

Stat 111 Spring 2011 Week 10: Generalized Linear Models

1. Poisson Regression

The Normal linear model assumes that the mean of the response variable is a linear function of one or more x variables, and that the errors are Normal. Generalized linear models allow for different error distributions and a more general “link function” between the mean of the response and the linear function of the x 's.

- Explain how the exponential family parameterization suggests a link function for a given error distribution. Use the Normal, Poisson and Gamma distributions as examples.
- Describe an algorithm for finding the maximum likelihood estimate for β in the case of Poisson regression (note that this is also the posterior mode if you assume a flat prior density for β). Show how to get standard errors for the estimates.
- Simulate data to evaluate the performance of such estimates.

2. Logistic regression

Logistic regression may be the most commonly used generalized linear model. If responses are coded 0 or 1, the mean response is the probability of a 1. It wouldn't make sense to model this as a linear function of x because you could end up fitting probabilities outside of $[0, 1]$.

- Show that the Bernoulli and Binomial distributions are exponential families, and identify the canonical parameter and link function. Write out the likelihood function for β .
- Work out an algorithm for maximizing the likelihood (or posterior density) for β . Demonstrate using real data.
- Discuss how to decide whether or not to include predictors in a logistic regression model, and for other generalized linear models.

3. Latent Variables and Probit Regression

Probit regression models the probabilities of success/failure outcomes as depending on one or more possibly continuous covariates. With one predictor variable x , the Probit model can be written as:

$$Y_i | \pi_i \stackrel{\text{indep}}{\sim} \text{Bin}(1, \pi_i), \quad \pi_i = \Phi(\beta_0 + \beta_1 x_i), \quad i = 1, \dots, n,$$

where Φ is the CDF for the standard Normal distribution (the `pnorm` function in R). Probit regression differs from Logistic regression in that the logistic model assumes $\text{logit}(\pi_i) = X_i' \beta$, while the probit model assumes $\Phi^{-1}(\pi_i) = X_i' \beta$ (the functions $\text{logit}()$ and $\Phi^{-1}()$ are the *link functions* for the logistic and probit regression models).

This model implicitly assumes a *latent variable* $\theta_i \stackrel{\text{indep}}{\sim} N(\mu_i, 1)$, where $\mu_i = X_i' \beta$. For $i = 1, \dots, n$, we observe $Y_i = 1$ if $\theta_i \geq 0$ and $Y_i = 0$ if $\theta_i < 0$.

- Show that the latent variable assumption leads to the Probit model.
- Write out the conditional density for $\theta_i | Y_i, \beta$ and derive an expression for $E(\theta_i | Y_i, \beta)$. It will involve $\Phi()$ and $\beta = (\beta_0, \beta_1)'$.
- For the Probit model, the *EM Algorithm* alternates between taking the expectation of $\sum X_i \theta_i$ (the “E-step”) given the Y_i ’s and a current estimate of β , and replacing the estimate of β with its maximum likelihood estimate (the “M-step”) given that the *complete-data sufficient statistic* $S = \mathbf{X}'\theta$ takes its expected value from the E-step. This algorithm converges to the marginal MLE of β (i.e., unconditional on the θ_i ’s). Note that if we assume a prior density $p(\beta) \propto c$, then this is also the *posterior mode* for $f(\beta | y)$. Implement this algorithm in R for estimating winning probabilities for NFL teams, given the Vegas betting spread. Verify that the marginal joint-Likelihood function $L(\beta_0, \beta_1)$ increases at each step in the algorithm.

Problem to turn in: Download the Soccer height and weight data and read it into R:

```
x = scan("/ **specify path ** /Soccer data wk10.txt",what="",sep=",")
# create an nx3 matrix of values and remove first row, which contains headers.
x=matrix(x,ncol=3,byrow=T); x=x[-1,]

# create numeric variables for height, height - and weight, and a categorical variable for p
ht = as.numeric(x[,1])
wt = as.numeric(x[,2])
pos = x[,3]

### make a data frame for these data.
## Some commands in R require you to specify a data frame.
soccer = data.frame(ht, wt, pos)

# make boxplots of the heights and weights by position:
boxplot(wt~pos, data = soccer, ylab="Weight in Pounds",xlab="Position",col="blue")
boxplot(ht~pos, data = soccer, ylab="Weight in Pounds",xlab="Position",col="red")
```

- Make indicator variables for the position categories forward, defender and goalkeeper, and a new height variable centered at 71 inches (very close to the average height). Use these to fit a multiple regression of weight on height and position, with no interaction.

```
x1 = as.numeric(pos=="1: Forward"); x2 = as.numeric(pos=="2: Def")
x3 = as.numeric(pos=="3: Goal"); ht71 = ht-71
out = lm(wt ~ ht71 +x1+x2+x3); out0 = lm(wt ~ ht+pos, data=soccer)
```

The second call to `lm` references the data frame “soccer” (which doesn’t contain the variable `height-71`). This allows you to use categorical variables in the model. Use

summary to compare the output for these two regression models. Find the fitted weights for a 71 inch player in each of the four position categories (midfielder is the baseline position category). Notice that you can only get a standard error directly from the output for one of these fits.

- b) Based on this no interaction model, find a 95% confidence interval for the mean weight of all male college midfielders who are 71 inches tall. Also find a 95% prediction interval for guessing the weight of a particular 71 inch midfielder. Make note of the sum of squares error for part d.
- c) Fit a model that includes an interaction between height and position. I give commands without and with the data frame reference.

```
out1 = lm(wt ~ ht71 + x1+x2+x3 + x1*ht71 + x2*ht71 + x3*ht71)
out2 = lm(wt ~ ht + pos + ht*pos, data = soccer)
```

Work out the equations for the fitted mean weights for each of the four position categories. Make a graph of the data and draw in the four fitted lines. I'll give the commands for the scatterplot and for the line for midfielders. You'll need to add commands for the other three lines.

```
plot(ht, wt, xlab="height", ylab="weight", main="College Soccer Players", type="n")
legend(63,220, c("Midfielder", "Forward", "Defender", "Goalkeeper"),
fill=c("green", "red", "blue", "purple"))
points(ht[pos=="0: Mid"], wt[pos=="0: Mid"], col="green")
points(ht[pos=="1: Forward"], wt[pos=="1: Forward"], col="red")
points(ht[pos=="2: Def"], wt[pos=="2: Def"], col="blue")
points(ht[pos=="3: Goal"], wt[pos=="3: Goal"], col="purple")
abline(-119.322, 3.98593, col="green")
```

- d) Compare the sum of squares error for the model with and without the interaction. Carry out an extra sum of squares F test of the significance of the interaction term. Explain in words what you conclude from this test.

To get the sum of squares from the results of calling `out = lm(...)`, you can type `sse = out$residuals`. Then you can compute the sum of squares. You should also be able to find the sum of squares error from `Rsquare` and by computing the sum of squares total from the response variable directly.