НИУ ИТМО ФИТИП ПМИ

Методы оптимизации в машинном обучении

Отчет по лабораторной работе 3

Работу выполнили:

Цицин К.А., М3235

Балакин Д.А., М3235

Санкт-Петербург, 2023

Цель работы: изучение Метода стохастического градиентного спуска (SGD) и его модификаций.

Задачи:

Реализовать и исследовать на эффективность SGD для решения линейной регрессии:

1. с разным размером батча – от одного до размера полной коллекции (обычный GD);

2. с разной функцией изменения шага (learning rate scheduling).

3. scipy.optimize: SGD, и модификации SGD (Nesterov, Momentum, AdaGrad, RMSProp, Adam). Изучить параметры вызываемых библиотечных функций.

Реализовать и исследовать на эффективность SGD для полиномиальной регрессии с добавлением регуляризации в модель разных методов регуляризации (L1, L2, Elastic регуляризации).

Разобрать подробнее постановку задачи оптимизации в методе опорных векторов. Привести пример, иллюстрирующий задачу и её решение.

Ссылка на реализацию:

https://github.com/DmitryBalakin54/Metopts

Стохастический градиентный спуск (Stochastic Gradient Descent, SGD) — это метод оптимизации, широко используемый для обучения машинных моделей, особенно в задачах машинного обучения и нейронных сетей. Его основная идея заключается в использовании случайных подвыборок данных для обновления параметров модели, что позволяет ускорить процесс обучения и сделать его более эффективным в условиях больших данных.

Стохастичность: В отличие от стандартного градиентного спуска, который вычисляет градиенты, используя весь набор данных, стохастический градиентный спуск обновляет параметры модели на основе градиентов, вычисленных только на одной случайно выбранной обучающей выборке или на небольшом подмножестве данных (мини-пакете). Это делает процесс обучения значительно быстрее, особенно для больших наборов данных.

Шум и обобщение: Благодаря стохастичности, процесс обучения содержит элемент шума, который может помочь модели избежать локальных минимумов и улучшить обобщающие способности модели.

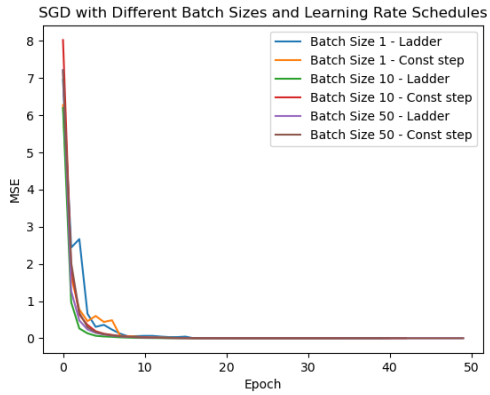
Преимущества SGD

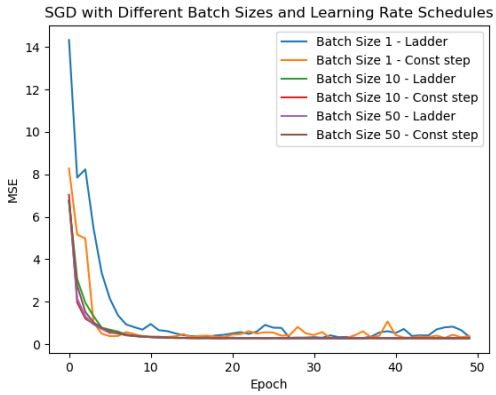
- **Скорость**: Быстрее в обновлении параметров для больших наборов данных.
- **Память**: Требует меньше памяти, так как обрабатывает небольшие части данных за раз.

Недостатки SGD

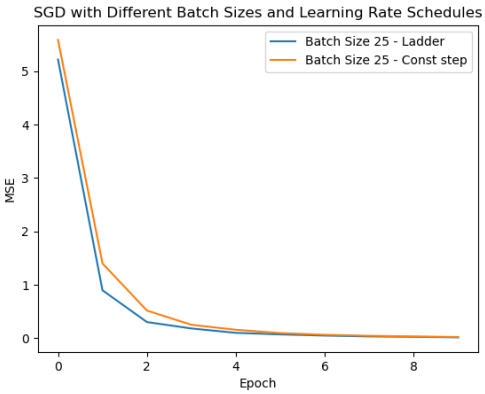
- **Шум**: Высокий уровень шума может затруднить достижение точной минимизации функции потерь.
- **Требует настройки гиперпараметров**: Необходимость выбора подходящего шага обучения и размера мини-пакетов.

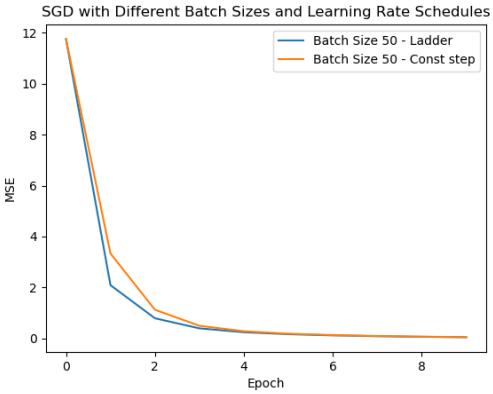
Анализ работы метода при различных размерах батча и функциях изменения шага





На картинках приведены графики сходимости соответственно обычных и зашумленных значений. Видно, что при увеличении размера батча сходимость стохастического градиентного спуска ускоряется, а различные неточности и отклонения уменьшаются. Ступенчатое уменьшение шага также заметно влияет на скорость сходимости.



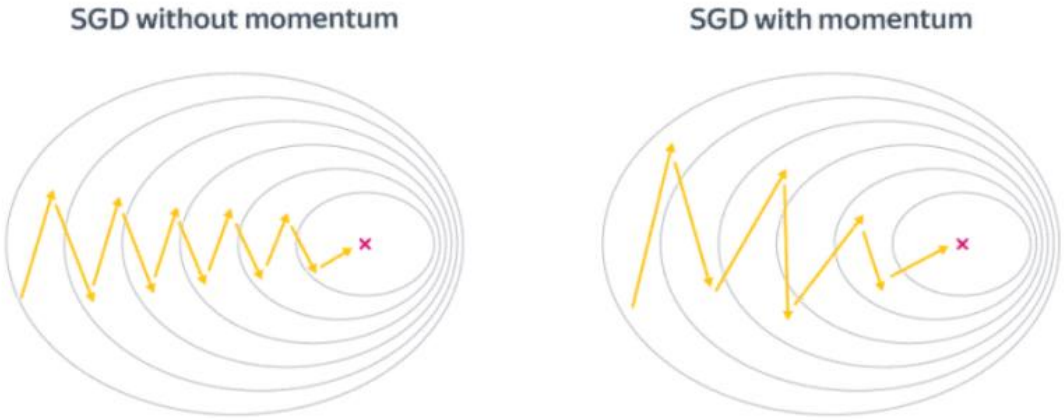


Также при ближайшем рассмотрении видно, что ступенчатый шаг показывает более быструю сходимость к оптимальному результату

Модификации SGD

Momentum

Данный метод добавляет градиентному спуску дополнительный аргумент, который задает некую инерцию движению. Таким образом алгоритм может совершать неоптимальные шаги, однако на практике это оказывается достаточно эффективным.



Nesterov

Данный алгоритм является модификацией предыдущего. Для этого он вычисляет градиент не в текущей точке, а в точке, в которую планирует пойти алгоритм следуя инерции.

AdaGrad

Данная модификация позволяет выбрать размер шага, учитывая ситуацию, когда алгоритм выходит на плато. В таком случае он замедляет уменьшение шага, но не останавливает его, чтобы алгоритм мог сойтись. В общем случае он делит шаг на норму сумм градиентов.

RMSProp

Эта модификация AdaGrad делит не просто на сумму, а на усредненную норму градиентов. Это позволяет не уменьшать шаг так быстро.

Adam

Этот алгоритм является обьединением всех предыдущих оптимизаций. Как правило он считается одним из лучших в SGD, однако его использование в силу множества дополнительных вычислений требует большего числа ресурсов.

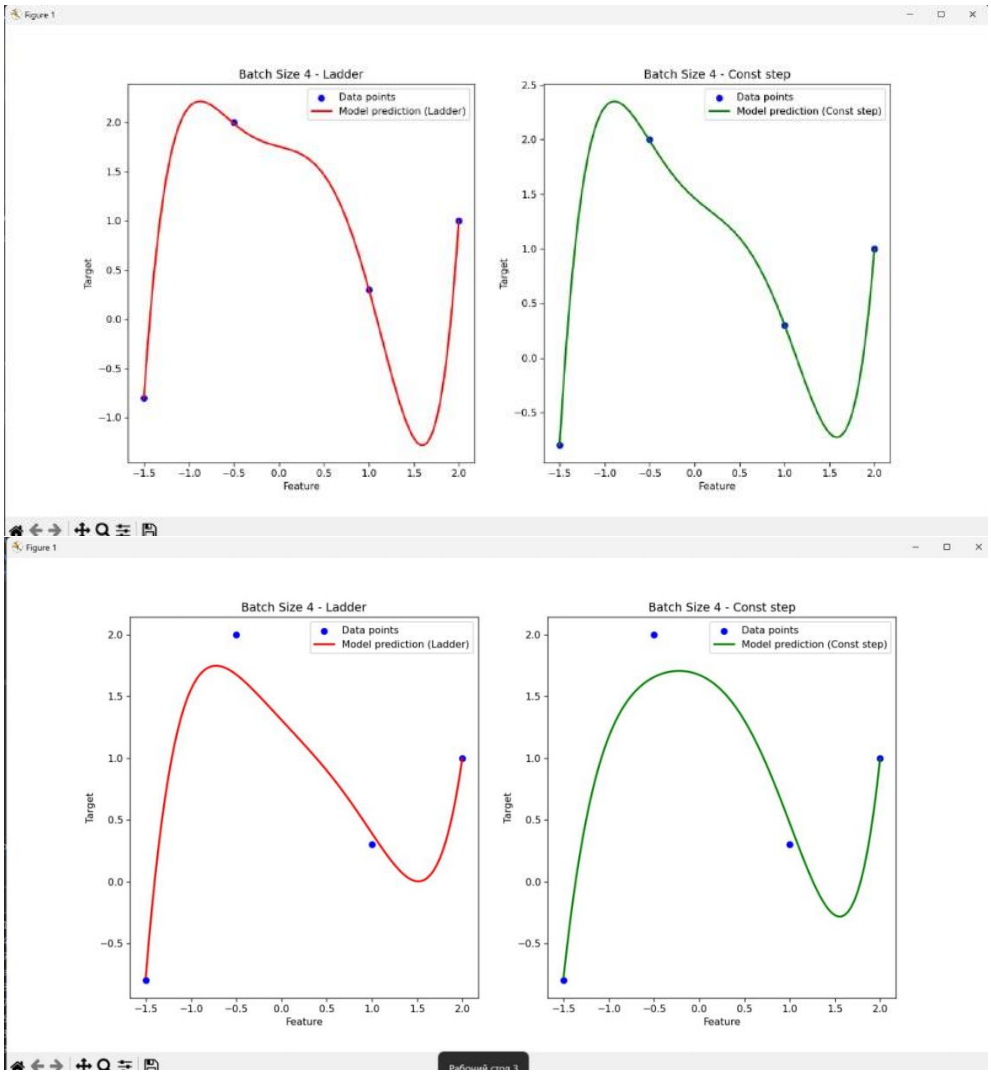
Анализ скорости работы модификаций

	SGD	Momentum	Nesterov	AdaGrad	RMSProp	Adam
Количество итераций	7063	5576	2434	4095	1219	637
Количество арифм. операций	29134	37752	100120	62148	40422	20101
Скорость работы	108.826	32.528	84.937	60.248	16.013	12.989

В итоге наилучшую итеративную производительность показал, закономерно, adam. Однако в силу большого количества вычислений. В свою очередь сопоставимую с ним скорость в данном случае показал RMSProp. Стоит понимать, что каждая оптимизация более оптимальна для своего класса задач, из-за чего важно выбирать метод исходя из поставленной цели.

Регуляризация

Регуляризация используется для упрощения итоговой функции. Часто, особенно на больших размерностях, SGD для полиномиальной регрессии может выдавать огромные по модулю результаты, из-за чего итоговая функция ведет себя более скачкообразно и непредсказуемо. Регуляризация добавляет в качестве параметра минимизации норму результата умноженную на коэффициент регуляризации.
L1 регуляризация добавляет к параметру минимизации октаэдрическую норму результата. L2 же добавляет евклидову норму результата. Elastic же добавляет оба параметра.



На графиках отчетливо видны плюсы и минусы регуляризации - график становится более гладким и предсказуемым, однако теряется точность

Метод опорных векторов

Задачей оптимизации в методе опорных векторов является поиск гиперплоскости, которая максимально разделяет данные двух классов. Метод опорных векторов выбирает ту гиперплоскость, которая максимизирует так называемый отступ между классами. Рассмотрим задачу поиска прямой, разделяющей точки плоскости на две группы. Ниже приведено решение задачи при помощи SVC из библиотеки sklearn: