+ QUIZ

Gradient Boosting

Exploitez les arbres de décision pour le Machine Learning (XGBoost, CatBoost, LightGBM)

En téléchargement

Code source

informatique technique

Collection epsilon

Guillaume SAUPIN

Table des matières

Les exemples à télécharger sont disponibles à l'adresse suivante :
http://www.editions-eni.fr
Saisissez la référence ENI de l'ouvrage **EPGRAD** dans la zone de recherche et validez. Cliquez sur le titre du livre puis sur le bouton de téléchargement.

Chapitre 2
Gradient Boosted Tree : fonctionnement

Chapitre 3
Entraîner efficacement un modèle

Chapitre 4
Comprendre et expliquer un modèle

Chapitre 5
Hyperparameters Tuning

Chapitre 6

Du bon usage des fonctions objectifs

Chapitre 7
Gradient Boosting pour les séries temporelles

Chapitre 8
XGBoost, LightGBM ou CatBoost ?

Chapitre 9
Approche multimodèle

Chapitre 10
Pour aller plus loin

Chapitre 3
Entraîner efficacement un modèle

1. Entraîner efficacement

La théorie, les fondements mathématiques et une implémentation des méthodes de Gradient Boosting ayant été présentés dans le précédent chapitre, il est temps de passer à la pratique. Pour cela, nous allons décrire dans ce chapitre les conditions à respecter pour assurer un entraînement efficace, permettant de produire des modèles précis et généralisant correctement.

Après un aperçu des grandes étapes de l'entraînement d'un modèle, une seconde section traitera en détail de la préparation des données, de leur nettoyage et enrichissement, puis de la construction des datasets d'entraînement puis d'évaluation.

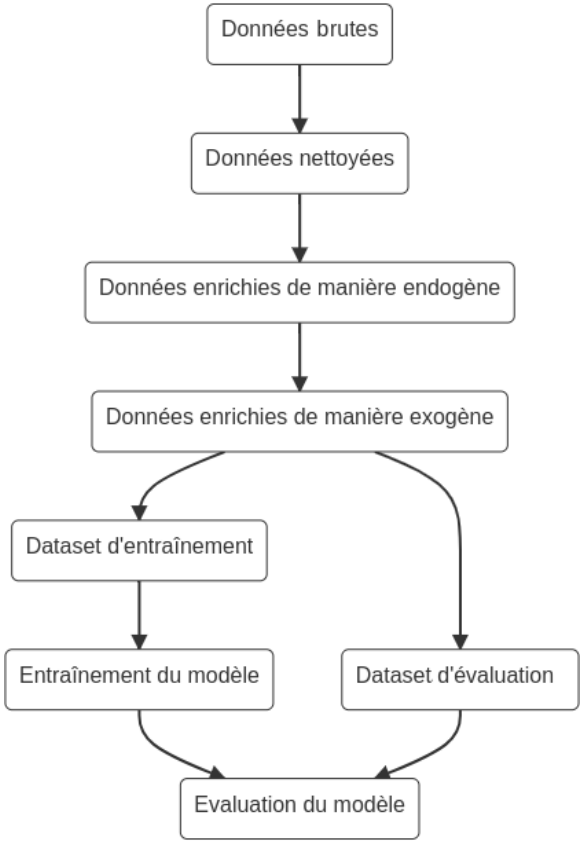
La troisième section présentera rapidement les principes de l'entraînement en lui-même, tandis que la quatrième plongera dans le détail des différentes métriques utilisées pour évaluer la qualité d'un modèle de régression ou de classification.

La cinquième section traitera du problème du sur-apprentissage, de sa détection et des leviers qu'offrent les méthodes de Gradient Boosting pour l'éviter.

Enfin, quelques cas pratiques issus de données open source seront étudiés.

Démarche globale

Avant de plonger dans les arcanes de la réalisation d'un entraînement, il est important de donner une vision globale des grandes étapes impliquées. Le schéma ci-dessous en dresse une vue synthétique :



Partant des données brutes, qui sont successivement nettoyées puis enrichies de manière endogène et exogène, l'entraînement se fait sur une sous-partie des données, tandis que les données restantes sont mises de côté pour une évaluation future.

Remarque

Il est crucial de bien faire la distinction entre datasets de test et datasets d'évaluation. Les datasets de test sont utilisés durant l'entraînement, afin d'évaluer la progression de l'apprentissage. Les datasets d'évaluation ne sont utilisés qu'une fois le modèle construit, afin de s'assurer de la capacité de généralisation de ce dernier. Il est impératif que ces deux types de datasets, test et validation, soient bien cloisonnés, afin d'éviter tout biais lors de l'évaluation.

2. Préparation des données

La pièce maîtresse, lors de l'entraînement d'un modèle, représente les données. Peu importe la complexité ou la sophistication d'un modèle, si les données à disposition ne sont pas représentatives du problème, pas assez riches ou assez nombreuses, alors la qualité de la prédiction ne sera pas satisfaisante.

Comme l'a mis en évidence le schéma ci-dessus, cette phase doit se faire de manière préalable à la construction des datasets d'entraînement et d'évaluation, en veillant toutefois à ne pas contaminer le dataset d'évaluation en faisant fuiter des données de l'un à l'autre.

2.1 Enrichissement des données

La richesse d'un corpus de données tient en la présence de nombreuses caractéristiques, ou *features* en anglais, qui offrent une vision selon plusieurs angles du problème.

Dans le contexte d'analyses socio-économiques, plus la population étudiée est qualifiée, et ce à travers de nombreux indicateurs, plus les modèles construits seront précis. Revenus, mode de vie, alimentation, niveau d'éducation, loisirs, temps de sommeil, taille de la famille, liens avec les ascendants et descendants, patrimoine... toute information mérite d'être collectée.

Dans cet exemple, les données enrichissant le corpus sont externes. Il s'agit donc de données exogènes.

Une autre voie est à envisager pour enrichir un ensemble de données : l'augmentation à partir de données endogènes.

Ce sont des caractéristiques additionnelles construites non pas en se tournant vers l'extérieur, mais au contraire en restant dans les données existantes et en les retravaillant.

L'exemple des séries temporelles est parlant. Le signal brut qui les constitue est généralement issu de l'échantillonnage à une fréquence donnée des mesures d'un capteur. Ce type de signal contient intrinsèquement beaucoup d'informations, mais qu'il faut extraire. De nombreux traitements peuvent être appliqués comme une transformée de Fourier, une décomposition en ondelettes, un calcul d'énergie, d'amplitude, ou d'autres convolutions... autant d'informations qui amendent la donnée brute et accroissent la performance d'un modèle.

2.2 Volume de données

Le volume de données est le premier critère à examiner. Plus le système à modéliser est complexe, plus il faudra de données pour en capturer la complexité. Si par exemple la tâche à réaliser consiste à classer une image dans une catégorie parmi 1000 autres, il faut nécessairement avoir au moins 1000 échantillons de données.

Le volume de données est aussi à mettre en relation avec les hyperparamètres, qui président à la construction de l'ensemble d'arbres de décision. S'il est par exemple le fruit de l'assemblage de 100 arbres de profondeur 4, il faut a minima 100 $*4^2$ = 1600 échantillons de données différents pour alimenter les 1600 feuilles de ces arbres.

Le nombre de lignes n'est d'ailleurs pas le seul critère à considérer lorsqu'il est question de volume. Il faut aussi tenir compte du nombre de colonnes et de la cardinalité des éléments uniques au sein d'une colonne.

Il est impossible par exemple de classer des images dans 1000 catégories avec seulement deux colonnes contenant chacune deux valeurs distinctes. Même si le dataset d'entraînement contient des millions de lignes, il n'y aura toujours que 2x2 = 4 types d'images différents.

2.3 Nettoyage des données

Une phase à ne pas négliger reste celle du nettoyage des données. Même si la collecte et le stockage des informations ont nettement progressé, le Data Scientist est toujours confronté à trois grands types de nettoyages : les données mal formatées, les données aberrantes et les données incohérentes.

Les données mal formatées sont pénibles à traiter, mais s'identifient et se corrigent facilement. Il s'agit par exemple de problème de format dans des fichiers CSV. Il est impossible d'exploiter des données tant qu'elles ne sont pas dans le format idoine.

Les données aberrantes sont plus facilement identifiables. Ce sont par exemple des températures en dessous du zéro absolu ou des vitesses dépassant celle de la lumière. Quelques règles métier permettent de les identifier. Se pose alors la question de savoir qu'en faire : faut-il les corriger ou les supprimer ? La réponse est souvent à chercher dans le volume et la représentativité des données à disposition. Si elles sont nombreuses, il est plus prudent de les écarter. Si le dataset est pauvre, il peut être intéressant de chercher à les corriger.

Enfin, les données incohérentes sont les plus difficiles à reconnaître. Prise individuellement, chaque feature peut sembler correcte, mais une analyse plus globale met souvent en évidence ces incohérences. Dans ce cas, il faut s'appuyer sur l'homme de l'art pour les identifier et les traiter.

2.4 Complétude des données

Un autre sujet assez proche du nettoyage est celui des données manquantes. Il arrive fréquemment que pour un échantillon de données, toutes les caractéristiques ne soient pas présentes.

Dans ce cas, deux alternatives sont possibles, arbitrer là encore suivant le volume de données à disposition : soit ces lignes de données sont purement et simplement supprimées, soit les données absentes sont remplies par des valeurs par défaut.

Plusieurs stratégies peuvent être utilisées pour déterminer la valeur à utiliser comme substitut : il peut s'agir d'une constante, d'une moyenne, d'une médiane, de la catégorie la plus fréquente… Il est du ressort du Data Scientist de juger quelle est la plus pertinente.

2.5 Intégration des données catégorielles

L'essence du fonctionnement des arbres de décision est de partitionner les données de manière à appliquer des corrections propres à chaque partition. Ce découpage se base sur les données d'une caractéristique donnée, en déterminant le seuil de séparation optimal.

Cela implique qu'il est possible d'ordonner les valeurs stockées d'une feature. Si les valeurs sont numériques, cela ne pose pas de souci. En revanche, si les données sont de type catégoriel, cette relation d'ordre n'existe pas.

Pour alimenter un modèle de type arbre de décision, suivant les librairies utilisées, il faudra appliquer un prétraitement sur ces données.

Remarque

CatBoost est une implémentation des méthodes de Gradient Boosting pour les arbres de décision qui fonctionne directement sur des données catégorielles. Aucun prétraitement n'est nécessaire.

Plusieurs solutions sont applicables pour réaliser cette conversion. La plus connue est le *One Hot Encoding*, qui procède en créant une colonne pour chaque catégorie. Si une ligne de données contient une catégorie donnée, alors la nouvelle colonne associée à celle-ci contiendra un 1, tandis que les autres colonnes resteront à 0.

Le tableau ci-dessous illustre cette mécanique :

Ligne	Couleur	Rouge	Vert	Bleu
1	Rouge	1	0	0
2	Vert	0	1	0
3	Bleu	0	0	1

L'inconvénient majeur de cette méthode est qu'elle introduit autant de colonnes qu'il y a de catégories différentes. Dans l'exemple ci-dessus, trois couleurs ont généré trois colonnes additionnelles.

Dès que le nombre de catégories dépasse la centaine, cela peut devenir problématique pour des raisons de temps de calcul.

Il faut alors se tourner vers d'autres solutions, telles que le *Target Encoding* ou le *GLMM Encoding*, qui vont permettre de n'ajouter qu'une seule colonne.

2.6 Dataset d'entraînement

Construire le dataset d'entraînement, une fois que les données ont été prétraitées comme nous venons de le voir, est assez simple. Cela se fait généralement en creux de la constitution du dataset d'évaluation : le dataset d'entraînement contenant généralement les données restantes.

Cela est à nuancer cependant, en n'oubliant pas que ce dataset doit être représentatif du problème, et qu'il convient donc de s'assurer de l'équilibrage de ce dernier. Il est possible que les données collectées contiennent davantage tel ou tel type de cas. Il faut donc, dans ce cas, s'assurer de la représentativité et de l'équilibrage.

2.7 Dataset d'évaluation

2.7.1 Rôle du dataset d'évaluation

L'autre pièce maîtresse lors de la construction d'un modèle n'est autre que le dataset d'évaluation. Il est tout à fait déterminant, car c'est lui qui va permettre de juger de la qualité du modèle en le soumettant à des données qu'il n'a jamais rencontrées. C'est en cela que cette phase évalue la capacité à généraliser du modèle.

Ce dataset doit donc être soigneusement construit de manière à être suffisamment représentatif du problème testé. La difficulté qui émerge généralement lors de sa constitution est qu'il ne peut pas être trop grand, dans la mesure où toutes les données qui se retrouvent dans le dataset d'évaluation sont autant de données qui ne bénéficieront pas à l'entraînement.