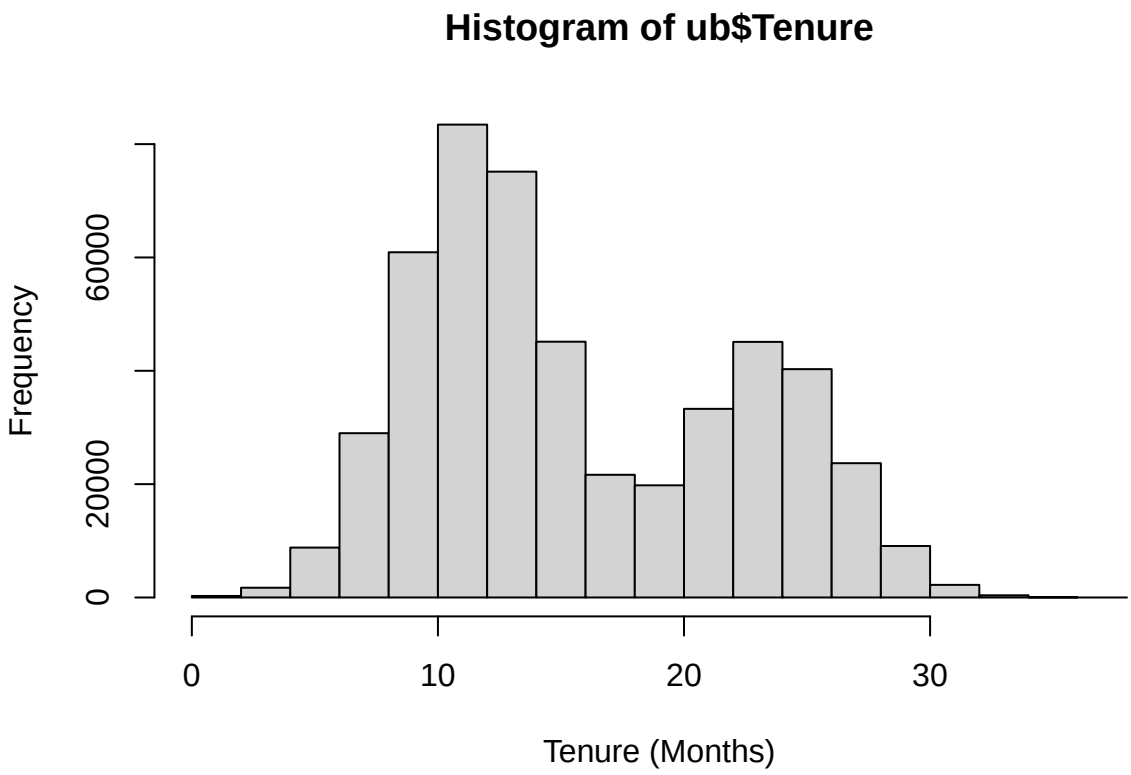
DStest

Q1)

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   18.00   27.00   31.00   31.62   36.00   63.00
```

Q2) The histogram has a bimodal shape (there are two peaks). Tenure ranges from 0-38 months. Tenures of 15-19 months are less common, and the vast majority of customers have either 10-14 months of tenure, or 20-25 months of tenure.

```
hist(ub$Tenure, xlab = 'Tenure (Months)')
```



Q3) No, the average tenure is ~16 months with 95% confidence. Since the sample size is very large, we can apply the Central Limit Theorem. Therefore, I am assuming sample mean is ~normally distributed.

```
#calculate mean, standard deviation, standard error of mean
t.mean = mean(ub$Tenure)
t.size = length(ub$Tenure)
t.sd = sd(ub$Tenure)
t.se = t.sd/sqrt(t.size)
print(t.se)
```

```
## [1] 0.009146445
```

```
#calculate t-score, use alpha of 0.05 since we want 95% confidence
alpha = 0.05
degrees.freedom = t.size - 1
t.score = qt(p = alpha/2, df=degrees.freedom, lower.tail=F)
print(t.score)
```

```
## [1] 1.959969
```

```
#calculate margin of error, use it to calculate upper and lower bounds
margin.error = t.score * t.se
t.lower = t.mean - margin.error
t.upper = t.mean + margin.error
print(c(t.lower, t.upper))
```

```
## [1] 16.17489 16.21074
```

```
#simple one sample t-test to confirm results
t.test(ub$Tenure)
```

```
##
##  One Sample t-test
##
## data:  ub$Tenure
## t = 1770.4, df = 5e+05, p-value < 2.2e-16
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  16.17489 16.21074
## sample estimates:
## mean of x
##  16.19282
```

Q4)

```
## # A tibble: 4 x 4
## # Groups:   Gender [2]
##   Gender Type  `Average Age` `Proportion of Customers`
##   <chr>  <chr>         <dbl>                     <dbl>
## 1 F      B              38.0                    0.0696
## 2 F      C              30.0                    0.280
## 3 M      B              38.0                    0.130
## 4 M      C              30.0                    0.520
```

Q5)

```
cbye = ub %>% select(Email_Address) %>%
  group_by('Email Domain' = str_extract(Email_Address,'(?<=@)(\\w+)')) %>%
  select('Email Domain') %>%
  summarize('Number of Customers' = n())
#check that number of customers is what we expect (500k)
head(cbye)
```

```
## # A tibble: 6 x 2
##   `Email Domain` `Number of Customers`
##   <chr>                          <int>
## 1 aol                            49975
## 2 comcast                        34820
## 3 gmail                         150266
```

```
## 4 hotmail                       125482
## 5 msn                            74626
## 6 yahoo                          64831
```

Q6)

```
sent = read.csv('test_dataset/Sent_Table.csv', stringsAsFactors = FALSE)
head(sent)
```

```
##    Sent_Date Customer_ID SubjectLine_ID
## 1 2016-01-28        1413              2
## 2 2016-03-02       83889              2
## 3 2016-03-09      457832              3
## 4 2016-01-20      127772              1
## 5 2016-02-03      192123              3
## 6 2016-02-07      399506              2
```

```
s.wkdays = data.frame(Sent_Date = as.Date(sent$Sent_Date))
s.wkdays$Weekdays = weekdays(s.wkdays$Sent_Date)

#calculate occurrence of each weekday in data
nperwkday = s.wkdays %>% group_by(Sent_Date) %>%
  summarize(nperday = n()) %>%
  mutate(Weekdays = weekdays((Sent_Date)))
cperwkday = nperwkday %>% count(Weekdays)

#calculate total number of emails sent by day
sentbyday = s.wkdays %>% select(Weekdays) %>%
  group_by(Weekdays) %>% summarize('npbyday' = n())

#join weekday occurrence with dataframe containing total emails by day,
#calculate average number of emails sent by day
sentbyday = sentbyday %>%
  left_join(cperwkday, by=c('Weekdays')) %>%
  mutate('Average Number of Emails Sent' = npbyday/n) %>%
  select(Weekdays, 'Average Number of Emails Sent')
head(sentbyday)
```

```
## # A tibble: 6 x 2
##   Weekdays `Average Number of Emails Sent`
##   <chr>                             <dbl>
## 1 Friday                            28424
## 2 Monday                            28460.
## 3 Saturday                          28456.
## 4 Sunday                            28483.
## 5 Thursday                          28570.
## 6 Tuesday                           28468.
```

Q7) In this problem, I used one-way ANOVA to test for any statistical differences between the 3 groups. The resulting p-value of 0.3 is greater than 0.05. Therefore, we can't reject the null hypothesis (no difference between subjectline ID's) Thus, there are no significant differences between the SubjectLine_IDs. That being said, SubjectLine_ID 3 seems to be underperforming with an 8.54% open rate.

```
resp = read.csv('test_dataset/Responded_Table.csv')

#calculate valid responses by subjectline ID
rns = inner_join(resp, sent, by=c('Customer_ID', 'SubjectLine_ID'))
```

```
vbyid = rns %>% filter(Responded_Date == Sent_Date) %>%
  group_by(SubjectLine_ID) %>%
  summarize(totvalidresponses = n())

#calculate total emails sent by subject line ID
subjsent = sent %>% count(SubjectLine_ID)

#join valid responses with total emails sent, calculate average open rate by subject line ID
openrbyid = inner_join(vbyid, subjsent, by=c('SubjectLine_ID')) %>%
  mutate(AvgOpenRate = totvalidresponses/n)
head(openrbyid)
```

```
## # A tibble: 3 x 4
##   SubjectLine_ID totvalidresponses      n AvgOpenRate
##            <int>             <int>  <int>       <dbl>
## 1              1             79677 826717      0.0964
## 2              2             78967 824837      0.0957
## 3              3             70466 824800      0.0854
```

```
#SubjectLine_ID 3 seems to be underperforming, with an 8.54% open rate.

#model AvgOpenRate as a function of the subjectline_ID
#use one-way ANOVA
one.way = aov(openrbyid$AvgOpenRate ~ openrbyid$SubjectLine_ID, data = openrbyid)
summary(one.way)
```

```
##                            Df    Sum Sq   Mean Sq F value Pr(>F)
## openrbyid$SubjectLine_ID  1 5.988e-05 5.988e-05   3.849    0.3
## Residuals                  1 1.556e-05 1.556e-05
```

Q8) We are seeing a 10.2% average open rate for businesses, and a 9% average open rate for consumers.

```
sub = sent %>% left_join(ub, by=c('Customer_ID'))
atype = sub %>%
  inner_join(resp, by=c('Customer_ID', 'SubjectLine_ID','Sent_Date' = 'Responded_Date'))
#calculate number of valid emails by type
cbytype = atype %>% count(Type)

#calculate number of emails sent by type
sentbytype = sub %>% group_by(Type) %>% summarize(sent.type = n())

avgbytype = cbytype %>% left_join(sentbytype, by=c('Type')) %>% mutate(type.AvgOpen = n/sent.type)
print(avgbytype)
```

```
##   Type      n sent.type type.AvgOpen
## 1    B  50233    494705   0.10154132
## 2    C 178877   1981649   0.09026674
```

Q9) For this problem, since we are predicting the open rate based on the customer attributes and subject line ID received, I use logistic since regression generates values between 0 and 1 that can be interpreted as a percentage. First, I created an indicator variable to show whether the email was opened. Then, I cleaned up the data and dropped duplicates and unnecessary columns. After, I split the data into training and test data in a ratio of 70% training data, 30% test/validation data. With the customer parameters provided, this model predicts an open rate of 8.4%.

```
#create indicator variable opened
ind = rns %>% filter(Responded_Date == Sent_Date) %>% mutate(opened = 1)
#
oprate = sent %>% left_join(ind, by=c('Sent_Date','Customer_ID','SubjectLine_ID')) %>%
  left_join(ub, by=c('Customer_ID'))
#drop duplicate rows, unnecessary columns
oprate = oprate[!duplicated(oprate[,1:3]),]
oprate = oprate %>% select(-c(Sent_Date,Responded_Date,Customer_ID)) %>% replace(is.na(.),0)
oprate$Email_Address = str_extract(oprate$Email_Address,'(?<=@)(\\w+)')
cols = c('SubjectLine_ID', 'Gender','Type','Email_Address')
oprate[cols] = lapply(oprate[cols], factor)

#divide data into 70% training and 30% test/validation data
m = nrow(oprate)
trn = sample(1:m, size=round(m*0.7), replace=FALSE)
train = oprate[trn,]
valid = oprate[-trn,]


o.glm  = glm(opened ~ SubjectLine_ID + Gender + Type + Email_Address + Age + Tenure, data =train, family
#create new dataframe with parameters given
newdata = data.frame(Gender='F',Type='B',Email_Address='aol', Age = 50, Tenure = 12, SubjectLine_ID = '
head(newdata)
```

```
##   Gender Type Email_Address Age Tenure SubjectLine_ID
## 1      F    B           aol  50     12              3
```

```
#use model created above to predict open rate of customer = 8.4%
predict(o.glm, newdata, type='response')
```

```
##          1
## 0.08446998
```

When testing the model, I chose a threshold of 10%, which resulted in an accuracy of 72%. However, the ROC curve and AUC calculation shows that our model is not very good at predicting the open rate. Because the AUC value is only ~52%, model is only slightly better than guessing. We could probably make this model much more accurate with something like k-fold cross validation, stepwise regression, or PCA, but for time and simplicity's sake I'll stick with the model we have.

```
#check accuracy of model
valid$results = predict(o.glm, newdata=valid, type='response')
summary(valid$results)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.05690 0.08453 0.09184 0.09245 0.09969 0.14231
```

```
#choose threshhold of 10%
fitround = ifelse(valid$results > 0.1, 1,0)

t = table(fitround, valid$opened)
acc <- (t[1,1] + t[2,2]) / sum(t)
acc
```

```
## [1] 0.7183358
```

```
t
```

```
##
```

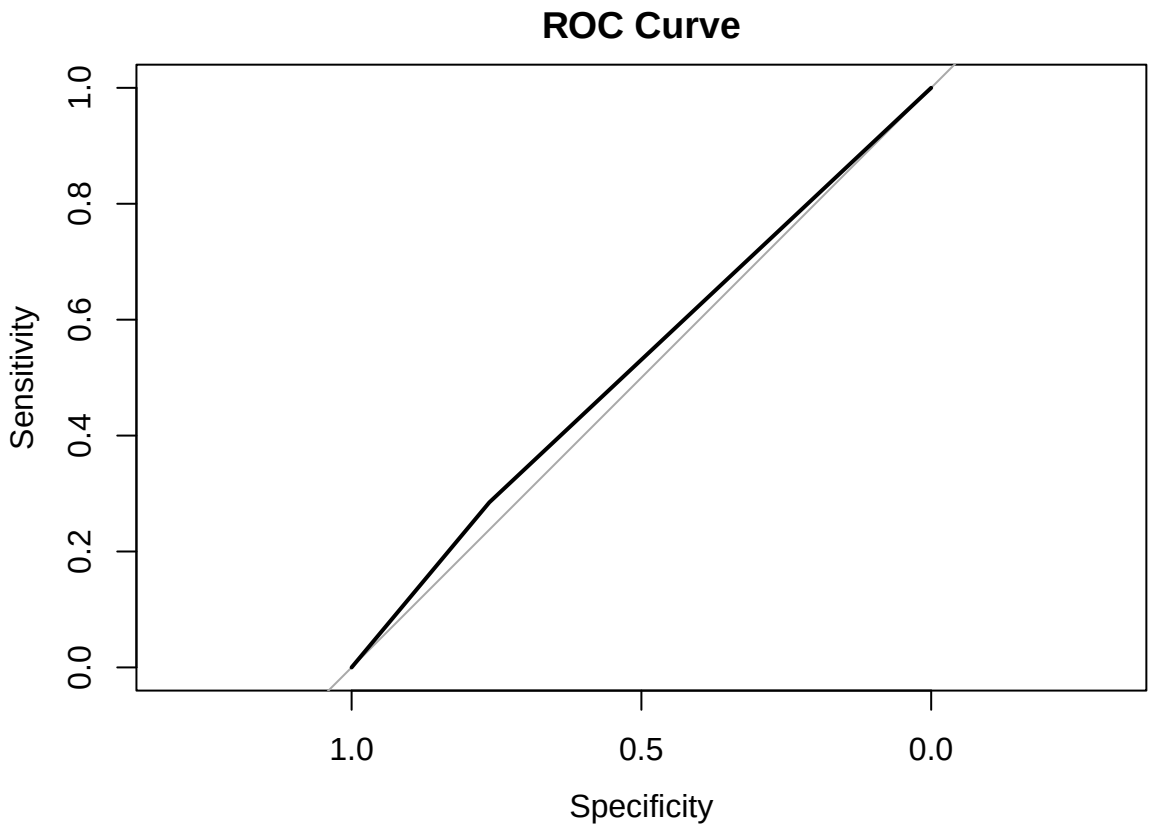
```
## fitround      0      1
##        0 514138  49033
##        1 160217  19518
```

```
r<-roc(valid$opened,fitround)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```
plot(r, main='ROC Curve')
```




```
r
```

```
##
## Call:
## roc.default(response = valid$opened, predictor = fitround)
##
## Data: fitround in 674355 controls (valid$opened 0) < 68551 cases (valid$opened 1).
## Area under the curve: 0.5236
```

Q10) For this problem, I created a logistic regression model with the response (opened) as a function of age, type, and gender. Then I created a new dataframe using the information that I have to predict and plot the logistic regression curve. Lastly, I used ggplot to plot the relationship between open rate, age, type with separate plots for each gender. Since the curves for both genders look very similar, that indicates to us that gender is not a significant factor when predicting open rate as a function of age, type, and gender.

```
openr = glm(opened ~ Age + Type, family=binomial, data=oprate)

newdat = with(oprate, expand.grid(Type=unique(Type),
```

```
                                    Age=quantile(Age),
                                    Gender=Gender))
#use model created above to predict results
newdat$prob = predict(openr, newdat, type='response')

ggplot(newdat, aes(Age, prob, color=factor(Type))) +
  geom_line() +
  facet_grid(.~Gender)
```