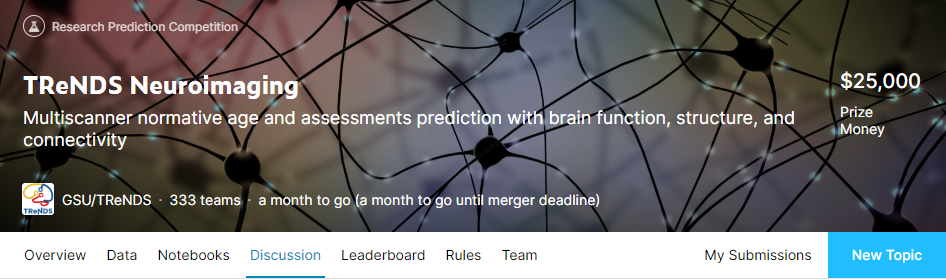
Mehdi DOUBIANI

Openclassrooms

Parcours « Ingénieur Machine Learning »

25 Mai 2020

**Projet 8 IML :** **Compétition KAGGLE :**

**Essai de 3 approches : Algorithme Génétique, Deep Learning, PyCaret**

Ce rapport présente mes travaux sur le projet 8 Openclassrooms du parcours Ingénieur Machine Learning.

Dans le cadre de la participation à une compétition Kaggle, j’ai développé 3 approches différentes pour prédire 5 variables de type numérique : l’âge d’une personne, et 4 autres variables venant de 2 sites (IRM) différents dont les détails de ce qu’elles représentent ne seront dévoilés qu’à la fin de la compétition. **Il s’agit donc d’un problème de régression multi-variables.**

**1ère approche :** L’outil PyCaret

**2ème approche :** Deep Learning pour regression

**3ème approche :** Algorithme Génétique avec la librairie gplearn et la régression symbolique (SymbolicRegressor)

Le kernel Kaggle de l’approche avec l’outil PyCaret est disponible ici :

<https://www.kaggle.com/doubiani/trends-pycaret-training-inference-doubiani>

Le kernel Kaggle de l’approche Deep Learning est disponible ici :

<https://www.kaggle.com/doubiani/neuroimaging-in-depth-understanding-eda-doubiani>

Le kernel Kaggle de l’approche Algorithme Génétique est disponible ici :

<https://www.kaggle.com/doubiani/gplearn-genetic-algorithm-doubiani>

**SOMMAIRE**

[1. La plateforme Kaggle 3](#_Toc40984624)

[2. La problématique et son Interprétation 3](#_Toc40984625)

[2.1. Contexte et interprétation 3](#_Toc40984626)

[3. Exploration du dataset 5](#_Toc40984627)

[3. Approche Algorithme Génétique 11](#_Toc40984628)

[3.1. Principe 11](#_Toc40984629)

[3.2. Implémentation 12](#_Toc40984630)

[4. Package PyCARET : Test de l’AutoML 16](#_Toc40984631)

[4.1. Principe 16](#_Toc40984632)

[4.2. Implémentation 16](#_Toc40984633)

[5. Modèle Deep Learning (AUTOML Pour Deep Learning : AutoKeras) 22](#_Toc40984634)

[5.1. Principe 23](#_Toc40984635)

[5.2. Implémentation 23](#_Toc40984636)

[5.3. Résultat 24](#_Toc40984637)

[6. Conclusions 24](#_Toc40984638)

[7. Axes d’améliorations 25](#_Toc40984639)

[8. Sources bibliographiques 25](#_Toc40984640)

# La plateforme Kaggle

Kaggle est une plateforme web créé en avril 2010 par Anthony Goldbloom, organisant des compétitions en science des données. Sur ce site, des entreprises proposent des problèmes de data science et offrent des récompenses aux participants obtenant les meilleures performances. Kaggle fourni un outil de travail permettant l’utilisation de CPU et GPU pour concevoir du code et un notebook. Il est orienté pour une utilisation en communauté, avec la possibilité de participer aux discutions avec de nombreux Data Scientist et Data Analyst, de suivre des formations, de fournir des Datasets et d’évoluer graduellement dans chaque catégorie.

# La problématique et son Interprétation

### Contexte et interprétation

La recherche sur le cerveau humain est l'un des domaines d'études les plus complexes pour les scientifiques. Nous savons que l'âge et d'autres facteurs peuvent affecter son fonctionnement et sa structure, mais il est nécessaire de poursuivre les recherches sur ce qui se passe spécifiquement dans le cerveau. Une grande partie de la recherche utilisant l'IRM, les data scientists sont bien placés pour apporter de nouveaux éclairages au domaine en analysant les données issues des IRMs. Plus particulièrement, les spécialistes de la neuroimagerie recherchent des marqueurs mesurables du comportement, de la santé ou des troubles pour aider à identifier les régions cérébrales pertinentes et leur contribution aux effets typiques ou symptomatiques.

Qu'est-ce que la neuroimagerie ?

La neuroimagerie ou imagerie cérébrale est l'utilisation de diverses techniques pour obtenir, directement ou indirectement, des images de la structure, de la fonction ou de la pharmacologie du système nerveux. Il s'agit d'une discipline relativement nouvelle en médecine, en neurosciences et en psychologie. Les médecins spécialisés dans la réalisation et l'interprétation de la neuroimagerie en milieu clinique sont des neuroradiologues.

La neuroimagerie se divise en deux grandes catégories :

* L'imagerie structurelle, qui traite de la structure du système nerveux et du diagnostic des maladies intracrâniennes (comme les tumeurs) et des blessures graves (à grande échelle).
* L'imagerie fonctionnelle, qui est utilisée pour diagnostiquer des maladies et des lésions métaboliques à une échelle plus fine (comme la maladie d'Alzheimer), ainsi que pour la recherche en psychologie neurologique, cognitive et la construction d'interfaces cerveau-ordinateur.

Comment se déroule l'IRM du cerveau ?

L'imagerie par résonance magnétique (IRM) du cerveau est un test indolore et non invasif qui produit des images détaillées de votre cerveau et de votre tronc cérébral. Un appareil d'IRM crée les images en utilisant un champ magnétique et des ondes radio. Ce test est également connu sous le nom d'IRM cérébrale ou d'IRM crânienne. Vous vous rendrez dans un hôpital ou un centre de radiologie pour passer une IRM crânienne.

Un IRM est différent d'un CT scan ou d'une radiographie en ce sens qu'il n'utilise pas de radiation pour produire des images. Un IRM combine les images pour créer une image en 3D de vos structures internes. Il est donc plus efficace que les autres scanners pour détecter les anomalies dans les petites structures du cerveau telles que l'hypophyse et le tronc cérébral. Parfois, un agent de contraste, ou un colorant, peut être administré par voie intraveineuse (IV) pour mieux visualiser certaines structures ou anomalies.

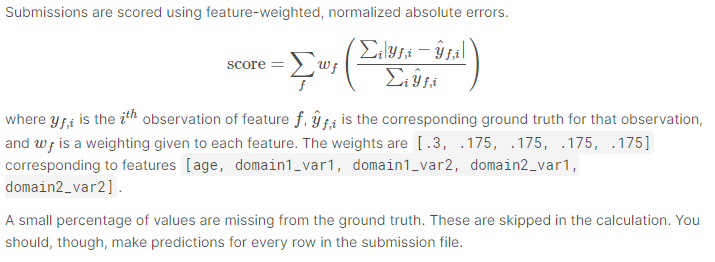
Une IRM fonctionnelle du cerveau (IRMf) est utile pour les personnes qui pourraient devoir subir une chirurgie du cerveau. L'IRMf permet de localiser les zones du cerveau responsables de la parole et du langage, ainsi que des mouvements du corps. Pour ce faire, elle mesure les changements métaboliques qui se produisent dans votre cerveau lorsque vous effectuez certaines tâches. Au cours de ce test, vous devrez peut-être effectuer de petites tâches, comme répondre à des questions de base ou taper du bout des doigts sur votre pouce.

**Quelles sont les attentes vis-à-vis de l'hôte du concours ?**

* **Les hôtes attendent des modèles qui se généralisent bien sur les données d'un autre scanner/site (site 2). Tous les sujets du site 2 ont été assignés aux données de tests, leurs scores ne sont donc pas disponibles.**
* **Bien qu'il y ait moins d'individus du site 2 que d’individus du site 1 dans les données de tests, le nombre total d'individus du site 2 ne sera révélé qu'après la fin du concours. Pour rendre les choses plus intéressantes, les identifiants de certains sujets du site 2 ont été révélés.**
* **Utilisez-les pour que vos modèles prennent en compte les effets du site. Les effets de site sont une forme de biais. Pour bien généraliser, les modèles doivent apprendre des caractéristiques qui ne sont pas liées aux effets de site ou qui ne sont pas en relation avec eux.**

**Le but de cette compétition Kaggle est de prédire l'âge et d’autres variables issues de deux domaines (scanners) différents en utilisant des caractéristiques dérivées des images IRM du cerveau comme données d'entrée. Il faut donc prédire au total 5 variables numériques. Il s’agit donc d’un problème de régression.**

**L’évaluation des soumissions se fera selon une métrique spécifique :**

****

# Exploration du dataset

Comprendre les données.

* fMRI\_train - un dossier contenant 53 cartes spatiales 3D pour les échantillons de trains au format [.mat].
* fMRI\_test - un dossier contenant 53 cartes spatiales 3D pour des échantillons de test au format [.mat].
* fnc.csv - fonctions de corrélation FNC statique pour les échantillons de train et de test.

SCN - Sub-cortical Network

ADN - Auditory Network

SMN - Sensorimotor Network

VSN - Visual Network

CON - Cognitive-control Network

DMN - Default-mode Network

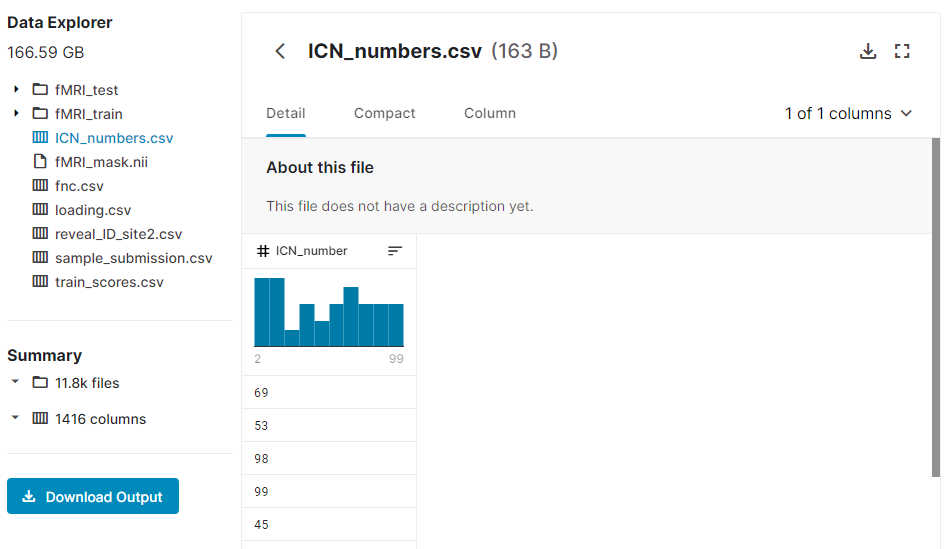
CBN - Cerebellar Network

* loading.csv - chargements sMRI SBM pour les échantillons de train et d'essai.
* train\_scores.csv - âge et valeurs d'évaluation pour les échantillons de train.
* reveal\_ID\_site2.csv - une liste d'identifiants de sujets dont les données ont été collectées avec un scanner différent de celui des échantillons d’entrainement.
* fMRI\_mask.nii - une carte spatiale binaire en 3D.
* ICN\_numbers.txt - numéros de réseau de connectivité intrinsèque (intrinsic connectivity network numbers) pour chaque carte spatiale d'IRMf ; correspond aux noms des FNC.
* [Note] - Les fichiers [.mat] peuvent être lus en python en utilisant h5py, et le fichier [.nii] peut être lu en python en utilisant nilearn.

Comment les variables ont été obtenues ?

Une stratégie non biaisée a été utilisée pour obtenir les variables fournies. Cela signifie qu'un grand ensemble de données d'imagerie distinct et sans rapport a été utilisé pour apprendre les modèles des variables. Ensuite, ces modèles ont été "projetés" sur les données d'imagerie originales de chaque individu utilisé pour ce concours en utilisant l'analyse des composantes indépendantes limitées dans l'espace (scICA) via l'ICA (independent component analysis) guidé par les informations de groupe (GIG-ICA).

* Le premier ensemble de variables est le chargement de la morphométrie basée sur la source (SBM : Source Based Morphometry). Il s'agit de poids au niveau de l'individu provenant d'une décomposition ICA au niveau du groupe de cartes de concentration de matière grise provenant de scans d'IRM structurale (sMRI).
* Le deuxième ensemble est constitué de matrices de connectivité de réseau fonctionnel (FNC :functional network connectivity) statiques. Il s'agit des valeurs de corrélation croisée au niveau du sujet parmi 53 composantes temporelles estimées à partir de l'ICA GIG de l'IRM fonctionnelle au repos (IRMf).
* Le troisième ensemble de caractéristiques est constitué par les cartes spatiales des composantes (SM). Il s'agit d'images 3D de 53 réseaux spatiaux au niveau du sujet, estimées à partir du GIG-ICA de l'IRM fonctionnelle au repos (IRMf).



**Pour des raisons de coûts et de temps de calcul, je n’utiliserais que les données numériques fournies dans les fichiers suivants :**

**'fnc.csv',**

**'ICN\_numbers.csv',**

**'loading.csv',**

**'reveal\_ID\_site2.csv',**

**'sample\_submission.csv',**

**'train\_scores.csv'**

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 5877 entries, 0 to 5876

Data columns (total 6 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 Id 5877 non-null int64

1 age 5877 non-null float64

2 domain1\_var1 5439 non-null float64

3 domain1\_var2 5439 non-null float64

4 domain2\_var1 5838 non-null float64

5 domain2\_var2 5838 non-null float64

dtypes: float64(5), int64(1)

memory usage: 275.6 KB

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 11754 entries, 0 to 11753

Data columns (total 27 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 Id 11754 non-null int64

1 IC\_01 11754 non-null float64

2 IC\_07 11754 non-null float64

3 IC\_05 11754 non-null float64

4 IC\_16 11754 non-null float64

5 IC\_26 11754 non-null float64

6 IC\_06 11754 non-null float64

7 IC\_10 11754 non-null float64

8 IC\_09 11754 non-null float64

9 IC\_18 11754 non-null float64

10 IC\_04 11754 non-null float64

11 IC\_12 11754 non-null float64

12 IC\_24 11754 non-null float64

13 IC\_15 11754 non-null float64

14 IC\_13 11754 non-null float64

15 IC\_17 11754 non-null float64

16 IC\_02 11754 non-null float64

17 IC\_08 11754 non-null float64

18 IC\_03 11754 non-null float64

19 IC\_21 11754 non-null float64

20 IC\_28 11754 non-null float64

21 IC\_11 11754 non-null float64

22 IC\_20 11754 non-null float64

23 IC\_30 11754 non-null float64

24 IC\_22 11754 non-null float64

25 IC\_29 11754 non-null float64

26 IC\_14 11754 non-null float64

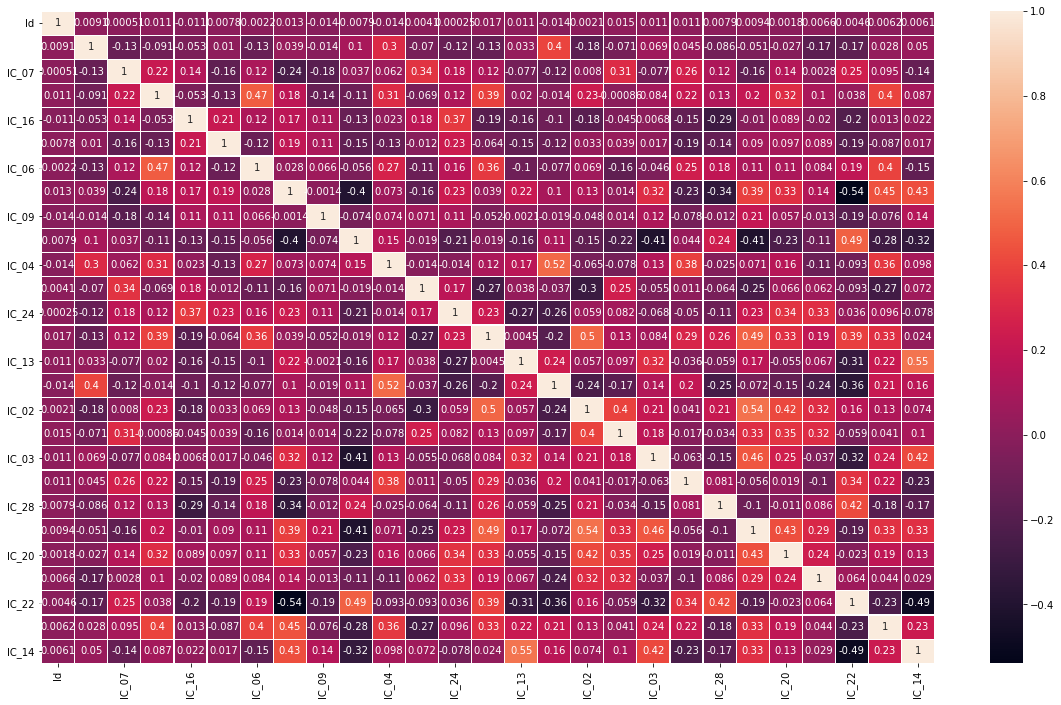
dtypes: float64(26), int64(1)

memory usage: 2.4 MB

**CORRELATIONS ENTRE VARIABLES CIBLES**

****

**CORRELATIONS ENTRE VARIABLES EXPLICATIVES**

****

Interprétation de la HeatMap

La première chose à noter est que seules les caractéristiques numériques sont comparées, car il est évident que nous ne pouvons pas établir de corrélation entre les alphabets ou les chaînes de caractères. Avant de comprendre le tracé, voyons ce qu'est exactement la corrélation.

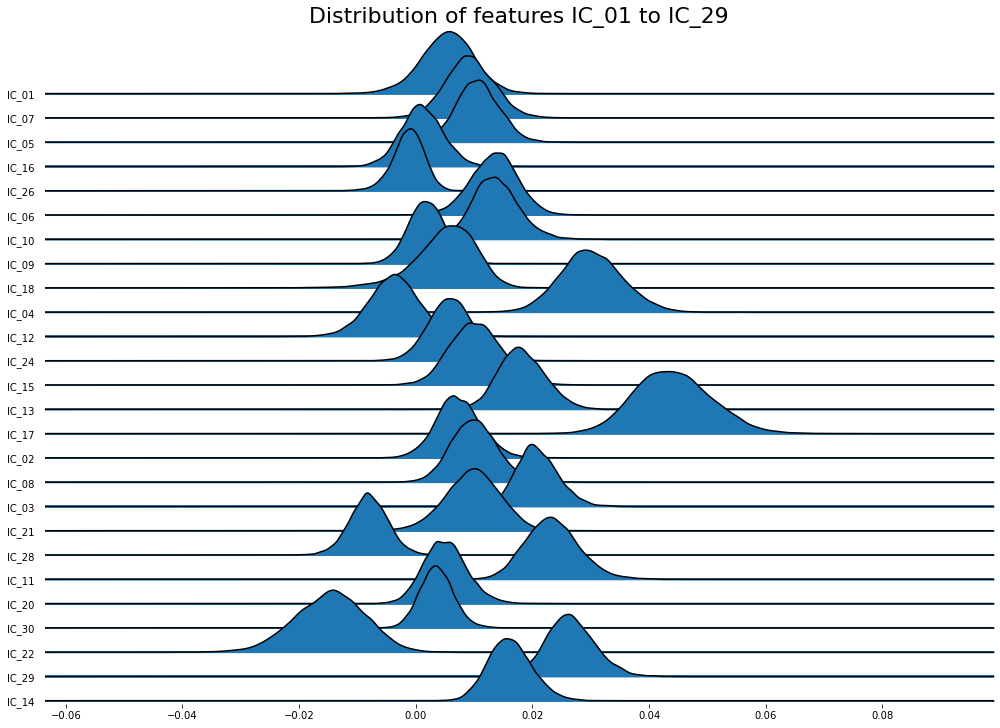
**CORRELATION POSITIVE** : si une augmentation de la caractéristique A entraîné une augmentation de la caractéristique B, alors elles sont corrélées positivement. Une valeur de 1 signifie une corrélation positive parfaite.

**CORRELATION NEGATIVE :** si une augmentation de la caractéristique A entraîné une diminution de la caractéristique B, alors elles sont corrélées négativement. Une valeur de -1 signifie une corrélation négative parfaite.

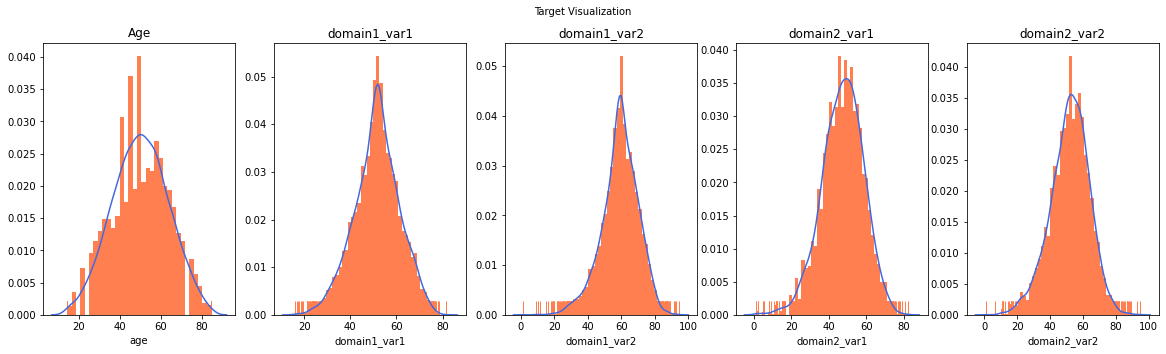
Disons maintenant que deux caractéristiques sont fortement ou parfaitement corrélées, de sorte que l'augmentation de l'une entraîne l'augmentation de l'autre. Cela signifie que les deux caractéristiques contiennent des informations très similaires et qu'il y a très peu ou pas de variation dans les informations. C'est ce qu'on appelle la multi colinéarité, car les deux contiennent presque les mêmes informations.

Lors de la création ou de l’entrainement de modèles, nous devrions essayer d'éliminer les caractéristiques redondantes car cela réduit le temps de formation et de nombreux autres avantages.

**VARIABLE LOADING\_DATA :**



**REPARTITION DES DONNEES DES VARIABLES CIBLES :**



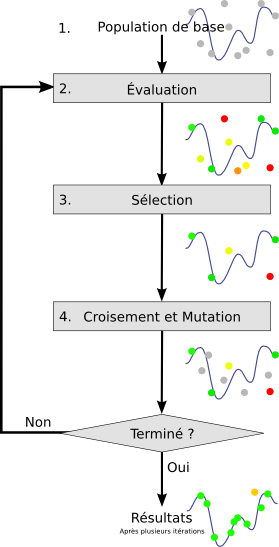
**Les données spatiales font appel à des réseaux convultifs et donc une approche différente de celle que j’ai choisi.**

**C’est pourquoi j’ai donc choisi de ne pas expliquer en détails toute la partie visualisation des données 3D du cerveau, une explication de ces données et des exemples sont néanmoins donnés dans ce notebook : neuroimaging-in-depth-understanding-eda-model-deep-learning-doubiani.ipynb**

# 3. Approche Algorithme Génétique

## 3.1. Principe

* Le progrès scientifique, en particulier dans les sciences physiques et biologiques, est une histoire d'hypothèses produisant des prédictions vérifiables qui sont ensuite soit confirmées soit rejetées par des observations (c'est-à-dire des données). Même dans la modélisation prédictive, nous ajustons généralement un modèle donné aux données observées. Et si nous pouvions aller dans l'autre sens ?
* Et si nous pouvions prendre les données, et trouver l'équation qui aurait produit le plus fidèlement les données que nous observons ? La régression symbolique nous offre une opportunité de le faire. Elle recherche les espaces de solutions des équations possibles, en combinant des opérateurs mathématiques avec des formes fonctionnelles de manière aléatoire, guidée par le succès de l'évolution (par exemple, en rassemblant les formes mathématiques les plus prometteuses à l'aide d'algorithmes génétiques). De cette façon, l'équation résultante est exempte d'hypothèses préalables (par exemple, en supposant que le modèle est linéaire, régression a-la linéaire), ou de biais sur la façon dont la variable dépendante est liée aux variables indépendantes, etc.

Le principe de l’algorithme génétique se rapproche du principe de l’évolution des espèces vivantes sur terre.

On part d’une population initiale constituée d’individus qui possèdent chacun des gènes.

Cette population est évaluée suivant un ou plusieurs critères.

Ensuite, une sélection est faite. Par exemple les individus les moins bons seront éliminés.

Puis, pour les meilleurs, des croisements entre les individus seront effectués.

Enfin, des mutations de quelques gènes de certains individus de la population peut avoir lieu.

Dans notre cas,

 La population est un ensemble de plusieurs équations constituée au hasard avec des opérateurs de base (addition, soustraction, division, multiplication)

 Un individu est donc une seule équation parmi la population générée.

 Les gènes sont les opérations de base constituants les équations.

La mutation d’un gène correspond à la modification d’un opérateur de base.

La métrique d’évaluation sera la « mean average error ».

## 3.2. Implémentation

Le notebook **P08\_01\_gplearn.ipynb** contient toutes les routines de fonctionnement de l’algo génétique.

#### 3.2.1. Algorithme

Maintenant que nous avons défini les caractéristiques de la programmation génétique, l'algorithme suivant décrit le processus global :

**Entrée :** Taille de la population 𝑛, taille de la sélection 𝑛𝑠 < 𝑛, fonction fitness, critères de de fin

**Sortie :** Une collection (taille 𝑛) de programmes

1. Initialiser les génotypes 𝑛 au hasard.

2. Créer des programmes (phénotypes) à partir des génotypes.

3. Évaluer l'adéquation de chaque phénotype avec la fonction d'adéquation.

4. Si (critères de résiliation non satisfaits) alors :

a. Sélectionnez 𝑛s les individus les plus aptes.

b. Effectuer des croisements et des mutations pour créer une nouvelle génération de génotypes 𝑛.

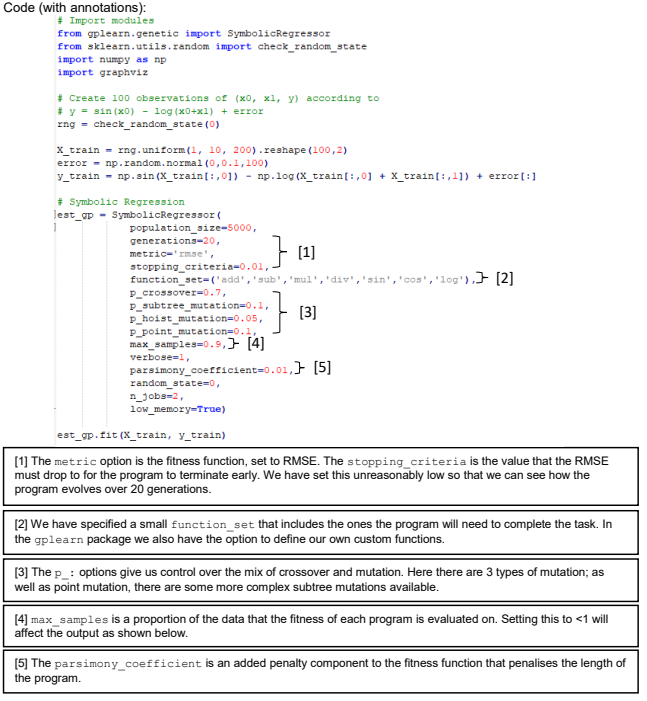
c. Retournez à l'étape 2.

Sinon :

a. Produire la génération actuelle de phénotypes.

b. Terminer.

#### 3.2.2. Programme et explications



La régression symbolique est un algorithme de "montée de colline", où l'aptitude du meilleur individu est amélioré au fil des itérations par exemple en cherchant dans l'espace modèle et en "grimpant" lorsqu'une amélioration est trouvée. Parce que la régression symbolique explore un espace modèle aussi vaste, il est impossible de garantir que nous trouverons le maximum global à la fonction de fitness et il est probable qu'au lieu de cela, nous convergerons vers un maximum local. Notre objectif est de choisir des critères de fin qui évitent s'arrêter trop tôt, par exemple à un maximum local qui n'est pas de qualité satisfaisante, mais évite aussi de trop dépenser de longues heures de recherches lorsqu'aucune amélioration ne sera apportée dans un délai raisonnable.

Nous disposons de plusieurs options pour les critères de fin, notamment :

1. Un modèle satisfaisant est trouvé

2. Il y a eu un nombre prédéterminé de nouvelles générations sans qu'aucune amélioration ne soit apportée aux meilleurs modèles dans la population

3. Un nombre maximum déterminé de générations a été créé

Une bonne idée est d'utiliser une combinaison de plusieurs critères, car le processus peut être coûteux en termes de calcul avec une grande population.

#### 3.2.3. Avantages et Inconvénients

**AVANTAGES**

* La régression symbolique présente des avantages évidents de par sa conception : elle est conçue pour explorer un large espace de modèles possibles.
* La probabilité qu'il en trouve une qui corresponde bien aux données est donc très élevée, et elle peut être réglée par la fonction de fitness pour sanctionner de manière appropriée le surapprentissage en données de formation.
* L'inclusion de fonctions telles que log () et sin () donne à l'algorithme la possibilité d'expérimenter diverses transformations de chaque variable, ce qui, sans le contexte des données, peut être très difficile à trouver.
* Si notre objectif est de sélectionner le meilleur modèle, nous pouvons prendre le meilleur individu de la dernière génération comme dans l'exemple utilisant la librairie gplearn.
* Toutefois, il est également possible de mettre en place un programme qui retient l'ensemble de la population et d'envisager de ne retenir que les meilleurs modèles de la dernière génération.

**INCONVÉNIENTS**

* La régression symbolique est très coûteuse en termes de calcul, c'est pourquoi il est judicieux d'inclure le nombre de générations ou la durée d'exécution du programme dans les critères de fin.
* Il y a également de nombreux paramètres qui doivent être testés et qui peuvent influencer la vitesse d'apprentissage, tels que l'équilibre entre la mutation et le crossover, qui nécessitent de l'expérience pour s'accorder correctement.

#### 3.2.4. Résultats

**Pour la variable Âge, la Mean Average Error est de 10.918846772701464**

**Pour la variable Domain1\_var1, la Mean Average Error est de 7.650291769242**

**Pour la variable Domain1\_var2, la Mean Average Error est de 8.270192160941**

**Pour la variable Domain2\_var1, la Mean Average Error est de 8.761652683147**

**Pour la variable Domain2\_var2, la Mean Average Error est de 9.248416619867**

**Après soumission à la plateforme Kaggle, La feature weighted normalized error (métrique du concours) est de : 0.188 ce qui en fait un résultat correct mais sans plus par rapport aux autres méthodes et aux autres candidats à la compétition.**

# 4. Package PyCARET : Test de l’AutoML

Le notebook Kaggle suivant m'a permis de faire ces tests : <https://www.kaggle.com/rohitsingh9990/trends-pycaret-training-inference>

Le fichier du notebook adapté par mes soins est disponible ici : **P08\_01\_trends-pycaret-training-inference-doubiani.ipynb**

## 4.1. Principe

Introduction à PyCaret

PyCaret est une bibliothèque d'apprentissage machine en Python, à code source ouvert et à faible niveau de codage, qui vise à réduire le temps du passage entre l'hypothèse et la compréhension jusqu’au déploiement du modèle en production. Elle est bien adaptée aux \*\*scientifiques de données chevronnés\*\* qui veulent augmenter la productivité de leurs expériences de ML en utilisant PyCaret dans leurs flux de travail ou aux \*\*scientifiques de données citoyens\*\* et aux \*\*nouveaux en science des données\*\* ayant peu ou pas d'expérience en codage. PyCaret vous permet de passer de la préparation de vos données au déploiement de votre modèle en quelques secondes en utilisant l'environnement de votre choix sur votre ordinateur. Veuillez cliquer sur le lien (<https://pycaret.org/guide/>) pour en savoir plus sur PyCaret.

## 4.2. Implémentation

#### 4.2.1. Setup

**Remarque** :

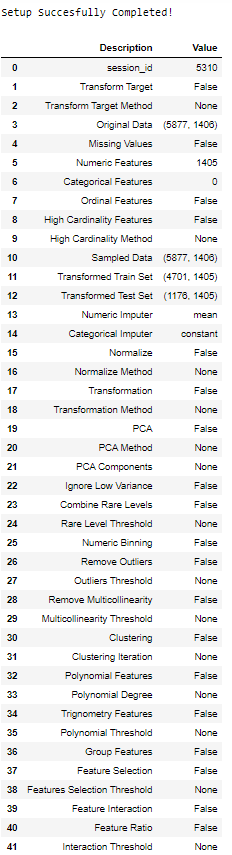
Actuellement, PyCaret n'a pas de support pour la régression multi-variable. Donc, au lieu d'un modèle pour nos 5 variables cibles, nous devons créer un modèle individuel pour chaque variable cible.

Configuration de notre ensemble de données (utiliser uniquement la variable cible "Age")

* La fonction "setup" initialise l'environnement dans pycaret et crée le pipeline de transformation pour préparer les données à la modélisation et au déploiement. setup() doit être appelée avant d'exécuter toute autre fonction dans pycaret.
* Elle prend deux paramètres obligatoires : dataframe {array-like, sparse matrix} et le nom de la colonne cible. Tous les autres paramètres sont facultatifs.

Pour plus d'informations, [https://pycaret.org/regression/#setup](https://pycaret.org/regression/%23setup)

Cela donne :

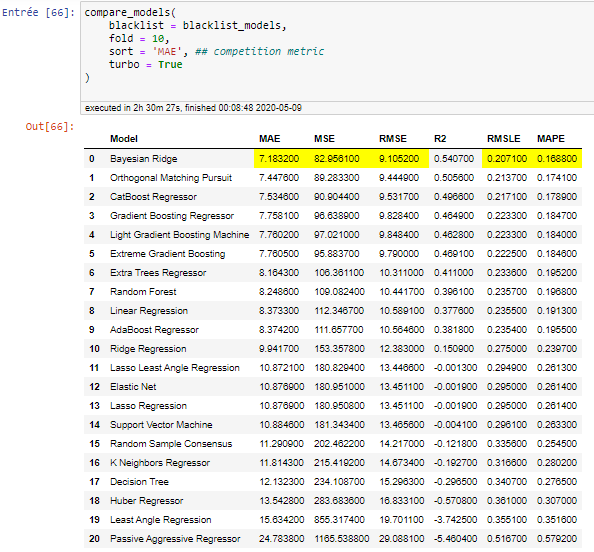


#### 4.2.2. Compare Model

Comparer les modèles

* La fonction `compare\_models` utilise tous les modèles de la bibliothèque de modèles et les note en utilisant la validation croisée K-fold. La sortie imprime une grille de score qui montre les MAE, MSE, RMSE, R2, RMSLE et MAPE par fold (CV par défaut = 10 folds) de tous les modèles disponibles dans la bibliothèque de modèles.

Pour plus d'informations[, https://pycaret.org/regression/#compare-models](,%20https:/pycaret.org/regression/%23compare-models)



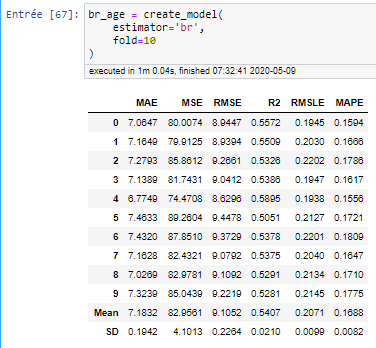
Il en ressort que pour la variable Âge, le meilleur modèle est une Bayesian Ridge Regression avec une MAE à 7.183

#### 4.2.3. Create Model

Créer un modèle

Créons une régression Bayesian Ridge conformément à ce qui a été trouvé à l'étape précédente.

* La fonction `create\_model` crée un modèle et le note en utilisant la validation croisée K-fold. (par défaut = 10 fois).
* La sortie imprime une grille de score qui montre MAE, MSE, RMSE, RMSLE, R2 et MAPE. Cette fonction renvoie un objet modèle entrainé.
* La fonction setup() doit être appelée avant d'utiliser create\_model()



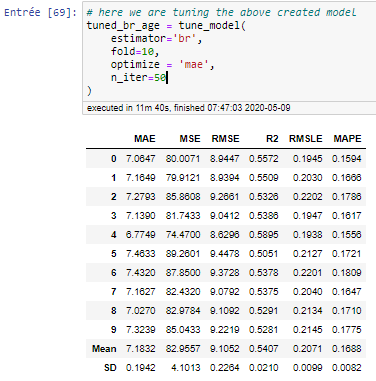
Le modèle est donc bien créé conformément aux étapes précédentes.

#### 4.2.3. Tuning Model

Régler les hyperparamètres du modèle

* La fonction `tune\_model` règle les hyperparamètres d'un modèle et le classe en utilisant la validation croisée K-fold. La sortie imprime la grille de score qui indique MAE, MSE, RMSE, R2, RMSLE et MAPE par folds (par défaut = 10 folds). Cette fonction renvoie un objet modèle entrainé.

Pour plus d'informations, [https://pycaret.org/regression/#tune-model](https://pycaret.org/regression/%23tune-model).

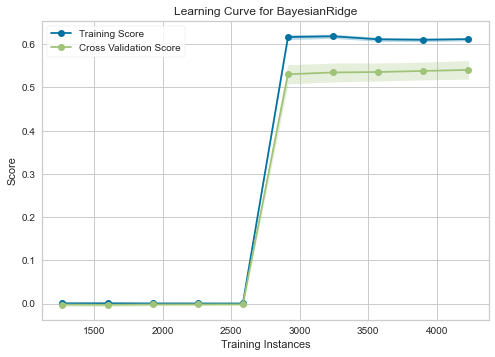


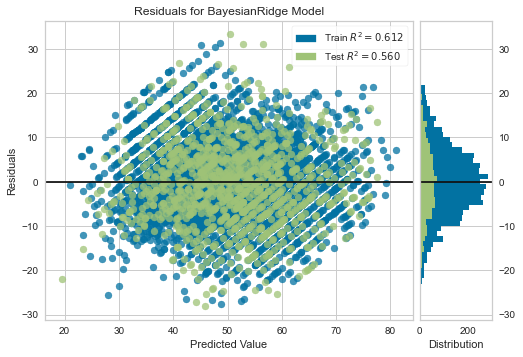
#### 4.2.4. Plot Model

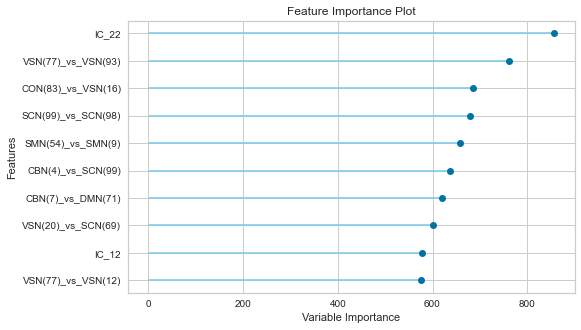
Modèle de tracé

La fonction ‘plot\_model’ prend un objet modèle entraîné et retourne un tracé basé sur l'ensemble test / hold-out. Dans certains cas, le processus peut nécessiter un nouvel entrainement du modèle. Le modèle doit être créé à l'aide de la fonction create\_model() ou tune\_model().

Pour plus d'informations, [https://pycaret.org/regression/#plot-model](https://pycaret.org/regression/%23plot-model)





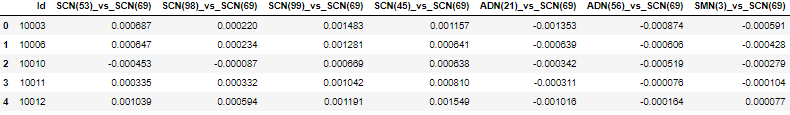


#### 4.2.5. Predict Model

Modèle de prédiction

La fonction `predict\_model` est utilisée pour prédire de nouvelles données en utilisant un estimateur entrainé. Elle accepte un estimateur créé à l'aide d'une des fonctions de pycaret qui renvoie un objet modèle entrainé ou une liste d'objets modèles entrainés créés à l'aide de stack\_models() ou de create\_stacknet(). Les nouvelles données non vues peuvent être transmises aux paramètres de données sous forme de pandas Dataframe. Si les données ne sont pas passées, le jeu de test / hold-out séparé au moment du setup() est utilisé pour générer des prédictions.

Pour plus d'informations, <https://pycaret.org/regression/#predict-model>



#### 4.2.6. Résultats

**En observant de près les comparaisons des modèles ci-dessus, nous avons fait les observations suivantes :**

* **"âge" : Le modèle Bayesian Ridge a le MAE minimum : 7.228200**
* **`domaine1\_var1` : Le régresseur CatBoost a le MAE minimum : 7.888200**
* **`domaine1\_var2` : Les machines à vecteurs de soutien ont le MAE minimum : 8.982500**
* **`domaine2\_var1` : Le régresseur CatBoost a le MAE minimum : 8.748100**
* **`domaine2\_var2` : Le régresseur CatBoost a le MAE minimum : 9.239200**

**Après soumission des prédictions sur la plateforme KAGGLE, la feature-weighted normalized error (Métrique de la compétition est de : 0.161) ce qui est le meilleur résultat des 3 approches (PyCARET, Algo Génétique et Deep Learning)**

# 5. Modèle Deep Learning (AUTOML Pour Deep Learning : AutoKeras)

Le fichier du notebook adapté par mes soins est disponible ici : **P08\_01\_neuroimaging-in-depth-understanding-eda-model-deep-learning-doubiani.ipynb**

## 5.1. Principe

Conception de réseau neurones automatiquement

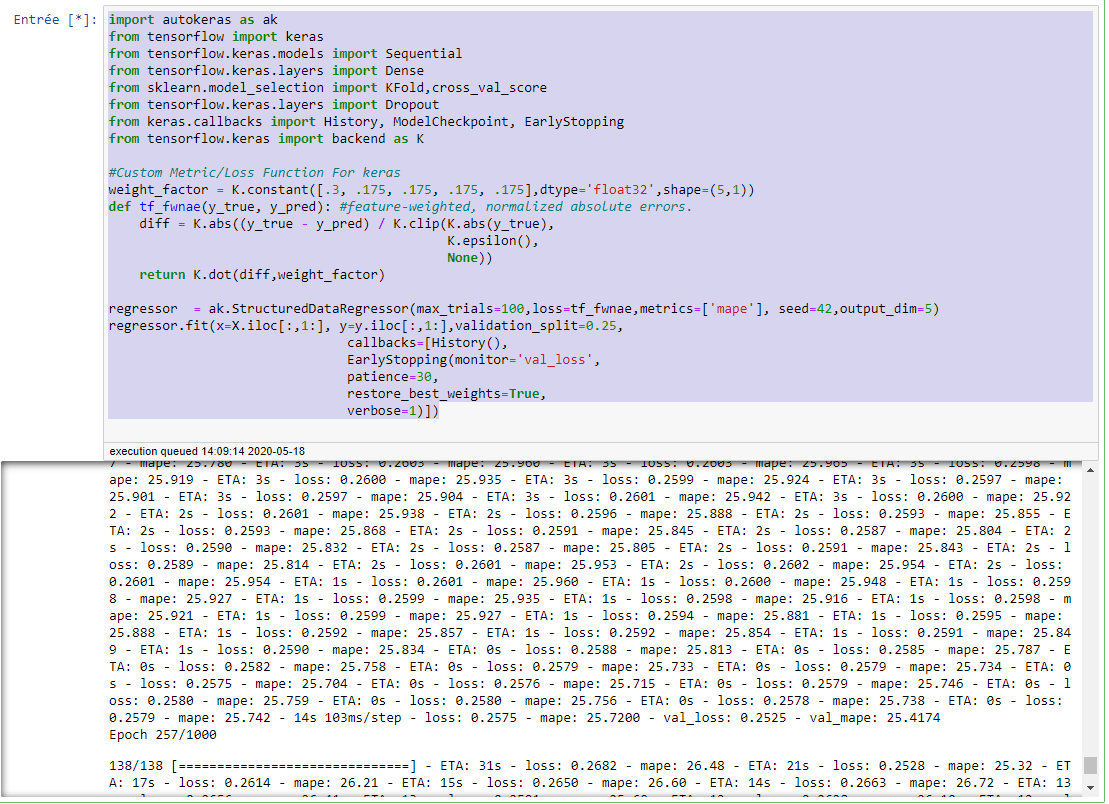
La recherche d'architecture neuronale (NAS) utilise l'apprentissage machine pour automatiser la conception des RNA (Réseaux de Neurones Artificiels). Diverses approches de la NAS ont permis de concevoir des réseaux qui se comparent bien aux systèmes conçus à la main. L'algorithme de recherche de base consiste à proposer un modèle candidat, à l'évaluer par rapport à un ensemble de données et à utiliser les résultats comme retour d'information pour apprendre le réseau NAS. Les systèmes disponibles comprennent AutoML et AutoKeras.

Les questions de conception comprennent le nombre, le type et la connectivité des couches du réseau, ainsi que la taille de chacune et le type de connexion (complète, groupée, ...).

Les hyperparamètres doivent également être définis dans le cadre de la conception (ils ne sont pas appris), régissant des questions telles que le nombre de neurones dans chaque couche, le taux d'apprentissage, le pas, la foulée, la profondeur, le champ de réception et le padding (pour les CNN), etc.

## 5.2. Implémentation

Elle est très simple :



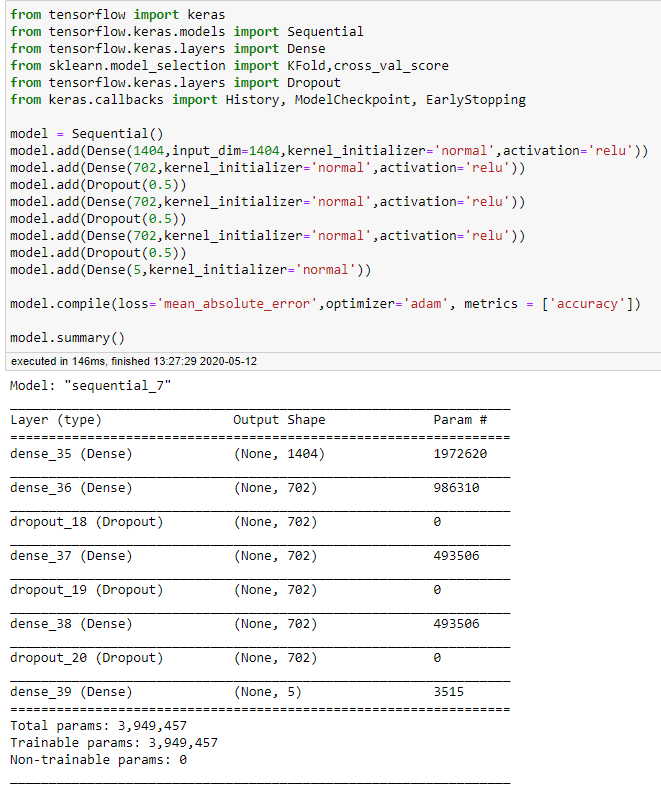
Pour des raisons techniques et de temps, ce package Autokeras est utilisé comme proof of concept. J’ai soumis les résultats à la plateforme Kaggle mais cette soumission ne sera pas celle qui sera utilisée pour l’évaluation finale de la compétition de part ses mauvais résultats.

## 5.3. Résultat

L’avantage de cette technique est qu’elle permet d’avoir une prédiction unique pour nos 5 variables en une seule et même prédiction (régression multi-variables)

L’inconvénient est que ce package est toujours en cours de développement et ne peut donc être utilisé de manière optimale pour l’instant (bugs d’export de modèles…)

La « MAE » (Mean Average Error) avec un reseau neuronal structuré ainsi :



**Est de 8.8129, ceci pour l’ensemble des prévisions des 5 variables.**

**Cependant après soumission sur la plateforme, le résultat avec le feature weighted normalized error (la métrique de la compétition) n’est que de 0.194, soit le plus mauvais résultat des 3 approches.**

# 6. Conclusions

* L’AutoML (PyCARET) permet d’arriver à un très bon résultat rapidement et avec peu de code et de temps entre l’idée et le résultat de la modélisation.
* Par ailleurs, PyCARET permet de générer le code de chaque étape afin que l’on puisse intervenir manuellement et faire les ajustements nécessaires pour améliorer encore plus le résultat ou l’intégration dans un pipeline encore plus important.
* Donc des 3 approches voici le classement des méthodes avec la métrique **feature weighted normalized error** :
  1. PyCARET (0.161)
  2. Algorithme Génétique (0.188)
  3. Deep Learning (0.194)
* A titre comparatif, en date du 25 Mai 2020 le leader de la compétition a un score **feature weighted normalized error** de 0.158 avec une approche Deep Learning Convolutionnelle 3D sur les scans.

# 7. Axes d’améliorations

* Faire plus d’explorations avec les algorithmes génétiques en faisant un **GridSearchCV** par exemple.
* Utiliser la fonction **Symbolic Transformer** de gplearn pour créer des variables non liées linéaire à la variable cible et/ou explicatives afin d’avoir de nouvelles variables à utiliser dans notre modèle d’algorithme génétique.
* Faire de **l’optimisation d’hyperparamètres** encore plus poussée sous **PyCARET**
* Faire de **l’optimisation d’hyperparamètres** encore plus poussée pour l’approche **Deep Learning**
* Faire une approche de prévision avec l’aides **caractéristiques 3D spatiales** des scans fournis (écartée dans mes approches)
* Faire à la main des **modèles multi-variables cibles de régression**
* Réutiliser les approches **PyCARET et Symbolic Regression ainsi que l’approche deep learning** avec comme fonction d'optimisation la métrique **Feature weighted normalized error** (celle de la compétition)

# 8. Sources bibliographiques

Généralités sur algo génétique : <https://khayyam.developpez.com/articles/algo/genetic/>

* Convolutional Neural Networks with Intermediate Loss for 3D Super-Resolution of CT and MRI Scans(<https://arxiv.org/pdf/2001.01330>)
* Multi-Resolution 3D CNN for MRI Brain Tumor Segmentation and Survival Prediction: <https://arxiv.org/abs/1911.08388>
* Automatic Post-Stroke Lesion Segmentation on MR Images using 3D Residual Convolutional Neural Network: (<https://arxiv.org/pdf/1911.11209>)
* Previous competition occur on kaggle regard to brain imaging. (<https://www.kaggle.com/c/mlsp-2014-mri/overview>)
* Plot brain visualization using nilearn: <https://nilearn.github.io/plotting/index.html>
* Check out the various library in python for visualization of Brain MRI. <https://www.kaggle.com/c/trends-assessment-prediction/discussion/148175>
* Thanks to soham for their great kernal help and giving Idea of visualization. <https://www.kaggle.com/soham1024/visualization-using-nilearn>
* Machine Learning for Neuroimaging with Scikit-Learn : <https://arxiv.org/pdf/1412.3919.pdf>
* Symbolic Regressor avec la libraire gplearn : <https://gplearn.readthedocs.io/en/stable/examples.html#symbolic-regressor>
* Symbolic Regression: Survival of the Best Fit : <https://www.lexjansen.com/phuse/2019/ml/ML03.pdf>
* AutoKeras : [www.autokeras.com](http://www.autokeras.com), <https://en.wikipedia.org/wiki/Artificial_neural_network>
* MultiOutput Regression <https://scikit-learn.org/stable/modules/multiclass.html#multioutput-regression>
* <https://www.kaggle.com/c/trends-assessment-prediction/discussion/145818>
* <https://www.kaggle.com/c/trends-assessment-prediction/discussion/145791>
* <https://www.kaggle.com/c/trends-assessment-prediction/discussion/145597>
* <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3181933/>
* <https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0066572>
* <https://www.kaggle.com/c/trends-assessment-prediction/discussion/146205>