

STAT 516 Lec 12

Logistic regression

Karl Gregory

2026-04-20

Programming task data from Kutner et al. (2005)

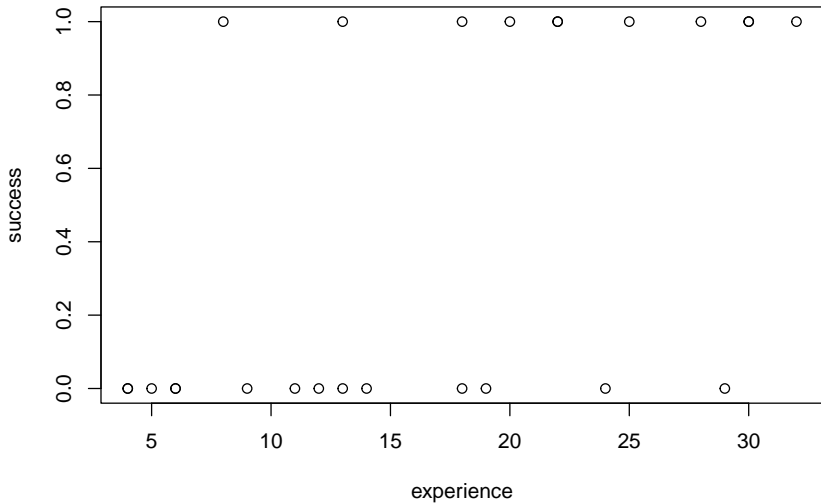
Twenty-five people succeeded or failed at a programming task.

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(0,0,0,1,1,0,0,0,1,0,1,0,1,0,0,1,0,1,0,0,1,0,1,1,1)
```

Can we predict probability of success based on experience?

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



Logistic regression model

Assume

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

for $i = 1, \dots, n$, 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.

Odds



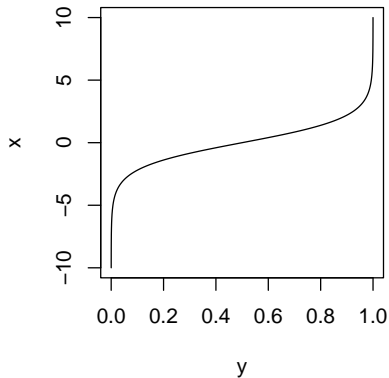
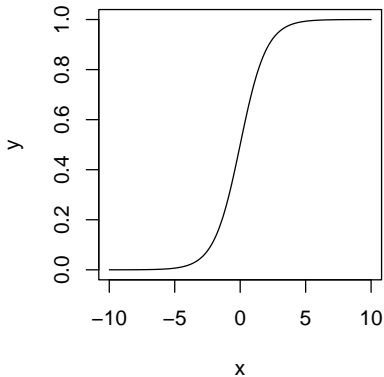
- ▶ Let π be the probability of success.
- ▶ Then $\pi/(1 - \pi)$ is called the odds in favor of success.
 - a. If $\pi = 1/2$ then $\pi/(1 - \pi) = 1$. "One-to-one" odds of success.
 - b. If $\pi = 2/3$ then $\pi/(1 - \pi) = 2$. Success 2x more likely than failure.
 - c. If $\pi = 1/4$ then $\pi/(1 - \pi) = 1/3$. Failure 3x more likely than success.

The logit and logistic transformations

- ▶ The transformation $y = \frac{e^x}{1+e^x}$ is called the logistic transformation.
- ▶ Its inverse $x = \log\left(\frac{y}{1-y}\right)$ is called the logit 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 $\hat{\pi}_1, \dots, \hat{\pi}_n$.
3. Build CI for β_1 and test $H_0: \beta_1 = 0$.
4. Give interpretations of the estimated regression coefficients.
5. Check goodness of fit of the logistic regression model.
6. Add additional covariates...

Maximum likelihood estimation in logistic regression

- ▶ 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

$$\mathcal{L}_n(b_0, b_1) = \prod_{i=1}^n [\pi_i(b_0, b_1)]^{Y_i} [1 - \pi_i(b_0, b_1)]^{1-Y_i}$$

with $\pi_i(b_0, b_1) = \frac{e^{b_0 + b_1 x_i}}{1 + e^{b_0 + b_1 x_i}}$ for $i = 1, \dots, n$.

- ▶ The MLEs $\hat{\beta}_0, \hat{\beta}_1$ 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.

Computing the MLEs in logistic regression

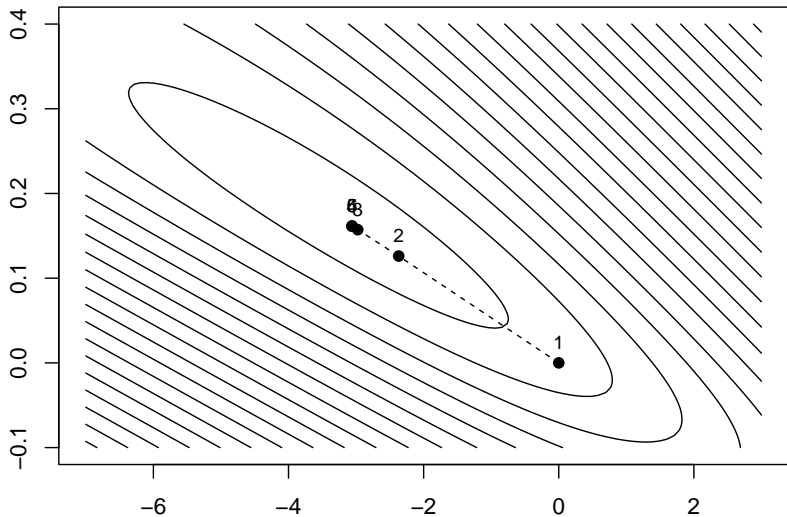
- ▶ There is no “closed-form” expression for $\hat{\beta}_0$ and $\hat{\beta}_1$.
- ▶ One must find their values numerically, that is with an algorithm.
- ▶ More convenient to work with $\overline{\log \mathcal{L}_n(b_0, b_1)}$, which is given by

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

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

Programming task data (cont)

Newton-Raphson algorithm for computing $\hat{\beta}_0$ and $\hat{\beta}_1$.



Generalized linear models

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

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

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

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 *

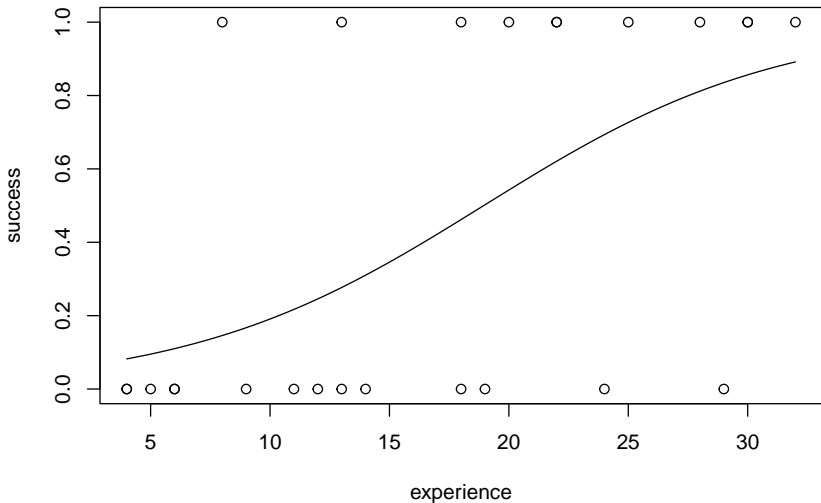
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(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

```
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)
```



Fitted probabilities

- ▶ Define the fitted probabilities as

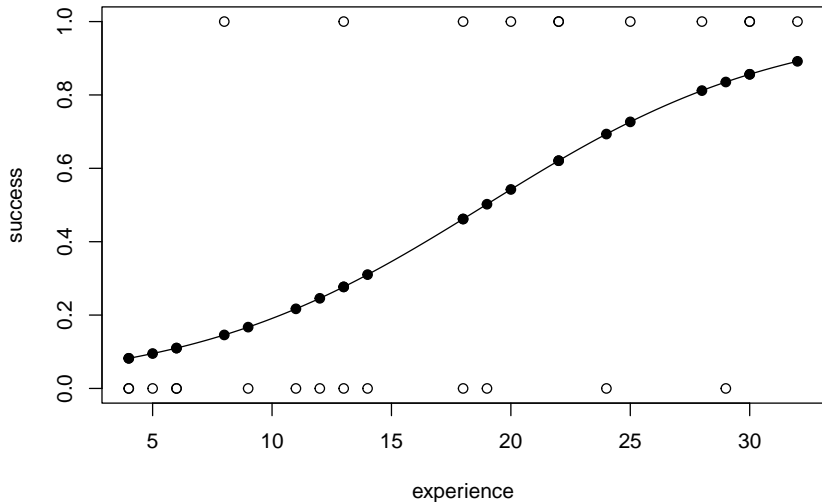
$$\hat{\pi}_i = \frac{e^{\hat{\beta}_0 + \hat{\beta}_1 x_i}}{1 + e^{\hat{\beta}_0 + \hat{\beta}_1 x_i}} \quad \text{for } i = 1, \dots, n.$$

- ▶ 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)

```
plot(success ~ experience); lines(pihat_x~x)  
points(glm_out$fitted.values~experience,pch = 19)
```



Asymptotic distribution of slope estimator and CI

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

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

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

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

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

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

Programming task data (cont)

```
# Confidence interval for beta1
pihat <- glm_out$fitted.values
bihat <- coef(glm_out)[2]
w <- pihat*(1-pihat)
se <- sqrt(1/(sum(w*experience^2) - sum(w*experience)^2/sum(w)))
lo <- bihat - 1.96 * se
up <- bihat + 1.96 * se
c(lo,up)
```

```
experience experience
0.03412491 0.28884692
```

```
# 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
```

Testing whether the slope coefficient is zero

To test $H_0: \beta_1 = 0$ versus $H_1: \beta_1 \neq 0$, do:

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

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

Odds ratios

- ▶ Let π_0 and π_1 be success probs under an initial and an altered condition, respectively.

- ▶ Then we call the ratio

$$\frac{\pi_1/(1 - \pi_1)}{\pi_0/(1 - \pi_0)}$$

the odds ratio associated with the change from the initial to the altered condition.

- ▶ The odds ratio is the factor by which the odds are multiplied when the initial condition is changed to the altered condition.

Interpreting the logistic regression parameters

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

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

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

Odds ratio from a unit increase in x

- ▶ From the previous slide, we have

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

- ▶ Can build a CI for e^{β_1} by exponentiating the CI for β_1 .
- ▶ Gives CI for e^{β_1} as $[e^{\hat{\beta}_1 - z_{\alpha/2} \widehat{\text{se}}\{\hat{\beta}_1\}}, e^{\hat{\beta}_1 + z_{\alpha/2} \widehat{\text{se}}\{\hat{\beta}_1\}}]$.
- ▶ A unit increase in x multiplies the odds of success by the factor e^{β_1} .
- ▶ What if the CI for e^{β_1} contains 1?

Programming task data (cont)

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

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

Each additional month of experience increases the odds of completing the programming task by a factor of 1.035 to 1.335, with 95% confidence.

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

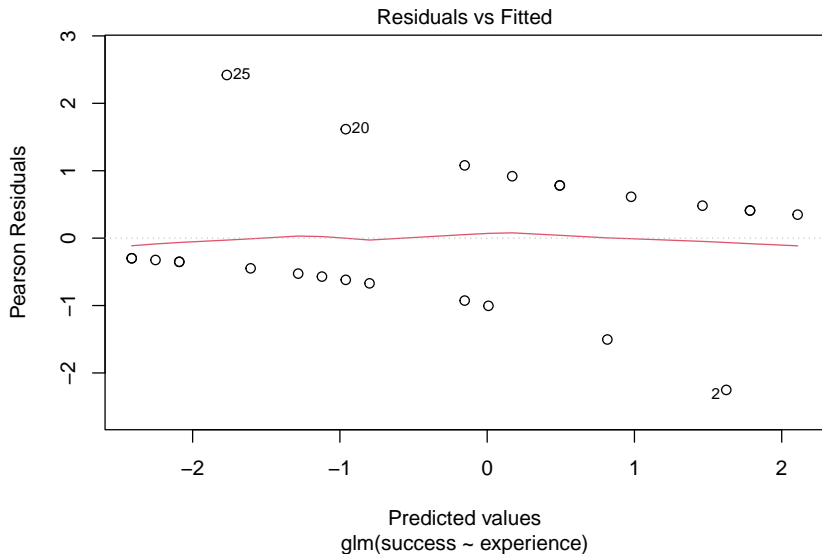
$$\hat{d}_i = \text{sign}(Y_i - \hat{\pi}_i) \sqrt{-2[Y_i \log \hat{\pi}_i + (1 - Y_i) \log(1 - \hat{\pi}_i)]}$$

for $i = 1, \dots, n$.

- ▶ 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

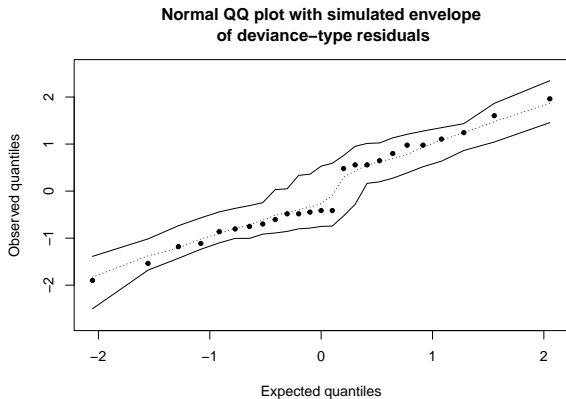
The simulated envelope method is described in Kutner et al. (2005):

- ▶ Fit the logistic regression model and obtain $\hat{\pi}_1, \dots, \hat{\pi}_n$.
- ▶ Obtain the deviance residuals; sort them as $\hat{d}_{(1)} < \hat{d}_{(2)} < \dots < \hat{d}_{(n)}$.
- ▶ Generate many new data sets $Y_i^* \sim \text{Bernoulli}(\hat{\pi}_i)$, $i = 1, \dots, n$.
- ▶ For each new data set, obtain sorted $\hat{d}_{(1)}^* < \hat{d}_{(2)}^* < \dots < \hat{d}_{(n)}^*$.
- ▶ 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?

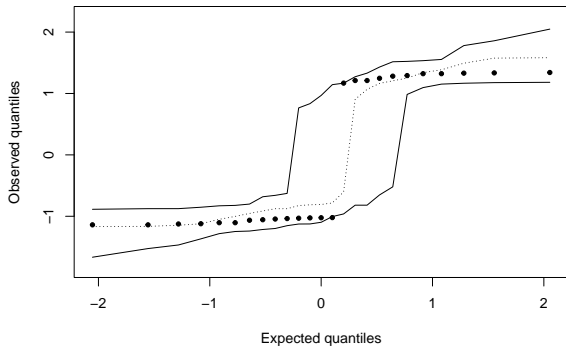
Programming task data (cont)

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



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

Normal QQ plot with simulated envelope
of deviance-type residuals



German credit score data from Hofmann (1994)

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

```
library(foreign) # credit-g dataset from https://www.openml.org/  
link <- url("https://people.stat.sc.edu/gregorkb/data/dataset_31_credit-g.arff")  
credg <- read.arff(link)  
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"
```

```
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

Observe $(\mathbf{x}_1, Y_1), \dots, (\mathbf{x}_n, Y_n)$, $\mathbf{x}_i = (x_{i1}, \dots, x_{ip})^T$ and assume

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

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

- ▶ Y_i is the response for observation i .
- ▶ x_{i1}, \dots, 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 β_1, \dots, β_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 x_{0k} fixed for all $k \neq j$.

- ▶ 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)

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

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

Coefficients:

	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
purposeeducation	-2.173e-01	8.041e-01	-0.270	0.786976
purposefurniture/equipment	-7.764e-01	4.660e-01	-1.666	0.095718 .
purposeother	5.152e-02	3.543e-01	0.145	0.884391
purposecar	-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
purposeother	-5.237e-01	5.933e-01	-0.883	0.377428
purposeother	1.319e+00	1.233e+00	1.070	0.284625
purposecar	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

Numeric predictors to encode the levels of the categorical predictor:

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

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

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

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

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

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
```

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.

German credit score data (cont)

```
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:

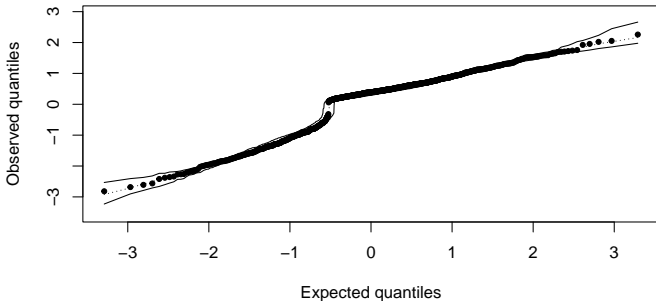
```
glm(formula = class ~ checking_status + duration + credit_history +
  purpose + credit_amount + savings_status + installment_commitment +
  personal_status + other_parties + age + other_payment_plans +
  housing + own_telephone + foreign_worker, family = "binomial",
  data = credg)
```

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
purpose domestic appliance	-2.576e-01	7.763e-01	-0.332	0.740041
purpose education	-9.262e-01	4.569e-01	-2.027	0.042628 *
purpose furniture/equipment	-4.216e-02	3.415e-01	-0.123	0.901748
purpose new car	-7.827e-01	3.272e-01	-2.392	0.016752 *
purpose other	6.523e-01	7.832e-01	0.833	0.404946
purpose radio/tv	1.368e-01	3.288e-01	0.416	0.677335
purpose repairs	-6.402e-01	5.808e-01	-1.102	0.270365
purpose retraining	1.382e+00	1.240e+00	1.114	0.265228
purpose used car	8.246e-01	4.288e-01	1.923	0.054495 .
credit_amount	-1.294e-04	4.221e-05	-3.066	0.002169 **

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

Normal QQ plot with simulated envelope
of deviance-type residuals



Classification with the logistic regression model

Consider classifying the observations as 1 or 0 according to $\hat{\pi}_i$:

- ▶ Choose a threshold $c \in [0, 1]$ and make the classification

$$\hat{Y}_i = \begin{cases} 1, & \hat{\pi}_i \geq c \\ 0, & \hat{\pi}_i < c. \end{cases}$$

- ▶ Can compute misclassification rate $\#\{\hat{Y}_i \neq Y_i\}/n$.
- ▶ Can compute observed true positive and false positive rates

$$\text{TP} = \frac{\#\{\hat{Y}_i = 1 \cap Y_i = 1\}}{\#\{Y_i = 1\}} \quad \text{and} \quad \text{FP} = \frac{\#\{\hat{Y}_i = 1 \cap Y_i = 0\}}{\#\{Y_i = 0\}}.$$

- ▶ Plotting TP against FP over all $c \in [0, 1]$ creates the receiver operating characteristic (ROC) curve.

German credit score data (cont)

Compute TP and FP over a range of thresholds c . Plot ROC curve.

```
Y <- ifelse(credg$class == "good",1,0)
pi_hat <- step_back$fitted.values

n1 <- sum(Y == 1)
n0 <- sum(Y == 0)

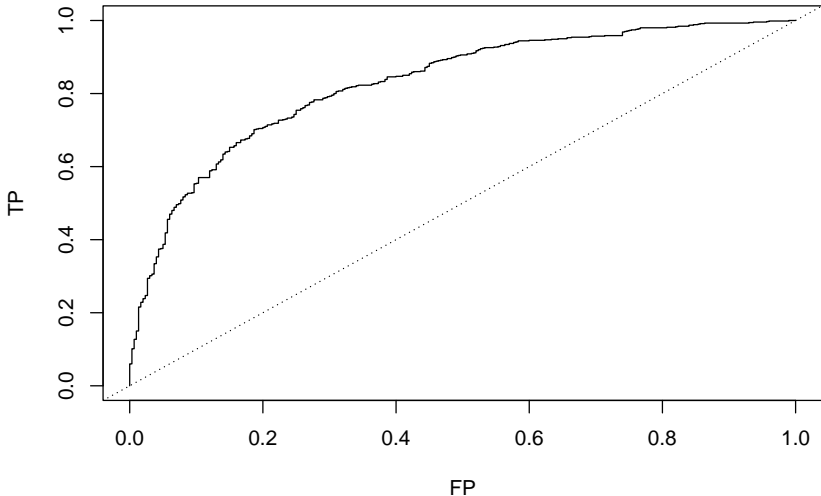
cc <- sort(c(0,pi_hat,1))
TP <- FP <- numeric(length(cc))
for(j in 1:length(cc)){

  Yhat <- pi_hat >= cc[j]
  TP[j] <- sum(Yhat == 1 & Y == 1) / n1
  FP[j] <- sum(Yhat == 1 & Y == 0) / n0

}
```

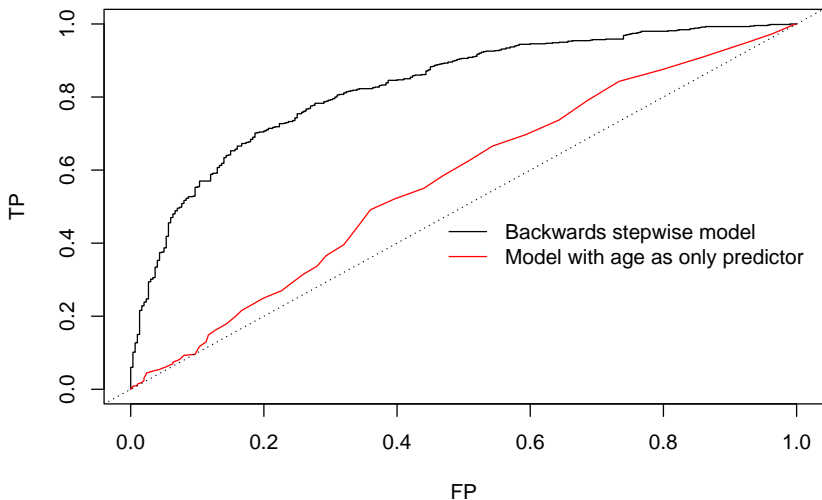
```
plot(TP ~ FP, type = "l", main = "ROC curve")  
abline(0,1,lty = 3)
```

ROC curve



Can use ROC curves to compare models. ¹

ROC curves



¹Best to evaluate model performance on a set of data not used in fitting the model.

Training and testing data sets

- ▶ Overfitting is fitting a model such that it performs well on the data used to fit it, but will perform poorly when it is given new data.
- ▶ To avoid, split data into training and testing data sets

$$\{(\mathbf{x}_i^{\text{train}}, Y_i^{\text{train}})\}_{i \in \mathcal{J}_{\text{train}}} \quad \text{and} \quad \{(\mathbf{x}_i^{\text{test}}, Y_i^{\text{test}})\}_{i \in \mathcal{J}_{\text{test}}},$$

where $\mathcal{J}_{\text{train}} \cap \mathcal{J}_{\text{test}} = \emptyset$ and $\mathcal{J}_{\text{train}} \cup \mathcal{J}_{\text{test}} = \{1, \dots, n\}$.

- ▶ Fit models on training data. Compare performance on testing data.
- ▶ For example, compute misclassification rate or draw ROC curve for testing data.

German credit score data (cont)

```
set.seed(1)
n <- nrow(credg)
n_train <- ceiling(.5*n)
n_test <- n - n_train
ind_train <- sample(n,n_train) # randomly select a training set

credg_train <- credg[ind_train,]
credg_test <- credg[-ind_train,]

glm_train <- glm(class ~ ., family = "binomial", data = credg_train)

ypred_train <- glm_train$fitted.values >= 0.5
y_train <- as.numeric(credg_train$class == "good")
misclass_train <- mean(ypred_train != y_train)
misclass_train
```

[1] 0.204

```
pihat_test <- predict(glm_train,newdata = credg_test,type="response")

ypred_test <- pihat_test >= 0.5
y_test <- as.numeric(credg_test$class == "good")

misclass_test <- mean(ypred_test != y_test)
misclass_test
```

[1] 0.258

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.