Baseline Tutorial

Rina Friedberg, Evan Rosenman, Mike Baiocchi

Here, we detail the data cleaning and baseline evaluation process in R. A basic working knowledge of R will be helpful, albeit not strictly necessary, for following along. We have provided a sample dataset, randomly generated so that the response to each question occurs with probability equal to the frequency in the original dataset. Hence it contains no private information and any signal we may find therein is due to random chance. There are short exercises sprinkled throughout to help beginners explore the dataset.

Section 1: Preliminaries

We begin by loading and cleaning the data.

```r
# Load the data
data <- read.csv("sample-dataset")
```

First we cast the data to a dataframe, which is a useful data structure in R. We can easily find how many individuals and how many schools are recorded in our dataset.

```r
df<- data.frame(data)
n <- nrow(df) # total number of girls
cat("There are", n, "total girls in our study.")
```

```
There are 400 total girls in our study.
```

```r
cat("There are", length(unique(df$schoolid)), "total schools.")
```

```
There are 4 total schools.
```

Now we identify treatment and control schools. For this example, schools 1 and 2 are control schools, and schools 3 and 4 are treatment schools. The “ifelse” command works as follows: ifelse(statement, result if True, result if False). Note df$treatment adds a column called “treatment” since none exists yet; if it did, df$treatment would replace it by this new column. We will let treatment = 0 denote control, and treatment = 1 denote intervention.

```r
control<- 1:2
trt<- 3:4
df$assignment <- sapply(df$schoolid, function(id){ifelse(id%in%trt,1,0)})

# Isolate treatment and control rows.
treat.rows<- which(df$assignment == 1)
control.rows<- which(df$assignment == 0)

# Find number of individuals in treatment group and control group.
num.trt<- length(treat.rows)
num.control<- length(control.rows)
cat("We assigned", num.trt, "girls to treatment and", num.control, "girls to control.")
```

```
## We assigned 209 girls to treatment and 191 girls to control.
```

- Exercise: how many girls are in school 3?

Section 2: Covariate Balance

Now that we have identified treatment and control schools, we can move on to analyzing the balance of important covariates across them. We check features like relationship history and alcohol use, which have both been shown to be predictive of gender-based violence in previous work. We also check that rates of past gender-based violence experience are close between the two groups.

One major part of the data analysis for this project was handling missing data. Many girls did not respond to questions (coded as 99, depending on the question), or answered "I don't know" (coded as 88). Sometimes the data was lost in transmission, resulting in NA values. A handy trick in R is using the %in% command instead of testing for equality with ==; the latter will not work if your data has NA values, but the former will. We have to address all of these carefully as we analyze the data.

Relationship History

We begin with relationship history: we check how many girls in each group has had a boyfriend. The corresponding column is df$boyfriend. To compare, we will compute the percentage of girls who have had a boyfriend, and then check a Wald interval. A quick disclaimer: one can often do much better than Wald intervals! But they are a good quick check.

Begin with the treatment group. First we find our sample size, num.bf.trt. Here we are counting all rows where the girl either answered yes (1) or no (0). Some girls may not have known (88), or not responded ("NA"). We do not count them as either category. R doesn't like "NA", and will complain if you test using "=" when there are any missing values. The "%in%" command is a handy way to avoid problems with "NA".

- Exercise: how many girls have had a boyfriend?

We create a dataframe with only the rows that (a) correspond to treatment girls, and (b) have a valid answer (0 or 1).

```r
treat.rows.df <- df[treat.rows,] # isolate treatment girls
treat.rows.df <- treat.rows.df[treat.rows.df$boyfriend %in% c(0,1),] # remove NAs from relevant row
```

Now we find the proportion of girls in the treatment group who have had a boyfriend.

```r
# Find proportion of interest
num.bf.trt<- nrow(treat.rows.df) # number of treatment girls with valid answers
prop.bf.trt <- sum(treat.rows.df$boyfriend==1)/num.bf.trt # Proportion who have had a boyfriend
```

Last we compute a basic Wald interval.

```r
lower.bf.trt <- prop.bf.trt - 1.96*sqrt(prop.bf.trt*(1-prop.bf.trt)/num.bf.trt) # Lower bound for 95% i
upper.bf.trt<- prop.bf.trt + 1.96*sqrt(prop.bf.trt*(1-prop.bf.trt)/num.bf.trt) # Upper bound for 95% in
```

We repeat this process for the control group, and compare our results.

```r
control.rows.df <- df[control.rows,] # Isolate control girls
control.rows.df <- control.rows.df[control.rows.df$boyfriend %in% c(0,1),] # Remove NAs
num.bf.control<- nrow(control.rows.df) # Find number of girls with valid answer
prop.bf.control <- sum(control.rows.df$boyfriend==1)/num.bf.control # Proportion of interest
lower.bf.control <- prop.bf.control - 1.96*sqrt(prop.bf.control*(1-prop.bf.control)/num.bf.control) # L
upper.bf.control<- prop.bf.control + 1.96*sqrt(prop.bf.control*(1-prop.bf.control)/num.bf.control) # Up

# Report and compare results
cat("Girls in treatment schools had relationship prevalence", prop.bf.trt, "(", lower.bf.trt, ",", uppe
```

```
Girls in treatment schools had relationship prevalence 0.2049 ( 0.1496 , 0.2601 )
```

```
cat("Girls in control schools had relationship prevalence", prop.bf.control, "(", lower.bf.control, ","
```

```
Girls in control schools had relationship prevalence 0.2 ( 0.1424 , 0.2576 )
```

Gender-Based Violence History

Now tha we understand how the covariate balance is checked, it's time to move on to looking at GBV. In our methods paper, we discuss the difficulty of aggregating survey information and justify our methodological choices. We start by isolating the relevant survey answers.

```
forced_cols <- c(35:43, 48:59)
forced_df <- df[,forced_cols] # Isolate Relevant data
colnames(forced_df)
```

```
 [1] "physicalforced"        "threatforced"
 [3] "othersexforced"        "forcedbynotbf"
 [5] "whoforced___0"         "whoforced___1"
 [7] "whoforced___2"         "whoforced___3"
 [9] "whoforced___88"        "forcedtoodrunk"
[11] "forcedbymanyboys"      "forcedbymanyboysdrunk"
[13] "forcedintomultiple"    "pregnancyforced"
[15] "forcedhivtest"         "timesforcedlife"
[17] "timesforcedlifenum"    "forcedbyid___1"
[19] "forcedbyid___2"        "forcedbyid___3"
[21] "forcedbyid___88"
```

Clean relevant data

If you examine the dataset, you'll see several surprising values. This is because of how the data was encoded. Here is a mini data dictionary:

NA = 100, 999, NA Don't know = 88 Refuse = 99

Knowing this, we can easily isolate these values by re-casting them to be negative. This serves several purposes. One, it's easy to see if we've made a mistkae, since the number of times something has happened cannot be negative. Two, we can preserve information by keeping refusal, did not know, and NA separate. Moreover, when we calculate means, it is easy to isolate only non-negative values.

```
forced_df[is.na(forced_df)] <- -999 # Missing values
forced_df[forced_df == 999] <- -999 # N/A
forced_df[forced_df == 100] <- -999 # N/A
forced_df[forced_df == 99] <- -99 # Refused
forced_df[forced_df == 88] <- -88 # Did not know
```

Estimate rate of GBV experiences

Now, we move on to a tricky topic: using the data we have to estimate how many times each girl has been raped. When the survey was written, there were about 9 questions with the goal of understanding specific gender-based violence experiences (examples: were you raped by somebody who was not your boyfriend? Did you have to take an HIV test as a result of rape?). At the end of those, we asked the girls how many times in their life they had been raped: they could answer 0, 1, or fill in a number if it was more than 1. They could also refuse to answer or mark "I don't know." The goal of this last question was to ask the total number of rapes, including all previously reported rapes. It's coded as two separate columns in the data: timesforcedlife is their answer to the first section (0 or 1, often NA), and timesforcedlifenum is the number they wrote in.

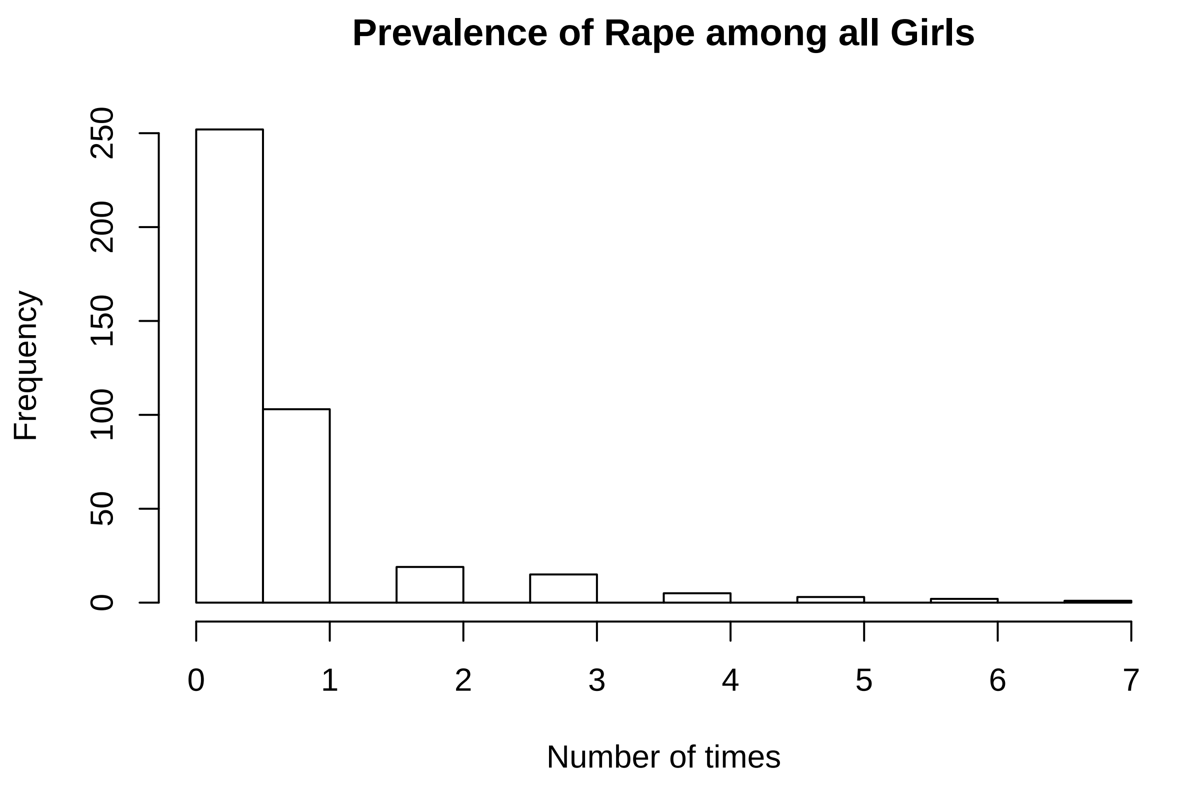
To clarify the problem, we approach the survey from the perspective of the girls taking it. Consider one girl who was raped once by her boyfriend. When asked "how many times have you been raped by a boyfriend?," she answers 1. When later asked, "how many times in your life have you been raped?", she answers "0," believing the question to be about rapes she has not yet reported on the survey. We want to count her in the group of girls who have been assaulted, even though the report is contradictory. If we aggregate her answers across all specific questions about gender-based violence, we will see 1 rape; in the times forced life questions, we will see either 0s or NAs.

Consider another girl, who was also raped once by her boyfriend, and interpreted the questions as we thought when we wrote them. She will answer 1 for " how many times have you been raped by a boyfriend?", and will also answer "1" for "how many times in your life have you been raped?". We want to count her as somebody who has been raped once; note that here, we must make sure to not double-count this rape. So we do not sum across all of her columns; we check the aggregated sum across all specific columns (1), and the answer to times forced life (also 1). We look at these separately to make sure we do not over-count, and then take the maximum (here, just 1).

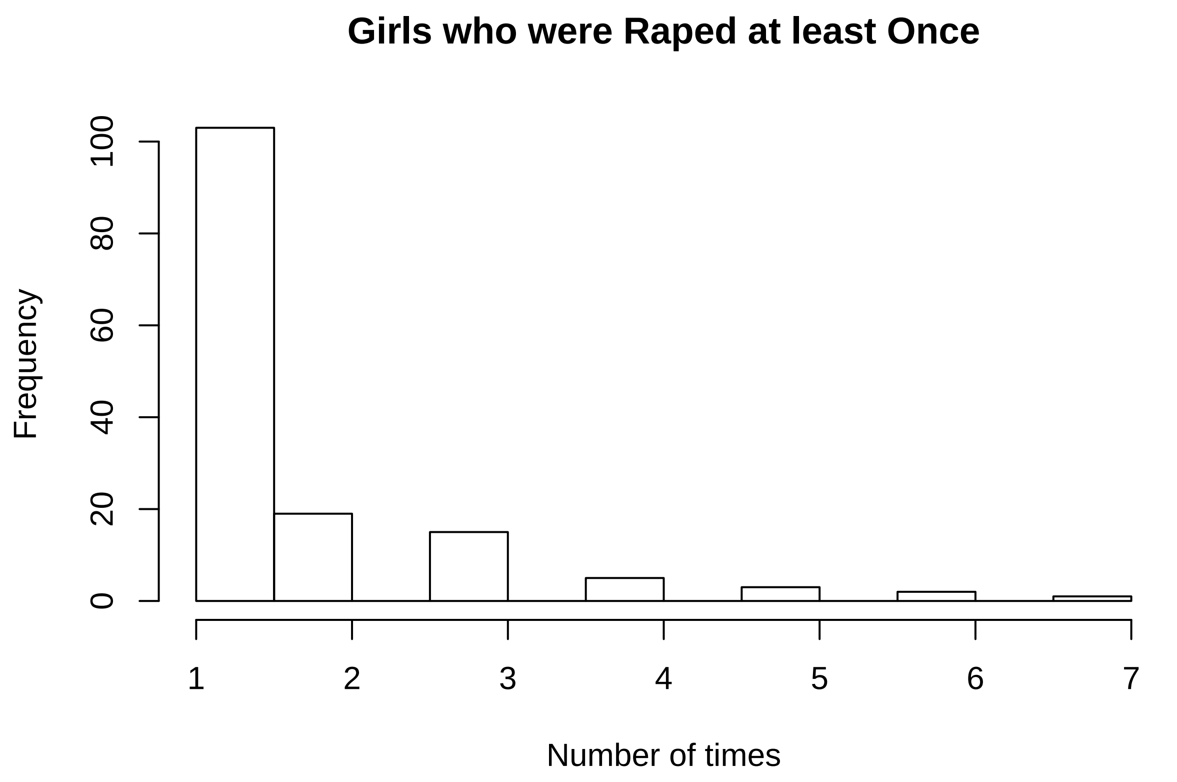
```
# Isolate columns with individual perpetrator counts
perpcounts = 62:65

# We create a vector called "forced", which gives the number of times each individual in the
# study reported a forced sexual experience

forced <- sapply(1:n, function(i){
  row <- forced_df[i,]
  # Recall we cast all "NA" and refused answers to be negative.
  # So first we check if there are any non-negative entries in the row.
  # If there are only negative entries, there is no useful information in the row.
  if(length(row>0) == 0){
    return(-99)
  }
  # First, we count the number of times a girl listed by each perpetrator.
   numreports = sum(df[i,perpcounts], na.rm = TRUE)

  # As discussed at length above and in our papers, we assume that each report in columns physicalforce
  # This is backed up by how many girls reported rapes in one column, and then 0 for timesforcedlife, d
  # So we assume the number of times a girl has been forced is max(timesforcedlifenum, sum over previou

  # This line sums up all columns other than the two listed, and then compares to the others.
  numAggregated<- sum(row[!(colnames(row) %in% c("timesforcednum", "timesforcedlife")) & row>0])
  return(max(numAggregated, numreports, row$timesforcedlife, row$timesforcedlifenum))
})
df$forced <- forced
```

Now we summarize the data. We include both Wald confidence intervals, and useful histograms. First, we recommend thinking about why we might want to include these confidence intervals. Why is that kind of analysis helpful, given our goal of checking balance between treatment and control schools? What do you think might happen?

```
# Histograms
hist(forced[forced>=0], main = "Prevalence of Rape among all Girls", xlab = "Number of times")
```



```r
hist(forced[forced>0], main = "Girls who were Raped at least Once", xlab = "Number of times")
```



```r
# Report Numbers
prop_forced <- sum(forced>0)/n
cat("We observed a frequency of", prop_forced, "girls who were sexually assaulted at least once.")
```

```
We observed a frequency of 0.37 girls who were sexually assaulted at least once.
```

```r
lower_forced <- prop_forced - 1.96*sqrt(prop_forced*(1-prop_forced)/n)
upper_forced<- prop_forced+ 1.96*sqrt(prop_forced*(1-prop_forced)/n)
cat("The Wald Interval for this proportion is (",lower_forced, ",", upper_forced, ")")
```

```
The Wald Interval for this proportion is ( 0.3227 , 0.4173 )
```

```r
prop_once <- sum(forced==1)/sum(forced>0)
cat("Among all girls who were raped,", prop_once, "were raped once.")
```

```
Among all girls who were raped, 0.6959 were raped once.
```

- Exercise: can you find the Wald intervals for prevalence of rape, for just treatment schools? Just control schools?

Bootstrap Confidence Intervals

Instead of Wald Intervals, we can use the bootstrap to more robustly estimate the distribution of our data. To generate bootstrapped datasets, we take a bootstrap sample within each cluster, here each unique school. Then, we check how many girls in that sample reported any non-NA response (the second element returned), and how many were sexually assaulted at least once (the first element returned). Repeating this process 1000 times gives us an estimate of the spread of our data.

```r
B = 1000
results = sapply(1:B, function(b){
  ids = unique(df$schoolid)
  numbers = sapply(ids, FUN=function(id){
    # isolate school
    cluster = df[df$schoolid==id,]

    # sample with replacement from the school
    n.id = nrow(cluster)
    f = cluster$forced[sample(n.id,replace=TRUE)]

    # return number of girls who were forced, and total girls who reported non-NA answers
    return(c(sum(f[!is.na(f) & f > 0]), length(f[!is.na(f)])))
  })
  sum(numbers[1,])/sum(numbers[2,])
})

lower_boot = quantile(results,prob=0.025)
upper_boot = quantile(results,prob=0.975)
cat("We observed a proportion of", prop_forced, "girls who were raped.")
```

```
We observed a proportion of 0.37 girls who were raped.
```

```r
cat("The 95% bootstrap interval for this proportion is (", lower_boot, ",", upper_boot, ")")
```

```
The 95% bootstrap interval for this proportion is ( 0.4999 , 0.7075 )
```

Testing Treatment/Control Balance of Prior GBV

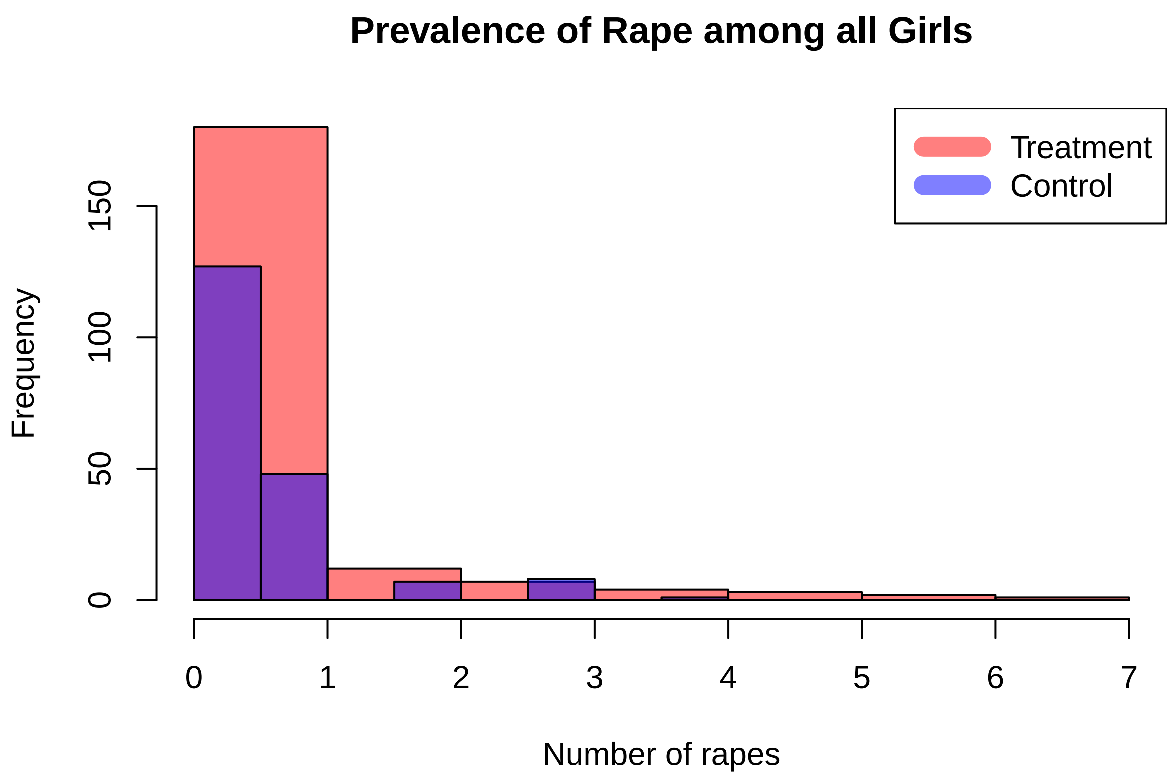
```r
# Treatment: Isolate rows
forced_treat <- forced[treat.rows]
prop_treat_forced <- sum(forced_treat>0)/num.trt
```

```
# Treatment: Calculate Wald interval
lower_treat_forced <- prop_treat_forced - 1.96*sqrt(prop_treat_forced*(1-prop_treat_forced)/num.trt)
upper_treat_forced<- prop_treat_forced+ 1.96*sqrt(prop_treat_forced*(1-prop_treat_forced)/num.trt)

# Control: Isolate rows
forced_control <- forced[control.rows]
prop_control_forced <- sum(forced_control>0)/num.control
# Control: Calculate Wald interval
lower_control_forced <- prop_control_forced - 1.96*sqrt(prop_control_forced*(1-prop_control_forced)/num
upper_control_forced<- prop_control_forced+ 1.96*sqrt(prop_control_forced*(1-prop_control_forced)/num.c

cat("In the treatment group, we see a proportion ", prop_treat_forced, "(",lower_treat_forced,",",upper_
```

```
In the treatment group, we see a proportion  0.4019 ( 0.3354 , 0.4684 ) of girls who have experienced GI
```

Last, we can look at a layered histogram to visually check that the distributions of prior GBV are comparable in treatment and control groups.

```
# Prevalence of rape among all girls (stratified)
forced_trt <- forced[treat.rows] # Isolate treatment girls
forced_control<- forced[control.rows] # Isolate control girls

# This creates a nice histogram with some transparent colors.
hist(forced_trt[forced_trt>=0], main = "Prevalence of Rape among all Girls", xlab = "Number of rapes", c
hist(forced_control[forced_control>=0], add=T, col=rgb(0,0,1,0.5))
legend("topright", c("Treatment", "Control"), col=c(rgb(1,0,0,0.5), rgb(0,0,1,0.5)),lwd=10)
```



Prediction

In this section, we will try to predict whether a girl has experienced GBV using other covariates. We use a basic logistic regression model: if you aren't familiar with the glm package, just note that specifying family=binomial induces logistic regression.

One crucial thing to note here is that we generated all of these covariates randomly. So you should not expect to replicate the results from our study, and should be suspicious of any significant relationships. Nevertheless, you can experiment with the regression models to get a sense of our methods!

```r
# Let's add a column indicating whether or not a girl has experienced GBV to our data frame.
forced.indicator <- ifelse(forced>0,1,0)
df$forced.indicator <- forced.indicator
```

One useful trick in R is creating formula objects that can be fed to a regression model.

```r
# Store all of the potential predictors in a vector of strings
candidates <- c("takenselfdefense", "takennmn","timesalcohol","motherdead","fatherdead","dadhitmom")

# This creates the formula: forced.indicator ~ takenselfdefense + takennmn + ...
sum.candidates <- paste(candidates, collapse = " + ")
sum.formula <- as.formula(paste("forced.indicator",sum.candidates,sep="~"))
```

Now, we will fit a logistic regression to predict probability of being raped. If you are interested in logistic regression and generalized linear models, we recommend Chapter 4 of Agresti's Categorical Data Analysis. For a simple understanding: GLM fits a family of models called generalized linear models (GLMs). Logistic regression, which is commonly used for modeling probabilities, is a popular member of that family. One major advantage it has is that it allows us to model probabilities, because it returns estimates between 0 and 1. Linear regression can't be easily constrained to do that! For a logistic regression, we give the model lots of covariates ("candidates") and a vector of 0s and 1s, indicating if a girl was raped or not. We get a model that uses those covariates to predict the probability of each girl being raped or not.

```r
logistic.fit <- glm(sum.formula, data=df, family="binomial")
```

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

```r
summary(logistic.fit)
```

```

Call:
glm(formula = sum.formula, family = "binomial", data = df)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-1.124  -0.966  -0.947   1.405   1.634

Coefficients:
                 Estimate Std. Error z value Pr(>|z|)
(Intercept)      -0.51997    0.12182   -4.27    2e-05 ***
takenselfdefense -0.00194    0.00817   -0.24     0.81
takennmn         -0.04837    0.05493   -0.88     0.38
timesalcohol      0.00361    0.00822    0.44     0.66
motherdead        0.38970    0.61890    0.63     0.53
fatherdead        0.00297    0.00630    0.47     0.64
dadhitmom        -0.17093    0.18522   -0.92     0.36
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

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

    Null deviance: 506.91  on 384  degrees of freedom
Residual deviance: 493.12  on 378  degrees of freedom
  (15 observations deleted due to missingness)
AIC: 507.1

Number of Fisher Scoring iterations: 9
```

- Exercise: what other covariates might you include? Does the result change if you scale them? How so?

Adjudication

In this section, we will address the "adjudication" of conflicting survey responses.

Our policy is to count a girl as having been raped if any of her responses indicate she has been raped (even if other results conflict with this response). We believe this is justifiable, because sexual assault is an emotionally difficult topic, and it is thus wholly possible for victims to provide answers that seem to be in contradiction – perhaps indicating they have never been raped in one response, but later describing an incident of rape. Moreover, we believe it is very unlikely that a non-victim would provide a false, and inconsistent report.

In the following code, we pull those records that have conflicting survey responses.

```r
# get the column names for the individual reports of rape
reports <- which(names(df) %in% c("physicalforced", "threatforced", "othersexforced", "forcedbynotbf"))

# get the column names for the aggregate reports of rape
times <- which(names(df) %in% c("timesforcedlife", "timesforcedlifenum"))

# look at any times that there are conflicts between the reports
contradictionRows <- apply(df, 1, FUN = function(row) {
  times_NAorZero <- ifelse(sum(as.numeric(row[times]),
                               na.rm = TRUE) == 0, 1, 0) # had 0 or NA for all timesforcedlife
  report_true <- ifelse(sum(as.numeric(row[reports]),
                            na.rm = TRUE) > 0, 1, 0) # reported at least once
  report_true & times_NAorZero  # return yes if contradiction
})

# pull the conflicting and non-conflicting rows
nonContradictions <- df[!contradictionRows & forced > 0,]
contradictions <- df[contradictionRows,]
print(paste("There are", sum(contradictionRows), "records with contradictory responses and",
            nrow(nonContradictions), "records with internally consistent responses."))
```

```
## [1] "There are 15 records with contradictory responses and 134 records with internally consistent res
```

One way to statistically justify our adjudication procedure is to make use of the model we fit in the last section. The basic idea: if the girls with conflicting reports are more similar to girls with non-conflicting rape reports, and less similar to non-victims, that bolsters our case that they should be classified as victims.

Below, we refit the model to the subset of data excluding the contradictory rows, and compute the mean predicted rape probability for each of the three groups, using the model. This is one measure of "closeness" – if we see that the girls with contradicting reports have a mean probability much closer rape victims with non-contradicting reports than to non-victims, this provides strong justification for our approach.

```r
# re fit the model to the data, excluding the contradictory responses
logistic.fit.no.contradictions <- glm(forced.indicator ~ intercourse + bribeforsex + boyfriend + insult
  stayingfriends + weaponbybf + takenselfdefense + payattention +
  otherhitmom + taughtnmnskill + takennmn + motherdead + timesdrugs +
  studyinteresting + preventnervous, family = "binomial", data = df,  subset = !contradictionRows)

# compare the mean of this model on the the three groups
mean(predict(logistic.fit.no.contradictions, newdata = nonContradictions,
             type = "response"), na.rm = TRUE)
```

```
[1] 0.3735
```

```r
mean(predict(logistic.fit.no.contradictions, newdata = contradictions,
             type = "response"), na.rm = TRUE)
```

```
[1] 0.413
```

```r
mean(predict(logistic.fit.no.contradictions, newdata = df[forced == 0,],
             type = "response"), na.rm = TRUE)
```

```
[1] 0.3146
```

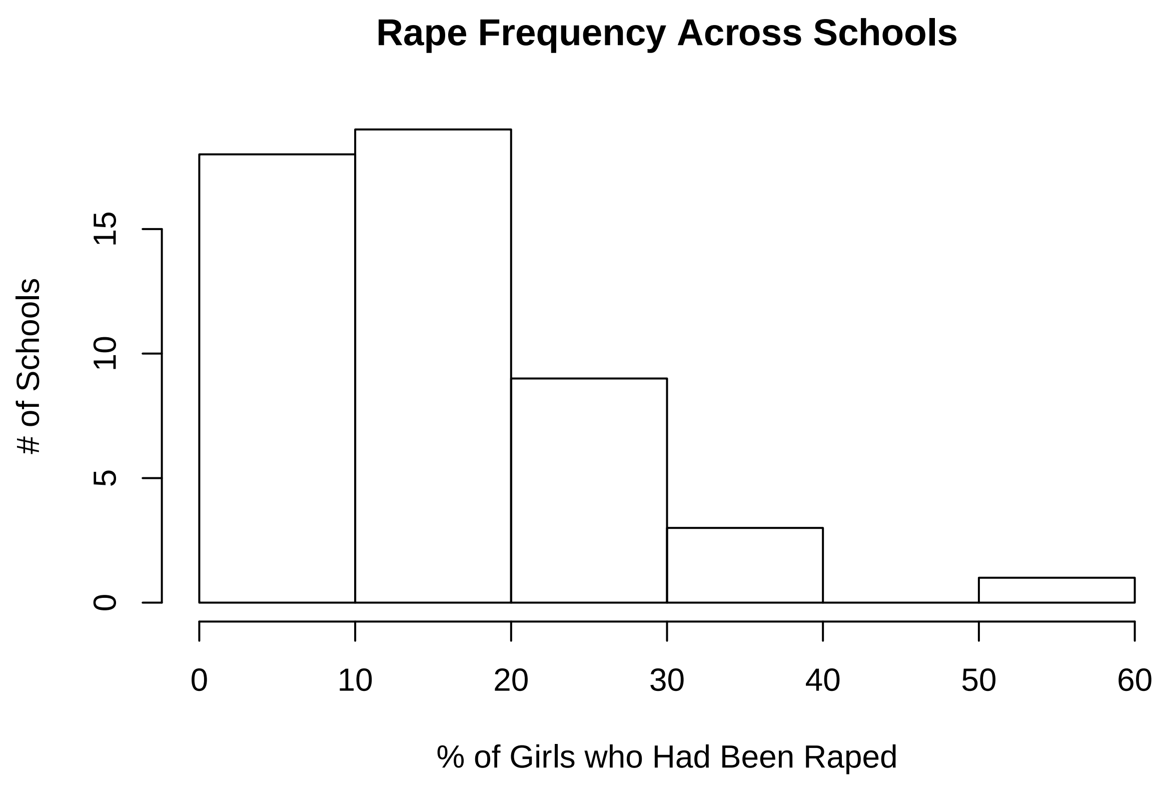
```r
# if we see separation in these numbers, it provides a good justification to
# simply include all rows with evidence of rape
```

We get a mean prediction of 41.3% probability for the contradicting rows, 37.4% for the non-contradicting rows, and 31.5% for the non-victims. Using real data, we would expect the "gap" to actually be somewhat larger.

School Prediction

In this section, we analyze the association between school-level predictors and the frequency of rape among girls at that school. In practice, you will aggregate from the individual-level data to estimate these frequencies. In this example, we load the data from a file that also contains our school-level covariates. We then explore the variation in rape frequency at different schools.

```r
# load the school data
school.df <- read.csv("sample-school-dataset")

# inspect the variation in rape at school level
hist(100*school.df$avg, xlab = "% of Girls who Had Been Raped",
     ylab = "# of Schools", main = "Rape Frequency Across Schools")
```



Next, we generate a few "derived features" from the school-level features. Not all of these turn out to be meaningful, but they provide an example of how the raw features can be exploited if researchers have a particular hypothesis about what might be associated with rape frequency at a school.

```
# create a few derived attributes that might be relevant
school.df$maleStudents <- apply(school.df, 1, FUN = function(x) { # total number of male students
  sum(as.numeric(x[grep("Boysclass", names(school.df))]))
})
school.df$femaleStudents <- apply(school.df, 1, FUN = function(x) { # total number of female students
  sum(as.numeric(x[grep("Girlsclass", names(school.df))]))
})
school.df$totalStudents <- apply(school.df, 1, FUN = function(x) { # total number of students
  sum(as.numeric(x[grep("class", names(school.df))]))
})
school.df$totalTeachers <- apply(school.df, 1, FUN = function(x) { # total number of teachers
  sum(as.numeric(x[grep("Teachers", names(school.df))]))
})
school.df$totalClassrooms <- apply(school.df, 1, FUN = function(x) { # total number of classrooms
  sum(as.numeric(x[grep("Classrooms", names(school.df))]))
})
school.df$studentsPerClassRoom <- school.df$totalStudents/school.df$totalClassrooms # students per clas
school.df$studentsPerTeacher <- school.df$totalStudents/school.df$totalTeachers # students per teacher
school.df$genderRatio <- school.df$maleStudents/school.df$femaleStudents # ratio of male to female stud
school.df$relativeDropout <- school.df$Boysclass5/school.df$Girlsclass5/ # relative dropout rate from s
  (school.df$Boysclass8/school.df$Girlsclass8)
school.df$boysToiletsPerBoy <- school.df$Boystoilets/school.df$maleStudents # toilets per boy
school.df$girlsToiletsPerGirl <- school.df$Girlstoilet/school.df$femaleStudents # toilets per girl
```

Now, we compute the individual associations between these covariates and our outcome measure, the frequency of rape at the school. There measures tell us if the variables are associated with rape frequency, though we make no claims that the relationships are causal, nor that they hold once we account for other variables in the dataset.

For continuous covariates (i.e. numeric variables), we use as our measure of association the p-value for a univariate linear regression of rape frequency on the covariate. For categorical covariates, we use the F-statistic from the regression of rape frequency on the covariate. In either case, we weight the schools by the number of girls at the school, so as not to overweight smaller schools.

The code below may seem a bit complicated, but it is simply looping over the covariates and collecting these statistics into a table.

```r
# collect list of all predictors
predictors <- c("Meanscore", "Boystoilets", "Girlstoilet",
                "totalStudents", "totalTeachers", "totalClassrooms", "studentsPerClassRoom",
                "studentsPerTeacher", "genderRatio", "relativeDropout",
                "boysToiletsPerBoy","girlsToiletsPerGirl", "Area", "Roof", "Floor")

# to undertaand whether these covariates are associated with the sexual assault
# rate directly, we compute univariate regressions (weighted by # of girls in the school)
v_weighted <- sapply(predictors, FUN = function(x) {
  mod <- lm(as.formula(paste("avgFrequency ~", x, sep = "")), data = school.df,
          weights = school.df$num.girls)

  if(is.numeric(school.df[[x]])) {
    c(x, summary(mod)$coef[2,4], sign(summary(mod)$coef[2, 1]),
      mean(school.df[[x]], na.rm = TRUE), min(school.df[[x]], na.rm = TRUE),
      max(school.df[[x]], na.rm = TRUE), sd(school.df[[x]], na.rm = TRUE))
  } else {
    f <- summary(mod)$fstatisti
    c(x, pf(f[1],f[2],f[3],lower.tail=F), "NA", "NA", "NA", "NA", "NA")
  }
})
predictorsTable <- data.frame(t(v_weighted))
names(predictorsTable) <- c("Covariate", "p-value", "Sign", "Covariate Mean",
                            "Covariate Min", "Covariate Max", "Covariate SD")
rownames(predictorsTable) <- NULL
predictorsTable$Covariate <- as.character(predictorsTable$Covariate)
predictorsTable$`p-value` <- as.numeric(as.character(predictorsTable$`p-value`))
predictorsTable$Sign <- as.numeric(as.character(predictorsTable$Sign))
predictorsTable$'Covariate Mean' <- as.numeric(as.character(predictorsTable$'Covariate Mean'))
predictorsTable$'Covariate Min' <- as.numeric(as.character(predictorsTable$'Covariate Min'))
predictorsTable$'Covariate Max' <- as.numeric(as.character(predictorsTable$'Covariate Max'))
predictorsTable$'Covariate SD' <- as.numeric(as.character(predictorsTable$'Covariate SD'))


# order and inspect the table
predictorsTable <- predictorsTable[order(predictorsTable$`p-value`),]
rownames(predictorsTable) <- c()
```

The resulting predictors table is given below. As in the real data analysis, the "relative dropout rate" (the ratio of how quickly boys drop out between 5th and 8th grade to how quickly girls drop out) is the most strongly associated predictor, while the floor type and geographical location (area) of the school are also strongly associated. For relative dropout rate, the relationship is negative. When girls drop out faster than boys, the relative dropout rate is lower, and rape frequency is higher. This makes some intuitive sense, as girls may drop out of school faster if they are victims of sexual violence, and areas with higher female dropout rates may have poorer attitudes about gender equality.

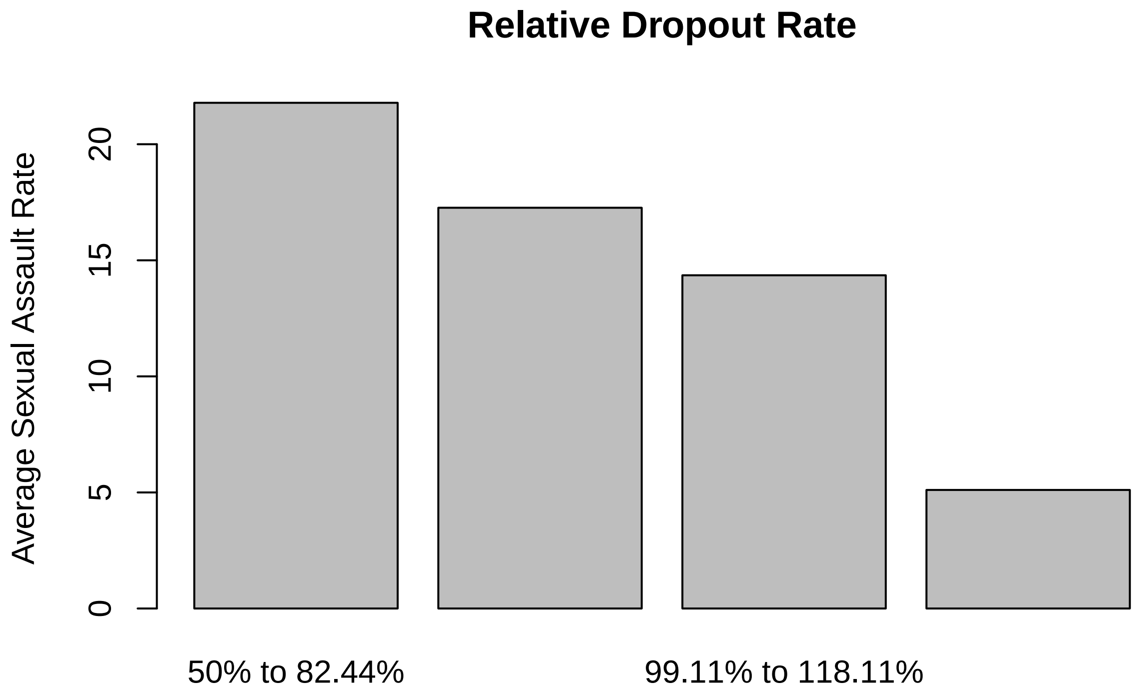
Among other associated covariates, the floor type is likely an indication of the socioeconomic status of the students attending the school, as is the area.

In the table below, “Sign” indicates the direction of the relationship between the covariate and rape frequency (+1 is positive, -1 is negative). Categorical variables do not have a sign, nor a minimum, maximum, or standard deviation.

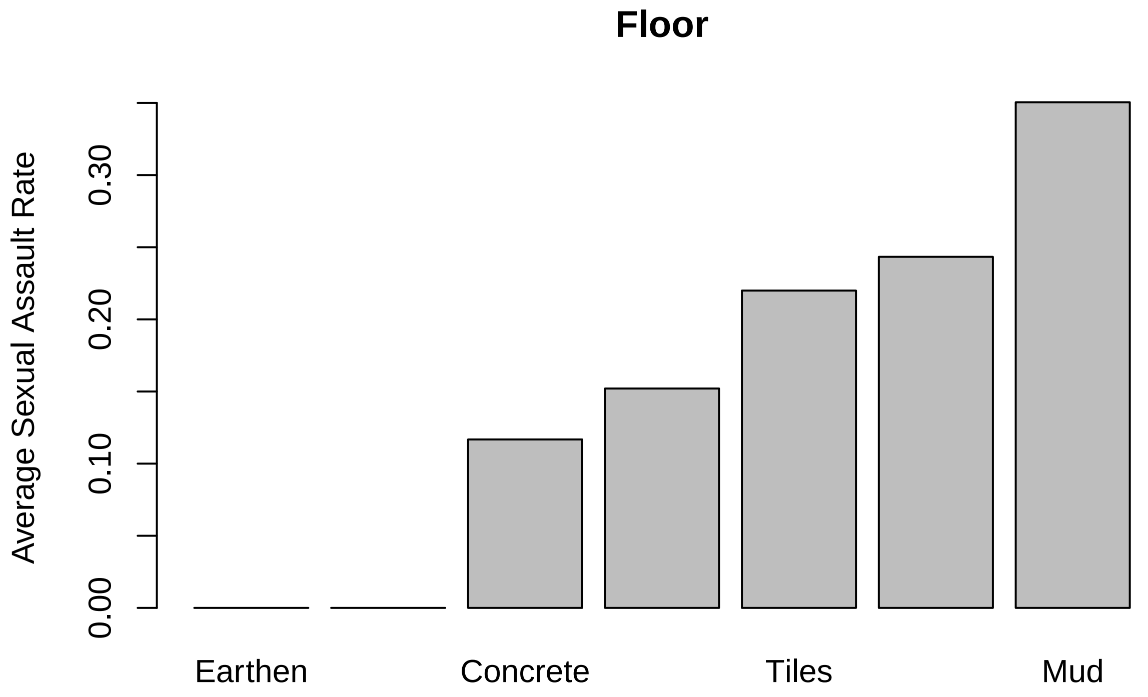
```
              Covariate   p-value Sign Covariate Mean Covariate Min
1       relativeDropout 5.107e-06   -1        1.04324     5.000e-01
2                 Floor 8.830e-03   NA             NA            NA
3                  Area 1.667e-02   NA             NA            NA
4         totalTeachers 1.451e-01   -1       15.36000     6.000e+00
5           genderRatio 1.912e-01    1        0.95193     7.018e-01
6             Meanscore 3.557e-01   -1      266.80000     1.970e+02
7   studentsPerClassRoom 3.913e-01  -1       51.86368     1.308e+01
8         totalStudents 5.862e-01   -1      569.44000     1.130e+02
9                  Roof 6.569e-01   NA             NA            NA
10    studentsPerTeacher 6.738e-01   1       41.21841     5.481e+00
11     boysToiletsPerBoy 6.752e-01   1        0.04312     6.250e-03
12          Girlstoilet 6.874e-01   -1       11.04000     1.000e+00
13   girlsToiletsPerGirl 9.264e-01  -1        0.04646     5.102e-03
14       totalClassrooms 9.460e-01   1       10.68000     5.000e+00
15          Boystoilets 9.824e-01   -1        9.44000     1.000e+00
   Covariate Max Covariate SD
1         1.8669      0.32118
2             NA           NA
3             NA           NA
4        38.0000      7.55594
5         1.6032      0.15112
6       379.0000     37.94733
7       111.3846     26.42700
8      1455.0000    373.82382
9             NA           NA
10      100.5000     23.79693
11        0.1757      0.03124
12       23.0000      6.15766
13        0.1587      0.03287
14       23.0000      3.71670
15       18.0000      5.02711
```

Next, we plot our top three associations, so as to see what the relationship looks like visually. Code to generate the plots is given below – note that we can use the simple barplot() command for categorical variables, but need to work a little harder to create the quartile plot used for the continuous “Relative Dropout Rate” covariate.

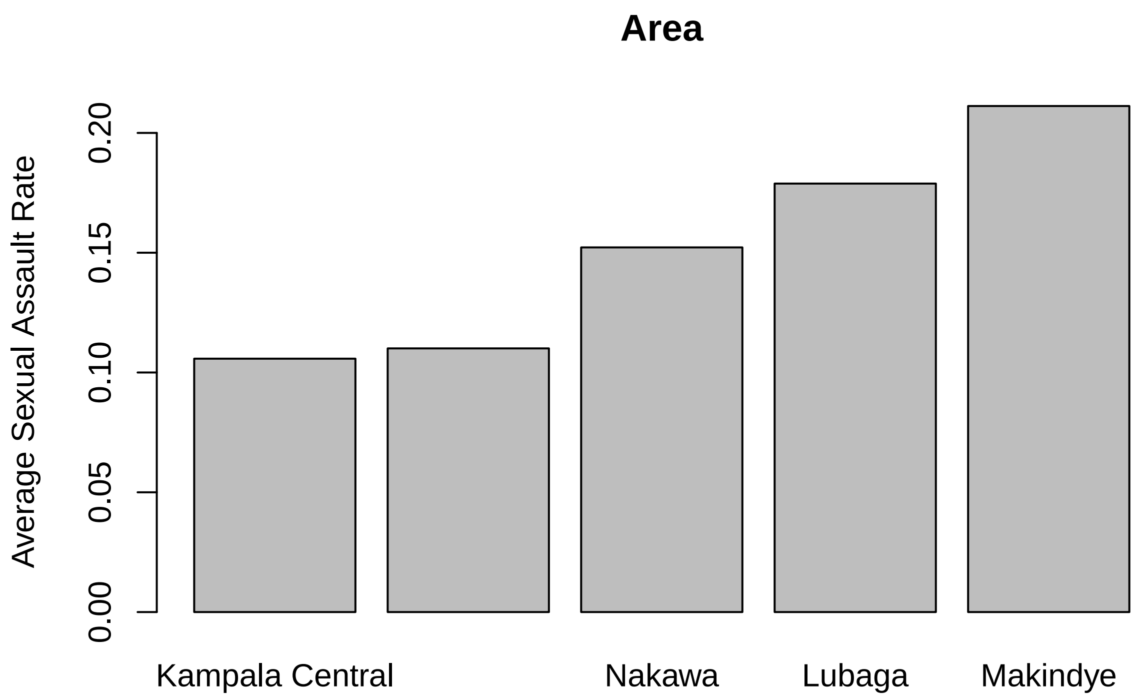
```r
# relative dropout rate
quant <- quantile(school.df$relativeDropout*100)
results <- sapply(2:length(quant), FUN = function(i) {
  # get relevant rows
  mean(school.df$avg[100*school.df$relativeDropout > quant[i-1] &
                     100*school.df$relativeDropout < quant[i]])
})
labels <- sapply(2:length(quant), FUN = function(i) {
  paste(round(quant[i-1], 2), "% to ", round(quant[i], 2), "%", sep = "") })

barplot(100*results, main = "Relative Dropout Rate",
        ylab = "Average Sexual Assault Rate",
        names.arg = labels)
```



```
# floor plot
barplot(sort(tapply(school.df$avgFrequency, INDEX = school.df$Floor, FUN = mean)),
        main = "Floor", ylab = "Average Sexual Assault Rate")
```



```
# area plot
barplot(sort(tapply(school.df$avgFrequency, INDEX = school.df$Area, FUN = mean)),
        main = "Area", ylab = "Average Sexual Assault Rate")
```



Lastly, we put our predictors together into a linear regression model to estimate the rape frequency at a school based on its covariates. This is useful to us, since rape frequencies will not generally be easy to compute directly (since leanring this information requires interviewing girls at the school). But many covariates are easier to obtain, so we may be able to get a decent estimate of a school's rape frequency without conducting interviews.

In the code below, we use an automated variable selection procedure in R to select variables for the model. We make use of forward stepwise selection, where all covariates are provided as candidates to the algorithm. The code and resulting model are provided below:

```
# build a model
emptyMod <- lm(avgFrequency ~ relativeDropout, data = school.df)
fullMod <- lm(as.formula(paste("avgFrequency ~", paste(predictors, collapse = "+"))), data = school.df)
steppedModel <- step(emptyMod, direction = "forward",
                     scope = list(lower = emptyMod, upper = fullMod))
```

```
Start:  AIC=-249.8
avgFrequency ~ relativeDropout

                       Df Sum of Sq   RSS  AIC
+ Area                  4    0.0991 0.213 -261
+ totalTeachers         1    0.0139 0.298 -250
<none>                              0.312 -250
+ totalStudents         1    0.0109 0.301 -250
+ Boystoilets           1    0.0096 0.303 -249
+ Girlstoilet           1    0.0094 0.303 -249
+ studentsPerClassRoom  1    0.0093 0.303 -249
+ Meanscore             1    0.0040 0.308 -248
+ girlsToiletsPerGirl   1    0.0039 0.308 -248
+ totalClassrooms       1    0.0037 0.309 -248
+ boysToiletsPerBoy     1    0.0026 0.310 -248
+ studentsPerTeacher    1    0.0012 0.311 -248
+ genderRatio           1    0.0005 0.312 -248
+ Floor                 6    0.0531 0.259 -247
+ Roof                  5    0.0151 0.297 -242

Step:  AIC=-260.9
avgFrequency ~ relativeDropout + Area

                       Df Sum of Sq   RSS  AIC
+ Boystoilets           1    0.0088 0.204 -261
<none>                              0.213 -261
+ totalTeachers         1    0.0073 0.206 -261
+ totalStudents         1    0.0064 0.207 -260
+ studentsPerClassRoom  1    0.0052 0.208 -260
+ Girlstoilet           1    0.0044 0.209 -260
+ girlsToiletsPerGirl   1    0.0042 0.209 -260
+ boysToiletsPerBoy     1    0.0022 0.211 -259
+ Meanscore             1    0.0016 0.212 -259
+ totalClassrooms       1    0.0015 0.212 -259
+ studentsPerTeacher    1    0.0012 0.212 -259
+ genderRatio           1    0.0002 0.213 -259
+ Floor                 6    0.0326 0.181 -257
+ Roof                  5    0.0164 0.197 -255

Step:  AIC=-261
avgFrequency ~ relativeDropout + Area + Boystoilets

                       Df Sum of Sq   RSS  AIC
+ totalTeachers         1    0.0087 0.196 -261
<none>                              0.204 -261
+ Meanscore             1    0.0040 0.201 -260
+ girlsToiletsPerGirl   1    0.0016 0.203 -259
+ Girlstoilet           1    0.0015 0.203 -259
+ studentsPerTeacher    1    0.0012 0.203 -259
+ studentsPerClassRoom  1    0.0011 0.203 -259
+ totalStudents         1    0.0009 0.204 -259
+ boysToiletsPerBoy     1    0.0003 0.204 -259
+ totalClassrooms       1    0.0000 0.204 -259
+ genderRatio           1    0.0000 0.204 -259
+ Floor                 6    0.0342 0.170 -258
+ Roof                  5    0.0133 0.191 -254

Step:  AIC=-261.1
avgFrequency ~ relativeDropout + Area + Boystoilets + totalTeachers

                       Df Sum of Sq   RSS  AIC
<none>                              0.196 -261
+ girlsToiletsPerGirl   1    0.0056 0.190 -261
+ Meanscore             1    0.0044 0.192 -260
+ Girlstoilet           1    0.0040 0.192 -260
+ boysToiletsPerBoy     1    0.0028 0.193 -260
+ totalClassrooms       1    0.0015 0.194 -260
+ genderRatio           1    0.0003 0.196 -259
+ totalStudents         1    0.0003 0.196 -259
+ studentsPerTeacher    1    0.0001 0.196 -259
+ studentsPerClassRoom  1    0.0000 0.196 -259
+ Floor                  6    0.0346 0.161 -259
+ Roof                   5    0.0128 0.183 -254
```

```r
summary(steppedModel)
```

```

Call:
lm(formula = avgFrequency ~ relativeDropout + Area + Boystoilets +
    totalTeachers, data = school.df)

Residuals:
     Min       1Q   Median       3Q      Max
-0.13294 -0.03961 -0.00275  0.04040  0.20584

Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)      0.36905    0.04608    8.01  5.4e-10 ***
relativeDropout -0.20906    0.03090   -6.77  3.1e-08 ***
AreaKawempe      0.01526    0.02874    0.53  0.59820
AreaLubaga       0.09063    0.03991    2.27  0.02834 *
AreaMakindye     0.11798    0.03105    3.80  0.00046 ***
AreaNakawa       0.06508    0.02709    2.40  0.02079 *
Boystoilets     -0.00297    0.00201   -1.48  0.14726
totalTeachers   -0.00184    0.00135   -1.36  0.17997
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.0683 on 42 degrees of freedom
Multiple R-squared:  0.617, Adjusted R-squared:  0.553
F-statistic: 9.66 on 7 and 42 DF,  p-value: 4.23e-07
```

As we can see, the algorithm picks up a set of covariates that is similar to our most highly associated covariates from the prior section. While we include relative dropout rate and area in the model, floor type is not selected, while several other variables are included instead. In practice, the researcher should validate that these variables make sense and add or remove them as needed.

Let's proceed with this model for now and estimate its accuracy. Rather than using the R-squared measure provided by the lm() function, we use ten-fold cross-validation to estimate our algorithm's performance on unseen data. This gives us a much better estimate for how our model would perform in practice. Code is provided below:

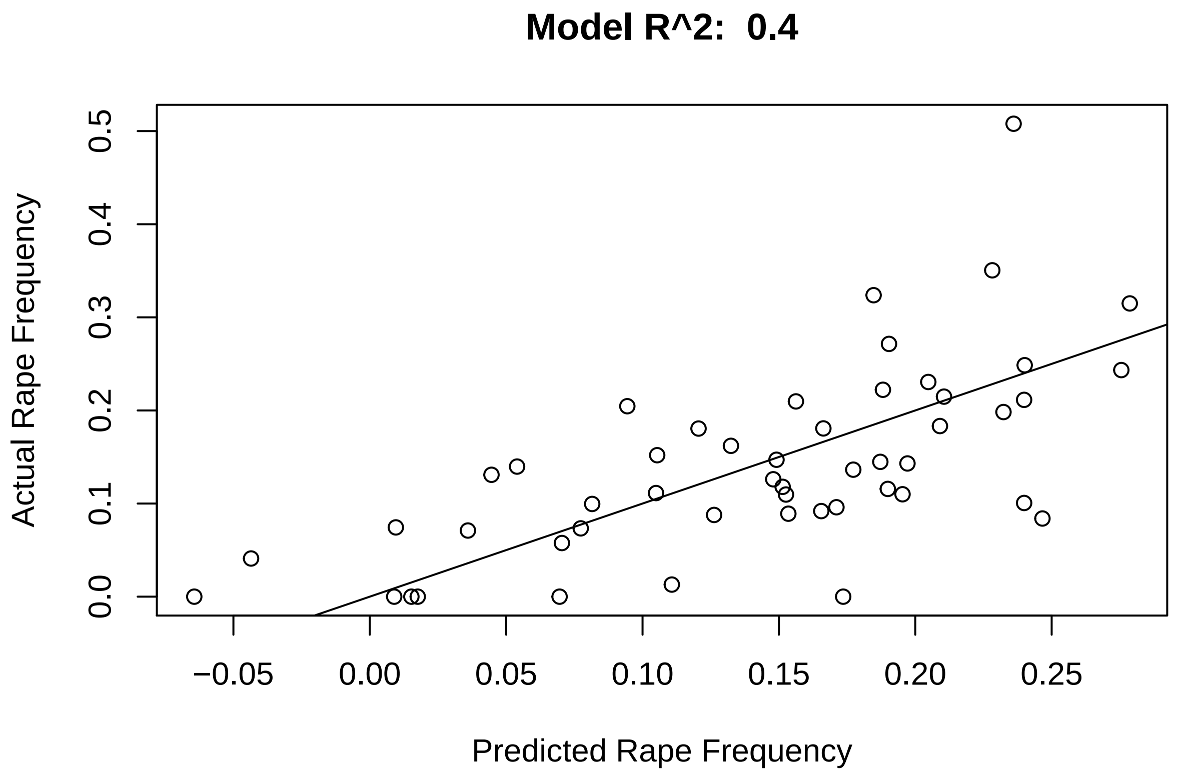
```r
# evaluate model accuracy via CV
folds <- sample(rep(1:10, 5))
preds <- rep(0, nrow(school.df))
sapply(unique(folds), FUN = function(fold) {
  mod <- lm(steppedModel, data = school.df[folds != fold,])
  preds[folds == fold] <<- predict(mod, newdata = school.df[folds == fold,])
})
```

```
##       [,1]   [,2]      [,3]   [,4]    [,5]    [,6]   [,7]    [,8]     [,9]
## 1  0.17110 0.1881  0.246614 0.1107 0.04461 0.16630 0.1262 0.01758  0.27864
## 18 0.06959 0.1971 -0.043531 0.2401 0.08154 0.18472 0.2399 0.18992  0.17727
## 36 0.05400 0.1491  0.008949 0.1479 0.23234 0.23605 0.1736 0.15138  0.01525
## 46 0.23991 0.1953  0.077343 0.1526 0.18718 0.07044 0.2755 0.03598  0.00955
## 50 0.22823 0.1655  0.094416 0.2090 0.10538 0.19036 0.1562 0.20477 -0.06435
##     [,10]
## 1  0.2105
## 18 0.1205
## 36 0.1324
## 46 0.1049
## 50 0.1535
```

```r
r2.cv <- 1 - sum((preds - school.df$avgFrequency)^2)/
  sum((school.df$avgFrequency - mean(school.df$avgFrequency))^2)
```

We plot the model accuracy below. Our model appears to do a decent – but not great – job at predicting rape frequency, achieving a cross-validated R-squared of about 42%.

```r
plot(x = preds, y = school.df$avgFrequency, xlab = "Predicted Rape Frequency",
     ylab = "Actual Rape Frequency", main = paste("Model R^2: ", round(r2.cv, 2)))
abline(0, 1)
```


Model R^2: 0.4
Actual Rape Frequency
Predicted Rape Frequency