Logistic Regression

Arnab Maity - Modified by Justin Post

library(MASS)  
library(klaR)  
library(tidyverse)  
library(caret)  
library(rsample)  
library(ISLR2)  
library(knitr)  
library(AppliedPredictiveModeling)  
library(kableExtra)  
library(nnet)  
library(glmnet)

We’ve now discussed the main ideas associated with the predictive modeling tasks:

* Regression task:
  + numeric response variable
  + MSE or RMSE is the commonly used metric
* Classification task:
  + categorical response variable
  + accuracy, confusion matrix-based metrics, and log-loss are common metrics

Now we’ll talk about other models that are commonly used for the classification task!

# Logistic Regression

We talked about two major ways to create classification models:

* Models that directly try to model the conditional class probabilities, . Here is a categorical response taking on values 1, 2, …, K where these values represent categories. We don’t want to treat this variable as numeric.
* Generative models that model ’s distribution and use Bayes’ theorem to obtain estimated conditional class probabilities

Logistic regression takes the first approach.

* LR models the model the probabilities associated with the distribution of as functions of the data vector without actually specifying any distribution of .

Often, logistic regression models are used for their abilities to help us understand the role of the predictors in explaining the outcome (our discrimination task)

## Why not an MLR Model?

We might first ask, why can’t we use our usual MLR model for the classification setting?

Consider just a binary response. Here the variable takes on only two values. Let’s say or .

* What is the mean of the response?
  + Consider a small population of values: 5 1’s and 10 0’s
  + The population mean is just the sample average, .
  + this value is the proportion of 1’s in the population. That is, it represents the probability of obtaining a 1.
  + This means we are really modeling when we have a binary response!

Suppose you have a predictor variable as well, call it

* Given two values of we could model separate proportions
* For a continuous , we could consider a SLR model
* What’s wrong with this? Consider data about [heart disease](https://www4.stat.ncsu.edu/online/datasets/heart.csv)

library(tidyverse)  
heart\_data <- read\_csv("https://www4.stat.ncsu.edu/online/datasets/heart.csv") |>  
 filter(RestingBP > 0) |> #remove one value  
 mutate(HeartDiseaseFac = factor(HeartDisease))  
heart\_data |>   
 dplyr::select(HeartDiseaseFac, everything()) |>  
 slice(1:4) |>  
 kable()

| HeartDiseaseFac | Age | Sex | ChestPainType | RestingBP | Cholesterol | FastingBS | RestingECG | MaxHR | ExerciseAngina | Oldpeak | ST\_Slope | HeartDisease |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 40 | M | ATA | 140 | 289 | 0 | Normal | 172 | N | 0.0 | Up | 0 |
| 1 | 49 | F | NAP | 160 | 180 | 0 | Normal | 156 | N | 1.0 | Flat | 1 |
| 0 | 37 | M | ATA | 130 | 283 | 0 | ST | 98 | N | 0.0 | Up | 0 |
| 1 | 48 | F | ASY | 138 | 214 | 0 | Normal | 108 | Y | 1.5 | Flat | 1 |

* Heart Disease is our response variable. We can look at the mean of the binary variable to estimate

heart\_data |>  
 group\_by(HeartDiseaseFac) |>   
 summarize(prop = n()/nrow(heart\_data)) |>  
 kable()

| HeartDiseaseFac | prop |
| --- | --- |
| 0 | 0.4471101 |
| 1 | 0.5528899 |

* Of course, we may think that these probabilities may differ based on the values of a predictor variable

heart\_data |>  
 group\_by(HeartDiseaseFac) |>  
 summarize(mean\_Age = mean(Age),  
 mean\_RestingBP = mean(RestingBP)) |>  
 kable()

| HeartDiseaseFac | mean\_Age | mean\_RestingBP |
| --- | --- | --- |
| 0 | 50.55122 | 130.1805 |
| 1 | 55.90138 | 134.4497 |

We can try to fit our SLR model with Age as our predictor.

ggplot(heart\_data, aes(x = Age, y = HeartDisease)) +  
 geom\_point() +  
 geom\_smooth(method = "lm")

|  |
| --- |
| The image is a scatter plot with a regression line. The x-axis is labeled "Age" and ranges from 30 to 80, marked at intervals of 10. The y-axis is labeled "HeartDisease" and ranges from 0.00 to 1.00. The plot contains numerous black dots aligned horizontally at the y-values of 0.00 and 1.00, representing data points. A blue regression line stretches from the bottom left to the top right, indicating a positive correlation between age and heart disease. The line gives the estimated probability of heart disease and gives reasonable values across all values of Age plotted. |

That doesn’t quite look right… why? The points all lie on top of one another.

We really are modeling probabilities (proportions) at each value Age can take on. We can visualize this with small Age group means.

* Obtain proportion with heart disease for different Age groups

Age\_x <- seq(from = min(heart\_data$Age), to = max(heart\_data$Age), length = 20)  
heart\_data\_grouped <- heart\_data |>  
 mutate(Age\_groups = cut(Age, breaks = Age\_x)) |>  
 group\_by(Age\_groups) |>  
 summarize(HeartDisease\_mean = mean(HeartDisease), counts = n())  
heart\_data\_grouped |>  
 kable()

| Age\_groups | HeartDisease\_mean | counts |
| --- | --- | --- |
| (28,30.6] | 0.0000000 | 4 |
| (30.6,33.2] | 0.4444444 | 9 |
| (33.2,35.7] | 0.3333333 | 18 |
| (35.7,38.3] | 0.4242424 | 33 |
| (38.3,40.9] | 0.2857143 | 28 |
| (40.9,43.5] | 0.3030303 | 66 |
| (43.5,46.1] | 0.3606557 | 61 |
| (46.1,48.6] | 0.5200000 | 50 |
| (48.6,51.2] | 0.4938272 | 81 |
| (51.2,53.8] | 0.4927536 | 69 |
| (53.8,56.4] | 0.5503876 | 129 |
| (56.4,58.9] | 0.6750000 | 80 |
| (58.9,61.5] | 0.7448980 | 98 |
| (61.5,64.1] | 0.7241379 | 87 |
| (64.1,66.7] | 0.6470588 | 34 |
| (66.7,69.3] | 0.7368421 | 38 |
| (69.3,71.8] | 0.6666667 | 12 |
| (71.8,74.4] | 0.7500000 | 12 |
| (74.4,77] | 0.7142857 | 7 |
| NA | 0.0000000 | 1 |

* Now plot that data as well!

ggplot(data = heart\_data, aes(x = Age, y = HeartDisease)) +  
 geom\_jitter(aes(color = RestingBP)) +  
 geom\_point(data = heart\_data\_grouped, aes(x = Age\_x, y = HeartDisease\_mean, size = counts)) +  
 geom\_smooth(method = "lm", color = "Green")

`geom\_smooth()` using formula = 'y ~ x'

|  |
| --- |
| The image is a scatter plot with many blue data points representing a relationship between "Age" on the x-axis and "HeartDisease" on the y-axis. The x-axis is numbered in increments of 10, ranging from 30 to 70, while the y-axis is labeled from 0.0 to 1.0. A line diagonally intersects the plot, indicating a trend within the data. Around this line the estimated mean for each Age group is plotted as well. The trend line roughly goes through the middle of these Age group means. |

Here SLR isn’t actually terrible! However, theoretically we have some issues.

* Response = success/failure (0/1)
  + We are modeling the average number of successes for a given . This is a probability!
    - predictions should never go below 0
    - predictions should never go above 1

The SLR model doesn’t require this! We just got lucky. Consider a much wider set of Age values.

ggplot(data = heart\_data, aes(x = Age, y = HeartDisease)) +  
 geom\_jitter(aes(color = RestingBP)) +  
 geom\_point(data = heart\_data\_grouped, aes(x = Age\_x, y = HeartDisease\_mean, size = counts)) +  
 geom\_smooth(method = "lm", color = "Green") +  
 xlim(0, 100)

`geom\_smooth()` using formula = 'y ~ x'

|  |
| --- |
| The image is a scatter plot with many blue data points representing a relationship between "Age" on the x-axis and "HeartDisease" on the y-axis. The x-axis is numbered in increments of 10, ranging from 0 to 100, while the y-axis is labeled from 0.0 to 1.0. A line diagonally intersects the plot, indicating a trend within the data. Around this line the estimated proportion of people with heart disease for each Age group is plotted as well. The trend line roughly goes through the middle of these Age group means. While the data for Age only goes from 30 to 80, the plot shows that the line would estimate a probability smaller than 0 for Age less than 20 and would estimate a probability greater than 1 for an age above 85. |

The predicted probability (the value of the line) will go below 0 and above 1 sometimes!

Ideally, we should consider a model that requires this. One option is a logistic regression model.

## Logistic Regression Model

* The basic Logistic Regression model uses a *logistic function* to model the success probability:

This is a nonlinear function of the ’s that never goes below 0 or above 1!

|  |
| --- |
| A line graph displays the relationship between a predictor x, located on the x-axis, and the probability of success, located on the y-axis. A few curves are shown representing different logistic curves. A curve with intercept equal to -5 and slope equal to 11 starts near 0, increases rapidly, and then asymptotes near 1.  A line with intercept equal to 10 and slope equal to -11 starts near 1, decreases, and then asymptotes near 0. |

This is actually a non-linear model! It is much harder to fit the model…

* The logistic regression model doesn’t have a closed form solution (maximum likelihood often used to fit parameters)

### Interpreting the Model

How can we relate this to our usual linear model?

* Back-solving shows the *logit* or *log-odds* of success is linear in the parameters

- Coefficient interpretation changes greatly from linear regression model!

* + is call the odds of success
  + represents a change in the log-odds of success

Let’s plot the logistic regression model that uses Age to predict HeartDisease:

|  |
| --- |
| The image is a scatter plot showing the relationship between "Age" on the x-axis and "Logistic_Pred" on the y-axis. The plot is overlaid with a logistic curve. The x-axis ranges from 30 to 80, whereas the y-axis ranges from 0.00 to 1.00. Bubbles representing the estimated proportion of people with heart disease for age groups are shown and the fitted logistic curve travels roughly through these points in a non-linear manner. |

This is a sigmoid function that looks linear close up! Let’s zoom out:

|  |
| --- |
| The same graph as the previous is shown except the Age range is wider, from 0 to 125. With this wider range, the fitted logistic curve shows the sigmoid type shape that starts near 0, increases roughly through the estimated proportion of people with heart disease group means, and asymptotes near 1. |

Nice!

### Including More Than One Predictor

In the MLR model we could easily add other numeric predictors, interactions, quadratics, and categorical predictors via the use of indicator variables. We can do this here too!

* Adding a dummy variable corresponding to a binary predictor just changes the ‘intercept’
  + Call Age
  + Let’s add a predictor for Sex ( if ‘Male’ and if “Female”)

|  |
| --- |
| The image is a scatter plot showing the relationship between "Age" on the x-axis and "Logistic_Pred" on the y-axis. The plot is overlaid with two logistic curves, one fit for the Males and the dataset and one fit for the females in the dataset. The x-axis ranges from 30 to 80, whereas the y-axis ranges from 0.00 to 1.00. Bubbles representing the estimated proportion of people with heart disease for age groups are shown for both males and females. Across the plot, the male proportions are generally about 0.2 higher than the corresponding female proportions. The fitted logistic curves roughly travel through these estimated proportion, except for a few values that are estimated to be zero for the females. |

We can see that we don’t have parallel lines in this case. Let’s investigate the model fit a bit more.

* Similar to the lm() function to fit MLR models, glm() with family = "binomial" allows us to fit logistic regression models.

log\_reg\_fit <- glm(HeartDiseaseFac ~ Age + Sex,   
 data = heart\_data, family = "binomial")  
summary\_fit <- summary(log\_reg\_fit)  
summary\_fit$coefficients |>  
 kable()

|  | Estimate | Std. Error | z value | Pr(>|z|) |
| --- | --- | --- | --- | --- |
| (Intercept) | -4.631848 | 0.4810615 | -9.628389 | 0 |
| Age | 0.066476 | 0.0081630 | 8.143581 | 0 |
| SexM | 1.638859 | 0.1893462 | 8.655358 | 0 |

What are our fitted equations for Males? For Females?

* Female model:
* Male model:
* If we fit an interaction term between Age and our Sex dummy variable, we essentially fit two separate logistic regression models

log\_reg\_fit\_int <- glm(HeartDiseaseFac ~ Age + Sex + Age\*Sex,   
 data = heart\_data, family = "binomial")  
summary\_fit\_int <- summary(log\_reg\_fit\_int)  
summary\_fit\_int$coefficients |>  
 kable()

|  | Estimate | Std. Error | z value | Pr(>|z|) |
| --- | --- | --- | --- | --- |
| (Intercept) | -4.2045323 | 1.0497495 | -4.0052720 | 0.0000619 |
| Age | 0.0587056 | 0.0188855 | 3.1084912 | 0.0018805 |
| SexM | 1.1201380 | 1.1553058 | 0.9695598 | 0.3322660 |
| Age:SexM | 0.0095098 | 0.0209435 | 0.4540682 | 0.6497797 |

* Let’s check out the visual of the fits

|  |
| --- |
| The image is a scatter plot showing the relationship between "Age" on the x-axis and "Logistic_Pred" on the y-axis. The plot is overlaid with two logistic curves, one fit for the Males and the dataset and one fit for the females in the dataset. The x-axis ranges from 30 to 80, whereas the y-axis ranges from 0.00 to 1.00. Bubbles representing the estimated proportion of people with heart disease for age groups are shown for both males and females. Across the plot, the male proportions are generally about 0.2 higher than the corresponding female proportions. The fitted logistic curves roughly travel through these estimated proportion, except for a few values that are estimated to be zero for the females. The estimated curves are slightly more different here than in the previous plot. |

* We can also include more than one numeric predictor and add in polynomial terms increases flexibility as well!
  + The plot below shows a model with a linear and quadratic term included

|  |
| --- |
| A line graph displays the relationship between a predictor x, located on the x-axis, and the probability of success, located on the y-axis. A logistic curve with intercept equal to -1, slope on a linear x term equal to 3, and slope on a quadratic x term equal to -1 is shown. The curve initially starts near 0 and increases for a bit before beginning to decrease towards 0. The graph essentially looks like a mound, indicating that logistic curves can be reasonably flexible given enough terms. |

### General Logistic Regression Model

Suppose we have a set of covariates and a binary response.

* We can model the conditional (posterior) probabilities as follows:
* Since we have only two classes for , we know . This means we can write the following:

Another way to write the same model is using the *log-odds*:

* The parameters quantify the impact of the covariates to the prediction of
* The ’s are still called slope terms and we often test vs .
* The group used in the denominator (class 0 in our formulation above) is called the *reference group*. The choice of reference group is arbitrary as the estimates of the conditional (posterior) probabilities are same.

### Fitting the Logistic Regression Model

For the MLR model we said we could use least squares or maximum likelihood to fit the model and the fit (estimated coefficients) would be the same.

* Here, the use of least squares doesn’t make sense
* Maximum likelihood still works! However, we don’t get a closed form solution as we did in the MLR case
  + We can show that the equation to find the ‘best’ ’s is equivalent to minimizing the negative log loss over the training data!

# Inference on the Logistic Regression Model

## Classification Using the Logistic Regression Model

We estimate our conditional probabilities using the fitted model:

As with our model, we can predict the class for a particular observation via the following rule:

The item is classified in group if , otherwise in group .

* Again, in the binary case, this is equivalent to

Logistic regression can be performed using the glm() function in base R. **Make sure your response variable is a factor.** Use family = "binomial" to specify the logistic regression model.

* We use the same formula notation as before!
  + response ~ predictors
  + Separate predictor terms with +
  + I() or poly() for polynomial terms
  + pred1\*pred2 for an interaction
  + pred1:pred2 for main effects and an interaction
  + glm() automatically creates dummy variables

Let’s fit a logistic regression model for our HeartDisease response. We’ll use Age, RestingBP, and Sex as predictors.

# Logistic regression  
heart\_glm = glm(HeartDiseaseFac ~ Age + RestingBP + Sex,   
 family = "binomial",   
 data = heart\_data)

The estimated coefficients are as follows.

heart\_coef <- heart\_glm$coefficients  
heart\_coef |>  
 kable()

|  | x |
| --- | --- |
| (Intercept) | -5.3488430 |
| Age | 0.0633399 |
| RestingBP | 0.0066189 |
| SexM | 1.6481678 |

We can interpret as the *log-odds* when both Age and RestingBP are zero and Sex is Female.

* We have odds of heart disease vs not having heart disease in this case.
* Of course, zero values for Age and RestingBP don’t make sense!

The estimated value of can be interpreted as the amount the log-odds will *change* due to one unit *increase* in Age while *keeping the RestingBP level fixed* for a Sex = Female.

* We expect a unit change in log-odds.
* Equivalently, our odds will change by a multiplicative factor of ()
  + In other words, increase by percent ()

Suppose we have a new sample with Age = 60 and RestingBP = 130. We can compute the estimated conditional (posterior) probabilities of heart disease for Males and Females as follows:

In R, we can simply use the predict() function to compute the probabilities shown above.

newx <- data.frame(Age = c(60, 60),  
 RestingBP = c(130, 130),  
 Sex = c("M", "F"))  
predict(heart\_glm,   
 newdata = newx,  
 type = "response")

1 2   
0.7231577 0.3344806

* We can see that a Male with these values of Age and RestingBP is classified as having heart disease but a female would not.

Let’s check out our confusion matrix.

# Confusion matrix  
conf\_mat <- confusionMatrix(table(predicted = ifelse(predict(heart\_glm, type = "response") > 0.5,  
 1, 0),  
 heart\_data$HeartDiseaseFac))  
conf\_mat$table |>   
 kable()

|  | 0 | 1 |
| --- | --- | --- |
| 0 | 226 | 105 |
| 1 | 184 | 402 |

stats <- data.frame(Statistic = names(conf\_mat$overall), Value = conf\_mat$overall)  
stats |>  
 kable(row.names = FALSE)

| Statistic | Value |
| --- | --- |
| Accuracy | 0.6848419 |
| Kappa | 0.3505809 |
| AccuracyLower | 0.6536690 |
| AccuracyUpper | 0.7148220 |
| AccuracyNull | 0.5528899 |
| AccuracyPValue | 0.0000000 |
| McnemarPValue | 0.0000045 |

As we have only a couple of predictors, we can plot the decision boundary of the classifier!

* Here we need to find all values of Age and RestingBP that have the same conditional (posterior) probability of being heart disease or not (once for males and then once for females).
* The boundary (for females) is given by all the solutions of the linear equation (the estimated formula of log-odds):

|  |
| --- |
| This plot is for only female data. The image shows a scatterplot with Age on the x-axis ranging from 30 to 75 and RestingBP on the y-axis ranging from 75 to 200. Points on the plot are displayed as either having heart disease or not having heart disease. Points for larger Age values and larger RestingBP generally show more cases of heart disease. The decision boundary for classifying a point as having heart disease is displayed. This is a line that starts at age of 65 and restingBP of 200 and ends at age of 78 and restingBP of 75. All points to the upper right of this line would be assigned to have heart disease whereas points to the left of this line would be classified to have no heart disease. This decision boundary does not look good when comparing to the points in the data that have heart disease. The model is clearly under predicting heart disease for females.  Decision boundary of a 2-class logistic regression based classifier. |

Now the confusion matrix for Females:

conf\_mat\_female <- confusionMatrix(table(predicted = ifelse(predict(heart\_glm,   
 newdata = heart\_data |> filter(Sex == "F"),   
 type = "response") > 0.5,  
 1, 0),  
 heart\_data |>   
 filter(Sex == "F") |>   
 pull(HeartDiseaseFac)  
 ))  
conf\_mat\_female$table |>  
 kable()

|  | 0 | 1 |
| --- | --- | --- |
| 0 | 140 | 47 |
| 1 | 3 | 3 |

stats\_female <- data.frame(Statistic = names(conf\_mat\_female$overall), Value = conf\_mat\_female$overall)  
stats\_female |>  
 kable(row.names = FALSE)

| Statistic | Value |
| --- | --- |
| Accuracy | 0.7409326 |
| Kappa | 0.0546630 |
| AccuracyLower | 0.6731135 |
| AccuracyUpper | 0.8011915 |
| AccuracyNull | 0.7409326 |
| AccuracyPValue | 0.5379395 |
| McnemarPValue | 0.0000000 |

* The boundary (for males) is given by all the solutions of the linear equation (the estimated formula of log-odds):

|  |
| --- |
| This plot is only for male data. The image shows a scatterplot with Age on the x-axis ranging from 30 to 75 and RestingBP on the y-axis ranging from 75 to 200. Points on the plot are displayed as either having heart disease or not having heart disease. Points for larger Age values and larger RestingBP generally show more cases of heart disease. The decision boundary for classifying a point as having heart disease is displayed. This is a line that starts at age of 40 and restingBP of 200 and ends at age of 50 and restingBP of 75. All points to the upper right of this line would be assigned to have heart disease whereas points to the left of this line would be classified to have no heart disease. This decision boundary appears to separate the data reasonably well.  Decision boundary of a 2-class logistic regression based classifier. |

Now the confusion matrix for Males:

conf\_mat\_male <- confusionMatrix(table(predicted = ifelse(predict(heart\_glm,   
 newdata = heart\_data |> filter(Sex == "M"),   
 type = "response") > 0.5,  
 1, 0),  
 heart\_data |>   
 filter(Sex == "M") |>   
 pull(HeartDiseaseFac)  
 ))  
conf\_mat\_male$table |>  
 kable()

|  | 0 | 1 |
| --- | --- | --- |
| 0 | 86 | 58 |
| 1 | 181 | 399 |

stats\_male <- data.frame(Statistic = names(conf\_mat\_male$overall), Value = conf\_mat\_male$overall)  
stats\_male |>  
 kable(row.names = FALSE)

| Statistic | Value |
| --- | --- |
| Accuracy | 0.6698895 |
| Kappa | 0.2158537 |
| AccuracyLower | 0.6343052 |
| AccuracyUpper | 0.7040823 |
| AccuracyNull | 0.6312155 |
| AccuracyPValue | 0.0165364 |
| McnemarPValue | 0.0000000 |

* Perhaps an interaction term would help us here!

Note: **Ideally, we should use cross-validation, training-testing sets to estimate the accuracy, as we have learned before!**

## Odds and log-odds

Let’s investigate the concepts of *odds* and *log-odds* in more detail.

Reconsider the heart disease data with classes (1 and 0) and only one covariate, $X = `Age`$.

* The odds are defined as
  + If then the odds of class 1 are .
  + In contrast, if then the odds are
* Thus greater odds relate to higher posterior probability of class 1.
* In terms of log-odds,
  + Model implies that for a one unit increase in (Age), the log-odds will change by units
  + This change in log-odds does *not* depend on the value of , just that it increases by 1
  + Equivalently, due to one unit increase in , the *odds* gets multiplied by .

## Hypothesis testing and confidence intervals

Reconsider our logistic regression model with Age, RestingBP, and Sex. We can get a summary of the fit as follows.

summary(heart\_glm)$coefficients |>  
 kable()

|  | Estimate | Std. Error | z value | Pr(>|z|) |
| --- | --- | --- | --- | --- |
| (Intercept) | -5.3488430 | 0.6684218 | -8.002197 | 0.0000000 |
| Age | 0.0633399 | 0.0083843 | 7.554615 | 0.0000000 |
| RestingBP | 0.0066189 | 0.0042047 | 1.574165 | 0.1154494 |
| SexM | 1.6481678 | 0.1900027 | 8.674444 | 0.0000000 |

* summary() of the fit produces -tests for coefficient of each covariate
  + This is an approximate (large sample) test
  + We test whether vs after accounting for the other predictors in the model
  + The test statistic is
  + This statistics approximately follows a distribution under . We reject for large (in absolute value) values.
  + P-value is

We can produce large sample confidence intervals for as

where is the quantile of the distribution. For example, a confidence intervals for is

We can interpret this intervals as follows:

* with every unit increase in the value of (Age in our example), we can expect an *increase in log-odds* by an amount of 0.0469067 to 0.0797731 (while holding RestingBP and Sex constant)
* Equivalently, every unit increase in the value of , the *odds* will be changed by a factor of 1.0480243 to 1.0830413 (while holding RestingBP and Sex constant)
  + In other words, the increase in odds will be between 4.802 percent and 8.304 percent

# Logistic regression with multiple classes

We can extend logistic regression presented for two classes to the case of multiple classes.

* We call this a *Multinomial Logistic Regression* model.
* Suppose we have classes, and we take the -th class as the reference. The log-odds of classes vs the reference class are modeled as follows:

Some algebra shows that the corresponding conditional (posterior) probabilities are as follows:

We can similarly build a classification rule as follows.

An item with covariate is predicted to be in class if the estimated probability is larger than the other posterior probabilities.

# Issues to consider

## Separation of Data

There are some situations where logistic regression might not perform well. One such situation is *complete (or quasi-complete) separation* of the data.

This situation happens when the outcome variable separates a predictor completely. This leads to perfect prediction of the outcome by the predictor.

* Consider the following data set with binary response and two predictors and .
* The figure below shows relationship between and the two predictors.
* In such a case, logistic regression may produce unreasonable over-inflated estimates of regression coefficients.

|  |
| --- |
| A scatterplot is shown between two variables: x1 and x2. The data points in the upper right all represent class 1 and the data points in the bottom left all indicate class 2. There is a clear and complete separation of data points by a diagonal line going from the top left to the bottom right.  Simulated data set. The response separates the data completely – the boundary (dashed line) is . Negative values of corresponds to , and positive values corresponds to . |

fit <- glm(y ~ x1 + x2, family = binomial())

Warning: glm.fit: algorithm did not converge

Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

In general, if there is a linear combination that is completely separated by , logistic regression will fail to produce reasonable results! Figure shows one such example where the data is completely separated by the line .

* Although the examples above shows complete separation using continuous predictors, it is more like to happen when using categorical predictors coded by dummy variables. Small sample size might contribute to this problem as well.
  + In such situations, applying other classification methods (e.g., LDA) is preferred.

## Sample Size

Since logistic regression usually deals with binary outcome, often it requires a larger sample size that linear regression.

Multinomial logistic regression requires even more sample size than binary logistic regression due to the fact that it estimates parameters for multiple classes.

## Combinations Present in the Data

In the presence of categorical predictors, it might happen that there are some combinations of predictor and response values that are not present in the data.

In such a case, logistic fit may become unstable, or might even fail to converge.

# High-Dimensional Problems

When the number of predictors is larger than (or close to) the sample size , the methods described in this section suffer from numerical instability or simply can not be applied to the data. We saw these issues in the MLR case as well!

* We can apply similar strategies discussed for linear regression here as well: regularization/shrinkage and dimension reduction methods.
* Like linear regression, we can develop ridge, lasso and elastic net methods for logistic regression.
* All these methods are available in glmnet() package.
  + As before, these methods shrink the regression coefficients towards zero and can be used in high-dimensional setting.
  + We show the lasso based logistic regression fit of the heart disease data with the penalty parameter chosen by CV below.
  + We only include numeric predictors here but could add in our own dummy variables for each categorical predictor
  + Note: **glmnet() automatically scales the predictors before estimating the regression coefficients, and then outputs the coefficients in the original scale**

set.seed(1102)  
# CV to choose lambda  
logit\_cv <- cv.glmnet(x = as.matrix(heart\_data |>   
 dplyr::select(-starts\_with("Heart"), -Sex, -ChestPainType, -RestingECG, -ExerciseAngina, -ST\_Slope)),   
 y = heart\_data$HeartDiseaseFac,  
 family = binomial(),  
 alpha = 1)

|  |
| --- |
| A plot displays the relationship between log of lambda on the x-axis (ranging from -7 to -1) and GLM Deviance on the y-axis. The deviance (a measure of error) starts off lower before increasing around log(lambda) equal to -4. Standard error bars are present. The minimum error occurs around log(lambda) equal to -6 but the one standard error minimum occurs around log(lambda) equal to -3.  CV results for logistic regression of wines data with lasso penalty. |

# Final fit with lambda chosen by 1-SE rule  
heart\_lasso <- glmnet(x = as.matrix(heart\_data |>   
 dplyr::select(-starts\_with("Heart"), -Sex, -ChestPainType, -RestingECG, -ExerciseAngina, -ST\_Slope)),  
 y = heart\_data$HeartDiseaseFac,  
 family = binomial(),  
 alpha = 1,  
 lambda = logit\_cv$lambda.1se)

# Estimated coefs  
coef\_mat <- coef(heart\_lasso) |>   
 as.matrix()  
dimnames(coef\_mat)[[2]] <- c("Estimate")  
coef\_mat |>  
 kable()

|  | Estimate |
| --- | --- |
| (Intercept) | 2.8547941 |
| Age | 0.0005391 |
| RestingBP | 0.0000000 |
| Cholesterol | -0.0016404 |
| FastingBS | 0.6981183 |
| MaxHR | -0.0220519 |
| Oldpeak | 0.6536631 |

Dimension reduction method like PCA can still be applied to the predictors before building classifiers!