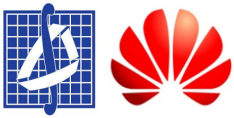
Регрессия и стохастический градиентный спуск

Необходимые сведения

Бабин Д.Н., Иванов И.Е., Петюшко А.А.

кафедра Математической Теории Интеллектуальных Систем

30 октября 2020 г.



План

1. Регрессия
2. Коэффициент детерминации
3. Стохастический градиентный спуск
4. Численная производная
5. Итераторы

Постановка задачи и допущения

- **Обучающая выборка**: $X_{train} = \{(x^{(1)}, y_1), \ldots, (x^{(N)}, y_N)\}$
- **Пространства признаков и ответов**: $X = \mathbb{R}^n$, $Y = \mathbb{R}$
- $a(x) = f_w(x) = w_0 + w_1 x_1 + w_2 x_2 + ... + w_n x_n$, где $w = (w_0, w_1, ..., w_n)^T \in \mathbb{R}^{n+1}$ — параметры модели.
- Удобно писать в векторном виде

$$a(x) = w^T \cdot x,$$

где $x = (x_0, x_1, ..., x_n)^T$ и $x_0 = 1$.

Линейная регресси: аналитическое решение

Метод наименьших квадратов

- **Функция потерь**: $L(w, X_{train}) = MSE(w, X_{train}) = \frac{1}{N}\sum_i (w^T \cdot x^{(i)} - y_i)^2$
- **Задача**: найти $\hat{w} = \arg\min_w (L(w, X_{train}))$

Теорема

Решением задачи $\arg\min_w (\frac{1}{N}\sum_i (w^T \cdot x^{(i)} - y_i)^2)$ является $\hat{w} = (X^T X)^{-1} \cdot X^T \cdot y$, где $X_{i,j} = x_j^{(i)}$, $y = (y_1, ..., y_N)$.

Гребневая регрессия

L2-регуляризация

- **Функция потерь**:
  $L(w, X_{train}) = MSE(w, X_{train}) + \gamma \sum_{i=0}^{n} w_i^2 = \sum_i (w^T \cdot x^{(i)} - y_i)^2 + \alpha \sum_{i=0}^{n} w_i^2$
- Перенормировка: $\alpha = N\gamma$
- **Задача**: найти $\hat{w} = \arg\min_w (L(w, X_{train}))$

Теорема

Решением задачи $\arg\min_w (\sum_{i=1}^{\ell} (w^T \cdot x^{(i)} - y_i)^2 + \alpha \sum_{i=0}^{n} w_i^2)$ является
$\hat{w} = (X^T X + 2\alpha I_n)^{-1} \cdot X^T \cdot y$, где $X_{i,j} = x_j^{(i)}$, $y = (y_1, ..., y_\ell)$, I_n — единичная матрица.

LASSO

L1-регуляризация

- **Функция потерь**:

$$L(w, X_{train}) = MSE(w, X_{train}) + \gamma \sum_{i=0}^{n} |w_i| = \sum_{i} (w^T \cdot x^{(i)} - y_i)^2 + \alpha \sum_{i=0}^{n} |w_i|$$

- Перенормировка: $\alpha = N\gamma$
- **Задача**: найти $\hat{w} = \arg\min_{w}(L(w, X_{train}))$

Свойства

- Нет аналитического решения

Elastic Net

L1-регуляризация и L2-регуляризация

- **Функция потерь**: $L(w, X_{train}) = MSE(w, X_{train}) + \gamma_1 \sum_{i=0}^{n} |w_i| + \gamma_2 \sum_{i=0}^{n} w_i^2 =$

  $= \sum_i (w^T \cdot x^{(i)} - y_i)^2 + \alpha \sum_{i=0}^{n} |w_i| + \beta \sum_{i=0}^{n} w_i^2$
- Перенормировка: $\alpha = N\gamma_1, \beta = N\gamma_2$
- **Задача**: найти $\hat{w} = \arg\min_w (L(w, X_{train}))$

Свойства

- Нет аналитического решения

Коэффициент детерминации

- Пусть $Y = \{y_1, \ldots, y_N\}$ — множество правильных ответов
- $Y_{pred} = \{y_{pred1}, \ldots, y_{predN}\}$ — множество предсказанных ответов

Коэффициент детерминации

R^2-коэффициент (определяющий качество предсказания):

$$R^2 = 1 - \frac{\sum\limits_i (y_i - y_{predi})^2}{\sum\limits_i (y_i - \frac{1}{N}\sum\limits_j y_j)^2}$$

Замечание. $-\infty < R^2 \leq 1$. Идеальное предсказание дает $R^2 = 1$.

Классический градиентный спуск

- **Функция потерь** для линейной регрессии (без регуляризации):
  $L(w, X_{train}) = \frac{1}{N}\sum_i (w^T \cdot x^{(i)} - y_i)^2 = \frac{1}{N}\sum_i L(w, x^{(i)}) = \frac{1}{N}\sum_i L_i(w)$
- **Задача**: минимизировать $L(w, X_{train})$ путем обучения весов w: $L(w, X_{train}) \to \min_w$

Численная оптимизация методом градиентного спуска

- **Начальное приближение**: $w^{(0)} := 0$
- **Итерация алгоритма**: $w^{(t+1)} := w^{(t)} - \eta \cdot \nabla_w L(w^{(t)}, X_{train})$
- **Градиентный шаг**: η

Проблема: сложно считать в условиях большого количества объектов в обучающей выборке.

Стохастический градиентный спуск

Алгоритм стохастического градиентного спуска

- Инициализация весов $w^{(0)}$
- Инициализация функции потерь $L(w^{(0)}, X_{train}) := \frac{1}{N} \sum_i L_i(w^{(0)})$

Итерации

- Выбор объекта $x_i \in X^m$ (например, случайным образом)
- Вычисление ошибки на данном объекте: $L_i(w^{(t)})$
- Шаг градиентного спуска: $w^{(t+1)} := w^{(t)} - \eta \cdot \nabla_w L_i(w^{(t)})$

Вариативность SGD

Инициализация

- $w_j = 0$
- $w_j = rand(-\frac{1}{2n}, \frac{1}{2n})$

Пакетный SGD

Идея: на каждом шаге использовать более надежную оценку градиента не на одном примере, а на нескольких

Итерации

- Выбор подмножества объектов мощности $1 < k < N$: $J = \{i_1, \ldots, i_k\}$
- Вычисление ошибки на этих объектах: $L_{i_1}(w^{(t)}), \ldots, L_{i_k}(w^{(t)})$
- Шаг градиентного спуска: $w^{(t+1)} := w^{(t)} - \eta \cdot \frac{1}{k} \sum_{j=1}^{k} \nabla_w L_{i_j}(w^{(t)})$

Численный градиент

Численная производная

- Пусть $x \in \mathbb{R}$
- Используем разложение Тейлора до первого порядка: $f(x+\delta) \approx f(x) + \delta f'(x)$
- Производная с помощью конечных разностей первого порядка: $f'(x) \approx \frac{f(x+\delta)-f(x)}{\delta}$
- Для более надежной оценки производной: $\delta \to 0, \delta > 0$

Численный градиент

- Пусть $x = (x_1, \ldots, x_n) \in \mathbb{R}^n$
- Тогда градиентом $\nabla f(x)$ называется вектор $\nabla f(x) = (\frac{\partial f(x)}{\partial x_1}, \ldots, \frac{\partial f(x)}{\partial x_n})$
- Частная производная: $\frac{\partial f(x)}{\partial x_i} \approx \frac{f(x+\delta e_i)-f(x)}{\delta}$, где $e_i = (0, \ldots, 0, 1, 0, \ldots, 0)$ — единичный базисный вектор с 1 на месте i

Итераторы в Python

- Нужны для упрощения навигации по элементам объекта (некоторая коллекция)
- Применяются в цикле “for i in iterator:”
- Имеют достаточно строгий синтаксис
- В задачах подразумевается, что мы будем ходить по обучающей выборке некоторое количество полных раз по кругу, после чего завершаем работу

Итераторы в Python

```
class MyIterator:
        def __iter__(self):
                return self
        def __init__(self, limit):
                self.limit = limit
                self.counter = 0
        def __next__(self):
                if self.counter < self.limit:
                        self.counter += 1
                        return 1
                else:
                        raise StopIteration
iterate = MyIterator(3)
for i in iterate:
        print(i)
```

Удачи в решении задач!