FSAN815/ELEG815: Foundations of Statistical Learning

Gonzalo R. Arce

Chapter 14: Logistic Regression

Fall 2014

Course Objectives & Structure

The course provides an introduction to the mathematics of data analysis and a detailed overview of statistical models for inference and prediction.

Course Structure:

- Weekly lectures [notes: www.ece.udel.edu/∼arce/Courses]
- Homework & computer assignments [30%]
- Midterm & Final examinations [70%]

Textbooks:

- Papoulis and Pillai, Probability, random variables, and stochastic processes.
- Hastie, Tibshirani and Friedman, The elements of statistical learning.
- Haykin, Adaptive Filter Theory.

Logistic Regression

Logistic Regression

The logistic regression model arises from the desire to model the posterior probabilities of the K classes via linear functions in x, while at the same time ensuring that they sum to one and remain in $[0,1]$.

$$
\begin{aligned}
log \frac{Pr(G=1|X=\mathbf{x})}{Pr(G=K|X=\mathbf{x})} &= \beta_{10} + \boldsymbol{\beta}_1^T \mathbf{x} \\
log \frac{Pr(G=2|X=\mathbf{x})}{Pr(G=K|X=\mathbf{x})} &= \beta_{20} + \boldsymbol{\beta}_2^T \mathbf{x} \\
&. \\
&. \\
log \frac{Pr(G=K-1|X=\mathbf{x})}{Pr(G=K|X=\mathbf{x})} &= \beta_{(K-1)0} + \boldsymbol{\beta}_1^T \mathbf{x}
\end{aligned} \tag{1}
$$

Logistic Regression

Here $\boldsymbol{\beta}_{\boldsymbol{k}} = [\beta_{k1}, \beta_{k2}, ..., \beta_{kp}]^T$, $\mathbf{x} = [x_1, x_2, ..., x_p]^T$.
The model is specified in terms of $K-1$ log-odds or logit transformations(reflecting the constraint that the probabilities sum to one).
A simple calculation shows that

$$\begin{aligned} Pr(G=k|X=\mathbf{x}) &= \frac{exp(\beta_{k0}+\boldsymbol{\beta}_k^T\mathbf{x})}{1+\sum_{l=1}^{K-1} exp(\beta_{l0}+\boldsymbol{\beta}_l^T\mathbf{x})}, k=1,2,...,K-1, \\ Pr(G=K|X=\mathbf{x}) &= \frac{1}{1+\sum_{l=1}^{K-1} exp(\beta_{l0}+\boldsymbol{\beta}_l^T\mathbf{x})} \end{aligned} \tag{2}$$

To emphasize the dependence on the entire parameter set $\theta = \{\beta_{10}, \boldsymbol{\beta}_1^Y, ..., \beta_{(K-1)0}, \boldsymbol{\beta}_{K-1}^T\}$, we denote the probabilities $Pr(G=k|X=\mathbf{x}) = p_k(\mathbf{x};\theta)$

Fitting Logistic Regression Models

- Logistic regression models are usually fit by maximum likelihood, using the conditional likelihood of G given X.

The log-likelihood for N observations is

$$l(\theta) = \sum_{i=1}^{N} \log p_{g_i}(\mathbf{x}_i;\theta) \tag{3}$$

where $p_k(x_i;\theta) = Pr(G = k|X = \mathbf{x}_i;\theta)$

Fitting Logistic Regression Models

Discuss in detail the two-class case.

- Code the two-class g_i via a $0/1$ response y_i, where $y_i = 1$ when $g_i = 1$, and $y_i = 0$ when $g_i = 2$.
- Let $p_1(\mathbf{x};\theta) = p(\mathbf{x};\theta)$, $p_2(\mathbf{x};\theta) = 1 - p(\mathbf{x};\theta)$

The log-likelihood can be written as

$$
\begin{aligned}
l(\theta) &= \sum_{i=1}^{N} \{ y_i \log p(\mathbf{x}_i;\boldsymbol{\beta}) + (1-y_i)\log(1-p(\mathbf{x}_i;\boldsymbol{\beta}))\} \\
&= \sum_{i=1}^{N} \{ y_i(\log p(\mathbf{x}_i;\boldsymbol{\beta}) - \log(1-p(\mathbf{x}_i;\boldsymbol{\beta}))) + \log(1-p(\mathbf{x}_i;\boldsymbol{\beta}))\} \\
&= \sum_{i=1}^{N} \{ y_i \log \frac{Pr(G=1|X=\mathbf{x}_i)}{Pr(G=2|X=\mathbf{x}_i)} + \log(Pr(G=2|X=\mathbf{x}_i))\} \\
&= \sum_{i=1}^{N} \{ y_i \boldsymbol{\beta}^T \mathbf{x}_i - \log(1+e^{\boldsymbol{\beta}^T \mathbf{x}_i})\}
\end{aligned}
\tag{4}
$$

Fitting Logistic Regression Models

Here $\boldsymbol{\beta} = \{\beta_{10}, \boldsymbol{\beta}_1\}$, and we assume that the vector of inputs $\mathbf{x}_i$ includes the constant term 1 to accommodate the intercept.

To maximize the log-likelihood, we set its derivatives to zero. These score equations are

$$\frac{\partial l(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} = \sum_{i=1}^{N} \mathbf{x}_i (y_i - p(\mathbf{x}_i; \boldsymbol{\beta})) = 0 \tag{5}$$

which are $p+1$ equations nonlinear in $\boldsymbol{\beta}$.

Since the first component of $\mathbf{x}_i$ is 1, the first score equation specifies that $\sum_{i=1}^{N} y_i = \sum_{i=1}^{N} p(\mathbf{x}_i; \boldsymbol{\beta})$, the expected number of class ones matches the observed number.

Newton–Raphson algorithm

To solve the score equations (5), we use the Newton–Raphson algorithm, which requires the second-derivative or Hessian matrix

$$\frac{\partial^2 l(\boldsymbol{\beta})}{\partial \boldsymbol{\beta} \partial \boldsymbol{\beta}^T} = -\sum_{i=1}^{N} \mathbf{x}_i \mathbf{x}_i^T p(\mathbf{x}_i; \boldsymbol{\beta})(1 - p(\mathbf{x}_i; \boldsymbol{\beta})) \tag{6}$$

Starting with $\boldsymbol{\beta}^{old}$, a single Newton update is

$$\boldsymbol{\beta}^{new} = \boldsymbol{\beta}^{old} - \left(\frac{\partial^2 l(\boldsymbol{\beta})}{\partial \boldsymbol{\beta} \partial \boldsymbol{\beta}^T}\right)^{-1} \frac{\partial l(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} \tag{7}$$

where the derivatives are evaluated at $\boldsymbol{\beta}^{old}$.

Newton Raphson algorithm

- Write the score and Hessian in matrix notation.
- $\mathbf{y}$: N vector of y_i values.
- $\mathbf{x}_i$: $p+1$ vector of input.
- $\mathbf{X}$: $N \times (p+1)$ matrix of $\mathbf{x}_i$ values.
- $\mathbf{p}$: N vector of fitted probabilities with ith element $p(\mathbf{x}_i; \boldsymbol{\beta}^{old})$.
- $\mathbf{W}$: $N \times N$ diagonal matrix of weights with ith diagonal element $p(\mathbf{x}_i; \boldsymbol{\beta}^{old})(1 - p(\mathbf{x}_i; \boldsymbol{\beta}^{old}))$.

Fitting Logistic Regression Models

Then we have

$$\begin{aligned} \frac{\partial l(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} &= \mathbf{X}^T(\mathbf{y}-\mathbf{p}) \\ \frac{\partial^2 l(\boldsymbol{\beta})}{\partial \boldsymbol{\beta} \partial \boldsymbol{\beta}^T} &= -\mathbf{X}^T\mathbf{W}\mathbf{X} \end{aligned} \tag{8}$$

The Newton step is thus

$$\begin{aligned} \boldsymbol{\beta}^{new} &= \boldsymbol{\beta}^{old} + (\mathbf{X}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T(\mathbf{y}-\mathbf{p}) \\ &= (\mathbf{X}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}(\mathbf{X}\boldsymbol{\beta}^{old} + \mathbf{W}^{-1}(\mathbf{y}-\mathbf{p})) \\ &= (\mathbf{X}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}\mathbf{z} \end{aligned} \tag{9}$$

Fitting Logistic Regression Models

In the second and third line we have re-expressed the Newton step as a weighted least squares step, with the response

$$\mathbf{z} = \mathbf{X}\boldsymbol{\beta}^{old} + \mathbf{W}^{-1}(\mathbf{y} - \mathbf{p}) \tag{10}$$

sometimes known as the *adjusted response*.

- These equations get solved repeatedly, since at each iteration $\mathbf{p}$ changes, and hence so does $\mathbf{W}$ and $\mathbf{z}$.

This algorithm is referred to as *iteratively reweighted least squares* or IRLS, since each iteration solves the weighted least squares problem:

$$\boldsymbol{\beta}^{new} \leftarrow \arg\min_{\boldsymbol{\beta}} (\mathbf{z} - \mathbf{X}\boldsymbol{\beta})^T \mathbf{W} (\mathbf{z} - \mathbf{X}\boldsymbol{\beta}) \tag{11}$$

Exercise 4.10

Weekly data set is part of the ISLR package. It contains 1089 weekly percentage returns for 21 years, from the beginning of 1990 to the end of 2010.

Lag1 to Lag5	The percentage returns for each of the five previous trading weeks
Volume	The number of shares traded on the previous week, in billions
Today	The percentage return on the week in question
Direction	Whether the market was Up or Down on this week

Exercise

(b) Use the full data set to perform a logistic regression.

```
library(ISLR)
attach(Weekly)
glm.fit=glm(Direction∼Lag1+Lag2+Lag3+Lag4+Lag5+Volume,data=Weekly,famil
summary(glm.fit)
```

Coefficients:

	Estimate	Std. Error	z value	Pr(>|z|)	
(Intercept)	0.26686	0.08593	3.106	0.0019	**
Lag1	-0.04127	0.02641	-1.563	0.1181	
Lag2	0.05844	0.02686	2.175	0.0296	*
Lag3	-0.01606	0.02666	-0.602	0.5469	
Lag4	-0.02779	0.02646	-1.050	0.2937	
Lag5	-0.01447	0.02638	-0.549	0.5833	
Volume	-0.02274	0.03690	-0.616	0.5377	

According to p-value, lag1, lag2 and lag4 appear to be statistically significant.

Exercise

(c) Compute the confusion matrix and overall fraction of correct predictions.

```
glm.probs=predict(glm.fit,type="response")
glm.pred =rep("Down",1089)
glm.pred [glm.probs>.5]="Up"
table(glm.pred,Direction)
mean(glm.pred==Direction)
```

```
> table(glm.pred,Direction)
        Direction
glm.pred Down  Up
    Down   54  48
    Up    430 557
> mean(glm.pred==Direction)
[1] 0.5610652
```

Exercise

(d) Fit the logistic regression model using a training data then predict.

```
train=(Year<2009)
A=(Year==2009)
B=(Year==2010)
Weekly.2009=Weekly[A,]
Weekly.2010=Weekly[B,]
Direction.2009=Direction[A]
Direction.2010=Direction[B]
glm.fit=glm(Direction~Lag2,data=Weekly,family=binomial,subset=train)
glm.probs=predict(glm.fit,Weekly.2009,type="response")
glm.pred =rep("Down",52)
glm.pred [glm.probs>.5]="Up"
table(glm.pred,Direction.2009)
mean(glm.pred==Direction.2009)
```

Exercise

```
> table(glm.pred,Direction.2009)
        Direction.2009
glm.pred Down Up
    Down    4  4
    Up     19 25
> mean(glm.pred==Direction.2009)
[1] 0.5576923
```

Figure: prediction for 2009

```
> table(glm.pred,Direction.2010)
        Direction.2010
glm.pred Down Up
    Down    5  1
    Up     15 31
> mean(glm.pred==Direction.2010)
[1] 0.6923077
```

Figure: prediction for 2010

Exercise

(d) Fit the LDA model using a training data then predict.

```
library(MASS)
lda.fit=lda(Direction~Lag2,data=Weekly,subset=train)
lda.pred=predict(lda.fit,Weekly.2009)
lda.class=lda.pred$class
table(lda.class,Direction.2009)
mean(lda.class==Direction.2009)
```

```
> table(lda.class,Direction.2009)
         Direction.2009
lda.class Down Up
     Down    4  4
     Up     19 25
> mean(lda.class==Direction.2009)
[1] 0.5576923
```

Figure: prediction for 2009

```
> table(lda.class,Direction.2010)
         Direction.2010
lda.class Down Up
     Down    5  1
     Up     15 31
> mean(lda.class==Direction.2010)
[1] 0.6923077
```

Figure: prediction for 2010

Exercise

(e) Fit the QDA model using a training data then predict.

```
qda.fit=qda(Direction~Lag2,data=Weekly,subset=train)
qda.pred=predict(qda.fit,Weekly.2009)
qda.class=qda.pred$class
table(qda.class,Direction.2009)
mean(qda.class==Direction.2009)
```

```
> table(qda.class,Direction.2009)
         Direction.2009
qda.class Down Up
     Down    0  0
     Up     23 29
> mean(qda.class==Direction.2009)
[1] 0.5576923
```

Figure: prediction for 2009

```
> table(qda.class,Direction.2010)
         Direction.2010
qda.class Down Up
     Down    0  0
     Up     20 32
> mean(qda.class==Direction.2010)
[1] 0.6153846
```

Figure: prediction for 2010