

Version en ligne

OFFERTE !

pendant 1 an

+ 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



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.

Avant-propos

- 1. Constructionnisme 11
- 2. Objectifs 12
- 3. Structure de l'ouvrage 12
- 4. Matériel 13

Chapitre 1

Gradient Boosted Tree : contexte et théorie

- 1. Contexte 15
 - 1.1 Océan de données 16
 - 1.2 Quels bénéfices en tirer ? 16
 - 1.3 Souci de l'efficacité 19
- 2. Cas d'application des méthodes de Gradient Boosting 20
 - 2.1 Palmarès reconnu 20
 - 2.2 Large spectre d'application 20
 - 2.3 Focus sur les données structurées 21
- 3. Avantages indéniables 22
 - 3.1 Simplicité de configuration 22
 - 3.2 Polyvalence au service de l'efficacité 22
 - 3.3 Interprétabilité essentielle pour l'adoption 23
 - 3.4 Capacité d'ingestion des données importantes 23
 - 3.5 Grande robustesse 24
 - 3.6 Personnalisation au cas métier 24
 - 3.7 Très bon niveau de précision 24

2 --- Gradient Boosting

Exploitez les arbres de décision pour le Machine Learning

4. Limitations	25
4.1 Absence de support de l'extrapolation	25
4.2 Aucun travail sur les features	25

Chapitre 2

Gradient Boosted Tree : fonctionnement

1. Méthodes ensemblistes	27
1.1 Principes et motivations.	28
2. Arbres de décision.	31
2.1 Principes et motivations.	32
2.2 Exemple minimal	33
2.3 Ensemble d'arbres de décision	36
3. Gradient Boosting.	39
3.1 Principe du boosting.	40
3.2 Fondements mathématiques	41
3.2.1 Paramètres.	41
3.2.2 Fonction objectif.	42
3.2.3 Optimisation	44
3.2.4 Interprétation	47
3.3 Implémentation minimale.	47
3.3.1 Classe DecisionEnsemble.	48
3.3.2 Exemple.	55
3.4 Régularisation	59

Chapitre 3

Entraîner efficacement un modèle

1. Entraîner efficacement	67
2. Préparation des données.	69
2.1 Enrichissement des données	69
2.2 Volume de données	70
2.3 Nettoyage des données.	71

- 2.4 Complétude des données 71
- 2.5 Intégration des données catégorielles 72
- 2.6 Dataset d'entraînement 73
- 2.7 Dataset d'évaluation. 73
 - 2.7.1 Rôle du dataset d'évaluation 73
 - 2.7.2 Importance d'un bon découpage 74
- 3. Entraînement 75
 - 3.1 Choix des hyperparamètres 75
 - 3.2 Choix de l'objectif 75
- 4. Métriques à considérer 76
 - 4.1 Rôle des métriques 77
 - 4.2 Périmètre de calcul des métriques 77
 - 4.3 Métriques pour la régression 78
 - 4.3.1 Mean Absolute Error 78
 - 4.3.2 Mean Absolute Percentage Error 79
 - 4.3.3 Mean Squared Error 81
 - 4.3.4 RMSE. 82
 - 4.3.5 Coefficient de détermination : R^2 83
 - 4.3.6 Modèle simple de référence 84
 - 4.4 Métriques pour la classification. 84
 - 4.4.1 Accuracy 85
 - 4.4.2 Faux positifs 85
 - 4.4.3 Faux négatifs. 85
 - 4.4.4 Matrice de confusion 86
 - 4.4.5 Précision 86
 - 4.4.6 Rappel 87
 - 4.4.7 F1-score 87
 - 4.5 Cross-validation 87
 - 4.5.1 Motivation de la cross-validation 87
 - 4.5.2 Principe de la cross-validation 88
 - 4.5.3 Découpages possibles 88

4 --- Gradient Boosting

Exploitez les arbres de décision pour le Machine Learning

5. Le piège du sur-apprentissage	90
5.1 Description	90
5.2 Détection	90
5.2.1 Contraintes sur les hyperparamètres	93
5.2.2 Régularisation	93
5.2.3 Sous-échantillonnage	94
5.2.4 Early stopping	94
6. Application (digits dataset)	97
6.1 Configuration par défaut	97
6.2 Simplification du modèle	99

Chapitre 4

Comprendre et expliquer un modèle

1. Explicabilité	103
1.1 Motivation	103
1.2 Vue globale	104
1.3 Vue locale	105
1.4 Particularité des arbres de décision	105
2. Feature importances	106
2.1 Présentation	106
2.2 Calcul basé sur le niveau d'utilisation	106
2.3 Calcul basé sur les gains	107
2.4 Calcul basé sur la couverture	107
2.5 Implémentation	107
2.6 Interprétation des trois modes de calculs	116
3. SHAP : SHapley Additive exPlanation	119
3.1 Objectif : construire un modèle explicatif	119
3.2 Valeurs de Shapley	120
3.3 Implémentation	122
3.3.1 Modèle linéaire	122
3.3.2 Modèle générique	124

3.4	Valeur de Shapley pour les arbres de décision	128
3.4.1	Informations stockées pour les arbres de décision	129
3.4.2	Exploitation pour la méthode SHAP	129
3.5	Interprétation et visualisation des valeurs de Shapley	130
3.5.1	Explicabilité locale	130
3.5.2	Explicabilité globale	131

Chapitre 5

Hyperparameters Tuning

1.	Principes	133
1.1	Motivations	134
1.2	Fonctionnement	134
1.3	Méthodes disponibles	135
1.3.1	Brute force	135
1.3.2	HalvingGridSearch	135
1.3.3	Le hasard	136
1.3.4	Approche de type substitut	136
2.	Hyperparamètres	137
2.1	Définition	137
2.2	Paramètres structurels	138
2.2.1	Nombre d'estimateurs	138
2.2.2	Profondeur maximale	139
2.3	Paramètres d'apprentissage	142
2.3.1	Impact des paramètres d'apprentissage	142
2.3.2	Taux d'apprentissage	143
2.3.3	Paramètre de régularisation gamma	146
2.3.4	Paramètre de régularisation de type L2 : lambda	149
2.3.5	Paramètre de régularisation de type L1 : alpha	153
3.	Bibliothèques existantes pour l'optimisation des hyperparamètres	158
3.1	Scikit-learn	159
3.2	Scikit-optimize	160

6 Gradient Boosting

Exploitez les arbres de décision pour le Machine Learning

3.3	SMAC	161
3.4	Ray Tune	162
4.	Optimisation d'XGBoost avec XGBoost	164
4.1	Objectif	164
4.2	Principe général	164
4.3	Espace de configuration et échantillonnage	165
4.4	Optimiseur	167
4.5	Application au Boston dataset	170
5.	AutoML et Hyperparameters Tuning	174

Chapitre 6

Du bon usage des fonctions objectifs

1.	Raison d'être des fonctions objectifs	175
2.	Importance des fonctions objectifs	176
3.	Objectifs usuels	177
3.1	Classification	177
3.1.1	Fonction logistique	178
3.1.2	Soft Max	180
3.2	Régression	180
3.2.1	Squared error	181
3.2.2	Pseudo Huber Loss	182
4.	Objectifs régularisés	183
4.1	Logcosh	184
4.2	Quantile regression	186
5.	Objectifs personnalisés	187
5.1	XGBoost et régression par quantile	187
5.2	Intervalle de confiance	189

Chapitre 7
Gradient Boosting pour les séries temporelles

- 1. Séries temporelles 193
 - 1.1 Définition 193
 - 1.2 Spécificités. 194
 - 1.2.1 Intrinsèque 194
 - 1.2.2 Traitement 195
- 2. Séries temporelles et arbres de décision. 195
 - 2.1 Extrapolation et arbres de décision 195
 - 2.2 Démonstration 196
 - 2.3 Détournement des fonctions objectifs 199
 - 2.4 Alternatives. 199
- 3. Capture des caractéristiques temporelles 200
 - 3.1 Exemple simple de périodicité 200
 - 3.2 Capture manuelle 203
 - 3.2.1 Principe 203
 - 3.2.2 Exemple. 204
 - 3.2.3 Bestiaire des features calculables 207
 - 3.3 Extraction automatique 208
 - 3.3.1 Application à un cas simple 209
 - 3.3.2 Particularités des features de type caractéristiques temporelles 213
 - 3.4 Approche multimodèle. 213
- 4. Construction des datasets d'entraînement et de test. 214
 - 4.1 Cloisonnement 214
 - 4.2 Fuite de données 214
 - 4.3 Respect de la temporalité. 214

8 --- Gradient Boosting

Exploitez les arbres de décision pour le Machine Learning

Chapitre 8 XGBoost, LightGBM ou CatBoost ?

1. Pourquoi choisir ?	215
1.1 Motivations	215
1.2 De la théorie à la pratique	216
1.2.1 Sélection de la feature et du critère de décision	216
1.2.2 Support des features catégorielles	217
1.2.3 Parallélisation et distribution	217
1.2.4 Hyperparamètres	217
1.2.5 Choix et customisation de la fonction objectif	218
1.2.6 Interfaçage	218
1.2.7 Simplicité d'usage	218
1.2.8 Langage d'implémentation	218
1.2.9 Support des valeurs manquantes	219
1.3 Convictions	219
2. XGBoost	219
2.1 Partis pris	220
2.1.1 Sélection de la feature et du critère de décision	220
2.1.2 Support des features catégorielles	224
2.1.3 Parallélisation et distribution	225
2.1.4 Hyperparamètres	226
2.1.5 Choix et customisation de la fonction objectif	226
2.1.6 Interfaçage	226
2.1.7 Simplicité d'usage	226
2.1.8 Langage d'implémentation	227
2.1.9 Support des valeurs manquantes	227
2.2 Exemple	227
3. LightGBM	230
3.1 Partis pris	230
3.1.1 Sélection de la feature et du critère de décision	230
3.1.2 Support des features catégorielles	231
3.1.3 Parallélisation et distribution	231

- 3.1.4 Hyperparamètres 232
- 3.1.5 Choix et customisation de la fonction objectif 232
- 3.1.6 Interfaçage..... 232
- 3.1.7 Simplicité d’usage..... 232
- 3.1.8 Langage d’implémentation 232
- 3.1.9 Support des valeurs manquantes..... 233
- 3.2 Exemple..... 233
- 4. CatBoost 235
 - 4.1 Partis pris..... 235
 - 4.1.1 Sélection de la feature et du critère de décision 235
 - 4.1.2 Support des features catégorielles 235
 - 4.1.3 Parallélisation et distribution..... 235
 - 4.1.4 Hyperparamètres 236
 - 4.1.5 Choix et customisation de la fonction objectif 236
 - 4.1.6 Interfaçage..... 236
 - 4.1.7 Simplicité d’usage..... 236
 - 4.1.8 Langage d’implémentation 236
 - 4.1.9 Support des valeurs manquantes..... 236
 - 4.2 Exemple..... 237

Chapitre 9

Approche multimodèle

- 1. Modèle global 239
 - 1.1 Principe 239
 - 1.2 Motivations..... 240
- 2. Modèle local 240
 - 2.1 Principe 240
 - 2.2 Motivations..... 241
 - 2.3 Cas d’applications..... 241
 - 2.4 Identification des résolutions 242
 - 2.5 Limitations 242

10 **Gradient Boosting**

Exploitez les arbres de décision pour le Machine Learning

3. Modèle optimisé global	243
3.1 Principe	243
3.2 Motivations	243
3.3 Fonctionnement	244

Chapitre 10

Pour aller plus loin

1. Un large aperçu	249
2. Que retenir ?	250
3. Un domaine en constante évolution	250
4. Perspectives	251
4.1 Couplage avec des réseaux de neurones	251
4.2 Programmation différentiable	252
4.3 Modèles hétérogènes	252

Index	253
-----------------	-----



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.

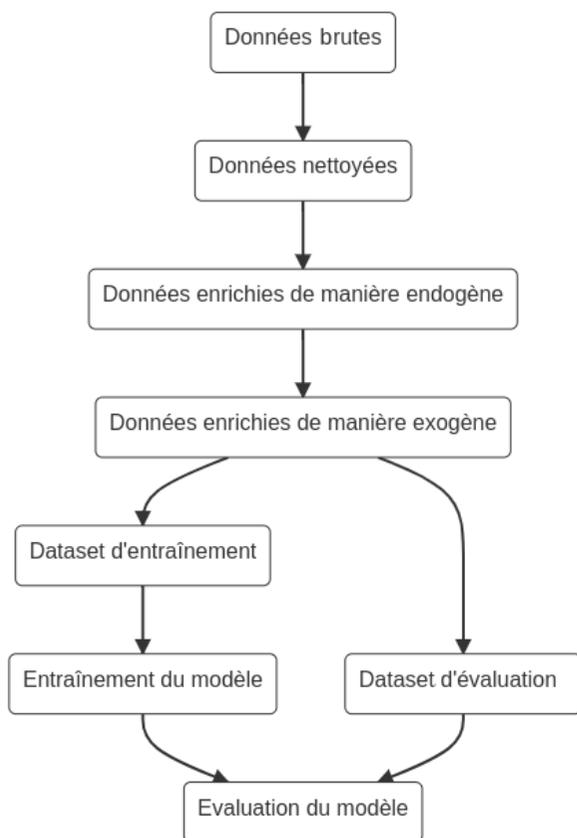
La troisième section présentera rapidement les principes de l'entraînement en lui-même, tandis que la quatrième plongera dans le détail des différentes métriques utilisées pour évaluer la qualité d'un modèle de régression ou de classification.

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 $2 \times 2 = 4$ 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 bibliothèques 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.