Простая линейная регрессия

$$Q(\beta_0, \beta_1) = \frac{1}{n} \sum_{i=1}^{n} Q_i(\beta_0, \beta_1) = \frac{1}{n} \sum_{i=1}^{n} ((\beta_0 + \beta_1 x_i) - y_i)^2 \to \min_{\beta_0, \beta_1}$$

Стохастический градиентный спуск:

1. Выбрать начальный вектор параметров (β_0, β_1) и интенсивность обучения η

2. Повторять до сходимости:

   (a) Случайно перемешать выборочные данные

   (b) Для $i = 1, 2, \ldots, n$ выполнить

$$\begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix} := \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix} - \eta \begin{bmatrix} 2(\beta_0 + \beta_1 x_i - y_i) \\ 2x_i(\beta_0 + \beta_1 x_i - y_i) \end{bmatrix}$$

$\eta = 0.001$, ~ 10000 iterations

x_i	y_i	β_0	β_1
–	–	1	1
10.0	8.04	0.99408	0.9408
8.0	6.95	0.99093904	0.91567232
13.0	7.58	0.98030968	0.77749066
9.0	8.81	0.98197423	0.7924716
11.0	8.33	0.97923591	0.76235004
14.0	9.96	0.97585163	0.71497022
6.0	7.24	0.97980029	0.73866214
4.0	4.26	0.98045139	0.74126655
12.0	10.84	0.98238009	0.76441095
7.0	4.82	0.97935358	0.74322536
5.0	5.68	0.98132262	0.75307055
		⋮	⋮
		3.03339774	0.50766523

Стохастический градиентный спуск (SGD)

- Теорема сходимости. При убывающем η и выполнении некоторых слабых условий SGD сходится почти наверно к глобальному минимуму целевой функции в случае выпуклой или псевдовыпуклой целевой функции и сходится почти наверно к локальному минимуму в других случаях.

- Применение SGD: линейная регрессия, логистическая регрессия, метод опорных векторов, искусственные нейронные сети.

Варианты SGD

- Averaged stochastic gradient descent
- AdaGrad (Adaptive)
- Adam (Adaptive moment estimation)
- Kalman-based Stochastic Gradient Descent (kSGD)

`https://en.wikipedia.org/wiki/Stochastic_gradient_descent`

Пример применения линейной регрессии
(из документации к scikit-learn)

```
import matplotlib.pyplot as plt
import numpy as np
from sklearn import datasets, linear_model
from sklearn.metrics import mean_squared_error, r2_score

# Load the diabetes dataset
diabetes = datasets.load_diabetes()
# Use only one feature
diabetes_X = diabetes.data[:, np.newaxis, 2]
# Split the data into training/testing sets
diabetes_X_train = diabetes_X[:-20]
diabetes_X_test = diabetes_X[-20:]
```

```
# Split the targets into training/testing sets
diabetes_y_train = diabetes.target[:-20]
diabetes_y_test = diabetes.target[-20:]

# Create linear regression object
regr = linear_model.LinearRegression()
# Train the model using the training sets
regr.fit(diabetes_X_train, diabetes_y_train)
# Make predictions using the testing set
diabetes_y_pred = regr.predict(diabetes_X_test)

# The coefficients
print('Coefficients: \n', regr.coef_)
# The mean squared error
print("Mean squared error: %.2f"
```

```
% mean_squared_error(diabetes_y_test, diabetes_y_pred))
# Explained variance score: 1 is perfect prediction
print('Variance score: %.2f'
% r2_score(diabetes_y_test, diabetes_y_pred))
# Plot outputs
plt.scatter(diabetes_X_test, diabetes_y_test, color='black')
plt.plot(diabetes_X_test, diabetes_y_pred,
color='blue', linewidth=3)
plt.xticks(())
plt.yticks(())
plt.show()
```

Требования к проекту

- Описание набора данных
- Что Вы хотите сделать с набором данных
- По крайней мере одна иллюстрация, обосновывающая Ваши намерения
- Какая от этого будет польза (5V – value)
- Презентация 5 мин. макс.

Требования к проекту

- Много самостоятельной работы: поиск массивов данных, анализ методов
- Оценивается не столько содержание, сколько форма (так будет не всегда)
- Представьте, что вы просите инвестиции на свой проект; нужно, чтобы презентация была убедительной для инвесторов
- В принципе, проект можно не делать (-25 баллов)

Где искать вдохновение для проекта

- How to prepare for the data incubator
- Data Sources for Cool Data Science Projects: Part 1