IST 687 FINAL PROJECT: GROUP 1

Cheapseats Airlines Satisfaction Analysis

Team Members:

Vishal Anantharaman
Alexander Bailey
Priyanka Guha
Ruchita Jadhav
Parth Mehta
Ishan Wagle

TABLE OF CONTENTS:

INTRODUCTION:

For this project, we have been given a dataset on various airlines and their customer details and a level of satisfaction for each customer. This dataset consisted of 129890 rows and 28 columns. We had various parameters given, which focused on aspects such as the information on customer metrics, details about airlines, parameters related to travel for customer and the airports. We try to understand how these parameters together contribute to the satisfaction rating.

The problem which we are trying to address by means of this project is to improve the customer satisfaction ratings for Cheapseats airlines, which does not know where it should focus its questions- the type of audience it should address or locations which might need more attention. Our objective is to provide recommendation for the airline to address this issue and improve their satisfaction rating.

The team began with the cleaning of data where we made the column names in a uniform format and ensured that the data in the rows and columns is consistent and without any kind of anomaly such as whitespaces. We also made the airline names easy to understand and rounded up satisfaction values. We have created various business questions which we have addressed during the course of the project.

For the analysis, we used Linear modelling, Support Vector Machines, and Association rules. By means of these models, we tried to figure out the ways in which customer satisfaction is affected and how we can improve it. We have used the models to prove the findings of the business questions .

BUSINESS QUESTIONS:

1. Is there satisfaction disparity for males and females?
2. Do loyalty cards increase satisfaction?
3. Where (geographically) are customers more/less satisfied?
4. How does airline status (Blue, Gold, Silver, Platinum) affect satisfaction?
5. What attributes explain customer satisfaction?
6. Which customer population should Cheapseats focus on to improve its customer satisfaction ratings?
7. Which customers have the higher satisfaction ratings? Delay with greater than 5 minutes or delays less than five minutes?
8. What is the age of the customers who gives higher ratings?
9. What is the range of the number of loyalty cards that a customer have?

DATA ACQUISITION, CLEANING AND TRANSFORMATION:

To gain important insights from the data, we performed a number of analysis. In order to get more accurate results, it is necessary to remove the dirty data from the database. Dirty data can be present in various forms like outliers, white spaces, NA, blank data, values in wrong format, etc. which should be taken care of to gain more accurate insights. After removing dirty data, we need to transform it into a more consistent and standard form.

To do this, we first imported the data which had 129889 rows and 28 columns at the start. We changed the names of the columns to simplify working on it. First step then was to remove the white spaces from every vector. Then we checked for unique values of variables to find out any anomaly.

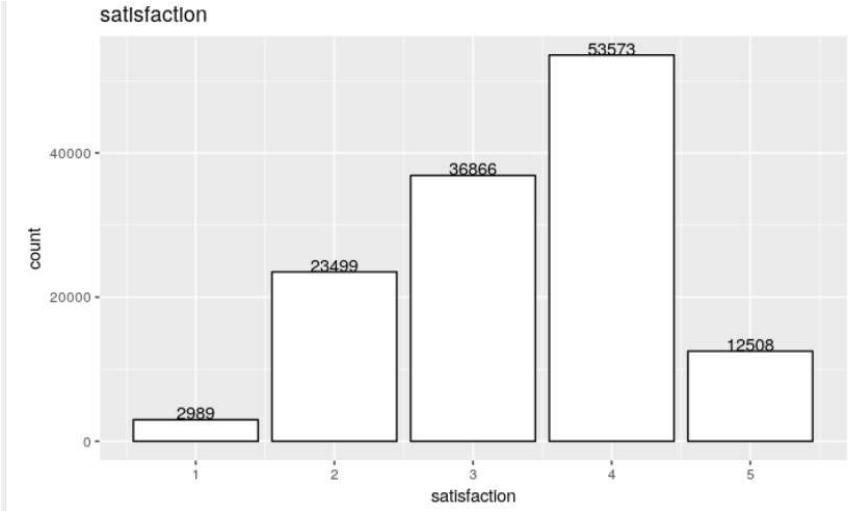
The three rows where values which were not in proper format in satisfaction column were then removed. We also rounded off of the satisfaction values to the next whole number to provide more consistency in the values. Then we targeted the NA values and replaced them by average of the whole column. To make our work simpler and to make the flight data in more accessible format, we changed the flight date column to date datatype.

To make our working more easier the type of travel variables were changed to a factor Personal, Mileage and Business; the class values were changed to Economy, Plus and Business; the gender was changed to a factor which was then set as Male and Female; the airline status was changed to Blue, Silver, Gold and Platinum; the airline names were made more standardised and concise; the flight cancelled column variables were changed to order of Yes and No; and the arrival delay greater than 5 were changed to factor of Yes and No. Finally, after cleaning and transforming the data, the resulting database had 129435 rows and 28 columns. This clean database was then converted to a .csv file to start working with it.

DESCRIPTIVE STATISTICS AND VISUALISATIONS:

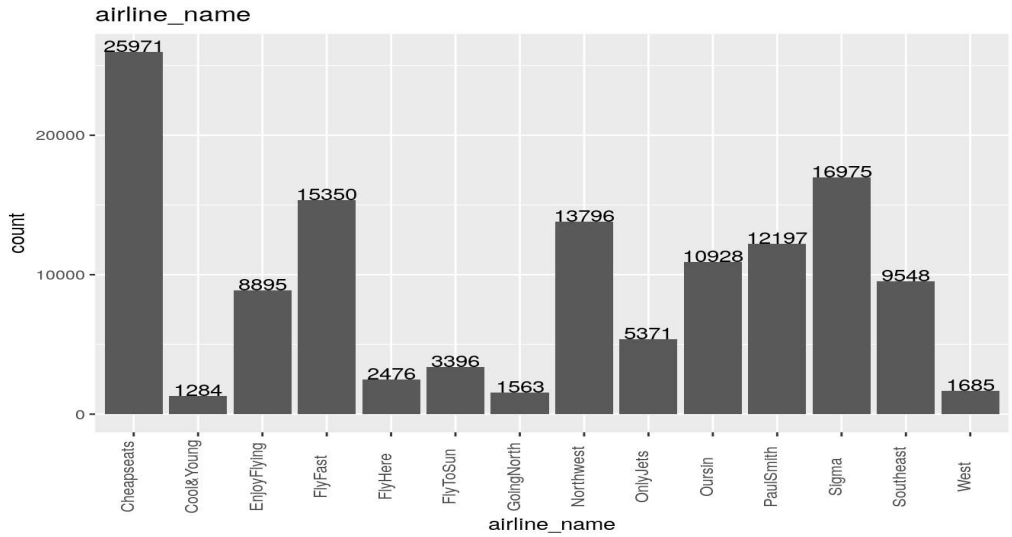
The purpose of performing the descriptive statistics and analysis on the dataset was to gain more insights and better understand the variables. Performing the analysis helped us draw certain inferences about the variables to understand them better. Following are the various analysis and statistics we performed on the variables:

1. The satisfaction distribution of various customers.



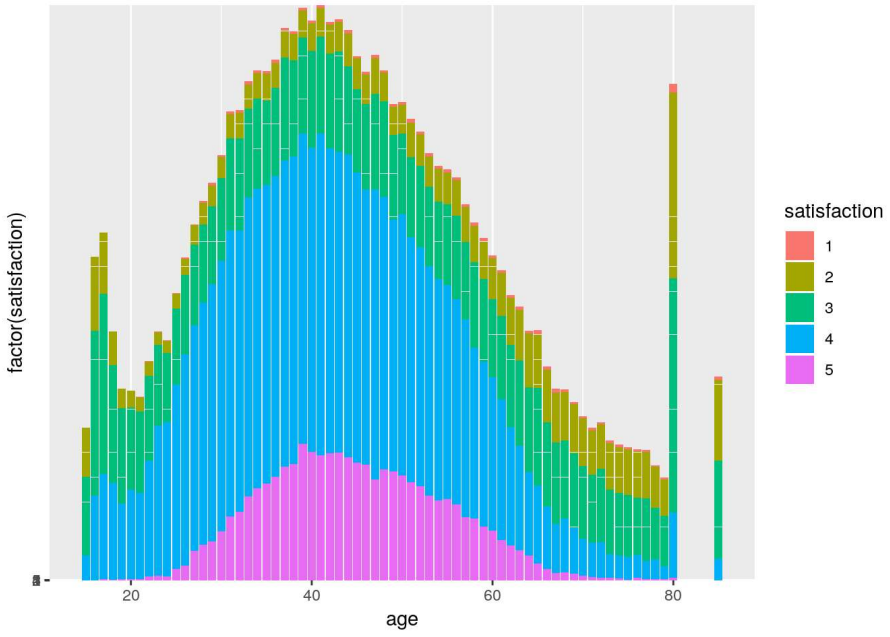
From this graph, we can see that the majority of customers have a satisfaction rate ranging from the 2.5 to 4.5

2. airline_name vs the number of people travelling through it.



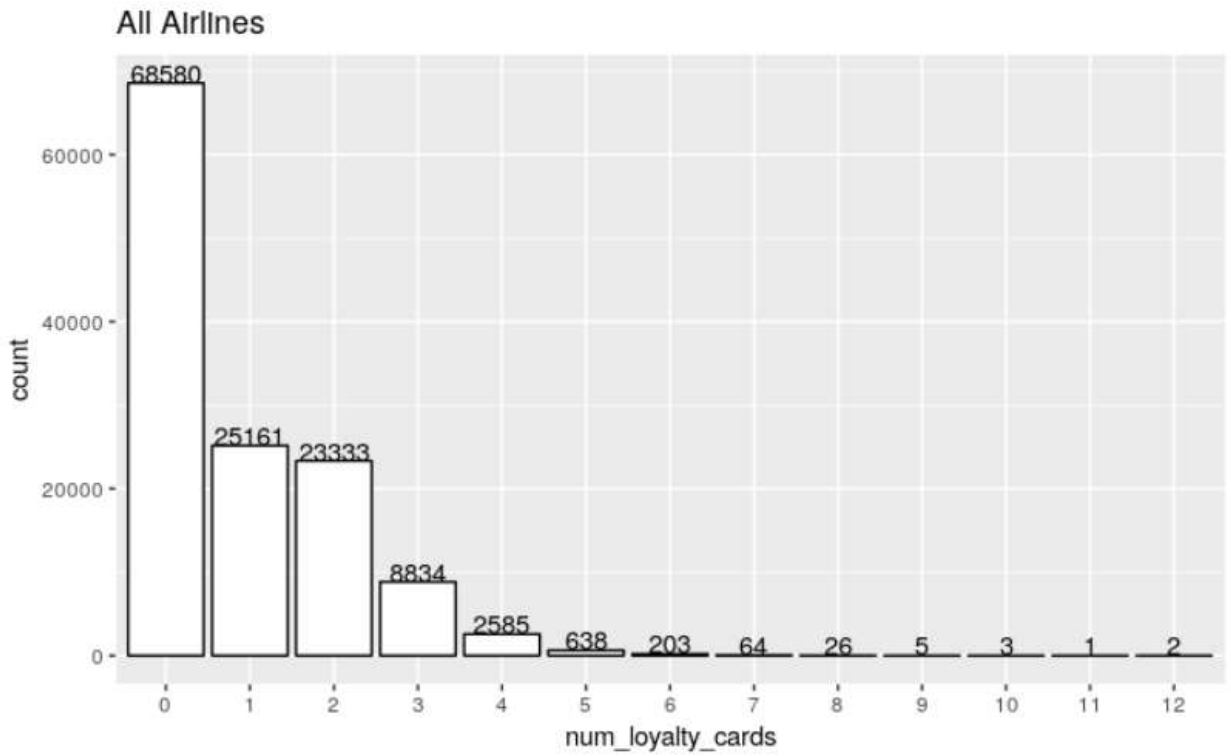
This helped us understand the most busiest and the least busiest airline. Cheapseats with 25971 passengers is the busiest one and Cool&Young with 1284 is the least busiest.

3. Customer satisfaction based on the Age.



From the analysis we can see that, the customers around the age range of 40 years have higher customer satisfaction than others. Whereas, passengers with age range less than 20 years and more than 80 years have lower satisfaction rates.

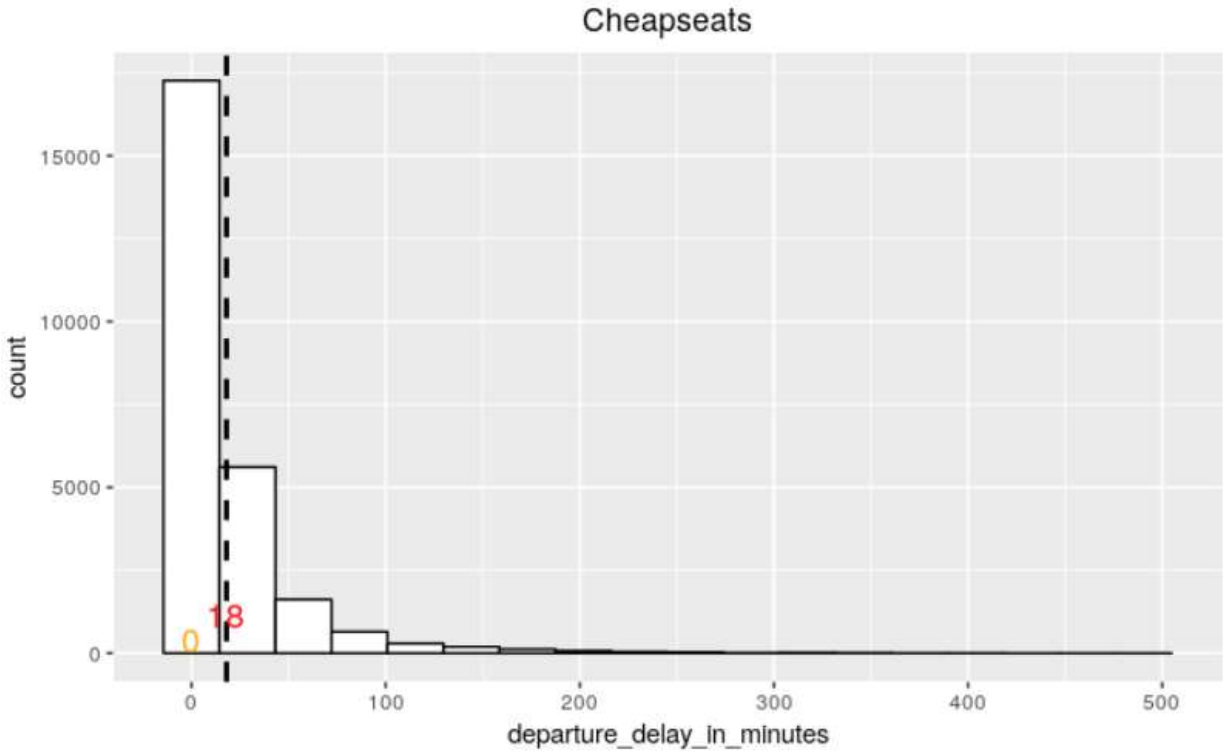
4. The number of loyalty cards of customers.

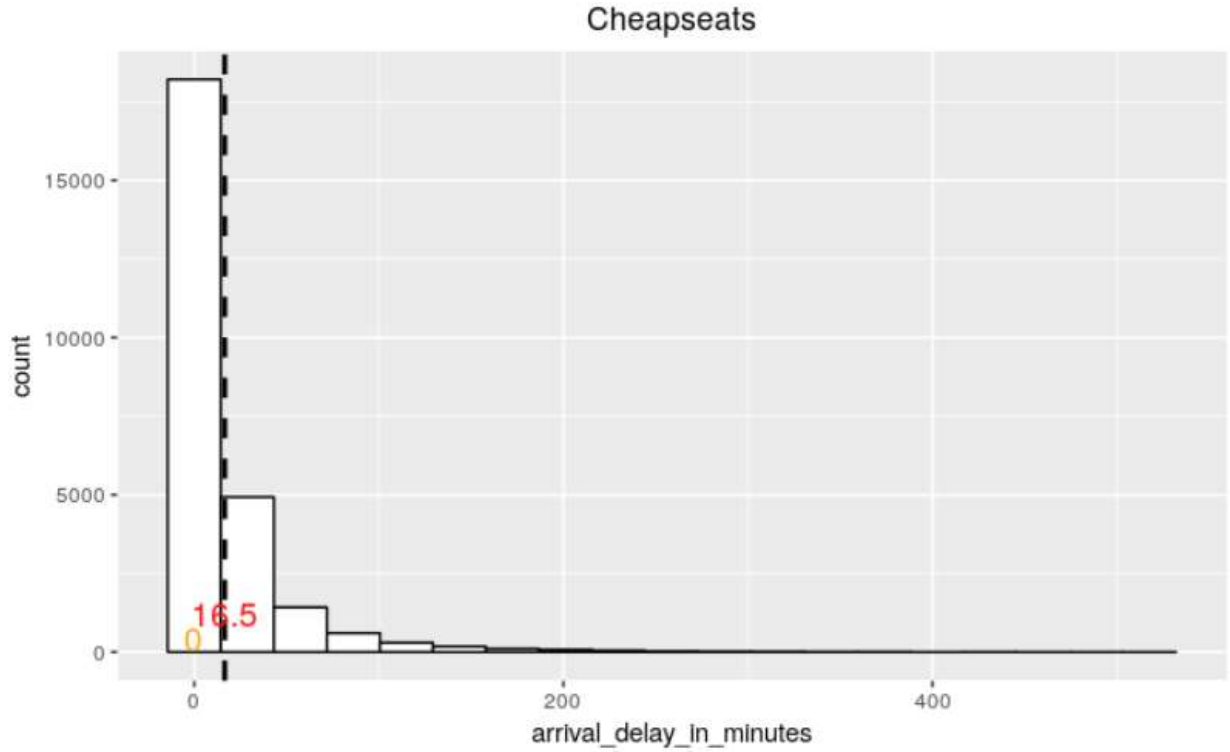


From the graph we can see that most of the customers do not have a loyalty card and those that do generally have only 1 or 2.

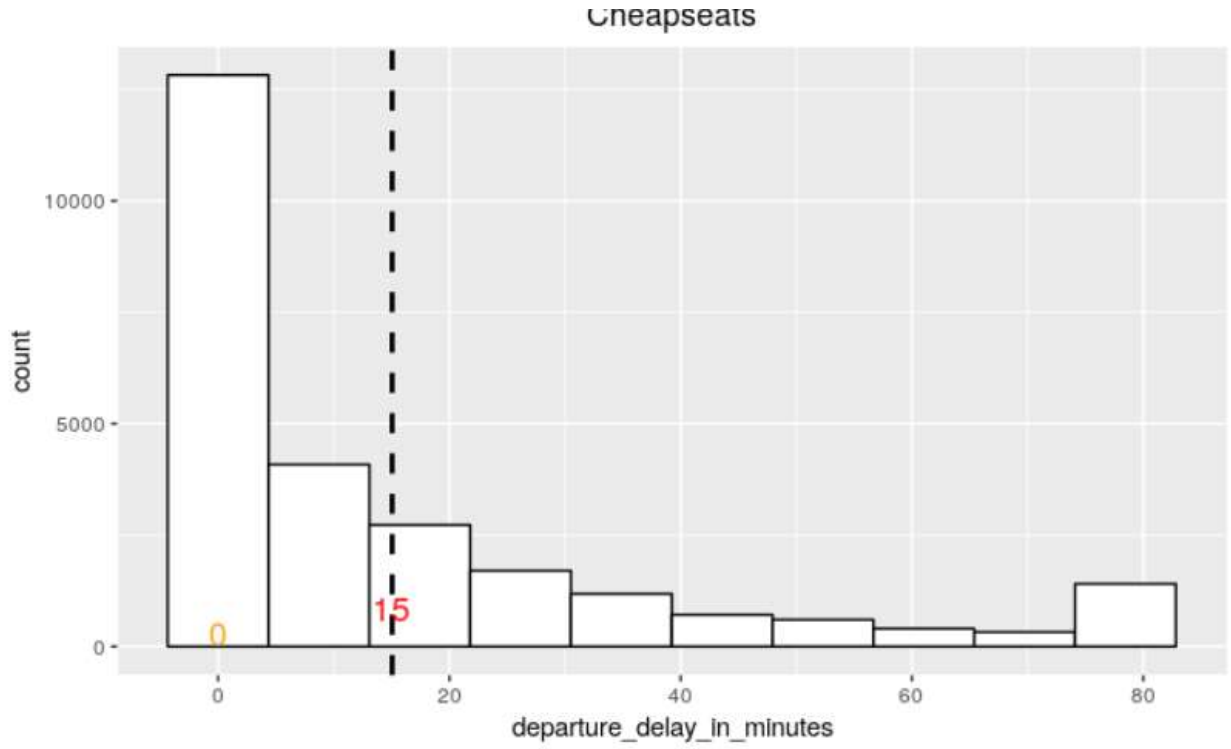
5. Winsoring the delay in departure in minutes of the flights. Cheapseats' departure and arrival delay in minutes histograms are heavily right skewed. To reduce the effect that the outliers have, we will set outliers that are greater than the 95th percentile of the entire vector to the 95th percentile value (winsorizing).

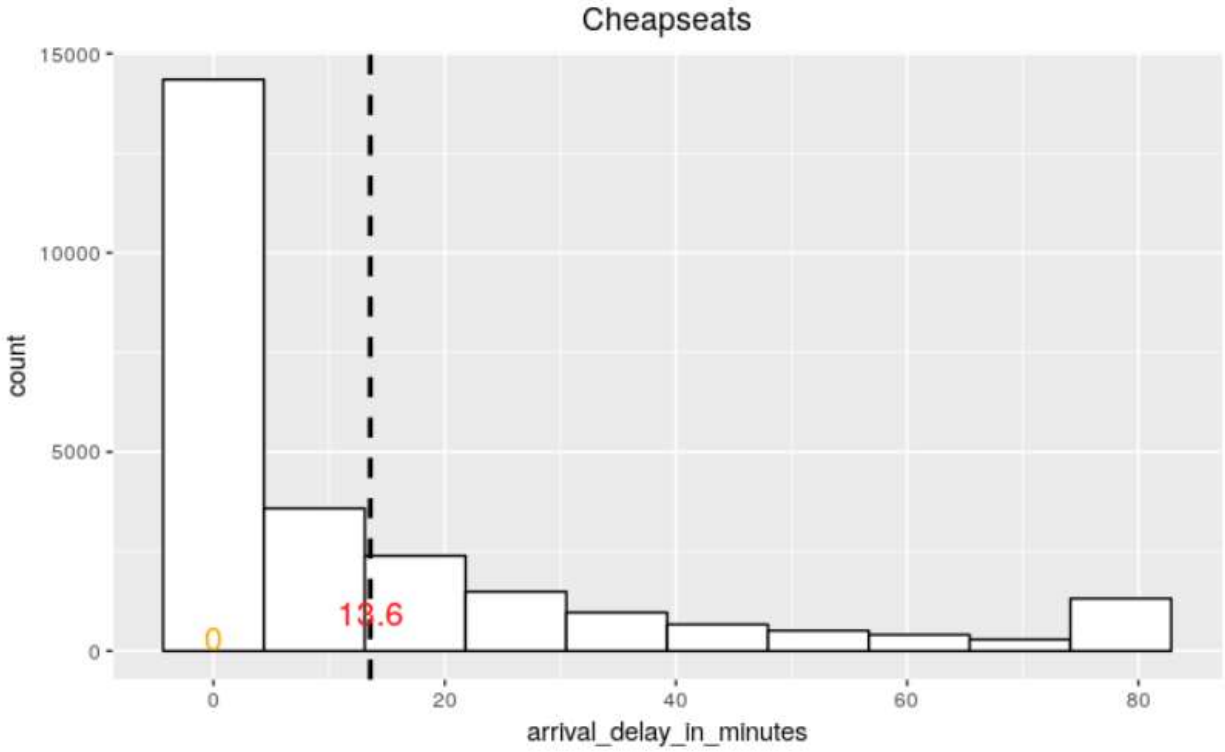
a. Before winsorizing:



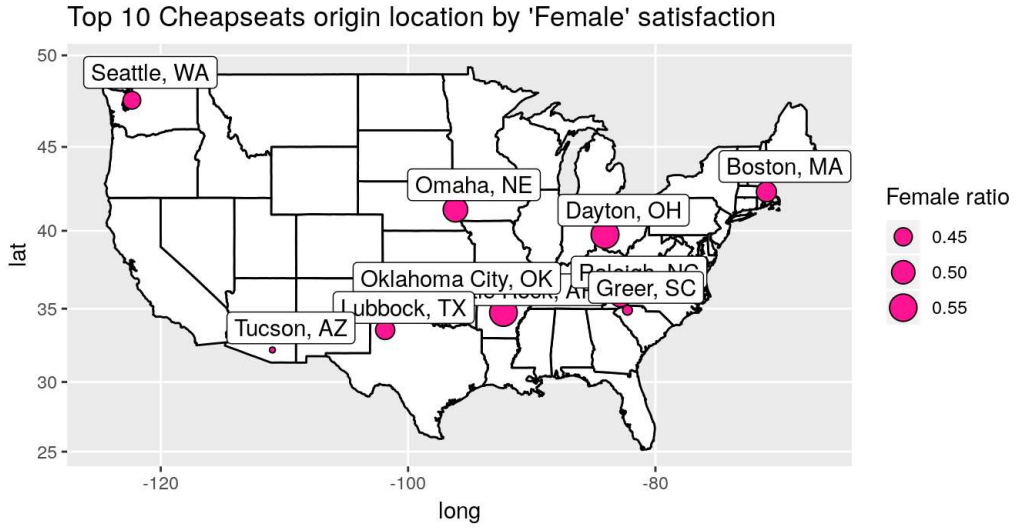


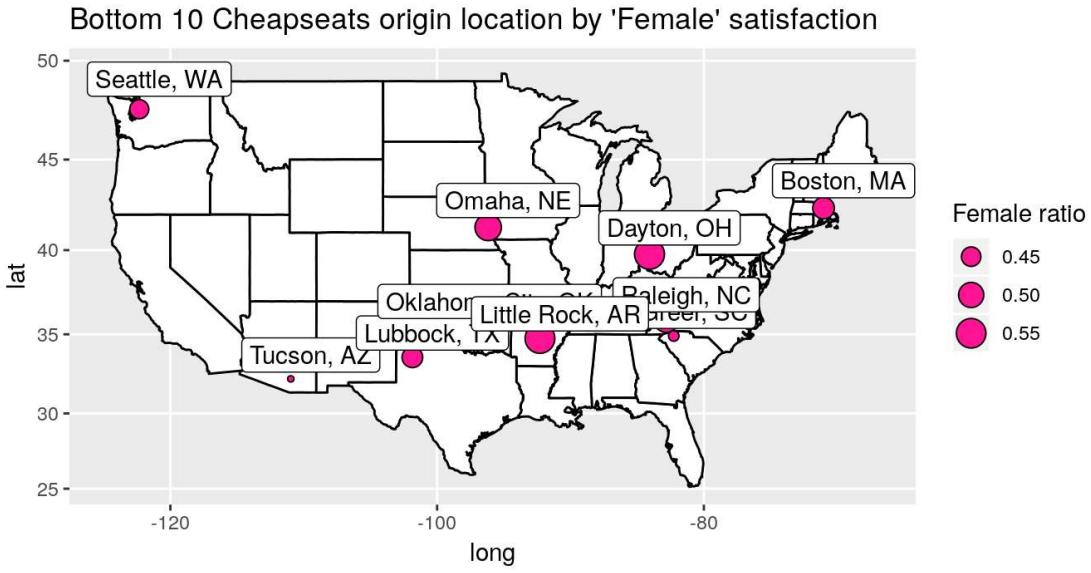
b. After winsorizing:


Cheapseats
count
10000
5000
0
0
15
0
20
40
60
80
departure_delay_in_minutes

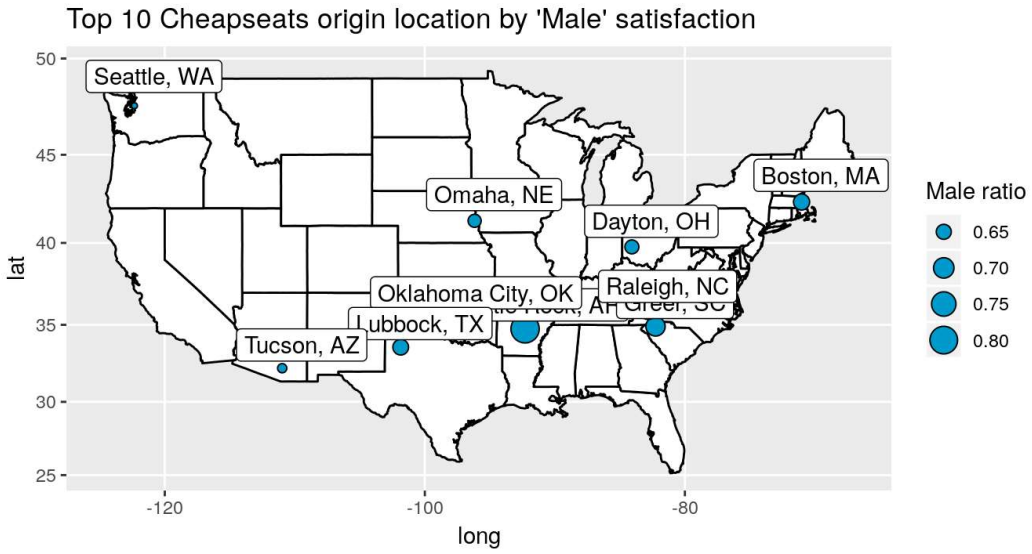

Cheapseats
count
15000
10000
5000
0
0
13.6
0
20
40
60
80
arrival_delay_in_minutes

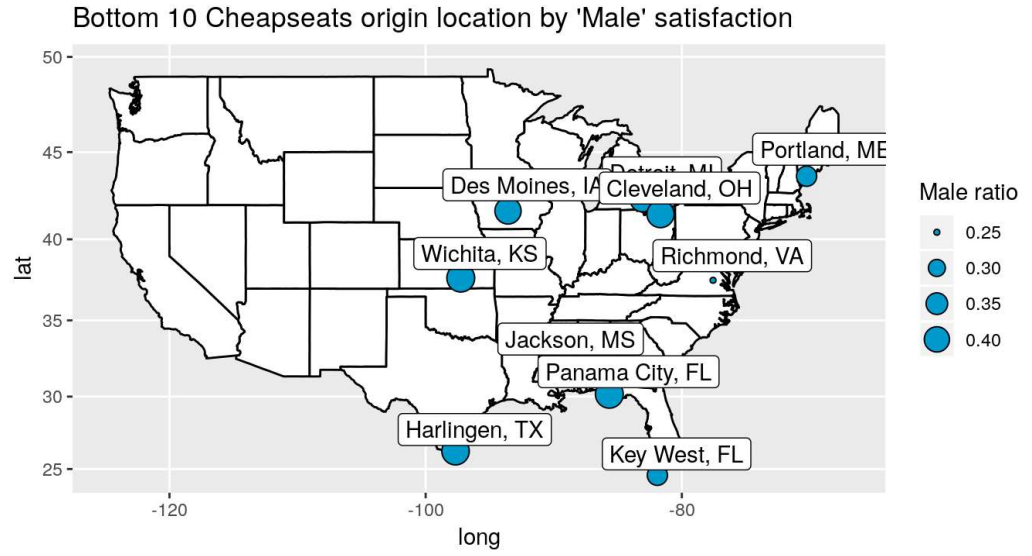
6. Satisfaction based on male and female





The above graphs shows the 10 locations with highest female satisfaction for Cheapseats airlines as well the 10 locations with least satisfaction for Cheapseats airlines. The

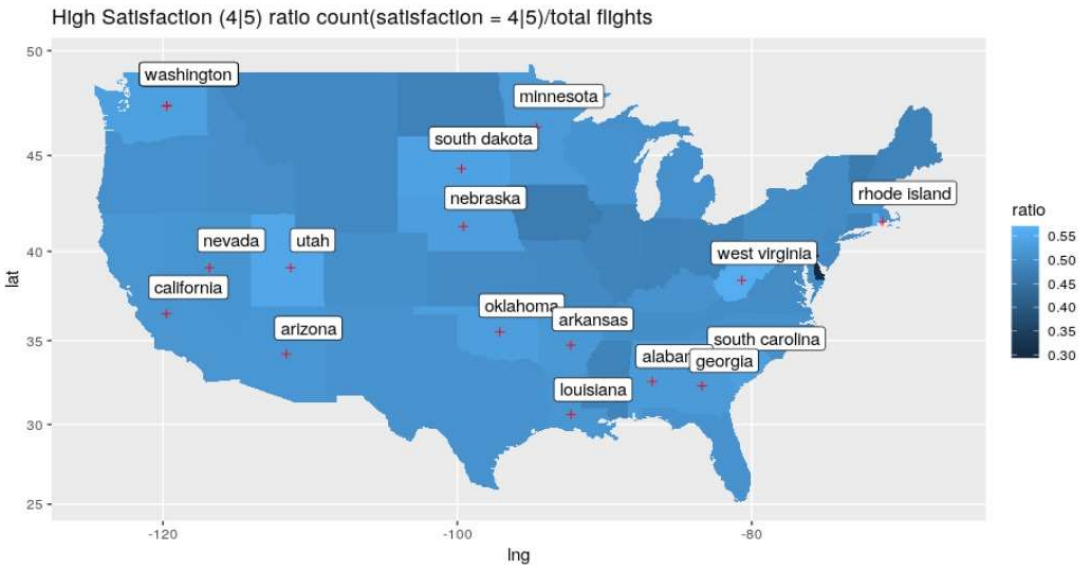




The above graphs shows the 10 locations with highest male satisfaction for Cheapseats airlines as well the 10 locations with least satisfaction for Cheapseats airlines.

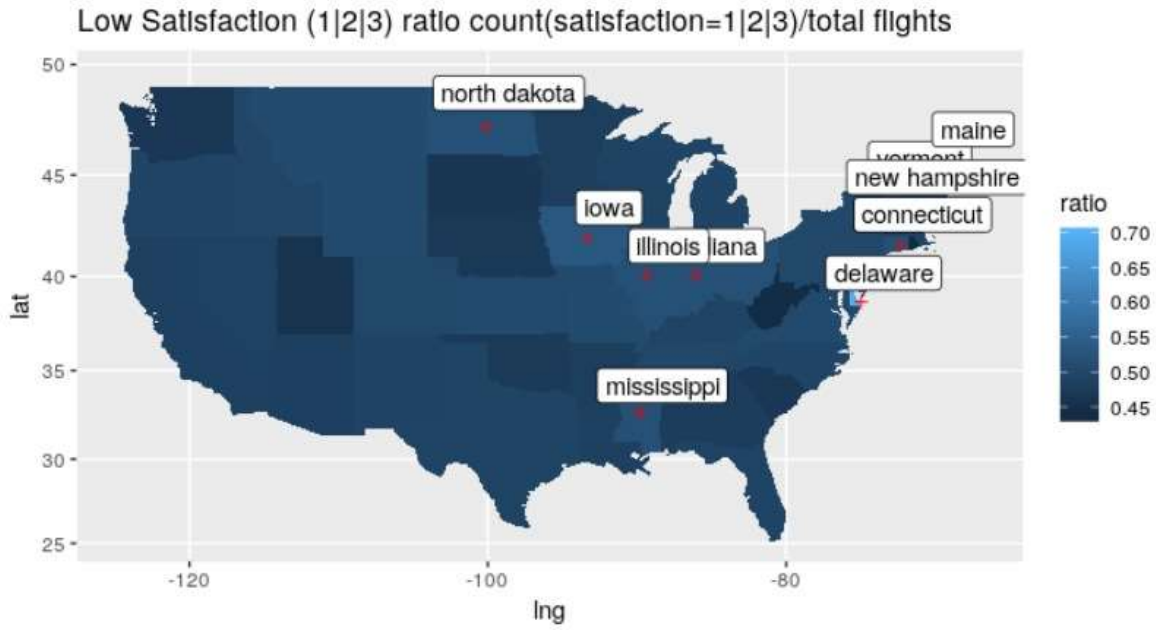
8. Satisfaction based on geography:

A) High Satisfaction:



The above graph shows the customers with high satisfaction based on locations. It also shows some of the locations with the highest customer satisfactions.

B) Low Satisfaction:



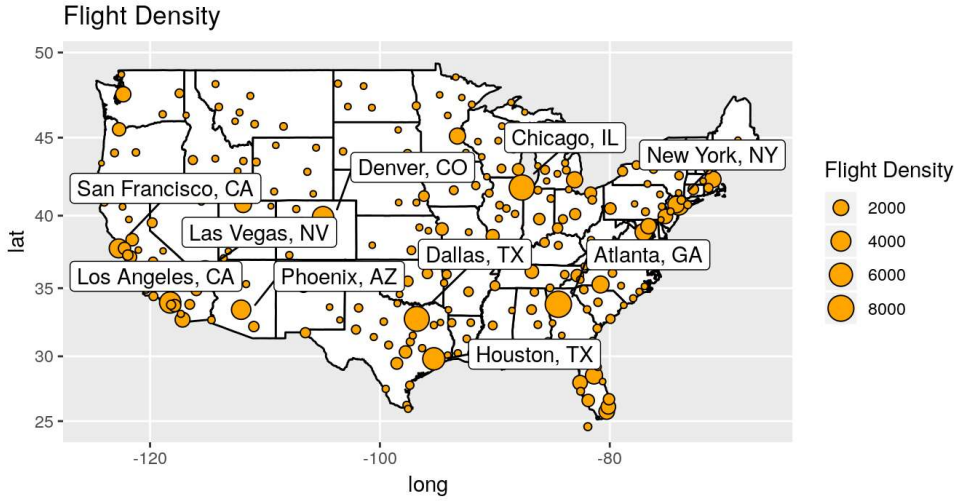
The above graph shows the customers with low satisfaction based on locations and some states where the customers are least satisfied.

9. Satisfaction based on airline status:

```
                        Estimate Std. Error t value Pr(>|t|)
(Intercept)            3.7727675  0.0150387 250.871  < 2e-16 ***
airline_statusGold     0.4449571  0.0167983  26.488  < 2e-16 ***
airline_statusPlatinum 0.2906776  0.0260358  11.165  < 2e-16 ***
airline_statusSilver   0.6496885  0.0115067  56.462  < 2e-16 ***
```

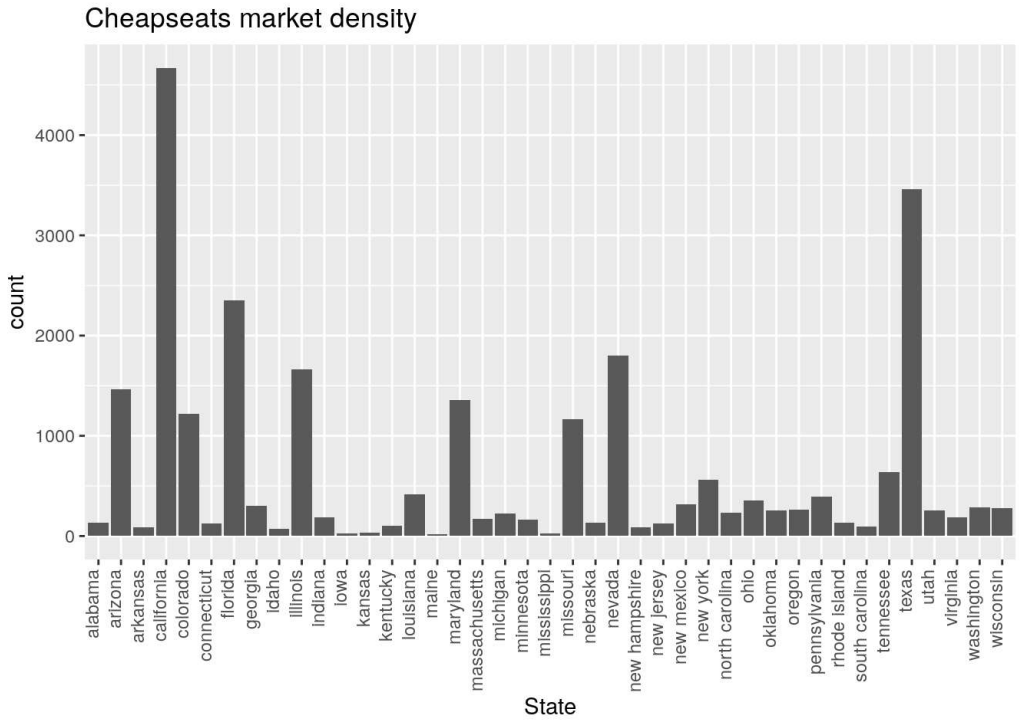
The above snippet shows us the satisfaction of customers based on the airline status. By performing linear modelling technique on the airline status versus the satisfaction we get Silver status with highest satisfaction ratings as compared to others.

10. Flight density for all Airlines -- including Cheapseats:



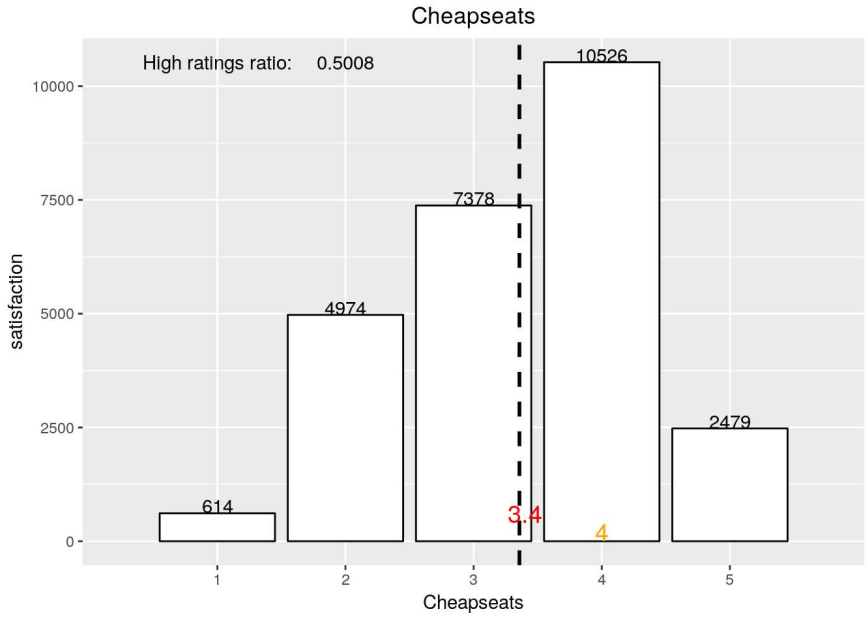
The above graph depicts the density of the flights based on their locations. As seen above, greater the size of the circle, there are more flights to that location, and smaller the size of the circle lower is the density which means that there are fewer flights to that location..

11. Market density for Cheapseats:



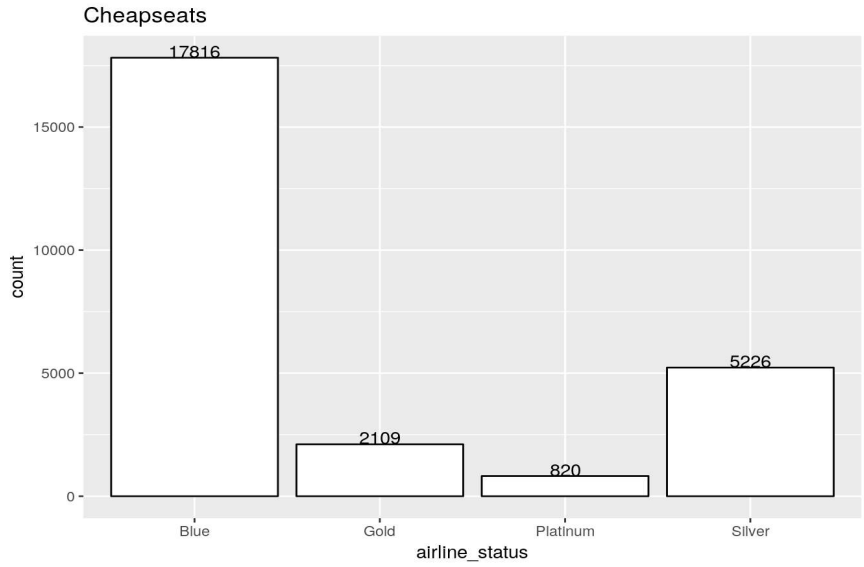
The above graph depicts the market density for Cheapseats airlines. It can be inferred that california has the highest density while states like Maine, Mississippi, etc. have low densities.

12. Satisfaction count for Cheapseats:



The above graphs shows the number of customers in various satisfaction range. It can be inferred that there are a large number of customers having satisfaction of 4 with average being 3.4 and a mode of 4.

13. Customer distribution based on airline status for Cheapseats:



The graphs depicts the number of customers based on airline status. It can be inferred that there are 17816 customers having the blue status when flying through Cheapseats airlines and the lowest being Platinum status with 820 customers.

MODELLING TECHNIQUES:

1. LINEAR MODELLING:

To find relationships between the independent variables like airline_status, age, gender, type_of_travel, arrival_delay_greater_5_mins, num_flights we performed a number of linear models that can explain 44% of the variance around the mean of the dependent variable satisfaction.

The first step was to winsorize departure_delay_in_minutes and arrival_delay_in_minutes which we found out by observing the number of rows that falls above the 95th percentile.

```
departure_delay_95 <- quantile(df$departure_delay_in_minutes, .95)
sprintf("%d rows will be changed to the 95th percentile of departure_delay_in_minutes", length(which(df$departure
_delay_in_minutes > departure_delay_95)))
```

```
## [1] "1290 rows will be changed to the 95th percentile of departure_delay_in_minutes"
```

```
df[which(df$departure_delay_in_minutes > departure_delay_95), "departure_delay_in_minutes"] = departure_delay_95

arrival_delay_95 <- quantile(df$arrival_delay_in_minutes, .95)
sprintf("%d rows will be changed to the 95th percentile of arrival_delay_in_minutes", length(which(df$arrival_del
ay_in_minutes > arrival_delay_95)))
```

```
## [1] "1289 rows will be changed to the 95th percentile of arrival_delay_in_minutes"
```

```
df[which(df$arrival_delay_in_minutes > arrival_delay_95),"arrival_delay_in_minutes"] = arrival_delay_95
```

The second step was to get the r-squared values for all the variables as predictors of satisfaction

```
createLM <- function(mydf) {
  r_squares <- list()
  for(i in 2:ncol(mydf)) {
    model <- lm(formula=satisfaction~mydf[[i]], data=mydf)
    r_squares[[i]] <- summary(model)
  }

  model_names <- c('satisfaction','airline_status', 'age', 'gender', 'price_sensitivity', 'year_first_flight','nu
m_flights', 'percent_flight_other_airlines', 'type_of_travel','num_loyalty_cards', 'airport_shopping', 'airport_d
ining','class', 'day_of_month', 'flight_date', 'airline_code', 'airline_name', 'origin_city', 'origin_state', 'de
stination_city','destination_state', 'scheduled_departure_hour', 'departure_delay_in_minutes', 'arrival_delay_in_
minutes', 'flight_cancelled','flight_time_in_minutes', 'flight_distance', 'arrival_delay_greater_5_mins')

  names(r_squares) <- model_names  # name lists

  return(r_squares)
}

r_squares <- createLM(df)
for(i in 2:length(r_squares)) {
  v_name <- names(r_squares)[i]
  print(sprintf("The r-squared value for %s: %f", v_name, r_squares[[i]]$r.squared))
}
```

The values which we got for the r-squared values for each column.

```
## [1] "The r-squared value for airline_status: 0.123284"
## [1] "The r-squared value for age: 0.049398"
## [1] "The r-squared value for gender: 0.018651"
## [1] "The r-squared value for price_sensitivity: 0.009431"
## [1] "The r-squared value for year_first_flight: 0.000059"
## [1] "The r-squared value for num_flights: 0.055542"
## [1] "The r-squared value for percent_flight_other_airlines: 0.004124"
## [1] "The r-squared value for type_of_travel: 0.335375"
## [1] "The r-squared value for num_loyalty_cards: 0.007634"
## [1] "The r-squared value for airport_shopping: 0.000169"
## [1] "The r-squared value for airport_dining: 0.000049"
## [1] "The r-squared value for class: 0.002290"
## [1] "The r-squared value for day_of_month: 0.000017"
## [1] "The r-squared value for flight_date: 0.003585"
## [1] "The r-squared value for origin_city: 0.002952"
## [1] "The r-squared value for origin_state: 0.001570"
## [1] "The r-squared value for destination_city: 0.003460"
## [1] "The r-squared value for destination_state: 0.001681"
## [1] "The r-squared value for scheduled_departure_hour: 0.000588"
## [1] "The r-squared value for departure_delay_in_minutes: 0.009210"
## [1] "The r-squared value for arrival_delay_in_minutes: 0.011927"
## [1] "The r-squared value for flight_cancelled: 0.000671"
## [1] "The r-squared value for flight_time_in_minutes: 0.000013"
## [1] "The r-squared value for flight_distance: 0.000005"
## [1] "The r-squared value for arrival_delay_greater_5_mins: 0.026848"
```

The third step was to winsorizing the departure/arrival delays changed the coefficients from *.005 and .007* to *.010 and .014* respectively. For that we performed stepwise regression (backwards) to find best predictors.

```
library(MASS)
model <- lm(satisfaction~airline_status+age+gender+type_of_travel+arrival_delay_greater_5_mins+num_flights+arriva
l_delay_in_minutes+departure_delay_in_minutes+num_loyalty_cards, data=df)
selected_model <- stepAIC(model, direction="backward", trace=TRUE)
```

```
## Start:  AIC=-16894.93
## satisfaction ~ airline_status + age + gender + type_of_travel +
##     arrival_delay_greater_5_mins + num_flights + arrival_delay_in_minutes +
##     departure_delay_in_minutes + num_loyalty_cards
##
##                                Df Sum of Sq   RSS      AIC
## - num_loyalty_cards             1       0.1 13537 -16896.7
## - arrival_delay_in_minutes      1       0.5 13538 -16896.0
## <none>                                      13537 -16894.9
## - departure_delay_in_minutes    1       1.7 13539 -16893.6
## - age                           1      27.8 13565 -16843.7
## - num_flights                   1      39.5 13577 -16821.3
## - gender                        1      94.9 13632 -16715.5
## - arrival_delay_greater_5_mins  1     375.9 13913 -16185.5
## - airline_status                3    1820.4 15358 -13624.2
## - type_of_travel                2    5208.8 18746  -8444.4
##
## Step:  AIC=-16896.7
## satisfaction ~ airline_status + age + gender + type_of_travel +
##     arrival_delay_greater_5_mins + num_flights + arrival_delay_in_minutes +
##     departure_delay_in_minutes
##
##                                Df Sum of Sq   RSS      AIC
## - arrival_delay_in_minutes      1       0.5 13538 -16897.8
## <none>                                      13537 -16896.7
## - departure_delay_in_minutes    1       1.7 13539 -16895.4
## - age                           1      31.7 13569 -16838.0
## - num_flights                   1      39.6 13577 -16822.9
## - gender                        1      95.2 13633 -16716.6
## - arrival_delay_greater_5_mins  1     375.9 13913 -16187.5
## - airline_status                3    1820.3 15358 -13626.2
## - type_of_travel                2    5234.5 18772  -8410.7
##
## Step:  AIC=-16897.8
## satisfaction ~ airline_status + age + gender + type_of_travel +
##     arrival_delay_greater_5_mins + num_flights + departure_delay_in_minutes
##
##                                Df Sum of Sq   RSS      AIC
## <none>                                      13538 -16897.8
## - departure_delay_in_minutes    1       2.2 13540 -16895.6
## - age                           1      31.6 13570 -16839.2
## - num_flights                   1      39.7 13578 -16823.8
## - gender                        1      95.3 13633 -16717.7
## - arrival_delay_greater_5_mins  1     440.1 13978 -16068.9
## - airline_status                3    1820.2 15358 -13627.6
## - type_of_travel                2    5235.1 18773  -8411.2
```

```
summary(selected_model)
```

```
##
## Call:
## lm(formula = satisfaction ~ airline_status + age + gender + type_of_travel +
##     arrival_delay_greater_5_mins + num_flights + departure_delay_in_minutes,
##     data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -3.08100 -0.40458  0.02695  0.49302  2.76463
##
## Coefficients:
##                                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)                     3.7727675  0.0150387 250.871  < 2e-16 ***
## airline_statusGold              0.4449571  0.0167983  26.488  < 2e-16 ***
## airline_statusPlatinum          0.2906776  0.0260358  11.165  < 2e-16 ***
## airline_statusSilver            0.6496885  0.0115067  56.462  < 2e-16 ***
## age                            -0.0021793  0.0002799  -7.786 7.19e-15 ***
## genderMale                      0.1243370  0.0091996  13.515  < 2e-16 ***
## type_of_travelMileage          -0.1572769  0.0173133  -9.084  < 2e-16 ***
## type_of_travelPersonal         -1.0909831  0.0109545 -99.592  < 2e-16 ***
## arrival_delay_greater_5_minsyes -0.3496571  0.0120358 -29.052  < 2e-16 ***
## num_flights                    -0.0029503  0.0003382  -8.723  < 2e-16 ***
## departure_delay_in_minutes      0.0005591  0.0002716   2.058   0.0396 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7221 on 25960 degrees of freedom
## Multiple R-squared:  0.4487, Adjusted R-squared:  0.4485
## F-statistic:  2113 on 10 and 25960 DF,  p-value: < 2.2e-16
```

Interpretation of Adj R-squared and coefficients

- This model explains 44% (Adj. R-squared value) of the variance around the mean of the dependent variable satisfaction
- Customers who are either Gold, Platinum, or Silver have satisfaction ratings of (.44, .29, and .64) higher than customers that don't have a flight_status
- The "Silver" status has the strongest positive correlation with customer satisfaction amongst the airline status categories. Its' coefficient is interpreted as, "for Silver status, and holding all other variables constant, the satisfaction score is .64 higher for Silver flyers than for all other airline_status categories"
- Holding all other variables constant, male customers' satisfaction scores are .13 higher than female customers

- Holdling all other variables constant, customers traveling for “Mileage” have satisfaction scores -.15 less than customers traveling for “Business”
- Holding all other variables constant, customers traveling for “Personal” have satisfaction scores -1.1 less than customers traveling for “Business”
- Holding all other variables constant, customers with flights delays greater than 5 minutes have satisfaction scores -.33 less than customers with flight delays less than 5 minutes

The fourth step was to reorder the factors for airline_status and type_of_travel to see what are the coefficients for the current reference variables “Blue” and “Business”.

```
df$airline_status <- factor(df$airline_status, levels=c("Silver","Gold","Platinum", "Blue"))
df$type_of_travel <- factor(df$type_of_travel, levels=c("Personal","Business","Mileage"))
# create the model again
model <- lm(satisfaction~airline_status+age+gender+type_of_travel+arrival_delay_greater_5_mins+num_flights+arriva
l_delay_in_minutes+departure_delay_in_minutes+num_loyalty_cards, data=df)
selected_model <- stepAIC(model, direction="backward", trace=TRUE)
```

```
## Start:  AIC=-16894.93
## satisfaction ~ airline_status + age + gender + type_of_travel +
##     arrival_delay_greater_5_mins + num_flights + arrival_delay_in_minutes +
##     departure_delay_in_minutes + num_loyalty_cards
##
##                                Df Sum of Sq   RSS      AIC
## - num_loyalty_cards             1       0.1 13537 -16896.7
## - arrival_delay_in_minutes      1       0.5 13538 -16896.0
## <none>                                      13537 -16894.9
## - departure_delay_in_minutes    1       1.7 13539 -16893.6
## - age                           1      27.8 13565 -16843.7
## - num_flights                   1      39.5 13577 -16821.3
## - gender                        1      94.9 13632 -16715.5
## - arrival_delay_greater_5_mins  1     375.9 13913 -16185.5
## - airline_status                3    1820.4 15358 -13624.2
## - type_of_travel                2    5208.8 18746  -8444.4
##
## Step:  AIC=-16896.7
## satisfaction ~ airline_status + age + gender + type_of_travel +
##     arrival_delay_greater_5_mins + num_flights + arrival_delay_in_minutes +
##     departure_delay_in_minutes
##
##                                Df Sum of Sq   RSS      AIC
## - arrival_delay_in_minutes      1       0.5 13538 -16897.8
## <none>                                      13537 -16896.7
## - departure_delay_in_minutes    1       1.7 13539 -16895.4
## - age                           1      31.7 13569 -16838.0
## - num_flights                   1      39.6 13577 -16822.9
## - gender                        1      95.2 13633 -16716.6
## - arrival_delay_greater_5_mins  1     375.9 13913 -16187.5
```

```
## - airline_status                 3    1820.3 15358 -13626.2
## - type_of_travel                 2    5234.5 18772  -8410.7
##
## Step:  AIC=-16897.8
## satisfaction ~ airline_status + age + gender + type_of_travel +
##     arrival_delay_greater_5_mins + num_flights + departure_delay_in_minutes
##
##                                Df Sum of Sq   RSS      AIC
## <none>                                      13538 -16897.8
## - departure_delay_in_minutes    1       2.2 13540 -16895.6
## - age                           1      31.6 13570 -16839.2
## - num_flights                   1      39.7 13578 -16823.8
## - gender                        1      95.3 13633 -16717.7
## - arrival_delay_greater_5_mins  1     440.1 13978 -16068.9
## - airline_status                3    1820.2 15358 -13627.6
## - type_of_travel                2    5235.1 18773  -8411.2
```

```
summary(selected_model)
```

```
##
## Call:
## lm(formula = satisfaction ~ airline_status + age + gender + type_of_travel +
##     arrival_delay_greater_5_mins + num_flights + departure_delay_in_minutes,
##     data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -3.08100 -0.40458  0.02695  0.49302  2.76463
""
##
## Coefficients:
##                                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)                      3.3314730  0.0208838 159.525  < 2e-16 ***
## airline_statusGold              -0.2047314  0.0186525 -10.976  < 2e-16 ***
## airline_statusPlatinum          -0.3590109  0.0272312 -13.184  < 2e-16 ***
## airline_statusBlue              -0.6496885  0.0115067 -56.462  < 2e-16 ***
## age                             -0.0021793  0.0002799  -7.786 7.19e-15 ***
## genderMale                       0.1243370  0.0091996  13.515  < 2e-16 ***
## type_of_travelBusiness           1.0909831  0.0109545  99.592  < 2e-16 ***
## type_of_travelMileage            0.9337062  0.0185542  50.323  < 2e-16 ***
## arrival_delay_greater_5_minsyes -0.3496571  0.0120358 -29.052  < 2e-16 ***
## num_flights                     -0.0029503  0.0003382  -8.723  < 2e-16 ***
## departure_delay_in_minutes       0.0005591  0.0002716   2.058   0.0396 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7221 on 25960 degrees of freedom
## Multiple R-squared:  0.4487, Adjusted R-squared:  0.4485
## F-statistic:  2113 on 10 and 25960 DF,  p-value: < 2.2e-16
```

According to google, a lower AIC is always better, so below is the best model:
satisfaction ~ airline_status + age + gender + type_of_travel + arrival_delay_greater_5_mins + num_flights (best linear model). This model explains 44% of the variance around the mean of the dependent variable satisfaction

summary of analysis

- The independent variables: airline_status + age + gender + type_of_travel + arrival_delay_greater_5_mins + num_flights product a linear model that can explain 44% of the variance around the mean of the dependent variable “satisfaction”"
- “Silver” status customers have the highest satisfaction ratings
- “Business” travelers have the highest satisfaction ratings
- “Males” have higher ratings than females
- Customers with delays less than five minutes have higher satisfaction ratings
- *All interpretations have the condition of keeping all other independent variables constant*

2. ASSOCIATION RULES:

Firstly, it is important to convert the columns in the data set which are not factors into buckets. The buckets for those columns are created by forming subsets of values belonging to that particular column.

```
#Creating buckets for all columns
df$age <- cut(df$age, c(14,34,46,58,86))
df$price_sensitivity <- cut(df$price_sensitivity, c(-1,0,1,3,6))
df$year_first_flight <- cut(df$year_first_flight, breaks=3)
df$num_flights <- cut(df$num_flights, c(-1,0,9,18,21,30,101))
df$percent_flight_other_airlines <- cut(df$percent_flight_other_airlines, c(0,4,7,9.2,10,51))
df$num_loyalty_cards <- cut(df$num_loyalty_cards, c(-1,0,1,2,3,7,11,13))
df$airport_shopping <- cut(df$airport_shopping, c(-1,0,27,30,880))
df$airport_dining <- cut(df$airport_dining, c(-1,0,31,61,69,91,896))
df$day_of_month <- cut(df$day_of_month, c(0,1,9,17,24,32))
df$scheduled_departure_hour <- cut(df$scheduled_departure_hour, c(0,1,10,14,18,23))
df$flight_time_in_minutes <- cut(df$flight_time_in_minutes, c(7,61,95,112,142,670))
df$flight_distance <- cut(df$flight_distance, c(0,32,363,630,794,1025,4984))
df$arrival_delay_in_minutes<- cut(df$arrival_delay_in_minutes,c(-1,30,60,90,120,1585))
df$departure_delay_in_minutes<- cut(df$departure_delay_in_minutes,c(-1,30,60,90,120,1593))
```

remove arrival_delay_in_minutes column

```
df <- df[,-which(colnames(df)=="arrival_delay_in_minutes")]
```

check for NA values

```
sapply(df, function(x) sum(is.na(x)))
```

change departure_delay_in_minutes to a factor of yes or no if the departure delay is greater than 15 minutes

change the name of the column to departure_delay_greater_5_mins

```
df$departure_delay_in_minutes <- ifelse(df$departure_delay_in_minutes > 11.7,"yes","no")
colnames(df)[which(colnames(df)=="departure_delay_in_minutes")] = "departure_delay_greater_5_mins"
df$departure_delay_greater_5_mins <- factor(df$departure_delay_greater_5_mins)
```

create sparse matrix from df

```
dfX <- as(df, "transactions")
```

Let's analyze flyers by the ratings groups of 1-5

After that the data set is coerced into a matrix using transactions and the coerced matrix is stored in a variable. Then apriori models are built for coerced matrix by setting the support =0.01, confidence as 0.5 and by varying the satisfaction in lhs for each rule. The rules are then sort by count for lift greater than 1 to give the variables with which satisfaction of a particular value is most dependent on.

customer satisfaction rating = 1

```
ruleset <- apriori(dfX, parameter=list(support=.01, confidence=.5, minlen=2, maxlen=6, maxtime=10),
                   appearance = list(lhs="satisfaction=(0,1]"))
```

```
ruleset_filter <- subset(ruleset, subset=lhs %ain% c("satisfaction=(0,1]") & lift>1)
summary(ruleset_filter)
```

```
ruleset_filter <- sort(ruleset_filter, decreasing=T, by="count")
inspect(ruleset_filter)
```

```
##     lhs                     rhs                                  support confidence     lift count
## [1] {satisfaction=(0,1]} => {flight_cancelled=No}             0.02266775  0.9815992 1.000065  2934
## [2] {satisfaction=(0,1]} => {airline_status=Blue}             0.01999459  0.8658414 1.263959  2588
## [3] {satisfaction=(0,1]} => {class=Economy}                   0.01921428  0.8320509 1.022516  2487
## [4] {satisfaction=(0,1]} => {num_loyalty_cards=(-1,0]}        0.01464828  0.6343259 1.197200  1896
## [5] {satisfaction=(0,1]} => {type_of_travel=Personal}         0.01428516  0.6186015 2.001167  1849
## [6] {satisfaction=(0,1]} => {gender=Female}                   0.01399930  0.6062228 1.073941  1812
## [7] {satisfaction=(0,1]} => {airport_shopping=(-1,0]}         0.01321126  0.5720977 1.006367  1710
## [8] {satisfaction=(0,1]} => {arrival_delay_greater_5_mins=yes} 0.01313401  0.5687521 1.659485  1700
```

For other customer satisfaction ratings we just need to change the LHS values to the corresponding satisfaction rating.

customer satisfaction rating = 2

```
##     lhs                     rhs                          support
## [1] {satisfaction=(1,2]} => {airline_status=Blue}      0.1617955
## [2] {satisfaction=(1,2]} => {class=Economy}            0.1497122
## [3] {satisfaction=(1,2]} => {type_of_travel=Personal}  0.1449608
## [4] {satisfaction=(1,2]} => {num_loyalty_cards=(-1,0]} 0.1172094
## [5] {satisfaction=(1,2]} => {gender=Female}            0.1125507
## [6] {satisfaction=(1,2]} => {airport_shopping=(-1,0]}  0.1116545
##     confidence lift     count
## [1] 0.8911869  1.300958 20942
## [2] 0.8246308  1.013398 19378
## [3] 0.7984595  2.583005 18763
## [4] 0.6456019  1.218482 15171
## [5] 0.6199413  1.098244 14568
## [6] 0.6150049  1.081844 14452
```

customer satisfaction rating = 3

```
##     lhs                     rhs                                   support confidence     lift count
## [1] {satisfaction=(2,3]} => {class=Economy}                     0.2328427  0.8175012 1.004636 30138
## [2] {satisfaction=(2,3]} => {departure_delay_greater_5_mins=no} 0.2257658  0.7926545 1.017900 29222
## [3] {satisfaction=(2,3]} => {airline_status=Blue}               0.2176382  0.7641187 1.115464 28170
## [4] {satisfaction=(2,3]} => {arrival_delay_greater_5_mins=no}   0.1925291  0.6759616 1.028435 24920
## [5] {satisfaction=(2,3]} => {gender=Female}                     0.1789161  0.6281669 1.112816 23158
## [6] {satisfaction=(2,3]} => {airport_shopping=(-1,0]}           0.1663151  0.5839256 1.027173 21527
## [7] {satisfaction=(2,3]} => {num_loyalty_cards=(-1,0]}          0.1547727  0.5434004 1.025591 20033
```

customer satisfaction rating = 4

```
##     lhs                     rhs                                   support confidence     lift count
## [1] {satisfaction=(3,4]} => {flight_cancelled=No}               0.4078650  0.9854218 1.003960 52792
## [2] {satisfaction=(3,4]} => {type_of_travel=Business}           0.3445899  0.8325462 1.357443 44602
## [3] {satisfaction=(3,4]} => {departure_delay_greater_5_mins=no} 0.3414996  0.8250798 1.059540 44202
## [4] {satisfaction=(3,4]} => {arrival_delay_greater_5_mins=no}   0.3054815  0.7380584 1.122912 39540
## [5] {satisfaction=(3,4]} => {price_sensitivity=(0,1]}           0.2900220  0.7007074 1.033727 37539
```

customer satisfaction rating = 5

```
##     lhs                     rhs                           support    confidence lift     count
## [1] {satisfaction=(4,5]} => {gender=Male}                 0.06416348 0.6639751  1.524571  8305
## [2] {satisfaction=(4,5]} => {type_of_travel=Business}     0.09016881 0.9330828  1.521365 11671
## [3] {satisfaction=(4,5]} => {price_sensitivity=(0,1]}     0.07535829 0.7798209  1.150440  9754
## [4] {satisfaction=(4,5]} => {flight_cancelled=No}         0.09635724 0.9971218  1.015880 12472
## [5] {satisfaction=(4,5]} => {class=Economy}               0.07901263 0.8176367  1.004802 10227
```

Summary of analysis:

The various rulesets created returned the following trends:

- There are relatively few customers that give a rating of 5, but those that do have the following attributes: male, business trip, economy class, flight not cancelled, price_sensitivity=1
- Female business travelers–while few–also give higher ratings (4-5). They expect no cancellations and low delays. No female business travelers were Blue status in any of the rules; this correlates well with the negative coefficient for Blue status in the linear model.
- Female Blue status customers, traveling for personal reasons in the economy class give the lowest ratings
- Cheapseats airlines with airline_code=WN recieves high ratings from female travelers age (34-46)

3. SUPPORT VECTOR MACHINES:

Support Vector Machines(SVM) model is used to classify the data set into two categories. In case of this data set, we have used SVM to classify the customer satisfaction into happy and unhappy customers. The SVM maps a low-dimensional problem into a higher-dimensional space with the goal of being able to describe geometric boundaries between different regions. The input data (the independent variables) from a given case are processed through a mapping algorithm called a kernel, and the resulting kernel output determines the position of that case in multidimensional space.

We first import the Cheapseat dataset and create the test and train data, after which we build the svm model.

```
ksvm_df <- read.csv("Cheapseats.csv", stringsAsFactors = F)
```

After we have created columns which represent the happy and unhappy customer based on satisfaction. For happy customers , we have the satisfaction as greater than 3.
For unhappy customers , the satisfaction is 3 or less.

```
ksvm_df$satisfactionLabel <- ifelse(ksvm_df$satisfaction>3,"happy","unhappy")
ksvm_df$satisfactionLabel <- factor(ksvm_df$satisfactionLabel)
```

We created a function to divide the dataset into training dataset and test dataset. We have randomized data and assigned 70% of dataset as training and 30% as test dataset.

```
createTestTrain <- function() {
  n = dim(svm_df)[1]
  train_index = sample(1:n, size=.7*n, replace = F)  # create random sample with size 70% of n
  train <- ksvm_df[train_index,]  # set train data to the random indices generated
  test <- ksvm_df[-train_index,] # set test data to exclude the random indices
}
```

Finally , we build the SVM model for happy customer with respect to all other columns in the dataset . We build the model using the training data we computed earlier airline_status, type_of_travel, arrival_delay_greater_5_mins, num_flights

```
svm_output <- ksvm(satisfactionLabel~airline_status + age + gender + type_of_travel + arrival_delay_greater_5_mins + num_fli
ghts, data=train, model="rfbfot", kpar="automatic", C=10, cross=3, prob.model=T)
svm_output
```

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

SV type: C-svc  (classification)
 parameter : cost C = 10

Gaussian Radial Basis kernel function.
 Hyperparameter : sigma =  0.269200859333213

Number of Support Vectors : 7896

Objective Function Value : -74386
Training error : 0.19957
Cross validation error : 0.204034
Probability model included.
```

The SVM produces an cross validation error of 20%. We can reduce that by removing the variables with less correlation to satisfaction.

Using this SVM model , we will then predict the relationship for the test data. After that , we a confusion matrix between the test data and SVM prediction to find the total error rate from SVM prediction.

```
svm_predict <- predict(svm_output, test, type="votes")
compTable <- data.frame(test[,29], svm_predict[1,])  # creates a dataframe from the from test$classify and the first row in
 the svm_predict matrix
table(compTable)
```

```
          svm_predict.1...
test...29.    0    1
   happy    254 3674
   unhappy 2520 1379
```

Calculation of total error rate:

t <-table(compTable)
error <- (t1[1,1]+t[2,2])/nrow(test)
The error is calculated by adding number of votes in (happy, 1) and (unhappy,0) in the confusion matrix and dividing it by total number of test values.

The total error rate of this SVM model is 0.208 which is equivalent to 20.8%

Now, we are creating the additional models to achieve the better error rate.

```
createTestTrain()
svm_output2 <- ksvm(satisfactionLabel~airline_status + type_of_travel + arrival_delay_greater_5_mins + num_flights, data=tra
in, model="rfbfot", kpar="automatic", C=10, cross=3, prob.model=T)
svm_output2
```

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

SV type: C-svc  (classification)
 parameter : cost C = 10

Gaussian Radial Basis kernel function.
 Hyperparameter : sigma =  0.541010239551846

Number of Support Vectors : 9000

Objective Function Value : -81463.58
Training error : 0.22371
Cross validation error : 0.225805
Probability model included.
```

Cross validation error is 0.2258 i.e 22.58%

```
svm_predict2 <- predict(svm_output2, test, type="votes")
compTable2 <- data.frame(test[,29], svm_predict2[1,])  # creates a dataframe from the from test$classify and the first row i
n the svm_predict matrix
table(compTable2)
```

```
          svm_predict2.1...
test...29.    0    1
   happy    201 3727
   unhappy 2316 1583
```

The total error rate of this SVM model is 0.2279 which is equivalent to 22.79%

From this error rate, we can say that error rate has increased. Hence we will try to create another SVM model to get better error rate.

The prediction rate does not seem to improve with removing variables, so we removed all but the strongest predictors

```
createTestTrain()
svm_output3 <- ksvm(satisfactionLabel~airline_status + type_of_travel, data=train, model="rfbfot", kpar="automatic", C=10, c
ross=3, prob.model=T)
```

```
line search fails -1.288316 -0.4215751 3.173035e-05 3.172833e-05 -1.823531e-08 -1.822239e-08 -1.156779e-12
```

```
svm_output3
```

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

SV type: C-svc  (classification)
 parameter : cost C = 10

Gaussian Radial Basis kernel function.
 Hyperparameter : sigma =  0.666666666666667

Number of Support Vectors : 8835

Objective Function Value : -88247.08
Training error : 0.243166
Cross validation error : 0.243276
Probability model included.
```

Cross validation error is 0.2432 i.e. 24.32%.

```
svm_predict3 <- predict(svm_output3, test, type="votes")
compTable3 <- data.frame(test[,29], svm_predict3[1,])  # creates a dataframe from the from test$classify and the first row i
n the svm_predict matrix
table(compTable3)
```

```
          svm_predict3.1...
test...29.    0    1
   happy    459 3469
   unhappy 2406 1493
```

Total error rate is 0.2493 i.e. 24.93%

Summary of Analysis:

From the three models we have built , we have found we need airline_status, type_of_travel, arrival_delay_greater_5_mins, num_flights together in a model to provide best accuracy rate . This proves that these 5 attributes are the best indicators for satisfaction rating of the customers.

OVERALL INTERPRETATION OF RESULTS

1. The attributes that are contributing to the satisfaction to index are airline status, age, gender, type of travel, number of flights and arrival delay.
2. Male passengers that travel from the same class in airline that has same status and travel to same location give higher ratings than female passenger.
3. Male travellers give better ratings by 13% than female travellers. Cheapseat airlines should target female customers that travel by economy for personal travel.
4. Loyalty cards are not a good linear predictor of satisfaction, and seldom occur with association pairings for highly satisfied customers; However, female dissatisfied travelers commonly have 0 loyalty cards
5. Middle aged travellers tend to give higher satisfaction ratings. Efforts should be made to provide the better services to age group young customers in the below age 20 and above the age of 80.
6. Cities where Cheapseats is doing good are:
a. Gunnison, CO; Oklahoma City, OK; Billings, MT; Rochester, NY; Omaha, NE; Grand Rapids, MI; Montrose, CO; Hayden, CO; Albuquerque, NM; Columbus, OH
7. Cities where Cheapseats needs to improve are:
a. Less: St. Louis, MO; Wichita, KS; Albany, NY; Ontario, CA; Hartford, CT; Atlanta, GA, Pittsburgh, PA
b. Four of the largest markets: California, Nevada, Arizona, and Texas aren't in Cheapseats' top 10 satisfaction list. Airline should focus on these markets for better business as they are losing on bigger profit markets.
8. Silver status customers have (.63) higher satisfaction scores
9. Blue status customers have (-.63) lower satisfaction scores

Miscellaneous Observations

Surprisingly loyalty cards don't seem to influence higher satisfaction ratings. Loyalty cards were hardly present in pairings, and when they were, it was for 0 cards.

As an example: the top 3 of the 32 rules returned for female business travelers, age 46-58, were for 0 loyalty cards.

ACTIONABLE INSIGHTS

1. From our findings, we found that female passengers are also more into convenience and comfortability during their travel. We can improve this by increasing the leg space in flights seats which may increase in satisfaction . Also , there are bills in congress pending regarding airline leg room. We can be ahead of the curve and make changes which will us attract more customers especially female customers
2. We also found that female passengers provide low satisfaction because they are very particular about delay of flights. In case of moderate delay , we can provide customers with some additional services like meal coupons, future travel discount coupon etc.
3. We can provide better escort services for senior citizens who are travelling which may increase their satisfaction . We can provide them seats either which are close to restroom or the special seats which have additional leg space than the normal seats based on their requirement and convenience.
4. For younger passengers, we can improve the entertainment features in the airlines like games and movies in the flight we may enrich their flight experience . Providing better Wifi in the airport will improve their satisfaction
5. We found that the satisfaction of four of the largest markets are not high . This may be due to the presence of frequent flyers in those regions . We can improve their satisfaction by improving provision of loyalty points in loyalty cards. For example, a passenger has certain number of loyalty points they should be free access to airline lounge for a day.
6. We can also improve the satisfaction of frequent flyers by providing them upgrade of class of travel after every 50,000 miles(ie) they can travel in business class after acquiring 50, 000 miles of travel
7. We need to improve the features of blue status customers and provide them with opportunities to upgrade to silver status

CONCLUSION

In this report, we have found the reasons for low and high satisfaction of customers who travelled using Cheapseat airlines. We have answered the various business questions that were presented to improve the customer satisfaction . We have validated our findings using the linear model , association rules and support vector machines . We have further provided the actionable insights to improve the flying experience of the less satisfied customers.

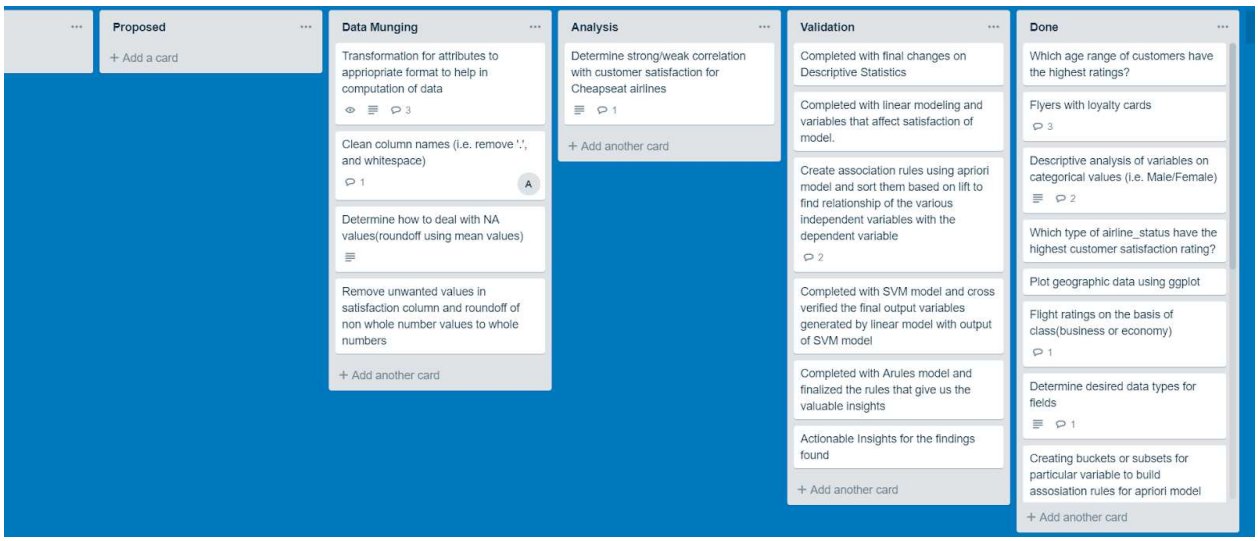
APPENDIX:

Trello

We have six columns in our trello board To Do, Proposed, Data Munging , Analysis, Validation and Done. The Proposed column provides information on the the brainstormed idea provided by the team members before they were agreed by all other member and approved by the team leader. Since we have completed the project there are no entries in the proposed column.

The To Do column provides information on tasks to be completed in the upcoming weeks. The Data Munging Column provides details on cleaning done in data so as to help in the performing the further tasks in the project. The analysis column provides information on the main analysis performed in the project

The Validation column provides information on the models , statistics found that prove the analysis performed in the project . It also gives information on insights gained from the project . The Done column provides information on the completed tasks including the business question and various smaller tasks during data cleaning and creation of models.



Code:
We have used 5 r scripts for this project

1. **Data Cleaning.**

Clean airline data

TODO

- None

import data and change column names

```
df = read.csv("satisfaction_survey.csv", stringsAsFactors=F)

new_colnames <- c('satisfaction', 'airline_status', 'age', 'gender', 'price_sensitivity', 'year_first_flight', 'num_flights', 'percent_flight_other_airlines', 'type_of_travel', 'num_loyalty_cards', 'airport_shopping', 'airport_dining', 'class', 'day_of_month', 'flight_date', 'airline_code', 'airline_name', 'origin_city', 'origin_state', 'destination_city', 'destination_state', 'scheduled_departure_hour', 'departure_delay_in_minutes', 'arrival_delay_in_minutes', 'flight_cancelled', 'flight_time_in_minutes', 'flight_distance', 'arrival_delay_greater_5_mins')

colnames(df) <- new_colnames
```

Beginning row length is: 129889, and number of columns is: 28

Check for leading and trailing whitespace for every vector in df

```
test_blanks <- c(paste(" ", 1:129889, " "))  # create an array of characters with spaces at the front and end
df <- cbind(df, test_blanks)  # add test array to df
sapply(df, function(x) sum(grepl("^\\s*.*\\s+$", x)))  # find all vectors in df with leading/trailing spaces
```

```
##                  satisfaction                airline_status
##                             0                             0
##                           age                        gender
##                             0                             0
##             price_sensitivity             year_first_flight
##                             0                             0
##                   num_flights percent_flight_other_airlines
##                             0                             0
##                type_of_travel             num_loyalty_cards
##                             0                             0
##              airport_shopping                airport_dining
##                             0                             0
##                         class                  day_of_month
##                             0                             0
##                   flight_date                  airline_code
##                             0                             0
##                  airline_name                   origin_city
##                        118481                             0
##                  origin_state              destination_city
##                             0                             0
##             destination_state      scheduled_departure_hour
##                             0                             0
##    departure_delay_in_minutes      arrival_delay_in_minutes
##                             0                             0
##              flight_cancelled        flight_time_in_minutes
##                             0                             0
##               flight_distance  arrival_delay_greater_5_mins
##                             0                             0
##                   test_blanks
##                        129889
```

- Looks like our test case and airline_name has leading/trailing spaces; let's deal with the whitespace in *airline_name*

```
df$airline_name = trimws(df$airline_name)
colsKeep <- c('satisfaction', 'airline_status', 'age', 'gender', 'price_sensitivity', 'year_first_flight', 'num_flights', 'percent_flight_other_airlines', 'type_of_travel', 'num_loyalty_cards', 'airport_shopping', 'airport_dining', 'class', 'day_of_month', 'flight_date', 'airline_code', 'airline_name', 'origin_city', 'origin_state', 'destination_city', 'destination_state', 'scheduled_departure_hour', 'departure_delay_in_minutes', 'arrival_delay_in_minutes', 'flight_cancelled', 'flight_time_in_minutes', 'flight_distance', 'arrival_delay_greater_5_mins')
df <- df[,colsKeep]
```

- whitespace in *airline_name* is gone, run the sapply() command from above to verify

Look for anomalies by reviewing the unique values of the variables

- you're looking for anomalies–specifically in the levels

```
unique(df$satisfaction)
```

```
##  [1] "4.5"       "4"         "2.5"       "5"         "3.5"
##  [6] "2"         "3"         "1"         "4.00.5"    "4.00.2.00"
```

Satisfaction column has strange values, *"4.00.5"* and *"4.00.2.00"*. We can find how many rows have these values with:

```
which(df$satisfaction=="4.00.5")
```

```
## [1] 38898
```

```
which(df$satisfaction=="4.00.2.00")
```

```
## [1] 38899 38900
```

Considering there are only 3 rows with the eroneous data, let's delete them

```
library(magrittr)
drop_rows <- c(38898, 38899, 38900)
df <- df[-drop_rows,]
df$satisfaction %<>% as.numeric  # covert vector to numeric
```

- The dataset now has 129886 rows after dropping the 3 with bad data

Find all NA values in df

```
sapply(df, function(x) sum(is.na(x)))
```

```
##                  satisfaction                airline_status
##                             0                             0
##                           age                        gender
##                             0                             0
##             price_sensitivity             year_first_flight
##                             0                             0
##                   num_flights percent_flight_other_airlines
##                             0                             0
##                type_of_travel             num_loyalty_cards
##                             0                             0
##              airport_shopping                airport_dining
##                             0                             0
##                         class                  day_of_month
##                             0                             0
##                   flight_date                  airline_code
##                             0                             0
##                  airline_name                   origin_city
##                             0                             0
##                  origin_state              destination_city
##                             0                             0
##             destination_state      scheduled_departure_hour
##                             0                             0
##    departure_delay_in_minutes      arrival_delay_in_minutes
##                          2345                          2738
##              flight_cancelled        flight_time_in_minutes
##                             0                          2738
##               flight_distance  arrival_delay_greater_5_mins
##                             0                             0
```

- Flight_time_in_minutes, departure_delay_in_minutes, and arrival_delay_in_minutes have NA values. How should we deal with them? Our options are:
    - Drop the entire row
    - Set them to the average of the vector

Set NA values to average of column

```
flight_time_in_minutes_avg <- mean(df$flight_time_in_minutes, na.rm=T)
departure_delay_in_minutes_avg <- mean(df$departure_delay_in_minutes, na.rm=T)
arrival_delay_in_minutes_avg <- mean(df$arrival_delay_in_minutes, na.rm=T)

df[which(is.na(df$flight_time_in_minutes)),"flight_time_in_minutes"] = round(flight_time_in_minutes_avg,1)
df[which(is.na(df$departure_delay_in_minutes)),"departure_delay_in_minutes"] = round(departure_delay_in_minutes_av
g,1)
df[which(is.na(df$arrival_delay_in_minutes)),"arrival_delay_in_minutes"] = round(arrival_delay_in_minutes_avg,1)
```

Change the *flight_date* column to Date datatype.

```
df$flight_date <- as.Date(df$flight_date, "%m/%d/%y")
```

- date if formated: YYYY-mm-dd (i.e. 2014-03-18)

The arrangment of the categories does not follow an ordinal hierachy such as lowest to highest (i.e. Eco, Eco Plus, Business). We will make the following changes:

1. Change the values for the *type_of_travel* variable to the more concise names of: "Personal, Mileage, Business"
    - Convert the *type_of_travel* variable to a factor with factor order of: "Personal, Mileage, Business"
2. Change the values for the *class* variable to the more concise names of: "Economy, Plus, Business"
    - Set the order of the factors for the *class* variable to: "Economy, Plus, Business"
3. Change *gender* to a factor and change the factor order to: "Male, Female"
4. Change *airline_status* to an "unordered" factor: "Blue, Silver, Gold, Platinum"
5. Change *airline_name* to more consise names of: "'Cheapseats', 'Cool&Young', 'EnjoyFlying', 'FlyFast', 'FlyHere', 'FlyToSun', 'GoingNorth', 'Northwest', 'OnlyJets', 'Oursin', 'PaulSmith', 'Sigma', 'Southeast', 'West'"
6. Change *flight_cancelled* to a factor with order of: "Yes, No"
7. Change *arrival_delay_greater_5_mins* to a factor with order: "yes", "no"

```
# 1. Change the values for the *class* variable to the more concise names of: "Economy, Plus, Business"
df[which(df$type_of_travel=="Personal Travel"), "type_of_travel"] = "Personal"
df[which(df$type_of_travel=="Mileage tickets"), "type_of_travel"] = "Mileage"
df[which(df$type_of_travel=="Business travel"), "type_of_travel"] = "Business"
# change to factor and set order
df$type_of_travel <- factor(df$type_of_travel, levels=c("Personal","Mileage","Business"), ordered=TRUE)
```

```
# 2. Change the values for the *class* variable to the more concise names of: "Economy, Plus, Business"
df[which(df$class=="Eco"), "class"] = "Economy"
df[which(df$class=="Eco Plus"), "class"] = "Plus"
df$class <- factor(df$class, levels=c("Economy", "Plus", "Business"), ordered=TRUE)
```

```
# 3. Change *gender* to a factor and change the factor order to: "Male, Female"
df$gender <- factor(df$gender, levels=c("Male", "Female"), ordered=TRUE)
```

```
# 4. Change *airline_status* to a factor with order of: "Blue, Silver, Gold, Platinum"
df$airline_status <- factor(df$airline_status, levels=c("Blue", "Silver", "Gold", "Platinum"), ordered=TRUE)
```

5. Change *airline_name* to more concise names of: "'Cheapseats', 'Cool&Young', 'EnjoyFlying', 'FlyFast', 'FlyHere', 'FlyToSun', 'GoingNorth', 'Northwest', 'OnlyJets', 'Oursin', 'PaulSmith', 'Sigma', 'Southeast', 'West'"

```
df[which(df$airline_name=="Cheapseats Airlines Inc."), "airline_name"] = "Cheapseats"
df[which(df$airline_name=="Cool&Young Airlines Inc."), "airline_name"] = "Cool&Young"
df[which(df$airline_name=="EnjoyFlying Air Services"), "airline_name"] = "EnjoyFlying"
df[which(df$airline_name=="FlyFast Airways Inc."), "airline_name"] = "FlyFast"
df[which(df$airline_name=="FlyHere Airways"), "airline_name"] = "FlyHere"
df[which(df$airline_name=="FlyToSun Airlines Inc."), "airline_name"] = "FlyToSun"
df[which(df$airline_name=="GoingNorth Airlines Inc."), "airline_name"] = "GoingNorth"
df[which(df$airline_name=="Northwest Business Airlines Inc."), "airline_name"] = "Northwest"
df[which(df$airline_name=="OnlyJets Airlines Inc."), "airline_name"] = "OnlyJets"
df[which(df$airline_name=="Oursin Airlines Inc."), "airline_name"] = "Oursin"
df[which(df$airline_name=="Paul Smith Airlines Inc."), "airline_name"] = "PaulSmith"
df[which(df$airline_name=="Sigma Airlines Inc."), "airline_name"] = "Sigma"
df[which(df$airline_name=="Southeast Airlines Co."), "airline_name"] = "Southeast"
df[which(df$airline_name=="West Airways Inc."), "airline_name"] = "West"

new_airline_names <- c('Cheapseats', 'Cool&Young', 'EnjoyFlying', 'FlyFast', 'FlyHere', 'FlyToSun', 'GoingNorth',
'Northwest', 'OnlyJets', 'Oursin', 'PaulSmith', 'Sigma', 'Southeast', 'West')

df$airline_name <- factor(df$airline_name, levels=new_airline_names, ordered=TRUE)
```

6. Change *flight_cancelled* to a factor with order of: "Yes, No"

```
df$flight_cancelled <- factor(df$flight_cancelled, levels=c("Yes","No"), ordered=TRUE)
```

7. Change *arrival_delay_greater_5_mins* to a factor with order: "yes", "no"

```
df$arrival_delay_greater_5_mins <- factor(df$arrival_delay_greater_5_mins, levels=c("yes","no"), ordered=TRUE)
```

Drop the rows with percent_flight_other_airlines > 50

```
df <-  df[-which(df$percent_flight_other_airlines > 50),]
```

- dropped 451 of outliers data

Round up the values for satisfaction with decimal values

```
table(df$satisfaction)
```

```
##
##     1     2   2.5     3   3.5     4   4.5     5
##  2989 23499     2 36864     2 53571     2 12506
```

```
df[which(df$satisfaction==2.5), "satisfaction"] = 3
df[which(df$satisfaction==3.5), "satisfaction"] = 4
df[which(df$satisfaction==4.5), "satisfaction"] = 5
table(df$satisfaction)
```

```
##
##     1     2     3     4     5
##  2989 23499 36866 53573 12508
```

Ending row length is: 129435, and number of columns is: 28

Write csv to file

```
write.csv(df, "flight_survey_updated.csv", row.names=F)
```

2. **Summary Statistics.**

summary_stats

Import cleaned dataset

```
df <- read.csv("flight_survey_updated.csv", stringsAsFactors=T)
df$satisfaction <- factor(df$satisfaction)
df$year_first_flight <- factor(df$year_first_flight)
df$num_loyalty_cards <- factor(df$num_loyalty_cards)
```

create two data frame (industry and cheapseats)

```
which(df$airline_name=="Cheapseats") %>% length()
```

```
## [1] 25971
```

```
industry <- df[-which(df$airline_name=="Cheapseats"),]
cheapseatsDf <- df[which(df$airline_name=="Cheapseats"),]
```

Summary of anlayis

- Higher ratings are centered around 40 year old customers
- Older customers (80 and above) and younger customers (20 and below) give lower ratings
- Cheapseats is assumed to be the busiest airline (most flights); they have the most observations in the dataset
- Cool&Young is assumed to be the least busiest airline (least flights); they have the least amount of observations in the dataset
- departure_delay_in_minutes and arrival_delay_in_minutes are heavily right skewed; their outliers should be handled before building the linear model. recommend using the winsor method to set the upper limit to a chose percentile (i.e. 95%)
- most flyers do not have a loyalty card, and most that do have between 1-5.

Revised business questions

- There are 25161 with only 1 loyalty card. Are they more/less satisfied with those with 0 or greater than 1?
- There are 0 with greater than 1 loyalty card. Are they more/less satisfied?
- Who has higher ratings, males or females?
- Why do customers centered around the age of 40 have higher ratings?
- Why do older customers (80 and above) and younger customers (20 and below) give lower ratings?

TODO

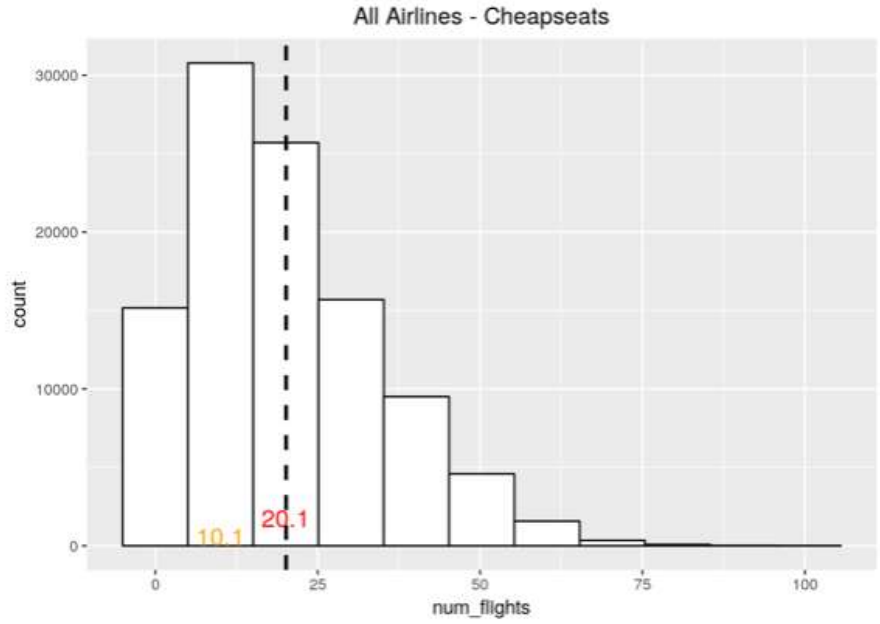
- Pivot tables or dplyr summary statistics

Plost histograms using createHist()

```
library(magrittr)
getBinWidth <- function(v) {
  my_min <- min(v)
  my_max <- max(v)
  l <- length(unique(v))
  return((my_max - my_min)/sqrt(l))
}

createIntHist <- function(df,cols,airlineName) {
  for(c in cols) {
    myVector <- df[[c]]  # df must be in local environment
    bins_binwidth <- getBinWidth(myVector)  # call getBinWidth() to get binwidth argument value
    g <- ggplot(df) +  # create ggplot instance
      geom_histogram(aes(x=myVector), binwidth=bins_binwidth, color="black", fill="white") +
      labs(title=airlineName, x=c) +
      theme(plot.title = element_text(hjust=0.5))

    # from github: https://stackoverflow.com/questions/47000494/how-to-add-mean-and-mode-to-ggplot-histogram
    data<- g %+% ggplot_build(g)$data
    hist_peak<-data[[1]]%>%filter(y==max(y))%>%.$x

    g <- g + geom_vline(aes(xintercept=mean(myVector)),col="black", lwd=1, linetype=2)
    g <- g + annotate("text", label=round(hist_peak,1), x=hist_peak, y=0,vjust="bottom",col='orange',size=5)
    g <- g + annotate("text", label=round(mean(myVector),1), x=round(mean(myVector),1), y=0,vjust=-1,col='red',siz
e=5)
    print(g)
  }
}
```

plot all continuous variables for df

```
createIntHist(industry,c("num_flights","percent_flight_other_airlines","departure_delay_in_minutes","arrival_delay
_in_minutes","flight_time_in_minutes","flight_distance","airport_shopping","airport_dining"),"All Airlines - Cheap
seats")
```



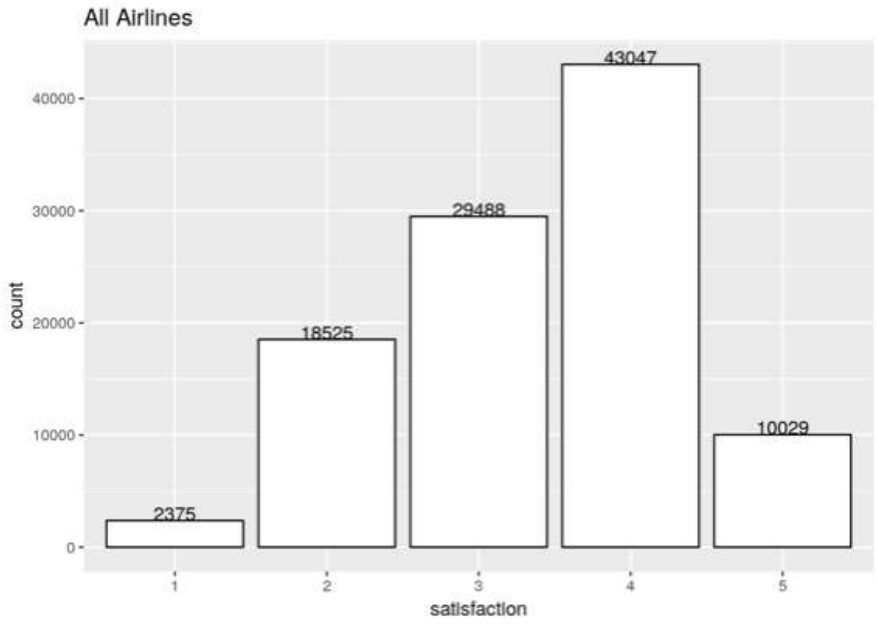
… <charts omitted due to length>

Create barplots for factor datatypes

```
createFactorBarPlot <- function(df,cols,airlineName) {
  for(col in cols) {
    myVector <- df[[col]]
    if(col=="airline_name") {
      g <- ggplot(df, aes(x=myVector)) +
        geom_bar(color="black",fill="white") +
        geom_text(stat="count", aes(x=myVector, label=..count..), vjust=0) +
        labs(title=col, x=col) +
        theme(axis.text.x = element_text(angle=90, vjust=0))
      print(g)
    } else {
      g <- ggplot(df, aes(x=myVector)) +
        geom_bar(color="black",fill="white") +
        geom_text(stat="count", aes(x=as.numeric(myVector), label=..count..), vjust=0) +
        labs(title=airlineName, x=col)
      print(g)
    }
  }
  #return(data)
}
```

create barplots for discrete variables

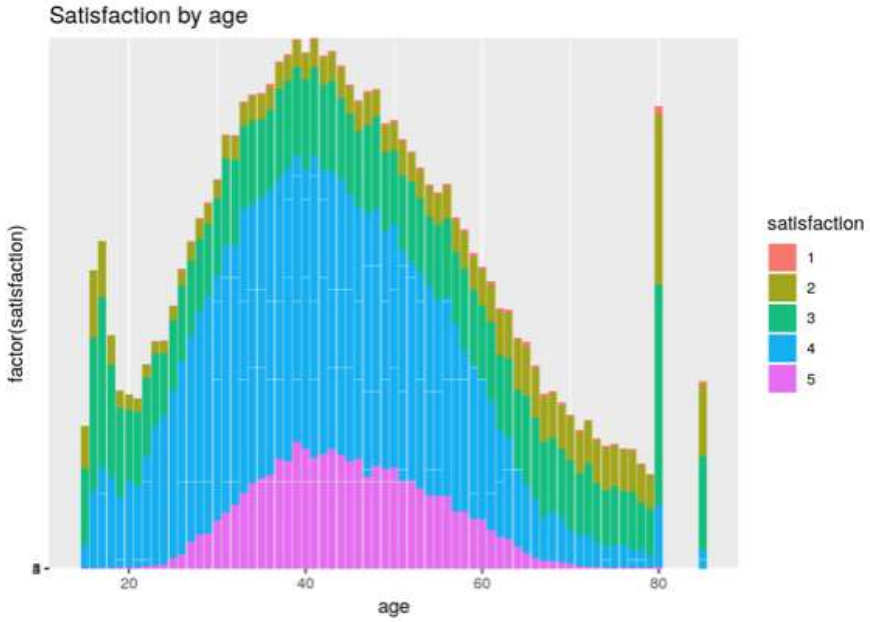
```
createFactorBarPlot(industry,c("satisfaction","gender","type_of_travel","class","airline_name","flight_cancelled",
"year_first_flight","num_loyalty_cards","airline_code"),"All Airlines")
```



… <charts omitted due to length>

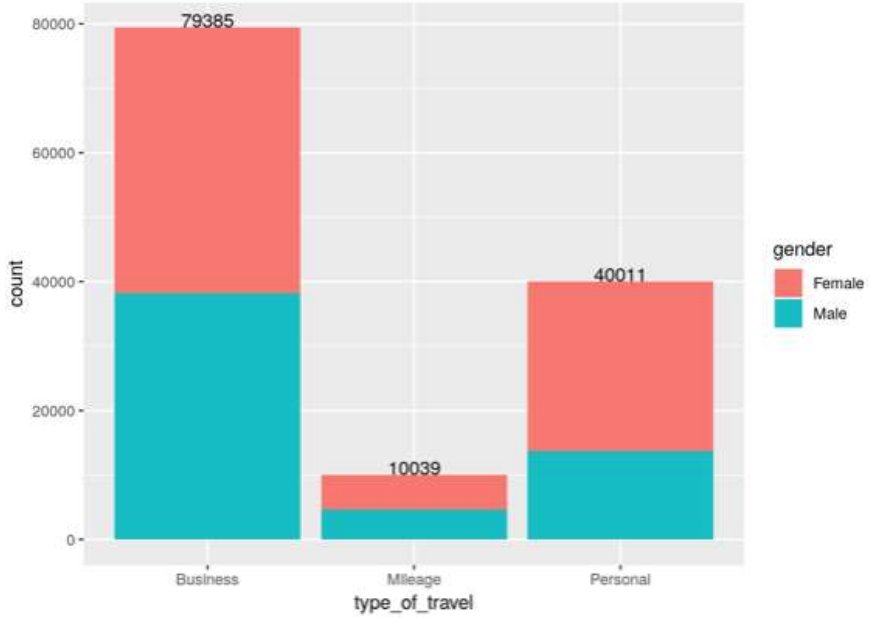
Analyze satisfaction by age

```
#df$satisfaction <- factor(df$satisfaction, levels=c(1,2,3,4,5), ordered = T)
g <- ggplot(industry, aes(x=age,y=factor(satisfaction))) +
  geom_col(aes(fill=satisfaction)) +
  labs(title="Satisfaction by age")
g
```



- looks like the distribution for satisfied customers is centered around 40 year olds

```
g <- ggplot(df, aes(x=type_of_travel)) +
  geom_bar(aes(fill=gender)) +
  geom_text(stat="count", aes(x=type_of_travel, label=..count..), vjust=0)
g
```

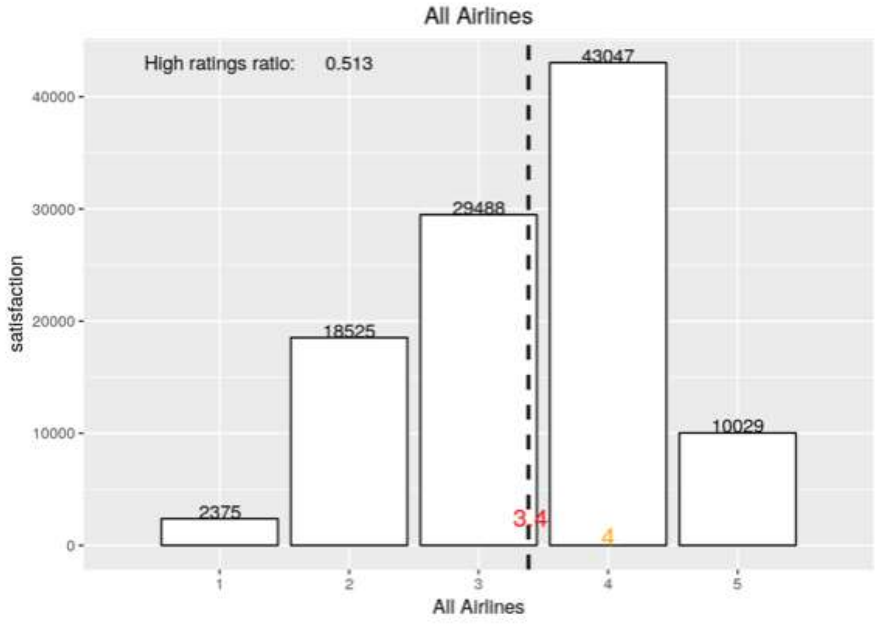


creat histogram of satisfaction by airline

```
createSatHist <- function(airlineDf, airlineName) {
  airlineDf$satisfaction <- airlineDf$satisfaction %>% as.numeric()
  g <- ggplot(airlineDf) +  # create ggplot instance
    geom_bar(aes(x=satisfaction), color="black", fill="white") +
    geom_text(stat="count", aes(x=satisfaction, label=..count..), vjust=0) +
    scale_x_discrete(limits=c(1,2,3,4,5)) +
    labs(title=airlineName, x=airlineName, y="satisfaction") +
    theme(plot.title = element_text(hjust=0.5))

    data <- g %+% ggplot_build(g)$data
    hist_peak<-data[[1]]%>%filter(y==max(y))%>%.$x

    g <- g + geom_vline(aes(xintercept=mean(airlineDf$satisfaction)),col="black", lwd=1, linetype=2)
    g <- g + annotate("text", label=round(hist_peak,1), x=hist_peak,y=0,vjust="bottom",col='orange',size=5)
    g <- g + annotate("text", label=round(mean(airlineDf$satisfaction),1), x=round(mean(airlineDf$satisfaction),1)
, y=0,vjust=-1,col='red',size=5)
    # get the ratio of 4&5 ratings/all ratings
    good_ratings_ratio <- round(length(which(airlineDf$satisfaction >= 4))/nrow(airlineDf),4)
    #cat(sprintf("%s: %.4f\n", airlineName, good_ratings_ratio))
    g <- g + annotate("text", label="High ratings ratio:", x=min(airlineDf$satisfaction), y=length(which(airlineDf
$satisfaction==4)))
    g <- g + annotate("text", label=good_ratings_ratio, x=min(airlineDf$satisfaction)+1, y=length(which(airlineDf$
satisfaction==4)))
  print(g)
}
```

get satisfaction barplot for all airlines

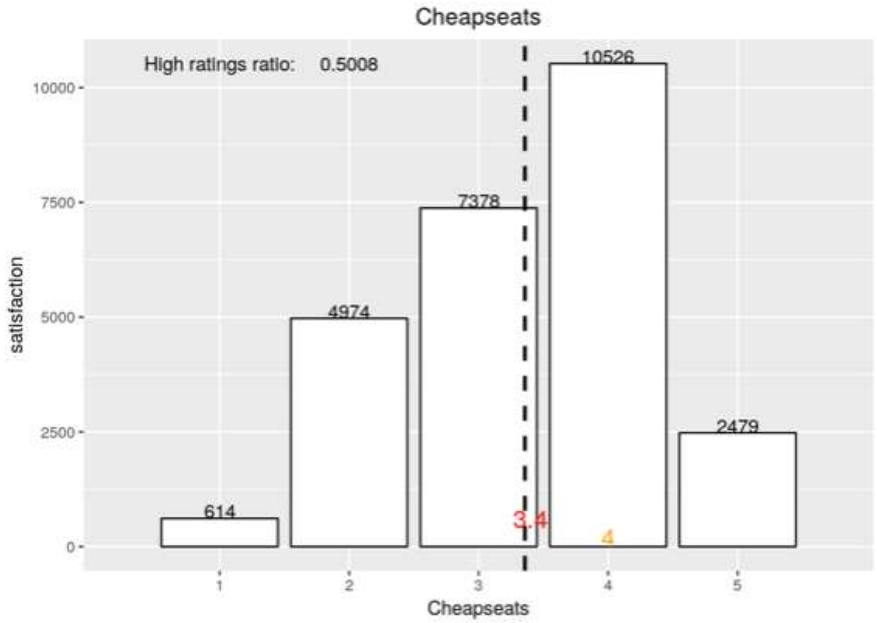
```
createSatHist(industry, "All Airlines")
```

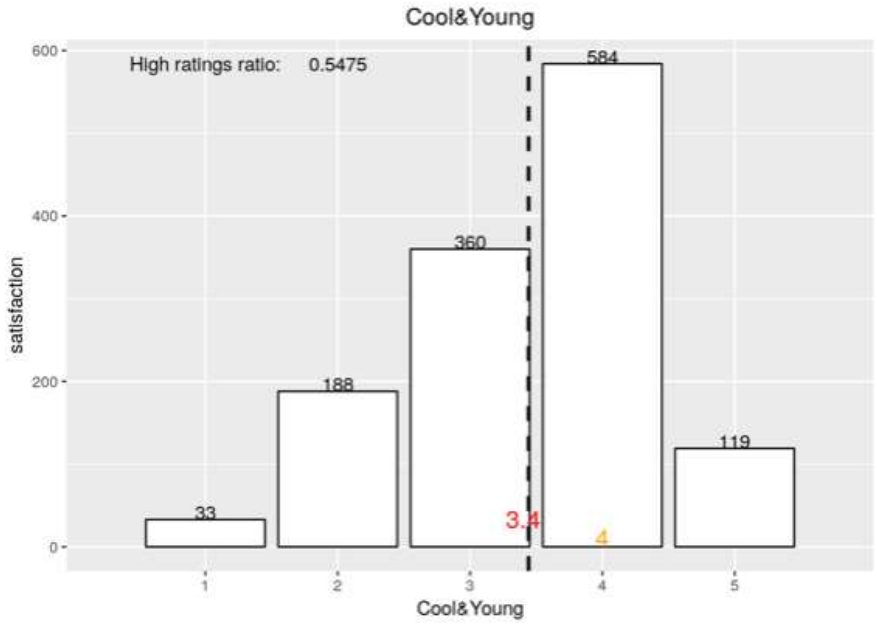


create histogram for all airline ratings

- The dashed line with red number is the mean satisfaction rating and the orange number is the mode (most frequent rating)

```
airlineNames <- c('Cheapseats', 'Cool&Young', 'EnjoyFlying', 'FlyFast', 'FlyHere', 'FlyToSun', 'GoingNorth', 'Nort
hwest', 'OnlyJets', 'Oursin', 'PaulSmith', 'Sigma', 'Southeast', 'West')
for(airline in airlineNames) {
  airlineDf <- df[which(df$airline_name==airline),]
  airlineDf$satisfaction <- as.numeric(airlineDf$satisfaction)
  createSatHist(airlineDf, airline)
}
```

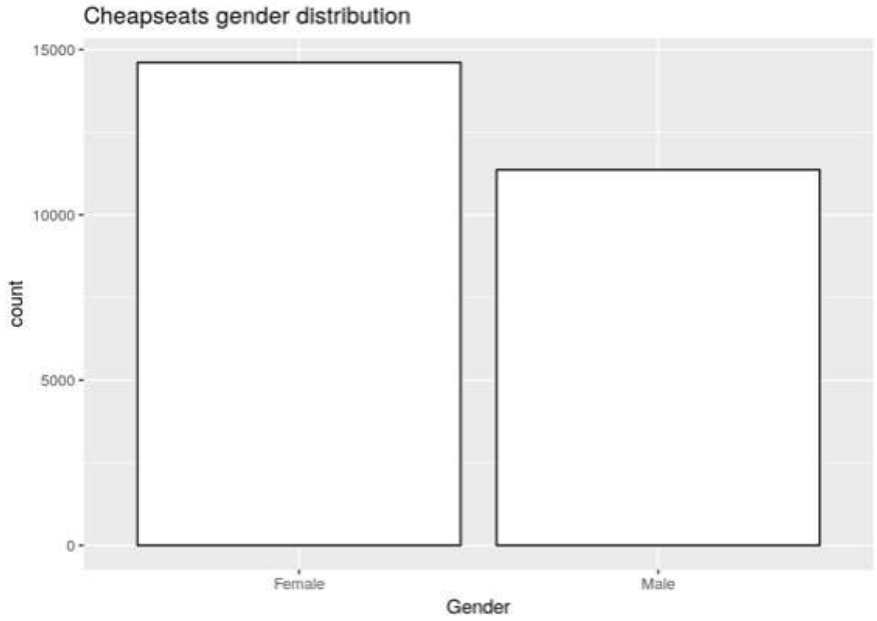




... <output truncated>

get male/female customer count for CheapAirlines

```
cheap_male_female <- df %>% dplyr::select(airline_name,gender) %>% dplyr::filter(airline_name=="Cheapseats") %>% d
plyr::group_by(gender) %>% dplyr::summarise(count=n())
g <- ggplot(cheap_male_female, aes(x=gender,y=count)) +
  geom_bar(stat="identity", color="black",fill="white") +
  labs(title="Cheapseats gender distribution", x="Gender")
g
```



- The dashed line with red number is the mean satisfaction rating and the orange number is the mode (most frequent rating)
- GoingNorth and OnlyJets have the lowest high ratings ratio at 48% and 49% respectively
- West and Cool&Young have the highest high ratings ratio at 57% and 55% respectively, but they are are the smallest airlines

Business quesitons

- Who has higher ratings, males or females?
- Why do customers centered around the age of 40 have higher ratings?
- Why do older customers (80 and above) and younger customers (20 and below) give lower ratings?

how many female, blue, economy, personal travelers are there?

```
df %>% dplyr::select(satisfaction,airline_name,gender,airline_status,type_of_travel) %>%
  dplyr::filter(gender=="Female",airline_name=="Cheapseats",airline_status=="Blue",type_of_travel=="Personal") %>%
  dplyr::group_by(satisfaction) %>%
  dplyr::summarise(count=n()) %>%
  dplyr::arrange(satisfaction)
```

```
## # A tibble: 5 x 2
##   satisfaction count
##   <fct>        <int>
## 1 1              256
## 2 2             2434
## 3 3             1385
## 4 4               83
## 5 5                4
```

how many male, blue, economy, personal travelers are there?

```
df %>% dplyr::select(satisfaction,airline_name,gender,airline_status,type_of_travel) %>%
  dplyr::filter(gender=="Male",airline_name=="Cheapseats",airline_status=="Blue",type_of_travel=="Personal") %>%
  dplyr::group_by(satisfaction) %>%
  dplyr::summarise(count=n()) %>%
  dplyr::arrange(satisfaction)
```

```
## # A tibble: 5 x 2
##   satisfaction count
##   <fct>        <int>
## 1 1               87
## 2 2             1201
## 3 3              667
## 4 4              101
## 5 5               10
```

how many blue, cheapseats, personal travelers are there in total

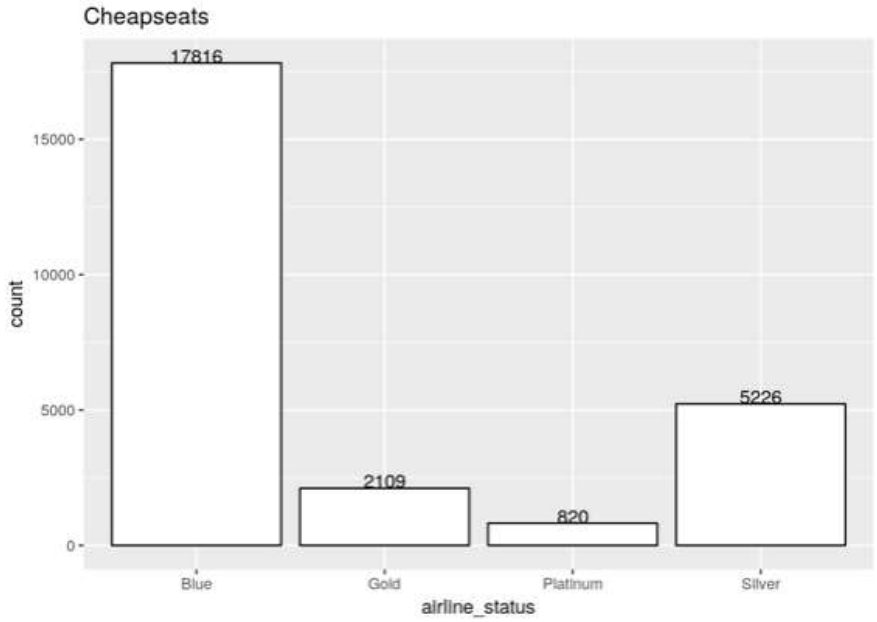
```
df %>% dplyr::select(satisfaction,airline_name,gender,airline_status,type_of_travel) %>%
  dplyr::filter(airline_name=="Cheapseats",airline_status=="Blue",type_of_travel=="Personal") %>%
  dplyr::group_by(satisfaction) %>%
  dplyr::summarise(count=n()) %>%
  dplyr::arrange(satisfaction)
```

```
## # A tibble: 5 x 2
##   satisfaction count
##   <fct>        <int>
## 1 1              343
## 2 2             3635
## 3 3             2052
## 4 4              184
## 5 5               14
```

run summary statistics for Cheapseats

- factor variables

```
# cheapseatsDf <- df[which(df$airline_name=="Cheapseats"),]
createFactorBarPlot(cheapseatsDf,c("airline_status","satisfaction","gender","type_of_travel","class","flight_cancelled","year_first_flight","num_loyalty_cards","airline_code"),"Cheapseats")
```



<...output truncated>

3. **Linear Modeling.**

Cheapseats_linear_modeling.Rmd

TODO:

- Winsorize departure_delay_in_minutes and arrival_delay_in_minutes to handle outliers (done)
- Create scatter plots for variables, w/ abline
- Get correlation coefficints for all variables

import data from cleaning step

```
df <- read.csv("Cheapseats.csv", stringsAsFactors = T)
```

summary of analysis

- The independent variables: airline_status + age + gender + type_of_travel + arrival_delay_greater_5_mins + num_flights product a linear model that can explain 44% of the variance around the mean of the dependent variable "satisfaction""
- "Silver" status customers have the highest satisfaction ratings
- "Business" travelors have the highest satisfaction ratings
- "Males" have higher ratings than females
- Customers with delays less than five minutes have higher satisfaction ratings
- *All interpretations have the condition of keeping all other independent variables constant*

Refined business quesitons

- Why are Silver status customers more satisfied than the higher Platinum and Gold counterparts?
- Why are Males more satisfied than females?

Winsorize departure_delay_in_minutes and arrival_delay_in_minutes

- see how many rows fall above the 95th percentile

```
departure_delay_95 <- quantile(df$departure_delay_in_minutes, .95)
sprintf("%d rows will be changed to the 95th percentile of departure_delay_in_minutes", length(which(df$departure_
delay_in_minutes > departure_delay_95)))
```

```
## [1] "1290 rows will be changed to the 95th percentile of departure_delay_in_minutes"
```

```
df[which(df$departure_delay_in_minutes > departure_delay_95), "departure_delay_in_minutes"] = departure_delay_95

arrival_delay_95 <- quantile(df$arrival_delay_in_minutes, .95)
sprintf("%d rows will be changed to the 95th percentile of arrival_delay_in_minutes", length(which(df$arrival_dela
y_in_minutes > arrival_delay_95)))
```

```
## [1] "1289 rows will be changed to the 95th percentile of arrival_delay_in_minutes"
```

```
df[which(df$arrival_delay_in_minutes > arrival_delay_95),"arrival_delay_in_minutes"] = arrival_delay_95
```

get the r-squared values for all the variables as predictors of satisfaction

```
createLM <- function(mydf) {
  r_squares <- list()
  for(i in 2:ncol(mydf)) {
    model <- lm(formula=satisfaction~mydf[[i]], data=mydf)
    r_squares[[i]] <- summary(model)
  }

  model_names <- c('satisfaction','airline_status', 'age', 'gender', 'price_sensitivity', 'year_first_flight','num
_flights', 'percent_flight_other_airlines', 'type_of_travel','num_loyalty_cards', 'airport_shopping', 'airport_din
ing','class', 'day_of_month', 'flight_date','origin_city', 'origin_state', 'destination_city','destination_state',
'scheduled_departure_hour', 'departure_delay_in_minutes', 'arrival_delay_in_minutes', 'flight_cancelled','flight_t
ime_in_minutes', 'flight_distance', 'arrival_delay_greater_5_mins')

  names(r_squares) <- model_names  # name lists

  return(r_squares)
}

df <- df[,-which(colnames(df)=="airline_code")]
df <- df[,-which(colnames(df)=="airline_name")]
r_squares <- createLM(df)
for(i in 2:length(r_squares)) {
  v_name <- names(r_squares)[i]
  print(sprintf("The r-squared value for %s: %f", v_name, r_squares[[i]]$r.squared))
}
```

```
## [1] "The r-squared value for airline_status: 0.123284"
## [1] "The r-squared value for age: 0.049398"
## [1] "The r-squared value for gender: 0.018651"
## [1] "The r-squared value for price_sensitivity: 0.009431"
## [1] "The r-squared value for year_first_flight: 0.000059"
## [1] "The r-squared value for num_flights: 0.055542"
## [1] "The r-squared value for percent_flight_other_airlines: 0.004124"
## [1] "The r-squared value for type_of_travel: 0.335375"
## [1] "The r-squared value for num_loyalty_cards: 0.007634"
## [1] "The r-squared value for airport_shopping: 0.000169"
## [1] "The r-squared value for airport_dining: 0.000049"
## [1] "The r-squared value for class: 0.002290"
## [1] "The r-squared value for day_of_month: 0.000017"
## [1] "The r-squared value for flight_date: 0.003585"
## [1] "The r-squared value for origin_city: 0.002952"
## [1] "The r-squared value for origin_state: 0.001570"
## [1] "The r-squared value for destination_city: 0.003460"
## [1] "The r-squared value for destination_state: 0.001681"
## [1] "The r-squared value for scheduled_departure_hour: 0.000588"
## [1] "The r-squared value for departure_delay_in_minutes: 0.009210"
## [1] "The r-squared value for arrival_delay_in_minutes: 0.011927"
## [1] "The r-squared value for flight_cancelled: 0.000671"
## [1] "The r-squared value for flight_time_in_minutes: 0.000013"
## [1] "The r-squared value for flight_distance: 0.000005"
## [1] "The r-squared value for arrival_delay_greater_5_mins: 0.026848"
```

- winsorizing the departure/arrival delays changed the coefficients from *.005 and .007* to *.010 and .014* respectively

perform stepwise regression (backwards) to find best predictors

```
library(MASS)
model <- lm(satisfaction~airline_status+age+gender+type_of_travel+arrival_delay_greater_5_mins+num_flights+arrival
_delay_in_minutes+departure_delay_in_minutes+num_loyalty_cards, data=df)
selected_model <- stepAIC(model, direction="backward", trace=TRUE)
```

<...output truncated>

```
summary(selected_model)
```

```
##
## Call:
## lm(formula = satisfaction ~ airline_status + age + gender + type_of_travel +
##     arrival_delay_greater_5_mins + num_flights + departure_delay_in_minutes,
##     data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -3.08100 -0.40458  0.02695  0.49302  2.76463
##
## Coefficients:
##                                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)                     3.7727675  0.0150387 250.871  < 2e-16 ***
## airline_statusGold              0.4449571  0.0167983  26.488  < 2e-16 ***
## airline_statusPlatinum          0.2906776  0.0260358  11.165  < 2e-16 ***
## airline_statusSilver            0.6496885  0.0115067  56.462  < 2e-16 ***
## age                            -0.0021793  0.0002799  -7.786 7.19e-15 ***
## genderMale                      0.1243370  0.0091996  13.515  < 2e-16 ***
## type_of_travelMileage          -0.1572769  0.0173133  -9.084  < 2e-16 ***
## type_of_travelPersonal         -1.0909831  0.0109545 -99.592  < 2e-16 ***
## arrival_delay_greater_5_minsyes -0.3496571  0.0120358 -29.052  < 2e-16 ***
## num_flights                    -0.0029503  0.0003382  -8.723  < 2e-16 ***
## departure_delay_in_minutes      0.0005591  0.0002716   2.058   0.0396 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7221 on 25960 degrees of freedom
## Multiple R-squared:  0.4487, Adjusted R-squared:  0.4485
## F-statistic:  2113 on 10 and 25960 DF,  p-value: < 2.2e-16
```

Interpretation of Adj R-squared and coefficients

- This model explains 44% (Adj. R-squared value) of the variance around the mean of the dependent variable satisfaction
- Customers who are either Gold, Platinum, or Silver have satisfaction ratings of (.44, .25, and .62) higher than customers that don't have a flight_status
- The "Silver" status has the strongest positive correlation with customer satisfaction amongst the airline status categories. Its' coefficient is interpreted as, "for Silver status, and holding all other variables constant, the satisfaction score is .62 higher for Silver flyers than for all other airline_status categories"
- Holding all other variables constant, male customers' satisfaction scores are .13 higher than female customers
- Holdling all other variables constant, customers traveling for "Mileage" have satisfaction scores -.15 less than customers traveling for "Business"
- Holding all other variables constant, customers traveling for "Personal" have satisfaction scores -1.1 less than customers traveling for "Business"
- Holding all other variables constant, customers with flights delays greater than 5 minutes have satisfaction scores -.33 less than customers with flight delays less than 5 minutes

reorder the factors for airline_status and type_of_travel to see what the coefficients are for the current reference variables "Blue","Business"

```
df$airline_status <- factor(df$airline_status, levels=c("Silver","Gold","Platinum", "Blue"))
df$type_of_travel <- factor(df$type_of_travel, levels=c("Personal","Business","Mileage"))
# create the model again
model <- lm(satisfaction~airline_status+age+gender+type_of_travel+arrival_delay_greater_5_mins+num_flights+arrival
_delay_in_minutes+departure_delay_in_minutes+num_loyalty_cards, data=df)
selected_model <- stepAIC(model, direction="backward", trace=TRUE)
```

```
summary(selected_model)
```

```
##
## Call:
## lm(formula = satisfaction ~ airline_status + age + gender + type_of_travel +
##     arrival_delay_greater_5_mins + num_flights + departure_delay_in_minutes,
##     data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -3.08100 -0.40458  0.02695  0.49302  2.76463
##
## Coefficients:
##                                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)                     3.3314730  0.0208838 159.525  < 2e-16 ***
## airline_statusGold             -0.2047314  0.0186525 -10.976  < 2e-16 ***
## airline_statusPlatinum         -0.3590109  0.0272312 -13.184  < 2e-16 ***
## airline_statusBlue             -0.6496885  0.0115067 -56.462  < 2e-16 ***
## age                            -0.0021793  0.0002799  -7.786 7.19e-15 ***
## genderMale                      0.1243370  0.0091996  13.515  < 2e-16 ***
## type_of_travelBusiness          1.0909831  0.0109545  99.592  < 2e-16 ***
## type_of_travelMileage           0.9337062  0.0185542  50.323  < 2e-16 ***
## arrival_delay_greater_5_minsyes -0.3496571  0.0120358 -29.052  < 2e-16 ***
## num_flights                    -0.0029503  0.0003382  -8.723  < 2e-16 ***
## departure_delay_in_minutes      0.0005591  0.0002716   2.058   0.0396 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7221 on 25960 degrees of freedom
## Multiple R-squared:  0.4487, Adjusted R-squared:  0.4485
## F-statistic:  2113 on 10 and 25960 DF,  p-value: < 2.2e-16
```

4. Association Analysis.

cheap_associations.Rmd

Code

Summary of analysis

The various rulesets created returned the following trends:

- There are relatively few customers that give a rating of 5, but those that do have the following attributes: male, business trip, economy class, flight not cancelled, price_sensitivity=1
- Female business travelers–while few–also give higher ratings (4-5). They expect no cancellations and low delays. No female business travelers were Blue status in any of the rules; this correlates well with the negative coefficient for Blue status in the linear model.
- Female Blue status customers, traveling for personal reasons in the economy class give the lowest ratings

Miscellaneous Observations

- Surprisingly loyalty cards don't seem to influence higher satisfaction ratings. Loyalty cards were hardly present in pairings, and when they were, it was for 0 cards.

TODO

- Add the proportion of travelers in the assessment to increase the impact of the assessment. Who cares about the assesment if it accounts for only a small proportion of the population

Unanswered business questions

- What locations recieve the best/worst ratings? (done)

import data

Hide

```
df <- read.csv("Cheapseats.csv", stringsAsFactors = T)
```

discretize the following values

- cut returns a factor with the levels specified in the array indicated with (start, finish]; start is exclusive, finish is inclusive

Hide

```
df$satisfaction <- cut(df$satisfaction, c(0,1,2,3,4,5))
df$age <- cut(df$age, c(14,34,46,58,86))
df$price_sensitivity <- cut(df$price_sensitivity, c(-1,0,1,2,3,4))
df$year_first_flight <- cut(df$year_first_flight, breaks=3)
df$num_flights <- cut(df$num_flights, c(-1,0,9,18,21,30,94))
df$percent_flight_other_airlines <- cut(df$percent_flight_other_airlines, c(0,4,7,10,51))
df$num_loyalty_cards <- cut(df$num_loyalty_cards, c(-1,0,1,2,3,8))
df$airport_shopping <- cut(df$airport_shopping, c(-1,0,27,30,745))
df$airport_dining <- cut(df$airport_dining, c(-1,0,31,61,69,91,765))
df$day_of_month <- cut(df$day_of_month, c(0,1,9,17,24,32))
df$scheduled_departure_hour <- cut(df$scheduled_departure_hour, c(0,1,10,14,18,23))
df$flight_time_in_minutes <- cut(df$flight_time_in_minutes, c(0,27,59,87,102,360))
df$flight_distance <- cut(df$flight_distance, c(0,148,365,587,710,957,2329))
```

remove arrival_delay_in_minutes column

Hide

```
df <- df[,-which(colnames(df)=="arrival_delay_in_minutes")]
```

check for NA values

Hide

```
sapply(df, function(x) sum(is.na(x)))
```

```
                satisfaction              airline_status                         age                      g
ender
                           0                           0                           0
0
           price_sensitivity           year_first_flight                 num_flights percent_flight_other_air
lines
                           0                           0                           0
0
              type_of_travel           num_loyalty_cards           airport_shopping                airport_d
ining
                           0                           0                           0
0
                       class                day_of_month                 flight_date                 airline
_code
                           0                           0                           0
0
                airline_name                 origin_city                origin_state             destination
_city
                           0                           0                           0
0
           destination_state     scheduled_departure_hour   departure_delay_in_minutes             flight_canc
elled
                           0                           0                           0
0
       flight_time_in_minutes             flight_distance arrival_delay_greater_5_mins
                           0                           0                           0
```

change departure_delay_in_minutes to a factor of yes or no if the departure delay is greater than 15 minutes

- change the name of the column to departure_delay_greater_5_mins
- 11.7 came from theh industry average for departure delay

Hide

```
df$departure_delay_in_minutes <- ifelse(df$departure_delay_in_minutes > 11.7,"yes","no")
colnames(df)[which(colnames(df)=="departure_delay_in_minutes")] = "departure_delay_greater_5_mins"
df$departure_delay_greater_5_mins <- factor(df$departure_delay_greater_5_mins)
```

create sparse matrix from df

Hide

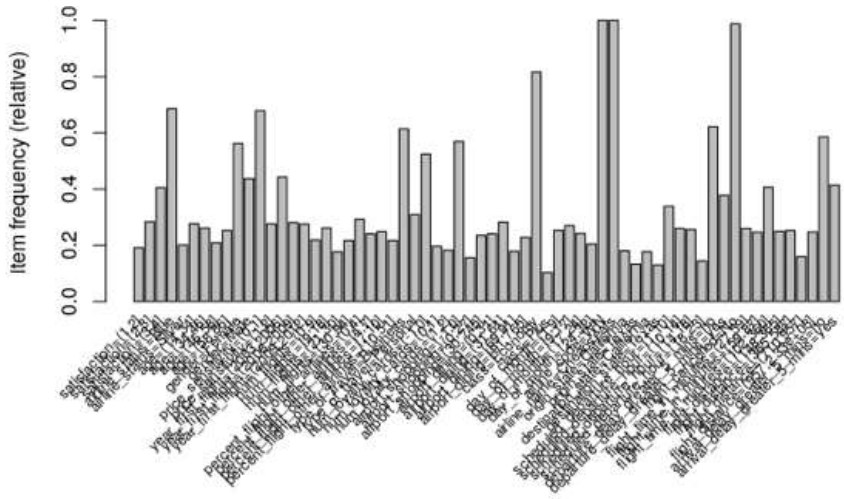
```
dfX <- as(df, "transactions")
```

inspect the first 2 rows from the matrix

show the item frequency to see which categories are present the most in the matrix

Hide

```
itemFrequencyPlot(dfX, support=.1, cex.names=.7)
```



Let's analyze flyers by the ratings groups of 1-5

customer satisfaction rating = 1

Hide

```
ruleset <- apriori(dfX, parameter=list(support=.01, confidence=.5, minlen=2, maxlen=6, maxtime=10),
                   appearance = list(lhs="satisfaction=(0,1]"))
```

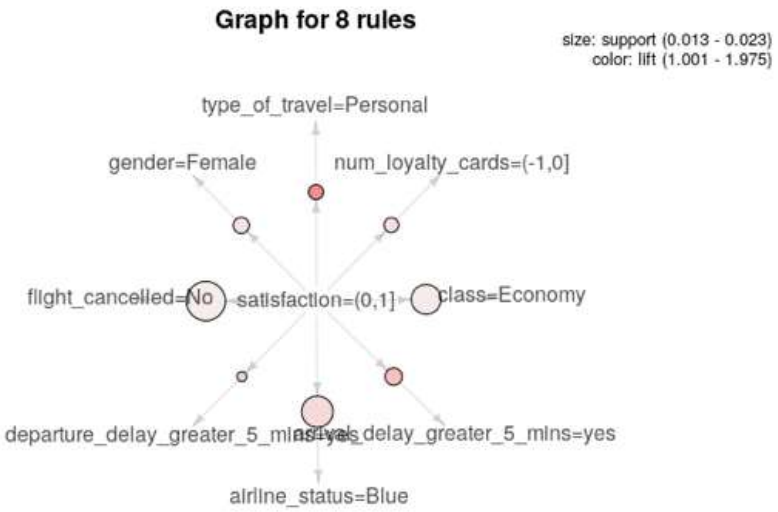
Hide

```
ruleset_filter <- subset(ruleset, subset=lhs %ain% c("satisfaction=(0,1]") & lift>1)
# plot(ruleset_filter)
# summary(ruleset_filter)
ruleset_filter <- sort(ruleset_filter, decreasing=T, by="count")
inspect(ruleset_filter)
```

```
    lhs                     rhs                                  support    confidence lift     count
[1] {satisfaction=(0,1]} => {flight_cancelled=No}                0.02337222 0.9885993  1.000737 607
[2] {satisfaction=(0,1]} => {airline_status=Blue}                0.02056140 0.8697068  1.267802 534
[3] {satisfaction=(0,1]} => {class=Economy}                      0.01994532 0.8436482  1.034338 518
[4] {satisfaction=(0,1]} => {arrival_delay_greater_5_mins=yes}   0.01536329 0.6498371  1.568633 399
[5] {satisfaction=(0,1]} => {gender=Female}                      0.01493974 0.6319218  1.123469 388
[6] {satisfaction=(0,1]} => {type_of_travel=Personal}            0.01443918 0.6107492  1.974576 375
[7] {satisfaction=(0,1]} => {num_loyalty_cards=(-1,0]}           0.01443918 0.6107492  1.164081 375
[8] {satisfaction=(0,1]} => {departure_delay_greater_5_mins=yes} 0.01262947 0.5342020  1.412663 328
```

Hide

```
plot(ruleset_filter, method="graph")
```



- there are relatively few customers who give a customer satisfaction rating of 1, but those that do have one or more of the following attributes: female, personal travel, arrivel delay > 5 mins, blue status, economy class, 0 loyalty cards, and 0 airport_shopping

customer satisfaction rating = 2

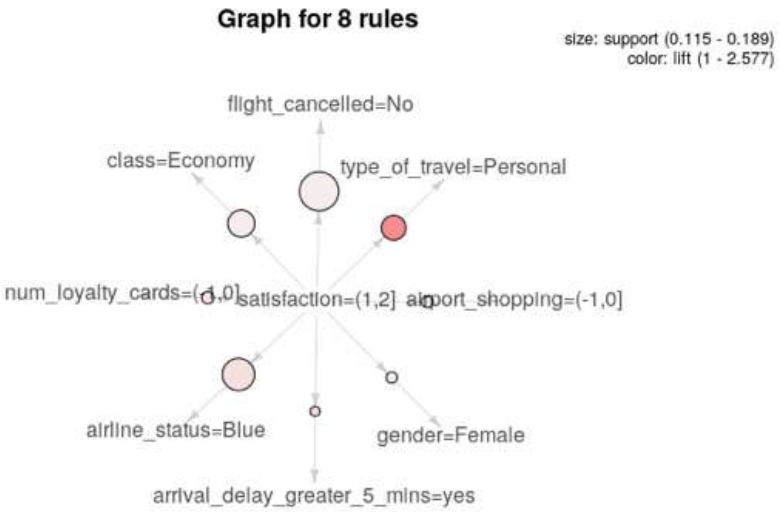
Hide

```
ruleset <- apriori(dfX, parameter=list(support=.1, confidence=.5, minlen=2, maxtime=10),
                   appearance = list(lhs="satisfaction=(1,2]"))
```

```
    lhs                    rhs                                  support   confidence lift     count
[1] {satisfaction=(1,2]} => {flight_cancelled=No}             0.1892495 0.9881383  1.000271 4915
[2] {satisfaction=(1,2]} => {airline_status=Blue}             0.1719225 0.8976679  1.308562 4465
[3] {satisfaction=(1,2]} => {class=Economy}                   0.1586385 0.8283072  1.015530 4120
[4] {satisfaction=(1,2]} => {type_of_travel=Personal}         0.1526318 0.7969441  2.576551 3964
[5] {satisfaction=(1,2]} => {num_loyalty_cards=(-1,0]}        0.1212121 0.6328910  1.206283 3148
[6] {satisfaction=(1,2]} => {airport_shopping=(-1,0]}         0.1194794 0.6238440  1.095756 3103
[7] {satisfaction=(1,2]} => {gender=Female}                   0.1183243 0.6178126  1.098385 3073
[8] {satisfaction=(1,2]} => {arrival_delay_greater_5_mins=yes} 0.1151284 0.6011259  1.451049 2990
```

Hide

```
plot(ruleset_filter, method="graph")
```



- Virtually the same result from customers with ratings=1; female, personal travel, blue status, economy class, 0 loyalty cards, and 0 airport_shopping

customer satisfaction rating = 3

Hide

```
ruleset <- apriori(dfX, parameter=list(support=.1, confidence=.5, minlen=2, maxlen=6, maxtime=10),
                   appearance = list(lhs="satisfaction=(2,3]"))
```

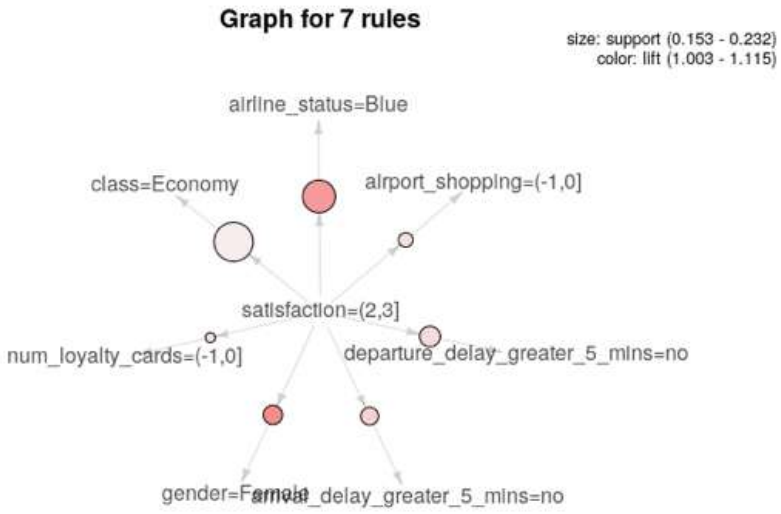
Hide

```
ruleset_filter <- sort(ruleset_filter, decreasing=T, by="count")
inspect(ruleset_filter)
```

```
    lhs                     rhs                                  support   confidence lift     count
[1] {satisfaction=(2,3]} => {class=Economy}                      0.2323746 0.8179724  1.002859 6035
[2] {satisfaction=(2,3]} => {airline_status=Blue}                0.2150090 0.7568447  1.103279 5584
[3] {satisfaction=(2,3]} => {departure_delay_greater_5_mins=no} 0.1821262 0.6410951  1.030952 4730
[4] {satisfaction=(2,3]} => {gender=Female}                      0.1781603 0.6271347  1.114959 4627
[5] {satisfaction=(2,3]} => {arrival_delay_greater_5_mins=no}    0.1742328 0.6133098  1.047086 4525
[6] {satisfaction=(2,3]} => {airport_shopping=(-1,0]}            0.1652613 0.5817295  1.021784 4292
[7] {satisfaction=(2,3]} => {num_loyalty_cards=(-1,0]}           0.1525933 0.5371374  1.023778 3963
```

Hide

```
plot(ruleset_filter, method="graph")
```



- Similar assessment to customers with rating=2 *except* for one key difference: arrival and departure delays are NOT greater than 5 mins

customer satisfaction rating = 4

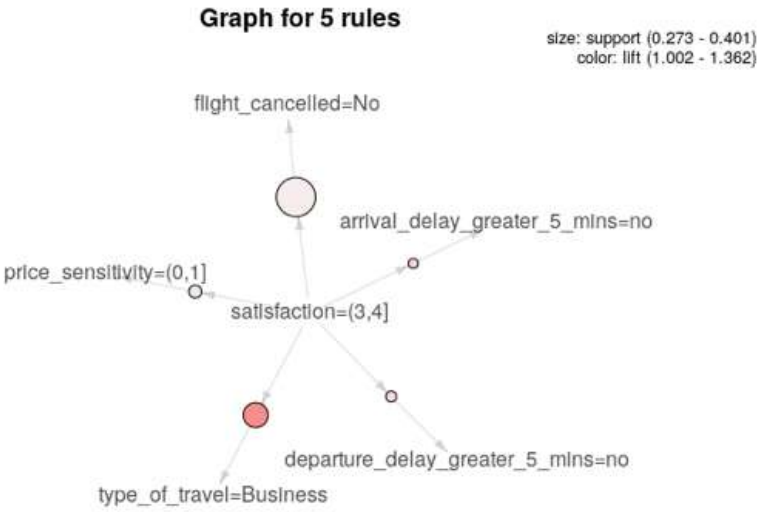
Hide

```
ruleset <- apriori(dfX, parameter=list(support=.05, confidence=.5, minlen=2, maxtime=10),
                   appearance = list(lhs="satisfaction=(3,4]"))
```

```
    lhs                     rhs                                  support   confidence lift     count
[1] {satisfaction=(3,4]} => {flight_cancelled=No}                0.4012938 0.9901197  1.002276 10422
[2] {satisfaction=(3,4]} => {type_of_travel=Business}            0.3389550 0.8363101  1.361658  8803
[3] {satisfaction=(3,4]} => {price_sensitivity=(0,1]}            0.2853567 0.7040661  1.036640  7411
[4] {satisfaction=(3,4]} => {departure_delay_greater_5_mins=no} 0.2768473 0.6830705  1.098453  7190
[5] {satisfaction=(3,4]} => {arrival_delay_greater_5_mins=no}    0.2728813 0.6732852  1.149480  7087
```

Hide

```
plot(ruleset_filter, method="graph")
```



- Customers with one or more of the following attributes (business, departure/arrival delay < 5 min, flight not cancelled, price_sensitivity=1) give ratings=4

customer satisfaction rating = 5

Hide

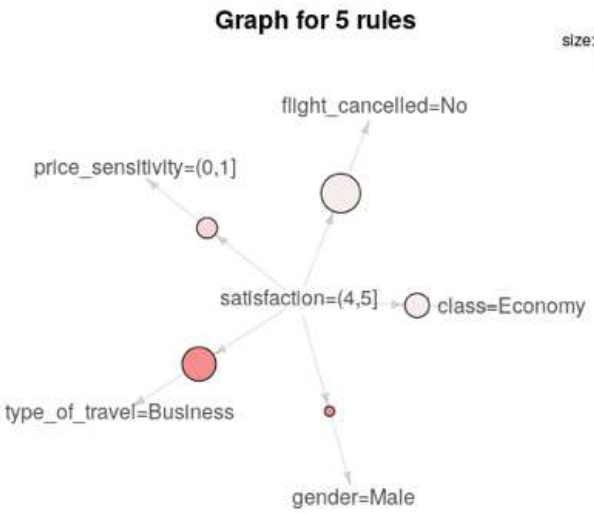
```
ruleset <- apriori(dfX, parameter=list(support=.01, confidence=.5, minlen=2, maxlen=6, maxtime=10),
                   appearance = list(lhs="satisfaction=(4,5]"))
```

```
ruleset_filter <- sort(ruleset_filter, decreasing=T, by="count")
inspect(ruleset_filter)
```

```
    lhs                     rhs                          support    confidence lift     count
[1] {satisfaction=(4,5]} => {flight_cancelled=No}      0.09526010 0.9979831  1.010236 2474
[2] {satisfaction=(4,5]} => {type_of_travel=Business} 0.08959994 0.9386850  1.528342 2327
[3] {satisfaction=(4,5]} => {class=Economy}            0.07908821 0.8285599  1.015840 2054
[4] {satisfaction=(4,5]} => {price_sensitivity=(0,1]} 0.07492973 0.7849939  1.155796 1946
[5] {satisfaction=(4,5]} => {gender=Male}              0.06314736 0.6615571  1.512039 1640
```

Hide

```
plot(ruleset_filter, method="graph")
```



- Customers with one or more of the following attributes give a rating=5: male, business trip, economy class, price sensitivity=1, flight not cancelled

5. Support Vector Machine.

Cheapseats SVM

Code

import the Cheapseats dataset

Hide

```
ksvm_df <- read.csv("Cheapseats.csv", stringsAsFactors = F)
```

label Cheapseats airline satisfaction

Hide

```
ksvm_df$satisfactionLabel <- ifelse(ksvm_df$satisfaction>3,"happy","unhappy")
ksvm_df$satisfactionLabel <- factor(ksvm_df$satisfactionLabel)
```

create test/train data

Hide

```
createTestTrain <- function() {
  n = dim(svm_df)[1]
  train_index = sample(1:n, size=.7*n, replace = F)  # create random sample with size 70% of n
  train <- ksvm_df[train_index,]  # set train data to the random indices generated
  test <- ksvm_df[-train_index,] # set test data to exclude the random indices
}
```

build the svm model

Hide

```
svm_output <- ksvm(satisfactionLabel~airline_status + age + gender + type_of_travel + arrival_delay_greater_5_mins
+ num_flights, data=train, model="rfbfot", kpar="automatic", C=10, cross=3, prob.model=T)
svm_output
```

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

SV type: C-svc  (classification)
 parameter : cost C = 10

Gaussian Radial Basis kernel function.
 Hyperparameter : sigma =  0.269200859333213

Number of Support Vectors : 7896

Objective Function Value : -74386
Training error : 0.19957
Cross validation error : 0.204034
Probability model included.
```

Hide

```
svm_predict <- predict(svm_output, test, type="votes")
compTable <- data.frame(test[,29], svm_predict[1,])  # creates a dataframe from the from test$classify and the fir
st row in the svm_predict matrix
table(compTable)
```

```
          svm_predict.1...
test...29.    0    1
   happy    254 3674
   unhappy 2520 1379
```

the svm produces an error rate of 20%, let's see if we can reduce that by removing the variables with less correlation to satisfaction

- "The r-squared value for gender: 0.017487"
- "The r-squared value for age: 0.049643"
- create new test/train data before running

Hide

```
createTestTrain()
svm_output2 <- ksvm(satisfactionLabel~airline_status + type_of_travel + arrival_delay_greater_5_mins + num_flights
, data=train, model="rfbfot", kpar="automatic", C=10, cross=3, prob.model=T)
svm_output2
```

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

SV type: C-svc  (classification)
 parameter : cost C = 10

Gaussian Radial Basis kernel function.
 Hyperparameter : sigma =  0.541010239551846

Number of Support Vectors : 9000

Objective Function Value : -81463.58
Training error : 0.22371
Cross validation error : 0.225805
Probability model included.
```

Hide

```
svm_predict2 <- predict(svm_output2, test, type="votes")
compTable2 <- data.frame(test[,29], svm_predict2[1,])  # creates a dataframe from the from test$classify and the f
irst row in the svm_predict matrix
table(compTable2)
```

```
          svm_predict2.1...
test...29.    0    1
   happy    201 3727
   unhappy 2316 1583
```

prediction rate does not seem to improve with removing variables; nevertheless, let's remove all but the strongest predictors

- create new test/train data before running

Hide

```
createTestTrain()
svm_output3 <- ksvm(satisfactionLabel~airline_status + type_of_travel, data=train, model="rfbfot", kpar="automatic
", C=10, cross=3, prob.model=T)
```

```
line search fails -1.288316 -0.4215751 3.173035e-05 3.172833e-05 -1.823531e-08 -1.822239e-08 -1.156779e-12
```

Hide

```
svm_output3
```

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

SV type: C-svc  (classification)
 parameter : cost C = 10

Gaussian Radial Basis kernel function.
 Hyperparameter : sigma =  0.666666666666667

Number of Support Vectors : 8835

Objective Function Value : -88247.08
Training error : 0.243166
Cross validation error : 0.243276
Probability model included.
```

Hide

```
svm_predict3 <- predict(svm_output3, test, type="votes")
compTable3 <- data.frame(test[,29], svm_predict3[1,])  # creates a dataframe from the from test$classify and the f
irst row in the svm_predict matrix
table(compTable3)
```

```
          svm_predict3.1...
test...29.    0    1
   happy    459 3469
   unhappy 2406 1493
```

6. This code was used to generate latitude and longitude for all cities and states and to plot map based visualizations

goecode

Code ▾

TODO

- display cancellation frequency/ratio
- display flight code frequency/ratio
- display total flight frequency/ratio

import code

Hide

```
df <- read.csv("flight_survey_updated.csv", stringsAsFactors = F)
```

Get geocode information for *origin_city* and *destination_city*

- Some city state values have multi-city values such as "Bristol/Johnson City/Kingsport, TN". The gecode package won't accept such as string as an argument, therefore we will use regex to parse the string to build the string stated above as "Bristol, TN". It is assummed that most of the multi city values are relatively close to each other (i.e Bristol is < 10 miles to Johnson City < 10 miles to Kingsport).
- There are 9877 cases for origin city that have a "/" with multi-city values
- There are 9792 cases for destination city that have a "/" with multi-city values

Create two additional columns for lat/lon coordinates for origin city,state and desitination city,state

Hide

```
df$origin_city_state <- gsub("/.*,", ",", df$origin_city)
df$destination_city_state <- gsub("/.*,", ",", df$destination_city)
```

re-order the columns to put the geocode columns side-by-side to make validation easier

Hide

```
cols <- c('satisfaction', 'airline_status', 'age', 'gender', 'price_sensitivity', 'year_first_flight', 'num_flights', 'percent_flight_other_airlines', 'type_of_travel', 'num_loyalty_cards', 'airport_shopping', 'airport_dining', 'class', 'day_of_month', 'flight_date', 'airline_code', 'airline_name', 'origin_city', 'origin_city_state', 'origin_state', 'destination_city', 'destination_city_state', 'destination_state', 'scheduled_departure_hour', 'departure_delay_in_minutes', 'arrival_delay_in_minutes', 'flight_cancelled', 'flight_time_in_minutes', 'flight_distance', 'arrival_delay_greater_5_mins')
df <- df[,cols]
```

change origin_state and destination_state to lowercase

Hide

```
df$origin_state <- tolower(df$origin_state)
df$destination_state <- tolower(df$destination_state)
#df$destination_city <- tolower(df$destination_city)
#df$origin_city <- tolower(df$origin_city)
#df$destination_city_state <- tolower(df$destination_city_state)
#df$origin_city_state <- tolower(df$origin_city_state)
#mydf$origin_city <- tolower(mydf$origin_city)
#mydf$destination_city <- tolower(mydf$destination_city)
```

create a function to send a http request to a geocode api

- this process is a lot faster if you build the data science toolkit (vagrant VM) locally using the instructions here: http://www.datasciencetoolkit.org/developerdocs#setup

Hide

```
getGeocode <- function(cityState) {
  root <- "http://192.168.3.106:8080/maps/api/geocode/"
  url <- paste(root, "json?address=", cityState, "&sensor=false", sep="") %>%  URLencode()
  result <- getURL(url)
  myJson <- fromJSON(result, simplify = F)
  lat <- NA
  ln <- NA
  try(lat <- myJson$results[[1]]$geometry$location$lat)
  try(lng <- myJson$results[[1]]$geometry$location$lng)
  return(c(lat,lng))
}
```

call getLatLon function to get the origin coordinates

Hide

```
getOriginLatLng <- function() {
  latLngOrigin <- unique(df$origin_city_state)
  origin_df <- data.frame(origin=latLngOrigin,orig_lat=rep(NA, length(latLngOrigin)),orig_lng=rep(NA, length(latLn
gOrigin)),stringsAsFactors = F)
  for(i in 1:length(latLngOrigin)) {
    tryResult <- tryCatch(
      {
        origin_df[i,"orig_lat"] <- getGeocode(latLngOrigin[i])[1]
        origin_df[i,"orig_lng"] <- getGeocode(latLngOrigin[i])[2]
        message(sprintf("Successfully queried the dstk geocode api for: %s", latLngOrigin[i]))
      },
      error=function(cond) {
        message(paste("An error occured when calling the geocode api for %s: ", latLngOrigin[i]))
        message(cond)
      },
      warning=function(cond) {
        messagge(cond)
      },
      finally={

      }
    )
  }
  return(origin_df)
}
origin_df <- getOriginLatLng()
```

call getLatLon function to get the destination coordinates

Hide

```
getOriginLatLng <- function() {
  latLngOrigin <- unique(df$origin_city_state)
  origin_df <- data.frame(origin=latLngOrigin,orig_lat=rep(NA, length(latLngOrigin)),orig_lng=rep(NA, length(latLn
gOrigin)),stringsAsFactors = F)
  for(i in 1:length(latLngOrigin)) {
    tryResult <- tryCatch(
      {
        origin_df[i,"orig_lat"] <- getGeocode(latLngOrigin[i])[1]
        origin_df[i,"orig_lng"] <- getGeocode(latLngOrigin[i])[2]
        message(sprintf("Successfully queried the dstk geocode api for: %s", latLngOrigin[i]))
      },
      error=function(cond) {
        message(paste("An error occured when calling the geocode api for %s: ", latLngOrigin[i]))
        message(cond)
      },
      warning=function(cond) {
        messagge(cond)
      },
      finally={

      }
    )
  }
  return(origin_df)
}
origin_df <- getOriginLatLng()
```

call getLatLon function to get the destination coordinates

Hide

```
getDestinationLatLng <- function() {
  latLngDestination <- unique(df$destination_city_state)
  destination_df <- data.frame(origin=latLngDestination, dest_lat=rep(NA, length(latLngDestination)),dest_lng=rep(
NA, length(latLngDestination)),stringsAsFactors = F)
  for(i in 1:length(latLngDestination)) {
    tryResult <- tryCatch(
      {
        destination_df[i,"dest_lat"] <- getGeocode(latLngDestination[i])[1]
        destination_df[i,"dest_lng"] <- getGeocode(latLngDestination[i])[2]
        message(sprintf("Successfully queried the dstk geocode api for: %s", latLngDestination[i]))
      },
      error=function(cond) {
        message(paste("An error occured when calling the geocode api for %s: ", latLngDestination[i]))
        message(cond)
      },
      warning=function(cond) {
        messagge(cond)
      },
      finally={

      }
    )
  }
  return(destination_df)
}
destination_df <- getDestinationLatLng()
```

map the lat/lng columns in airline dataset to origin_df and destination_df

- create the lat/lng columns first

St. Louis, MO didn't return lat/lng in the script–ughh! dstk prefers Saint vs. St.

- let's add the lat/lon for saint louis, mo manualy

Find the NA values for lat/lng

Hide

```
sapply(df, function(x) sum(is.na(x)))
df[which(is.na(df$orig_lat)),] %>% head()
```

	destination_city_state <chr>	origin_city_state <chr>	satisfaction <int>	airline_status <chr>	a... <int>	gender <chr>
2067	Atlanta, GA	St. Louis, MO	1	Blue	62	Male
2153	Atlanta, GA	St. Louis, MO	4	Blue	59	Female
2155	Atlanta, GA	St. Louis, MO	4	Blue	45	Male
2333	Atlanta, GA	St. Louis, MO	4	Blue	63	Female
2391	Atlanta, GA	St. Louis, MO	3	Blue	30	Female
2430	Atlanta, GA	St. Louis, MO	3	Gold	45	Female

6 rows | 1-7 of 34 columns

Hide

```
df[which(is.na(df$dest_lat)),] %>% head()
```

	destination_city_state <chr>	origin_city_state <chr>	satisfaction <int>	airline_status <chr>	a... <int>	gender <chr>
121149	St. Louis, MO	Dallas, TX	4	Blue	39	Male
121150	St. Louis, MO	Las Vegas, NV	4	Blue	33	Male
121151	St. Louis, MO	Dallas, TX	3	Blue	41	Male
121152	St. Louis, MO	Atlanta, GA	2	Blue	45	Female
121153	St. Louis, MO	Atlanta, GA	5	Silver	69	Male
121154	St. Louis, MO	Denver, CO	4	Silver	53	Female

6 rows | 1-7 of 34 columns

set lat/lng for St. Louis

write updated dataset with lat/lng to disk

Hide

```
write.csv(df, "flight_surcey_udated_geo.csv", row.names=F)
```

Remove origin/destination not in the contienental US

- the map_data state map doesn't have u.s. pacific trust territories and possessions; let's remove those 19 rows
- the map_data state map doesn't have puerto rico; let's remove those 700 rows
- the map_data state map doesn't have hawaii; let's remove those 2921 rows
- the map_data state map doesn't have alaska; let's remove those 720, rows
- number of observerations before: 129435

Hide

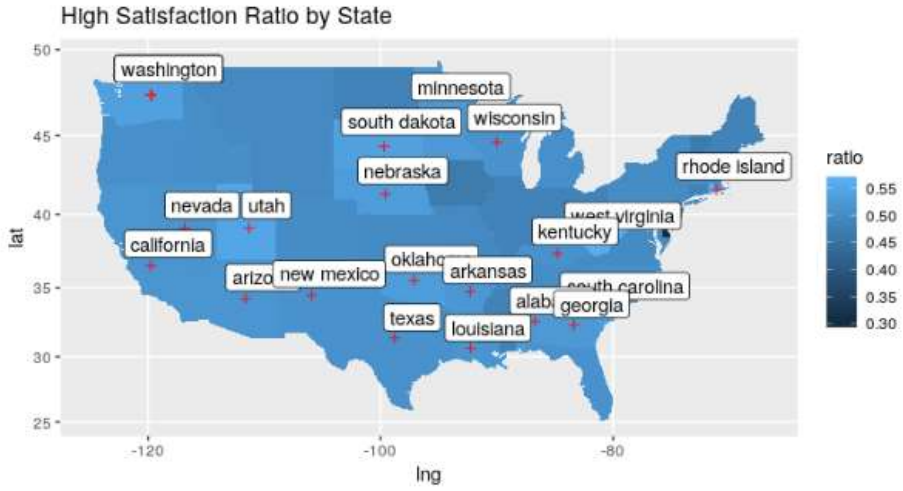
```
df <- df[-which(df$origin_state=="u.s. pacific trust territories and possessions"),]
df <- df[-which(df$destination_state=="u.s. pacific trust territories and possessions"),]
df <- df[-which(df$origin_state=="puerto rico"),]
df <- df[-which(df$destination_state=="puerto rico"),]
df <- df[-which(df$origin_state=="hawaii"),]
df <- df <- df[-which(df$destination_state=="hawaii"),]
df <- df[-which(df$origin_state=="alaska"),]
df <- df[-which(df$destination_state=="alaska"),]
```

- number of observations after: 129435

plot states' rating=4 or 5/total flights ratio on map

Hide

```
plotHighSatisfaction <- function() {
  states = map_data("state")
  high_ratings <- df %>% dplyr::select(satisfaction,origin_state) %>%
  dplyr::filter(satisfaction==4 | satisfaction==5) %>%
  dplyr::group_by(origin_state) %>%
  dplyr::summarise(count=n()) %>%
  dplyr::arrange(desc(count))
  state_freq <- table(df$origin_state) %>% data.frame()
  high_ratings <- merge(high_ratings,state_freq, by.x="origin_state", by.y="Var1")
  high_ratings$ratio <- high_ratings$count/high_ratings$Freq
  state_center <- as.data.frame(matrix(unlist(state.center), nrow=length(unlist(state.center[1])), byrow=FALSE))
  state_center$name <- state.name %>% tolower()
  high_ratings <- merge(high_ratings,state_center, by.x="origin_state", by.y="name")
  high_ratings <- high_ratings[,c('origin_state', 'count', 'Freq', 'ratio', 'V2', 'V1')]
  colnames(high_ratings) <- c('origin_state', 'count', 'Freq', 'ratio', 'lat', 'lng')
  row.names(high_ratings) <- NULL
  myStates <- states[,c("region","group")]
  high_ratings <- merge(high_ratings,myStates,by.x="origin_state",by.y="region")
  top_high_ratings <- high_ratings[order(high_ratings$ratio, decreasing = T),] %>% unique() %>% head(24)  # accoun
t for Washington being returned 5 times
  g <- ggplot(high_ratings, aes(x=lng,y=lat,map_id=origin_state)) +
      #geom_polygon(aes(group=group), color="black") +
      geom_map(map=states,aes(fill=ratio)) +
      geom_point(data=top_high_ratings,aes(x=lng,y=lat), shape=3, color="red" ) +
      geom_label(data=top_high_ratings,aes(label=(origin_state)),nudge_x = 1.5,nudge_y=1.5) +
      expand_limits(x=states$long,y=states$lat) +
      coord_map() +
      labs(title="High Satisfaction Ratio by State") # ratio = count(satisfaction = 4|5)/total flights
  g

  #return(top_high_ratings)
}
plotHighSatisfaction()
```

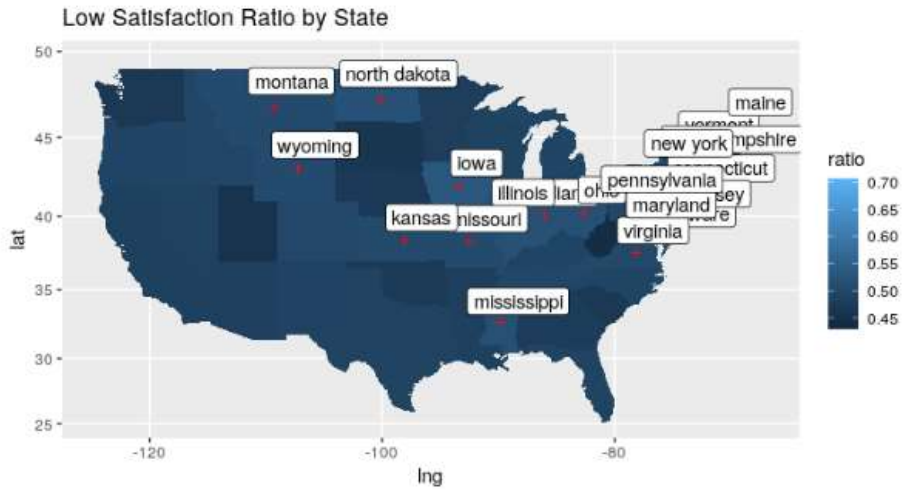


Hide

```
#test code
#test <- plotHighSatisfaction()
#high_ratings %>% dplyr::arrange(desc(ratio), origin_state, count, Freq) %>% unique() %>% head(20)
```

Hide

```
plotLowSatisfaction <- function() {
  low_ratings <- df %>% dplyr::select(satisfaction,origin_state) %>%
  dplyr::filter(satisfaction==1 | satisfaction==2 | satisfaction==3) %>%
  dplyr::group_by(origin_state) %>%
  dplyr::summarise(count=n()) %>%
  dplyr::arrange(desc(count))
  state_freq <- table(df$origin_state) %>% data.frame()
  low_ratings <- merge(low_ratings,state_freq, by.x="origin_state", by.y="Var1")
  low_ratings$ratio <- low_ratings$count/low_ratings$Freq
  state_center <- as.data.frame(matrix(unlist(state.center), nrow=length(unlist(state.center[1])), byrow=FALSE))
  state_center$name <- state.name %>% tolower()
  low_ratings <- merge(low_ratings,state_center, by.x="origin_state", by.y="name")
  low_ratings <- low_ratings[,c('origin_state', 'count', 'Freq', 'ratio', 'V2', 'V1')]
  colnames(low_ratings) <- c('origin_state', 'count', 'Freq', 'ratio', 'lat', 'lng')
  row.names(low_ratings) <- NULL
  us <- map_data("state")
  top_low_ratings <- low_ratings[order(low_ratings$ratio, decreasing = T),] %>% unique() %>% head(20)

  g <- ggplot(low_ratings, aes(x=lng,y=lat,map_id=origin_state)) +
      #geom_polygon(aes(group=group), color="black") +
      geom_map(map=us,aes(fill=ratio)) +
      geom_point(data=top_low_ratings,aes(x=lng,y=lat), shape=3, color="red" ) +
      geom_label(data=top_low_ratings,aes(label=(origin_state)),nudge_x = 1.5,nudge_y=1.5) +
      expand_limits(x=us$long,y=us$lat) +
      coord_map() +
      labs(title="Low Satisfaction Ratio by State") # ratio count(satisfaction=1|2|3)/total flights
  g

  # low_ratings %>% dplyr::arrange(desc(ratio)) %>% head(10)
  #return(top_low_ratings)
}
plotLowSatisfaction()
```



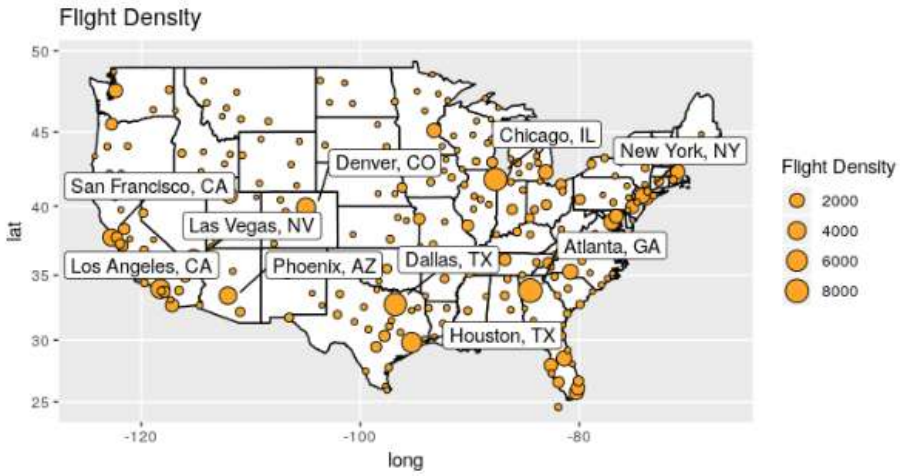
Hide

```
#test code
#test <- plotLowSatisfaction()
```

plot flight density

Hide

```
library(ggrepel)
plotFlightDensity <- function() {
  orig_flight_count <- df %>% dplyr::select(origin_city_state,origin_state,orig_lng,orig_lat) %>%
  dplyr::group_by(origin_city_state,origin_state,orig_lng,orig_lat) %>%
  dplyr::summarise(count=n()) %>%
  dplyr::arrange(desc(count),origin_state,orig_lng,orig_lat)

  biggerThan6000 <- orig_flight_count[which(orig_flight_count$count >= 3000),]
  states <- map_data("state")
  g <- ggplot(data=states,aes()) +
      geom_polygon(data=states,aes(x=long,y=lat,group=group), color="black",fill="white") +
      #geom_map(data=orig_flight_count,map=states, aes(fill=origin_state)) +
      geom_point(data=orig_flight_count, aes(x=orig_lng, y=orig_lat, size=count), shape=21, fill="orange") +
      scale_color_manual(name="Flight Density", values=myColors) +
      geom_label_repel(data=biggerThan6000,
                aes(x=orig_lng, y=orig_lat,label=origin_city_state),
                nudge_x = 1.5,nudge_y = 1.5,
                inherit.aes = F, point.padding = 1) +
      labs(title="Flight Density",size="Flight Density") +
      guides(fill=FALSE) +  # do this to leave off the color legend
      coord_map()
  print(g)
#return(orig_flight_count)
}
plotFlightDensity()
```



What origin locations recieve the best ratings for Cheapseats?

filter the dataset for all cheapseats flights that recevied a rating of 4|5

plot the top/bottom locations to a map

Hide

```
plotTopBottom <- function(myData,myTitle) {
  states <- map_data("state")
  g <- ggplot(data=states) +
    geom_polygon(aes(x=long,y=lat,fill=region, group=group), color="black",fill="white") +
    geom_point(data=myData, aes(x=orig_lng, y=orig_lat, size=ratio), shape=21, fill="deepskyblue3") +
    labs(title=myTitle,size="Satisfaction Ratio") +
    geom_label(data=myData,
               aes(x=orig_lng, y=orig_lat,label=origin_city_state),
               nudge_x = 1.5,nudge_y = 1.5,
               inherit.aes = F) +
    coord_map()
  print(g)
}
```

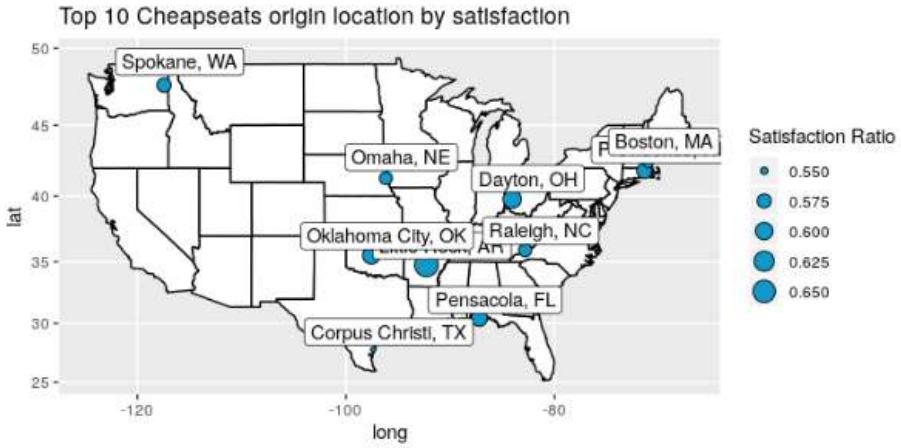
plot cheapseats top 20 and bottom 20 origin locations by customer satisfaction ratio 4|5 divided by all flights from origin location

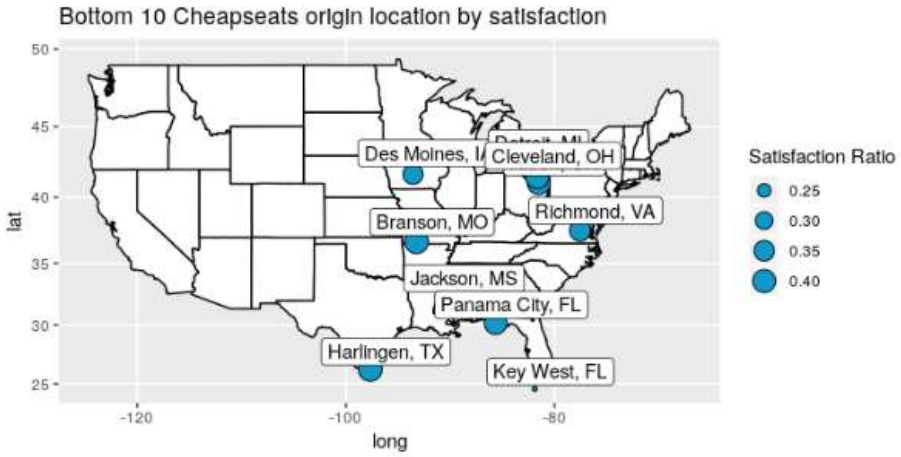
- Use dplyr to filter top/bottom 20 origin locations for airlineName argument
- Does not filter for gender

Loop over airlines and get top/bottom origin location by satisfaction

Hide

```
airlines <- c("Cheapseats","Oursin")  # pass in airline names to get topBottom stats by origin location
for(airline in airlines) {
    #title <- paste(airline,status,"origin locations by satisfaction",sep=" ")
    #print(title)
    data <- getTopBottomData(airline) # returns array with top/bottom data
    plotTopBottom(data[[1]],paste("Top 10",airline,"origin location by satisfaction",sep=" "))
    plotTopBottom(data[[2]],paste("Bottom 10",airline,"origin location by satisfaction",sep=" "))
}
```


Top 10 Cheapseats origin location by satisfaction
Spokane, WA
Omaha, NE
Boston, MA
Dayton, OH
Oklahoma City, OK
Raleigh, NC
Pensacola, FL
Corpus Christi, TX
lat
long
50
45
40
35
30
25
-120
-100
-80
Satisfaction Ratio
0.550
0.575
0.600
0.625
0.650


Bottom 10 Cheapseats origin location by satisfaction
Des Moines, I
Cleveland, OH
Branson, MO
Richmond, VA
Jackson, MS
Panama City, FL
Harlingen, TX
Key West, FL
lat
long
50
45
40
35
30
25
-120
-100
-80
Satisfaction Ratio
0.25
0.30
0.35
0.40

plot top bottom origin locations by satisfaction

Hide

```
plotTopBottomFemale <- function(myData,myTitle) {
  states <- map_data("state")
  g <- ggplot(data=states) +
    geom_polygon(aes(x=long,y=lat,fill=region, group=group), color="black",fill="white") +
    geom_point(data=myData, aes(x=orig_lng, y=orig_lat, size=ratio), shape=21, fill="deeppink") +
    labs(title=myTitle,size="Female ratio") +
    geom_label(data=myData,
               aes(x=orig_lng, y=orig_lat,label=origin_city_state),
               nudge_x = 1.5,nudge_y = 1.5,
               inherit.aes = F) +
    coord_map()
  print(g)
}
```

```
The working directory was changed to /home/asoa inside a notebook chunk. The working directory will be reset when
the chunk is finished running. Use the knitr root.dir option in the setup chunk to change the working directory fo
r notebook chunks.
```

- TODO: Find out wha arules are associated with the top/bottom few locations
- TODO: Can we plot satisfation by male/female? (done)

Hide

```
plotTopBottomMale <- function(myData,myTitle) {
  states <- map_data("state")
  g <- ggplot(data=states) +
    geom_polygon(aes(x=long,y=lat,fill=region, group=group), color="black",fill="white") +
    geom_point(data=myData, aes(x=orig_lng, y=orig_lat, size=ratio), shape=21, fill="deepskyblue3") +
    labs(title=myTitle,size="Male ratio") +
    geom_label(data=myData,
               aes(x=orig_lng, y=orig_lat,label=origin_city_state),
               nudge_x = 1.5,nudge_y = 1.5,
               inherit.aes = F) +
    coord_map()
  print(g)
}
```

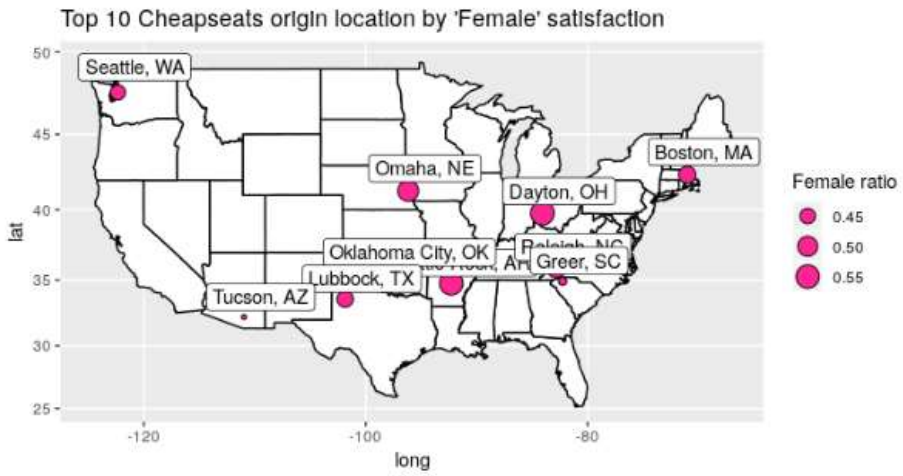
Get top/bottom locations by gender

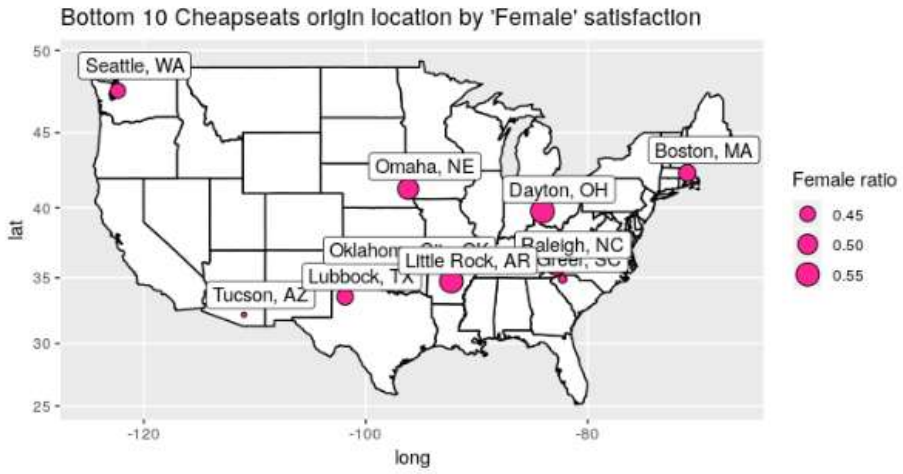
Hide

```
getTopBottomFemale <- function(airlineName){
  top_ratings <- df %>% dplyr::select(satisfaction, gender, origin_city_state, origin_state, orig_lat, orig_lng, a
irline_name) %>%  # summarized top ratings by city_state
    dplyr::filter(airline_name==airlineName & gender=="Female" & origin_city_state %in% c("Little Rock, AR","Oklah
oma City, OK","Greer, SC","Raleigh, NC","Boston, MA","Lubbock, TX","Dayton, OH","Omaha, NE","Tucson, AZ","Seattle,
WA") & (satisfaction==4 | satisfaction==5)) %>%
    dplyr::group_by(origin_city_state,origin_state, orig_lat, orig_lng, airline_name) %>%
    dplyr::summarise(count=n()) %>%
    dplyr::arrange(origin_city_state,origin_state, orig_lat, orig_lng, airline_name)

  city_state_freq <- df %>% dplyr::select(airline_name,gender,origin_city_state,origin_state) %>%  # summarized ai
rline flights by city_state
    dplyr::filter(airline_name==airlineName & gender=="Female") %>%
    dplyr::group_by(airline_name,origin_city_state,origin_state) %>%
    dplyr::summarise(count=n()) %>%
    dplyr::arrange(airline_name,origin_city_state,origin_state,count)

  city_state_freq <- city_state_freq[,-1]  # prep for merge
  city_state_freq <- city_state_freq[,-2]

  top_ratings <- merge(top_ratings,city_state_freq, by.x="origin_city_state",by.y="origin_city_state")
  top_ratings$ratio <- top_ratings$count.x/top_ratings$count.y
  top10 <- top_ratings[order(top_ratings$ratio,decreasing = T),] %>% head(10)
  bottom10 <- top_ratings[order(top_ratings$ratio,decreasing = F),] %>% head(10)

  plotTopBottomFemale(top10, paste("Top 10",airlineName,"origin location by 'Female' satisfaction",sep=" "))
  plotTopBottomFemale(bottom10, paste("Bottom 10",airlineName,"origin location by 'Female' satisfaction",sep=" "))
  return(top10)
}
getTopBottomFemale("Cheapseats")
```

	origin_city_state <chr>	origin_state <chr>	orig_lat <dbl>	orig_lng <dbl>	airline_name <chr>	count.x <int>	count.y <int>	ratio <dbl>
2	Dayton, OH	ohio	39.76419	-84.07615	Cheapseats	5	9	0.5555556
4	Little Rock, AR	arkansas	34.74648	-92.28960	Cheapseats	26	47	0.5531915
7	Omaha, NE	nebraska	41.29174	-96.17110	Cheapseats	38	74	0.5135135
8	Raleigh, NC	north carolina	35.88904	-82.83104	Cheapseats	56	112	0.5000000
6	Oklahoma City, OK	oklahoma	35.49161	-97.56282	Cheapseats	30	64	0.4687500
1	Boston, MA	massachusetts	42.37057	-71.02696	Cheapseats	43	92	0.4673913
5	Lubbock, TX	texas	33.57786	-101.85517	Cheapseats	17	37	0.4594595
9	Seattle, WA	washington	47.60621	-122.33207	Cheapseats	55	123	0.4471545
3	Greer, SC	south carolina	34.89825	-82.25785	Cheapseats	11	27	0.4074074
10	Tucson, AZ	arizona	32.21798	-110.97087	Cheapseats	33	82	0.4024390

1-10 of 10 rows


Top 10 Cheapseats origin location by 'Female' satisfaction
Seattle, WA
Omaha, NE
Dayton, OH
Boston, MA
Oklahoma City, OK
Greer, SC
Lubbock, TX
Tucson, AZ
Female ratio
0.45
0.50
0.55
lat
long
50
45
40
35
30
25
-120
-100
-80

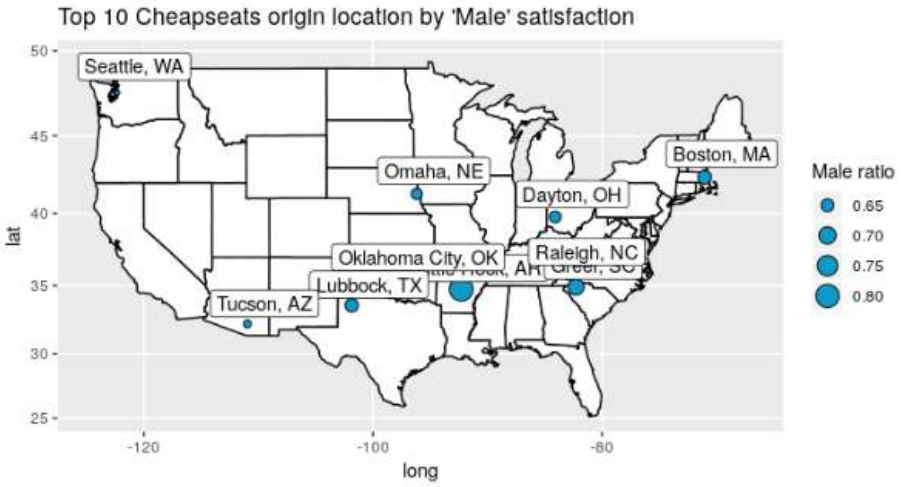

Bottom 10 Cheapseats origin location by 'Female' satisfaction
Seattle, WA
Omaha, NE
Dayton, OH
Boston, MA
Raleigh, NC
Little Rock, AR
Lubbock, TX
Tucson, AZ
Female ratio
0.45
0.50
0.55
lat
long
50
45
40
35
30
25
-120
-100
-80

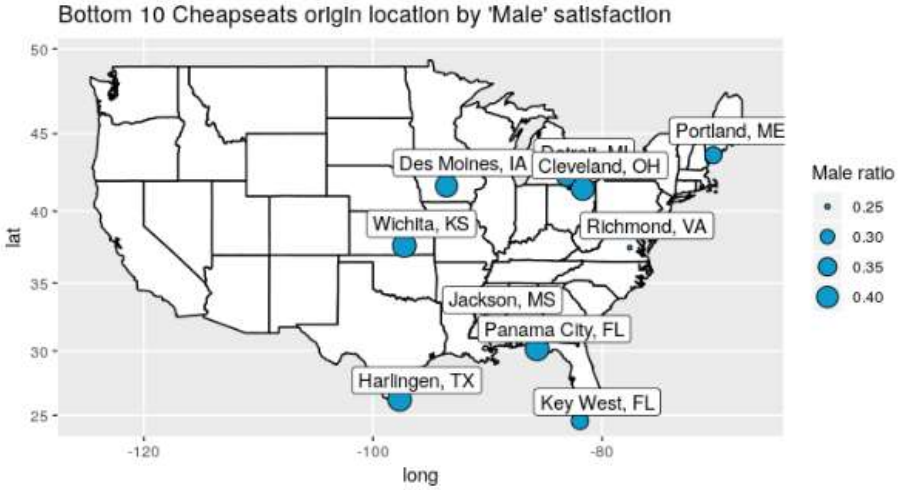
Hide

```
getTopBottomMale <- function(airlineName){
  top_ratings <- df %>% dplyr::select(satisfaction, gender, origin_city_state, origin_state, orig_lat, orig_lng, a
irline_name) %>%  # summarized top ratings by city_state
    dplyr::filter(airline_name==airlineName & gender=="Male" & (satisfaction==4 | satisfaction==5)) %>%
    dplyr::group_by(origin_city_state,origin_state, orig_lat, orig_lng, airline_name) %>%
    dplyr::summarise(count=n()) %>%
    dplyr::arrange(origin_city_state,origin_state, orig_lat, orig_lng, airline_name)

  city_state_freq <- df %>% dplyr::select(airline_name,gender,origin_city_state,origin_state) %>%  # summarized ai
rline flights by city_state
    dplyr::filter(airline_name==airlineName & gender=="Male") %>%
    dplyr::group_by(airline_name,origin_city_state,origin_state) %>%
    dplyr::summarise(count=n()) %>%
    dplyr::arrange(airline_name,origin_city_state,origin_state,count)

  city_state_freq <- city_state_freq[,-1]  # prep for merge
  city_state_freq <- city_state_freq[,-2]

  top_ratings <- merge(top_ratings,city_state_freq, by.x="origin_city_state",by.y="origin_city_state")
  top_ratings$ratio <- top_ratings$count.x/top_ratings$count.y
  top10 <- top_ratings[order(top_ratings$ratio,decreasing = T),] %>% head(10)
  bottom10 <- top_ratings[order(top_ratings$ratio,decreasing = F),] %>% head(10)

  plotTopBottomMale(top10, paste("Top 10",airlineName,"origin location by 'Male' satisfaction",sep=" "))
  plotTopBottomMale(bottom10, paste("Bottom 10",airlineName,"origin location by 'Male' satisfaction",sep=" "))
  return(top10)
}
getTopBottomMale("Cheapseats")
```

	origin_city_state <chr>	origin_state <chr>	orig_lat <dbl>	orig_lng <dbl>	airline_name <chr>	count.x <int>	count.y <int>	ratio <dbl>
40	Little Rock, AR	arkansas	34.74648	-92.28960	Cheapseats	30	37	0.8108108
55	Oklahoma City, OK	oklahoma	35.49161	-97.56282	Cheapseats	47	64	0.7343750
29	Greer, SC	south carolina	34.89825	-82.25785	Cheapseats	11	16	0.6875000
67	Raleigh, NC	north carolina	35.88904	-82.83104	Cheapseats	51	76	0.6710526
9	Boston, MA	massachusetts	42.37057	-71.02696	Cheapseats	52	79	0.6582278
43	Lubbock, TX	texas	33.57786	-101.85517	Cheapseats	19	29	0.6551724
20	Dayton, OH	ohio	39.76419	-84.07615	Cheapseats	9	14	0.6428571
56	Omaha, NE	nebraska	41.29174	-96.17110	Cheapseats	37	58	0.6379310
82	Tucson, AZ	arizona	32.21798	-110.97087	Cheapseats	28	45	0.6222222
78	Seattle, WA	washington	47.60621	-122.33207	Cheapseats	55	89	0.6179775

1-10 of 10 rows





test code to verify dplyr blocks

Hide

```
which(df$airline_name=="Cheapseats" & df$origin_city_state=="Raleigh, NC" & df$gender=="Female" & (df$satisfaction
==4 | df$satisfaction==5)) %>% length()
```

```
[1] 56
```

Hide

```
which(df$airline_name=="Cheapseats" & df$origin_city_state=="Las Vegas, NV" & df$gender=="Female" & (df$satisfacti
on==4 | df$satisfaction==5)) %>% length()
```

```
[1] 422
```

Hide