Université Pierre et Marie Curie

Compléments sur le modèle linéaire

Challenge Data

Rapport de projet

Auteurs
Heikel Jazi
Inès Krissaane
Laurent Temo

Professeur
Agathe Guilloux

14 Janvier 2018

Sommaire

Introduction

Ce rapport présente un projet du type *data challenge* réalisé dans le cadre du cours **Compléments sur le modèle linéaire** mené par Mme Guilloux. Ce challenge proposé par la Société Générale sur la plateforme *Challenge Data* s'intitule **Prédire la tendance de la production de pétrole brut.**

Comme pour chaque challenge, un échantillon d'entraînement et un échantillon de test sont proposés. L'échantillon d'entraînement se compose d'une liste d'individus pour lesquels sont fournis un certain nombre de caractéristiques liées à la production de pétrole ainsi qu'une donnée sur laquelle se feront les prédictions. Il permet de construire et de perfectionner notre algorithme. Un échantillon de test est également fourni contenant les individus pour lesquels la prédiction doit être faite à partir de notre algorithme. La performance de cet algorithme est ensuite évaluée par le site en fonction du taux de prédiction correct pour ce jeu de données. Un classement en temps réel est disponible.

Dans le cadre de ce projet, nous avons décidé d'utiliser les langages R et Python respectivement pour faire l'analyse des données et pour implémenter nos algorithmes. En effet, R est le langage par excellence dédié à l'analyse statistique puisqu'il met à disposition une grande variété de techniques statistiques et graphiques. Il permet également une rapide intégration de toutes les méthodes statistiques. De même, en tant que langage de programmation fonctionnelle, avec Python, on peut très facilement faire appel à des bibliothèques implémentant déjà les algorithmes utilisés et en l'occurrence, la bibliothèque **scikit-learn** qui fournit la plupart des algorithmes de machine learning.

1 Observation et préparation des données

1.1 Description des données

L'échantillon d'entraînement contient des données sur 76 pays producteurs de pétrole brut dans le monde entier pour la période allant de janvier 2002 à août 2016 avec plusieurs caractéristiques telles que les niveaux de stock primaire à la fin du mois (*'Closing stocks'*), les exportations (*'Exports'*), les importations (*'Imports'*), la consommation des raffineries (*'Refinery Intake'*), les coûts de clotûre (*'WTI Price'*), et la somme des features (*'SumConsing stocks'*, *'SumExports'*, *'SumImports'*, *'SumProduction'*, *'SumRefinery intake'*). Ces différentes observations sont données pour chaque mois et le préfixe "diff" nous indique le mois d'enregistrement. Par exemple, 12 diffExports est la valeur la plus proche de la tendance et 1 diffExports la valeur la plus éloignée de la tendance que nous essayons de prédire. L'objectif du challenge est de prédire l'augmentation ou la diminution de la production de pétrole brut pour un échantillon d'individus test. Il s'agit alors d'une **régression logistique**.

1.2 Traitement des données

L'objectif de ce challenge est de **prédire la tendance de la production de pétrole brut** des différents pays présentés et anonymisés dans l'échantillon de test. Dans ce but, l'un des premiers comportements à adopter face à ce type de challenge est la compréhension et le traitement optimal des données. Décrite dans le paragraphe précédent, la compréhension est indispensable à l'élaboration de stratégies permettant notamment de traiter les valeurs manquantes : l'échantillon d'entraînement ne peut pas être utilisé s'il possède ne serait-ce qu'une valeur manquante. Diverses stratégies peuvent être adoptées. Certaines sont plus rapides mais néanmoins moins rigoureuses consistant par exemple à remplir le tableau de données par des zéros ou simplement supprimer les lignes où figurent des valeurs manquantes. Ces techniques ont pour seul but d'obtenir rapidement un score référence. En visualisant les données sur des échantillons choisis aléatoirement (cf partie 2.3), on observe qu'une **interpolation quadratique** est la méthode la plus adaptée pour répondre au problème 3.3. A noter que notre échantillon d'entrainement comprend environ 0.36% de données manquantes dans 24 variables explicatives soit une faible proportion par rapport aux 122 variables du tableau de données : ce n'est donc pas un enjeu essentiel contrairement à d'autres challenges.

L'une des méthodes permettant d'obtenir une amélioration des performances est la **binarisation** c'est-à-dire la construction pour chaque catégorie d'une variable, un nouveau feature binaire. Par exemple, pour la variable *'country'* choisie après de nombreux tests, la binarisation consiste à faire comprendre à la machine que chaque pays correspond à un nombre. Cette stratégie menée sur les variables *'country'* et *'month'* peut paraître banale et assez coûteuse puisqu'elle augmente drastiquement le nombre de dimension. Au contraire, cette astuce s'est avérée déterminante puisqu'elle nous a permis avant une quelconque réduction de dimension d'obtenir un score honorable.

1.3 Visualisation graphique

Après avoir observé attentivement les variables comportant des données manquantes (voir annexe 1), l'interpolation quadratique s'avère être le choix le plus pertinent contrairement à une simple interpolation linéaire ou au remplissaqe par les valeurs précédentes. Ce choix a été confirmé par une amélioration de notre de score sur la plateforme *Data Challenge*.

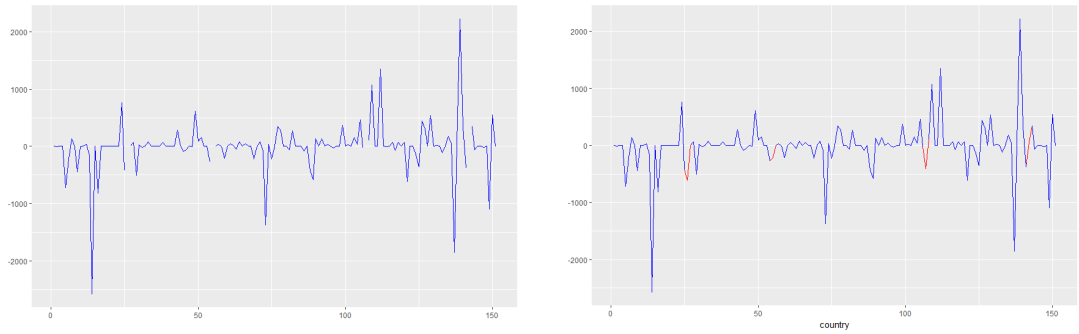

Figure 1: Représentation partielle de la variable X1 diffImports. En rouge , l'interpolation quadratique permet de remplacer les données manquantes.

Nous avons également réalisé une analyse graphique afin d'établir les liens plus ou moins élévés entre les variables explicatives et la variable label.

Ces premières analyses donnent une idée globale du jeu de données mais sont insuffisantes. En effet, on se heurte au nombre élevé de variables explicatives, 123 avec uniquement deux variables catégorielles *country* et *month*.

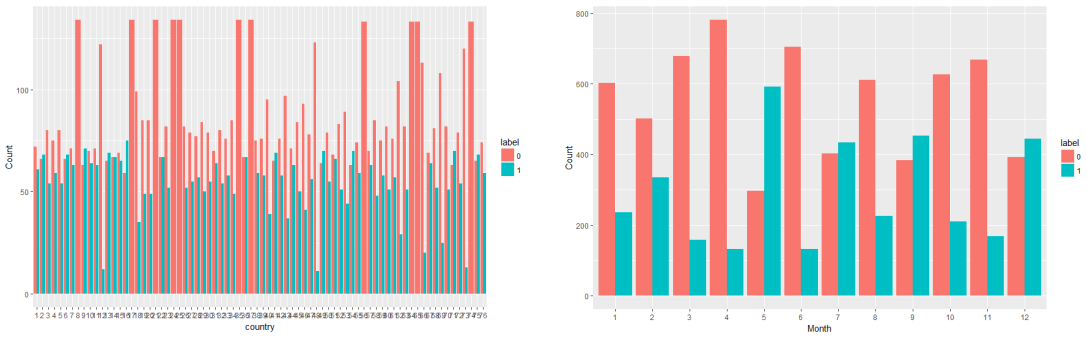

Figure 2: Impact du pays et du mois sur l'évolution du pétrole brut

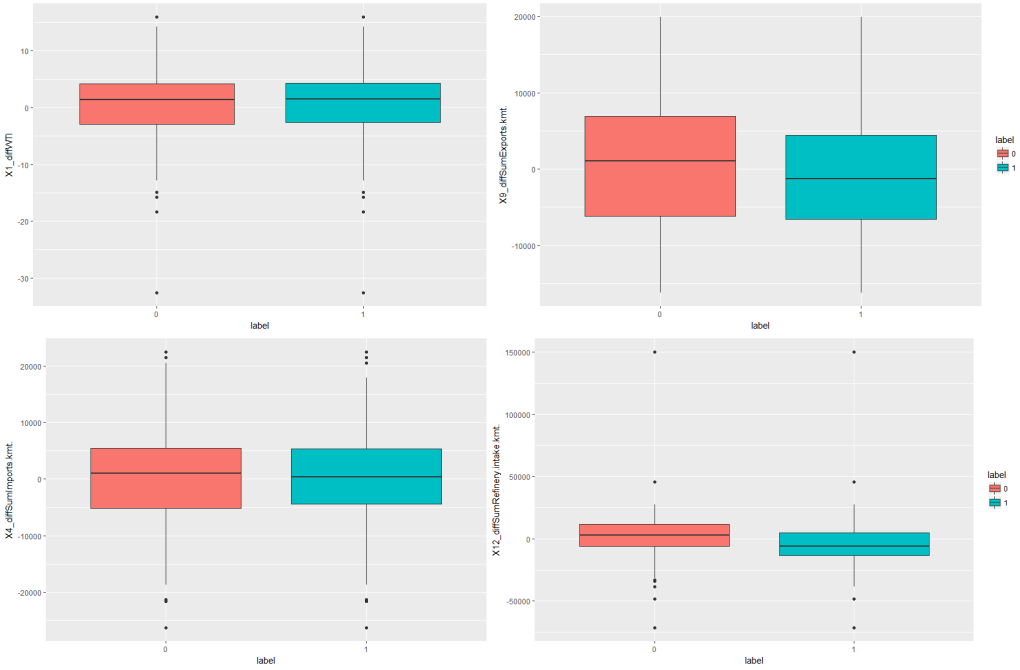

Figure 3: Boxplot des variables *X1 diffWTI*, *X9 diffSumExports*, *X4 diffSumImports*, *X12 diffSumRefinery.intake.kmt* en fonction du label. Les distributions sont proches : ces variables explicatives ne semblent pas significatives.

2 Réduction de dimension

Après avoir réalisé une étude graphique des données, il est difficile de prime abord d'éliminer des variables explicatives. Intuitivement, elles ont toutes un lien sensé avec la production de pétrole. Cependant, les différences notables avec d'autres jeux de données sont la présence de ces variables pour 12 mois différents sur 14 années consécutives. On s'apercoit alors très vite qu'elles sont très liées : il semble nécessaire de réaliser une **réduction de dimension**.

2.1 Corrélation des variables

Au vu du grand nombre de variables que propose le tableau de données, il semble intéressant d'observer la matrice de corrélation qu'on peut visualiser avec la bibliothèque **seaborn**. Elle permet de voir aisément les différents liens entre les variables et par conséquent de les supprimer et éviter la redondance de plusieurs variables. L'établissement d'une liste exhaustive de ces variables paraît dans cette partie du rapport peu utile car celles-ci sont très nombreuses et sont beaucoup plus visibles au sein de la matrice de corrélation **??**[1]. Les recherches que nous avons effectué montrent en effet que toutes les personnes adeptes de ce type de challenge utilisent cette technique afin d'améliorer leur résultat. Nous avons donc dans un premier temps supprimer les variables ayant un fort taux de corrélation tant négativement que positivement. De même, quand un choix entre deux variables s'imposait, on a conservé la plus corrélée avec la variable cible. Le résultat fut saisissant puisque notre score a très fortement augmenté. Cela nous a conduit dans un second temps à retirer à l'aide d'une fonction sur Python toutes les variables ayant un taux de corrélation strictement supérieur à 0.60 en valeur absolue nous permettant ainsi d'affiner nos performances.

2.2 Sélection de variables

Ces mêmes études ont été effectuées en parallèle sur R en mettant en avant les différents critères de pénalité BIC et Cp de Mallows. En effet, la fonction *regsubsets* du package **leaps** en R permet de réaliser une recherche exhaustive des meilleurs sous-ensembles de variables pour prédire la cible sous l'hypothèse que l'on se trouve dans un modèle de régression linéaire (dont les

[1] Voir Annexe 2.

performances ne sont pas si mauvaises comme nous le verrons dans la partie 3). En faisant également appel aux **régressions ridge et lasso** 3.3, on s'est aperçu très vite qu'il fallait retirer beaucoup de variables explicatives.

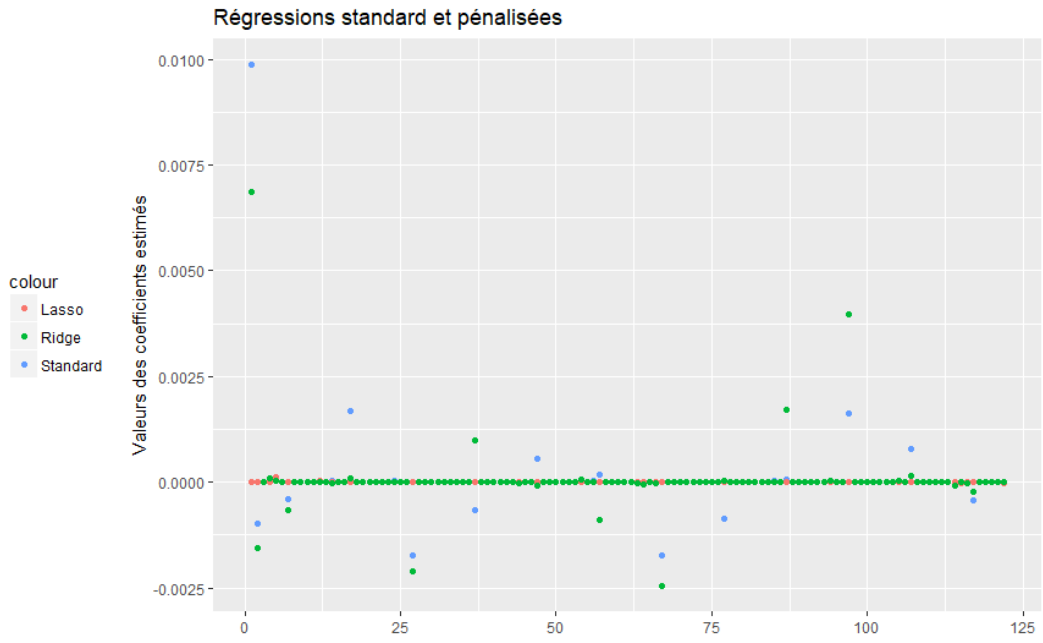

Figure 4: Estimation des coefficients pour différents types de régression.

Cette recherche par tâtonnement confrontée au score obtenu sur le site a permis d'aboutir au choix de 27 variables explicatives 3.3(incluant les deux variables catégorielles *'month'* et *'country'*) dans notre modèle final sur 122 variables au départ.

3 Choix du modèle de prédiction

Une fois le travail sur les données effectué, il nous reste alors à prédire l'évolution de la production de pétrole sur l'échantillon de test à l'aide des différents algorithmes présents dans la bibliothèque **scikit-learn** de Python.

3.1 Validation des modèles

Le nombre de soumissions sur le site Data Challenge étant limité à 6 par jour, nous devons évaluer nous-même la qualité de nos prédictions. Pour cela, on réalise un split train/test de nos données. Celui-ci sépare les données de l'échantillon d'entraînement en deux sous-ensembles : un jeu d'apprentissage X_{train} et un jeu de validation X_{test}. On effectue un apprentissage sur l'échantillon X_{train} et une prédiction sur les données restantes X_{test}. La fonction *train-test-split* de **sklearn** permet de réaliser cet échantillonnage de manière aléatoire. Un bon ratio est le 70/30. Pour s'assurer que notre échantillonnage a été réalisé de manière homogène, on peut faire une visualisation graphique de nos variables. 3.3 Cette méthode très classique nous permet alors d'évaluer le score de nos performances et de valider ou non notre modèle. Évidemment, pour un algorithme donné, ce score de prédiction ne sera pas le même que celui obtenu sur Data Challenge car les données sur lesquelles il est calculé ne sont pas les mêmes. Ce score est toutefois un bon indicateur car il est, la plupart du temps, un peu supérieur à celui du site mais reste très proche.

3.2 Algorithmes

Pour comparer les performances des algorithmes de Machine Learning les plus connus, nous avons créé une fonction *Evaluation*. Cette dernière prend en entrée un dictionnaire de classifieurs et renvoie la moyenne et l'écart-type des performances pour une cross validation à trois ensembles avec comme choix d'erreur : l'erreur **AUC**. On a ainsi testé les modèles suivants :

1. Gradient Boosting
2. Régression linéaire
3. Forêts aléatoires
4. XgBoost/AdaBoost

Les meilleures performances sont obtenues avec l'algorithme du **Gradient Boosting** sans très grande surprise. À présent, il fallait optimiser les paramètres de l'algorithme, autrement dit *tuner* notre algorithme. Pour cela, on a réalisé une *gridsearch aléatoire* avec un espace de paramètres défini comme les distributions des hyperparamètres tout en choisissant un nombre d'itérations maximal. Par exemple, le nombre d'estimateurs d'une forêt aléatoire peut

Figure 5: Classification des résultats des différents modèles

Classifieurs	AUC
Random Forest	0.757
Gradient Boosting	0.812
XgBoost	0.809
Linear Regression	0.798

être compris entre 100 et 10000. Au lieu de chercher une valeur exacte, on peut tirer des nombres dans une loi uniforme discrète entre ces deux valeurs. Une fois l'ordre de grandeur des valeurs des paramètres établi, on effectue cette fois-ci une *gridsearch exhaustive*. Ainsi, on effectue pour chaque n-uplet de paramètres, un calcul des performances du modèle sur une K-CV. On récupère ainsi les paramètres ayant obtenu le meilleur score afin d'obtenir une prédiction optimale.

Grâce à cette optimisation des paramètres, nous avons gagné près de 2% en précision avec un score de 0.833 ce qui nous a permis d'atteindre la 20ème place du classement établi sur 25% des données

3.3 Combinaison de résultats

Pour améliorer notre score, nous nous sommes inspirés du **Bagging** et avons réalisé 200 simulations avec pour chacune d'entre elles une optimisation des paramètres (cf partie 3.2) puis nous avons combiné nos résultats. On a alors modifié les valeurs du fichier avec le meilleur score en regardant si il y avait égalité à 10^{-3} près des vecteurs de probabilité cible que nous avions estimé. Cette technique nous a permis d'assurer la stabilité de nos prédictions. Ici, le classifieur ne changeait pas contrairement au bagging mais cette méthode nous a permis d'obtenir un score de 0.839 afin d'atteindre la 4ème place.

Conclusion

Ce projet sous la forme d'un challenge de données a été extrêmement bénéfique à notre formation. Il nous a permis de nous former rapidement et de renforcer nos acquis en programmation. Avec les languages Python et R, nous avons mis en pratique les différents algorithmes de machine learning à l'aide des packages et bibliothèques dédiées.

À travers ce sujet, nous avons été exposés à des problématiques concrètes telles que la prédiction binaire et le grand nombre de variables explicatives. Nous avons réussi à nous adapter à cette difficulté en faisant appel notamment aux notions apprises en cours (sélection de variables, pénalisations) et à notre bon sens en tant que statisticiens.

L'aspect compétitif de ce type de challenge a également été très stimulant. Il a permis de renforcer notre cohésion d'équipe et a favorisé la communication et l'échange d'informations dans notre trinôme.

A la clôture du data challenge, nous sommes à la 4ème position ce qui nous place dans le top 3%. Par ailleurs, le classement final établi au 1er janvier 2018 par la plateforme *Challenge Data* et portant sur 100% des données prédites nous place à la 12ème position, c'est-à-dire dans le top 8%.

Annexes

Annexe 1

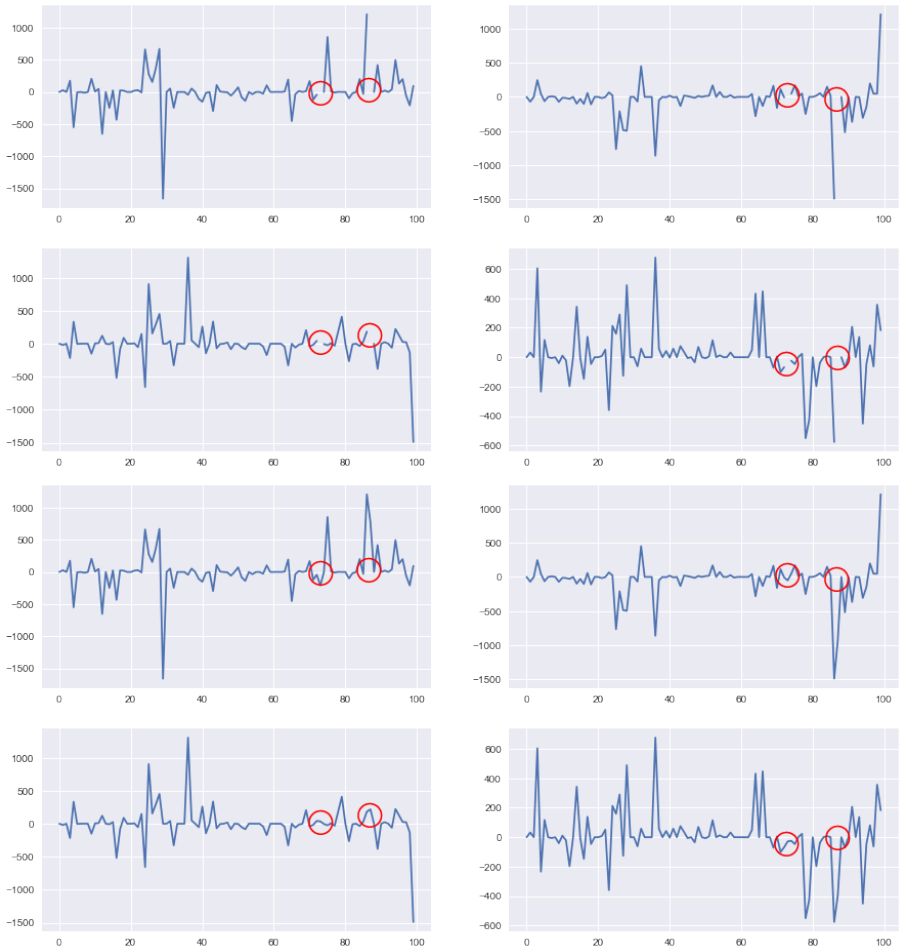

Figure 6: Traitement des valeurs manquantes pour les variables xdiffClosing stocks(kmt) avec x $\in \{1, 2, 3, 4\}$.

Annexe 2

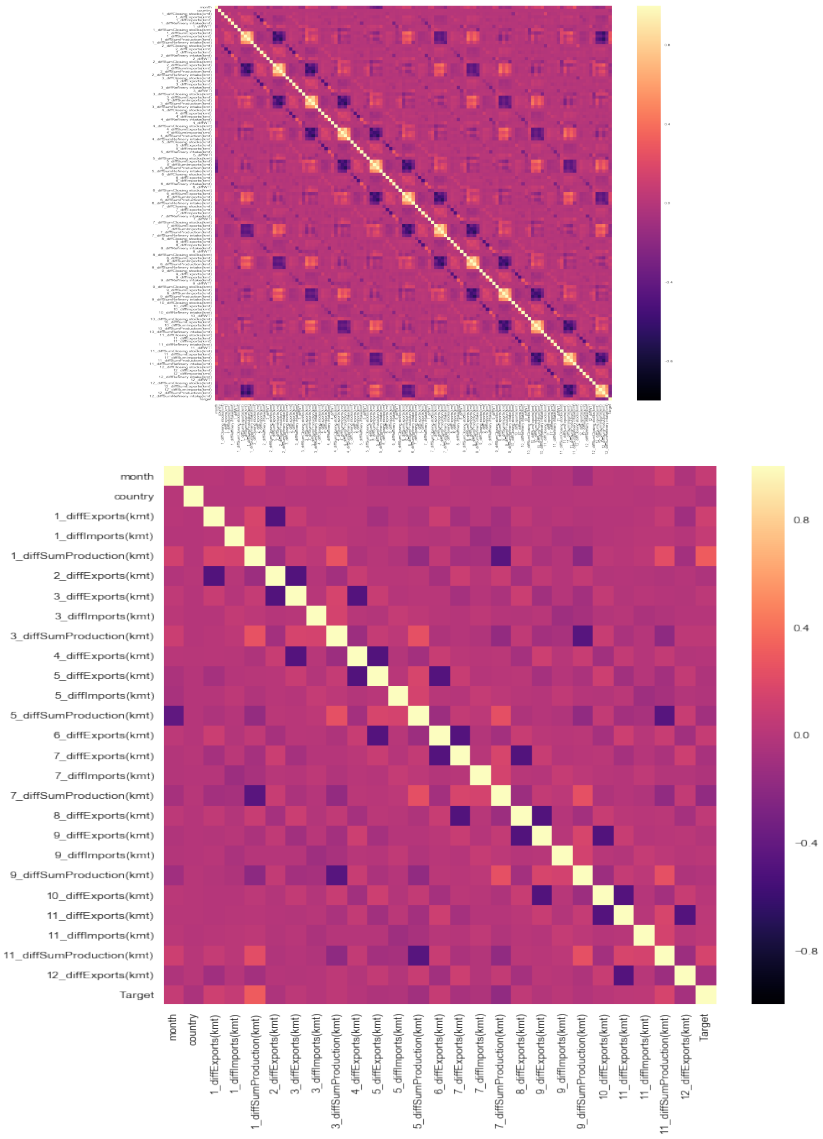

Figure 7: Représentation des matrices de corrélation : avec toutes les variables (en haut), avec les variables ayant un taux de corrélation compris entre $]-0.60\ ;\ 0.60[$ (en bas).

Annexe 3

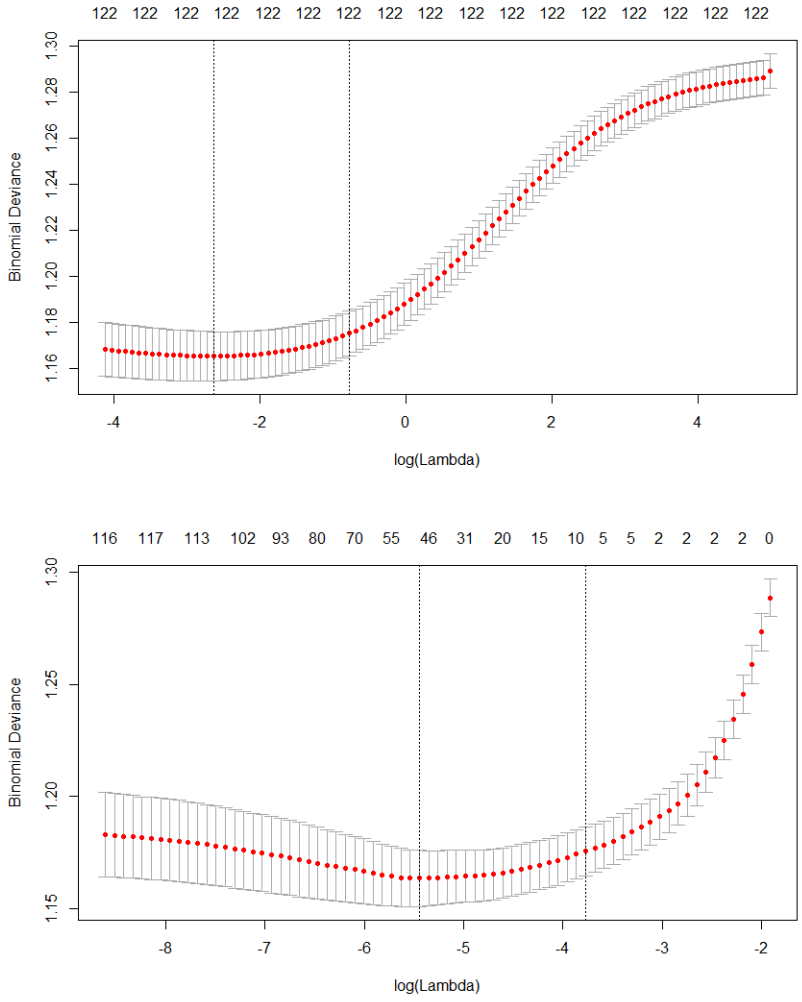

Figure 8: Régressions ridge (en haut) et lasso (en bas). Le point minimal de la courbe indique le lambda optimal.

Annexe 4

Variables explicatives	Type	Explications brèves
country	Catégoriel	Indice des pays producteurs
month	Catégoriel	Indice des mois
1diffExports(kmt)	Numérique	Exportations
1diffImports(kmt)	Numérique	Importations
1diffSumProduction(kmt)	Numérique	Production totale en milliers de tonnes
2diffExports(kmt)	Numérique	Exportations
3diffExports(kmt)	Numérique	Exportations
3diffImports(kmt)	Numérique	Importations
4diffExports(kmt)	Numérique	Exportations
5diffExports(kmt)	Numérique	Exportations
5diffImports(kmt)	Numérique	Importations
5diffSumProduction(kmt))	Numérique	Production totale en milliers de tonnes
6diffExports(kmt)	Numérique	Exportations
7diffExports(kmt)	Numérique	Exportations
7diffImports(kmt)	Numérique	Importations
7diffSumProduction(kmt)	Numérique	Production totale en milliers de tonnes
8diffExports(kmt)	Numérique	Exportations
9diffExports(kmt)	Numérique	Exportations
9diffImports(kmt)	Numérique	Importations
9diffSumProduction(kmt)	Numérique	Production totale en milliers de tonnes
10diffExports(kmt)	Numérique	Exportations
11diffExports(kmt)	Numérique	Exportations
11diffImports(kmt)	Numérique	Importations
11diffSumProduction(kmt)	Numérique	Production totale en milliers de tonnes
12diffExports(kmt)	Numérique	Exportations

Figure 9: Tableau des variables explicatives retenues après la sélection de variable.

Annexe 5

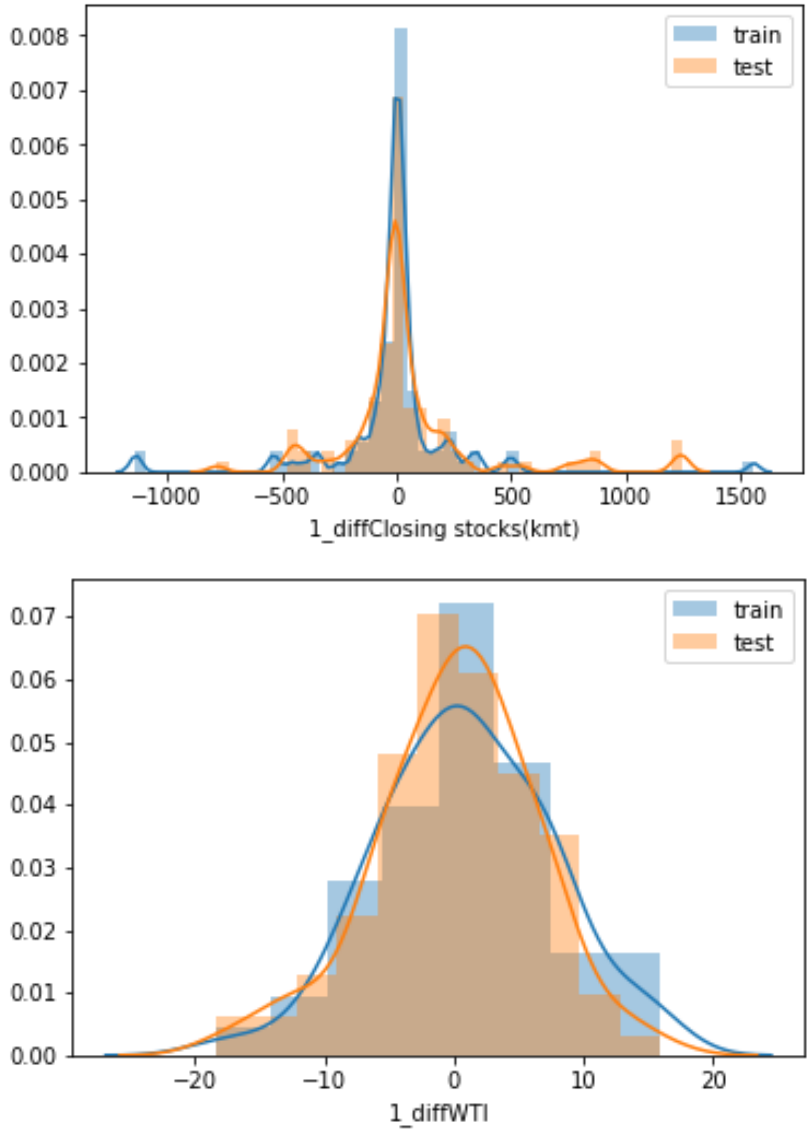

Figure 10: Histogramme de deux variables explicatives pour les jeux de données test et train. Notre choix d'échantillonnage est bien homogène, on peut l'utiliser pour appliquer les algorithmes de Machine Learning.