Задание 3. Ансамбли алгоритмов. Веб-сервер. Композиции алгоритмов для решения задачи регрессии.

Отчет
о выполненном задании

студента 317 группы факультета ВМК МГУ
Воробьева Сергея Юрьевича

Декабрь, 2021

Содержание

1 Введение

В данном задании предлагается разработать алгоритмы, основанные на использовании большого числа простых «базовых» алгоритмов – ансамблей. В качестве таких алгоритмов используются `Random Forest` и `Gradient Boosting`. Предстоит подобрать оптимальные гиперпараметры для каждого алгоритма и измерить зависимость ошибки и времени обучения от различных гиперпараметров. В качестве данных выступает информация по проданной недвижимости в округе Кинг, Вашингтон. Данные собраны за период Май 2014 — Май 2015. Ставится задача предсказания цены дома по его признакам – задача регрессии.

2 Обработка данных

Так как колонок в наших данных не очень много, постараемся аккуратно преобразовывать данные, чтобы не потерять информацию. Так как наши данные предоставлены за год, преобразуем колонку с датой в номер дня в году. Таким образом каждому дню из датасета будет соответствовать уникальное число. Это может быть полезно для выявления каких-то трендов на рынке недвижимости в течение года, от которых может зависеть цена. В остальном же оставим данные как есть, измерив корреляцию между некоторыми колонками («sqft_lot» и «sqft_lot15» например) во избежание излишних данных. Преобразуем наши данные в `numpy arrays`

3 Эксперименты

3.1 Эксперимент 1

Для начала отметим что все измерения делались по кросс-валидации с 5 фолдами.

В первом эксперименте нужно выявить зависимость ошибки и время обучения градиентного бустинга в зависимости от параметров: `n_estimators`, `feature_subsample_size`, `max_depth`.
Посмотрим, как количество базовых алгоритмов влияет на ошибку и время выполнения.

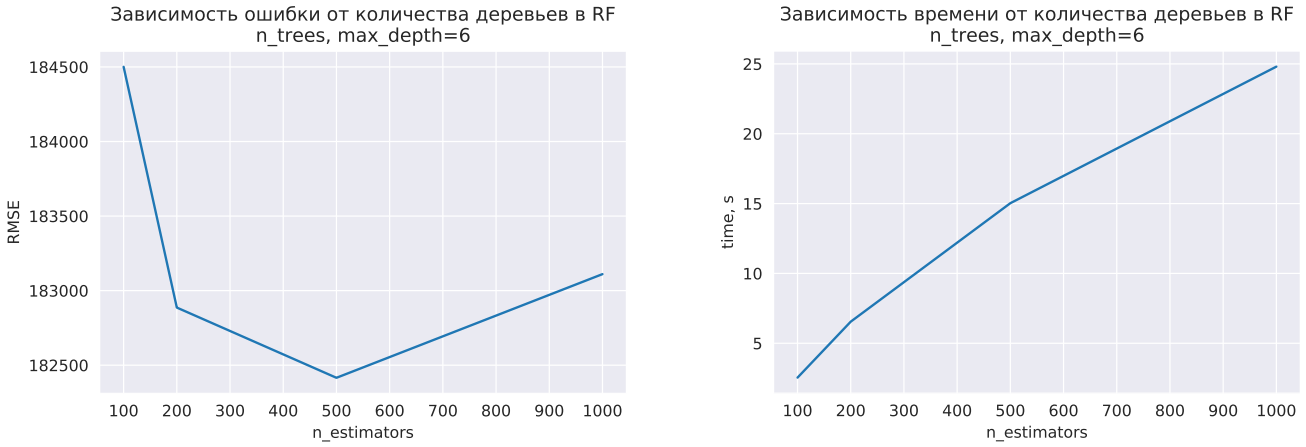

Рис. 1: Исследуем n_estimators.

Можем заметить, что, хоть ошибка и колеблется, все равно изменяется не сильно в зависимости от количества деревьев. Предположительно, это связано с тем, что более чем 100 деревьев уже достаточно для погашения разброса, и при увеличении количества деревьев заметных улучшений не происходит. Выгодно взять 100, так как обучение происходит быстрее. На графике для времени нет ничего необычного – время линейно зависит от количества деревьев. Далее, для компактности, в экспериментах не будут приводиться графики по времени. Их можно найти в приложении.

Посмотрим теперь как размер пространства признаков влияет на качество и время.

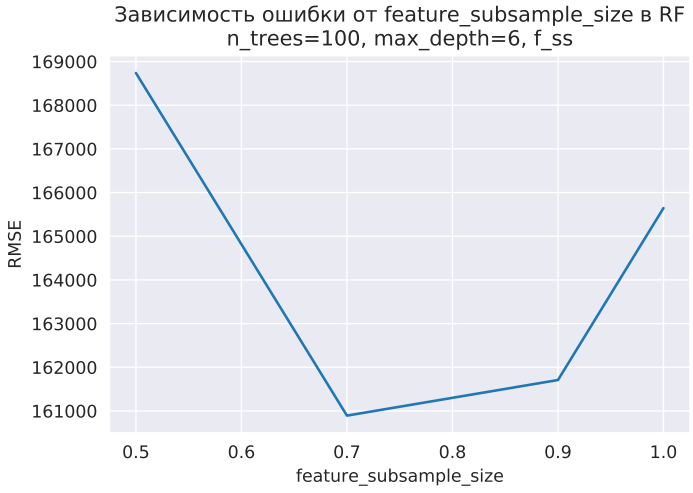

Рис. 2: Исследуем feature_subsample_size.

По графику видно, что оптимальнее всего обучать каждое дерево примерно на 70-90% признаках. Это выглядит вполне логичным. Удаляя часть признаков мы делаем деревья более разнообразными при этом оставляем достаточно признаков, чтобы делать точные прогнозы. Далее будем использовать значение `0.7` – качество у него

не сильно уступает более высоким процентам, однако при этом он быстрее обучается, что для кросс-валидации очень актуально.

Посмотрим теперь как максимальная глубина деревьев влияет на качество и время.

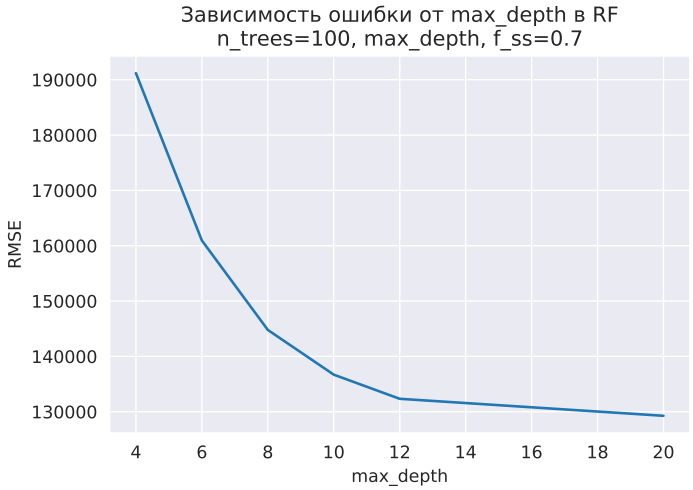

Рис. 3: Исследуем max_depth.

Заметим, что при увеличении максимальной глубины ошибка заметно снижается. Однако после значения 10-12 улучшение происходит медленно. Можно сделать вывод, что для случайного леса лучше брать более разнообразные и сложные деревья.

3.2 Эксперимент 2

В этом эксперименте нужно выявить зависимость ошибки и время обучения градиентного бустинга в зависимости от параметров: `n_estimators`, `feature_subsample_size`, `max_depth`, `learning_rate`.

Как и в превом случае посмотрим на зависимость качества и времени обучения от количесва базовых алгоритмов.

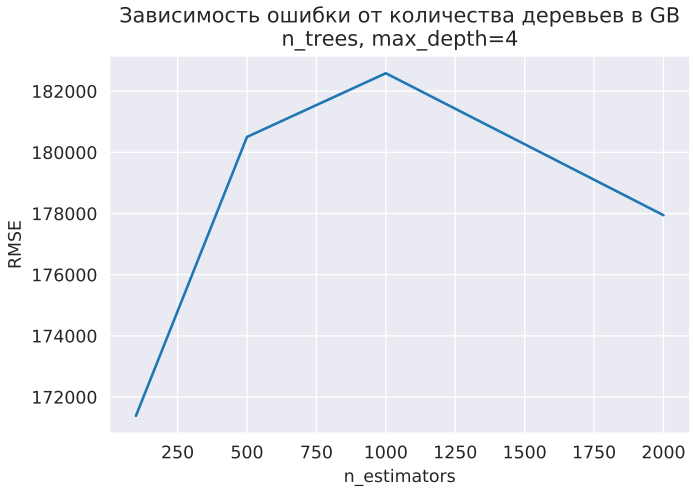

Рис. 4: Исследуем n_estimators.

В данном случае, в отличие от 1-го эксперимента, увеление количества деревьев почти гарантированно уменьшает ошибку. Это объясняется внутренним утройстром алгоритма – градиентный бустинг каждым следующим построением базового алгоритма стремится уменьшить имеющуюся ошибку.

Посмотрим теперь как размер признакого пространства влияет на качество и время.

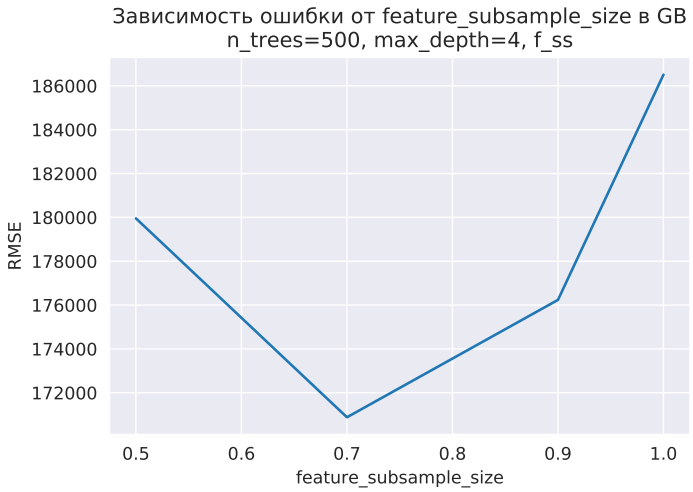

Рис. 5: Исследуем feature_subsample_size.

Здесь, как и в первом случае, выгодно оставлять 70% признаков.

Посмотрим теперь как максимальная глубина деревьев влияет на качество и время.

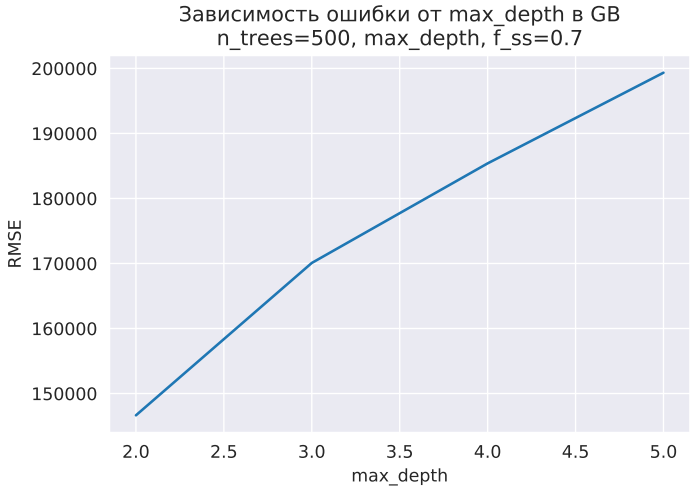

Рис. 6: Исследуем max_depth.

Здесь мы видим обратную ситуацию, нежели со случайным лесом. В случае градиентного бустинга мы заинтересованы в том, чтобы строить более простые деревья для постепенного уменьшения ошибки.

Посмотрим теперь как темп обучения влияет на ошибку и время.

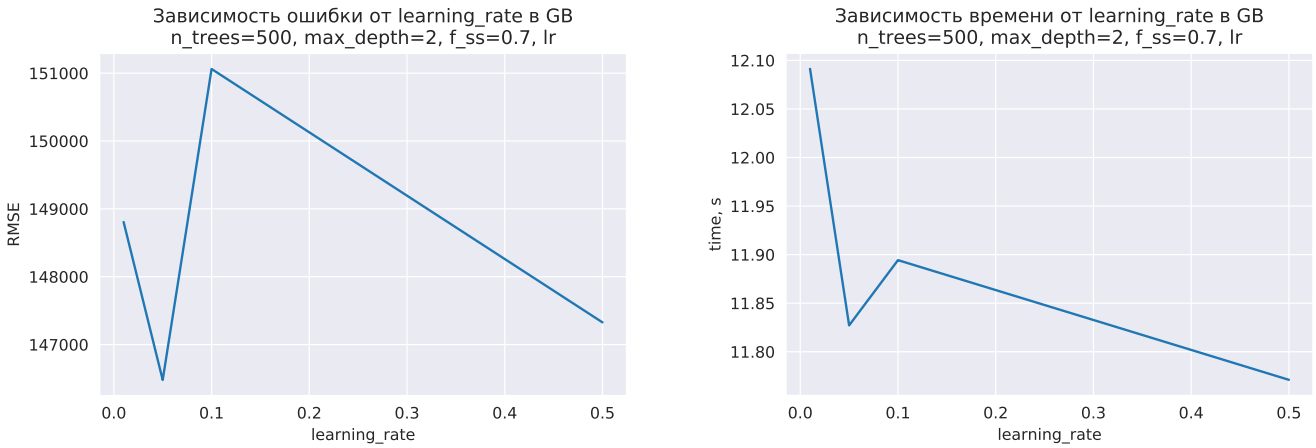

Рис. 7: Исследуем learning_rate.

Судя по графику, чем меньше темп обучения – тем лучше модель обучается. Это поисходит из-за более аккуратной настройки. Также можно отметить, что низкий темп обучения сопровождается чуть более долгим временем обучения. Имеет смысл взять значение `0.05` – при нем достигается очень хорошее качество и он практически сопостовим по времени обучения с аналогичными моделями с большим темпом обучения.

4 Вывод

В данной работе были построены алгоритмы ансамблей – градиентный бустинг и случайны лес. Рассматривалась задача предсказания цены квартиры по ее параметрам – задача регрессии. В ходе экспериментов были подобраны параметры для наших моделей, а также разобраны некоторые аспекты обучения каждого ансамбля.

5 Приложение