Gradient Boosting XGBoost

Francesco Pugliese, PhD

neural1977@gmail.com

Boosting

- ✓ Nel Machine Learning, la **Tecnica di Boosting** è un **meta-algoritmo ensemble** per ridurre principalmente il **bias** e la **varianza** nel Supervised Learning.

- ✓ **Bias** in Machine Learning: si tratta di un errore sistematico che si verifica a causa di assunzioni non corrette nei processi di Machine Learning (per esempio training set biased, ecc.). In sostanza, il bias rappresenta l'errore tra la predizione del modello medio e il ground truth e descrive quanto bene il modello identifica gli elementi del training data set. I più alti bias corrispondono a queste situazioni: a) Fallimento nel catturare i trend appropriati nei dati; b) Underfitting; c) Elevato tasso di errore.

- ✓ **Varianza** si riferisce all'enorme variazine del modello quando usa differenti porzioni del training data set. In altre parole, la varianza è la variabilità della predizione del modello. In genera, la varianza emerge in modelli altamente complessi con un elevato numero di feature:

    1. Modelli con un elevato bias avranno una bassa varianza.
    2. Models con un'alta varianza avranno un basso bias.

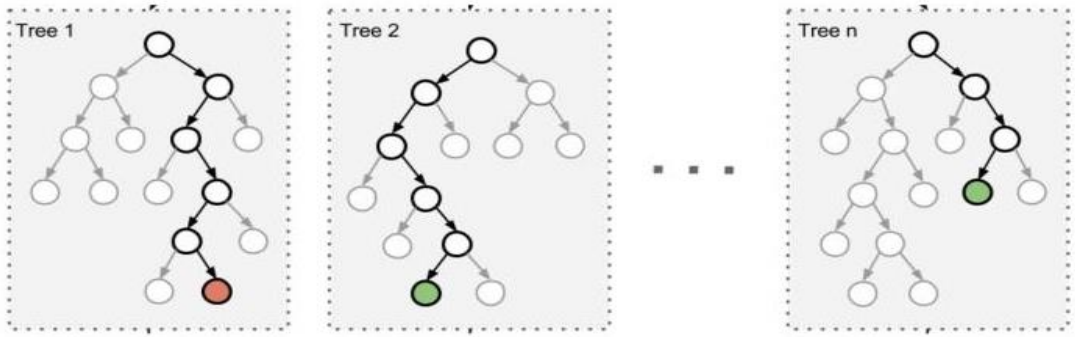
- ✓ Tutti questi elementi contribuiscono alla **flessibilità** del modello. Per esempio, un modello che non riesce a catturare bene un dataset con un elevato bias creerà un modello poco flessibile con una bassa varianza che risulterà in un modello di machine learning a bassa varianza. Le caratteristiche dell'alta varianza sono: a) Rumore nel data set, Overfitting, Modelli Complessi

Gradient boosting

- ✓ **Meta-learning** si riferisce a quella categoria di algoritmi di machine learning che apprendono dall'output di altri algoritmi di machine learning. Gli algoritmi di Meta-learning tipicamente fanno riferimento agli algoritmi di ensemble learning algorithms (come lo stacking) che apprendono come combinare le predizioni dai membri dell'ensemble.

- ✓ Il **Boosting** è basato sulla questione posta da **Kearns** e **Valiant**: "Può un insieme di learner deboli creare un singolo learner forte?" Un learner debole è definito per essere un classificatore che è solo leggermente correlato con la classificazione vera (esso comunque può etichettare gli esempi meglio di un decisore casual). Al contrario, uno learner forte è un classificatore che è ben correlato con la classificazione vera.

- ✓ Il **Gradient Tree Boosting** o **Gradient Boosted Decision Trees (GBDT)** è una generalizzazione del concetto di boosting ad arbitrarie loss function differenziabili. **GBDT** è una procedura "prêt-à-porter" accurata ed efficace che può essere usata sia per problem di regression che per problem di classificazione in una varietà di aree che includono il Web search ranking e l'ecologia.



Gradient boosting

- ✓ **GBM** è un algoritmo di boosting usato quando abbiamo a che fare con un enorme quantità di dati per fare predizioni con un alto potere predittivo.
- ✓ Il **Boosting** è concretamente un ensemble di algoritmi di apprendimento che combina la predizione di parecchi stimatori di base al fine di migliorare la robustezza rispetto ad un singolo stimatore.
- ✓ Esso combina multipli predittori deboli o media questi predittori per costruire un predittore **forte.**
- ✓ Questi algoritmi di boosting lavorano sempre molto bene in competizioni di data science su **Kaggle**, AV Hackathon, CrowdAnalytix, ecc.
- ✓ Sono generalmente usati come **baseline** per le competizioni su Kaggle, per ottenere l'estremo inferior dell'accuratezza che possiamo raggiungere e poter applicare poi architetture di **Deep Learning** più complesse e accurate che provano ad incrementare le prestazioni ottenuti con gli algoritmi di boosting.
- ✓ Il Boosting potrebbe portare ad avere over fitting se il campione dei dati è molto piccolo.

Qual è la differenza principale tra Random Forest e Gradient boosting?

- ✓ Come **Random Forest**, **Gradient boosting** è un insieme di alberi di decisione.

- ✓ Le due principali differenze sono:

    1. Il modo in cui gli alberi sono costruiti: random forest costruisce ciascun albero indipendentemente tra loro mentre il gradient boosting costruisce un albero alla volta. Questo **modello additivo** (ensemble) lavora in una modalità "alimentata in avanti", introducendo un nuovo weak learner per migliorare le discordanze nei weak learner esistenti.

    2. Il modo in cui sono combinati i risultati: random forest combina i risultati alla fine del processo (calcolando una media o per regole di "elezione a maggioranza") mentre il gradient boosting combina i risultati lungo la strada del processo di training.

- ✓ Se intendi tunare i parametri, il gradient boosting può risultare di possedere migliori performance di un random forest. Tuttavia, il gradient boosting potrebbe non essere la scelta migliore nel caso hai un sacco di **rumore** nei dai, in quanto può generare **overfitting**. Inoltre questi algoritmi tendon ad essere anche più difficili da tunare di un random forest.

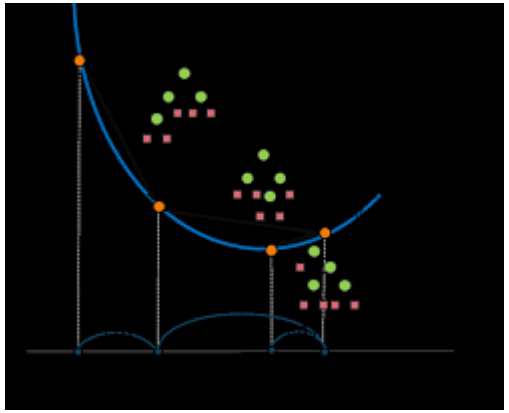
- ✓ Random forest e gradient boosting eccellono ciascuno in aree molto differenti. Random forests esegue bene la multi-class object detection e bioinformatica, che sono aree che tendono ad avere un enorme rumore statistico. Gradient Boosting lavora molto bene quando si hanno dati sbilanciati in **"real time risk assessment"**.

Gradient boosting

- ✓ In Gradient Boosting, ciascun predittore cerca di migliorare il suo predecessore riducendo gli errori. Ma la più interessante intuizione del Gradient Boosting è che invece di fittare un predittore sui dati a ciascuna iterazione, esso fitta un nuovo predittore sugli **errori residui** prodotti da un predittore precedente. Ora esaminiamo step by step come **Gradient Boosting Classification** lavora**.**

- ✓ Gradient boosting implica tre elementi:

1) **Una loss function che debba essere ottimizzata**: La loss function da usare dipende dal tipo di problema che si intende risolvere. La regression potrebbe usare un errore quadratico medio e la classificazione potrebbe usare una loss logaritmica come per esempio la cross-entropy. Un beneficio del gradient boosting è che non si deve utilizzare un nuovo algoritmo di boosting per ciascuna lost function che si intende utilizzare. Invece, si tratta di un framework generale con qualsiasi loss differenziabile che può essere usata allo scopo.



Gradient boosting

2) Un weak learner per effettuare predizioni: Gli alberi di Decisione sono usati come un weak learner nel gradient boosting. Nello specifico, gli alberi di decisione sono usati in modo tale che i valori reali per lo split e i valori di output possano essere combinati insieme, permettendo agli output modelli seguenti di essere aggiunti e "correggere" i residui nelle predizioni. Gli alberi sono costruiti in modalità **greedy**, scegliendo i migliori punti di split sulla base dei punteggi di misure di purezza come Gini per minimizzare la loss. All'inizio sono costruiti alberi di decisione molto corti, ossia poco profondi, che hanno subito solo un singolo split, chiamato **"decision stump"**. Alberi di decisione più grandi possono essere usati generalmento con livelli da 4 a 8. E' molto commune vincolare i weak learner in modi specifici, come un certo numero di strati Massimo, o un Massimo numero di nodi o split o nodi foglia. Questo assicura che i weak learner rimangano deboli, in modo da costruire un learner forte attraverso il paradigma greedy.

3) Un modello additivo per aggiungere i weak learner allo scopo di minimizzare la loss: Gli alberi vengono aggiunti uno alla volta, e gli alberi già esistenti nel modello non vengono cambiati. Una procedura di **discesa del gradiente** viene usata per minimizzare la loss quando si aggiungono gli alberi. Tradizionalmente, il gradient descent viene usato per minimizzare un insieme di parametri, come i coefficienti di una equazione di regressione o i pesi di una rete neurale. Dopo aver calcolato l'errore o loss, i pesi vengono aggiornati per minimizzare l'errore. Invece dei parametri, abbiamo sottomodelli che sono i weak learner, e nello specifico alberi di decisione. Dopo aver calcolato la loss, per eseguire la procedura di gradient descent, dobbiamo aggiungere un albero al modello che riduca la loss (ovvero che segue il gradiente). Facciamo questo parametrizzando l'albero, poi modificando i parametri dell'albero e muovendosi nella giusta direzione riducendo la loss residua. In generale, questo approccio è chiamato **"gradient descent funzionale"** o **gradient descent con funzioni**.

Passi del Gradient boosting

- ✓ **Passo 1** : Assumi che la media sia la predizione di tutte le variabili.
- ✓ **Passo 2** : Calcola gli errori di ciascuna osservazione con discostamento dalla media (ultima predizione).
- ✓ **Passo 3** : Trova la variabile che può splittare gli errori perfettamente e trovare il miglior valore per lo split. Questa diventa l'ultima predizione.
- ✓ **Passo 4** : Calcola gli errori di ciascuna osservazione dalla media di entrambi i lati dello split (ultima predizione).
- ✓ **Passo 5** : Ripeti il passo 3 e 4 fino a che la funzione obiettivo si massimizzi / minimizzi.
- ✓ **Passo 6** : Prendi una media pesata di tutti i classificatore per giungere al modello finale.

XGBoost

- ✓ **XGBoost** è un algoritmo di **Gradient Boosting** che usa gli alberi di decisione come i suoi predittori "deboli" predictors. Oltre questo, la sua implementazione è specificatamente reingegnerizzata per ottenere prestazioni ottimali e velocità.

- ✓ **XGBoost** ha avuto risultati ottimali anche per dati strutturati e tabulari. Per quanto riguarda dati non-strutturati come immagini o testi, certamente le reti neurali deep sono un'opzione migliore.

- ✓ Quando si usa il **Gradient Boosting** per la regressione, i weak learner sono alberi di regressione, e ciascun albero di regressione mappa un punto dei dati di input verso uno contenuto nelle sue foglie che contengono uno score continuo.

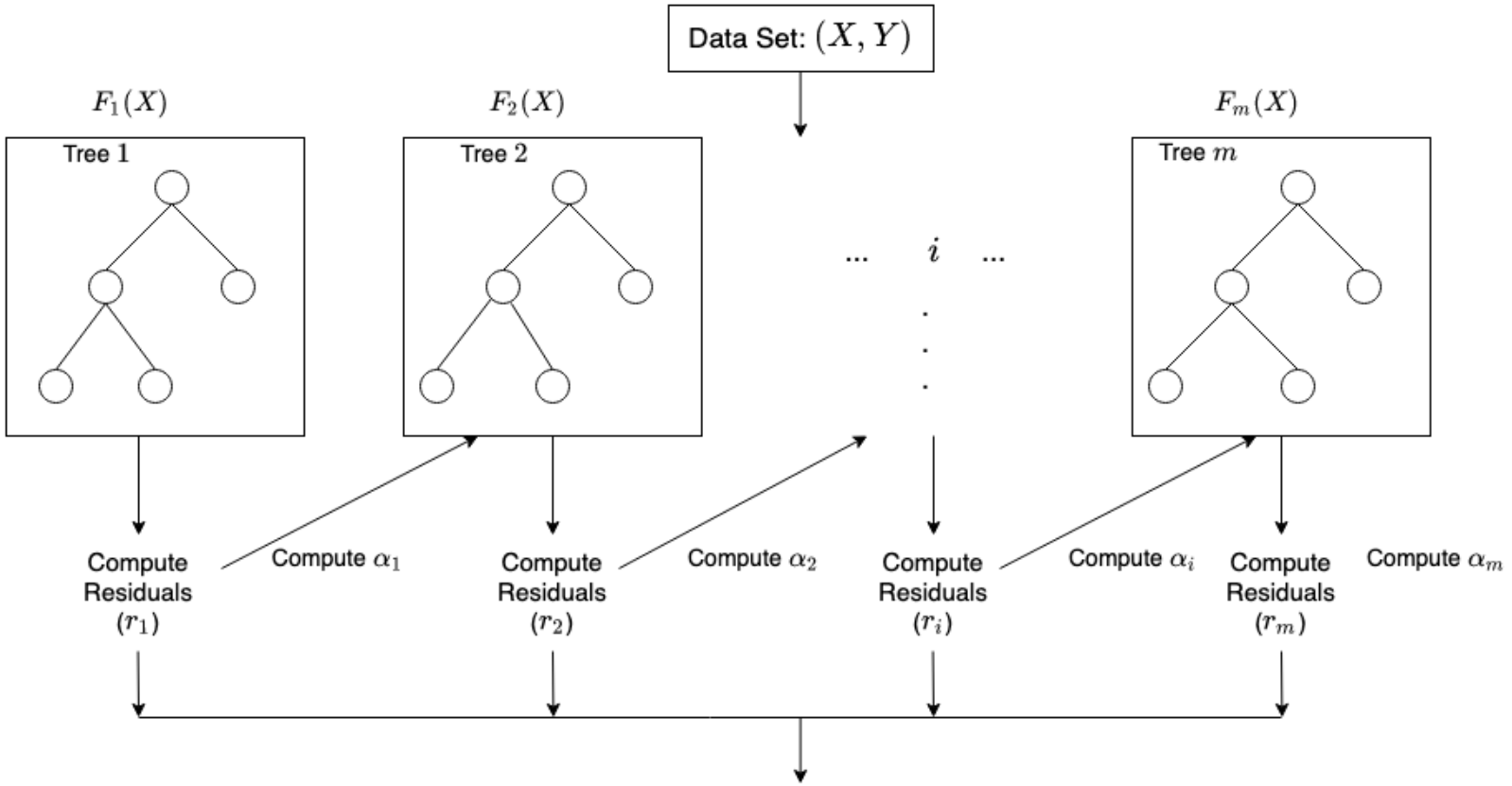
- ✓ **XGBoost** minimizza una funzione obiettivo regolarizzata (L1 e L2) che combina una loss function convessa (basata sulla differenza tra il valore predetto e il target reale) e un termine di penalità per la complessità del modello (in altre parole, le funzioni dell'albero di regressione).

- ✓ Il **training** procede iterativamente, aggiunge nuovi alberi che predicono gli errori e i residui degli alberi precedenti che sono poi combinati insieme con i precedenti alberi per ottenere la predizione finale. Questo è chiamato **gradient boosting** dal momento che usa un algoritmo di gradient descent per minimizzare la loss quando si aggiungono nuovi modelli.

XGBoost

- ✓ L' XGBoost ha un potere predittivo immensamente alto il che lo rende la migliore scelta per un alta accuratezza.
- ✓ Possiede sia algoritmi di apprendimento di alberi che di modelli lineari, rendendolo 10 volte più veloce delle tecniche di gradient boosing esistenti.
- ✓ Le librerie di supporto includono funzioni obiettivo, regressione, classificazione e ranking.
- ✓ Uno degli aspetti più interessanti del **XGBoost** è che è anche chiamato tecnica di boosting regolarizzato. Questo aiuta a ridurre l'overfitting nei modelli.
- ✓ Ha un supporto massivo per una gamma di linguaggi come Scala, Java, R, Python, Julia e C++.
- ✓ Supporta un addestramento distribuito e diffuso su molte macchine che incorporano GCE, AWS, Azure e cluster Yarn.
- ✓ **XGBoost** può anche essere integrato con Spark, Flink e altri sistemi di cloud dataflow con una cross validation built in a ciascun iterazione del processo di boosting.

XGBoost



$$F_m(X) = F_{m-1}(X) + \alpha_m h_m(X, r_{m-1}),$$

where α_i, and r_i are the regularization parameters and residuals computed with the i^{th} tree respectfully, and h_i is a function that is trained to predict residuals, r_i using X for the i^{th} tree. To compute α_i we use the residuals computed, r_i and compute the following: $arg \min_{\alpha} = \sum_{i=1}^{m} L(Y_i, F_{i-1}(X_i) + \alpha h_i(X_i, r_{i-1}))$ where $L(Y, F(X))$ is a differentiable loss function.

Applicazioni di XGBoost e Gradient Boosting

- ✓ Compiti di Regressione.
- ✓ Classificazione.
- ✓ Ranking.
- ✓ Problemi di predizione definiti dall'Utente.

Bibliografia

- ✓ https://towardsdatascience.com/top-10-algorithms-for-machine-learning-beginners-149374935f3c

Chen, T., He, T., Benesty, M., Khotilovich, V., Tang, Y., Cho, H., & Chen, K. (2015). Xgboost: extreme gradient boosting. *R package version 0.4-2*, *1*(4), 1-4.

Francesco **Pugliese**