Membership History - Bang Personal Training (Jan 2018 - Oct 2020)

01/01/2021

INTRODUCTION

OBJECTIVE

Perform an exploratory analysis examining the demographics and membership behavior of past and current Bang Personal Training members. The intention is to observe these factors in relation to membership retention as measured by length of membership before churn as well as measured by retention status at 3 major time points (3 months, 6 months and 12 months). Ultimately, through looking at membership churn over the last two years, I would get an idea of which factors play a significant role in membership retention so as to revamp the on-boarding and membership service operating procedure going forward.

METHOD

Data

For this analysis, I will be using a data set that I had created through the data that was *painfully* collected through the scheduling/payment software Wellness Living, entries from our CRM software Air Table, memory recall based on one-on-one interaction and the email history of all of the past/current members that existed between the period of January 1st, 2018 - October 5th, 2020. However, it is important to note that that this data set had looked at the entire history of our members since their initial start date which could be as early as January 2012. The following information that was collected for this dataset includes:

Table1. Variables within the data set

Variable Name	Description
name	~~Name of member~~ (redacted for privacy purposes)
id	Identifier of the member
age_group	Age grouping for members
employment_sector	Employment sector for member
start_date	First ever day as a member of Bang Personal Training
end_date	Last ever day as a member of Bang

	Personal Training ^*^
length	Total length of days as a member at Bang Personal Training
membership	Predominant membership type as a member at Bang Personal Training
reason_to_leave	Reason to leave Bang Personal Training
churn_type	Type of membership churn for former Bang Personal Training members
lifetime_revenue	Lifetime revenue of a Bang Personal Training member
num_membership_change	Number of membership changes as a member of Bang Personal Training
retention_3m/6m/12m	Retention status at 3 months, 6 months and 12 months
current	Membership status (as of 10-05-2020)
active/former 1x/2x/3x/4x/unlimited/group/distance	Number of months at a given membership type as an active/former member
active/former 1x/2x/3x/4x/unlimited/group/distance rate	Weighted monthly rate at a given membership as an active/former member
num_active_breaks / num_former_breaks	Number of payment cycle breaks as an active/former member
num_active_reups / num_former_reups	Number of membership renewals as an active/former member (per 66 days) ^***^
num_ticket_billing	Number of email-interactions pertaining to billing issues
num_ticket_cx	Number of email-interactions not pertaining to billing/service/scheduling
num_ticket_scheduling	Number of email-interactions pertaining to scheduling requests/changes ^****^
num_ticket_service	Number of email-interactions pertaining to service-related requests ****
total_sessions	Total possible number of sessions that could potentially be attended
attended	Number of attended sessions
cancelled	Number of canceled sessions
lost	Number of sessions lost
pending	Number of sessions with pending status (50:50)
new_month	Was this the month that member started or returned

current_month	Was this the month that member was a member at Bang Personal Training
leave_month	Was this the month that member left Bang Personal Training

- For current members, this "end" will be listed at Oct 5th, 2020

** reupping the membership was based on 66 days as a baseline given its significance with behavior and habit adoption as noted by Lally P, van Jaarsveld CHM, Wardle J (2010). How are habits formed: modelling habit formation in the real world. Euro J Soc Psychol, 40: 998-1009; As long as member had at least 1 month of actual payments, will puy down "1" as an entry but will use analysis to clean this up.

*** This does not include clincal based services like chiropractic, acupuncture, RMT or physiotherapy

**** This includes things like receipts, sending items, putting membership holds and whatever else.

Tools Used

The data set was compiled into a CSV file using Excel and all statistical analyses were conducted using R and R Studio.

Data cleaning

Upon loading this data set into R, the first process was to properly format every variable to its correct data type. Each variable was formatted to the following type:

Numeric: id, length, lifetime revenue, number of membership change, active/former 1x/2x/3x/4x/unlimited/group/distance, active/former 1x/2x/3x/4x/unlimited/group/distance rate, number of breaks for active/former members, number of re-ups for active/former members, number of tickets for service/billing/scheduling/customer experience, total sessions, attended session, canceled sessions, lost sessions and pending sessions.

Categorical: age groups, employment sectors, reason to leave, churn type, retention at 3m/6m/12m, current, new month, current month and leave month

AGE GROUPS

Age was determined based on birthday provided by member at time of signing up at Bang. It was then classified within 5 groups to somewhat match the generation division. These include (a) "Under 18", (b) "18-29", (c) "30-44", (d) "45-64", (e) "65+". For those with unknown age, they are listed as "NaN".

EMPLOYMENT SECTORS

Employment sectors was determined based on Google / LinkedIn search of the member (definitely not creepy at all). Based on area of work, member's were classified into one of the following options that best described their employment: (a) finance/insurance, (b) scientific/academic/educational, (c) technology/information, (d) social services/non-profits, (e) government/legal, (f) advertisement/media/art/culture, (g) real estate/construction/waste, (h) natural resources/energy, (i) manufacturing/trade, (j) transportation, (k) health care/health services, (l) professional/technical services, (m) hospitality/retail/accommodation, (n) student, (o) entrepreneur/owns business and (p) other. For unknown entries, listed as "NaN".

MEMBERSHIP

Seeing as some members had either increased or decreased their frequency, this can be a bit confusing. So in defining the membership of the member, it will be based on which membership the member had been frequently billed out as.

ACTIVE / FORMER MONTH & RATES

In determining the bulk of the lifetime revenue of the member, their weighted monthly averages were determined from the total number of months at a particular membership type. This was further divided between those that were currently active and those that are not. This should be reflected as a numeric variable.

REASON TO LEAVE

Based on email correspondence/exit surveys/CSM entries/memory recall, I've listed reasons for past members deciding to discontinue their membership at Bang based on the following categories: (a) loss of employment – unrelated to any global economic/pandemic reasonings, (b) finance/cost of membership, (c) medical/health-related reasons relating to themselves or immediate social circle, (d) moving away outside of neighbourhood area, (e) lack of accessibility or availability due to prior commitments in life, (f) pursuing other fitness interests, (g) just "ghosted" us, (h) was a time-based arrangement, (i) noted displeasure with Bang's service or experience, (j) pandemic/global economic crisis or (k) anything other thing. For unknown reasons, it is listed as "NaN".

CHURN TYPE

Using Lincoln Murphy's description categorization of membership churn, which was originally used for software-as-a-service industry, former members were classified into one of four categories based on how they've left Bang through their email correspondence, recollection of exit, entries within CSM and other notes.

These categories include: (a) unexpected and unavoidable (i.e. came out of nowhere and really no way of really "saving this"), (b) unexpected and avoidable (i.e. came out of nowhere, but intervention could have been done at any earlier time to have avoided this), (c) expected and unavoidable (i.e. we knew that this was coming for some time but there was no way of preventing this), (d) expected and avoidable (i.e. we knew that this was coming, but could have been addressed earlier in some way to have prevented or at least acted upon it prior to notice)

RETENTION 3M/6M/12M

Often used within the area of clinical addiction research as significant time points for the adoption of a behavior change, these same timelines were used here as a measure of membership retention. With a simple response of either "Yes" or "No", it asks whether a member had continuously been a member at Bang for at least a 3-month, 6-month or 12-month stretch.

CURRENT

This is just a determinant to see if said member is currently a Hybrid Training member currently attending sessions as of October 5th, 2020.

NEW / CURRENT / LEAVE STATUS

These variables were used in unison to determine the membership status from the period of January 1st 2018 to October 5th, 2020. For each month in this period a member was listed as being either: (a) first-ever month at Bang or first month returning as a member at Bang, (b) are they still a member at Bang during this month or (c) did they leave Bang at this month. Using the New/Current/Leave variables, which are either yes or no, there are 8 possible combinations. However, this variable will have the following classifications:

Table2. Combinations of membership status across months/years

NEW	CURRENT	LEAVE	DESCRIPTION
No	No	No	Was not a member during this month
Yes	No	No	First month as a new/returning member
No	Yes	No	Was already a member during this month
No	No	Yes	Last month as a member
No	Yes	Yes	Last month as a member
Yes	No	Yes	Started and Left Bang within the same month
Yes	Yes	No	First month as a new/returning member
Yes	Yes	Yes	Started and left Bang within the same month

From these options, the monthly status variable will classify each individual as being either (a) new/returning, (b) current member, (c) leaving or (d) started and left in the same month.

MEMBERSHIP RENEWAL

In determining the number of renewals, it was operated on the idea that it takes a minimum of 66 days to adopt a health behavior change, based on a study by Lally et al.

2010. Despite our memberships are month-to-month, which would make sense to account as every month of continuation = 1 renewal, I have set it up so that renewals were all based on 2-month intervals throughout instead. Now the troublesome part comes with those that undergone less than 66 days (i.e. only stayed for one month). For those members, the renewals will now be set as 0. So, going through fixing this variable, I made the cut-off that those that engaged in less than 66 days will be listed as 0.

NUMBER OF PAYMENT BREAKS

As it was inevitable that there were cases where some members will have a break in their payment cycle due to various reasons. So we want to also take note of the number of times that this has happened. This will be recorded as a numeric.

MISSINGNESS

In making the data set, there were cases where I was not able to attain information on certain demographic variables (on a side note: this was pretty creepy and also scary to think that anyone can find stuff on you). So will create a variable to identify those with or without these pieces of information. This will be used in determining how we will be able to handle missing variables for our analysis.

START DATE & END DATE

I will be converting the information on start + end date of membership for each member. However, it should be noted that for the purpose of this analysis, those that are current members will have their end date listed as Oct 5th, 2020.

NUMBER OF SESSIONS

In order to calculate the attendance of the member, I've collected information pertaining to their attendance history such as number of attended sessions, number of cancelled classes (as confirmed on the scheduling software Wellness Living), number of lost session (based upon the projected theoretical total of sessions that could've been attended) and number of unconfirmed attendance. These variables should be reflected as numeric variables.

ATTENDANCE & CANCELLATION RATE

This variable is a numeric variable that sums up the attendance history of the member based on the following formulas: (a) Attendance rate = (Attended + (1/2 pending)) / Total Sessions (b) Cancellation rate = (canceled + lost + (1/2 pending)) / Total Sessions

The use of the "pending" items is based on the theoretical likelihood that there is a 50% chance that the member attended the appointment or not have attended the appointment.

(FUTURE MIKE) When doing preliminary analysis, I realized that I will be handling non-normally distributed data. Considering how this would cause violations in certain regression analyses, I will need to transform this data through multiple means to allow these assumptions to hold.

EMAIL INTERACTIONS

In assessing the sort of interactions that membership service teams have with members, I've exclusively used Emails as that was the only thing that had a paper trail. By paper-trail, I basically mean that counting the number of a particular email interactions from the inbox. I've categorized this into 4 categories:

Table3. Types of Email Interactions

Email Type	Description
Billing	Refers to any email interaction relating to any inquiries or events of unknown charge and/or billing errors
CX	Refers to Customer Experience interactions that relate to check-in or anything not related to providing service or rescheduling
Scheduling	Refers to any email interactions that pertain to request for rescheduling or scheduling appointments (Hybrid Training Only)
Service	Refers to any requests or inquiries pertaining to administrative tasks, memberships, etc. that isn't relating to scheduling

Realizing that some folks may not have any other emails types than one particular kind, I will look at this in terms of percentage of total emails.

(FUTURE MIKE) I realized that billing-related emails isn't really great in terms of treating it as a percentage since it's fairly rare occurrence and will throw off the analyses downstream. So, I will instead create two new variables to break this aspect down and separate it from the inclusion of email types. I've also rearranged percentage to only assess CX, Scheduling and Service-related emails. This will lead to having a new sum total of emails which include all types EXCEPT for Billing-related emails. Lastly, I realize that the total number of these emails may not be as useful considering that it's an absolute number. Thus, I've created a measure of mean number of emails per month.

MONTHLY MEMBERSHIP RATES

In order to determine the monthly rates of a member, it will be based on the weighted average of monthly rates across each membership type that the member had engaged in.

(Future Mike here) Realized that I will run into issues with making regression models due to issues of assumptions not being met (like linearity, proportionality, etc.) So to correct for these issues, I will try to categorize monthly rate in the same way that I had done earlier with attendance rate.

At some point in analysing the distribution of certain data, namely the non-billing related emails. There happens to be some evidence of zero-inflation, which is to be expected since some folks are just not the email type of folks. Seeing as this will inevitably give me some suspect findings down the road, I'll use the advice found here and here to have **both** a continuous variable of percent composition of a specific non-billing type of email interaction along with a variable that dictates whether there is a particular type of email interaction or not. Hopefully, this will give some clarity on the impact of that variable once we get into regression analyses.

Creating Multiple Data Sets

In order to be observe and perform our analysis the way that is intended, I've divided the data across multiple data sets.The initial data set will only include the necessary data for our analysis. This will be considered our "final" data set that includes those with missing demographics variables. I've also made a series of data sets based on the month and year.

HANDLING INCONSISTENT FINDINGS & MISSINGNESS

Now, as this data set was compiled by me, there would definitely be typo errors, missing values and irregularities with some of the entries. Using both Wellness Living and Air Table as a final check for entries in the data set, adjustment were made in filling out incorrect entries. As for unknown variables, they only pertain to demographics (i.e. age and/or employment sector). Looking at the make up of this group, it seems to account for 7.45% of the entire data set. While the best bet is to just drop this subset, there could potentially be some bias introduced as a result. So, the plan is to create an entire subset of this data and compare its descriptive findings with the non-missing data to assert if there is any noted difference. However, for the sake of performing inferential statistics, we will only keep entries with no missing variables in the age and employment sector.

CREATING MULTIPLE DATA SETS

Additional data sets will be created that are divided based on:

(1) only Hybrid vs Group vs Distance
(2) former vs active members
(3) age groups
(4) employment sectors
(5) membership types
(6) retention status at 3/6/12 months
(7) reasons to leave Bang Personal Training
(8) churn type
(9) membership status at a given month/year
(10) monthly membership rates
(11) attendance rates

I've also created pivoted data sets which stack related data together to get a more comprehensive look at the distribution of data. These include types of email interactions, monthly membership updates, etc.

(Future Mike Here) I will also be creating a separate data set that will contain only the necessary variable that would be used in formulating models based on our data set.

However, knowing that most of these variables will be not normally distributed and this will cause problems down the line, I will also be doing some data transformation in order to be able to build models that meet their own assumptions.

Now since I will be log transforming these data, there will be issues with entries of 0 which creates an undefined entry. A workaround that I've chosen to do is to add a constant that with a magnitude that is small enough as to not influence the overall impact of said variable. In this case, we will be doing the following:

- weighted average monthly membership rate will add 0.1, which is the equivalent of $0.10
- attendance rate will add 0.1, which is the equivalent of 0.1%
- total non-billing email interactions will add 1, which is the equivalent of 1 email
- percentage of total emails pertaining to CX-/scheduling-/service-related email interactions will add 0.1, which is the equivalent of 0.1%
- number of email-interactions per month will add 0.1, which is the equivalent of 0.1

Analysis Plan

DESCRIPTIVE STATISTICS

Based on the data type, we will use either mean +/- SD or median to summarize the distribution of a given variable. However in the case of non-normal distributed data, we will instead be using median. This will be displayed on the appropriate medium. Primarily, we are interested in observing the distribution of demographics (i.e. age + employment sector), attendance rates, number of breaks email correspondence, etc. In comparing differences b/t groups, Student's T-test will be used for continuous variable whilst the Pearson's Chi-Square test will be run for categorical variables with significance cut-off set at p = 0.05. ANOVA will be used for cases where there are more than 3 groups for analyzing differences in continuous variable between groups with the Holm correction for pairwise adjustments. For instances of non-normal distribution, the Mann-Whitney Tests, Kruskal Wallis test (w. Holm correction) will be used instead.

INFERENTIAL STATISTICS

In order to assess the influence of various data gathered on membership churn, I will be looking to use a Cox Regression model in order to gain insight on how these factors play on length of membership prior to leaving. This will be determined through two methods: (a) random survival forest as well as (b) Cox regressional analysis. Additionally, the infuence of various predictors on retention status at 3-, 6- and 12-months were also examined through both (a) logisic regression analysis and (b) random forest modelling. The selection of predictors for the linear modelling approach (i.e. Cox-regression + logistic regression) were determined via stepwise regression using AIC as the measure for determining variable retention.

RESULTS

Overall

Looking at the overall number of members over the period of Jan 2018 to Oct 2020, there were 483 members (98 current vs. 385 former). The majority of our members existed b/t the ages of 30-44 & the 45-64 age group. Notably, most of the members came from the advertising/media/culture/art, technology/information, professional/technical services and financial/insurance sectors. The most popular membership types were the 2x/week and 3x/week Hybrid Training memberships. As it pertained to the length of membership, the median duration is 121 days (i.e. approx. 4 months) with the average monthly membership rate of ~ $350. The average attendance rate for our members being approximately 60~ish% which isn't surprising as the majority of our former members cited the lack of availability or accessibility as a reason for leaving Bang Personal Training. However, it is important to recognize that the pandemic played a noticeable role in loss of membership as noted by the drop in membership in March 2020.

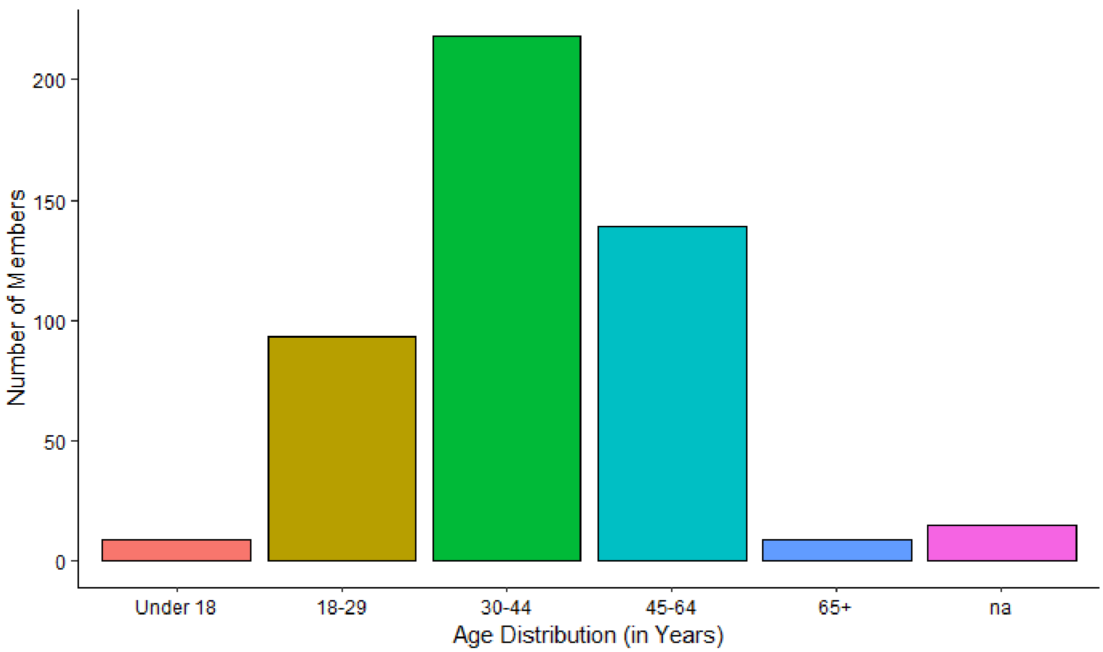


Figure1. Distribution of Bang Personal Training Members Across Age Groups

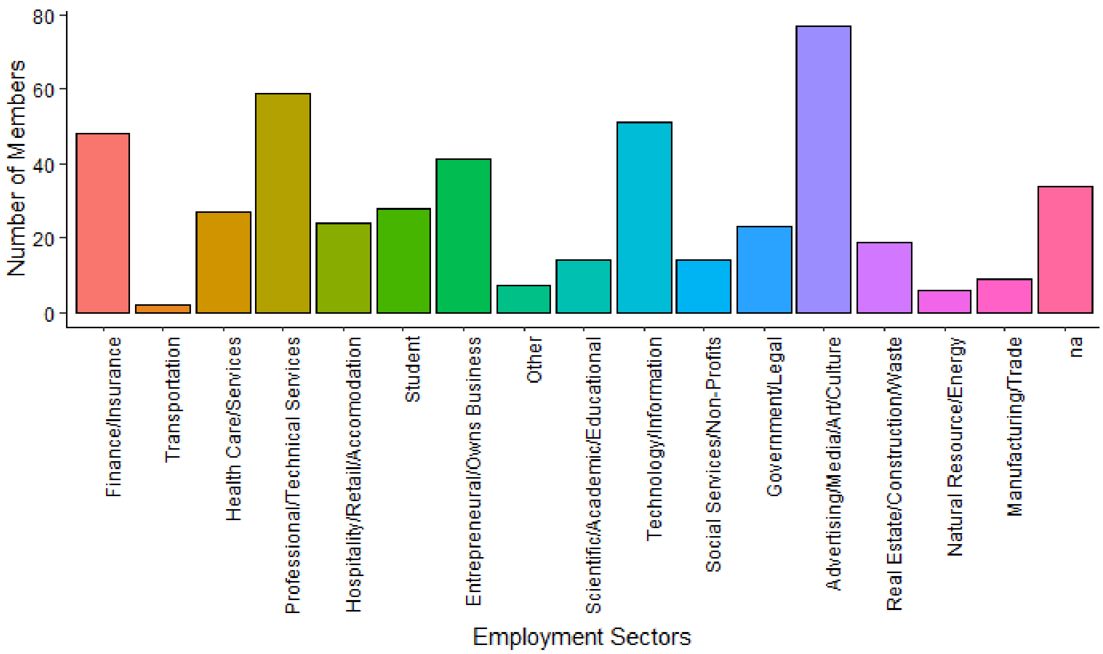


Figure2. Distribution of Bang Personal Training Members Across Employment Sectors

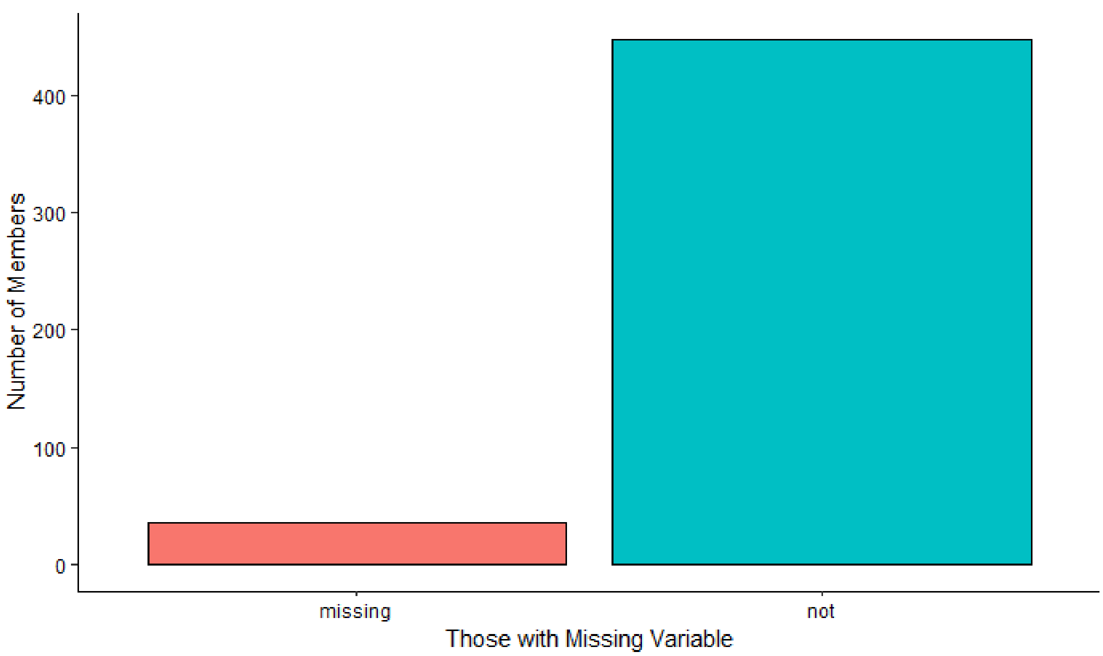


Figure3. Number of Bang Personal Training Members with Unidentified Demographic Variables

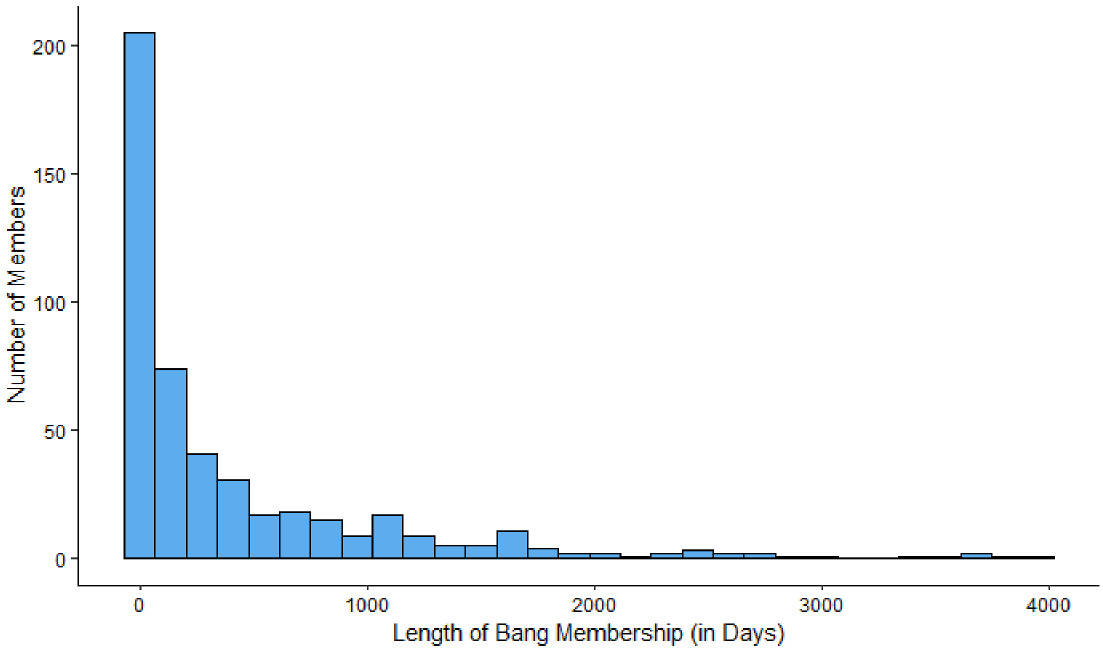


Figure4. Length of Membership for Bang Personal Training Members

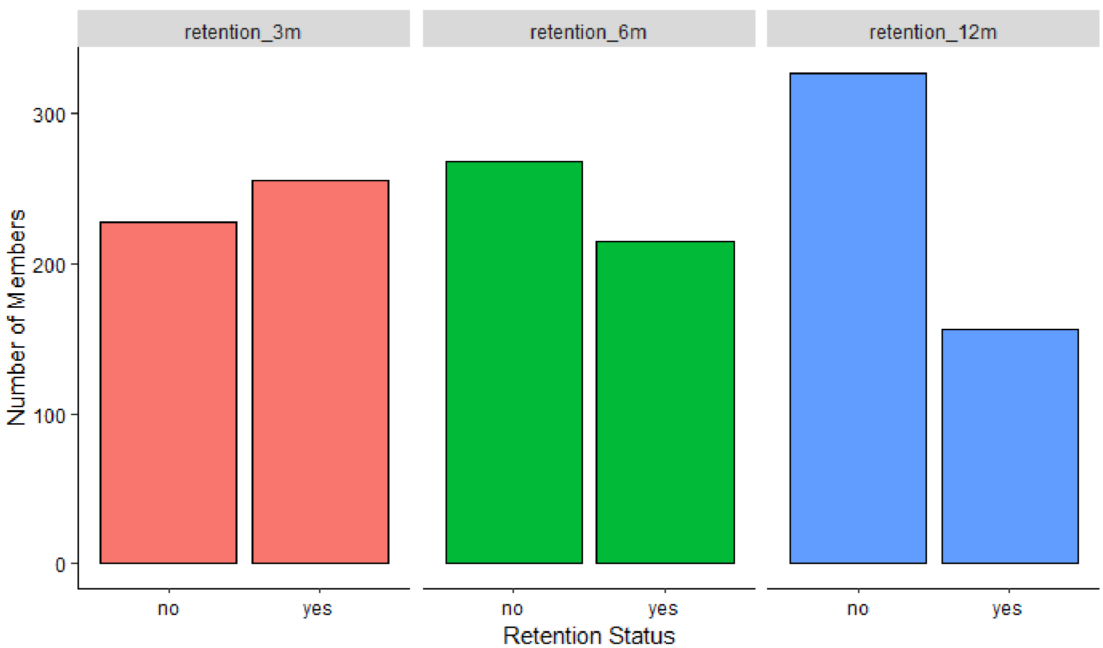


Figure5. Continuous Retention Status for Bang Personal Training Members at 3-Months, 6-Months and 12-Months

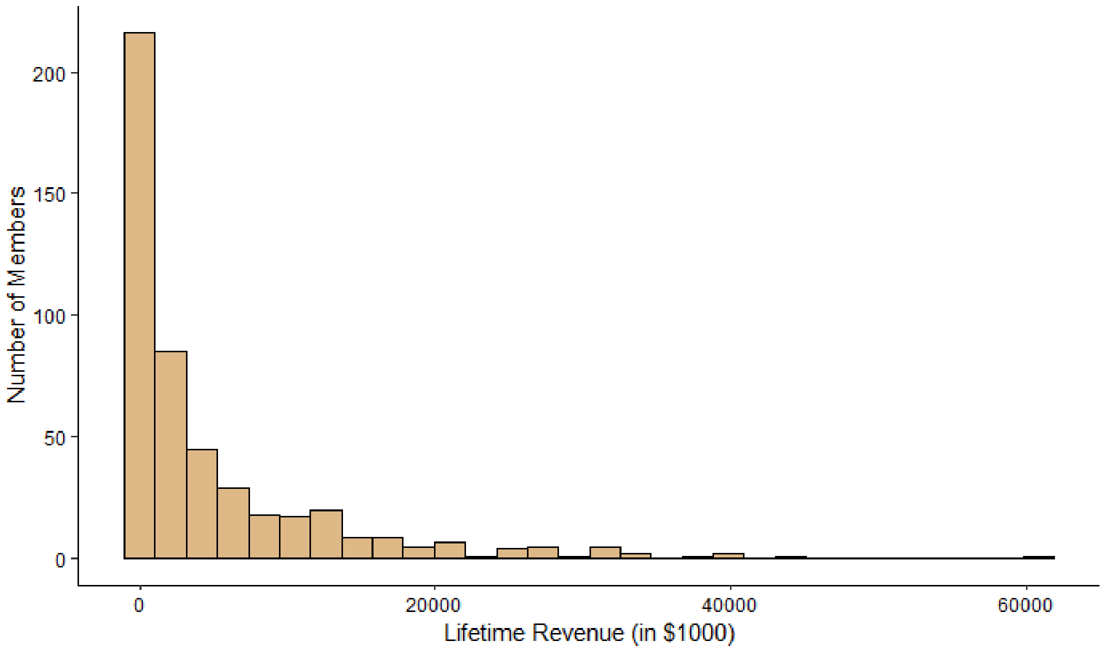


Figure6. Lifetime Revenue of Bang Personal Training Members

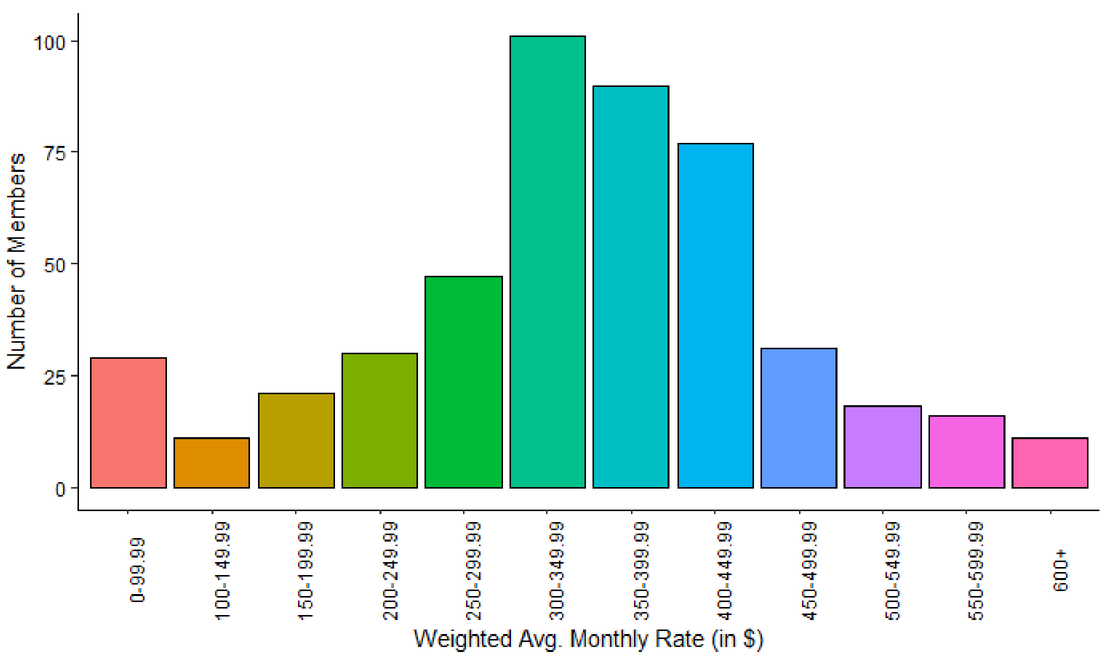


Figure7. Bang Personal Training Members Grouped by Weighted Monthly Rates

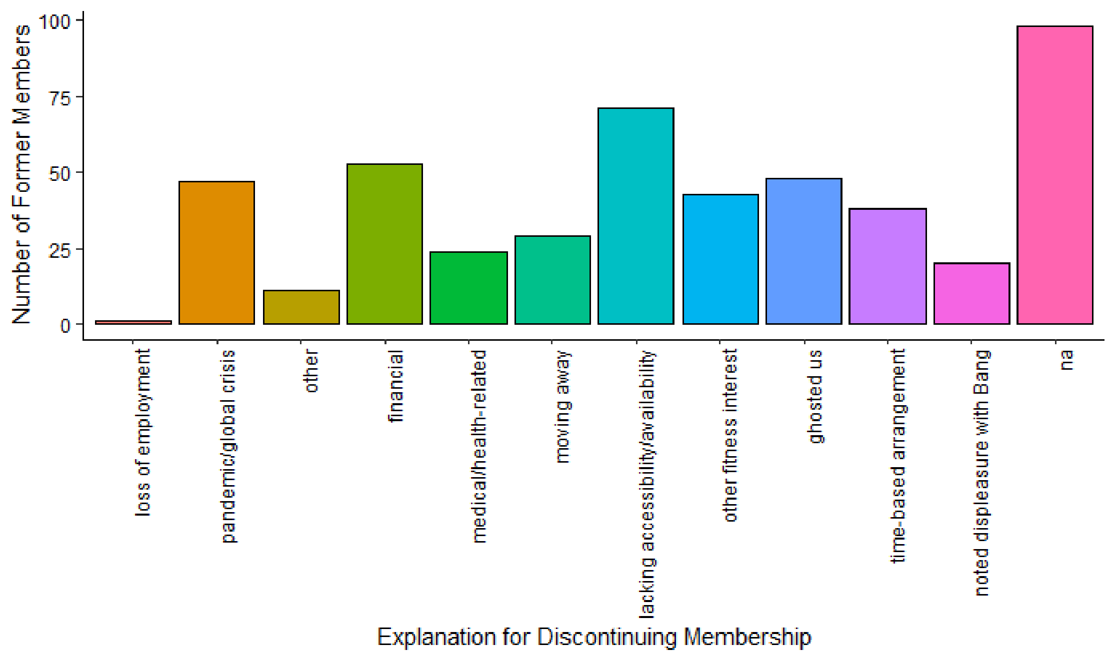


Figure8. Reasons for Discontinuing Membership Amongst Former Bang Personal Training Members

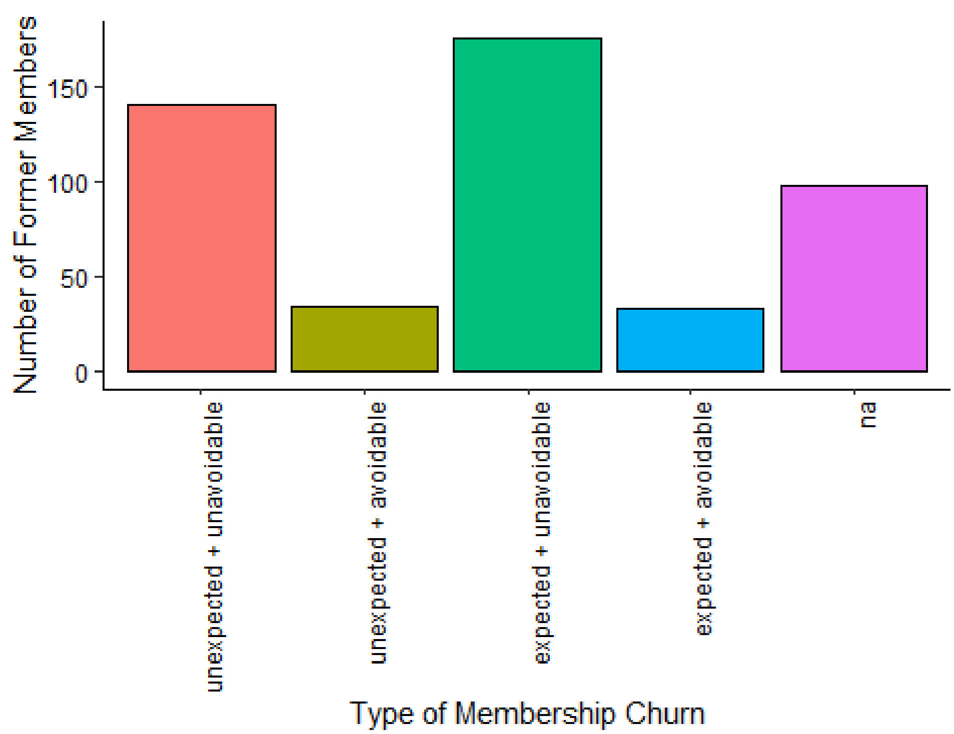


Figure9. Churn Type of Former Bang Personal Training Members

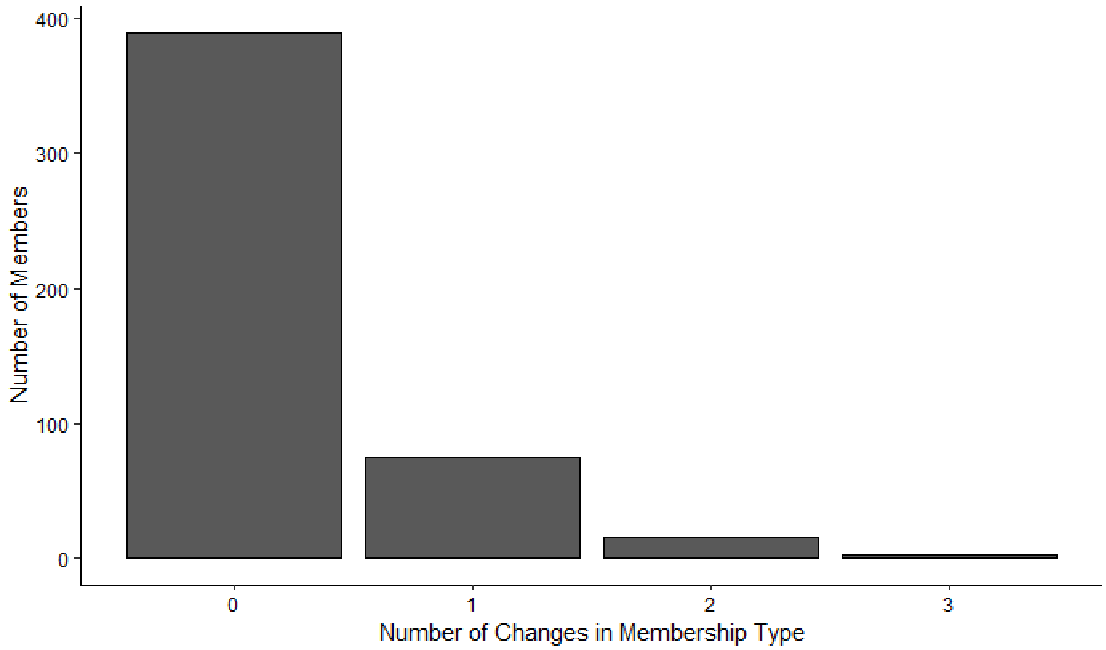


Figure10. Number of Membership Changes Amongst Bang Personal Training Members

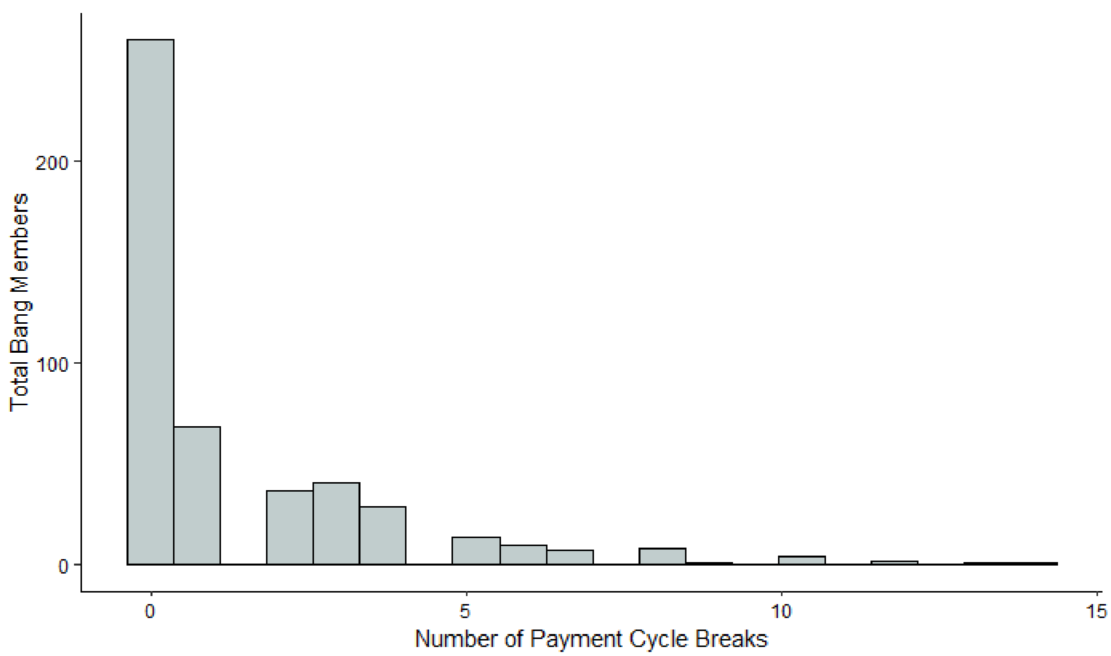


Figure11. Number of Payment Cycle Breaks Amongst Bang Personal Training Members

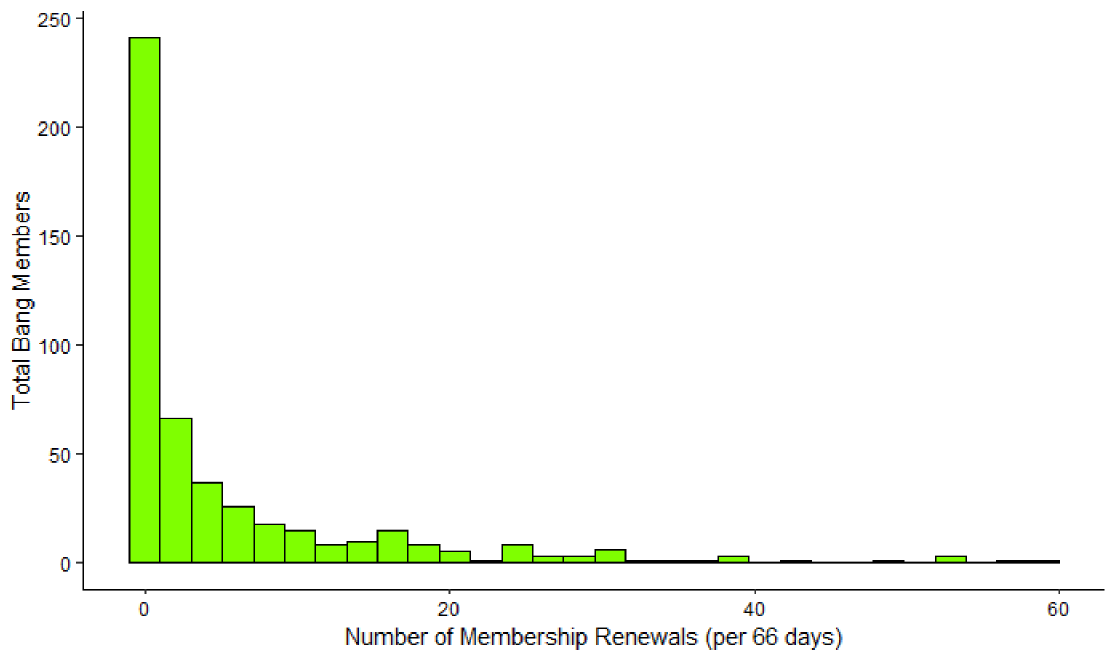


Figure12. Number of Membership Renewals Amongst Bang Personal Training Members

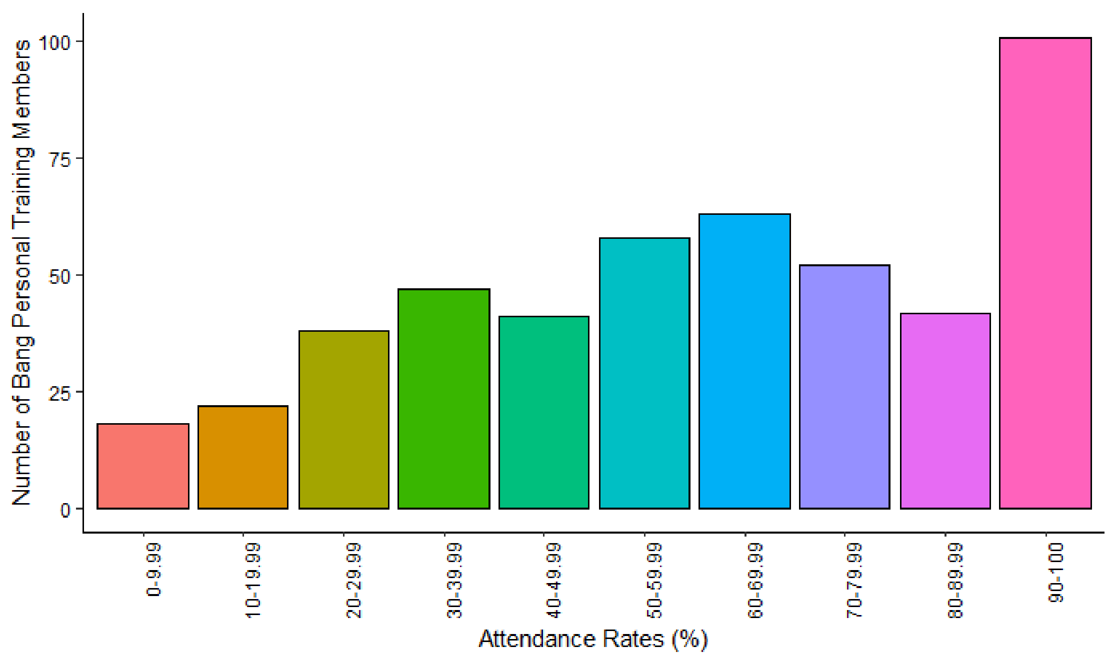


Figure13. Bang Personal Training Members Grouped by Attendance Rate

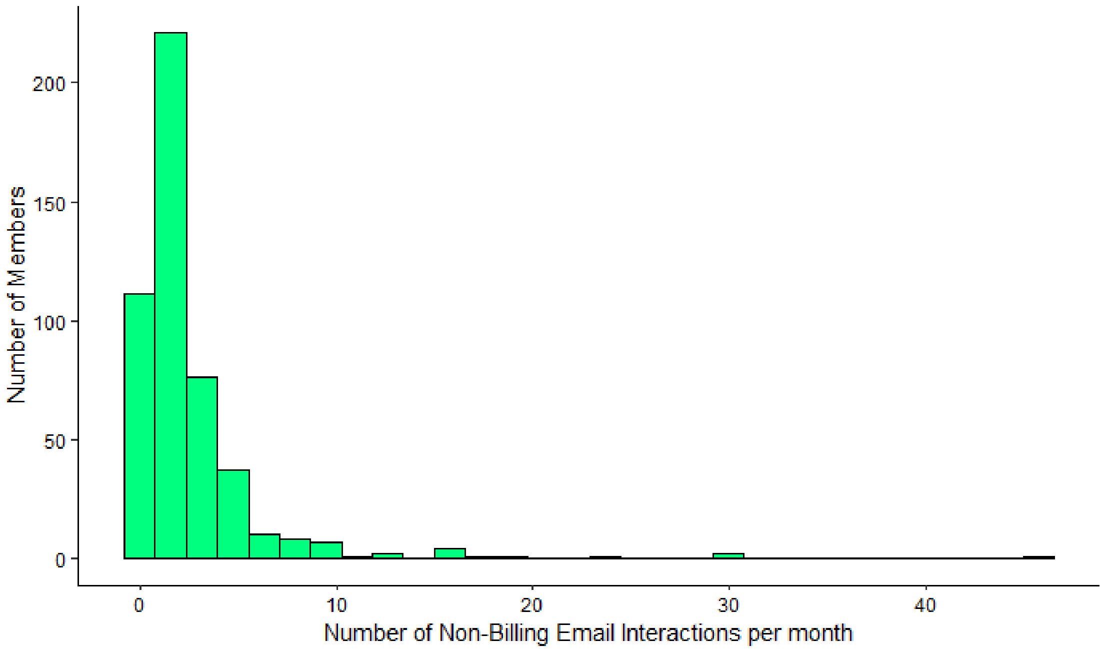


Figure14a. Number of Non-Billing Email Interactions Amongst Bang Personal Training Members

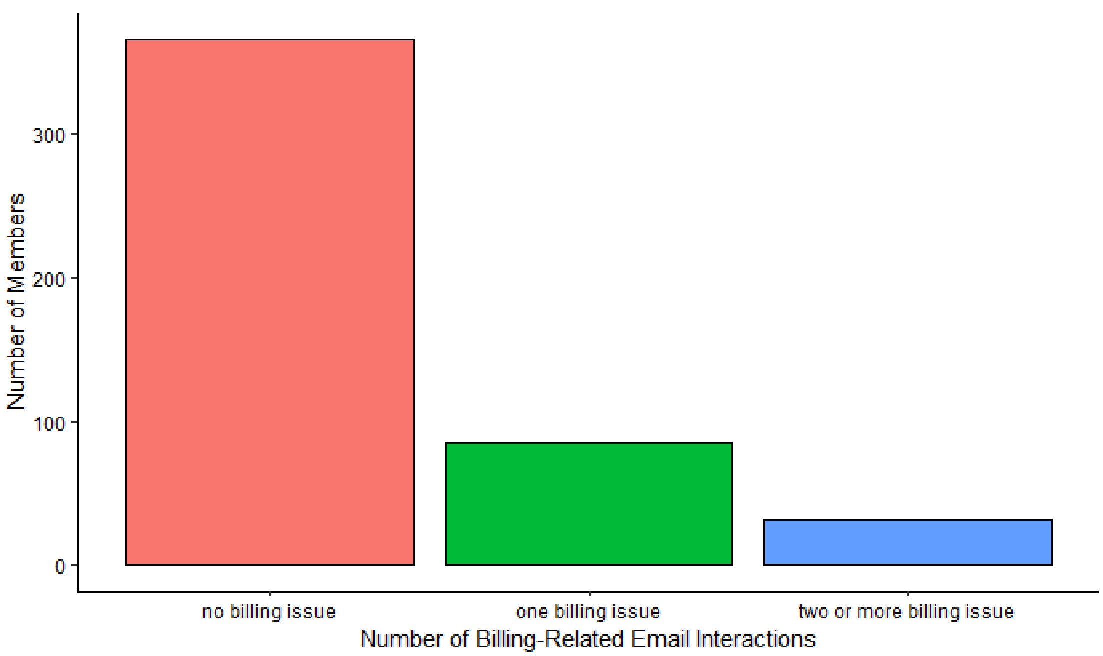


Figure14b. Number of Billing Email Interactions Amongst Bang Personal Training Members

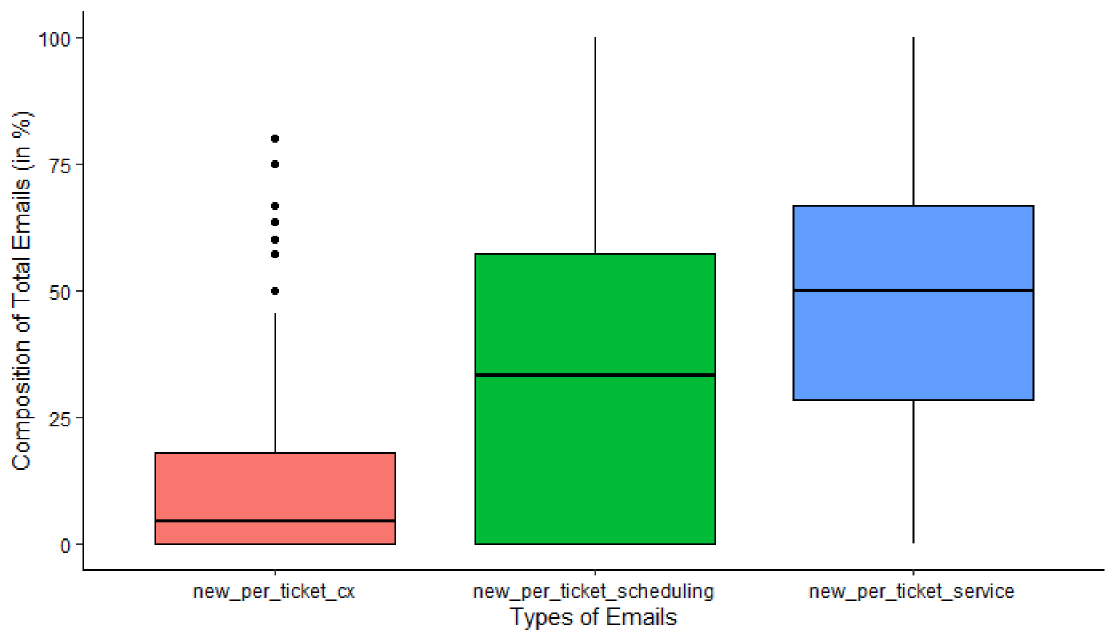


Figure14c. Percent Composition of Types of Email-Interaction Between Bang Personal Training Members and Membership Service Staff

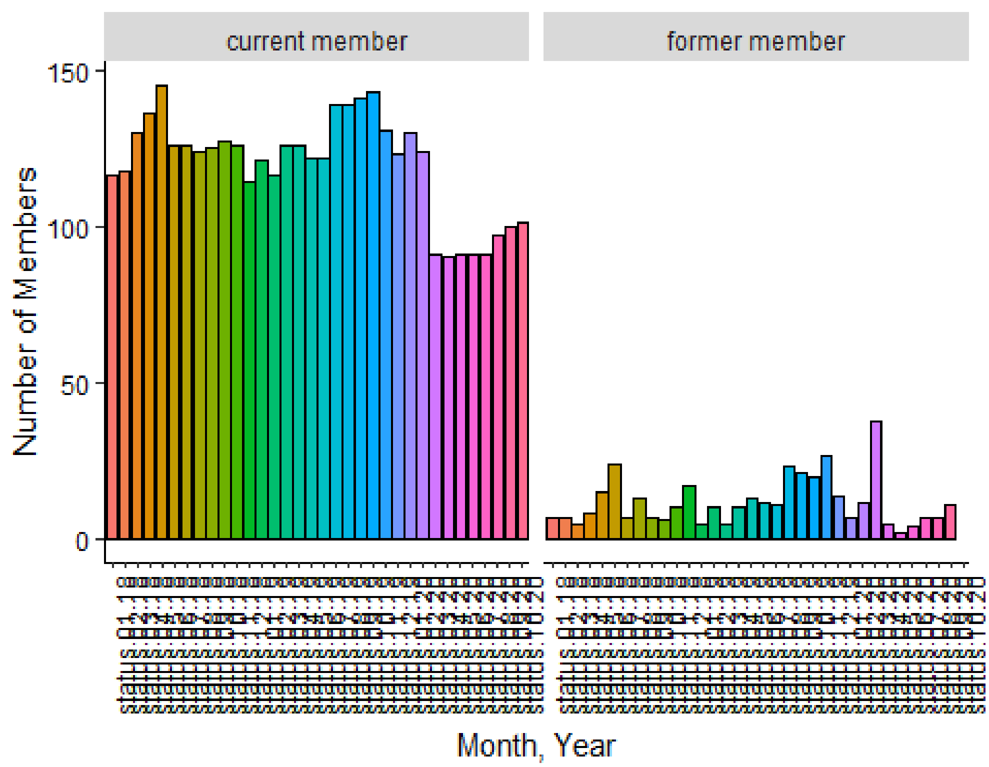


Figure15. Membership Count from January 2018 - October 2020

Impact of Age, Employment and Membership Type

(NOTE: Going forward, we will be using the data set **WITHOUT** entries with missing demographic variables)

Looking further into the distribution of members, there was significant differences observed with respect to Age x Employment Sector and Age x Membership Type. While there were many differences found, it was notable to find that the majority of those within the 30-44 crowd were from the Technology/Information as well as Professional/Technical Services sector whilst those in the 45-64 crowd were the predominant age group within the finance/insurance sector. Additionally, although the 30-44 crowd were the predominant age group across most membership types, those in the 45-64 crowd were actually the predominant age group for distance coaching.

Looking at the impact of age, membership and employment sector on attendance rate, we see that attendance rate varied significantly with respect to age and employment sector, employment sector and membership type as well as membership type and age. Notably we see that:

- Greater attendance rates among those with 2x/week membership across various age groups relative to other membership types
- Group membership were predominantly those within the 30-44 age category
- Those within the Technology/Information sector had the highest attendance rates relative to all other employment sector; the lowest were those within the hospitality/retail/accommodation sector
- Across age groups, we see those within the entrepreneurial space having the lowest median attendance rate.
- Lowest length was found in the Social Services/Non-Profit + Hospitality/Retail/Accommodation sector across age and membership types
- Highest overall across age and membership types were noted amongst those in the Entrepreneurial and Tech sector (particularly at age 30-44); interestingly those within the Health Care sector had a very high length of membership for those aged 30-44
- As it relates to membership types, 2x/week had the highest length of membership across age + employment sector.

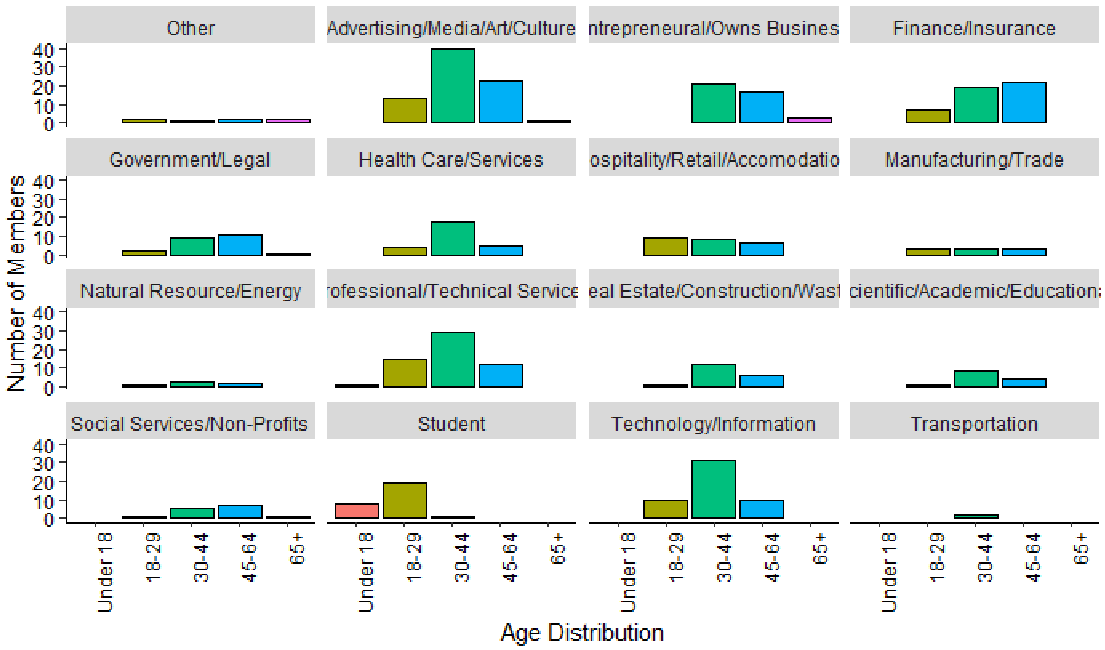


Figure16. Number of Bang Members by Age Group and Employment Sector (χ2 = 251.52, p < 0.001)

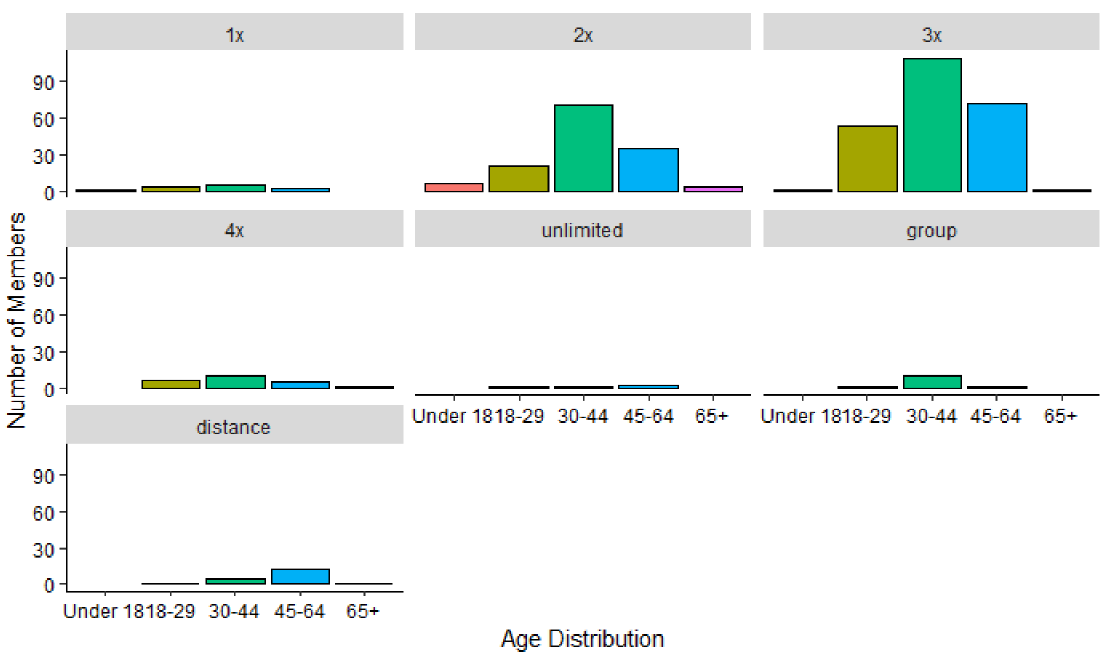


Figure17. Number of Bang Members by Age Group and Membership Type (χ2 = 36.85, p = 0.045)

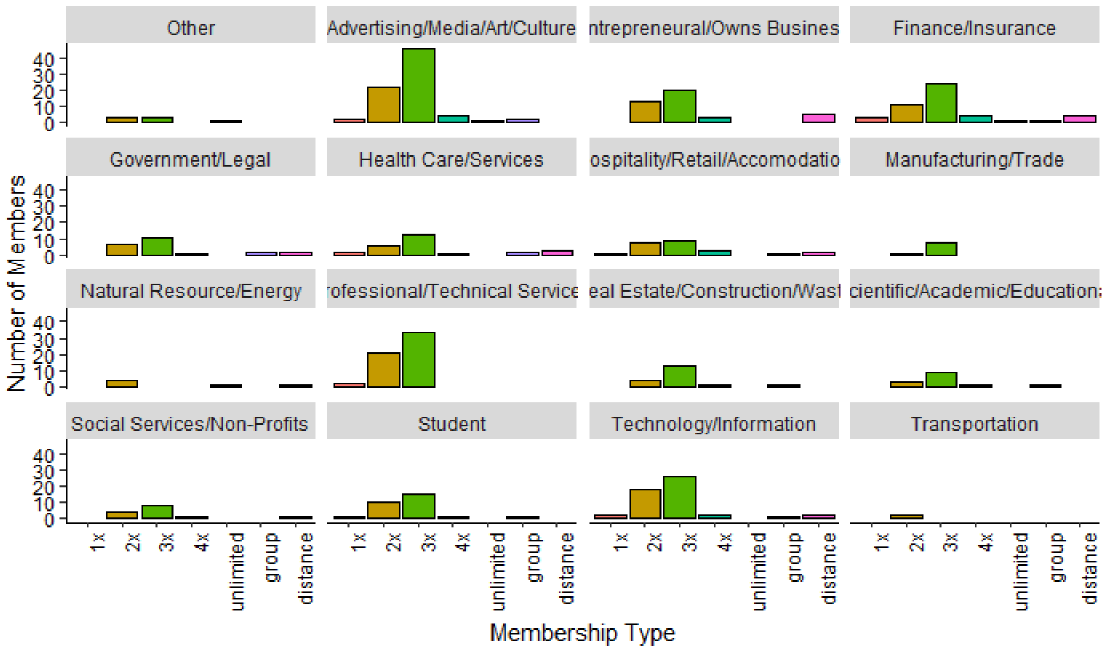


Figure18. Number of Bang Members by Membership Type and Employment Sector ($\chi2$ = 101.73, p = 0.187)

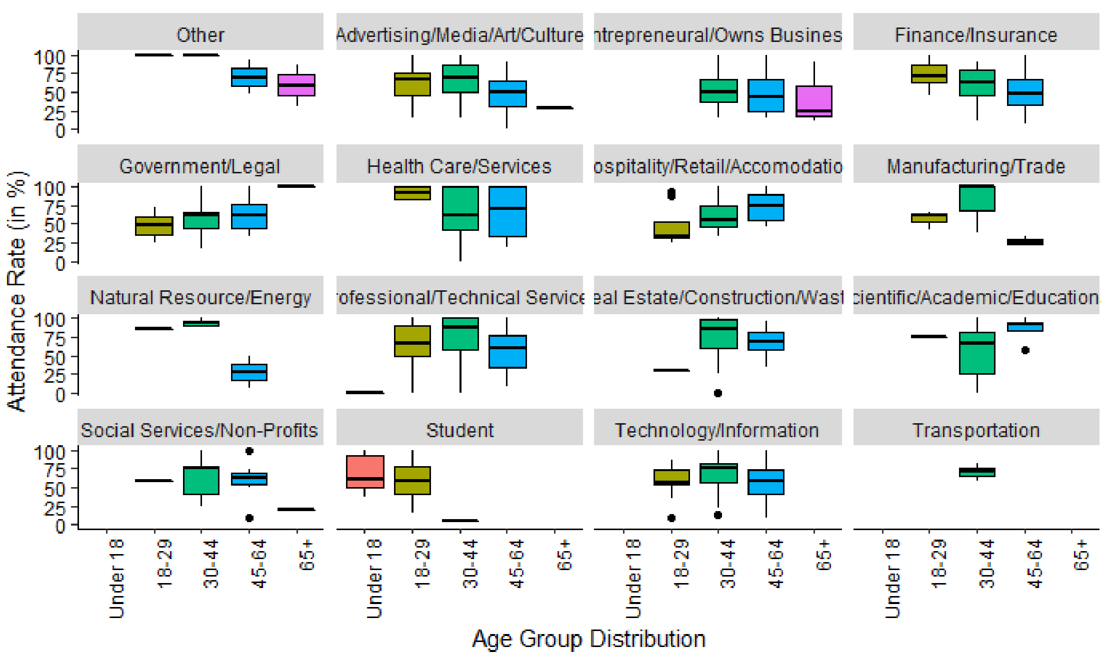


Figure19. Attendance Rate of Bang Members by Age Groups and Employment Sector

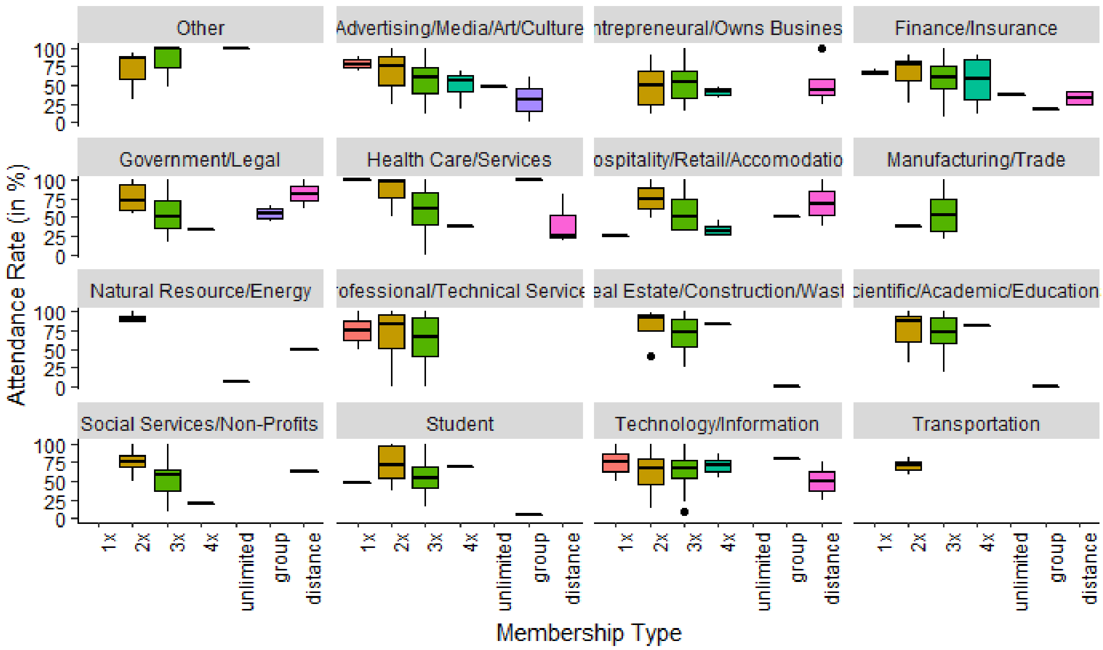


Figure20. Attendance Rate of Bang Members by Membership Type and Employment Sector

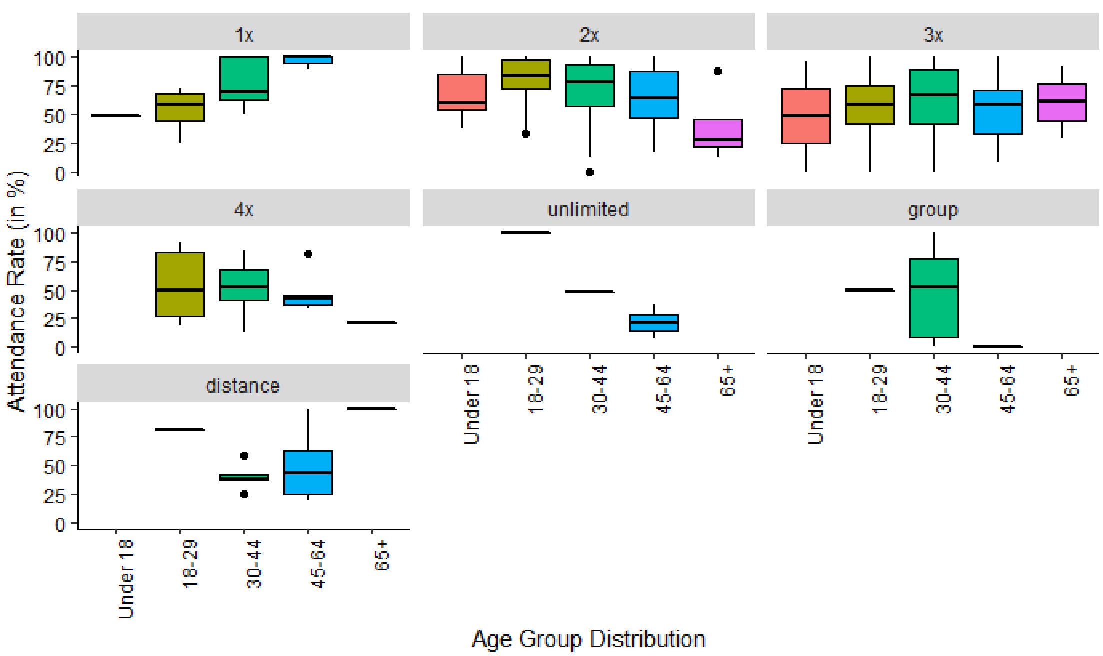


Figure21. Attendance Rate of Bang Members by Age Groups and Membership Type

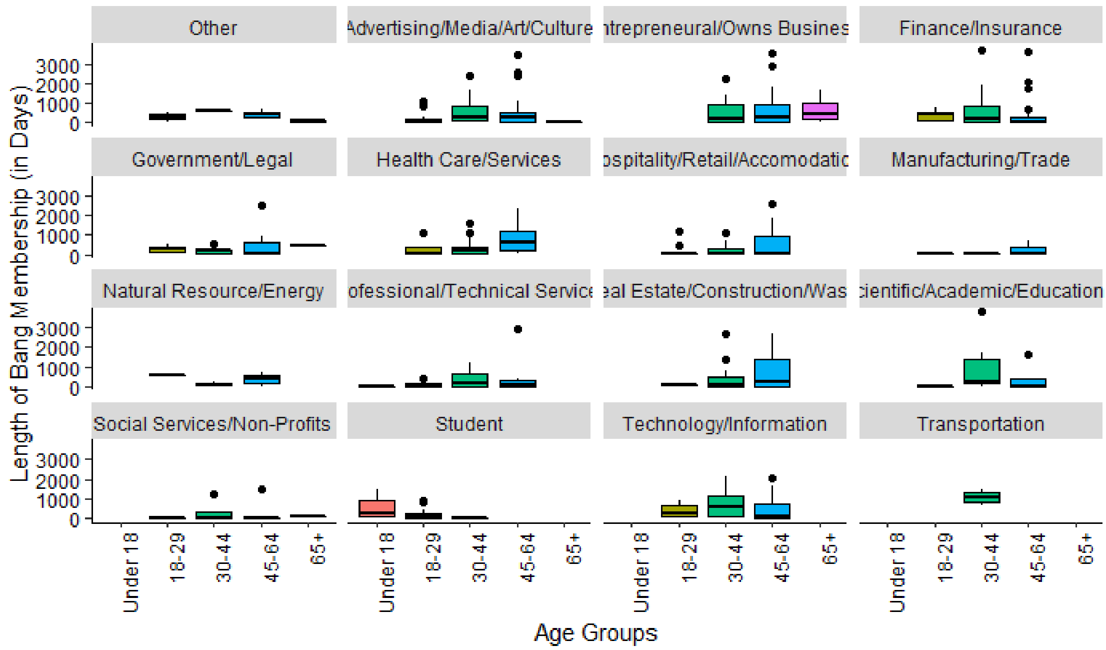


Figure22. Length of Membership by Age Groups and Employment Sector

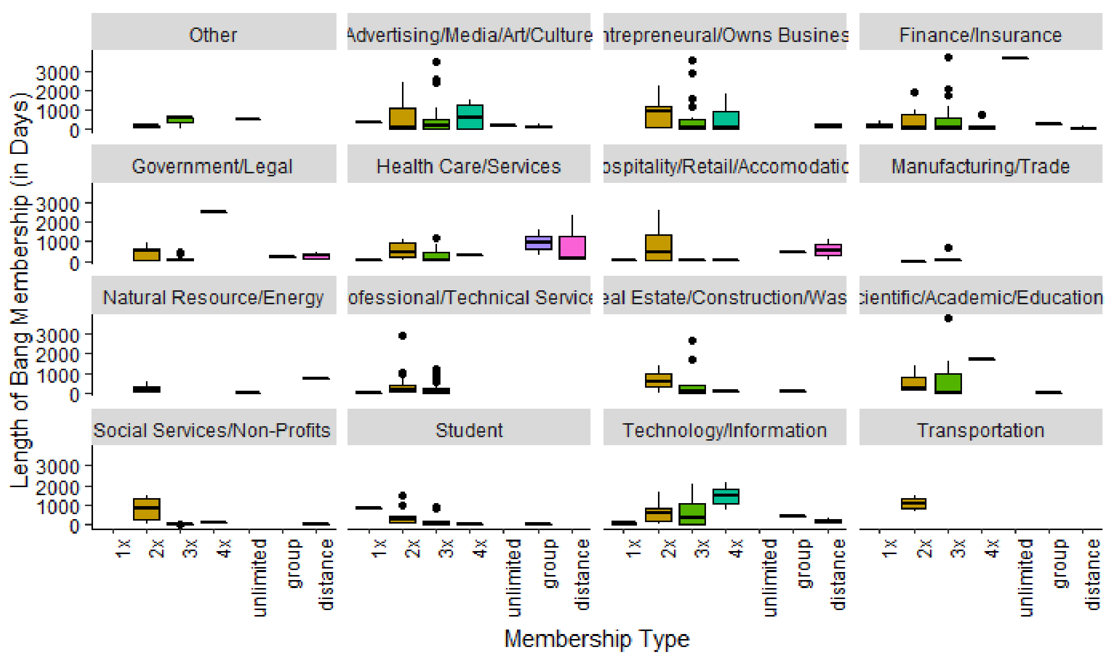


Figure23. Length of Membership by Membership Type and Employment Sector

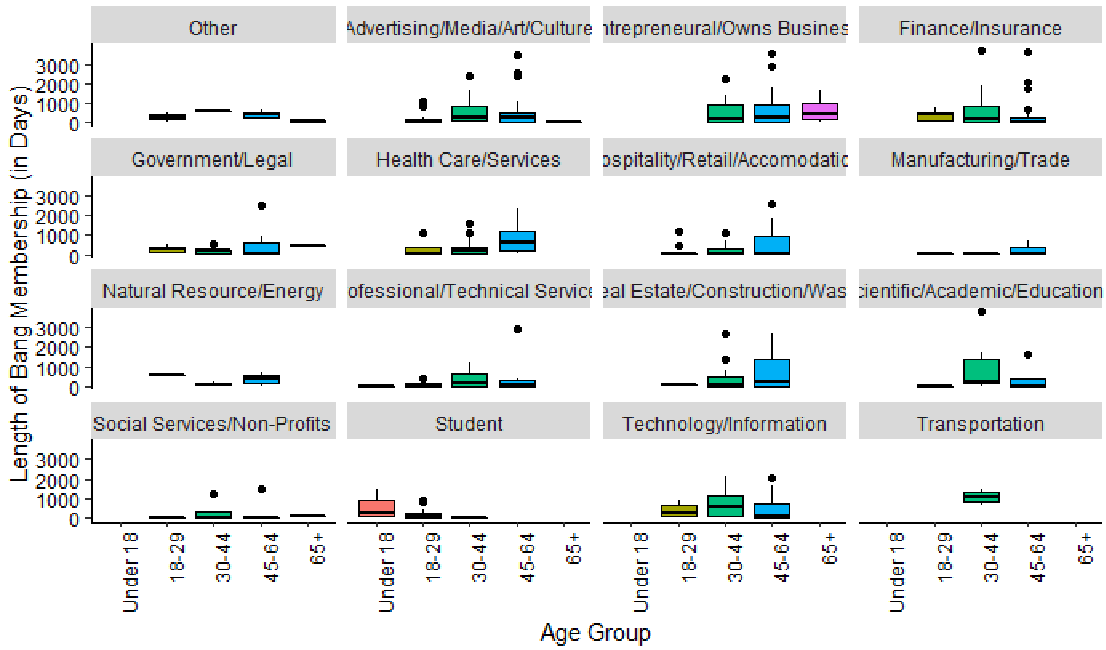


Figure24. Length of Membership by Age Group and Membership Type

Current vs Former Members

It was found that there was roughly and equivalent split of current members across 3 age categories (18-29, 30-44 and 45-64) for the most popular membership type (3x/week) between current and former members. However, overall, 30-44 was the most predominant age-group. Consistent with the overall data, the most common sector have been those within the advertisement/art/media/culture sector.

There were difference in the number of breaks in payment cycle as it was found there was significantly greater number of payment cycle breaks amongst current members than with former members. Additionally, the number of renewals were found to be significantly higher among active members than former members. However, there were no differences with respect to attendance rates or with number of membership changes b/t current and former members.

As it pertains to weighted average monthly membership rates, there were no difference b/t current and former members. However, once average monthly rates were categorized, it was found that those that a significantly greater proportion of active members have a higher monthly membership rate than former members.

Looking at email interactions, it was shown that those that are active members were more significantly more likely to have reported ever having a billing-related issue as compared to former members. Similarly, those that were current members were also found to have significantly greater percentage of their email interaction to be related to scheduling requests as compared to former members. Interestingly enough, while not statistically significant, those that were former members had a greater number of service-related email interaction as compared to former members. This relationship was also noted with respect to the number of non-billing related email interactions per month as those that were current members reported significantly less email interactions as compared to those that were former members.

Figure25. Number of Bang Members Based on Membership Status (as of 10-05-2020)

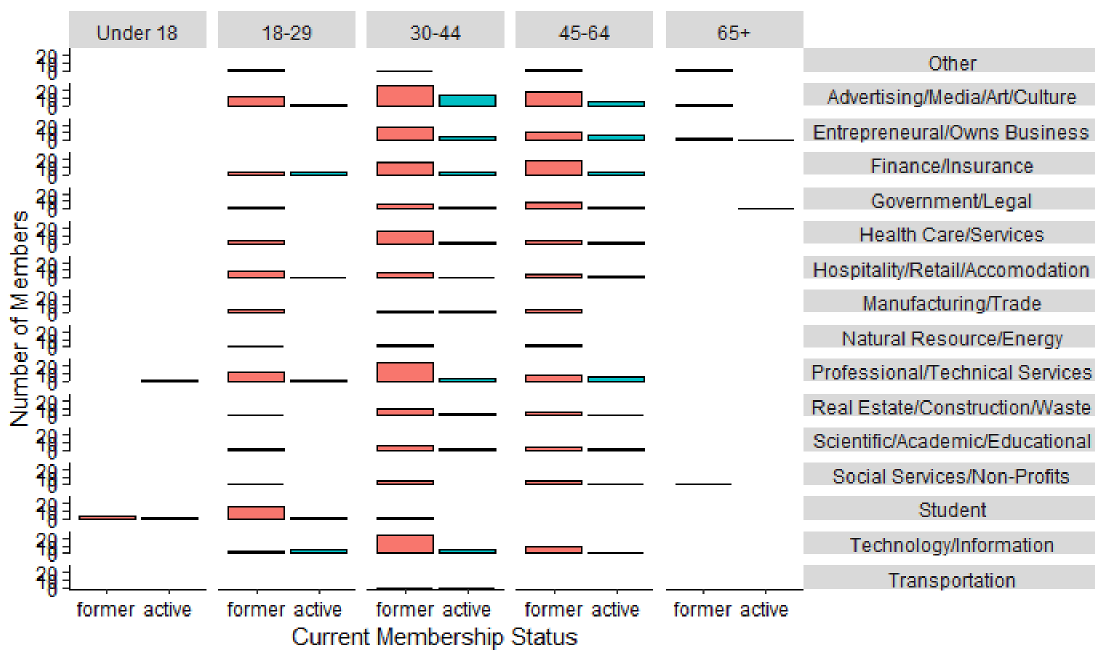


Figure26. Membership Status of Bang Personal Training Members by Age and Employment Sector

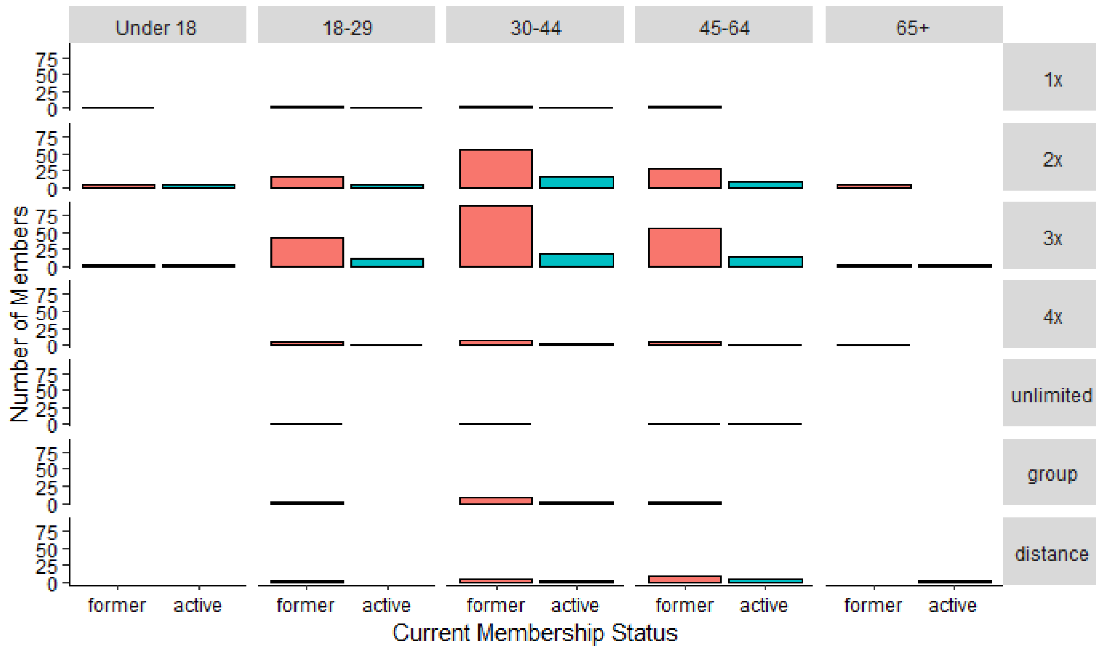


Figure27. Membership Status of Bang Personal Training Members by Age and Membership Type

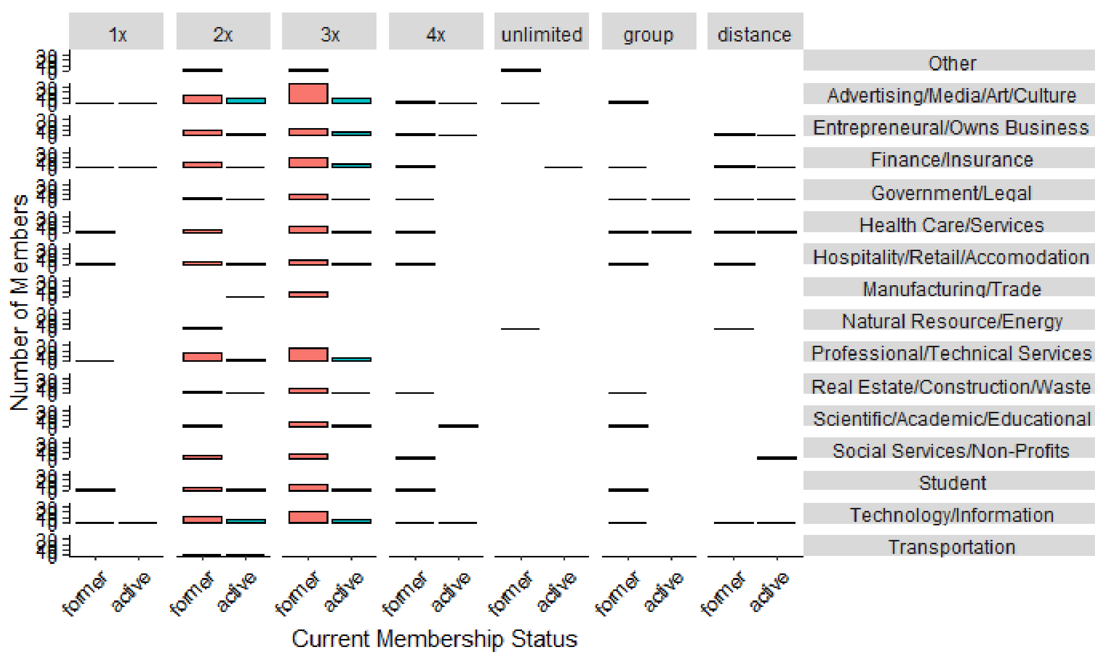


Figure28. Membership Status of Bang Personal Training Members by Employment Sector and Membership Type

```
shapiro.test(clean_bang_final$num_breaks) # Not a normal distribution

##
##  Shapiro-Wilk normality test
##
## data:  clean_bang_final$num_breaks
## W = 0.7006, p-value < 2.2e-16
```

```r
wilcox.test(num_breaks ~ current, data = clean_bang_final)

##
##  Wilcoxon rank sum test with continuity correction
##
## data:  num_breaks by current
## W = 10640, p-value = 6.27e-09
## alternative hypothesis: true location shift is not equal to 0
```

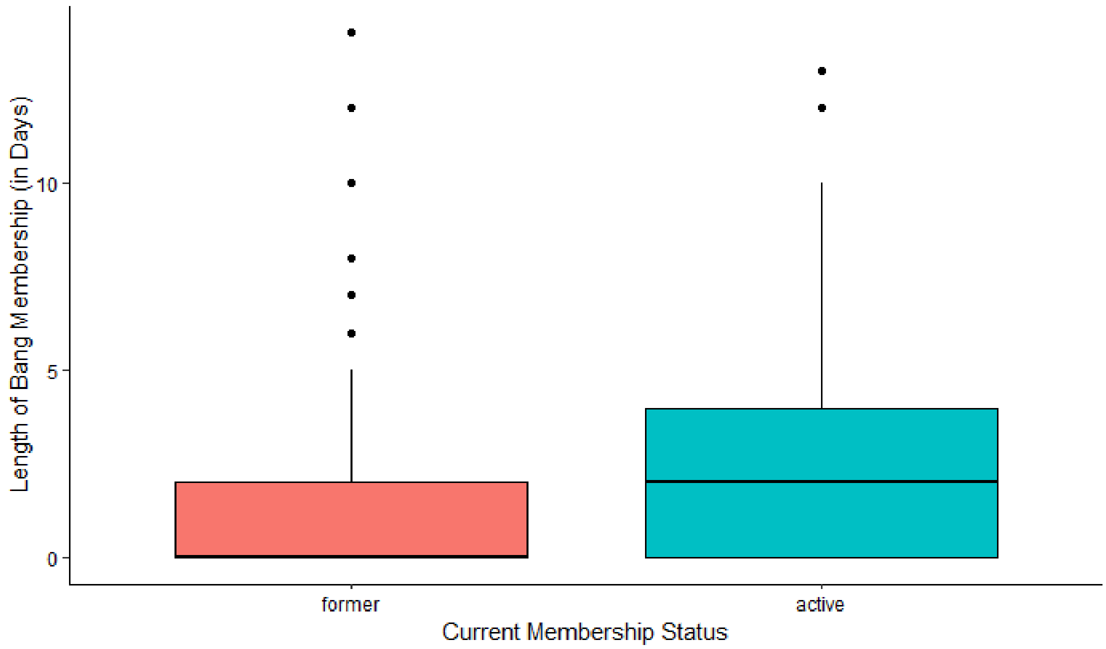


Figure29. Number of Payment Cycle Breaks by Current Membership Status (W = 10640, p < 0.001)

```r
shapiro.test(clean_bang_final$num_renewals) # Not a normal distribution

##
##  Shapiro-Wilk normality test
##
## data:  clean_bang_final$num_renewals
## W = 0.65629, p-value < 2.2e-16

wilcox.test(num_renewals ~ current, data = clean_bang_final)

##
##  Wilcoxon rank sum test with continuity correction
##
## data:  num_renewals by current
## W = 10478, p-value = 1.262e-08
## alternative hypothesis: true location shift is not equal to 0
```

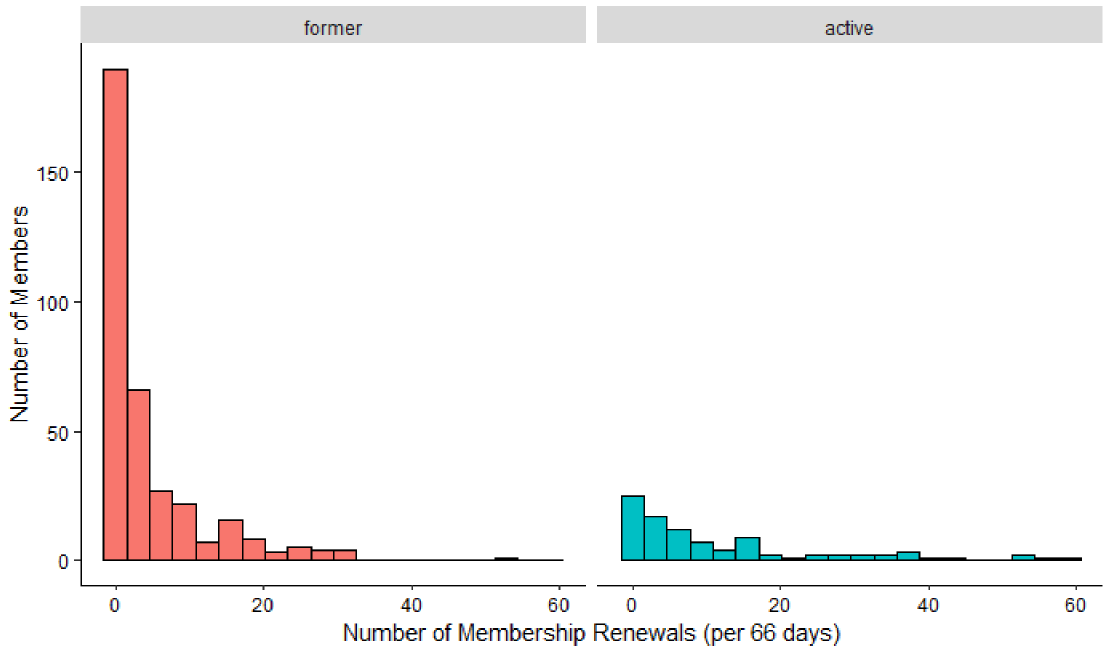


Figure30. Number of Membership Renewals by Current Membership Status (W = 10478, p < 0.001)

```
shapiro.test(clean_bang_final$num_membership_change) # Not a normal
distribution

##
##  Shapiro-Wilk normality test
##
## data:  clean_bang_final$num_membership_change
## W = 0.51418, p-value < 2.2e-16

wilcox.test(num_membership_change ~ current, data = clean_bang_final)

##
##  Wilcoxon rank sum test with continuity correction
##
## data:  num_membership_change by current
## W = 15402, p-value = 0.1289
## alternative hypothesis: true location shift is not equal to 0
```

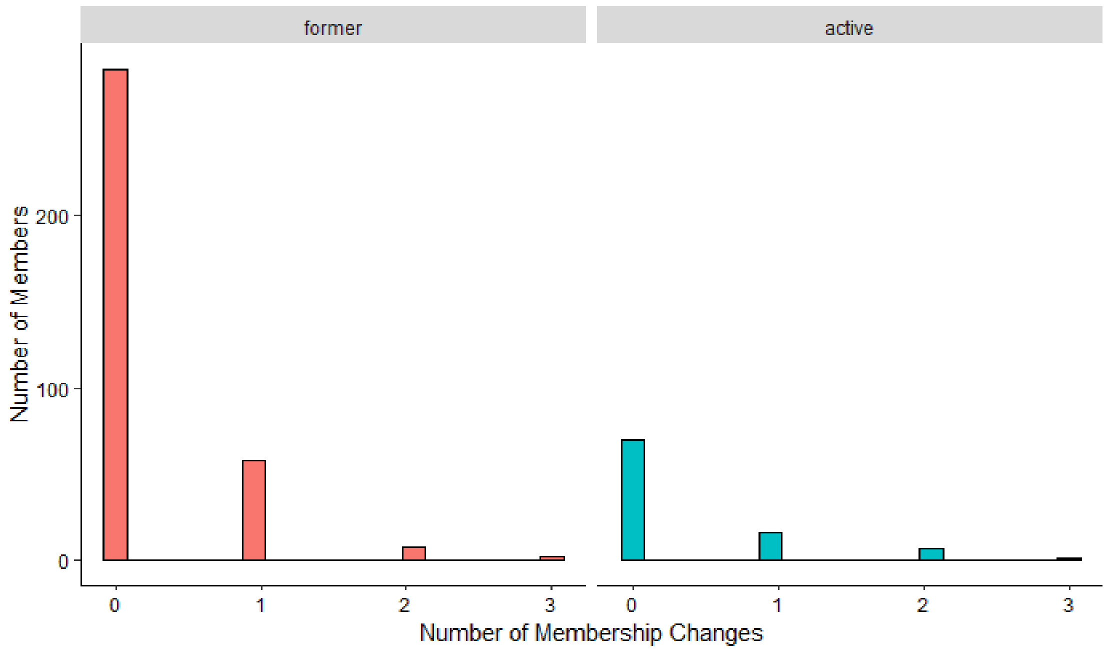


Figure31. Number of Membership Changes by Current Membership Status (W = 15402, p = 0.129)

```
shapiro.test(clean_bang_final$attendance_rate)

##
##  Shapiro-Wilk normality test
##
## data:  clean_bang_final$attendance_rate
## W = 0.95157, p-value = 6.257e-11

chisq.test(clean_bang_final$current,
clean_bang_final$attendance_grouping_ver.1)

##
##  Pearson's Chi-squared test
##
## data:  clean_bang_final$current and
clean_bang_final$attendance_grouping_ver.1
## X-squared = 10.63, df = 5, p-value = 0.05924
```

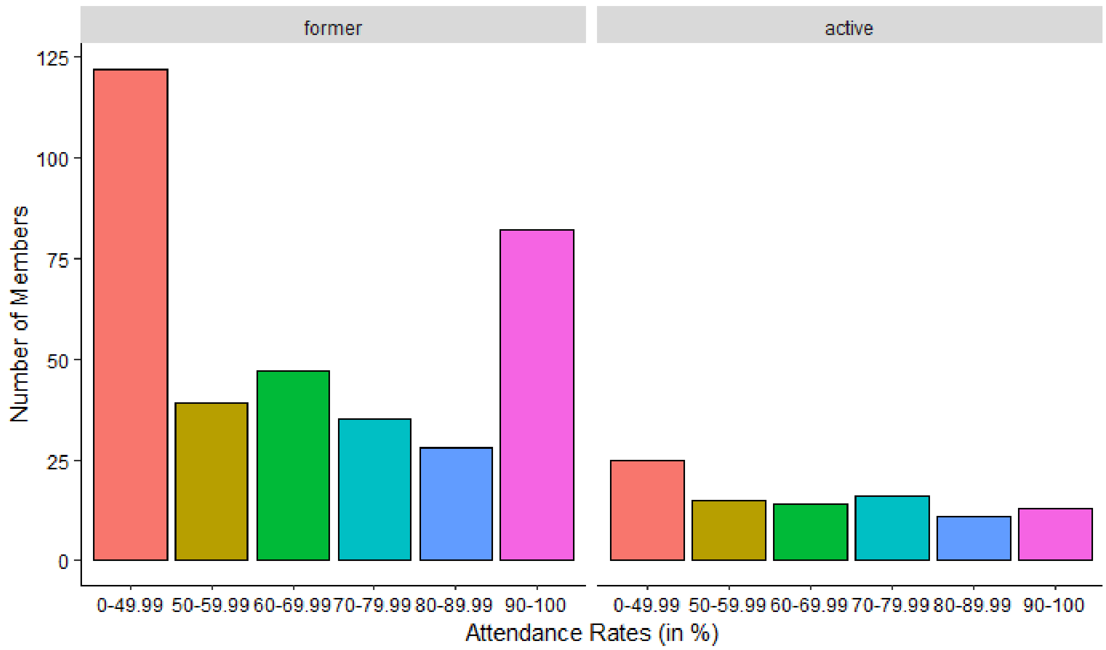


Figure32. Attendance Rate Groupings by Current Membership Status ($\chi 2$ = 10.63, p = 0.059)

```
shapiro.test(clean_bang_final$avg_monthly_rate) # Not a normal distribution

##
##  Shapiro-Wilk normality test
##
## data:  clean_bang_final$avg_monthly_rate
## W = 0.95243, p-value = 8.273e-11

wilcox.test(avg_monthly_rate ~ current, data = clean_bang_final)

##
##  Wilcoxon rank sum test with continuity correction
##
## data:  avg_monthly_rate by current
## W = 15328, p-value = 0.2563
## alternative hypothesis: true location shift is not equal to 0
```

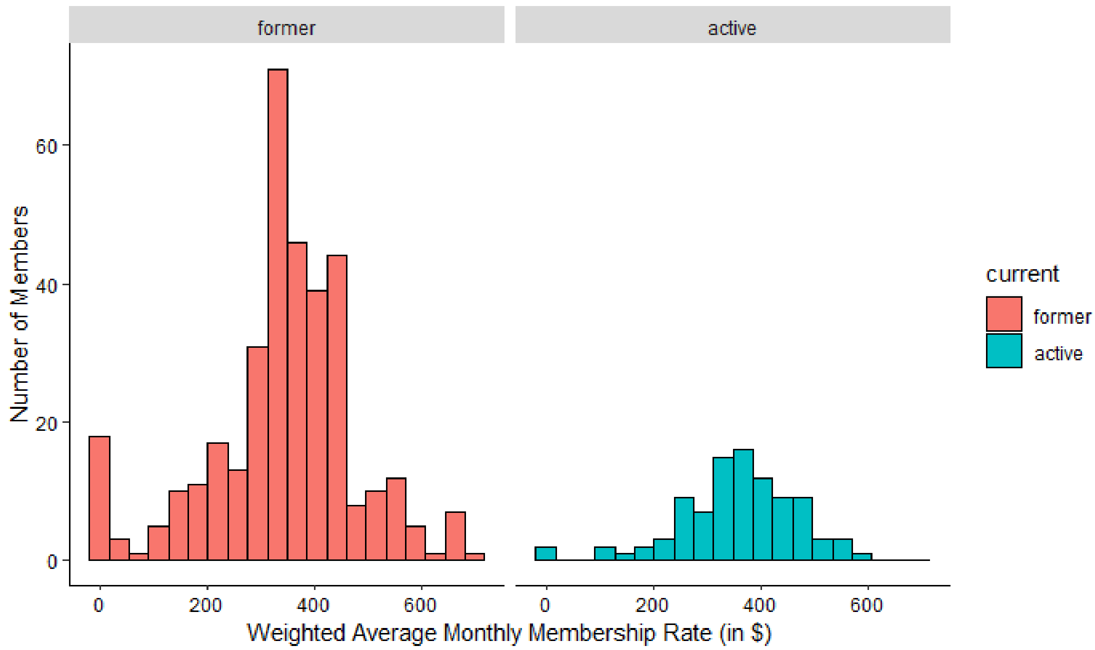


Figure33a. Average Monthly Membership Rate by Current Membership Status (W = 15328, p = 0.256)

```
chisq.test(clean_bang_final$current, clean_bang_final$monthly_rate_group)

## Warning in chisq.test(clean_bang_final$current,
## clean_bang_final$monthly_rate_group): Chi-squared approximation may be
incorrect

##
##  Pearson's Chi-squared test
##
## data:  clean_bang_final$current and clean_bang_final$monthly_rate_group
## X-squared = 21.873, df = 11, p-value = 0.02537
```

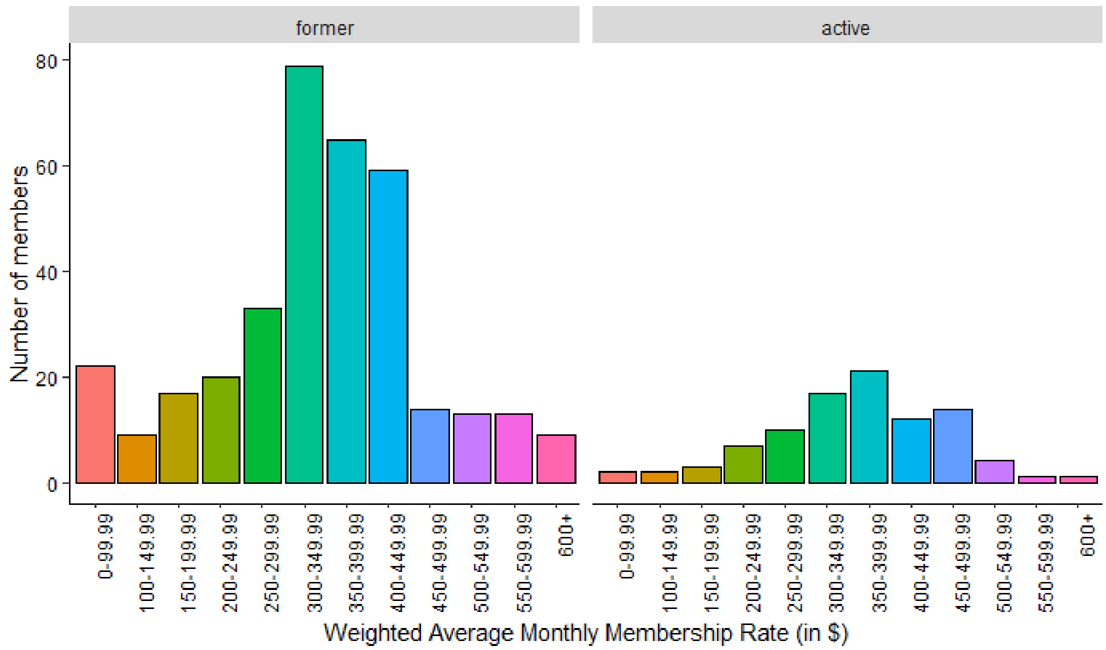


Figure33b. Average Monthly Membership Rate Groupings by Current Membership Status ($\chi2$ = 21.87, p = 0.025)

```
chisq.test(clean_bang_final$num_billing_issue, clean_bang_final$current)

##
##  Pearson's Chi-squared test
##
## data:  clean_bang_final$num_billing_issue and clean_bang_final$current
## X-squared = 4.7508, df = 2, p-value = 0.09298
```

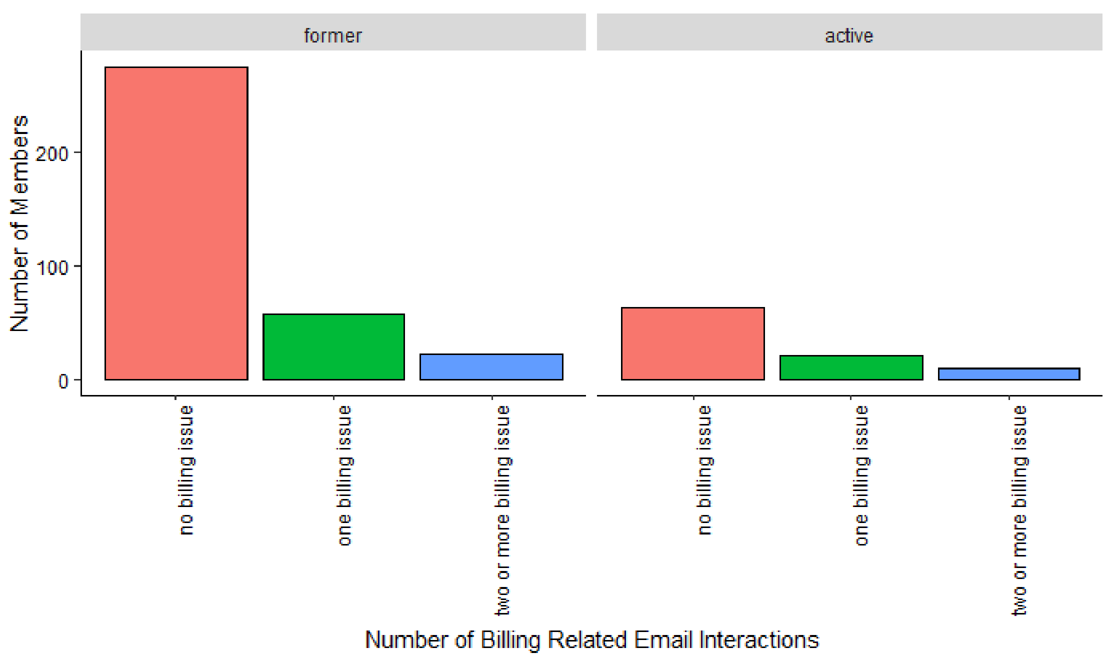


Figure34a. Number of Billing-Related Email Interactions by Current Membership Status ($\chi2$ = 4.75, p = 0.093)

```
chisq.test(clean_bang_final$ever_billing_issue, clean_bang_final$current)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  clean_bang_final$ever_billing_issue and clean_bang_final$current
## X-squared = 3.9418, df = 1, p-value = 0.0471
```

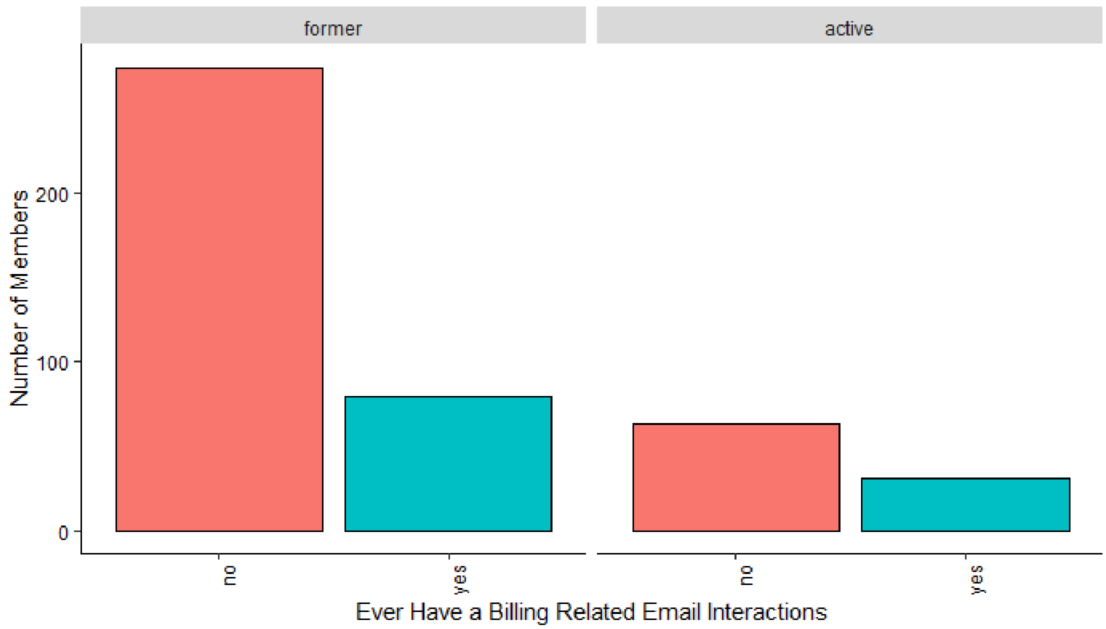


Figure34b. Status of Ever Having a Billing-Related Email Interaction by Current Membership Status ($\chi2$ = 3.94, p = 0.047)

```
shapiro.test(clean_bang_final$new_per_ticket_cx) # Not normal distribution

##
##  Shapiro-Wilk normality test
##
## data:  clean_bang_final$new_per_ticket_cx
## W = 0.75859, p-value < 2.2e-16

wilcox.test(new_per_ticket_cx ~ current, data = clean_bang_final)

##
##  Wilcoxon rank sum test with continuity correction
##
## data:  new_per_ticket_cx by current
## W = 15993, p-value = 0.5739
## alternative hypothesis: true location shift is not equal to 0
```

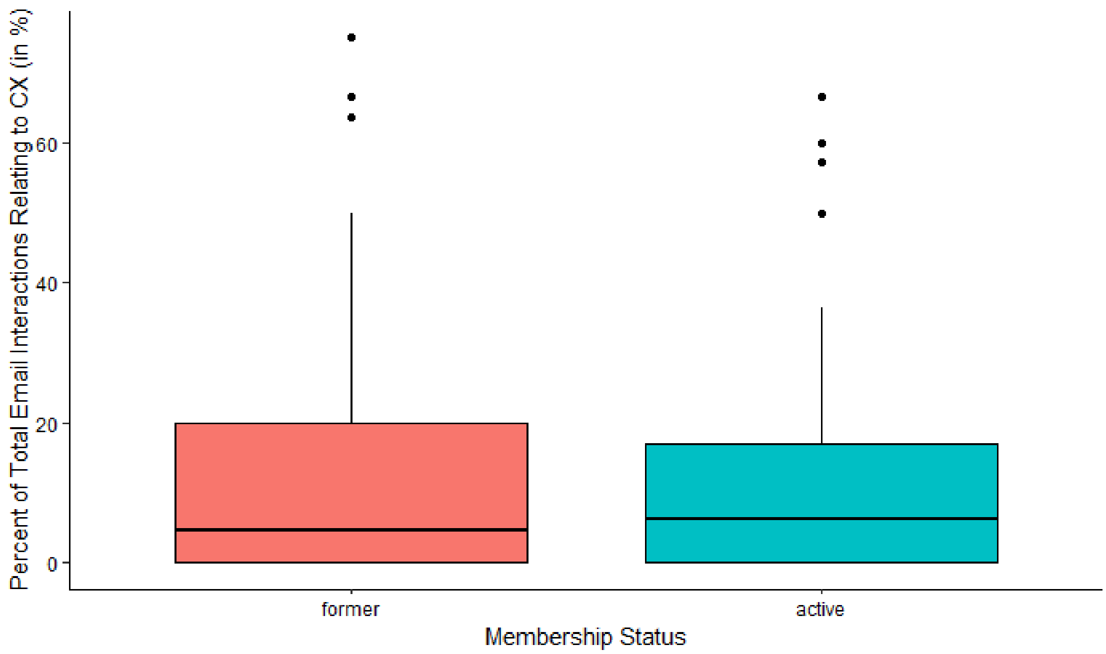


Figure35. Percent of CX-Related Email Interactions by Current Membership Status (W = 15993, p = 0.574)

```r
shapiro.test(clean_bang_final$new_per_ticket_scheduling) # Not normal
distribution

##
##  Shapiro-Wilk normality test
##
## data:  clean_bang_final$new_per_ticket_scheduling
## W = 0.90146, p-value < 2.2e-16

wilcox.test(new_per_ticket_scheduling ~ current, data = clean_bang_final)

##
##  Wilcoxon rank sum test with continuity correction
##
## data:  new_per_ticket_scheduling by current
## W = 14302, p-value = 0.03696
## alternative hypothesis: true location shift is not equal to 0
```

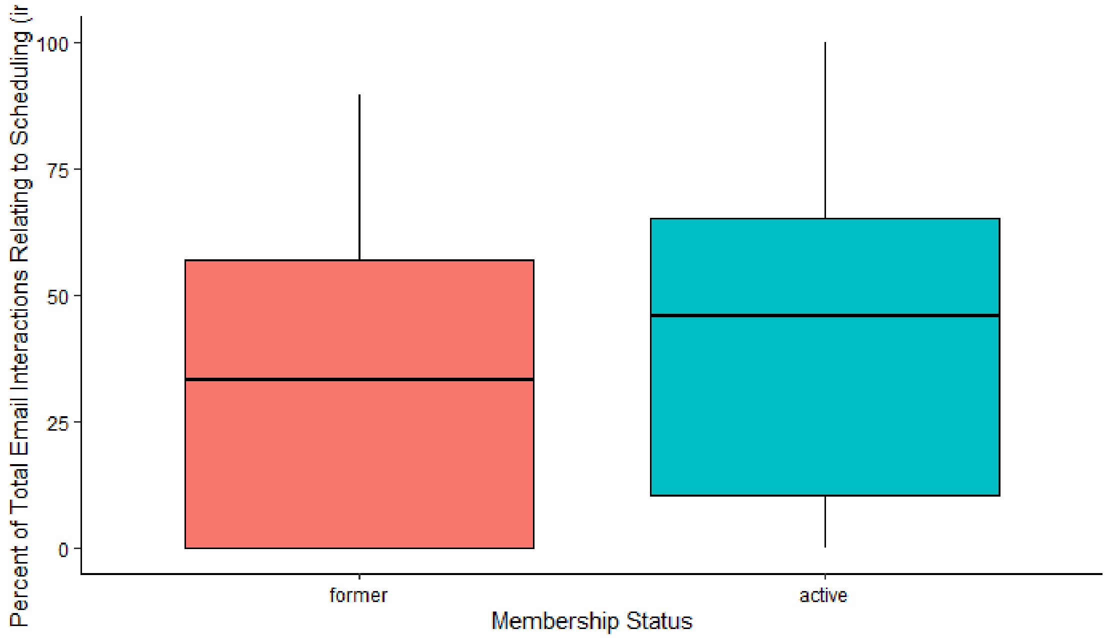


Figure36. Percent of Scheduling-Related Email Interactions by Current Membership Status
(W = 14302, p = 0.037)

```
shapiro.test(clean_bang_final$new_per_ticket_service) # Not normal
distribution

##
##  Shapiro-Wilk normality test
##
## data:  clean_bang_final$new_per_ticket_service
## W = 0.93109, p-value = 1.71e-13

wilcox.test(new_per_ticket_service ~ current, data = clean_bang_final)

##
##  Wilcoxon rank sum test with continuity correction
##
## data:  new_per_ticket_service by current
## W = 18718, p-value = 0.05499
## alternative hypothesis: true location shift is not equal to 0
```

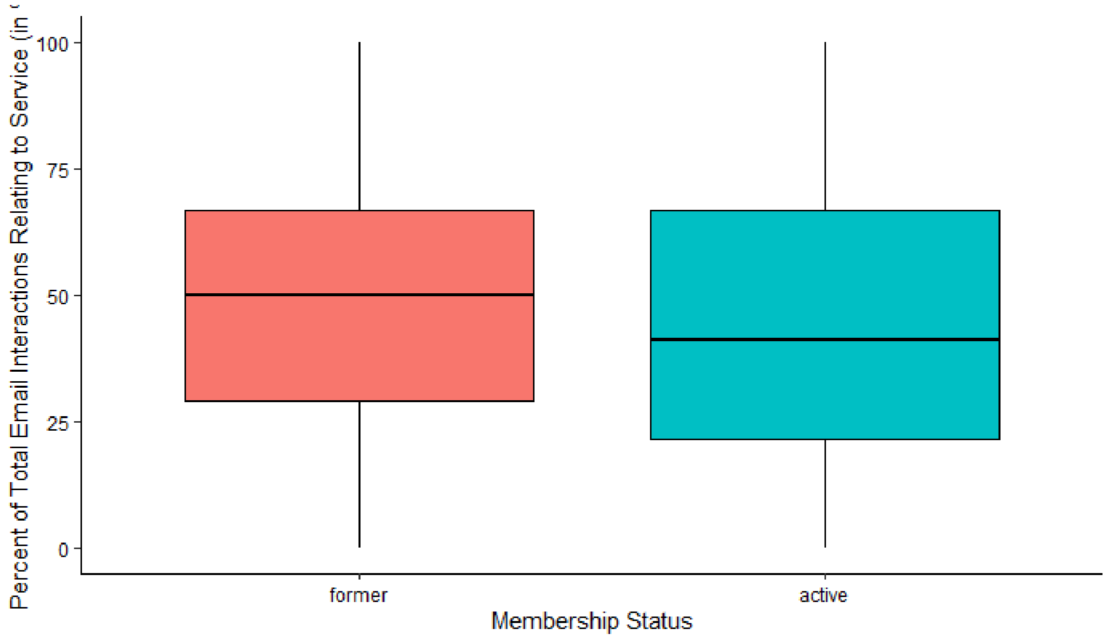


Figure37. Percent of Service-Related Email Interactions by Current Membership Status (W = 18718, p = 0.054)

```
shapiro.test(clean_bang_final$new_num_total) # not normally distributed

##
##  Shapiro-Wilk normality test
##
## data:  clean_bang_final$new_num_total
## W = 0.54485, p-value < 2.2e-16

wilcox.test(new_num_total ~ current, data = clean_bang_final)

##
##  Wilcoxon rank sum test with continuity correction
##
## data:  new_num_total by current
## W = 12510, p-value = 0.0002348
## alternative hypothesis: true location shift is not equal to 0
```

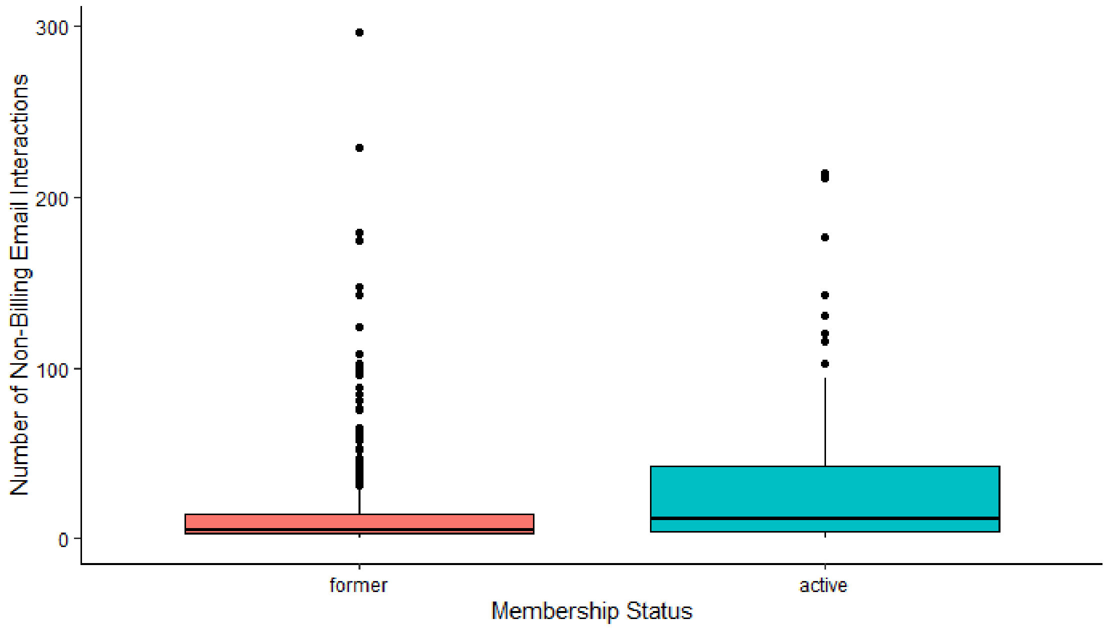


Figure38a. Non-Billing-Related Email Interactions by Current Membership Status
(W = 12510, p < 0.001)

```r
shapiro.test(clean_bang_final$num_emails_month) # not normally distributed
```

```
##
##  Shapiro-Wilk normality test
##
## data:  clean_bang_final$num_emails_month
## W = 0.6185, p-value < 2.2e-16
```

```r
wilcox.test(num_emails_month ~ current, data = clean_bang_final)
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  num_emails_month by current
## W = 22726, p-value = 3.558e-08
## alternative hypothesis: true location shift is not equal to 0
```

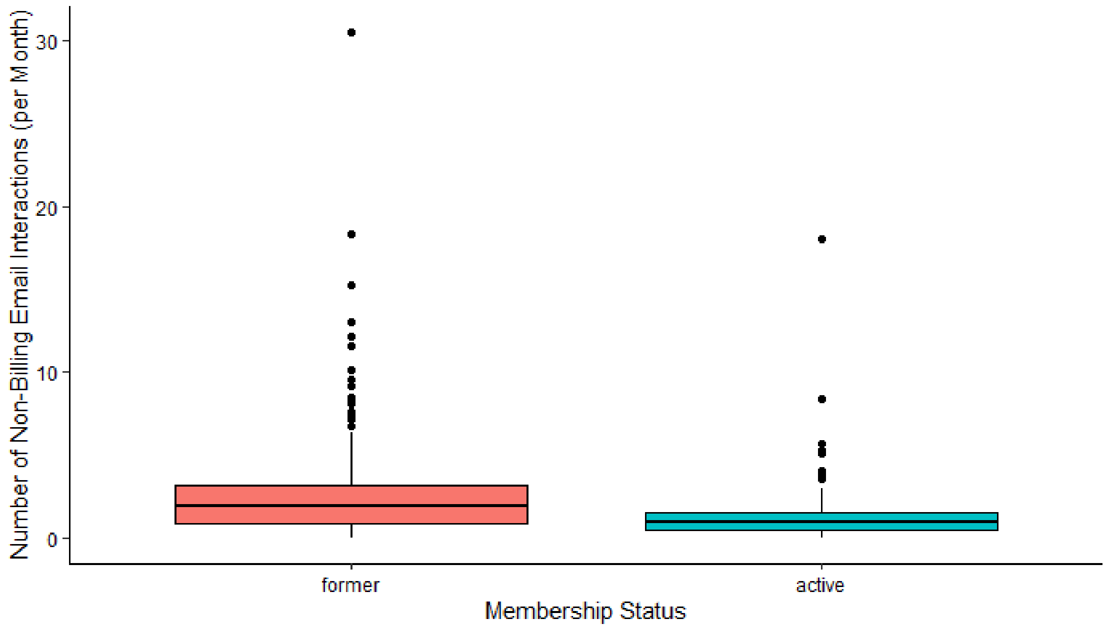


Figure38b. Non-Billing-Related Email Interactions/Month by Current Membership Status (W = 22726, p < 0.001)

Length of Membership

Examining the length of membership across age groups, significantly longer membership length was observed in those aged 30-44 as compared to 18-29. This difference was also noted in terms of membership types with those with a 2x/week membership had significantly longer membership length as compared to 3x/week. As it relates to attendance rates, those that attended 70%-79% of the time had significantly longer membership length as compared to those engaging in less than 50% of the time.

In terms of email interactions, those that reported any billing-related issues were found to have longer membership rates than those without. Looking at the other types of email interactions, there was a significant correlation with increased percentage of email interaction with length of membership. In fact this was supported when examining the relationship between total non-billing related email interactions and length of membership.

Lastly in terms of average monthly rate, those with 350 to 449 per month were found to have significantly longer membership length as compared to all other monthly rates.

```
kruskal.test(length ~ age_group, data = clean_bang_final)

##
##  Kruskal-Wallis rank sum test
##
## data:  length by age_group
## Kruskal-Wallis chi-squared = 15.059, df = 4, p-value = 0.004581

clean_bang_final %>% dunn_test(length ~ age_group, p.adjust.method =  'holm')
```

```
## # A tibble: 10 x 9
##    .y.    group1   group2    n1    n2 statistic        p   p.adj
p.adj.signif
##   * <chr>  <chr>    <chr>  <int> <int>     <dbl>    <dbl>   <dbl> <chr>
##   1 length Under 18 18-29      9    88   -1.36   0.175     1        ns
##   2 length Under 18 30-44      9   211    0.0454 0.964     1        ns
##   3 length Under 18 45-64      9   131   -0.384  0.701     1        ns
##   4 length Under 18 65+        9     8   -0.346  0.730     1        ns
##   5 length 18-29    30-44     88   211    3.86   0.000114 0.00114 **
##   6 length 18-29    45-64     88   131    2.48   0.0131   0.117    ns
##   7 length 18-29    65+       88     8    0.829  0.407     1        ns
##   8 length 30-44    45-64    211   131   -1.33   0.184     1        ns
##   9 length 30-44    65+      211     8   -0.509  0.611     1        ns
## 10 length 45-64    65+      131     8   -0.0984 0.922     1        ns
```

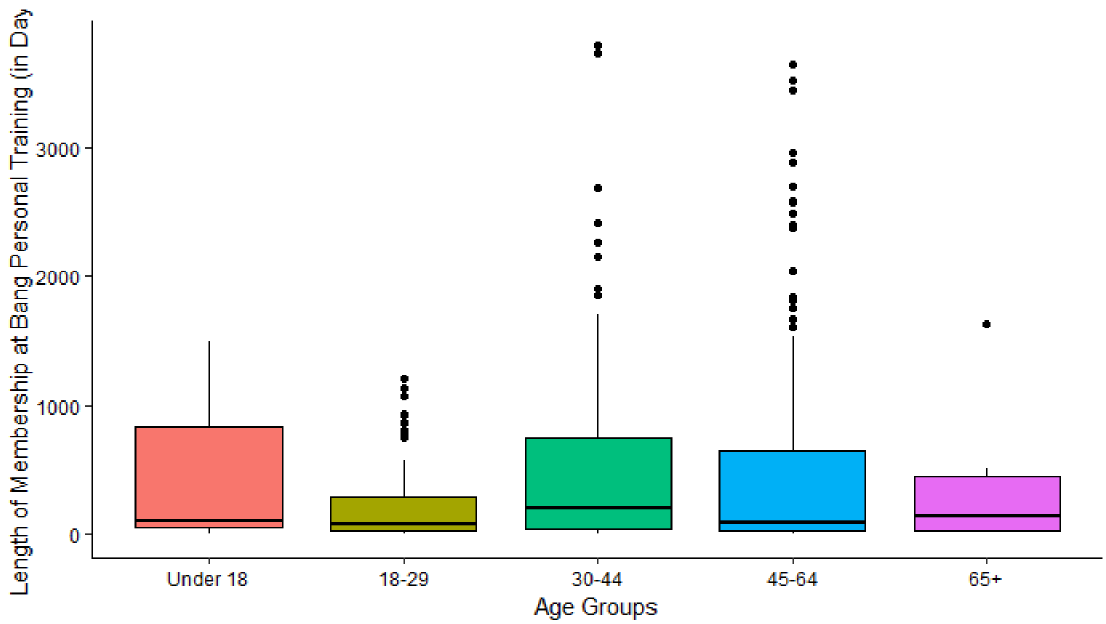


Figure39. Length of Membership Distributed Across Age Groups (H = 15.06, p = 0.005). Following Pairwise Comparisons, Longer Membership Length Observed Amongst 30-44 as compared to 45-64 (Z = 3.86, p = 0.001)

```
kruskal.test(length ~ employment_sector, data = clean_bang_final)

##
##  Kruskal-Wallis rank sum test
##
## data:  length by employment_sector
## Kruskal-Wallis chi-squared = 28.526, df = 15, p-value = 0.0185

clean_bang_final %>% dunn_test(length ~ employment_sector, p.adjust.method =
'holm')

## # A tibble: 120 x 9
##    .y.    group1 group2             n1    n2 statistic      p p.adj
p.adj.signif
##   * <chr>  <chr>  <chr>            <int> <int>     <dbl>  <dbl> <dbl> <chr>
```

```
## 1 length Other  Advertising/Me~      7     77     -0.153 0.878      1 ns
## 2 length Other  Entrepreneural~      7     41      0.183 0.855      1 ns
## 3 length Other  Finance/Insura~      7     48     -0.591 0.554      1 ns
## 4 length Other  Government/Leg~      7     23     -0.880 0.379      1 ns
## 5 length Other  Health Care/Se~      7     27     -0.382 0.702      1 ns
## 6 length Other  Hospitality/Re~      7     24     -0.969 0.332      1 ns
## 7 length Other  Manufacturing/~      7      9     -1.84  0.0658     1 ns
## 8 length Other  Natural Resour~      7      6     -0.463 0.643      1 ns
## 9 length Other  Professional/T~      7     57     -1.05  0.292      1 ns
## 10 length Other Real Estate/Co~      7     19     -0.202 0.840      1 ns
## # ... with 110 more rows
```

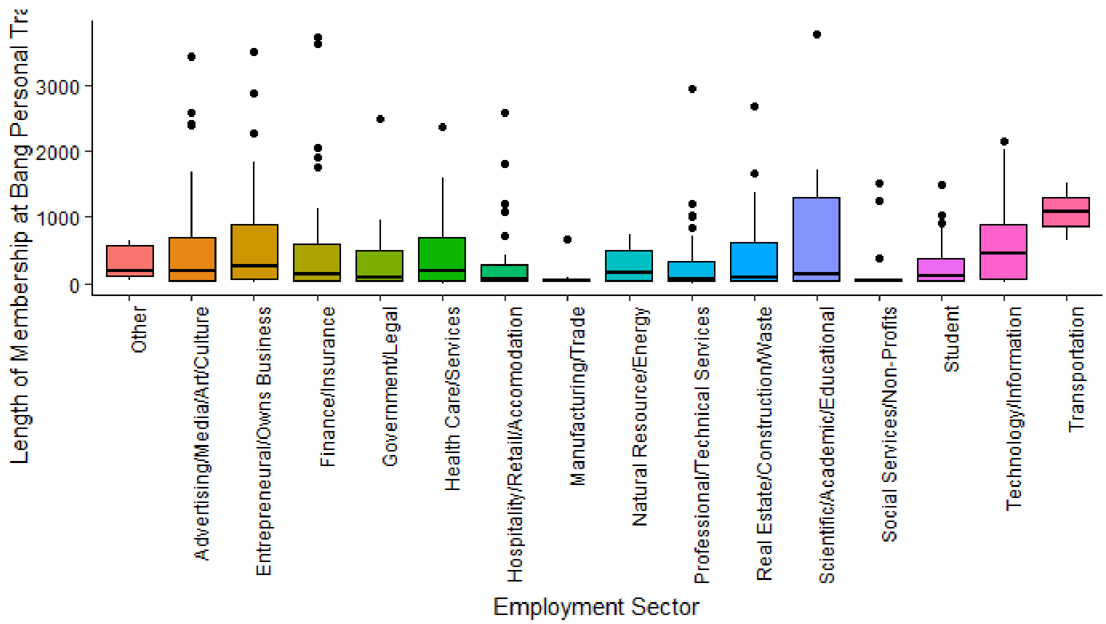


Figure40. Length of Membership Distributed Across Employment Sector (H = 28.53, p = 0.019). There was No Significant Difference Following Pairwise Comparisons

```
kruskal.test(length ~ membership, data = clean_bang_final)

##
##  Kruskal-Wallis rank sum test
##
## data:  length by membership
## Kruskal-Wallis chi-squared = 20.321, df = 6, p-value = 0.002427

clean_bang_final %>% dunn_test(length ~ membership, p.adjust.method =
'holm')

## # A tibble: 21 x 9
##    .y.    group1 group2      n1    n2 statistic         p   p.adj
p.adj.signif
##  * <chr>  <chr>  <chr>    <int> <int>     <dbl>     <dbl>   <dbl> <chr>
##  1 length 1x     2x          13   137      2.52  0.0118    0.235  ns
##  2 length 1x     3x          13   239      1.04  0.300     1      ns
##  3 length 1x     4x          13    22      1.42  0.155     1      ns
```

```
##  4 length 1x     unlimited    13     4     1.56  0.119     1       ns
##  5 length 1x     group        13    12     1.37  0.171     1       ns
##  6 length 1x     distance     13    20     0.965 0.335     1       ns
##  7 length 2x     3x          137   239    -4.07  0.0000480 0.00101 **
##  8 length 2x     4x          137    22    -1.02  0.309     1       ns
##  9 length 2x     unlimited   137     4     0.315 0.753     1       ns
## 10 length 2x     group       137    12    -0.607 0.544     1       ns
## # ... with 11 more rows
```

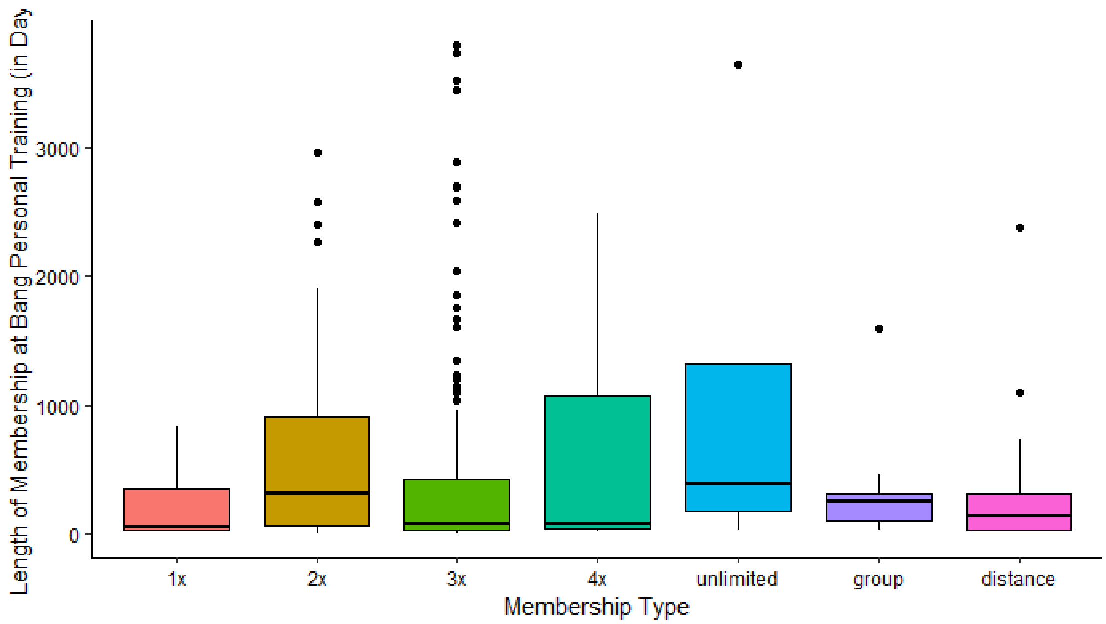


Figure41. Length of Membership Distributed Across Membership Type (H = 19.94, p = 0.003. After Pairwise Adjustment, Longer Membership Length Observed Amongst 2x/Week vs. 3x/Week (Z = 4.07, p = 0.001)

```
kruskal.test(length ~ attendance_rate_group, data = clean_bang_final)

##
##  Kruskal-Wallis rank sum test
##
## data:  length by attendance_rate_group
## Kruskal-Wallis chi-squared = 40.936, df = 9, p-value = 5.138e-06

dunn_test(length ~ attendance_rate_group, data = clean_bang_final,
p.adjust.method = 'holm')

## # A tibble: 45 x 9
##    .y.    group1   group2     n1    n2 statistic        p    p.adj
p.adj.signif
##  * <chr>  <chr>    <chr>   <int> <int>     <dbl>    <dbl>    <dbl> <chr>
##  1 length 0-9.99   10-19.~    16    19      2.83   4.72e-3  1.51e-1 ns
##  2 length 0-9.99   20-29.~    16    32      1.50   1.34e-1  1.00e+0 ns
##  3 length 0-9.99   30-39.~    16    42      3.63   2.83e-4  1.07e-2 *
##  4 length 0-9.99   40-49.~    16    38      4.45   8.46e-6  3.72e-4 ***
##  5 length 0-9.99   50-59.~    16    54      4.04   5.45e-5  2.18e-3 **
##  6 length 0-9.99   60-69.~    16    61      4.02   5.85e-5  2.28e-3 **
```

```
##  7 length 0-9.99   70-79.~    16    51      4.97   6.62e-7  2.98e-5 ****
##  8 length 0-9.99   80-89.~    16    39      4.09   4.24e-5  1.74e-3 **
##  9 length 0-9.99   90-100     16    95      4.16   3.19e-5  1.34e-3 **
## 10 length 10-19.99 20-29.~    19    32     -1.72   8.47e-2  1.00e+0 ns
## # ... with 35 more rows
```

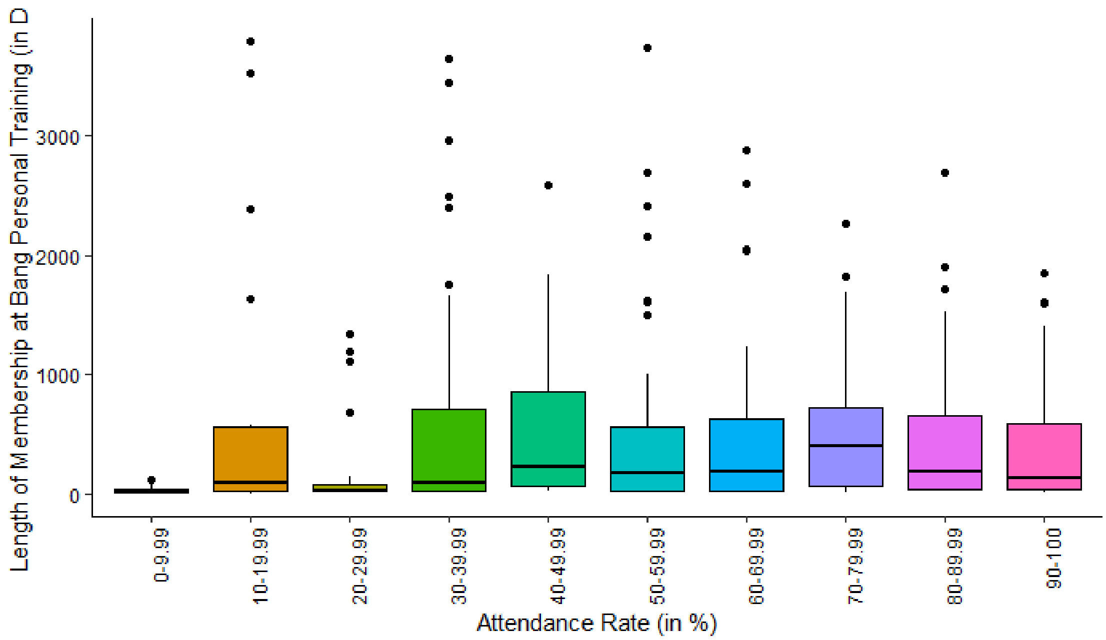


Figure42a. Length of Membership by Attendance Rate
(H = 40.94, p < 0.001)

```
kruskal.test(length ~ attendance_grouping_ver.1, data = clean_bang_final)

##
##  Kruskal-Wallis rank sum test
##
## data:  length by attendance_grouping_ver.1
## Kruskal-Wallis chi-squared = 13.715, df = 5, p-value = 0.01752

dunn_test(length ~ attendance_grouping_ver.1, data = clean_bang_final,
p.adjust.method = 'holm')

## # A tibble: 15 x 9
##    .y.    group1   group2     n1    n2 statistic        p  p.adj
p.adj.signif
##  * <chr>  <chr>    <chr>   <int> <int>     <dbl>    <dbl>  <dbl> <chr>
##  1 length 0-49.99  50-59.99  147    54      1.74  0.0819   0.983  ns
##  2 length 0-49.99  60-69.99  147    61      1.69  0.0914   0.983  ns
##  3 length 0-49.99  70-79.99  147    51      3.40  0.000666 0.0100 **
##  4 length 0-49.99  80-89.99  147    39      1.91  0.0563   0.775  ns
##  5 length 0-49.99  90-100    147    95      1.92  0.0553   0.775  ns
##  6 length 50-59.99 60-69.99   54    61    -0.106  0.916    1      ns
##  7 length 50-59.99 70-79.99   54    51      1.41  0.157    1      ns
##  8 length 50-59.99 80-89.99   54    39     0.318  0.750    1      ns
##  9 length 50-59.99 90-100     54    95    -0.144  0.885    1      ns
```

```
## 10 length 60-69.99 70-79.99     61     51     1.56    0.119     1        ns
## 11 length 60-69.99 80-89.99     61     39     0.423   0.672     1        ns
## 12 length 60-69.99 90-100       61     95    -0.0294  0.977     1        ns
## 13 length 70-79.99 80-89.99     51     39    -0.984   0.325     1        ns
## 14 length 70-79.99 90-100       51     95    -1.73    0.0831    0.983    ns
## 15 length 80-89.99 90-100       39     95    -0.481   0.630     1        ns
```

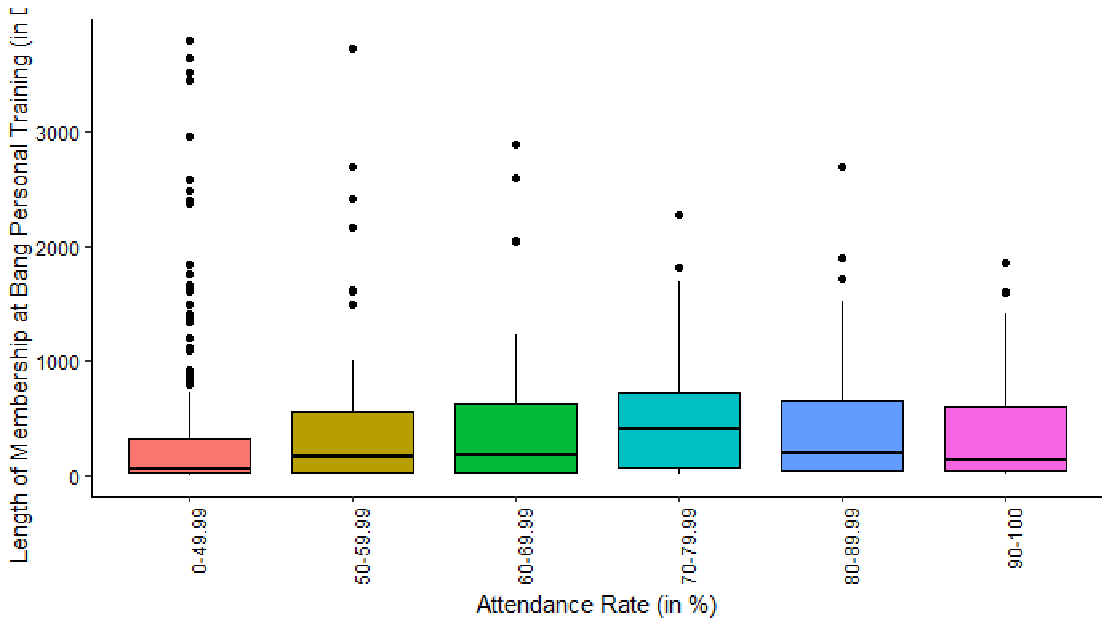


Figure42b. Length of Membership by Attendance Rate (H = 13.48, p = 0.009).
After pairwise adjustments, longer membership duration b/t those with 70%-79.99% attendance rate as compared to 0-49.99% (Z = 3.40, p = 0.007)

```r
kruskal.test(length ~ num_billing_issue, data = clean_bang_final)

##
##  Kruskal-Wallis rank sum test
##
## data:  length by num_billing_issue
## Kruskal-Wallis chi-squared = 31.239, df = 2, p-value = 1.646e-07

dunn_test(length ~ num_billing_issue, data = clean_bang_final,
p.adjust.method = 'holm')
```

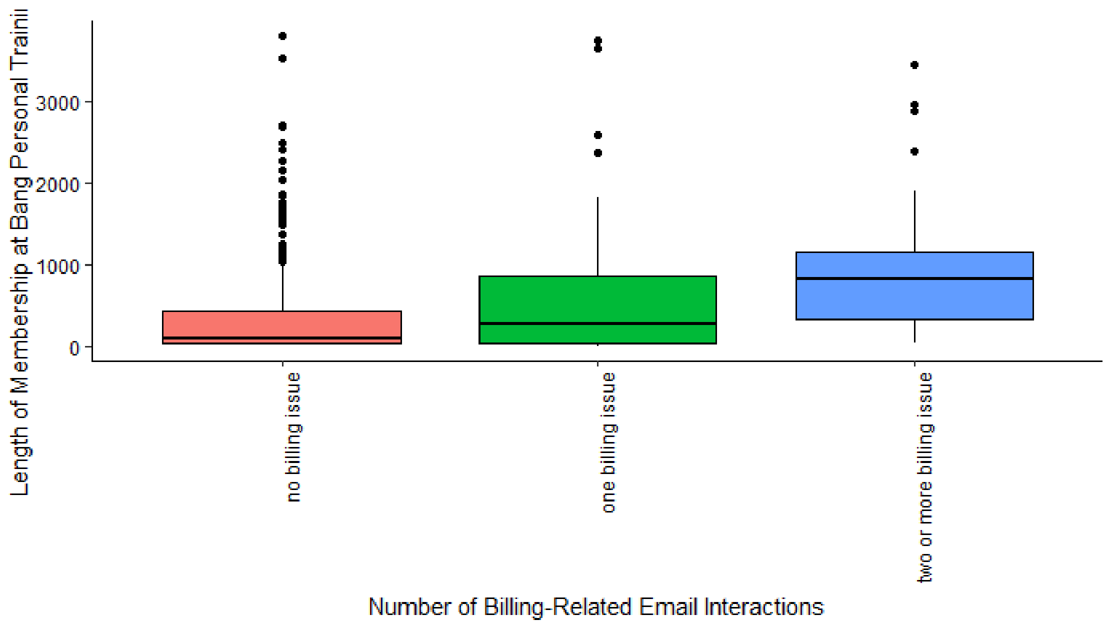


Figure43a. Length of Membership by Number of Billing-Related Email Interactions (H = 31.24, p < 0.001). Following Pairwise Comparisons, Greater Membership Lengths Noted for Members with 2 or More such Interactions.

```
wilcox.test(length ~ ever_billing_issue, data = clean_bang_final)

##
##  Wilcoxon rank sum test with continuity correction
##
## data:  length by ever_billing_issue
## W = 13570, p-value = 2.414e-05
## alternative hypothesis: true location shift is not equal to 0
```

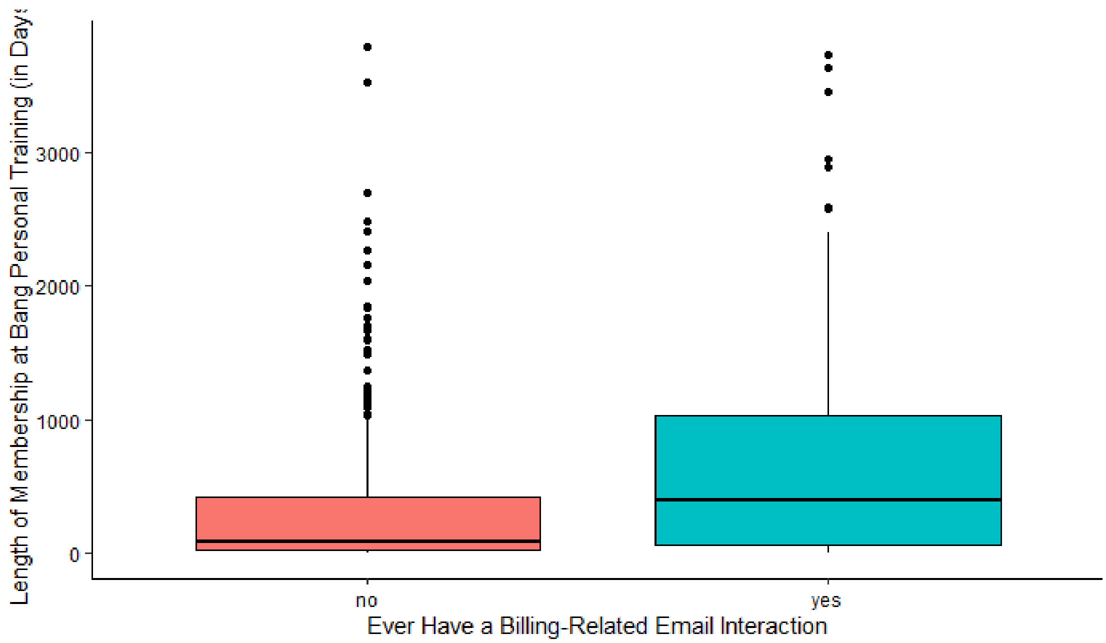


Figure43b. Length of Membership by Status of Ever Having a Billing-Related Issue (W = 13570, p < 0.001)

```
cor.test(x = clean_bang_final$new_per_ticket_cx, y = clean_bang_final$length,
method = 'spearman')

## Warning in cor.test.default(x = clean_bang_final$new_per_ticket_cx, y =
## clean_bang_final$length, : Cannot compute exact p-value with ties

##
##  Spearman's rank correlation rho
##
## data:  clean_bang_final$new_per_ticket_cx and clean_bang_final$length
## S = 11297350, p-value = 2.488e-07
## alternative hypothesis: true rho is not equal to 0
## sample estimates:
##     rho
## 0.24106
```

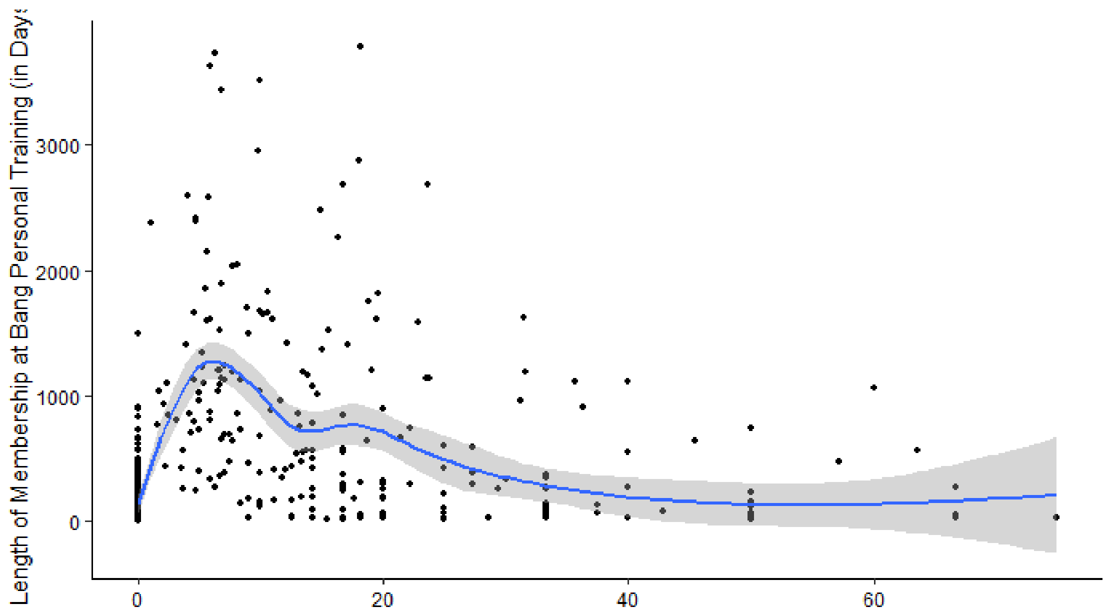


Figure44. Length of Membership By Percentage of Email Interactions Relating to CX
($\rho = 0.241$, $p < 0.001$)

```
cor.test(x = clean_bang_final$new_per_ticket_scheduling, y =
clean_bang_final$length, method = 'spearman')

## Warning in cor.test.default(x =
clean_bang_final$new_per_ticket_scheduling, :
## Cannot compute exact p-value with ties

##
##  Spearman's rank correlation rho
##
## data:  clean_bang_final$new_per_ticket_scheduling and
clean_bang_final$length
## S = 7629040, p-value < 2.2e-16
## alternative hypothesis: true rho is not equal to 0
## sample estimates:
##       rho
## 0.4874919
```

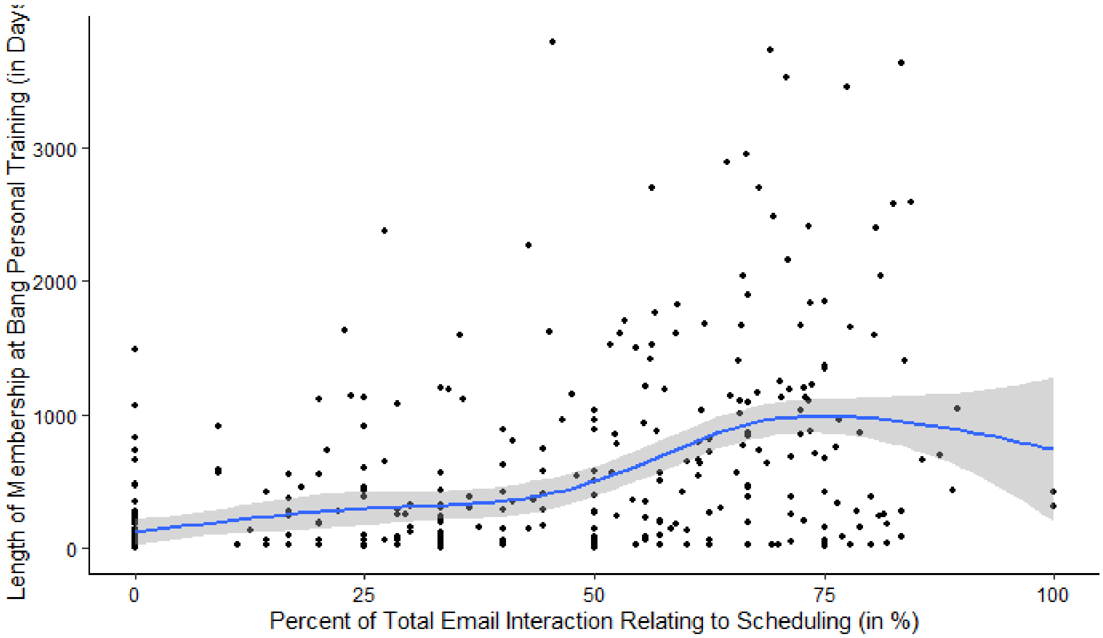


Figure45. Length of Membership By Percentage of Email Interactions Relating to Scheduling (ρ = 0.487, p < 0.001)

```
cor.test(x = clean_bang_final$new_per_ticket_service, y =
clean_bang_final$length, method = 'spearman')

## Warning in cor.test.default(x = clean_bang_final$new_per_ticket_service, :
## Cannot compute exact p-value with ties

##
##  Spearman's rank correlation rho
##
## data:  clean_bang_final$new_per_ticket_service and clean_bang_final$length
## S = 21968204, p-value < 2.2e-16
## alternative hypothesis: true rho is not equal to 0
## sample estimates:
##        rho
## -0.4757929
```

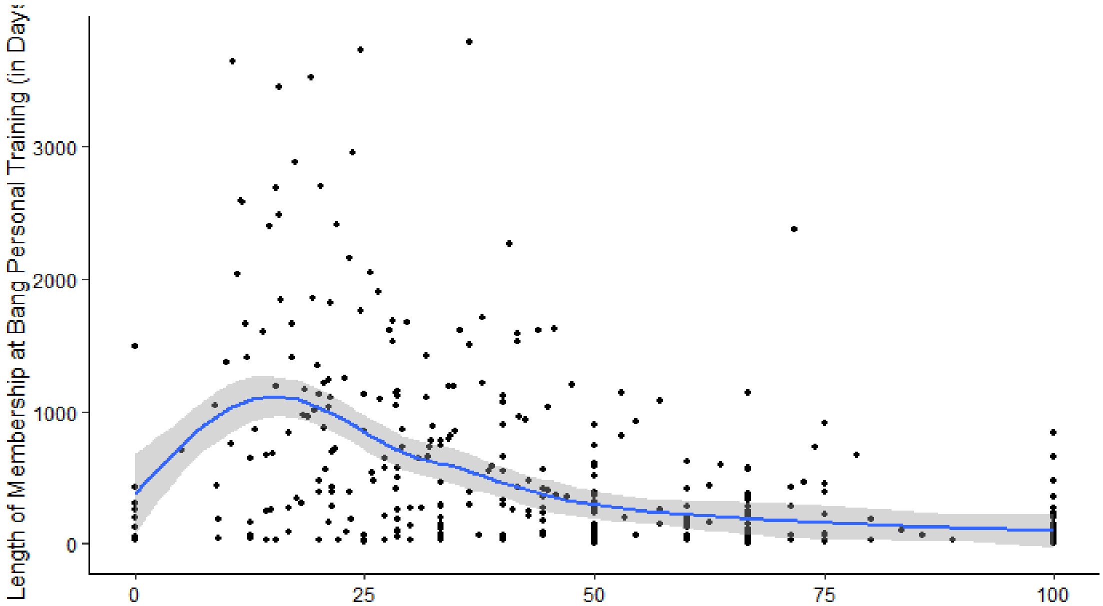


Figure46. Length of Membership By Percentage of Email Interactions Relating to Service (ρ = -0.476, p < 0.001)

```
cor.test(x = clean_bang_final$new_num_total, y = clean_bang_final$length,
method = 'spearman')

## Warning in cor.test.default(x = clean_bang_final$new_num_total, y =
## clean_bang_final$length, : Cannot compute exact p-value with ties

##
##  Spearman's rank correlation rho
##
## data:  clean_bang_final$new_num_total and clean_bang_final$length
## S = 3263782, p-value < 2.2e-16
## alternative hypothesis: true rho is not equal to 0
## sample estimates:
##       rho
## 0.7807438
```

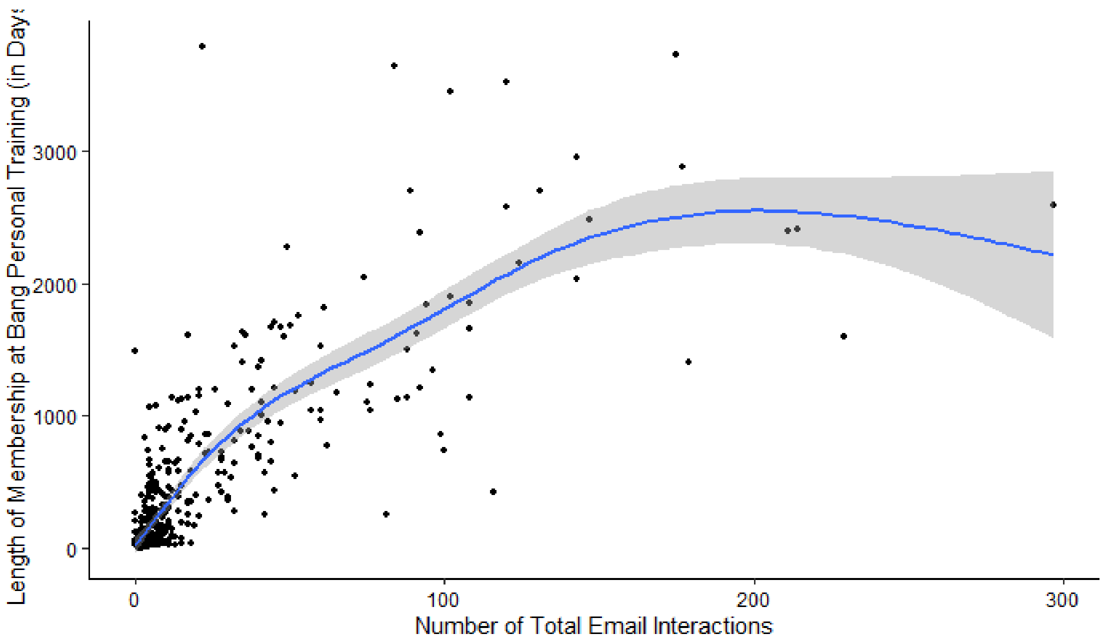


Figure47a. Length of Membership by Total Number of Non-Billing Emails
(ρ = 0.781, p < 0.001)

```
cor.test(x = clean_bang_final$num_emails_month, y = clean_bang_final$length,
method = 'spearman')

## Warning in cor.test.default(x = clean_bang_final$num_emails_month, y =
## clean_bang_final$length, : Cannot compute exact p-value with ties

##
##  Spearman's rank correlation rho
##
## data:  clean_bang_final$num_emails_month and clean_bang_final$length
## S = 23277953, p-value < 2.2e-16
## alternative hypothesis: true rho is not equal to 0
## sample estimates:
##        rho
## -0.5637799
```

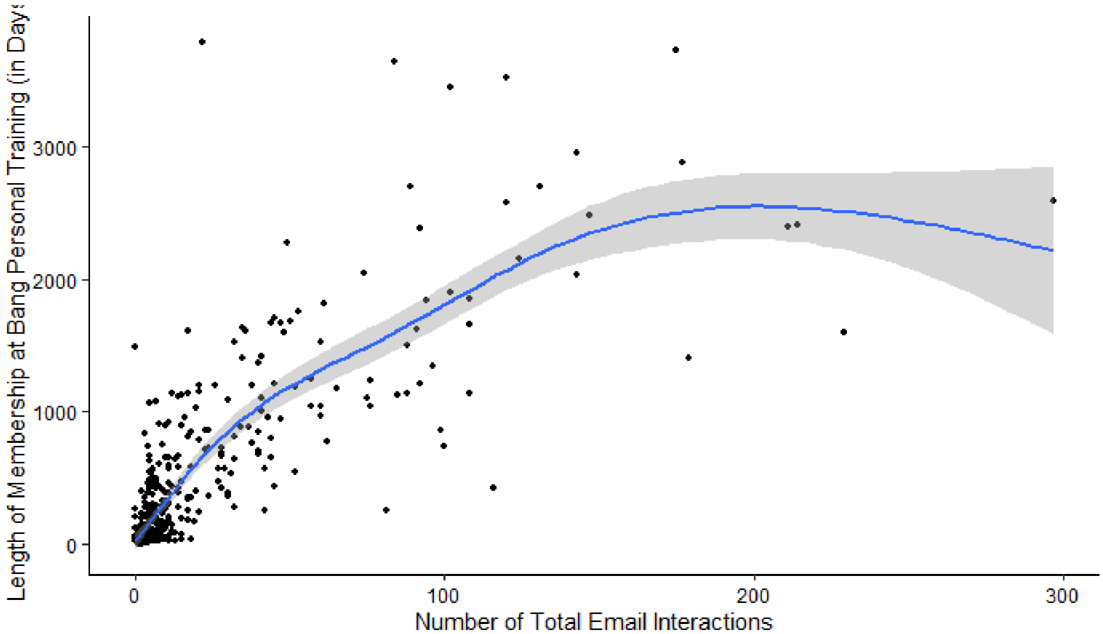


Figure47b. Length of Membership by Total Number of Non-Billing Email Interactions per Month
(ρ = -0.564, p < 0.001)

```
kruskal.test(length ~ monthly_grouping_ver.1, data = clean_bang_final)

##
##  Kruskal-Wallis rank sum test
##
## data:  length by monthly_grouping_ver.1
## Kruskal-Wallis chi-squared = 71.341, df = 7, p-value = 7.912e-13

dunn_test(length ~ monthly_grouping_ver.1, data = clean_bang_final,
p.adjust.method = 'holm')

## # A tibble: 28 x 9
##    .y.    group1   group2     n1    n2 statistic        p     p.adj
p.adj.signif
##  * <chr>  <chr>    <chr>   <int> <int>     <dbl>    <dbl>     <dbl> <chr>
##  1 length 0-149.99 150-19~    35    20     1.15  2.52e- 1   1.00e+0 ns
##  2 length 0-149.99 200-29~    35    70     3.90  9.74e- 5   2.24e-3 **
##  3 length 0-149.99 300-34~    35    96     3.78  1.55e- 4   3.24e-3 **
##  4 length 0-149.99 350-39~    35    86     6.56  5.23e-11   1.46e-9 ****
##  5 length 0-149.99 400-44~    35    71     4.50  6.80e- 6   1.70e-4 ***
##  6 length 0-149.99 450-49~    35    28     0.995 3.20e- 1   1.00e+0 ns
##  7 length 0-149.99 500+       35    41     1.01  3.14e- 1   1.00e+0 ns
##  8 length 150-199~ 200-29~    20    70     1.92  5.54e- 2   6.10e-1 ns
##  9 length 150-199~ 300-34~    20    96     1.73  8.31e- 2   8.31e-1 ns
## 10 length 150-199~ 350-39~    20    86     4.01  6.12e- 5   1.47e-3 **
## # ... with 18 more rows
```

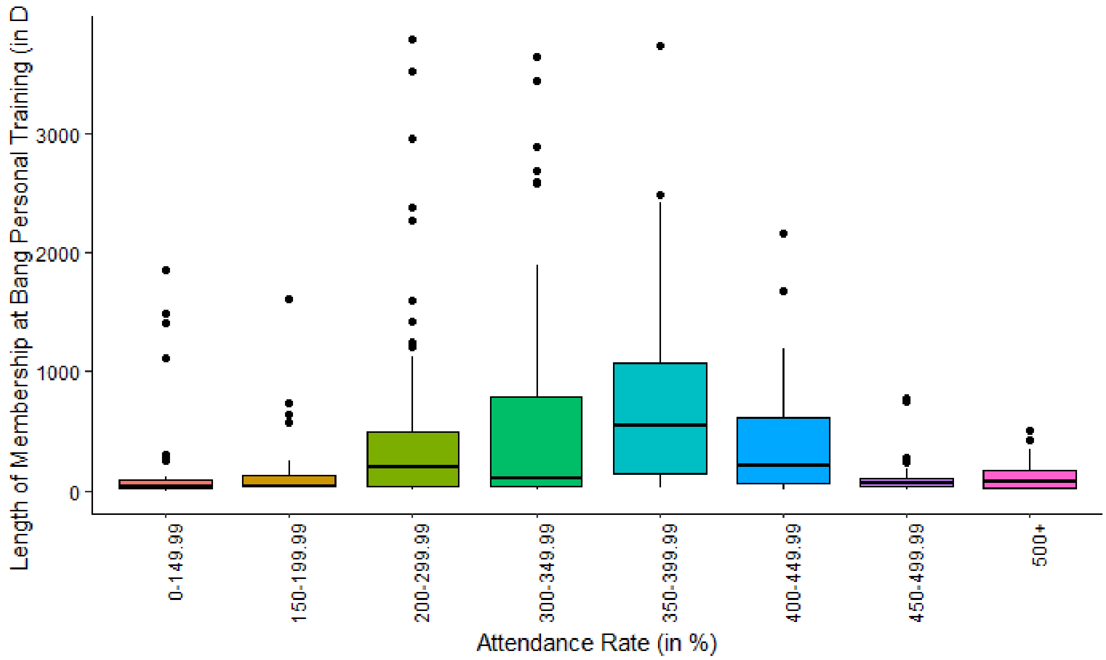


Figure48. Length of Membership by Monthly Membership Rate (H = 71.33, p < 0.001)

Retention Status at 3-Months, 6-Months and 12-Months

Looking at membership retention across the three time points, it was found that the rates of retention are 54.6% at 3 months, 46.8% at 6 months and 33.8% at 12 months. Notably it was found that retention status significantly differed between employment sectors at 3 and 6 months. This difference was also noted with respect to membership types as well. It was also found that those that had a higher attendance rate also were more likely to have remained a member at Bang. This relationship appeared to not have changed across all three time point. Similarly, there were also significant differences with respect to average monthly membership rates with greater retention rates across all time points with those with 350/month to 399/month as a membership rate.

Looking at the impact of email interactions, those that had continued their membership reported more billing-related email interactions than those that did not. Further look into the other types of email interactions, there were greater percentage of CX and scheduling-related email interactions amongst those that had maintained their membership at Bang Personal Training at each time point. However, there was a significantly lower percentage of service-related email interactions amongst those that had maintained their membership as compared to those that did not. Overall, those that retained their Bang Personal Training membership were found to have greater email interactions than those that did not across all time points. However, there is a lower number of email interaction per month amongst those that retained their membership, which was evident across all time points.

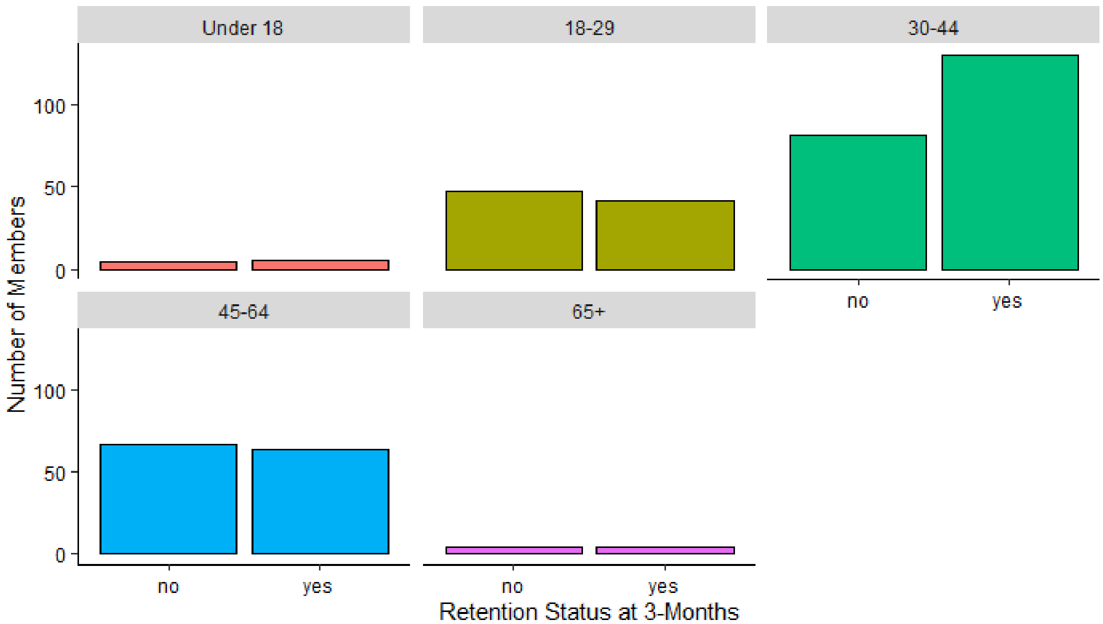


Figure49. Continuous Membership Retention at 3-Months Across Age Groups ($\chi 2 = 8.28$, $p = 0.082$)

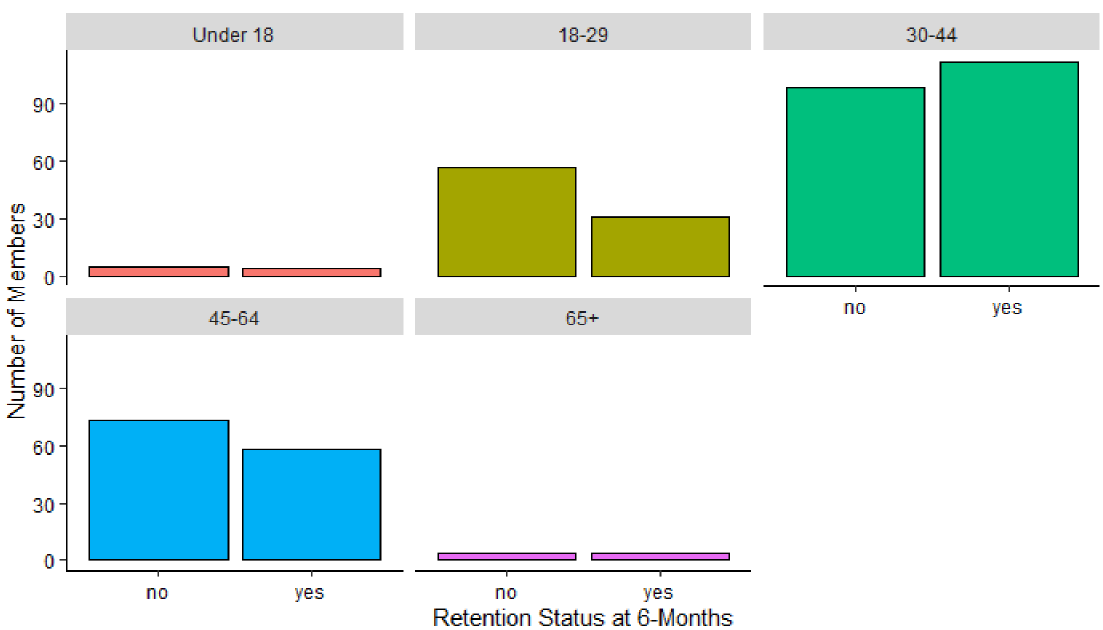


Figure50. Continuous Membership Retention at 6-Months Across Age Groups ($\chi 2 = 8.47$, $p = 0.076$)

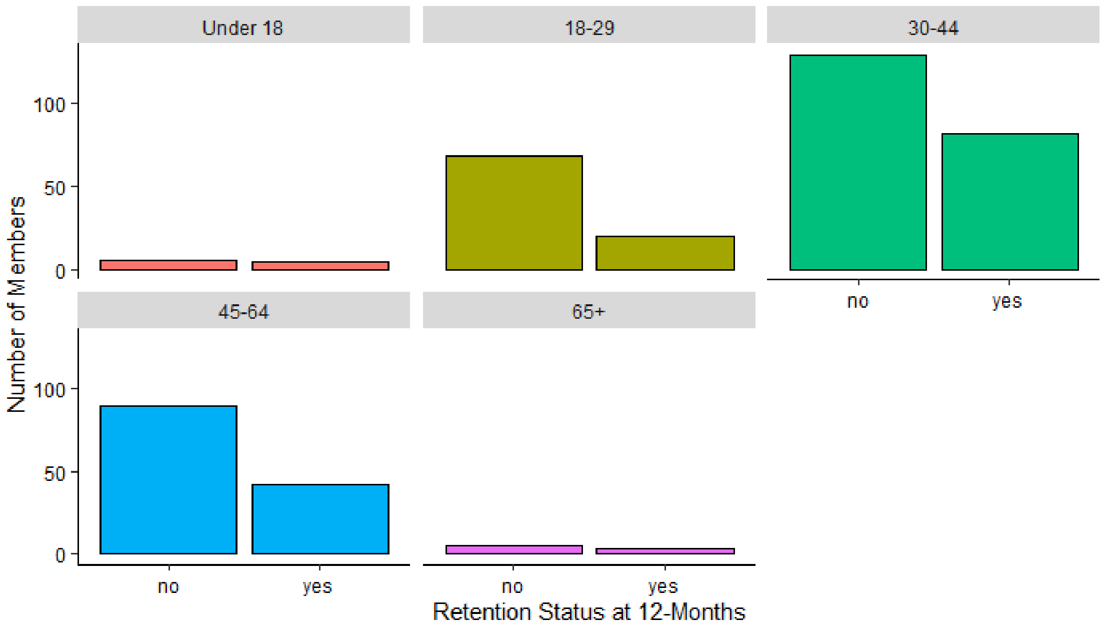


Figure51. Continuous Membership Retention at 12-Months Across Age Groups ($\chi2$ = 7.92, p = 0.094)

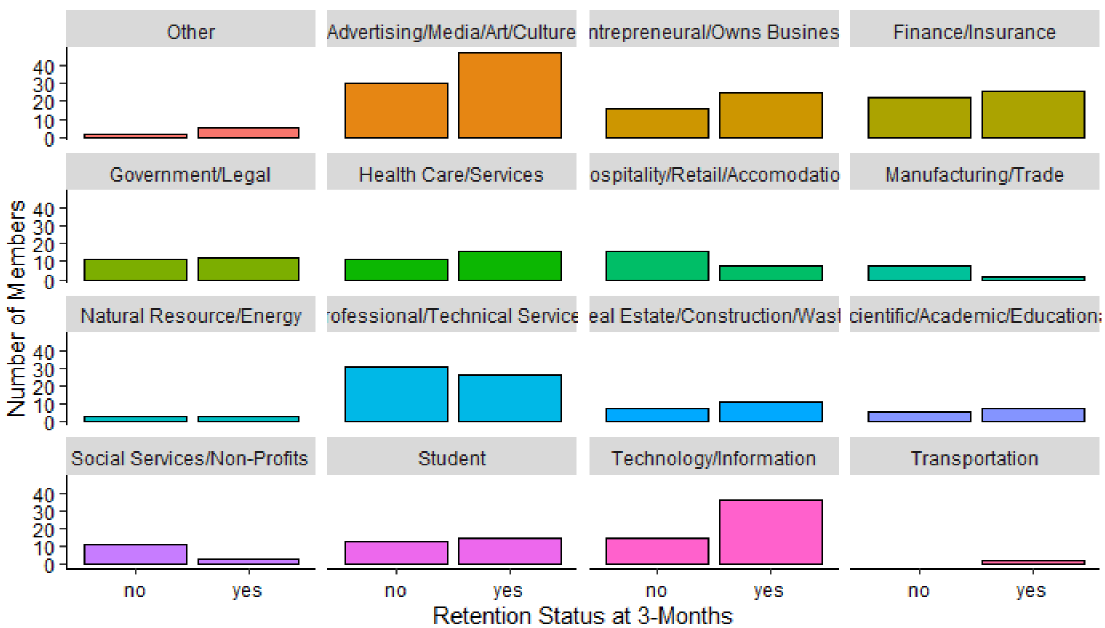


Figure52. Continuous Membership Retention at 3-Months Across Employment Sectors ($\chi2$ = 29.48, p = 0.014)

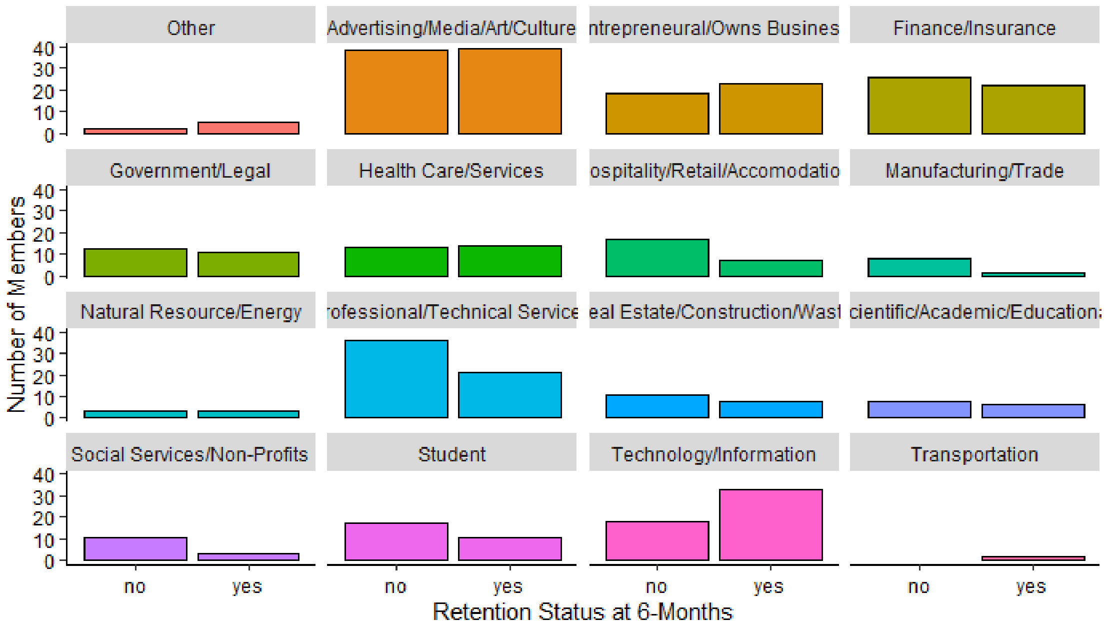


Figure53. Continuous Membership Retention at 6-Months Across Employment Sectors ($\chi2 = 27.14$, $p = 0.028$)

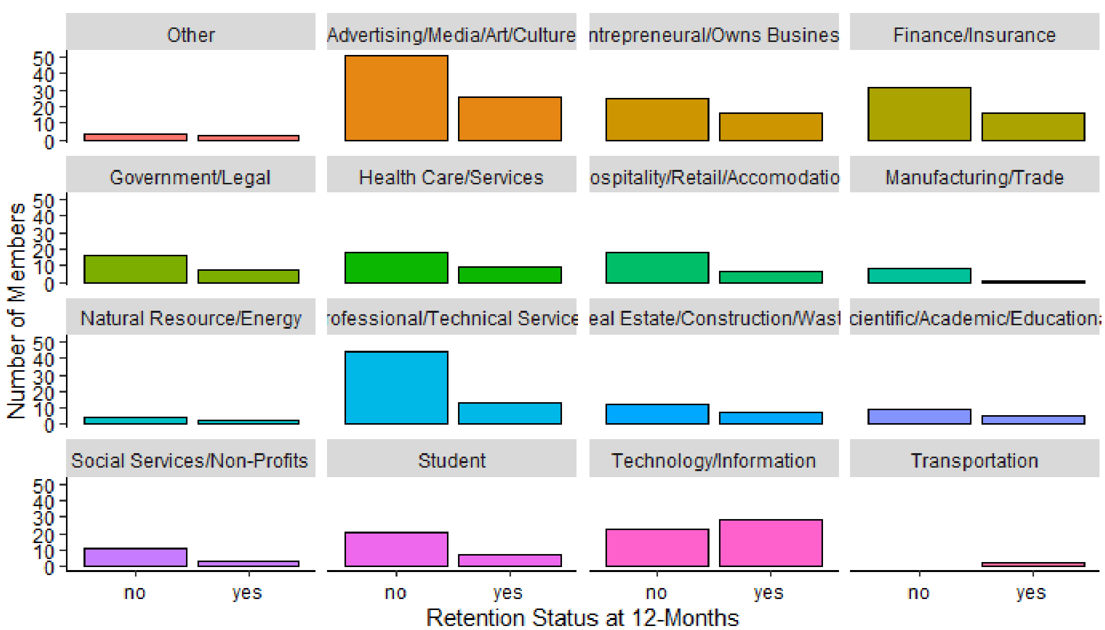


Figure54. Continuous Membership Retention at 12-Months Across Employment Sectors ($\chi2 = 22.96$, $p = 0.085$)

Figure55. Continuous Membership Retention at 3-Months Across Membership Types ($\chi2$ = 16.99, p = 0.009)

Figure56. Continuous Membership Retention at 6-Months Across Membership Types ($\chi2$ = 18.40, p = 0.005)

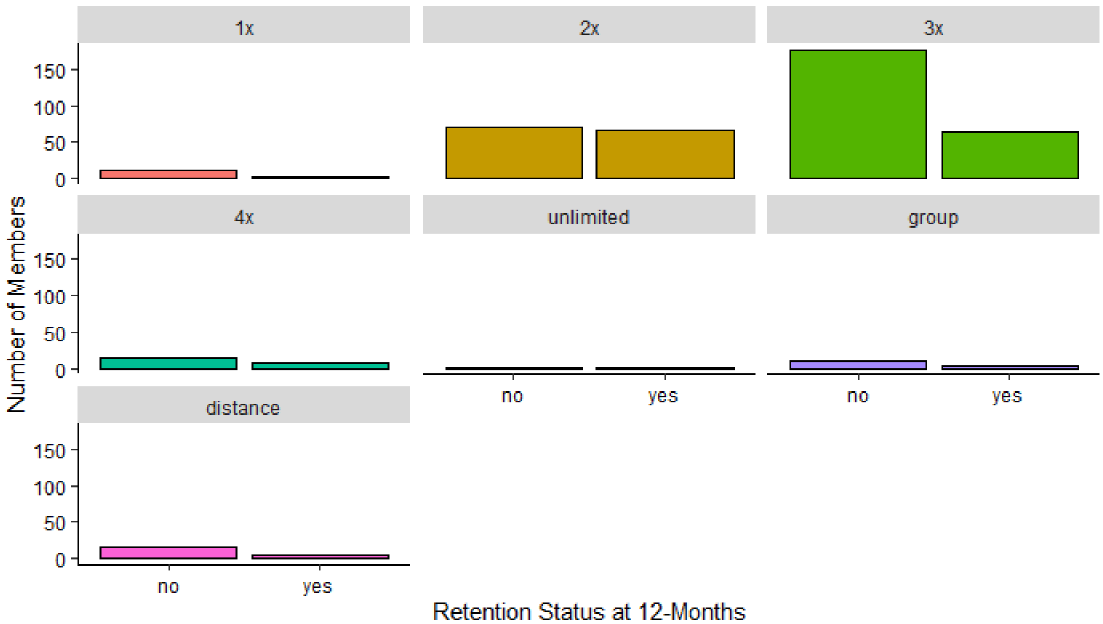


Figure57. Continuous Membership Retention at 12-Months Across Membership Types ($\chi2$ = 22.02, p = 0.001)

```
clean_bang_final %>% wilcox_test(attendance_rate ~ retention_3m) %>%
add_significance()

## # A tibble: 1 x 8
##   .y.             group1 group2    n1    n2 statistic       p p.signif
##   <chr>           <chr>  <chr>  <int> <int>     <dbl>   <dbl> <chr>
## 1 attendance_rate no     yes      203   244    20526. 0.00179 **

clean_bang_final %>% wilcox_test(attendance_rate ~ retention_6m) %>%
add_significance()

## # A tibble: 1 x 8
##   .y.             group1 group2    n1    n2 statistic       p p.signif
##   <chr>           <chr>  <chr>  <int> <int>     <dbl>   <dbl> <chr>
## 1 attendance_rate no     yes      238   209    20674. 0.00203 **

clean_bang_final %>% wilcox_test(attendance_rate ~ retention_12m) %>%
add_significance()

## # A tibble: 1 x 8
##   .y.             group1 group2    n1    n2 statistic       p p.signif
##   <chr>           <chr>  <chr>  <int> <int>     <dbl>   <dbl> <chr>
## 1 attendance_rate no     yes      296   151    18576. 0.00345 **

kruskal.test(attendance_rate[retention_status == "yes"] ~
retention_type[retention_status == "yes"], data =
clean_bang_longer_retention)

##
##  Kruskal-Wallis rank sum test
##
```

```
## data:  attendance_rate[retention_status == "yes"] by
retention_type[retention_status == "yes"]
## Kruskal-Wallis chi-squared = 0.44316, df = 2, p-value = 0.8013

dunnTest(attendance_rate[retention_status == "yes"] ~
retention_type[retention_status == "yes"], data =
clean_bang_longer_retention, method = 'holm')

## Dunn (1964) Kruskal-Wallis multiple comparison

##   p-values adjusted with the Holm method.

##                      Comparison          Z    P.unadj      P.adj
## 1 retention_12m - retention_3m  0.6618624 0.5080594 1.0000000
## 2 retention_12m - retention_6m  0.4535289 0.6501679 1.0000000
## 3  retention_3m - retention_6m -0.2131715 0.8311932 0.8311932
```

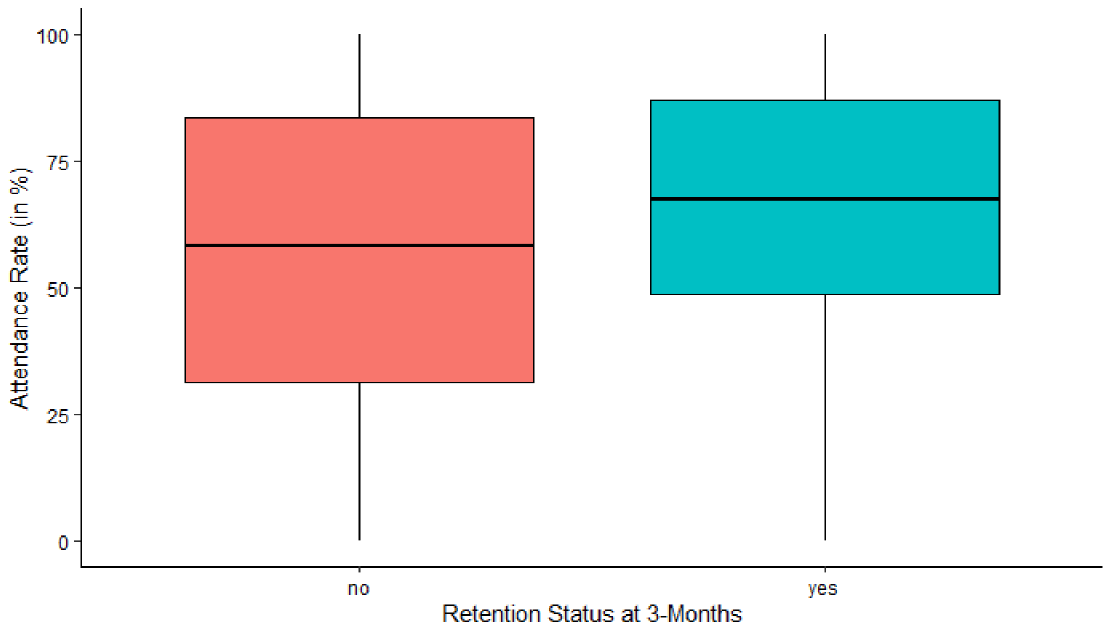


Figure58. Attendance Rate of Bang Personal Training Members by 3-Month Membership Retention Status (W = 20526, p = 0.002)

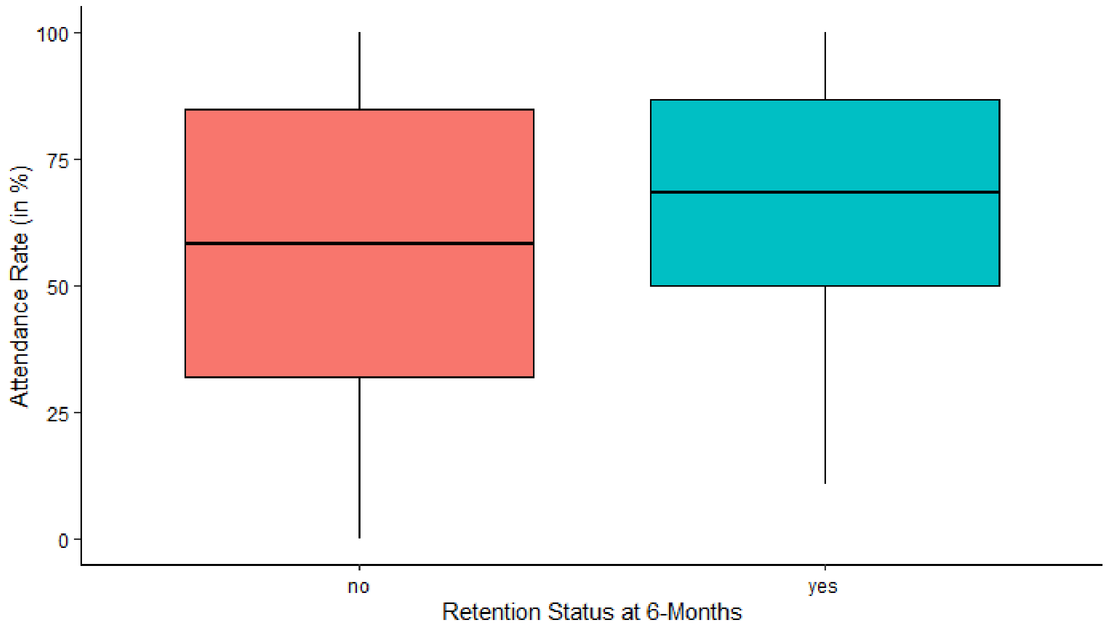


Figure59. Attendance Rate of Bang Personal Training Members by 6-Month Membership Retention Status (W = 20674, p = 0.002)

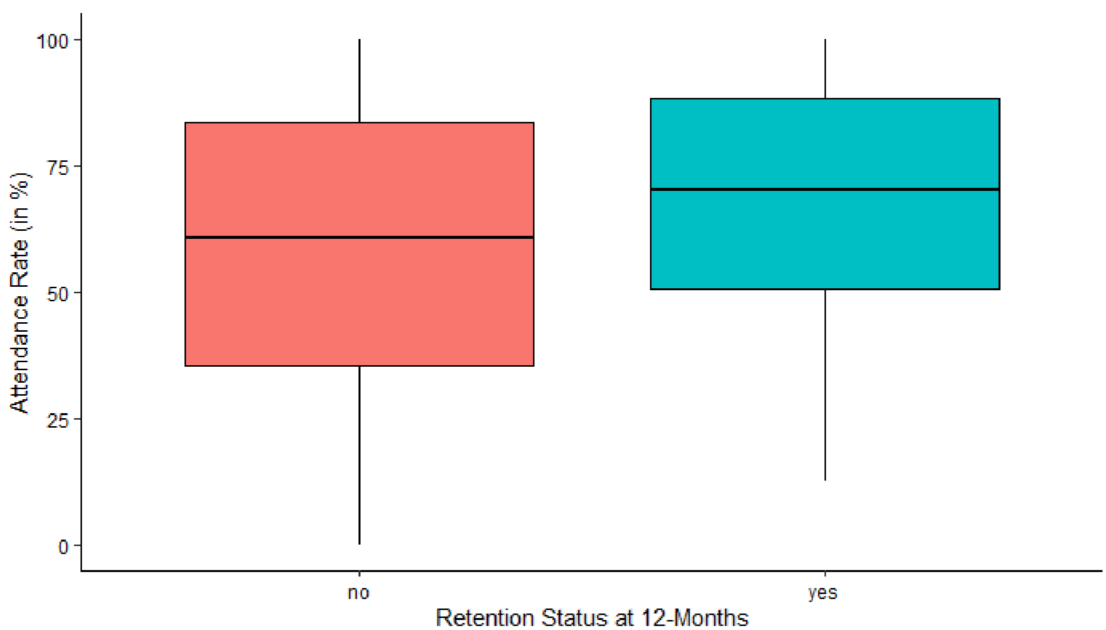


Figure60. Attendance Rate of Bang Personal Training Members by 12-Month Membership Retention Status (W = 18576, p = 0.003)

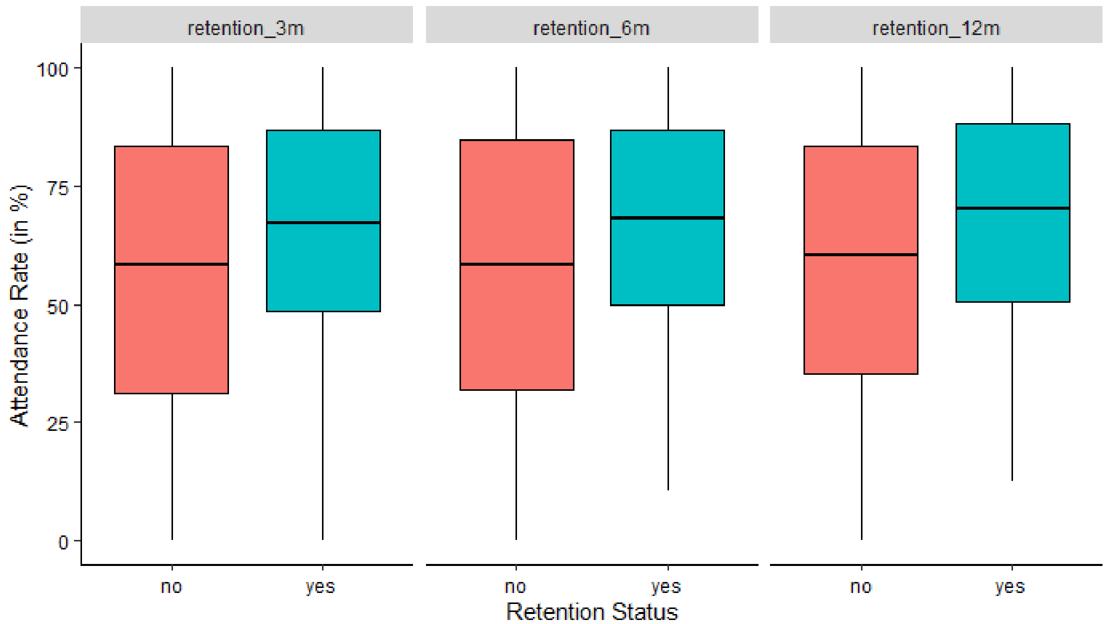


Figure61. Attendance Rate by Continuous Retention Status of Bang Personal Training Members Across 3-, 6- and 12-Months (H = 0.443, p = 0.801)

```
chisq.test(clean_bang_final$monthly_rate_group,
clean_bang_final$retention_3m)

## Warning in chisq.test(clean_bang_final$monthly_rate_group,
## clean_bang_final$retention_3m): Chi-squared approximation may be incorrect

##
##  Pearson's Chi-squared test
##
## data:  clean_bang_final$monthly_rate_group and
clean_bang_final$retention_3m
## X-squared = 61.448, df = 11, p-value = 4.986e-09
```

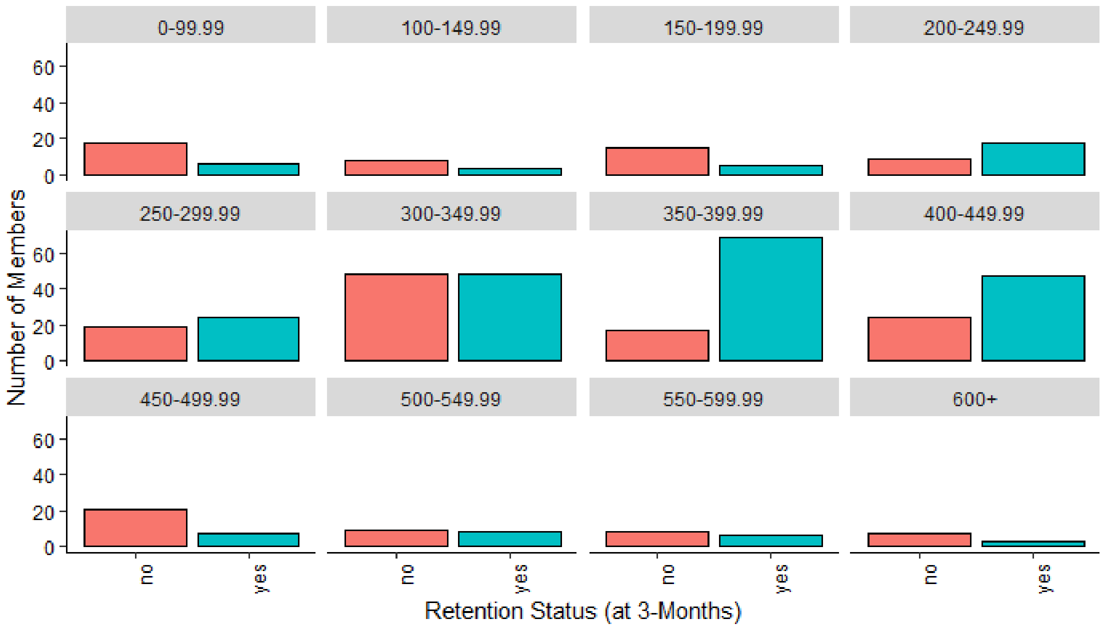


Figure62. Retention Status of Bang Personal Training Members at 3-Months by Attendance Rate ($\chi 2 = 61.45$, $p < 0.001$)

```
chisq.test(clean_bang_final$monthly_rate_group,
clean_bang_final$retention_6m)

## Warning in chisq.test(clean_bang_final$monthly_rate_group,
## clean_bang_final$retention_6m): Chi-squared approximation may be incorrect

##
##  Pearson's Chi-squared test
##
## data:  clean_bang_final$monthly_rate_group and
clean_bang_final$retention_6m
## X-squared = 58.152, df = 11, p-value = 2.04e-08
```

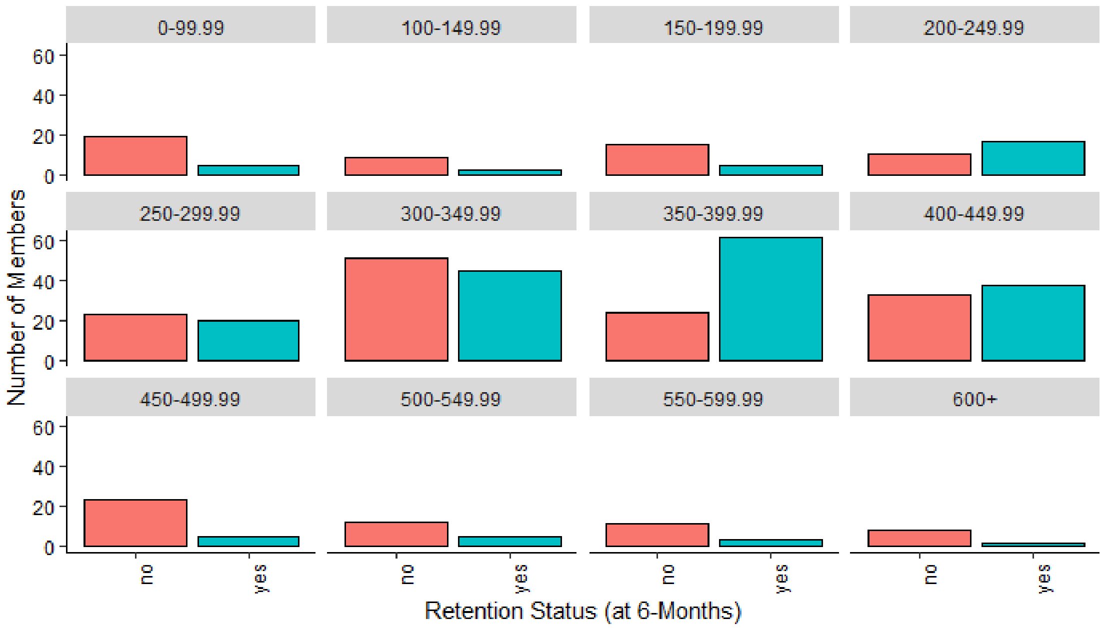


Figure63. Retention Status of Bang Personal Training Members at 6-Months by Attendance Rate ($\chi2$ = 58.15, $p < 0.001$)

```
chisq.test(clean_bang_final$monthly_rate_group,
clean_bang_final$retention_12m)

## Warning in chisq.test(clean_bang_final$monthly_rate_group,
## clean_bang_final$retention_12m): Chi-squared approximation may be
incorrect

##
##  Pearson's Chi-squared test
##
## data:  clean_bang_final$monthly_rate_group and
clean_bang_final$retention_12m
## X-squared = 57.036, df = 11, p-value = 3.277e-08
```

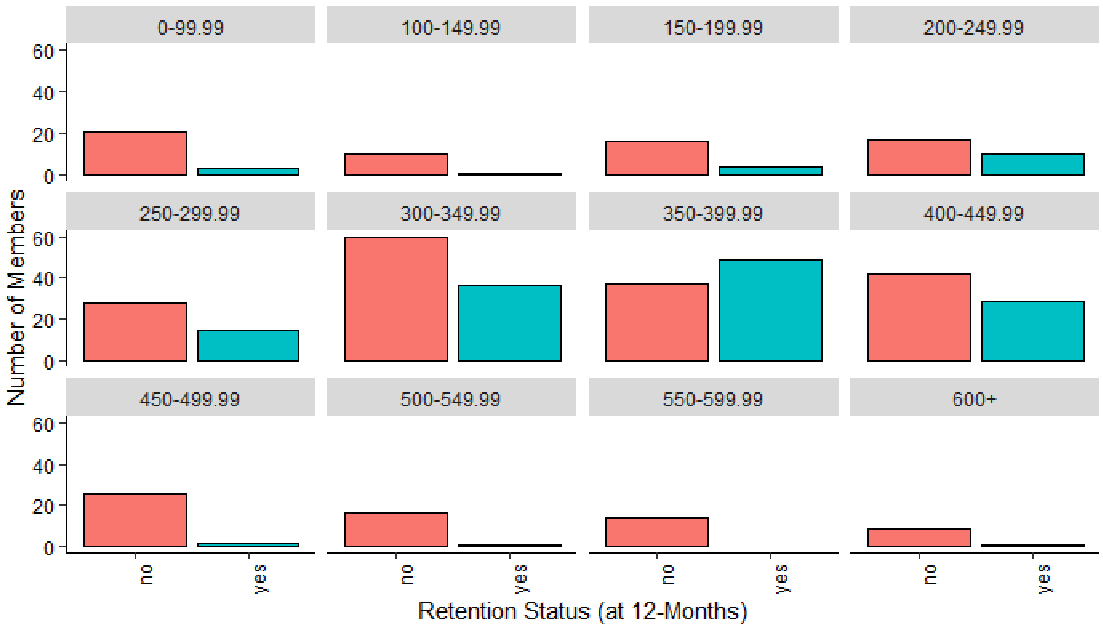


Figure64. Retention Status of Bang Personal Training Members at 12-Months by Attendance Rate ($\chi 2$ = 57.04, $p < 0.001$)

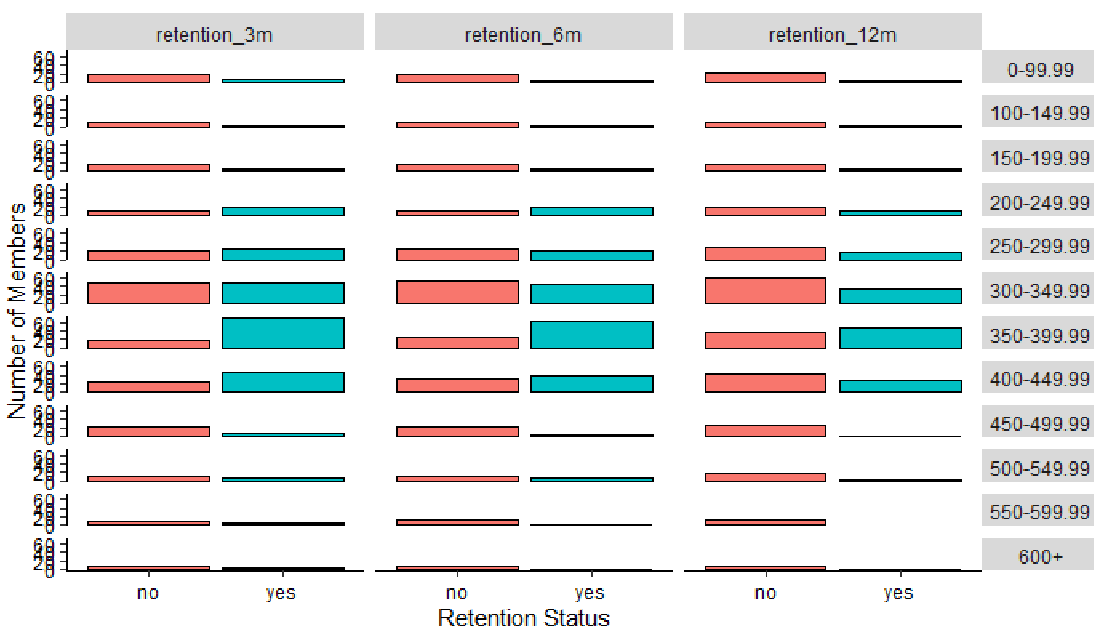


Figure65. Continuous Retention Status of Bang Personal Training Members Across 3-, 6- and 12-Months by Monthly Membership Rates.

```
chisq.test(clean_bang_final$retention_3m, clean_bang_final$num_billing_issue)

##
##  Pearson's Chi-squared test
##
## data:  clean_bang_final$retention_3m and
clean_bang_final$num_billing_issue
## X-squared = 22.235, df = 2, p-value = 1.485e-05
```

```r
chisq.test(clean_bang_final$retention_6m, clean_bang_final$num_billing_issue)

##
##  Pearson's Chi-squared test
##
## data:  clean_bang_final$retention_6m and
clean_bang_final$num_billing_issue
## X-squared = 26.074, df = 2, p-value = 2.178e-06

chisq.test(clean_bang_final$retention_12m,
clean_bang_final$num_billing_issue)

##
##  Pearson's Chi-squared test
##
## data:  clean_bang_final$retention_12m and
clean_bang_final$num_billing_issue
## X-squared = 29.337, df = 2, p-value = 4.262e-07
```

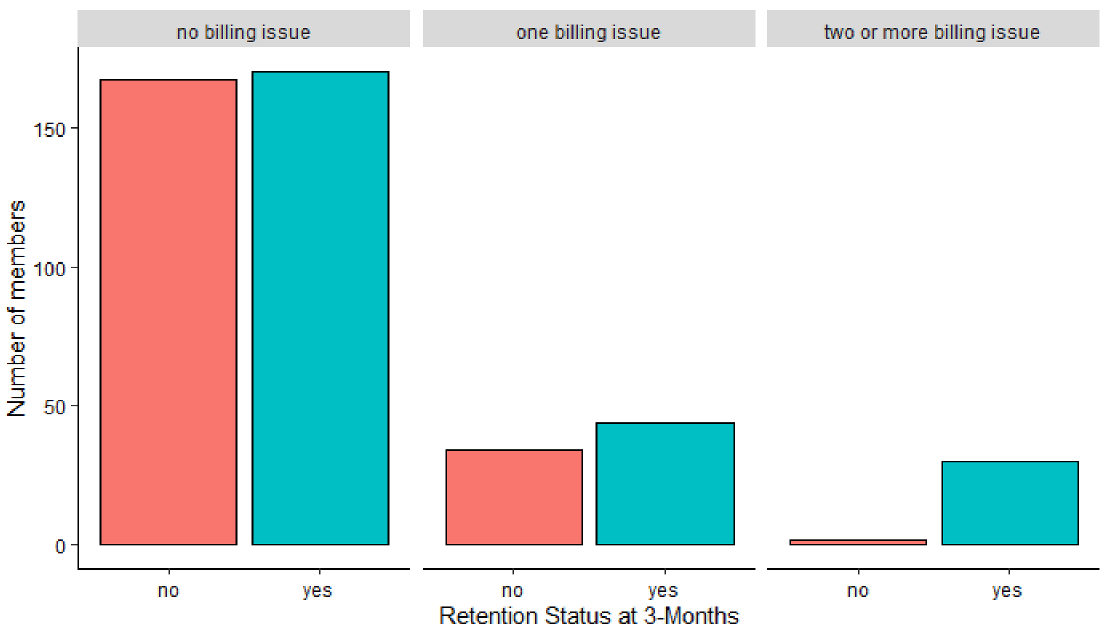


Figure66. Continuous Retention Status at 3 months by Number of Billing Issue ($\chi 2 = 22.24$, $p < 0.001$)

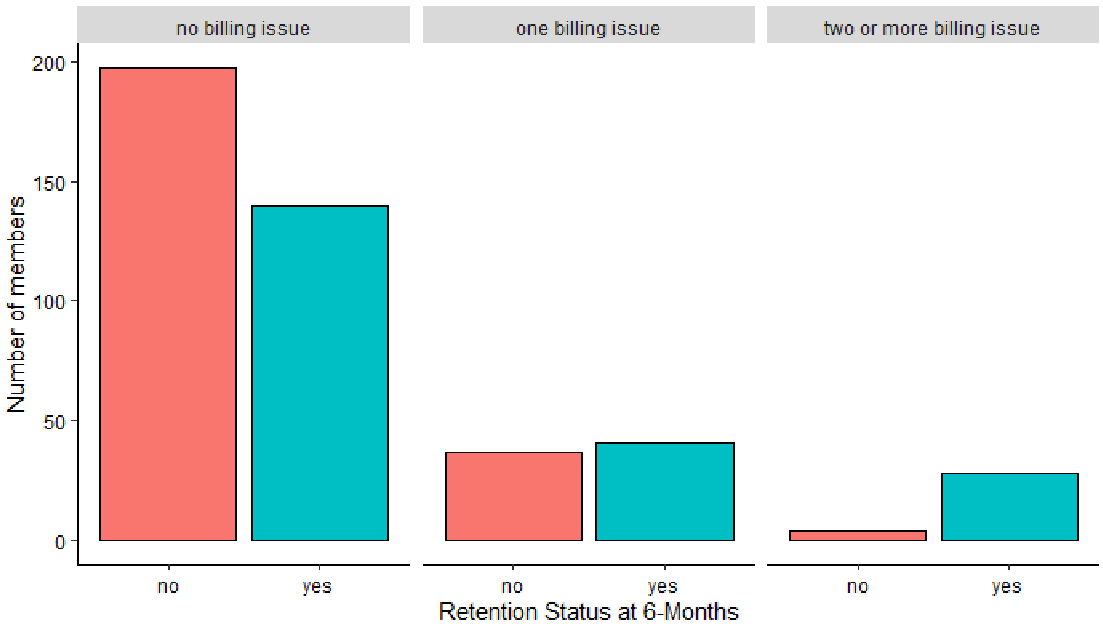


Figure67. Continuous Retention Status at 6 months by Number of Billing Issue
(χ2 = 26.07, p < 0.001)

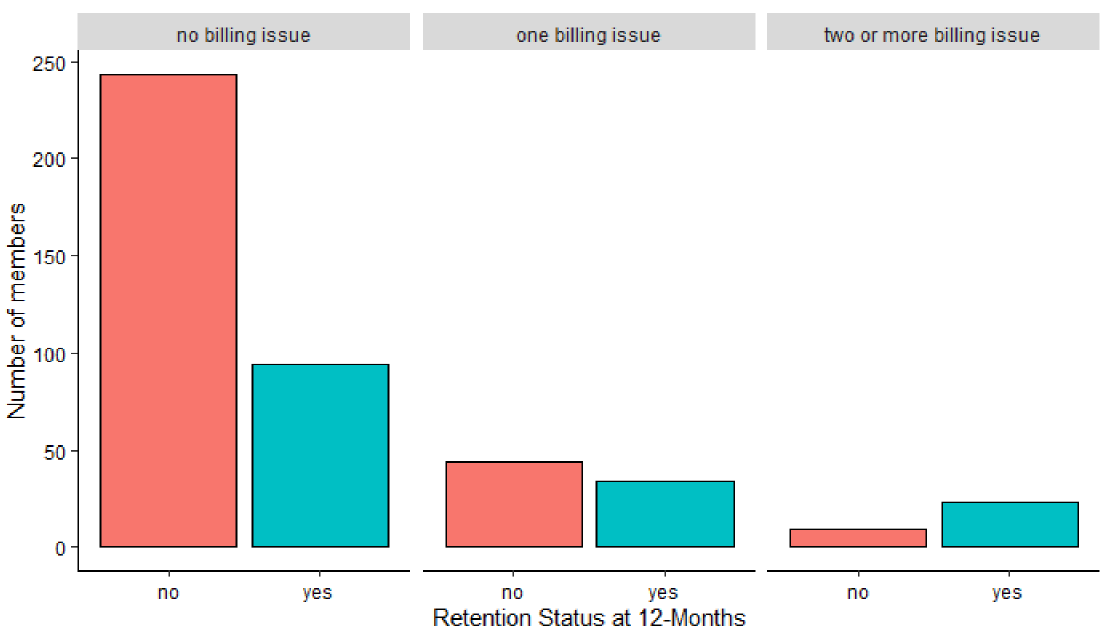


Figure68. Continuous Retention Status at 12 months by Number of Billing Issue
(χ2 = 29.34, p < 0.001)

```
clean_bang_final %>% wilcox_test(new_per_ticket_cx ~ retention_3m) %>%
add_significance()

## # A tibble: 1 x 8
##   .y.               group1 group2    n1    n2 statistic         p p.signif
##   <chr>             <chr>  <chr>  <int> <int>     <dbl>     <dbl> <chr>
## 1 new_per_ticket_cx no     yes      203   244    19284. 0.0000242 ****
```

```
clean_bang_final %>% wilcox_test(new_per_ticket_cx ~ retention_6m) %>%
add_significance()

## # A tibble: 1 x 8
##   .y.               group1 group2    n1    n2 statistic        p p.signif
##   <chr>             <chr>  <chr>  <int> <int>     <dbl>    <dbl> <chr>
## 1 new_per_ticket_cx no     yes      238   209     19388 0.000025 ****

clean_bang_final %>% wilcox_test(new_per_ticket_cx ~ retention_12m) %>%
add_significance()

## # A tibble: 1 x 8
##   .y.               group1 group2    n1    n2 statistic          p
p.signif
##   <chr>             <chr>  <chr>  <int> <int>     <dbl>      <dbl> <chr>
## 1 new_per_ticket_cx no     yes      296   151    16706. 0.00000478 ****

kruskal.test(new_per_ticket_cx[retention_status == "yes"] ~
retention_type[retention_status == "yes"], data =
clean_bang_longer_retention)

##
##  Kruskal-Wallis rank sum test
##
## data:  new_per_ticket_cx[retention_status == "yes"] by
retention_type[retention_status == "yes"]
## Kruskal-Wallis chi-squared = 0.95568, df = 2, p-value = 0.6201

dunnTest(new_per_ticket_cx[retention_status == "yes"] ~
retention_type[retention_status == "yes"], data =
clean_bang_longer_retention, method = 'holm')

## Dunn (1964) Kruskal-Wallis multiple comparison
##   p-values adjusted with the Holm method.

##                     Comparison          Z   P.unadj     P.adj
## 1 retention_12m - retention_3m  0.9611324 0.3364856 1.0000000
## 2 retention_12m - retention_6m  0.7186186 0.4723760 0.9447519
## 3  retention_3m - retention_6m -0.2415444 0.8091332 0.8091332
```

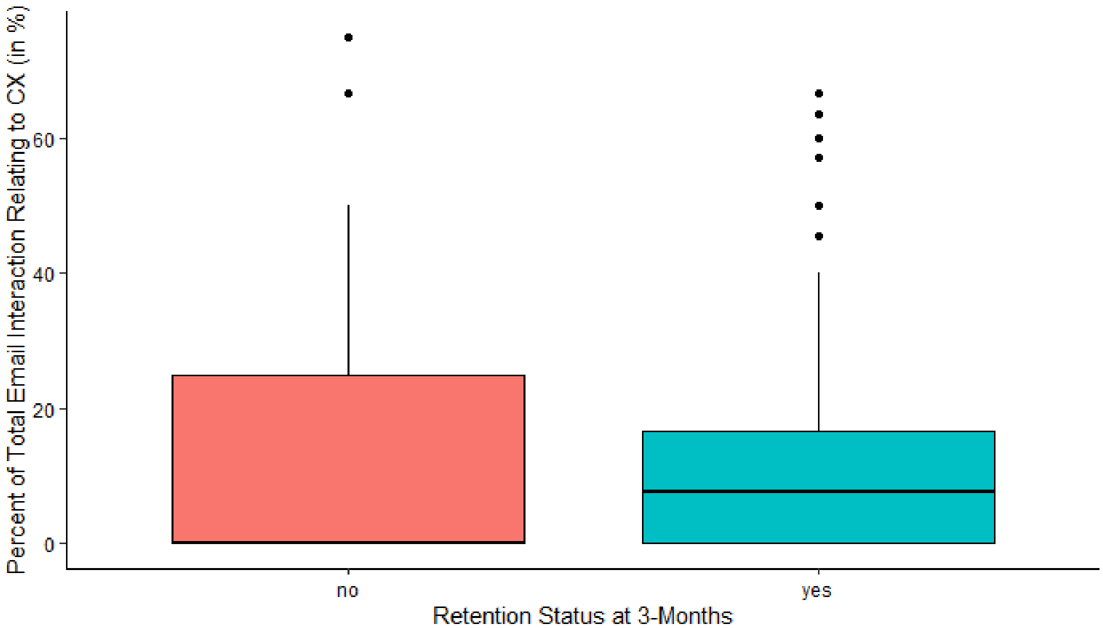


Figure69. Continuous Retention Status at 3-Months of Bang Personal Training Members By Percentage of Email Interactions Relating to CX (W = 19284.5, p < 0.001)

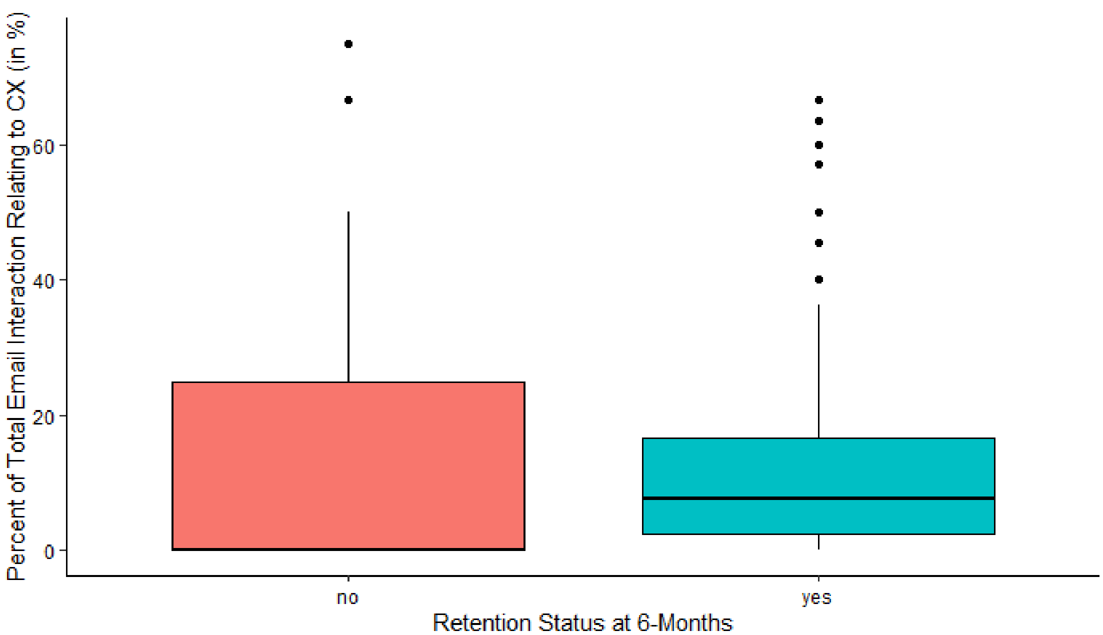


Figure70. Continuous Retention Status at 6-Months of Bang Personal Training Members By Percentage of Email Interactions Relating to CX (W = 19388, p < 0.001)

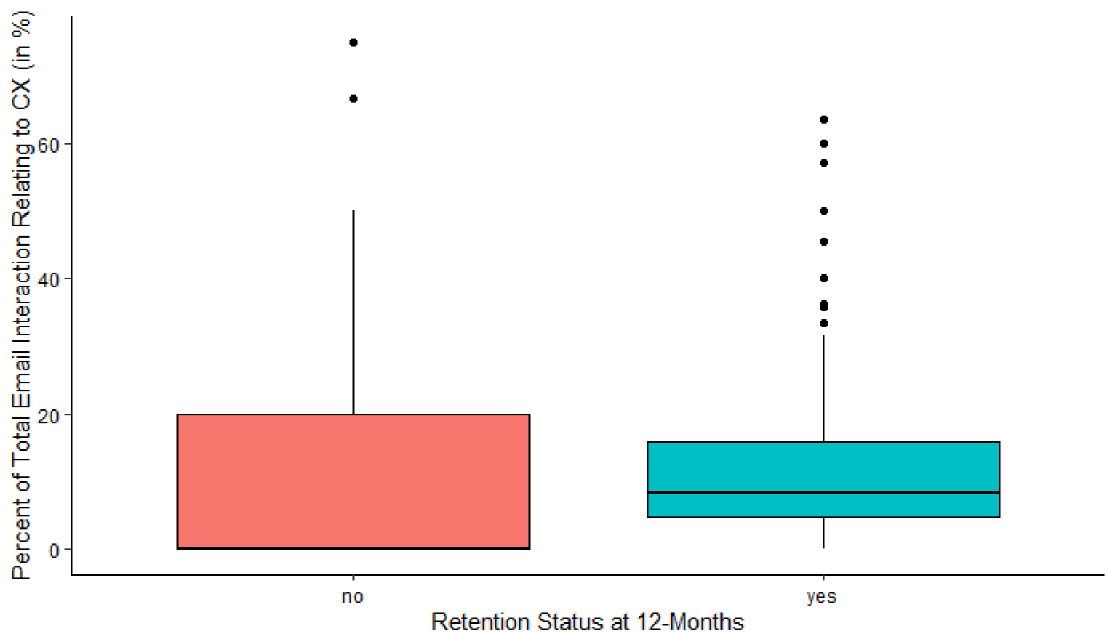


Figure71. Continuous Retention Status at 12-Months of Bang Personal Training Members By Percentage of Email Interactions Relating to CX (W = 16706.5, p < 0.001)

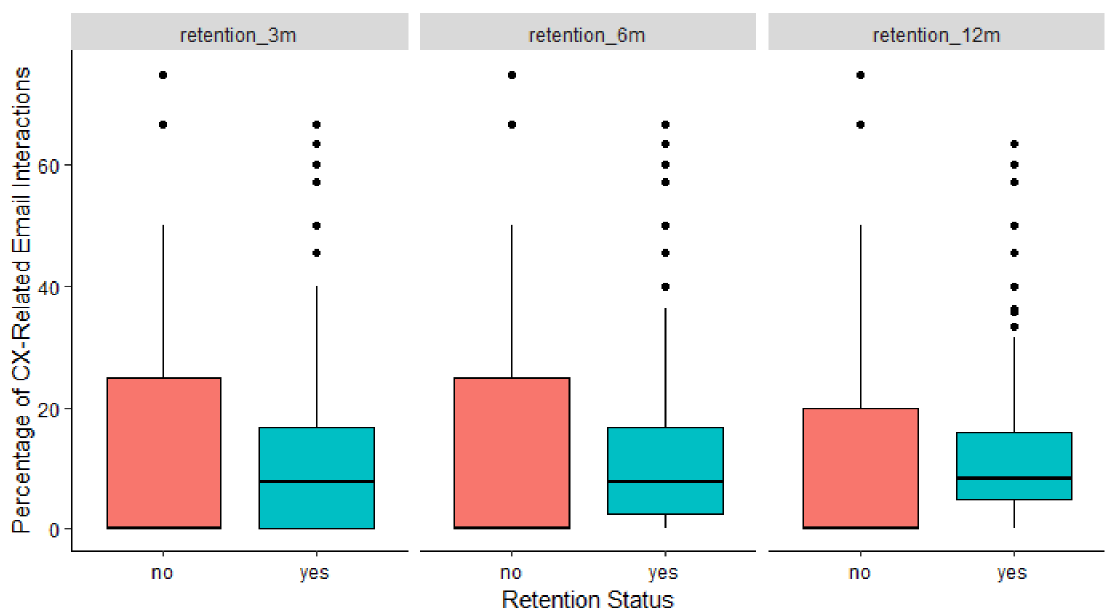


Figure72. Percentage of Email Interactions Relating to CX by Continuous Retention Status of Bang Personal Training Members Across 3-, 6- and 12-Months (H = 0.956, p = 0.602)

```
clean_bang_final %>% wilcox_test(new_per_ticket_scheduling ~ retention_3m)
%>% add_significance()

## # A tibble: 1 x 8
##   .y.                          group1 group2    n1    n2 statistic        p
p.signif
##   <chr>                        <chr>  <chr>  <int> <int>     <dbl>    <dbl>
<chr>
```

```
## 1 new_per_ticket_scheduli~ no     yes      203   244     13837 3.52e-16
****

clean_bang_final %>% wilcox_test(new_per_ticket_scheduling ~ retention_6m)
%>% add_significance()

## # A tibble: 1 x 8
##   .y.                      group1 group2    n1    n2 statistic        p
p.signif
##   <chr>                    <chr>  <chr>  <int> <int>     <dbl>    <dbl>
<chr>
## 1 new_per_ticket_scheduli~ no     yes      238   209    12250. 5.66e-21
****

clean_bang_final %>% wilcox_test(new_per_ticket_scheduling ~ retention_12m)
%>% add_significance()

## # A tibble: 1 x 8
##   .y.                      group1 group2    n1    n2 statistic        p
p.signif
##   <chr>                    <chr>  <chr>  <int> <int>     <dbl>    <dbl>
<chr>
## 1 new_per_ticket_scheduli~ no     yes      296   151      9877 1.18e-22
****

kruskal.test(new_per_ticket_scheduling[retention_status == "yes"] ~
retention_type[retention_status == "yes"], data =
clean_bang_longer_retention)

##
##  Kruskal-Wallis rank sum test
##
## data:  new_per_ticket_scheduling[retention_status == "yes"] by
retention_type[retention_status == "yes"]
## Kruskal-Wallis chi-squared = 8.4598, df = 2, p-value = 0.01455

dunnTest(new_per_ticket_scheduling[retention_status == "yes"] ~
retention_type[retention_status == "yes"], data =
clean_bang_longer_retention, method = 'holm')

## Dunn (1964) Kruskal-Wallis multiple comparison
##   p-values adjusted with the Holm method.

##                      Comparison         Z      P.unadj      P.adj
## 1 retention_12m - retention_3m  2.907167 0.003647185 0.01094155
## 2 retention_12m - retention_6m  1.668665 0.095183755 0.19036751
## 3  retention_3m - retention_6m -1.302833 0.192631726 0.19263173
```

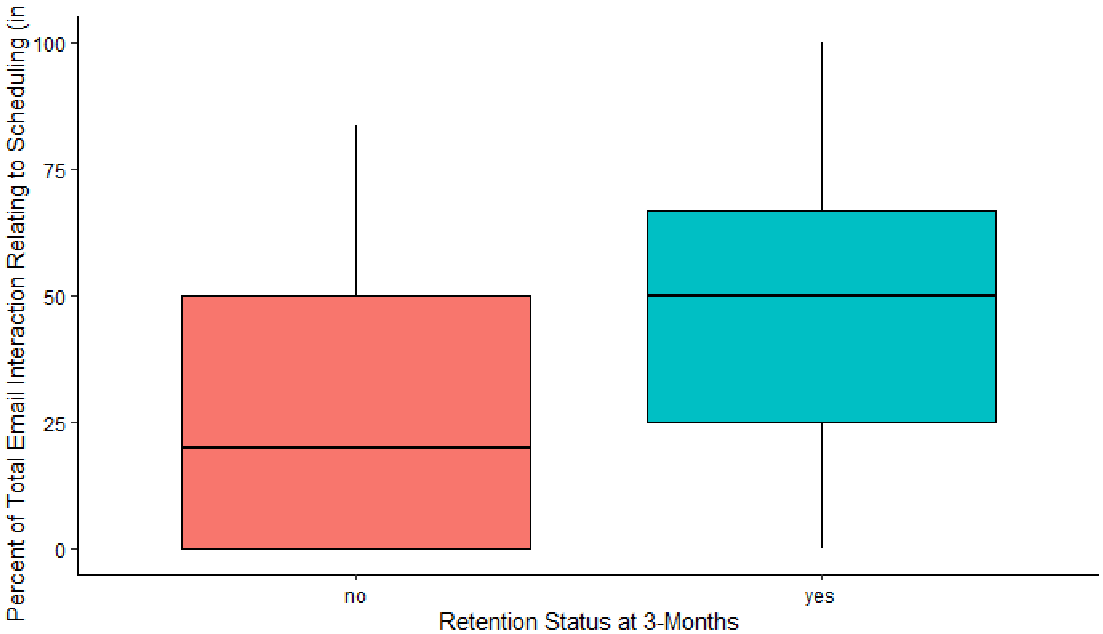


Figure73. Continuous Retention Status at 3-Months of Bang Personal Training Members By Percentage of Email Interactions Relating to Scheduling (W = 13837, p < 0.001)

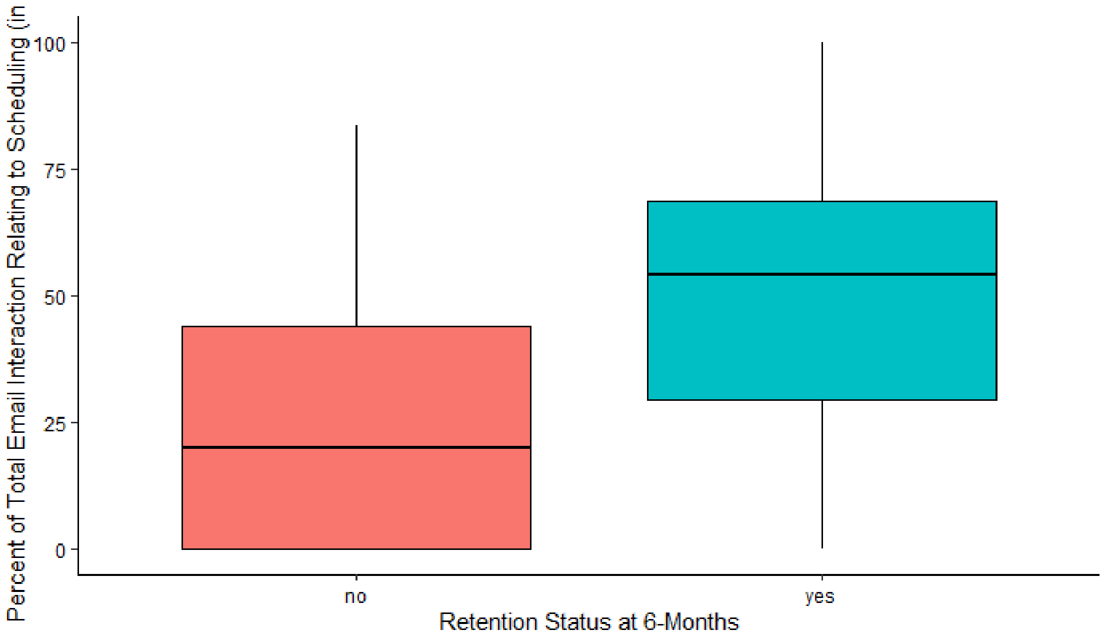


Figure74. Continuous Retention Status at 6-Months of Bang Personal Training Members By Percentage of Email Interactions Relating to Scheduling ($W = 12520.5$, $p < 0.001$)

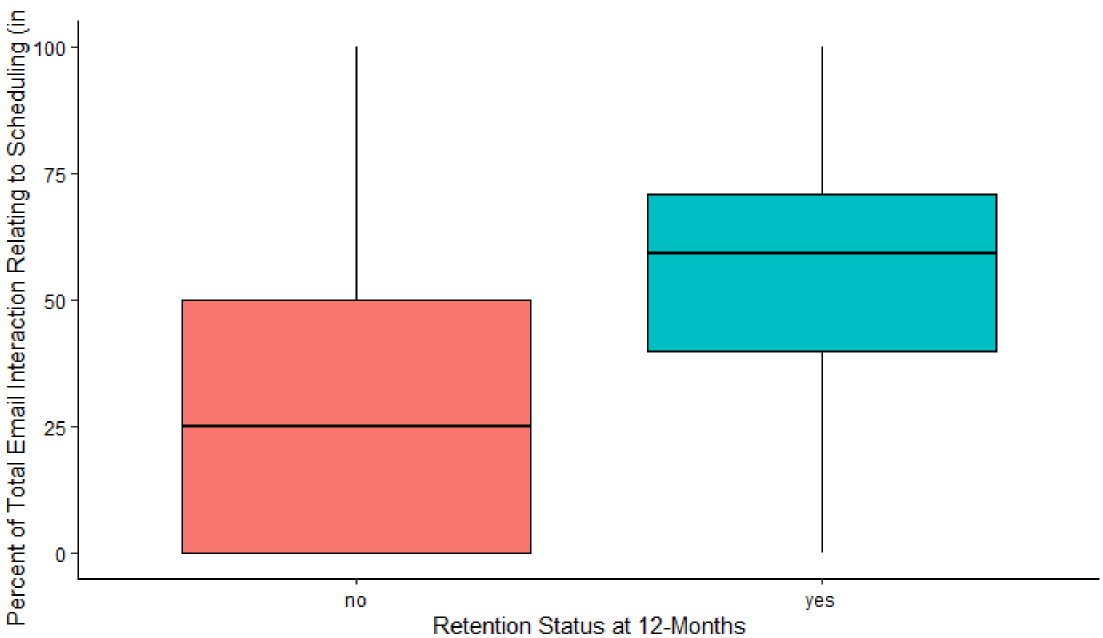
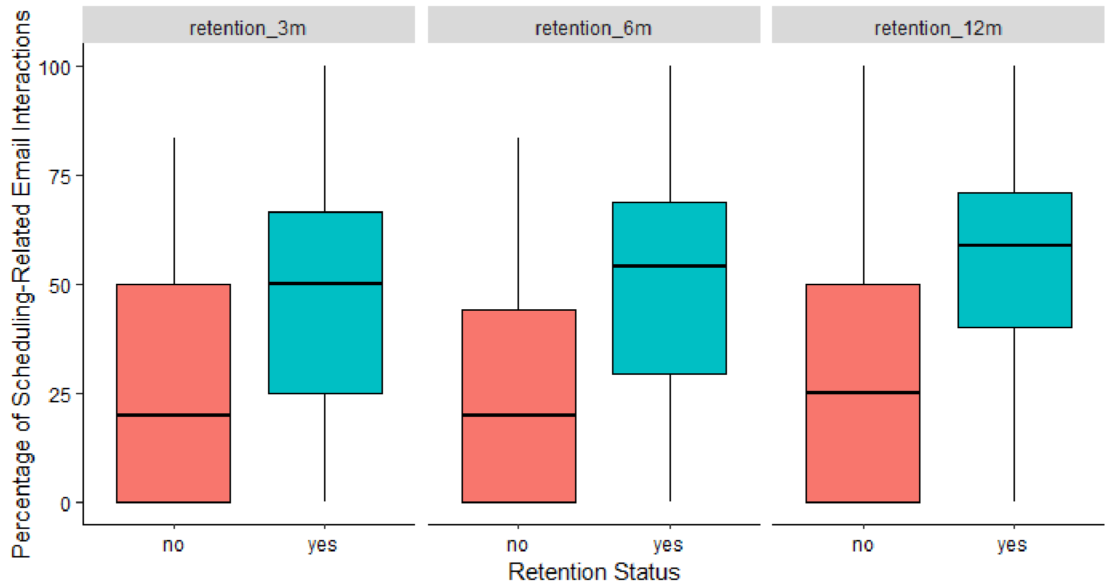


Figure75. Continuous Retention Status at 12-Months of Bang Personal Training Members By Percentage of Email Interactions Relating to Scheduling ($W = 9877$, $p < 0.001$)



. Percentage of Email Interactions Relating to Scheduling by Continuous Retention Status of Bang Personal Training Members
Across Retention Status 3-, 6- and 12-Months (H = 8.46, p = 0.015).
llowing Pairwise Comparisons, Greater Proportion of Scheduling-Related Email Interactions Observed Amongst Members that
Retain Membership at 12-Months than at 3-Months (Z = 2.91, p = 0.011)

```
clean_bang_final %>% wilcox_test(new_per_ticket_service ~ retention_3m) %>%
add_significance()

## # A tibble: 1 x 8
##   .y.                    group1 group2    n1    n2 statistic        p
p.signif
##   <chr>                  <chr>  <chr>  <int> <int>     <dbl>    <dbl>
<chr>
## 1 new_per_ticket_service no     yes      203   244    35410. 3.79e-15 ****

clean_bang_final %>% wilcox_test(new_per_ticket_service ~ retention_6m) %>%
add_significance()

## # A tibble: 1 x 8
##   .y.                    group1 group2    n1    n2 statistic        p
p.signif
##   <chr>                  <chr>  <chr>  <int> <int>     <dbl>    <dbl>
<chr>
## 1 new_per_ticket_service no     yes      238   209    36710. 2.64e-18 ****

clean_bang_final %>% wilcox_test(new_per_ticket_service ~ retention_12m) %>%
add_significance()

## # A tibble: 1 x 8
##   .y.                    group1 group2    n1    n2 statistic        p
p.signif
##   <chr>                  <chr>  <chr>  <int> <int>     <dbl>    <dbl>
```

```
<chr>
## 1 new_per_ticket_service no     yes      296   151    34180. 3.61e-20 ****
```

```
kruskal.test(new_per_ticket_service[retention_status == "yes"] ~
retention_type[retention_status == "yes"], data =
clean_bang_longer_retention)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  new_per_ticket_service[retention_status == "yes"] by
retention_type[retention_status == "yes"]
## Kruskal-Wallis chi-squared = 7.3388, df = 2, p-value = 0.02549
```

```
dunnTest(new_per_ticket_service[retention_status == "yes"] ~
retention_type[retention_status == "yes"], data =
clean_bang_longer_retention, method = 'holm')
```

```
## Dunn (1964) Kruskal-Wallis multiple comparison
##   p-values adjusted with the Holm method.
```

```
##                      Comparison         Z     P.unadj     P.adj
## 1 retention_12m - retention_3m -2.707895 0.006771134 0.0203134
## 2 retention_12m - retention_6m -1.684061 0.092169894 0.1843398
## 3  retention_3m - retention_6m  1.066470 0.286211367 0.2862114
```

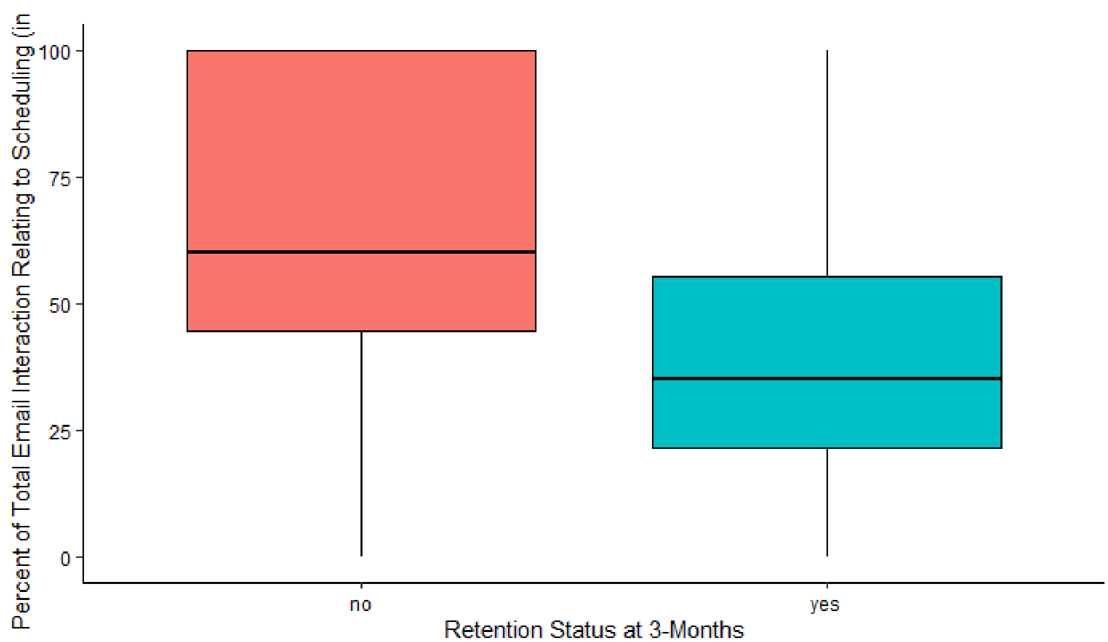


Figure77. Continuous Retention Status at 3-Months of Bang Personal Training Members By Percentage of Email Interactions Relating to Scheduling (W = 35410.5, p < 0.001)

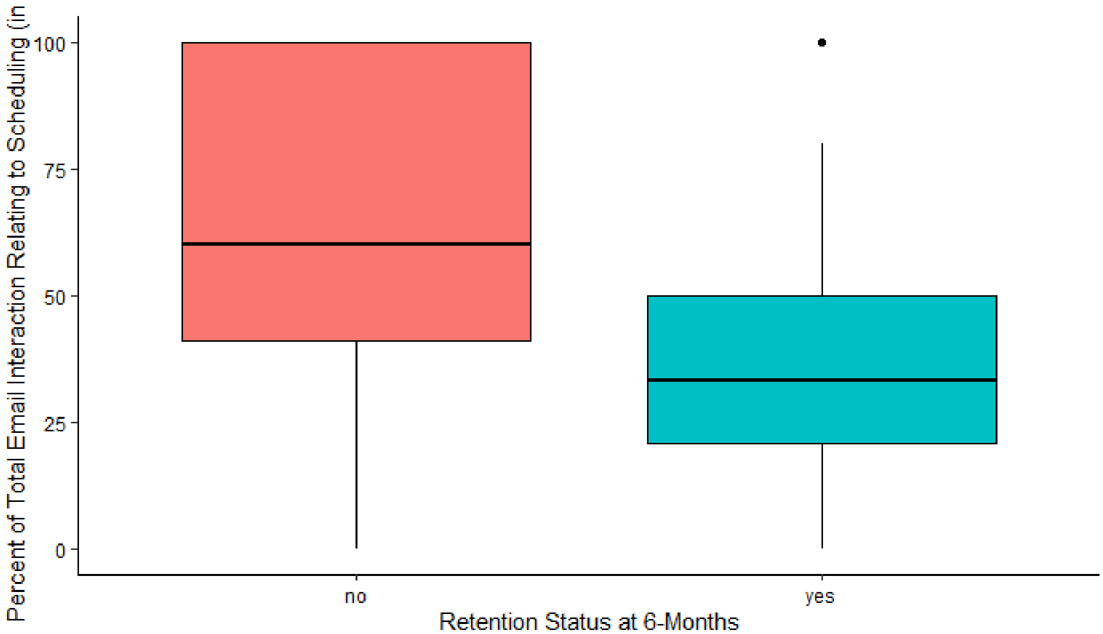


Figure78. Continuous Retention Status at 6-Months of Bang Personal Training Members By Percentage of Email Interactions Relating to Scheduling (W = 36710.5, p < 0.001)

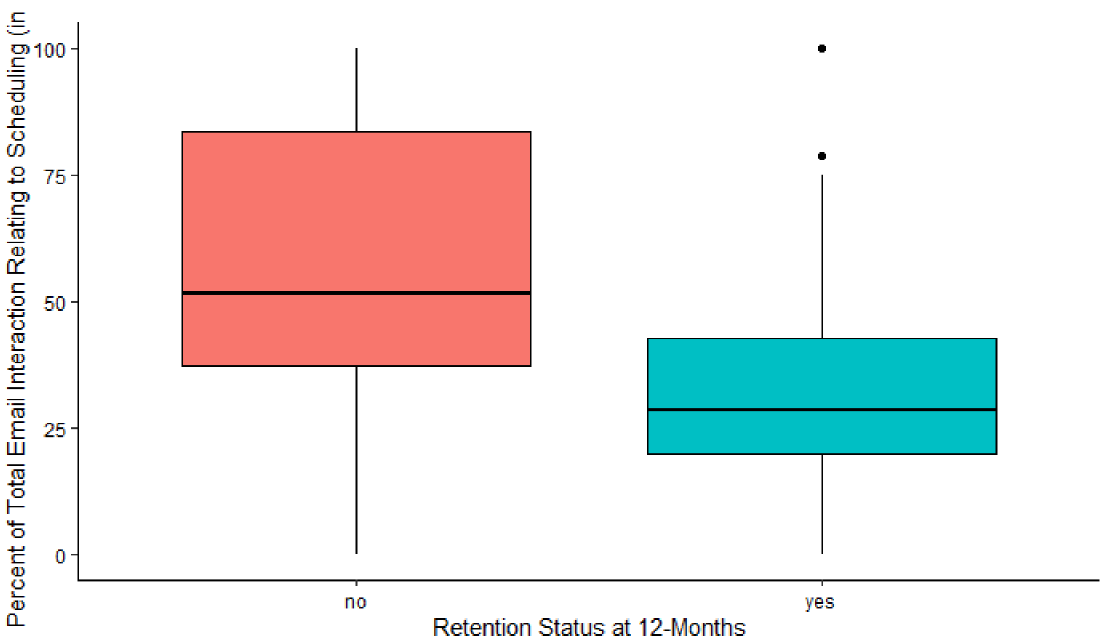


Figure79. Continuous Retention Status at 12-Months of Bang Personal Training Members By Percentage of Email Interactions Relating to Scheduling (W = 34179.5, p < 0.001)

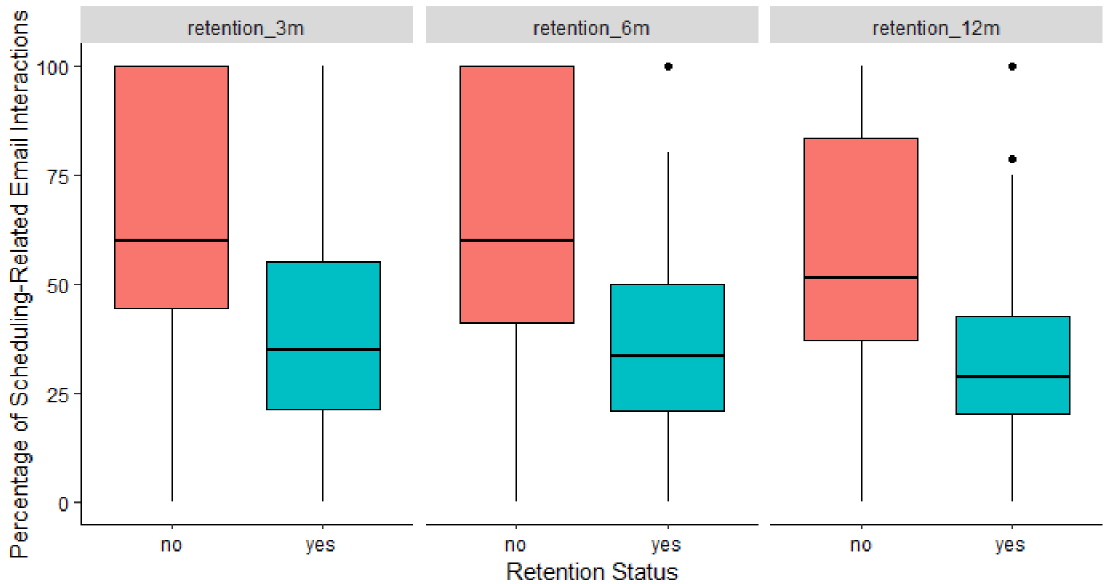


Figure80. Percentage of Email Interactions Relating to Service by
uous Retention Status of Bang Personal Training Members Across Retention Status 3-, 6- and 12-Months (H = 7.34, p = 0.025).
Following Pairwise Comparisons, Greater Proportion of Service-Related Email Interactions Observed Amongst
Members that Retain Membership at 3-Months than at 12-Months (Z = - 2.71 , p = 0.020)

```
clean_bang_final %>% wilcox_test(new_num_total ~ retention_3m) %>%
add_significance()

## # A tibble: 1 x 8
##   .y.           group1 group2    n1    n2 statistic        p p.signif
##   <chr>         <chr>  <chr>  <int> <int>     <dbl>    <dbl> <chr>
## 1 new_num_total no     yes      203   244      6059 2.49e-43 ****

clean_bang_final %>% wilcox_test(new_num_total ~ retention_6m) %>%
add_significance()

## # A tibble: 1 x 8
##   .y.           group1 group2    n1    n2 statistic        p p.signif
##   <chr>         <chr>  <chr>  <int> <int>     <dbl>    <dbl> <chr>
## 1 new_num_total no     yes      238   209     4900. 6.18e-49 ****

clean_bang_final %>% wilcox_test(new_num_total ~ retention_12m) %>%
add_significance()

## # A tibble: 1 x 8
##   .y.           group1 group2    n1    n2 statistic        p p.signif
##   <chr>         <chr>  <chr>  <int> <int>     <dbl>    <dbl> <chr>
## 1 new_num_total no     yes      296   151      3364 3.34e-49 ****

kruskal.test(new_num_total[retention_status == "yes"] ~
retention_type[retention_status == "yes"], data =
clean_bang_longer_retention)

##
##  Kruskal-Wallis rank sum test
##
```

```
## data:  new_num_total[retention_status == "yes"] by
retention_type[retention_status == "yes"]
## Kruskal-Wallis chi-squared = 21.333, df = 2, p-value = 2.331e-05

dunnTest(new_num_total[retention_status == "yes"] ~
retention_type[retention_status == "yes"], data =
clean_bang_longer_retention, method = 'holm')

## Dunn (1964) Kruskal-Wallis multiple comparison
##   p-values adjusted with the Holm method.

##                      Comparison         Z      P.unadj        P.adj
## 1 retention_12m - retention_3m  4.612334 3.981725e-06 1.194518e-05
## 2 retention_12m - retention_6m  2.957483 3.101621e-03 6.203241e-03
## 3  retention_3m - retention_6m -1.715611 8.623325e-02 8.623325e-02

clean_bang_final %>% wilcox_test(num_emails_month ~ retention_3m) %>%
add_significance()

## # A tibble: 1 x 8
##   .y.              group1 group2    n1    n2 statistic        p p.signif
##   <chr>            <chr>  <chr>  <int> <int>     <dbl>    <dbl> <chr>
## 1 num_emails_month no     yes      203   244    41870. 2.77e-36 ****

clean_bang_final %>% wilcox_test(num_emails_month ~ retention_6m) %>%
add_significance()

## # A tibble: 1 x 8
##   .y.              group1 group2    n1    n2 statistic        p p.signif
##   <chr>            <chr>  <chr>  <int> <int>     <dbl>    <dbl> <chr>
## 1 num_emails_month no     yes      238   209     40174 2.91e-29 ****

clean_bang_final %>% wilcox_test(num_emails_month ~ retention_12m) %>%
add_significance()

## # A tibble: 1 x 8
##   .y.              group1 group2    n1    n2 statistic        p p.signif
##   <chr>            <chr>  <chr>  <int> <int>     <dbl>    <dbl> <chr>
## 1 num_emails_month no     yes      296   151     32838 4.63e-16 ****

kruskal.test(num_emails_month[retention_status == "yes"] ~
retention_type[retention_status == "yes"], data =
clean_bang_longer_retention)

##
##  Kruskal-Wallis rank sum test
##
## data:  num_emails_month[retention_status == "yes"] by
retention_type[retention_status == "yes"]
## Kruskal-Wallis chi-squared = 0.15253, df = 2, p-value = 0.9266
```

```r
dunnTest(num_emails_month[retention_status == "yes"] ~
retention_type[retention_status == "yes"], data =
clean_bang_longer_retention, method = 'holm')
```

```
## Dunn (1964) Kruskal-Wallis multiple comparison
##   p-values adjusted with the Holm method.

##                      Comparison          Z    P.unadj     P.adj
## 1 retention_12m - retention_3m 0.04694502 0.9625570 0.962557
## 2 retention_12m - retention_6m 0.33862690 0.7348908 1.000000
## 3  retention_3m - retention_6m 0.33216064 0.7397680 1.000000
```

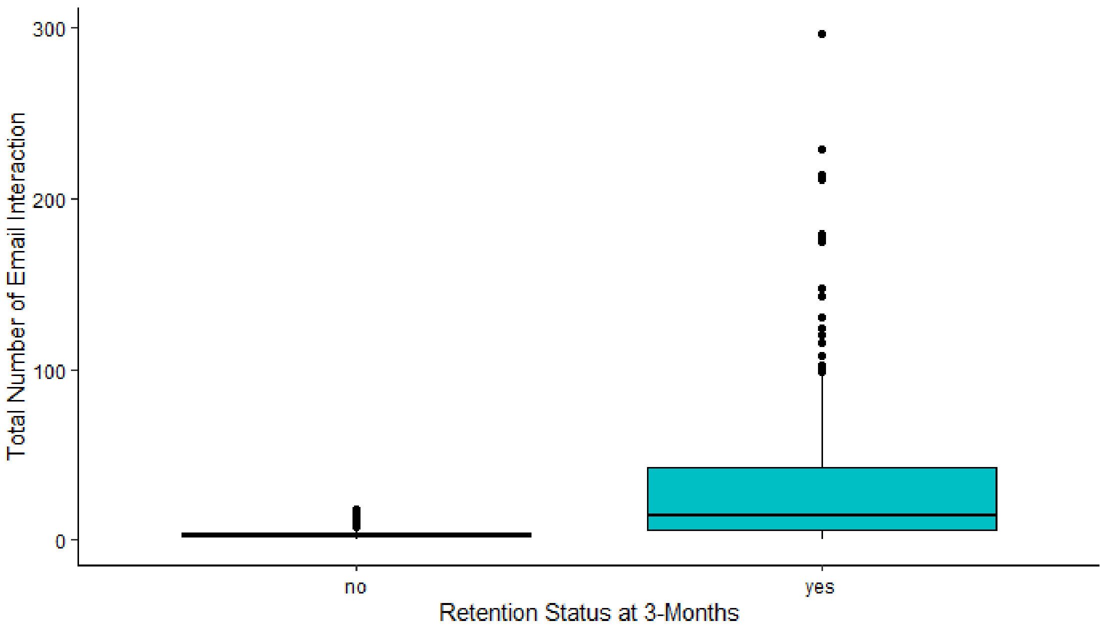


Figure81a. Continuous Retention Status at 3-Months of Bang Personal Training Members By Total Number of Non-Billing Email Interactions (W = 6059, p < 0.001)

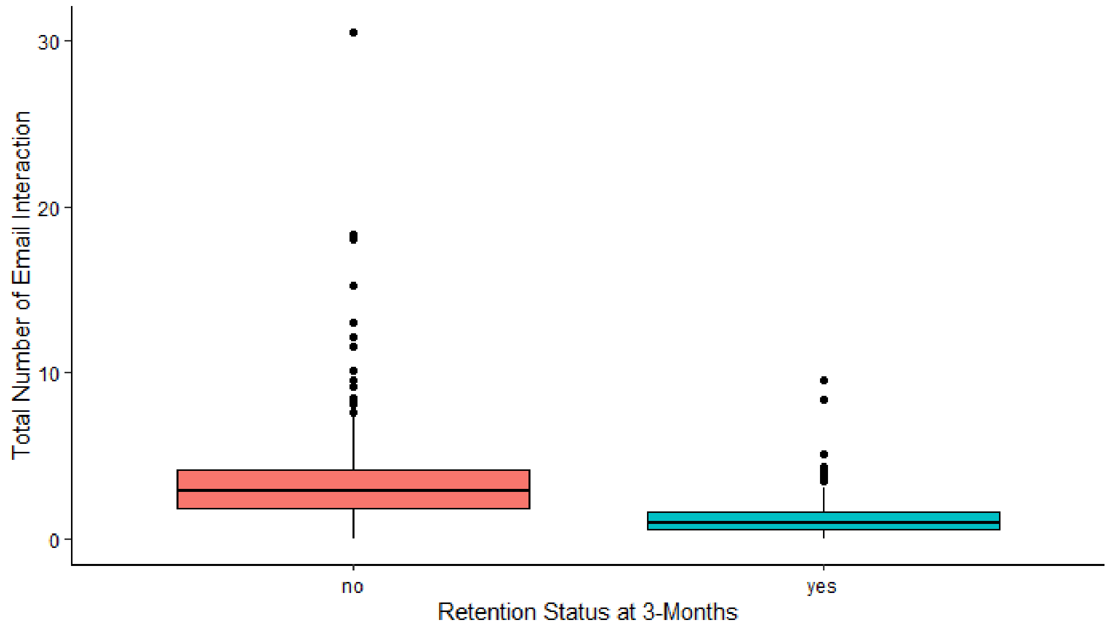


Figure81b. Continuous Retention Status at 3-Months of Bang Personal Training Members By Number of Non-Billing Email Interactions per Month (W = 40874, p < 0.001)

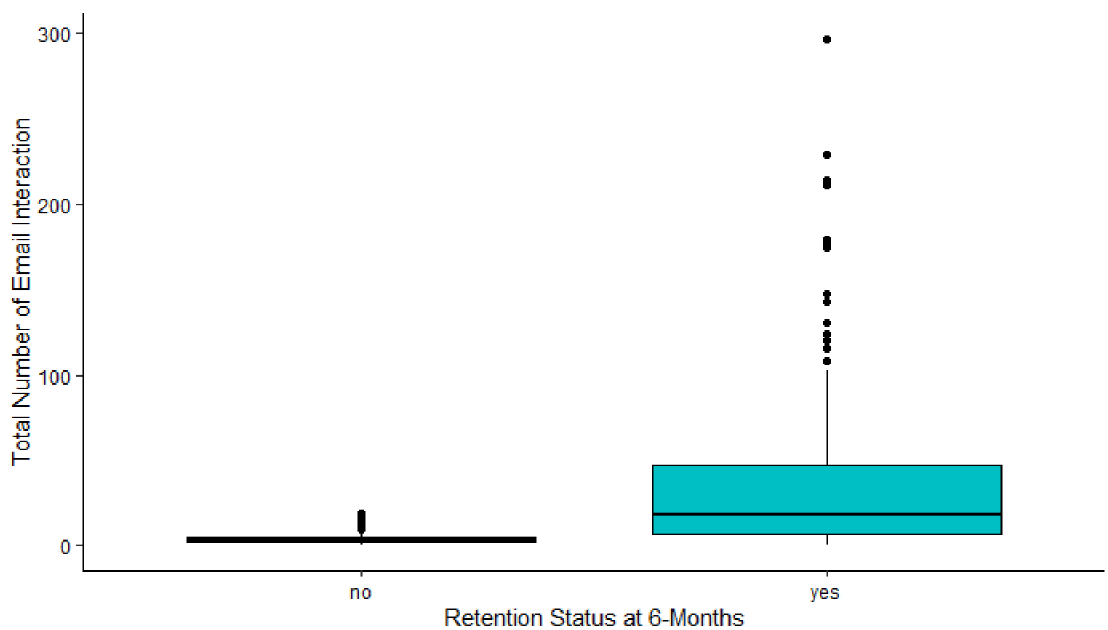


Figure82a. Continuous Retention Status at 6-Months of Bang Personal Training Members By Total Number of Non-Billing Email Interactions (W = 4900.5, p < 0.001)

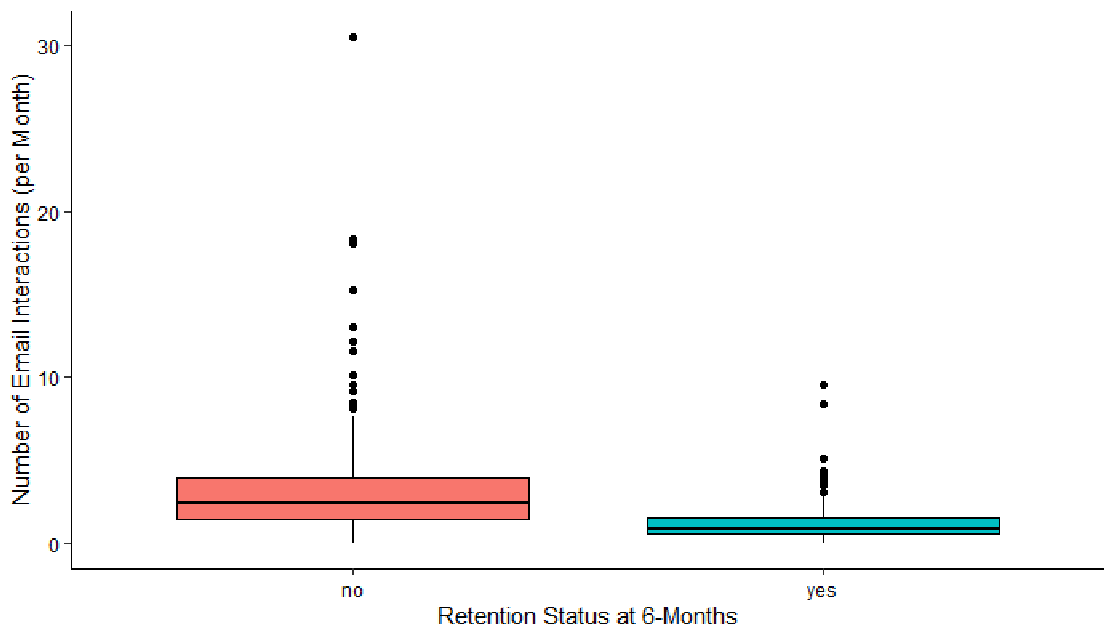


Figure82b. Continuous Retention Status at 6-Months of Bang Personal Training Members By Number of Non-Billing Email Interactions per Month ($W = 40174$, $p < 0.001$)

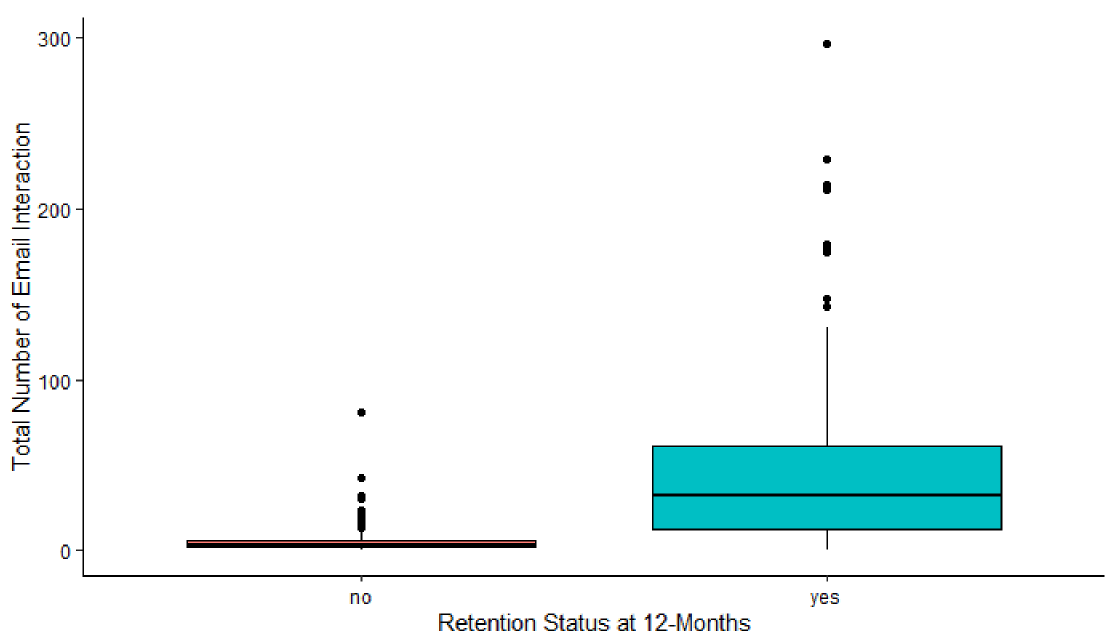


Figure83a. Continuous Retention Status at 12-Months of Bang Personal Training Members By Total Number of Non-Billing Email Interactions ($W = 3364$, $p < 0.001$)

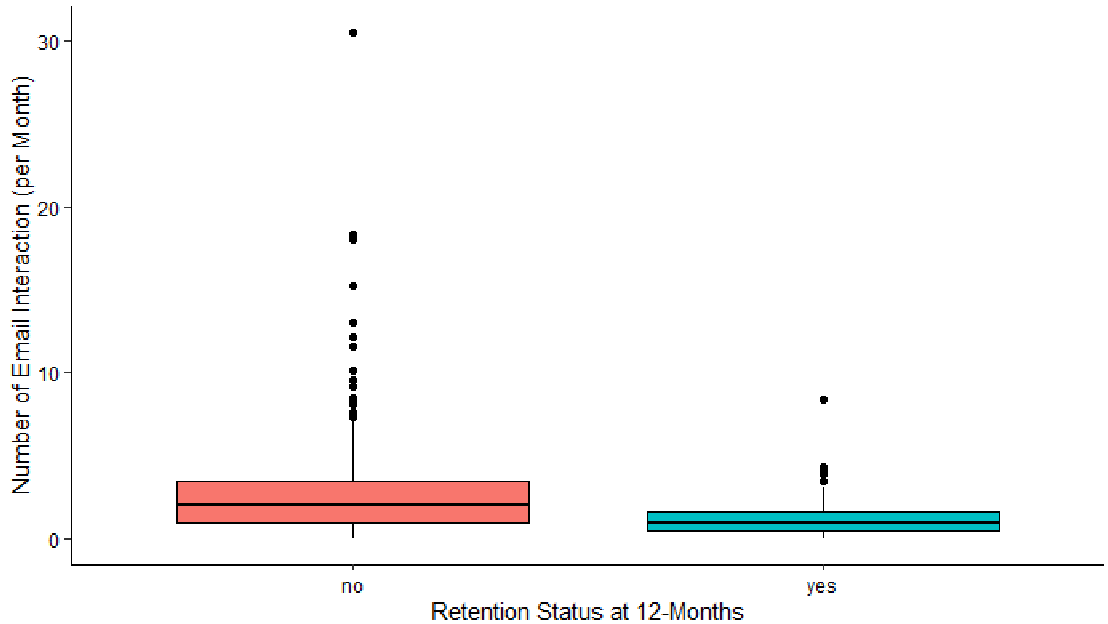


Figure83b. Continuous Retention Status at 12-Months of Bang Personal Training Members By Number of Non-Billing Email Interactions (w = 32838, p < 0.001)

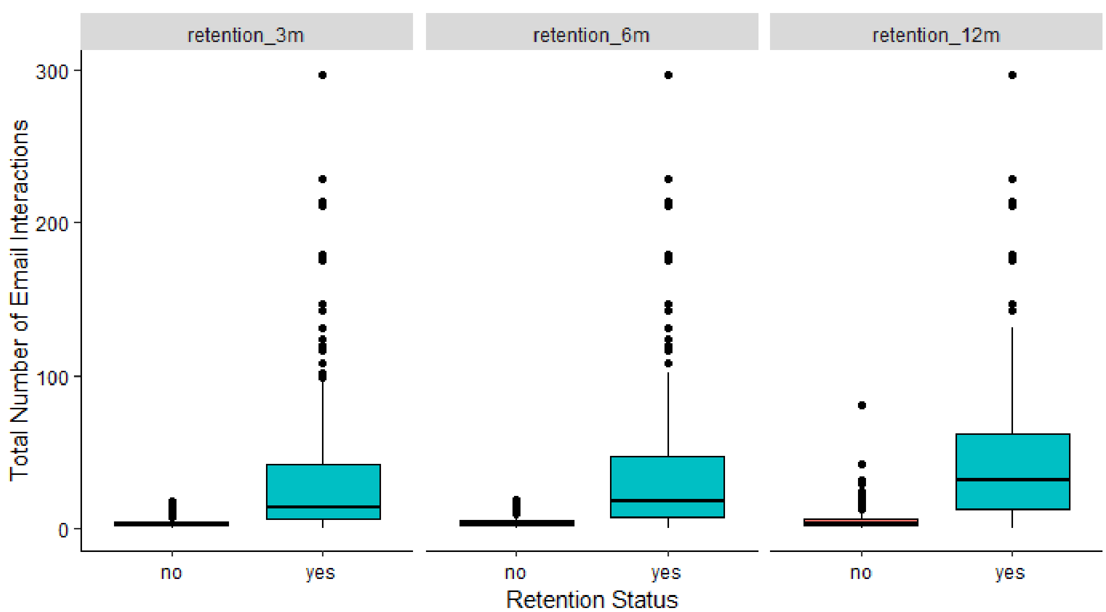


Figure84a. Number ofNon-Billing Email Interactions by Continuous Retention Status of Bang Personal Training Members Across Retention Status 3-, 6- and 12-Months ($H = 21.33$, $p < 0.001$).

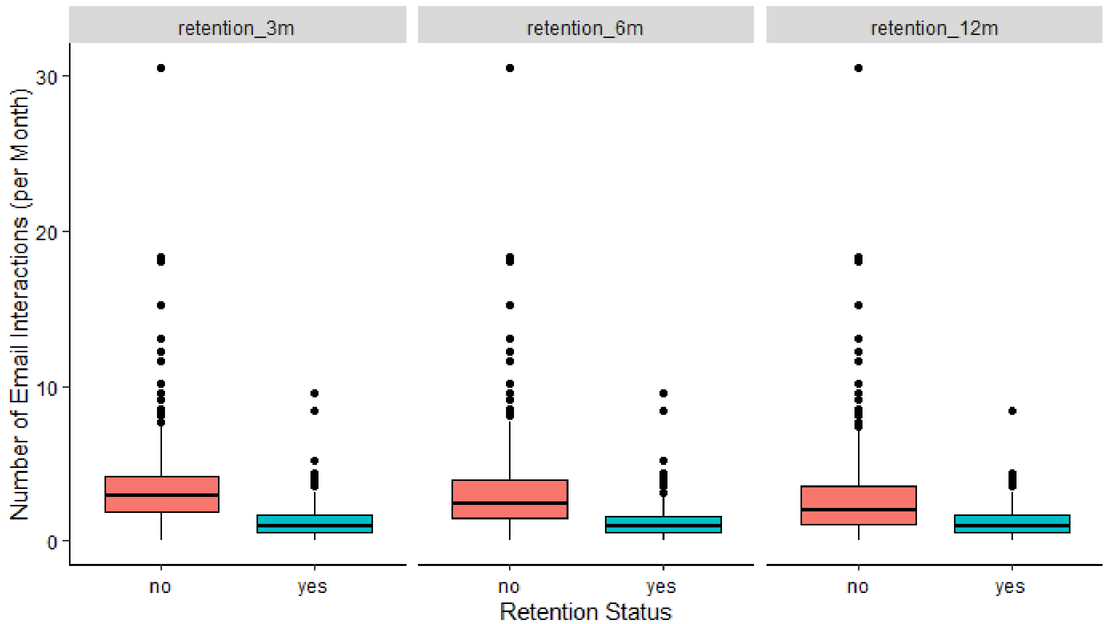


Figure84b. Number of Non-Billing Email Interactions per Month by
ious Retention Status of Bang Personal Training Members Across Retention Status 3-, 6- and 12-Months ($H = 0.152$, $p = 0.927$).

Reason to Leave

Examining past members, it was found that there were significant differences in rationale for leaving Bang Personal Training across age groups and membership types. Whilst lack of accessibility or availability in schedule was the most commonly cited reason amongst those aged 30-44, financial cost of the membership was found to join this rationale as the most commonly cited reason for those aged 45-64. However for those aged 18-29, time-based arrangement was commonly cited as the most prevalent reason to leave Bang Personal Training. In terms of membership types, those that were in the popular 3x/week membership were found to have left due to lack of availability or accessibility to use the membership. Although this was the most commonly cited reason, those that were in the 2x/week membership also cited moving away or desire to pursue other fitness interest for discontinuing membership. Interestingly enough, those with group memberships predominantly left due to the 2020 Pandemic.

Significant differences in citing reasons to discontinue membership were also noted across various attendance rates and average monthly rates. Notably, those that often cite financial cost or a time-based arrangement as a reason to discontinue membership tend to have the highest attendance rate. On the otherhand, those that had ghosted us or had cited lack of accessibility or availability tend to attend their appointments less than 50% of the time. As it relates to monthly membership rates, those citing lacking accessibility/availability tend to have higher membership rates compared to other cited reasons. Lastly, examining email interactions, it was found that there were significant differences with respect to both various types of email interactions.

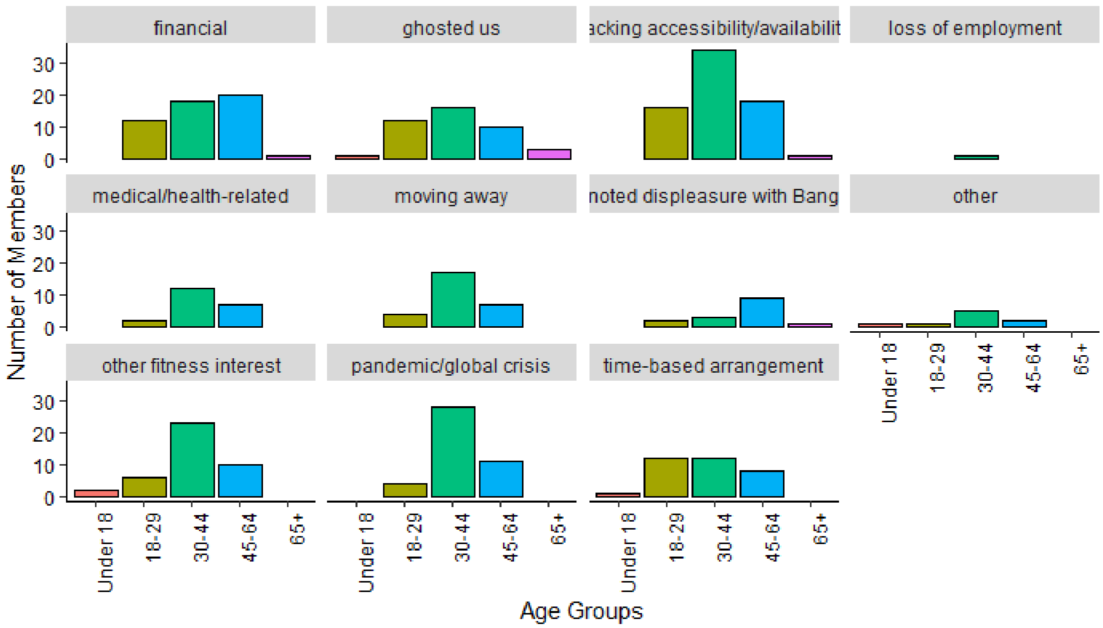


Figure85. Reasons for Discontinuing Membership by Former Bang Personal Training Members Distributed Across Ag
(χ2 = 58.25, p = 0.031)

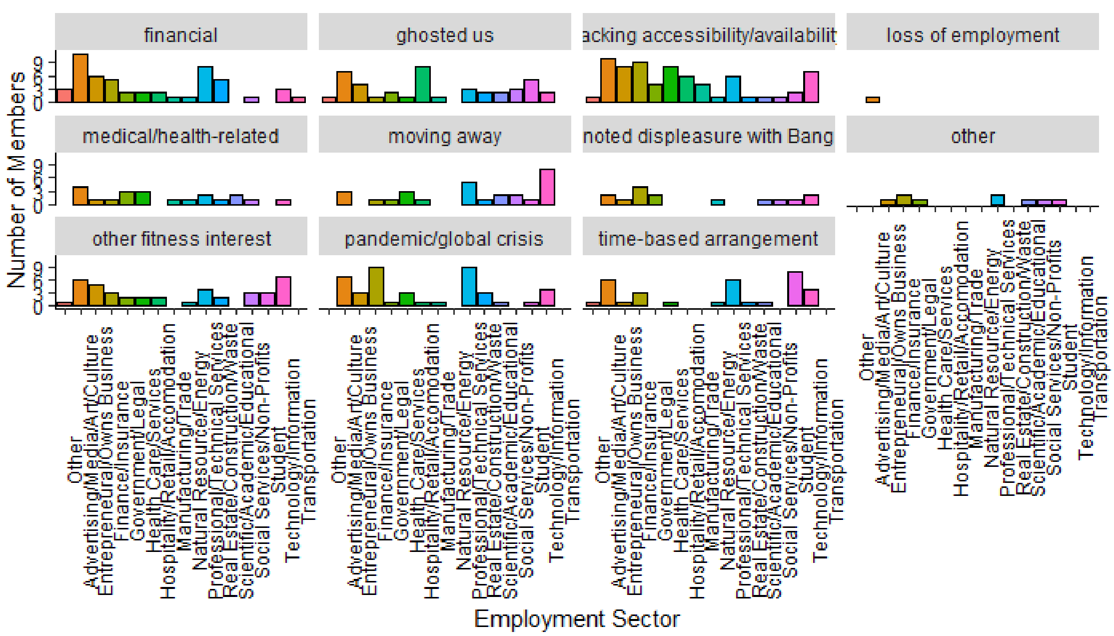


Figure86. Reasons for Discontinuing Membership by Former Bang Personal Training Members Distributed Across Em
(χ2 = 170.04, p = 0.126)

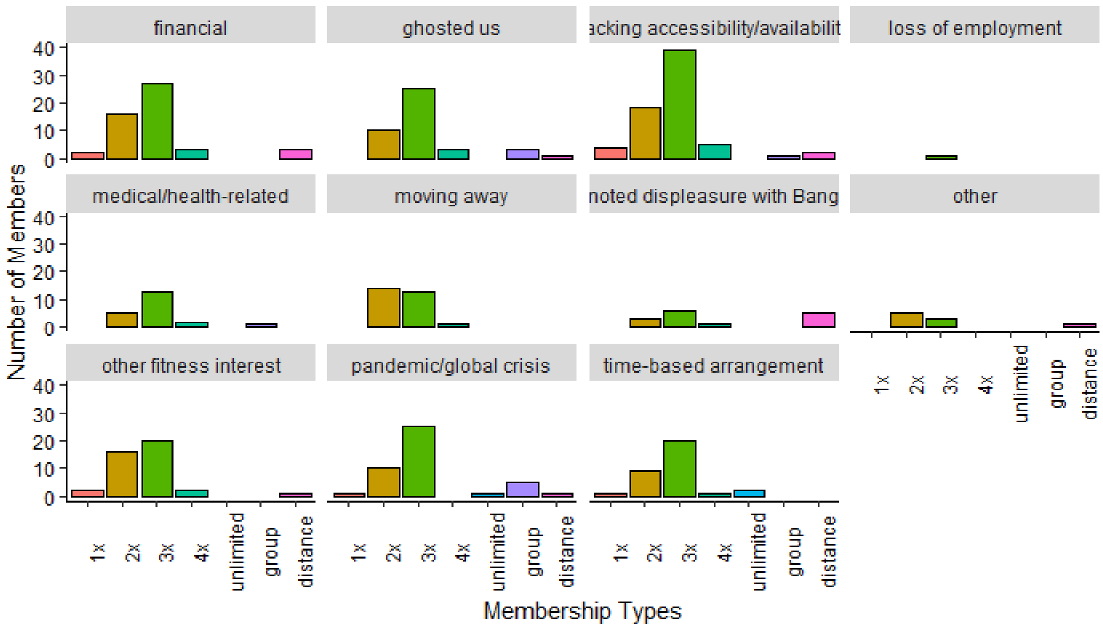


Figure87. Reasons for Discontinuing Membership by Former Bang Personal Training Members Distributed Across M
($\chi2$ = 87.41, p = 0.012)

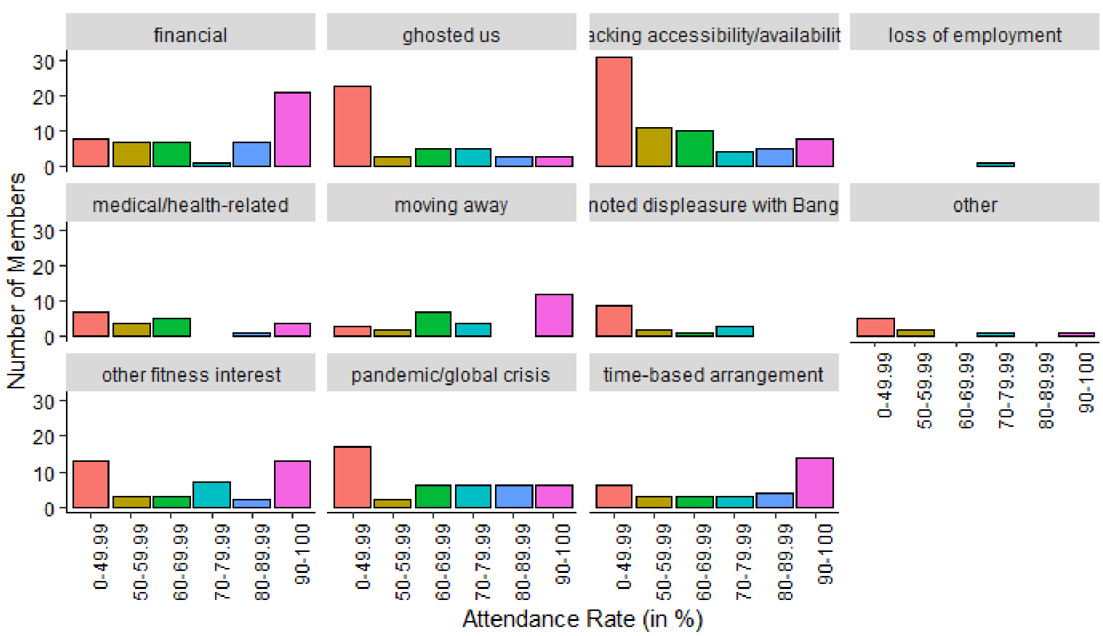


Figure88. Attendance Rate of Former Bang Personal Training Members by Reasons to Discontinue Membership
($\chi2$ = 90.12, p < 0.001)

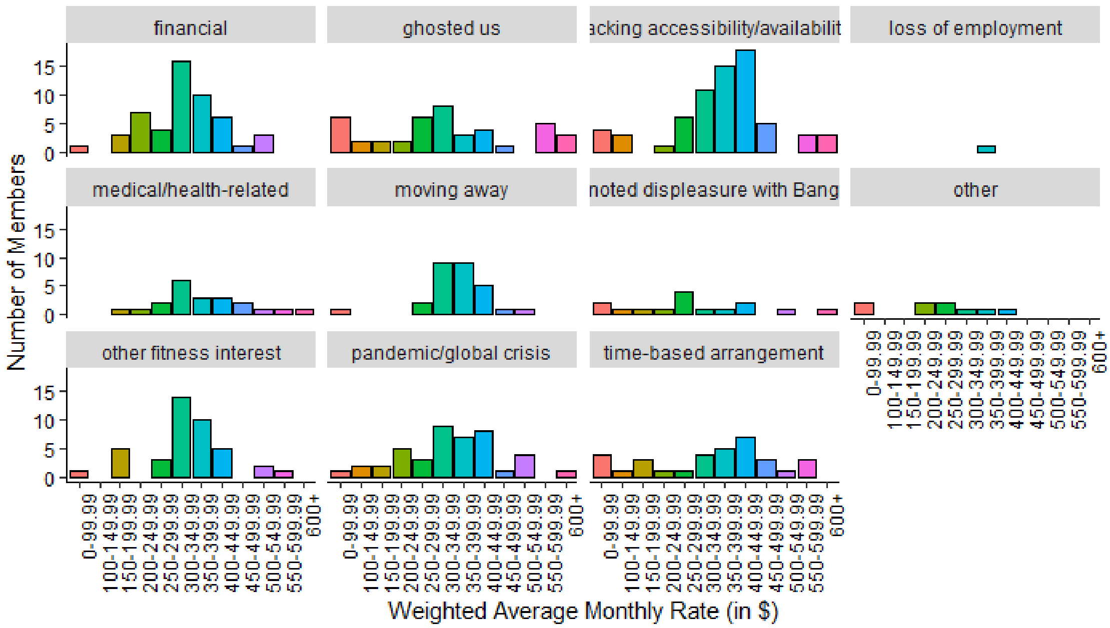


Figure89. Average Monthly Rate of Former Bang Personal Training Members by Reasons to Discontinue Membership ($\chi2$ = 140.11, p < 0.028)

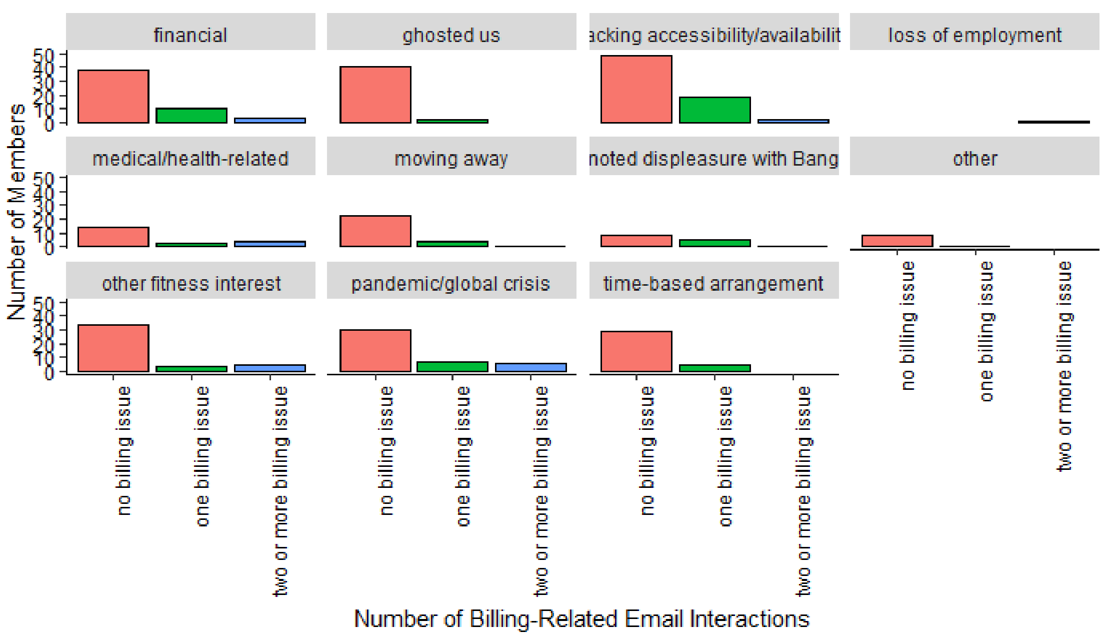


Figure90. Number of Email Interactions Relating to Billing by Former Bang Personal Training Members by Reasons to Discontinue Membership ($\chi2$ = 49.80, p < 0.001)

```
kruskal.test(new_per_ticket_cx ~ reason_to_leave, data = former_bang)

##
##  Kruskal-Wallis rank sum test
##
## data:  new_per_ticket_cx by reason_to_leave
## Kruskal-Wallis chi-squared = 19.63, df = 10, p-value = 0.03295
```

```r
dunn_test(new_per_ticket_cx ~ reason_to_leave, data = former_bang,
p.adjust.method = 'holm')
```

```
## # A tibble: 55 x 9
##    .y.      group1 group2        n1    n2 statistic       p  p.adj
p.adj.signif
##  * <chr>    <chr>  <chr>       <int> <int>     <dbl>   <dbl>  <dbl> <chr>
##  1 new_per~ finan~ ghosted us     51    42    3.36   7.90e-4 0.0434 *
##  2 new_per~ finan~ lacking ac~    51    69    1.34   1.81e-1 1      ns
##  3 new_per~ finan~ loss of em~    51     1    0.650  5.16e-1 1      ns
##  4 new_per~ finan~ medical/he~    51    21    0.833  4.05e-1 1      ns
##  5 new_per~ finan~ moving away    51    28    1.57   1.17e-1 1      ns
##  6 new_per~ finan~ noted disp~    51    15    0.390  6.96e-1 1      ns
##  7 new_per~ finan~ other          51     9    1.64   1.02e-1 1      ns
##  8 new_per~ finan~ other fitn~    51    41    1.85   6.36e-2 1      ns
##  9 new_per~ finan~ pandemic/g~    51    43   -0.0701 9.44e-1 1      ns
## 10 new_per~ finan~ time-based~    51    33   -0.0628 9.50e-1 1      ns
## # ... with 45 more rows
```

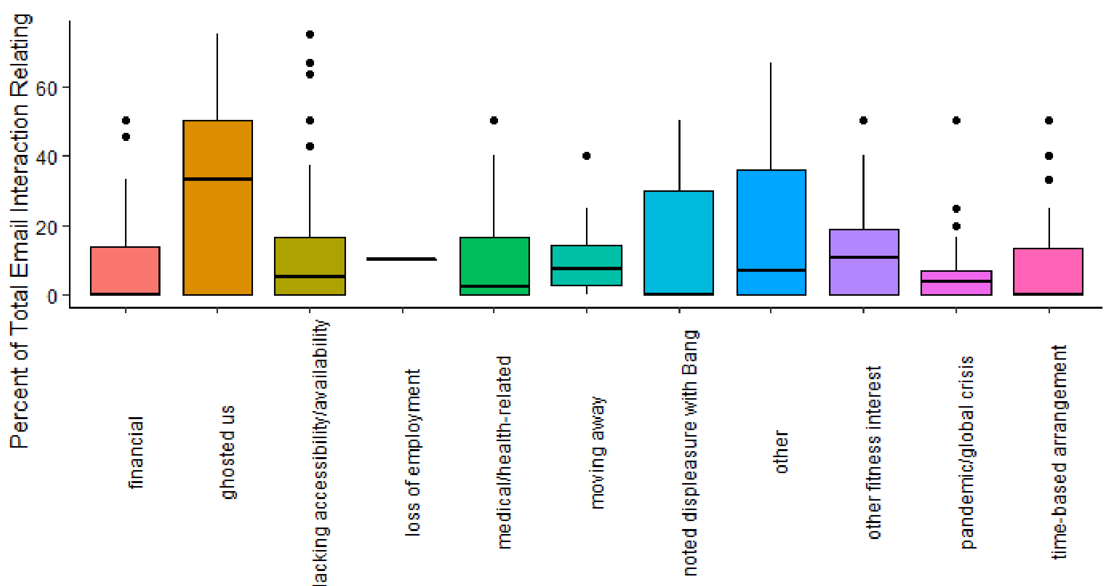


Figure91. Percentage of Email Interactions Relating to CX by Former Bang Personal Training Members by Reasons tc (W = 19.63, p = 0.032). Pairwise comparisons saw signicant lower percentage of CX-related Email Interactions be due to financial cost and those that left as a result of moving away (Z = 3.36, p = 0.043)

```r
kruskal.test(new_per_ticket_scheduling ~ reason_to_leave, data = former_bang)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  new_per_ticket_scheduling by reason_to_leave
## Kruskal-Wallis chi-squared = 32.769, df = 10, p-value = 0.0002978
```

```r
dunn_test(new_per_ticket_scheduling ~ reason_to_leave, data = former_bang,
p.adjust.method = 'holm')
```

```
## # A tibble: 55 x 9
##    .y.       group1  group2        n1    n2 statistic       p p.adj
p.adj.signif
##   * <chr>     <chr>   <chr>       <int> <int>     <dbl>   <dbl> <dbl> <chr>
##   1 new_per_~ financ~ ghosted us     51    42    -0.558  0.577   1     ns
##   2 new_per_~ financ~ lacking a~     51    69     2.05   0.0405  1     ns
##   3 new_per_~ financ~ loss of e~     51     1     0.0311 0.975   1     ns
##   4 new_per_~ financ~ medical/h~     51    21     0.807  0.420   1     ns
##   5 new_per_~ financ~ moving aw~     51    28     3.04   0.00238 0.119 ns
##   6 new_per_~ financ~ noted dis~     51    15    -1.35   0.178   1     ns
##   7 new_per_~ financ~ other          51     9     0.163  0.870   1     ns
##   8 new_per_~ financ~ other fit~     51    41     2.50   0.0124  0.545 ns
##   9 new_per_~ financ~ pandemic/~     51    43     2.99   0.00284 0.139 ns
## 10 new_per_~ financ~ time-base~     51    33     0.263  0.793   1     ns
## # ... with 45 more rows
```

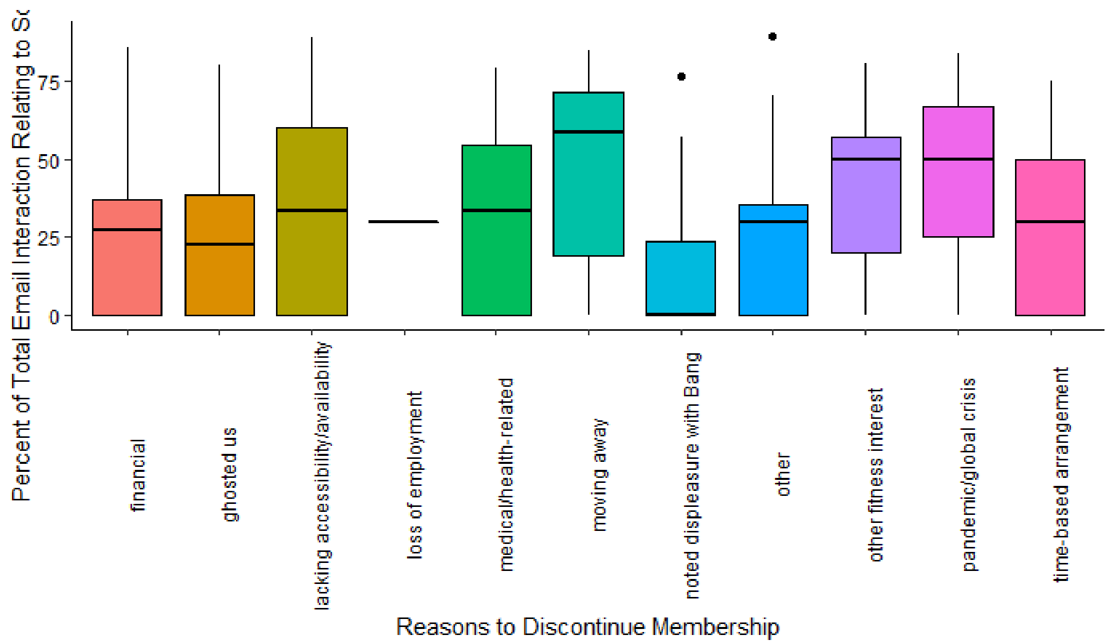


Figure92. Percentage of Email Interactions Relating to Scheduling by Former Bang Personal Training Members by Re
Discontinue Membership (W = 32.77, p < 0.001)

```
kruskal.test(new_per_ticket_service ~ reason_to_leave, data = former_bang)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  new_per_ticket_service by reason_to_leave
## Kruskal-Wallis chi-squared = 20.452, df = 10, p-value = 0.02526
```

```
dunn_test(new_per_ticket_service ~ reason_to_leave, data = former_bang,
p.adjust.method = 'holm')
```

```
## # A tibble: 55 x 9
##    .y.      group1  group2        n1    n2 statistic       p   p.adj
p.adj.signif
```

```
##   * <chr>     <chr>    <chr>        <int> <int>     <dbl>    <dbl>  <dbl> <chr>
##   1 new_per~ financ~ ghosted us      51    42  -2.15     0.0314  1      ns
##   2 new_per~ financ~ lacking a~      51    69  -2.79     0.00531 0.281  ns
##   3 new_per~ financ~ loss of e~      51     1   0.0778   0.938   1      ns
##   4 new_per~ financ~ medical/h~      51    21  -0.825    0.410   1      ns
##   5 new_per~ financ~ moving aw~      51    28  -3.18     0.00146 0.0804 ns
##   6 new_per~ financ~ noted dis~      51    15  -1.46     0.143   1      ns
##   7 new_per~ financ~ other           51     9  -1.38     0.166   1      ns
##   8 new_per~ financ~ other fit~      51    41  -2.49     0.0128  0.655  ns
##   9 new_per~ financ~ pandemic/~      51    43  -2.66     0.00771 0.401  ns
## 10 new_per~ financ~ time-base~      51    33   0.00472 0.996   1      ns
## # ... with 45 more rows
```

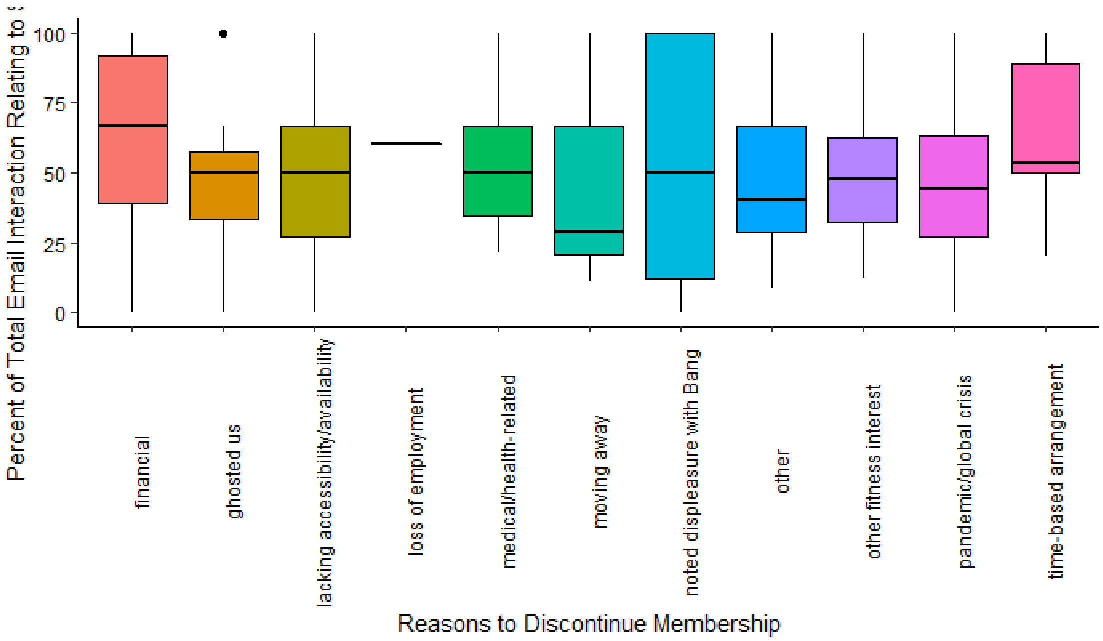


Figure93. Percentage of Email Interactions Relating to Service by Former Bang Personal Training Members by Reas Discontinue Membership (W = 20.45, p = 0.025)

```
kruskal.test(new_num_total ~ reason_to_leave, data = former_bang)

## 
##  Kruskal-Wallis rank sum test
## 
## data:  new_num_total by reason_to_leave
## Kruskal-Wallis chi-squared = 58.121, df = 10, p-value = 8.203e-09

dunn_test(new_num_total ~ reason_to_leave, data = former_bang,
p.adjust.method = 'holm')

## # A tibble: 55 x 9
##    .y.      group1  group2         n1    n2 statistic       p  p.adj
p.adj.signif
##   * <chr>    <chr>   <chr>        <int> <int>     <dbl>   <dbl>  <dbl> <chr>
##   1 new_nu~ financ~ ghosted us      51    42    -3.39   7.04e-4 0.0338 *
##   2 new_nu~ financ~ lacking ac~     51    69     0.783  4.34e-1 1      ns
```

```
##   3 new_nu~ financ~ loss of em~    51     1     0.760  4.47e-1 1      ns
##   4 new_nu~ financ~ medical/he~    51    21     0.901  3.68e-1 1      ns
##   5 new_nu~ financ~ moving away    51    28     2.88   3.97e-3 0.171  ns
##   6 new_nu~ financ~ noted disp~    51    15    -1.77   7.64e-2 1      ns
##   7 new_nu~ financ~ other          51     9    -0.0636 9.49e-1 1      ns
##   8 new_nu~ financ~ other fitn~    51    41     2.01   4.40e-2 1      ns
##   9 new_nu~ financ~ pandemic/g~    51    43     2.26   2.36e-2 0.944  ns
##  10 new_nu~ financ~ time-based~    51    33    -1.16   2.45e-1 1      ns
## # ... with 45 more rows
```

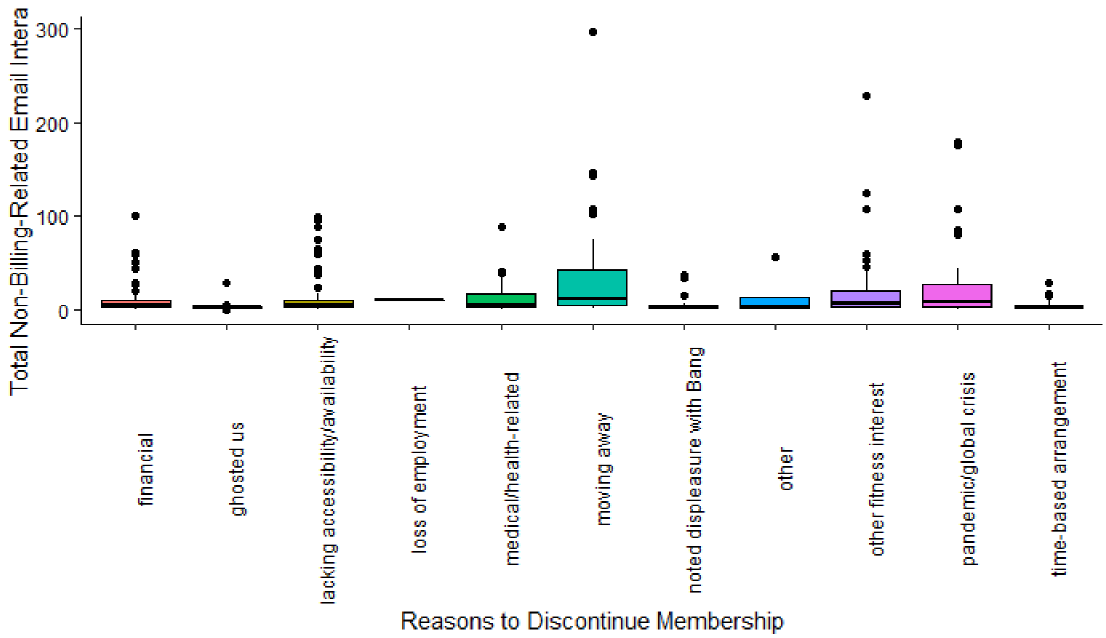


Figure94a. Total Number of Non-Billing-Related Email Interactions Relating to Service by Former Bang Personal Trai by Reasons to Discontinue Membership (W = 58.12, p < 0.001)

```
kruskal.test(num_emails_month ~ reason_to_leave, data = former_bang)

##
##  Kruskal-Wallis rank sum test
##
## data:  num_emails_month by reason_to_leave
## Kruskal-Wallis chi-squared = 31.534, df = 10, p-value = 0.0004787

dunn_test(num_emails_month ~ reason_to_leave, data = former_bang,
p.adjust.method = 'holm')

## # A tibble: 55 x 9
##    .y.       group1  group2          n1    n2 statistic      p p.adj
p.adj.signif
##  * <chr>     <chr>   <chr>        <int> <int>     <dbl>  <dbl> <dbl> <chr>
##  1 num_ema~ financ~ ghosted us       51    42     2.34   0.0195 0.898 ns
##  2 num_ema~ financ~ lacking acc~     51    69     1.63   0.102  1     ns
##  3 num_ema~ financ~ loss of emp~     51     1     0.0525 0.958  1     ns
##  4 num_ema~ financ~ medical/hea~     51    21     1.21   0.227  1     ns
##  5 num_ema~ financ~ moving away      51    28    -1.87   0.0612 1     ns
```

```
##   6 num_ema~ financ~ noted displ~     51     15    -0.0886 0.929  1      ns
##   7 num_ema~ financ~ other           51      9    -0.517  0.605  1      ns
##   8 num_ema~ financ~ other fitne~     51     41    -0.918  0.359  1      ns
##   9 num_ema~ financ~ pandemic/gl~     51     43    -1.44   0.150  1      ns
## 10 num_ema~ financ~ time-based ~     51     33     1.51   0.130  1      ns
## # ... with 45 more rows
```

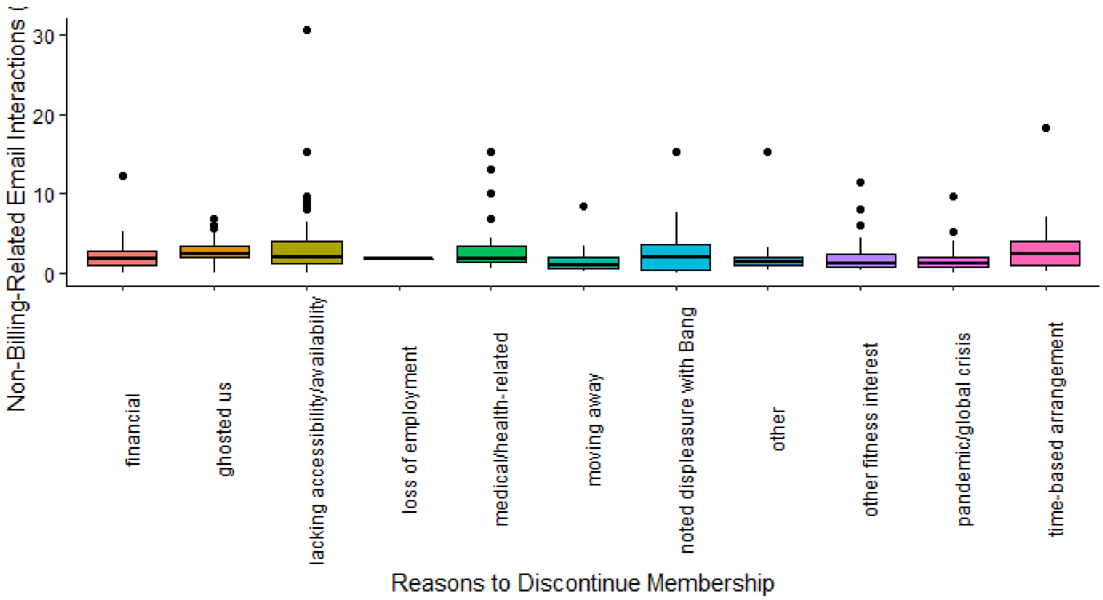


Figure95b. Number of Non-Billing-Related Email Interactions per Month Relating to Service by Former Bang Personal Reasons to Discontinue Membership (W = 31.53, p < 0.001). Following pairwise comparisons, lacked availability rep (Z = - 3.31, p = 0.049), along with those that had 'ghosted us' with more email interactions than those that moved awa and those that had left due to the Pandemic (Z = -3.61, p = 0.016)

Modelling Length of Bang Personal Training Membership

Examining the length of membership of members, it was found to range from as low as 2 days to as much as 3790 days with the median duration length being around 4.5 months. Based on the kaplan meier curves, it was suspected that age, employment sector, membership, attendance rate, average monthly rate and number of billing issues.

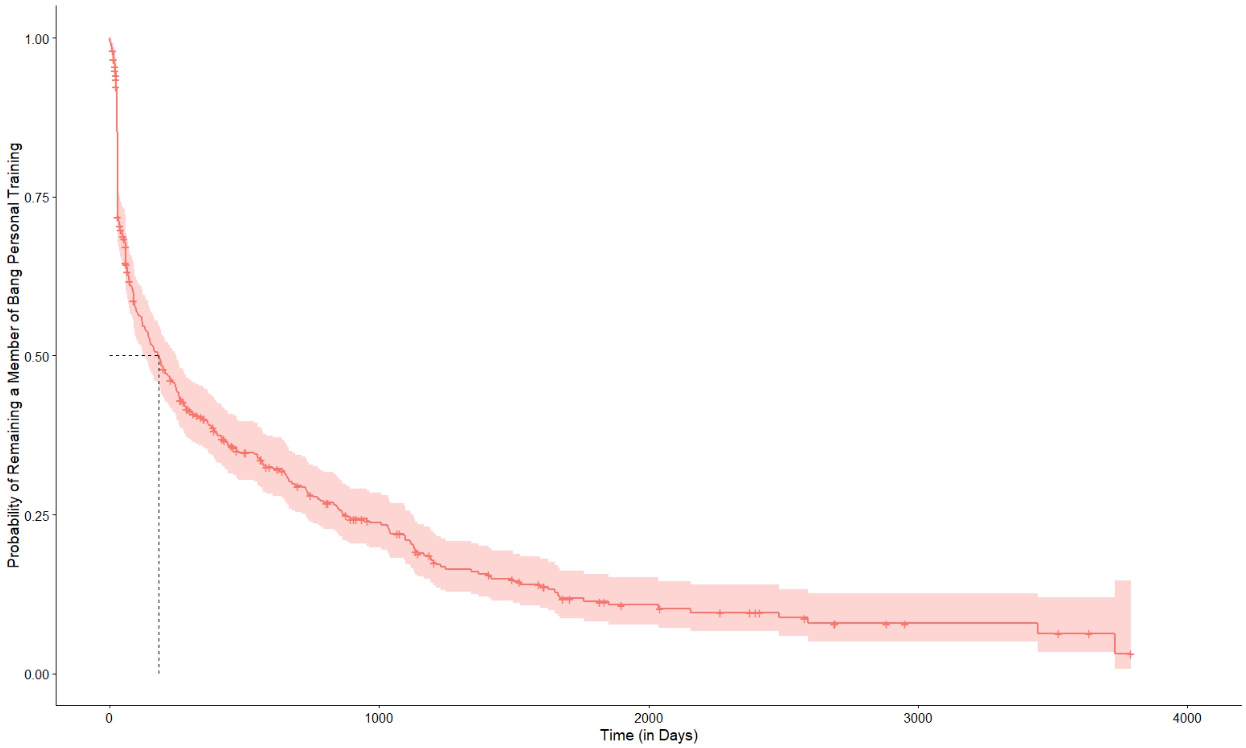


Figure96. Kaplan-Meier Estimates of Length of Membership

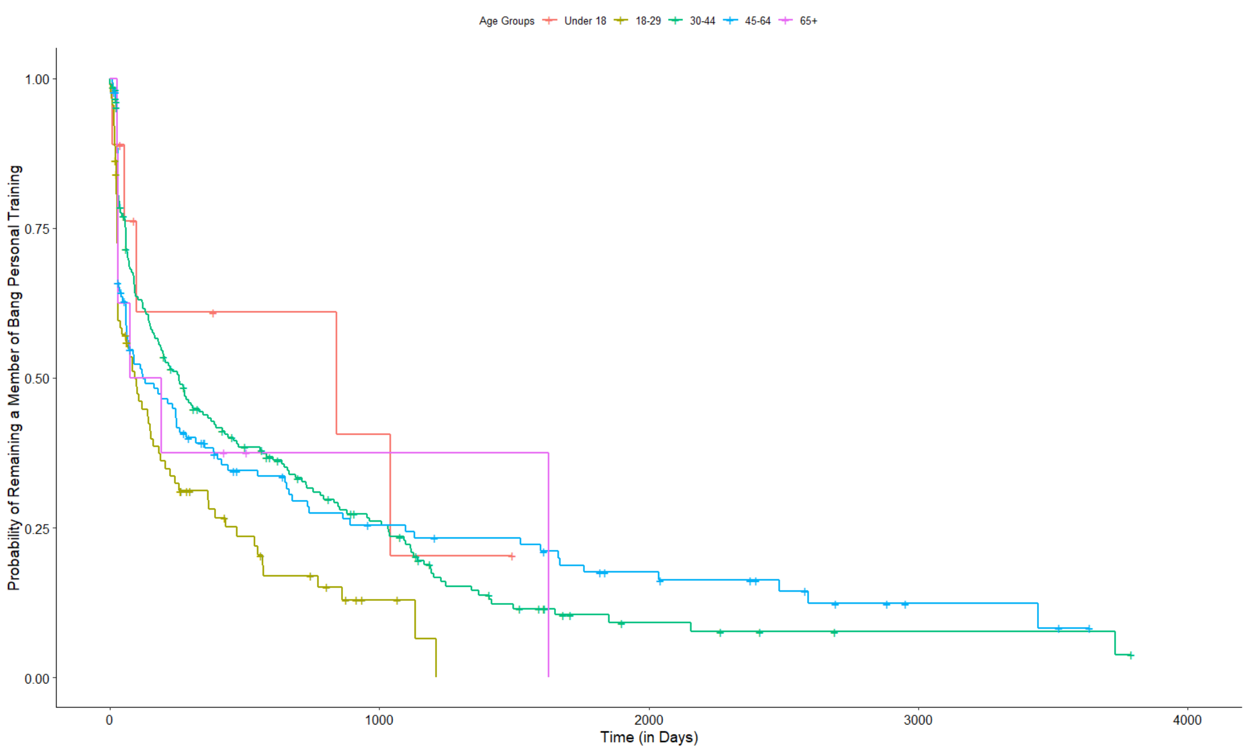


Figure97. Kaplan-Meier Estimates of Length of Membership across Age Groups

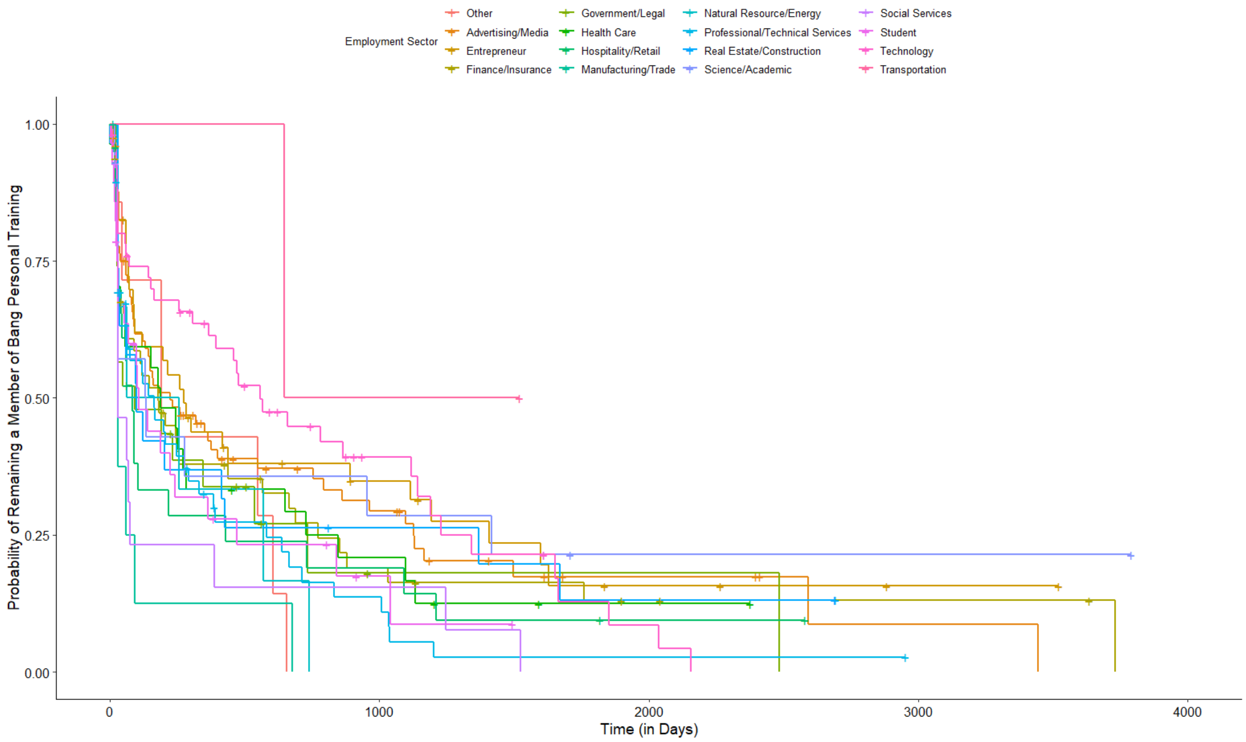


Figure98. Kaplan-Meier Estimates of Length of Membership across Employment Sectors

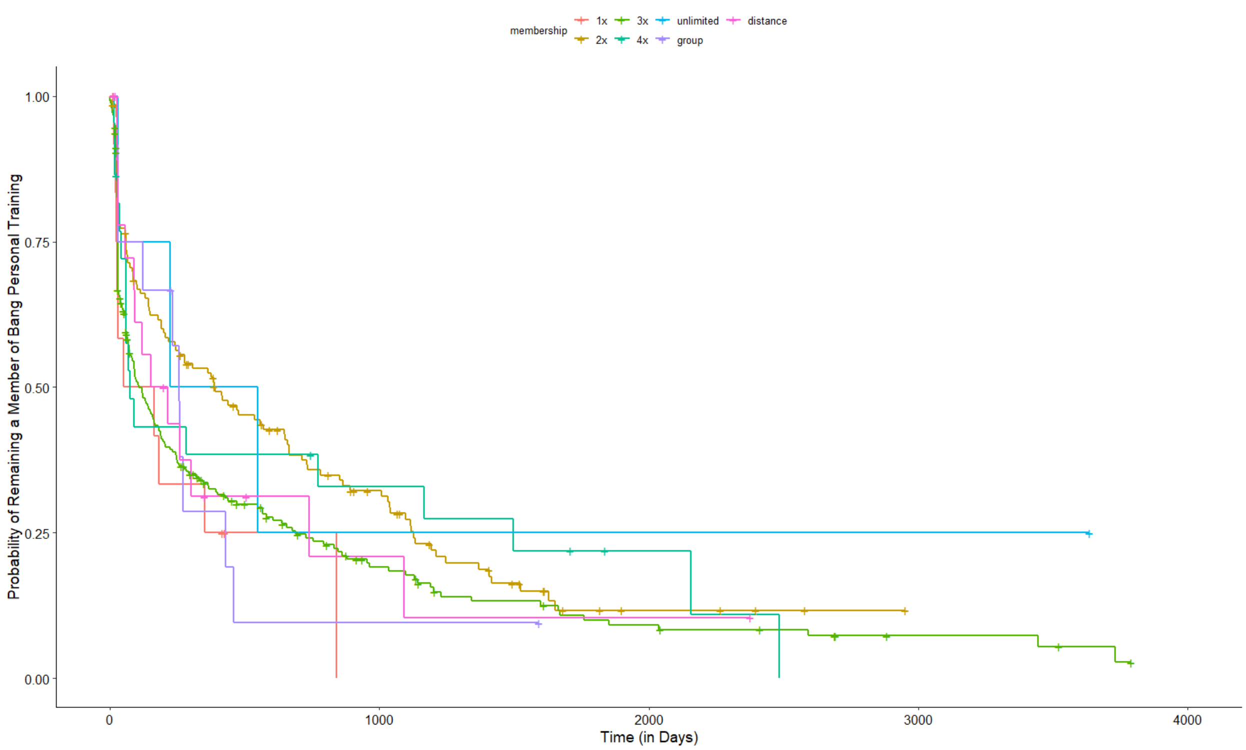


Figure99. Kaplan-Meier Estimates of Length of Membership Across Membership Types

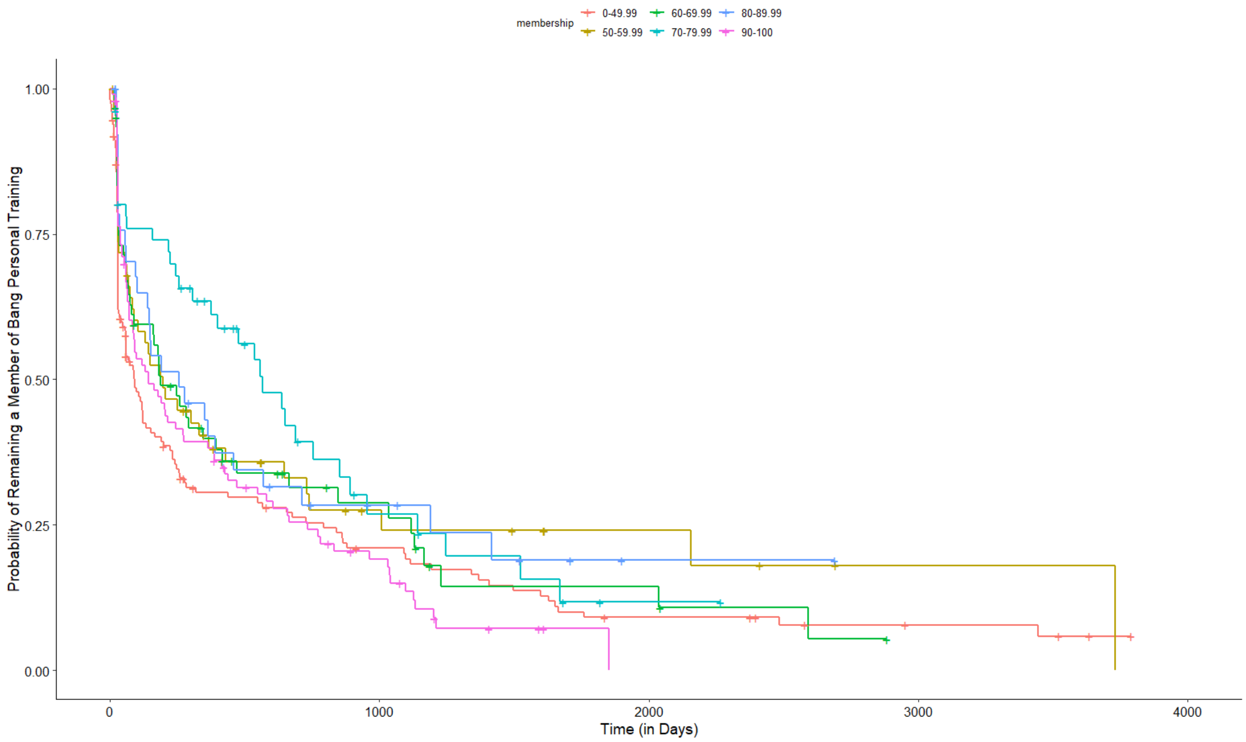


Figure100. Kaplan-Meier Estimates of Length of Membership Across Attendance Groupings

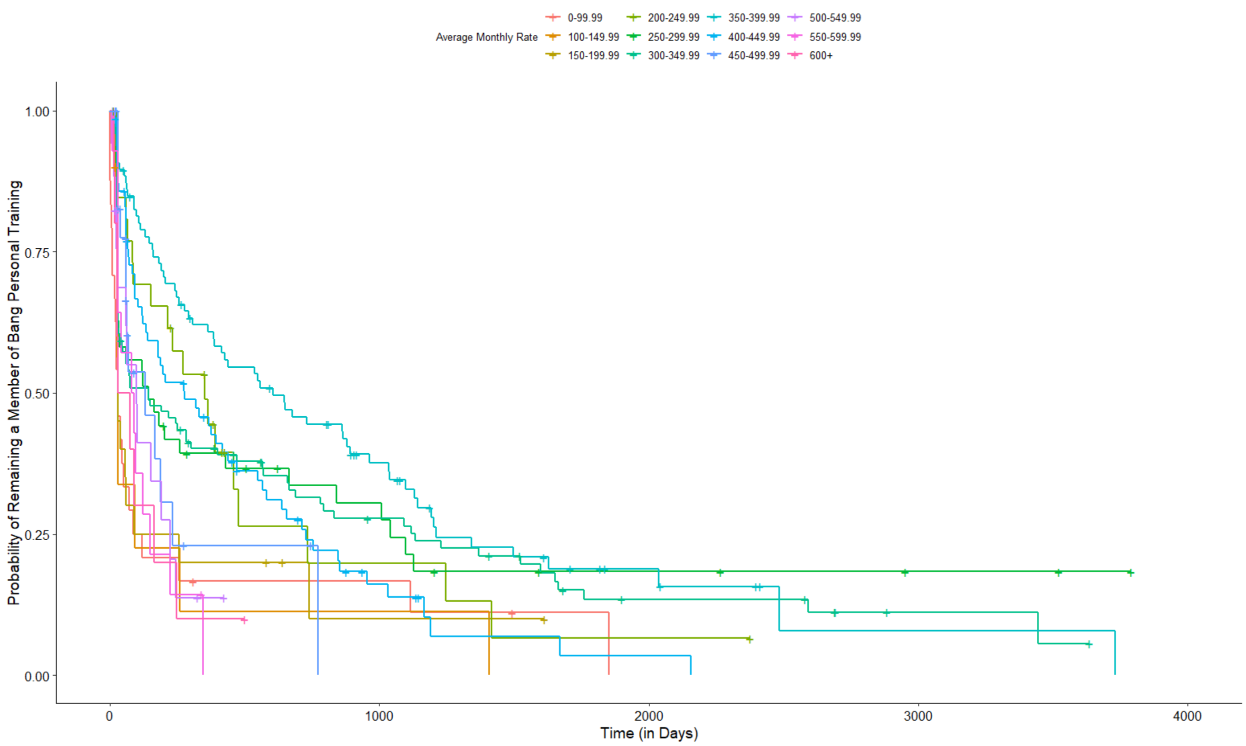


Figure101. Kaplan-Meier Estimates of Length of Membership Across Weighted Averages of Monthly Membership Rates

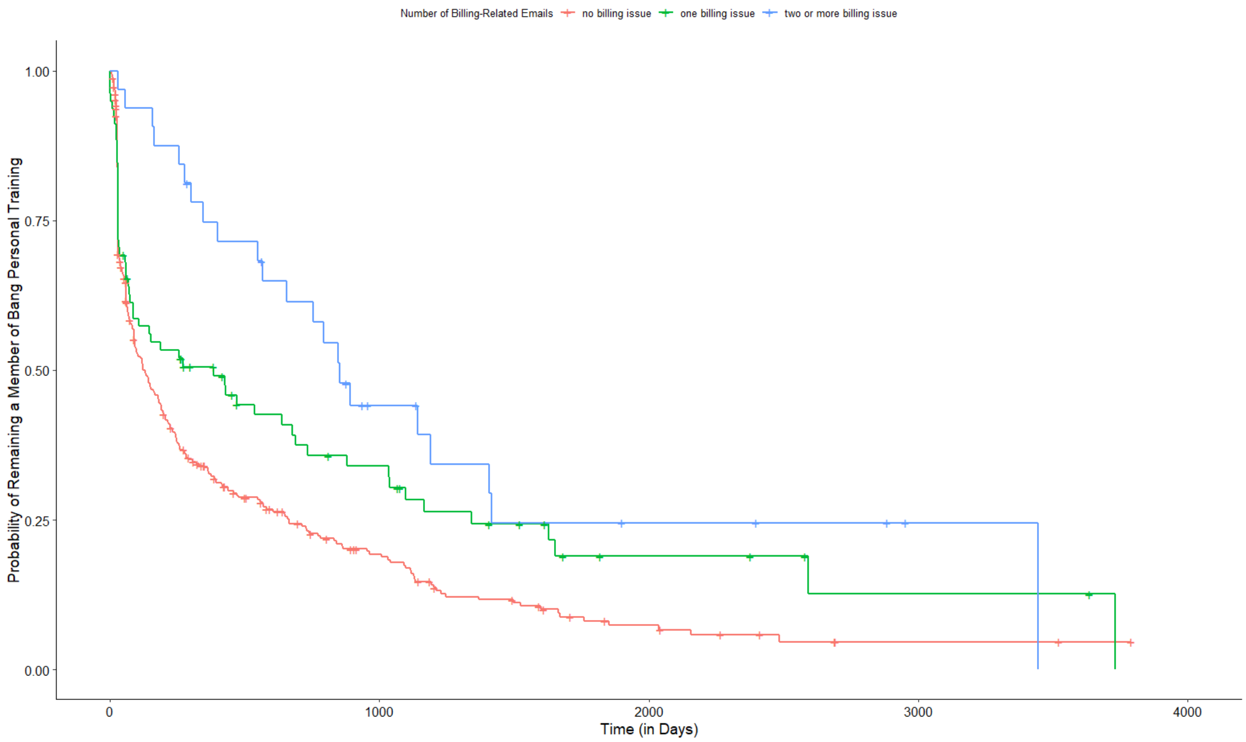


Figure102. Kaplan-Meier Estimates of Length of Membership Across Number of Billing-Related Email Interactions

Seeing as I would like to gain some insight on how certain variable in our data set would play a role in predicting length of membership, this would in effect be performing a survival analysis. As such, I would be approaching this through two means: one is through Random Survival Forest (data science~y way) and the other is through the Cox Regression Proportional Hazard model. For the second case, as there will be assumptions that needs to be satisfied in order for the model to be "valid", several tests will need to be conducted to ensure this (as noted in this YouTube Video). As such, there will likely be the use of log-transformation or categorization of certain numeric variables that will be included into this model.

NOTE: For reference: random survival forest was modelled similarly as shown here

CHURN ANALYSIS - RANDOM SURVIVAL FOREST

In developing the Random Survival Forest, I'll need the dataset to only include variables that I would like to be tested to explain outcomes from happening. Thus a separate dataset will need to be created that contains only the variables that are of interest to us to determine churn outcomes. This will include: * age_group * employment_sector * became_former_member (necessary for censoring) * length (necessary as dependent variable) * membership * monthly_rate_group * attendance_grouping_ver.1 * num_emails_month * ever_emails_month * ever_billing_issue * new_per_ticket_cx * ever_cx * new_per_ticket_scheduling * ever_scheduling * new_per_ticket_service * ever_service

Using the random survival forest specific dataset, I've split the data set 80:20 with respect to training:test. In forming the training model, which has an error rate of **16.95%**, it was found that the error rate in predicting membership length to churn with the test data was

17.59% (NOTE: this would be equivalent to the C-index via 1 - error rate, so 0.8241), so overall an OK model. Looking at the various ways to modify the parameters, it was found that the error rate more-or-less stabilized after 1000 trees as evident by the marginal differences in error rates at the higher number of trees. Similarly, findings were found with respect to mtry with the largest being between 7 and the default of 3.

Examining the importance of all of the testable predictors in impacting the outcome of predicting churn, it was found that the **number of non-billing email interactions per month** played the largest role, followed by ever having a CX-related email interaction, percent composition of scheduling-related email interaction, ever having a scheduling-related email interaction, percent composition of CX-related email interaction and percent composition of service-related email interaction. The rest plays a minimal importance. It is important to note that neither has a negative impact on membership churn. Notably, the degree of importance appears to hold regardless of the method of computing variable importance. Interestingly, looking at all of the possible combination of interactions of variables, doesn't appear to be one.

```
# STEP 1: Create a RSF-specific dataset

clean_bang_rsf = clean_bang_select %>%
                    select(
                      age_group,
                      employment_sector,
                      became_former_member,
                      length,
                      membership,
                      avg_monthly_rate,
                      attendance_grouping_ver.1,
                      ever_email_month,
                      num_emails_month,
                      ever_billing_issue,
                      ever_cx,
                      new_per_ticket_cx,
                      ever_scheduling,
                      new_per_ticket_scheduling,
                      ever_service,
                      new_per_ticket_service
                    )


# STEP 2: Partition the data set into training data & testing data

training.index.rsf = createDataPartition(clean_bang_rsf$length, p = 0.8, list
= FALSE)
clean_bang_rsf.train = clean_bang_rsf[training.index.rsf,]
clean_bang_rsf.test = clean_bang_rsf[-training.index.rsf,]

# STEP 3: Formulate the training model + modify parameters
```

```r
# modify the number of trees
train.model.base = rfsrc(Surv(length, became_former_member) ~ ., data =
clean_bang_rsf.train, ntree = 500, splitrule = "logrank", importance = TRUE)
train.model = rfsrc(Surv(length, became_former_member) ~ ., data =
clean_bang_rsf.train, ntree = 1000, splitrule = "logrank", importance = TRUE)
train.model.1 = rfsrc(Surv(length, became_former_member) ~ ., data =
clean_bang_rsf.train, ntree = 2000, splitrule = "logrank", importance = TRUE)
train.model.2 = rfsrc(Surv(length, became_former_member) ~ ., data =
clean_bang_rsf.train, ntree = 3000, splitrule = "logrank", importance = TRUE)
train.model.3 = rfsrc(Surv(length, became_former_member) ~ ., data =
clean_bang_rsf.train, ntree = 4000, splitrule = "logrank", importance = TRUE)

train.model.base  # 287 membership churn out of a possible 358 occured in
this dataset (~ 80.2%); err.rate = 16.34%

##                          Sample size: 358
##                     Number of deaths: 287
##                      Number of trees: 500
##            Forest terminal node size: 15
##        Average no. of terminal nodes: 14.95
## No. of variables tried at each split: 4
##               Total no. of variables: 14
##        Resampling used to grow trees: swor
##     Resample size used to grow trees: 226
##                             Analysis: RSF
##                               Family: surv
##                       Splitting rule: logrank *random*
##        Number of random split points: 10
##                           Error rate: 16.34%

train.model       # 287 membership churn out of a possible 358 occured in
this dataset (~ 80.2%); err.rate = 16.06%

##                          Sample size: 358
##                     Number of deaths: 287
##                      Number of trees: 1000
##            Forest terminal node size: 15
##        Average no. of terminal nodes: 14.731
## No. of variables tried at each split: 4
##               Total no. of variables: 14
##        Resampling used to grow trees: swor
##     Resample size used to grow trees: 226
##                             Analysis: RSF
##                               Family: surv
##                       Splitting rule: logrank *random*
##        Number of random split points: 10
##                           Error rate: 16.06%

train.model.1     # 287 membership churn out of a possible 358 occured in
this dataset (~ 80.2%); err.rate = 16.20%
```

```
##                          Sample size: 358
##                     Number of deaths: 287
##                      Number of trees: 2000
##            Forest terminal node size: 15
##        Average no. of terminal nodes: 14.88
## No. of variables tried at each split: 4
##               Total no. of variables: 14
##        Resampling used to grow trees: swor
##     Resample size used to grow trees: 226
##                             Analysis: RSF
##                               Family: surv
##                       Splitting rule: logrank *random*
##        Number of random split points: 10
##                           Error rate: 16.2%
```

```
train.model.2     # 287 membership churn out of a possible 358 occured in
this dataset (~ 80.2%); err.rate = 16.26%
```

```
##                          Sample size: 358
##                     Number of deaths: 287
##                      Number of trees: 3000
##            Forest terminal node size: 15
##        Average no. of terminal nodes: 14.897
## No. of variables tried at each split: 4
##               Total no. of variables: 14
##        Resampling used to grow trees: swor
##     Resample size used to grow trees: 226
##                             Analysis: RSF
##                               Family: surv
##                       Splitting rule: logrank *random*
##        Number of random split points: 10
##                           Error rate: 16.26%
```

```
train.model.3     # 287 membership churn out of a possible 358 occured in
this dataset (~ 80.2%); err.rate = 16.16%
```

```
##                          Sample size: 358
##                     Number of deaths: 287
##                      Number of trees: 4000
##            Forest terminal node size: 15
##        Average no. of terminal nodes: 14.82275
## No. of variables tried at each split: 4
##               Total no. of variables: 14
##        Resampling used to grow trees: swor
##     Resample size used to grow trees: 226
##                             Analysis: RSF
##                               Family: surv
##                       Splitting rule: logrank *random*
##        Number of random split points: 10
##                           Error rate: 16.16%
```

```r
# modify mtry
train.model.a = rfsrc(Surv(length, became_former_member) ~ ., data =
clean_bang_rsf.train, ntree = 2000, splitrule = "logrank", importance = TRUE,
mtry = 1)
train.model.b = rfsrc(Surv(length, became_former_member) ~ ., data =
clean_bang_rsf.train, ntree = 2000, splitrule = "logrank", importance = TRUE,
mtry = 2)
train.model.c = rfsrc(Surv(length, became_former_member) ~ ., data =
clean_bang_rsf.train, ntree = 2000, splitrule = "logrank", importance = TRUE,
mtry = 3)
train.model.d = rfsrc(Surv(length, became_former_member) ~ ., data =
clean_bang_rsf.train, ntree = 2000, splitrule = "logrank", importance = TRUE,
mtry = 4)
train.model.e = rfsrc(Surv(length, became_former_member) ~ ., data =
clean_bang_rsf.train, ntree = 2000, splitrule = "logrank", importance = TRUE,
mtry = 5)
train.model.f = rfsrc(Surv(length, became_former_member) ~ ., data =
clean_bang_rsf.train, ntree = 2000, splitrule = "logrank", importance = TRUE,
mtry = 6)
train.model.g = rfsrc(Surv(length, became_former_member) ~ ., data =
clean_bang_rsf.train, ntree = 2000, splitrule = "logrank", importance = TRUE,
mtry = 7)
train.model.h = rfsrc(Surv(length, became_former_member) ~ ., data =
clean_bang_rsf.train, ntree = 2000, splitrule = "logrank", importance = TRUE,
mtry = 8)
train.model.i = rfsrc(Surv(length, became_former_member) ~ ., data =
clean_bang_rsf.train, ntree = 2000, splitrule = "logrank", importance = TRUE,
mtry = 9)
train.model.j = rfsrc(Surv(length, became_former_member) ~ ., data =
clean_bang_rsf.train, ntree = 2000, splitrule = "logrank", importance = TRUE,
mtry = 10)

train.model.a    # 287 membership churn out of a possible 358 occured in this
dataset (~ 80.2%); err.rate =  19.56%

##                          Sample size: 358
##                     Number of deaths: 287
##                      Number of trees: 2000
##            Forest terminal node size: 15
##        Average no. of terminal nodes: 11.0855
## No. of variables tried at each split: 1
##               Total no. of variables: 14
##        Resampling used to grow trees: swor
##     Resample size used to grow trees: 226
##                             Analysis: RSF
##                               Family: surv
##                       Splitting rule: logrank *random*
##        Number of random split points: 10
##                           Error rate: 19.56%
```

```
train.model.b    # 287 membership churn out of a possible 358 occured in this dataset (~ 80.2%); err.rate =  17.51%

##                          Sample size: 358
##                     Number of deaths: 287
##                      Number of trees: 2000
##            Forest terminal node size: 15
##        Average no. of terminal nodes: 14.5025
## No. of variables tried at each split: 2
##               Total no. of variables: 14
##        Resampling used to grow trees: swor
##     Resample size used to grow trees: 226
##                             Analysis: RSF
##                               Family: surv
##                       Splitting rule: logrank *random*
##        Number of random split points: 10
##                           Error rate: 17.51%

train.model.c    # 287 membership churn out of a possible 358 occured in this dataset (~ 80.2%); err.rate =  16.52%

##                          Sample size: 358
##                     Number of deaths: 287
##                      Number of trees: 2000
##            Forest terminal node size: 15
##        Average no. of terminal nodes: 14.848
## No. of variables tried at each split: 3
##               Total no. of variables: 14
##        Resampling used to grow trees: swor
##     Resample size used to grow trees: 226
##                             Analysis: RSF
##                               Family: surv
##                       Splitting rule: logrank *random*
##        Number of random split points: 10
##                           Error rate: 16.52%

train.model.d    # 287 membership churn out of a possible 358 occured in this dataset (~ 80.2%); err.rate =  16.24%

##                          Sample size: 358
##                     Number of deaths: 287
##                      Number of trees: 2000
##            Forest terminal node size: 15
##        Average no. of terminal nodes: 14.7715
## No. of variables tried at each split: 4
##               Total no. of variables: 14
##        Resampling used to grow trees: swor
##     Resample size used to grow trees: 226
##                             Analysis: RSF
##                               Family: surv
##                       Splitting rule: logrank *random*
```

```
##         Number of random split points: 10
##                            Error rate: 16.24%
```

```
train.model.e    # 287 membership churn out of a possible 358 occured in this dataset (~ 80.2%); err.rate =  16.03%
```

```
##                           Sample size: 358
##                      Number of deaths: 287
##                       Number of trees: 2000
##             Forest terminal node size: 15
##         Average no. of terminal nodes: 14.893
## No. of variables tried at each split: 5
##                Total no. of variables: 14
##         Resampling used to grow trees: swor
##      Resample size used to grow trees: 226
##                              Analysis: RSF
##                                Family: surv
##                        Splitting rule: logrank *random*
##         Number of random split points: 10
##                            Error rate: 16.03%
```

```
train.model.f    # 287 membership churn out of a possible 358 occured in this dataset (~ 80.2%); err.rate =  16.08%
```

```
##                           Sample size: 358
##                      Number of deaths: 287
##                       Number of trees: 2000
##             Forest terminal node size: 15
##         Average no. of terminal nodes: 14.7905
## No. of variables tried at each split: 6
##                Total no. of variables: 14
##         Resampling used to grow trees: swor
##      Resample size used to grow trees: 226
##                              Analysis: RSF
##                                Family: surv
##                        Splitting rule: logrank *random*
##         Number of random split points: 10
##                            Error rate: 16.08%
```

```
train.model.g    # 287 membership churn out of a possible 358 occured in this dataset (~ 80.2%); err.rate =  16.11%
```

```
##                           Sample size: 358
##                      Number of deaths: 287
##                       Number of trees: 2000
##             Forest terminal node size: 15
##         Average no. of terminal nodes: 14.809
## No. of variables tried at each split: 7
##                Total no. of variables: 14
##         Resampling used to grow trees: swor
##      Resample size used to grow trees: 226
```

```
##                             Analysis: RSF
##                               Family: surv
##                       Splitting rule: logrank *random*
##        Number of random split points: 10
##                           Error rate: 16.11%
```

```
train.model.h    # 287 membership churn out of a possible 358 occured in this dataset (~ 80.2%); err.rate =  15.95%
```

```
##                          Sample size: 358
##                     Number of deaths: 287
##                      Number of trees: 2000
##            Forest terminal node size: 15
##        Average no. of terminal nodes: 14.8365
## No. of variables tried at each split: 8
##               Total no. of variables: 14
##        Resampling used to grow trees: swor
##     Resample size used to grow trees: 226
##                             Analysis: RSF
##                               Family: surv
##                       Splitting rule: logrank *random*
##        Number of random split points: 10
##                           Error rate: 15.95%
```

```
train.model.i    # 287 membership churn out of a possible 358 occured in this dataset (~ 80.2%); err.rate =  16.06%
```

```
##                          Sample size: 358
##                     Number of deaths: 287
##                      Number of trees: 2000
##            Forest terminal node size: 15
##        Average no. of terminal nodes: 14.996
## No. of variables tried at each split: 9
##               Total no. of variables: 14
##        Resampling used to grow trees: swor
##     Resample size used to grow trees: 226
##                             Analysis: RSF
##                               Family: surv
##                       Splitting rule: logrank *random*
##        Number of random split points: 10
##                           Error rate: 16.06%
```

```
train.model.j    # 287 membership churn out of a possible 358 occured in this dataset (~ 80.2%); err.rate =  16.08%
```

```
##                          Sample size: 358
##                     Number of deaths: 287
##                      Number of trees: 2000
##            Forest terminal node size: 15
##        Average no. of terminal nodes: 14.9255
## No. of variables tried at each split: 10
```

```
##                Total no. of variables: 14
##         Resampling used to grow trees: swor
##      Resample size used to grow trees: 226
##                              Analysis: RSF
##                                Family: surv
##                        Splitting rule: logrank *random*
##         Number of random split points: 10
##                            Error rate: 16.08%
```

```
train.model.proposed = rfsrc(Surv(length, became_former_member) ~ ., data =
clean_bang_rsf.train, ntree = 2000, splitrule = "logrank", importance = TRUE,
mtry = 7)
train.model.proposed     # 287 membership churn out of a possible 358 occured
in this dataset (~ 80.2%); err.rate = 16.01%
```

```
##                           Sample size: 358
##                      Number of deaths: 287
##                       Number of trees: 2000
##             Forest terminal node size: 15
##         Average no. of terminal nodes: 14.898
## No. of variables tried at each split: 7
##                Total no. of variables: 14
##         Resampling used to grow trees: swor
##      Resample size used to grow trees: 226
##                              Analysis: RSF
##                                Family: surv
##                        Splitting rule: logrank *random*
##         Number of random split points: 10
##                            Error rate: 16.01%
```

```
# STEP 4: Determining the important variables within the forest model

vimp(train.model, importance = "permute")$importance
```

```
##                  age_group         employment_sector
membership
##               1.180646e-04              6.693940e-04
8.724652e-05
##           avg_monthly_rate attendance_grouping_ver.1
ever_email_month
##               4.596463e-03              7.516281e-04
1.079695e-02
##           num_emails_month        ever_billing_issue
ever_cx
##               9.865368e-02              7.988894e-04
3.200971e-02
##          new_per_ticket_cx           ever_scheduling
new_per_ticket_scheduling
##               1.963392e-02              1.723374e-02
2.487068e-02
```

```
##              ever_service    new_per_ticket_service
##              3.245604e-06              1.372441e-02
```

```r
vimp(train.model, importance = "random")$importance
```

```
##                 age_group         employment_sector
membership
##              0.0023852900              0.0016243457
0.0021772994
##           avg_monthly_rate attendance_grouping_ver.1
ever_email_month
##              0.0141494624              0.0017012156
0.0180112071
##           num_emails_month        ever_billing_issue
ever_cx
##              0.1121255872              0.0014601847
0.0339431283
##          new_per_ticket_cx           ever_scheduling
new_per_ticket_scheduling
##              0.0251023681              0.0246676294
0.0313281251
##              ever_service    new_per_ticket_service
##              0.0006796067              0.0178245294
```

```r
vimp(train.model.proposed, importance = 'permute')$importance
```

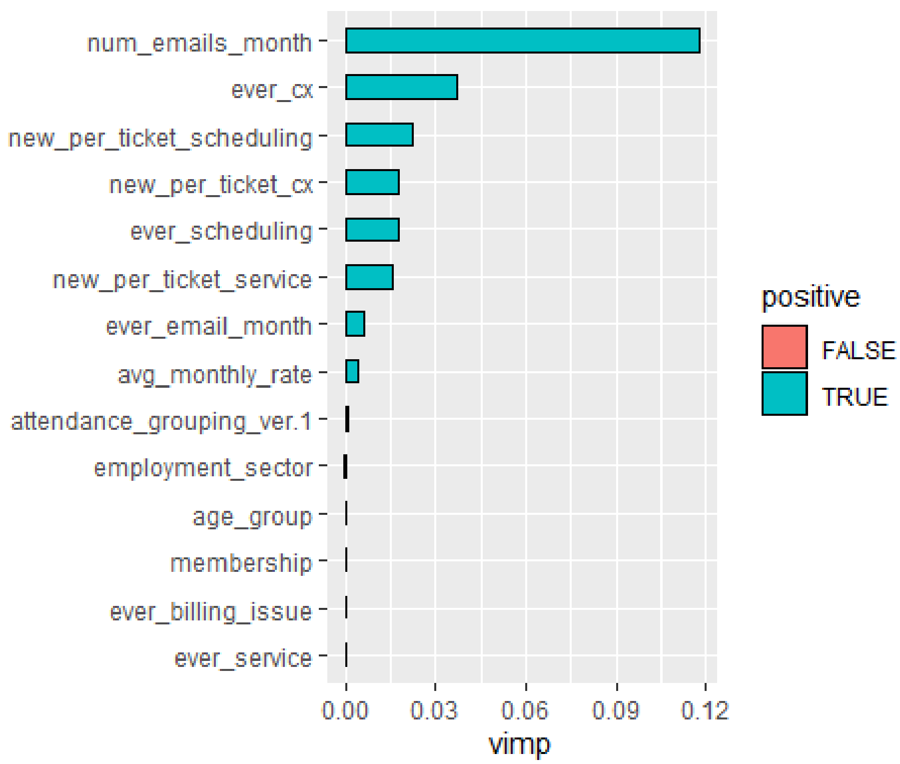
```
##                 age_group         employment_sector
membership
##              1.024009e-04              5.491658e-04
1.851202e-04
##           avg_monthly_rate attendance_grouping_ver.1
ever_email_month
##              4.467203e-03              6.638167e-04
5.972264e-03
##           num_emails_month        ever_billing_issue
ever_cx
##              1.187566e-01              2.148423e-04
3.924261e-02
##          new_per_ticket_cx           ever_scheduling
new_per_ticket_scheduling
##              1.835277e-02              1.789232e-02
2.224094e-02
##              ever_service    new_per_ticket_service
##              3.133369e-06              1.565345e-02
```

```r
vimp(train.model.proposed, importance = 'random')$importance
```

```
##                 age_group         employment_sector
membership
##              0.0022821057              0.0008977696
0.0020200669
```

```
##          avg_monthly_rate attendance_grouping_ver.1
ever_email_month
##              0.0124594751              0.0009813869
0.0106837805
##           num_emails_month         ever_billing_issue
ever_cx
##              0.1347783847              0.0005491734
0.0380180142
##          new_per_ticket_cx           ever_scheduling
new_per_ticket_scheduling
##              0.0221306152              0.0246362171
0.0268971429
##              ever_service     new_per_ticket_service
##              0.0001299225              0.0189836193
```

```
plot(gg_vimp(train.model.proposed)) # Top Predictors = num_emails_month,
ever_cx, new_per_ticket_scheduling, ever_scheduling, new_per_ticket_cx,
new_per_ticket_service
```

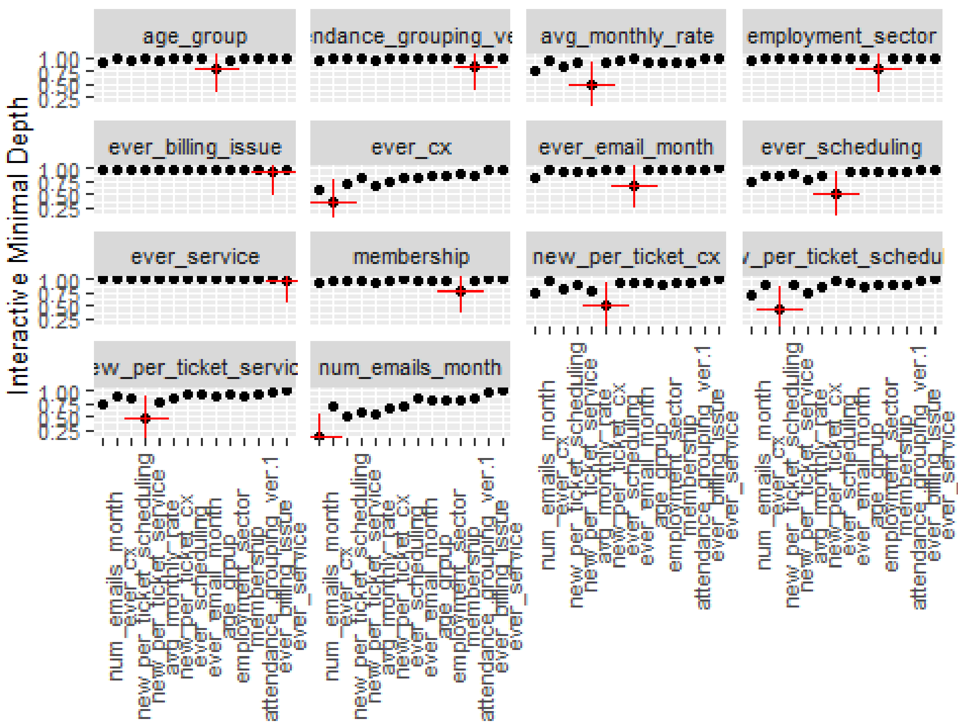



```
var.select(train.model.proposed, method = 'md') # Top predictors:
num_emails_month > new_per_ticket_scheduling > new_per_ticket_service >
ever_cx > new_per_ticket_cx

## minimal depth variable selection ...
##
##
## -----------------------------------------------------------
```

```
## family              : surv
## var. selection      : Minimal Depth
## conservativeness    : medium
## x-weighting used?   : TRUE
## dimension           : 14
## sample size         : 358
## ntree               : 2000
## nsplit              : 10
## mtry                : 7
## nodesize            : 15
## refitted forest     : FALSE
## model size          : 6
## depth threshold     : 3.5947
## PE (true OOB)       : 16.0104
##
##
## Top variables:
##                            depth  vimp
## num_emails_month           0.890 0.118
## ever_cx                    2.499 0.037
## new_per_ticket_scheduling 3.102 0.022
## new_per_ticket_service     3.276 0.016
## avg_monthly_rate           3.298 0.004
## new_per_ticket_cx          3.537 0.018
## ------------------------------------------------------------

max.model.3 <- max.subtree(train.model.proposed)
max.model.3$topvars # Top predictors: num_emails_month, ever_cx,
new_per_ticket_scheduling, avg_monthly_rate, new_per_ticket_service,
new_per_ticket_cx

## [1] "avg_monthly_rate"          "num_emails_month"
## [3] "ever_cx"                   "new_per_ticket_cx"
## [5] "new_per_ticket_scheduling" "new_per_ticket_service"

train.model.proposed.ver1 = rfsrc(Surv(length, became_former_member) ~
num_emails_month +
                                    new_per_ticket_scheduling +
                                    new_per_ticket_service +
                                    ever_cx +
                                    new_per_ticket_cx,
                                    data = clean_bang_rsf.train, ntree =
2000, mtry = 7, splitrule = 'logrank', importance = TRUE)

train.model.proposed.ver1 # 287 membership churn out of a possible 358
occured in this dataset (~ 80.2%); err.rate = 17.12%

##                           Sample size: 358
##                      Number of deaths: 287
##                       Number of trees: 2000
##             Forest terminal node size: 15
```

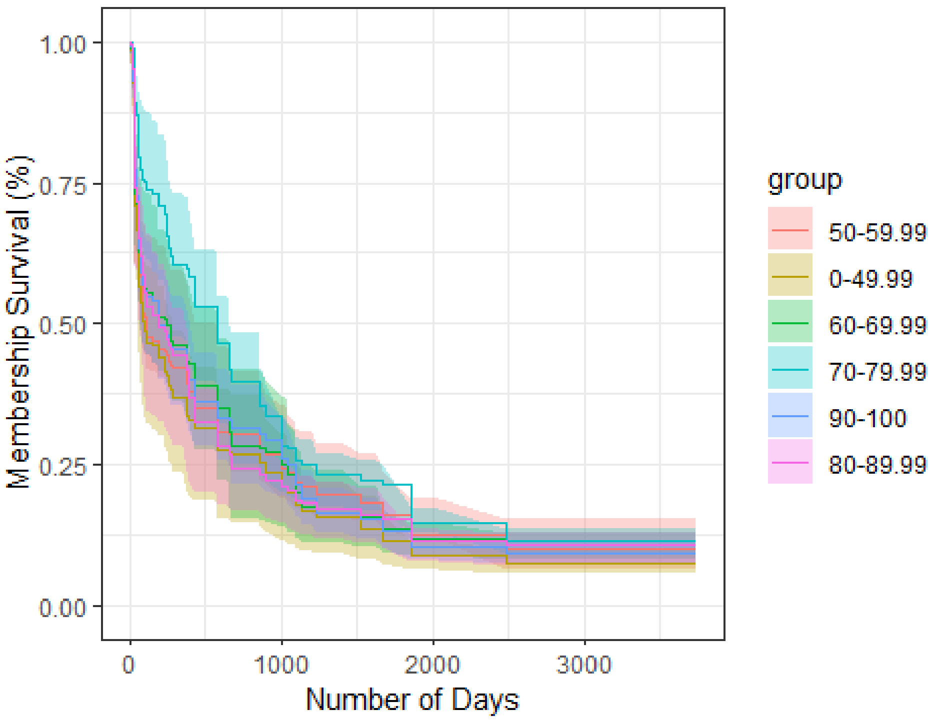
```
##         Average no. of terminal nodes: 14.676
## No. of variables tried at each split: 5
##               Total no. of variables: 5
##         Resampling used to grow trees: swor
##      Resample size used to grow trees: 226
##                              Analysis: RSF
##                                Family: surv
##                        Splitting rule: logrank *random*
##         Number of random split points: 10
##                            Error rate: 17.12%
```

```
# STEP 5: Explore any potential interaction effects that may exist within the model

plot(gg_interaction(train.model.proposed))
```

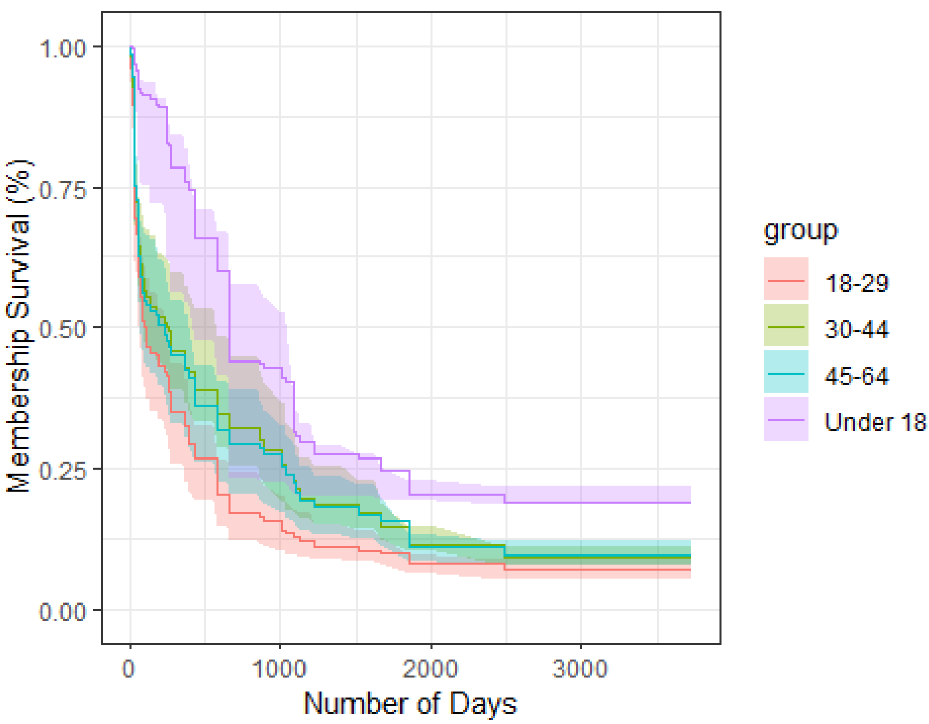



```
# Higher values indicate lower interactivity with target variable marked in red. Overall, there doesn't seem to be an interactive effect found


# STEP 6: Evaluate the performance of the training model with test data set and compare

pred_churn = predict(train.model.proposed, clean_bang_rsf.test, outcome = 'test')
pred_churn # Out of 89 individuals, 66 were found to have reported to have
```

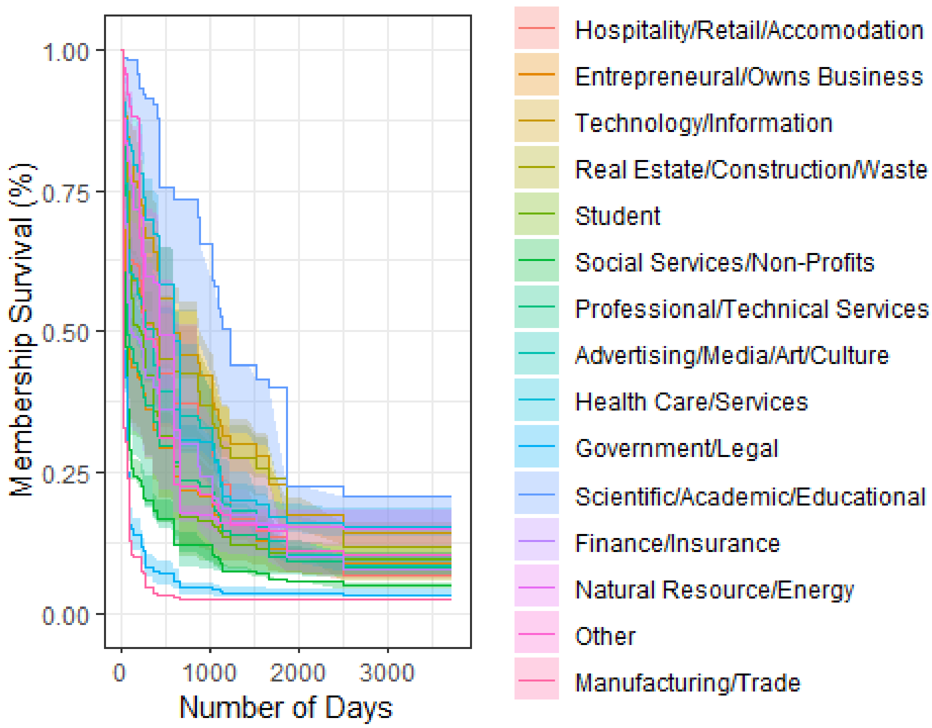
```
churn.  However in terms of the model predicting outcomes, there was an error rate of 16.76%

##   Sample size of test (predict) data: 89
##        Number of deaths in test data: 66
##                 Number of grow trees: 2000
##   Average no. of grow terminal nodes: 14.898
##          Total no. of grow variables: 14
##        Resampling used to grow trees: swor
##     Resample size used to grow trees: 56
##                             Analysis: RSF
##                               Family: surv
##                  Test set error rate: 16.76%

pred_churn.a = predict(train.model.proposed.ver1, clean_bang_rsf.test,
outcome = 'test')
pred_churn.a # out of 89 individuals, 66 found to have reported to churn membership. However in terms of model predicting outcomes, there was an error rate of 17.21%

##   Sample size of test (predict) data: 89
##        Number of deaths in test data: 66
##                 Number of grow trees: 2000
##   Average no. of grow terminal nodes: 14.676
##          Total no. of grow variables: 5
##        Resampling used to grow trees: swor
##     Resample size used to grow trees: 56
##                             Analysis: RSF
##                               Family: surv
##                  Test set error rate: 17.21%
```



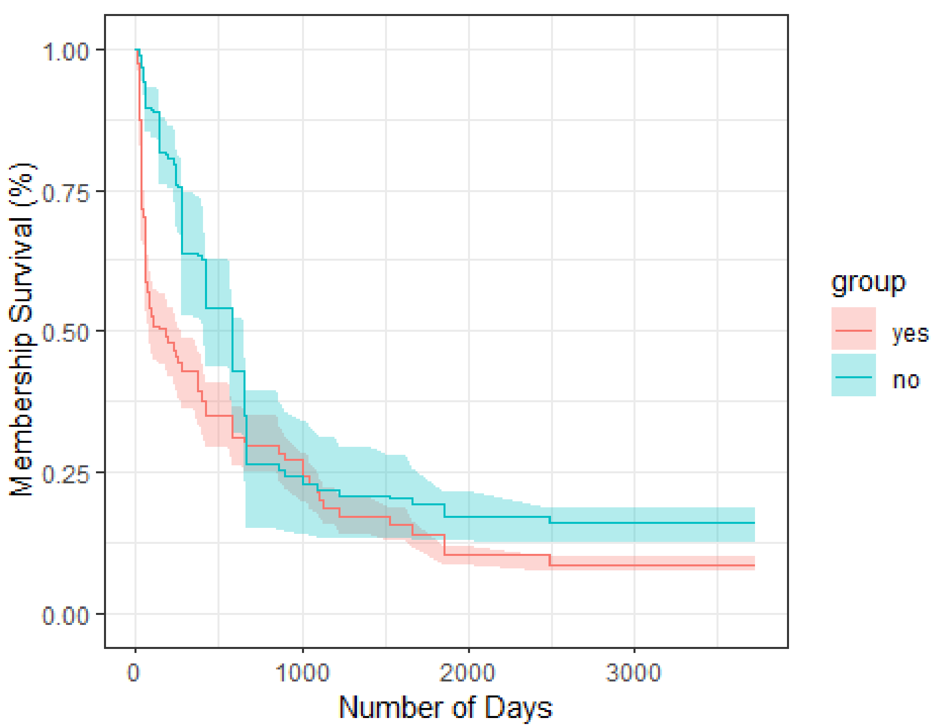
Predicted Length of Membership Retention by Attendance Rates



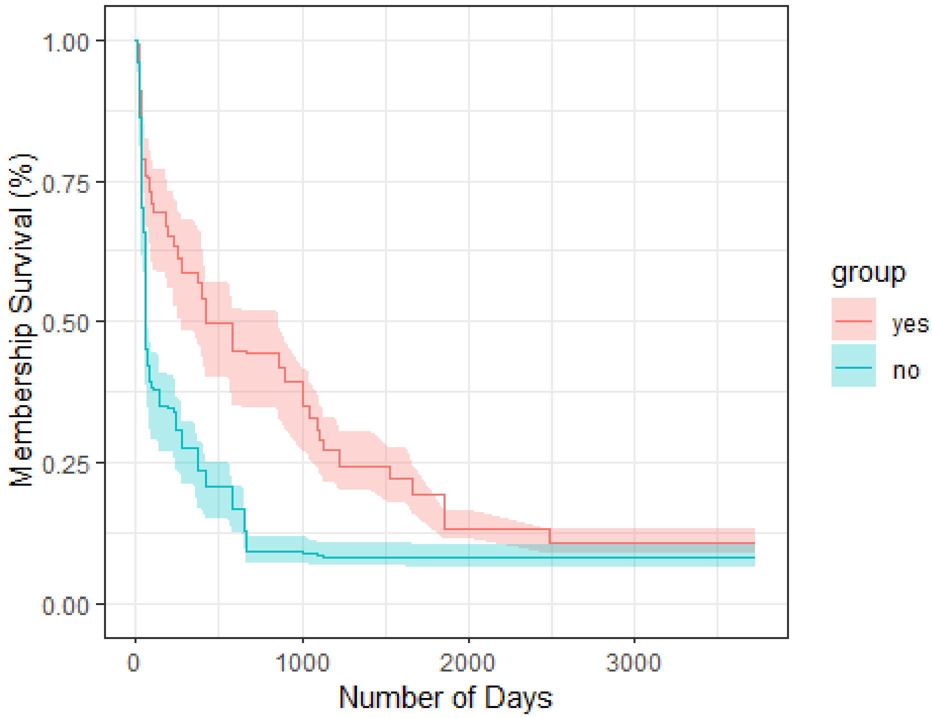
Predicted Length of Membership Retention by Age



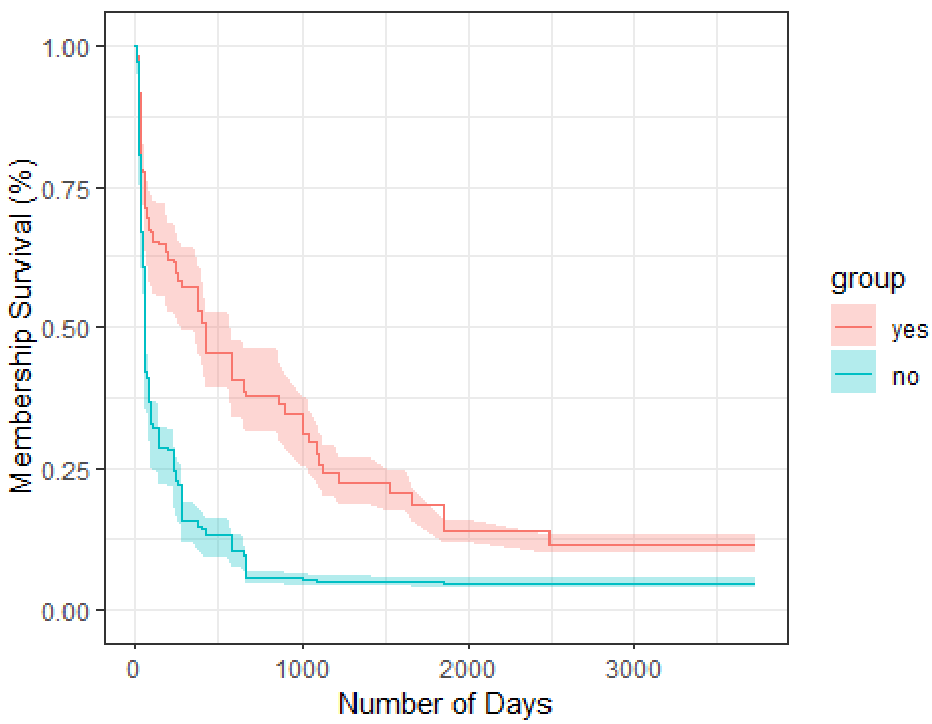
:mbership Retention by Employment Sector



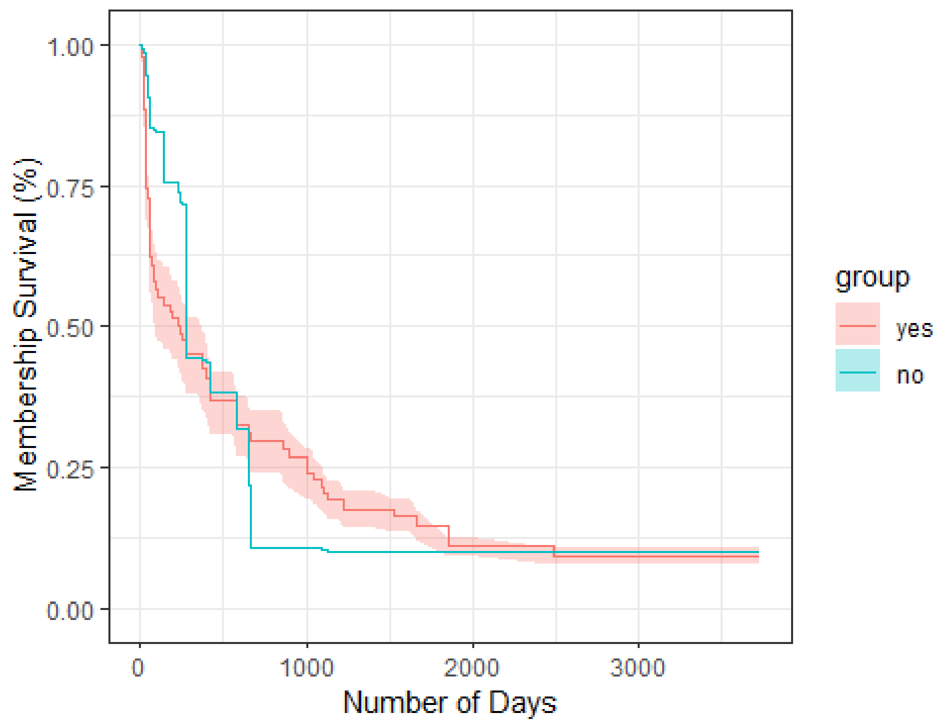
of Membership Retention by Ever Having an Email Interactions per Month



ship Retention by Ever Having a CX-Related Email Interactions per Month



ention by Ever Having a Scheduling-Related Email Interactions per Month



Retention by Ever Having a Service-Related Email Interactions per Month

CHURN ANALYSIS : COX-REGRESSION PROPORTIONAL HAZARD MODEL

Through a bi-directional Stepwise Regression to determine retained predictors of membership length before membership loss through a training dataset partitioning (80%), it was found that the variables that were retained were:

(a) number of non-billing related email interactions per month
(b) ever having a non-billing related email interaction
(c) percent composition of email interactions relating to scheduling
(d) percent composition of email interactions relating to CX
(e) ever having a CX-related email interaction

(f) attendance rate
(g) age
(h) weighted average monthly membership rate
(i) ever having a billing-related email interaction

(j) ever having a scheduling-related email interaction

However, this model failed to meet either the assumptions of proportionlity or non-linearity. The model met assumption following the removal of (i) ever having a billing-related email interaction, (ii) weighted average monthly membership rate, (iii) number of non-billing email interactions per month, along with the stratification by age groups. This new proposed model was found to have an error rate of 22.55% (C-statistic = 0.7745).

Using this model to predict outcomes with the test data, it was found that the error rate was 22.44% (C-statistic = 0.7766). Overall, the model appears to be decent.

In observing the impact of each of these predictors with the entire data set, it was found that:

- Compared to those that attended less than 50% of their possible allowance, those that attended 50%-59% of the time had a 0.551 times the change in odds of leaving whilst those that attended 70%-89% of the time had a 0.570 times change in the odds of leaving.

- Those that had ever had an email interaction in a given month were found to have a **6.28** times the change in odds of leaving as compared to those that had not had an email interaction in a given month.

- While there was no significant impact with respect to those that did have an scheduling-related email interaction as compared to those that did not, there appears to be a 0.985 times the change in odds of leaving for those that had a 1 factor increase in the percent composition of scheduling-related email interactions.

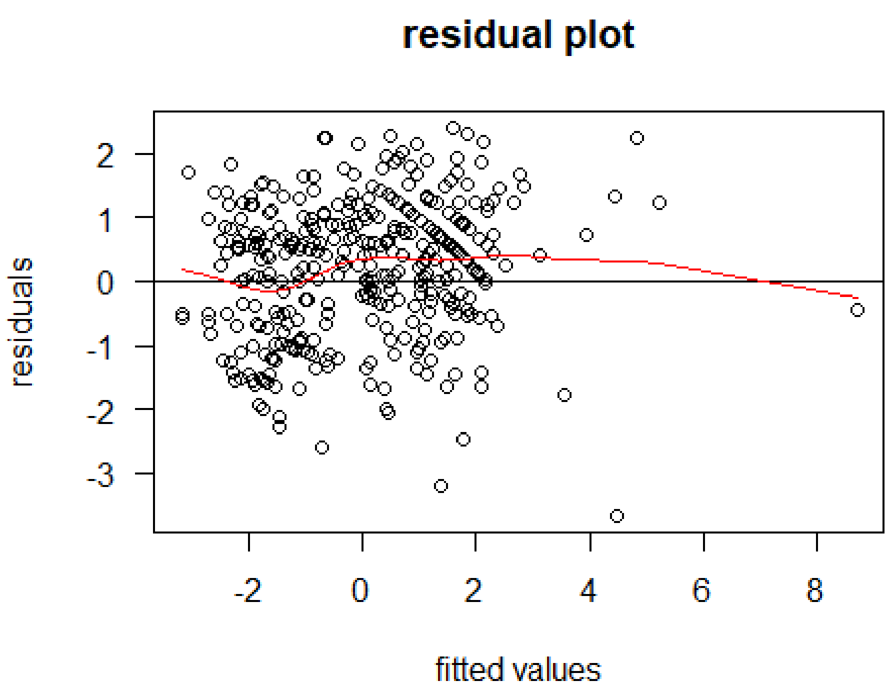
- Lastly with respect to those that had a 1 factor increase in the percent composition of CX-related email interactions, there was also a 1.03 times the change in odds of leaving. However, there was a large reduction in odds of leaving amongst those that ever had a CX-related email interaction as compared to those that did not.

```r
# STEP 1: Partition data set to a training + testing data set

training.index.cox = createDataPartition(clean_bang_select$length, p = 0.8,
list = FALSE)
clean_bang_cox.train = clean_bang_select[training.index.cox,]
clean_bang_cox.test = clean_bang_select[-training.index.cox,]

survival.object = with(clean_bang_select, Surv(length, became_former_member))

# STEP 2a: Model selection using backward selection

selectCox(Surv(length, became_former_member) ~  age_group +
                             employment_sector +
                             membership +
                             attendance_grouping_ver.1 +
                             monthly_rate_group +
                             ever_billing_issue +
                             num_emails_month +
                             ever_email_month +
                             new_per_ticket_scheduling +
                             ever_scheduling +
                             new_per_ticket_service +
                             ever_service +
                             new_per_ticket_cx +
```

```
                                ever_cx,
                                data = clean_bang_cox.train,
                                rule = "aic")$In

## [1] "num_emails_month"          "ever_email_month"
## [3] "new_per_ticket_scheduling" "new_per_ticket_service"
## [5] "ever_service"              "ever_cx"

# Top variables retained were: "num_emails_month", "ever_email_month",
"new_per_ticket_scheduling", "new_per_ticket_cx" and "ever_cx"

# STEP 2a: Model selection using bi-direction stepwise regression selection


start.cox = coxph(Surv(length, became_former_member) ~ 1, data =
clean_bang_cox.train)
all.cox = coxph(Surv(length, became_former_member) ~ age_group +
                                employment_sector +
                                membership +
                                attendance_grouping_ver.1 +
                                monthly_rate_group +
                                ever_billing_issue +
                                num_emails_month +
                                ever_email_month +
                                new_per_ticket_scheduling +
                                ever_scheduling +
                                new_per_ticket_service +
                                ever_service +
                                new_per_ticket_cx +
                                ever_cx, data = clean_bang_cox.train)

step(start.cox, direction = 'both', scope = formula(all.cox))

## Call:
## coxph(formula = Surv(length, became_former_member) ~ num_emails_month +
##     new_per_ticket_scheduling + ever_cx + ever_email_month +
##     new_per_ticket_cx + attendance_grouping_ver.1 + age_group +
##     ever_scheduling + ever_billing_issue, data = clean_bang_cox.train)
##
##                                      coef exp(coef)  se(coef)      z
p
## num_emails_month                 0.236043  1.266229  0.020893 11.298  <
2e-16
## new_per_ticket_scheduling       -0.014229  0.985872  0.004364 -3.260
0.00111
## ever_cxyes                      -2.061703  0.127237  0.243559 -8.465  <
2e-16
## ever_email_monthyes              1.647599  5.194495  0.224810  7.329
2.32e-13
## new_per_ticket_cx                0.026816  1.027179  0.006372  4.209
```

```
2.57e-05
## attendance_grouping_ver.150-59.99 -0.639316  0.527653  0.213196 -2.999
0.00271
## attendance_grouping_ver.160-69.99 -0.462470  0.629726  0.223210 -2.072
0.03827
## attendance_grouping_ver.170-79.99 -0.485612  0.615320  0.221934 -2.188
0.02866
## attendance_grouping_ver.180-89.99 -0.455561  0.634092  0.232720 -1.958
0.05028
## attendance_grouping_ver.190-100   -0.117967  0.888725  0.180385 -0.654
0.51313
## age_group18-29                     1.108708  3.030441  0.480538  2.307
0.02104
## age_group30-44                     0.802486  2.231080  0.465659  1.723
0.08483
## age_group45-64                     0.500639  1.649775  0.478934  1.045
0.29587
## age_group65+                       0.217688  1.243200  0.653308  0.333
0.73898
## ever_schedulingyes                -0.541749  0.581730  0.237778 -2.278
0.02270
## ever_billing_issueyes             -0.270696  0.762848  0.155513 -1.741
0.08174
##
## Likelihood ratio test=349.4  on 16 df, p=< 2.2e-16
## n= 358, number of events= 283

# new_per_ticket_scheduling, num_emails_month, ever_cx, ever_email_month,
new_per_ticket_cx, attendance_grouping_ver.1, age_group, monthly_rate_group,
ever_billing_issue and ever_scheduling



testing = cph(Surv(length, became_former_member) ~
                num_emails_month +
                ever_email_month +
                new_per_ticket_scheduling +
                new_per_ticket_cx +
                ever_cx +
                attendance_grouping_ver.1 +
                age_group +
                monthly_rate_group +
                ever_billing_issue +
                ever_scheduling, data = clean_bang_cox.train, x = T, y = T,
surv = T)

# Step 3a: Testing Assumption of the Model

cox.zph(testing) # Issue with num_emails_month + new_per_ticket_cx
```

```
##                                 chisq df       p
## num_emails_month             6.32e+00  1   0.012
## ever_email_month             3.42e+00  1   0.064
## new_per_ticket_scheduling 2.21e-06  1   0.999
## new_per_ticket_cx            5.53e+00  1   0.019
## ever_cx                      3.96e+00  1   0.047
## attendance_grouping_ver.1 5.94e+00  5   0.312
## age_group                    2.10e+00  4   0.718
## monthly_rate_group           5.51e+01 11 7.4e-08
## ever_billing_issue           2.94e-01  1   0.588
## ever_scheduling              6.66e-01  1   0.415
## GLOBAL                       8.34e+01 27 1.1e-07

plot(   predict(testing),
    residuals(testing, type = "deviance"),
    xlab = "fitted values",
    ylab = "residuals",
    main = "residual plot", las = 1
)

abline(h = 0)


lines(smooth.spline(predict(testing), residuals(testing, type = 'deviance')),
col = 'red')
```



```r
# Not even close to satisfying the assumptions of non-linearity; need to do
some re-working



# Step 3b: Rework predictor selection + Re-testing Assumption of the Model

proposed.cox.model.train = cph(Surv(length, became_former_member) ~
                ever_email_month +
                new_per_ticket_scheduling +
                new_per_ticket_cx +
                ever_cx +
                attendance_grouping_ver.1 +
                strat(age_group) +
                ever_scheduling, data = clean_bang_cox.train, x = T, y = T,
surv = T)


cox.zph(proposed.cox.model.train) # Holds the assumption of proportionality

##                              chisq df     p
## ever_email_month            4.6284  1 0.031
## new_per_ticket_scheduling   0.0133  1 0.908
## new_per_ticket_cx           3.2393  1 0.072
## ever_cx                     1.2691  1 0.260
## attendance_grouping_ver.1   2.3750  5 0.795
## ever_scheduling             1.9318  1 0.165
## GLOBAL                     12.1511 10 0.275

plot(
    predict(proposed.cox.model.train),
    residuals(proposed.cox.model.train, type = "deviance"),
    xlab = "fitted values",
    ylab = "residuals",
    main = "residual plot", las = 1
)

abline(h = 0)

lines(smooth.spline(predict(proposed.cox.model.train),
residuals(proposed.cox.model.train, type = 'deviance')), col = 'red') # meh

train_surv = with(clean_bang_cox.train, Surv(length, became_former_member))
train.estimates = survest(proposed.cox.model.train, newdata =
clean_bang_cox.train, times = 69)$surv
rcorr.cens(train.estimates, train_surv) # c = 0.7781 (aka. err.rate = 22.19%)

##        C Index            Dxy           S.D.              n        missing
##   7.781534e-01   5.563068e-01   2.341529e-02   3.580000e+02   0.000000e+00
```

```
##     uncensored Relevant Pairs     Concordant      Uncertain
##   2.830000e+02   1.073760e+05   8.355500e+04   1.874000e+04

# Step 4: validating my proposed mode with test data set

test_surv = with(clean_bang_cox.test, Surv(length, became_former_member)) #
this is the survival object in which to test against
estimates = survest(proposed.cox.model.train, newdata = clean_bang_cox.test,
times = 69)$surv # time is just arbitrary here; survival estimates based on
the training model using the test data set
rcorr.cens(estimates, test_surv)  # C = 0.7606 or err.rate = 23.94%

##        C Index            Dxy           S.D.              n        missing
##   7.606494e-01   5.212988e-01   5.083363e-02   8.900000e+01   0.000000e+00
##     uncensored Relevant Pairs     Concordant      Uncertain
##   7.000000e+01   6.714000e+03   5.107000e+03   9.080000e+02

# Step 5: Summary of the whole dataset using the proposed model

cox.model.churn = coxph(Surv(length, became_former_member) ~
                ever_email_month +
                new_per_ticket_scheduling +
                new_per_ticket_cx +
                ever_cx +
                attendance_grouping_ver.1 +
                strata(age_group) +
                ever_scheduling, data = clean_bang_select)

summary(cox.model.churn) # C-statistic = 0.781

## Call:
## coxph(formula = Surv(length, became_former_member) ~ ever_email_month +
##     new_per_ticket_scheduling + new_per_ticket_cx + ever_cx +
##     attendance_grouping_ver.1 + strata(age_group) + ever_scheduling,
##     data = clean_bang_select)
##
##   n= 447, number of events= 353
##
##                                      coef exp(coef)  se(coef)       z
## ever_email_monthyes              1.838344  6.286117  0.206464   8.904
## new_per_ticket_scheduling       -0.015299  0.984817  0.003925  -3.898
## new_per_ticket_cx                0.027677  1.028064  0.005403   5.123
## ever_cxyes                      -2.112674  0.120914  0.205221 -10.295
## attendance_grouping_ver.150-59.99 -0.596172  0.550916  0.194012  -3.073
## attendance_grouping_ver.160-69.99 -0.112368  0.893715  0.180600  -0.622
## attendance_grouping_ver.170-79.99 -0.556584  0.573164  0.198920  -2.798
## attendance_grouping_ver.180-89.99 -0.561434  0.570391  0.217047  -2.587
## attendance_grouping_ver.190-100   -0.182059  0.833552  0.157828  -1.154
## ever_schedulingyes              -0.177018  0.837765  0.208249  -0.850
##                                 Pr(>|z|)
```

```
## ever_email_monthyes              < 2e-16 ***
## new_per_ticket_scheduling      9.70e-05 ***
## new_per_ticket_cx              3.01e-07 ***
## ever_cxyes                      < 2e-16 ***
## attendance_grouping_ver.150-59.99  0.00212 **
## attendance_grouping_ver.160-69.99  0.53381
## attendance_grouping_ver.170-79.99  0.00514 **
## attendance_grouping_ver.180-89.99  0.00969 **
## attendance_grouping_ver.190-100    0.24869
## ever_schedulingyes                 0.39531
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##                                   exp(coef) exp(-coef) lower .95 upper .95
## ever_email_monthyes                  6.2861     0.1591   4.19411    9.4216
## new_per_ticket_scheduling            0.9848     1.0154   0.97727    0.9924
## new_per_ticket_cx                    1.0281     0.9727   1.01723    1.0390
## ever_cxyes                           0.1209     8.2703   0.08087    0.1808
## attendance_grouping_ver.150-59.99    0.5509     1.8152   0.37665    0.8058
## attendance_grouping_ver.160-69.99    0.8937     1.1189   0.62730    1.2733
## attendance_grouping_ver.170-79.99    0.5732     1.7447   0.38811    0.8464
## attendance_grouping_ver.180-89.99    0.5704     1.7532   0.37275    0.8728
## attendance_grouping_ver.190-100      0.8336     1.1997   0.61177    1.1357
## ever_schedulingyes                   0.8378     1.1937   0.55701    1.2600
##
## Concordance= 0.781  (se = 0.011 )
## Likelihood ratio test= 294.5  on 10 df,   p=<2e-16
## Wald test            = 259.2  on 10 df,   p=<2e-16
## Score (logrank) test = 300.5  on 10 df,   p=<2e-16
```

```
AIC(cox.model.churn) # 2649.277
```

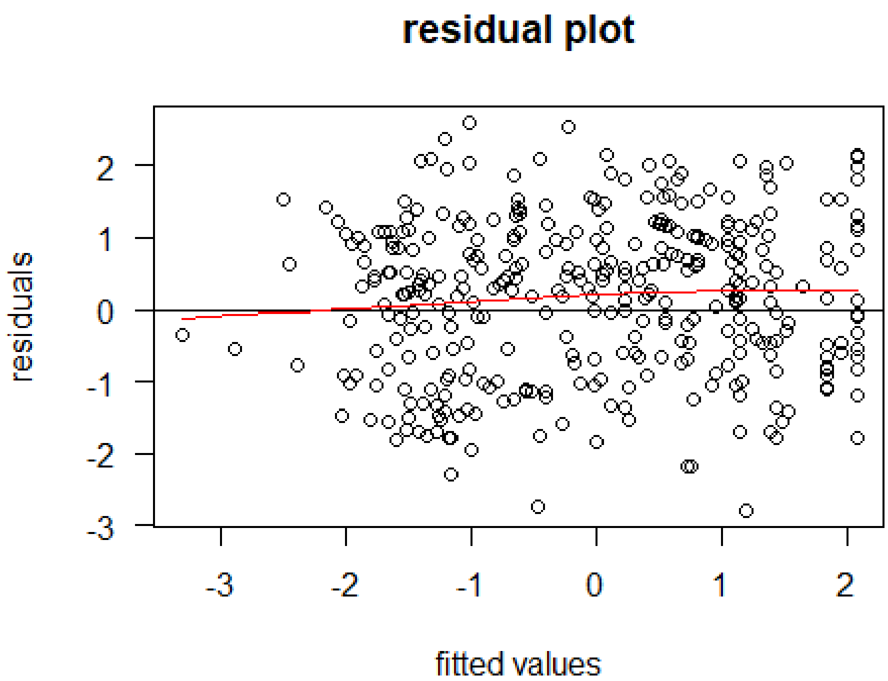
```
## [1] 2649.227
```

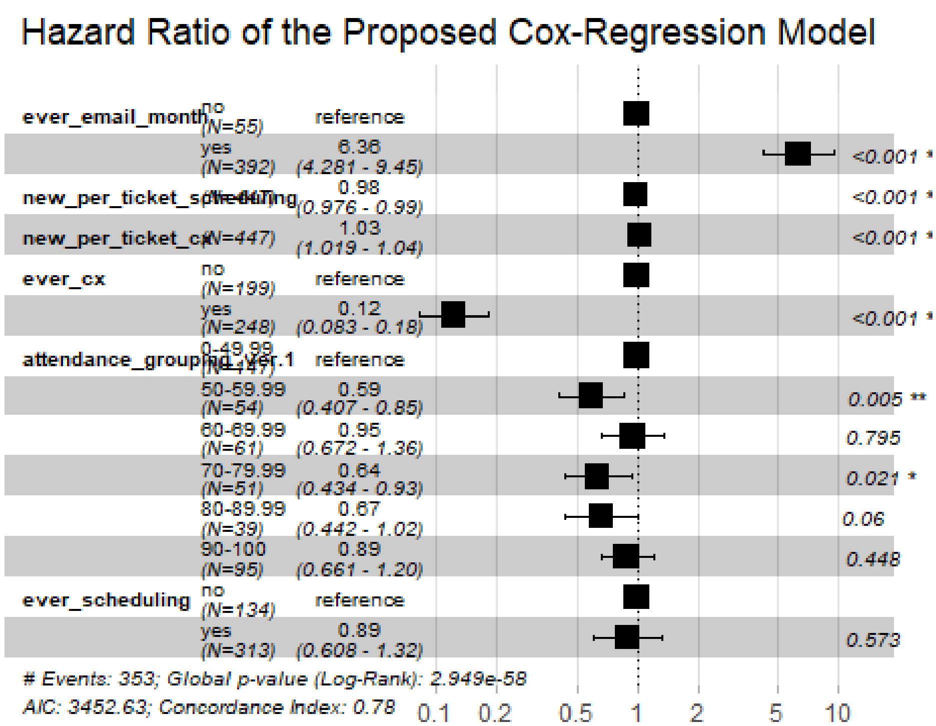
```
vif(cox.model.churn) # No issue of collinearity here
```

```
##               ever_email_monthyes         new_per_ticket_scheduling
##                          1.172078                          3.095399
##                 new_per_ticket_cx                        ever_cxyes
##                          3.019967                          2.660034
## attendance_grouping_ver.150-59.99 attendance_grouping_ver.160-69.99
##                          1.226746                          1.281800
## attendance_grouping_ver.170-79.99 attendance_grouping_ver.180-89.99
##                          1.204111                          1.168101
##   attendance_grouping_ver.190-100                ever_schedulingyes
##                          1.442219                          2.552844
```

```
cox.model.churn.a = coxph(Surv(length, became_former_member) ~
                ever_email_month +
                new_per_ticket_scheduling +
                new_per_ticket_cx +
```

```
                ever_cx +
                attendance_grouping_ver.1 +
                ever_scheduling, data = clean_bang_select)

ggforest(cox.model.churn.a, main = "Hazard Ratio of the Proposed Cox-
Regression Model")

## Warning in .get_data(model, data = data): The `data` argument is not
provided.
## Data will be extracted from model fit.
```






```
# Note: no idea on how to include the strata function into the ggforest
```

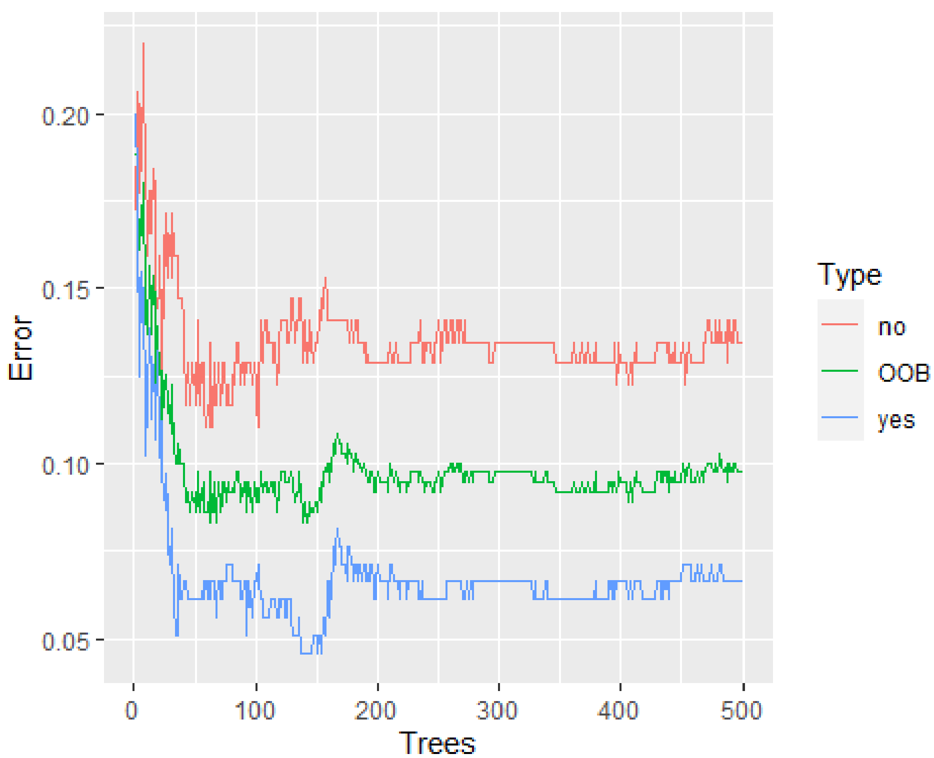
Modeling retention status at 3-/6- and 12- months.

RETENTION ANALYSIS: Membership status at 3-Months via Random Forest

Using the random survival forest specific dataset, I've split the data set 80:20 with respect to training:test. In forming the training model, which has an error rate of **6.69%**, it was found that the error rate in predicting membership length to churn with the test data was **4.55%**. So really a large differential. Looking at the various ways to modify the parameters, it was found that the error rate more-or-less stabilized after 1000 trees as evident by the marginal differences in error rates at the higher number of trees. However, in terms of tuning this model, I've adjusted the model to include ntree = 2000 and mtry at 4. Examining the importance of each variable used in this model, it was found that number of non-billing email interaction played the largest role, followed by the percent composition of non=billing related email interactions (scheduling, service and CX).

```
# Step 1: Create a specific data set to be used for retention status analysis

clean_bang_retention_3m = clean_bang_select %>%
  select(
    age_group,
    employment_sector,
    retention_3m,
    avg_monthly_rate,
    attendance_grouping_ver.1,
    ever_email_month,
    num_emails_month,
    ever_billing_issue,
    ever_cx,
    new_per_ticket_cx,
    ever_scheduling,
    new_per_ticket_scheduling,
    ever_service,
    new_per_ticket_service
  )

# Step 2: create a partition of this data set by splitting it based on
retention status at 3 Months

trainIndex_3m = createDataPartition(clean_bang_retention_3m$retention_3m, p =
0.8, list = FALSE)
clean_bang_retention_3m.train = clean_bang_retention_3m[trainIndex_3m,]
clean_bang_retention_3m.test = clean_bang_retention_3m[-trainIndex_3m,]

# Step 3: Create a random forest model using training data

training.model.3m = randomForest(retention_3m ~., data =
clean_bang_retention_3m.train, proximity = T)
training.model.3m # OOB error rate is 9.75%
```

```
##
## Call:
##  randomForest(formula = retention_3m ~ ., data =
clean_bang_retention_3m.train,      proximity = T)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 9.75%
## Confusion matrix:
##      no yes class.error
## no  141  22  0.13496933
## yes  13 183  0.06632653

# Step 4: Create a data frame to see how the error rate changes as a function
of increasing number of trees (currently capped at 500)

oob.error.data.3m = data.frame(
  Trees = rep(1:nrow(training.model.3m$err.rate), times = 3),
  Type = rep(c("OOB", "no", 'yes'), each = nrow(training.model.3m$err.rate)),
  Error = c(training.model.3m$err.rate[, "OOB"],
            training.model.3m$err.rate[,"no"],
            training.model.3m$err.rate[, 'yes']))
```



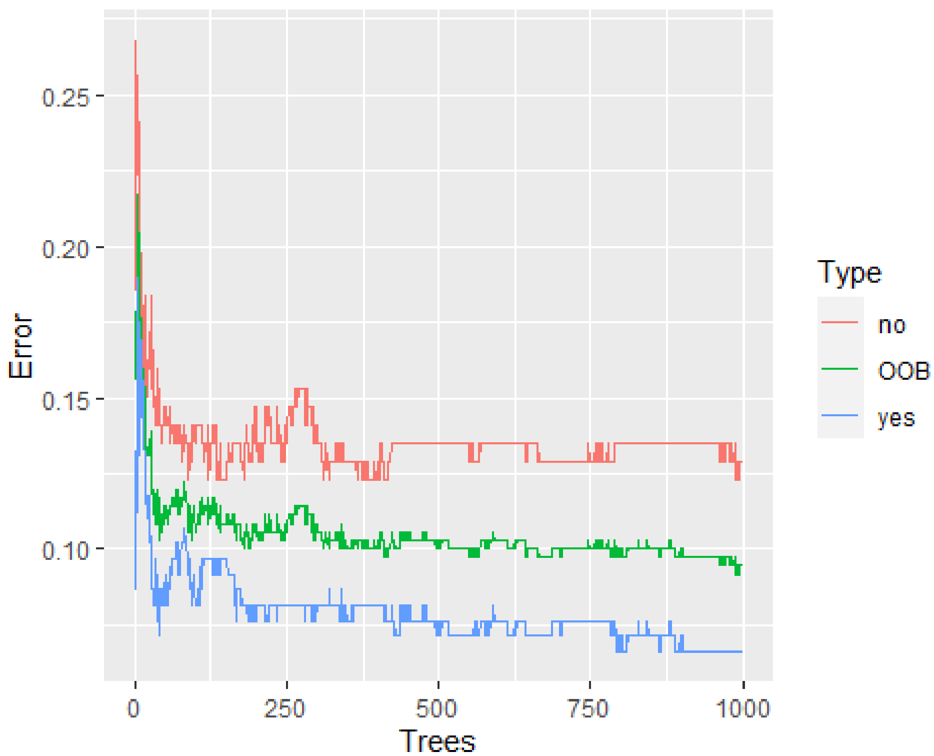
```
# Step 4a: Add more trees and see what happens:

training.model.3m_ver1 = randomForest(retention_3m ~., data =
clean_bang_retention_3m.train, proximity = T, ntree = 1000)
training.model.3m_ver2 = randomForest(retention_3m ~., data =
clean_bang_retention_3m.train, proximity = T, ntree = 2000)
training.model.3m_ver3 = randomForest(retention_3m ~., data =
clean_bang_retention_3m.train, proximity = T, ntree = 3000)

training.model.3m # REFERENCE

##
## Call:
##  randomForest(formula = retention_3m ~ ., data =
clean_bang_retention_3m.train,      proximity = T)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 9.75%
## Confusion matrix:
##      no yes class.error
## no  141  22  0.13496933
## yes  13 183  0.06632653

training.model.3m_ver1 # 9.47%

##
## Call:
##  randomForest(formula = retention_3m ~ ., data =
clean_bang_retention_3m.train,      proximity = T, ntree = 1000)
##                Type of random forest: classification
##                      Number of trees: 1000
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 9.47%
## Confusion matrix:
##      no yes class.error
## no  142  21  0.12883436
## yes  13 183  0.06632653

training.model.3m_ver2 # 10.03%

##
## Call:
##  randomForest(formula = retention_3m ~ ., data =
clean_bang_retention_3m.train,      proximity = T, ntree = 2000)
##                Type of random forest: classification
##                      Number of trees: 2000
## No. of variables tried at each split: 3
##
```

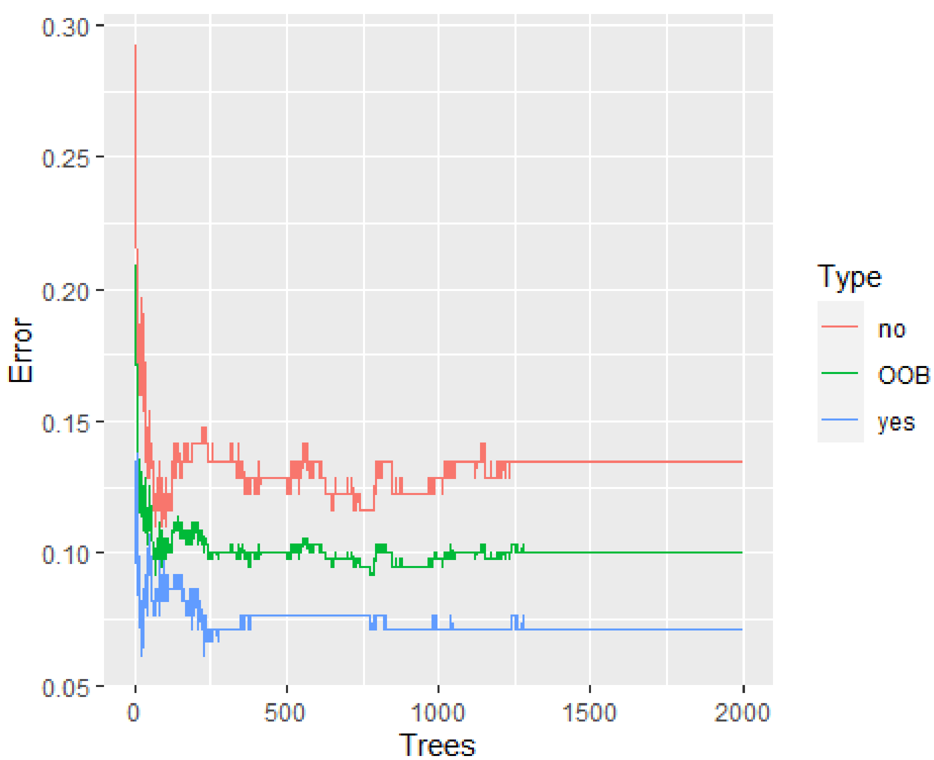
```
##         OOB estimate of  error rate: 10.03%
## Confusion matrix:
##      no yes class.error
## no  141  22  0.13496933
## yes  14 182  0.07142857
```

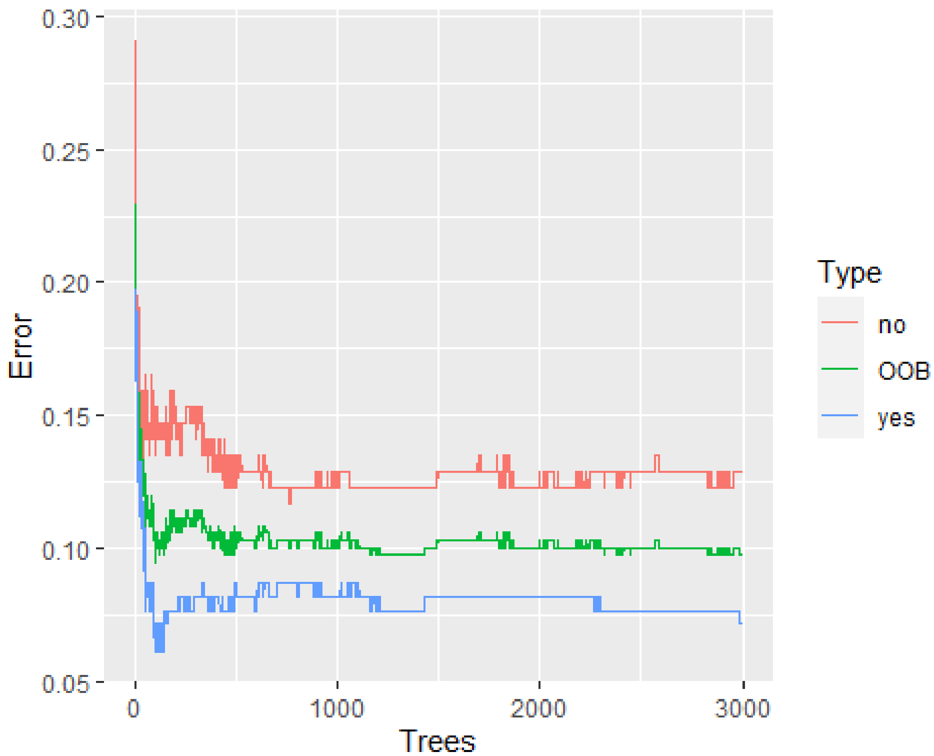
```
training.model.3m_ver3 # 9.75%
```

```
##
## Call:
##  randomForest(formula = retention_3m ~ ., data =
clean_bang_retention_3m.train,      proximity = T, ntree = 3000)
##                Type of random forest: classification
##                      Number of trees: 3000
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 9.75%
## Confusion matrix:
##      no yes class.error
## no  142  21  0.12883436
## yes  14 182  0.07142857
```

```
oob.error.data.3m_ver1 = data.frame(
  Trees = rep(1:nrow(training.model.3m_ver1$err.rate), times = 3),
  Type = rep(c("OOB", "no", 'yes'), each =
nrow(training.model.3m_ver1$err.rate)),
  Error = c(training.model.3m_ver1$err.rate[, "OOB"],
            training.model.3m_ver1$err.rate[,"no"],
            training.model.3m_ver1$err.rate[, 'yes']))

oob.error.data.3m_ver2 = data.frame(
  Trees = rep(1:nrow(training.model.3m_ver2$err.rate), times = 3),
  Type = rep(c("OOB", "no", 'yes'), each =
nrow(training.model.3m_ver2$err.rate)),
  Error = c(training.model.3m_ver2$err.rate[, "OOB"],
            training.model.3m_ver2$err.rate[,"no"],
            training.model.3m_ver2$err.rate[, 'yes']))

oob.error.data.3m_ver3 = data.frame(
  Trees = rep(1:nrow(training.model.3m_ver3$err.rate), times = 3),
  Type = rep(c("OOB", "no", 'yes'), each =
nrow(training.model.3m_ver3$err.rate)),
  Error = c(training.model.3m_ver3$err.rate[, "OOB"],
            training.model.3m_ver3$err.rate[,"no"],
            training.model.3m_ver3$err.rate[, 'yes']))
```


Error
0.25
0.20
0.15
0.10
0
250
500
750
1000
Trees
Type
no
OOB
yes


Error
0.30
0.25
0.20
0.15
0.10
0.05
0
500
1000
1500
2000
Trees
Type
no
OOB
yes

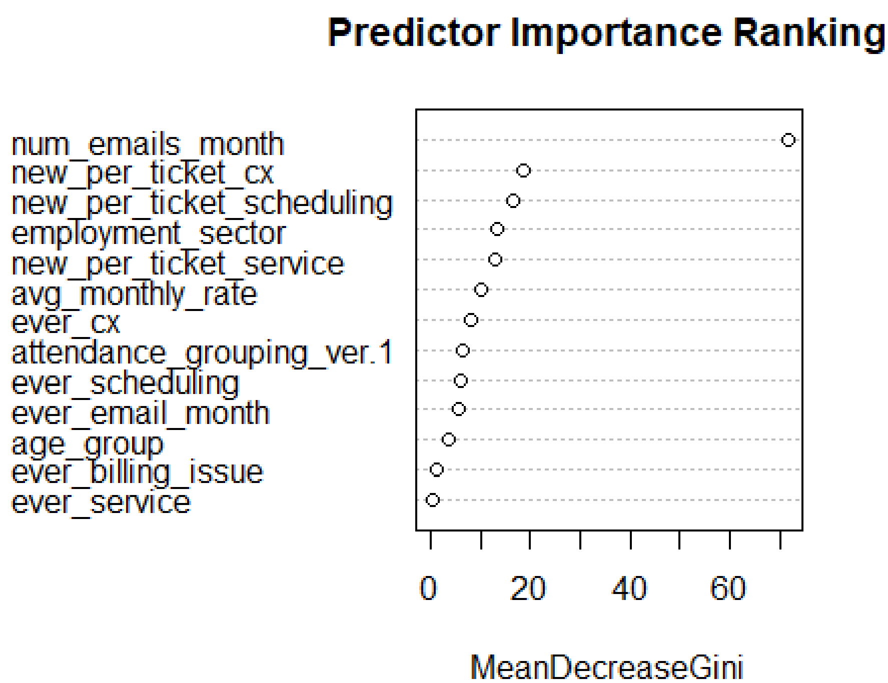



```
# Looks like we did a worse job with increasing number of trees, but this
leveled off after 2000.

# STEP 3B: Fine tuning mtry

oob.values <- vector(length = 10)
for(i in 1:10) {
  temp.model <- randomForest(retention_3m ~., data =
clean_bang_retention_3m.train, mtry = i, ntree = 2000)
  oob.values[i] <- temp.model$err.rate[nrow(temp.model$err.rate), 1]
}

oob.values

##  [1] 0.14206128 0.09749304 0.08635097 0.09749304 0.11142061 0.10027855
##  [7] 0.10306407 0.10306407 0.10306407 0.10027855

# Looks like optimal value is 4

proposed.training.model.3m =  randomForest(retention_3m ~., data =
clean_bang_retention_3m.train, proximity = T, mtry = 4, ntree = 2000) #
err.rate = 6.69%


proposed.training.model.3m$confusion
```

```
##      no yes class.error
## no  142  21  0.12883436
## yes  15 181  0.07653061

# Step 4: Test this proposed model against testing data

pred_3m_rf <- predict(proposed.training.model.3m, newdata =
clean_bang_retention_3m.test)


confusionMatrix(pred_3m_rf, clean_bang_retention_3m.test$retention_3m) #
accuracy = 0.9545 or err.rate of 4.55%

## Confusion Matrix and Statistics
##
##           Reference
## Prediction no yes
##        no  37   1
##        yes  3  47
##
##                Accuracy : 0.9545
##                  95% CI : (0.8877, 0.9875)
##     No Information Rate : 0.5455
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9079
##
##  Mcnemar's Test P-Value : 0.6171
##
##             Sensitivity : 0.9250
##             Specificity : 0.9792
##          Pos Pred Value : 0.9737
##          Neg Pred Value : 0.9400
##              Prevalence : 0.4545
##          Detection Rate : 0.4205
##    Detection Prevalence : 0.4318
##       Balanced Accuracy : 0.9521
##
##        'Positive' Class : no
##

# STEP 5: Determining which variables are important predictors

varImp(proposed.training.model.3m)

##                              Overall
## age_group                  3.7686084
## employment_sector         13.5593857
## avg_monthly_rate          10.2912174
## attendance_grouping_ver.1  6.4924624
## ever_email_month           5.6714667
```

```
## num_emails_month          71.7819837
## ever_billing_issue         1.3997070
## ever_cx                    7.9958873
## new_per_ticket_cx         18.6814021
## ever_scheduling            5.9690840
## new_per_ticket_scheduling 16.7379079
## ever_service               0.5452137
## new_per_ticket_service    13.1416123
```

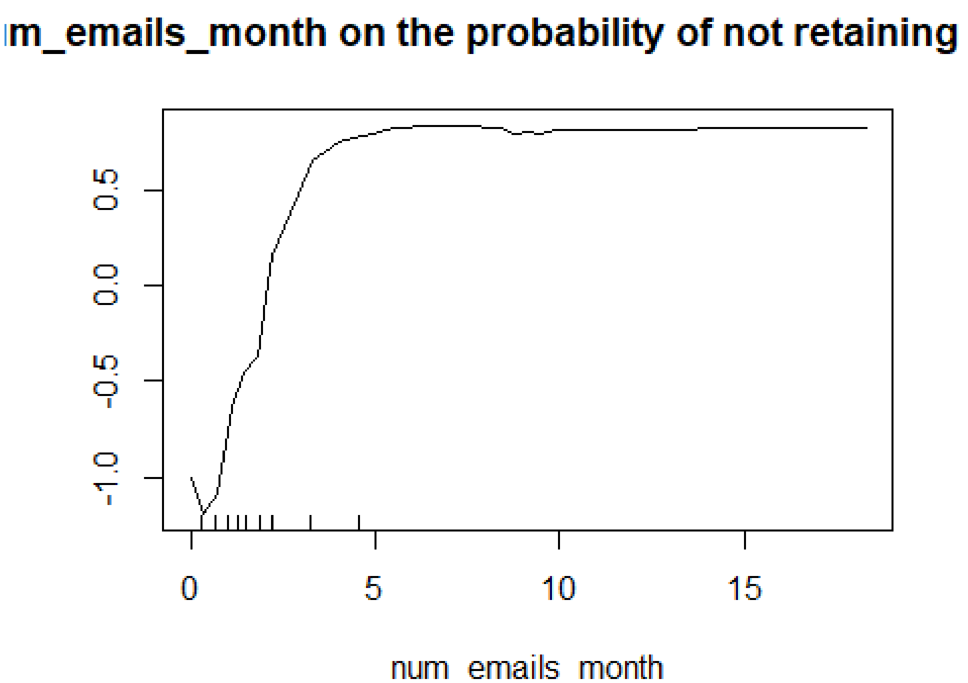
```
varImpPlot(proposed.training.model.3m, sort = T, main = "Predictor Importance Ranking")
```

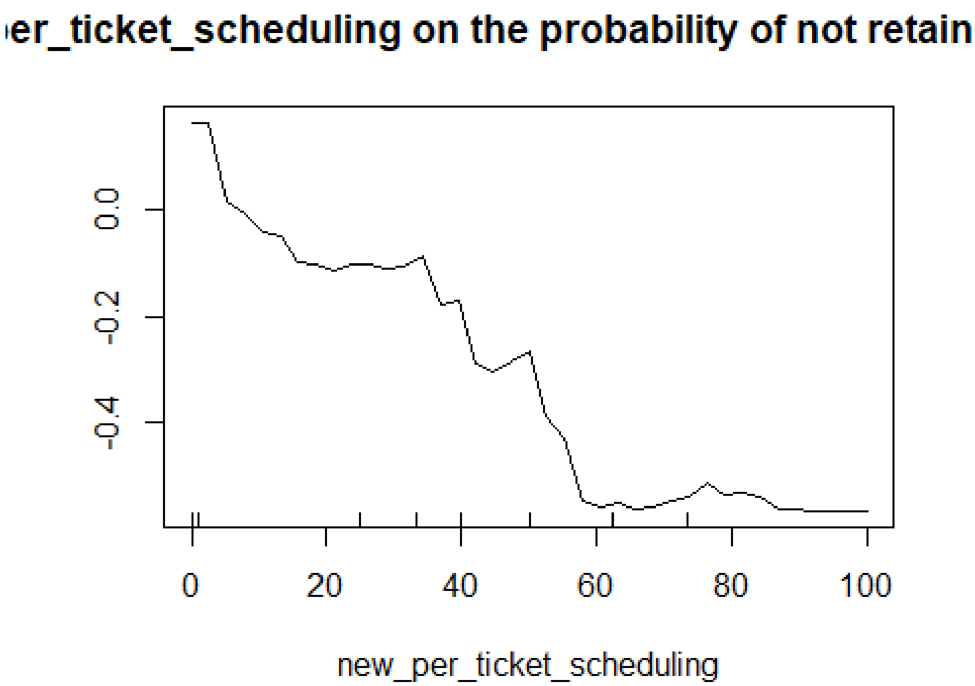


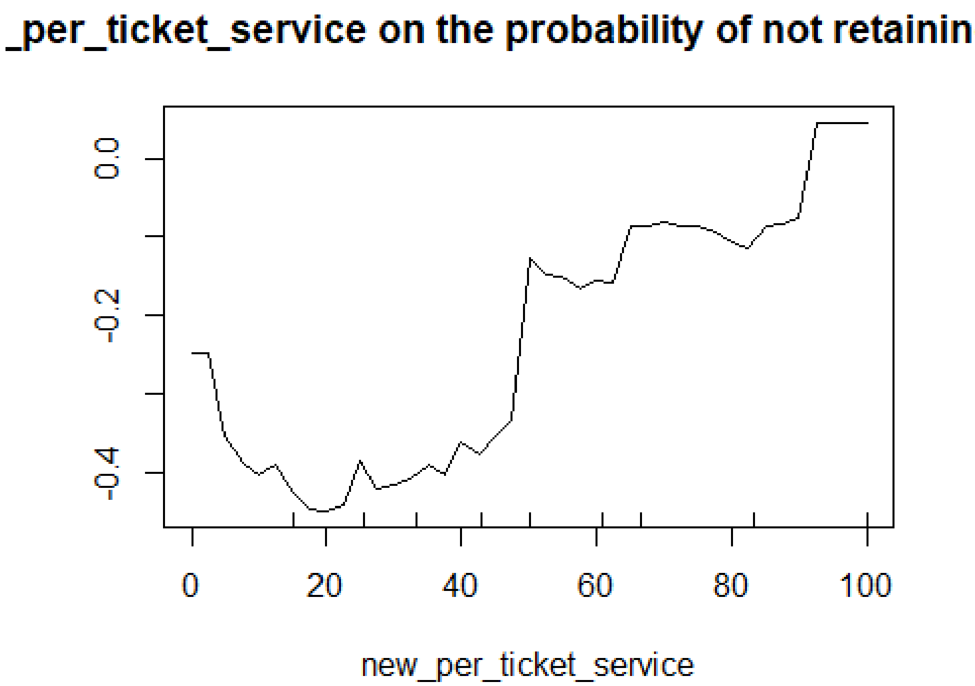

```
importance(proposed.training.model.3m)
```

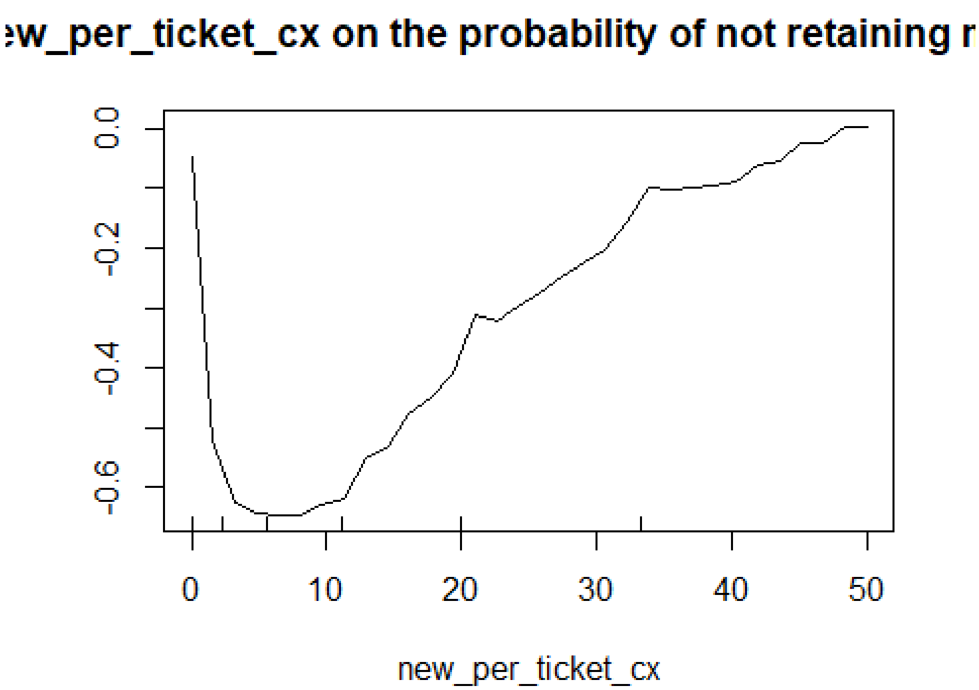
```
##                           MeanDecreaseGini
## age_group                        3.7686084
## employment_sector               13.5593857
## avg_monthly_rate                10.2912174
## attendance_grouping_ver.1        6.4924624
## ever_email_month                 5.6714667
## num_emails_month                71.7819837
## ever_billing_issue               1.3997070
## ever_cx                          7.9958873
## new_per_ticket_cx               18.6814021
## ever_scheduling                  5.9690840
## new_per_ticket_scheduling       16.7379079
```

```
## ever_service                          0.5452137
## new_per_ticket_service               13.1416123

varUsed(proposed.training.model.3m)

##  [1]  4093  9340  9273  6096  1752 14407  1638  1629  6455  1693  6616
557
## [13]  6760

# Examining the model, it seems that num_emails_month played the most
important role in predicting outcomes follwed by the percent compositions
from each of the non-billing email interactions ( CX > scheduling > service).

# Step 5a: Examining the effects of each variable on retention status (Top 4
predictors)
```




er_ticket_scheduling on the probability of not retain
0.0
-0.2
-0.4
0
20
40
60
80
100
new_per_ticket_scheduling





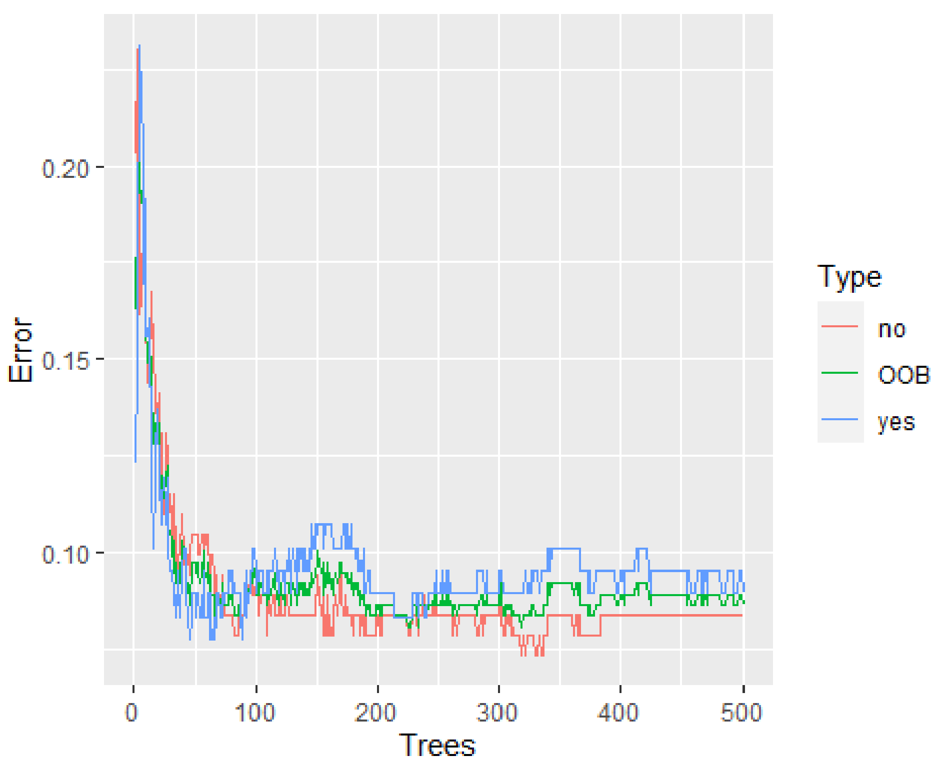
RETENTION ANALYSIS: Membership status at 6-Months via Random Forest

Using the random survival forest specific dataset, I've split the data set 80:20 with respect to training:test. In forming the training model, which has an error rate of **8.64%**, it was found that the error rate in predicting membership length to churn with the test data was **10.33%**, which is not a very good sign. Looking at the various ways to modify the parameters, it was found that the error rate more-or-less stabilized after 1000 trees as evident by the marginal differences in error rates at the higher number of trees. However, in terms of tuning this model, I've adjusted the model to include ntree = 1000 and mtry at 3. Examining the importance of each variable used in this model, it was found that number of non-billing email interaction played the largest role, followed by the percent composition of non-billing related email interactions (scheduling, service and CX).

```
clean_bang_retention_6m = clean_bang_select %>%
  select(
    age_group,
    employment_sector,
    retention_6m,
    avg_monthly_rate,
    attendance_grouping_ver.1,
    ever_email_month,
    num_emails_month,
    ever_billing_issue,
    ever_cx,
    new_per_ticket_cx,
    ever_scheduling,
    new_per_ticket_scheduling,
    ever_service,
    new_per_ticket_service
  )

# Step 2: create a partition of this data set by splitting it based on
retention status at 6 Months

trainIndex_6m = createDataPartition(clean_bang_retention_6m$retention_6m, p =
0.8, list = FALSE)
clean_bang_retention_6m.train = clean_bang_retention_6m[trainIndex_6m,]
clean_bang_retention_6m.test = clean_bang_retention_6m[-trainIndex_6m,]

# Step 3: Create a random forest model using training data

training.model.6m = randomForest(retention_6m ~., data =
clean_bang_retention_6m.train, proximity = T)
training.model.6m # OOB error rate is 8.64%

##
## Call:
##  randomForest(formula = retention_6m ~ ., data =
clean_bang_retention_6m.train,      proximity = T)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 3
##
```

```
##         OOB estimate of  error rate: 8.64%
## Confusion matrix:
##      no yes class.error
## no  175  16  0.08376963
## yes  15 153  0.08928571
```

```
# Step 4: Create a data frame to see how the error rate changes as a function
of increasing number of trees (currently capped at 500)

oob.error.data.6m = data.frame(
  Trees = rep(1:nrow(training.model.6m$err.rate), times = 3),
  Type = rep(c("OOB", "no", 'yes'), each = nrow(training.model.6m$err.rate)),
  Error = c(training.model.6m$err.rate[, "OOB"],
            training.model.6m$err.rate[,"no"],
            training.model.6m$err.rate[, 'yes']))
```




```
# Step 4a: Add more trees and see what happens:

training.model.6m_ver1 = randomForest(retention_6m ~., data =
clean_bang_retention_6m.train, proximity = T, ntree = 1000)
training.model.6m_ver2 = randomForest(retention_6m ~., data =
clean_bang_retention_6m.train, proximity = T, ntree = 2000)
training.model.6m_ver3 = randomForest(retention_6m ~., data =
clean_bang_retention_6m.train, proximity = T, ntree = 3000)

training.model.6m # REFERENCE
```

```
##
## Call:
##  randomForest(formula = retention_6m ~ ., data =
clean_bang_retention_6m.train,      proximity = T)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 8.64%
## Confusion matrix:
##      no yes class.error
## no  175  16  0.08376963
## yes  15 153  0.08928571
```

```
training.model.6m_ver1 # 8.91%
```

```
##
## Call:
##  randomForest(formula = retention_6m ~ ., data =
clean_bang_retention_6m.train,      proximity = T, ntree = 1000)
##                Type of random forest: classification
##                      Number of trees: 1000
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 8.91%
## Confusion matrix:
##      no yes class.error
## no  174  17  0.08900524
## yes  15 153  0.08928571
```

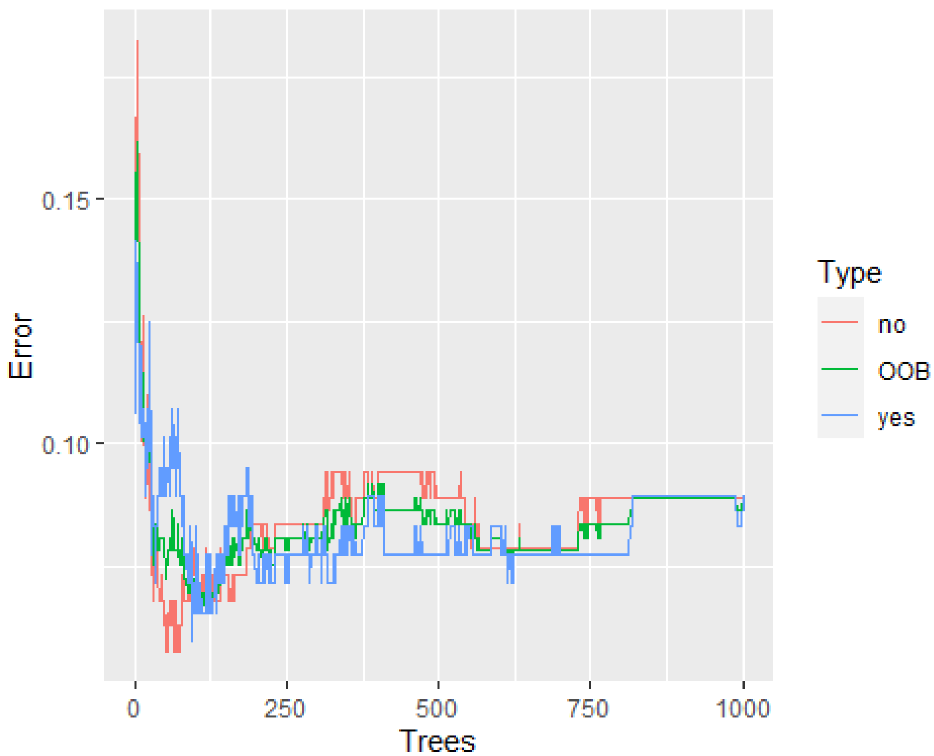
```
training.model.6m_ver2 # 8.91%
```

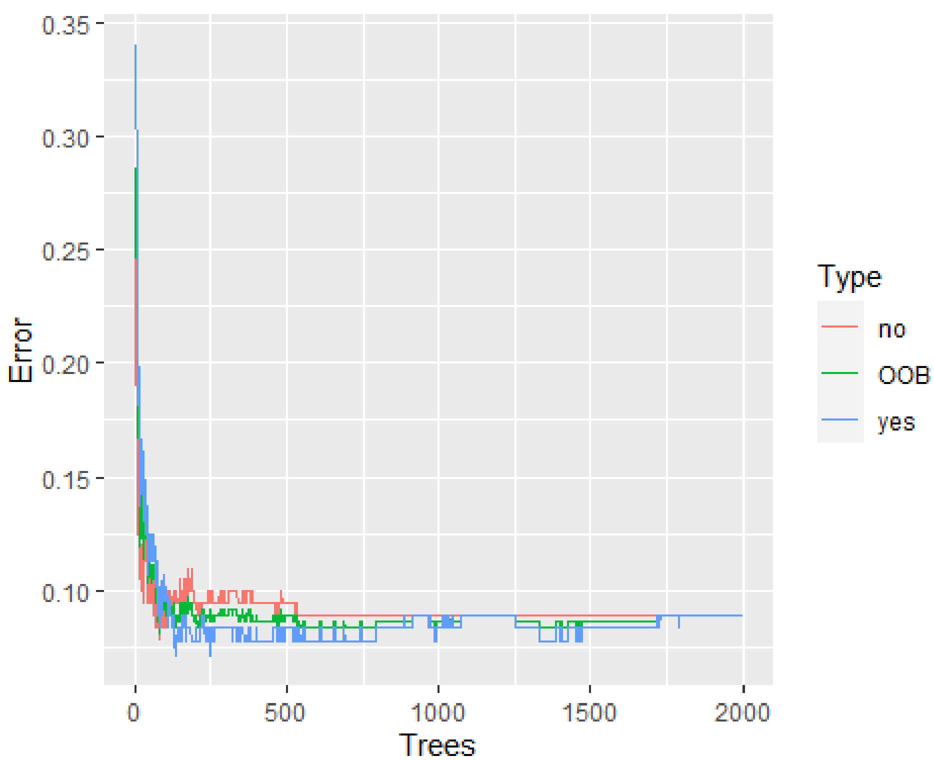
```
##
## Call:
##  randomForest(formula = retention_6m ~ ., data =
clean_bang_retention_6m.train,      proximity = T, ntree = 2000)
##                Type of random forest: classification
##                      Number of trees: 2000
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 8.91%
## Confusion matrix:
##      no yes class.error
## no  174  17  0.08900524
## yes  15 153  0.08928571
```

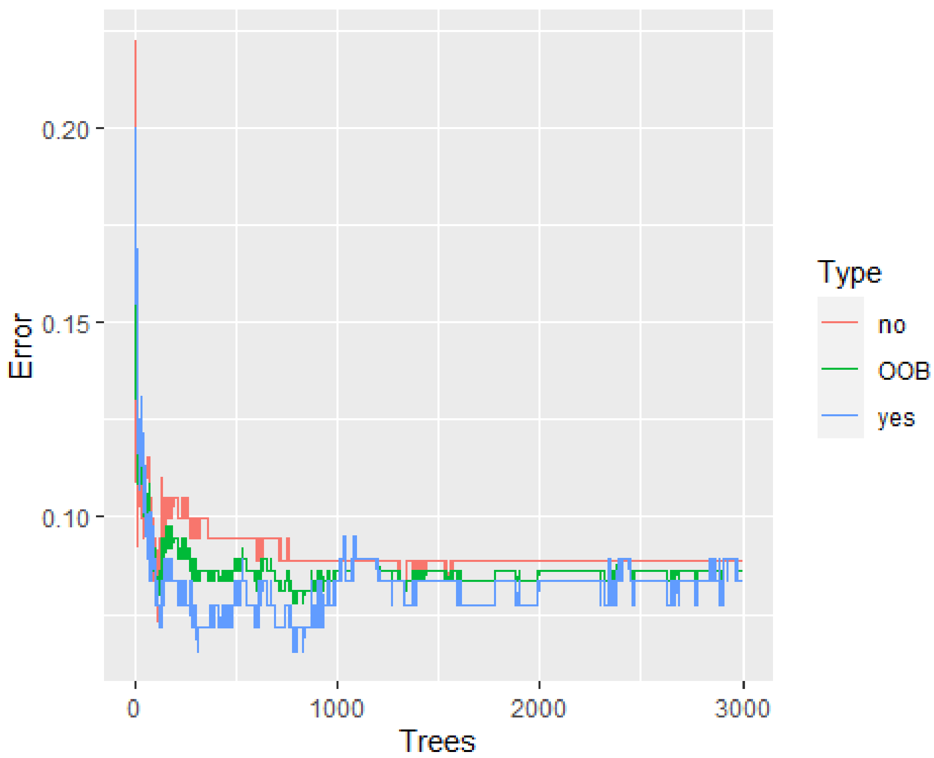
```
training.model.6m_ver3 # 8.64%
```

```
##
## Call:
##  randomForest(formula = retention_6m ~ ., data =
```

```
clean_bang_retention_6m.train,      proximity = T, ntree = 3000)
##                  Type of random forest: classification
##                        Number of trees: 3000
## No. of variables tried at each split: 3
##
##          OOB estimate of  error rate: 8.64%
## Confusion matrix:
##      no yes class.error
## no  174  17  0.08900524
## yes  14 154  0.08333333

oob.error.data.6m_ver1 = data.frame(
  Trees = rep(1:nrow(training.model.6m_ver1$err.rate), times = 3),
  Type = rep(c("OOB", "no", 'yes'), each =
nrow(training.model.6m_ver1$err.rate)),
  Error = c(training.model.6m_ver1$err.rate[, "OOB"],
            training.model.6m_ver1$err.rate[,"no"],
            training.model.6m_ver1$err.rate[, 'yes']))

oob.error.data.6m_ver2 = data.frame(
  Trees = rep(1:nrow(training.model.6m_ver2$err.rate), times = 3),
  Type = rep(c("OOB", "no", 'yes'), each =
nrow(training.model.6m_ver2$err.rate)),
  Error = c(training.model.6m_ver2$err.rate[, "OOB"],
            training.model.6m_ver2$err.rate[,"no"],
            training.model.6m_ver2$err.rate[, 'yes']))

oob.error.data.6m_ver3 = data.frame(
  Trees = rep(1:nrow(training.model.6m_ver3$err.rate), times = 3),
  Type = rep(c("OOB", "no", 'yes'), each =
nrow(training.model.6m_ver3$err.rate)),
  Error = c(training.model.6m_ver3$err.rate[, "OOB"],
            training.model.6m_ver3$err.rate[,"no"],
            training.model.6m_ver3$err.rate[, 'yes']))
```


0.15
0.10
Error
0
250
500
750
1000
Trees
Type
no
OOB
yes


0.35
0.30
0.25
0.20
0.15
0.10
Error
0
500
1000
1500
2000
Trees
Type
no
OOB
yes




```
# Looks like we did a worse job with increasing number of trees, but this
leveled off after 2000.

# STEP 3B: Fine tuning mtry

oob.values <- vector(length = 10)
for(i in 1:10) {
  temp.model <- randomForest(retention_6m ~., data =
clean_bang_retention_6m.train, mtry = i, ntree = 1000)
  oob.values[i] <- temp.model$err.rate[nrow(temp.model$err.rate), 1]
}

oob.values

##  [1] 0.13370474 0.09192201 0.08913649 0.08635097 0.09192201 0.08356546
##  [7] 0.08635097 0.10027855 0.10584958 0.11142061

# Looks like optimal value is 4

proposed.training.model.6m =  randomForest(retention_6m ~., data =
clean_bang_retention_6m.train, proximity = T, mtry = 3, ntree = 1000) #
err.rate = 7.52%

proposed.training.model.6m$confusion
```

```
##      no yes class.error
## no  175  16  0.08376963
## yes  13 155  0.07738095
```

```r
# Step 4: Test this proposed model against testing data

pred_6m_rf <- predict(proposed.training.model.6m, newdata =
clean_bang_retention_6m.test)


confusionMatrix(pred_6m_rf, clean_bang_retention_6m.test$retention_6m) #
accuracy = 0.8977 or err.rate of 10.33%
```

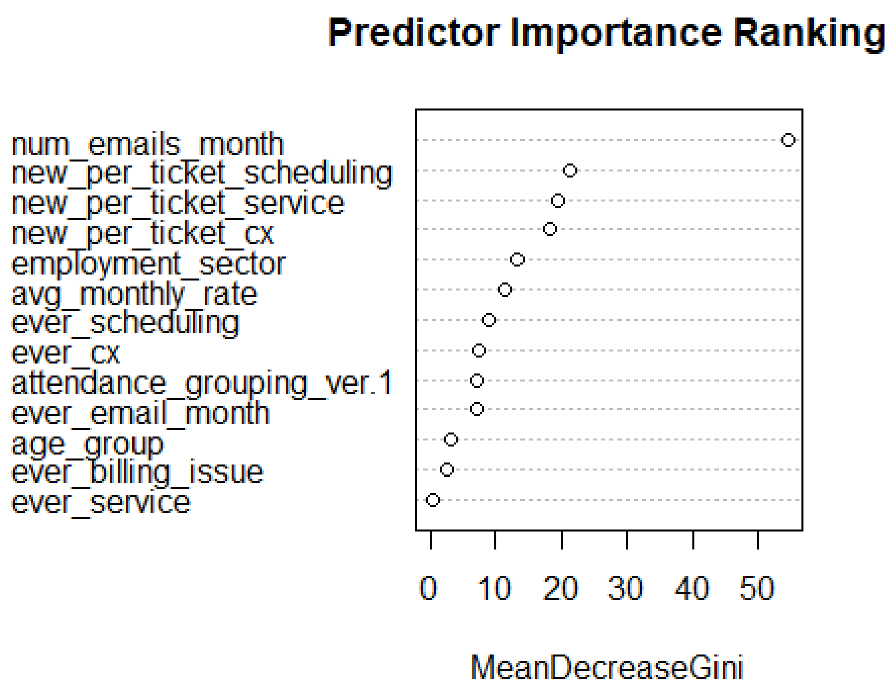
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction no yes
##        no  40   2
##        yes  7  39
##
##                Accuracy : 0.8977
##                  95% CI : (0.8147, 0.9522)
##     No Information Rate : 0.5341
##     P-Value [Acc > NIR] : 2.051e-13
##
##                   Kappa : 0.7961
##
##  Mcnemar's Test P-Value : 0.1824
##
##             Sensitivity : 0.8511
##             Specificity : 0.9512
##          Pos Pred Value : 0.9524
##          Neg Pred Value : 0.8478
##              Prevalence : 0.5341
##          Detection Rate : 0.4545
##    Detection Prevalence : 0.4773
##       Balanced Accuracy : 0.9011
##
##        'Positive' Class : no
##
```

```r
# STEP 5: Determining which variables are important predictors

varImp(proposed.training.model.6m)
```

```
##                              Overall
## age_group                   3.253068
## employment_sector          13.255506
## avg_monthly_rate           11.464820
## attendance_grouping_ver.1   7.061348
## ever_email_month            7.027686
```

```
## num_emails_month           54.578905
## ever_billing_issue          2.362975
## ever_cx                     7.443069
## new_per_ticket_cx          18.160248
## ever_scheduling             8.904537
## new_per_ticket_scheduling 21.260068
## ever_service                0.475466
## new_per_ticket_service    19.306840
```

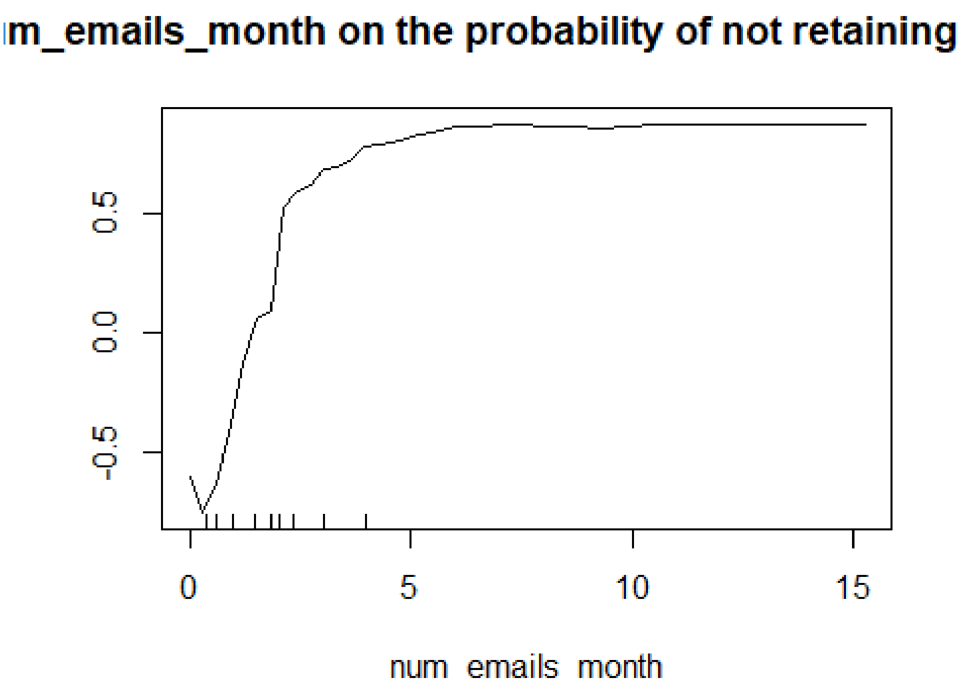
```
varImpPlot(proposed.training.model.6m, sort = T, main = "Predictor Importance
Ranking")
```



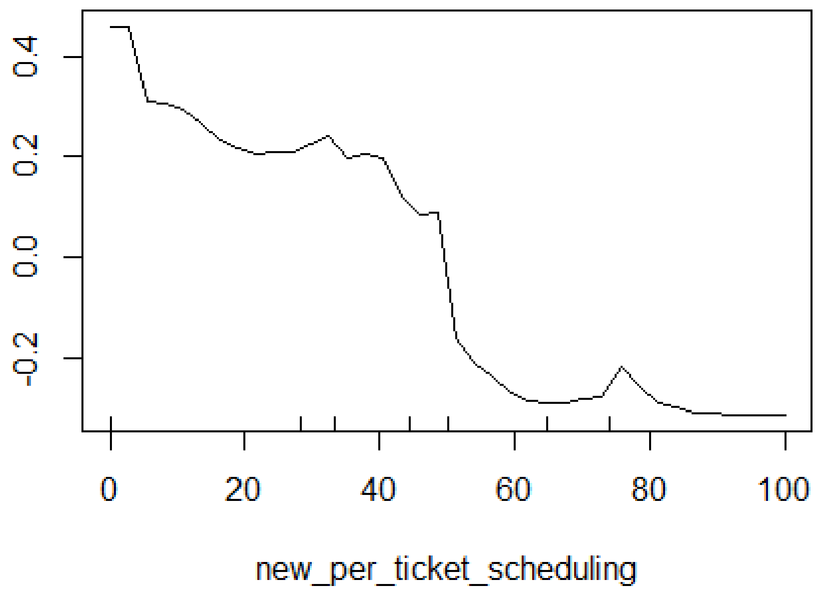

```
importance(proposed.training.model.6m)
```

```
##                            MeanDecreaseGini
## age_group                          3.253068
## employment_sector                 13.255506
## avg_monthly_rate                  11.464820
## attendance_grouping_ver.1          7.061348
## ever_email_month                   7.027686
## num_emails_month                  54.578905
## ever_billing_issue                 2.362975
## ever_cx                            7.443069
## new_per_ticket_cx                 18.160248
## ever_scheduling                    8.904537
## new_per_ticket_scheduling         21.260068
```

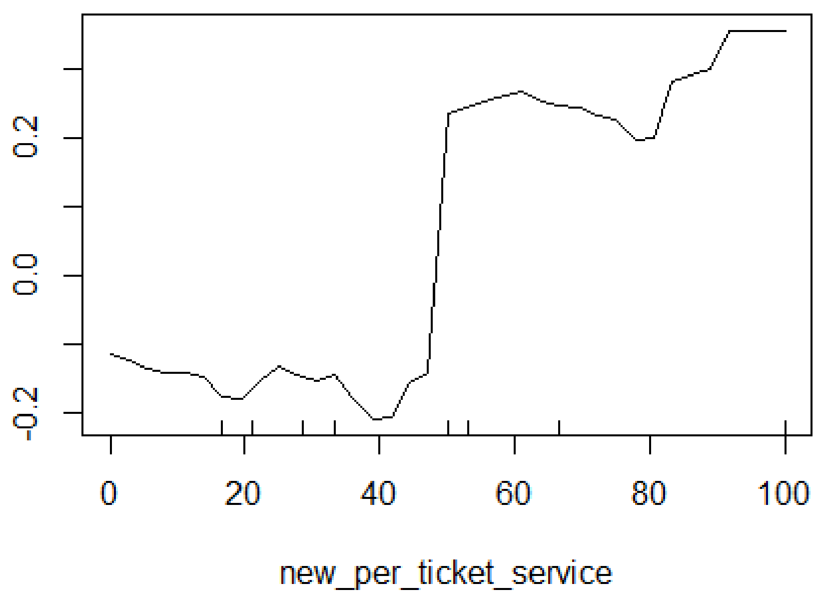
```
## ever_service                          0.475466
## new_per_ticket_service               19.306840

varUsed(proposed.training.model.6m)

##  [1] 2268 4508 4719 3371 1213 6772 1281 1073 3566 1096 4028  236 4237

# Examining the model, it seems that num_emails_month played the most
important role in predicting outcomes follwed by the percent compositions
from each of the non-billing email interactions (scheduling >  service > CX).

# Step 5a: Examining the effects of each variable on retention status (Top 4
predictors)
```

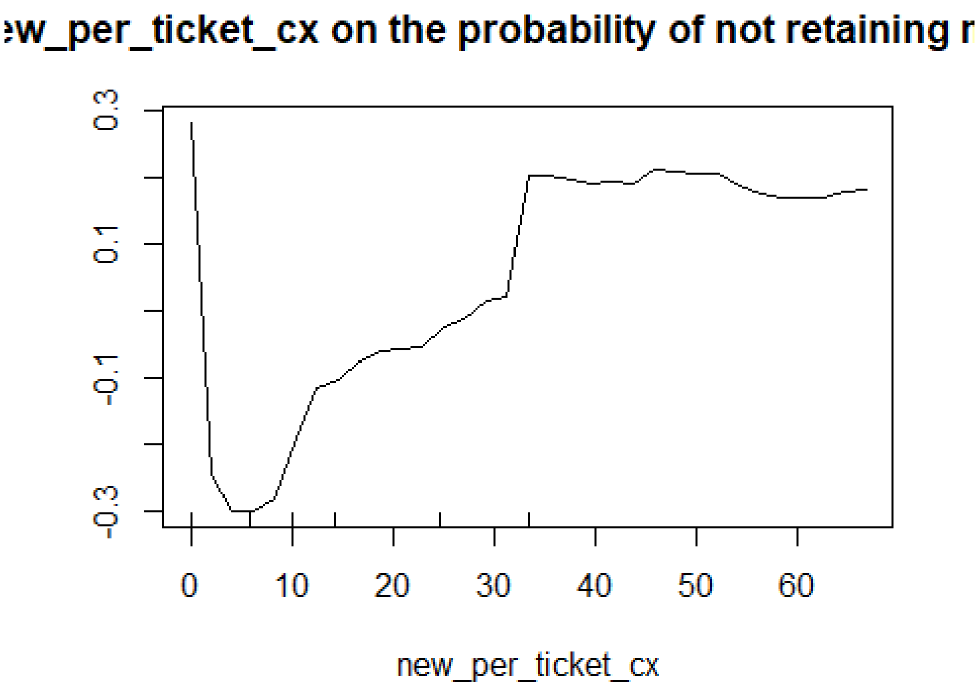


er_ticket_scheduling on the probability of not retain



_per_ticket_service on the probability of not retainin





RETENTION ANALYSIS: Membership status at 12-Months via Random Forest

Using the random survival forest specific data set, I've split the data set 80:20 with respect to training:test. In forming the training model, which has an error rate of **10.89%**, it was found that the error rate in predicting membership length to churn with the test data was **12.36%**. Looking at the various ways to modify the parameters, it was found that the error rate more-or-less stabilized after 1000 trees as evident by the marginal differences in error rates at the higher number of trees. However, in terms of tuning this model, I've adjusted the model to include ntree = 2000 and mtry at 3. Examining the importance of each variable used in this model, it was found that number of non-billing email interaction played the largest role, followed by the percent composition of non-billing related email interactions (scheduling, service and CX).

```
clean_bang_retention_12m = clean_bang_select %>%
  select(
    age_group,
    employment_sector,
    retention_12m,
    avg_monthly_rate,
    attendance_grouping_ver.1,
    ever_email_month,
    num_emails_month,
    ever_billing_issue,
    ever_cx,
    new_per_ticket_cx,
    ever_scheduling,
```

```
    new_per_ticket_scheduling,
    ever_service,
    new_per_ticket_service
  )

# Step 2: create a partition of this data set by splitting it based on
retention status at 12 Months

trainIndex_12m = createDataPartition(clean_bang_retention_12m$retention_12m,
p = 0.8, list = FALSE)
clean_bang_retention_12m.train = clean_bang_retention_12m[trainIndex_12m,]
clean_bang_retention_12m.test = clean_bang_retention_12m[-trainIndex_12m,]

# Step 3: Create a random forest model using training data

training.model.12m = randomForest(retention_12m ~., data =
clean_bang_retention_12m.train, proximity = T)
training.model.12m # OOB error rate is 10.89%

##
## Call:
##  randomForest(formula = retention_12m ~ ., data =
clean_bang_retention_12m.train,      proximity = T)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 10.89%
## Confusion matrix:
##      no yes class.error
## no  216  21  0.08860759
## yes  18 103  0.14876033

# Step 4: Create a data frame to see how the error rate changes as a function
of increasing number of trees (currently capped at 500)

oob.error.data.12m = data.frame(
  Trees = rep(1:nrow(training.model.12m$err.rate), times = 3),
  Type = rep(c("OOB", "no", 'yes'), each =
nrow(training.model.12m$err.rate)),
  Error = c(training.model.12m$err.rate[, "OOB"],
            training.model.12m$err.rate[,"no"],
            training.model.12m$err.rate[, 'yes']))
```


```
# Step 4a: Add more trees and see what happens:

training.model.12m_ver1 = randomForest(retention_12m ~., data =
clean_bang_retention_12m.train, proximity = T, ntree = 1000)
training.model.12m_ver2 = randomForest(retention_12m ~., data =
clean_bang_retention_12m.train, proximity = T, ntree = 2000)
training.model.12m_ver3 = randomForest(retention_12m ~., data =
clean_bang_retention_12m.train, proximity = T, ntree = 3000)

training.model.12m # REFERENCE

##
## Call:
##  randomForest(formula = retention_12m ~ ., data =
clean_bang_retention_12m.train,      proximity = T)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 10.89%
## Confusion matrix:
##      no yes class.error
## no  216  21  0.08860759
## yes  18 103  0.14876033

training.model.12m_ver1 # 11.17%
```

```
##
## Call:
##  randomForest(formula = retention_12m ~ ., data =
clean_bang_retention_12m.train,      proximity = T, ntree = 1000)
##                Type of random forest: classification
##                      Number of trees: 1000
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 11.17%
## Confusion matrix:
##      no yes class.error
## no  215  22   0.0928270
## yes  18 103   0.1487603
```

```
training.model.12m_ver2 # 10.61%
```

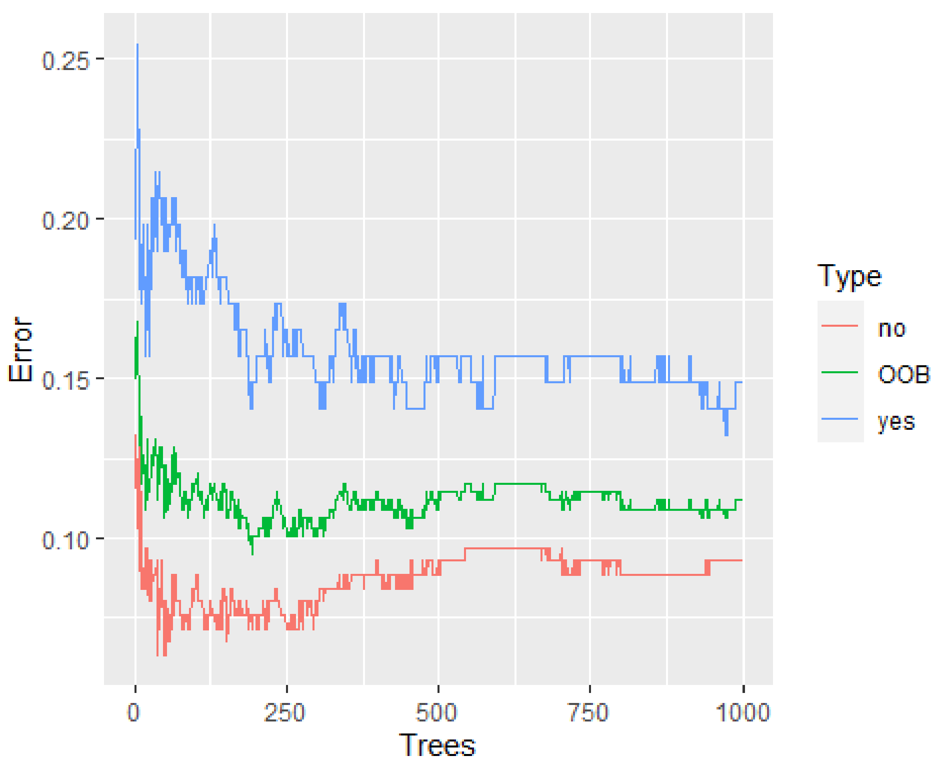
```
##
## Call:
##  randomForest(formula = retention_12m ~ ., data =
clean_bang_retention_12m.train,      proximity = T, ntree = 2000)
##                Type of random forest: classification
##                      Number of trees: 2000
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 10.61%
## Confusion matrix:
##      no yes class.error
## no  216  21  0.08860759
## yes  17 104  0.14049587
```

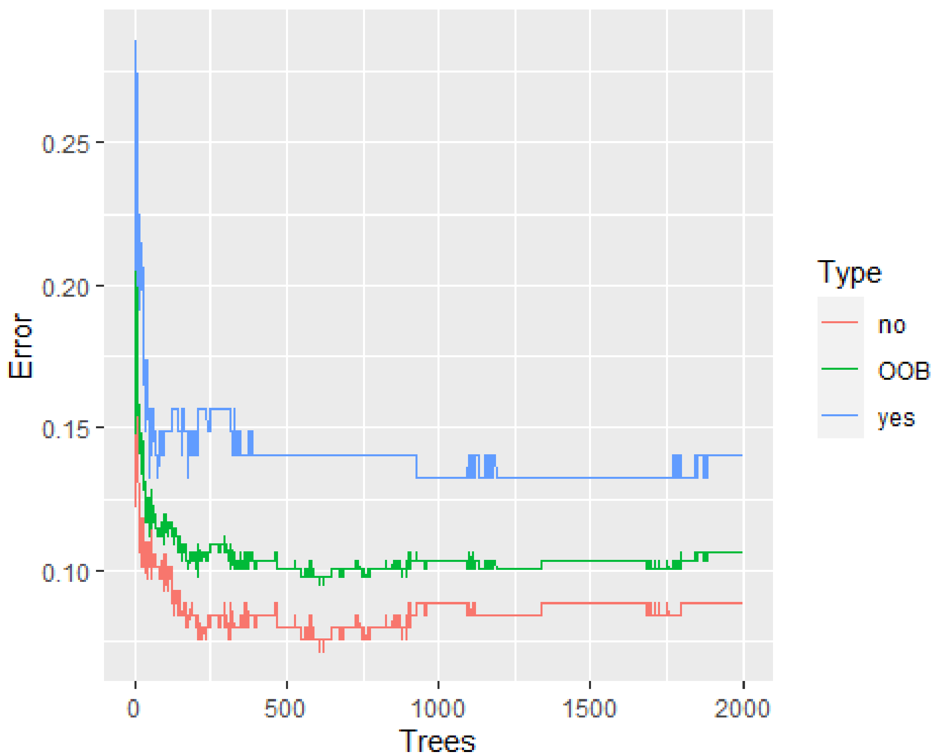
```
training.model.12m_ver3 # 10.61%
```

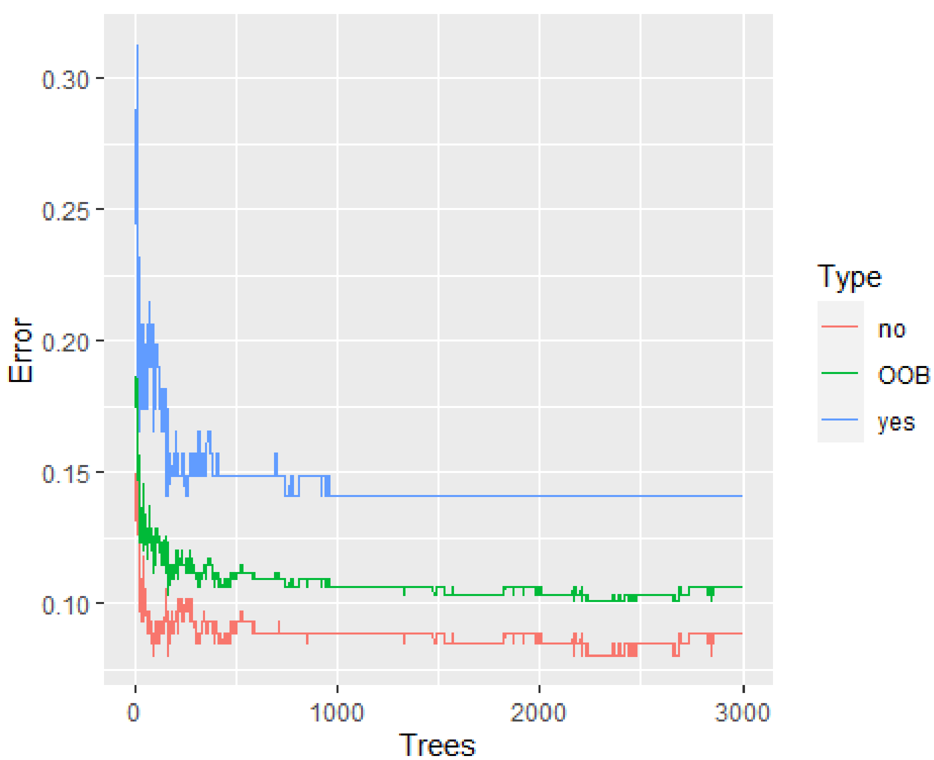
```
##
## Call:
##  randomForest(formula = retention_12m ~ ., data =
clean_bang_retention_12m.train,      proximity = T, ntree = 3000)
##                Type of random forest: classification
##                      Number of trees: 3000
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 10.61%
## Confusion matrix:
##      no yes class.error
## no  216  21  0.08860759
## yes  17 104  0.14049587
```

```
oob.error.data.12m_ver1 = data.frame(
  Trees = rep(1:nrow(training.model.12m_ver1$err.rate), times = 3),
  Type = rep(c("OOB", "no", 'yes'), each =
nrow(training.model.12m_ver1$err.rate)),
  Error = c(training.model.12m_ver1$err.rate[, "OOB"],
```

```
            training.model.12m_ver1$err.rate[,"no"],
            training.model.12m_ver1$err.rate[, 'yes']))

oob.error.data.12m_ver2 = data.frame(
  Trees = rep(1:nrow(training.model.12m_ver2$err.rate), times = 3),
  Type = rep(c("OOB", "no", 'yes'), each =
nrow(training.model.12m_ver2$err.rate)),
  Error = c(training.model.12m_ver2$err.rate[, "OOB"],
            training.model.12m_ver2$err.rate[,"no"],
            training.model.12m_ver2$err.rate[, 'yes']))

oob.error.data.12m_ver3 = data.frame(
  Trees = rep(1:nrow(training.model.12m_ver3$err.rate), times = 3),
  Type = rep(c("OOB", "no", 'yes'), each =
nrow(training.model.12m_ver3$err.rate)),
  Error = c(training.model.12m_ver3$err.rate[, "OOB"],
            training.model.12m_ver3$err.rate[,"no"],
            training.model.12m_ver3$err.rate[, 'yes']))
```








```
# Looks like we did a worse job with increasing number of trees, but this
leveled off after 1000.
```

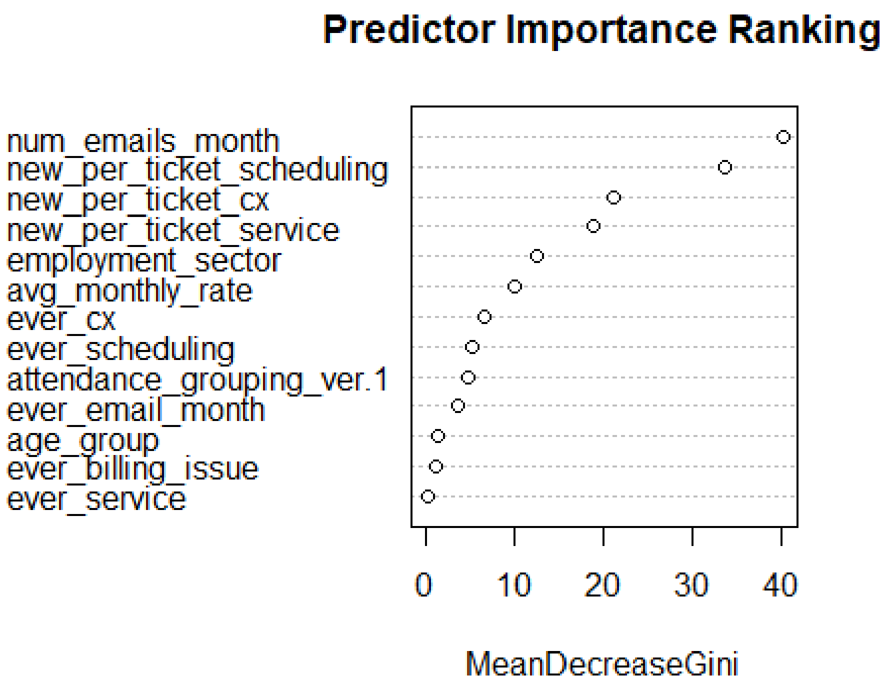
```
# STEP 3B: Fine tuning mtry

oob.values <- vector(length = 10)
for(i in 1:10) {
  temp.model <- randomForest(retention_12m ~., data =
clean_bang_retention_12m.train, mtry = i, ntree = 1000)
  oob.values[i] <- temp.model$err.rate[nrow(temp.model$err.rate), 1]
}

oob.values

##  [1] 0.12849162 0.11452514 0.10893855 0.10614525 0.10893855 0.10893855
##  [7] 0.10335196 0.09776536 0.11173184 0.10055866

# Looks like optimal value is 8

proposed.training.model.12m =  randomForest(retention_12m ~., data =
clean_bang_retention_12m.train, proximity = T, mtry = 8, ntree = 1000) #
err.rate = 13.41%

proposed.training.model.12m$confusion

##      no yes class.error
## no  217  20  0.08438819
## yes  18 103  0.14876033

# Step 4: Test this proposed model against testing data

pred_12m_rf <- predict(proposed.training.model.12m, newdata =
clean_bang_retention_12m.test)


confusionMatrix(pred_12m_rf, clean_bang_retention_12m.test$retention_12m) #
accuracy = 0.8764 or err.rate of 12.36%

## Confusion Matrix and Statistics
##
##           Reference
## Prediction no yes
##        no  57   9
##        yes  2  21
##
##                Accuracy : 0.8764
##                  95% CI : (0.7896, 0.9367)
##     No Information Rate : 0.6629
##     P-Value [Acc > NIR] : 3.758e-06
##
##                   Kappa : 0.7066
##
##  Mcnemar's Test P-Value : 0.07044
##
```

```
##             Sensitivity : 0.9661
##             Specificity : 0.7000
##          Pos Pred Value : 0.8636
##          Neg Pred Value : 0.9130
##              Prevalence : 0.6629
##          Detection Rate : 0.6404
##    Detection Prevalence : 0.7416
##       Balanced Accuracy : 0.8331
##
##        'Positive' Class : no
##
```

```
# STEP 5: Determining which variables are important predictors

varImp(proposed.training.model.12m)
```

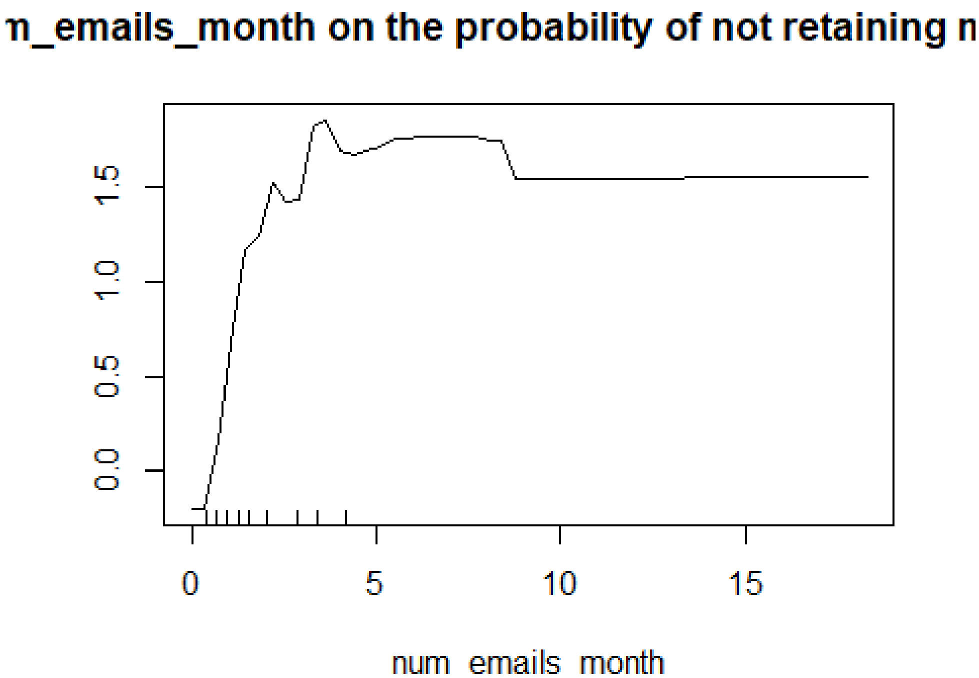
```
##                               Overall
## age_group                   1.4262676
## employment_sector          12.6415438
## avg_monthly_rate           10.0441763
## attendance_grouping_ver.1   4.7512721
## ever_email_month            3.7767475
## num_emails_month           40.3538007
## ever_billing_issue          1.0745443
## ever_cx                     6.6241465
## new_per_ticket_cx          21.1135284
## ever_scheduling             5.1853804
## new_per_ticket_scheduling 33.7148842
## ever_service                0.2897049
## new_per_ticket_service     18.8905452
```

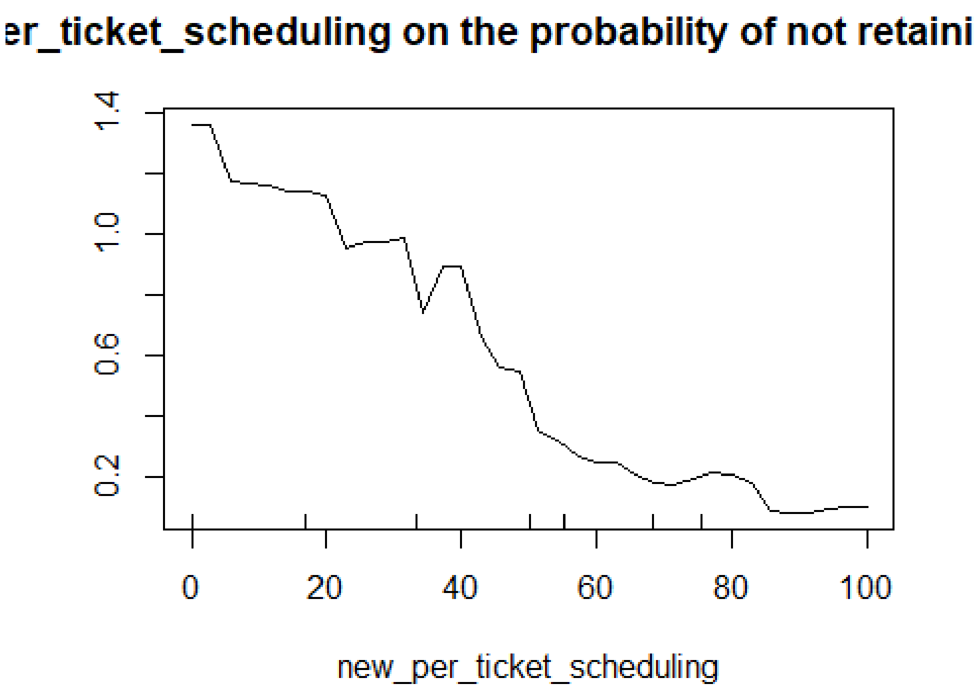
```
varImpPlot(proposed.training.model.12m, sort = T, main = "Predictor
Importance Ranking")
```

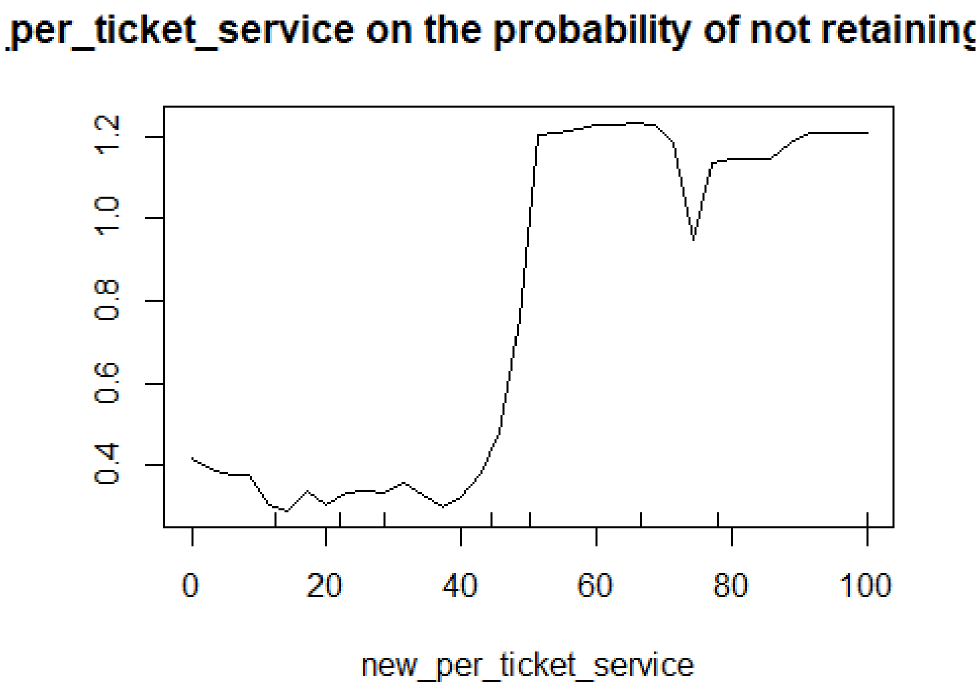


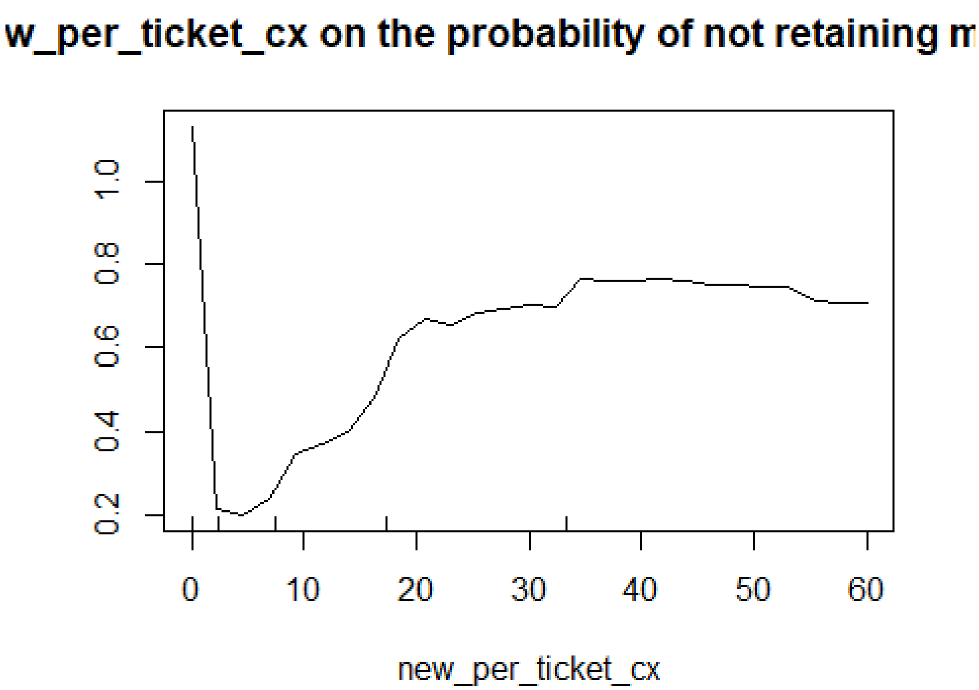

```
importance(proposed.training.model.12m)

##                               MeanDecreaseGini
## age_group                            1.4262676
## employment_sector                   12.6415438
## avg_monthly_rate                    10.0441763
## attendance_grouping_ver.1            4.7512721
## ever_email_month                     3.7767475
## num_emails_month                    40.3538007
## ever_billing_issue                   1.0745443
## ever_cx                              6.6241465
## new_per_ticket_cx                   21.1135284
## ever_scheduling                      5.1853804
## new_per_ticket_scheduling           33.7148842
## ever_service                         0.2897049
## new_per_ticket_service              18.8905452

varUsed(proposed.training.model.12m)

##  [1]  698 3986 3468 1974  461 5499  489  509 3372  435 3791   54 3600

# Examining the model, it seems that num_emails_month played the most
important role in predicting outcomes follwed by the percent compositions
from each of the non-billing email interactions (scheduling > CX > service).

# Step 5a: Examining the effects of each variable on retention status (Top 4
```

```
predictors)
```


n_emails_month on the probability of not retaining n
num_emails_month


er_ticket_scheduling on the probability of not retaini
new_per_ticket_scheduling


.per_ticket_service on the probability of not retaining
1.2
1.0
0.8
0.6
0.4
0
20
40
60
80
100
new_per_ticket_service


w_per_ticket_cx on the probability of not retaining m
1.0
0.8
0.6
0.4
0.2
0
10
20
30
40
50
60
new_per_ticket_cx

RETENTION ANALYSIS: 3 Months via Logistic Regression

In generating a logistic regression model for membership status at 3-month, it was found that the variables that were retained through bi-directional stepwise regression through partitioned data (80-:20) were:

- number of non-billing email interactions per month
- status of ever having a CX-related email interaction
- status of ever having a non-billing email interaction
- percent composition of total interactions being CX-related email interactions
- percent composition of total interactions being scheduling-related email interactions
- status of ever having a service-related email interaction

In cross-validating the proposed model through the validation set approach as well as repeated K-fold validation that the accuracy of the model ranged b/t **90% - 91%**. The major predictors turned out to be num_emails_month, new_per_ticket_scheduling, ever_cx and new_per_ticket_cx.

```
# Step 1: Partition data

trainIndex_3m = createDataPartition(clean_bang_select$retention_3m, p = 0.8,
list = F)

clean_bang_select.3m_train = clean_bang_select[trainIndex_3m,] # This is the
Training Data (80% of the data)
clean_bang_select.3m_test = clean_bang_select[-trainIndex_3m,] # This is the
Testing Data (20% of the data)


# Step 2: Bi-directional Stepwise regression

model.start.train_3m = glm(retention_3m ~ 1, data =
clean_bang_select.3m_train , family = binomial(link = 'logit'))
model.all.train_3m = glm(retention_3m ~ age_group +
                    employment_sector +
                    membership +
                    attendance_grouping_ver.1 +
                    monthly_rate_group +
                    ever_email_month +
                    num_emails_month +
                    ever_cx+
                    new_per_ticket_cx +
                    ever_service+
                    new_per_ticket_service +
```

```
                    ever_scheduling +
                    new_per_ticket_scheduling,
                    data = clean_bang_select.3m_train , family =
binomial(link = 'logit'))

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

step(model.start.train_3m, direction = 'both', scope =
formula(model.all.train_3m))

##
## Call:  glm(formula = retention_3m ~ num_emails_month +
new_per_ticket_scheduling +
##     ever_cx + new_per_ticket_cx + ever_email_month +
new_per_ticket_service,
##     family = binomial(link = "logit"), data = clean_bang_select.3m_train)
##
## Coefficients:
##               (Intercept)           num_emails_month
##                 1.186e-11                 -1.647e+00
## new_per_ticket_scheduling                 ever_cxyes
##                 1.431e+02                  6.812e+00
##         new_per_ticket_cx        ever_email_monthyes
##                 1.429e+02                 -1.430e+04
##    new_per_ticket_service
##                 1.430e+02
##
## Degrees of Freedom: 358 Total (i.e. Null);  352 Residual
## Null Deviance:       494.6
## Residual Deviance: 145.3     AIC: 159.3

model.retained.train_3m = glm(retention_3m ~ num_emails_month +
                                new_per_ticket_scheduling +
                                ever_cx +
                                new_per_ticket_cx +
                                ever_email_month +
                                ever_service,
                                family = binomial(link = "logit"), data =
clean_bang_select.3m_train)

# Step 3: Assessing the proposed model

summary(model.retained.train_3m)

##
## Call:
## glm(formula = retention_3m ~ num_emails_month + new_per_ticket_scheduling
+
##     ever_cx + new_per_ticket_cx + ever_email_month + ever_service,
##     family = binomial(link = "logit"), data = clean_bang_select.3m_train)
##
```

```
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.2514  -0.2642   0.0266   0.1255   3.3670
##
## Coefficients:
##                           Estimate Std. Error z value Pr(>|z|)
## (Intercept)                0.18571    0.68750   0.270  0.78706
## num_emails_month          -1.66858    0.25730  -6.485 8.88e-11 ***
## new_per_ticket_scheduling  0.06047    0.01117   5.412 6.22e-08 ***
## ever_cxyes                 6.30430    1.08245   5.824 5.74e-09 ***
## new_per_ticket_cx         -0.11841    0.02829  -4.185 2.85e-05 ***
## ever_email_monthyes       -4.09016    1.41914  -2.882  0.00395 **
## ever_serviceyes            3.85723    1.39342   2.768  0.00564 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 494.64  on 358  degrees of freedom
## Residual deviance: 148.83  on 352  degrees of freedom
## AIC: 162.83
##
## Number of Fisher Scoring iterations: 8
```

```
exp(cbind(OR = coef(model.retained.train_3m),
confint.default(model.retained.train_3m)))
```

```
##                                     OR         2.5 %       97.5 %
## (Intercept)                 1.20407462  0.312936270    4.6328784
## num_emails_month            0.18851368  0.113848649    0.3121461
## new_per_ticket_scheduling   1.06233668  1.039325821    1.0858570
## ever_cxyes                546.91711064 65.544235859 4563.6099345
## new_per_ticket_cx           0.88833497  0.840418461    0.9389834
## ever_email_monthyes         0.01673656  0.001036799    0.2701705
## ever_serviceyes            47.33411138  3.083872067  726.5275768
```

```
varImp(model.retained.train_3m, sort = T) # Biggest predictors =
num_emails_month, new_per_ticket_scheduling, ever_cx, new_per_ticket_cx
```

```
##                            Overall
## num_emails_month          6.484953
## new_per_ticket_scheduling 5.412243
## ever_cxyes                5.824076
## new_per_ticket_cx         4.185332
## ever_email_monthyes       2.882146
## ever_serviceyes           2.768183
```

```
# Step 4a: validating the proposed model

pred_3m_log <- predict(model.retained.train_3m, newdata =
clean_bang_select.3m_test)
```

```r
pred_3m_log = ifelse(pred_3m_log > 0.5, 'yes', 'no')
table(pred_3m_log, clean_bang_select.3m_test$retention_3m)

##
## pred_3m_log no yes
##         no  36   4
##         yes  4  44

accuracy = table(pred_3m_log, clean_bang_select.3m_test[, "retention_3m"])
accuracy

##
## pred_3m_log no yes
##         no  36   4
##         yes  4  44

sum(diag(accuracy))/sum(accuracy)

## [1] 0.9090909

mean(pred_3m_log == clean_bang_select.3m_test$retention_3m) # 90.9% accuracy
(or err.rate = 9.1%)

## [1] 0.9090909

# Step 4b: repeated k-fold validation

repeat_ctrl <- trainControl(method = "repeatedcv", number = 10, repeats = 3)
proposed.model.retained.3m = train(retention_3m ~ num_emails_month +
                                new_per_ticket_scheduling +
                                ever_cx +
                                new_per_ticket_cx +
                                ever_email_month +
                                ever_service,
                                data = clean_bang_select,
                                method = 'glm',
                                family = 'binomial',
                                trControl = repeat_ctrl, tuneLength = 5)

proposed.model.retained.3m # accuracy = 91.51%

## Generalized Linear Model
##
## 447 samples
##   6 predictor
##   2 classes: 'no', 'yes'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 3 times)
## Summary of sample sizes: 403, 402, 403, 402, 402, 401, ...
## Resampling results:
##
```

```
##   Accuracy   Kappa
##   0.9151354  0.8289641

# Step 4c: k-fold validation

ctrl = trainControl(method = 'cv', number = 10)
proposed.model.retained.3m = train(retention_3m ~ num_emails_month +
                                new_per_ticket_scheduling +
                                ever_cx +
                                new_per_ticket_cx +
                                ever_email_month +
                                ever_service,
                                data = clean_bang_select,
                                method = 'glm',
                                family = 'binomial',
                                trControl = ctrl, tuneLength = 5)

proposed.model.retained.3m # accuracy 91.72%

## Generalized Linear Model
##
## 447 samples
##   6 predictor
##   2 classes: 'no', 'yes'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 402, 402, 403, 402, 403, 403, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.9172661  0.8333126

# Step 5: sumamry of proposed model on original data set

log.model.3m = glm(retention_3m ~ num_emails_month +
                                new_per_ticket_scheduling +
                                ever_cx +
                                new_per_ticket_cx +
                                ever_email_month +
                                ever_service,
                                family = binomial(link = 'logit'), data =
clean_bang_select)

summary(log.model.3m)

##
## Call:
## glm(formula = retention_3m ~ num_emails_month + new_per_ticket_scheduling
+
##     ever_cx + new_per_ticket_cx + ever_email_month + ever_service,
```

```
##     family = binomial(link = "logit"), data = clean_bang_select)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.3234  -0.2408   0.0277   0.1306   3.5831
##
## Coefficients:
##                           Estimate Std. Error z value Pr(>|z|)
## (Intercept)               0.320314   0.650792   0.492 0.622585
## num_emails_month         -1.767266   0.247842  -7.131 9.99e-13 ***
## new_per_ticket_scheduling  0.059788   0.009958   6.004 1.93e-09 ***
## ever_cxyes                6.359700   0.968783   6.565 5.22e-11 ***
## new_per_ticket_cx        -0.116918   0.024910  -4.694 2.68e-06 ***
## ever_email_monthyes      -3.422673   0.951603  -3.597 0.000322 ***
## ever_serviceyes           3.102725   1.046623   2.965 0.003032 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 615.91  on 446  degrees of freedom
## Residual deviance: 180.21  on 440  degrees of freedom
## AIC: 194.21
##
## Number of Fisher Scoring iterations: 8
```

```
exp(cbind(OR = coef(log.model.3m), confint.default(log.model.3m)))
```

```
##                                   OR        2.5 %       97.5 %
## (Intercept)                1.3775596  0.384728921    4.9324870
## num_emails_month           0.1707994  0.105080669    0.2776194
## new_per_ticket_scheduling  1.0616119  1.041092452    1.0825359
## ever_cxyes               578.0730454 86.567190966 3860.2205069
## new_per_ticket_cx          0.8896585  0.847266306    0.9341717
## ever_email_monthyes        0.0326251  0.005052965    0.2106480
## ever_serviceyes           22.2585223  2.861604241  173.1342884
```

```
exp(coef(log.model.3m))
```

```
##               (Intercept)          num_emails_month 
new_per_ticket_scheduling 
##                 1.3775596                 0.1707994 
1.0616119 
##                ever_cxyes         new_per_ticket_cx 
ever_email_monthyes 
##               578.0730454                 0.8896585 
0.0326251 
##           ever_serviceyes 
##                22.2585223
```

```
vif(log.model.3m) # A potential concern of collinearity regarding ever_cx +
new_per_ticket_cx

##           num_emails_month new_per_ticket_scheduling
ever_cxyes
##                   2.139657                  1.686200
5.845623
##          new_per_ticket_cx        ever_email_monthyes
ever_serviceyes
##                   5.362813                  2.637056
2.419064
```

RETENTION ANALYSIS: 6-Months via Logistic Regression

In generating a logistic regression model for membership status at 6-month, it was found that the variables that were retained through bi-directional stepwise regression through partitioned data (80-:20) were

- number of non-billing email interactions per month
- percent composition of scheduling-related email interactions
- status of ever having a non-billing-related email interaction
- status of ever having a CX-related email interaction
- percent composition of service-related email interactions
- percent composition of CX-related email interactions

In cross-validating the proposed model through the validation set approach as well as repeated K-fold validation that the accuracy of the model ranged b/t **84% - 90%**. The major predictors turned out to be num_emails_month, new_per_ticket_scheduling, ever_cx and ever_email_month.

```
# Step 1: Partition data

trainIndex_6m = createDataPartition(clean_bang_select$retention_6m, p = 0.8,
list = F)

clean_bang_select.6m_train = clean_bang_select[trainIndex_6m,] # This is the
Training Data (80% of the data)
clean_bang_select.6m_test = clean_bang_select[-trainIndex_6m,] # This is the
Testing Data (20% of the data)



# Step 2: Bi-directional Stepwise regression

model.start.train_6m = glm(retention_6m ~ 1, data =
clean_bang_select.6m_train, family = binomial(link = 'logit'))
model.all.train_6m = glm(retention_6m ~ age_group +
```

```
                    employment_sector +
                    membership +
                    attendance_grouping_ver.1 +
                    monthly_rate_group +
                    ever_email_month +
                    num_emails_month +
                    ever_cx+
                    new_per_ticket_cx +
                    ever_service+
                    new_per_ticket_service +
                    ever_scheduling +
                    new_per_ticket_scheduling,
                    data = clean_bang_select.6m_train, family = binomial(link
= 'logit'))

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

step(model.start.train_6m, direction = 'both', scope =
formula(model.all.train_6m))
## Call:  glm(formula = retention_6m ~ num_emails_month +
new_per_ticket_scheduling +
##     ever_cx + new_per_ticket_cx + ever_email_month + ever_service,
##     family = binomial(link = "logit"), data = clean_bang_select.6m_train)
##
## Coefficients:
##               (Intercept)           num_emails_month
##                  -0.56632                   -1.81654
## new_per_ticket_scheduling                 ever_cxyes
##                   0.06898                    4.32293
##         new_per_ticket_cx        ever_email_monthyes
##                  -0.05707                   -2.12075
##           ever_serviceyes
##                   1.43947
##
## Degrees of Freedom: 358 Total (i.e. Null);  352 Residual
## Null Deviance:       496.2
## Residual Deviance: 153.5     AIC: 167.5

model.retained.train_6m = glm(retention_6m ~ num_emails_month +
                                new_per_ticket_scheduling +
                                ever_cx +
                                ever_email_month +
                                new_per_ticket_service +
                                new_per_ticket_cx,
                                family = binomial(link = "logit"), data =
clean_bang_select.6m_train)

# Step 3: Assessing the proposed model

summary(model.retained.train_6m)
```

```
##
## Call:
## glm(formula = retention_6m ~ num_emails_month + new_per_ticket_scheduling
+
##     ever_cx + ever_email_month + new_per_ticket_service +
new_per_ticket_cx,
##     family = binomial(link = "logit"), data = clean_bang_select.6m_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.2986  -0.2409  -0.0018   0.2226   4.2011
##
## Coefficients:
##                           Estimate Std. Error z value Pr(>|z|)
## (Intercept)               -0.51104    0.73032  -0.700  0.48408
## num_emails_month          -1.81042    0.29301  -6.179 6.46e-10 ***
## new_per_ticket_scheduling  0.08122    0.01424   5.703 1.17e-08 ***
## ever_cxyes                 4.39892    0.74503   5.904 3.54e-09 ***
## ever_email_monthyes       -2.08283    0.72833  -2.860  0.00424 **
## new_per_ticket_service     0.01385    0.00947   1.462  0.14364
## new_per_ticket_cx         -0.04698    0.02184  -2.151  0.03151 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 496.21  on 358  degrees of freedom
## Residual deviance: 154.25  on 352  degrees of freedom
## AIC: 168.25
##
## Number of Fisher Scoring iterations: 8

AIC(model.retained.train_6m)# 168.251

## [1] 168.2514

exp(cbind(OR = coef(model.retained.train_6m),
confint.default(model.retained.train_6m)))

##                                   OR       2.5 %      97.5 %
## (Intercept)                0.5998701  0.14335428   2.5101735
## num_emails_month           0.1635850  0.09211617   0.2905032
## new_per_ticket_scheduling  1.0846086  1.05475519   1.1153071
## ever_cxyes                81.3631710 18.89093742 350.4307616
## ever_email_monthyes        0.1245772  0.02988686   0.5192745
## new_per_ticket_service     1.0139450  0.99529864   1.0329408
## new_per_ticket_cx          0.9541074  0.91412046   0.9958436

varImp(model.retained.train_6m, sort = T) # Top predictors are:
num_emails_month, new_per_ticket_scheduling, ever_cx, ever_email_month and
new_per_ticket_service
```

```
##                              Overall
## num_emails_month           6.178782
## new_per_ticket_scheduling 5.703464
## ever_cxyes                 5.904322
## ever_email_monthyes        2.859721
## new_per_ticket_service     1.462353
## new_per_ticket_cx          2.150635
```

```
# Step 4a: validating the proposed model

pred_6m_log <- predict(model.retained.train_6m, newdata =
clean_bang_select.6m_test)
pred_6m_log = ifelse(pred_6m_log > 0.5, 'yes', 'no')
table(pred_6m_log, clean_bang_select.6m_test$retention_6m)
```

```
##
## pred_6m_log no yes
##         no  43  10
##         yes  4  31
```

```
accuracy = table(pred_6m_log, clean_bang_select.6m_test[, "retention_6m"])
accuracy
```

```
##
## pred_6m_log no yes
##         no  43  10
##         yes  4  31
```

```
sum(diag(accuracy))/sum(accuracy)
```

```
## [1] 0.8409091
```

```
mean(pred_6m_log == clean_bang_select.6m_test$retention_6m) # 84.09% or
err.rate of 15.91%
```

```
## [1] 0.8409091
```

```
# Step 4b: repeated k-fold validation

repeat_ctrl <- trainControl(method = "repeatedcv", number = 10, repeats = 3)
proposed.model.retained.6m = train(retention_6m ~ num_emails_month +
                                new_per_ticket_scheduling +
                                ever_cx +
                                ever_email_month +
                                new_per_ticket_service +
                                new_per_ticket_cx,
                                data = clean_bang_select,
                                method = 'glm',
                                family = 'binomial',
                                trControl = repeat_ctrl, tuneLength = 5)

proposed.model.retained.6m # accuracy = 90.46%
```

```
## Generalized Linear Model
##
## 447 samples
##   6 predictor
##   2 classes: 'no', 'yes'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 3 times)
## Summary of sample sizes: 403, 402, 403, 402, 402, 402, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.9046449  0.8084935
```

```
# Step 4c: k-fold validation

ctrl = trainControl(method = 'cv', number = 10)
proposed.model.retained.6m = train(retention_6m ~ num_emails_month +
                                   new_per_ticket_scheduling +
                                   ever_cx +
                                   ever_email_month +
                                   new_per_ticket_service +
                                   new_per_ticket_cx,
                                   data = clean_bang_select,
                                   method = 'glm',
                                   family = 'binomial',
                                   trControl = ctrl, tuneLength = 5)

proposed.model.retained.6m # accuracy 90.59%
```

```
## Generalized Linear Model
##
## 447 samples
##   6 predictor
##   2 classes: 'no', 'yes'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 402, 402, 403, 402, 402, 403, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.9059091  0.8110452
```

```
# Step 5: sumamry of proposed model on original data set

log.model.6m = glm(retention_6m ~ num_emails_month +
                                   new_per_ticket_scheduling +
                                   ever_cx +
                                   ever_email_month +
                                   new_per_ticket_service +
```

```r
                                new_per_ticket_cx,
                                family = binomial(link = "logit"),
                                data = clean_bang_select)

summary(log.model.6m)

##
## Call:
## glm(formula = retention_6m ~ num_emails_month + new_per_ticket_scheduling
+
##     ever_cx + ever_email_month + new_per_ticket_service +
new_per_ticket_cx,
##     family = binomial(link = "logit"), data = clean_bang_select)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.9774  -0.2802  -0.0124   0.2499   3.5261
##
## Coefficients:
##                            Estimate Std. Error z value Pr(>|z|)
## (Intercept)               -0.693312   0.707126  -0.980   0.3269
## num_emails_month          -1.392663   0.209425  -6.650 2.93e-11 ***
## new_per_ticket_scheduling  0.084125   0.012671   6.639 3.15e-11 ***
## ever_cxyes                 4.206692   0.614087   6.850 7.37e-12 ***
## ever_email_monthyes       -3.044388   0.648682  -4.693 2.69e-06 ***
## new_per_ticket_service     0.021232   0.008894   2.387   0.0170 *
## new_per_ticket_cx         -0.044074   0.019326  -2.281   0.0226 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 617.79  on 446  degrees of freedom
## Residual deviance: 214.87  on 440  degrees of freedom
## AIC: 228.87
##
## Number of Fisher Scoring iterations: 7

exp(coef(log.model.6m))

##               (Intercept)          num_emails_month
new_per_ticket_scheduling
##                0.49991747                0.24841293
1.08776434
##                ever_cxyes       ever_email_monthyes
new_per_ticket_service
##               67.13410796                0.04762543
1.02145950
##         new_per_ticket_cx
##                0.95688338
```

```
vif(log.model.6m)

##          num_emails_month new_per_ticket_scheduling
ever_cxyes
##                 1.781654                  3.387625
3.042386
##       ever_email_monthyes    new_per_ticket_service
new_per_ticket_cx
##                 1.999301                  1.932293
3.356861

exp(cbind(OR = coef(log.model.6m), confint.default(log.model.6m)))

##                                   OR      2.5 %      97.5 %
## (Intercept)               0.49991747  0.1250234   1.9989653
## num_emails_month          0.24841293  0.1647825   0.3744875
## new_per_ticket_scheduling  1.08776434  1.0610830   1.1151166
## ever_cxyes               67.13410796 20.1479688 223.6944326
## ever_email_monthyes       0.04762543  0.0133561   0.1698236
## new_per_ticket_service    1.02145950  1.0038068   1.0394226
## new_per_ticket_cx         0.95688338  0.9213156   0.9938243
```

RETENTION ANALYSIS: 12M Membership Retention Status

In generating a logistic regression model for membership status at 12-month, it was found that the variables that were retained through bi-directional stepwise regression through partitioned data (80-:20) were

- status of ever having a CX-related email interaction
- percent composition of scheduling-related email interactions
- percent composition of service-related email interactions
- status of ever having a non-billing email interaction
- monthly membership rates
- number of non-billing related email interaction per month.

In cross-validating the proposed model through the validation set approach as well as repeated K-fold validation that the accuracy of the model ranged b/t **90% - 94%**. The major predictors turned out to be ever_cx, new_per_ticket_scheduling, num_emails_month and ever_email_month.

```
# Step 1: Partition data

trainIndex_12m = createDataPartition(clean_bang_select$retention_12m, p =
0.8, list = F)

clean_bang_select.12m_train = clean_bang_select[trainIndex_12m,] # This is
```

```
the Training Data (80% of the data)
clean_bang_select.12m_test = clean_bang_select[-trainIndex_12m,] # This is
the Testing Data (20% of the data)


# Step 2: Bi-directional Stepwise regression

model.start.train_12m = glm(retention_12m ~ 1, data =
clean_bang_select.12m_train, family = binomial(link = 'logit'))
model.all.train_12m = glm(retention_12m ~ age_group +
                    employment_sector +
                    membership +
                    attendance_grouping_ver.1 +
                    monthly_rate_group +
                    ever_email_month +
                    num_emails_month +
                    ever_cx+
                    new_per_ticket_cx +
                    ever_service+
                    new_per_ticket_service +
                    ever_scheduling +
                    new_per_ticket_scheduling,
                    data = clean_bang_select.12m_train, family =
binomial(link = 'logit'))

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

step(model.start.train_12m, direction = 'both', scope =
formula(model.all.train_12m))

## Call:  glm(formula = retention_12m ~ new_per_ticket_scheduling +
num_emails_month +
##     ever_cx + ever_email_month + new_per_ticket_cx + ever_service +
##     age_group + monthly_rate_group, family = binomial(link = "logit"),
##     data = clean_bang_select.12m_train)
##
## Coefficients:
##                 (Intercept)     new_per_ticket_scheduling
##                    -0.75350                       0.07200
##            num_emails_month                   ever_cxyes
##                    -1.00049                       6.09768
##         ever_email_monthyes             new_per_ticket_cx
##                    -5.23329                      -0.07431
##             ever_serviceyes                age_group18-29
##                     2.74153                      -5.43270
##              age_group30-44                age_group45-64
##                    -4.55369                      -4.90912
##               age_group65+   monthly_rate_group100-149.99
##                    -2.32706                     -13.41350
## monthly_rate_group150-199.99  monthly_rate_group200-249.99
```

```
##                        4.76924                         3.51436
## monthly_rate_group250-299.99  monthly_rate_group300-349.99
##                        2.26112                         2.20111
## monthly_rate_group350-399.99  monthly_rate_group400-449.99
##                        2.12920                         1.33212
## monthly_rate_group450-499.99  monthly_rate_group500-549.99
##                       -0.09764                         2.38252
## monthly_rate_group550-599.99        monthly_rate_group600+
##                      -17.13445                        -0.77228
##
## Degrees of Freedom: 357 Total (i.e. Null);  336 Residual
## Null Deviance:       458
## Residual Deviance: 147.3     AIC: 191.3
```

```
model.retained.train_12m = glm(retention_12m ~ new_per_ticket_scheduling +
                                 num_emails_month +
                                 ever_cx +
                                 ever_email_month +
                                 monthly_rate_group +
                                 new_per_ticket_service,
                                family = binomial(link = "logit"),
                               data = clean_bang_select.12m_train)



# Step 3: Assessing the proposed model

summary(model.retained.train_12m)
```

```
##
## Call:
## glm(formula = retention_12m ~ new_per_ticket_scheduling + num_emails_month
+
##     ever_cx + ever_email_month + monthly_rate_group +
new_per_ticket_service,
##     family = binomial(link = "logit"), data = clean_bang_select.12m_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.1077  -0.2056  -0.0457   0.3382   3.2547
##
## Coefficients:
##                                Estimate Std. Error z value Pr(>|z|)
## (Intercept)                    -5.29163    2.20899  -2.396   0.0166 *
## new_per_ticket_scheduling       0.10369    0.01651   6.281 3.37e-10 ***
## num_emails_month               -0.87199    0.18845  -4.627 3.71e-06 ***
## ever_cxyes                      4.38159    0.70550   6.211 5.28e-10 ***
## ever_email_monthyes            -4.39315    0.79284  -5.541 3.01e-08 ***
## monthly_rate_group100-149.99  -14.02040 1946.44247  -0.007   0.9943
## monthly_rate_group150-199.99    3.25067    2.04037   1.593   0.1111
## monthly_rate_group200-249.99    2.54751    1.76262   1.445   0.1484
```

```
## monthly_rate_group250-299.99    1.39061    1.73531   0.801   0.4229
## monthly_rate_group300-349.99    1.44718    1.65584   0.874   0.3821
## monthly_rate_group350-399.99    1.29377    1.65376   0.782   0.4340
## monthly_rate_group400-449.99    0.64073    1.65687   0.387   0.6990
## monthly_rate_group450-499.99   -0.93255    1.99840  -0.467   0.6408
## monthly_rate_group500-549.99    1.03832    2.18781   0.475   0.6351
## monthly_rate_group550-599.99  -17.52737 1458.29768  -0.012   0.9904
## monthly_rate_group600+         -1.27247    6.28458  -0.202   0.8395
## new_per_ticket_service          0.03175    0.01694   1.874   0.0609 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 458.02  on 357  degrees of freedom
## Residual deviance: 165.74  on 341  degrees of freedom
## AIC: 199.74
##
## Number of Fisher Scoring iterations: 17
```

```
exp(cbind(OR = coef(model.retained.train_12m),
confint.default(model.retained.train_12m)))
```

```
##                                       OR        2.5 %       97.5 %
## (Intercept)                 5.033532e-03 6.631021e-05 3.820897e-01
## new_per_ticket_scheduling   1.109253e+00 1.073937e+00 1.145731e+00
## num_emails_month            4.181200e-01 2.889954e-01 6.049381e-01
## ever_cxyes                  7.996525e+01 2.006204e+01 3.187334e+02
## ever_email_monthyes         1.236176e-02 2.613457e-03 5.847167e-02
## monthly_rate_group100-149.99 8.147388e-07 0.000000e+00          Inf
## monthly_rate_group150-199.99 2.580761e+01 4.731338e-01 1.407705e+03
## monthly_rate_group200-249.99 1.277528e+01 4.036677e-01 4.043123e+02
## monthly_rate_group250-299.99 4.017295e+00 1.339176e-01 1.205119e+02
## monthly_rate_group300-349.99 4.251122e+00 1.655958e-01 1.091334e+02
## monthly_rate_group350-399.99 3.646492e+00 1.426232e-01 9.323105e+01
## monthly_rate_group400-449.99 1.897871e+00 7.377939e-02 4.882006e+01
## monthly_rate_group450-499.99 3.935472e-01 7.833437e-03 1.977158e+01
## monthly_rate_group500-549.99 2.824481e+00 3.878557e-02 2.056872e+02
## monthly_rate_group550-599.99 2.443197e-08 0.000000e+00          Inf
## monthly_rate_group600+       2.801390e-01 1.252933e-06 6.263533e+04
## new_per_ticket_service       1.032262e+00 9.985456e-01 1.067116e+00
```

```
varImp(model.retained.train_12m, sort = T) # Top predictors
are:new_per_ticket_scheduling, num_emails_month, ever_email_month and ever_cx
```

```
##                                  Overall
## new_per_ticket_scheduling    6.280943734
## num_emails_month             4.627121100
## ever_cxyes                   6.210582562
## ever_email_monthyes          5.541038990
## monthly_rate_group100-149.99 0.007203089
```

```
## monthly_rate_group150-199.99 1.593178543
## monthly_rate_group200-249.99 1.445296949
## monthly_rate_group250-299.99 0.801361775
## monthly_rate_group300-349.99 0.873986697
## monthly_rate_group350-399.99 0.782316463
## monthly_rate_group400-449.99 0.386712374
## monthly_rate_group450-499.99 0.466649581
## monthly_rate_group500-549.99 0.474595140
## monthly_rate_group550-599.99 0.012019064
## monthly_rate_group600+       0.202474807
## new_per_ticket_service       1.874062773

# Step 4: validating the proposed model

pred_12m_log <- predict(model.retained.train_12m, newdata =
clean_bang_select.12m_test)
pred_12m_log = ifelse(pred_12m_log > 0.5, 'yes', 'no')
table(pred_12m_log, clean_bang_select.12m_test$retention_12m)

##
## pred_12m_log no yes
##          no  58   4
##          yes  1  26

accuracy = table(pred_12m_log, clean_bang_select.12m_test[, "retention_12m"])
accuracy

##
## pred_12m_log no yes
##          no  58   4
##          yes  1  26

sum(diag(accuracy))/sum(accuracy)

## [1] 0.9438202

mean(pred_12m_log == clean_bang_select.12m_test$retention_12m) # 94.38% or
err.rate of 5.62%

## [1] 0.9438202

#Step 4b: repeated k-fold validation

repeat_ctrl <- trainControl(method = "repeatedcv", number = 10, repeats = 3)
proposed.model.retained.12m = train(retention_12m ~ new_per_ticket_scheduling
+
                                 num_emails_month +
                                 ever_cx +
                                 ever_email_month +
                                 monthly_rate_group +
                                 new_per_ticket_service,
                                data = clean_bang_select,
```

```r
                                method = 'glm',
                                family = 'binomial',
                                trControl = repeat_ctrl, tuneLength = 5)

proposed.model.retained.12m # accuracy = 90.45%

## Generalized Linear Model
##
## 447 samples
##   6 predictor
##   2 classes: 'no', 'yes'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 3 times)
## Summary of sample sizes: 403, 403, 402, 402, 402, 402, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.9045286  0.7862865

# Step 4c: k-fold validation

ctrl = trainControl(method = 'cv', number = 10)
proposed.model.retained.12m = train(retention_12m ~ new_per_ticket_scheduling
+
                                 num_emails_month +
                                 ever_cx +
                                 ever_email_month +
                                 monthly_rate_group +
                                 new_per_ticket_service,
                                data = clean_bang_select,
                                method = 'glm',
                                family = 'binomial',
                                trControl = ctrl, tuneLength = 5)

proposed.model.retained.12m # accuracy 90.38%

## Generalized Linear Model
##
## 447 samples
##   6 predictor
##   2 classes: 'no', 'yes'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 403, 402, 402, 402, 402, 403, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.9037879  0.7839525
```

```r
# Step 5: sumamry of proposed model on original data set

log.model.12m = glm(retention_12m ~ new_per_ticket_scheduling +
                                 num_emails_month +
                                 ever_cx +
                                 ever_email_month +
                                 monthly_rate_group +
                                 new_per_ticket_service,
                                family = binomial(link = "logit"),
                                data = clean_bang_select)

summary(log.model.12m)

##
## Call:
## glm(formula = retention_12m ~ new_per_ticket_scheduling + num_emails_month
+
##     ever_cx + ever_email_month + monthly_rate_group +
new_per_ticket_service,
##     family = binomial(link = "logit"), data = clean_bang_select)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.0834  -0.2069  -0.0405   0.3074   3.2094
##
## Coefficients:
##                                Estimate Std. Error z value Pr(>|z|)
## (Intercept)                    -5.14442    2.13520  -2.409   0.0160 *
## new_per_ticket_scheduling       0.10195    0.01464   6.963 3.34e-12 ***
## num_emails_month               -1.03940    0.20950  -4.961 7.00e-07 ***
## ever_cxyes                      4.61203    0.65038   7.091 1.33e-12 ***
## ever_email_monthyes            -4.23238    0.74469  -5.683 1.32e-08 ***
## monthly_rate_group100-149.99    1.04721    2.31017   0.453   0.6503
## monthly_rate_group150-199.99    3.14763    1.98533   1.585   0.1129
## monthly_rate_group200-249.99    2.45729    1.76467   1.392   0.1638
## monthly_rate_group250-299.99    1.42599    1.77426   0.804   0.4216
## monthly_rate_group300-349.99    1.50211    1.69796   0.885   0.3763
## monthly_rate_group350-399.99    1.44543    1.68461   0.858   0.3909
## monthly_rate_group400-449.99    0.67973    1.69283   0.402   0.6880
## monthly_rate_group450-499.99   -1.04489    2.02148  -0.517   0.6052
## monthly_rate_group500-549.99    0.66005    2.11065   0.313   0.7545
## monthly_rate_group550-599.99  -17.37800 1263.34754  -0.014   0.9890
## monthly_rate_group600+         -2.17309    4.52303  -0.480   0.6309
## new_per_ticket_service          0.02884    0.01515   1.904   0.0569 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 571.78  on 446  degrees of freedom
```

```
## Residual deviance: 194.40  on 430  degrees of freedom
## AIC: 228.4
##
## Number of Fisher Scoring iterations: 17
```

```
exp(coef(log.model.12m))
```

```
##                   (Intercept)    new_per_ticket_scheduling
##                  5.831828e-03                 1.107330e+00
##               num_emails_month                  ever_cxyes
##                  3.536678e-01                 1.006879e+02
##           ever_email_monthyes monthly_rate_group100-149.99
##                  1.451780e-02                 2.849694e+00
## monthly_rate_group150-199.99 monthly_rate_group200-249.99
##                  2.328082e+01                 1.167309e+01
## monthly_rate_group250-299.99 monthly_rate_group300-349.99
##                  4.161993e+00                 4.491163e+00
## monthly_rate_group350-399.99 monthly_rate_group400-449.99
##                  4.243667e+00                 1.973345e+00
## monthly_rate_group450-499.99 monthly_rate_group500-549.99
##                  3.517288e-01                 1.934892e+00
## monthly_rate_group550-599.99       monthly_rate_group600+
##                  2.836799e-08                 1.138254e-01
##        new_per_ticket_service
##                  1.029261e+00
```

```
vif(log.model.12m)
```

```
##    new_per_ticket_scheduling             num_emails_month
##                     4.073049                     1.576013
##                   ever_cxyes          ever_email_monthyes
##                     2.457218                     2.308606
## monthly_rate_group100-149.99 monthly_rate_group150-199.99
##                     2.061469                     3.774348
## monthly_rate_group200-249.99 monthly_rate_group250-299.99
##                     8.014555                     6.860786
## monthly_rate_group300-349.99 monthly_rate_group350-399.99
##                    13.992423                    16.815340
## monthly_rate_group400-449.99 monthly_rate_group450-499.99
##                    14.532071                     2.944180
## monthly_rate_group500-549.99 monthly_rate_group550-599.99
##                     2.605817                     1.000002
##       monthly_rate_group600+       new_per_ticket_service
##                     1.164648                     3.903893
```

```
exp(cbind(OR = coef(log.model.12m), confint.default(log.model.12m)))
```

```
##                                        OR        2.5 %       97.5 %
## (Intercept)                  5.831828e-03 8.878077e-05 3.830809e-01
## new_per_ticket_scheduling    1.107330e+00 1.076003e+00 1.139569e+00
## num_emails_month             3.536678e-01 2.345684e-01 5.332386e-01
```

```
## ever_cxyes                     1.006879e+02 2.814326e+01 3.602304e+02
## ever_email_monthyes            1.451780e-02 3.372995e-03 6.248642e-02
## monthly_rate_group100-149.99 2.849694e+00 3.078777e-02 2.637657e+02
## monthly_rate_group150-199.99 2.328082e+01 4.754237e-01 1.140028e+03
## monthly_rate_group200-249.99 1.167309e+01 3.673659e-01 3.709139e+02
## monthly_rate_group250-299.99 4.161993e+00 1.285427e-01 1.347582e+02
## monthly_rate_group300-349.99 4.491163e+00 1.610834e-01 1.252181e+02
## monthly_rate_group350-399.99 4.243667e+00 1.562433e-01 1.152607e+02
## monthly_rate_group400-449.99 1.973345e+00 7.149334e-02 5.446790e+01
## monthly_rate_group450-499.99 3.517288e-01 6.691511e-03 1.848808e+01
## monthly_rate_group500-549.99 1.934892e+00 3.090777e-02 1.211284e+02
## monthly_rate_group550-599.99 2.836799e-08 0.000000e+00          Inf
## monthly_rate_group600+         1.138254e-01 1.607776e-05 8.058478e+02
## new_per_ticket_service         1.029261e+00 9.991531e-01 1.060277e+00
```

DISCUSSION

During this analysis, several notable results were shown that can be taken in our approach to ensuring membership retention.

1) The most commonly noted reason for membership churn being related to finance + lack of accessibility/availability. There were several findings that seemed to support this such as:

   - The majority of our clientele being in scheduling demanding fields: technology, advertising/media and finance.
   - Majority having an attendance rate of less than 70%
   - Approx. 1/3 of total email interaction with staff pertaining to scheduling or rescheduling requests.

2) While the median membership length was approximately 4.5 months, this differed across membership type and demographics

   - Technology sector having one of the longest membership durations as compared to others VS. government/social services + retail/accommodation/hospitality having the lowest
   - 30-44 demographic had the longest membership length out of all age groups
   - 2x/week membership > 3x/week membership in terms of length. Interestingly, those that have unlimited number of Hybrid sessions also tend to have a longer membership retention
   - Monthly rates b/t 300-399 appear to be "sweet spot" in terms of length of membership with noticeable degradation in membership length with rates beyond $400

3) The impact of the COVID pandemic has significantly reduced the clientele by 31% from our initial shutdown. This was cited as the 3rd most common reason for membership loss over the last two years.

4) The importance of customer interactions was also noted.

Namely, it was found that an increase in the number of non-billing email interaction per month was associated with an increased odds of membership churn. However in terms of type of email interaction, having even a single CX-related email interaction outside of the onboarding process was found to significantly increased the odds of membership retention at 3-, 6- and 12-months. Although significant, the impact of an increasing the number of CX-related email interaction had a small decrease in the odds of membership churn.

In terms of improving retention status, efforts should be made with respect to minimizing the drop in attendance seeing as we are not going to be changing our business model anytime soon. Habit-focus development appears to be a key area to dive into to address this issue. Namely the adoption of a “fallback” option could be introduced and highlighted early on during the on-boarding transition phase highlighting strategies to prepare and implement for situations of inevitable scheduling disruptions. From the membership-service end, a work flow could be developed to track and assess attendance across several time points over a 90 day span to see what really is the best membership is for a new member. While the default option has been the push to 3x/week, this option has been shown to fail to sustain membership retention over 3-months as compared to the 2x/week option. This is possibly due to the sense of inadequate return in value considering the noticeable price point, along with some potential difficulty to attend enough sessions to justify said price point.

Possible solutions can be:

(1) revision into pricing –> considering that most current members were under older or modified pricing as compared to the updated membership rates

(2) reframing return of value based on attendance or provide lower cost additions to justify membership price point (i.e. increased group class schedule, nutritional-habit coaching, etc.)

(3) early membership service team intervention to handle finding out the right membership type based on previous attendance rates –> similar to those wine/snackbox subscription where we will find the best solution for the member for their value = improve CX –> this can be stratified based on certain demographics

(4) Improving on schedule availability could be an option