XGBoost for Regression

Sjoerd De Witte

November 2022

1 Algemeen

XGBoost valt onder de categorie boosting techniques in Ensambling Learning, het gebruik van meerdere kleine voorspellers om iets te voorspellen, en staat voor: extreme gradient boost. XGBoost maakt net als RandomForest gebruik van meerdere regression trees en kan gebruikt worden voor regressie, maar het kan even goed gebruikt worden voor classificatie. Waar RandomForest vanuit dezelfde trainingsdata steeds parallel van elkaar regression trees genereert, doet XgBoost het in een reeks. Dit betekent dat elke nieuwe tree gebasseerd is op informatie uit de vorige tree. Dit is de reden dat over het algemeen XGBoost beter presteert. Verder heeft het nog een aantal dingen geoptimaliseerd voor computersnelheid, maar dat valt nu buiten de scoop. Nadelen van XgBoost zijn dat het sensitief is voor sterk afwijkende datapunten al is dit met de juiste hyperparameters goed op te lossen.

2 XGBoost stap voor stap

XGBoost heeft drie belangrijke hyperparameters: λ voor regularization, γ voor termination en η voor de learning rate. Hier later meer over.

Ter verduidelijking van de werking van XGBoost volgen we een eenvoudig voorbeeld, het voorspellen van de effectiviteit van een geneesmiddel op basis van de grootte van de dosis.

dosis(mg)	effectiviteit
10	-10
20	7
25	8
35	-7

Er wordt gestart met het maken van een eerste voorspelling die vaak het gemiddelde is van alle y-waarde, in dit geval 0.5(in principe maakt het niet uit wat je doet maar het gaat wel sneller als je begint met een 'normale' voorspelling).

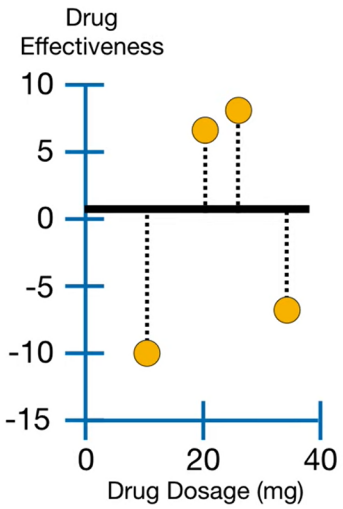


Figure 1: Voorspelling effectiviteit voor elke dosis is gelijk aan 0.5

Vervolgens worden de residuals bepaald tussen de voorspelde waarde en de echte waarde.

dosis(mg)	effectiviteit	residuals
10	-10	-10.5
20	7	6.5
25	8	7.5
35	-7	-7.5

Met deze voorspelling kan er als volgt worden bepaald wat de simularity score(S.S) van elke leaf in de tree is.

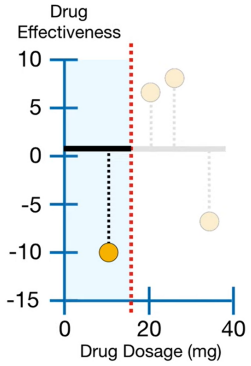
$$\text{S.S} = \frac{(\text{sum of residuals})^2}{\text{number of residuals} + \lambda}$$

Waarbij de residual wordt berekent door de voorspelde waarde min de echte waarde. En λ is regularization hyperparameter, die kan worden getunend om overfitting tegen te gaan.

In dit voorbeeld is de S.S van de leaf gelijk aan 4(met λ=0) en bestaat de tree maar uit één leaf.

Met de simularity score kan worden bepaald hoe goed een bepaalde leaf is in het scheiden van voorspellingen. Voor de echte wiskunde is een apart hoofdstuk aan het eind, maar intuïtief moeten een aantal dingen duidelijk zijn. Ten eerste is het de bedoeling van een leaf om waardes te bevatten die zoveel mogelijk op elkaar lijken, want dan is er dus goed onderscheid gemaakt tussen verschillende uitkomsten. Omdat we eerst de som van de residuals nemen en dan pas kwadrateren is het logisch dat een zo hoog mogelijk simularity score wenselijk is. Verder laat ik λ nu nog even buitenbeschouwing, deze kan ook nul zijn.

Vervolgens wordt er een tree gemaakt waarbij tussen elke punten een treshold word gekozen waar de regression tree splitst. De tree die uiteindelijk het beste splits word gekozen als de nieuwe tree.



(a) Choosing threshold as an average dosage between the first dosages.

(b) The regression tree as an outcome of choosing this threshold.

Figure 2: After checking all thresholds the tree with the highest gain is chosen.

Wat de beste regression tree is wordt bepaald door de gain van de Tree.

$$\text{Gain} = \text{sum S.S of all leafs} - \text{S.S of the root}$$

.

De linkerleaf heeft hier een simularity score van 110.25 en de rechterleaf van 14.08 dat betekent dat de Gain van de tree gelijk is aan 120.33. Dit wordt ook gedaan voor de threshold 22.5mg waar de gain gelijk is aan 4 en de threshold 30mg waar de gain gelijk is aan 56.33. De beste splitter is dus 15mg, de rechterleaf kan nogmaals worden gesplit dit gaat op dezelfde manier alleen hoeven er nu nog maar 2 thresholds worden gecheckt. Met threshold is 22.5mg de Gain is gelijk aan 28.17 en met threshold is 30mg wordt er een Gain verkregen van 140.17.

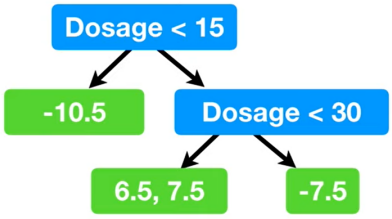


Figure 3: Regression Tree na tweede splitter

Als nu ook nog de laatse leaf wordt gesplitst zal deze tree waarschijnlijk heel erg overfitten, daarom wordt de max-depth nu gehouden op twee. Dit is alleen niet de enige manier om overfitting tegen te gaan. De hyperparameter gamma zorgt ervoor dat een branch uit de tree wordt verwijderd op het moment dat de Gain - Gamma < 0. Het start te kijken bij de onderste branch als deze waarde kleiner is dan nul zal het naar de volgende branch kijken, zodoende kan het voorkomen dat de hele tree wordt verwijderd. Als er een branch is waar de waarde groter is dan nul blijft de tree zoals hij dan is. Merk op dat als labda groter wordt de kans ook groter is dan een branch wordt verwijderd. Verder kan het ook zo zijn dat als gamma=0 de gain - gamma <0 nog steeds waar is. Bijvoorbeld als de laatste leaf in dit voorbeeld word gesplitst, dus voor termination is gamma niet per se nodig.

Tot slot moet er ook nog een output value worden berekend om een nieuwe voorspelling te maken. Deze wordt per leaf als volgt berekent.

Ook hier heeft lambda weer een functie in de regularization, als lambda groter wordt heeft een individuele observatie minder invloed op de alghele voorspelling. Van links naar rechts krijgen we de volgende Output Values: -10.5, 7, -7.5. De nieuwe Voorspelling = oude voorspelling + eta * Output Values. Na deze eerste tree zijn voorspelling de residuals als volgt.

dosis(mg)	effectiviteit	voorspelling	residuals
10	-10	-2.65	-7.35
20	7	2.6	4.4
25	8	2.6	5.4
35	-7	-1.75	5.25

Te zien is dat alle voorspellingen een klein stukje de goede richting in zijn verschoven. Een volgende tree zal dit waarschijnlijk weer doen, totdat de Gains van de trees zo klein worden dat ze constant worden afgebroken.