Austin Animal Shelter Data - Occasion

```
## Parsed with column specification:
## cols(
##   .default = col_character(),
##   monthyear = col_datetime(format = ""),
##   count = col_double(),
##   Periods = col_double(),
##   `Period Range` = col_double(),
##   `outcome_age_(days)` = col_double(),
##   `outcome_age_(years)` = col_double(),
##   dob_year = col_double(),
##   dob_month = col_double(),
##   outcome_month = col_double(),
##   outcome_year = col_double(),
##   outcome_hour = col_double(),
##   cfa_breed = col_logical(),
##   domestic_breed = col_logical()
## )
## See spec(...) for full column specifications.
## Observations: 29,421
## Variables: 37
## $ age_upon_outcome      <chr> "2 weeks", "1 month", "3 months", "1 year…
## $ animal_id             <chr> "A684346", "A685067", "A678580", "A675405…
## $ animal_type           <chr> "Cat", "Cat", "Cat", "Cat", "Cat", "Cat",…
## $ breed                 <chr> "domestic shorthair", "domestic shorthair…
## $ color                 <chr> "orange", "blue /white", "white/black", "…
## $ date_of_birth         <chr> "7/7/2014 0:00", "6/16/2014 0:00", "3/26/…
## $ datetime              <chr> "7/22/2014 16:04", "8/14/2014 18:45", "6/…
## $ monthyear             <dttm> 2014-07-22 16:04:00, 2014-08-14 18:45:00…
## $ name                  <chr> NA, "Lucy", "*Frida", "Stella Luna", NA, …
## $ outcome_subtype       <chr> "Partner", NA, "Offsite", NA, "Partner", …
## $ outcome_type          <chr> "Transfer", "Adoption", "Adoption", "Retu…
## $ sex_upon_outcome      <chr> "Intact Male", "Intact Female", "Spayed F…
## $ count                 <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,…
## $ sex                   <chr> "Male", "Female", "Female", "Female", "Ma…
## $ `Spay Neuter`         <chr> "No", "No", "Yes", "Yes", "No", "Yes", "Y…
## $ Periods               <dbl> 2, 1, 3, 1, 3, 2, 8, 5, 2, 1, 2, 2, 2, 3,…
## $ `Period Range`        <dbl> 7, 30, 30, 365, 7, 30, 30, 30, 30, 365, 3…
## $ `outcome_age_(days)`  <dbl> 14, 30, 90, 365, 21, 60, 240, 150, 60, 36…
## $ `outcome_age_(years)` <dbl> 0.03835616, 0.08219178, 0.24657534, 1.000…
## $ `Cat/Kitten (outcome)` <chr> "Kitten", "Kitten", "Kitten", "Cat", "Kit…
## $ sex_age_outcome        <chr> "Intact Male Kitten", "Intact Female Kitt…
## $ age_group              <chr> "(-0.022, 2.2]", "(-0.022, 2.2]", "(-0.02…
## $ dob_year               <dbl> 2014, 2014, 2014, 2013, 2013, 2014, 2013,…
## $ dob_month              <dbl> 7, 6, 3, 3, 12, 6, 7, 3, 8, 12, 4, 5, 4, …
## $ dob_monthyear          <chr> "2014-07", "2014-08", "2014-06", "2014-03…
## $ outcome_month          <dbl> 7, 8, 6, 3, 1, 8, 3, 8, 10, 12, 7, 6, 7, …
## $ outcome_year           <dbl> 2014, 2014, 2014, 2014, 2014, 2014, 2014,…
## $ outcome_weekday        <chr> "Tuesday", "Thursday", "Sunday", "Friday"…
## $ outcome_hour           <dbl> 16, 18, 17, 14, 19, 15, 14, 15, 18, 13, 1…
## $ breed1                 <chr> "domestic shorthair", "domestic shorthair…
## $ breed2                 <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, N…
## $ cfa_breed              <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE,…
## $ domestic_breed         <lgl> TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE,…
## $ coat_pattern           <chr> "tabby", "tabby", NA, NA, NA, "tabby", "t…
## $ color1                 <chr> "orange", "blue", "white", "black", "blac…
## $ color2                 <chr> NA, "white", "black", "white", "white", N…
## $ coat                   <chr> "orange", "blue", "white", "black", "blac…
## [1] 29421
## [1] 37
```

The effects of Occasion

During what time of the year are adoption rates the highest? When are they the lowest? Do these concide with specific months, holidays, or seasons?

Finding adoption trends with the respect to the time period could potentially serve as valuable data for animal shelters who need to plan adoption events. Should these events be

- Planned around specific seasons: Do more adoptions take place in the Winter, Fall, Summer,or Spring?
- Organized in certain months: Does the holiday atmosphere of November and December have an effect on adoptions?
- Held on a specific weekday: What day of the week is the most common day for adoption? Do most adoptions occur during the weekend, or the weekday?

The `lubridate` package may once again be very useful as it can parse the variables into a date centric format, but the current `outcome_month`, `outcome_hour` and `outcome_weekday` variables are formatted nicely for filtering and grouping purposes.

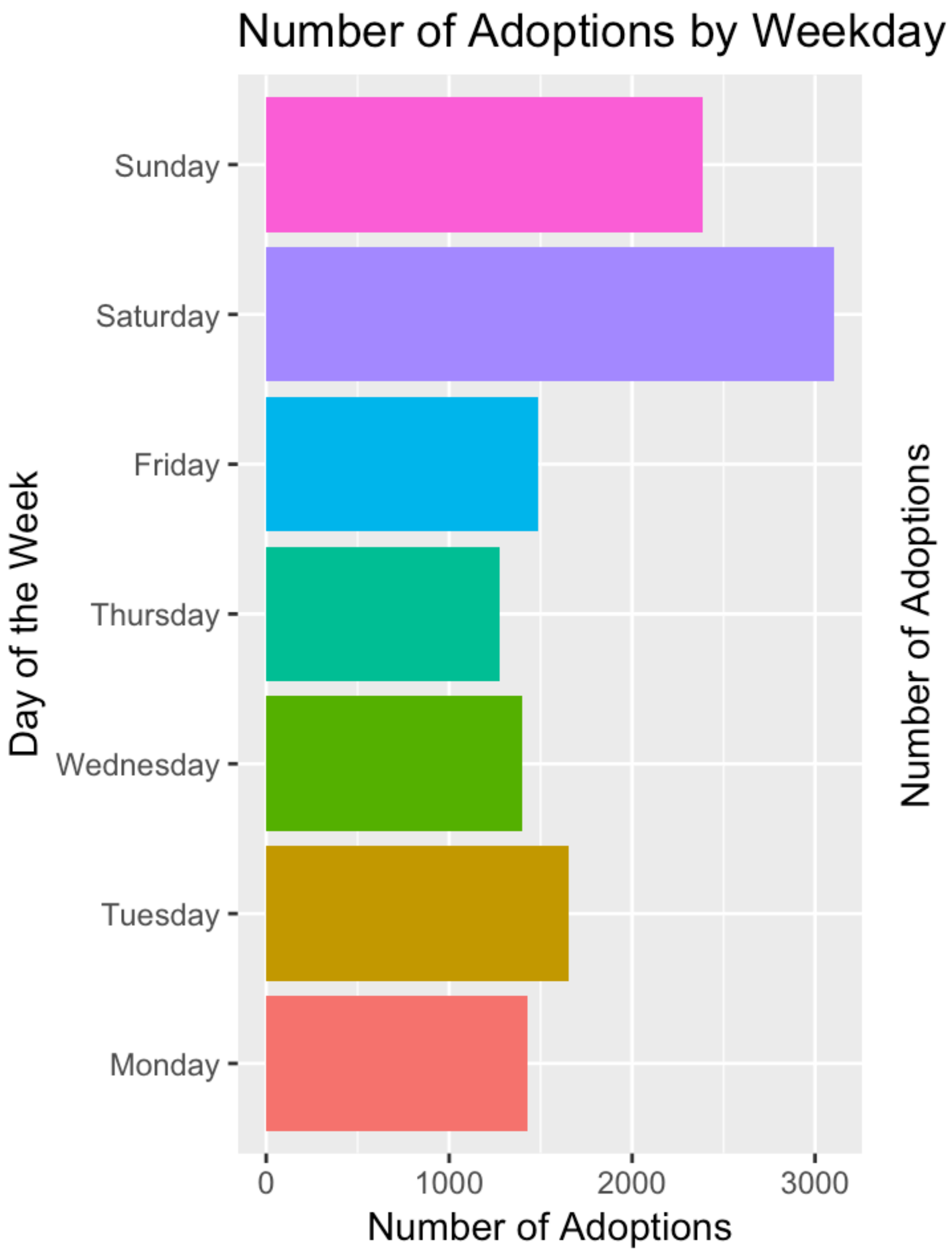
As an example, here is the data for all the adopted cats being grouped by the day. The number of adoptions by day can then be tallied using the `group_by` and `summary` dyplr functions.

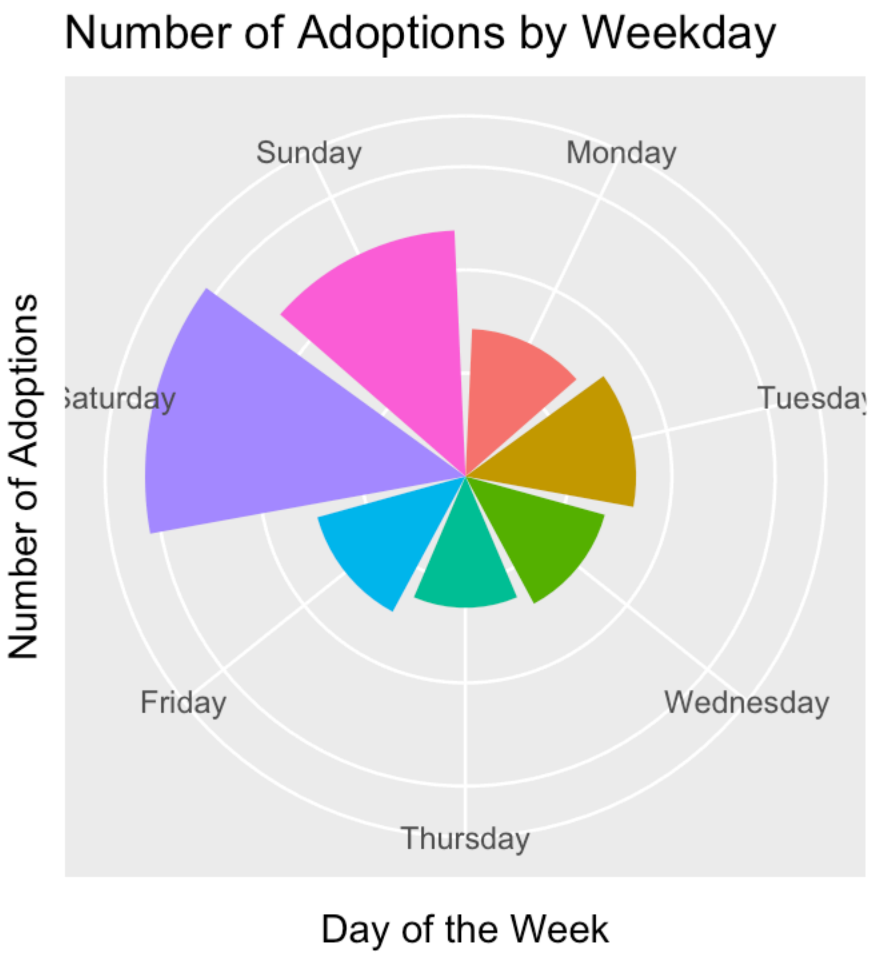
```
## # A tibble: 7 x 2
##   outcome_weekday numAdoptions
##   <chr>                  <int>
## 1 Friday                  1489
## 2 Monday                  1429
## 3 Saturday                3104
## 4 Sunday                  2384
## 5 Thursday                1273
## 6 Tuesday                 1653
## 7 Wednesday               1400
```

The data can the be used for further analyzation, such as determining if the adoption rates by weekday are signifcantly different using a goodness of fit test.

```
## # A tibble: 7 x 2
##   outcome_weekday numAdoptions
##   <fct>                  <int>
## 1 Monday                  1429
## 2 Tuesday                 1653
## 3 Wednesday               1400
## 4 Thursday                1273
## 5 Friday                  1489
## 6 Saturday                3104
## 7 Sunday                  2384
```

An initial visualization based on the questions we asked, is plotting the number of adoptions by weekday.





We can see from the graphs that Saturday and Sunday have the highest number of adoptions. We can explore this further, and conduct statistical tests to see how much of a difference there is between the weekdays.

We start by setting up our data for EDA based on occasion, by selecting the related occasion variables, and using feature engineering to find new variables that would be helpful in initial modeling for adoption. We will conduct EDA on the set up data to see what interesting points we can learn from the data, where it leads and what questions we can come up with.

Setting up new datasets for EDA:

Here, we are feature engineering a season variable, to better look at how seasons affect adoptions.

```
# mutates in outcome date per month
cats_fixed <- cats_select %>%
  mutate(outcome_date = format(as.Date(cats$datetime,format="%m/%d/%Y"), "%d"))

# adds in season column to cats_fixed
yq <- as.yearqtr(as.yearmon(cats_fixed$datetime, "%m/%d/%Y") + 1/12) # use zoo
fn as.yearmon
cats_fixed$Season <- factor(format(yq, "%q"), levels = 1:4,
                labels = c("Winter", "Spring", "Summer", "Fall"))
# output
cats_fixed
```

```
## # A tibble: 29,421 x 9
##    datetime outcome_subtype outcome_type outcome_month outcome_year
##    <chr>    <chr>           <chr>                <dbl>        <dbl>
##  1 7/22/20… Partner         Transfer                 7         2014
##  2 8/14/20… <NA>            Adoption                 8         2014
##  3 6/29/20… Offsite         Adoption                 6         2014
##  4 3/28/20… <NA>            Return to O…             3         2014
##  5 1/9/201… Partner         Transfer                 1         2014
##  6 8/13/20… <NA>            Adoption                 8         2014
##  7 3/6/201… SCRP            Transfer                 3         2014
##  8 8/31/20… <NA>            Adoption                 8         2014
##  9 10/31/2… Foster          Adoption                10         2014
## 10 12/16/2… Partner         Transfer                12         2013
## # … with 29,411 more rows, and 4 more variables: outcome_weekday <chr>,
## #   outcome_hour <dbl>, outcome_date <chr>, Season <fct>
```

From what we know, events are a great way of bringing in more adopters and spreading awareness for shelter activities. Doing some research online, I brought in outside data for events held throughout the year. I can partition the events to only include the relevant years for our data. The data can be found here: https://do512.com/venues/austin-animal-center/past_events (https://do512.com/venues/austin-animal-center/past_events)

```
# create table of events
events <- tribble(
  ~event, ~event_date, ~event_month, ~event_day,  # format of tibble
  #--|--|--/----
  "Hot Summer, cool seniors adoption event", "8/24/2013", 8, as.character(24),
  "Act of Kindness with Princess B Unique", "1/18/2014", 1, as.character(18),
  "Service project at Austin animal shelter","8/9/2014", 8, as.character(9),
  "Austin Pittie Limits", "9/27/2014", 9, as.character(27)
)
# output
events
```

```
## # A tibble: 4 x 4
##   event                                    event_date event_month event_day
##   <chr>                                    <chr>            <dbl> <chr>
## 1 Hot Summer, cool seniors adoption event  8/24/2013            8 24
## 2 Act of Kindness with Princess B Unique   1/18/2014            1 18
## 3 Service project at Austin animal shelter 8/9/2014             8 9
## 4 Austin Pittie Limits                     9/27/2014            9 27
```

```
# add season column for events
yq <- as.yearqtr(as.yearmon(events$event_date, "%m/%d/%Y") + 1/12)
events$event_Season <- factor(format(yq, "%q"), levels = 1:4,
                labels = c("Winter", "Spring", "Summer", "Fall"))
# output
events
```

```
## # A tibble: 4 x 5
##   event                          event_date event_month event_day event_Season
##   <chr>                          <chr>            <dbl> <chr>     <fct>
## 1 Hot Summer, cool seniors a… 8/24/2013            8 24        Summer
## 2 Act of Kindness with Princ… 1/18/2014            1 18        Winter
## 3 Service project at Austin … 8/9/2014             8 9         Summer
## 4 Austin Pittie Limits        9/27/2014            9 27        Fall
```

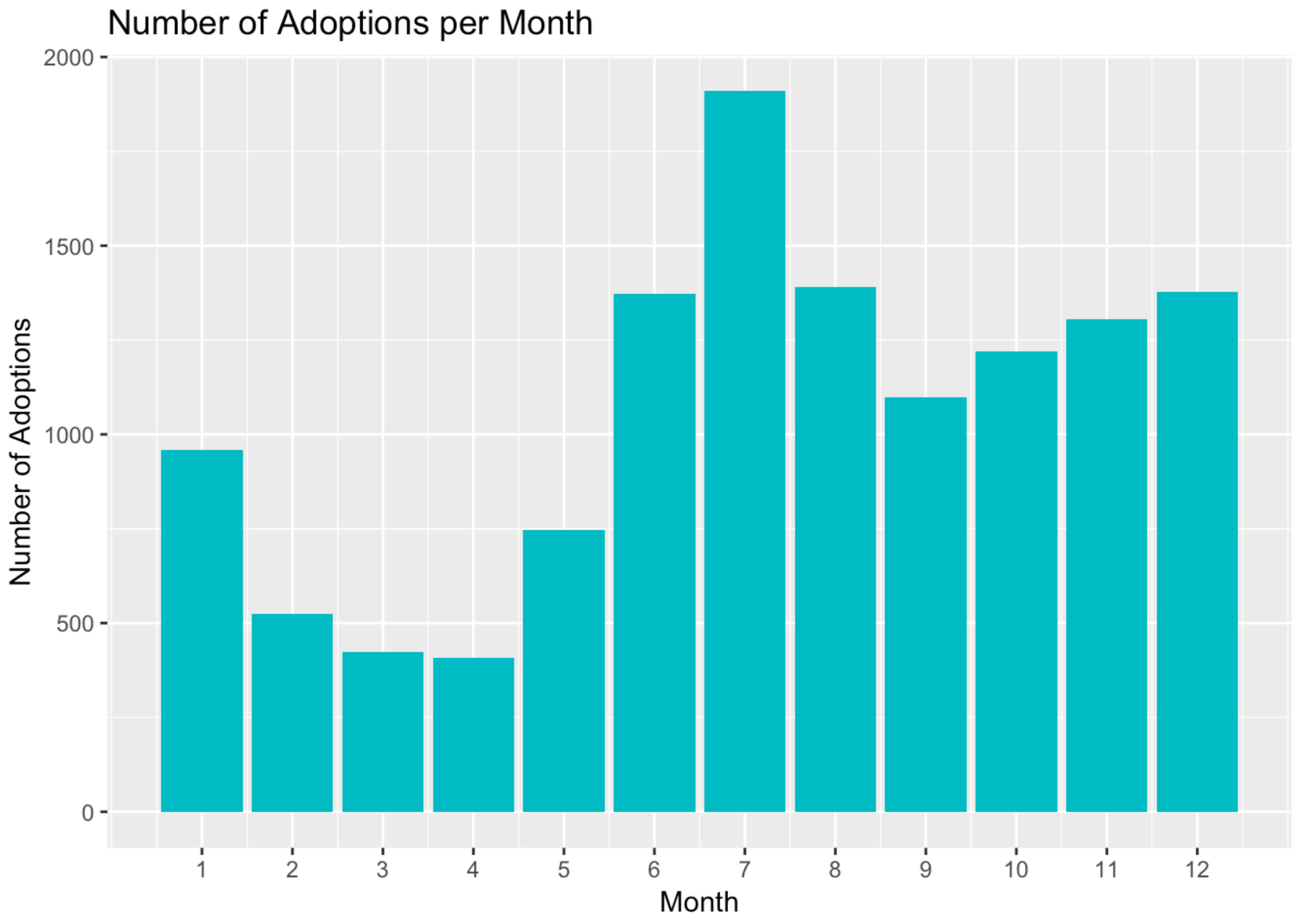
I have created a new table with the event name, date, month, and season. These are the four events that were relevant to the data. For 2014, an event was held throughout each season, except for spring. Next, we will join the outside events data with our main cats dataset.

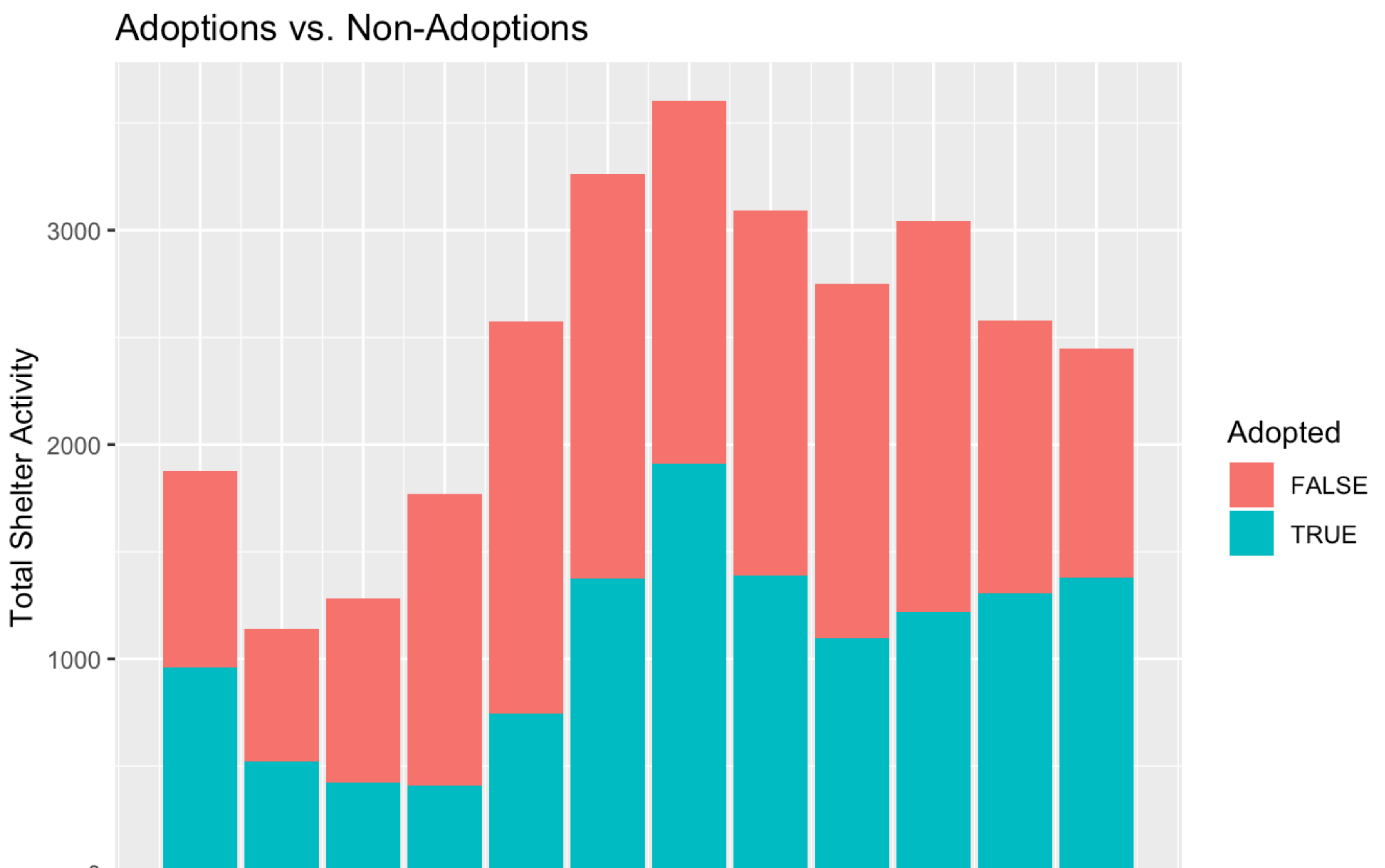
```
# add in bool column for if a date has an even
cats_fixed <- cats_fixed %>%
  mutate(has_event = !(is.na(event)))
# turn into 0/1 value
cats_fixed$has_event <- as.integer(as.logical(cats_fixed$has_event))
# output
#cats_fixed
```

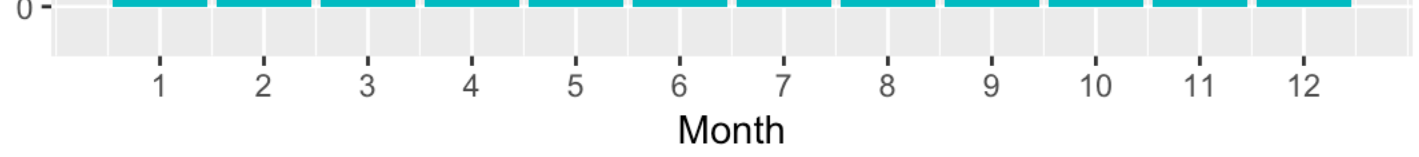
```
## # A tibble: 121 x 13
##    datetime outcome_subtype outcome_type outcome_month outcome_year
##    <chr>    <chr>           <chr>                <dbl>        <dbl>
##  1 8/24/20… SCRP            Transfer                 8         2014
##  2 8/24/20… Partner         Transfer                 8         2014
##  3 8/24/20… Partner         Transfer                 8         2014
##  4 8/24/20… SCRP            Transfer                 8         2014
##  5 8/24/20… Partner         Transfer                 8         2014
##  6 8/24/20… Partner         Transfer                 8         2014
##  7 8/24/20… Partner         Transfer                 8         2014
##  8 8/24/20… <NA>            Adoption                 8         2014
##  9 8/24/20… Partner         Transfer                 8         2014
## 10 8/24/20… <NA>            Return to O…             8         2014
## # … with 111 more rows, and 8 more variables: outcome_weekday <chr>,
## #   outcome_hour <dbl>, outcome_date <chr>, Season <fct>, event <chr>,
## #   event_date <chr>, event_Season <fct>, has_event <int>
```

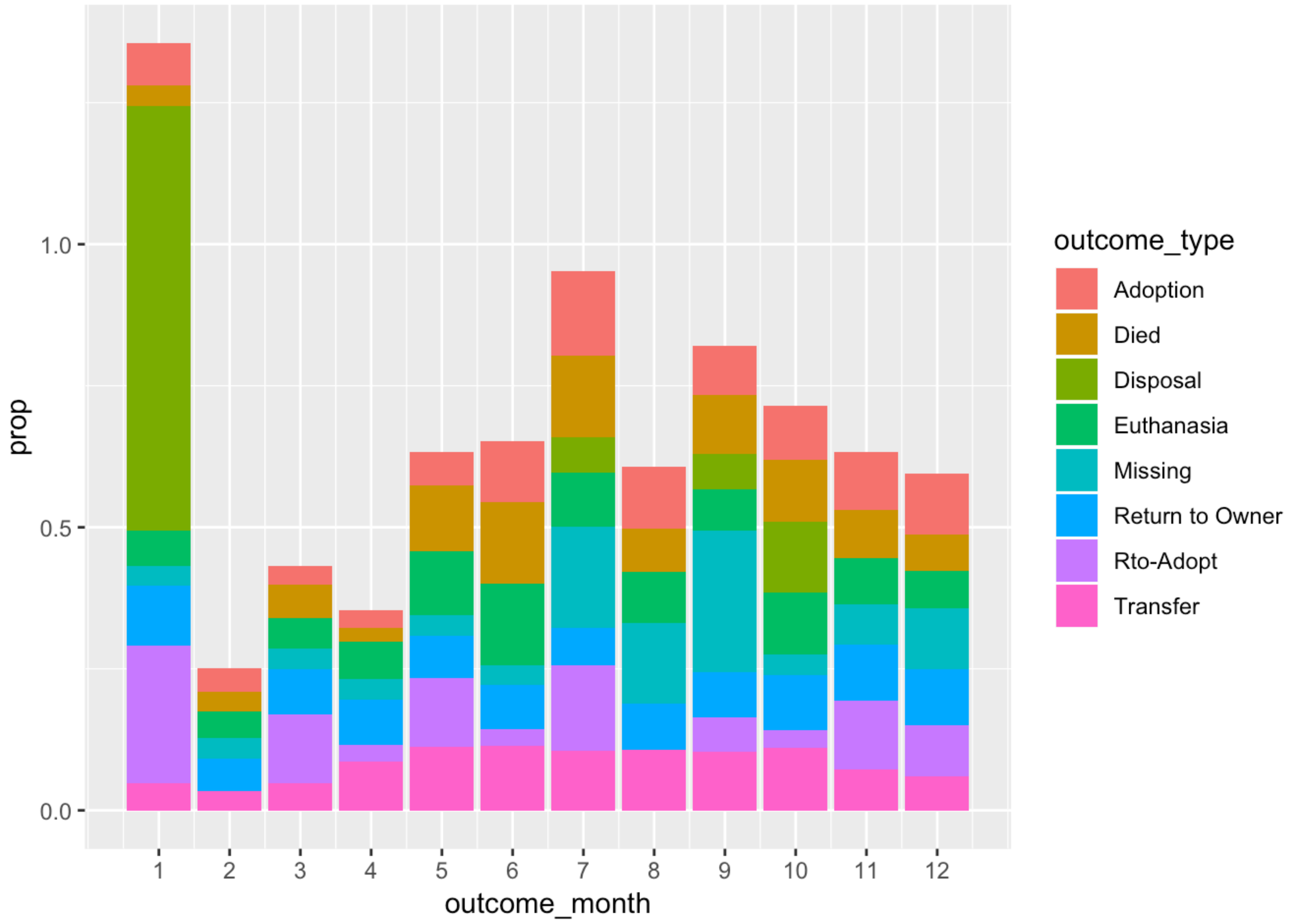
Now that we have joined our outside data, cleaned and transformed our data by filtering out 3 NA values and creating a binary column for checking whether a date had an event, we are now ready to start EDA.

Initial EDA on number of adoptions throughout the year:





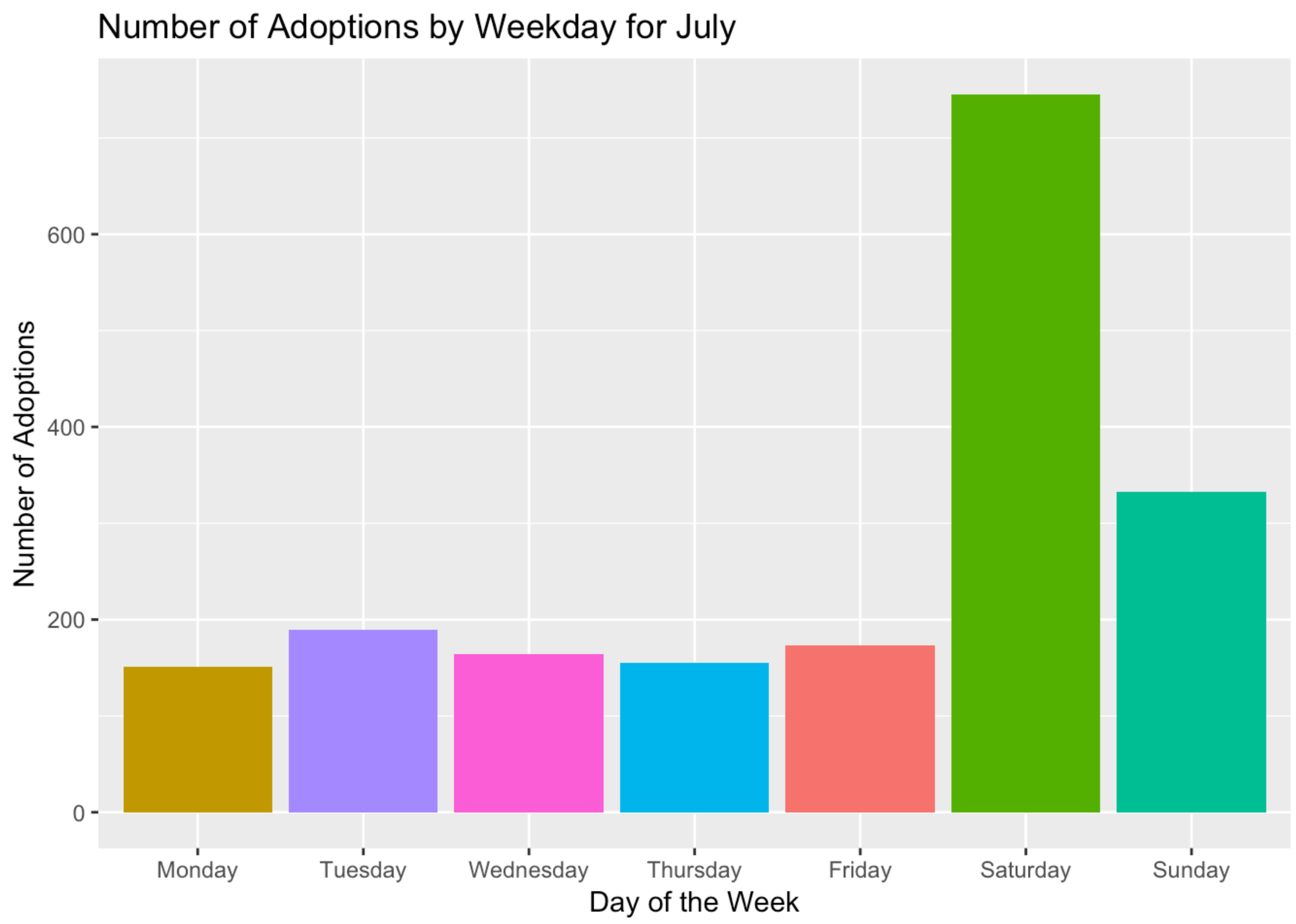




What variables attribute to whether or not a cat is adopted? We can build a regression model and infer the significance of the variables.

Based on the plots, we see that adoptions make up a good amount of total shelter activity. We also see that the data shows a large spike in adoption numbers for the second half of the year. It’s very noticable that July has the highest number of adoptions for the year. We can continue EDA by focusing on July.

More exploration on July:



We see that Saturday has by far the highest number of adoptions. We recognize these patterns from the previous graphs. We can check these differences with statistical testing.

Conduct goodness of fit test to check whether proportions are equal. Assumptions are satisfied since there is a large sample size, and the variables in question are categorical.

```
## # A tibble: 7 x 2
##   outcome_weekday count
##   <chr>           <int>
## 1 Monday           3941
## 2 Tuesday          4473
## 3 Wednesday        3833
## 4 Thursday         3618
## 5 Friday           3934
## 6 Saturday         5151
## 7 Sunday           4468
```

```
## [1] 4202.571 4202.571 4202.571 4202.571 4202.571 4202.571 4202.571
## [1] 3941 4473 3833 3618 3934 5151 4468
```

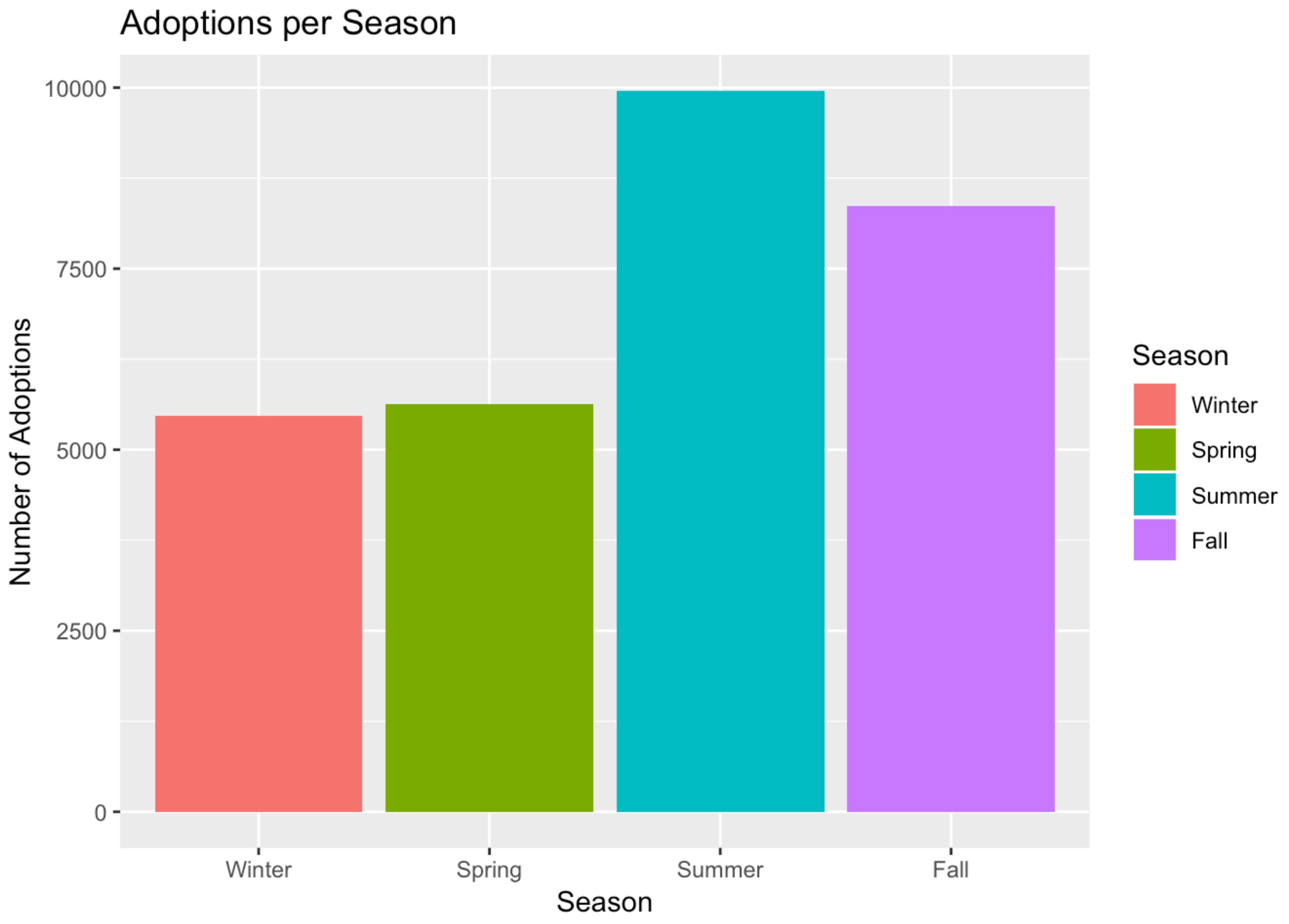
```
## [1] -4.358190  4.505764 -6.157639 -9.739876 -4.474821 15.802306  4.422456
```

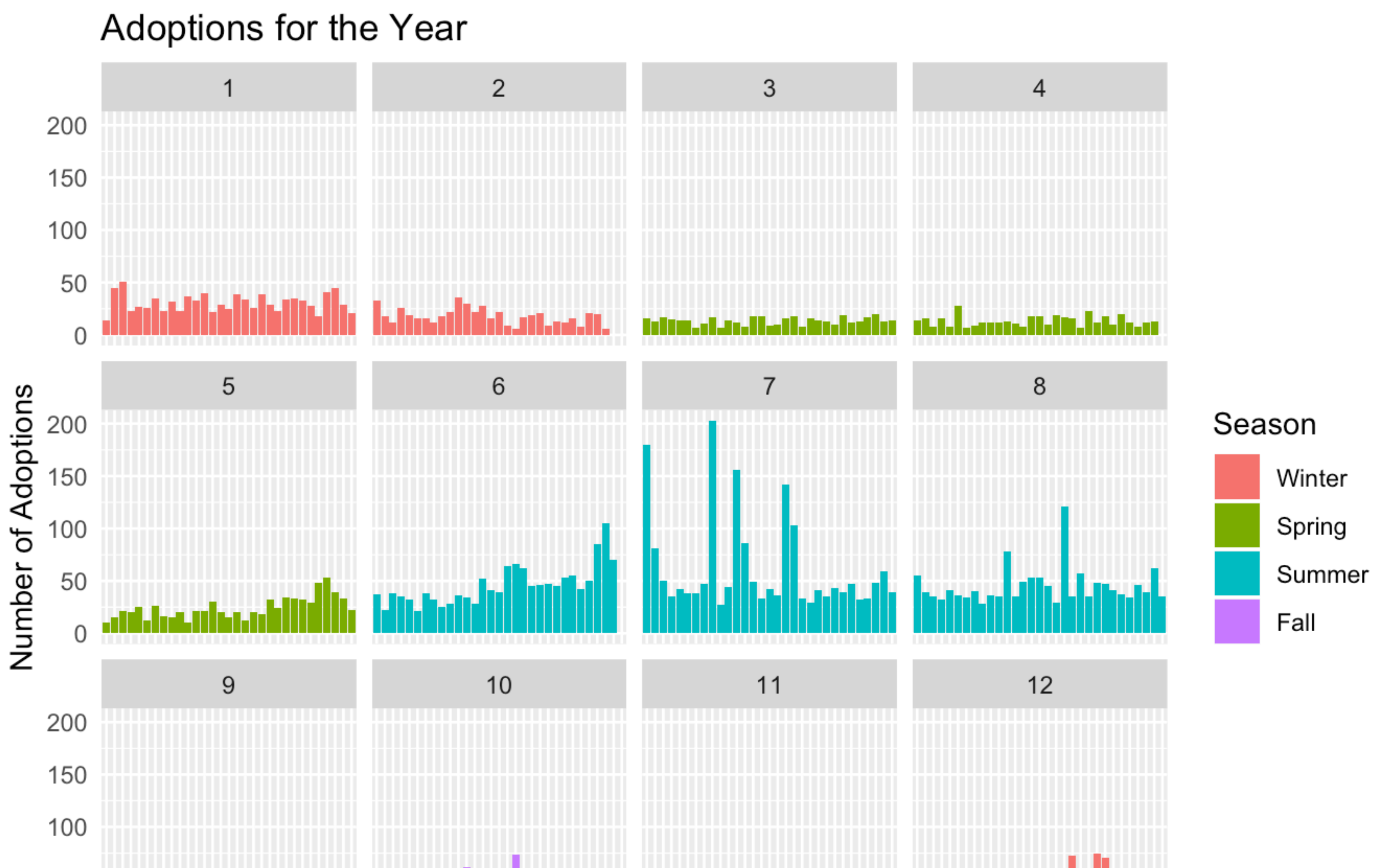
```
##
##  Chi-squared test for given probabilities
##
## data:  weekdays_test$count
## X-squared = 395.46, df = 6, p-value < 2.2e-16
```

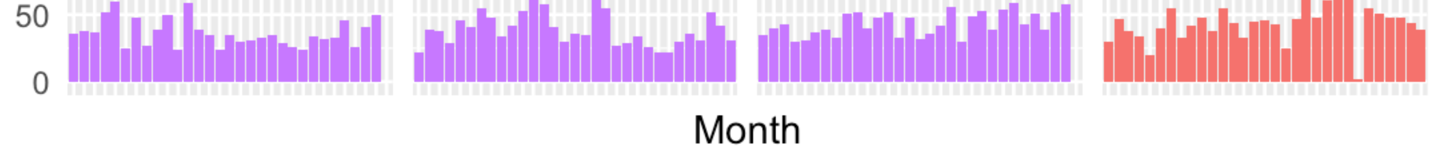
Based on the results of the Goodness of Fit test, Saturday is highly above average, and Sunday and Tuesday are above average. Monday, Wednesday, and Friday are below average, with Thursday very below average. This falls in line with our graphs, and we can see so far that weekdays play an important role in adoption numbers.

Next, we shift our focus to looking at adoptions for the entire year, starting with seasons.

General look at adoption numbers for the whole year.







We see from the first graph that Summer and Fall are the busiest seasons. Looking at the second graph, it shows again that the second half of the year has overall higher adoptions than the first half of the year.

Now let’s look at averages for shelter adoptions for the year.

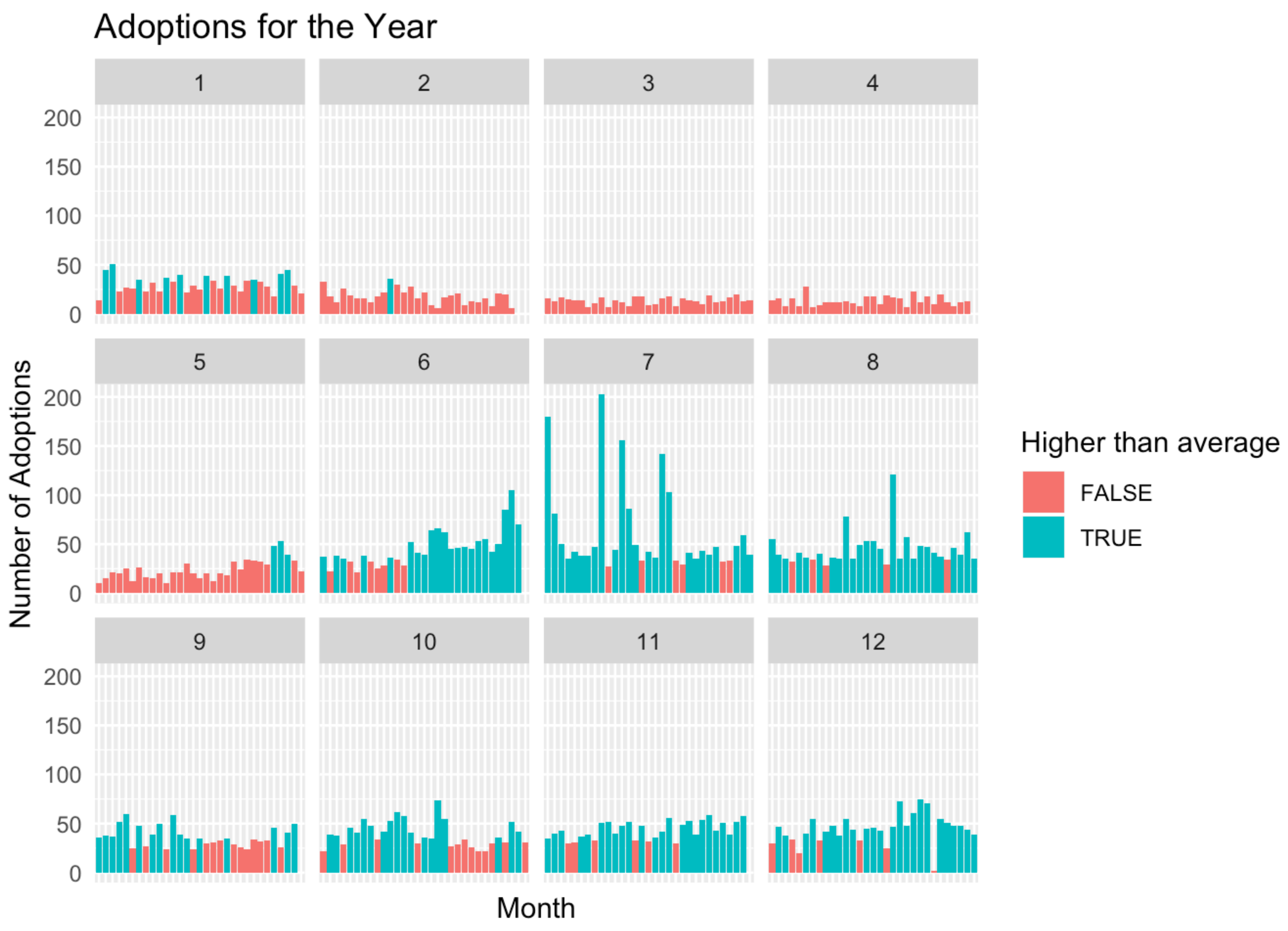
EDA on averages for adoptions:

```
## [1] 1061
```

```
## [1] 34.88219
```

The data shows that the average number of adoptions per month is 1,061, and the average number of adoptions per day is about 35. We can plot the adoptions average per day against the total number of adoptions for the day, to see the distribution of above average days and below average days.

Graphing averages vs daily adoptions:

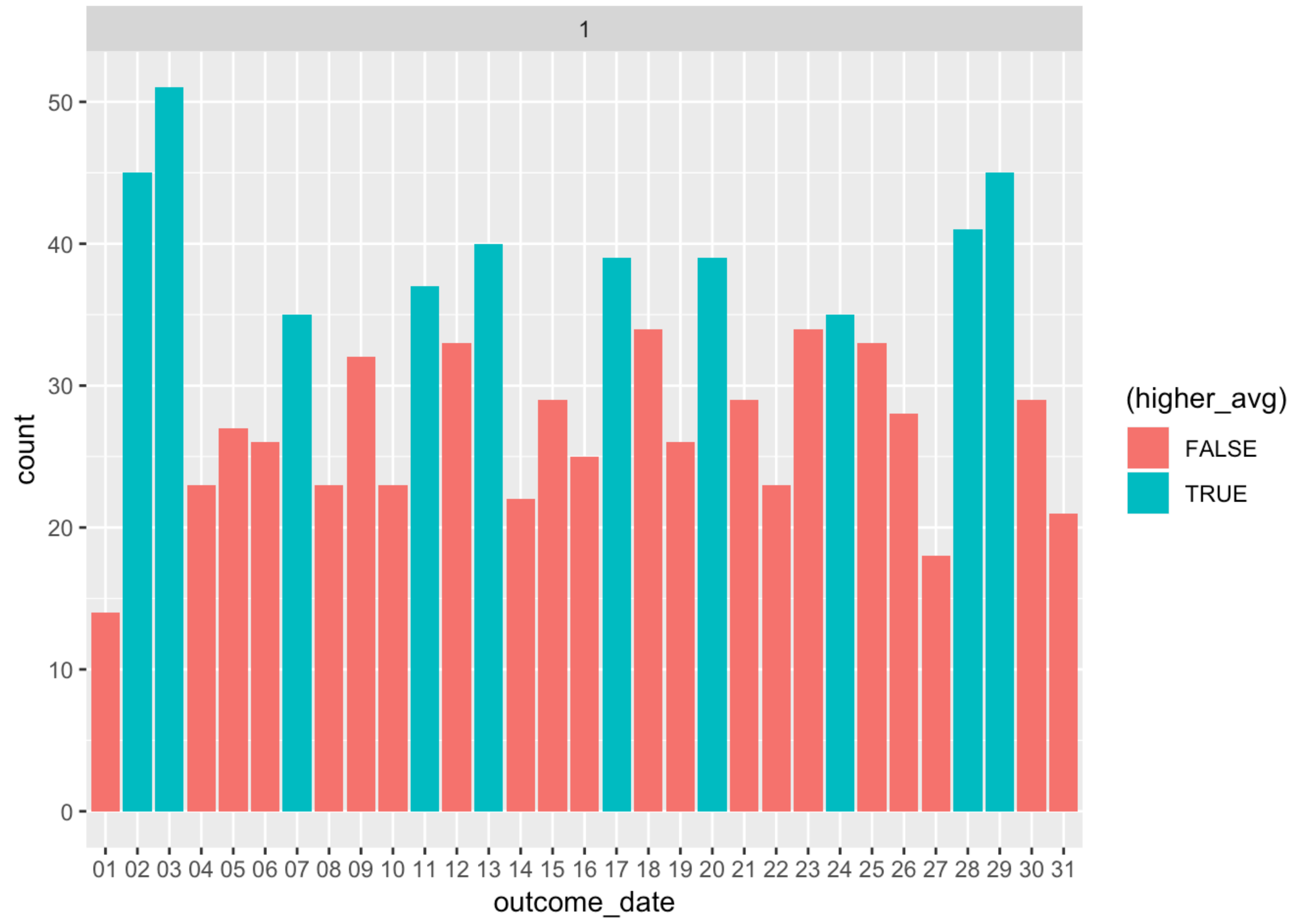


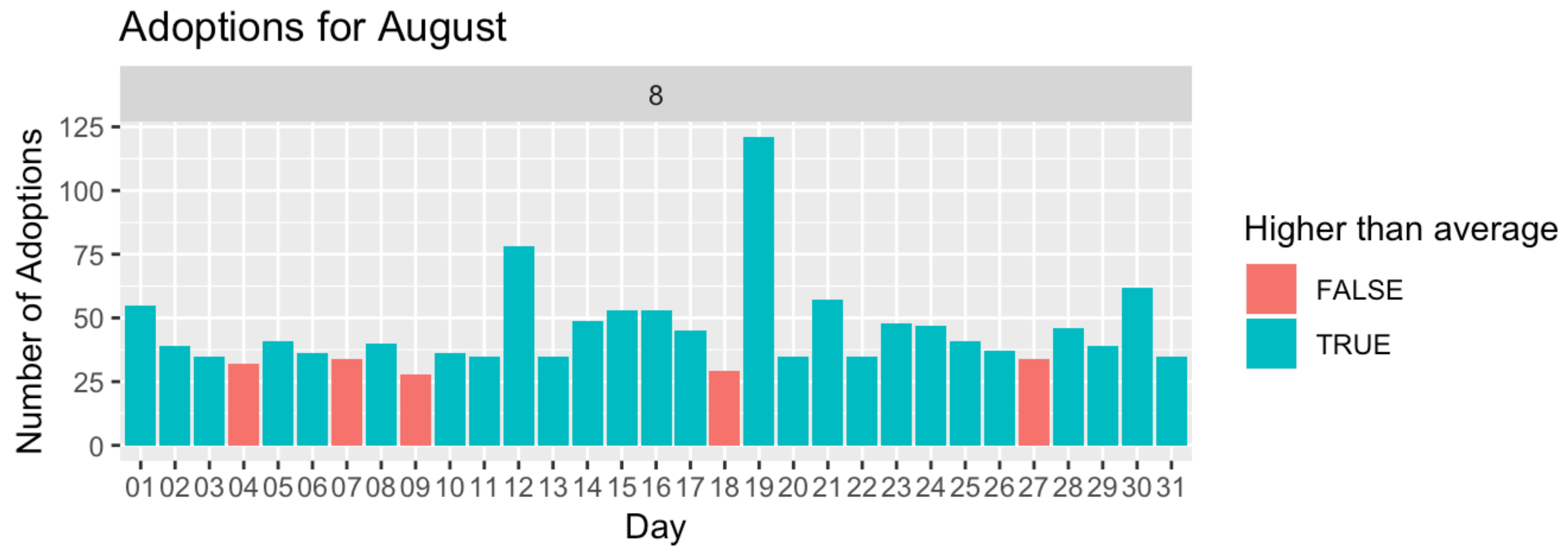
In this plot, we see the sharp difference between the first half and the second half of the year more pronounced.

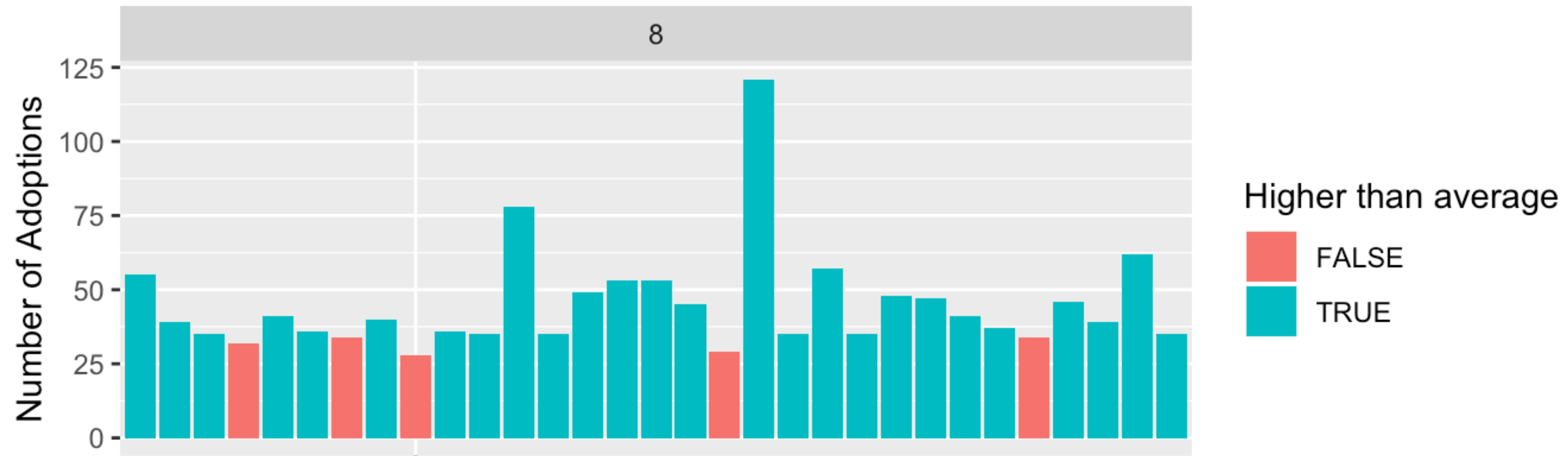
We notice that February through May had only a handful of days that were above the average amount of daily adoptions for the year. Based on these graphs, we can recommend focusing efforts on earlier months, for bringing up adoption numbers. Now, we can explore how events played a role in adoption numbers.

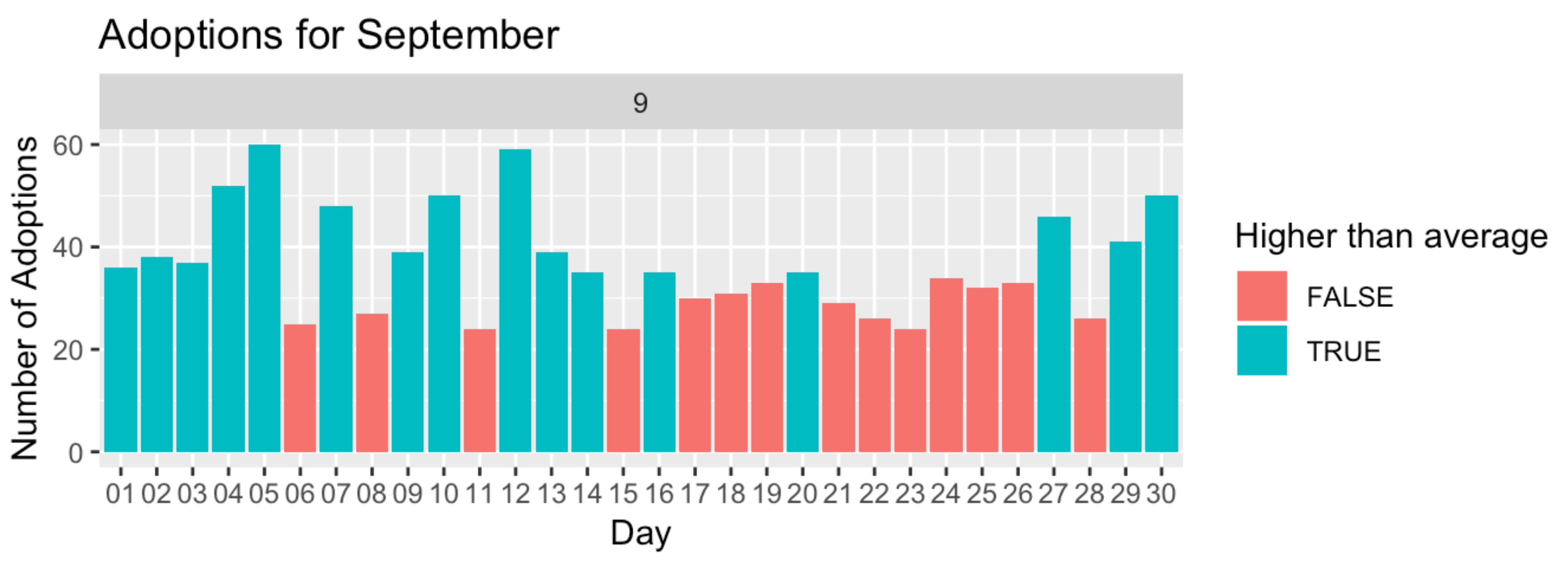
Analysing events impact on adoption counts:

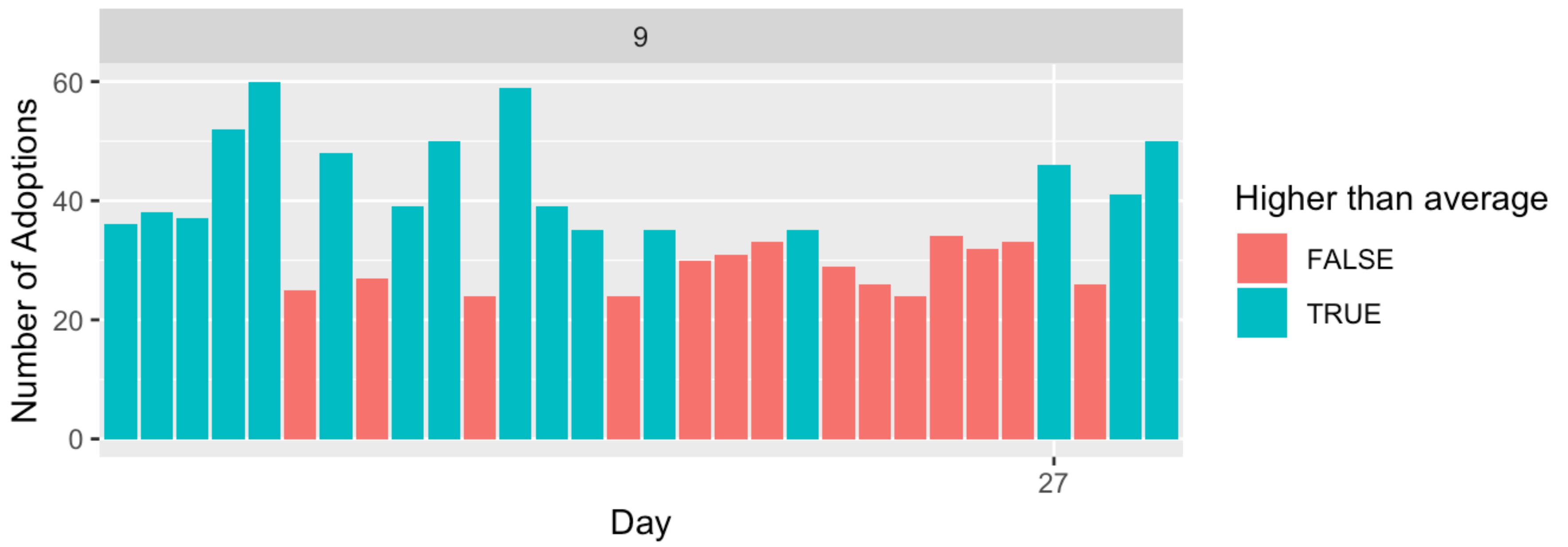
```
## # A tibble: 4 x 5
##   event                          event_date event_month event_day event_Season
##   <chr>                          <chr>            <dbl> <chr>     <fct>
## 1 Hot Summer, cool seniors a… 8/24/2013            8 24        Summer
## 2 Act of Kindness with Princ… 1/18/2014            1 18        Winter
## 3 Service project at Austin … 8/9/2014             8 9         Summer
## 4 Austin Pittie Limits        9/27/2014            9 27        Fall
```


1
count
outcome_date
(higher_avg)
FALSE
TRUE


Adoptions for August
8
Number of Adoptions
Day
Higher than average
FALSE
TRUE


8
Number of Adoptions
Higher than average
FALSE
TRUE





Comparing the average adoption count for the year, it seems that the only event that had an above average adoption count for the day was the Austin Pittie Limits event held on 9/27/2014. This was an event that coincided with a music festival, and had live events and many sponsors. It is hard to tell whether the event contributed to the higher average, due to the small sample size of events. Initial thoughts from this would be to research more on how events affect adoption rates, and whether or not it would be feasible for the shelter to host more events.

Checking the shelter calendar for the year, it seems that the shelter has focused on hosting more events throughout the year. This could be a step in right direction, as there may be a positive impact on adoption rates from outreach.

Now, we can use a Goodnes of Fit test to see the differences between seasons, and analyze the difference between the different parts of the year.

Conduct Goodness of Fit test on Seasons:

```
## # A tibble: 4 x 2
##   Season count
##   <fct>  <int>
## 1 Winter  5466
## 2 Spring  5629
## 3 Summer  9954
## 4 Fall    8369
```

```
## [1] 7354.5 7354.5 7354.5 7354.5
```

```
## [1] 5466 5629 9954 8369
```

```
## [1] -25.42786 -23.23313  35.00117  13.65982
```

```
## 
##  Chi-squared test for given probabilities
## 
## data:  season_test$count
## X-squared = 1948.5, df = 3, p-value < 2.2e-16
```

Based on the results of the Goodness of Fit test, the rankings for month adoptions were from greatest to least: Summer, Fall, Spring, Winter. This confirms what we saw in the previous plots, that there is a significant difference between the different parts of the year.

Initial modeling:

Setting up new datasets for modeling:

We can set up new data sets for modeling by only selecting the variables that would be useful the model. We can eliminate datetime since we have grouped that data within other columns, among other variables that have many NA's and are already represented with other variables.

Since we are trying to see how occasion affects adoption as a binary event, we can run a logistic regression model based on the occasion features we have created: adoption date, season, whether the date has an event.

Logistic Model:

```
# Adopted = adoption_month + year + weekday + hour + date + season + event
# create log reg model based on all 7 vars
log.adopted <- glm(adopted ~ ., data = cats_log,
                family = binomial)
summary(log.adopted)
```

```
## 
## Call:
## glm(formula = adopted ~ ., family = binomial, data = cats_log)
## 
## Deviance Residuals: 
##     Min       1Q   Median       3Q      Max  
## -2.1496  -0.9849  -0.6102   1.0700   2.7451  
## 
## Coefficients:
##                            Estimate Std. Error z value Pr(>|z|)    
## (Intercept)              -3.717e+02  2.108e+01 -17.638  < 2e-16 ***
## outcome_month             4.971e-02  5.329e-03   9.329  < 2e-16 ***
## outcome_year              1.831e-01  1.045e-02  17.521  < 2e-16 ***
## outcome_weekdayMonday    -1.325e-01  4.970e-02  -2.665  0.00769 ** 
## outcome_weekdaySaturday   8.671e-01  4.638e-02  18.695  < 2e-16 ***
## outcome_weekdaySunday     5.963e-01  4.738e-02  12.585  < 2e-16 ***
## outcome_weekdayThursday  -1.036e-01  5.079e-02  -2.039  0.04144 *  
## outcome_weekdayTuesday    1.114e-02  4.801e-02   0.232  0.81646    
## outcome_weekdayWednesday -1.405e-02  4.983e-02  -0.282  0.77805    
## outcome_hour              1.641e-01  3.912e-03  41.954  < 2e-16 ***
## outcome_date02           -4.483e-02  9.947e-02  -0.451  0.65218    
## outcome_date03           -8.906e-03  1.008e-01  -0.088  0.92961    
## outcome_date04           -1.398e-01  1.017e-01  -1.375  0.16906    
## outcome_date05           -1.524e-01  1.004e-01  -1.517  0.12918    
## outcome_date06           -4.188e-01  1.007e-01  -4.159 3.19e-05 ***
## outcome_date07            2.984e-02  1.014e-01   0.294  0.76860    
## outcome_date08           -4.598e-01  1.003e-01  -4.586 4.53e-06 ***
## outcome_date09            1.642e-02  9.730e-02   0.169  0.86597    
## outcome_date10           -7.684e-02  1.020e-01  -0.754  0.45109    
## outcome_date11           -3.206e-01  9.893e-02  -3.240  0.00119 ** 
## outcome_date12            3.036e-01  9.606e-02   3.160  0.00158 ** 
## outcome_date13           -7.513e-02  9.755e-02  -0.770  0.44122    
## outcome_date14           -1.531e-01  9.899e-02  -1.546  0.12206    
## outcome_date15           -5.037e-02  1.016e-01  -0.496  0.62020    
## outcome_date16           -1.397e-01  1.006e-01  -1.388  0.16519    
## outcome_date17           -1.807e-01  9.965e-02  -1.814  0.06970 .  
## outcome_date18           -7.934e-02  9.613e-02  -0.825  0.40918    
## outcome_date19            1.322e-01  9.436e-02   1.401  0.16121    
## outcome_date20           -3.020e-01  9.913e-02  -3.046  0.00232 ** 
## outcome_date21           -1.798e-01  9.803e-02  -1.834  0.06671 .  
## outcome_date22           -2.283e-01  9.812e-02  -2.327  0.01997 *  
## outcome_date23           -9.952e-02  9.915e-02  -1.004  0.31549    
## outcome_date24           -8.099e-02  9.916e-02  -0.817  0.41408    
## outcome_date25           -1.460e-01  9.989e-02  -1.461  0.14395    
## outcome_date26           -2.620e-01  9.786e-02  -2.677  0.00743 ** 
## outcome_date27           -1.462e-01  9.910e-02  -1.475  0.14021
## outcome_date28            2.392e-02  9.823e-02   0.244  0.80759
## outcome_date29            1.827e-01  9.679e-02   1.887  0.05913 .
## outcome_date30            2.220e-01  9.800e-02   2.266  0.02348 *
## outcome_date31           -2.333e-01  1.188e-01  -1.964  0.04953 *
## SeasonSpring             -1.059e+00  4.390e-02 -24.135  < 2e-16 ***
## SeasonSummer             -3.402e-01  3.658e-02  -9.300  < 2e-16 ***
## SeasonFall               -5.305e-01  4.267e-02 -12.434  < 2e-16 ***
## has_event                -9.080e-02  1.340e-01  -0.678  0.49788
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 40249  on 29417  degrees of freedom
## Residual deviance: 35688  on 29374  degrees of freedom
## AIC: 35776
##
## Number of Fisher Scoring iterations: 4
```

Our logistic model is **Adopted = adoption_month + year + weekday + hour + date + season + event**. We can see that adoption month, year, hour, and season are significant. Some adoption weekdays and dates are signicant, which is supported with our previous statistical tests. This also shows that some dates are more important than others for adoption count for the day, however, we should be careful with overfitting the model with the data.

Unfortunately, the event variable is insignificant. This is most likely due to a very small sample size. We would need more data from recent years to see how much of an impact events have on adoptions.

```
##         1         2         3         4         5         6
## 0.3859894 0.4686269 0.6561995 0.1870401 0.5498921 0.3892125
```

```
## 1 2 3 4 5 6
## 0 0 1 0 1 0
## Levels: 0 1
```

```
## [1] 0.3212319
```

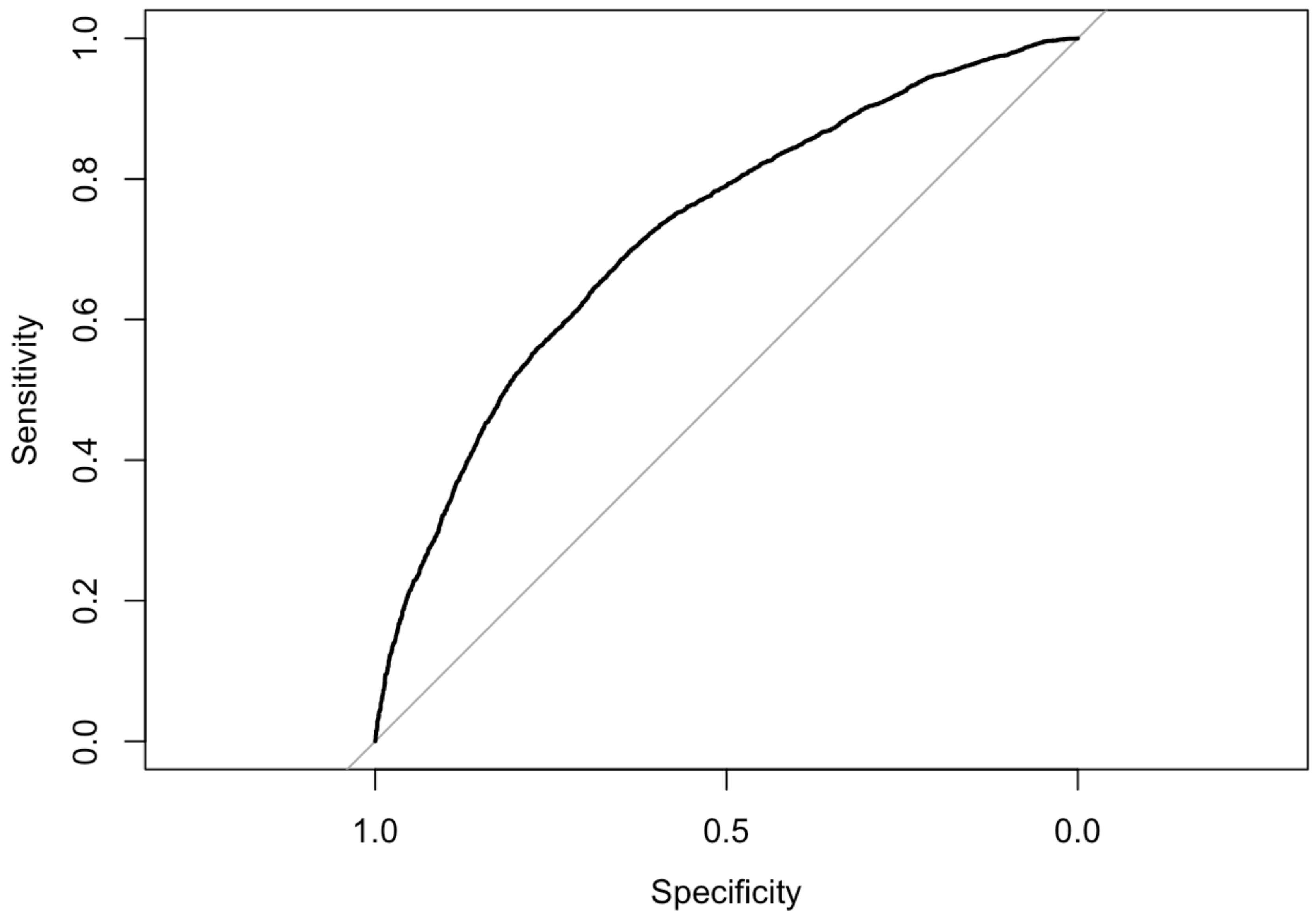
[1] 0.3212319

```
## [1] 0.2099368 0.2099030
```

[1] 0.2099641 0.2099286

```
## Area under the curve: 0.7223
```

```
##   threshold specificity sensitivity
##   0.4286203   0.6505454   0.6852026
```



We found that the misclassification rate for our model is 0.3212319 and the 10-fold cross validation MSE is 0.2099286 unbiased. This leaves us with an accuracy of 67.9%. This initial model does not perform too well, but we can conclude that there is only so much information occasion can give overall. Although unsuccessful, this model serves good insights as a preliminary model.

We can also use another powerful classification model, Random Forest, to help give insights on what occasion features play a role in determining adoption.

Random forest model dataset preparation:

```
# make cats random forest data set by getting rid of columns with many NAs
cats_rf <- cats_fixed %>%
  select(-event ,-event_date, -event_Season, -outcome_subtype, -datetime, - out
come_type) %>%
  mutate_if(is.character, as.factor) # RF doesn't work with char

# check NA count for each column
sapply(cats_rf, function(x) sum(is.na(x)))
```

```
##   outcome_month    outcome_year outcome_weekday    outcome_hour
##               0               0               0               0
##    outcome_date          Season       has_event         adopted
##               0               0               0               0
```

```
# change adopted to factor
cats_rf$adopted <- as.character(cats_rf$adopted)
cats_rf$adopted <- as.factor(cats_rf$adopted)
```

In order to use random forest, we must remove any columns with many NAs. Luckily, these columns are represented by other variables, so we are not losing much data. We also turn any character data in factors.

We split the data into training and validation sets.

```
## # A tibble: 20,592 x 8
##    outcome_month outcome_year outcome_weekday outcome_hour outcome_date
##            <dbl>        <dbl> <fct>                  <dbl> <fct>
##  1             8         2014 Sunday                    16 31
##  2             9         2017 Thursday                   0 28
##  3             7         2016 Saturday                  19 09
##  4             5         2016 Wednesday                  9 04
##  5             3         2017 Friday                    13 17
##  6             5         2017 Friday                    18 26
##  7             4         2016 Monday                    11 25
##  8             3         2014 Monday                    15 03
##  9             8         2014 Sunday                     9 03
## 10            12         2015 Wednesday                 19 02
## # … with 20,582 more rows, and 3 more variables: Season <fct>,
## #   has_event <int>, adopted <fct>
## # A tibble: 8,826 x 8
##    outcome_month outcome_year outcome_weekday outcome_hour outcome_date
##            <dbl>        <dbl> <fct>                  <dbl> <fct>
##  1             8         2014 Thursday                  18 14
##  2             8         2014 Wednesday                 16 27
##  3             7         2014 Saturday                  15 12
##  4            11         2014 Sunday                     9 09
##  5             9         2014 Saturday                  18 20
##  6             9         2014 Sunday                     9 14
##  7             5         2014 Tuesday                   10 13
##  8            12         2013 Tuesday                   12 24
##  9            12         2013 Tuesday                   16 24
## 10            10         2014 Sunday                    16 05
## # … with 8,816 more rows, and 3 more variables: Season <fct>,
## #   has_event <int>, adopted <fct>
```

Run Random forest model based on occasion data.

```
# rf model to classify adopted
rf.model <- randomForest::randomForest(adopted ~ ., data = TrainSet, importance
= TRUE)
```

```
# outoput
rf.model
```

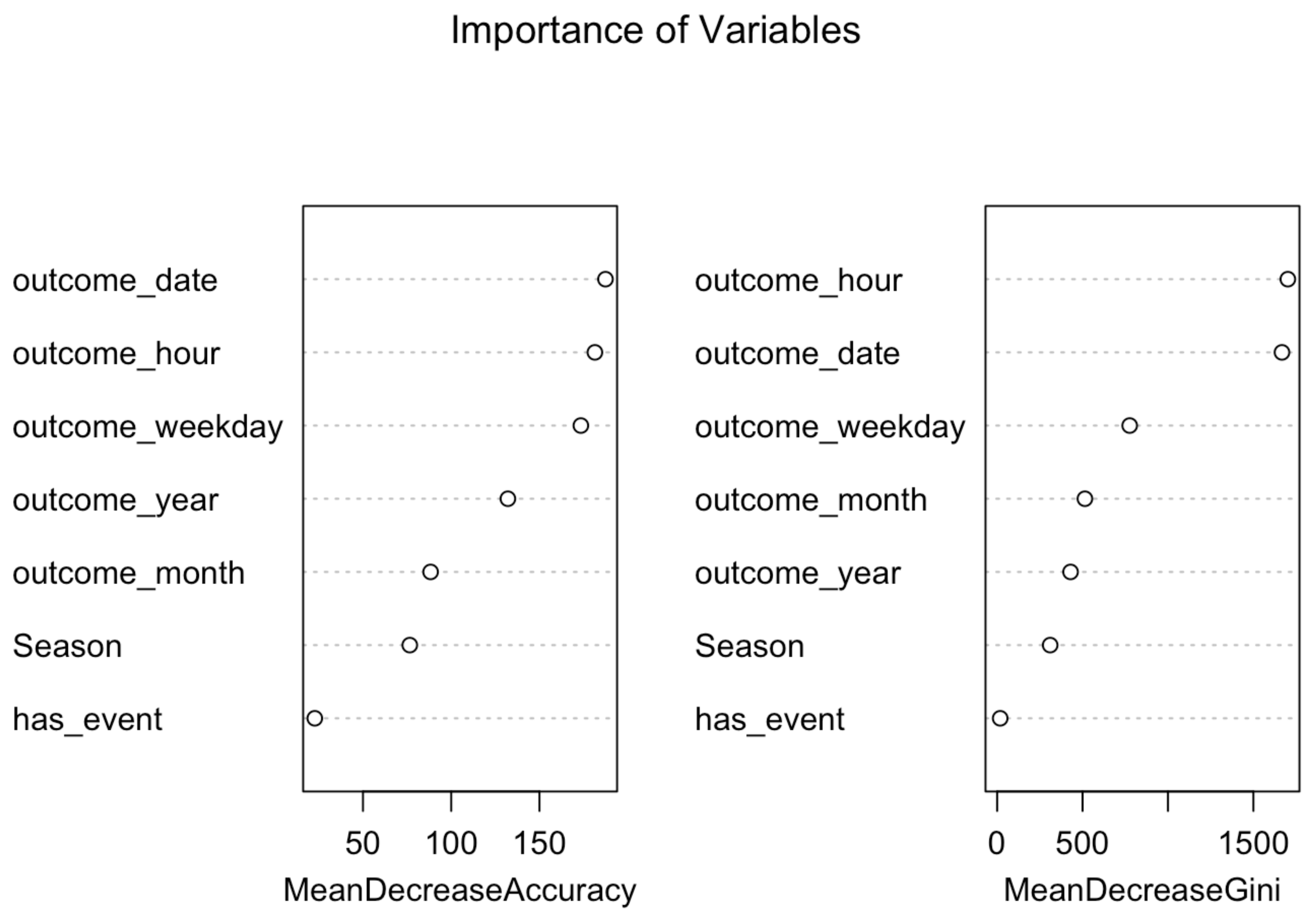
```
##
## Call:
##  randomForest(formula = adopted ~ ., data = TrainSet, importance = TRUE)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 23.18%
## Confusion matrix:
##      0    1 class.error
## 0 9678 1942   0.1671256
## 1 2832 6140   0.3156487
```

```
##
## predTrain     0     1
##         0 10523  1931
##         1  1097  7041
## [1] 0.7745298
```

```
##
## predValid    0    1
##         0 4240 1164
##         1  826 2596
```

```
## [1] 0.2254702
```

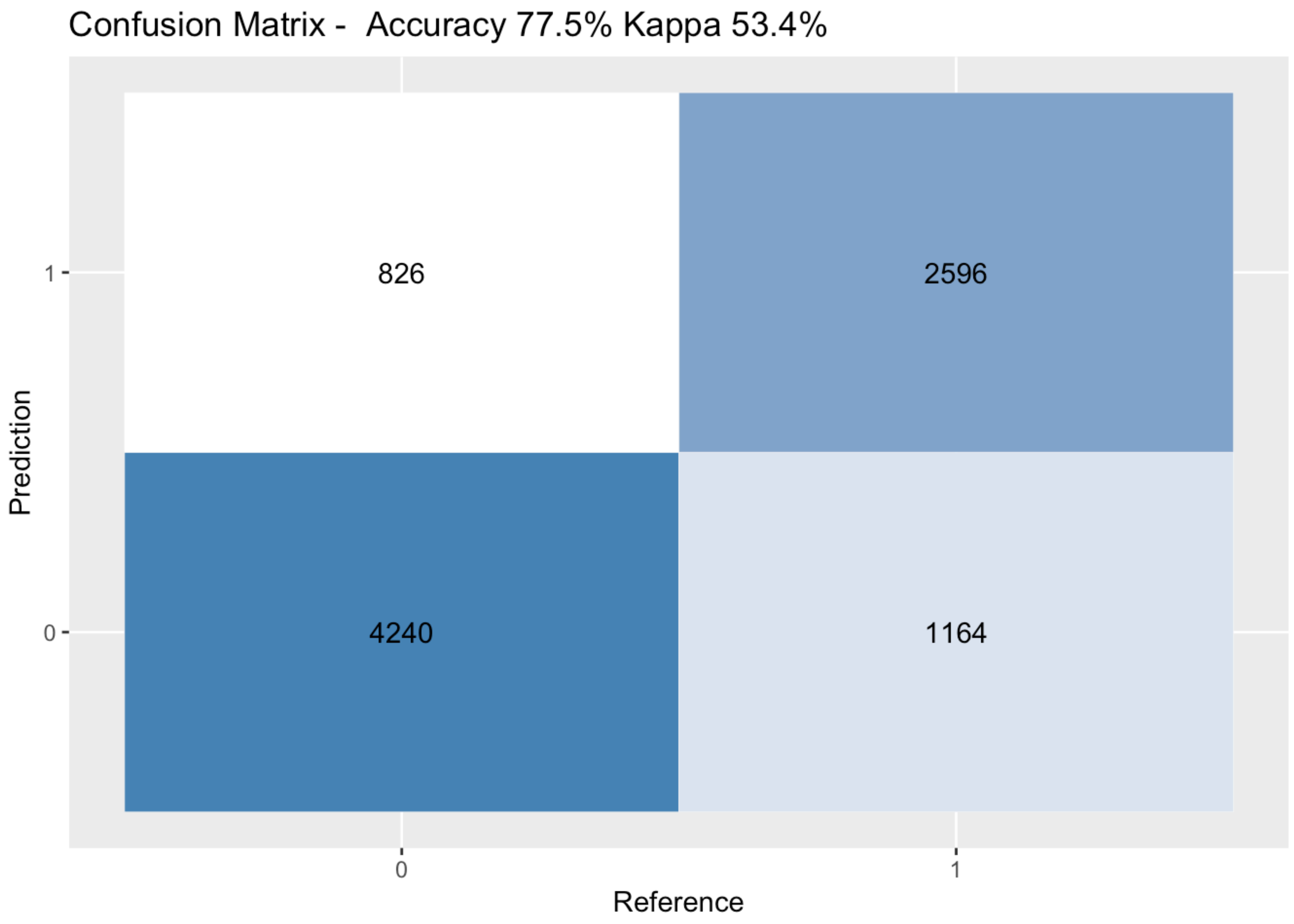
```
##                         0          1 MeanDecreaseAccuracy MeanDecreaseGini
## outcome_month    85.37289  30.738968             88.31464        513.69625
## outcome_year    125.39619  28.315716            132.10532        429.75191
## outcome_weekday 161.27538  61.213992            173.46658        775.88310
## outcome_hour    165.45555 153.838350            181.41747       1701.68535
## outcome_date    186.12921  35.185746            187.40819       1667.36740
## Season           73.30158  35.244908             76.52054        310.32886
## has_event        29.12108  -5.508636             22.68133         17.75381
```



We can use the caret package to plot a confusion matrix, which will show the prediction vs reference numbers for the model.

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    0    1
##          0 4240 1164
##          1  826 2596
## 
##                Accuracy : 0.7745
##                  95% CI : (0.7657, 0.7832)
##     No Information Rate : 0.574
##     P-Value [Acc > NIR] : < 2.2e-16
## 
##                   Kappa : 0.5336
## 
##  Mcnemar's Test P-Value : 4.206e-14
## 
##             Sensitivity : 0.8370
##             Specificity : 0.6904
##          Pos Pred Value : 0.7846
##          Neg Pred Value : 0.7586
##              Prevalence : 0.5740
##          Detection Rate : 0.4804
##    Detection Prevalence : 0.6123
##       Balanced Accuracy : 0.7637
## 
##        'Positive' Class : 0
## 
```

```
## [1] "Accuracy of Random Forest Model =  0.774529798323136"
```



We can see that the random forest model, based on occasion features, performs much better, with a higher accuracy. Although more powerful, this model is harder to interpret, but we can check the overall accuracy from the confusion matrix. Now that we have a model that performs better, we can tune the random forest model based on accuracy.

Tuning random forest for mtry.

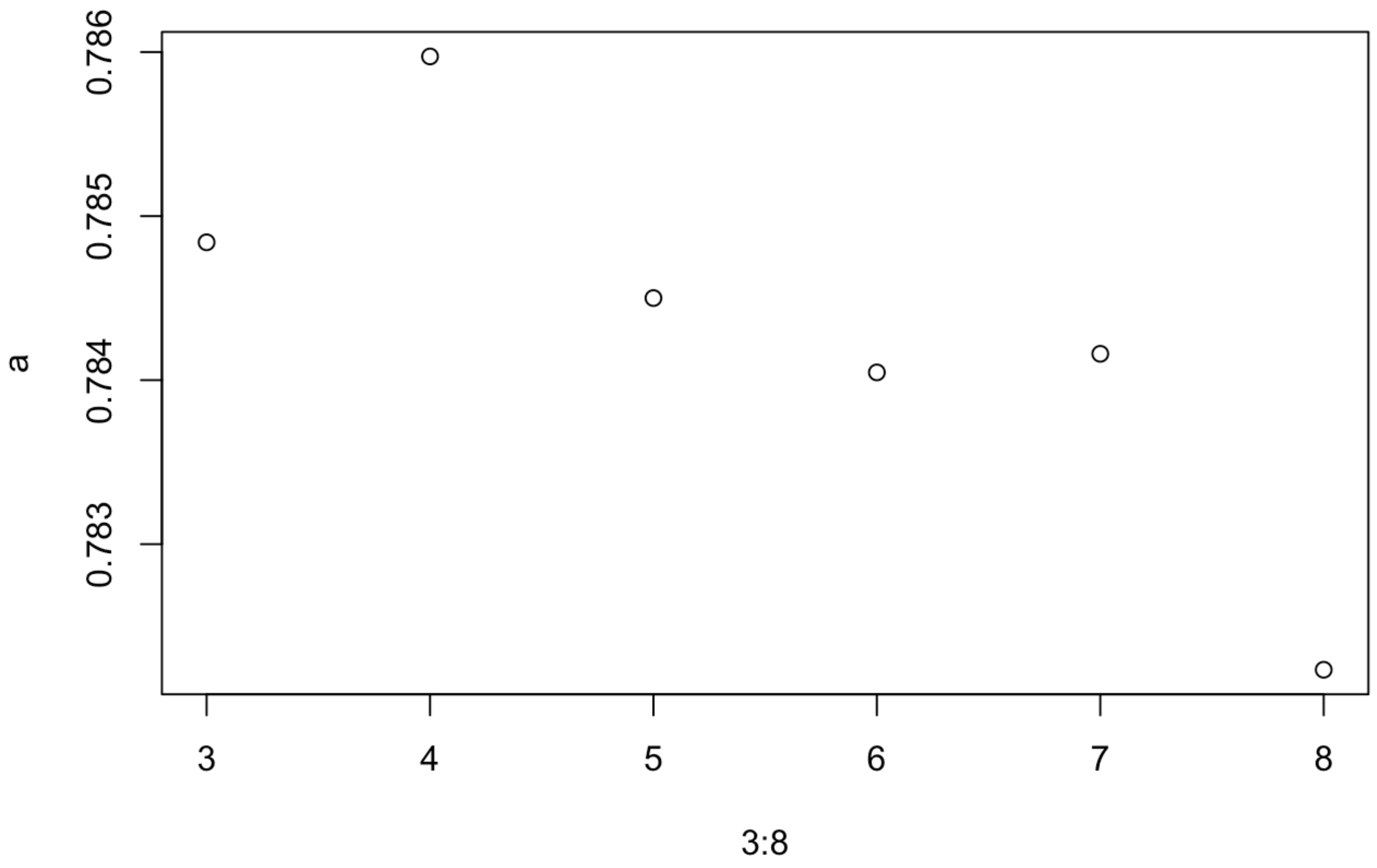
```
# Using For loop to identify the right mtry for model
a=c()
i=5
for (i in 3:8) {
  model.temp <- randomForest::randomForest(adopted ~ ., data = TrainSet, ntree
= 500, mtry = i, importance = TRUE)
  predValid <- predict(model.temp, ValidSet, type = "class")
  a[i-2] = mean(predValid == ValidSet$adopted)
}
```

```
## Warning in randomForest.default(m, y, ...): invalid mtry: reset to within
## valid range
```

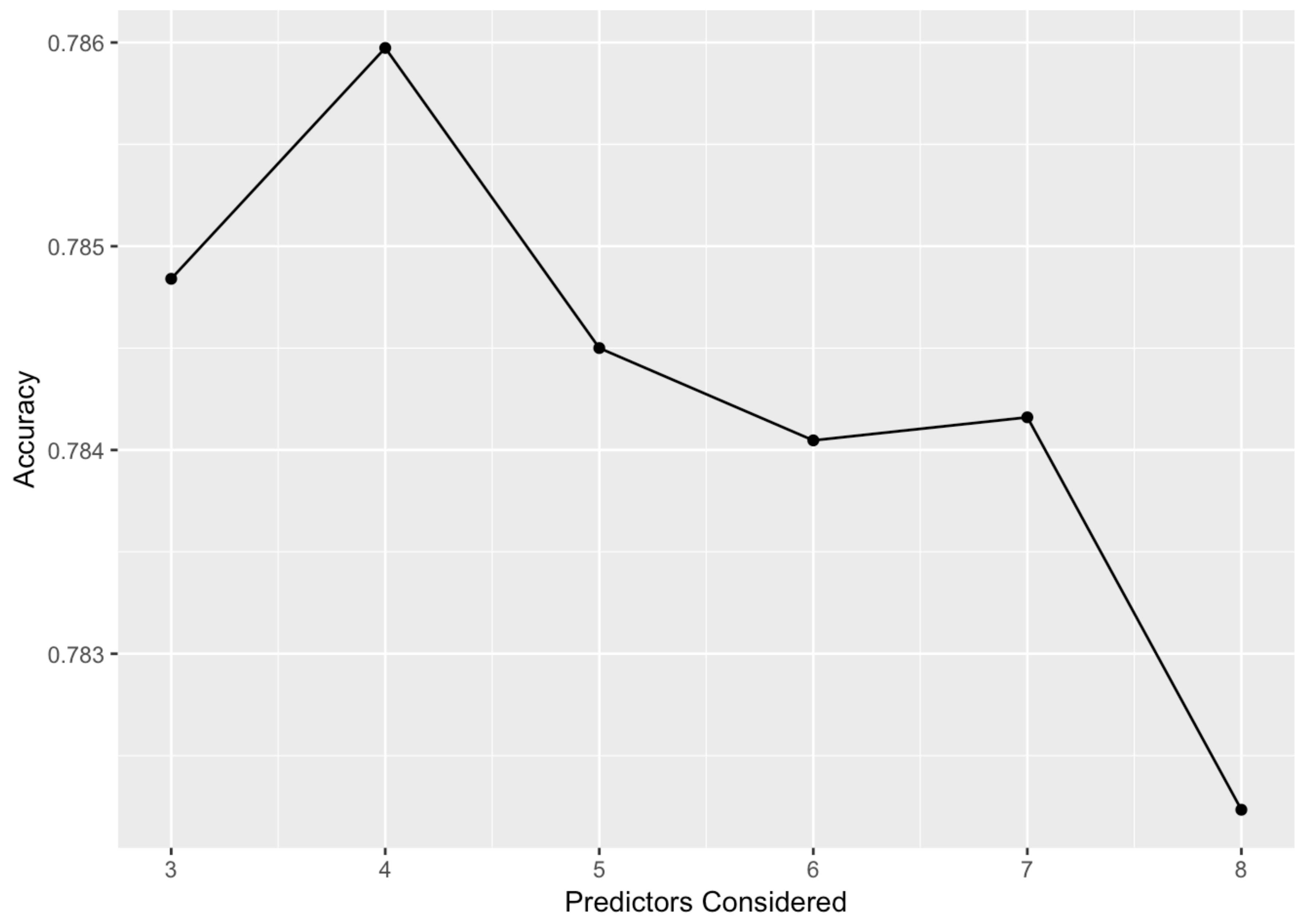
```
# output
a
```

```
## [1] 0.7848402 0.7859733 0.7845003 0.7840471 0.7841604 0.7822343
```

```
# plot
plot(3:8,a)
```



```
##       value index
## 1 0.7848402     3
## 2 0.7859733     4
## 3 0.7845003     5
## 4 0.7840471     6
## 5 0.7841604     7
## 6 0.7822343     8
```



We see that the highest accuracy for the model is with 4 predictors considered. The mtry parameter is the number of variables randomly sampled as predictor candidates at each tree split. Now we can make a new model with these new parameters.

New random forest model:

```
# rf model tuned with mtry = 4
rf.model.tuned <- randomForest::randomForest(adopted ~ ., data = TrainSet, ntre
e = 500, mtry = 4, importance = TRUE)
```

```
##
## Call:
##  randomForest(formula = adopted ~ ., data = TrainSet, ntree = 500,      mtry
= 4, importance = TRUE)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 4
##
##         OOB estimate of  error rate: 22.74%
## Confusion matrix:
##      0    1 class.error
## 0 9453 2167   0.1864888
## 1 2515 6457   0.2803165
```

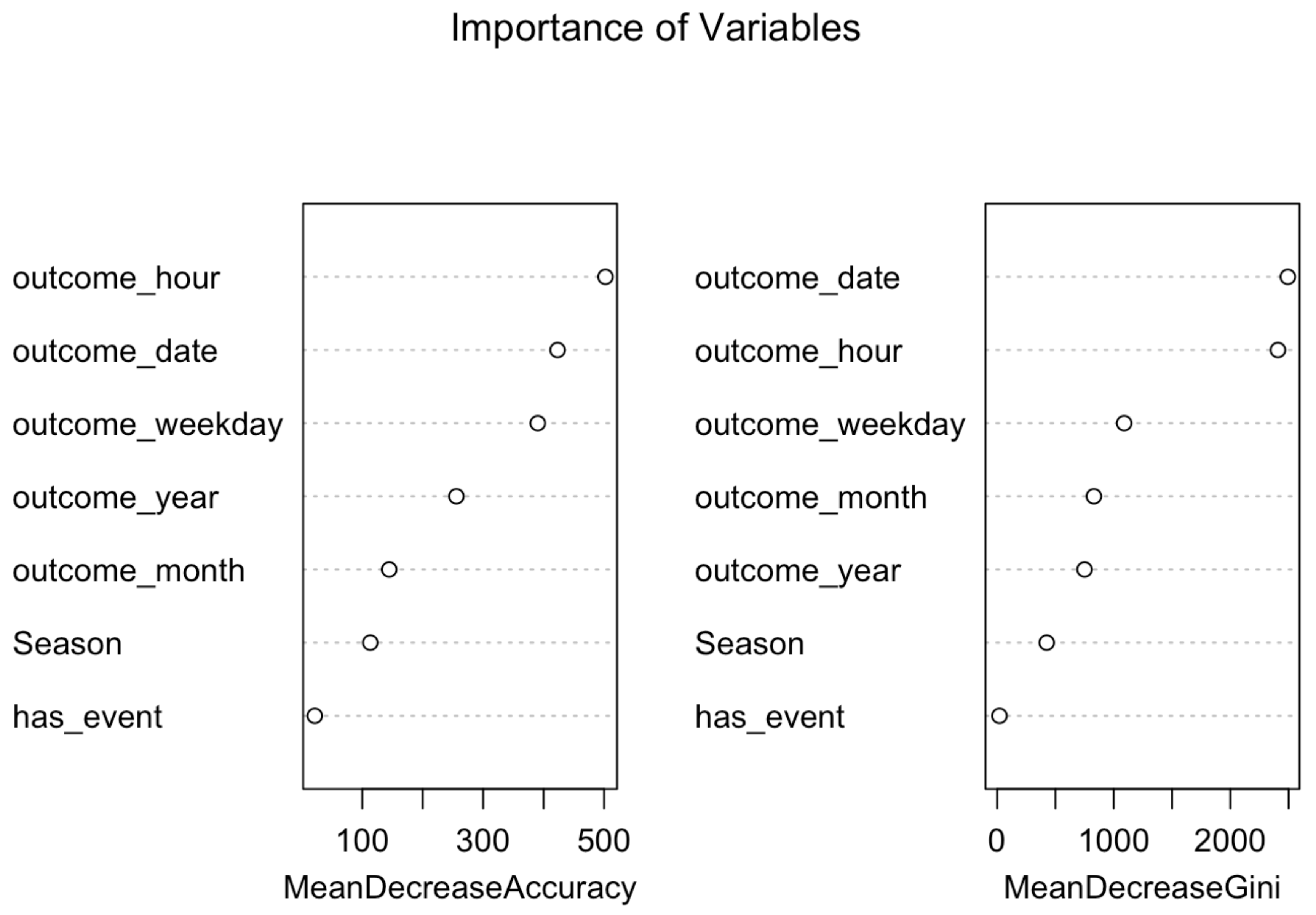
```
##
## predTrain.tuned     0     1
##               0 10897  1138
##               1   723  7834
```

```
## [1] 0.7865398
```

```
##
## predValid.tuned    0    1
##               0 4197 1015
##               1  869 2745
```

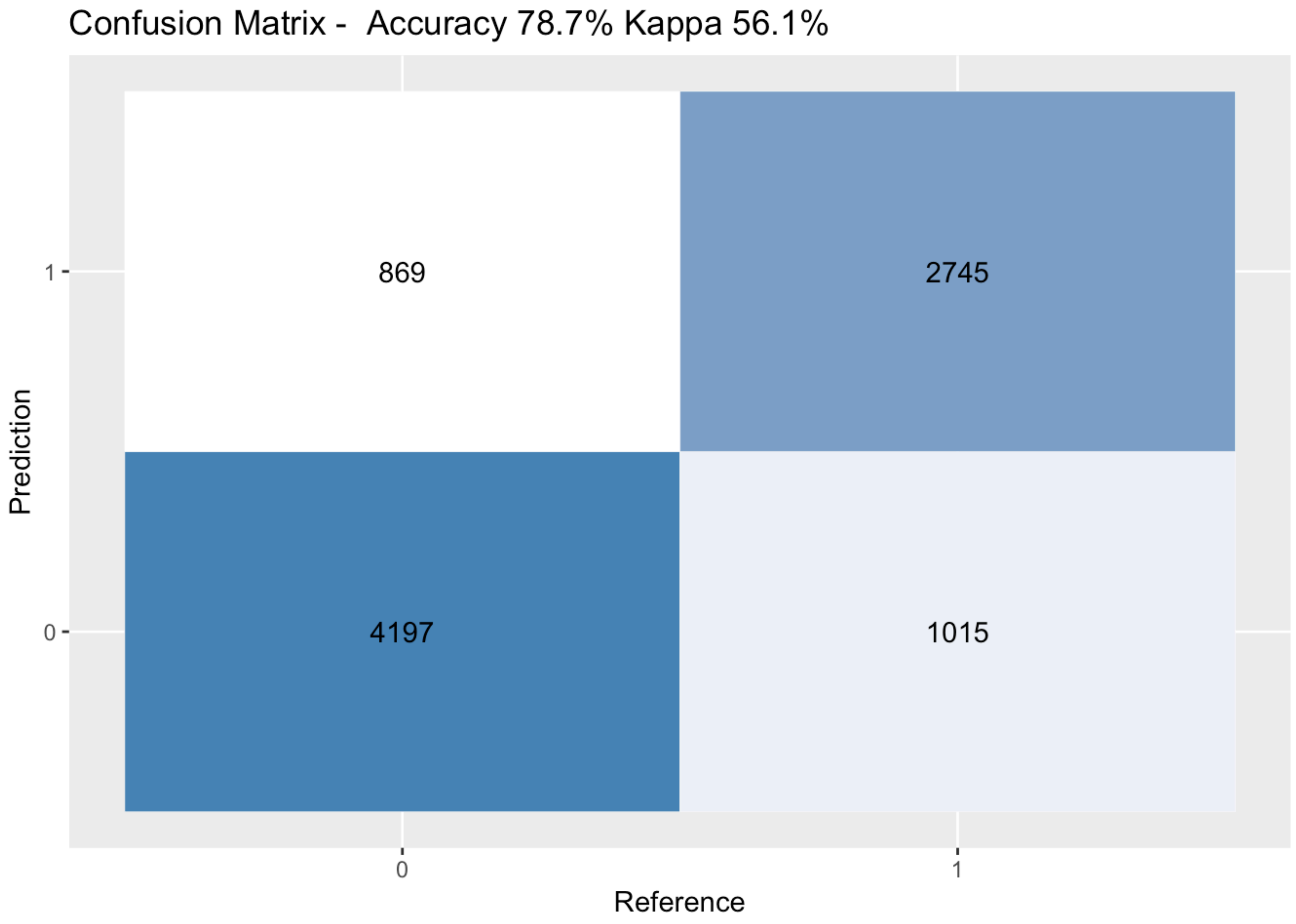
```
## [1] 0.2134602
```

```
##                         0          1 MeanDecreaseAccuracy MeanDecreaseGini
## outcome_month   137.06328  61.732629            144.64502        829.59895
## outcome_year    247.35633  56.320811            255.72589        749.34920
## outcome_weekday 365.32038 110.121550            390.30635       1089.25346
## outcome_hour    484.75598 217.568587            502.29728       2408.14068
## outcome_date    411.48894  83.796226            423.03974       2493.01864
## Season          113.20217  52.883515            113.32279        424.71895
## has_event        25.04228  -4.058483             21.52359         19.81836
```


Importance of Variables
outcome_hour
outcome_date
outcome_weekday
outcome_year
outcome_month
Season
has_event
100
300
500
MeanDecreaseAccuracy
outcome_date
outcome_hour
outcome_weekday
outcome_month
outcome_year
Season
has_event
0
1000
2000
MeanDecreaseGini

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    0    1
##          0 4197 1015
##          1  869 2745
##                                           
##                Accuracy : 0.7865          
##                  95% CI : (0.7778, 0.795) 
##     No Information Rate : 0.574           
##     P-Value [Acc > NIR] : < 2.2e-16       
##                                           
##                   Kappa : 0.5613          
##                                           
##  Mcnemar's Test P-Value : 0.0008359       
##                                           
##             Sensitivity : 0.8285          
##             Specificity : 0.7301          
##          Pos Pred Value : 0.8053          
##          Neg Pred Value : 0.7595          
##              Prevalence : 0.5740          
##          Detection Rate : 0.4755          
##    Detection Prevalence : 0.5905          
##       Balanced Accuracy : 0.7793          
##                                           
##        'Positive' Class : 0               
## 
```

```
## [1] "Accuracy of Random Forest Model =  0.786539768864718"
```



Final model accuracy: 78.6%, Out of sample error rate: 22.74%, which is similar to LOOCV.

The chart displays the variables that affected the random forest, in order from highest to lowest. We use the Gini index for classifying node purity, whether or not the node contains observations predominantly from a single class. We see that Adoption date and hour are the two most powerful variables.

From this model, we have a better outlook on what occasion variables affect adoption the most, and we can use this to better tune our final overall model.