Méthodes D'évaluation Appliquées Au Marché Du Travail

EVALUATION DE LA PUISSANCE FISCALE (CV)

Machine Learning avec R

PROFESSEUR : DANIEL Christophe

Réalisé par

BARRY Mamadou Yaya

SOMMAIRE

Introduction

A travers les nouvelles technologies, les datas s'accumulent de jour en jour. Ces données représentent la nouvelle mine d'or du XXI^{e} siècle. De ce fait, économètres, informaticiens et autres, accourent à leur traitement.

C'est dans ce processus que de nouvelles méthodes et algorithmes ne cessent d'être mis au point pour la recherche d'éventuelles relations de causalité entre différents phénomènes (économiques, sociales, démographiques…). Et notamment pour tester la pertinence empirique d'un modèle.

Raison de plus, d'introduire avec vous, une méthode d''‘Apprentissage automatique'' communément connue sous le nom de « Machine Learning » sur un jeu de données automobiles.

L'apprentissage automatique, par le biais d'*algorithmes* (succession d'étapes exécutées dans l'ordre), entraînent nos données à des fins de prédiction via des mécanismes de modélisation et d'ajustement tout en corrigeant les biais de sélection pouvant apparaître dans nos données.

Dans un premier temps, nous présenterons l'étude économique, avant d'aborder dans un second temps le processus de traitement de nos données ainsi que d'entraînement de notre modèle à travers différentes méthodes de régression qui nous permettront donc de confirmer ou infirmer notre approche ou choix méthodologique. Et en dernier lieu, dans une conclusion, nous effectuerons un rapprochement entre l'étude réalisée et les littératures existantes sur l'évaluation de la puissance fiscale ou administrative (notée en CV) des véhicules.

Le but principal reste d'explorer plusieurs possibilités de traitement des données et de méthodes vues en et en dehors du cours .

I. Étude Économique

Les méthodes d'évaluation sont fréquemment utilisées pour estimer les effets d'une politique d'interventions publiques en termes d'éducation, assurance, emploi etc. Elles ont donné lieu ces dernières années à d'énormes avancées méthodologiques[1] . Les techniques disponibles sont certes variées, néanmoins, il est important de rappeler que leurs spécificités et leurs hypothèses conditionnent fortement les résultats. Pour la plupart, ces méthodes sont fondées sur une comparaison entre, d'un côté : des individus, ménages ou entreprises bénéficiant d'une nouvelle réforme ou de l'intervention publique que l'on souhaite évaluer ; et de l'autre : d'individus, ménages ou entreprises n'en bénéficiant pas.

Dans notre cas, nous voulons évaluer la puissance fiscale ou administrative d'un véhicule tout en essayant de voir quels paramètres s'ajustent le mieux au modèle de départ afin de l'optimiser.

I.1. Régression quantile

La régression quantile[2] est un dispositif d'évaluation en statistique, ayant pour objet d'expliquer l'impact des facteurs explicatifs sur une variable d'intérêt. La régression quantile ne s'intéresse pas qu'à la moyenne, elle décrit de manière plus riche la variable d'intérêt en s'intéressant à l'ensemble de la distribution conditionnelle de celle-ci. Elle permet à l'économètre de faire face aux limites de la régression par la moyenne.

Les quantiles les plus souvent utilisés sont la médiane (q=0,5), les premiers et derniers déciles (q=0,1 et q=0,9) ainsi que les premiers et derniers quartiles (q=0,25 et q=0,75).

I.2. Présentation des données

Nous avons à faire à une base de données nationale (data.gouv.fr) de 2015 constituée de 20880 observations de 18 variables. Les variables sont réparties de manière suivantes:

[1] Heckman, Lalonde et Smith (1999)

[2] La régression quantile a été formalisée par Koenker et Bassett (1978) comme une méthode d'extension de la relation entre une ou plusieurs variables indépendantes Xj (j = 1, . . ., p) et la moyenne conditionnelle de variable dépendante Y sachant Xj = xji (régression standard) au lien entre ces variables Xj et les quantiles conditionnels de Y sachant Xj = xji.

- **Les variables actives :** qui sont principalement des variables quantitatives..
  - «*puiss_admin*» la puissance administrative ou puissance fiscale nationale du véhicule. Elle est une mesure qui sert à évaluer la puissance du moteur d'un véhicule et se reconnaît généralement par l'abréviation CV.
  - « *puiss_max*» la puissance maximale (en KW).
  - «*conso_urb*» la consommation urbaine de carburant (en l/100km).
  - «*conso_exurb*» la consommation extra urbaine de carburant (en l/100km).
  - «*conso_mixte*» la consommation mixte de carburant (en l/100km).
  - «*co2*» l'émission de CO2 (en g/km).
  - «*co_typ_1*» le résultat d'essai de CO type I (en g/km).
  - «*hc*» les résultats d'essai de HC (en g/km).
  - «*nox*» les résultats d'essai de NOX (en g/km).
  - «*hcnox*» les résultats d'essai de HC+NOX (en g/km).
  - «*ptcl*» le résultat d'essai de particules (en g/km).

  Quant à la variable «*energ*», elle a été transformée en 5 variables dummies, à savoir:
  - «*hyb*» la variable hybride qui prend la modalité 1 si les véhicules sont hybrides et 0 sinon.
  - «*elec*» la variable électrique qui prend la modalité 1 si l'énergie utilisée par les véhicules est électrique et 0 sinon.
  - «*ess*» la dummy essence qui prend pour 1 la modalité des véhicules à essence et 0 sinon.
  - «*gazoil*» cette dummy prend la modalité 1 si les véhicules consomment du gazoil et 0 sinon.
  - «*AT_energ*» cette variable prend la modalité 1 si les véhicules consomment d'autres types d'énergies différentes des 4 premiers et 0 sinon.
- **Les variables qualitatives :** «*lib_mrq_doss*» et «*lib_mod_doss*» seront codés en une seule variable «*marque*» qui est une association des deux.

II. Étude Économétrique

Un modèle économique est une représentation simplifiée de la réalité économique ou d'une partie de celle-ci : par exemple la croissance, le commerce international, la monnaie, une entreprise ou un ménage.

Afin de mener à bien notre étude, nous pouvons établir notre modèle économique (qui nous permettra de décrire les liens divers entre les variables) comme suit :

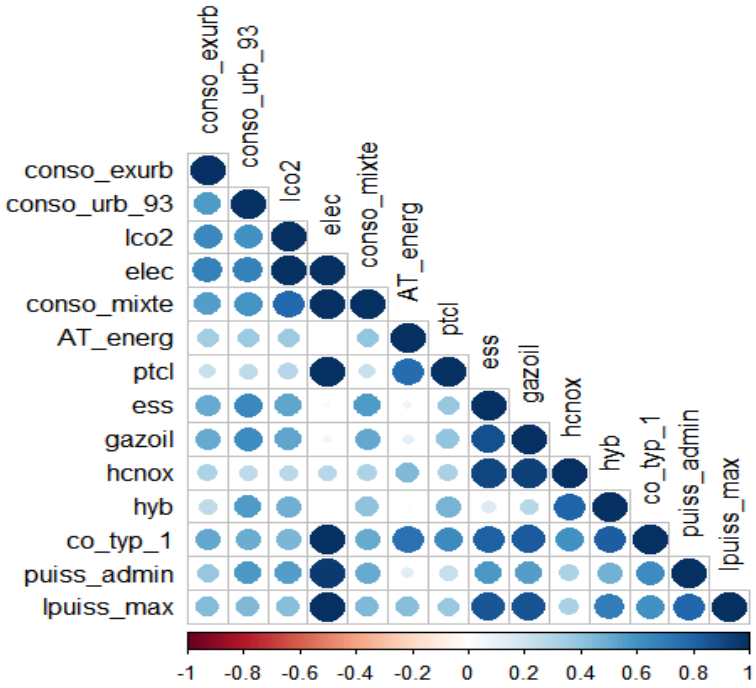
$Y = f(X_1, X_2, X_3, \ldots, X_n)$

Notre variable d'intérêt (la puissance fiscale) sera fonction de plusieurs variables explicatives.

II.1. Traitement des données

Notre base de données est constituée de plusieurs données manquantes. Nous avons donc le choix entre supprimer les observations contenant des données manquantes (ce n'était pas pertinent en soi car cela aurait réduit le nombre d'observations de 20880 à 11729.) ou de procéder à la gestion des données manquantes par la méthode d'imputation dans **caret** (library dans R) via des arbres de décisions « ensachés » (*bagging* ou encore *bootstrap aggregating*). Ce qui permettra de remplacer les ***NA****s* par des valeurs prédites par l'algorithme de bagging (méthode plus précise).

II.2. Analyses descriptives



La matrice V de cramer nous permet d'analyser les relations entre les variables afin de vérifier si elles sont bien toutes indépendantes entre-elles. Cette matrice a pour but d'étudier le lien numérique entre deux variables, via le Khi2, puis en renormalisant les résultats entre 0 et 1.

Ici nous pouvons constater que nos V de Cramer sont proches de 0, ce qui signifie que nos variables sont plus ou moins indépendantes les unes des autres. A la seule exception des variables dummy qui sont fortement corrélées avec toutes les autres variables.

Le but ici, est de faire l'analyse statistique tout en décrivant la structure des données, c'est-à-dire définir la composition des variables, la significativité des variables, les nuages de points et les corrélations. Dans un premier temps nous ferons une analyse des statistiques descriptives de l'ensemble des variables (nombre d'observations, minimum, maximum, moyenne, médiane et quartiles), ensuite nous ferons les tests nécessaires pour notre étude afin de déterminer le modèle correspondant.

```
   marque          hyb       ess       gazoil    elec      AT_energ   puiss_admin
Length:20880       0:19857   0:13946   0: 8205   0:20824   0:20688   Min.   : 1.00
Class :character   1: 1023   1: 6934   1:12675   1:   56   1:  192   1st Qu.: 8.00
Mode  :character                                                     Median : 9.00
                                                                     Mean   :12.39
                                                                     3rd Qu.:12.00
                                                                     Max.   :80.00
 conso_urb_93     conso_exurb      conso_mixte       co_typ_1          hcnox
Min.   : 0.000   Min.   : 2.600   Min.   : 0.60   Min.   :0.0000   Min.   :0.00000
1st Qu.: 5.700   1st Qu.: 4.300   1st Qu.: 4.80   1st Qu.:0.1770   1st Qu.:0.09034
Median : 7.100   Median : 5.100   Median : 5.80   Median :0.2490   Median :0.10233
Mean   : 7.725   Mean   : 5.339   Mean   : 6.21   Mean   :0.2569   Mean   :0.13918
3rd Qu.: 8.600   3rd Qu.: 6.200   3rd Qu.: 7.10   3rd Qu.:0.3080   3rd Qu.:0.19695
Max.   :27.300   Max.   :13.700   Max.   :17.20   Max.   :0.9680   Max.   :0.30100
     ptcl            lpuiss_max        lco2
Min.   :0.0000000   Min.   :3.332   Min.   :2.565
1st Qu.:0.0000000   1st Qu.:4.605   1st Qu.:4.804
Median :0.0002165   Median :4.787   Median :4.990
Mean   :0.0006833   Mean   :4.873   Mean   :5.003
3rd Qu.:0.0010000   3rd Qu.:5.011   3rd Qu.:5.193
Max.   :0.0040000   Max.   :6.372   Max.   :5.986
```

NB : On remarque que la variable « *elec* » est presque qu'inexistante car seulement 56 voitures sur 20880 sont électriques. De même pour la variable « *ptcl* » qui a une moyenne presque nulle.

II.3. Régression multivariée

La régression linéaire multiple est une analyse statistique qui décrit les variations d'une variable endogène associée aux variations de plusieurs variables exogènes.

```
Call:
lm(formula = puiss_admin ~ ., data = projet)

Residuals:
     Min       1Q   Median       3Q      Max
-19.6456  -1.5532  -0.5094   1.1248  24.8322

Coefficients: (1 not defined because of singularities)
              Estimate Std. Error t value Pr(>|t|)
(Intercept)    25.39747    1.59223  15.951   <2e-16 ***
hyb1            8.89157    0.25991  34.210   <2e-16 ***
ess1            8.91472    0.23987  37.165   <2e-16 ***
gazoil1        11.63483    0.25153  46.257   <2e-16 ***
elec1          13.14265    0.46330  28.368   <2e-16 ***
AT_energ1            NA         NA      NA       NA
conso_urb_93   -0.77392    0.08454  -9.154   <2e-16 ***
conso_exurb    -2.71370    0.12530 -21.658   <2e-16 ***
conso_mixte     8.32491    0.23332  35.681   <2e-16 ***
co_typ_1        3.13004    0.18582  16.845   <2e-16 ***
hcnox          12.54315    0.51182  24.507   <2e-16 ***
ptcl         -475.54408   23.59609 -20.154   <2e-16 ***
lpuiss_max     16.27599    0.09374 173.629   <2e-16 ***
lco2          -27.24091    0.43465 -62.673   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.985 on 20867 degrees of freedom
Multiple R-squared:  0.9064,    Adjusted R-squared:  0.9063
F-statistic: 1.683e+04 on 12 and 20867 DF,  p-value: < 2.2e-16
```

```
Call:
lm(formula = puiss_admin ~ ., data = projet[, -5])

Residuals:
     Min       1Q   Median       3Q      Max
-19.6456  -1.5532  -0.5094   1.1248  24.8322

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)    25.39747    1.59223  15.951   <2e-16 ***
hyb1            8.89157    0.25991  34.210   <2e-16 ***
ess1            8.91472    0.23987  37.165   <2e-16 ***
gazoil1        11.63483    0.25153  46.257   <2e-16 ***
elec1          13.14265    0.46330  28.368   <2e-16 ***
conso_urb_93   -0.77392    0.08454  -9.154   <2e-16 ***
conso_exurb    -2.71370    0.12530 -21.658   <2e-16 ***
conso_mixte     8.32491    0.23332  35.681   <2e-16 ***
co_typ_1        3.13004    0.18582  16.845   <2e-16 ***
hcnox          12.54315    0.51182  24.507   <2e-16 ***
ptcl         -475.54408   23.59609 -20.154   <2e-16 ***
lpuiss_max     16.27599    0.09374 173.629   <2e-16 ***
lco2          -27.24091    0.43465 -62.673   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.985 on 20867 degrees of freedom
Multiple R-squared:  0.9064,    Adjusted R-squared:  0.9063
F-statistic: 1.683e+04 on 12 and 20867 DF,  p-value: < 2.2e-16
```

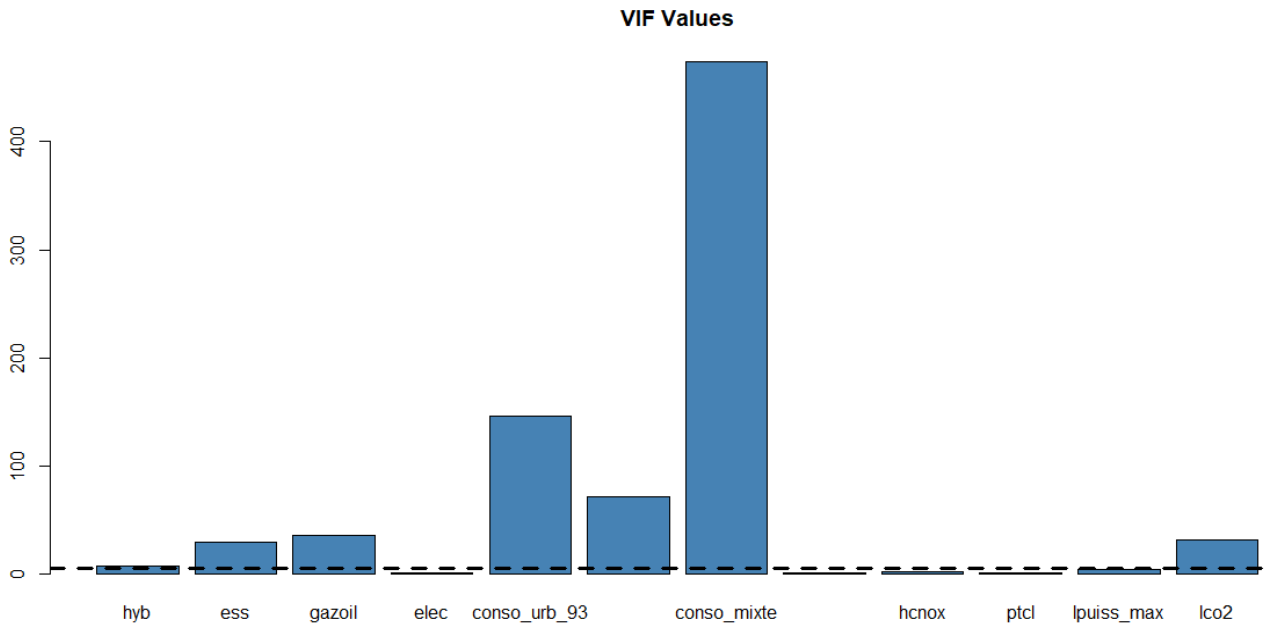
NB : on constate que la variable « *AT_energ* » apparait en NA malgré l'absence des données manquantes. Cela est dû à un problème de multi-colinéarité entre les variables, causé à la suite des traitements des données manquantes via des valeurs prédites.

Après retrait de la dite variable, nous obtenons un modèle dont les variables sont toutes significatives avec une distribution de 90,63% de la variabilité de ses paramètres.

Le modèle semble tenir la route. Cependant nous ne pouvons conclure cela avant de vérifier le test VIF (Variance Inflation Factor).

```
vif(lm.all2)
       hyb           ess        gazoil          elec conso_urb_93  conso_exurb
  7.377667     29.912688     35.373274      1.345712    146.610100     71.220213
conso_mixte      co_typ_1         hcnox          ptcl    lpuiss_max          lco2
 473.820271      1.134759      2.536684      1.120056      4.001869     31.977088
```

Nos doutes sont finalement confirmés par le VIF (facteur d'inflation de la variance). Ce qui en ressort du résultat est inquiétant car les valeurs de certaines variables dépassent largement 5. Du coup, les estimations des coefficients ainsi que la p-value de notre modèle de régression sont probablement peu fiables.



Plot des valeurs du VIF pour une meilleure visualisation du problème.

Régularisation du modèle de régression

Cette technique consiste à rééchelonner les coefficients en fonction de leur impact réel sur le modèle de régression.

	Value	Std. Error	Z score	Pr(>|Z|)
(Intercept)	-8.826849	1.38050560	-6.393925	1.616811e-10
hyb1	3.440908	0.22750776	15.124353	0.000000e+00
ess1	3.390536	0.20172764	16.807492	0.000000e+00
gazoil1	5.260601	0.20036099	26.255616	0.000000e+00
elec1	7.048885	0.44911770	15.694961	0.000000e+00
conso_urb_93	1.103987	0.07229919	15.269697	0.000000e+00
conso_exurb	0.000000	0.10830417	0.000000	1.000000e+00
conso_mixte	2.222187	0.18181721	12.222094	0.000000e+00
co_typ_1	3.647312	0.19015307	19.180929	0.000000e+00
hcnox	5.673164	0.49491906	11.462812	0.000000e+00
ptcl	-482.942287	24.20386098	-19.953109	0.000000e+00
lpuiss_max	15.525114	0.09423162	164.754826	0.000000e+00
lco2	-16.525493	0.35229777	-46.907742	0.000000e+00

Cela peut entraîner dans certains cas, la réduction de certaines caractéristiques à une valeur si faible qu'elles s'approchent de zéro, comme c'est le cas pour la variable « *conso_exurb* » dont la valeur du coefficient est de 0. On la retire de la régression multiple et on recommence le même processus jusqu'à obtention d'un résultat final.

```
lm.lasso5<-l1ce(puiss_admin ~. -conso_exurb-conso_mixte-ess-hyb-gazoil-elec-hcnox-co_typ_1-ptcl-lco2, data = projet[,-5])
summary(lm.lasso5)$coefficients
```

```
$coefficients
                    Value Std. Error   Z score Pr(>|Z|)
(Intercept)  -34.4749675 0.43812114 -78.68821        0
conso_urb_93   0.3114443 0.01684945  18.48394        0
lpuiss_max     9.1233393 0.11308017  80.68028        0
```

Le résultat final n'a que 2 caractéristiques au lieu de 12 :

$$puiss\_admin = -34.47 + 9.12lpuiss\_max + 0.31conso\_urb\_93$$

La caractéristique la plus importante est la puissance maximale suivi de la consommation urbaine.

En remplaçant ces deux caractéristiques par leur valeur moyenne on obtient une valeur d'environ ***12.37*** pour la puissance administrative qui se rapproche de sa moyenne initiale (***12.39***).

$$puiss\_admin = -34.47 + 9.12*4.873 + 0.31*7.725 = 12.37$$

Cette modélisation qui stipule que la puissance administrative est fonction de la puissance maximale de la voiture ainsi que de sa consommation urbaine (au lieu de son émission en CO_2) semble tenir la route parce que tout simplement toute consommation de carburant implique forcément une émission de CO_2. Cependant le modèle demeure moins inclusif car ne tient pas en compte des autres types de consommation.

II.4. Analyse Par Composantes Partielles

L'analyse en composantes principales (ou PCA) est un type d'apprentissage automatique dont nous nous servons comme étape de prétraitement des données pour aider avec quelques approches différentes. Injecter ces données dans un modèle d'apprentissage automatique sans aucune procédure de présélection préalable de caractéristiques pourrait entraîner une erreur supplémentaire lors de la modélisation, précisément du fait de la forte corrélation entre nos caractéristiques. D'où la nécessité d'exécuter un algorithme PCA pour réduire la dimensionnalité des données en un plan bidimensionnel.

II.4.1. Algorithme PCA

L'un des premiers objectifs est de visualiser en quoi certain nombre de composantes principales peut expliquer la variance dans nos données.

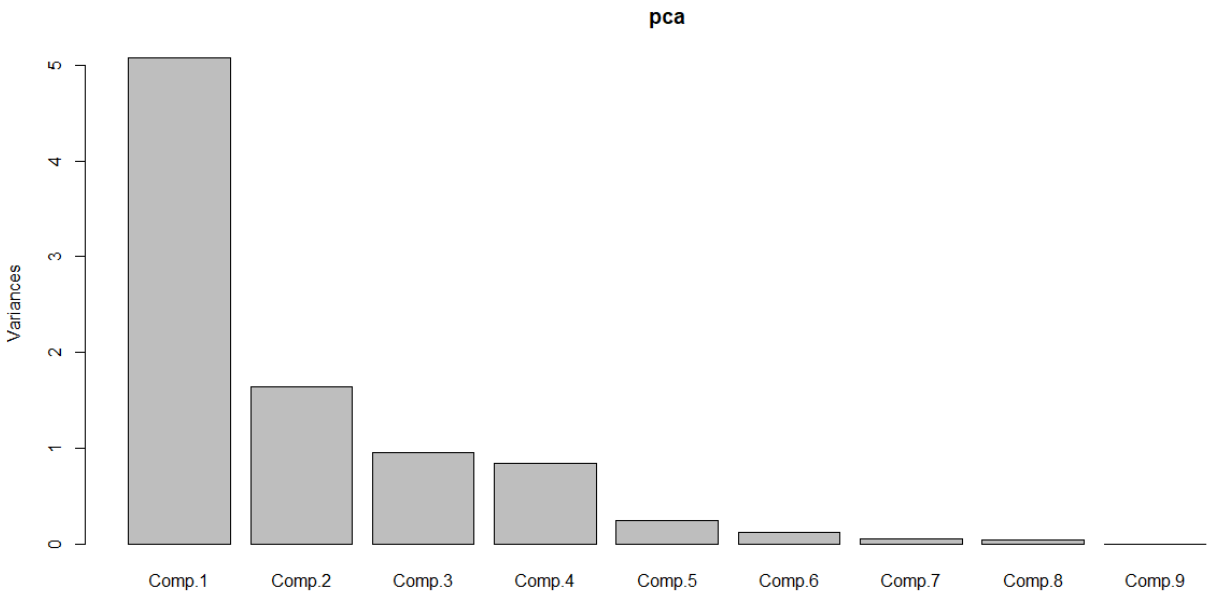
Jetons un coup d'œil à la sortie de l'objet pca :

```
summary(pca)
```

```
Importance of components:
                          Comp.1    Comp.2    Comp.3    Comp.4    Comp.5
Standard deviation     2.2523351 1.2839674 0.9787237 0.91820022 0.49900523
Proportion of Variance 0.5636682 0.1831747 0.1064333 0.09367685 0.02766736
Cumulative Proportion  0.5636682 0.7468429 0.8532762 0.94695303 0.97462039

                           Comp.6      Comp.7      Comp.8       Comp.9
Standard deviation     0.34912216 0.239828646 0.217650346 0.0405062407
Proportion of Variance 0.01354292 0.006390864 0.005263519 0.0001823062
Cumulative Proportion  0.98816331 0.994554175 0.999817694 1.0000000000
```

Les composantes sont toujours triées en fonction de leur poids, de sorte que les plus importantes apparaîtrons en premier. Dans la sortie précédente, nous pouvons voir que la composante.1 explique environ 56,37% des données, la composante.2 suivant avec 18,32 %, le reste devenant très rapidement marginal. Combinées, on arrive à expliquer plus de 74,68% du jeu de données avec seulement deux caractéristiques au lieu de 12 comme auparavant.

II.4.2. Histogramme



Variances de nos diverses composantes. C'est une façon plus visuelle de représenter dans quelle mesure nos composantes expliquent les données.

II.4.3. Visualisation des données

```
pca$loadings[,1:4]
```

	Comp.1	Comp.2	Comp.3	Comp.4
puiss_admin	0.37947155	0.31381904	0.103458312	0.06966640
conso_urb_93	0.42805506	-0.02293178	0.045693650	-0.06149206
conso_exurb	0.41939650	-0.16075896	-0.054831877	-0.12894309
conso_mixte	0.43493815	-0.08998391	0.001965433	-0.09521722
co_typ_1	0.09502520	-0.28973657	0.720313742	0.61336574
hcnox	0.02237538	-0.69775069	0.048171007	-0.34350717
ptcl	0.10616201	-0.28516874	-0.676221485	0.66081647
lpuiss_max	0.34241199	0.42613440	-0.010099538	0.10842179
lco2	0.41189659	-0.18258193	-0.075154780	-0.15173753

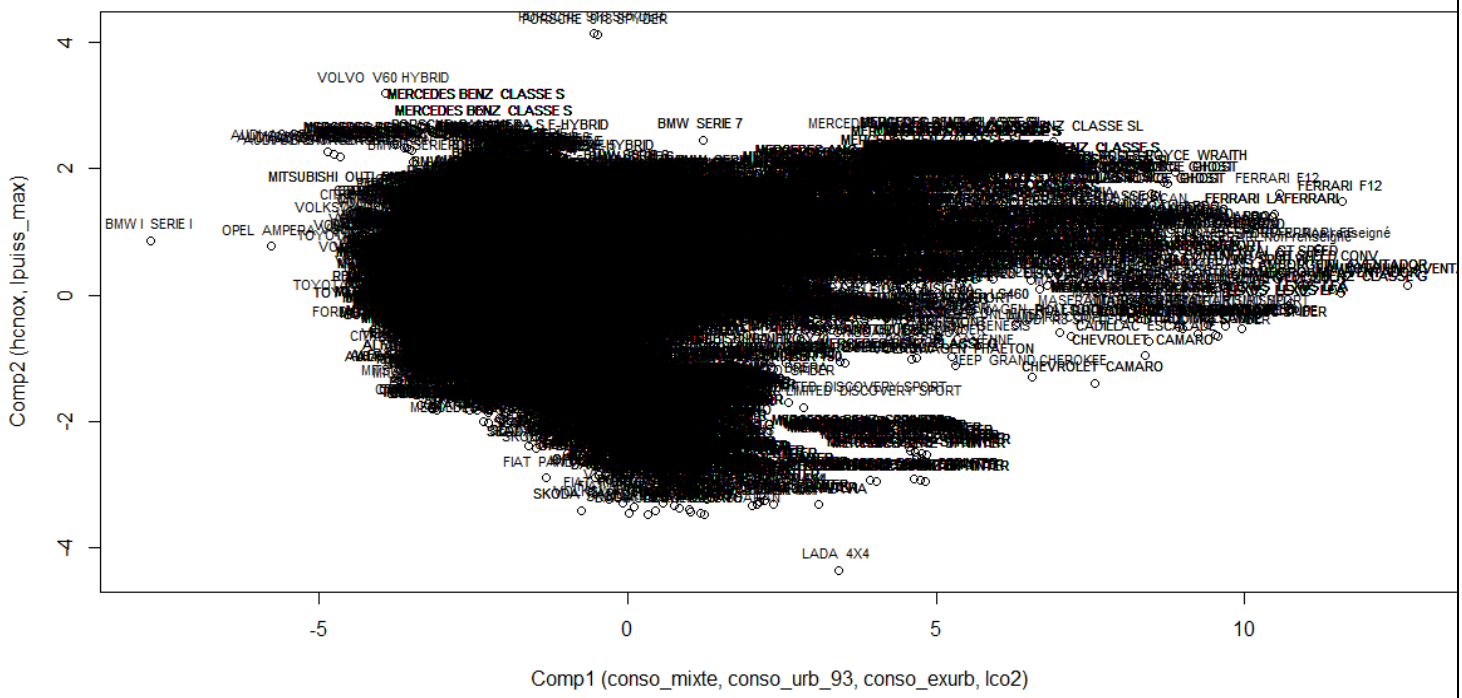
Ces valeurs appelés « Loadings » en anglais, reflètent (mais ne sont pas) les corrélations entre les composantes principales et les caractéristiques avec lesquelles nous avons modéliser notre

modèle. Ils correspondent aux coordonnées sur les axes factoriels. On considère qu'une valeur supérieure à 0,4 (en valeur absolue) indique une liaison significative. Ce seuil est arbitraire (on descend parfois à 0,25). Les variables les plus excentrées sont les plus représentatives (ont une CTR plus élevée). Cet exemple ne montre que les quatre premières composantes principales pour économiser de l'espace , les autres (de 5 à 6) n'étant de toute façon pas vraiment utiles.

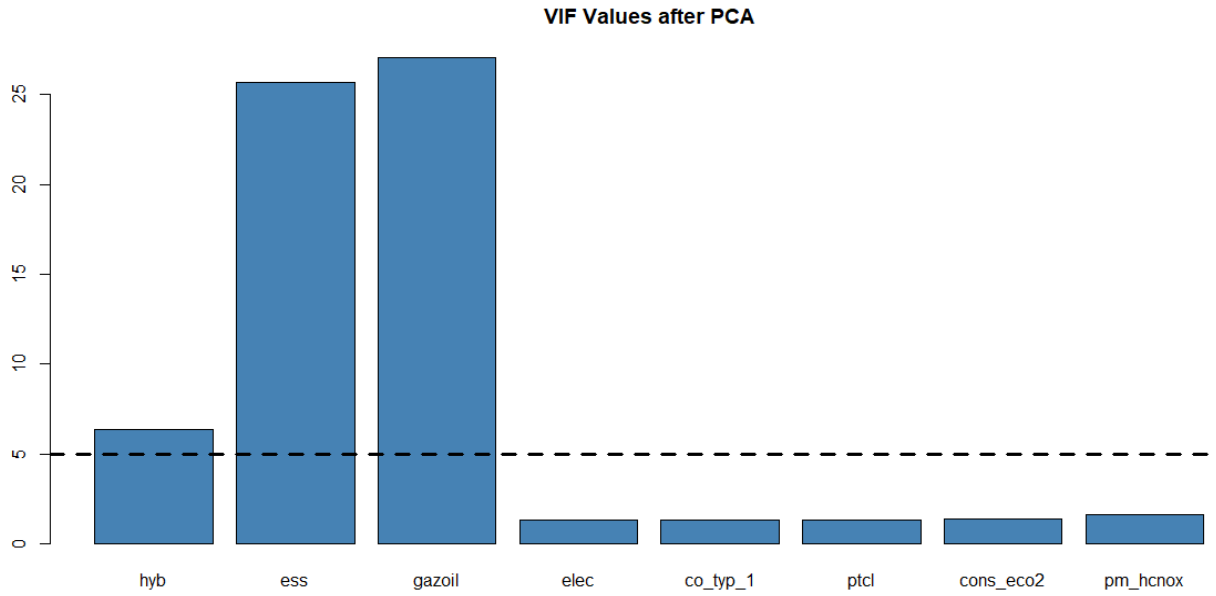
Plus la valeur de corrélation est proche de 1 ou -1 pour chaque combinaison de composantes et de caractéristiques, et plus cette caractéristique est importante pour cette composante.

Si par exemple, nous devrions attribuer une sorte de relation entre les composantes et les caractéristiques, nous pourrions dire quelque chose comme ceci :

- L'axe 1 représente principalement les différentes consommations des véhicules ainsi que leur émission de CO_2. Cet axe peut être réduit en une seule caractéristique : « *cons_eco2* ».
- L'axe 2 met en opposition la puissance de motorisation des véhicules *lpuiss_max* et le niveau d'émission des polluants de l'air *hcnox*. Il sera à son tour réduit en « *pm_hcnox* » comme unique caractéristique.



Un tracé des données en fonction des deux composantes principales. Les voitures regroupées sur ce tracé sont très similaires les unes des autres en fonction des composantes utilisées pour les décrire.



Après réalisation de l'algorithme PCA, les nouvelles valeurs du VIF s'améliorent pour les nouvelles caractéristiques créées.

Régularisation du modèle PCA

```
summary(lm.lassof2)$coefficients
```

```
                  Value Std. Error   Z score Pr(>|Z|)
(Intercept) 12.3901341 0.03726116 332.52142        0
cons_eco2    2.3391163 0.01307866 178.84979        0
pm_hcnox     0.6723171 0.02294259  29.30433        0
```

Le nouveau résultat final est le suivant :

puiss_admin = 12.39 + 2.34*cons_eco2 + 0.67*pm_hcnox

Ce modèle est beaucoup plus inclusive que le précédent car tient en compte des différents types de consommation, d'émission de gaz carbonique et des résultats d'essais de particules polluantes essence (HC) et gazoil (Nox). Nous pouvons également dire, ceteris paribus, qu'en l'abscence de caractéristiques (à niveau nul), la puissance fiscale reste à son niveau moyen (soit 12,39).

II.5. Entraînement du modèle

Dans cette section, nous allons procéder à un apprentissage automatique avec le package ***caret*** qui est un acronyme qui signifie « Classification and Regression Training » (soit entraînement pour la classification et la régression). Cependant, il est capable de faire beaucoup plus (Bootstrap, Cross Validation, Réseaux neurones etc.).

Fractionnement des données

Suite à la présence de données factorielles, nous allons procéder à un échantillonage aléatoire stratifié. Mais aussi en raison du déséquilibre entre les diffrérentes proportions du facteur d'energie. Cette prochaine étape va permettre de conserver les proportions de la caractéristique « ***energ*** » dans chacun des partages stratifiés. On indique à la fonction **`createDataPartition`** que nous voulons que ce fractionnement ne soit exécuté qu'une seule fois (possibilité de l'exécuter en plusieurs fois).

```
set.seed(123)
partition_indexes <- createDataPartition(projet.df2$energ, times = 1,
                        p = 0.7, list = F)
voit.train<-projet.df2[partition_indexes,] # 70 % pour l'entraînement
voit.test<-projet.df2[-partition_indexes,] # 30 % pour le test
```

Entraînement du modèle

Contrairement à la méthode de l'algorithme PCA pour la création des variables indexes regroupant plusieurs autres variables afin de réduire le problème de multicolinéarité entre les différentes caractérisques, nous pouvons tout simplement utiliser ici, la méthode PLSR (*Partiel Least Squares Regressions*) connue pour son éfficacité de traitement de la multicolinéarité entre les variables explicatives qui biaise le résultat en maximisant la (les) variance(s) expliquée(s).

Dans cette partie, nous allons procéder à l'entrainement de notre modèle via le package ***caret*** car il permet de réaliser beaucoup de méthodes (Cross Validation (CV), Bootstrapp, Réseaux de neurones(nnet), les forêts alétoires (rf) etc).

PLSR avec *Caret*

```
Partial Least Squares

14617 samples
    5 predictor

Pre-processing: centered (8), scaled (8)
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 13155, 13155, 13156, 13156, 13154, 13156,
Resampling results across tuning parameters:

  ncomp  RMSE      Rsquared   MAE
  1      5.360543  0.6995475  3.868179
  2      3.101271  0.8994679  2.087912
  3      2.976887  0.9073936  2.056198

RMSE was used to select the optimal model using the smallest value.
The final value used for the model was ncomp = 3.
```

On constate que le RMSE (Root Mean Square Error) est le plus bas pour **ncomp = 3 (soit 2.976887)**. Il s'agit d'une valeur utile à connaître car elle nous donne aussi une idée de la distance moyenne entre les valeurs de données observées et les valeurs de données prédites. Et plus il est faible, mieux un modèle donné est capable de s'adapter (correspond) à un ensemble de données.

Ceci est à l'opposé du **R^2 (0.9073936)** du modèle, qui nous indique la proportion de la variance de la variable de réponse qui peut être expliquée par la ou les variables prédicteurs du modèle.

Nous pouvons également les calcaluer de la manière suivante :

```
# RMSE = 2.983899
sqrt(mean((test.target - predictions)^2))


# R² = 0.9050699
cor(test.target, predictions)^2
```

Cependant nous notons une légère différence par rapport à l'agorithme qui affiche un résultat optimisé.

Place à la matrice de confusion qui , en apprentissage automatique supervisé, est une matrice qui mesure la qualité d'un système de classification. Chaque ligne correspond à une classe réelle et chaque colonne à une classe estimée.

Matrice de Confusion avec *caret* ou *confusionTableR*

```
Confusion Matrix and Statistics

          Reference
Prediction  HYB   EL   ES  ATE   GO
       HYB  299    0    0    0    1  (299+1)/6261 = Detection Prevalence
       EL     0   16    0    0    0   299/6261 = Detection Rate
       ES     2    0 1989    5   35   16/6261 = Prevalence
       ATE    0    0    2   52    0
       GO     5    0   89    0 3766 → Diagonale des valeurs positives (1)

Overall Statistics

               Accuracy : 0.9778   → (6122/6261) Précision du modèle
                 95% CI : (0.9738, 0.9813)
    No Information Rate : 0.6073
    P-Value [Acc > NIR] : < 2.2e-16 → Modèle très significatif

                  Kappa : 0.9569   → Il y a une grande différence entre
                                     la précision et le taux d'erreur nul.

 Mcnemar's Test P-Value : NA → la variable catégorielle est non binaire.

Statistics by Class:

                     Class:       HYB        EL        ES       ATE        GO
Sensitivity    (TP/actual yes)  0.97712  1.000000    0.9563  0.912281    0.9905
Specificity    (TN/actual no)   0.99983  1.000000    0.9900  0.999678    0.9618
Pos Pred Value (FP/actual no)   0.99667  1.000000    0.9793  0.962963    0.9756
Neg Pred Value (FN/actual yes)  0.99883  1.000000    0.9785  0.999194    0.9850
Prevalence  (actual yes/total)  0.04887  0.002556    0.3322  0.009104    0.6073
Detection Rate                  0.04776  0.002556    0.3177  0.008305    0.6015
Detection Prevalence            0.04792  0.002556    0.3244  0.008625    0.6165
Balanced Accuracy               0.98848  1.000000    0.9731  0.955979    0.9762
```

Conclusion

En somme, nous retenons que l'aboutissement à un bon modèle se trouve au préalable dans la construction et la gestion des données via *caret* ou *missMDA*. Cette étape nous permet d'avoir, notamment avec un nuage de point ou une matrice V de cramer, une idée sur le choix des variables visant à expliquer le phénomène d'étude.

En effet, la méthodologie de travail utilisée dans ce projet, visait au-delà d'une simple analyse économétrique. L'objectif était de réaliser des relations de causes à effets afin de confirmer ou infirmer certaines littératures qui le plus souvent des cas, sont basées sur des cas factuels et non sur la recherche de causalités via des méthodes qui pourraient exposer des phénomènes non apparents. Mais aussi d'approfondir les Cours et TD par des méthodes d'apprentissage automatique (*Machine Learning*).

Le niveau de technologie de nos jours, est tel que les consommateurs deviennent de plus en plus exigeants et ont une préférence pour les véhicules de fortes motorisation qui ne sont pas sans conséquences pour l'écologie. Donc des coûts supplémentaires (par exemple, des coûts liés au type d'énergie consommée ou au résultat d'essai des particules polluantes) pour cette préférence pourraient être envisagées par les politiques. Cela aura pour conséquence, non seulement de responsabiliser les gens vis-à-vis des enjeux climatiques mais aussi de compenser les générations futures pour les défis qu'ils auront à hériter des aïeux.

Bibliographie

Pr. Daniel CHRISTOPHE. « Cours et TD des Méthodes d'Evaluation Appliquées au Marché du Travail », *Doyen de la Faculté d'Economie et de Gestion de l'Université d'Angers*, Master II - IEE (2020- 2021).

Scott V. BURGER. Livre : « Machine learning avec R», *Pour une modélisation mathématique rigoureuse*, O'REILLY.

https://github.com/

https://rdrr.io/

Annexes

CODE R

```
#PATH & INSTALLATION DES PACKAGES EXTERNES

list.files("c:/Rtools40", "make\\.exe", recursive = TRUE, full.names = TRUE)
writeLines('PATH="${RTOOLS40_HOME}/usr/bin;${PATH}"', con = "~/.Renviron")
Sys.which("make")
install.packages("dplyr")
library(devtools)
install_github("hadley/dplyr")
install_github('skardhamar/rga')
require(devtools)
install_github("dustinfife/fifer")
Install.packages("remotes")
remotes::install_github("dustinfife/flexplot")
remotes::install_github("dustinfife/fifer")

# libraries
library(fifer)
library(Rcmdr)
library(FactoMineR)
library(Factoshiny)
library(PerformanceAnalytics)
library(sampleSelection)
library(stargazer)
library(tidyverse)
library(magrittr)
library(corrplot)
library(caTools)
library(censReg)
library(broom)
library(dplyr)
```

```
library(nnet)
library(car)

setwd("C:/Users/HP/Desktop/M2_EA-IEE/Semestre 2/Evaluation_MT")
fic_etiq_edition_40.mars.2015   <-   read.csv("fic_etiq_edition_40-mars-2015.csv",
sep=";", na.strings = "NA")
data<-(fic_etiq_edition_40.mars.2015)
View(data)
fix(data)
names(data)
str(data)
summary(data)

# Traitement de la base de données.

data1<-data[,-c(1:7,9,12:13,23:26)]
data1$energ<-factor(data1$energ)
levels(data1$energ)<-c("HYB","HYB","EL","ES","ATE","ATE","ATE","HYB","HYB","ATE",
                       "ATE","GO","ATE")

data1$hyb<-NA
data1$hyb[which(data1$energ=="HYB")]<-1
data1$hyb[which(data1$energ!="HYB")]<-0
data1$hyb<-as.factor(data1$hyb)

data1$ess<-NA
data1$ess[which(data1$energ=="ES")]<-1
data1$ess[which(data1$energ!="ES")]<-0
data1$ess<-as.factor(data1$ess)

data1$gazoil<-NA
data1$gazoil[which(data1$energ=="GO")]<-1
data1$gazoil[which(data1$energ!="GO")]<-0
data1$gazoil<-as.factor(data1$gazoil)

data1$elec<-NA
```

```
data1$elec[which(data1$energ=="EL")]<-1
data1$elec[which(data1$energ!="EL")]<-0
data1$elec<-as.factor(data1$elec)

data1$AT_energ<-NA
data1$AT_energ[which(data1$energ=="ATE")]<-1
data1$AT_energ[which(data1$energ!="ATE")]<-0
data1$AT_energ<-as.factor(data1$AT_energ)

marque <-paste(data$lib_mrq_doss, data$lib_mod_doss, sep =" ")

base <- cbind.data.frame(marque, data1)
base$lpuiss_max <-log(base$puiss_max)
base$lco2 <-log(base$co2_mixte)
base1 <- base[,-c(2,4,8,10:11)]
dim(base1)
str(base1)

base2 <-na.omit(base1) # il ne reste que 11729 observations sur 20880 --> pas
pertinent.

# GESTION DES DONNES MANQUANTES
## Via (missMDA)

library(missMDA)
nb<-estim_ncpPCA(base1[,-c(1,9:13)], scale=TRUE)  # détermine la valeur ncp
newdata <- imputePCA(base1[,-c(1,9:13)], ncp=2 , scale=TRUE)   # changer ncp=?
selon le résultat précédent
newdata <- as.data.frame(newdata$completeObs)
data.df<-cbind.data.frame(base1[,c(1,9:13)], newdata)
str(data.df)
names(data.df)
summary(data.df)

## Via (preProcess)
```

```
library(caret)
dummy <- dummyVars(~., data = base1[,-c(1,9:13)])
dummy_predict <- predict(dummy, base1[,-c(1,9:13)])
head(dummy_predict)
pre_process <- preProcess(dummy_predict, method = "bagImpute")
imputed.base <-predict(pre_process, dummy_predict)
head(imputed.base)
energ <-base[,2]
base.df<-cbind.data.frame(base1[,c(1,9:13)], energ, imputed.base)

summary(base.df)

write.csv(base.df,file="Projet.csv", row.names = F)

#matrice v de cramer

cv <- function(x, y) {
  t <- table(x, y)
  chi <- suppressWarnings(chisq.test(t))$statistic
  cramer <- sqrt(chi / (NROW(x) * (min(dim(t)) - 1)))
  cramer
}

cramer.matrix<-function(y, fill = TRUE){
  col.y<-ncol(y)
  V<-matrix(ncol=col.y,nrow=col.y)
  for(i in 1:(col.y - 1)){
    for(j in (i + 1):col.y){
      V[i,j]<-cv(y[,i],y[,j])
    }
  }
  diag(V) <- 1
  if (fill) {
    for (i in 1:ncol(V)) {
      V[, i] <- V[i, ] }
```

```
}
  colnames(V)<-names(y)
  rownames(V)<-names(y)
  V
}

corrplot(cramer.matrix(base.df[,-c(1,7)]),type="lower",order="hclust",
tl.col="black")

# Régression multivariée

projet <-base.df[,-c(1,7)]
lm.all<-lm(puiss_admin ~., data = projet)
summary(lm.all)

lm.all2<-lm(puiss_admin ~., data = projet[,-5])
summary(lm.all2)

vif(lm.all2)%>% barplot(main= "VIF Values", horiz = F, col = "steelblue")
abline(h = 5, lwd = 3, lty = 2)

# Ce qui ressort du résultats du VIF est inquiétant car les valeurs de certaines
# variables dépassent largement 5. Du coup, les estimations des coefficients
# ainsi que la p-value de notre modèle de régression sont probablement peu
faibles.

# Régularisation
## Cette technique consiste à rééchelonner les coefficients en fonction de
## leur impact réel sur le modèle de régression.

library(lasso2)
lm.lasso <-l1ce(puiss_admin ~., data = projet[,-5])
summary(lm.lasso)$coefficients

# la valeur du coefficient de "conso_exurb" est de 0. Du coup on le retire
# de la régression multiple.
```

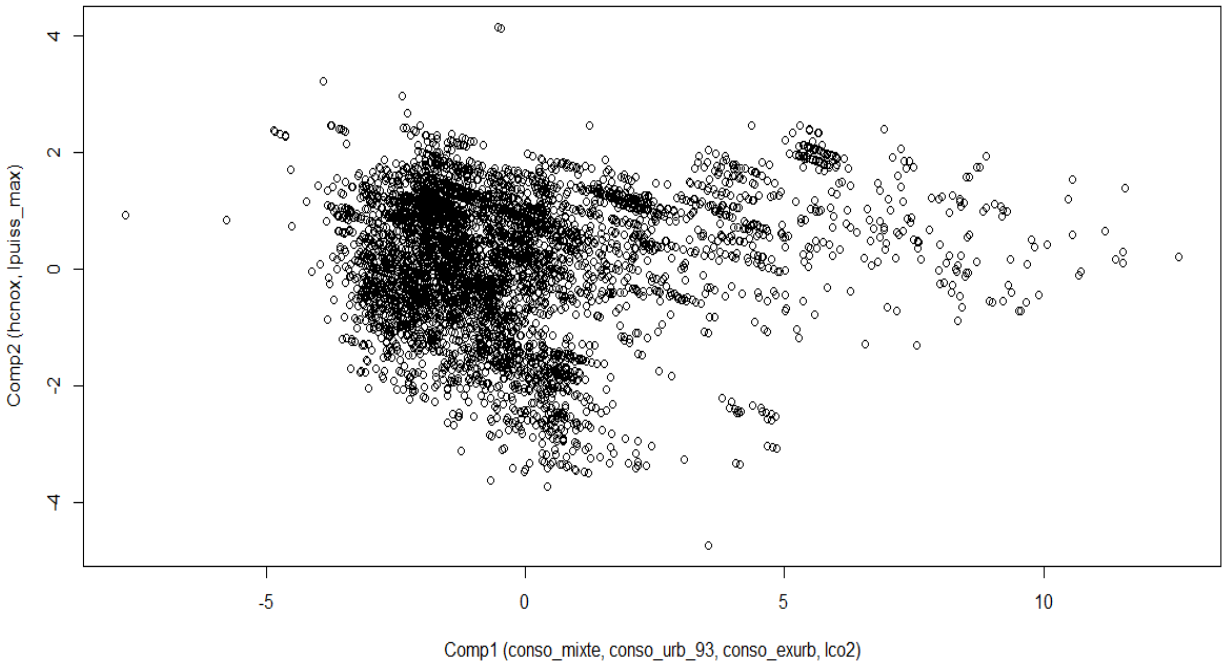
```
lm.lasso2<-l1ce(puiss_admin ~.-conso_exurb, data = projet[,-5])
summary(lm.lasso2)$coefficients

# le coefficient "ess" n'est pas significatif car p-vavue =0.99

lm.lasso3<-l1ce(puiss_admin ~.-conso_exurb-ess, data = projet[,-5])
summary(lm.lasso3)$coefficients

lm.lasso4<-l1ce(puiss_admin ~.-conso_exurb-ess-hyb-gazoil-hcnox, data = projet[,-
5])
summary(lm.lasso4)$coefficients

lm.lasso5<-l1ce(puiss_admin ~.-conso_exurb-conso_mixte-ess-hyb-gazoil
                -elec-hcnox-co_typ_1-ptcl-lco2, data = projet[,-5])
summary(lm.lasso5)$coefficients

# Le résultat final n'a que 2 caractéristiques au lieu de 12:
# puiss_admin = -34.47 + 9.12*lpuiss_max + 0.31*conso_urb_93
# La caractéristique la plus importante est la puissance maximale suivi de la
# consommation urbaine
summary(lm.lasso5)

# PCA

pairs(projet[,-c(1:5)], lower.panel = NULL)  # ou chart.Correlation()
chart.Correlation(projet[,-c(1:5)], histogram=TRUE, pch=19)
pca <-princomp(projet[,-c(1:5)], scores= T, cor =T)
plot(pca)
summary(pca) # on ne garde que les 4 premières composantes

pca$loadings[,1:4]

scores.df <-data.frame(pca$scores)
scores.df$car <-base.df$marque
```

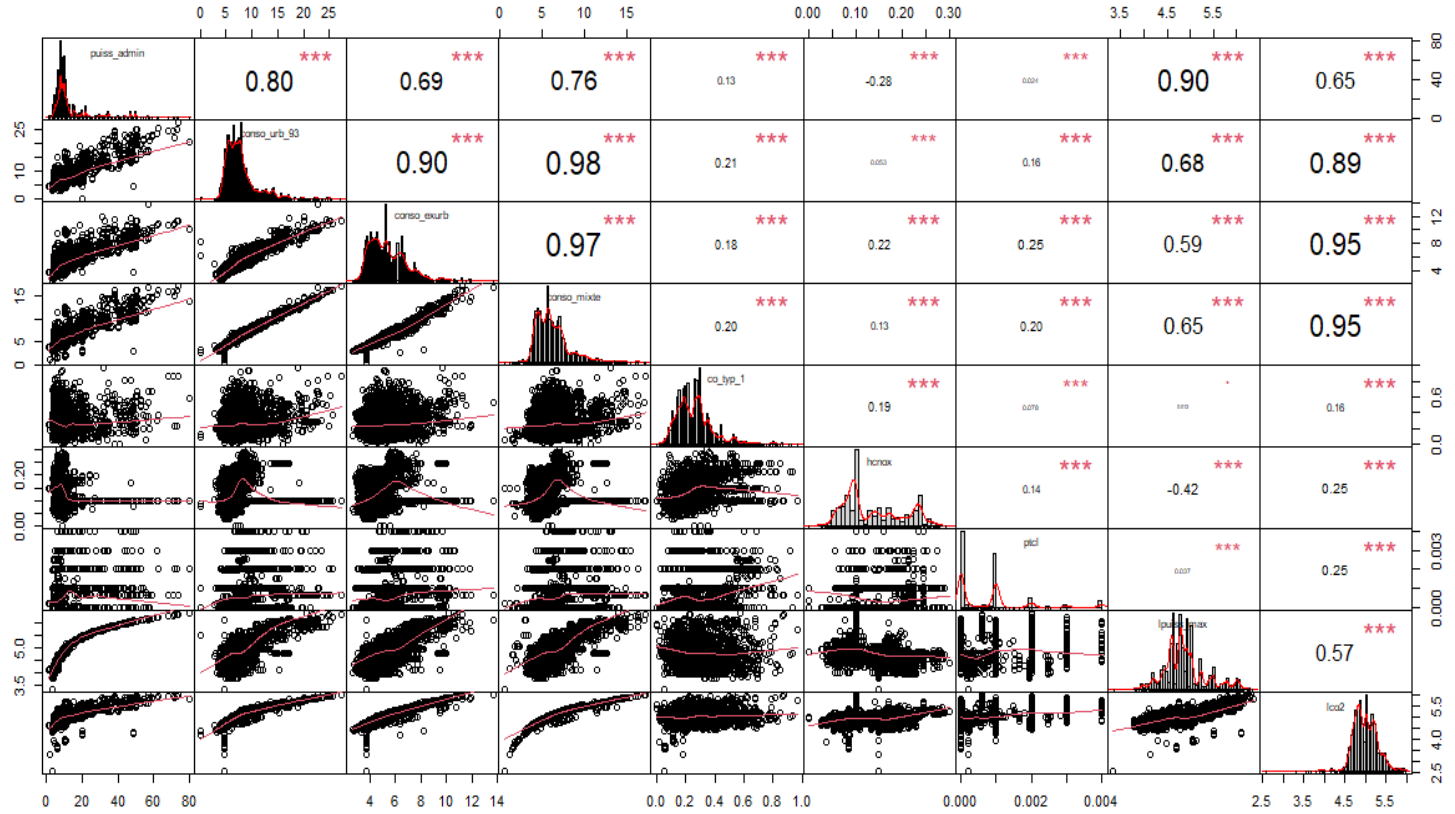
```
plot(x = scores.df$Comp.1, y =scores.df$Comp.2,
     xlab = "Comp1 (conso_mixte, conso_urb_93, conso_exurb, lco2)",
     ylab = "Comp2 (hcnox, lpuiss_max)")
text(scores.df$Comp.1, scores.df$Comp.2, labels = scores.df$car,
     cex = 0.7, pos = 3)

plot(x = scores.df$Comp.3, y =scores.df$Comp.4,
     xlab = "Comp3 (co_typ_1, ptcl)",
     ylab = "Comp4 (ptcl, co_typ_1)")
text(scores.df$Comp.3, scores.df$Comp.4, labels = scores.df$car,
     cex = 0.7, pos = 3)

scores.df2<-scores.df[,c(1:2,10)]
names(scores.df2) <- c("cons_eco2","pm_hcnox","car")

projet.df<-cbind.data.frame(base.df,scores.df2[,-3])
projet.df2<-projet.df[,c(1:8,12,14,17,18)]
model <-lm(puiss_admin~., data = projet.df2[,-c(1,6,7)])
summary(model)
vif(model) %>% barplot(main= "VIF Values after PCA", horiz = F, col = "steelblue")
abline(h = 5, lwd = 3, lty = 2)

# Régularisation2

lm.lassof <-l1ce(puiss_admin ~., data = projet.df2[,-c(1,6,7)])
summary(lm.lassof)$coefficients

lm.lassof2 <-l1ce(puiss_admin ~.-co_typ_1-ptcl, data = projet.df2[,-c(1:7)])
summary(lm.lassof2)$coefficients

# puiss_admin = 12.39 + 2.34*cons_eco2 + 0.67*pm_hcnox

# Echantillonage et Entraînement des modèles dans R
# Suite à la présence de variables factorielles, il serait judicieux de procéder
# à un échantillonage aléatoire stratifié.
```

```
#library(fifer)
#strate.df <-stratified(projet.df2[,-c(1:6)],c("ptcl","cons_eco2"), 0.7)
#summary(strate.df)

str(projet.df2)
train <- subset (projet.df2, select = -c(1:6))
train.df <-na.omit(train)

library(caret)
set.seed(1)
partition_indexes<-createDataPartition(train.df$puiss_admin, times = 1,
                                       p =0.7, list = F)

voit.train<-train.df[partition_indexes,]
voit.test<-train.df[-partition_indexes,]

# Entraînement du modèle

library(pls)
model<-plsr(puiss_admin ~., data = voit.train)
summary(model)

model1<-train(puiss_admin ~., data = voit.train, method = 'pls')
model1
plot(model1)

set.seed(1)
model2<-train(puiss_admin ~., data = voit.train, method = 'pls',
                preProcess = c("center", "scale"));model2

test.features = subset(voit.test, select=-c(puiss_admin))
test.target = subset(voit.test, select=puiss_admin)[,1]

predictions = predict(caret.pls, newdata = test.features)
```

```
# RMSE = 2.983899
sqrt(mean((test.target - predictions)^2))

# R² = 0.9050699
cor(test.target, predictions)^2

# Cross Validation
set.seed(3)
train.control <-trainControl(method = "cv", number = 10,
                             #repeats = 3,
                             #sampling = "up",
                             #search = "grid"
                             )
library(gbm)
tune.grid <-expand.grid(ncomp = seq(1, 10, by = 1))
                        #eta = c(0.05, 0.075, 0.1),
                        #nrounds = c(50, 75, 100),
                        #max_depth = 6:8,
                        #min_child_weight = c(2.0, 2.25, 2.5),
                        #colsample_bytree = c(0.3, 0.4, 0.5),
                        #gamma = 0,
                        #subsample = 1
                        #n.trees = seq(10, 1000, by = 100),
                        #interaction.depth = c(4),
                        #shrinkage = c(0.01, 0.1),
                        #n.minobsinnode = c(5, 10, 20, 30)
                        )

head(tune.grid)

install.packages("doSNOW")
library(doSNOW)

cl <-makeCluster(3, type = "SOCK")
registerDoSNOW(cl)
```

```
set.seed(1)
caret.cv <- train(puiss_admin ~., data = voit.train, method = "xgbTree",
                  preProcess = c("center", "scale"),
                  #tuneGrid = tune.grid,
                  trControl = train.control
                  )
stopCluster(cl);caret.pls

test.features = subset(voit.test, select=-c(puiss_admin))
test.target = subset(voit.test, select=puiss_admin)

#confusionMatrix(predict(caret.cv, voit.test), voit.test$energ)
#rf_class <-predict(caret.cv, newdata = voit.test, type = "raw")
#predictions <-cbind.data.frame(train_preds=rf_class, voit.test$energ)

library(ConfusionTableR)
mc_df <- ConfusionTableR::multi_class_cm(predictions$train_preds,
predictions$`voit.test$energ`)
print(mc_df$record_level_cm)
tibble::glimpse(mc_df$record_level_cm)
print(mc_df$confusion_matrix)
print(mc_df$cm_tbl)

set.seed(1)
caret.rf <- train(energ ~.,
                  data = voit.train,
                  method = "rf",
                  tuneGrid = tune.grid,
                  trControl = train.control)
stopCluster(cl);caret.rf
predictions <-predict(caret.rf, voit.test)
cm <-confusionMatrix(predictions, voit.test$energ)
```


Comp2 (hcnox, lpuiss_max)
Comp1 (conso_mixte, conso_urb_93, conso_exurb, lco2)


puiss_admin
conso_urb_93
conso_exurb
conso_mixte
co_typ_1
hcnox
ptcl
lpuiss_lmax
lco2