機械学習特論

~ 理論とアルゴリズム ~

第９回

(Regularization for Logistic Regression)

講師：西郷浩人

Previous exercise 1: Implement Logistic Regression by Newton's method

- Download todays' data
- Executing ex1.m loads and plots data
- Complete logistic.m
- Logistic function h(x) is already declared.

logistic.m

```
function [b,J]=logistic(X,y)
[n p]=size(X); b = zeros(p);
J=0; itr=0;

while 1
  itr = itr + 1;
  disp(["iteration: ",num2str(itr)]);
  prev_J = J;

  % Compute J here
  % J = -(y'*log(h(X*b))+(ones(n,1)-y)'*log(ones(n,1)-h(X*b)))
  J = -(y'*X*b-sum(log(1+exp(X*b))))
  if abs(prev_J-J) < 1/n, break, end

  % Compute the first gradient here
  grad = X'*(h(X*b)-y);

  % Compute the second gradient here
  W=diag(h(X*b).*(ones(n,1)-h(X*b)));
  H=X'*W*X;

  % Update beta here
  b = b – inv(H)*grad; % update weights
end
endfunction

function [y] = h(x)
y = 1.0 ./ (1.0 + exp(-x));
endfunction
```

$$
\begin{aligned}
&J(\boldsymbol{\beta})\\
&= -\left(\boldsymbol{y}'\log h(\boldsymbol{X}\boldsymbol{\beta}) + (\mathbf{1}-\boldsymbol{y})'\log(\mathbf{1}-h(\boldsymbol{X}\boldsymbol{\beta}))\right)\\
&= -\left(\boldsymbol{y}'\boldsymbol{X}\boldsymbol{\beta} - \sum\left(\log(1+\exp(\boldsymbol{X}\boldsymbol{\beta}))\right)\right)
\end{aligned}
$$

$$\nabla_{\boldsymbol{\beta}} J = \boldsymbol{X}'(h(\boldsymbol{X}\boldsymbol{\beta}) - \boldsymbol{y})$$

$$\boldsymbol{H} = \boldsymbol{X}'\boldsymbol{W}\boldsymbol{X}$$

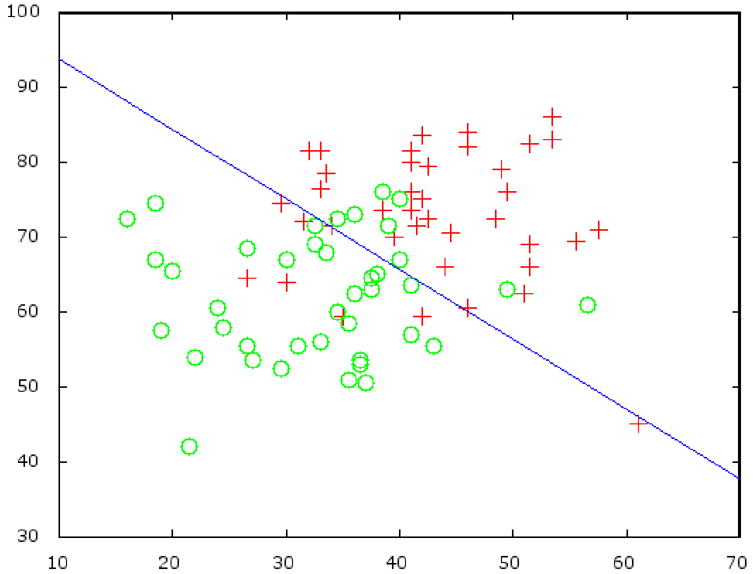
$$\boldsymbol{\beta} \leftarrow \boldsymbol{\beta} - \boldsymbol{H}^{-1}\nabla_{\boldsymbol{\beta}} J$$

Decision boundary on test data

- Coefficients and decision boundary

$$
\begin{aligned}
& h(\boldsymbol{\beta}'\boldsymbol{x}) = 1 - h(\boldsymbol{\beta}'\boldsymbol{x}) \\
\leftarrow\rightarrow\ & \frac{1}{1+\exp(\boldsymbol{\beta}'\boldsymbol{x})} = \frac{\exp(\boldsymbol{\beta}'\boldsymbol{x})}{1+\exp(\boldsymbol{\beta}'\boldsymbol{x})} \\
\leftarrow\rightarrow\ & \exp(\boldsymbol{\beta}'\boldsymbol{x}) = 1 \\
\leftarrow\rightarrow\ & \beta_0 + \beta_1 x_1 + \beta_2 x_2 = 0
\end{aligned}
$$

```
beta =
  -16.37874
    0.14834
    0.15891
J =  32.436
```



- Equation for the decision boundary

$$
\begin{aligned}
& -16.4 + 0.15\, x_1 + 0.16\, x_2 = 0 \\
\leftarrow\rightarrow\ & x_2 = \frac{16.4}{0.16} - \frac{0.15}{0.16} x_1
\end{aligned}
$$

Solving Logistic Regression by Iteratively Reweighted Least Squares (IRLS)

Algorithm: Logistic Regression

- Input:
  - X: n x p data matrix
  - y: n x 1 binary response vector
- Output
  - β: p x 1 coefficient vector
- Initialize
  - J=0; prev_J = LARGE_NUMBER;
- Repeat
  - Compute J $J(\boldsymbol{\beta})$
  - If prev_J – J < SMALL_NUMBER
    - Break
  - Compute nabla_J $\nabla_{\beta} J = \boldsymbol{X}'(h(\boldsymbol{X}\boldsymbol{\beta}) - \boldsymbol{y})$
  - Compute H $\boldsymbol{W} = diag(h(\boldsymbol{X}\boldsymbol{\beta})(\mathbf{1} - h(\boldsymbol{X}\boldsymbol{\beta})))$ $\boldsymbol{H} = \boldsymbol{X}'\boldsymbol{W}\boldsymbol{X}$
  - Update β $\boldsymbol{\beta} \leftarrow \boldsymbol{\beta} - \boxed{\boldsymbol{H}^{-1}} \nabla_{\beta} J$

Drawbacks

- It requires to calculate the inverse of a Hessian matrix.
  - This step can be numerically unstable, and/or slow.
  - Therefore IRLS (Iteratively Re-weighted Least Squares) is often used instead for solving logistic regression.

Rewriting the update rule

$$\nabla_\beta J = X'(h(X\beta) - y) \qquad H = X'WX$$

- Rewrite the update rule

$$\begin{aligned}\beta &\leftarrow \beta - H^{-1}\nabla_\beta J \\ &= \beta + (X'WX)^{-1}X'(y - h(X\beta)) \\ &= (X'WX)^{-1}X'W(X\beta + W^{-1}(y - h(X\beta))) \\ &= (X'WX)^{-1}X'Wz\end{aligned}$$

  - where $z = X\beta + W^{-1}(y - h(X\beta))$
- Remember that the OLS solution for $X\beta = y$ is $\beta = (X'X)^{-1}X'y$
- New update rule iteratively solves weighted least squares problem $(X'WX)\beta = X'Wz$
  - Whose solution is $\beta = (X'WX)^{-1}X'Wz$
  - However, **z** and **W** are functions of **β**, so update is repeated until convergence.

Algorithm: Logistic Regression by IRLS

- Input:
  - X: n x p data matrix
  - y: n x 1 binary response vector
- Output
  - β: p x 1 coefficient vector
- Initialize
  - J=0; prev_J = LARGE_NUMBER;
- Repeat
  - Compute J $J(\boldsymbol{\beta})$
  - If prev_J – J < SMALL_NUMBER
    - Break
  - Compute w $\boldsymbol{W} = diag\left(h(\boldsymbol{X}\boldsymbol{\beta})(\mathbf{1} - h(\boldsymbol{X}\boldsymbol{\beta}))\right)$
  - Compute z $\boldsymbol{z} = \left(\boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{W}^{-1}(\boldsymbol{y} - h(\boldsymbol{X}\boldsymbol{\beta}))\right)$
  - Solve $(\boldsymbol{X}'\boldsymbol{W}\boldsymbol{X})\boldsymbol{\beta} = \boldsymbol{X}'\boldsymbol{W}\boldsymbol{z}$ for β

Ex. 1: implement logistic regression by IRLS

- Implement logistic regression by IRLS and name it as logisticIRLS.m
- You can load data by typing "init"
- Remember to use "back slash operator" for solving normal equation.
- You can check time spent for calculation using "tic" and "toc".

```
>tic, [beta J]=logistic(X,y), toc
```

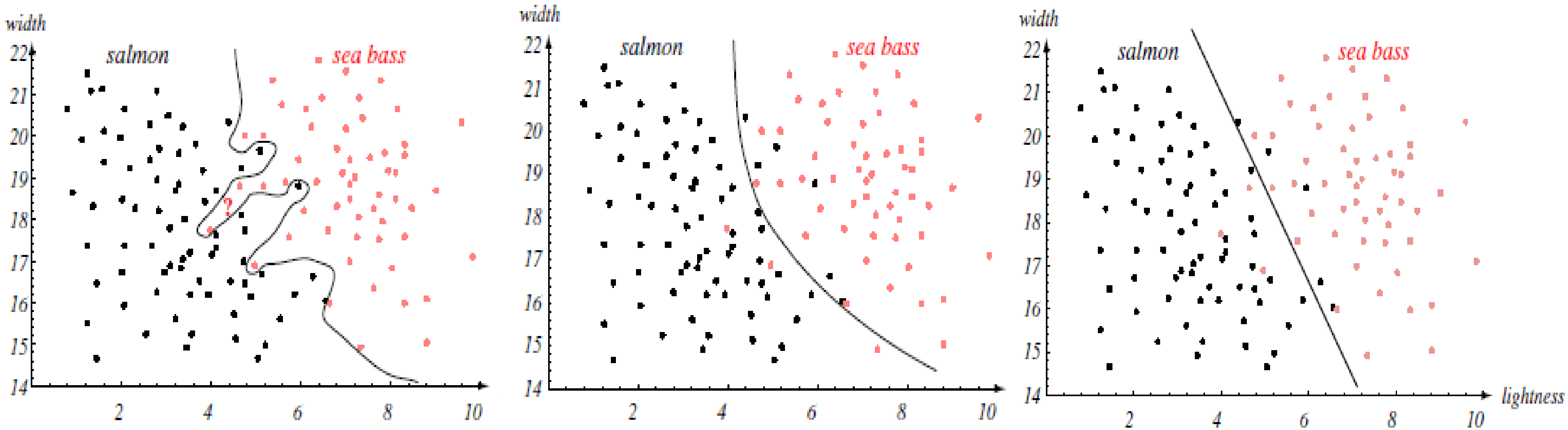
- Compare the time and the solution of logistic.m with that of logisticIRLS.m.

logisticIRLS.m

```
function [beta, J] = logisticIRLS(X,y)
[n p] = size(X);
beta = zeros(p,1);
itr = 0;
J = 0;
while 1
  itr = itr + 1;
  prev_J = J;

% compute J here
 %

  if abs(prev_J-J) <1/n break, end

  % update beta here
  %

end
endfunction

function [y] = h(x)
y = 1.0 ./ (1.0 + exp(-x));
endfunction
```

Regularization in classification

- An optimal classification rule is deemed to exist between overfitting and underfitting.
- How can we adjust between the two ?
  - > via regularization parameter



Overfitting Optimal Underfitting

© Duda, Hart, Stork, Pattern Classification, 2001 Wiley

Regularization for regression

$$RSS = \|\boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{y}\|_2^2$$

- OLS

$$min_{\boldsymbol{\beta}}\, RSS$$

- Ridge regression (L2-regularization)

$$min_{\boldsymbol{\beta}}\, RSS + \lambda \|\boldsymbol{\beta}\|_2$$

- LASSO regression (L1-regularization)

$$min_{\boldsymbol{\beta}}\, RSS + \lambda \|\boldsymbol{\beta}\|_1$$

Regularization for classification

$$J = -l(\boldsymbol{\beta}) = -\sum_{i=1}^{n} \{ y_i h(\boldsymbol{x_i}) + (1 - y_i) h(\mathbf{1} - \boldsymbol{x_i}) \}$$

- Logistic regression

$$min_{\boldsymbol{\beta}} J$$

- L2-regularized Logistic regression

$$min_{\boldsymbol{\beta}} J + \lambda \|\boldsymbol{\beta}\|_2$$

- L1-regularized Logistic regression

$$min_{\boldsymbol{\beta}} J + \lambda \|\boldsymbol{\beta}\|_1$$

Solving L2-Logistic Regression by Iteratively Reweighted Ridge Regression (IRRR)

$$min_{\boldsymbol{\beta}}\, J + \lambda \left\| \boldsymbol{\beta} \right\|_2$$

Solving L2-Logistic Regression

- Objective function

$$J = -l(\boldsymbol{\beta}) = -\sum_{i=1}^{n} \{ y_i \boldsymbol{\beta}' \boldsymbol{x_i} - \log(1 + \exp(\boldsymbol{\beta}' \boldsymbol{x_i})) \} + \frac{\lambda}{2} \sum_{j=1}^{p} \beta_j^2$$

Deriving the first derivative

$$J = -l(\boldsymbol{\beta}) = -\sum_{i=1}^{n} \{ y_i \boldsymbol{\beta}' \boldsymbol{x_i} - \log(1 + \exp(\boldsymbol{\beta}' \boldsymbol{x_i})) \} + \frac{\lambda}{2} \sum_{j=1}^{p} \beta_j^2$$

- Same way as the previous lecture.

$$\nabla J = \frac{-\partial l}{\partial \beta_j} = \frac{-\partial}{\partial \beta_j} \left(\sum_{i=1}^{n} \{ y_i \boldsymbol{\beta}' \boldsymbol{x_i} - \log(1 + \exp(\boldsymbol{\beta}' \boldsymbol{x_i})) \} + \frac{\lambda}{2} \sum_{j=1}^{p} \beta_j^2 \right)$$
$$= -\sum_{i=1}^{n} \boldsymbol{x_i} (y_i - h(\boldsymbol{\beta}' \boldsymbol{x_i})) + \lambda \beta_j$$

- Vector form

$$\nabla_{\boldsymbol{\beta}} J = \boldsymbol{X}' (h(\boldsymbol{X} \boldsymbol{\beta}) - \boldsymbol{y}) + \lambda \boldsymbol{\beta}$$

Deriving the second derivative

$$\nabla J = \frac{-\partial l}{\partial \boldsymbol{\beta}} = -\boldsymbol{X}'\left(\boldsymbol{y} - h\left(\boldsymbol{X}\boldsymbol{\beta}\right)\right) + \lambda \boldsymbol{\beta}$$

- Same way as the previous lecture.

$$\frac{\partial J}{\partial \beta^2} = \frac{-\partial l^2}{\partial \beta \partial \beta} == \frac{-\partial}{\partial \beta}\left\{\sum_{i=1}^{n} -\boldsymbol{x_i}\left(1 - \frac{\exp\left(\boldsymbol{\beta}'\boldsymbol{x_i}\right)}{1+\exp\left(\boldsymbol{\beta}'\boldsymbol{x_i}\right)}\right) + \lambda \beta_j\right\}$$

$$= -\sum_{i=1}^{n}\left(\frac{-\boldsymbol{x_i}'\boldsymbol{x_i}\exp\left(\boldsymbol{\beta}'\boldsymbol{x_i}\right)}{\left(1+\exp\left(\boldsymbol{\beta}'\boldsymbol{x_i}\right)\right)^2}\right) + \lambda$$

$$\boldsymbol{H} = \sum_{i=1}^{n} h\left(\boldsymbol{x_i}'\boldsymbol{\beta}\right)\left(1 - h\left(\boldsymbol{x_i}'\boldsymbol{\beta}\right)\right)\boldsymbol{x_i}\boldsymbol{x_i}' + \lambda \begin{bmatrix} 1 & 0 & 0 & . \\ 0 & 1 & 0 & . \\ 0 & 0 & 1 & . \\ . & . & . & . \end{bmatrix} = \boldsymbol{X}'\boldsymbol{W}\boldsymbol{X} + \lambda \boldsymbol{I}$$

$$\boldsymbol{W} = diag\left(h\left(\boldsymbol{X}\boldsymbol{\beta}\right)\left(\boldsymbol{1} - \boldsymbol{h}\left(\boldsymbol{X}\boldsymbol{\beta}\right)\right)\right) = \begin{bmatrix} h\left(\boldsymbol{x_1}'\boldsymbol{\beta}\right)\left(1-h\left(\boldsymbol{x_1}'\boldsymbol{\beta}\right)\right) & 0 & 0 & . \\ 0 & h\left(\boldsymbol{x_2}'\boldsymbol{\beta}\right)\left(1-h\left(\boldsymbol{x_2}'\boldsymbol{\beta}\right)\right) & 0 & . \\ 0 & 0 & . & . \\ . & . & . & . \end{bmatrix}$$

Rewriting the update rule

$$\nabla_\beta J = \boldsymbol{X}'(h(\boldsymbol{X}\boldsymbol{\beta}) - \boldsymbol{y}) + \lambda\boldsymbol{\beta}$$

$$\boldsymbol{H} = \boldsymbol{X}'\boldsymbol{W}\boldsymbol{X} + \lambda\boldsymbol{I}$$

- Rewrite the update rule

$$\begin{aligned}
\boldsymbol{\beta} &\leftarrow \boldsymbol{\beta} - \boldsymbol{H}^{-1}\nabla_{\boldsymbol{\beta}} J \\
&= \boldsymbol{\beta} - (\boldsymbol{X}'\boldsymbol{W}\boldsymbol{X} + \lambda\boldsymbol{I})^{-1}(\boldsymbol{X}'(h(\boldsymbol{X}\boldsymbol{\beta}) - \boldsymbol{y}) + \lambda\boldsymbol{\beta}) \\
&= (\boldsymbol{I} - (\boldsymbol{X}'\boldsymbol{W}\boldsymbol{X} + \lambda\boldsymbol{I})^{-1}\lambda)\boldsymbol{\beta} + (\boldsymbol{X}'\boldsymbol{W}\boldsymbol{X} + \lambda\boldsymbol{I})^{-1}\boldsymbol{X}'(h(\boldsymbol{X}\boldsymbol{\beta}) - \boldsymbol{y}) \\
&= (\boldsymbol{X}'\boldsymbol{W}\boldsymbol{X} + \lambda\boldsymbol{I})^{-1}\boldsymbol{X}'\boldsymbol{W}\boldsymbol{X}\beta + (\boldsymbol{X}'\boldsymbol{W}\boldsymbol{X} + \lambda\boldsymbol{I})^{-1}\boldsymbol{X}'(h(\boldsymbol{X}\boldsymbol{\beta}) - \boldsymbol{y}) \\
&= (\boldsymbol{X}'\boldsymbol{W}\boldsymbol{X} + \lambda\boldsymbol{I})^{-1}\boldsymbol{X}'\boldsymbol{W}(\boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{W}^{-1}(\boldsymbol{y} - h(\boldsymbol{X}\boldsymbol{\beta}))) \\
&= (\boldsymbol{X}'\boldsymbol{W}\boldsymbol{X} + \lambda\boldsymbol{I})^{-1}\boldsymbol{X}'\boldsymbol{W}\boldsymbol{z}
\end{aligned}$$

  - where $\boldsymbol{z} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{W}^{-1}(\boldsymbol{y} - h(\boldsymbol{X}\boldsymbol{\beta}))$

- New update rule iteratively solves weighted ridge regression problem $(\boldsymbol{X}'\boldsymbol{W}\boldsymbol{X} + \lambda\boldsymbol{I})\boldsymbol{\beta} = \boldsymbol{X}'\boldsymbol{W}\boldsymbol{z}$

Algorithm: L2-Logistic Regression by IRLS

- Input:
  - X: n x p data matrix
  - y: n x 1 binary response vector
  - λ
- Output
  - β: p x 1 coefficient vector
- Initialize
  - J=0; prev_J = LARGE_NUMBER;
- Repeat
  - Compute J $\quad J(\beta) = -\sum_{i=1}^{n} \{y_i \boldsymbol{\beta}' \boldsymbol{x}_i - \log(1+\exp(\boldsymbol{\beta}' \boldsymbol{x}_i))\} + \frac{\lambda}{2} \|\boldsymbol{\beta}\|_2$
  - If prev_J – J < SMALL_NUMBER
    - Break
  - Compute w $\quad \boldsymbol{W} = diag(h(\boldsymbol{X}\boldsymbol{\beta})(\mathbf{1} - h(\boldsymbol{X}\boldsymbol{\beta})))$
  - Compute z $\quad \boldsymbol{z} = (\boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{W}^{-1}(\boldsymbol{y} - h(\boldsymbol{X}\boldsymbol{\beta})))$
  - Solve $(\boldsymbol{X}'\boldsymbol{W}\boldsymbol{X} + \lambda \boldsymbol{I})\boldsymbol{\beta} = \boldsymbol{X}'\boldsymbol{W}\boldsymbol{z}$ for β

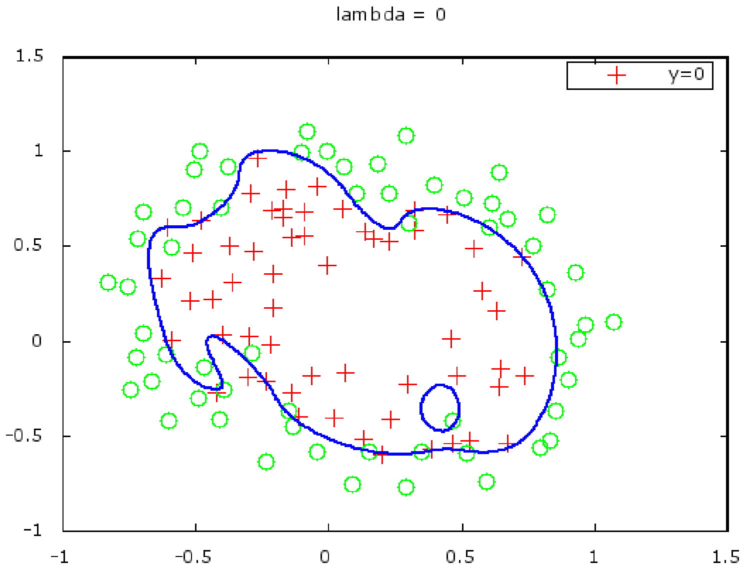
Exercise. 2: Implement L2-Logistic Regression by IRRR

- Implement Algorithm 2, and name it as "logisticIRLSL2.m".
- After completion, you can plot the decision boundary by using "plotLR.m" function.
  - Usage example when λ=0

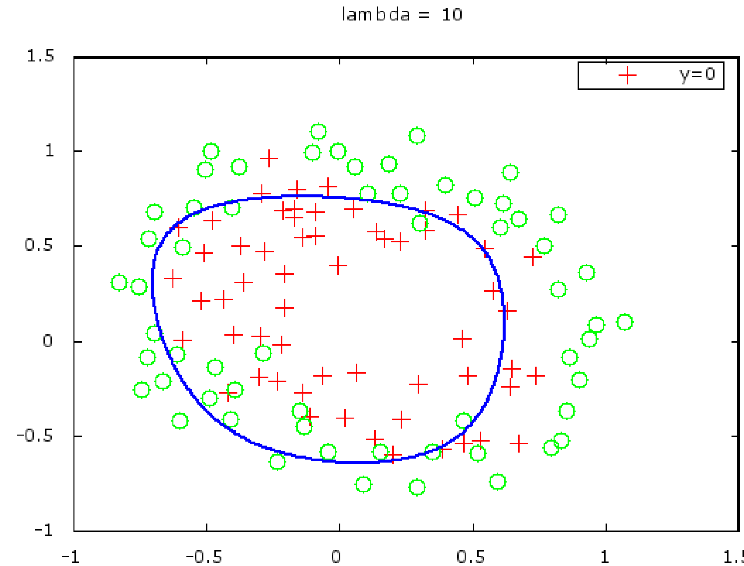
```
>plotLR(X,y,0)
```

  - It uses fifth order polynomial function via map_feature.m
- Observe how decision boundary changes as you increase λ.

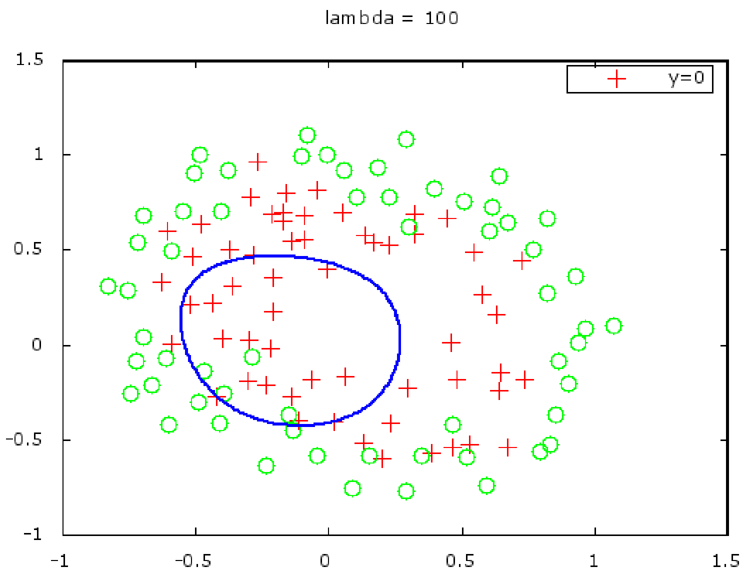
Demonstration: L2



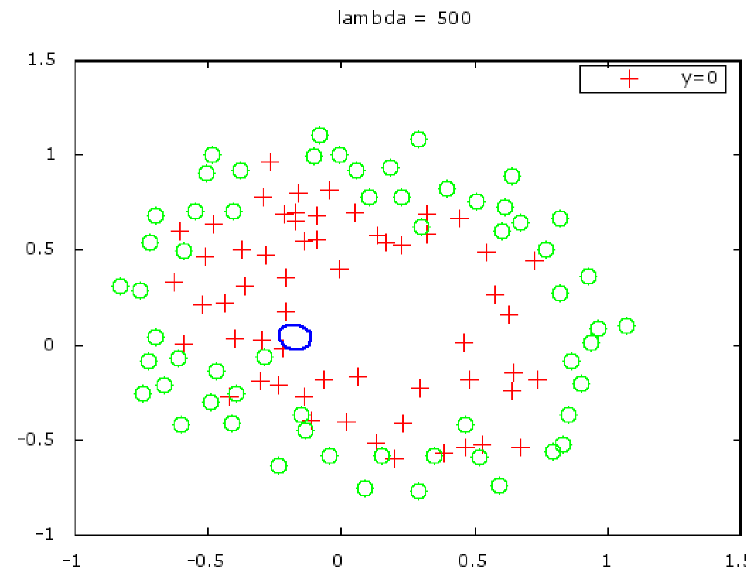
Sparsity: 0%, Likelihood: -23.4



Sparsity: 0%, Likelihood: -76.8



Sparsity: 4%, Likelihood: -85.5



Sparsity: 28.6%, Likelihood: -87.1

Solving L1-Logistic Regression by Iteratively Reweighted Ridge Regression (IRRR)

$$min_{\boldsymbol{\beta}} J + \lambda \|\boldsymbol{\beta}\|_1$$

Solving L1-logistic regression

$$min_{\boldsymbol{\beta}} J + \lambda \|\boldsymbol{\beta}\|_1 \qquad \|\boldsymbol{x}\|_1 = \sum_{i=1}^{n} |x_i|$$

Remember that we have solved L1-regularized least squares problem (LASSO) by using the following update rule.

$$\boldsymbol{\beta}^{lasso} \leftarrow \left(\boldsymbol{X}' \boldsymbol{X} + \lambda \mathbf{B} \right)^{-1} \boldsymbol{X}' \boldsymbol{y}$$

where

$$\mathbf{B} = diag\left(1/|\beta_1|, 1/|\beta_2|, \dots, 1/|\beta_p| \right)$$

In logisitic regression, L1-norm solution is easily obtained by solving the following normal equation in IRLS.

$$\boldsymbol{\beta}^{L1LR} \leftarrow \left(\boldsymbol{X}' \boldsymbol{W} \boldsymbol{X} + \lambda \mathbf{B} \right)^{-1} \boldsymbol{X}' \boldsymbol{W} \boldsymbol{z}$$

Algorithm: L1-Logistic Regression by IRLS

- Input:
  - X: n x p data matrix
  - y: n x 1 binary response vector
  - λ
- Output
  - β: p x 1 coefficient vector
- Initialize
  - J=0; prev_J = LARGE_NUMBER;
- Repeat
  - Compute J $\quad J(\beta) = -\sum_{i=1}^{n} \{y_i \boldsymbol{\beta}' \boldsymbol{x}_i - \log(1+\exp(\boldsymbol{\beta}' \boldsymbol{x}_i))\} + \lambda \|\boldsymbol{\beta}\|_1$
  - If prev_J – J < SMALL_NUMBER
    - Break
  - Compute w $\quad \boldsymbol{W} = diag(h(\boldsymbol{X}\boldsymbol{\beta})(\mathbf{1} - h(\boldsymbol{X}\boldsymbol{\beta})))$
  - Compute z $\quad \boldsymbol{z} = (\boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{W}^{-1}(\boldsymbol{y} - h(\boldsymbol{X}\boldsymbol{\beta})))$
  - Solve $(\boldsymbol{X}'\boldsymbol{W}\boldsymbol{X} + \lambda \mathbf{B})\boldsymbol{\beta} = \boldsymbol{X}'\boldsymbol{W}\boldsymbol{z}$ for β

Hint

- You can build $\mathbf{B} = diag\left(1/|\beta_1|, 1/|\beta_2|, \ldots, 1/|\beta_p|\right)$ by

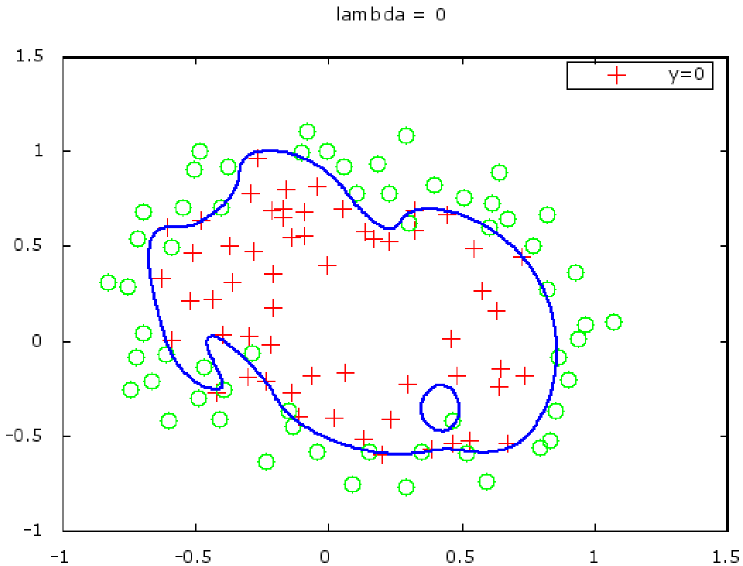
```
pinv(diag(abs(beta)))
```

  - where pinv means "psuedo inverse", which is useful even when inverse of a matrix cannot be defined because of singularity.

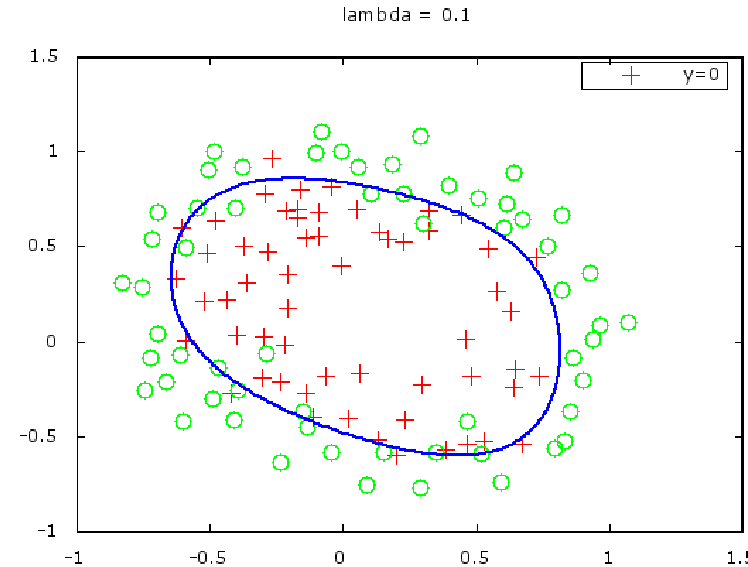
Ex. 3: Implement L1-Logistic Regression by IRLS

- Modify two lines in “logisticIRLSL2.m”.
  - Objective function
  - Least squares fit
- Compare the obtained likelihood and sparsity with those obtained by L2-Logistic Regression.
- You can plot data with
  - plotLR(X,y,0.3,1), where the last option specifies norm, and takes either '1' or '2'.

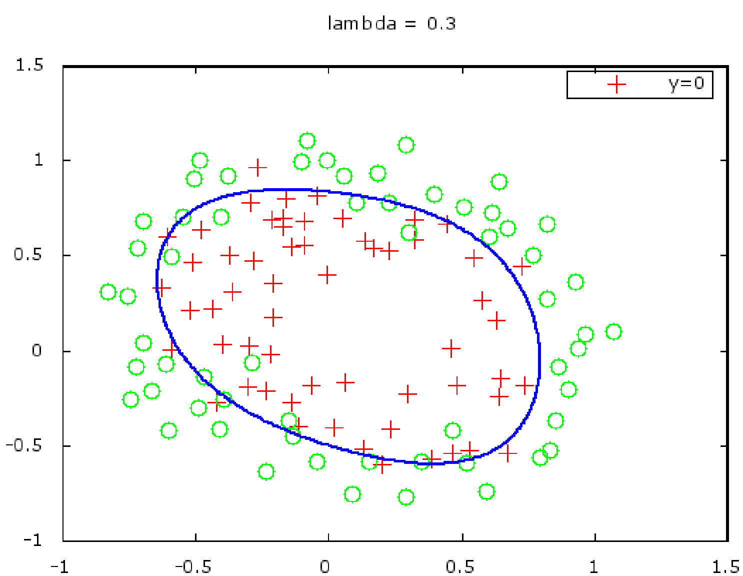
Demonstration: L1



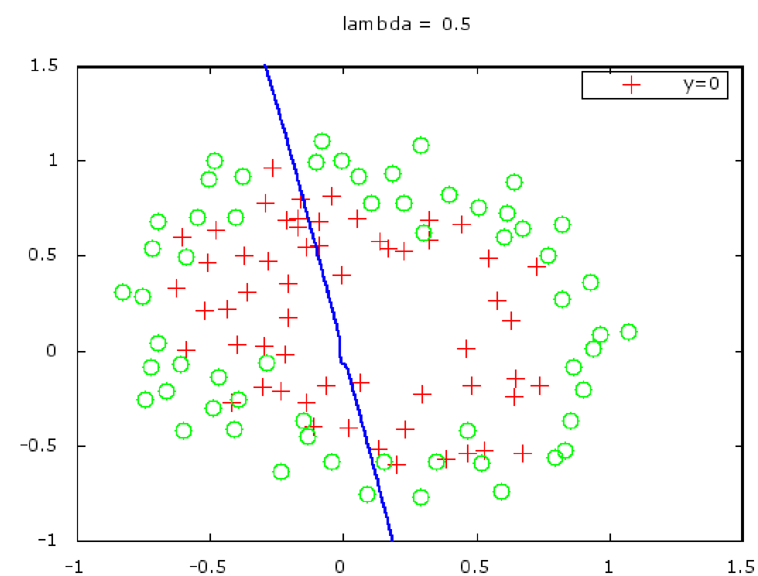
Sparsity: 0%, Likelihood: -0.2



Sparsity: 32%, Likelihood: -88



Sparsity: 60%, Likelihood:-73



Sparsity: 100%, Likelihood: -81

Summary

- Iteratively reweighted least squares (IRLS) can solve logistic regression in a robust and efficient manner.
- OLS
  - L1-norm (Ridge)
  - L2-norm (LASSO)
- Logistic Regression
  - L1-norm
  - L2-norm