CH 13: Logistic Regression

Motivation

Let's assume that we have access to the underlying candy face off data.

Consider the following model:

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

where $y = 1$ if the i^{th} candy beats smarties and $y_i = 0$ if i^{th} candy does not beat smarties, x_i is an indicator variable that denotes whether the i^{th} candy has chocolate, and $\epsilon_i \sim N(0, \sigma^2)$.

Q: What issues might we have with this model?

Q: What are some possible solutions?

Logistic regression is a special case of *a generalized linear model*

Logistic Regression

The logistic function maps an input from the unit range (0,1) to the real line:

$$logit(x) = \log\left(\frac{x}{1-x}\right)$$

More importantly, the inverse-logit function maps a continous variable to the unit range (0,1)

$$logit(x)^{-1} = \frac{\exp(x)}{1+\exp(x)}$$

.

The `qlogis` (for logit) and `plogis` (inverse-logit) functions in R can be used for this calculation. For instance `plogis(1) =` 0.7310586.

Formally, the inverse-logistic function is used as part of the GLM:

$$y \sim Bernoulli \tag{1}$$

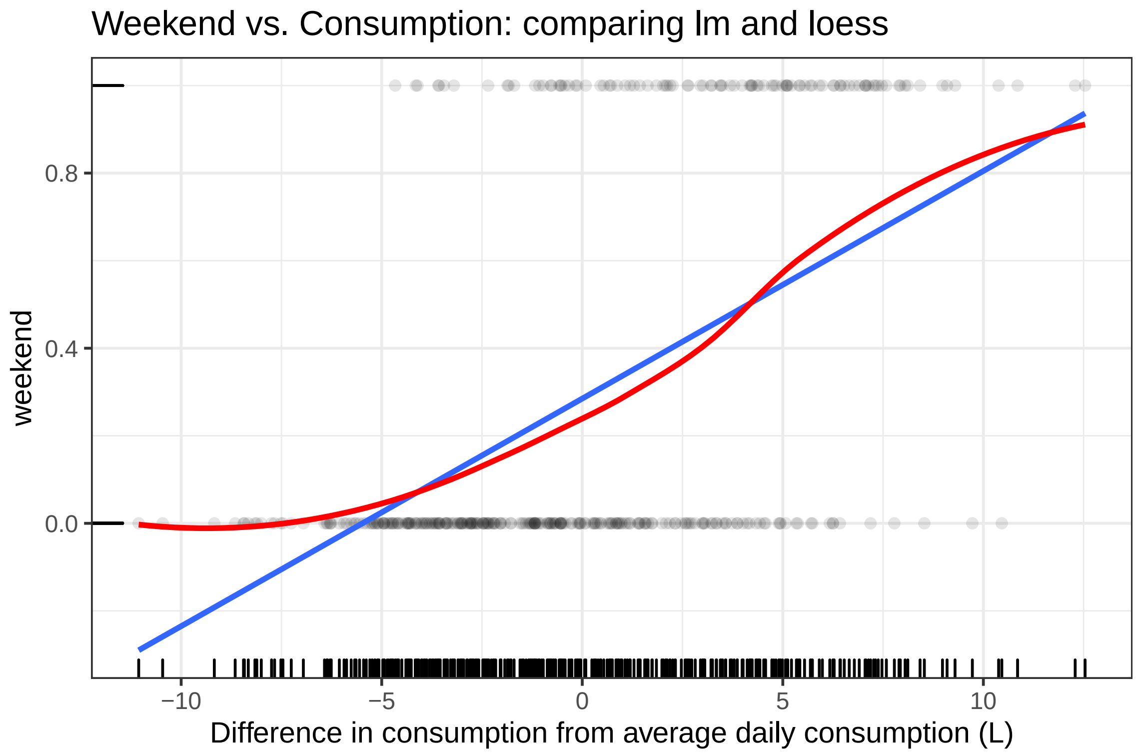
$$Pr(y_i = 1) = \pi_i = logit^{-1}(X\underline{\beta}) \tag{2}$$

Note there is not an ϵ term in this model. The randomness comes from the Bernoulli distribution.

Recall the `beer` dataset, but now instead of trying to model consumption, lets consider whether a day is a weekday or weekend.

```r
beer <- read_csv('http://math.montana.edu/ahoegh/Data/Brazil_cerveja.csv') %>% mutate(consumed = consume
```

```r
beer %>% ggplot(aes(y = weekend, x = consumed)) +
  geom_point(alpha = .1) +
  geom_smooth(formula = 'y~x', method = 'lm', se =F) +
  geom_smooth(formula = 'y~x', method = 'loess', color = 'red', se = F) +
  geom_rug() + ggtitle('Weekend vs. Consumption: comparing lm and loess') +
  theme_bw() + xlab('Difference in consumption from average daily consumption (L)')
```




```r
bayes_logistic <- stan_glm(weekend ~ consumed, data = beer,
                           family = binomial(link = "logit"), refresh = 0)
```

```r
freq_logistic <- glm(weekend ~ consumed, data = beer,
                     family = binomial(link = "logit"))
```

Now how to interpret the model coefficients?

```
bayes_logistic
```

```
## stan_glm
##  family:       binomial [logit]
##  formula:      weekend ~ consumed
##  observations: 365
##  predictors:   2
## ------
##             Median MAD_SD
## (Intercept) -1.2    0.2
## consumed     0.3    0.0
##
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

```
summary(freq_logistic)
```

```
##
## Call:
## glm(formula = weekend ~ consumed, family = binomial(link = "logit"),
##     data = beer)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.0968  -0.6859  -0.4178   0.7367   2.3624
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.24466    0.15059  -8.265   <2e-16 ***
## consumed     0.31791    0.03773   8.427   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 436.21  on 364  degrees of freedom
## Residual deviance: 333.74  on 363  degrees of freedom
## AIC: 337.74
##
## Number of Fisher Scoring iterations: 5
```

Interpreting the coefficients can be challenging due to the non-linear relationship between the outcome and the predictors.

Predictive interpretation

One way to interpret the coefficients is in a predictive standpoint. For instance, consider an day with average consumption, then the probability of a weekend would be `invlogit(-1.2 + 0.3 * 0) =` 0.23, where as the probability of a day with 10 more liters of consumption (relative to an average day) would have a weekend probability of `invlogit(-1.2 + 0.3 * 10) =` 0.86

Of course, we should always think about uncertainty, so we can extract simulations from the model.

`posterior_linpred` was useful with regression

```
new_data <- data.frame(consumed = c(0,10))
posterior_sims <- posterior_linpred(bayes_logistic, newdata = new_data)
summary(posterior_sims)
```

```
##        1                  2
##  Min.   :-1.7585   Min.   :0.917
##  1st Qu.:-1.3492   1st Qu.:1.728
##  Median :-1.2448   Median :1.950
##  Mean   :-1.2482   Mean   :1.960
##  3rd Qu.:-1.1436   3rd Qu.:2.181
##  Max.   :-0.7179   Max.   :3.282
```

This doesn't return probabilities, so we need to consider `posterior_epred` instead

```
posterior_sims <- posterior_epred(bayes_logistic, newdata = new_data)
summary(posterior_sims)
```

```
##        1                 2
##  Min.   :0.1470   Min.   :0.7144
##  1st Qu.:0.2060   1st Qu.:0.8492
##  Median :0.2236   Median :0.8754
##  Mean   :0.2241   Mean   :0.8718
##  3rd Qu.:0.2417   3rd Qu.:0.8985
##  Max.   :0.3279   Max.   :0.9638
```

It can also be useful to consider predictions of an individual data point. *This is how you would conduct posterior predictive checks.*

```
new_obs <- posterior_predict(bayes_logistic, newdata = new_data)
head(new_obs)
```

```
##      1 2
## [1,] 0 0
## [2,] 1 1
## [3,] 1 1
## [4,] 1 0
## [5,] 0 1
## [6,] 0 1
```

```
colMeans(new_obs)
```

```
##       1       2
## 0.22050 0.86425
```

Model Comparison

We can use cross validation in the same manner a standard linear models.

```
loo(bayes_logistic)
```

```
##
## Computed from 4000 by 365 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo   -168.9 10.5
## p_loo         2.0  0.2
## looic       337.8 20.9
## ------
## Monte Carlo SE of elpd_loo is 0.0.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

```
temp_model <- stan_glm(weekend~max_tmp, data = beer, refresh=0)
loo(temp_model)
```

```
##
## Computed from 4000 by 365 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo   -230.0  9.1
## p_loo         2.4  0.2
## looic       460.0 18.3
## ------
## Monte Carlo SE of elpd_loo is 0.0.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```