Regularized Logistic Regression

Rob McCulloch

1. Logistic Regression
2. The Logistic Likelihood
3. L2 and L1 Regularized Logistic Regression
4. Simulated Example
5. We8There
6. Multinomial Logit

1. Logistic Regression

Logistic regression is a key model.

It allows us to use ideas from linear modeling to predict a binary outcome.

The model is:

$$P(Y = 1 \mid x) = F(x'\beta), \quad F(\eta) = \frac{e^{\eta}}{1 + e^{\eta}} = \frac{1}{1 + e^{-}\eta}.$$

As in linear regression, $x'\beta$ is a linear combination of the features which may include a 1 for an intercept.

Note that $F : R \to (0, 1)$.

F is called the logistic function or the sigmoid function.

To start off as simply as possible, we will first consider the case where we have a binary y and one numeric x.

Lets' look at the Default data (from the ISLR book):

- y:
  whether or not a customer defaults on their credit card (No or Yes).

- x:
  The average balance that the customer has remaining on their credit card after making their monthly payment.

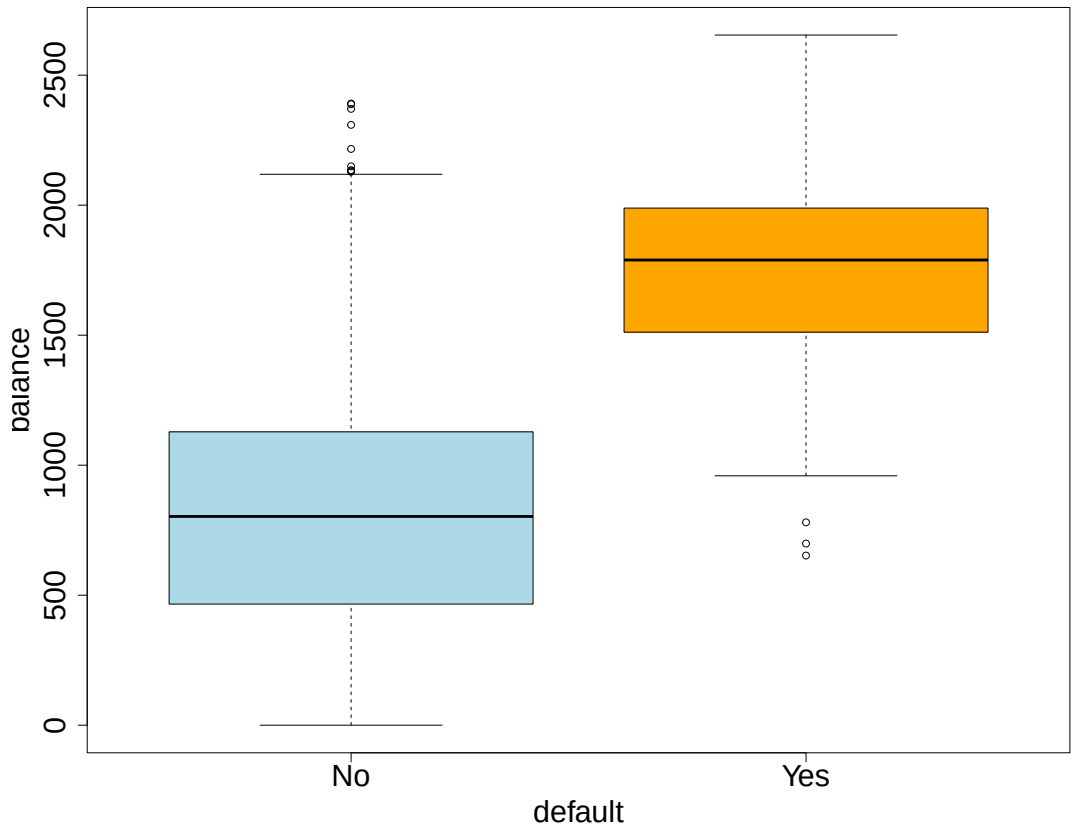
- 10,000 observations, 333 defaults (.0333 default rate).

Let's look at the data.

Divide the data into two groups, one group has y=No and other other group has y=Yes.

Use boxplots to display the x=balance values in each subgroup.

The balance values are bigger for the default y=Yes observations!



Logistic regression uses the power of linear modeling and estimates $Pr(Y = y \mid x)$ by using a two step process:

- Step 1:

  apply a linear function to x: $x \rightarrow \eta = \beta_0 + \beta_1 x$.

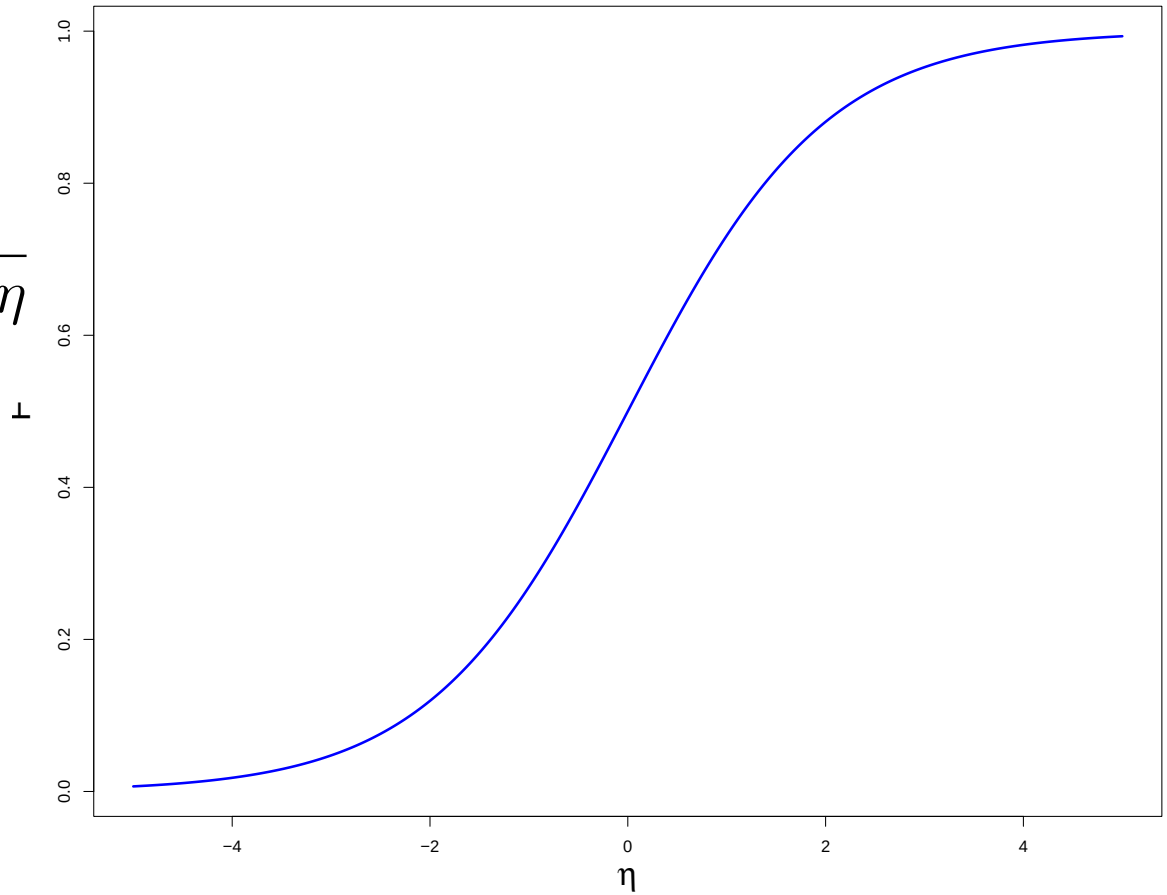
- Step 2:

  apply the *logistic function* F,
  to η to get a number between 0 and 1.

  $P(Y = 1 \mid x) = F(\eta)$.

The logistic function:

$$F(\eta) = \frac{e^{\eta}}{1+e^{\eta}} = \frac{1}{1+e^{-\eta}}$$



The key idea is that $F(\eta)$ is always between 0 and 1 so we can use it as a probability.

Note that F is increasing, so if η goes up $P(Y = 1 \mid x)$ goes up.

$$F(-3) = .05,\ F(-2) = .12,\ F(-1) = .27$$

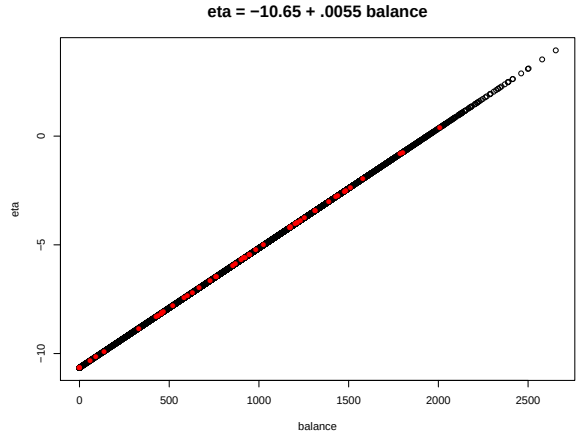
$$F(0) = .5$$

$$F(1) = .73,\ F(2) = .88,\ F(3) = .95$$

Logistic fit to the y=default, x=balance data.

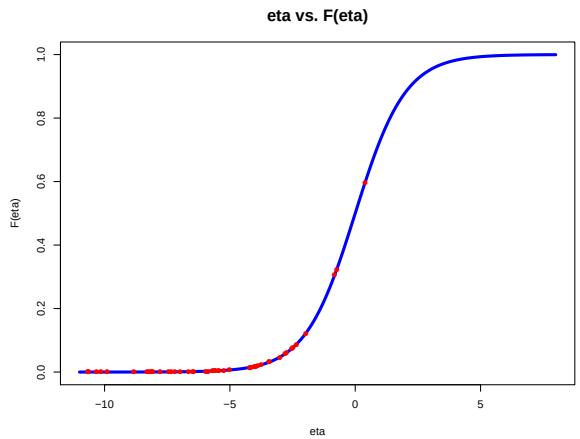
First, logistic looks for a linear function of x it can feed into the logistic function.

Here we have
$\eta = -10.65 + .0055\,x$.

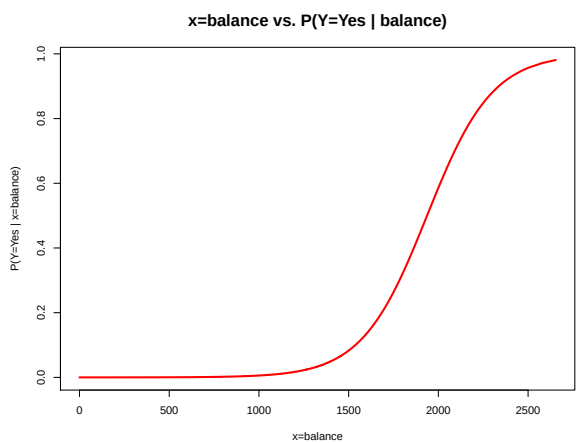


Next we feed the η values into the logistic function.

100 randomly sampled observations are plotted with red dots.



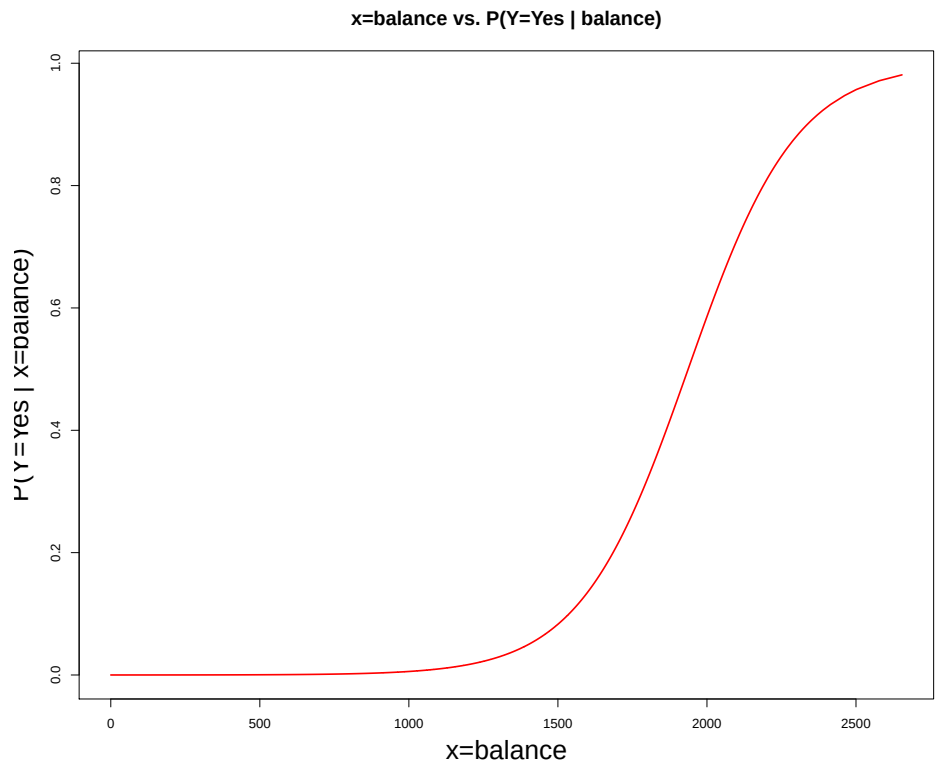
We can combine the two steps together and plot
x=balance vs.
$P(Y = Yes \mid x) = F(-10.65 + .0055\,x)$.



Logistic Regression:

Combining the two steps, our logistic regression model is:

$$P(Y = 1 \mid X = x) = F(\beta_0 + \beta_1 x).$$



Multiple Logistic Regression

We can extend our logistic model to several numeric x by letting η be a linear combination of the x's instead of just a linear function of one x:

- Step 1:
  $\eta = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots \beta_p x_p = x' \beta.$
- Step 2:
  $P(Y = 1 \mid x = (x_1, x_2, \ldots, x_p)) = F(\eta).$

Or, in one step, our model is:

$$Y_i \sim \text{Bernoulli}(p_i), \quad p_i = F(x' \beta).$$

Our first step keeps some of the structure we are used to in linear regression.

We combine the x's together into one weighted sum that we hope will capture all the information they provide about y.

We then turn the combination into a probability by applying F.

Log Odds

Our model is:

$$p(Y = 1 \,|\, x, \beta) = \frac{e^{x'\beta}}{1 + e^{x'\beta}}.$$

So,

$$\log\left(\frac{p(Y = 1 \,|\, x, \beta)}{1 - p(Y = 1 \,|\, x, \beta)}\right) = x' \beta.$$

The log of the odds ratio Y=1 vs Y=0, is linear in x.

For example:

$$P(Y = 1|x, \beta) = .5 \iff \text{log odds} = 0 \iff x' \beta = 0.$$

The Default Data, More than One x

Here is the logistic regression output using all three x's in the data set: balance, income, and student.
student is coded up as a factor, so R automatically turns it into a dummy.

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

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 ***
studentYes  -6.468e-01  2.363e-01  -2.738  0.00619 **
income       3.033e-06  8.203e-06   0.370  0.71152
---
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
```

Note:

```
Number of Fisher Scoring iterations: 8
```

The estimates are MLE.

The maximization uses Newton's method which is an iterative procedure.

It took 8 iterations for the maximization to converge !!!

The estimates are MLE.

Confidence intervals are estimate +/- 2 standard errors.
e.g for `studentYes` coefficient : -.65 +/- 2(.24) = -.65 +/- .5

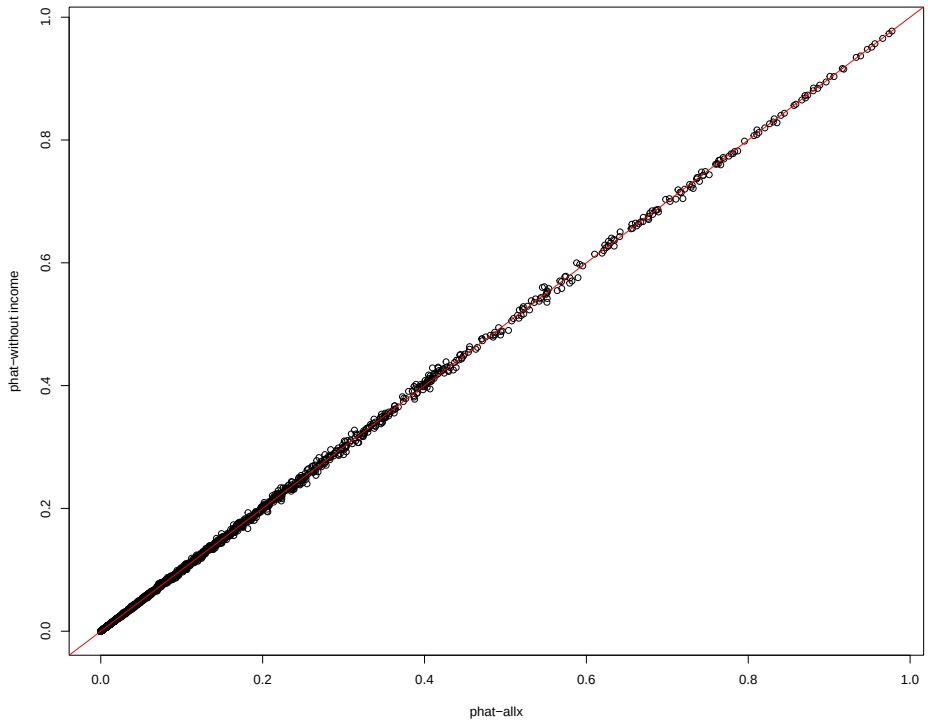
Z-stats are (estimate-proposed)/se.

To test whether the coefficient for income is 0, we have z = (3.033-0)/8.203 = .37, so we fail to reject.

The p-value is 2*P(Z < -.37) = 2*pnorm(-.37) = 0.7113825.

So, the output suggests we may not need `income`.

Here is a plot of the fitted probabilities with and without `income` in the model.



We get almost the same probabilities, so, as a practical matter, income does not change the fit.

Here is the output using `balance` and `student`.

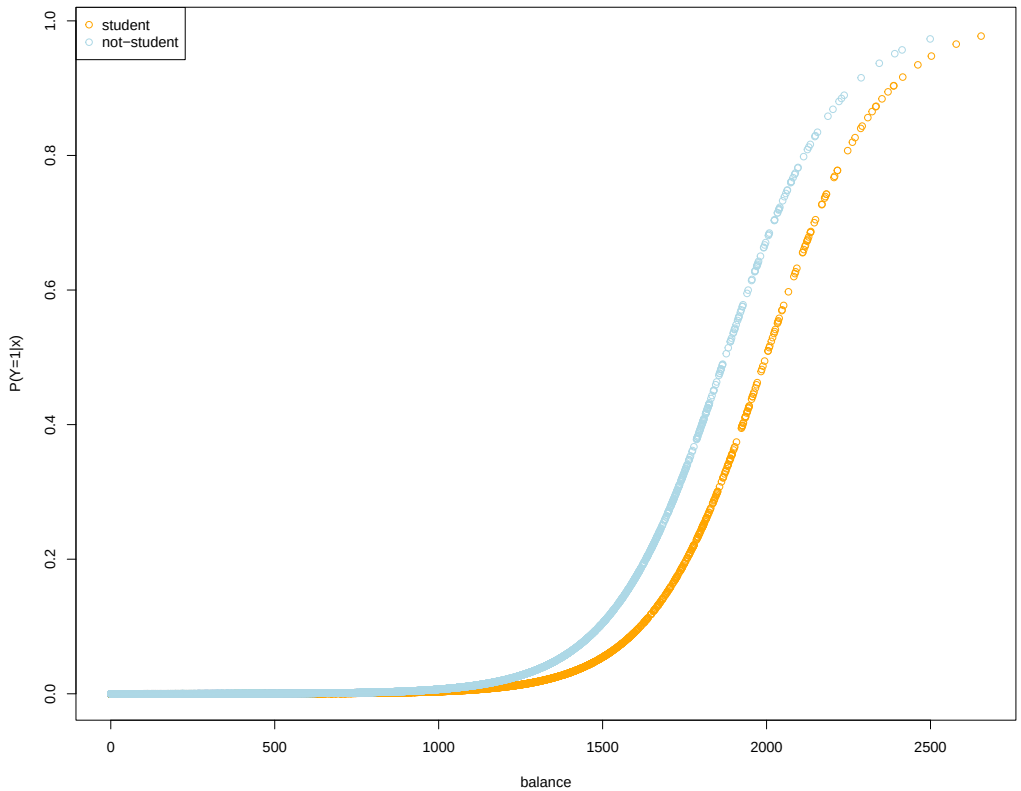
```
Call:
glm(formula = default ~ balance + student, family = binomial,
    data = Default)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.4578  -0.1422  -0.0559  -0.0203   3.7435

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.075e+01  3.692e-01 -29.116  < 2e-16 ***
balance      5.738e-03  2.318e-04  24.750  < 2e-16 ***
studentYes  -7.149e-01  1.475e-01  -4.846 1.26e-06 ***
---
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: 1571.7  on 9997  degrees of freedom
AIC: 1577.7

Number of Fisher Scoring iterations: 8
```

$$P(Y = 1 \mid x = (balance, studentYes)) =$$
$$F(-1.075 + .005738\, balance - .7149\, studentYes)$$

With just `balance` and `student` in the model, we can plot $P(Y = 1 \mid x)$ vs. x.

The orange points are for the students and the blue are for the non-students.

In both cases the probability of default increases with the balance, but at any fixed balance, a student is less likely to default.



AIC and BIC in Logistic Regression

In the logistic regression model, the *deviance* is just -2 times the logLikelihood (usually evaluated at the mle).

$$L(\beta) = \prod_{i=1}^{n} P(Y = y_i | X = x_i, \beta)$$

For logistic regression, $P(Y = 1|x, \beta) = F(x'\beta)$, $F(\eta) = \frac{\exp(\eta)}{(1+\exp(\eta))}$.

Given an estimate (usually a MLE) $\hat{\beta}$,

$$deviance = -2 \sum_{i=1}^{n} log(P(Y = y_i | X = x_i, \hat{\beta}))$$

The better the fit of the model, the bigger the likelihood, the smaller the deviance.

AIC:

AIC is the deviance + a model complexity penalty term.

$$AIC = deviance + 2 * (p + 1)$$

We penalize our fit as measured by the deviance with a penalty of 2 for each parameter.

Here we assume our parameters are p slopes for p x features + 1 intercept giving $p + 1$.

We like the model with a small AIC, good fit as measured by the deviance but not too complex.

AIC for the Default example:

With balance+student we had:

```
Residual deviance: 1571.7 on 9997 degrees of freedom
AIC: 1577.7
```

A parameter (a coefficient) costs 2.

- balance + student:
  Residual deviance: 1571.7, AIC: 1577.7 = 1571.7 + 2*(3).
- balance:
  Residual deviance: 1596.5, AIC: 1600.5 = 1593.5+2*(2).
- balance + student + income:
  Residual deviance: 1571.5, AIC: 1579.5 = 1571.5 +2*(4).
- student:
  Residual deviance: 2908.7, AIC: 2912.7 = 2908.7 + 2*(2).

$\Rightarrow$ pick balance+student

BIC:

BIC is an alternative to AIC, but the penalty is different.

$$BIC = deviance + log(n) * (p + 1)$$

$\log(n)$ tends to be bigger than 2, so BIC has a bigger penalty, so it suggest smaller models than AIC.

BIC for the Default example:

$\log(10000) = 9.21034$.

A parameter (a coefficient) costs 9.2.

- balance:
  1596.5, BIC: = 1593.5 + 9.2*(2) =1611.9.
- balance + student + income:
  BIC: = 1571.5 + 9.2*(4) = 1608.3.
- balance + student:
  BIC: = 1571.7 + 9.2*(3) = 1599.3.
- student:
  BIC: = 2908.7 + 9.2*(2) = 2927.1.

$\Rightarrow$ pick balance+student

Which is better, AIC or BIC??

nobody knows.

`R` prints out AIC, which suggests you might want to use it, but a lot of people like the fact that BIC suggests simpler models.

A lot of academic papers report both AIC and BIC and if they pick the same model are happy with that. Lame.

Checking the out of sample performance is safer !!!

2. The Logistic Likelihood

We have a very general intuition that we can use the negative log likelood as a measure of fit on training data.

In linear regression with iid normal errors, we saw that the form of the log likelihood is very intutive in terms of the error sum of squares.

In the linear case we had:

$$y_i \sim N(x_i^T\beta, \sigma^2)$$

$$L(\beta,\sigma) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}} \frac{1}{\sigma} e^{-\frac{1}{2\sigma^2}(y_i - x_i^T\beta)^2}$$

$$= (2\pi)^{-\frac{n}{2}} \sigma^{-n} e^{-\frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i - x_i^T\beta)^2}$$

$$\log L(\beta,\sigma) = -\frac{n}{2}\log(2\pi) - n\log(\sigma) - \frac{1}{2\sigma^2}\sum(y_i - x_i^T\beta)^2$$

ignore terms without β:

$$-\log L(\beta) = \frac{1}{2\sigma^2}\sum(y_i - x_i^T\beta)^2$$

So the contribution of each observation to the loss is

$$(y_i - x_i'\beta)^2 = (y_i - \beta_0 - \beta_1 x_{i1} - \ldots - \beta_p x_{ip})^2$$

So in the linear case, miminizing the log likelihood is the least squared problem and the LASSO could be written

$$\underset{\beta}{\textit{minimize}} \quad -\textit{log}(L(\beta)) + \lambda \sum_{j=1}^{p} |\beta_j|.$$

We follow exactly this approach for logistic regression.

But, let's have a closer look at the log likelihood and see if it is intuitive!!

For logistic regression we have:

$$p(y|x,\beta) = \begin{cases} F(x^T\beta) & y=1 \\ 1-F(x^T\beta) & y=0 \end{cases} \qquad F(\eta) = \frac{e^\eta}{1+e^\eta}$$

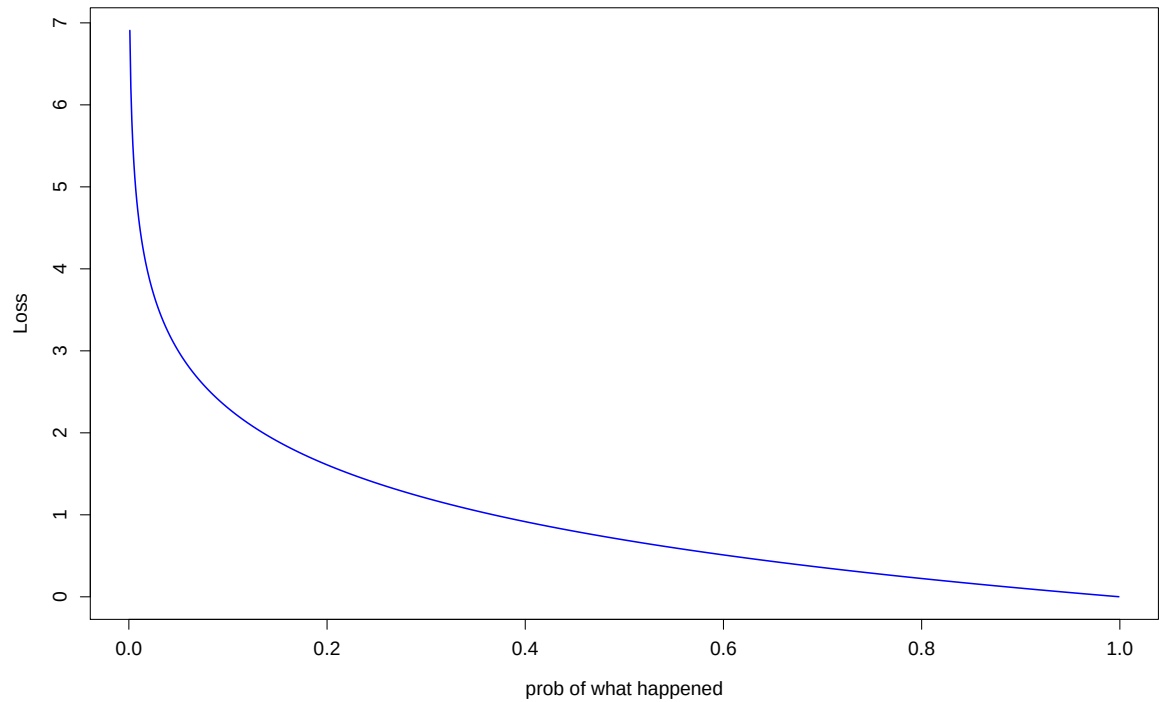
$$L(\beta) = \prod_{i=1}^{n} p(y_i|x_i,\beta)$$

$$-\log L(\beta) = \sum_{i=1}^{n} -\log\left(p(y_i|x_i,\beta)\right)$$

$$= \sum_{i=1}^{n} -\log(\text{probability of what happened})$$

So, for an individual observation the contribution is -log(prob of what happened).

x axis: probability of what happened (according to your model).
y axis: your loss.
In our logit problem "what happened" is either a 0 or a 1.



If your model gives a high probability to what happened, you have a small loss.
If your model gives a low probability to what happend, you have a big loss.

3. L2 and L1 Regularized Logisitic Regression

For linear models, we found that regularized versions gave us a very nice way to explore the bias-variance tradeoff.

We added a penalty term whose weighting parameter λ controlled the extent to which the coefficients were shrunk towards 0.

We would like to be able to do the same thing with logistic regression.

We want *regularized logistic regression*.

L2:

To get the ridge regression of logistic regression we just add the L2 penalty:

$$\underset{\beta}{\text{minimize}} \quad -log(L(\beta)) + \frac{\lambda}{2}\sum \beta_j^2$$

L1:

To get the Lasso regression of logistic regression we just add the L1 penalty:

$$\underset{\beta}{\text{minimize}} \quad -\frac{1}{n}log(L(\beta)) + \lambda\sum |\beta_j|$$

As in our discussion of the deviance, $-log(L(\beta))$ is a measure of fit on the training data.
Big L means good fit, so small $-log(L(\beta))$ means good fit.

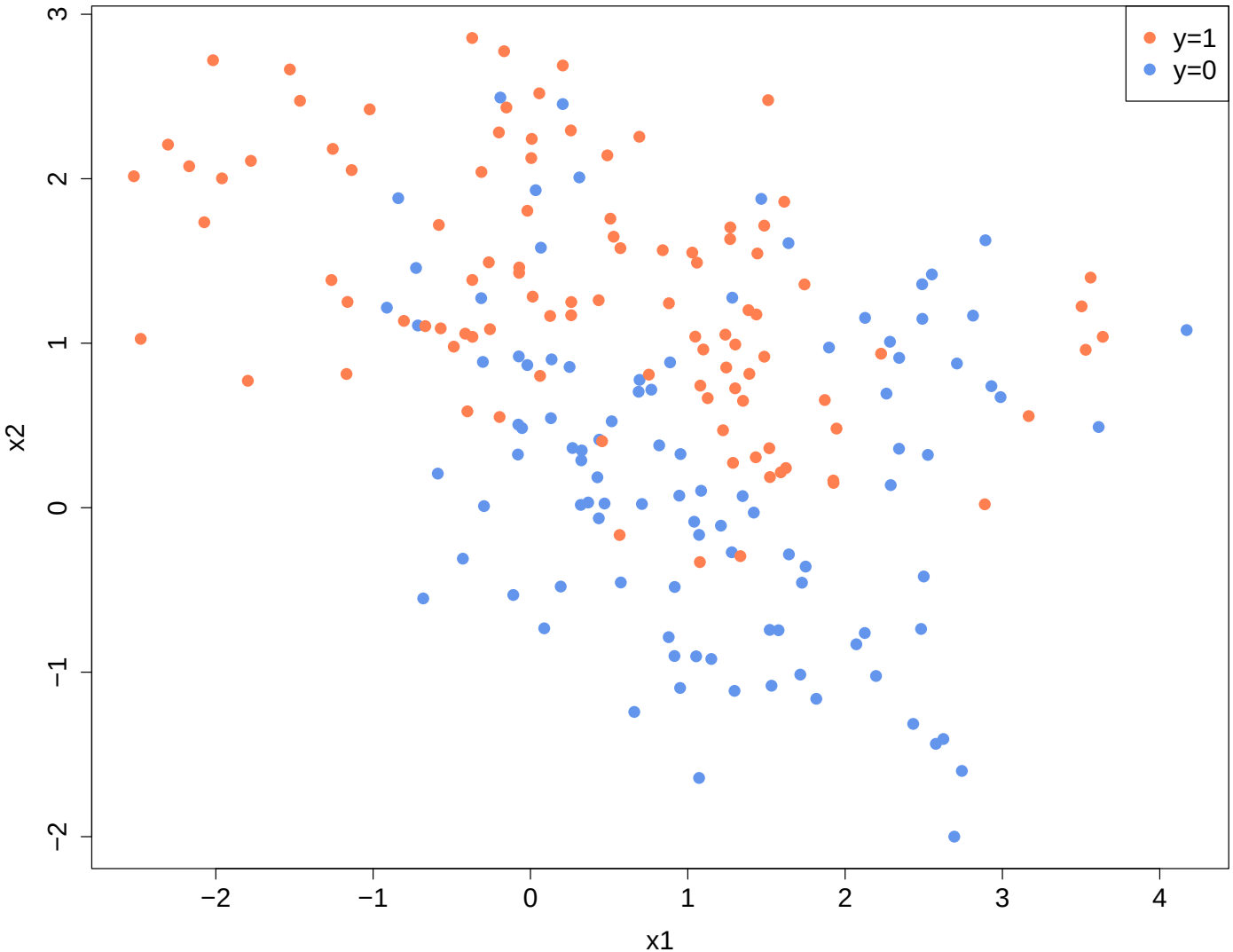
4. Simulated Example

Let's look at a simulated example from the "Elements of Statistical Learning" (`library(ElemStatLearn)`) R package.

> *Description*
> *This is a simulated mixture example with 200 instances and two classes. 100 members in each class.*

We have two numeric x's creatively called x_1 and x_2 and one binary outcome y.

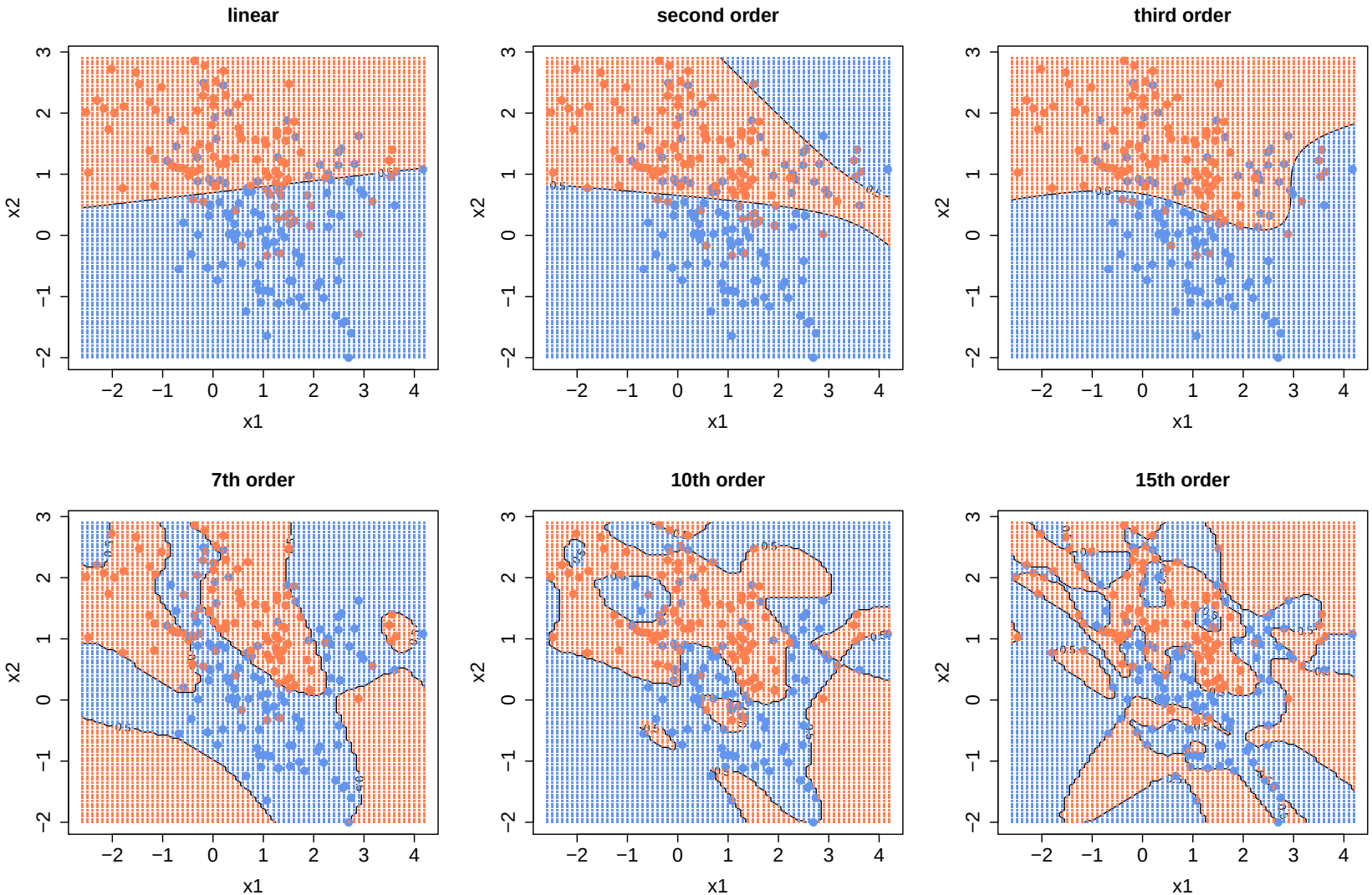
Plot the data: two numeric x and a binary y.



Now let's run logistic regression so y on x, throwing in more and more polynomials in x_1 and x_2.

We use the R function `poly` to construct all the polynomials.

Decision Boundary plots of logit fits with varying degrees of polynomial terms thrown in.



which one do you like the best ???

Ok, now let's try the Lasso.

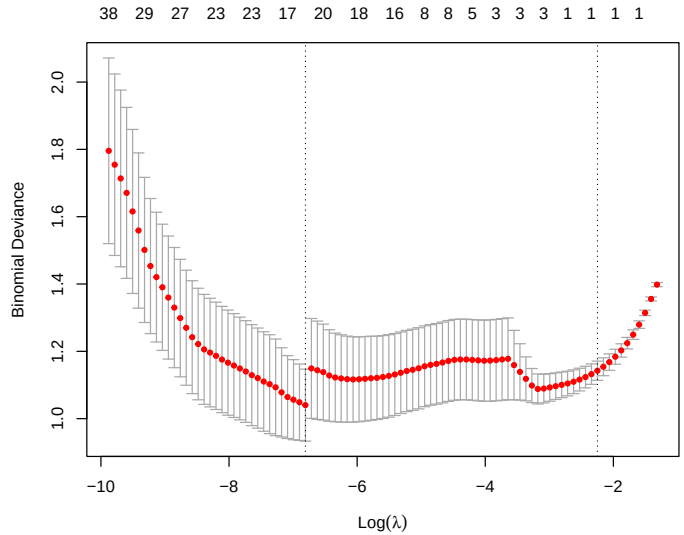
We will throw in all the 15th order terms.

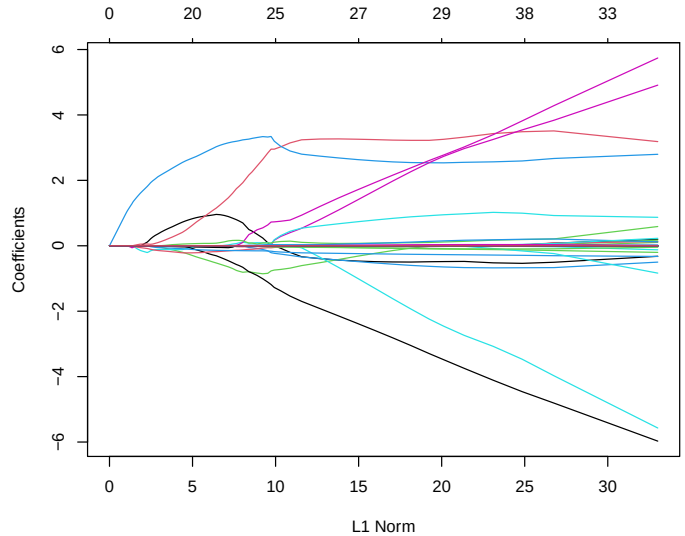
This gives us 135 x's !!!!!!!

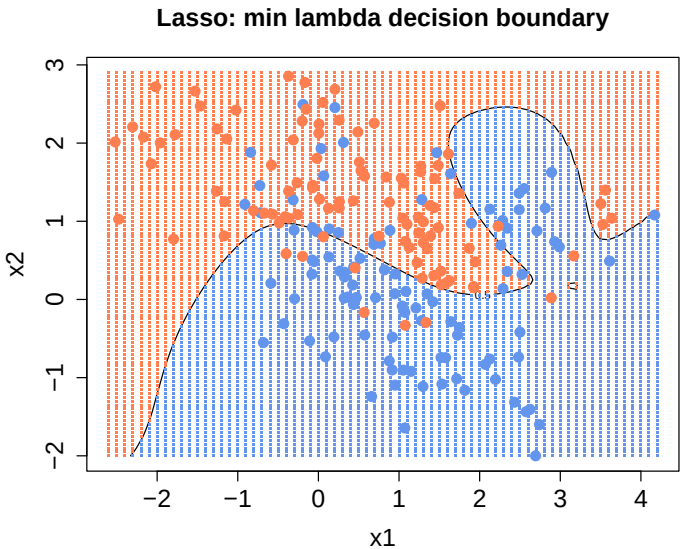
```
> 15*14/2 + 2*15
[1] 135
```

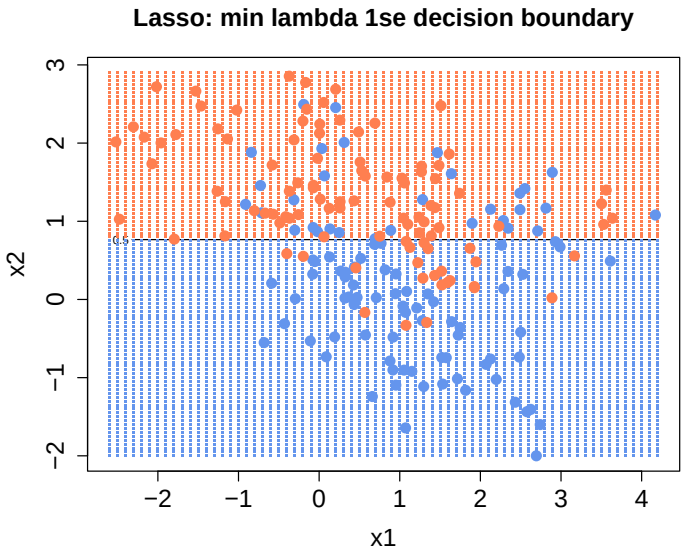
Of course, the fit using all of these is overfit.

But, can we use the Lasso to explore the possibility that some subset of the 135 is good?









Both of these look reasonable.

We got to try using all 135 x's, but the cv process pulls us back to something more restrained.

What transformations did the Lasso choose?

Here are the non-zero coefficients from lambda.min.

(Intercept)	x.1.0	x.2.0	x.3.0	x.7.0
-2.495723e+00	9.241824e-01	1.000302e+00	-4.584982e-01	1.136215e-03
x.8.0	x.15.0	x.0.1	x.3.1	x.2.2
1.362927e-04	-3.150041e-08	2.922533e+00	-4.609735e-02	-2.478831e-01
x.3.2	x.8.2	x.1.3	x.7.3	x.8.3
-1.885774e-01	3.671665e-04	7.296183e-02	1.149181e-04	1.348425e-04
x.0.4	x.2.7	x.0.15		
-1.329917e-01	4.745585e-03	1.255572e-06		

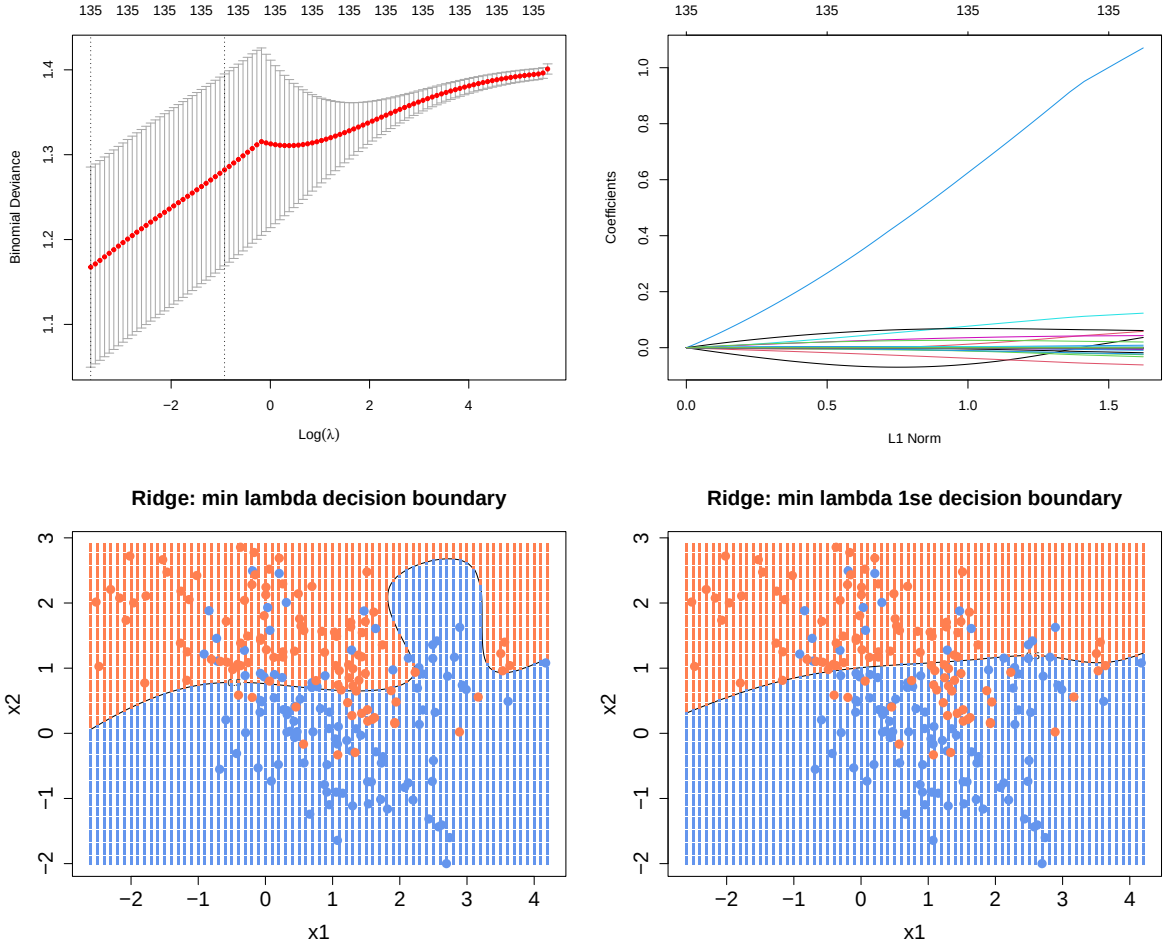
Wow, $x_1^8 \times x_3^2$ comes in, not obvious!!

Here are the non-zero coefficients from lambda.1se.

(Intercept)	x.0.1
-0.5352522	0.6994122

And the simple model is just use x_2 which is clearly a reasonable choice give the plot of the data.

Here are the results for Ridge regression.



What x do you think the big coefficient is for?

Note that the default choice of λ grid does not give us the bottom of the U.

Here are the largest absolute values of the Ridge coefficients at lambda.min.

```
      x.0.1 (Intercept)       x.1.1       x.1.2       x.0.2       x.2.0
1.070560515 0.856423790 0.123268634 0.061601542 0.061160591 0.058390390
      x.2.1       x.1.0       x.2.2       x.3.0       x.3.2       x.0.3
0.043099658 0.036520614 0.032405780 0.024975048 0.022550293 0.020280928
      x.3.1       x.0.4       x.1.3       x.3.3       x.0.5       x.3.4
0.018121108 0.009203506 0.008768088 0.006534978 0.003175778 0.003083115
      x.4.2       x.2.3
0.002955913 0.002828573
```

So, x_2 (same as `x.0.1`) dominates the fit.

5. We8There

Each observation consists of a restaurant review.

We also have the Zagat ratings for each restaurant.

We want to see how the Zagat rating relates to reviews.

What kind of customer review correspond to a good rating?

We have 6,166 observations.

Here are the Zagat ratings summarized:

```
   1    2    3    4    5
 615  493  638 1293 3127
```

We will dichotomize the outcome by making y 1 if the overall rating is greater than 3 and 0 otherwise.

```
y
   0    1
1746 4420
```

What is our x?

We want to use the information in the text of the restaurant reviews.

How do we convert the text of a review into a numeric x??

Bag of Words:

The most common way to look at a text document is to treat it as a bag of words.

That is, we look all the words in all the documents and then make a list of words or "terms" of interest.

For example the word "the" will probably occur with some frequency, but that may not be deemed to be of interest.

Given the list of terms, we just count the number of times each term occurs in the text of the document.

Note that this approach ignores the *word order*, the document is treated as a "bag of words".

bigrams:

A bi-gram is just a pair of words appearing in sequence.

For example, the pair of words "good food" appearing in sequence may mean more than "good" or "food" by themselves.

We will make a list of bigrams and then x will the number of times each bigram occurs in the text of a review.

The dimension of our X matrix is:

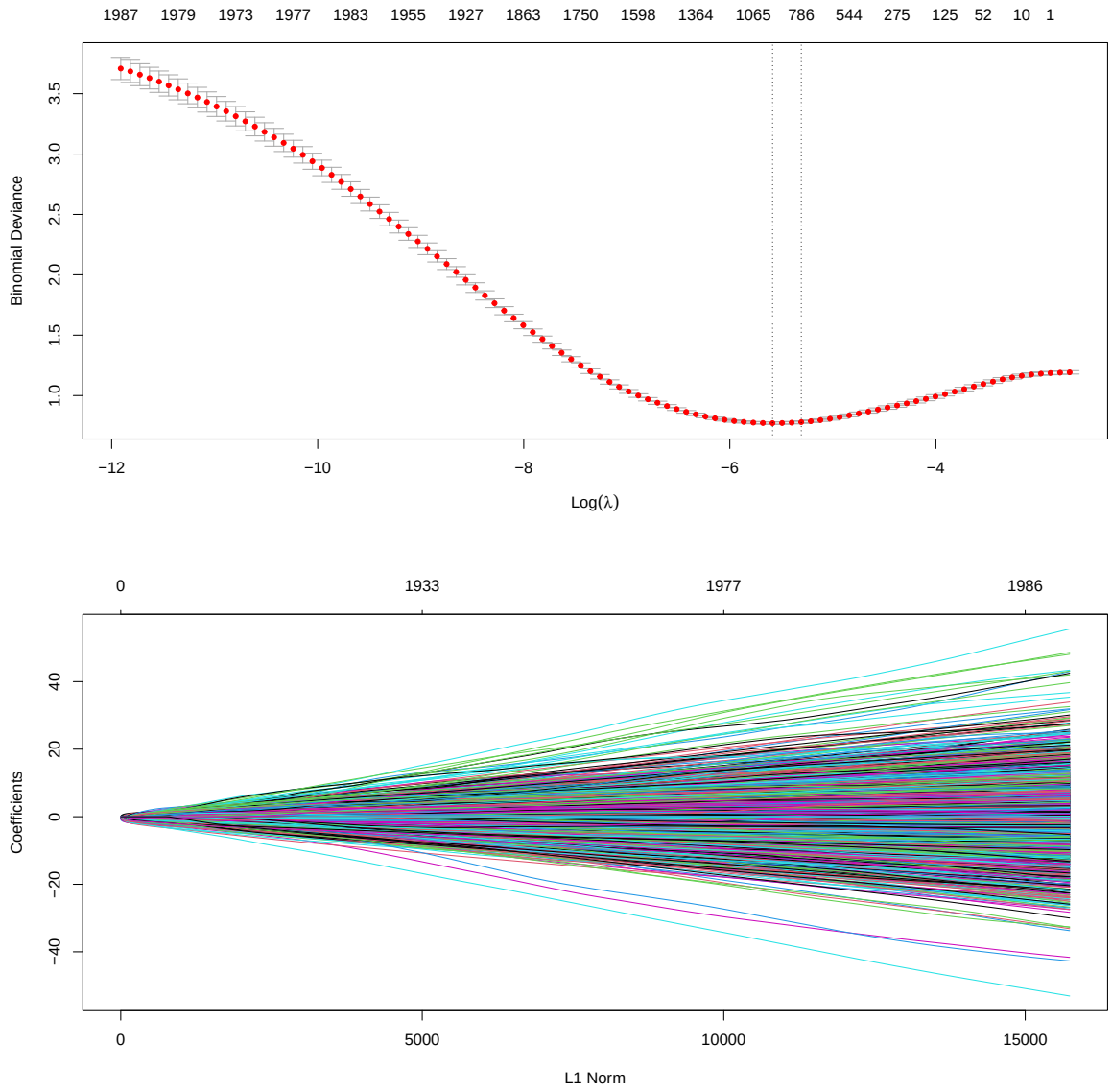
```
[1] 6166 2640
```

There are 2,640 bigrams in our list of ones we are counting.

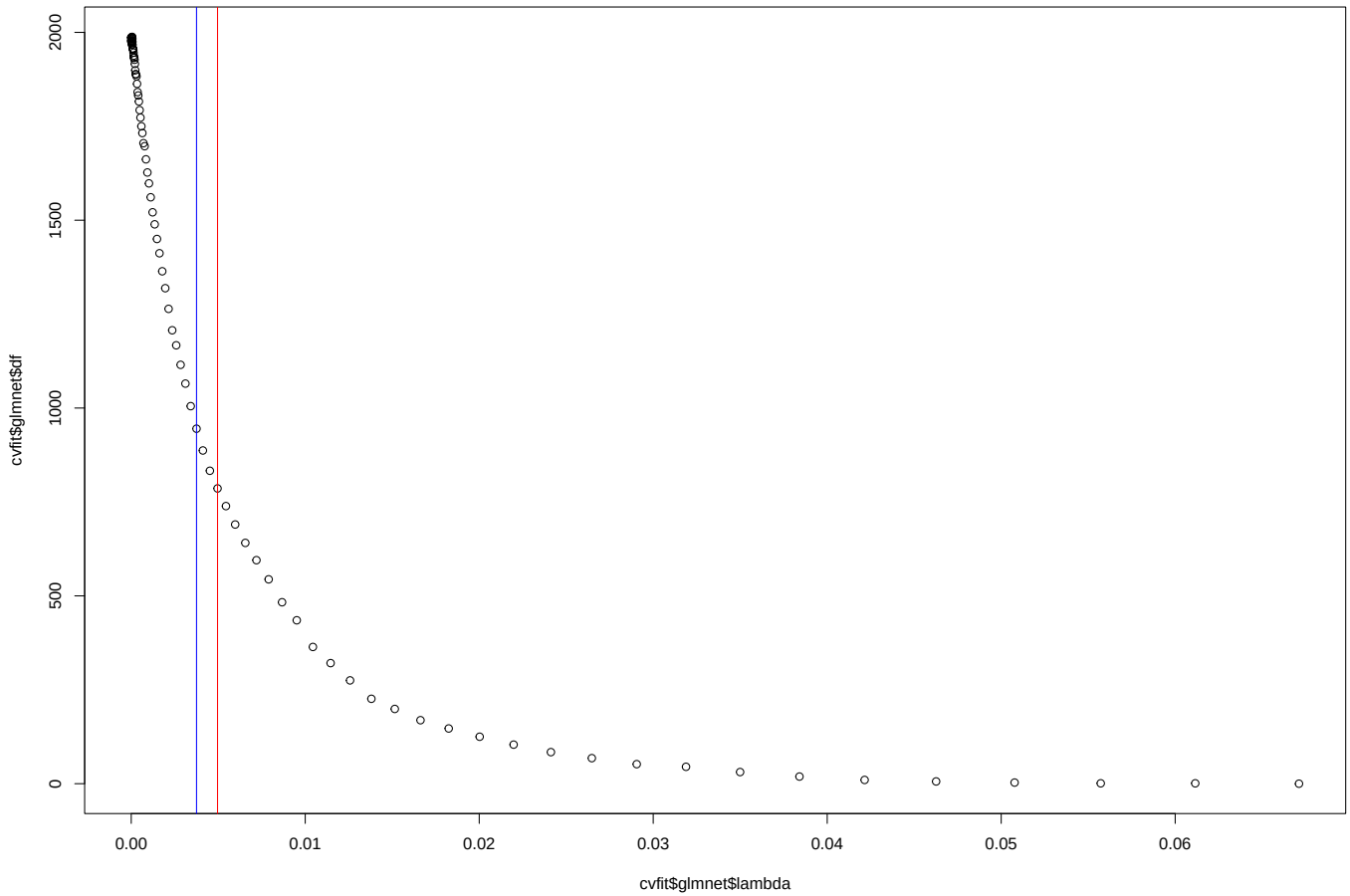
In our first observation, the bigrams with a non-zero count are:

```
 even though larg portion  mouth water     red sauc    babi back     back rib
           1             1            1            1            1            1
chocol mouss veri satisfi
           1             1
```

Here is the Lasso fit:



Here is the number of non-zero coefficients plotted against λ.



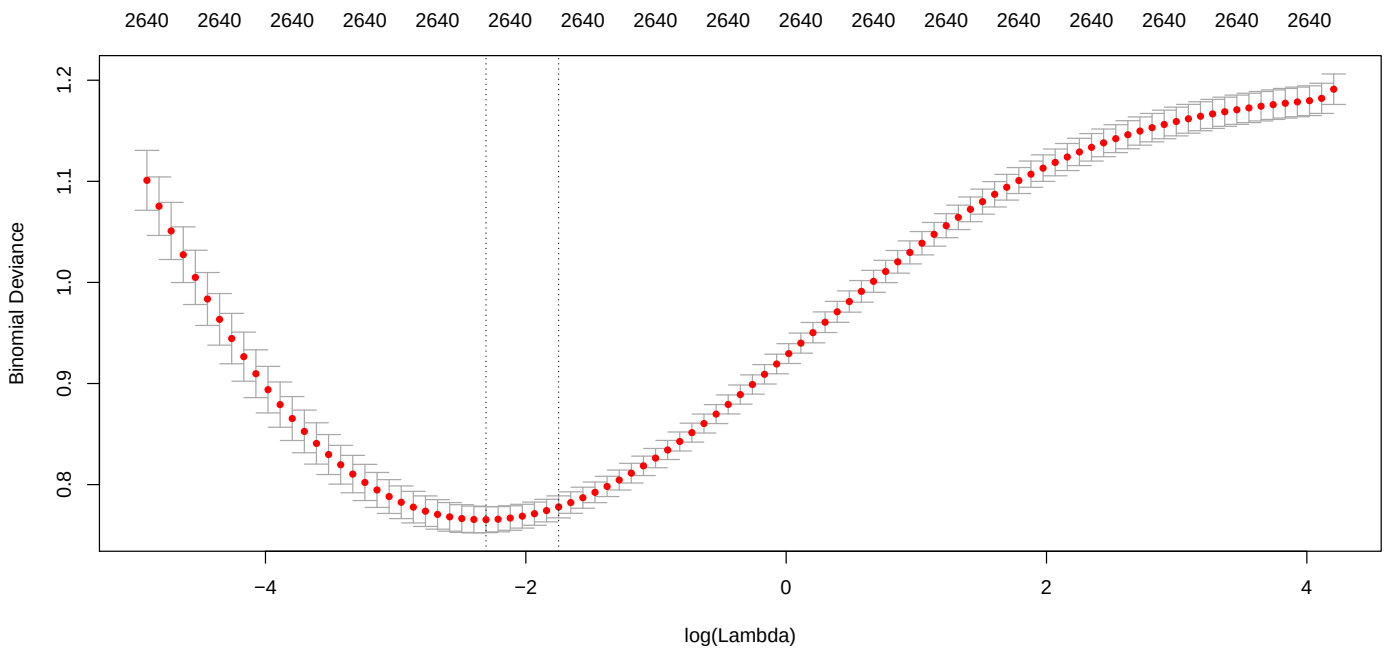
Lines drawn at `lambda.min` and `lambda.1se`.

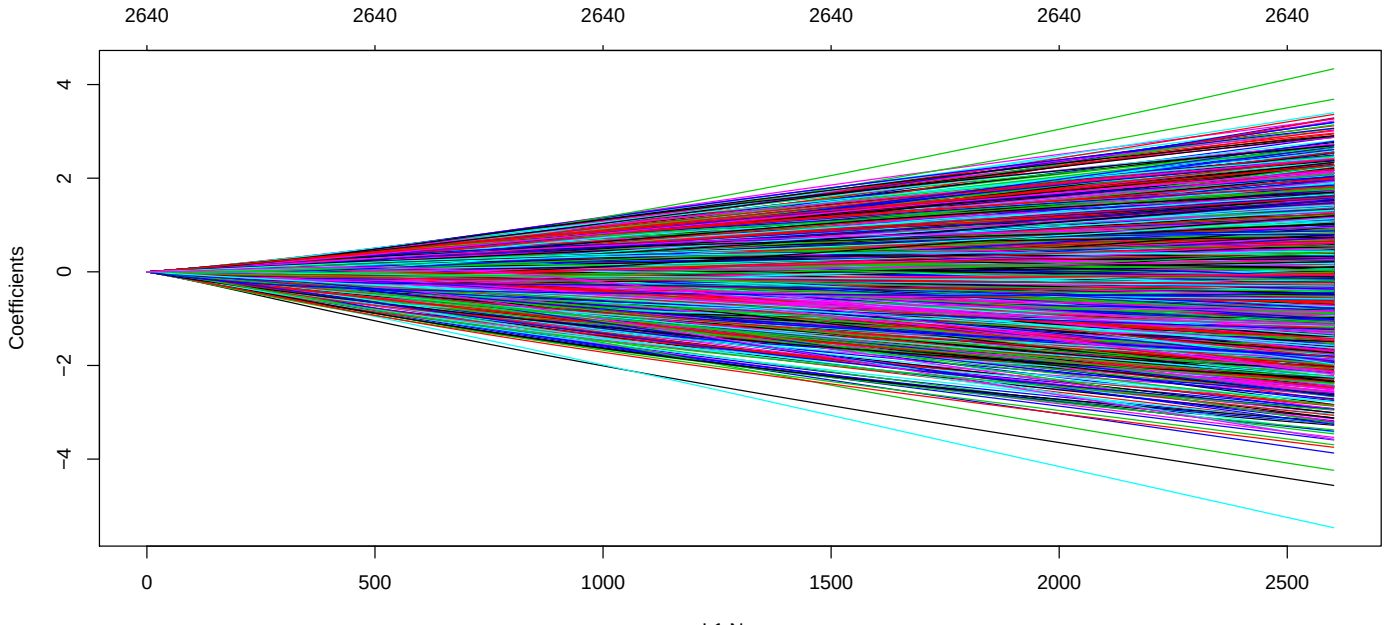
Here are the big positive and big negative coefficients.

```
Big positive coefficients:
 [1] "between two"       "can wait"          "beef sandwich"
 [4] "friend help"       "best meal"         "high recommend"
 [7] "cannot wait"       "melt mouth"        "food delici"
[10] "portion huge"      "wonder experi"     "veri uniqu"
[13] "pretti quick"      "can beat"          "full bar"
[16] "look out"          "absolut best"      "definit recommend"
[19] "(Intercept)"       "now live"          "quit few"
[22] "well worth"        "better most"       "never bad"
[25] "out world"         "great food"        "wall cover"
[28] "up good"           "great experi"      "absolut delici"
 [1] 1.6235758 1.3860122 1.3672099 1.3413956 1.2976254 1.2863832 1.1719763
 [8] 1.1502664 1.1288480 1.1032885 1.0835135 1.0807977 1.0533985 1.0492868
[15] 1.0435863 1.0429624 1.0308216 0.9933000 0.9842953 0.9840952 0.9834227
[22] 0.9633459 0.9578412 0.9512289 0.9375038 0.9236589 0.9104929 0.9073509
[29] 0.9026439 0.8981764


Big negative coefficients:
 [1] "over cook"       "waitress seem"   "just ok"         "bad food"
 [5] "food poison"     "servic ok"       "servic slow"     "mediocr food"
 [9] "one worst"       "wast money"      "food okay"       "mani option"
[13] "anoth chanc"     "veri disappoint" "food averag"     "terribl servic"
[17] "veri slow"       "veri poor"       "veri bland"      "quick lunch"
[21] "never go"        "gone down"       "servic terribl"  "food terribl"
[25] "never return"    "stay away"       "far better"      "mediocr best"
[29] "veri rude"       "extrem rude"
 [1] -1.484979 -1.501437 -1.526967 -1.528732 -1.540570 -1.569690 -1.585480
 [8] -1.591799 -1.642964 -1.652540 -1.672776 -1.673099 -1.685418 -1.704577
[15] -1.752020 -1.770758 -1.780688 -1.792626 -1.803219 -1.831518 -1.899799
[22] -2.025952 -2.035374 -2.044341 -2.087831 -2.097159 -2.267862 -2.277634
[29] -2.322275 -2.524030
```

Not surprising that it is bad to be “extrem rude”.

Here are Ridge results.





```
Big positive coefficients:
 [1] "(Intercept)"    "coconut shrimp" "look out"       "veri uniqu"
 [5] "year food"      "servic attent"  "friend help"    "well food"
 [9] "everyth menu"   "half shell"     "veri cozi"      "absolut best"
[13] "absolut delici" "serv hot"       "staff make"     "same peopl"
[17] "fair price"     "can beat"       "between two"    "excel price"
[21] "salad great"    "cannot wait"    "restaur make"   "time re"
[25] "back home"      "best meal"      "food superb"    "thorough enjoy"
[29] "reserv suggest" "keep go"
 [1] 0.9126494 0.6568706 0.6541469 0.6532986 0.6495907 0.6485409 0.6415978
 [8] 0.6413095 0.6396719 0.6378453 0.6257804 0.6232927 0.6146492 0.6085780
[15] 0.6063171 0.6015257 0.6008904 0.5989794 0.5986393 0.5957714 0.5942530
[22] 0.5889521 0.5856139 0.5836239 0.5766060 0.5764588 0.5733635 0.5724580
[29] 0.5715484 0.5708517


Big negative coefficients:
 [1] "one worst"       "anoth chanc"     "just anoth"      "day befor"
 [5] "veri disappoint" "quick lunch"     "food mediocr"    "veri poor"
 [9] "wast money"      "servic slow"     "terribl servic"  "servic ok"
[13] "got bill"        "stay away"       "food terribl"    "mani option"
[17] "veri limit"      "servic terribl"  "complain manag"  "veri bland"
[21] "just ok"         "mediocr food"    "food averag"     "veri rude"
[25] "food okay"       "veri slow"       "mediocr best"    "gone down"
[29] "far better"      "extrem rude"
 [1] -0.8566305 -0.8635339 -0.8699492 -0.8787698 -0.8881075 -0.8920423
 [7] -0.9201352 -0.9373039 -0.9438574 -0.9541782 -0.9581192 -0.9590981
[13] -0.9771998 -0.9841832 -0.9843469 -0.9920617 -1.0010190 -1.0014385
[19] -1.0104797 -1.0179680 -1.0222027 -1.0260030 -1.0503238 -1.0819638
[25] -1.0859973 -1.0939149 -1.1249573 -1.1382567 -1.2025776 -1.2955933
```

6. Multinomial Logit

The problem where Y is a binary outcome is very common.

But how do we extend logistic regression to the multinomial outcome case?

Let's look at the forensic glass data and use two of the x's (so we can plot) and all three of the outcomes.

Example, Forensic Glass:

Can you tell what kind of glass it was from measurements on the broken shards??

Y: glass type, 3 categories.

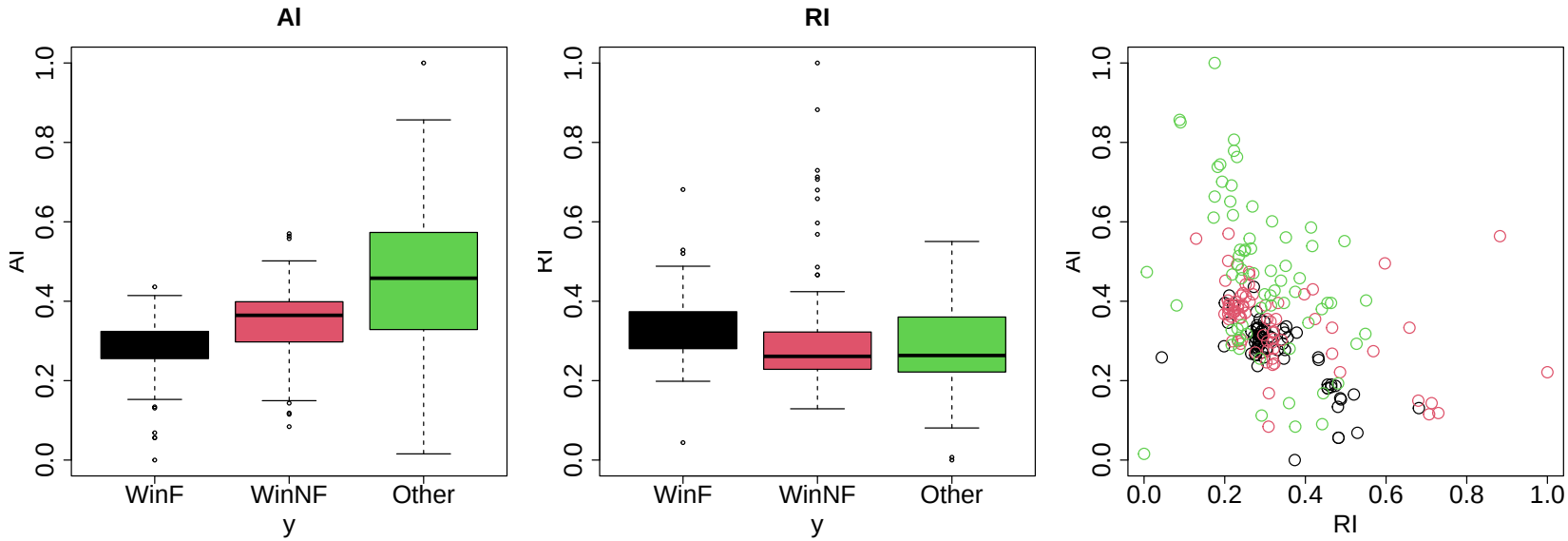
$Y \in S = \{\text{WinF}, \text{WinNF}, \text{Other}\}$.

WinF: float glass window
WinNF: non-float window
Other.

x: 3 numeric x's:
$x_1 = \text{RI}$: refractive index
$x_2 = \text{Al}$
$x_3 = \text{Na}$

Let's use two of the x's (so we can plot) and all three of the outcomes.

Here is the three outcome Y plotted agains the two x's x=(RI,AI).



kNN:

Before we go into the linear multinomial model, let's just note that KNN for classification is obvious.

Given test x and training (x_i, y_i):

Numeric Y:

- find the k training observations with x_i closest to x.
- predict y with the average of the y values for the neighbors.

Categorical Y:

- find the k training observations with x_i closest to x.
- predict Y with the most frequent of the y values for the neighbors.
- estimate $P(Y = y \mid x)$ with the proportion of neighbors having $Y = y$.

Multinomial Logit:

The multinomial logit model for $Y \in \{1, 2, \ldots, C\}$ is

$$P(Y = j|x) \propto \exp(x'\beta_j), \quad j = 1, 2, \ldots, C.$$

Or,

$$P(Y = j|x) = \frac{\exp(x'\beta_j)}{\sum_{j=1}^{C} \exp(x'\beta_j)}$$

So, each category gets a linear (affine) function of x !!!

Softmax Function:

For $x = (x_1, x_2, \ldots, x_C)$ The softmax function is

$$\sigma(x)_j = \frac{e^{x_j}}{\sum_j e^{x_j}}$$

Exponentiate and then normalize.

Takes a vector of real numbers x_j and maps them to a probability vector.

We use this a lot.

Identification:

Suppose we add β to each β_j.

$$\begin{aligned} P(Y = j|x) &= \frac{\exp(x'(\beta_j + \beta)}{\sum \exp(x'(\beta_j + \beta))} \\ &= \frac{\exp(x'\beta)\exp(x'\beta_j)}{\exp(x'\beta)\sum \exp(x'\beta_j)} \\ &= \frac{\exp(x'\beta_j)}{\sum \exp(x'\beta_j)} \end{aligned}$$

So, if we add any vector to all the β_j we get the exact some model!!

In this case we say the the set of parameters $\{\beta_j\}$ is not identified in that two different sets can give you the exact same likelihood.

The common identification strategy is to pick one of the β_j and set it equal to 0. Usually, it is either the "first" or the "last" β_j.

Note that as usual x may be $(1, x_2, \ldots, x_p)'$, that is we have included an intercept.

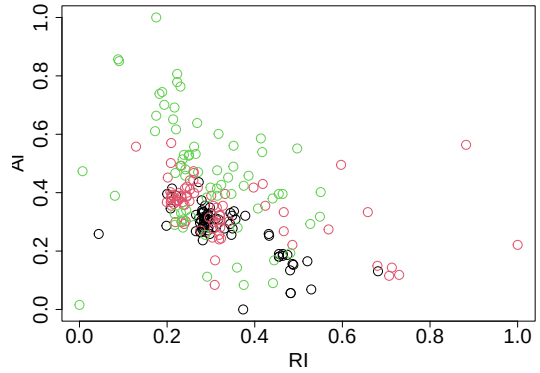
Here is ouput from fitting a multinomial logit using the forensic glass data.
(R package nnet, function multinom).

```
Call:
multinom(formula = y ~ ., data = ddf, maxit = 1000)

Coefficients:
      (Intercept)       RI        Al
WinNF   -3.277900 2.819056  7.861631
Other   -5.651027 2.718534 13.616921

Std. Errors:
      (Intercept)       RI       Al
WinNF    1.030785 1.610635 2.049922
Other    1.165932 1.872040 2.263372

Residual Deviance: 402.6627
AIC: 414.6627
```

The first β has been set to 0.

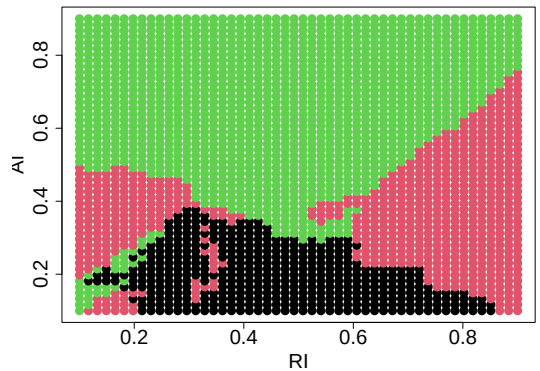
Note: $402.6627+12 = 414.6627$

plot of the data.

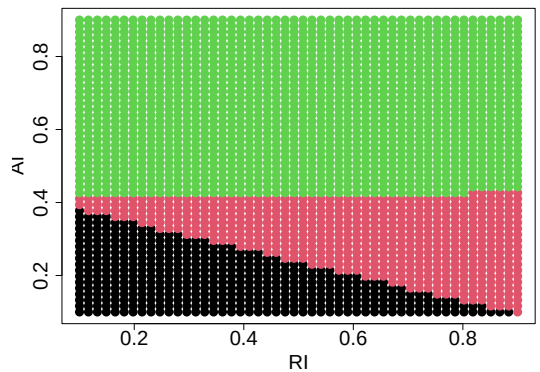


The color indicates the most probable outcome. black=WinF, red=WinNF, green=Other.

Most probable from knn. k=20.



Most probable from multinomial logit.



So, all but one class gets it's own coefficent vector.

```
Coefficients:
      (Intercept)       RI        Al
WinNF   -3.277900 2.819056  7.861631
Other   -5.651027 2.718534 13.616921

> table(y)
y
 WinF WinNF Other
   70    76    68
```

The coefficent vector for WinF has been set to 0.
$\beta_1 = (0.0, 0)$, for WinF
$\beta_2 = (-3.277900, 2.819056, 7.861631)$, for WinNF.
$\beta_3 = (-5.651027, 2.718534, 13.616921)$, for Other.

Let

$$\eta_1 = 0$$

$$\eta_2 = -3.28 + 2.82RI + 7.86AI$$

$$\eta_3 = -5.65 + 2.72RI + 13.62AI$$

$$P(Y = WinF = 1|x) = \frac{1}{1 + e^{\eta_2} + e^{\eta_3}}$$

$$P(Y = WinNF = 2|x) = \frac{e^{\eta_2}}{1 + e^{\eta_2} + e^{\eta_3}}$$

$$P(Y = Other = 3|x) = \frac{e^{\eta_3}}{1 + e^{\eta_2} + e^{\eta_3}}$$

When both AI and RI are small, the negative intercepts mean Y=WinF=1 is more likely.
When AI increases, Y=Other, becomes much more likely because of the large 13.62 coefficient.

$$P(Y = WinF = 1|x) = \frac{1}{1 + e^{\eta_2} + e^{\eta_3}}$$

$$P(Y = WinNF = 2|x) = \frac{e^{\eta_2}}{1 + e^{\eta_2} + e^{\eta_3}}$$

$$P(Y = Other = 3|x) = \frac{e^{\eta_3}}{1 + e^{\eta_2} + e^{\eta_3}}$$

Note

$$P(Y = i|x)/P(Y = j|x) = e^{\eta_i - \eta_j}$$

and the log odds is just $\eta_i - \eta_j$ with one of the η set to 0.

So the large difference in coefficients for AI (13.62-7.86) tells us the as AI increases the odds for 3=Other vs 2=WinNF will change quite a bit in favor of 3=Other.