South China University of Technology

《机器学习》课程实验报告

学　　院　软件学院

专　　业　软件工程

组　　员　黄浩填

学　　号　201530611708

邮　　箱　1025686131@qq.com

指导教师　吴庆耀

提交日期　2017 年 12 月 15 日

1. 实验题目：逻辑回归、线性分类与随机梯度下降

2. 实验时间：2017 年 12 月 15 日

3. 报告人：黄浩填

4. 实验目的：

1. 对比理解梯度下降和随机梯度下降的区别与联系。
2. 对比理解逻辑回归和线性分类的区别与联系。
3. 进一步理解 SVM 的原理并在较大数据上实践。

5. 数据集以及数据分析：

实验使用的是 LIBSVM Data 的中的 a9a 数据，包含 32561 / 16281(testing)个样本，每个样本有 123/123 (testing)个属性。

6. 实验步骤：

逻辑回归与随机梯度下降

1. 读取实验训练集和验证集。
2. 逻辑回归模型参数初始化，可以考虑全零初始化，随机初始化或者正态分布初始化。
3. 选择Loss函数及对其求导，过程详见课件ppt。
4. 求得**部分样本**对Loss函数的梯度G。
5. **使用不同的优化方法更新模型参数（NAG，RMSProp，AdaDelta和Adam）。**
6. 选择合适的阈值，将验证集中计算结果**大于阈值的标记为正类，反之为负类**。在验证集上测试并得到不同优化方法的Loss函数值L_{NAG}，$L_{RMSProp}$，$L_{AdaDelta}$和L_{Adam}。
7. 重复步骤4-6若干次，**画出L_{NAG}，$L_{RMSProp}$，$L_{AdaDelta}$和L_{Adam}随迭代次数的变化图。**

线性分类与随机梯度下降

1. 读取实验训练集和验证集。
2. 支持向量机模型参数初始化，可以考虑全零初始化，随机初始化或者正态分布初始化。
3. 选择Loss函数及对其求导，过程详见课件ppt。
4. 求得**部分样本**对Loss函数的梯度G。
5. **使用不同的优化方法更新模型参数（NAG，RMSProp，AdaDelta和Adam）。**
6. 选择合适的阈值，将验证集中计算结果**大于阈值的标记为正类，反之为负类**。在验证集上测试并得到不同优化方法的Loss函数值L_{NAG}，$L_{RMSProp}$，$L_{AdaDelta}$和L_{Adam}。
7. 重复步骤4-6若干次，**画出L_{NAG}，$L_{RMSProp}$，$L_{AdaDelta}$和L_{Adam}随迭代次数的变化图。**

7. 代码内容：

（针对逻辑回归和线性分类分别填写 8-11 内容）

逻辑回归：

lam = 10

```
def loss (X, W, y):
    tmp = X.dot(W) * y * -1.0
    ret = 0
    for i in range(0, X.shape[0]):
        ret += math.log(1.0 + math.exp(tmp[i][0]))
    ret /= X.shape[0]
    global lam
    ret += lam * (np.square(W).sum() - W[W.shape[0]-1][0] * W[W.shape[0]-1][0]) / 2.0
    return ret

def grad (X, W, y):
    tmp = X.dot(W) * y
    #print(X)
    #print(W)
    for i in range(0, tmp.shape[0]):
        tmp[i][0] = 1.0 / (1.0 + math.exp(tmp[i][0]))
    ret = (X * y).T.dot(tmp)
    ret = ret / X.shape[0] * -1.0
    global lam
    ret = ret + lam * W
    ret[ret.shape[0]-1][0] -= W[W.shape[0]-1][0] * lam
    return ret
```

线性回归：

```
C = 0.9

def loss (X, W, y):
    ret = 0
    tmp = X.dot(W) * y
    for i in range(0,tmp.shape[0]):
        if tmp[i][0] < 1 :
            ret += 1 - tmp[i][0]
    global C
    ret *= C
    ret += ((W * W).sum() - W[W.shape[0]-1][0] * W[W.shape[0]-1][0]) / 2
    return ret / X.shape[0]

def grad (X, W, y):
    tmp = X.dot(W) * y
    X_tmp = []
    y_tmp = []
    for i in range(0,tmp.shape[0]):
```

```
        if tmp[i][0] <= 1 :
            X_tmp.append(X[i])
            y_tmp.append(y[i])
    X_tmp = np.array(X_tmp)
    y_tmp = np.array(y_tmp)
    global C
    tmp = X_tmp.T.dot(y_tmp) * C
    ret = W - tmpj
    ret[ret.shape[0]-1][0] = -1.0 * C * y_tmp.sum()
    return ret
```

随机梯度下降：

```
v = np.zeros((X_train.shape[1], 1))
def NAG (X, W, y):
    gamma = 0.9
    eta = 0.001
    global v
    #print(v)
    g = grad(X, W - gamma * v, y)
    v = gamma * v + eta * g
    return v

G1 = np.zeros((X_train.shape[1], 1))
def RMSProp (X, W, y):
    g = grad(X, W, y)
    global G1
    gamma = 0.9
    eta = 0.001
    G1 = gamma * G1 + (1.0 - gamma) * np.square(g)
    return (eta / np.sqrt(G1 + 1e-8)) * g

G2 = np.zeros((X_train.shape[1], 1))
delta = np.zeros((X_train.shape[1], 1))
def AdaDelta (X, W, y):
    g = grad(X, W, y)
    global G2
    global delta
    gamma = 0.95
    G2 = gamma * G2 + (1.0 - gamma) * np.square(g)
    ret = (np.sqrt(delta + 1e-8) / np.sqrt(G2 + 1e-8)) * g
    delta = gamma * delta + (1.0 - gamma) * ret * ret
    return ret

G3 = np.zeros((X_train.shape[1], 1))
```

```
mt = np.zeros((X_train.shape[1], 1))
t = 0
eta = 0.01
def Adam (X, W, y):
    g = grad(X, W, y)
    beta = 0.9
    gamma = 0.999
    eta = 0.001
    global t
    global G3
    global mt
    #global eta
    t += 1
    #print(t)
    #eta /= math.sqrt(t)
    mt = beta * mt + (1.0 - beta) * g
    G3 = gamma * G3 + (1.0 - gamma) * np.square(g)
    alpha = eta * math.sqrt(1.0 - gamma ** t) / (1.0 - beta ** t)
    return alpha * mt / np.sqrt(G3 + 1e-8)
```

8. 模型参数的初始化方法:

全零初始化

9.选择的 loss 函数及其导数:

逻辑回归:

$$J(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n}\log(1+e^{-y_i\cdot\mathbf{w}^\top\mathbf{x}_i}) + \frac{\lambda}{2}||\mathbf{w}||_2^2$$

$$\mathbf{w}' \to \mathbf{w} - \eta\frac{\partial J(\mathbf{w})}{\partial\mathbf{w}} = (1-\eta\lambda)\mathbf{w} + \eta\frac{1}{n}\sum_{i=1}^{n}\frac{y_i\mathbf{x}_i}{1+e^{y_i\cdot\mathbf{w}^\top\mathbf{x}_i}}$$

线性回归:

$$\min_{\mathbf{w},b}\ \frac{\|\mathbf{w}\|^2}{2} + C\sum_{i=1}^{n}\max(0, 1-y_i(\mathbf{w}^\top\mathbf{x}_i+b))$$

$$g_{\mathbf{w}}(\mathbf{x}_i) = \begin{cases} -y_i\mathbf{x}_i & 1 - y_i(\mathbf{w}^\top\mathbf{x}_i + b) >= 0 \\ 0 & 1 - y_i(\mathbf{w}^\top\mathbf{x}_i + b) < 0 \end{cases}$$

$$g_b(\mathbf{x}_i) = \begin{cases} -y_i & 1 - y_i(\mathbf{w}^\top\mathbf{x}_i + b) >= 0 \\ 0 & 1 - y_i(\mathbf{w}^\top\mathbf{x}_i + b) < 0 \end{cases}$$

$$\frac{\partial f(\mathbf{w}, b)}{\mathbf{w}} = \mathbf{w} + C\sum_{i=1}^{N} g_{\mathbf{w}}(\mathbf{x}_i)$$

$$\frac{\partial f(\mathbf{w}, b)}{b} = C\sum_{i=1}^{N} g_b(\mathbf{x}_i)$$

10.实验结果和曲线图：（各种梯度下降方式分别填写此项）

超参数选择：

ε = 1e-8

NAG：γ = 0.9 η = 0.001

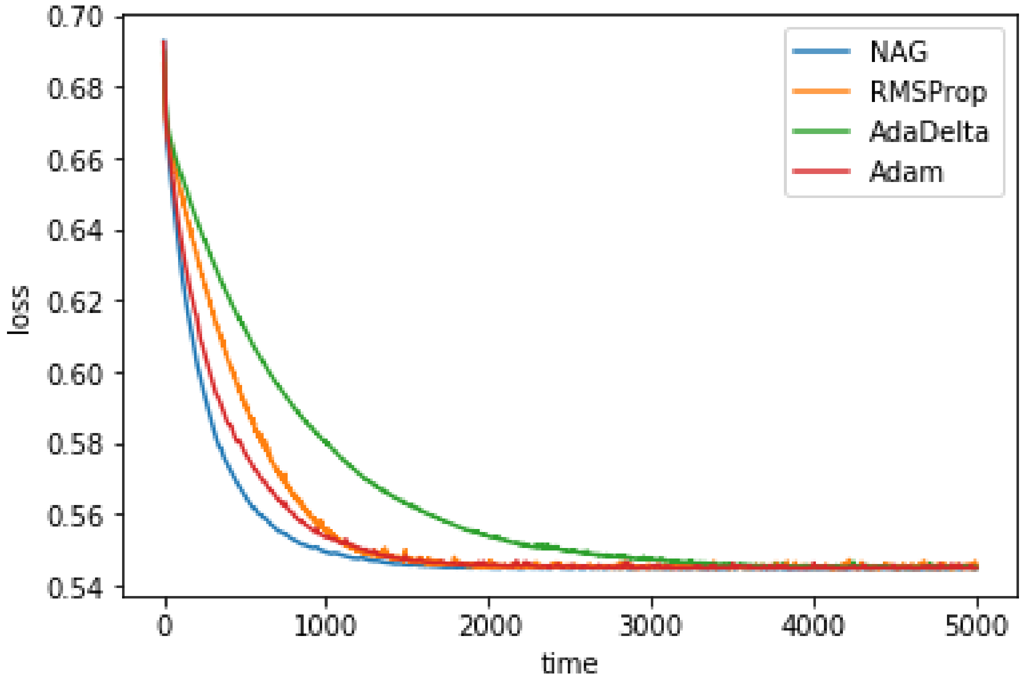
RMSProp：γ = 0.9 η = 0.001

AdaDelta：γ = 0.95

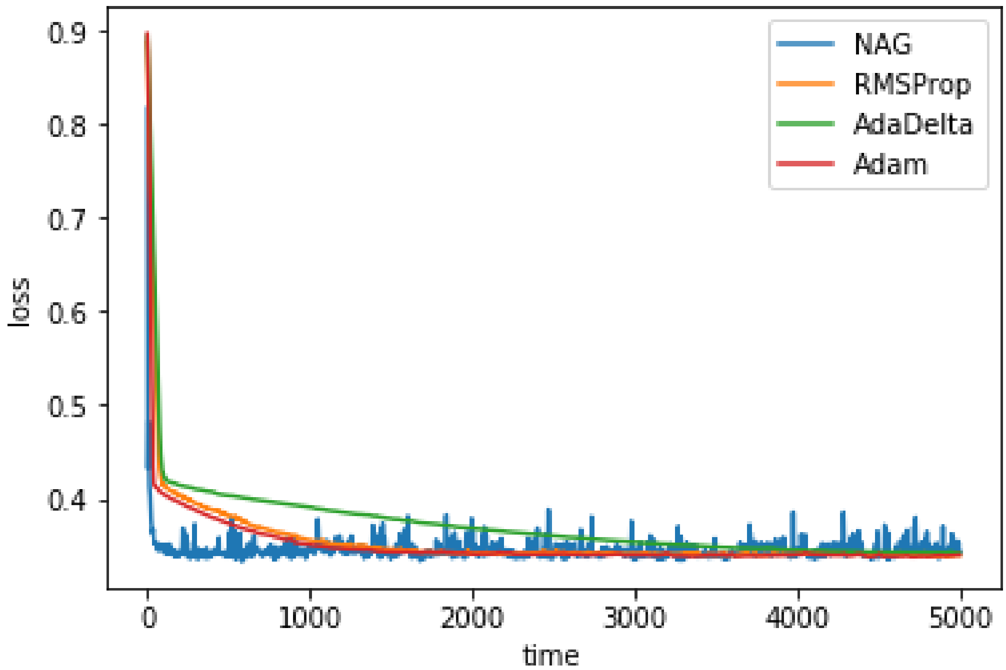
Adam：γ = 0.999 η = 0.001 β = 0.9

loss 曲线图：

逻辑回归：



线性回归：



11.实验结果分析:

Loss 函数收敛达到预期结果，预测值也较为符合。

12.对比逻辑回归和线性分类的异同点:

逻辑回归能够更好地处理非线性分类，有更广的应用范围。逻辑回归相当于给非线性分类的点强行加了一个超平面。

13.实验总结：

逻辑回归与线性分类各有特点，逻辑回归适用范围更广；在测试集较大时，可以使用随即梯度下降快速收敛。