Unit2

K Iwasaki

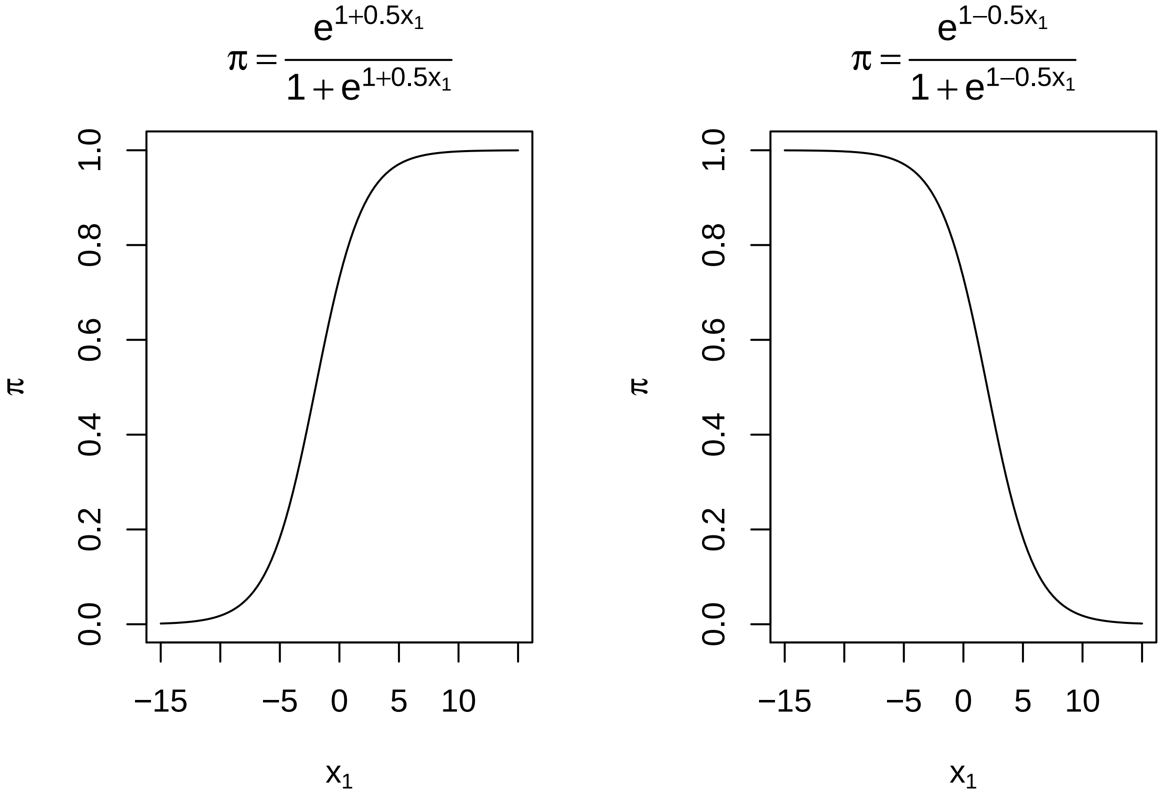
September 13, 2017

Plot the logistic regression model

```
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))

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))
```



$$\pi = \frac{e^{1+0.5x_1}}{1+e^{1+0.5x_1}}$$

$$\pi = \frac{e^{1-0.5x_1}}{1+e^{1-0.5x_1}}$$

Example

```r
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 ...
```

```r
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
```

```r
# check NA in each column
apply(is.na(df), 2, sum)
```

```
##     week distance    change    elap30       PAT      type     field      wind
##        0        0         0         0         0         0         0         0
##     good
##        0
```

```r
# check the dependent variable of interest
table(df$good)
```

```
##
##    0    1
##  163 1262
```

```r
prop.table(table(df$good))
```

```
##
##        0        1
## 0.114386 0.885614
```

use distance explanatory variable to estimate the probability of a successful placekick

```r
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
```

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

```
# there are many information stored within the mod.fit object
names(mod.fit)
```

```
##  [1] "coefficients"      "residuals"         "fitted.values"
##  [4] "effects"           "R"                 "rank"
##  [7] "qr"                "family"            "linear.predictors"
## [10] "deviance"          "aic"               "null.deviance"
## [13] "iter"              "weights"           "prior.weights"
## [16] "df.residual"       "df.null"           "y"
## [19] "converged"         "boundary"          "model"
## [22] "call"              "formula"           "terms"
## [25] "data"              "offset"            "control"
## [28] "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
```

```
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
```

```r
class(mod.fit)
```

```
## [1] "glm" "lm"
```

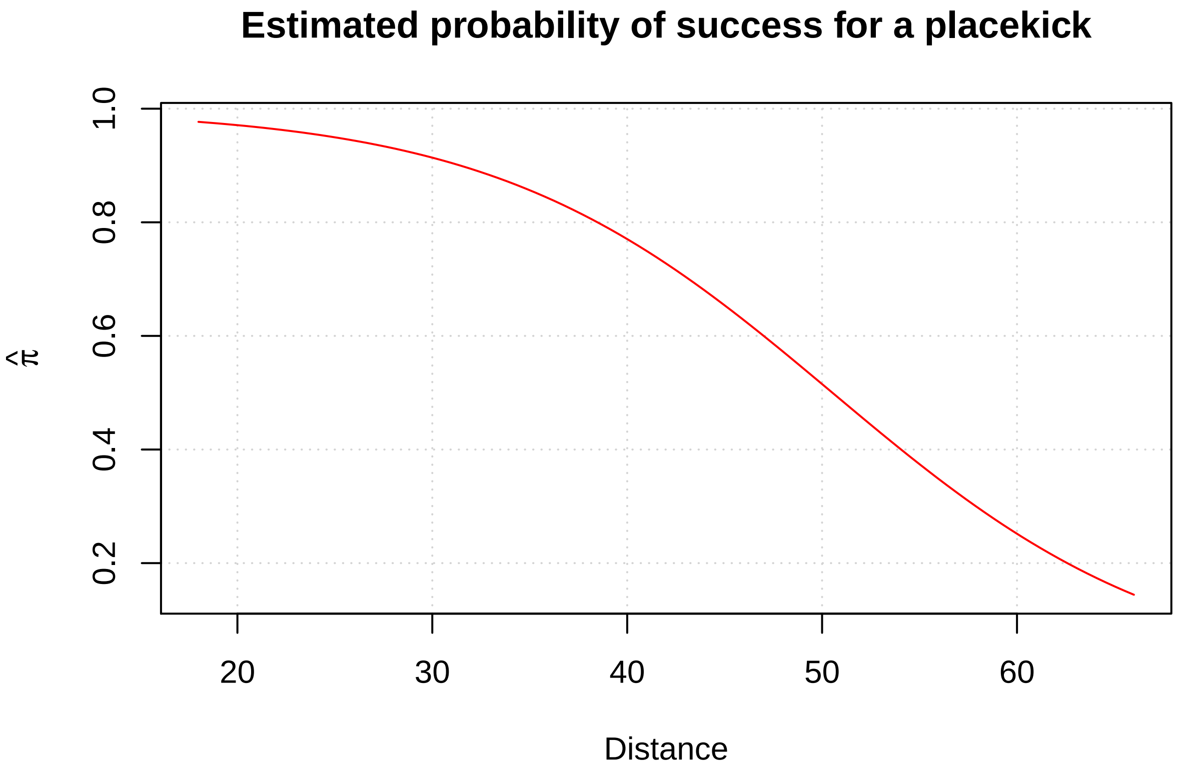
```r
methods(class = glm)
```

```
##  [1] add1           anova          coerce         confint
##  [5] cooks.distance deviance       drop1          effects
##  [9] extractAIC     family         formula        influence
## [13] initialize     logLik         model.frame    nobs
## [17] predict        print          residuals      rstandard
## [21] rstudent       show           slotsFromS3    summary
## [25] vcov           weights
## see '?methods' for accessing help and source code
```

```r
summary(df$distance)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   18.00   20.00   20.00   27.55   36.00   66.00
```

```r
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()
        )
```



```r
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
```

```
mod.fit2$coefficients
```

```
## (Intercept)      change    distance
##   5.8931814  -0.4477832  -0.1128888
```

It takes 6 iterations to come up with the parameters

```
newdata = data.frame(change = 0.5, distance = 50)
predict(mod.fit2, newdata, type="response")
```

```
##         1
## 0.5062124
```

```
b0 = mod.fit2$coefficients[1]
b1 = mod.fit2$coefficients[2]
b2 = mod.fit2$coefficients[3]

summary(df$change)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0000  0.0000  0.0000  0.2519  1.0000  1.0000
```

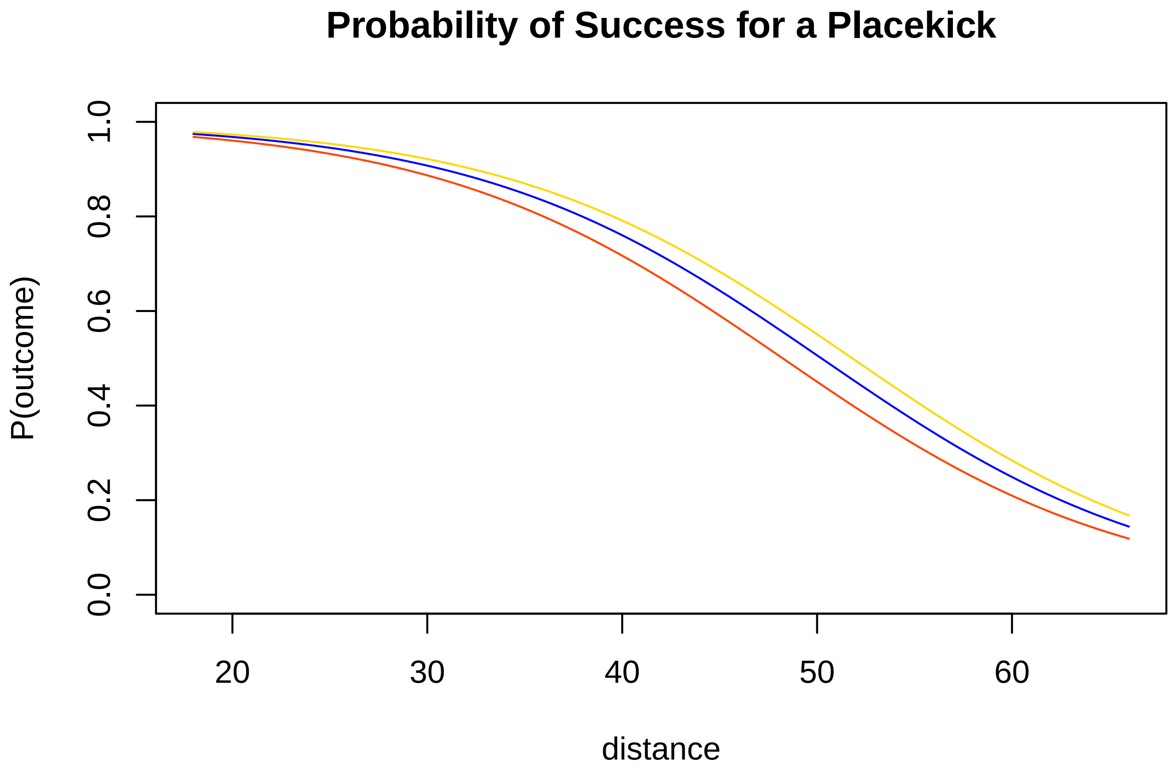
```
x_range = seq(from = min(df$distance), to=max(df$distance), by = .1)

curve1 = exp(b0 + b1*0.5 + b2*x_range) /
  (1 + exp(b0 + b1*0.5 + b2*x_range))

curve2 = exp(b0 + b1*0.1 + b2*x_range) /
  (1 + exp(b0 + b1*0.1 + b2*x_range))

curve3 = exp(b0 + b1*1 + b2*x_range) /
  (1 + exp(b0 + b1*1 + b2*x_range))

plot(x_range, curve1, ylim = c(0, 1), col = "blue", type= "l", xlab = "distance",
     ylab = "P(outcome)", main = "Probability of Success for a Placekick")
lines(x_range, curve2, col = "gold", type= "l")
lines(x_range, curve3, col = "orangered", type= "l")
```



The blue line is with change = 0.5, the red line 1.0 and the yellow line 0.1. The line with smaller change value has higher probability of sucess.

Variance - Covariance Matrix

```
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
```

```
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
```

```
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
```

when Anova() is not available with more complex models than logistic regression

```
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") # note "a" nova, not Anova
```

```
## 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
```

General approach for log-likelihood function

check page 71 textbook

```
logL = function(beta, x, Y) {
  pi = exp(beta[1] + beta[2] *x) / ( 1 + exp(beta[1] + beta[2] *x))
  sum(Y*log(pi) + (1-Y)*log(1-pi))
}

logL(beta = mod.fit$coefficients, x = df$distance, Y = df$good)
```

```
## [1] -387.8725
```

```
# find starting values for parameter estimate
reg.mod = lm(formula = good ~ distance, data = df)
reg.mod$coefficients
```

```
## (Intercept)    distance
##  1.25202444 -0.01330212
```

```
# use control = list(fnscale = -1) to "maximize" instead of minimize
mod.fit.optim = optim(par = reg.mod$coefficients, fn = logL,
                      hessian = TRUE, x = df$distance, Y = df$good, control = list(fnscale = -1),
                      method = "BFGS")
names(mod.fit.optim)
```

```
## [1] "par"          "value"        "counts"       "convergence" "message"
## [6] "hessian"
```

```
mod.fit.optim$par
```

```
## (Intercept)    distance
##   5.8112544  -0.1150046
```

```
mod.fit.optim$convergence
```

```
## [1] 0
```

```
-solve(mod.fit.optim$hessian)
```

```
##              (Intercept)      distance
## (Intercept)  0.106482867 -2.607258e-03
## distance    -0.002607258  6.957463e-05
```

Deviance

```
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)

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.fit2$deviance, Ha.resid.dev = mod.fit2$deviance,
           df = dframe, stat = round(stat, 4),
           pvalue = round(pvalue, 4))
```

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