

# 1 Logistic Regression

## 1.1 1

考虑仅有一个样本点的情况, 此时

$$J(\theta) = \log(1 + e^{-y\theta^T x}) \quad (1)$$

考虑

$$\begin{aligned} \frac{\partial J(\theta)}{\partial \theta_i \theta_j} &= \frac{\partial}{\partial \theta_i} \left( \frac{-e^{-y\theta^T x}}{1 + e^{-y\theta^T x}} x_j y \right) \\ &= \frac{e^{-\theta^T xy} (1 + e^{-\theta^T xy}) - (e^{-\theta^T xy})^2}{(1 + e^{-y\theta^T x})^2} x_i x_j y^2 \\ &= \frac{e^{-\theta^T xy}}{(1 + e^{-y\theta^T x})^2} x_i x_j y^2 \end{aligned} \quad (2)$$

对任意矢量  $z$ , 有

$$\begin{aligned} z^T H z &= \sum_i \sum_j H_{ij} z_i z_j = \sum_i \sum_j \frac{e^{-\theta^T xy}}{(1 + e^{-y\theta^T x})^2} y^2 x_i x_j z_i z_j \\ &= \frac{e^{-\theta^T xy}}{(1 + e^{-y\theta^T x})^2} y^2 \sum_i \sum_j x_i x_j z_i z_j \\ &= \frac{e^{-\theta^T xy}}{(1 + e^{-y\theta^T x})^2} y^2 (x^T z)^2 \geq 0 \end{aligned} \quad (3)$$

这一结论很容易扩展到多个样本点, 记

$$J_i(\theta) = \log(1 + e^{-y^{(i)} \theta^T x^{(i)}}) \quad (4)$$

则有

$$J(\theta) = \frac{1}{m} \sum_i^m J_i(\theta) \quad (5)$$

$$H = \frac{1}{m} \sum_i^m H_i \quad (6)$$

因此

$$z^T H z = \frac{1}{m} \sum_i^m z^T H_i z \geq 0 \quad (7)$$

因此这是一个半正定矩阵.

## 1.2 2

Newton 方法的迭代公式为

$$\theta = \theta - H^{-1} \nabla_{\theta} l(\theta) \quad (8)$$

此处

$$\nabla_{\theta} l(\theta) = \frac{1}{m} \sum_{i=1}^m \frac{y^{(i)}}{1 + e^{y^{(i)} \theta^T x^{(i)}}} x^{(i)} \quad (9)$$

$$H = \frac{1}{m} \sum_{i=1}^m \frac{e^{-\theta^T x^{(i)} y^{(i)}} y^{(i)2}}{(1 + e^{-y^{(i)} \theta^T x^{(i)}})^2} x^{(i)} x^{(i)T} \quad (10)$$

写出 MATLAB 代码

```
% Function to calculate the cost in logistic regression
% INPUT: variable x in training set, variable y in training set,
% parameters theta
% OUTPUT: cost

function loss = cost(x, y, theta)
    m = size(x,1);
    loss = sum(-log(sigmoid(x, y, theta))) / m;
end

% Function to calculate the derivative vector of the cost function for
% introducing Newton's method
% INPUT: variable x[m * n_x] in training set, variable y[m * 1] in training set,
% parameters theta[n*x * 1]
% OUTPUT: derivative vector [n_x * 1]

function d = derivative(x, y, theta)
    m = size(x,1);
    d = -sum((y .* sigmoid(x, -y, theta)) .* x)/m;
end

% Function to calculate the hessian matrix of the cost function for
% introducing Newton's method
% INPUT: variable x[m * n_x] in training set, variable y[m * 1] in training set,
% parameters theta[n*x * 1]
% OUTPUT: hessian matrix h[n_x * n_x]
```

```

function [ h ] = hessian( x, y, theta )
    s = sigmoid(x, y, theta);
    m = (size(x, 1));
    h = (x' * (((s .* (1 - s)) .* (y .^ 2)) .* x)) ./ m;
end

% Function to get value of the sigmoid function as part of calculations
% of the cost/derivate/hessian
% INPUT: variable x[m * n_x] in training set, variable y[m * 1] in training set,
% parameters theta[n*x * 1]
% OUTPUT: sigmoid function value s[m * 1]

function s = sigmoid(x,y,theta)
    s = 1 ./ (1 + exp(-y .* (x * theta)));
end

% load the training set and initialize the parameters and cost list vector
train_x = load("assets/logistic_x.txt");
train_y = load("assets/logistic_y.txt");
train_x(:,size(train_x,2) + 1) = ones(size(train_x,1),1);
theta = zeros(size(train_x,2),1);
lostlist = zeros(1,500);

% apply the Newton's method to train the logistic regression model
for i = 1:500

    lostlist(1,i) = cost(train_x, train_y, theta);
    d = derivative(train_x, train_y , theta);
    h = hessian(train_x, train_y, theta);
    theta = theta - h\d';

end

```

最终得到的参数为[0.760;1.172;-2.621]

### 1.3 3

实际上

$$h_{\theta}(x) > 0.5 \rightarrow \frac{1}{1 + e^{-\theta^T x}} > 0.5 \rightarrow \theta^T x > 0$$

因此在图中画出  $\theta^T x = 0$  线即可.

```
% draw the distribution of training data and the final model
ind1 = train_y == 1;
ind2 = train_y == -1;
scatter(train_x(ind1,1), train_x(ind1,2), 'g');
hold on;
scatter(train_x(ind2,1), train_x(ind2,2), 'r');
hold on;
x = 0:8;
y = -(theta(1) * x + theta(3))/theta(2);
plot(x,y);
```

得到图5.

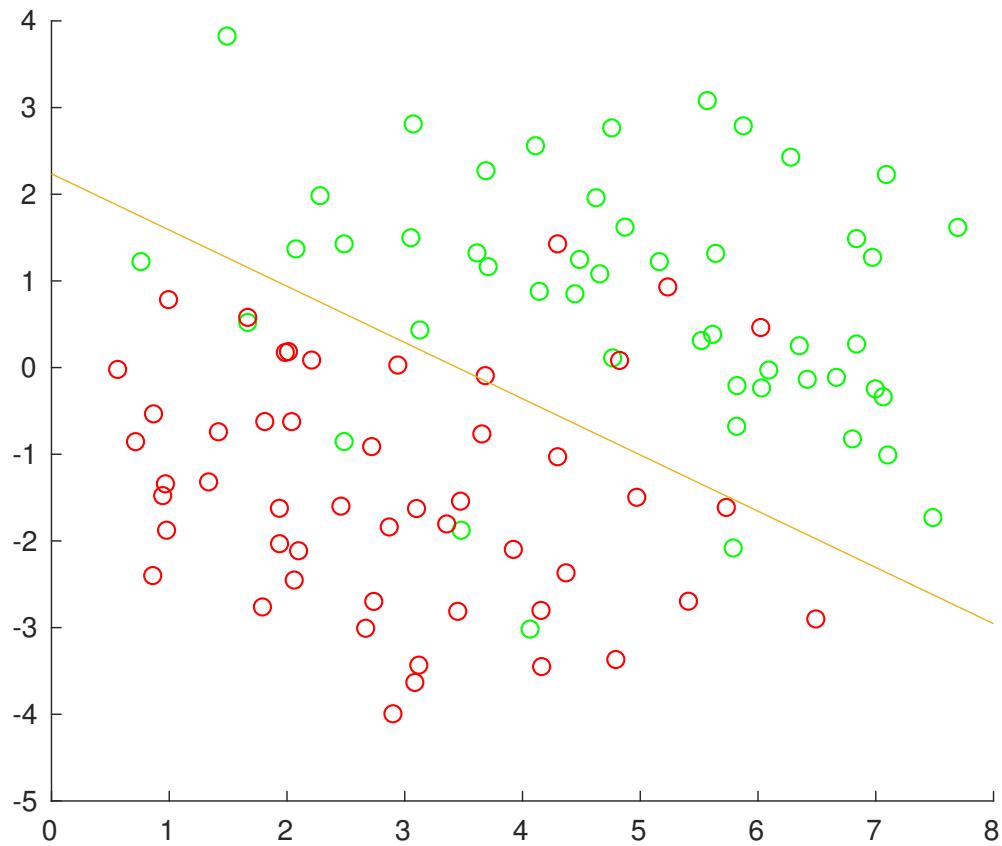


图 1: logistic 回归及其训练集

## 2 Poisson Regression

### 2.1 1

对 Poisson 分布的概率密度稍作变换

$$p(y; \lambda) = \frac{e^{-\lambda} \lambda^y}{y!} = \frac{e^{y \log \lambda - \lambda}}{y!} \quad (11)$$

通过比较系数可得

$$\begin{aligned} b(y) &= \frac{1}{y!} \\ \eta &= \log(\lambda) \\ T(y) &= y \\ a(\eta) &= \lambda = e^\eta \end{aligned}$$

因此 Poisson 分布属于指数分布族.

### 2.2 2

$$h_\theta(x) = E[y|x; \theta] = \lambda = e^\eta = e^{\theta^T x}$$

### 2.3 3

假设只有一个样本点

$$l(\theta) = \log(L(\theta)) = \log\left(\frac{e^{-\lambda} \lambda^y}{y!}\right) = -\lambda + y \log(\lambda) - \log(y!) \quad (12)$$

则有

$$\frac{\partial l(\theta)}{\partial \theta_j} = -e^{\theta^T x} x_j + y x_j = (y - e^{\theta^T x}) x_j \quad (13)$$

因此随机梯度下降的学习规则是

$$\theta_j := \theta_j + \alpha (y^{(i)} - e^{\theta^T x^{(i)}}) x_j^{(i)} \quad (14)$$

### 2.4 4

我们有下式恒成立

$$\int_{-\infty}^{\infty} b(y) e^{\eta y - a(\eta)} dy = e^{-a(\eta)} \int_{-\infty}^{\infty} b(y) e^{\eta y} dy = 1 \quad (15)$$

两边对  $\eta$  求导

$$-e^{-a(\eta)} \frac{\partial a(\eta)}{\partial \eta} \int_{-\infty}^{\infty} b(y) e^{\eta y} dy + e^{-a(\eta)} \int_{-\infty}^{\infty} y b(y) e^{\eta y} dy = 0$$

整理得到

$$-\frac{\partial a(\eta)}{\partial \eta} \int_{-\infty}^{\infty} b(y) e^{\eta y - a(\eta)} dy + \int_{-\infty}^{\infty} y b(y) e^{\eta y - a(\eta)} dy = E(y; \eta) - \frac{\partial a(\eta)}{\partial \eta} = 0$$

因此有

$$E(y; \eta) = \frac{\partial a(\eta)}{\partial \eta} \quad (16)$$

考虑单样本情况, 其最大似然对数

$$l(\theta) = \log(p(y; \eta)) = \log(b(y)) + \eta y - a(\eta)$$

计算其导数

$$\frac{\partial l(\theta)}{\partial \theta_j} = y x_j - \frac{\partial a(\eta)}{\partial \eta} \frac{\partial \eta}{\partial \theta_j} = (y - E(y; \eta)) x_j = (y - h_{\theta}(x)) x_j \quad (17)$$

因此对于任意  $T(y) = y$  的 GLM, 其随机梯度下降的学习规则皆为

$$\theta_j := \theta_j + \alpha(y^{(i)} - e^{\theta^T x^{(i)}}) x_j^{(i)} \quad (18)$$

### 3 Gaussian Discriminant Analysis

#### 3.1 1

$$\begin{aligned} p(y=1|x) &= \frac{p(x|y=1)P(y=1)}{p(x|y=1)P(y=1) + p(x|y=0)P(y=0)} \\ &= \frac{p(x|y=1)\phi}{p(x|y=1)\phi + p(x|y=0)(1-\phi)} \\ &= \frac{\phi}{\phi + \frac{p(x|y=0)}{p(x|y=1)}(1-\phi)} \\ &= \frac{\phi}{\phi + e^{\frac{1}{2}[(x-\mu_1)^T \Sigma^{-1}(x-\mu_1) - (x-\mu_{-1})^T \Sigma^{-1}(x-\mu_{-1})]}(1-\phi)} \\ &= \frac{\phi}{\phi + e^{\frac{1}{2}[u_{-1}^T \Sigma^{-1} - u_1^T \Sigma^{-1} + u_{-1}^T \Sigma^{-1} - u_1 \Sigma^{-1}]x + \frac{1}{2}[u_1^T \Sigma^{-1} u_1 - u_{-1}^T \Sigma^{-1} u_{-1}]}(1-\phi)} \\ &= \frac{1}{1 + e^{[u_{-1}^T \Sigma^{-1} - u_1^T \Sigma^{-1}]x + \frac{1}{2}[u_1^T \Sigma^{-1} u_1 - u_{-1}^T \Sigma^{-1} u_{-1}]}(\frac{1}{\phi} - 1)} \end{aligned} \quad (19)$$

对比系数可得

$$\theta^T = -(u_{-1}^T \Sigma^{-1} - u_1^T \Sigma^{-1}) \quad (20)$$

$$\theta_0 = -\frac{1}{2}(u_1^T \Sigma^{-1} u_1 - u_{-1}^T \Sigma^{-1} u_{-1}) + \log\left(\frac{1}{\phi} - 1\right) \quad (21)$$

### 3.2 2

当  $n = 1$  时, 概率退化为

$$p(x|y = -1) = \frac{1}{(2\pi)^{\frac{1}{2}}\sigma} e^{-\frac{(x-\mu_{-1})^2}{2\sigma^2}} \quad (22)$$

$$p(x|y = -1) = \frac{1}{(2\pi)^{\frac{1}{2}}\sigma} e^{-\frac{(x-\mu_1)^2}{2\sigma^2}} \quad (23)$$

首先讨论  $\phi$

$$\begin{aligned} l(\phi, \mu_{-1}, \mu_1, \Sigma) &= \log\left(\prod_{i=1}^m p(x^{(i)}|y^{(i)}; \mu_{-1}, \mu_1, \Sigma) p(y^{(i)}; \phi)\right) \\ &= \log\left(\prod_{i=1}^m p(x^{(i)}|y^{(i)}; \mu_{-1}, \mu_1, \Sigma)\right) + \log\left(\prod_{i=1}^m (p(y^{(i)}; \phi))\right) \end{aligned} \quad (24)$$

记

$$\begin{aligned} M &= \sum_{i=1}^m 1\{y^{(i)} = 1\} \\ N &= \sum_{i=1}^m 1\{y^{(i)} = -1\} \end{aligned}$$

对  $\phi$  求导

$$\begin{aligned} \frac{\partial l}{\partial \phi} &= \frac{\partial \log(\prod_{i=1}^m (p(y^{(i)}; \phi)))}{\partial \phi} \\ &= \frac{\partial \sum_{i=1}^m \log(p(y^{(i)}; \phi))}{\partial \phi} \\ &= \frac{\partial (M \log(\phi) + N \log(1 - \phi))}{\partial \phi} \\ &= \frac{M}{\phi} - \frac{N}{1 - \phi} = 0 \end{aligned}$$

解出

$$\phi = \frac{M}{M+N} = \frac{1}{m} \sum_{i=1}^m 1\{y^{(i)} = 1\} \quad (25)$$

注意这一结论不依赖于  $n$  的阶数.

讨论  $\mu_1$

$$\begin{aligned} \frac{\partial l}{\partial \mu_1} &= \frac{\partial \log(\prod_{i=1}^m p(x^{(i)}|y^{(i)}; \mu_{-1}, \mu_1, \Sigma))}{\partial \mu_1} \\ &= \frac{\partial \sum_{i=1}^m \log(p(x^{(i)}|y^{(i)}; \mu_{-1}, \mu_1, \Sigma))}{\partial \mu_1} \\ &= \frac{\partial \sum_{i=1}^m \log\left(\frac{1}{(2\pi)^{\frac{1}{2}}\sigma} e^{-\frac{(x^{(i)} - \mu_{y^{(i)}})^2}{2\sigma^2}}\right)}{\partial \mu_1} \\ &= \frac{-\partial \sum_{i=1}^m \frac{(x^{(i)} - \mu_{y^{(i)}})^2}{2\sigma^2}}{\partial \mu_1} \\ &= \sum_{i=1}^m 1\{y^{(i)} = 1\} \frac{x^{(i)} - \mu_1}{\sigma^2} = 0 \end{aligned}$$

整理得

$$\mu_1 = \frac{\sum_{i=1}^m 1\{y^{(i)} = 1\} x^{(i)}}{\sum_{i=1}^m 1\{y^{(i)} = 1\}} \quad (26)$$

由  $\mu_1$  和  $\mu_{-1}$  的对称性, 可以直接写出

$$\mu_{-1} = \frac{\sum_{i=1}^m 1\{y^{(i)} = -1\} x^{(i)}}{\sum_{i=1}^m 1\{y^{(i)} = -1\}} \quad (27)$$

讨论  $\Sigma$

$$\begin{aligned} \frac{\partial l}{\partial \sigma} &= \frac{\partial \log(\prod_{i=1}^m p(x^{(i)}|y^{(i)}; \mu_{-1}, \mu_1, \Sigma))}{\partial \Sigma} \\ &= \frac{\partial \sum_{i=1}^m \log(p(x^{(i)}|y^{(i)}; \mu_{-1}, \mu_1, \Sigma))}{\partial \Sigma} \\ &= \frac{\partial \sum_{i=1}^m \log\left(\frac{1}{(2\pi)^{\frac{1}{2}}\Sigma} e^{-\frac{(x^{(i)} - \mu_{y^{(i)}})^2}{2\sigma^2}}\right)}{\partial \Sigma} \\ &= \frac{\partial(-m \log((2\pi)^{\frac{1}{2}}\sigma) - \sum_{i=1}^m \frac{(x^{(i)} - \mu_{y^{(i)}})^2}{2\sigma^2})}{\partial \Sigma} \\ &= \left(-\frac{m}{\sigma} + \sum_{i=1}^m \frac{(x^{(i)} - \mu_{y^{(i)}})^2}{\sigma^3}\right) \frac{d\sigma}{d\Sigma} = 0 \end{aligned}$$

整理得

$$\Sigma = \sigma^2 = \frac{\sum_{i=1}^m (x^{(i)} - \mu_{y^{(i)}})^2}{m} \quad (28)$$



## 3.3 3

注意上述关于  $\phi$  的讨论并不依赖  $n$  的阶数, 因而

$$\phi = \frac{1}{m} \sum_{i=1}^m 1\{y^{(i)} = 1\} \quad (29)$$

而

$$\begin{aligned} \frac{\partial l}{\partial \mu_1} &= \frac{\partial \log(\prod_{i=1}^m p(x^{(i)}|y^{(i)}; \mu_{-1}, \mu_1, \Sigma))}{\partial \mu_1} \\ &= \frac{\partial \sum_{i=1}^m \log(p(x^{(i)}|y^{(i)}; \mu_{-1}, \mu_1, \Sigma))}{\partial \mu_1} \\ &= \frac{\partial \sum_{i=1}^m \log\left(\frac{1}{(2\pi)^{\frac{n}{2}} |\Sigma|^{1/2}} e^{-\frac{1}{2}(x^{(i)} - \mu_{y^{(i)}})^T |\Sigma|^{-1} (x^{(i)} - \mu_{y^{(i)}})}\right)}{\partial \mu_1} \\ &= \frac{\frac{1}{2} \partial \sum_{i=1}^m (x^{(i)} - \mu_{y^{(i)}})^T |\Sigma|^{-1} (x^{(i)} - \mu_{y^{(i)}})}{\partial \mu_1} \\ &= \frac{1}{2} \sum_{i=1}^m 1\{y^{(i)} = 1\} (|\Sigma|^{-1} + (|\Sigma|^{-1})^T) (x^{(i)} - \mu_{y^{(i)}}) = 0 \end{aligned}$$

最后一个等式使用了式

$$\frac{\partial X^T A X}{\partial X} = (A + A^T) X$$

整理得

$$\mu_1 = \frac{\sum_{i=1}^m 1\{y^{(i)} = 1\} x^{(i)}}{\sum_{i=1}^m 1\{y^{(i)} = 1\}} \quad (30)$$

类似的可以得到

$$\mu_{-1} = \frac{\sum_{i=1}^m 1\{y^{(i)} = -1\} x^{(i)}}{\sum_{i=1}^m 1\{y^{(i)} = -1\}} \quad (31)$$

关于  $\Sigma$

$$\begin{aligned} \frac{\partial l}{\partial \Sigma} &= \frac{\partial \log(\prod_{i=1}^m p(x^{(i)}|y^{(i)}; \mu_{-1}, \mu_1, \Sigma))}{\partial \Sigma} \\ &= \frac{\partial \sum_{i=1}^m \log(p(x^{(i)}|y^{(i)}; \mu_{-1}, \mu_1, \Sigma))}{\partial \Sigma} \\ &= \frac{\partial \sum_{i=1}^m \log\left(\frac{1}{(2\pi)^{\frac{n}{2}} |\Sigma|^{1/2}} e^{-\frac{1}{2}(x^{(i)} - \mu_{y^{(i)}})^T |\Sigma|^{-1} (x^{(i)} - \mu_{y^{(i)}})}\right)}{\partial \Sigma} \\ &= \frac{\partial (-m \log((2\pi)^{\frac{n}{2}} |\Sigma|^{1/2}) - \sum_{i=1}^m \frac{1}{2} (x^{(i)} - \mu_{y^{(i)}})^T |\Sigma|^{-1} (x^{(i)} - \mu_{y^{(i)}}))}{\partial \Sigma} \\ &= \left(-\frac{m}{2|\Sigma|} + \frac{1}{2} \sum_{i=1}^m \frac{(x^{(i)} - \mu_{y^{(i)}})(x^{(i)} - \mu_{y^{(i)}})^T}{|\Sigma|^2}\right) \frac{\partial |\Sigma|}{\partial \Sigma} = 0 \end{aligned}$$

最后一个等式使用了式

$$\frac{\partial X^T A X}{\partial A} = X X^T$$

整理得到

$$\Sigma = \frac{\sum_{i=1}^m (x^{(i)} - \mu_{y^{(i)}})(x^{(i)} - \mu_{y^{(i)}})^T}{m} \quad (32)$$

## 4 Linear invariance of optimization algorithm

### 4.1 1

$x^{(i)}$  的更新规则

$$x^{(i+1)} = x^{(i)} - H_f^{-1} \nabla_x f(x)$$

$z^{(i)}$  的更新规则

$$z^{(i+1)} = z^{(i)} - H_g^{-1} \nabla_z g(z)$$

$$\frac{\partial g(z)}{\partial z} = \frac{\partial f(Az)}{\partial z} = \frac{\partial Az}{\partial z} \frac{\partial f(Az)}{\partial Az} = A^T \nabla_x f(x) \quad (33)$$

$$H_g = \frac{\partial}{\partial z} \left( \frac{\partial g(z)}{\partial z} \right)^T = A^T \frac{\partial}{\partial Az} \left( A^T \frac{\partial f(Az)}{\partial Az} \right)^T = A^T H_f A \quad (34)$$

由此得到

$$z^{(i+1)} = z^{(i)} - A^{-1} H_f^{-1} (A^T)^{-1} A^T \nabla_x f(x) = A^{-1} (x^{(i)} - H_f^{-1} \nabla_x f(x)) = A^{-1} x^{(i+1)} \quad (35)$$

通过初始条件  $z^0 = A^{-1} x^{(0)}$ , 利用数学归纳法可以导出结论.

### 4.2 2

$x^{(i)}$  的更新规则

$$x^{(i+1)} = x^{(i)} - \nabla_x f(x)$$

$z^{(i)}$  的更新规则

$$z^{(i+1)} = z^{(i)} - \nabla_z g(z) = z^{(i)} - A^T \nabla_x f(x) = A^{-1} (x^{(i)} - A A^T \nabla_x f(x)) \neq A^{-1} x^{(i+1)}$$

因此梯度下降法不是重参数化不变的.

## 5 Regression for Denoising Quasar Spectra

### 5.1

#### 5.1.1

$$J(\theta) = \frac{1}{2} \sum_{i=1}^m \omega^{(i)} (\theta^T x^{(i)} - y^{(i)})^2 = (X\theta - y)^T W (X\theta - y)$$

$$W = \begin{bmatrix} w^1 & 0 & \cdots & 0 \\ 0 & w^2 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & w^m \end{bmatrix} \quad (36)$$

#### 5.1.2

$$\frac{\partial J(\theta)}{\partial \theta} = \frac{\partial (X\theta - y)}{\partial \theta} \frac{\partial J(\theta)}{\partial (X\theta - y)} = 2X^T W (X\theta - y) = 0 \quad (37)$$

由此得到

$$X^T W X \theta = X^T W y \quad (38)$$

#### 5.1.3

最大化其最大似然的对数

$$\begin{aligned} l(\theta) &= \log L(\theta) = \log \left( \prod_{i=1}^m \frac{1}{\sqrt{2\pi}\sigma^{(i)}} e^{-\frac{(y^{(i)} - \theta^T x^{(i)})^2}{2(\sigma^{(i)})^2}} \right) \\ &= - \sum_{i=1}^m \log(\sqrt{2\pi}\sigma^{(i)}) - \sum_{i=1}^m \frac{(y^{(i)} - \theta^T x^{(i)})^2}{2(\sigma^{(i)})^2} \end{aligned} \quad (39)$$

最大化  $l(\theta)$ , 即最小化

$$\sum_{i=1}^m \frac{(y^{(i)} - \theta^T x^{(i)})^2}{2(\sigma^{(i)})^2}$$

即解加权系数为

$$w^{(i)} = \frac{1}{(\sigma^{(i)})^2}$$

的加权线性回归.

## 5.2 2

### 5.2.1 1

```
d = quasar_train(2,:);  
% construct x  
x = [lambdas, ones(size(lambdas,1),1)];  
y = d';  
% use normal equation to get theta instead of the gradient descent method  
theta = (x' * x) \ x' * y;  
scatter(lambdas, d', 20, 'g');  
hold on;  
plot(lambdas, theta(1) * lambdas + theta(2));
```

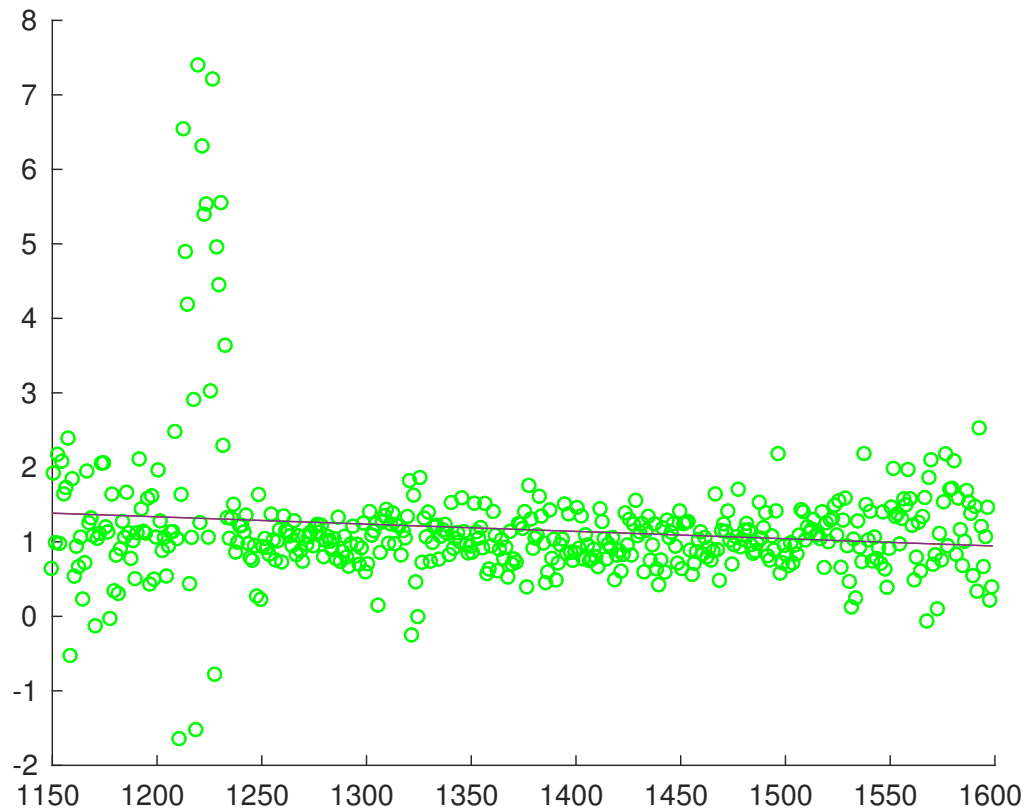


图 2: linear regression

## 5.2.2 2

```

d = quasar_train(2,:);
% construct x
x = [lambdas, ones(size(lambdas,1),1)];
y = d';
% use normal equation to get theta instead of the gradient descent method
tau = [1,5,10,100,1000];
y_predict = zeros(size(lambdas,1),5);
for k = 1:size(tau,2)
    for i = 1:size(lambdas, 1)
        W = diag(exp(-((lambdas(i,1) - x(:,1)) .^ 2) ./ (2 * (tau(1,k) ^ 2))));
        theta = (x' * W * x) \ (x' * W * y);
        y_predict(i,k) = theta(1) * lambdas(i,1) + theta(2);
    end
    plot(lambdas, y_predict(:,k));
    hold on;
end
scatter(lambdas, d', 20, 'b');
legend('tau=1', 'tau=5', 'tau=10', 'tau=100', 'tau=1000', 'train_data');
hold on;

```

$\tau$  值越大, 加权线性回归对于局部值越不敏感, 越接近于正常的回归, 当  $\tau$  值比较小时, 回归曲线的波动比较大, 受单值影响很大.

## 6 3

按照说明写出 MATLAB 程序

```

% function to smooth training and test data with locally weighted
% regressions
% INPUT:
% data[m * n], m is the number of the training data, n is the length
% of a training sample
% lambdas[n * 1], lambdas is the corresponding wavelength of the data of
% intensity
%
% OUTPUT:
% smooth_data[m * n], data smoothed with locally weighted regressions

```

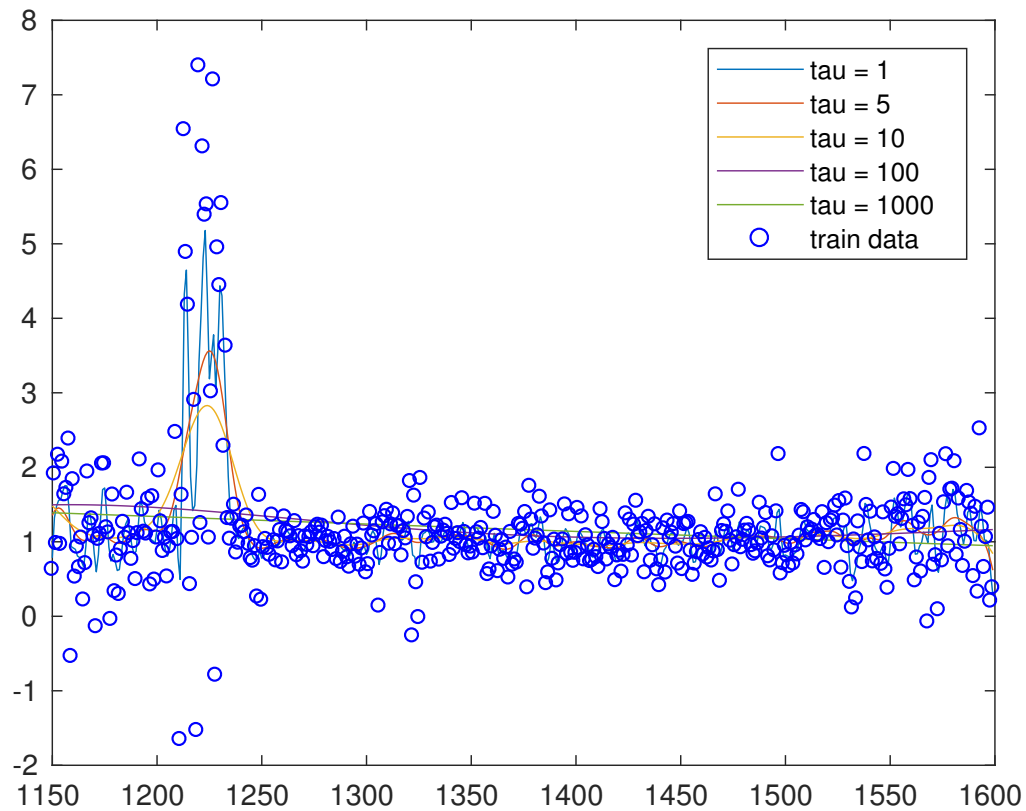


图 3: 加权线性回归

```
function smooth_data = smooth(data, lambdas)
    % use normal equation to get theta instead of the gradient descent method
    x = [lambdas, ones(size(lambdas,1),1)];
    tau = 5;
    smooth_data = zeros(size(data,1),size(data,2));

    for k = 1:size(data, 1)
        y = data(k,:)' ;
        for i = 1:size(data,2)
            W = diag(exp(-((lambdas(i,1) - x(:,1))' ) .^ 2) ./ (2 * (tau ^ 2))));
            theta = (x' * W * x) \ (x' * W * y);
            smooth_data(k,i) = theta(1) * lambdas(i,1) + theta(2);
        end
    end
end
```

```

end

% Function to calculate the distance defined in the problem of the right
% part of two spectra
% INPUT:
% spectra f1,f2 [1 * n], n is the length of wavelength vector
% OUTPUT:
% distance d, scalar

function d = distance(f1, f2)
    d = sum((f1(151:end) - f2(151:end)) .^ 2);
end

% Function to calculate the distance defined in the problem of the left
% part of two spectra
% INPUT:
% spectra f1,f2 [1 * n], n is the length of wavelength vector or the length
% of left part prediction vector
% OUTPUT:
% distance d, scalar

function d = distance_left(f1, f2)
    d = sum((f1(1:50) - f2(1:50)) .^ 2);
end

% Function to calculate the denotation ker
% INPUT:
% t, scalar
% OUTPUT:
% k, max{1-t, 0}
function k = ker(t)
    k = max(1-t,0);
end

% smooth the data
smooth_train_qso = smooth(train_qso, lambdas);
smooth_test_qso = smooth(test_qso, lambdas);
% set neighbor = 3

```

```

k = 3;

f_left_errors = 0;

% use the training data set to test the functional regression
for j = 1:size(smooth_train_qso,1)
    f_right = smooth_train_qso(j,:);

    % calculate the distance between f_right with each f^(i)_right in the
    % training set
    dis_list = zeros(size(smooth_train_qso,1),1);
    for i = 1:size(smooth_train_qso,1)
        dis_list(i,1) = distance(smooth_train_qso(i,:), f_right);
    end

    [V I] = sort(dis_list);

    den = zeros(1,50);
    num = 0;
    for i = 1:k
        den = den + ker(V(i)/V(end)) * smooth_train_qso(I(i),1:50);
        num = num + ker(V(i)/V(end));
    end

    f_left = den/num;
    f_left_errors = f_left_errors + distance_left(f_left, smooth_train_qso(j,:));
end

ave_f_left_error = f_left_errors / size(smooth_train_qso,1);

f_left_errors_t = 0;

% use the training test set to test the functional regression
for j = 1:size(smooth_test_qso,1)
    f_right = smooth_test_qso(j,:);

    dis_list = zeros(size(smooth_train_qso,1),1);
    for i = 1:size(smooth_train_qso,1)

```



```

        dis_list(i,1) = distance(smooth_train_qso(i,:), f_right);
    end

    [V I] = sort(dis_list);

    den = zeros(1,50);
    num = 0;
    for i = 1:k
        den = den + ker(V(i)/V(end)) * smooth_train_qso(I(i),1:50);
        num = num + ker(V(i)/V(end));
    end

    f_left = den/num;
    f_left_errors_t = f_left_errors_t + distance_left(f_left, smooth_test_qso(j,:))
        ;

    if(j == 1)
        plot(lambdas, smooth_test_qso(j,:)');
        hold on;
        plot(lambdas(1:50), f_left');
    end
end

ave_f_left_error_t = f_left_errors_t / size(smooth_test_qso,1);

```

在训练集上取得的平均误差为 1.0664, 在测试集上取得的平均误差为 2.7100.

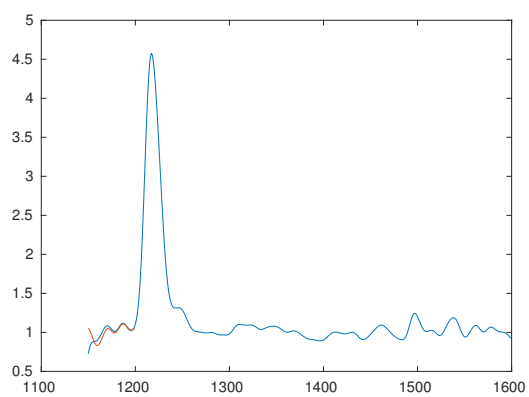


图 4: 测试集合 1-拟合

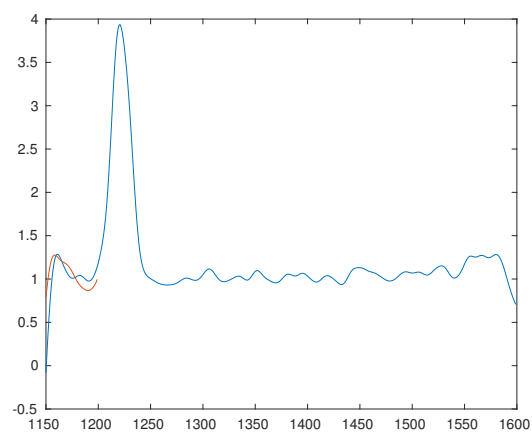


图 5: 测试集合 6-拟合