Grobe Idee/Richtung

Christian

March 4, 2021

Randomisierung - wieso wird nicht optimiert

Bezüglich Ihrer Frage wieso man eigentlich die Variablen immer zufällig auswählt ist mein Verständnis bis jetzt zumindest folgendes:

- Ein einzelner decision tree hat eine ziemlich hohe Varianz. Um den MSE zu reduzieren, greift man auf random forests zurück, die die Varianz ordentlich reduzieren.
- Damit das funktioniert, dürfen die einzelnen trees nicht wirklich korrelieren. Wenn sie alle genau gleich wären, hätte man im Endeffekt wieder nur einen einzelnen tree. Deswegen versucht man die trees in einem forest unterschiedlich von einander zu machen und das erreicht man über randomization.
- Im Groben gibt es drei Arten von randomization:
    1. Es wird nur ein random subset der data points für die Konstruktion eines trees genommen
    2. Es wird bei jeder Node zufällig nur ein subset an Variablen auserwählt, an denen gesplittet werden kann
    3. Der split point gegeben einer ausgewählten Variable kann zufällig sein

Dabei kommen ganz unterschiedlich Dinge vor. Manche paper machen nur 2.) und 3.), andere wiederum machen 1.) und 2.) und wählen dann den Median in der ausgewählten split dimension. Andere wiederum machen 1.) und 2.) und optimieren dann, um den geeigneten split point zu finden. Oder auch 1.) und 2.) und wählt dann gegeben einer split dimension zufällig m split points aus, und sucht nur unter denen nach dem optimalen split. Kurzum - irgendwie macht gefühlt jeder so ein bisschen das, wonach ihm ist.

Konsistenz

Um Konsistenz zu zeigen, müssen aber zwei Dinge halten wenn $n \to \infty$:

- Der Durchmesser der einzelnen Zellen/Partionierungen am Ende des decision trees muss gegen 0 gehen (damit der Schätzer auch wirklich lokal ist)
- Die Anzahl an Zellen ist verhältnismäßig klein gegenüber n (soll gewährleisten, dass die Wahrscheinlichkeit hoch ist, dass viele Beobachtungen in einer Zelle liegen)

Alle Studien, die bis sich bis jetzt mit dieser Thematik beschäftigt hatten, haben einen leicht abgewandelten Algorithmus benutzt. Soweit ich es bis jetzt durschaue, liegt das Problem am klassichen Algorithmus von Breimann (1984) daran, dass er sehr gerne am Anfang oder Ende einer Dimension den Split setzt, wenn man den split point via Optimierung suchen möchte (end-cut-preference). Sprich er tendiert dazu, bei Ausreißern den split zu machen. Biau & Scornet "A random forest guided tour" (2016) und Athey & Wager: "Estimation and Inference of Heterogeneous Treatment Effects using Random Forests", Appendix B (2018) gehen jeweils kruz darauf ein.

Bei den meisten papern, die Konsistenz zeigen, nehmen die Autoren an, dass bei jedem split mindestens ein Anteil α auf beiden Seiten des splits bleiben muss. Mit dieser Annahme umgehen sie im Endeffekt die end-cut-preference und erzwingen den split irgendwo in der Mitte. Die einzigen, die auf diese Annahme verzichten sind Biau et al. "Consistency of Random Forests" (2015). Die können Konsistenz allerdings nur für ein additives Modell zeigen.
Ein anderes paper, Ishwaran: "The effect of splitting on random forests" (2014), zeigt die Vorteile/Nachteile von end-cut-preference auf. Vor allem bei vielen noisy variables ist die end-cut-preference insofern vorteilhaft, dass bei einer uninformativen Variable dann eher am Rand gesplittet wird. Dadurch bleibt ein Großteil des samples zusammen und wenn danach auf einer Variable mit signal gesplittet wird, ist dieser Split eben umso besser.

Meine Überlegung war jetzt folgende: Ein bisschen genauer zu verstehen, wieso bei dem additiven Modell von Biau et al. (2015) sie diese Annahme mit dem Anteil α nicht benötigen und wie die Verbindung zu mehr generellen Modellen ist, die diese Annahme aber treffen. Außerdem wird in den papern mit der Annahme, α immer gleich einer Konstanten gesetzt und dann nicht weiter betrachtet.
Auf jeden Fall hätte ich vor, mir über die Weihnachtstage ein wenig Gedanken dazu zu machen, wie dieser Anteil α sich auf die gezeigten Konsistenz-Ergebnisse auswirkt und was genau der Zusammenhang zwischen Konsistenz und der an sich gewünschten end-cut-preference mit vielen noisy variables ist.