HW 3 - Customer Segmentation for Sun Country Airlines

Jesse Sprinkel
Haojin Jia
Chandrakanth Tolupunoori
Anshuman Vijayvargia
Priyanka Singhal

10/20/2018

Abstract

This is the white paper and technical details of our analysis and recommendations.

Contents

1 Main Problem and Our Approach:

1.1 Situation:

Our objective is to analyze the historical flight bookings data and profiles the customers into segments which would eventually help SunCountry to better market and advertise to their customers.

1.2 Complication:

While the data set is quite rich and extensive, it is computationally expensive. Adding to this problem, we would like to use a technique called clustering where we attempt to find natural segments in the data. Using clustering involves a couple assumptions: we need to assume that you do have customer groups that behave differently, and we need to pick the right features to cluster on that accurately differentiate the groups. Being scientific in our approach is difficult but will allow us to base our customer segments on data as much as possible.

1.3 Our Approach

Because the dataset is so large we ran our analysis on multiple samples of 5,000 unique trips out of the total amount of 1.17 million trips. We used a particular method of clustering called k-prototypes and separated the groups based on the following: starting city, destination, booking channel, group size, type of trip (round trip vs. one-way), time of year, membership status, and how far in advance the ticket was booked. While there may be differences between our samples and the final dataset, we think they would be negligible and not impact our final conclusions.

2 Preparedness

2.0.1 Brief Summary of the Dataset

What does the data look like?:

- 3.4M+ records containing 2 years of historical flight bookings
- ~ 1.1M unique booking PNRs
- Demographic details like EncryptedName, Gender, Age, PostalCode
- Details about Ufly Member status, TravelClass, BookingChannel

3 Data Analysis

3.1 R Libraries

Please make sure successfully install the packages before reproducing the research

```
library(dplyr)
library(tidyr)
library(ggplot2)
library(stringr)
library(lubridate)
library(rsconnect)
library(R.utils)
library(clustMixType)
library(cluster)
library(stringr)
library(cowplot)
library(magrittr)
```

Set working directory

```
knitr::opts_knit$set(root.dir = 'D:\\Fall Term\\MSBA 6410 - Exploratory Data Analytics and Visualizatio
```

3.2 Data Exploration

Load the dataset and look at the various columns and datatypes

```
# Make sure data with huge numbers is read properly
options("digits")
```

```
## $digits
## [1] 7
```

```
# We are making the options to 14 so that it can read bigger integers. It was 7 previously.
options(digits = 14)
# Read the data
suncountry <- read.csv("SunCountry.csv", stringsAsFactors = FALSE)
# Look at the data types and head values in every column
glimpse(suncountry)
```

```
## Observations: 3,435,388
## Variables: 26
## $ PNRLocatorID         <chr> "AAABJK", "AAABJK", "AAABMK", "AAABMK", "...
## $ TicketNum            <dbl> 3377365159634, 3377365159634, 33721073819...
## $ CouponSeqNbr         <int> 2, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1,...
## $ ServiceStartCity     <chr> "JFK", "MSP", "MSP", "SFO", "MCO", "PSP",...
## $ ServiceEndCity       <chr> "MSP", "JFK", "SFO", "MSP", "MSP", "MSP",...
## $ PNRCreateDate        <chr> "2013-11-23", "2013-11-23", "2014-02-04",...
## $ ServiceStartDate     <chr> "2013-12-13", "2013-12-08", "2014-02-23",...
## $ PaxName              <chr> "BRUMSA", "BRUMSA", "EILDRY", "EILDRY", "...
## $ EncryptedName        <chr> "4252554D4241434B44696420493F7C2067657420...
## $ GenderCode           <chr> "F", "F", "M", "M", "F", "M", "F", "M", "...
## $ birthdateid          <int> 35331, 35331, 46161, 46161, 34377, 39505,...
## $ Age                  <int> 66, 66, 37, 37, 69, 54, 25, 69, 49, 58, 2...
## $ PostalCode           <chr> "", "", "", "", "", "", "", "", "", "5511...
## $ BkdClassOfService    <chr> "Coach", "Coach", "Coach", "Coach", "Coac...
## $ TrvldClassOfService  <chr> "Coach", "First Class", "Discount First C...
## $ BookingChannel       <chr> "Outside Booking", "Outside Booking", "SC...
## $ BaseFareAmt          <dbl> 234.20, 234.20, 293.96, 293.96, 112.56, 1...
## $ TotalDocAmt          <dbl> 0.0, 0.0, 338.0, 338.0, 132.0, 194.8, 191...
## $ UFlyRewardsNumber    <int> NA, NA, NA, NA, NA, NA, NA, NA, NA, 20236...
## $ UflyMemberStatus     <chr> "", "", "", "", "", "", "", "", "", "Stan...
## $ CardHolder           <chr> "", "", "", "", "", "", "", "", "", "fals...
## $ BookedProduct        <chr> "CHEOPQ", "CHEOPQ", "", "", "", "", "SSWM...
## $ EnrollDate           <chr> "", "", "", "", "", "", "", "", "", "2010...
## $ MarketingFlightNbr   <chr> "244", "243", "397", "392", "342", "610",...
## $ MarketingAirlineCode <chr> "SY", "SY", "SY", "SY", "SY", "SY", "SY",...
## $ StopoverCode         <chr> "O", "", "O", "", "", "", "", "", "", "",...
```

Filtering the data for SunCountry

```
table(suncountry$MarketingAirlineCode)
```

```
##
##      DE      F9      FI      HA      SY
##       1    3319      13    1705 3430350
```

```
#We are filtering the rows which have airline as Sun Country
Sun <- suncountry %>% filter(MarketingAirlineCode == "SY")
```

3.3 Data Cleaning

Replacing missing values and re-grouping

```r
# Member status column
table(Sun$UflyMemberStatus)
```

```
##
##             Elite Standard
##  2735876    14361   680113
```

```r
#We observe that all the non-members have blank values in Status column

#We are considering flyers to be non-member if they have member status as blank.
Sun <- Sun %>%
  mutate(UflyMemberStatus =
         ifelse(UflyMemberStatus == "", "Non Ufly Member", UflyMemberStatus))

table(Sun$BookingChannel,useNA = "always")
```

```
##
##                 ANC                 BOS                 DCA
##                   9                   1                  24
##                 DFW                 FCM                 GJT
##                 521                3817                   1
##                 HRL                 JFK                 LAN
##                  18                 256                 272
##                 LAS                 LAX                 MCO
##                 180                 414                  17
##                 MDW                 MIA                 MKE
##                 204                   1                 262
##                 MSN                 MSP       Outside Booking
##                   1                4969             1471179
##                 PHX                 PSP  Reservations Booking
##                  21                  28              262327
##                 RSW   SCA Website Booking                 SEA
##                  91             1457127                  42
##                 SFO          SY Vacation Tour Operator Portal
##                 141               94601              133201
##                 UFO                 XTM                <NA>
##                 149                 476                   0
```

```r
#We are classifying all the booking channel which were via airports as 'Other'
Sun$BookingChannel[Sun$BookingChannel!="Outside Booking" &
                     Sun$BookingChannel!="SCA Website Booking" &
                     Sun$BookingChannel!="Tour Operator Portal" &
                     Sun$BookingChannel!="Reservations Booking" &
                     Sun$BookingChannel!="SY Vacation"] <- "Other"
# Converting the column to factor for later analysis
Sun$BookingChannel<-as.factor(Sun$BookingChannel)
```

Exploring the data for missing and erroneous values

```r
# GenderCode column
table(Sun$GenderCode)
```

```
##
##               F       M       U
##   43975 1767909 1618426      40
```

```r
# We see that there are blank values in GenderCode, So we removed those rows
Sun <- Sun %>% filter(GenderCode %in% c("F", "M", "U"))
# birthdateid column
summary(Sun$birthdateid)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -675290   39618   45085   44981   50250 1112840
```

```r
# There are negative values in the id, but we are not going to remove it
# as they are just unique id's

# Age Column
summary(Sun$Age)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -2883.00    26.00    40.00    40.05    55.00  2012.00
```

```r
#We are removing rows with error values and select age between [0,115]
Sun <- Sun[!(Sun$Age < 0 | Sun$Age > 115),]
# Distribution after cleaning the Age column
x <- Sun$Age
#Plotting a normal distribution curve for age
h <-hist(x, breaks=20, xlab="Age", main="Age Distribution")
xfit<-seq(0,110,length = 50)
yfit<-dnorm(xfit,mean=mean(x, na.rm = T),sd=sd(x, na.rm = T))
yfit <- yfit*diff(h$mids[1:2])*length(x)
lines(xfit, yfit, col="blue", lwd=2)
```

Missing and erroneous values continued

```
# Coupon Sequence Number column
table(Sun$CouponSeqNbr)
```

```
##
##       1       2       3       4       5       6       7       8
## 1927236 1386570   42608   29206     592     120       6       1
```

```
# We can see that as the coupon seq nbr increases, count decreases.
# We wanted to see if all PNR's is not starting with a couponSeqNo of 1.
Sun_temp<- Sun %>% select(PNRLocatorID, CouponSeqNbr) %>%
  group_by(PNRLocatorID) %>% summarise(min_seq_nbr = min(CouponSeqNbr)) %>%
  filter(min_seq_nbr > 1)
#Looking at the top 6 rows of Sun_temp
head(Sun_temp)
```

```
## # A tibble: 6 x 2
##   PNRLocatorID min_seq_nbr
##   <chr>              <dbl>
## 1 AAAWGV                 3
## 2 AADOFC                 2
## 3 AAJIBZ                 2
## 4 AAJUAM                 2
## 5 AAKOYN                 2
## 6 AALHSE                 2
```

```
# As we can see that the min_seq_nbr is greater than 1,
# we will remove them as the journey has to start with CouponSeqNbr 1
Sun <- Sun %>% group_by(PNRLocatorID) %>%
  mutate(flag = ifelse(min(CouponSeqNbr) != 1, 1, 0)) %>% filter(flag == 0)
```

Reference: https://www.flyuia.com/hu/en/information/uia-flight-coupons

3.4 Data Transformation

Now that we have cleaned all the variables of interest, we went ahead to create a sample to transform our columns. We also observed that we have multiple rows in single PNR for each leg of the trip for each customer. We want to aggregate the data to bring it to a level where we have one row for each customer without losing much of the other information by generating new columns and using aggregation.

Sampling the data for data transformation

The size of the dataset is very large. So we wanted to take a sample and run all the data transformation steps on a smaller sample consisting of data related to 5000 unique PNR's.

```
# Obtain Unique PNRs
uniquePNRs <- unique(Sun$PNRLocatorID)

# To produce the same samples every time set the seed
set.seed(2000)
sample_PNRs <- sample(uniquePNRs,5000)

# Obtaining data related to the sampled 5000 PNRs
Sun_sample <- Sun %>% filter(PNRLocatorID %in% sample_PNRs)
```

Transformation to populate the First_City for booking

In order to aggregate multiple rows for each PNR, we need a unique id. We Assume that a combination of

four columns “PaxName”, “EncryptedName”, “GenderCode”, “birthdateid” will be able to generate a unique id needed for aggregation. Based on the unique id we group the data and pick the ServiceStartCity in the first row of the group as the First_City of the journey.

```
#As we can see there are multiple rows for many columns,
#we wanted to find a primary key by which we can group and transform the data.

Sun_sample <- Sun_sample %>%
  mutate(uid=paste(PaxName,EncryptedName,GenderCode,birthdateid,sep=""))

First_City <- Sun_sample %>%
  arrange(PNRLocatorID, CouponSeqNbr) %>%
  group_by(PNRLocatorID, PaxName) %>%
  do(data.frame(First_City = first(.$ServiceStartCity)))

#Here, we join both the tables by PnrLocatorId and Passenger Name
Sun_sample <- merge(Sun_sample,First_City,
                    by.x=c("PNRLocatorID","PaxName"),
                    by.y = c("PNRLocatorID","PaxName"))
```

Transformation to populate the Last_City for booking

Based on the unique id we group the data and pick the ServiceEndCity in the last row of the group as the Last_City of the journey.

```
#Add the Last_City of Journey
Last_City <-Sun_sample %>%
  arrange(PNRLocatorID,CouponSeqNbr)%>%
  group_by(PNRLocatorID,PaxName)%>%
  do(data.frame(Last_City=last(.$ServiceEndCity)))

#Here, we join both the tables by PnrLocatorId and Passenger Name
Sun_sample <-merge(Sun_sample,Last_City,
                    by.x=c("PNRLocatorID","PaxName"),
                    by.y = c("PNRLocatorID","PaxName"))
```

Transformation to find if customers stay at a destination in a round trip

If we only look at the First_City and Last_city we might be missing out on information about the intermediate stay where the customer actually intended to travel. So we are looking at the time difference between all the legs of the journey and Assume that the one with the maximum difference and pick the ServiceEndCity of that leg as the Final_Destination.

```
#Convert Service Start date to Date type
Sun_sample$ServiceStartDate <- as.Date(Sun_sample$ServiceStartDate)

#The place of maximum stay during the trip.
Max_Stay <- Sun_sample%>%
  arrange(PNRLocatorID, CouponSeqNbr) %>%
  group_by(PNRLocatorID, PaxName) %>%
  mutate(stay = lead(ServiceStartDate)-ServiceStartDate, default=0) %>%
  select(PNRLocatorID, PaxName, ServiceStartCity, ServiceEndCity, ServiceStartDate, stay)

#Filling zero for direct flights
Max_Stay$stay[is.na(Max_Stay$stay)] <- 0
Max_Stay$stay <- as.numeric(Max_Stay$stay)
```

```
#Merge the new column with the data
Final_Destination <- Max_Stay %>%
  group_by(PNRLocatorID, PaxName) %>%
  do(data.frame(Final_Destination =
                  first(as.character(.$ServiceEndCity)[.$stay==max(.$stay)])))

Sun_sample <- merge(Sun_sample, Final_Destination,
                    by.x=c("PNRLocatorID","PaxName"),
                    by.y = c("PNRLocatorID","PaxName"))
```

Transformation for round trip and group size

We wanted to differentiate customer based on if they booked for a round trip or on way and also based on the fact that if they booked as a group or as individuals. So we generated a flag "round_trip" to capture this information based on First_City and Last_City. Then we use the unique id generated to count the group size based on the number of unique values within each PNR and then create a flag "group" if the group size is greater than 1.

```
Sun_sample <- Sun_sample%>%
  mutate(round_trip = ifelse(as.character(First_City)==as.character(Last_City), 1, 0))

#We look at the group size, number of people who traveled together in each trip.
Sun_sample <- Sun_sample %>%
  group_by(PNRLocatorID) %>%
  mutate(group_size= length(unique(uid)))

Sun_sample <- Sun_sample %>%
  group_by(PNRLocatorID)%>%
  mutate(group = ifelse(group_size > 1, 1, 0))
```

Transformation to calculate the number of days in advance the booking was made and also look for seasonality

We also wanted to differentiate and target our customers differently based on the number of days in advance they book their tickets and also the time of the year they travel. So we used the "ServiceStartDate" and "PNRCreateDate" to generate these two values.

```
library(lubridate)
Sun_sample$ServiceStartDate<-as.Date(Sun_sample$ServiceStartDate)
#Convert ServiceStartDate from factor to Date format
Sun_sample<- Sun_sample %>%
  group_by(PNRLocatorID, PaxName) %>% mutate(month_no = month(ServiceStartDate))

#We look at the number of days the ticket was booked in advance.
Sun_sample$PNRCreateDate <- as.Date(Sun_sample$PNRCreateDate)
Sun_sample$ServiceStartDate <- as.Date(Sun_sample$ServiceStartDate)

Sun_sample <- Sun_sample%>%
  mutate(days_pre_booked = as.numeric(floor(difftime(ServiceStartDate,
                                                     PNRCreateDate,units=c("days")))))
```

Selecting Columns of Interest for clustering

Now that we have all the columns we think might impact our clustering results, we select those columns and generate the final aggregated table.

```r
#We transformed the data such that each row represents a unique customer-PNR combination.
Sun_sample <- Sun_sample %>%
  select(PNRLocatorID, uid, PaxName, ServiceStartDate, BookingChannel, Age,
         UFlyRewardsNumber,UflyMemberStatus,First_City, Last_City,Final_Destination,
         round_trip,group_size,group, month_no ,days_pre_booked)

data_transformed <- Sun_sample %>%
  group_by(PNRLocatorID, uid, PaxName) %>%
  summarise(ServiceStartDate = first(ServiceStartDate),
            BookingChannel = first(BookingChannel),
            UFlyRewards = first(UFlyRewardsNumber),
            UflyMemberStatus = first(UflyMemberStatus),
            Age = max(Age),
            First_City = first(First_City),
            Last_City = last(Last_City),
            Final_Destination = first(Final_Destination),
            round_trip = first(round_trip),
            group_size = first(group_size),
            group = first(group),
            month_no = last(month_no),
            days_pre_booked = max(days_pre_booked))
#Retaining only those attributes that are meaningful for clustering
data_transformed <- data_transformed %>%
  select(-PNRLocatorID, -uid, -PaxName, -ServiceStartDate, -UFlyRewards)
```

4 Clustering

4.1 Normalization of numeric columns

Normalization the numeric columns of interest

As we are going to use clustering, which uses distance metric to measure similarity, we wanted to bring all the numeric variables to the same scale by using Min-Max Normalization.

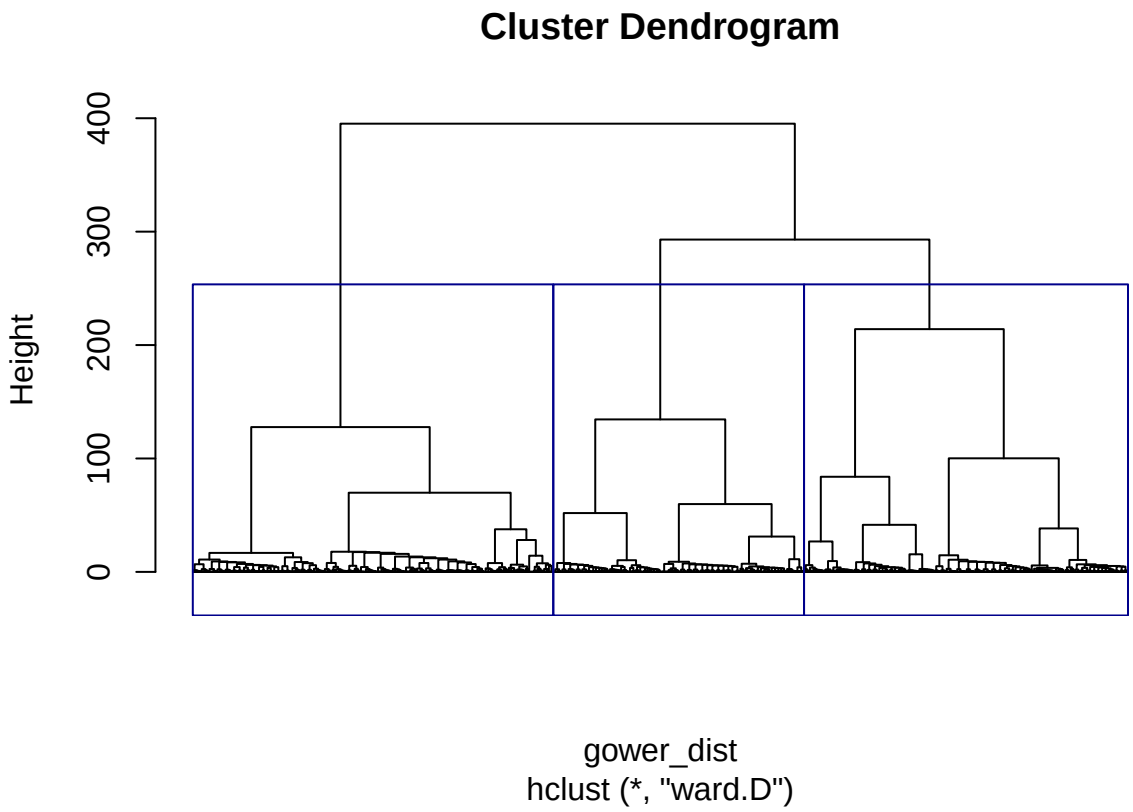
```r
normalize <- function(x){return ((x - min(x))/(max(x) - min(x)))}

temp <- ungroup(data_transformed)

customer_data_clust = mutate(temp,
                      Age = normalize(Age),
                      days_pre_booked = normalize(days_pre_booked),
                      group_size=normalize(group_size))
```

4.2 Hierarchical clustering and Dendogram

Explanation of Approach and Goals: Using Hierarchical clustering to check how many clusters would be optimal for clustering

```r
# Calculate distance - daizy
customer_daizy <- customer_data_clust[,c(3:5,8:13)]
customer_daizy$BookingChannel<-as.factor(customer_daizy$BookingChannel)
customer_daizy$UflyMemberStatus<-as.factor(customer_daizy$UflyMemberStatus)
customer_daizy$month_no<-as.factor(customer_daizy$month_no)
customer_daizy$Final_Destination <- as.factor(customer_daizy$Final_Destination)
```

```
gower_dist <- daisy(customer_daizy,
                    metric = "gower",
                    type = list(logratio = 3))
h_cluster <- hclust(gower_dist, method = "ward.D")
plot(h_cluster, hang = 0, label = F, main = "Cluster Dendrogram")
groups<-cutree(h_cluster,k=3)
rect.hclust(h_cluster, k = 3, border = "darkblue")
```



Interpretation from Approach: We can see from the dendrogram that the data can be separated with 3 optimal clusters.

Conclusions from Approach: We decided to go ahead with 3 clusters.

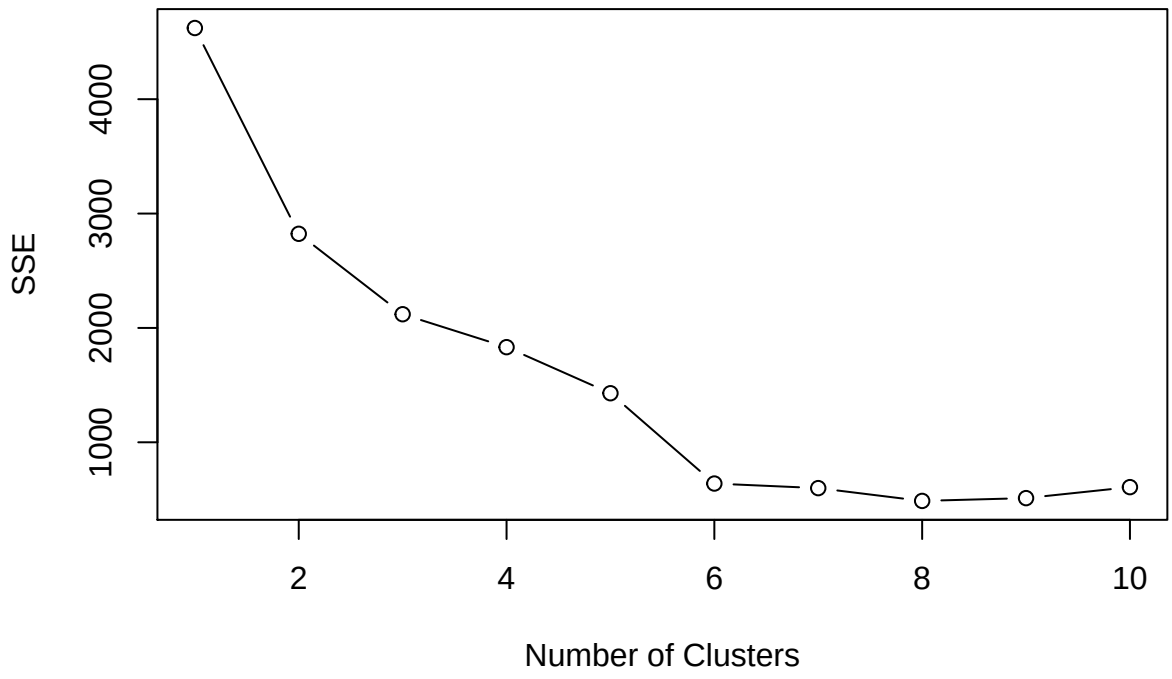
4.3 K-Prototyes Clustering

Explanation of Approach and Goals 2: Now we cluster the data using k-prototypes, as this will allow us to handle a mix of continuous and categorical data. We will use start city, final destination, type of trip, group size, whether it is a group, how far in advance the ticket was purchased, booking channel, and membership status to try and define the groups. First, we will plot the sum of squared errors curve to try and find the optimal number of clusters. Then, we will loop through the clustering several times to try and get a sense of generalized performance, as k-prototypes chooses random starting points for the clustering.

```
prototype_data <- customer_data_clust %>%
  select(First_City, Final_Destination, round_trip,
         group_size, group, days_pre_booked,
         BookingChannel, UflyMemberStatus) %>% as.data.frame()
```

```
SSE_curve <- c()
for (i in 1:10){
  sse_avg <- c()
  k = i
  for (t in 1:5){
    kpro <- kproto(prototype_data, k)
    sse <- sum(kpro$withinss)
    sse_avg[k] <- sse
  }
  SSE_curve[i] <- mean(sse_avg, na.rm = T)}
```

```
plot(1:10, SSE_curve, type="b", xlab="Number of Clusters", ylab="SSE")
```



Interpretation from Approach: Based on running our analysis on multiple samples, there seems to be a general cutoff point around 3 clusters. You could make an argument for a higher amount of clusters based on the elbow plot, but based on our business assumptions and other evidence from the hierarchical dendrogram we think 3 clusters is a good choice.

Conclusions from Approach: Now that we have chosen 3 as our number of clusters we can then run the clustering again and view the clusters.

5 Exploring the clusters generated

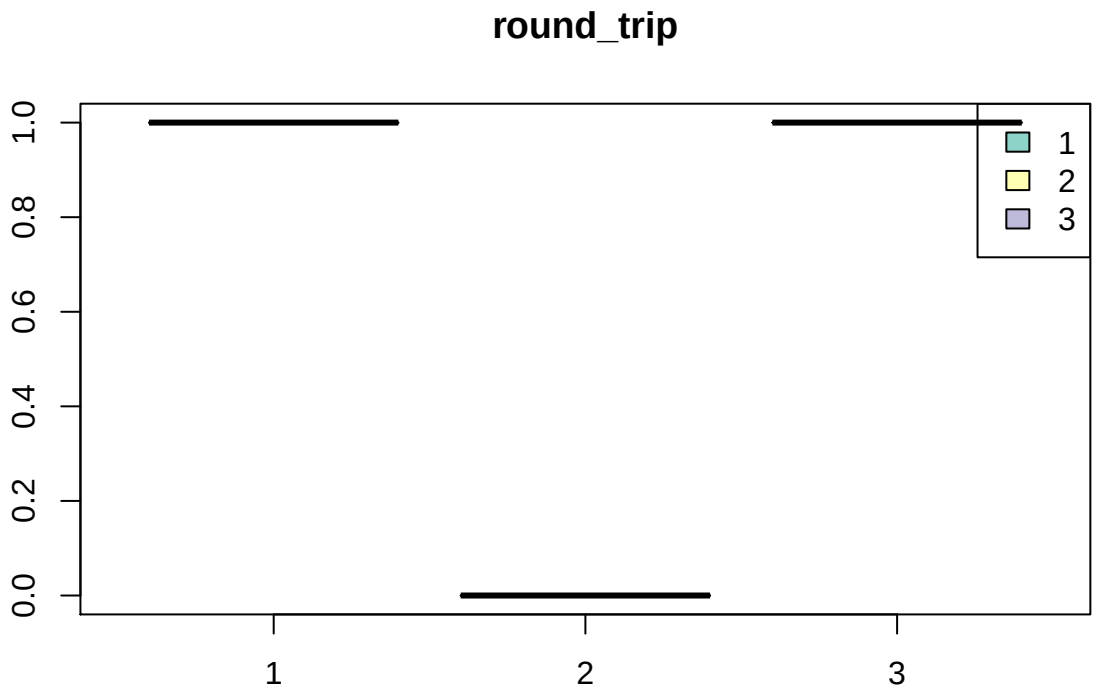
5.1 Visualization of cluster attributes

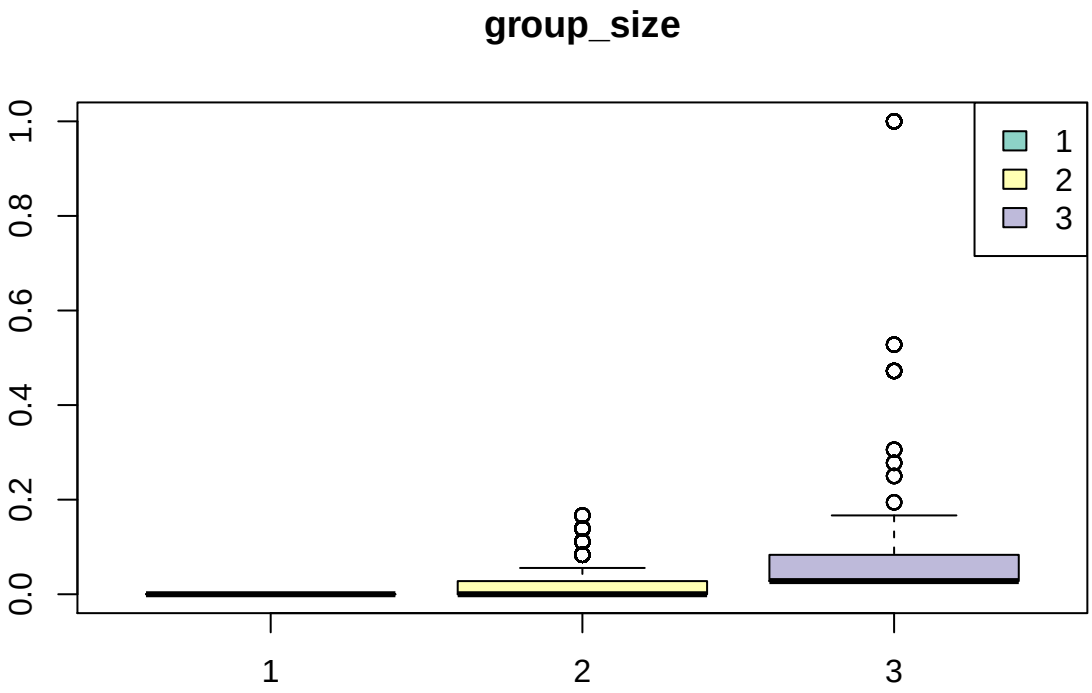
Explanation of Approach and Goals 3: Here we will cluster the data and try to initially visualize the results (Every time we run the clustering algorithm the results change as k-prototypes start with a random set of points. All the interpretation are based on the intial set of analysis.)

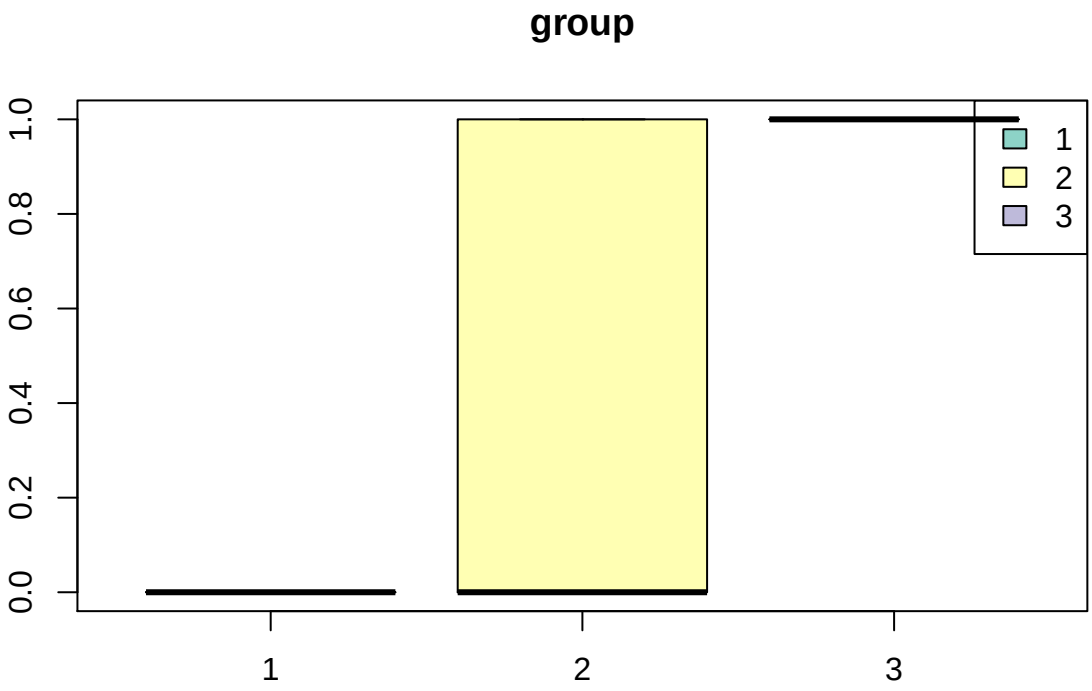
```
num_proto_clusters = 3
kpro <- kproto(as.data.frame(prototype_data), num_proto_clusters)
```

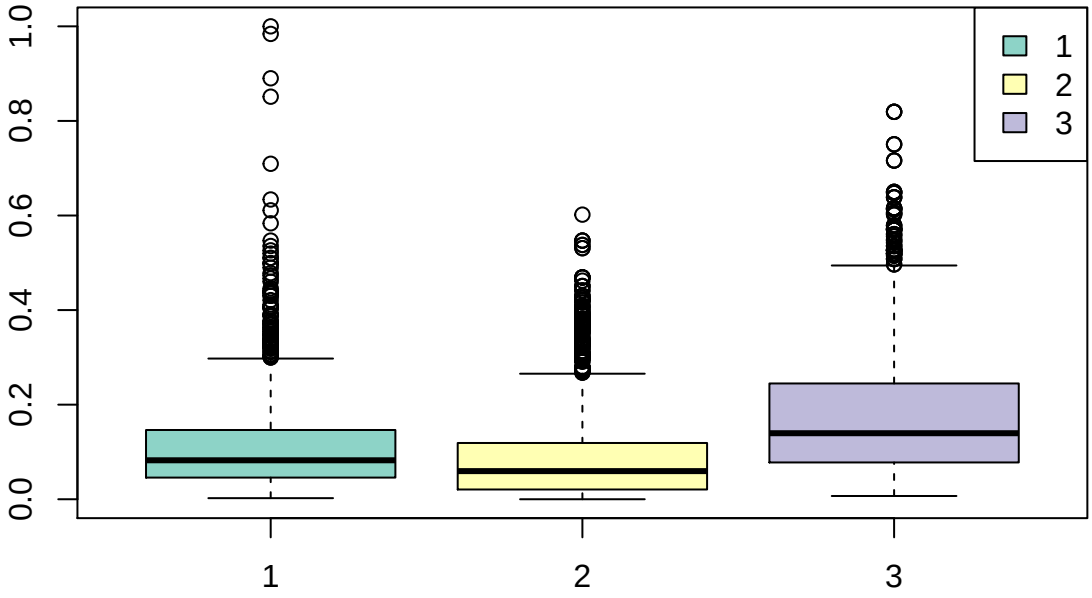
```
## # NAs in variables:
##        First_City Final_Destination        round_trip        group_size
##                 0                 0                 0                 0
##             group   days_pre_booked    BookingChannel  UflyMemberStatus
##                 0                 0                 0                 0
## 0 observation(s) with NAs.
##
## Estimated lambda: 0.1983115
```

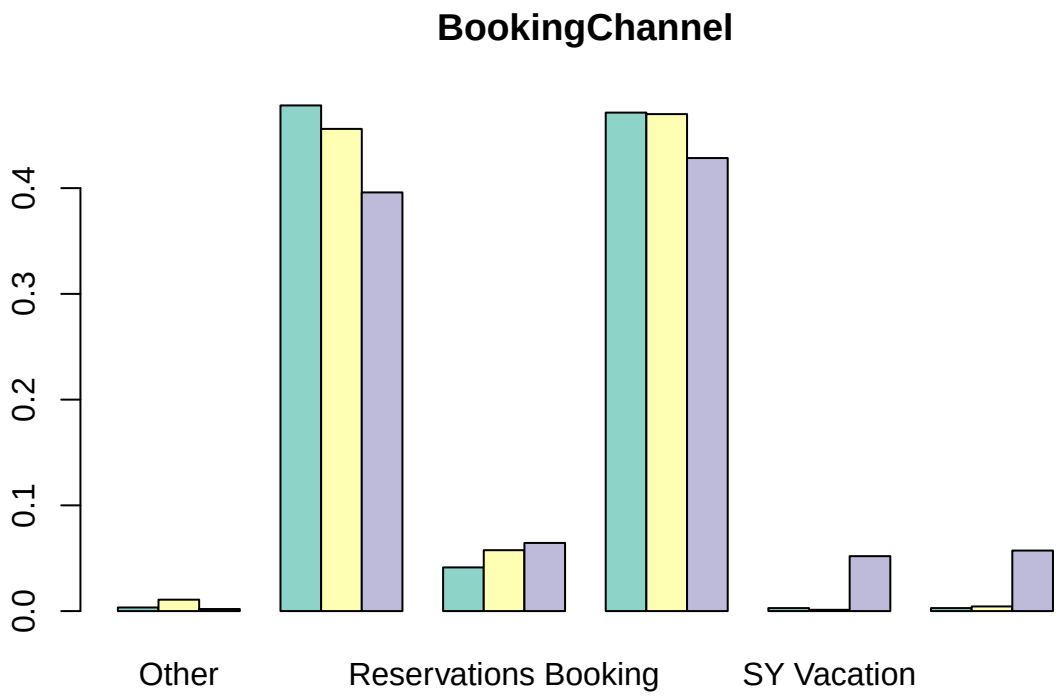
```
clprofiles(kpro, as.data.frame(prototype_data))
```




group_size
0.0
0.2
0.4
0.6
0.8
1.0
1
2
3
1
2
3


group
0.0
0.2
0.4
0.6
0.8
1.0
1
2
3
1
2
3


days_pre_booked
0.0
0.2
0.4
0.6
0.8
1.0
1
2
3
1
2
3



Interpretation from Approach: The data here is still normalized so it's hard to interpret some of the differences in the continuous variables, but we can see that there are some differences. Cluster 1 is mostly round-trip tickets whereas cluster 2 is individual travelers. Days booked in advance also have differences that look like they could be significant.

Conclusions from Approach: As an initial exploration, we can see that there are differences between the clusters. Furthur visualizations will help us gain a better understanding of these differences.

Explanation of Approach and Goals 4: Because the goal of our analysis is to understand our customers, we want to merge our clusters back into our sample data to examine the data at a lower customer level instead of a trip level.

```
data_transformed$cluster <- kpro$cluster
final_segments <- merge(suncountry, data_transformed, by = 'PNRLocatorID')
```

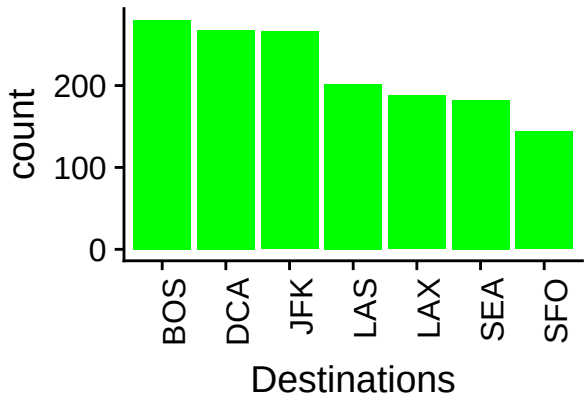
Interpretation from Approach: Now that our sample data has a 'cluster number' column with which we can examine the differences in between clusters in variables that we didn't cluster on.

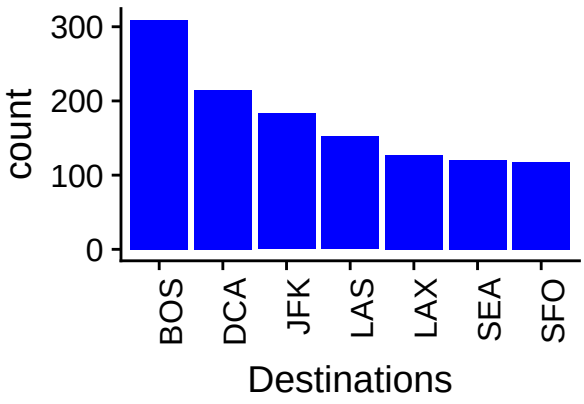
5.2 Top travel destinations by cluster

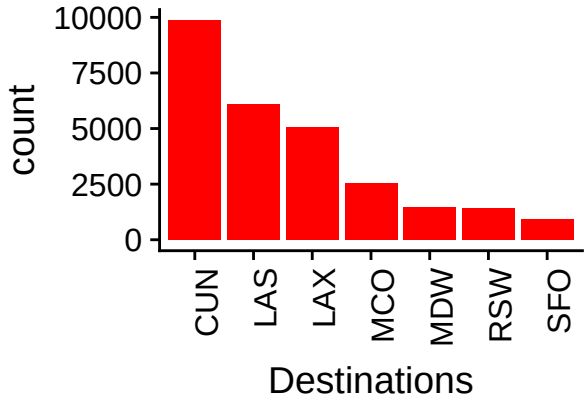
Explanation of Approach and Goals 5: Now, we are going to plot where are people in different clusters going to on a normal basis.

```
cl1_cities <- final_segments %>%
  filter(cluster == 1 & Final_Destination != 'MSP') %>%
  select(Final_Destination) %>%
  group_by(Final_Destination) %>%
  summarise(count = n()) %>%
```

```
  arrange(count) %>%
  top_n(7) %>%
  ggplot() + geom_bar(aes(sort(Final_Destination, decreasing = T), count),
                      stat = 'identity', fill = 'green') + labs(x = 'Destinations') +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))

cl2_cities <- final_segments %>%
  filter(cluster == 2 & Final_Destination != 'MSP') %>%
  select(Final_Destination) %>%
  group_by(Final_Destination) %>%
  summarise(count = n()) %>%
  arrange(count) %>%
  top_n(7) %>%
  ggplot() + geom_bar(aes(sort(Final_Destination, decreasing = T), count),
                      stat = 'identity', fill = 'blue') + labs(x = 'Destinations')+
  theme(axis.text.x = element_text(angle = 90, hjust = 1))

cl3_cities <- final_segments %>%
  filter(cluster == 3 & Final_Destination != 'MSP') %>%
  select(Final_Destination) %>%
  group_by(Final_Destination) %>%
  summarise(count = n()) %>%
  arrange(count) %>%
  top_n(7) %>%
  ggplot() + geom_bar(aes(sort(Final_Destination, decreasing = T), count),
                      stat = 'identity', fill = 'red') + labs(x = 'Destinations')+
  theme(axis.text.x = element_text(angle = 90, hjust = 1))

plot_grid(cl1_cities, cl2_cities, cl3_cities)
```







Interpretation from Approach: Cluster 3 is flying to locations like Cancun, Las Vegas, LA, and Orlando which suggests these might be vacationers. Cluster 2 flies most often to cities like Boston, D.C, New York. Considering that these are solo travelers it is possible they are flying for business or other non-vacation reason. Cluster 1 and two both have the top 7 destinations as same cities. It is less clear what this could mean.

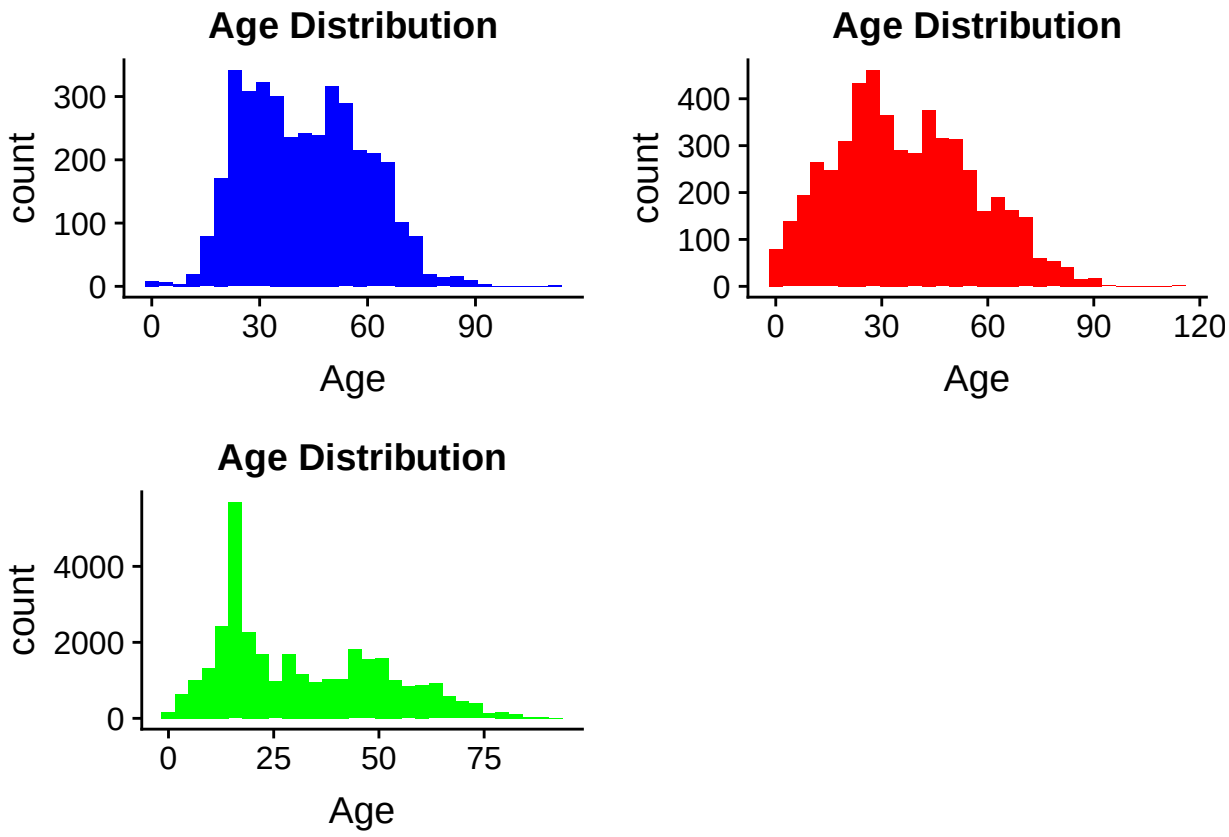
Conclusions from Approach: The clusters are exhibiting differences in destinations that they are visiting, we can now further see what defines the segments.

5.3 Age distribution by cluster

Explanation of Approach and Goals 6: We want to look at the age distribution to see if we can see any insights. Based on the destinations and group sizes of the clusters, we think that cluster 1 could be families looking to vacation and cluster 2 could be solo travelers going for business or other reasons.

```
h1 <- final_segments %>%
  filter(cluster == 1) %>%
  select(Age.x) %>%
  ggplot() + geom_histogram(aes(Age.x), fill = 'blue') +
  labs(title = 'Age Distribution', x = 'Age')

h2 <- final_segments %>%
  filter(cluster == 2) %>%
  select(Age.x) %>%
  ggplot() + geom_histogram(aes(Age.x), fill = 'red') +
  labs(title = 'Age Distribution', x = 'Age')

h3 <- final_segments %>%
```

```
  filter(cluster == 3) %>%
  select(Age.x) %>%
  ggplot() + geom_histogram(aes(Age.x), fill = 'green') +
  labs(title = 'Age Distribution', x = 'Age')

plot_grid(h1, h2, h3)
```



Interpretation from Approach: Cluster31 has 2 distinct groups which suggest that these represent parents and their children. Cluster 2 is clearly more centered around the working ages of 25-50 which could mean that these are that "business" group which we think is there. Again, cluster 3 seems to be somewhat of a mix, but it is worth noting that it captures more of the teaenagers than the other groups.

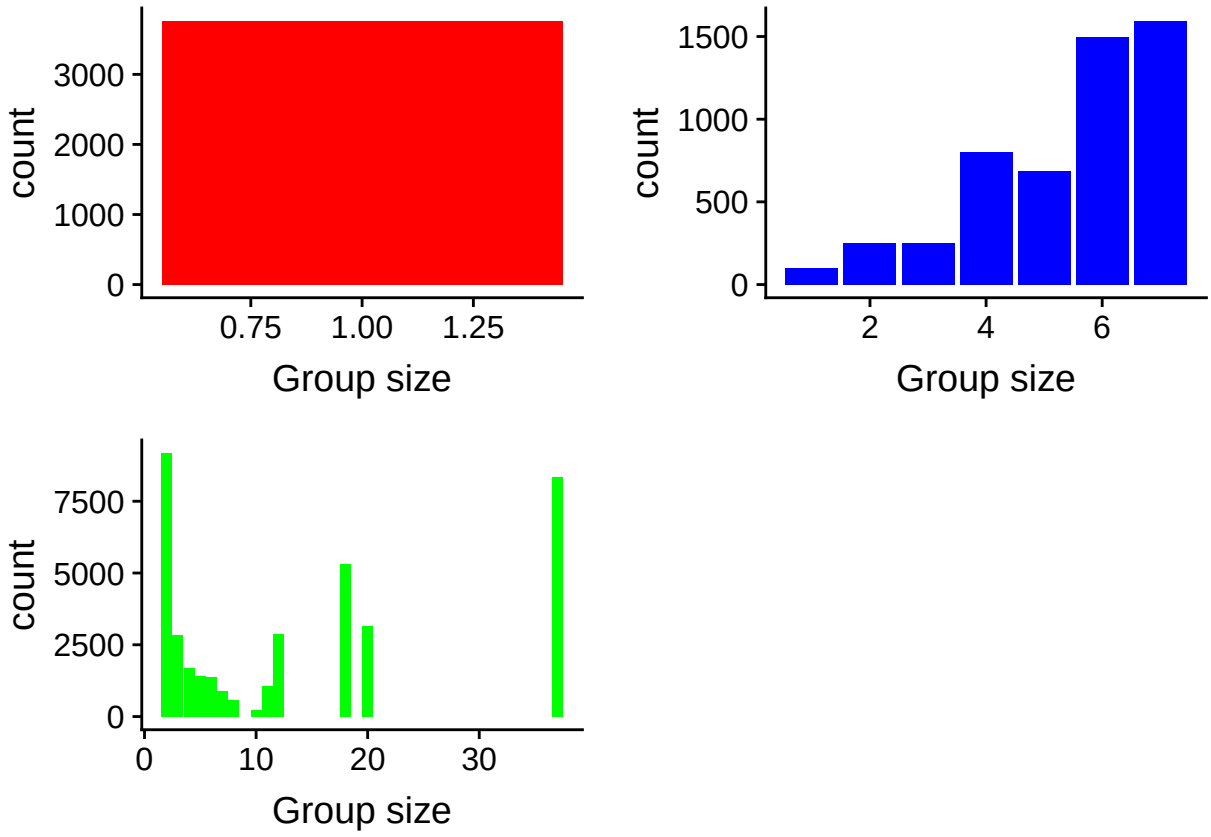
Conclusions from Approach: We now can say that cluster 3 appears to be families, cluster 2 solo travelers for business or other reasons, and cluster 1 looks like a mix, possibly some other traveling groups like tours and also some "snow birds" escaping the winter.

5.4 Group size by cluster

Explanation of Approach and Goals 7: After the clustering we saw the differences in group size with the normalized data, now we can look at it in real terms to get a sense of the distributions.

```
groups_c1 <- final_segments %>%
  filter(cluster == 1) %>%
  select(group_size) %>%
  group_by(group_size) %>%
  summarise(count = n()) %>%
  ggplot() + geom_bar(aes(sort(group_size, decreasing = T), count),
```

```
                      stat = 'identity', fill = 'red') + labs(x = 'Group size')

groups_c2 <- final_segments %>%
  filter(cluster == 2) %>%
  select(group_size) %>%
  group_by(group_size) %>%
  summarise(count = n()) %>%
  ggplot() + geom_bar(aes(sort(group_size, decreasing = T), count),
                      stat = 'identity', fill = 'blue') + labs(x = 'Group size')

groups_c3 <- final_segments %>%
  filter(cluster == 3) %>%
  select(group_size) %>%
  group_by(group_size) %>%
  summarise(count = n()) %>%
  ggplot() + geom_bar(aes(sort(group_size, decreasing = T), count),
                      stat = 'identity', fill = 'green') + labs(x = 'Group size')

plot_grid(groups_c1,groups_c2,groups_c3)
```



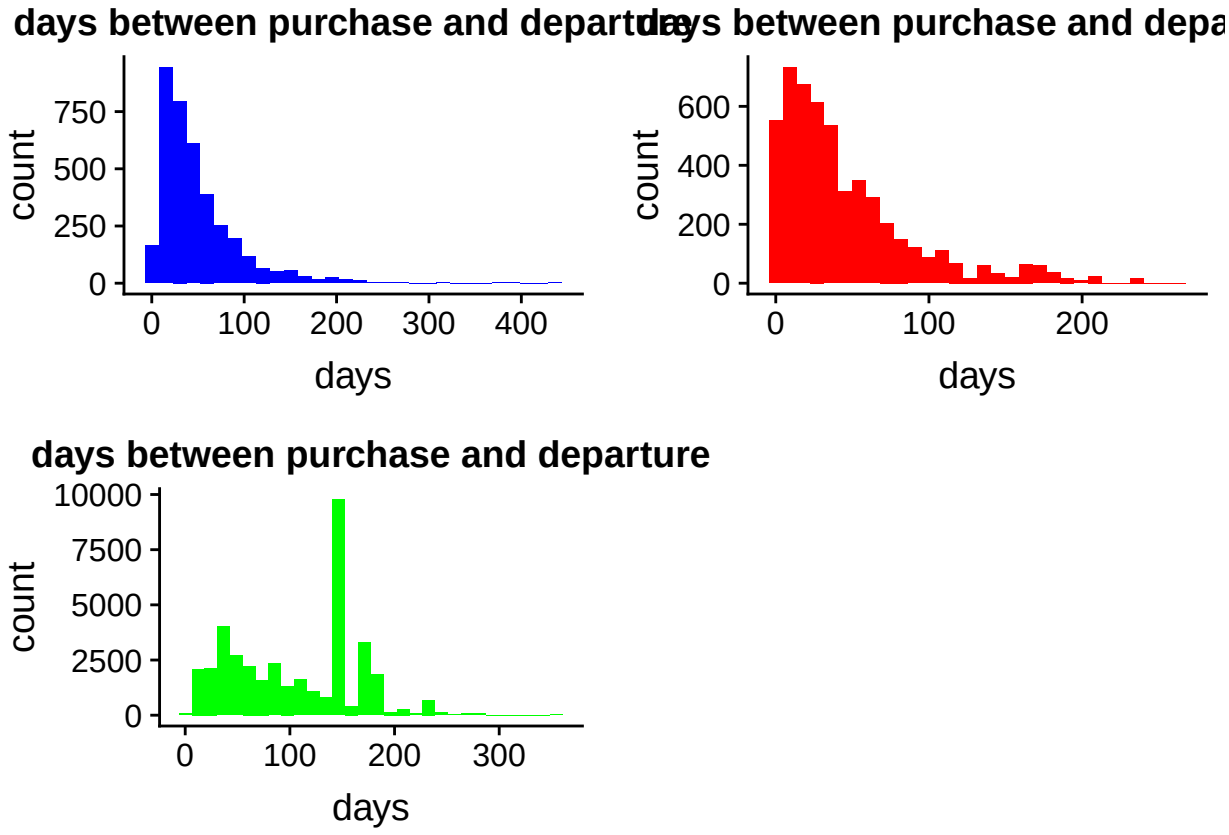
Interpretation from Approach: Cluster 3 has a large outlier that skews the results, but it looks like it matches the distribution for family travel. Cluster 2 is skewed. Cluster 1 is all single travelers.

Conclusions from Approach: In terms of group size, clusters behave differently. We get one for solo travelers which is uniform . This weakens our argument that cluster 3 is families somewhat, but we think this could just be due to noise.

5.5 How many days in advance tickets are booked

Explanation of Approach and Goals 8: To understand consumer behavior, it is important to see whether groups behave differently in terms of when they book their tickets. Here we will plot a distribution of the days in advance of the trip for ticket purchase.

```
h1 <- final_segments %>%
  filter(cluster == 1) %>%
  select(days_pre_booked) %>%
  ggplot() + geom_histogram(aes(days_pre_booked), fill = 'blue') +
  labs(title= 'days between purchase and departure', x = 'days')

h2 <- final_segments %>%
  filter(cluster == 2) %>%
  select(days_pre_booked) %>%
  ggplot() + geom_histogram(aes(days_pre_booked), fill = 'red') +
  labs(title= 'days between purchase and departure', x = 'days')

h3 <- final_segments %>%
  filter(cluster == 3) %>%
  select(days_pre_booked) %>%
  ggplot() + geom_histogram(aes(days_pre_booked), fill = 'green') +
  labs(title= 'days between purchase and departure', x = 'days')

plot_grid(h1, h2, h3)
```



Interpretation from Approach: Cluster 3 books well in advance, with most of the records coming more than

150 days in advance. Cluster 2 is booked closer to departure date.

Conclusions from Approach: Cluster 3 makes sense because families like to plan their vacations well in advance because of limited vacation time. Cluster 2 make sense as people in this cluster are traveling for business reasons too.

6 Final Word

6.1 Conclusion

We have identified 3 main segments that your customer base is comprised of. They are Families, Solo Travelers, and Snowbirds. These groups fly to different destinations, have different flights needs, and different demographics. Targeting these groups separately based on their needs and behaviors will allow you to gain a competitive edge over the other airlines in the industry. Going forward we recommend scaling the analysis to all your data. Also, for future data analysis purpose, we recommend investigating the data collection process to eliminate some of the errors, as well as generating more data in general.