**TD2 : optimisation de modele**

Pour ce TD nous reprenons une base classique de prix de l'immobilier. L'objectif est de prédire le prix « SalePrice ».

L'objectif est de sélectionner le meilleur modele et de l'optimiser.

Il s'agit d'une régression notre fonction de performance pourra être le R2. Pour rappel R2 mesure la part de variance expliquée. R2=1 au maximum (modele parfait .. douteux ..) , R2=0 : le modele est aussi frustre qu'une moyenne (on rédit le même prix pour toutes les maisons, et R2<0 : c'est possible .. vous faites moins bien que prédire la moyenne pour toutes les maisons.

La distribution des prix à prédire est étalée sur plusieurs ordres de grandeurs : un bon réflexe est de travailler le log du prix plutôt que le prix : cela évite une trop forte sensibilité aux outliers, « gaussianise » la distribution de prix (faites en l'observation) ainsi que les résudus (très utile pour les régressions linéaires

Le feature engeneering est déjà fait (déjà proposé plutôt .. on peut toujours le poursuivre mais ce n'est pas notre objectif aujourd'hui

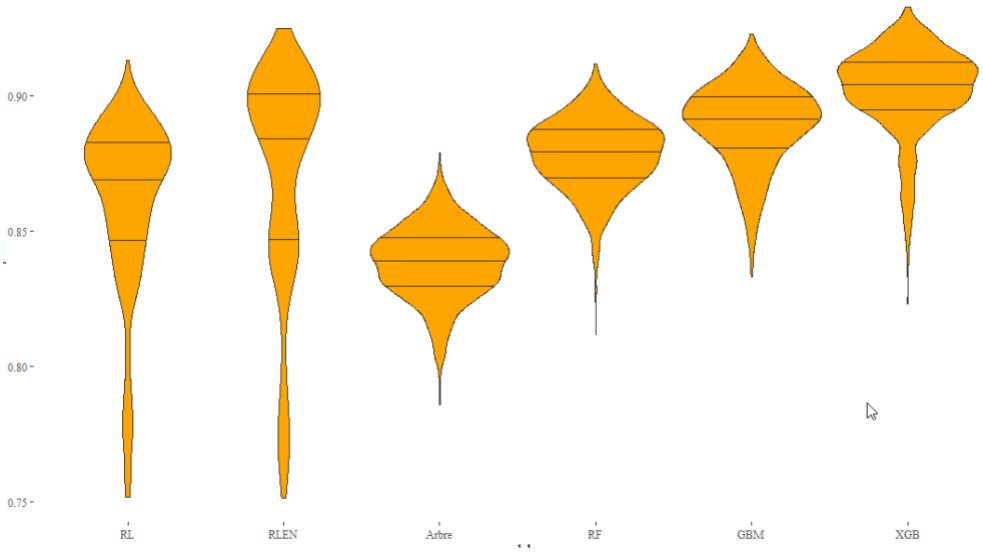
Ce que nous devons faire :

1) **Sélectionner un bon modele (le meilleur .. serait prétentieux)**

Ce travail est déjà fait : 6 modeles ont été réglés (choix des hyperparametres) et testés 1000 fois chacun et le graphique ci-dessous montre la distribution de performance observée. Il est essentiel de comprendre que le résultat d'un modele est une variable aléatoire. Le tester une fois et prendre la performance obtenue pour argent comptant est une erreur lourde

Modeles :

- RL : régression linéaire (ce n'est pas du machine learning car pas d'hyperparamètres et aucun réglage d'overfitting)
- RLEN : Elastic Net (RL pénalisé .. ca c'est de l'apprentissage automatique !)
- Arbre : avec réglage de pénalisation
- RF : random Forest
- GBM : gradient boosting
- XGB : Extreme Gradient Boosting




**A la vue de ce graphique : quel modèle sélectionnez-vous ? pourquoi ?**

### 2) Dépouiller le modèle : quelles sont les informations importantes ?

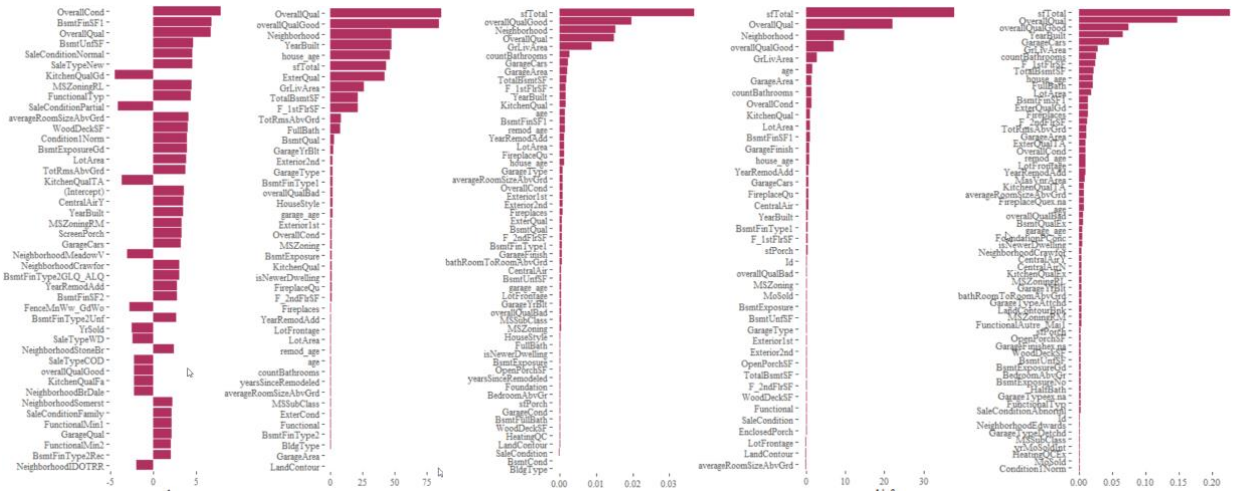
A quoi sert une prédiction ? à prédire l'avenir ou à le changer ? en deep learning il s'agit souvent de prédire le présent (c'est un chat ou un chien sur l'image ?), en entreprise il s'agit souvent d'anticiper le futur …pour le changer.

➔ L'interprétation et l'appropriation des modèles par les opérationnels est essentiel. En outre ce travail est essentiel pour optimiser le modèle. Le feature engineering n'est jamais terminé, identifier les variables les plus utilisées oriente l'effort. Chaque modèle apporte des informations utile

Accès aux features importances (package et commande)

- (Régression linéaire) : summary(mod.lm)$coefficients
- (Arbre) rpart : mod.arbre$variable.importance
- randomForest : mod.rF$importance
- gbm : summary.gbm(mod.gbm)
- Xgboost : xgb.importance(model=mod.xgb)

Rég lin (tvalue) Arbre Random Forest GBM XGBoost



Dans la vie réelle vous devez développer une vraie familiarité avec toutes les variables importantes : que signifient elles, comment l'information est collectée ? puis explorer si un feature engineering complémentaire est possible pour qu'elle s'exprime mieux.

Le package lime est à explorer aussi pour identifier les combinaisons de features importantes, extraire une importance locale (sur une prédiction spécifique, quelles sont les données qui font pencher la balance d'un côté ou de l'autre

### 3) Optimiser le modèle

Nous allons nous focaliser sur le modèle xgboost et l'optimiser : le rendre le plus sobre possible

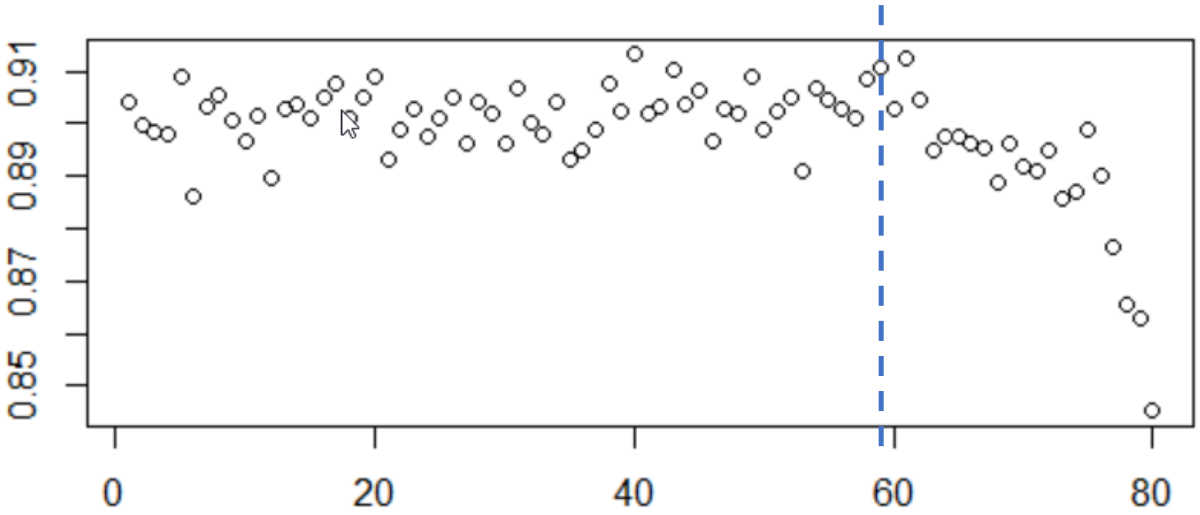
2 intérêts

- Moins de temps de calcul
- Meilleure stabilité

Le meilleur modèle optimisé n'est pas nécessairement celui qui réuni les features importantes.

Itération, suppression des 3 dernières features les moins importantes  chaque cycle. Chaque cycle 12 modélisations pour calculer une moyenne de performance

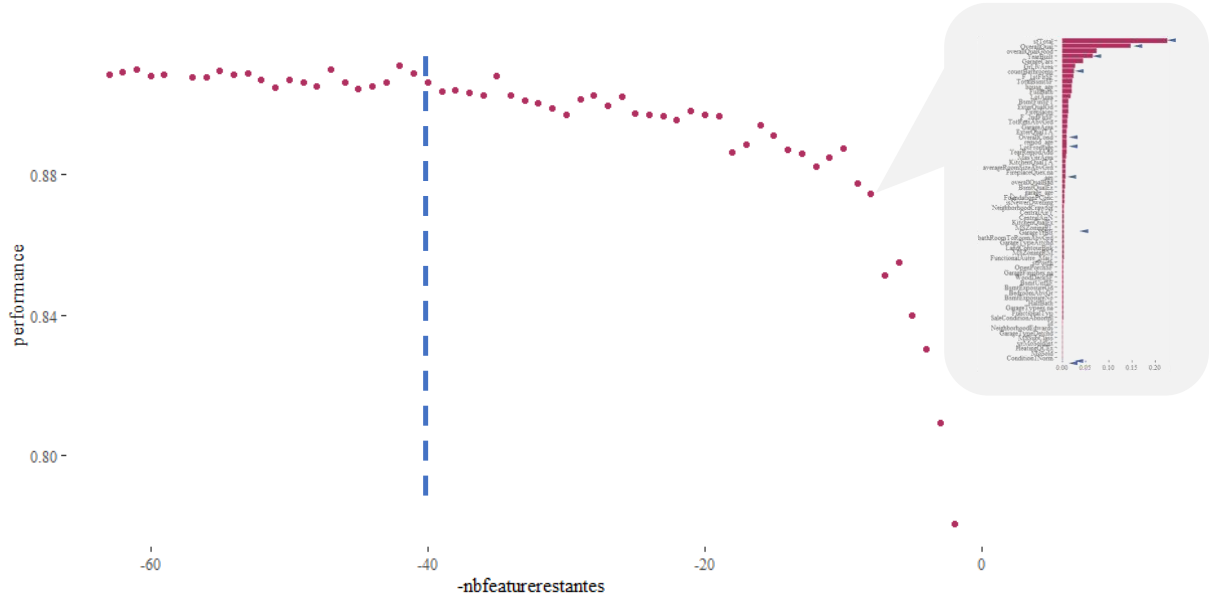
Ordonnée : R2 / abscisse : itérations (3 features supprimées à chaque itération)




Le modèle est passé de 250 features à 72, sans perte de performance.

Peut on aller plus loin ? peut on supprimer des features avec une importance non nulle sans réduire la performance du modele ?

Le travail d'allègement du modèle peut continuer de manière plus fine : à chaque itération tester la suppression une à une de chaque features, plusieurs fois et supprimer celle qui en moyenne conduit à la meilleure performance. Le process continue jusqu'à ce qu'il n'y ait plus de features. En ordonnée : R2 moyen sur 10 tentatives par features



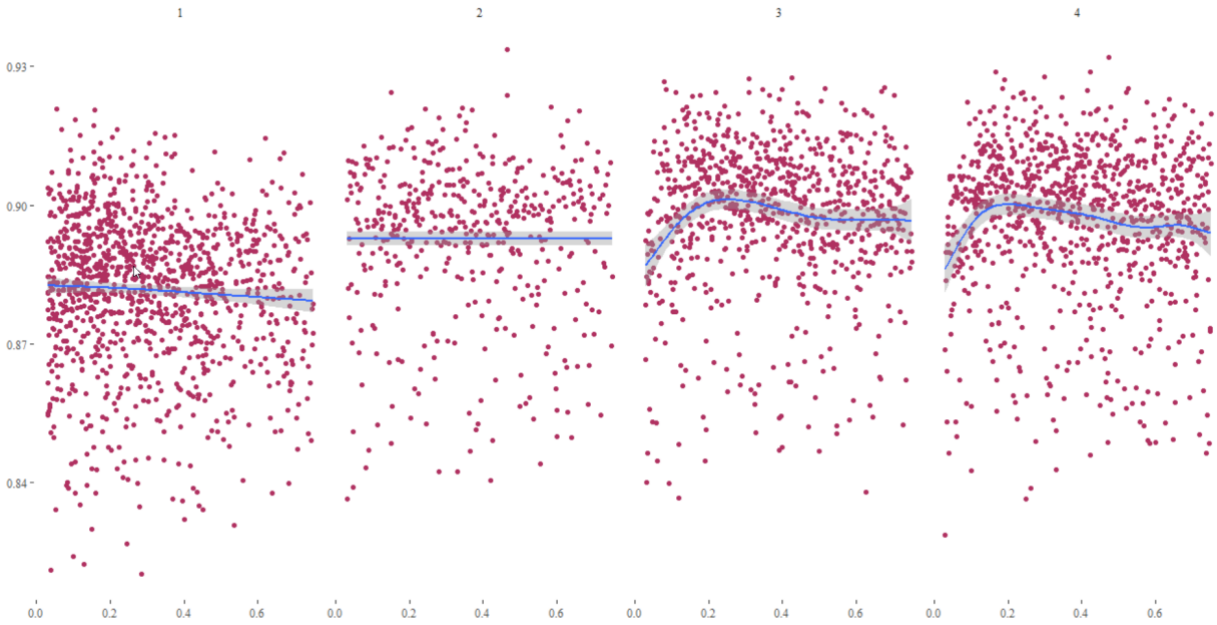

Notez dans la bulle à droite que les 10 features finales (triangle bleu) ne sont pas les 10 features avec le plus d'importance initialement (barres rouges).

On peut régler le modèle avec une quarantaine de features.

Notez aussi un R2 de 75% avec seulement 2 features !

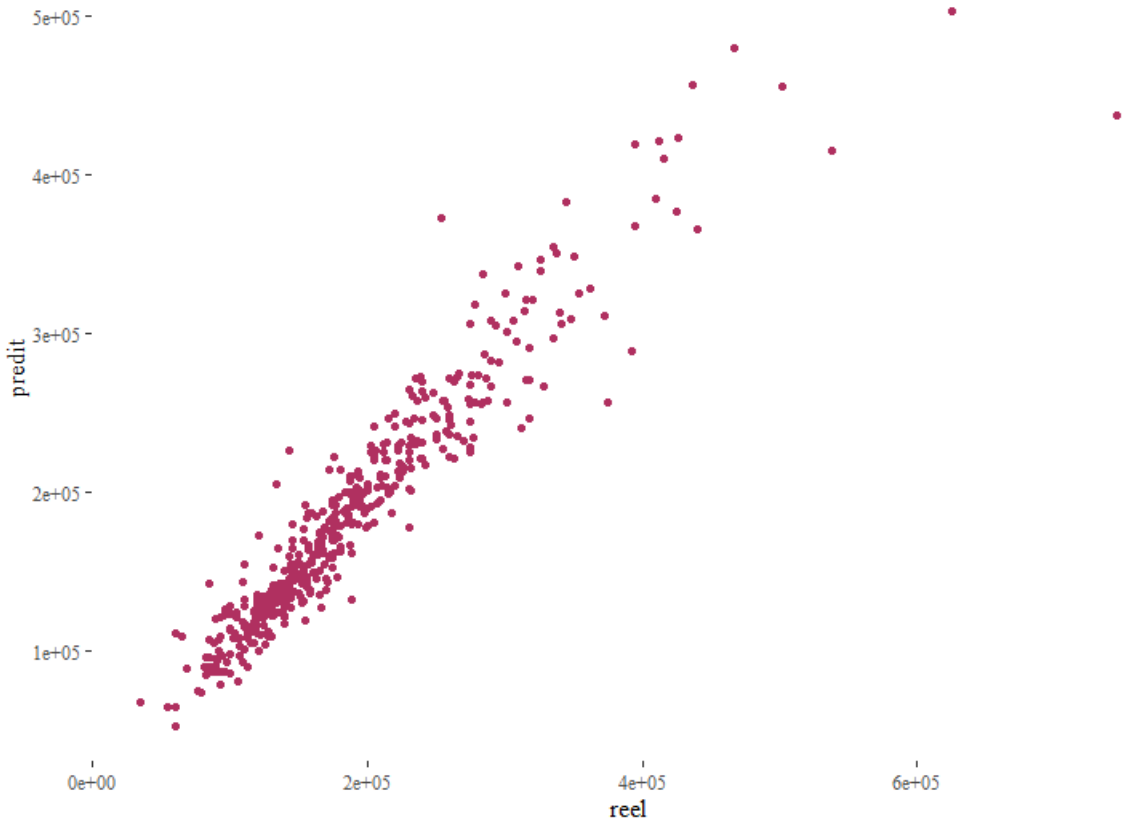
Il est ensuite envisageable de rerégler le modele sur ce jeu de données ajusté

Graphes : maxdepth de 1 à 4, an abscisse % de colonnes pris en compte à chaque étape, ordonnées R2,




Retour à la réalité

Scatter en brut


5e+05
4e+05
3e+05
2e+05
1e+05
predit
0e+00
2e+05
4e+05
6e+05
reel