

# Машинное обучение

## Теоретический минимум

ЭФ МГУ

2020

## 1 | Матричное дифференцирование

## 2 | Градиентный спуск

**Определение 1** (Градиентный спуск). *метод нахождения локального экстремума (минимума или максимума) функции с помощью движения вдоль градиента.*

Для минимизации функции в направлении градиента используются методы одномерной оптимизации, например, метод золотого сечения. Также можно искать не наилучшую точку в направлении градиента, а какую-либо лучше текущей.

Наиболее простой в реализации из всех методов локальной оптимизации. Имеет довольно слабые условия сходимости, но при этом скорость сходимости достаточно мала (линейна). Шаг градиентного метода часто используется как часть других методов оптимизации, например, метод Флетчера — Ривса.

### 2.1 Gradient Descent

Основное свойство антиградиента — он указывает в сторону наискорейшего убывания функции в данной точке. Соответственно, будет логично стартовать из некоторой точки, сдвинуться в сторону антиградиента, пересчитать антиградиент и снова сдвинуться в его сторону и т.д. Запишем это более формально. Пусть  $w^{(0)}$  — начальный набор параметров (например, нулевой или сгенерированный из некоторого случайного распределения). Тогда градиентный спуск состоит в повторении следующих шагов до сходимости:

$$w^{(k)} = w^{(k-1)} - \eta_k \nabla Q(w^{(k-1)}). \quad (1)$$

Здесь под  $Q(w)$  понимается значение функционала ошибки для набора параметров  $w$ .

Через  $\eta_k$  обозначается длина шага, которая нужна для контроля скорости движения. Можно делать её константной:  $\eta_k = c$ . При этом если длина шага слишком большая, то есть риск постоянно «перепрыгивать» через точку минимума, а если шаг слишком маленький, то движение к минимуму может занять слишком много итераций. Иногда длину шага монотонно уменьшают по мере движения — например, по простой формуле

$$\eta_k = \frac{1}{k}.$$

## 2.2 Stochastic GD

Проблема метода градиентного спуска (1) состоит в том, что на каждом шаге необходимо вычислять градиент всей суммы (будем его называть полным градиентом):

$$\nabla_w Q(w) = \sum_{i=1}^{\ell} \nabla_w q_i(w).$$

Это может быть очень трудоёмко при больших размерах выборки. В то же время точное вычисление градиента может быть не так уж необходимо — как правило, мы делаем не очень большие шаги в сторону антиградиента, и наличие в нём неточностей не должно сильно сказаться на общей траектории. Опишем несколько способов оценивания полного градиента.

Оценить градиент суммы функций можно градиентом одного случайно взятого слагаемого:

$$\nabla_w Q(w) \approx \nabla_w q_{i_k}(w),$$

где  $i_k$  — случайно выбранный номер слагаемого из функционала. В этом случае мы получим метод **стохастического градиентного спуска** (stochastic gradient descent, SGD):

$$w^{(k)} = w^{(k-1)} - \eta_k \nabla q_{i_k}(w^{(k-1)}).$$

Для выпуклого и гладкого функционала может быть получена следующая оценка:

$$[Q(w^{(k)}) - Q(w^*)] = O(1/\sqrt{k}).$$

Таким образом, метод стохастического градиента имеет менее трудоемкие итерации по сравнению с полным градиентом, но и скорость сходимости у него существенно меньше.

Отметим одно важное преимущество метода стохастического градиентного спуска. Для выполнения одного шага в данном методе требуется вычислить градиент лишь одного слагаемого — а поскольку одно слагаемое соответствует ошибке на одном объекте, то получается, что на каждом шаге необходимо держать в памяти всего один объект из выборки. Данное наблюдение позволяет обучать линейные модели на очень больших выборках: можно считывать объекты с диска по одному, и по каждому делать один шаг метода SGD.

## 3 | Настройка гиперпараметров с помощью кросс-валидации

**Определение 2** (Cross-validation). *метод оценки аналитической модели и её поведения на независимых данных.*

При оценке модели имеющиеся в наличии данные разбиваются на  $k$  частей. Затем на  $k-1$  частях данных производится обучение модели, а оставшаяся часть данных используется для тестирования. Процедура повторяется  $k$  раз; в итоге каждая из  $k$  частей данных используется для тестирования. В результате получается оценка эффективности выбранной модели с наиболее равномерным использованием имеющихся данных.

Обычно кросс-валидация используется в ситуациях, где целью является предсказание, и хотелось бы оценить, насколько предсказывающая модель способна работать на практике. Один цикл кросс-валидации включает разбиение набора данных на части, затем построение модели на одной части (называемой тренировочным набором), и валидация модели на другой части (называемой тестовым набором). Чтобы уменьшить разброс результатов, разные циклы кросс-валидации проводятся на разных разбиениях, а результаты валидации усредняются по всем циклам.