华南理工大学

South China University of Technology

The Experiment Report of
Machine Learning

College Software College

Subject Software Engineering

Members 钟恩俊

Student ID 201530613818

E-mail 286843911@qq.com

Tutor Prof.Mingkui Tan

Date submitted 2017.12.05

1. Topic: Logistic Regression, Linear Classification and Stochastic Gradient Descent

2. Time: 2017.12.05

3. Reporter: 钟恩俊

4. Purposes:

Compare and understand the difference between gradient descent and stochastic gradient descent.

Compare and understand the differences and relationships between Logistic regression and linear classification.

Further understand the principles of SVM and practice on larger data.

5. Data sets and data analysis:

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

6. Experimental steps:

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 G 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. Predict under validation set and get the different optimized method loss L_{NAG}, $L_{RMSProp}$, $L_{AdaDelta}$and L_{Adam}.
7. Repeate step 4 to 6 for several times, and drawing graph of L_{NAG}, $L_{RMSProp}$, $L_{AdaDelta}$ and L_{Adam} with the number of iterations.

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 G 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. Predict under validation set and get the different optimized method loss L_{NAG}, $L_{RMSProp}$, $L_{AdaDelta}$ and L_{Adam}.
7. Repeate step 4 to 6 for several times, and drawing graph of L_{NAG}, $L_{RMSProp}$, $L_{AdaDelta}$ and L_{Adam} with the number of iterations.

7. Code:

Logistic Regression Code see in the RegressionExperiment.ipynb
Linear Classification Code see in the ClassificationExperiment.ipynb

8. The initialization method of model parameters:

Logistic Regression:

9. The selected loss function and its derivatives:

Logistic Regression:

Loss function: $-\frac{1}{m}[\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1-y^{(i)}) \log(1-h_\theta(x^{(i)}))]$

Derivatives: $-\frac{1}{m}\sum_{i=1}^{m} \alpha(h_\theta(x^{(i)}) - y^{(i)})x^{(i)}$

Linear Classification

Loss function: hingeloss $\frac{\|w\|^2}{2} + C\sum_{i=1}^{n} max(0, 1-y_i(x_i w+b))$

Derivatives: $g_t = \begin{cases} w + C\sum_{i=1}^{n} -x_i^T y_i & 1-y_i(x_i w+b) \geq 0 \\ w & 1-y_i(x_i w+b) < 0 \end{cases}$

NAG:

$$g_t = \nabla J(\theta_{t-1})$$
$$v_t = \gamma v_{t-1} + \eta g_t$$
$$\theta_t = \theta_{t-1} - v_t$$

RMSProp

$$g_t = \nabla \mathrm{J}(\theta_{t-1})$$
$$G_t = \gamma G_t + (1-\gamma) g_t \odot g_t$$
$$\theta_t = \theta_{t-1} - \frac{\eta}{\sqrt{G_t + \epsilon}} \odot g_t$$

AdaDelta

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

Adam

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

10. Experimental results and curve:

Hyper-parameter selection:

Logistic Regression:
NAG: γ=0.9 η=0.01 $v_t = 0$
RMSProp: γ=0.9 η=0.001
AdaDelta: γ=0.95 $\Delta_t = 0$
Adam: $\beta_t = 0.9\ \gamma = 0.95\ \eta = 0.01\ m_t = 0$

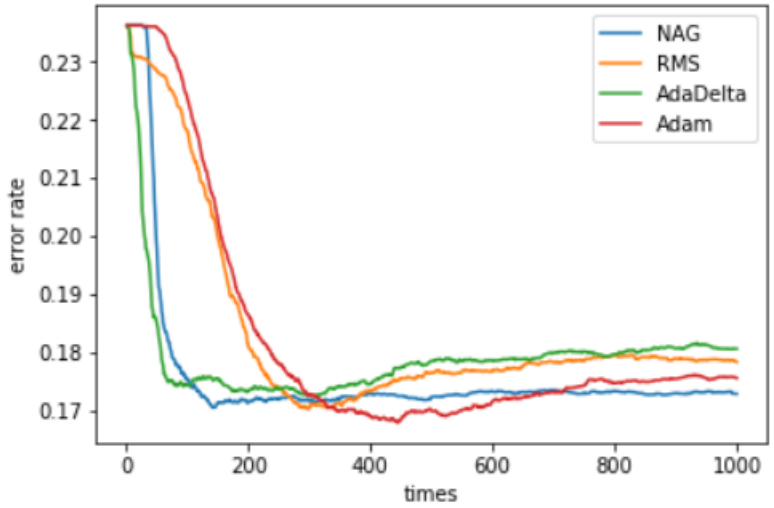
Linear Classification:
NAG: γ=0.9 η=0.01 $v_t = 0$
RMSProp: γ=0.9 η=0.001
AdaDelta: γ=0.95 $\Delta_t = 0$
Adam: $\beta_t = 0.9\ \gamma = 0.95\ \eta = 0.01\ m_t = 0$

Predicted Results (Best Results):

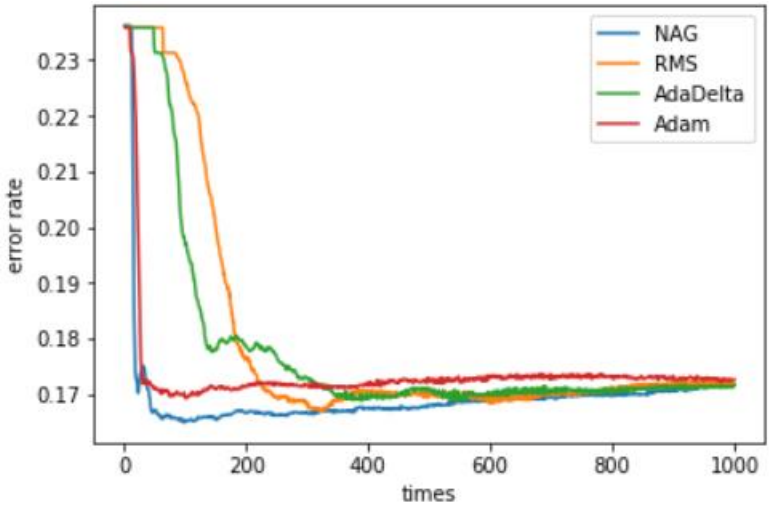
	NAG	RMS	AdaDelta	Adam
Linear Regression	0.1729	0.1783	0.1806	0.1756
Linear Classification	0.1716	0.1716	0.1715	0.1725

Loss curve:

Logistic Regression:



Linear Classification:



11. Results analysis:

Logistic Regression:

NAG with η=0.1 can decline quickly and reach its convergence quickly, but when the η=0.01 it perform better than η=0.1 for the reason that its loss is smaller

RMS with η=0.1 will decline quicker than NAG but the curve will get some fluctuations. So it need smaller η.

AdaDelta can adjust its learning rate itself. So the parameter γ will not influence the error rate when reaching the convergence. I try γ=1 an 0 and it's error rate finally reach the same value (maybe I try too few values). Maybe this prove that it can adjust the learning rate.

Adam also decline quickly. But different parameter may reach convergence in different value.

Linear Classification:

NAG with large η will get convergence in a high value.

RMS is similar to RMS in Linear Regression.

AdaDelta is similar to Linear Regression. The value of γ will not influence the error rate greatly.

Adam is similar to Adam in Linear Regression.

12. Similarities and differences between logistic regression and linear classification：

Similarities: Two models slove the classification problem. Both are linear model.

Differences: Logistic Regression can represent a probability throw mapping features on the sigmoid function. According the threshold classify the data. Linear Classification classify the datas by training a hyperplane to split the datas.

13. Summary:

SGD run quicker than GD. SGD choose one example to compute the gradient to represent the whole gradient. So SGD can improve the algroithm speed. But learning curve using SGD will get some fluctuations, because we may choose the noise as training data. To avoid the noise, we randomly choose a part of data to compute the gradient.