Projet de machine learning

Alexis Buckens

July 16, 2017

1 Introduction

1.1 Présentation du projet

Le but du projet est d'évaluer les deux variables "heating" et "cooling" en fonction d'une série de variables. Pour ce faire, plusieurs méthodes peuvent être mises en œuvre, de la plus stable à la plus flexible. Deux problèmes se posent afin de réaliser ce projet. Le premier problème est de sélectionner les variables les plus pertinentes. Il est naturellement possible d'élaborer un modèle en utilisant l'ensemble des variables, mais cette approche peut poser des problèmes de multicollinéarité et peut nous pousser à utiliser des variables qui ne sont pas liées aux variables dépendantes, ce qui peut fausser les prédictions dans le cas de certains modèles tels que la régression linéaire par les moindres carrés. Une première approche afin de sélectionner les variables pertinentes consiste à préalablement appliquer une Analyse en composante principale, et à n'utiliser que les premières dimensions résumant l'essentiel de l'information contenue dans les variables. Une seconde approche consiste à appliquer une méthode de régularisation telle que le LASSO qui sélectionne automatiquement les variables en ajoutant une pénalité à la fonction à minimiser dans la régression linéaire.

Le second problème consiste à déterminer quel modèle utiliser et quelle valeur donner aux meta-paramètres des différents modèles utilisés. Le choix sera fait sur base de la RMSE, qui sera caclulée par 10-fold cross-validation (5 fold dans le cas des réseaux de neurones afin de limiter le temps de calculs), et en répétant éventuellement le processus plusieurs fois afin d'obtenir la meilleure estimation possible. Le modèle sélectionné sera celui qui nous permettra d'obtenir la RMSE la plus basse avec les paramètres choisis.

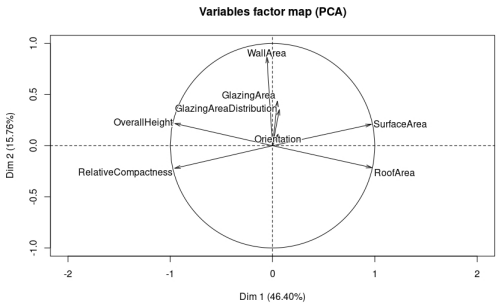
Les différents modèles que nous utiliserons et leurs paramètres seront détaillés plus précisément dans la section suivante. Nous tenterons de prédire les deux variables en utilisant une régression linéaire par moindres carrés, une régression linéaire avec régularisation (ElasticNet), une knn-regression, un multi-layer perceptron et une régression par composantes principales.

1.2 Préparation des données

La première étape consiste à préparer les données pour leur analyse ultérieure, et à jeter un premier coup d'œuil sur celles-ci afin de mieux les comprendre.

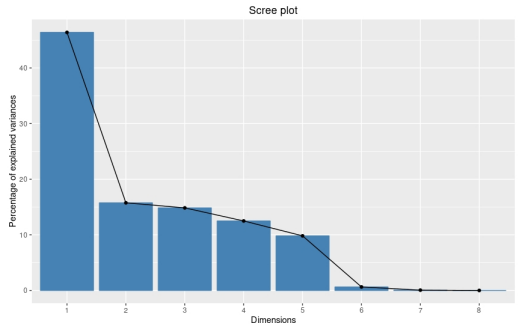
On remarque immédiatement que les données semblent prendre un nombre limité de valeurs différentes. Puisque les valeurs des variables ont du sens en tant que valeurs continues, on peut les considérer comme telles mais il sera intéressant de les analyser ultérieurement comme des variables discrètes et de voir ce qui change dans notre analyse.

Une première façon d'obtenir une vue générale sur nos variables consiste à effectuer une analyse des composantes principales. Celle-ci nous permet de voir où réside l'information parmi les variables, et comment limiter celles-ci le cas échéant. L'analyse nous permet d'obtenir le graphique suivant :



On observe sur ce graphique que d'une part "WallArea", "GlazingArea", "GlazingAreaDistribution" et "Orientation" forment un premier groupe de variables contribuant principalement à la seconde dimension, tandis que les 4 autres variables forment un second groupe contribuant essentiellement à la première dimension.

Si l'on s'intéresse cette fois aux valeurs propres, à la proportion de la variance expliquée, et au nombre de dimensions nécessaires afin de comprendre nos données, on peut rapidement constater que les 5 premières composantes permettent d'expliquer 99% de la variance totale. Le graphique suivant permet d'observer visuellement en un coup d'œuil, la proportion de la variance expliquée par chacune des composantes.



2 Modèles utilisés

2.1 Première tentative de prédiction : modèles linéaires

Une première approche qui semble la plus simple serait de prédire les valeurs de heating et cooling par une régression lineaire simple en utilisant toutes les variables. Naturellement cette approche pose de nombreux problèmes : certaines variables risquent d'etre corrélées (ce qui est par ailleurs confirmé par l'ACP de la section précédente), les unités différentes peuvent fausser les prédictions et le grand nombre de variables prises en compte augmente le risque de surapprentissage. On voit par ailleurs que lorsqu'on essaye de calculer ce modèle linéaire simple, R renvoie un message d'erreur lié précisément à ce genre de problème. La fonction "findLinearCombos" du package "Caret" nous permet d'indentifier si une colonne est une combinaison linéaire d'une ou plusieurs autres colonnes. En l'occurence, la fonction nous retourne en effet un résultat pour les colonnes 4, 2, et 3. On a donc probablement en outre un problème de multicollinéarité. Une régression linéaire simple n'est donc probablement pas une bonne approche pour résoudre ce problème.

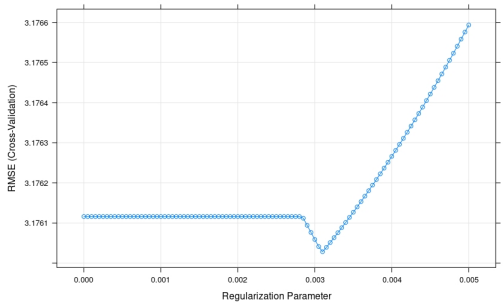
Le premier problème à résoudre est celui des échelles et des unités. Ce problème peut être facilement résolu en centrant et standardisant les données, par exemple en utilisant "PreProcess"

de la fonction "train". Le problème de multicollinéarité peut être résolu en diminuant le nombre de dimensions utilisées dans la prédiction. Dans le pré-processing, on peut donc inclure une analyse en composantes principales ou une analyse en composantes indépendantes, et n'utiliser qu'un nombre plus limité de composantes sans perdre trop d'informations. Cette méthode nous donne un RMSE de 7.01 si on essaye de prédire les deux variables à la fois, et de respectivement 3.45 et 3.84 pour la prédiction de heating et de cooling séparément. R ne renvoie cette fois plus aucun message d'erreur.

2.2 Seconde tentative : GLMNET

Une seconde méthode est d'utiliser des méthodes de régularisation. On peut par exemple limiter le nombre de variables et l'influence de certaines d'entre elles en imposant une pénalité sur les paramètres lors de l'élaboration du modèle. Deux modèles permettent d'atteindre un tel résultat : ridge regression et lasso. Là où le premier limite l'influence de certaines variable en réduisant la valeur des paramètres, le second réduit la valeur de certains à 0, limitant ainsi le nombre de variables prises en compte. Un modèle permet d'utiliser ces deux types de pénalités à la fois : l'Elastic net qui est présent dans R via la fonction glmnet. Ce modele, en plus du paramètre lamdba sur la "taille" de la pénalité, possède un second paramètre, alpha, qui indique en quelque sorte la proportion de chaque type de pénalité. La fonction que cherche a minimiser glmnet est $\min(\frac{1}{N}\sum(w_i l(y_i, \beta_0 + \beta^T x_i) + \lambda[\frac{(1-\alpha)\|\beta\|_2^2}{2} + \alpha\|\beta\|_1]$. Cette expression nous permet de voir simplement comment les deux facteurs interagissent. Cette methode nous permet ainsi d'éviter le surapprentissage. Glmnet ne nous permet malheureusement pas de prédire deux variables dépendantes ensemble, et il faudra donc construire un modèle pour chacune d'entre elles, et comparer ces modèles avec un modèle linéaire ne prédisant qu'une seule variable à la fois.

On peut par exemple utiliser la fonction train pour calculer la RMSE en utilisant une cross-validation en 10 parties en faisant varier les paramètres sur 15 valeurs différentes et en répétant la procédure 15 fois. Les valeurs de lambda retournées par caret sont excessivement faibles et il pourrait sembler a priori que le modèle n'est pas réellement plus performant qu'une régression linéaire simple. Néanmoins, l'erreur a diminué pour la prédiction des deux variables! Une observation plus détaillée de l'évolution de l'erreur en fonction de lambda pour la variable "cooling" indique que celle-ci diminue à un certain point avant de réaugmenter brusquement, comme sur le graphique suivant :



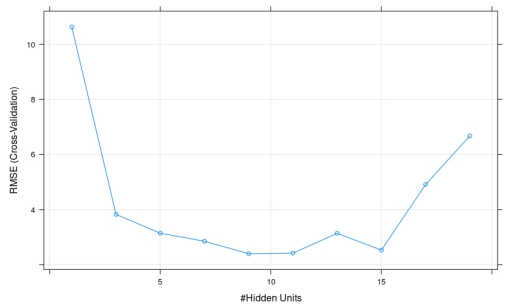
2.3 Troisième tentative : KNN regression

On pourrait également essayer de prédire le modèle en utilisant des méthodes non-paramétriques, telle que la régression en utilisant les k voisins les plus proches. Cette méthode est plus flexible mais moins stable que la régression linéaire. Le seul meta-paramètre est ici le nombre k de voisins choisis pour estimer la valeur d'une nouvelle donnée. La valeur choisie automatiquement par Caret comme minimisant la RMSE est k=7 pour heating comme pour cooling. L'erreur de

ce modèle est étonnamment basse. Comme "knn" est une fonction destinée en théorie uniquement aux problèmes de classification, on peut vérifier les valeurs obtenues avec Caret avec la fonction "knn.reg", destinée plus spécifiquement à la régression mais qui n'est pas implémentée au sein de caret. Une rapide observation des résultats de ce modèle montre que ceux-ci restent proches des valeurs estimées par Caret. On a donc ici un modèle plus performant dans ce cas que la régression linéaire.

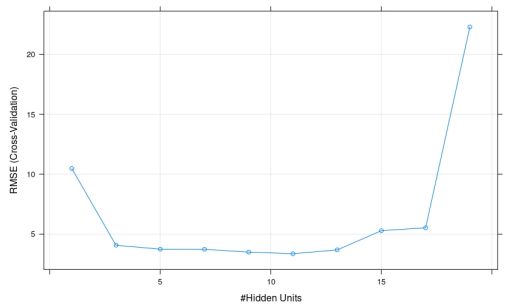
2.4 Quatrième tentative : Multilayer perceptron

Une autre méthode que l'on peut utiliser est le multilayer perceptron. La fontion d'activation par défaut de la fonction "mlp" dans R est une fonction d'activation logistique. On verra ultérieurement si une fonction linéaire est plus ou moins efficace. On commencera par utiliser un seul layer. Pour estimer combien de neurones utiliser dans celui-ci, on choisira une fois de plus la valeur qui minimise la RMSE avec train du package caret. Pour la variable heating par exemple, l'erreur en fonction du nombre de perceptrons varie de la façon suivante :



On voit ici que pour prédire la variable "heating", le modèle le plus performant est celui qui utilise 10 perceptrons.

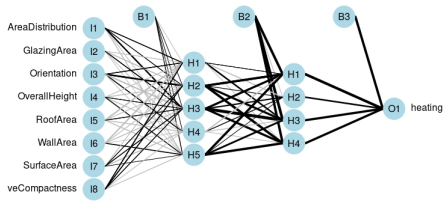
Dans le cas de la variable "cooling", le graphique prend une forme légèrement différente :



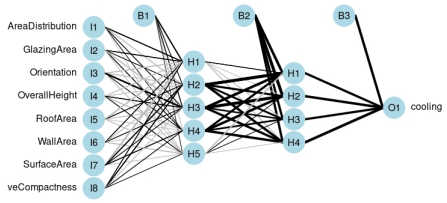
La valeur choisie par Caret est de 11, mais on peut voir sur le graphique que celle-ci n'est pas beaucoup plus performante qu'un modèle n'utilisant que 5 neurones, qui a l'avantage d'être nettement plus parcimonieux. Le modèle est assez performant pour heating, l'erreur n'étant que de 2.39, mais ne l'est pas tant pour la prédiction de la variable cooling, pour laquelle l'erreur est de 3.36, ce qui est à peine meilleur que la régression linéaire!

Si on remplace la fonction d'activation logistique par une fonction d'activation linéaire, on obtient une erreur considérablement plus élevée ; il ne s'agit donc probablement pas d'une bonne idée!

On peut également essayer de construire un modèle légèrement plus complexe, possédant plus d'un hidden layer. Si on essaye de trouver le modèle minimisant la RMSE pour la prédiction des deux variables ensemble en faisant varier le nombre de neurones sur le premier hidden layer entre 2 et 6, ceux sur le second entre 1 et 4 et en fixant ceux sur le troisième layer à 0 pour raccourcir le temps de calcul, et en utilisant une fonction d'activation logistique, on obtient le modèle suivant pour heating :



Et celui-ci pour cooling :



L'erreur de ces deux modèles est assez petite : 1.8 et 0.5. Il s'agit donc des modèles qui semblent les plus performants pour le moment.

2.5 Cinquieme tentative : PCR

On peut aussi élaborer un modèle de régression à partir des composantes principales. Contrairement au premier modèle dans lequel on élaborait des nouvelles variables à partir des composantes principales en effectuant alors une régression sur celle-ci par moindres carrés, on va ici estimer les coefficients du modèle directement à partir des résultats de l'ACP. Cette méthode nous permet aussi de sélectionner indirectement les variables les plus pertinentes. On utilisera également la méthode des moindres carrés partiels qui est proche de la méthode précédente mais qui possède l'avantage d'également projeter les variables dépendantes. Ces deux méthodes peuvent se révéler intéressantes puisque l'on a un nombre relativement élevé de variables et potentiellement des problèmes de collinéarité.

2.6 Sixième tentative : SVR

Le dernier modèle utilisé est le support vector regression, implémenté dans R via "svm". Le fonction n'est pas implémentée dans Caret, mais possède sa propre fonction qui permet également de faire varier les meta-parametres et de sélectionner ceux qui minimisent la RMSE via une cross-validation en 10 parties. Les deux paramètres sont la largeur de la marge, gamma, et la

tolérance à l'erreur, cost. Le modèle utilisera un radial basis function kernel. Les paramètres sélectionnés pour "cooling" sont gamma = 0.08, et cost = 200, et pour "heating" : 0.035 et 1860. L'erreur calculée est respectivement de 3.02 et 0.88. La fonction "tuneResult" possède également un autre avantage : elle permet facilement de tracer un graphique de l'évolution de l'erreur en fonction des deux paramètres. En "zoomant" sur la zone dans laquelle l'erreur est la plus faible, on peut obtenir ces deux graphiques pour la variable cooling et heating, en représentant les zones avec l'erreur la plus basse en bleu plus fonçé :

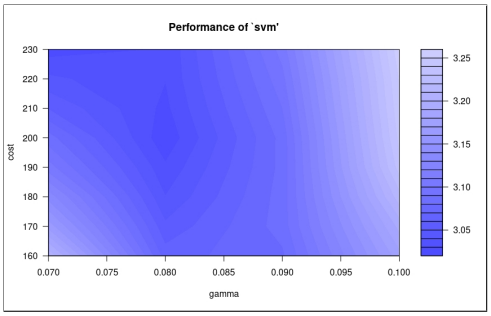


Figure 1: Erreur de la variable "cooling"

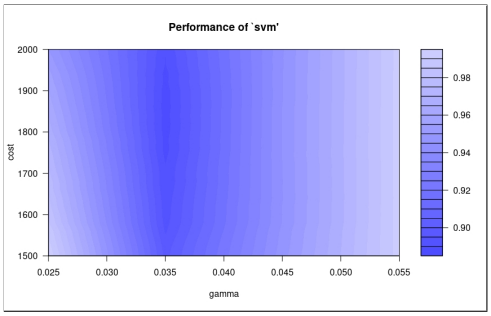


Figure 2: Erreur de la variable "heating"

3 Résultats

Il s'agit désormais de sélectionner le modèle le plus performant sur base de la RMSE. Les valeurs minimales obtenues pour les différents modèles sont présentées dans les table suivantes. La premières table présente les valeurs obtenues pour la variable cooling, et la seconde présente les valeurs pour la variable heating.

Modèle utilisé pour "cooling"	RMSE
Moindres carrés	3.8340
ElasticNet	2.8789
KNN	2.8442
MLP, one hidden layer	3.3641
MLP, two hidden layer	1.8056
PCR	3.1854
PLS	3.1746
SVR	3.0225

Modèle utilisé pour "heating"	RMSE
Moindres carrés	3.4958
ElasticNet	2.8647
KNN	2.4247
MLP, one hidden layer	2.3979
MLP, two hidden layer	0.5175
PCR	2.8687
PLS	2.8708
SVR	0.8873

On peut voir que le modèle le plus performant est bien le multi-layer perceptron pour chacune des deux variables, suivi de près par la SVR.

3.1 Conclusions

Finalement, bien que considérer les variables comme continues ait du sens, il aurait également été possible de les considérer comme des variables discrètes, puisqu'elles ne prennent qu'un nombre limité de valeurs. Nénmoins, cette autre approche pose certains problèmes. Premièrement, d'un point de vue pratique, certains des modèles utilisés n'auraient pas pu l'être avec une variable catégorielle. Deuxièmement, certains modèles comme la régression linéaire bâtissent une série de variables factices afin de pouvoir estimer la régression, augmentant ainsi le nombre de dimensions et le risque de collinéarité. Bien qu'il soit toujours possible de réduire le nombre de dimensions, par exemple via une MCA, il semble néanmoins plus intéressant de ne considérer ces variables que comme des variables continues.

Il aurait pu être intéressant également de regarder comment un mlp à 3 layers prédit notre variable, ou d'élargir la grille des meta-parametres. Malheureusement, une prédiction pour plus de paramètres implique un temps de calcul plus long, qui n'était pas dispobnible. Les prédictions obtenues semblent néanmoins assez proches des valeurs cibles.

Finalement, dû à des problèmes liés à R (la session ne répond plus à chaque usage de la fonction compute), ce sera le second meilleur modèle qui sera utilisé, à savoir la SVR.