Ve406 Lecture 17

Jing Liu

UM-SJTU Joint Institute

July 16, 2018

- So far, we have only consider the following class of regression models

$$Y_i = \mathbb{E}\left[Y_i \mid \mathbf{X}\right] + \varepsilon_i$$

where the conditional mean is assumed to take an additive form

$$\mathbb{E}\left[Y_i \mid \mathbf{X}\right] = \beta_0 + g(x_{i1}) + g(x_{i2}) + \cdots + g(x_{ik})$$

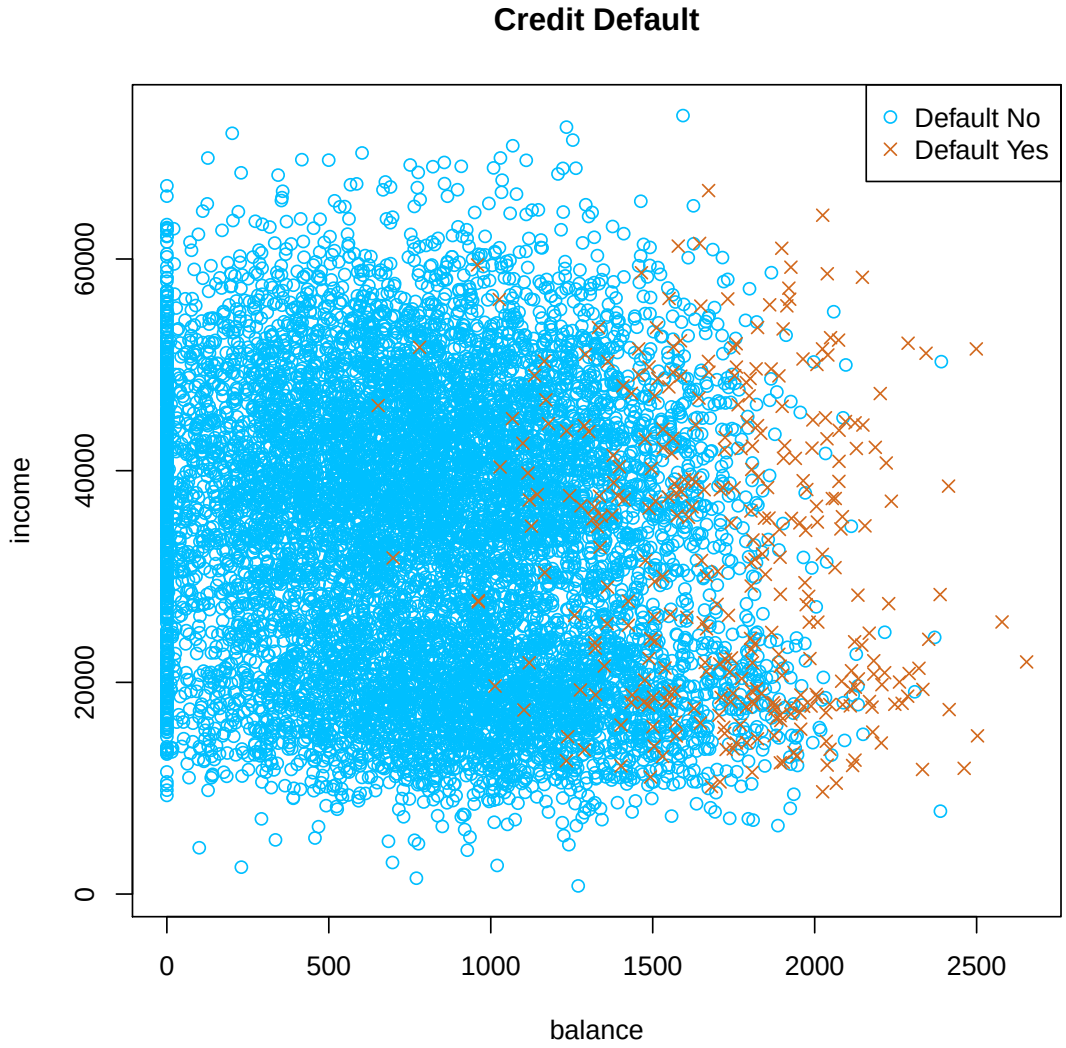
and errors are assumed to be normal

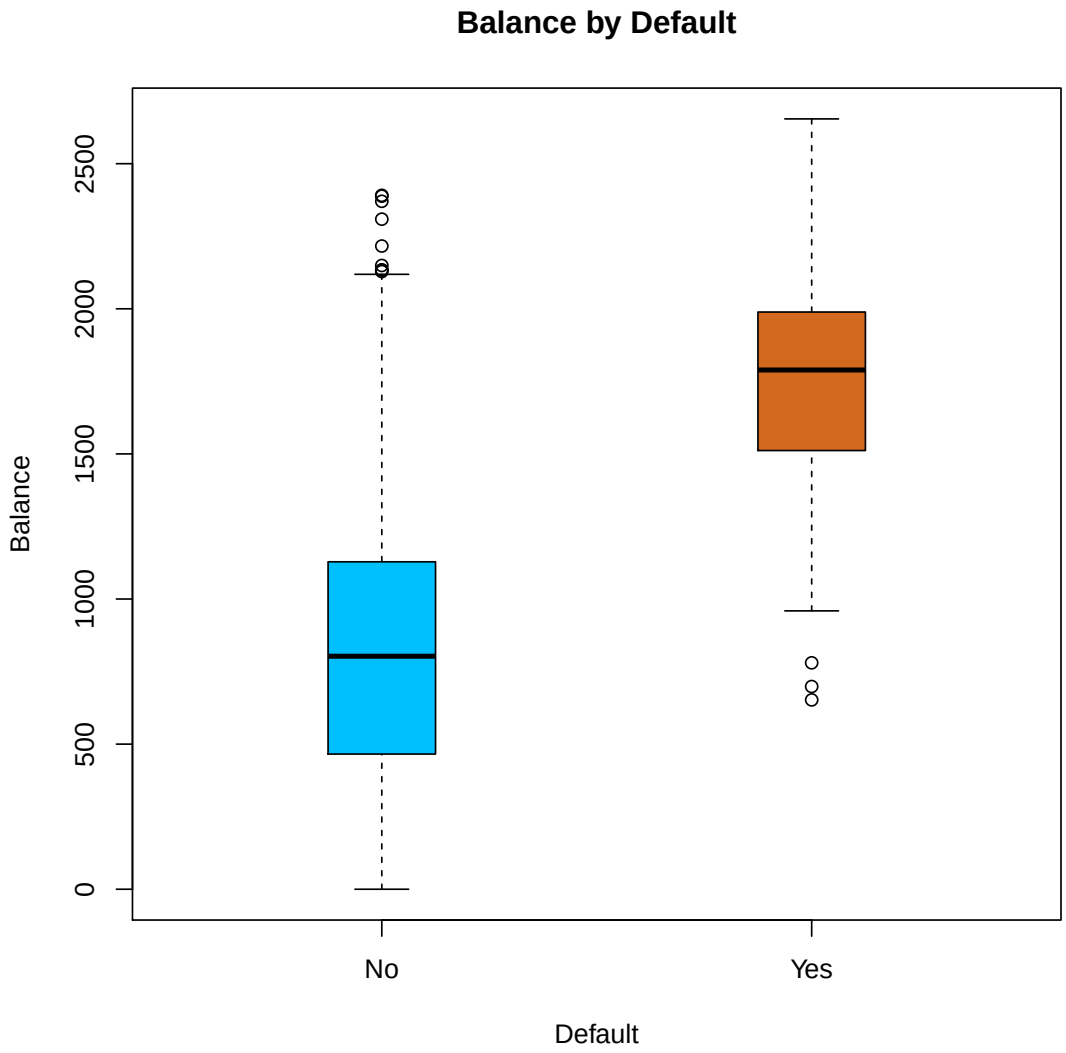
$$\varepsilon_i = \text{Normal}(0, \sigma^2)$$

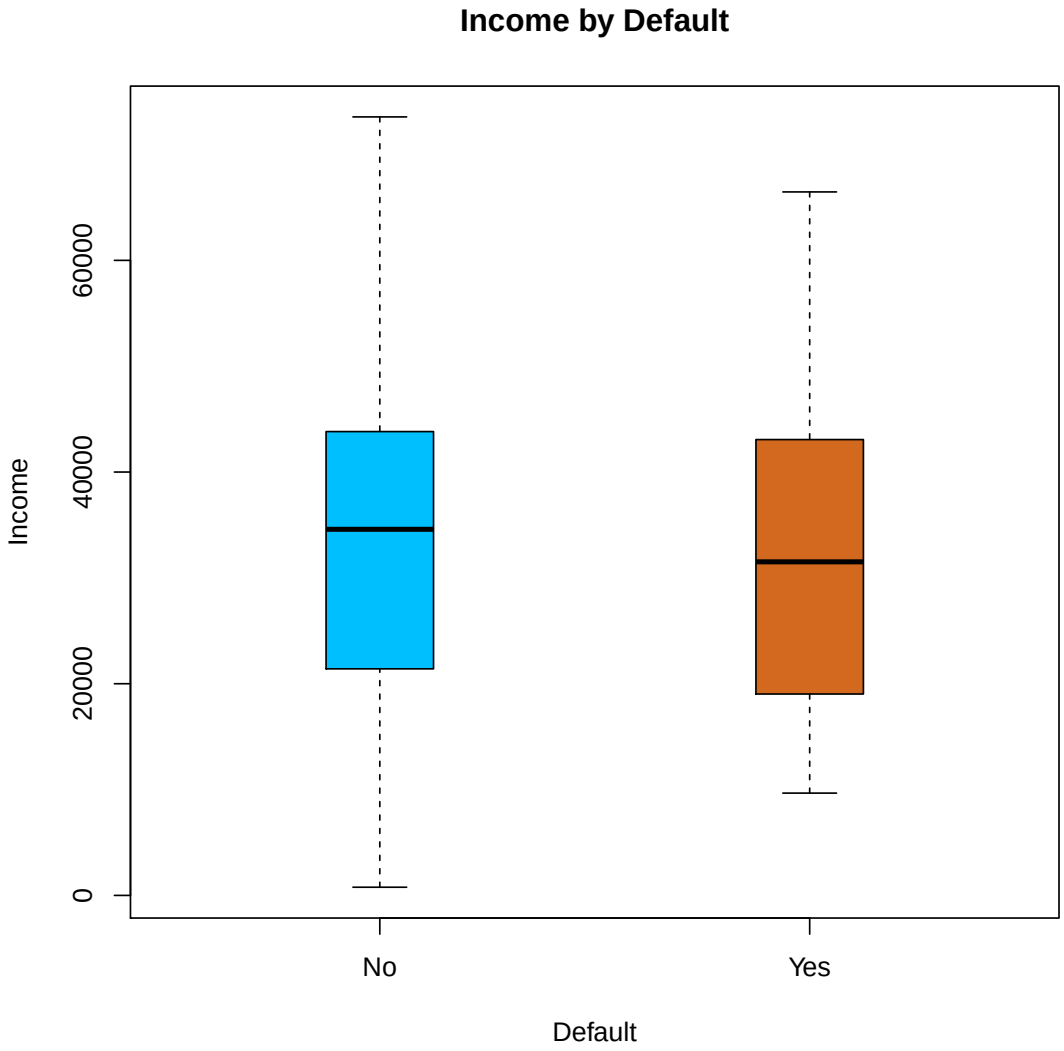
- We are ready to deviate from those two fundamental aspects of our model.
- Consider the following dataset to see how the first may take a different form

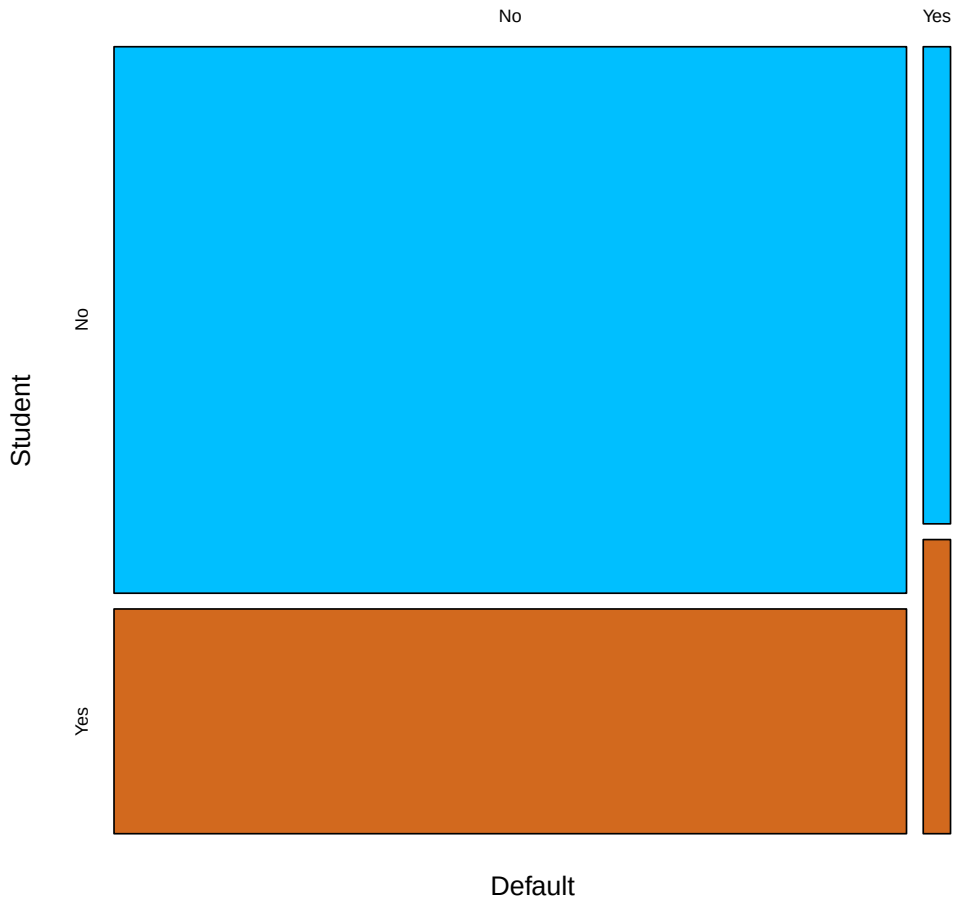
`Income`	Annual income
`Balance`	Credit card balance
`Default`	Whether the card holder has defaulted
`Student`	Whether the card holder is a student

- Suppose a bank is interested in predicting `Dafault`.




Balance by Default
Balance
0
500
1000
1500
2000
2500
No
Yes
Default


Income by Default
Income
0
20000
40000
60000
No
Yes
Default


mosaicplot of Default by Student
No
Yes
Student
No
Yes
Default

Q: Why a linear regression is not going to be useful/meaningful here? e.g.

$$y_i = \beta_0 + \beta_1 x_i + e_i$$

where X corresponds to `Balance` and Y corresponds `Default`

$$Y = \begin{cases} 1 & \text{if } \texttt{Default=No}; \\ 2 & \text{if } \texttt{Default=Yes}. \end{cases}$$

- Recall a simple linear regression models the conditional mean

$$\mathbb{E}\left[Y \mid X = x\right] = \beta_0 + \beta_1 x$$

by finding $\hat{\beta}_0$ and $\hat{\beta}_1$ for β_0 and β_1, and we essentially use the estimate

$$\hat{Y} = \hat{\beta}_0 + \hat{\beta}_1 x$$

to predict the conditional mean for every possible value of $X = x$.

- However, here `Default` is a categorical variable, the conditional mean

$$\mathbb{E}\left[Y \mid X = x\right] = \beta_0 + \beta_1 x$$

  is not meaningful to take any value between 1 and 2

- If we change Y for the binary variable, we see what is really needed

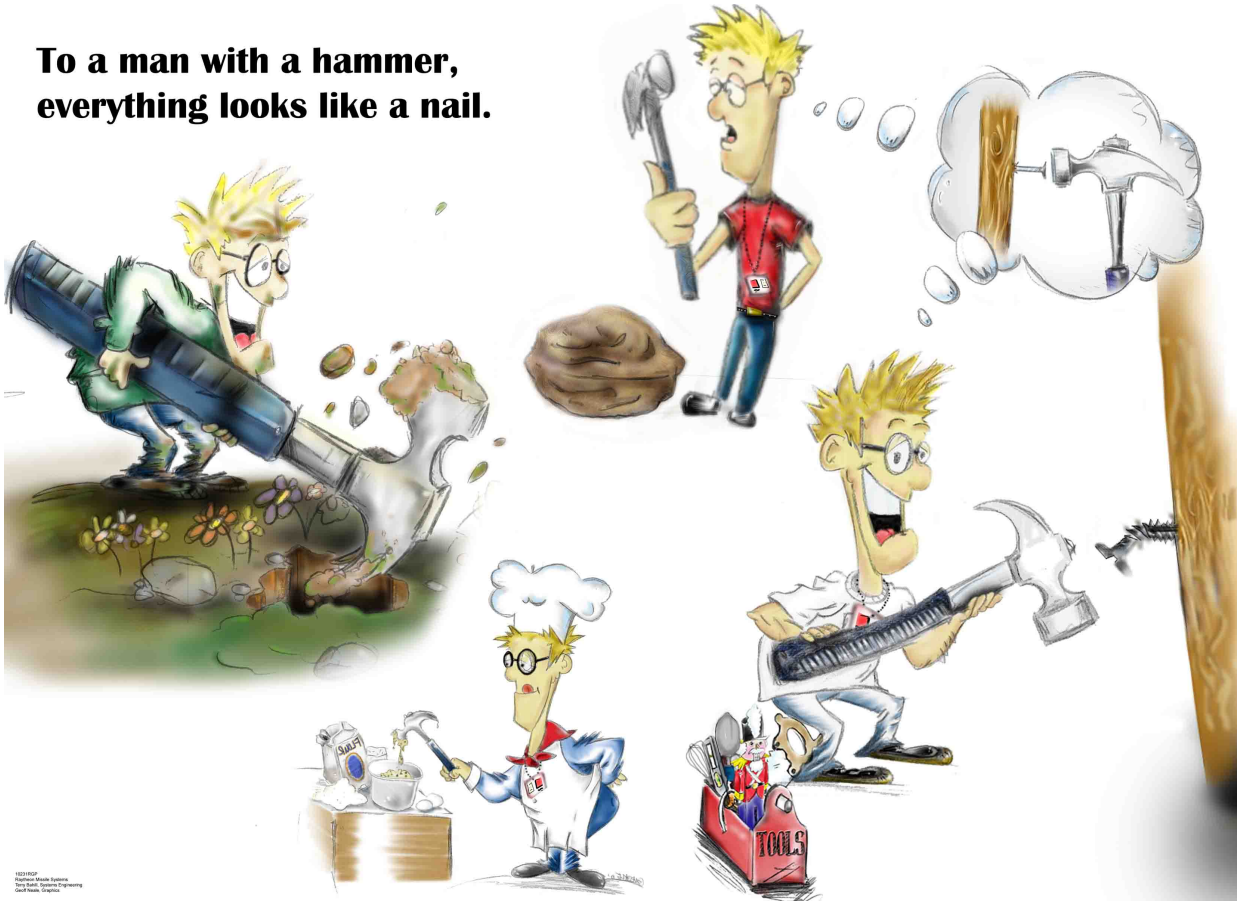
$$Y = \begin{cases} 0 & \text{if } \texttt{Default=No}; \\ 1 & \text{if } \texttt{Default=Yes}. \end{cases}$$

  since the conditional mean is also the conditional probability

$$\Pr\left(Y = 1 \mid X = x\right) = \mathbb{E}\left[Y \mid X = x\right]$$

- However, linear regression will not restrict $\hat{y}$ to be between $[0, 1]$ in general.
- We could indulge in a discussion on constrained optimisation problem.

- The real problem is not we don't have a more sophisticated hammer,



it is we don't have a nail to start with.

- To avoid having an estimated probability outside of $[0, 1]$, we model

$$\Pr(Y = 1 \mid X = x)$$

using a function that has the range $[0, 1]$ instead of using

$$\Pr(Y = 1 \mid X = x) = \mathbb{E}[Y \mid X = x] = \beta_0 + \beta_1 x$$

- There are many functions that meet this requirement, if the logistic function

$$\Pr(Y = 1 \mid X = x) = \frac{\exp(\beta_0 + \beta_1 x)}{1 + \exp(\beta_0 + \beta_1 x)}$$

is used, then we will end up with so-called logistic regression.

Q: Can you see why the logistic function is a natural choice?

Q: Can you see why logistics regression is a generalised linear regression?

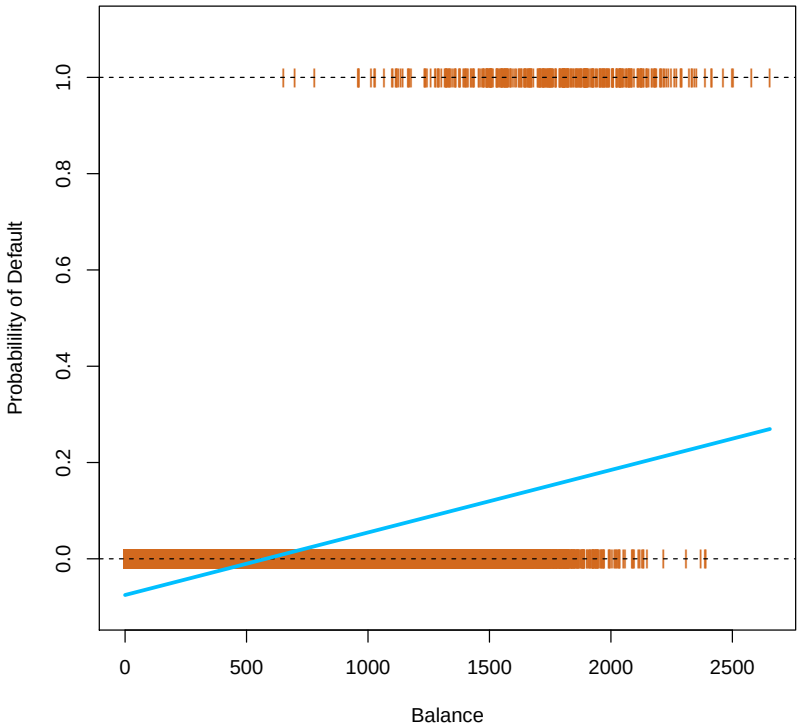
Q: Conditional probability is modelled instead of mean, how about normality?

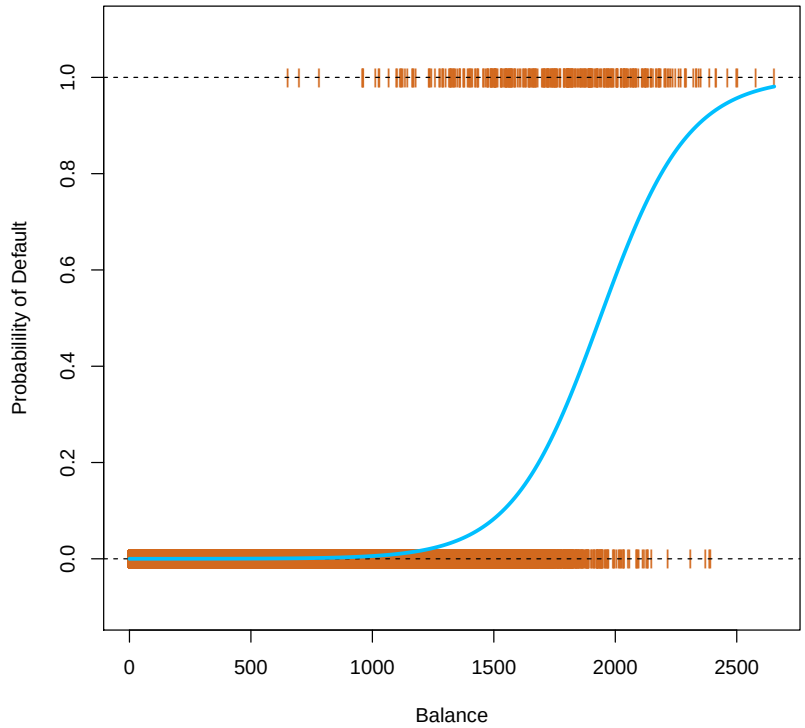
- Once this parametric form is decided, β_0 and β_1 can be estimated using the

principle of maximum likelihood

$$\hat{\boldsymbol{\beta}} \in \left\{ \arg\max_{\boldsymbol{\beta}} \mathcal{L}\left(\boldsymbol{\beta}; x, y\right) \right\}$$

- Using the maximum likelihood estimate (MLE) of β_0 and β_1, we will have





Q: How to obtain MLEs $\hat{\beta}_0$ and $\hat{\beta}_1$? What is the likelihood function here?

$$\mathcal{L}(\beta_0, \beta_1) = \prod_{i=1}^{n} \Pr(Y = y_i)$$

- Under the assumption that

$$\Pr(Y = 1 \mid X = x) = \frac{\exp(\beta_0 + \beta_1 x)}{1 + \exp(\beta_0 + \beta_1 x)}$$

the log-likelihood function is given by

$$\begin{aligned}
\ell(\beta_0, \beta_1) &= \log \mathcal{L}(\beta_0, \beta_1) \\
&= \log \prod_{i=1}^{n} \big(\Pr(Y = 1 \mid X = x_i)\big)^{y_i} \big(\Pr(Y = 0 \mid X = x_i)\big)^{1-y_i} \\
&= \sum_{i=1}^{n} \left[-\log\left(1 + e^{\beta_0 + \beta_1 x_i}\right) + y_i(\beta_0 + \beta_1 x_i)\right]
\end{aligned}$$

- We obtain two nonlinear equations when setting the first derivatives to zero

$$\sum_{i=1}^{n}\left(y_i - \frac{\exp\left(\beta_0+\beta_1 x_i\right)}{1+\exp\left(\beta_0+\beta_1 x_i\right)}\right) = 0$$

$$\sum_{i=1}^{n} x_i\left(y_i - \frac{x_i\exp\left(\beta_0+\beta_1 x_i\right)}{1+\exp\left(\beta_0+\beta_1 x_i\right)}\right) = 0$$

- Of course, R will automatically fit and solve the nonlinear equations for us

```
> credit.LG = glm(default~balance, family = binomial,
+                 data = credit.df)
>
> coef(credit.LG)
```

```
  (Intercept)       balance
-10.651330614   0.005498917
```

- Of course, we base our prediction on our estimates, .e.g.

$$\hat{\Pr}(Y = 1 \mid X = 1000) = \hat{\pi} = \frac{\exp\left(\hat{\beta}_0 + \hat{\beta}_1 x\right)}{1 + \exp\left(\hat{\beta}_0 + \hat{\beta}_1 x\right)}$$

$$= \frac{\exp\left(-10.6513 + 0.0055 \cdot 1000\right)}{1 + \exp\left(-10.6513 + 0.0055 \cdot 1000\right)}$$

$$= 0.00576$$

- Of course, this can be easily extended to more than one predictor

$$\hat{\pi} = \frac{\exp\left(\hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2 + \hat{\beta}_3 x_3\right)}{1 + \exp\left(\hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2 + \hat{\beta}_3 x_3\right)}$$

```
> credit.all3.LG = glm(
+   default~balance+income+student, family = binomial,
+   data = credit.df)
```

```
> summary(credit.all3.LG)
```

```
Call:
glm(formula = default ~ balance + income + student, family = binomial,
    data = credit.df)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.4691  -0.1418  -0.0557  -0.0203   3.7383

Coefficients:
              Estimate Std. Error z value Pr(>z)
(Intercept) -1.087e+01  4.923e-01 -22.080  < 2e-16 ***
balance      5.737e-03  2.319e-04  24.738  < 2e-16 ***
income       3.033e-06  8.203e-06   0.370  0.71152
studentYes  -6.468e-01  2.363e-01  -2.738  0.00619 **
---
Signif. codes:  0 ?***? 0.001 ?**? 0.01 ?*? 0.05 ?.? 0.1 ? ? 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2920.6  on 9999  degrees of freedom
Residual deviance: 1571.5  on 9996  degrees of freedom
AIC: 1579.5

Number of Fisher Scoring iterations: 8
```

- It is similar what we had before, but inference and diagnostics are different.

- Diagnostics for logistics regression models are very different

  Pearson residuals

$$\hat{e}_i = \frac{s_i - n_i\hat{\pi}_i}{\sqrt{n_i\hat{\pi}_i\left(1-\hat{\pi}_i\right)}}$$

  where $\hat{\pi}_i$ is the "fitted valued", that is, the estimated probability of

$$\Pr\left[Y_i = 1 \mid \mathbf{X}_i = \mathbf{x}_i\right]$$

  and s_i is the number of observations belonging to $Y = 1 \cap \mathbf{X} = \mathbf{x}_i$ and n_i is the number of observations belonging to $\mathbf{X} = \mathbf{x}_i$.

- Pearson residuals are similar to standardised residuals for MLR.

Q: Can you see the similarity?

$$\hat{e}_i' = \frac{y_i - \hat{y}_i}{\hat{\sigma}\sqrt{1-p_{ii}}}$$

- The only assumption that we are making regarding logistic regression is

$$\Pr\left[Y_i = 1 \mid \mathbf{X}_i\right] = \pi_i = \frac{\exp\left(\beta_0 + \beta_1 x_{i1} + \cdots + \beta_k x_{ik}\right)}{1 + \exp\left(\beta_0 + \beta_1 x_{i1} + \cdots + \beta_k x_{ik}\right)}$$

$$\Longrightarrow \ln\left(\frac{\pi_i}{1-\pi_i}\right) = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_k x_{ik}$$

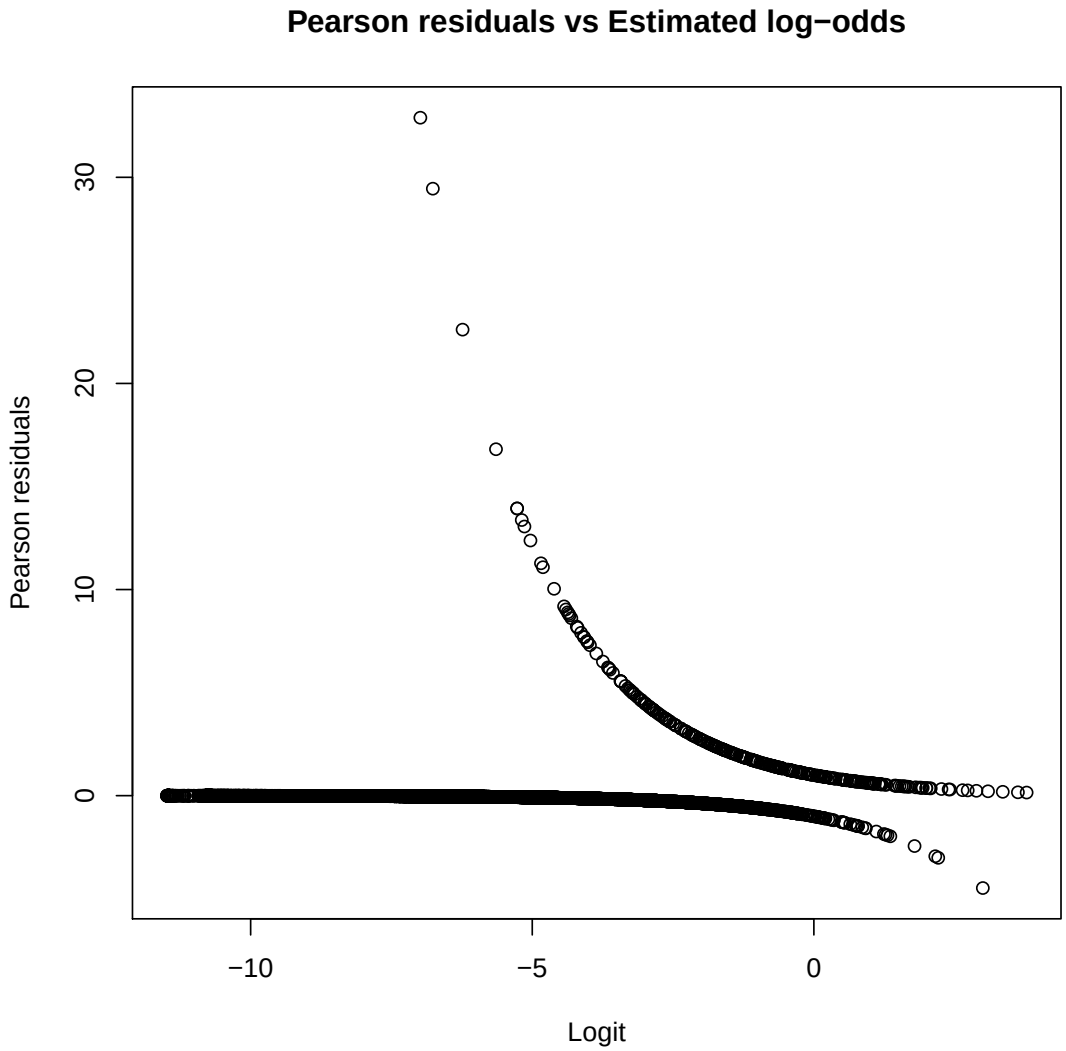
- However, the residuals

$$\hat{e}_i = \frac{s_i - n_i\hat{\pi}_i}{\sqrt{n_i\hat{\pi}_i\left(1-\hat{\pi}_i\right)}}$$

  are most useful for datasets with many observations having the same $\mathbf{x}_i$.

- If most of $\mathbf{x}_i$ are unique, then the residual is large if the observation is a success, $Y = 1$, but the model gives it a low probability of success.

- Not useful at all!



- Consider the following dataset

```
> ingots.df
```

	heat	soak	notready	total
1	7	1.0	0	10
2	14	1.0	0	31
3	27	1.0	1	56
4	51	1.0	3	13
5	7	1.7	0	17
6	14	1.7	0	43
7	27	1.7	4	44
8	51	1.7	0	1
9	7	2.2	0	7
10	14	2.2	2	33
11	27	2.2	0	21
12	51	2.2	0	1
13	7	2.8	0	12
14	14	2.8	0	31
15	27	2.8	1	22
16	51	2.8	0	0
17	7	4.0	0	9
18	14	4.0	0	19
19	27	4.0	1	16
20	51	4.0	0	1

- Fitting logistic regression to this dataset, we have

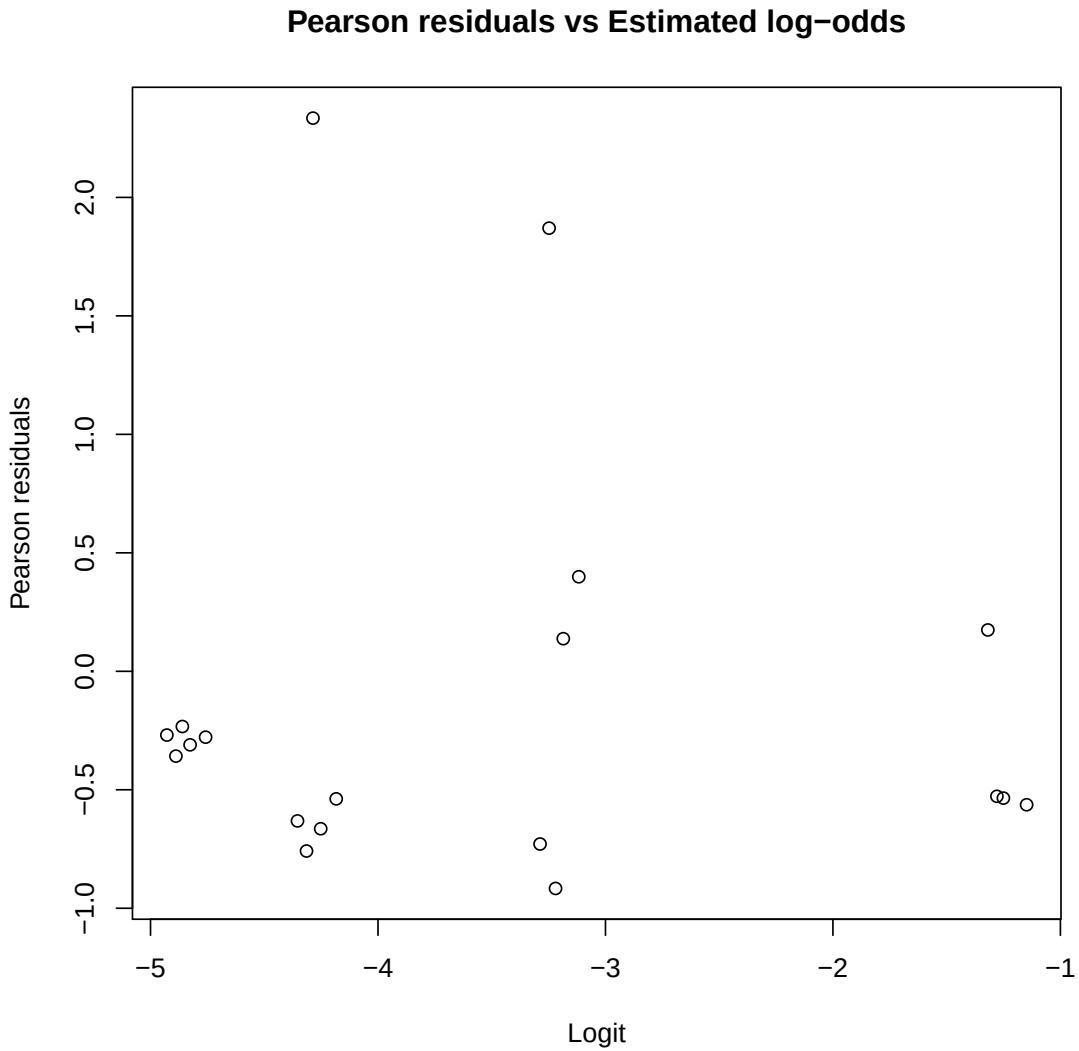
```
> ingots.LG = glm(notready/total~heat+soak,
+                 family=binomial,data=ingots.df,
+                 weight=total)

> summary(ingots.LG)
```

```
Call:
glm(formula = notready/total ~ heat + soak, family = binomial,
    data = ingots.df, weights = total)

Coefficients:
            Estimate Std. Error z value Pr(>z)
(Intercept) -5.55917    1.11969  -4.965 6.87e-07 ***
heat         0.08203    0.02373   3.456 0.000548 ***
soak         0.05677    0.33121   0.171 0.863906
---
Signif. codes:  0 ?***? 0.001 ?**? 0.01 ?*? 0.05 ?.? 0.1 ? ? 1
```

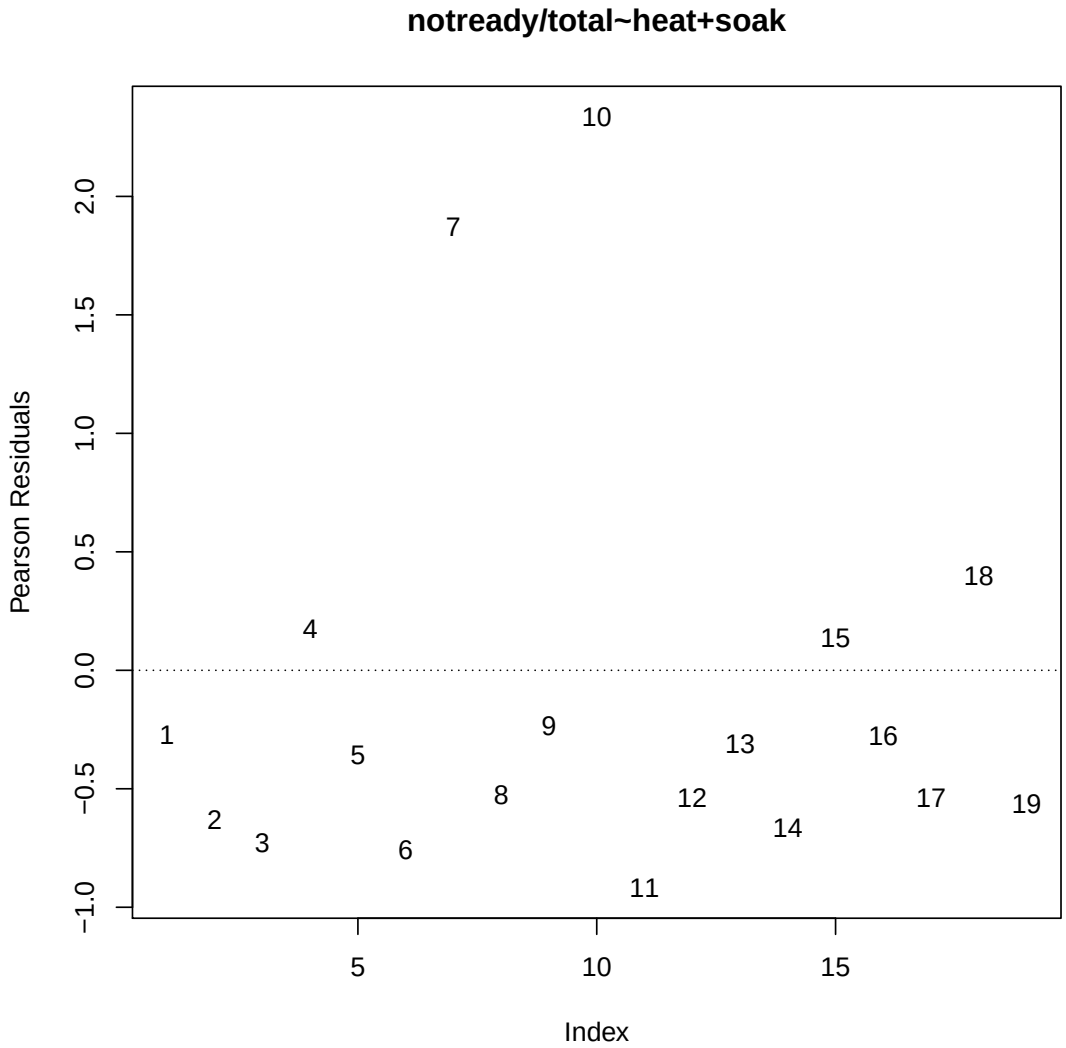
- In this case, we can try checking the residuals and get some feedback

```
> plot(residuals(ingots.LG, type="pearson"),
+      type="n", ylab = "Pearson Residuals",
+      main="notready/total~heat+soak")
> text(res); abline(h=0,lty=3)
```

- Since the dataset is very small, it is not very informative



- Observation 7 and 10 are two possible outliers.

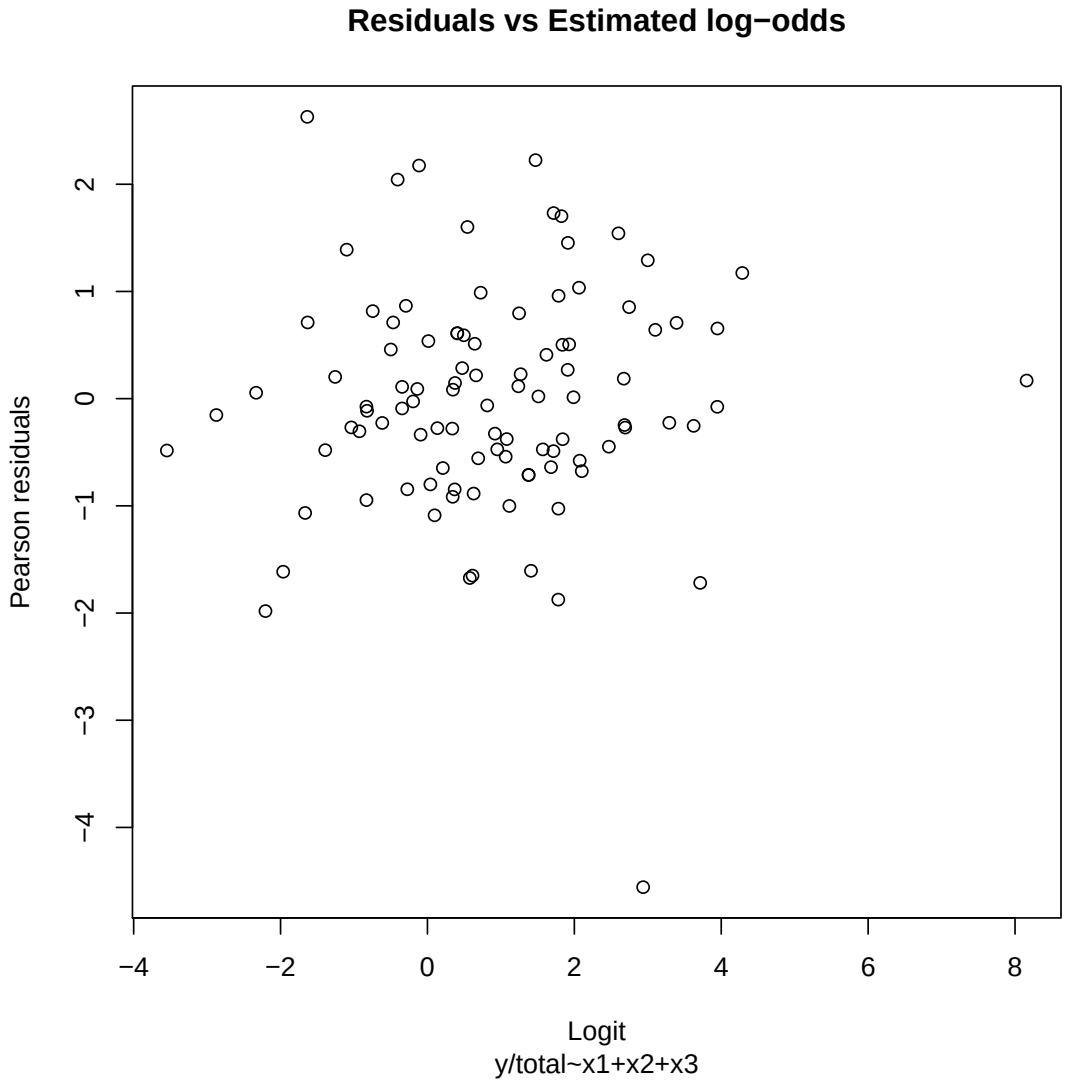


- Consider the following simulation study

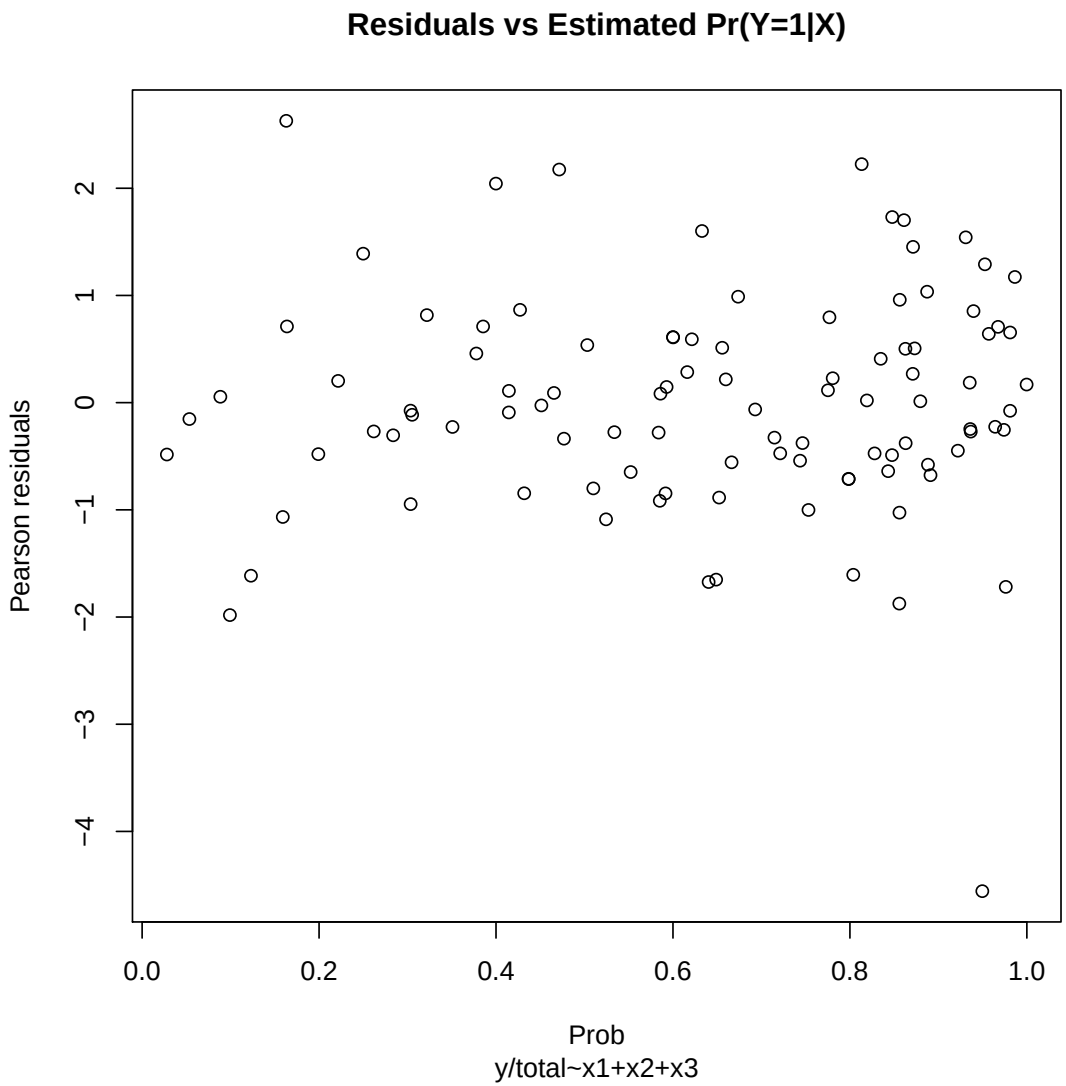
```
> set.seed(1)
> n = 100
>
> x1 = rnorm(n, mean = 0, sd = 6)
> x2 = rt(n, df = 1)
> x3 = rbinom(n, size = 1, prob = 0.45)

> beta0 = 0.6
> beta1 = 0.3
> beta2 = 0.1
> beta3 = 0.1
>
> logit = beta0 + beta1*x1 + beta2*x2 + beta3*x3

> true.pi = exp(logit)/(1+exp(logit))
```

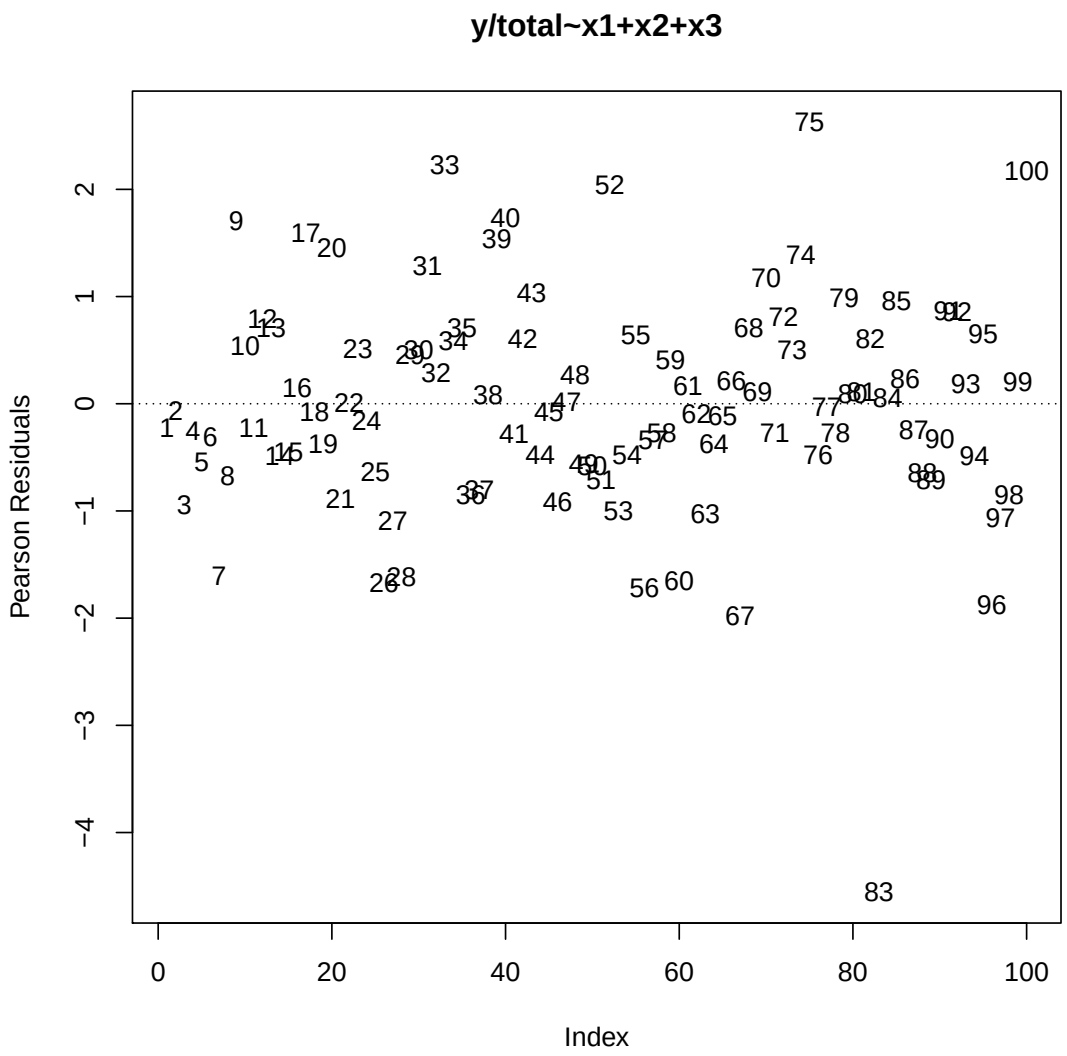
```
> m = 100
>
> y = integer(n)
> for (i in 1:n){
+   y[i] = rbinom(1, size = m, prob = true.pi[i])
+ }

> sim.df = data.frame(
+   y = y, x1 = x1, x2 = x2, x3 = x3,
+   total = rep(m, n))

> sim.LG = glm(y/total~x1+x2+x3, family = binomial,
+              weights = total, data = sim.df)

> res = residuals(sim.LG, type="pearson")
>
> logit = predict(sim.LG) # default logit
>
> prob = predict(sim.LG, type = "response") # Prob
```

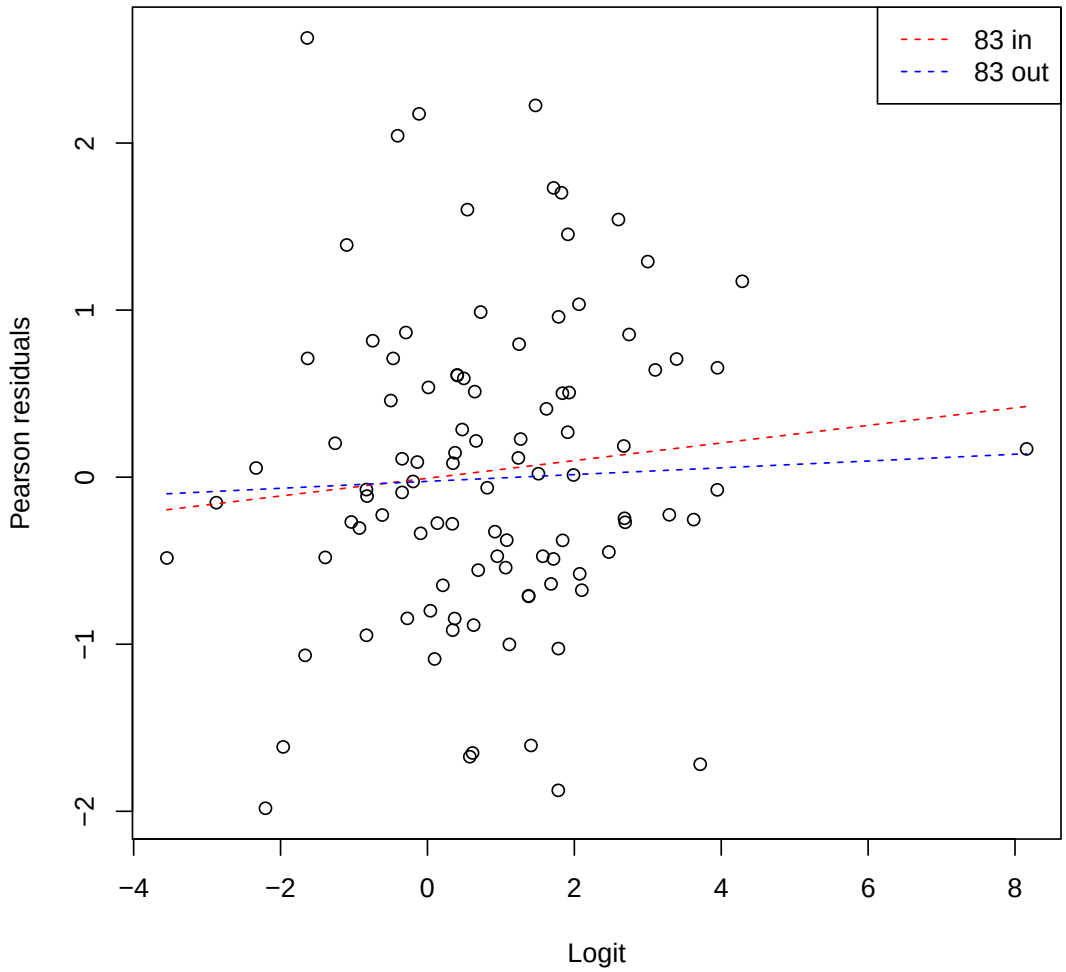
```
> plot(logit, res, sub = "y/total~x1+x2+x3"
+       xlab="Logit", ylab="Pearson residuals",
+       main="Residuals vs Estimated log-odds")
```



```
> plot(prob, res, sub = "y/total~x1+x2+x3"
+       xlab="Prob", ylab="Pearson residuals",
+       main="Residuals vs Estimated Pr(Y=1|X)")
```




y/total~x1+x2+x3
Pearson Residuals
Index
2
1
0
−1
−2
−3
−4
0
20
40
60
80
100


83 in
83 out
Pearson residuals
Logit
−4
−2
0
2
4
6
8
−2
−1
0
1
2

```
> plot(res, type="n",
+       ylab = "Pearson Residuals",
+       main="y/total~x1+x2+x3")
>
> text(logit); abline(h=0,lty=3)
>
> plot(logit[-c(83)], res[-c(83)],
+       xlab="Logit", ylab="Pearson residuals")
>
> lines(smooth.spline(logit[-c(83)], res[-c(83)]),
+        col = 2, lty = 2)
>
> lines(smooth.spline(logit, res), col = 4, lty = 2)
>
> legend("topright", c("83 in", "83 out"),
+         col = c(2, 4), lty = 2)
```

- Checking high leverage points

```
> head(sort(hatvalues(sim.LG), decreasing = TRUE))
```

```
        21         46         62         36         71         45
0.37676714 0.15760469 0.13174284 0.07640166 0.06673067 0.06273533
```

- Check influential points

```
> head(sort(cooks.distance(sim.LG),
+               decreasing = TRUE))
```

```
        21         83         75         67         52         46
0.19014483 0.11674260 0.08128077 0.06087118 0.05277872 0.04652510
```

- 83 is an influential outlier and 21 is a high leverage influential point, thus

```
> sim.final.LG =
+    glm(y/total~x1+x2+x3, family=binomial,
+          weights = total, data=sim.df[-c(83, 21),])
```

```
> summary(sim.final.LG)
```

```
Call:
glm(formula = y/total ~ x1 + x2 + x3, family = binomial, data = sim.df[-c(83,
    21), ], weights = total)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2.06696  -0.59007  -0.07006   0.57910   2.56599

Coefficients:
            Estimate Std. Error z value Pr(>z)
(Intercept) 0.593201   0.033575  17.668   <2e-16 ***
x1          0.303756   0.006812  44.594   <2e-16 ***
x2          0.100487   0.010559   9.517   <2e-16 ***
x3          0.129673   0.051550   2.515   0.0119 *
---
Signif. codes:  0 ?***? 0.001 ?**? 0.01 ?*? 0.05 ?.? 0.1 ? ? 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 3474.622  on 97  degrees of freedom
Residual deviance:   87.842  on 94  degrees of freedom
AIC: 521.32

Number of Fisher Scoring iterations: 4
```

```
> c(beta0, beta1, beta2, beta3)
```

```
[1] 0.6 0.3 0.1 0.1
```

- If most of $\mathbf{x}_i$ are unique, we have to reply on test to check goodness-of-fit

```
> library(ResourceSelection)
> hoslem.test(credit.all3.LG$y,
+                 fitted(credit.all3.LG))
```

```
Hosmer and Lemeshow goodness of fit (GOF) test

data:  credit.all3.LG$y, fitted(credit.all3.LG)
X-squared = 3.6823, df = 8, p-value = 0.8846
```

  this is known as Hosmer-Lemeshow test, it examines whether the observed proportion is consistent with predicted probability using a Pearson χ^2-test.

- Large p-values indicate a good fit to the data.

- `> hoslem.test(ingots.LG$y, fitted(ingots.LG))`

```
        Hosmer and Lemeshow goodness of fit (GOF) test

data:  ingots.LG$y, fitted(ingots.LG)
X-squared = 0.86523, df = 8, p-value = 0.999
```

- The idea of Hosmer-Lemeshow is to form groups of observed and expected π

```
> # prob = predict(credit.all3.LG, type="response")
> prob = fitted(credit.all3.LG)

> n.group = 10
>
> p = seq(from = 0.1, to = 0.9,
+         length.out = n.group - 1)

> mid = quantile(prob, p)

> b.vec = c(0, mid, 1)

> prob.cut = cut(prob, breaks = b.vec)
> table(prob.cut)
```

```
prob.cut
       (0,5.14e-05] (5.14e-05,0.000176] (0.000176,0.000443] (0.000443,0.000945]
               1000                1000                1000                1000
 (0.000945,0.00197]   (0.00197,0.00402]    (0.00402,0.0088]      (0.0088,0.021]
               1000                1000                1000                1000
    (0.021,0.0709]          (0.0709,1]
               1000                1000
```

```
> prob.cut = cut(prob, breaks = b.vec,
+                labels = FALSE)
> table(prob.cut)
```

```
prob.cut
   1    2    3    4    5    6    7    8    9   10
1000 1000 1000 1000 1000 1000 1000 1000 1000 1000
```

```
> E = matrix(0, nrow = 2, ncol = n.group)
> O = matrix(0, nrow = 2, ncol = n.group)
> for (j in 1:n.group){
+   E[1, j] = sum(prob[prob.cut == j])
+   E[2, j] = sum((1-prob)[prob.cut == j])
+   O[1, j] = sum(credit.all3.LG$y[prob.cut == j])
+   O[2, j] = sum((1-credit.all3.LG$y)[prob.cut == j]
+ }
```

- Notice we obtain the same test-statistics and p-value

```
> 1 - pchisq(sum((O-E)^2/E), n.group - 2)
```

```
[1] 0.8845912
```

- When the dataset is large enough,

```
> summary(credit.all3.LG)
```

```
Call:
glm(formula = default ~ balance + income + student, family = binomial,
    data = credit.df)

Coefficients:
              Estimate Std. Error z value Pr(>z)
(Intercept) -1.087e+01  4.923e-01 -22.080  < 2e-16 ***
balance      5.737e-03  2.319e-04  24.738  < 2e-16 ***
income       3.033e-06  8.203e-06   0.370  0.71152
studentYes  -6.468e-01  2.363e-01  -2.738  0.00619 **
---
Signif. codes:  0 ?***? 0.001 ?**? 0.01 ?*? 0.05 ?.? 0.1 ? ? 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2920.6  on 9999  degrees of freedom
Residual deviance: 1571.5  on 9996  degrees of freedom
AIC: 1579.5

Number of Fisher Scoring iterations: 8
```

Wald-test can be used to decide whether a variable is significant.

- It can be shown that for MLE $\hat{\boldsymbol{\theta}}$ of the unknown true parameter vector $\boldsymbol{\theta}$

$$\hat{\boldsymbol{\theta}} \overset{a}{\sim} \text{Normal}\left(\boldsymbol{\theta}, \mathbf{V}\right)$$

where $\overset{a}{\sim}$ denotes asymptotically distributed and

$$\mathbf{V}$$

is the variance-covariance matrix and is a function of $\boldsymbol{\theta}$ and simple size

- Since $\mathbf{V}$ is not observed, it needs to be estimated, and the negative Hessian

$$-\mathbf{H}$$

of the log-likelihood function evaluated at $\hat{\boldsymbol{\theta}}$ is often used to estimated $\mathbf{V}$.

$$\hat{\mathbf{V}} = -\mathbf{H}^{-1}$$

- This approximate normality is utilised in the construction of Wald tests.