Sun Country Customer Analysis

October 22, 2019

MSBA 6410: Exploratory Data Analysis & Visualization

Authors: Sarah Black, Michael DeGuire, Anthony Meyers, Danny Moncada, and Jonathan Watkins

Table of Contents

1 Problem Statement and Approach

1.1 Sun Countries Objectives

1.2 Our Approach

2 Data Preparation

3 Exploratory Analysis

3.1 Descriptive Analysis - Full Data Set

3.1.1 Examining UFly Enrollments Per Month

3.1.2 Examining UFly Member Percentages by Status

3.1.3 Examining Customer Flights (Service Starts) per Month

3.1.4 Identifying Top Customer Destinations and Routes

3.1.5 Visualizing Sun Country Routes

3.1.6 Examining Customer Routes and Flights by Season

4 Clustering Strategies Towards Customer Segmentation

4.1 Characteristics of Fliers

4.2 Characteristics of UFly Reward members

5 Conclusions

6 Recommendations

1 Problem Statement and Approach

1.1 Sun Countries Objectives

In order to compete with major airline firms, Sun Country Airlines needs to be savvy and sophisticated in their marketing and customer interaction strategy. Sun Countries Ufly reward program and digital experience provide Sun Country a platform in which to enable modern marketing and analytics techniques to create value to enable Sun Country to compete with other, more resource enabled, airline firms. Leveraging exploratory analytics techniques, such as clustering analysis, will enable Sun Country to generate insights and actionable hypotheses from the data. Examining the characteristics of customer cohorts and their flying habits will help Sun Counry focus their strategic initiatives to delivering best in class service and options to the Sun Country customer base.

1.2 Our Approach

We propose a customer segmentation analytics approach to guide the exploration of customer characteristics and how they travel. Specifically, we choose to explore the portion of customer who use the Minneapolis - St.Paul Sun Country hub as their start point for their journeys, as coded in the data. We then leverage a k-medioids approach to cluster mixed data-types. This partioning around mediods algorithm leverages Gower distance to appropriately compute partial dissimilarities and is very intuitive. It also has an extremely useful feature in that it can produce statistics around what a typical customer might look like for each cluster, allowing us to see who is, or isn't, traveling with a UFly membership or booking through the SCA site.

2 Data Preparation

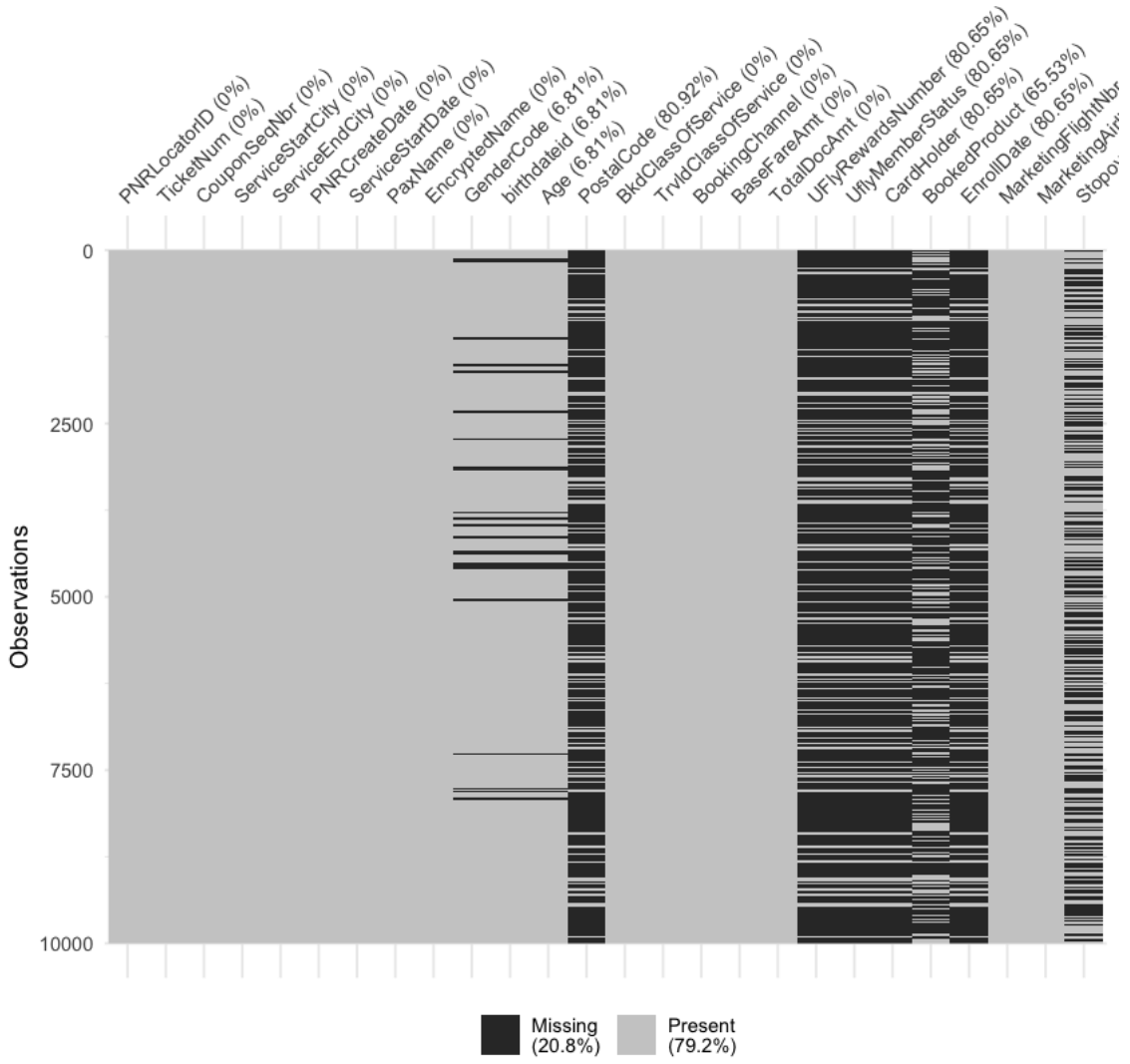
[1]:
```
# Importing Packages
suppressPackageStartupMessages(library(tidyverse))
suppressPackageStartupMessages(library(cluster))
suppressPackageStartupMessages(library(Rtsne))
suppressPackageStartupMessages(library(lubridate))

# Numeric Formatting
options(scipen = 999)

# Import and parse data
suppressWarnings(suppressMessages(df <- readr::read_csv(file = "~/Downloads/HW 2/
 ↪SunCountry.csv")))

# Initial Review of Data
str(df)
```

```
Classes ‘spec_tbl_df’, ‘tbl_df’, ‘tbl’ and 'data.frame':        3435388 obs. of
26 variables:
 $ PNRLocatorID        : chr  "AAABJK" "AAABJK" "AAABMK" "AAABMK" ...
 $ TicketNum           : num  3377365159634 3377365159634 3372107381942
3372107381942 3372107470782 ...
 $ CouponSeqNbr        : num  2 1 2 1 1 1 1 1 1 1 ...
 $ ServiceStartCity    : chr  "JFK" "MSP" "MSP" "SFO" ...
 $ ServiceEndCity      : chr  "MSP" "JFK" "SFO" "MSP" ...
 $ PNRCreateDate       : Date, format: "2013-11-23" "2013-11-23" ...
 $ ServiceStartDate    : Date, format: "2013-12-13" "2013-12-08" ...
 $ PaxName             : chr  "BRUMSA" "BRUMSA" "EILDRY" "EILDRY" ...
 $ EncryptedName       : chr  "4252554D4241434B44696420493F7C2067657420746869732
0726967687453414E445241204C4545" "4252554D4241434B44696420493F7C2067657420746869
732072696768745341 4E445241204C4545"
"45494C4445525344696420493F7C20676574207468697320726976874525941 4E204C"
"45494C4445525344696420493F7C20676574207468697320726976874525941 4E204C" ...
 $ GenderCode          : chr  "F" "F" "M" "M" ...
 $ birthdateid         : num  35331 35331 46161 46161 34377 ...
 $ Age                 : num  66 66 37 37 69 54 25 69 49 58 ...
 $ PostalCode          : num  NA NA NA NA NA ...
 $ BkdClassOfService   : chr  "Coach" "Coach" "Coach" "Coach" ...
 $ TrvldClassOfService : chr  "Coach" "First Class" "Discount First Class"
"Discount First Class" ...
 $ BookingChannel      : chr  "Outside Booking" "Outside Booking" "SCA Website
Booking" "SCA Website Booking" ...
 $ BaseFareAmt         : num  234 234 294 294 113 ...
 $ TotalDocAmt         : num  0 0 338 338 132 ...
 $ UFlyRewardsNumber   : num  NA NA NA NA NA ...
 $ UflyMemberStatus    : chr  NA NA NA NA ...
 $ CardHolder          : logi  NA NA NA NA NA NA ...
 $ BookedProduct       : chr  "CHEOPQ" "CHEOPQ" NA NA ...
 $ EnrollDate          : POSIXct, format: NA NA ...
 $ MarketingFlightNbr  : chr  "244" "243" "397" "392" ...
 $ MarketingAirlineCode: chr  "SY" "SY" "SY" "SY" ...
 $ StopoverCode        : chr  "O" NA "O" NA ...
 - attr(*, "problems")=Classes ‘tbl_df’, ‘tbl’ and 'data.frame':        1200
obs. of  5 variables:
  ..$ row     : int  4620 5245 5302 5381 5386 5387 6873 6874 18536 18537 ...
  ..$ col     : chr  "PostalCode" "PostalCode" "PostalCode" "PostalCode" ...
  ..$ expected: chr  "a double" "a double" "a double" "a double" ...
  ..$ actual  : chr  "MN 55" "V0R 2" "V0R 2" "MN 55" ...
  ..$ file    : chr  "'~/Downloads/HW 2/SunCountry.csv'" "'~/Downloads/HW
2/SunCountry.csv'" "'~/Downloads/HW 2/SunCountry.csv'" "'~/Downloads/HW
2/SunCountry.csv'" ...
 - attr(*, "spec")=
  .. cols(
  ..   PNRLocatorID = col_character(),
  ..   TicketNum = col_double(),
```

```
..   CouponSeqNbr = col_double(),
..   ServiceStartCity = col_character(),
..   ServiceEndCity = col_character(),
..   PNRCreateDate = col_date(format = ""),
..   ServiceStartDate = col_date(format = ""),
..   PaxName = col_character(),
..   EncryptedName = col_character(),
..   GenderCode = col_character(),
..   birthdateid = col_double(),
..   Age = col_double(),
..   PostalCode = col_double(),
..   BkdClassOfService = col_character(),
..   TrvldClassOfService = col_character(),
..   BookingChannel = col_character(),
..   BaseFareAmt = col_double(),
..   TotalDocAmt = col_double(),
..   UFlyRewardsNumber = col_double(),
..   UflyMemberStatus = col_character(),
..   CardHolder = col_logical(),
..   BookedProduct = col_character(),
..   EnrollDate = col_datetime(format = ""),
..   MarketingFlightNbr = col_character(),
..   MarketingAirlineCode = col_character(),
..   StopoverCode = col_character()
.. )
```

To understand our data and verify the accuracy of our parsed data types, we first examined the structure of our data frame. While the *read_csv* function parsed our numeric and date fields correctly, we decided to convert several of our "character" fields to "factors" to satisfy the parameter requirements for the clustering algorithm. Additionally, we also chose to create several re-binned date fields to facilitate our analyses.

[2]:
```
# Checking for Missing Values
naniar::vis_miss(head(df, 10000), warn_large_data=FALSE)
```



Missing Values Within our data preparation, we examined our data for missing values. Through this process, we discovered a substantial number of missing values among a relatively small number of data fields, including UFly Member Status, Cardholder, and Enrollment Date. To resolve missing UFly data, we recoded missing values to reflect non-member/cardholder status.

To address the (lesser) issue of missing Age and Gender data, where only 6.8% of values were found to be missing, we decided to forgo the imputation process and drop those values as necessary to satisfy the parameter requirements for the clustering algorith.

[3]:
```
# Reformatting Data
suppressWarnings(df <- df %>%
mutate(UflyMemberStatus = case_when(
    UflyMemberStatus=="Standard"~"Standard",
    UflyMemberStatus=="Elite"~"Elite",
```

```
    is.na(UflyMemberStatus)==TRUE~"Non-Member")) %>%
mutate(PNRCreateMonth = month(PNRCreateDate),
       ServiceStartMonth = month(ServiceStartDate)) %>%
mutate(PNRCreateSeason = case_when(
    PNRCreateMonth %in% c(11, 12, 1, 2, 3) ~ "Winter",
    PNRCreateMonth %in% c(4, 5) ~ "Spring",
    PNRCreateMonth %in% c(6, 7, 8) ~ "Summer",
    PNRCreateMonth %in% c(9, 10) ~ "Fall"),
       ServiceStartSeason = case_when(
           ServiceStartMonth %in% c(11, 12, 1, 2, 3) ~ "Winter",
           ServiceStartMonth %in% c(4, 5) ~ "Spring",
           ServiceStartMonth %in% c(6, 7, 8) ~ "Summer",
           ServiceStartMonth %in% c(9, 10) ~ "Fall")) %>%
mutate(EnrollDate = ymd_hms(EnrollDate),
       EnrollDate = dmy(paste0("15-",month(EnrollDate),"-",year(EnrollDate)))))
```

Here, we have implement several of the recoding and variable changes described above. The function, *flight_legs*, shown below allowed us to tidy our data to display a single ticket observation per line by preserving our repeated customer data and combining our trip legs onto a single line.

[4]:
```
flight_legs <- function(dataframe_in) {
    a <- dataframe_in %>%
    select(PNRLocatorID, TicketNum, CouponSeqNbr, ServiceStartCity,
           ServiceEndCity) %>%
    group_by(TicketNum) %>% mutate(trip_max = max(CouponSeqNbr))
    a1 <- a %>% filter(CouponSeqNbr == 1) %>%
    select(PNRLocatorID, TicketNum, ServiceStartCity, ServiceEndCity,
           trip_max) %>%
    rename(City1 = ServiceStartCity, City2 = ServiceEndCity)
    a2 <- a %>% filter(CouponSeqNbr == 2) %>%
    select(PNRLocatorID, TicketNum, ServiceEndCity, trip_max) %>%
    rename(City3 = ServiceEndCity)
    a3 <- a %>% filter(CouponSeqNbr == 3) %>%
    select(PNRLocatorID, TicketNum, ServiceEndCity, trip_max) %>%
    rename(City4 = ServiceEndCity)
    a4 <- a %>% filter(CouponSeqNbr == 4) %>%
    select(PNRLocatorID, TicketNum, ServiceEndCity, trip_max) %>%
    rename(City5 = ServiceEndCity)
    a5 <- a %>% filter(CouponSeqNbr == 5) %>%
    select(PNRLocatorID, TicketNum, ServiceEndCity, trip_max) %>%
    rename(City6 = ServiceEndCity)
    a6 <- a %>% filter(CouponSeqNbr == 6) %>%
    select(PNRLocatorID, TicketNum, ServiceEndCity, trip_max) %>%
    rename(City7 = ServiceEndCity)
    jc <- c("trip_max" = "trip_max", "PNRLocatorID"="PNRLocatorID",
            "TicketNum"="TicketNum")
    j1 <- left_join(a1, a2, by=jc); rm(a1, a2)
```

```
    j2 <- left_join(j1, a3, by=jc); rm(j1, a3)
    j3 <- left_join(j2, a4, by=jc); rm(j2, a4)
    j4 <- left_join(j3, a5, by=jc); rm(j3, a5)
    j5 <- left_join(j4, a6, by=jc) %>%
    select(PNRLocatorID, TicketNum, trip_max,
           City1, City2, City3, City4, City5,
           City6, City7)
    j5 <- j5 %>%
        mutate(mid_dest = case_when(
            trip_max==1~City2, trip_max==2~City2,
            trip_max==3~"Ambiguous", trip_max==4~City3,
            trip_max==5~"Ambiguous", trip_max==6~City4))
    jx <- as_tibble(j5) %>%
    unite("Airport_Sequence", City1:City7, sep = "->", na.rm = TRUE)
    jx <- jx[!duplicated(jx),]
    return(jx)}
```

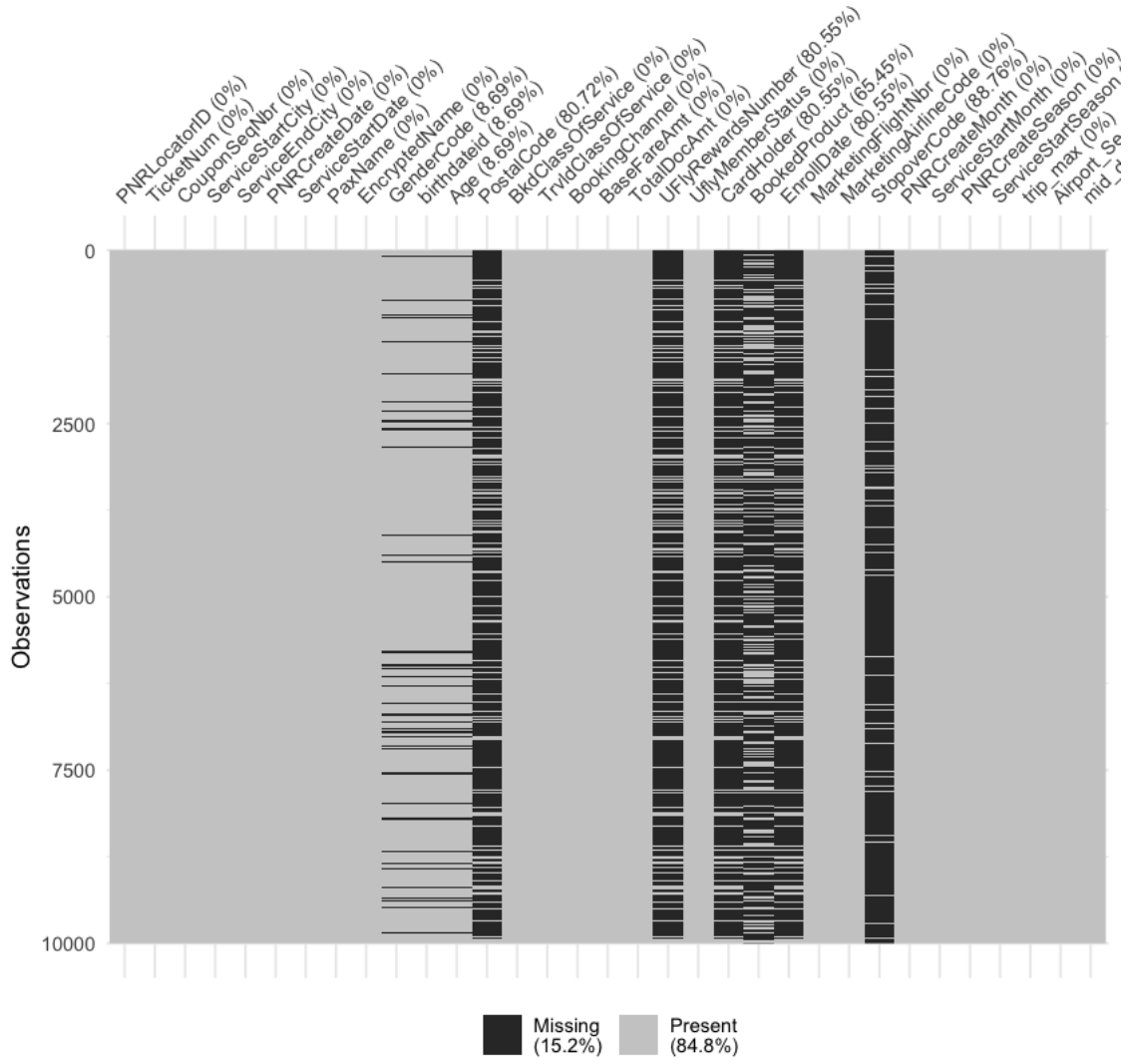
[5]:
```
# Transforming our Data (Note: This step takes some time.)
df <- left_join(df %>% filter(CouponSeqNbr == 1) , flight_legs(df), by =␣
 ↪c("PNRLocatorID", "TicketNum"))
```

[6]:
```
dim(df)
```

1. 1951244 2. 33

Here, we have reduced the volume of data from 3,435,388 rows to 1,951,244 rows by transforming our data to reflect a single customer ticket per row. Within the missing value table below, we have resolved our most pertinent missing UFly data. The remaining missing values were addressed on an ad hoc basis.

[7]:
```
naniar::vis_miss(head(df, 10000), warn_large_data=FALSE)
```



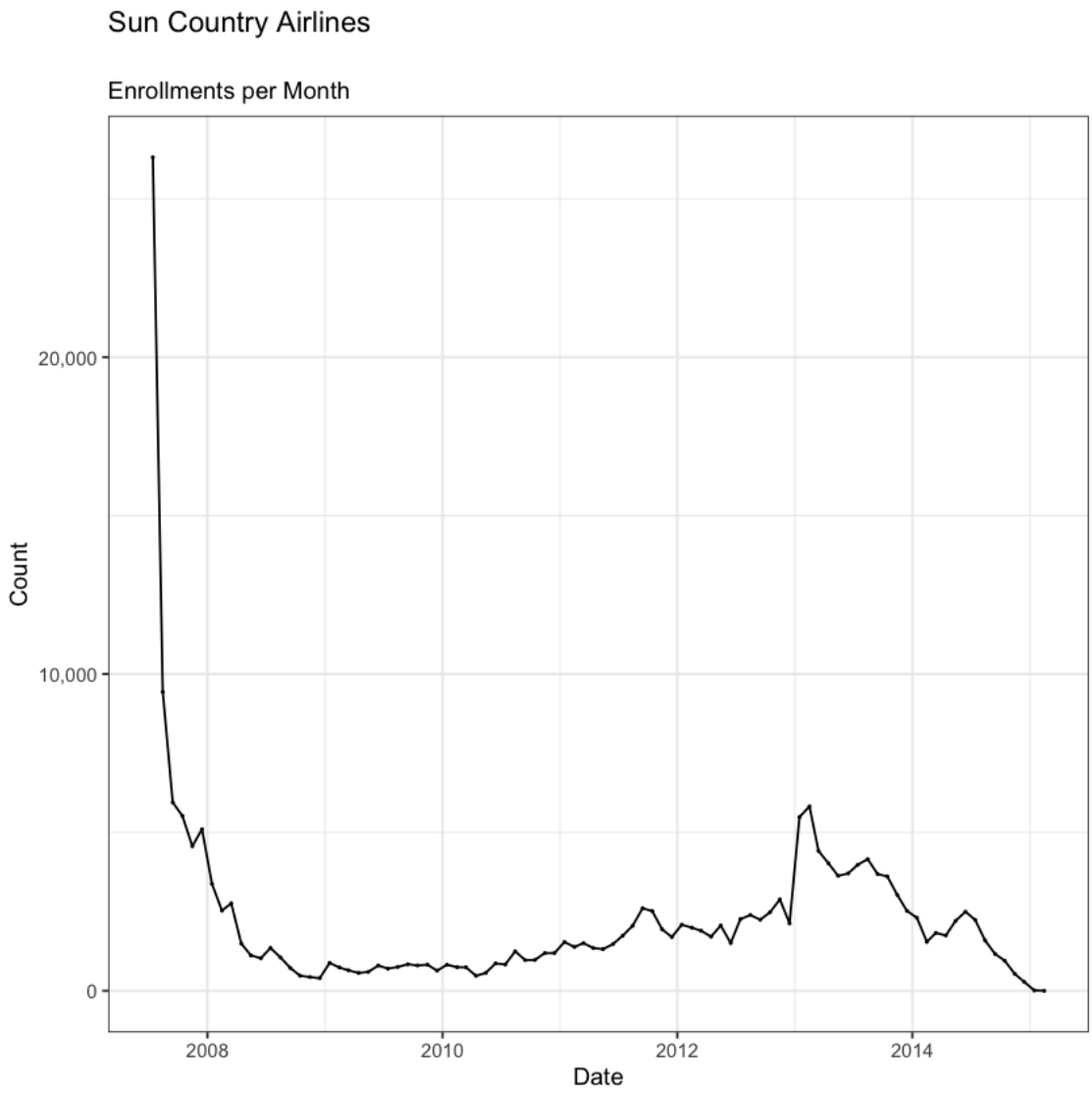
3 Exploratory Analysis

3.1 Descriptive Analysis - Full Data Set

Due to the sizable volume of the given data set, and pursuant computational load, we ultimately approach the clustering process using a sample of our data. However, to make the fullest use of our data, we also perform our initial descriptive analysis using our full transformed (tidy) data set.

3.1.1 Examining UFly Enrollments Per Month

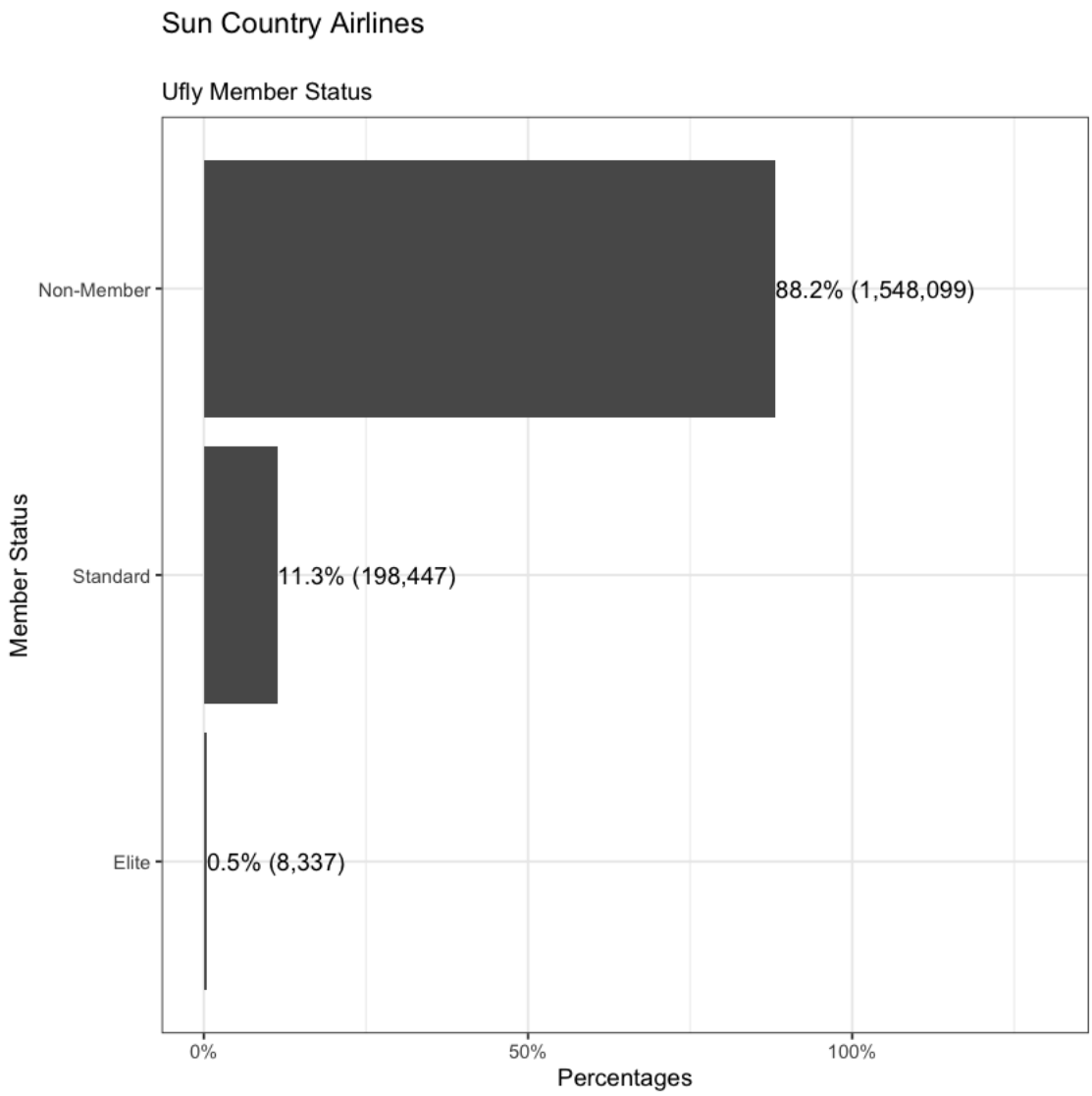
[8]:
```
# Deduplicating
df_dedupe <- df[duplicated(df[c("UFlyRewardsNumber")]),]

# Graphing Ufly Membership Enrollments per Month (ALL DATA)
df2 <- df_dedupe %>% group_by(EnrollDate) %>% summarise(counts = n()) %>%␣
 ↪filter(is.na(EnrollDate)!=TRUE)

ggplot(df2, aes(x=EnrollDate, y=counts)) +
geom_line() + geom_point(size=.2) +
labs(title="Sun Country Airlines\n", subtitle="Enrollments per Month", x="Date",␣
 ↪y="Count") +
scale_y_continuous(labels=scales::comma) +
theme_bw()
```



Here, we observe a considerable decline in UFly enrollment following the introduction of the program in 2007. Following this decline we observe a gradual increase in monthly enrollments that spikes in early 2013. Following this spike, enrollment again declines. This figure suggests we might consider consulting with our marketing team to potentially identify the source of the observed 2013 increase.

3.1.2 Examining UFly Member Percentages by Status

[9]:
```
# Graphing Ufly Member Status Percentages
graph_df <- df_dedupe %>%
    count(UflyMemberStatus) %>%
    mutate(Percentages = n/sum(n))

ggplot(graph_df, aes(x=reorder(UflyMemberStatus, Percentages), y=Percentages)) +
geom_col() + coord_flip() +
labs(title="Sun Country Airlines\n", subtitle="Ufly Member Status", x="Member␣
 ↪Status") +
scale_y_continuous(limits=c(0,1.3), labels = scales::percent) +
geom_text(label=paste0(scales::percent(graph_df$Percentages),
                       " (", scales::comma(graph_df$n), ")"), hjust=-.0005) +
theme_bw()
```

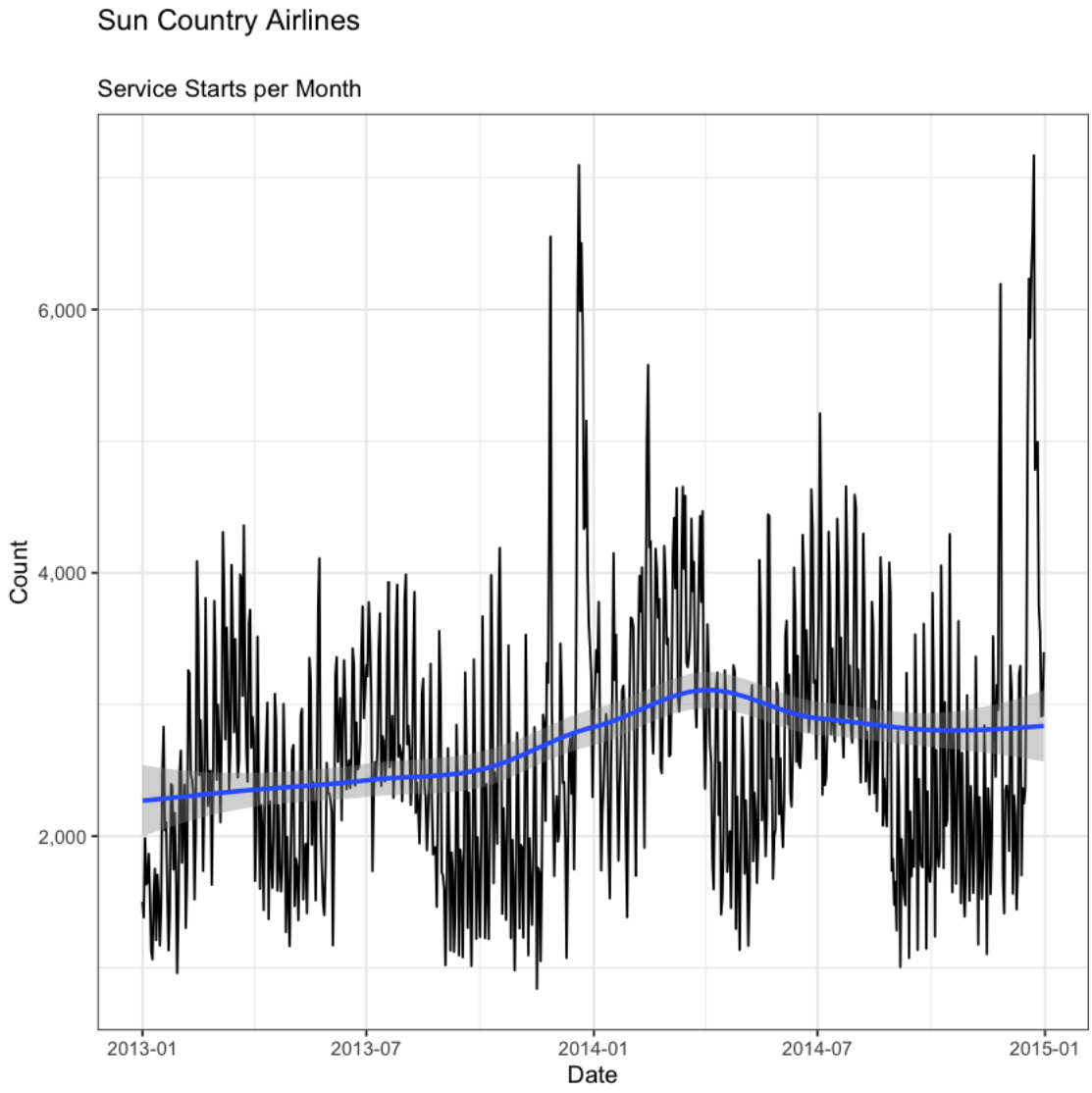


This figure suggests a majority of Sun Country fliers are non-members. Due to the relatively negligible number of observed elite members, we subsequently recoded the standard and elite classes to simply reflect UFly membership. Due to the de-duplicated nature of this figure, we can also intuit the fact that our observed 206,784 members account for a total of unique 403,145 flights.

3.1.3 Examining Customer Flights (Service Starts) per Month

[10]:
```
# Graphing Customer Flights (Service Starts) per Month
df3 <- df %>%
    group_by(ServiceStartDate) %>%
    summarise(counts = n()) %>%
    filter(is.na(ServiceStartDate)!=TRUE)
```

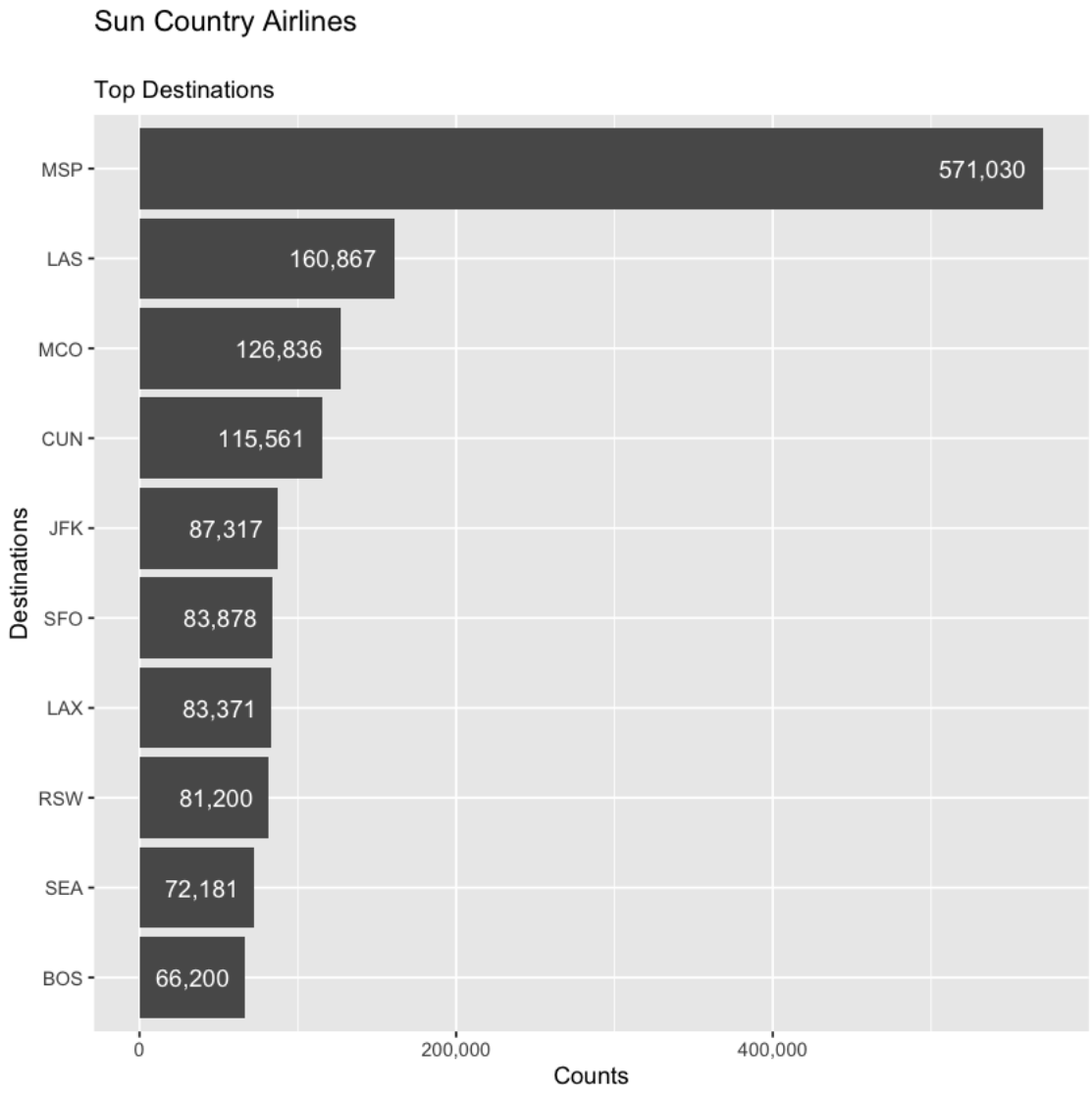
```
ggplot(df3, aes(x=ServiceStartDate, y=counts)) +
geom_line() + labs(title="Sun Country Airlines\n", subtitle="Service Starts per␣
 ↪Month", x="Date", y="Count") +
scale_y_continuous(labels=scales::comma) +
geom_smooth(method = 'loess', formula = 'y ~ x') + theme_bw()

rm(df3)
```



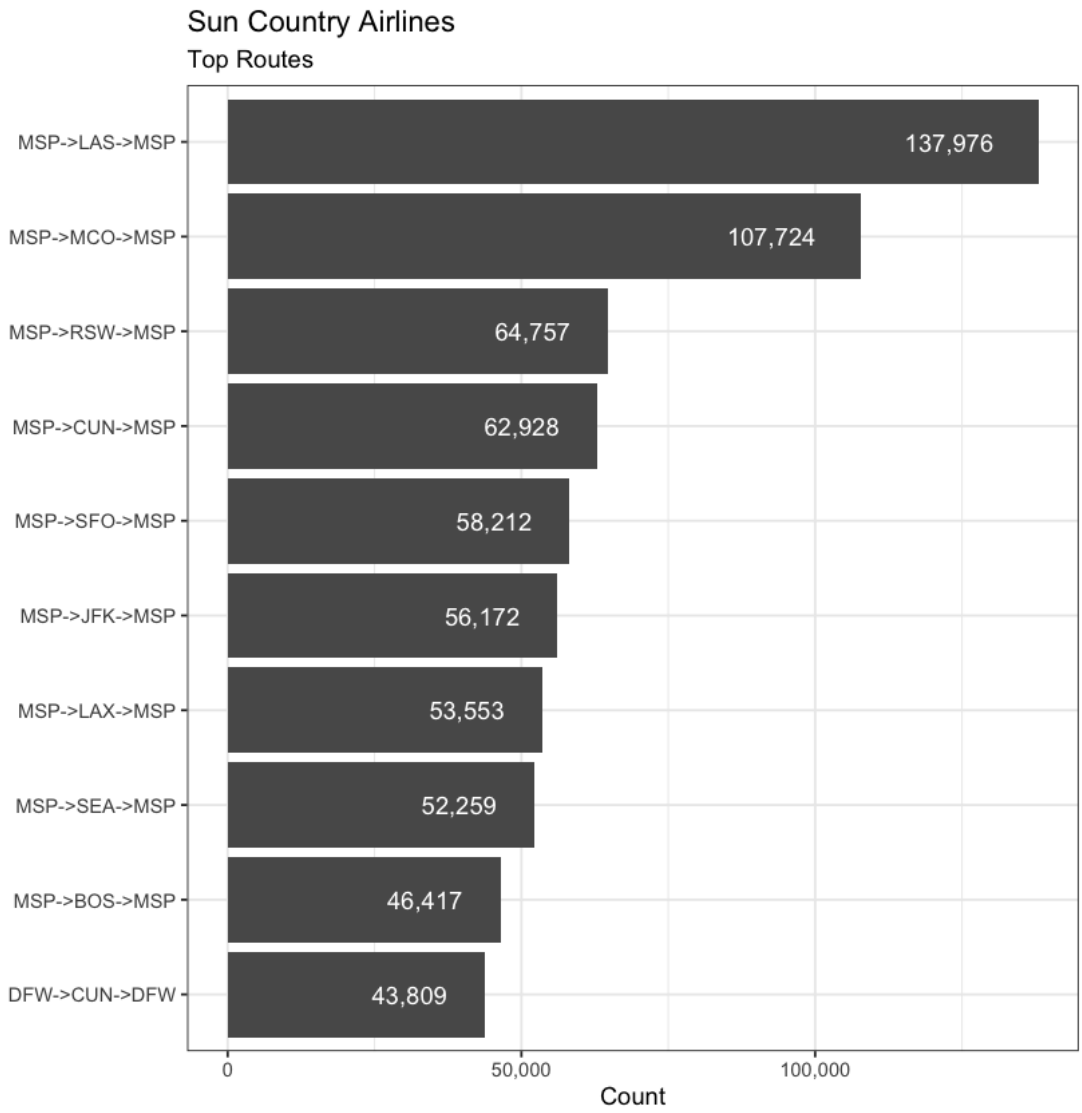
By totaling the number of tickets associated with each date, we can begin to identify a seasonal trend in the total number of flights occurring each day. Here, we can clearly observe significant increases in trips near the Christmas Holiday season, April, and August. It also appears the volume of Sun Country flyers has, on average, improved over the last few years (relative to 2013).

3.1.4 Identifying Top Customer Destinations and Routes

[11]:
```
df4 <- df %>%
    group_by(mid_dest) %>%
    summarize(counts = n()) %>%
    arrange(desc(counts)) %>%
    top_n(10, counts)

ggplot(df4, aes(x=reorder(mid_dest, counts), y=counts)) +
geom_col() + scale_y_continuous(labels=scales::comma) +
labs(title="Sun Country Airlines\n", subtitle="Top Destinations",␣
 ↪x="Destinations", y="Counts") +
geom_text(label = scales::comma(df4$counts), hjust = 1.2, colour = "white") +
coord_flip()
```



[12]:
```
top_routes <- df %>%
    group_by(Airport_Sequence) %>%
    summarise(top_route = n()) %>%
    arrange(desc(top_route)) %>%
    top_n(10, top_route)

ggplot(top_routes, aes(x = reorder(Airport_Sequence, top_route), y = top_route))␣
 ↪+
geom_col() + coord_flip() +
scale_y_continuous(labels=scales::comma) +
labs(x="", y="Count", title="Sun Country Airlines", subtitle="Top Routes") +
geom_text(label = scales::comma(top_routes$top_route), hjust = 1.5, colour =␣
 ↪"white") +
theme_bw()
```



The graphs above confirm the hub status of MSP, and perhaps more importantly, also underline the Southern/Sun Belt orientation of Sun Country's current offerings. It is also important to note the presence of New York, Seattle, Boston, and Los Angeles destinations. (While it is likely that a sizable portion of those customers are business travelers, these customers could also be tourists.)

3.1.5 Visualizing Sun Country Routes

[21]:
```
# Import ALL Data
suppressWarnings(suppressMessages(
dt <- readr::read_csv(file = "~/Downloads/HW 2/SunCountry.csv", n_max = 12000)))

# Basic Map
```

```
suppressPackageStartupMessages(library(maps))
suppressPackageStartupMessages(library(geosphere))

xlim <- c(-171.738281, -56.601563)
ylim <- c(12.039321, 71.856229)
map("world", col="#191919", fill=TRUE, bg="#000000",
    lwd=0.05, xlim=xlim, ylim=ylim)

# Colors
pal <- colorRampPalette(c("#f2f2f2", "black"))
pal <- colorRampPalette(c("#f2f2f2", "red"))
colors <- pal(100)

dt <- dt %>%
select(ServiceStartCity, ServiceEndCity) %>%
group_by(ServiceStartCity, ServiceEndCity) %>% summarise(cnt = n()) %>%
rename(airport1 = ServiceStartCity, airport2 = ServiceEndCity) %>%
mutate(airline = "Sun Country")

airports <- read.csv("~/Desktop/airports.dat.csv") %>%
select(GKA, X.6.081689834590001, X145.391998291) %>%
rename(iata = GKA, lat = X.6.081689834590001, long = X145.391998291)

dt <- dt[(dt$airport1 %in% airports$iata & dt$airport2 %in% airports$iata),]

fsub <- dt[dt$airline == "Sun Country",]
fsub <- fsub[order(fsub$cnt),]
maxcnt <- max(fsub$cnt)
for (j in 1:length(fsub$airline)) {
        air1 <- airports[airports$iata == fsub[j,]$airport1,]
        air2 <- airports[airports$iata == fsub[j,]$airport2,]
        inter <- gcIntermediate(c(air1[1,]$long, air1[1,]$lat),
                                c(air2[1,]$long, air2[1,]$lat), n=100,␣
 ↪addStartEnd=TRUE)
        colindex <- round( (fsub[j,]$cnt / maxcnt) * length(colors) )
        lines(inter, col=colors[colindex], lwd=0.8)
        title(main = "Sun Country Flights", col.main = "White")
}

# Code adapted from https://flowingdata.com/2011/05/11/
 ↪how-to-map-connections-with-great-circles/


rm(dt)
```

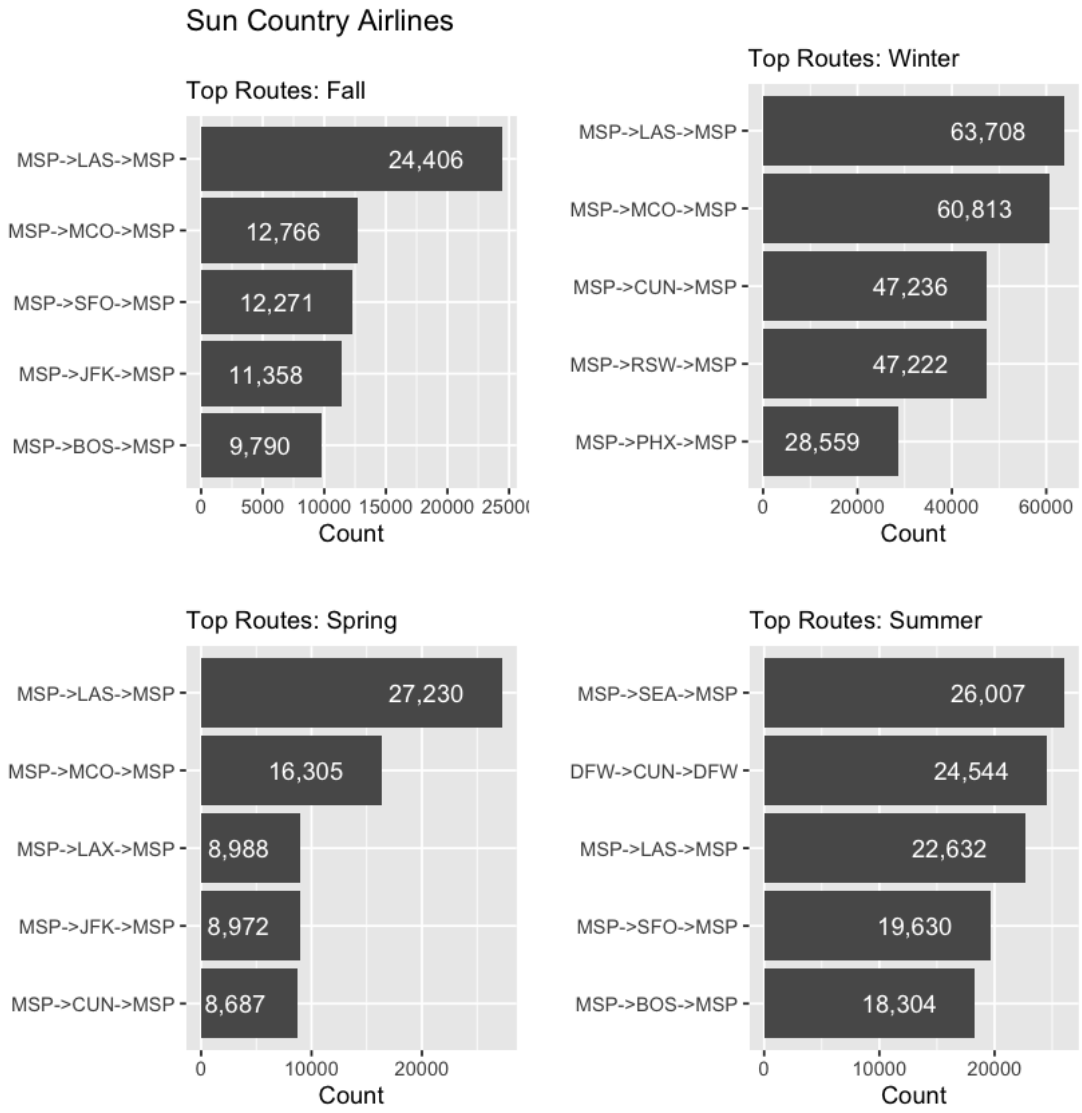
Visualizing the underlying data, it is clear that Sun Country ferries passengers to a variety of of destinations in North America, Central America, the Caribbean, and the Pacific. In addition to Minneapolis-Saint Paul, the airport at Dallas Fort Worth also appears to function as a hub for Sun Country.

3.1.6 Examining Customer Routes and Flights by Season

[22]:
```
df_msp_start <- df

# SUBSETTING BY SEASON
top_routes_fall <- head(df_msp_start %>% group_by(Airport_Sequence,␣
 ↪ServiceStartSeason) %>%
```

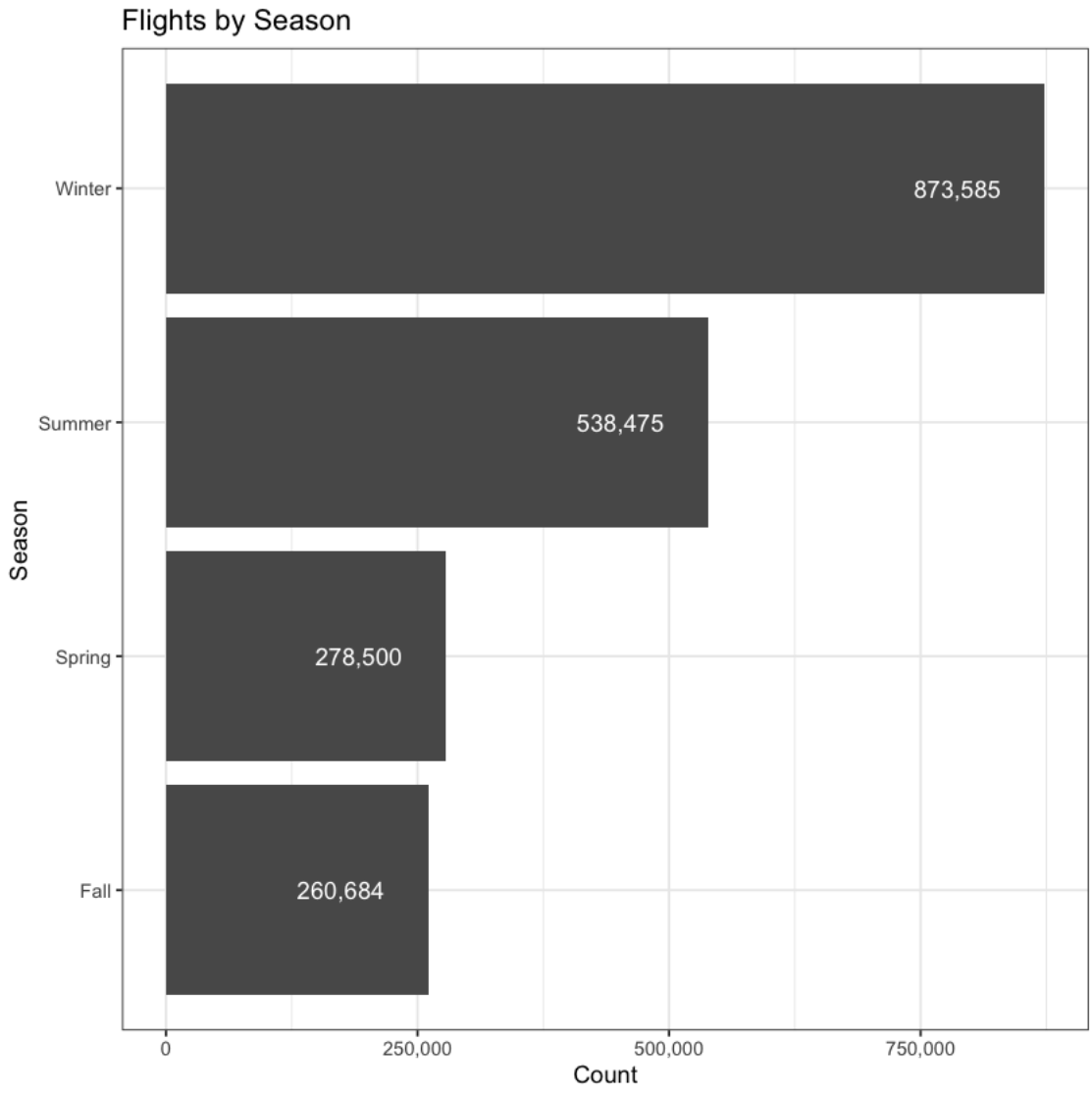
```
                        summarise(top_route = n()) %>% arrange(desc(top_route))␣
 ↪%>%
                        filter(ServiceStartSeason == "Fall"), 5)

top_routes_winter <- head(df_msp_start %>% group_by(Airport_Sequence,␣
 ↪ServiceStartSeason) %>%
                          summarise(top_route = n()) %>%␣
 ↪arrange(desc(top_route)) %>%
                          filter(ServiceStartSeason == "Winter"), 5)

top_routes_spring <- head(df_msp_start %>% group_by(Airport_Sequence,␣
 ↪ServiceStartSeason) %>%
                          summarise(top_route = n()) %>%␣
 ↪arrange(desc(top_route)) %>%
                          filter(ServiceStartSeason == "Spring"), 5)

top_routes_summer <- head(df_msp_start %>% group_by(Airport_Sequence,␣
 ↪ServiceStartSeason) %>%
                          summarise(top_route = n()) %>%␣
 ↪arrange(desc(top_route)) %>%
                          filter(ServiceStartSeason == "Summer"), 5)

# SEASON GGPLOTS
g1 <- ggplot(top_routes_fall, aes(x = reorder(Airport_Sequence, top_route), y =␣
 ↪top_route)) +
geom_col() + coord_flip() +
labs(x = "", y="Count", title="Sun Country Airlines\n", subtitle="Top Routes:␣
 ↪Fall") +
geom_text(label = scales::comma(top_routes_fall$top_route), hjust = 1.5, colour␣
 ↪= "white")

g2 <- ggplot(top_routes_winter, aes(x = reorder(Airport_Sequence, top_route), y␣
 ↪= top_route)) +
geom_col() + coord_flip() +
labs(x="", y="Count", title="", subtitle="Top Routes: Winter") +
geom_text(label = scales::comma(top_routes_winter$top_route), hjust = 1.5,␣
 ↪colour = "white")

g3 <- ggplot(top_routes_spring, aes(x = reorder(Airport_Sequence, top_route), y␣
 ↪= top_route)) +
geom_col() + coord_flip() +
labs(x="", y="Count", title="", subtitle="Top Routes: Spring") +
geom_text(label = scales::comma(top_routes_spring$top_route), hjust = 1.5,␣
 ↪colour = "white")
```

```
g4 <- ggplot(top_routes_summer, aes(x = reorder(Airport_Sequence, top_route), y␣
 ↪= top_route)) +
geom_col() + coord_flip() +
labs(x="", y="Count", title="", subtitle="Top Routes: Summer") +
geom_text(label = scales::comma(top_routes_summer$top_route), hjust = 1.5,␣
 ↪colour = "white")

cowplot::plot_grid(g1, g2, g3, g4)
```



We can also examine the distribution of Sun Country's flights throughout the year. This graph suggests a substantial number of Sun Country's passengers choose to depart from MSP during the winter to visit warmer climes. Overall, Winter and Summer appear to account for the bulk of Sun Country's passengers.

[23]:
```
df_season <- df_msp_start %>% group_by(ServiceStartSeason) %>%
summarise(flights_by_season = n())

ggplot(df_season, aes(x = ServiceStartSeason, y = flights_by_season)) +
geom_col() + coord_flip() +  theme_bw() +
labs(x="Season",y="Count",title="Flights by Season") +
scale_y_continuous(label=scales::comma) +
geom_text(label = scales::comma(df_season$flights_by_season), colour = "white",␣
 ↪hjust = 1.5)
```



While Winter and Summer appear to be the most popular travel months for Sun Country Passengers, it should be noted that Spring and Fall are the shortest seasons for most MSP residents.

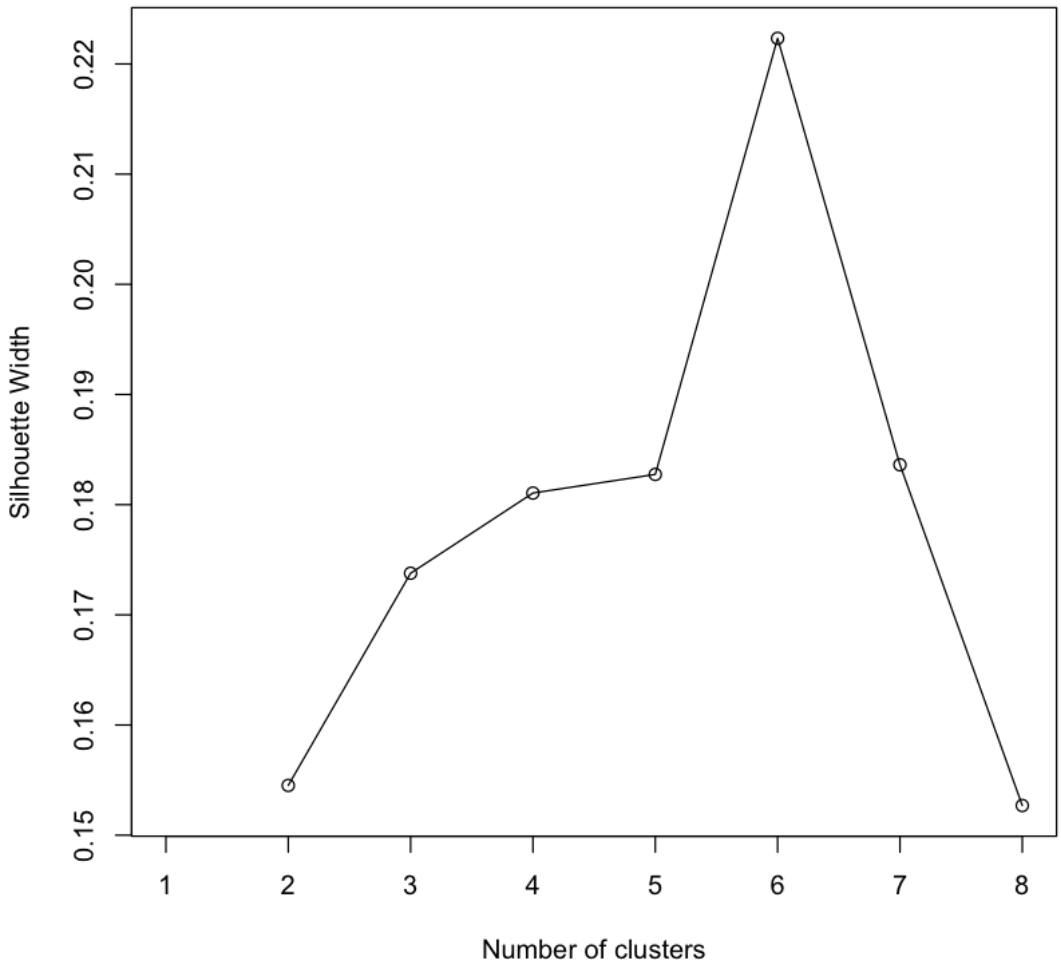
4 Clustering Strategies Towards Customer Segmentation

In reviewing the data set, we observed that a majority of the observed trips were initiated for MSP. On that basis we decided to focus our attention on flights departing MSP. In consideration of technology constraints, we took a random sample of 20,000 rows from the subset of the data. To minimize the time necessary to perform our computations, we exported the cluster results, and imported the outputs into the notebook. If one were to randomly sample the data set, they might obtain different results.

[3]:
```
suppressPackageStartupMessages(library(tidyverse))
suppressPackageStartupMessages(library(cluster))
suppressPackageStartupMessages(library(factoextra))
suppressPackageStartupMessages(library(corrplot))
suppressPackageStartupMessages(library(PerformanceAnalytics))
```

[4]:
```
## Set the working directory
## This can be changed based on where you have your data

data.dir = "~/Documents/GitHub/msba6410_exploratory_analytics/HW2/"

## Set the file name
data.file = "SunCountry_MSPDepartures_20kSample_wCluster.csv"
cluster_data = read.csv(paste(data.dir, data.file, sep = ""))
```

[14]:
```
gower = daisy(cluster_data, metric = "gower")

gower_mat = as.matrix(gower)

#find similar characteristics
#data_feats_sample[which(gower_mat == min(gower_mat[gower_mat !=␣
 ↪min(gower_mat)]), arr.ind = TRUE)[1, ], ]

#find not similar characteristics
#data_feats_sample[which(gower_mat == min(gower_mat[gower_mat !=␣
 ↪max(gower_mat)]), arr.ind = TRUE)[1, ], ]

#determine a number of clusters ----------------
sil_width = c(NA)

plot(1:8, sil_width,
     xlab = "Number of clusters",
     ylab = "Silhouette Width")
lines(1:8, sil_width)
```



Examining the results of our plot of silhouette widths suggests we should select k=6 to produce the best clustering results.

[15]:
```
#get a summary of the clusters ----------------
k = 6
pam_fit = pam(gower, diss = TRUE, k)
pam_results = cluster_data %>%
  mutate(cluster = pam_fit$clustering) %>%
  group_by(cluster) %>%
  do(the_summary = summary(.))
pam_results$the_summary
```

```
[[1]]
```

```
      X          ServiceStartCity ServiceEndCity GenderCode
 Min.   :    1   MSP:4001         MCO    : 806   F: 377
 1st Qu.: 4793                    JFK    : 330   M:3624
 Median : 9884                    SFO    : 319   U:   0
 Mean   : 9907                    LAX    : 297
 3rd Qu.:14869                    BOS    : 264
 Max.   :19994                    LAS    : 244
                                  (Other):1741
             BookingChannel    TimeGap          TripMonth         Age
 Outside Booking     :3239   Min.   :  0.00   December: 876   Min.   :  0.00
 Reservations Booking: 334   1st Qu.: 17.00   July    : 348   1st Qu.: 21.00
 SCA Website Booking : 224   Median : 40.00   June    : 338   Median : 31.00
 Tour Operator Portal: 123   Mean   : 58.16   August  : 330   Mean   : 32.67
 SY Vacation         :  61   3rd Qu.: 82.00   May     : 325   3rd Qu.: 44.00
 MSP                 :  16   Max.   :482.00   April   : 309   Max.   :108.00
 (Other)             :   4                    (Other) :1475
   UflyMember      CardHolderFlag        cluster
 Min.   :0.00000   Min.   :0.000000   Min.   :1
 1st Qu.:0.00000   1st Qu.:0.000000   1st Qu.:1
 Median :0.00000   Median :0.000000   Median :1
 Mean   :0.05499   Mean   :0.001999   Mean   :1
 3rd Qu.:0.00000   3rd Qu.:0.000000   3rd Qu.:1
 Max.   :1.00000   Max.   :1.000000   Max.   :1


[[2]]
      X          ServiceStartCity ServiceEndCity GenderCode
 Min.   :    2   MSP:1902         LAS    :335    F:  94
 1st Qu.: 5002                    LAX    :141    M:1808
 Median : 9928                    JFK    :131    U:   0
 Mean   : 9941                    SFO    :131
 3rd Qu.:15004                    BOS    :123
 Max.   :19992                    MCO    :106
                                  (Other):935
             BookingChannel    TimeGap          TripMonth        Age
 SCA Website Booking :1336   Min.   :  0.00   March   :300   Min.   :  0.00
 Outside Booking     : 333   1st Qu.: 19.00   November:200   1st Qu.: 36.00
 Reservations Booking: 129   Median : 45.00   October :182   Median : 48.00
 SY Vacation         :  71   Mean   : 61.89   February:161   Mean   : 46.11
 Tour Operator Portal:  25   3rd Qu.: 87.00   August  :150   3rd Qu.: 59.00
 FCM                 :   8   Max.   :373.00   January :150   Max.   :111.00
 (Other)             :   0                    (Other) :759
   UflyMember CardHolderFlag       cluster
 Min.   :1    Min.   :0.00000   Min.   :2
 1st Qu.:1    1st Qu.:0.00000   1st Qu.:2
 Median :1    Median :0.00000   Median :2
 Mean   :1    Mean   :0.04942   Mean   :2
 3rd Qu.:1    3rd Qu.:0.00000   3rd Qu.:2
```

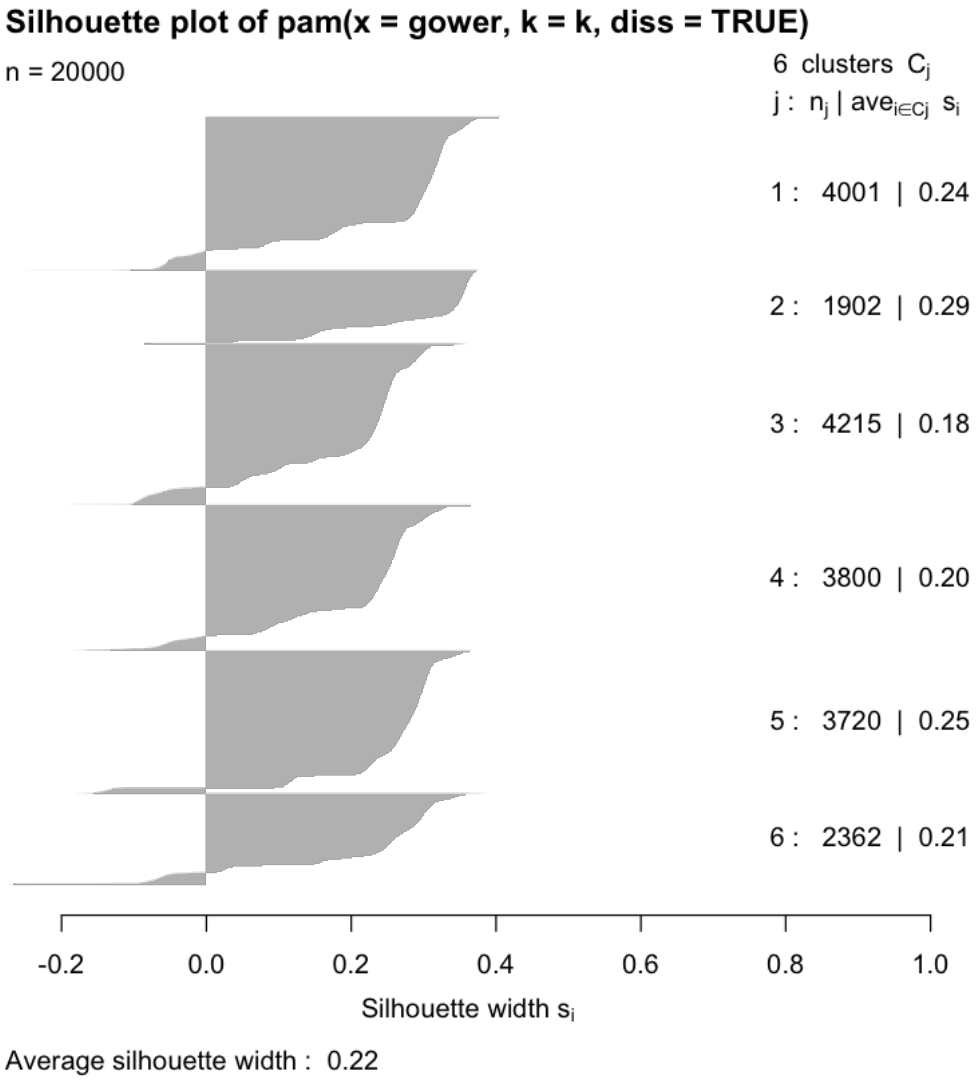
```
 Max.   :1    Max.   :1.00000   Max.   :2


[[3]]
      X         ServiceStartCity ServiceEndCity GenderCode
 Min.   :    3   MSP:4215         LAS    : 932   F:3951
 1st Qu.: 4820                    JFK    : 326   M: 264
 Median : 9982                    SEA    : 294   U:   0
 Mean   : 9951                    SFO    : 285
 3rd Qu.:14936                    CUN    : 276
 Max.   :20000                    LAX    : 253
                                  (Other):1849
             BookingChannel    TimeGap          TripMonth         Age
 Outside Booking     :3357   Min.   :  0.00   February: 964   Min.   :  0.00
 Reservations Booking: 258   1st Qu.: 22.00   July    : 370   1st Qu.: 29.00
 SCA Website Booking : 200   Median : 46.00   June    : 357   Median : 43.00
 SY Vacation         : 195   Mean   : 61.91   October : 337   Mean   : 42.54
 Tour Operator Portal: 189   3rd Qu.: 86.50   August  : 336   3rd Qu.: 55.00
 MSP                 :  13   Max.   :507.00   January : 334   Max.   :114.00
 (Other)             :   3                    (Other) :1517
   UflyMember      CardHolderFlag        cluster
 Min.   :0.00000   Min.   :0.000000   Min.   :3
 1st Qu.:0.00000   1st Qu.:0.000000   1st Qu.:3
 Median :0.00000   Median :0.000000   Median :3
 Mean   :0.05504   Mean   :0.001424   Mean   :3
 3rd Qu.:0.00000   3rd Qu.:0.000000   3rd Qu.:3
 Max.   :1.00000   Max.   :1.000000   Max.   :3


[[4]]
      X         ServiceStartCity ServiceEndCity GenderCode
 Min.   :    4   MSP:3800         MCO    : 666   F:3760
 1st Qu.: 5218                    LAX    : 276   M:  39
 Median :10167                    SEA    : 257   U:   1
 Mean   :10106                    JFK    : 253
 3rd Qu.:15025                    SFO    : 224
 Max.   :19988                    CUN    : 215
                                  (Other):1909
             BookingChannel    TimeGap          TripMonth         Age
 SCA Website Booking :2842   Min.   :  0.00   March  :1054   Min.   :  0.00
 Reservations Booking: 386   1st Qu.: 24.75   July   : 316   1st Qu.: 18.00
 Outside Booking     : 251   Median : 53.00   June   : 294   Median : 32.00
 Tour Operator Portal: 180   Mean   : 70.57   August : 288   Mean   : 34.04
 SY Vacation         : 119   3rd Qu.:100.00   January: 281   3rd Qu.: 49.00
 MSP                 :  14   Max.   :514.00   October: 281   Max.   :100.00
 (Other)             :   8                    (Other):1286
   UflyMember    CardHolderFlag        cluster
 Min.   :0.000   Min.   :0.000000   Min.   :4
```

```
 1st Qu.:0.000   1st Qu.:0.000000   1st Qu.:4
 Median :0.000   Median :0.000000   Median :4
 Mean   :0.045   Mean   :0.001053   Mean   :4
 3rd Qu.:0.000   3rd Qu.:0.000000   3rd Qu.:4
 Max.   :1.000   Max.   :1.000000   Max.   :4


[[5]]
       X          ServiceStartCity ServiceEndCity GenderCode
 Min.   :   13   MSP:3720         LAS    : 817   F: 155
 1st Qu.: 5138                    RSW    : 236   M:3565
 Median :10024                    LAX    : 226   U:   0
 Mean   :10058                    JFK    : 215
 3rd Qu.:15102                    CUN    : 204
 Max.   :19995                    PHX    : 189
                                  (Other):1833
              BookingChannel    TimeGap         TripMonth         Age
 SCA Website Booking :2587   Min.   :  0.0   March   : 813   Min.   : 0.00
 Outside Booking     : 335   1st Qu.: 18.0   January : 365   1st Qu.:32.00
 Reservations Booking: 332   Median : 43.0   February: 347   Median :47.00
 SY Vacation         : 239   Mean   : 62.5   October : 313   Mean   :44.34
 Tour Operator Portal: 201   3rd Qu.: 90.0   July    : 278   3rd Qu.:57.00
 MSP                 :  22   Max.   :475.0   April   : 269   Max.   :92.00
 (Other)             :   4                   (Other) :1335
   UflyMember CardHolderFlag    cluster
 Min.   :0    Min.   :0      Min.   :5
 1st Qu.:0    1st Qu.:0      1st Qu.:5
 Median :0    Median :0      Median :5
 Mean   :0    Mean   :0      Mean   :5
 3rd Qu.:0    3rd Qu.:0      3rd Qu.:5
 Max.   :0    Max.   :0      Max.   :5


[[6]]
       X          ServiceStartCity ServiceEndCity GenderCode
 Min.   :   16   MSP:2362         RSW    : 546   F:2142
 1st Qu.: 5032                    SFO    : 162   M: 220
 Median :10026                    JFK    : 159   U:   0
 Mean   :10036                    LAX    : 154
 3rd Qu.:15106                    PHX    : 138
 Max.   :19989                    SEA    : 129
                                  (Other):1074
              BookingChannel    TimeGap          TripMonth        Age
 SCA Website Booking :1836   Min.   :  0.00   December:533   Min.   :  0.00
 Outside Booking     : 213   1st Qu.: 24.00   February:231   1st Qu.: 45.00
 Reservations Booking: 207   Median : 50.00   October :231   Median : 55.00
 SY Vacation         :  64   Mean   : 65.57   January :204   Mean   : 51.72
 Tour Operator Portal:  30   3rd Qu.: 90.00   November:191   3rd Qu.: 63.00
```

```
FCM                :   9   Max.   :650.00   May      :154   Max.   :100.00
(Other)            :   3                    (Other) :818
  UflyMember    CardHolderFlag       cluster
Min.   :0.000   Min.   :0.00000   Min.   :6
1st Qu.:1.000   1st Qu.:0.00000   1st Qu.:6
Median :1.000   Median :0.00000   Median :6
Mean   :0.876   Mean   :0.04996   Mean   :6
3rd Qu.:1.000   3rd Qu.:0.00000   3rd Qu.:6
Max.   :1.000   Max.   :1.00000   Max.   :6
```

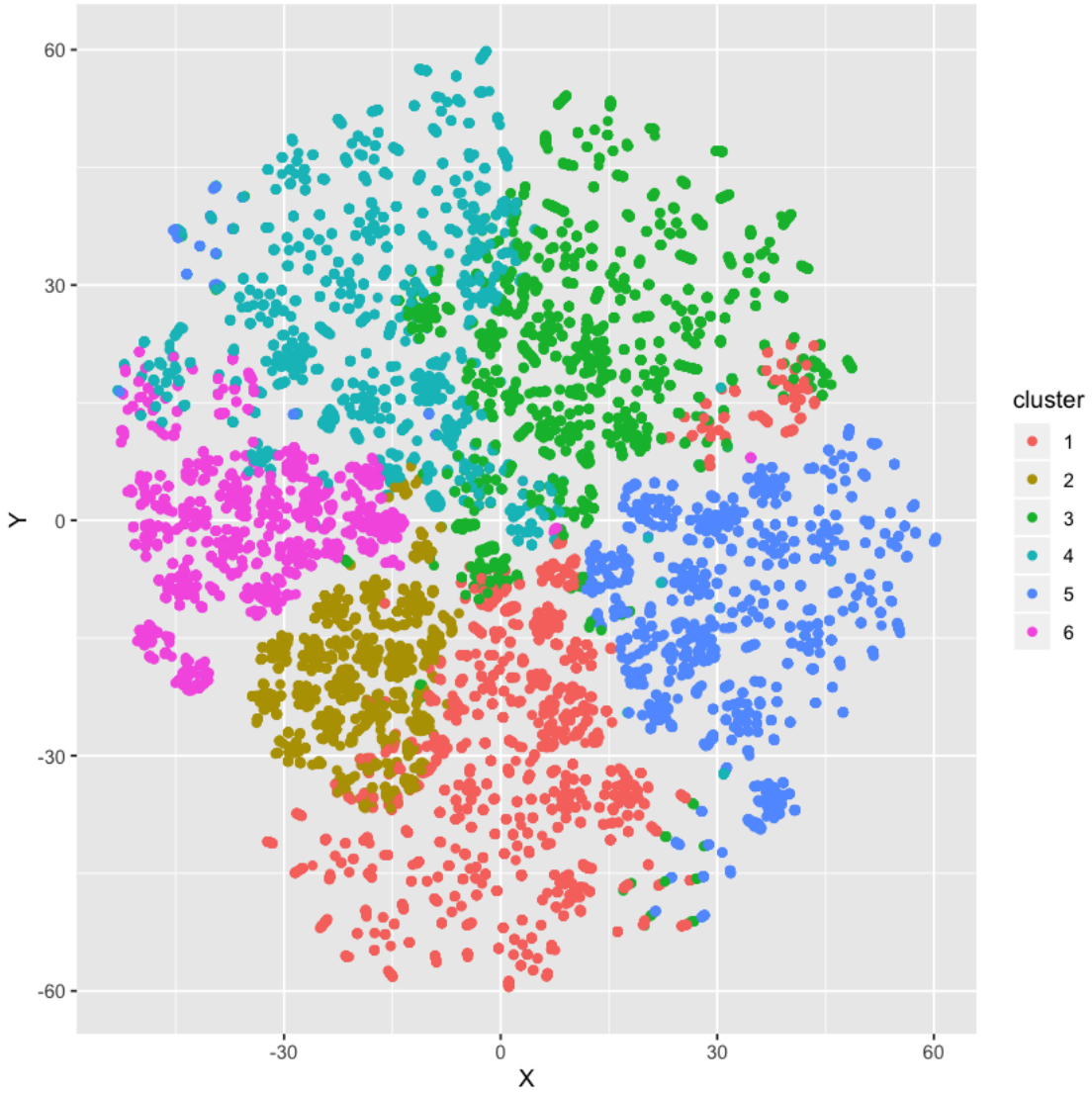
N.B: Summary results are examined in full detail in Section 4.1.

[9]:
```
plot(pam_fit)
```



The silhouette plot (above) provides a graphical representation of the consistency of our clusters. Our results, suggest clusters 1 and 3 are our largest clusters while cluster 2 is the smallest. Overall, it appears our clusters are roughly equivalent in consistency. Several of our clusters - namely, clusters 1, 3, 4, 5, and 6 - possess data that could belong to other clusters. By contrast, it is worth noting that cluster 2 appears to possess the highest level of consistency.

[13]:
```
library("Rtsne")
# Using T-SNE to construct lower dimensional (2D) emedding of our data.

tsne_obj <- Rtsne(gower_mat, is_distance = TRUE)

tsne_data <- tsne_obj$Y %>%
  data.frame() %>%
  setNames(c("X", "Y")) %>%
  mutate(cluster = factor(pam_fit$clustering),
         name = cluster_data$TicketNum)

ggplot(aes(x = X, y = Y), data = tsne_data) +
  geom_point(aes(color = cluster))
```



The T-Distributed Stochastic Neighboring Entities (t-SNE) plot shown above allows us to reduce and thereby visualize the results of our clustering in two dimensions. Our results clearly display 6 distinct (but not entirely separate) clusters. The graph appears to confirm the findings of our earlier silhouette plot. As we would expect, cluster 2 is our most compact cluster, while clusters 1, 3, 4, 5, and 6 are somewhat intermingled

4.1 Characteristics of Fliers

[3]:
```
## Make a copy of the data - now that we have it loaded
## We don't want to screw up any transformations
suncountry = cluster_data
```

```
## View the first few rows to see the structure of the data

head(suncountry)
```

id	ServiceStartCity	ServiceEndCity	GenderCode	BookingChannel	TimeGap	TripMonth	Age
1	MSP	SAN	F	Outside Booking	27	December	21
2	MSP	DFW	M	SCA Website Booking	20	October	45
3	MSP	SEA	F	Outside Booking	63	December	63
4	MSP	SEA	F	SCA Website Booking	63	August	39
5	MSP	ANC	F	Outside Booking	137	July	25
6	MSP	CUN	M	Outside Booking	24	December	21

[4]:
```
## View some summary statistics of the clustered dataset

summary(suncountry)
```

```
      id         ServiceStartCity ServiceEndCity  GenderCode
Min.   :    1   MSP:20000        LAS     : 2550   F:10479
1st Qu.: 5001                    MCO     : 1939   M: 9520
Median :10000                    JFK     : 1414   U:    1
Mean   :10000                    LAX     : 1347
3rd Qu.:15000                    SFO     : 1308
Max.   :20000                    RSW     : 1293
                                 (Other):10149
             BookingChannel    TimeGap          TripMonth         Age
SCA Website Booking :9025   Min.   :  0.00   March    :2726   Min.   :  0.00
Outside Booking     :7728   1st Qu.: 20.00   February:2074   1st Qu.: 26.00
Reservations Booking:1646   Median : 46.00   December:2039   Median : 42.00
SY Vacation         : 749   Mean   : 63.35   January :1632   Mean   : 40.71
Tour Operator Portal: 748   3rd Qu.: 89.00   October :1631   3rd Qu.: 55.00
MSP                 :  66   Max.   :650.00   July    :1596   Max.   :114.00
(Other)             :  38                    (Other) :8302
  UflyMember     CardHolderFlag      cluster
Min.   :0.0000   Min.   :0.0000   Min.   :1.000
1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:2.000
Median :0.0000   Median :0.0000   Median :3.000
Mean   :0.2297   Mean   :0.0115   Mean   :3.421
3rd Qu.:0.0000   3rd Qu.:0.0000   3rd Qu.:5.000
Max.   :1.0000   Max.   :1.0000   Max.   :6.000
```

[5]:
```
## Generate some quick tables to view categorical variables and their␣
 ↪distributions

table(suncountry$ServiceStartCity)
table(suncountry$ServiceEndCity)
table(suncountry$TripMonth)
```

```
table(suncountry$GenderCode)
table(suncountry$UflyMember)
table(suncountry$CardHolderFlag)
table(suncountry$cluster)
```

```
  MSP
20000
```

```
 ANC  BOS  CUN  CZM  DCA  DFW  GRB  HRL  HUX  IFP  JFK  LAN  LAS  LAX  LIR  MBJ
 249 1121 1048  112  614  706    3  386   35   10 1414  303 2550 1347   68  132
 MCO  MDW  MIA  MZT  PHX  PNS  PSP  PUJ  PVR  RSW  SAN  SEA  SFO  SJD  SJU  STT
1939  556  217   81  871    3  526  209  368 1293  642 1164 1308  138   58   78
 SXM  TPA  ZIH
  72  283   96
```

```
    April    August  December  February   January      July      June     March
     1466      1468      2039      2074      1632      1596      1493      2726
      May  November   October September
     1416      1398      1631      1061
```

```
    F     M     U
10479  9520     1
```

```
    0     1
15406  4594
```

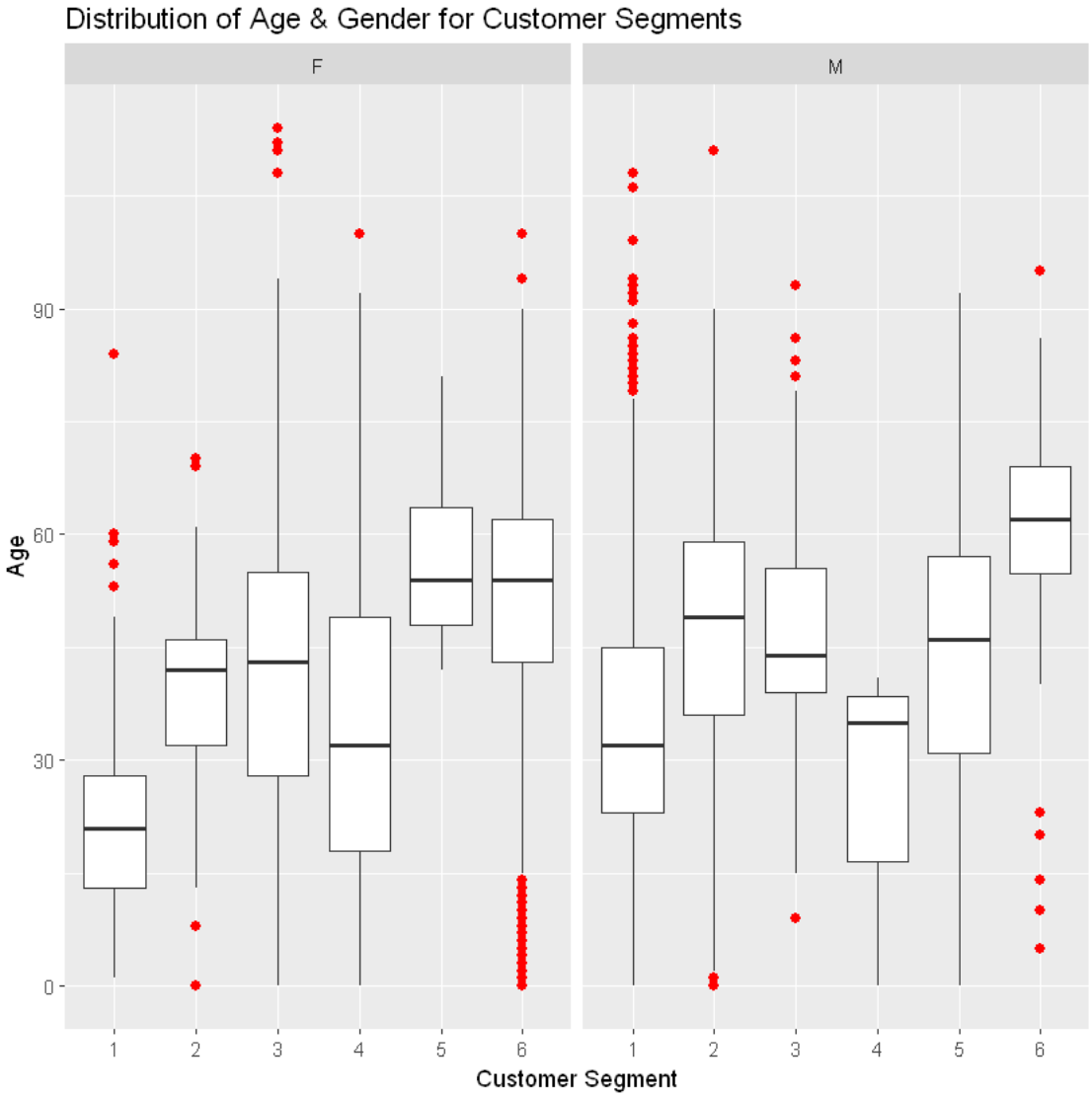
```
    0     1
19770   230
```

```
   1    2    3    4    5    6
4001 1905 4212 3800 3720 2362
```

[6]:
```
## Taking one extra step here to filter out the one row with U for the Gender␣
 ↪Analysis
suncountry <- suncountry %>% filter(GenderCode != 'U')

## Set cluster as the customer segment and also make it at factor
suncountry$cluster <- as.factor(suncountry$cluster)
```

[7]:
```
## Distributions for Age based on the customer segment - we see a lot of outliers
ggplot(suncountry, aes(x = cluster, y = Age)) +
        geom_boxplot(outlier.colour="red", outlier.size=2) + facet_grid(. ~␣
 ↪GenderCode) +
```

```
      labs(x="Customer Segment", title = "Distribution of Age & Gender for␣
↪Customer Segments")
```



This figure provides an insight into the age distributions of our customer clients among males and females. Overall, clusters one and four are the two youngest cohorts on average across both sexes. By contrast, cluster six is, on average, the oldest cluster for both sexes. For most of the identified clusters, males are, on average, somewhat older than females within their respective clusters. Cluster five represents a reversal of this trend insofar as women are older than men.

[8]:
```
## Distributions of Final Destinations for Each Customer Segment
customer_segment_final_destination <- suncountry %>%
        ## Group by the cluster and end city
        group_by(cluster, ServiceEndCity) %>%
```

```
        ## Count the number of times each end city appears
        summarise(id = n()) %>%
        ## Grab the top 10 final destinations for each to make for easier␣
 ↪plotting
        top_n(10)

## Now we pivot the table a bit so that we organize by the final destinations␣
 ↪for each segment - we'll use this in the next plot
final_dest_total_trips <- customer_segment_final_destination %>%
        group_by(ServiceEndCity, cluster) %>%
        summarise(total_trips = sum(id))
```

```
Selecting by id
```

[9]:
```
## Generate a table showing the grand total for each final desination
sorted_table <- final_dest_total_trips %>%
        group_by(ServiceEndCity) %>%
        summarise(grand_total = sum(total_trips)) %>%
        arrange(desc(grand_total))

levels = sorted_table$ServiceEndCity

## Re-arranging the sort order for our final destionation to help generate a␣
 ↪more understandable graphic
final_dest_total_trips$ServiceEndCity <-␣
 ↪factor(final_dest_total_trips$ServiceEndCity, levels = levels)
```

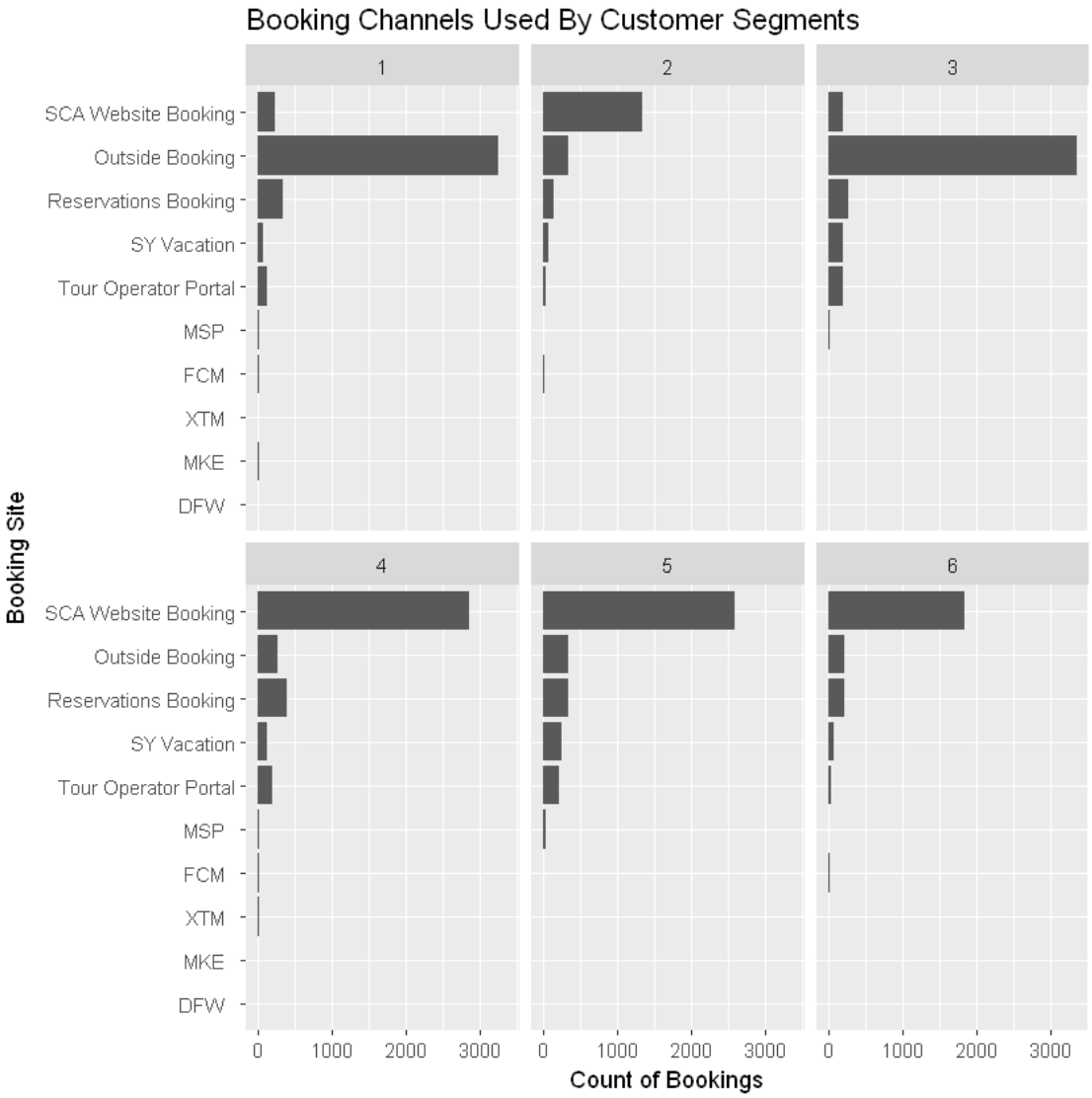
[34]:
```
## Plot out the final desitinations for our identified customer segments
ggplot(final_dest_total_trips) +
        aes(x = ServiceEndCity, fill = cluster, weight = total_trips) +
        geom_bar() +
        scale_fill_hue() +
        labs(x = "Airport Code", y = "Number of Trips", title = "Final␣
 ↪Destinations for Customer Segments", fill = "Customer Segment")
```

This figure identifies LAS (McCarran International Airport, Las Vegas, Nevada) as the busiest final destination in the given data set. Overall, travel to LAS is largely comprised of customers from clusters three and five. MCO is the second busiest destination for travelers. In contrast to LAS, travel to MCO is dominated by customers from clusters one and four. Turning our attention to clusters two and six, travel to RSW (Southwest Florida International Airport) is primarily composed of customers from cluster six. While cluster two is represented among each of our top airports, it does not appear to overwhelmingly drive travel any of the given destinations.

[12]:
```
## Visualize how these flights were booked

## Group by booking channel and cluster
customer_segments_booking_channel <-
        suncountry %>%
```

```
                group_by(BookingChannel, cluster) %>%
                summarise(booking_count = n()) %>%
                arrange(desc(booking_count))

sorted_booking_channel <- customer_segments_booking_channel %>%
        group_by(BookingChannel) %>%
        summarise(grand_total = sum(booking_count)) %>%
        arrange((grand_total))

levels = sorted_booking_channel$BookingChannel

## Re-arranging the sort order for our final destionation to help generate a␣
 ↪more understandable graphic
customer_segments_booking_channel$BookingChannel <-␣
 ↪factor(customer_segments_booking_channel$BookingChannel, levels = levels)
```

[13]:
```
## Interesting observation - we see that customer segments 1 and 3 predominately␣
 ↪used outside
## booking channels
## Customer Segments 4 & 5 used the Sun Country website

ggplot(customer_segments_booking_channel) +
        aes(x = BookingChannel, weight = booking_count) +
        geom_bar() +
        scale_fill_hue() +
        coord_flip() +
        labs(x = "Booking Site", y = "Count of Bookings",
             title = "Booking Channels Used By Customer Segments", fill =␣
 ↪"Customer Segment") +
        facet_wrap(cluster ~ .)
```

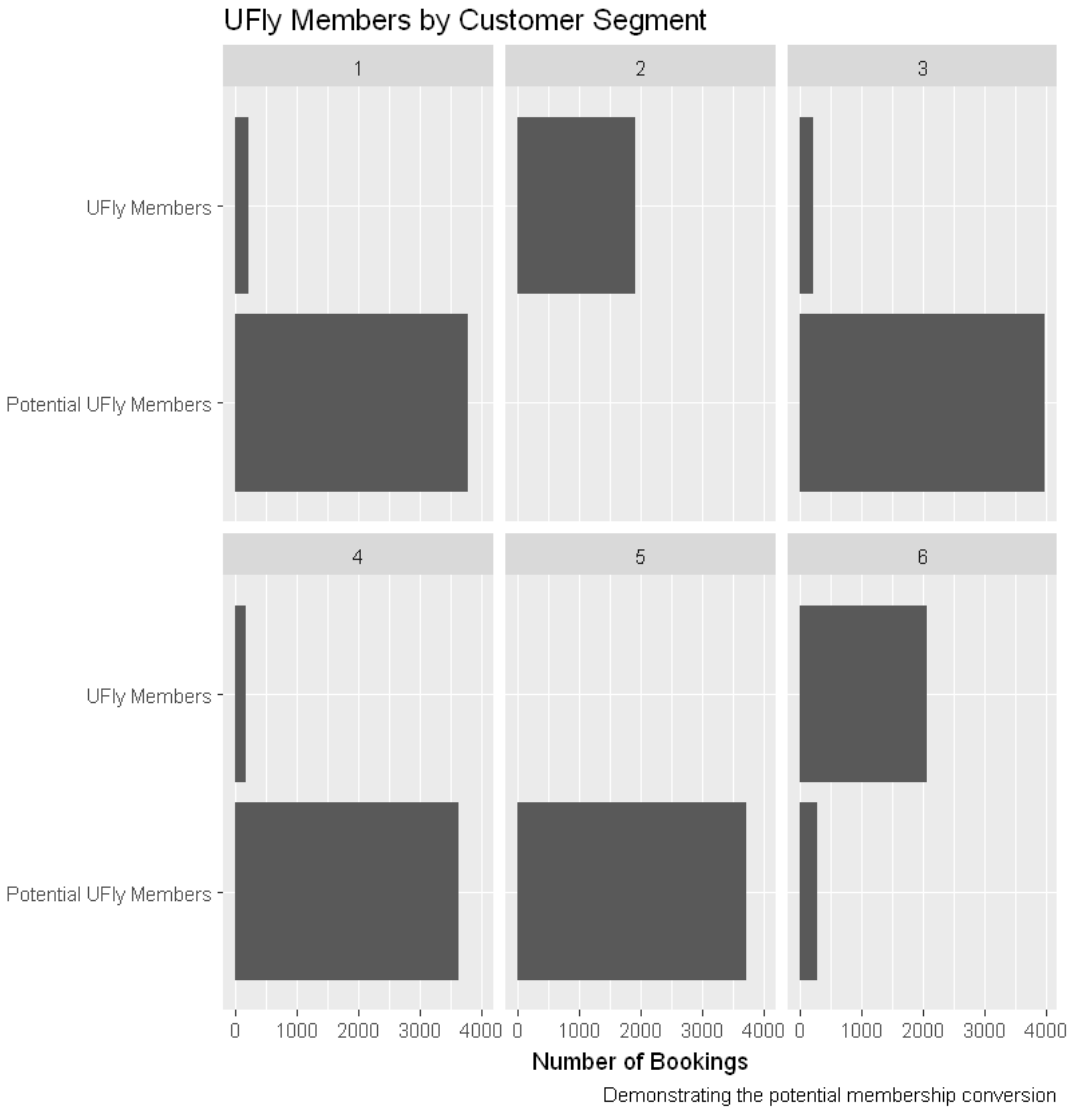


This figure suggests SCA Website Booking accounts for a substantial number of bookings within each of our clusters. While SCA booking still accounts for a sizable portion of bookings in clusters one and three, Outside Booking appears to be their preferred method.

[35]:
```
## Create a summary table for UFlyMembership - to identify which of our segments␣
 ↪are actually
## UFly members
customer_segment_ufly <-

suncountry %>%
        group_by(UflyMember, cluster) %>%
        summarise(grp_count = n()) %>%
        arrange(desc(grp_count))
```

```
## Convert UFlyMember column to factor variable
customer_segment_ufly$UflyMember <- as.factor(customer_segment_ufly$UflyMember)

customer_segment_ufly <-
        customer_segment_ufly %>%
                mutate(UflyMember_Factor = fct_recode(UflyMember,
                        "UFly Members" = "1",
                        "Potential UFly Members" = "0"))
```

[17]:
```
## Visualize UFly Membership for our customer segments - we see a lot of␣
 ↪potential for membership!
ggplot(customer_segment_ufly) +
 aes(x = UflyMember_Factor, weight = grp_count) +
 geom_bar() +
 coord_flip() +
 labs(x = "", title = "UFly Members by Customer Segment", y = "Number of␣
 ↪Bookings", caption = "Demonstrating the potential membership conversion") +
 theme_gray() +
 facet_wrap(vars(cluster))
```



To prioritize our efforts, this figure suggests we should focus our efforts on clusters one, three, four, and five. Clusters two and six are overwhelmingly comprised of current UFly members. This, of course, does not imply that we should neglect clusters two and six. Instead, we may simply want to adopt a different set of objectives for member and non-member segments.

[18]:
```
## This code block contains the steps to demonstrate how to pre-process the␣
 ↪entire dataset

## Takes about 5 minutes to load the three million row table


## data.dir = "C:/Users/monca016/Documents/Fall 2019/MSBA 6410 - Exploratory␣
 ↪Analytics/HW2/"
```

```
## Set the file name
## data.file = "SunCountry.csv"

##  Subset the data based on
## data = data %>% filter(MarketingAirlineCode == "SY") %>% select(TicketNum,␣
↪PNRLocatorID, CouponSeqNbr, ServiceStartCity,ServiceEndCity, PNRCreateDate,␣
↪ServiceStartDate, GenderCode,
  #             Age, PostalCode, BkdClassOfService, TruldClassOfService,␣
↪BookingChannel, BaseFareAmt, TotalDocAmt, UflyMemberStatus,
   #            CardHolder, BookedProduct, EnrollDate, StopoverCode)


## Remove any NULL/empty values from the data set
## data = na.omit(data)

## Update our data columns so they are Date fields - part of the cleanup
## data$PNRCreateDate = as.Date(data$PNRCreateDate)
## data$ServiceStartDate = as.Date(data$ServiceStartDate)

## Create trip month to filter on as well
## data$TripMonth = months.Date(data$ServiceStartDate)
```

[19]:
```
## Filter on the clusters we care about investigating on the whole data set
target_clusters_subset <-
        suncountry %>%
                filter(cluster == 1 | cluster == 3 | cluster == 4 | cluster == 5)


## Bring together columns to use as a look-up
target_clusters_subset <- target_clusters_subset %>%
        unite(key, ServiceStartCity, ServiceEndCity, GenderCode, TripMonth, Age)

## Confirm the function combined the fields correctly
head(target_clusters_subset)
```

id	key	BookingChannel	TimeGap	UflyMember	CardHolderFlag	clust
1	MSP_SAN_F_December_21	Outside Booking	27	0	0	1
3	MSP_SEA_F_December_63	Outside Booking	63	0	0	3
4	MSP_SEA_F_August_39	SCA Website Booking	63	0	0	4
5	MSP_ANC_F_July_25	Outside Booking	137	0	0	3
6	MSP_CUN_M_December_21	Outside Booking	24	0	0	1
7	MSP_MCO_M_December_39	Outside Booking	127	0	0	1

[33]:
```
## 1.  Create a subset of the data based on that filter criteria
```

```
## We look up customers who took the same flights, at the same time of year, at␣
 ↪the same age

## data_subset <- data %>%
   ##     filter(key %in% target_clusters_subset$key)

## 2.  Because all of the columns were merged together to form the key, we have␣
 ↪split them out again

## data_subset <-
    ##    data_subset %>%
      ##      separate(key, c("ServiceStartCity", "ServiceEndCity",␣
 ↪"GenderCode", "TripMonth", "Age"))

## 3.  Looks like we have about 759,000 observations - way too much.
## We can whittle this down a little more by merging on the flight trips
## And only get one trip per passenger

## summary(data_subset)


## 4.  We use a function created to build out flight paths -
## We'll leave this function out of the write-up
## But we can show a few of the lines in the function

## flight_legs <- function(dataframe_in) {

##  a <- dataframe_in %>%
##  select(PNRLocatorID, TicketNum, CouponSeqNbr, ServiceStartCity,␣
 ↪ServiceEndCity) %>%
##  group_by(TicketNum) %>% mutate(trip_max = max(CouponSeqNbr))

##  a1 <- a %>% filter(CouponSeqNbr == 1) %>%
##    select(PNRLocatorID, TicketNum, ServiceStartCity, ServiceEndCity,␣
 ↪trip_max) %>%
##    rename(City1 = ServiceStartCity, City2 = ServiceEndCity)
##  jx <- as_tibble(j5) %>%
##    unite("Airport_Sequence", City1:City7, sep = "->", na.rm = TRUE)

##  jx <- jx[!duplicated(jx),]

##  return(jx)
## }


## 5.  Using our pre-built function, we can generate a "flight sequence" to␣
 ↪determine the trip
```

```
## locations for each passenger/trip

## 6.  For this step, we can filter on the first "ticket" for the trip, since we␣
 ↪don't want duplicate rows

## df_seq1 <- data_subset %>% filter(CouponSeqNbr == 1)
## fl_sequence <- left_join(df_seq1, flight_legs(data_subset), by =␣
 ↪c("PNRLocatorID", "TicketNum"))


## 7.  Add in a few descriptive fields for final table - to mark if the customer␣
 ↪is a UFly
## rewards member, and if they have the credit card

## fl_sequence$UflyMember = case_when(fl_sequence$UflyMemberStatus == "" ~␣
 ↪"Non-member",
##                                    fl_sequence$UflyMemberStatus != "" ~ "UFly␣
 ↪Member")

## fl_sequence$CardHolderFlag = case_when(fl_sequence$CardHolder == "true" ~␣
 ↪"Card Holder",
##                                  fl_sequence$CardHolder != "true" ~ "Non-card␣
 ↪Holder")

## 8.  Create a list of columns that we want to subset on

## final_column_set <- c("ServiceStartCity", "ServiceEndCity", "GenderCode",␣
 ↪"BkdClassOfService",
## "TruldClassOfService","TripMonth", "Age", "BookingChannel", "BaseFareAmt",
## "Airport_Sequence", "UflyMember", "CardHolderFlag")

## 9.  Subset on the desired columns
## final_clean_output <- fl_sequence[, final_column_set]

## 10.  Remove duplicate rows from our target cluster subset of data

## remove_dups <- distinct(target_clusters_subset)

## 11.  We use a SQL query to pull the columns that we want and get the cluster
## information included as well

## final_summary_output <-

## sqldf("SELECT f.ServiceStartCity, f.ServiceEndCity, f.GenderCode,
## f.BkdClassOfService, f.TruldClassOfService, f.TripMonth, f.Age,
```

```
## f.BookingChannel, f.BaseFareAmt, f.Airport_Sequence, f.UflyMember, f.
 ↪CardHolderFlag, r.cluster
## from final_clean_output f
## INNER JOIN remove_dups r
## ON (f.ServiceStartCity = r.ServiceStartCity AND f.ServiceEndCity = r.
 ↪ServiceEndCity
## AND f.GenderCode = r.GenderCode AND f.TripMonth = r.TripMonth AND f.Age = r.
 ↪Age)")

## 12.   Remove any duplicate rows generated from our SQL query, so we can start␣
 ↪with a clean
##       unique data set

## final_summary_output <- distinct(final_summary_output)

## 13.  Last step - we save the output from this pre-processing as a CSV, so␣
 ↪that we can easily read
##       it back and use it for our data tables below

## final_summary_output_as_csv <- write.csv(final_summary_output,
##   file = "SunCountry_Summary_Output.csv")
```

[22]:
```
## Read it back in as a CSV, since this is much faster than re-running our␣
 ↪processing steps
final_summary_output <-read.csv("SunCountry_Summary_Output.csv")
```

[23]:
```
## View some summary statistics - No NAs!  Thats great!
summary(final_summary_output)
```

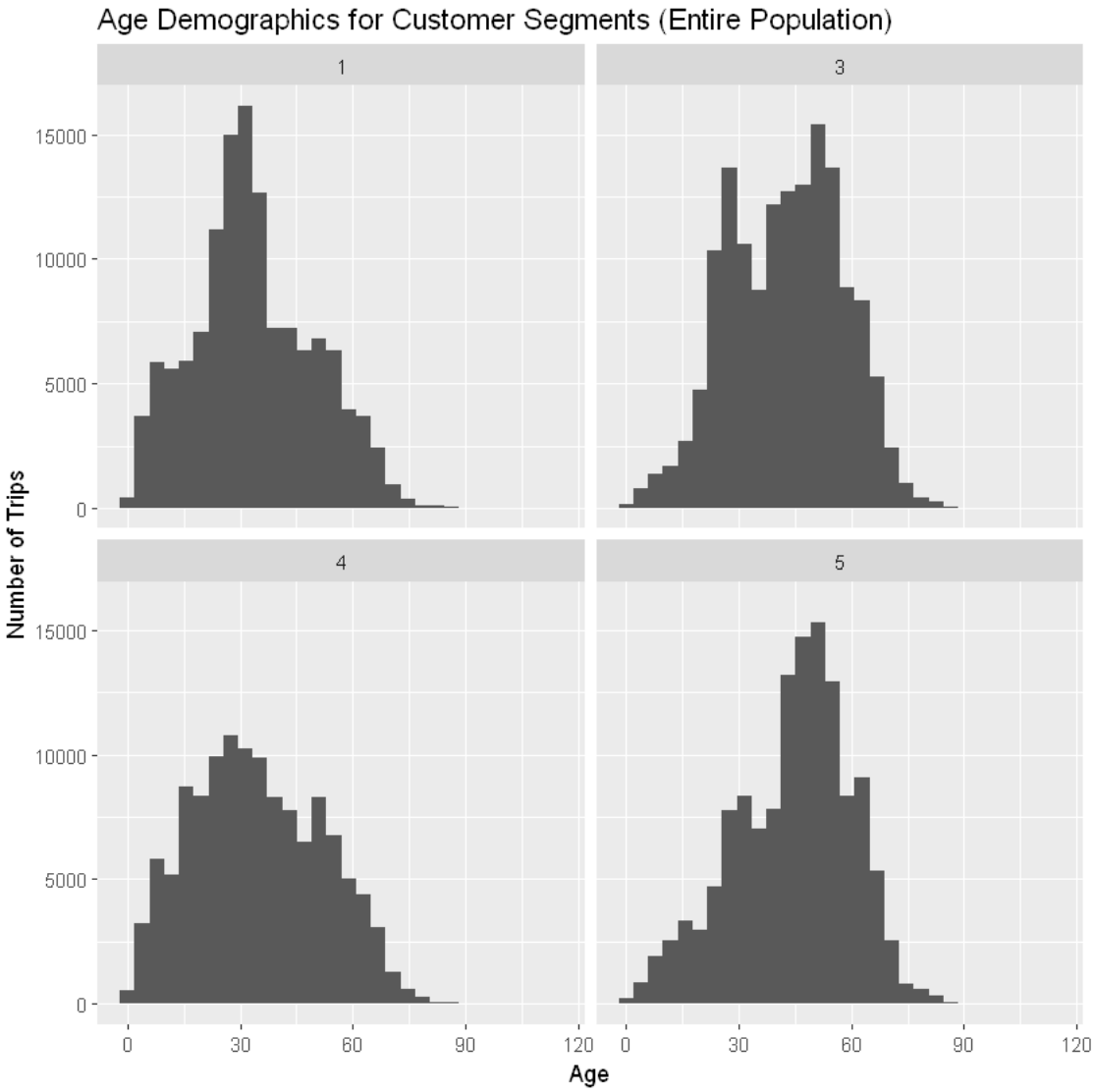
```
      id         ServiceStartCity ServiceEndCity   GenderCode
Min.   :     1   MSP:533596       LAS     :106229   F:282697
1st Qu.:133400                    MCO     : 72553   M:250898
Median :266799                    JFK     : 46403   U:     1
Mean   :266799                    SFO     : 41271
3rd Qu.:400197                    LAX     : 38705
Max.   :533596                    BOS     : 31781
                                  (Other):196654
           BkdClassOfService            TrvldClassOfService    TripMonth
Coach               :516775   Coach               :483851   March    : 79354
Discount First Class:   161   Discount First Class: 14774   February: 53633
First Class         : 16660   First Class         : 34971   July     : 48674
                                                            December: 47985
                                                            June     : 45929
                                                            August   : 44864
                                                            (Other) :213157
     Age                       BookingChannel    BaseFareAmt
Min.   :  0.00   SCA Website Booking :243636   Min.   :   0.0
```

```
1st Qu.: 27.00   Outside Booking     :230992   1st Qu.: 195.9
Median : 39.00   Reservations Booking: 27282   Median : 283.8
Mean   : 39.05   SY Vacation         : 26300   Mean   : 306.4
3rd Qu.: 52.00   Tour Operator Portal:  2891   3rd Qu.: 378.6
Max.   :114.00   MSP                 :  1685   Max.   :1840.0
                 (Other)             :   810
Airport_Sequence        UflyMember             CardHolderFlag
MSP->LAS:106229   Non-member :404641   Card Holder    :  7457
MSP->MCO: 72552   UFly Member:128955   Non-card Holder:526139
MSP->JFK: 46392
MSP->SFO: 41265
MSP->LAX: 38697
MSP->BOS: 31777
(Other) :196684
   cluster
Min.   :1.000
1st Qu.:3.000
Median :3.000
Mean   :3.242
3rd Qu.:4.000
Max.   :5.000
```

[24]:
```
## Look at the first few rows to see a nice, clean dataset
head(final_summary_output)
```

id	ServiceStartCity	ServiceEndCity	GenderCode	BkdClassOfService	TrvldClassOfService	TripMonth
1	MSP	JFK	M	Coach	Coach	August
2	MSP	JFK	M	Coach	Coach	August
3	MSP	SFO	F	Coach	Coach	December
4	MSP	LAS	F	Coach	Coach	Septembe
5	MSP	LAS	F	Coach	Discount First Class	Septembe
6	MSP	JFK	F	Coach	Coach	July

[117]:
```
ggplot(data = final_summary_output, aes(final_summary_output$Age)) +
    geom_histogram(bins = 30) +
    facet_wrap(. ~ cluster) +
    labs(x = "Age", y = "Number of Trips", title = "Age Demographics for␣
 ↪Customer Segments (Entire Population)")
```

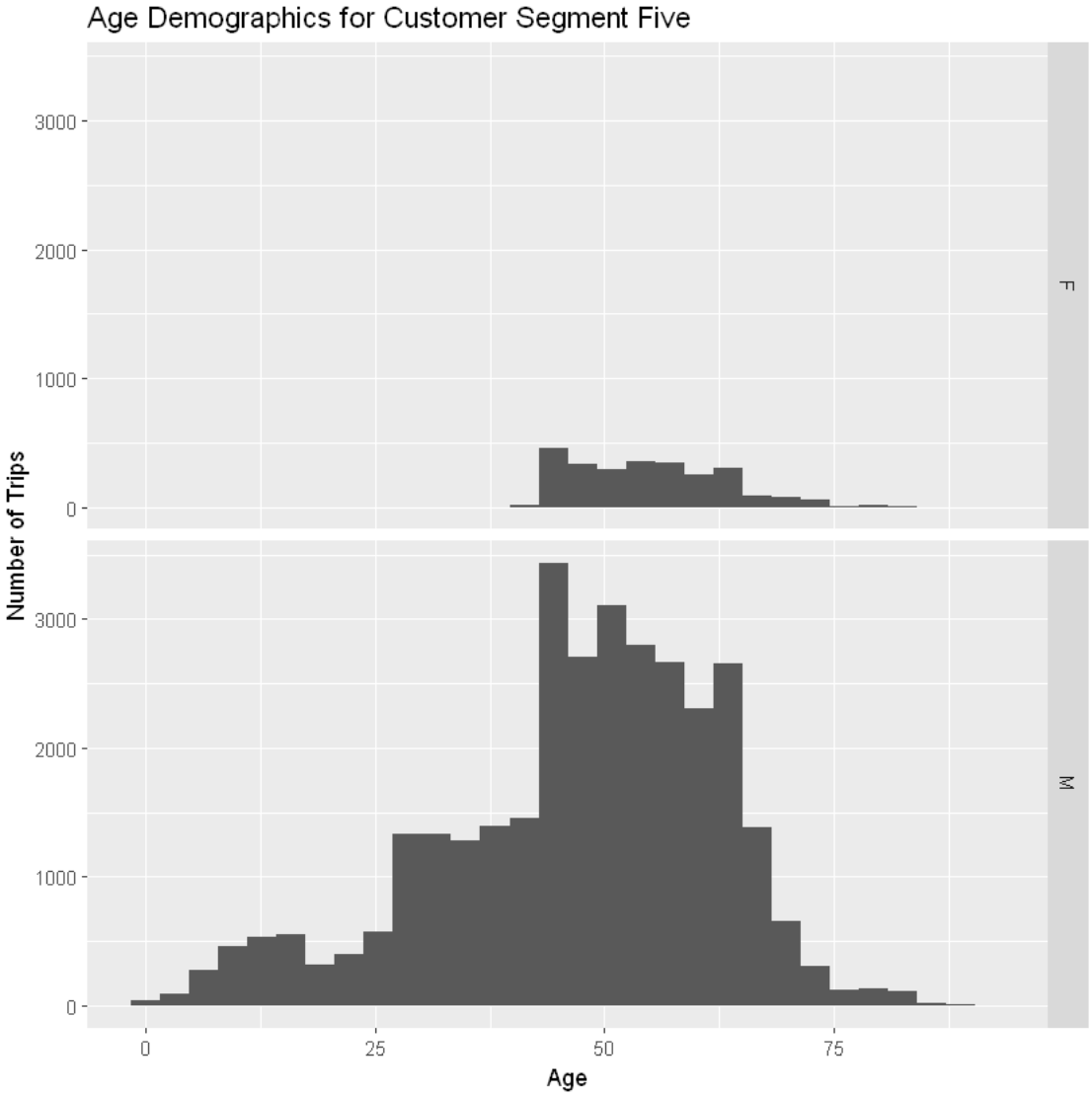


This figure depicts the age distribution of the clusters that offer the highest conversion potential.

4.2 Characteristics of UFly Reward members

[49]:
```
## Create a subset of our clustered data, combined with the full data set
## Filter on UFlyMembers and customer segment five
customer_segment_five <-
final_summary_output %>%
    filter(UflyMember == "UFly Member") %>%
    filter(cluster == 5)
```

[77]:
```
## Generate a plot to view Age and Gender demographics for customer segment five

ggplot(data = customer_segment_five, aes(customer_segment_five$Age)) +
    geom_histogram(bins = 30) +
    labs(x = "Age", y = "Number of Trips", title = "Age Demographics for␣
 ↪Customer Segment Five") +
    facet_grid(GenderCode ~ .)
```



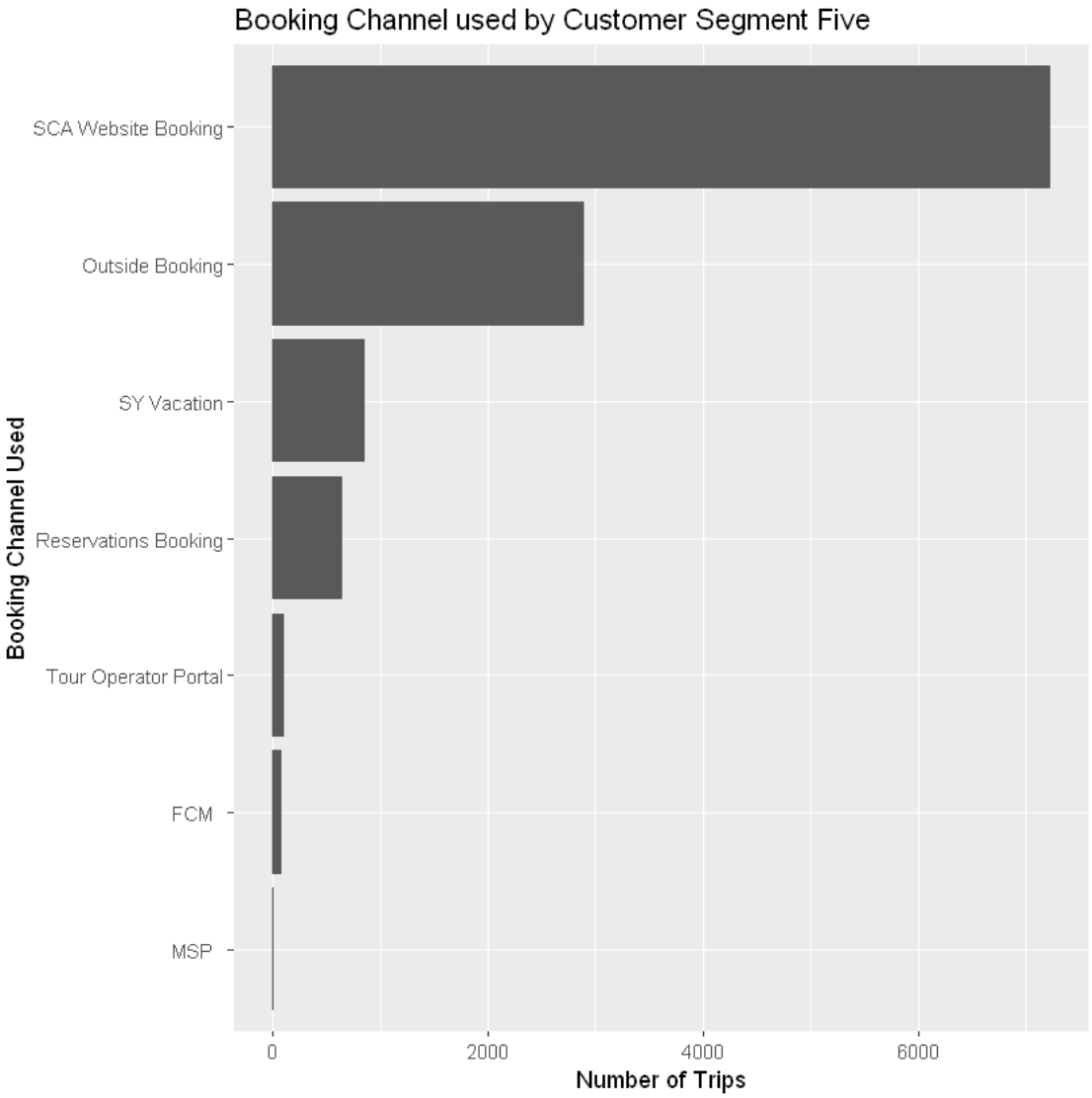
Within this plot, we take a closer look at the age and sex composition of cluster five. According to this plot, cluster five is overwhelming (but not exclusively) male.

[87]:
```
## Segment on customer booking information and where they were flying to
customer_segment_five_booking_info <-
```

```
customer_segment_five %>%
        group_by(BookingChannel, BkdClassOfService, GenderCode, Age) %>%
        summarise(trip_count = n()) %>%
        top_n(10) %>%
        arrange(desc(trip_count))

## Create a quick analysis to sort our booking channel information for easier␣
 ↪viewing
sorted_table <- customer_segment_five_booking_info %>%
        group_by(BookingChannel) %>%
        summarise(grand_total = sum(trip_count)) %>%
        arrange(grand_total)

levels = sorted_table$BookingChannel

## Re-arranging the sort order for our final destionation to help generate a␣
 ↪more understandable graphic
customer_segment_five_booking_info$BookingChannel <-␣
 ↪factor(customer_segment_five_booking_info$BookingChannel, levels = levels)
```

```
Selecting by trip_count
```

[110]:
```
## Generate a plot to visualize what booking channel was used by this segment
ggplot(customer_segment_five_booking_info) +
        aes(x = BookingChannel, fill = Age, weight = trip_count) +
        geom_bar() +
        scale_fill_hue() +
        coord_flip() +
        labs(x = "Booking Channel Used", y = "Number of Trips", title = "Booking␣
 ↪Channel used by Customer Segment Five", fill = "Customer Segment")
```



This figure suggests cluster five already makes extensive use of SCA Website Booking. At the same time, a substantial number of customers in cluster five also use Outside Booking.

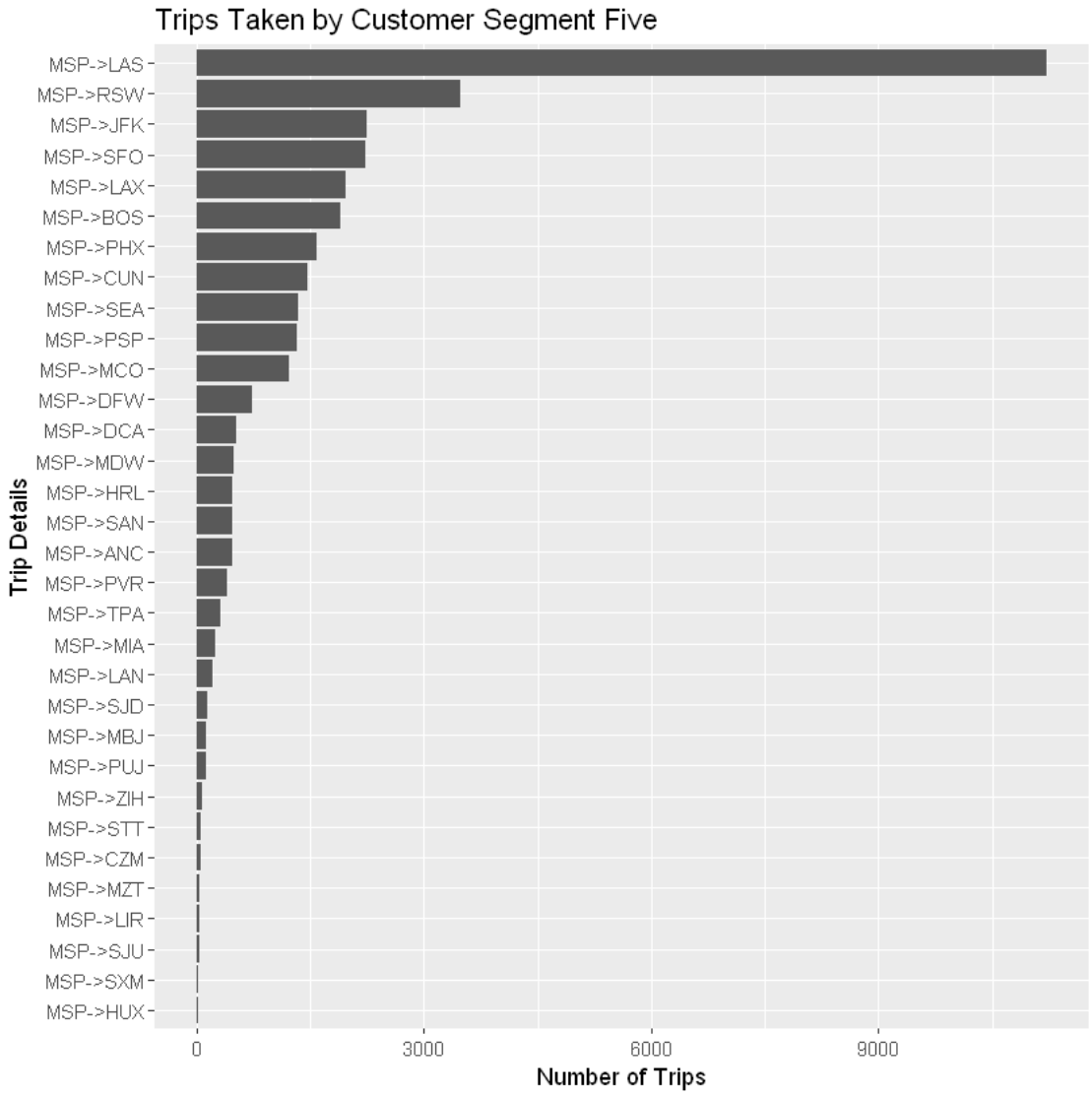
[114]:
```
## Where are our identified UFlyMembers flying?
customer_segment_five_trip_info <-
customer_segment_five %>%
        group_by(TripMonth, Airport_Sequence, GenderCode) %>%
        summarise(trip_count = n()) %>%
        top_n(10) %>%
        arrange(desc(trip_count))

head(customer_segment_five_trip_info)
```

```
Selecting by trip_count
```

TripMonth	Airport_Sequence	GenderCode	trip_count
March	MSP->RSW	M	1557
October	MSP->LAS	M	1253
March	MSP->LAS	M	1065
May	MSP->LAS	M	950
November	MSP->LAS	M	900
January	MSP->LAS	M	861

[115]:
```
## Create a function to order the trip locations correctly needed for our plot
airport_sorted_table <- customer_segment_five_trip_info %>%
        group_by(Airport_Sequence) %>%
        summarise(grand_total = sum(trip_count)) %>%
        arrange(grand_total)

airport_levels = airport_sorted_table$Airport_Sequence

## Re-arranging the sort order for our final destionation to help generate a␣
 ↪more understandable graphic
customer_segment_five_trip_info$Airport_Sequence <-␣
 ↪factor(customer_segment_five_trip_info$Airport_Sequence, levels =␣
 ↪airport_levels)
```

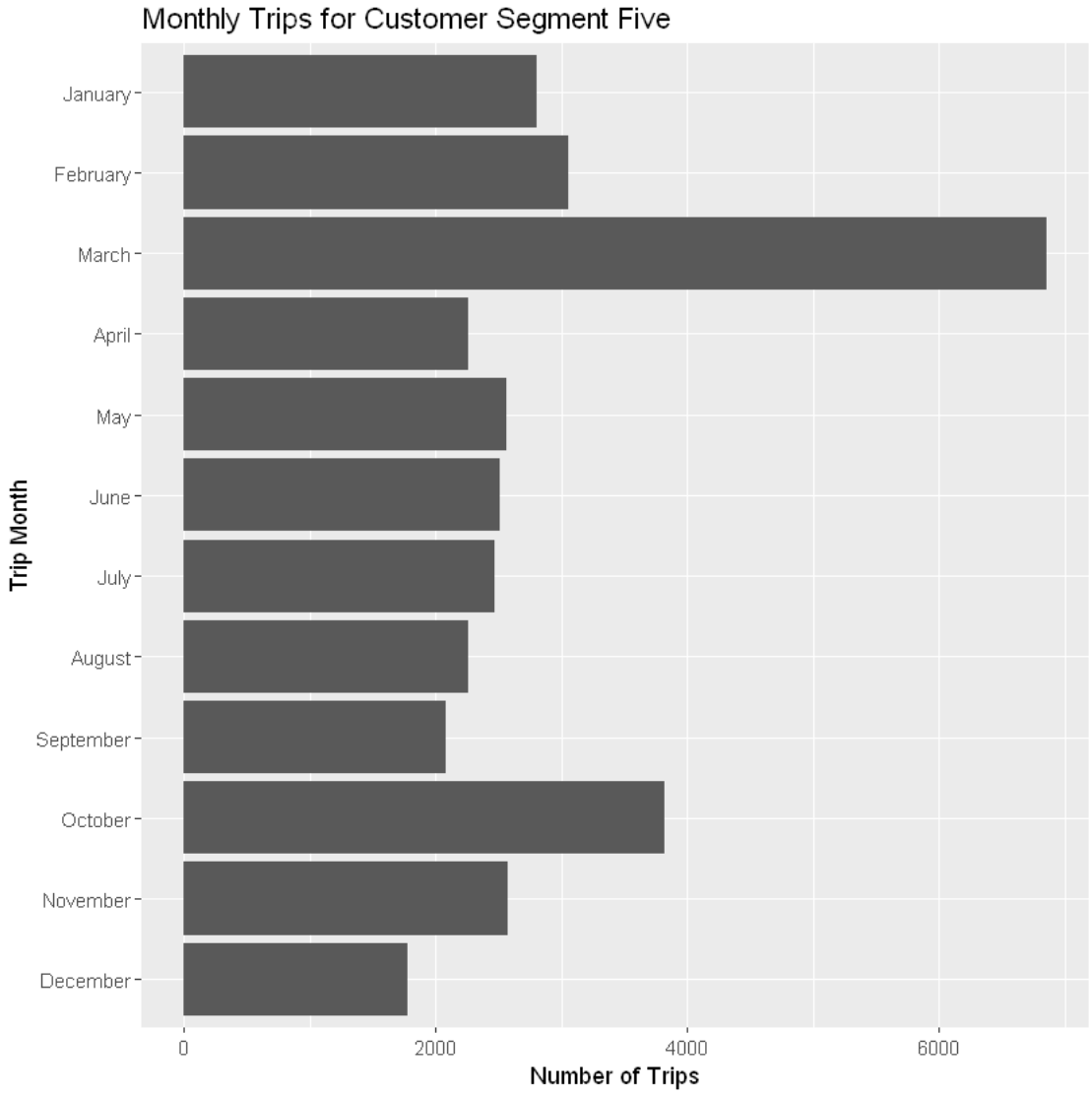
[116]:
```
## Generate a plot to demonstrate where customers in this segment are flying
ggplot(customer_segment_five_trip_info) +
        aes(x = Airport_Sequence, weight = trip_count) +
        geom_bar() +
        scale_fill_hue() +
        coord_flip() +
        labs(x = "Trip Details", y = "Number of Trips", title = "Trips Taken by␣
 ↪Customer Segment Five", fill = "Customer Segment")
```



[106]:
```
## Create a function to order the months correctly needed for our plot
month_levels = c("January", "February", "March", "April", "May", "June", "July",␣
 ↪"August", "September",
          "October", "November", "December")

## Re-arranging the sort order for our trip months to help generate a more␣
 ↪understandable graphic
customer_segment_five_trip_info$TripMonth <-␣
 ↪factor(customer_segment_five_trip_info$TripMonth, levels = rev(month_levels))
```

[108]:
```
## Generate a plot to get a sense of when our travelers are flying
ggplot(customer_segment_five_trip_info) +
        aes(x = TripMonth, weight = trip_count) +
```

```
        geom_bar() +
        scale_fill_hue() +
        coord_flip() +
        labs(x = "Trip Month", y = "Number of Trips", title = "Monthly Trips for␣
↪Customer Segment Five", fill = "Customer Segment")
```



This figure suggests March is the most popular trip month for cluster five by a rather large margin.

5 Conclusions

One particular cluster of interest was Men in their 50's who fly from Minneapolis to Las Vegas in coach, booking via the Sun Country Website, that are not Ufly Rewards members or credit

card holders. Minneapolis to Las Vegas is a very popular flight for Sun Country and is offered frequently and this group of fliers in particular stands out. This customer segment is already booking through the website so an extra push to get them to be Ufly Rewards members may help to get them to make more trips on Sun Country.

6 Recommendations

We can get the most potential benefit by getting older men who are already booking via Sun County's website to sign up for Ufly memberships. This group mostly flies to Vegas but another frequent route is Fort Meyers in March; which would put them there for the Twin's spring training.

We recommend lowering the barriers to sign up by making it incredibly easy to opt-in during check out and incentivizing with a reward like two drink vouchers. This group is already on the website so by offering any incentive and making this easy during checkout it should have a good conversion rate.

To make being a Ufly member a bigger deal in general, as an ongoing benefit, it could be worth offering priority boarding for all members. Membership in the program is low enough overall that this should be a perk for a while. If membership becomes more popular, this may lose value.

Both of these promotions should hold appeal outside of the target segment as well, but those not booking via the website may not be aware of them.

An additional marketing promotion to consider for this group would be to expand on the Spring Training appeal and market going for short trips to see baseball games. This would probably work best for the for destinations with AL teams (in or nearby) Boston, Seattle, NY, Chicago, San Francisco, LA, Dallas, Tampa but may also work for NL teams in Miami, Denver, San Diego, Phoenix, DC, Philly, St. Louis. As this group is booking via the website, they should be easy to target digitally for re-marketing.

Encouraging people to book via the website is more complicated, however, Sun Country has the opportunity to convince these travelers while they a captive audience in flight. Announcing the priority boarding for Ufly members might be one nudge but it would be easy to include in other materials or announcements that there are benefits to booking via SCA and/or joining Ufly Rewards.

[ ]:

[ ]: