The Experiment Report of Machine Learning

SCHOOL: SCHOOL OF SOFTWARE ENGINEERING

SUBJECT: SOFTWARE ENGINEERING

Author:
Zhaohui Huang

Supervisor:
Qingyao Wu

Student ID：201720145112

Grade: Graduate

December 15, 2017

Linear Regression, Linear Classification and Gradient Descent

Abstract—Gradient descent is a first-order iterative optimization algorithm for finding the minimum of a function. In this paper we use compare the difference between gradient descent and stochastic gradient descent in SVM.

I. Introduction

The main purposes of this report can be concluded as the following:

1. Compare and understand the difference between gradient descent and stochastic gradient descent.
2. Compare and understand the differences and relationships between Logistic regression and linear classification.
3. Further understand the principles of SVM and practice on larger data.

II. Methods and Theory

The loss function of logistic regression is:

$$L_{reg} = -\frac{1}{N}\sum_{i=1}^{N} y^{(i)} log(h(x^{(i)})) + (1-y^{(i)}) log(1-h(x^{(i)})),$$

$$\text{where } h(x^{(i)}) = \frac{1}{1+e^{-w^T x}}$$

The corresponding gradient with respect to weight in logis tic regression

$$\frac{\partial L_{reg}}{\partial w} = \frac{1}{N}\sum_{i=1}^{N}(h(x^{(i)}) - y^{(i)}) * x_j^{(i)}$$

The loss function of linear classification, e.g.Support Vector Machine (SVM):

$$L_{cls} = \frac{\|w\|^2}{2} + C\frac{1}{N}\sum_{i=1}^{N} max(0, 1 - y_i(w^T x_i + b))$$

Stochastic Gradient Descent (SGD). Stochastic gradient descent (SGD) in contrast performs a parameter update for each training example x (i) and y (i) :

$$\theta = \theta - \eta \nabla_\theta L(\theta; x^{(i)}; y^{(i)})$$

NAG(Nesterov accelerated gradient)

Nesterov accelerated gradient (NAG) is a way to give our momentum term this kind of prescience. We know that we will use our momentum term γv_{t-1} to move the parameters θ. Computing $\theta - \gamma v_{t-1}$ thus gives us an approximation of the next position of the parameters (the gradient is missing for the full update), a rough idea where our parameters are going to be. We can now effectively look ahead by calculating the gradient not w.r.t. to our current parameters θ but w.r.t. the approximate future position of our parameters:

$$\begin{aligned} v_t &= \gamma v_{t-1} + \eta \nabla_\theta J(\theta - \gamma v_{t-1}) \\ \theta &= \theta - v_t \end{aligned}$$

RMSProp

RMSprop and Adadelta have both been developed independently around the same time stemming from the need to resolve Adagrad's radically diminishing learning rates. RMSprop in fact is identical to the first update vector of Adadelta that we derived above:

$$\begin{aligned} E[g^2]_t &= 0.9E[g^2]_{t-1} + 0.1g_t^2 \\ \theta_{t+1} &= \theta_t - \frac{\eta}{\sqrt{E[g^2]_t + \epsilon}} g_t \end{aligned}$$

AdaDelta

Adadelta is an extension of Adagrad that seeks to reduce its aggressive, monotonically decreasing learning rate. Instead of accumulating all past squared gradients, Adadelta restricts the window of accumulated past gradients to some fixed size w. The steps of updating in the following:

$$\begin{aligned} g_t &= \nabla L(\theta_{t-1}) \\ G_t &= \gamma G_t + (1-\gamma) g_t \odot g_t \\ \Delta\theta_t &= -\frac{\sqrt{\Delta + \epsilon}}{\sqrt{G_t + \epsilon}} \odot g_t \\ \theta_t &= \theta_{t-1} + \Delta\theta_t \\ \Delta_t &= \gamma \Delta_{t-1} + (1-\gamma)\Delta\theta_t \odot \Delta\theta_t \end{aligned}$$

Adam

Adaptive Moment Estimation (Adam) is another method that computes adaptive learning rates for each parameter. In addition to storing an exponentially decaying average of past squared gradients v t like Adadelta and RMSprop, Adam also keeps an exponentially decaying average of past gradients m t , similar to momentum. We show the updating steps as the following:

$$\begin{aligned} g_t &= \nabla L(\theta_{t-1}) \\ m_t &= \beta_1 m_{t-1} + (1-\beta_1) g_t \\ G_t &= \gamma G_t + (1-\gamma) g_t \odot g_t \\ \alpha &= \eta \frac{\sqrt{1-\gamma^t}}{1-\beta^t} \\ \theta_t &= \theta_{t-1} - \alpha \frac{m_t}{\sqrt{G_t + \epsilon}} \end{aligned}$$

III. EXPERIMENT

Dataset

Experiment uses a9a of LIBSVM Data, including 32561/16281(testing) samples and each sample has 123/123 (testing) features.

Experiment Step

Logistic Regression and Stochastic Gradient Descent

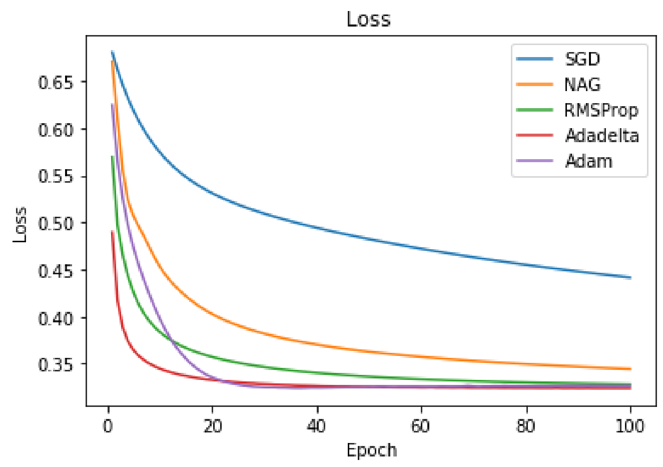
1. Load the training set and validation set.
2. Initalize logistic regression model parameters, you can consider initalizing zeros, random numbers or normal distribution.
3. Select the loss function and calculate its derivation, find more detail in PPT.
4. Calculate gradient toward loss function from partial samples.
5. Update model parameters using different optimized methods(NAG，RMSProp，AdaDelta and Adam).
6. Select the appropriate threshold, mark the sample whose predict scores greater than the threshold as positive, on the contrary as negative.

Linear Classification and Stochastic Gradient Descent

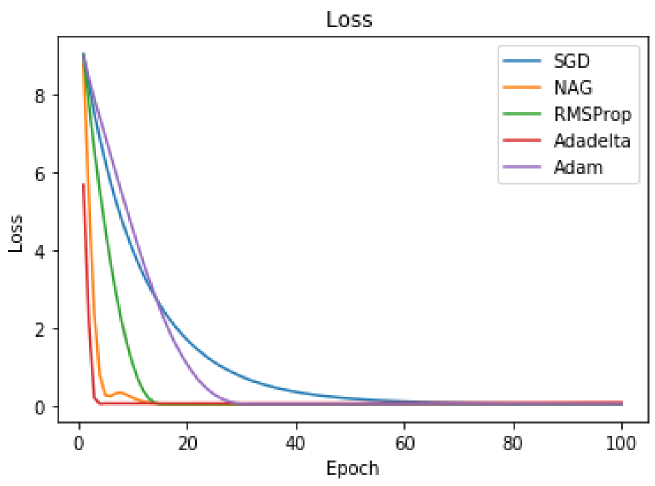
1. Load the training set and validation set.
2. Initalize SVM model parameters, you can consider initalizing zeros, random numbers or normal distribution.
3. Select the loss function and calculate its derivation, find more detail in PPT.
4. Calculate gradient toward loss function from partial samples.
5. Update model parameters using different optimized methods(NAG，RMSProp，AdaDelta and Adam).
6. Select the appropriate threshold, mark the sample whose predict scores greater than the threshold as positive, on the contrary as negative.

Result

Logistic Regression and Stochastic Gradient Descent



Linear Classification and Stochastic Gradient Descent



IV. CONCLUSION

In this paper, we have then investigated algorithms that are most commonly used for optimizing SGD: Nesterov accelerated gradient, Adadelta, RMSprop, Adam. Meanwhile, we conduct some experiments and visualize the results of these optimized methods in logistic regression and linear classification.