Case Study Data Science
Graduate Program Axa 2020

Alioune Badara Ba GAHN

Sommaire

Contexte et Problématique

- Ce case study se place dans le contexte de candidature au Graduate Program d'Axa
- Il est aujourd'hui indispensable d'utiliser la Data science dans le milieu de l'assurance
- Les données sont massives et il ya un besoin d'en tirer de la valeur
- D'où la nécessité de Challenger nos compétences techniques en Data science.

- Base de données portant sur les assurances automobiles
- Prédire le bénéfice net annuel à partir de plusieurs variables
- Problème d'apprentissage supervisé
- Variable à expliquer continue: problème de régression

Démarche de développement

Analyse descriptive globale et Feature Engineering

- Nous effectuons cette phase sous un environnement python
- Nous utilisons Pandas-profiling pour une description du Dataset:

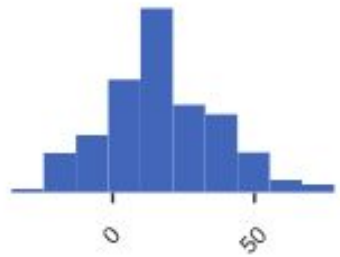
  - 10 variables quantitatives: âge, prime mensuelle, Kms parcourus par mois, Coût entretien, Score CRM, score credit, coefficient bonus malus niveau de vie, salaire annuel.

  - 03 variables factorielles: marque de la voiture, catégorie socioprofessionnelle, type du véhicule

  - 80 valeurs manquantes dans le train

- Vu la composition de notre dataset, nous pensons que les méthodes de machine learning marcheront mieux (se généralisent mieux) que les méthodes deep-learning qui ont tendance à facilement over-fitter sur ce genre de problème.

Mean	16.9758436
Minimum	-35.7208309
Maximum	78.11467791
Zeros	0
Zeros (%)	0
Memory size	8128



Stats descriptives sur la variable à prédire

Démarche de développement

Analyse descriptive globale et Feature Engineering

- Nous transformons nos variables factorielles en quantitatives par labelEncoder()

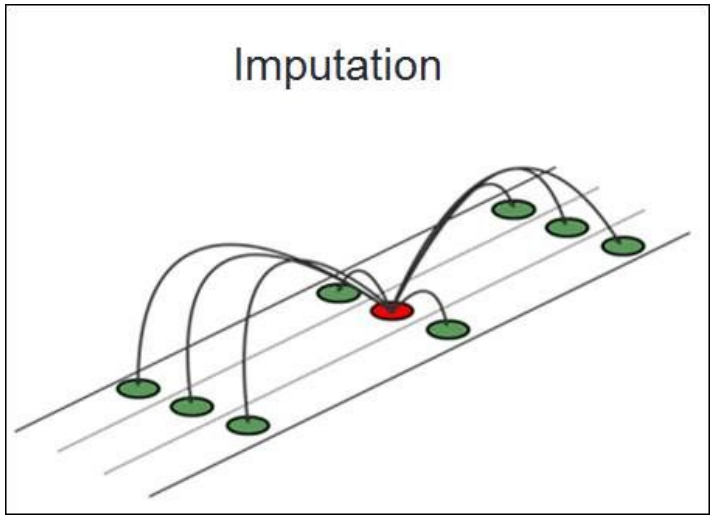
- Nous créons une nouvelle variable age_label en regroupant les âges par intervalles de 5 entre 20 et 90 ans

- On a vu qu'il y a de potentiels outliers dans les données

- Donc pour remplacer les valeurs manquantes, nous ne prenons ni la moyenne ni la médiane.

- Nous faisons une imputation par la méthode du plus proche voisin (1-nearest neighbors).

- Maintenant que nous avons toutes nos données propres, nous passons à la modélisation statistique avancée



Démarche de développement

Modélisation Statistique avancée et Feature Selection

- Dans cette étape, nous travaillons sous un environnement R qui est plus adéquat à des études statistiques poussées.

- L'étude peut se subdiviser en 3 étapes principales:

- Etude des effets des variables qualitatives sur le bénéfice par Anova
- Etude des effets des variables quantitatives par regréssion
- Analyse de covariance sur toutes les variables et feature selection en minimisant l'AIC.

- A la fin de cette étape nous obtenons les features qui expliquent le mieux le bénéfice annuel. C'est ce sur quoi nous allons nous baser pour construire nos modèles.

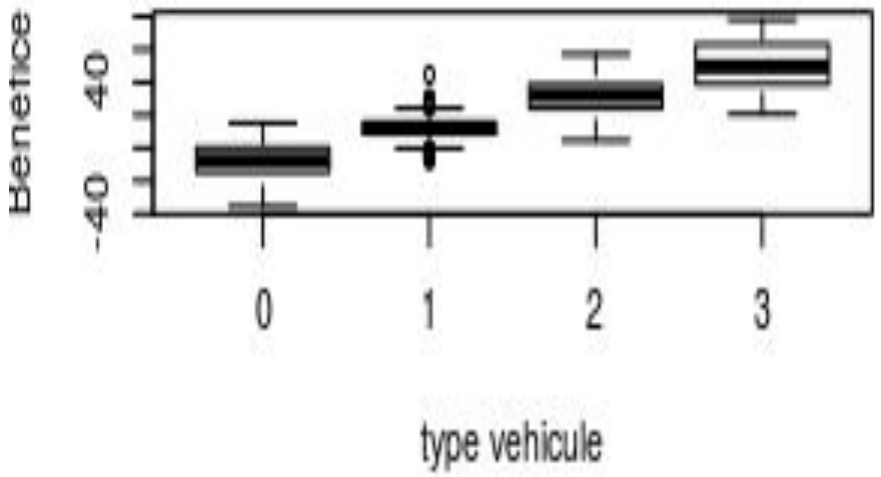
Démarche de développement

Modélisation Statistique avancée et Feature Selection

Etude des variables qualitatives:

Pour étudier les variables qualitatives, nous procédons comme suit:

- Analyse descriptive des variables p/r au bénéfice: Ici nous voyons que le type véhicule est significatif pour expliquer le bénéfice



- Anova sur chaque variable pour tester son effet sur le bénéfice: Ici nous observons une p-value<< 0.05. Donc on conclut un effet sur le bénéfice net annuel

```
lm.tv = lm(Benefice.net.annuel~le_type_vehicule)
anova(lm.tv)
```

```
Analysis of Variance Table

Response: Benefice.net.annuel
                  Df Sum Sq Mean Sq F value    Pr(>F)
le_type_vehicule   3 296709   98903    1221 < 2.2e-16 ***
Residuals        996  80677      81
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

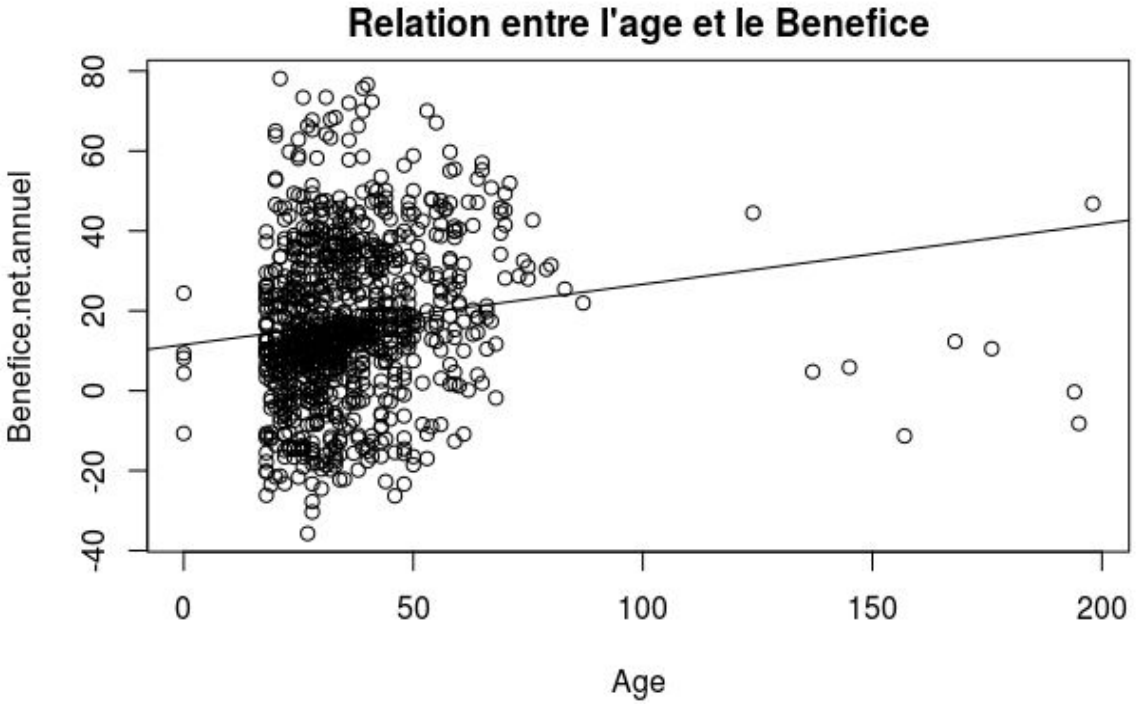
Démarche de développement

Modélisation Statistique avancée et Feature Selection

Etude des variables quantitatives:

L'étude des variables quantitatives se fait aussi suivant le même procédé:

- Analyse de l'effet de la variable: Visuellement nous voyons qu'il ya un effet. Cependant il ya peut etre de potentiels outliers



- Test de significativité de chaque variable par un glm classique: Pour l'âge le test d'effet sur le bénéfice est signifcatif

```
reg.age= lm(Benefice.net.annuel ~ Age)
anova(reg.age) #Nous en déduisons qu'il ya un lien entre l'age et le benefice
```

```
Analysis of Variance Table

Response: Benefice.net.annuel
           Df Sum Sq Mean Sq F value    Pr(>F)
Age         1   7308  7307.5  19.706 1.004e-05 ***
Residuals 998 370078   370.8
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Démarche de développement

Modélisation Statistique avancée et Feature Selection

Analyse de covariance et Feature Selection:

- Maintenant nous avons tous nos features qualitatives et quantitatives qui ont un effet sur le bénéfice
- Nous construisons un modèle d'analyse de covariance avec toutes ces variables
- Nous gardons à la fin le meilleur modèle en procédant à un automatique feature selection avec la méthode descendante.

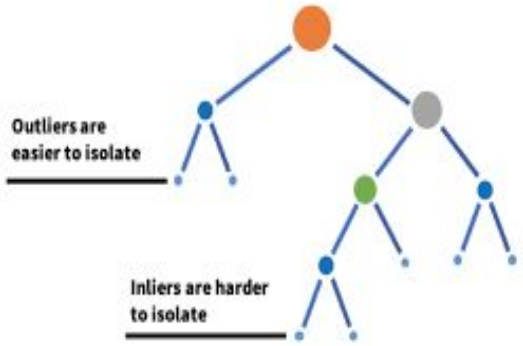
```
Step:  AIC=3812.07
Benefice.net.annuel ~ Kilometres.parcourus.par.mois + Score.CRM +
    Salaire.annuel + le_type_vehicule + le_csp + age_label +
    le_type_vehicule:age_label

                                Df Sum of Sq   RSS    AIC
<none>                                      39968 3812.1
- le_type_vehicule:age_label    36    4035.0 44003 3836.2
- Score.CRM                      1    1633.8 41601 3850.1
- Kilometres.parcourus.par.mois  1    3373.8 43341 3891.1
- le_csp                         4    3908.9 43876 3897.4
- Salaire.annuel                 1   14820.5 54788 4125.5
```

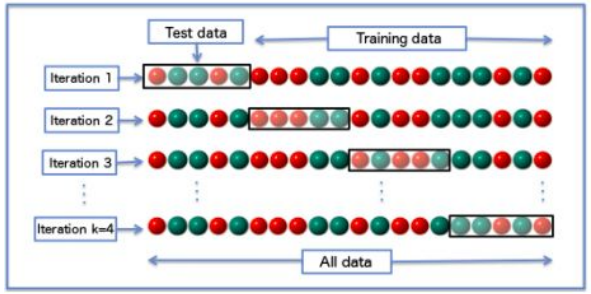
A la fin de cette étape, nous retenons le nombre de km parcourus, le score crm, le salaire annuel, le type véhicule, age_label et la catégorie socio-professionnelle qui minimisent le critère d'AIC

Démarche de développement

Stratégie de Construction des modèles

- Pour construire nos modèles, nous devons prendre en compte qu'il y a de potentiels outliers dans notre dataset
- Notre stratégie pour minimiser le biais des anomalies est la suivante:





- Nous utilisons un Isolation Forest pour détecter les outliers dans notre dataset

- Les fine-tuning du Isolation forest se fait simultanément que l'entraine de nos modèles predictifs:

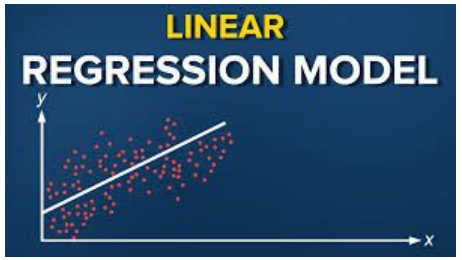
- Pour chaque valeur de l'hyperparamètre dans un range donné;
  - nous répartissons notre dataset en jeu de train et validation (70-30%)
  - nous prédisons des outliers sur notre train set
  - nous filtrons les observations prédites comme outliers de notre base de train
  - nous entrainons notre modèle prédictif sur la base des inliers
  - nous prédisons le bénéfice sur le jeu de validation
  - nous mesurons la rmse de cette prédiction

- Nous choisissons la valeur de l'hyperparamètre qui minimise la rmse comme l'optimale pour l'isolation forest.

- Ainsi, nous trouvons un moyen de réduire le biais qui serait apportait par les anomalies

Modèles utilisés et Résultats

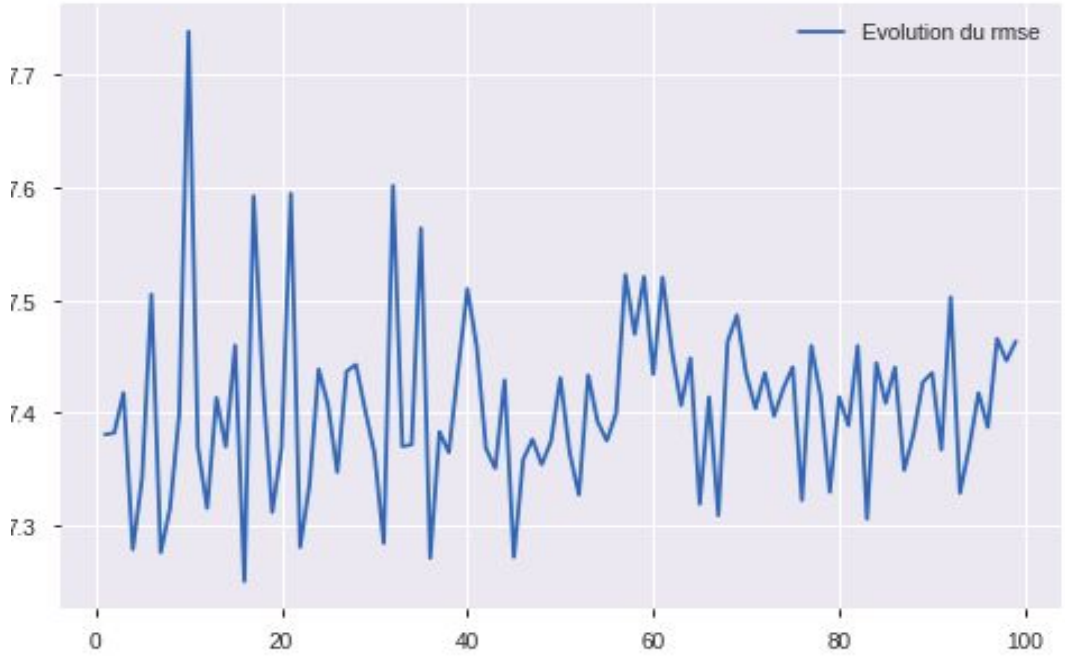
Régression linéaire: Baseline



- Comme Baseline nous utilisons la régression linéaire sachant que nos features sélectionnés expliquent à 89% le bénéfice net annuel.
Nous enrengistrons une rmse de : **7.21**

- Ensuite nous faisons un deuxième modèle en appliquant notre stratégie de détection d'outliers simultanément. On obtient une rmse de **7.25**

Nous remarquons qu'aucun hyper-parametre n'améliore la rmse par rapport à la baseline.
Nous allons donc tester d'autres algorithmes qui vont minimiser ce biais.



Evolution de la rmse en fonction de l'hyper paramètre de l'Isolation Forest

Modèles utilisés et Résultats

Xgboost, LightGbm

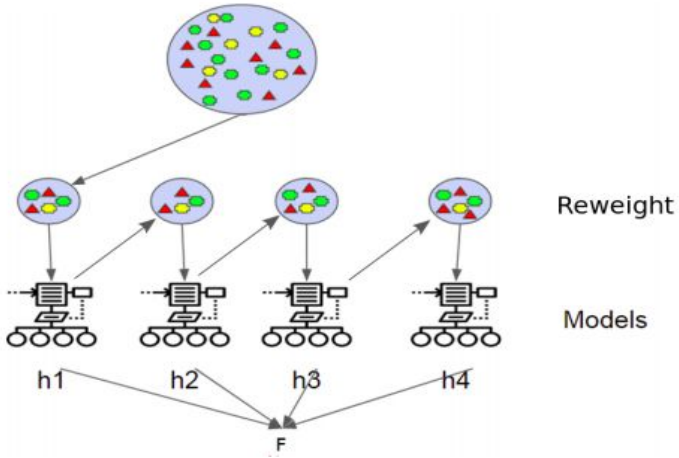
- Nous voyons que la détection d'outliers n'a pas amélioré notre baseline sur la régression

- Nous utilisons donc ces algos qui vont naturellement réduire le terme d'erreur lié au biais de ces potentiels outliers en mettant un grand poids sur les observations mal prédites de notre validation set

- Pour chacune de ces deux algo, nous construisons deux modèles avec les mêmes features renvoyées lors de la phase de modélisation statistique:

  - Un modèle par défaut + cross-validation pour éviter l'overfitting

  - Un modèle pré-calibré à l'aide d'un Gridsearch + crossvalidation



Principe du Boosting

	Score using default tuning	Score using fine tuned method	min score during gridsearch
Xgboost	2.42	2.55	0.98
LighGBM	2.37	2.29	0.70

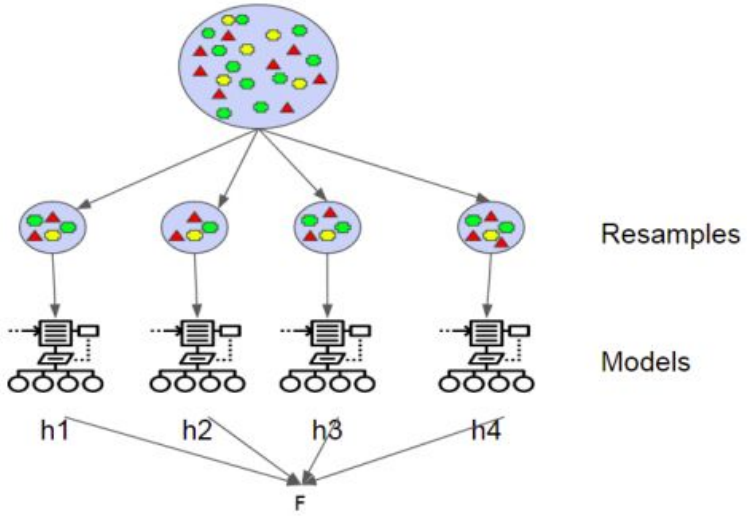
A présent, nous devons réfléchir à une bonne stratégie pour la soumission de nos résultats

Model Selection: Bagging

- Nous avons nos 4 modèles qui réduisent considérablement la rmse par rapport à notre baseline

- Nous devons donc les assembler afin d'obtenir les meilleurs prédictions sur notre test: Bagging

- Pour cela notre stratégie est la suivante:
  - Pour chacun de ces 4 algorithmes nous faisons une prédiction sur le test set
  - Ensuite nous mesurons 2 à 2 les similarités obtenus en comparant les similarités des différents modèles
  - La mesure de similarité utilisée ici est le cosine similarity
  - La tableau ci après montre que les prédictions du Xgboost non fine-tuné sur notre échantillon sont très similaires à celles du lightgbm: Nous prédisons donc la moyenne des prédictions de ces 2 modèles.



Principe du Bagging de modèles

	test_pred_modelf	test_pred_xgb_model	test_pred_model	test_pred_xgb_model_ft
test_pred_modelf	1.000000	0.999271	0.454021	0.455671
test_pred_xgb_model	0.999271	1.000000	0.454219	0.456011
test_pred_model	0.454021	0.454219	1.000000	0.999123
test_pred_xgb_model_ft	0.455671	0.456011	0.999123	1.000000

L'intuition derrière cette stratégie est: de plus on a de modèls qui prédisent les mêmes choses, plus notre travail est généralisable.

Take Away

- Nous avons essayé dans notre démarche de prendre en compte au mieux le contexte métier et de construire un modèle interprétable

- La phase de modélisation Statistique avancée nous a permis d'avoir un modèle généralisable au mieux (avec un R2-ajusté de 0.89)

- Nous avons remplacé nos valeurs manquantes par la méthode du plus proche voisin pour éviter le biais des outliers en remplaçant par la moyenne ou médiane

- Nous n'avons pas modifié les variables qui nous semblent aberrantes car dans les consignes il est dit que c'est un jeu de données fictif et aléatoire. Nous avons donc laissé l'algorithme de détection d'outliers isolation forest s'en charger.

- Nous avons pas testé des méthodes de deep learning car le jeu de données n'est pas assez complexe

- Les méthodes de boosting ont bien marché dans notre situation en réduisant le rmse observée de 7.21 à 0.70 pour le lightGBM et 0.98 pour le Xgboost. Les étapes de cross validation nous ont permis d'éviter au mieux le l'over-fitting.

- En effectuant un bagging (méthode diminuant le terme d'erreur lié à la variance) sur des modèles de boosting (réduction de l'erreur lié au biais du modèle) nous pensons trouver un bon compromis biais-variance.