Etude de cas : Challenge TOTAL

Challenge Mines-Telecom

Prédiction de la performance des puits

Groupe : 6
Réalisateurs : Mohammed Taha ABDELMOULA
Anass AKRIM
Imane HAOUDI
François RULLIERE

Date : 1 décembre 2017

Table des matières

1 Introduction

Le travail d'un Data Analysis est d'extraire des informations utiles pour les décideurs à partir des données collectées pour prévoir des productions, des potentiels risques, trouver des anomalies, analyser la perfomance des scénarios possibles ou pour classer des individus dans des classes.
Dans cet étude de cas, nous allons aborder l'un des fréquents problèmatiques dans le domaine industriel : prévoir un intervalle de prédiction pour des productions. Il s'agit de la production pétrolières qui est classiquement abordé avecune approche ne permettant pas de prendre en compte les spécifités de chaque puits.

1.1 Contexte du challenge

Le challenge TOTAL a été intégré dans l'enseignement de la majeur Science des données comme un outil d'évaluation des connaissances acquis tous au long du semestre. Les élèves ont été donc amenés a travailler dessus tout au long du semestre pour renforcer leurs connaissances et leur maîtrise du domaine.

Le challenge porte sur la prédiction de deux indicateurs de performance : les quantités de gaz (GAS360) et de condensat (CUM360) extraits d'un puits, un an après sa mise en service, c'est-à-dire une fois que le système de production est stabilisé, ceci en se basant sur des données collectées caractérisant la physique de chaque puits.

1.2 Nature des données

L'entreprise TOTAL a fournie deux table de données : données pour l'apprentissage et données pour tester la validité du modèle sur le site (un score est donné à chaque contribution de résultats). On cherche à prédire deux réponses : GASCUM360 et OILCUM360 en fonction de 44 prédicteurs qui présentent les caractéristiques de chaque puits. Un analyse plus approfondie des données sera faite dans la partie analyse exploratoire

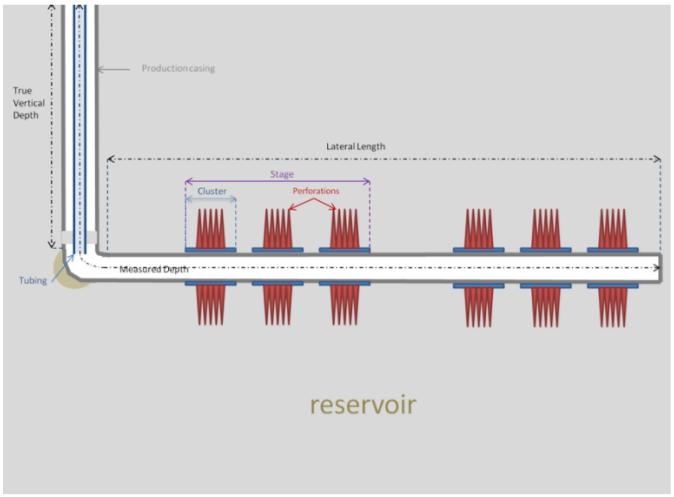


FIGURE 1 – caractéristiques d'un puits

1.3 Analyse exploratoire

L'analyse exploratoire permet de jeter un premier regard métier sur nos données pour ensuite trouver des pistes sur l'exploitation de ces dernières. Nous allons essayer de comprendre comment se comportent nos individus, de juger les prédicteurs et de juger les réponses proposés et enfin de proposer des axes de travail pour le traitement des données.

1.3.1 Visualisation

1.3.1.1 En chiffres Nous avons deux jeux de données : les données à évaluer pour le challenge et les données d'entraînement du modèle. Elles sont décrites par 45 prédicteurs, les données d'entraînement nous fournies deux réponses supplémentaires à prédire. Les données d'entraînement comportent 460 individus, celles de test 235.

Parmi ces individus, certaines valeurs sont manquantes, généralement sur les mêmes individus et sur les mêmes variables (voir paragraphe suivant). Les données d'entraînement ont 8,4% de valeurs manquantes et les données de tests 7,4%

Les données sont soit des dates pour les prédicteurs *Date_ Drilling, Date_ Completion, Date_ Production*, soit des valeurs numériques pour les autres prédicteurs. On remarque aussi que la première variable *API* renseigne le numéro du puit et n'est donc pas intéressante pour l'étude à proprement dite de notre cas.

Toutes les données ont été préalablement centrées et réduites. Ainsi, hormis les prédicteurs *API, Date_ Drilling, Date_ Completion, Date_ Production*, les valeurs se situes presque toujours entre -3 et 3. On remarques quelques exceptions prononcées avec une plus grande valeur à 23,8 et une plus petite à -8,8.

Il y a t-il des variables qualitatives ? Ceci est une information que nous ne pouvons pas récupérer simplement. Toutefois une études plus approfondie dans la partie 2.3 nous montre que l'on peut considérer sur ces données initiales 9 variables qualitatives.

1.3.1.2 Les données manquantes On remarque plus de 40% des variables possèdent 20% de données manquantes, regardons cette répartition afin d'avoir une idées de l'amputation de certaines variables. S'il manque trop de valeurs il peut être intéressant de ne pas prendre en compte un prédicteur.

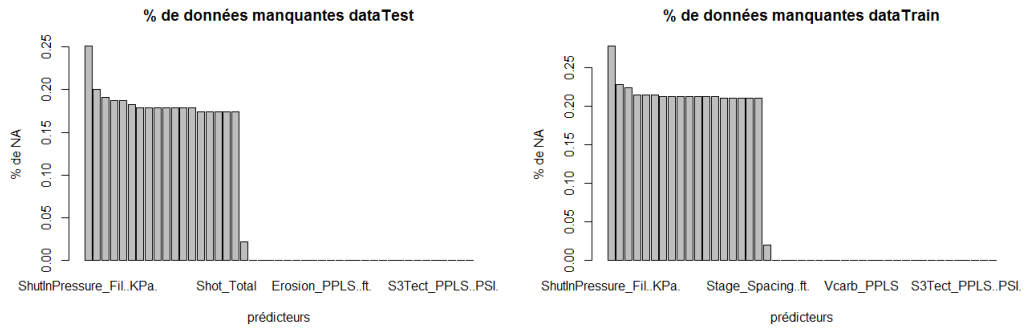


FIGURE 2 – Pourcentage de valeurs manquantes pour les données d'entraînement et de test par prédicteur

Sur nos deux ensembles de données, la proportion de valeurs manquantes par prédicteur

est très proche (soit nul, soit de l'ordre de 20%). La variable la plus amputée est *ShutInPressure_ Fil..KPa.* dans les deux jeux de données.

1.3.1.3 Nos individus

Ici nous allons représenter graphiquement la valeur des prédicteurs de chaque individus les uns par rapport aux autres. On ne visualise uniquement les données d'entraînement puisque dans un contexte de data scientist, nous n'aurons pas toujours les données tests pour visualiser plus d'individus. Nous pouvons aussi justifier ce choix en disant que le nombre d'individu est suffisamment représentatif.

Ayant beaucoup de prédicteurs, nous n'allons mettre en valeur ici uniquement des représentations en deux dimensions de paires de variables qui nous semblent intéressantes pour l'étude. Pour cela nous allons regrouper les prédicteurs en familles, c'est-à-dire des groupes de prédicteurs se rapportant au même domaine. Cette classification permettra de mettre exergue certaines relations physiques et de raisonner sur des ensembles de données compréhensibles. Bien sur cela ne veut pas dire que nous devons uniquement regarder les prédicteurs uniquement au sein de leur famille. Nous aborderons ce sujet de façon plus mathématique dans l'étude des transformations de prédicteurs (cf. 2.6).

1.3.2 Les prédicteurs

1.3.2.1 Familles de prédicteurs

Nous avons déterminé les familles suivantes :

- Les positions : *Surf_ X, Surf_ Y, Zone*
- Les dates : *Date_ Drilling, Date_ Completion, Date_ Production*
- Géométrie et environnement physique :
  - Valeurs mesurées dans le réservoir : *Erosion_ PPLS..ft. Pressure_ PPLS..PSI PR_ PPLS YM_ PPLS..PSI. RHOB_ PPLS..g.cc. Res_ PPLS..Ohmm. GR_ PPLS..API. DT_ PPLS..us.ft. DTs_ PPLS..us.ft. Temperature..F. Temp_ Anomaly..F. S3Tect_ PPLS..PSI. S3_ contrast_ PPLS..PSI. Heat_ Flow..W.m2.*
  - Valeurs physiques dans le réservoir : *Lateral_ Length..ft. Depth_ TVD_ PPLS..ft. Nbr_ Stages*
  - Volumes introduis : *TOC_ PPLS.... Vcarb_ PPLS Vsand_ PPLS Vclay_ PPLS*
- Données d'exploitations :
  - Données post Shut-in : *ShutInPressure_ Fil..KPa. ShutInPressure_ Initial..KPa. ISIP..KPa.*
  - Plage de ratio de pompage de fluide : *Avg_ Rate_ Slurry..bpm. Max_ Rate_ Slurry..bpm. Min_ Rate_ Slurry..bpm.*
  - Plage de pression d'injection : *Avg_ Treating_ Pressure..KPa. Max_ Treating_ pressure..KPa. Min_ Treating_ Pressure..KPa.*
  - Les Shots : *Shot_ Density..shots.ft. Shot_ Total*
  - L'utilisation de proppant : *Proppant_ Designed..kg. Proppant_ in_ Formation..kg.*
  - Autres : *Frac_ Gradient..PSI.ft. Avg_ Breakdown_ Pressure..KPa.*
- Données de production : *OilCum360 GasCUm360*

1.3.2.2 Analyse des familles de prédicteurs

Nous pouvons regarder plus précisément certaines combinaisons de prédicteurs en fonction de nos données d'entraînement. Il est alors plus facile de voir des corrélations ou des anomalies. C'est le cas avec la famille des *Autres données d'exploitation* présentée en figure 3.

On remarque que se débarrasser de la valeur extrême améliore la lisibilité et la prédiction sur ces données, il en est donc d'autant plus probable que cette donnée soit une erreur. Ainsi un travail nécessaire est de se débarrasser de outliers comme ici sur le prédicteur *Frac_ Gradient..PSI.ft.*

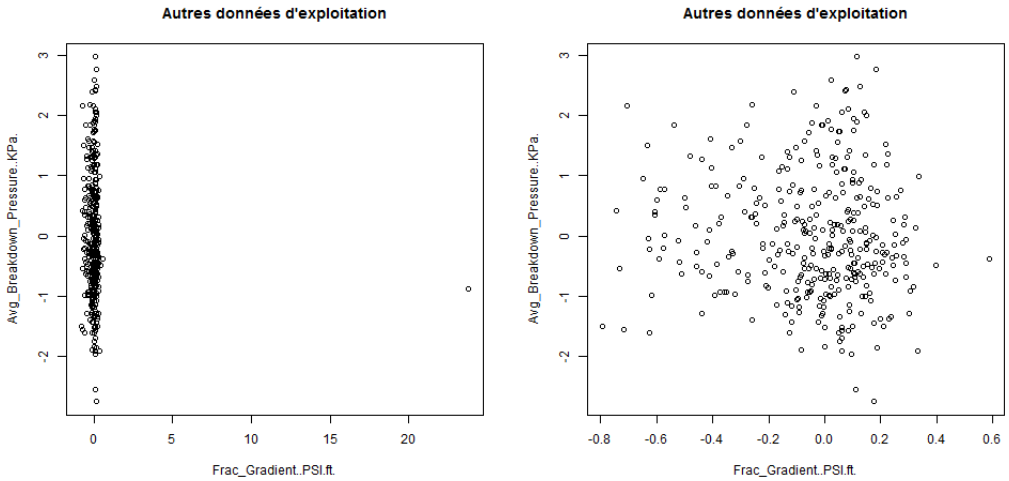


FIGURE 3 – Famille A *Autres données d'exploitation*, tirées des données brutes (à gauche), et avec la valeur extrême en moins (à droite).

qui polluent les données afin de pouvoir mieux interpréter les données. Ici nous n'auront pas pu nous représenter correctement ce graphique avant d'avoir épurer les données.

1.3.2.3 Bilan du groupement des prédicteurs La représentation des familles de prédicteur est très parlante parce qu'elle se rapproche des notions physiques. Pour ne pas à avoir à afficher les graphiques deux fois, nous vous redirigeons directement vers les parties 2.6 et 2.7, où une étude plus en profondeur sera faite.

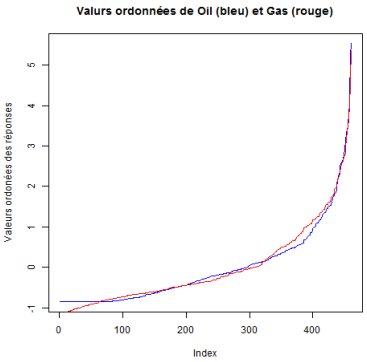
Globalement, l'idée est de supprimer des variables trop corrélées avec d'autres afin d'éviter les données redondantes. Nous pouvons aussi faire du clustering sur certains prédicteurs (comme $Surf\_X$ *et* $Surf\_Y$) pour avoir une donnée plus pertinente.

1.3.3 Forme des Réponses

Regardons maintenant ce que nous avons en sortie. Sur la figure 4 nous avons tracé dans un premier temps les points de GasCum360 contre OilCum360.

Nous pouvons voir que :

- Les valeurs de Gas et Oil sont souvent comprises entre -1 et 2 avec quelques valeurs allant jusqu'à 5.
- Les valeurs de Gas ont un seuil proche de -1
- Les valeurs de Oil ont une droite seuil passant par aux alentours de $[0, 1.7]$ et $[7, 0]$
- Les valeurs de Gas et Oil sont très concentrées vers les valeurs basses
- Il n'y a pas de valeurs avec des valeurs de Gas et Oil grandes. Nous pouvons tracer le carré $[1, 5]^2$ sur la figure 4 qui ne contient que 3 valeurs (sur 460).



Au vu de la répartition de la figure 4 qui montre une accumulation de valeur, il est intéressant de regarder la répartition des valeurs de Oil

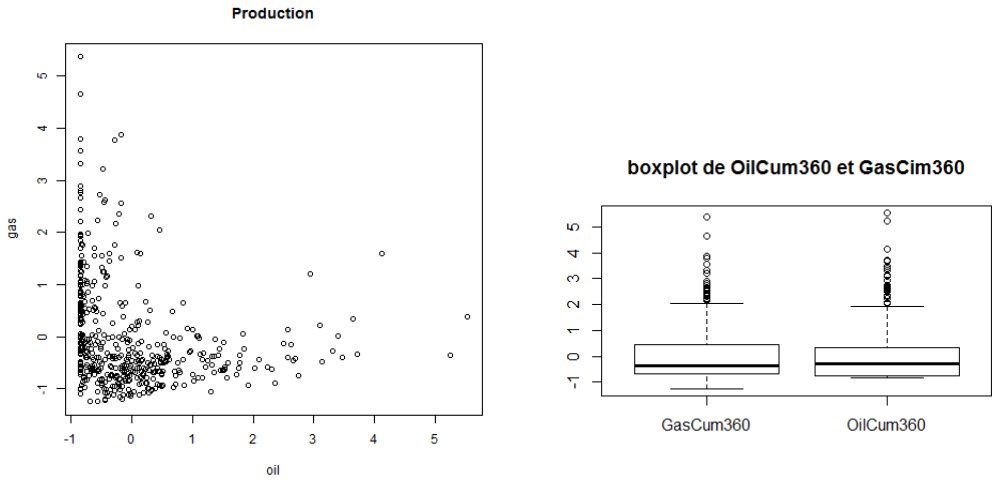


FIGURE 4 – Réponse OilCum360 et GasCum360 des données d'entraînement.

et Gas. La figure **??** ci-contre montre cette répartition. On note une accumulation qui semble être de la forme exponentielle, ou polynomiale.
Il sera intéressant de transformer ces réponses avec la fonction qui l'interpolera le mieux. Avec cette dernière nous aurons des réponses transformées mieux réparties et donc plus facile à prévoir.

1.4 Organisation de l'étude de cas

1.4.1 Les enseignements dans le challenge

Plusieurs notions et méthodes de prédictions ont été abordés au cours de la majeur. Le but de cet étude de cas étant l'approfondissement des notions vues en cours et la découvert d'autres méthodes, il s'est avéré nécessaire de'abordant l'ensemble de ces notiosn en partant des méthodes de l'optimisation (CMAE-ES principalement), outils de l'analyse exploratoires, méthodes de réduction de dimension, méthode de construction des modèles (régression sous ses plusieurs formes, arbres et forêtes aléatoires, réseaux de neuronnes), méthodes de clustering et de classification, notions de bagging, bootstrap et cross-validation jusqu'aux outils de validation des modèles, constructions des intervalles de prédiction, blending et interprétation.
Nous utiliserons aussi d'autres méthodes et notions non abordés en cours que nous présenterons brièvement.

1.4.2 Répartition du travail

Étant un projet qui est assez lourd, une bonne organisation au sein de l'équipe était indispensable. Chaque membre du groupe s'est intéressé à une partie qui l'intéresse. La bonne communication et entendement au sein de notre groupe était un avantage qui nous a permis de bien avancer le projet.
Concernant la répartition du travail :

— Anass s'est occupé de la partie des forêts aléatoires, traitement des outliers et des autres méthodes de régressions.

— François s'est occupé principalement de la partie prétraitement des données, méthode de bootstrap et de la construction des intervalles de prédictions non symétriques.
— Imane s'est occupée de la construction des modèles de la régression linéaire, des réseaux de neurones, de l'utilisation de xgBoost, implémentation de la Cross-validation et la réalisation d'un bilan des modèles utilisés dans le projet.
— Taha s'est occupé de la partie construction d'un nouveau espace (nouveaux prédicteurs à partir des anciens), forêts aléatoires et du choix des intervalles de prédictions.

2 Prétraitement

2.1 Introduction

Dans cette partie nous allons formater et traiter les données afin qu'elles représentent au mieux les individus et qu'elles soient le moins redondantes possibles. Cette analyse permettra d'améliorer les modèles obtenus dans la partie traitement.

Nous considérons dans un premier temps les données tests et d'entraînement de façon équivalente. Puis dans un second temps, certain modèles devront être enseignés par les données d'entraînement et appliqués sur les données tests.

2.2 Traitements des outliers

Nous avons vu l'exemple d'un cas de outlier plus tôt sur dans figure 3. Intuitivement nous avons enlever le outlier qui rendait la variance du prédicteur excessivement grande. De façon plus générale, nous considérons un ou des individus comme des outliers lorsque les enlever réduit fortement la variance de l'échantillon.

Pour cela nous regardons sur chaque prédicteurs, pour chaque individus se situant dans les 10% des valeurs extrêmes de l'échantillon, si l'enlever change significativement la variance. Si c'est le cas, nous le considérons comme et outlier et le supprimons. Notre critère de considération est le suivant

$$(var(pred[-indiv])/var[pred]) - 1) > 0.01$$

Cela nous a permis d'enlever 156 outliers pour les données d'entraînement. Tous ces outliers ont été remplacés par des *NA*. Pour les données tests, nous avons pris l'ensemble des données d'entraînement, ajouté les individus tests, et relancé le programme. Cela permet de ne pas avoir de conflit d'échelle entre les deux ensemble de données. Nous trouvons donc 56 outliers pour les données tests.

2.3 Traitement des données qualitatives

Lors de l'analyse exploratoire, on remarque que certain prédicteurs sont à valeurs discrètes, mais ce n'est pas évident de les déterminer à l'œil. Afin de déterminer les prédicteurs qualitatifs, nous allons regarder avec la fonction *table* le nombre de valeurs en sorti. Si le nombre est grand, on pourra considérer la variable comme quantitative, sinon elle sera qualitative.

Nous avons tracé la cardinale de l'image de chacun des prédicteurs (cf. figure 5), ainsi nous pouvons donner nos 9 variables qualitatives que l'on peut regrouper par famille :
— Vcarb_PPLS, Vsand_PPLS, Vclay_PPLS
— PR_PPLS, RHOB_PPLS..g.cc., Nbr_Stages
— Zone,
— Shot_Density..shots.ft., Shot_Total

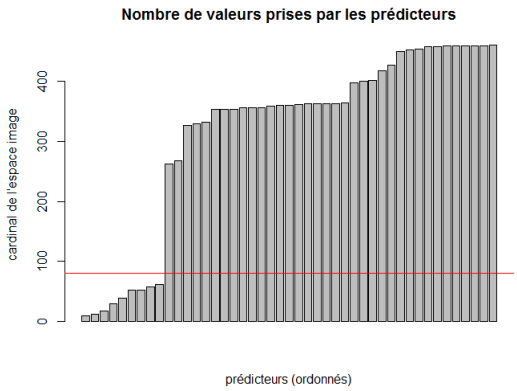


FIGURE 5 – Valeurs manquantes (en blanc) pour les données d'entraînement par individus et par variables

Toutefois par simplicité, afin de pouvoir utiliser *missForest* sans inconvénient, nous allons prendre uniquement les prédicteurs avec moins de 53 valeurs. Nous laissons donc *hot_Density..shots.ft.* et *RHOB_PPLS..g.cc.* en tant variable quantitative.

Nous transformons alors ces variables en objet *factor* pour le traitement ultérieur des données.

2.4 Traitement des données manquantes

Nous avons vu dans l'analyse exploratoire la quantité de données manquante. Regardons de plus près sur les données d'entraînement.

On peut voir sur le graphique 6 que :

— La plupart des individus on des valeurs manquantes sur les même 18 prédicteurs,
— La plupart des valeurs manquantes se font sur ces 18 prédicteurs,
— La variable *ShutInPressure_Fil..KPa.* a le plus de valeurs manquantes, faut-il la supprimer ?

Pour résoudre ce problème, il y a plusieurs solutions. Nous pouvons imputer les données manquantes par la moyennes des données restantes, nous pouvons supprimer les individus à imputer... Nous avons utilisé la fonction *missForest* qui utilise les forêts aléatoires pour imputer les données. Avec les erreurs d'imputations nous pouvons voir si il est intéressant de garder certain prédicteur : si l'erreur est inférieur à un certain seuil de pollution, on retirera le prédicteur en question.

Une fois les variables qualitatives créées, l'utilise de *missForest* est bien plus lente.

Il se trouve que lorsque l'on faisait l'imputation sans enlever les outliers, notre erreur d'imputation se retrouvait relativement grande. Nous devions enlever certain prédicteurs comme *Frac_Gradient..PSI.ft.* qui possédait un très grand outlier.

Finalement notre code (cf. figure 7) nous donne le nombre de variables imputées. Il nous faut supprimer le prédicteur*Avg_Breakdown_Pressure..KPa.*.

2.4.1 Du quantitatif au qualitatif : Clustering

Les prédicteurs *Surf_X* et *Surf_Y* représente des coordonnées de puits dans l'espace. Il est plus pertinent de regarder à quel puits sont proches et quel puits sont éloignés. Deux puits

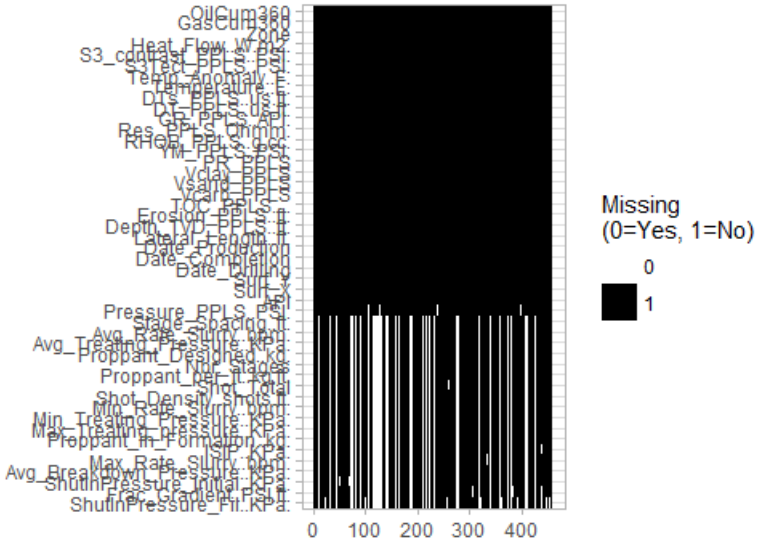


FIGURE 6 – Valeurs manquantes (en blanc) pour les données d'entraînement par individus et par variables

```
> dataTrain<-preTraitement(dataTrain, 'train')
Nombre d'outliers pour  train :  156
% de variance gagnée :  3.75842
7  variables qualitatives
  missForest iteration 1 in progress...done!
  missForest iteration 2 in progress...done!
  missForest iteration 3 in progress...done!
  missForest iteration 4 in progress...done!
/!\ Avg_Breakdown_Pressure..KPa.  erreur d'imputation à  0.8016869
Erreur totale d'imputation de :  2.996721 %
> dataTest <-preTraitement(dataTest, 'test')
Nombre d'outliers pour  test :  56
% de variance gagnée :  1.319822
  missForest iteration 1 in progress...done!
  missForest iteration 2 in progress...done!
  missForest iteration 3 in progress...done!
  missForest iteration 4 in progress...done!
  missForest iteration 5 in progress...done!
  missForest iteration 6 in progress...done!
/!\ Nbr_Stages  erreur d'imputation à  0.7938144
Erreur totale d'imputation de :  3.788582 %
```

FIGURE 7 – Résultat de l'imputation avec *missForest*

proches pourraient utiliser la même poche de Gas et Oil et donc la production de l'un influerait sur l'autre.

Mathématiquement, nous allons regrouper les puits proches dans un même cluster. La méthode de clustering doit ici regrouper les individus proches avec une norme euclidienne, s'ils sont trop éloignés, on les met dans une autre classe. Pour cela nous utilisons la méthode *DBSCAN*, nous obtenons la figure 8.

2.5 Traitement des dates

Les dates sont des données *factor* de la forme $jj/mm/yyyy$. Nous avons donc converti ces données en nombre de jours. Ensuite nous les avons centrées et réduites et appliqué la même transformation (avec la même moyenne et écart type) aux données de tests.

Les trois dates *Date_ Drilling, Date_ Completion, Date_ Production* représentent des dates clefs de l'évolution du puit. Respectivement la date de perforage du puit, sa date de fin de perforage, et sa date de début de production. Il est plus judicieux de raisonner en age plutôt

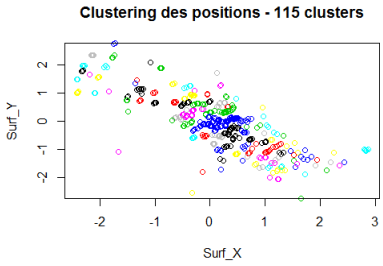


FIGURE 8 – Cluster des positions des puits

qu'en date : il se peut que la durée de forage influe sur la profondeur du puit ou sur sa production.

Ainsi nous créons *Age_ Production_ Completion, Age_ Completion_ Drilling* mais nous laissons la date *Date_ Drilling*. En effet si nous avions pris trois ages, l'un serait la somme des deux autres et les données seront redondantes. Nous aurions pu enlever cette dernière date, mais afin de ne pas enlever de données, nous l'avons gardée.

2.6 Transformations des prédicteurs

Nous allons utiliser ce qui à été trouvé lors de l'analyse exploratoire. Nous allons donc produire des combinaisons linéaires de variables, ou des transformations simples afin que cela se rapproche d'une valeur plus physique et plus pertinente pour l'étude.

Nous avons 5 transformations à faire :

— Volumes introduits, relation presque linéaire entre *Vsand_ PPLS* et *Vcarb_ PPLS*
— Les deux plages de données (avec *Avg*, Min* , Max**
— Les shots
— Les réponses

2.6.0.1 Vsand et Vcarb En appliquant une régression linéaire sur ce couple de valeur on obtient un bon résultat avec une pente très significative, et une explication de 84%. Nous conservons donc cette transformation en supprimant le prédicteur *Vsand*. Nous conservons les valeurs de la régressions pour opérer les mêmes transformations sur les données tests.

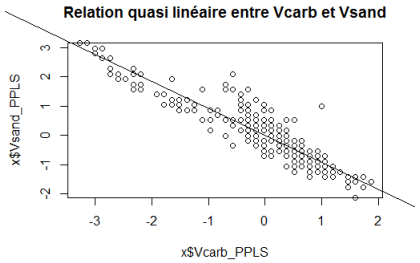


FIGURE 9 – Représentation de la relation linéaire entre V_carb et V_Sand

2.6.0.2 Plages de données Les prédicteurs **_ Rate_ Slurry..bpm* et **_ Treating_ pressure..KPa.* possèdent des valeurs *Min, Max, Avg*. Regardons si il est intéressant de comparer *Avg* avec $\frac{Min+Max}{2}$.

Tous calculs fait, nous obtenons deux graphiques avec des données qui ne paraissent peu corrélées.

Nous pensions obtenir des graphiques plus parlant, avec une allure polynomiale, mais rien ne semble intéressant. Le tableau des corrélations des anciens avec les nouveaux prédicteurs ne donne rien d'intéressant. Nous laissons les variables telles quelles.

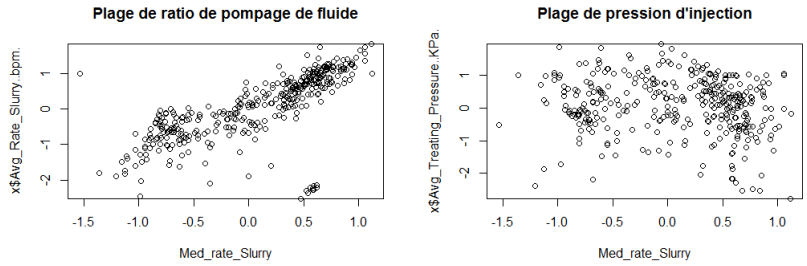


FIGURE 10 – Représentation des plages *Slurry* et *Treating* avec leur moyenne contre la moyenne de leur écart

2.6.0.3 Shots Le prédicteur *Shot_ Density* représente un nombre de shot sur une distance, *Shot_ Total* représente le nombre de shots. Le nouveau prédicteur $Shot_Dist = \frac{Shot\_Total}{Shot\_Density}$.

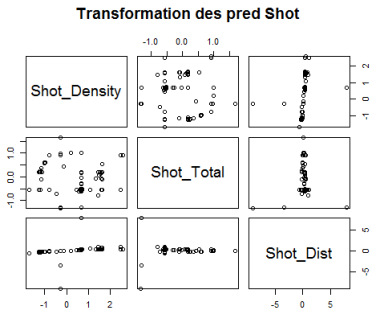


FIGURE 11 – Représentation des prédicteurs *Shots* du prédicteur crée *Shots_ Dist*

Le prédicteur obtenue et bien horizontal et vertical par rapport aux deux anciens. Toutefois il n'apporte pas de valeur à la prédiction du point de vue des corrélations, il ne fait que créer des outliers. Nous n'irons pas plus loin dans l'étude des Shots.

2.7 Transformation des réponses

Nous avons vu la forme des réponses sur la figure 4 ainsi que son profile ordonnée sur la figure **??**. Cette dernière ressemble à une accumulation exponentielle ou puissance, les transformations intéréssantes sont donc de la forme :

— $f_1 : x \longmapsto log(a * x + b)$
— $f_2 : x \longmapsto exp(-(a * x + b)$

Les paramètres a et b sont ici des facteurs d'échelles, ils permettent de ne pas avoir de valeurs négatives dans le logarithme ou dans la puissance. De plus la réponse sera à la fin centrée est réduite qui est une transformation supplémentaire.

2.7.1 Logarithme - f_1

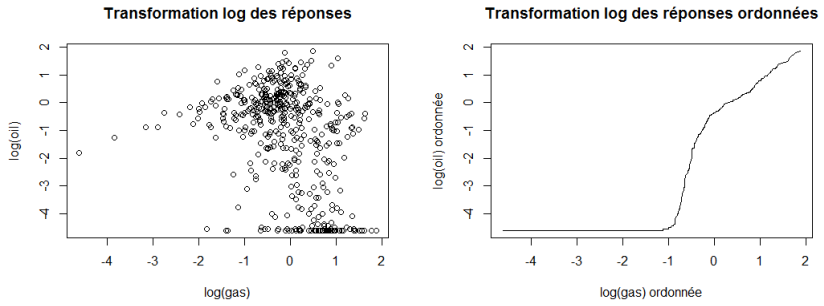


FIGURE 12 – Transformation logarithmique des réponses

L'application logarithmique nous donne la figure ci-dessus. On remarque que nous avons toujours les seuils de Oil et Gas mais a différents endroits. Les valeurs sont toujours regroupées dans un même endroit, mais ce regroupement est bien moins marqué.

Sur le second graphique nous remarquons que la transformation a agrandie le seuil des valeurs faibles, la variabilité est trop faible par rapport à la variabilité globale.

2.7.2 Exponentiel négative - f_2

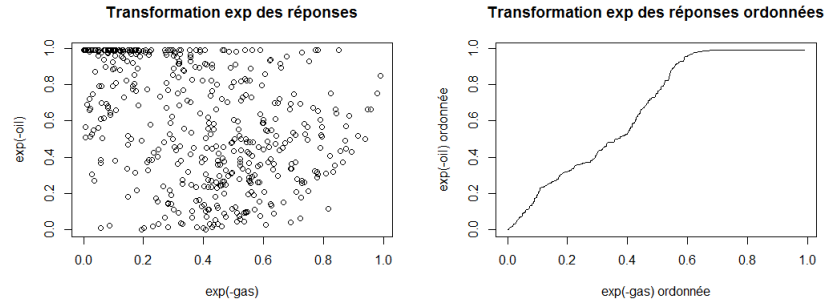


FIGURE 13 – Transformation exponentielle négative des réponses

LA transformation en exponentielle négative donne des résultats plus intéressant : la dispersion est plus flagrante, il y a donc moins de corrélation entre les réponses transformée. De plus la répartition des valeurs est plus linéaire que la transformation logarithmique.

3 Nos modèles

3.1 Nouvel espace : ACP & PLS

L'analyse en composantes principales est une méthode d'extraction de variables importantes (sous forme de composantes) à partir d'un grand ensemble de variables disponibles dans une dataset. Elle permet d'extraire un ensemble de caractéristiques de faible dimension à partir d'un ensemble de données de grande dimension avec comme objectif de capturer autant d'informations que possible. Avec moins de variables, la visualisation devient aussi beaucoup plus significative. De plus, on obtient des composantes indépendantes, ce qui permet d'éliminer les redondances et cela améliore les performances de certaines méthodes prédictives. Nous avons utilisé le package « FactoMineR » pour dérouler l'ACP, et le package « factoextra » pour la visualisation des résultats Ci-dessous, le graphe du pourcentage expliqué pour chaque composante (19 premières) :

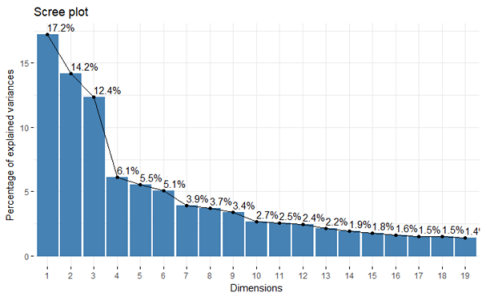


FIGURE 14 – scree plot

Et ci-dessous le cercle des corrélations des variables :

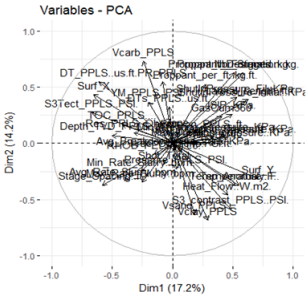


FIGURE 15 – cercle des corrélations

Avec la fonction pcr() du package « pls » on peut appliquer un régression sur les composantes principales, ce qui donne :

La recherche d'un modèle réduit avec des composantes indépendantes peut être un bon outil pour améliorer les modèles et les optimiser, typiquement pour les réseaux de neuronnes il peut être intéressant de considérer la projection des ind

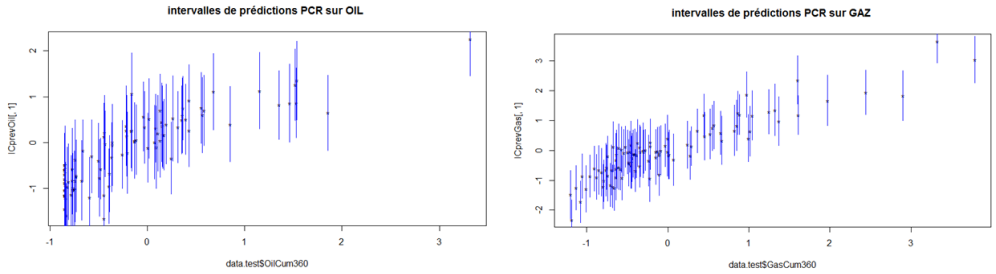


FIGURE 16 – intervalles de prédiction

3.2 Les différents modèles utilisés

3.2.1 Régression linéaire

La régression linéaire consiste a exprimer les réponses en fonction d'une combinaison linéaire des prédicteurs. Cette méthode a été présenté en cours donc une explication poussée n'est pas nécessaire.

On commencera par donner un modèle naïf pour les deux réponses Gas et Oil en utilisant tous les prédicteurs, puis on présentera une méthode pour trouver les prédicteurs les plus explicatifs des résultats pour avoir un modèle réduit, ensuite on effectuera une transformation sur les réponses et on essayera d'obtenir un meilleur score en jouant sur la longuer des intervalles.

Dans toute cette partie on utilisera des données résultant d'une partie du prétraitement effectué auparavant, il s'agit de l'imputation des données, des traitements des dates, traitement des outliers et la création des ages(nouveaux variables à partir des dates). On a réparti nos données Train en deux parties : une pour l'apprentissage et l'autre pour le test et la visualisation des résultats, ainsi on a utilisé comme un niveau de 95% pour les intervalles de prédictions sur ces données et la metric que nous avons codé pour voir notre score appliqué à cette data de test.

Modèle linéaire naïf

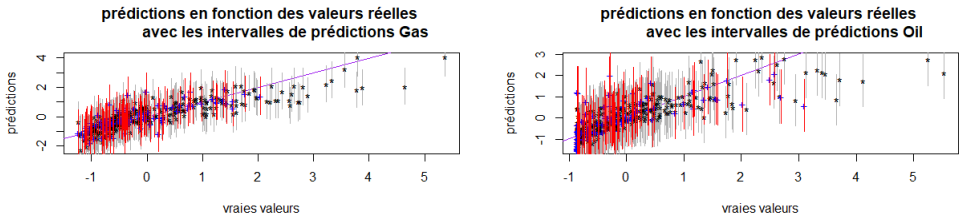


FIGURE 17 – les points en noir présente nos données d'apprentissage les intervalles en gris leurs intervalles de prédictions à 95% et les points en bleu sont les données du test avec leurs intervalles de prédictions à 95% en rouge

Avec nos données de test, nous obtenons un score de *6.731632*. En visualisant le graphe, on remarque que la majorité des points appartiennent aux intervalles mais ces intervalles sont clairement grands (surtout pour la réponse), ce qui explique le score.En faisant un *Summary* du modèle, plusieurs coefficients ont été considérés comme nuls selon les tests statistiques (données

par summary)(on peut vérifié ceci avec anova aussi). Avec ce modèle on obtient pour Multiple R-squared : 0.7197 et pour Adjusted R-squared : 0.6805.

Analyse des résidus

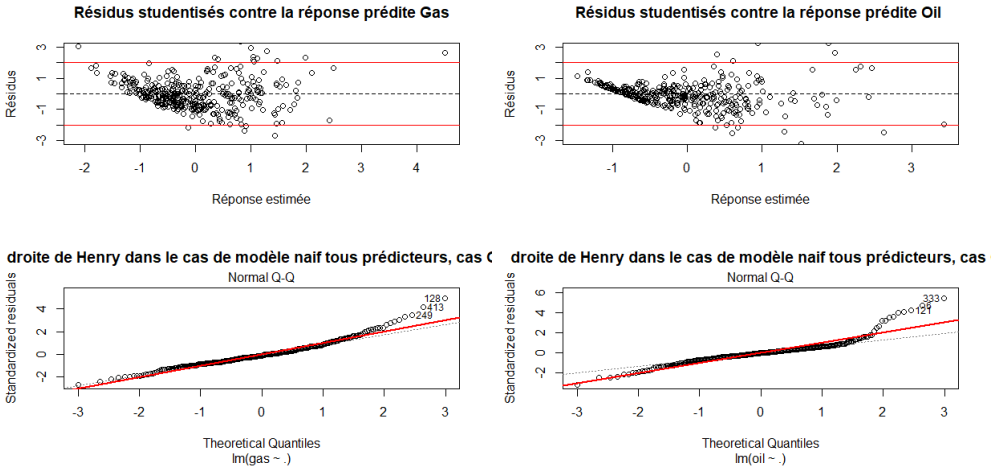


FIGURE 18 – Distributions des résidus par rapport aux réponses et test de normalité

L'analyse des résidus nous montre une dépendance entre les résidus et la réponse estimée, les résidus donc n'ont pas l'allure d'un bruit blanc.

Modèle naïf amélioré

Nous utilisons une fonction sur R step de la librairie stats à laquelle nous donnons en entrée le modèle complet et la zone les prédicteurs à tester, elle nous renvoie le meilleur modèle en se basant sur des tests statistiques AIC. Cette fonction teste tous les modèles possibles en utilisant tous les sous ensembles des prédicteurs possibles. Elle étourne le meilleur modèle en se basant sur le test de Ficher.Son but est d'obtenir le modèle le plus simple et le plus bon possible. Le meilleur modèle pour chacune des réponses sont :

```
Coefficients:
                                Estimate Std. Error t value Pr(>|t|)
(Intercept)                      0.03064    0.03375   0.908 0.364602
Surf_X                          -0.48494    0.14971  -3.239 0.001317 **
Surf_Y                           0.39445    0.15459   2.552 0.011161 *
Depth_TVD_PPLS..ft.              0.86646    0.09927   8.728  < 2e-16 ***
Pressure_PPLS..PSI.             -0.10347    0.04382  -2.361 0.018770 *
TOC_PPLS....                     0.25636    0.05771   4.442 1.21e-05 ***
Vsand_PPLS                      -0.06676    0.04654  -1.434 0.152355
RHOB_PPLS..g.cc.                 0.26424    0.06064   4.358 1.74e-05 ***
DT_PPLS..us.ft.                  0.27451    0.15928   1.723 0.085711 .
DTs_PPLS..us.ft.                 0.18775    0.05048   3.719 0.000234 ***
Temperature..F.                  0.09035    0.05076   1.780 0.075998 .
S3_contrast_PPLS..PSI.           0.25525    0.07311   3.491 0.000544 ***
Nbr_Stages                       0.35448    0.17945   1.975 0.049032 *
Proppant_Designed..kg.           1.64323    0.63220   2.599 0.009750 **
Proppant_in_Formation..kg.      -1.54792    0.60994  -2.538 0.011600 *
Avg_Breakdown_Pressure..KPa.    -0.05322    0.03865  -1.377 0.169410
Avg_Treating_Pressure..KPa.      0.10195    0.04984   2.045 0.041584 *
ShutInPressure_Initial..KPa.     0.11104    0.05609   1.980 0.048558 *
Shot_Total                       0.11654    0.06548   1.780 0.076002 .
Stage_Spacing..ft.               0.20392    0.09533   2.139 0.033134 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.6104 on 340 degrees of freedom
Multiple R-squared:  0.6494,    Adjusted R-squared:  0.6298
F-statistic: 33.14 on 19 and 340 DF,  p-value: < 2.2e-16
```

```
Coefficients:
                                Estimate Std. Error t value Pr(>|t|)
(Intercept)                      0.17496    0.08713   2.008 0.045439 *
Surf_X                          -0.47803    0.11074  -4.316 2.09e-05 ***
Surf_Y                          -0.82602    0.15156  -5.450 9.71e-08 ***
Date_Drilling                   -0.14419    0.06463  -2.231 0.026349 *
Lateral_Length..ft.              0.19176    0.06861   2.795 0.005489 **
Depth_TVD_PPLS..ft.             -0.81105    0.09443  -8.589 3.27e-16 ***
Erosion_PPLS..ft.               -0.10248    0.03733  -2.745 0.006365 **
Pressure_PPLS..PSI.              0.11226    0.03859   2.909 0.003861 **
TOC_PPLS....                    -0.25178    0.04657  -5.406 1.22e-07 ***
Vclay_PPLS                      -0.19667    0.03911  -5.028 8.04e-07 ***
RHOB_PPLS..g.cc.                -0.14254    0.05224  -2.728 0.006699 **
GR_PPLS..API.                    0.24860    0.09122   2.725 0.006758 **
Temp_Anomaly..F.                 0.08830    0.04506   1.960 0.050869 .
Zone                             0.30054    0.06324   4.753 2.98e-06 ***
Proppant_in_Formation..kg.       0.34965    0.11721   2.983 0.003060 **
Avg_Breakdown_Pressure..KPa.    -0.09877    0.03621  -2.727 0.006717 **
Avg_Rate_Slurry..bpm.            0.18348    0.04330   4.237 2.92e-05 ***
ISIP..KPa.                       0.09797    0.05052   1.939 0.053320 .
Proppant_per_ft..kg.ft.         -0.24678    0.09611  -2.568 0.010665 *
Stage_Spacing..ft.              -0.28816    0.06828  -4.220 3.14e-05 ***
Age_Production_Completion       -0.29010    0.11167  -2.598 0.009791 **
Age_Completion_Drilling         -0.56559    0.16654  -3.396 0.000765 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.5639 on 338 degrees of freedom
Multiple R-squared:  0.7206,    Adjusted R-squared:  0.7032
F-statistic: 41.51 on 21 and 338 DF,  p-value: < 2.2e-16
```

FIGURE 19 – Coef pour Oil sont à gauche et celle pour gas sont à droite

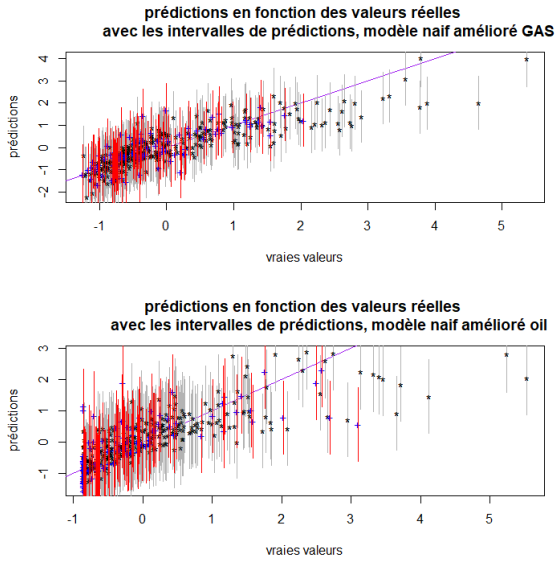


FIGURE 20 – les points en noir présente nos données d'apprentissage les intervalles en gris leurs intervalles de prédictions à 95% et les points en bleu sont les données du test avec leurs intervalles de prédictions à 95% en rouge (modèle amélioré - réduit)

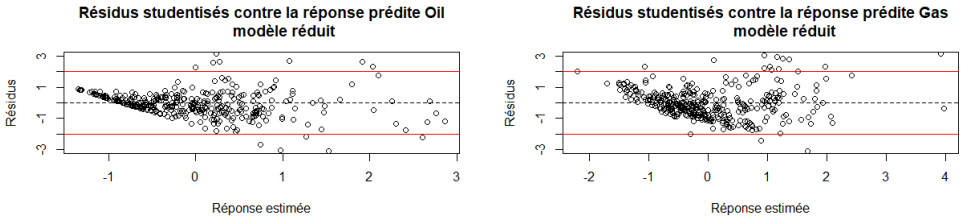


FIGURE 21 – Résidus VS réponses

On remarque que 73% de la variance de la réponse Gas est expliquée par ce modèle et que la la majorité des coefficients sont significatifs. Pour la réponse Oil nous avons 65% de la réponse expliquée est les coefficients sont aussi majoritairement significatifs. Les résidus ont presque la même allure qu'on prenant tous les prédicteurs (le nombre de prédicteurs retenus pour chaque modèles est presque la moitié de nombre total des prédicteurs). Le modèle a légèrement changé mais nous utilisé moins de prédicteurs (on ne garde que les prédicteurs expliquant le plus nos réponses) pour avoir ce modèle (presque la moitié).

Transformation des réponses avec modèle réduit

Dans cette partie, On va considérer la transformation log des réponses vu qu'ils présentent des quantités cumulées. Nous utiliserons en plus la fonction *Step* pour avoir le plus simple modèle possible. Pour les prédicteurs choisis par la fonction step on obtient ce qui suit(pred_o pour les prédicteurs de Oil et pred_g pour les prédicteurs de gas) : Avec ces prédicteurs et cette transformations notre modèle s'est amélioré : en effet nous obtenons pour Multiple R-squared

```
> pred_G
 [1] "(Intercept)"                  "Surf_X"
 [3] "Surf_Y"                       "Date_Drilling"
 [5] "Lateral_Length..ft."          "Depth_TVD_PPLS..ft."
 [7] "Erosion_PPLS..ft."            "Pressure_PPLS..PSI."
 [9] "TOC_PPLS...."                 "Vsand_PPLS"
[11] "Vclay_PPLS"                   "RHOB_PPLS..g.cc."
[13] "Res_PPLS..Ohmm."              "GR_PPLS..API."
[15] "Temperature..F."              "Temp_Anomaly..F."
[17] "S3Tect_PPLS..PSI."            "Zone"
[19] "Frac_Gradient..PSI.ft."       "Avg_Breakdown_Pressure..KPa."
[21] "Avg_Treating_Pressure..KPa."  "Avg_Rate_Slurry..bpm."
[23] "Min_Rate_Slurry..bpm."        "ShutInPressure_Initial..KPa."
[25] "ISIP..KPa."                   "Shot_Total"
[27] "Proppant_per_ft..kg.ft."      "Stage_Spacing..ft."
[29] "Age_Production_Completion"
> pred_O
 [1] "(Intercept)"                  "Surf_X"
 [3] "Surf_Y"                       "Lateral_Length..ft."
 [5] "Depth_TVD_PPLS..ft."          "Erosion_PPLS..ft."
 [7] "Pressure_PPLS..PSI."          "TOC_PPLS...."
 [9] "Vcarb_PPLS"                   "Vsand_PPLS"
[11] "Vclay_PPLS"                   "YM_PPLS..PSI."
[13] "Res_PPLS..Ohmm."              "DT_PPLS..us.ft."
[15] "Temperature..F."              "Temp_Anomaly..F."
[17] "S3Tect_PPLS..PSI."            "S3_contrast_PPLS..PSI."
[19] "Heat_Flow..W.m2."             "Zone"
[21] "Avg_Treating_Pressure..KPa."  "Max_Rate_Slurry..bpm."
[23] "ShutInPressure_Fil..KPa."     "ISIP..KPa."
[25] "Shot_Density..shots.ft."      "Proppant_per_ft..kg.ft."
[27] "Stage_Spacing..ft."           "Age_Completion_Drilling"
```

0.8104, pour Adjusted R-squared *0.7944*, pour F-statistic *50.53 on 28 and 331 DF* et pour p-value une valeur $<$ 2.2e-16.

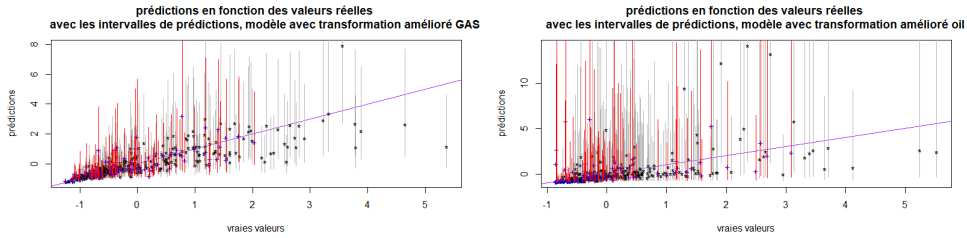


FIGURE 22 – Intervalles en utilisant la transformation log : les points en noir présente nos données d'apprentissage les intervalles en gris leurs intervalles de prédictions à 95% et les points en bleu sont les données du test avec leurs intervalles de prédictions à 95% en rouge

Vu la transformation inverse du logarithme (exp), on obtient des intervalles très grands ce qui augmente significativement notre score (on obtient un score d'environ 7.09). L'idée par la suite sera d'essayer de réduire la longueur de ces intervalles.

Analyse des résidus

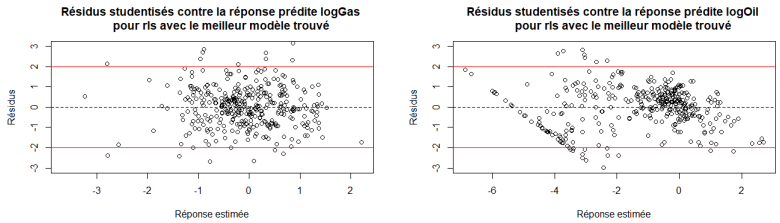


FIGURE 23 – Résidus Vs les réponses

On remarque bien que nos résidus sont meilleurs et ont une distribution moins dépendante de nos réponses que sans faire les transformations. Ce qui montre l'intérêt encore une fois de l'analyse exploratoire et d'effectuer des transformations aux variables.

Vers des intervalles plus optimisés

Dans cette partie, nous essayerons de diminuer la longueur des intervalles tout en essayant d'avoir le meilleur score possible. L'idée est de détecter les points Gas et Oil donnant des surfaces qui dépassent 10, donc éviter des intervalles dont la longueur est plus grande de $\sqrt{10}$. Puis on modifie ces intervalles en mettant un décalage par rapport à la valeur moyenne d'une valeur plus petite que $\sqrt{10}$ à trouver. Après avoir fait plusieurs essaies on conclue que la meilleur valeur à ajouter et retrancher à la valeur ponctuelle prédite pour avoir les intervalles de prédictions est de $\sqrt{7}$. Avec cette façon de faire notre score passe de *7.09* à *3.9*. Ceci représente un gain assez considérable. pour les graphes des données avec les intervalles on obtient ce qui suit :

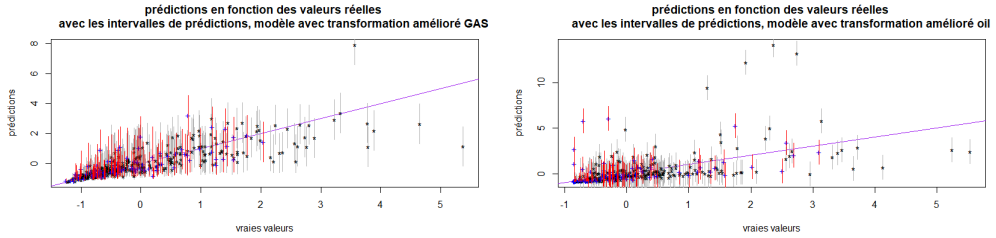


FIGURE 24 – Intervalles optimisés en utilisant la transformation log : les points en noir présente nos données d'apprentissage les intervalles en gris leurs intervalles de prédictions à 95% et les points en bleu sont les données du test avec leurs intervalles de prédictions à 95% en rouge

Cette méthode d'optimisation des intervalles peut aussi être utiliser avec les autres modèles pour essayer d'avoir le meilleur score possible.

3.2.2 Régression pénalisée

Pour cette partie, nous avons recours à la fonction glmnet (package glmnet) avec le paramètre α en plus (égal à 1 pour la méthode LASSO, α=0 pour RIDGE et égal à 0.5 pour Elastic-Net).

a) Le LASSO, ou Least Absolute Shrinkage and Selection Operator

La méthode LASSO, ou pénalité L1, introduit ici à notre problème d'optimisation une nouvelle pénalité λ >=0, paramètre de régularisation tel que :

$$\min_{\beta_0,\beta_1,\ldots,\beta_p} \frac{1}{2}\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{i,j}\right)^2 + \lambda\sum_{j=1}^{p}|\beta_j|$$

(Si $\lambda = 0$, alors on revient à l'estimateur $\hat{\beta}$ usuel des moindres carrés ordinaires).
Nous avons ici un problème convexe, ce qui facilite donc la minimisation.
Le choix de λ est important. Quand λ augmente, alors plus $\hat{\beta}(\lambda)$ est «creux» (cf. graphique en dessus), donc on autorise moins de variables et alors le biais augmente et la variance diminue. C'est pour cela qu'on parle d'une méthode de «contraction» des coefficients qui permet de contrôler la variance de l'estimateur. La méthode lasso par validation croisée (fonction cv.glmnet) nous aide à choisir le λ optimal, valeur de λ qui nous donne le minimum de la moyenne de l'erreur de cross-validation.
La méthode Lasso permet donc une sélection parcimonieuse des variables explicatives en fonction du paramètre λ, ce qui permet de mieux interpréter notre modèle.

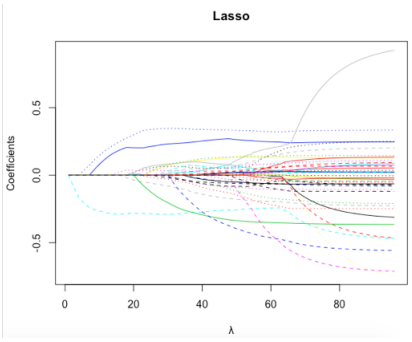


FIGURE 25 – Plus λ est grand, plus des coefficients tendent vers 0 et on autorise donc moins de variables explicatives.

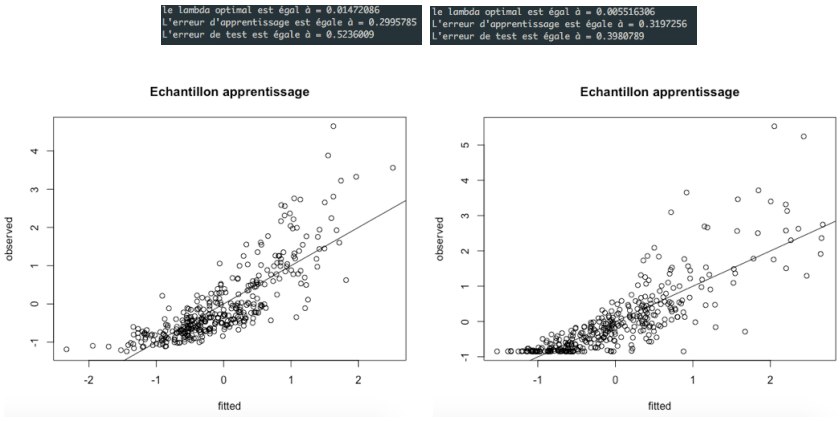


FIGURE 26 – Predictions ponctuelles en fonction des données observées pour l'échantillon d'apprentissage (Gas à gauche et ce lui de Oil à droite

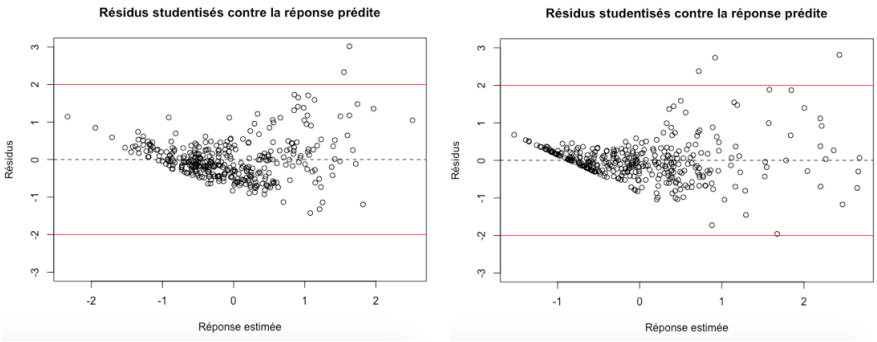


FIGURE 27 – Modele Gas à gauche, celui de Oil à droite.

b) Pénalisation Ridge :

La méthode Ridge est basée sur la même idée que la régression LASSO, en remplaçant la norme Euclidienne L1 de la pénalité par la norme L2. En effectuant la même procédure que pour

le modèle Lasso (avec alpha=0 sur la fonction glmnet de r, alors qu'on avait alpha par défaut =1)

```
le lambda optimal est égal à = 0.1543474
L'erreur d'apprentissage est égale à = 0.2981498
L'erreur de test est égale à = 0.560735
```

```
le lambda optimal est égal à = 0.06051603
L'erreur d'apprentissage est égale à = 0.3404783
L'erreur de test est égale à = 0.3924303
```

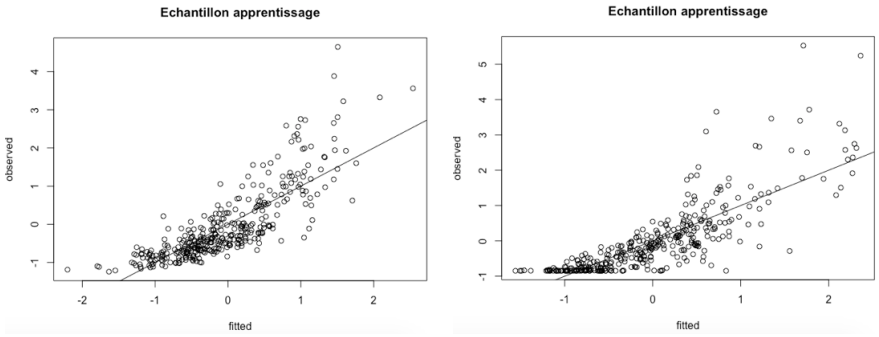


FIGURE 28 – Predictions ponctuelles en fonction des données observées pour l'échantillon d'apprentissage (Gas à gauche et ce lui de Oil à droite

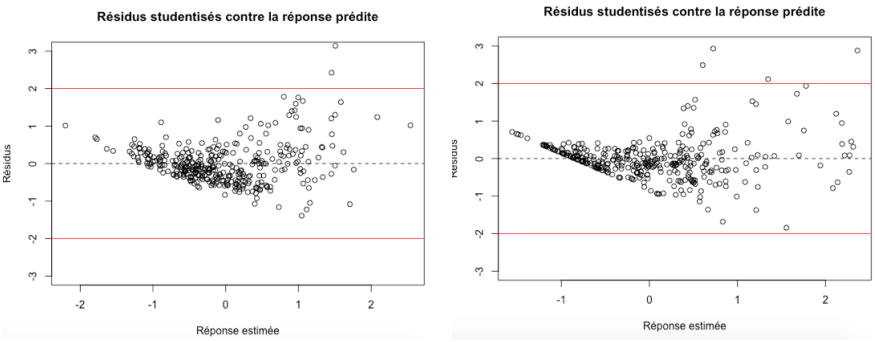


FIGURE 29 – Modele Gas à gauche, celui de Oil à droite.

c) Pénalisation Elastic-Net :

L'idée est donc d'ajouter au Lasso la pénalité Ridge. Le problème d'optimisation est donc le suivant :

$$\min_{\beta \in \mathbb{R}^p} \frac{1}{2}\|y - X\beta\|_2^2 + \lambda_1 \|\beta\|_1 + \lambda_2 \|\beta\|_2^2$$

avec $\lambda_1 \geq 0$ et $\lambda_2 \geq 0$.

```
le lambda optimal est égal à = 0.02675883
L'erreur d'apprentissage est égale à = 0.2990213
L'erreur de test est égale à = 0.5253332
```

```
le lambda optimal est égal à = 0.005017854
L'erreur d'apprentissage est égale à = 0.3139895
L'erreur de test est égale à = 0.4008217
```

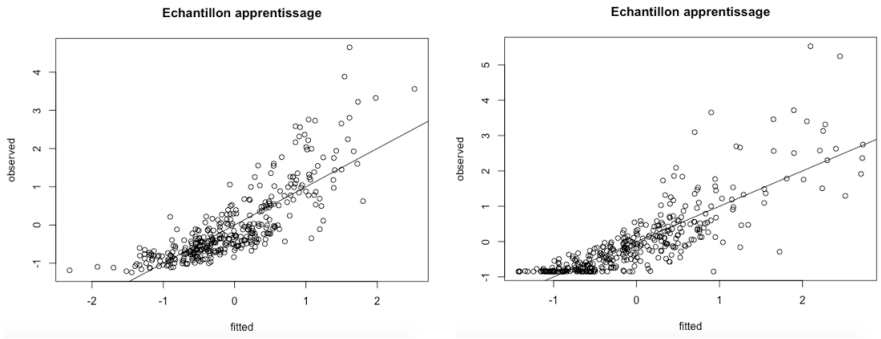


FIGURE 30 – Predictions ponctuelles en fonction des données observées pour l'échantillon d'apprentissage (Gas à gauche et ce lui de Oil à droite

Analyse des résidus

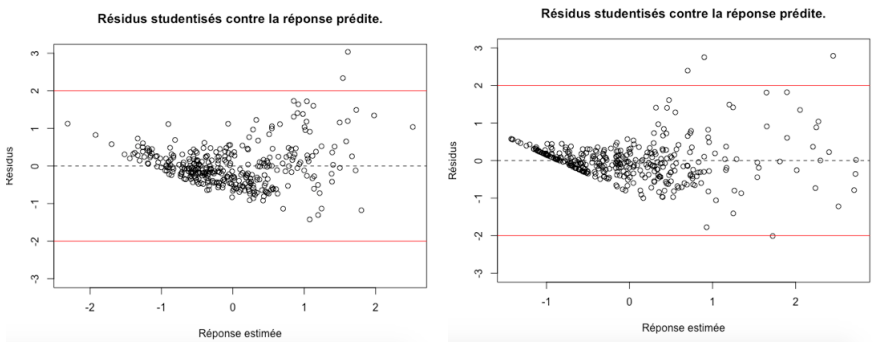


FIGURE 31 – Modele Gas à gauche, celui de Oil à droite.

Retour sur les résidus pour les 3 méthodes

Dans les figures des résidus pour les 3 méthodes, on remarque que nous avons une certaine dépendance par rapport aux résultats, ce qui peut remettre en question notre modèle. Ceci n'empêche que cette régression est intéressante dans le sens où elle permet d'effectuer une meilleur sélection des prédicteurs.

On remarque que la méthode Elastic-Net est un peu plus performante que la méthode Ridge et Lasso.
Ce qui nous permet de conclure qu'une pénalisation L1 et L2 associées permet une prévision un peu plus fiable que si l'on utilisait les deux toutes seules dans notre modèle.
Seulement pour les méthodes Lasso, Ridge et Elastic-Net, nous pouvons seulement obtenir les prévisions ponctuelles, et non les intervalles de prédiction/confiance. De ce fait, afin de pouvoir obtenir l'intervalle de prédiction, nous pouvons obtenir une estimation de l'intervalle de prédiction avec du bootstrapping.

3.2.3 Réseaux de neurones

L'utilisation des réseaux de neuronnes est devenue répondue ces derniers temps en deep learning et machine learning. Dans notre cas, on va faire une utilisation basique des réseaux de neuronnes.

Avec tout les prédicteurs

En utilisant un nouvel espace de prédicteurs indépendant

3.2.4 Forêts aléatoires

Dans cette partie, nous avons recours au randomForest, particulièrement performant pour les problématiques de prédiction. Le principe du randomForest est de faire la moyenne des prévisions de plusieurs modèles indépendants pour réduire la variance et donc l'erreur de prévision, modèles construits en sélectionnant plusieurs échantillons bootstrap, soit des tirages avec remise. Nous obtenons donc les résultats suivants :

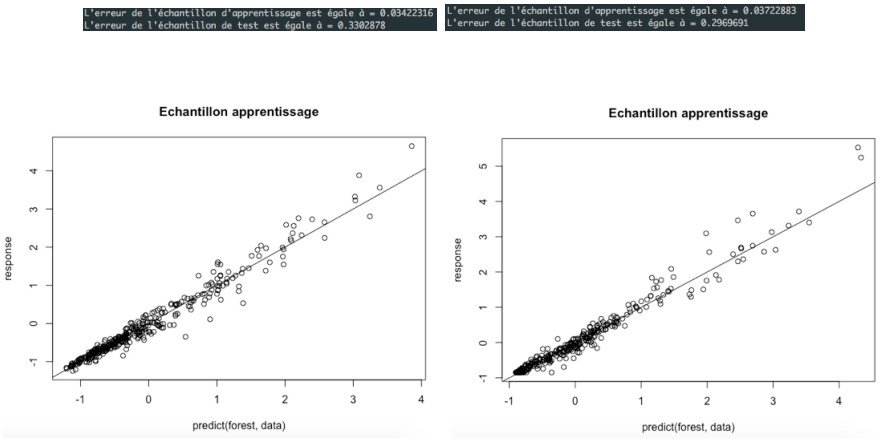


FIGURE 32 – Predictions ponctuelles en fonction des données observées pour l'échantillon d'apprentissage (Gas à gauche et ce lui de Oil à droite

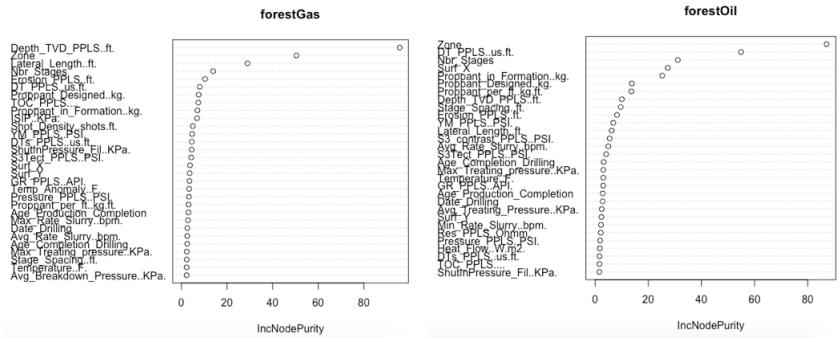


FIGURE 33 – Importance de chaque variable, présentant leur contribution au modèle. (Gas à gauche et ce lui de Oil à droite

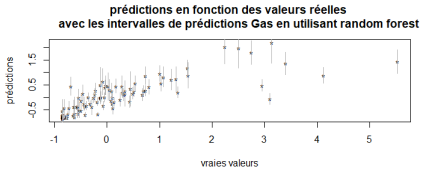
```
               Type of random forest: regression
                     Number of trees: 200
No. of variables tried at each split: 42

          Mean of squared residuals: 0.261797
                    % Var explained: 71.49
 Bias correction applied:
 Intercept:  0.003262061
     Slope:  1.086054
```

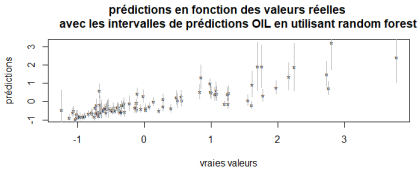
```
Call:
 randomForest(formula = response ~ ., data = data, mtry = 42,      ntree = 200, corr.bias = TRUE, nPerm
= 1)
               Type of random forest: regression
                     Number of trees: 200
No. of variables tried at each split: 42

          Mean of squared residuals: 0.2571748
                    % Var explained: 74.63
 Bias correction applied:
 Intercept:  -0.02270551
     Slope:  1.036444
```

En effet, nous pouvons améliorer les prédictions du modèle en jouant sur les paramètres :

-Les paramètres mtry (appartient à [0,p], p le nombre de variables), qui est le nombre de variables explicatives échantillonnées à chaque noeud et ntree le nombre d'arbres développés. Pour ce faire, nous pouvons faire appel à une boucle pour étudier la précision du modèle en fonction de la variation des paramètres.Le paramètre mtree effecte significativement notre modèle dans le sens où si mtree est trop grand on arrive à un *overfitting* et s'il est petit nous avons un *underfitting*

-corr.bias, paramètre à utiliser avec des pincettes. En effet ce paramètre nous est utile afin d'effectuer une correction du biais de la régression. Après l'avoir essayé plusieurs fois, nous avons remarqué que soit l'on obtenait un résultat un peu meilleur, soit un peu moins bon. De ce fait il est nécessaire de tester les deux possibilités avant de fixer le paramètre corr.bias, même si le paramètre n'offre pas un résultat très différent du résultat initial. En utilisant les prédictions sur data.test on obtient un score de 3.77794725 obtenu pour un mtree de 500.

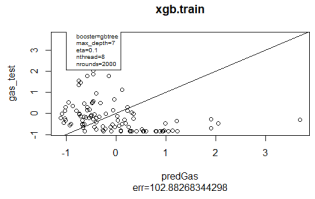




Les figures ci-dessus donnent les prédictions en fonction des valeurs réelles et nous remarquons qu'on obtient un overfiting, les intervalles sont petits et dépendent trop de nos données ce qui donne un scrore qui est satisfaisant mais à améliorer.

3.2.5 XGBoost

Comme connu, le BOOSTING est une technique ensembliste qui consiste à agréger des classifieurs (modèles) élaborés séquentiellement sur un échantillon d'apprentissage dont les poids des individus sont corrigés au fur et à mesure. La descente du gradient est une technique itérative qui permet d'approcher la solution d'un problème d'optimisation. Xgboosting est donc une méthode gradient boosting qui minimise une fonction coût. Dans cette partie nous avons tenter l'utilisation de cette méthode vu sa réputation d'être performante dans les cas non linéaire. Nous n'avons pas pu obtenir un très bon modèle, on obtient ce qui suit :



Cependant, cet axe restera une bonne route à exploiter. Sachant que des fonctiosn utilisant la même méthode pour construire des modèles, sont implémenter en python, les essayer peut être une bonne idée.

NB : Il reste toujours intéressant de tenter l'utilisation de python vu qu'il est le numéro 1 utilisé par les entreprises en dataScience et que connaître construire les modèles avec d'autres langages est une bonne compétence.

3.2.6 Les intervalles de prédictions

Beaucoup de modèles ne permettent pas nativement de faire des intervalles de prédictions. Pour cela nous répétons les modèles sur différents échantillons bootstrapés depuis les données d'entraînement. Il suffit alors de prendre sur l'ensemble des résultats bootstrap, un intervalle de prédiction comme $[2.5\%, 97.5\%]$. Le bootstrap empirique est :

$$prediction(data, \hat{\beta}^{*b}) + \epsilon^{**b}$$

Nous avons ainsi une fonction qui prend en entrée le modèle, qui lance 1000 bootstraps, et qui ressort l'intervalle de prédiction.

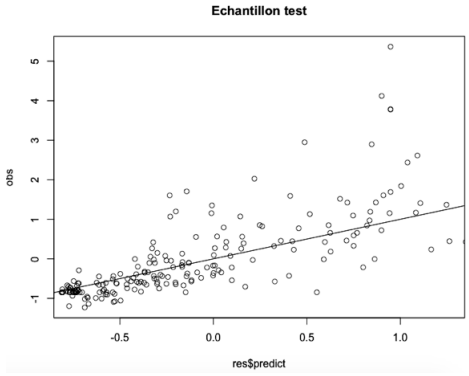
Après avoir obtenue les intervalles de prédictions, il faudra s'assurer que nos intervalles ne sont pas grands (ils doivent être les plus petits possibles) qu'il contiennent les valeurs prédites (la majorité). On peut jouer sur la longueur en utilisant plusieurs méthodes :

- Modifier directement les intervalles de prédictions obtenus et s'ils dépassent racine de 10 on diminue la marge entre la moyenne et les extrémités.
- Modifier le choix des paramètres notamment les quantiles à utiliser, le nombre de bootstrap à faire...

3.3 Blending

3.3.1 COBRA

COBRA, ou COmBined Regression Alternative, est une stratégie d'agrégation de techniques existantes pour la prédiction, afin de tirer le meilleur de chacune d'elles et d'éliminer autant que possible la phase de spécification d'un modèle.Plus précisément, les modèles de régression utilisés sont : Lasso, Ridge, RandomTree et RandomForest. Nous obtenons les résultats suivants pour la variable à expliquer Gas : Cette méthode de prédiction récente a pour avantages d'être




```
Quadratic risk
     Lars     Ridge      CART        RF     COBRA
0.7572065 0.4585731 0.7517398 0.3354827 0.3480605
L'erreur de test est égale à = 0.3548282
```

très performante et rapide. De plus, on a la possibilité d'améliorer notre score en jouant avec

les paramètres de COBRA (vu la multitude de paramètres de la fonction) et en appliquant une transformation à la variable réponse.

3.3.2 Agrégation des modèles avec le CMAES

Nous allons ici essayer d'améliorer les prédictions en prenant des combinaisons linéaires des prédictions offertes par les différents modèles.

D'un point de vue mathématique, ceci n'est pas intéressant, mais il se trouve que dans la pratique, en utilisant de la cross validation et un algorithme d'optimisation, qu'il est possible d'améliorer notre score. Le but est de prendre les meilleurs prédiction de chaque de modèles. Soit trouver le meilleur $\alpha = (\alpha_1 ... \alpha_k)$ afin de maximiser :

$$\alpha_1 * prediction_1(x[bootstrap,]) + ... + \alpha_k * prediction_k(x[bootstrap,])$$

Nous n'avons pas eu le temps d'implémenter cette méthode mais il est possible d'utiliser l'algorithme du CMAES comme pour les intervalles non symétriques pour améliorer le score.

3.4 Finalisation

3.4.1 Cross-Validation

Plusieurs fonctions peuvent être utilisées pour la cross validation notamment train de la biblio caret qui s'occupe d'optimiser les paramètres. Plusieurs fonctions qui sont déjà utilisées ont une cross-validation intégrée comme pls ou xgboost ainsi qu'on ridge. Cette étape est importante pour rendre nos modèle les plus robuste possible et les moins dépendant de nos données d'échantillonage, ainsi que pour estimer des paramètres

3.4.2 Taille des intervalles

3.4.2.1 Motivations En remarquant qu'en changeant juste la taille des intervalles de prédictions (level) le score s'améliorait, nous avons eu l'idée de boucler sur le level des intervalles pour le gaz et pour le oil, puis on stocke les scores obtenus dans une grille, pour finalement trouver le meilleur score et donc les niveaux d'intervalles de prédiction de chaque réponse relatifs à ce score.

3.4.3 Intervalles non symétriques - CMAES

3.4.3.1 Motivations Afin d'obtenir une meilleure prédiction, nous pouvons ajuster le centrage des intervalles vers les zones où la prédiction a plus de probabilités d'être afin d'obtenir le maximum de points dans un intervalle le plus petit possible.

Lorsque que l'on trace Oil contre Gas (sans transformations), les points sont clairement concentrés vers les valeurs basses. Il serait intéressant de décaler les intervalles de prédiction dans la zone où il y a le plus de probabilité de rencontrer une bonne valeur. Ainsi, si après avoir fait le modèle il est encore possible d'améliorer succinctement le score.

3.4.3.2 Application Ici nous traitons avec la régression linéaire et les intervalles de prédiction sont donc symétriques. Nous pourrions tâtonner et déplacer légèrement les intervalles vers les valeurs de Oil et Gas basses pour améliorer le score ; poussons le raisonnement et essayons de résoudre le problème complet : Quel de pourcentage de l'intervalle de prédiction doit être au-dessus de la prédiction pour les réponses Oil et Gas pour minimiser le score ?

Mathématiquement, si on appelle $\alpha = (\alpha_1 \alpha_2)$ le vecteur des proportions des intervalles au-dessus de la prédiction, le problème revient à trouver α qui maximise le score.

Pour cela nous avons utilisé l'algorithme CMAES vu en optimisation globale sur une régression linéaire avec tous les prédicteurs sur des données pré-traitées. Nous allons ici montrer ses résultats avec une régression linéaire simple, sur des données non pré-traitées. Nous avons le résultat suivant :

```
xbest = 0.4296694 0.5518359
fini = 6.471526
fbest = 5.892317
```

FIGURE 34 – Intervalle de prédiction asymétrique par rapport à a prédiction initiale $\alpha > 0.5$

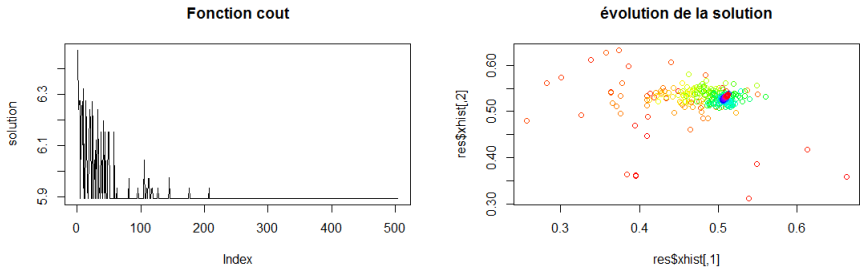


FIGURE 35 – Évolution de la fonction coût (score) et de la solution $(\alpha_1 \alpha_2)$ en fonction itération du CMAES

La solution finale est donc décalée de 8% « en bas » et 5% « à gauche » si l'on se réfère aux graphiques plus haut. Le score est bien amélioré en décalant les intervalles.

Nous pouvons toutefois dire que cette opération n'a pour but que d'améliorer le score sans prendre en considérations les données. Cette fonction pourrait d'avérer inutile si la réponse Oil contre Gas est bien transformée et ne contient plus de « paquets » de données.

4 Conclusion

A la lumière de cette analyse, nous avons constaté que manipuler des données réelles était un exercice extrêmement instructif. On a tout d'abord remarqué que l'analyse exploratoire était une étape importante dans le traîtement de données, car celle-ci nous permettait d'avoir une vision globale de nos données : relations possibles entre plusieurs variables, présence d'outliers...

Contrairement aux données présentes dans les packages prédéfinis, les données réelles (ou 'raw' data) présentent des irrégularités parfois trop importantes et nécessitent donc un prétraîtement afin de les rendre exploitables. Ainsi, on constate que l'étape du prétraitement est une étape primordiale car elle fertilise le terrain pour les méthodes prédictives.

Effectivement, au cours des premiers jours du lancement du challenge, nous avons rencontré plusieurs difficultés avec les valeurs manquantes et outliers : remplacer les NA par la moyenne ou bien supprimer les individus concernés, réduire les valeurs aberrantes ou carrément supprimer les variables avec trop de valeurs aberrantes...

Nous avons pu surmonter ces difficultés en faisant appel à la fonction de MissForest.

Après avoir effectué ces étapes de prétraitement, nous nous sommes tournés vers la transformation de la variable réponse, qui s'est avéré être un moyen très efficace pour augmenter la précision de nos prédictions.

Nous avons essayé plusieurs méthodes de prédiction dans notre étude, et nous avons pu noter que les modèles les plus performants sont le randomForest et le xgboost. Le xgboost et le COBRA sont des alternatives interessantes pour la prédiction, mais leur lancement récent (2014 pour xgboost, 2016 pour COBRA) les rend plus complexes à utiliser pour manque de documentation sur les paramètres utilisés. Nous avons aussi pu voir que la variation des paramètres dans les fonctions des méthodes prédictions pouvait jouer un rôle important dans la précision de nos prédiction, et donc qu'on pouvait optimiser celle-ci en faisant varier ces paramètres. Enfin, concernant les méthodes utilisés , Random Forest semble la meilleure méthode qui donne le meilleur score avec des petits intervalles. Face aux limites que les modèles linéaires présentent, les modèles non linéaire sont les plus performant dans un cas avec des données aussi varié comme celui là.Xgboost et les réseaux de neuronnes sont aussi desméthodes répondus par leurs performances, vu le manque de maitrise de ces méthode nous n'avons pas montré leur efficacité, il sera donc envisageable de developper encore plus nos script la dessus pour avoir des meilleurs score.