Logistic Regression Notes

Lt Col Ken Horton Professor Bradley Warner

31 July, 2020

Objectives

1)

2)

3)

Logistic regression

In this lesson we introduce **logistic regression** as a tool for building models when there is a categorical response variable with two levels. Logistic regression is a type of **generalized linear model** (GLM) for response variables where regular multiple regression does not work very well. In particular, the response variable in these settings often takes a form where residuals look completely different from the normal distribution. We are prepping you for Math 378, where we will explore predictive models of many different types including ones that don't assume an underlying functional relationship between inputs and outputs. So cool!

GLMs can be thought of as a two-stage modeling approach. We first model the response variable using a probability distribution, such as the binomial or Poisson distribution. Second, we model the parameter of the distribution using a collection of predictors and a special form of multiple regression.

We will be using the `email` data set for our work in this lesson. These emails were collected from a single email account, and we will work on developing a basic spam filter using these data. The response variable, `spam`, has been encoded to take value 0 when a message is not spam and 1 when it is spam. Our task will be to build an appropriate model that classifies messages as spam or not spam using email characteristics coded as predictor variables. While this model will not be the same as those used in large-scale spam filters, it shares many of the same features.

Email data

In the `email` data set there are many variables available that might be useful for classifying spam. Descriptions of these variables are presented in help menu in `R` under the `openintro` package. The `spam` variable will be the outcome, and the other 10 variables will be the model predictors. While we have limited the predictors used in this section to be categorical variables (where many are represented as indicator variables), numerical predictors may also be used in logistic regression. See the footnote for an additional discussion on this topic.[1]

[1] Recall that if outliers are present in predictor variables, the corresponding observations may be especially influential on the resulting model. This is the motivation for omitting the numerical variables, such as the number of characters and line breaks in emails. These variables exhibited extreme skew. We could resolve this issue by transforming these variables (e.g. using a log-transformation), but we will omit this further investigation for brevity.

variable	description
spam	Specifies whether the message was spam.
to_multiple	An indicator variable for if more than one person was listed in the **To** field of the email.
cc	An indicator for if someone was CCed on the email.
attach	An indicator for if there was an attachment, such as a document or image.
dollar	An indicator for if the word "dollar" or dollar symbol ($) appeared in the email.
winner	An indicator for if the word "winner" appeared in the email message.
inherit	An indicator for if the word "inherit" (or a variation, like "inheritance") appeared in the email.
password	An indicator for if the word "password" was present in the email.
format	Indicates if the email contained special formatting, such as bolding, tables, or links.
re_subj	Indicates whether "Re:" was included at the start of the email subject.
exclaim_subj	Indicates whether any exclamation point was included in the email subject.

Modeling the probability of an event3

The outcome variable for a GLM is denoted by Y_i, where the index i is used to represent observation i. In the email application, Y_i will be used to represent whether email i is spam ($Y_i = 1$) or not ($Y_i = 0$).

The predictor variables are represented as follows: $x_{1,i}$ is the value of variable 1 for observation i, $x_{2,i}$ is the value of variable 2 for observation i, and so on.

Logistic regression is a generalized linear model where the outcome is a two-level categorical variable. The outcome, Y_i, takes the value 1 (in our application, this represents a spam message) with probability p_i and the value 0 with probability $1-p_i$. It is the probability p_i that we model in relation to the predictor variables.

The logistic regression model relates the probability an email is spam (p_i) to the predictors $x_{1,i}$, $x_{2,i}$, ..., $x_{k,i}$ through a framework much like that of multiple regression:

$$transformation(p_i) = \beta_0 + \beta_1 x_{1,i} + \beta_2 x_{2,i} + \cdots \beta_k x_{k,i}$$

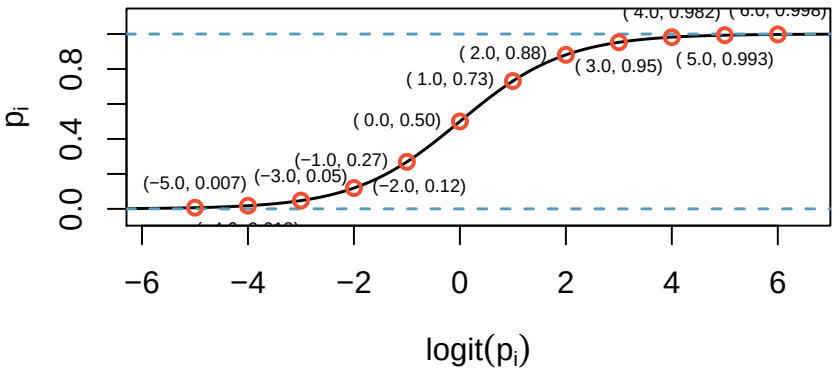
We want to choose a transformation that makes practical and mathematical sense. For example, we want a transformation that makes the range of possibilities on the left hand side of the above equation equal to the range of possibilities for the right hand side; if there was no transformation for this equation, the left hand side could only take values between 0 and 1, but the right hand side could take values outside of this range. A common transformation for p_i is the **logit transformation**, which may be written as

$$logit(p_i) = \log_e \left(\frac{p_i}{1 - p_i} \right)$$

Below, we expand the equation using the logit transformation of p_i:

$$\log_e \left(\frac{p_i}{1 - p_i} \right) = \beta_0 + \beta_1 x_{1,i} + \beta_2 x_{2,i} + \cdots + \beta_k x_{k,i}$$

The logit transformation is shown in the next figure.



In our spam example, there are 10 predictor variables, so $k = 10$. This model isn't very intuitive, but it still has some resemblance to multiple regression, and we can fit this model using software. In fact, once we look at results from software, it will start to feel like we're back in multiple regression, even if the interpretation of the coefficients is more complex.

First model

Here we create a spam filter with a single predictor: `to_multiple`. This variable indicates whether more than one email address was listed in the **To** field of the email.

```
email %>%
  select(spam,to_multiple) %>%
  summary()
```

```
##       spam          to_multiple
##  Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.0000   1st Qu.:0.0000
##  Median :0.0000   Median :0.0000
##  Mean   :0.0936   Mean   :0.1581
##  3rd Qu.:0.0000   3rd Qu.:0.0000
##  Max.   :1.0000   Max.   :1.0000
```

In `R` we use the `glm()` function. It has the same formula format but also requires a `family` argument. Since our response is binary, we use `binomial`. If we wanted to use `glm()` for linear regression assuming normally distributed residual, the family argument would be `gaussian`.

```
email_mod <- glm(spam~to_multiple,data=email,family="binomial")
```

```
summary(email_mod)
```

```
##
## Call:
```

```
## glm(formula = spam ~ to_multiple, family = "binomial", data = email)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -0.477  -0.477  -0.477  -0.477   2.809
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -2.11609    0.05618 -37.665  < 2e-16 ***
## to_multiple -1.80918    0.29685  -6.095  1.1e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2437.2  on 3920  degrees of freedom
## Residual deviance: 2372.0  on 3919  degrees of freedom
## AIC: 2376
##
## Number of Fisher Scoring iterations: 6
```

The following logistic regression model was found using `R`:

$$\log\left(\frac{p_i}{1-p_i}\right) = -2.12 - 1.81 \times tomultiple$$

If an email is randomly selected and it has just one address in the **To** field, what is the probability it is spam? What if more than one address is listed in the **To** field?

If there is only one email in the **To* field, then `to_multiple` takes value 0 and the right side of the model equation equals -2.12. Solving for p_i: $\frac{e^{-2.12}}{1+e^{-2.12}} = 0.11$. Just as we labeled a fitted value of y_i with a "hat" in single-variable and multiple regression, we will do the same for this probability: $\hat{p}_i = 0.11$.

If there is more than one address listed in the **To** field, then the right side of the model equation is $-2.12 - 1.81 \times 1 = -3.93$, which corresponds to a probability $\hat{p}_i = 0.02$.

Notice that we could examine -2.12 and -3.93 in our logistic graph to estimate the probability before formally calculating the value.

To convert from values on the regression-scale (e.g. -2.12 and -3.93 in our example}), use the following formula, which is the result of solving for p_i in the regression model:

$$p_i = \frac{e^{\beta_0+\beta_1 x_{1,i}+\cdots+\beta_k x_{k,i}}}{1 \ + \ e^{\beta_0+\beta_1 x_{1,i}+\cdots+\beta_k x_{k,i}}}$$

As with most applied data problems, we substitute the point estimates for the parameters (the β_i) so that we may make use of this formula. In our example, the probabilities were calculated as

$$\frac{e^{-2.12}}{1 \ + \ e^{-2.12}} = 0.11, \frac{e^{-2.12-1.81}}{1 \ + \ e^{-2.12-1.81}} = 0.02$$

While the information about whether the email is addressed to multiple people is a helpful start in classifying email as spam or not, the probabilities of 11% and 2% are not dramatically different, and neither provides very strong evidence about which particular email messages are spam. To get more precise estimates, we'll need to include many more variables in the model.

Model with multiple predictors

Exercise: Fit a logistic regression model with all ten predictors listed above.

Remove the columns we don't need.

```
email_sub<-email %>%
  select(spam,to_multiple,cc,attach,dollar,winner,inherit,password,format,re_subj,exclaim_subj)
```

```
email_mod2 <- glm(spam~.,data=email_sub,family="binomial")
```

```
summary(email_mod2)
```

```
##
## Call:
## glm(formula = spam ~ ., family = "binomial", data = email_sub)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6348  -0.4325  -0.2566  -0.0945   3.8846
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -0.79976    0.08935  -8.950  < 2e-16 ***
## to_multiple  -2.84097    0.31158  -9.118  < 2e-16 ***
## cc            0.03134    0.01895   1.654 0.098058 .
## attach        0.20351    0.05851   3.478 0.000505 ***
## dollar       -0.07304    0.02306  -3.168 0.001535 **
## winneryes     1.83103    0.33641   5.443 5.24e-08 ***
## inherit       0.32999    0.15223   2.168 0.030184 *
## password     -0.75953    0.29597  -2.566 0.010280 *
## format       -1.52284    0.12270 -12.411  < 2e-16 ***
## re_subj      -3.11857    0.36522  -8.539  < 2e-16 ***
## exclaim_subj  0.24399    0.22502   1.084 0.278221
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2437.2  on 3920  degrees of freedom
## Residual deviance: 1936.2  on 3910  degrees of freedom
## AIC: 1958.2
##
## Number of Fisher Scoring iterations: 7
```

Like multiple regression, the result are presented in a summary table. The structure of this table is almost identical to that of multiple regression; the only notable difference is that the p-values are calculated using the normal distribution rather than the t distribution.

In Math 378 we will learn more about tuning models but in this lesson we will do a simple procedure of just trimming variables from the model using the p-value. Using a method called backwards elimination with a p-value cutoff of 0.05 (start with the full model and trim the predictors with p-values greater than 0.05), we

ultimately eliminate the `exclaim_subj` and `cc` predictors. The remainder of this section will rely on this smaller model. The code for this process is below, note there are more efficient ways to do this.

Take out `exclaim_subj` because it has largest p-value.

```
email_mod2 <- glm(spam~to_multiple+cc+attach+dollar+winner+inherit+password+format+re_subj,
                  data=email_sub,family="binomial")
summary(email_mod2)
```

```
##
## Call:
## glm(formula = spam ~ to_multiple + cc + attach + dollar + winner +
##     inherit + password + format + re_subj, family = "binomial",
##     data = email_sub)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6496  -0.4362  -0.2549  -0.0936   3.8713
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.79012    0.08883  -8.894  < 2e-16 ***
## to_multiple -2.82306    0.31120  -9.071  < 2e-16 ***
## cc           0.03107    0.01891   1.643 0.100381
## attach       0.20265    0.05841   3.470 0.000521 ***
## dollar      -0.06891    0.02234  -3.085 0.002039 **
## winneryes    1.85422    0.33543   5.528 3.24e-08 ***
## inherit      0.33128    0.15049   2.201 0.027708 *
## password    -0.75634    0.29635  -2.552 0.010706 *
## format      -1.51455    0.12232 -12.381  < 2e-16 ***
## re_subj     -3.12436    0.36504  -8.559  < 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: 2437.2  on 3920  degrees of freedom
## Residual deviance: 1937.3  on 3911  degrees of freedom
## AIC: 1957.3
##
## Number of Fisher Scoring iterations: 7
```

Take out `cc` because it has largest p-value.

```
email_mod2 <- glm(spam~to_multiple+attach+dollar+winner+inherit+password+format+re_subj,
                  data=email_sub,family="binomial")
summary(email_mod2)
```

```
##
## Call:
## glm(formula = spam ~ to_multiple + attach + dollar + winner +
##     inherit + password + format + re_subj, family = "binomial",
##     data = email_sub)
```

```
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6591  -0.4373  -0.2544  -0.0944   3.8707
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.78138    0.08860  -8.820  < 2e-16 ***
## to_multiple -2.77682    0.30752  -9.030  < 2e-16 ***
## attach       0.20419    0.05789   3.527  0.00042 ***
## dollar      -0.06970    0.02239  -3.113  0.00185 **
## winneryes    1.86675    0.33652   5.547  2.9e-08 ***
## inherit      0.33614    0.15073   2.230  0.02575 *
## password    -0.76035    0.29680  -2.562  0.01041 *
## format      -1.51770    0.12226 -12.414  < 2e-16 ***
## re_subj     -3.11329    0.36519  -8.525  < 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: 2437.2  on 3920  degrees of freedom
## Residual deviance: 1939.6  on 3912  degrees of freedom
## AIC: 1957.6
##
## Number of Fisher Scoring iterations: 7
```

> **Exercise**:
> Examine the summary of the reduced model above, and in particular, examine the `to_multiple` row. Is the point estimate the same as we found before, -1.81, or is it different? Explain why this might be.[2]

Point estimates will generally change a little – and sometimes a lot – depending on which other variables are included in the model. This is usually due to colinearity in the predictor variables. We previously saw this in the Ebay auction example when we compared the coefficient of `cond` in a single-variable model and the corresponding coefficient in the multiple regression model that used three additional variables.

> *Example*: Spam filters are built to be automated, meaning a piece of software is written to collect information about emails as they arrive, and this information is put in the form of variables. These variables are then put into an algorithm that uses a statistical model, like the one we've fit, to classify the email. Suppose we write software for a spam filter using the reduced model in the `email_mod2` object. If an incoming email has the word "winner" in it, will this raise or lower the model's calculated probability that the incoming email is spam?

The estimated coefficient of `winner` is positive (1.86675). A positive coefficient estimate in logistic regression, just like in multiple regression, corresponds to a positive association between the predictor and response variables when accounting for the other variables in the model. Since the response variable takes value 1 if an email is spam and 0 otherwise, the positive coefficient indicates that the presence of "winner" in an email raises the model probability that the message is spam.

[2]The new estimate is different: -2.78. This new value represents the estimated coefficient when we are also accounting for other variables in the logistic regression model.

Example: Suppose the same email from the last example was in HTML format, meaning the `format` variable took value 1. Does this characteristic increase or decrease the probability that the email is spam according to the model?

Since HTML corresponds to a value of 1 in the `format` variable and the coefficient of this variable is negative (-1.51770), this would lower the probability estimate returned from the model.

Practical decisions in the email application

The last two examples highlight a key feature of logistic and multiple regression. In the spam filter example, some email characteristics will push an email's classification in the direction of spam while other characteristics will push it in the opposite direction.

If we were to implement a spam filter using the model we have fit, then each future email we analyze would fall into one of three categories based on the email's characteristics:

1. The email characteristics generally indicate the email is not spam, and so the resulting probability that the email is spam is quite low, say, under 0.05.
2. The characteristics generally indicate the email is spam, and so the resulting probability that the email is spam is quite large, say, over 0.95.
3. The characteristics roughly balance each other out in terms of evidence for and against the message being classified as spam. Its probability falls in the remaining range, meaning the email cannot be adequately classified as spam or not spam.

If we were managing an email service, we would have to think about what should be done in each of these three instances. In an email application, there are usually just two possibilities: filter the email out from the regular inbox and put it in a "spambox", or let the email go to the regular inbox.

Exercise:
The first and second scenarios are intuitive. If the evidence strongly suggests a message is not spam, send it to the inbox. If the evidence strongly suggests the message is spam, send it to the spambox. How should we handle emails in the third category?[3]

Exercise:
Suppose we apply the logistic model we have built as a spam filter and that 100 messages are placed in the spambox over 3 months. If we used the guidelines above for putting messages into the spambox, about how many legitimate (non-spam) messages would you expect to find among the 100 messages?[4]

Almost any classifier will have some error. In the spam filter guidelines above, we have decided that it is okay to allow up to 5% of the messages in the spambox to be real messages. If we wanted to make it a little harder to classify messages as spam, we could use a cutoff of 0.99. This would have two effects. Because it raises the standard for what can be classified as spam, it reduces the number of good emails that are classified as spam. However, it will also fail to correctly classify an increased fraction of spam messages. No matter the complexity and the confidence we might have in our model, these practical considerations are absolutely crucial to making a helpful spam filter. Without them, we could actually do more harm than good by using our statistical model. This tradeoff is similar to the one we found between Type 1 and Type 2 errors.

[3]In this particular application, we should err on the side of sending more mail to the inbox rather than mistakenly putting good messages in the spambox. So, in summary: emails in the first and last categories go to the regular inbox, and those in the second scenario go to the spambox.

[4]First, note that we proposed a cutoff for the predicted probability of 0.95 for spam. In a worst case scenario, all the messages in the spambox had the minimum probability equal to about 0.95. Thus, we should expect to find about 5 or fewer legitimate messages among the 100 messages placed in the spambox.

Diagnostics for the email classifier

There are two key conditions for fitting a logistic regression model:

1. Each predictor x_i is linearly related to logit(p_i) if all other predictors are held constant.
2. Each outcome Y_i is independent of the other outcomes.

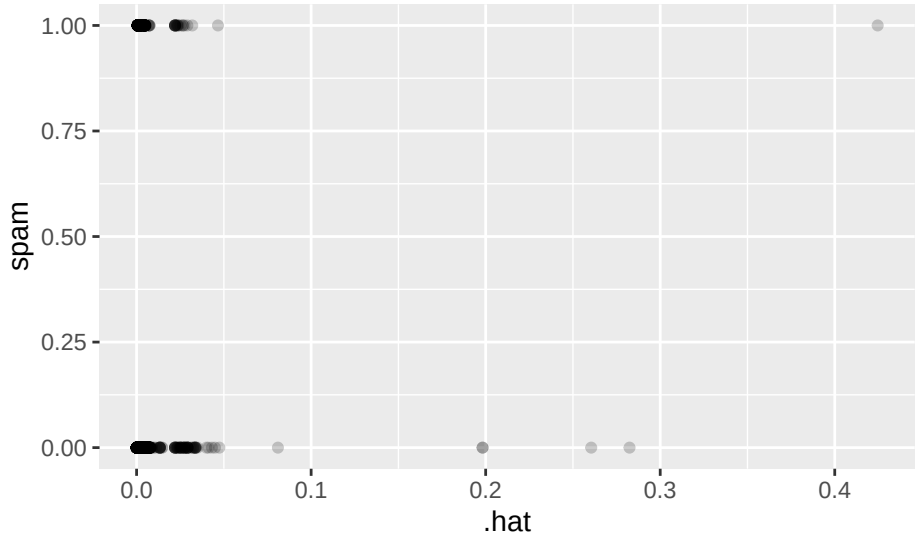
}

The first condition of the logistic regression model is not easily checked without a fairly sizable amount of data. Luckily, we have 3,921 emails in our data set! Let's first visualize these data by plotting the true classification of the emails against the model's fitted probabilities.

First get the predicted values.

```
library(broom)
```

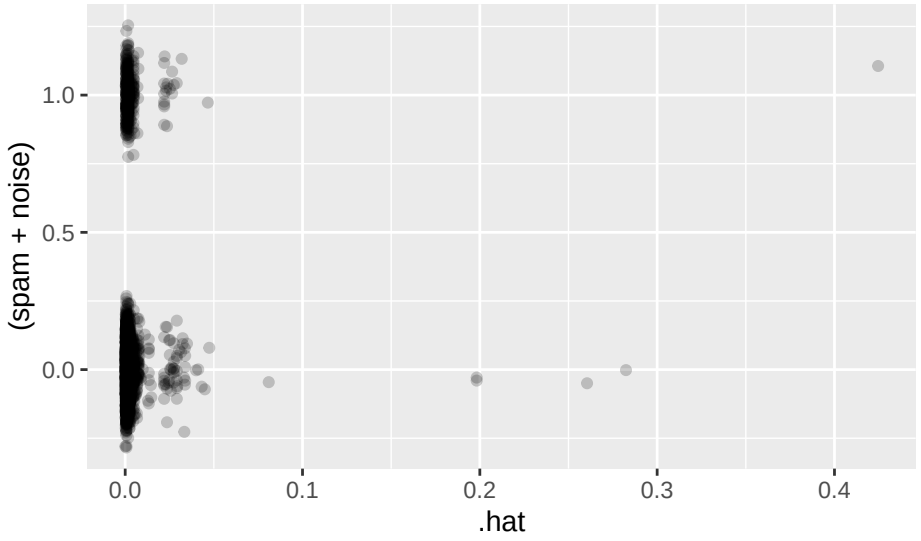
```
augment(email_mod2) %>%
  gf_point(spam~.hat,alpha=.2)
```



There is too much overlap so let's add some noise to separate the data points.

```
noise <- rnorm(3921, sd=0.08)
```

```
augment(email_mod2) %>%
  gf_point((spam+noise)~.hat,alpha=.2)
```



All the emails (spam or not) still have fitted probabilities below 0.5.

This may at first seem very discouraging: we have fit a logistic model to create a spam filter, but no emails have a fitted probability of being spam above 0.5. Don't despair; we will discuss ways to improve the model through the use of better variables.

We'd like to assess the quality of our model. For example, we might ask: if we look at emails that we modeled as having a 10% chance of being spam, do we find about 10% of them actually are spam? In Math 378 you will learn about ROC curves and smoothing splines that may help to assess the quality of the fit. That is beyond the scope.

We could evaluate the second logistic regression model assumption – independence of the outcomes – using the model residuals. The residuals for a logistic regression model are calculated the same way as with multiple regression: the observed outcome minus the expected outcome. For logistic regression, the expected value of the outcome is the fitted probability for the observation, and the residual may be written as

$$e_i = Y_i - \hat{p}_i$$

We could plot these residuals against a variety of variables or in their order of collection, as we did with the residuals in multiple regression. However, since the model will need to be revised to effectively classify spam and you have already seen similar residual plots in previous lessons on regression, we won't investigate the residuals here.

Improving the set of variables for a spam filter

If we were building a spam filter for an email service that managed many accounts (e.g. Gmail or Hotmail), we would spend much more time thinking about additional variables that could be useful in classifying emails as spam or not. We also would use transformations or other techniques that would help us include strongly skewed numerical variables as predictors.

Take a few minutes to think about additional variables that might be useful in identifying spam. Below is a list of variables we think might be useful:

(1) An indicator variable could be used to represent whether there was prior two-way correspondence with a message's sender. For instance, if you sent a message to john@example.com and then John sent you an email, this variable would take value 1 for the email that John sent. If you had never sent John an email, then the variable would be set to 0.

(2) A second indicator variable could utilize an account's past spam flagging information. The variable could take value 1 if the sender of the message has previously sent messages flagged as spam.

(3) A third indicator variable could flag emails that contain links included in previous spam messages. If such a link is found, then set the variable to 1 for the email. Otherwise, set it to 0.

The variables described above take one of two approaches. Variable (1) is specially designed to capitalize on the fact that spam is rarely sent between individuals that have two-way communication. Variables~(2) and~(3) are specially designed to flag common spammers or spam messages. While we would have to verify using the data that each of the variables is effective, these seem like promising ideas.

The table below shows a contingency table for spam and also for the new variable described in (1) above. If we look at the 1,090 emails where there was correspondence with the sender in the preceding 30 days, not one of these message was spam. This suggests variable (1) would be very effective at accurately classifying some messages as not spam. With this single variable, we would be able to send about 28% of messages through to the inbox with confidence that almost none are spam.

	prior correspondence		
	no	yes	Total
spam	367	0	367
not spam	2464	1090	3554
Total	2831	1090	3921

The variables described in (2) and (3) would provide an excellent foundation for distinguishing messages coming from known spammers or messages that take a known form of spam. To utilize these variables, we would need to build databases: one holding email addresses of known spammers, and one holding URLs found in known spam messages. Our access to such information is limited, so we cannot implement these two variables in this textbook. However, if we were hired by an email service to build a spam filter, these would be important next steps.

In addition to finding more and better predictors, we would need to create a customized logistic regression model for each email account. This may sound like an intimidating task, but its complexity is not as daunting as it may at first seem. We'll save the details for a statistics course where computer programming plays a more central role.

For what is the extremely challenging task of classifying spam messages, we have made a lot of progress. We have seen that simple email variables, such as the format, inclusion of certain words, and other circumstantial characteristics, provide helpful information for spam classification. Many challenges remain, from better understanding logistic regression to carrying out the necessary computer programming, but completing such a task is very nearly within your reach.

File Creation Information

- File creation date: 2020-07-31
- Windows version: Windows 10 x64 (build 18362)
- R version 3.6.3 (2020-02-29)
- `mosaic` package version: 1.7.0
- `tidyverse` package version: 1.3.0