«Московский государственный университет имени М. В. Ломоносова»
Факультет вычислительной математики и кибернетики
Кафедра методов математического прогнозирования

Отчет по практическому заданию
Задание 3. Ансамбли алгоритмов. Веб-сервер. Композиции алгоритмов для решения задачи регрессии.

Дмитриева Татьяна, 3 курс

Декабрь 2021

Содержание

1 Введение

В данном задании предлагается написать собственную реализацию методов градиентного бустинга и случайного леса на языке Python и провести серию экспериментов с их использованием. Целью данных экспериментов является исследование поведения данных методов в зависимости от разных параметров. Эксперименты этого задания проводятся на датасете данных о продажах недвижимости House Sales in King County, USA. Также требуется реализовать дополнительный функционал - веб-сервер, обернутый в docker контейнер.

1.1 1 эксперимент

1.1.1 Описание эксперимента

В данном эксперименте требовалось произвести предварительную обработку данных, а также разделить данные на тестовую и обучающую выборку. Для того чтобы исследуемые нами методы, правильно работали, нам следует привести все признаки к вещественным числам, так как у нас задача регрессии. В ходе изучения имеющихся у нас признаков оказалось, что единственным не вещественным признаком был столбец с названием "date"в строковом формате. Было решено оставить в строке только цифры и затем преобразовать ее в целое число для каждого объекта. Для этого были использованы следующие функции языка Python: **str.lower** для приведения к нижнему регистру и **re.sub**. Затем наши данные были разделены в соотношении 4 к 1 на обучающую и тестовую выборку, а также отделена целевая переменная - "price".

1.1.2 Вывод

Таким образом, в этом эксперименте мы преобразовали наши данные для корректной работы исследуемых методов и разделили данные для дальнейшего исследования. Можно было бы также преобразовать найденный нами не численный признак с помощью быстрого кодирования(One-Hot Encoding), однако это используется для категориальных признаков. А для даты, заданной с использованием года, месяца и числа, более разумно использовать преобразование в целое число, если уникальных значений для этого признака достаточно много.

1.2 2 эксперимент

1.2.1 Описание эксперимента

Во втором эксперименте требовалось изучить поведение алгоритма случайный лес при решении задачи регрессии в зависимости от используемых для этого параметров:

- количество деревьев в ансамбле
- размерность подвыборки признаков для одного дерева
- максимальная глубина дерева (дополнительно разберите случай, когда глубина неограниченна)

Для изучения поведения случайного леса мы построили зависимость значения среднеквадратичной ошибки(здесь и далее **RMSE**) и времени от количества деревьев на отложенной выборке, использовав для его конструирования следующие значения параметров **max_depth** = **None**, **feature_subsample_size** = **None**. Для решающих деревьев были взяты параметры по умолчанию.

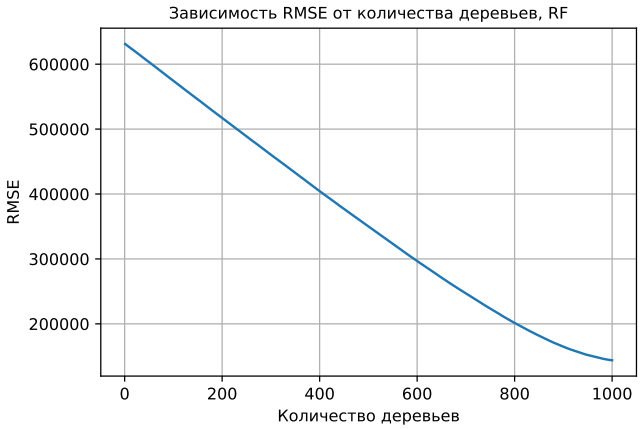

Рис. 1: RMSE

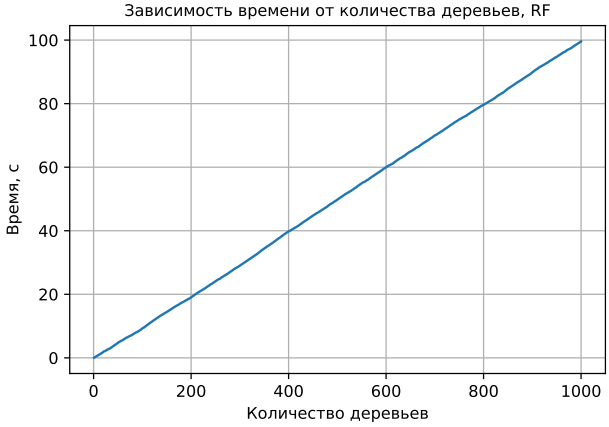

Рис. 2: Время

Изучая график зависимости времени от количества построенных деревьев(рис.2), мы можем сказать, что при возрастании их количества время увеличивается. Это довольно ожидаемый результат, ведь каждое из деревьев усложняет наш алгоритм и считается вслед за предыдущим. Что касается зависимости **RMSE** от количества деревьев(рис.1), то тут мы получили хорошо интерпретируемую зависимость: при увеличении количества деревьев наша мера эффективности работы алгоритма возрастает. Это, разумеется, связано с тем, что каждое последующее дерево уточняет и дополняет предыдущие. Каждый базовый алгоритм в ансамбле по отдельности не смог бы показать такого результата, однако вместе в среднем они довольно хорошо приближают наши предсказания к правильным ответам.

Далее построим зависимость **RMSE** и времени от размерности признакового подпространства, которое мы будем использовать для построения наших базовых алгоритмов. Здесь были использованы следующие значения параметров $\mathbf{n\_estimators} = \mathbf{1000}$, $\mathbf{max\_depth} = \mathbf{None}$. Для решающих деревьев были взяты параметры по умолчанию.

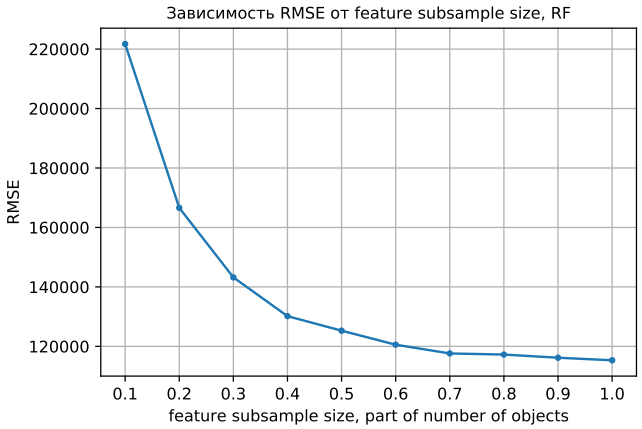

Рис. 3: RMSE

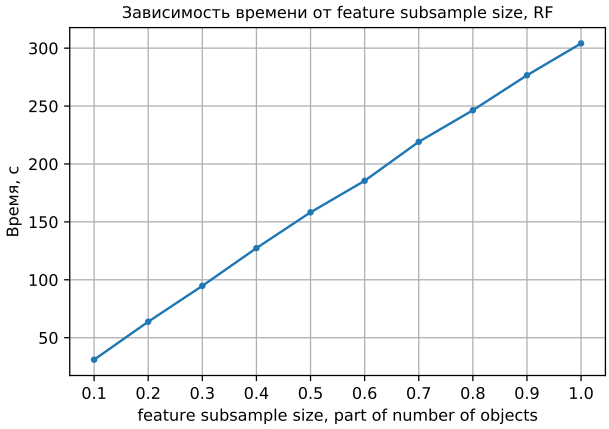

Рис. 4: Время

Как мы видим, графики зависимости **RMSE** и времени от размерности взятого при построении решающих деревьев признакового пространства показывают, что при увеличении их количества **RMSE** понижается, а время растет. Этот вывод мы могли бы сделать интуитивно, предполагая, что при увеличении количества признаков, построение одного базового алгоритма в ансамбле становится более долгим, ведь для этого используются вся имеющаяся информация. **RMSE** же понижается, потому что при увеличении количества признаков деревья становятся более разнообразными. Однако здесь следует отметить, что после достижения отношения размерности подпространства к общему размеру в значении 6 к 10 ощутимого прироста в качестве уже не наблюдается, поэтому для оптимальной работы алгоритма лучше брать не все признаки, чтобы время обучения алгоритма стало значительно меньше.

Теперь мы рассматриваем зависимость времени и **RMSE** от глубины используемых деревьев. Для конструирования алгоритма будем использовать следующие значения параметров $\mathbf{n\_estimators} = \mathbf{1000}$, **feature_sub-**

sample_size - размерность всего признакового пространства данных. Для решающих деревьев были взяты параметры по умолчанию.

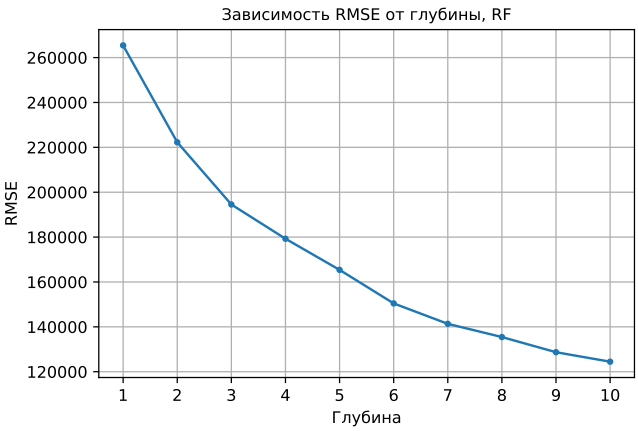

Рис. 5: RMSE

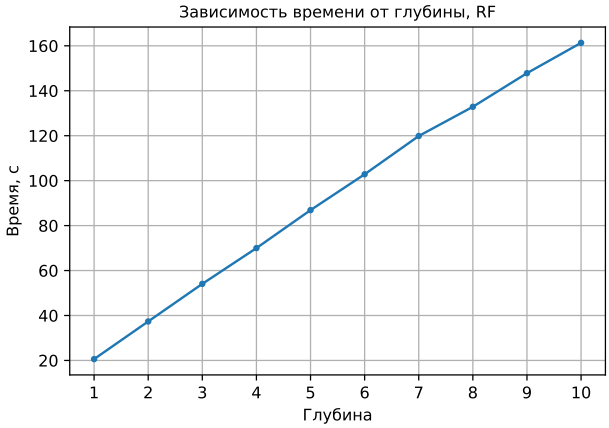

Рис. 6: Время

На представленных выше графиках(рис.3, 4) мы видим, что при увеличении глубины деревьев возрастает как время, так и эффективность нашего алгоритма, так как **RMSE** уменьшается. Это можно объяснить тем, что алгоритмы становятся сложнее. Отдельно отметим, что при выборе неограниченной глубины RMSE оказывается очень высок. Алгоритм в этом случае, вероятно, переобучается, поэтому очень важно указывать глубину. На графике решено было не указывать эту точку, так значение в ней слишком велико, поэтому она будет указана в тексте, **RMSE = 344662**.

1.2.2 Вывод

Изучая все построенные зависимости, мы можем сказать, что алгоритм случайный лес работает тем лучше, чем больше деревьев мы используем для его построения. Кроме того, оптимальным решением оказывается брать не всю подвыборку, а лишь ее часть - это ускоряет алгоритм, почти не ухудшая качество работы.

1.3 3 эксперимент

1.3.1 Описание эксперимента

В третьем эксперименте требовалось изучить поведение алгоритма градиентный бустинг при решении задачи регрессии в зависимости от используемых для этого параметров:

- количество деревьев в ансамбле
- размерность подвыборки признаков для одного дерева
- максимальная глубина дерева (дополнительно разберите случай, когда глубина неограниченна)
- выбранный learning_rate

При исследовании поведения градиентного бустинга мы посчитали зависимость **RMSE** и времени от количества деревьев на отложенной выборке, использовав для конструирования алгоритма следующие значения параметров **max_depth = None**, **feature_subsample_size = None**, **learning_rate = 0.1**. Для решающих деревьев были взяты параметры по умолчанию.

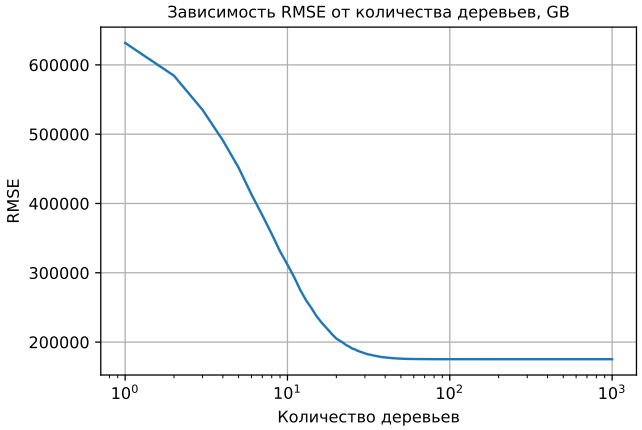

Рис. 7: RMSE

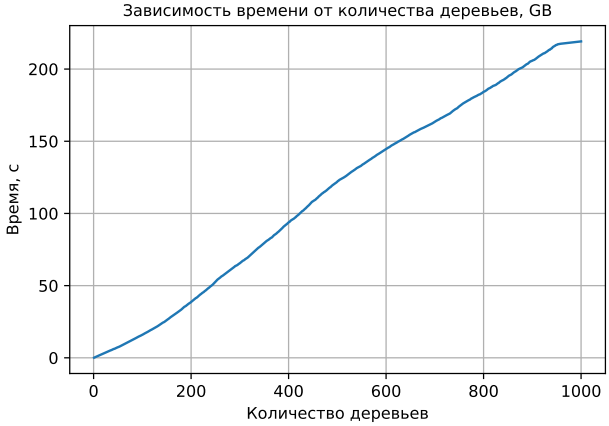

Рис. 8: Время

Как мы видим на графиках(рис.7, 8), наш алгоритм довольно хорошо справляется с задачей уже при количестве деревьев равным 100, а затем **RMSE** перестает понижаться, а время продолжает возрастать. Из этого можно сделать вывод о том, что данный алгоритм не стоит использовать с большим количеством деревьев, так как это просто увеличит время его обучения, не давая прироста в качестве.

Следующим исследуемым параметром возьмем размерность признакового подпространства. Для этого исследования будем использовать параметры: **n_estimators = 300**, **max_depth = None**, **learning_rate = 0.1**. Для решающих деревьев были взяты параметры по умолчанию.

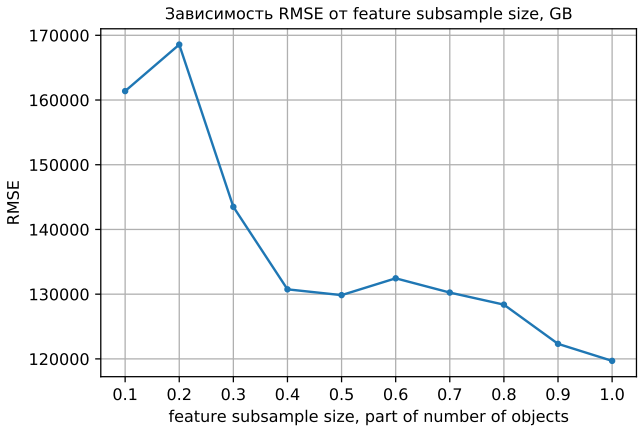

Рис. 9: RMSE

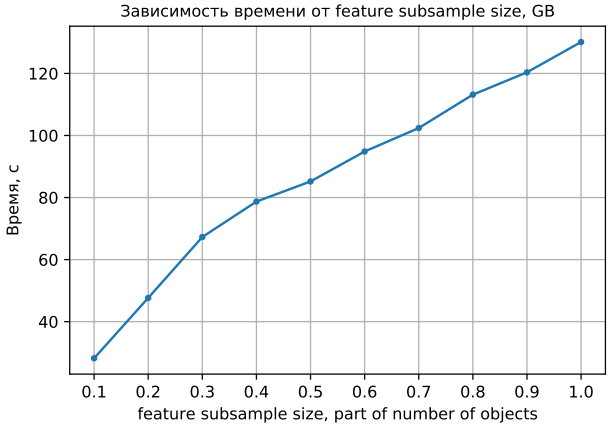

Рис. 10: Время

В графиках зависимости времени и **RMSE** от размерности признакового подпространства(рис.9, 10) в этот раз можно заметить довольно интересное поведение алгоритма. При достижении некоторого значения, понижающееся до этого момента значение **RMSE** начинает снова увеличиваться для нескольких значений размерности. Вероятно, это связано с тем, что наш алгоритм раньше строил различные деревья, но при использовании все увеличивающегося количества признаков, каждое из его деревьев становится в чем-то похожим на остальные, и поэтому идея с домножением каждого дерева на собственный вес перестает быть эффективной, ведь в среднем деревья оказываются похожи. Тем не менее, когда мы берем всю выборку, **RMSE** оказывается самым лучшим. Это, вероятно, связано с тем, что мы берем

Далее изучим поведение алгоритма в зависимости от глубины решающих деревьев. Для этого возьмем параметры: **n_estimators = 300**, **feature_subsample_size - все признаковое пространство объектов**, **learning_rate = 0.1**.

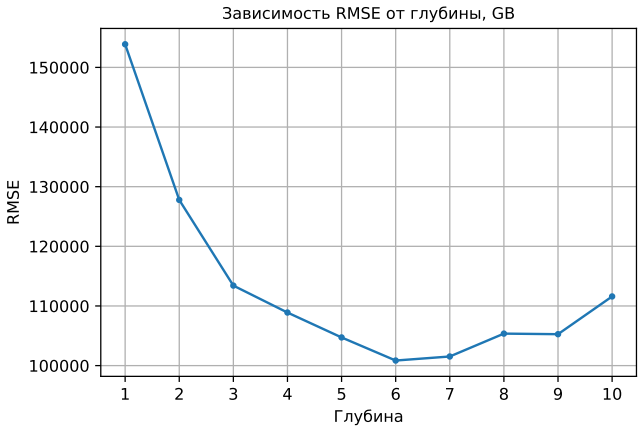

Рис. 11: RMSE

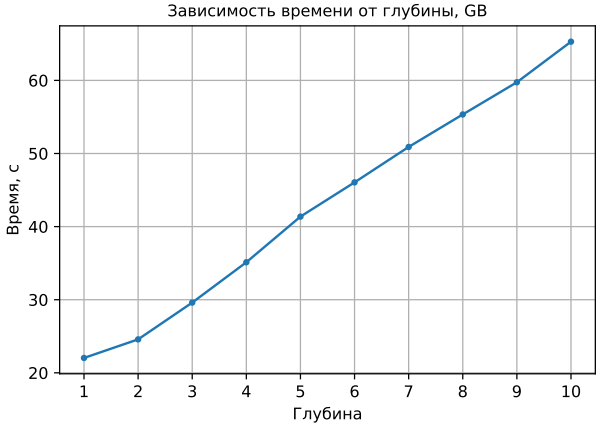

Рис. 12: Время

Как мы можем заметить, на этих графиках(рис.11, 12) наблюдается увеличение времени и понижение **RMSE** при увеличении глубины построенных деревьев. Однако после достижения оптимальной глубины качество нашего алгоритма снова начинает ухудшаться, достигая самого высокого значения **RMSE** при неограниченной глубине, на графике решено было не указывать эту точку, так значение в ней слишком велико, поэтому она будет указана в тексте, **RMSE = 1339060**. Это можно объяснить тем, что мы используем все признаки, поэтому деревья при более сильном углублении начинают походить друг на друга, переставая создавать уникальные признаки.

Наконец, приступим к исследованию поведения нашего алгоритма при изменении **learning_rate**. Для этого возьмем следующие параметры для нашего алгоритма: **n_estimators = 300**, **feature_subsample_size = None**, **max_depth = None**.

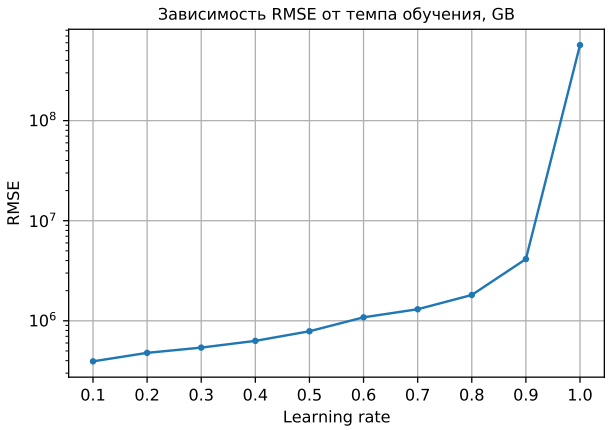

Рис. 13: RMSE

Рис. 14: Время

Как мы можем заметить, полученные графики(рис.13, 14) изображают довольно интересные зависимости. Самое низкое значение **RMSE** достигается при **learning_rate = 0.1**. При этом время довольно сильно меняется в зависимости от этого параметра, достигая наименьшего значения при наибольшем значении **learning_rate**.

1.3.2 Вывод

Таким образом, мы в ходе этого эксперимента выяснили, что для данного алгоритма отличным решением будет строить более сложные деревья при небольшом количестве, тем самым создавая качественные базовые алгоритмы - наши новые признаки и подбирая по ним наилучшие веса.

2 Вывод

Подводя итог, мы можем сказать, что построенные нами алгоритмы имеют свои недостатки и преимущества. Они похожи по идее построения, однако способ их работы отличает их друг от друга. Для случайного леса мы подбираем большее количество несложных базовых алгоритмов, в то время как для градиентного бустинга мы строим сложные деревья в небольшом количестве, создавая из них новые признаки для наших объектов. Кроме того, следует отметить, что при работе над данными экспериментами мы использовали систему контроля версий, что значительно облегчило проведение экспериментов.