УДК 004.8

ПЫЛОВ П. А., аспирант гр. МРа-231 (КузГТУ)
Научный руководитель ПИМОНОВ А.Г., д.т.н., доцент (КузГТУ)
г. Кемерово

МАТЕМАТИЧЕСКАЯ ОПТИМИЗАЦИЯ ШАГОВОГО РЕГРЕССИОННОГО АЛГОРИТМА ДЛЯ УСКОРЕННОГО ОПРЕДЕЛЕНИЯ ГЛОБАЛЬНОГО МИНИМУМА

Регрессия – это один из представителей семейства алгоритмов жадного добавления [1]. От всего семейства алгоритмов полного перебора шаговая регрессия выгодно отличается тем, что её применимость не является узкой. Например, в том случае, если в наборе данных количество признаков будет достигать 1000 (а оптимальный состав признаков в алгоритмах полного перебора равен ~100), то модели полного перебора будут очень долго проходить обучение, причём вне зависимости от типа предоставленного им аппаратного обеспечения и вычислительных мощностей [2].

Неэффективность алгоритмов полного перебора вынуждает искать новую эвристику для создания иной концептуальной модели машинного обучения. Первоначальной целью шаговой регрессии в формализованном математическом языке была идея поиска оптимума параметров обобщающей функции — ещё более быстрого, чем в алгоритмах полного перебора, но за счет некоторой дополнительной аппроксимации. В общем виде эту задачу можно описать следующим образом (1).

$$J_0: \emptyset;\ Q^* \coloneqq Q(\emptyset); \quad (\mathbf{1})$$

В условиях (1) поставлена цель оптимизировать Q(J), где J – это подмножество из конечного множества. Эта задача NP-трудная, то есть в общем случае необходимо перебрать все 2^N вариантов, чтобы гарантированно получить решение. Однако на практике приходится создавать оптимизацию, чтобы алгоритмическая сложность решения не вызывала больших временных затрат, требуемых на поиск ответа задачи.

Программная (алгоритмическая) аппроксимация основана на следующих знаниях:

1. Данные имеют нижнюю огибающую, которая, в свою очередь, имеет минимум. Нам нужно как можно скорее приближенно выяснить форму нижней огибающей функции данных;
2. Предположение о том, что если добавить/исключить один признак из данных, то функция Q(J) изменится не очень сильно. Это и есть аппроксимация: если мы будем совершать локальные изменения множества J, то таким образом будем приближаться к некоторому оптимуму искомой функции.

Самый очевидный и простой способ реализовать концепт шаговой регрессии — это поочередное добавление признаков в цикле для всех $j = 1, \dots, n,$ где $j -$ это сложность наборов.

Находим признак, который наиболее выгоден для добавления к исходному набору (то есть к приближению) согласно ограничениям (2):

$$f^*: arg\ min_{f\ \epsilon\ F \setminus J_{j-1}} Q\left(J_{j-1} \cup \{f\}\right) \quad (2)$$

После этого, согласно условиям (2) добавления (f проходит всё множество признаков F $f\ \epsilon\ F \setminus J_{j-1}$), добавляем признак в набор по новым условиям (3):

$$\begin{cases} J_j \coloneqq J_{j-1} \cup \{f^*\}; \\ \text{если } Q(J_j) < Q^*, \text{то } j^* \coloneqq j;\ Q^* \coloneqq Q(J_j); \\ \text{если } j - j^* \geq d, \text{то вернуть } J_{j^*}; \end{cases} \quad (3)$$

Из ограничений **(3)** выходит следующее: если в случае полного перебора для каждого значения j строятся все возможные наборы признаков с данной мощностью, то здесь **(3)** совершается гораздо меньше математических операций: в частности, строятся не все наборы, а только те, которые появляются за счет добавления нового признака $J_{j-1} \cup \{f\}$. Таким образом, экономия ресурсов на каждом шаге составляет N – J.

Главной целью условий (3) является поиск самой нижней точки огибающей функции данных (глобального минимума множества). При этом эвристики (2) и (3) позволяют нам в программном виде добиваться того, чтобы в явном виде это множество никогда не строить [3]. Преимущества метода шаговой регрессии состоят в том, что наиболее близкий к глобальному минимум (т.е. минимум ошибки алгоритма) будет найден всего за несколько шагов, причем сам шаг возможно регулировать со стороны исследователя. Пример определения такого минимума в данных приведен на рисунках 1 – 8.

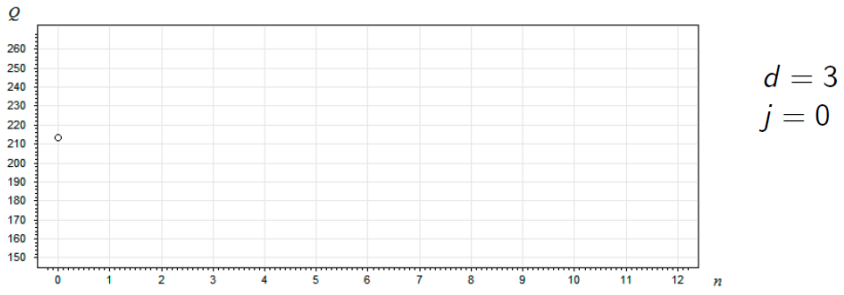

Рисунок 1. Логика построения шаговой регрессии; параметры d = 3, j = 0

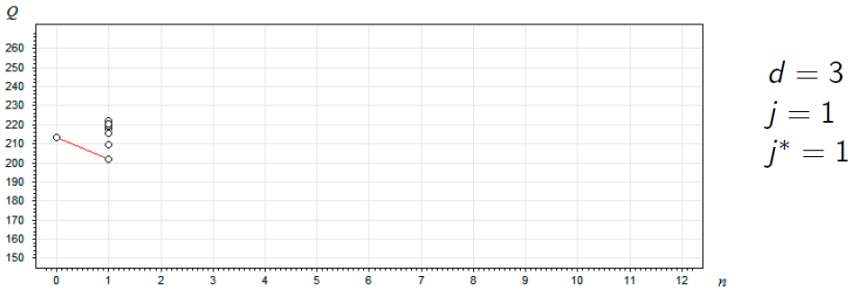

Рисунок 2. Логика построения шаговой регрессии; параметры d = 3, j = 1, $j^* = 1$

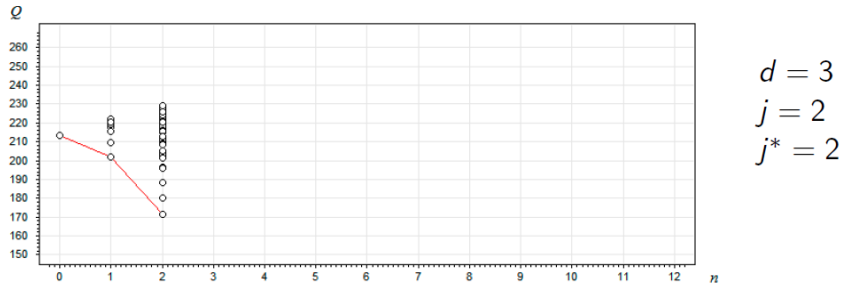

Рисунок 3. Логика построения шаговой регрессии; параметры d = 3, j = 2, $j^* = 2$

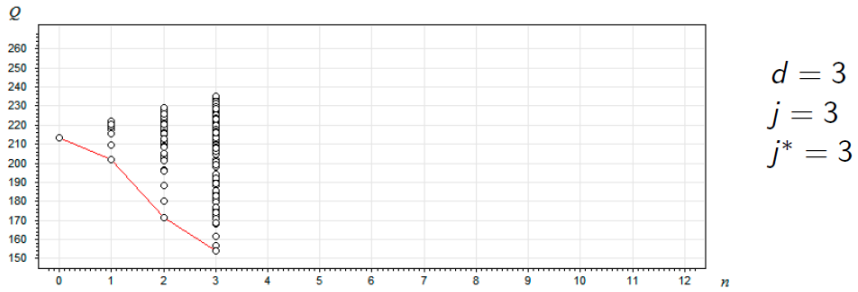

Рисунок 4. Логика построения шаговой регрессии; параметры d = 3, j = 3, $j^* = 3$

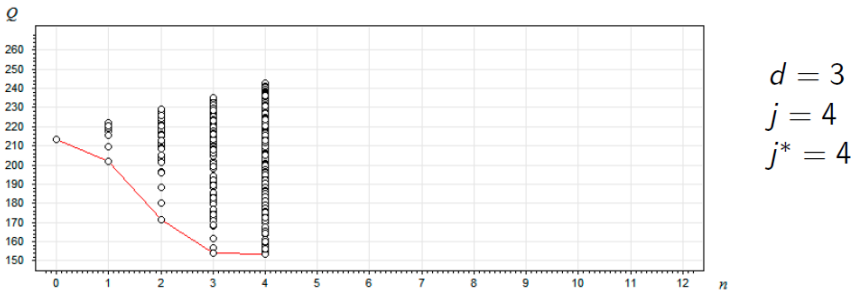

Рисунок 5. Логика построения шаговой регрессии; параметры d = 3, j = 4, $j^* = 4$

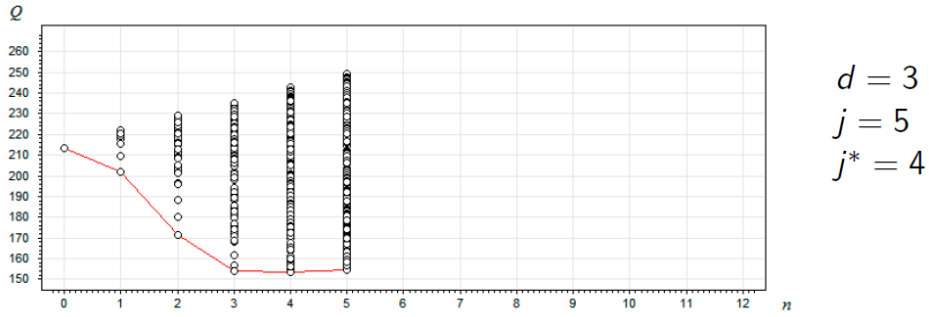

Рисунок 6. Логика построения шаговой регрессии; параметры d = 3, j = 5, $j^* = 4$

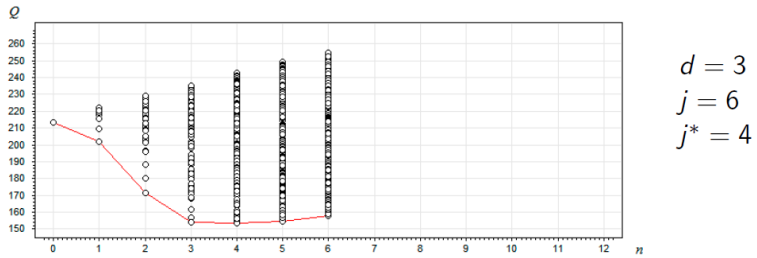

Рисунок 7. Логика построения шаговой регрессии; параметры d = 3, j = 6, $j^* = 4$

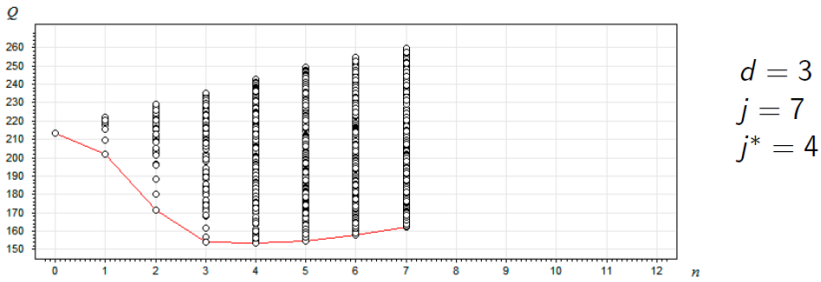

Рисунок 8. Логика построения шаговой регрессии; параметры d = 3, j = 7, $j^* = 4$

Из рисунков 1-8 очевиден факт того, что при обходе набора данных по шагам следования n (до значения $n = 7$) глобальный минимум был определен в точке $n = 4$. Отметим, что по сравнению с алгоритмами полного перебора алгоритмическая сложность снижается с квадратичной до полиномиальной [3–5], что крайне важно в прикладных задачах машинного обучения. Программная реализация алгоритма при этом напрямую зависит от его математической составляющей, поэтому данные особенности следует учитывать при большом количестве признаков обрабатываемой информации [4].

Список литературы:

1. Richard Moore. Python Machine Learning - O'Reilly Media. 2019. – 254 с.
2. Томас Кормен, Чарльз Лейзерсон. Алгоритмы: построение и анализ, 3-е издание – М.: ООО И.Д. Вильямс. 2013. – 1328 с.
3. Allen Downey, Jeffrey Elkner, Chris Meyers. How to Think Like a Computer Scientist - Green Tea Press, 2008. – 250 с.

4. Дж. Клейнберг, Е. Тардос. Алгоритмы: Разработка и применение – СПб.: Питер. 2016 – 800 с.
5. Майкл Солтис. Введение в анализ алгоритмов – СПб.: ДМК. 2019. – 269 с.