Композиции алгоритмов для решения задачи регрессии. Отчет.

Пацация Александр, 317 группа

19 декабря 2021 г.

1 Введение

В данном отчете рассматривается поведение собственной реализации алгоритмов случайного леса (RF – RandomForest) и градиентного бустинга (GB – GradientBoosting), основанных на классе **DecisionTreeRegressor** из библиотеки **scikit-learn**. Для анализа используется датасет с данными о продаже недвижимости **House Sales in King's County, USA** с 2014 по 2015 год. Рассматривается задача регрессии, целевой переменной является цена проданной недвижимости в долларах. используется метрика качества **RMSE**:

$$\sqrt{\frac{1}{l}\sum_{i=1}^{l}(y_i - a(x_i))},$$

где y_i – значение целевой переменной на объекте x_i, $a(x_i)$ – предсказание алгоритма на объекте x_i, l – размер обучающей выборки.

RMSE является одной из самых часто используемых метрик качества в задачах регрессии. Данная метрика сильно штрафует за большие отклонения от целевого значения, поэтому имеет свойство переобучаться на выбросах. Однако в рассматриваемой задаче она очень хорошо подходит, так как вероятность ошибки в ней достаточно мала (цены продаж хорошо задокументированы, поэтому источником ошибки может быть только человеческий фактор).

Обучающая выборка состоит из 70% объектов датасета, тестовая из 30%

1.1 Предварительная обработка данных

Первоначальной задачей анализа данных является первичная обработка данных. С датасете присутствует столбец **id**, который сразу же можно удалить, так как он не связан с таргетом и может только ухудшить качество модели, которая будет искать ложные зависимости, основываясь на этом этом признаке. Теперь можно определить что означают остальные признаки:

1. date – дата продажи дома.
2. price – цена продажи дома, целевая переменная.
3. bedrooms – количество спальных комнат.
4. bathrooms – количество ванных комнат. значение 0.5 означает, что ванной комнате нет душа.
5. sqft_living – количество квадратных футов внутренней части дома.
6. sqft_lot – количество квадратным футов земельного участка.
7. floors – количество этажей
8. waterfront – признак, равный 1, если апартаменты выходят на набережную, 0 – иначе.
9. view – субъективная оценка общего внешнего вида здания от 0 до 4.
10. condition – субъективная оценка общего состояния дома от 1 до 5,
11. grade – оценка качества постройки и дизайна от 1 до 13.
12. sqft_above – количество квадратных футов над уровнем земли (выше первого этажа).
13. sqft_basement – количество квадратных футов под уровнем земли (подвал).
14. yr_built – год постройки дома.

15. yr_renovated – год реновации дома. 0, если реновация не проводилась.
16. zipcode – почтовый индекс, в рамках которого находится дом.
17. lat – широта, на которой находится дом.
18. long – долгота, на которой находится дом.
19. sqft_living15 – суммарное количество квадратных метров внутренней части домов 15 ближайших соседних домов.
20. sqft_lot15 – суммарное количество квадратных метров земельного участка 15 ближайших соседних домов.

В выборке присутствует несколько категориальных признаков, однако только признак **date** имеет тип **object**. Его стоит перевести в какой либо численный формат, чтобы алгоритм был способен его обработать (DecisionTreeRegressor не умеет работать с необработанными категориальными признаками). Предварительно переведем его в целочисленный формат, нумеруя значения в порядке их поступления. Теперь все признаки имеют целочисленный формат и могу быть обработаны нашими алгоритмами. Стоит посмотреть на корреляцию признаков между собой, чтобы извлечь какую-то полезную информацию. Для этого можно построить корреляционную матрицу (рис. 1).

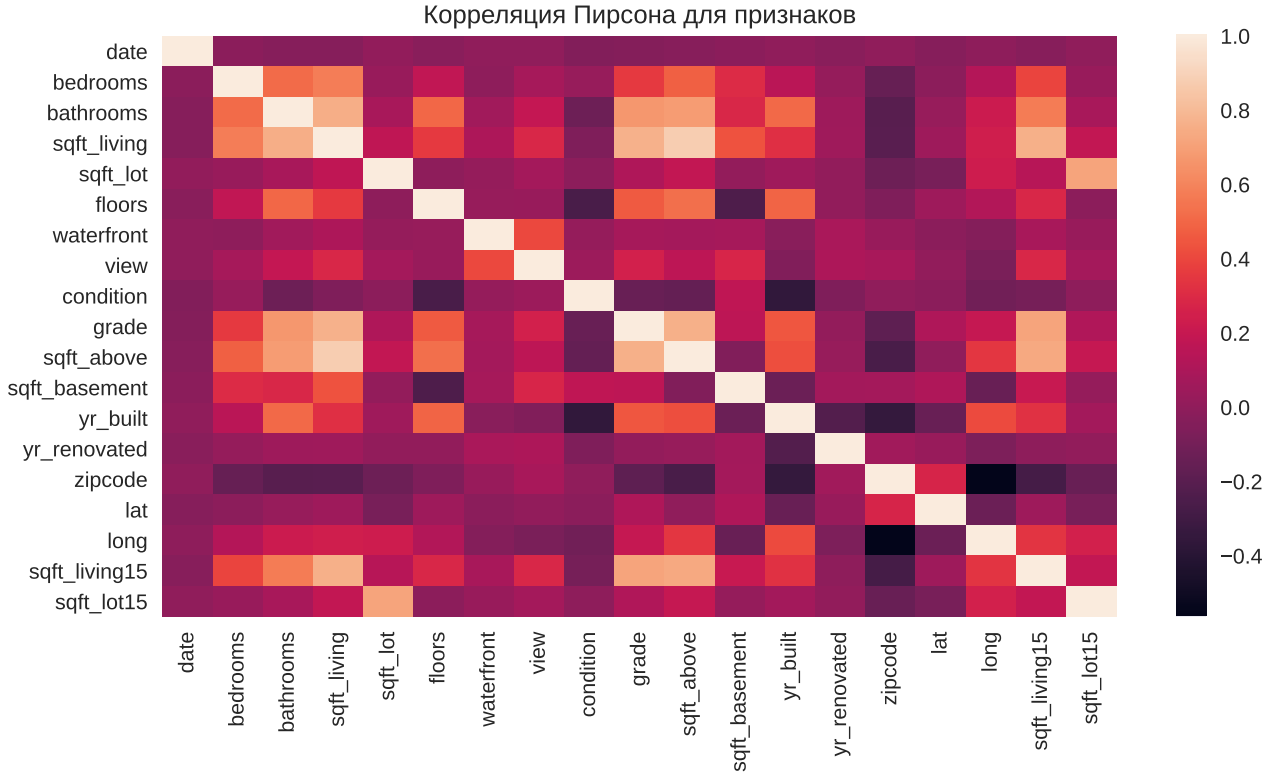

Рис. 1: Корреляционная матрица признаков

Как можно заметить, есть несколько признаков хорошо коррелирующих между собой. Хорошо коррелируют признаки **sqft_living, sqft_living15, sqft_above, grade, bathrooms**, что, в целом, не является удивительным. Связь **sqft_living** и **sqft_living15**, означает, что все дома находятся в районах соответствующего достатка, связь **sqft_living** и grade говорит о том, что качество постройки крупных зданий лучше, остальные зависимости достаточно очевидны.

Стоит отметить отрицательную зависимость некоторых признаков. **yr_built** и **condition** имеют большую отрицательную корреляцию, что говорит о том, что чем старше здание, тем хуже его состояние. Это происходит из-за того, что лишь небольшая часть домов была отреставрирована (– доля ненулевых значений признака **yr_renovated**).

Признак date имеет слабую корреляцию со всеми другими признаками, что может сигнализировать о его бесполезности.

В задаче присутствуют следующие категориальные признаки: **date, bedrooms, bathrooms, floors, waterfront, view, condition, grade, yr_built, yr_renovated, zipcode**. Признаки **view, condition, grade** являются порядковыми по определению, признаки **bedrooms, bathrooms, floors** также можно рассматривать как порядковые ввиду их неплохой коррелированности с численными признаками и интуитивных соображений (чем больше, тем лучше).

Встает вопрос о том, какой способ кодирования категориальных признаков использовать. **One Hot Encoding** не очень хорошо работает с решающими деревьями и с алгоритмами RF и GB в частности. Происходит сильное расширение признакового пространства, причем много признаков будут иметь разреженный вид. Учитывая то, что каждое базовое дерево в RF и GB обучается на подвыборке признаков и эта подвыбрка может состоять в основном из разреженных признаков, можно часто наблюдать ситуацию когда в вершинах происходит разбиение почти не увеличивающее информативность. То есть лимит глубины будет будет тратится на разбиения, не улучшающие модель.

Альтернативой **One Hot encoding'a** является обыкновенный **Ordinal Encoding**, ставящий в соответствие каждому категориальному значению целое неотрицательное число. Данный способ плох тем, что он может задать порядок там, где он не имеет смысла, из-за этого он редко используется. Однако в нашей задачи много порядковых признаков, поэтому данный способ является неплохим вариантом. Проблема возникает только с признаками **zipcode, date** (в данной задаче нет оснований считать его порядковым признаком)**, yr_built, yr_renovated**. разбиение выборки по этим признакам в вершине дерева, строго говоря, не имеет смысла (в плане восприятия). Можно сказать, что в таком случае решающее дерево теряет свою интерпретируемость. Но это никак не должно сказаться на качестве модели, поэтому используется именно этот метод.

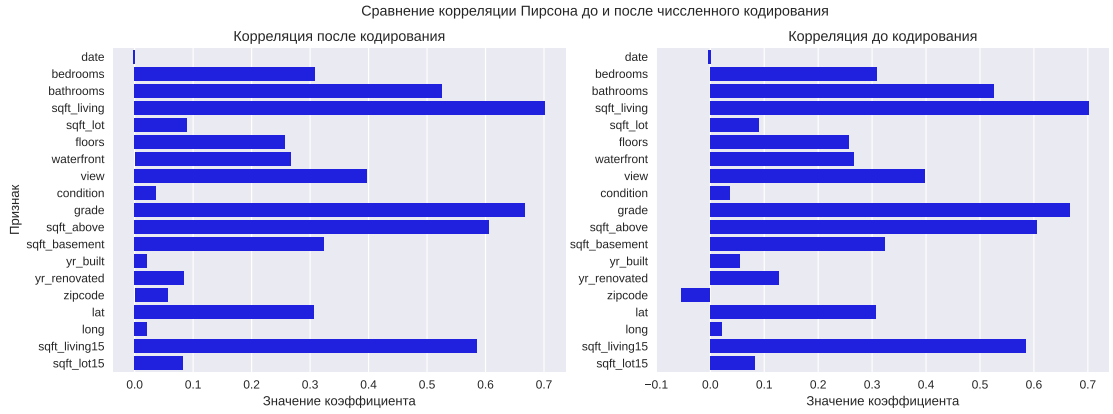

Рис. 2: Корреляция признаков с таргетом

На рис. 2 представлена корреляция признаков с таргетом до и после кодирования категориальных признаков. Видно, что большинство признаков отлично коррелируют с целевой переменной. Также, после кодирования корреляция признака **zipcode** стала положительной. Стоит заметить, что корреляция признака **date** очень маленькая. Учитывая все сказанное выше про данный признак, можно сделать вывод, что в период с 2014 по 2015 год не происходило никаких событий способных повлиять на цену недвижимости, поэтому его можно убрать из выборки. Принимая во внимание специфику задачи, не стоит ожидать какую-либо немонотонную зависимость в прогнозировании цены недвижимости, поэтому отбор корреляцией дает достаточно точную информацию о влиянии признаков на таргет. Стоит также учесть, что признак **date** можно было и не удалять, так как деревья решений имеют свойство самостоятельного отбора признаков. Но, если бы мы использовали One Hot Encoding, качество алгоритма могло бы сильно ухудшится, так как семплировние признаков происходит один раз для каждого базового алгоритма. Из-за этого, при высокой доле разреженных признаков дерево дает плохой результат.

1.2 Случайный лес

Предсказания случайного леса представляют из себя среднее арифметическое предсказаний всех базовых алгоритмов обученных на выборках, полученных бутстрапом (выборка с возвращением) из оригинального датасета (бэггинг). Первым делом, стоит отметить, что рассматривается не классическая реализация алгоритма RF, в которой пространство признаков семплируется из исходного в каждой вершине базового дерева. В рассматриваемом алгоритме признаки семплируются один раз, перед обучением дерева.

Рассмотрим поведение алгоритма случайного леса в зависимости от числа деревьев и глубины (рис. 3) при размерности признакового подпространства по умолчанию ($frac13$ от всего пространства). Сразу стоит отметить заметный разброс результатов работы алгоритма с одинаковыми параметрами (для каждого набора параметров алгоритм запускался 20 раз). На участке 100-125 базовых алгоритмов RF выходит на плато и прирост точности становится незначительным. Результат работы алгоритма с глубиной 15 и 20 почти не отличаются, то есть делать максимальную глубину больше 20 уже имеет мало смысла.

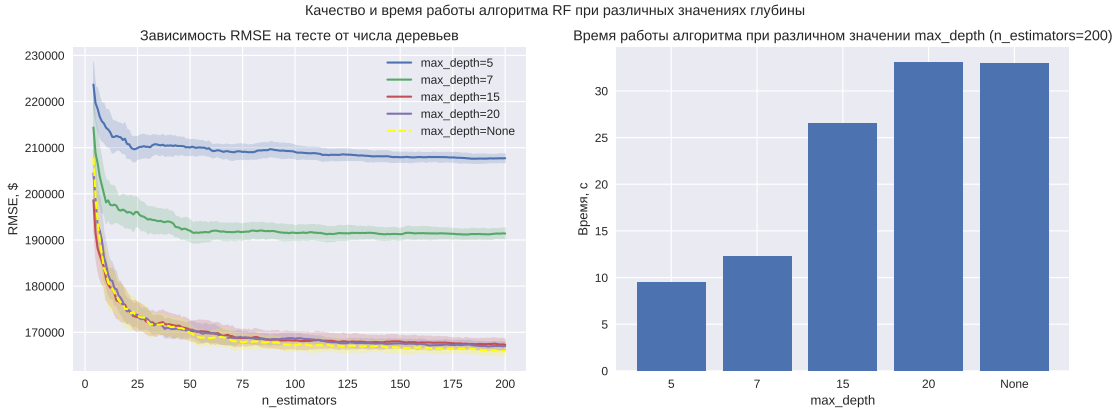

Рис. 3: Зависимость RMSE и времени работы RF при различных параметрах.

Также стоит отметить, что алгоритм без ограничения глубины показывает очень схожий результат с алгоритмом при max_depth=20 как по качеству, так и по времени работы. Можно предположить, что итоговые деревья получающиеся при max_depth=None в данной задаче имеют глубину около 20. Стоит также пояснить почему увеличение глубины дерева приводит к улучшению качества алгоритма. Это следует из разложения ошибки беггинга на смещение и разброс (Bias variance decomposotion). Дисперсия алгоритма зависит от числа базовых деревьев и их коррелированности. Если базовые алгоритмы не коррелированны, то дисперсия RF в N раз меньше их дисперсии, где N – число базовых деревьев. Слабую коррелированность обеспечивает рандомизация по признакам и бутстрап. Таким образом, при построении переобученных базовых алгоритмов, их композиция может давать хорошую точность. Однако, как видно из рис. 3 время работы алгоритма заметно увеличивается.

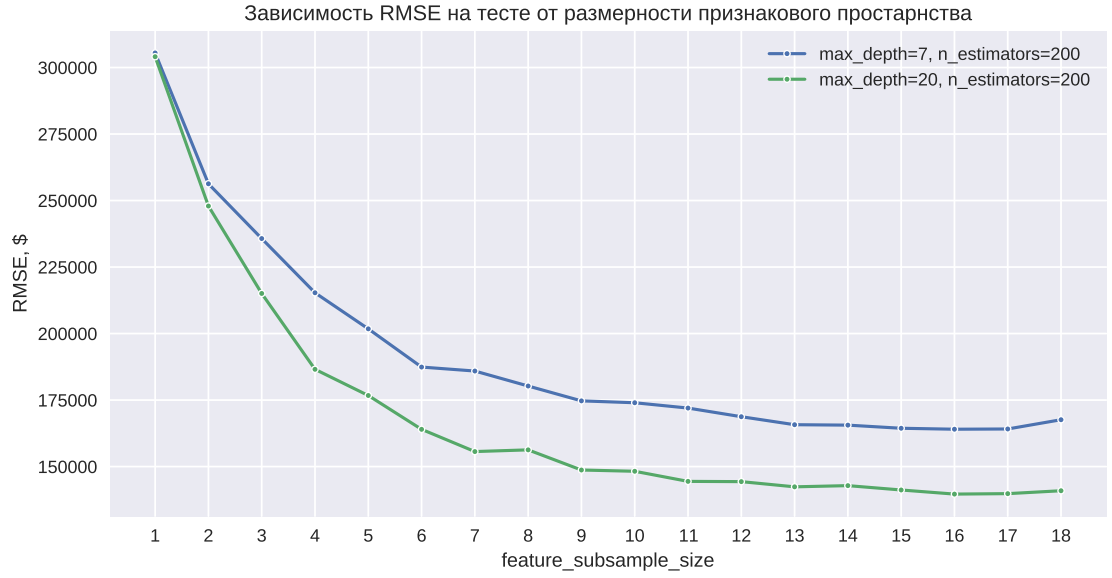

Рис. 4: Зависимость RMSE алгоритма RF при различных размерностях признакового подпространства.

Из рис. 4 видно, что при увеличении размерности признакового подпространства растет качество моде-

ли (feature_subsample_size). Эвристическое значение параметра $\lfloor d/3 \rfloor$, предложенная авторами алгоритма, в данной задаче не является оптимальным. График выходит на плато при feature_subsample_size=13.

Таким образом, можно сделать вывод, что в рассматриваемой задаче, алгоритм RF достигает оптимального качества при n_estimators ≥ 100, max_depth ≥ 15 и feature_subsample_size ≥ 13.

1.3 Градиентный бустинг

Рассматривается классическая реализация градиентного бустинга по метрике MSE. Очередной базовый алгоритм обучается на антиградиент функции потерь (в нашем случае MSE) и добавляется в композицию с наилучшим коэффициентом.

Сначала фиксируется количество деревьев и подбираются оптимальная размерность признакового пространства и глубина.

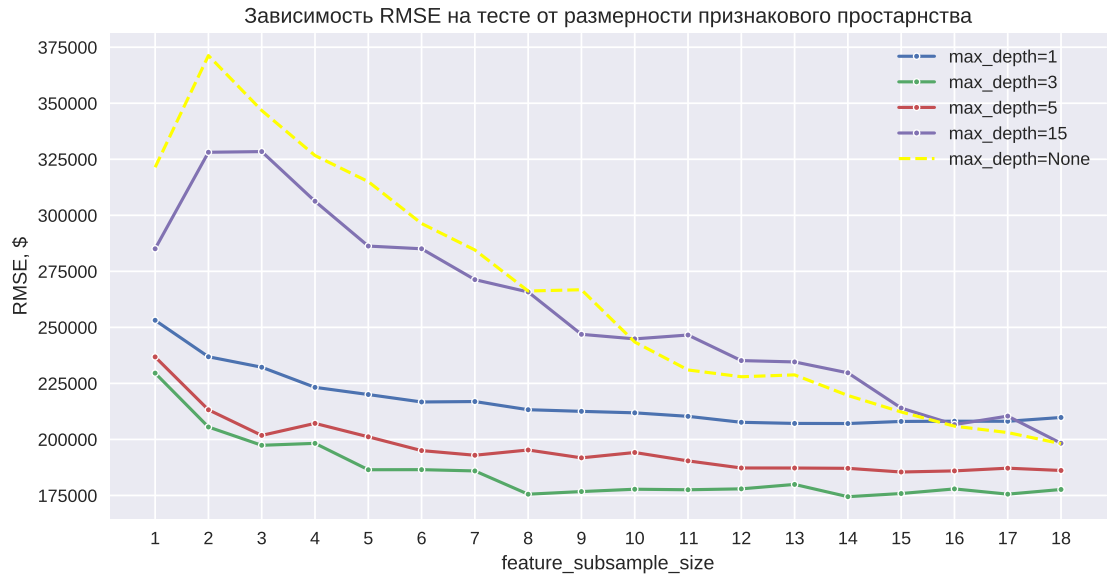

Рис. 5: Зависимость RMSE алгоритма GB при различных размерностях признакового подпространства и значений глубины.

На рис. 5 представлен усредненный результат под 20 прогонам алгоритма для каждого набора исследуемых параметров (глубина, размерность подпространства). Как можно заметить, лучший результат показывает алгоритмов с max_depth=3. При дальнейшем увеличении максимальной глубины качество алгоритма падает. В отличие от RF, который работает тем лучше, чем больше глубина базовых деревьев, алгоритм GB показывает лучшие результаты на деревьях относительно малой глубины. Очередное дерево сдвигает композицию на антиградиент функции потерь, поэтому деревья малой глубины все равно будут заметно улучшать модель. Эмпирические данные также говорят о том, что GB не очень хорошо уменьшает разброс (variance) поэтому увеличение глубины деревьев не улучшает качество.

На рис. 6 изображен усредненный по 20 прогонам результат работы GB при различных значениях темпа обучения и количества базовых алгоритмов. Как можно видеть, время работы GB увеличивается в уменьшением темпа. Лучшее значение **RMSE** достигается при learning_rate=0.1. При большом количестве базовых деревьев алгоритмы с меньшим темпом обучения обгоняют по точности модели с более высоким темпом. Также, можно заметить, что при больших коэффициентах алгоритм теряет в точности при большом количестве деревьев, что говорит о том, что алгоритм GB способен переобучаться, в отличие от случайного леса, качество которого с добавлением деревьев может только расти.

Рис. 6: Зависимость RMSE и времени алгоритма GB при различных значениях темпа обучения и количества базовых деревьев.

1.4 Вывод

По результатам данного отчета можно сделать следующие выводы относительно алгоритмов RF и GB:

1. В данной задаче качество алгоритма RF выходит на плато, начиная, примерно, с 100 базовых моделей. Дальнейший прирост качества незначителен.

2. Для разных значений темпа обучения GB имеет различные значения оптимального количества базовых деревьев. Малый темп обучения позволяет достичь лучшей точности ценой увеличения числа базовых алгоритмов. Оптимальное значение признакового подпространства составляет примерно 8 признаков.

3. RF имеет высокое значение оптимальной глубины деревьев, в отличие от GB.

4. При больших значениях темпа обучения GB переобучается, в то время как RF нет.

5. Увеличение глубины значительно сказывается на времени обучения.