Discrete Response Model
Lecture 2

datascience@berkeley

Introduction to Binary Response Models and Linear Probability Model

Response Probability

In a binary response model, we model the response probability

$$P(y_i = 1|\mathbf{x_i}) = P(y = 1|x_{1,i}, x_{k,i}, \ldots, x_{k,i})$$

where

y_i is a $n \times 1$ vector of response, which takes the value of 0 or 1

$x_{j,i}$ is a $n \times 1$ vector of covariates (or explanatory variables) associated with individual $i, i = 1, 2, \ldots, n$

Many authors use π to denote the response probability, and this is the notation used in our text. So, I will follow this notation in this course.

$$P(y_i = 1|\mathbf{x_i}) = \pi_i(\mathbf{x_i})$$

To the precise, this is a conditional response probability, conditional on the set of explanatory variables $\mathbf{x_i}$ of individual i.

General Formulation of the Class of Binary Response Model

Suppressing the individual subscript for the time being:

$$P(y = 1|\mathbf{x}) = F(\beta_0 + \mathbf{x}\beta)$$

where $F()$ is a function taking on values between 0 and $1: 0 < F(z) < 1$, for all real numbers z.

Various nonlinear transformation have been proposed for the function $F(.)$ to make sure that probabilities are between 0 and 1.

Logit and Probit Specification of Binary Response Models

The logit transformation uses the logistic function, resulting in the following form. We will cover logistic regression model in great details in this course.

$$F(z) = \Lambda(z) = \frac{exp(z)}{1 + exp(z)}$$

This is the for a standard logistic random variable.

The probit transformation, resulting in the probit model, uses the standard normal distribution and takes the following form:

$$F(z) = \Phi(z) = \int_{-\infty}^{z} \phi(v)dv$$

where $\phi(z)$ is the standard normal PDF:

$$\phi(z) = (2\pi)^{-1/2} exp(-z^2/2)$$

Linear Probability Model

Before studying the logistic regression model, let's examine using the classical linear regression covered in $w203$ to binary response variable.

Recall that linear regression model takes the following form (suppressing the individual subscript):

$$y = \mathbf{x}\beta + \epsilon = \beta_0 + \beta_1 x_1 + \cdots + \beta_k x_k + \epsilon$$

where

$\mathbf{x}$ is a $1 \times k$ covariate vector (associated with individual i) and y is a $k \times 1$ response vector.

When y is a binary variable taking on values 0 and 1, we have

$$E(y|\mathbf{x}) = P(y|\mathbf{x})$$

This means that the probability of success (we are defining that the event $y = 1$ as "success"), $\pi(\mathbf{x}) = P(y|\mathbf{x})$ is a *linear function* of the explanatory variables, $x's$.

Introduction to Binary Response Models and Linear Probability Model

Shortcoming of Linear Probability Model

In LPM, β_j measures the chnage in probability of success when x_j changs by 1 unit, holding other explanatory variables constant:

$$\triangle P(y = 1|\mathbf{x}) = \beta_j \triangle x_j$$

I am going to refer you to the textbook for examples, because we will not study further LPM in this course.

The mechanics of applying the OLS to binary response variable is the same as those covered in $w203$. However, there are a few shortcomings of LPM which makes it very unattractive in modeling binary response variable.

1. The predicted probabilities can go out of the range of 0 and 1.
2. A probability is linearly related to the explanatory variables for all their possible values. This is not realistic. (See the example in the book for more details.)
3. LPM violates the homoskedastic assumption underneath the Classical Linear Regression Model. That is,

$$Var(y|\mathbf{x}) = p(\mathbf{x})[1 - p(\mathbf{x})]$$

Binomial Logistic Regression Model

The Logit Transformation and the Logistic Curve

The Most Common Transformation

- There are a number of solutions to prevent i from being outside the range of a probability.
- Most solutions rely on non-linear transformations to prevent these types of problems from occurring.
- The most commonly used transformation results in the logistic regression model

$$\begin{aligned}\pi_i &= P(y = 1|\mathbf{x}) \\ &= \frac{exp(\beta_0 + \mathbf{x}\beta)}{1 + exp(\beta_0 + \mathbf{x}\beta)} \\ &= \frac{exp(\beta_0 + \beta_1 x_1 + \cdots + \beta_k x_p)}{1 + exp(\beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p)}\end{aligned}$$

- Note that $exp(\beta_0 + \mathbf{x}\beta) > 0$ so that the numerator is always less than the denominator, forcing the response probability to go fall within the zero and one range: $0 < \pi_i < 1$

The Logistic regression can be written as log(Odd Ratio):

$$log\left(\frac{\pi_i}{1 - \pi_i}\right) = \beta_0 + \boldsymbol{x_i\beta}$$

Binomial Logistic Regression Model

The Logit Transformation

- The $\log\left(\frac{\pi_i}{1-\pi_i}\right)$ transformation is often referred to as the logit transformation:

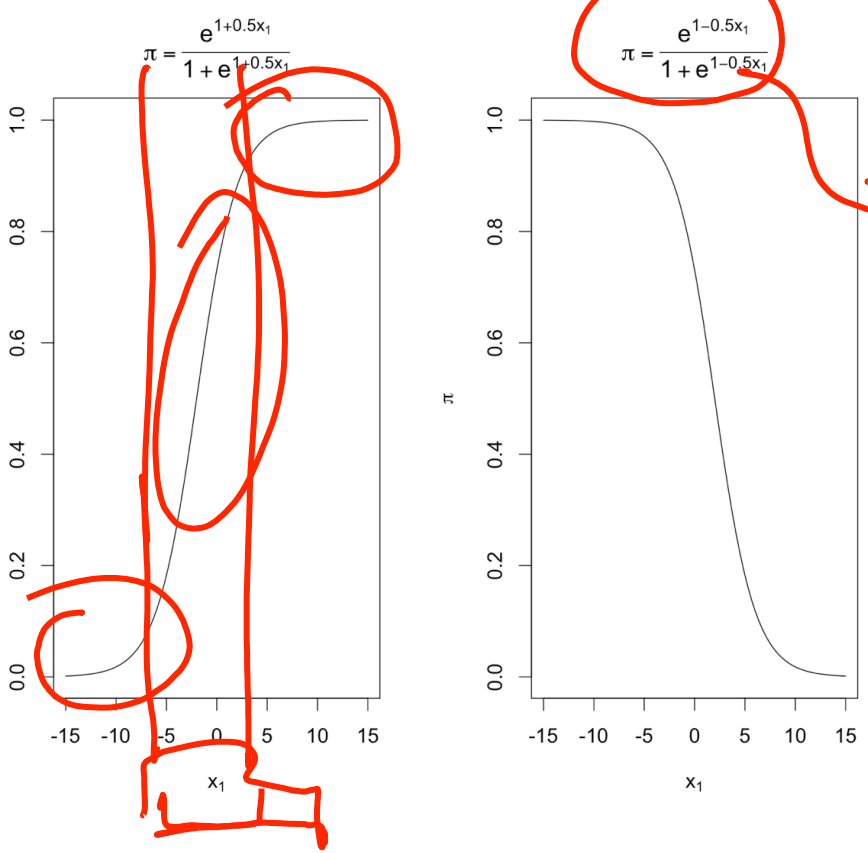
$$\text{logit}(\pi_i) = \beta_0 + \beta_1 x_{i1} + \quad + \beta_p x_{ip}$$

$$\beta_0 + \beta_1 x_{i1} + \quad + \beta_p x_{ip}$$

This part of the model is often referred to as the linear predictor.

Visualizing the Logistic Curve

When there is only one explanatory variable, $\beta_0 = 1$, and $\beta_1 = 0.5$ (or -0.5), a plot of π vs. x looks like the following:




```
par(mfrow = c(1,2))

beta0<-1
beta1<-0.5
curve(expr = exp(beta0+beta1*x)/(1+exp(beta0+beta1*x)),
    xlim = c(-15, 15), col = "black", main = expression(pi
    == frac(e^{1+0.5*x[1]}, 1+e^{1+0.5*x[1]})), xlab =
    expression(x[1]), ylab = expression(pi))

beta0<-1
beta1<--0.5
curve(expr = exp(beta0+beta1*x)/(1+exp(beta0+beta1*x)),
    xlim = c(-15, 15), col = "black", main = expression(pi
    == frac(e^{1-0.5*x[1]}, 1+e^{1-0.5*x[1]})), xlab =
    expression(x[1]), ylab = expression(pi))
```

Observations:

- $0 < \pi < 1$.
- When $\beta_1 > 0$, there is a positive relationship between x_1 and π. When $\beta_1 < 0$, there is a negative relationship between x_1 and π.
- The shape of the curve is somewhat similar to the letter *s*.
- Above $\pi = 0.5$ is a mirror image of below $\pi = 0.5$.
- The slope of the curve is dependent on the value of x_1. This is an important property worth remembering when interpreting the coefficients of a logistic regression model.
  - We can show this mathematically by taking the derivative with respect to x_1:

$$\frac{d\pi}{dx_1} = \beta_1 \pi(1-\pi)$$

Statistical Assumption of Binomial Logistic Regression Models

Statistical Assumptions

Recall from Lecture 1 the five assumptions underlying the binomial probability model. Some of them have their counterparts in the context of conditional response probability.

1. The dependent variable needs to be binary (and not ordinal); specifically, the conditional distribution of y given follows a *Bernoulli distribution*.
2. Observations are independent of each other. In fact, the error term of the model needs to follow an independent and identically distributed random variable.
3. No perfect collinearity.
4. Linearity assumption: linearity of independent variables and log-odds ratio.

The Requirement of No Complete Separation

Complete Separation:

There is a complete separation of data points if there exists a vector $\mathbf{b}$ that correctly allocates all observations to their response groups; that is,

$$\begin{cases} \mathbf{b}'\mathbf{x}_i > 0 & Y_i = 1 \\ \mathbf{b}'\mathbf{x}_i < 0 & Y_i = 2 \end{cases}$$

In this case, the maximum likelihood estimate does not exist; the log-likehood goes to 0 as iteration increases.

Parameter Estimation

Likelihood Function

Maximum likelihood estimation is used to estimate the parameters of the model. The likelihood function is

$$L(\beta_0, \ldots, \beta_p) = \prod_{i=1}^{n} \pi_i^{y_i}(1-\pi_i)^{(1-y_i)}$$

where

$$\begin{aligned}\pi_i &= P(y=1|\mathbf{x}) \\ &= \frac{exp(\beta_0 + \mathbf{x}\beta)}{1 + exp(\beta_0 + \mathbf{x}\beta)} \\ &= \frac{exp(\beta_0 + \beta_1 x_1 + \cdots + \beta_k x_p)}{1 + exp(\beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p)}\end{aligned}$$

and the corresponding log-likelihood function is

$$log(L(\beta_0, \ldots, \beta_p|y_1, \ldots, y_n)) = log\left(\prod_{i=1}^{n} \pi_i^{y_i}(1-\pi_i)^{(1-y_i)}\right)$$

$$= \sum_{i=1}^{n} y_i log(\pi_i) + (1-y_i)log(1-\pi_i)$$

Deriving the MLE and the glm( ) Function in R

Taking derivatives with respect to $\beta_0, \ldots, \beta_p$, setting them equal to 0, and then solving for the parameters lead to the MLEs. These parameter estimates are denoted by $\hat{\beta}_0, \ldots, \hat{\beta}_p$. Corresponding estimates of are

$$\hat{\pi} = \frac{exp(\hat{\beta}_0 + \hat{\beta}_1 x_1 + \cdots + \hat{\beta}_k x_p)}{1 + exp(\hat{\beta}_0 + \hat{\beta}_1 x_1 + \cdots + \hat{\beta}_p x_p)}$$

Unfortunately, there are no closed form expressions that can be written out for $\hat{\beta}_0, \ldots, \hat{\beta}_p$. The MLEs are estimated using numerical procedures.

The Newton-Raphson procedure can be used for finding the MLE of π in a homogeneous population setting. (See the text for an example.)

In this course, we will use extensively the R function, glm(). Though we will explain the utility of this function, you are expected to read the documentation of this function, which can be shown in R by typing "?glm".

Iterated Reweighted Least Squares

glm() in R uses iteratively reweighted least squares as the default numerical method, which is shown in an excerpt of R documentation below.

> `method`
>
> the method to be used in fitting the model. The default method `"glm.fit"` uses iteratively reweighted least squares (IWLS): the alternative `"model.frame"` returns the model frame and does no fitting. User-supplied fitting functions can be supplied either as a function or a character string naming a function, with a function which takes the same arguments as `glm.fit`. If specified as a character string it is looked up from within the stats namespace.

This is an iterative process, which is typical for many numerical methods, and continues until convergence is found or a prior-specified maximum number of iterations is reached.

The function for Iterated Reweighted LS is documented, among many places, in the following website: http://www.inside-r.org/packages/cran/Rfit/docs/irls.

An Example

Example

Goal: Estimate the probability of success for a placekick (based on 1,425 placekicks from the 1995 NFL season.

The response variable is referred to as "Good" in the dataset. It is a 1 for successful placekicks and a 0 for failed placekicks.

Explanatory variables in the dataset:

- Week: week of the season
- Distance: Distance of the placekick in yards
- Change: Binary variable denoting lead-change (1) vs. non-lead-change (0) placekicks; successful lead-change placekicks are those that change which team is winning the game.
- Elap30: Number of minutes remaining before the end of the half with overtime placekicks receiving a value of 0
- PAT: Binary variable denoting the type of placekick where a point after touchdown (PAT) is a 1 and a field goal is a 0
- Type: Binary variable denoting dome (0) vs. outdoor (1) placekicks
- Field: Binary variable denoting grass (1) vs. artificial turf (0) placekicks
- Wind: Binary variable for placekicks attempted in windy conditions (1) vs. non-windy conditions (0); I define windy as a wind stronger than 15 miles per hour at kickoff in an outdoor stadium

Example

There are 1,425 placekick observations from the 1995 NFL season that are within this dataset.

```
setwd("/Users/jeffrey/Documents/JStuff/AdvStat/pgms/CatData/Chapter2")
list.files("/Users/jeffrey/Documents/JStuff/AdvStat/pgms/CatData/Chapter2")

df<-read.table(file = "placekick.csv", header = TRUE, sep = ",")
str(df)
```

```
'data.frame':   1425 obs. of  9 variables:
 $ week    : int  1 1 1 1 1 1 1 1 1 1 ...
 $ distance: int  21 21 20 28 20 25 20 27 44 32 ...
 $ change  : int  1 0 0 0 0 0 0 1 1 0 ...
 $ elap30  : num  24.72 15.85 0.45 13.55 21.87 ...
 $ PAT     : int  0 0 1 0 1 0 1 0 0 0 ...
 $ type    : int  1 1 1 1 0 0 0 0 0 0 ...
 $ field   : int  1 1 1 1 0 0 0 0 0 0 ...
 $ wind    : int  0 0 0 0 0 0 0 0 0 0 ...
 $ good    : int  1 1 1 1 1 1 1 1 1 1 ...
```

```
> head(df)
  week distance change  elap30 PAT type field wind good
1    1       21      1 24.7167   0    1     1    0    1
2    1       21      0 15.8500   0    1     1    0    1
3    1       20      0  0.4500   1    1     1    0    1
4    1       28      0 13.5500   0    1     1    0    1
5    1       20      0 21.8667   1    0     0    0    1
6    1       25      0 17.6833   0    0     0    0    1
```

Frequency table of the dependent variable of interest:

```
> table(df$good)

   0    1
 163 1262
> prop.table(table(df$good))

       0        1
0.114386 0.885614
```

Example

For this particular example, we are only going to use the distance explanatory variable to estimate the probability of a successful placekick. Thus, our logistic regression model is

$$logit(\pi) = \beta_0 + \beta_1 x_1$$

where Y is the good response variable and x_1 denotes the for the placekick.

```
> mod.fit<-glm(formula = good ~ distance, family =binomial(link = logit), data = df)
> mod.fit

Call:  glm(formula = good ~ distance, family = binomial(link = logit),
    data = df)

Coefficients:
(Intercept)     distance
      5.812       -0.115

Degrees of Freedom: 1424 Total (i.e. Null);  1423 Residual
Null Deviance:      1013
Residual Deviance: 775.7        AIC: 779.7
```

The estimated logistic regression model is

$$logit(\pi) = 5.812 - 0.115 distance$$

Example

There is actually much more information stored within the mod.fit object than showed so far. Through the use of the names() function, we obtain the following list of items:

```
> names(mod.fit)
 [1] "coefficients"      "residuals"       "fitted.values"   "effects"          "R"
 [6] "rank"              "qr"              "family"          "linear.predictors" "deviance"
[11] "aic"               "null.deviance"   "iter"            "weights"          "prior.weights"
[16] "df.residual"       "df.null"         "y"               "converged"        "boundary"
[21] "model"             "call"            "formula"         "terms"            "data"
[26] "offset"            "control"         "method"          "contrasts"        "xlevels"
>
```

```
> length(mod.fit$coefficients)
[1] 2
> mod.fit$coefficients
(Intercept)    distance
  5.8120798  -0.1150267
> mod.fit$coefficients[1]
(Intercept)
    5.81208
> mod.fit$coefficients[1]
(Intercept)
    5.81208
```

Example

To see a summary of all the information in mod.fit, we can use the summary() function.

```
> summary(object = mod.fit)

Call:
glm(formula = good ~ distance, family = binomial(link = logit),
    data = df)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.7441   0.2425   0.2425   0.3801   1.6092

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  5.812080   0.326277   17.81   <2e-16 ***
distance    -0.115027   0.008339  -13.79   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1013.43  on 1424  degrees of freedom
Residual deviance:  775.75  on 1423  degrees of freedom
AIC: 779.75

Number of Fisher Scoring iterations: 6
```

Example

Remember that R is an "object-oriented language" in the sense that every object in R has a class associated with it. The classes for mod.fit are:

```
> class(mod.fit)
[1] "glm" "lm"
```

Associated with each class, there are a number of "method" functions.

```
> methods(class = glm)
 [1] add1            anova           coerce          confint         cooks.distance deviance
 [7] drop1           effects         extractAIC      family          formula        influence
[13] initialize      logLik          model.frame     nobs            predict        print
[19] residuals       rstandard       rstudent        show            slotsFromS3    summary
[25] vcov            weights
see '?methods' for accessing help and source code
```

Example

Recall that the effect of an explanatory variable on the response probability depends on the specific value taken by the the explanatory variable.

In this particular example, the estimated probability of success for a particular distance using:

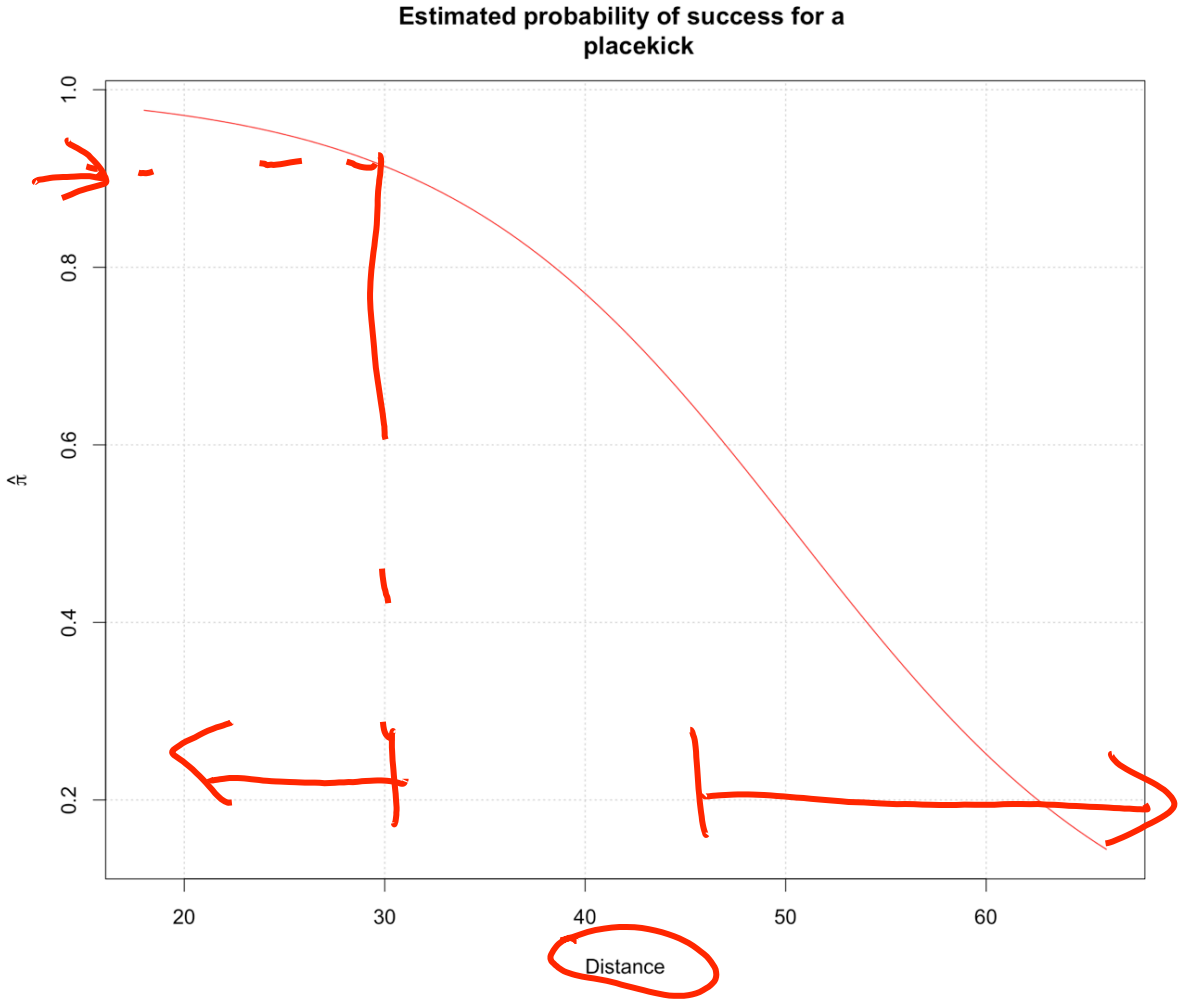
$$\hat{\pi} = \frac{e^{5.812 - 0.115 distance}}{1 + e^{5.812 - 0.115 distance}}$$

For example, the probability of success at a distance of 20 is 0.97. Likewise, the estimated probability of success for a distance of 50 yards is 0.52.

To get a perspective the above numbers, let's examine the distribution of distance in our sample:

```
summary(df$distance)
 Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
18.00   20.00   20.00   27.55   36.00   66.00
```

Example




```
curve(expr = exp(mod.fit$coefficients[1] + mod.fit$coefficients[2]*x) /
        (1 + exp(mod.fit$coefficients[1] + mod.fit$coefficients[2]*x)),
      col = "red", xlim = c(18, 66), ylab = expression(hat(pi)), xlab = "Distance",
    main = "Estimated probability of success for a
    placekick", panel.first = grid())
```

Example

If more than one explanatory variable is included in the model, the variable names can be separated by "+" symbols in the formula argument.

For example, suppose we include the change variable in addition to distance in the model:

```
mod.fit2<-glm(formula = good ~ change + distance, family= binomial(link = logit), data = df)
summary(mod.fit2)
```

```
Call:
glm(formula = good ~ change + distance, family = binomial(link = logit),
    data = df)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.7061   0.2282   0.2282   0.3750   1.5649

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  5.893181   0.333184  17.687   <2e-16 ***
change      -0.447783   0.193673  -2.312   0.0208 *
distance    -0.112889   0.008444 -13.370   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1013.4  on 1424  degrees of freedom
Residual deviance:  770.5  on 1422  degrees of freedom
AIC: 776.5

Number of Fisher Scoring iterations: 6
```

The estimated logistic regression model is

$$\widehat{logit}(\hat{\pi}) = 5.8932 - 0.4478change - 0.1129distance$$

Example

While we will use the glm() function extensively to find MLEs in this course, and this function is widely used in the industry, one could also find these estimates by programming the log likelihood function and maximizing it using another R function, optim().

This can come in handy for other "nonstandard" optimization problems that may occur in practice. We will not get into the details of programming the likelihood function manually. Please refer to the text for a simple illustration of how to program an MLE and optimize it manually.

Variance-Covariance Matrix of the Estimators

Variance-Covariance Matrix of the Estimators

The estimated variance-covariance matrix for $\hat{\beta}_0, \ldots, \hat{\beta}_p$ has the usual form:

$$\begin{bmatrix} \mathrm{Var}(\hat{\beta}_0) & \mathrm{Cov}(\hat{\beta}_0,\hat{\beta}_1) & \mathrm{Cov}(\hat{\beta}_0,\hat{\beta}_p) \\ \mathrm{Cov}(\hat{\beta}_0,\hat{\beta}_1) & \mathrm{Var}(\hat{\beta}_1) & \mathrm{Cov}(\hat{\beta}_1,\hat{\beta}_p) \\ \mathrm{Cov}(\hat{\beta}_0,\hat{\beta}_p) & \mathrm{Cov}(\hat{\beta}_1,\hat{\beta}_p) & \mathrm{Var}(\hat{\beta}_p) \end{bmatrix}$$

This matrix is found through using the same likelihood-based methods discussed in Lecture 1.

Variance-Covariance Matrix and MLE

If $\hat{\theta}$ is the MLE for θ, we can say that

$$\hat{\theta} \sim N\left(\theta, Var(\hat{\theta})\right)$$

for a large sample, where

$$Var(\hat{\theta}) = -\left[E\left(\frac{\partial^2 \log\left[L(\theta \mid Y_1, \ldots, Y_n)\right]}{\partial \theta^2}\right)\right]^{-1}\Bigg|_{\theta=\hat{\theta}}$$

Hessian Matrix

Note that we need to account for not only 1 MLE, but for p + 1 MLEs.

The result is a matrix of second partial derivatives (a Hessian matrix) of the log-likelihood function evaluated at the parameter estimates, and multiplying this resulting matrix by -1.

For example, if there are only parameters β_0 and β_1, the matrix is

$$-E\left(\begin{bmatrix} \frac{\partial^2 \log[L(\beta_0,\beta_1 \mid y_1,\ldots,y_n)]}{\partial \beta_0^2} & \frac{\partial^2 \log[L(\beta_0,\beta_1 \mid y_1,\ldots,y_n)]}{\partial \beta_0 \partial \beta_1} \\ \frac{\partial^2 \log[L(\beta_0,\beta_1 \mid y_1,\ldots,y_n)]}{\partial \beta_0 \partial \beta_1} & \frac{\partial^2 \log[L(\beta_0,\beta_1 \mid y_1,\ldots,y_n)]}{\partial \beta_1^2} \end{bmatrix}\right)^{-1}\Bigg|_{\beta_0=\hat{\beta}_0,\beta_1=\hat{\beta}_1}$$

The Sandwich Estimators

In the end, the matrix in general ends up having the nice sandwich form of $(X'VX)^{-1}$ where

$$X = \begin{bmatrix} 1 & x_{11} & x_{12} & x_{1p} \\ 1 & x_{21} & x_{22} & x_{2p} \\ & & & \\ 1 & x_{n1} & x_{n2} & x_{np} \end{bmatrix}$$

and $V = \text{Diag}(\hat{\pi}_i(1-\hat{\pi}_i))$ is a n×n matrix with $\hat{\pi}_i(1-\hat{\pi}_i)$ on the diagonal and 0's elsewhere.

Variance-Covariance Matrix in R

After estimating a model, the vcov() function can be used to produce the estimated variance-covariance matrix associated with the MLE:

```
> vcov(mod.fit2)
             (Intercept)        change      distance
(Intercept)  0.111011379 -0.0094878323 -2.625598e-03
change      -0.009487832  0.0375091687 -1.311512e-04
distance    -0.002625598 -0.0001311512  7.129494e-05
```

Hypothesis Tests for the Binomial Logistic Regression Parameters

Setting Up the Hypothesis

We often want to assess the importance of an explanatory variable or groups of explanatory variables. One way to make this assessment is through using hypothesis tests. For example, suppose we are interested in the r^{th} explanatory variable x_r in the model

$$\text{logit}(\pi) = \beta_0 + \beta_1 x_1 + \quad + \beta_r x_r + \quad + \beta_p x_p$$

If $\beta_r = 0$, we see that x_r would be excluded from the model. Thus, we are interested in hypothesis tests of the form:

Alternatively, we could state the hypotheses as:

$$H_0: \text{logit}(\pi) = \beta_0 + \beta_1 x_1 + \quad + \beta_{r-1} x_{r-1} + \beta_{r+1} x_{r+1} + \quad + \beta_p x_p$$

$$H_a: \text{logit}(\pi) = \beta_0 + \beta_1 x_1 + \quad + \beta_r x_r + \quad + \beta_p x_p$$

- The null hypothesis model terms are all included within the alternative hypothesis model: The null hypothesis model is a special case of the alternative hypothesis model.
- The null hypothesis model is often referred to as a reduced model, and the alternative hypothesis model is often referred to as a full model.

The Wald Test

The Wald statistic

$$Z_0 = \frac{\hat{\beta}_r - \beta_r}{\sqrt{\text{Var}(\hat{\beta}_r)}}$$

is used to test $H_0: \beta_r = 0$ vs. $H_a: \beta_r \neq 0$.

For a large sample, the test statistic has an approximate standard normal distribution if the null hypothesis of $\beta_r = 0$ is true. Thus, reject the null hypothesis if a test statistic value is "unusual" for a standard normal distribution, where "unusual" is defined to be $|Z_0| > Z_{1-\alpha/2}$.

The p-value is $2P(Z > |Z_0|)$ where $Z \sim N(0,1)$.

In R, the Wald test statistics and p-values are automatically provided for individual β parameters using code like *summary(mod.fit)*.

The Wald test can also be performed for more than one parameter at the same time. However, because the Wald test has similar problems to those we saw in Lecture 1 for Wald tests and confidence intervals, we are not going to pursue how to perform these types of tests.

Likelihood Ratio Test

Recall that the likelihood ratio test statistic is defined as

$$\Lambda = \frac{\text{Maximum of likelihood function under } H_0}{\text{Maximum of likelihood function under } H_0 \text{ or } H_a}$$

To perform a test of $H_0: \beta_r = 0$ vs. $H_a: \beta_r \neq 0$, we obtain the estimated probabilities of success from estimating

$$\text{logit}(\pi) = \beta_0 + \beta_1 x_1 + \quad + \beta_{r-1} x_{r-1} + \beta_{r+1} x_{r+1} + \quad + \beta_p x_p$$

(suppose the estimates are $\hat{\pi}_i^{(0)}$, $\hat{\beta}_0^{(0)}$, … $\hat{\beta}_p^{(0)}$) and the estimated probabilities of success from estimating

$$\text{logit}(\pi) = \beta_0 + \beta_1 x_1 + \quad + \beta_r x_r + \quad + \beta_p x_p$$

(suppose the estimates are $\hat{\pi}_i^{(a)}$, $\hat{\beta}_0^{(a)}$, … $\hat{\beta}_p^{(a)}$). We can then find

$$-2\log(\Lambda) = -2\log\left(\frac{L(\hat{\beta}^{(0)} \mid y_1,\ldots,y_n)}{L(\hat{\beta}^{(a)} \mid y_1,\ldots,y_n)}\right) = -2\left[\sum_{i=1}^{n} y_i \log\left(\frac{\hat{\pi}_i^{(0)}}{\hat{\pi}_i^{(a)}}\right) + (1-y_i)\log\left(\frac{1-\hat{\pi}_i^{(0)}}{1-\hat{\pi}_i^{(a)}}\right)\right]$$

If the null hypothesis was true, $-2\log(\Lambda)$ has an approximate χ_1^2 distribution for a large sample.

More General Hypotheses

The hypotheses can be generalized to allow for **q** different βs in H_0 to be set to 0.

Under the null hypothesis, $-2\log(\Lambda)$ has an approximate χ^2_q distribution for a large sample.

Example: Using the Wald Test

Recall from our previous example in which both *change* and *distance* are used.

```
mod.fit2<-glm(formula = good ~ change + distance, family = binomial(link = logit), data = placekick)
summary(mod.fit2)
```

```
Call:
glm(formula = good ~ change + distance, family = binomial(link = logit),
    data = placekick)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.7061   0.2282   0.2282   0.3750   1.5649

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  5.893181   0.333184  17.687   <2e-16 ***
change      -0.447783   0.193673  -2.312   0.0208 *
distance    -0.112889   0.008444 -13.370   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1013.4  on 1424  degrees of freedom
Residual deviance:  770.5  on 1422  degrees of freedom
AIC: 776.5

Number of Fisher Scoring iterations: 6
```

Below is a summary of the tests using $\alpha = 0.05$.

Change	**Distance**
H_0: $\beta_1 = 0$ H_a: $\beta_1 \neq 0$	H_0: $\beta_2 = 0$ H_a: $\beta_2 \neq 0$
$Z_0 = -2.31$	$Z_0 = -13.37$
p-value = 0.0208	p-value $< 2 \times 10^{-16}$
Reject H_0 because p-value is small	Reject H_0 because p-value is small
There is sufficient evidence to indicate change has an effect on the probability of success given distance is in the model.	There is sufficient evidence to indicate distance has an effect on the probability of success given change is in the model.

Example: Using the Likelihood Ratio Test

- There are a number of ways to perform LRTs in R.
- The easiest way is to use the **Anova() function** from the **car package**, authored by Professor John Fox.

```
> library(car)
> Anova(mod = mod.fit2, test = "LR")
Analysis of Deviance Table (Type II tests)

Response: good
          LR Chisq Df Pr(>Chisq)
change      5.246  1      0.022 *
distance  218.650  1     <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Difference in p-values

- The p-value for distance is the same as from using Anova(), but the p-value for change is not.
- The reason is due to the hypotheses being tested.
  - The Anova() function tests the model's explanatory variables in an sequential manner. Thus, the change test p-value is actually for the test of

$$H_0: \text{logit}(\pi) = \beta_0$$
$$H_a: \text{logit}(\pi) = \beta_0 + \beta_1 \text{change}$$

because it is listed first in the formula argument of glm(). The distance variable is listed second so anova() tests:

$$H_0: \text{logit}(\pi) = \beta_0 + \beta_1 \text{change}$$
$$H_a: \text{logit}(\pi) = \beta_0 + \beta_1 \text{change} + \beta_2 \text{distance}$$

where change is assumed to be in both models.

Produce the Tests Similar to Anova()

In order to produce the tests like Anova(), estimate the H_0 and H_a models separately and then use their model fit objects in a different way with Anova():

```
> mod.fit2<-glm(formula = good ~ change + distance, family = binomial(link = logit), data = df)
> mod.fit.Ho<-glm(formula = good ~ distance, family = binomial(link = logit), data = df)
> anova(mod.fit.Ho, mod.fit2, test = "Chisq")
Analysis of Deviance Table

Model 1: good ~ distance
Model 2: good ~ change + distance
  Resid. Df Resid. Dev Df Deviance Pr(>Chi)
1      1423     775.75
2      1422     770.50  1   5.2455    0.022 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

- This use of Anova() helps to emphasize a reduced- and full-model approach to obtaining the $-2\log(\Lambda)$ statistic.
- This approach is helpful to know how to do when Anova() may not be available (with more complex models than logistic regression) or when more than one variable is being tested at a time.

The Notions of Deviance

Deviance, Residual Deviance, Null Deviance

- Deviance refers to the amount that a particular model deviates from another model as measured by **$-2\log(\Lambda)$**.

- Example: The **$-2\log(\Lambda) = 5.246$** value used for testing the change variable in the last example (given distance is in the model) is a measure of **how much the estimated probabilities of success for the null hypothesis model deviate from the estimated probabilities of success for the alternative hypothesis model**.

- Residual deviance denotes how much the null hypothesis model deviates from using the observed proportion of successes for each "observation" (y_i = 0 or 1 for Bernoulli response data or w_j/n_j for binomial response data) to estimate π (π_i or π_j).

Example

Deviance statistics are often calculated as an intermediary step for performing a LRT to compare two models. For example, consider testing:

$$H_0: \text{logit}(\pi^{(0)}) = \beta_0^{(0)} + \beta_1^{(0)} x_1$$

$$H_a: \text{logit}(\pi^{(a)}) = \beta_0^{(a)} + \beta_1^{(a)} x_1 + \beta_2^{(a)} x_2 + \beta_3^{(a)} x_3$$

The residual deviance for the model $\text{logit}(\pi^{(0)}) = \beta_0 + \beta_1 x_1$ tests

$$H_0: \text{logit}(\pi^{(0)}) = \beta_0^{(0)} + \beta_1^{(0)} x_1$$

H_a: Saturated model

with

$$-2\log(\Lambda) = -2\left[\sum_{i=1}^{n} y_i \log\left(\frac{\hat{\pi}_i^{(0)}}{y_i}\right) + (1-y_i)\log\left(\frac{1-\hat{\pi}_i^{(0)}}{1-y_i}\right)\right] \quad (2)$$

Written in terms of binomial response data, we would have:

$$-2\log(\Lambda) = -2\left[\sum_{j=1}^{J} w_j \log\left(\frac{\hat{\pi}_j^{(0)}}{w_j / n_j}\right) + (n_j - w_j)\log\left(\frac{1-\hat{\pi}_j^{(0)}}{1-w_j/n_j}\right)\right]$$

The residual deviance for the model

$$\text{logit}(\pi^{(a)}) = \beta_0^{(a)} + \beta_1^{(a)}x_1 + \beta_2^{(a)}x_2 + \beta_3^{(a)}x_3$$

tests

$$H_0: \text{logit}(\pi^{(a)}) = \beta_0^{(a)} + \beta_1^{(a)}x_1 + \beta_2^{(a)}x_2 + \beta_3^{(a)}x_3$$

H_a: Saturated model

with

$$-2\log(\Lambda) = -2\left[\sum_{i=1}^{n} y_i \log\left(\frac{\hat{\pi}_i^{(a)}}{y_i}\right) + (1-y_i)\log\left(\frac{1-\hat{\pi}_i^{(a)}}{1-y_i}\right)\right] \quad (3)$$

By finding (2) – (3) we obtain Equation (1); thus, we obtain the desired $-2\log(\Lambda)$ by subtracting the residual deviances of the models in our original hypotheses. The degrees of freedom for the test can be found from subtracting the corresponding degrees of freedom for (3) from (2).

Example in R

Below is the R code demonstrating the above discussion with respect to

$$H_0: \text{logit}(\pi) = \beta_0 + \beta_1 \text{distance}$$

$$H_a: \text{logit}(\pi) = \beta_0 + \beta_1 \text{distance} + \beta_2 \text{change}$$

```
mod.fit.Ho<-glm(formula = good ~ distance, family =
    binomial(link = logit), data = df)

dframe<-mod.fit.Ho$df.residual-mod.fit2$df.residual

stat<-mod.fit.Ho$deviance-mod.fit2$deviance
pvalue<-1-pchisq(q = stat, df = dframe)
data.frame(Ho.resid.dev = mod.fit.Ho$deviance,
    Ha.resid.dev = mod.fit2$deviance, df = dframe, stat =
    round(stat,4), pvalue = round(pvalue,4))
```

	Ho.resid.dev	Ha.resid.dev	df	stat	pvalue
1	775.745	770.4995	1	5.2455	0.022

Odds Ratios

Odds Ratios

Recall that a logistic regression model can be written as

$$log(\frac{\pi}{1-\pi}) = \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p$$

where the left-side of the model is the log odds of a success.

Using a similar interpretation as the classical linear regression models, we can use β_r to interpret the effect that x_r has on the log odds of a success.

We can then form odds ratios by looking at these odds of success at different values of x_r.

For ease of presentation, consider the logistic regression with only one explanatory variable x:

$$\log\left(\frac{\pi}{1-\pi}\right) = \beta_0 + \beta_1 x$$

Or, rewrite this model as

$$\text{Odds}_x = e^{\beta_0 + \beta_1 x}$$

If we increase x by c-units, the odds of a success becomes

$$\text{Odds}_{x+c} = e^{\beta_0 + \beta_1 (x+c)}$$

Interpretation

To interpret the effect of increasing x by c-units, we can form an odds ratio:

$$OR = \frac{Odds_{x+c}}{Odds_x} = \frac{e^{\beta_0 + \beta_1(x+c)}}{e^{\beta_0 + \beta_1 x}} = e^{c\beta_1}$$

Observe the beauty of the exponential functions.

➔ The x drops out!

- Regardless of the value of x, the estimated odds of a success change by the same amount for every c-unit increase in x.
- This is one of the main reasons why logistic regression is so popular for modeling binary response data.

Discrete Response Model
Lecture 2

datascience@berkeley

Odds Ratios

Interpretation (cont.)

There are a number of ways to interpret the odds ratio in the context of logistic regression. The one following is commonly used.

The odds of a success change by $e^{c\beta_1}$ times for every c-unit increase in x

If x is a binary (or indicator) explanatory variable having only two levels coded as 0 or 1, then

$$Odds_{x=0} = e^{\beta_0 + \beta_1 0} = e^{\beta_0} \quad \text{and} \quad Odds_{x=1} = e^{\beta_0 + \beta_1}$$

$$OR = \frac{e^{\beta_0 + \beta_1}}{e^{\beta_0}} = e^{\beta_1}$$

The odds of a success are e^{β_1} times as large for x = 1 than for x = 0

Estimated Odds Ratio

To find the estimated odds ratio, simply replace the parameter with its corresponding estimate:

$$OR = e^{c\hat{\beta}_1}$$

- The interpretation of the odds ratio now needs to be qualified with an "estimated" in the appropriate location in the sentence.
- This estimate is the MLE.

Wald Confidence Interval for OR

Wald confidence intervals are the easiest to calculate. First, an interval for $c\beta_1$ needs to be found:

$$c\hat{\beta}_1 \pm cZ_{1-\alpha/2}\sqrt{Var(\hat{\beta}_1)}$$

where $Var(\hat{\beta}_1)$ is obtained from the estimated covariance matrix for the parameter estimates. Notice where c is located in the interval calculation. The second c comes about through $Var(c\hat{\beta}_1) = c^2 Var(\hat{\beta}_1)$.

To find the $(1 - \alpha)$ Wald confidence interval for OR, use the exponential function:

$$e^{c\hat{\beta}_1 \pm cZ_{1-\alpha/2}\sqrt{Var(\hat{\beta}_1)}}$$

Profile Likelihood Ratio Confidence Interval

- As you might expect, Wald confidence intervals do not always work as well as we would like for smaller sample sizes.
- Instead, a better interval is a profile likelihood ratio interval.
- For a $(1 - \alpha)100\%$ interval, we find the set of β_1 values such that

$$-2\log\left(\frac{L(\beta_0,\beta_1 \mid y_1,\ldots,y_n)}{L(\hat{\beta}_0,\hat{\beta}_1 \mid y_1,\ldots,y_n)}\right) < \chi^2_{1,1-\alpha}$$

  is satisfied.
- On the left side, we have the usual $-2\log(\Lambda)$ form, but without a specified value of β_1.
- The $\hat{\beta}_0$ is an estimate of β_0 for a fixed value of β_1. Iterative numerical procedures can be used to find the β_1 values that satisfy the above equation.
- The $(1 - \alpha)100\%$ interval for OR is then

$$e^{c \times lower} < OR < e^{c \times upper}$$

  using the lower and upper limits found for β_1 in the above equation.

Some Remarks

1) Inverting odds ratios less than 1 is helpful for interpretation purposes.
2) An appropriate value of c should be chosen in the context of the explanatory variable. For example, if $0.1 < x < 0.2$, a value of $c = 1$ would not be appropriate. Additionally, if $0 < x < 1000$, a value of $c = 1$ may not be appropriate as well.
3) When there is more than one explanatory variable, the same interpretation of the odds ratio *generally* can be made with the addition of "holding the other explanatory variables constant" added.
4) If the specification includes transformations of the explanatory variables, the odds ratio is not simply $e^{c\beta r}$ as given previously, because the odds ratio is no longer constant for every c-unit increase in x.
5) A categorical explanatory variable represented by multiple indicator variables does not have the same type of interpretation as given previously.

Example

Consider the model with only distance as the explanatory variable:

$$logit(\hat{\pi}) = 5.8121 - 0.150 distance$$

To estimate the odds ratio, we can simply use the exp() function:

```
> exp(mod.fit$coefficients[2])
 distance
0.8913424
> 1/exp(10*mod.fit$coefficients[2])
distance
3.159035
```

The first odds ratio is for a 1-yard (c = 1) increase in distance. This is not very meaningful in the current context. Instead, c = 10 would be more meaningful because 10 yards are needed for a first down in football.

Also, it is more meaningful to look at a 10 yard decrease (another first down) rather than a 10 yard increase. Therefore, the estimated odds of a success change by $\frac{1}{e^{10(-0.1150)}} = 3.16$ times for every 10 yard decrease in the distance of the placekick.

- Note that the 3.16 holds for comparing 30- to 20-yard placekicks as well as 55- to 45-yard placekicks or any other 10-yard decrease.

Example

To account for the variability in the odds ratio estimator, we would like to calculate a confidence interval for the actual odds ratio itself. Below is the code for the profile likelihood ratio interval:

```
> beta.ci<-confint(object = mod.fit, parm = "distance", level = 0.95)
Waiting for profiling to be done...
> beta.ci
      2.5 %      97.5 %
-0.13181435 -0.09907103
> rev(1/exp(beta.ci*10)) #Invert OR C.I. and c=10
  97.5 %    2.5 %
2.693147 3.736478
> as.numeric(rev(1/exp(beta.ci*10)))
[1] 2.693147 3.736478
```

- The **confint() function** first finds an interval for β_1 itself.
- We then use the **exp() function** to find the confidence interval for OR.
- The 95% profile likelihood ratio confidence interval is $2.69 < OR < 3.74$.
- Pay special attention to how this was found with the rev() function and the **beta.ci** object.
- Because the interval is entirely above 1, there is sufficient evidence that a 10-yard decrease in distance increases the odds of a successful placekick.

Probability of Success and the Corresponding Confidence Intervals

Probability of Success

As shown earlier, the estimate for π is

$$\hat{\pi} = \frac{e^{\hat{\beta}_0 + \hat{\beta}_1 x_1 + \quad + \hat{\beta}_p x_p}}{1 + e^{\hat{\beta}_0 + \hat{\beta}_1 x_1 + \quad + \hat{\beta}_p x_p}}$$

To find a confidence interval for π, consider again the logistic regression model with only one explanatory variable x:

$$\log\left(\frac{\pi}{1-\pi}\right) = \beta_0 + \beta_1 x \quad \text{or} \quad \pi = \frac{e^{\beta_0 + \beta_1 x}}{1 + e^{\beta_0 + \beta_1 x}}$$

Wald Confidence Interval

To find a Wald confidence interval for π, we need to first find an interval for $\beta_0 + \beta_1 x$ (or equivalently for logit(π)):

$$\hat{\beta}_0 + \hat{\beta}_1 x \pm Z_{1-\alpha/2} \sqrt{Var(\hat{\beta}_0 + \hat{\beta}_1 x)}$$

where

$$Var(\hat{\beta}_0 + \hat{\beta}_1 x) = Var(\hat{\beta}_0) + x^2 Var(\hat{\beta}_1) + 2x Cov(\hat{\beta}_0, \hat{\beta}_1)$$

and $Var(\hat{\beta}_0)$, $Var(\hat{\beta}_1)$, and $Cov(\hat{\beta}_0, \hat{\beta}_1)$ are obtained from the estimated covariance matrix for the parameter estimates.

To find the $(1 - \alpha)100\%$ Wald confidence interval for π, we use the $\exp(\cdot)/[1+\exp(\cdot)]$ transformation:

$$\frac{e^{\hat{\beta}_0 + \hat{\beta}_1 x \pm Z_{1-\alpha/2} \sqrt{Var(\hat{\beta}_0 + \hat{\beta}_1 x)}}}{1 + e^{\hat{\beta}_0 + \hat{\beta}_1 x \pm Z_{1-\alpha/2} \sqrt{Var(\hat{\beta}_0 + \hat{\beta}_1 x)}}}$$

Wald Confidence Interval (cont.)

For a model with p explanatory variables, the interval is

$$\frac{e^{\hat{\beta}_0+\hat{\beta}_1 x_1+\quad+\hat{\beta}_p x_p \pm Z_{1-\alpha/2}\sqrt{Var(\hat{\beta}_0+\hat{\beta}_1 x_1+\quad+\hat{\beta}_p x_p)}}}{1+e^{\hat{\beta}_0+\hat{\beta}_1 x_1+\quad+\hat{\beta}_p x_p \pm Z_{1-\alpha/2}\sqrt{Var(\hat{\beta}_0+\hat{\beta}_1 x_1+\quad+\hat{\beta}_p x_p)}}}$$

where

$$Var(\hat{\beta}_0+\hat{\beta}_1 x_1+\quad+\hat{\beta}_p x_p)=\sum_{i=0}^{p} x_i^2 Var(\hat{\beta}_i)+2\sum_{i=0}^{p-1}\sum_{j=i+1}^{p} x_i x_j Cov(\hat{\beta}_i,\hat{\beta}_j)$$

and $x_0 = 1$. Verify on your own that the interval given for p explanatory variables is the same as the original interval given for p = 1 explanatory variable.

Profile Likelihood Ratio Interval

Profile LR confidence intervals for π can be found as well, but they can be much more difficult computationally to find than for OR. This is because a larger number of parameters are involved.

For example, the one explanatory variable model $\text{logit}(\pi) = \beta_0 + \beta_1 x$ is a linear combination of β_0 and β_1. The numerator of $-2\log(\Lambda)$ involves maximizing the likelihood function with a constraint for this linear combination.

- The **mcprofile package** provides a general way to compute profile likelihood ratio intervals.
  - Earlier versions sometimes produced questionable results; current versions generally do not have problems.

Recommend using the following approach with this package:

1. Calculate a Wald interval.
2. Calculate a profile likelihood ratio interval with the **mcprofile** package.
3. Use the profile LR interval as long as it is not outlandishly different than the Wald and there are no warning messages given by R when calculating the interval. Otherwise, use the Wald interval.

Example

Consider the model with only distance as the explanatory variable:

$$\text{logit}(\hat{\pi}) = 5.8121 - 0.1150\text{distance}$$

where the results from glm() are saved in the object mod.fit.

Wald Interval

Estimate the probability of success for a distance of 20 yards:

```
> linear.pred<-mod.fit$coefficients[1] +
+     mod.fit$coefficients[2]*20
> linear.pred
(Intercept)
   3.511547
> exp(linear.pred)/(1+exp(linear.pred))
(Intercept)
  0.9710145
```

Use the predict() function

```
> predict.data<-data.frame(distance = 20)
> predict(object = mod.fit, newdata = predict.data, type = "link")
       1
3.511547
> predict(object = mod.fit, newdata = predict.data, type = "response")
       1
0.9710145
```

Example (cont.)

Note that the predict() function is a generic function, so predict.glm() is actually used to perform the calculations. Also, notice the argument value of type = "link" calculates $\hat{\beta}_0 + \hat{\beta}_1 x$ (equivalently, $\text{logit}(\hat{\pi})$).

To find the Wald confidence interval, we can calculate components of the interval for $\beta_0 + \beta_1 x$ through adding arguments to the predict() function:

```
> linear.pred<-predict(object = mod.fit, newdata =
+     predict.data, type = "link", se = TRUE)
> linear.pred
$fit
       1
3.511547

$se.fit
[1] 0.1732707

$residual.scale
[1] 1

> pi.hat<-exp(linear.pred$fit) / (1 + exp(linear.pred$fit))
> CI.lin.pred<-linear.pred$fit + qnorm(p = c(alpha/2, 1- alpha/2))*linear.pred$se
> CI.pi<-exp(CI.lin.pred)/(1+exp(CI.lin.pred))
> CI.pi
[1] 0.9597647 0.9791871
> data.frame(predict.data, pi.hat, lower = CI.pi[1], upper = CI.pi[2])
  distance    pi.hat     lower     upper
1       20 0.9710145 0.9597647 0.9791871
```

The 95% Wald confidence interval for π is $0.9598 < \pi < 0.9792$; thus, the probability of success for the placekick is quite high at a distance of 20 yards.

Example: A More General Case

Using the original Wald confidence interval equation again, we can also calculate more than one interval at a time and include more than one explanatory variable. Below is an example using the estimated model.

$$\text{logit}(\hat{\pi}) = 5.8932 - 0.4478\text{change} - 0.1129\text{distance}$$

```
> alpha<-0.05
> linear.pred<-predict(object = mod.fit2, newdata =
+     predict.data, type = "link", se = TRUE)
> CI.lin.pred.x20<-linear.pred$fit[1] + qnorm(p =
+     c(alpha/2, 1-alpha/2)) * linear.pred$se[1]
> CI.lin.pred.x30<-linear.pred$fit[2] + qnorm(p =
+     c(alpha/2, 1-alpha/2)) * linear.pred$se[2]
> round(exp(CI.lin.pred.x20)/(1+exp(CI.lin.pred.x20)), 4)  #CI for distance = 20
[1] 0.9404 0.9738
> round(exp(CI.lin.pred.x30)/(1+exp(CI.lin.pred.x30)), 4) #CI for distance = 30
[1] 0.8493 0.9159
```

Example: Profile Likelihood Ratio Interval

```
> library(mcprofile)
Loading required package: ggplot2
Need help? Try the ggplot2 mailing list: http://groups.google.com/group/ggplot2.
Warning message:
package 'ggplot2' was built under R version 3.2.4
> K<-matrix(data = c(1, 20), nrow = 1, ncol = 2)
> linear.combo<-mcprofile(object = mod.fit, CM = K)  #Calculate -2log(Lambda)
> ci.logit.profile<-confint(object = linear.combo, level = 0.95) #CI for beta_0 + beta_1 * x
> ci.logit.profile

   mcprofile - Confidence Intervals

level:           0.95
adjustment:      single-step

   Estimate lower upper
C1     3.51  3.19  3.87

> names(ci.logit.profile)
[1] "estimate"    "confint"     "CM"          "quant"       "alternative" "level"       "adjust"
> exp(ci.logit.profile$confint)/(1 + exp(ci.logit.profile$confint))
      lower    upper
1 0.9603165 0.979504
```

The 95% interval for π is $0.9603 < \pi < 0.9795$, which is similar to the Wald interval due to the large sample size.

Visual Assessment of the Logistic Regression Model

Visual Assessment of the Logistic Regression Model With One Explanatory Variable

Because the response variable in logistic regression is binary, constructing a scatterplot with the raw binary response variable is not very informative because all plotted points would be at y = 0 or 1 on the y-axis.

Instead, we can plot the observed proportion of successes at each x instead to obtain a general understanding of how well the model fits the data.

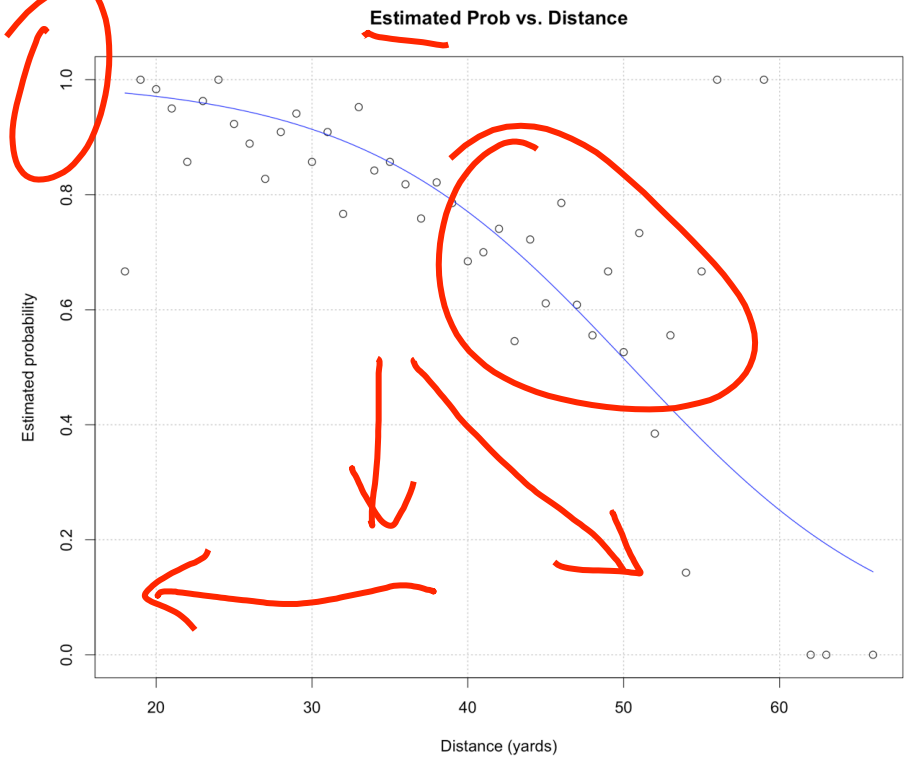
Note: This works well only when the number of observations at each possible x is not small. For truly continuous x, the number of observations would be 1 and this plot would not be very useful. An alternative for this situation include grouping observations by x and finding the observed proportion of successes for each group; however, this leads to potentially different results depending on how the grouping is done.

Example

```
> w<-aggregate(formula = good ~ distance, data = placekick, FUN = sum)
> n<-aggregate(formula = good ~ distance, data = placekick, FUN = length)
> w.n<-data.frame(distance = w$distance, success = w$good,
+     trials = n$good, proportion = round(w$good/n$good,4))
> head(w.n)
  distance success trials proportion
1       18       2      3     0.6667
2       19       7      7     1.0000
3       20     776    789     0.9835
4       21      19     20     0.9500
5       22      12     14     0.8571
6       23      26     27     0.9630
```

This was used to estimate a logistic regression model using a binomial response form of the data. Instead, we can plot the observed proportion of successes at each distance and overlay the estimated logistic regression model.

Aggregated Scatterplot of Estimated Probability vs. Distance

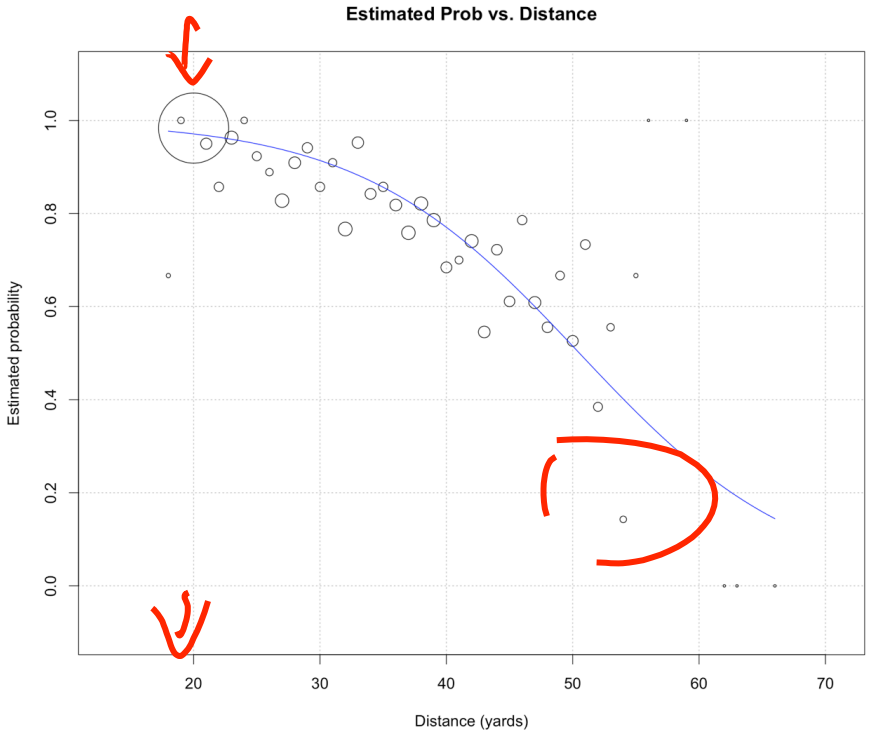



```
w<-aggregate(formula = good ~ distance, data = placekick, FUN = sum)
n<-aggregate(formula = good ~ distance, data = placekick, FUN = length)
w.n<-data.frame(distance = w$distance, success = w$good,
    trials = n$good, proportion = round(w$good/n$good,4))
head(w.n)

win.graph(width = 7, height = 6, pointsize = 12)
plot(x = w$distance, y = w$good/n$good, main="Estimated Prob vs. Distance",
    xlab="Distance (yards)", ylab="Estimated probability",
    panel.first = grid(col = "gray", lty = "dotted"))
```

Bubble Plot

To include a measure of how many observations are at each distance, we can use a bubble plot. For this plot, we make the plotting point size proportional to the observed number of observations at each unique distance.




```
win.graph(width = 7, height = 6, pointsize = 12)
symbols(x = w$distance, y = w$good/n$good, circles =
    sqrt(n$good), inches = 0.5,  main="Estimated Prob vs. Distance",
    xlab="Distance (yards)",
    ylab="Estimated probability", panel.first = grid(col =
    "gray", lty = "dotted"))
curve(expr = predict(object = mod.fit, newdata =
    data.frame(distance = x), type = "response"), col =
    "blue", add = TRUE, xlim = c(18, 66))
```

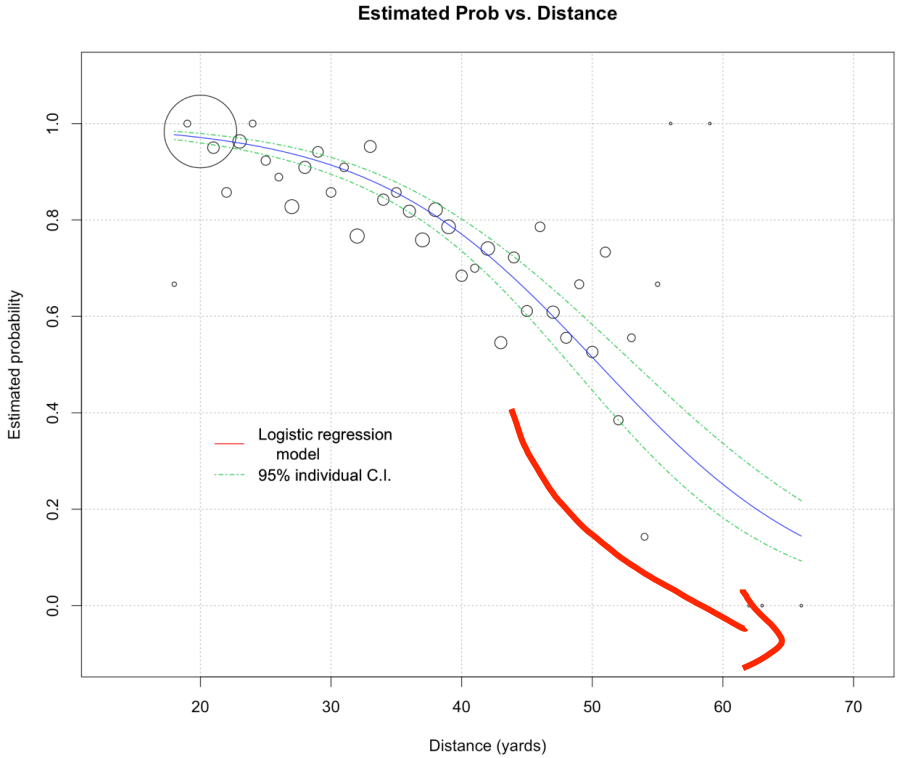
Further Remarks

Points that may have caused us concern before, now generally do not because we see they represent a small number of observations. For example,

- 18-yard placekicks: Only 3 observations occurred and there were 2 successes.
- The largest in distance placekicks: Many of these correspond to 1 observation only.

However, there are a few large plotting points, such as at 32 yards ($\hat{\pi}$ = 0.89, observed proportion = 23/30 = 0.77) and 51 yards ($\hat{\pi}$ = 0.49, observed proportion = 11/15 = 0.73), that may not be fit well by the model. How to more formally assess these observations and others will be an important subject of Week 5 when we examine model diagnostic measures.

Adding Confidence Bands to the Plot




```
# Add confidence bands to the previous plot
curve(expr = ci.pi(newdata = data.frame(distance = x),
    mod.fit.obj = mod.fit, alpha = 0.05)$lower, col =
    "green", lty = "dotdash", add = TRUE, xlim = c(18, 66))
curve(expr = ci.pi(newdata = data.frame(distance = x),
    mod.fit.obj = mod.fit, alpha = 0.05)$upper, col =
    "green", lty = "dotdash", add = TRUE, xlim = c(18, 66))
legend(x = 20, y = 0.4, legend = c("Logistic regression
    model", "95% individual C.I."), lty = c("solid",
    "dotdash"), col = c("red", "green"), bty = "n")
```

```
ci.pi<-function(newdata, mod.fit.obj, alpha){
   linear.pred<-predict(object = mod.fit.obj, newdata =
     newdata, type = "link", se = TRUE)
   CI.lin.pred.lower<-linear.pred$fit - qnorm(p = 1-
     alpha/2)*linear.pred$se
   CI.lin.pred.upper<-linear.pred$fit + qnorm(p = 1-
     alpha/2)*linear.pred$se
   CI.pi.lower<-exp(CI.lin.pred.lower) / (1 +
     exp(CI.lin.pred.lower))
   CI.pi.upper<-exp(CI.lin.pred.upper) / (1 +
     exp(CI.lin.pred.upper))
   list(lower = CI.pi.lower, upper = CI.pi.upper)
}
```

Berkeley SCHOOL OF INFORMATION