Southeast Airline Analysis

IST 687

Introduction to Data Science

Name : Sonamgyalpo Sherpa

SUID - 398497433

INDEX

Table of Contents

Introduction

Southeast Airlines is a leading carrier in the United States and as of 2014, has partnered with 14 airlines. The airline aims to lower customer churn(or customer attrition) in order to retain more loyal passengers to their customer base. Till Recent times, Southeast believed the best way to apply Customer Churn is through a robust Loyalty Program. But unfortunately, this was just an "accepted industry best practice".

Moreover, Customer Churn was a lagging indicator, meaning the loss had already occurred. In order to better understand the sentiments of the passengers and understand how to retain them by behavioral patterns in buying services and customer features, we can use NPS to understand customer behavior.

NPS is the ***Net Promoter Score*** and research has indicated NPS as three times more sensitive than at predicting customer churn than customer satisfaction. NPS is divided into three stages and this methodology will be used to gather insights in the upcoming pages :-

1) Promoter ~ (9-10) " Provide free of mouth advertising, good customers to keep"
2) Neutral ~ (7-8)
3) Detractor ~ (1-6) " Really problematic, may spread rumors to decrease customer base"

South east airlines have collected data from passengers in the form of ***Surveys*** and these surveys can be used to predict the NPS scores by varying metrics such as market segments and behavioral patterns of passengers.

Dataset Metrics

Attributes:

1. ***Likelihood to Recommend –*** *rated on a scale of 1 to 10, which shows how likely the customer is to recommend the airline to their friends (10 is very likely, and 1 is not very likely).*
2. ***Airline Flyer Status*** *– each customer has a different type of airline status, which are platinum, gold, silver, and blue (based on level of travel with the airline)*
3. ***Age*** *– the specific customer's age. Ranging from 15 to 85 years old.*
4. ***Gender*** *– male or female.*
5. ***Price Sensitivity*** *– the grade to which the price affects to customers purchasing. The price sensitivity has a range from 0 to 5.*
6. ***Year of First Flight*** *– this attribute shows the first flight of each single customer. The range of year of the first flight for each customer has been started in 2003 until 2012.*
7. ***Flights Per Year*** *– The number of flights that each customer has taken in the most recent 12 months. The range starting from 0 to 100.*
8. ***Loyalty*** *– An index of loyalty ranging from -1 to 1 that reflects the proportion of flights taken on other airlines versus flights taken on this airline. A higher index means more loyalty.*
9. ***Type of Travel*** *– One of business travel, mileage tickets, or personal travel (ex. vacation)*
10. ***Total Frequent Flyer Accounts*** *– How many frequent flyer accounts the customer has.*
11. ***Shopping Amount at Airport*** *– The spending on non-food & services at the airport (in $)*
12. ***Eating and Drinking at Airport*** *– The spending on food/drink at the airport (in $).*
13. ***Class*** *– three different kinds of service level (business, economy plus, and economy).*
14. ***Day of Month*** *–the traveling day of each costumer (ranges from 1 to 31).*
15. ***Flight date*** *– the passenger's flight date of travel.*
16. ***Partner Code*** *– This airline works with wholly- and partially-owned subsidiary companies to deliver regional flights. For example, AA, AS, B6, and DL.*
17. ***Partner Name*** *– These are the full names of the partner airline companies.*
18. ***Origin City*** *– the place where passenger departed from. For example, Boston MA.*
19. ***Origin State*** *– the place where passenger departed from. For example, Texas.*
20. ***Destination City*** *– the place to which passenger travels to. For example, Boston MA.*
21. ***Destination State*** *– the place to which passenger travels to. For example, Texas.*
22. ***Scheduled Departure Hour*** *– the specific time at which the plane was scheduled to depart.*
23. ***Departure Delay in Minutes*** *– How long the flight's departure was delayed, when compared to schedule.*
24. ***Arrival Delay in Minutes*** *– How long the arrival was delayed.*
25. ***Flight Cancelled*** *– occurs when the airline does not operate the flight.*
26. ***Flight time in minutes*** *–the length of time, in minutes, to reach the destination.*
27. ***Flight Distance*** *– the distance between the departure and arrival destination.*
28. ***Comment*** *– a free form text field of the passenger comment, with respect to the flight.*

Business Questions

1)What behavioral and geographical market segments can be selected from passenger data to determine their Likelihood to recommend the airline to other potential customers?

2)What are the customer sentiments for the passengers with highest loyalty to the airline?

3)Does Price of airline services, gender, purposes for travel (Personal, business) affect the NPS scores of the customer?

4)How satisfied are the customers who have high airline flyer status as compared to customers who have low airline flyer status?

5)Are there any patterns in NPS score, when two or more customer behaviors are brought into consideration?

6)How well are the Partner Airlines performing and which Airlines provide low customer satisfaction?

7)What kind of passenger is most likely to be a promoter or a detractor?

8)What are the lowest satisfaction routes(lowest nps) in the dataset?

9)Which states have the Highest and lowest NPS scores?

****All the above questions are solved in the upcoming visualizations and Machine Learning Models.**

PHASE 01

Libraries Used

The following libraries have been used for the Project

1) RCurl
2) Jsonlite
3) Dplyr
4) Tidyverse
5) imputeTS
6) kernlab
7) ggplots
8) ggmap
9) arules
10) arulesviz
11) caTools

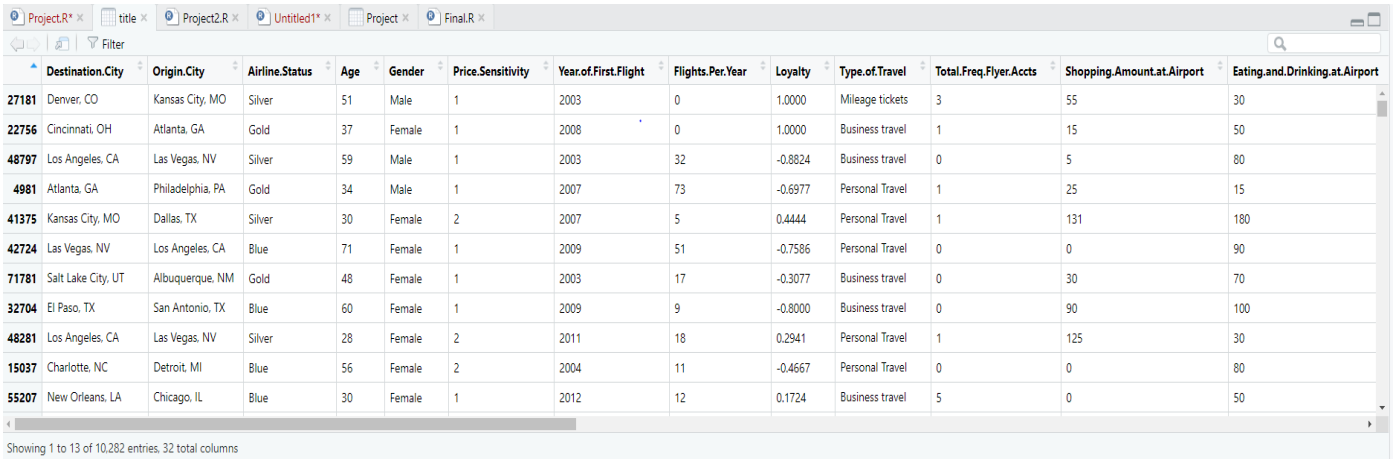
Importing the Data

Code Snippet

```
airURL <- "https://s3.us-east-1.amazonaws.com/blackboard.learn.xythos.prod/5956621d575cd/9644551?response-content-di
apiresult<- getURL(airURL)
results <- fromJSON(apiresult)

head(results)
```

The file is saved in Json format. Using the URL function the dataset has been imported into RStudio and stored in a dataframe – "**results**"

Output



Project.R* | title | Project2.R | Untitled1* | Project | Final.R

Filter

	Destination.City	Origin.City	Airline.Status	Age	Gender	Price.Sensitivity	Year.of.First.Flight	Flights.Per.Year	Loyalty	Type.of.Travel	Total.Freq.Flyer.Accts	Shopping.Amount.at.Airport	Eating.and.Drinking.at.Airport
27181	Denver, CO	Kansas City, MO	Silver	51	Male	1	2003	0	1.0000	Mileage tickets	3	55	30
22756	Cincinnati, OH	Atlanta, GA	Gold	37	Female	1	2008	0	1.0000	Business travel	1	15	50
48797	Los Angeles, CA	Las Vegas, NV	Silver	59	Male	1	2003	32	-0.8824	Business travel	0	5	80
4981	Atlanta, GA	Philadelphia, PA	Gold	34	Male	1	2007	73	-0.6977	Personal Travel	1	25	15
41375	Kansas City, MO	Dallas, TX	Silver	30	Female	2	2007	5	0.4444	Personal Travel	1	131	180
42724	Las Vegas, NV	Los Angeles, CA	Blue	71	Female	1	2009	51	-0.7586	Personal Travel	0	0	90
71781	Salt Lake City, UT	Albuquerque, NM	Gold	48	Female	1	2003	17	-0.3077	Business travel	0	30	70
32704	El Paso, TX	San Antonio, TX	Blue	60	Female	1	2009	9	-0.8000	Business travel	0	90	100
48281	Los Angeles, CA	Las Vegas, NV	Silver	28	Female	2	2011	18	0.2941	Personal Travel	1	125	30
15037	Charlotte, NC	Detroit, MI	Blue	56	Female	2	2004	11	-0.4667	Personal Travel	0	0	80
55207	New Orleans, LA	Chicago, IL	Blue	30	Female	1	2012	12	0.1724	Business travel	5	0	50

Showing 1 to 13 of 10,282 entries, 32 total columns

Data Cleaning

After importing the dataset and using the view(results) function, it is observed that the dataset has 10282 observations and 32 variables, example :- age, gender, loyalty index etc…

Using the sum(is.na(results)), it is observed that there are **11,435** missing values in the dataset. The number of missing values in each column of the dataset is displayed.

Code Snippet

```
> colSums(is.na(results))
              Destination.City                   Origin.City                Airline.Status
                            26                            26                            26
                           Age                        Gender             Price.Sensitivity
                            26                            26                            26
         Year.of.First.Flight               Flights.Per.Year                       Loyalty
                            26                            26                            26
                Type.of.Travel         Total.Freq.Flyer.Accts      Shopping.Amount.at.Airport
                            26                            26                            26
Eating.and.Drinking.at.Airport                         Class                  Day.of.Month
                            26                            26                            26
                   Flight.date                  Partner.Code                  Partner.Name
                            26                            26                            26
                Origin.State               Destination.State       Scheduled.Departure.Hour
                            26                            26                            26
    Departure.Delay.in.Minutes        Arrival.Delay.in.Minutes             Flight.cancelled
                           217                           245                            26
        Flight.time.in.minutes               Flight.Distance        Likelihood.to.recommend
                           245                            26                            26
                         olong                          olat                         dlong
                            26                            26                            26
                          dlat                      freeText                  Flight_Month
                            26                         10000                            26
```

These values are handled in the following steps. The other cleaning steps done are :-

1.) Deleting *redundant* column values
2.) *Data Extraction* (Creation of new columns for analysis)

Step 1

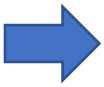
Deleting Initials in *Flight Destination city* and *Flight origin city* columns

Code Snippet

```
## Regular expressions to remove statename initials from the column names - destination city and state city
results$Destination.City <- gsub("(.*),.*", "\\1",results$Destination.City)
results$Origin.City <- gsub("(.*),.*", "\\1",results$Origin.City)
```

Output

Destination.City	Origin.City
Denver, CO	Kansas City, MO
Cincinnati, OH	Atlanta, GA
Los Angeles, CA	Las Vegas, NV
Atlanta, GA	Philadelphia, PA
Kansas City, MO	Dallas, TX



Destination.City	Origin.City
Denver	Kansas City
Cincinnati	Atlanta
Los Angeles	Las Vegas
Atlanta	Philadelphia
Kansas City	Dallas

This can be utilized in creating concise visualizations with ggmap and ggplot and to decrease clutter due to long names in bar charts or histograms.

Step 2

Extraction of flight months from Flight.date to categorize passengers and their ratings, loyalty scores etc... in the basis of their month of travel (in this case, either – January, February or March)

Code Snippet

```
## Now we extract month from flight dates and create a new column to show months
results$Flight.date <- as.Date(results$Flight.date, format = "%m/%d/%y") # convert flight date from chr to date
library(data.table)
setDT(results)[, Flight_Month := format(as.Date(Flight.date), "%m") ]
results$Flight_Month <- as.numeric(results$Flight_Month)
# We now have a new column for months and we can do analysis on a monthly basis
table(results$Flight_Month)
```

It is necessary to convert the flight month to numeric type. Since, it is stored to “string” type by default

Output

```
   1    2    3
3483 3062 3711
```

Here, the least number of flights are in the month of ***February***. Nevertheless, the data is almost evenly distributed across three months.

Step 3 – Numeric Missing values

To deal with the 26 missing values in the Loyalty column, we can use linear interpolation to impute the missing values.

Code Snippet

```
## dealing with numeric missing values with interpolation
library(imputeTS)
results$Loyalty <- na_interpolation(results$Loyalty)
```

Step 4 – Categorical Missing values

Now we deal with categorical missing values in the dataset, the maximum number of missing values in the dataset are categorical variables, as will be shown in the following code snippet.

Code Snippet

```
##  dealing missing values in necesesary categorical attributes except column 'free text'
 results <- results%>%
   mutate(Destination.City = replace(Destination.City, is.na(Destination.City), "N/A"))

 results <- results%>%
   mutate(Origin.City = replace(Origin.City, is.na(Origin.City), "N/A"))


 results <- results%>%
   mutate(Partner.Code = replace(Partner.Code, is.na(Partner.Code), "N/A"))

 results <- results%>%
   mutate(Partner.Name = replace(Partner.Name, is.na(Partner.Name), "N/A"))

 results <- results%>%
   mutate(Origin.State = replace(Origin.State, is.na(Origin.State), "N/A"))

 results <- results%>%
   mutate(Destination.State = replace(Destination.State,is.na(Destination.State), "N/A"))

results <- results%>%
   mutate(Flight.cancelled = replace(Flight.cancelled, is.na(Flight.cancelled), "N/A"))

 results <- results%>%
   mutate(freeText = replace(freeText, is.na(freeText), "N/A"))

 results <- results%>%
   mutate(Flight.cancelled = replace(Flight.cancelled, is.na(Flight.cancelled), "No"))
```

We can observe, from the above code, that using pipelines and the mutate function, all missing categorical variables in the “results” dataset have been replaced with “N/A” values.

Output

```
> sum(is.na(results))
[1] 1279
> results <- na.omit(results)
> sum(is.na(results))
[1] 0
```

The current number of missing values has been reduced from 11,435 to 1,279. Lastly, we omit the remaining 1,279 missing values from the dataset. Since, the number of missing values is not significant as compared to the size of the entire dataset. The result gives us “***0***” missing values in the dataset!!!

RESULTS OF DATA CLEANING

1.) The Destination City and the Origin city columns have been cleaned using *gsub* (regular expressions) to replace state name initials eg:- CO,NY,CA.
2.) All the numeric missing values have been *interpolated*.
3.) All the missing values in the categorical variables for the regression model are replaced with "*N/A*" Or required values.
4.) The flight date column has been converted from chr to date.
5.) A new *Flight_month* column has been created. Since, the dataset is of the year 2014 only and analysis can be done on a monthly basis.

***This attribute is not included in the Machine Learning model and for any visualizations:-

1.) freetext - due to large number of missing values .

DataType Transformations

Step 1 – View and convert data types

Code Snippet

```
> str(results)
'data.frame':   10037 obs. of  33 variables:
 $ Destination.City              : chr  "Denver" "Cincinnati" "Los Angeles" "Atlanta" ...
 $ Origin.City                   : chr  "Kansas City" "Atlanta" "Las Vegas" "Philadelphia" ...
 $ Airline.Status                : chr  "Silver" "Gold" "Silver" "Gold" ...
 $ Age                           : int  51 37 59 34 30 71 48 60 28 56 ...
 $ Gender                        : chr  "Male" "Female" "Male" "Male" ...
 $ Price.Sensitivity             : int  1 1 1 1 2 1 1 1 2 2 ...
 $ Year.of.First.Flight          : int  2003 2008 2003 2007 2007 2009 2003 2009 2011 2004 ...
 $ Flights.Per.Year              : int  0 0 32 73 5 51 17 9 18 11 ...
 $ Loyalty                       : num  1 1 -0.882 -0.698 0.444 ...
 $ Type.of.Travel                : chr  "Mileage tickets" "Business travel" "Business travel" "Personal Travel" ...
 $ Total.Freq.Flyer.Accts        : int  3 1 0 1 1 0 0 0 1 0 ...
 $ Shopping.Amount.at.Airport    : int  55 15 5 25 131 0 30 90 125 0 ...
 $ Eating.and.Drinking.at.Airport: int  30 50 80 15 180 90 70 100 30 80 ...
 $ Class                         : chr  "Business" "Eco" "Eco" "Eco" ...
 $ Day.of.Month                  : int  5 18 25 27 27 12 23 13 5 4 ...
 $ Flight.date                   : Date, format: "2014-01-05" "2014-03-18" "2014-01-25" "2014-01-27" ...
 $ Partner.Code                  : chr  "WN" "DL" "OO" "US" ...
 $ Partner.Name                  : chr  "Cheapseats Airlines Inc." "Sigma Airlines Inc." "Northwest Business Airlines
Inc." "Southeast Airlines Co." ...
 $ Origin.State                  : chr  "Missouri" "Georgia" "Nevada" "Pennsylvania" ...
 $ Destination.State             : chr  "Colorado" "Kentucky" "California" "Georgia" ...
 $ Scheduled.Departure.Hour      : int  8 7 10 14 18 11 12 9 9 7 ...
 $ Departure.Delay.in.Minutes    : int  38 0 0 0 24 0 0 0 0 0 ...
 $ Arrival.Delay.in.Minutes      : int  66 0 0 0 21 0 0 1 0 0 ...
 $ Flight.cancelled              : chr  "No" "No" "No" "No" ...
 $ Flight.time.in.minutes        : int  86 60 42 110 65 38 76 77 41 71 ...
 $ Flight.Distance               : int  533 373 236 666 460 236 493 496 236 500 ...
 $ Likelihood.to.recommend       : int  9 9 10 6 7 8 10 9 7 4 ...
 $ olong                         : num  -94.6 -84.3 -115.2 -75.3 -97 ...
 $ olat                          : num  39 33.8 36.1 40 32.8 ...
 $ dlong                         : num  -105 -84.5 -118.1 -84.3 -94.6 ...
 $ dlat                          : num  39.7 39.2 34 33.8 39 ...
 $ freeText                      : chr  "N/A" "N/A" "N/A" "N/A" ...
 $ Flight_Month                  : num  1 3 1 1 1 3 3 3 2 1 ...
```

In the above code snippet, we notice that maximum number of values are separated into integers(int,num) or strings(chr), with an exception of one column with Data Format.

Post Speculation, it is observed that several int and chr variables can be converted into “factor” data type(data distributed in levels). This creation of levels will ensure in refined visualizations, using Charts and feasibility in Machine Learning models

Code Snippet

```
## Now we encode the categorical variables into a factor format for application in Predictive models

results<- results %>%
  mutate(Airline.Status = as.factor(Airline.Status),Gender = as.factor(Gender), Flight.cancelled= as.factor(Flight.cancelled),
         Type.of.Travel = as.factor(Type.of.Travel),Class = as.factor(Class),Price.Sensitivity = as.factor(Price.Sensitivity),
                     Partner.Name = as.factor(Partner.Name))
str(results)
```

The necessary columns have been transformed to factor type. All the selected columns can be encoded into 2 or more levels.

Output

```
> str(results)
'data.frame':   10037 obs. of  33 variables:
 $ Destination.City             : chr  "Denver" "Cincinnati" "Los Angeles" "Atlanta" ...
 $ Origin.City                  : chr  "Kansas City" "Atlanta" "Las Vegas" "Philadelphia" ...
 $ Airline.Status               : Factor w/ 4 levels "Blue","Gold",..: 4 2 4 2 4 1 2 1 4 1 ...
 $ Age                          : int  51 37 59 34 30 71 48 60 28 56 ...
 $ Gender                       : Factor w/ 2 levels "Female","Male": 2 1 2 2 1 1 1 1 1 1 ...
 $ Price.Sensitivity            : Factor w/ 5 levels "0","1","2","3",..: 2 2 2 2 3 2 2 2 3 3 ...
 $ Year.of.First.Flight         : int  2003 2008 2003 2007 2007 2009 2003 2009 2011 2004 ...
 $ Flights.Per.Year             : int  0 0 32 73 5 51 17 9 18 11 ...
 $ Loyalty                      : num  1 1 -0.882 -0.698 0.444 ...
 $ Type.of.Travel               : Factor w/ 3 levels "Business travel",..: 2 1 1 3 3 3 1 1 3 3 ...
 $ Total.Freq.Flyer.Accts       : int  3 1 0 1 1 0 0 0 1 0 ...
 $ Shopping.Amount.at.Airport   : int  55 15 5 25 131 0 30 90 125 0 ...
 $ Eating.and.Drinking.at.Airport: int  30 50 80 15 180 90 70 100 30 80 ...
 $ Class                        : Factor w/ 3 levels "Business","Eco",..: 1 2 2 2 2 2 2 2 2 2 ...
 $ Day.of.Month                 : int  5 18 25 27 27 12 23 13 5 4 ...
 $ Flight.date                  : Date, format: "2014-01-05" "2014-03-18" "2014-01-25" "2014-01-27" ...
 $ Partner.Code                 : chr  "WN" "DL" "OO" "US" ...
 $ Partner.Name                 : Factor w/ 14 levels "Cheapseats Airlines Inc.",..: 1 12 8 13 1 1 8 1 10 4 ...
 $ Origin.State                 : chr  "Missouri" "Georgia" "Nevada" "Pennsylvania" ...
 $ Destination.State            : chr  "Colorado" "Kentucky" "California" "Georgia" ...
 $ Scheduled.Departure.Hour     : int  8 7 10 14 18 11 12 9 9 7 ...
 $ Departure.Delay.in.Minutes   : int  38 0 0 0 24 0 0 0 0 0 ...
 $ Arrival.Delay.in.Minutes     : int  66 0 0 0 21 0 0 1 0 0 ...
 $ Flight.cancelled             : Factor w/ 1 level "No": 1 1 1 1 1 1 1 1 1 1 ...
 $ Flight.time.in.minutes       : int  86 60 42 110 65 38 76 77 41 71 ...
 $ Flight.Distance              : int  533 373 236 666 460 236 493 496 236 500 ...
 $ Likelihood.to.recommend      : int  9 9 10 6 7 8 10 9 7 4 ...
 $ olong                        : num  -94.6 -84.3 -115.2 -75.3 -97 ...
 $ olat                         : num  39 33.8 36.1 40 32.8 ...
 $ dlong                        : num  -105 -84.5 -118.1 -84.3 -94.6 ...
 $ dlat                         : num  39.7 39.2 34 33.8 39 ...
 $ freeText                     : chr  "N/A" "N/A" "N/A" "N/A" ...
 $ Flight_Month                 : Factor w/ 3 levels "1","2","3": 1 3 1 1 1 3 3 3 2 1 ...
>
```

Several columns have been converted into factors for better analysis during visualizations and application in Linear Regression, Support Vector Machines and Apriori.

Step 2 – Create new derived variables.

In the following code, Age and Likelihood to recommend columns have been divided according to their '*Age'* and '*Likelihood.to.recommend'* respectively. A new column age group and npsgroup has been created. Using the 'case_when' an environment of conditional statements is created.

Code Snippet

```
#Create age group and nps group

results <- results %>% mutate(agegroup = case_when(Age >= 60  & Age <= 85 ~ 'Senior Citizens',
                                                   Age >= 31  & Age <= 59 ~ 'Middle Aged',
                                                   Age >= 15  & Age <= 30 ~ 'Young'))
results <- results %>% mutate(npsgroup = case_when(Likelihood.to.recommend >= 9 ~ 'Promoter',
                                                   Likelihood.to.recommend >= 7  & Likelihood.to.recommend <= 8 ~ 'Neutral',
                                                   Likelihood.to.recommend >= 0  & Likelihood.to.recommend <= 6 ~ 'Detractor'))
```

Output

agegroup	npsgroup
Middle Aged	Promoter
Middle Aged	Promoter
Middle Aged	Promoter
Middle Aged	Detractor
Young	Neutral
Senior Citizens	Neutral
Middle Aged	Promoter
Senior Citizens	Promoter
Young	Neutral

Passengers have been divided according to age – ***Young, Middle aged, Senior Citizens*** and NPS Scores – ***Detractors, Neutrals, Promoters***.

Data Visualizations

In the following steps, we will be executing several visualizations to better understand the data using variables such as Loyalty, Age, Gender, Class, Type of Travel etc. Also, garnering insights based on a customer's first flight ! The following Charts and graphs will be used to better understand Customer Behavior and gather insights hidden in the data.

1) Histograms
2) Boxplots
3) Faceting
4) Bar Charts and many more!!

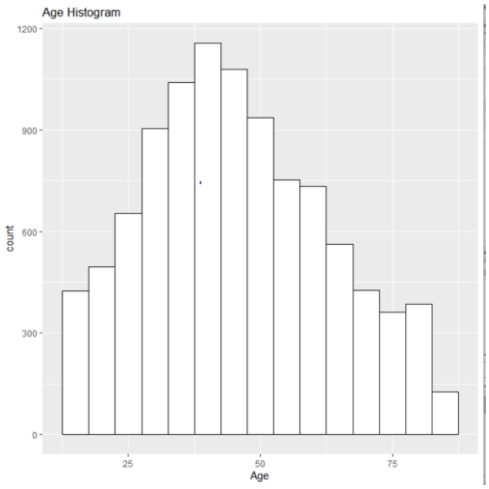
Further, using libraries such as ggplots, ggmaps for visualization and tidyverse and dplyr for data wrangling.

Step 1 (Histograms For Numeric and Factor Variables)

Code Snippet- Histogram for Age

```
hist_age <- ggplot(results,aes(x=Age)) +
  geom_histogram(binwidth = 5,color="black", fill = "white") +
  ggtitle("Age Histogram")
hist_age
```

Output

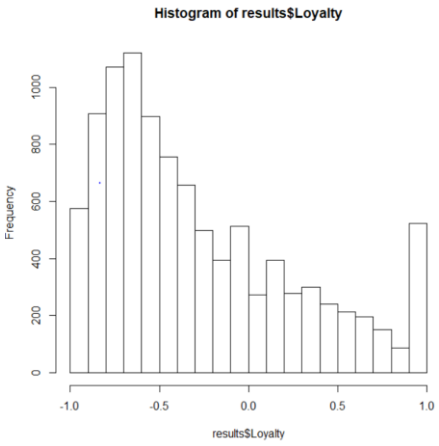


The distribution is normal and most passengers travelling are middle aged. This can be an indicator that most passengers travelling are employees in some company or they travel for business reasons(second step will be proved).

Code Snippet 2 – Histogram for Loyalty

```
hist(results$Loyalty)
g_da <- filter(results,Loyalty == 1)
table(g_da$npsgroup)
```

Output



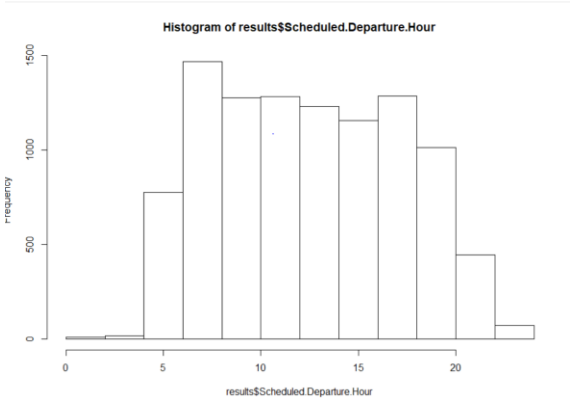

```
Detractor   Neutral  Promoter
      124       165       205
```

Here, we can see that around 600 passengers are customers with highest loyalty, suggesting almost 100% of flights were taken in southeast airlines. Hence, marketing team can leverage plans to convert detractors and Neutrals into Promoters by leveraging parameters that take advantage of customer loyalty.

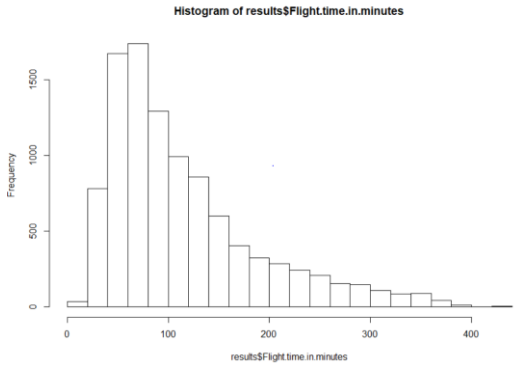
Code Snippet 3 – More Histograms!

```
hist(results$Scheduled.Departure.Hour)
hist(results$Flight.time.in.minutes)
mean(results$Flight.time.in.minutes)
median(results$Flight.Distance)
hist(results$Flight.Distance)
```

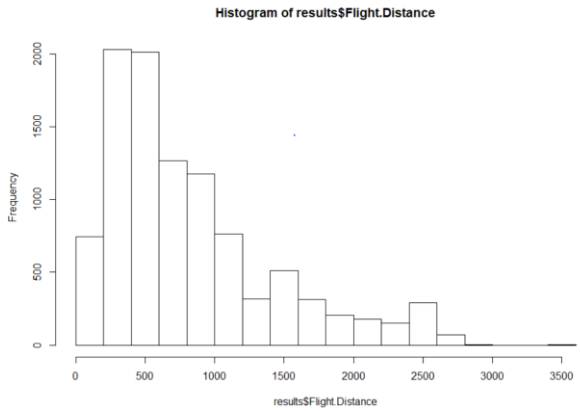
Output



Most of the flights are scheduled after 5:00 hrs to 23:59 hrs. Therefore, no flights operate from 12am – 5 am(5hours) in these three months.



The distribution is right skewed, and the mean flight time is 114.6 minutes.

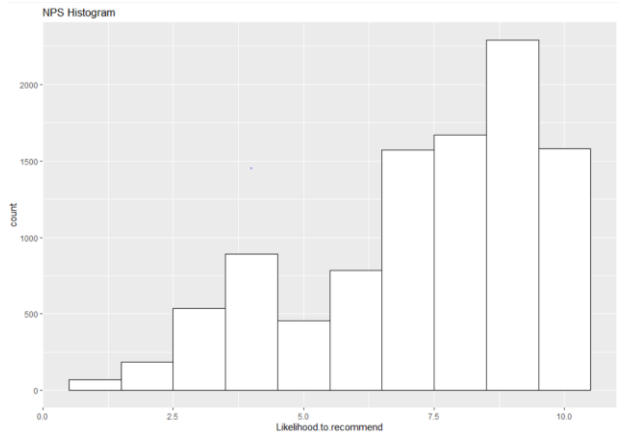


The distribution is right skewed and median flight distance is 626 km.

Code Snippet 4

```
hist_NPS <- ggplot(results,aes(x=Likelihood.to.recommend)) +
  geom_histogram(binwidth = 1,color="black", fill = "white") +
  ggtitle("NPS Histogram")
hist_NPS
```

Output



This indicates a left-skewed distribution and *most passengers can be seen to be approving of southeast airline.*

Step 2(Data Tables)

Code Snippet

```
# factor variables

table(results$Price.Sensitivity)
table(results$Gender)
table(results$Type.of.Travel)
table(results$Class)
table(results$Partner.Code)
```

Output

```
> table(results$Price.Sensitivity)

   0    1    2    3    4
 304 6727 2852  134   20
> table(results$Gender)

Female   Male
  5757   4280
> table(results$Type.of.Travel)

Business travel Mileage tickets Personal Travel
           6163              833             3041
> table(results$Class)

Business      Eco Eco Plus
     783     8199     1055
> table(results$Partner.Code)

  AA   AS   B6   DL   EV   F9   FL   HA   MQ   OO   OU   US   VX   WN
 500  302  368 1625 1088  140  201   12  505 1199  936  887  114 2160
```

Results:

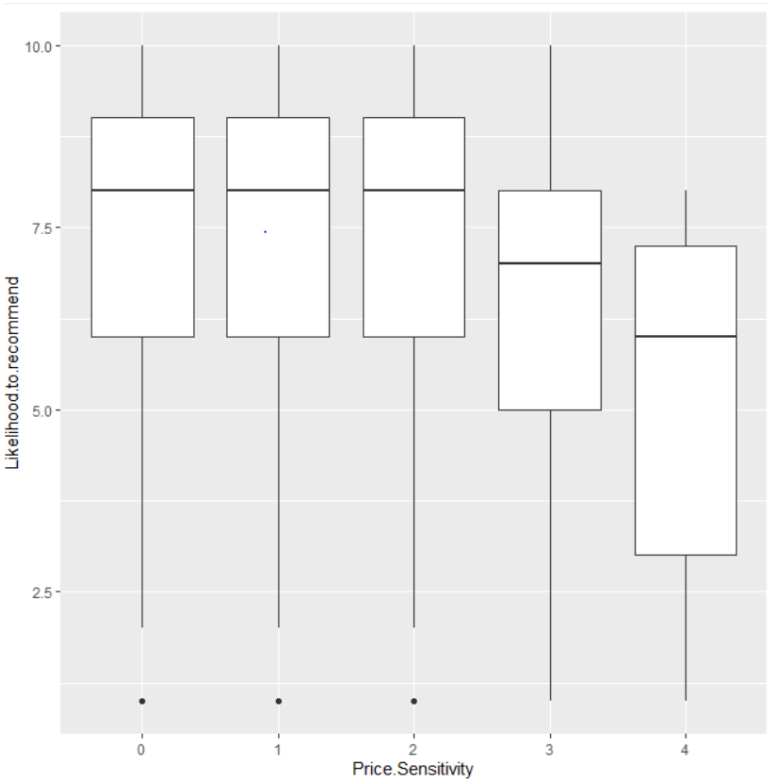
1) In the 1st line of code, the price sensitivity is denoted. **Low price sensitivity** means that a passenger is more willing to pay more for the airline's product or service. Therefore, maximum passengers score either 1 or 2 out of 5 levels. This suggests that majority of passengers are ***insensitive*** to the prices fixed by the airlines.
2) Maximum passengers in this dataset are ***Female***. But the dataset is more or less balanced based on gender.
3) Most of the passengers travel through travel quota through companies, maybe for ***Business*** reasons. Least number of passengers utilize mileage tickets to travel.
4) Maximum number of passengers fly in the ***Economy*** class.
5) Lastly, the partner code of all partner airlines in USA are listed

Step 3 (Boxplots)

Code Snippet 1 – Likelihood to recommend vs price sensitivity

```
ggplot(data=results)+
  geom_boxplot(mapping = aes(x=Price.Sensitivity,y=Likelihood.to.recommend))
table(results$Price.Sensitivity)
```

Output



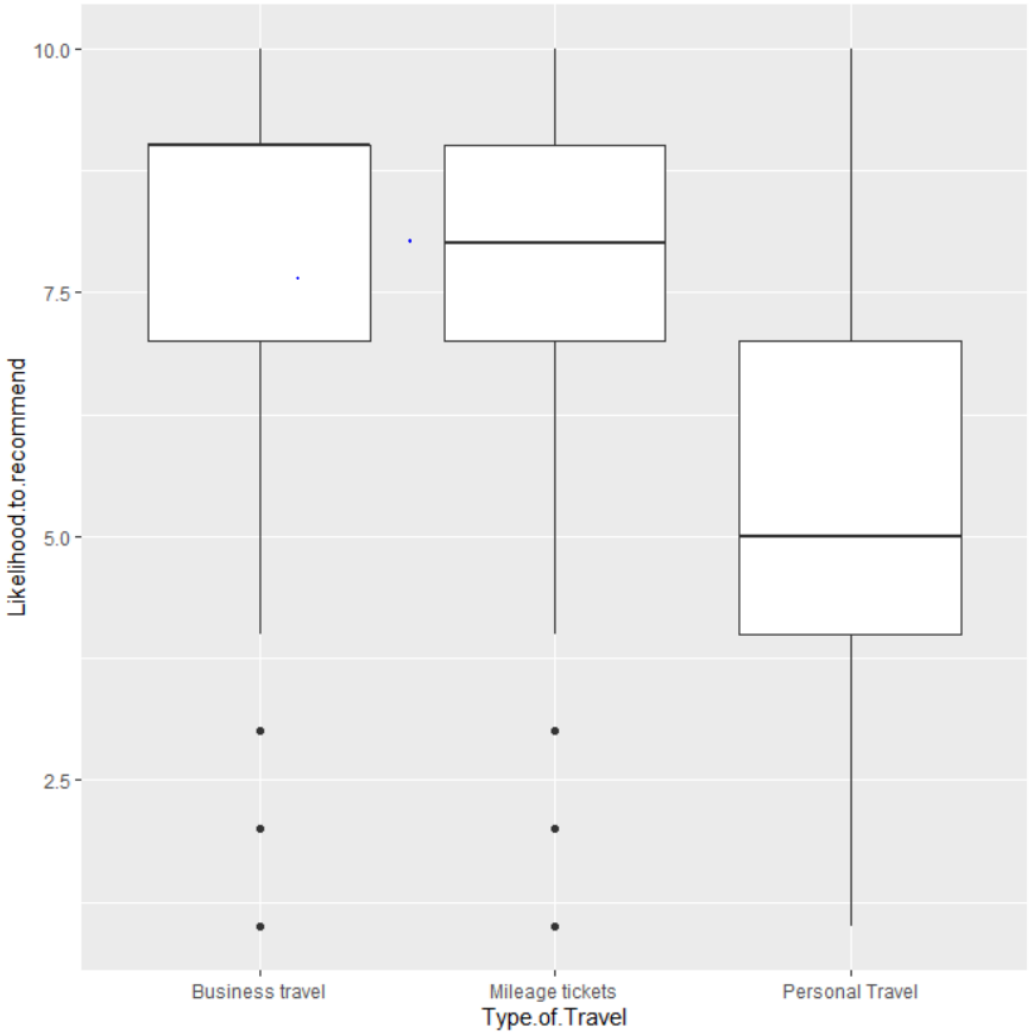
From Step 2 in Phase2- **Visualizations,** we observed that most passengers(Price sensitivity = 1 and 2) are ***not affected by price*** in purchasing airline services. In the above box plot, NPS scores are high for passengers in the category 1 and 2 of price sensitivity. Therefore, passengers scores are high for these two categories

Here, we notice as price sensitivity increases, NPS scores(Likelihood.to.recommend) decrease. Customers who are more inclined to price sensitivity (higher price sensitivity score) tend to give less NPS scores. Moreover, Customers in either 0,1,2 levels of price sensitivity have the same range of NPS scores. High NPS scores have no relation with how high or low a southeast airline plane ticket costs.

Code Snippet 2 – Type of Travel vs Likelihood to recommend

```
ggplot(data=results)+
  geom_boxplot(mapping = aes(x=Type.of.Travel,y=Likelihood.to.recommend))
table(results$Type.of.Travel)
```

Output



As seen before, the maximum number of passengers travel for ***Business travel*** and the median NPS is more than 8 (More than half (around 3000) are Promoters). Therefore, **passengers travelling for business reasons are most satisfied with southeast airlines.**

We can also see that Passengers who travel for ***Personal reasons*** are the most dissatisfied with a median score of 5 and more than 75% customers rate are either neutral or detractors(around 2500 passengers).

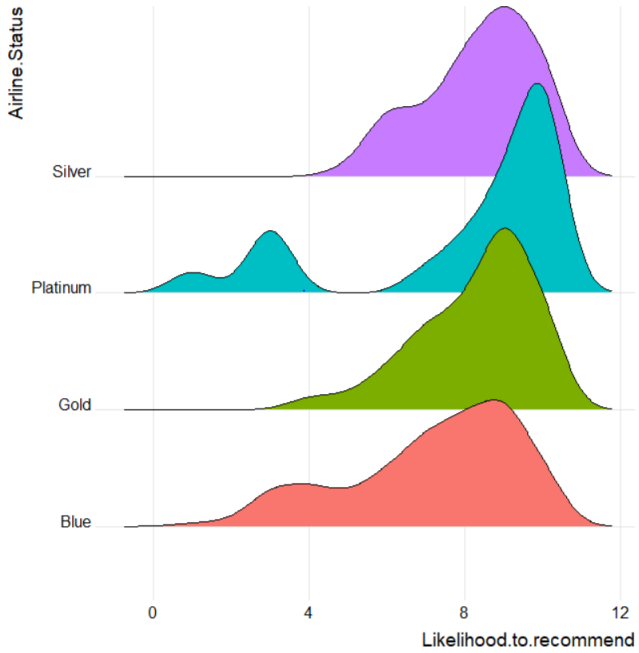
Step 4 (Ridge Line plots)

This code has been generated after filtering out passengers whose *Loyalty* = 1 (100% loyal as compared to other airlines). A ridge plot has been created to understand the variation of NPS scores according to Airline Status

Code Snippet

```
## 1.) Ridge Line Plot
library(ggridges)

plotdata <- filter(results, Loyalty==1)
table(plotdata$npsgroup)

ggplot(plotdata,
       aes(x = Likelihood.to.recommend,
           y = Airline.Status,
           fill = Airline.Status)) +
  geom_density_ridges() +
  theme_ridges() +
  labs("Highway mileage by auto class") +
  theme(legend.position = "none")
```

Output



In this plot, the customers who are most loyal to the airlines are targeted (Loyalty == 1). We notice that the most unsatisfied group of customers who are loyal to the airline are in the Platinum group, where a noticeable set of passengers rate NPS<4. This can denote a fallacy in the service provided amongst the Platinum customers who are most loyal.

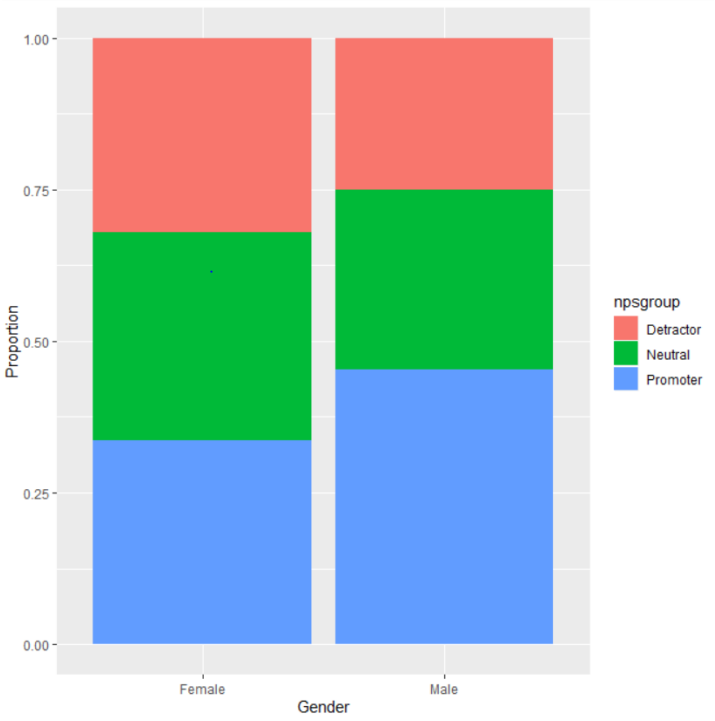
Step 5 (Stacked Bar Chart)

When plotting the relationship between two categorical variables, a stacked bar chart can be used. In this case these variables are *Gender* and *NPSgroup*.

Code Snippet

```
ggplot(results,
       aes(x = Gender,
           fill = npsgroup)) +
  geom_bar(position = "fill") +
  labs(y = "Proportion")
```

Output



The number of passengers in the dataset has been relatively calculated (according to proportion) in a stacked bar chart. Following observations are made:

1) Proportion of female promoters are less as compared to male promoters
2) Proportion of female detractors are more as compared to male detractors

Therefore, this dataset suggests ***females are more unsatisfied as compared to males.***

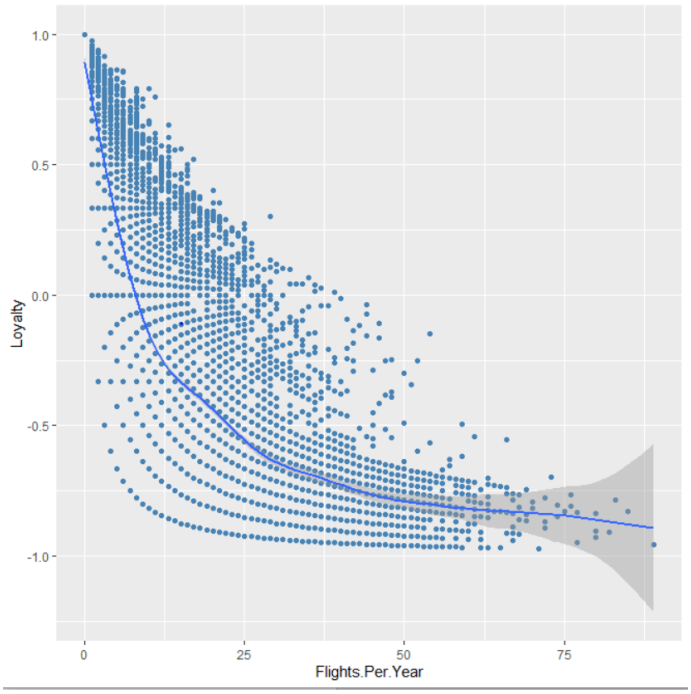
Step 6 (Scatter Plot with Regression)

When plotting the relationship between two quantitative variables, a scatter plot chart can be used. In this case these variables are Loyalty and Flights.Per.Year.

Code Snippet

```
ggplot(results,
       aes(x = Flights.Per.Year,
           y = Loyalty))·+
  geom_point(color = "steelblue") +
  geom_smooth(mehod ="lm")
```

Output



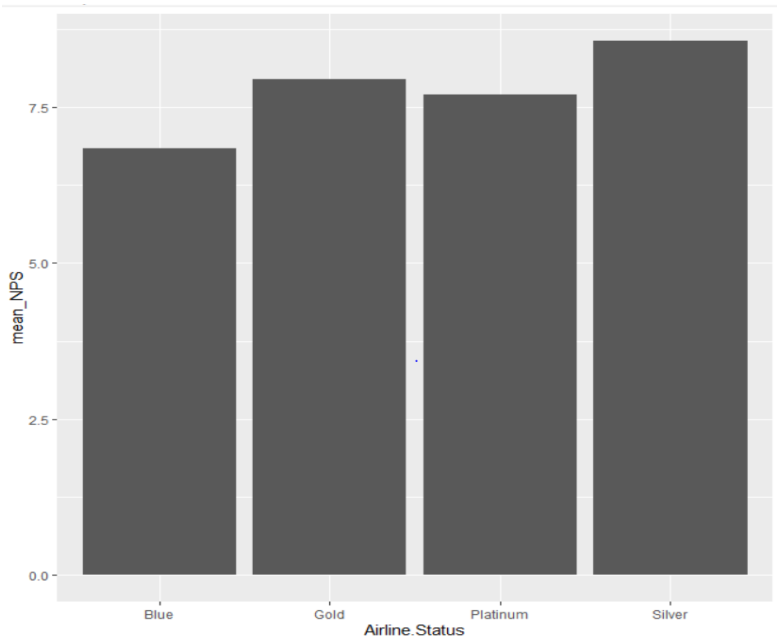
Here, we notice a negative exponential curve. This suggests, as the number of flights per year for a passenger increases, the loyalty of a customer decreases. Therefore, ***frequent flyers*** are expected to be ***less loyal*** to the airline. Hence, less frequent flyers (according to year) can be targeted by the airline. They should be provided with lucrative deals and offers to be more loyal customers.

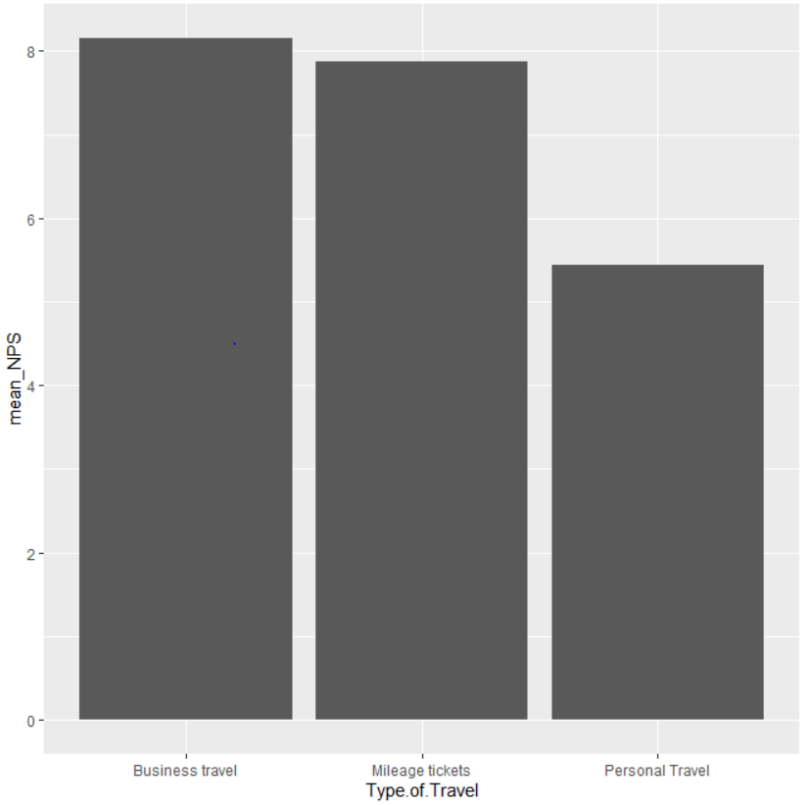
Step 7 (Bar Charts and Data Tables)

Code Snippet

```
#4.)calculate mean nps for each airline status

Status_viz <- results %>%
  group_by(Airline.Status) %>%
  summarize(mean_NPS = mean(Likelihood.to.recommend))

ggplot(Status_viz,
       aes(x = Airline.Status,
           y = mean_NPS)) +
  geom_bar(stat = "identity")
# The mean NPS of passengers with Blue airline status provide less scores

#5.)calculate mean nps for each type of travel
TOT_viz <-  results %>%
  group_by(Type.of.Travel) %>%
  summarize(mean_NPS = mean(Likelihood.to.recommend))

ggplot(TOT_viz,
       aes(x = Type.of.Travel,
           y = mean_NPS)) +
  geom_bar(stat = "identity")
# The mean NPS of paseengers who travel for personal reasons provide less scores
```

Output





In the above two figures, we observe the minimum mean NPS scores in the overall dataset is lowest for Airline Status = **Blue** and Type of Travel = **Personal Reason.**

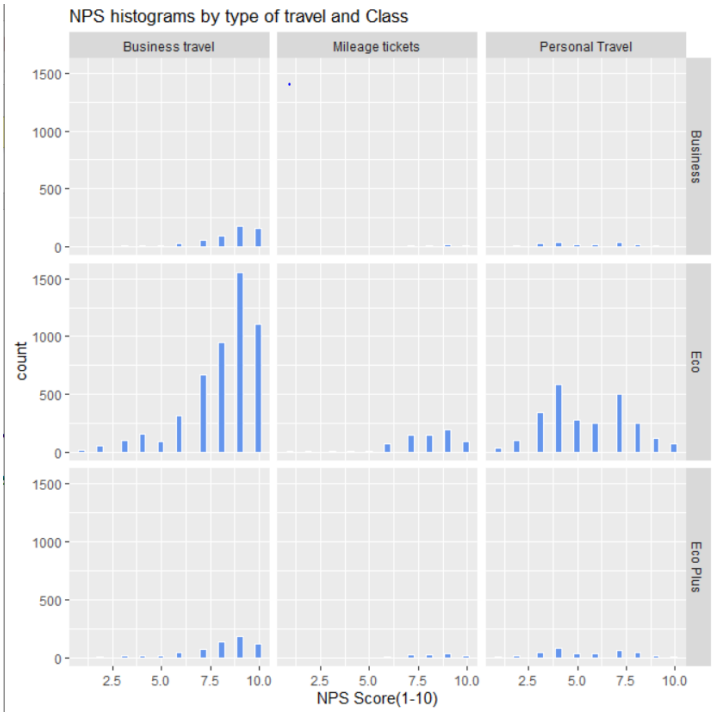
Step 8 (Faceting – Histogram)

A facet grid has been created to visualize two categorical variables and a quantitative variable in a histogram. In this case, "*likelihood to recommend*" is the quantitative variable and "*Class* and *Type of Travel*" are qualitative.

Code Snippet

```
#5.)faceting

ggplot(results, aes(x = Likelihood.to.recommend)) +
  geom_histogram(color = "white",
                 fill = "cornflowerblue") +
  facet_grid(Class ~ Type.of.Travel) +
  labs(title = "NPS histograms by type of travel and Class",
       x = "NPS Score(1-10)")
```

Output



In the above chart, from the heights of histogram it is noticed that the greatest number of passengers travel in the economy class. Further, passengers in economy class travel mostly for business reasons. It can be noticed that customers who travel in Economy Class for business reasons have a left skewed distribution and most passengers are quite approving of the airline. Therefore, **passengers travelling in economy class for business reasons are mostly promoters.**

Lastly, the histogram for the passengers in the economy class who travel for personal reasons has a near right skewed distribution and less passengers are promoters of the airline.

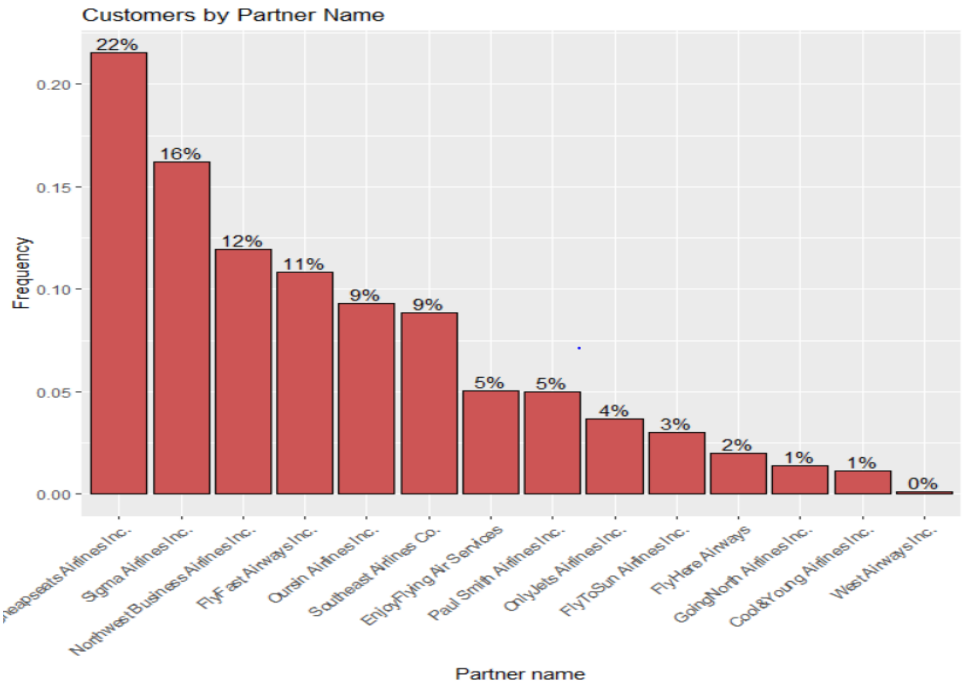
Step 9 (Bar Chart)

First, a new *partnerdata* dataframe has been created where all the percentage values of passengers according to partner airline names are stored. Using the bar chart, data is sorted in a descending order.

Code Snippet

```
#6.)  viz on partner airlines
partnerdata <- results %>%
  count(Partner.Name) %>%
  mutate(pct = n / sum(n),
         pctlabel = paste0(round(pct*100), "%"))

ggplot(partnerdata,
       aes(x = reorder(Partner.Name, -pct),
           y = pct)) +
  geom_bar(stat = "identity",
           fill = "indianred3",
           color = "black") +
  geom_text(aes(label = pctlabel),
            vjust = -0.25) +
  labs(x = "Partner name",
       y = "Frequency",
       title  = "Customers by Partner Name") +
  theme(axis.text.x = element_text(angle = 45,
                                   hjust = 1))
```

Output



In the above figure, the distribution of customers according to partner airlines is sorted in a decreasing order. ***Cheapseats Airlines*** has the highest(22%) number of customers followed by ***Sigma Airlines***(16%) .

PHASE 3

Linear Models

In order to understand the Customer Sentiments based on the NPS scores provided by the customers. Linear Regression is implemented to understand the effect of the explanatory variables on the target variable(Likelihood to recommend).

Linear Regression is performed on a new data frame (after eliminating all categorical and latitude longitude variables). Only the *numerical* and *factor* data types are taken into consideration for Linear Modelling.

Step 1 – Splitting the dataset

Code Snippet

```
reg_results <- results[-c(1,2,16,18:20,28:32)] # dropping all categorical variables

library(caTools)
set.seed(131)
split = sample.split(reg_results$Likelihood.to.recommend,SplitRatio = 0.8)
training_set = subset(reg_results,split==TRUE)
test_set = subset(reg_results,split==FALSE)
```

In the above code, we use a library ‘caTools’ used to split the data set into a training and test set . This is possible, by setting a seed value (of any number). The Split ratio is 0.8, which means the training set consists of 80% of the data and the remaining 20% is used for testing the Linear Model.

Output

test_set	2006 obs. of 24 variables
training_set	8031 obs. of 24 variables

The new dataset consists of only 24 variables, instead of the original 32 variables

Step 2 – Backward Elimination and Final Linear Model

To determine which variables are significant for prediction of the NPS score for a customer, we use a process of "*Backward Elimination*" to get the final Linear Regression Model. The process executes the following steps

Here we use a process a backward elimination, where we first, consider all the variables in the training dataset.

1.) Select Model Significance 0.05

2.) fit the model with all predictors

3.) Consider the predictor with highest p-value. If, P > Significance level, go to step

4, otherwise model is ready

4.) Remove the predictor variable with highest P-value

5.) Fit model with variables. Return to step 3 if model is not ready

The code snippets that follow, execute Linear Regression until all variables in the final model have P-value(significance) of less than "0.05". The steps have been divided into three code snippets (Code Snippet 1 and 2 are the intermediary stages and in these two stages, variables are deleted according to subsequent highest p-value. Code Snippet 3 is the final model)

Code Snippet

```
#flight.canceled removed due to error in regression model
regressor = lm(formula= Likelihood.to.recommend ~ Airline.Status + Age + Gender + Price.Sensitivity + Year.of.First.Flight +
                 Flights.Per.Year + Loyalty + Type.of.Travel + Total.Freq.Flyer.Accts + Shopping.Amount.at.Airport + Class +
                 Eating.and.Drinking.at.Airport  + Day.of.Month + Scheduled.Departure.Hour + Departure.Delay.in.Minutes+
                 Arrival.Delay.in.Minutes  + Flight.time.in.minutes + Flight.Distance,
               data = training_set)
summary(regressor)
# remove price.sensitivity

regressor = lm(formula= Likelihood.to.recommend ~ Airline.Status + Age + Gender  + Year.of.First.Flight +
                 Flights.Per.Year + Loyalty + Type.of.Travel + Total.Freq.Flyer.Accts + Shopping.Amount.at.Airport + Class +
                 Eating.and.Drinking.at.Airport  + Day.of.Month + Scheduled.Departure.Hour + Departure.Delay.in.Minutes+
                 Arrival.Delay.in.Minutes  + Flight.time.in.minutes + Flight.Distance,
               data = training_set)
summary(regressor)
# remove scheduled departure hour

regressor = lm(formula= Likelihood.to.recommend ~ Airline.Status + Age + Gender  + Year.of.First.Flight +
                 Flights.Per.Year + Loyalty + Type.of.Travel + Total.Freq.Flyer.Accts + Shopping.Amount.at.Airport + Class +
                 Eating.and.Drinking.at.Airport  + Day.of.Month  + Departure.Delay.in.Minutes+
                 Arrival.Delay.in.Minutes  + Flight.time.in.minutes + Flight.Distance,
               data = training_set)
summary(regressor)
# remove shopping amount at airport

regressor = lm(formula= Likelihood.to.recommend ~ Airline.Status + Age + Gender  + Year.of.First.Flight +
                 Flights.Per.Year + Loyalty + Type.of.Travel + Total.Freq.Flyer.Accts  + Class +
                 Eating.and.Drinking.at.Airport  + Day.of.Month  + Departure.Delay.in.Minutes+
                 Arrival.Delay.in.Minutes  + Flight.time.in.minutes + Flight.Distance,
               data = training_set)
summary(regressor)
#remove total freq flyer accounts
```

Output

```
Coefficients:
                                 Estimate  Std. Error t value Pr(>|t|)
(Intercept)                   -16.5589365  13.2610743  -1.249 0.211816
Airline.Status[T.Gold]          0.7540909   0.0736593  10.238  < 2e-16 ***
Airline.Status[T.Platinum]      0.4101794   0.1144158   3.585 0.000339 ***
Airline.Status[T.Silver]        1.4181278   0.0501603  28.272  < 2e-16 ***
Age                            -0.0056215   0.0013528  -4.155 3.28e-05 ***
Gender[T.Male]                  0.1591961   0.0410845   3.875 0.000108 ***
Year.of.First.Flight            0.0122650   0.0066070   1.856 0.063440 .
Flights.Per.Year               -0.0078125   0.0020443  -3.822 0.000134 ***
Loyalty                        -0.0633698   0.0548994  -1.154 0.248416
Type.of.Travel[T.Mileage tickets] -0.0890761   0.0724719  -1.229 0.219066
Type.of.Travel[T.Personal Travel] -2.3983219   0.0482621 -49.694  < 2e-16 ***
Total.Freq.Flyer.Accts          0.0145522   0.0199130   0.731 0.464931
Shopping.Amount.at.Airport      0.0001494   0.0003498   0.427 0.669292
Class[T.Eco]                   -0.2025657   0.0732167  -2.767 0.005676 **
Class[T.Eco Plus]              -0.1011213   0.0932368  -1.085 0.278147
Eating.and.Drinking.at.Airport  0.0028731   0.0003897   7.372 1.84e-13 ***
Scheduled.Departure.Hour       -0.0006095   0.0042098  -0.145 0.884880
Arrival.Delay.in.Minutes       -0.0056877   0.0004959 -11.468  < 2e-16 ***
Flight.time.in.minutes         -0.0025333   0.0012517  -2.024 0.043019 *
Flight.Distance                 0.0003706   0.0001520   2.439 0.014765 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.756 on 8011 degrees of freedom
Multiple R-squared:  0.3883,    Adjusted R-squared:  0.3868
F-statistic: 267.6 on 19 and 8011 DF,  p-value: < 2.2e-16
```

The above output is obtained after the completion of the final line in the above code snippet. We can see, R's Linear Model function has converted several variables into dummy variables. Dummy variables change the intercept, but the slope remains the same.

Code Snippet 2

```
regressor = lm(formula= Likelihood.to.recommend ~ Airline.Status + Age + Gender  + Year.of.First.Flight +
                 Flights.Per.Year + Loyalty + Type.of.Travel   + Class +
                 Eating.and.Drinking.at.Airport  + Day.of.Month  + Departure.Delay.in.Minutes+
                 Arrival.Delay.in.Minutes  + Flight.time.in.minutes + Flight.Distance,
               data = training_set)
summary(regressor)
# remove loyalty

regressor = lm(formula= Likelihood.to.recommend ~ Airline.Status + Age + Gender  + Year.of.First.Flight +
                 Flights.Per.Year  + Type.of.Travel   + Class +
                 Eating.and.Drinking.at.Airport  + Day.of.Month  + Departure.Delay.in.Minutes+
                 Arrival.Delay.in.Minutes  + Flight.time.in.minutes + Flight.Distance,
               data = training_set)
summary(regressor)
# remove day of month

regressor = lm(formula= Likelihood.to.recommend ~ Airline.Status + Age + Gender  + Year.of.First.Flight +
                 Flights.Per.Year  + Type.of.Travel   + Class +
                 Eating.and.Drinking.at.Airport    + Departure.Delay.in.Minutes+
                 Arrival.Delay.in.Minutes  + Flight.time.in.minutes + Flight.Distance,
               data = training_set)
summary(regressor)
#remove departure delay in minutes

regressor = lm(formula= Likelihood.to.recommend ~ Airline.Status + Age + Gender  + Year.of.First.Flight +
                 Flights.Per.Year  + Type.of.Travel   + Class +
                 Eating.and.Drinking.at.Airport    +
                 Arrival.Delay.in.Minutes  + Flight.time.in.minutes + Flight.Distance,
               data = training_set)
summary(regressor)
# remove year of first flight
```

Output

```
Coefficients:
                                  Estimate  Std. Error t value Pr(>|t|)
(Intercept)                     -15.9300652  13.2489628  -1.202 0.229258
Airline.Status[T.Gold]            0.7488063   0.0735191  10.185  < 2e-16 ***
Airline.Status[T.Platinum]        0.4050881   0.1142655   3.545 0.000395 ***
Airline.Status[T.Silver]          1.4149749   0.0500608  28.265  < 2e-16 ***
Age                              -0.0058304   0.0012307  -4.737 2.20e-06 ***
Gender[T.Male]                    0.1593289   0.0408788   3.898 9.79e-05 ***
Year.of.First.Flight              0.0119539   0.0066013   1.811 0.070204 .
Flights.Per.Year                 -0.0063364   0.0014955  -4.237 2.29e-05 ***
Type.of.Travel[T.Mileage tickets] -0.0884821   0.0724518  -1.221 0.222025
Type.of.Travel[T.Personal Travel] -2.3981231   0.0481392 -49.816  < 2e-16 ***
Class[T.Eco]                     -0.2010231   0.0731756  -2.747 0.006025 **
Class[T.Eco Plus]                -0.0995027   0.0931889  -1.068 0.285664
Eating.and.Drinking.at.Airport    0.0029100   0.0003886   7.489 7.68e-14 ***
Arrival.Delay.in.Minutes         -0.0057055   0.0004937 -11.557  < 2e-16 ***
Flight.time.in.minutes           -0.0025232   0.0012498  -2.019 0.043540 *
Flight.Distance                   0.0003696   0.0001518   2.435 0.014931 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.756 on 8015 degrees of freedom
Multiple R-squared:  0.3881,    Adjusted R-squared:  0.387
F-statistic: 338.9 on 15 and 8015 DF,  p-value: < 2.2e-16
```

In the above output, we can see most variables have reached a significance value of less than 0.05. To reach the Final Model, 8 variables were deleted according to decreasing p-values. In the next code snippet, the final model is executed.

Step 3 - Final Model

Code Snippet

```
regressor = lm(formula= Likelihood.to.recommend ~ Airline.Status + Age + Gender   +
                 Flights.Per.Year  + Type.of.Travel   + Class + .
                 Eating.and.Drinking.at.Airport    +
                 Arrival.Delay.in.Minutes  + Flight.time.in.minutes + Flight.Distance,
               data = training_set)
summary(regressor)
# The final model
```

Output

```
Coefficients:
                                 Estimate Std. Error t value Pr(>|t|)
(Intercept)                     8.0609211  0.1000078  80.603  < 2e-16 ***
Airline.Status[T.Gold]          0.7506519  0.0735225  10.210  < 2e-16 ***
Airline.Status[T.Platinum]      0.4069265  0.1142773   3.561 0.000372 ***
Airline.Status[T.Silver]        1.4166646  0.0500592  28.300  < 2e-16 ***
Age                            -0.0057648  0.0012304  -4.685 2.84e-06 ***
Gender[T.Male]                  0.1594516  0.0408845   3.900 9.70e-05 ***
Flights.Per.Year               -0.0063341  0.0014958  -4.235 2.31e-05 ***
Type.of.Travel[T.Mileage tickets] -0.0854460  0.0724427  -1.179 0.238235
Type.of.Travel[T.Personal Travel] -2.3971212  0.0481429 -49.792  < 2e-16 ***
Class[T.Eco]                   -0.2026522  0.0731804  -2.769 0.005632 **
Class[T.Eco Plus]              -0.1000533  0.0932017  -1.074 0.283073
Eating.and.Drinking.at.Airport  0.0028989  0.0003886   7.460 9.53e-14 ***
Arrival.Delay.in.Minutes       -0.0057257  0.0004936 -11.600  < 2e-16 ***
Flight.time.in.minutes         -0.0024717  0.0012497  -1.978 0.047980 *
Flight.Distance                 0.0003641  0.0001518   2.399 0.016460 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.756 on 8016 degrees of freedom
Multiple R-squared:  0.3879,    Adjusted R-squared:  0.3868
F-statistic: 362.8 on 14 and 8016 DF,  p-value: < 2.2e-16
```

In the above code, we can see that almost all values are below significance level (0.05), except two explanatory variables(Type of Travel- Mileage and Class – Eco Plus). But, since these insignificant variables are ‘encoded dummy variables’ by the linear model and the other dummy variables of the

same predictor variable are of high significance(example – Type of travel[Personal] has a p-value of <2e-16), we can ignore this discrepancy.

The p-value of the model is <2.2e-16, suggesting the model as statistically significant. The adjusted R-square value is 0.388 or 38.68%. Therefore, the predictor variables explain 38.68% variability in Likelihood.to.recommend or NPS scores. From the final model, we can interpret

1) As airline status increases, NPS scores increase.

2) As age decreases, NPS scores increase.

3) As flights per year increase, NPS scores decrease(Also proved in step 6 of visualization).

4) As class decreases, NPS scores decreases.

5) As arrival delay in minutes decrease, NPS scores increase

Further, the level of increase or decrease is determined by the coefficients of the predictor variables in the Final model. For example :- For every unit increase in flights per year of the customer, the NPS scores decrease by 0.006 units.

According to standards, for an analysis of human behavior a R-square of *0.20 or 0.30* is considered exceptional. Since, the adjusted R-square is around ***0.3868***, the industry standards are met by the Final Linear Model.

Step 4 – Prediction and Accuracy

Since, we have performed training of the model on training data. Now, we predict the NPS values in the test set.

Code Snippet

```
# predicting NPS
y_pred <- predict(regressor,newdata = test_set)
y_pred <- as.data.frame(y_pred)

actuals_preds <- data.frame(cbind(actuals=test_set$Likelihood.to.recommend, y_pred=y_pred))  # make actuals_predicteds dataframe.
correlation_accuracy <- cor(actuals_preds)  # 64.03%
correlation_accuracy
## We have built a predictive model to predict the NPS of customer from the test set with 64 % accuracy
```

Output

```
> correlation_accuracy
           actuals    y_pred
actuals 1.0000000 0.6403876
y_pred  0.6403876 1.0000000
```

The predicted values are stored in a data frame '*y_pred*'. The confusion matrix calculates the accuracy score by comparing the values of NPS scores in the test set to the values predicted by the linear model. Results indicate an accuracy of ***64.03%*** by comparing the NPS values of test set and the predicted values by the model.

Unsupervised Machine Learning Model

(Association Rules Mining)

The association rules algorithm (often called “Market Basket Analysis”) will be used on a fixed number of variables from the main dataset’ ‘results’. Using the ‘*arules’* package in RStudio, data will be prepared for implementing the model and various patterns that influences a passenger on being a Promoter or Detractor will be highlighted.

These patterns can be utilized to discover customer sentiments and can be further used by marketing teams to influence pricing, coupon offers or advertising offerings.

1) Data Preparation

Step 1 – Selecting Data

Code Snippet

```
#create new variable and filter out explanatory variables required for modelling
ap_results <- results
str(ap_results)
apriori_results <- ap_results[,-c(1,2,4,8,9,12,13,15,16,17,18,19,20,21,22,23,25;26,27,28,29,30,31,32,33)]
str(apriori_results)
```

Output

```
> str(ap_results)
'data.frame':   10037 obs. of  35 variables:
 $ Destination.City              : chr  "Denver" "Cincinnati" "Los Angeles" "Atlanta" ...
 $ Origin.City                   : chr  "Kansas City" "Atlanta" "Las Vegas" "Philadelphia" ...
 $ Airline.Status                : Factor w/ 4 levels "Blue","Gold",..: 4 2 4 2 4 1 2 1 4 1 ...
 $ Age                           : int  51 37 59 34 30 71 48 60 28 56 ...
 $ Gender                        : Factor w/ 2 levels "Female","Male": 2 1 2 2 1 1 1 1 1 1 ...
 $ Price.Sensitivity             : Factor w/ 5 levels "0","1","2","3",..: 2 2 2 2 3 2 2 2 3 3 ...
 $ Year.of.First.Flight          : int  2003 2008 2003 2007 2007 2009 2003 2009 2011 2004 ...
 $ Flights.Per.Year              : int  0 0 32 73 5 51 17 9 18 11 ...
 $ Loyalty                       : num  1 1 -0.882 -0.698 0.444 ...
 $ Type.of.Travel                : Factor w/ 3 levels "Business travel",..: 2 1 1 3 3 3 1 1 3 3 ...
 $ Total.Freq.Flyer.Accts        : int  3 1 0 1 1 0 0 0 1 0 ...
 $ Shopping.Amount.at.Airport    : int  55 15 5 25 131 0 30 90 125 0 ...
 $ Eating.and.Drinking.at.Airport: int  30 50 80 15 180 90 70 100 30 80 ...
 $ Class                         : Factor w/ 3 levels "Business","Eco",..: 1 2 2 2 2 2 2 2 2 2 ...
 $ Day.of.Month                  : int  5 18 25 27 27 12 23 13 5 4 ...
 $ Flight.date                   : Date, format: "2014-01-05" "2014-03-18" ...
 $ Partner.Code                  : chr  "WN" "DL" "OO" "US" ...
 $ Partner.Name                  : Factor w/ 14 levels "Cheapseats Airlines Inc.",..: 1 12 8 13 1 1 8 1 10 4 ...
 $ Origin.State                  : chr  "Missouri" "Georgia" "Nevada" "Pennsylvania" ...
 $ Destination.State             : chr  "Colorado" "Kentucky" "California" "Georgia" ...
 $ Scheduled.Departure.Hour      : int  8 7 10 14 18 11 12 9 9 7 ...
 $ Departure.Delay.in.Minutes    : int  38 0 0 0 24 0 0 0 0 0 ...
 $ Arrival.Delay.in.Minutes      : int  66 0 0 0 21 0 0 1 0 0 ...
 $ Flight.cancelled              : Factor w/ 1 level "No": 1 1 1 1 1 1 1 1 1 1 ...
 $ Flight.time.in.minutes        : int  86 60 42 110 65 38 76 77 41 71 ...
 $ Flight.Distance               : int  533 373 236 666 460 236 493 496 236 500 ...
 $ Likelihood.to.recommend       : int  9 9 10 6 7 8 10 9 7 4 ...
 $ olong                         : num  -94.6 -84.3 -115.2 -75.3 -97 ...
 $ olat                          : num  39 33.8 36.1 40 32.8 ...
 $ dlong                         : num  -105 -84.5 -118.1 -84.3 -94.6 ...
 $ dlat                          : num  39.7 39.2 34 33.8 39 ...
 $ freeText                      : chr  "N/A" "N/A" "N/A" "N/A" ...
 $ Flight_Month                  : Factor w/ 3 levels "1","2","3": 1 3 1 1 1 3 3 3 2 1 ...
 $ agegroup                      : chr  "Middle Aged" "Middle Aged" "Middle Aged" "Middle Aged" ...
 $ npsgroup                      : chr  "Promoter" "Promoter" "Promoter" "Detractor" ...
```

```
> str(apriori_results)
'data.frame':   10037 obs. of  9 variables:
 $ Airline.Status        : Factor w/ 4 levels "Blue","Gold",..: 4 2 4 2 4 1 2 1 4 1 ...
 $ Gender                : Factor w/ 2 levels "Female","Male": 2 1 2 2 1 1 1 1 1 1 ...
 $ Price.Sensitivity     : Factor w/ 5 levels "0","1","2","3",..: 2 2 2 2 3 2 2 2 3 3 ...
 $ Year.of.First.Flight  : int  2003 2008 2003 2007 2007 2009 2003 2009 2011 2004 ...
 $ Type.of.Travel        : Factor w/ 3 levels "Business travel",..: 2 1 1 3 3 3 1 1 3 3 ...
 $ Total.Freq.Flyer.Accts: int  3 1 0 1 1 0 0 0 1 0 ...
 $ Class                 : Factor w/ 3 levels "Business","Eco",..: 1 2 2 2 2 2 2 2 2 2 ...
 $ agegroup              : chr  "Middle Aged" "Middle Aged" "Middle Aged" "Middle Aged" ...
 $ npsgroup              : chr  "Promoter" "Promoter" "Promoter" "Detractor" ...
```

It is observed that 9 explanatory variables are extracted from the model, originally containing 32 variables. In this filtered dataset, we notice that some variables are not in factor format. Therefore, in the next step these variables are converted to factors

Step 2 – Convert data type to factors

Code Snippet

```
#Convert data type of desired variables to factor for sparse matrix, to calculate aprirori insights
apriori_results <- apriori_results %>%
  mutate(agegroup = as.factor(agegroup),npsgroup = as.factor(npsgroup),Year.of.First.Flight = as.factor(Year.of.First.Flight)
         ,Total.Freq.Flyer.Accts = as.factor(Total.Freq.Flyer.Accts))

str(apriori_results)
```

Output

```
> str(apriori_results)
'data.frame':   10037 obs. of  9 variables:
 $ Airline.Status        : Factor w/ 4 levels "Blue","Gold",..: 4 2 4 2 4 1 2 1 4 1 ...
 $ Gender                : Factor w/ 2 levels "Female","Male": 2 1 2 2 1 1 1 1 1 1 ...
 $ Price.Sensitivity     : Factor w/ 5 levels "0","1","2","3",..: 2 2 2 2 3 2 2 2 3 3 ...
 $ Year.of.First.Flight  : Factor w/ 10 levels "2003","2004",..: 1 6 1 5 5 7 1 7 9 2 ...
 $ Type.of.Travel        : Factor w/ 3 levels "Business travel",..: 2 1 1 3 3 3 1 1 3 3 ...
 $ Total.Freq.Flyer.Accts: Factor w/ 10 levels "0","1","2","3",..: 4 2 1 2 2 1 1 1 2 1 ...
 $ Class                 : Factor w/ 3 levels "Business","Eco",..: 1 2 2 2 2 2 2 2 2 2 ...
 $ agegroup              : Factor w/ 3 levels "Middle Aged",..: 1 1 1 1 3 2 1 2 3 1 ...
 $ npsgroup              : Factor w/ 3 levels "Detractor","Neutral",..: 3 3 3 1 2 2 3 3 2 1 ...
```

All variables in the dataset have been converted to factors to be utilized in the apriori algorithm. The factors consist of 2 or more levels.

Step 3 - Convert Data frame into Transactions

Code Snippet

```
# Convert dataframe into transactions
apriori_resultsx <- as(apriori_results,"transactions")
```

All values in the dataset “apriori_results” are stored as transaction values in a new dataset “apriori_resultsx”.

2) Model Development and Prediction – Promoters

Step 1 – Model Development

The model has a minimum support of 0.005 and confidence of 0.5. All patterns uncovered will be to determine what variables are necessary for a passenger to be a promoter. This will also help us understand the patterns that led to a passenger be a 'Promoter'.

Code Snippet

```
# run model for promoters
magic <- apriori(apriori_resultsx, # initiates apriori library on magic
                 parameter = list(support = 0.005,confidence = 0.5), # sets the support and confidence as parameters
                 appearance = list(default='lhs',rhs=("npsgroup=Promoter"))) # # to find connections of passengers who are Promoters
```

Output

```
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen maxlen target   ext
        0.5    0.1    1 none FALSE            TRUE       5   0.005      1     10  rules FALSE

Algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE  FALSE TRUE    2    TRUE

Absolute minimum support count: 50

set item appearances ...[1 item(s)] done [0.00s].
set transactions ...[43 item(s), 10037 transaction(s)] done [0.00s].
sorting and recoding items ... [38 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 4 5 6 7 8 done [0.02s].
writing ... [1741 rule(s)] done [0.00s].
creating S4 object  ... done [0.00s].
```

The above output tells us:

1) Subsets of Size – number of elements in a rule vary from size 1 to 8.
2) Number of rules – *1741* rules are created.

Step 2 – Model Prediction

The data is sorted into the first 10 rules having highest lift.

Code Snippet

```
inspectDT(sort(magic,by='lift')[1:10])
```

Output

Show 10 entries Search:

	LHS	RHS	support	confidence	lift	count
	All	All	All	All	All	All
[1]	{Airline.Status=Silver,Price.Sensitivity=1,Type.of.Travel=Business travel,Class=Business}	{npsgroup=Promoter}	0.009	0.837	2.168	87.000
[2]	{Airline.Status=Silver,Gender=Female,Type.of.Travel=Business travel,Class=Business}	{npsgroup=Promoter}	0.006	0.827	2.143	62.000
[3]	{Airline.Status=Silver,Price.Sensitivity=1,Type.of.Travel=Business travel,Class=Business,agegroup=Middle Aged}	{npsgroup=Promoter}	0.006	0.824	2.137	61.000
[4]	{Airline.Status=Silver,Gender=Male,Type.of.Travel=Mileage tickets}	{npsgroup=Promoter}	0.005	0.823	2.132	51.000
[5]	{Airline.Status=Silver,Price.Sensitivity=1,Type.of.Travel=Mileage tickets,Class=Eco}	{npsgroup=Promoter}	0.006	0.816	2.115	62.000
[6]	{Airline.Status=Silver,Gender=Male,Total.Freq.Flyer.Accts=3}	{npsgroup=Promoter}	0.005	0.813	2.106	52.000
[7]	{Airline.Status=Silver,Type.of.Travel=Mileage tickets,Class=Eco}	{npsgroup=Promoter}	0.009	0.813	2.106	91.000
[8]	{Airline.Status=Silver,Type.of.Travel=Mileage tickets}	{npsgroup=Promoter}	0.011	0.812	2.104	112.000
[9]	{Airline.Status=Silver,Price.Sensitivity=1,Class=Business,agegroup=Middle Aged}	{npsgroup=Promoter}	0.007	0.810	2.098	68.000
[10]	{Airline.Status=Silver,Gender=Male,Year.of.First.Flight=2006,Type.of.Travel=Business travel,Class=Eco}	{npsgroup=Promoter}	0.005	0.810	2.098	51.000

Showing 1 to 10 of 10 entries Previous 1 Next

Considering the 2nd rule, the above model indicates:

Support indicates the probability of LHS and RHS occurring together

If a female passenger is travelling for business purposes in business class (Airline Status is Silver), the passenger is a promoter for *82.7%* of the time (confidence = 0.827).

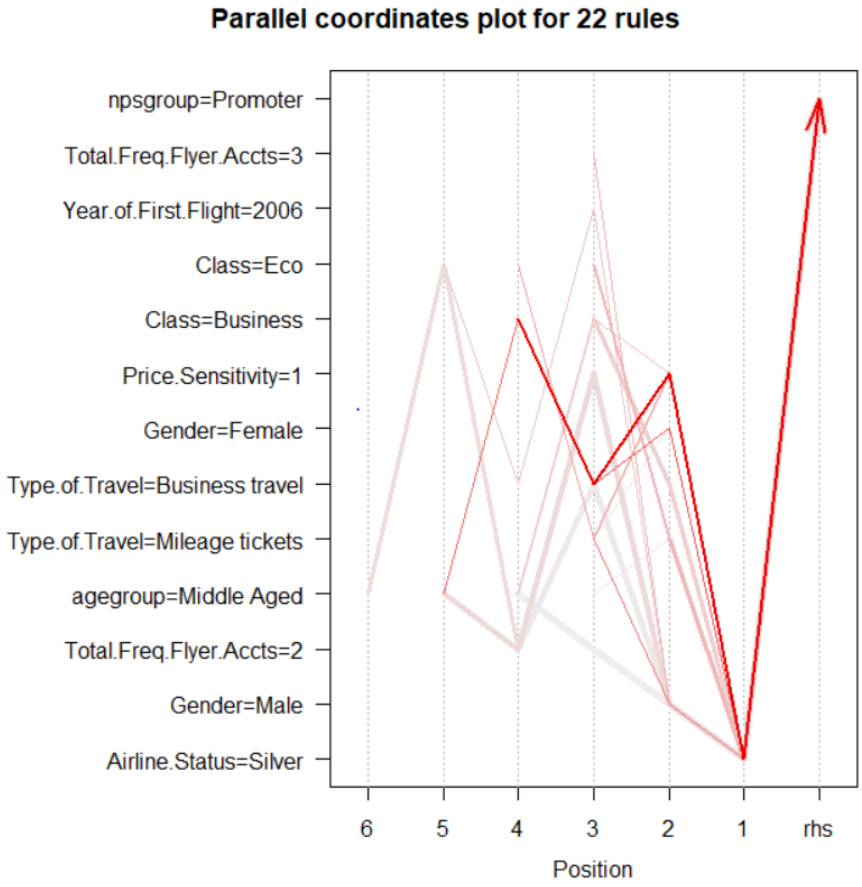
The lift indicates the items in LHS and RHS are *2.143* times more likely to occur together compared to the occurrence when they are assumed to be unrelated.

Step 3 -Model Visualization

Code Snippet

```
sub_magic = magic[quality(magic)$confidence>0.8]
plot(sub_magic,method = "paracoord")
```

Output



From the above graph, we attain a visualization of 22 rules with a confidence level of above 0.8. From this viz, we can observe that the most likely rule in LHS, which will results in a passenger being a Promoter is if, the passenger travels in ***"business class for business travel, Price sensitivity = 1 and Airline status is sliver".***

3) Model Development and Prediction – Detractors

The model has a minimum support of 0.005 and confidence of 0.5. All patterns uncovered will be to determine what variables are necessary for a passenger to be a Detractor. This will also help us understand the patterns, creating rules that led a passenger to be 'Detractors'.

Step 1 – Model Development

Code Snippet

```
magic2 <- apriori(apriori_resultsx, # initiates apriori library on magic
                  parameter = list(support = 0.005,confidence = 0.5), # sets the support and confidence as parameters
                  appearance = list(default='lhs',rhs=("npsgroup=Detractor"))) # # to find what kind of passengers survived
```

Output

```
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen maxlen target   ext
        0.5    0.1    1 none FALSE            TRUE       5   0.005      1     10  rules FALSE

Algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE  FALSE TRUE    2    TRUE

Absolute minimum support count: 50

set item appearances ...[1 item(s)] done [0.00s].
set transactions ...[43 item(s), 10037 transaction(s)] done [0.00s].
sorting and recoding items ... [38 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 4 5 6 7 8 done [0.03s].
writing ... [1227 rule(s)] done [0.01s].
creating S4 object  ... done [0.00s].
```

The above output tells us:

1) Subsets of Size – number of elements in a rule vary from size 1 to 8.
2) Number of rules – 1227 rules are created.

Step 2 – Model Prediction

Code Snippet

```
inspectDT(sort(magic2,by='lift')[1:10])
```

Output

Show 10 entries Search:

	LHS	RHS	support	confidence	lift	count
	All	All	All	All	All	All
[1]	{Airline.Status=Blue,Year.of.First.Flight=2007,Type.of.Travel=Personal Travel,Class=Eco,agegroup=Senior Citizens}	{npsgroup=Detractor}	0.007	0.840	2.886	68.000
[2]	{Airline.Status=Blue,Price.Sensitivity=1,Type.of.Travel=Personal Travel,Total.Freq.Flyer.Accts=1,agegroup=Young}	{npsgroup=Detractor}	0.006	0.838	2.881	57.000
[3]	{Airline.Status=Blue,Price.Sensitivity=1,Year.of.First.Flight=2011,Type.of.Travel=Personal Travel,Class=Eco,agegroup=Senior Citizens}	{npsgroup=Detractor}	0.005	0.836	2.874	51.000
[4]	{Airline.Status=Blue,Price.Sensitivity=2,Year.of.First.Flight=2003,Type.of.Travel=Personal Travel,Total.Freq.Flyer.Accts=0,Class=Eco}	{npsgroup=Detractor}	0.006	0.836	2.873	56.000
[5]	{Airline.Status=Blue,Gender=Female,Price.Sensitivity=2,Year.of.First.Flight=2011,Type.of.Travel=Personal Travel}	{npsgroup=Detractor}	0.006	0.833	2.864	60.000
[6]	{Airline.Status=Blue,Year.of.First.Flight=2007,Type.of.Travel=Personal Travel,Total.Freq.Flyer.Accts=0,Class=Eco,agegroup=Senior Citizens}	{npsgroup=Detractor}	0.006	0.833	2.864	65.000
[7]	{Airline.Status=Blue,Year.of.First.Flight=2007,Type.of.Travel=Personal Travel,agegroup=Senior Citizens}	{npsgroup=Detractor}	0.008	0.832	2.858	79.000
[8]	{Airline.Status=Blue,Gender=Female,Year.of.First.Flight=2007,Type.of.Travel=Personal Travel,agegroup=Senior Citizens}	{npsgroup=Detractor}	0.006	0.831	2.856	59.000
[9]	{Airline.Status=Blue,Price.Sensitivity=1,Year.of.First.Flight=2007,Type.of.Travel=Personal Travel,agegroup=Senior Citizens}	{npsgroup=Detractor}	0.005	0.831	2.856	54.000
[10]	{Airline.Status=Blue,Gender=Male,Year.of.First.Flight=2006,Type.of.Travel=Personal Travel,Class=Eco}	{npsgroup=Detractor}	0.006	0.829	2.848	58.000

Showing 1 to 10 of 10 entries Previous 1 Next

Considering the 2nd rule, the above model indicates:

Support indicates the probability of LHS and RHS occurring together (0.006)

If a young(15-30 years) passenger is travelling for personal purposes having 1 frequent flier account (Airline Status is Blue), the passenger is a detractor for 83.8% of the time (confidence = 0.838).

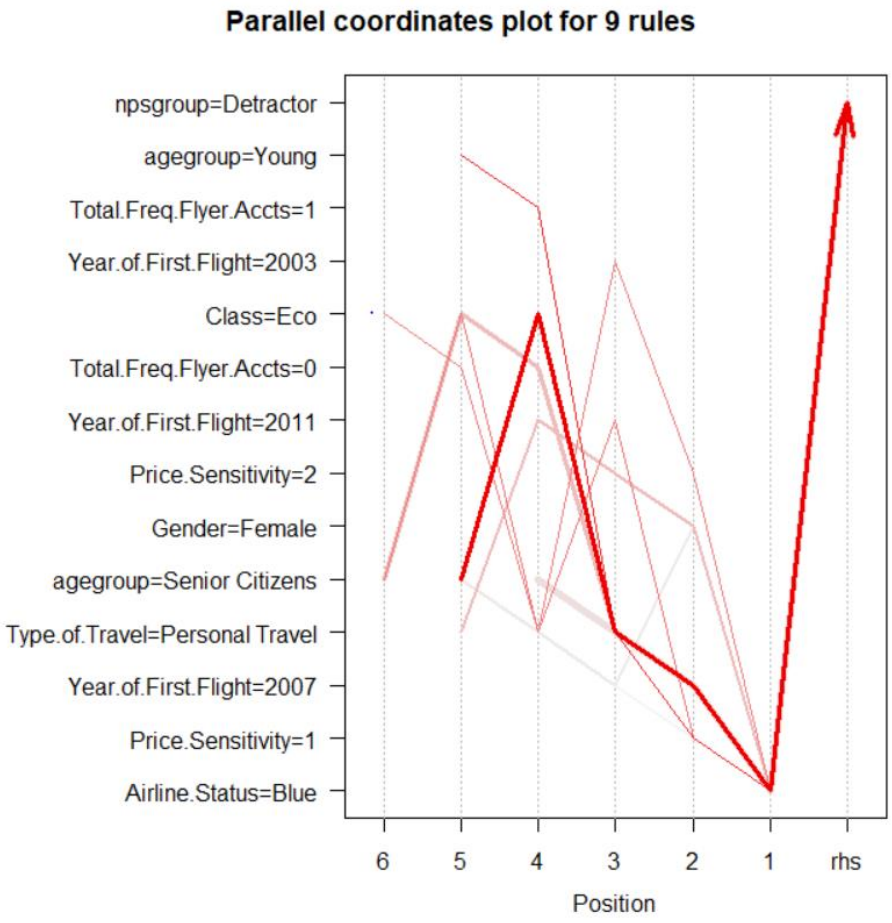
The lift indicates the items in LHS and RHS are 2.2.881 times more likely to occur together compared to the occurrence when they are assumed to be unrelated.

Step 3 – Model Visualization

Code Snippet

```
sub_magic2 = magic2[quality(magic2)$confidence>0.83]
plot(sub_magic2,method = "paracoord")
```

Output



From the above graph, we attain a visualization of 9 rules with a confidence level of above 0.83. From this viz, we can observe that the most likely rule in LHS, which will result in a passenger being a Detractor, is ***if the passenger is a senior citizen (60-80 years) flying in economy class(airline status = Blue) for personal reasons.***

Supervised Learning Model (kernel-SVM)

Supervised Learning Model can be accomplished when a training set can be prepared for the Model. It is less complex as compared to Unsupervised Learning. In the following steps, ksvm algorithm has been implemented which uses support vectors to understand patterns in the training data.

Cross Validation is performed at the end to ensure the accuracy of the model on a novel data set. A low cross-validation error indicates the SVM model can be confidently implemented on new data.

Data Preparation

Step 1

Code Snippet

```
# data preparation
sv_results <- results[,c(3,4,6,7,8,9,11:13,15,21:23,25,26,35)] # categorical variables are excluded
str(sv_results)

# Output variable is converted to factor
sv_results <- sv_results %>%
  mutate(npsgroup = as.factor(npsgroup))
```

Output

```
> str(sv_results)
'data.frame':   10037 obs. of  16 variables:
 $ Airline.Status                : Factor w/ 4 levels "Blue","Gold",..: 4 2 4 2 4 1 2 1 4 1 ...
 $ Age                           : int  51 37 59 34 30 71 48 60 28 56 ...
 $ Price.Sensitivity             : Factor w/ 5 levels "0","1","2","3",..: 2 2 2 2 3 2 2 2 3 3 ...
 $ Year.of.First.Flight          : int  2003 2008 2003 2007 2007 2009 2003 2009 2011 2004 ...
 $ Flights.Per.Year              : int  0 0 32 73 5 51 17 9 18 11 ...
 $ Loyalty                       : num  1 1 -0.882 -0.698 0.444 ...
 $ Total.Freq.Flyer.Accts        : int  3 1 0 1 1 0 0 0 1 0 ...
 $ Shopping.Amount.at.Airport    : int  55 15 5 25 131 0 30 90 125 0 ...
 $ Eating.and.Drinking.at.Airport: int  30 50 80 15 180 90 70 100 30 80 ...
 $ Day.of.Month                  : int  5 18 25 27 27 12 23 13 5 4 ...
 $ Scheduled.Departure.Hour      : int  8 7 10 14 18 11 12 9 9 7 ...
 $ Departure.Delay.in.Minutes    : int  38 0 0 0 24 0 0 0 0 0 ...
 $ Arrival.Delay.in.Minutes      : int  66 0 0 0 21 0 0 1 0 0 ...
 $ Flight.time.in.minutes        : int  86 60 42 110 65 38 76 77 41 71 ...
 $ Flight.Distance               : int  533 373 236 666 460 236 493 496 236 500 ...
 $ npsgroup                      : Factor w/ 3 levels "Detractor","Neutral",..: 3 3 3 1 2 2 3 3 2 1 ...
```

The necessary explanatory variables have been extracted from the main data set. The new dataset does not contain any categorical variables and will be used in the Kernel – Support Vector Machine Model (ksvm).

Step 2 (Splitting the data)

Code Snippet

```
## spliting into training and test set

randIndex <- sample(1:dim(sv_results)[1])
summary(randIndex)

length(randIndex)

cutpoint2_3 <- floor(2*dim(sv_results)[1]/3)
cutpoint2_3

str(sv_results)
train_data <- sv_results[randIndex[1:cutpoint2_3],]
test_data <- sv_results[randIndex[(cutpoint2_3+1):dim(sv_results)[1]],]
```

Output

```
> dim(train_data)
[1] 6691   16
> dim(test_data)
[1] 3346   16
```

Using Sampling, values are distributed in the test set and training set. The ordering of the Samples is not serial wise. This will lead to less cross validation error.

The new dataset "sv_results" has been split by a ratio of, 2/3(63.67%) of sv_results in *'training set'* and 1/3(33.33%) of sv_results in '*test set'*.

Final Model

Step 1 (training training set and predicting test set values)

Code Snippet

```
# train the model
svmOutput <- ksvm(npsgroup ~.,data=train_data,kernel="rbfdot",kpar="automatic",C=15,cross=3,prob.model=TRUE)
svmOutput

#predict values
svmPred <- predict(svmOutput,test_data)
#view(svmPred)
```

Output

```
> svmOutput
Support Vector Machine object of class "ksvm"

SV type: C-svc  (classification)
 parameter : cost C = 15

Gaussian Radial Basis kernel function.
 Hyperparameter : sigma =  0.0535847028269627

Number of Support Vectors : 5523

Objective Function Value : -33380.29 -24185.61 -37535.86
Training error : 0.278284
Cross validation error : 0.48767
Probability model included.
```

In the above code we run the ksvm model on the training data. The kernel is set to “rbfdot” – meaning radial base function, kpar is set to automatic. C is the cost of constraints set to 15 and cross validation model = 3 nearest neighbors, probabilistic model is set to True and the number of support vectors is 5523. Cross validation can deal with overfitting. The training error in the model is ***0.27*** and the cross-validation error for new test set is ***0.48767***.

Step 2 – Accuracy

Code Snippet

```
#confusion matrix
confmatrix <- table(test_data$npsgroup,svmPred)
confmatrix

#accuracy rate
accuracy_rate <- ((confmatrix[1,1] +confmatrix[2,2] + confmatrix[3,3])/(sum(confmatrix[1,]) +
                                  sum(confmatrix[2,])+sum(confmatrix[3,])))*100
accuracy_rate
```

Output

```
> confmatrix
            svmPred
             Detractor Neutral Promoter
  Detractor        511     242      220
  Neutral          306     350      462
  Promoter         136     225      894
> accuracy_rate
[1] 52.45069
```

In the first part of the code, a 3x3 confusion matrix is created to test the values of the test set with that of the values predicted by the **ksvm** model.

We can see that the true predictions for each category were higher than the false predictions.

Lastly, an accuracy rate of ***52.45%*** is attained. Explaining that 52.45% of the values predicted in the confusion matrix is true.

PHASE 4

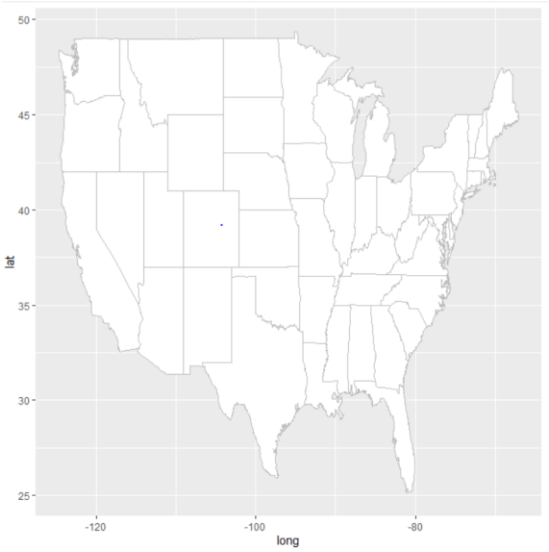
Map Low Customer Satisfaction Routes

Data and Map Preparation

Code Snippet

```
# Plot flight routes for all likelihood to recommend scores below 2
flight_results <- results[results$Likelihood.to.recommend <= 2,]


usMap <- borders("state", colour="grey", fill="white")
ggplot() + usMap
```

Output



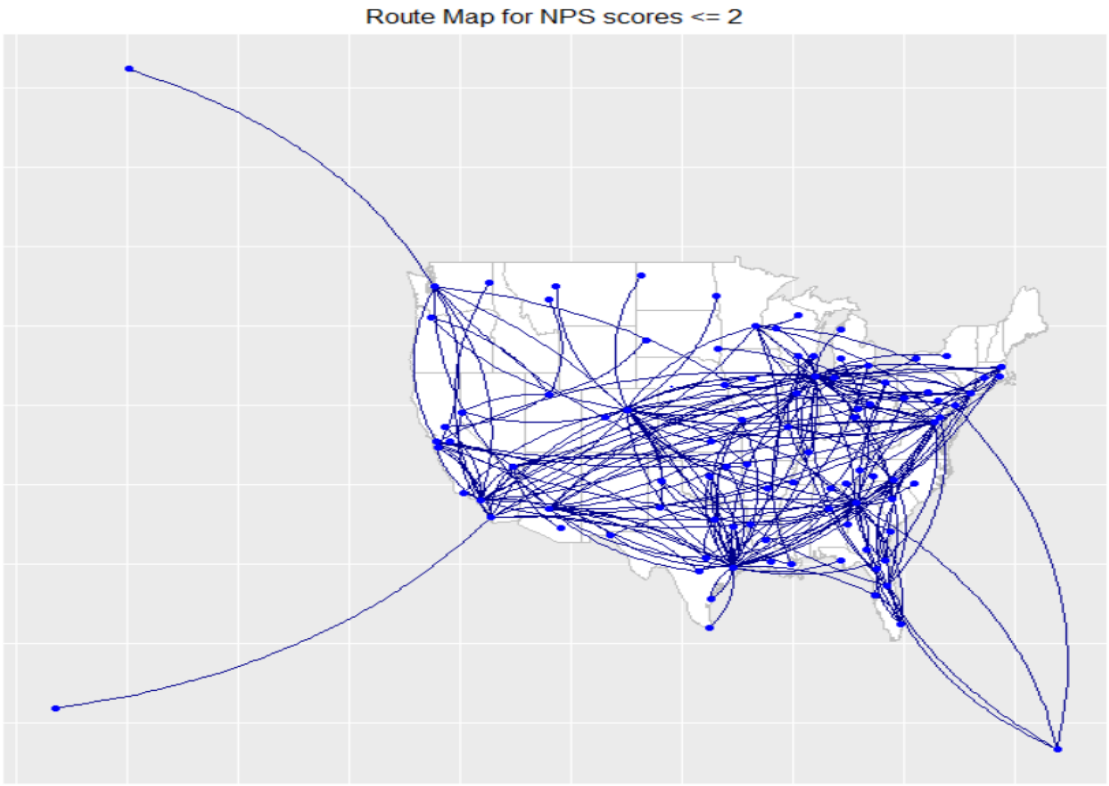
A plain map of the United States is first created to map low satisfaction routes. **Low satisfaction routes** indicate the flight routes in the entire country which have received low NPS scores. In the next step, we will be creating a map which indicates all the routes that have received the lowest NPS scores of 1 and 2.

Route Map Visualization

Code Snippet

```
allUSA <- ggplot() + usMap +
  geom_curve(data=flight_results,
             aes(x=olong, y=olat, xend=dlong, yend=dlat),
             col="#00008b",
             size=.5,
             curvature=0.2) +
  geom_point(data=flight_results,
             aes(x=olong, y=olat),
             colour="blue",
             size=1.5) +
  geom_point(data=flight_results,
             aes(x=dlong, y=dlat),
             colour="blue") +
  theme(axis.line=element_blank(),
        axis.text.x=element_blank(),
        axis.text.y=element_blank(),
        axis.title.x=element_blank(),
        axis.title.y=element_blank(),
        axis.ticks=element_blank(),
        plot.title=element_text(hjust=0.5, size=12)) +
  ggtitle("Route Map for NPS scores <= 2")
```

Output



In the above map visualization created by ggmaps, the cities have been marked by a point in the map, while the flight routes are covered by route curves. The map covers all flight routes for customers who gave an NPS of 1 or 2**. Most of the flights covered by Southeast Airlines were in the Central or Eastern Zone of the country.**

PHASE 5

Analysis of Low Satisfaction Routes

To further analyze the customers who are Detractors (NPS<=6) some visualizations have been created. First, we filter the data.

Code Snippet

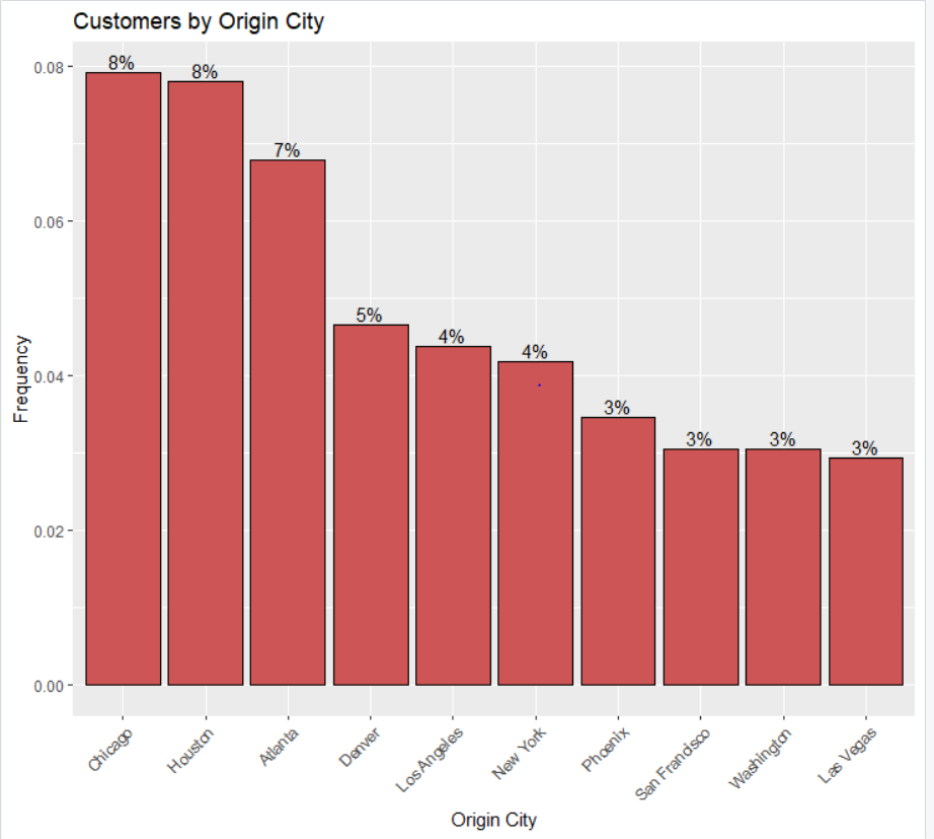
```
# filter data with passengers having NPS<6(Detractors)
low_score_flight <- filter(results,Likelihood.to.recommend <= 6)
view(low_score_flight)
summary(low_score_flight)
```

Analysis 1a)

Code Snippet

```
# origin cities having lowest nps scores
flightdata <- low_score_flight %>%
  count(Origin.City) %>%
  mutate(pct = n / sum(n),
         pctlabel = paste0(round(pct*100), "%"))

flightdata <- flightdata[with(flightdata,order(-pct)),]
flightdata <- flightdata[1:10,]

ggplot(flightdata,
       aes(x = reorder(Origin.City, -pct),
           y = pct)) +
  geom_bar(stat = "identity",
           fill = "indianred3",
           color = "black") +
  geom_text(aes(label = pctlabel),
            vjust = -0.25) +
  labs(x = "Origin City",
       y = "Frequency",
       title  = "Customers by Origin City") +
  theme(axis.text.x = element_text(angle = 45,
                                   hjust = 1))
```

Output



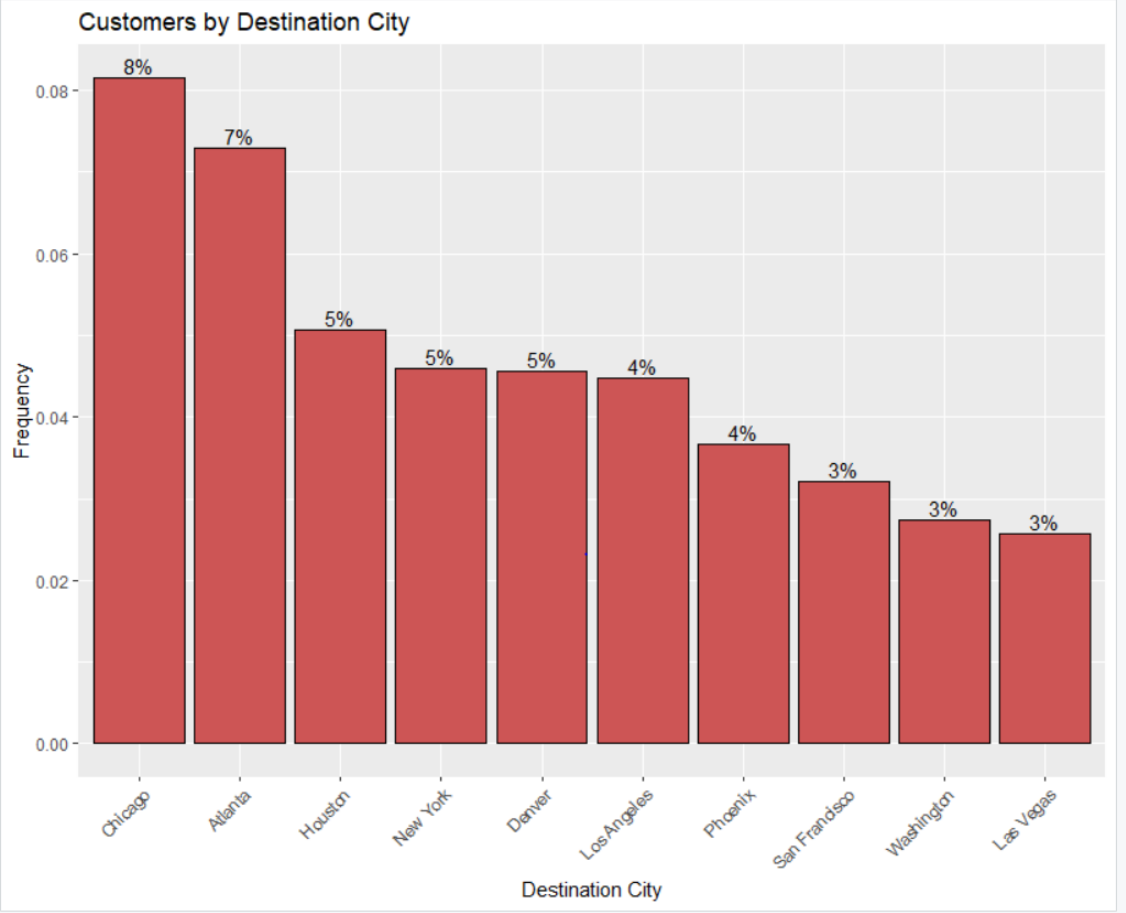
Here we arrange the passengers who are Detractors according to the origin of the flight in a descending order. The top 3 affected passengers are in Chicago, Houston and Atlanta.

Analysis 1b)

Code Snippet

```
# destination cities with lowest nps scores
flightdata2 <- low_score_flight %>%
  count(Destination.City) %>%
  mutate(pct = n / sum(n),
         pctlabel = paste0(round(pct*100), "%"))

flightdata2 <- flightdata2[with(flightdata2,order(-pct)),]
flightdata2 <- flightdata2[1:10,]

ggplot(flightdata2,
       aes(x = reorder(Destination.City, -pct),
           y = pct)) +
  geom_bar(stat = "identity",
           fill = "indianred3",
           color = "black") +
  geom_text(aes(label = pctlabel),
            vjust = -0.25) +
  labs(x = "Destination City",
       y = "Frequency",
       title  = "Customers by Destination City") +
  theme(axis.text.x = element_text(angle = 45,
                                   hjust = 1))
```

Output



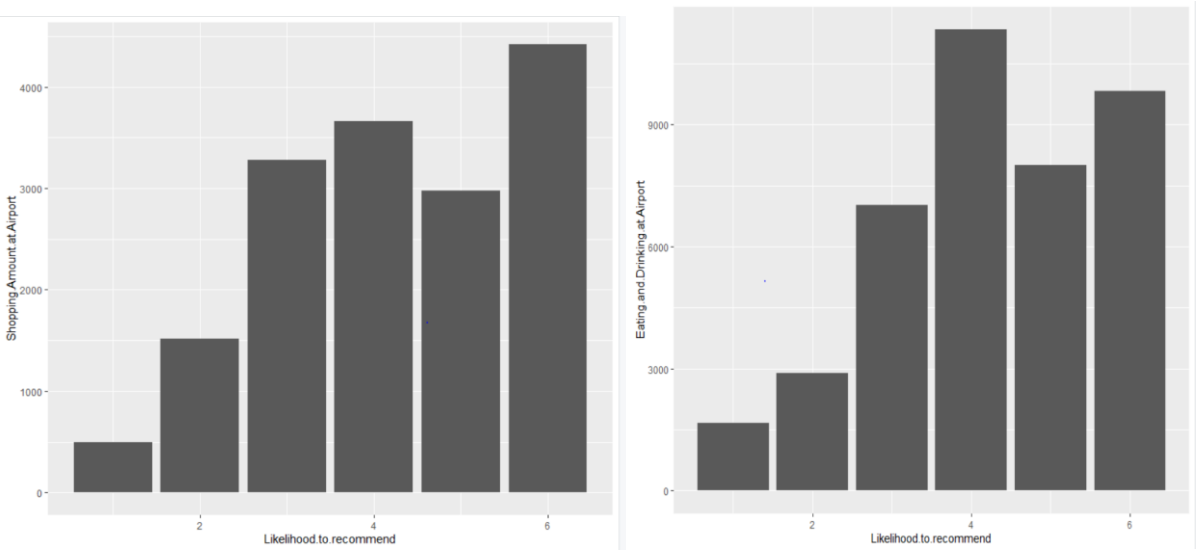
We get the same top three cities for flight destinations, consisting of detractors. Therefore, from the next step analysis and visualizations will be created on those detractors whose origin city is Chicago, Houston and Atlanta. These 3 cities will be mentioned as ***'tri-city'***

Analysis 2 a) - Chicago, Houston and Atlanta (Tri-city)

Code Snippet

```
#filter data with passengers only from top three flight origin cities where NPS are lowest amongst the majority

tri_city <- filter(low_score_flight,Origin.City == "Chicago" | Origin.City == "Atlanta" | Origin.City == "Houston")

# 1)
ggplot(tri_city,
       aes(x = Likelihood.to.recommend,
           y = Shopping.Amount.at.Airport)) +
  geom_bar(stat = "identity")

ggplot(tri_city,
       aes(x = Likelihood.to.recommend,
           y = Eating.and.Drinking.at.Airport)) +
  geom_bar(stat = "identity")
```

Output



Here, we create histograms to notice the spending habits of detractors

From the above two distributions , passengers(detractors) rating higher nps scores (Range 4-6) spend more in airport for tri-city

Spending is on shopping, food and beverages.

Therefore, customers who spend less in airport facilities tend to give low NPS score

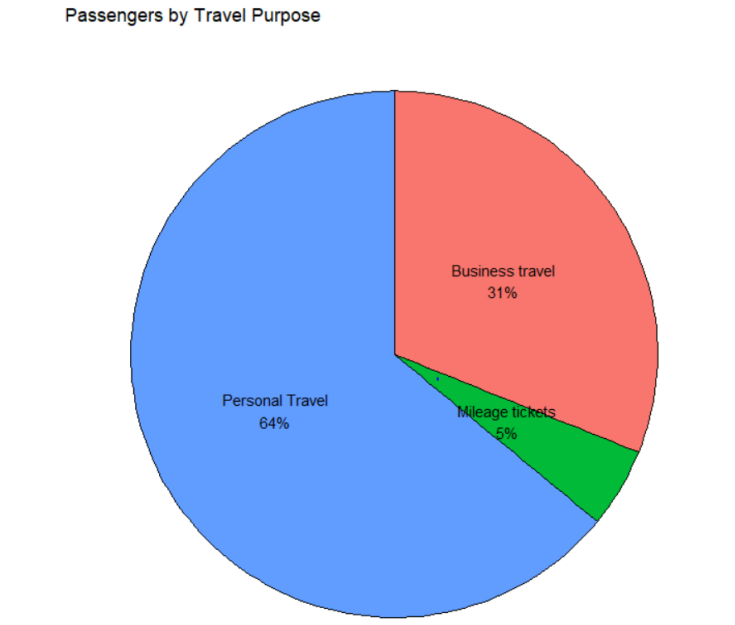
And higher spending customers are likely to give higher NPS score.

Analysis 2(b) - Chicago, Houston and Atlanta

Code Snippet

```
#2) Pie chart to show travel purpose for unsatisfied customer
tri_city2 <- tri_city %>%
  count(Type.of.Travel) %>%
  arrange(desc(Type.of.Travel)) %>%
  mutate(prop = round(n*100/sum(n), 1),
         lab.ypos = cumsum(prop) - 0.5*prop)

tri_city2$label <- paste0(tri_city2$Type.of.Travel, "\n",
                          round(tri_city2$prop), "%")

ggplot(tri_city2,
       aes(x = "",
           y = prop,
           fill = Type.of.Travel)) +
  geom_bar(width = 1,
           stat = "identity",
           color = "black") +
  geom_text(aes(y = lab.ypos, label = label),
            color = "black") +
  coord_polar("y",
              start = 0,
              direction = -1) +
  theme_void() +
  theme(legend.position = "FALSE") +
  labs(title = "Passengers by Travel Purpose")
# the most un satisfied customers travel for personal reasons.
```

Output



A pie-chart consisting of percentage values of count of passengers according to travel purpose. The most unsatisfied customers are also travelling for personal reasons 64% of the time.

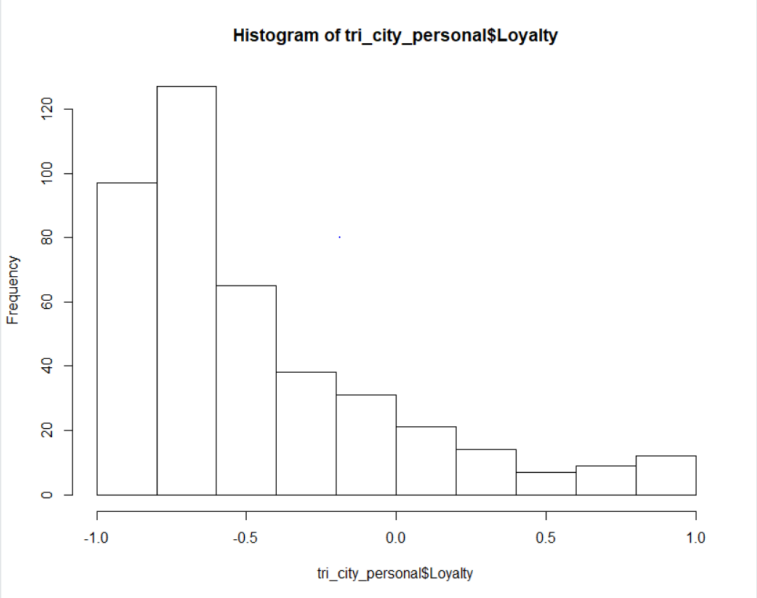
Analysis 2(c) – Personal Travel & lowest NPS scores (<6) in Chicago, Houston & Atlanta

Code Snippet

```
# 3.)
# Analysis of passengers who travel for personal reasons
tri_city_personal <- filter(tri_city, Type.of.Travel == "Personal Travel")
table(tri_city_personal$Airline.Status)
# most of the passengers who travel for personal reasons have blue airline status

table(tri_city_personal$Gender)
# most of the passengers who travel for personal reason are females

table(tri_city_personal$Class)
# passengers travel in economy class mostly while travelling for personal reasons

hist(tri_city_personal$Loyalty)
# high number of passengers who are unsatisfied are disloyal customers are frequently customers to other airlines
```

Output

```
> table(tri_city_personal$Airline.Status)

    Blue     Gold Platinum   Silver
     368       24        6       23
> table(tri_city_personal$Gender)

Female   Male
   296    125
> table(tri_city_personal$Class)

Business      Eco Eco Plus
      24      351       46
```



From the 1 st output of the data table, we observe passengers who are detractors and travel for personal reasons, mostly have BLUE airline status and around 75% of detractors are females and most of them travel by Economy Class.

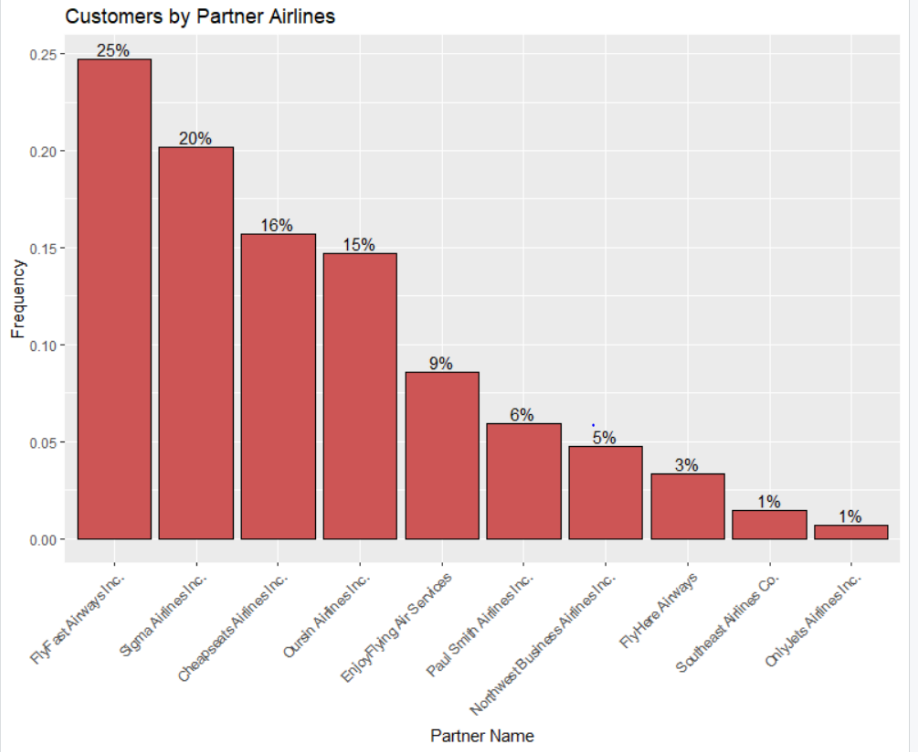
The 2nd output of a histogram indicates a right skewed distribution. Hence, ***customers who are detractors are more likely to not be loyal to the airline and mostly have a loyalty level of below 0.5***

Analysis 2(d) – No. of Customers by Partner airlines

Code Snippet

```
#4)
#Analysis of Partner Airlines on tri_city data
tri_city_partner <- tri_city_personal %>%
  count(Partner.Name) %>%
  mutate(pct = n / sum(n),
         pctlabel = paste0(round(pct*100), "%"))

ggplot(tri_city_partner,
       aes(x = reorder(Partner.Name, -pct),
           y = pct)) +
  geom_bar(stat = "identity",
           fill = "indianred3",
           color = "black") +
  geom_text(aes(label = pctlabel),
            vjust = -0.25) +
  labs(x = "Partner Name",
       y = "Frequency",
       title  = "Customers by Partner Airlines") +
  theme(axis.text.x = element_text(angle = 45,
                                   hjust = 1))

# Customers who travel with Flyfast(25% ) and Sigma(20%) for personal reasons are the most unsatisfied.
```

Output



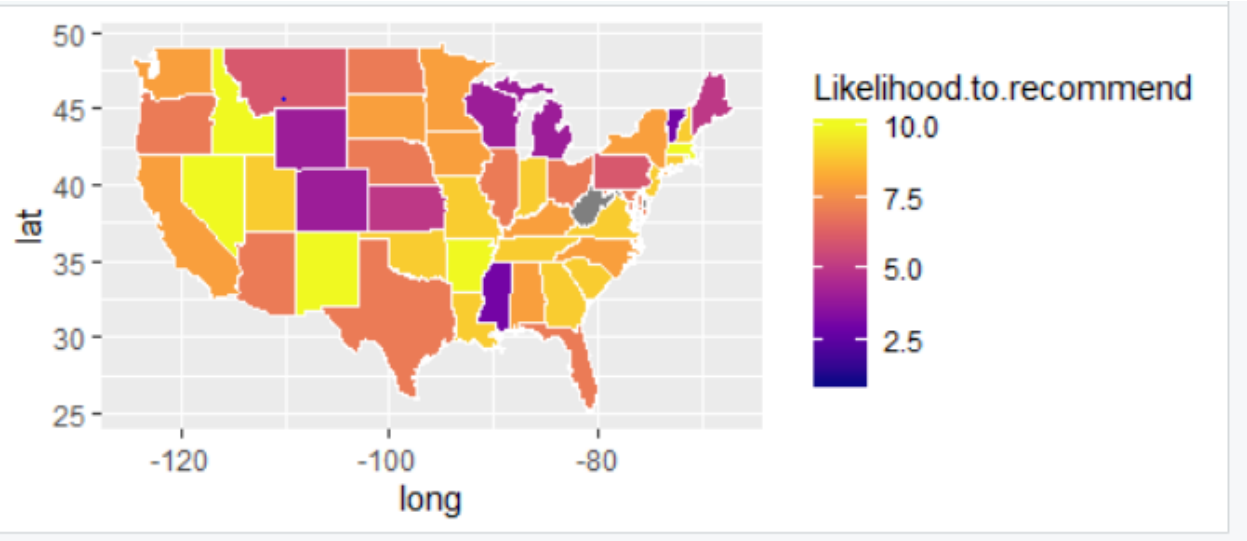
Previously, in Step 9 of Phase 2 – Visualization, it was observed that the Partner Airline - Cheapseats had the greatest number of passengers for any partner airline. But in the above output, the highest number of passengers(25% of total detractors) who are detractors, travel by the Partner Airline - FlyFast Airways Inc, followed by Sigma Airlines in the tri_city area.

Analysis 3) – Likelihood to recommend vs States

Code Snippet

```
# Map depicting NPS scores by State
results$region <- tolower(results$Origin.State)
states_map <- map_data("state")
nps_map <- left_join(states_map, results, by = "region")

ggplot(nps_map, aes(long, lat, group = group))+
  geom_polygon(aes(fill = Likelihood.to.recommend), color = "white")+
  scale_fill_viridis_c(option = "C")
```

Output



The yellow colored states indicate states with high NPS values.

The orange colored states indicate neutral to promoter behavior

The light orange – purpled colored states represent Detractors

The map indicates the NPS scores across states for total values. This map indicates the western region have high nps scores. While a couple of states in the central region have low nps scores. The eastern coast has low to average scores.