y $\quad x_1 \cdots x_p$

0
1
0
0
0
1
1
1
0
1
0
$\vdots$

STAT 516 Lec 12

Logistic regression

Karl Gregory

2024-04-15

Programming task data from Kutner et al. (2005)

Twenty-five people succeeded or failed at a programming task.

Months of programming experience was recorded for each person.

```
experience <- c(14,29,6,25,18,4,18,12,22,6,30,11,30,5,20,13,9,32,24,13,19,4,28,22,8)
success <- c(0,0,0,1,1,0,0,0,1,0,1,0,1,0,1,0,0,1,0,1,0,0,1,1,1)
```

Can we predict probability of success based on experience?

(Simple) Linear regression: $Y_i = \beta_0 + \beta_1 x_i + \varepsilon_i$

Logistic regression model

$P(Y_i = 1) = \pi_i$

$P(Y_i = 0) = 1 - \pi_i$

$Y_i \in \{0, 1\}$

Assume

$$\boxed{Y_i \sim \text{Bernoulli}(\pi_i),} \quad \log\left(\underbrace{\frac{\pi_i}{1-\pi_i}}_{}\right) = \beta_0 + \beta_1 x_i,$$

$\frac{P(\text{success})}{P(\text{failure})}$ = "odds" of success.

for $i = 1, \ldots, n$, where

- Y_i is the response for observation i.
- x_i is the value of a predictor/covariate/explanatory variable for obs i.
- π_i is the probability of "success" for observation i.
- β_0 and β_1 are slope and intercept parameters.
- $\pi_i/(1-\pi_i)$ is the odds of "success" for obs i.
- $\log(\pi_i/(1-\pi_i))$ is the log-odds for obs i.

Logistic regression assumes the log-odds are linear in the predictor.

The logit and logistic transformations

logit : $\mathbb{R} \to (0,1)$

logistic : $(0,1) \to \mathbb{R}$

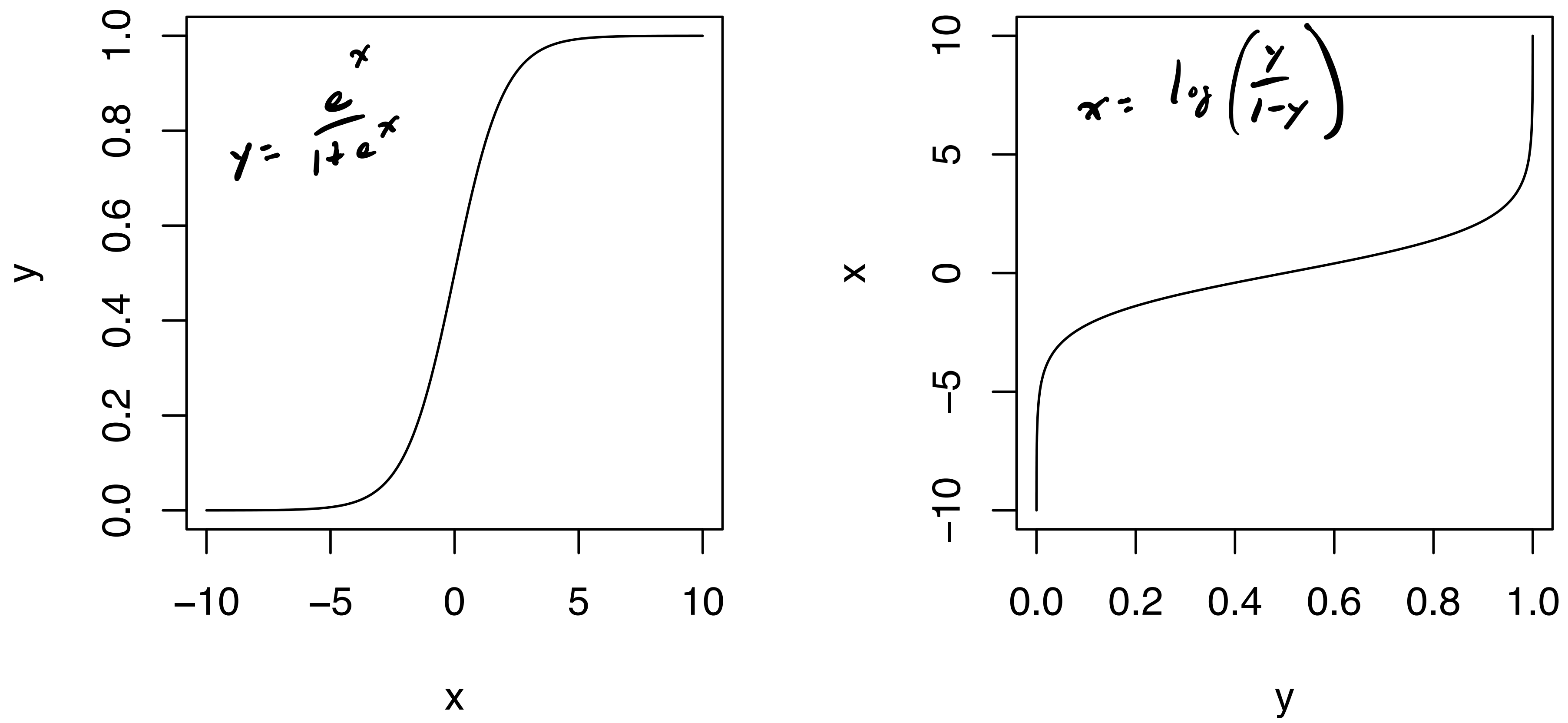
in $(0,1)$.

input is any real number

- The transformation $y = \log(\frac{x}{1-x})$ is called the logit transformation.
- Its inverse $x = \frac{e^y}{1+e^y}$ is called the logistic transformation.
- We have

$$\log\left(\frac{\pi_i}{1-\pi_i}\right) = \beta_0 + \beta_1 x_i \iff \pi_i = \frac{e^{\beta_0+\beta_1 x_i}}{1+e^{\beta_0+\beta_1 x_i}}$$

```
x <- seq(-10,10,length=200)
y <- exp(x) / (1 + exp(x))
par(mfrow= c(1,2))
plot(y~x,type = "l")
plot(x~y,type = "l")
```



Goals in logistic regression

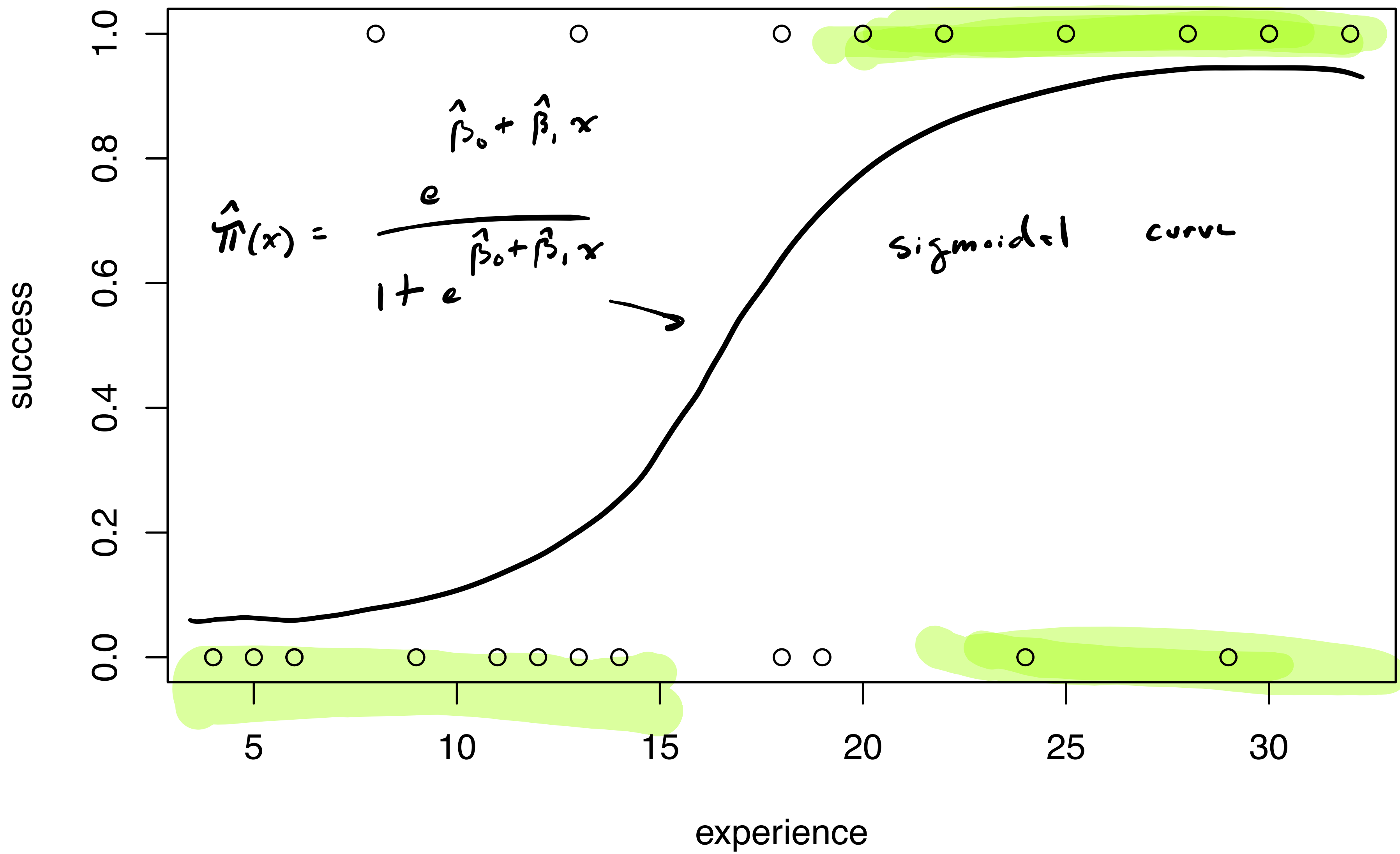
1. Estimate β_0 and β_1.
2. Obtain fitted probabilities $\hat{\pi}_1, \ldots, \hat{\pi}_n$.
3. Build CI for β_1 and test H_0: $\beta_1 = 0$.
4. $\boxed{\text{Give interpretations of the estimated regression coefficients.}}$
5. Check goodness of fit of the logistic regression model.
6. Add additional covariates...

$$\log\left(\frac{\pi_i}{1-\pi_i}\right) = \beta_0 + \beta_1 x_i \iff \pi_i = \frac{e^{\beta_0+\beta_1 x_i}}{1+e^{\beta_0+\beta_1 x_i}}$$

Programming task data (cont)

```
plot(success ~ experience)
```

$y \sim x$



$$\hat{\pi}(x) = \frac{e^{\hat{\beta}_0 + \hat{\beta}_1 x}}{1 + e^{\hat{\beta}_0 + \hat{\beta}_1 x}}$$

Maximum likelihood estimation in logistic regression

$Y \sim \text{Bernoulli}(\pi) \Rightarrow p(y) = \pi^y (1-\pi)^{1-y}$ for $y \in \{0,1\}$.

$Y_i \sim \text{Bernoulli}(\pi_i) \Rightarrow p_i(y_i) = \pi_i^{y_i} (1-\pi_i)^{1-y_i}$ for $y_i \in \{0,1\}$.

- We do not use least-squares to estimate β_0 and β_1.
- Instead we use maximum likelihood estimators (MLEs).
- The MLEs are the parameter values giving the observed data the highest possible probability.
- Intercept b_0 and slope b_1 give to the observed data the probability

prob. of observing my data

$$L \quad \mathcal{L}_n(b_0, b_1) = \prod_{i=1}^{n} \pi_i^{Y_i}(1-\pi_i)^{1-Y_i}, \qquad \pi_i = \frac{e^{b_0+b_1 x_i}}{1+e^{b_0+b_1 x_i}}.$$

- The MLEs $\widehat{\beta}_0$, $\widehat{\beta}_1$ are the values of b_0, b_1 that maximize $\mathcal{L}_n(b_0, b_1)$.
- $\mathcal{L}_n(b_0, b_1)$ is called the likelihood function.

$a_1 \times a_2 \times \cdots \times a_n = \prod_{i=1}^{n} a_i$

$a_1 + a_2 + \cdots + a_n = \sum_{i=1}^{n} a_i$

Computing the MLEs in logistic regression

- There is no "closed-form" expression for $\widehat{\beta}_0$ and $\widehat{\beta}_1$.
- One must find their values numerically, that is with an algorithm.
- More convenient to work with $\log \mathcal{L}_n(b_0, b_1)$, which is given by

$$\ell_n(b_0, b_1) = \sum_{i=1}^{n} [Y_i(b_0 + b_1 x_i) - \log(1 + e^{b_0 + b_1 x_i})].$$

log-likelihood.

- Newton's method is one way to find the maximizers of $\ell_n(b_0, b_1)$.

```r
# DIY Newton method for finding the MLEs
logreg <- function(x,y,tol=1e-6,max_step = 100){

  b <- rep(0,2) # initialize at b0=0 and b1=0
  n <- length(y)
  X <- cbind(rep(1,n),x) # build design matrix
  B <- matrix(0,max_step,2) # matrix to record path of algorithm

  k <- 1
  conv <- F
  while(conv == F){ # Newton's method until convergence

    b_old <- b # store b before updating it
    eta <- X %*% b
    pr <- 1/(1 + exp(-eta))
    w <- pr*(1-pr)
    z <- eta + (y - pr)/w
    XtW <- scale(t(X),F,scale = 1/w)
    Iinv <- solve(XtW %*% X)
    b <- Iinv %*% XtW %*% z # update b
    k <- k + 1
    B[k,] <- b
    conv <- max(abs(b - b_old)) < tol # stop algorithm if change is small

  }

  return(list(bhat = as.numeric(b),
              B = B[1:k,],
              pihat = as.numeric(pr)))

}
```

Programming task data (cont)

```r
# compute the MLEs with the logreg function
logreg_out <- logreg(x = experience,y = success)
B <- logreg_out$B
bhat <- logreg_out$bhat

B
```

```
           [,1]      [,2]
[1,]  0.000000 0.0000000
[2,] -2.368787 0.1261130
[3,] -2.975281 0.1572456
[4,] -3.058263 0.1614150
[5,] -3.059695 0.1614859
[6,] -3.059696 0.1614859
```

beginning of algorithm

```r
bhat
```

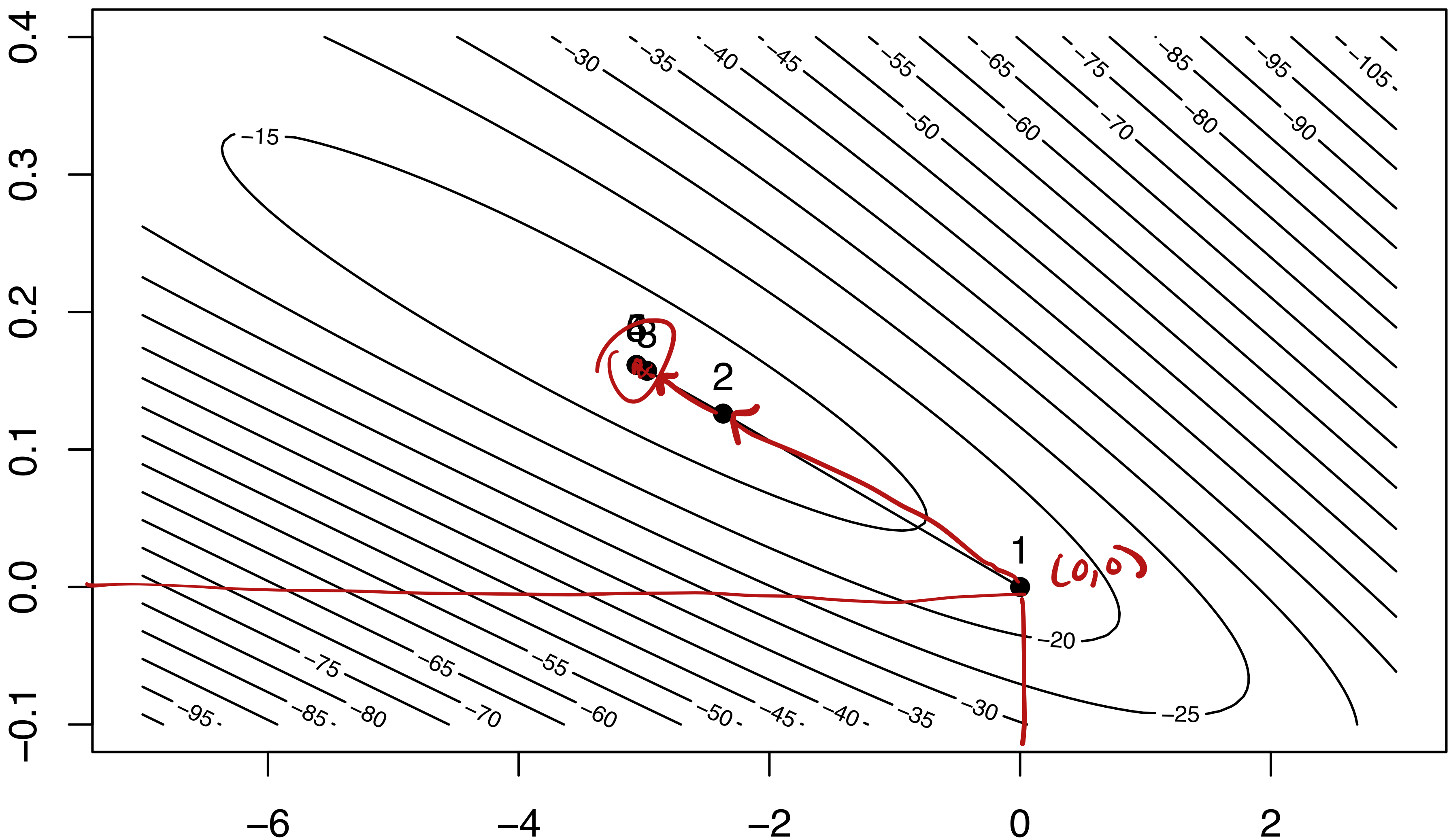
```
[1] -3.0596959  0.1614859
```

$\hat{\beta}_0$ $\hat{\beta}_1$

```r
# draw a contour plot of the log-likelihood and overlay the algorithm's path

# log-likelihood function
ll <- function(b0,b1,x,y){

  eta <- b0 + b1*x
  pr <- 1/(1 + exp(-eta))
  ll <- sum( y*eta - log(1 + exp(eta)))
  return(ll)

}

# evaluate log-likelihood over a grid of b0 and b1 values
ngrid <- 100
b0_seq <- seq(-7,3,length=ngrid)
b1_seq <- seq(-.1,.4,length=ngrid)
ll_vals <- matrix(NA,ngrid,ngrid)
for(i in 1:ngrid)
  for(j in 1:ngrid){

    ll_vals[i,j] <- ll(b0_seq[i],b1_seq[j],x = experience,y = success)

  }
```

```r
contour(x = b0_seq,y = b1_seq,z = ll_vals,nlevels = 20)
lines(x = B[,1],y = B[,2])
points(x = B[,1],y = B[,2],pch = 19)
text(x = B[,1],y = B[,2],labels = paste(1:nrow(B)),pos = 3)
```

$\ell_n(b_0, b_1)$



Generalized linear models

- The logistic regression model is in a class of models called GLMs.
- GLM stands for <u>generalized linear model</u>.
- Poisson regression, binomial response regression, i.a. are GLMs too.
- Use `glm()` function in R to obtain $\hat{\beta}_0$ and $\hat{\beta}_1$.

\# success in n indep. Bernoulli trials

\# independent Bernoulli trials

success probability

$X \sim \text{Binomial}(n, p)$ $\qquad p(x) = \binom{n}{x} p^x (1-p)^{n-x} \qquad x = 0, 1, \ldots, n$

$n = 1$

$Y \sim \text{Binomial}(n=1, p)$ $\qquad p(y) = p^y (1-p)^{1-y}, \qquad y = 0, 1.$

same as Bernoulli(p)

Use glm() function with the option family = "binomial".

```r
glm_out <- glm(success ~ experience, family = "binomial")
summary(glm_out)
```

$y \in \{0,1\}$

x

p-value for testing $H_0: \beta_0 = 0$, but this is usually not interesting.

```
Call:
glm(formula = success ~ experience, family = "binomial")

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -3.05970    1.25935  -2.430   0.0151 *
experience   0.16149    0.06498   2.485   0.0129 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 34.296  on 24  degrees of freedom
Residual deviance: 25.425  on 23  degrees of freedom
AIC: 29.425

Number of Fisher Scoring iterations: 4
```

$\hat\beta_0$

$\hat\beta_1$

p-value for testing $H_0: \beta_1 = 0$

Reject this, conclude there is some relationship between experience and probability of success.

Fitted probabilities

- Define the fitted probabilities as

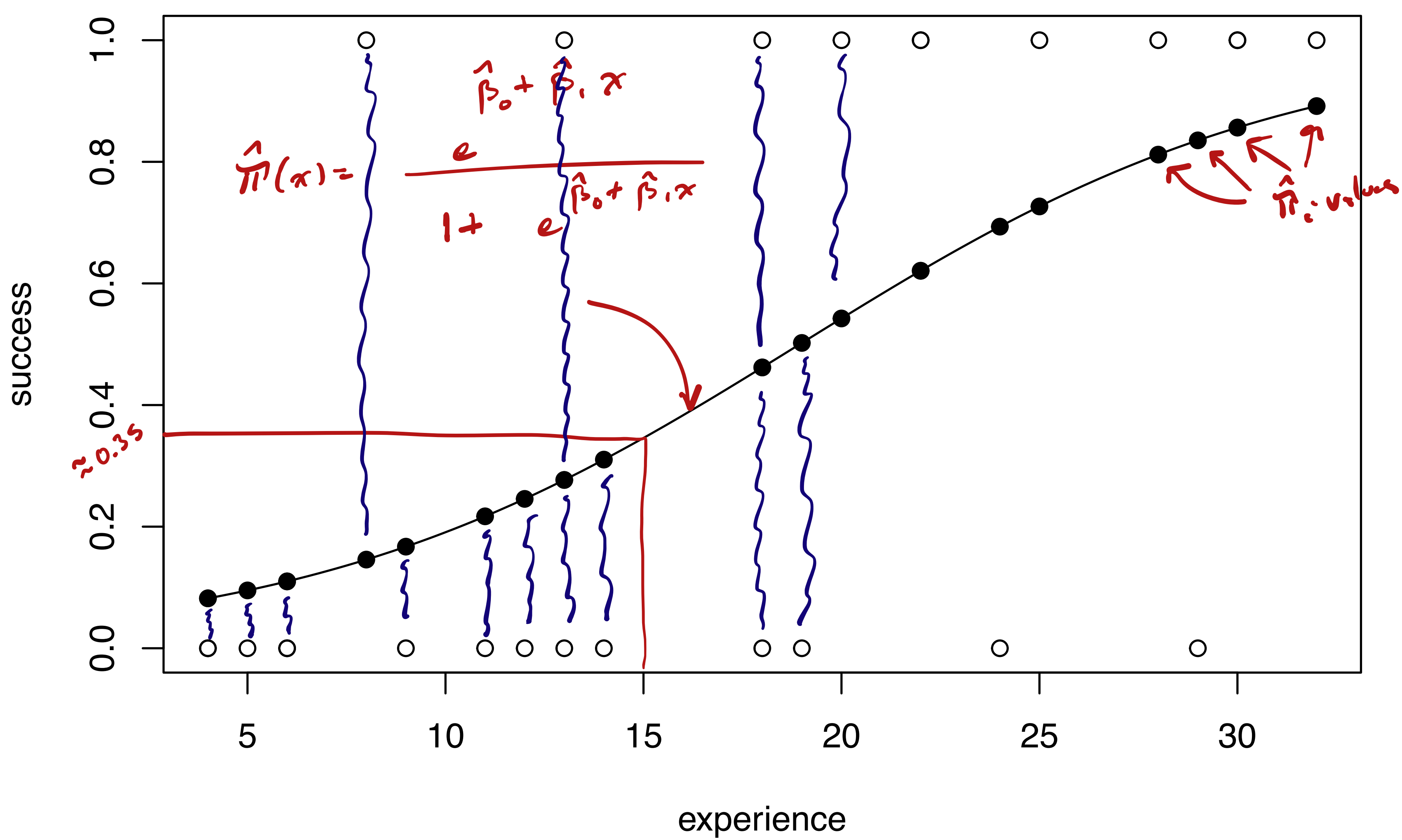
$$\hat{\pi}_i = \frac{e^{\hat{\beta}_0 + \hat{\beta}_1 x_i}}{1 + e^{\hat{\beta}_0 + \hat{\beta}_1 x_i}} \quad \text{for} \quad i = 1, \dots, n.$$

- For any value x_{new}, we estimate the probability of "success" as

$$\hat{\pi}_{\text{new}} = \frac{e^{\hat{\beta}_0 + \hat{\beta}_1 x_{\text{new}}}}{1 + e^{\hat{\beta}_0 + \hat{\beta}_1 x_{\text{new}}}}.$$

Programming task data (cont)

```
x <- seq(min(experience),max(experience),length = 200)
pihat_x <- 1/(1 + exp( -(coef(glm_out)[1] + coef(glm_out)[2]*x)))
plot(success ~ experience); lines(pihat_x~x)
points(glm_out$fitted.values~experience,pch = 19)
```



Asymptotic distribution of slope estimator and CI

For large n...

- For large enough n, $\hat{\beta}_1$ is approximately Normal, such that

$$\frac{\hat{\beta}_1 - \beta_1}{\widehat{\mathrm{se}}\{\hat{\beta}_1\}} \overset{\text{approx}}{\sim} \text{Normal}\,(0, 1)\,,$$

where, setting $\hat{w}_i = \hat{\pi}_i(1 - \hat{\pi}_i)$ for $i = 1, \dots, n$, we may write

$$\widehat{\mathrm{se}}\{\hat{\beta}_1\} = \left[\sum_{i=1}^n \hat{w}_i x_i^2 - (\sum_{i=1}^n \hat{w}_i)^{-1}(\sum_{i=1}^n \hat{w}_i x_i)^2\right]^{-\frac{1}{2}}.$$

- We can make an approximate $(1 - \alpha)100\%$ CI for β_1 as

$$\hat{\beta}_1 \pm z_{\alpha/2}\, \widehat{\mathrm{se}}\{\hat{\beta}_1\}.$$

"standard error" and "standard deviation" are both the square root of the variance.

Programming task data (cont)

```
# DIY confidence interval for beta1
pihat <- logreg_out$pihat
w <- pihat*(1-pihat)
se <- sqrt(1/(sum(w*experience^2) - sum(w*experience)^2/sum(w)))
c(bhat[2] - 1.96 * se,bhat[2] + 1.96 * se)
```

```
[1] 0.03412492 0.28884691
```

```
# CIs for both beta0 and beta1 automatically from glm_out
confint.default(glm_out)
```

```
                   2.5 %     97.5 %
(Intercept) -5.52797622 -0.5914155
experience   0.03412744  0.2888444
```

95% C.I. for β_0

95% C.I. for β_1

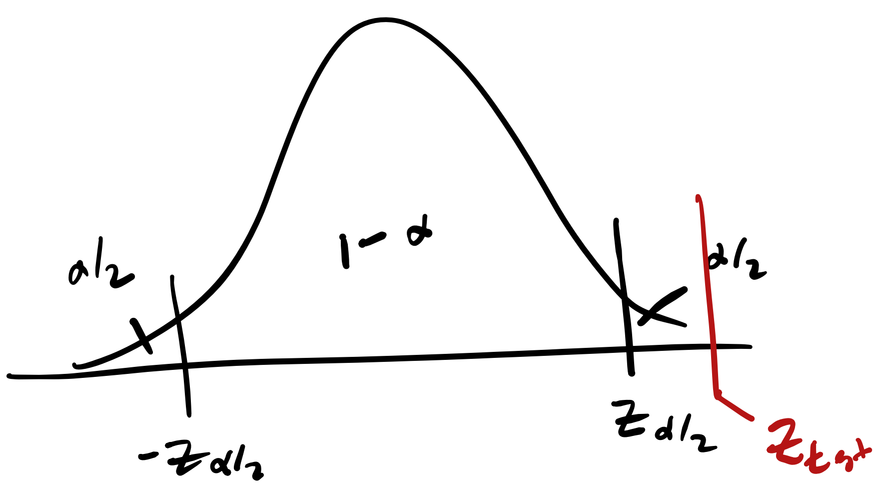
Testing whether the slope coefficient is zero

To test H_0: $\beta_1 = 0$: vs $H_1: \beta_1 \neq 0$.

How many std. dev. is $\hat{\beta}_1$ from 0?

1. Compute $Z_{\text{test}} = \dfrac{\hat{\beta}_1}{\widehat{\text{se}}\{\hat{\beta}_1\}}$.
2. Reject H_0 at α if $|Z_{\text{test}}| > z_{\alpha/2}$.
3. The p value is $2(1 - P(Z > |Z_{\text{test}}|))$, $Z \sim \text{Normal}(0, 1)$.

The `summary()` function on the `glm()` output prints this p value.



Interpreting the logistic regression parameters

$Y_i \sim \text{Bernoulli}(\pi_i)$, $\log\left(\frac{\pi_i}{1-\pi_i}\right) = \beta_0 + \beta_1 x_i$

- Let π_0 and π_1 be the "success" probabilities at x_0 and $x_0 + 1$.
- Then we have the two equations

  $\beta_0 + \beta_1(x_0+1) - (\beta_0 + \beta_1 x_0) = \beta_1$

  1. $\log\left(\frac{\pi_0}{1-\pi_0}\right) = \beta_0 + \beta_1 x_0$
  2. $\log\left(\frac{\pi_1}{1-\pi_1}\right) = \beta_0 + \beta_1 (x_0 + 1)$
- Subtracting the first equation from the second gives

$$\beta_1 = \log\left(\frac{\pi_1}{1-\pi_1}\right) - \log\left(\frac{\pi_0}{1-\pi_0}\right) = \log\left(\frac{\pi_1/(1-\pi_1)}{\pi_0/(1-\pi_0)}\right).$$

odds of success at x_0+1

odds of success at x_0

- The quantity $\boxed{\frac{\pi_1/(1-\pi_1)}{\pi_0/(1-\pi_0)}}$ is called an odds ratio.
- So β_1 is log of the odds ratio associated with a unit increase in x.

factor by which the odds change due to a unit increase in x.

$\log(a \cdot b) = \log a + \log b$

$\log(a/b) = \log a - \log b$

Odds ratio from a unit increase in x

$$\beta_1 = \log\left(\frac{\pi_1/(1-\pi_1)}{\pi_0/(1-\pi_0)}\right)$$

- From the previous slide, we have

$$e^{\beta_1} = \boxed{\frac{\pi_1/(1-\pi_1)}{\pi_0/(1-\pi_0)}}.$$

- Can build a CI for e^{β_1} by exponentiating the CI for β_1.
- Gives CI for e^{β_1} as $(e^{\hat{\beta}_1 - z_{\alpha/2}\widehat{\text{se}}\{\hat{\beta}_1\}}, e^{\hat{\beta}_1 + z_{\alpha/2}\widehat{\text{se}}\{\hat{\beta}_1\}})$.

at x_0: Odds of success vs failure are $\frac{\pi_0}{1-\pi_0}$

at x_0+1: Odds of success vs failure are $\frac{\pi_1}{1-\pi_1}$

$$\frac{\pi_0}{1-\pi_0}\overbrace{\left(\frac{\pi_1/(1-\pi_1)}{\pi_0/(1-\pi_0)}\right)}^{\text{odds ratio}} = \frac{\pi_1}{1-\pi_1}$$

Say $\frac{\pi_0}{1-\pi_0} = 3$

means you are 3x more likely to succeed than to fail.

$\pi_0 = .75$

$1-\pi_0 = .25$

Programming task data (cont)

gives C.I. for β_1

```
exp(confint.default(glm_out,parm = "experience"))
```

specifies we only care about β_1

```
              2.5 %   97.5 %
experience 1.034716 1.334884
```

C.I. for $e^{\beta_1} = \dfrac{\pi_1/(1-\pi_1)}{\pi_0/(1-\pi_0)}$

If I add 1 month of experience, the odds of success increase by a factor in the interval (1.035, 1.335), with 95 % conf.

Residuals for logistic regression

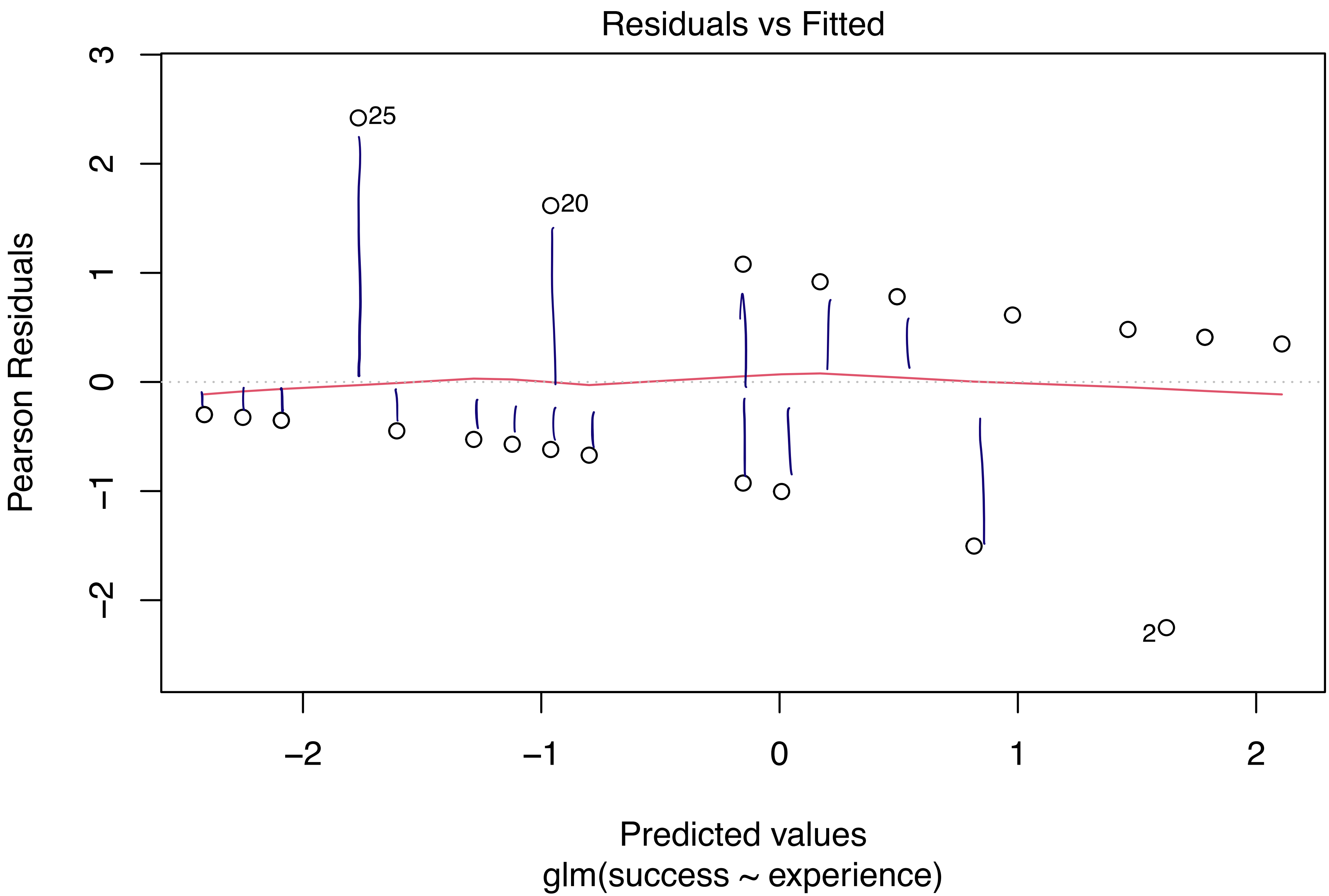
- Ordinary residuals $Y_i - \hat{\pi}_i$ cannot be Normally distributed.
- In GLMs, one looks at special residuals called deviance residuals.
- In logistic regression, the deviance residuals are defined as

$$\hat{d}_i = \operatorname{sign}(Y_i - \hat{\pi}_i)\sqrt{-2[Y_i \log \hat{\pi}_i + (1 - Y_i)\log(1 - \hat{\pi}_i)]}$$

  for $i = 1, \ldots, n$.
- These are not Normal either, but are useful for assessing model fit.

Programming task data (cont)

```
plot(glm_out,which = 1)
```



Checking model fit with a simulated envelope

optional

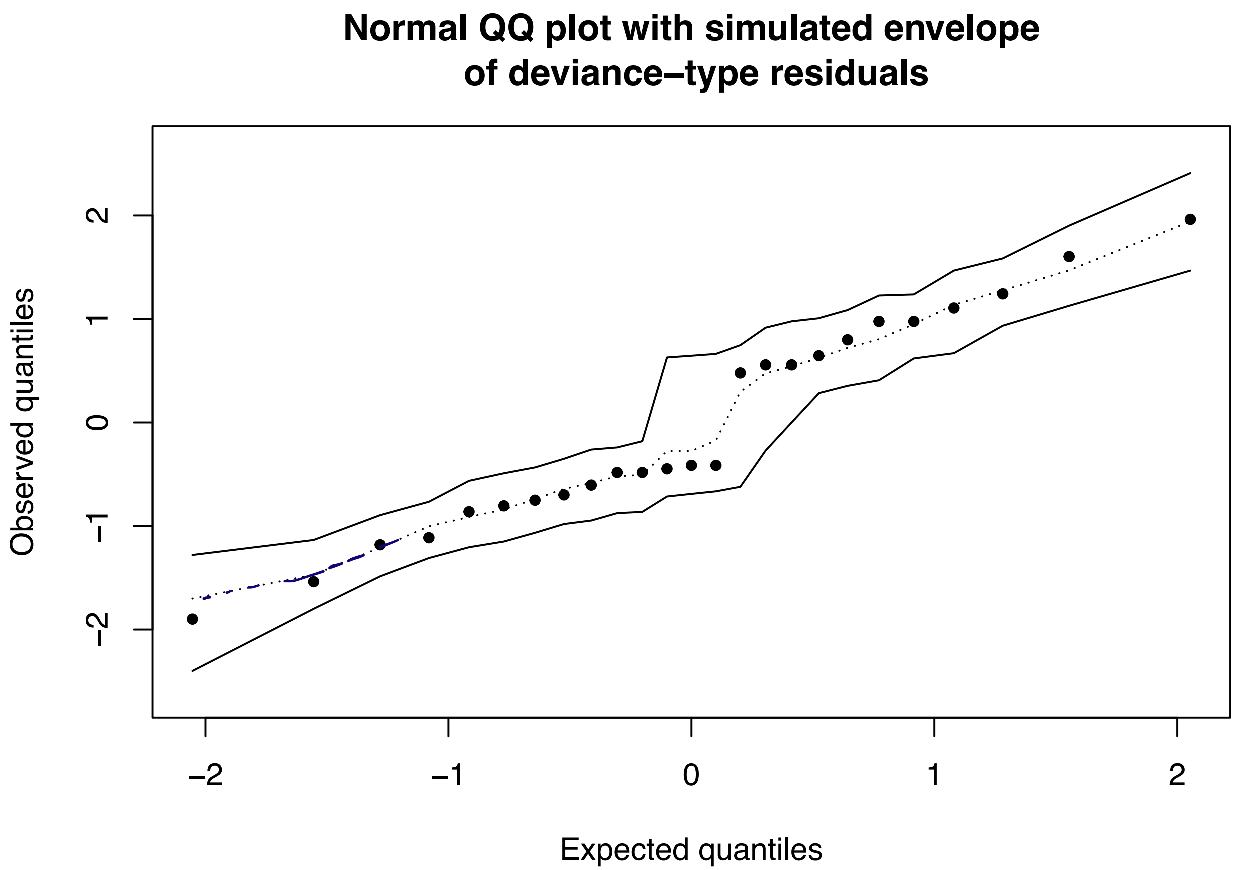
The simulated envelope method is described in Kutner et al. (2005):

- Fit the logistic regression model and obtain $\hat{\pi}_1, \ldots, \hat{\pi}_n$.
- Obtain the deviance residuals; sort them as $\hat{d}_{(1)} < \hat{d}_{(2)} < \cdots < \hat{d}_{(n)}$.
- Generate many new data sets $Y_i^* \sim \text{Bernoulli}(\hat{\pi}_i)$, $i = 1, \ldots, n$.
- For each new data set, obtain sorted $\hat{d}^*_{(1)} < \hat{d}^*_{(2)} < \cdots < \hat{d}^*_{(n)}$.
- Plot $\hat{d}_{(i)}$ as well as the 0.025 and 0.975 quantiles and the mean of the $\hat{d}^*_{(i)}$ $\forall\ i$ (it doesn't matter what is chosen as the x-axis).
- The quantiles of the $\hat{d}^*_{(i)}$ make a band. If the model fits, then the $\hat{d}_{(i)}$ should lie within the band and close to the mean.

Asks: If the model is correct, how would the deviance residuals behave?

Programming task data (cont)

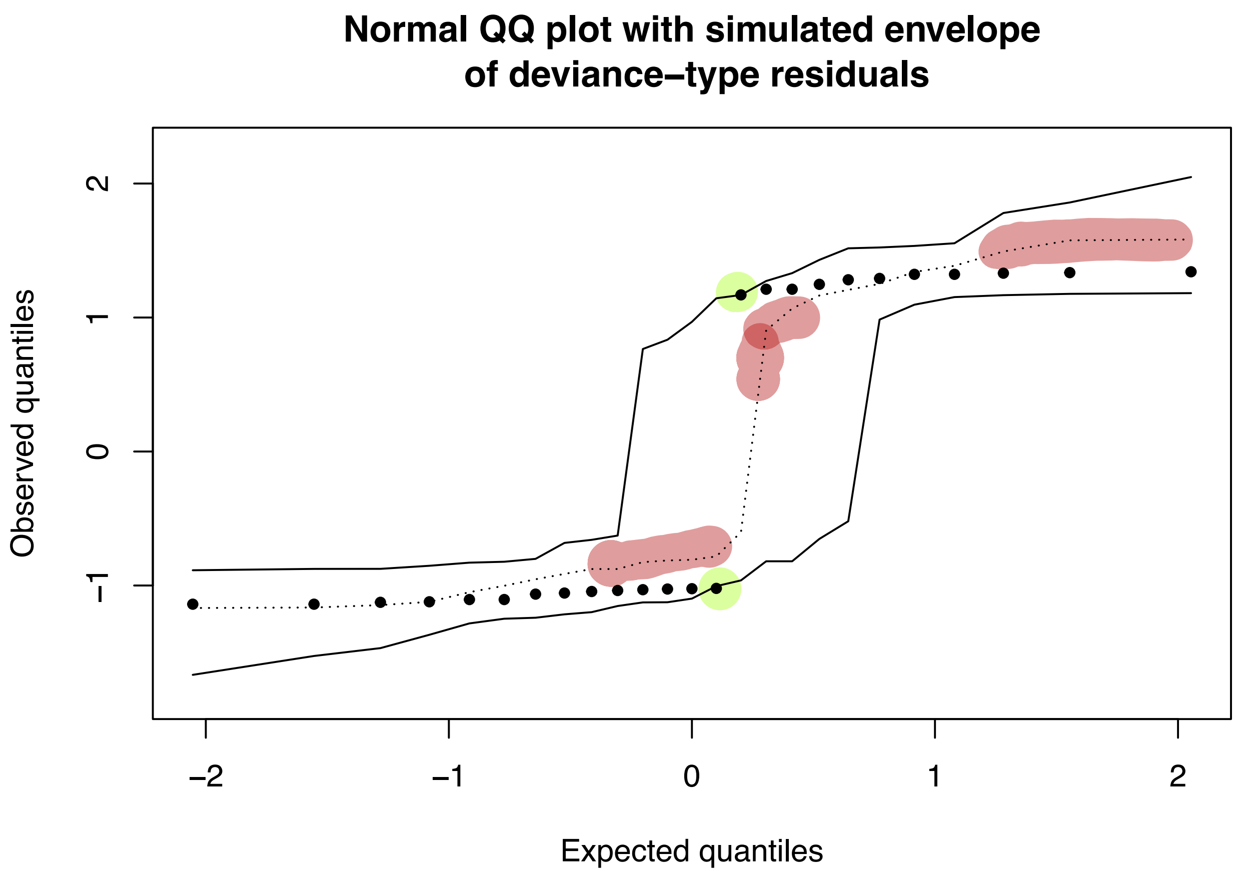
```
library(glmtoolbox) # first time run install.packages("glmtoolbox")
envelope(glm_out,type = "deviance")
```



model 1: $\log\left(\frac{\pi_i}{1-\pi_i}\right) = \beta_0 + \beta_1 x_i$ ← pretty good fit

```
experience2 <- (experience - mean(experience))^2
envelope(glm(success ~ experience2, family = "binomial"),type = "deviance")
```

model 2: $\log\left(\frac{\pi_i}{1-\pi_i}\right) = \beta_0 + \beta_1 (x_i - \bar{x}_n)^2$ ← poor fit.

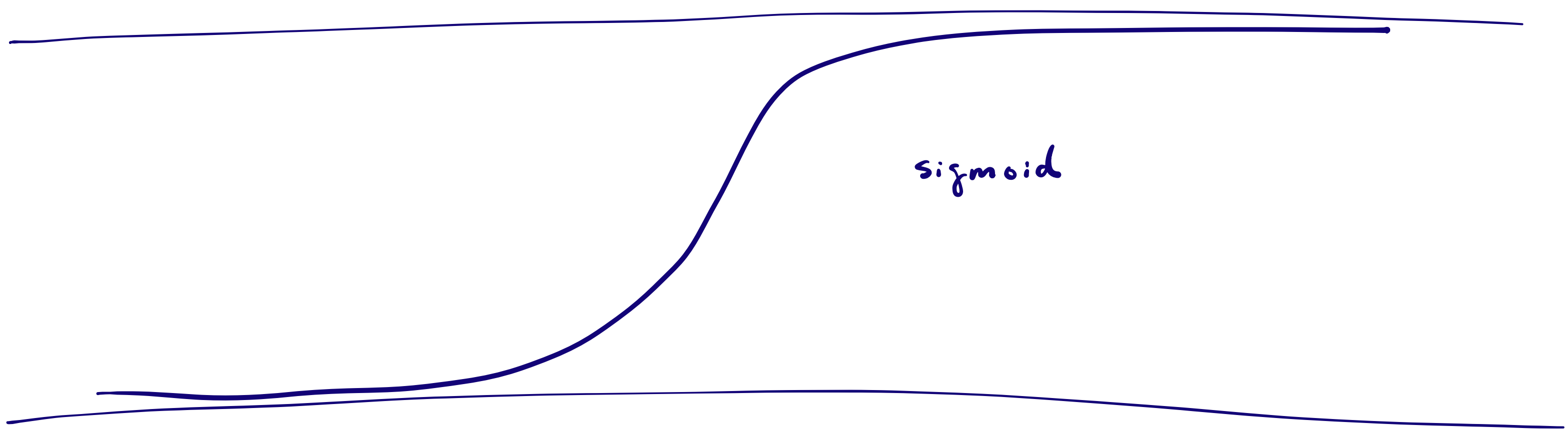


End of Final Exam material

__logistic:__ $Y_i \sim \text{Bernoulli}(\pi_i)$ $\qquad \log\left(\frac{\pi_i}{1-\pi_i}\right) = \beta_0 + \beta_1 x_i$

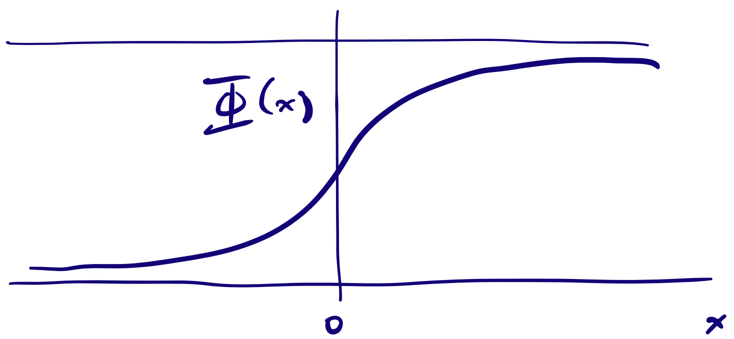
$$\iff \pi_i = \frac{e^{\beta_0 + \beta_1 x_i}}{1 + e^{\beta_0 + \beta_1 x_i}}$$

Gives parameter β_1 interpretations in terms of odds ratios.



Alternative to logistic regression:

$\boxed{\text{Probit regression}}$ $\qquad Y_i \sim \text{Bernoulli}(\pi_i)$ $\qquad \Phi^{-1}(\pi_i) = \beta_0 + \beta_1 x_i$



$$\iff \pi_i = \Phi(\beta_0 + \beta_1 x_i)$$

"phi"

cumulative dist. function of Normal(0,1) dist.

German credit score data from Hofmann (1994)

Response is credit rating (good/bad), various predictors.

```r
library(foreign) # credit-g dataset from https://www.openml.org/
credg <- read.arff("../data/dataset_31_credit-g.arff")
colnames(credg)
```

```
 [1] "checking_status"        "duration"               "credit_history"
 [4] "purpose"                "credit_amount"          "savings_status"
 [7] "employment"             "installment_commitment" "personal_status"
[10] "other_parties"          "residence_since"        "property_magnitude"
[13] "age"                    "other_payment_plans"    "housing"
[16] "existing_credits"       "job"                    "num_dependents"
[19] "own_telephone"          "foreign_worker"         "class"
```

```r
summary(credg[,1:3])
```

```
  checking_status    duration                             credit_history
 <0         :274   Min.   : 4.0   all paid                      : 49
 >=200      : 63   1st Qu.:12.0   critical/other existing credit:293
 0<=X<200   :269   Median :18.0   delayed previously            : 88
 no checking:394   Mean   :20.9   existing paid                 :530
                   3rd Qu.:24.0   no credits/all paid           : 40
                   Max.   :72.0
```

Logistic multiple regression model

Assume

$$Y_i \sim \text{Bernoulli}(\pi_i), \quad \log\left(\frac{\pi_i}{1-\pi_i}\right) = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip},$$

for $i = 1, \ldots, n$, where

- Y_i is the response for observation i.
- $x_{i1}, \ldots, x_{ip}$ are the values of the predictors for obs i.
- π_i is the probability of "success" for observation i.
- β_0 is an intercept and $\beta_1, \ldots, \beta_p$ are slope parameters.
- $\pi_i/(1-\pi_i)$ is the odds of "success" for obs i.
- $\log(\pi_i/(1-\pi_i))$ is the log-odds for obs i.

So we assume the log-odds are a linear function of the predictors.

Interpreting multiple logistic regression parameters

- Let π_{0j} and π_{1j} be the "success" probabilities at x_{0j} and $x_{0j}+1$ but with all other x_{0k} fixed for $k \neq j$.
- Then we have the two equations
  1. $\log\left(\frac{\pi_{0j}}{1-\pi_{0j}}\right) = \beta_0 + \sum_{k\neq j} \beta_k x_{0k} + \beta_j x_{0j}$
  2. $\log\left(\frac{\pi_{1j}}{1-\pi_{1j}}\right) = \beta_0 + \sum_{k\neq j} \beta_k x_{0k} + \beta_j (x_{0j}+1)$
- Subtracting the first equation from the second gives

$$\beta_j = \log\left(\frac{\pi_{1j}/(1-\pi_{1j})}{\pi_{0j}/(1-\pi_{0j})}\right) \quad \text{and} \quad e^{\beta_j} = \frac{\pi_{1j}/(1-\pi_{1j})}{\pi_{0j}/(1-\pi_{0j})}.$$

- So β_1 is log of the odds ratio associated with a unit increase in x_j with all other predictors held fixed.

German credit score data (cont)

```r
glm_out <- glm(class ~ ., family = "binomial", data = credg)
summary(glm_out)
```

```
Call:
glm(formula = class ~ ., family = "binomial", data = credg)

Coefficients:
```

$\hat{\beta}_0$

factor

numeric

```
                                    Estimate Std. Error z value Pr(>|z|)
(Intercept)                        1.505e+00  1.248e+00   1.206 0.227801
checking_status>=200               9.657e-01  3.692e-01   2.616 0.008905 **
checking_status0<=X<200            3.749e-01  2.179e-01   1.720 0.085400 .
checking_statusno checking         1.712e+00  2.322e-01   7.373 1.66e-13 ***
duration                          -2.786e-02  9.296e-03  -2.997 0.002724 **
credit_historycritical/other existing credit  1.579e+00  4.381e-01   3.605 0.000312 ***
credit_historydelayed previously   9.965e-01  4.703e-01   2.119 0.034105 *
credit_historyexisting paid        7.295e-01  3.852e-01   1.894 0.058238 .
credit_historyno credits/all paid  1.434e-01  5.489e-01   0.261 0.793921
purposedomestic appliance         -2.173e-01  8.041e-01  -0.270 0.786976
purposeeducation                  -7.764e-01  4.660e-01  -1.666 0.095718 .
purposefurniture/equipment         5.152e-02  3.543e-01   0.145 0.884391
purposenew car                    -7.401e-01  3.339e-01  -2.216 0.026668 *
purposeother                       7.487e-01  7.998e-01   0.936 0.349202
purposeradio/tv                    1.515e-01  3.370e-01   0.450 0.653002
purposerepairs                    -5.237e-01  5.933e-01  -0.883 0.377428
purposeretraining                  1.319e+00  1.233e+00   1.070 0.284625
purposeused car                    9.264e-01  4.409e-01   2.101 0.035645 *
credit_amount                     -1.283e-04  4.444e-05  -2.887 0.003894 **
savings_status>=1000               1.339e+00  5.249e-01   2.551 0.010729 *
savings_status100<=X<500           3.577e-01  2.861e-01   1.250 0.211130
savings_status500<=X<1000          3.761e-01  4.011e-01   0.938 0.348476
savings_statusno known savings     9.467e-01  2.625e-01   3.607 0.000310 ***
employment>=7                      2.097e-01  2.947e-01   0.712 0.476718
employment1<=X<4                   1.159e-01  2.423e-01   0.478 0.632415
employment4<=X<7                   7.641e-01  3.051e-01   2.504 0.012271 *
employmentunemployed              -6.691e-02  4.270e-01  -0.157 0.875475
installment_commitment            -3.301e-01  8.828e-02  -3.739 0.000185 ***
```

Note that `glm()` estimates three coefficients for `checking_status`.

```
summary(credg$checking_status)
```

```
 <0        >=200    0<=X<200 no checking
274           63         269         394
```

$$\log\left(\frac{\pi_i}{1-\pi_i}\right) = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \beta_3 x_{i3} + \beta_4 x_{i4} + \dots$$

Numeric predictors to encode the levels of the categorical predictor:

"dummy" variables or "indicator" variables.

$$x_{i1} = \begin{cases} 1 & 200 \leq \texttt{checking} \\ 0 & \text{otherwise} \end{cases}$$

$$x_{i2} = \begin{cases} 1 & 0 \leq \texttt{checking} < 200 \\ 0 & \text{otherwise} \end{cases}$$

$$x_{i3} = \begin{cases} 1 & \texttt{no checking} \\ 0 & \text{otherwise} \end{cases}$$

Likewise for other categorical predictors.

Deviances replace error sums of squares in GLMs

- The deviance is the sum of squared *deviance* residuals $\sum_{i=1}^{n} \hat{d}_i^2$.
- In logistic regression the deviance can be computed as

$$\text{Dev} = -2\sum_{i=1}^{n}[Y_i \log \hat{\pi}_i + (1 - Y_i)\log(1 - \hat{\pi}_i)].$$

- Full-reduced model test: Reject H_0: $\beta_j = 0$ for all $j \in D$ if

$$\text{Dev}(\text{Reduced}) - \text{Dev}(\text{Full}) > \chi^2_{s,\alpha},$$

where s is the number of predictors in D (need large n).

German credit score data (cont)

Test whether any level of checking status is important to the credit score.

```
credg_red <- credg[,-1] # remove checking_status column

glm_full <- glm(class ~ ., family = "binomial", data = credg)
glm_red <-  glm(class ~ ., family = "binomial", data = credg_red)

p <- length(coef(glm_full)) - 1
s <- nlevels(credg$checking_status) - 1

1 - pchisq(glm_red$deviance - glm_full$deviance,s)
```

```
[1] 2.731149e-14
```

Print sequential changes in the deviance with the anova() function.

```
anova(glm_out) # gives reduction in deviance due to adding the predictor (sequential)
```

```
Analysis of Deviance Table

Model: binomial, link: logit

Response: class

Terms added sequentially (first to last)

                       Df Deviance Resid. Df Resid. Dev
NULL                                     999    1221.73
checking_status         3  131.336       996    1090.39
duration                1   38.497       995    1051.90
credit_history          4   29.311       991    1022.58
purpose                 9   33.509       982     989.08
credit_amount           1    1.504       981     987.57
savings_status          4   19.068       977     968.50
employment              4   12.496       973     956.01
installment_commitment  1   11.907       972     944.10
personal_status         3    9.459       969     934.64
other_parties           2    8.137       967     926.51
residence_since         1    0.155       966     926.35
property_magnitude      3    2.520       963     923.83
age                     1    3.725       962     920.11
other_payment_plans     2    8.357       960     911.75
housing                 2    3.517       958     908.23
existing_credits        1    2.328       957     905.90
job                     3    1.110       954     904.79
num_dependents          1    1.049       953     903.74
own_telephone           1    1.863       952     901.88
foreign_worker          1    6.063       951     895.82
```

Compute the first few rows of the above table:

```
glm0 <- glm(class ~ 1, family = "binomial", data = credg)
glm1 <- glm(class ~ checking_status, family = "binomial", data = credg)
glm2 <- glm(class ~ checking_status + duration, family = "binomial", data = credg)
glm3 <- glm(class ~ checking_status + duration + credit_history,
            family = "binomial", data = credg)

y <- as.numeric(credg$class == "good")
pi0 <- mean(y)
pi1 <- glm1$fitted.values
pi2 <- glm2$fitted.values
pi3 <- glm3$fitted.values

ll0 <- sum(y*log(pi0) + (1-y) * log(1-pi0))
ll1 <- sum(y*log(pi1) + (1-y) * log(1-pi1))
ll2 <- sum(y*log(pi2) + (1-y) * log(1-pi2))
ll3 <- sum(y*log(pi3) + (1-y) * log(1-pi3))
```

```r
n <- length(y)
df0 <- 1
df1 <- nlevels(credg$checking_status) - 1
df2 <- 1
df3 <- nlevels(credg$credit_history) - 1

devtab <- rbind(c(NA,NA,n - df0,-2*ll0),
                c(df1,-2*(ll0 - ll1),n - df0 - df1,-2*ll1),
                c(df2,-2*(ll1 - ll2),n - df0 - df1 - df2,-2*ll2),
                c(df3,-2*(ll2 - ll3),n - df0 - df1 - df2 - df3,-2*ll3))

colnames(devtab) <- c("Df","Deviance","Resid Df","Resid Dev")
rownames(devtab) <- c("(Intercept)","checking_status","duration","credit_history")

devtab
```

```
                Df  Deviance Resid Df Resid Dev
(Intercept)     NA        NA      999  1221.729
checking_status  3 131.33592      996  1090.393
duration         1  38.49674      995  1051.896
credit_history   4  29.31100      991  1022.585
```

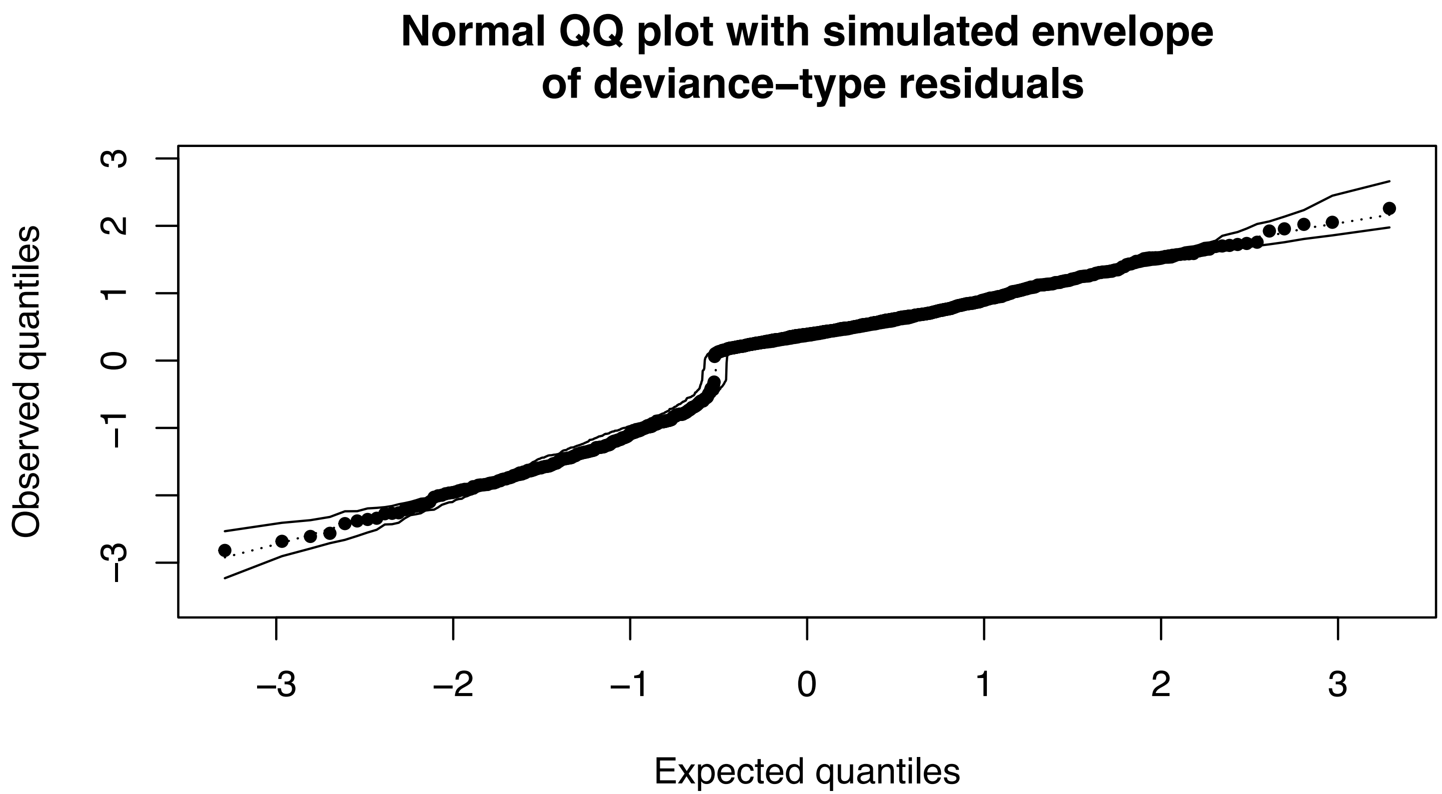
Variable selection in logistic regression

- We may want to discard some of our predictors.
- One way is to add/remove variables stepwise according to AIC.
- Can do this just as we did in multiple linear regression.
- Be cautious about making inferences after selecting a model.

```r
glm_all <- glm(class ~ ., family = "binomial", data = credg)
step_back <- step(glm_all,
                  direction = "backward",
                  scope = formula(glm_all),
                  criterion = "aic",
                  trace = 0) # suppress printed output
summary(step_back)
```

```
Call:
glm(formula = class ~ checking_status + duration + credit_history +
    purpose + credit_amount + savings_status + installment_commitment +
    personal_status + other_parties + age + other_payment_plans +
    housing + own_telephone + foreign_worker, family = "binomial",
    data = credg)

Coefficients:
                                              Estimate Std. Error z value Pr(>|z|)
(Intercept)                                  4.838e-01  1.017e+00   0.476 0.634362
checking_status>=200                         1.024e+00  3.626e-01   2.824 0.004739 **
checking_status0<=X<200                      3.900e-01  2.121e-01   1.839 0.065928 .
checking_statusno checking                   1.718e+00  2.281e-01   7.531 5.05e-14 ***
duration                                    -2.568e-02  8.940e-03  -2.872 0.004074 **
credit_historycritical/other existing credit  1.373e+00  4.041e-01   3.397 0.000680 ***
credit_historydelayed previously             7.910e-01  4.488e-01   1.762 0.077985 .
credit_historyexisting paid                  7.115e-01  3.788e-01   1.879 0.060305 .
credit_historyno credits/all paid           -1.188e-01  5.268e-01  -0.225 0.821612
purposedomestic appliance                   -2.576e-01  7.763e-01  -0.332 0.740041
purposeeducation                            -9.262e-01  4.569e-01  -2.027 0.042628 *
purposefurniture/equipment                  -4.216e-02  3.415e-01  -0.123 0.901748
purposenew car                              -7.827e-01  3.272e-01  -2.392 0.016752 *
purposeother                                 6.523e-01  7.832e-01   0.833 0.404946
purposeradio/tv                              1.368e-01  3.288e-01   0.416 0.677335
purposerepairs                              -6.402e-01  5.808e-01  -1.102 0.270365
purposeretraining                            1.382e+00  1.240e+00   1.114 0.265228
purposeused car                              8.246e-01  4.288e-01   1.923 0.054495 .
credit_amount                               -1.294e-04  4.221e-05  -3.066 0.002169 **
savings_status>=1000                         1.289e+00  5.072e-01   2.542 0.011008 *
savings_status100<=X<500                     3.282e-01  2.767e-01   1.186 0.235477
savings_status500<=X<1000                    4.304e-01  3.933e-01   1.094 0.273900
savings_statusno known savings               9.628e-01  2.570e-01   3.746 0.000179 ***
```

```
envelope(step_back,type="deviance")
```



Classification with logistic regression.

$$\underset{\sim}{x}_{new} = (x_{new,1}, \dots, x_{new,p})$$

$\hat{\pi}(\underset{\sim}{x}_{new}) \leftarrow$ estimated probability of "success".

$$\widehat{pred}_\tau(\underset{\sim}{x}_{new}) = \begin{cases} 1 & \hat{\pi}(\underset{\sim}{x}_{new}) \geq \tau \\ 0 & \hat{\pi}(\underset{\sim}{x}_{new}) < \tau \end{cases}$$

Compute $\widehat{pred}_\tau(\underset{\sim}{x}_i)$ for all observed $\underset{\sim}{x}_1, \dots, \underset{\sim}{x}_n$.

	obs Y_i: 1	0
predicted Y_i: 1	correct	X
0	X	correct

$$\text{Misclassification Rate} = \frac{\#\text{ misclassified}}{n}$$

Data set

split.

Training set

build

model

Testing set

test

model

References

Hofmann, Hans. 1994. "Statlog (German Credit Data)." UCI Machine Learning Repository.

Kutner, Michael H, Christopher J Nachtsheim, John Neter, and William Li. 2005. *Applied Linear Statistical Models*. McGraw-hill.