Capstone Project - Predicting Patient No-Shows Using Appointment Data

Derek Samsom

Missed medical appointments are a major problem in the medical industry, resulting in lost revenue. Medical providers can over-book appointments to try to minimize the lost revenue, but without any way to predict the probability of an appointment being missed, there will be times where more or fewer patients show up at a given time than expected. The result will be that lost revenue will be reduced but not eliminated, as there will still be times that more appointments are missed than expected. There will aslo be times more apppointments show up than expected, which can overwhelm staff and resources and affect the level of patient care.

This project is a classification problem that will explore the prediction of whether a medical appointment will be missed, and its probability of being kept or missed. The goal of this project is to minimize the prediction error of missed appointments, as the error results in times where there are too many or too few patients at a given time. This is important to the client because it reduces the revenue loss caused by missed appontments in a way that reduces the undesired consequences of overbooking.

There are countless reasons and circumstances that can lead someone to miss an appointment, such as a last minute work meeting or a family emergency, that aren't directly captured in the data and are impossible to know in advance of future appointments. Missed appointments can only be predicted based on indirect factors that are known, such as patient history and demographics. Because of this, there will be a level of error that cannot be eliminated, however, any reduction in error compared to having no predictive model at all is still beneficial as it allows more overbooking to be done with fewer negative consequences.

Medical providers can use the missed appointment predictions by incorporating them into their booking methods and systems. The methods used in booking will have to consider the implications of the inherent prediction errors and balance the risk the errors represent: too many patients leading to staff/resource shortage, and too few patients leading to lost revenue. The methods of implementing the use of missed appointentment predictions into a booking system are client-specific and not included in the scope of this project, which is limited to minimizing the error while predicting the classification and associated probability that an appointment will be missed or kept.

I will start off by loading the required packages and the data.

```
library(tidyverse)
library(lubridate)
library(caret)

appointments <- read_csv("Final_Data.csv")
appointments_original <- appointments
zipcodes <- read_csv("zipcodes.csv")
```

The raw data, which has been named `appointments`, contains information on 342862 past appointments, pre-sorted by the date and time of appointment. The dependended variable, `kept_status`, shows whether the appintment was kept or missed.

There is no field that can be used to identify a specific patient in the data set. A patient may have had more than one appointment during the time-period represented in the data, meaning that one individual patient may make up one or multiple observations. If there was a patient ID field, it would allow the data to be grouped by patient and give the option of organizing the data by patient rather than by appointment.

A secondary data set, `zipcodes`, has information about the county the offices are located in. This will be used to see if the location can help predict whether an appointment will be missed. The county names are converted to a 2-letter code for confidentiality.

Data Summary and Structure

```
summary(appointments)
```

```
##  kept_status         appt_date          appt_time         appt_length
##  Length:342862      Length:342862      Length:342862     Min.   : 10
##  Class :character   Class :character   Class1:hms        1st Qu.: 60
##  Mode  :character   Mode  :character   Class2:difftime   Median : 60
##                                        Mode  :numeric    Mean   : 57
##                                                          3rd Qu.: 60
##                                                          Max.   :600
##
##  date_scheduled      patient_age     patient_gender     billing_type
##  Length:342862      Min.   :  0.00   Length:342862      Length:342862
##  Class :character   1st Qu.: 17.00   Class :character   Class :character
##  Mode  :character   Median : 34.00   Mode  :character   Mode  :character
##                     Mean   : 35.56
##                     3rd Qu.: 54.00
##                     Max.   :264.00
##
##   prior_missed       prior_kept     patient_distance  office_zip
##  Min.   :  0.000   Min.   :  0.00   Min.   :   0.0   Length:342862
##  1st Qu.:  1.000   1st Qu.:  2.00   1st Qu.:   0.0   Class :character
##  Median :  2.000   Median :  6.00   Median :   3.0   Mode  :character
##  Mean   :  2.451   Mean   :  8.02   Mean   :  10.8
##  3rd Qu.:  3.000   3rd Qu.: 11.00   3rd Qu.:   9.0
##  Max.   :117.000   Max.   :676.00   Max.   :2688.0
##                                     NA's   :974
##  provider_specialty remind_call_result
##  Length:342862      Length:342862
##  Class :character   Class :character
##  Mode  :character   Mode  :character
##
##
##
##
```

```
str(appointments, give.attr = FALSE)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    342862 obs. of  14 variables:
##  $ kept_status       : chr  "Kept" "Kept" "Kept" "Kept" ...
##  $ appt_date         : chr  "9/1/16" "9/1/16" "9/1/16" "9/1/16" ...
##  $ appt_time         :Classes 'hms', 'difftime'  atomic [1:342862] 19800 28800 28800 28800 28800 288
##  $ appt_length       : int  90 60 120 60 60 60 60 60 60 90 ...
##  $ date_scheduled    : chr  "8/1/16" "1/18/16" "2/3/16" "6/8/16" ...
##  $ patient_age       : int  7 75 31 45 49 71 49 38 36 13 ...
##  $ patient_gender    : chr  "Male" "Female" "Male" "Male" ...
##  $ billing_type      : chr  "DMAP" "Commercial" "DMAP" "DMAP" ...
##  $ prior_missed      : int  1 2 1 6 5 6 8 0 2 3 ...
##  $ prior_kept        : int  3 5 5 15 6 6 20 0 5 12 ...
##  $ patient_distance  : int  41 29 5 5 0 5 0 539 0 4 ...
##  $ office_zip        : chr  "AP" "BL" "BL" "BL" ...
```

```
##  $ provider_specialty: chr  "A" "A" "A" "B" ...
##  $ remind_call_result: chr  "Left Message" "Answered - Confirmed" "Left Message" "Answered - No Resp
```

```r
head(appointments[, 1:5])
```

```
## # A tibble: 6 x 5
##   kept_status appt_date appt_time appt_length date_scheduled
##   <chr>       <chr>     <time>          <int> <chr>
## 1 Kept        9/1/16    05:30              90 8/1/16
## 2 Kept        9/1/16    08:00              60 1/18/16
## 3 Kept        9/1/16    08:00             120 2/3/16
## 4 Kept        9/1/16    08:00              60 6/8/16
## 5 Missed      9/1/16    08:00              60 6/28/16
## 6 Kept        9/1/16    08:00              60 7/12/16
```

```r
head(appointments[, 6:10])
```

```
## # A tibble: 6 x 5
##   patient_age patient_gender billing_type prior_missed prior_kept
##         <int> <chr>          <chr>               <int>      <int>
## 1           7 Male           DMAP                    1          3
## 2          75 Female         Commercial              2          5
## 3          31 Male           DMAP                    1          5
## 4          45 Male           DMAP                    6         15
## 5          49 Male           Commercial              5          6
## 6          71 Male           DMAP                    6          6
```

```r
head(appointments[, 11:14])
```

```
## # A tibble: 6 x 4
##   patient_distance office_zip provider_specialty remind_call_result
##              <int> <chr>      <chr>              <chr>
## 1               41 AP         A                  Left Message
## 2               29 BL         A                  Answered - Confirmed
## 3                5 BL         A                  Left Message
## 4                5 BL         B                  Answered - No Response
## 5                0 BL         B                  Answered - No Response
## 6                5 BL         A                  Answered - Confirmed
```

Data Dictionary

```r
var_descriptions <- c(
    "Dependent variable: kept or missed",
    "Appointment date",
    "Appointment time",
    "Appointment length in minutes",
    "Date appointment was scheduled",
    "Patient age",
    "Patient gender",
    "Billing type",
    "Number of prior missed appointments",
    "Number of prior kept appointments",
    "Patient distance from office in miles",
    "Office Zip Code - Anonymized",
    "Provider primary specialty code",
```

```
    "Reminder Call result")
var <- colnames(appointments)
var_type <- unlist(map(appointments, class))
var_type <- var_type[-4]
as_data_frame(cbind(c(1:length(var)), var, var_type, var_descriptions))
```

```
## # A tibble: 14 x 4
##    V1    var                 var_type  var_descriptions
##    <chr> <chr>               <chr>     <chr>
##  1 1     kept_status         character Dependent variable: kept or missed
##  2 2     appt_date           character Appointment date
##  3 3     appt_time           hms       Appointment time
##  4 4     appt_length         integer   Appointment length in minutes
##  5 5     date_scheduled      character Date appointment was scheduled
##  6 6     patient_age         integer   Patient age
##  7 7     patient_gender      character Patient gender
##  8 8     billing_type        character Billing type
##  9 9     prior_missed        integer   Number of prior missed appointments
## 10 10    prior_kept          integer   Number of prior kept appointments
## 11 11    patient_distance    integer   Patient distance from office in mil~
## 12 12    office_zip          character Office Zip Code - Anonymized
## 13 13    provider_specialty  character Provider primary specialty code
## 14 14    remind_call_result  character Reminder Call result
```

The `appt_date` and `appt_time` variables can be combined into one variable, `appt_datetime`.

```
appointments <- appointments %>%
    mutate(appt_datetime = lubridate::mdy_hms(paste(appt_date, appt_time)))

appointments$date_scheduled <- lubridate::as_date(
    appointments$date_scheduled, format = "%m/%d/%y", tz = "UTC")
```

Data Exploration

First I want to calculate the percent of missed appointments overall by creating a logical variable `missed`, where 1 represents a missed appointment and 0 represents a kept appointment. This will determine the degree of class imbalance.

```
appointments <- appointments %>%
    mutate(missed = ifelse(appointments$kept_status == "Missed", 1, 0))
missed_rate <- mean(appointments$missed)
missed_rate
```

```
## [1] 0.1592944
```

15.93 % of the total appointments are missed. This is an imbalanced classification, which will have implications in the modeling. For example, the model could predict all of the appointments will be kept and be correct 84.07 % of the time. This results in a high accuracy without providing any useful prediction of which appointments will be missed.

Next I want to check the data to see if there are any missing values that could indicate reduced data integrity or adversely affect the modelling.

```
map_dbl(appointments, ~sum(is.na(.)))
```

```
##        kept_status          appt_date          appt_time
```

```
##                   0                  0                  0
##         appt_length     date_scheduled        patient_age
##                   0                  0                  0
##      patient_gender       billing_type       prior_missed
##                   0                  0                  0
##          prior_kept   patient_distance         office_zip
##                   0                974                  0
## provider_specialty remind_call_result      appt_datetime
##                   0                  0                  0
##              missed
##                   0
```

One variable, `patient_distance` has 974 missing value. This is fairly minor considering the size of the data set and will be evaluated later on when exploring the variable further.

Patient Age

I expect missed appointments to vary across age ranges. Perhaps older patients have fewer commitments with children or work, and make their appointments more regularly, or perhaps younger adults might skip more appointments because they aren't as critical.

There are a small number of observations where the age is higher than plausible. Therefore, the observations greater than age 110 will be removed from the data. To make the plot easier to read, I will group the ages. The final model will still use the original continuous variable.

```
age_labels <- c("0-10", "10-20","20-30", "30-40", "40-50", "50-60", "60-70",
                "Over 70")
age_breaks <- c(-1, 10, 20, 30, 40, 50, 60, 70, 111)

appointments <- appointments %>%
    filter(patient_age <= 110) %>%
    mutate(
        age_cat = cut(patient_age, breaks = age_breaks, labels = age_labels))


ggplot(
    appointments,
    aes(x = age_cat, color = kept_status, fill = kept_status)
) +
    stat_count() +
    labs(x = "Age Category")
```

There is a significant difference in missed appointments across the age groups. Missed appointments are highest with young adults, and decrease with older patients. This follows a similar pattern to what I expected.

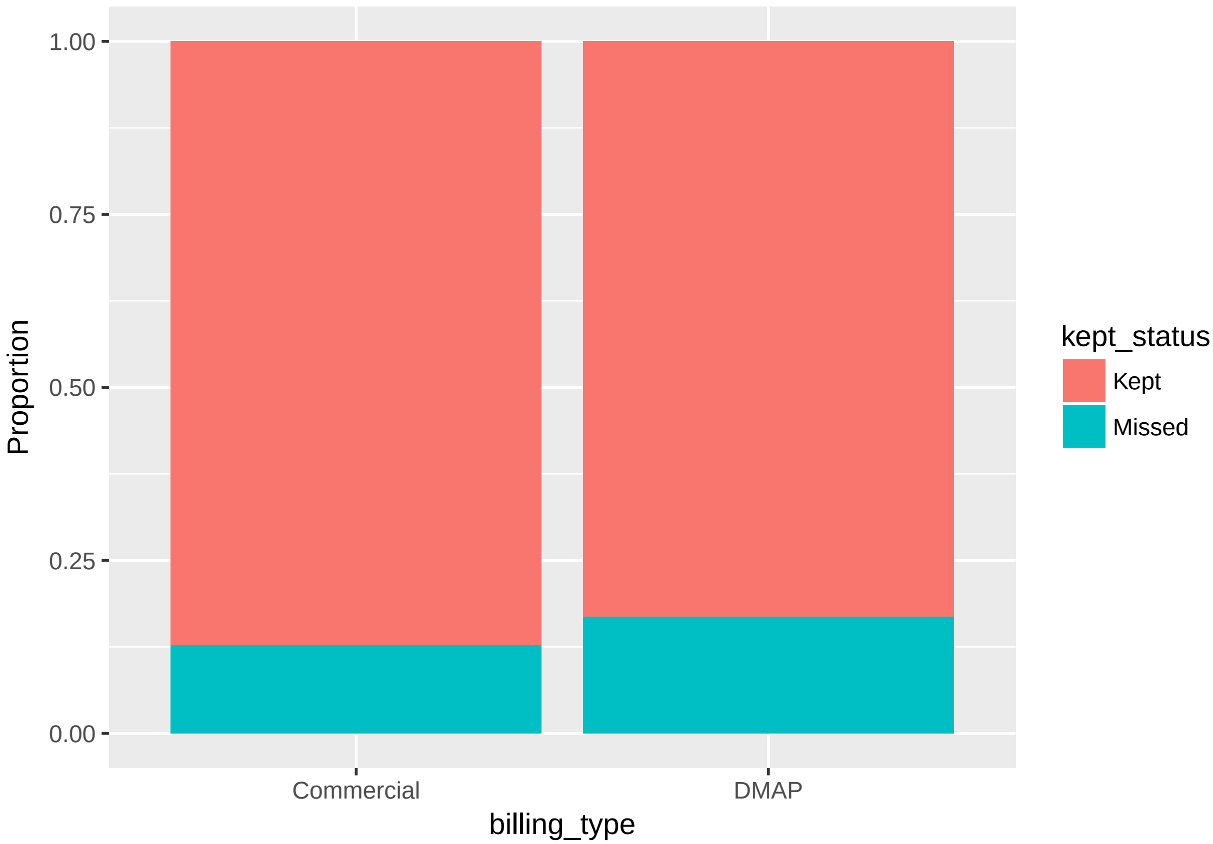
Billing Type

```
table(appointments$billing_type)
```

```
##
##     Commercial            DMAP To Be Assigned
##          78282          264500              1
```

There is only one observation of "To Be Assigned", therefore it will be removed from the data.

```
appointments <- subset(appointments, billing_type != "To Be Assigned")

ggplot(
    appointments,
    aes(x = billing_type, fill = kept_status)
) +
    geom_bar(position = "fill") +
    labs(y = "Proportion")
```



There is a fairly small yet significant difference between billing types. DMAP has a higher proportion of missed appointments than commercial.

Appointment Datetime

For the variable `appt_datetime`, I will create an `hour` variable to see the variation in missed appointments by hour of day. There are many ways the time of day can have an effect, such as rush hour traffic in the morning and afternoon causing more missed appointments, whereas mid-day appointments could be more likely to be missed by work factors.

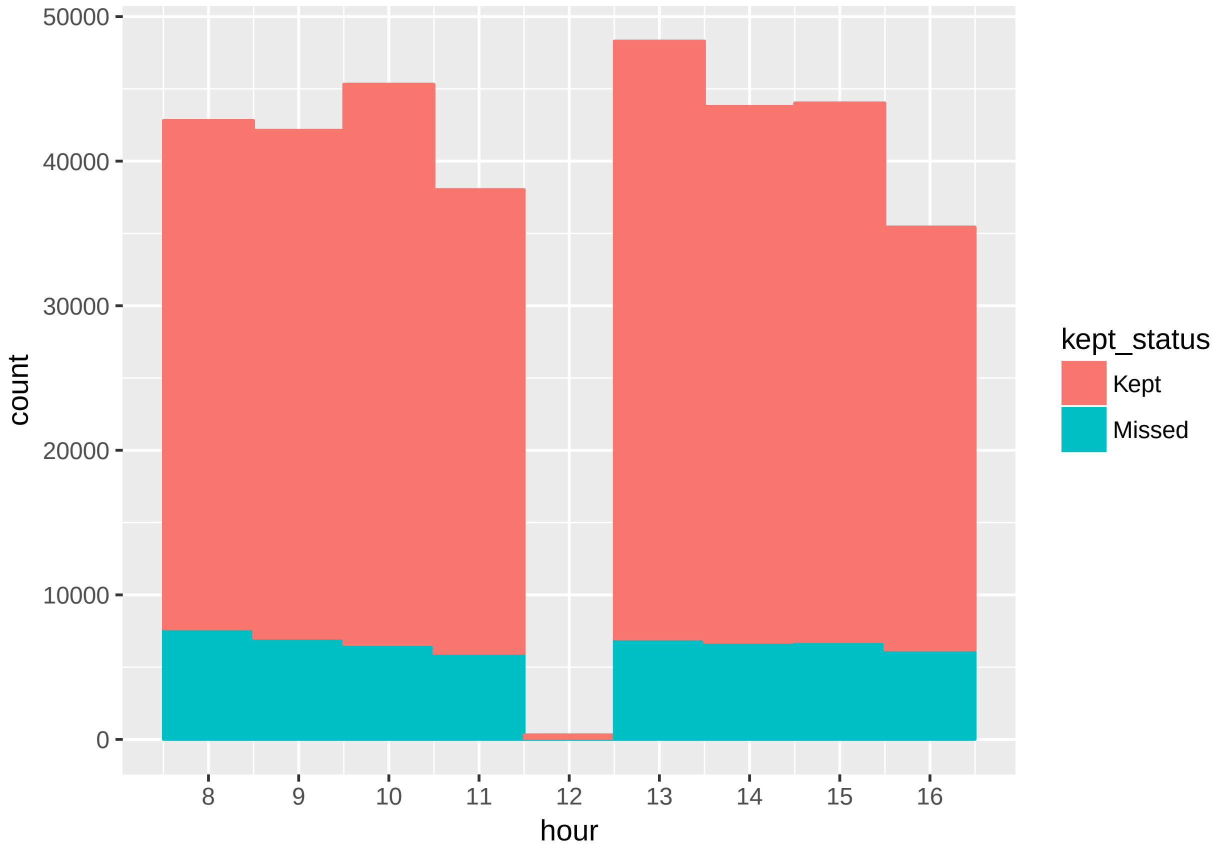
```
appointments <- appointments %>%
    mutate(hour = lubridate::hour(appointments$appt_datetime))

table(appointments$hour)
```

```
##
##     0     5     6     7     8     9    10    11    12    13    14    15
##     7    24    25    98 42816 42133 45326 38033   321 48307 43787 44033
##    16    17    18    19    20    21
## 35449  2180   205    33     3     2
```

Most appointments are scheduled between 8:00 AM and 5:00 PM, with a one hour gap starting at 12:00.

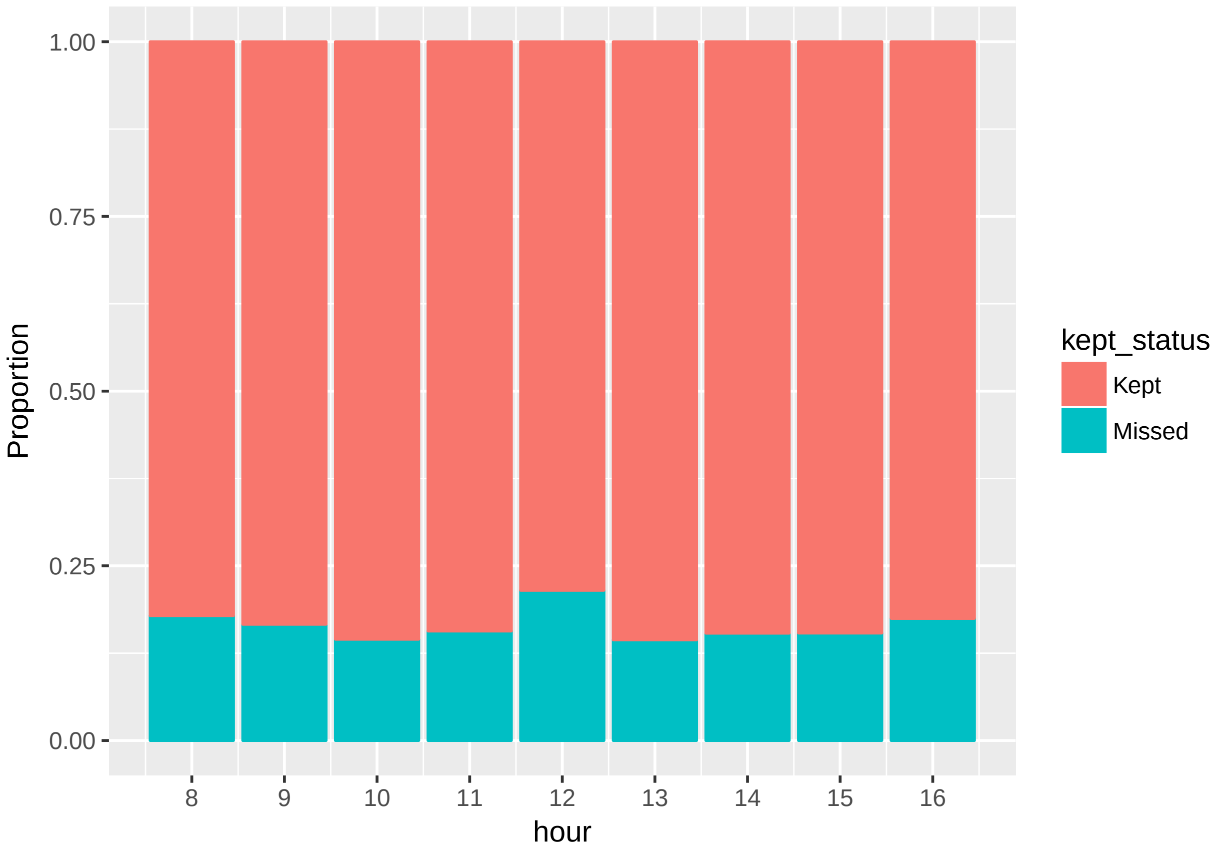
```
appointments_hour <- appointments %>%
    select(kept_status, hour) %>%
    filter(hour >= 8 & hour <= 16)

ggplot(
```

```r
    appointments_hour,
    aes(x = hour, col = kept_status, fill = kept_status)
) +
    geom_histogram(binwidth = 1) +
    scale_x_continuous(breaks = seq(8, 17, 1))
```



There is a decline in the total number of missed appointments as both the morning and afternoon period progress, however, there are fewer appointments towards the end of the two periods. The ratio of missed appointments is hard to read, so I will create a proportional plot to see if it shows any trends.

```r
ggplot(
    appointments_hour,
    aes(x = hour, col = kept_status, fill = kept_status)
) +
    geom_bar(position = "fill") +
    scale_x_continuous(breaks = seq(8, 17, 1)) +
    labs(y = "Proportion")
```



Proportionally more appointments are missed at the beginning and end of the typical scheduling hours, and during the few noon appointments.

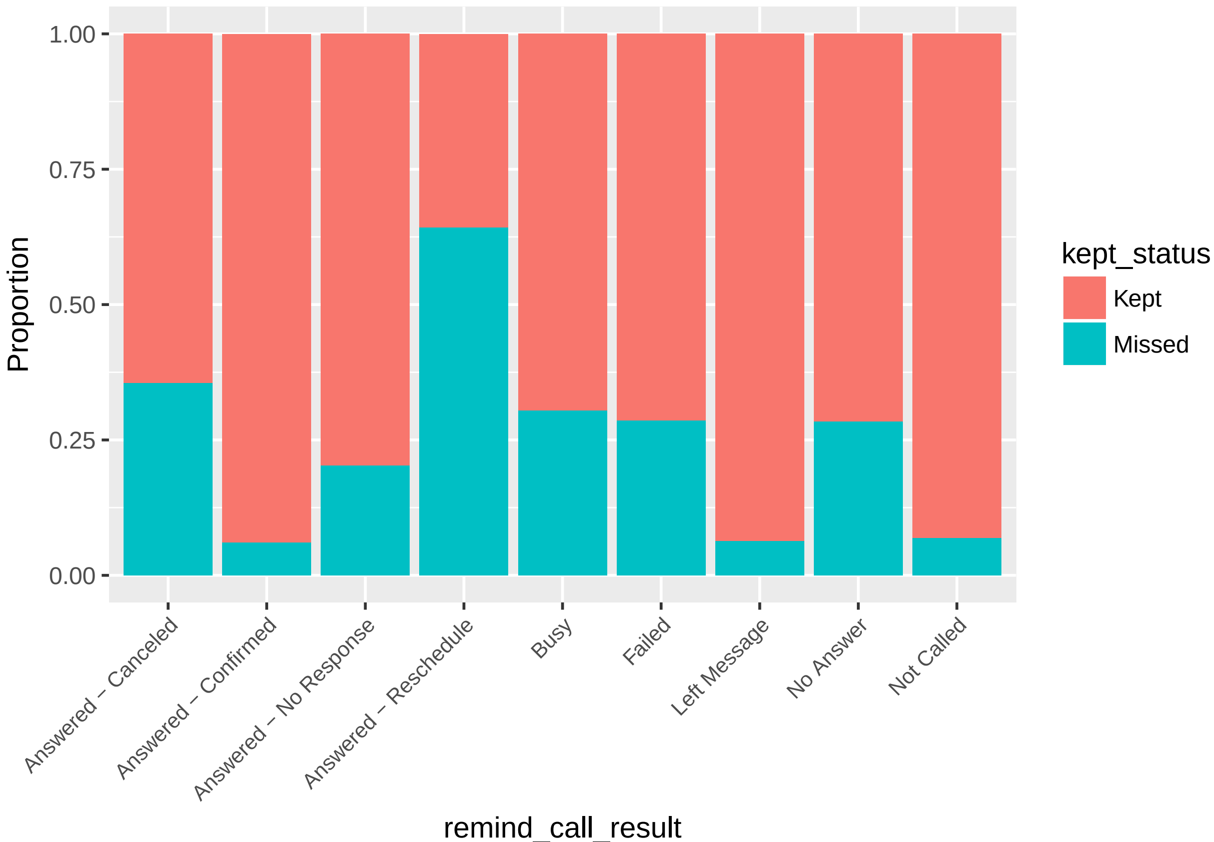
Reminder Call Result

```
table(appointments$remind_call_result)
```

```
##
##    Answered - Canceled   Answered - Confirmed Answered - No Response
##                    152                  49108                 180869
##   Answered - Reschedule                   Busy                 Failed
##                   1369                   1104                  27944
##           Left Message              No Answer             Not Called
##                  18430                    377                  63429
```

There are relatively few instances of "Answered - Cancelled", "Answered - Reschedule", "Busy", and "No Answer"

```
ggplot(
    appointments,
    aes(x = remind_call_result, fill = kept_status)
) +
    geom_bar(position = "fill") +
    theme(axis.text.x = element_text(size = 8, angle = 45,hjust = 1, vjust = 1)) +
    labs(y = "Proportion")
```



Reminder call responses of "Answered - Confirmed", "Left Message", and "Not Called" have the lowest ratios of missed appointments. This seems logical although I would not expect "Not Called" to have as low of a rate, since I would rate it a more neutral response. The best explanation for the low miss rate is a bias in choosing who doesn't need a reminder call.

Responses of "Answered - No Response", "Busy", "Failed", and "No Answer" have average or higher rates of missed appointments, which makes sense as I would rate these responses as neutral to slightly negative.

Responses "Answered - Cancelled" and "Answered - Reschedule" have the highest missed rates of about 35% and 65%, respectively. It's not surprising that these have the highest missed rates, but I would expect them to nearly 100%, particularly for "Answered - Cancelled". Perhaps the patients wanted to cancel or reschdule, but circumstances changed and they ultimately decided to come. Since there are so few cases with these responses, I won't look into this further, but it would be interesting to learn the reasons for these results.

Provider Specialty

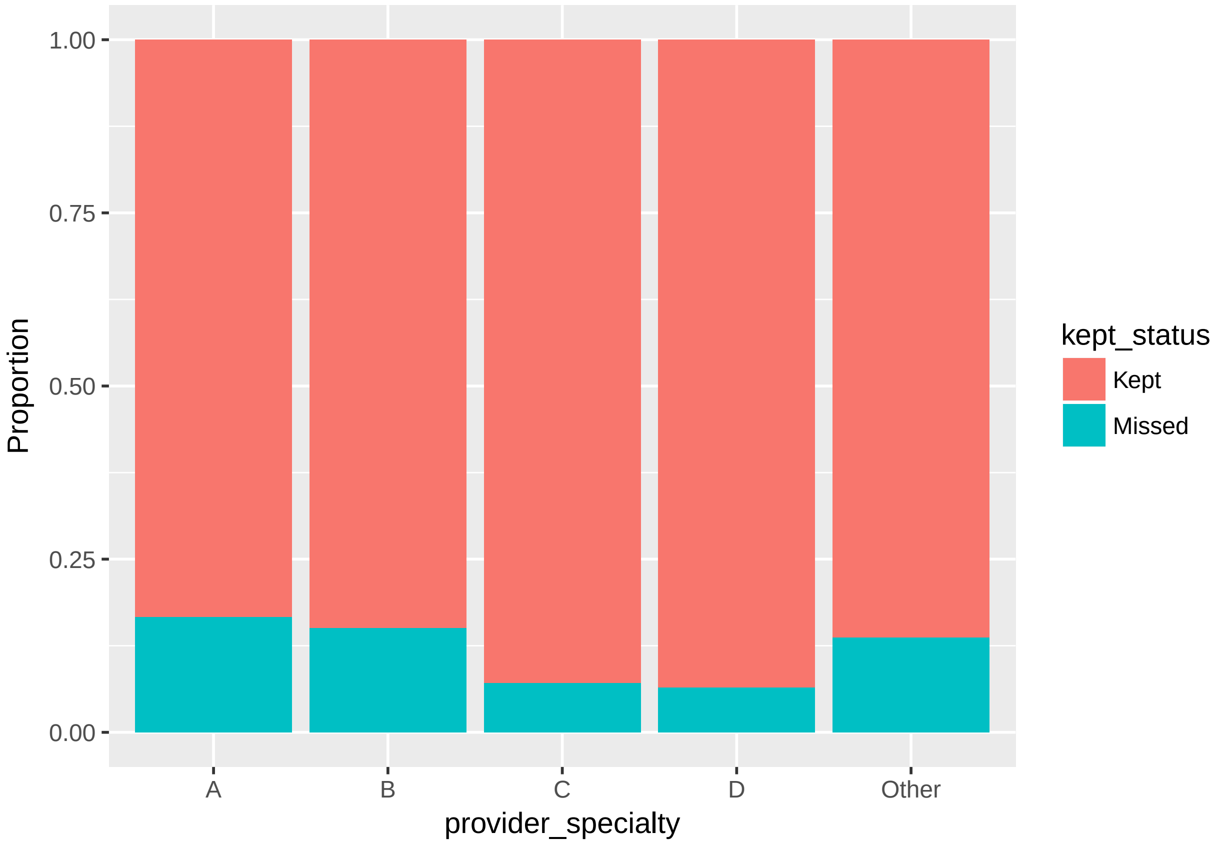
The variable `provider_specialty` indicates what the medical staff assigned to the appointment specializes in, but is encoded for confidentiality.

```
table(appointments$provider_specialty)
```

```
##
##      A      B      C      D      E      F      G
## 246917  84115   5623   5512     42    525     48
```

Most observations are specialty A or B. Specialties E, F, and G have very few observations and will be grouped as "Other".

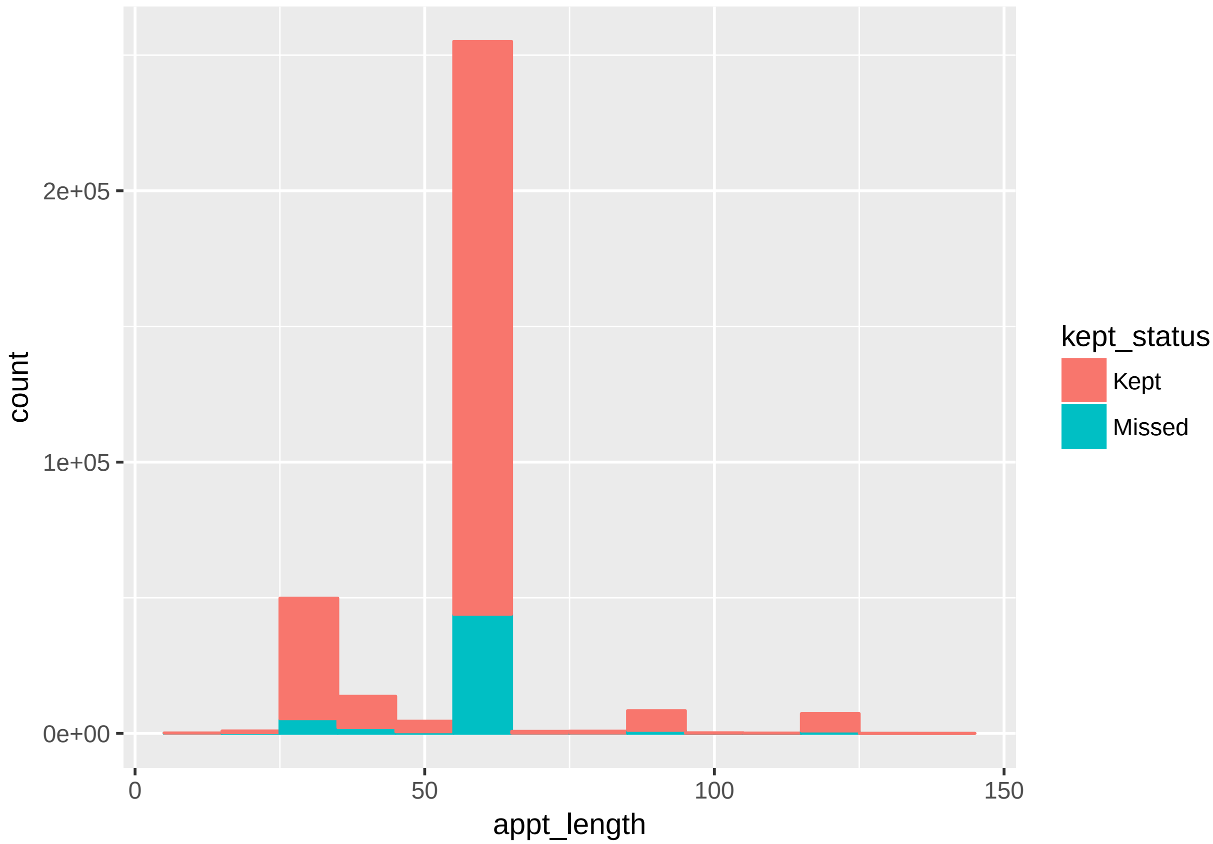
```r
appointments$provider_specialty <- appointments$provider_specialty %>%
    fct_collapse(Other = c("E", "F", "G"))

ggplot(
    appointments,
    aes(x = provider_specialty, fill = kept_status)
) +
    geom_bar(position = "fill") +
    labs(y = "Proportion")
```



The plot shows that specialties “C” and “D” have the lowest rates of missed appointments. Without knowing the details of the specialties, my best hypothesis is that specialties “C” and “D” could be for more critical but less common types of appointments, leading to fewer misses.

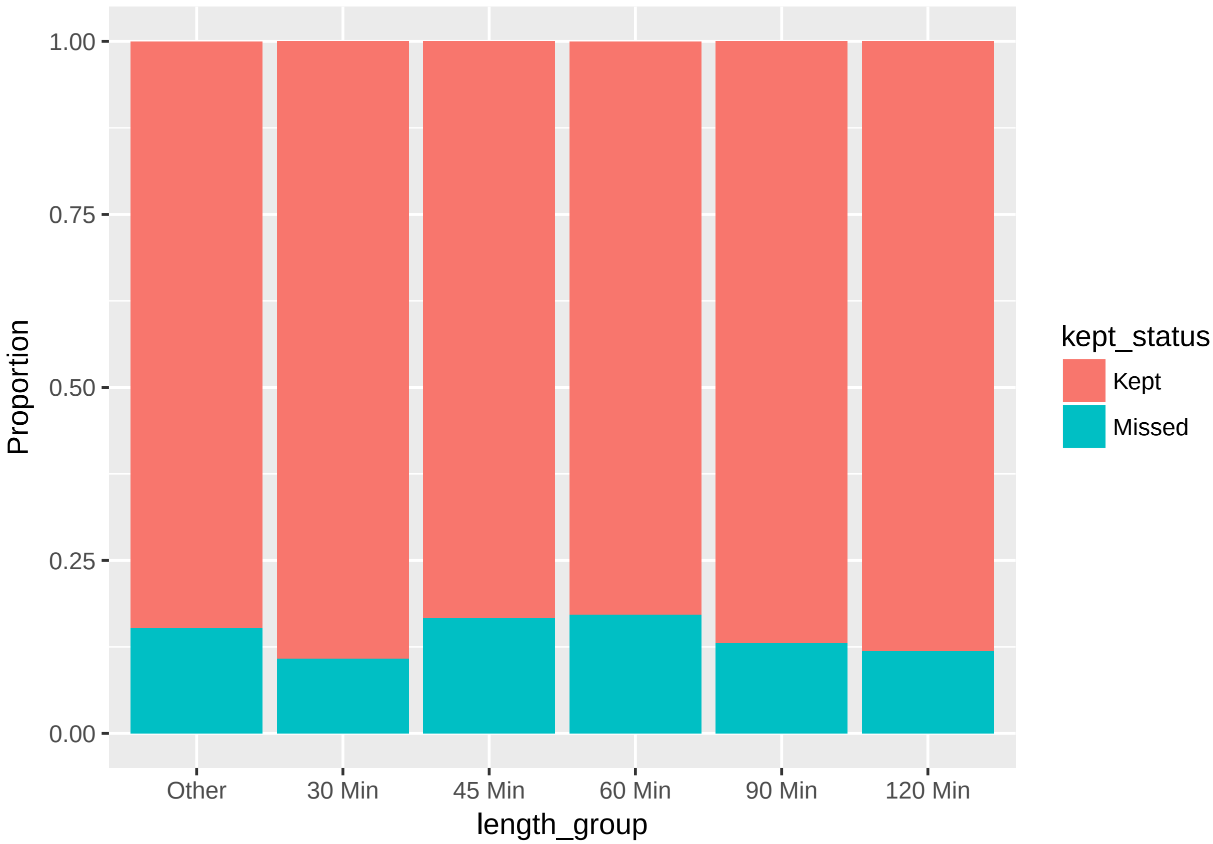
Appointment Length

```r
appointments %>%
    filter(appt_length < 150) %>%
    ggplot(
        aes(x = appt_length, color = kept_status, fill = kept_status)
    ) +
        geom_histogram(binwidth = 10)
```



Most Appointments are 60 minutes long. 30-minute appointments are the next most common, followed by 45-minute, 90-minute, and 120-minute. I'll group them as 30, 45, 60, 90, or 120 minutes, and the rest as "Other", and take a look at the differences.

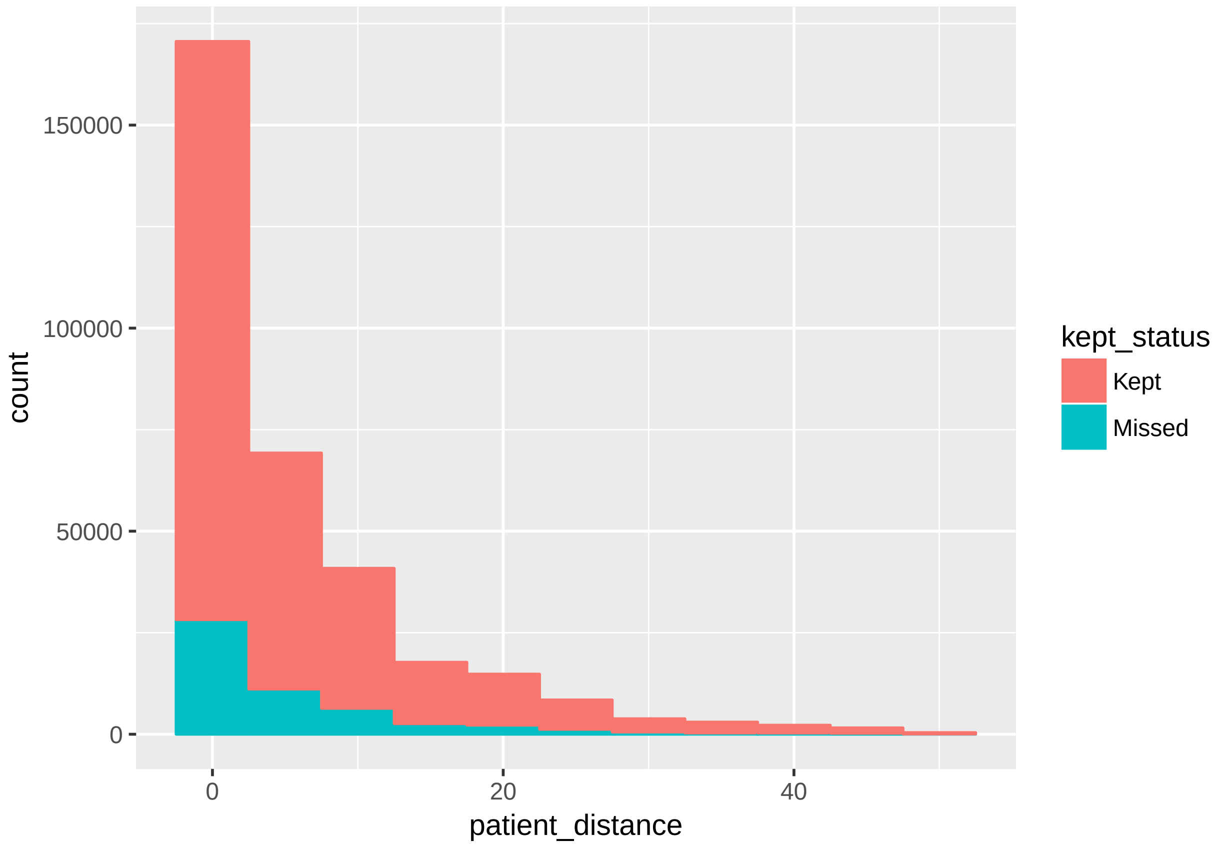
```
length_breaks <- c(-1, 29, 30, 44, 45, 59, 60, 89, 90, 119, 120, 1000)

length_labels <- c(
    "Other1", "30 Min", "Other2", "45 Min", "Other3", "60 Min", "Other4",
    "90 Min", "Other5", "120 Min", "Other6")

appointments <- appointments %>%
    mutate(
        length_group = cut(
            appt_length, breaks = length_breaks, labels = length_labels))

appointments$length_group <- appointments$length_group %>%
    fct_collapse(Other = c("Other1", "Other2", "Other3", "Other4", "Other5",
                           "Other6"))

ggplot(
    data = appointments,
    mapping = aes(x = length_group, fill = kept_status)
) +
    geom_bar(position = "fill") +
    labs(y = "Proportion")
```



Of the most common lengths, 30 and 60 minutes, the longer appointments are more likely to be missed. This could be because a patient is more inclined to go to a shorter appointment, or because shorter appointments are less likely to be impacted by scheduling conflicts. 90 and 120 minute appointments also have lower miss rates, which might be because they are more likely to be for a more important procedure. There could be some interaction between appointment length and provider specialty.

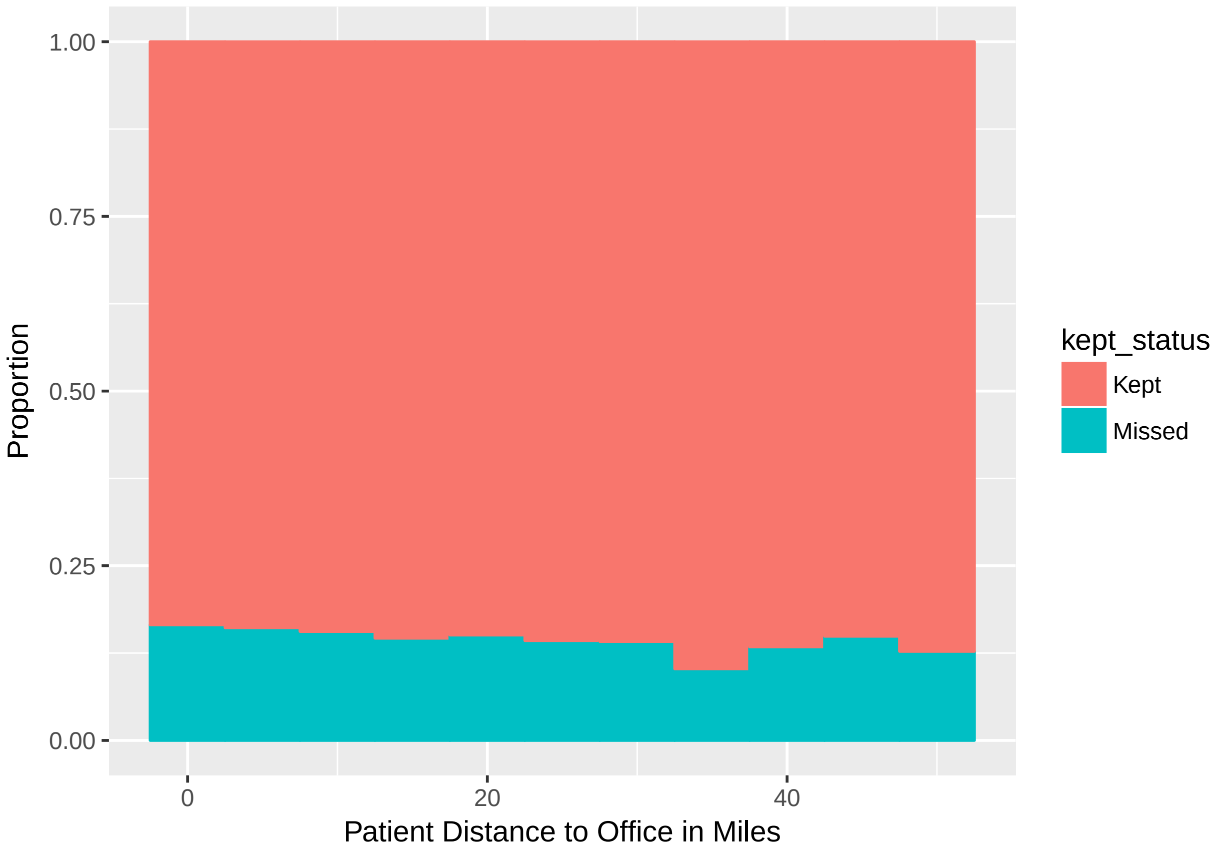
Patient Distance

```r
appointments %>%
    filter(patient_distance < 50) %>%
    ggplot(
        aes(x = patient_distance, color = kept_status, fill = kept_status)
    ) +
        geom_histogram(binwidth = 5)
```



The variable `patient_distance` is the only one with missing values, which will be replaced with median rather than mean since it is a right-skewed distribution as would be expected. I'll add a proportional plot to make it easier to see the change in the miss rate based on distance.

```
appointments$patient_distance <- appointments$patient_distance %>%
    replace_na(median(appointments$patient_distance, na.rm = TRUE))

appointments %>%
    filter(patient_distance < 50) %>%
    ggplot(
        aes(x = patient_distance, color = kept_status, fill = kept_status)
    ) +
        geom_histogram(position = "fill", binwidth = 5) +
        labs(x = "Patient Distance to Office in Miles", y = "Proportion")
```



In general, the closer a patient lives to the office, the more likely they are to miss their appointment. This seems a little counter-intuitive, since the greater the distance from the office, the more potential for there to be hurdles to getting to the appointment. Those who are further away live most likely live in a more rural area, where perhaps trips to town are better planned out and prepared for.

New Variables / Feature Creation

In addition to the original variables, there are several additional variables that can be calculated. Five new variables will be created, summarized below:

The `prior_percent_missed` variable is the percentage of prior appointments missed, calculated by dividing the prior missed appointments by the total number of prior appointments. For new patients, this calculation will result in an error because it will be attempting to divide by zero. The errors will be replaced with zero.

The variable `is_new_patient` will specify whether a patient is new, represented by a 1, or existing, represented by 0. My hypothesis is that new patients are more likely to keep their appointments, since I think it is human nature to try to give a good first impression. This is calculated by searching for appointments where `prior_missed` and `prior_kept` are both 0.

The variable `appt_weekday` is the day of the week the appointments occurs, and `weekday_scheduled` is the date the appointment was booked.

The variable `appt_lead_time` will calculate how far in advance an appointment was booked. This is calculated by taking the difference between `date_scheduled` and `appt_date`. If people are more likely to forget appointments booked farther in advance, or they are more likely to be for less urgent preventative care than last-minute appointments, this will pick that up.

```
appointments <- appointments %>%
    mutate(
        prior_percent_missed = prior_missed / (prior_missed + prior_kept)) %>%
    mutate(
        is_new_patient = ifelse(prior_missed == 0 & prior_kept == 0, 1, 0)) %>%
    mutate(appt_lead_time = date(appt_datetime) - date(date_scheduled)) %>%
    mutate(appt_weekday = strftime(appt_datetime, "%A")) %>%
    mutate(weekday_scheduled = strftime(date_scheduled, "%A"))


appointments$prior_percent_missed <- appointments$prior_percent_missed %>%
    tidyr::replace_na(0)
```

In addition to the calculated variables, the variable `county_code` will be brought in from the zipcode dataset. This will allow differences between geographical areas to be modeled. For example, a more populous county might have more traffic-related missed appointments, or a county more prone to inclement weather could have more weather-related missed appointments.

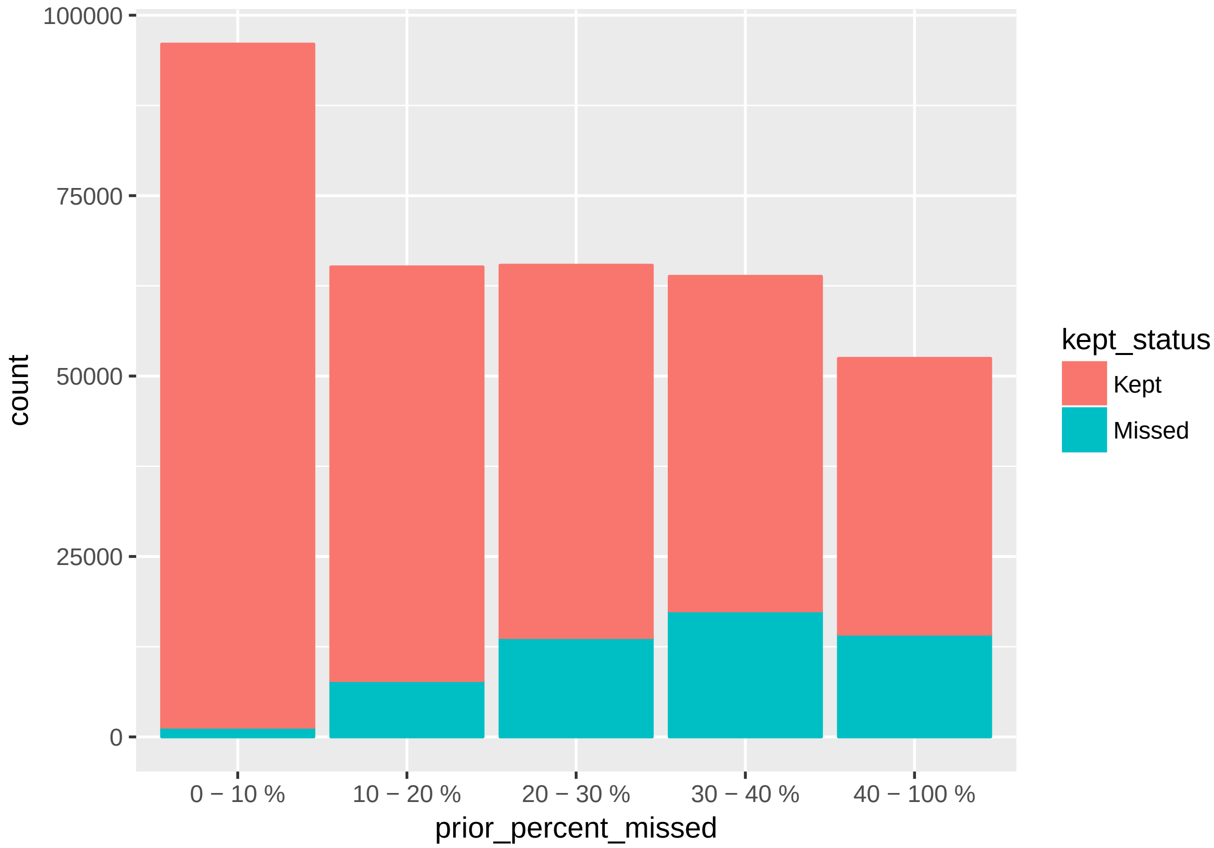
The original data has a zip code variable that could offer more granularity than the county, but I will use the county variable instead since there are fewer possible values that will simplify the model while still offering a decent ability to model different geographical areas.

```
appointments <- dplyr::left_join(appointments, zipcodes, by = "office_zip")
```

Prior Percent Missed

This is expected to be an important variable, because regardless of the reason a patient missed appointments in the past, those same reasons are probably more likely to occur with future appointments. For example, a patient that has missed appointments in the past because they have a very hectic schedule, will be more likely to miss future appointments due to the same hectic schedule. I will break the prior percent missed into groups just to make the plot easier to read, however, the original continuous variable will still be used for the modelling.

```
prior_missed_labels <- c(
    "0 - 10 %", "10 - 20 %", "20 - 30 %", "30 - 40 %", "40 - 100 %")
prior_missed_breaks <- c(-.01, 0.10, 0.20, 0.30, 0.40, 1.01)

appointments <- appointments %>%
    mutate(
        prior_missed_category = cut(
            prior_percent_missed,
            breaks = prior_missed_breaks,
            labels = prior_missed_labels))

ggplot(
    appointments,
    aes(x = prior_missed_category, color = kept_status, fill = kept_status)
) +
    geom_bar() +
    labs(x = "prior_percent_missed")
```



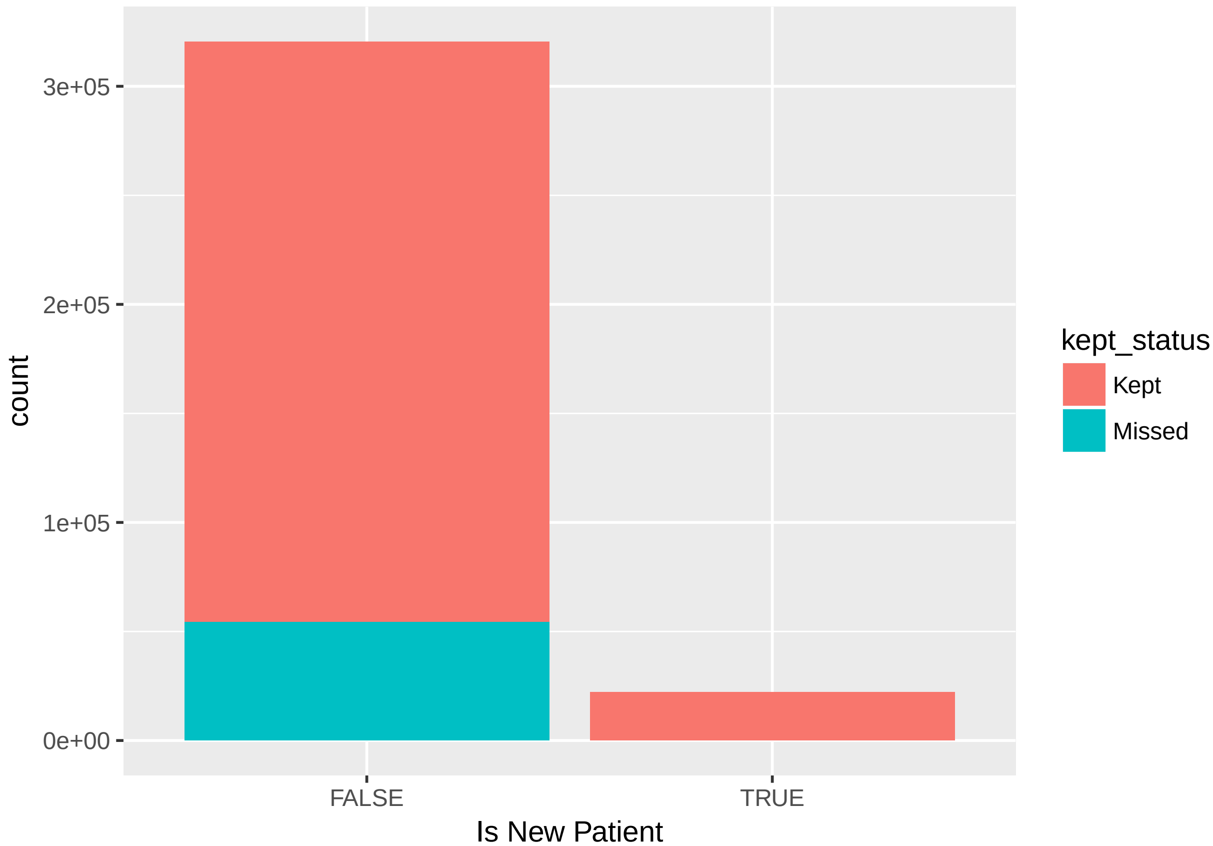
As expected, past percent of appointments missed is a strong predictor of future missed appointments. Appointments where the prior rate is 0 - 10% overwhelmingly kept their appointments. Given this low rate, I would recommend not overbooking these patients, since they are so unlikely to miss to begin with and it is best to keep these patients happy.

Is New Patient

```
table(appointments$is_new_patient)
```

```
##
##      0      1
## 320442  22340
```

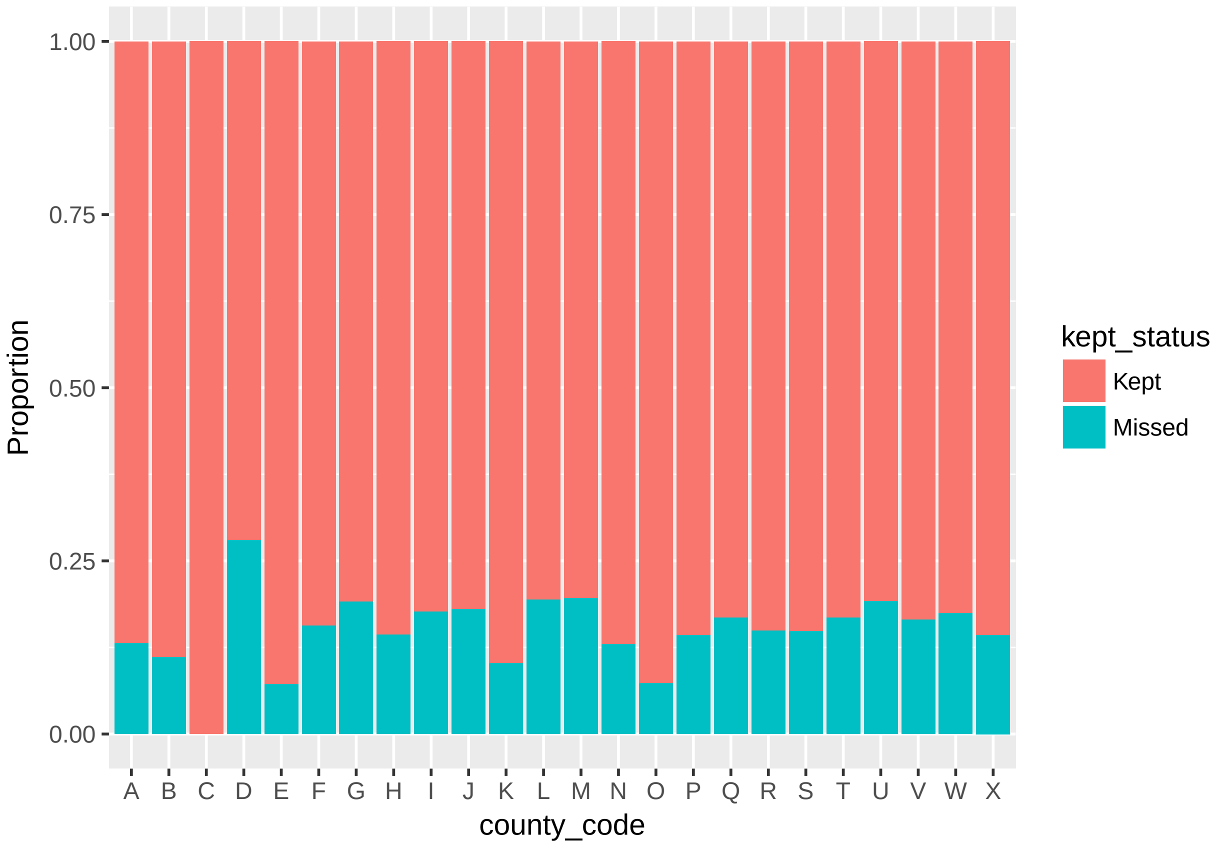
```
ggplot(
    appointments,
    aes(x = as.logical(is_new_patient), fill = kept_status)
) +
    geom_bar() +
    labs(x = "Is New Patient")
```



New patients have a very high percentage of kept appointments, but make up a small percentage of the total appointments.

County Code

```
ggplot(
    appointments,
    aes(x = county_code, fill = kept_status)
) +
    geom_bar(position = "fill") +
    labs(y = "Proportion")
```



There is a significant difference in the rate of missed apppointments between the counties.

Appointment Weekday

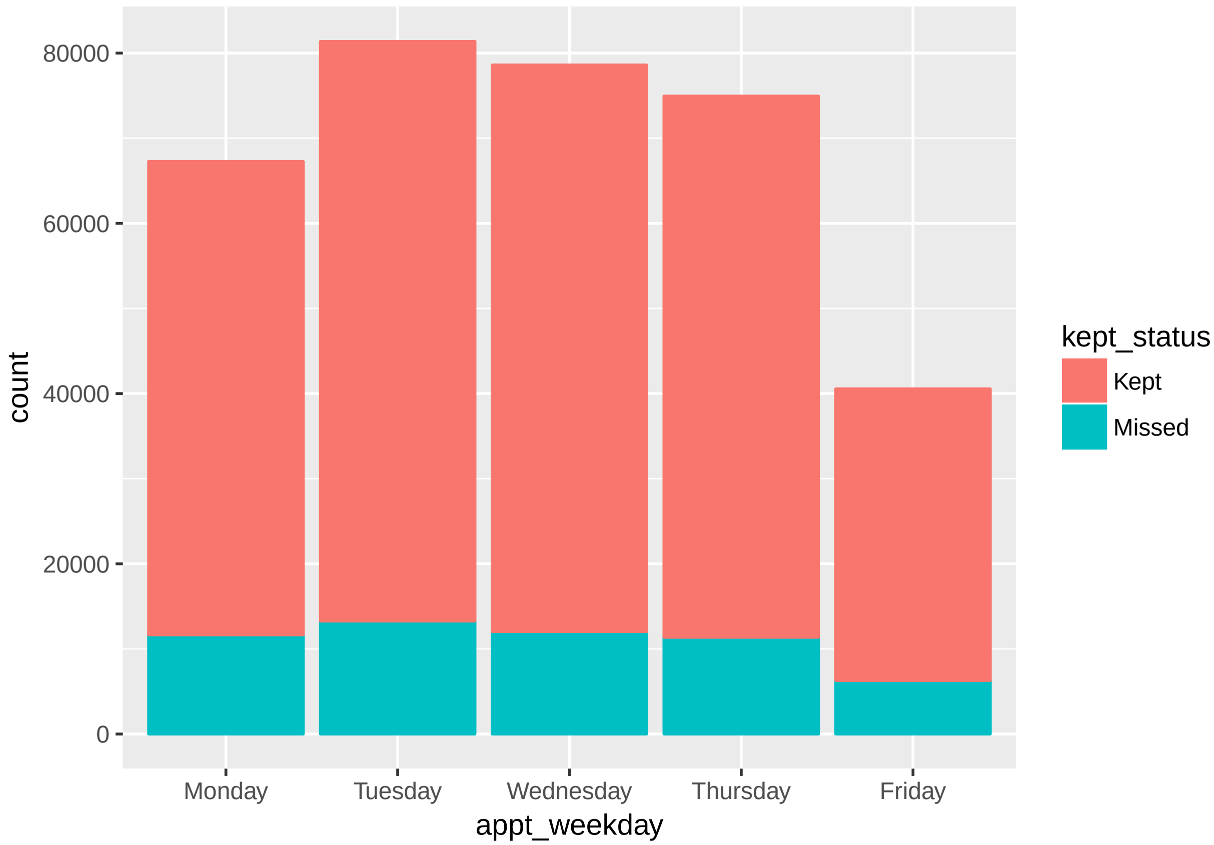
The `appointment_weekday` variable represents the weekday the appointment occurs.

```
table(appointments$appt_weekday)
```

```
##
##    Friday    Monday    Sunday  Thursday   Tuesday Wednesday
##     40558     67280        12     74952     81376     78604
```

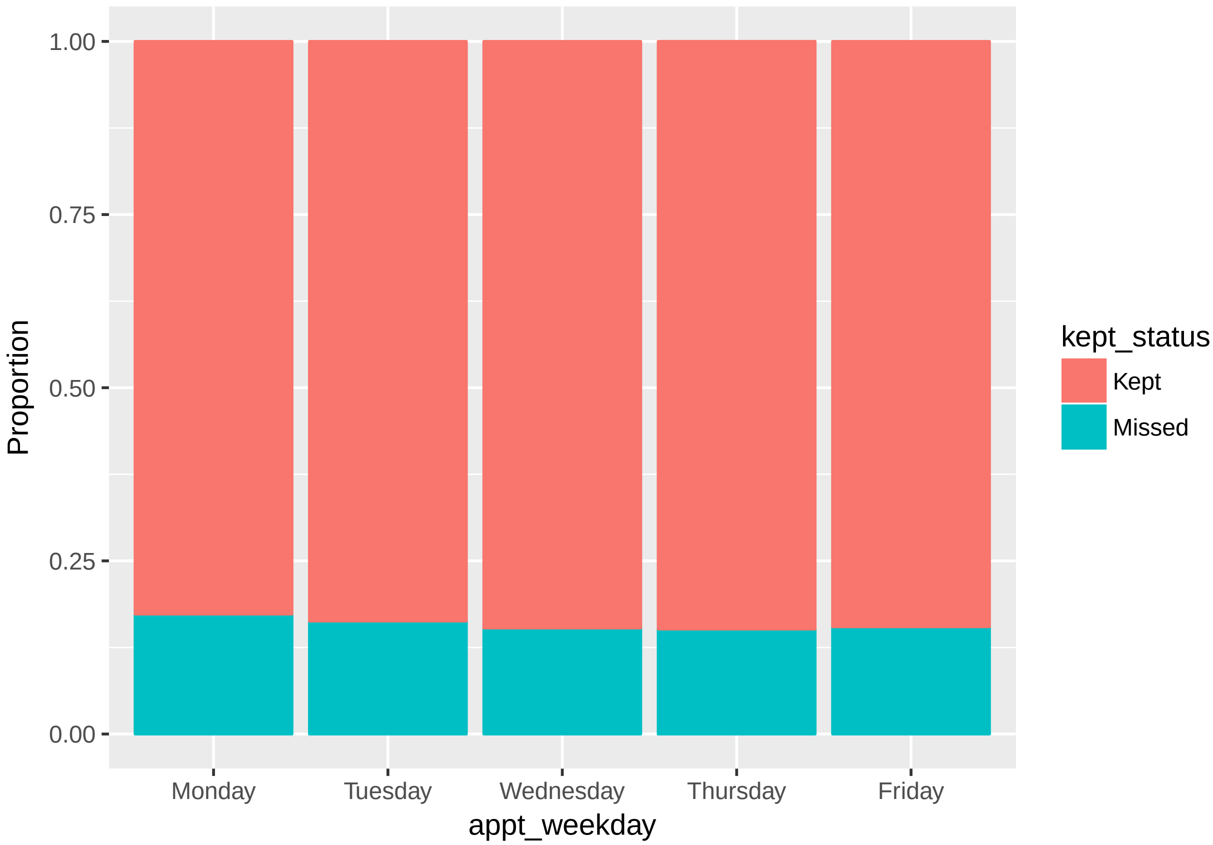
There are only 12 Sunday appointments, so I will remove the observations. I will also convert the character variable to an ordered factor to see the days of the week in the correct order.

```
appointments <- appointments %>%
    filter(appt_weekday != "Sunday")


appointments$appt_weekday <- factor(
    appointments$appt_weekday,
    levels = c("Monday", "Tuesday", "Wednesday", "Thursday", "Friday"))

ggplot(
    appointments,
    aes(x = appt_weekday, color = kept_status, fill = kept_status)
) +
    geom_bar()
```



Most appointments occur on Tuesdays, and trail off later in the week with Friday having the fewest. Due to the difference between the number of appointments each day, it is difficult to see the ratio of missed appoiinments each day, so I will create a proportional plot.

```
ggplot(
    appointments,
    aes(x = appt_weekday, color = kept_status, fill = kept_status)
) +
    geom_bar(position = "fill") +
    labs(y = "Proportion")
```



There is a small variance in the ratio of missed appointments across the days of the week. The highest ratio of missed appointments occurs on Monday, and drops off slightly through Wednesday before leveling off.

Weekday Scheduled

The prevous variable, `appt_weekday`, is the weekday that the appointment occurs, while `weekday_scheduled` is the weekday that the appointment booking was made. For example, if a patient calls on Friday to schedule an appointment for the following Monday, the `appt_weekday` value would be "Monday", and the `weekday_scheduled` value would be "Friday".

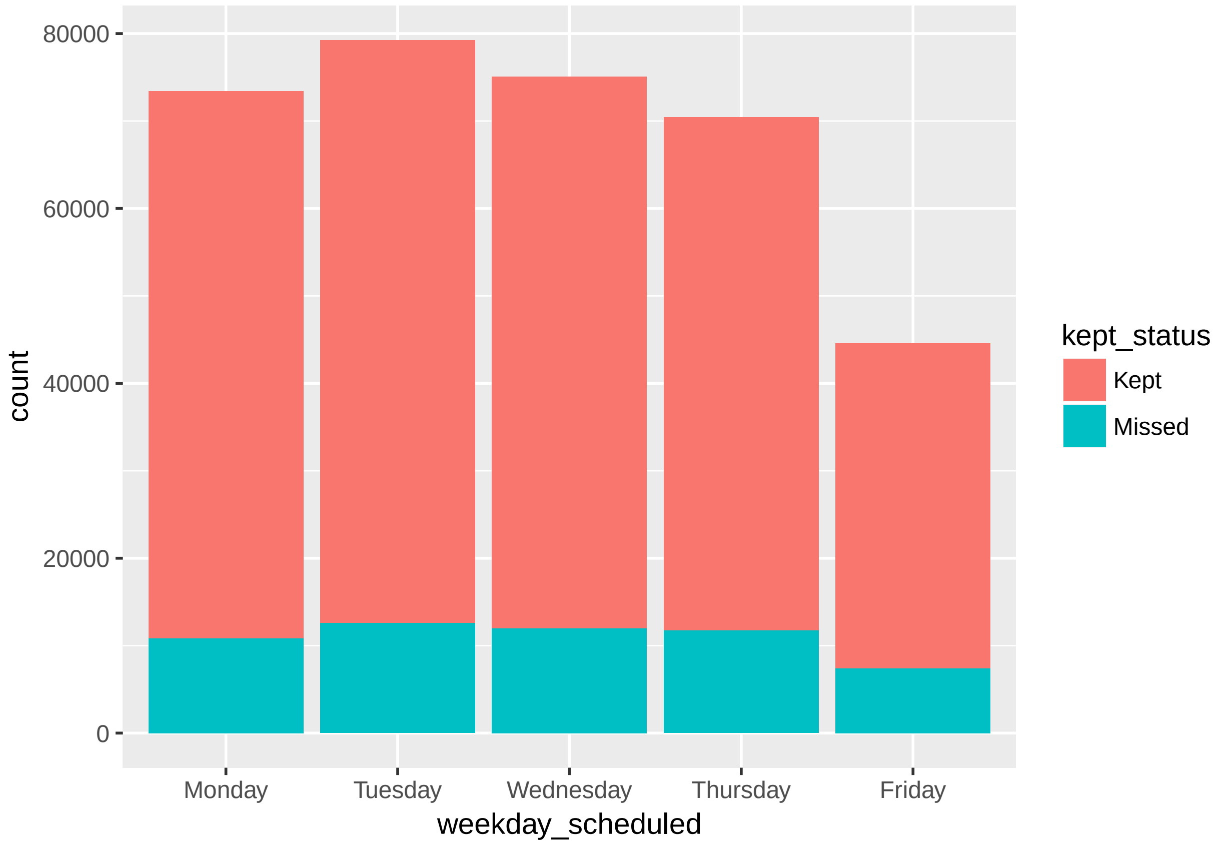
```
table(appointments$weekday_scheduled)
```

```
##
##    Friday     Monday  Saturday     Sunday  Thursday    Tuesday Wednesday
##     44553      73413        43          7     70441      79261     75052
```

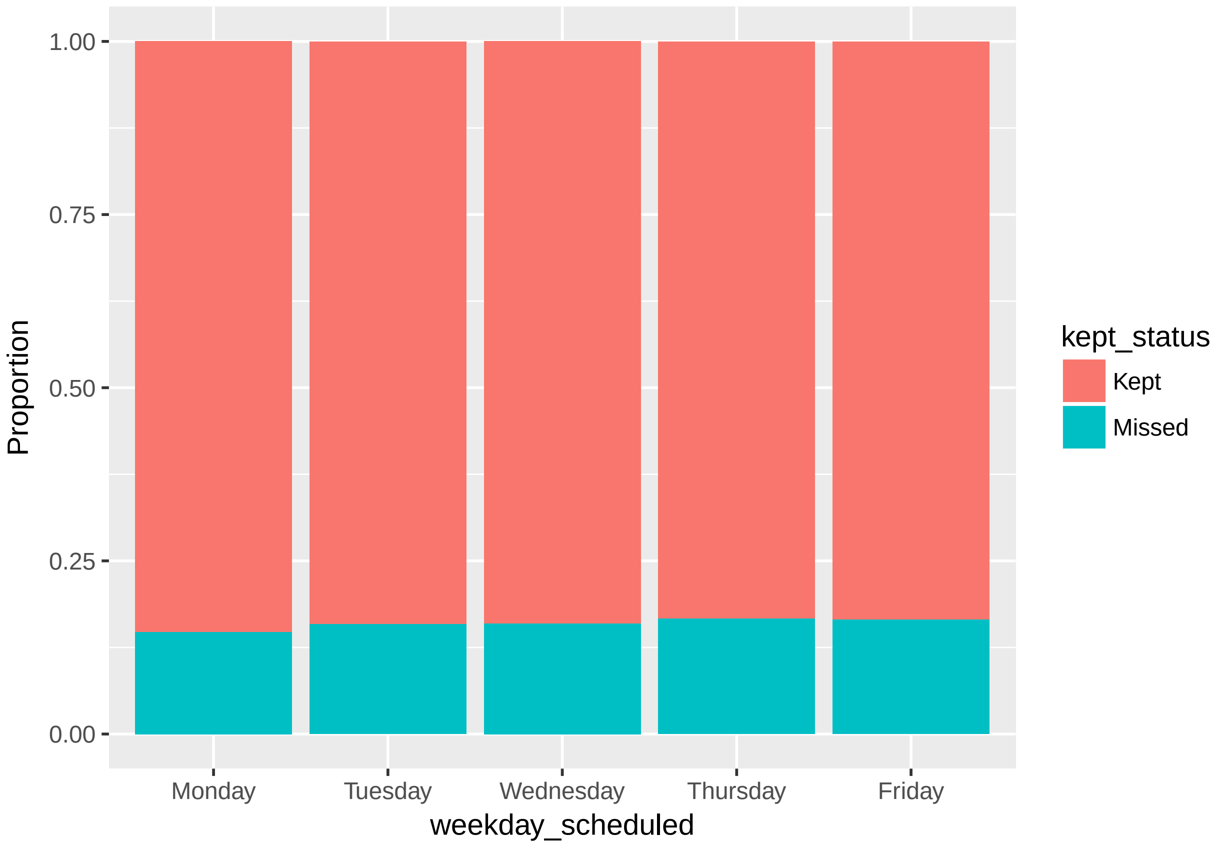
A very small percentage of the observations occur on Saturday or Sunday, so I will remove them.

```
appointments <- appointments %>%
    filter(weekday_scheduled != "Sunday") %>%
    filter(weekday_scheduled != "Saturday")

appointments$weekday_scheduled <- factor(
    appointments$weekday_scheduled,
    levels = c("Monday", "Tuesday", "Wednesday", "Thursday", "Friday"))

ggplot(
    appointments,
```

```r
    aes(x = weekday_scheduled, fill = kept_status)
) +
    geom_bar()
```




```r
ggplot(
    appointments,
    aes(x = weekday_scheduled, fill = kept_status)
) +
    geom_bar(position = "fill") +
    labs(y = "Proportion")
```



The weekday the appointments are scheduled follows a similar pattern as the weekday the appointments occur, peaking on Tuesday and trailing off as the week progresses. However, the weekday the appointment is initially booked has the opposite trend as the weekday the appointment occurs, with the ratio of missed appointments gradually rising throughout the week.

Appt Lead Time

The variable `appt_lead_time` is the number of days between the events in the previous variables `weekday_scheduled` and `appt_weekday`. In the example where a patient calls in Friday to book an appointment for the following Monday, `appt_lead_time` would be 3.

First I will check for negative values. Negative values suggest the appointment occured before it was booked, which wouldn't be possible and must represent a data entry error.

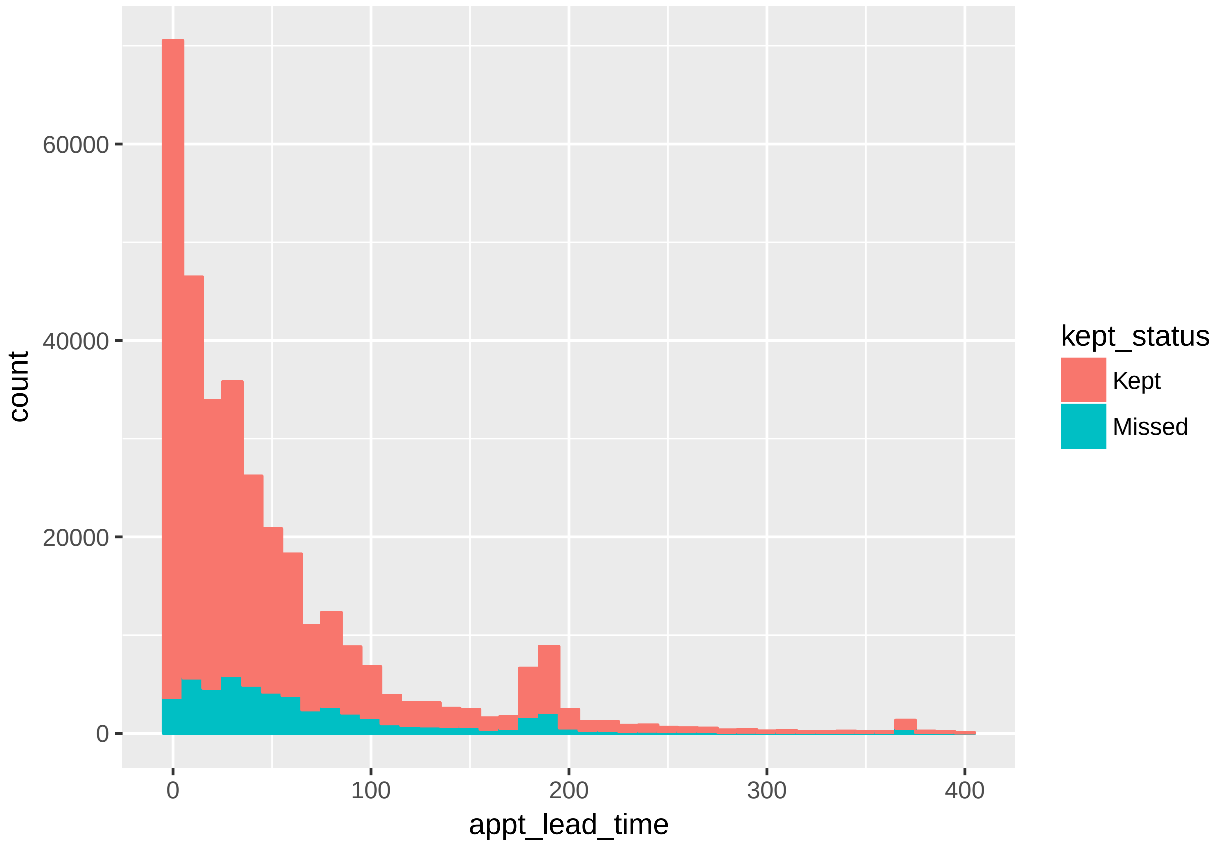
```
length(which(appointments$appt_lead_time < 0))
```

```
## [1] 82
```

There are 82 negative values, which will be replaced with zero.

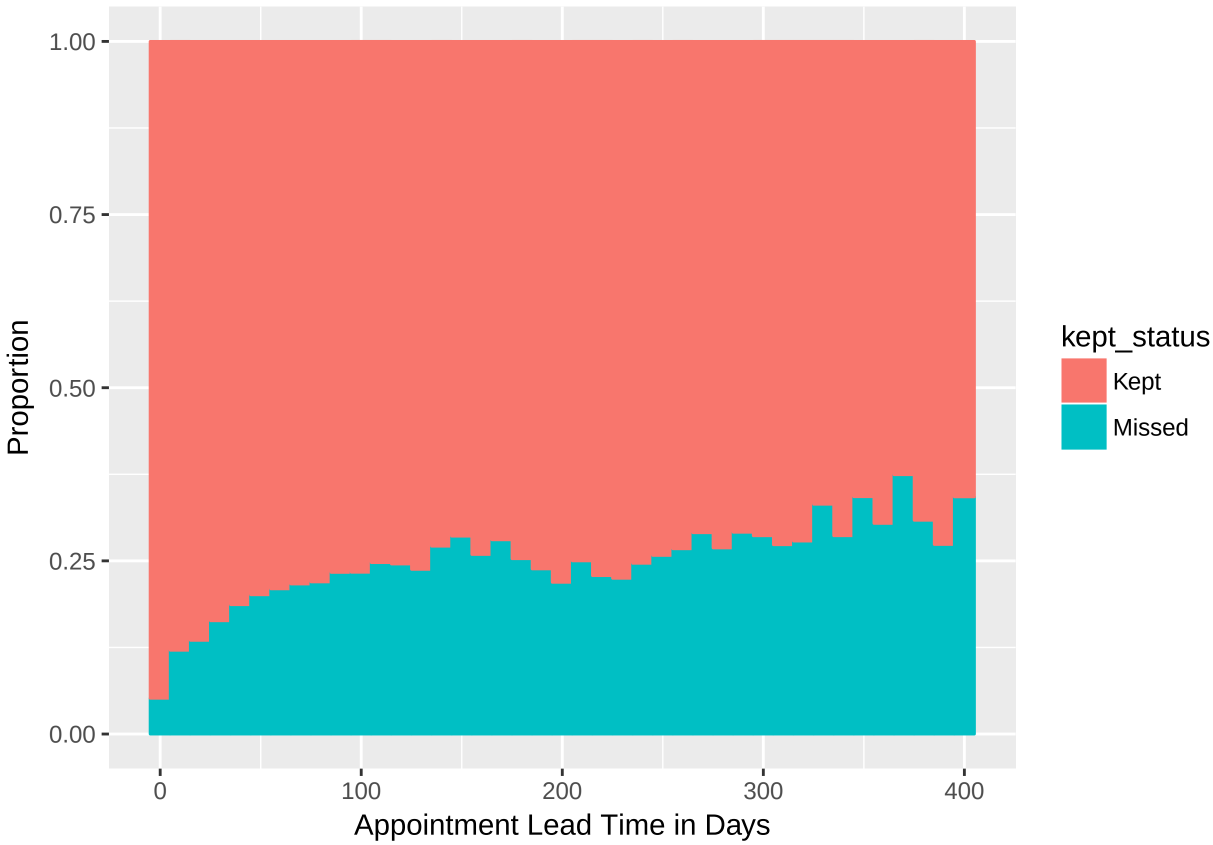
```
appointments$appt_lead_time <- ifelse(
    appointments$appt_lead_time < 0, 0, appointments$appt_lead_time)

appointments %>%
    filter(appt_lead_time >= 0 & appt_lead_time < 400) %>%
    ggplot(
        aes(x = appt_lead_time, color = kept_status, fill = kept_status)
    ) +
```

```
        geom_histogram(binwidth = 10)
```



There is a lower proportion of missed appointments among those with the shortest lead times. This backs up my theory that shorter lead times could indicate more urgent health matters which patients have a higher incentive to keep. Also, there are small bumps in the histogram around 90, 180 and 360 days, which is probably indicitive of regular 3, 6, and 12-month checkups.

```
appointments %>%
    filter(appt_lead_time >= 0 & appt_lead_time < 400) %>%
    ggplot(
        aes(x = appt_lead_time, color = kept_status, fill = kept_status)
    ) +
        geom_histogram(binwidth = 10, position = "fill") +
        labs(x = "Appointment Lead Time in Days", y = "Proportion")
```



Looking at `appt_lead_time` proportionally, the drop in the ratio of missed appointments among those with the shortest lead times is easier to see.

Modeling

Create Modeling Data

I will select the data to be used in modeling and assign to `model_data`, then convert the categorical information to dummy variables to help with the modelling. Since I am converting to dummy variables, I will leave out the dependent variable `kept_status` and add it back once I finish selecting the dummy variables, making sure not to remove any observations or change the order in any way while they are separate.

```
model_data <- appointments %>%
    select(
        appt_length, patient_age, patient_gender, billing_type,
        patient_distance, provider_specialty, remind_call_result, hour,
        prior_percent_missed, appt_lead_time, appt_weekday, is_new_patient,
        weekday_scheduled, county_code)


factor_columns <- c(
    "patient_gender", "billing_type",
    "provider_specialty", "remind_call_result", "hour",
    "appt_weekday", "weekday_scheduled", "is_new_patient",
    "county_code")
model_data[factor_columns] <- map(model_data[factor_columns], factor)
```

```r
dummy_vars <- caret::dummyVars(~ ., data = model_data)

model_data_dummy <- data.frame(predict(dummy_vars, newdata = model_data))
```

The next step is to look for linear combinations, highly correlated variables, and variables with near zero variance. These variables add complexity to the model without providing any significant information, so I will remove them to help the model work better and more efficiently.

First I will look for linear combinations.

```r
linear_combos <- caret::findLinearCombos(model_data_dummy)

colnames(model_data_dummy[, linear_combos$remove])
```

```
## [1] "billing_type.DMAP"             "provider_specialty.Other"
## [3] "remind_call_result.Not.Called" "hour.21"
## [5] "appt_weekday.Friday"           "is_new_patient.1"
## [7] "weekday_scheduled.Friday"      "county_code.X"
```

```r
model_data_dummy <- model_data_dummy[, -linear_combos$remove]
```

Next I will check for highly correlated variables, and create a table that lists the correlated pairs and their column index.

```r
cor_matrix <- cor(model_data_dummy)

high_cor <- as.data.frame(which(abs(cor_matrix) > 0.90, arr.ind = TRUE))

cm_index <- high_cor %>% filter(row != col)

cor_matrix[cm_index[, 1], cm_index[, 2]]
```

```
##                        patient_gender.Female patient_gender.Male
## patient_gender.Male             -0.997632962         1.000000000
## patient_gender.Female            1.000000000        -0.997632962
## provider_specialty.B            -0.004807503         0.004841524
## provider_specialty.A             0.008441594        -0.008524185
##                        provider_specialty.A provider_specialty.B
## patient_gender.Male            -0.008524185          0.004841524
## patient_gender.Female           0.008441594         -0.004807503
## provider_specialty.B           -0.915215490          1.000000000
## provider_specialty.A            1.000000000         -0.915215490
```

```r
cbind(
    cm_index[, 1],
    colnames(model_data_dummy[cm_index[, 1]]),
    cm_index[, 2],
    colnames(model_data_dummy[cm_index[, 2]]))
```

```
##      [,1] [,2]                    [,3] [,4]
## [1,] "4"  "patient_gender.Male"   "3"  "patient_gender.Female"
## [2,] "3"  "patient_gender.Female" "4"  "patient_gender.Male"
## [3,] "10" "provider_specialty.B"  "9"  "provider_specialty.A"
## [4,] "9"  "provider_specialty.A"  "10" "provider_specialty.B"
```

Most cases of `patient_gender` are "Male"" or "Female"", with few values of"Other"" and "Unknown"", leading to a high negative correlation between male and female. Similary, most cases of `provider_specialty` are either"A" or “B”. Due to the dominance of two possible values, there is a high correlation between them and one of each pair will be eliminated. I will eliminate the dummy variables `patient_gender.Male` and `provider_specialty.B` (columns 4 and 10).

```
model_data_dummy <- model_data_dummy[, -c(4, 10)]
```

Finally, I will check for variables with a variance near zero, and remove them. These variables are either all one value, or have very few unique values relative to the sample size, and thus do not provide a significant amount of information.

```
near_zero_var <- caret::nearZeroVar(model_data_dummy)

colnames(model_data_dummy[, near_zero_var])
```

```
##  [1] "patient_gender.Other"
##  [2] "patient_gender.Unknown"
##  [3] "provider_specialty.C"
##  [4] "provider_specialty.D"
##  [5] "remind_call_result.Answered...Canceled"
##  [6] "remind_call_result.Answered...Reschedule"
##  [7] "remind_call_result.Busy"
##  [8] "remind_call_result.No.Answer"
##  [9] "hour.0"
## [10] "hour.5"
## [11] "hour.6"
## [12] "hour.7"
## [13] "hour.12"
## [14] "hour.17"
## [15] "hour.18"
## [16] "hour.19"
## [17] "hour.20"
## [18] "county_code.A"
## [19] "county_code.B"
## [20] "county_code.C"
## [21] "county_code.D"
## [22] "county_code.E"
## [23] "county_code.G"
## [24] "county_code.H"
## [25] "county_code.K"
## [26] "county_code.L"
## [27] "county_code.M"
## [28] "county_code.N"
## [29] "county_code.O"
## [30] "county_code.Q"
## [31] "county_code.R"
## [32] "county_code.S"
## [33] "county_code.T"
## [34] "county_code.V"
## [35] "county_code.W"
```

```
model_data_dummy <- model_data_dummy[, -near_zero_var]
```

Now that the dummy variables are created and removed when necessary, I will add back the dependent variable `kept_status`. There has been no change to the number of rows, and only dummy variable columns have been removed, ensuring that dependent variable values line up with the rest of their respective observations.

```
model_data_dummy <- cbind(appointments[, 1], model_data_dummy)

model_data_dummy$kept_status <- as.factor(model_data_dummy$kept_status)
```

Divide model_data into train, validate, and test sets

The data will be divided into three sets: train, validate, and test. I will use 60% of the tata for training, and 20% each for validation and test.

Because the data is sorted by date, I want to use the most recent data for the test data, the next oldest for validation, and the oldest for training. Since I am trying to predict future appointments, testing on the most recent data will result in the best measure of the model's performance on future appointments.

```
train <- model_data_dummy[1:205660, ]
validate <- model_data_dummy[205661:274200, ]
test <- model_data_dummy[274201:nrow(model_data_dummy), ]
table(train$kept_status)
```

```
##
##   Kept Missed
## 174610  31050
```

```
train_balance_subset <- train[168750:205660, ]
table(train_balance_subset$kept_status)
```

```
##
##   Kept Missed
##  31050   5861
```

```
train_kept <- train_balance_subset[train_balance_subset$kept_status == "Kept", ]
train_missed <- train[train$kept_status == "Missed", ]
train_balanced <- rbind(train_kept, train_missed)
table(train_balanced$kept_status)
```

```
##
##   Kept Missed
##  31050  31050
```

The models chosen to test the missed appointment predictions are glm and random forest, which will both be used with the caret package. The glm model is chosen because this is a binary classification problem, where it is also useful to predict the probabilities of each outcome, and the glm is well suited for these tasks.

Random forest will be used because it will be able to pick up on some non-linearities that the glm can't since it is a linear model. LIke the glm, it also has the ability to predict the class probabilities.

Set Up Model Parameters

```
control <- caret::trainControl(method = "cv", number = 2, classProbs = TRUE)
seed <- 7
metric <- "Accuracy"
set.seed(seed)
mtry <- 3
tunegrid <- expand.grid(.mtry = mtry)
```

Logistic Regression Model

```
glm_model <- caret::train(
    kept_status ~ .,
    data = train_balanced,
    method = "glm",
    trControl = control
)

summary(glm_model)
```

```
##
## Call:
## NULL
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -4.5329  -0.9149  -0.0397   0.9272   3.1002
##
## Coefficients:
##                                           Estimate Std. Error z value
## (Intercept)                             -3.5588693  0.1591281 -22.365
## appt_length                              0.0034860  0.0005925   5.884
## patient_age                             -0.0106897  0.0005068 -21.094
## patient_gender.Female                   -0.0738172  0.0187077  -3.946
## billing_type.Commercial                 -0.0229548  0.0238574  -0.962
## patient_distance                        -0.0001205  0.0001377  -0.876
## provider_specialty.A                     0.4240106  0.0216762  19.561
## remind_call_result.Answered...Confirmed -0.1086364  0.0408456  -2.660
## remind_call_result.Answered...No.Response 0.9752926  0.0298636  32.658
## remind_call_result.Failed                1.5393403  0.0394227  39.047
## remind_call_result.Left.Message         -0.0618027  0.0575144  -1.075
## hour.8                                  -0.8119763  0.0875658  -9.273
## hour.9                                  -0.8578192  0.0876758  -9.784
## hour.10                                 -0.9707282  0.0875451 -11.088
## hour.11                                 -0.8118429  0.0881924  -9.205
## hour.13                                 -0.9326348  0.0874299 -10.667
## hour.14                                 -0.9383090  0.0877224 -10.696
## hour.15                                 -0.9980499  0.0876302 -11.389
## hour.16                                 -0.9174440  0.0882678 -10.394
## prior_percent_missed                     4.6934109  0.0666385  70.431
## appt_lead_time                           0.0039104  0.0001156  33.840
## appt_weekday.Monday                      0.0307168  0.0350072   0.877
## appt_weekday.Tuesday                    -0.2071644  0.0333709  -6.208
## appt_weekday.Wednesday                  -0.3156885  0.0338466  -9.327
## appt_weekday.Thursday                   -0.1626847  0.0344622  -4.721
## is_new_patient.0                         2.1170108  0.1270893  16.658
## weekday_scheduled.Monday                -0.0372543  0.0328691  -1.133
## weekday_scheduled.Tuesday                0.0857184  0.0321358   2.667
## weekday_scheduled.Wednesday              0.0585337  0.0325356   1.799
## weekday_scheduled.Thursday               0.1096276  0.0327157   3.351
## county_code.F                           -0.3880327  0.0364226 -10.654
## county_code.I                            0.2866920  0.0303165   9.457
## county_code.J                            0.0196280  0.0347897   0.564
```

```
## county_code.P                              -0.0115906  0.0254215  -0.456
## county_code.U                               0.3414571  0.0364940   9.357
##                                            Pr(>|z|)
## (Intercept)                                 < 2e-16 ***
## appt_length                                4.01e-09 ***
## patient_age                                 < 2e-16 ***
## patient_gender.Female                      7.95e-05 ***
## billing_type.Commercial                    0.335966
## patient_distance                           0.381242
## provider_specialty.A                        < 2e-16 ***
## remind_call_result.Answered...Confirmed    0.007821 **
## remind_call_result.Answered...No.Response   < 2e-16 ***
## remind_call_result.Failed                   < 2e-16 ***
## remind_call_result.Left.Message            0.282571
## hour.8                                      < 2e-16 ***
## hour.9                                      < 2e-16 ***
## hour.10                                     < 2e-16 ***
## hour.11                                     < 2e-16 ***
## hour.13                                     < 2e-16 ***
## hour.14                                     < 2e-16 ***
## hour.15                                     < 2e-16 ***
## hour.16                                     < 2e-16 ***
## prior_percent_missed                        < 2e-16 ***
## appt_lead_time                              < 2e-16 ***
## appt_weekday.Monday                        0.380247
## appt_weekday.Tuesday                       5.37e-10 ***
## appt_weekday.Wednesday                      < 2e-16 ***
## appt_weekday.Thursday                      2.35e-06 ***
## is_new_patient.0                            < 2e-16 ***
## weekday_scheduled.Monday                   0.257040
## weekday_scheduled.Tuesday                  0.007645 **
## weekday_scheduled.Wednesday                0.072009 .
## weekday_scheduled.Thursday                 0.000805 ***
## county_code.F                               < 2e-16 ***
## county_code.I                               < 2e-16 ***
## county_code.J                              0.572624
## county_code.P                              0.648434
## county_code.U                               < 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: 86089  on 62099  degrees of freedom
## Residual deviance: 68705  on 62065  degrees of freedom
## AIC: 68775
##
## Number of Fisher Scoring iterations: 6
```

Random Forest Model

```r
rf_model <- caret::train(
    kept_status ~ ., data = train_balanced, method = "rf", metric = metric,
```

```
    tuneGrid = tunegrid, trControl = control)
```

Model Selection

To select the best model, I will first make predictions for each model on the validation set using the `predict` function in caret. This will allow me to have caret produce a confusion matrix and calculate the F1 score. The F1 score, which is a harmonic average of precision and recall, will be used to select the best model. A high recall is desired because when the actual appointment is missed, we want the model to predict it, while a high precision is desired because when the model predicts missed, we want the actual appointment to be missed.

```
pred_glm <- predict(glm_model, validate)

conf_mat_glm <- caret::confusionMatrix(
    pred_glm, validate$kept_status, positive = "Missed")

conf_mat_glm
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  Kept Missed
##     Kept   41656   3329
##     Missed 15428   8127
##
##                Accuracy : 0.7263
##                  95% CI : (0.723, 0.7297)
##     No Information Rate : 0.8329
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.3088
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.7094
##             Specificity : 0.7297
##          Pos Pred Value : 0.3450
##          Neg Pred Value : 0.9260
##              Prevalence : 0.1671
##          Detection Rate : 0.1186
##    Detection Prevalence : 0.3437
##       Balanced Accuracy : 0.7196
##
##        'Positive' Class : Missed
##
```

```
conf_mat_glm$byClass["F1"]
```

```
##        F1
## 0.4642541
```

```
pred_rf <- predict(rf_model, validate)

conf_mat_rf <- caret::confusionMatrix(
    pred_rf, validate$kept_status, positive = "Missed")
```

```r
conf_mat_rf
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  Kept Missed
##     Kept   40187   2442
##     Missed 16897   9014
##
##                Accuracy : 0.7178
##                  95% CI : (0.7145, 0.7212)
##     No Information Rate : 0.8329
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.3263
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.7868
##             Specificity : 0.7040
##          Pos Pred Value : 0.3479
##          Neg Pred Value : 0.9427
##              Prevalence : 0.1671
##          Detection Rate : 0.1315
##    Detection Prevalence : 0.3780
##       Balanced Accuracy : 0.7454
##
##        'Positive' Class : Missed
##
```

```r
conf_mat_rf$byClass["F1"]
```

```
##        F1
## 0.4824578
```

Looking at the F1 scores based on the model predictions on the validation set, the random forest model outperforms the glm model, with a score of 0.4824578 compared to the glm model with a score of 0.4642541. The difference is fairly small and likely indicates that there are some non-linearities that the random forest is better able to pick up. Because of this advantage, random forst will be the chosen model. The next step is to evaluate the expected performance on new data.

Expected Model Performance

Now that the random forest model has been selected, I will see how well it performs on the test data. Since the test data is the most recent, the performance on this data should be the best representation of how well the model will perform on future unseen information.

```r
final_pred_rf <- predict(rf_model, test)

final_conf_mat_rf <- caret::confusionMatrix(
    final_pred_rf, test$kept_status, positive = "Missed")

final_conf_mat_rf
```

```
## Confusion Matrix and Statistics
##
```

```
##           Reference
## Prediction  Kept Missed
##     Kept   41596   2671
##     Missed 14838   9415
##
##                Accuracy : 0.7445
##                  95% CI : (0.7412, 0.7477)
##     No Information Rate : 0.8236
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.3698
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.7790
##             Specificity : 0.7371
##          Pos Pred Value : 0.3882
##          Neg Pred Value : 0.9397
##              Prevalence : 0.1764
##          Detection Rate : 0.1374
##    Detection Prevalence : 0.3540
##       Balanced Accuracy : 0.7580
##
##        'Positive' Class : Missed
##
```

```r
final_conf_mat_rf$byClass["F1"]
```

```
##        F1
## 0.5181761
```

The F1 score is 0.4824578 on the test set, which is higher than it was based on the validation set. This is a fairly significant improvement over the performance on the validation set. I'm not sure why it performs better on the test data, but my speculation is there is some time dependence that works in favor of the test set.

The positive predictive value of 0.3881994 means that when the model predicts a miss, it is expcted to be correct 38.82 % of the time and incorrect 61.18 % of the time. Predictions of "Miss" are expected to be incorrect more often than they are expected to be correct. While this is a little disappointing, it helps to look at the improvement over having no model at all, and it also helps to remember that the misses the model is trying to predict can be caused by manny things that the data doesn't directly capture. With no model, predicting an appointment as missed is expected to be correct approximately 15.93 % of the time, based on the overall rate of missed appointments. 61.18 % incorrect is a significant improvement over 84.07 % incorrect.

Recommendations

There are several additional steps that can be taken to help with the missed appointment predictions, such as additional features and further analysis of the results.

One feature I suspect would be a good predictor is weather. For example, a snowy day might lead to more missed appointments because transportation becomes more difficult. While weather was not included in the original data, there is widely available historical weather data that could be added in. Since the data covers multiple geographic locations, there would ideally be weather data for each location matched to the appointment date.

Further analysis of the model's higher performance on the test data (most recent) than the validation data (next most recent) would also be useful. If the difference is due to the time dependence working in favor of the test data, this can be tested by using different methods to split the data such as rolling windows in the caret package.

There are also some other insights that can be used from the exploratory data analysis. For example, looking at the percent of prior appointments missed revealed that patients who missed 10% or less of their previous appountments overwhelmingly keep their appintments.

Class Probabilities

The model can calculate the class probabilities in addition to the predicted class. The class probabilities can be much more valuable when implementing the predictions into the appointment booking system than just the classes alone because the combined probabilities can be used in a particular time slot. Below is the probability prediction on the first few rows of the test data.

```
rf_pred_probs <- predict(rf_model, newdata = test, type = "prob")
rf_probs <- cbind(test$kept_status, rf_pred_probs)
head(rf_probs)
```

```
##        test$kept_status  Kept Missed
## 274201             Kept 0.350  0.650
## 274202             Kept 0.702  0.298
## 274203             Kept 0.040  0.960
## 274204             Kept 0.940  0.060
## 274205             Kept 0.984  0.016
## 274206           Missed 0.114  0.886
```

To illustrate the potential use for the class probabilities, let's say there was an office that could handle two patients in a time slot which had the following two appointments booked: one with a 0.60 chance of being kept, and one with a 0.70 chance of being kept.

Based on classification alone, it looks like they will both be kept and it would be hard to justify overbooking. Combining the probabilities, it is easy to calculate the expected probability of having neither, 1, or both patients show up in the time slot:

Both: 0.60 x 0.70 = 0.42

1 patient: (0.60 x (1-0.70)) + ((1 - 0.60) x 0.70) = 0.46

Neither: 0.40 x 0.30 = 0.12

With this information, it might be more tempting to overbook by one, since there is less than 50% chance that both appointments are going to show up. This is a simplified example that is just meant to demonstrate the general idea of how the class probabilities can be used. The implementation up to the client and must consider the cost of a missed appointments (lost revenue) and the cost of having too many patients (frustrated patients and staff).

Conclusion

Predicting missed appointments is challenging because there are many reasons an appointment can be missed that can't be known in advance of future appointments due to their random nature, such as having a flat tire or getting called in to a last-minute work meeting.

Despite the challenges, indirect information about the patient and the appointment can still help predict whether an appointment will be missed. While these predictions have an inherent level of error, they still provide useful information.

With a perfect prediction, a medical provider could completely make up for the lost revenue due to missed appointments by overbooking by just the right amount, without resulting in times the overbooking leads to having more patients than desired. Due to the inherent prediction error, there will still be times that the overbooking is too much or too little, meaning there will still times where there are more or fewer patients than desired. However, since the model predictions are better than predicting without a model, the model allows a medical provider to over-book more intelligently, which will allow more lost revenue to be made up for with over-booking with fewer negative consequences.