Композиции алгоритмов для решения задачи регрессии

Тыцкий Владислав

Декабрь 2020

1. Введение

В данном отчете исследуется поведение алгоритмов Random Forest и Gradient Boosting на примере датасета стоимости квартир, где целевая переменная это цена квартиры.

Датасет был разделен на обучающую и тестирующую выборки. Объекты имеют много категориальных признаков, но было решено не использовать One Hot Encoding в виду специфики алгоритмов RF и GB — Оба алгоритма работают над деревьями, поэтому "категоризация" числовых величин(хотя на деле их необходимо считать категориальными) может происходить путем построения более глубоких деревьев. Да, GB строит неглубокие деревья, но так как каждый следующий алгоритм старается исправить ошибки уже построенной композиции, то признак тоже может "категоризироваться".[1]

2. Random Forest

Random Forest — это композиционный алгоритм машинного обучения над деревьями. Каждое дерево вносит одинаковый вклад в итоговый прогноз и строится независимо. Обычно для обучения очередного дерева используют бэггинг для генерации обучающих выборок. Это уменьшает скорреллированность алгоритмов , а значит и разброс всей композиции. Кроме того можно генерировать случайную подвыборку признаков и обучать алгоритм только на этих признаках. Такой подход тоже может уменьшать скоррелированность алгоритмов.

[1] Под категоризацией признака подразумевается то, что алгоритм "не вводит" на нем отношение порядка.

2.1. Зависимость ошибки от количества признаков

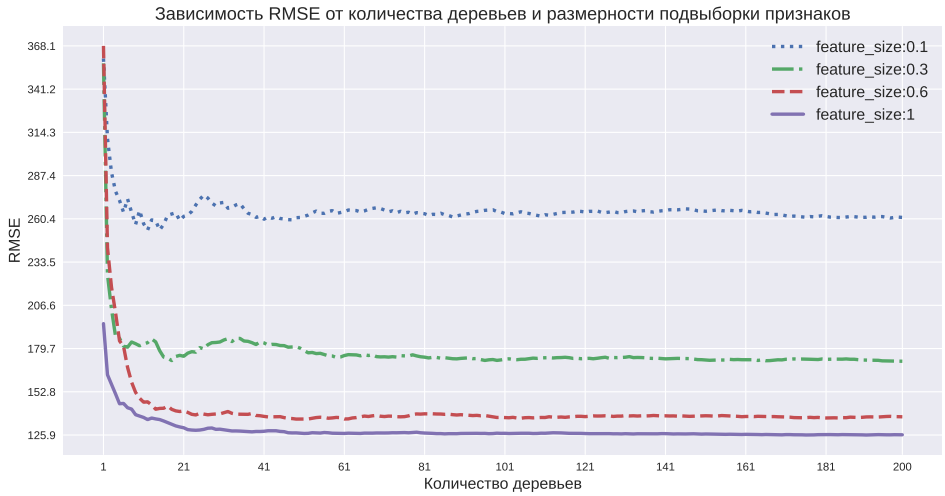

Рис. 1: Влияние размерности подвыборки на RMSE

На Рис. 1 видно, что лучшее качество получается при полном наборе признаков для любого количества деревьев в композиции. Отметим, что чем больше feature_size, тем лучше качество для данного датасета. Также можно отметить, что качество перестает расти где-то после 60-ого дерева в композиции.

2.2. Зависимость ошибки от количества и глубины деревьев

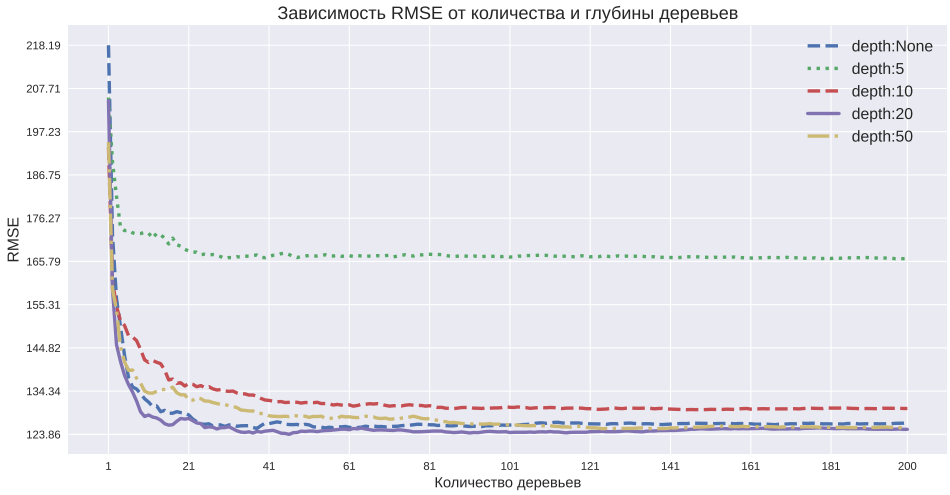

Рис. 2: Влияние глубины деревьев в композиции

Важным параметром при обучении алгоритма Random Forest является максимальная глубина деревьев в композиции. Если максимальная глубина мала, то дере-

вья недообучатся и смещение композиции будет большим, если глубина большая, то может возникнуть риск переобучения(большой разброс), но в случае RF такое происходит нечасто т.к. вся идея алгоритма в том, чтобы строить слабо коррелированные переобученные деревья и составлять из них композицию, тем самым уменьшая разброс.

На Рис. 2 видно, что деревья без ограничений на глубину[2] имеют, хоть и немного, большую ошибку, чем деревья с ограничением на глубину 20, 50. Как и ожидалось деревья с маленькой максимальной глубиной имеют самую большую ошибку.

3. Gradient Boosting

Gradient boosting — это композиционный алгоритм машинного обучения над деревьями[3]. Каждое дерево строится последовательно так, чтобы исправлять ошибки предыдущих.

3.1. Зависимость ошибки от количества признаков

Как и в случае RF при построении алгоритмов можно брать не все признаки, а лишь некоторое стохастическое подмножество признаков для каждого алгоритма.

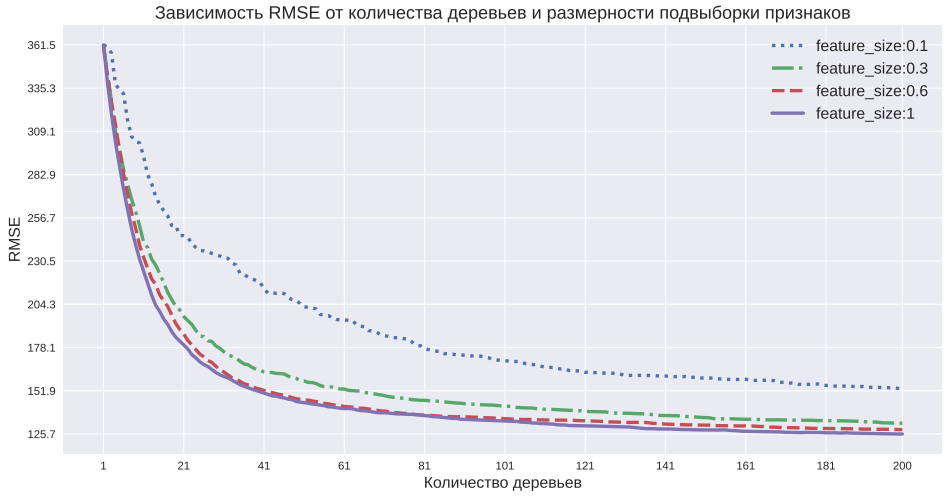

Рис. 3: Влияние размерности подвыборки на RMSE

На Рис. 1 видно, что лучшее качество получается при полном наборе признаков для любого количества деревьев в композиции. Отметим, что чем больше feature_size, тем лучше качество для данного датасета. Оптимальное количество деревьев около 150.

[2]detph: None

[3]На самом деле Gradient boosting не обязательно составляет композицию деревьев, но в чаще всего используют именно деревья.

3.2. Зависимость RMSE от количества и глубины деревьев

В отличие от RF градиентный бустинг подразумевает построение неглубоких деревьев, чтобы каждое следующее дерево исправляло ошибки предыдущих. Поэтому обычно деревья в GB имеют максимальную глубину не больше 10.

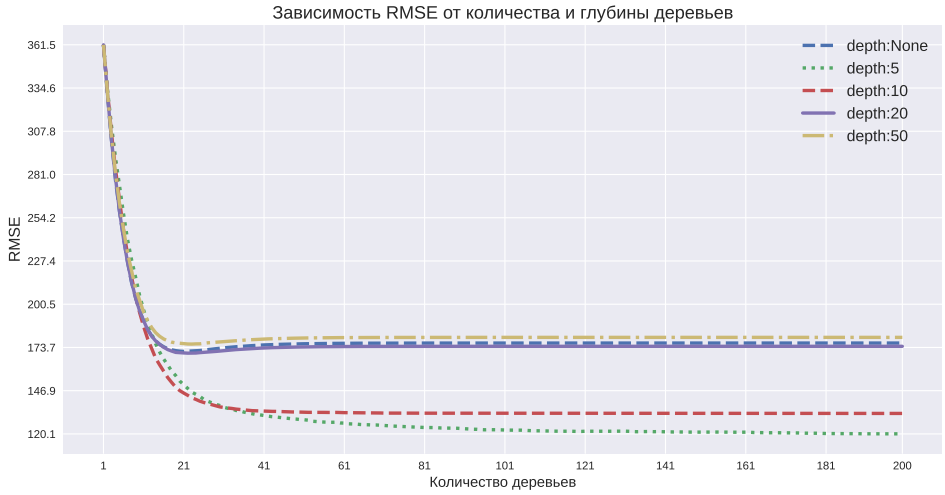

Рис. 4: Влияние глубины деревьев в композиции

На Рис. 4 видно, что лучше всего показали себя алгоритмы с маленькой глубиной. Алгоритмы с большой/неограниченной глубиной очень быстро выходят на "плато" при увеличении количества деревьев, причем RMSE заметно больше. Это объясняется тем, что при построении "мощных" деревьев в GB в какой-то момент новые деревья перестают быть значимыми, ведь с самого начала мы строили деревья, которые пытаются "объяснить все"[4].

[4] Очень ненаучно, ну и пусть!

3.3. Зависимость ошибки от скорости обучения

У градиентного бустинга есть параметр, влияющий на скорость обучения — learning rate. Если он будет слишком большим, то алгоритм рискует не сойтись, а если слишком маленьким, то время на построение алгоритма существенно увеличивается.

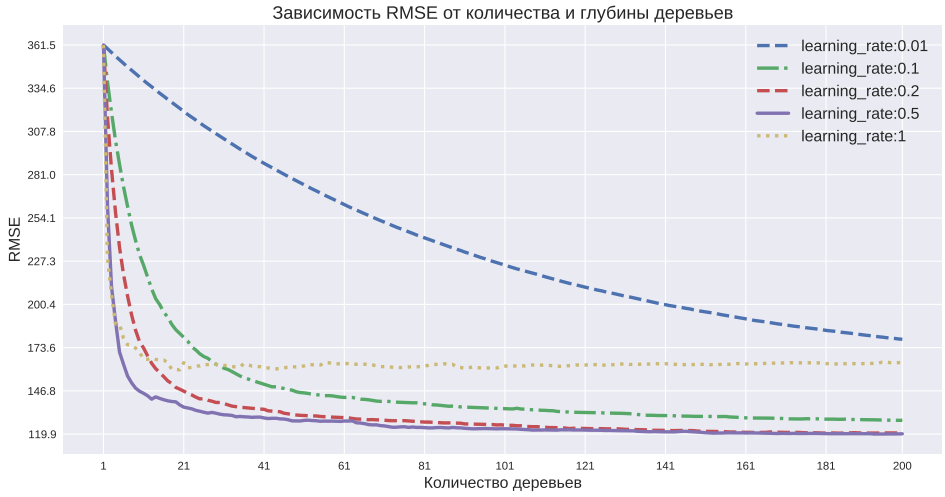

Рис. 5: Влияние глубины деревьев в композиции

Оптимальным в данном случае оказалось значение learning_rate 0.2, 0.5.

Заключение