIANTE : 21158
##  60: OUVRIER     : 35489   INFIRMIERE: 11813
##  (Other)         :130298   (Other)   :546288
##            SIT_FAMD          NB_ENF      trsf_dev      ANC_VEH
##  A             :197020   Min.   : 0.00   N:547668   Min.   :-1.000
##  M             :176331   1st Qu.: 0.00   O:181006   1st Qu.: 5.000
##  2: CELIBATAIRE:127349   Median : 0.00              Median : 9.000
##  1: MARIE      :124116   Mean   : 0.89              Mean   : 9.403
##                : 50419   3rd Qu.: 2.00              3rd Qu.:14.000
##  4: CONCUBIN   : 33546   Max.   :20.00              Max.   :60.000
##  (Other)       : 19893   NA's   :99983
```

```r
dim(dv16)
```

```
## [1] 728674     23
```

```r
attach(dv16)

dv17=na.omit(dv16)

summary(dv17)
```

```
##    ID_CLIENT            NUM_DEV                          FLE_MRSQ
##  Min.   :211978166   Min.   :  1.000   1: PERFORMANCE        : 48653
##  1st Qu.:212234208   1st Qu.:  1.000   2: EQUILIBRE          :250854
##  Median :212379122   Median :  2.000   4: URBAIN             : 57730
##  Mean   :212395481   Mean   :  2.145   5: TIERS VOL INCENDIE :116130
##  3rd Qu.:212567554   3rd Qu.:  2.000   6: TIERS BRIS DE GLACE: 69059
##  Max.   :212760888   Max.   :318.000   7: TIERS              : 71622
##
##     C_AN_TTC          COEFF               COM_MRSQ         ZON_MRSQ
##  906,30 :  1466   Min.   : 35.00   MARSEILLE  : 16618   6      :160490
##  821,30 :  1419   1st Qu.: 40.00   PARIS      :  8479   B      :120252
##  442,30 :  1394   Median : 65.00   TOULOUSE   :  5838   5      :116279
##  811,30 :  1370   Mean   : 66.27   NICE       :  4853   C      :103447
##  735,30 :  1227   3rd Qu.: 95.00   MONTPELLIER:  4448   7      : 36494
##  783,30 :  1204   Max.   :256.00   LYON       :  4404   4      : 28014
##  (Other):605968                    (Other)    :569408   (Other): 49072
##  BL_ORI_NET    MODL_VEH        PUIS_FISC           NRJ         GAR_GN
##  N:614048   CLIO   : 62674   Min.   : 0.000   D      :373804     :   464
##             206    : 31719   1st Qu.: 4.000   E      :236070   G1:183649
##             TWINGO : 22149   Median : 5.000   H      :  1881   G2:406620
##             207    : 15065   Mean   : 5.577   G      :  1669   G8: 23315
##             MEGANE : 13592   3rd Qu.: 6.000   Z      :   430
```

```
##             FIESTA : 12259   Max.   :68.000   F      :   149
##             (Other):456590                    (Other):    45
##       COEF        SEXE                  IDENT           AGE_CLI
##  Min.   : 35.00    :   825   3: TIERS       :395372   Min.   : 13.00
##  1st Qu.: 40.00   F:267300   2: ENFANT      :150940   1st Qu.: 23.00
##  Median : 65.00   M:345805                  : 35457   Median : 32.00
##  Mean   : 66.27   P:   118   1: CONJOINT    : 31841   Mean   : 36.88
##  3rd Qu.: 95.00              4: EMPL. MATMUT:   160   3rd Qu.: 49.00
##  Max.   :256.00              5: EMPL.MUTPUBL:   143   Max.   :168.00
##                              (Other)        :   135
##                CSP                 PROF        CD_FONC
##  54: EMPLOYE PRIV:186492             : 49331   N:447441
##  51: EMPLOYE PUBL:128585   ETUDIANT  : 21424   O:166607
##  83: ETUDIANT    :103179   SANS      : 18646
##  81: SANS EMPLOI : 42967   ETUDIANTE : 18462
##  60: OUVRIER     : 30626   RETRAITE  : 16567
##  70: RETRAITE    : 29404   INFIRMIERE: 10779
##  (Other)         : 92795   (Other)   :478839
##             SIT_FAMD          NB_ENF        trsf_dev      ANC_VEH
##  A             :181363   Min.   : 0.0000   N:450163   Min.   :-1.000
##  M             :165438   1st Qu.: 0.0000   O:163885   1st Qu.: 5.000
##  2: CELIBATAIRE:107143   Median : 0.0000              Median : 9.000
##  1: MARIE      : 98980   Mean   : 0.8838              Mean   : 9.495
##  4: CONCUBIN   : 28444   3rd Qu.: 2.0000              3rd Qu.:14.000
##                : 16567   Max.   :20.0000              Max.   :60.000
##  (Other)       : 16113
```

```r
(table(dv17$trsf_dev)/nrow(dv17)*100)
```

```
##
##        N        O
## 73.31072 26.68928
```

```r
dim(dv17)
```

```
## [1] 614048     23
```

```r
sum(is.na(dv17))
```

```
## [1] 0
```

```r
str(dv17)
```

```
## 'data.frame':    614048 obs. of  23 variables:
##  $ ID_CLIENT : int  211978166 211978166 211978166 211978166 211978166 211978166 211978182 211978204 21
1978204 211978226 ...
##  $ NUM_DEV   : int  1 2 3 4 5 1 1 1 1 1 ...
##  $ FLE_MRSQ  : Factor w/ 6 levels "1: PERFORMANCE",..: 2 4 4 3 2 2 2 4 4 4 ...
##  $ C_AN_TTC  : Factor w/ 5312 levels "0,00","1 000,00",..: 1090 151 151 474 1090 864 4126 171 171 4778
...
##  $ COEFF     : int  100 100 100 100 100 100 40 100 100 95 ...
##  $ COM_MRSQ  : Factor w/ 21992 levels ".MONTBELIARD",..: 4592 4592 4592 4592 4592 4592 11935 16442 164
42 11935 ...
##  $ ZON_MRSQ  : Factor w/ 9 levels "0","4","5","6",..: 7 7 7 7 7 7 1 7 7 1 ...
##  $ BL_ORI_NET: Factor w/ 1 level "N": 1 1 1 1 1 1 1 1 1 1 ...
##  $ MODL_VEH  : Factor w/ 4932 levels "","- 500 KG",..: 1364 1364 1364 1364 1364 1364 3410 1613 1613 38
66 ...
##  $ PUIS_FISC : int  5 5 5 5 5 5 4 4 4 1 ...
##  $ NRJ       : Factor w/ 8 levels "","D","E","F",..: 3 3 3 3 3 3 3 2 2 3 ...
##  $ GAR_GN    : Factor w/ 4 levels "","G1","G2","G8": 2 2 2 2 2 2 2 3 3 3 ...
##  $ COEF      : int  100 100 100 100 100 100 40 100 100 95 ...
##  $ SEXE      : Factor w/ 4 levels "","F","M","P": 2 2 2 2 2 2 3 2 2 3 ...
##  $ IDENT     : Factor w/ 13 levels "","1: CONJOINT",..: 4 4 4 4 4 4 4 4 4 1 ...
##  $ AGE_CLI   : int  25 25 25 25 25 25 47 24 24 23 ...
##  $ CSP       : Factor w/ 23 levels "","10: AGRICULTEUR",..: 15 15 15 15 15 15 15 15 15 21 ...
##  $ PROF      : Factor w/ 39513 levels "","-","--","?",..: 2274 2274 2274 2274 2274 2274 35917 18264 18
264 8719 ...
##  $ CD_FONC   : Factor w/ 2 levels "N","O": 1 1 1 1 1 1 1 1 1 1 ...
##  $ SIT_FAMD  : Factor w/ 8 levels "","1: MARIE",..: 7 7 7 7 7 7 7 7 7 1 ...
##  $ NB_ENF    : int  0 0 0 0 0 0 2 0 0 0 ...
##  $ trsf_dev  : Factor w/ 2 levels "N","O": 1 1 1 1 1 2 2 2 2 1 ...
##  $ ANC_VEH   : int  9 9 9 9 9 9 4 6 6 6 ...
```

```
##  - attr(*, "na.action")=Class 'omit'  Named int [1:114626] 10 36 42 43 59 60 75 76 77 78 ...
##   .. ..- attr(*, "names")= chr [1:114626] "10" "36" "42" "43" ...
```

```r
# passage de > 600 000 lignes à 1000 lignes
dv18=dv17[sample(1:nrow(dv17),614048),]

# passage de > 614 048 lignes à 6140 lignes
# cad poids échantillon = 0.99% x échantillon total observé
#dv18=dv17[sample(1:nrow(dv17),6140),]

dim(dv18)
```

```
## [1] 614048     23
```

```r
(table(dv18$trsf_dev)/nrow(dv18)*100)
```

```
##
##        N        O
## 73.31072 26.68928
```

```r
summary(dv18)
```

```
##    ID_CLIENT            NUM_DEV                          FLE_MRSQ
##  Min.   :211978166   Min.   :  1.000   1: PERFORMANCE        : 48653
##  1st Qu.:212234208   1st Qu.:  1.000   2: EQUILIBRE          :250854
##  Median :212379122   Median :  2.000   4: URBAIN             : 57730
##  Mean   :212395481   Mean   :  2.145   5: TIERS VOL INCENDIE :116130
##  3rd Qu.:212567554   3rd Qu.:  2.000   6: TIERS BRIS DE GLACE: 69059
##  Max.   :212760888   Max.   :318.000   7: TIERS              : 71622
##
##     C_AN_TTC          COEFF               COM_MRSQ         ZON_MRSQ
##  906,30 :  1466   Min.   : 35.00   MARSEILLE  : 16618   6      :160490
##  821,30 :  1419   1st Qu.: 40.00   PARIS      :  8479   B      :120252
##  442,30 :  1394   Median : 65.00   TOULOUSE   :  5838   5      :116279
##  811,30 :  1370   Mean   : 66.27   NICE       :  4853   C      :103447
##  735,30 :  1227   3rd Qu.: 95.00   MONTPELLIER:  4448   7      : 36494
##  783,30 :  1204   Max.   :256.00   LYON       :  4404   4      : 28014
##  (Other):605968                    (Other)    :569408   (Other): 49072
##  BL_ORI_NET    MODL_VEH        PUIS_FISC           NRJ         GAR_GN
##  N:614048   CLIO    : 62674   Min.   : 0.000   D      :373804     :   464
##             206     : 31719   1st Qu.: 4.000   E      :236070   G1:183649
##             TWINGO  : 22149   Median : 5.000   H      :  1881   G2:406620
##             207     : 15065   Mean   : 5.577   G      :  1669   G8: 23315
##             MEGANE  : 13592   3rd Qu.: 6.000   Z      :   430
##             FIESTA  : 12259   Max.   :68.000   F      :   149
##             (Other):456590                    (Other):    45
##       COEF        SEXE                   IDENT           AGE_CLI
##  Min.   : 35.00    :   825   3: TIERS       :395372   Min.   : 13.00
##  1st Qu.: 40.00   F:267300   2: ENFANT      :150940   1st Qu.: 23.00
##  Median : 65.00   M:345805                  : 35457   Median : 32.00
##  Mean   : 66.27   P:   118   1: CONJOINT    : 31841   Mean   : 36.88
##  3rd Qu.: 95.00              4: EMPL. MATMUT:   160   3rd Qu.: 49.00
##  Max.   :256.00              5: EMPL.MUTPUBL:   143   Max.   :168.00
##                              (Other)        :   135
##                CSP                 PROF        CD_FONC
##  54: EMPLOYE PRIV:186492             : 49331   N:447441
##  51: EMPLOYE PUBL:128585   ETUDIANT  : 21424   O:166607
##  83: ETUDIANT    :103179   SANS      : 18646
##  81: SANS EMPLOI : 42967   ETUDIANTE : 18462
##  60: OUVRIER     : 30626   RETRAITE  : 16567
##  70: RETRAITE    : 29404   INFIRMIERE: 10779
##  (Other)         : 92795   (Other)   :478839
##            SIT_FAMD          NB_ENF        trsf_dev      ANC_VEH
##  A             :181363   Min.   : 0.0000   N:450163   Min.   :-1.000
##  M             :165438   1st Qu.: 0.0000   O:163885   1st Qu.: 5.000
##  2: CELIBATAIRE:107143   Median : 0.0000              Median : 9.000
##  1: MARIE      : 98980   Mean   : 0.8838              Mean   : 9.495
##  4: CONCUBIN   : 28444   3rd Qu.: 2.0000              3rd Qu.:14.000
##                : 16567   Max.   :20.0000              Max.   :60.000
##  (Other)       : 16113
```

```r
str(dv18)
```

```
## 'data.frame':    614048 obs. of  23 variables:
##  $ ID_CLIENT : int  212753725 212232107 212547161 212507824 212167454 212679778 212135634 212663825 21
2575321 212749153 ...
##  $ NUM_DEV   : int  1 1 1 2 2 4 2 1 3 1 ...
##  $ FLE_MRSQ  : Factor w/ 6 levels "1: PERFORMANCE",..: 5 2 4 2 3 3 4 4 2 1 ...
##  $ C_AN_TTC  : Factor w/ 5312 levels "0,00","1 000,00",..: 878 4030 3607 3665 4726 4300 3657 171 4370
4538 ...
##  $ COEFF     : int  185 40 65 35 57 65 50 100 65 40 ...
##  $ COM_MRSQ  : Factor w/ 21992 levels ".MONTBELIARD",..: 21660 21394 15294 21600 3794 5622 20422 14319
4568 5034 ...
##  $ ZON_MRSQ  : Factor w/ 9 levels "0","4","5","6",..: 2 7 4 8 4 4 8 4 4 7 ...
##  $ BL_ORI_NET: Factor w/ 1 level "N": 1 1 1 1 1 1 1 1 1 1 ...
##  $ MODL_VEH  : Factor w/ 4932 levels "","- 500 KG",..: 1613 3409 4222 3885 278 1613 4780 248 168 2596
...
##  $ PUIS_FISC : int  4 4 6 7 8 5 6 5 4 5 ...
##  $ NRJ       : Factor w/ 8 levels "","D","E","F",..: 2 2 3 3 2 3 2 2 2 2 ...
##  $ GAR_GN    : Factor w/ 4 levels "","G1","G2","G8": 3 2 3 2 3 3 3 3 2 3 ...
##  $ COEF      : int  185 40 65 35 57 65 50 100 65 40 ...
##  $ SEXE      : Factor w/ 4 levels "","F","M","P": 3 3 3 2 3 2 3 3 3 3 ...
##  $ IDENT     : Factor w/ 13 levels "","1: CONJOINT",..: 4 1 3 2 1 3 1 4 3 4 ...
##  $ AGE_CLI   : int  24 60 21 48 39 19 60 18 19 52 ...
##  $ CSP       : Factor w/ 23 levels "","10: AGRICULTEUR",..: 19 16 21 15 19 21 20 21 21 13 ...
##  $ PROF      : Factor w/ 39513 levels "","-","--","?",..: 22030 26547 21586 10396 35045 23402 35023 20
829 20829 24379 ...
##  $ CD_FONC   : Factor w/ 2 levels "N","O": 1 1 1 1 1 1 1 1 1 1 ...
##  $ SIT_FAMD  : Factor w/ 8 levels "","1: MARIE",..: 3 1 7 8 2 7 2 8 7 2 ...
##  $ NB_ENF    : int  0 3 0 4 2 0 0 0 0 4 ...
##  $ trsf_dev  : Factor w/ 2 levels "N","O": 1 1 2 1 1 1 1 1 2 1 ...
##  $ ANC_VEH   : int  8 7 19 12 10 7 15 14 5 0 ...
##  - attr(*, "na.action")=Class 'omit'  Named int [1:114626] 10 36 42 43 59 60 75 76 77 78 ...
##   .. ..- attr(*, "names")= chr [1:114626] "10" "36" "42" "43" ...
```

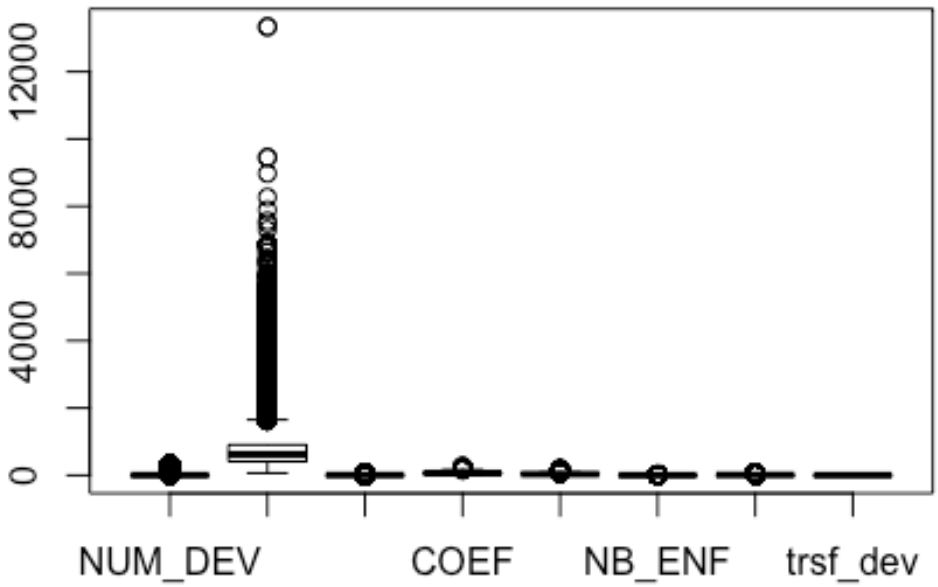
```
# transfrome C_AN_TTC en var numerique
dv18$C_AN_TTC = gsub(",",".",dv18$C_AN_TTC)
dv18$C_AN_TTC = gsub(" ","",dv18$C_AN_TTC)
dv18$C_AN_TTC = as.double(dv18$C_AN_TTC)

str(dv18)
```

```
## 'data.frame':    614048 obs. of  23 variables:
##  $ ID_CLIENT : int  212753725 212232107 212547161 212507824 212167454 212679778 212135634 212663825 21
2575321 212749153 ...
##  $ NUM_DEV   : int  1 1 1 2 2 4 2 1 3 1 ...
##  $ FLE_MRSQ  : Factor w/ 6 levels "1: PERFORMANCE",..: 5 2 4 2 3 3 4 4 2 1 ...
##  $ C_AN_TTC  : num  1479 443 344 373 720 ...
##  $ COEFF     : int  185 40 65 35 57 65 50 100 65 40 ...
##  $ COM_MRSQ  : Factor w/ 21992 levels ".MONTBELIARD",..: 21660 21394 15294 21600 3794 5622 20422 14319
4568 5034 ...
##  $ ZON_MRSQ  : Factor w/ 9 levels "0","4","5","6",..: 2 7 4 8 4 4 8 4 4 7 ...
##  $ BL_ORI_NET: Factor w/ 1 level "N": 1 1 1 1 1 1 1 1 1 1 ...
##  $ MODL_VEH  : Factor w/ 4932 levels "","- 500 KG",..: 1613 3409 4222 3885 278 1613 4780 248 168 2596
...
##  $ PUIS_FISC : int  4 4 6 7 8 5 6 5 4 5 ...
##  $ NRJ       : Factor w/ 8 levels "","D","E","F",..: 2 2 3 3 2 3 2 2 2 2 ...
##  $ GAR_GN    : Factor w/ 4 levels "","G1","G2","G8": 3 2 3 2 3 3 3 3 2 3 ...
##  $ COEF      : int  185 40 65 35 57 65 50 100 65 40 ...
##  $ SEXE      : Factor w/ 4 levels "","F","M","P": 3 3 3 2 3 2 3 3 3 3 ...
##  $ IDENT     : Factor w/ 13 levels "","1: CONJOINT",..: 4 1 3 2 1 3 1 4 3 4 ...
##  $ AGE_CLI   : int  24 60 21 48 39 19 60 18 19 52 ...
##  $ CSP       : Factor w/ 23 levels "","10: AGRICULTEUR",..: 19 16 21 15 19 21 20 21 21 13 ...
##  $ PROF      : Factor w/ 39513 levels "","-","--","?",..: 22030 26547 21586 10396 35045 23402 35023 20
829 20829 24379 ...
##  $ CD_FONC   : Factor w/ 2 levels "N","O": 1 1 1 1 1 1 1 1 1 1 ...
##  $ SIT_FAMD  : Factor w/ 8 levels "","1: MARIE",..: 3 1 7 8 2 7 2 8 7 2 ...
##  $ NB_ENF    : int  0 3 0 4 2 0 0 0 0 4 ...
##  $ trsf_dev  : Factor w/ 2 levels "N","O": 1 1 2 1 1 1 1 1 2 1 ...
##  $ ANC_VEH   : int  8 7 19 12 10 7 15 14 5 0 ...
##  - attr(*, "na.action")=Class 'omit'  Named int [1:114626] 10 36 42 43 59 60 75 76 77 78 ...
##   .. ..- attr(*, "names")= chr [1:114626] "10" "36" "42" "43" ...
```

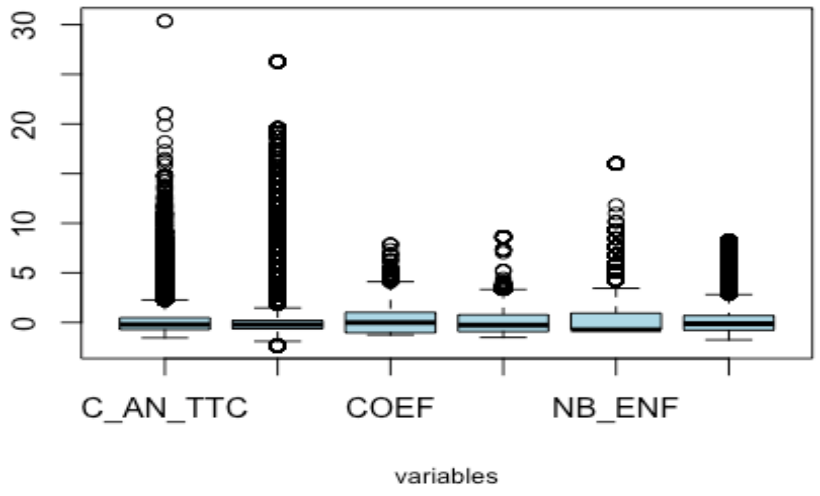
```
dv18$C_AN_TTC[1:100]
```

```
##   [1] 1479.3  443.3  344.3  373.3  720.3  537.3  369.3 1086.3  571.3  635.3
##  [11]  854.3  776.3 1235.3 1085.3  314.3  553.3 2429.3  488.3  793.3 1111.3
##  [21]  738.3  690.3  335.3  283.3  685.3  349.3  342.3  501.3  460.3  746.3
##  [31]  365.3  170.0  316.3  339.3  543.3 1123.3  977.3  644.3  266.0 1264.3
##  [41]  257.3  698.3  551.3  286.3  533.3  359.3  268.0  735.3  695.3  912.3
##  [51]  453.3  340.3  726.3  710.3  781.3 1241.3  447.3 1247.3 1104.3 3366.3
##  [61]  598.3  759.3  553.3  970.3  519.3 1298.0  901.3 1558.3  586.3 1055.3
##  [71] 1441.3  779.3  189.0  660.3  773.3  884.3  907.3  281.3  263.0  491.3
##  [81] 1128.3  171.3  714.3  793.3  811.3  283.0  861.3  361.3  863.3  360.3
##  [91]  435.3  162.0  464.3  796.3  981.3  978.3 1295.3 1230.3 1086.3  568.3

# IDENTIFIER LES QUANTI
library(plyr)
type_var_num = colwise(is.numeric)
var_num=names(dv18)[which(type_var_num(dv18)==TRUE)]

dv18Quanti = dv18[, c(var_num)]
str(dv18Quanti)

## 'data.frame':    614048 obs. of  9 variables:
##  $ ID_CLIENT: int  212753725 212232107 212547161 212507824 212167454 212679778 212135634 212663825 212
575321 212749153 ...
##  $ NUM_DEV  : int  1 1 1 2 2 4 2 1 3 1 ...
##  $ C_AN_TTC : num  1479 443 344 373 720 ...
##  $ COEFF    : int  185 40 65 35 57 65 50 100 65 40 ...
##  $ PUIS_FISC: int  4 4 6 7 8 5 6 5 4 5 ...
##  $ COEF     : int  185 40 65 35 57 65 50 100 65 40 ...
##  $ AGE_CLI  : int  24 60 21 48 39 19 60 18 19 52 ...
##  $ NB_ENF   : int  0 3 0 4 2 0 0 0 0 4 ...
##  $ ANC_VEH  : int  8 7 19 12 10 7 15 14 5 0 ...

etiq<-dv18[,-2:-21]
etiq<-etiq[,-3]
str(etiq)

## 'data.frame':    614048 obs. of  2 variables:
##  $ ID_CLIENT: int  212753725 212232107 212547161 212507824 212167454 212679778 212135634 212663825 212
575321 212749153 ...
##  $ trsf_dev : Factor w/ 2 levels "N","O": 1 1 2 1 1 1 1 1 2 1 ...

dq=merge(dv18Quanti,etiq,by="ID_CLIENT")
str(dq)

## 'data.frame':    2521044 obs. of  10 variables:
##  $ ID_CLIENT: int  211978166 211978166 211978166 211978166 211978166 211978166 211978166 211978166 211
978166 211978166 ...
##  $ NUM_DEV  : int  1 1 1 1 1 1 4 4 4 4 ...
##  $ C_AN_TTC : num  1471 1471 1471 1471 1471 ...
##  $ COEFF    : int  100 100 100 100 100 100 100 100 100 100 ...
##  $ PUIS_FISC: int  5 5 5 5 5 5 5 5 5 5 ...
##  $ COEF     : int  100 100 100 100 100 100 100 100 100 100 ...
##  $ AGE_CLI  : int  25 25 25 25 25 25 25 25 25 25 ...
##  $ NB_ENF   : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ ANC_VEH  : int  9 9 9 9 9 9 9 9 9 9 ...
##  $ trsf_dev : Factor w/ 2 levels "N","O": 2 1 1 1 1 1 2 1 1 1 ...

summary(dq)

##    ID_CLIENT            NUM_DEV           C_AN_TTC           COEFF
##  Min.   :211978166   Min.   :  1.000   Min.   :   65.0   Min.   : 35.00
##  1st Qu.:212238396   1st Qu.:  1.000   1st Qu.:  409.3   1st Qu.: 40.00
##  Median :212387709   Median :  2.000   Median :  621.3   Median : 65.00
##  Mean   :212392913   Mean   :  9.169   Mean   :  709.5   Mean   : 65.37
##  3rd Qu.:212551367   3rd Qu.:  4.000   3rd Qu.:  906.3   3rd Qu.: 90.00
##  Max.   :212760888   Max.   :318.000   Max.   :13335.3   Max.   :256.00
##    PUIS_FISC           COEF            AGE_CLI          NB_ENF
##  Min.   : 0.000   Min.   : 35.00   Min.   : 13.0   Min.   : 0.0000
##  1st Qu.: 4.000   1st Qu.: 40.00   1st Qu.: 22.0   1st Qu.: 0.0000
##  Median : 5.000   Median : 65.00   Median : 32.0   Median : 0.0000
##  Mean   : 5.524   Mean   : 65.37   Mean   : 36.1   Mean   : 0.8806
##  3rd Qu.: 6.000   3rd Qu.: 90.00   3rd Qu.: 48.0   3rd Qu.: 2.0000
##  Max.   :68.000   Max.   :256.00   Max.   :168.0   Max.   :20.0000
```

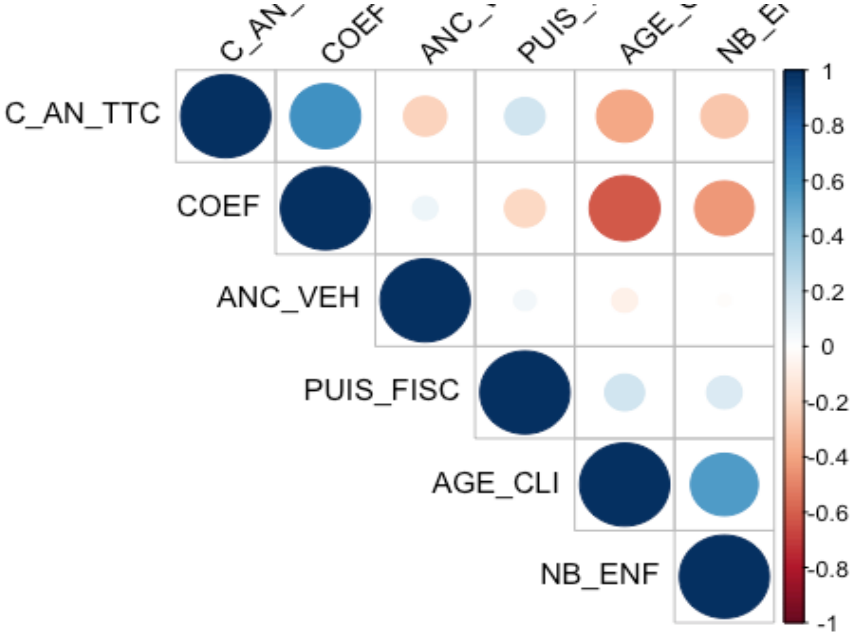
```
##     ANC_VEH       trsf_dev
##  Min.   :-1.000   N:1820577
##  1st Qu.: 5.000   O: 700467
##  Median : 9.000
##  Mean   : 9.769
##  3rd Qu.:14.000
##  Max.   :60.000

dq1=dq[,-4]
str(dq1)

## 'data.frame':    2521044 obs. of  9 variables:
##  $ ID_CLIENT: int  211978166 211978166 211978166 211978166 211978166 211978166 211978166 211978166 211
978166 211978166 ...
##  $ NUM_DEV  : int  1 1 1 1 1 1 4 4 4 4 ...
##  $ C_AN_TTC : num  1471 1471 1471 1471 1471 ...
##  $ PUIS_FISC: int  5 5 5 5 5 5 5 5 5 5 ...
##  $ COEF     : int  100 100 100 100 100 100 100 100 100 100 ...
##  $ AGE_CLI  : int  25 25 25 25 25 25 25 25 25 25 ...
##  $ NB_ENF   : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ ANC_VEH  : int  9 9 9 9 9 9 9 9 9 9 ...
##  $ trsf_dev : Factor w/ 2 levels "N","O": 2 1 1 1 1 1 2 1 1 1 ...

dq2=dq1[,-1]
str(dq2)

## 'data.frame':    2521044 obs. of  8 variables:
##  $ NUM_DEV  : int  1 1 1 1 1 1 4 4 4 4 ...
##  $ C_AN_TTC : num  1471 1471 1471 1471 1471 ...
##  $ PUIS_FISC: int  5 5 5 5 5 5 5 5 5 5 ...
##  $ COEF     : int  100 100 100 100 100 100 100 100 100 100 ...
##  $ AGE_CLI  : int  25 25 25 25 25 25 25 25 25 25 ...
##  $ NB_ENF   : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ ANC_VEH  : int  9 9 9 9 9 9 9 9 9 9 ...
##  $ trsf_dev : Factor w/ 2 levels "N","O": 2 1 1 1 1 1 2 1 1 1 ...

# boxplot sur données non normalisées
boxplot(dq2)
```



```
################################
# PARTIE 2. ANALYSE DESCRIPTIVE
################################

# lecture du graphe : on ne peut rien dire, pas de résultats significatifs

dq2norm=as.data.frame(scale(dq2[,2:7]))
sum(is.na(dq2$trsf_dev))

## [1] 0

str(dq2norm)

## 'data.frame':    2521044 obs. of  6 variables:
##  $ C_AN_TTC : num  1.83 1.83 1.83 1.83 1.83 ...
##  $ PUIS_FISC: num  -0.221 -0.221 -0.221 -0.221 -0.221 ...
##  $ COEF     : num  1.43 1.43 1.43 1.43 1.43 ...
##  $ AGE_CLI  : num  -0.724 -0.724 -0.724 -0.724 -0.724 ...
##  $ NB_ENF   : num  -0.737 -0.737 -0.737 -0.737 -0.737 ...
##  $ ANC_VEH  : num  -0.125 -0.125 -0.125 -0.125 -0.125 ...

summary(dq2norm)

##     C_AN_TTC         PUIS_FISC            COEF            AGE_CLI
##  Min.   :-1.5504   Min.   :-2.3236   Min.   :-1.2515   Min.   :-1.5065
##  1st Qu.:-0.7222   1st Qu.:-0.6411   1st Qu.:-1.0454   1st Qu.:-0.9194
##  Median :-0.2122   Median :-0.2205   Median :-0.0152   Median :-0.2672
##  Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000
##  3rd Qu.: 0.4734   3rd Qu.: 0.2001   3rd Qu.: 1.0150   3rd Qu.: 0.7764
##  Max.   :30.3720   Max.   :26.2789   Max.   : 7.8558   Max.   : 8.6036
##      NB_ENF           ANC_VEH
##  Min.   :-0.7368   Min.   :-1.7532
##  1st Qu.:-0.7368   1st Qu.:-0.7764
##  Median :-0.7368   Median :-0.1252
##  Mean   : 0.0000   Mean   : 0.0000
##  3rd Qu.: 0.9366   3rd Qu.: 0.6888
##  Max.   :15.9976   Max.   : 8.1778

class(dq2norm)

## [1] "data.frame"

boxplot(dq2norm, col="lightblue",xlab="variables",cex.lab=0.8,cex.main=0.8)
```

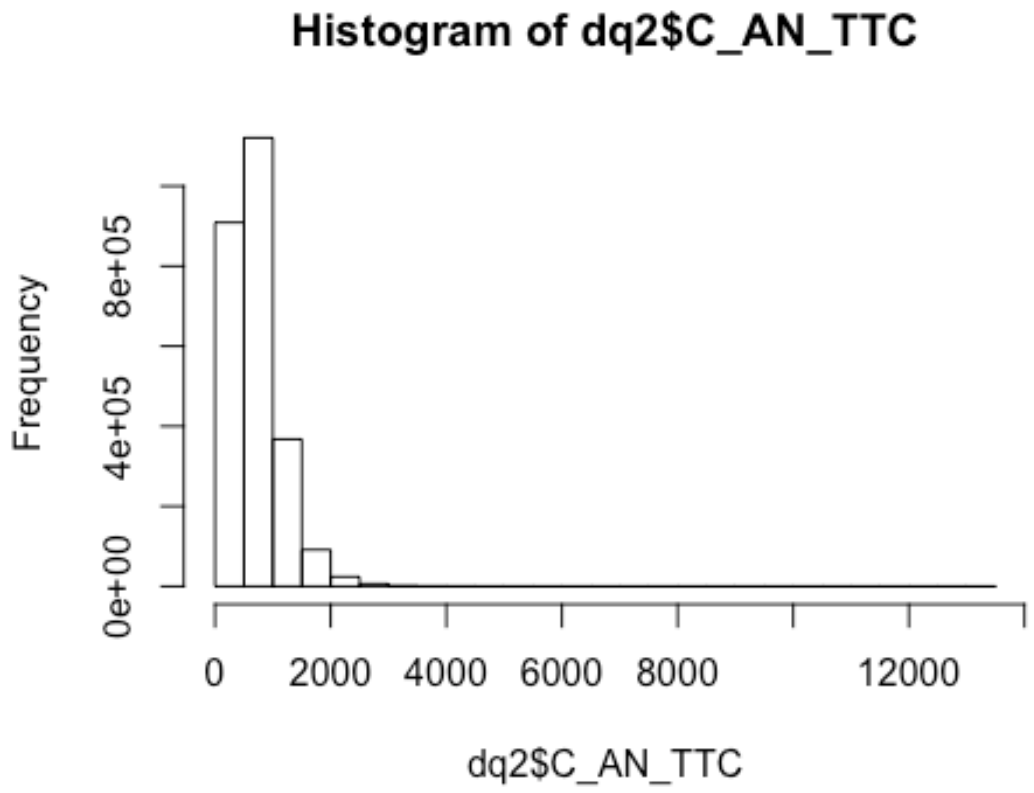


```r
# mise en place d'un modèle d'analyse discriminante linéaire avec le package MASS
dq2norm_y = dq2norm
dq2norm_y$trsf_dev = dq2$trsf_dev
str(dq2norm_y)

## 'data.frame':    2521044 obs. of  7 variables:
##  $ C_AN_TTC : num  1.83 1.83 1.83 1.83 1.83 ...
##  $ PUIS_FISC: num  -0.221 -0.221 -0.221 -0.221 -0.221 ...
##  $ COEF     : num  1.43 1.43 1.43 1.43 1.43 ...
##  $ AGE_CLI  : num  -0.724 -0.724 -0.724 -0.724 -0.724 ...
##  $ NB_ENF   : num  -0.737 -0.737 -0.737 -0.737 -0.737 ...
##  $ ANC_VEH  : num  -0.125 -0.125 -0.125 -0.125 -0.125 ...
##  $ trsf_dev : Factor w/ 2 levels "N","O": 2 1 1 1 1 1 2 1 1 1 ...

library(MASS)
lda(dq2norm_y$trsf_dev ~.,data=dq2norm_y)

## Call:
## lda(dq2norm_y$trsf_dev ~ ., data = dq2norm_y)
##
## Prior probabilities of groups:
##        N        O
## 0.722152 0.277848
##
## Group means:
##      C_AN_TTC   PUIS_FISC        COEF     AGE_CLI      NB_ENF     ANC_VEH
## N  0.06174682  0.04834374  0.02012921  0.03479996  0.02729816 -0.04189115
## O -0.16048557 -0.12564976 -0.05231764 -0.09044824 -0.07095039  0.10887887
##
## Coefficients of linear discriminants:
##                  LD1
## C_AN_TTC  -0.5074008
## PUIS_FISC -0.3589032
## COEF      -0.4774354
## AGE_CLI   -0.6411595
## NB_ENF    -0.2028685
## ANC_VEH    0.3199303

getwd()

## [1] "/Users/lepicaut/Desktop/sos/MATMUT/0_0_matmut/fevrier_2017/00000_code"

write.csv2(dq2norm_y,file='dq2norm_y$trsf_dev.csv')
head(dq2norm_y)

##   C_AN_TTC  PUIS_FISC     COEF    AGE_CLI     NB_ENF    ANC_VEH trsf_dev
## 1 1.832529 -0.2205163 1.427131 -0.7237554 -0.7368313 -0.1252035        O
## 2 1.832529 -0.2205163 1.427131 -0.7237554 -0.7368313 -0.1252035        N
## 3 1.832529 -0.2205163 1.427131 -0.7237554 -0.7368313 -0.1252035        N
## 4 1.832529 -0.2205163 1.427131 -0.7237554 -0.7368313 -0.1252035        N
## 5 1.832529 -0.2205163 1.427131 -0.7237554 -0.7368313 -0.1252035        N
## 6 1.832529 -0.2205163 1.427131 -0.7237554 -0.7368313 -0.1252035        N

head(dq2norm_y$trsf_dev)

## [1] O N N N N N
## Levels: N O

# corrélation linéaire entre les variables explicatives
library(corrplot)
corrplot(cor(dq2norm),type="upper", order="hclust",tl.col="black", tl.srt=45)
```

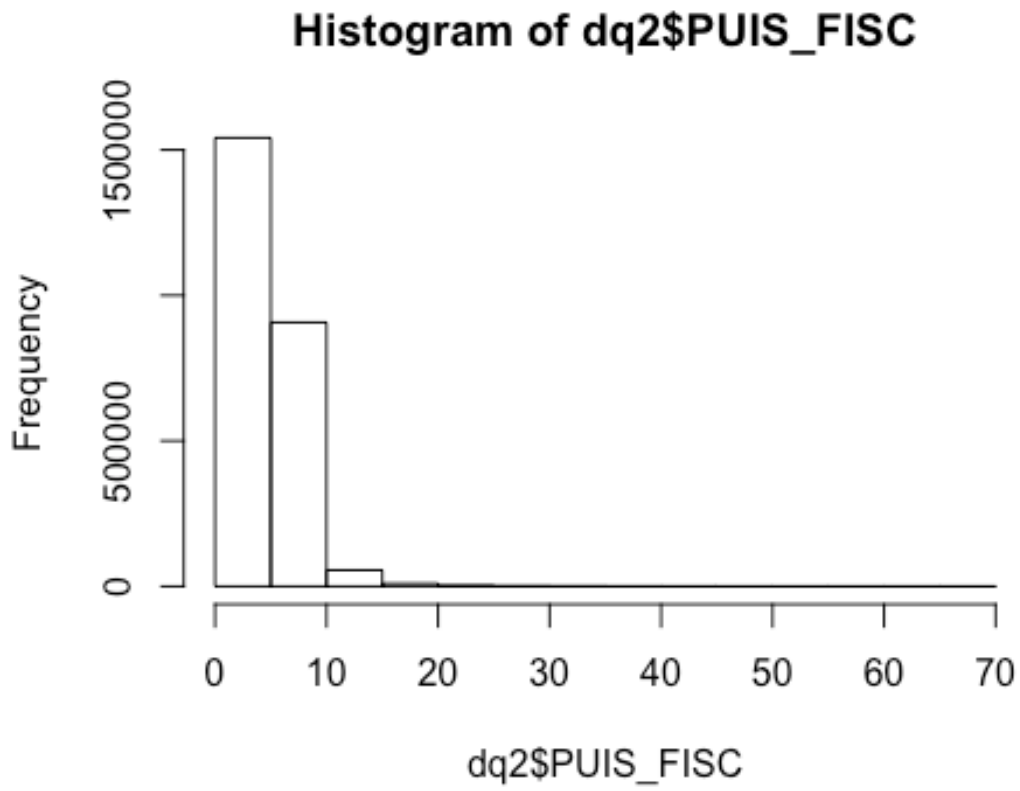



```
# analyse
#ANALYSE :
# . la Cotisation Annuelle est corrélée avec le Coefficient B/M,
# . le Coefficient est corrélé avec Age Client.
#COMMENTAIRE :
#  . ces éléments sont illustratifs du modèle tarifaire MATMUT pour un contrat Auto.
```

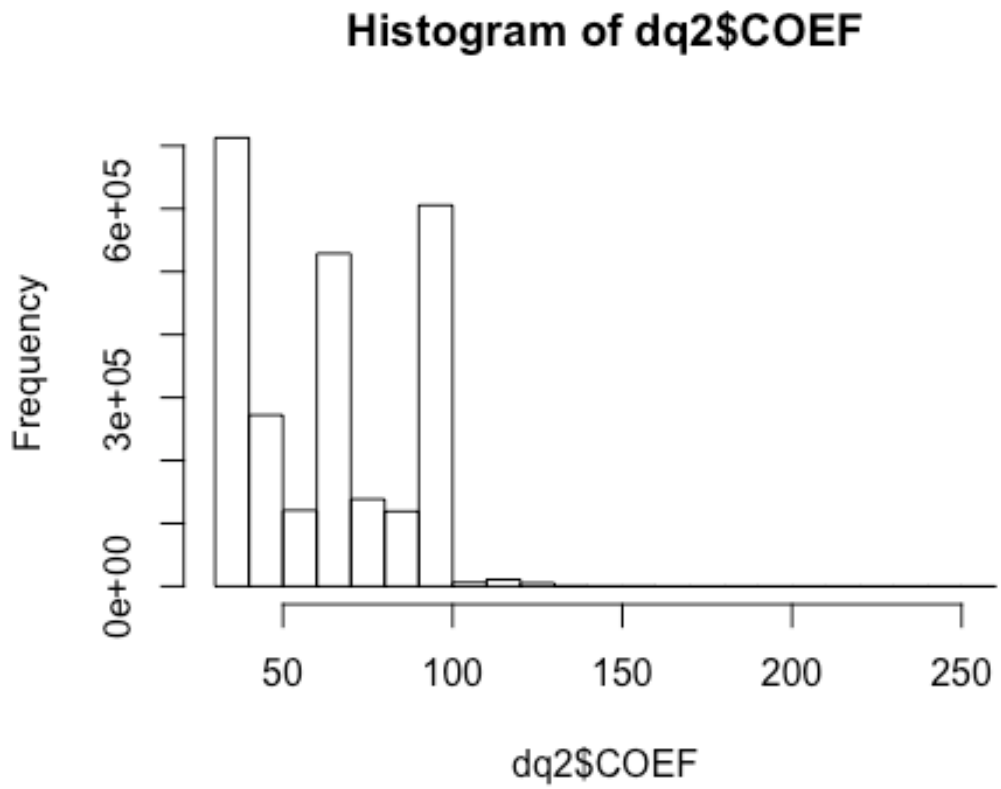
```r
hist(dq2$C_AN_TTC)
```



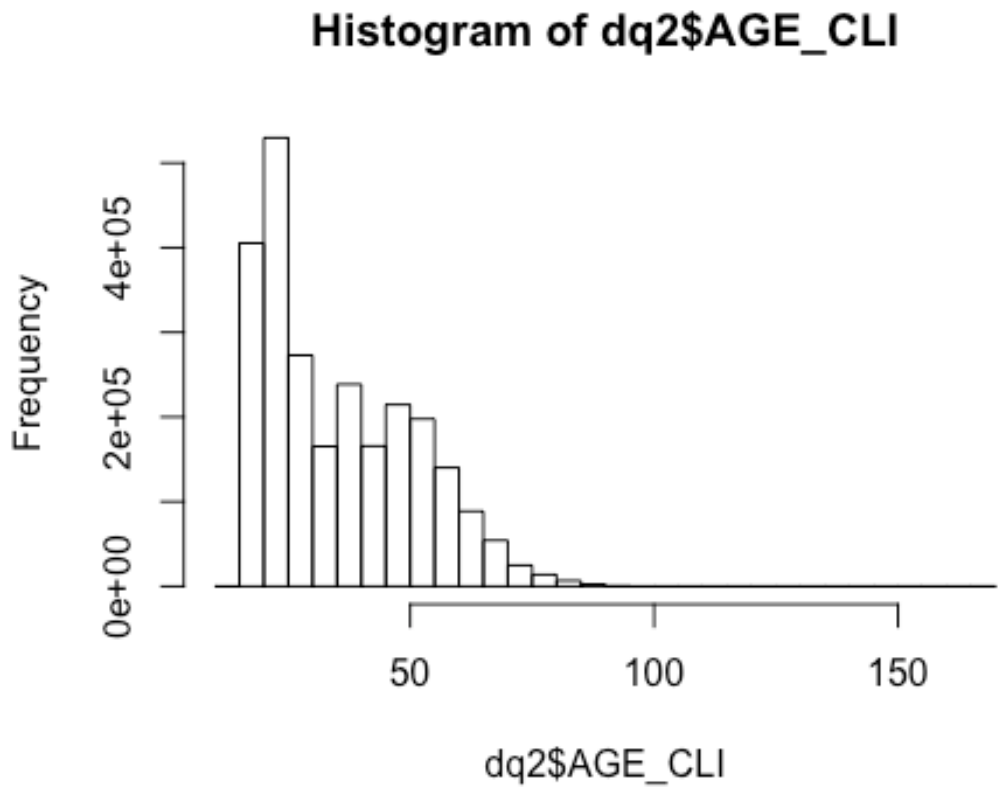

```r
hist(dq2$PUIS_FISC)
```



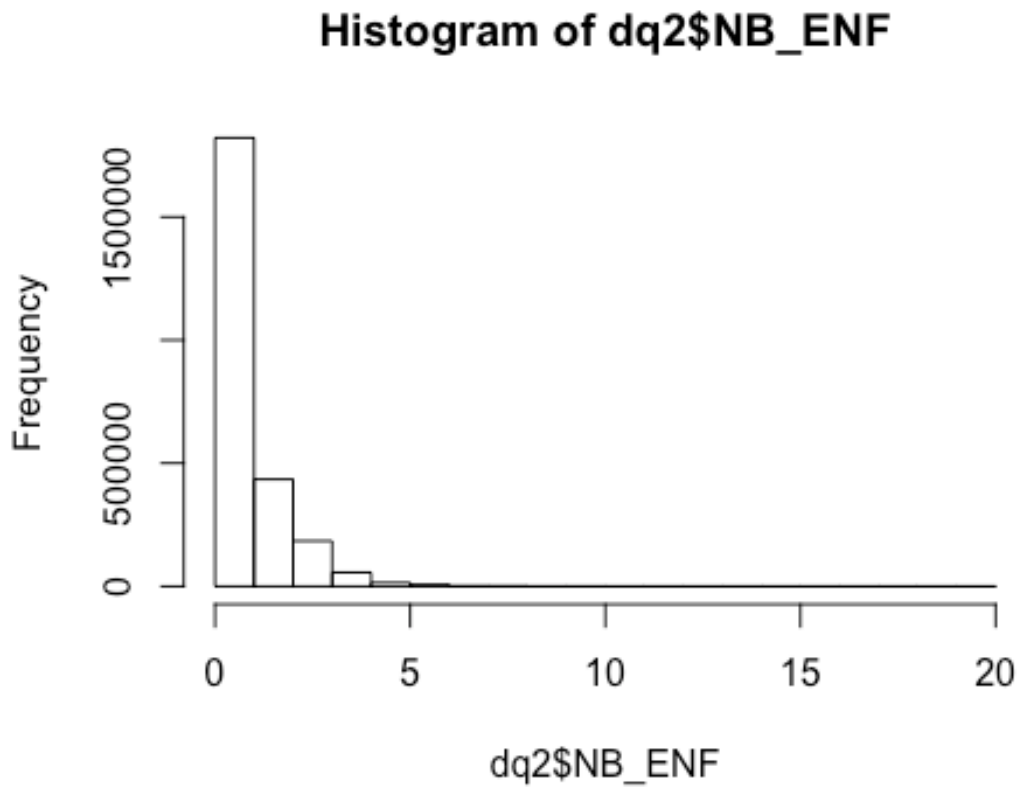
```r
hist(dq2$COEF)
```




```r
hist(dq2$AGE_CLI)
```



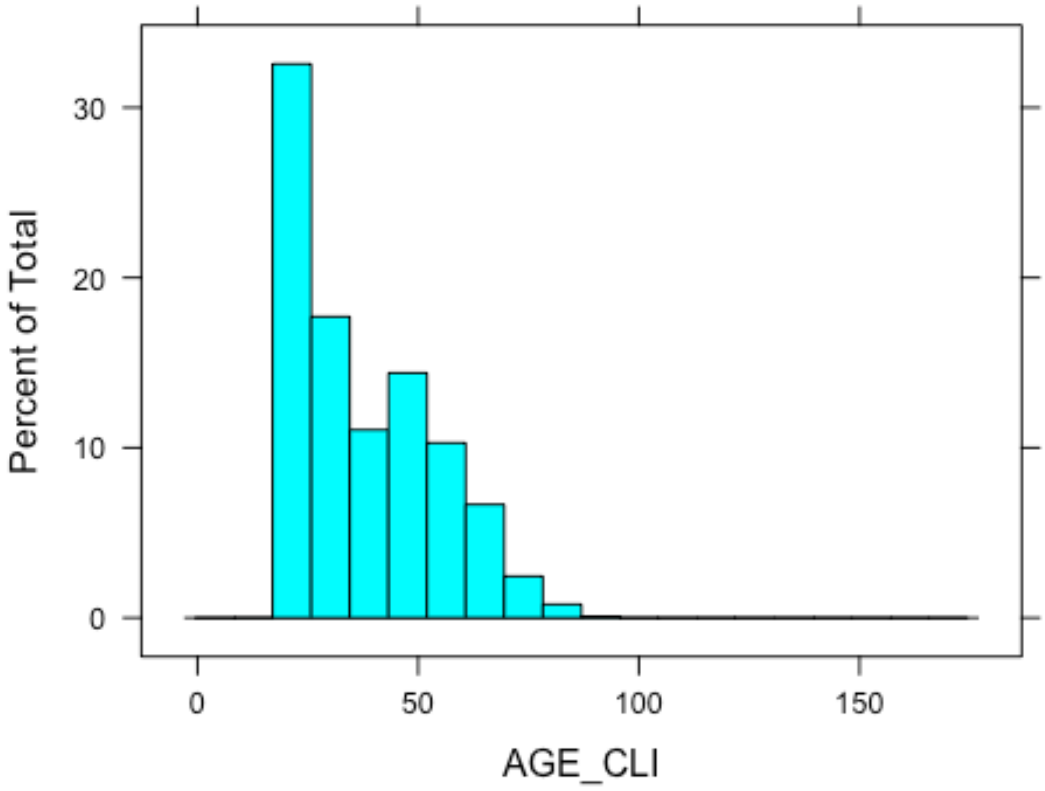
```r
hist(dq2$NB_ENF)
```



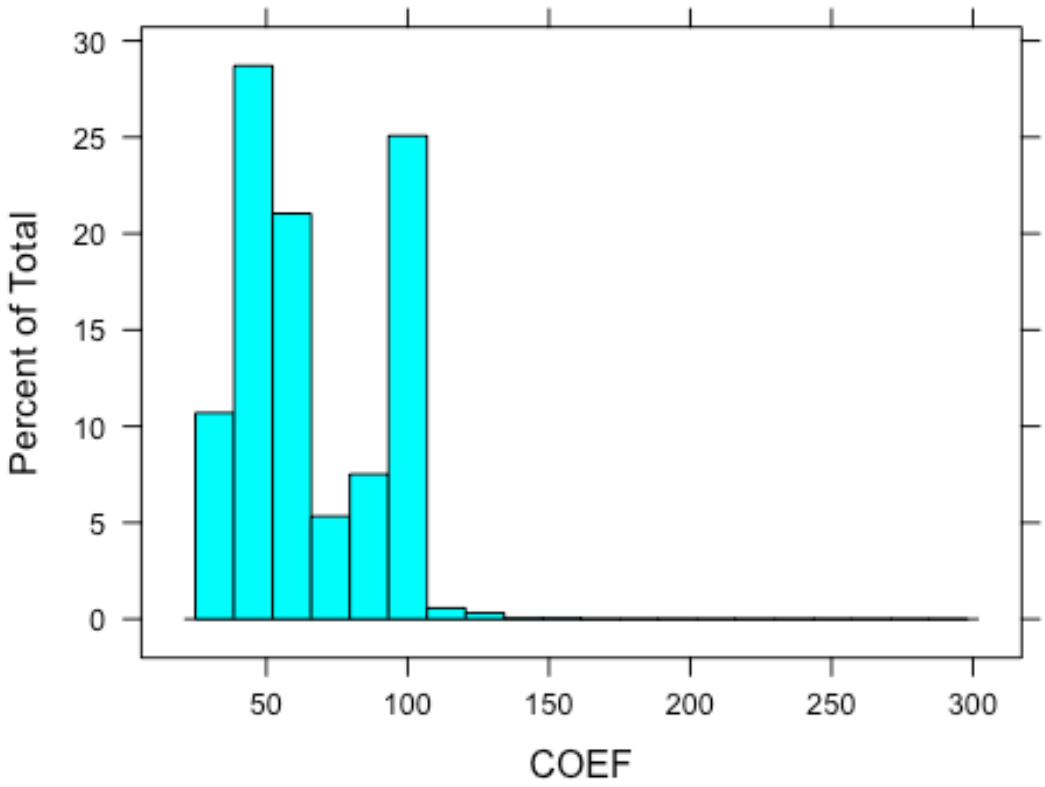
```
################################
# PARTIE 3. MODELISATION DES CLASSIFIEURS
################################

################################
# PARTIE 3.1. : SELECTION DU MODELE (Feature Ingineering)
################################

################################
# PARTIE 3.1.1 : ANALYSE LINEAIRE DISCRIMINANTE (LDA)
################################

# mise en place d'un modèle d'analyse linéaire discriminante avec le package MASS
# modele complet : 6 var quanti + var etiquette
dq2norm_y = dq2norm
dq2norm_y$trsf_dev = dq2$trsf_dev
str(dq2norm_y)

## 'data.frame':    2521044 obs. of  7 variables:
##  $ C_AN_TTC : num  1.83 1.83 1.83 1.83 1.83 ...
##  $ PUIS_FISC: num  -0.221 -0.221 -0.221 -0.221 -0.221 ...
##  $ COEF     : num  1.43 1.43 1.43 1.43 1.43 ...
##  $ AGE_CLI  : num  -0.724 -0.724 -0.724 -0.724 -0.724 ...
##  $ NB_ENF   : num  -0.737 -0.737 -0.737 -0.737 -0.737 ...
##  $ ANC_VEH  : num  -0.125 -0.125 -0.125 -0.125 -0.125 ...
##  $ trsf_dev : Factor w/ 2 levels "N","O": 2 1 1 1 1 1 2 1 1 1 ...

library(MASS)
lda(dq2norm_y$trsf_dev ~.,data=dq2norm_y)

## Call:
## lda(dq2norm_y$trsf_dev ~ ., data = dq2norm_y)
##
## Prior probabilities of groups:
##        N        O
## 0.722152 0.277848
##
## Group means:
##      C_AN_TTC   PUIS_FISC        COEF     AGE_CLI      NB_ENF     ANC_VEH
## N  0.06174682  0.04834374  0.02012921  0.03479996  0.02729816 -0.04189115
## O -0.16048557 -0.12564976 -0.05231764 -0.09044824 -0.07095039  0.10887887
##
## Coefficients of linear discriminants:
##                  LD1
## C_AN_TTC  -0.5074008
## PUIS_FISC -0.3589032
## COEF      -0.4774354
## AGE_CLI   -0.6411595
## NB_ENF    -0.2028685
## ANC_VEH    0.3199303

# analyse résultats :

# 1. quels sont les variables discriminantes :
# critère de sélection des variables dans les axes discriminants
# on retient les var dont le coeff. > 0.45
# les variables discriminantes entre les "oui" et les "non"
# COEF > C_AN_TTC > AGE_CLI



# 2. on qualifie les groupes "OUI" et "NON"
# chez les oui : COEF, AGE_CLI & C_AN_TTC qualifient ce groupe (coordonnées négatives sur l'axe)
# chez les non : C_AN_TTC, AGE_CLI & COEF qualifient ce groupe.(coordonnées positives sur l'axe)

lda(dq2norm_y$trsf_dev ~C_AN_TTC+AGE_CLI+COEF,data=dq2norm_y)

## Call:
## lda(dq2norm_y$trsf_dev ~ C_AN_TTC + AGE_CLI + COEF, data = dq2norm_y)
##
```

```
## Prior probabilities of groups:
##        N        O
## 0.722152 0.277848
##
## Group means:
##      C_AN_TTC     AGE_CLI        COEF
## N  0.06174682  0.03479996  0.02012921
## O -0.16048557 -0.09044824 -0.05231764
##
## Coefficients of linear discriminants:
##                 LD1
## C_AN_TTC -0.9064616
## AGE_CLI  -0.8761939
## COEF     -0.2136779
```

```
# analyse résultats :
# 1. quels sont les variables discriminantes :
# critère de sélection des variables dans les axes discriminants
# on retient les var dont le coeff. > 0.5
# les variables discriminantes entre les "oui" et les "non"
# AGE_CLI > C_AN_TTC
# 2. on qualifie les groupes "OUI" et "NON"
# chez les oui : AGE_CLI > C_AN_TTC (coordonnées négatives sur l'axe)
# chez les non : C_AN_TTC > AGE_CLI (coordonnées positives sur l'axe)


# représentation graphique des données avec lattice :
# en ordonnée trsf_dev dépendante en abscisse les var explicatives quantitatives
library(lattice)
histogram(C_AN_TTC~AGE_CLI)
```



```r
histogram(C_AN_TTC~COEF)
```




```r
lda(dq2norm_y$trsf_dev ~C_AN_TTC+AGE_CLI,data=dq2norm_y)

## Call:
## lda(dq2norm_y$trsf_dev ~ C_AN_TTC + AGE_CLI, data = dq2norm_y)
##
## Prior probabilities of groups:
##        N        O
## 0.722152 0.277848
##
## Group means:
##      C_AN_TTC     AGE_CLI
## N  0.06174682  0.03479996
## O -0.16048557 -0.09044824
##
## Coefficients of linear discriminants:
##                 LD1
## C_AN_TTC -1.0095862
## AGE_CLI  -0.7891436

# analyse résultats du modèle 2 variables discriminantes : confirmation des résultats du modèle à 3 varia
bles discri.
# 1. quels sont les variables discriminantes :
# critère de sélection des variables dans les axes discriminants
# on retient les var dont le coeff. > 0.5
# les variables discriminantes entre les "oui" et les "non"
# AGE_CLI > C_AN_TTC
# 2. on qualifie les groupes "OUI" et "NON"
# chez les oui : AGE_CLI > C_AN_TTC (coordonnées négatives sur l'axe)
# chez les non : C_AN_TTC > AGE_CLI (coordonnées positives sur l'axe)

lda(dq2norm_y$trsf_dev ~C_AN_TTC+AGE_CLI,data=dq2norm_y,prior=c(0.5,0.5))

## Call:
## lda(dq2norm_y$trsf_dev ~ C_AN_TTC + AGE_CLI, data = dq2norm_y,
##     prior = c(0.5, 0.5))
##
## Prior probabilities of groups:
```

```
##   N   O
## 0.5 0.5
##
## Group means:
##      C_AN_TTC     AGE_CLI
## N  0.06174682  0.03479996
## O -0.16048557 -0.09044824
##
## Coefficients of linear discriminants:
##                 LD1
## C_AN_TTC -1.0095862
## AGE_CLI  -0.7891436
```

```
# après modification de la répartition du prior pour valider la spécification du modèle
# on fixe la proba Oui = proba Non = 1/2

# résultats 1 : en terme de variable discriminante, la sélection des vars discriminantes est stable,
# AGE_CLI & C_AN_TTC sont bien les 2 variables qui discriminent les étiquettes "Oui" et "Non".

# résultats 2 : concernant la qualification des groupes étiquetés par les variables discriminantes est st
able / valide
# chez les oui : AGE_CLI > C_AN_TTC (coordonnées négatives sur l'axe)
# chez les non : C_AN_TTC > AGE_CLI (coordonnées positives sur l'axe)



# estimer le taux de mauvais classement par validation croisée
prev<-lda(dq2norm_y$trsf_dev ~ C_AN_TTC + AGE_CLI, data = dq2norm_y,CV=TRUE)$class
#prev
table(prev,dq2norm_y$trsf_dev)
```

```
##
## prev       N       O
##    N 1820577  700467
##    O       0       0
```

```
t=sum(prev!=dq2norm_y$trsf_dev)/nrow(dq2norm_y)
100*t
```

```
## [1] 27.7848
```

```
# analyse de la matrice de confusion et du taux de mauvais classement
# on voit que le premier modèle (0.7/0.2) est identique avec le modèle avec prior modifié
# (0.5/0.5)
# puisque le taux de mauvaix classement est de 27.784%,
# il correspond aux lignes étiquettées "Oui" et prédite "Non" par le modèle à
# 2 var discriminantes. Avec notre dataset initial, dans le cadre de la théorie des grands
# nombres (i.e. modèle fréquentiste), si on tire 100 ligne aléatoirement, on obtiendra en moyenne 72% Non
et 28% Oui.
# ce dataset en terme de modèle d'apprentissage, apprend à 72% du Non et 28% de Oui.
# ce dataset, en moyenne (en fréquence d'apparition), explique mieux le "non" que le "oui".


# matrice de confusion - indicateur de performance du classifieur - lda avec 2 variables explicatives
#
library(caret)
confusionMatrix(data=prev, reference=dq2norm_y$trsf_dev)
```

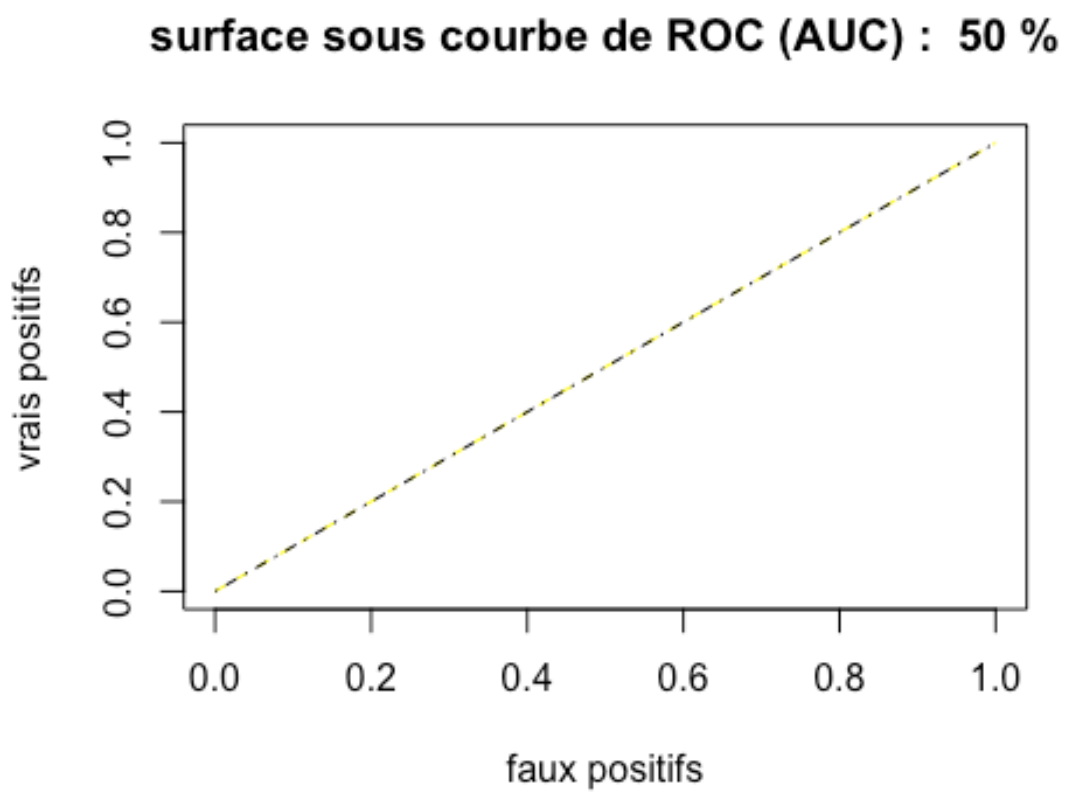
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction       N       O
##          N 1820577  700467
##          O       0       0
##
##                Accuracy : 0.7222
##                  95% CI : (0.7216, 0.7227)
##     No Information Rate : 0.7222
##     P-Value [Acc > NIR] : 0.5003
##
##                   Kappa : 0
##  Mcnemar's Test P-Value : <2e-16
```

```
##
##             Sensitivity : 1.0000
##             Specificity : 0.0000
##          Pos Pred Value : 0.7222
##          Neg Pred Value :    NaN
##              Prevalence : 0.7222
##          Detection Rate : 0.7222
##    Detection Prevalence : 1.0000
##       Balanced Accuracy : 0.5000
##
##        'Positive' Class : N
##
```

```
#confusionMatrix(data=prev, reference=dq2norm_y$trsf_dev,mode="prec_recall")

library(ROCR)
fr <- data.frame(score = c(prev),label =c(dq2norm_y$trsf_dev))
pred <- prediction(fr$score, fr$label)
perf <- performance(pred,"tpr", "fpr")
y.auc.perf= performance(pred, measure="auc",x.measure="cutoff")
auc=round(unlist(y.auc.perf@y.values),2)*100
print(auc)
```

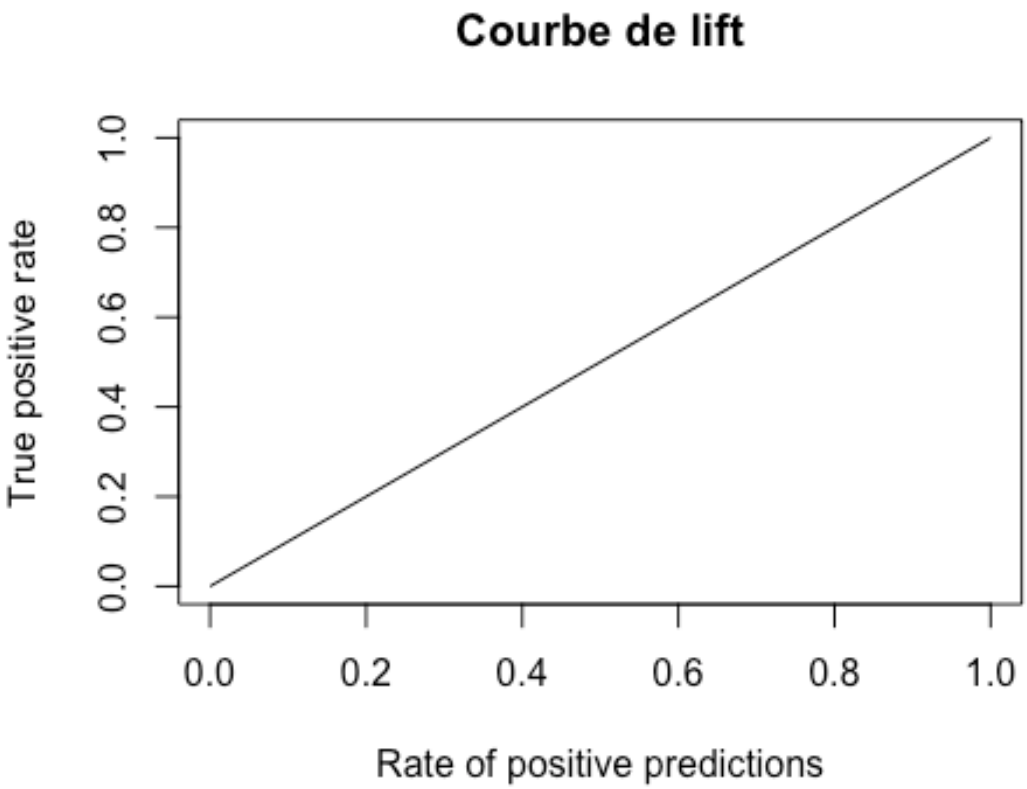
```
## [1] 50
```

```
plot(perf, col="yellow", lty=2, pch=32,
     main = paste("surface sous courbe de ROC (AUC) : ",auc, "%"),
     xlab = "faux positifs",
     ylab = "vrais positifs")

segments(0,0,1,1,lty=4)
```

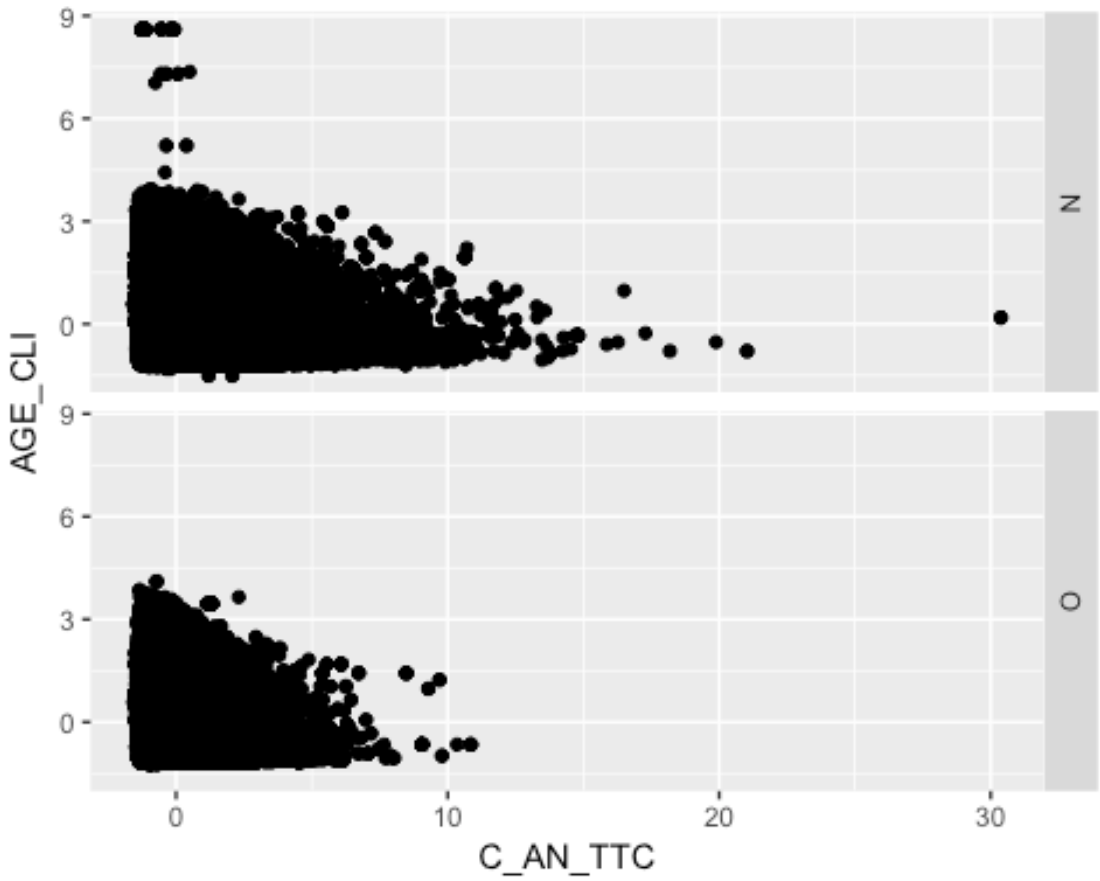



```
#graphics.off()

# la courbe ROC du modèle lda, est alignée sur la courbe de prédiction basée sur le hasard (bissectrice d
u cadran).
# la prédiction de mon modèle lda ne fait pas mieux que "le modèle du hasard".

# courbe de lift
```

```
lift<- performance(pred,"tpr","rpp")
plot(lift, main='Courbe de lift')
```

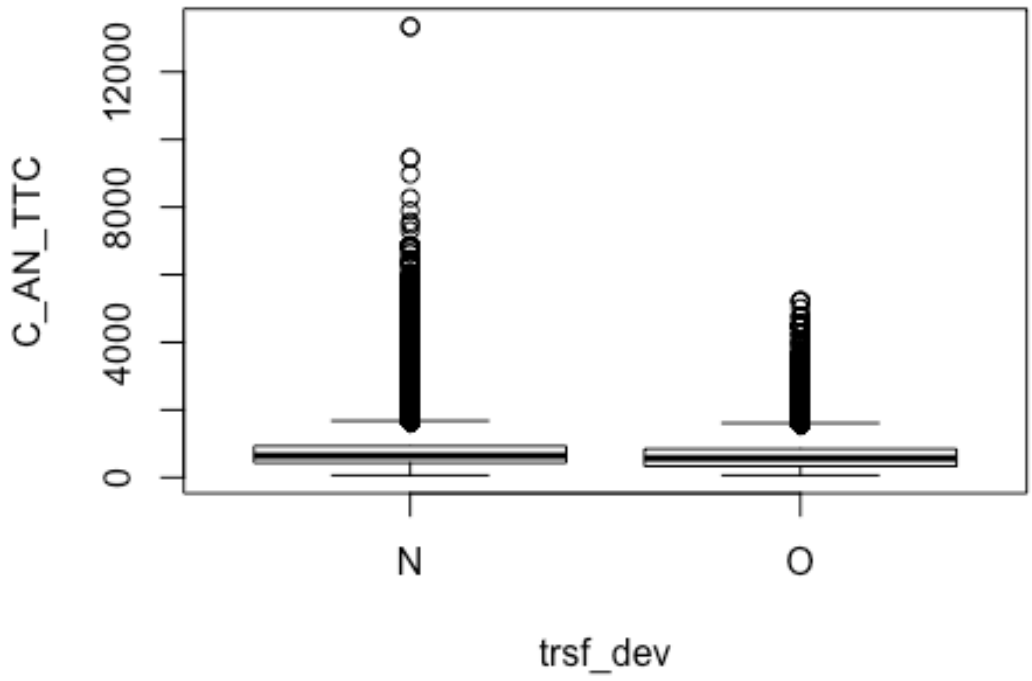



```
################################
# VISUALISATION GRAPHIQUE DES ELEMENTS DE CLASSIFICATION
################################

# VISUALISATION DES CLASSES QUALIFIEES AVEC 2 VAR DISCRIMINANTES
library(ggplot2)
qplot(C_AN_TTC, AGE_CLI, data=dq2norm_y, facets= trsf_dev~.)
```




```
# ANALYSE
# croisement var cotisation & âge pour expliquer l'étiquette Oui / Non devis transformé

# analyse en diagonale :
# analyse 1 : age=vieux en relation avec cotisation = élevée alors étiquette = Non
# analyse 2 : age=jeune en relation avec cotisation = élevée alors étiquette = Non
# analyse 1 & 2 : connue / inconnue sur les cibles jeunes et vieux par les métiers MATMUT.

# ANALYSE DISCRIMINANTE - AVEC LE MODELE OPTIMAL - Probabilité du Prior fixé à (0.5;0.5)
prev<-lda(dq2norm_y$trsf_dev ~ C_AN_TTC + AGE_CLI, data = dq2norm_y,
    prior = c(0.5, 0.5),CV=TRUE)$class
table(prev,dq2norm_y$trsf_dev)

##
## prev       N      O
##    N 939164 263333
##    O 881413 437134

t=sum(prev!=dq2norm_y$trsf_dev)/nrow(dq2norm_y)
100*t

## [1] 45.40762

# analyse de la matrice de confusion et du taux de mauvais classement
# en analyse comparée, avec la distribution du prior redressé (0.5,0.5)
# le taux de mauvais classé est dégradé = il prend en compte la partie du Oui redressé
# de 27.78% à 50,00%

# analyse discriminante avec fonction quadratique
qda(dq2norm_y$trsf_dev~C_AN_TTC+AGE_CLI,data=dq2norm_y)

## Call:
## qda(dq2norm_y$trsf_dev ~ C_AN_TTC + AGE_CLI, data = dq2norm_y)
##
## Prior probabilities of groups:
##        N        O
## 0.722152 0.277848
##
## Group means:
```

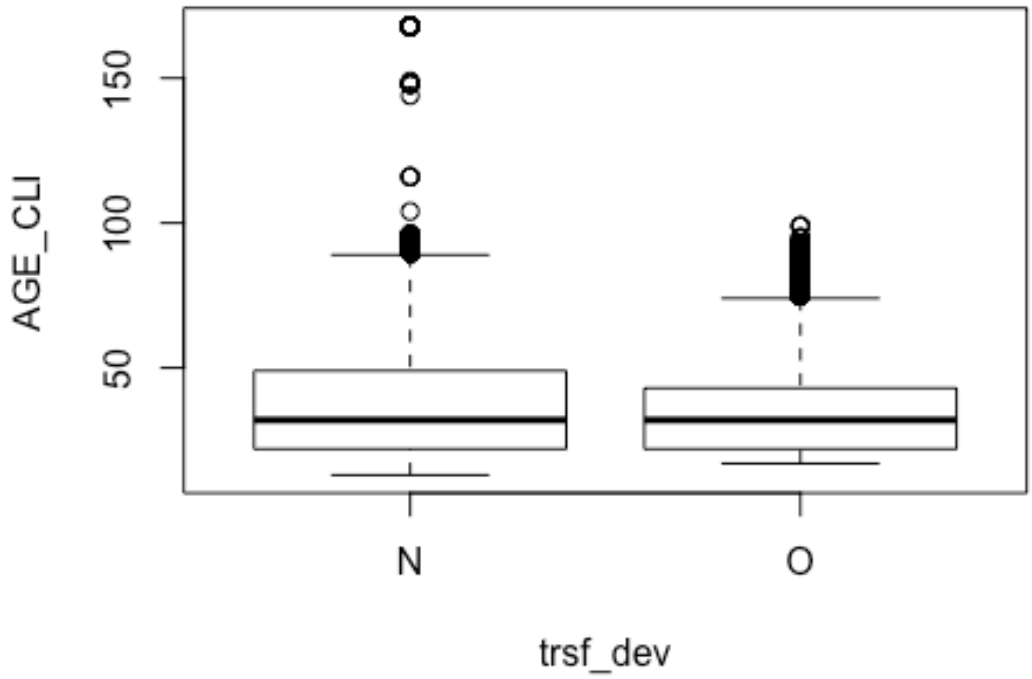
```
##      C_AN_TTC     AGE_CLI
## N  0.06174682  0.03479996
## O -0.16048557 -0.09044824

# pour rappel l'analyse linéaire discriminante, fait l'hypothèse  que la matrice
# de variance-covariance des var. explicatives sont les mêmes pour chaque groupe (Oui / Non).
# l'analyse discriminante quadratique s'affranchit de cette hypothèse,
# c'est à dire que la matrice de var-cov est réellement différente entre les groupes
# de "oui" et groupe de "non".
# mais ici : il n'y a pas de différence significative entre les analyses "lda" et "qda".
# donc il n'y a pas de réelle différence entre le groupe de "oui" et "non".
# ce qui confirme qu'il y a un groupe sur-représenté le groupe de "non".

# estimer le taux de mauvais classement par validation croisée avec méthode qda
prev<-qda(dq2norm_y$trsf_dev ~ C_AN_TTC + AGE_CLI, data = dq2norm_y,CV=TRUE)$class
table(prev,dq2norm_y$trsf_dev)

##
## prev       N       O
##    N 1820577  700467
##    O       0       0

t=sum(prev!=dq2norm_y$trsf_dev)/nrow(dq2norm_y)
100*t

## [1] 27.7848

# ANALYSE DU POUVOIR DISCRIMINANT DES VAR EXPLICATIVES
# test de moyennes sur les var. discriminantes entre les Groupes "Oui" et "Non".
# Estimer les statistiques de base par sous-population :

##### avec la variable C_AN_TTC
# 1. comparer graphiquement les 2 sous-populations
boxplot(C_AN_TTC ~ trsf_dev, ylab ="C_AN_TTC", xlab = "trsf_dev", data = dq2)
```




```
# analyse : visuellement la boxplot des "Oui" plus basse que la boxplot des "Non"

# 2. estimer les statistiques de bases par sous-populations
tapply(dq2[,"C_AN_TTC"], dq2[,"trsf_dev"], mean, na.rm=TRUE)
```

```
##        N        O
## 735.1762 642.7929

tapply(dq2[,"C_AN_TTC"], dq2[,"trsf_dev"], sd, na.rm=TRUE)

##        N        O
## 427.2770 375.8921

tapply(dq2[,"C_AN_TTC"], dq2[,"trsf_dev"], quantile, na.rm=TRUE)

## $N
##      0%     25%     50%     75%    100%
##    65.0   433.3   642.3   931.3 13335.3
##
## $O
##     0%    25%    50%    75%   100%
##   65.0  334.3  568.3  845.0 5221.3

# analyse:
# sur la variable cotisation annuelle TTC, les valeurs calculées.
# moyenne, écart-type, quartiles sont plus faibles chez les "Oui" que chez les "Non".

# 3. test de normalité
# 3.1 test de normalité des "Oui" avec C_AN_TTC
dq2_e<-dq2[sample(1:nrow(dq2),5000),]
select.oui<-dq2_e[,"trsf_dev"]=="O"
shapiro.test(dq2_e[select.oui,"C_AN_TTC"])

##
##  Shapiro-Wilk normality test
##
## data:  dq2_e[select.oui, "C_AN_TTC"]
## W = 0.89177, p-value < 2.2e-16

# analyse : valeur p-value < 5%, on rejette la normalité de la var C_AN_TTC pour les "Oui".

# 3.1 test de normalité des "Non" avec C_AN_TTC
dq2_e<-dq2[sample(1:nrow(dq2),5000),]
select.oui<-dq2_e[,"trsf_dev"]=="N"
shapiro.test(dq2_e[select.oui,"C_AN_TTC"])

##
##  Shapiro-Wilk normality test
##
## data:  dq2_e[select.oui, "C_AN_TTC"]
## W = 0.89653, p-value < 2.2e-16

# analyse : valeur p-value < 5%, on rejette la normalité de la var C_AN_TTC pour les "Non".

# conclusion : pour réaliser une analyse linéaire discriminante, il faut normaliser,
# les variables explicatives du dataset initiale

# 4. test d'égalité des variances
var.test(C_AN_TTC~trsf_dev, conf.level=.95,data=dq2)

##
##  F test to compare two variances
##
## data:  C_AN_TTC by trsf_dev
## F = 1.2921, num df = 1820600, denom df = 700470, p-value < 2.2e-16
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
##  1.287814 1.296373
## sample estimates:
## ratio of variances
##            1.29209

# analyse : valeur p-value < à 5%, la variance est significativement différence pour la var C_AN_TTC
# entre le groupe des  "Oui" et le groupe des "Non" avec un intervalle de confiance à 95%

# 5. test d'égalité des moyennes
t.test(C_AN_TTC~trsf_dev, alternative='two.sided',conf.level=0.95,var.equal=FALSE, data=dq2)
```

```
##
##  Welch Two Sample t-test
##
## data:  C_AN_TTC by trsf_dev
## t = 168.11, df = 1433700, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  91.30615 93.46031
## sample estimates:
## mean in group N mean in group O
##        735.1762        642.7929

# analyse : C_AN_TTC~trsf_dev, on rejette l'égalité des moyennes, la moyenne des Cotisations des "Oui"
# est significativement différente de la moyenne des Cotisations des "Non" avec un intervalle de confianc
e à 95%.
# le comportement de la transformation est différent différent on fonction de la cotisisation annuelle =
prime d'assurance auto,
# (et donc de la gamme de véhicule à assurer, du B/M, ..., la tarification assurance auto joue bien son r
ôle,
# CONCLUSION : la var explicative C_AN_TTC discrimine bien la var à expliquer taux de transformation (Oui
/ Non)

##### avec la variable AGE_CLI
# 1. comparer graphiquement les 2 sous-populations
boxplot(AGE_CLI ~ trsf_dev, ylab ="AGE_CLI", xlab = "trsf_dev", data = dq2)
```




```
# analyse : visuellement la boxplot des "Oui" plus basse que la boxplot des "Non".

# 2. estimer les statistiques de bases par sous-populations
tapply(dq2[,"AGE_CLI"], dq2[,"trsf_dev"], mean, na.rm=TRUE)

##        N        O
## 36.62959 34.70938

tapply(dq2[,"AGE_CLI"], dq2[,"trsf_dev"], sd, na.rm=TRUE)

##        N        O
## 15.72065 14.17588
```

```r
tapply(dq2[,"AGE_CLI"], dq2[,"trsf_dev"], quantile, na.rm=TRUE)

## $N
##   0%  25%  50%  75% 100%
##   13   22   32   49  168
##
## $O
##   0%  25%  50%  75% 100%
##   17   22   32   43   99

# analyse:
# sur la AGE_CLI, les valeurs calculées.
# moyenne, écart-type, quartiles sont plus faibles chez les "Oui" que chez les "Non".
# en revanche, il y a égalité pour le quartile (50%), c'est à dire la médiane.

# 3. test de normalité
# 3.1 test de normalité des "Oui" avec AGE_CLI
dq2_e<-dq2[sample(1:nrow(dq2),5000),]
select.oui<-dq2_e[,"trsf_dev"]=="O"
shapiro.test(dq2_e[select.oui,"AGE_CLI"])

##
##  Shapiro-Wilk normality test
##
## data:  dq2_e[select.oui, "AGE_CLI"]
## W = 0.88941, p-value < 2.2e-16

# analyse : valeur p-value < 5%, on rejette la normalité de la var AGE_CLI pour les "Oui".

# 3.1 test de normalité des "Non" avec AGE_ClI
dq2_e<-dq2[sample(1:nrow(dq2),5000),]
select.oui<-dq2_e[,"trsf_dev"]=="N"
shapiro.test(dq2_e[select.oui,"AGE_CLI"])

##
##  Shapiro-Wilk normality test
##
## data:  dq2_e[select.oui, "AGE_CLI"]
## W = 0.90128, p-value < 2.2e-16

# analyse : valeur p-value < 5%, on rejette la normalité de la var AGE_CLI pour les "Non".

# conclusion :
#. pour réaliser une analyse linéaire discriminante, il faut normaliser,
# les variables explicatives du dataset initiale (changement de var = appliquer une transformation sur le
s Xi,
# pour viser une distribution de Xi en cloche "gaussiennne")

# 4. test d'égalité des variances
var.test(AGE_CLI~trsf_dev, conf.level=.95,data=dq2)

##
##  F test to compare two variances
##
## data:  AGE_CLI by trsf_dev
## F = 1.2298, num df = 1820600, denom df = 700470, p-value < 2.2e-16
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
##  1.225749 1.233895
## sample estimates:
## ratio of variances
##           1.229819

# analyse : valeur p-value < à 5%, la variance est significativement différence pour la var AGE_CLI
# entre le groupe des  "Oui" et le groupe des "Non" avec un intervalle de confiance à 95%

# 5. test d'égalité des moyennes
t.test(AGE_CLI~trsf_dev, alternative='two.sided',conf.level=0.95,var.equal=FALSE, data=dq2)

##
##  Welch Two Sample t-test
##
```

```
## data:  AGE_CLI by trsf_dev
## t = 93.404, df = 1399600, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  1.879916 1.960502
## sample estimates:
## mean in group N mean in group O
##        36.62959        34.70938

# analyse : AGE_CLI~trsf_dev, on rejette l'égalité des moyennes, la moyenne des Ages Clients des "Oui"
# est significativement différente de la moyenne des Ages Clients des "Non" avec un intervalle de confian
ce à 95%.
# le comportement de la transformation est différent on fonction de l'Age Client,
# CONCLUSION : la var explicative AGE_CLI discrimine bien la var à expliquer taux de transformation (Oui
/ Non)

################################
# PARTIE 3.1.2 : ANALYSE FACTORIELLE DISCRIMINANTE
################################
# A FAIRE

################################
# PARTIE 3.1.3 : ANALYSE REGRESSION LOGISTIQUE (RegLog)
################################

# mise en place d'un modèle de régression logistique
library(MASS)
modele_complet<-glm(trsf_dev ~.,data=dq2norm_y,family=binomial)
modele_complet

##
## Call:  glm(formula = trsf_dev ~ ., family = binomial, data = dq2norm_y)
##
## Coefficients:
## (Intercept)     C_AN_TTC    PUIS_FISC         COEF      AGE_CLI
##    -0.98966     -0.21440     -0.15202     -0.16488     -0.23867
##      NB_ENF      ANC_VEH
##    -0.07483      0.10822
##
## Degrees of Freedom: 2521043 Total (i.e. Null);  2521037 Residual
## Null Deviance:       2979000
## Residual Deviance: 2912000   AIC: 2912000

summary(modele_complet)

##
## Call:
## glm(formula = trsf_dev ~ ., family = binomial, data = dq2norm_y)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.4722  -0.8422  -0.7329   1.4129   2.9785
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.989661   0.001450 -682.31   <2e-16 ***
## C_AN_TTC    -0.214397   0.002458  -87.22   <2e-16 ***
## PUIS_FISC   -0.152017   0.001963  -77.46   <2e-16 ***
## COEF        -0.164879   0.002389  -69.01   <2e-16 ***
## AGE_CLI     -0.238674   0.002087 -114.34   <2e-16 ***
## NB_ENF      -0.074831   0.001838  -40.72   <2e-16 ***
## ANC_VEH      0.108216   0.001576   68.67   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2979417  on 2521043  degrees of freedom
## Residual deviance: 2911743  on 2521037  degrees of freedom
## AIC: 2911757
##
## Number of Fisher Scoring iterations: 4
```

```r
# modele selectionne avec procedure de selection (step) backward (méthode)
modele_selectionne<-step(modele_complet,direction="backward")

## Start:  AIC=2911757
## trsf_dev ~ C_AN_TTC + PUIS_FISC + COEF + AGE_CLI + NB_ENF + ANC_VEH
##
##             Df Deviance     AIC
## <none>          2911743 2911757
## - NB_ENF     1  2913424 2913436
## - ANC_VEH    1  2916435 2916447
## - COEF       1  2916548 2916560
## - PUIS_FISC  1  2918145 2918157
## - C_AN_TTC   1  2919698 2919710
## - AGE_CLI    1  2925302 2925314

modele_selectionne

##
## Call:  glm(formula = trsf_dev ~ C_AN_TTC + PUIS_FISC + COEF + AGE_CLI +
##     NB_ENF + ANC_VEH, family = binomial, data = dq2norm_y)
##
## Coefficients:
## (Intercept)     C_AN_TTC    PUIS_FISC         COEF      AGE_CLI
##    -0.98966     -0.21440     -0.15202     -0.16488     -0.23867
##      NB_ENF      ANC_VEH
##    -0.07483      0.10822
##
## Degrees of Freedom: 2521043 Total (i.e. Null);  2521037 Residual
## Null Deviance:       2979000
## Residual Deviance: 2912000   AIC: 2912000

summary(modele_selectionne)

##
## Call:
## glm(formula = trsf_dev ~ C_AN_TTC + PUIS_FISC + COEF + AGE_CLI +
##     NB_ENF + ANC_VEH, family = binomial, data = dq2norm_y)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.4722  -0.8422  -0.7329   1.4129   2.9785
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.989661   0.001450 -682.31   <2e-16 ***
## C_AN_TTC    -0.214397   0.002458  -87.22   <2e-16 ***
## PUIS_FISC   -0.152017   0.001963  -77.46   <2e-16 ***
## COEF        -0.164879   0.002389  -69.01   <2e-16 ***
## AGE_CLI     -0.238674   0.002087 -114.34   <2e-16 ***
## NB_ENF      -0.074831   0.001838  -40.72   <2e-16 ***
## ANC_VEH      0.108216   0.001576   68.67   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2979417  on 2521043  degrees of freedom
## Residual deviance: 2911743  on 2521037  degrees of freedom
## AIC: 2911757
##
## Number of Fisher Scoring iterations: 4

# modele selectionne avec procedure de selection (step) both (méthode)
modele_selectionne<-step(modele_complet,direction="both")

## Start:  AIC=2911757
## trsf_dev ~ C_AN_TTC + PUIS_FISC + COEF + AGE_CLI + NB_ENF + ANC_VEH
##
##             Df Deviance     AIC
## <none>          2911743 2911757
## - NB_ENF     1  2913424 2913436
## - ANC_VEH    1  2916435 2916447
```

```
## - COEF       1  2916548 2916560
## - PUIS_FISC  1  2918145 2918157
## - C_AN_TTC   1  2919698 2919710
## - AGE_CLI    1  2925302 2925314
```

```
modele_selectionne
```

```
##
## Call:  glm(formula = trsf_dev ~ C_AN_TTC + PUIS_FISC + COEF + AGE_CLI +
##     NB_ENF + ANC_VEH, family = binomial, data = dq2norm_y)
##
## Coefficients:
## (Intercept)     C_AN_TTC    PUIS_FISC         COEF      AGE_CLI
##    -0.98966     -0.21440     -0.15202     -0.16488     -0.23867
##      NB_ENF      ANC_VEH
##    -0.07483      0.10822
##
## Degrees of Freedom: 2521043 Total (i.e. Null);  2521037 Residual
## Null Deviance:       2979000
## Residual Deviance: 2912000   AIC: 2912000
```

```
summary(modele_selectionne)
```

```
##
## Call:
## glm(formula = trsf_dev ~ C_AN_TTC + PUIS_FISC + COEF + AGE_CLI +
##     NB_ENF + ANC_VEH, family = binomial, data = dq2norm_y)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.4722  -0.8422  -0.7329   1.4129   2.9785
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.989661   0.001450 -682.31   <2e-16 ***
## C_AN_TTC    -0.214397   0.002458  -87.22   <2e-16 ***
## PUIS_FISC   -0.152017   0.001963  -77.46   <2e-16 ***
## COEF        -0.164879   0.002389  -69.01   <2e-16 ***
## AGE_CLI     -0.238674   0.002087 -114.34   <2e-16 ***
## NB_ENF      -0.074831   0.001838  -40.72   <2e-16 ***
## ANC_VEH      0.108216   0.001576   68.67   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2979417  on 2521043  degrees of freedom
## Residual deviance: 2911743  on 2521037  degrees of freedom
## AIC: 2911757
##
## Number of Fisher Scoring iterations: 4
```

```
# methode 1 : matrice de confusion - indicateur de performance du classifieur - RegLog

# modele selectionne avec procedure de selection (step) both (méthode)
modele_selectionne<-step(modele_complet,direction="both")
```

```
## Start:  AIC=2911757
## trsf_dev ~ C_AN_TTC + PUIS_FISC + COEF + AGE_CLI + NB_ENF + ANC_VEH
##
##             Df Deviance     AIC
## <none>          2911743 2911757
## - NB_ENF     1  2913424 2913436
## - ANC_VEH    1  2916435 2916447
## - COEF       1  2916548 2916560
## - PUIS_FISC  1  2918145 2918157
## - C_AN_TTC   1  2919698 2919710
## - AGE_CLI    1  2925302 2925314
```

```
modele_selectionne
```

```
##
## Call:  glm(formula = trsf_dev ~ C_AN_TTC + PUIS_FISC + COEF + AGE_CLI +
##     NB_ENF + ANC_VEH, family = binomial, data = dq2norm_y)
##
## Coefficients:
## (Intercept)     C_AN_TTC    PUIS_FISC         COEF      AGE_CLI
##    -0.98966     -0.21440     -0.15202     -0.16488     -0.23867
##      NB_ENF      ANC_VEH
##    -0.07483      0.10822
##
## Degrees of Freedom: 2521043 Total (i.e. Null);  2521037 Residual
## Null Deviance:       2979000
## Residual Deviance: 2912000   AIC: 2912000
```

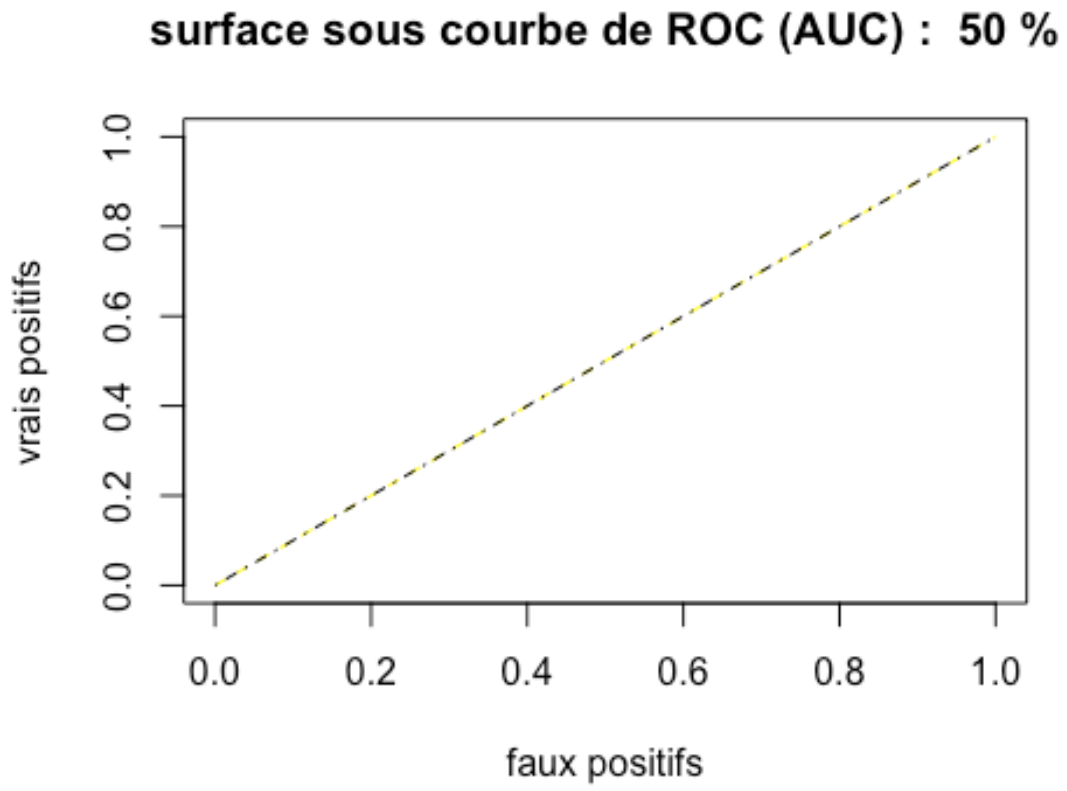
```r
summary(modele_selectionne)
```

```
##
## Call:
## glm(formula = trsf_dev ~ C_AN_TTC + PUIS_FISC + COEF + AGE_CLI +
##     NB_ENF + ANC_VEH, family = binomial, data = dq2norm_y)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.4722  -0.8422  -0.7329   1.4129   2.9785
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.989661   0.001450 -682.31   <2e-16 ***
## C_AN_TTC    -0.214397   0.002458  -87.22   <2e-16 ***
## PUIS_FISC   -0.152017   0.001963  -77.46   <2e-16 ***
## COEF        -0.164879   0.002389  -69.01   <2e-16 ***
## AGE_CLI     -0.238674   0.002087 -114.34   <2e-16 ***
## NB_ENF      -0.074831   0.001838  -40.72   <2e-16 ***
## ANC_VEH      0.108216   0.001576   68.67   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2979417  on 2521043  degrees of freedom
## Residual deviance: 2911743  on 2521037  degrees of freedom
## AIC: 2911757
##
## Number of Fisher Scoring iterations: 4
```

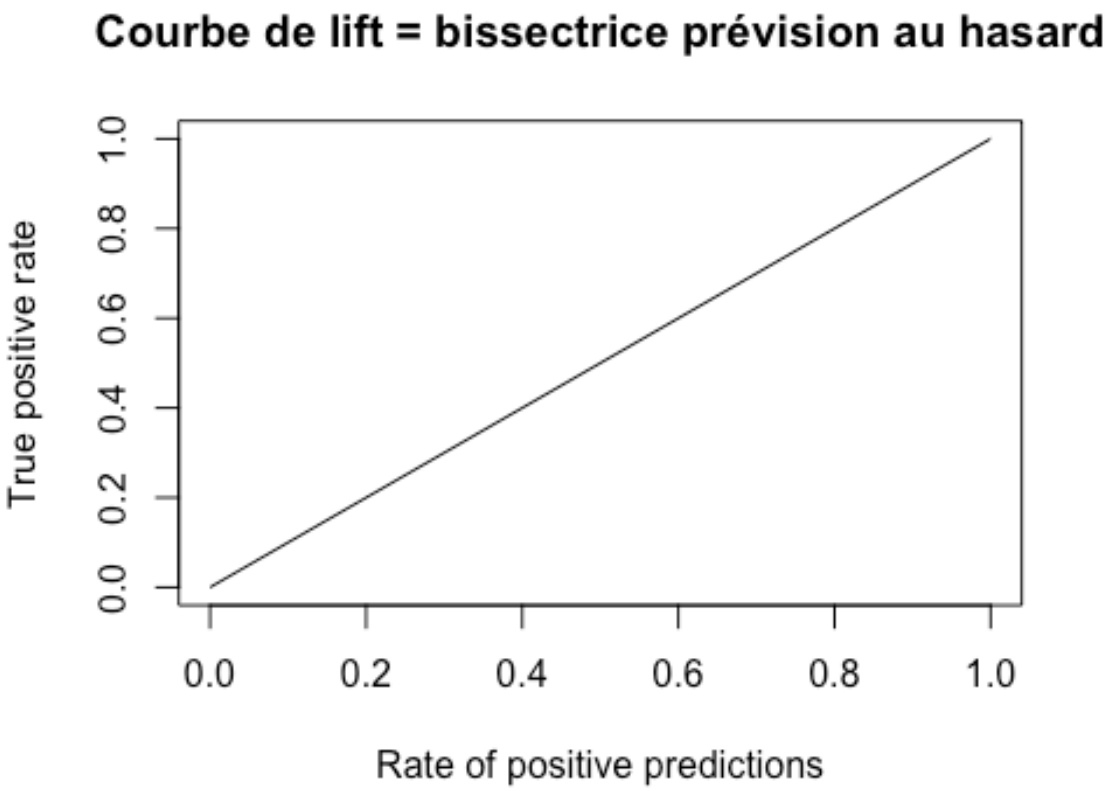
```r
library(caret)
confusionMatrix(data=prev, reference=dq2norm_y$trsf_dev)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction       N       O
##          N 1820577  700467
##          O       0       0
##
##                Accuracy : 0.7222
##                  95% CI : (0.7216, 0.7227)
##     No Information Rate : 0.7222
##     P-Value [Acc > NIR] : 0.5003
##
##                   Kappa : 0
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 1.0000
##             Specificity : 0.0000
##          Pos Pred Value : 0.7222
##          Neg Pred Value :    NaN
##              Prevalence : 0.7222
##          Detection Rate : 0.7222
##    Detection Prevalence : 1.0000
##       Balanced Accuracy : 0.5000
##
```

```
##        'Positive' Class : N
##

#confusionMatrix(data=prev, reference=dq2norm_y$trsf_dev,mode="prec_recall")

# analyse :
# valeur accuracy = 0.72
# valeur kappa = 0 (à regarder en classe déséquilibrée)
# le classifieur mal spécifié ou pas adapté à ce dataset - défaut des Xi discriminants dans le modèle Reg
Log ?


library(ROCR)
fr <- data.frame(score = c(prev),label =c(dq2norm_y$trsf_dev))
pred <- prediction(fr$score, fr$label)
perf <- performance(pred,"tpr", "fpr")
y.auc.perf= performance(pred, measure="auc",x.measure="cutoff")
auc=round(unlist(y.auc.perf@y.values),2)*100
print(auc)

## [1] 50

plot(perf, col="yellow", lty=2, pch=32,
     main = paste("surface sous courbe de ROC (AUC) : ",auc, "%"),
     xlab = "faux positifs",
     ylab = "vrais positifs")

segments(0,0,1,1,lty=4)
```




```
#graphics.off()


# courbe de lift
lift<- performance(pred,"tpr","rpp")
plot(lift, main='Courbe de lift = bissectrice prévision au hasard')
```




```
# le modele RegLog produit les résultats identiques au modèle lda.

# sélection à la main
modele_select2<-glm(trsf_dev ~C_AN_TTC+AGE_CLI,data=dq2norm_y,family=binomial)
summary(modele_select2)

##
## Call:
## glm(formula = trsf_dev ~ C_AN_TTC + AGE_CLI, family = binomial,
##     data = dq2norm_y)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.1101  -0.8455  -0.7443   1.4330   3.1465
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.985035   0.001445  -681.8   <2e-16 ***
## C_AN_TTC    -0.373313   0.001804  -206.9   <2e-16 ***
## AGE_CLI     -0.273062   0.001620  -168.6   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2979417  on 2521043  degrees of freedom
## Residual deviance: 2923050  on 2521041  degrees of freedom
## AIC: 2923056
##
## Number of Fisher Scoring iterations: 4
```

CODE R	RESULTATS
Modèle ACP	
Modèle Analyse Factorielle Discriminante	

quanti_acp_afd_dlp_v170208.R

lepicaut

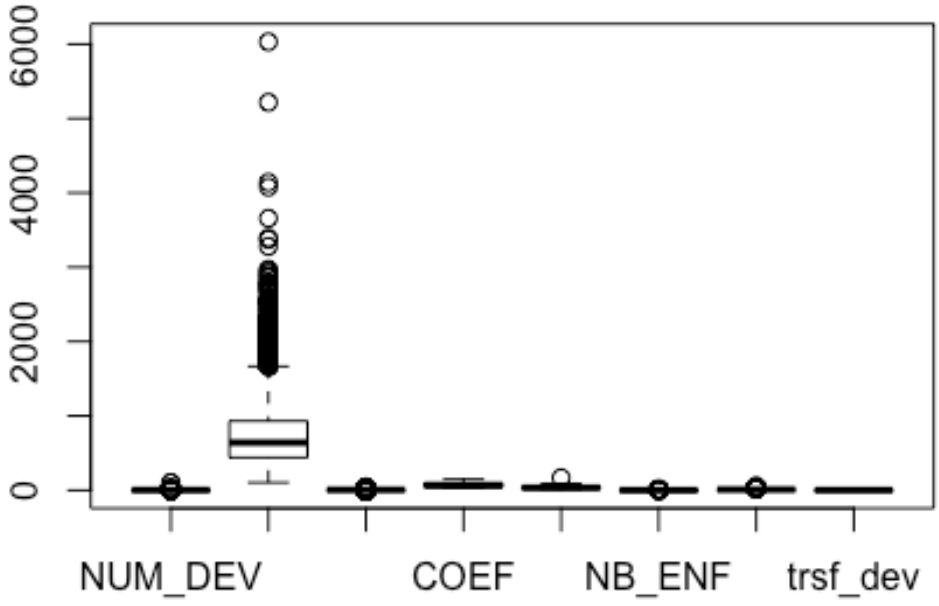
Sun Aug 27 23:10:54 2017

```
####################
# travaux : illustratio graphique avant spécification du modèle
# script 1 : ACP
# script 2 : Analyse Factorielle Discriminante (Geometrique)
###################

############
# compile rapport : quanti_acp_afd_dlp_v170208.R - en date du 27 aout17
############

# 0. nettoyer historique
rm(list = ls(all = TRUE)) # make sure previous work is clear
ls()

## character(0)

tic <- Sys.time()

# 1. importing and exploring data

path<-"/Users/lepicaut/Desktop/sos/MATMUT/0_0_matmut/fevrier_2017/00000_code"
#path <- "/Users/lepicaut/Desktop/00000_code"

# 2. set working directory
setwd(path)
dv<- read.csv("devis_trsf_2015.csv", header=TRUE, sep=";")


# chargement des librairies R
library(plyr)
library(MASS)
library(corrplot)
library(lattice)
library(caret)

## Loading required package: ggplot2

library(ROCR)

## Loading required package: gplots

##
## Attaching package: 'gplots'

## The following object is masked from 'package:stats':
##
##     lowess

library(ggplot2)
library(FactoMineR)
library(ade4)

## Warning: package 'ade4' was built under R version 3.4.1

##
## Attaching package: 'ade4'
```

```
## The following object is masked from 'package:FactoMineR':
##
##     reconst

library(colorspace)
library(e1071)
library(tidyr)

## Warning: package 'tidyr' was built under R version 3.4.1

library(rmarkdown)

################################
# PARTIE 1. PREPROCESSING DATA
################################


#str(dv)

#names(dv)

dv1<-dv[,-3:-5]

dv2<-dv1[,-2]

dv3<-dv2[,-3:-5]

dv4<-dv3[,-4:-5]

dv5<-dv4[,-4]

dv6<-dv5[,-5:-8]

dv7<-dv6[,-7:-14]

dv8<-dv7[,-7:-19]

dv9<-dv8[,-7:-9]

dv10<-dv9[,-8:-9]

dv11<-dv10[,-9:-11]

dv12<-dv11[,-14:-15]

#names(dv12)

which(names(dv12) %in% c('NB_DEVI_ARCH_MRSQ','DAT_ADH'))

## [1] 15 47

dv13<-dv12[,-15:-47]

#names(dv13)

which(names(dv13) %in% c('AGE_E1','TYP_REAL'))

## [1] 24 32

dv14<-dv13[,-24:-32]

which(names(dv14) %in% c('CD_ADMIN'))

## [1] 21

dv15<-dv14[,-21]

which(names(dv15) %in% c('MARQ_VEH'))

## [1] 9
```

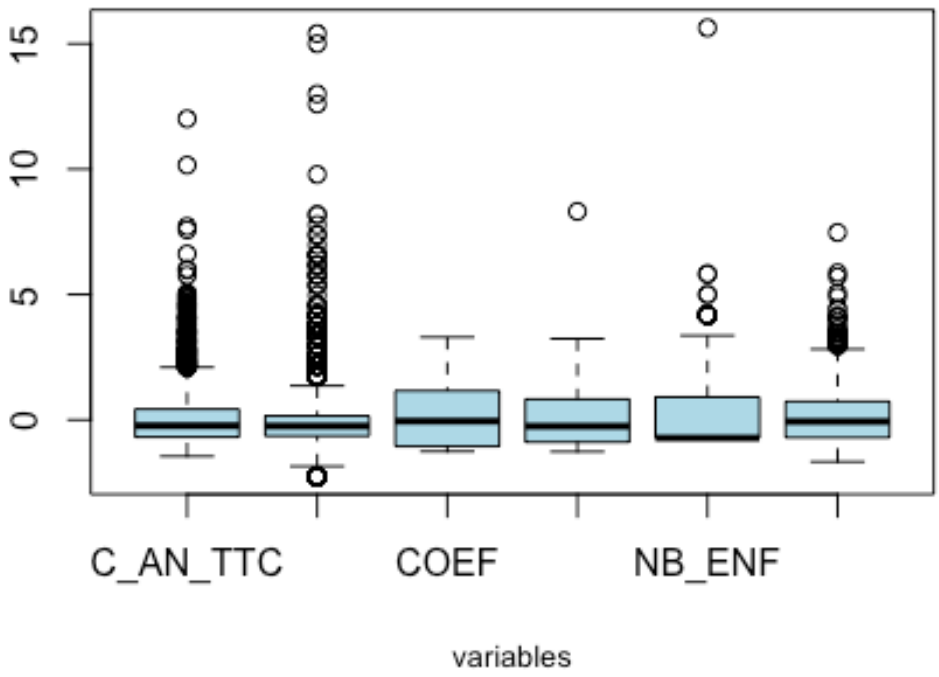
```
dv16<-dv15[,-9]

#summary(dv16)

#dim(dv16)

attach(dv16)

dv17=na.omit(dv16)

#summary(dv17)

(table(dv17$trsf_dev)/nrow(dv17)*100)

##
##        N        O
## 73.31072 26.68928

#dim(dv17)

sum(is.na(dv17))

## [1] 0

#str(dv17)

# passage de > 600 000 lignes à 1000 lignes
#dv18=dv17[sample(1:nrow(dv17),614048),]

# passage de > 614 048 lignes à 6140 lignes
# cad poids échantillon = 0.99% x échantillon total observé
dv18=dv17[sample(1:nrow(dv17),6140),]

#dim(dv18)

(table(dv18$trsf_dev)/nrow(dv18)*100)

##
##       N       O
## 72.9316 27.0684

#summary(dv18)

#str(dv18)

# transfrome C_AN_TTC en var numerique
dv18$C_AN_TTC = gsub(",",".",dv18$C_AN_TTC)
dv18$C_AN_TTC = gsub(" ","",dv18$C_AN_TTC)
dv18$C_AN_TTC = as.double(dv18$C_AN_TTC)

#str(dv18)

#dv18$C_AN_TTC[1:100]

# IDENTIFIER LES QUANTI
library(plyr)
type_var_num = colwise(is.numeric)
var_num=names(dv18)[which(type_var_num(dv18)==TRUE)]

dv18Quanti = dv18[, c(var_num)]
str(dv18Quanti)

## 'data.frame':    6140 obs. of  9 variables:
##  $ ID_CLIENT: int  212633904 212116297 212379562 212689272 212634779 212184764 211990558 212623889 212
452218 212503632 ...
##  $ NUM_DEV  : int  5 1 1 2 1 1 5 1 2 1 ...
##  $ C_AN_TTC : num  208 454 1109 540 474 ...
##  $ COEFF    : int  35 38 100 40 65 40 35 100 40 40 ...
##  $ PUIS_FISC: int  1 6 5 5 5 4 4 5 4 3 ...
##  $ COEF     : int  35 38 100 40 65 40 35 100 40 40 ...
##  $ AGE_CLI  : int  42 46 19 67 22 85 52 33 51 66 ...
```

```
##  $ NB_ENF   : int  1 4 0 1 0 1 2 1 2 0 ...
##  $ ANC_VEH  : int  7 12 11 8 14 15 5 0 2 0 ...

etiq<-dv18[,-2:-21]
etiq<-etiq[,-3]
str(etiq)

## 'data.frame':    6140 obs. of  2 variables:
##  $ ID_CLIENT: int  212633904 212116297 212379562 212689272 212634779 212184764 211990558 212623889 212
452218 212503632 ...
##  $ trsf_dev : Factor w/ 2 levels "N","O": 1 1 1 1 2 1 1 1 1 1 ...

dq=merge(dv18Quanti,etiq,by="ID_CLIENT")
str(dq)

## 'data.frame':    6300 obs. of  10 variables:
##  $ ID_CLIENT: int  211978250 211978340 211978469 211978486 211978552 211978552 211978552 211978552 211
978645 211978656 ...
##  $ NUM_DEV  : int  1 2 2 1 6 6 5 5 1 3 ...
##  $ C_AN_TTC : num  1438 1231 389 748 316 ...
##  $ COEFF    : int  100 100 40 65 42 42 42 42 132 47 ...
##  $ PUIS_FISC: int  4 4 8 5 3 3 3 3 6 6 ...
##  $ COEF     : int  100 100 40 65 42 42 42 42 132 47 ...
##  $ AGE_CLI  : int  26 26 44 21 27 27 27 27 23 40 ...
##  $ NB_ENF   : int  0 0 2 0 0 0 0 0 0 3 ...
##  $ ANC_VEH  : int  2 0 14 10 17 17 17 17 16 16 ...
##  $ trsf_dev : Factor w/ 2 levels "N","O": 1 1 1 2 1 1 1 1 1 1 ...

summary(dq)

##    ID_CLIENT            NUM_DEV           C_AN_TTC          COEFF
##  Min.   :211978250   Min.   :  1.000   Min.   : 103.0   Min.   : 35.00
##  1st Qu.:212234937   1st Qu.:  1.000   1st Qu.: 440.3   1st Qu.: 40.00
##  Median :212381888   Median :  2.000   Median : 640.3   Median : 65.00
##  Mean   :212397440   Mean   :  2.114   Mean   : 742.4   Mean   : 66.23
##  3rd Qu.:212571577   3rd Qu.:  3.000   3rd Qu.: 930.5   3rd Qu.: 95.00
##  Max.   :212760812   Max.   :102.000   Max.   :6031.3   Max.   :148.00
##    PUIS_FISC          COEF           AGE_CLI           NB_ENF
##  Min.   : 0.00   Min.   : 35.00   Min.   : 17.00   Min.   : 0.0000
##  1st Qu.: 4.00   1st Qu.: 40.00   1st Qu.: 23.00   1st Qu.: 0.0000
##  Median : 5.00   Median : 65.00   Median : 33.00   Median : 0.0000
##  Mean   : 5.61   Mean   : 66.23   Mean   : 36.99   Mean   : 0.8871
##  3rd Qu.: 6.00   3rd Qu.: 95.00   3rd Qu.: 50.00   3rd Qu.: 2.0000
##  Max.   :44.00   Max.   :148.00   Max.   :168.00   Max.   :20.0000
##     ANC_VEH       trsf_dev
##  Min.   :-1.000   N:4599
##  1st Qu.: 5.000   O:1701
##  Median : 9.000
##  Mean   : 9.413
##  3rd Qu.:14.000
##  Max.   :56.000

dq1=dq[,-4]
#str(dq1)

dq2=dq1[,-1]
#str(dq2)

# boxplot sur données non normalisées
boxplot(dq2)
```



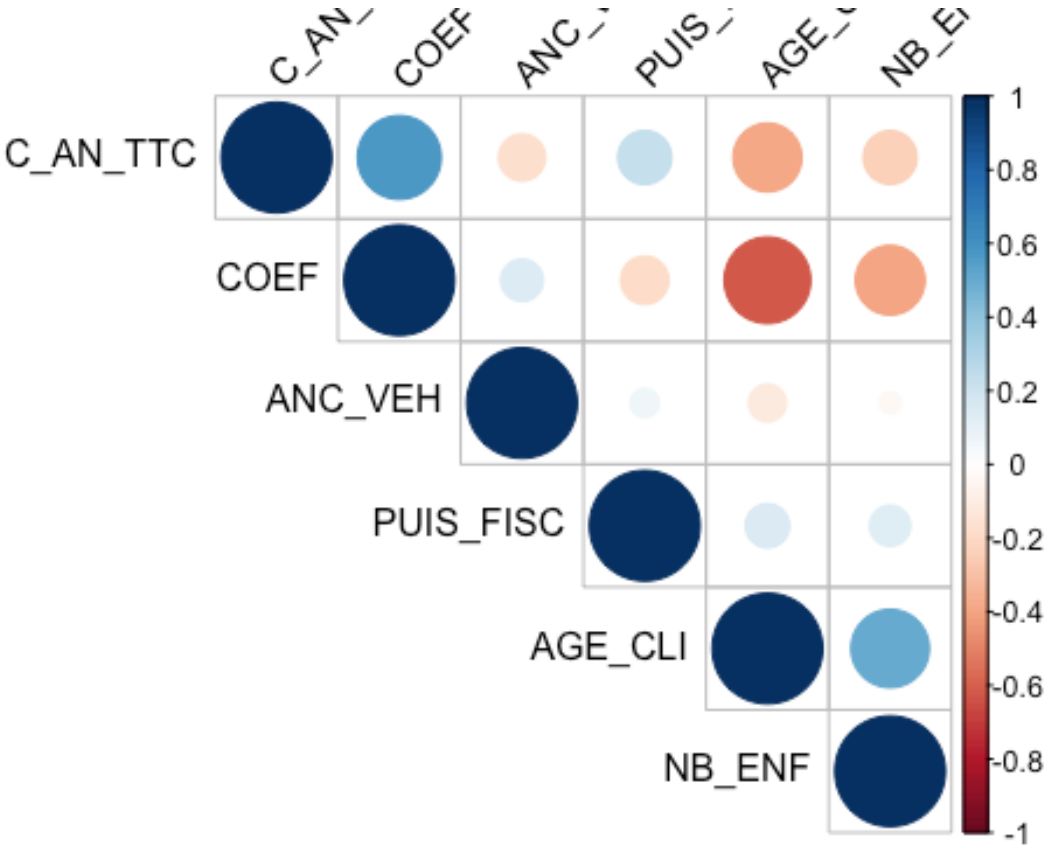

```
################################
# PARTIE 2. ANALYSE DESCRIPTIVE
################################

# lecture du graphe : sur ces données non normalisées, on ne peut rien dire.

dq2norm=as.data.frame(scale(dq2[,2:7]))
sum(is.na(dq2$trsf_dev))

## [1] 0

str(dq2norm)

## 'data.frame':    6300 obs. of  6 variables:
##  $ C_AN_TTC : num  1.5799 1.11 -0.8016 0.0134 -0.9674 ...
##  $ PUIS_FISC: num  -0.646 -0.646 0.959 -0.245 -1.047 ...
##  $ COEF     : num  1.3618 1.3618 -1.0578 -0.0496 -0.9772 ...
##  $ AGE_CLI  : num  -0.697 -0.697 0.445 -1.015 -0.634 ...
##  $ NB_ENF   : num  -0.726 -0.726 0.91 -0.726 -0.726 ...
##  $ ANC_VEH  : num  -1.1891 -1.5099 0.7358 0.0942 1.217 ...

summary(dq2norm)

##     C_AN_TTC         PUIS_FISC            COEF             AGE_CLI
##  Min.   :-1.4516   Min.   :-2.2509   Min.   :-1.25946   Min.   :-1.2685
##  1st Qu.:-0.6858   1st Qu.:-0.6460   1st Qu.:-1.05782   1st Qu.:-0.8877
##  Median :-0.2318   Median :-0.2447   Median :-0.04963   Median :-0.2532
##  Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.00000   Mean   : 0.0000
##  3rd Qu.: 0.4272   3rd Qu.: 0.1565   3rd Qu.: 1.16020   3rd Qu.: 0.8256
##  Max.   :12.0074   Max.   :15.4030   Max.   : 3.29757   Max.   : 8.3133
##      NB_ENF           ANC_VEH
##  Min.   :-0.7257   Min.   :-1.67028
##  1st Qu.:-0.7257   1st Qu.:-0.70786
##  Median :-0.7257   Median :-0.06625
##  Mean   : 0.0000   Mean   : 0.00000
##  3rd Qu.: 0.9104   3rd Qu.: 0.73577
##  Max.   :15.6352   Max.   : 7.47270

class(dq2norm)
```

```
## [1] "data.frame"

boxplot(dq2norm, col="lightblue",xlab="variables",cex.lab=0.8,cex.main=0.8)
```

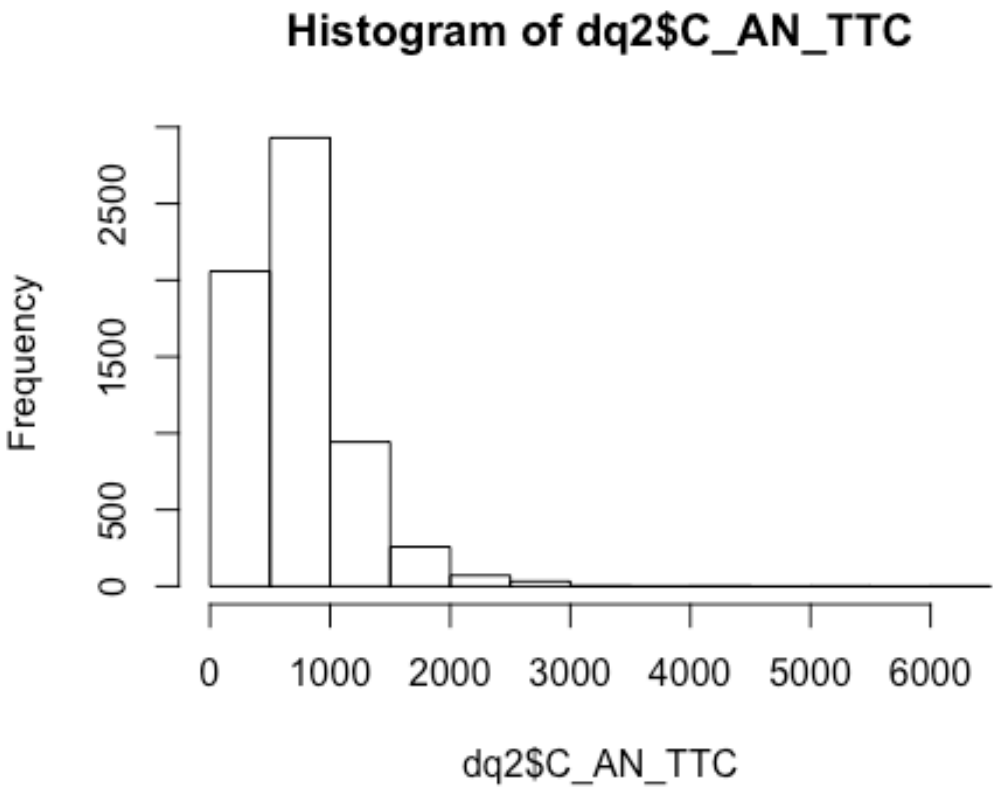



```
# mise en place d'un modèle d'analyse discriminante linéaire avec le package MASS
dq2norm_y = dq2norm
dq2norm_y$trsf_dev = dq2$trsf_dev
str(dq2norm_y)

## 'data.frame':    6300 obs. of  7 variables:
##  $ C_AN_TTC : num  1.5799 1.11 -0.8016 0.0134 -0.9674 ...
##  $ PUIS_FISC: num  -0.646 -0.646 0.959 -0.245 -1.047 ...
##  $ COEF     : num  1.3618 1.3618 -1.0578 -0.0496 -0.9772 ...
##  $ AGE_CLI  : num  -0.697 -0.697 0.445 -1.015 -0.634 ...
##  $ NB_ENF   : num  -0.726 -0.726 0.91 -0.726 -0.726 ...
##  $ ANC_VEH  : num  -1.1891 -1.5099 0.7358 0.0942 1.217 ...
##  $ trsf_dev : Factor w/ 2 levels "N","O": 1 1 1 2 1 1 1 1 1 1 ...

library(MASS)
lda(dq2norm_y$trsf_dev ~.,data=dq2norm_y)

## Call:
## lda(dq2norm_y$trsf_dev ~ ., data = dq2norm_y)
##
## Prior probabilities of groups:
##    N    O
## 0.73 0.27
##
## Group means:
##     C_AN_TTC   PUIS_FISC        COEF     AGE_CLI      NB_ENF     ANC_VEH
## N  0.0533739  0.05012997  0.01008684  0.05853924  0.03415726 -0.01267670
## O -0.1443072 -0.13553660 -0.02727183 -0.15827277 -0.09235111  0.03427404
##
## Coefficients of linear discriminants:
##                   LD1
## C_AN_TTC  -0.61157093
## PUIS_FISC -0.24303049
## COEF      -0.41679410
## AGE_CLI   -0.94838970
```

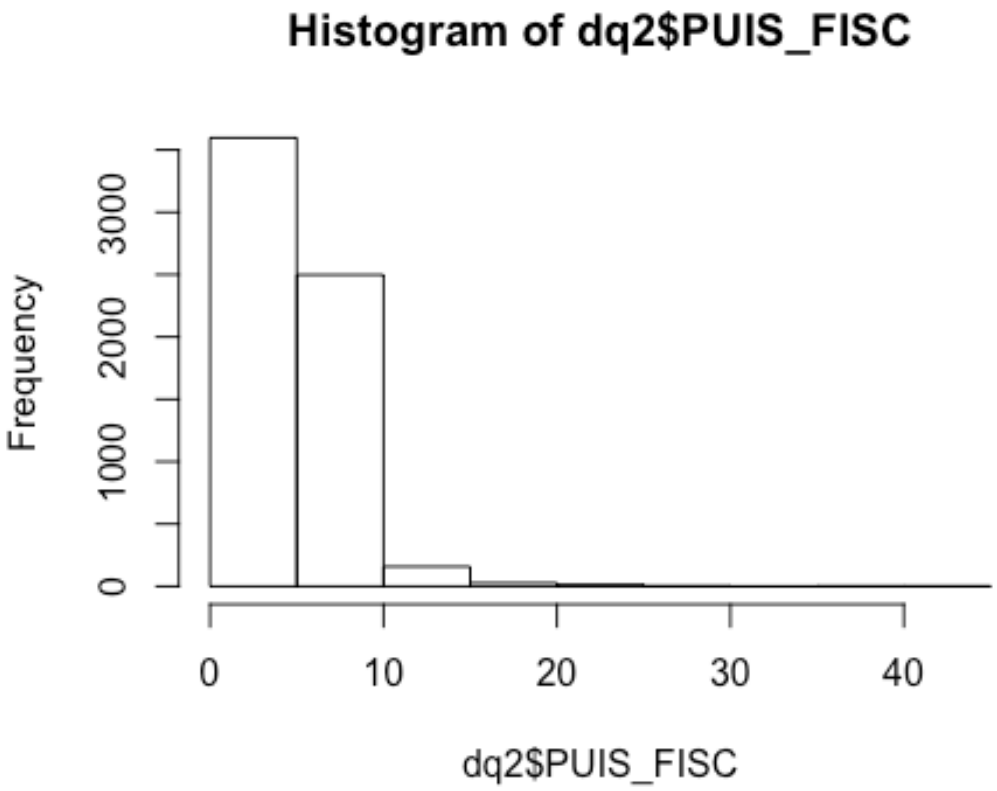
```
## NB_ENF     -0.12573192
## ANC_VEH    -0.02644307

getwd()

## [1] "/Users/lepicaut/Desktop/sos/MATMUT/0_0_matmut/fevrier_2017/00000_code"

write.csv2(dq2norm_y,file='dq2norm_y$trsf_dev.csv')

head(dq2norm_y)

##      C_AN_TTC  PUIS_FISC       COEF    AGE_CLI     NB_ENF     ANC_VEH
## 1  1.57991173 -0.6459718  1.3618414 -0.6973609 -0.7257230 -1.18907125
## 2  1.10995959 -0.6459718  1.3618414 -0.6973609 -0.7257230 -1.50987759
## 3 -0.80163321  0.9589271 -1.0578217  0.4448299  0.9103675  0.73576680
## 4  0.01340457 -0.2447471 -0.0496287 -1.0146361 -0.7257230  0.09415412
## 5 -0.96736513 -1.0471966 -0.9771662 -0.6339058 -0.7257230  1.21697631
## 6 -0.96736513 -1.0471966 -0.9771662 -0.6339058 -0.7257230  1.21697631
##   trsf_dev
## 1        N
## 2        N
## 3        N
## 4        O
## 5        N
## 6        N

head(dq2norm_y$trsf_dev)

## [1] N N N O N N
## Levels: N O

# corrélation linéaire entre les variables explicatives
library(corrplot)
corrplot(cor(dq2norm),type="upper", order="hclust",tl.col="black", tl.srt=45)
```



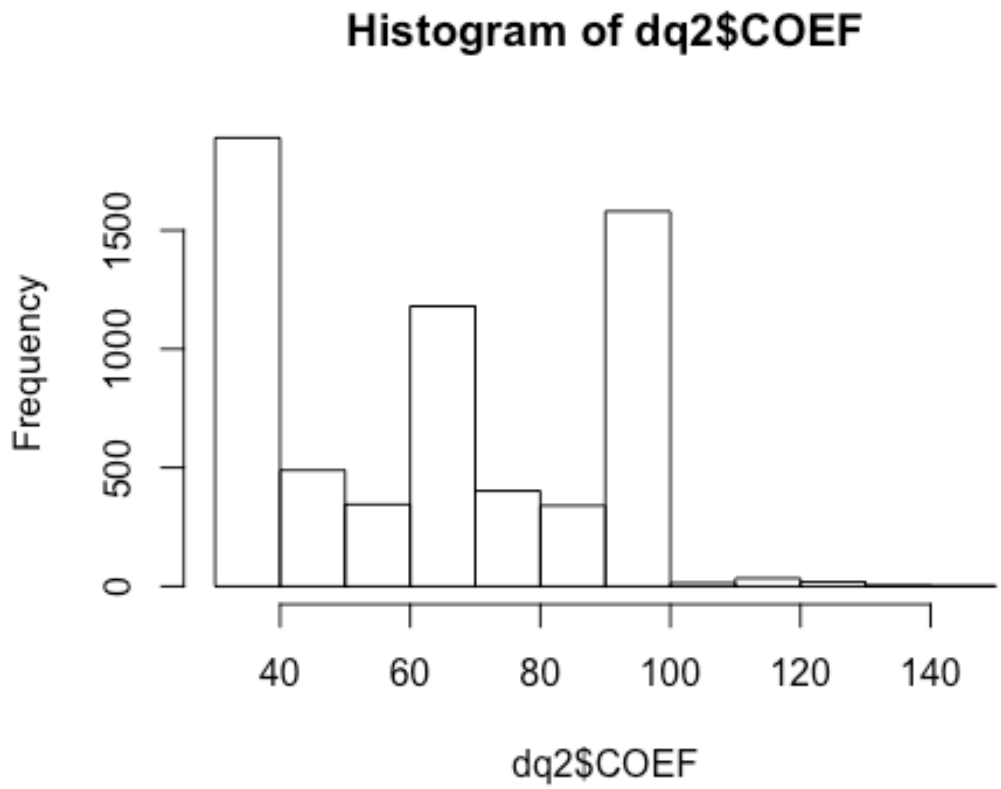
```r
hist(dq2$C_AN_TTC)
```



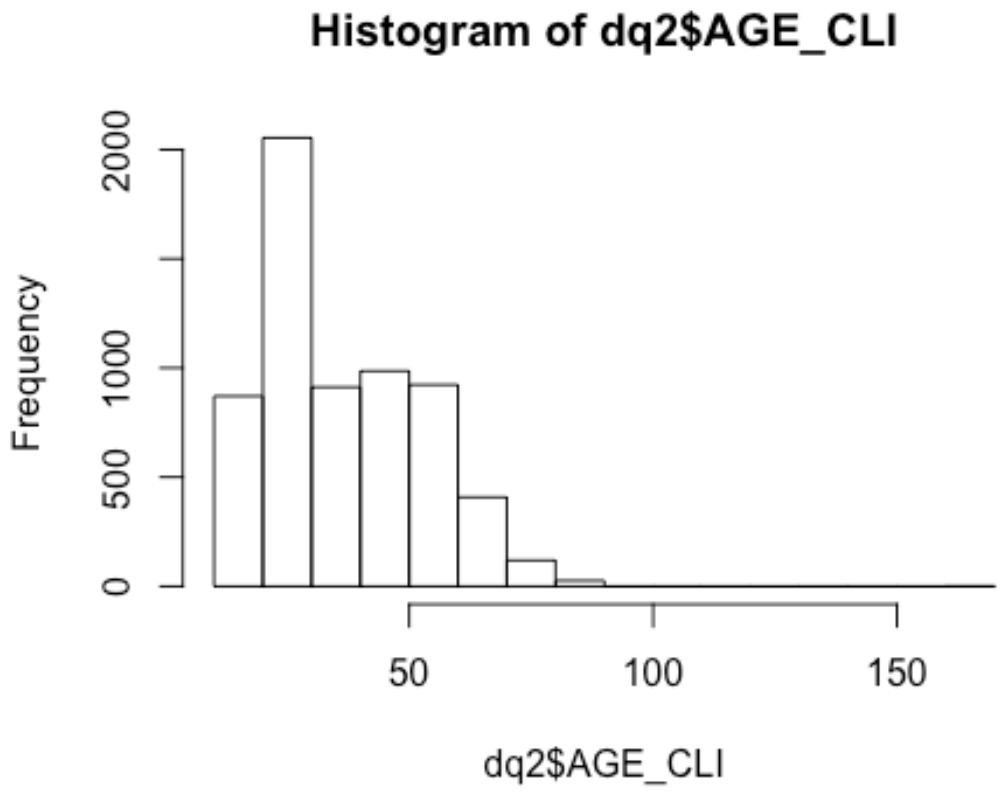

```r
hist(dq2$PUIS_FISC)
```



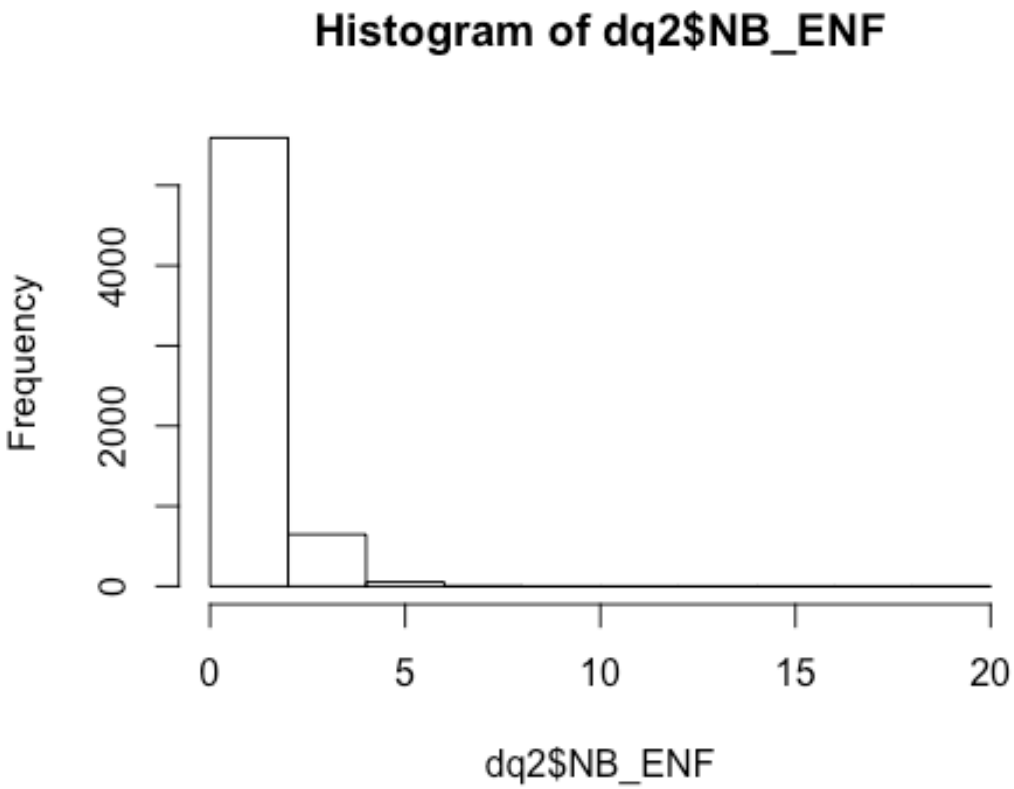
```r
hist(dq2$COEF)
```



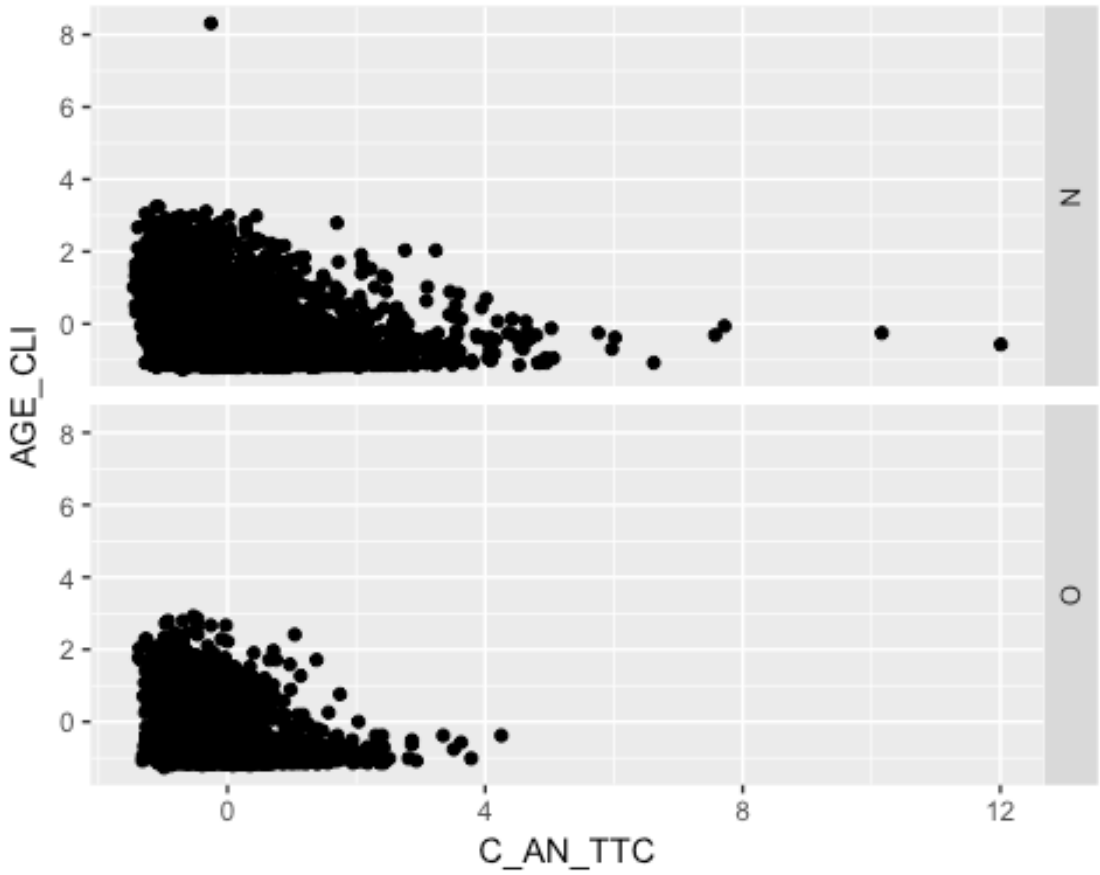

```r
hist(dq2$AGE_CLI)
```



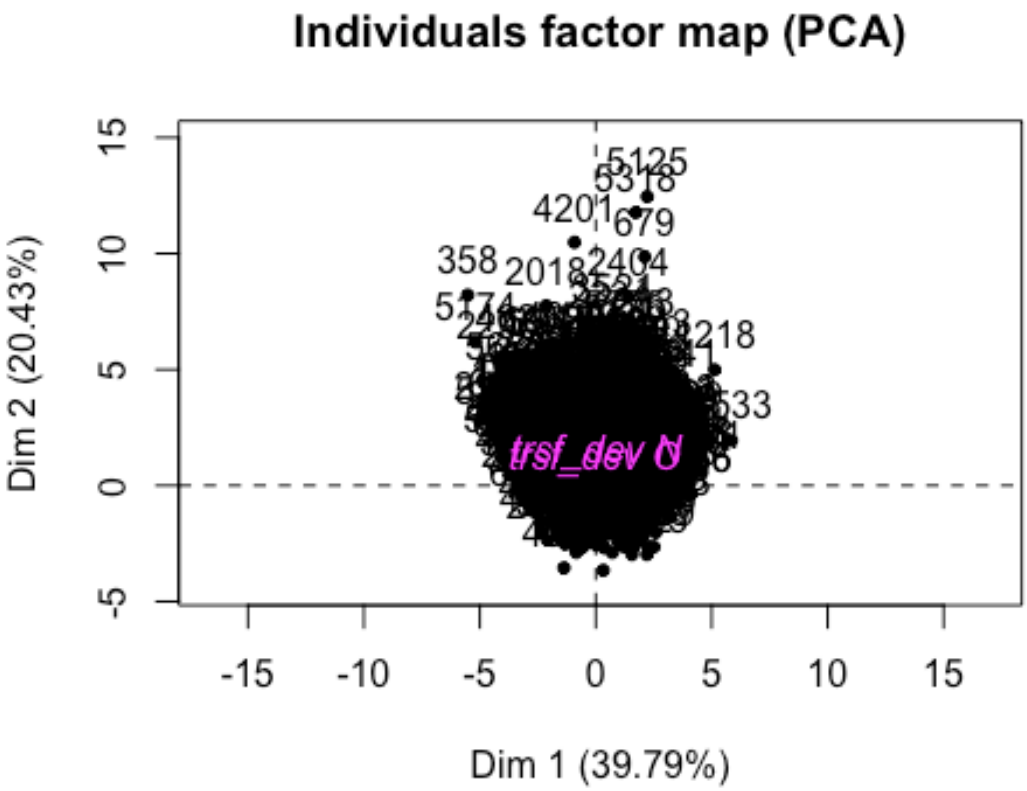
```r
hist(dq2$NB_ENF)
```

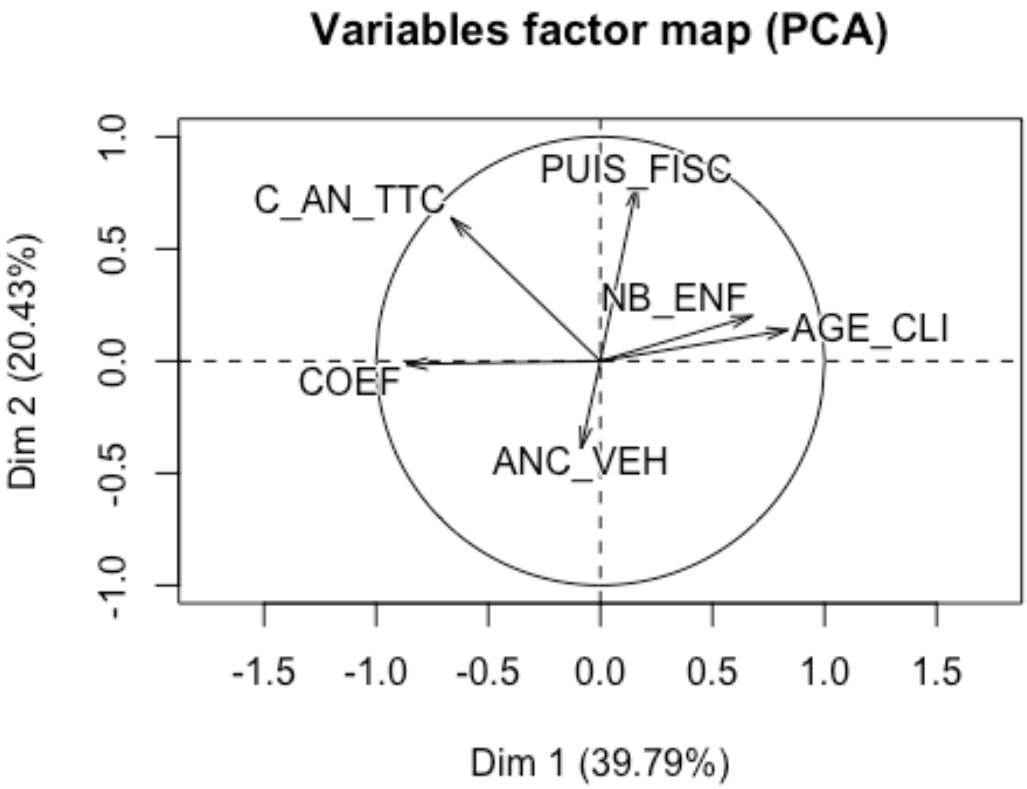



```r
################################
# PARTIE 3. MODELISATION DES CLASSIFIEURS
################################

################################
# PARTIE 3.1. : SELECTION DU MODELE (Feature Ingineering)
################################

################################
# PARTIE 3.1.1 : ANALYSE LINEAIRE DISCRIMINANTE (LDA)
################################

# mise en place d'un modèle d'analyse linéaire discriminante avec le package MASS
# modele complet : 6 var quanti + var etiquette
dq2norm_y = dq2norm
dq2norm_y$trsf_dev = dq2$trsf_dev
str(dq2norm_y)

## 'data.frame':    6300 obs. of  7 variables:
##  $ C_AN_TTC : num  1.5799 1.11 -0.8016 0.0134 -0.9674 ...
##  $ PUIS_FISC: num  -0.646 -0.646 0.959 -0.245 -1.047 ...
##  $ COEF     : num  1.3618 1.3618 -1.0578 -0.0496 -0.9772 ...
##  $ AGE_CLI  : num  -0.697 -0.697 0.445 -1.015 -0.634 ...
##  $ NB_ENF   : num  -0.726 -0.726 0.91 -0.726 -0.726 ...
##  $ ANC_VEH  : num  -1.1891 -1.5099 0.7358 0.0942 1.217 ...
##  $ trsf_dev : Factor w/ 2 levels "N","O": 1 1 1 2 1 1 1 1 1 1 ...

################################
# VISUALISATION GRAPHIQUE DES ELEMENTS DE CLASSIFICATION
################################

# VISUALISATION DES CLASSES QUALIFIEES AVEC 2 VAR DISCRIMINANTES
library(ggplot2)
qplot(C_AN_TTC, AGE_CLI, data=dq2norm_y, facets= trsf_dev~.)
```




```
# ANALYSE
# croisement var cotisation & âge pour expliquer l'étiquette Oui / Non devis transformé

# analyse en diagonale :
# analyse 1 : age=vieux en relation avec cotisation = élevée alors étiquette = Non
# analyse 2 : age=jeune en relation avec cotisation = élevée alors étiquette = Non
# analyse 1 & 2 : connue / inconnue sur les cibles jeunes et vieux.


### VISUALISATION DESCRIPTIVE = ANALYSE EN COMPOSANTE PRINCIPALE
# PACKAGE FactoMineR
# ACP (7 variables) : représentation graphique des vars et des individus
library(FactoMineR)
res.pca<-PCA(dq2norm_y, quali.sup=7)
```





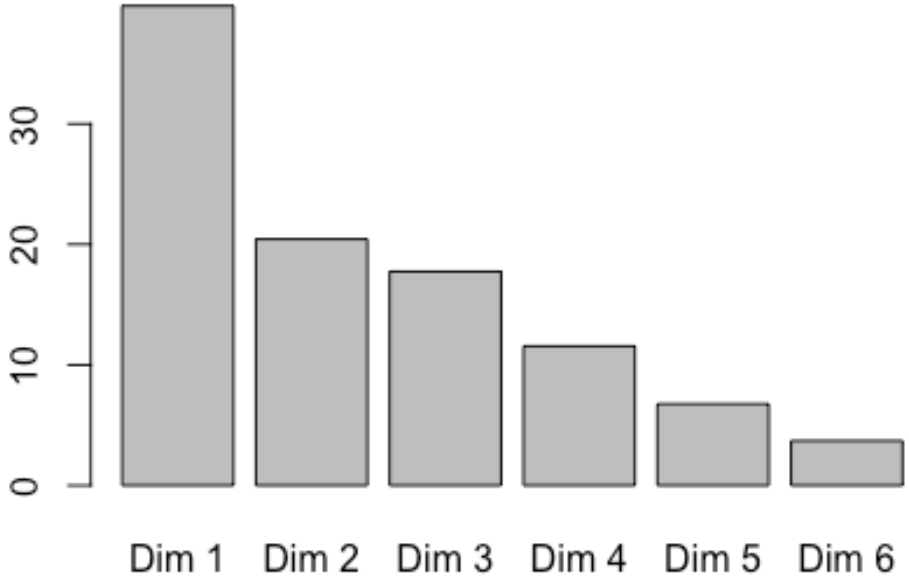

```
summary(res.pca)

##
## Call:
## PCA(X = dq2norm_y, quali.sup = 7)
##
##
```

```
## Eigenvalues
##                        Dim.1   Dim.2   Dim.3   Dim.4   Dim.5   Dim.6
## Variance               2.387   1.226   1.065   0.694   0.406   0.222
## % of var.             39.789  20.432  17.751  11.564   6.765   3.700
## Cumulative % of var.  39.789  60.221  77.972  89.536  96.300 100.000
##
## Individuals (the 10 first)
##                Dist    Dim.1    ctr   cos2    Dim.2    ctr   cos2    Dim.3
## 1          |  2.683 | -2.143  0.031  0.638 |  0.636  0.005  0.056 | -1.385
## 2          |  2.607 | -1.923  0.025  0.544 |  0.478  0.003  0.034 | -1.631
## 3          |  2.062 |  1.640  0.018  0.633 |  0.190  0.000  0.008 |  1.129
## 4          |  1.276 | -0.877  0.005  0.473 | -0.457  0.003  0.128 | -0.031
## 5          |  2.323 |  0.125  0.000  0.003 | -1.921  0.048  0.684 |  0.553
## 6          |  2.323 |  0.125  0.000  0.003 | -1.921  0.048  0.684 |  0.553
## 7          |  2.252 |  0.045  0.000  0.000 | -1.814  0.043  0.649 |  0.540
## 8          |  2.252 |  0.045  0.000  0.000 | -1.814  0.043  0.649 |  0.540
## 9          |  3.093 | -2.445  0.040  0.625 | -0.382  0.002  0.015 |  1.123
## 10         |  2.354 |  1.634  0.018  0.482 | -0.418  0.002  0.032 |  1.098
##               ctr   cos2
## 1           0.029  0.266 |
## 2           0.040  0.391 |
## 3           0.019  0.300 |
## 4           0.000  0.001 |
## 5           0.005  0.057 |
## 6           0.005  0.057 |
## 7           0.004  0.057 |
## 8           0.004  0.057 |
## 9           0.019  0.132 |
## 10          0.018  0.218 |
##
## Variables
##               Dim.1    ctr   cos2    Dim.2    ctr   cos2    Dim.3    ctr
## C_AN_TTC    | -0.665 18.541  0.443 |  0.638 33.250  0.408 | -0.071  0.475
## PUIS_FISC   |  0.162  1.104  0.026 |  0.779 49.545  0.607 |  0.499 23.407
## COEF        | -0.866 31.442  0.751 | -0.014  0.016  0.000 |  0.057  0.303
## AGE_CLI     |  0.836 29.306  0.700 |  0.139  1.574  0.019 | -0.050  0.239
## NB_ENF      |  0.679 19.303  0.461 |  0.202  3.324  0.041 |  0.058  0.313
## ANC_VEH     | -0.085  0.303  0.007 | -0.388 12.292  0.151 |  0.895 75.263
##              cos2
## C_AN_TTC    0.005 |
## PUIS_FISC   0.249 |
## COEF        0.003 |
## AGE_CLI     0.003 |
## NB_ENF      0.003 |
## ANC_VEH     0.802 |
##
## Supplementary categories
##                Dist    Dim.1   cos2 v.test    Dim.2   cos2 v.test    Dim.3
## trsf_dev N |  0.101 |  0.024  0.056  2.029 |  0.084  0.690  9.896 |  0.009
## trsf_dev O |  0.273 | -0.065  0.056 -2.029 | -0.227  0.690 -9.896 | -0.025
##              cos2 v.test
## trsf_dev N  0.008  1.161 |
## trsf_dev O  0.008 -1.161 |
```

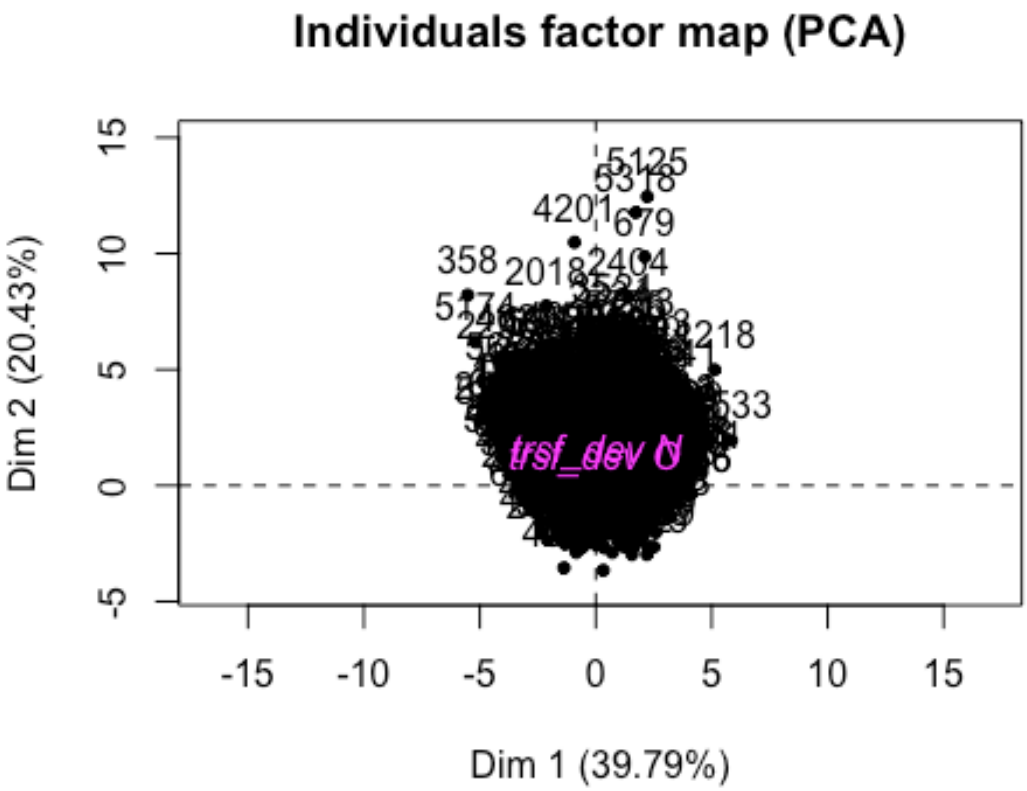
```
res.pca$eig[,2]
```

```
## [1] 39.788872 20.431724 17.750985 11.563993  6.764622  3.699804
```

```
barplot(res.pca$eig[,2],names=paste("Dim",1:nrow(res.pca$eig)))
```




```
round(res.pca$eig[1:4,],2)

##        eigenvalue percentage of variance cumulative percentage of variance
## comp 1       2.39                  39.79                             39.79
## comp 2       1.23                  20.43                             60.22
## comp 3       1.07                  17.75                             77.97
## comp 4       0.69                  11.56                             89.54

#res.pca$ind
#res.pca$var
#res.pca$quali.sup

#plot(res.pca, choix="ind",habillage=7,axes=1:2,cex=0.7)
#plot(res.pca, choix="ind",habillage=7,axes=3:4,cex=0.7)
#plot(res.pca, choix="var",habillage=7,axes=1:2,new.plot=TRUE)
#plot(res.pca, choix="var",habillage=7,axes=3:4,new.plot=TRUE)
#plot(res.pca, choix="var",habillage=7,axes=1:2,new.plot=F)
#plot(res.pca, choix="var",habillage=7,axes=3:4,new.plot=F)
#dev.off()
plot.PCA(res.pca)
```

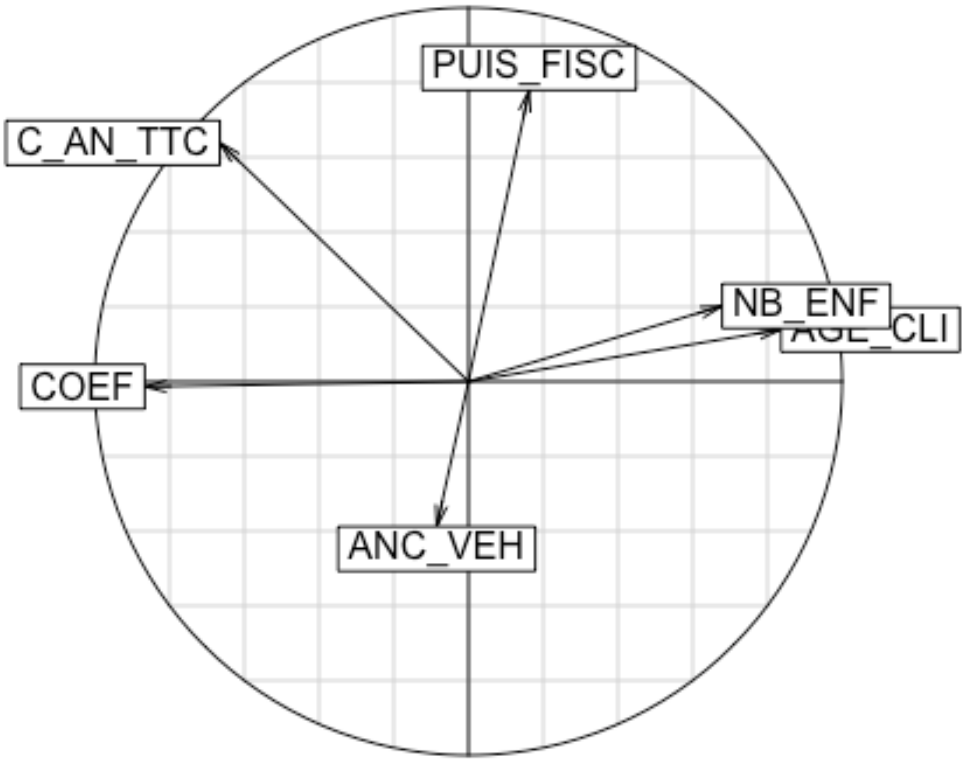



```
# graphe(s) des individus amélioré(s)
#plot(res.pca, autoLab="auto")

# sélection des individus par leur qualité de représentation
# plot(res.pca,habillage="trsf_dev",select="cos2 0.6")
# plot(res.pca,habillage="trsf_dev",select="cos2 0.6",new.plot=F)


# sélection des variables par leur qualité de représentation
#plot(res.pca,invisible="ind",selectMod="cos2 0.6",new.plot=F)


#####
# sélection automatique des variables
dimdesc(res.pca)

## $Dim.1
## $Dim.1$quanti
##           correlation      p.value
## AGE_CLI    0.83643593 0.000000e+00
## NB_ENF     0.67884852 0.000000e+00
## PUIS_FISC  0.16235647 1.804922e-38
## ANC_VEH   -0.08504776 1.366936e-11
## C_AN_TTC  -0.66531375 0.000000e+00
## COEF      -0.86639298 0.000000e+00
##
## $Dim.1$quali
##                     R2    p.value
## trsf_dev 0.0006537395 0.04242241
##
## $Dim.1$category
##      Estimate    p.value
## N  0.04449237 0.04242241
## O -0.04449237 0.04242241
##
##
## $Dim.2
## $Dim.2$quanti
##           correlation       p.value
```

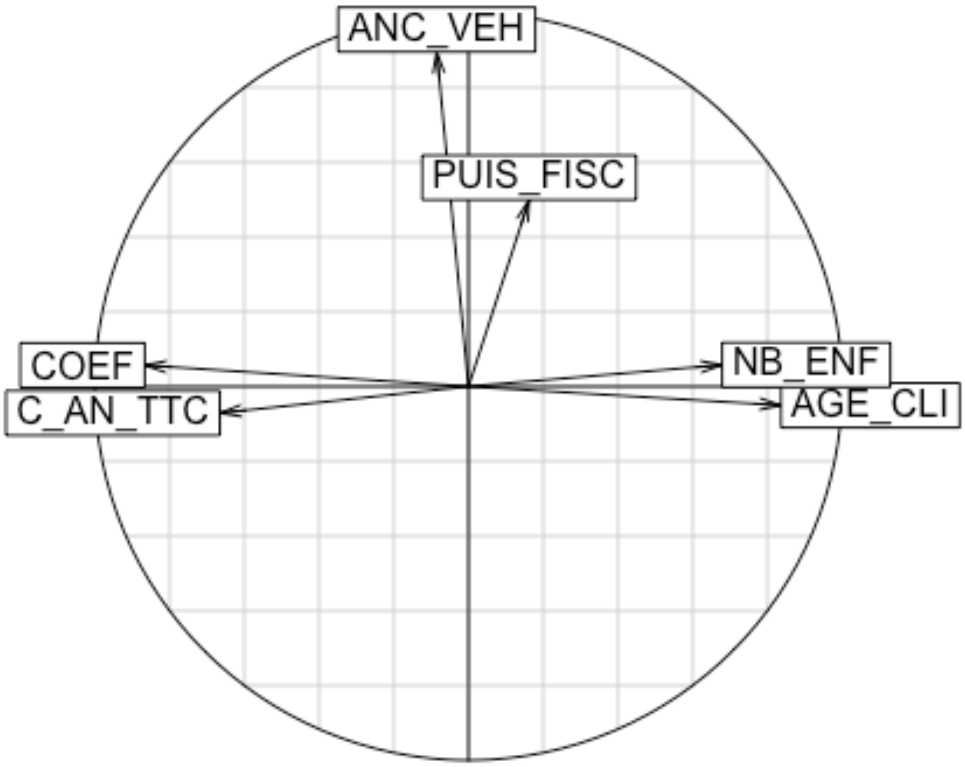
```
## PUIS_FISC   0.7793385  0.000000e+00
## C_AN_TTC    0.6384423  0.000000e+00
## NB_ENF      0.2018753  6.218450e-59
## AGE_CLI     0.1388932  1.643389e-28
## ANC_VEH    -0.3881874 1.105099e-225
##
## $Dim.2$quali
##                  R2      p.value
## trsf_dev 0.01554645 2.978712e-23
##
## $Dim.2$category
##     Estimate      p.value
## N  0.1554785 2.978712e-23
## O -0.1554785 2.978712e-23
##
##
## $Dim.3
## $Dim.3$quanti
##           correlation      p.value
## ANC_VEH    0.89531739 0.000000e+00
## PUIS_FISC  0.49929649 0.000000e+00
## NB_ENF     0.05772248 4.550888e-06
## COEF       0.05679729 6.455551e-06
## AGE_CLI   -0.05048883 6.092776e-05
## C_AN_TTC  -0.07114741 1.574928e-08

# plot
#plot(res.pca, choix="ind",habillage=7,axes=1:2,cex=0.7)
#plot(res.pca, choix="ind",habillage=7,axes=3:4,cex=0.7)
#plot(res.pca, choix="ind",habillage=7,axes=1:2,cex=0.7,new.plot=F)
#plot(res.pca, choix="ind",habillage=7,axes=3:4,cex=0.7,new.plot=F)


#plot(res.pca, choix="var",habillage=7,axes=1:2,new.plot=TRUE)
#plot(res.pca, choix="var",habillage=7,axes=3:4,new.plot=TRUE)
#plot(res.pca, choix="var",habillage=7,axes=1:2,new.plot=F)
#plot(res.pca, choix="var",habillage=7,axes=3:4,new.plot=F)


# graphe(s) des individus amélioré(s)
#plot(res.pca, autoLab="auto")

# sélection des individus par leur qualité de représentation
#plot(res.pca,habillage="trsf_dev",select="cos2 0.6")
#plot(res.pca,habillage="trsf_dev",select="cos2 0.6",new.plot=F)


# sélection des variables par leur qualité de représentation
#plot(res.pca,invisible="ind",selectMod="cos2 0.6",new.plot=F)





### ACP AVEC LE PACKAGE ADE4
acpdq2norm_y=dudi.pca(dq2norm_y[,1:6],center=TRUE,scale=TRUE,scannf=F,nf=4)
acpdq2norm_y

## Duality diagramm
## class: pca dudi
## $call: dudi.pca(df = dq2norm_y[, 1:6], center = TRUE, scale = TRUE,
##     scannf = F, nf = 4)
##
## $nf: 4 axis-components saved
## $rank: 6
## eigen values: 2.387 1.226 1.065 0.6938 0.4059 ...
##   vector length mode    content
## 1 $cw    6      numeric column weights
## 2 $lw    6300   numeric row weights
## 3 $eig   6      numeric eigen values
##
##   data.frame nrow ncol content
```

```
## 1 $tab       6300 6    modified array
## 2 $li        6300 4    row coordinates
## 3 $l1        6300 4    row normed scores
## 4 $co        6    4    column coordinates
## 5 $c1        6    4    column normed scores
## other elements: cent norm

# pourcentage d'inertie
iner=inertia.dudi(acpdq2norm_y,col.inertia=T,row.inertia=T)
#iner

# eboulis des VP
scatterutil.eigen(acpdq2norm_y$eig,box=T,sub="eboulis des VP")
```

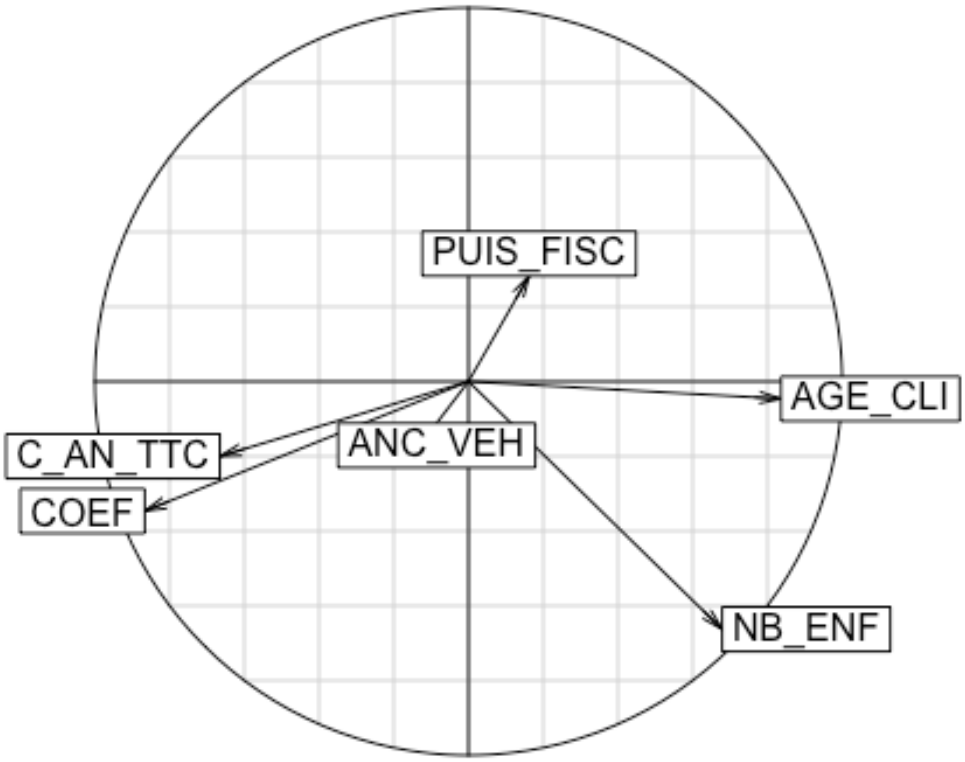
```
# cercle des correlations des CP12
s.corcircle(acpdq2norm_y$co,xax=1,yax=2)
```



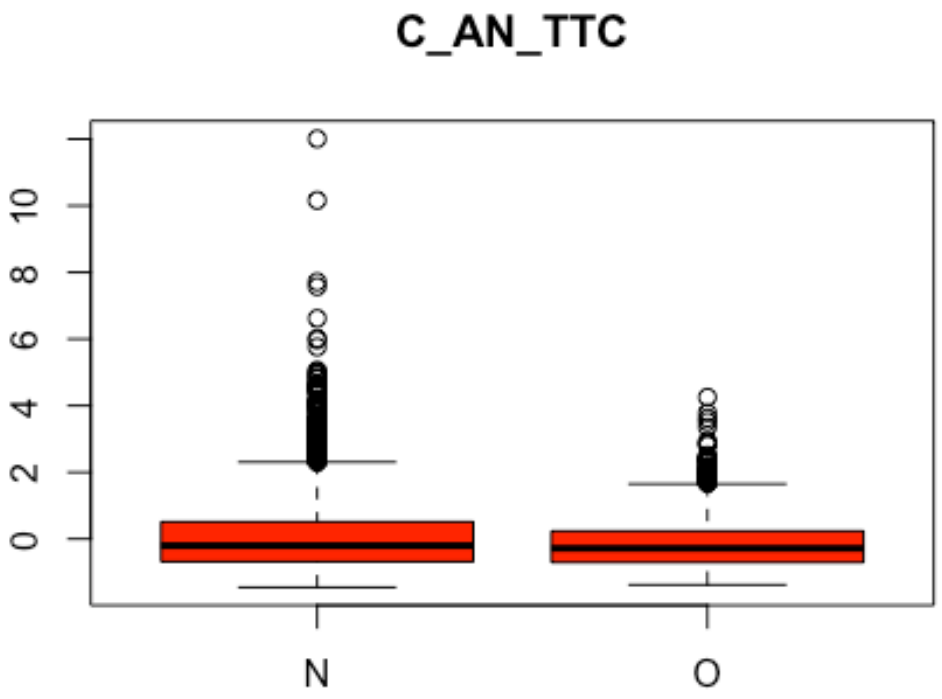

```
# cercle des correlations des CP13
s.corcircle(acpdq2norm_y$co,xax=1,yax=3)
```

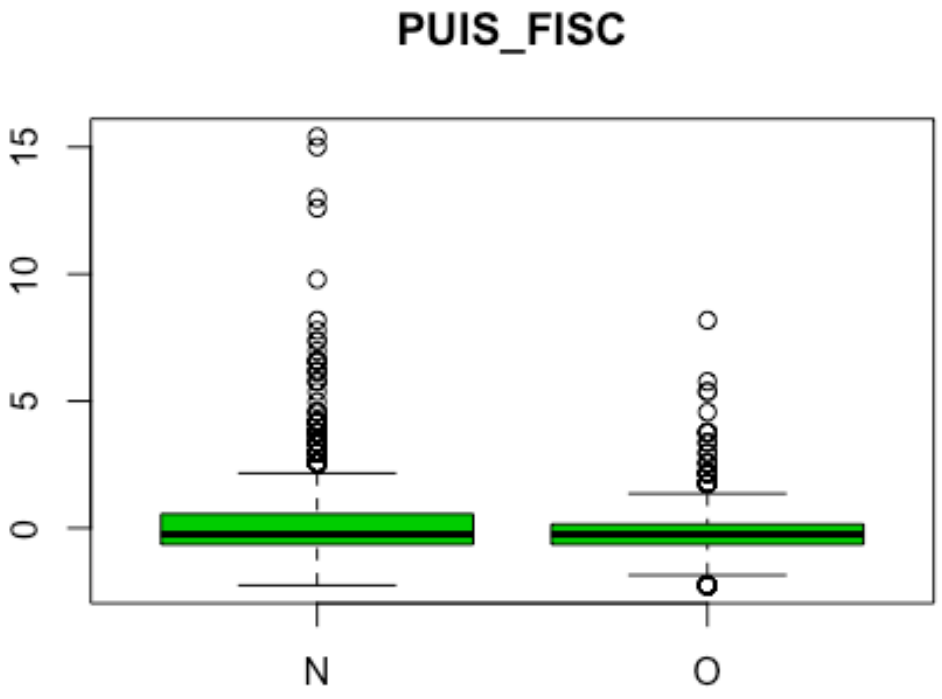


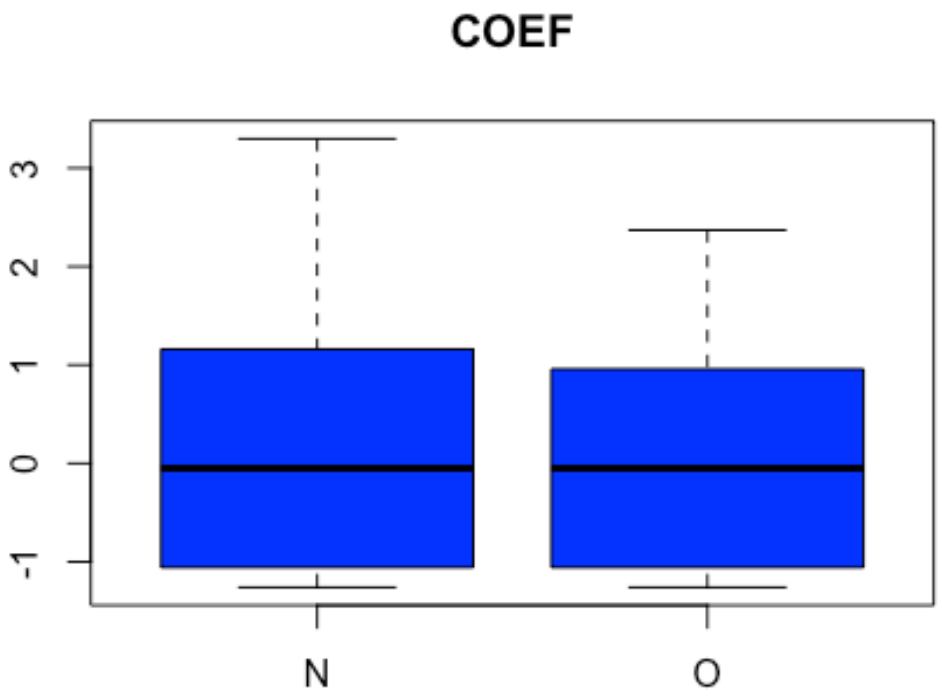

```
# cercle des correlations des CP14
s.corcircle(acpdq2norm_y$co,xax=1,yax=4)
```

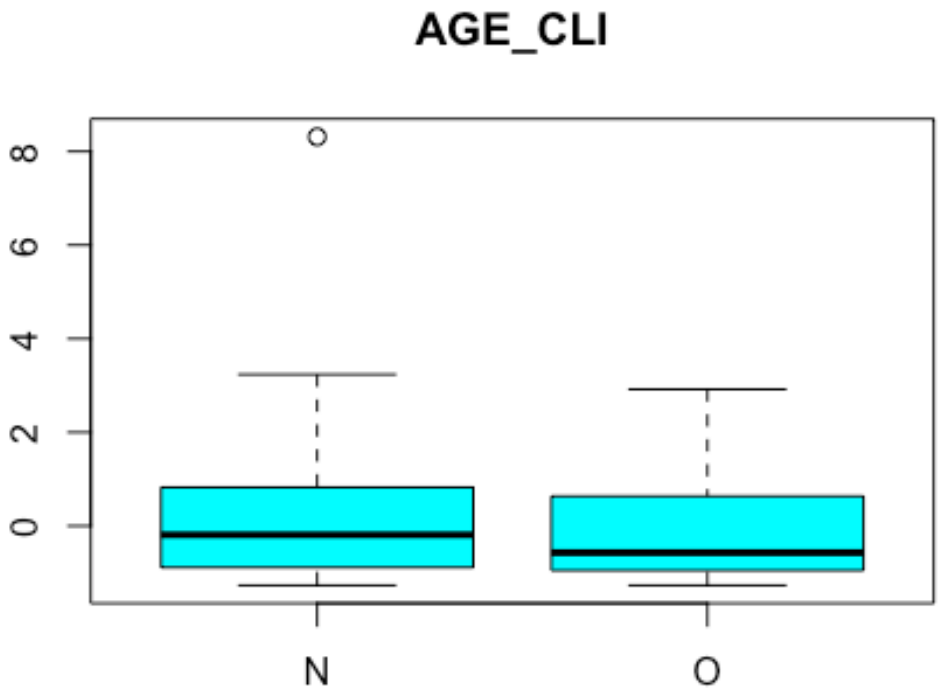


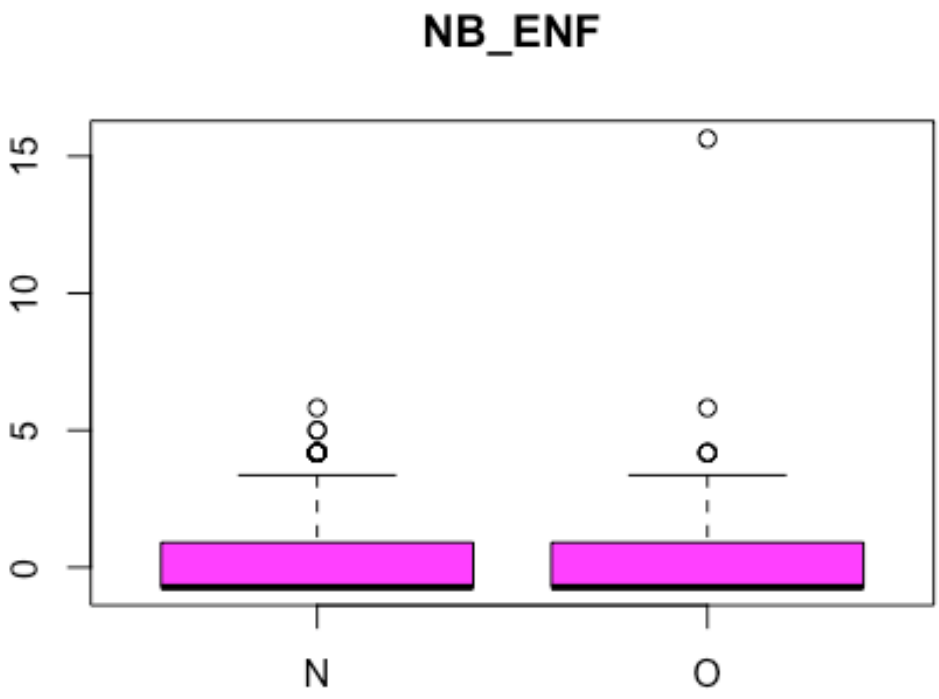

```
################################
# PARTIE 3.1.2 : ANALYSE FACTORIELLE DISCRIMINANTE (LDA)
################################

# mise en place d'une analyse factorielle discriminante
# packages (ade4 et Mass)

library(ade4)
library(MASS)
tab=dq2norm_y[,1:6]
#cla=dq2norm_y[,7]
cla=dq2norm_y$trsf_dev

par(mfrow=c(1,1))
for (k in 1:6){
  boxplot(tab[,k]~cla, main=names(tab)[k], col=k+1)
}
```

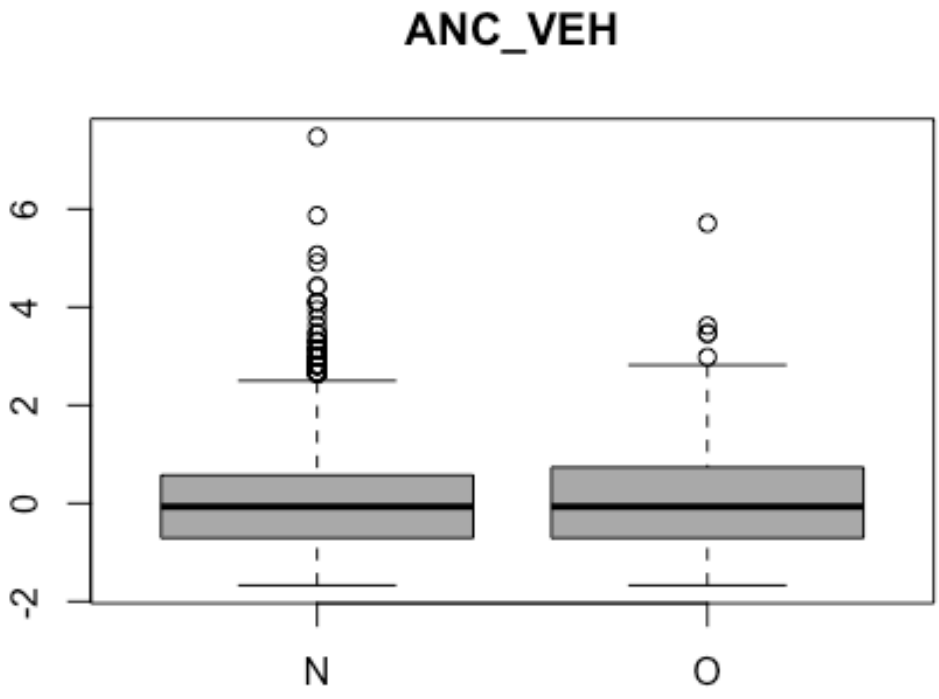

C_AN_TTC
0
2
4
6
8
10
N
O


PUIS_FISC
0
5
10
15
N
O


COEF
3
2
1
0
-1
N
O


AGE_CLI
8
6
4
2
0
N
O






```
anova(lm(tab[,1]~cla))

## Analysis of Variance Table
##
## Response: tab[, 1]
##             Df Sum Sq Mean Sq F value    Pr(>F)
## cla          1   48.5  48.524  48.893 2.982e-12 ***
```

```
## Residuals 6298 6250.5   0.992
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
anova(lm(tab[,1]~cla))$Pr[1]
```

```
## [1] 2.98209e-12
```

```r
anova(lm(tab[,2]~cla))
```

```
## Analysis of Variance Table
##
## Response: tab[, 2]
##             Df Sum Sq Mean Sq F value    Pr(>F)
## cla          1   42.8  42.805  43.091 5.642e-11 ***
## Residuals 6298 6256.2   0.993
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
anova(lm(tab[,2]~cla))$Pr[2]
```

```
## [1] NA
```

```r
anova(lm(tab[,3]~cla))
```

```
## Analysis of Variance Table
##
## Response: tab[, 3]
##             Df Sum Sq Mean Sq F value Pr(>F)
## cla          1    1.7 1.73305  1.7332  0.188
## Residuals 6298 6297.3 0.99988
```

```r
anova(lm(tab[,3]~cla))$Pr[3]
```

```
## [1] NA
```

```r
anova(lm(tab[,4]~cla))
```

```
## Analysis of Variance Table
##
## Response: tab[, 4]
##             Df Sum Sq Mean Sq F value    Pr(>F)
## cla          1   58.4  58.371  58.907 1.904e-14 ***
## Residuals 6298 6240.6   0.991
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
anova(lm(tab[,4]~cla))$Pr[4]
```

```
## [1] NA
```

```r
anova(lm(tab[,5]~cla))
```

```
## Analysis of Variance Table
##
## Response: tab[, 5]
##             Df Sum Sq Mean Sq F value    Pr(>F)
## cla          1   19.9  19.873  19.933 8.161e-06 ***
## Residuals 6298 6279.1   0.997
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

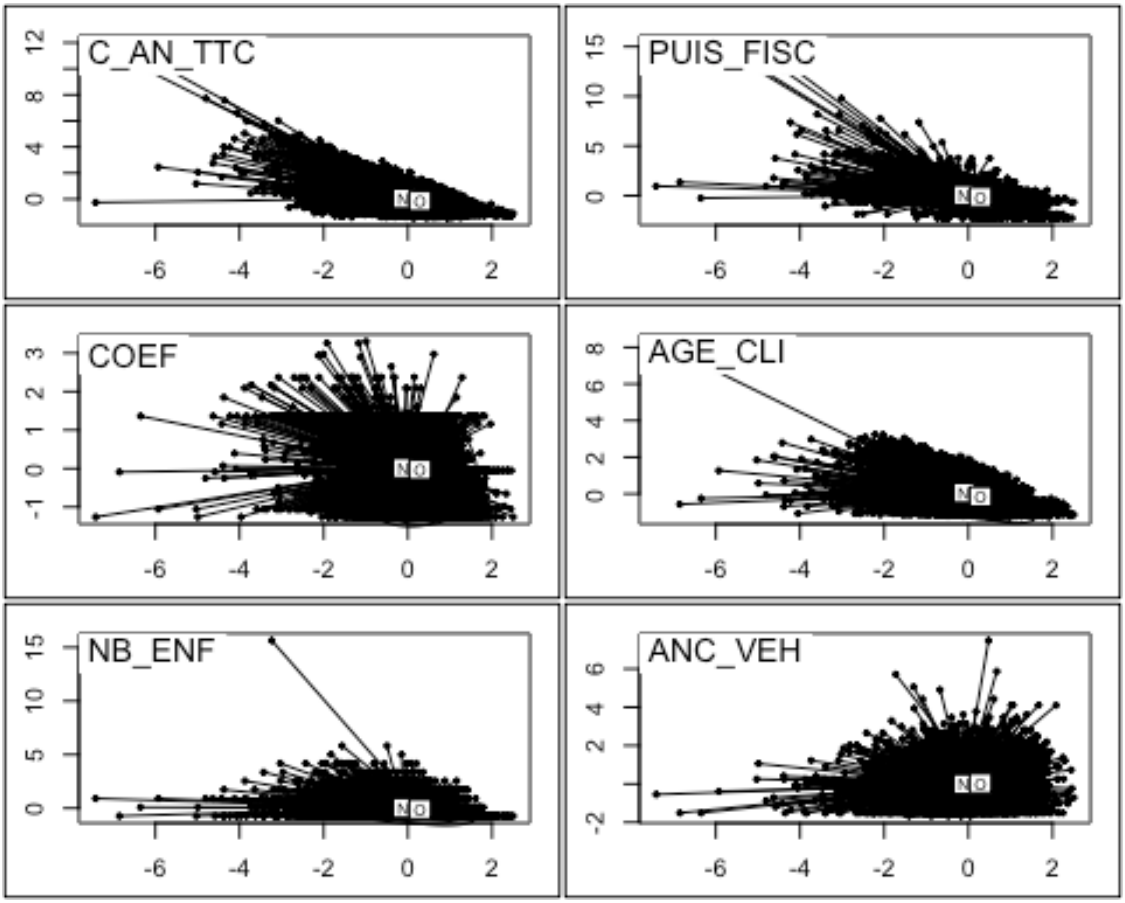
```r
anova(lm(tab[,5]~cla))$Pr[5]
```

```
## [1] NA
```

```r
anova(lm(tab[,6]~cla))
```

```
## Analysis of Variance Table
##
## Response: tab[, 6]
##             Df Sum Sq Mean Sq F value  Pr(>F)
## cla          1    2.7 2.73723   2.738 0.09804 .
```

```
## Residuals 6298 6296.3 0.99972
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

anova(lm(tab[,6]~cla))$Pr[6]

## [1] NA

#afd avec lda
library(MASS)
afd1=lda(cla~.,tab)

# coordonnées des var ini sur axe discri
round(afd1$scaling,2)

##              LD1
## C_AN_TTC  -0.61
## PUIS_FISC -0.24
## COEF      -0.42
## AGE_CLI   -0.95
## NB_ENF    -0.13
## ANC_VEH   -0.03

# analyse discrimante avec discrimin
afd2=discrimin(dudi.pca(tab,scan=FALSE),cla,scan=FALSE,nf=2)
afd2

## Discriminant analysis
## call: discrimin(dudi = dudi.pca(tab, scan = FALSE), fac = cla, scannf = FALSE,
##     nf = 2)
## class: discrimin
##
## $nf (axis saved) : 1
##
## eigen values: 0.03086
##
##   data.frame nrow ncol content
## 1 $fa        6    1    loadings / canonical weights
## 2 $li        6300 1    canonical scores
## 3 $va        6    1    cos(variables, canonical scores)
## 4 $cp        6    1    cos(components, canonical scores)
## 5 $gc        2    1    class scores

plot(afd2)
```


C_AN_TTC
PUIS_FISC
COEF
AGE_CLI
NB_ENF
ANC_VEH
No

CODE R	RESULTATS
Modèle Random forest « forêts aléatoires »	
Modèle XGboost	

(i) Modèle Random forest « forêts aléatoires »

RandomForest.R

lepicaut

Mon Aug 28 18:58:49 2017

```
############
# specification du modele : randomForest
############

############
# compile rapport : RandomForest.R - en date du 28 aout17
############

# 0. nettoyer historique
rm(list = ls(all = TRUE)) # make sure previous work is clear
ls()

## character(0)

tic <- Sys.time()

# 1. importing and exploring data

path<-"/Users/lepicaut/Desktop/sos/MATMUT/0_0_matmut/fevrier_2017/00000_code"
#path <- "/Users/lepicaut/Desktop/00000_code"

# 2. set working directory
setwd(path)
dv<- read.csv("devis_trsf_2015.csv", header=TRUE, sep=";")


# chargement des librairies R
library(plyr)
library(MASS)
library(corrplot)
library(lattice)
library(caret)

## Loading required package: ggplot2

library(ROCR)

## Loading required package: gplots

##
## Attaching package: 'gplots'

## The following object is masked from 'package:stats':
##
##     lowess

library(ggplot2)
library(FactoMineR)
library(ade4)
```

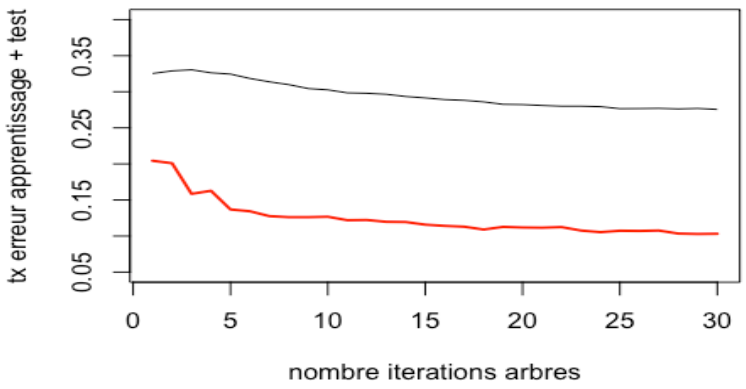
```
## Warning: package 'ade4' was built under R version 3.4.1

##
## Attaching package: 'ade4'

## The following object is masked from 'package:FactoMineR':
##
##     reconst

library(colorspace)
library(e1071)
library(tidyr)

## Warning: package 'tidyr' was built under R version 3.4.1

library(rmarkdown)
library(randomForest)

## randomForest 4.6-12

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:ggplot2':
##
##     margin

library(MLmetrics)

##
## Attaching package: 'MLmetrics'

## The following object is masked from 'package:caret':
##
##     RMSE

## The following object is masked from 'package:base':
##
##     Recall

library(ranger)

##
## Attaching package: 'ranger'

## The following object is masked from 'package:randomForest':
##
##     importance

################################
# PARTIE 1. PREPROCESSING DATA
################################
#str(dv)
#names(dv)

dv1<-dv[,-3:-5]

dv2<-dv1[,-2]

dv3<-dv2[,-3:-5]

dv4<-dv3[,-4:-5]

dv5<-dv4[,-4]
```

```r
dv6<-dv5[,-5:-8]

dv7<-dv6[,-7:-14]

dv8<-dv7[,-7:-19]

dv9<-dv8[,-7:-9]

dv10<-dv9[,-8:-9]

dv11<-dv10[,-9:-11]

dv12<-dv11[,-14:-15]

#names(dv12)

which(names(dv12) %in% c('NB_DEVI_ARCH_MRSQ','DAT_ADH'))

## [1] 15 47

dv13<-dv12[,-15:-47]

names(dv13)

##  [1] "ID_CLIENT"  "NUM_DEV"    "FLE_MRSQ"   "C_AN_TTC"   "COEFF"
##  [6] "COM_MRSQ"   "ZON_MRSQ"   "BL_ORI_NET" "MARQ_VEH"   "MODL_VEH"
## [11] "PUIS_FISC"  "NRJ"        "GAR_GN"     "COEF"       "SEXE"
## [16] "IDENT"      "AGE_CLI"    "CSP"        "PROF"       "CD_FONC"
## [21] "CD_ADMIN"   "SIT_FAMD"   "NB_ENF"     "AGE_E1"     "AGE_E2"
## [26] "AGE_E3"     "AGE_E4"     "AGE_E5"     "LIB_MOTIV"  "ACT_REAL"
## [31] "SIT_REAL1"  "TYP_REAL"   "trsf_dev"   "ANC_VEH"

which(names(dv13) %in% c('AGE_E1','TYP_REAL'))

## [1] 24 32

dv14<-dv13[,-24:-32]

which(names(dv14) %in% c('CD_ADMIN'))

## [1] 21

dv15<-dv14[,-21]

which(names(dv15) %in% c('MARQ_VEH'))

## [1] 9

dv16<-dv15[,-9]

#summary(dv16)

#dim(dv16)

attach(dv16)

dv17=na.omit(dv16)

#summary(dv17)

(table(dv17$trsf_dev)/nrow(dv17)*100)

##
##        N        O
## 73.31072 26.68928
```

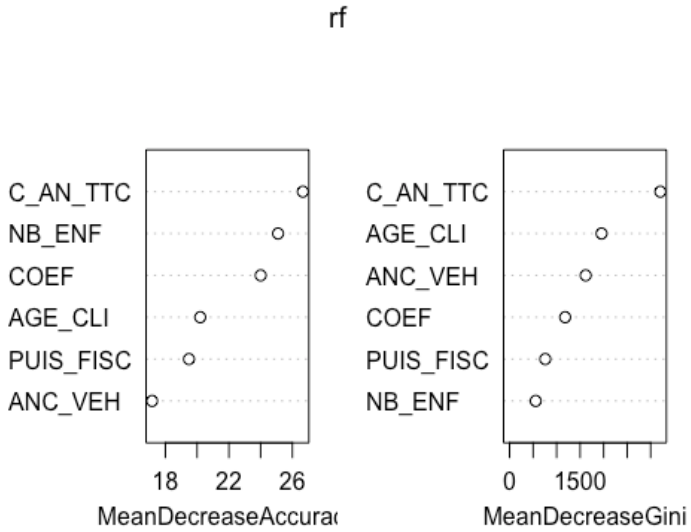
```
#dim(dv17)

#sum(is.na(dv17))

#str(dv17)

# passage de > 600 000 lignes à 1000 lignes
# 5% effectif tot
dv18=dv17[sample(1:nrow(dv17),30702),]

# passage de > 614048 lignes à 6140 lignes
# cad poids échantillon = 0.99% x échantillon total observé
#dv18=dv17[sample(1:nrow(dv17),6140),]

#dim(dv18)
(table(dv18$trsf_dev)/nrow(dv18)*100)

##
##        N        O
## 73.37307 26.62693

#summary(dv18)
#str(dv18)

# transfrome C_AN_TTC en var numerique
dv18$C_AN_TTC = gsub(",",".",dv18$C_AN_TTC)
dv18$C_AN_TTC = gsub(" ","",dv18$C_AN_TTC)
dv18$C_AN_TTC = as.double(dv18$C_AN_TTC)

#str(dv18)
#dv18$C_AN_TTC[1:100]

# IDENTIFIER LES QUANTI
library(plyr)
type_var_num = colwise(is.numeric)
var_num=names(dv18)[which(type_var_num(dv18)==TRUE)]

dv18Quanti = dv18[, c(var_num)]
#str(dv18Quanti)

etiq<-dv18[,-2:-21]
etiq<-etiq[,-3]
#str(etiq)

dq=merge(dv18Quanti,etiq,by="ID_CLIENT")
#str(dq)

#summary(dq)

dq1=dq[,-4]
#str(dq1)

dq2=dq1[,-1]
#str(dq2)

# boxplot sur données non normalisées
#boxplot(dq2)

################################
# PARTIE 2. ANALYSE DESCRIPTIVE
################################

# lecture du graphe : sur ces données non normalisées, on ne peut rien dire.
```

```r
dq2norm=as.data.frame(scale(dq2[,2:7]))
sum(is.na(dq2$trsf_dev))

## [1] 0

#str(dq2norm)
#summary(dq2norm)
class(dq2norm)

## [1] "data.frame"

#boxplot(dq2norm, col="lightblue",xlab="variables",cex.lab=0.8,cex.main=0.8)

# mise en place d'un modèle d'analyse discriminante linéaire avec le package MASS
dq2norm_y = dq2norm
dq2norm_y$trsf_dev = dq2$trsf_dev
#str(dq2norm_y)
#library(MASS)
#lda(dq2norm_y$trsf_dev ~.,data=dq2norm_y)

getwd()

## [1] "/Users/lepicaut/Desktop/sos/MATMUT/0_0_matmut/fevrier_2017/00000_code"

#write.csv2(dq2norm_y,file='dq2norm_y$trsf_dev.csv')
#head(dq2norm_y)
#head(dq2norm_y$trsf_dev)

# corrélation linéaire entre les variables explicatives
#library(corrplot)
#corrplot(cor(dq2norm),type="upper", order="hclust",tl.col="black", tl.srt=45)

#hist(dq2$C_AN_TTC)
#hist(dq2$PUIS_FISC)
#hist(dq2$COEF)
#hist(dq2$AGE_CLI)
#hist(dq2$NB_ENF)

################################
# PARTIE 3. MODELISATION DES CLASSIFIEURS
################################

################################
# PARTIE 3.1. : SELECTION DU MODELE (Feature Ingineering)
################################

################################
# PARTIE 3.1.1 : ANALYSE randomFOREST
################################

# mise en place d'un modèle d'analyse linéaire discriminante avec le package MASS
# modele complet : 6 var quanti + var etiquette
dq2norm_y = dq2norm
dq2norm_y$trsf_dev = dq2$trsf_dev
#str(dq2norm_y)

# procedure RF
set.seed(235)

# apprentissage-validation
e=sort(sample(nrow(dq2norm_y),nrow(dq2norm_y)*0.9))

# test
t=sort(sample(nrow(dq2norm_y),nrow(dq2norm_y)*0.1))

# ALGO randomForest
```

```
rf<-randomForest(dq2norm_y$trsf_dev[e]~.,data=dq2norm_y[e,1:6],importance=TRUE,ntree=30,mtry=2,
replace=TRUE,keep.forest=TRUE,
                 nodesize=1,ytest=dq2norm_y[t,7],xtest=dq2norm_y[t,1:6])
rf

##
## Call:
##  randomForest(formula = dq2norm_y$trsf_dev[e] ~ ., data = dq2norm_y[e,      1:6], importance
= TRUE, ntree = 30, mtry = 2, replace = TRUE,      keep.forest = TRUE, nodesize = 1, ytest = dq
2norm_y[t, 7],      xtest = dq2norm_y[t, 1:6])
##                Type of random forest: classification
##                      Number of trees: 30
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 27.55%
## Confusion matrix:
##       N    O class.error
## N 21108 2301  0.09829553
## O  6522 2091  0.75722745
##                 Test set error rate: 10.31%
## Confusion matrix:
##      N   O class.error
## N 2545  49  0.01888975
## O  318 646  0.32987552

# taux d'erreur sur l'echantillon apprentissage
tail(rf$err.rate)

##             OOB          N         O
## [25,] 0.2765685 0.10176008 0.7516545
## [26,] 0.2767160 0.10115767 0.7538604
## [27,] 0.2769971 0.10154214 0.7538604
## [28,] 0.2762476 0.10021786 0.7546732
## [29,] 0.2768722 0.10013243 0.7572274
## [30,] 0.2755293 0.09829553 0.7572274

# taux d'erreur sur l'echantillon test
tail(rf$test$err.rate)

##            Test          N         O
## [25,] 0.1073637 0.01966076 0.3433610
## [26,] 0.1070826 0.01927525 0.3433610
## [27,] 0.1076447 0.02120278 0.3402490
## [28,] 0.1034289 0.01966076 0.3288382
## [29,] 0.1028668 0.01888975 0.3288382
## [30,] 0.1031478 0.01888975 0.3298755

# le taux d'erreur en fonction iteration algo RF
plot(rf$err.rate[,1],type="l",ylim=c(.05,.4), xlab="nombre iterations arbres",ylab="tx erreur a
pprentissage + test")
lines(rf$test$err.rate[1:rf$ntree,1],col='red',lwd=2)
```




```
# importance des variables
varImpPlot(rf)
```



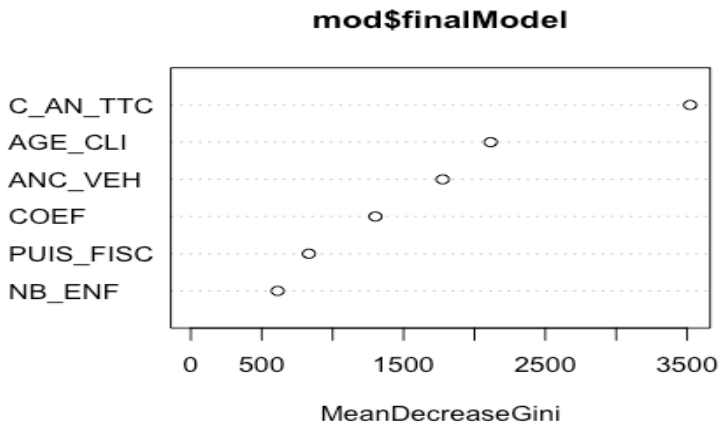

```
#  graphe des proximités
#set.seed(235)
#dq2norm_y.rf <- randomForest(trsf_dev ~ ., dq2norm_y, proximity=TRUE,
#                        keep.forest=FALSE)
#MDSplot(dq2norm_y.rf, dq2norm_y$trsf_dev)

# matrice de confusion RF
matriceconfusion<-rf$confusion
matriceconfusion

##       N    O class.error
## N 21108 2301  0.09829553
## O  6522 2091  0.75722745

# MLmetrics
# air sous courbe de roc
#
# PRAUC(y_pred = rf$fitted.values, y_true = dq2norm_y$trsf_dev)

# caret - version simplifie
#
#set.seed(235)
#library(caret)
mod <- train(trsf_dev ~ ., data = dq2norm_y, method = "rf")
print(mod)
```

```
## Random Forest
##
## 35580 samples
##     6 predictor
##     2 classes: 'N', 'O'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
##
## Summary of sample sizes: 35580, 35580, 35580, 35580, 35580, 35580, ...
##
## Resampling results across tuning parameters:
##
##   mtry  Accuracy  Kappa  Accuracy SD  Kappa SD
##   2     0.735     0.175  0.00285      0.00604
##   4     0.720     0.187  0.00288      0.00528
##   6     0.717     0.185  0.00297      0.00608
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 2.
```

```
print(mod$finalModel)
```

```
##
## Call:
##  randomForest(x = x, y = y, mtry = param$mtry)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 26.31%
## Confusion matrix:
##       N    O class.error
## N 24272 1740   0.0668922
## O  7622 1946   0.7966137
```

```
varImpPlot(mod$finalModel)
```

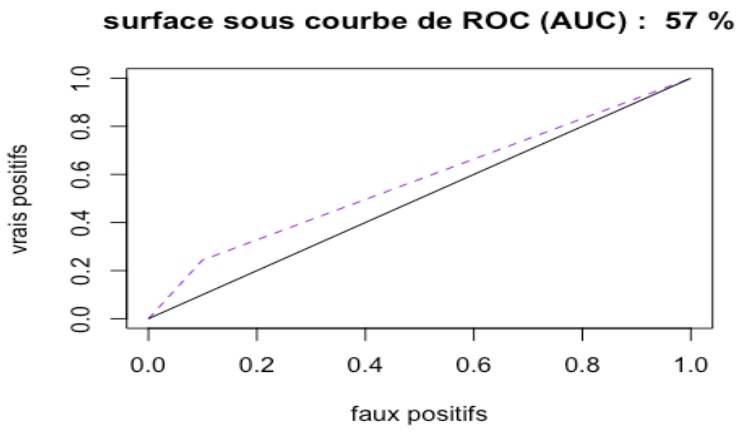



```
varImp(mod)
```

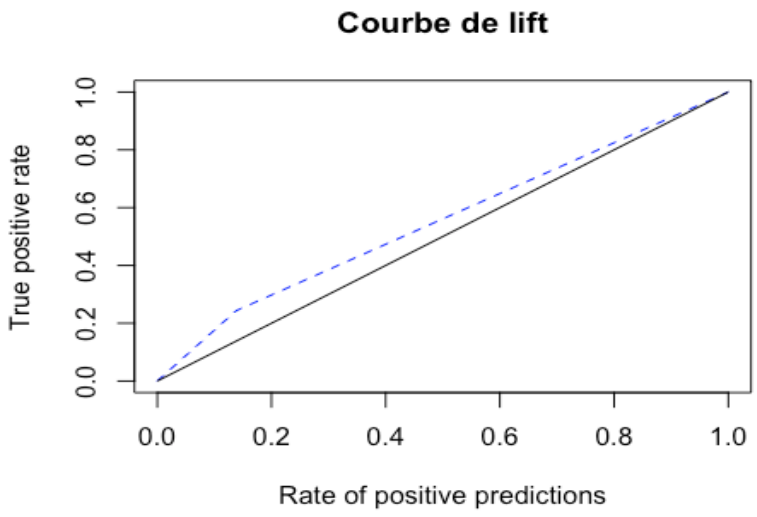
```
## rf variable importance
##
##           Overall
## C_AN_TTC  100.000
## AGE_CLI    51.627
## ANC_VEH    40.003
## COEF       23.697
## PUIS_FISC   7.578
## NB_ENF      0.000
```

```
# estimer le taux de mauvais classement par validation croisée
# prev<-lda(dq2norm_y$trsf_dev ~ C_AN_TTC + AGE_CLI, data = dq2norm_y,CV=TRUE)$class

prev<-randomForest(dq2norm_y$trsf_dev[e]~.,data=dq2norm_y[e,1:6],importance=TRUE,ntree=30,mtry=
2,replace=TRUE,keep.forest=TRUE,
                                          nodesize=1,ytest=dq2norm_y[t,7],xtest=dq2norm_y[t,1:6]
)

predt<-prev$predicted
length(predt)

## [1] 32022

t=sort(sample(nrow(dq2norm_y),nrow(dq2norm_y)*0.9))
length(dq2norm_y[t,7])

## [1] 32022

table(predt,dq2norm_y[t,7])

##
## predt     N     O
##     N 20397  7199
##     O  3007  1419

# matrice de confusion - indicateur de performance du classifieur - RF avec 6 variables explica
tives
#
library(caret)
confusionMatrix(data=predt, reference=dq2norm_y[t,7])

## Confusion Matrix and Statistics
##
##           Reference
## Prediction     N     O
##          N 20397  7199
##          O  3007  1419
##
##                Accuracy : 0.6813
##                  95% CI : (0.6761, 0.6864)
##     No Information Rate : 0.7309
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.0427
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.8715
##             Specificity : 0.1647
##          Pos Pred Value : 0.7391
##          Neg Pred Value : 0.3206
##              Prevalence : 0.7309
##          Detection Rate : 0.6370
##    Detection Prevalence : 0.8618
##       Balanced Accuracy : 0.5181
##
##        'Positive' Class : N
##

#confusionMatrix(data=predt, reference=dq2norm_y[e,7],mode="prec_recall")

library(ROCR)
fr <- data.frame(score = c(predt),label =c(dq2norm_y[e,7]))
pred <- prediction(fr$score, fr$label)
perf <- performance(pred,"tpr", "fpr")

y.auc.perf= performance(pred, measure="auc",x.measure="cutoff")
```

```r
auc=round(unlist(y.auc.perf@y.values),2)*100
print(auc)

## [1] 57

plot(perf, col="purple", lty=2, pch=32,
     main = paste("surface sous courbe de ROC (AUC) : ",auc, "%"),
     xlab = "faux positifs",
     ylab = "vrais positifs")

segments(0,0,1,1,lty=1)
```




```r
#graphics.off()

# la courbe ROC du modèle RF, est alignée sur la courbe de prédiction basée sur le hasard.
# la prédiction de mon modèle RF ne fait pas mieux que "le modèle du hasard".

# courbe de lift
lift<- performance(pred,"tpr","rpp")
plot(lift, main='Courbe de lift',col="blue", lty=2, pch=32)
segments(0,0,1,1,lty=1)
```




```r
# comparer la performance 2 classifieurs
# TEST MCNEMAR
# matrice confusion RF DLP
mat<-matrix(c(1457246,181394,315166,315128),2)
mcnemar.test(mat)

##
##  McNemar's Chi-squared test with continuity correction
```

```
##
## data:  mat
## McNemar's chi-squared = 36037, df = 1, p-value < 2.2e-16

# matrice confusion RF NAIMA
mat<-matrix(c(108188,8257,19542,12123),2)
mcnemar.test(mat)

##
##  McNemar's Chi-squared test with continuity correction
##
## data:  mat
## McNemar's chi-squared = 4580.3, df = 1, p-value < 2.2e-16

# matrice de confusion RF
mat<-matrix(c(1565434,94825,167354,327251),2)
mcnemar.test(mat)

##
##  McNemar's Chi-squared test with continuity correction
##
## data:  mat
## McNemar's chi-squared = 20064, df = 1, p-value < 2.2e-16

toc <-  Sys.time() -tic
print(toc)

## Time difference of 44.76621 min
```

(ii) Modèle XGboost

xgboost.R

lepicaut

Mon Aug 28 22:04:53 2017

```
############
# specification du modele : XGboost - 28 aout 17
############

# 0. nettoyer historique
rm(list = ls(all = TRUE)) # make sure previous work is clear
ls()

## character(0)

tic <- Sys.time()

# 1. importing and exploring data
path<-"/Users/lepicaut/Desktop/sos/MATMUT/0_0_matmut/fevrier_2017/00000_code"

# 2. set working directory
setwd(path)
dv<- read.csv("devis_trsf_2015.csv", header=TRUE, sep=";")

# chargement des librairies R
library(plyr)
library(MASS)
library(corrplot)
library(lattice)
library(caret)

## Loading required package: ggplot2

library(ROCR)

## Loading required package: gplots

##
## Attaching package: 'gplots'

## The following object is masked from 'package:stats':
##
##     lowess

library(ggplot2)
library(FactoMineR)
library(ade4)

## Warning: package 'ade4' was built under R version 3.4.1

##
## Attaching package: 'ade4'

## The following object is masked from 'package:FactoMineR':
##
##     reconst

library(colorspace)
library(e1071)
library(tidyr)
```

```
## Warning: package 'tidyr' was built under R version 3.4.1

library(rmarkdown)
library(randomForest)

## randomForest 4.6-12

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:ggplot2':
##
##     margin

library(MLmetrics)

##
## Attaching package: 'MLmetrics'

## The following object is masked from 'package:caret':
##
##     RMSE

## The following object is masked from 'package:base':
##
##     Recall

library(xgboost)
library(Ckmeans.1d.dp)
library(gbm)

## Loading required package: survival

##
## Attaching package: 'survival'

## The following object is masked from 'package:caret':
##
##     cluster

## Loading required package: splines

## Loading required package: parallel

## Loaded gbm 2.1.3

library(survival)

################################
# PARTIE 1. PREPROCESSING DATA
################################

#str(dv)

#names(dv)

dv1<-dv[,-3:-5]

dv2<-dv1[,-2]

dv3<-dv2[,-3:-5]

dv4<-dv3[,-4:-5]

dv5<-dv4[,-4]
```

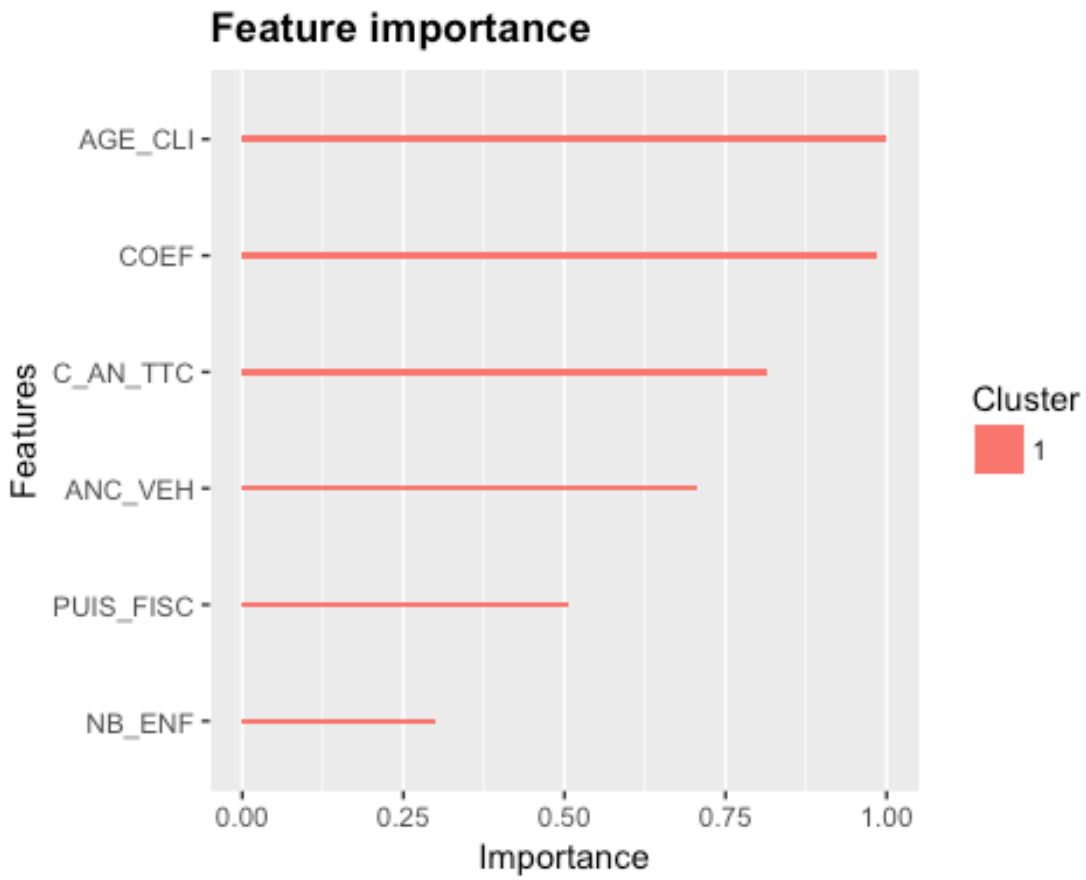
```
dv6<-dv5[,-5:-8]

dv7<-dv6[,-7:-14]

dv8<-dv7[,-7:-19]

dv9<-dv8[,-7:-9]

dv10<-dv9[,-8:-9]

dv11<-dv10[,-9:-11]

dv12<-dv11[,-14:-15]

#names(dv12)

which(names(dv12) %in% c('NB_DEVI_ARCH_MRSQ','DAT_ADH'))

## [1] 15 47

dv13<-dv12[,-15:-47]

names(dv13)

##  [1] "ID_CLIENT"  "NUM_DEV"    "FLE_MRSQ"   "C_AN_TTC"   "COEFF"
##  [6] "COM_MRSQ"   "ZON_MRSQ"   "BL_ORI_NET" "MARQ_VEH"   "MODL_VEH"
## [11] "PUIS_FISC"  "NRJ"        "GAR_GN"     "COEF"       "SEXE"
## [16] "IDENT"      "AGE_CLI"    "CSP"        "PROF"       "CD_FONC"
## [21] "CD_ADMIN"   "SIT_FAMD"   "NB_ENF"     "AGE_E1"     "AGE_E2"
## [26] "AGE_E3"     "AGE_E4"     "AGE_E5"     "LIB_MOTIV"  "ACT_REAL"
## [31] "SIT_REAL1"  "TYP_REAL"   "trsf_dev"   "ANC_VEH"

which(names(dv13) %in% c('AGE_E1','TYP_REAL'))

## [1] 24 32

dv14<-dv13[,-24:-32]

which(names(dv14) %in% c('CD_ADMIN'))

## [1] 21

dv15<-dv14[,-21]

which(names(dv15) %in% c('MARQ_VEH'))

## [1] 9

dv16<-dv15[,-9]

#summary(dv16)

#dim(dv16)

attach(dv16)

dv17=na.omit(dv16)

#summary(dv17)

(table(dv17$trsf_dev)/nrow(dv17)*100)

##
##        N        O
## 73.31072 26.68928
```

```
#dim(dv17)

sum(is.na(dv17))

## [1] 0

#str(dv17)

# passage de > 600 000 lignes à 1000 lignes
# 20% effectif tot
dv18=dv17[sample(1:nrow(dv17),614048),]

# passage de > 614 048 lignes à 6140 lignes
# cad poids échantillon = 0.99% x échantillon total observé
#dv18=dv17[sample(1:nrow(dv17),6140),]

#dim(dv18)
(table(dv18$trsf_dev)/nrow(dv18)*100)

##
##        N        O
## 73.31072 26.68928

#summary(dv18)
#str(dv18)

# transfrome C_AN_TTC en var numerique
dv18$C_AN_TTC = gsub(",",".",dv18$C_AN_TTC)
dv18$C_AN_TTC = gsub(" ","",dv18$C_AN_TTC)
dv18$C_AN_TTC = as.double(dv18$C_AN_TTC)

#str(dv18)
#dv18$C_AN_TTC[1:100]

# IDENTIFIER LES QUANTI
library(plyr)
type_var_num = colwise(is.numeric)
var_num=names(dv18)[which(type_var_num(dv18)==TRUE)]

dv18Quanti = dv18[, c(var_num)]
#str(dv18Quanti)

etiq<-dv18[,-2:-21]
etiq<-etiq[,-3]
#str(etiq)

dq=merge(dv18Quanti,etiq,by="ID_CLIENT")
#str(dq)

#summary(dq)

dq1=dq[,-4]
#str(dq1)

dq2=dq1[,-1]
#str(dq2)

# boxplot sur données non normalisées
#boxplot(dq2)

################################
# PARTIE 2. ANALYSE DESCRIPTIVE
################################

# lecture du graphe : sur ces données non normalisées, on ne peut rien dire.
```

```r
dq2norm=as.data.frame(scale(dq2[,2:7]))
sum(is.na(dq2$trsf_dev))

## [1] 0

#str(dq2norm)
#summary(dq2norm)
class(dq2norm)

## [1] "data.frame"

#boxplot(dq2norm, col="lightblue",xlab="variables",cex.lab=0.8,cex.main=0.8)

# mise en place d'un modèle d'analyse discriminante linéaire avec le package MASS
dq2norm_y = dq2norm
dq2norm_y$trsf_dev = dq2$trsf_dev
#str(dq2norm_y)
#library(MASS)
#lda(dq2norm_y$trsf_dev ~.,data=dq2norm_y)

getwd()

## [1] "/Users/lepicaut/Desktop/sos/MATMUT/0_0_matmut/fevrier_2017/00000_code"

#write.csv2(dq2norm_y,file='dq2norm_y$trsf_dev.csv')
#head(dq2norm_y)
#head(dq2norm_y$trsf_dev)

# corrélation linéaire entre les variables explicatives
#library(corrplot)
#corrplot(cor(dq2norm),type="upper", order="hclust",tl.col="black", tl.srt=45)

#hist(dq2$C_AN_TTC)
#hist(dq2$PUIS_FISC)
#hist(dq2$COEF)
#hist(dq2$AGE_CLI)
#hist(dq2$NB_ENF)

################################
# PARTIE 3. MODELISATION DES CLASSIFIEURS
################################

################################
# PARTIE 3.1. : SELECTION DU MODELE (Feature Ingineering)
################################

################################
# PARTIE 3.1.1 : ANALYSE randomFOREST
################################

# mise en place d'un modèle d'analyse linéaire discriminante avec le package MASS
# modele complet : 6 var quanti + var etiquette
dq2norm_y = dq2norm
#dq2norm<-data.frame(lapply(dq2norm,as.numeric))
#dq2norm<-as.matrix(dq2norm, rownames.force = TRUE)

#e<-data.frame(lapply(e,as.numeric))
#e<-as.matrix(e, rownames.force = TRUE)

dq2norm_y$trsf_dev = dq2$trsf_dev
#str(dq2norm_y)

# procedure RF
set.seed(235)
```

```r
# apprentissage-validation
e=sort(sample(nrow(dq2norm_y),nrow(dq2norm_y)*0.70))
train=dq2norm_y[e,1:7]
train<-data.frame(lapply(train,as.numeric))
train$trsf_dev[train$trsf_dev==1]=0
train$trsf_dev[train$trsf_dev==2]=1
train<-as.matrix(train, rownames.force = TRUE)


# test
t=sort(sample(nrow(dq2norm_y),nrow(dq2norm_y)*0.30))
test=dq2norm_y[t,1:7]
test<-data.frame(lapply(test,as.numeric))
test$trsf_dev[test$trsf_dev==1]=0
test$trsf_dev[test$trsf_dev==2]=1
test<-as.matrix(test, rownames.force = TRUE)

# xgboost
require (xgboost)
dtrain <- xgb.DMatrix (data = train[,-7], label = train[,7])
dtest <- xgb.DMatrix (data = test[,-7], label = test[,7])
watchlist <- list (train = dtrain, test = dtest)

#foret <- xgb.train (data = dtrain, watchlist = watchlist, nrounds = 10,
#                     params = list (objective = "multi:softmax", num.class = 2))


foret <- xgb.train (data = dtrain, watchlist = watchlist, nrounds = 10,
                    params = list (objective = "multi:softmax", num.class = 2))

## [1]  train-merror:0.238422   test-merror:0.238670
## [2]  train-merror:0.238297   test-merror:0.238519
## [3]  train-merror:0.237121   test-merror:0.237263
## [4]  train-merror:0.236956   test-merror:0.237116
## [5]  train-merror:0.236964   test-merror:0.237087
## [6]  train-merror:0.236925   test-merror:0.237083
## [7]  train-merror:0.236802   test-merror:0.236963
## [8]  train-merror:0.236756   test-merror:0.236919
## [9]  train-merror:0.236757   test-merror:0.236923
## [10] train-merror:0.236769   test-merror:0.236927

foret

## ##### xgb.Booster
## raw: 89 Kb
## call:
##   xgb.train(params = list(objective = "multi:softmax", num.class = 2),
##     data = dtrain, nrounds = 10, watchlist = watchlist)
## params (as set within xgb.train):
##   objective = "multi:softmax", num_class = "2", silent = "1"
## xgb.attributes:
##   niter
## callbacks:
##   cb.print.evaluation(period = print_every_n)
##   cb.evaluation.log()
## niter: 10
## evaluation_log:
##     iter train_merror test_merror
##        1     0.238422    0.238670
##        2     0.238297    0.238519
## ---
##        9     0.236757    0.236923
##       10     0.236769    0.236927
```

```r
importance_matrix<-xgb.importance(colnames(dtrain),model=foret)
xgb.ggplot.importance(importance_matrix,measure = "Frequency", rel_to_first = TRUE)
```




```r
# prediction de l'etiquette dataset
#pred <- predict(foret, test[,-7])
#pred

# taux d'erreur sur dtest
#length(which (pred != test[,7]/nrow(test)))
#length

toc <-  Sys.time() -tic
print(toc)

## Time difference of 2.040327 mins
```

10.6.2. Analyse

MODEL	ANALYSE
Modèle Analyse Discriminante Linéaire	Document powerpoint
Modèle Régression Logistique	
Modèle ACP	
Modèle Analyse Factorielle Discriminante	

MRSQ_2015 > transformation devis en sociétaire(s) ?

phase exploratoire !
Trouver les variables discriminantes
Et trouver le modèle statistique
adapté à notre jeu de données

Rappel : les variables qualitatives ne sont pas prises en compte dans la spécification du modèle d'analyse,

car le résultat de l'audit de ces données, montre que ces variables n'apportent rien pour expliquer et prédire la séparation des « Oui » et « Non » de notre variable à expliquer.

SPECIFICATION DU MODELE DES DONNEES	
date	13.fevrier.2017
version	version 3.1
source des données	données MRSQ - année 2015
nombre de variables	112 « features » - variables
nombre d'enregistrements	728674 lignes
périmètre devis concerné	Devis auto (hors 2 roues) et particulier (hors usage professionnel)
problème	classification binaire supervisée **avec <u>CLASSES DESEQUILIBREES</u>**
type d'analyse	modèle de discrimination avec sélection de variables **quantitatives**
Note destiné au lecteur	***la variable nombre de jour qui sépare le 1er devis (delai_1_ctc) non prise en compte pour cette version du rapport***

- SOMMAIRE -

partie 1. : analyse méthodes descriptives p. 5

partie 1.1. : 1er éléments de Datavizualisation p. 5

partie 1.2. : modèles géométriques p. 14

partie 2. : inférence statistique p. 17

partie 2.1. : modèle d'analyse discriminante (lda) p. 19

partie 2.2. : modèle de régression logistique (RegLog) p. 29

A RETENIR : pour faire du Machine Learning sur un problème de classification binaire supervisée <u>avec CLASSES DESEQUILIBREES</u>

<u>BUT</u>: avant de faire de l'apprentissage statistique (Machine Learning), il faut réaliser une phase d'analyse exploratoire des données (i.e. feature engineering) dont les objectifs sont : parmi différents modèles statistiques utilisables, trouver et paramétrer le modèle statistique « le meilleur » pour la phase de machine learning sur notre jeu des données. C'est-à-dire, vérifier les prérequis définis comme il suit :

. Prérequis 1 : identifier les véritables variables discriminantes,

. Prérequis 2 : qualifier le type de relation linéaire ou non entre les variables explicatives et la variable à expliquer (étiquette Oui / Non)

. Prérequis 3 : sur le critère la Parcimonie (Data Science), trouver le modèle le plus simple possible qualifiant le mieux la relation entre les variables explicatives et la variable à expliquer,

. Prérequis 4 : sur le critère de performance du classifieur, mettre en œuvre la procédure de validation croisée pour s'assurer que le modèle statistique retenu peut être généralisé.

<u>RESULTATS</u>: sur ce jeu de données
. Utiliser les outils de la datavizualisation pour comprendre les relations entre les variables **
. En analyse descriptive : ACP fonctionne, AF Discriminante Géométrique ne fonctionne pas ,
. En inférence statistique : Analyse discriminante linéaire et Régression logistique ne fonctionnent pas.

➔ <u>R1</u>: pas de relation linéaire entre les Xi et le Y
➔ <u>R2</u>: classes déséquilibrées du Y,
➔ <u>R3</u>: les Xi discriminantes pas toutes identifiées ?

Pour tous ces modèles :
. 2 variables explicatives AGE_CLI et C_AN_TTC sont « discriminantes et plus petit dénominateur commun à tous les modèles »,
. Convergence « en variable » avec les résultats de l'algorithme de méthode gradient boosting (ou xgboost) utilisé en Machine Learning.

<u>BONNES PRATIQUES</u>: pour les futurs sujets avec phase de Machine Learning (i.e. transformation)
. Passer du temps sur la phase d'analyse exploratoire (« feature engineering »),
. Mobiliser des modèles statistiques différents et variés (« Algo ») pour s'assurer que les 4 prérequis sont vérifiés,
. Mettre en œuvre les tests statistiques adaptés pour vérifier que le modèle retenu « est valide ».
*** entre les Xi, mais aussi entre les Xi et Yi.*

R1: pas de relation linéaire entre les Xi et le Y
. pas de relation « directe / mécanique » entre les variables explicatives et la variable à expliquer (étiquette Oui / Non)

R2: classes déséquilibrées du Y,
. les différents modèles statistiques de classification recherchent à minimiser l'erreur de classement, ils sont implémentés « c'est à dire codés » pour détecter en priorité la classe majoritaire. En conséquence, pour obtenir un classifieur performant en terme de taux de bien classés (i.e. matrice de confusion) le modèle statistique doit prendre en compte et corriger ce déséquilibre de classe de la variable à prédire (i.e. paramètres de l'algorithme).

R3: les Xi discriminantes pas toutes identifiées ?
. « … manque de variables discriminantes … »
. une variable discriminante = variable qui aide à séparer les « Oui » et les « Non ».

Objectif et méthode :
. En parallèle de la mise en œuvre de l'algorithme XGBOOST (Python) pour prédire une classification « OUI » et « NON » avec un % de taux de « biens classés », il faut s'assurer que le modèle statistique utilisé par l'algorithme mobilise bien les bonnes variables discriminantes et que la prédiction de « biens classés » est fiable.
. Construire une grille de modèles statistiques pour déterminer un socle stable et convergent de variables discriminantes, puis identifier un modèle de classification dont la performance de « biens classés » se généralise en phase Machine Learning.

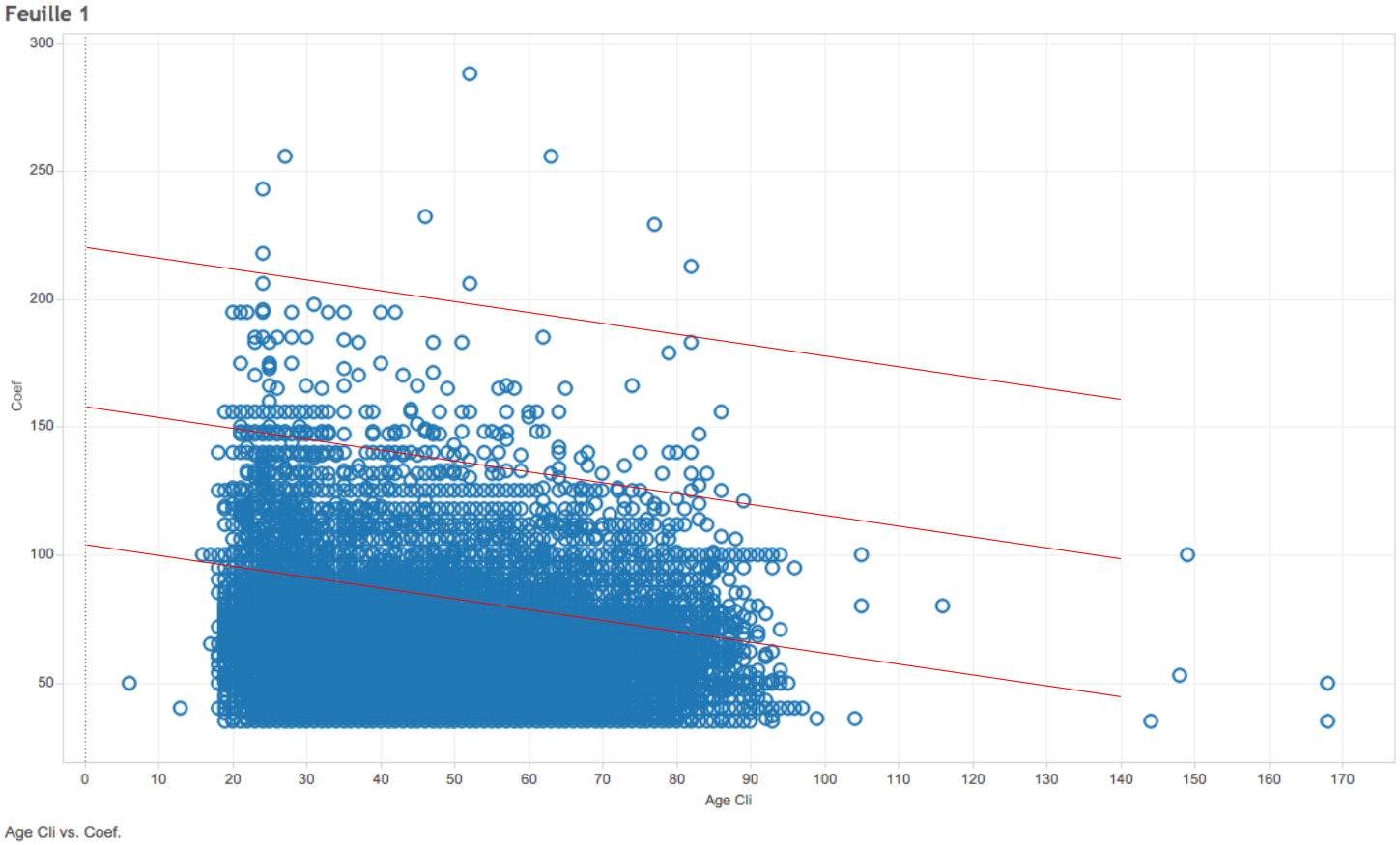
Dataviz !
Les éléments présentés ici ne sont pas des résultats d'enquêtes. Ils sont présentés à titre illustratif. Ils montrent que l'analyste peut visualiser rapidement des relations entre les variables et identifier des questions à mobiliser lors de la spécification du modèle de classification.

MRSQ_2015 > transformation devis en sociétaire(s)
partie 1. : analyse descriptive

partie 1.1. : tableau.com > 1er éléments de Datavizualisation

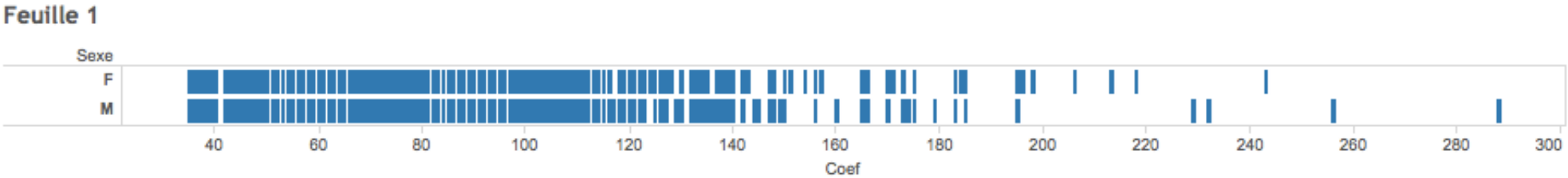
Objectifs:
. visualiser la structure des données,
. qualifier la ou les relations entre les variables explicatives,
. identifier les variables qui participent à expliquer les modalités de la variable à expliquer.

Coef B/M et Age client



Existent-ils des tranches âges et coef ?

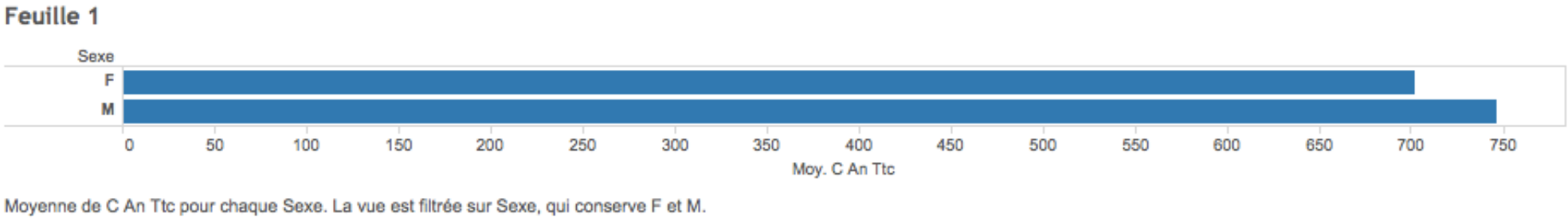
Sexe et Coef B/M



Coef pour chaque Sexe. La vue est filtrée sur Sexe, qui conserve F et M.

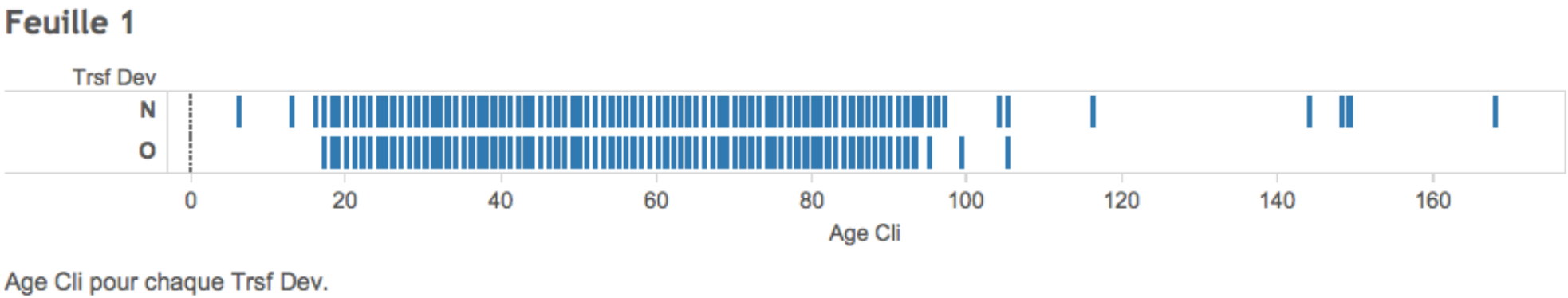
Les coef max des hommes > coef max des femmes

Sexe et Cotise Annuelle



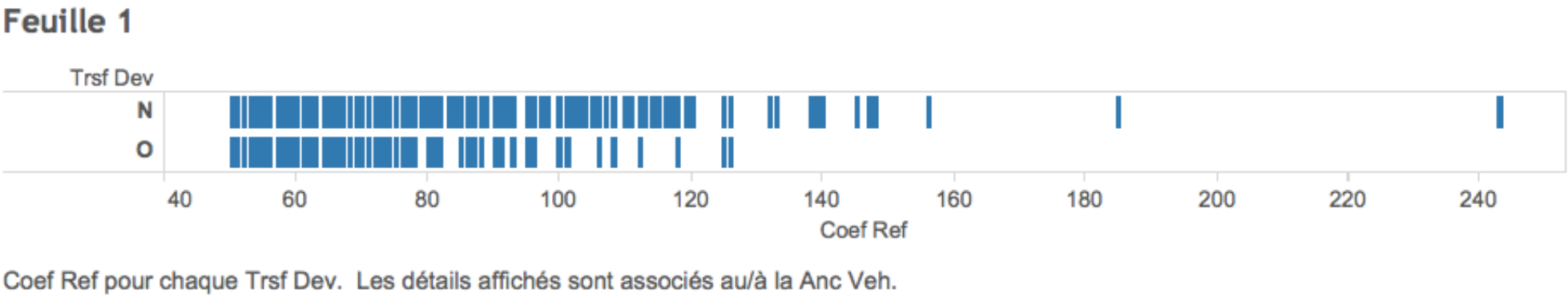
Les cotisations moyennes max des hommes > aux cotises moyennes max des femmes

Transfo Devis et Age Client



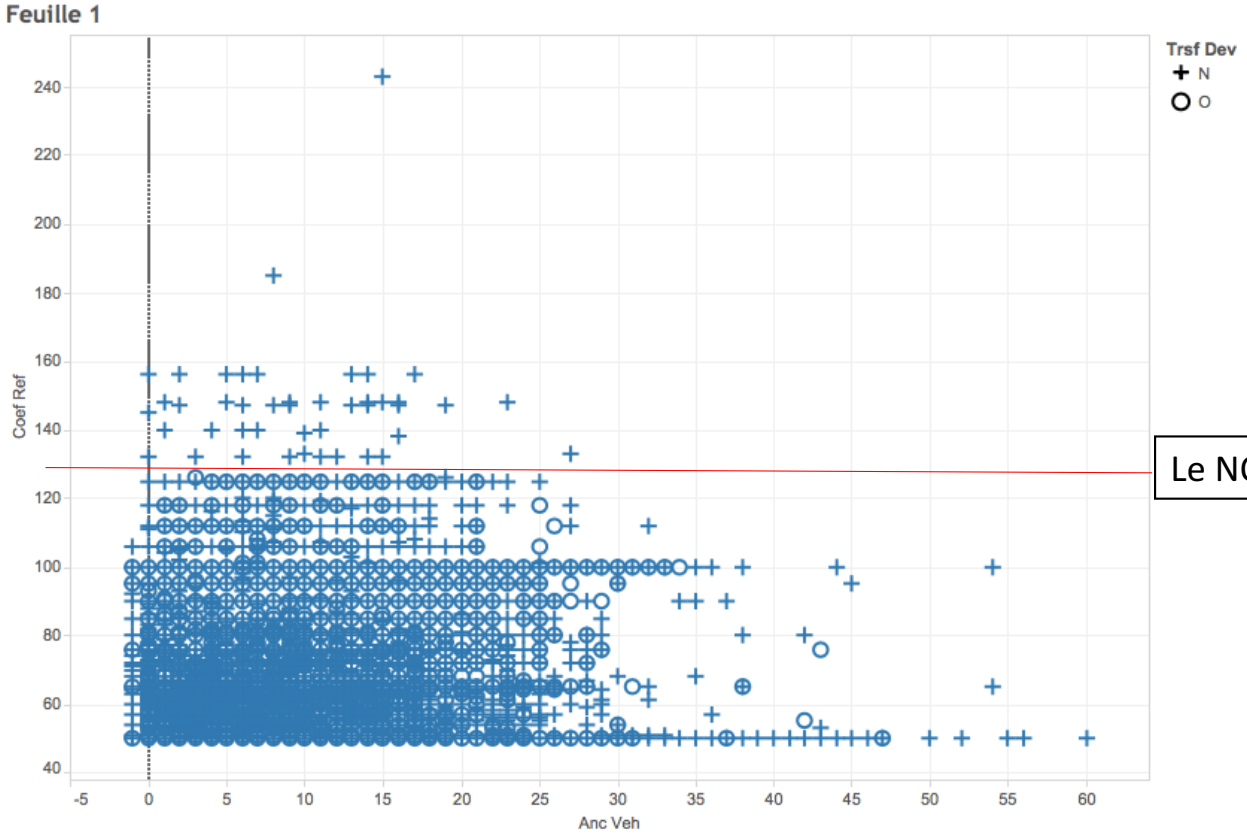
Le NON est-il plus fort chez les âges élevés ?

Transfo Devis et Coefficient B/M



Le NON est-il plus fort chez les gros coefficients B/M?

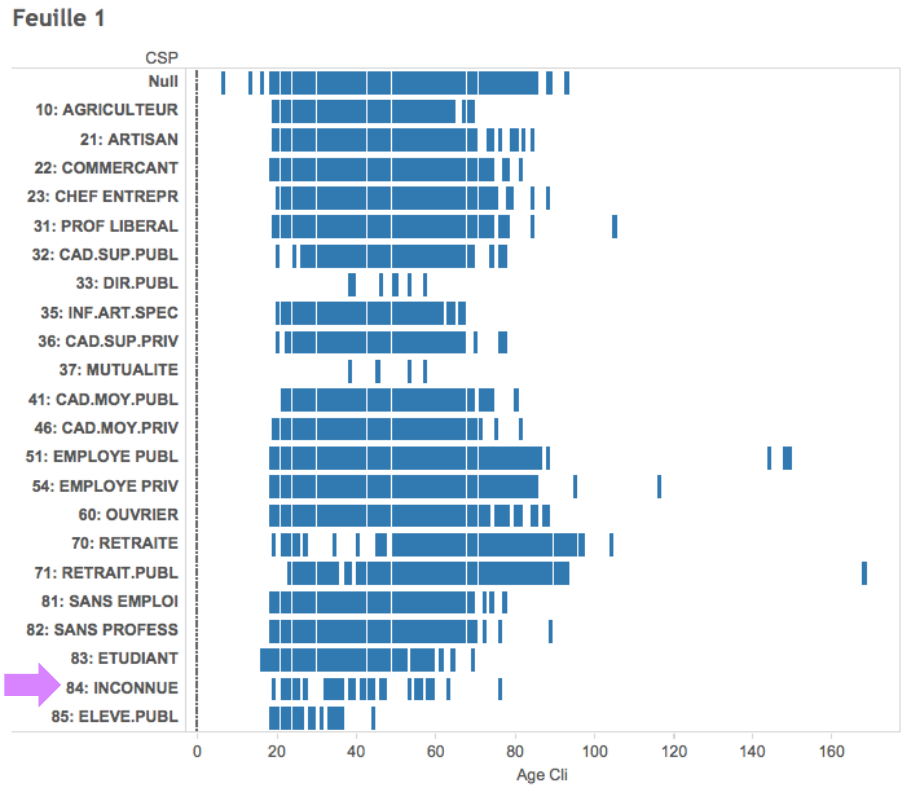
Coef B/M & Ancienneté Véhicule => Transfo devis O/N



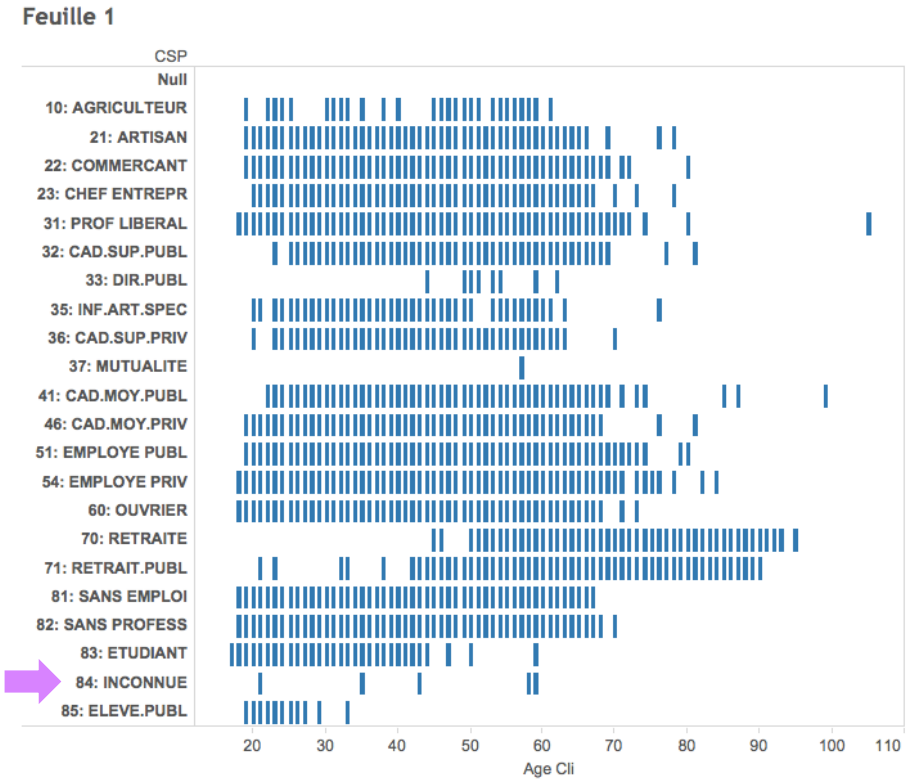
Anc Veh vs. Coef Ref. La forme affiche des détails associés au/à la Trsf Dev.

Le NON est plus fort chez les coef > 120

CSP & Age & Transfo devis O/N



Age Cli pour chaque CSP. Les détails affichés sont associés au/à la Trsf Dev. La vue est filtrée sur Trsf Dev, qui conserve N.



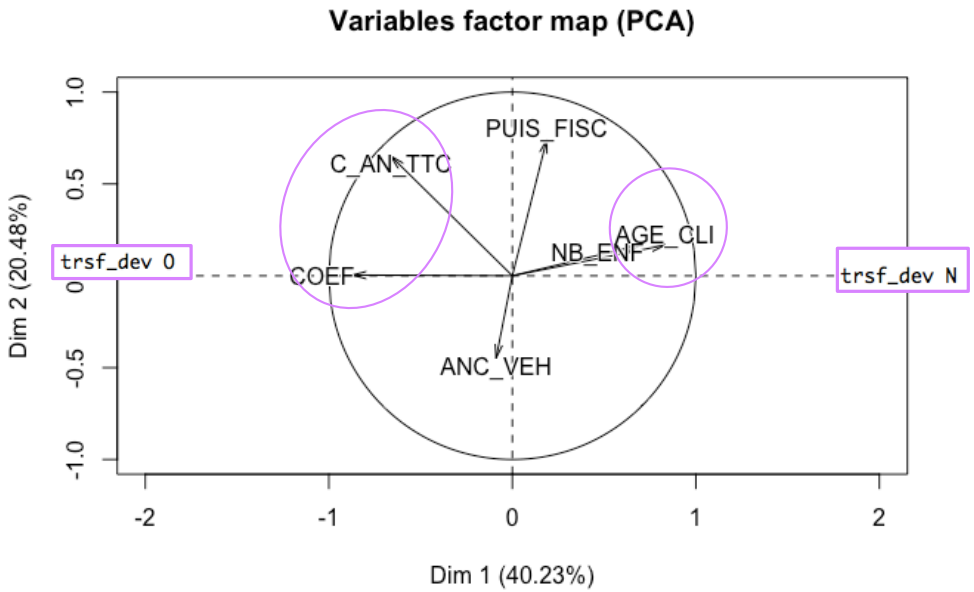
Age Cli pour chaque CSP. Les détails affichés sont associés au/à la Trsf Dev. La vue est filtrée sur Trsf Dev, qui conserve O.

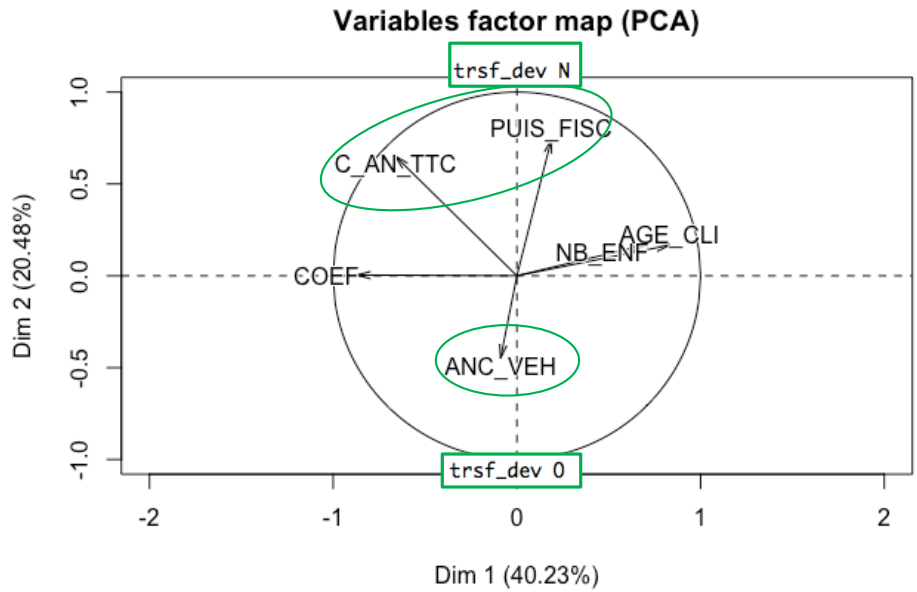
La CSP n'impacte la transformation O/N ?... sauf pour la modalité 84

MRSQ_2015 > transformation devis en sociétaire(s)
partie 1. : analyse descriptive

partie 1.2. : modèles géométriques

PROJECTION DES VARIABLES – PLAN FACTORIEL 1 & 2 = **résume 60.71% de la Variance du nuage initial des données**



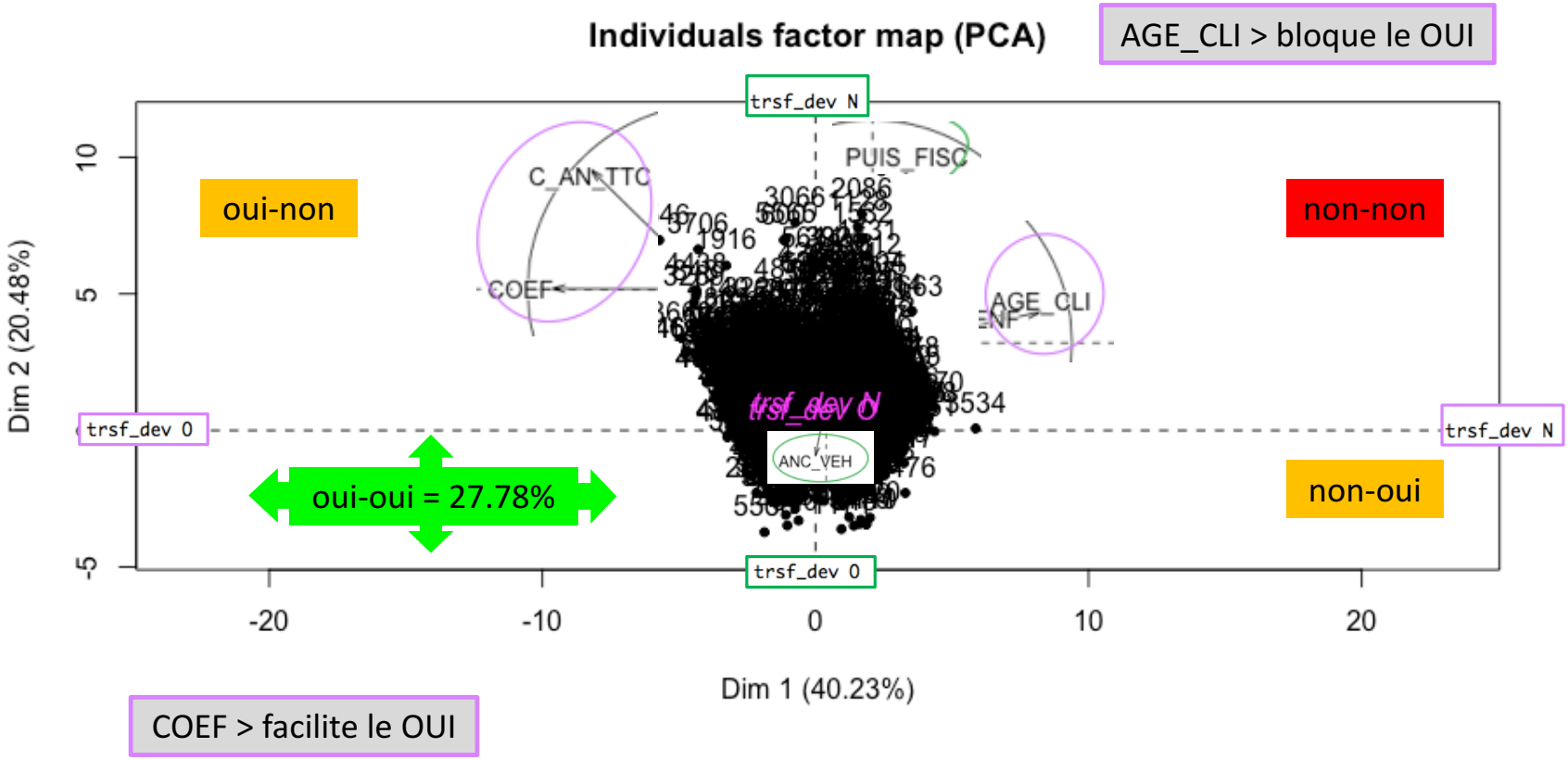


Composante Principale 1 (horizontal) : **l'axe de l'ASSURE**
. 40.23% de la variance expliquée
. COEF - AGE_CLI contribuent et bien représentées
. COEF s'oppose à AGE_CLI
. COEF - C_AN_TTC reliées au OUI de transformation
. AGE_CLI reliée au NON de transformation

Composante Principale 2 (vertical) : **l'axe du VEHICULE**
. 20.48% de la variance expliquée
. PUIS_FISC - C_AN_TTC contribuent et bien représentées
. PUIS_FISC s'oppose ANC_VEH
. PUIS_FISC - C_AN_TTC reliées au NON
. ANC_VEH reliée au OUI

C_AN-TTC > transforme ou bloque le OUI

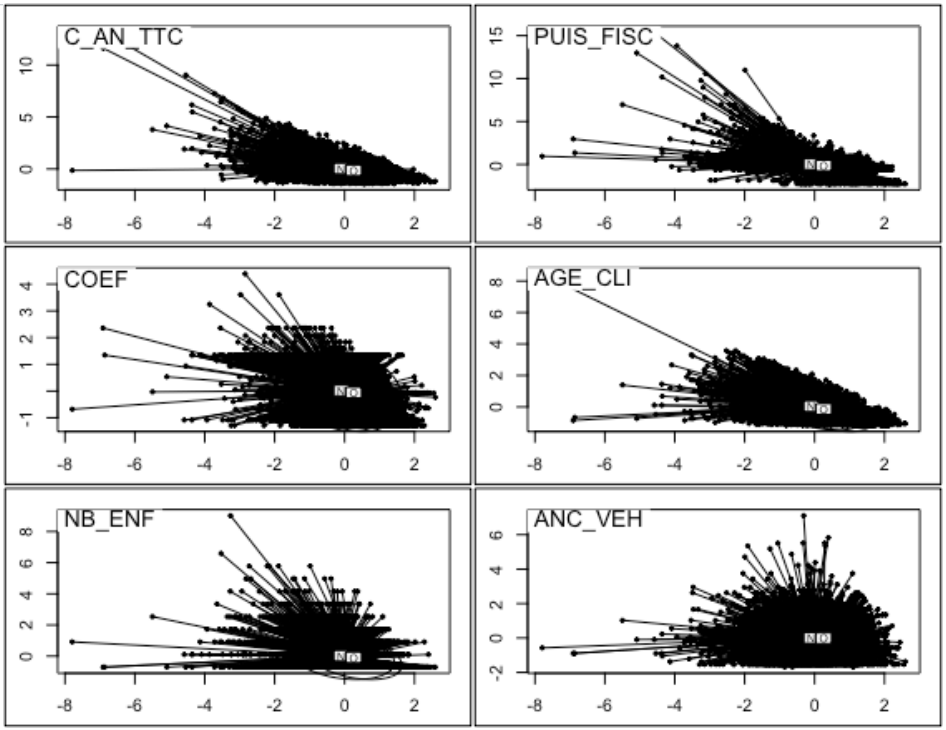
PROJECTION DES INDIVIDUS DANS LE PLAN DES COMPOSANTES PRINCIPALES



COEF > facilite le OUI

PROJECTION DES INDIVIDUS – CARTES AXE FACTORIEL DISCRIMINANTE (1 seul axe factoriel)

```
> afd1=lda(cla~.,tab)
> round(afd1$scaling,2)
            LD1
C_AN_TTC  -0.51
PUIS_FISC -0.27
COEF      -0.56
AGE_CLI   -0.95
NB_ENF    -0.11
ANC_VEH   -0.15
> afd2=discrimin(dudi.pca(tab,scan=FALSE),cla,scan=FALSE,nf=2)
> afd2
Discriminant analysis
call: discrimin(dudi = dudi.pca(tab, scan = FALSE), fac = cla, scannf = FALSE,
    nf = 2)
class: discrimin

$nf (axis saved) : 1

eigen values: 0.02488
```



RESULTATS :

. 1 seul axe discriminant ressorti par cette méthode

. Graphiquement (géométrie), cette technique n'arrive pas avec chaque variable explicative à séparer les « OUI » des « NON ».

Pourquoi ? : cela veut dire qu'il n'y a pas de relation linéaire entre la variable à expliquer « OUI / NON » et les 6 variables quantitatives explicatives.

MRSQ_2015 > transformation devis en sociétaire(s)

partie 2. : inférence statistique
> modèle d'analyse discriminante linéaire (i.e. lda)

rappel 1 – conditions de mise en eouvre du lda:

. hypothèse de multi-normalité des variables explicatives

. hypothèse d'homoscédasticité entre la variable à expliquer et les variables explicatives (i.e. relation linéaire)

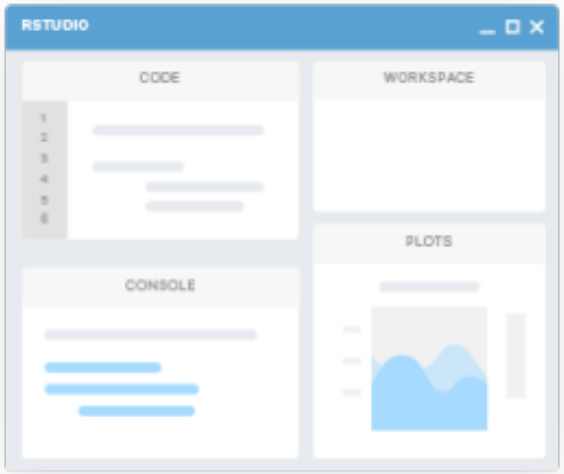
rappel 2 – pourquoi ce modèle lda:

. Lors d'une phase exploratoire d'enquête, s'assurer que le futur modèle d'apprentissage supervisé garantit,

(1) le modèle est bien spécifié,

(2) le modèle spécifié contient bien les variables explicatives discriminantes de la variable d'intérêt.

$$\begin{cases} D(y_1, X) = a_0 + a_1 \times X_1 + \cdots + a_J \times X_J \\ D(y_2, X) = b_0 + b_1 \times X_1 + \cdots + b_J \times X_J \\ \cdots \end{cases}$$



MRSQ_2015 > transformation devis en sociétaire(s)

partie 2. : inférence statistique

« … expliquer le modèle supervisé Oui / Non… »

MRSQ_2015 > transformation devis en sociétaire(s)
partie 2. : inférence statistique

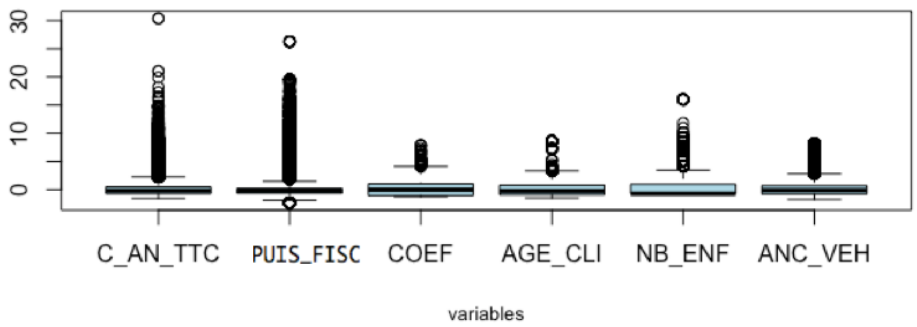
partie 2.1. : modèle d'analyse discriminante linéaire

avec des classes déséquilibrées de la variable à prédire.

SPECIFICATION DU MODELE DES DONNEES AVEC VARIABLES QUANTITATIVES

```
'data.frame':   2521044 obs. of  6 variables:
 $ C_AN_TTC : num  1.29 1.29 1.29 1.29 1.29 ...
 $ PUIS_FISC: num  -0.221 -0.221 -0.221 -0.221 -0.221 ...
 $ COEF     : num  1.43 1.43 1.43 1.43 1.43 ...
 $ AGE_CLI  : num  -0.724 -0.724 -0.724 -0.724 -0.724 ...
 $ NB_ENF   : num  -0.737 -0.737 -0.737 -0.737 -0.737 ...
 $ ANC_VEH  : num  -0.125 -0.125 -0.125 -0.125 -0.125 ...
```

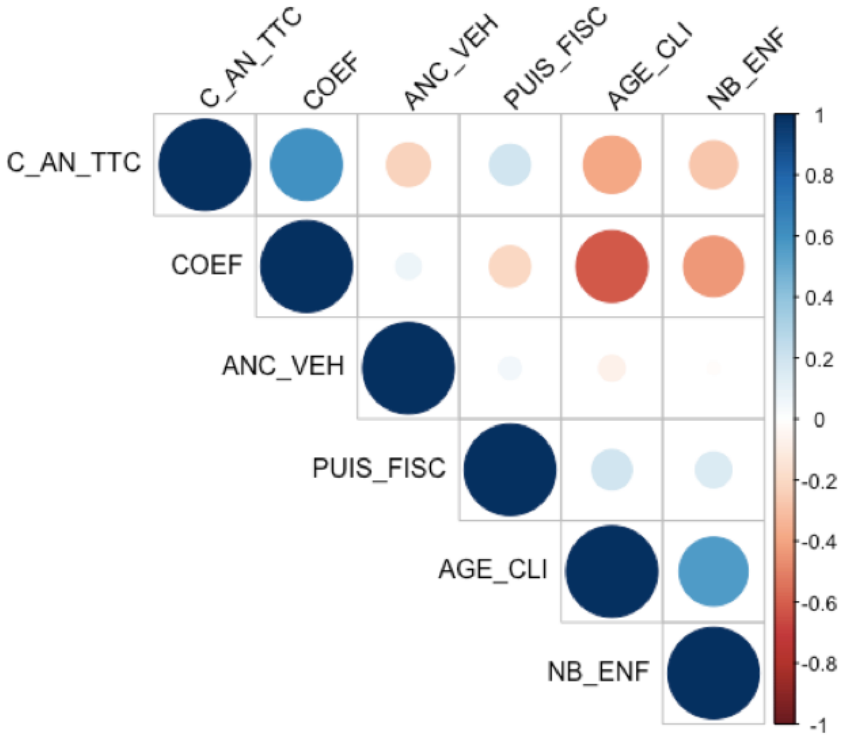
BOXPLOT VARIABLES CENTREES-REDUITES



ANALYSE :

. Après centrage-réduction, les 6 variables explicatives présentent encore une fluctuation importante et hétérogène (i.e. à prendre en compte pour l'interprétation des modèles d'analyse).

MATRICE DES CORRELATIONS



ANALYSE :

. la Cotisation Annuelle est corrélée avec le Coefficient B/M,

. le Coefficient est corrélé avec Age Client.

COMMENTAIRE :

. ces éléments sont illustratifs du modèle tarifaire MATMUT pour un contrat Auto.

ANALYSE DISCRIMINANTE LINEAIRE MODELE COMPLET (6 var. explicatives)

```
Call:
lda(dq2norm_y$trsf_dev ~ ., data = dq2norm_y)

Prior probabilities of groups:
       N        O
0.722152 0.277848

Group means:
     C_AN_TTC   PUIS_FISC        COEF     AGE_CLI      NB_ENF     ANC_VEH
N  0.06174682  0.04834374  0.02012921  0.03479996  0.02729816 -0.04189115
O -0.16048557 -0.12564976 -0.05231764 -0.09044824 -0.07095039  0.10887887

Coefficients of linear discriminants:
                LD1
C_AN_TTC  -0.5074008
PUIS_FISC -0.3589032
COEF      -0.4774354
AGE_CLI   -0.6411595
NB_ENF    -0.2028685
ANC_VEH    0.3199303
```

ANALYSE :

1. quelles sont les variables discriminantes :

. critère de sélection des variables dans les axes discriminants

. on retient les var dont le coeff. > 0.45

. les variables discriminantes entre les "oui" et les "non »

. **COEF > C_AN_TTC > AGE_CLI sont sélectionnées**

2. on qualifie les groupes "OUI" et "NON » :

. chez les oui,

COEF, AGE_CLI & C_AN_TTC qualifient ce groupe (coordonnées négatives sur l'axe)

. chez les non,

C_AN_TTC, AGE_CLI & COEF qualifient ce groupe (coordonnées positives sur l'axe)

. A noter : la distribution empirique des 72% de Non I 28% de Oui qui lors de la phase de Machine Learning fera apprendre du Non.

ANALYSE DISCRIMINANTE LINEAIRE MODELE SELECTIONNE (3 var. explicatives)

```
Call:
lda(dq2norm_y$trsf_dev ~ C_AN_TTC + AGE_CLI + COEF, data = dq2norm_y)

Prior probabilities of groups:
       N        O
0.722152 0.277848

Group means:
     C_AN_TTC     AGE_CLI        COEF
N  0.06174682  0.03479996  0.02012921
O -0.16048557 -0.09044824 -0.05231764

Coefficients of linear discriminants:
                LD1
C_AN_TTC -0.9064616
AGE_CLI  -0.8761939
COEF     -0.2136779
```

ANALYSE :

1. quelles sont les variables discriminantes :

. critère de sélection des variables dans les axes discriminants

. on retient les var dont le coeff. > 0.5

. les variables discriminantes entre les "oui" et les "non »

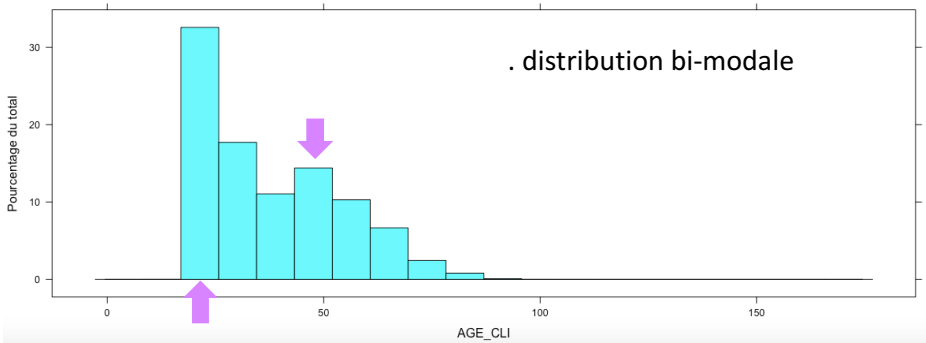
. AGE_CLI > C_AN_TTC sont sélectionnées

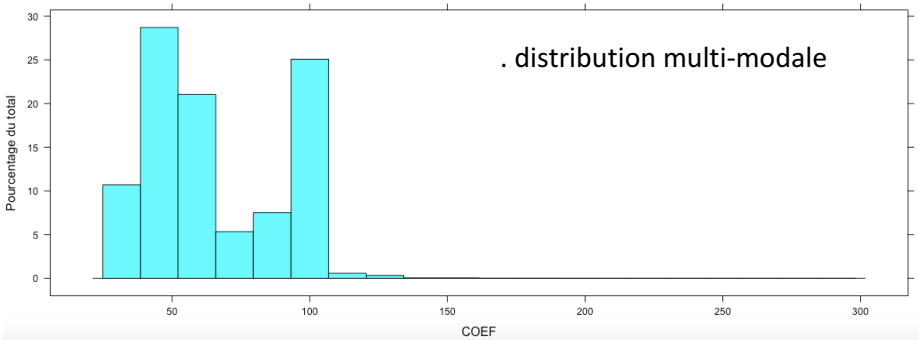
2. on qualifie les groupes "OUI" et "NON » :

. chez les oui : AGE_CLI > C_AN_TTC (coordonnées négatives sur l'axe)

. chez les non : C_AN_TTC > AGE_CLI (coordonnées positives sur l'axe)

POPULATION TOTALE : HISTOGRAMME DES VARIABLES DISCRIMINANTES





ANALYSE DISCRIMINANTE LINEAIRE MODELE SELECTIONNE (2 var. explicatives)

```
Call:
lda(dq2norm_y$trsf_dev ~ C_AN_TTC + AGE_CLI, data = dq2norm_y)

Prior probabilities of groups:
       N        O
0.722152 0.277848

Group means:
     C_AN_TTC     AGE_CLI
N  0.06174682  0.03479996
O -0.16048557 -0.09044824

Coefficients of linear discriminants:
                 LD1
C_AN_TTC -1.0095862
AGE_CLI  -0.7891436
```

ANALYSE :

. analyse résultats du modèle 2 variables discriminantes :

confirmation des résultats du modèle à 3 variables discriminantes

1. quelles sont les variables discriminantes

. critère de sélection des variables dans les axes discriminants

. on retient les var dont le coeff. > 0.5

. les variables discriminantes entre les "oui" et les "non »

. AGE_CLI > C_AN_TTC sont sélectionnées

2. on qualifie les groupes "OUI" et "NON »

. chez les oui : AGE_CLI > C_AN_TTC (coordonnées négatives sur l'axe)

. chez les non : C_AN_TTC > AGE_CLI (coordonnées positives sur l'axe)

ANALYSE DISCRIMINANTE LINEAIRE MODELE SELECTIONNE (2 var. explicatives & sélection du prior)

```
Call:
lda(dq2norm_y$trsf_dev ~ C_AN_TTC + AGE_CLI, data = dq2norm_y,
    prior = c(0.5, 0.5))

Prior probabilities of groups:
  N   O
0.5 0.5

Group means:
     C_AN_TTC     AGE_CLI
N  0.06174682  0.03479996
O -0.16048557 -0.09044824

Coefficients of linear discriminants:
                 LD1
C_AN_TTC -1.0095862
AGE_CLI  -0.7891436
```

ANALYSE :

. analyse résultats du modèle 2 variables discriminantes :

Avec modification du « prior » et utilisation de la distribution a priori optimale (0.5;0.5)

confirmation des résultats du modèle à 2 variables.

CONCLUSION : sur le critère de la PARCIMONIE, **le modèle d'analyse discriminante (lda) à 2 variables est suffisant pour la phase « Machine Learning ».**

ANALYSE DISCRIMINANTE LINEAIRE MODELE SELECTIONNE (2 var. explicatives)

matrice de confusion - indicateur de performance du classifieur - lda avec 2 variables explicatives

```
> confusionMatrix(data=prev, reference=dq2norm_y$trsf_dev,mode="prec_recall")
Confusion Matrix and Statistics

          Reference
Prediction       N       O
         N 1820577  700467
         O       0       0

               Accuracy : 0.7222
                 95% CI : (0.7216, 0.7227)
    No Information Rate : 0.7222
    P-Value [Acc > NIR] : 0.5003

                  Kappa : 0
 Mcnemar's Test P-Value : <2e-16

              Precision : 0.7222
                 Recall : 1.0000
                     F1 : 0.8387
             Prevalence : 0.7222
         Detection Rate : 0.7222
   Detection Prevalence : 1.0000
      Balanced Accuracy : 0.5000

       'Positive' Class : N
```

ANALYSE :

. le taux d'erreur est important > à 10% (i.e. 27.78%)

. la prédiction du modèle lda avec ces données = 0%

. la capacité à séparer les vrais - des faux événements = 50% (1 chance / 2)

CONCLUSION:

. Le modèle est contraint par le modèle des données.

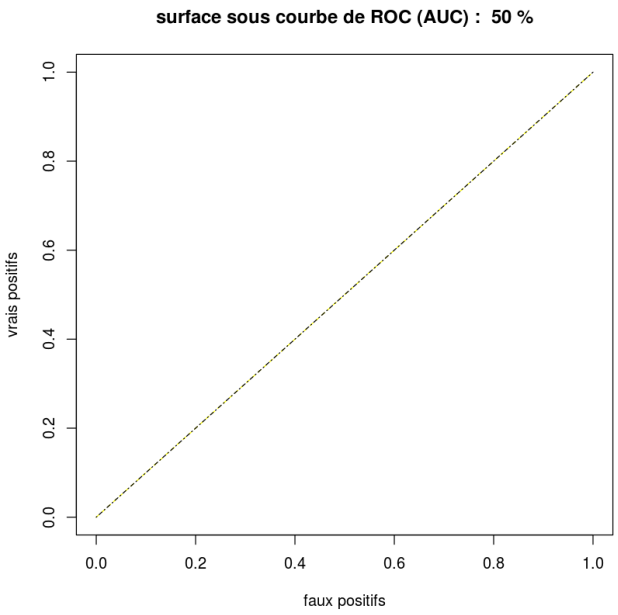
Indicateur	Taux Erreur	Précision	Se: Sensibilité	Sp: Spécificité	Rappel	F-mesure	ROC	indice de Gini
formule	(1-((TP+TN)/N))*100	((TP+TN)/(P+N))*100	(TP/(TP+FN))*100	(TN/(TN+FP))*100	(TP/(TP+FN))*100	(2*Précision*Rappel)/(Précision+Rappel)	AUC	IG=(2*AUC)-1
detail	poids des mal classés dans la prédiction totale	Accuracy : Mesure la capacité du système à refuser les solutions non-pertinentes	prob d'un individu d'appartenir à 1 classe sachant qu'il devrait y appartenir	proba d'un individu n'appartienne pas à la classe à juste titre	Mesure la capacité du système à donner toutes les solutions pertinentes.	Mesure la capacité du système à donner toutes les solutions pertinentes et à refuser les autres	capacité à séparer les vrais, des faux événements	" Accuracy Ratio" - plus il est élevé - meilleur est le modèle
	27,78%	72,22%	0,00%	100%	0,00%	#DIV/0!	50%	0%

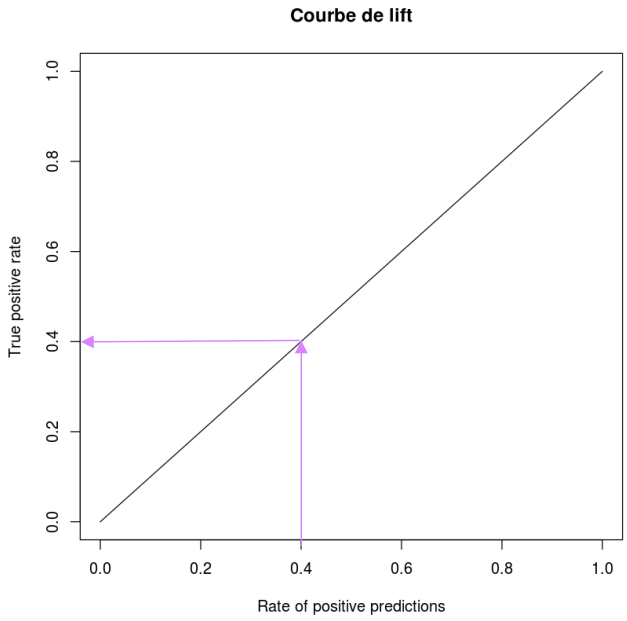
CONCLUSION : sur le critère de Performance de taux de biens classés, **le modèle d'analyse discriminante (lda) à 2 variables n'est pas retenu pour la phase « Machine Learning ». CAUSE PRINCIPALE : des classes déséquilibrées de la variable à prédire.**

➔ ACTION : corriger le déséquilibre dans la spécification du modèle.

ANALYSE DISCRIMINANTE LINEAIRE MODELE SELECTIONNE (2 var. explicatives)

Courbe de ROC / Courbe de Lift - lda avec 2 variables explicatives





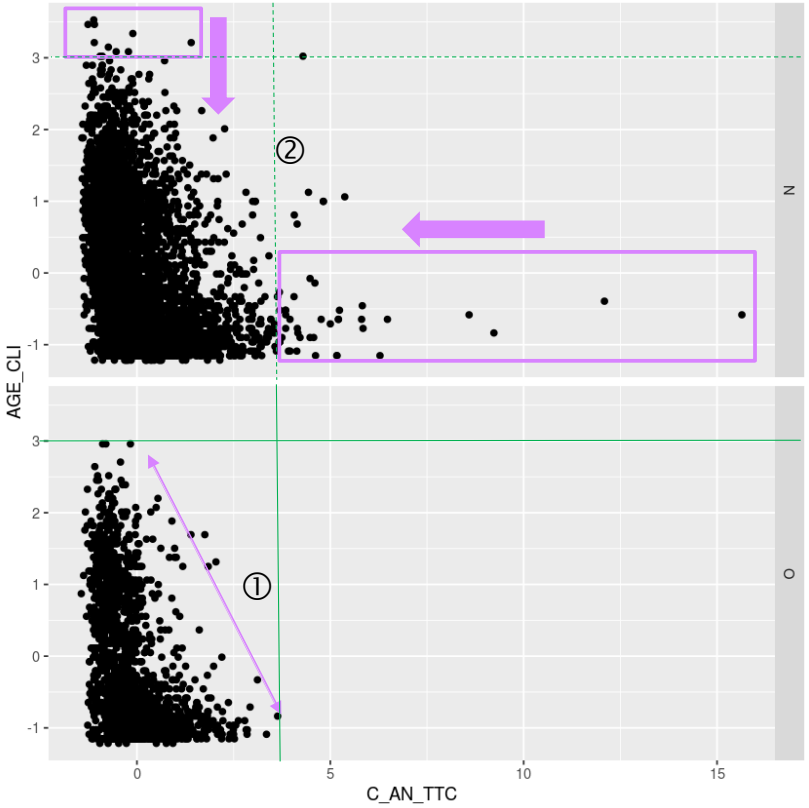
ANALYSE :

. la courbe ROC / LIFT du modèle lda est alignée sur la courbe de prédiction basée sur la bissectrice du hasard.

. la prédiction de mon modèle lda ne fait pas mieux que "le modèle du hasard" (résultat unitaire pour ROC / LIFT).

ANALYSE DISCRIMINANTE LINEAIRE MODELE SELECTIONNE (2 var. explicatives)

VISUALISATION DES GRAPPES D'INDIVIDUS (cluster) avec modèle LDA sélectionné 2 variables explicatives



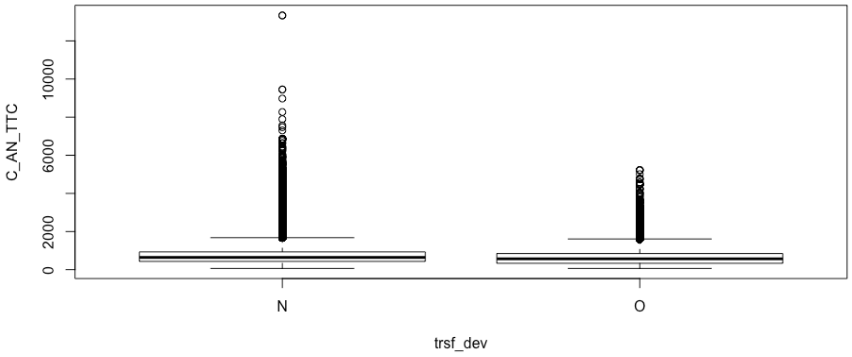
ANALYSE :
. croisement var cotisation & âge pour expliquer l'étiquette Oui / Non devis transformé

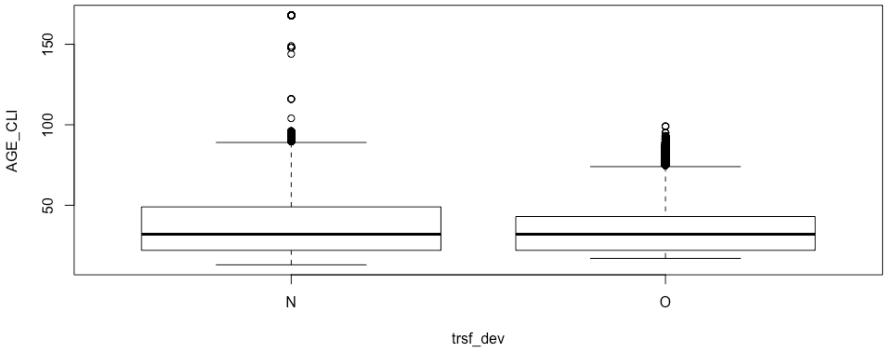
Analyse en diagonale :
. résultat : [âge = vieux] en relation avec [cotisation = faible] alors étiquette = Non
. résultat : [âge = jeune] en relation avec [cotisation = élevée] alors étiquette = Non

COMMENTAIRES :
. les résultats empiriques sur les segments jeunes et vieux sont-ils connus / inconnus par les métiers ?

ANALYSE DISCRIMINANTE LINEAIRE MODELE SELECTIONNE (2 var. explicatives)

POUVOIR DISCRIMANT DES 2 VARIABLES EXPLICATIVES RETENUES





ANALYSE :

. les distributions des 2 variables Cotisation Annuelle et AGE CLIENT sont différentes selon les grappes d'individus OUI / NON.

(pour le détail se rendre aux résultats des tests d'égalité de moyenne / variance de ces 2 variables notebook Jupyter)

MRSQ_2015 > transformation devis en sociétaire(s)

partie 2. : inférence statistique

partie 2.2. : modèle de régression logistique

MODELE DE REGRESSION LOGISTIQUE (modèle complet)

ANALYSE :

. Résultat nominal / lda

. **Résultat naïf** = on garde seulement 2 variables !

```
Call:  glm(formula = trsf_dev ~ ., family = binomial, data = dq2norm_y)

Coefficients:
(Intercept)    C_AN_TTC   PUIS_FISC        COEF     AGE_CLI      NB_ENF     ANC_VEH
   -0.98966    -0.21440    -0.15202    -0.16488    -0.23867    -0.07483     0.10822

Degrees of Freedom: 2521043 Total (i.e. Null);  2521037 Residual
Null Deviance:      2979000
Residual Deviance: 2912000      AIC: 2912000
```

MODELE DE REGRESSION LOGISTIQUE (modèle sélectionné 2 var)

```
Call:  glm(formula = trsf_dev ~ C_AN_TTC + PUIS_FISC + COEF + AGE_CLI +
    NB_ENF + ANC_VEH, family = binomial, data = dq2norm_y)

Coefficients:
(Intercept)    C_AN_TTC   PUIS_FISC        COEF     AGE_CLI      NB_ENF     ANC_VEH
   -0.98966    -0.21440    -0.15202    -0.16488    -0.23867    -0.07483     0.10822

Degrees of Freedom: 2521043 Total (i.e. Null);  2521037 Residual
Null Deviance:      2979000
Residual Deviance: 2912000      AIC: 2912000
```

Procédure sélection automatique « step » + méthode backward, je retire des variables quand je minimise le critère AIC

Procédure sélection à la main, je retire des variables quand je minimise le critère AIC

```
Call:
glm(formula = trsf_dev ~ C_AN_TTC + AGE_CLI, family = binomial,
    data = dq2norm_y)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.1101  -0.8455  -0.7443   1.4330   3.1465

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.985035   0.001445  -681.8   <2e-16 ***
C_AN_TTC    -0.373313   0.001804  -206.9   <2e-16 ***
AGE_CLI     -0.273062   0.001620  -168.6   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2979417  on 2521043  degrees of freedom
Residual deviance: 2923050  on 2521041  degrees of freedom
AIC: 2923056

Number of Fisher Scoring iterations: 4

> 2923056-2912000
[1] 11056
```

CONCLUSION : sur le critère AIC min, **le modèle régression logistique à 2 variables n'est pas le meilleur modèle ➔ on devrait garder les 6 variables (?)**

Résultat : ce modèle RegLog n'est pas bien spécifié pour ce Modèle des données ➔ pas de garantie d'avoir sourcé toutes les var. explicatives réellement discriminantes et interrogation sur la nature linéaire entre var. explicatives et var. à expliquer (var. *delai_1_ctc* de Cyrille et mobiliser d'autres modèles Logit / t-SNE « représentation graphique » pour croiser les résultats des différents modèles).

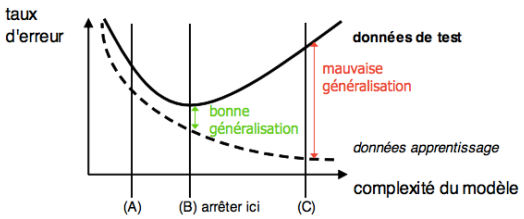
Annexe 1:

Pourquoi faut il simplifier un modèle de classification afin d'obtenir un bon taux de classement en phase de Machine Learning

. sélectionner les véritables variables discriminantes et réduire le nombre de variables,

. utiliser le modèle de classification le plus adapté à notre problème de classification et qui « généralise bien en Machine Learning »

Taux d'erreur en fonction de la complexité du modèle



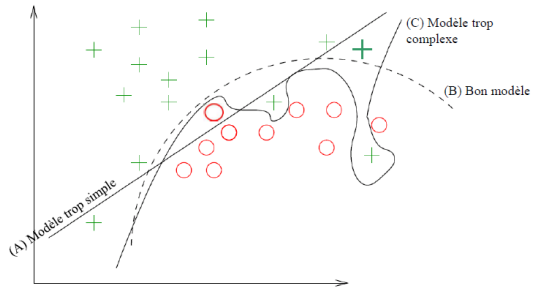
Théorème de Vapnik :

$$R < R_{emp} + \sqrt{\frac{h\,(\log(2n/h)+1) - \log(\alpha/4)}{n}}$$

Qualités attendues d'une technique prédictive 1/2

- La précision
  - le taux d'erreur doit être le plus bas possible, et l'aire sous la courbe ROC la plus proche possible de 1
- La robustesse
  - être le moins sensible possible aux fluctuations aléatoires de certaines variables et aux valeurs manquantes
  - ne pas dépendre de l'échantillon d'apprentissage utilisé et bien se généraliser à d'autres échantillons
- La concision
  - les règles du modèle doivent être les plus simples et les moins nombreuses possible

Sur-apprentissage en classement



Source : Olivier Bousquet

07/02/2014 © Stéphane Tufféry - Usage réservé à l'Université Rennes 1 94

Qualités attendues d'une technique prédictive 2/2

- Des résultats explicites
  - les règles du modèle doivent être accessibles et compréhensibles
- La diversité des types de données manipulées
  - toutes les méthodes ne sont pas aptes à traiter les données qualitatives, discrètes, continues et… manquantes
- La rapidité de calcul du modèle
  - un apprentissage trop long limite le nombre d'essais possibles
- Les possibilités de paramétrage
  - dans un classement, il est parfois intéressant de pouvoir pondérer les erreurs de classement, pour signifier, par exemple, qu'il est plus grave de classer un patient malade en « non-malade » que l'inverse

Source : Tuffery_ _Master_Rennes_2013-2014_ _Data_Mining_ _Presentation

Data Scientist – Groupe CREDIT MUTUEL

Annexe 2:
Pourquoi les variables qualitatives ne participent pas à la construction du modèle d'analyse pour expliquer et prédire les « OUI » et « NON ».

➔ modèle d'Analyse en Correspondance Multiple afin d'identifier les variables discriminantes qui séparent les « OUI » et « NON »

3. choisir le nb d'axes pour faire notre modèle d'analyse de sélection de variables discriminantes ?

eigenvalue	percentage of variance	cumulative percentage of variance	
dim 1	0.34	0.49	0.49
dim 2	0.32	0.46	0.95
dim 3	0.31	0.45	1.40
dim 4	0.29	0.42	1.82
dim 5	0.28	0.41	2.23

ANALYSE : choix du nombre d'axe selon la méthode du coude ?
. la faible valeur d'inertie récupérée par les axes factoriels,
. graphiquement, pas de coude identifié.

CONCLUSION :
. on se trouve dans l'impossibilité de choisir le nombre d'axe factoriel et donc de produire une analyse factorielle pour construire des relations entre la variable qualitative trsf_dev et les autres variables.

=> sur le jeu de données initiales, il est difficile d'expliquer clairement le rôle des variables qualitatives sur la variable transformation des devis.

MODEL	ANALYSE
Modèle Random forest « forêts aléatoires »	Document powerpoint
Modèle XGboost	

MRSQ_2015 > transformation devis en sociétaire(s)

partie 2. : inférence statistique

partie 2.3. : autres modèles

DATASET	CHIFFRES
nb enregistrements	614048
nb var quanti	6
nb var quali	0
var. transformation_devis	
Etiquette-N	73.31 %
Etiquette-0	26.68 %

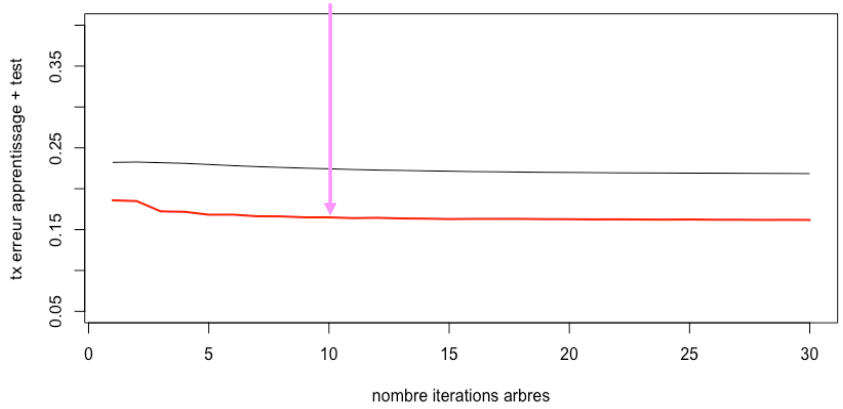
randomForest = modèle complet 6 VARIABLES QUANTITATIVES > Taux BIEN CLASSES OOB =

83.82%

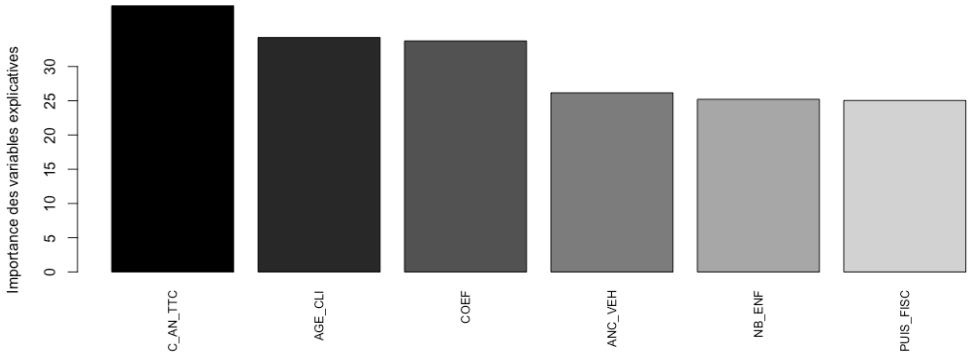
```
        OOB estimate of  error rate: 21.85%
Confusion matrix:
        N      O class.error
N 1457868 180776   0.1103205
O  314882 315411   0.4995804
                Test set error rate: 16.18%
Confusion matrix:
       N     O class.error
N 169509 12767  0.07004213
O  28024 41804  0.40132898
```

83.82 % = (100 -16.18) %

Tx Erreur apprentissage + test convergent bien et rapidement



Importances des variables



Var. discriminantes = idem autres modèles stats.

A RETENIR :
. randomForest Modèle prédictif le moins « overfitted » et qui « généralise bien ».

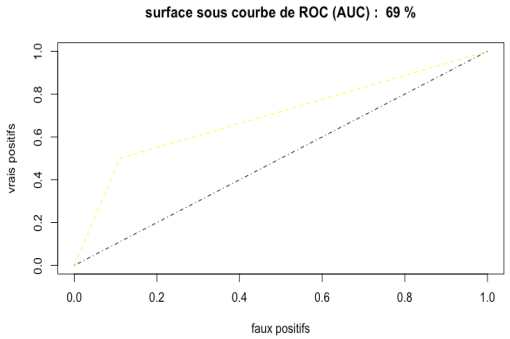
PREUVE :
. Delta Erreur OOB (Apprentissage - Test) = 21.85% - 16.18% = 5.67 % (proche 5%)
. Gain (Tx bien classés - Tx Distribué [NON]) = 83.82% - 73.31% = 10.51% (sup à 10%)

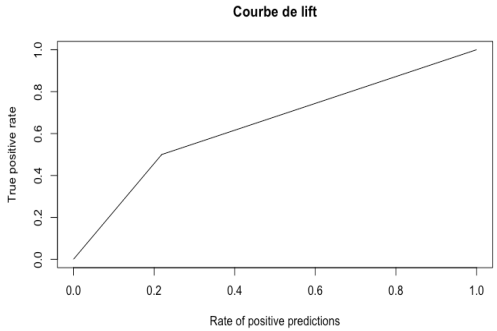
randomForest = modèle complet 6 VARIABLES QUANTITATIVES > F1 score = 85.44%

F-mesure : mesure la capacité du système à donner toutes les solutions pertinentes et à refuser les autres

sur données test

```
          Accuracy : 0.7811
            95% CI : (0.7806, 0.7817)
 No Information Rate : 0.7222
 P-Value [Acc > NIR] : < 2.2e-16

               Kappa : 0.4165
Mcnemar's Test P-Value : < 2.2e-16

            Precision : 0.8222
               Recall : 0.8893
                   F1 : 0.8544
           Prevalence : 0.7222
       Detection Rate : 0.6423
 Detection Prevalence : 0.7812
    Balanced Accuracy : 0.6946
```





Résultats RF : N. MOKHEDDES

Accuracy : 0.8123
95% CI : (0.8103, 0.8143)
No Information Rate : 0.7862
P-Value [Acc > NIR] : < 2.2e-16

Kappa : 0.3584
Mcnemar's Test P-Value : < 2.2e-16

Sensitivity : 0.9291
Specificity : 0.3829
Pos Pred Value : 0.8470
Neg Pred Value : 0.5948
Prevalence : 0.7862
Detection Rate : 0.7305
Detection Prevalence : 0.8624
Balanced Accuracy : 0.6560

'Positive' Class : 0

A RETENIR :
. Performance Moyenne, Courbes ROC - LIFT

MACHINE LEARNING
. méthode pour comparer la performance de 2 classifieurs = réaliser 1 test statistique

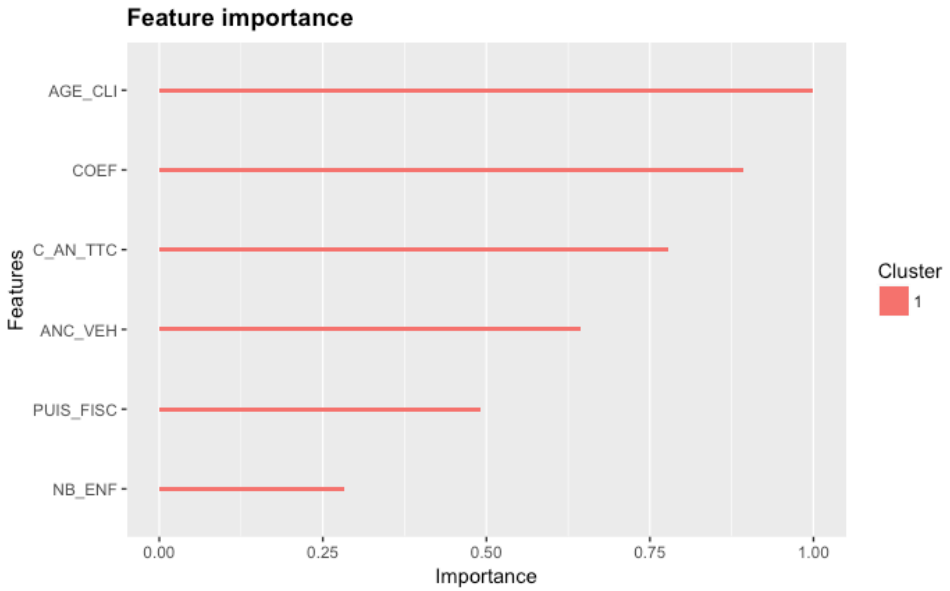
Test de Mc Nemar ?
. le taux 0.8123 est il significativement meilleur que le taux 0.7811

. RESULTATS :
McNemar's chi-squared = 20064, df = 1, p-value < 2.2e-16
= le test est accepté.

La performance du classifieur RF de Naima > La performance du classifieur RF de Naima.

XGBOOST = MODELE COMPLET 6 VARIABLES QUANTITATIVES > Taux BIEN CLASSES = 76.25%

```
niter: 10
evaluation_log:
    iter train_merror test_merror
       1     0.238359    0.238548
       2     0.237093    0.237128
---
       9     0.237444    0.237581
      10     0.237447    0.237581
```



CONCLUSION :
. sur le critère de Performance de taux de biens classés, **le modèle Gradient Boosting (packag XGboost) à 6 variables est moins performant que le modèle de forêt aléatoire (randomForest) avec les paramètres d'algorithme par défaut.**
. sur le critère d'importance des « features », le socle de variables discriminantes du modèle ne converge par vers le socle convergent des autres modèles.

➔ ACTION : Algorithme à paramétrer, arbitrage temps de paramétrage du modèle optimal XGboost vs. gain de taux de bien classés obtenu ?

10.7. Calculs numériques - Modélisations Période 2

10.7.1. Formalisation du « dataset » de travail

00_prepadata.R

lepicaut

Fri Sep 1 21:35:04 2017

```
############
## FORMATAGE D UN DATASET UNIQUE POUR TOUS LES CLASSIFIEURS
############


#*****************************************************************************************************
*****************************************************************************************
#*****************************************************************************************************
*****************************************************************************************

# 0. nettoyer historique
rm(list = ls(all = TRUE)) # make sure previous work is clear
ls()

## character(0)

tic <- Sys.time()

# chargement des librairies R
library(plyr)
library(dplyr)

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:plyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

library(MASS)

##
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':
##
##     select

library(corrplot)
library(lattice)
library(caret)

## Loading required package: ggplot2

library(ROCR)
```

```
## Loading required package: gplots

##
## Attaching package: 'gplots'

## The following object is masked from 'package:stats':
##
##     lowess

library(ggplot2)
library(FactoMineR)
library(ade4)

## Warning: package 'ade4' was built under R version 3.4.1

##
## Attaching package: 'ade4'

## The following object is masked from 'package:FactoMineR':
##
##     reconst

library(colorspace)
library(e1071)
library(tidyr)

## Warning: package 'tidyr' was built under R version 3.4.1

library(rmarkdown)
library(randomForest)

## randomForest 4.6-12

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:ggplot2':
##
##     margin

## The following object is masked from 'package:dplyr':
##
##     combine

library(MLmetrics)

##
## Attaching package: 'MLmetrics'

## The following object is masked from 'package:caret':
##
##     RMSE

## The following object is masked from 'package:base':
##
##     Recall

library(ranger)

##
## Attaching package: 'ranger'

## The following object is masked from 'package:randomForest':
##
##     importance
```

```r
#=====================================
# PARTIE 1. PREPROCESSING DATA
#=====================================

# 1. importing and exploring data
path<-"/Users/lepicaut/Desktop/sos/MATMUT/0_0_matmut/fevrier_2017/00000_code"

# 2. set working directory - netttoyer Xi inutiles
setwd(path)
dv<- read.csv("devis_trsf_2015.csv", header=TRUE, sep=";")
dv1<-dv[,-2:-4]
dv2<-dv1[,-4:-10]
dv3<-dv2[,-5:-8]
dv4<-dv3[,-6:-39]
dv5<-dv4[,-7:-46]
dv6<-dv5[,-8:-12]
dv7<-dv6[,-9:-17]

dv16<-dv7
attach(dv16)
summary(dv16)

##    ID_CLIENT            NUM_CTC          NUM_DEV           C_AN_TTC
##  Min.   :211978166   Min.   : 1.000   Min.   :  1.000   906,30 :  1654
##  1st Qu.:212233606   1st Qu.: 1.000   1st Qu.:  1.000   442,30 :  1628
##  Median :212378405   Median : 1.000   Median :  1.000   821,30 :  1626
##  Mean   :212395235   Mean   : 1.576   Mean   :  2.123   811,30 :  1554
##  3rd Qu.:212568373   3rd Qu.: 2.000   3rd Qu.:  2.000   735,30 :  1393
##  Max.   :212760888   Max.   :18.000   Max.   :318.000   783,30 :  1393
##                                                         (Other):719426
##      COEFF          PUIS_FISC         AGE_CLI           NB_ENF
##  Min.   : 35.00   Min.   : 0.000   Min.   :  6.00   Min.   : 0.00
##  1st Qu.: 40.00   1st Qu.: 4.000   1st Qu.: 23.00   1st Qu.: 0.00
##  Median : 65.00   Median : 5.000   Median : 33.00   Median : 0.00
##  Mean   : 66.11   Mean   : 5.655   Mean   : 37.51   Mean   : 0.89
##  3rd Qu.: 95.00   3rd Qu.: 6.000   3rd Qu.: 50.00   3rd Qu.: 2.00
##  Max.   :288.00   Max.   :85.000   Max.   :168.00   Max.   :20.00
##  NA's   :4832     NA's   :4730     NA's   :29135    NA's   :99983
##  trsf_dev      ANC_VEH
##  N:547668   Min.   :-1.000
##  O:181006   1st Qu.: 5.000
##             Median : 9.000
##             Mean   : 9.403
##             3rd Qu.:14.000
##             Max.   :60.000
##

dim(dv16)

## [1] 728674     10

# [1] 728674 lignes x 10 colonnes

# 3. étude des valeurs manquantes
Pourcentage_NA = colMeans(is.na(dv16))
Pourcentage_NA = Pourcentage_NA * 100
Pourcentage_NA

##  ID_CLIENT    NUM_CTC    NUM_DEV   C_AN_TTC      COEFF  PUIS_FISC
##  0.0000000  0.0000000  0.0000000  0.0000000  0.6631223  0.6491243
##    AGE_CLI     NB_ENF   trsf_dev    ANC_VEH
##  3.9983587 13.7212251  0.0000000  0.0000000

# Analyse : taux de valeurs manquantes (%)
# COEFF      =  0.66%
# PUIS_FISC  = 0.64%
# AGE_CLI    = 3.99%
# NB_ENF     = 13.72%
# sans surprise les var Xi concernant les données personnelles de l'assuré,
# affichent les taux de Na les plus élevés (AGE CLIENT et NOMBRE D ENFANT)
```

```r
summary(Pourcentage_NA)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0000  0.0000  0.0000  1.9032  0.6596 13.7212

mean(Pourcentage_NA)

## [1] 1.903183

# = [1] 1.903183 % en moyenne pour l'ensemble du DATASET

# 4. FORMATAGE D UN DATASET NETTOYE > [1] 614048 lignes
#nettoyage des NA sur dv16
dv17=na.omit(dv16)

#summary(dv17)
dim(dv17)

## [1] 614048     10

# = [1] 614048 lignes   x  10 colonnes

# "sanity check" -- poids des valeurs manquantes nettoyés
PoidsNAnettoyees = dim(dv17)[[1]]/dim(dv16)[[1]]
PoidsNAnettoyees

## [1] 0.8426923

# [1] 0.8426923

# Nombre de lignes retirees
NbligneRetiree = dim(dv17)[[1]] - dim(dv16)[[1]]
NbligneRetiree

## [1] -114626

# [1] -114626 lignes supprimées avec NA

# sanity check des valeur manquantes sur dv17
Pourcentage_NA = colMeans(is.na(dv17))
Pourcentage_NA = Pourcentage_NA * 100
Pourcentage_NA

## ID_CLIENT   NUM_CTC   NUM_DEV  C_AN_TTC     COEFF PUIS_FISC   AGE_CLI
##         0         0         0         0         0         0         0
##    NB_ENF  trsf_dev   ANC_VEH
##         0         0         0

sum(is.na(dv17))

## [1] 0

# resultats [1] 0
# toutes les valeurs manquantes retirées dans dv17

# sanity check
dim(dv17)

## [1] 614048     10

str(dv17)
```

```
## 'data.frame':    614048 obs. of  10 variables:
##  $ ID_CLIENT: int  211978166 211978166 211978166 211978166 211978166 211978166 211978182 211978204 211
978204 211978226 ...
##  $ NUM_CTC  : int  3 3 3 3 3 6 2 2 3 2 ...
##  $ NUM_DEV  : int  1 2 3 4 5 1 1 1 1 1 ...
##  $ C_AN_TTC : Factor w/ 5312 levels "0,00","1 000,00",..: 1090 151 151 474 1090 864 4126 171 171 4778
...
##  $ COEFF    : int  100 100 100 100 100 100 40 100 100 95 ...
##  $ PUIS_FISC: int  5 5 5 5 5 5 4 4 4 1 ...
##  $ AGE_CLI  : int  25 25 25 25 25 25 47 24 24 23 ...
##  $ NB_ENF   : int  0 0 0 0 0 0 2 0 0 0 ...
##  $ trsf_dev : Factor w/ 2 levels "N","O": 1 1 1 1 1 2 2 2 2 1 ...
##  $ ANC_VEH  : int  9 9 9 9 9 9 4 6 6 6 ...
##  - attr(*, "na.action")=Class 'omit'  Named int [1:114626] 10 36 42 43 59 60 75 76 77 78 ...
##   .. ..- attr(*, "names")= chr [1:114626] "10" "36" "42" "43" ...
```

```r
summary(dv17)
```

```
##    ID_CLIENT            NUM_CTC         NUM_DEV           C_AN_TTC
##  Min.   :211978166   Min.   : 1.00   Min.   :  1.000   906,30 :  1466
##  1st Qu.:212234208   1st Qu.: 1.00   1st Qu.:  1.000   821,30 :  1419
##  Median :212379122   Median : 1.00   Median :  2.000   442,30 :  1394
##  Mean   :212395481   Mean   : 1.61   Mean   :  2.145   811,30 :  1370
##  3rd Qu.:212567554   3rd Qu.: 2.00   3rd Qu.:  2.000   735,30 :  1227
##  Max.   :212760888   Max.   :17.00   Max.   :318.000   783,30 :  1204
##                                                        (Other):605968
##      COEFF          PUIS_FISC         AGE_CLI           NB_ENF
##  Min.   : 35.00   Min.   : 0.000   Min.   : 13.00   Min.   : 0.0000
##  1st Qu.: 40.00   1st Qu.: 4.000   1st Qu.: 23.00   1st Qu.: 0.0000
##  Median : 65.00   Median : 5.000   Median : 32.00   Median : 0.0000
##  Mean   : 66.27   Mean   : 5.577   Mean   : 36.88   Mean   : 0.8838
##  3rd Qu.: 95.00   3rd Qu.: 6.000   3rd Qu.: 49.00   3rd Qu.: 2.0000
##  Max.   :256.00   Max.   :68.000   Max.   :168.00   Max.   :20.0000
##
##  trsf_dev      ANC_VEH
##  N:450163   Min.   :-1.000
##  O:163885   1st Qu.: 5.000
##             Median : 9.000
##             Mean   : 9.495
##             3rd Qu.:14.000
##             Max.   :60.000
##
```

```r
# POURCENTAGE : poids de l'etiquette NON / OUI
PoidsY=(table(dv17$trsf_dev)/nrow(dv17)*100)
PoidsY
```

```
##
##        N        O
## 73.31072 26.68928
```

```r
#   N        O
#   73.31072 26.68928
#   dataset avec classes déséquilibrées (les NON majoritaires)

# FULL DATASET = taille = 614048
dv18=dv17[sample(1:nrow(dv17),614048),]

# sanity check des valeur manquantes sur dv18
dim(dv18)
```

```
## [1] 614048     10
```

```r
(table(dv18$trsf_dev)/nrow(dv18)*100)
```

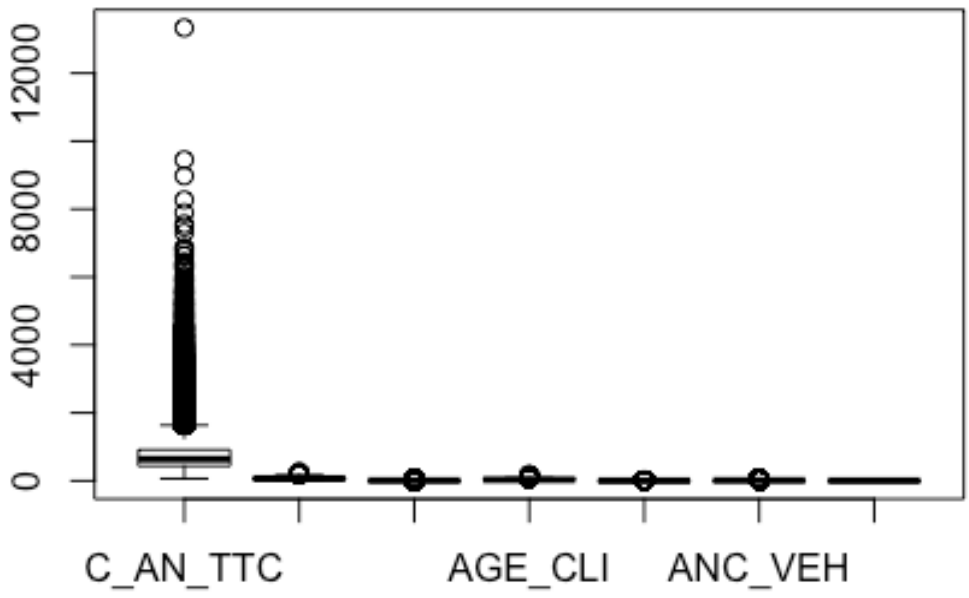
```
##
##        N        O
## 73.31072 26.68928
```

```r
summary(dv18)

##    ID_CLIENT            NUM_CTC         NUM_DEV           C_AN_TTC
##  Min.   :211978166   Min.   : 1.00   Min.   :  1.000   906,30 :  1466
##  1st Qu.:212234208   1st Qu.: 1.00   1st Qu.:  1.000   821,30 :  1419
##  Median :212379122   Median : 1.00   Median :  2.000   442,30 :  1394
##  Mean   :212395481   Mean   : 1.61   Mean   :  2.145   811,30 :  1370
##  3rd Qu.:212567554   3rd Qu.: 2.00   3rd Qu.:  2.000   735,30 :  1227
##  Max.   :212760888   Max.   :17.00   Max.   :318.000   783,30 :  1204
##                                                        (Other):605968


##      COEFF          PUIS_FISC         AGE_CLI           NB_ENF
##  Min.   : 35.00   Min.   : 0.000   Min.   : 13.00   Min.   : 0.0000
##  1st Qu.: 40.00   1st Qu.: 4.000   1st Qu.: 23.00   1st Qu.: 0.0000
##  Median : 65.00   Median : 5.000   Median : 32.00   Median : 0.0000
##  Mean   : 66.27   Mean   : 5.577   Mean   : 36.88   Mean   : 0.8838
##  3rd Qu.: 95.00   3rd Qu.: 6.000   3rd Qu.: 49.00   3rd Qu.: 2.0000
##  Max.   :256.00   Max.   :68.000   Max.   :168.00   Max.   :20.0000
##
##  trsf_dev      ANC_VEH
##  N:450163   Min.   :-1.000
##  O:163885   1st Qu.: 5.000
##             Median : 9.000
##             Mean   : 9.495
##             3rd Qu.:14.000
##             Max.   :60.000
##

str(dv18)

## 'data.frame':    614048 obs. of  10 variables:
##  $ ID_CLIENT: int  212669223 212111414 212121127 212527388 212153429 212163897 212327385 212413341 212
659463 212370976 ...
##  $ NUM_CTC  : int  2 1 2 3 1 1 2 1 1 2 ...
##  $ NUM_DEV  : int  1 1 1 8 2 2 2 1 1 4 ...
##  $ C_AN_TTC : Factor w/ 5312 levels "0,00","1 000,00",..: 4100 3601 97 3591 4330 535 3677 505 4925 357
0 ...
##  $ COEFF    : int  40 35 65 57 40 65 45 100 40 80 ...
##  $ PUIS_FISC: int  5 4 4 4 6 5 5 4 12 7 ...
##  $ AGE_CLI  : int  40 31 44 35 50 27 44 20 70 23 ...
##  $ NB_ENF   : int  0 2 2 1 1 0 3 0 0 0 ...
##  $ trsf_dev : Factor w/ 2 levels "N","O": 1 2 1 1 1 2 1 1 1 2 ...
##  $ ANC_VEH  : int  1 8 7 12 2 6 17 6 6 18 ...
##  - attr(*, "na.action")=Class 'omit'  Named int [1:114626] 10 36 42 43 59 60 75 76 77 78 ...
##   .. ..- attr(*, "names")= chr [1:114626] "10" "36" "42" "43" ...

# transfrome C_AN_TTC en var numerique
dv18$C_AN_TTC = gsub(",",".",dv18$C_AN_TTC)
dv18$C_AN_TTC = gsub(" ","",dv18$C_AN_TTC)
dv18$C_AN_TTC = as.double(dv18$C_AN_TTC)

# sanity check des valeur manquantes sur dv18$C_AN_TTC
dv18$C_AN_TTC[1:50]

##  [1]  478.3  341.3 1048.3  336.3  551.3 1278.3  379.3 1261.3  814.3  326.3
## [11]  379.3 1292.3  862.3  327.3 2145.3  741.0 2205.3 1017.0  483.3 1639.3
## [21]  676.3  489.3 1147.3 1970.3  773.3  683.3  802.3 1102.3 1164.3  240.3
## [31]  393.3  187.0 1532.3  657.3  413.3  302.3  290.0 1076.3 1006.3  865.3
## [41]  418.3  784.0  444.3  979.3 1269.3  435.3  287.3  390.3  281.0  909.3

# 4. IDENTIFIER LES QUANTI
library(plyr)
type_var_num = colwise(is.numeric)
var_num=names(dv18)[which(type_var_num(dv18)==TRUE)]
```

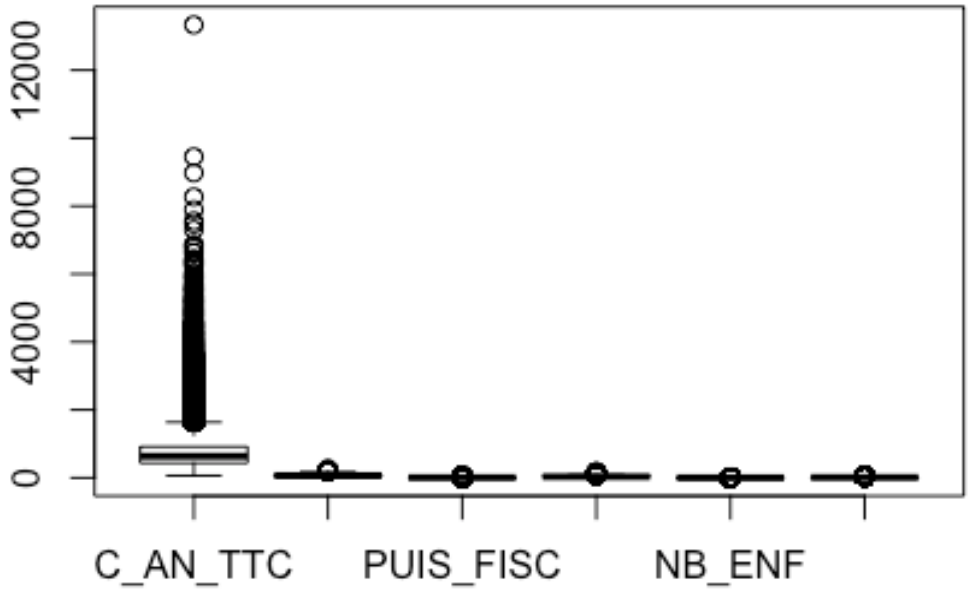
```r
# sanity check des valeurs manquantes sur dv18Quanti
dv18Quanti = dv18[, c(var_num)]
str(dv18Quanti)

## 'data.frame':    614048 obs. of  9 variables:
##  $ ID_CLIENT: int  212669223 212111414 212121127 212527388 212153429 212163897 212327385 212413341 212
659463 212370976 ...
##  $ NUM_CTC  : int  2 1 2 3 1 1 2 1 1 2 ...
##  $ NUM_DEV  : int  1 1 1 8 2 2 2 1 1 4 ...
##  $ C_AN_TTC : num  478 341 1048 336 551 ...
##  $ COEFF    : int  40 35 65 57 40 65 45 100 40 80 ...
##  $ PUIS_FISC: int  5 4 4 4 6 5 5 4 12 7 ...
##  $ AGE_CLI  : int  40 31 44 35 50 27 44 20 70 23 ...
##  $ NB_ENF   : int  0 2 2 1 1 0 3 0 0 0 ...
##  $ ANC_VEH  : int  1 8 7 12 2 6 17 6 6 18 ...
```

```r
summary(dv18Quanti)

##    ID_CLIENT            NUM_CTC         NUM_DEV           C_AN_TTC
##  Min.   :211978166   Min.   : 1.00   Min.   :  1.000   Min.   :   65.0
##  1st Qu.:212234208   1st Qu.: 1.00   1st Qu.:  1.000   1st Qu.:  434.3
##  Median :212379122   Median : 1.00   Median :  2.000   Median :  635.3
##  Mean   :212395481   Mean   : 1.61   Mean   :  2.145   Mean   :  729.5
##  3rd Qu.:212567554   3rd Qu.: 2.00   3rd Qu.:  2.000   3rd Qu.:  915.3
##  Max.   :212760888   Max.   :17.00   Max.   :318.000   Max.   :13335.3
##      COEFF          PUIS_FISC         AGE_CLI           NB_ENF
##  Min.   : 35.00   Min.   : 0.000   Min.   : 13.00   Min.   : 0.0000
##  1st Qu.: 40.00   1st Qu.: 4.000   1st Qu.: 23.00   1st Qu.: 0.0000
##  Median : 65.00   Median : 5.000   Median : 32.00   Median : 0.0000
##  Mean   : 66.27   Mean   : 5.577   Mean   : 36.88   Mean   : 0.8838
##  3rd Qu.: 95.00   3rd Qu.: 6.000   3rd Qu.: 49.00   3rd Qu.: 2.0000
##  Max.   :256.00   Max.   :68.000   Max.   :168.00   Max.   :20.0000
##     ANC_VEH
##  Min.   :-1.000
##  1st Qu.: 5.000
##  Median : 9.000
##  Mean   : 9.495
##  3rd Qu.:14.000
##  Max.   :60.000

# 5. variable etiquette
etiq<-dv18[,-4:-8]
etiq<-etiq[,-5]
#str(etiq)

# 6. pour avoir 1 client = 1 devis = 1 etiquette & unicité ligne
# specifications : pour l'apprentissage, on prend tous les devis
# (du premier au final qui dit OUI ou NON - NUM_DEV = numero compteur devis)

dq=merge(dv18Quanti,etiq,by=c("ID_CLIENT","NUM_CTC","NUM_DEV"))

# sanity check des valeur manquantes sur dq
str(dq)

## 'data.frame':    614928 obs. of  10 variables:
##  $ ID_CLIENT: int  211978166 211978166 211978166 211978166 211978166 211978166 211978182 211978204 211
978204 211978226 ...
##  $ NUM_CTC  : int  3 3 3 3 3 6 2 2 3 2 ...
##  $ NUM_DEV  : int  1 2 3 4 5 1 1 1 1 1 ...
##  $ C_AN_TTC : num  1630 1076 1076 1245 1630 ...
##  $ COEFF    : int  100 100 100 100 100 100 40 100 100 95 ...
##  $ PUIS_FISC: int  5 5 5 5 5 5 4 4 4 1 ...
##  $ AGE_CLI  : int  25 25 25 25 25 25 47 24 24 23 ...
##  $ NB_ENF   : int  0 0 0 0 0 0 2 0 0 0 ...
##  $ ANC_VEH  : int  9 9 9 9 9 9 4 6 6 6 ...
##  $ trsf_dev : Factor w/ 2 levels "N","O": 1 1 1 1 1 2 2 2 2 1 ...

summary(dq)##    ID_CLIENT            NUM_CTC          NUM_DEV           C_AN_TTC
##  Min.   :211978166   Min.   : 1.000   Min.   :  1.000   Min.   :   65.0
##  1st Qu.:212234268   1st Qu.: 1.000   1st Qu.:  1.000   1st Qu.:  434.3
##  Median :212379487   Median : 1.000   Median :  2.000   Median :  635.3
##  Mean   :212395540   Mean   : 1.611   Mean   :  2.145   Mean   :  729.4
##  3rd Qu.:212567575   3rd Qu.: 2.000   3rd Qu.:  2.000   3rd Qu.:  915.3
##  Max.   :212760888   Max.   :17.000   Max.   :318.000   Max.   :13335.3
##      COEFF          PUIS_FISC         AGE_CLI           NB_ENF
##  Min.   : 35.00   Min.   : 0.000   Min.   : 13.00   Min.   : 0.0000
##  1st Qu.: 40.00   1st Qu.: 4.000   1st Qu.: 23.00   1st Qu.: 0.0000
##  Median : 65.00   Median : 5.000   Median : 32.00   Median : 0.0000
##  Mean   : 66.26   Mean   : 5.578   Mean   : 36.88   Mean   : 0.8842
##  3rd Qu.: 95.00   3rd Qu.: 6.000   3rd Qu.: 49.00   3rd Qu.: 2.0000
##  Max.   :256.00   Max.   :68.000   Max.   :168.00   Max.   :20.0000
##     ANC_VEH       trsf_dev
##  Min.   :-1.000   N:450751
##  1st Qu.: 5.000   O:164177
##  Median : 9.000
##  Mean   : 9.495
##  3rd Qu.:14.000
##  Max.   :60.000
```

```r
# 7. formatage du dataset final
dq1=dq[,-1:-3]
dq2=dq1
str(dq2)

## 'data.frame':    614928 obs. of  7 variables:
##  $ C_AN_TTC : num  1630 1076 1076 1245 1630 ...
##  $ COEFF    : int  100 100 100 100 100 100 40 100 100 95 ...
##  $ PUIS_FISC: int  5 5 5 5 5 5 4 4 4 1 ...
##  $ AGE_CLI  : int  25 25 25 25 25 25 47 24 24 23 ...
##  $ NB_ENF   : int  0 0 0 0 0 0 2 0 0 0 ...
##  $ ANC_VEH  : int  9 9 9 9 9 9 4 6 6 6 ...
##  $ trsf_dev : Factor w/ 2 levels "N","O": 1 1 1 1 1 2 2 2 2 1 ...

summary(dq2)

##     C_AN_TTC           COEFF          PUIS_FISC         AGE_CLI
##  Min.   :   65.0   Min.   : 35.00   Min.   : 0.000   Min.   : 13.00
##  1st Qu.:  434.3   1st Qu.: 40.00   1st Qu.: 4.000   1st Qu.: 23.00
##  Median :  635.3   Median : 65.00   Median : 5.000   Median : 32.00
##  Mean   :  729.4   Mean   : 66.26   Mean   : 5.578   Mean   : 36.88
##  3rd Qu.:  915.3   3rd Qu.: 95.00   3rd Qu.: 6.000   3rd Qu.: 49.00
##  Max.   :13335.3   Max.   :256.00   Max.   :68.000   Max.   :168.00
##      NB_ENF           ANC_VEH       trsf_dev
##  Min.   : 0.0000   Min.   :-1.000   N:450751
##  1st Qu.: 0.0000   1st Qu.: 5.000   O:164177
##  Median : 0.0000   Median : 9.000
##  Mean   : 0.8842   Mean   : 9.495
##  3rd Qu.: 2.0000   3rd Qu.:14.000
##  Max.   :20.0000   Max.   :60.000

# boxplot sur données non normalisées et non centrées-réduites

par(mfrow=c(1,1))
boxplot(dq2)       # avec l'etiquette
```



```r
boxplot(dq2[,1:6]) # sans l'etiquette
```




```r
# avec les variables explicatives Xi
# on ne peut rien dire, car C_AN_TTC n'rest pas sur la même échelle de grandeur des 5 autres var explivat
ives

# AUDIT DE LA COHERENCE DES DONNEES
# DETECTION  DES ANOMALIES dans les données

# 1. C_AN_TTC

# Min.   :   65.0
# Max.   :13335.3 ?

# affichage par ordre croissant des valeurs C_AN_TTC
#sort(dq2$C_AN_TTC)

# affichage par ordre decroissant des valeurs C_AN_TTC
#rev(sort(dq2$C_AN_TTC))

# pas de recodage,  C_AN_TTC MIN et MAX, correspondent a une situation de gestion

# sanity check
summary(dq2)

##     C_AN_TTC           COEFF          PUIS_FISC         AGE_CLI
##  Min.   :   65.0   Min.   : 35.00   Min.   : 0.000   Min.   : 13.00
##  1st Qu.:  434.3   1st Qu.: 40.00   1st Qu.: 4.000   1st Qu.: 23.00
##  Median :  635.3   Median : 65.00   Median : 5.000   Median : 32.00
##  Mean   :  729.4   Mean   : 66.26   Mean   : 5.578   Mean   : 36.88
##  3rd Qu.:  915.3   3rd Qu.: 95.00   3rd Qu.: 6.000   3rd Qu.: 49.00
##  Max.   :13335.3   Max.   :256.00   Max.   :68.000   Max.   :168.00
##      NB_ENF           ANC_VEH       trsf_dev
##  Min.   : 0.0000   Min.   :-1.000   N:450751
##  1st Qu.: 0.0000   1st Qu.: 5.000   O:164177
##  Median : 0.0000   Median : 9.000
##  Mean   : 0.8842   Mean   : 9.495
##  3rd Qu.: 2.0000   3rd Qu.:14.000
##  Max.   :20.0000   Max.   :60.000
```

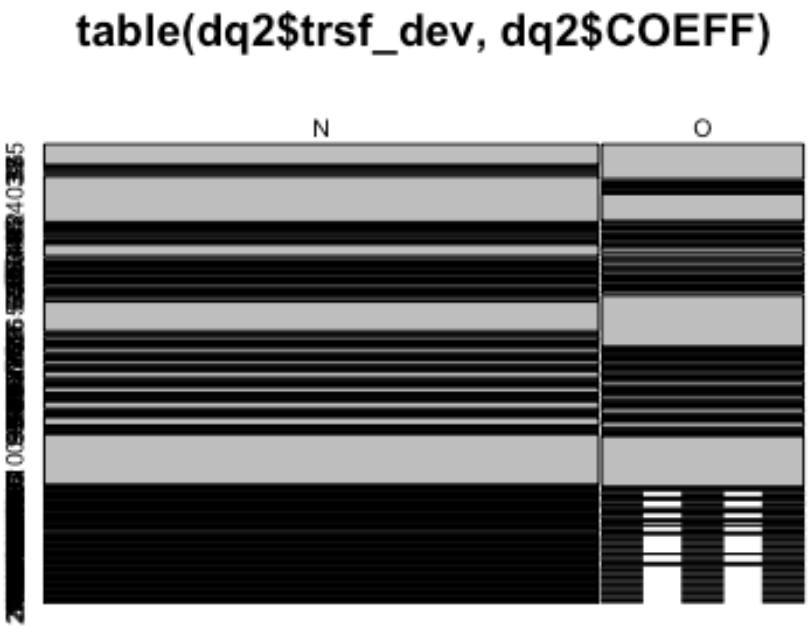
```r
# recodage C_AN_TTC vérifié

# 2. COEFF

# Min.   : 35.00
# Max.   :256.00 ?

# affichage par ordre croissant des valeurs COEFF
#sort(dq2$COEFF)

# affichage par ordre decroissant des valeurs COEFF
#rev(sort(dq2$COEFF))

# recodage valeur avec majorant = 225
# apres verification aupres des gestionnaires
dq2$COEFF[dq2$COEFF>=225]=225;
#rev(sort(dq2$COEFF))

# recodage, BM 256 en BM 225, correspondance a situation de gestion

# sanity check
summary(dq2)

##     C_AN_TTC           COEFF          PUIS_FISC         AGE_CLI
##  Min.   :   65.0   Min.   : 35.00   Min.   : 0.000   Min.   : 13.00
##  1st Qu.:  434.3   1st Qu.: 40.00   1st Qu.: 4.000   1st Qu.: 23.00
##  Median :  635.3   Median : 65.00   Median : 5.000   Median : 32.00
##  Mean   :  729.4   Mean   : 66.26   Mean   : 5.578   Mean   : 36.88
##  3rd Qu.:  915.3   3rd Qu.: 95.00   3rd Qu.: 6.000   3rd Qu.: 49.00
##  Max.   :13335.3   Max.   :225.00   Max.   :68.000   Max.   :168.00
##      NB_ENF           ANC_VEH       trsf_dev
##  Min.   : 0.0000   Min.   :-1.000   N:450751
##  1st Qu.: 0.0000   1st Qu.: 5.000   O:164177
##  Median : 0.0000   Median : 9.000
##  Mean   : 0.8842   Mean   : 9.495
##  3rd Qu.: 2.0000   3rd Qu.:14.000
##  Max.   :20.0000   Max.   :60.000

# recodage COEFF  vérifié

# 3. PUIS_FISC

# Min.   : 0.000
# Max.   :68.000 ?

# affichage par ordre croissant des valeurs PUIS_FISC
#sort(dq2$PUIS_FISC)
#list(dq2$PUIS_FISC[dq2$PUIS_FISC<3])
# recodage valeur avec minorant fixé a 3 (sinon velo electrique, perimetre etude VL 4 roues)
dq2$PUIS_FISC[dq2$PUIS_FISC<3]=3
#sort(dq2$PUIS_FISC)

# affichage par ordre decroissant des valeurs PUIS_FISC
#rev(sort(dq2$PUIS_FISC))
# pas de recodage, PUIS_FISC 68 CV, correspond a une supercar
# sanity check


summary(dq2)##     C_AN_TTC           COEFF          PUIS_FISC        AGE_CLI
##  Min.   :   65.0   Min.   : 35.00   Min.   : 3.00   Min.   : 13.00
##  1st Qu.:  434.3   1st Qu.: 40.00   1st Qu.: 4.00   1st Qu.: 23.00
##  Median :  635.3   Median : 65.00   Median : 5.00   Median : 32.00
##  Mean   :  729.4   Mean   : 66.26   Mean   : 5.66   Mean   : 36.88
##  3rd Qu.:  915.3   3rd Qu.: 95.00   3rd Qu.: 6.00   3rd Qu.: 49.00
##  Max.   :13335.3   Max.   :225.00   Max.   :68.00   Max.   :168.00
##      NB_ENF           ANC_VEH       trsf_dev
##  Min.   : 0.0000   Min.   :-1.000   N:450751
##  1st Qu.: 0.0000   1st Qu.: 5.000   O:164177
##  Median : 0.0000   Median : 9.000
##  Mean   : 0.8842   Mean   : 9.495
##  3rd Qu.: 2.0000   3rd Qu.:14.000
##  Max.   :20.0000   Max.   :60.000
```

```r
# recodage PUIS_FISC  vérifié

# 4. AGE_CLI

# Min.   : 13.00
# Max.   :168.00 ?

# affichage par ordre croissant des valeurs AGE_CLI
#sort(dq2$AGE_CLI)
#list(dq2$AGE_CLI[dq2$AGE_CLI<15])
# recodage valeur avec minorant fixé a 15 (conduite accompagnee)
# apres verification aupres des gestionnaires
dq2$AGE_CLI[dq2$AGE_CLI<15]=16
#sort(dq2$AGE_CLI)

# affichage par ordre decroissant des valeurs AGE_CLI
#rev(sort(dq2$AGE_CLI))

# recodage valeur avec majorant = 100
# apres verification aupres des gestionnaires
dq2$AGE_CLI[dq2$AGE_CLI>=95]=95;
#rev(sort(dq2$AGE_CLI))

# sanity check
summary(dq2)

##     C_AN_TTC           COEFF          PUIS_FISC        AGE_CLI
##  Min.   :   65.0   Min.   : 35.00   Min.   : 3.00   Min.   :16.00
##  1st Qu.:  434.3   1st Qu.: 40.00   1st Qu.: 4.00   1st Qu.:23.00
##  Median :  635.3   Median : 65.00   Median : 5.00   Median :32.00
##  Mean   :  729.4   Mean   : 66.26   Mean   : 5.66   Mean   :36.88
##  3rd Qu.:  915.3   3rd Qu.: 95.00   3rd Qu.: 6.00   3rd Qu.:49.00
##  Max.   :13335.3   Max.   :225.00   Max.   :68.00   Max.   :95.00
##      NB_ENF           ANC_VEH       trsf_dev
##  Min.   : 0.0000   Min.   :-1.000   N:450751
##  1st Qu.: 0.0000   1st Qu.: 5.000   O:164177
##  Median : 0.0000   Median : 9.000
##  Mean   : 0.8842   Mean   : 9.495
##  3rd Qu.: 2.0000   3rd Qu.:14.000
##  Max.   :20.0000   Max.   :60.000

# recodage AGE_CLI vérifié

# 5. NB_ENF

# Min.   : 0.0000
# Max.   :20.0000 ?

# affichage par ordre croissant des valeurs NB_ENF
#sort(dq2$NB_ENF)

# affichage par ordre decroissant des valeurs NB_ENF
#rev(sort(dq2$NB_ENF))

# recodage valeur avec majorant fixé a 10 (>10, valeur aberrante)
dq2$NB_ENF[dq2$NB_ENF>=10]=10;
#rev(sort(dq2$NB_ENF))

# sanity check
summary(dq2)
```

```
##     C_AN_TTC           COEFF          PUIS_FISC        AGE_CLI
##  Min.   :   65.0   Min.   : 35.00   Min.   : 3.00   Min.   :16.00
##  1st Qu.:  434.3   1st Qu.: 40.00   1st Qu.: 4.00   1st Qu.:23.00
##  Median :  635.3   Median : 65.00   Median : 5.00   Median :32.00
##  Mean   :  729.4   Mean   : 66.26   Mean   : 5.66   Mean   :36.88
##  3rd Qu.:  915.3   3rd Qu.: 95.00   3rd Qu.: 6.00   3rd Qu.:49.00
##  Max.   :13335.3   Max.   :225.00   Max.   :68.00   Max.   :95.00
##      NB_ENF           ANC_VEH       trsf_dev
##  Min.   : 0.0000   Min.   :-1.000   N:450751
##  1st Qu.: 0.0000   1st Qu.: 5.000   O:164177
##  Median : 0.0000   Median : 9.000
##  Mean   : 0.8839   Mean   : 9.495
##  3rd Qu.: 2.0000   3rd Qu.:14.000
##  Max.   :10.0000   Max.   :60.000

# recodage NB_ENF vérifié

# 6. ANC_VEH

# Min.   :-1.000
# Max.   :60.000  ?

# affichage par ordre croissant des valeurs ANC_VEH
#sort(dq2$ANC_VEH)

# recodage logique des valeur "-1" en "O" pour changement de variable
dq2$ANC_VEH[dq2$ANC_VEH==-1]=0
#sort(dq2$ANC_VEH)

# affichage par ordre decroissant des valeurs ANC_VEH
#rev(sort(dq2$ANC_VEH))
# pas de recodage,ANC_VEH 60, correspond VL DE COLLECTION

# sanity check
summary(dq2)

##     C_AN_TTC           COEFF          PUIS_FISC        AGE_CLI
##  Min.   :   65.0   Min.   : 35.00   Min.   : 3.00   Min.   :16.00
##  1st Qu.:  434.3   1st Qu.: 40.00   1st Qu.: 4.00   1st Qu.:23.00
##  Median :  635.3   Median : 65.00   Median : 5.00   Median :32.00
##  Mean   :  729.4   Mean   : 66.26   Mean   : 5.66   Mean   :36.88
##  3rd Qu.:  915.3   3rd Qu.: 95.00   3rd Qu.: 6.00   3rd Qu.:49.00
##  Max.   :13335.3   Max.   :225.00   Max.   :68.00   Max.   :95.00
##      NB_ENF           ANC_VEH       trsf_dev
##  Min.   : 0.0000   Min.   : 0.000   N:450751
##  1st Qu.: 0.0000   1st Qu.: 5.000   O:164177
##  Median : 0.0000   Median : 9.000
##  Mean   : 0.8839   Mean   : 9.495
##  3rd Qu.: 2.0000   3rd Qu.:14.000
##  Max.   :10.0000   Max.   :60.000

dim(dq2)

## [1] 614928      7

#=====================================
# PARTIE 2. ANALYSE DESCRIPTIVE
#=====================================

# PARTIE 2.1 ANALYSE DESCRIPTIVE :
# 2.1.1 POPULATION TOTALE "les OUI" et "les NON"

# Y : variable qualitative

table(dq2$trsf_dev)

##
##      N      O
## 450751 164177
```

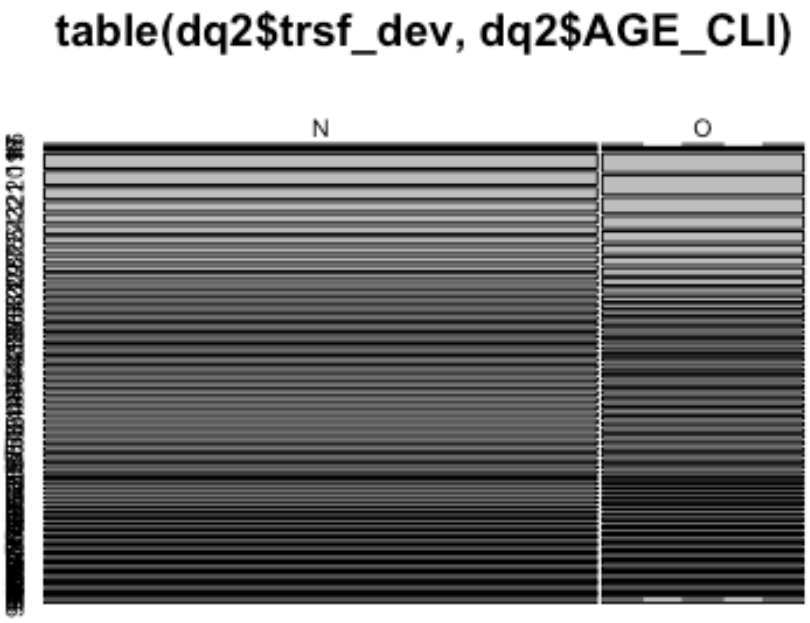
```r
freqabs=round(table(dq2$trsf_dev)/sum(table(dq2$trsf_dev)),2)
freqabs

##
##    N    O
## 0.73 0.27

# en pourcentage
freqabs=round(table(dq2$trsf_dev)/sum(table(dq2$trsf_dev))*100,4)
freqabs

##
##       N       O
## 73.3014 26.6986

# avec table des fréquences : croisement des modalités de Y avec les 6 explicatives Xi
par(mfrow=c(1,1))
plot(table(dq2$trsf_dev,dq2$C_AN_TTC))
```

```r
plot(table(dq2$trsf_dev,dq2$COEFF))
```

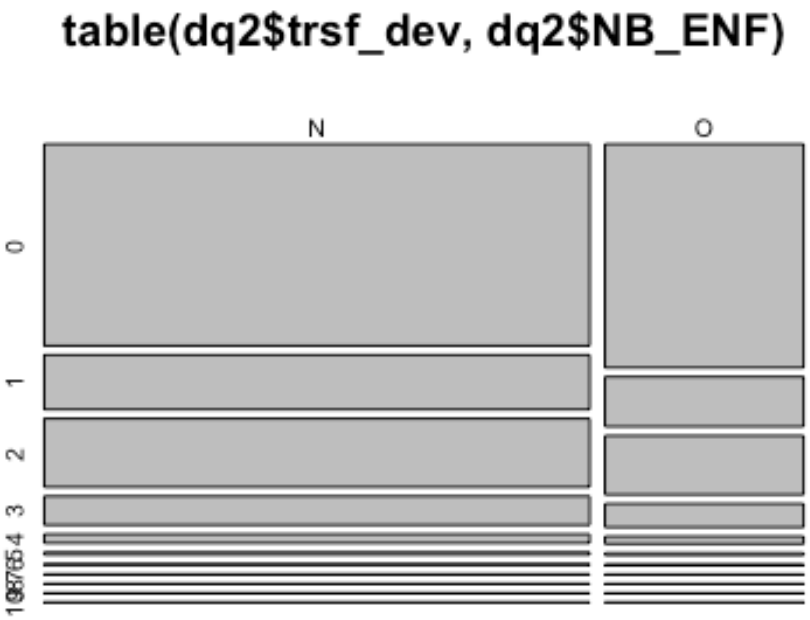



```r
plot(table(dq2$trsf_dev,dq2$PUIS_FISC))
```

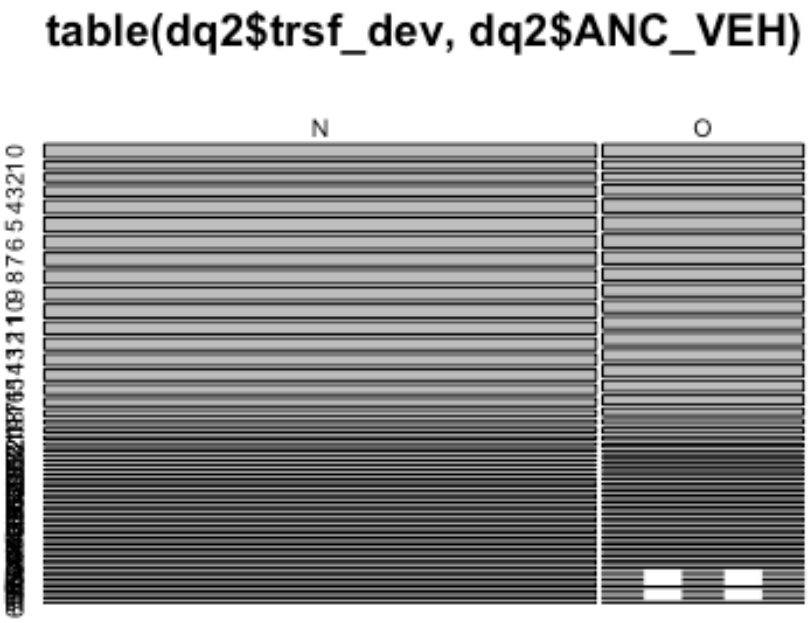
```r
plot(table(dq2$trsf_dev,dq2$AGE_CLI))
```




```r
plot(table(dq2$trsf_dev,dq2$NB_ENF))
```

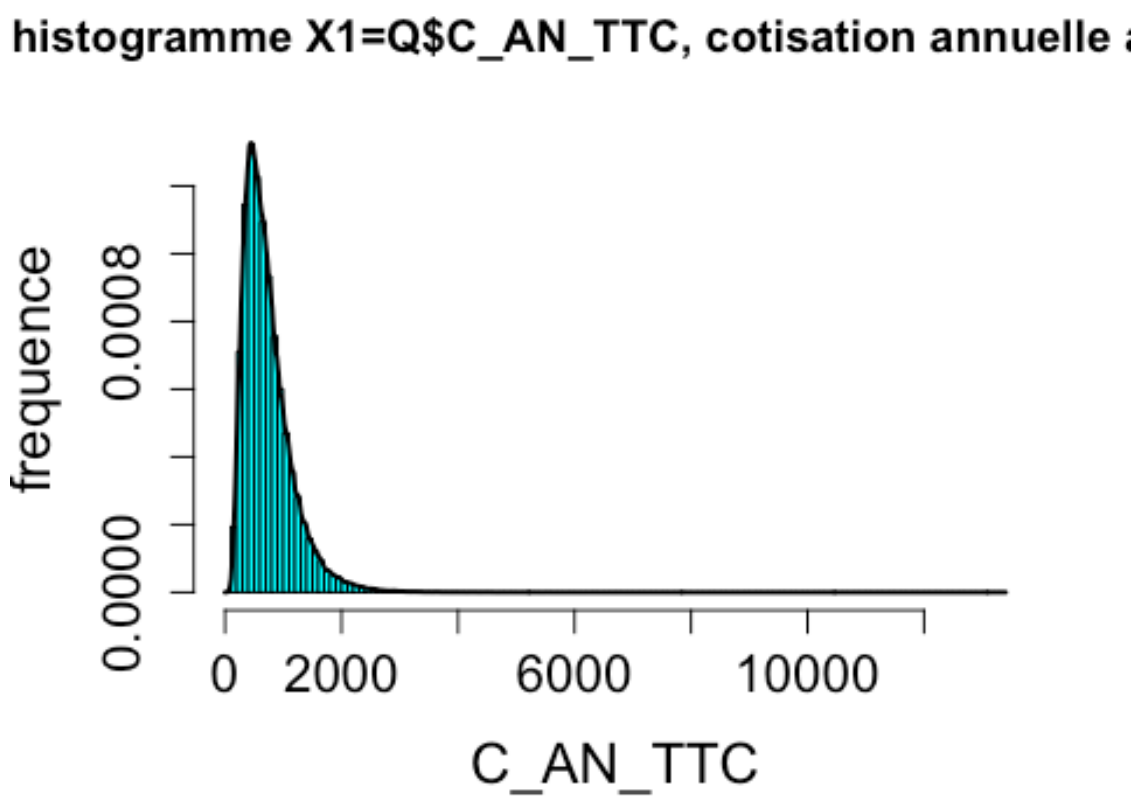


```r
plot(table(dq2$trsf_dev,dq2$ANC_VEH))
```

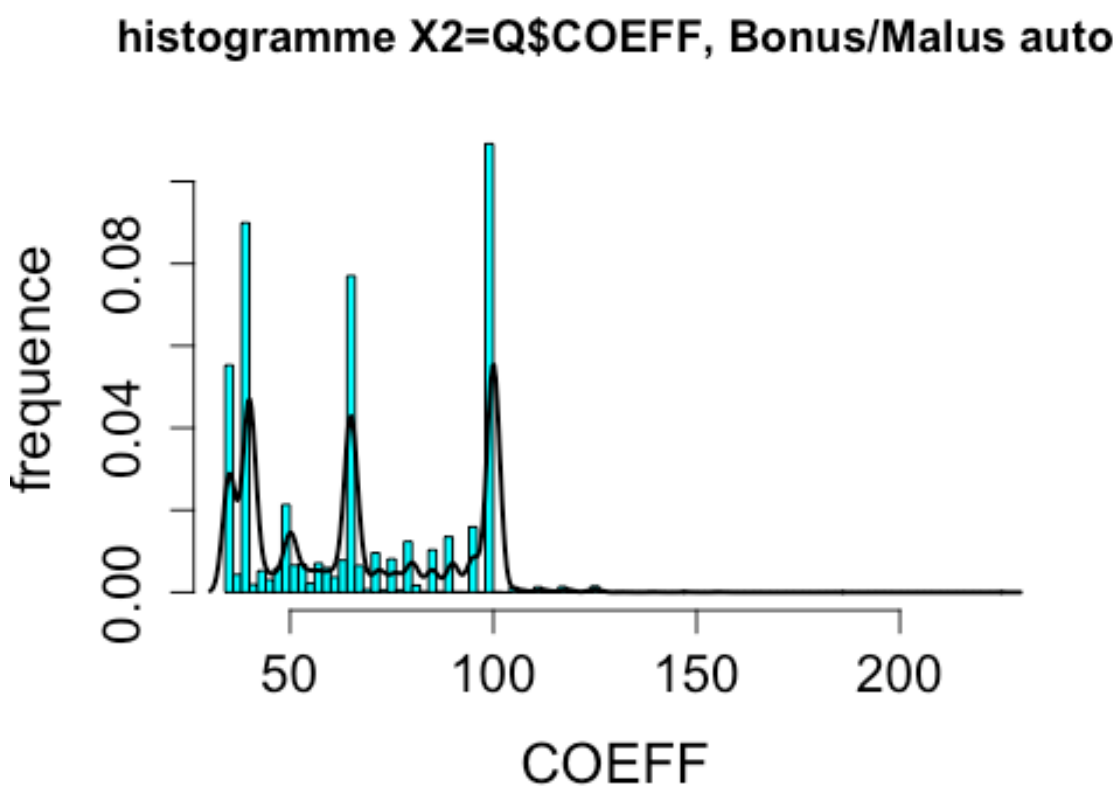



```r
# analyse :
# . la variable d'intérêt transformation devis est en relation avec tous les Xi quantitatives explicative
s.
# . la var Xi C_AN_TTC a une relation plus important avec trsf_dev, en comparaison avec les autres Xi

# Xi : pour les var. quantitatives

Q= dq2[,1:6]
# recodage des Xi
X1=Q$C_AN_TTC
X2=Q$COEFF
X3=Q$PUIS_FISC
X4=Q$AGE_CLI
X5=Q$NB_ENF
X6=Q$ANC_VEH

# 1. Tendances centrales

summary(Q)

##     C_AN_TTC           COEFF          PUIS_FISC        AGE_CLI
##  Min.   :   65.0   Min.   : 35.00   Min.   : 3.00   Min.   :16.00
##  1st Qu.:  434.3   1st Qu.: 40.00   1st Qu.: 4.00   1st Qu.:23.00
##  Median :  635.3   Median : 65.00   Median : 5.00   Median :32.00
##  Mean   :  729.4   Mean   : 66.26   Mean   : 5.66   Mean   :36.88
##  3rd Qu.:  915.3   3rd Qu.: 95.00   3rd Qu.: 6.00   3rd Qu.:49.00
##  Max.   :13335.3   Max.   :225.00   Max.   :68.00   Max.   :95.00
##      NB_ENF           ANC_VEH
##  Min.   : 0.0000   Min.   : 0.000
##  1st Qu.: 0.0000   1st Qu.: 5.000
##  Median : 0.0000   Median : 9.000
##  Mean   : 0.8839   Mean   : 9.495
##  3rd Qu.: 2.0000   3rd Qu.:14.000
##  Max.   :10.0000   Max.   :60.000

# 6 Xi quanti dont C_AN_TTC ne se trouve pas sur la même échelle de grandeur que les autres Xi
```

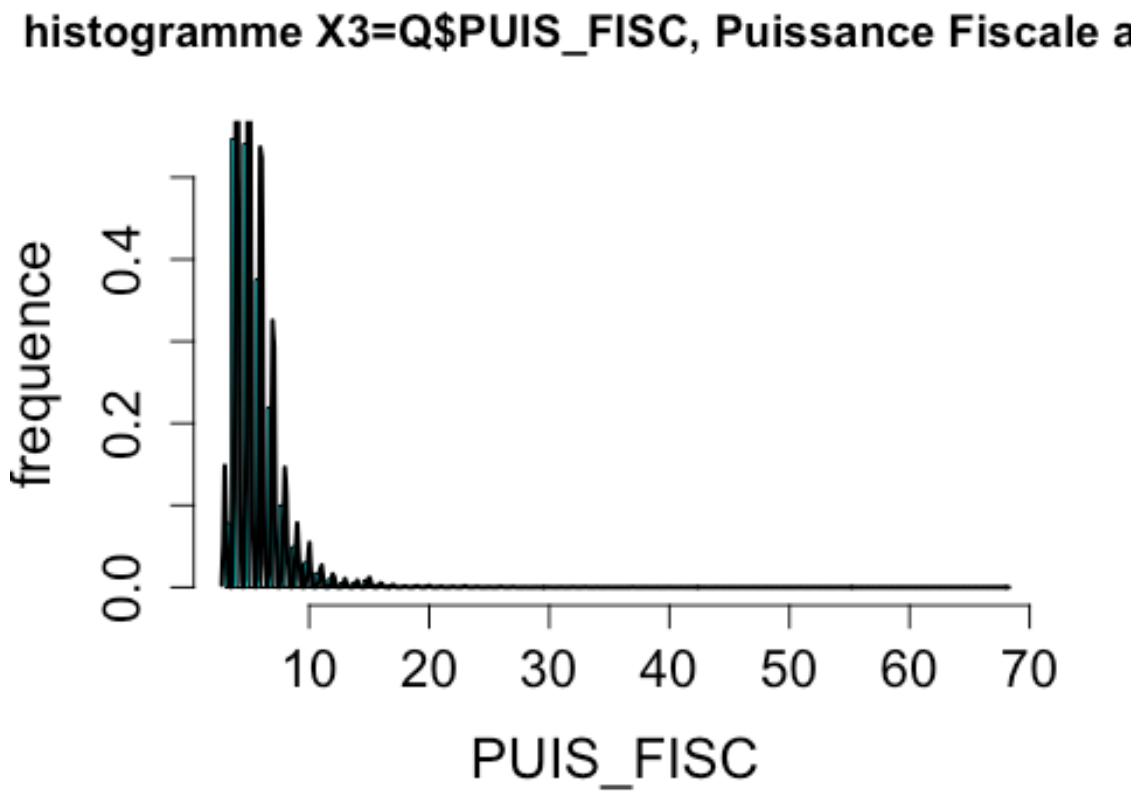
```r
# 2. Indicateurs de dispersion/repartition

# matrice variance-covariance
var(Q)

##              C_AN_TTC      COEFF   PUIS_FISC      AGE_CLI       NB_ENF
## C_AN_TTC  179468.7379 6109.84306 205.6420799 -2451.297652 -121.3654449
## COEFF       6109.8431  613.70555 -11.0109672  -235.350965  -12.0094598
## PUIS_FISC    205.6421  -11.01097   5.2168244     6.913562    0.4504491
## AGE_CLI    -2451.2977 -235.35097   6.9135624   246.338006    9.9166648
## NB_ENF      -121.3654  -12.00946   0.4504491     9.916665    1.4893831
## ANC_VEH     -466.3874   23.82561   0.5895995   -11.069806   -0.2369642
##                ANC_VEH
## C_AN_TTC  -466.3874120
## COEFF       23.8256108
## PUIS_FISC    0.5895995
## AGE_CLI    -11.0698060
## NB_ENF      -0.2369642
## ANC_VEH     38.3683223

# la var C_AN_TTC a une tres grosse variance par rapport aux autres Xi

# kurtosis / skewness

# EXPLICATIONS

# Sknewness
# Une valeur nulle du Sknewness indique une distribution symetrique des donnees
# autour de leur moyenne alors qu un Skewness positif (respectivement negatif)
# indique une queue distribution plus etalee vers la droite (resp. vers la gauche).

# Kurtosis
# Plus le Kurtosis est eleve, plus il y a de valeurs eloignees de la moyenne,
# les queues de distributions sont plus epaisses
# (que celles de la gaussienne, dont le kurtosis est 3).

library(moments)

##
## Attaching package: 'moments'

## The following objects are masked from 'package:e1071':
##
##     kurtosis, moment, skewness

# COTISATION
Titre="histogramme X1=Q$C_AN_TTC, cotisation annuelle auto"
hist(X1,breaks=100,col="cyan",prob=TRUE,cex.axis=1.4,cex.lab=1.5,ylab="frequence", xlab="C_AN_TTC",main=T
itre)
lines(density(X1), lwd = 2)
```



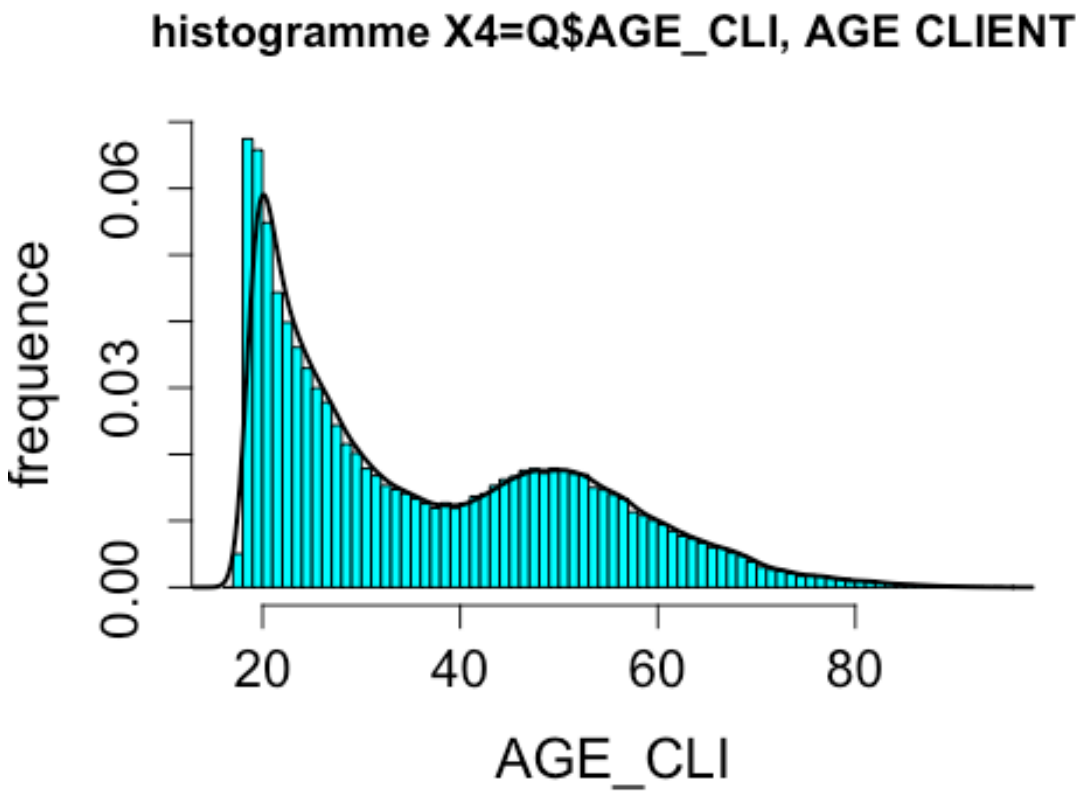

```
skewness(Q$C_AN_TTC, na.rm = T) # [1] 1.972588

## [1] 1.972588

kurtosis(Q$C_AN_TTC,na.rm=T) # [1] 12.76343

## [1] 12.76343

# distribution continue étalée à droite

# COEFF BONUS MALUS
Titre="histogramme X2=Q$COEFF, Bonus/Malus auto"
hist(X2,breaks=100,col="cyan",prob=TRUE,cex.axis=1.4,cex.lab=1.5,ylab="frequence", xlab="COEFF",main=Titr
e)
lines(density(X2), lwd = 2)
```



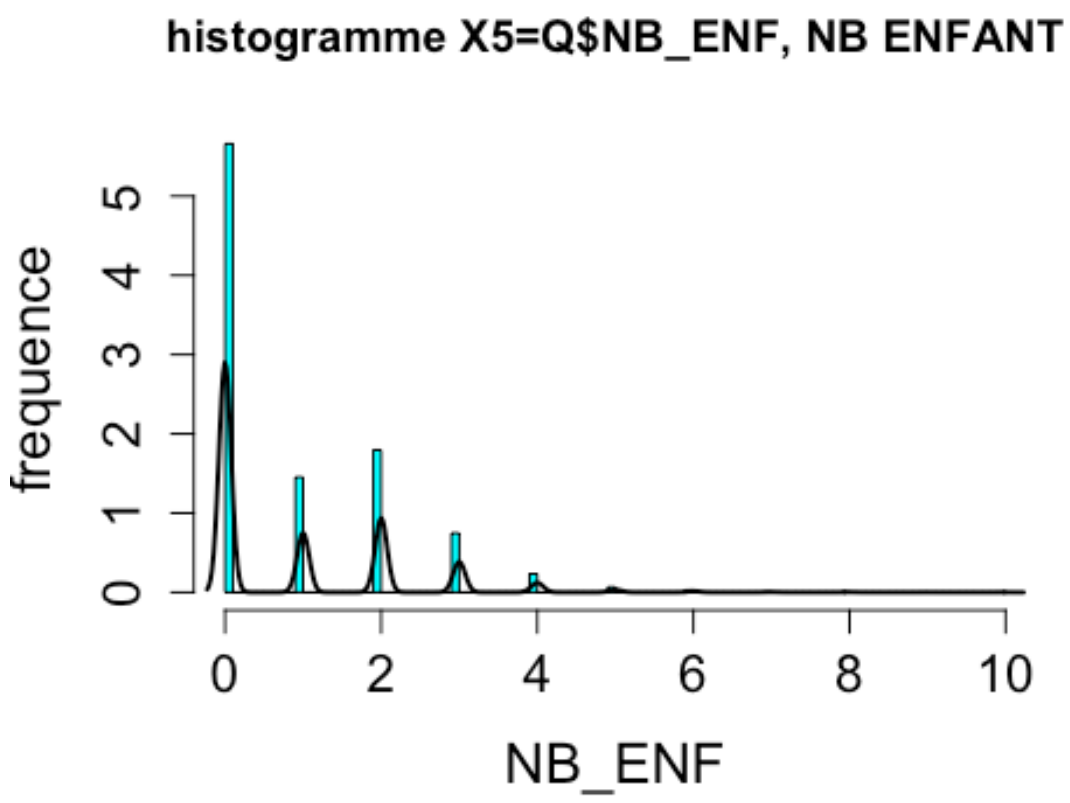

```
skewness(Q$COEFF, na.rm = T) # [1] 0.28

## [1] 0.2895364

kurtosis(Q$COEFF,na.rm=T) # [1] 1.75

## [1] 1.753285

# distribution multi-modales (5)

# PUISSANCE FISCAL
Titre="histogramme X3=Q$PUIS_FISC, Puissance Fiscale auto"
hist(X3,breaks=100,col="cyan",prob=TRUE,cex.axis=1.4,cex.lab=1.5,ylab="frequence", xlab="PUIS_FISC",main=
Titre)
lines(density(X3), lwd = 2)
```



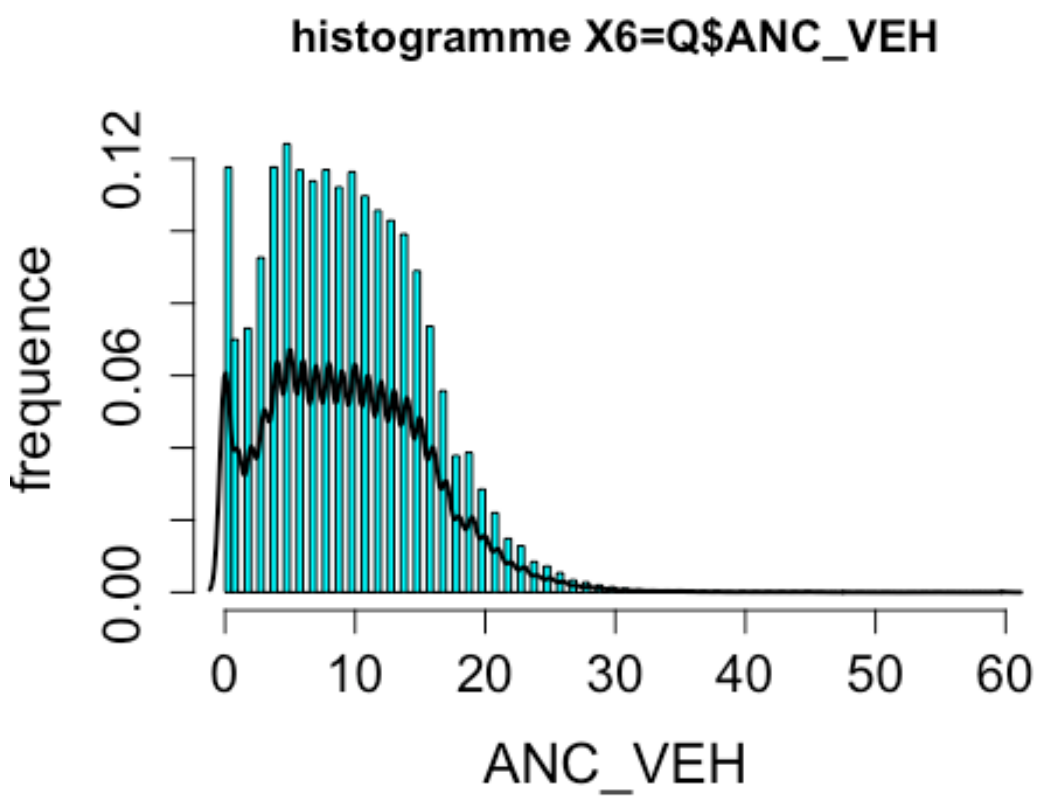

```
skewness(Q$PUIS_FISC, na.rm = T) # [1] 4.23

## [1] 4.2392

kurtosis(Q$PUIS_FISC,na.rm=T) # [1] 42.21

## [1] 42.21556

# distribution discrète étalée à droite

# AGE CLIENT
Titre="histogramme X4=Q$AGE_CLI, AGE CLIENT"
hist(X4,breaks=100,col="cyan",prob=TRUE,cex.axis=1.4,cex.lab=1.5,ylab="frequence", xlab="AGE_CLI",main=Ti
tre)
lines(density(X4), lwd = 2)
```



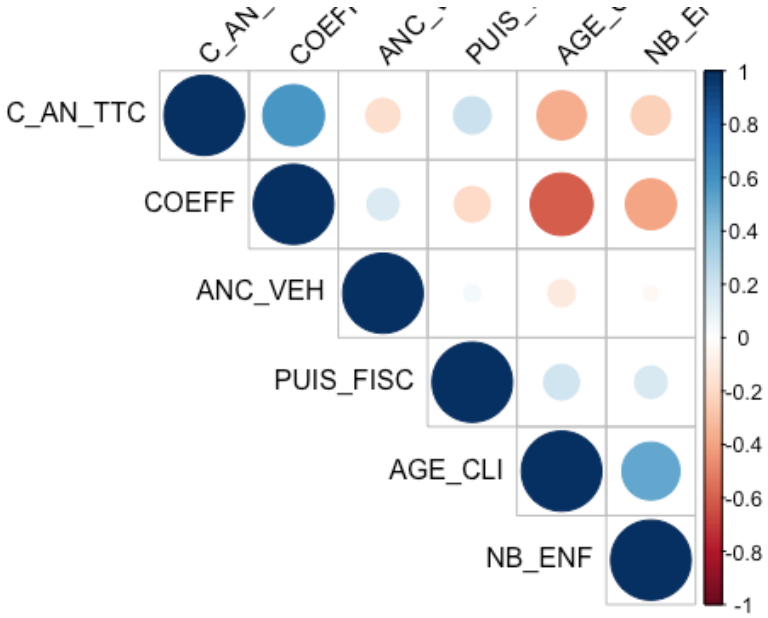

```
skewness(Q$AGE_CLI, na.rm = T) # [1] 0.67

## [1] 0.6724585

kurtosis(Q$AGE_CLI,na.rm=T) # [1] 2.43

## [1] 2.434862

# distribution continue bi-modales et étalée à droite

# NOMBRE D ENFANT
Titre="histogramme X5=Q$NB_ENF, NB ENFANT"
hist(X5,breaks=100,col="cyan",prob=TRUE,cex.axis=1.4,cex.lab=1.5,ylab="frequence", xlab="NB_ENF",main=Tit
re)
lines(density(X5), lwd = 2)
```




```
skewness(Q$NB_ENF, na.rm = T) # [1] 1.45

## [1] 1.455243

kurtosis(Q$NB_ENF,na.rm=T) # [1] 5.42

## [1] 5.429995

# distribution discrète, étalement à droite

# ANCIENNETE VEHICULE
Titre="histogramme X6=Q$ANC_VEH"
hist(X6,breaks=100,col="cyan",prob=TRUE,cex.axis=1.4,cex.lab=1.5,ylab="frequence", xlab="ANC_VEH",main=Ti
tre)
lines(density(X6), lwd = 2)
```




```
skewness(Q$ANC_VEH, na.rm = T) # [1] 0.75

## [1] 0.7512182

kurtosis(Q$ANC_VEH,na.rm=T) # [1] 4.61

## [1] 4.611772

# distribution continue étalée à droite

# 3. Indicateurs Correlations et associations & Graphiques de statistiques bivariées

# 3.1 corrélation linéaire entre les variables explicatives Xi
library(corrplot)
corrplot(cor(Q),type="upper", order="hclust",tl.col="black", tl.srt=45)
```



```r
# ANALYSE :
# . la Cotisation Annuelle est corrélée avec le Coefficient B/M,
# . le Coefficient est corrélé avec Age Client.
# .  Age Client avec NB d'enfant.

# COMMENTAIRE :
# . ces éléments sont illustratifs du modèle tarifaire MATMUT pour un contrat Auto.

# 3.2.1 comparer graphiquement les 2 sous-populations

# COTISATION
boxplot(dq2$C_AN_TTC ~ dq2$trsf_dev, main ="boîte à moustache Cotisation, selon Non / Oui")
```

```r
# la distribution de la cotisation se différencie entre les Non / oui
# plus forte chez les NON

# COEFF BONUS MALUS
boxplot(dq2$COEFF ~ dq2$trsf_dev, main ="boîte à moustache Coefficient, selon Non / Oui")
```

```r
# la distribution du coefficient se différencie entre les Non / oui
# plus forte chez les NON

# PUISSANCE FISCAL
boxplot(dq2$PUIS_FISC ~ dq2$trsf_dev, main ="boîte à moustache Puissance Fiscale, selon Non / Oui")
```

```r
# la distribution de la puissance fiscale se différencie entre les Non / oui
# plus forte chez les NON
```

```r
# AGE CLIENT
boxplot(dq2$AGE_CLI ~ dq2$trsf_dev, main ="boîte à moustache Age Client, selon Non / Oui")
```

```r
# la distribution de AGE CLIENT se différencie entre les Non / oui
# plus forte chez les NON

# NOMBRE D ENFANT
boxplot(dq2$NB_ENF ~ dq2$trsf_dev, main ="boîte à moustache Nombre d'enfant, selon Non / Oui")
```

```r
# la distribution du nombre d'enfant se différencie "mal" entre les Non / oui
# variable explicative à vérifier avec d'autres analyses pour décider si elle est bien discriminante.

# ANCIENNETE VEHICULE
boxplot(dq2$ANC_VEH ~ dq2$trsf_dev, main ="boîte à moustache Anciennete Vehicule, selon Non / Oui")
```

```r
# la distribution du Anciennete du Vehicule se différencie entre les Non / oui
# plus forte chez les oui

# 3.2.2  estimer les statistiques de bases par sous-populations

# COTISATION
tapply(dq2[,"C_AN_TTC"], dq2[,"trsf_dev"], mean, na.rm=TRUE)

##        N        O
## 751.9100 667.4761

tapply(dq2[,"C_AN_TTC"], dq2[,"trsf_dev"], sd, na.rm=TRUE)

##        N        O
## 450.4979 331.3307

tapply(dq2[,"C_AN_TTC"], dq2[,"trsf_dev"], var, na.rm=TRUE)

##        N        O
## 202948.4 109780.0

tapply(dq2[,"C_AN_TTC"], dq2[,"trsf_dev"], quantile, na.rm=TRUE)

## $N
##      0%     25%     50%     75%    100%
##    65.0   438.3   648.3   952.3 13335.3
##
## $O
##     0%    25%    50%    75%   100%
##   87.0  426.0  603.3  834.3 3307.3

tapply(dq2[,"C_AN_TTC"], dq2[,"trsf_dev"], skewness, na.rm=TRUE)

##        N        O
## 1.984051 1.281498

tapply(dq2[,"C_AN_TTC"], dq2[,"trsf_dev"], kurtosis, na.rm=TRUE)

##         N         O
## 12.683980  5.628591

tapply(dq2[,"C_AN_TTC"], dq2[,"trsf_dev"], density, na.rm=TRUE)

## $N
##
## Call:
##  density.default(x = X[[i]], na.rm = TRUE)
##
## Data: X[[i]] (450751 obs.);  Bandwidth 'bw' = 25.55
##
##        x                  y
```

```
##  Min.   :  -11.64   Min.   :0.000e+00
##  1st Qu.: 3344.26   1st Qu.:0.000e+00
##  Median : 6700.15   Median :2.750e-08
##  Mean   : 6700.15   Mean   :7.441e-05
##  3rd Qu.:10056.04   3rd Qu.:1.454e-06
##  Max.   :13411.94   Max.   :1.263e-03
##
## $O
##
## Call:
##  density.default(x = X[[i]], na.rm = TRUE)
##
## Data: X[[i]] (164177 obs.);  Bandwidth 'bw' = 24.83
##
##        x                y
##  Min.   :  12.5   Min.   :2.400e-09
##  1st Qu.: 854.8   1st Qu.:1.265e-06
##  Median :1697.2   Median :3.114e-05
##  Mean   :1697.2   Mean   :2.965e-04
##  3rd Qu.:2539.5   3rd Qu.:4.199e-04
##  Max.   :3381.8   Max.   :1.508e-03
```

```r
# analyse :
# sur la variable cotisation annuelle TTC, les valeurs calculées.
# sauf pour le 1er quartile, toutes les valeurs des metrics sont plus faibles chez les "Oui" que chez les "Non".

# COEFF BONUS MALUS
tapply(dq2[,"COEFF"], dq2[,"trsf_dev"], mean, na.rm=TRUE)
```

```
##        N        O
## 66.69916 65.06523
```

```r
tapply(dq2[,"COEFF"], dq2[,"trsf_dev"], sd, na.rm=TRUE)
```

```
##        N        O
## 24.94936 24.24227
```

```r
tapply(dq2[,"COEFF"], dq2[,"trsf_dev"], var, na.rm=TRUE)
```

```
##        N        O
## 622.4705 587.6878
```

```r
tapply(dq2[,"COEFF"], dq2[,"trsf_dev"], quantile, na.rm=TRUE)
```

```
## $N
##   0%  25%  50%  75% 100%
##   35   40   65   95  225
##
## $O
##   0%  25%  50%  75% 100%
##   35   40   65   90  148
```

```r
tapply(dq2[,"COEFF"], dq2[,"trsf_dev"], skewness, na.rm=TRUE)
```

```
##         N         O
## 0.2865105 0.2911275
```

```r
tapply(dq2[,"COEFF"], dq2[,"trsf_dev"], kurtosis, na.rm=TRUE)
```

```
##        N        O
## 1.764645 1.701488
```

```r
tapply(dq2[,"COEFF"], dq2[,"trsf_dev"], density, na.rm=TRUE)
```

```
## $N
##
## Call:
##  density.default(x = X[[i]], na.rm = TRUE)
##
## Data: X[[i]] (450751 obs.);  Bandwidth 'bw' = 1.662
##
##        x                y
```

```
##  Min.   : 30.02   Min.   :0.000e+00
##  1st Qu.: 80.01   1st Qu.:2.890e-06
##  Median :130.00   Median :6.884e-05
##  Mean   :130.00   Mean   :4.994e-03
##  3rd Qu.:179.99   3rd Qu.:5.537e-03
##  Max.   :229.98   Max.   :5.181e-02
##
## $O
##
## Call:
##  density.default(x = X[[i]], na.rm = TRUE)
##
## Data: X[[i]] (164177 obs.);  Bandwidth 'bw' = 1.976
##
##        x                y
##  Min.   : 29.07   Min.   :3.000e-08
##  1st Qu.: 60.29   1st Qu.:1.507e-04
##  Median : 91.50   Median :3.678e-03
##  Mean   : 91.50   Mean   :8.000e-03
##  3rd Qu.:122.71   3rd Qu.:8.197e-03
##  Max.   :153.93   Max.   :4.806e-02
```

```
# sauf pour le skewness, toutes les valeurs des metrics sont plus faibles chez les "Oui" que chez les "No
n".

# PUISSANCE FISCAL
tapply(dq2[,"PUIS_FISC"], dq2[,"trsf_dev"], mean, na.rm=TRUE)
```

```
##        N        O
## 5.778931 5.335138
```

```
tapply(dq2[,"PUIS_FISC"], dq2[,"trsf_dev"], sd, na.rm=TRUE)
```

```
##        N        O
## 2.398261 1.898446
```

```
tapply(dq2[,"PUIS_FISC"], dq2[,"trsf_dev"], var, na.rm=TRUE)
```

```
##        N        O
## 5.751654 3.604096
```

```
tapply(dq2[,"PUIS_FISC"], dq2[,"trsf_dev"], quantile, na.rm=TRUE)
```

```
## $N
##   0%  25%  50%  75% 100%
##    3    4    5    7   68
##
## $O
##   0%  25%  50%  75% 100%
##    3    4    5    6   51
```

```
tapply(dq2[,"PUIS_FISC"], dq2[,"trsf_dev"], skewness, na.rm=TRUE)
```

```
##        N        O
## 4.169483 4.270927
```

```
tapply(dq2[,"PUIS_FISC"], dq2[,"trsf_dev"], kurtosis, na.rm=TRUE)
```

```
##        N        O
## 40.52098 44.84106
```

```
tapply(dq2[,"PUIS_FISC"], dq2[,"trsf_dev"], density, na.rm=TRUE)
```

```
## $N
##
## Call:
##  density.default(x = X[[i]], na.rm = TRUE)
##
## Data: X[[i]] (450751 obs.);  Bandwidth 'bw' = 0.1491
##
##        x                y
##  Min.   : 2.553   Min.   :0.0000000
##  1st Qu.:19.026   1st Qu.:0.0000007
```

```
##  Median :35.500   Median :0.0000280
##  Mean   :35.500   Mean   :0.0151561
##  3rd Qu.:51.974   3rd Qu.:0.0004915
##  Max.   :68.447   Max.   :0.6317457
##
## $O
##
## Call:
##  density.default(x = X[[i]], na.rm = TRUE)
##
## Data: X[[i]] (164177 obs.);  Bandwidth 'bw' = 0.1216
##
##        x                y
##  Min.   : 2.635   Min.   :0.0000000
##  1st Qu.:14.818   1st Qu.:0.0000004
##  Median :27.000   Median :0.0000322
##  Mean   :27.000   Mean   :0.0204931
##  3rd Qu.:39.182   3rd Qu.:0.0005771
##  Max.   :51.365   Max.   :0.9891969

# sauf pour skewness & kurtosis, toutes les valeurs des autres metrics sont plus faibles chez les "Oui" q
ue chez les "Non".

# AGE CLIENT
tapply(dq2[,"AGE_CLI"], dq2[,"trsf_dev"], mean, na.rm=TRUE)

##        N        O
## 37.61410 34.84914

tapply(dq2[,"AGE_CLI"], dq2[,"trsf_dev"], sd, na.rm=TRUE)

##        N        O
## 15.74101 15.38760

tapply(dq2[,"AGE_CLI"], dq2[,"trsf_dev"], var, na.rm=TRUE)

##        N        O
## 247.7793 236.7783

tapply(dq2[,"AGE_CLI"], dq2[,"trsf_dev"], quantile, na.rm=TRUE)

## $N
##   0%  25%  50%  75% 100%
##   16   23   34   50   95
##
## $O
##   0%  25%  50%  75% 100%
##   17   22   28   47   95

tapply(dq2[,"AGE_CLI"], dq2[,"trsf_dev"], skewness, na.rm=TRUE)

##         N         O
## 0.6171345 0.8426014

tapply(dq2[,"AGE_CLI"], dq2[,"trsf_dev"], kurtosis, na.rm=TRUE)

##        N        O
## 2.392441 2.629879

tapply(dq2[,"AGE_CLI"], dq2[,"trsf_dev"], density, na.rm=TRUE)

## $N
##
## Call:
##  density.default(x = X[[i]], na.rm = TRUE)
##
## Data: X[[i]] (450751 obs.);  Bandwidth 'bw' = 1.048
##
##        x               y
##  Min.   :12.86   Min.   :6.000e-08
##  1st Qu.:34.18   1st Qu.:1.114e-03
##  Median :55.50   Median :1.031e-02
##  Mean   :55.50   Mean   :1.171e-02
```

```
##  3rd Qu.:76.82   3rd Qu.:1.736e-02
##  Max.   :98.14   Max.   :5.273e-02
##
## $O
##
## Call:
##  density.default(x = X[[i]], na.rm = TRUE)
##
## Data: X[[i]] (164177 obs.);  Bandwidth 'bw' = 1.254
##
##        x               y
##  Min.   :13.24   Min.   :9.000e-08
##  1st Qu.:34.62   1st Qu.:8.119e-04
##  Median :56.00   Median :9.208e-03
##  Mean   :56.00   Mean   :1.168e-02
##  3rd Qu.:77.38   3rd Qu.:1.460e-02
##  Max.   :98.76   Max.   :6.868e-02
```

```
# sauf pour skewness, kurtosis & density, toutes les valeurs des autres metrics sont plus faibles chez le
s "Oui" que chez les "Non".

# NOMBRE D ENFANT
tapply(dq2[,"NB_ENF"], dq2[,"trsf_dev"], mean, na.rm=TRUE)
```

```
##         N         O
## 0.9206258 0.7832157
```

```
tapply(dq2[,"NB_ENF"], dq2[,"trsf_dev"], sd, na.rm=TRUE)
```

```
##        N        O
## 1.236028 1.170544
```

```
tapply(dq2[,"NB_ENF"], dq2[,"trsf_dev"], var, na.rm=TRUE)
```

```
##        N        O
## 1.527765 1.370172
```

```
tapply(dq2[,"NB_ENF"], dq2[,"trsf_dev"], quantile, na.rm=TRUE)
```

```
## $N
##   0%  25%  50%  75% 100%
##    0    0    0    2   10
##
## $O
##   0%  25%  50%  75% 100%
##    0    0    0    2   10
```

```
tapply(dq2[,"NB_ENF"], dq2[,"trsf_dev"], skewness, na.rm=TRUE)
```

```
##        N        O
## 1.407730 1.597541
```

```
tapply(dq2[,"NB_ENF"], dq2[,"trsf_dev"], kurtosis, na.rm=TRUE)
```

```
##        N        O
## 5.311720 5.817007
```

```
tapply(dq2[,"NB_ENF"], dq2[,"trsf_dev"], density, na.rm=TRUE)
```

```
## $N
##
## Call:
##  density.default(x = X[[i]], na.rm = TRUE)
##
## Data: X[[i]] (450751 obs.);  Bandwidth 'bw' = 0.08232
##
##        x                 y
##  Min.   :-0.2469   Min.   :0.0000000
##  1st Qu.: 2.3765   1st Qu.:0.0000017
##  Median : 5.0000   Median :0.0002925
##  Mean   : 5.0000   Mean   :0.0950102
##  3rd Qu.: 7.6235   3rd Qu.:0.0120039
##  Max.   :10.2469   Max.   :2.6221703
```

```
##
## $O
##
## Call:
##  density.default(x = X[[i]], na.rm = TRUE)
##
## Data: X[[i]] (164177 obs.);  Bandwidth 'bw' = 0.0954
##
##        x                 y
##  Min.   :-0.2862   Min.   :0.0000000
##  1st Qu.: 2.3569   1st Qu.:0.0000112
##  Median : 5.0000   Median :0.0003738
##  Mean   : 5.0000   Mean   :0.0944350
##  3rd Qu.: 7.6431   3rd Qu.:0.0151621
##  Max.   :10.2862   Max.   :2.5289503

# sauf pour skewness, kurtosis & density, toutes les valeurs des autres metrics sont plus faibles chez

 les "Oui" que chez les "Non".

# ANCIENNETE VEHICULE
tapply(dq2[,"ANC_VEH"], dq2[,"trsf_dev"], mean, na.rm=TRUE)

##        N        O
## 9.461190 9.589541

tapply(dq2[,"ANC_VEH"], dq2[,"trsf_dev"], sd, na.rm=TRUE)

##        N        O
## 6.240353 6.064768

tapply(dq2[,"ANC_VEH"], dq2[,"trsf_dev"], var, na.rm=TRUE)

##        N        O
## 38.94201 36.78141

tapply(dq2[,"ANC_VEH"], dq2[,"trsf_dev"], quantile, na.rm=TRUE)

## $N
##   0%  25%  50%  75% 100%
##    0    5    9   13   60
##
## $O
##   0%  25%  50%  75% 100%
##    0    5    9   14   60

tapply(dq2[,"ANC_VEH"], dq2[,"trsf_dev"], skewness, na.rm=TRUE)

##         N         O
## 0.8402222 0.4874175

tapply(dq2[,"ANC_VEH"], dq2[,"trsf_dev"], kurtosis, na.rm=TRUE)

##        N        O
## 5.051273 3.275221

tapply(dq2[,"ANC_VEH"], dq2[,"trsf_dev"], density, na.rm=TRUE)

## $N
##
## Call:
##  density.default(x = X[[i]], na.rm = TRUE)
##
## Data: X[[i]] (450751 obs.);  Bandwidth 'bw' = 0.3976
##
##        x                y
##  Min.   :-1.193   Min.   :1.260e-06
##  1st Qu.:14.404   1st Qu.:9.968e-05
##  Median :30.000   Median :8.456e-04
##  Mean   :30.000   Mean   :1.601e-02
##  3rd Qu.:45.596   3rd Qu.:3.653e-02
##  Max.   :61.193   Max.   :6.706e-02
##
```

```
## $O
##
## Call:
##  density.default(x = X[[i]], na.rm = TRUE)
##
## Data: X[[i]] (164177 obs.);  Bandwidth 'bw' = 0.4943
##
##        x                y
##  Min.   :-1.483   Min.   :0.0000000
##  1st Qu.:14.259   1st Qu.:0.0000300
##  Median :30.000   Median :0.0004976
##  Mean   :30.000   Mean   :0.0158642
##  3rd Qu.:45.741   3rd Qu.:0.0372410
##  Max.   :61.483   Max.   :0.0610781
```

```
# moyenne, les OUI plus forts
# ecart-type, les OUI plus faibles
# variance, les OUI plus faibles
# quantile, distribution identique
# skewness, les NON plus forts
# kurtosis, les NON plus forts
# density, différentes
# # comportement différent de cette variable selon O/N pour les différentes métriques.

# 4. Tests statistiques

# 4.1. Tests d adequation a une loi donnee pour variable quantitative : test de normalité

# COTISATION
# test de normalité des "Oui"
dq2_e<-dq2[sample(1:nrow(dq2),5000),]
select.oui<-dq2_e[,"trsf_dev"]=="O"
shapiro.test(dq2_e[select.oui,"C_AN_TTC"])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  dq2_e[select.oui, "C_AN_TTC"]
## W = 0.90125, p-value < 2.2e-16
```

```
# analyse : valeur p-value < 5%, on rejette la normalité de la var C_AN_TTC pour les "Oui".

# test de normalité des "Non" avec C_AN_TTC
dq2_e<-dq2[sample(1:nrow(dq2),5000),]
select.oui<-dq2_e[,"trsf_dev"]=="N"
shapiro.test(dq2_e[select.oui,"C_AN_TTC"])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  dq2_e[select.oui, "C_AN_TTC"]
## W = 0.88829, p-value < 2.2e-16
```

```
# analyse : valeur p-value < 5%, on rejette la normalité de la var C_AN_TTC pour les "Non".
# conclusion : il faut normaliser cette var avec un transformation de variable.

# COEFF BONUS MALUS
# test de normalité des "Oui" avec COEFF
dq2_e<-dq2[sample(1:nrow(dq2),5000),]
select.oui<-dq2_e[,"trsf_dev"]=="O"
shapiro.test(dq2_e[select.oui,"COEFF"])
```

```
##
##  Shapiro-Wilk normality test
##
## data:  dq2_e[select.oui, "COEFF"]
## W = 0.8754, p-value < 2.2e-16
```

```
# analyse : valeur p-value < 5%, on rejette la normalité de la var COEFF pour les "Oui".
```

```r
# test de normalité des "Non" avec COEFF
dq2_e<-dq2[sample(1:nrow(dq2),5000),]
select.oui<-dq2_e[,"trsf_dev"]=="N"
shapiro.test(dq2_e[select.oui,"COEFF"])

##
##  Shapiro-Wilk normality test
##
## data:  dq2_e[select.oui, "COEFF"]
## W = 0.88329, p-value < 2.2e-16

# analyse : valeur p-value < 5%, on rejette la normalité de la var COEFF pour les "Non".
# conclusion : il faut normaliser cette var avec un transformation de variable.

# PUISSANCE FISCAL
# test de normalité des "Oui" avec PUIS_FISC
dq2_e<-dq2[sample(1:nrow(dq2),5000),]
select.oui<-dq2_e[,"trsf_dev"]=="O"
shapiro.test(dq2_e[select.oui,"PUIS_FISC"])

##
##  Shapiro-Wilk normality test
##
## data:  dq2_e[select.oui, "PUIS_FISC"]
## W = 0.79166, p-value < 2.2e-16

# analyse : valeur p-value < 5%, on rejette la normalité de la var PUIS_FISC pour les "Oui".

# test de normalité des "Non" avec PUIS_FISC
dq2_e<-dq2[sample(1:nrow(dq2),5000),]
select.oui<-dq2_e[,"trsf_dev"]=="N"
shapiro.test(dq2_e[select.oui,"PUIS_FISC"])

##
##  Shapiro-Wilk normality test
##
## data:  dq2_e[select.oui, "PUIS_FISC"]
## W = 0.74464, p-value < 2.2e-16

# analyse : valeur p-value < 5%, on rejette la normalité de la var PUIS_FISC pour les "Non".
# conclusion : il faut normaliser cette var avec un transformation de variable.

# test de normalité des "Oui" avec AGE_CLI
dq2_e<-dq2[sample(1:nrow(dq2),5000),]
select.oui<-dq2_e[,"trsf_dev"]=="O"
shapiro.test(dq2_e[select.oui,"AGE_CLI"])

##
##  Shapiro-Wilk normality test
##
## data:  dq2_e[select.oui, "AGE_CLI"]
## W = 0.87661, p-value < 2.2e-16

# analyse : valeur p-value < 5%, on rejette la normalité de la var AGE_CLI pour les "Oui".

# test de normalité des "Non" avec AGE_CLI
dq2_e<-dq2[sample(1:nrow(dq2),5000),]
select.oui<-dq2_e[,"trsf_dev"]=="N"
shapiro.test(dq2_e[select.oui,"AGE_CLI"])

##
##  Shapiro-Wilk normality test
##
## data:  dq2_e[select.oui, "AGE_CLI"]
## W = 0.91457, p-value < 2.2e-16

# analyse : valeur p-value < 5%, on rejette la normalité de la var AGE_CLI pour les "Non".
# conclusion : il faut normaliser cette var avec un transformation de variable.
```

```r
# NOMBRE D ENFANT
# test de normalité des "Oui" avec NB_ENF
dq2_e<-dq2[sample(1:nrow(dq2),5000),]
select.oui<-dq2_e[,"trsf_dev"]=="O"
shapiro.test(dq2_e[select.oui,"NB_ENF"])

##
##  Shapiro-Wilk normality test
##
## data:  dq2_e[select.oui, "NB_ENF"]
## W = 0.73222, p-value < 2.2e-16

# analyse : valeur p-value < 5%, on rejette la normalité de la var NB_ENF pour les "Oui".

# test de normalité des "Non" avec NB_ENF
dq2_e<-dq2[sample(1:nrow(dq2),5000),]
select.oui<-dq2_e[,"trsf_dev"]=="N"
shapiro.test(dq2_e[select.oui,"NB_ENF"])

##
##  Shapiro-Wilk normality test
##
## data:  dq2_e[select.oui, "NB_ENF"]
## W = 0.74663, p-value < 2.2e-16

# analyse : valeur p-value < 5%, on rejette la normalité de la var PUIS_FISC pour les "Non".
# conclusion : il faut normaliser cette var avec un transformation de variable.

# ANCIENNETE VEHICULE
# test de normalité des "Oui" avec ANC_VEH
dq2_e<-dq2[sample(1:nrow(dq2),5000),]
select.oui<-dq2_e[,"trsf_dev"]=="O"
shapiro.test(dq2_e[select.oui,"ANC_VEH"])

##
##  Shapiro-Wilk normality test
##
## data:  dq2_e[select.oui, "ANC_VEH"]
## W = 0.97198, p-value = 2.204e-15

# analyse : valeur p-value < 5%, on rejette la normalité de la var ANC_VEH pour les "Oui".

# test de normalité des "Non" avec ANC_VEH
dq2_e<-dq2[sample(1:nrow(dq2),5000),]
select.oui<-dq2_e[,"trsf_dev"]=="N"
shapiro.test(dq2_e[select.oui,"ANC_VEH"])

##
##  Shapiro-Wilk normality test
##
## data:  dq2_e[select.oui, "ANC_VEH"]
## W = 0.96243, p-value < 2.2e-16

# analyse : valeur p-value < 5%, on rejette la normalité de la var ANC_VEH pour les "Non".
# conclusion : il faut normaliser cette var avec un transformation de variable.



# 4.2. Test de comparaison de populations :

# 4.2.1 test d'égalité de variance

# COTISATION
var.test(C_AN_TTC~trsf_dev, conf.level=.95,data=dq2)

##
##  F test to compare two variances
##
## data:  C_AN_TTC by trsf_dev
## F = 1.8487, num df = 450750, denom df = 164180, p-value < 2.2e-16
## alternative hypothesis: true ratio of variances is not equal to 1
```

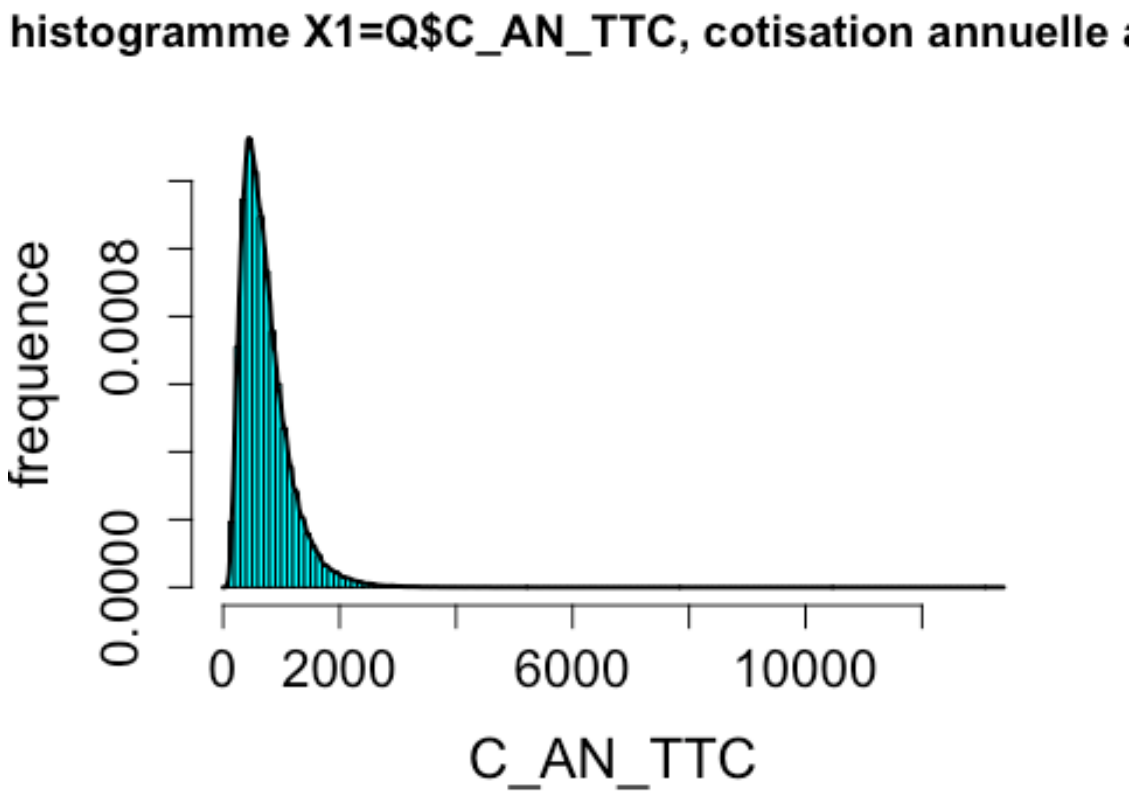
```
## 95 percent confidence interval:
##  1.836058 1.861351
## sample estimates:
## ratio of variances
##           1.848683

# analyse : valeur p-value < à 5%, la variance est significativement différence pour la var C_AN_TTC
# entre le groupe des  "Oui" et le groupe des "Non" avec un intervalle de confiance à 95%.

# COEFF BONUS MALUS
var.test(COEFF~trsf_dev, conf.level=.95,data=dq2)

##
##  F test to compare two variances
##
## data:  COEFF by trsf_dev
## F = 1.0592, num df = 450750, denom df = 164180, p-value < 2.2e-16
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
##  1.051952 1.066444
## sample estimates:
## ratio of variances
##           1.059186

# analyse : valeur p-value < à 5%, la variance est significativement différence pour la var COEFF
# entre le groupe des "Oui" et le groupe des "Non" avec un intervalle de confiance à 95%.

# PUISSANCE FISCAL
var.test(PUIS_FISC~trsf_dev, conf.level=.95,data=dq2)

##
##  F test to compare two variances
##
## data:  PUIS_FISC by trsf_dev
## F = 1.5959, num df = 450750, denom df = 164180, p-value < 2.2e-16
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
##  1.584968 1.606802
## sample estimates:
## ratio of variances
##           1.595866

# analyse : valeur p-value < à 5%, la variance est significativement différence pour la var PUIS_FISC
# entre le groupe des "Oui" et le groupe des "Non" avec un intervalle de confiance à 95%.


# AGE CLIENT
var.test(AGE_CLI~trsf_dev, conf.level=.95,data=dq2)

##
##  F test to compare two variances
##
## data:  AGE_CLI by trsf_dev
## F = 1.0465, num df = 450750, denom df = 164180, p-value < 2.2e-16
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
##  1.039315 1.053632
## sample estimates:
## ratio of variances
##           1.046461

# analyse : valeur p-value < à 5%, la variance est significativement différence pour la var AGE_CLI
# entre le groupe des  "Oui" et le groupe des "Non" avec un intervalle de confiance à 95%.

# NOMBRE D ENFANT
var.test(NB_ENF~trsf_dev, conf.level=.95,data=dq2)

##
##  F test to compare two variances
##
## data:  NB_ENF by trsf_dev
## F = 1.115, num df = 450750, denom df = 164180, p-value < 2.2e-16
```

```
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
##  1.107402 1.122657
## sample estimates:
## ratio of variances
##           1.115017

# analyse : valeur p-value < à 5%, la variance est significativement différence pour la var NB_ENF
# entre le groupe des  "Oui" et le groupe des "Non" avec un intervalle de confiance à 95%.

# ANCIENNETE VEHICULE
var.test(ANC_VEH~trsf_dev, conf.level=.95,data=dq2)

##
##  F test to compare two variances
##
## data:  ANC_VEH by trsf_dev
## F = 1.0587, num df = 450750, denom df = 164180, p-value < 2.2e-16
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
##  1.051511 1.065997
## sample estimates:
## ratio of variances
##           1.058742

# analyse : valeur p-value < à 5%, la variance est significativement différence pour la var ANC_VEH
# entre le groupe des  "Oui" et le groupe des "Non" avec un intervalle de confiance à 95%.


# 4.2.2 test d'égalité de moyenne
# les variances étant différentes, on utilise le test de WELCH pour tester l'égalité des moyennes

# COTISATION
t.test(C_AN_TTC~trsf_dev, alternative='two.sided',conf.level=0.95,var.equal=FALSE, data=dq2)

##
##  Welch Two Sample t-test
##
## data:  C_AN_TTC by trsf_dev
## t = 79.821, df = 394550, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  82.36072 86.50718
## sample estimates:
## mean in group N mean in group O
##        751.9100        667.4761

# analyse : C_AN_TTC~trsf_dev, on rejette l'égalité des moyennes, la moyenne des Cotisations des "Oui"
# est significativement différente de la moyenne des Cotisations des "Non" avec un intervalle de confianc
e à 95%.
# le comportement de la transformation est différent différent on fonction de la cotise,
# et donc de la gamme de véhicule à assurer
# CONCLUSION : la var explicative C_AN_TTC discrimine bien la var à expliquer taux de transformation (Oui
/ Non).

# COEFF BONUS MALUS
t.test(COEFF~trsf_dev, alternative='two.sided',conf.level=0.95,var.equal=FALSE, data=dq2)

##
##  Welch Two Sample t-test
##
## data:  COEFF by trsf_dev
## t = 23.199, df = 299070, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  1.495887 1.771973
## sample estimates:
## mean in group N mean in group O
##        66.69916        65.06523
```

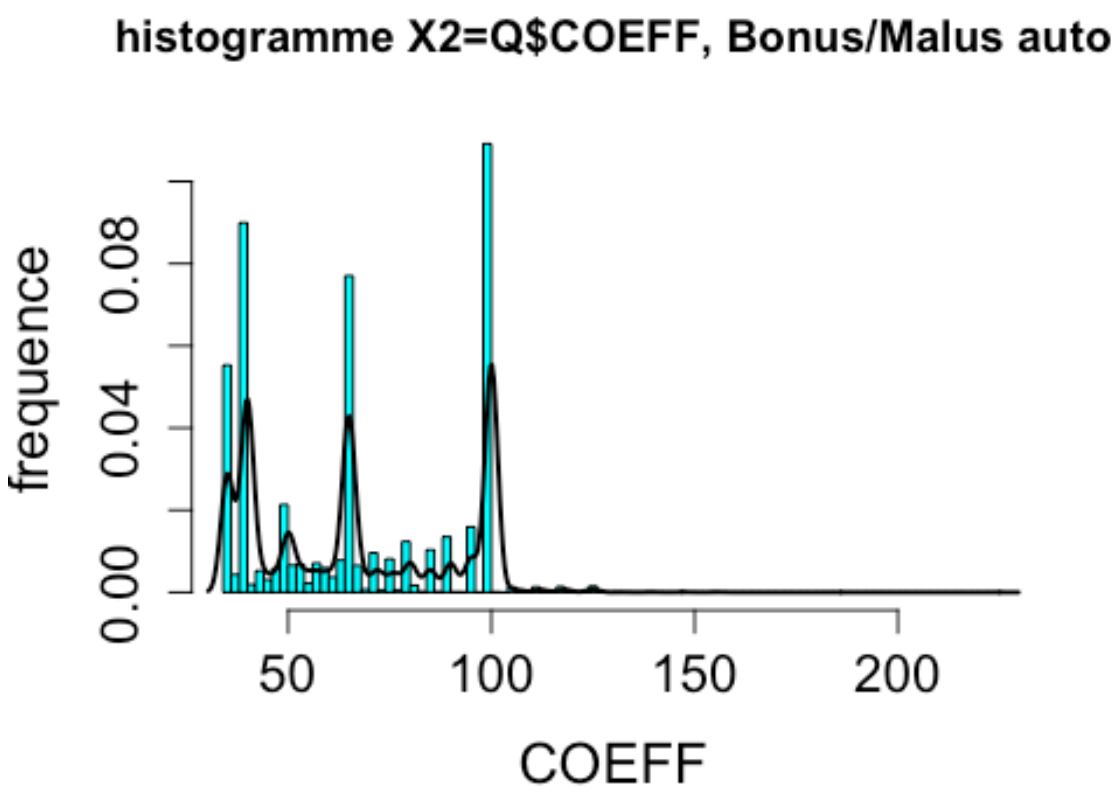
```r
# analyse : COEFF~trsf_dev, on rejette l'égalité des moyennes, la moyenne des coefficients des "Oui"
# est significativement différente de la moyenne des coefficients des "Non" avec un intervalle de confian
ce à 95%.
# le comportement de la transformation est différent différent on fonction du coefficient Bonus / Malus,
# CONCLUSION : la var explicative COEFF discrimine bien la var à expliquer taux de transformation (Oui /
Non).

# PUISSANCE FISCAL
t.test(PUIS_FISC~trsf_dev, alternative='two.sided',conf.level=0.95,var.equal=FALSE, data=dq2)

##
##  Welch Two Sample t-test
##
## data:  PUIS_FISC by trsf_dev
## t = 75.325, df = 365520, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.4322452 0.4553405
## sample estimates:
## mean in group N mean in group O
##        5.778931        5.335138

# analyse : PUIS_FISC~trsf_dev, on rejette l'égalité des moyennes, la moyenne des coefficients des "Oui"
# est significativement différente de la moyenne des coefficients des "Non" avec un intervalle de confian
ce à 95%.
# le comportement de la transformation est différent différent on fonction de la puissance fiscale du véh
icule,
# CONCLUSION : la var explicative PUIS_FISC discrimine bien la var à expliquer taux de transformation (Ou
i / Non).

# AGE CLIENT
t.test(AGE_CLI~trsf_dev, alternative='two.sided',conf.level=0.95,var.equal=FALSE, data=dq2)

##
##  Welch Two Sample t-test
##
## data:  AGE_CLI by trsf_dev
## t = 61.952, df = 297440, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  2.677482 2.852433
## sample estimates:
## mean in group N mean in group O
##        37.61410        34.84914

# analyse :AGE_CLI~trsf_dev, on rejette l'égalité des moyennes, la moyenne des coefficients des "Oui"
# est significativement différente de la moyenne des coefficients des "Non" avec un intervalle de confian
ce à 95%.
# le comportement de la transformation est différent différent on fonction de l'âge du client,
# CONCLUSION : la var explicative AGE_CLI discrimine bien la var à expliquer taux de transformation (Oui
/ Non).

# NOMBRE D ENFANT
t.test(NB_ENF~trsf_dev, alternative='two.sided',conf.level=0.95,var.equal=FALSE, data=dq2)

##
##  Welch Two Sample t-test
##
## data:  NB_ENF by trsf_dev
## t = 40.112, df = 306210, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.1306959 0.1441243
## sample estimates:
## mean in group N mean in group O
##       0.9206258       0.7832157

# analyse :NB_ENF~trsf_dev, on rejette l'égalité des moyennes, la moyenne des coefficients des "Oui"
# est significativement différente de la moyenne des coefficients des "Non" avec un intervalle de confian
ce à 95%.
```

```
# le comportement de la transformation est différent différent on fonction du nombre d'enfant,
# CONCLUSION : la var explicative NB_ENF discrimine bien la var à expliquer taux de transformation (Oui /
Non).

# ANCIENNETE VEHICULE
t.test(ANC_VEH~trsf_dev, alternative='two.sided',conf.level=0.95,var.equal=FALSE, data=dq2)

##
##  Welch Two Sample t-test
##
## data:  ANC_VEH by trsf_dev
## t = -7.2848, df = 299020, p-value = 3.23e-13
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.16288295 -0.09381751
## sample estimates:
## mean in group N mean in group O
##        9.461190        9.589541

# analyse :ANC_VEH~trsf_dev, on rejette l'égalité des moyennes, la moyenne des coefficients des "Oui"
# est significativement différente de la moyenne des coefficients des "Non" avec un intervalle de confian
ce à 95%.
# le comportement de la transformation est différent différent on fonction de l'ancienneté du véhicule,
# CONCLUSION : la var explicative ANC_VEH discrimine bien la var à expliquer taux de transformation (Oui
/ Non).

# 5. Detection des points aberrants (Outliers)

library(outliers)

##
## Attaching package: 'outliers'

## The following object is masked from 'package:randomForest':
##
##     outlier

# source : Anne B. Dufour, 11 juillet 2008, Quelques tests pour les Sciences Forensiques, page9.
# Le test du Chi-Deux pour les valeurs extremes
# La fonction realise un test simple pour d´etecter la presence d'une valeur extreme.
# Le test est base sur une distribution du chi-deux des carres des differences
# entre les donnees et la moyenne de l' ´echantillon [1]. La variance de la population
# est supposee connue.


Q= dq2[,1:6]
summary(Q)

##     C_AN_TTC           COEFF          PUIS_FISC        AGE_CLI
##  Min.   :   65.0   Min.   : 35.00   Min.   : 3.00   Min.   :16.00
##  1st Qu.:  434.3   1st Qu.: 40.00   1st Qu.: 4.00   1st Qu.:23.00
##  Median :  635.3   Median : 65.00   Median : 5.00   Median :32.00
##  Mean   :  729.4   Mean   : 66.26   Mean   : 5.66   Mean   :36.88
##  3rd Qu.:  915.3   3rd Qu.: 95.00   3rd Qu.: 6.00   3rd Qu.:49.00
##  Max.   :13335.3   Max.   :225.00   Max.   :68.00   Max.   :95.00
##      NB_ENF           ANC_VEH
##  Min.   : 0.0000   Min.   : 0.000
##  1st Qu.: 0.0000   1st Qu.: 5.000
##  Median : 0.0000   Median : 9.000
##  Mean   : 0.8839   Mean   : 9.495
##  3rd Qu.: 2.0000   3rd Qu.:14.000
##  Max.   :10.0000   Max.   :60.000

# C_AN_TTC           COEFF          PUIS_FISC        AGE_CLI          NB_ENF           ANC_VEH
# Min.   :   65.0   Min.   : 35.00   Min.   : 3.00   Min.   :16.00   Min.   : 0.0000   Min.   : 0.000
# 1st Qu.:  434.3   1st Qu.: 40.00   1st Qu.: 4.00   1st Qu.:23.00   1st Qu.: 0.0000   1st Qu.: 5.000
# Median :  635.3   Median : 65.00   Median : 5.00   Median :32.00   Median : 0.0000   Median : 9.000
# Mean   :  729.4   Mean   : 66.26   Mean   : 5.66   Mean   :36.88   Mean   : 0.8839   Mean   : 9.495
# 3rd Qu.:  915.3   3rd Qu.: 95.00   3rd Qu.: 6.00   3rd Qu.:49.00   3rd Qu.: 2.0000   3rd Qu.:14.000
# Max.   :13335.3   Max.   :225.00   Max.   :68.00   Max.   :95.00   Max.   :10.0000   Max.   :60.000
```

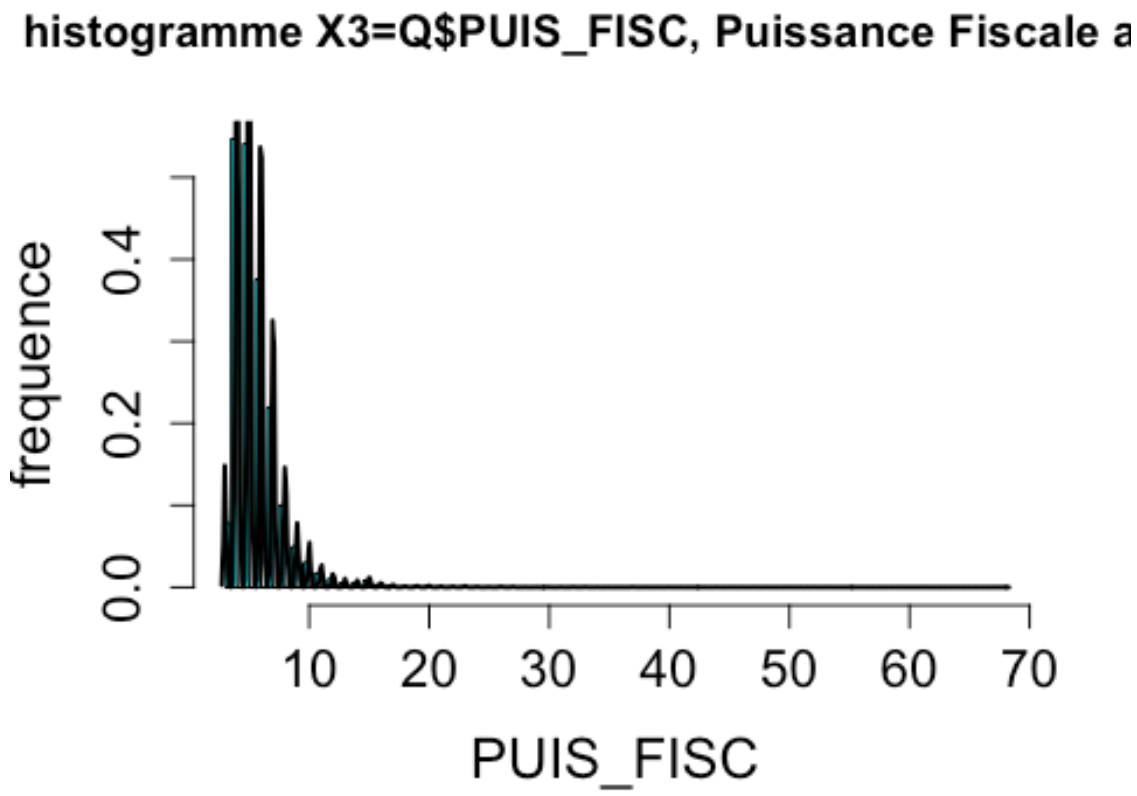
```r
# COTISATION
Titre="histogramme X1=Q$C_AN_TTC, cotisation annuelle auto"
hist(X1,breaks=100,col="cyan",prob=TRUE,cex.axis=1.4,cex.lab=1.5,ylab="frequence", xlab="C_AN_TTC",main=T
itre)
lines(density(X1), lwd = 2)
```



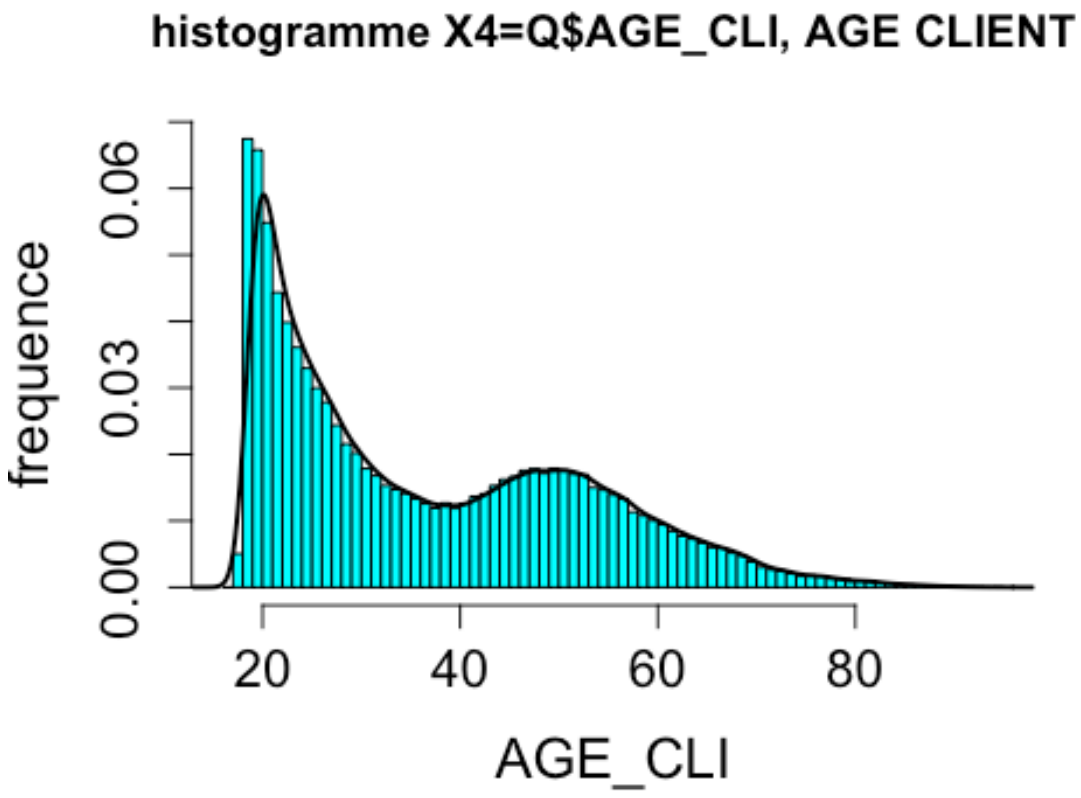

```r
chisq.out.test(X1, variance=var(X1), opposite = FALSE)

##
##  chi-squared test for outlier
##
## data:  X1
## X-squared = 885.44, p-value < 2.2e-16
## alternative hypothesis: highest value 13335.3 is an outlier

# analyse : valeur p-value < 5%, on rejette la présence d'outlers dans Xi Cotisation

# COEFF BONUS MALUS
Titre="histogramme X2=Q$COEFF, Bonus/Malus auto"
hist(X2,breaks=100,col="cyan",prob=TRUE,cex.axis=1.4,cex.lab=1.5,ylab="frequence", xlab="COEFF",main=Titr
e)
lines(density(X2), lwd = 2)
```



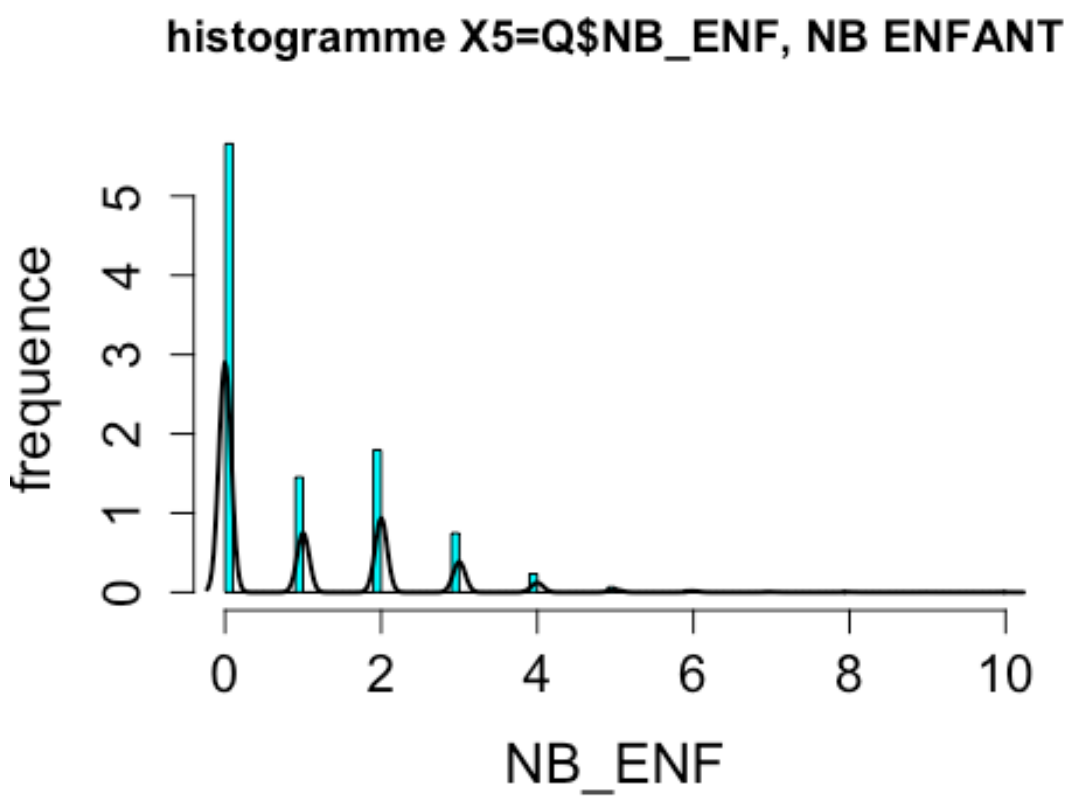

```
chisq.out.test(X2, variance=var(X2), opposite = FALSE)

##
##  chi-squared test for outlier
##
## data:  X2
## X-squared = 41.058, p-value = 1.478e-10
## alternative hypothesis: highest value 225 is an outlier

# analyse : valeur p-value < 5%, on rejette la présence d'outlers dans Xi COEFF BONUS MALUS

# PUISSANCE FISCAL
Titre="histogramme X3=Q$PUIS_FISC, Puissance Fiscale auto"
hist(X3,breaks=100,col="cyan",prob=TRUE,cex.axis=1.4,cex.lab=1.5,ylab="frequence", xlab="PUIS_FISC",main=
Titre)
lines(density(X3), lwd = 2)
```



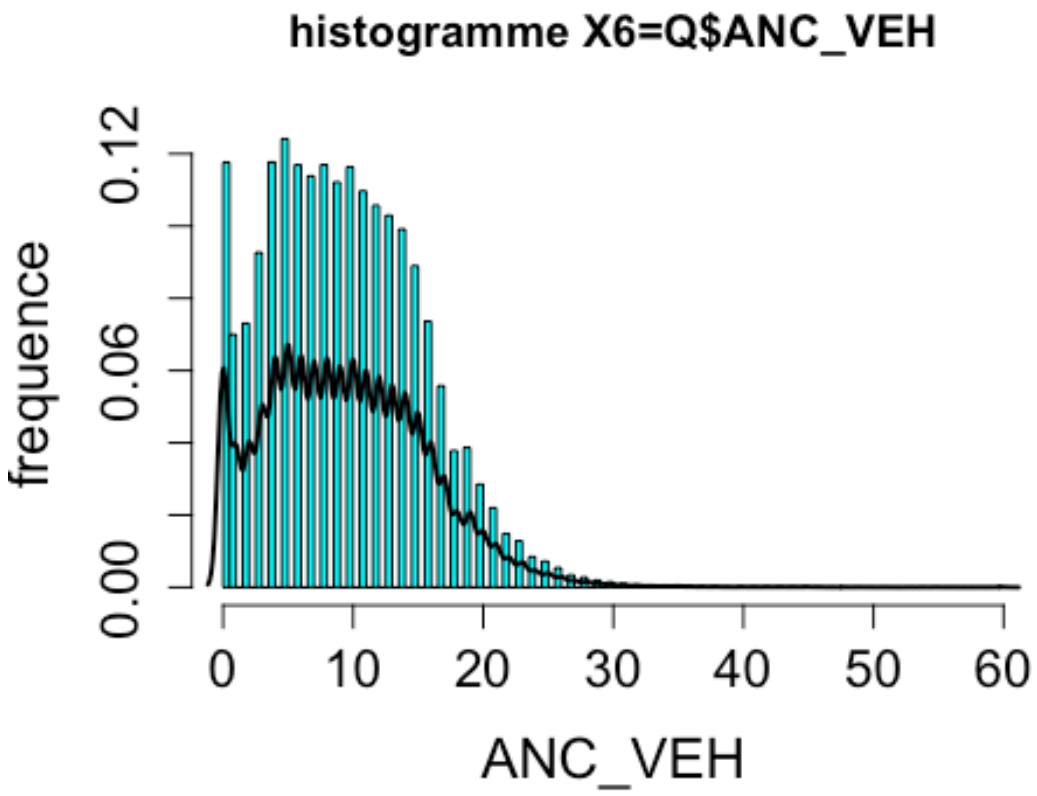

```
chisq.out.test(X3, variance=var(X3), opposite = FALSE)

##
##  chi-squared test for outlier
##
## data:  X3
## X-squared = 744.94, p-value < 2.2e-16
## alternative hypothesis: highest value 68 is an outlier

# analyse : valeur p-value < 5%, on rejette la présence d'outlers dans Xi PUISSANCE FISCAL

# AGE CLIENT
Titre="histogramme X4=Q$AGE_CLI, AGE CLIENT"
hist(X4,breaks=100,col="cyan",prob=TRUE,cex.axis=1.4,cex.lab=1.5,ylab="frequence", xlab="AGE_CLI",main=Ti
tre)
lines(density(X4), lwd = 2)
```




```
chisq.out.test(X4, variance=var(X4), opposite = FALSE)

##
##  chi-squared test for outlier
##
## data:  X4
## X-squared = 13.715, p-value = 0.0002128
## alternative hypothesis: highest value 95 is an outlier

# analyse : valeur p-value < 5%, on rejette la présence d'outliers dans Xi AGE CLIENT

# NOMBRE D ENFANT
Titre="histogramme X5=Q$NB_ENF, NB ENFANT"
hist(X5,breaks=100,col="cyan",prob=TRUE,cex.axis=1.4,cex.lab=1.5,ylab="frequence", xlab="NB_ENF",main=Tit
re)
lines(density(X5), lwd = 2)
```

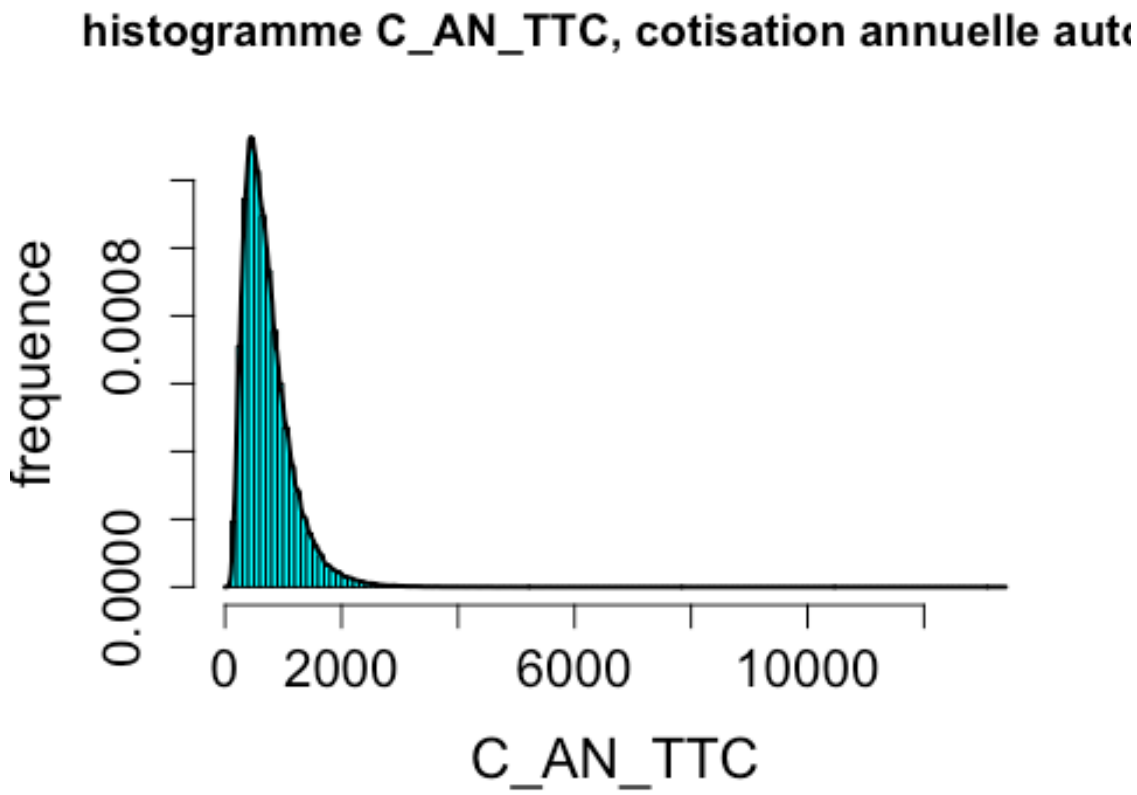



```
chisq.out.test(X5, variance=var(X5), opposite = FALSE)

##
##  chi-squared test for outlier
##
## data:  X5
## X-squared = 55.797, p-value = 8.038e-14
## alternative hypothesis: highest value 10 is an outlier

# analyse : valeur p-value < 5%, on rejette la présence d'outliers dans Xi NOMBRE D ENFANT

# ANCIENNETE VEHICULE
Titre="histogramme X6=Q$ANC_VEH"
hist(X6,breaks=100,col="cyan",prob=TRUE,cex.axis=1.4,cex.lab=1.5,ylab="frequence", xlab="ANC_VEH",main=Ti
tre)
lines(density(X6), lwd = 2)
```

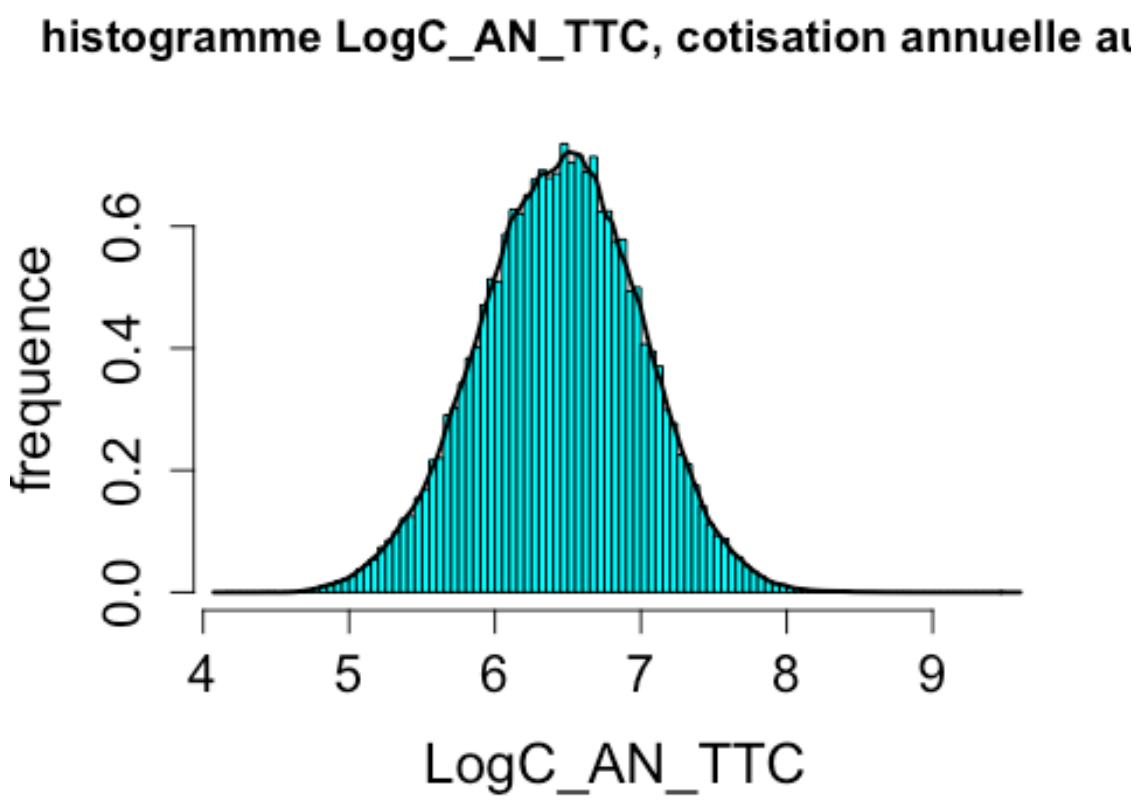



```
chisq.out.test(X6, variance=var(X6), opposite = FALSE)

##
##  chi-squared test for outlier
##
## data:  X6
## X-squared = 66.48, p-value = 3.331e-16
## alternative hypothesis: highest value 60 is an outlier

# analyse : valeur p-value < 5%, on rejette la présence d'outliers dans Xi ANCIENNETE VEHICULE

# 6. valeurs manquantes

Pourcentage_NA = colMeans(is.na(Q))
Pourcentage_NA = Pourcentage_NA * 100
Pourcentage_NA

##  C_AN_TTC     COEFF PUIS_FISC   AGE_CLI    NB_ENF   ANC_VEH
##         0         0         0         0         0         0

summary(Pourcentage_NA)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       0       0       0       0       0       0

mean(Pourcentage_NA)

## [1] 0

# résultats [1] 0
# analyse : pas de valeurs manquante dans les données

# 7. transformation ie changement de var pour rendre certaines distributions
# plus symetriques et plus gaussiennes.

# Sknewness
# Une valeur nulle du Sknewness indique une distribution symetrique des donnees
# autour de leur moyenne alors qu un Skewness positif (respectivement negatif)
# indique une queue distribution plus etalee vers la droite (resp. vers la gauche).
```

```r
# Kurtosis
# Plus le Kurtosis est eleve, plus il y a de valeurs eloignees de la moyenne,
# les queues de distributions sont plus epaisses
# (que celles de la gaussienne, dont le kurtosis est 3).

# 7.1. COTISATION

# dq2T = dataset base de transformatiob
dq2T=dq2

# avant
par(mfrow=c(1,1))
Titre="histogramme C_AN_TTC, cotisation annuelle auto"
hist(X1,breaks=100,col="cyan",prob=TRUE,cex.axis=1.4,cex.lab=1.5,ylab="frequence", xlab="C_AN_TTC",main=T
itre)
lines(density(X1), lwd = 2)
```

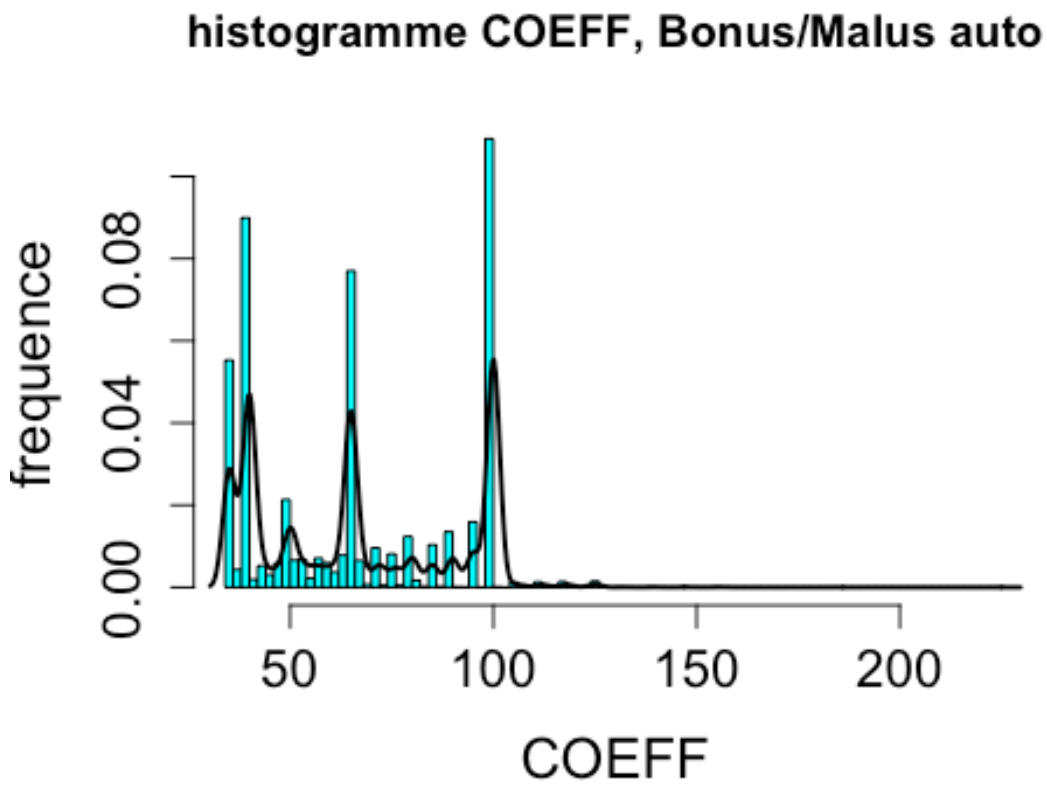



```r
skewness(Q$C_AN_TTC, na.rm = T) # [1] 1.97

## [1] 1.972588

kurtosis(Q$C_AN_TTC,na.rm=T) # [1] 12.76

## [1] 12.76343

# distribution continue étalée à droite

# CHANGEMENT DE VARIABLE
# on rajouter 1 var, donc 1 colonne
dq2T[,"LogC_AN_TTC"]=log(dq2T[,"C_AN_TTC"])
QT= dq2T[,1:8]
X1Ta=QT$LogC_AN_TTC

# apres(a)
Titre="histogramme LogC_AN_TTC, cotisation annuelle auto"
hist(X1Ta,breaks=100,col="cyan",prob=TRUE,cex.axis=1.4,cex.lab=1.5,ylab="frequence", xlab="LogC_AN_TTC",m
ain=Titre)
lines(density(X1Ta), lwd = 2)
```

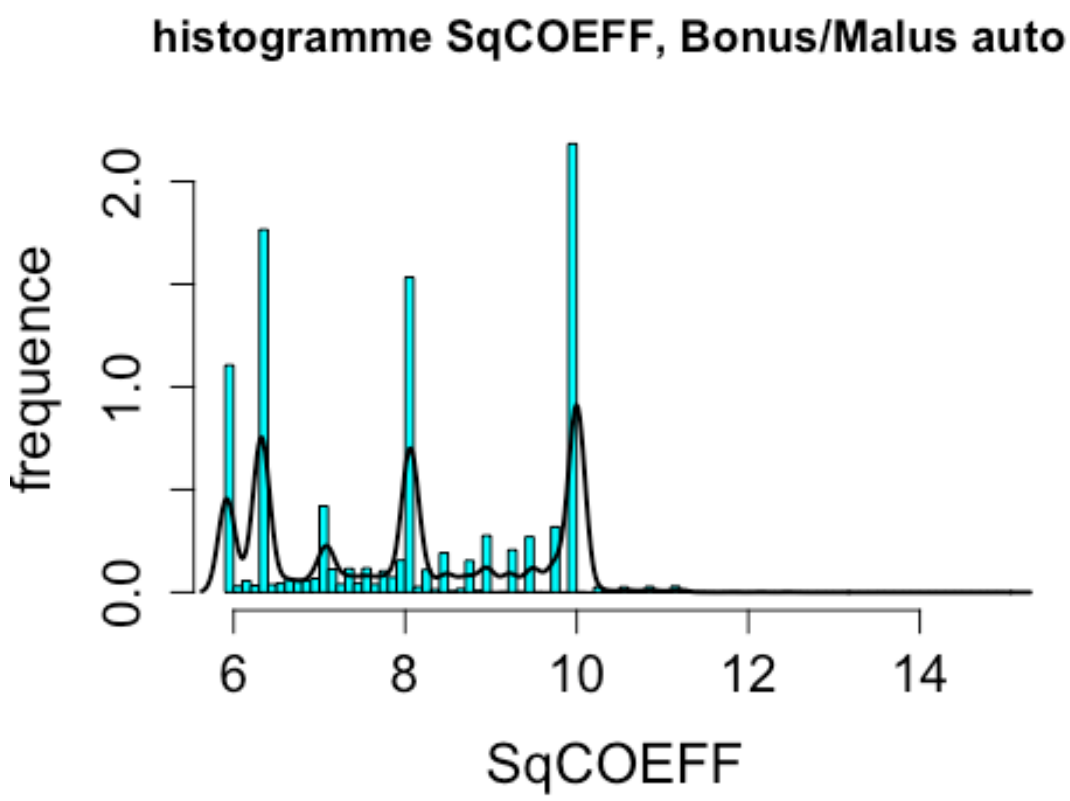



```
skewness(X1Ta, na.rm = T) # [1] -0.04

## [1] -0.04412198

kurtosis(X1Ta,na.rm=T) # [1] 2.9

## [1] 2.901051

# test de normalité des "Oui" avec LogC_AN_TTC
dq2_ea<-dq2T[sample(1:nrow(dq2T),5000),]
select.ouia<-dq2_ea[,"trsf_dev"]=="O"
shapiro.test(dq2_ea[select.ouia,"LogC_AN_TTC"])

##
##  Shapiro-Wilk normality test
##
## data:  dq2_ea[select.ouia, "LogC_AN_TTC"]
## W = 0.9966, p-value = 0.005592

# p-value = 0.14
# analyse : valeur p-value > 5%, on accepte la normalité de la var LogC_AN_TTC pour les "Oui".

# test de normalité des "Non" avec LogC_AN_TTC
dq2_ea<-dq2T[sample(1:nrow(dq2T),5000),]
select.ouia<-dq2_ea[,"trsf_dev"]=="N"
shapiro.test(dq2_ea[select.ouia,"LogC_AN_TTC"])

##
##  Shapiro-Wilk normality test
##
## data:  dq2_ea[select.ouia, "LogC_AN_TTC"]
## W = 0.99897, p-value = 0.02714

# p-value =  0.23
# analyse : valeur p-value > 5%, on accepte la normalité de la var LogC_AN_TTC pour les "Non".

#_____________________________________
# conclusion : on accepte la variable transformée LogC_AN_TTC
#_____________________________________
```

```r
# 7.2. COEFF BONUS MALUS

par(mfrow=c(1,1))

# avant
Titre="histogramme COEFF, Bonus/Malus auto"
hist(X2,breaks=100,col="cyan",prob=TRUE,cex.axis=1.4,cex.lab=1.5,ylab="frequence", xlab="COEFF",main=Titr
e)
lines(density(X2), lwd = 2)
```

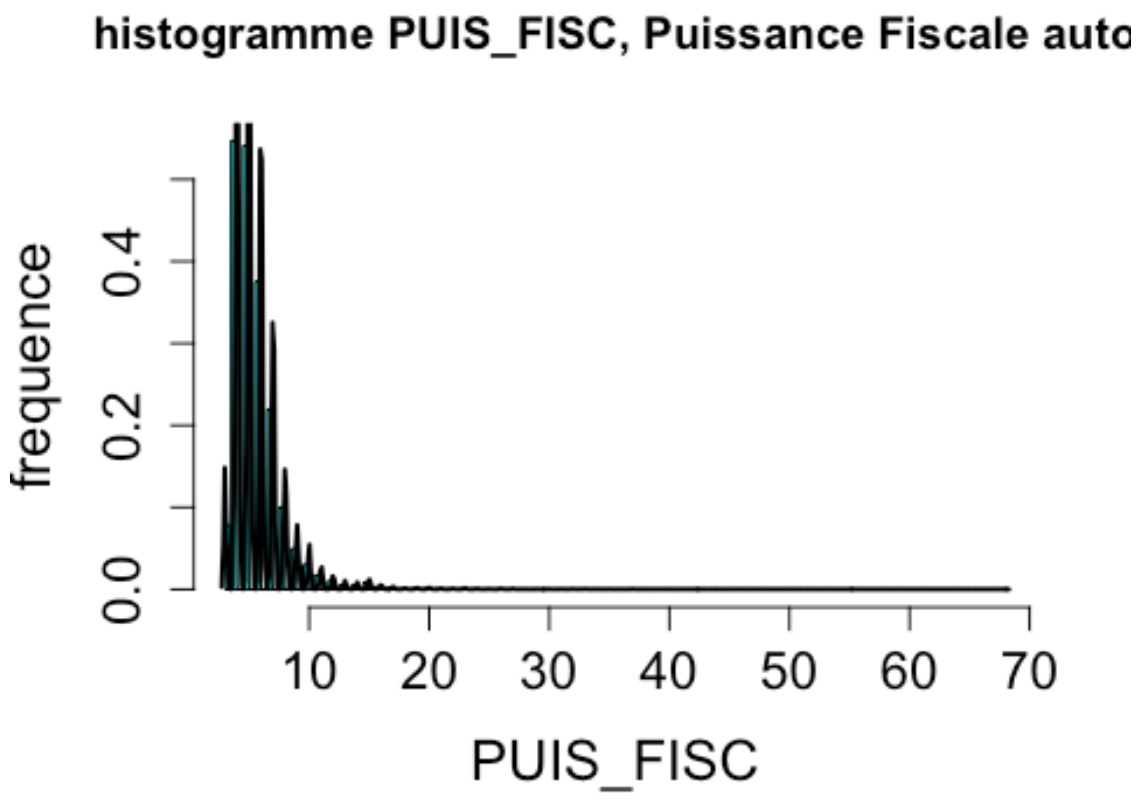



```r
skewness(Q$COEFF, na.rm = T) # [1] 0.28

## [1] 0.2895364

kurtosis(Q$COEFF,na.rm=T) # [1] 1.75

## [1] 1.753285

# distribution multi-modales (5)

# CHANGEMENT DE VARIABLE
dq2T[,"SqCOEFF"]=sqrt(dq2T[,"COEFF"])
QT= dq2T[,1:9]
X2Ta=QT$SqCOEFF

# apres
Titre="histogramme SqCOEFF, Bonus/Malus auto "
hist(X2Ta,breaks=100,col="cyan",prob=TRUE,cex.axis=1.4,cex.lab=1.5,ylab="frequence", xlab="SqCOEFF",main=
Titre)
lines(density(X2Ta), lwd = 2)
```

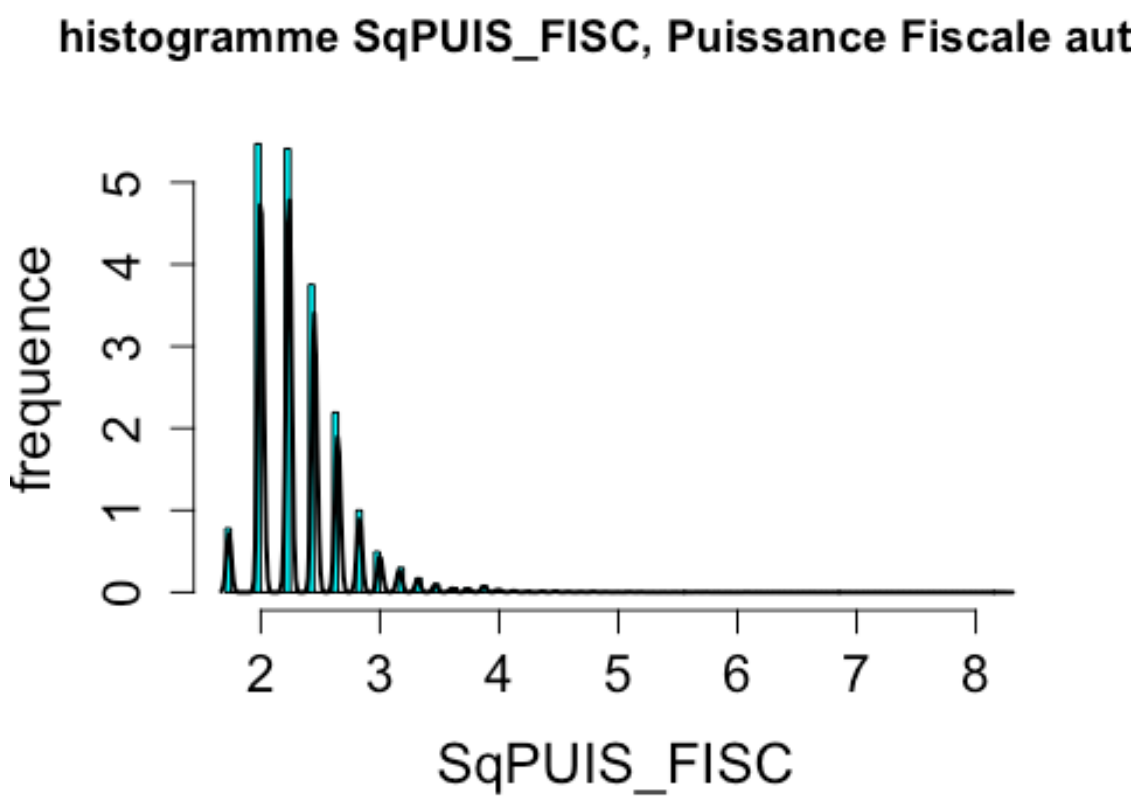



```
skewness(X2Ta, na.rm = T) # [1] 0.11

## [1] 0.1163656

kurtosis(X2Ta,na.rm=T) # [1] 1.61

## [1] 1.614179

# test de normalité des "Oui" avec SqCOEFF
dq2_ea<-dq2T[sample(1:nrow(dq2T),5000),]
select.ouia<-dq2_ea[,"trsf_dev"]=="O"
shapiro.test(dq2_ea[select.ouia,"SqCOEFF"])

##
##  Shapiro-Wilk normality test
##
## data:  dq2_ea[select.ouia, "SqCOEFF"]
## W = 0.87638, p-value < 2.2e-16

# p-value < 2.2e-16
# analyse : valeur p-value < 5%, on rejette la normalité de la var SqCOEFF pour les "Oui".

# test de normalité des "Non" avec SqCOEFF
dq2_ea<-dq2T[sample(1:nrow(dq2T),5000),]
select.ouia<-dq2_ea[,"trsf_dev"]=="N"
shapiro.test(dq2_ea[select.ouia,"SqCOEFF"])

##
##  Shapiro-Wilk normality test
##
## data:  dq2_ea[select.ouia, "SqCOEFF"]
## W = 0.88858, p-value < 2.2e-16

# p-value < 2.2e-16
# analyse : valeur p-value < 5%, on rejette la normalité de la var SqCOEFF pour les "Non".

#_____________________________________________________________
# conclusion : skewness SqCOEFF réduit, on conserve COEFF
#_____________________________________________________________
```

```r
# 7.3. PUISSANCE FISCAL

par(mfrow=c(1,1))
Titre="histogramme PUIS_FISC, Puissance Fiscale auto"
hist(X3,breaks=100,col="cyan",prob=TRUE,cex.axis=1.4,cex.lab=1.5,ylab="frequence", xlab="PUIS_FISC",main=
Titre)
lines(density(X3), lwd = 2)
```

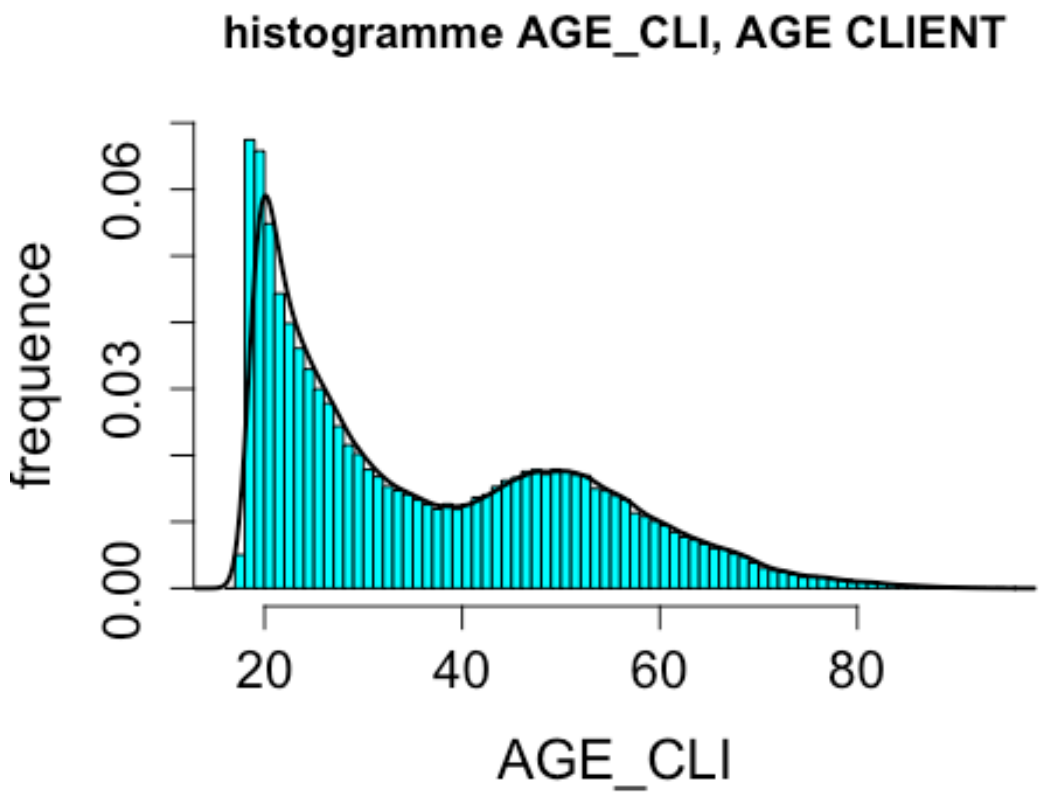



```r
skewness(Q$PUIS_FISC, na.rm = T) # [1] 4.23

## [1] 4.2392

kurtosis(Q$PUIS_FISC,na.rm=T) # [1] 42.21

## [1] 42.21556

# distribution discrète étalée à droite

# CHANGEMENT DE VARIABLE
dq2T[,"SqPUIS_FISC"]=sqrt(dq2T[,"PUIS_FISC"])
QT= dq2T[,1:10]
X3Ta=QT$SqPUIS_FISC

# apres
Titre="histogramme SqPUIS_FISC, Puissance Fiscale auto "
hist(X3Ta,breaks=100,col="cyan",prob=TRUE,cex.axis=1.4,cex.lab=1.5,ylab="frequence", xlab="SqPUIS_FISC",m
ain=Titre)
lines(density(X3Ta), lwd = 2)
```

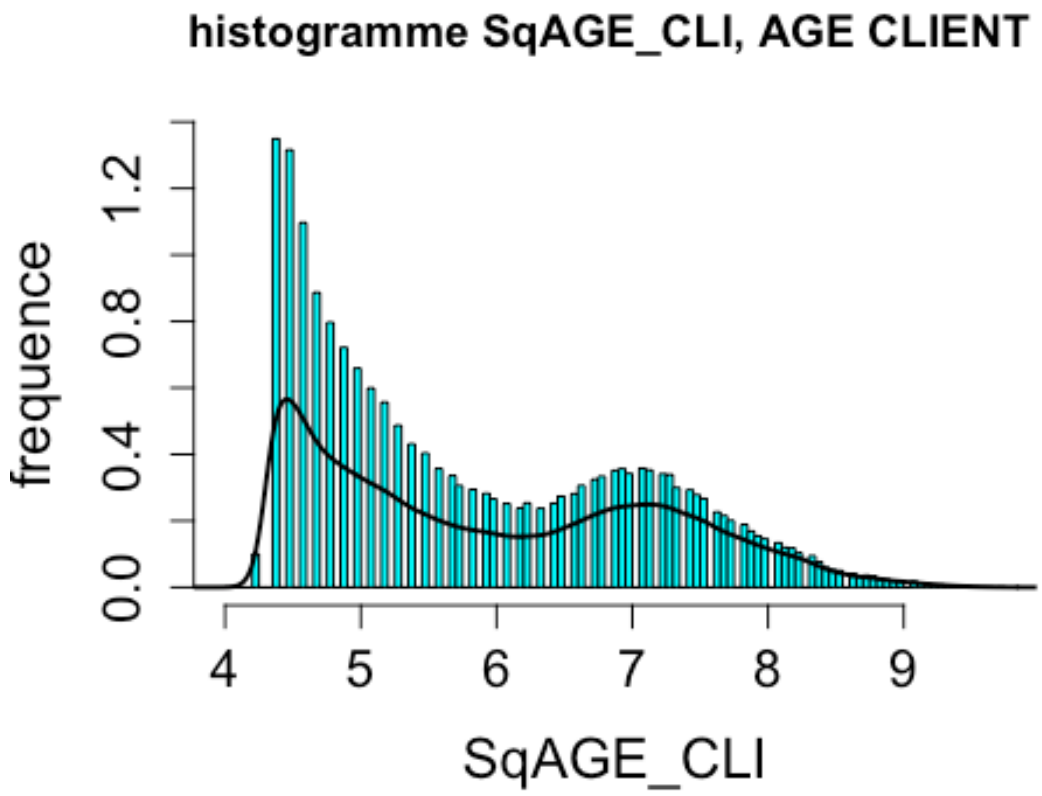



```
skewness(X3Ta, na.rm = T) # [1] 2.09

## [1] 2.096198

kurtosis(X3Ta,na.rm=T) # [1] 12.94

## [1] 12.94981

# test de normalité des "Oui" avec SqPUIS_FISC
dq2_ea<-dq2T[sample(1:nrow(dq2T),5000),]
select.ouia<-dq2_ea[,"trsf_dev"]=="O"
shapiro.test(dq2_ea[select.ouia,"SqPUIS_FISC"])

##
##  Shapiro-Wilk normality test
##
## data:  dq2_ea[select.ouia, "SqPUIS_FISC"]
## W = 0.83821, p-value < 2.2e-16

# p-value < 2.2e-16
# analyse : valeur p-value < 5%, on rejette la normalité de la var sqrt PUIS_FISC pour les "Oui".

# test de normalité des "Non" avec SqPUIS_FISC
dq2_ea<-dq2T[sample(1:nrow(dq2T),5000),]
select.ouia<-dq2_ea[,"trsf_dev"]=="N"
shapiro.test(dq2_ea[select.ouia,"SqPUIS_FISC"])

##
##  Shapiro-Wilk normality test
##
## data:  dq2_ea[select.ouia, "SqPUIS_FISC"]
## W = 0.85722, p-value < 2.2e-16

# p-value < 2.2e-16
# analyse : valeur p-value < 5%, on rejette la normalité de la var sqrt PUIS_FISC pour les "Non".

#_____________________________________________________________
# conclusion : skewness SqPUIS_FISC réduit, on conserve PUIS_FISC
#_____________________________________________________________
```

```r
# 7.4 AGE CLIENT

par(mfrow=c(1,1))

Titre="histogramme AGE_CLI, AGE CLIENT"
hist(X4,breaks=100,col="cyan",prob=TRUE,cex.axis=1.4,cex.lab=1.5,ylab="frequence", xlab="AGE_CLI",main=Ti
tre)
lines(density(X4), lwd = 2)
```

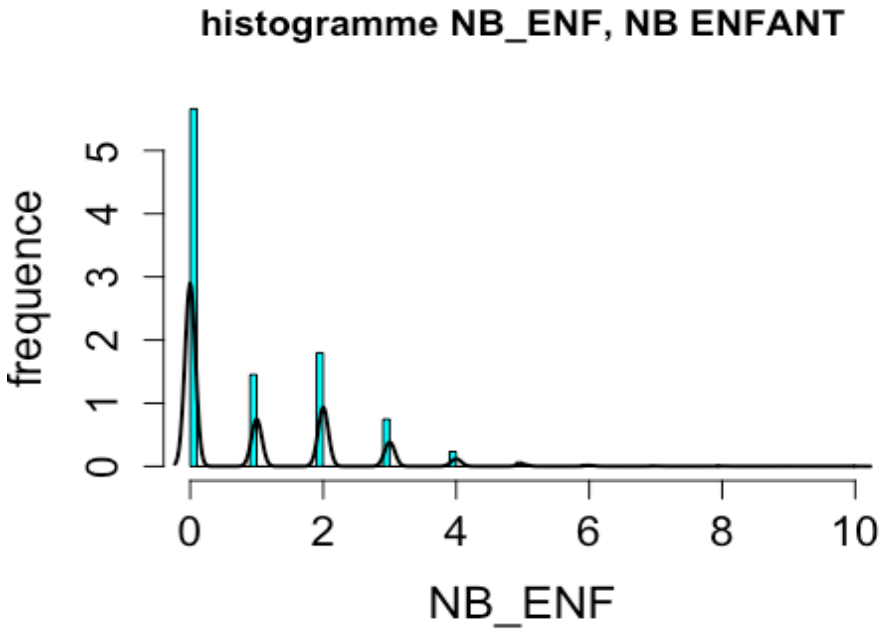



```r
skewness(Q$AGE_CLI, na.rm = T) # [1] 0.67

## [1] 0.6724585

kurtosis(Q$AGE_CLI,na.rm=T) # [1] 2.43

## [1] 2.434862

# distribution continue bi-modales et étalée à droite

# CHANGEMENT DE VARIABLE
dq2T[,"SqAGE_CLI"]=sqrt(dq2T[,"AGE_CLI"])
QT= dq2T[,1:11]
X4Ta=QT$SqAGE_CLI

# apres
Titre="histogramme SqAGE_CLI, AGE CLIENT "
hist(X4Ta,breaks=100,col="cyan",prob=TRUE,cex.axis=1.4,cex.lab=1.5,ylab="frequence", xlab="SqAGE_CLI",mai
n=Titre)
lines(density(X4Ta), lwd = 2)
```

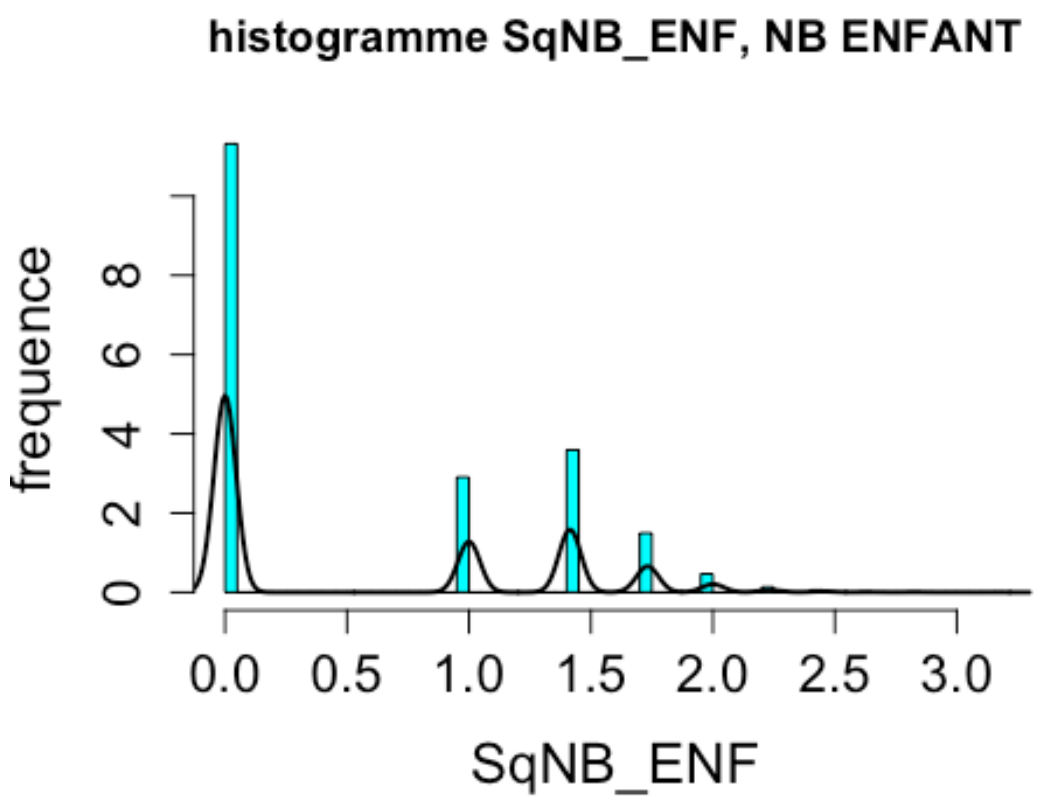



```
skewness(X4Ta, na.rm = T) # [1] 0.42

## [1] 0.4210727

kurtosis(X4Ta,na.rm=T) # [1] 1.96

## [1] 1.966202

# test de normalité des "Oui" avec SqAGE_CLI
dq2_ea<-dq2T[sample(1:nrow(dq2T),5000),]
select.ouia<-dq2_ea[,"trsf_dev"]=="O"
shapiro.test(dq2_ea[select.ouia,"SqAGE_CLI"])

##
##  Shapiro-Wilk normality test
##
## data:  dq2_ea[select.ouia, "SqAGE_CLI"]
## W = 0.89805, p-value < 2.2e-16

# p-value < 2.2e-16
# analyse : valeur p-value < 5%, on rejette la normalité de la var SqAGE_CLI pour les "Oui".

# test de normalité des "Non" avec SqAGE_CLI
dq2_ea<-dq2T[sample(1:nrow(dq2T),5000),]
select.ouia<-dq2_ea[,"trsf_dev"]=="N"
shapiro.test(dq2_ea[select.ouia,"SqAGE_CLI"])

##
##  Shapiro-Wilk normality test
##
## data:  dq2_ea[select.ouia, "SqAGE_CLI"]
## W = 0.93772, p-value < 2.2e-16

# p-value < 2.2e-16
# analyse : valeur p-value < 5%, on rejette la normalité de la var SqAGE_CLI pour les "Non".
#_____________________________________________________________
# conclusion : skewness SqAGE_CLI réduit, on conserve AGE_CLI
#_____________________________________________________________
```

```r
# 7.5 NOMBRE D ENFANT

par(mfrow=c(1,1))
Titre="histogramme NB_ENF, NB ENFANT"
hist(X5,breaks=100,col="cyan",prob=TRUE,cex.axis=1.4,cex.lab=1.5,ylab="frequence", xlab="NB_ENF",main=Tit
re)
lines(density(X5), lwd = 2)
```

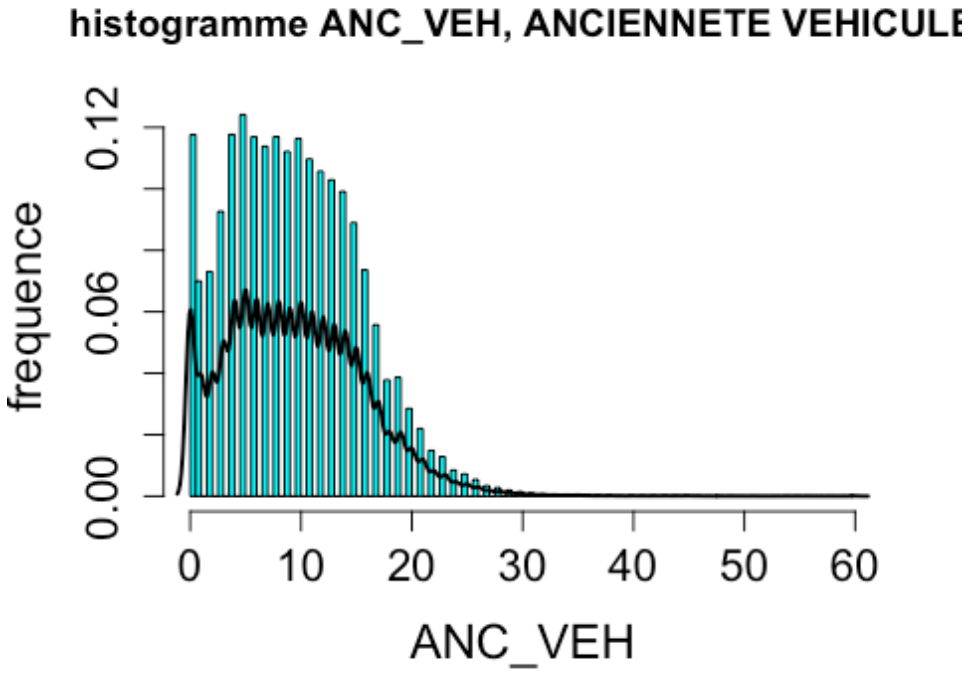



```r
skewness(Q$NB_ENF, na.rm = T) # [1] 1.45

## [1] 1.455243

kurtosis(Q$NB_ENF,na.rm=T) # [1] 5.42

## [1] 5.429995

# distribution discrète, étalement à droite

# CHANGEMENT DE VARIABLE
dq2T[,"SqNB_ENF"]=sqrt(dq2T[,"NB_ENF"])
QT= dq2T[,1:12]
X5Ta=QT$SqNB_ENF

# apres
Titre="histogramme SqNB_ENF, NB ENFANT"
hist(X5Ta,breaks=100,col="cyan",prob=TRUE,cex.axis=1.4,cex.lab=1.5,ylab="frequence", xlab="SqNB_ENF",main
=Titre)
lines(density(X5Ta), lwd = 2)
```

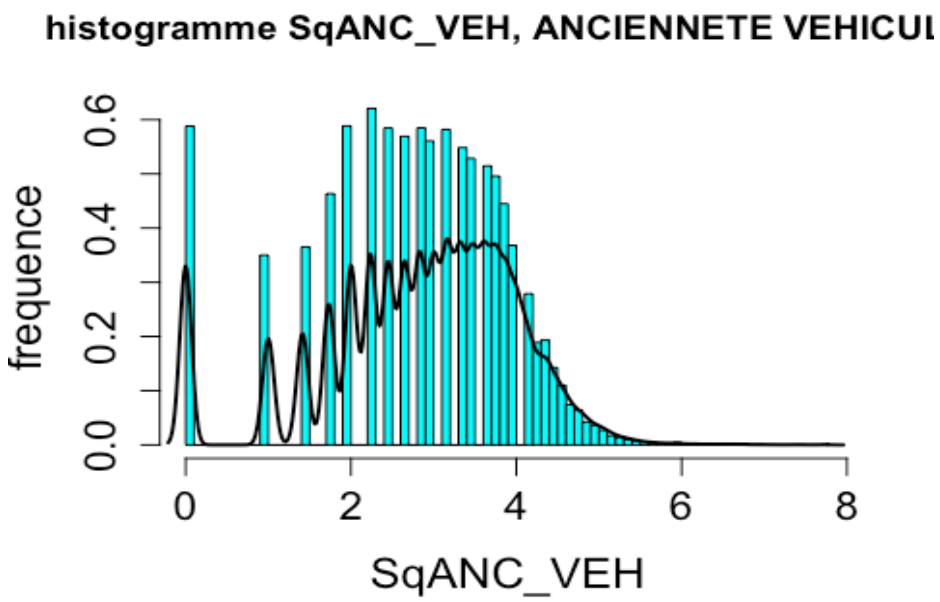



```
skewness(X5Ta, na.rm = T) # [1] 0.57

## [1] 0.5758119

kurtosis(X5Ta,na.rm=T) # [1] 1.77

## [1] 1.773607

# test de normalité des "Oui" avec SqNB_ENF
dq2_ea<-dq2T[sample(1:nrow(dq2T),5000),]
select.ouia<-dq2_ea[,"trsf_dev"]=="O"
shapiro.test(dq2_ea[select.ouia,"SqNB_ENF"])

##
##  Shapiro-Wilk normality test
##
## data:  dq2_ea[select.ouia, "SqNB_ENF"]
## W = 0.70224, p-value < 2.2e-16

# p-value < 2.2e-16
# analyse : valeur p-value < 5%, on rejette la normalité de la var SqNB_ENF pour les "Oui".

# test de normalité des "Non" avec SqNB_ENF
dq2_ea<-dq2T[sample(1:nrow(dq2T),5000),]
select.ouia<-dq2_ea[,"trsf_dev"]=="N"
shapiro.test(dq2_ea[select.ouia,"SqNB_ENF"])

##
##  Shapiro-Wilk normality test
##
## data:  dq2_ea[select.ouia, "SqNB_ENF"]
## W = 0.75816, p-value < 2.2e-16

# p-value < 2.2e-16
# analyse : valeur p-value < 5%, on rejette la normalité de la var SqNB_ENF pour les "Non".
#_____________________________________________________________
# conclusion : skewness SqNB_ENF réduit, on conserve NB_ENF
#_____________________________________________________________
```

```r
# 7.6 ANCIENNETE VEHICULE

par(mfrow=c(1,1))
Titre="histogramme ANC_VEH, ANCIENNETE VEHICULE"
hist(X6,breaks=100,col="cyan",prob=TRUE,cex.axis=1.4,cex.lab=1.5,ylab="frequence", xlab="ANC_VEH",main=Ti
tre)
lines(density(X6), lwd = 2)
```




```r
skewness(Q$ANC_VEH, na.rm = T) #[1] 0.75

## [1] 0.7512182

kurtosis(Q$ANC_VEH,na.rm=T) #[1] 4.61

## [1] 4.611772

# distribution continue étalée à droite


# CHANGEMENT DE VARIABLE
dq2T[,"SqANC_VEH"]=sqrt(dq2T[,"ANC_VEH"])
QT=dq2T[,1:13]
X6Ta=QT$SqANC_VEH

# apres
Titre="histogramme SqANC_VEH, ANCIENNETE VEHICULE"
hist(X6Ta,breaks=100,col="cyan",prob=TRUE,cex.axis=1.4,cex.lab=1.5,ylab="frequence", xlab="SqANC_VEH",mai
n=Titre)
lines(density(X6Ta), lwd = 2)
```




```
skewness(X6Ta, na.rm = T) # [1] -0.57

## [1] -0.5783773

kurtosis(X6Ta,na.rm=T) # [1] 3.30

## [1] 3.306019

# test de normalité des "Oui"
dq2_ea<-dq2T[sample(1:nrow(dq2T),5000),]
select.ouia<-dq2_ea[,"trsf_dev"]=="O"
shapiro.test(dq2_ea[select.ouia,"SqANC_VEH"])

##
##  Shapiro-Wilk normality test
##
## data:  dq2_ea[select.ouia, "SqANC_VEH"]
## W = 0.95525, p-value < 2.2e-16

# p-value < 2.2e-16
# analyse : valeur p-value < 5%, on rejette la normalité de la var SqANC_VEH pour les "Oui".

# test de normalité des "Non" avec C_AN_TTC
dq2_ea<-dq2T[sample(1:nrow(dq2T),5000),]
select.ouia<-dq2_ea[,"trsf_dev"]=="N"
shapiro.test(dq2_ea[select.ouia,"SqANC_VEH"])

##
##  Shapiro-Wilk normality test
##
## data:  dq2_ea[select.ouia, "SqANC_VEH"]
## W = 0.96279, p-value < 2.2e-16

# p-value < 2.2e-16
# analyse : valeur p-value < 5%, on rejette la normalité de la var SqANC_VEH pour les "Non".
#_____________________________________________________________
# conclusion :  on conserve ANC_VEH
#_____________________________________________________________
#_____________________________________________________________
# conclusion NORMALISATION des 6 Xi :
# on accepte la var transformée LogC_AN_TTC
# on conserve les 5 var COEFF, PUIS_FISC, AGE_CLI, NB_ENF, ANC_VEH
#_____________________________________________________________
```

```r
# 7.7 dataset final post normalisation

# dataset final
dq2norm = dq2T[,c(8,2:6,7)]

# sanity check
summary(dq2norm)

##   LogC_AN_TTC        COEFF          PUIS_FISC        AGE_CLI
##  Min.   :4.174   Min.   : 35.00   Min.   : 3.00   Min.   :16.00
##  1st Qu.:6.074   1st Qu.: 40.00   1st Qu.: 4.00   1st Qu.:23.00
##  Median :6.454   Median : 65.00   Median : 5.00   Median :32.00
##  Mean   :6.444   Mean   : 66.26   Mean   : 5.66   Mean   :36.88
##  3rd Qu.:6.819   3rd Qu.: 95.00   3rd Qu.: 6.00   3rd Qu.:49.00
##  Max.   :9.498   Max.   :225.00   Max.   :68.00   Max.   :95.00
##      NB_ENF           ANC_VEH       trsf_dev
##  Min.   : 0.0000   Min.   : 0.000   N:450751
##  1st Qu.: 0.0000   1st Qu.: 5.000   O:164177
##  Median : 0.0000   Median : 9.000
##  Mean   : 0.8839   Mean   : 9.495
##  3rd Qu.: 2.0000   3rd Qu.:14.000
##  Max.   :10.0000   Max.   :60.000

# 8. Centrage et reduction des Xi quantitatives

# CR (0,1)
dq2normCR=as.data.frame(scale(dq2norm[,1:6]))

# sanity check

summary(dq2normCR)

##   LogC_AN_TTC           COEFF            PUIS_FISC          AGE_CLI
##  Min.   :-4.15148   Min.   :-1.26197   Min.   :-1.1648   Min.   :-1.3301
##  1st Qu.:-0.67751   1st Qu.:-1.06014   1st Qu.:-0.7270   1st Qu.:-0.8841
##  Median : 0.01818   Median :-0.05098   Median :-0.2892   Median :-0.3107
##  Mean   : 0.00000   Mean   : 0.00000   Mean   : 0.0000   Mean   : 0.0000
##  3rd Qu.: 0.68606   3rd Qu.: 1.16001   3rd Qu.: 0.1487   3rd Qu.: 0.7725
##  Max.   : 5.58588   Max.   : 6.40764   Max.   :27.2936   Max.   : 3.7033
##      NB_ENF           ANC_VEH
##  Min.   :-0.7243   Min.   :-1.53296
##  1st Qu.:-0.7243   1st Qu.:-0.72575
##  Median :-0.7243   Median :-0.07999
##  Mean   : 0.0000   Mean   : 0.00000
##  3rd Qu.: 0.9145   3rd Qu.: 0.72722
##  Max.   : 7.4697   Max.   : 8.15350

sd(dq2normCR$LogC_AN_TTC) # [1] 1

## [1] 1

sd(dq2normCR$COEFF)       # [1] 1

## [1] 1

sd(dq2normCR$PUIS_FISC)   # [1] 1

## [1] 1

sd(dq2normCR$AGE_CLI)     # [1] 1

## [1] 1

sd(dq2normCR$NB_ENF)      # [1] 1

## [1] 1

sd(dq2normCR$ANC_VEH)     # [1] 1

## [1] 1
```

```r
# 9. Formatage dataset - final

dq2norm_y = dq2normCR
dq2norm_y$trsf_dev = dq2norm$trsf_dev

#========================
# dataset final
# dq2norm_y

str(dq2norm_y)

## 'data.frame':    614928 obs. of  7 variables:
##  $ LogC_AN_TTC: num  1.742 0.982 0.982 1.249 1.742 ...
##  $ COEFF      : num  1.36 1.36 1.36 1.36 1.36 ...
##  $ PUIS_FISC  : num  -0.289 -0.289 -0.289 -0.289 -0.289 ...
##  $ AGE_CLI    : num  -0.757 -0.757 -0.757 -0.757 -0.757 ...
##  $ NB_ENF     : num  -0.724 -0.724 -0.724 -0.724 -0.724 ...
##  $ ANC_VEH    : num  -0.08 -0.08 -0.08 -0.08 -0.08 ...
##  $ trsf_dev   : Factor w/ 2 levels "N","O": 1 1 1 1 1 2 2 2 2 1 ...

#========================

# sanity check
Pourcentage_NA = colMeans(is.na(dq2norm_y))
Pourcentage_NA = Pourcentage_NA * 100
Pourcentage_NA

## LogC_AN_TTC       COEFF   PUIS_FISC     AGE_CLI      NB_ENF     ANC_VEH
##           0           0           0           0           0           0
##    trsf_dev
##           0

summary(Pourcentage_NA)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       0       0       0       0       0       0

mean(Pourcentage_NA)

## [1] 0

# 10. Construction d un echantillon stratifie proportionnel

# objectif, ne pas travailler sur le dataset total (de 614928 lignes) mais prelever un echantillon de 615
0 lignes par la méthode de sondage,
# but traitement numérique : les calculs tiennent en RAM ordinateur et obtenir un temps de calcul réduit(
ie pour les classifieurs SVM).

# 10.1 méthode de prelevement

# selon JM POGGI, document arbres CART et Forets aleatoires ie refrence hal0138765v2
# construire un sondage stratifie proportionnel sur la variable d'interet,
# on preleve 1 pourcent de l'effectif total 614929 lignes, ie 6150 lignes,
# la méthode sondage stratifie optimal non nécessaire ici, car les explicatives Xi sont centrees-reduites
(0,1),
# prerequis : verifier les metrics moyenne et ecart-type sont bien respectees sur les sous populations No
n / oui,


dq2norm_y=dq2norm_y[sample(1:nrow(dq2norm_y),6150),]

# sanity check
```

```r
# 10.2 sur la variable d interet

# en nombre etiquette NON / OUI
summary(dq2norm_y$trsf_dev)

##    N    O
## 4510 1640

# N    O
# 4533 1617

# en POURCENTAGE : poids de l'etiquette NON / OUI
PoidsY6150=(table(dq2norm_y$trsf_dev)/nrow(dq2norm_y)*100)
PoidsY6150

##
##        N        O
## 73.33333 26.66667

# N        O
# 72.95935 27.04065
Ratio_OUI_NON_6150 = PoidsY6150[[2]]/PoidsY6150[[1]]*100
Ratio_OUI_NON_6150 # [1] 37.06

## [1] 36.36364

# comparaison avec les valeurs de la Population Totale
# POURCENTAGE : poids de l'etiquette NON / OUI
PoidsY=(table(dv17$trsf_dev)/nrow(dv17)*100)
PoidsY

##
##        N        O
## 73.31072 26.68928

#   N        O
#   73.31072 26.68928
#   dataset avec classes déséquilibrées (les NON majoritaires)

Ratio_OUI_NON = PoidsY[[2]]/PoidsY[[1]]*100
Ratio_OUI_NON # [1] 36.40

## [1] 36.4057

delta=(Ratio_OUI_NON-Ratio_OUI_NON_6150)
delta # [1] 0.73%

## [1] 0.04206557

# conclusions sur Yi :
# les conditions de proportionnalité de la var Yi sont respectées lors de la réduction
# dataset total à dataset sondage.

# 10.3 sur les variables explicatives
# COTISATION
tapply(dq2norm_y[,"LogC_AN_TTC"], dq2norm_y[,"trsf_dev"], mean, na.rm=TRUE)

##           N           O
##  0.03198455 -0.11955950

tapply(dq2norm_y[,"LogC_AN_TTC"], dq2norm_y[,"trsf_dev"], sd, na.rm=TRUE)

##        N        O
## 1.032652 0.915605

tapply(dq2norm_y[,"LogC_AN_TTC"], dq2norm_y[,"trsf_dev"], var, na.rm=TRUE)

##         N         O
## 1.0663696 0.8383326
```

```r
tapply(dq2norm_y[,"LogC_AN_TTC"], dq2norm_y[,"trsf_dev"], quantile, na.rm=TRUE)

## $N
##          0%         25%         50%         75%        100%
## -3.06083225 -0.67751202  0.05523066  0.72168035  4.19400945
##
## $O
##          0%         25%         50%         75%        100%
## -3.06083225 -0.72443469 -0.08851515  0.57324517  2.31916321

tapply(dq2norm_y[,"LogC_AN_TTC"], dq2norm_y[,"trsf_dev"], skewness, na.rm=TRUE)

##           N           O
## -0.04876516 -0.19778489

tapply(dq2norm_y[,"LogC_AN_TTC"], dq2norm_y[,"trsf_dev"], kurtosis, na.rm=TRUE)

##        N        O
## 2.894999 2.730105

tapply(dq2norm_y[,"LogC_AN_TTC"], dq2norm_y[,"trsf_dev"], density, na.rm=TRUE)

## $N
##
## Call:
##  density.default(x = X[[i]], na.rm = TRUE)
##
## Data: X[[i]] (4510 obs.);    Bandwidth 'bw' = 0.1727
##
##        x                 y
##  Min.   :-3.5790   Min.   :0.000007
##  1st Qu.:-1.5062   1st Qu.:0.002591
##  Median : 0.5666   Median :0.058453
##  Mean   : 0.5666   Mean   :0.120492
##  3rd Qu.: 2.6394   3rd Qu.:0.236853
##  Max.   : 4.7122   Max.   :0.379913
##
## $O
##
## Call:
##  density.default(x = X[[i]], na.rm = TRUE)
##
## Data: X[[i]] (1640 obs.);    Bandwidth 'bw' = 0.1875
##
##        x                 y
##  Min.   :-3.6233   Min.   :0.0000166
##  1st Qu.:-1.9971   1st Qu.:0.0106303
##  Median :-0.3708   Median :0.0986044
##  Mean   :-0.3708   Mean   :0.1535788
##  3rd Qu.: 1.2554   3rd Qu.:0.3061635
##  Max.   : 2.8816   Max.   :0.4055143

# analyse :
# moy.  proche nulle
# ecart-type proche un

# COEFF BONUS MALUS
tapply(dq2norm_y[,"COEFF"], dq2norm_y[,"trsf_dev"], mean, na.rm=TRUE)

##            N            O
##  0.009310166 -0.049724569

tapply(dq2norm_y[,"COEFF"], dq2norm_y[,"trsf_dev"], sd, na.rm=TRUE)

##         N         O
## 1.0076193 0.9851986

tapply(dq2norm_y[,"COEFF"], dq2norm_y[,"trsf_dev"], var, na.rm=TRUE)

##         N         O
## 1.0152967 0.9706163
```

```r
tapply(dq2norm_y[,"COEFF"], dq2norm_y[,"trsf_dev"], quantile, na.rm=TRUE)

## $N
##          0%         25%         50%         75%        100%
## -1.26197174 -1.06013976 -0.05097987  1.16001201  5.19665160
##
## $O
##          0%         25%         50%         75%        100%
## -1.26197174 -1.06013976 -0.05097987  0.95818003  2.37100389

tapply(dq2norm_y[,"COEFF"], dq2norm_y[,"trsf_dev"], skewness, na.rm=TRUE)

##         N         O
## 0.3115853 0.2982448

tapply(dq2norm_y[,"COEFF"], dq2norm_y[,"trsf_dev"], kurtosis, na.rm=TRUE)

##        N        O
## 1.864875 1.715510

tapply(dq2norm_y[,"COEFF"], dq2norm_y[,"trsf_dev"], density, na.rm=TRUE)

## $N
##
## Call:
##  density.default(x = X[[i]], na.rm = TRUE)
##
## Data: X[[i]] (4510 obs.);    Bandwidth 'bw' = 0.1685
##
##        x                  y
##  Min.   :-1.76759   Min.   :0.0000000
##  1st Qu.: 0.09987   1st Qu.:0.0005233
##  Median : 1.96734   Median :0.0114409
##  Mean   : 1.96734   Mean   :0.1337243
##  3rd Qu.: 3.83481   3rd Qu.:0.2318659
##  Max.   : 5.70227   Max.   :0.6449515
##
## $O
##
## Call:
##  density.default(x = X[[i]], na.rm = TRUE)
##
## Data: X[[i]] (1640 obs.);    Bandwidth 'bw' = 0.2017
##
##        x                 y
##  Min.   :-1.8672   Min.   :0.0000545
##  1st Qu.:-0.6563   1st Qu.:0.0195199
##  Median : 0.5545   Median :0.1774616
##  Mean   : 0.5545   Mean   :0.2062195
##  3rd Qu.: 1.7654   3rd Qu.:0.3588232
##  Max.   : 2.9762   Max.   :0.5616269

# idem

# PUISSANCE FISCAL
tapply(dq2norm_y[,"PUIS_FISC"], dq2norm_y[,"trsf_dev"], mean, na.rm=TRUE)

##           N           O
##  0.04411161 -0.12657560

tapply(dq2norm_y[,"PUIS_FISC"], dq2norm_y[,"trsf_dev"], sd, na.rm=TRUE)

##         N         O
## 1.0049436 0.8715495

tapply(dq2norm_y[,"PUIS_FISC"], dq2norm_y[,"trsf_dev"], var, na.rm=TRUE)

##         N         O
## 1.0099116 0.7595986
```

```r
tapply(dq2norm_y[,"PUIS_FISC"], dq2norm_y[,"trsf_dev"], quantile, na.rm=TRUE)

## $N
##         0%        25%        50%        75%       100%
## -1.1647994 -0.7269781 -0.2891568  0.1486645 15.0345888
##
## $O
##         0%        25%        50%        75%       100%
## -1.1647994 -0.7269781 -0.2891568  0.1486645 11.9698397

tapply(dq2norm_y[,"PUIS_FISC"], dq2norm_y[,"trsf_dev"], skewness, na.rm=TRUE)

##        N        O
## 4.159423 4.191676

tapply(dq2norm_y[,"PUIS_FISC"], dq2norm_y[,"trsf_dev"], kurtosis, na.rm=TRUE)

##        N        O
## 39.85803 38.91834

tapply(dq2norm_y[,"PUIS_FISC"], dq2norm_y[,"trsf_dev"], density, na.rm=TRUE)

## $N
##
## Call:
##  density.default(x = X[[i]], na.rm = TRUE)
##
## Data: X[[i]] (4510 obs.);    Bandwidth 'bw' = 0.1093
##
##        x                y
##  Min.   :-1.493   Min.   :0.0000000
##  1st Qu.: 2.721   1st Qu.:0.0000082
##  Median : 6.935   Median :0.0007702
##  Mean   : 6.935   Mean   :0.0592527
##  3rd Qu.:11.149   3rd Qu.:0.0135187
##  Max.   :15.362   Max.   :0.9678135
##
## $O
##
## Call:
##  density.default(x = X[[i]], na.rm = TRUE)
##
## Data: X[[i]] (1640 obs.);    Bandwidth 'bw' = 0.1338
##
##        x                y
##  Min.   :-1.566   Min.   :0.000000
##  1st Qu.: 1.918   1st Qu.:0.000000
##  Median : 5.403   Median :0.001625
##  Mean   : 5.403   Mean   :0.071704
##  3rd Qu.: 8.887   3rd Qu.:0.014618
##  Max.   :12.371   Max.   :1.014453

# idem

# AGE CLIENT
tapply(dq2norm_y[,"AGE_CLI"], dq2norm_y[,"trsf_dev"], mean, na.rm=TRUE)

##           N           O
##  0.05735243 -0.13832416

tapply(dq2norm_y[,"AGE_CLI"], dq2norm_y[,"trsf_dev"], sd, na.rm=TRUE)

##         N         O
## 1.0056809 0.9749114

tapply(dq2norm_y[,"AGE_CLI"], dq2norm_y[,"trsf_dev"], var, na.rm=TRUE)

##         N         O
## 1.0113940 0.9504522
```

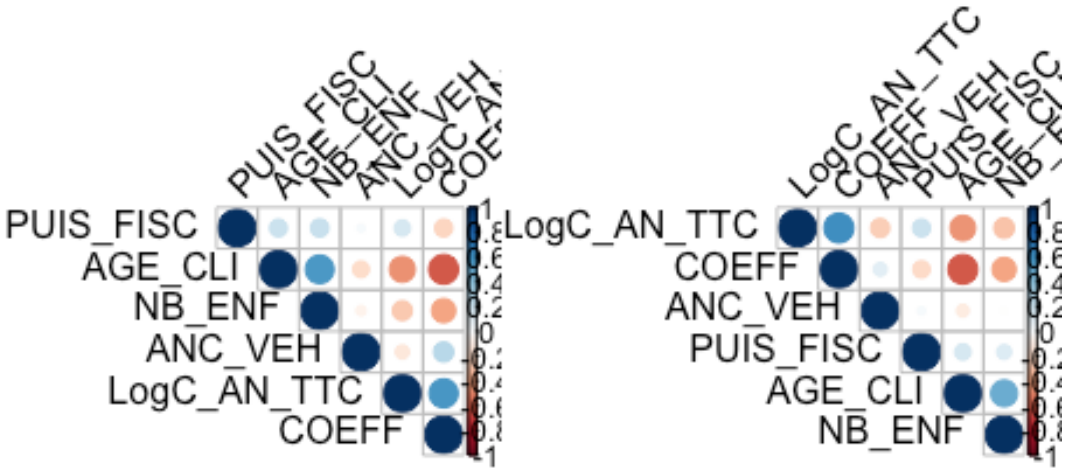
```r
tapply(dq2norm_y[,"AGE_CLI"], dq2norm_y[,"trsf_dev"], quantile, na.rm=TRUE)

## $N
##         0%        25%        50%        75%       100%
## -1.2026573 -0.8840878 -0.1832347  0.8361879  3.5758863
##
## $O
##         0%        25%        50%        75%       100%
## -1.2026573 -0.9478017 -0.5655182  0.6450462  3.5758863

tapply(dq2norm_y[,"AGE_CLI"], dq2norm_y[,"trsf_dev"], skewness, na.rm=TRUE)

##         N         O
## 0.5816806 0.8621316

tapply(dq2norm_y[,"AGE_CLI"], dq2norm_y[,"trsf_dev"], kurtosis, na.rm=TRUE)

##        N        O
## 2.298630 2.651131

tapply(dq2norm_y[,"AGE_CLI"], dq2norm_y[,"trsf_dev"], density, na.rm=TRUE)

## $N
##
## Call:
##  density.default(x = X[[i]], na.rm = TRUE)
##
## Data: X[[i]] (4510 obs.);    Bandwidth 'bw' = 0.1682
##
##        x                 y
##  Min.   :-1.7073   Min.   :0.000006
##  1st Qu.:-0.2603   1st Qu.:0.016257
##  Median : 1.1866   Median :0.158324
##  Mean   : 1.1866   Mean   :0.172602
##  3rd Qu.: 2.6336   3rd Qu.:0.266058
##  Max.   : 4.0805   Max.   :0.614470
##
## $O
##
## Call:
##  density.default(x = X[[i]], na.rm = TRUE)
##
## Data: X[[i]] (1640 obs.);    Bandwidth 'bw' = 0.1996
##
##        x                 y
##  Min.   :-1.8016   Min.   :0.0000138
##  1st Qu.:-0.3075   1st Qu.:0.0080580
##  Median : 1.1866   Median :0.1250324
##  Mean   : 1.1866   Mean   :0.1671548
##  3rd Qu.: 2.6807   3rd Qu.:0.2214371
##  Max.   : 4.1748   Max.   :0.7698807

# idem

# NOMBRE D ENFANT
tapply(dq2norm_y[,"NB_ENF"], dq2norm_y[,"trsf_dev"], mean, na.rm=TRUE)

##           N           O
##  0.04023134 -0.08376901

tapply(dq2norm_y[,"NB_ENF"], dq2norm_y[,"trsf_dev"], sd, na.rm=TRUE)

##         N         O
## 1.0276240 0.9761089

tapply(dq2norm_y[,"NB_ENF"], dq2norm_y[,"trsf_dev"], var, na.rm=TRUE)

##         N         O
## 1.0560110 0.9527886
```

```r
tapply(dq2norm_y[,"NB_ENF"], dq2norm_y[,"trsf_dev"], quantile, na.rm=TRUE)

## $N
##         0%        25%        50%        75%       100%
## -0.7243012 -0.7243012 -0.7243012  0.9145019  7.4697145
##
## $O
##          0%         25%         50%         75%        100%
## -0.72430122 -0.72430122 -0.72430122  0.09510035  7.46971452

tapply(dq2norm_y[,"NB_ENF"], dq2norm_y[,"trsf_dev"], skewness, na.rm=TRUE)

##        N        O
## 1.440697 1.837020

tapply(dq2norm_y[,"NB_ENF"], dq2norm_y[,"trsf_dev"], kurtosis, na.rm=TRUE)

##        N        O
## 5.447944 7.850222

tapply(dq2norm_y[,"NB_ENF"], dq2norm_y[,"trsf_dev"], density, na.rm=TRUE)

## $N
##
## Call:
##  density.default(x = X[[i]], na.rm = TRUE)
##
## Data: X[[i]] (4510 obs.);    Bandwidth 'bw' = 0.1719
##
##        x                y
##  Min.   :-1.240   Min.   :0.000000
##  1st Qu.: 1.066   1st Qu.:0.000629
##  Median : 3.373   Median :0.008547
##  Mean   : 3.373   Mean   :0.108223
##  3rd Qu.: 5.679   3rd Qu.:0.100124
##  Max.   : 7.985   Max.   :1.263511
##
## $O
##
## Call:
##  density.default(x = X[[i]], na.rm = TRUE)
##
## Data: X[[i]] (1640 obs.);    Bandwidth 'bw' = 0.1252
##
##        x                y
##  Min.   :-1.100   Min.   :0.000000
##  1st Qu.: 1.136   1st Qu.:0.000447
##  Median : 3.373   Median :0.003855
##  Mean   : 3.373   Mean   :0.111607
##  3rd Qu.: 5.609   3rd Qu.:0.054141
##  Max.   : 7.845   Max.   :1.932566

# idem

# ANCIENNETE VEHICULE
tapply(dq2norm_y[,"ANC_VEH"], dq2norm_y[,"trsf_dev"], mean, na.rm=TRUE)

##            N            O
## -0.006640756  0.051626526

tapply(dq2norm_y[,"ANC_VEH"], dq2norm_y[,"trsf_dev"], sd, na.rm=TRUE)

##         N         O
## 1.0370565 0.9925247

tapply(dq2norm_y[,"ANC_VEH"], dq2norm_y[,"trsf_dev"], var, na.rm=TRUE)

##         N         O
## 1.0754862 0.9851053
```

```r
tapply(dq2norm_y[,"ANC_VEH"], dq2norm_y[,"trsf_dev"], quantile, na.rm=TRUE)

## $N
##          0%         25%         50%         75%        100%
## -1.53295538 -0.72575083 -0.07998719  0.72721736  8.15349919
##
## $O
##          0%         25%         50%         75%        100%
## -1.53295538 -0.72575083  0.08145372  0.72721736  5.40900373

tapply(dq2norm_y[,"ANC_VEH"], dq2norm_y[,"trsf_dev"], skewness, na.rm=TRUE)

##         N         O
## 0.9219607 0.4984167

tapply(dq2norm_y[,"ANC_VEH"], dq2norm_y[,"trsf_dev"], kurtosis, na.rm=TRUE)

##        N        O
## 5.687345 3.400454

tapply(dq2norm_y[,"ANC_VEH"], dq2norm_y[,"trsf_dev"], density, na.rm=TRUE)

## $N
##
## Call:
##  density.default(x = X[[i]], na.rm = TRUE)
##
## Data: X[[i]] (4510 obs.);    Bandwidth 'bw' = 0.1735
##
##        x                 y
##  Min.   :-2.0533   Min.   :0.0000000
##  1st Qu.: 0.6285   1st Qu.:0.0005448
##  Median : 3.3103   Median :0.0052299
##  Mean   : 3.3103   Mean   :0.0931223
##  3rd Qu.: 5.9921   3rd Qu.:0.2034370
##  Max.   : 8.6739   Max.   :0.3619404
##
## $O
##
## Call:
##  density.default(x = X[[i]], na.rm = TRUE)
##
## Data: X[[i]] (1640 obs.);    Bandwidth 'bw' = 0.2032
##
##        x                 y
##  Min.   :-2.1427   Min.   :0.0000002
##  1st Qu.:-0.1023   1st Qu.:0.0015840
##  Median : 1.9380   Median :0.0322726
##  Mean   : 1.9380   Mean   :0.1223999
##  3rd Qu.: 3.9784   3rd Qu.:0.2752342
##  Max.   : 6.0187   Max.   :0.3482492
```

```r
# 11. matrice var-cov / correlation sur Xi normalise

# matrice de variance_covariance
OUI=subset(dq2norm_y,trsf_dev=="O")
ZOUI=OUI[,1:6]
var(ZOUI)

##             LogC_AN_TTC      COEFF  PUIS_FISC    AGE_CLI      NB_ENF
## LogC_AN_TTC   0.7666614  0.4994631  0.1430350 -0.3873439 -0.22488399
## COEFF         0.4994631  0.9445906 -0.1986361 -0.5782529 -0.37611623
## PUIS_FISC     0.1430350 -0.1986361  0.9118192  0.1933030  0.20586912
## AGE_CLI      -0.3873439 -0.5782529  0.1933030  0.9273076  0.53286990
## NB_ENF       -0.2248840 -0.3761162  0.2058691  0.5328699  0.94196442
## ANC_VEH      -0.1076536  0.2528968  0.0299503 -0.1673134 -0.05848213
##                 ANC_VEH
## LogC_AN_TTC -0.10765360
## COEFF        0.25289679
## PUIS_FISC    0.02995030
## AGE_CLI     -0.16731337
## NB_ENF      -0.05848213
## ANC_VEH      0.91947773

NON=subset(dq2norm_y,trsf_dev=="N")
ZNON=NON[,1:6]
var(ZNON)

##             LogC_AN_TTC      COEFF   PUIS_FISC    AGE_CLI       NB_ENF
## LogC_AN_TTC   1.0436978  0.6375884  0.22337676 -0.4595053 -0.283316591
## COEFF         0.6375884  1.0186477 -0.20695742 -0.6204232 -0.391032938
## PUIS_FISC     0.2233768 -0.2069574  1.07647090  0.1833634  0.144382918
## AGE_CLI      -0.4595053 -0.6204232  0.18336342  1.0110409  0.491747719
## NB_ENF       -0.2833166 -0.3910329  0.14438292  0.4917477  0.965755819
## ANC_VEH      -0.2502493  0.1220908  0.04047481 -0.1051487 -0.008641134
##                 ANC_VEH
## LogC_AN_TTC -0.250249316
## COEFF        0.122090841
## PUIS_FISC    0.040474809
## AGE_CLI     -0.105148665
## NB_ENF      -0.008641134
## ANC_VEH      0.995886624

par(mfrow=c(1,2))
corrplot(cor(ZOUI),type="upper", order="hclust",tl.col="black", tl.srt=45, main="matrice corrélation : OUI")
corrplot(cor(ZNON),type="upper", order="hclust",tl.col="black", tl.srt=45, main="matrice corrélation : NON")
```

matrice correlation : OUI matrice correlation : NON




```
# conclusion sur les Xi :
# le passage dataset total à dataset sondage, garantit le centrage réduction des Xi.


# CONCLUSION
# au regard des résultats vérifiés au paragraphe 10.1, 10.2 et 10.3
# on obtient le dataset sondage,
dim(dq2norm_y) # [1] 6150 lignes x 7 colonnes

## [1] 6150    7

toc <-  Sys.time() -tic
print(toc)

## Time difference of 3.257581 mins

#*****************************************************************************************************
*****************************************************************************************
#*****************************************************************************************************
*****************************************************************************************
```

10.7.2. Le classifieur forêts aléatoires « Random Forest ».

RF_prepada.R
lepicaut
Thu Oct 5 15:54:41 2017

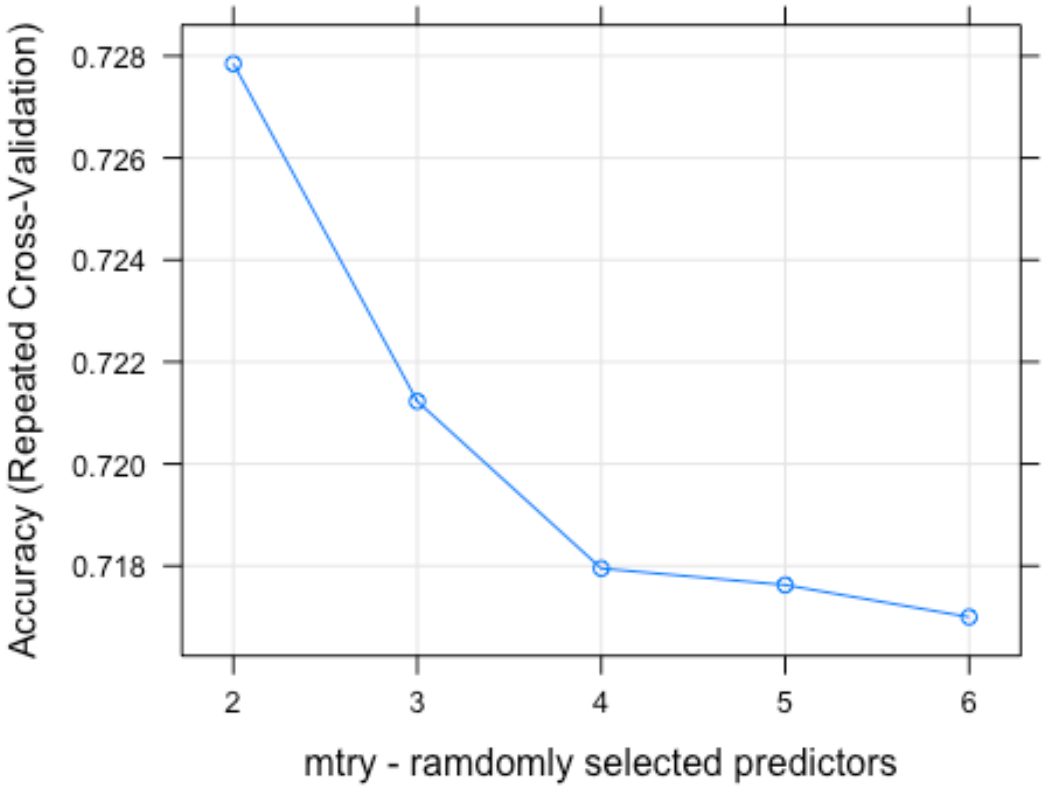
```
#*****************************************************************************************************
*****************************************************************************************
#*****************************************************************************************************
*****************************************************************************************
#*****************************************************************************************************
*****************************************************************************************

#=================================================================================================
==
# TAILLE : n = 24560

# train + caret + parallel
# https://github.com/lgreski/datasciencectacontent/blob/master/markdown/pml-randomForestPerformance.md

#*************************************************************

# PROCEDURE FINE TUNING AUTOMATIQUE RANDOM FOREST --- MODEL FINAL


# dq2norm_y=dq2norm_y[sample(1:nrow(dq2norm_y),614),]
#*************************************************************

##############################################################
# ITERATION 1 = random search
##############################################################

# DEBUT-- TIC & TOC
tic <- Sys.time()

# montage de code
#dq2norm_y1=dq2norm_y[sample(1:nrow(dq2norm_y),307),] #montage de code
dq2norm_y1=dq2norm_y[sample(1:nrow(dq2norm_y),6150),]


# step 0 : initialisation
library(caret)
seed=12345
set.seed(seed)

# create training & testing data sets
inTraining <- createDataPartition(dq2norm_y1$trsf_dev, p = .7, list=FALSE)
training <- dq2norm_y1[inTraining,]
testing <- dq2norm_y1[-inTraining,]

# set up training run for x / y syntax because model format performs poorly
x <- training[,-7]
y <- training[,7]

# creation of weights - also fast for very big datasets - correction des tailles des classes

# pris en compte du poids des classes sur dataset = apprentissage
summary(training$trsf_dev)
##    N    O
## 3181 1125
# N    O
# 4031 1505

table(training$trsf_dev)
##
##    N    O
## 3181 1125
wts=100/table(training$trsf_dev)
wts
```

```
##
##          N          O
## 0.03143666 0.08888889
# N          O
# 0.02480774 0.06644518
# 100/modalite1   100/modalite2


# Step 1: Configure parallel processing
library(parallel)
library(doParallel)
## Loading required package: foreach
## Loading required package: iterators
cluster <- makeCluster(detectCores() - 1) # convention to leave 1 core for OS
#cluster <- makeCluster(detectCores()) # si pas de core pur OS = saturation mémoire OS
registerDoParallel(cluster)

# Step 2: Configure trainControl object

fitControl <- trainControl(method = "repeatedcv",
                           number = 10,
                           repeats = 10,
                           allowParallel = TRUE)

# nb var optimal racine(nombre de Xi)
set.seed(seed)
mtry <- sqrt(ncol(x))

# search
search = "random"

# metric
metric = "Accuracy"

#centrage-reduction :
preProcess <- c("center", "scale")

# Step 3: Develop training model
#modelRFopti <- train(x,y, method="rf",data=training ,trControl = fitControl)
modelRFopti <- train(x,y, data=training, method="rf", tuneLength=5, metric=metric, search=search, preProc
ess=preProcess , trControl=fitControl, weight=wts)

# Step 4: De-register parallel processing cluster
stopCluster(cluster)
registerDoSEQ()
modelRFopti
## Random Forest
##
## 4306 samples
##    6 predictor
##    2 classes: 'N', 'O'
##
## Pre-processing: centered, scaled
## Resampling: Cross-Validated (10 fold, repeated 10 times)
##
## Summary of sample sizes: 3875, 3876, 3876, 3875, 3876, 3875, ...
##
## Resampling results across tuning parameters:
##
##   mtry  Accuracy  Kappa   Accuracy SD  Kappa SD
##   2     0.728     0.0911  0.0118       0.0380
##   3     0.721     0.1090  0.0137       0.0452
##   4     0.718     0.1096  0.0143       0.0469
##   5     0.718     0.1136  0.0144       0.0460
##   6     0.717     0.1152  0.0134       0.0433
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 2.
modelRFopti$resample
##      Accuracy       Kappa     Resample
## 1   0.7302326 0.060711058 Fold09.Rep03
## 2   0.7285383 0.094228800 Fold07.Rep08
```

```
## 3   0.7186047 0.067261841 Fold08.Rep08
## 4   0.7209302 0.071541673 Fold09.Rep08
## 5   0.7232558 0.089404563 Fold10.Rep03
## 6   0.7215777 0.100803227 Fold01.Rep04
## 7   0.7401392 0.129732458 Fold07.Rep01
## 8   0.7255814 0.066249540 Fold03.Rep06
## 9   0.7262181 0.087323236 Fold10.Rep08
## 10  0.7465116 0.147166927 Fold05.Rep01
## 11  0.7291667 0.094518094 Fold02.Rep04
## 12  0.7308585 0.096076659 Fold08.Rep01
## 13  0.7465116 0.147166927 Fold04.Rep06
## 14  0.7325581 0.093425732 Fold01.Rep09
## 15  0.7099768 0.032295727 Fold06.Rep01
## 16  0.7186047 0.067261841 Fold03.Rep04
## 17  0.7302326 0.088983051 Fold09.Rep01
## 18  0.7401392 0.109996681 Fold05.Rep06
## 19  0.7238979 0.105107044 Fold02.Rep09
## 20  0.7192575 0.034830567 Fold08.Rep06
## 21  0.7401392 0.142410460 Fold04.Rep04
## 22  0.7378190 0.150227697 Fold10.Rep01
## 23  0.7325581 0.072406678 Fold06.Rep06
## 24  0.7285383 0.100850525 Fold03.Rep09
## 25  0.7465116 0.140725259 Fold09.Rep06
## 26  0.7354988 0.120691506 Fold05.Rep04
## 27  0.7199074 0.070422535 Fold01.Rep02
## 28  0.7186047 0.087481146 Fold07.Rep06
## 29  0.7348837 0.117964589 Fold04.Rep09
## 30  0.7199074 0.109566639 Fold10.Rep06
## 31  0.7162791 0.083251782 Fold06.Rep04
## 32  0.7046512 0.006332108 Fold02.Rep02
## 33  0.7418605 0.097697455 Fold09.Rep04
## 34  0.7116279 0.068223123 Fold05.Rep09
## 35  0.7279070 0.111224958 Fold01.Rep07
## 36  0.7325581 0.100221988 Fold07.Rep04
## 37  0.7215777 0.046425016 Fold03.Rep02
## 38  0.7116279 0.047653069 Fold10.Rep04
## 39  0.7331787 0.089598295 Fold06.Rep09
## 40  0.7302326 0.102490284 Fold02.Rep07
## 41  0.7169374 0.052029999 Fold08.Rep04
## 42  0.7215777 0.060626975 Fold04.Rep02
## 43  0.7122970 0.050525867 Fold01.Rep05
## 44  0.7331787 0.072632702 Fold07.Rep09
## 45  0.7122970 0.057391795 Fold03.Rep07
## 46  0.7169374 0.052029999 Fold10.Rep09
## 47  0.7255814 0.093734372 Fold05.Rep02
## 48  0.7534884 0.185896978 Fold02.Rep05
## 49  0.7325581 0.093425732 Fold08.Rep09
## 50  0.7232558 0.061892714 Fold04.Rep07
## 51  0.7354988 0.151809142 Fold01.Rep10
## 52  0.7511628 0.150072043 Fold06.Rep02
## 53  0.7262181 0.096468164 Fold03.Rep05
## 54  0.7192575 0.083510535 Fold09.Rep09
## 55  0.7395349 0.120397428 Fold05.Rep07
## 56  0.7122970 0.057391795 Fold02.Rep10
## 57  0.7302326 0.109094806 Fold07.Rep02
## 58  0.7116279 0.087923366 Fold04.Rep05
## 59  0.7238979 0.105107044 Fold10.Rep02
## 60  0.7192575 0.063262264 Fold06.Rep07
## 61  0.7401392 0.086207549 Fold03.Rep10
## 62  0.7378190 0.118705101 Fold08.Rep02
## 63  0.7395349 0.120397428 Fold05.Rep05
## 64  0.7470998 0.174402053 Fold01.Rep03
## 65  0.7534884 0.173736495 Fold07.Rep07
## 66  0.7279070 0.091385227 Fold04.Rep10
## 67  0.7162791 0.063013503 Fold09.Rep02
## 68  0.7378190 0.098643422 Fold06.Rep05
## 69  0.7285383 0.087508821 Fold02.Rep03
## 70  0.7447796 0.163815302 Fold08.Rep07
## 71  0.7238979 0.092117607 Fold05.Rep10
## 72  0.7238979 0.050783780 Fold01.Rep08
## 73  0.7139535 0.037628735 Fold07.Rep05
## 74  0.7146172 0.040714402 Fold03.Rep03
```

```
## 75  0.7331787 0.075676482 Fold09.Rep07
## 76  0.7006961 0.015824969 Fold06.Rep10
## 77  0.7285383 0.080688385 Fold02.Rep08
## 78  0.7354988 0.107595626 Fold08.Rep05
## 79  0.7348837 0.076905694 Fold04.Rep03
## 80  0.7245370 0.098990466 Fold10.Rep07
## 81  0.7465116 0.159764799 Fold07.Rep10
## 82  0.7308585 0.136928601 Fold03.Rep08
## 83  0.7331787 0.082690208 Fold09.Rep05
## 84  0.7099768 0.057684571 Fold05.Rep03
## 85  0.7215777 0.031787038 Fold01.Rep01
## 86  0.7192575 0.070110372 Fold08.Rep10
## 87  0.7418605 0.104670793 Fold04.Rep08
## 88  0.7169374 0.049322003 Fold10.Rep05
## 89  0.7378190 0.137893190 Fold06.Rep03
## 90  0.7325581 0.120012813 Fold02.Rep01
## 91  0.7116279 0.061465887 Fold09.Rep10
## 92  0.7162791 0.049017475 Fold05.Rep08
## 93  0.7587007 0.191894426 Fold01.Rep06
## 94  0.7186047 0.060321474 Fold07.Rep03
## 95  0.7418605 0.118299036 Fold03.Rep01
## 96  0.7186047 0.031495477 Fold10.Rep10
## 97  0.7285383 0.094228800 Fold06.Rep08
## 98  0.7076566 0.035211768 Fold02.Rep06
## 99  0.7308585 0.111782263 Fold08.Rep03
## 100 0.7285383 0.094228800 Fold04.Rep01
confusionMatrix.train(modelRFopti)
## Cross-Validated (10 fold, repeated 10 times) Confusion Matrix
##
## (entries are percentages of table totals)
##
##           Reference
## Prediction    N    O
##          N 69.3 22.7
##          O  4.5  3.5
# step 5 : best model = modelRFopti
# on récupère le meilleur modèle obtenu sur avec x train et y train
modelRFopti
## Random Forest
##
## 4306 samples
##    6 predictor
##    2 classes: 'N', 'O'
##
## Pre-processing: centered, scaled
## Resampling: Cross-Validated (10 fold, repeated 10 times)
##
## Summary of sample sizes: 3875, 3876, 3876, 3875, 3876, 3875, ...
##
## Resampling results across tuning parameters:
##
##   mtry  Accuracy  Kappa   Accuracy SD  Kappa SD
##   2     0.728     0.0911  0.0118       0.0380
##   3     0.721     0.1090  0.0137       0.0452
##   4     0.718     0.1096  0.0143       0.0469
##   5     0.718     0.1136  0.0144       0.0460
##   6     0.717     0.1152  0.0134       0.0433
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 2.
head(with(modelRFopti, results[order(results$Kappa, decreasing=T),]))
##   mtry  Accuracy      Kappa AccuracySD    KappaSD
## 5    6 0.7169969 0.11515292 0.01337941 0.04332454
## 4    5 0.7176255 0.11359448 0.01441295 0.04595123
## 3    4 0.7179505 0.10963826 0.01432472 0.04688996
## 2    3 0.7212268 0.10896667 0.01371382 0.04516993
## 1    2 0.7278466 0.09108725 0.01175939 0.03801785
summary(modelRFopti)
##                 Length Class      Mode
## call               7   -none-     call
## type               1   -none-     character
## predicted       4306   factor     numeric
```

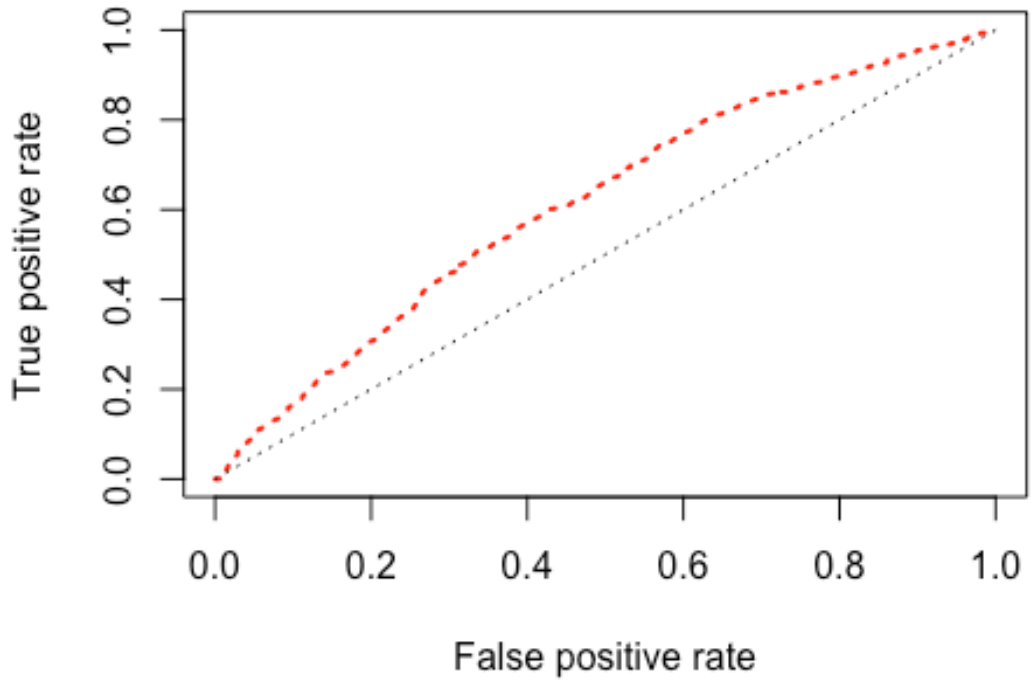
```
## err.rate        1500   -none-     numeric
## confusion          6   -none-     numeric
## votes           8612   matrix     numeric
## oob.times       4306   -none-     numeric
## classes            2   -none-     character
## importance         6   -none-     numeric
## importanceSD       0   -none-     NULL
## localImportance    0   -none-     NULL
## proximity          0   -none-     NULL
## ntree              1   -none-     numeric
## mtry               1   -none-     numeric
## forest            14   -none-     list
## y               4306   factor     numeric
## test               0   -none-     NULL
## inbag              0   -none-     NULL
## xNames             6   -none-     character
## problemType        1   -none-     character
## tuneValue          1   data.frame list
## obsLevels          2   -none-     character
# step 6 : graphe model
#plot(modelRFopti)
plot(modelRFopti, xlab = "mtry - ramdomly selected predictors") # vérifier graphiquement, quel est le mtry = 2 qui maximise l'accuracy.
```




```
# step 7 : prévision sur test
rfOptiTest = predict.train(object = modelRFopti, newdata = testing[,1:6], type='raw', testing[,7])
rfOptiTest
##    [1] N N N N N N N N O O N N N N N N N N N N N N N N N N N N N N N N N
##   [35] N N N N N N N N N N N N O N N N N N O N N N N N N N N N N N O N N
##   [69] N N N N O N N N N N N N N N N O O N N N N N N N N N N N O N O N N
##  [103] N N N N N N N N O N N N N N N N N N N N N N N N N O N N N O N N N
##  [137] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [171] N N O O N N N N N N N N N N N N N N N O N O N N N N N N N N N N N
##  [205] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [239] N N N O O N N N N N O N N N N N N N N N N N N N N O N N N N N N N
##  [273] N N N N N N N N N N N N N N N N N O N N N N N N N N N N N N N N N
##  [307] N N N N O N N N N O O N O N O N N N N N N N N N N N N N O N N N N
##  [341] N N N N N N N N N N N N O N N N N N N N O O O N N N O N N N N N N
##  [375] N N N O N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [409] N N N O N N N N O N N N N N N N O N N O N N N O N N N N N O N N N
##  [443] O N N N N N N N N N N N O N N N O N N N N N N N O N N O N N N N N N
```

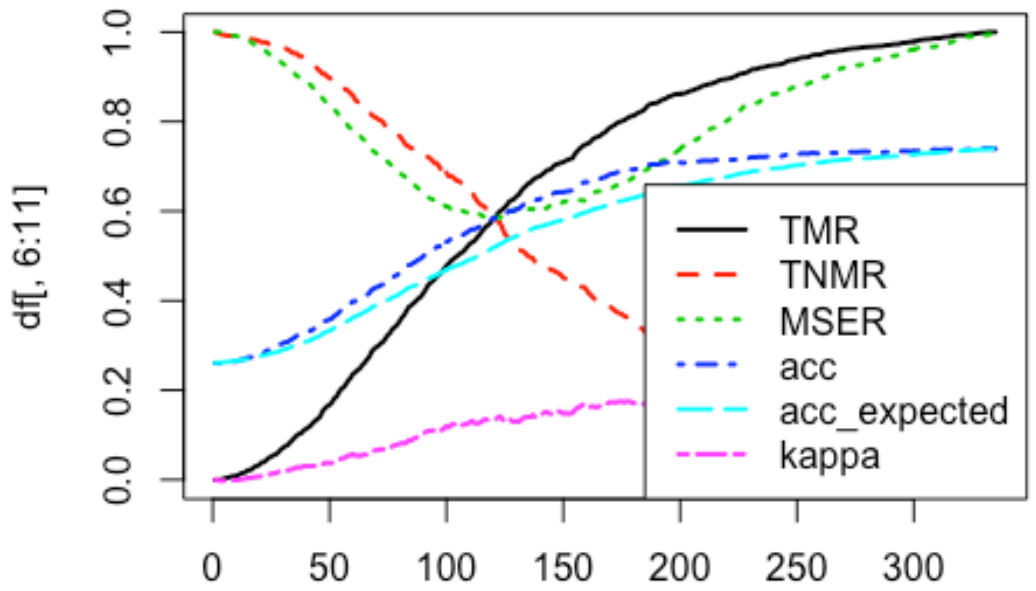
```
##  [477] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [511] O N N N N N N N N N N N N N N N N N N N N O N N N N N N N N O N N N
##  [545] O N N N N N N N N N N N O N N N N N N N N N N N N N N N N N N N N
##  [579] N N N N N N N N N N N N N N N N N N N N N N O N N N N N N N N N N
##  [613] N N N N N N N N N N N N N O N N N N N O N N O N N N N N N N N N N
##  [647] N N N N N N N N O N N N N N N N N N N N N N N N N O O N N N N N N
##  [681] N N N N N N N N N N N N N N N N N N N N N N N N N N N O N N N N N
##  [715] N O N N N N N N N N N N N N N N N N N N O N N N N N N N N N O N N
##  [749] N N N N N N N N N N N N N N N N O N N N N N N N N N O N N N N N N
##  [783] N O N N N N N N N N N N N N N N N N N N N N O N N N N N N N N N N
##  [817] N N N N N N N N O N N N N N N O O N N N O O N O N N N N N N N N O
##  [851] N N N N O N O N N N N N N N N N N N O N N N N N N N N N N N N N N
##  [885] N O N N N N N N N O N O N O N N O N N N N N N N O N N N N O N N N N
##  [919] O N N N N N N N N N N N N N N N N N N N N N N N N N N N N N O N N O
##  [953] N N N N N N N N O N N N O N N N N N N N N N N N N N N N N N N O N
##  [987] N O N N N N N N N N N N N N N N N N N N N O N N N N N N N N N N N N
## [1021] N N N N N N N N N N N N N N N N N N N N N N N N N O N N N N N N N N
## [1055] N N N N N N N N O N N N N N N O O N N N N N N N N N N N N N N N N N
## [1089] N N N N N N N N N N N N N N N N O N N N N N N N N N N N N N O O N N
## [1123] N N N N N N N N N N N N N O N N N N N N N N N O N N N N N N N N N
## [1157] N N N N N N N N O N N N N N N N N N O N N N N N N N N N N N O N N
## [1191] N N N N N N N N N O O N N N N N O N N N N N N N N N N O O N N N N N
## [1225] N N N N N N N N N N O N N N O N N N N N N N O N N N N N N N N N N
## [1259] N N N O N O O N N N N N N N N N N O N N N N N N N N N N N N N N N
## [1293] N N N N N N N N N O N N N N N N O N N N N N N N N O N N O N O N N N
## [1327] N O N N N N N N N N N N N N O N N N N N N N N N N N N O N N N N N
## [1361] N N O N N N N N N O N N N O N N N N N N N N N N N N N N N N N N N
## [1395] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N O N N N N
## [1429] N N N N O N N N N N O N O O N N N N N N N N N N N N N N O N O N N N
## [1463] N N N N N N N N N N N N N N N N N N N N N N N N N O N N N N N N N O
## [1497] N N N N N N N N N N N N O N N N N N N N N N O N N N N O O N N N N N
## [1531] N N N N N N N N N N N O N N N N N N N N N N O N N N N O N N N N N N
## [1565] N N N N N N N N N N N O N N N N N N O N N N N N N N N N N N N N N
## [1599] N N N N N N N N N N N O N N N N O N N N N N N N N N N O N N N N N N
## [1633] N N N N N N N N N N N N N O N N N N N N N N N N N N N N N O N N N N
## [1667] N N N N N N N N N O N N N N N N N N N N N N N N N N N N N N N N N
## [1701] N N N N N N N N N N N N N N N N O N N N N N N N N N N N N O N N N N
## [1735] N N N N N N O N N N N N N N N N N N O N N N N N N O N N N O N N O N
## [1769] N N O N N N N N N O N N N N N N N N N N N N N O O N N N N N N N N
## [1803] N N O N N N O N N N N N N N N N N N N N N N N N N N O N N N N N N
## [1837] N O N N O N N N
## Levels: N O
# step 8 : Confusion Matrix and Statistics === FONCTION PACKAGE CARET
CM = confusionMatrix(data=rfOptiTest, testing[,7])
CM
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    N    O
##          N 1265  414
##          O   98   67
##
##                Accuracy : 0.7223
##                  95% CI : (0.7013, 0.7427)
##     No Information Rate : 0.7392
##     P-Value [Acc > NIR] : 0.9519
##
##                   Kappa : 0.0856
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.9281
##             Specificity : 0.1393
##          Pos Pred Value : 0.7534
##          Neg Pred Value : 0.4061
##              Prevalence : 0.7392
##          Detection Rate : 0.6860
##    Detection Prevalence : 0.9105
##       Balanced Accuracy : 0.5337
##
##        'Positive' Class : N
##
```

```
# F-mesure
CM$byClass[3]
## Pos Pred Value
##      0.7534247
#     0.7521059

# step 9 : confusion matrix: model predicting classes of test data
#table.SVM=table(rfOptiTest$y,testing[,7]) #
table.RF=table(testing[,7],rfOptiTest)
table.RF
##    rfOptiTest
##        N    O
##   N 1265   98
##   O  414   67
TxBcRF=(sum(diag(table.RF))/sum(table.RF))*100
TxBcRF
## [1] 72.23427
# [1] 72.77657

# prediction probabilities of test data classes
probs <- predict.train(object = modelRFopti, newdata = testing[,1:6], type='prob')[,1]
isPositiveClass <- testing[,7] == 'N' # for a ROC curve there is a positive class (true match rate...) -
defining that class here
pred <- prediction(probs, isPositiveClass)
perf <- performance(pred, 'tpr', 'fpr')

# plot: either
plot(perf, col=2, lwd=2,lty=3)
segments(0,0,1,1,lty=3)
```

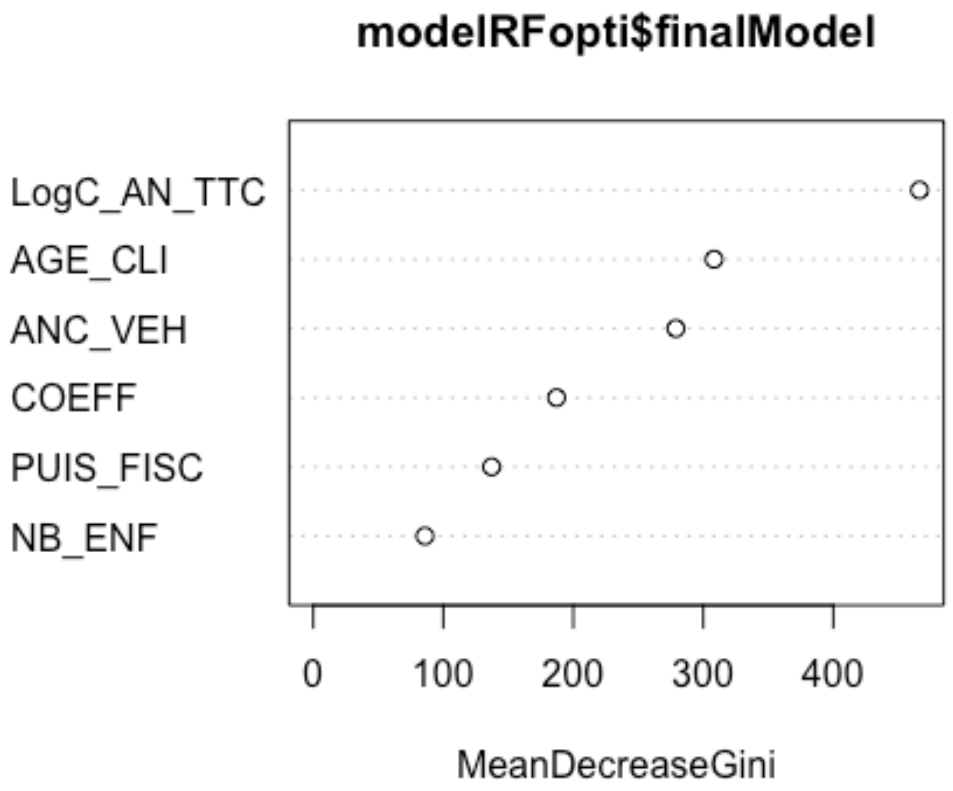



```
# or
#with(attributes(perf), plot(x=x.values[[1]], y=y.values[[1]], type='l'))

# step 10 :some metrics: AUC (area under curve), EER (equal error rate), MSER (minimum squared error rate
), Cohen's Kappa etc.
AUC <- attributes(performance(pred, 'auc'))$y.value[[1]] # area under curve
AUC
## [1] 0.6140088
# [1] 0.6266797

df <- with(attributes(pred), data.frame(cutoffs=cutoffs[[1]], tp=tp[[1]], fn=fn[[1]], tn=tn[[1]], fp=fp[[
```

```
1]], TMR=tp[[1]]/(tp[[1]]+fn[[1]]), TNMR=tn[[1]]/(tn[[1]]+fp[[1]])))
df$MSER <- with(df, sqrt((1-TMR)**2+(1-TNMR)**2))
MSER <- with(df, sqrt(TMR**2+TNMR**2)) # sqrt of minimum squared error rate = eucl. distance to point TMR
=TNMR=1
i_eer <- with(df, which.min(abs(TMR-TNMR)))
EER <- with(df[i_eer,], mean(c(TMR,TNMR))) # equal error rate: mean would not be required when using ROCR
as it's always exact
df$acc <- with(df, (tp + tn) / length(isPositiveClass)) # observed accuracy
df$acc_expected <- with(df, sum(isPositiveClass) * (tp+fp) / length(isPositiveClass) + sum(!isPositiveCla
ss) * (tn+fn) / length(isPositiveClass)) / length(isPositiveClass) # expected accuracy
df$kappa <- with(df, (acc - acc_expected) / (1 - acc_expected)) # cohen's kappa

# graphical representation
matplot(df[,6:11], lty=1:6, lwd=2, type='l'); legend('bottomright', legend=names(df[,6:11]), lty=1:6, lwd
=2, col=1:6)
```




```
# characteristics for settings with best EER, MSER etc.
df[i_eer,] # curve point for equal error rate
##     cutoffs  tp  fn  tn  fp       TMR      TNMR      MSER       acc
## 120   0.748 795 568 282 199 0.5832722 0.5862786 0.5872201 0.5840564
##     acc_expected     kappa
## 120    0.5186758 0.1358348
df[order(df$acc, decreasing=T)[[1]],] # curve point for max accuracy
##     cutoffs   tp fn tn  fp       TMR       TNMR      MSER       acc
## 329   0.248 1361  2  5 476 0.9985326 0.01039501 0.9896061 0.7407809
##     acc_expected     kappa
## 329    0.7373383 0.0131066
df[order(df$kappa, decreasing=T)[[1]],] # curve point for max kappa
##     cutoffs   tp  fn  tn  fp       TMR      TNMR      MSER       acc
## 177   0.634 1099 264 175 306 0.8063096 0.3638254 0.6650069 0.6908894
##     acc_expected     kappa
## 177    0.6252835 0.1750813
df[order(df$MSER, decreasing=F)[[1]],] # curve point for minimum squared error rate
##     cutoffs  tp  fn  tn  fp       TMR      TNMR      MSER       acc
## 123   0.742 818 545 276 205 0.6001467 0.5738046 0.5844016 0.5932755
##     acc_expected     kappa
## 123     0.526198 0.1415728
# step 11 : importance des variables
varImpPlot(modelRFopti$finalModel)
```



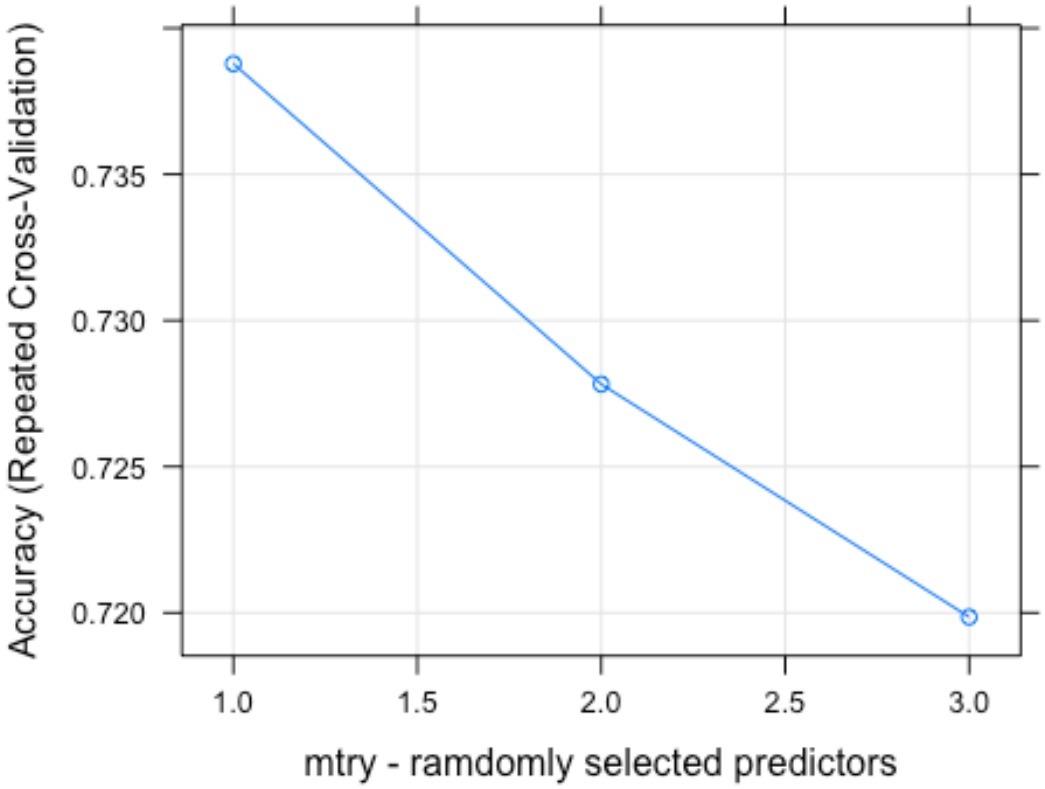

```
varImp(modelRFopti)
## rf variable importance
##
##             Overall
## LogC_AN_TTC  100.00
## AGE_CLI       58.45
## ANC_VEH       50.73
## COEFF         26.61
## PUIS_FISC     13.47
## NB_ENF         0.00
# timer
toc <-  Sys.time() -tic
print(toc)
## Time difference of 11.41732 mins

#################################################################
# ITERATION 2 = Grid Search
#################################################################
#  DEBUT-- TIC & TOC
tic <- Sys.time()

# montage de code
#dq2norm_y1=dq2norm_y[sample(1:nrow(dq2norm_y),307),] #montage de code
dq2norm_y1=dq2norm_y[sample(1:nrow(dq2norm_y),6150),]

# step 0 : initialisation
library(caret)
seed=12345
set.seed(seed)

# create training & testing data sets
inTraining <- createDataPartition(dq2norm_y1$trsf_dev, p = .7, list=FALSE)
training <- dq2norm_y1[inTraining,]
testing <- dq2norm_y1[-inTraining,]

# set up training run for x / y syntax because model format performs poorly
x <- training[,-7]
y <- training[,7]

# creation of weights - also fast for very big datasets - correction des tailles des classes
```

```r
# pris en compte du poids des classes sur dataset = apprentissage
summary(training$trsf_dev)
##    N    O
## 3181 1125
#N    O
#4031 1505

wts=100/table(training$trsf_dev)
wts
##
##          N          O
## 0.03143666 0.08888889
#N          O
#0.02480774 0.06644518


# Step 1: Configure parallel processing
library(parallel)
library(doParallel)
cluster <- makeCluster(detectCores() - 1) # convention to leave 1 core for OS
#cluster <- makeCluster(detectCores()) # si pas de core pur OS = saturation mémoire OS
registerDoParallel(cluster)

# Step 2: Configure trainControl object
fitControl <- trainControl(method = "repeatedcv",
                           number = 10,
                           repeats = 10,
                           allowParallel = TRUE)

# search
search = "grid"

# metric
metric = "Accuracy"

#centrage-reduction :
preProcess <- c("center", "scale")


set.seed(seed)
tunegrid <- expand.grid(.mtry=c(1:3))

# Step 3: Develop training model
#modelRFopti <- train(x,y, method="rf",data=training ,trControl = fitControl)
modelRFopti <- train(x,y, data=training, method="rf", tuneLength=5, trControl = fitControl, search=search
, metric=metric, preProcess=preProcess,weight = wts, tuneGrid=tunegrid)

# Step 4: De-register parallel processing cluster
stopCluster(cluster)
registerDoSEQ()
modelRFopti
## Random Forest
##
## 4306 samples
##    6 predictor
##    2 classes: 'N', 'O'
##
## Pre-processing: centered, scaled
## Resampling: Cross-Validated (10 fold, repeated 10 times)
##
## Summary of sample sizes: 3875, 3876, 3876, 3875, 3876, 3875, ...
##
## Resampling results across tuning parameters:
##
##   mtry  Accuracy  Kappa   Accuracy SD  Kappa SD
##   1     0.739     0.0111  0.00351      0.0133
##   2     0.728     0.0954  0.01461      0.0446
##   3     0.720     0.1087  0.01592      0.0486
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 1.
```

```
modelRFopti$resample
##      Accuracy         Kappa     Resample
## 1   0.7378190  0.0165579630 Fold04.Rep02
## 2   0.7378190  0.0165579630 Fold03.Rep02
## 3   0.7395349  0.0000000000 Fold05.Rep02
## 4   0.7361111 -0.0046101750 Fold01.Rep02
## 5   0.7447796  0.0386877002 Fold10.Rep01
## 6   0.7418605  0.0214850957 Fold06.Rep02
## 7   0.7348837 -0.0092234209 Fold02.Rep02
## 8   0.7354988 -0.0092018239 Fold08.Rep01
## 9   0.7348837 -0.0007349339 Fold07.Rep02
## 10  0.7424594  0.0258994564 Fold04.Rep01
## 11  0.7395349  0.0000000000 Fold09.Rep01
## 12  0.7378190  0.0083480952 Fold08.Rep02
## 13  0.7418605  0.0131497333 Fold05.Rep01
## 14  0.7378190  0.0083480952 Fold01.Rep01
## 15  0.7372093 -0.0046313526 Fold09.Rep02
## 16  0.7424594  0.0258994564 Fold06.Rep01
## 17  0.7325581 -0.0052856272 Fold02.Rep01
## 18  0.7308585 -0.0053893179 Fold10.Rep02
## 19  0.7354988  0.0200247317 Fold07.Rep01
## 20  0.7348837 -0.0092234209 Fold03.Rep01
## 21  0.7447796  0.0386877002 Fold01.Rep03
## 22  0.7401392  0.0130040075 Fold08.Rep05
## 23  0.7424594  0.0258994564 Fold05.Rep04
## 24  0.7424594  0.0258994564 Fold02.Rep03
## 25  0.7378190  0.0000000000 Fold09.Rep05
## 26  0.7395349  0.0250222690 Fold06.Rep04
## 27  0.7424594  0.0258994564 Fold03.Rep03
## 28  0.7331787 -0.0137442988 Fold10.Rep05
## 29  0.7441860  0.0343785726 Fold07.Rep04
## 30  0.7395349  0.0084822531 Fold04.Rep03
## 31  0.7424594  0.0258994564 Fold01.Rep06
## 32  0.7401392  0.0212092948 Fold08.Rep04
## 33  0.7424594  0.0215163725 Fold05.Rep03
## 34  0.7401392  0.0130040075 Fold02.Rep06
## 35  0.7395349  0.0084822531 Fold09.Rep04
## 36  0.7354988  0.0200247317 Fold06.Rep03
## 37  0.7372093  0.0121976011 Fold03.Rep06
## 38  0.7372093 -0.0046313526 Fold10.Rep04
## 39  0.7372093 -0.0046313526 Fold07.Rep03
## 40  0.7395349  0.0084822531 Fold04.Rep06
## 41  0.7424594  0.0258994564 Fold01.Rep05
## 42  0.7354988  0.0037308893 Fold08.Rep03
## 43  0.7401392  0.0130040075 Fold05.Rep06
## 44  0.7395349  0.0084822531 Fold02.Rep05
## 45  0.7372093  0.0038541966 Fold09.Rep03
## 46  0.7418605  0.0131497333 Fold06.Rep06
## 47  0.7378190  0.0165579630 Fold03.Rep05
## 48  0.7395349  0.0000000000 Fold10.Rep03
## 49  0.7418605  0.0296808294 Fold07.Rep06
## 50  0.7418605  0.0214850957 Fold04.Rep05
## 51  0.7401392  0.0212092948 Fold01.Rep04
## 52  0.7378190  0.0083480952 Fold08.Rep06
## 53  0.7418605  0.0296808294 Fold05.Rep05
## 54  0.7430556  0.0340048348 Fold02.Rep04
## 55  0.7372093 -0.0046313526 Fold09.Rep06
## 56  0.7308585  0.0028321832 Fold06.Rep05
## 57  0.7372093  0.0038541966 Fold03.Rep04
## 58  0.7430556  0.0259202080 Fold10.Rep06
## 59  0.7395349  0.0084822531 Fold07.Rep05
## 60  0.7378190  0.0083480952 Fold04.Rep04
## 61  0.7372093 -0.0046313526 Fold01.Rep07
## 62  0.7418605  0.0131497333 Fold08.Rep09
## 63  0.7395349  0.0168218194 Fold05.Rep08
## 64  0.7418605  0.0131497333 Fold02.Rep07
## 65  0.7424594  0.0339640167 Fold09.Rep09
## 66  0.7378190  0.0083480952 Fold06.Rep08
## 67  0.7378190  0.0165579630 Fold03.Rep07
## 68  0.7378190  0.0083480952 Fold10.Rep09
## 69  0.7378190  0.0083480952 Fold07.Rep08
## 70  0.7441860  0.0261879272 Fold04.Rep07
```

```
## 71  0.7354988  0.0119449807 Fold01.Rep10
## 72  0.7372093 -0.0046313526 Fold08.Rep08
## 73  0.7441860  0.0261879272 Fold05.Rep07
## 74  0.7401392  0.0212092948 Fold02.Rep10
## 75  0.7395349  0.0084822531 Fold09.Rep08
## 76  0.7308585 -0.0137475161 Fold06.Rep07
## 77  0.7401392  0.0168635438 Fold03.Rep10
## 78  0.7424594  0.0131603375 Fold10.Rep08
## 79  0.7372093 -0.0046313526 Fold07.Rep07
## 80  0.7418605  0.0214850957 Fold04.Rep10
## 81  0.7372093  0.0038541966 Fold01.Rep09
## 82  0.7331787 -0.0092032659 Fold08.Rep07
## 83  0.7378190  0.0000000000 Fold05.Rep10
## 84  0.7354988 -0.0046209209 Fold02.Rep09
## 85  0.7308585 -0.0137475161 Fold09.Rep07
## 86  0.7354988 -0.0046209209 Fold06.Rep10
## 87  0.7378190  0.0083480952 Fold03.Rep09
## 88  0.7430556  0.0340048348 Fold10.Rep07
## 89  0.7441860  0.0261879272 Fold07.Rep10
## 90  0.7418605  0.0131497333 Fold04.Rep09
## 91  0.7308585 -0.0053893179 Fold01.Rep08
## 92  0.7424594  0.0258994564 Fold08.Rep10
## 93  0.7302326 -0.0097983642 Fold05.Rep09
## 94  0.7424594  0.0258994564 Fold02.Rep08
## 95  0.7395349  0.0084822531 Fold09.Rep10
## 96  0.7401392  0.0292792793 Fold06.Rep09
## 97  0.7378190  0.0083480952 Fold03.Rep08
## 98  0.7395349  0.0084822531 Fold10.Rep10
## 99  0.7378190 -0.0046205574 Fold07.Rep09
## 100 0.7325581  0.0030643926 Fold04.Rep08
confusionMatrix.train(modelRFopti)
## Cross-Validated (10 fold, repeated 10 times) Confusion Matrix
##
## (entries are percentages of table totals)
##
##           Reference
## Prediction    N    O
##          N 73.6 25.8
##          O  0.3  0.3
# step 5 : best model = modelRFopti
# on récupère le meilleur modèle obtenu sur avec x train et y train
modelRFopti
## Random Forest
##
## 4306 samples
##    6 predictor
##    2 classes: 'N', 'O'
##
## Pre-processing: centered, scaled
## Resampling: Cross-Validated (10 fold, repeated 10 times)
##
## Summary of sample sizes: 3875, 3876, 3876, 3875, 3876, 3875, ...
##
## Resampling results across tuning parameters:
##
##   mtry  Accuracy  Kappa   Accuracy SD  Kappa SD
##   1     0.739     0.0111  0.00351      0.0133
##   2     0.728     0.0954  0.01461      0.0446
##   3     0.720     0.1087  0.01592      0.0486
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 1.
head(with(modelRFopti, results[order(results$Kappa, decreasing=T),]))
##   mtry  Accuracy      Kappa  AccuracySD    KappaSD
## 3    3 0.7198537 0.10865887 0.015915174 0.04862965
## 2    2 0.7278208 0.09543979 0.014613693 0.04464654
## 1    1 0.7387831 0.01105200 0.003510726 0.01331115
summary(modelRFopti)
##                 Length Class      Mode
## call               7   -none-     call
## type               1   -none-     character
## predicted       4306   factor     numeric
```

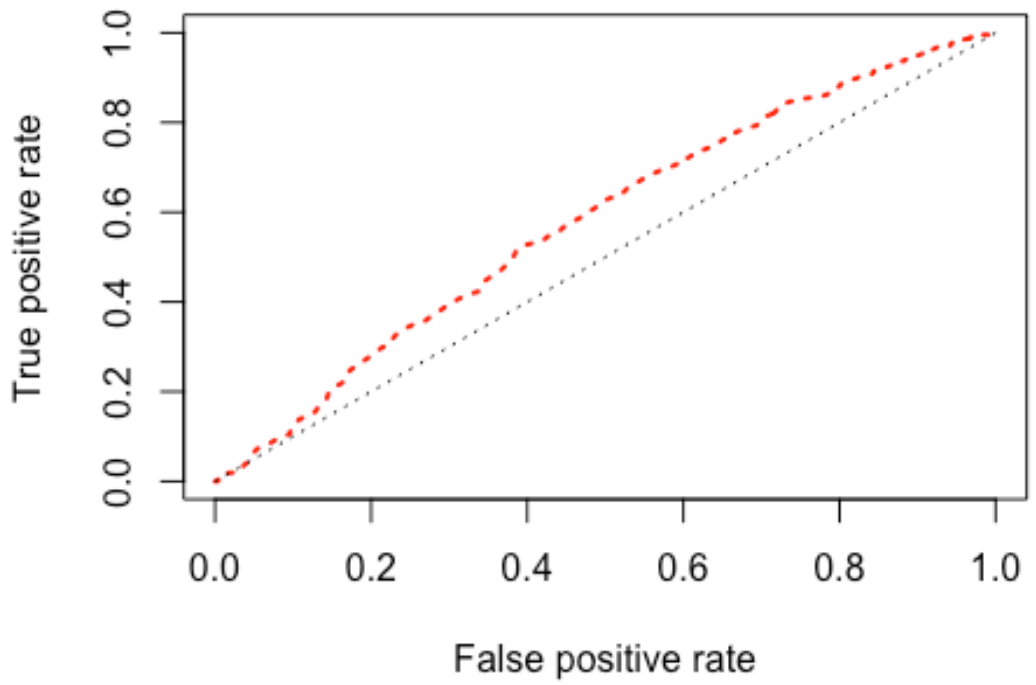
```
## err.rate        1500   -none-     numeric
## confusion          6   -none-     numeric
## votes           8612   matrix     numeric
## oob.times       4306   -none-     numeric
## classes            2   -none-     character
## importance         6   -none-     numeric
## importanceSD       0   -none-     NULL
## localImportance    0   -none-     NULL
## proximity          0   -none-     NULL
## ntree              1   -none-     numeric
## mtry               1   -none-     numeric
## forest            14   -none-     list
## y               4306   factor     numeric
## test               0   -none-     NULL
## inbag              0   -none-     NULL
## xNames             6   -none-     character
## problemType        1   -none-     character
## tuneValue          1   data.frame list
## obsLevels          2   -none-     character
# step 6 : graphe model
#plot(modelRFopti)
plot(modelRFopti, xlab = "mtry - ramdomly selected predictors") # vérifier graphiquement, quel est le mtry = 1 qui maximise l'accuracy.
```




```
# step 7 : prévision sur test
rfOptiTest = predict.train(object = modelRFopti, newdata = testing[,1:6], type='raw', testing[,7])
rfOptiTest
##    [1] N N N N N N N N N N N N N N N N N N N N N N O N N N N N N N N N
##   [35] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##   [69] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [103] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [137] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [171] N N N N N N N N N N N N N N N N N N N N N N N N O N N N N N N N N
##  [205] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [239] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [273] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [307] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [341] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [375] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [409] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [443] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
```

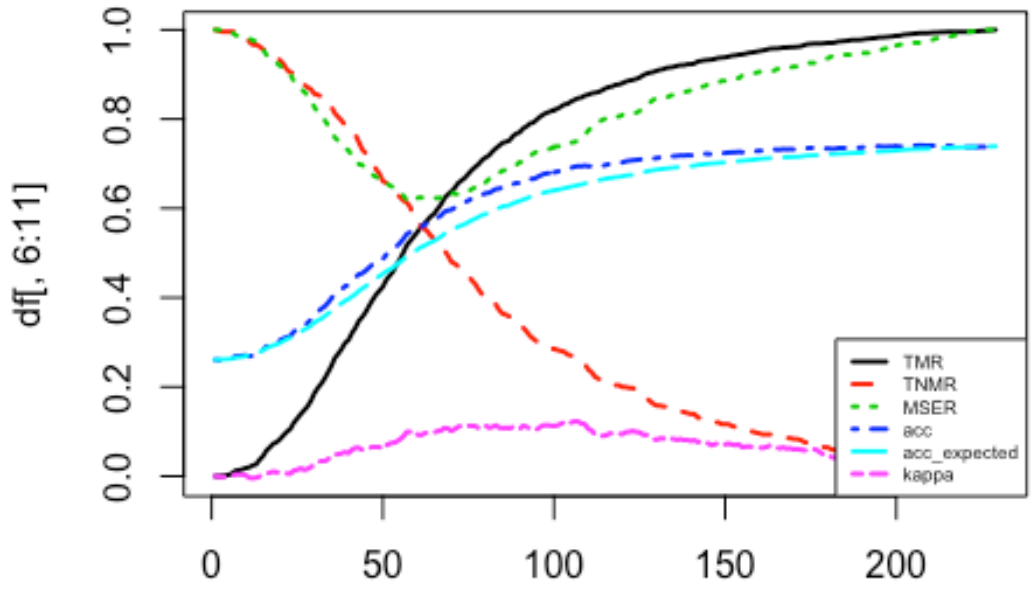
```
##  [477] N N N N N N N N N N N N N O N N N N N N N N N N N N N N N O N
##  [511] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [545] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [579] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [613] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [647] N N N N N N N N N N N N N N N N N O O N N N N N N N N N N N N
##  [681] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [715] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [749] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [783] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [817] N N N N N N N N N N N N N N N N O N N N N N N N N N N N N N N
##  [851] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [885] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [919] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [953] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [987] N N N N N N O N N N N N N N N N N N N N N N N N N N N N N N N
## [1021] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1055] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1089] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1123] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1157] N N N N N N N N O N N N N N N N N N N N N N N N N N N N N N N
## [1191] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1225] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1259] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1293] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1327] N N N N N N N N N N N O N N N N N N N N N N N N N N N N N N N
## [1361] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1395] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1429] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1463] N N N N N N N N N N N N N N N N N N N N N O N N N N N N N N N
## [1497] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1531] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1565] N N N N N N N N N N N N N N N N N N N N N N N N N N N O N N N
## [1599] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1633] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1667] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1701] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1735] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1769] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1803] N N N N N N N N N N N N N N N N N N N N N O N N N N N N N N N
## [1837] N N N N N N N
## Levels: N O
# step 8 : Confusion Matrix and Statistics === FONCTION PACKAGE CARET
CM = confusionMatrix(data=rfOptiTest, testing[,7])
CM
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    N    O
##          N 1357  474
##          O    6    7
##
##                Accuracy : 0.7397
##                  95% CI : (0.719, 0.7596)
##     No Information Rate : 0.7392
##     P-Value [Acc > NIR] : 0.4911
##
##                   Kappa : 0.0148
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.99560
##             Specificity : 0.01455
##          Pos Pred Value : 0.74113
##          Neg Pred Value : 0.53846
##              Prevalence : 0.73915
##          Detection Rate : 0.73590
##    Detection Prevalence : 0.99295
##       Balanced Accuracy : 0.50508
##
##        'Positive' Class : N
##
```

```
# F-mesure
CM$byClass[3]
## Pos Pred Value
##      0.7411251
# step 9 : confusion matrix: model predicting classes of test data
#table.SVM=table(rfOptiTest$y,testing[,7]) #
table.RFGS=table(testing[,7],rfOptiTest)
table.RFGS
##    rfOptiTest
##        N    O
##   N 1357    6
##   O  474    7
TxBcRFGS=(sum(diag(table.RFGS))/sum(table.RFGS))*100
TxBcRFGS
## [1] 73.96963
# [1] 73.28

# prediction probabilities of test data classes
probs <- predict.train(object = modelRFopti, newdata = testing[,1:6], type='prob')[,1]
isPositiveClass <- testing[,7] == 'N' # for a ROC curve there is a positive class (true match rate...) -
defining that class here
pred <- prediction(probs, isPositiveClass)
perf <- performance(pred, 'tpr', 'fpr')

# plot: either
plot(perf, col=2, lwd=2,lty=3)
segments(0,0,1,1,lty=3)
```

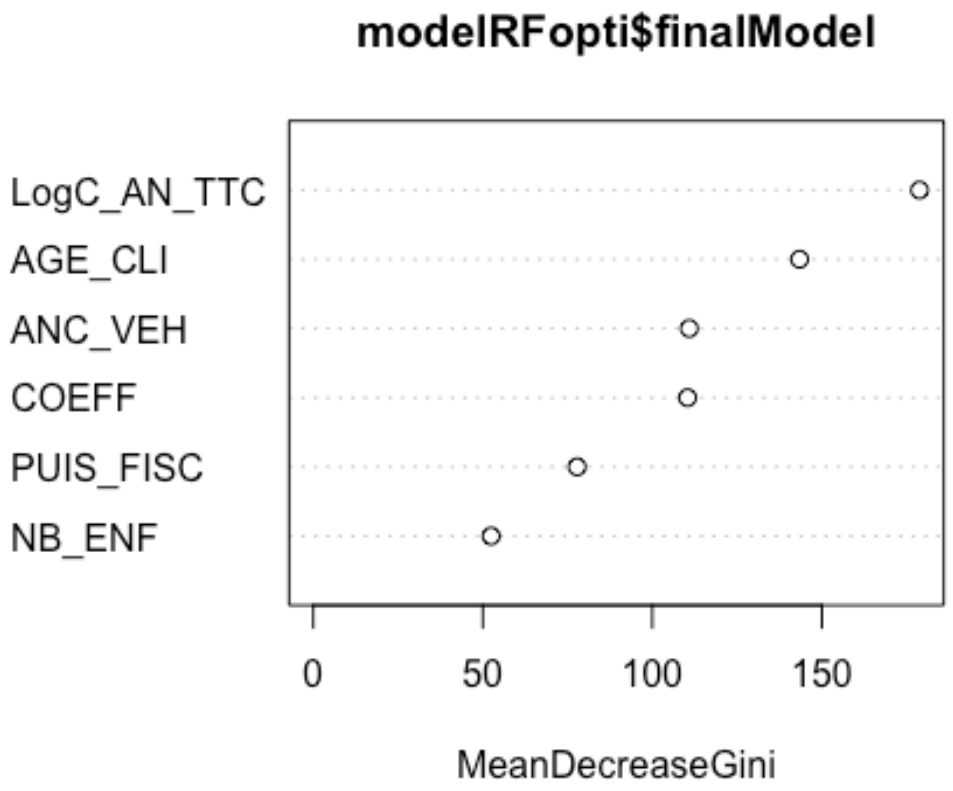



```
# or
#with(attributes(perf), plot(x=x.values[[1]], y=y.values[[1]], type='l'))

# step 10 :some metrics : AUC (area under curve), EER (equal error rate), MSER (minimum squared error rat
e), Cohen's Kappa etc.
AUC <- attributes(performance(pred, 'auc'))$y.value[[1]] # area under curve
AUC
## [1] 0.5799188
# [1] 0.6017

df <- with(attributes(pred), data.frame(cutoffs=cutoffs[[1]], tp=tp[[1]], fn=fn[[1]], tn=tn[[1]], fp=fp[[
1]], TMR=tp[[1]]/(tp[[1]]+fn[[1]]), TNMR=tn[[1]]/(tn[[1]]+fp[[1]])))
df$MSER <- with(df, sqrt((1-TMR)**2+(1-TNMR)**2))
```

```r
MSER <- with(df, sqrt(TMR**2+TNMR**2)) # sqrt of minimum squared error rate = eucl. distance to point TMR
=TNMR=1
i_eer <- with(df, which.min(abs(TMR-TNMR)))
EER <- with(df[i_eer,], mean(c(TMR,TNMR))) # equal error rate: mean would not be required when using ROCR
as it's always exact
df$acc <- with(df, (tp + tn) / length(isPositiveClass)) # observed accuracy
df$acc_expected <- with(df, sum(isPositiveClass) * (tp+fp) / length(isPositiveClass) + sum(!isPositiveCla
ss) * (tn+fn) / length(isPositiveClass)) / length(isPositiveClass) # expected accuracy
df$kappa <- with(df, (acc - acc_expected) / (1 - acc_expected)) # cohen's kappa

# graphical representation
matplot(df[,6:11], lty=1:6, lwd=2, type='l'); legend('bottomright', legend=names(df[,6:11]), lty=1:6, lwd
=2, col=1:6,cex=0.5)
```




```r
# characteristics for settings with best EER, MSER etc.
df[i_eer,] # curve point for equal error rate
##    cutoffs  tp  fn  tn  fp       TMR      TNMR      MSER       acc
## 61    0.88 759 604 269 212 0.5568599 0.5592516 0.6250059 0.5574837
##    acc_expected      kappa
## 61    0.5127099 0.09188328
df[order(df$acc, decreasing=T)[[1]],] # curve point for max accuracy
##     cutoffs   tp fn tn  fp       TMR       TNMR      MSER       acc
## 207   0.548 1353 10 14 467 0.9926632 0.02910603 0.9709217 0.7413232
##     acc_expected      kappa
## 207    0.7329287 0.03143155
df[order(df$kappa, decreasing=T)[[1]],] # curve point for max kappa
##     cutoffs   tp  fn  tn  fp       TMR      TNMR     MSER       acc
## 107   0.788 1154 209 127 354 0.8466618 0.2640333 0.751771 0.6946855
##     acc_expected     kappa
## 107    0.6520003 0.1226587
df[order(df$MSER, decreasing=F)[[1]],] # curve point for minimum squared error rate
##    cutoffs  tp  fn  tn  fp       TMR      TNMR      MSER       acc
## 58   0.886 718 645 290 191 0.5267792 0.6029106 0.6177523 0.5466377
##    acc_expected      kappa
## 58     0.496628 0.09934951
# step 11 : importance des variables
varImpPlot(modelRFopti$finalModel)
```



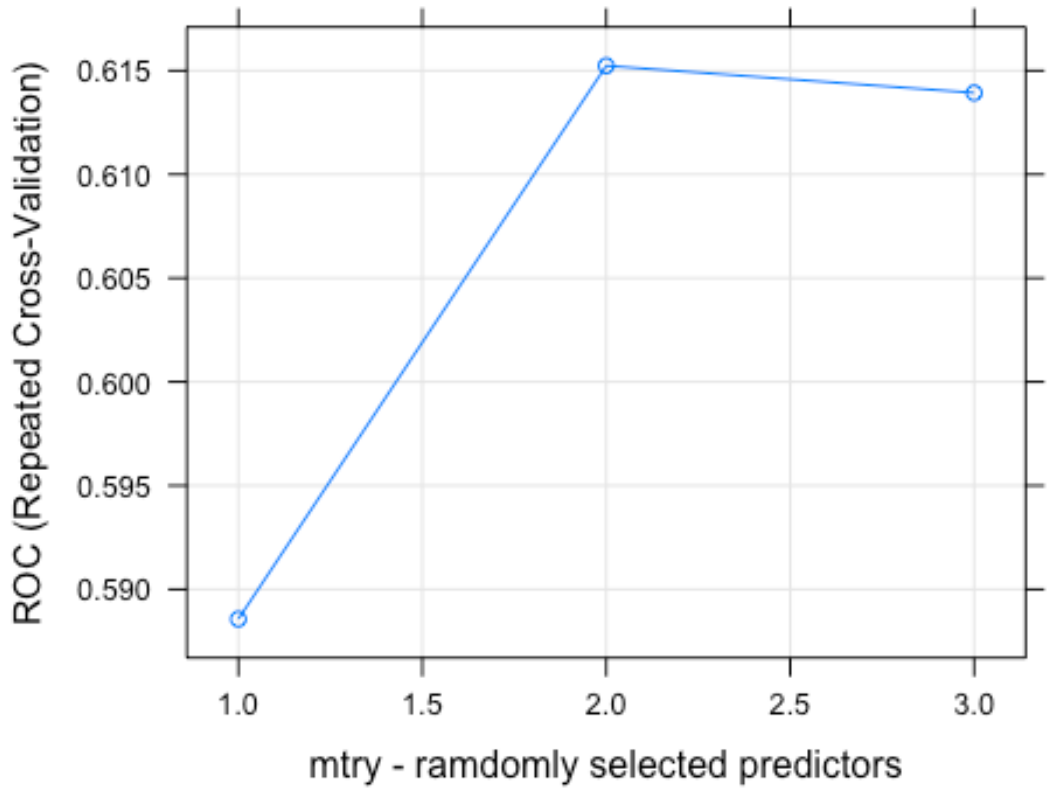

```
varImp(modelRFopti)
## rf variable importance
##
##             Overall
## LogC_AN_TTC  100.00
## AGE_CLI       72.01
## ANC_VEH       46.25
## COEFF         45.86
## PUIS_FISC     20.07
## NB_ENF         0.00
# timer
toc <-  Sys.time() -tic
print(toc)
## Time difference of 5.327904 mins

################################################################################
# ITERATION 3 = Grid Search - code pb classification binaire
################################################################################
#  DEBUT-- TIC & TOC
tic <- Sys.time()

# montage de code
#dq2norm_y1=dq2norm_y[sample(1:nrow(dq2norm_y),307),] #montage de code
dq2norm_y1=dq2norm_y[sample(1:nrow(dq2norm_y),6150),]

# step 0 : initialisation
library(caret)
seed=12345
set.seed(seed)

# create training & testing data sets
inTraining <- createDataPartition(dq2norm_y1$trsf_dev, p = .7, list=FALSE)
training <- dq2norm_y1[inTraining,]
testing <- dq2norm_y1[-inTraining,]

# set up training run for x / y syntax because model format performs poorly
x <- training[,-7]
y <- training[,7]

# creation of weights - also fast for very big datasets - correction des tailles des classes
```

```
# pris en compte du poids des classes sur dataset = apprentissage
summary(training$trsf_dev)
##    N    O
## 3181 1125
# N    O
# 4031 1505

wts=100/table(training$trsf_dev)
wts
##
##          N          O
## 0.03143666 0.08888889
# N          O
# 0.02480774 0.06644518


# Step 1: Configure parallel processing
library(parallel)
library(doParallel)
cluster <- makeCluster(detectCores() - 1) # convention to leave 1 core for OS
#cluster <- makeCluster(detectCores()) # si pas de core pur OS = saturation mémoire OS
registerDoParallel(cluster)

# Step 2: Configure trainControl object
fitControl <- trainControl(method = "repeatedcv",
                           number = 10,
                           repeats = 10,
                           classProbs = TRUE,
                           summaryFunction = twoClassSummary,
                           allowParallel = TRUE)

# search
search = "grid"

# metric
metric = "ROC"

#centrage-reduction :
preProcess <- c("center", "scale")


set.seed(seed)
tunegrid <- expand.grid(.mtry=c(1:3))

# Step 3: Develop training model
#modelRFopti <- train(x,y, method="rf",data=training ,trControl = fitControl)
modelRFopti <- train(x,y, data=training, method="rf", tuneLength=5, trControl = fitControl, search=search
, metric=metric, preProcess=preProcess,weight = wts, tuneGrid=tunegrid)
## Loading required package: pROC
## Type 'citation("pROC")' for a citation.
##
## Attaching package: 'pROC'
## The following object is masked from 'package:colorspace':
##
##     coords
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
# Step 4: De-register parallel processing cluster
stopCluster(cluster)
registerDoSEQ()
modelRFopti
## Random Forest
##
## 4306 samples
##    6 predictor
##    2 classes: 'N', 'O'
##
## Pre-processing: centered, scaled
## Resampling: Cross-Validated (10 fold, repeated 10 times)
##
```

```
## Summary of sample sizes: 3875, 3876, 3876, 3875, 3876, 3875, ...
##
## Resampling results across tuning parameters:
##
##   mtry  ROC    Sens   Spec    ROC SD  Sens SD  Spec SD
##   1     0.589  0.998  0.0032  0.0290  0.00241  0.00499
##   2     0.615  0.940  0.1303  0.0281  0.01333  0.03035
##   3     0.614  0.914  0.1715  0.0282  0.01589  0.03096
##
## ROC was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 2.
modelRFopti$resample
##           ROC      Sens       Spec     Resample
## 1   0.6223493 0.9496855 0.12389381 Fold04.Rep02
## 2   0.6271359 0.9245283 0.09734513 Fold03.Rep02
## 3   0.6504661 0.9622642 0.09821429 Fold02.Rep02
## 4   0.6227398 0.9308176 0.14285714 Fold05.Rep02
## 5   0.5924487 0.9090909 0.13274336 Fold01.Rep02
## 6   0.6156426 0.9433962 0.11504425 Fold10.Rep01
## 7   0.6060900 0.9622642 0.09821429 Fold06.Rep02
## 8   0.5864780 0.9182390 0.09821429 Fold03.Rep01
## 9   0.5775722 0.9278997 0.11607143 Fold08.Rep01
## 10  0.6056828 0.9308176 0.13392857 Fold07.Rep02
## 11  0.6236294 0.9371069 0.12389381 Fold04.Rep01
## 12  0.6280323 0.9465409 0.14285714 Fold09.Rep01
## 13  0.5962876 0.9213836 0.15044248 Fold08.Rep02
## 14  0.6544250 0.9559748 0.15178571 Fold05.Rep01
## 15  0.6373073 0.9308176 0.14159292 Fold01.Rep01
## 16  0.6466335 0.9465409 0.12500000 Fold09.Rep02
## 17  0.6117744 0.9591195 0.11504425 Fold06.Rep01
## 18  0.6134462 0.9245283 0.15178571 Fold02.Rep01
## 19  0.6173540 0.9496855 0.15044248 Fold10.Rep02
## 20  0.5917098 0.9150943 0.15929204 Fold07.Rep01
## 21  0.5789781 0.9182390 0.12389381 Fold04.Rep04
## 22  0.5677910 0.9308176 0.12389381 Fold01.Rep03
## 23  0.6456838 0.9496855 0.13274336 Fold08.Rep05
## 24  0.6354010 0.9308176 0.20353982 Fold05.Rep04
## 25  0.6714810 0.9339623 0.20353982 Fold02.Rep03
## 26  0.5731619 0.9371069 0.09734513 Fold09.Rep05
## 27  0.6030015 0.9496855 0.11607143 Fold06.Rep04
## 28  0.6280403 0.9308176 0.13274336 Fold03.Rep03
## 29  0.5887679 0.9278997 0.08928571 Fold10.Rep05
## 30  0.6177280 0.9276730 0.10714286 Fold07.Rep04
## 31  0.6388000 0.9591195 0.10714286 Fold04.Rep03
## 32  0.6397701 0.9685535 0.08849558 Fold01.Rep06
## 33  0.6294874 0.9433962 0.12389381 Fold08.Rep04
## 34  0.5858150 0.9310345 0.10714286 Fold05.Rep03
## 35  0.6319781 0.9591195 0.12389381 Fold02.Rep06
## 36  0.6315560 0.9528302 0.11607143 Fold09.Rep04
## 37  0.6578589 0.9433962 0.17699115 Fold06.Rep03
## 38  0.5620648 0.9433962 0.08928571 Fold03.Rep06
## 39  0.6152431 0.9433962 0.14285714 Fold10.Rep04
## 40  0.6099646 0.9371069 0.09821429 Fold07.Rep03
## 41  0.6012887 0.9433962 0.16071429 Fold04.Rep06
## 42  0.6367925 0.9371069 0.10619469 Fold01.Rep05
## 43  0.6010185 0.9528302 0.12389381 Fold08.Rep03
## 44  0.6328268 0.9182390 0.15044248 Fold05.Rep06
## 45  0.6035209 0.9245283 0.16071429 Fold02.Rep05
## 46  0.5812556 0.9465409 0.10714286 Fold09.Rep03
## 47  0.6099506 0.9433962 0.08035714 Fold06.Rep06
## 48  0.5953136 0.9496855 0.13274336 Fold03.Rep05
## 49  0.5890190 0.9308176 0.10714286 Fold10.Rep03
## 50  0.6295345 0.9528302 0.13392857 Fold07.Rep06
## 51  0.6331986 0.9433962 0.15178571 Fold04.Rep05
## 52  0.5512328 0.9433962 0.07079646 Fold01.Rep04
## 53  0.5949797 0.9276730 0.12389381 Fold08.Rep06
## 54  0.6188932 0.9496855 0.08035714 Fold05.Rep05
## 55  0.6253502 0.9341693 0.21238938 Fold02.Rep04
## 56  0.5826735 0.9276730 0.18750000 Fold09.Rep06
## 57  0.6449045 0.9465409 0.16814159 Fold06.Rep05
## 58  0.6269794 0.9245283 0.10714286 Fold03.Rep04
## 59  0.6851610 0.9278997 0.15929204 Fold10.Rep06
```

```
## 60  0.6018222 0.9150943 0.15178571 Fold07.Rep05
## 61  0.6199601 0.9496855 0.14285714 Fold04.Rep08
## 62  0.5825612 0.9308176 0.11607143 Fold01.Rep07
## 63  0.6233434 0.9402516 0.14285714 Fold08.Rep09
## 64  0.6621322 0.9245283 0.18750000 Fold05.Rep08
## 65  0.6282429 0.9559748 0.09821429 Fold02.Rep07
## 66  0.5827768 0.9276730 0.10619469 Fold09.Rep09
## 67  0.6032727 0.9433962 0.11504425 Fold06.Rep08
## 68  0.6322285 0.9339623 0.15929204 Fold03.Rep07
## 69  0.6453498 0.9496855 0.15929204 Fold10.Rep09
## 70  0.5873685 0.9276730 0.09734513 Fold07.Rep08
## 71  0.6155660 0.9559748 0.10714286 Fold04.Rep07
## 72  0.6566900 0.9779874 0.13274336 Fold01.Rep10
## 73  0.6149062 0.9433962 0.14285714 Fold08.Rep08
## 74  0.6444014 0.9685535 0.13392857 Fold05.Rep07
## 75  0.6011159 0.9276730 0.11504425 Fold02.Rep10
## 76  0.6508170 0.9433962 0.16071429 Fold09.Rep08
## 77  0.6000584 0.9308176 0.12389381 Fold06.Rep07
## 78  0.6056734 0.9592476 0.13392857 Fold03.Rep10
## 79  0.6133005 0.9498433 0.08928571 Fold10.Rep08
## 80  0.5882749 0.9371069 0.11607143 Fold07.Rep07
## 81  0.6094031 0.9276730 0.18750000 Fold04.Rep10
## 82  0.5924444 0.9371069 0.07142857 Fold01.Rep09
## 83  0.6077253 0.9339623 0.15044248 Fold08.Rep07
## 84  0.5702538 0.9276730 0.13274336 Fold05.Rep10
## 85  0.6122335 0.9371069 0.15929204 Fold02.Rep09
## 86  0.6385178 0.9465409 0.11504425 Fold09.Rep07
## 87  0.5690432 0.9371069 0.10619469 Fold06.Rep10
## 88  0.6235459 0.9371069 0.18584071 Fold03.Rep09
## 89  0.6555053 0.9404389 0.18584071 Fold10.Rep07
## 90  0.6021170 0.9276730 0.14285714 Fold07.Rep10
## 91  0.6998259 0.9591195 0.19642857 Fold04.Rep09
## 92  0.6366255 0.9465409 0.11504425 Fold01.Rep08
## 93  0.6291534 0.9496855 0.09734513 Fold08.Rep10
## 94  0.5714426 0.9245283 0.13392857 Fold05.Rep09
## 95  0.6190377 0.9433962 0.13274336 Fold02.Rep08
## 96  0.6773220 0.9433962 0.10714286 Fold09.Rep10
## 97  0.5923220 0.9622642 0.10619469 Fold06.Rep09
## 98  0.5776980 0.9245283 0.12389381 Fold03.Rep08
## 99  0.6127443 0.9308176 0.13392857 Fold10.Rep10
## 100 0.5831561 0.9435737 0.11607143 Fold07.Rep09
confusionMatrix.train(modelRFopti)
## Cross-Validated (10 fold, repeated 10 times) Confusion Matrix
##
## (entries are percentages of table totals)
##
##           Reference
## Prediction    N    O
##          N 69.4 22.7
##          O  4.5  3.4
# step 5 : best model = modelRFopti
# on récupère le meilleur modèle obtenu sur avec x train et y train
modelRFopti
## Random Forest
##
## 4306 samples
##    6 predictor
##    2 classes: 'N', 'O'
##
## Pre-processing: centered, scaled
## Resampling: Cross-Validated (10 fold, repeated 10 times)
##
## Summary of sample sizes: 3875, 3876, 3876, 3875, 3876, 3875, ...
##
## Resampling results across tuning parameters:
##
##   mtry  ROC    Sens   Spec    ROC SD  Sens SD  Spec SD
##   1     0.589  0.998  0.0032  0.0290  0.00241  0.00499
##   2     0.615  0.940  0.1303  0.0281  0.01333  0.03035
##   3     0.614  0.914  0.1715  0.0282  0.01589  0.03096
##
```

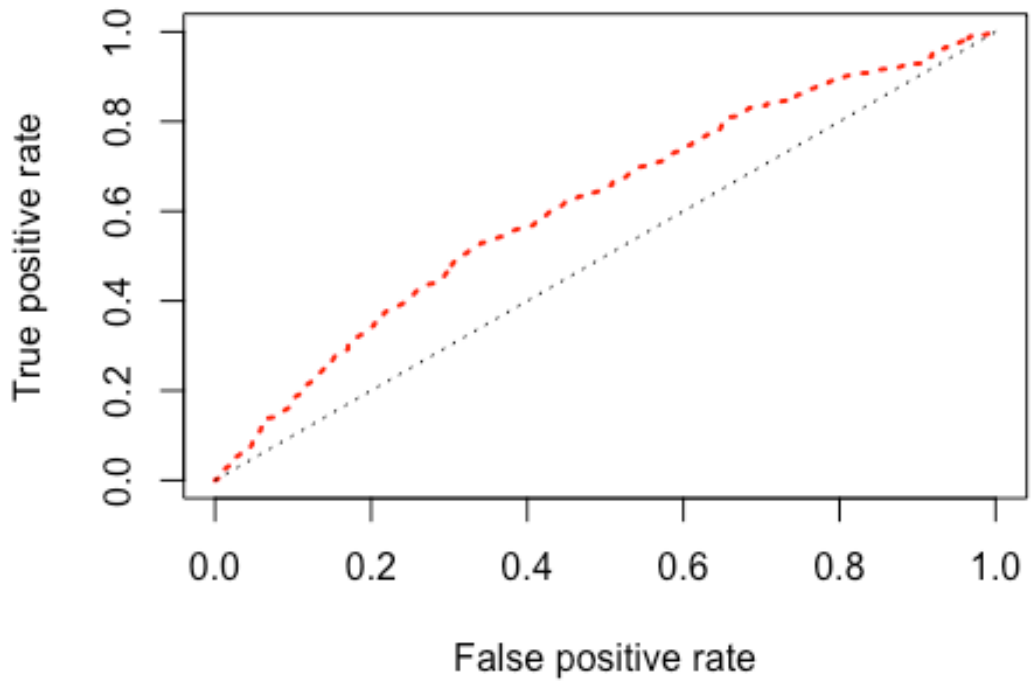
```
## ROC was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 2.
#head(with(modelRFopti, results[order(results$Kappa, decreasing=T),]))
head(with(modelRFopti, results[order(results$ROC, decreasing=T),]))
##   mtry       ROC      Sens        Spec      ROCSD     SensSD      SpecSD
## 2    2 0.6152365 0.9396116 0.130293932 0.02812289 0.01333257 0.030348296
## 3    3 0.6139289 0.9144622 0.171456226 0.02821276 0.01588606 0.030964466
## 1    1 0.5885656 0.9978940 0.003203224 0.02901371 0.00240808 0.004986773
summary(modelRFopti)
##                 Length Class      Mode
## call               7   -none-     call
## type               1   -none-     character
## predicted       4306   factor     numeric
## err.rate        1500   -none-     numeric
## confusion          6   -none-     numeric
## votes           8612   matrix     numeric
## oob.times       4306   -none-     numeric
## classes            2   -none-     character
## importance         6   -none-     numeric
## importanceSD       0   -none-     NULL
## localImportance    0   -none-     NULL
## proximity          0   -none-     NULL
## ntree              1   -none-     numeric
## mtry               1   -none-     numeric
## forest            14   -none-     list
## y               4306   factor     numeric
## test               0   -none-     NULL
## inbag              0   -none-     NULL
## xNames             6   -none-     character
## problemType        1   -none-     character
## tuneValue          1   data.frame list
## obsLevels          2   -none-     character
# step 6 : graphe model
#plot(modelRFopti)
plot(modelRFopti, xlab = "mtry - ramdomly selected predictors") # vérifier graphiquement, quel est le mtr
y = 2 qui maximise l'accuracy.
```



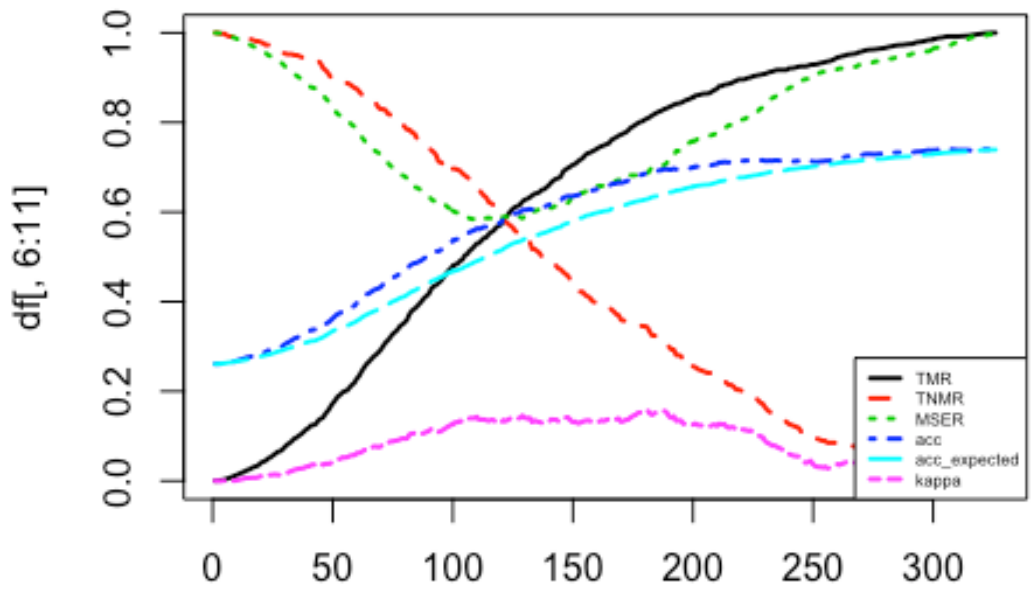

```
# step 7 : prévision sur test
rfOptiTest = predict.train(object = modelRFopti, newdata = testing[,1:6], type='raw', testing[,7])
rfOptiTest
```

```
##    [1] N N N N N N N N O N N N N N N N N N N N N N N O N N N N O N N N N N
##   [35] N N N N N N O N N N N N N N N N N N N N N O N N N N N O N N O N N N
##   [69] N N N N N N N N N N N N N N N N N N N N N N N N N O N N N N N N N N
##  [103] O N N N N N N N N N N N N N N O N O N N N N N N N N N N N N N O N N
##  [137] N N N N N O N N N N N N N N N N O N O N N N N N N N N N N N N N N N
##  [171] N N N N N N N N N N N N N N N N N N N N N N N N N N O N O N N N N N
##  [205] N N N N N N O N N N N N N N N N N N N N N O O N N N N N N N N N N N
##  [239] N N N N N N N N N N N N N N N O N N N N N N N N N N N N N N N N N N
##  [273] N O N N N N N O N N N N N N N N O N N N N N N N N N N N N N N N N N
##  [307] N N N N N N N N N N N N N N N N N N O N N N N N N N N N N N N N N N
##  [341] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [375] N N N N N N N N N N N N N N N N N N N N N N O N N N N N N N O O N N
##  [409] N N N N N N N N N N N N N O N N N N N O O N N N N N N N N N N N N N
##  [443] N N N N N N N N N O N N N N N N N N N N N N N N O N N N N N N O N N
##  [477] N N N N N O N N N N O N N N N N O N O N N N N N N O N N N N N N N N
##  [511] N N N N N N N O N N N N N N N N O N N N N N N N N N N N O O N N N N
##  [545] N N N N N N N N N N N N O N N N N N N N O N N N N N N N N N N N N N
##  [579] N N N N N N N N O N N N N N N N N N N N N N N N N N N N N N N N N O
##  [613] N N N N N N N N N N N N N N N N N O N N O O N N N N N N O N O N N N
##  [647] N N N N O N N N N N N N N N N N N N N O O N N N N N N N N O N N N N
##  [681] N O N N N N N N N N N N N N N N N N N N N N N N N N O N N N O O N
##  [715] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N O
##  [749] N O N N N N N N N N N N N N N N N N N N N N N N N N N O N N O N N N
##  [783] N N N N N N N N N N N N N N N N O N N N N N N N N N N N N N N N N N
##  [817] N N N N N N N N O N N O N N N N O N N N N N N N N N N N N N O N N N
##  [851] N N N N N N N N N N N N N N N N N N N N N N N N N O N O N N N O N N N
##  [885] N N N N N O O N N N N N N N O O N N N N N N N N N N O N N N N N N N
##  [919] N N N N N N N O N N N N N N N N N N N N N O N N N N N N N N N N N N
##  [953] N O N N N N O N N N N N N N N N N N N N O O N N N N N N N O N O N N
##  [987] N N N N N O O N O N N N N N N N N O N N N N N N O N N N N N N N N N
## [1021] N N N N N N N N N N N N O N N N N N N N O N O N N N N N O N N N N N
## [1055] N N N O N N O N N N N N N N N N N N O N N N N N N N N O N N N N N N
## [1089] N N N O N N N N N N O N N N N N N N N N N N N N N N O O N N N N N O
## [1123] N N N N N N N N N N N N N O N N N N N N N N N N N N N N N N N N N N
## [1157] N N N N N N N N N N N O N N N N N N N N N N O N N N N N N N N N N O
## [1191] N O N N N N N N N N N N N N N O N N N N N N N N N N N N N N N N N N
## [1225] N N N N N N N N N N N N O N N N N N N N N N N N N N N N N N N N N N
## [1259] N N N N N N N N N N N N N N N O N N N N N N N N N N N O N N N N N N
## [1293] N N N N N N N O N N N N O N N N N N N N N N N N N N N O N N N O N N
## [1327] N N N N N N N N O N N N N O N O N N N N N N N N N N O N N N N N N N
## [1361] N N N N N N N N N N O N N N N N N N N N N N N N N O N N N O N N N N
## [1395] N N N N N N N N N N N N N N N N N N O N N N N N N N N N N N N N N O
## [1429] N N N N N N N N N N N O N N N N N N N N N N N N N N N N O N N N N N
## [1463] O N N N N O N N N O N N N N N N N N N N N N N N N N N N N N N N N N
## [1497] N N N N N N N N N N N N N N N N N N O N N N O N N N N N O N N N N
## [1531] N N N N N N N N N O N N N N N N N N N N N N N N N N N N N N O O N
## [1565] N N N N N N N N N N N N N N N N O N N N N N N N N N N N N N N N N O
## [1599] N N N N N N N N N N N N O N N N N N N N N N N N N N N N O N N O N
## [1633] N N N N N N N O N N O N N N N N O O N N N O N O N N N N N N N N N
## [1667] O N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N O O
## [1701] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1735] N O O N N N N N N N N O N N N O N N N N O N N N N N N N N N N N N N
## [1769] O O N N N N N N N O N N N N N N N N N O N N N N N N O N O N N N N O
## [1803] N N N N N N N O N N N N N O N N N N N N N N N N N N N N N N N O N N
## [1837] N N N N N N N N
## Levels: N O
# step 8 : Confusion Matrix and Statistics === FONCTION PACKAGE CARET
CM = confusionMatrix(data=rfOptiTest, testing[,7])
CM
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    N    O
##          N 1256  422
##          O  107   59
##
##                Accuracy : 0.7131
##                  95% CI : (0.6919, 0.7337)
##     No Information Rate : 0.7392
##     P-Value [Acc > NIR] : 0.9946
##
##                   Kappa : 0.056
```

```
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.9215
##             Specificity : 0.1227
##          Pos Pred Value : 0.7485
##          Neg Pred Value : 0.3554
##              Prevalence : 0.7392
##          Detection Rate : 0.6811
##    Detection Prevalence : 0.9100
##       Balanced Accuracy : 0.5221
##
##        'Positive' Class : N
##
# F-mesure
CM$byClass[3]
## Pos Pred Value
##      0.7485101
# step 9 : confusion matrix: model predicting classes of test data
#table.SVM=table(rfOptiTest$y,testing[,7]) #
table.SVM=table(testing[,7],rfOptiTest)
table.SVM
##    rfOptiTest
##        N    O
##   N 1256  107
##   O  422   59
TxBcSVM=(sum(diag(table.SVM))/sum(table.SVM))*100
TxBcSVM
## [1] 71.31236
# [1] 72.80

# prediction probabilities of test data classes
probs <- predict.train(object = modelRFopti, newdata = testing[,1:6], type='prob')[,1]
isPositiveClass <- testing[,7] == 'N' # for a ROC curve there is a positive class (true match rate...) -
defining that class here
pred <- prediction(probs, isPositiveClass)
perf <- performance(pred, 'tpr', 'fpr')

# plot: either
plot(perf, col=2, lwd=2,lty=3)
segments(0,0,1,1,lty=3)
```

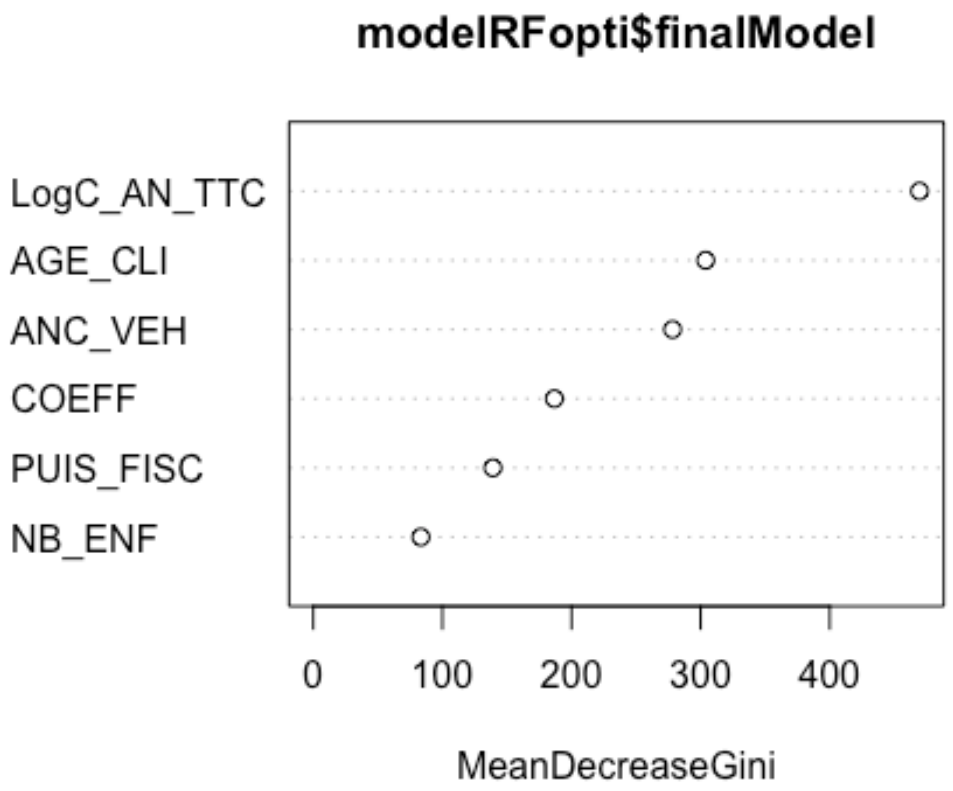


```
# or
#with(attributes(perf), plot(x=x.values[[1]], y=y.values[[1]], type='l'))


# step 10 : some metrics: AUC (area under curve), EER (equal error rate), MSER (minimum squared error rat
e), Cohen's Kappa etc.
AUC <- attributes(performance(pred, 'auc'))$y.value[[1]] # area under curve
AUC
## [1] 0.6143207
# [1] 0.6451

df <- with(attributes(pred), data.frame(cutoffs=cutoffs[[1]], tp=tp[[1]], fn=fn[[1]], tn=tn[[1]], fp=fp[[
1]], TMR=tp[[1]]/(tp[[1]]+fn[[1]]), TNMR=tn[[1]]/(tn[[1]]+fp[[1]])))
df$MSER <- with(df, sqrt((1-TMR)**2+(1-TNMR)**2))
MSER <- with(df, sqrt(TMR**2+TNMR**2)) # sqrt of minimum squared error rate = eucl. distance to point TMR
=TNMR=1
i_eer <- with(df, which.min(abs(TMR-TNMR)))
EER <- with(df[i_eer,], mean(c(TMR,TNMR))) # equal error rate: mean would not be required when using ROCR
as it's always exact
df$acc <- with(df, (tp + tn) / length(isPositiveClass)) # observed accuracy
df$acc_expected <- with(df, sum(isPositiveClass) * (tp+fp) / length(isPositiveClass) + sum(!isPositiveCla
ss) * (tn+fn) / length(isPositiveClass)) / length(isPositiveClass) # expected accuracy
df$kappa <- with(df, (acc - acc_expected) / (1 - acc_expected)) # cohen's kappa

# graphical representation
matplot(df[,6:11], lty=1:6, lwd=2, type='l'); legend('bottomright', legend=names(df[,6:11]), lty=1:6, lwd
=2, col=1:6,cex=0.45)
```




```
# characteristics for settings with best EER, MSER etc.
df[i_eer,] # curve point for equal error rate
##     cutoffs  tp  fn  tn  fp       TMR      TNMR      MSER       acc
## 121   0.756 792 571 283 198 0.5810712 0.5883576 0.5873252 0.5829718
##     acc_expected     kappa
## 121    0.5176383 0.1354451
df[order(df$acc, decreasing=T)[[1]],] # curve point for max accuracy
##     cutoffs   tp fn tn  fp       TMR       TNMR      MSER       acc
## 306   0.334 1352 11 15 466 0.9919296 0.03118503 0.9688486 0.7413232
##     acc_expected     kappa
## 306      0.73241 0.0333093
df[order(df$kappa, decreasing=T)[[1]],] # curve point for max kappa
```

```
##     cutoffs   tp  fn  tn  fp       TMR      TNMR      MSER       acc
## 181   0.636 1103 260 165 316 0.8092443 0.3430353 0.6840982 0.6876356
##     acc_expected     kappa
## 181    0.6289149 0.1582404
df[order(df$MSER, decreasing=F)[[1]],] # curve point for minimum squared error rate
##     cutoffs  tp  fn  tn  fp       TMR      TNMR      MSER      acc
## 110   0.778 722 641 317 164 0.5297139 0.6590437 0.5808789 0.563449
##     acc_expected    kappa
## 110    0.4906621 0.142905
# step 11 : importance des variables
varImpPlot(modelRFopti$finalModel)
```




```
varImp(modelRFopti)
## rf variable importance
##
##             Overall
## LogC_AN_TTC  100.00
## AGE_CLI       57.11
## ANC_VEH       50.47
## COEFF         26.77
## PUIS_FISC     14.44
## NB_ENF         0.00
# timer
toc <-  Sys.time() -tic
print(toc)
## Time difference of 5.43807 mins
# AUTOMATIC GRID SEARCH pour RF
# http://machinelearningmastery.com/tune-machine-learning-algorithms-in-r/
# https://geekoverdose.wordpress.com/2014/07/25/svm-classification-example-with-performance-measures-usin
g-r-caret/


#*****************************************************************************************************
*****************************************************************************************
#*****************************************************************************************************
*****************************************************************************************
#*****************************************************************************************************
*****************************************************************************************
```

SVM_prepadata.R

lepicaut

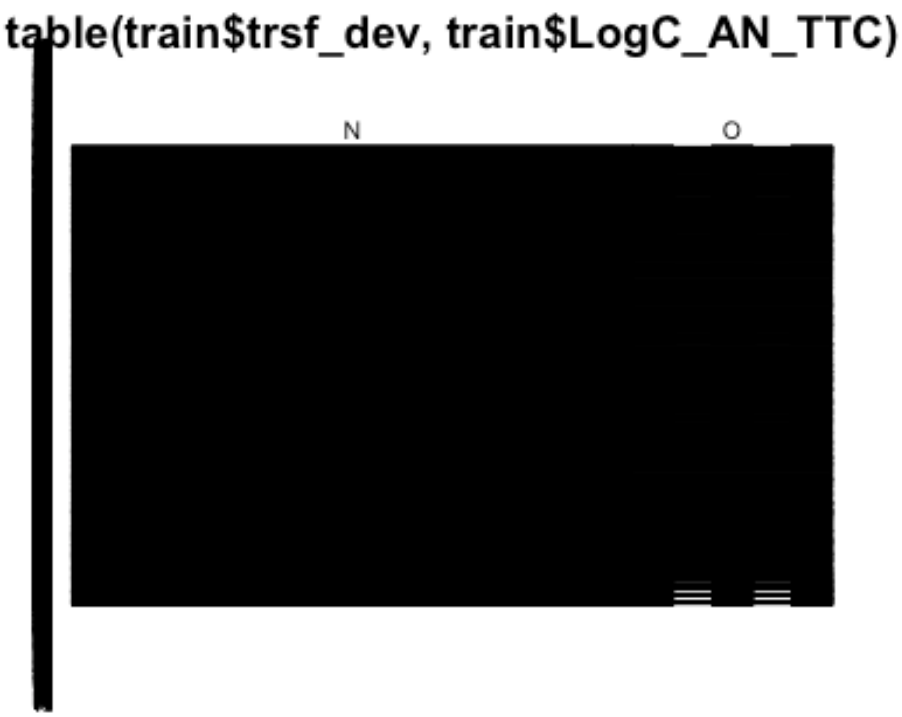
Sun Oct 8 11:38:18 2017

```
#==========================================================================================
======
#
#
#                                        * STOP : PREPARATION DES DONNEES *
#
#
#
#==========================================================================================
======

# TIC & TOC
tic <- Sys.time()

#==============================================
# PARTIE 2. : Support Vector Machine - SVM
#==============================================


#***************
# PARTIE 2.1 : SVM et donnees d'apprentissage
#***************

#*********************************************************************
# 11 juin 2017
# perso.mines-paristech.fr/fabien.moutarde/ES_MachineLearning/
#*********************************************************************

# recodage des niveaux facteur "O", "N" de trsf_dev "1", "0"
dataSVM=dq2norm_y
str(dataSVM)

## 'data.frame':    6150 obs. of  7 variables:
##  $ LogC_AN_TTC: num  -0.459 -0.559 0.733 -1.352 0.716 ...
##  $ COEFF      : num  -1.06 -0.051 -1.06 0.232 1.16 ...
##  $ PUIS_FISC  : num  -0.289 -1.165 1.462 -0.727 -0.289 ...
##  $ AGE_CLI    : num  -1.139 -1.139 0.772 -0.629 -0.82 ...
##  $ NB_ENF     : num  -0.724 -0.724 -0.724 -0.724 -0.724 ...
##  $ ANC_VEH    : num  -0.887 -0.403 -0.08 0.404 0.243 ...
##  $ trsf_dev   : Factor w/ 2 levels "N","O": 1 2 1 1 1 2 2 2 1 2 ...

# levels(dataSVM$trsf_dev) <- c("0","1")
# levels(dataSVM$trsf_dev) <- c("-1","1")

# sanity check
# head(dataSVM$trsf_dev)
# summary(dataSVM)
# summary(dq2norm_y)

df31=dataSVM[sample(1:nrow(dataSVM),6150),]
#df31=dataSVM[sample(1:nrow(dataSVM),614),] #==== pour la partie 2.4 sans calcul parallele


#*********************************************************************
#        reprise dataset df31
#        CARET : calcule parallel - automatic grid search
```

```r
#*********************************************************************

set.seed(1000) # Set Seed so that same sample can be reproduced in future also
# Now Selecting 70% of data as sample from total 'n' rows of the data
sample <- sample.int(n = nrow(df31), size = floor(.70*nrow(df31)), replace = F)
train <- df31[sample, ]
test  <- df31[-sample, ]
summary(df31)

##   LogC_AN_TTC           COEFF             PUIS_FISC
##  Min.   :-3.10793   Min.   :-1.26197   Min.   :-1.164799
##  1st Qu.:-0.67005   1st Qu.:-1.06014   1st Qu.:-0.726978
##  Median : 0.01530   Median :-0.05098   Median :-0.289157
##  Mean   :-0.01044   Mean   :-0.02098   Mean   : 0.007209
##  3rd Qu.: 0.65990   3rd Qu.: 1.02882   3rd Qu.: 0.148665
##  Max.   : 4.05752   Max.   : 5.19665   Max.   :18.099338
##     AGE_CLI             NB_ENF             ANC_VEH         trsf_dev
##  Min.   :-1.20266   Min.   :-0.724301   Min.   :-1.53296   N:4543
##  1st Qu.:-0.88409   1st Qu.:-0.724301   1st Qu.:-0.72575   O:1607
##  Median :-0.31066   Median :-0.724301   Median :-0.07999
##  Mean   : 0.01211   Mean   : 0.004633   Mean   :-0.01998
##  3rd Qu.: 0.77247   3rd Qu.: 0.914502   3rd Qu.: 0.56578
##  Max.   : 3.38475   Max.   : 5.830911   Max.   : 7.34629

train=na.omit(train)
test=na.omit(test)
summary(train)

##   LogC_AN_TTC            COEFF             PUIS_FISC
##  Min.   :-3.107934   Min.   :-1.26197   Min.   :-1.16480
##  1st Qu.:-0.669108   1st Qu.:-1.06014   1st Qu.:-0.72698
##  Median :-0.007915   Median :-0.05098   Median :-0.28916
##  Mean   :-0.012692   Mean   :-0.02346   Mean   : 0.02133
##  3rd Qu.: 0.649733   3rd Qu.: 0.95818   3rd Qu.: 0.14866
##  Max.   : 4.057522   Max.   : 3.29943   Max.   :18.09934
##     AGE_CLI             NB_ENF             ANC_VEH         trsf_dev
##  Min.   :-1.20266   Min.   :-0.72430   Min.   :-1.53296   N:3178
##  1st Qu.:-0.88409   1st Qu.:-0.72430   1st Qu.:-0.72575   O:1127
##  Median :-0.31066   Median :-0.72430   Median :-0.07999
##  Mean   : 0.01645   Mean   : 0.01078   Mean   :-0.01815
##  3rd Qu.: 0.77247   3rd Qu.: 0.91450   3rd Qu.: 0.56578
##  Max.   : 3.25732   Max.   : 5.83091   Max.   : 7.34629

summary(test)

##   LogC_AN_TTC            COEFF             PUIS_FISC
##  Min.   :-2.926394   Min.   :-1.26197   Min.   :-1.16480
##  1st Qu.:-0.681728   1st Qu.:-1.06014   1st Qu.:-0.72698
##  Median : 0.049579   Median :-0.05098   Median :-0.28916
##  Mean   :-0.005202   Mean   :-0.01521   Mean   :-0.02575
##  3rd Qu.: 0.680056   3rd Qu.: 1.16001   3rd Qu.: 0.14866
##  Max.   : 2.732643   Max.   : 5.19665   Max.   : 9.34291
##     AGE_CLI              NB_ENF             ANC_VEH         trsf_dev
##  Min.   :-1.202657   Min.   :-0.724301   Min.   :-1.53296   N:1365
##  1st Qu.:-0.884088   1st Qu.:-0.724301   1st Qu.:-0.72575   O: 480
##  Median :-0.310663   Median :-0.724301   Median :-0.07999
##  Mean   : 0.002002   Mean   :-0.009712   Mean   :-0.02425
##  3rd Qu.: 0.772474   3rd Qu.: 0.914502   3rd Qu.: 0.56578
##  Max.   : 3.384745   Max.   : 4.192108   Max.   : 5.40900

#***************************
#  poids des N / O en %
#  andruccioli.u-bordeaux4.fr/pages/statistiques/complements_corrections/R_et_traitement_tableaux_de_cont
ingence.pdf
table(trsf_dev)

## trsf_dev
##      N      O
## 547668 181006

freqabs=round(table(trsf_dev)/sum(table(trsf_dev))*100,2)
freqabs
```

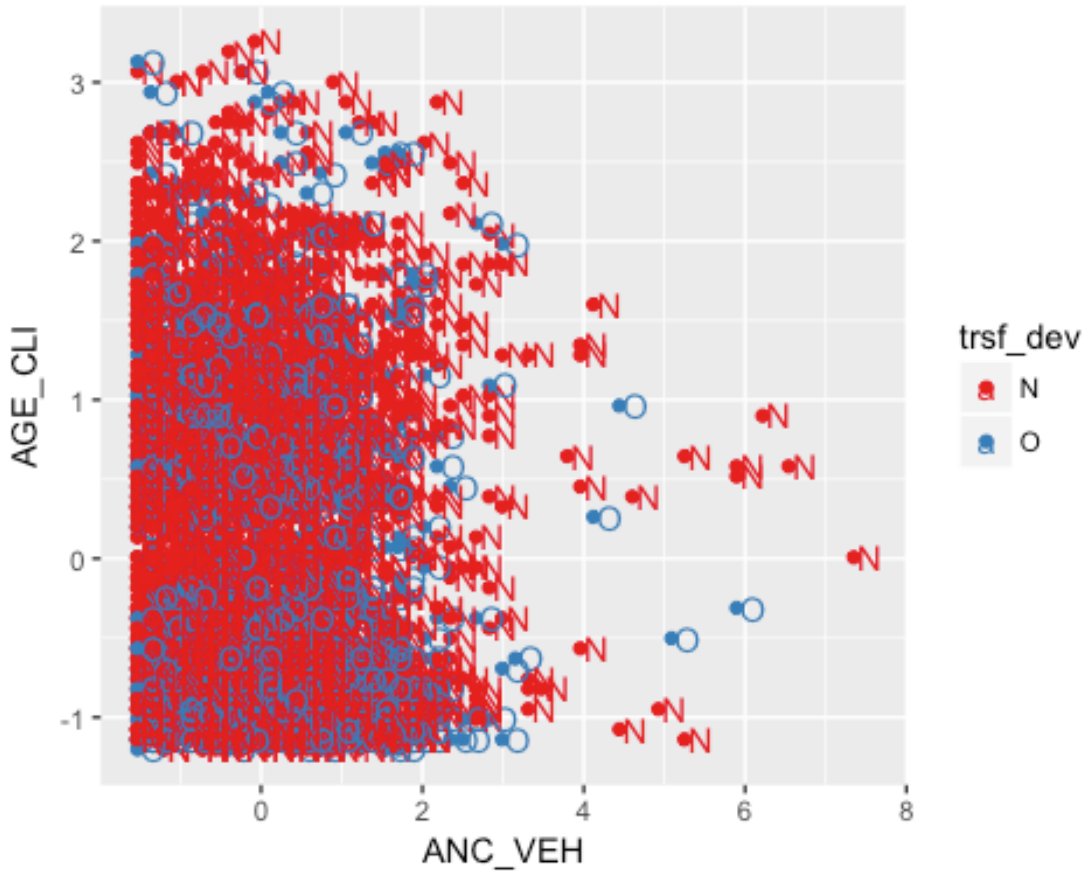
```
## trsf_dev
##     N     O
## 75.16 24.84

#trsf_dev
#N     O
#75.16 24.84
#**************************

# poids sur l'echantillon apprentissage
table(train$trsf_dev)

##
##    N    O
## 3178 1127

# N    O
# 3156 1149

freqabs=round(table(train$trsf_dev)/sum(table(train$trsf_dev)),2)
freqabs

##
##    N    O
## 0.74 0.26

#freqabs
# N    O
# 0.73 0.27

freqabs=round(table(train$trsf_dev)/sum(table(train$trsf_dev))*100,2)
freqabs

##
##     N     O
## 73.82 26.18

#freqabs
# N     O
# 73.31 26.69

# pris en compte du poids des classes sur dataset = apprentissage
summary(train$trsf_dev)

##    N    O
## 3178 1127

# N    O
# 3156 1149

wts=100/table(train$trsf_dev)
wts

##
##          N          O
## 0.03146633 0.08873114

#   N          O
#  0.03168568 0.08703220

# STOP
#*****************************************************************************************************
*****************************************************************************************


#==========================================
# TODO : graphe pairwise ?
#==========================================
# chrisalbon / == PROCEDURE DATAVIZ GGPLOT R

#*****************************************************************
# fxjollois.github.io/analyse-donnees
```

```r
# A l aide de la fonction pairs() visualiser sur l ensemble dapprentissage
# les nuages de points croises des variables quantitatives,
# en appliquant une coloration qui depend de la classe.
# Les variables quantitatives vous paraissent-elles suffisantes pour predire le niveau des eleves ?

# fxjollois.github.io/analyse-donnees

plot(table(train$trsf_dev,train$LogC_AN_TTC))
```



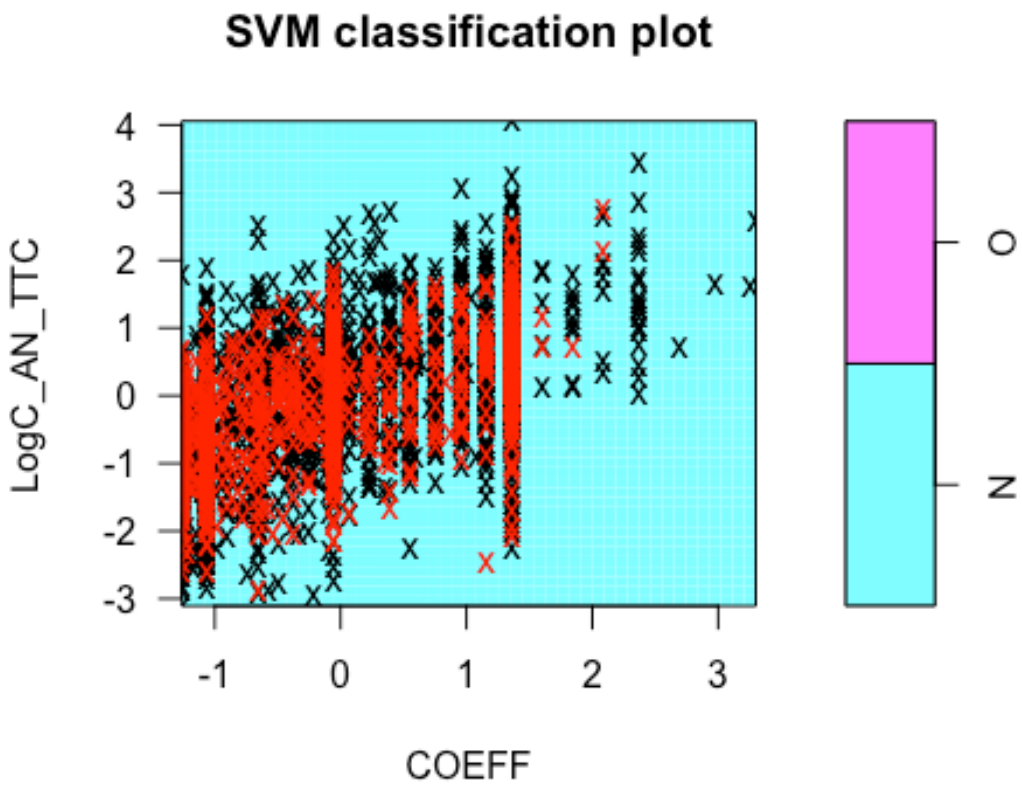

```r
# svm-tutorial.com/2014/10/support-vector-regression-r/
# DataMiningTP5_emilie_kaufman.pdf
# dm.pdf

# chrisalbon
# chrisalbon.com/r-stats/scatterplot-with-point-labels

# hypo1 :DATAVIZ only avec train
# hypo2 : suite à la procédure importance des variables du modèle RF (critère MeanDecreaseGini)
# 2 Xi (+) importants retenus LogC_AN_TTC et AGE_CLI

# load the ggplot2 package
library(ggplot2)


#*******  avec les 1e et 2e Xi (RF importance)

# reset the graphing device
# dev.off()

# create the scatterplot
sps <- ggplot(train, aes(x = AGE_CLI , y = LogC_AN_TTC, colour = trsf_dev)) +
  geom_point() +
  scale_colour_brewer(palette = "Set1")


# add a layer of point labels to the scatterplot, offset of .2 of the x axis
sps + geom_text(aes(x = AGE_CLI + .2, label = trsf_dev))
```

```
# lecture résultats graphiques :
# les classes étiquetées [N] et [O] sont mélangées,
# le problème de classification binaire n'est pas linéairement séparable
# le classifieur SVM linéaire ne permettra pas de séparer nos données d'apprentissage.

#*******  avec les 2e et 3e Xi (RF importance)

# reset the graphing device
# dev.off()

# create the scatterplot
sps <- ggplot(train, aes(x = ANC_VEH , y = AGE_CLI , colour = trsf_dev)) +
  geom_point() +
  scale_colour_brewer(palette = "Set1")

# add a layer of point labels to the scatterplot, offset of .2 of the x axis
sps + geom_text(aes(x = ANC_VEH + .2, label = trsf_dev))
```



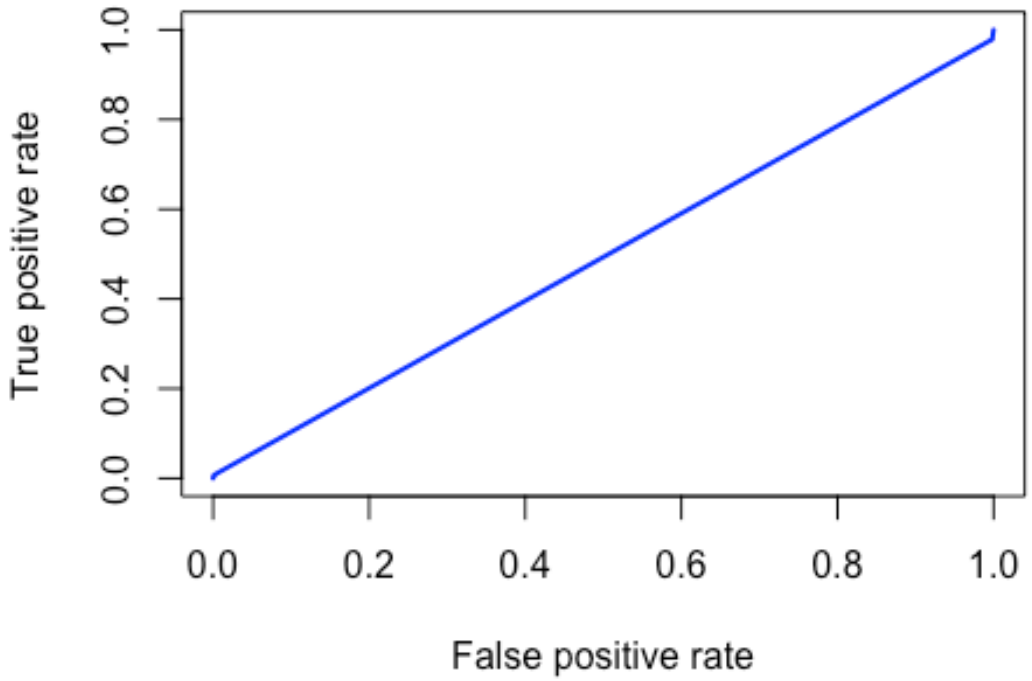

```
# lecture résultats graphiques :
# les classes étiquetées [N] et [O] sont mélangées,
# le problème de classification binaire n'est pas linéairement séparable
# le classifieur SVM linéaire ne permettra pas de séparer nos données d'apprentissage.


#==================================================================================
#  TODO : svm grid search - les questions à se poser
#  q1: problème de classification : bi-classe / multiclasse
#  q2: séparateur linéaire,
#  q3: séparateur non linéaire
#  q4 : le problème est-il séparable ou imparfaitement séparable
#*******
#  q4 : comment faire grid search
#  q5 : comment choisir le meilleur coût cost ? vecteur de valeurs coût ? pourquoi
#  q6 : comment choisir le meilleur noyaux choix du noyau
#  q7 : comment choisir la valeur du gamma du noyau
#  q8 : noyau ? polynome : comment chosir le degre du polynome ?
#==================================================================================

toc <-  Sys.time() -tic
print(toc)

## Time difference of 17.23967 secs

#*******
# PARTIE 2.2 : 1e iteration : initialisation du modèle avec radial
#*******

tic <- Sys.time()
library(e1071)

# justification :
# (1) problème classification binaire non linéairement séparable => on retire kernel linear,
# (2) lors du modèle régression logistique : le modèle n'est pas bien spécifié avec ce dataset, on retire
le noyau sigmoid,
# (3) on retient le kernel gaussien vs kernel polynomial ==> pour démarrer le kernel gaussien (radial-RBF
), "spectre d'utilisation large pour de nombreux problèmes"
# (4) avec ce kernel RBF : les auteurs proposent les valeurs de paramètes : cost=1000, gamma=1
```

```
#modelsvm=svm(trsf_dev~.,data=train,type='C-classification', kernel='linear',probability=TRUE, cost=1)
# http://ugrad.stat.ubc.ca/R/library/e1071/html/plot.svm.html


#modelsvm=svm(trsf_dev~.,data=train, kernel='sigmoid', type='C-classification', scale=TRUE, probability=T
RUE, cost=1000, gamma=1, class.weights=c("-1"=0.729, "1"=0.271))
modelsvm=svm(trsf_dev~.,data=train, kernel='radial', type='C-classification', scale=TRUE, probability=TRU
E, cost=1000, gamma=1000000, class.weights=wts)
plot(modelsvm, train, LogC_AN_TTC ~ COEFF,
     slice = list(PUIS_FISC = 3, AGE_CLI = 4, AGE_CLI = 5, NB_ENF = 6, ANC_VEH = 7), fill = TRUE, grid =
50)
```



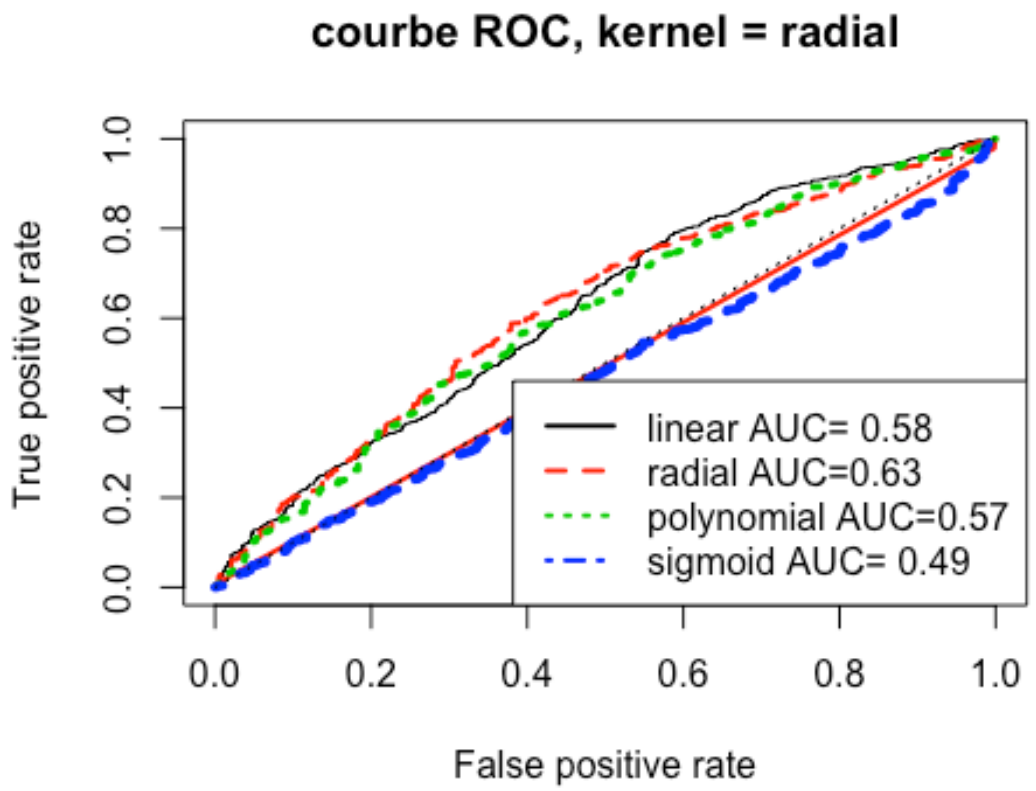

```
#Prédiction et matrice de confusion
pred.svm=predict(modelsvm,newdata=test)
table.SVM=table(pred.svm,test$trsf_dev)
table.SVM

##
## pred.svm    N    O
##        N 1363  472
##        O    2    8

TxBcSVM=(sum(diag(table.SVM))/sum(table.SVM))*100
TxBcSVM

## [1] 74.30894

# [1] 74.03794

#Erreur en test
ErreurSVM=100-TxBcSVM
ErreurSVM

## [1] 25.69106

# [1] 0.4920201

#Courbe ROC
```

```r
library(ROCR)
ROCsvm=predict(modelsvm,newdata=test,decision.values = T)
a=attr(ROCsvm,"decision.values")
predsvm=prediction(a,test$trsf_dev)
perfSVM=performance(predsvm,"tpr","fpr")
AUC.SVM=performance(predsvm,"auc")
plot(perfSVM,col=4,lwd=2)
```




```r
AUC.SVM@y.values[[1]]

## [1] 0.4934051

print(AUC.SVM@y.values[[1]])

## [1] 0.4934051

#[1] 0.4920201

plot(perfSVM,col=2,lwd=2, main="courbe ROC, kernel = radial", cex=0.3)
segments(0,0,1,1,lty=3)


toc <-  Sys.time() -tic
print(toc)

## Time difference of 13.94307 secs

# Time difference of 32.16972 secs


#*******
# PARTIE 2.3 : 2e iteration : grid_search "à la main " - differents noyaux comparés > trouver l'AUC max
#*******
tic <- Sys.time()
library(e1071)

# apprentissage (CONSTRUCTION CLASSIQUE DU MODELE)

# LINEAR
model1=svm(trsf_dev~LogC_AN_TTC+PUIS_FISC+COEFF+AGE_CLI+NB_ENF+ANC_VEH,data=train,kernel="linear",type='C
```

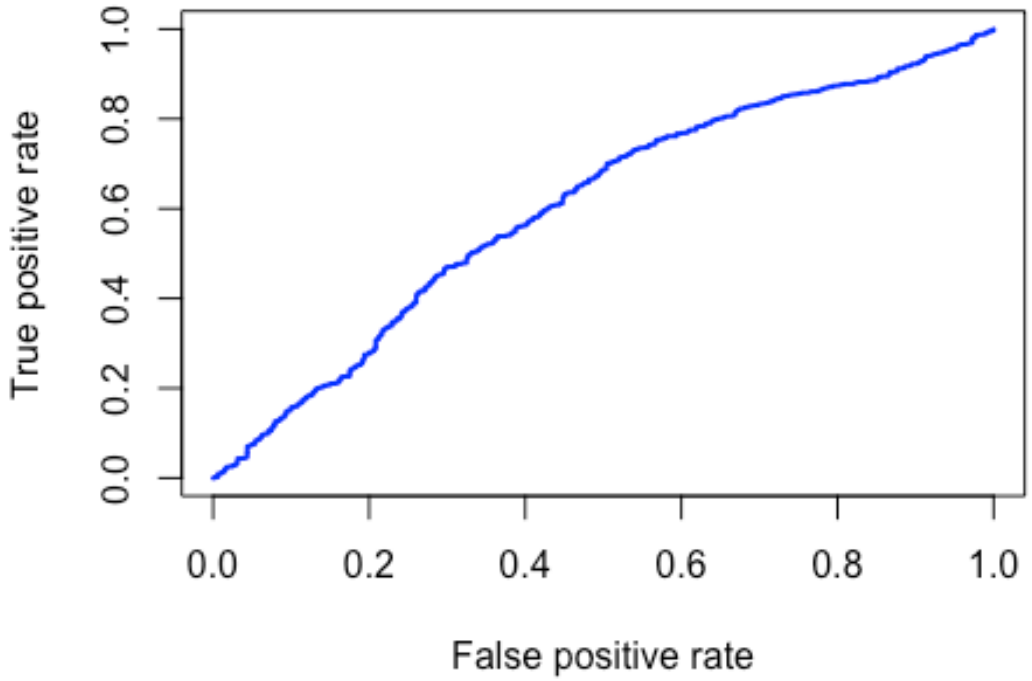
```
-classification',scale=TRUE, probability=TRUE, cost=1000, class.weights=wts)
ROCsvm1=predict(model1,newdata=test,decision.values = T)
a1=attr(ROCsvm1,"decision.values")
predsvm1=prediction(a1,test$trsf_dev=="N")
perfSVM1=performance(predsvm1,"tpr","fpr")
AUC.SVM1=performance(predsvm1,"auc")
AUC.SVM1@y.values[[1]]

## [1] 0.6219918

# [1] 0.5892

# RADIAL
model2=svm(trsf_dev~.,data=train,kernel="radial",type='C-classification',scale=TRUE, probability=TRUE, co
st=1000, gamma=0.17, class.weights=wts)
ROCsvm2=predict(model2,newdata=test,decision.values = T)
a2=attr(ROCsvm2,"decision.values")
predsvm2=prediction(a2,test$trsf_dev=="N")
perfSVM2=performance(predsvm2,"tpr","fpr")
AUC.SVM2=performance(predsvm2,"auc")
AUC.SVM2@y.values[[1]]

## [1] 0.625148

# [1] 0.6319

# POLYNOMIAL
model3=svm(trsf_dev~LogC_AN_TTC+PUIS_FISC+COEFF+AGE_CLI+NB_ENF+ANC_VEH,data=train,kernel="polynomial",typ
e='C-classification',scale=TRUE, probability=TRUE, degree=2 ,cost=1000, gamma=0.17, class.weights=wts)
ROCsvm3=predict(model3,newdata=test,decision.values = T)
a3=attr(ROCsvm3,"decision.values")
predsvm3=prediction(a3,test$trsf_dev=="N")
perfSVM3=performance(predsvm3,"tpr","fpr")
AUC.SVM3=performance(predsvm3,"auc")
AUC.SVM3@y.values[[1]]

## [1] 0.6087469

# 1] 0.5780

# SIGMOID
model4=svm(trsf_dev~LogC_AN_TTC+PUIS_FISC+COEFF+AGE_CLI+NB_ENF+ANC_VEH,data=train,kernel="sigmoid",type='
C-classification',scale=TRUE, probability=TRUE, cost=1000, gamma=0.17, class.weights=wts)
ROCsvm4=predict(model4,newdata=test,decision.values = T)
a4=attr(ROCsvm4,"decision.values")
predsvm4=prediction(a4,test$trsf_dev=="N")
perfSVM4=performance(predsvm4,"tpr","fpr")
AUC.SVM4=performance(predsvm4,"auc")
AUC.SVM4@y.values[[1]]

## [1] 0.4747985

# [1] 0.4957

# les 4 courbes AUC - 1 seul graphique
plot(perfSVM1,col=1,lwd=1,lty=1,add=TRUE)
plot(perfSVM2,col=2,lwd=2,lty=2,add=TRUE)
plot(perfSVM3,col=3,lwd=3,lty=3,add=TRUE)
plot(perfSVM4,col=4,lwd=4,lty=4,add=TRUE)
segments(0,0,1,1,lty=3)
legend("bottomright",c("linear AUC= 0.58","radial AUC=0.63","polynomial AUC=0.57"
                       ,"sigmoid AUC= 0.49"),col=c(1,2,3,4),lwd=2,lty=c(1,2,3,4))
```



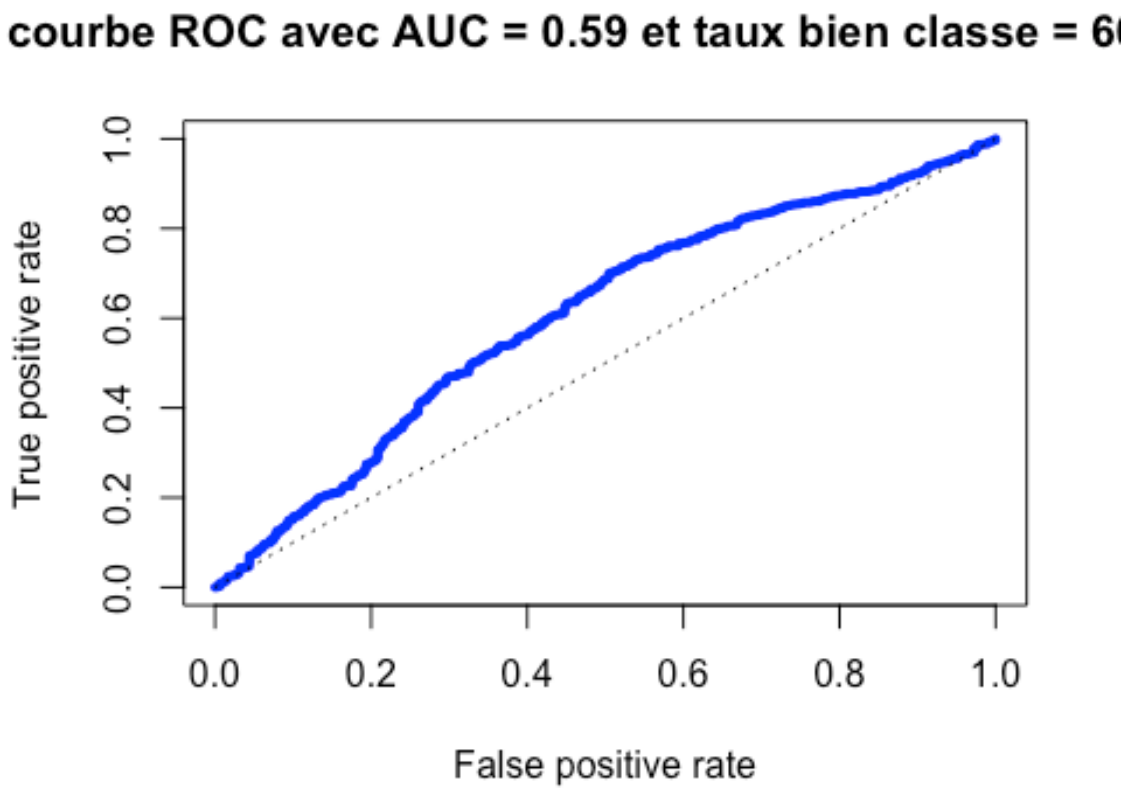

```
toc <-  Sys.time() -tic
print(toc)

## Time difference of 55.55205 secs

# analyse des résultats : gridsearch à la main

#*******
# pour un dataset 6150 lignes  x 7 Xi
# sur le critère de l'AUC (pouvoir discriminant d'un classifieur)
# le classifieur radial, avec un AUC = 0.63% semble être le meilleur classifieur.
# en revanche, résultat à modèrer car les taux de bien classés et AUC varient en sens inverse,
# le résultat peut être modifié par une taille plus importante du dataset.
# à confirmer avec la procédure de automatique de grid search tune() du package caret


#******
# maxAUC --> RBF ?
# freakonometrics.hypotheses.org/48818
#******
library(plotmo)

## Warning: package 'plotmo' was built under R version 3.4.1

## Loading required package: plotrix

## Warning: package 'plotrix' was built under R version 3.4.1

##
## Attaching package: 'plotrix'

## The following object is masked from 'package:gplots':
##
##     plotCI

## Loading required package: TeachingDemos

#Modèle
tic <- Sys.time()
```

```r
modelRBF=svm(trsf_dev~LogC_AN_TTC+PUIS_FISC+COEFF+AGE_CLI+NB_ENF+ANC_VEH,data=train,
             kernel="radial",
             type='C-classification',scale=TRUE,
             probability=TRUE,
             cost=1000,
             gamma=1,
             #class.weights=c("-1"=0.729, "1"=0.271)
             class.weights = wts
)

summary(modelRBF)

##
## Call:
## svm(formula = trsf_dev ~ LogC_AN_TTC + PUIS_FISC + COEFF + AGE_CLI +
##     NB_ENF + ANC_VEH, data = train, kernel = "radial", type = "C-classification",
##     probability = TRUE, cost = 1000, gamma = 1, class.weights = wts,
##     scale = TRUE)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  1000
##       gamma:  1
##
## Number of Support Vectors:  2877
##
##  ( 1941 936 )
##
##
## Number of Classes:  2
##
## Levels:
##  N O

print(modelRBF)

##
## Call:
## svm(formula = trsf_dev ~ LogC_AN_TTC + PUIS_FISC + COEFF + AGE_CLI +
##     NB_ENF + ANC_VEH, data = train, kernel = "radial", type = "C-classification",
##     probability = TRUE, cost = 1000, gamma = 1, class.weights = wts,
##     scale = TRUE)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  1000
##       gamma:  1
##
## Number of Support Vectors:  2877

#Prédiction et matrice de confusion
pred.svm=predict(modelRBF,newdata=test)
table.SVM=table(pred.svm,test$trsf_dev)
table.SVM

##
## pred.svm   N   O
##        N 875 223
##        O 490 257

TxBcSVM=(sum(diag(table.SVM))/sum(table.SVM))*100
TxBcSVM

## [1] 61.35501

# [1] 60.27
```

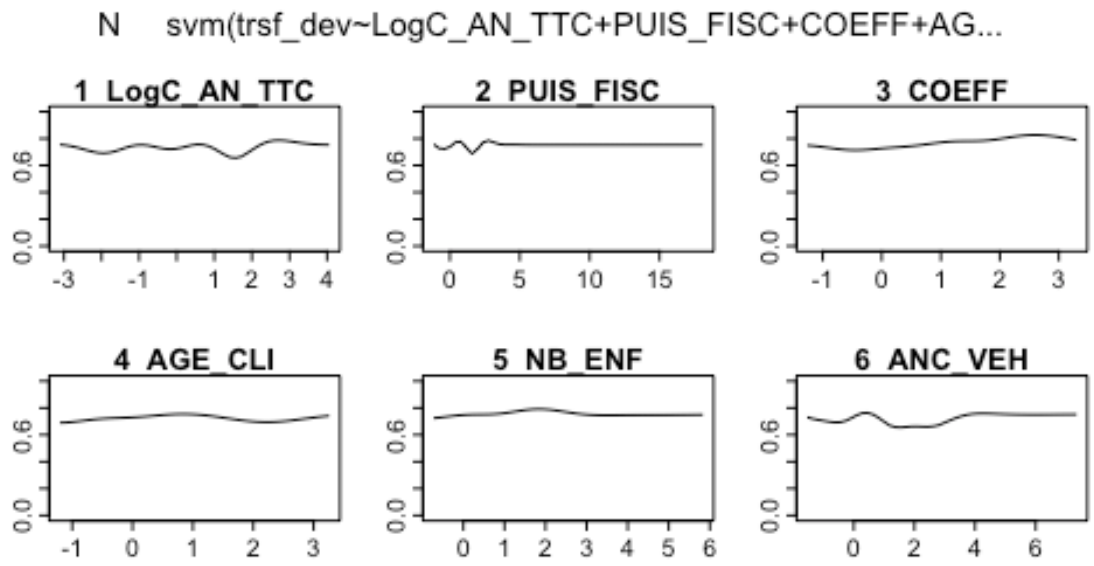
```r
#plot(cmdscale(dist(MarTest[,-14])),col=as.integer(MarTest[,14]))
ErreurSVM=100-TxBcSVM
ErreurSVM

## [1] 38.64499

# [1] 39.729


#Courbe ROC
library(ROCR)
ROCsvm=predict(modelRBF,newdata=test,decision.values = T)
a=attr(ROCsvm,"decision.values")
predsvm=prediction(a,test$trsf_dev=='N')
perfSVM=performance(predsvm,"tpr","fpr")
AUC.SVM=performance(predsvm,"auc")
plot(perfSVM,col=4,lwd=2)
```



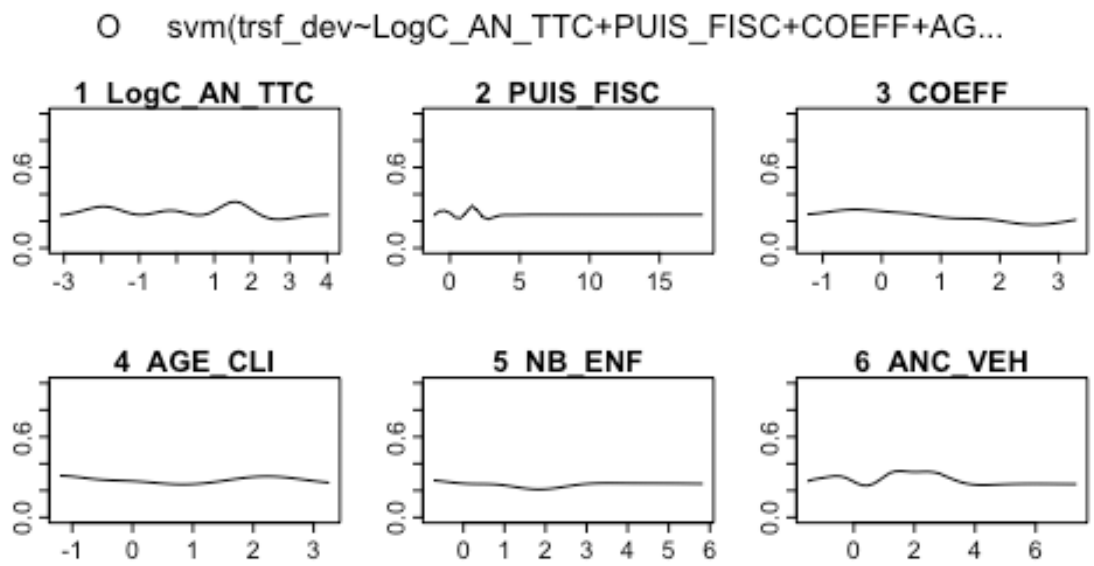

```r
AUC.SVM@y.values[[1]]

## [1] 0.6044017

# [1] 0.5980

plot(perfSVM,col=4,lwd=4, main="courbe ROC avec AUC = 0.59 et taux bien classe = 60.27")
segments(0,0,1,1,lty=3)
```



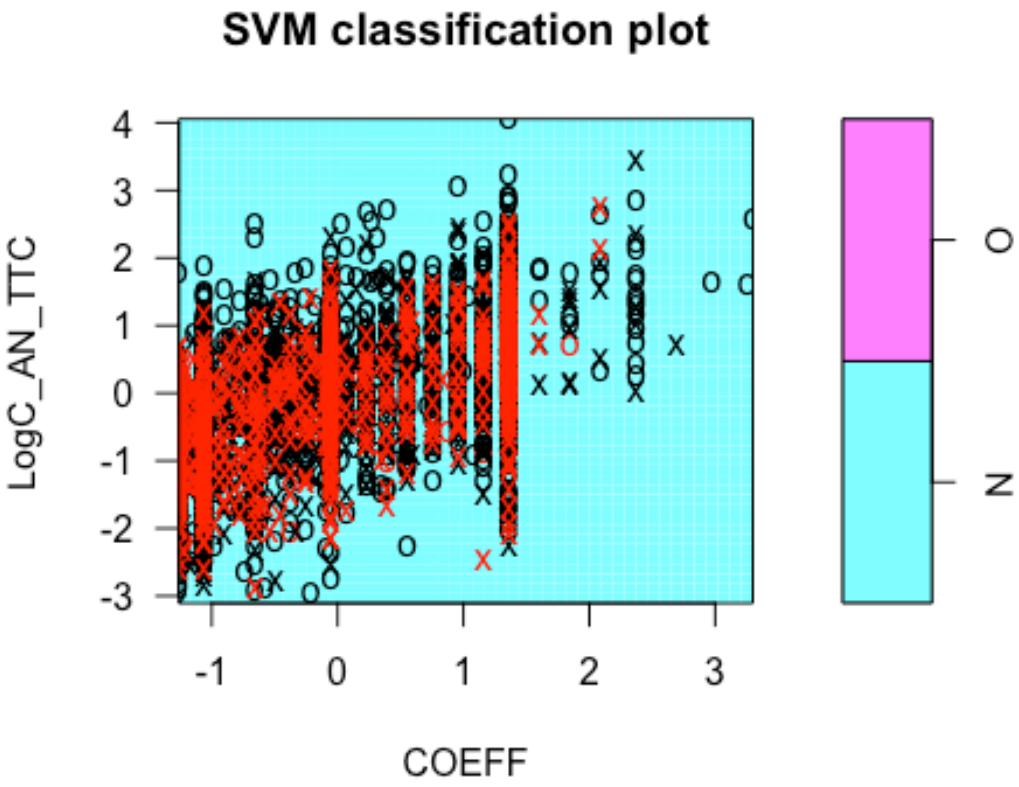

```r
# milbo.org/doc/plotmo-notes.pdf
par(mfrow=c(1,1))
plotmo(modelRBF, predict.probability=TRUE, nresponse="N", ylim=c(0,1))
```




```r
par(mfrow=c(1,1))
plotmo(modelRBF, predict.probability=TRUE, nresponse="O", ylim=c(0,1))
```



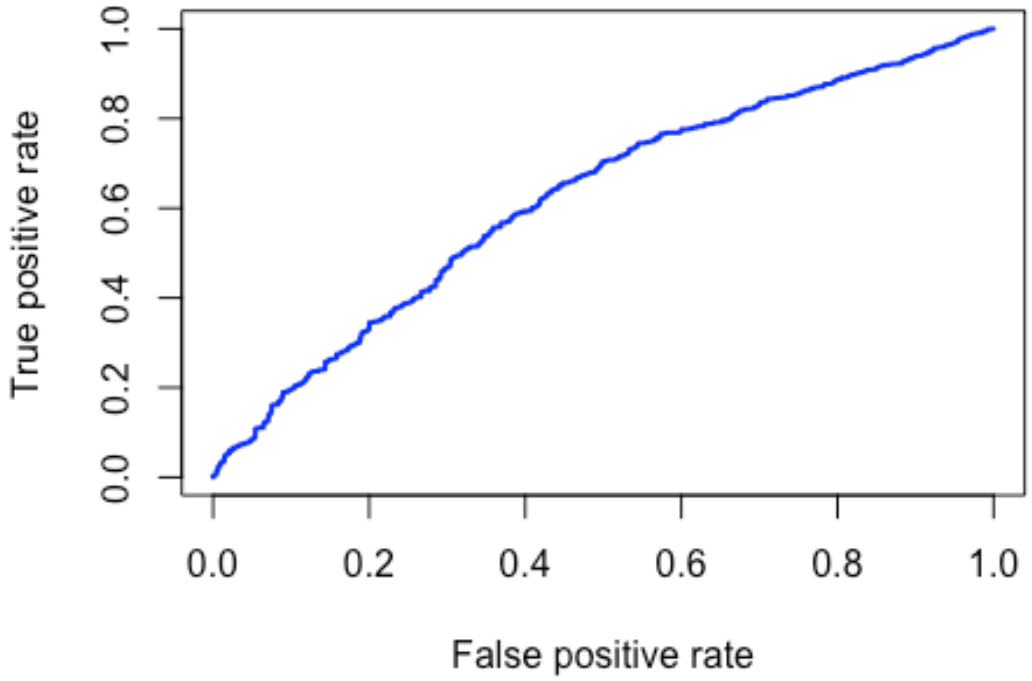

```r
# exemple : rstudio-pubs-static.s3.amazonaws.com/275847_f5915e73ec3f4e9091885fed379ed775

plot(modelRBF, train, LogC_AN_TTC ~ COEFF,
     slice = list(PUIS_FISC = 2, COEFF = 3, AGE_CLI = 4, NB_ENF = 5, ANC_VEH = 6), fill = TRUE, grid = 50
)
```



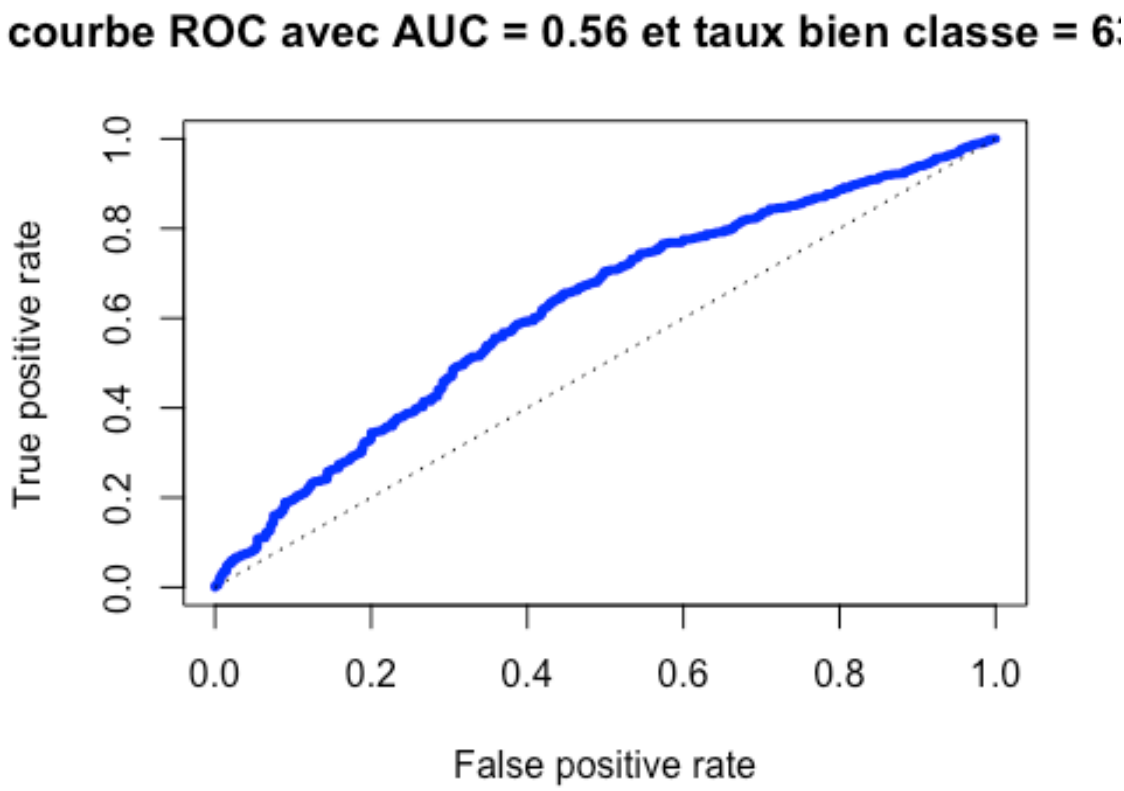
```r
toc <-  Sys.time() -tic
print(toc)

## Time difference of 21.81135 secs

#*******
# PARTIE 2.4 - 3e iteration : grid_search "avec libsvm et cross-validation k-folds"
# trouver les paramètres du best modèle
#*******


#===== e1071 - tune() - crossvalidation - multi kernels
#= procedure multi-noyaux time consuming

tic <- Sys.time()

set.seed(1000) #pour obtenir à chaque lancement le même résultat
obj <- tune (
  svm,
  trsf_dev~LogC_AN_TTC+PUIS_FISC+COEFF+AGE_CLI+NB_ENF+ANC_VEH,
  data = train,
  type='C-classification',
  scale=TRUE,
  probability=TRUE,
  class.weights=wts,
  ranges = list(kernel=c('linear','polynomial','radial', 'sigmoid'),

                #cost = c(0.1,0.5,1.0,2.0,10), # cost ini =1; born_ingf=1/10 et born_sup=1*10
                #cost = c(0.001 , 0.01 , 0.1, 1 ,5 ,10 ,100,1000),

                #cost = c(0.001, 0.1, 0.5, 1 ,10 ,100),
                #cost=seq(0.3,0.6,by=0.01),
                cost=1000,

                #gamma = 2^(-10:0)
                #gamma=seq(0.01,1,by=0.01)

                #gamma=c(0.05, 0.25, 0.5, 0.75, 1)
                #gamma=seq(0.3,0.6,by=0.01)
                gamma=seq(0.15,0.2,by=0.01)

  ), # gamma ini = 0.25; [0.25*4/1000;0.25*4]=[0.001;1]


  tunecontrol = tune.control(sampling=c("cross"))
)
Best=obj$best.parameters
print(Best)

##   kernel cost gamma
## 7 radial 1000  0.16

summary(Best)

##         kernel       cost          gamma
##  linear    :0   Min.   :1000   Min.   :0.16
##  polynomial:0   1st Qu.:1000   1st Qu.:0.16
##  radial    :1   Median :1000   Median :0.16
##  sigmoid   :0   Mean   :1000   Mean   :0.16
##                 3rd Qu.:1000   3rd Qu.:0.16
##                 Max.   :1000   Max.   :0.16

toc <-  Sys.time() -tic
print(toc)

## Time difference of 55.09498 mins

# resultats



#*******
```

```r
#  PARTIE 2 - 4e itération :  modele 2 post grid search
#  avec prise en compte du meilleur modèle avec les paramètres optimaux
# resultat du GridSearch multi_4_noyaux
#*******

tic <- Sys.time()
model2radial=svm(trsf_dev~LogC_AN_TTC+PUIS_FISC+COEFF+AGE_CLI+NB_ENF+ANC_VEH,data=train,
                 kernel="radial",
                 type='C-classification',
                 scale=TRUE,
                 probability=TRUE,
                 cost=1000,
                 gamma=0.2,
                 class.weights = wts
)
summary(model2radial)

##
## Call:
## svm(formula = trsf_dev ~ LogC_AN_TTC + PUIS_FISC + COEFF + AGE_CLI +
##     NB_ENF + ANC_VEH, data = train, kernel = "radial", type = "C-classification",
##     probability = TRUE, cost = 1000, gamma = 0.2, class.weights = wts,
##     scale = TRUE)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  1000
##       gamma:  0.2
##
## Number of Support Vectors:  3400
##
##  ( 2467 933 )
##
##
## Number of Classes:  2
##
## Levels:
##  N O

print(model2radial)

##
## Call:
## svm(formula = trsf_dev ~ LogC_AN_TTC + PUIS_FISC + COEFF + AGE_CLI +
##     NB_ENF + ANC_VEH, data = train, kernel = "radial", type = "C-classification",
##     probability = TRUE, cost = 1000, gamma = 0.2, class.weights = wts,
##     scale = TRUE)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  1000
##       gamma:  0.2
##
## Number of Support Vectors:  3400

#Prédiction et matrice de confusion
pred.svm=predict(model2radial,newdata=test)
table.SVM=table(pred.svm,test$trsf_dev)
table.SVM

##
## pred.svm   N   O
##        N 775 177
##        O 590 303

TxBcSVM=(sum(diag(table.SVM))/sum(table.SVM))*100
TxBcSVM
```

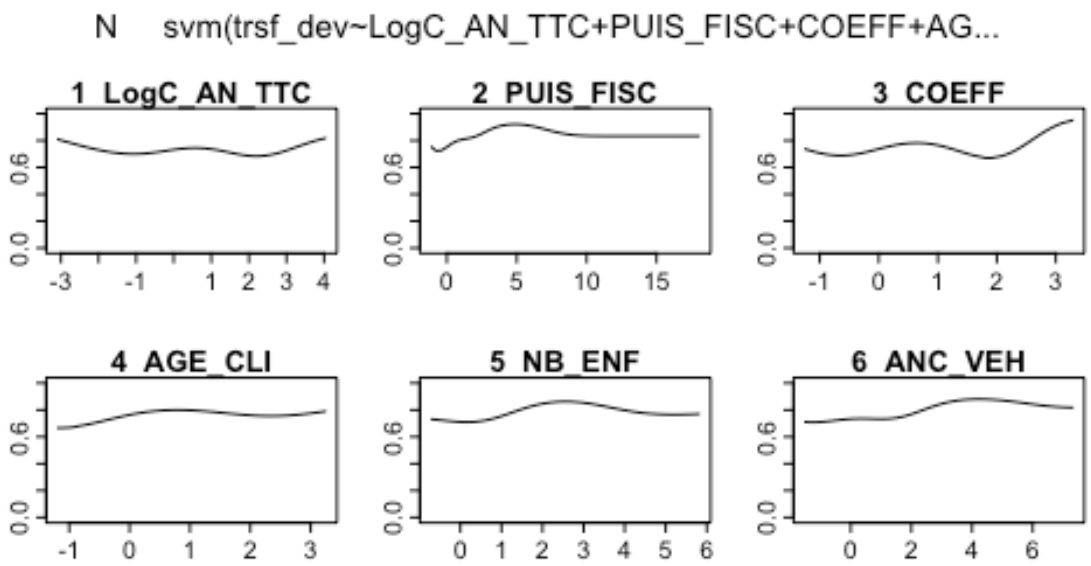
```
## [1] 58.42818

#

#plot(cmdscale(dist(MarTest[,-14])),col=as.integer(MarTest[,14]))
ErreurSVM=100-TxBcSVM
ErreurSVM

## [1] 41.57182

#

#Courbe ROC
library(ROCR)
ROCsvm=predict(model2radial,newdata=test,decision.values = T)
a=attr(ROCsvm,"decision.values")
predsvm=prediction(a,test$trsf_dev=="N")
perfSVM=performance(predsvm,"tpr","fpr")
AUC.SVM=performance(predsvm,"auc")
plot(perfSVM,col=4,lwd=2)
```



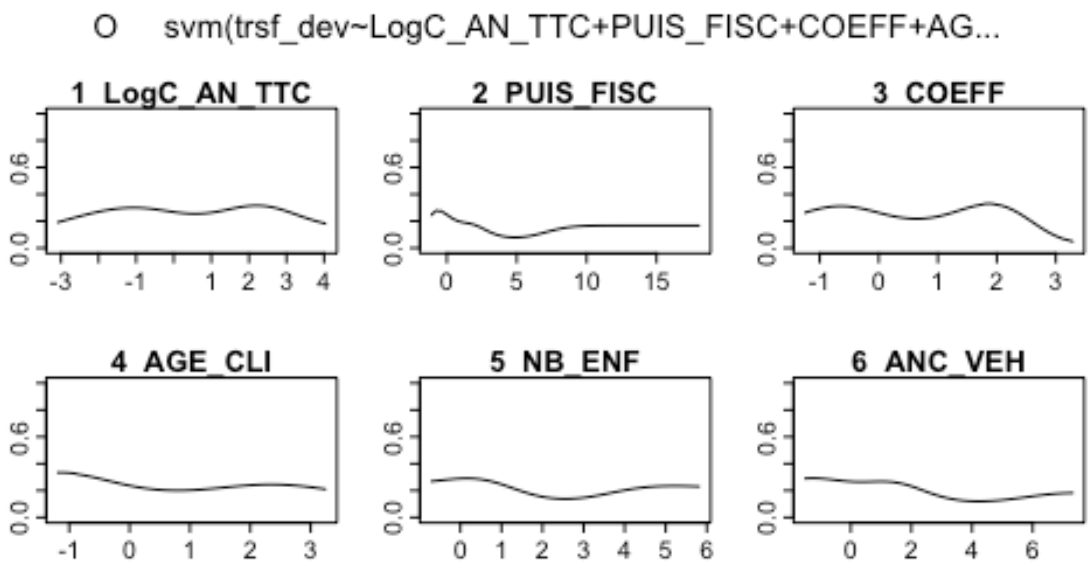

```
AUC.SVM@y.values[[1]]

## [1] 0.6215934

#

plot(perfSVM,col=4,lwd=4, main="courbe ROC avec AUC = 0.56 et taux bien classe = 63.24")
segments(0,0,1,1,lty=3)
```




```
toc <-  Sys.time() -tic
print(toc)

## Time difference of 11.29471 secs

#********

tic <- Sys.time()

# milbo.org/doc/plotmo-notes.pdf
plotmo(model2radial, predict.probability=TRUE, nresponse="N", ylim=c(0,1))
```

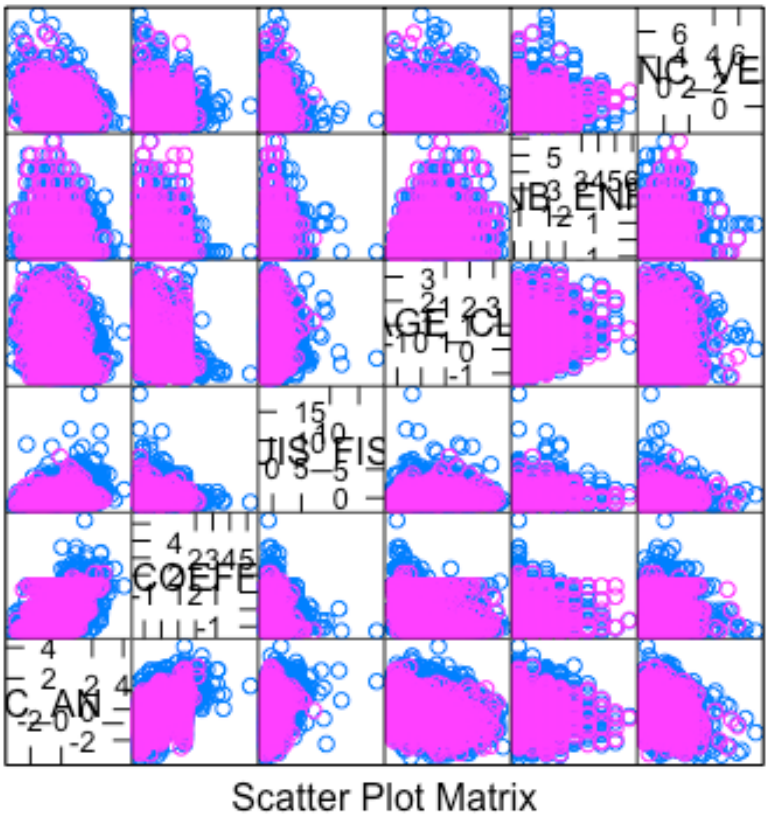



```r
plotmo(model2radial, predict.probability=TRUE, nresponse="O", ylim=c(0,1))
```




```r
# exemple rstudio-pubs-static.s3.amazonaws.com/275847_f5915e73ec3f4e9091885fed379ed775

toc <-  Sys.time() -tic
print(toc)
```

```
## Time difference of 2.152796 secs

#*****************************************************************************************************
*****************************************************************************************
#*****************************************************************************************************
*****************************************************************************************
#*****************************************************************************************************
*****************************************************************************************

#*****************************************************************************
# PARTIE 2.4       CARET : calcul parallel - automatic grid search
#
# Radial.a
#
# source :         geekoverdose.wordpress.com/2014/07/25/svm-classification-example-with-performance-meas
ures-using-r-caret/
#*****************************************************************************


#=================================================================================================
====================#

tic <- Sys.time()
library(caret)
library(plyr)
library(ROCR)
library(ggplot2)
library(kernlab)

##
## Attaching package: 'kernlab'

## The following object is masked from 'package:ggplot2':
##
##     alpha

library(parallel)
library(iterators)
library(foreach)

# optional - enable multicore processing
library(doMC)
registerDoMC(cores=3) # laisser 1 coeur pour l'OS

# create binary problem: only use two classes
x <- df31[,1:6]
str(x)

## 'data.frame':    6150 obs. of  6 variables:
##  $ LogC_AN_TTC: num  0.901 0.263 -1.263 -0.974 -2.004 ...
##  $ COEFF      : num  -0.051 -1.06 -1.262 -1.262 -0.253 ...
##  $ PUIS_FISC  : num  0.149 2.338 -0.727 0.586 -0.727 ...
##  $ AGE_CLI    : num  -1.012 1.091 1.218 1.537 -0.438 ...
##  $ NB_ENF     : num  -0.724 0.915 0.915 0.915 -0.724 ...
##  $ ANC_VEH    : num  0.243 0.404 -0.403 0.566 0.889 ...

y  <- as.factor(df31[,7])
str(y)

##  Factor w/ 2 levels "N","O": 1 1 2 2 1 1 2 1 2 1 ...

length(y)

## [1] 6150

# we see that data will probably not be perfectly seperable using linear separation
# library(ggplot2)
featurePlot(x = x, y = y, "pairs")
```




```
# split into train and test data. train = cv parameter grid search and model creation, test = performance
analysis
# source : rdocumentation.org/packages/caret/versions/4.61/topics/createDataPartition

# echantillon train avec un poids 70%
set.seed(123456)
indexes_y_test <- createDataPartition(y = 1:length(y), times = 1, p = 0.7)[[1]]

# creation of weights - also fast for very big datasets - correction des tailles des classes
weights <- as.numeric(y[-indexes_y_test])
for(val in unique(weights)) {weights[weights==val]=1/sum(weights==val)*length(weights)/2} # normalized to
sum to length(samples)


# faire gridsearch pour chaque noyau (et spécifier le C et le Gamma) et comparer ds tableau / pour choisi
r le meilleur modèle
#*** svmRadial exemple :  blog.revolutionanalytics.com/2015/10/the-5th-tribe-support-vector-machines-and-
caret
#*** caret website : tous les modèles
#*** r-bloggers.com/the-5th-tribe-support-vector-machines-and-caret/ :boxplot des roc / noyaux
#*** rstudio-pubs-static.s3.amazonaws.com/207681_811028f0e06a4fac9ccd109e07dc8a11


#*****************************************************************************************************
************************
#** Jason Brownlee
#** machinelearningmastery.com/tuning-machine-learning-models-using-the-caret-r-package/
#** This can be done by setting the ** tuneLength ** to indicate the number of different values to try fo
r each algorithm parameter.


#**  Grid Search: Manual Grid
#**  Grid Search: Automatic Grid
#*****************************************************************************************************
************************



# Grid search automatique
```

```
model <- train(
  #method = 'svmLinear',
  method = 'svmRadial',
  #method = 'svmPoly',

  #************************  method = 'svmSig',

  x = x[-indexes_y_test,], # on prend x train
  y = y[-indexes_y_test],  # on prend y train
  weights = weights,
  maximize = T,
  #tuneGrid = expand.grid(.sigma=3^(-2:2)),                                                    # linear
  tuneGrid = expand.grid(.C=seq(0.3,0.6,by=0.05),.sigma=seq(0.3,0.6,by=0.05)),                # Radial.
a
  #tuneGrid = expand.grid(.C=3^(-2:2),.degree=(3), .scale=3^(-2:2)),                           # Polynom
ial
  #tuneGrid = expand.grid(.C=seq(0.3,0.39,by=0.01),.degree=(3), .scale=seq(0.1,0.19,by=0.01)),  # Polynom
ial
  #Metric = "ROC", #*************************************************************************  # classif
ieur binaire

  #************************* tuneGrid = expand.grid(.scale=1^(-3:2)),

  #** analyticsvidhya.com/blog/2016/12/practical-guide-to-implement-machine-learning-with-caret-package-i
n-r-with-practice-problem/
  #** rdrr.io/cran/caret/man/models.html # tous les noyaux svm
  #** listendata.com/2017/01/support-vector-machine-in-r-tutorial.html

  preProcess = c('center', 'scale'),
  tuneLength=7,
  trControl = trainControl(method = 'repeatedcv', # cross validation
                           number = 10,   # nr of cv sets

                           repeats = 5, # use with method=repeatcv

                           returnResamp = 'none', # return accuracy per cv partition and parameter settin
g
                           classProbs = T, # return prediction probabilities along with predicted classes

                           #summaryFunction = twoClassSummary, #****************************** classifieu
r binaire

                           #savePredictions=T, # returns all predictions (for all cv paritions) for each
tuneGrid parameter set

                           returnData = F, # disable return of training data e.g. for big data sets
                           allowParallel = T

                           # tuneLength=10  # argument inutilisé (tuneLength = 10)

  )
)

toc <-  Sys.time() -tic
print(toc)

## Time difference of 10.55139 mins

# Time difference of 1.05665 hours

# we see some accuracy around 0.7-0.8
# on récupère le meilleur modèle obtenu sur avec x train et y train
model

## Support Vector Machines with Radial Basis Function Kernel
##
## Pre-processing: centered, scaled
## Resampling: Cross-Validated (10 fold, repeated 5 times)
##
## Summary of sample sizes: 1660, 1660, 1660, 1658, 1660, 1660, ...
##
## Resampling results across tuning parameters:
```

```
##
##   C     sigma  Accuracy  Kappa      Accuracy SD  Kappa SD
##   0.30  0.30   0.732     -0.000430  0.00229      0.00213
##   0.30  0.35   0.732     -0.000430  0.00229      0.00213
##   0.30  0.40   0.732     -0.002121  0.00435      0.00708
##   0.30  0.45   0.732     -0.000634  0.00288      0.00448
##   0.30  0.50   0.733     -0.000214  0.00221      0.00151
##   0.30  0.55   0.733      0.000000  0.00192      0.00000
##   0.30  0.60   0.732     -0.000639  0.00291      0.00334
##   0.35  0.30   0.732     -0.000430  0.00229      0.00213
##   0.35  0.35   0.733     -0.000214  0.00221      0.00151
##   0.35  0.40   0.733     -0.000215  0.00201      0.00152
##   0.35  0.45   0.732     -0.000855  0.00281      0.00363
##   0.35  0.50   0.733     -0.000215  0.00201      0.00152
##   0.35  0.55   0.732     -0.000859  0.00235      0.00294
##   0.35  0.60   0.733     -0.000214  0.00221      0.00151
##   0.40  0.30   0.732     -0.000645  0.00236      0.00258
##   0.40  0.35   0.732     -0.001061  0.00333      0.00489
##   0.40  0.40   0.732     -0.001061  0.00351      0.00489
##   0.40  0.45   0.732     -0.000430  0.00229      0.00213
##   0.40  0.50   0.732     -0.000645  0.00236      0.00258
##   0.40  0.55   0.732     -0.001064  0.00346      0.00443
##   0.40  0.60   0.732     -0.000642  0.00243      0.00335
##   0.45  0.30   0.732     -0.000847  0.00334      0.00469
##   0.45  0.35   0.732     -0.000847  0.00334      0.00469
##   0.45  0.40   0.733     -0.000214  0.00221      0.00151
##   0.45  0.45   0.733     -0.000215  0.00201      0.00152
##   0.45  0.50   0.732     -0.000639  0.00291      0.00334
##   0.45  0.55   0.732     -0.001068  0.00315      0.00388
##   0.45  0.60   0.732     -0.001274  0.00347      0.00552
##   0.50  0.30   0.732     -0.000428  0.00247      0.00212
##   0.50  0.35   0.732     -0.001276  0.00354      0.00507
##   0.50  0.40   0.732     -0.000644  0.00253      0.00257
##   0.50  0.45   0.732     -0.000428  0.00222      0.00212
##   0.50  0.50   0.732     -0.000645  0.00209      0.00258
##   0.50  0.55   0.732     -0.000849  0.00293      0.00471
##   0.50  0.60   0.732     -0.000431  0.00209      0.00213
##   0.55  0.30   0.732     -0.001060  0.00346      0.00489
##   0.55  0.35   0.733      0.000593  0.00212      0.00420
##   0.55  0.40   0.733      0.000000  0.00192      0.00000
##   0.55  0.45   0.732     -0.000644  0.00229      0.00257
##   0.55  0.50   0.732     -0.001270  0.00378      0.00550
##   0.55  0.55   0.733     -0.000214  0.00221      0.00151
##   0.55  0.60   0.732     -0.001273  0.00359      0.00551
##   0.60  0.30   0.732     -0.000845  0.00347      0.00468
##   0.60  0.35   0.733     -0.000215  0.00201      0.00152
##   0.60  0.40   0.732     -0.000858  0.00253      0.00294
##   0.60  0.45   0.732     -0.000639  0.00270      0.00334
##   0.60  0.50   0.732     -0.000854  0.00296      0.00363
##   0.60  0.55   0.732     -0.000645  0.00209      0.00258
##   0.60  0.60   0.733     -0.000215  0.00201      0.00152
##
## Accuracy was used to select the optimal model using  the largest value.
## The final values used for the model were sigma = 0.35 and C = 0.55.
```

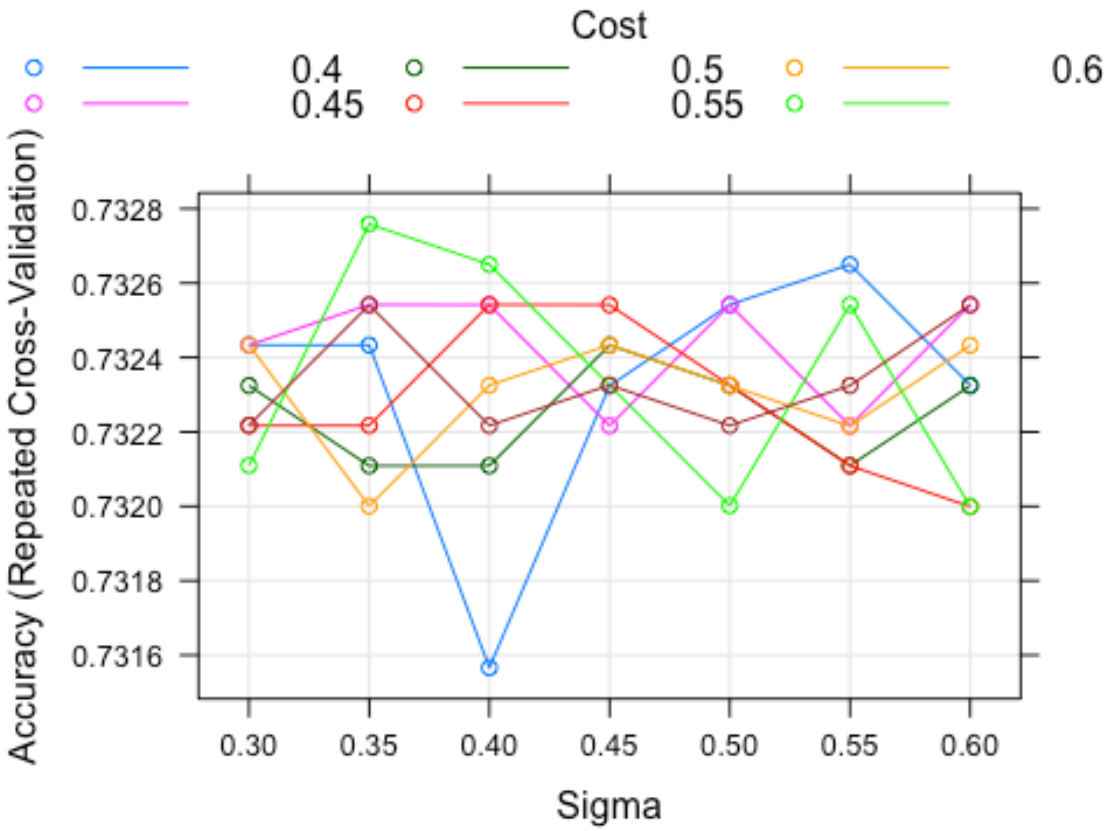
```r
head(with(model, results[order(results$Kappa, decreasing=T),]))
```

```
##       C sigma  Accuracy         Kappa  AccuracySD     KappaSD
## 37 0.55  0.35 0.7327588  0.0005932762 0.002121099 0.004195096
## 38 0.55  0.40 0.7326501  0.0000000000 0.001917391 0.000000000
## 6  0.30  0.55 0.7326501  0.0000000000 0.001917391 0.000000000
## 9  0.35  0.35 0.7325420 -0.0002142475 0.002214740 0.001514958
## 24 0.45  0.40 0.7325420 -0.0002142475 0.002214740 0.001514958
## 5  0.30  0.50 0.7325420 -0.0002142475 0.002214740 0.001514958
```

```r
summary(model)
```

```
## Length  Class   Mode
##      1   ksvm     S4
```

```r
# graphe model
plot(model) # vérifier graphiquement, quel est le C qui maximise l'accuracy.
```




```
#> model
# Support Vector Machines with Radial Basis Function Kernel
# Accuracy was used to select the optimal model using  the largest value.


#====================> preuve : nb de labels correctements prédits (on cherche le max acurracy sigma = 0.
50 and C = 0.55)

#    C      sigma Accuracy Kappa       AccuracySD  KappaSD
# 33 0.50  0.50 0.7291712 0.04836188 0.009343574 0.03577221
# 49 0.60  0.60 0.7285184 0.04756057 0.009536888 0.03908394
# 26 0.45  0.50 0.7284133 0.04751027 0.009650658 0.03641344
# 40 0.55  0.50 0.7292799 0.04744189 0.008974953 0.03991382

#**  prévision sur test
SVMtest = predict.train(object = model, newdata = x[indexes_y_test,], type='raw', y[indexes_y_test])
SVMtest

##    [1] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##   [35] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##   [69] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [103] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [137] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [171] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [205] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [239] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [273] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [307] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [341] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [375] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [409] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [443] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [477] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [511] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [545] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [579] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [613] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [647] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [681] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
```

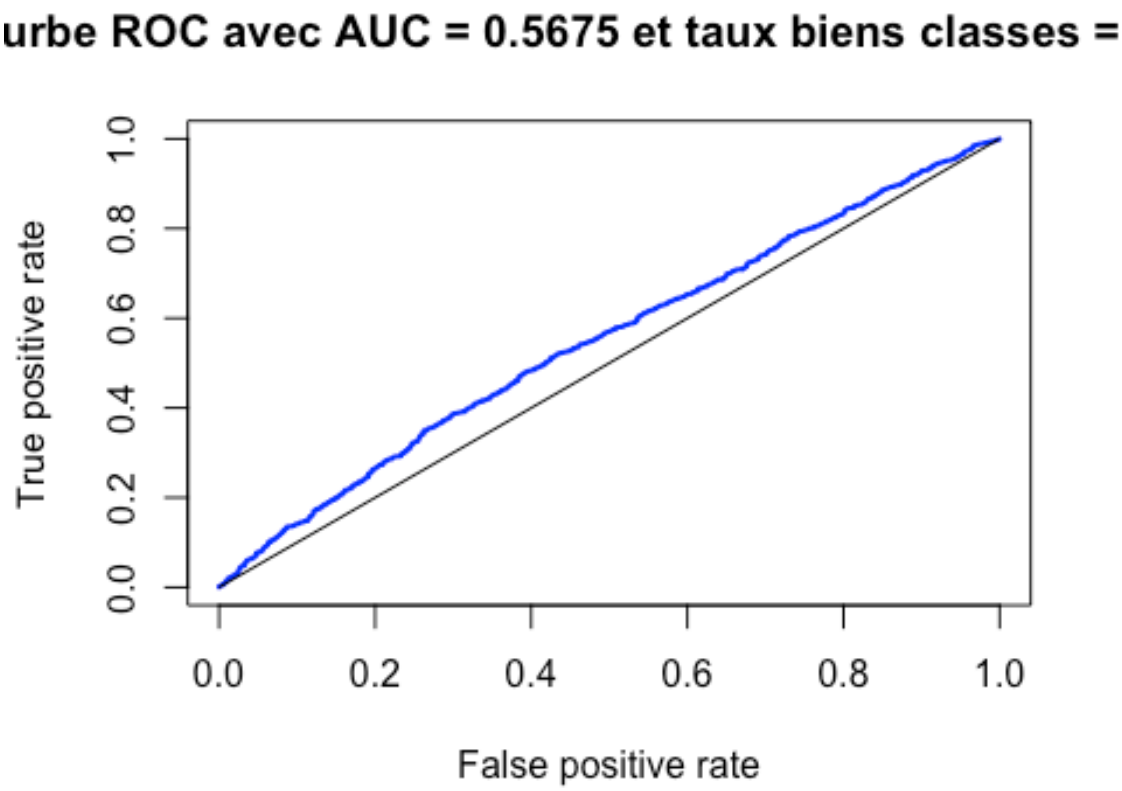
```
##  [715] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [749] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [783] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [817] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [851] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [885] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [919] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [953] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [987] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1021] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1055] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1089] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1123] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1157] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1191] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1225] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1259] N N N N N N N N N N N N N N N N N N N N N O N N N N N N N N N N N
## [1293] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1327] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1361] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1395] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1429] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1463] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1497] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1531] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1565] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1599] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1633] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1667] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1701] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1735] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1769] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1803] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1837] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1871] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1905] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1939] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1973] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2007] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2041] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2075] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2109] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2143] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2177] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2211] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2245] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2279] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2313] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2347] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2381] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N O N N
## [2415] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2449] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2483] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2517] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2551] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2585] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2619] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2653] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2687] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2721] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2755] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2789] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2823] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2857] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2891] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2925] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2959] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2993] N N N N N N N N N N N N N N N N N N N N O N N N N N N N N N N N N
## [3027] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3061] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3095] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3129] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
```

```
## [3163] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3197] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3231] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3265] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3299] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3333] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3367] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3401] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3435] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3469] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3503] N N N O N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3537] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3571] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3605] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3639] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3673] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3707] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3741] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3775] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3809] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3843] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3877] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3911] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3945] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3979] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [4013] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [4047] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [4081] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [4115] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N O N
## [4149] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [4183] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [4217] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [4251] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [4285] N N N N N N N N N N N N N N N N N N N N N
## Levels: N O

# Confusion Matrix and Statistics === FONCTION PACKAGE CARET
CM = confusionMatrix(data=SVMtest, y[indexes_y_test])
CM

## Confusion Matrix and Statistics
##
##           Reference
## Prediction    N    O
##          N 3188 1113
##          O    4    1
##
##                Accuracy : 0.7406
##                  95% CI : (0.7272, 0.7536)
##     No Information Rate : 0.7413
##     P-Value [Acc > NIR] : 0.5496
##
##                   Kappa : -5e-04
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.9987469
##             Specificity : 0.0008977
##          Pos Pred Value : 0.7412230
##          Neg Pred Value : 0.2000000
##              Prevalence : 0.7412912
##          Detection Rate : 0.7403623
##    Detection Prevalence : 0.9988388
##       Balanced Accuracy : 0.4998223
##
##        'Positive' Class : N
##

#Confusion Matrix and Statistics
#Reference
#Prediction    N    O
#N 3118 1093
#O   65   30
```

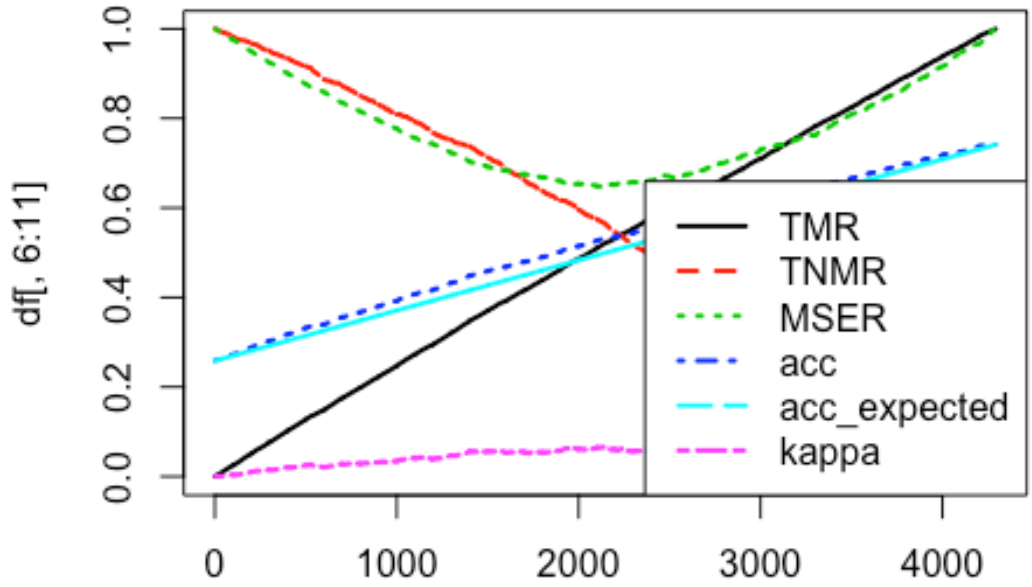
```
#Accuracy : 0.7311
#95% CI : (0.7176, 0.7443)
#No Information Rate : 0.7392
#P-Value [Acc > NIR] : 0.8908
#Kappa : 0.0089
#Mcnemar's Test P-Value : <2e-16
#Sensitivity : 0.97958
#Specificity : 0.02671
#Pos Pred Value : 0.74044
#Neg Pred Value : 0.31579
#Prevalence : 0.73920
#Detection Rate : 0.72411
#Detection Prevalence : 0.97794
#Balanced Accuracy : 0.50315
```

Positive' Class : N

```
# confusion matrix: model predicting classes of test data
table.SVM=table(predict.train(object = model, newdata = x[indexes_y_test,], type='raw'), y[indexes_y_test
]) #
table.SVM

##
##        N    O
##   N 3188 1113
##   O    4    1

TxBcSVM=(sum(diag(table.SVM))/sum(table.SVM))*100
TxBcSVM

## [1] 74.05945

# svmLinear > [1] 30.55556
# ******************************svmRadial > [1] 73.10729


# prediction probabilities of test data classes
probs <- predict.train(object = model, newdata = x[indexes_y_test,], type='prob')[,1]
isPositiveClass <- y[indexes_y_test] == 'N' # for a ROC curve there is a positive class (true match rate.
..) - defining that class here
pred <- prediction(probs, isPositiveClass)
perf <- performance(pred, 'tpr', 'fpr')

# plot: either
plot(perf, col=4, lwd=2, main="courbe ROC avec AUC = 0.5675 et taux biens classes = 0.7311")
segments(0,0,1,1,lty=1)
```

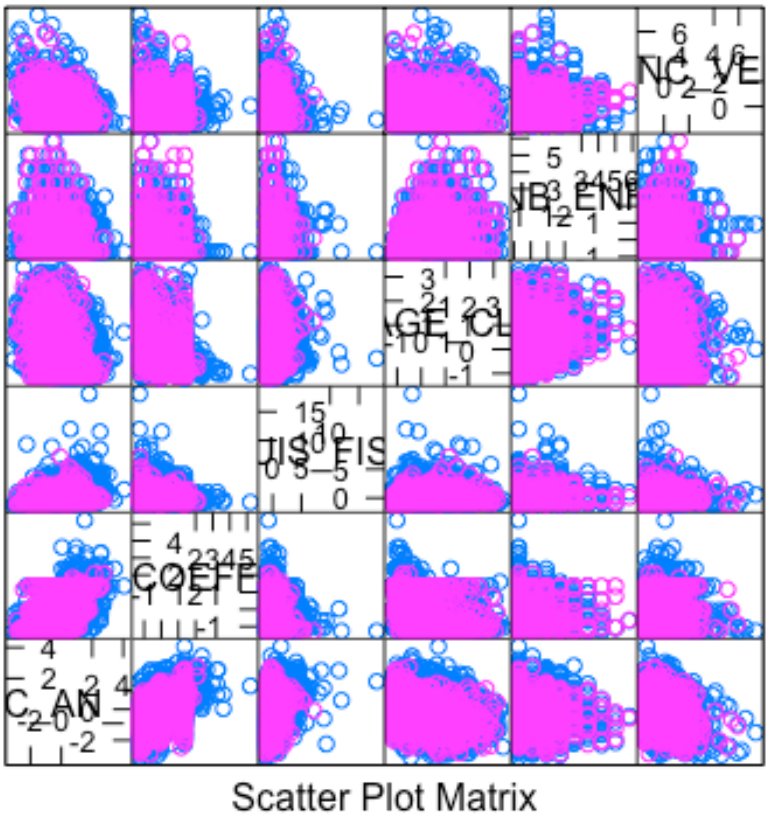



```
# or
#with(attributes(perf), plot(x=x.values[[1]], y=y.values[[1]], type='l'))


# some metrics: AUC (area under curve), EER (equal error rate), MSER (minimum squared error rate), Cohen'
s Kappa etc.

#***************
# definition de toutes les metrics
# https://machinelearningmastery.com/machine-learning-evaluation-metrics-in-r/
#***************

AUC <- attributes(performance(pred, 'auc'))$y.value[[1]] # area under curve
AUC

## [1] 0.5503608

# [1] 0.5675

df <- with(attributes(pred), data.frame(cutoffs=cutoffs[[1]], tp=tp[[1]], fn=fn[[1]], tn=tn[[1]], fp=fp[[
1]], TMR=tp[[1]]/(tp[[1]]+fn[[1]]), TNMR=tn[[1]]/(tn[[1]]+fp[[1]])))
df$MSER <- with(df, sqrt((1-TMR)**2+(1-TNMR)**2))
MSER <- with(df, sqrt(TMR**2+TNMR**2)) # sqrt of minimum squared error rate = eucl. distance to point TMR
=TNMR=1
i_eer <- with(df, which.min(abs(TMR-TNMR)))
EER <- with(df[i_eer,], mean(c(TMR,TNMR))) # equal error rate: mean would not be required when using ROCR
as it's always exact
df$acc <- with(df, (tp + tn) / length(isPositiveClass)) # observed accuracy
df$acc_expected <- with(df, sum(isPositiveClass) * (tp+fp) / length(isPositiveClass) + sum(!isPositiveCla
ss) * (tn+fn) / length(isPositiveClass)) / length(isPositiveClass) # expected accuracy
df$kappa <- with(df, (acc - acc_expected) / (1 - acc_expected)) # cohen's kappa

# graphical representation
matplot(df[,6:11], lty=1:6, lwd=2, type='l'); legend('bottomright', legend=names(df[,6:11]), lty=1:6, lwd
=2, col=1:6)
```




```
# characteristics for settings with best EER, MSER etc.
df[i_eer,] # curve point for equal error rate

##        cutoffs   tp   fn  tn  fp       TMR      TNMR      MSER       acc
## 2228 0.7389532 1719 1473 600 514 0.5385338 0.5385996 0.6525652 0.5385509
##      acc_expected      kappa
## 2228    0.5089658 0.06025057

df[order(df$acc, decreasing=T)[[1]],] # curve point for max accuracy

##        cutoffs   tp fn tn   fp TMR TNMR MSER       acc acc_expected kappa
## 4292 0.4079197 3192  0  0 1114   1    0    1 0.7412912    0.7412912     0

df[order(df$kappa, decreasing=T)[[1]],] # curve point for max kappa

##        cutoffs   tp   fn  tn  fp       TMR      TNMR      MSER       acc
## 2146 0.7391773 1665 1527 630 484 0.5216165 0.5655296 0.6462316 0.5329772
##      acc_expected      kappa
## 2146    0.4995517 0.06679118

df[order(df$MSER, decreasing=F)[[1]],] # curve point for minimum squared error rate

##        cutoffs   tp   fn  tn  fp       TMR      TNMR      MSER       acc
## 2146 0.7391773 1665 1527 630 484 0.5216165 0.5655296 0.6462316 0.5329772
##      acc_expected      kappa
## 2146    0.4995517 0.06679118

toc <-  Sys.time() -tic
print(toc)

## Time difference of 11.07466 mins

#*****************************************************************************************************
*****************************************************************************************
#*****************************************************************************************************
*****************************************************************************************
#*****************************************************************************************************
*****************************************************************************************
```

```
#*****************************************************************************************************
*****************************************************************************************
#*****************************************************************************************************
*****************************************************************************************
#*****************************************************************************************************
*****************************************************************************************

#*****************************************************************************
# PARTIE 2.4       CARET : calcul parallel - automatic grid search
#
# radial.b
#
# source :         geekoverdose.wordpress.com/2014/07/25/svm-classification-example-with-performance-meas
ures-using-r-caret/
#*****************************************************************************


#=================================================================================================
====================#

tic <- Sys.time()
library(caret)
library(plyr)
library(ROCR)
library(ggplot2)
library(kernlab)
library(parallel)
library(iterators)
library(foreach)

# optional - enable multicore processing
library(doMC)
registerDoMC(cores=3) # laisser 1 coeur pour l'OS

# create binary problem: only use two classes
x <- df31[,1:6]
str(x)

## 'data.frame':    6150 obs. of  6 variables:
##  $ LogC_AN_TTC: num  0.901 0.263 -1.263 -0.974 -2.004 ...
##  $ COEFF      : num  -0.051 -1.06 -1.262 -1.262 -0.253 ...
##  $ PUIS_FISC  : num  0.149 2.338 -0.727 0.586 -0.727 ...
##  $ AGE_CLI    : num  -1.012 1.091 1.218 1.537 -0.438 ...
##  $ NB_ENF     : num  -0.724 0.915 0.915 0.915 -0.724 ...
##  $ ANC_VEH    : num  0.243 0.404 -0.403 0.566 0.889 ...

y  <- as.factor(df31[,7])
str(y)

##  Factor w/ 2 levels "N","O": 1 1 2 2 1 1 2 1 2 1 ...

length(y)

## [1] 6150

# we see that data will probably not be perfectly seperable using linear separation
# library(ggplot2)
featurePlot(x = x, y = y, "pairs")
```




```
# split into train and test data. train = cv parameter grid search and model creation, test = performance analysis
# source : rdocumentation.org/packages/caret/versions/4.61/topics/createDataPartition

# echantillon train avec un poids 70%
set.seed(123456)
indexes_y_test <- createDataPartition(y = 1:length(y), times = 1, p = 0.7)[[1]]

# creation of weights - also fast for very big datasets - correction des tailles des classes
weights <- as.numeric(y[-indexes_y_test])
for(val in unique(weights)) {weights[weights==val]=1/sum(weights==val)*length(weights)/2} # normalized to sum to length(samples)


# faire gridsearch pour chaque noyau (et spécifier le C et le Gamma) et comparer ds tableau / pour choisir le meilleur modèle
#*** svmRadial exemple :  blog.revolutionanalytics.com/2015/10/the-5th-tribe-support-vector-machines-and-caret
#*** caret website : tous les modèles
#*** r-bloggers.com/the-5th-tribe-support-vector-machines-and-caret/ :boxplot des roc / noyaux
#*** rstudio-pubs-static.s3.amazonaws.com/207681_811028f0e06a4fac9ccd109e07dc8a11


#*****************************************************************************************************************
#** Jason Brownlee
#** machinelearningmastery.com/tuning-machine-learning-models-using-the-caret-r-package/
#** This can be done by setting the ** tuneLength ** to indicate the number of different values to try for each algorithm parameter.


#**  Grid Search: Manual Grid
#**  Grid Search: Automatic Grid
#*****************************************************************************************************************



# Grid search automatique
```

```
model <- train(
  #method = 'svmLinear',
  method = 'svmRadial',
  #method = 'svmPoly',

  #************************  method = 'svmSig',

  x = x[-indexes_y_test,], # on prend x train
  y = y[-indexes_y_test],  # on prend y train
  weights = weights,
  maximize = T,
  #tuneGrid = expand.grid(.sigma=3^(-2:2)),                                                    # linear
  #tuneGrid = expand.grid(.C=seq(0.3,0.6,by=0.05),.sigma=seq(0.3,0.6,by=0.05)),                # Radial.
a
  tuneGrid = expand.grid(.C=1000,.sigma=seq(0.15,0.20,by=0.01)),                               # Radial.
b
  #tuneGrid = expand.grid(.C=3^(-2:2),.degree=(3), .scale=3^(-2:2)),                           # Polynom
ial
  #tuneGrid = expand.grid(.C=seq(0.3,0.39,by=0.01),.degree=(3), .scale=seq(0.1,0.19,by=0.01)),  # Polynom
ial
  #Metric = "ROC", #*************************************************************************  # classif
ieur binaire

  #************************* tuneGrid = expand.grid(.scale=1^(-3:2)),

  #** analyticsvidhya.com/blog/2016/12/practical-guide-to-implement-machine-learning-with-caret-package-i
n-r-with-practice-problem/
  #** rdrr.io/cran/caret/man/models.html # tous les noyaux svm
  #** listendata.com/2017/01/support-vector-machine-in-r-tutorial.html

  preProcess = c('center', 'scale'),
  tuneLength=7,
  trControl = trainControl(method = 'repeatedcv', # cross validation
                           number = 10,   # nr of cv sets

                           repeats = 5, # use with method=repeatcv

                           returnResamp = 'none', # return accuracy per cv partition and parameter settin
g
                           classProbs = T, # return prediction probabilities along with predicted classes

                           #summaryFunction = twoClassSummary, #****************************** classifieu
r binaire

                           #savePredictions=T, # returns all predictions (for all cv paritions) for each
tuneGrid parameter set

                           returnData = F, # disable return of training data e.g. for big data sets
                           allowParallel = T

                           # tuneLength=10  # argument inutilisé (tuneLength = 10)

  )
)

## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info =
## trainInfo, : There were missing values in resampled performance measures.

toc <-  Sys.time() -tic
print(toc)

## Time difference of 14.93982 mins

# Time difference of 1.05665 hours

# we see some accuracy around 0.7-0.8
# on récupère le meilleur modèle obtenu sur avec x train et y train
model

## Support Vector Machines with Radial Basis Function Kernel
##
## Pre-processing: centered, scaled
```

```
## Resampling: Cross-Validated (10 fold, repeated 5 times)
##
## Summary of sample sizes: 1660, 1660, 1660, 1658, 1660, 1660, ...
##
## Resampling results across tuning parameters:
##
##   sigma  Accuracy  Kappa      Accuracy SD  Kappa SD
##   0.15   0.732     -0.001123  0.00330      0.00595
##   0.16   0.732     -0.001770  0.00346      0.00524
##   0.17   0.732     -0.001343  0.00320      0.00491
##   0.18   0.732     -0.001125  0.00322      0.00596
##   0.19   0.732     -0.001533  0.00319      0.00379
##   0.20   0.732     -0.000925  0.00325      0.00585
##
## Tuning parameter 'C' was held constant at a value of 1000
## Accuracy was used to select the optimal model using  the largest value.
## The final values used for the model were sigma = 0.2 and C = 1000.
```

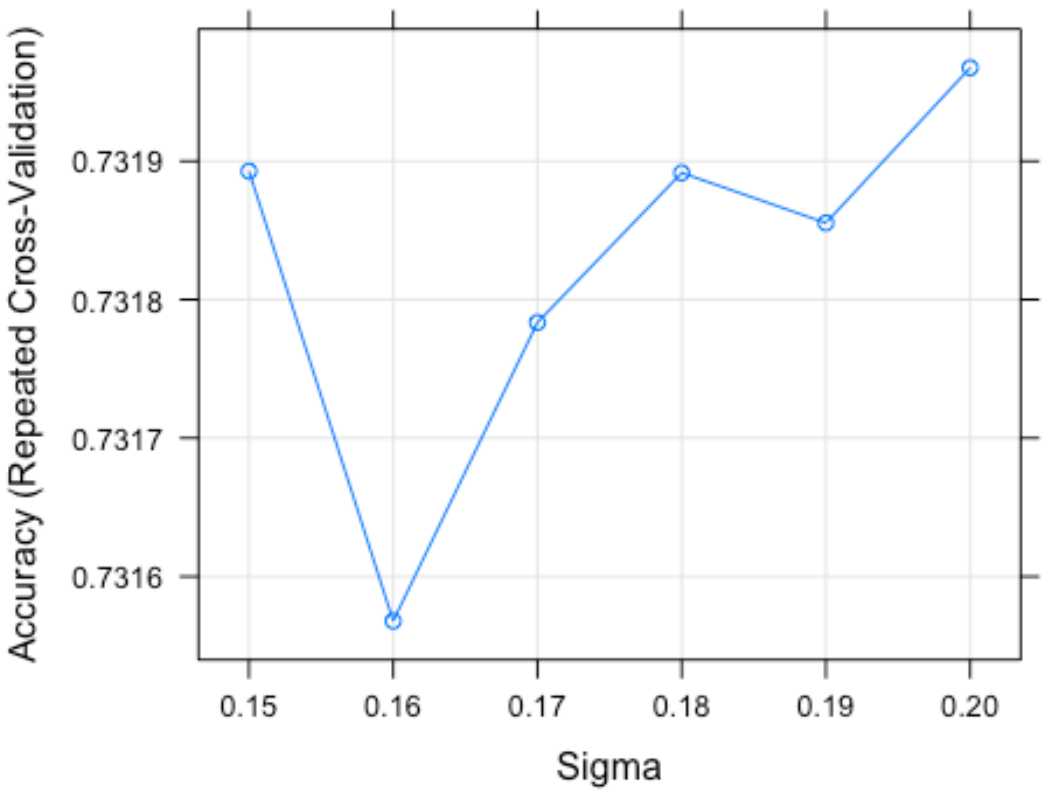
```r
head(with(model, results[order(results$Kappa, decreasing=T),]))
```

```
##      C sigma  Accuracy         Kappa  AccuracySD     KappaSD
## 6 1000  0.20 0.7319675 -0.0009247909 0.003245142 0.005847281
## 1 1000  0.15 0.7318927 -0.0011228088 0.003303443 0.005954271
## 4 1000  0.18 0.7318916 -0.0011250628 0.003220233 0.005957973
## 3 1000  0.17 0.7317835 -0.0013428478 0.003197757 0.004911107
## 5 1000  0.19 0.7318554 -0.0015325631 0.003189811 0.003792912
## 2 1000  0.16 0.7315678 -0.0017701785 0.003456779 0.005236414
```

```r
summary(model)
```

```
## Length  Class   Mode
##      1   ksvm     S4
```

```r
# graphe model
plot(model) # vérifier graphiquement, quel est le C qui maximise l'accuracy.
```




```r
#> model
# Support Vector Machines with Radial Basis Function Kernel
# Accuracy was used to select the optimal model using  the largest value.
```

```r
# The final values used for the model were sigma = 0.15 and C = 1000.

#====================> preuve : nb de labels correctements prédits (on cherche le max sigma = 0.15 and C
= 1000)


# sigma  Accuracy  Kappa     Accuracy SD  Kappa SD
# 0.15   0.728     0.000614  0.00166      0.00421 <====== kappa (+) fort
# 0.16   0.728     0.000000  0.00146      0.00000
# 0.17   0.728     0.000387  0.00199      0.00452

#**  prévision sur test
SVMtest = predict.train(object = model, newdata = x[indexes_y_test,], type='raw', y[indexes_y_test])
SVMtest

##    [1] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##   [35] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##   [69] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [103] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [137] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [171] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [205] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [239] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [273] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [307] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [341] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [375] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [409] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [443] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [477] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [511] N N N N N N N N N N N N N N N N N N N O N N N N N N N N N N N N N N
##  [545] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [579] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [613] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [647] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [681] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [715] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [749] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [783] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [817] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [851] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [885] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [919] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [953] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
##  [987] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1021] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1055] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1089] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1123] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1157] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1191] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1225] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1259] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1293] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1327] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1361] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1395] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1429] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1463] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1497] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1531] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1565] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1599] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1633] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1667] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1701] N N O N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1735] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1769] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1803] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1837] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1871] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1905] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
```

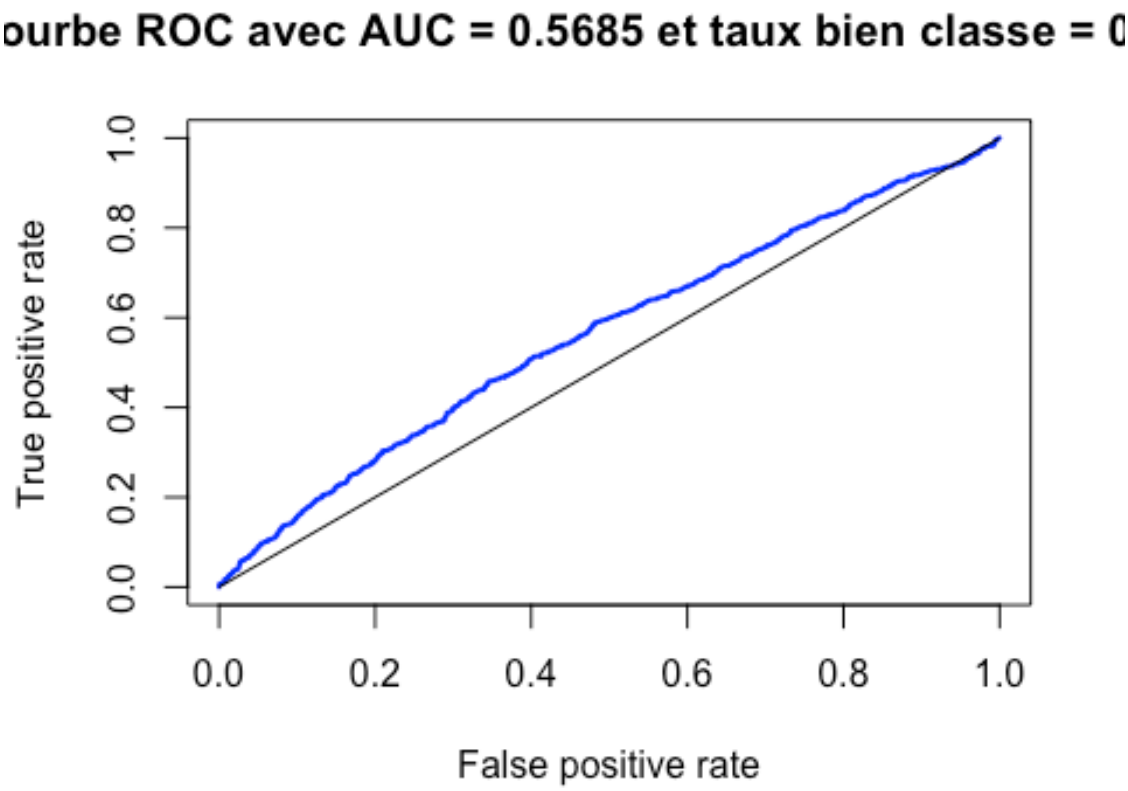
```
## [1939] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [1973] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2007] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2041] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2075] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2109] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2143] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2177] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2211] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2245] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2279] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2313] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2347] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2381] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2415] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2449] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2483] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2517] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2551] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2585] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2619] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2653] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2687] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2721] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2755] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2789] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2823] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2857] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2891] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2925] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2959] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [2993] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3027] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3061] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3095] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3129] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3163] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3197] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3231] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3265] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3299] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3333] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3367] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3401] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3435] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3469] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3503] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3537] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3571] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3605] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3639] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3673] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3707] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3741] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3775] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3809] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3843] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3877] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3911] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3945] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [3979] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [4013] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [4047] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [4081] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [4115] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [4149] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [4183] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [4217] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [4251] N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
## [4285] N N N N N N N N N N N N N N N N N N N N N
## Levels: N O
```

```
# Confusion Matrix and Statistics === FONCTION PACKAGE CARET
CM = confusionMatrix(data=SVMtest, y[indexes_y_test])
CM

## Confusion Matrix and Statistics
##
##           Reference
## Prediction    N    O
##          N 3191 1113
##          O    1    1
##
##                Accuracy : 0.7413
##                  95% CI : (0.7279, 0.7543)
##     No Information Rate : 0.7413
##     P-Value [Acc > NIR] : 0.5081
##
##                   Kappa : 9e-04
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.9996867
##             Specificity : 0.0008977
##          Pos Pred Value : 0.7414033
##          Neg Pred Value : 0.5000000
##              Prevalence : 0.7412912
##          Detection Rate : 0.7410590
##    Detection Prevalence : 0.9995355
##       Balanced Accuracy : 0.5002922
##
##        'Positive' Class : N
##

#Confusion Matrix and Statistics
#Reference
#Prediction    N    O
#N 3209 1097
#O    0    0
#Accuracy : 0.7452
#95% CI : (0.7319, 0.7582)
#No Information Rate : 0.7452
#P-Value [Acc > NIR] : 0.5081
#Kappa : 0
#Mcnemar's Test P-Value : <2e-16
#Sensitivity : 1.0000
#Specificity : 0.0000
#Pos Pred Value : 0.7452
#Neg Pred Value :    NaN
#Prevalence : 0.7452
#Detection Rate : 0.7452
#Detection Prevalence : 1.0000
#Balanced Accuracy : 0.5000
```

Positive' Class : N

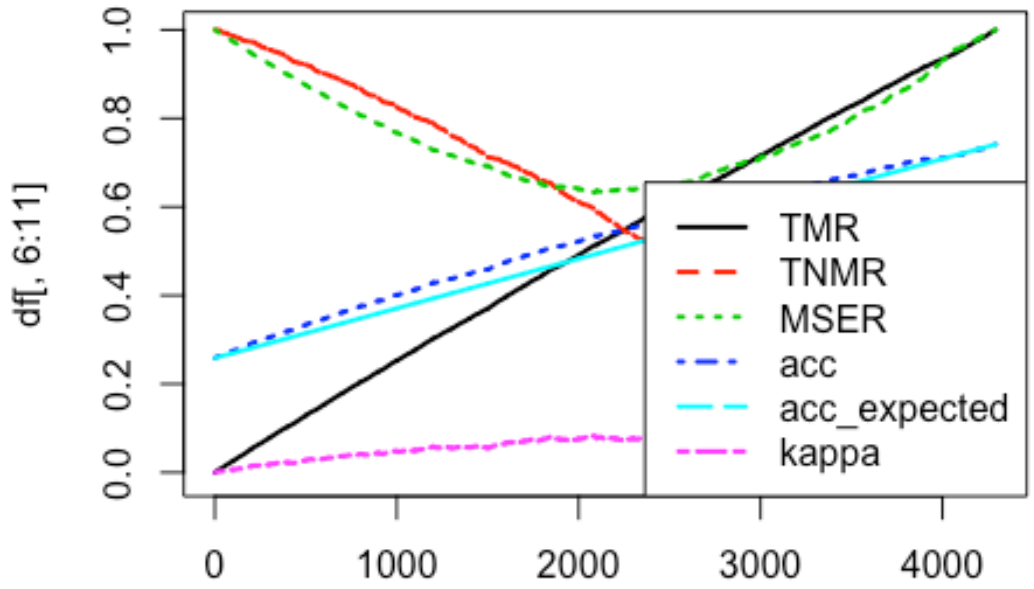
```
# confusion matrix: model predicting classes of test data
table.SVM=table(predict.train(object = model, newdata = x[indexes_y_test,], type='raw'), y[indexes_y_test
]) #
table.SVM

##
##        N    O
##   N 3191 1113
##   O    1    1

TxBcSVM=(sum(diag(table.SVM))/sum(table.SVM))*100
TxBcSVM

## [1] 74.12912

# svmLinear > [1] 30.55556
#********************************** svmRadial >[1] 74.52392
# svmPoly d=3  > [1] 1] 76.38889
```

```r
# prediction probabilities of test data classes
probs <- predict.train(object = model, newdata = x[indexes_y_test,], type='prob')[,1]
isPositiveClass <- y[indexes_y_test] == 'N' # for a ROC curve there is a positive class (true match rate.
..) - defining that class here
pred <- prediction(probs, isPositiveClass)
perf <- performance(pred, 'tpr', 'fpr')

# plot: either
plot(perf, col=4, lwd=2, main="courbe ROC avec AUC = 0.5685 et taux bien classe = 0.7452")
segments(0,0,1,1,lty=1)
```




```r
# or
#with(attributes(perf), plot(x=x.values[[1]], y=y.values[[1]], type='l'))


# some metrics: AUC (area under curve), EER (equal error rate), MSER (minimum squared error rate), Cohen'
s Kappa etc.

#***************
# definition de toutes les metrics
# https://machinelearningmastery.com/machine-learning-evaluation-metrics-in-r/
#***************

AUC <- attributes(performance(pred, 'auc'))$y.value[[1]] # area under curve
AUC

## [1] 0.5633323

# [1] 0.5685

df <- with(attributes(pred), data.frame(cutoffs=cutoffs[[1]], tp=tp[[1]], fn=fn[[1]], tn=tn[[1]], fp=fp[[
1]], TMR=tp[[1]]/(tp[[1]]+fn[[1]]), TNMR=tn[[1]]/(tn[[1]]+fp[[1]])))
df$MSER <- with(df, sqrt((1-TMR)**2+(1-TNMR)**2))
MSER <- with(df, sqrt(TMR**2+TNMR**2)) # sqrt of minimum squared error rate = eucl. distance to point TMR
=TNMR=1
i_eer <- with(df, which.min(abs(TMR-TNMR)))
EER <- with(df[i_eer,], mean(c(TMR,TNMR))) # equal error rate: mean would not be required when using ROCR
as it's always exact
df$acc <- with(df, (tp + tn) / length(isPositiveClass)) # observed accuracy
```

```
df$acc_expected <- with(df, sum(isPositiveClass) * (tp+fp) / length(isPositiveClass) + sum(!isPositiveCla
ss) * (tn+fn) / length(isPositiveClass)) / length(isPositiveClass) # expected accuracy
df$kappa <- with(df, (acc - acc_expected) / (1 - acc_expected)) # cohen's kappa

# graphical representation
matplot(df[,6:11], lty=1:6, lwd=2, type='l'); legend('bottomright', legend=names(df[,6:11]), lty=1:6, lwd
=2, col=1:6)
```




```
# characteristics for settings with best EER, MSER etc.
df[i_eer,] # curve point for equal error rate

##        cutoffs   tp   fn  tn  fp      TMR      TNMR      MSER      acc
## 2250 0.7260759 1748 1444 610 504 0.547619 0.5475763 0.6397935 0.547608
##      acc_expected      kappa
## 2250    0.5110951 0.07468294

df[order(df$acc, decreasing=T)[[1]],] # curve point for max accuracy

##        cutoffs   tp fn tn   fp TMR         TNMR      MSER       acc
## 4291 0.4962892 3192  0  1 1113   1 0.0008976661 0.9991023 0.7415235
##      acc_expected       kappa
## 4291    0.7411791 0.001330288

df[order(df$kappa, decreasing=T)[[1]],] # curve point for max kappa

##        cutoffs   tp   fn  tn  fp      TMR      TNMR      MSER       acc
## 2415 0.7250442 1882 1310 577 537 0.589599 0.5179533 0.6330861 0.5710636
##      acc_expected      kappa
## 2415    0.5298112 0.08773593

df[order(df$MSER, decreasing=F)[[1]],] # curve point for minimum squared error rate

##        cutoffs   tp   fn  tn  fp       TMR      TNMR      MSER       acc
## 2086 0.7268363 1638 1554 665 449 0.5131579 0.5969479 0.6320334 0.5348351
##      acc_expected      kappa
## 2086    0.4926032 0.08323244

toc <-  Sys.time() -tic
print(toc)
```

```
## Time difference of 15.08057 mins

#*****************************************************************************************************
*****************************************************************************************
#*****************************************************************************************************
*****************************************************************************************
#*****************************************************************************************************
*****************************************************************************************
```

11. Bibliographie

Référence	Détail
Rapport SFCR 2017 – Matmut Groupe	https://www.matmut.fr/le-groupe-Matmut/pdf/rapport-sfcr-matmut-groupe.pdf
L'actuariat avec R	La tarification a priori, p.37, Arthur Charpentier, Christophe Dutang, décembre 2012.
Modèles fréquence – coût	Quelles perspectives d'évolution ? Guillaume SERDECZNY, Maif 2014
Tarification	Les modélisations de risques innovantes : Maif, Actuaris, mai 2013
Data Sciences	Introduction au machine learning, p10, Ons Jelassi, Telecom Evolution.
Test de McNemar	http://pageperso.lif.univ-mrs.fr/~francois.denis/IAAM1/tp2M12016.pdf, francois denis, 2016
Test de McNemar - différencier des algo.	https://www.lri.fr/~mary/Présentation_diettrich.ppt, Thomas G. Dietterich,
Machine Learning Summarized in One Picture	http://www.datasciencecentral.com/profiles/blogs/machine-learning-summarized-in-one-picture
DATA MINING ET BIG DATA	Stéphane TUFFERY, UCO, Institut de Mathématiques Appliquées, 2015
Statistiques inférentielles 1,2,3 - TP	Changment de variable pour normaliser les Xi ~ N (μ, σ), Michel BOBBIA, 2017
Analyse Fonctionnelle - TP	Transformation des Xi selon la stabilisation de la variance, Jean-Marc.Lasgouttes, 2017
Proba, ADD et statistiques	P287, méthode delta, stabilisation de la variance, normalisation, Gilbert SAPORTA, 2006
Les différentes transformations des données	https://lemakistatheux.wordpress.com/2015/11/06/les-differentes-transformations-des-donnees-redaction-pour-le-071115/
Aperçu des méthodes de sélection de var. (R)	https://lilbigdataboy.wordpress.com/2016/01/04/apercu-des-methodes-de-selection-de-variables-avec-r/
SVM avec code R	An Introduction to Statistical Learning with Applications in R, Daniela Witten Trevor Hastie Robert Tibshiran, Springer, 2013
The Elements of Statistical Learning	The Elements of Statistical Learning: Data Mining, Inference and Prediction, Springer, Trevor Hastie, Robert Tibshirani, 2013
Analyse des leviers	Variables utiles, variables discriminantes, variable effets de colinéarité et hiérarchisation des impacts dans les études de marché et sociales, HAL Id: tel-01329190, 2016
Arbres CART et Forêts aléatoires	Arbres Cart, forêts aléatoires, importance et sélection de variables, 23 janvier 2017
Random Forests for Big Data	Robin Genuer, Jean-Michel Poggi, hal-01580178, version 1
Data Science	Fondamentaux et études de cas – Eric Biernat / Michel Lutz, 2015.
Do we Need Hundreds of Classifiers?	Do we Need Hundreds of Classifiers to Solve Real World Classification Problems? Manuel Fernandez-Delgado, 2014
Learning from Imbalanced Data	HAIBO HE, IEEE Transactions on Knowledge and Data Engineering (Vol.: 21, Issue: 9, Sept. 2009)
R package : ROSE	https://journal.r-project.org/archive/2014/RJ-2014-008/RJ-2014-008.pdf, Nicola Lunardon
QDA : analyse discri. avec faible effectif	http://emily.perso.math.cnrs.fr/Modelisation/Modelisation_files/AnalyseDiscriminante.pdf
3.9 Séparateurs à Vaste Marge (SVM)	https://moodle.insa-toulouse.fr/pluginfile.php/30362/mod_resource/content/0/TpOzoneRIup.pdf, p26, Helene Milhem, 2012
> Statistiques > Apprentissage > SVM	http://www.duclert.org/r-apprentissage/svm-support-vector-machines-R.php
Comment choisir les algo. dans Ms Azure ML	https://docs.microsoft.com/fr-fr/azure/machine-learning/machine-learning-algorithm-choice
Évaluation des performances d'un modèle	https://docs.microsoft.com/fr-fr/azure/machine-learning/machine-learning-evaluate-model-performance
A Short Introduction to the caret Package	https://cran.r-project.org/web/packages/caret/vignettes/caret.pdf
Predictive Modeling with R and the caret Pkg	https://www.r-project.org/conferences/useR-2013/Tutorials/kuhn/user_caret_2up.pdf
Package "caret"	https://cran.r-project.org/web/packages/caret/caret.pdf
Les SVM	Une nouvelle méthode d'apprentissage : Antoine Cornuéjols / afia – numero 51 – juin 2002
Évaluation de l'apprentissage	méthodes A. Cornuéjols -- LRI
Matrice Gram, matrice noyau	https://www.lri.fr/~antoine/Courses/Master-ISI/chap-14-svm.pdf
SVM avec sélection variable – TP Lasso,	**SVM linéaire + sélection de variables [rajouter Une pénalisation], Stéphane Canu, 2017**
Tentative syllabus for the course: Understanding SVM	**http://asi.insa-rouen.fr/enseignants/~scanu/ - Stéphane Canu, 2017**
Multi Class SVM	Lorette Noiret, Stéphane Canu, Gille Gasso, March 22, 2017
Méthodes à noyaux	https://www.di.ens.fr/~fbach/rasma_fbach.pdf,
Machines à vecteurs supports	https://www.math.univ-toulouse.fr/~besse/Wikistat/pdf/st-m-app-svm.pdf
Agrégation de modèles	Philippe BESSE - - https://www.math.univ-toulouse.fr/~besse/Wikistat/pdf/st-m-app-agreg.pdf]
Arbres CART et Forêts Aléatoires	Importance et sélection de variables – R Genuer & Jean-Michel POGGI - hal0138765v2
Méthodes de sondage et d'enquête	ULB, fun-mooc.fr, 44002, réalisation d'un sondage PESR
Taille opti. échantillon quotas : n=1000	https://www.franceculture.fr/emissions/la-methode-scientifique/peut-calculer-le-hasard
Relaxations de la régression logistique	Farid Beninel, Christophe Biernacki, CREST ENSAI
Interpréter la courbe ROC	https://www.youtube.com/watch?v=t3R96Kpm2mw&ab_channel=ThierryAncelle
PB linéaire séparable – minute 15	https://www.college-de-france.fr/site/yann-lecun/course-2016-02-12-14h30.htm
projection avec 1 noyau dans un espace en grande dimension	https://en.wikipedia.org/wiki/Cover%27s_theorem
A Practical Guide to SVM Classification	http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf
calcul en parallèle	http://archimede.mat.ulaval.ca/dokuwiki/lib/exe/fetch.php?media=r:amelioration:optimtempsr2017.pdf
Problème d'optimisation et lagrangien	Pattern recognition and machine learning / C. BISHOP / Springer 2006
Imbalanced Data Classif. Fuzzy SVM and SMOTE	IEEE International Conference on Systems, Canada, October 5-8, 2017, Ratchakoon Pruengkarn
Tests de comparaison de classifieurs	Thomas G. Dietterich Approximate Statistical Tests for Comparing, 1998
Data Mining (Concis et Pratique)	Julien Ah-Pine (julien.ah-pine@univ-lyon2.fr) - L3 CESTAT - 2016/2017
Présentation des SVM, par l'auteur des SVM	The Nature of Statistical Learning Theory, Vladimir Vapnik, springer, 1995

12. Glossaire des principaux termes et concepts

https://glossary.informs.org/ver2/mpgwiki/index.php?title=Box_constraint

TERME	EXPLICATIONS
Dataset	Jeu de données
Capacité d'un modèle	habilité d'un modèle à réduire son erreur d'entraînement, à mémoriser ces données
Capacité et erreur en modèle d'apprentissage	Plus la capacité est grande, plus la différence entre l'erreur d'entraînement et l'erreur test augmente
Modèle paramétrique	modèle dont la capacité n'augmente pas avec le nombre de données (régression logistique, SVM)
Modèle non-paramétrique	l'inverse de paramétrique, la capacité augmente avec la taille de l'ensemble d'entraînement (réseau de neurones
Pipeline (en data science)	« Generically speaking a pipeline has inputs go through a number of processing steps chained together in some way to produce some sort of output » - https://www.quora.com/What-is-a-data-analysis-pipeline
…	…

13. Certificat Mooc - Méthodes de sondage et d'enquête proposé par ULB

FUN

ULB UNIVERSITÉ LIBRE DE BRUXELLES

ATTESTATION DE SUIVI AVEC SUCCÈS

didier LE PICAUT
a suivi avec succès le MOOC*

Méthodes de sondage et d'enquête
proposé par Université libre de Bruxelles (ULB)
et diffusé sur la plate-forme FUN
Le 02/05/2016

Enseignants

Catherine Vermandele
Docteur en statistique

Nezar Bennala
Docteur en statistique

https://www.fun-mooc.fr

* MOOC : cours en ligne

La présente attestation n'est pas un diplôme et ne confère pas de crédits (ECTS). Elle n'atteste pas que le participant était inscrit à/au Université libre de Bruxelles (ULB).. L'identité du participant n'a pas été vérifiée.