Московский государственный университет
имени М. В. Ломоносова
Факультет вычислительной математики и кибернетики

Отчёт по заданию № 3. Ансамбли алгоритмов. Композиции алгоритмов для решения задачи регрессии.

Выполнила:
студентка 317 группы
Гайфутдинова Ф. Х.

Москва
2021

Содержание

Введение

В задании требовалось написать собственные реализации моделей - случайный лес и градиентный бустинг над деревьями. Также пронаблюдать зависимости RMSE и времени обучения модели от её различных параметров.

Подготовка данных

В данной части производилась предобработка датасета. В нем был 1 невещественный признак 'date'. Мы разбили его на 3 - 'день', 'месяц', 'год'. Соответсвенно, расширив признаковое пространство. Из нашего датасета выделили обучающую, валидационную и тестовую выборки. На валидации подбираем отимальные параметры модели, а на тесте с уже подобранными параметрами сравним алгоритмы случайный лес и градиентный бустинг.

Эксперимент №1. Random Forest

Количество деревьев

В данном эксперименте были изучены зависимости RMSE и времени обучения от количества деревьев в случайном лесе. В данном алгоритме каждая отдельная модель, то есть дерево, получает свой predict и в итоговом ансамбле происходит усреденением по полученным предсказаниям. Глубина каждого дерева не ограничена, размерность пространства признаков - треть от исходного.

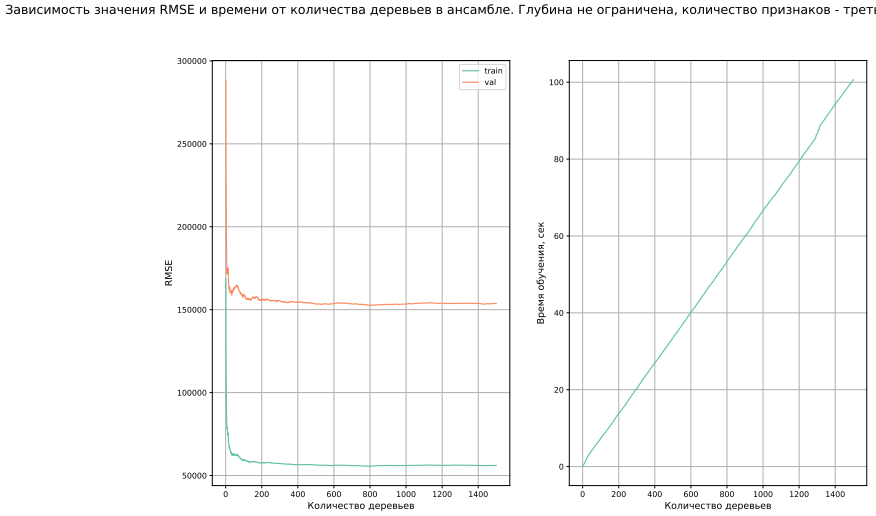

Рис. 1: Зависимость RMSE и времени обучения алгоритма от количества деревьев в случайном лесе

По графику (рис. 1) можно заметить, что поведение RMSE на трейне и валидации схоже: при маленьком числе деревьев (до 200) она с ростом числа деревьев быстро уменьшается, а далее на обучающей выборке выходит на плато, на валидации же на плато RMSE выходит только к 800 деревьям. Далее в качестве значения параметра **n_estimators** берем 812.

С возрастанием количества деревьев время обучения модели растет линейно, это логично, так как внутри алгоритма цикл по количеству деревьев в лесе, на каждой итерации происходят одинаковые действия.

Размерность пространства признаков для одного дерева

Далее выберем оптимальное значение размерности пространства признаков для одного дерева.

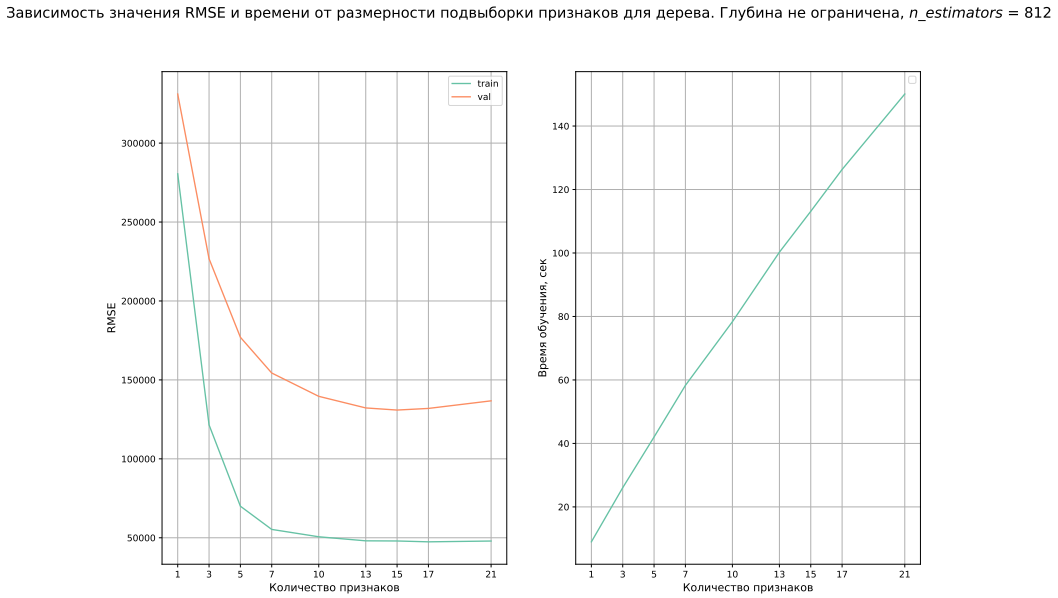

Рис. 2: Зависимость RMSE и времени обучения алгоритма от размерности пространства признаков для одного дерева в случайном лесе

Рассмотрим подробнее (рис. 2). Видим, что на обучении ошибка сильно уменьшается с увеличением числа признаков, а далее выходит на плато, начиная с момента, когда признаков становится чуть более половины. На тесте ошибка уменьшается медленнее, после преодоления отметки в 15 признаков, она начинает увеличиваться.

Время обучения тоже линейно.

Максимальная глубина дерева

Следующим шагом предлагается определить оптимальное значение максимальной глубины дерева.

По графику (рис. 3) видим, что опять же на обучении ошибка быстро уменьшается с увеличением глубины деревьев, наименьшее значение достигается при деревьях неограниченной глубины. На тесте ошибка уменьшается не так быстро, но ситуация аналогичная. Поэтому для итогового сравнения берем алгоритм без ограничения деревьев по глубине(**max_depth**=None).

Логично, что чем больше глубина деревьев в лесе, тем дольше времени потребуется на обучением, однако, начиная с глубины деревьев, равной 20, время увеличивается не так быстро, выходя практически на плато.

Рис. 3: Зависимость RMSE и времени обучения алгоритма от максимальной глубины дерева в случайном лесе

Эксперимент №2. Gradient Boosting

Количество деревьев и темп обучения

В градиентном бустинге каждый следующий базовый алгоритм(в нашем случае дерево) учится на ошибках предыдущего, тем самым можно быстрее и качественнее предсказать значение таргета, чем в случайном лесе. Темп обучения, обозначаемый как **learning_rate**, показывает, насколько быстро модель обучается. Каждое добавленное дерево изменяет общую модель. Величина модификации контролируется темпом обучения. То есть, чем ниже темп, тем медленнее обучается модель. Преимущество более низкой скорости обучения состоит в том, что модель становится более надежной и эффективной. Обычно модели, которые обучаются медленно, работают лучше. Однако медленное обучение обходится дорого. Обучение модели занимает очень много времени, что приводит нас к другому важному гиперпараметру. **n_estimators** - количество деревьев, используемых в ансамбле. Если темп обучения маленький, то нам нужно больше деревьев для обучения, чтобы уменьшить его время. Но следует помнить, что использование слишком большого количества деревьев создает высокий риск переобучения.

Исходя из вышеуказанного, будем подбирать количество деревьев и темп обучения одновременно. Здесь (рис. 4) RMSE указано для валидационной выборки. Рассмотрим график (рис. 4). Действительно, видим, что при наименьшем темпе обучения нам нужно больше деревьев, чтобы достичь минимума ошибки, так как кривая зависимости RMSE от количества деревьев убывает не так быстро, как при большем темпе обучения. Однако при больших значениях **learning_rate**, например 0.3 и 0.5, видим, что график выходит на плато очень быстро при маленьком числе деревьев, однако значение ошибки больше, наша модель не успевает хорошо обучиться. А если взять темп обучения очень большим - 1, то с увеличением числа деревьем ошибка возрастает при числе деревьев до 200, а далее выходит на плато, но с очень большим значением, что нам тоже не подходит.

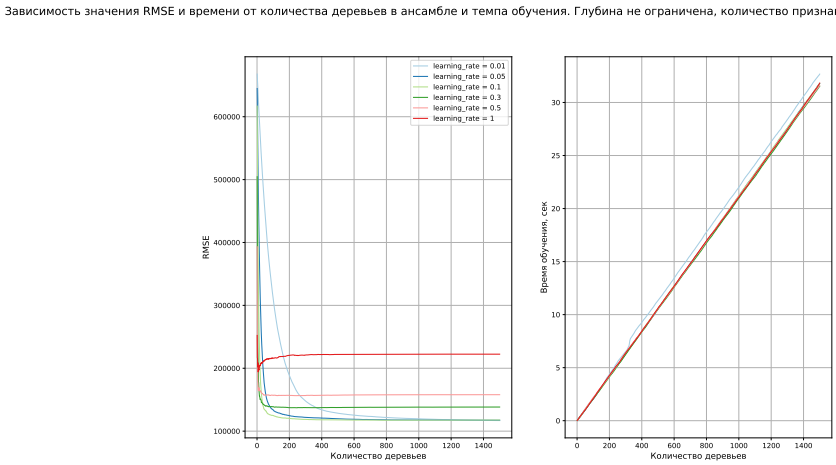

Рис. 4: Зависимость RMSE и времени обучения алгоритма от количества деревьев и темпа обучения в градиентном бустинге

Время обучения растет линейно. По графику видно, что оптимальным темпом обучения является значение 0.1, при таком темпе алгоритм достигает минимума RMSE при наименьшем числе деревьев - 400. Поэтому далее будем использовать **learning_rate**=0.1, **n_estimators**=400.

Размерность пространства признаков для одного дерева

Рассмотрим зависимость RMSE от размерности пространства признаков для одного дерева. Опять же построим графики.

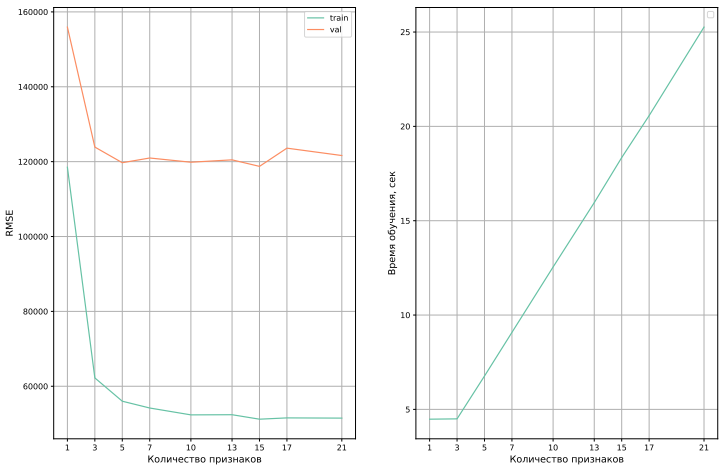

Рис. 5: Зависимость RMSE и времени обучения алгоритма от размерности пространства признаков для одного дерева в градиентном бустинге

Заметим, что график (рис. 5) очень отличается от аналогичного для случайного леса, на валидации теперь значения "скачут"и нет сначала монотонного убывания, а далее рост. Видим, что при 15 признаках значение ошибки на валидации наименьшее.

График времени растет линейно.

Максимальная глубина дерева

Рассмотрим подробнее график зависимости RMSE от максимальной глубины дерева в ансамбле.

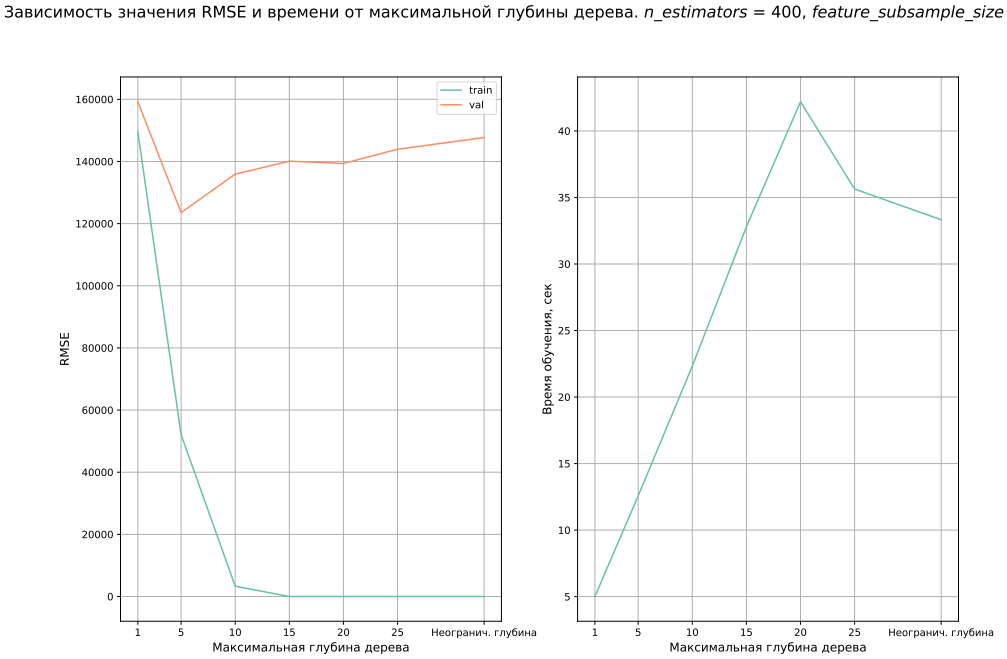

Рис. 6: Зависимость RMSE и времени обучения алгоритма от максимальной глубины для одного дерева в градиентном бустинге

По графику (рис. 6) видим, что алгоритм градиентный бустинг быстро переобучается и опять же в отличии от случайного леса нет монотонного уменьшения ошибки, при максимальной глубине деревьев, большей 5 RMSE на валидации возрастает.Оптимальное значение достигается при **max_depth**=5.

Время же сначала линейно растет, а далее, начиная с 20 признаков, убывает.

Итоговое сравнение алгоритмов

Теперь на тестовой выборке давайте проведем итоговое сравнение качества и скорости алгоритмов случайного леса и градиентного бустинга над деревьями. С помощью валидационной выборки мы подобрали следующие параметры:
Для случайного леса:
n_estimators=812,
feature_subsample_size=15,
max_depth=None.
Для градиентного бустинга:
n_estimators=400,
feature_subsample_size=15,
max_depth=5,
learning_rate=0.1

В таблице (табл. 1) представлены результаты работы алгоритмов. Видим, что при использовании алгоритма градиентный бустинг ошибка получилась меньше и время работы меньше в несколько раз.

Алгоритм	RMSE на тесте	Время обучения, сек
Случайный лес	145063.0965	109.14
Градиентный бустинг	125832.295	16.9

Таблица 1: Время и точность работы случайного леса и градиентного бустинга над деревьями

Вывод

В данном задании были построены графики зависимостей RMSE и времени обучения алгоритмов (случайный лес и градиентный бустинг над деревьями) от различных параметров, по полученным графикам эти параметры подбирались оптимальным образом. Далее было произведено сравнение двух моделей, можно сделать вывод, что на предоставленных данных лучше использовать градиентный бустинг над деревьями, а не случайный лес, так как он лучше предсказывает целевое значение переменной за меньшее время.