MTH499/599 Lecture Notes 05

Donghui Yan

Department of Math, Umass Dartmouth

Outline

- Logistic regression
- Multiple logistic regression

Motivation

- In many real problems, one is interested in estimating $P(Y \mid X)$
  - e.g., what's the chance a loan application will be approved given the customer's profile
  - e.g., what's the chance that one will click product B given that he is viewing A
  - e.g., what's the probability that an email is a spam
- In such case, $f(X) = P(Y \mid X)$
  - We hope to estimate $P(Y \mid X)$ given $(X_1, Y_1), ..., (X_n, Y_n)$
  - Can be done in many different ways
    - Depending on the class of functions f one is interested in
    - One particular choice is logistic regression.

Logistic regression

- A very old method but still popular
  - Applicable to discrete responses
  - Simple but often very competitive
  - People in applied domain like it due to easy interpretation
  - One of the most popular models used in industry
- Central idea is to model the log odds ratio

$$\log \frac{P(Y=C|\boldsymbol{X}=\boldsymbol{x})}{1-P(Y=C|\boldsymbol{X}=\boldsymbol{x})} \stackrel{def}{=} a(\boldsymbol{x})$$

where $P(Y=C|\boldsymbol{x})$ is the posterior probability for $C \in \{0,1\}$

  - $a(\boldsymbol{x})$ is often a linear function, e.g.,

$$a(\boldsymbol{x}) = \beta_0 + \beta_1 x^{(1)} + ... + \beta_p x^{(p)}.$$

What the data may look like

```
These columns are features                  Label
---------------------------------          -----
2.6606, 3.1681, 1.9619, 0.18662,             0
3.931, 1.8541, -0.023425, 1.2314,            0
0.01727, 8.693, 1.3989, -3.9668,             0
3.2414, 0.40971, 1.4015, 1.1952,             0
2.2504, 3.5757, 0.35273, 0.2836,             0
-1.3971, 3.3191, -1.3927, -1.9948,           1
0.3901, -0.1428, -0.0320, 0.3508,            1
-1.6677, -7.1535, 7.8929, 0.96765,           1
-3.8483, -12.8047, 15.6824, -1.281,          1
-3.5681, -8.2130, 10.0830, 0.9677,           1
```

♠ Future data for prediction will not have a label. Labels in test set only used for performance evaluation.

Why log odds ratio?

- Our goal is to model relationship $f : X \to Y$
  - Can we do simple linear regression $Y = f(X) = X\beta$?
    - No, that would require $Y \in \mathbb{R}$, instead of discrete responses
- Recall simple linear model can also be viewed as

$$\mathbb{E}(\boldsymbol{Y}|\boldsymbol{X}) = \mu = \boldsymbol{X}\boldsymbol{\beta}$$

  - Apply similar idea to discrete response when $Y \in \{0, 1\}$
    - $\mathbb{E}(Y|X) = P(Y = 1|X)$
  - **But** $P(Y = C|X) > 0$ while $\boldsymbol{X}\boldsymbol{\beta} \in \mathbb{R}$
    - Logarithm of $P(.)$ might be a fix
    - **But** $P(Y = C|X) \leq 1$, which then requires $\boldsymbol{X}\boldsymbol{\beta} \leq 0$
  - How to regress while conveniently enforce $0 \leq P(.) \leq 1$?
    - ♠ One solution is *logit* transformation to $P(.)$.

Generalized linear model

- Recall in simple linear model, *mean* response Y modeled as

$$\mathbb{E}\boldsymbol{Y} = \mu = \boldsymbol{X}\boldsymbol{\beta}$$

- It is possible to extend as

$$\mathbb{E}\boldsymbol{Y} = \mu = g^{-1}(\boldsymbol{X}\boldsymbol{\beta})$$

- Resulting model called generalized linear model (GLM)
  - $g()$ as a link function
  - Developed by Nelder and Wedderburn to unify various models
  - Standard text: *Generalized Linear Models* by McCullagh and Nelder (1989).

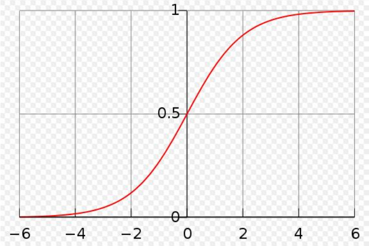
Example of GLM models

Distribution	Typical uses	Link	Mean function
Normal	Linear-response data	Identity	$\mu = X\beta$
Gamma	Exponential response data	Inverse	$\mu = -(X\beta)^{-1}$
$(\text{Gaussian})^{-1}$		$()^{-2}$	$\mu = (X\beta)^{-1/2}$
Poisson	Count of occurrences	Log	$\mu = exp(X\beta)$
Bernoulli	Outcome of 0/1	Logit	$\mu = \frac{1}{1+exp(-X\beta)}$
Binomial	Count of 0/1		
Categorial	K-way occurrences		
Multinomial	Multinomial response		

The logit function

Let $Y = 1$ with probability p and 0 otherwise. Then

$$\mathbb{E}(Y|x) = P(Y = 1|\boldsymbol{x}) = \frac{1}{1 + exp(-a(\boldsymbol{x}))}.$$



Model fitting

- Model fitting is often done via MLE
- Given data $(\boldsymbol{x}_i, y_i), i = 1, ..., n$ and assume parameters $\boldsymbol{\beta}$, the likelihood function is

$$l((y_i)_{i=1}^n; \boldsymbol{\beta}) = \Pi_{i=1}^n p(C|\boldsymbol{x}_i)^{y_i} [1 - p(C|\boldsymbol{x}_i)]^{1-y_i}$$

- The solution is given by solving

$$\hat{\beta}_{ML} = \arg\max_{\boldsymbol{\beta}} l((y_i)_{i=1}^n; \boldsymbol{\beta}).$$

The iterated Newton-Raphson

- The MLE can be solved by iterated Newton-Raphson

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

  whit the derivatives evaluated at β^{old}.
- A good starting value for β^{old} is often 0.

The idea of Newton-Raphson

- Newton-Raphson is a method to find zero points of a function
- It is related to Taylor's series expansion of a function $f(x)$

$$f(x_{n+1}) \approx f(x_n) + f'(x_n)(x_{n+1} - x_n).$$

- Now if x_{n+1} is a zero point of $f(x)$, then

$$0 \approx f(x_n) + f'(x_n)(x_{n+1} - x_n) \Leftrightarrow x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)},$$

  which gives a rule of updating the point series.
- MLE seeks to find solution for $\frac{\partial l(\beta)}{\partial \beta} = 0$.

Newton-Raphson updates in matrix

- Let $\boldsymbol{X}_{N\times p}$ be the design matrix, $\boldsymbol{y}_{N\times 1}$ be the response vector,
- $\boldsymbol{p}_{N\times 1} = (p(x_i;\beta^{old}))_{i=1}^{N}$ be vector of fitted probabilities,
- $\boldsymbol{W}_{N\times N} = diag(\ldots, p(x_i;\beta^{old})(1-p(x_i;\beta^{old})), \ldots)$ be a weight matrix. Then

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

Newton-Raphson updates in matrix

- The Newton-Raphon update thus becomes

$$\begin{aligned}\beta^{new} &= \beta^{old} + (\boldsymbol{X}^T\boldsymbol{W}\boldsymbol{X})^{-1}\boldsymbol{X}^T(\boldsymbol{y}-\boldsymbol{p}) \\ &= (\boldsymbol{X}^T\boldsymbol{W}\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{W}\left(\boldsymbol{X}\beta^{old} + \boldsymbol{W}^{-1}(\boldsymbol{y}-\boldsymbol{p})\right) \\ &\triangleq (\boldsymbol{X}^T\boldsymbol{W}\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{W}\boldsymbol{z}.\end{aligned}$$

- Each iteration updates $\beta^{old} \leftarrow \beta^{new}$, $\boldsymbol{p}$ and $\boldsymbol{W}$ accordingly.
- The algorithm typically converges to local optimum
  - As the log-likelihood is concave.

Logistic regression in R

- Treated as a special case of generalized linear model
- Use R function $glm()$
  - $glm(y \sim x, family = binomial(link = "logit"))$
  - Requires y be transformed to $0/1$.
- One extra step for classification
  - Truncate $P(Y = C|\boldsymbol{X} = \boldsymbol{x})$ to $0/1$ (label)
  - Output $0/1$ as the label of a given input $\boldsymbol{x}$.

Example: bank note classification

- Source: UC Irvine Machine Learning Repository
- Images taken for bank notes and evaluated for authenticity
  - 400 x 400 pixels 600 dpi
- Four features from wavelet transformation of the images
  - Variance
  - Skewness
  - Curtosis
  - Entropy
- Label: authentic or not authentic (Binary).

Logistic regression output

```
glm(formula = y ~ x, 1:4], family = binomial(link = "logit"))

Coefficients   Estimate   Std. Error   z-value   Pr(>|z|)
(Intercept)      17.701      8.594      2.060    0.0394 *
x[, 1]          -18.528      9.165     -2.022    0.0432 *
x[, 2]          -10.027      4.878     -2.056    0.0398 *
x[, 3]          -12.818      6.297     -2.036    0.0418 *
x[, 4]          -2.700       1.579     -1.710    0.0872 .
---
Signif. codes:  0 ?***? 0.001 ?**? 0.01 ?*? 0.05 ?.? 0.1 ? ? 1

Null deviance: 942.561  on 685  degrees of freedom
Residual deviance:  13.265  on 681  degrees of freedom
AIC: 23.265

Number of Fisher Scoring iterations: 15
```

Output from logistic regression

- Notation: LL=Log likelihood
- Models
  - *Saturated model*: assuming each data point has a parameter
  - *Proposed model*: p parameters + intercept (p+1 parameters)
  - *Null model*: Only intercept
- Deviance (Model) = 2(LL(Saturated Model) - LL(Model))
  - $\sim \chi^2, \;\; df = df_{Sat} - df_{Model} = n - (p+1)$
  - *Small means the model explains the data well*
- Null Deviance = 2(LL(Saturated Model) - LL(Null))
  - $df = df_{Sat} - df_{Null} = n - 1$
- Residual Deviance = 2(LL(Saturated Model) - LL(Model))
  - $df = df_{Sat} - df_{Model} = n - (p+1)$

♠ We do not use R^2 to assess the model fitting.

R code

```
tmp<-read.table(file="bankNote.Data", sep=",");
n<-nrow(tmp);
x<-matrix(0,n,ncol(tmp));
for(i in 1:ncol(tmp)) { x[,i]<-tmp[,i];}
idx<-sample(1:n, floor(n/2));
xtr<-x[idx,]; xts<-x[-idx,];

mylogit<-glm(xtr[,5]~xtr[,1:4], family=binomial(link="logit"));
b<-mylogit$coefficients;

logits<-matrix(0,nrow(xts),1);
for (i in 1:nrow(xts))
{
      logits[i]<-b[1]+sum(xts[i,1:4]*b[2:5]);
}
logits<-exp(logits)/(1+exp(logits));
```

R code (continued)

```
classDF <- data.frame(response = xts[,5],
                      predicted = round(logits,0))
xtabs(~ predicted + response, data = classDF)
```

- Output from classification (a confusion matrix)

```
         response
predicted   0   1
        0 371   2
        1  10 303
```

- So accuracy on the test set is

$$(371 + 303)/(371 + 303 + 10 + 2) = 98.25\%.$$

The spam filter example

- The goal is to design an automatic spam detector
- Information collected from 4601 email messages
  - For which one knows if it is *email* or *spam*
- Formulate as a supervised classification problem
  - Or (logistic) regression problem with discrete response
- Data available at *ftp.ics.uci.edu*
  - Donated by *George* Forman of HP Lab, Palo Alto, CA.

The spam filter example

- Similar as typical applications, most importantly
  - What features to use?
- Follow the tradition of document/text processing
  - Use relative frequencies of 57 commonly used words
  - One instance of the popular 'Bag of words'
- Why may this help?

	george	you	your	hp	free	hpl	!	our	re	edu	remove
spam	0.00	2.26	1.38	0.02	0.52	0.01	0.51	0.51	0.13	0.01	0.28
email	1.27	1.27	0.44	0.90	0.07	0.43	0.11	0.18	0.42	0.29	0.01

Features breakdown

- 48 quantitative predictors
    - e.g., *business, address, internet, free, and george* (customizable for users)
- % characters that match one of 6 special characters
- Avg length of uninterrupted sequences of capital letters
- Length of longest uninterrupted sequence of capital letters
- Sum of length of uninterrupted sequences of capital letters.

R code on the Spam data (training)

```
##The Spam data
tmp<-read.table("spam.Data",sep=",");
n<-nrow(tmp); p<-ncol(tmp);

x<-matrix(0,n,p);
for(j in 1:p) { x[,j]<-tmp[,j]; }

##Split the data according to HTF for comparison
T<-3065;
idx2<-sample(1:n,T, replace=FALSE);
xtr<-x[idx2,]; xts<-x[-idx2,];

mylogit<-glm(xtr[,p] ~ xtr[,1:(p-1)],
                      family=binomial(link="logit"));
```

R code on the Spam data (test)

```
b<-mylogit$coefficients;

logits<-matrix(0,nrow(xts),1);
for (i in 1:nrow(xts))
{
        logits[i]<-b[1]+sum(xts[i,1:(p-1)]*b[2:p]);
}
logits<-exp(logits)/(1+exp(logits));

classDF <- data.frame(response = xts[,p],
                   predicted = round(logits,0));
z<-xtabs(~ predicted + response, data = classDF);
acc<-sum(diag(z))/sum(z);
cat("The accuracy on the test set is", acc,"\n");
```

Test on the Spam data

The confusion matrix is given by

```
    cres
true   0   1
   0 891  49
   1  50 542
The accuracy on the test set is 0.9329
```

- Accuracy = (891+542)/(891+542+49+54) = 0.9329.

Multiple logistic regression

- In binary logistic regression, only two classes
  - $\mathcal{C} = \{0, 1\}$, and model as

$$\log \frac{P(Y = 1 \mid X)}{1 - P(Y = 1 \mid X)} = \boldsymbol{X\beta}$$

- However, real applications often have more classes
  - $\mathcal{C} = \{1, 2, ..., K\}$
- How to use binary formulation for multiple logistic regression?

What the data may look like

```
These columns are features              Label
--------------------------------------- -----
1.6136  1.1070  0.3213  0.2872  0.2805    1
-0.1487 -1.1058  1.6925  0.9297  0.6540    3
-1.2323 -0.2032 -0.9434  0.1295 -0.2289    2
-0.3057 -0.5924  1.4797  0.1016 -0.6104    2
0.8225  1.2524 -1.8159 -0.6828 -0.6753    1
0.5977 -1.1583  0.8941 -0.7861 -1.5364    4
-0.4745  0.0295  1.6211  1.4213  0.8820    1
0.1057  0.7401 -1.4808  1.7530  2.5282    4
1.3004  1.4805 -1.3826  1.0880 -0.0622    3
-0.5246  1.4543  0.7554 -1.6087  0.1330    2
```

♠ Future data for prediction will not have a label. Labels in test set only used for performance evaluation.

Multiple logistic regression

- Reformulate binary logistic regression as

$$\log \frac{P(Y=1 \mid X)}{P(Y=0 \mid X)} = \boldsymbol{X\beta}$$

  - View class $0 \in \mathcal{C} = \{0, 1\}$ as reference class
- In multiple case, $J \in \mathcal{C} = \{1, 2, ..., K\}$ as reference
  - Consider $K-1$ log odds-ratio like models

$$\begin{aligned}
\log \frac{P(Y=1 \mid X)}{P(Y=J \mid X)} &= \boldsymbol{X\beta_1}, \\
\log \frac{P(Y=2 \mid X)}{P(Y=J \mid X)} &= \boldsymbol{X\beta_2}, \\
\cdots \quad &\cdots \quad \cdots \\
\log \frac{P(Y=K \mid X)}{P(Y=J \mid X)} &= \boldsymbol{X\beta_K}.
\end{aligned}$$

Multiple logistic regression

The $K-1$ log odds-ratio like models imply

$$P_i = e^{\boldsymbol{X\beta_i}}.P_J, \quad i \in \{1, 2, ..., K\} \text{ and } i \neq J$$

Summing up the $K-1$ equations, we get

$$P_J \triangleq P(Y = J \mid X) = \left(1 + \sum_{j \neq J} e^{\boldsymbol{X\beta_j}}\right)^{-1}$$

and, all $P_i \triangleq P(Y = i \mid X)$ for $i \in \{1, 2, ..., K\} \setminus \{J\}$.

Classification based on multiple logistic regression

For $X = x$, calculate K posterior probabilities

$$P(Y = i \mid X = x),\ \ i \in \{1, 2, ..., K\}$$

Assign label to $X = x$ according to

$$\arg \max_{i \in \{1,2,...,K\}} P(Y = i \mid X = x)$$

- When $K = 2$, this reduces to rounding rule for label assignment

$$\begin{cases} 1, & \text{if } P(Y = 1 \mid X = x) > 1/2 \\ 0, & \text{otherwise.} \end{cases}$$

Example

- $P(Y = i \mid X)$ by multiple logistic regression, $i = 1, 2, 3, 4$

	P(Y=1|X)	P(Y=2|X)	P(Y=3|X)	P(Y=4|X)	Label
[1,]	0.2089	0.2567	0.2641	0.2703	4
[2,]	0.1865	0.2807	0.2694	0.2635	2
[3,]	0.2260	0.2362	0.2431	0.2946	4
[4,]	0.2222	0.1999	0.3191	0.2588	3
[5,]	0.2764	0.2503	0.2687	0.2046	1
[6,]	0.2453	0.2497	0.2502	0.2548	4
[7,]	0.2360	0.2683	0.2587	0.2370	2
[8,]	0.2967	0.2559	0.2513	0.1961	1
[9,]	0.2471	0.2669	0.2276	0.2584	2
[10,]	0.2002	0.2580	0.2471	0.2947	4

Multiple logistic regression in R

- Many R packages available
- A recently released R package "glmnet"
  - J. Friedman, T. Hastie, N. Simon and R. Tibshirani (2015)
  - Idea is to solve MLE via coordinate decent
    - Lasso or elastic-net regularized MLE
  - Optimizes over the entire regularization path.

Example: Normal data

- 2000 cases (Half for training and half for test)
- 20 features all from $\mathcal{N}(0, 1)$
- Label: randomly drawn from $\{1, 2, 3, 4\}$
- What would you expect about the accuracy on test set?

Multiple logistic regression in R

```
##Uncomment when first using "glmnet"
##install.packages("glmnet");
library(glmnet);

n<-2000;
x<-matrix(rnorm(n*20),n,20);

##Four-class multiple logistic regression
y4<-sample(1:4,n,replace=TRUE);

##To split the data into training and test set
idx<-sample(1:n,floor(n/2),replace=FALSE);
xtr<-x[idx,]; ytr<-y4[idx];
xts<-x[-idx,]; yts<-y4[-idx];
```

Multiple logistic regression in R (continued)

```
##Fit the multiple logistic regression model
myglm4<-glmnet(xtr,ytr,family="multinomial");

##Apply the trained model to the test set
mypred4<-predict(myglm4,newx=xts,type="response",s=0.01);
posteriprob<-mypred4[,,1];
yhat<-matrix(1,nrow(xts),1);
for(i in 1:nrow(xts))
{
        yhat[i]<-which.max(posteriprob[i,]);
}
acc<-sum(yhat==yts)/nrow(xts);
cat("Accuracy on the test set is", acc, "\n");
```

Example: UC Irvine wine quality (White)

- 4898 cases (Half for training and half for test)
- 11 features based on physicochemical tests
  - Fixed acidity
  - Volatile acidity
  - Citric acid
  - Residual sugar
  - Chlorides
  - Free sulfur dioxide
  - Total sulfur dioxide
  - Density
  - PH
  - Sulphates
  - Alcohol
- Score: 0–10 (3–9 indeed).

Multiple logistic regression in R

```
library(glmnet);

tmp<-read.csv("winequality.csv", header=TRUE, sep=";");
n<-nrow(tmp);
K<-ncol(tmp)-1;
x<-matrix(0,n,K);
for(i in 1:K) {x[,i]<-tmp[,i];}
y<-tmp[,K+1];

##To split the data into training and test set
idx<-sample(1:n,floor(n/2),replace=FALSE);
xtr<-x[idx,]; ytr<-y[idx];
xts<-x[-idx,]; yts<-y[-idx];
```

Multiple logistic regression in R (continued)

```
##Fit the multiple logistic regression model
myglm4<-glmnet(xtr,ytr,family="multinomial");

##Apply the trained model to the test set
mypred4<-predict(myglm4,newx=xts,type="response",s=0.01);
posteriprob<-mypred4[,,1];
yhat<-matrix(1,nrow(xts),1);
for(i in 1:nrow(xts))
{
        yhat[i]<-which.max(posteriprob[i,]);
}
acc<-sum(yhat+2==yts)/nrow(xts);
cat("Accuracy on the test set is", acc, "\n");
```