[2] Big Data: Regression

Matt Taddy, University of Chicago Booth School of Business

faculty.chicagobooth.edu/matt.taddy/teaching

[2] Regression

Regression through linear models, and how to do it in R.

Interaction, factor effects, design (`model`) matrices.

Logistic Regression: an essential BD tool.

Estimation: Maximum Likelihood and Minimum Deviance

Much of this should be review, but emphasis will be different.

Linear Models

Many problems in BD involve a response of interest (‘y’) and a set of covariates (‘**x**’) to be used for prediction.

A general tactic is to deal in averages and lines.
We’ll model the conditional mean for y given **x**,

$$\mathbb{E}[\, y \mid \mathbf{x} \,] = f(\mathbf{x}'\boldsymbol{\beta})$$

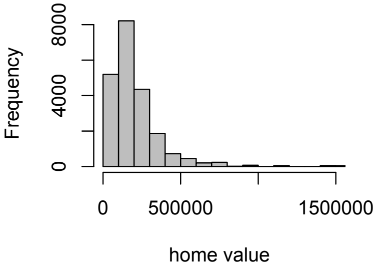
$\mathbf{x} = [1, x_1, x_2, \ldots x_p]$ is your vector of covariates.

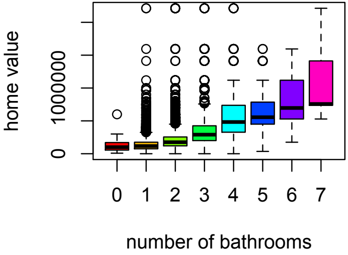
$\boldsymbol{\beta} = [\beta_0, \beta_1, \beta_2, \ldots \beta_p]$ are the corresponding coefficients.

The product is $\mathbf{x}'\boldsymbol{\beta} = \beta_0 + x_1\beta_1 + x_2\beta_2 + \ldots + x_p\beta_p$.

For notational convenience we use $x_0 = 1$ for the intercept.

Marginal and conditional distributions





On the left, all of the homes are grouped together.
On the right, home prices are grouped by # baths.

The marginal mean is a simple number.
The conditional mean is a function that depends on covariates.

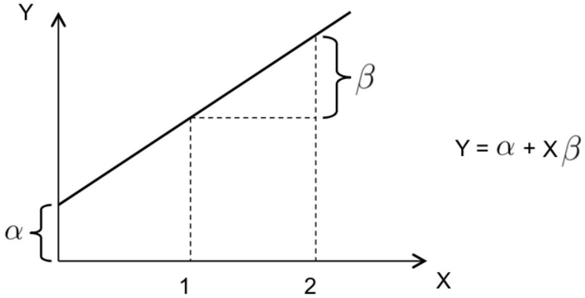
The data is distributed randomly around these means.

In a [Gaussian] linear regression,

$$y \mid \mathbf{x} \sim \mathrm{N}(\mathbf{x}'\boldsymbol{\beta}, \sigma^2)$$

Conditional mean is $\mathbb{E}[y|\mathbf{x}] = \mathbf{x}'\boldsymbol{\beta}$.

With just one x, we have simple linear regression.



$\mathbb{E}[y]$ increases by β for every unit increase in x.

Orange Juice

Three brands (*b*) Tropicana, Minute Maid, Dominicks

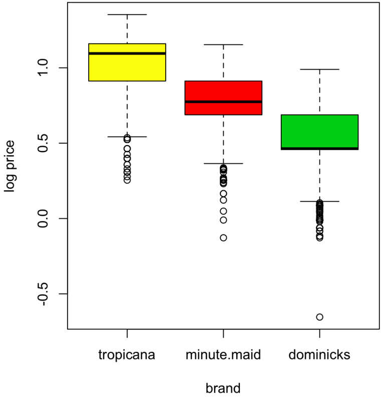
83 Chicagoland Stores
Demographic info for each

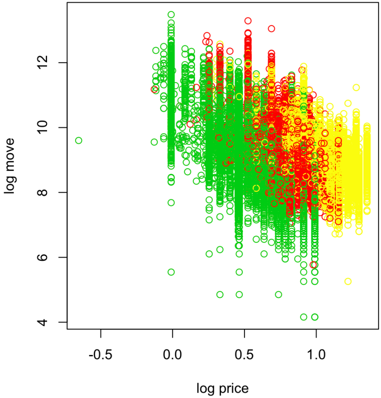
Price, sales (log units moved), and whether advertised (`feat`)

data in `oj.csv`, code in `oj.R`.

bayesm & Montgomery, 1987

The Juice: price, brand, and sales



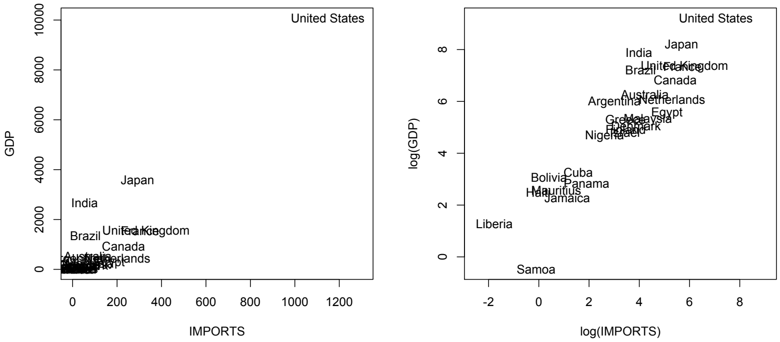


Each brand occupies a well defined price range.

Sales decrease with price.

Thinking About Scale

When making a linear point (this goes up, that goes down) think about the scale on which you expect find linearity.



If your scatterplots look like the left panel, consider using log.

log linear

We often model the mean for log(y) instead of y.
Why? Multiplicative (rather than additive) change.

$\log(y) = \log(a) + x\beta \Leftrightarrow y = ae^{x\beta}$.
Predicted y is multiplied by e^{β} after a unit increase in x.

Recall that $\log(y) = z \Leftrightarrow e^z = y$ where $e \approx 2.7$
$\log(ab) = \log(a) + \log(b)$ and $\log(a^b) = b\log(a)$.
I use log=ln, natural log. Anything else will be noted, e.g., $\log_2$.

Whenever y changes on a percentage scale, use log(y).
prices: "... Foreclosed homes sell at a 20% to 30% discount"
sales: "... our y.o.y. sales are up 20% across models"
volatility, fails, rainfall: most things that are strictly positive.

Price Elasticity

A simple orange juice 'elasticity model' for sales *y* has

$$\mathbb{E}[\log y] = \gamma \log(\texttt{price}) + \mathbf{x}'\boldsymbol{\beta}$$

Elasticities and log-log regression: for small values we can interpret γ as % change in *y* per 1% increase in `price`.

We run this in R:

```
glm(logmove ~ log(price) + brand, data=oj)
(Intercept) log(price) branBDinute.maid brandtropicana
    10.8288    -3.1387           0.8702         1.5299
```

and see sales drop by about 3.1% for every 1% price hike.

Regression in R

You need only one command

```
reg = glm(y ~ var1 + ...  + varP, data=mydata)
```

glm stands for Generalized Linear Model.

`lm` works too, but `glm` does more.

`y ~ a + b` is the ‘formula’ that defines your regression.
`y~.` is ‘regress on every variable in `mydata` not called `y`’

The object `reg` is a list of useful things (type `names(reg)`).
`summary(reg)` prints a bunch of information.
`coef(reg)` gives coefficients.
`predict(reg, newdata=mynewdata)` predicts.

`mynewdata` must be a data frame with exactly the same format as `mydata` (same variable names, same factor levels).

The Design Matrix

What happened to `branddominicks` or `makeDODGE`?

Our regression formulas look like $\beta_0 + \beta_1 x_1 + \beta_2 x_2 ...$
But `brand` is not a number, so you can't do `brand`$\times\beta$.

The first step of `glm` is to create a numeric *design matrix*.
It does this with a call to the `model.matrix` function:

```
"make"          "intercept" "makeFORD" "makeGMC"
 GMC       ⇒         1           0          1
 FORD                1           1          0
 DODGE               1           0          0
 FORD                1           1          0
```

The factor variable is on the left, and on the right we have numeric x that we can multiply against β coefficients.

Intercepts

Our OJ `glm` used `model.matrix` to build a 4 column design:

```
> x <- model.matrix( ~ log(price) + brand, data=oj)
> x[1,]
Intercept log(price) branBDinute.maid brandtropicana
  1.00000   1.353255         0.000000       1.000000
```

Each factor's reference level is absorbed by the intercept.
Coefficients are 'change relative to reference' (dominicks here).

To check the reference level of your factors do
`levels(myfactor)` The first level is reference.

To change this you can do
`myfactor = relevel(myfactor, "myref")`.

Interaction

Beyond additive effects: variables change how others act on y.

An interaction term is the product of two covariates,

$$\mathbb{E}[\, y \mid \mathbf{x}\,] = \ldots + \beta_j x_j + x_j x_k \beta_{jk}$$

so that the effect on $\mathbb{E}[y]$ of a unit increase in x_j is $\beta_j + x_k \beta_{jk}$.

It depends upon x_k!

Interactions play a massive role in statistical learning, and they are often central to social science and business questions.

- Does gender change the effect of education on wages?
- Do patients recover faster when taking drug A?
- How does advertisement affect price sensitivity?

Fitting interactions in R: use * in your formula.

```
glm(logmove ~ log(price)*brand, data=oj)
Coefficients:
              (Intercept)                log(price)
                 10.95468                  -3.37753
        branBDinute.maid           brandtropicana
                  0.88825                   0.96239
log(price):branBDinute.maid log(price):brandtropicana
                  0.05679                   0.66576
```

This is the model $\mathbb{E}[\log(v)] = \alpha_b + \beta_b \log(\texttt{price})$:
a separate intercept and slope for each brand '*b*'.

Elasticities are
dominicks: -3.4, minute maid: -3.3, tropicana: -2.7.
Where do these numbers come from? Do they make sense?

Advertisements

A key question: what changes when we feature a brand?

Here, this means in-store display promo or flier ad.

You could model the additive effect on log sales volume

$$\mathbb{E}[\log(\mathrm{v})] = \alpha_b + \mathbb{1}_{[\mathtt{feat}]}\alpha_{\mathtt{feat}} + \beta_b \log(\mathrm{p})$$

Or this *and* its effect on elasticity

$$\mathbb{E}[\log(v)] = \alpha_b + \beta_b \log(\mathrm{p}) + \mathbb{1}_{[\mathtt{feat}]}\left(\alpha_{\mathtt{feat}} + \beta_{\mathtt{feat}} \log(\mathrm{p})\right)$$

Or its *brand-specific* effect on elasticity

$$\mathbb{E}[\log(v)] = \alpha_b + \beta_b \log(\mathrm{p}) + \mathbb{1}_{[\mathtt{feat}]}\left(\alpha_{b,\mathtt{feat}} + \beta_{b,\mathtt{feat}} \log(\mathrm{p})\right)$$

See the R code for runs of all three models.

Connect the regression formula and output to these equations.

Brand-specific elasticities

	Dominicks	*Minute Maid*	*Tropicana*
Not Featured	-2.8	-2.0	-2.0
Featured	-3.2	-3.6	-3.5

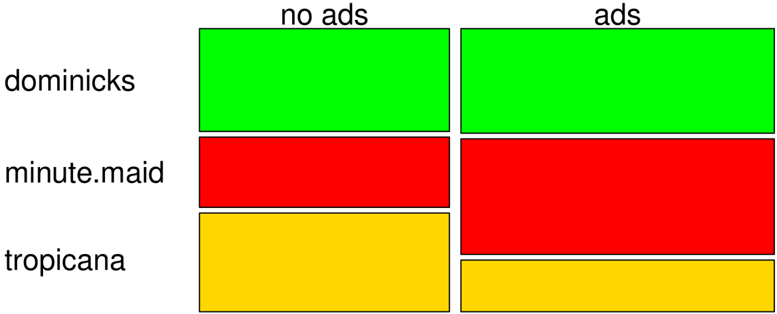
Ads always decrease elasticity.

Minute Maid and Tropicana elasticities drop 1.5% with ads, moving them from less to more price sensitive than Dominicks.

Why does marketing increase price sensitivity?
And how does this influence pricing/marketing strategy?

Confounding

Before including `feat`, Minute Maid behaved like Dominicks.
With `feat`, Minute Maid looks more like Tropicana. Why?



Because Minute Maid was more heavily promoted, and promotions have a negative effect on elasticity, we were *confounding* the two effects in the brand average elasticity.

Logistic Regression

Linear regression is just one type of linear model.
It is not even the most heavily practiced technique!

Logistic regression: when y is true or false (1/0).

Binary response as a prediction target:

- Profit or Loss, greater or less than, Pay or Default.
- Thumbs up or down, buy or not buy, potential customer?
- Win or Lose, Sick or Healthy, Republican or Democrat.

In high dimensions, it is often convenient to think binary.

Building a linear model for binary response data

Recall our original model specification: $\mathbb{E}[\ y \mid \mathbf{x}\ ] = f(\mathbf{x}'\beta)$.

The response 'y' is 0 or 1, leading to conditional mean:

$$\mathbb{E}[y|\mathbf{x}] = p(y=1|\mathbf{x}) \times 1 + p(y=0|\mathbf{x}) \times 0 \ = p(y=1|\mathbf{x}).$$

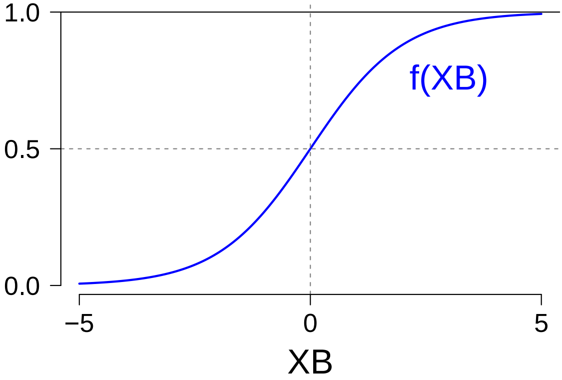
$\Rightarrow$ The expectation is a probability.

We'll choose $f(\mathbf{x}'\beta)$ to give values between zero and one.

We want a binary choice model

$$p = P(\ y = 1 \mid \mathbf{x}\ ) = f(\ \beta_0 + \beta_1 x_1 \ldots + \beta_p x_p\ )$$

where f is a function that increases in value from zero to one.



We'll use the logit link and do 'logistic regression'.

$$P(y = 1|\mathbf{x}) = \frac{e^{\mathbf{x}'\boldsymbol{\beta}}}{1 + e^{\mathbf{x}'\boldsymbol{\beta}}} = \frac{\exp[\beta_0 + \beta_1 x_1 \ldots + \beta_d x_d]}{1 + \exp[\beta_0 + \beta_1 x_1 \ldots + \beta_d x_d]}$$

The 'logit' link is common, for a couple good reasons.

One big reason: A little algebra shows

$$\log\left[\frac{p}{1-p}\right] = \beta_0 + \beta_1 x_1 \ldots + \beta_d x_d,$$

so that it is a linear model for log-odds.

Spam filter

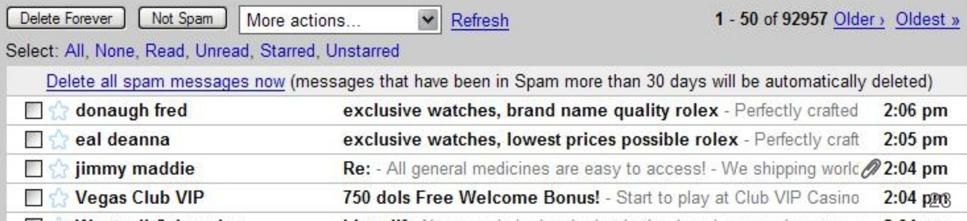
Your inbox does binary regression: spam v not spam.

Say $y = 1$ for spam, otherwise $y = 0$.

`spam.csv` has for 4600 emails (about 1800 spam) word/char frequencies (% of message) and related info.

Units here are % of total words + special characters.
If email i has length m_i, 1% implies $0.01 \times m_i$ tokens.

Logistic regression fits $\mathrm{p}(y = 1)$ as a function of email content.



Logistic regression is easy in R

Again using `glm`:

```
glm(Y ~ X, data=mydata, family=binomial)
```

The argument '`family=binomial`' indicates y is binary.

The reponse can take a few forms:

- **y** $= 1, 1, 0,$ .... numeric vector.
- **y** = `TRUE, TRUE, FALSE,` .... logical.
- **y** = `'win','win','lose',` .... factor.

Everything else is the same as for linear regression.

Perfect Separation

```
spammy <- glm(spam~., data=email, family='binomial')
Warning message:
glm.fit: fitted probabilities numerically 0 or 1 occurred
```

We're warned that some emails are clearly spam or not spam.

This is called 'perfect separation'. You don't need to worry.

The situation can introduce numeric instability in your algorithm (mess with standard errors, p-values, etc), but is largely benign.

It occurs here because some words are clear discriminators:

```
email$word_freq_george>0
                  FALSE TRUE
         important  2016  772
         spam       1805    8
```

Guy's named George; spammers in the early 90s weren't fancy.

Interpreting Coefficients

The model is

$$\frac{p}{1-p} = \exp\left[\beta_0 + x_1\beta_1 \ldots x_p\beta_p\right]$$

So $\exp(\beta_j)$ is the odds multiplier for a unit increase in x_j.

Recall our x_j units are % of total tokens (m_i) in an email.

`b["word_freq_george"]=-11.7`, so $0.01 \times m_i$ more `george` occurrences multiplies odds of spam by $\exp(-11.7) \approx 8/10^6$.

`b["char_freq_dollar"]=5.3`, so $0.01 \times m_i$ more '`$`' occurrences multiplies odds of spam by $\exp(5.3) \approx 200$.

What is the odds multiplier for a covariate coefficient of zero?

The summary function gives coefficients, plus some other info. The bit at the bottom is especially useful:

```
summary(spammy) ...
(Dispersion parameter for binomial family taken to be 1)
    Null deviance: 6170.2  on 4600  degrees of freedom
Residual deviance: 1815.8  on 4543  degrees of freedom
AIC: 1931.8
```

The same stuff is in output for our *linear* OJ regression.

```
summary(ojreg) ...
(Dispersion parameter for gaussian family taken to be 0.48)
    Null deviance: 30079  on 28946  degrees of freedom
Residual deviance: 13975  on 28935  degrees of freedom
AIC: 61094
```

These are stats on fit, and they are important in either linear or logistic regression. Understanding deviance ties it all together.

Estimation and Fit

Two complementary concepts:

Deviance refers to the distance between data and fit.
You want to make it as small as possible.

Likelihood is the probability of your data given parameters.
You want to make it as big as possible.

$$\text{Deviance} = -2\log[\text{ Likelihood }] + C$$

C is a constant you can mostly ignore.

We'll think of deviance as a cost to be minimized.
This is referred to as maximum likelihood estimation (MLE)

Least-Squares and deviance in linear regression

The probability model is $y \sim \mathrm{N}(\mathbf{x}'\boldsymbol{\beta}, \sigma^2)$.

$N(\mu, \sigma^2) = \exp\left[-(y-\mu)^2/2\sigma^2\right]/\sqrt{2\pi\sigma^2}$.

Given n independent observations, the likelihood is

$$\prod_{i=1}^{n} \mathrm{pr}(y_i|\mathbf{x}_i) = \prod_{i=1}^{n} \mathrm{N}(y_i; \mathbf{x}_i'\boldsymbol{\beta}, \sigma^2) \propto \exp\left[-\frac{1}{2}\sum_{i=1}^{n}\left(y_i - \mathbf{x}_i'\boldsymbol{\beta}\right)^2/\sigma^2\right]$$

This leads to Deviance $\propto \dfrac{1}{\sigma^2}\displaystyle\sum_{i=1}^{n}\left(y_i - \mathbf{x}_i'\boldsymbol{\beta}\right)^2$.

Minimizing deviance is the same as least squares!

And thus the MLE minimizes our sum of squared errors.

♦ MLE for Logistic Regression

Our logistic regression likelihood is the product

$$
\begin{aligned}
\text{LHD} &= \prod_{i=1}^{n} P(y_i|\mathbf{x}_i) = \prod_{i=1}^{n} p_i^{y_i}(1-p_i)^{1-y_i} \\
&= \prod_{i=1}^{n} \left(\frac{\exp[\mathbf{x}_i'\beta]}{1+\exp[\mathbf{x}_i'\beta]}\right)^{y_i} \left(\frac{1}{1+\exp[\mathbf{x}_i'\beta]}\right)^{1-y_i}
\end{aligned}
$$

This is maximized by minimizing the deviance

$$
\begin{aligned}
D &= -2\sum_{i=1}^{n} \left(y_i \log(p_i) + (1-y_i)\log(1-p_i)\right) \\
&\propto \sum_{i=1}^{n} \left[\log\left(1+e^{\mathbf{x}_i'\boldsymbol{\beta}_i}\right) - y_i\mathbf{x}_i'\boldsymbol{\beta}_i\right]
\end{aligned}
$$

All we've done is take the logarithm and multiply by -2.

We have the same output as for a linear/gaussian model.

But the ‘`dispersion parameter`’ here is always set to one.
Check this to make sure you’ve actually run logistic regression.

```
> summary(spammy)...
(Dispersion parameter for binomial family taken to be 1)
    Null deviance: 6170.2  on 4600  degrees of freedom
Residual deviance: 1815.8  on 4543  degrees of freedom
AIC: 1931.8
```

‘degrees of freedom’ is actually ‘number of observations - df’, where df is the number of coefficients estimated in the model.

That is, df(deviance) = nobs - df(regression).

From the R output, how many observations do we have?

Sum of Squares (Deviance) is the bit we need to minimize:

$$D \propto \sum_{i=1}^{n} (y_i - \mathbf{x}_i \boldsymbol{\beta})^2$$

This makes the observed data *as likely as possible*.

Error variance σ^2 measures variability around the mean.
i.e., $\sigma^2 = \mathrm{var}(\varepsilon)$, where $\varepsilon_i = y_i - \mathbf{x}_i \boldsymbol{\beta}$ are the 'residuals'.

R estimates σ^2 and calls it the `dispersion parameter`.
e.g., in output for our linear OJ regression:

```
(Dispersion parameter for gaussian family taken to be 0.48)
```

Even if we know $\boldsymbol{\beta}$, we only predict log sales *with uncertainty*.
e.g., there's a 95% probability of log sales in $\mathbf{x}'\boldsymbol{\beta} \pm 2\sqrt{0.48}$

Residual deviance D is what we've minimized, using $\mathbf{x}'\beta$.

Null deviance D_0 is for the model where you don't use **x**.
i.e., if you use $\hat{y}_i = \bar{y}$:

- $D_0 = \sum(y_i - \bar{y})^2$ in linear regression.
- $D_0 = -2\sum\left[y_i \log(\bar{y}) + (1 - y_i)\log(1 - \bar{y})\right]$ in logistic reg.

The difference between D and D_0 is due to info in x.

Proportion of deviance explained by **x** is called R^2:

$$R^2 = \frac{D_0 - D}{D_0} = 1 - \frac{D}{D_0}.$$

This measures how much variablity you are able to model.

in `spammy`: $R^2 = 1 - 1816/6170 = 0.71$
in `ojreg`: $R^2 = 1 - 13975/30079 = 0.54$

R^2 in linear regression

These forumulas should look pretty familiar.

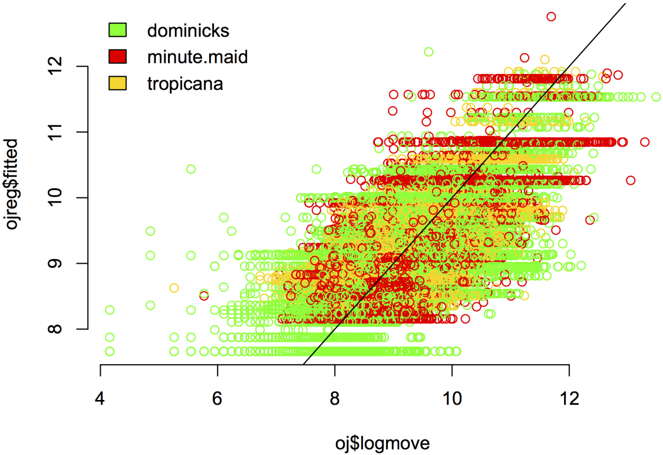
$R^2 = 1-$ SSE/SST from previous classes,
and linear deviance is just the Sum of Squares!

You'll also recall that $R^2 = \mathrm{cor}(y, \hat{y})^2$ in linear regression,
where $\hat{y}$ denotes 'fitted value' $\hat{y} = f(\mathbf{x}'\hat{\boldsymbol{\beta}}) = \mathbf{x}'\hat{\beta}$ in lin reg.

```
cor(ojreg$fitted,oj$logmove)^2
[1] 0.5353939
```

For linear regression, min deviance = max $\mathrm{cor}(y, \hat{y})$.
If y vs $\hat{y}$ makes a straight line, you have a perfect fit.

Fit plots: $\hat{y}$ vs y

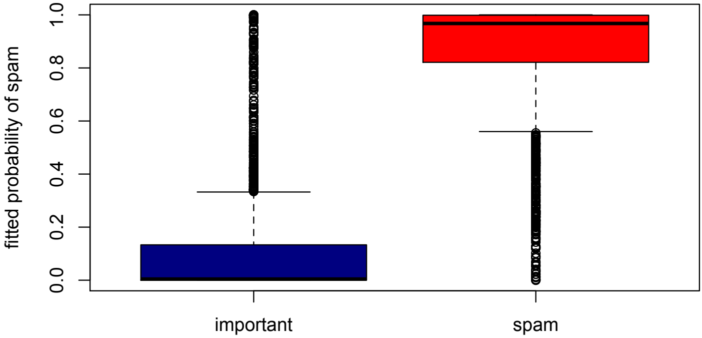


It's good practice to plot $\hat{y}$ vs y as a check for mispecification.

(e.g., non-constant variance, nonlinearity in residuals, ...)

Fit plots for logistic regression

We can plot $\hat{y}$ vs y in logistic regression using a boxplot.



The estimation pushes each distribution away from the middle.

Where would you choose for a classification cut-off?

Prediction

We've seen that prediction is easy with `glm`:

```
predict(spammy, newdata=email[1:4,])
        1          2          3          4
0.4852312  4.4135889 12.9939395  1.2575953
```

This outputs $\mathbf{x}'\hat{\beta}$ for each $\mathbf{x}$ row of `mynewdata`.

In logistic regression, to get probabilities $e^{\mathbf{x}'\hat{\beta}}/(1+e^{\mathbf{x}'\hat{\beta}})$, add the argument `type="response"`.

```
predict(spammy, newdata=email[1:4,], type="response")
        1         2         3         4
0.6189824 0.9880333 0.9999977 0.7786119
```

`newdata` *must* match the format of original data.

Out-of-Sample Prediction

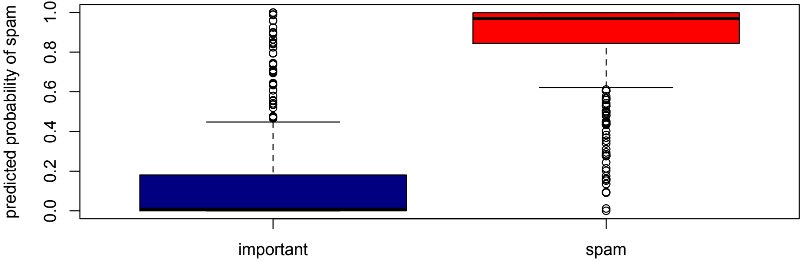
You care about how your model predicts out-of-sample (OOS).

One way to test this is to use a validation sample.
Fit your model to the remaining *training data*,
and see how well it predicts the *left-out data*.

```
# Sample 1000 random indices
leaveout <- sample(1:nrow(email), 1000)
# train the model WITHOUT these observations
spamtrain <- glm(spam~.,
              data=email[-leaveout,], family='binomial')
# predicted probability of spam on the left out data
pspam <- predict(spamtrain,
              newdata=email[leaveout,], type='response')
```

Out-of-Sample Prediction

Fit plots on the 1000 left out observations.



`deviance.R` has a function to get deviances from `y` and `pred`.

For the left-out data, we get $D0 = 1316$, $D = 568$, $R^2 = 0.57$.

Since the sample is random, you might get different results.

Note: OOS R^2 is lower than in-sample R^2 (> 0.7).

Week 2 Homework

American Housing Survey: 2004 home values

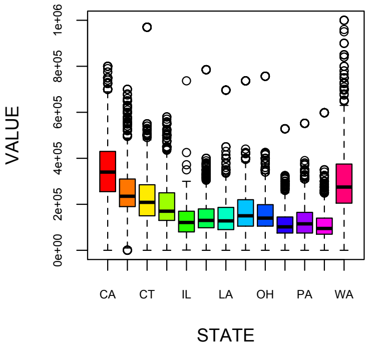
Demographics
School, Income

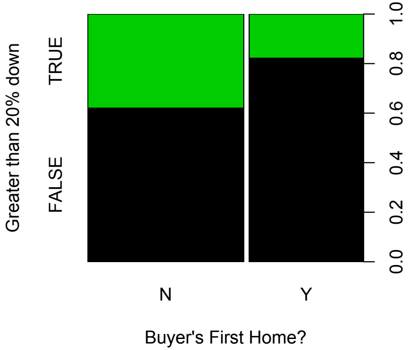
Finance
Mortgage, Sale

Neighborhood
Noise, Smells

Geography
State, Urban?

We'll be modelling log home value, and whether or not the buyer had at least 20% down (when I grabbed this data, there was lots of talk about stricter mortgage requirements).





Data is in `homes2004.csv`, with variable dictionary in `homes2004code.txt`. Primer is in `homes2004_start.R`.

Week 2 Homework

[1] Plot some relationships and tell a story.

[2] Regress log value onto all but mortgage and purchase $.

- How many coefficients are significant at 10% FDR?
- Re-run regression with only the significant covariates, and compare R^2 to the full model.

[3] Fit a regression for whether the buyer had $\geq$ 20% down (again, onto everything but `AMMORT` and `LPRICE`).

- Interpret effects for 1st home buyers and # of bathrooms.
- Add + describe interaction for 1st home-buyers and #baths.

[4] Re-fit your model from Q3 for only homes worth $> 100k$. Compare in-sample fit to R^2 for predicting homes worth $< 100k$.