Regressions cont.: Deviance/GLM, OOS, Bootstrap

Lecture 4

Connor Dowd

April 8th, 2021

Today's Class | Time permitting

1. Quick Review
    - Linear Regression
    - Logistic Regression
2. Deviance
3. Out-of-Sample performance
4. GLM more broadly:
    - Poisson, etc.
5. Inference
    - Bootstrap?
6. Preview HW2
7. Review HW1 Answers.

Quick Review

Linear Regression

Many problems involve a response or outcome (`y`),
And a bunch of covariates or predictors (`x`) to be used for regression.

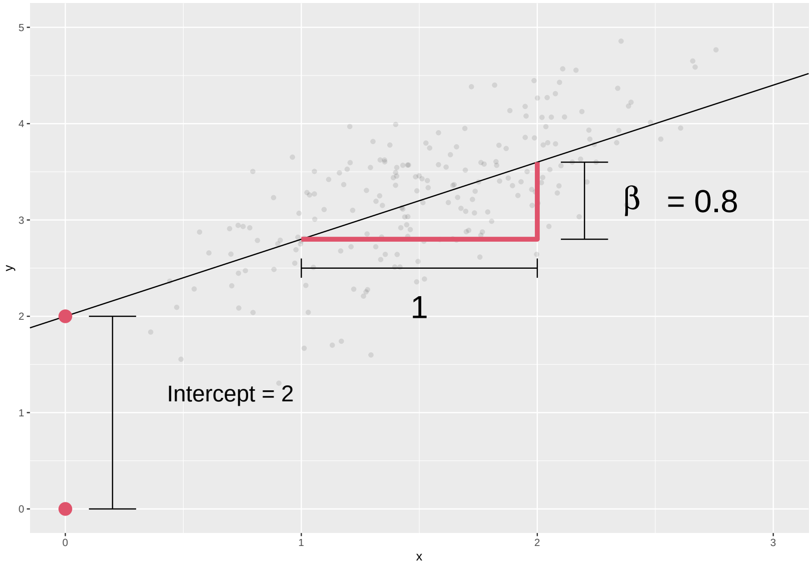
A general tactic is to deal in averages and lines.

$$E[y|x] = f(x'\beta)$$

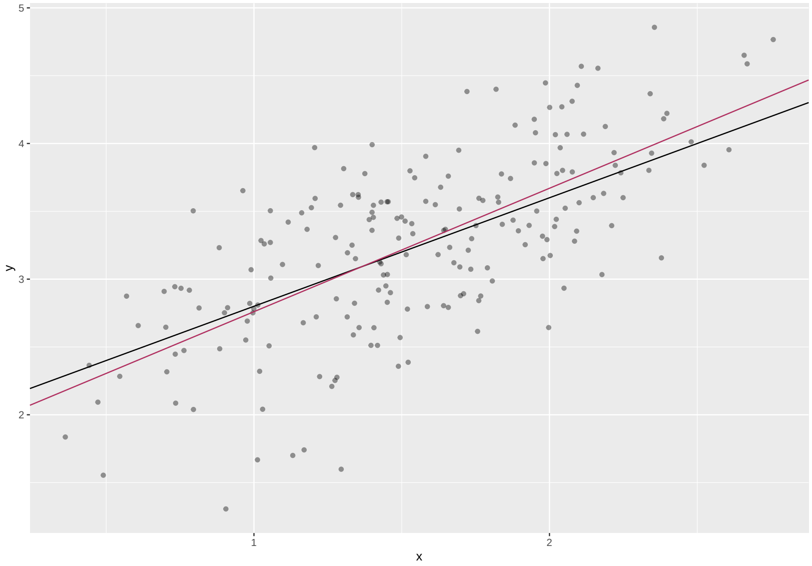
Where $x = [1, x_1, x_2, x_3, ..., x_p]$ is our vector of covariates. (Our number of covariates is p again)
$\beta = [\beta_0, \beta_1, \beta_2, ..., \beta_p]$ are the corresponding coefficients.
The product $x'\beta = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p$.

For simplicity we denote $x_0 = 1$ to estimate intercepts

Example



5
4
3
2
1
0
y
0
1
2
3
x
β = 0.8
1
Intercept = 2


5
4
3
2
y
1
2
x

Logistic Regression

Building a linear model for **binary data**.
Recall our original specification: $E[Y|X] = f(x'\beta)$

The Response y is 0 or 1, leading to a conditional mean:

$$E[y|x] = P[y = 1|x] \times 1 + P[y = 0|x] \times 0 = P[y = 1|x]$$

$\implies$ The expectation is a probability.

The 'logit' link is common, for a few reasons. One big reason?

$$log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 x_1 + ... + \beta_K x_K$$

This is a linear model for log odds.

Deviance

Deviance refers to the distance between our fit and the data. You generally want to minimize it.

$$Dev(\beta) = -2log(L(\beta|data)) + C$$

We can ignore C for now.

Deviance is useful for comparing models. It is a measure of GOF that is similar to the residual sum of squares, for a broader class of models (logistic regression, etc).

We'll think about deviance as a cost to be minimized.

Minimize Deviance $\iff$ Maximize likelihood

Bringing this full circle.

$$\phi(x) = \frac{1}{\sqrt{2\pi}} exp\left(-\frac{x^2}{2}\right)$$

Given *n* independent observations, the likelihood becomes:

$$\prod_{i=1}^{n} \phi\left(\frac{y - x'\beta}{\sigma}\right) \propto \prod_{i=1}^{n} exp\left(-\frac{(y - x'\beta)^2}{2\sigma^2}\right)$$

$$\propto exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{n}(y - x'\beta)^2\right)$$

This leads to Deviance of:

$$Dev(\beta) = -2log(L(\beta|data)) + C = \frac{1}{\sigma^2}\sum_{i=1}^{n}(y - x'\beta)^2 + C'$$

So:
Min Deviance $\iff$ Max likelihood $\iff$ Min l_2 loss

This is just a particular loss function, which is driven by the distribution of the ϵ terms.

MLE for Logistic Regression

Our logistic regression has the following likelihood:

$$L(\beta) = \prod_{i=1}^{n} P[y_i|x_i] = \prod_{i=1}^{n} p_i^{y_i}(1-p_i)^{1-y_i}$$

$$= \prod_{i=1}^{n} \left(\frac{exp(x_i'\beta)}{1+exp(x_i'\beta)}\right)^{y_i} \left(\frac{1}{1+exp(x_i'\beta)}\right)^{1-y_i}$$

Thus the deviance to minimize is:

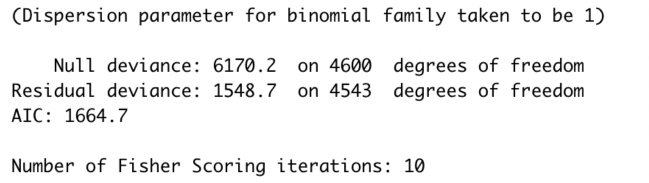
$$Dev(\beta) = -2\sum_{i=1}^{n}(y_i log(p_i) + (1-y_i)log(1-p_i))$$

$$\propto \sum_{i=1}^{n}[log(1+exp(x_i'\beta)) - y_i x_i'\beta]$$

This is just taking the logs and removing the factor of 2.

Back to our summary outputs. We can print the same output for both linear and logistic regressions.

But the "dispersion parameter" is always 1 for the logistic regression. `summary(spam)`



```
(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 6170.2  on 4600  degrees of freedom
Residual deviance: 1548.7  on 4543  degrees of freedom
AIC: 1664.7

Number of Fisher Scoring iterations: 10
```

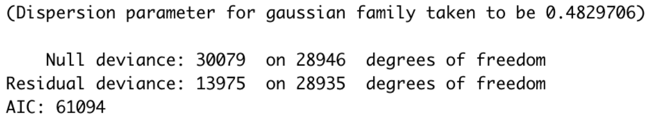
'degrees of freedom' is actually 'number of observations - df' where df is the number of coefficients estimated in the model.

Specifically `df(deviance) = nobs - df(regression)`

You should be able to back out number of observations from the R output.

Dispersion parameter for Linear regression?

`summary(reg.bse)`



```
(Dispersion parameter for gaussian family taken to be 0.4829706)

    Null deviance: 30079  on 28946  degrees of freedom
Residual deviance: 13975  on 28935  degrees of freedom
AIC: 61094
```

Remember our basic gaussian model was:

$$Y|X \sim N(X'\beta, \sigma^2)$$

And the implied deviance was:

$$Dev(\beta) = \frac{1}{\sigma^2}\sum_{i=1}^{n}(y - x'\beta)^2 + C'$$

σ is the dispersion parameter, and it is critical here. The logit has a mean-variance link, so we don't need the separate param.

Estimating σ

$$y_i = x_i'\beta + \epsilon_i; \quad \sigma^2 = Var(\epsilon)$$

Denote the residuals, $r_i = y_i - x_i'\hat{\beta}$.

$$\hat{\sigma}^2 = \frac{1}{n-p-1}\sum_{i=1}^{n} r_i^2$$

R calls $\hat{\sigma}^2$ the dispersion parameter.

Critically, even if we know β, we only predict sales with uncertainty.
E.g., approximately a 95% chance of sales in $x'\beta \pm 2\sqrt{0.48}$

R^2

Residual Deviance, D is what we've minimized using x.
Null Deviance D_0 is for the model without x (or more generally, the model under the null).
i.e. $\hat{y}_i = \bar{y}$

- $D_0 = \sum (y_i - \bar{y})^2$ in linear regression
- $D_0 = -2 \sum [y_i log(\bar{y}) + (1 - y_i) log(1 - \bar{y})]$ in logits

The difference between D and D_0 comes from information in x.

Proportion of deviance explained by x is called the R^2 in a linear regression, "Pseudo-R^2" in logit.

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

This measures how much variability you explain with your model.

- In `spam`: $R^2 = 1 - 1549/6170 = 0.75$
- In OJ – `reg.bse`: $R^2 = 1 - 13975/30079 = 0.54$

R^2 in linear regression

Recall that for linear model, deviance is the sum of squared errors (SSE) and D_0 is the total sum of squares (TSS).

$$R^2 = 1 - \frac{SSE}{TSS}$$

You may also recall that $R^2 = corr(y, \hat{y})^2$.

```r
cor(reg.bse$fitted,oj$logmove)^2
```

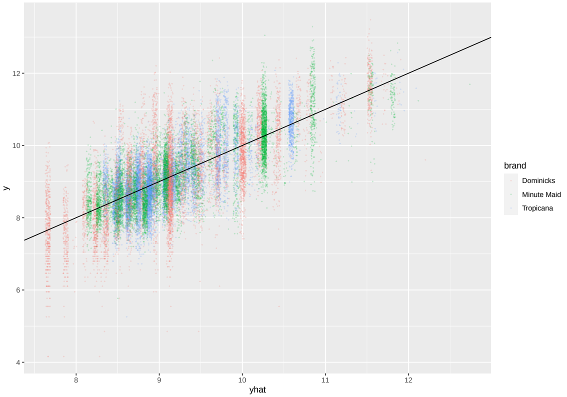
```
## [1] 0.5353939
```

For linear regression, min deviance $=$ max corr$(y, \hat{y})$. $\implies$ if y vs $\hat{y}$ is a straight line, you have a perfect fit.

Also implies that R^2 (weakly) increases whenever we add another variable.

Fit plots

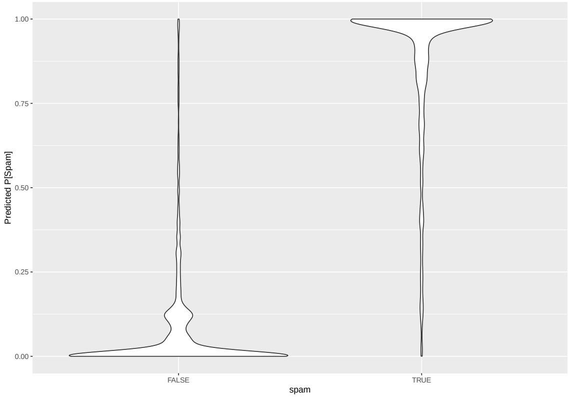
```
fitplotdf = data.frame(y = oj$logmove,yhat= predict(reg.bse
ggplot(fitplotdf,aes(y=y,x=yhat,col=brand)) +
  geom_jitter(alpha=0.2,size=0.2,width=0.03)+
  geom_abline(intercept=0,slope=1)
```



It is good practice to plot y vs $\hat{y}$. It helps you check for

Fit plots – logit

```
logit.fit.df = data.frame(spam=email$spam==1,yhat=predict(s
ggplot(logit.fit.df,aes(y=yhat,x=spam,)) +
  geom_violin()+ylab("Predicted P[Spam]")
```



New question: how do you choose the classification threshold? When do you send to the spam folder?

Prediction

Prediction is easy with `glm`.

```
predict(spam,newdata=email[1:4,])
```

```
##         1         2         3         4
##  2.029963 10.956507 10.034045  5.656989
```

But this output is $x'\beta$. To get probabilities $exp(x'\beta)/(1 + exp(x'\beta))$, add `type="response"`

```
predict(spam,newdata=email[1:4,],type="response")
```

```
##         1         2         3         4
## 0.8839073 0.9999826 0.9999561 0.9965191
```

Newdata must match the format of the original data.

Out-of-Sample Prediction

We care about how well our model works out-of-sample.

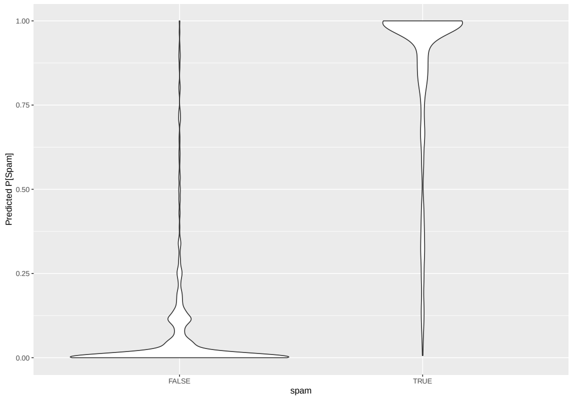
One way to test this is to use a "validation sample".

1. Randomly split your data into two samples
   - usually named "testing" and "training".
2. Fit your model using the "training" data
3. Test predictions on the left-out "testing" data.

```r
#Sample 1000 indices
leaveout = sample.int(nrow(email),1000)
#Train the model
spam_train = glm(spam~.,data=email[-leaveout,],family="bino
#Predict performance on the test data.
test_preds = predict(spam_train,newdata=email[leaveout,],ty
```

Out-of-Sample Prediction

Fit plots for those left-out observations.



For the leave out data, we get $D_0 = 1360$ and $D = 362$ for $R^2 = 0.73$.

Since the leave-out sample is random, your results may vary.

Note: OOS R^2 is lower than in-sample $R^2 = 0.75$

Out-of-Sample

More on this next week during model selection.

You might see notions of this in my code for predicting vaccinations.

More GLM

Link Functions Aren't everything

We spoken about GLM giving a basic model of:

$$E[y|x] = f(x'\beta)$$

This is the link function, allowing us to model non-linear functions as "linear in some other domain".

But we haven't spoken explicitly about the error distribution yet.

Error distributions

This lack of commentary on error distributions is odd – because the error distribution is critical to the likelihood, and the likelihood is how we are estimating our model.

We have in fact been baking in basic assumptions about the distributions.

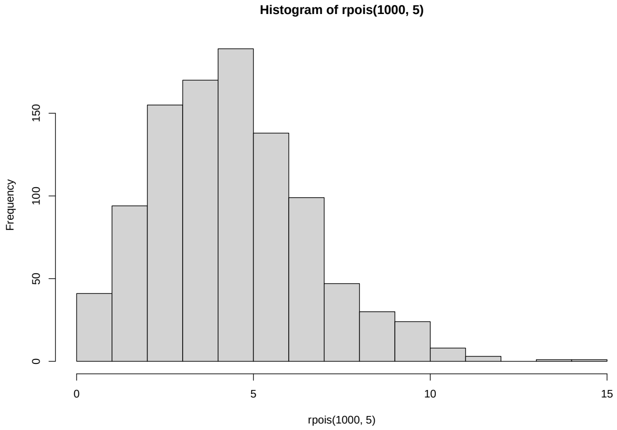
- For the logit: $\epsilon \sim \mathit{Binomial}(1, p)$
- For the linear: $\epsilon \sim N(0, \sigma^2)$

Asymptotically, for linear regressions we can make much weaker statements. See "quasi-maximum likelihood" models. Beyond this course.

Generalizing Further

If we have errors with other distributions, it is useful to take advantage of our knowledge of those other distributions.

A common alternative distribution is the poisson.



GLM with Poisson

Once we say: $\epsilon \sim \mathit{Poisson}(\lambda)$, it implies a likelihood function, and thus can help our estimation routine. It is easy to incorporate this new information.

```
mod = glm(y~.,data=df,family=poisson)
```

We can do the same with many other distributions:

- Easily in base-R for some: Poisson, binomial, Gamma.
- In principle with any distribution: t, Cauchy, etc.

Be aware, that sometimes this imposes a constant variance assumption.

Inference

Sample vs Population

As we've seen, even with very well behaved data, our estimate is not the same as the true value of a parameter.

A large vein of statistics is dedicated to estimating how large a gap there might be.

Typically, this relies on some notion of asymptotic normality of the parameter. With that in place, you can estimate a standard error, and do fairly well.

MOAR DATA

We don't want to do difficult theory work. We just want to know how far away we can expect our parameter to be.

If we had a few more similar samples of data, we could estimate it directly. We could just calculate our parameter in each sample, and look at how much it varies.

We could break our sample into pieces, but each one will be much smaller, and thus higher variance. How can we get more information without doing theory?

The BOOTSTRAP.

Pretend we have 10 thousand samples

Each observation in our data was independently sampled from some distribution.

What if every time we wanted a new sample, we just put all our observations in an urn, drew our n of them (with replacement), or $n - 10$ (without replacement) and calculated the parameter in that sample?

We could do this ten-thousand times. And have a theory-free distribution of possible values our sample could have taken.

This is the simplest form of the bootstrap. And it is powerful.

Example

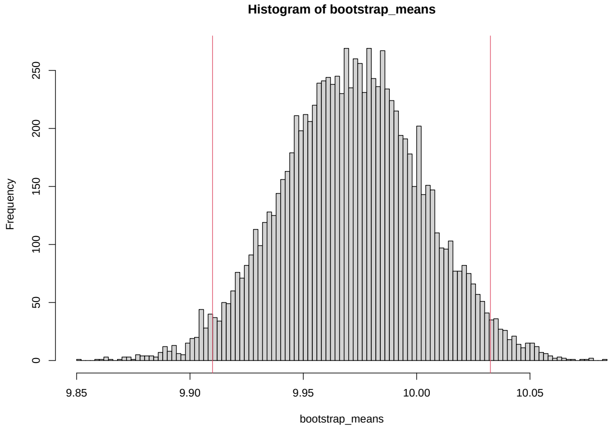
```r
summary(obs)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##    6.578    9.277    9.936    9.971   10.640   12.949
```

```r
bootstrap_means = replicate(10000,{
  new_sample = sample(obs,length(obs),replace=T)
  mean(new_sample)
})
```

Example

```
hist(bootstrap_means,breaks=100)
abline(v=quantile(bootstrap_means,c(0.025,0.975)),col=2)
```



Bootstrap Comments

This relying on the independence between observations. If there is some dependence structure, you can still bootstrap, you just bootstrap clusters.

This will only work well when the parameter of interest has finite variance.

Will only capture all the sampling variation – biased samples are still biased.

IF Time

Go over HW 2 introduction.

Go over HW 1 answers.

Wrap up

Things to do

Before Wednesday:

- HW 2

Rehash

- Regressions minimize prediction errors, loss, or deviance, or they maximize some likelihood function.
- Deviance is a measure of model fit.
- GLM is a fairly flexible tool for modelling in a wide variety of situations – linear only in some dimension.
- Out-of-sample performance is critical. Measuring this is a pain.
- Bootstraps help us simulate the notion of repeated samples.

Bye!