

W241 Final Project Experiment Results

Jack Workman & Yulia Zamriy

August 1, 2018

1. Pilot Study

TODO

2. Getting the Data

2.1 Load the Data

Survey responses were sourced from three different environments: 1. Amazon Mechanical Turk with Masters qualification 2. Amazon Mechanical Turk without Masters qualification 3. Friends & Family (Facebook, LinkedIn, I School Slack)

Although this step is not critical for analyzing the results of the experiment, it might be interesting to know later on where a subject was recruited from. So, here we pre-process the data to explicitly identify each subject's group. We do that by matching the mTurkCode from the Amazon Mechanical Turk (mturk) responses to the mTurkCode in the qualtrics data set.

```
mturk_masters =  
  read.csv("experiment_results/mturk_W241_Final-Project_Survey_Masters-Batch_Results.csv")  
mturk_regulares =  
  read.csv("experiment_results/mturk_W241_Final-Project_Survey_Non-Masters-Batch_Results.csv")  
  
# the second and third rows of the qualtrics results contain meta info  
# we do not need so we remove it here  
all_content =  
  readLines("experiment_results/qualtrics_W241_Final-Project_Survey_Experiment_August-1-2018_14-06.csv")  
all_content = all_content[-3]  
all_content = all_content[-2]  
qualtrics = read.csv(textConnection(all_content), header = TRUE, stringsAsFactors = FALSE)  
rm(all_content)  
# thank you: https://stackoverflow.com/questions/15860071/read-csv-header-on-first-line-skip-second-line
```

2.2 Concatenate MTurk Masters & MTurk Regulars

```
# assign groups  
mturk_masters['source_group'] = 1  
mturk_regulares['source_group'] = 2  
mturk_all = rbind(mturk_masters, mturk_regulares)  
  
# keep these columns and drop the rest (switched to reverse where we explicitly drop the not necessary  
#mturk_cols_to_keep = c('HITId', 'CreationTime', 'Expiration', 'AssignmentId', 'WorkerId', 'AssignmentS  
#mturk_all = mturk_all[mturk_cols_to_keep]  
  
# drop unnecessary columns  
mturk_all = subset(mturk_all, select = -c(HITTypeId, Title, Description, Keywords,
```

```

Reward, MaxAssignments, RequesterAnnotation,
AssignmentDurationInSeconds, AutoApprovalDelayInSeconds,
NumberOfSimilarHITs, LifetimeInSeconds, AssignmentId,
AutoApprovalTime, ApprovalTime, RejectionTime,
RequesterFeedback, Approve, Reject))

# rename mTurkCode column to match qualtrics
mturk_all['mTurkCode'] = mturk_all['Answer.surveycode']

# check out the combined dataset
#head(mturk_all)
table(mturk_all['source_group'])

##
##    1    2
## 42 300

dim(mturk_all)

## [1] 342  14

```

2.3 Merge MTurk and Qualtrics Datasets

```

#head(qualtrics)
responses = merge(x = qualtrics, y = mturk_all, by = "mTurkCode", all.x = TRUE)
dim(responses)

## [1] 519  42

table(responses$source_group)

##
##    1    2
## 42 298

#head(responses)
#colnames(responses)
responses$source_group[is.na(responses$source_group)] = 3
responses$source_group <- factor(responses$source_group, labels = c("Mturk Masters", "Mturk Regulars", "F&F"))
#hist(responses$source_group, breaks=3) # could probably make this prettier
table(responses$source_group)

##
## Mturk Masters Mturk Regulars F&F
##          42          298          179

rm(list = c('mturk_all', 'mturk_masters', 'mturk_regulars', 'qualtrics'))

```

3. Data Cleaning

3.1 Identifying Invalid Data Rows

Potentially invalid responses are those that:

- have Status = 'Spam' or 'Survey Preview'

- Duplicate IP Address occurrence. We'll keep the first response for the analysis (alternatively, we can exclude all of them)
- not finished (progress less than 100%). These cases need to be investigated for potential bias

3.1.1 Invalid Status

```
table(responses$Status)
```

```
##
##      IP Address      Spam Survey Preview
##      514           1           4
```

3.1.2 Duplicate IP Addresses

```
# record duplicate IP Addresses and first timestamp
responses %>%
  group_by(IPAddress) %>%
  arrange(StartDate) %>%
  summarize(ip_count = n(),
            StartDate = first(StartDate)) %>%
  filter(ip_count > 1) -> duplicate_ips

# merge duplicate IP Addresses based on first time stamp
responses <- merge(x = responses, y = duplicate_ips,
                  by = c("IPAddress", "StartDate"), all.x = TRUE)

# merge all duplicate IP Addresses
responses <- merge(x = responses, y = duplicate_ips[,c("IPAddress", "ip_count")],
                  by = "IPAddress", all.x = TRUE)

responses$duplicate_ip <- 0
responses[!is.na(responses$ip_count.y),]$duplicate_ip <- 1
responses$duplicate_ip_xfirst <- 0
responses[!is.na(responses$ip_count.x),]$duplicate_ip_xfirst <- 1
responses$ip_count.x <- NULL
colnames(responses)[colnames(responses)=="ip_count.y"] <- "ip_count"
responses[is.na(responses$ip_count),]$ip_count <- 1
rm(duplicate_ips)

table(responses$duplicate_ip, responses$duplicate_ip_xfirst)
```

```
##
##      0      1
## 0 430      0
## 1   60     29
```

```
table(responses$ip_count)
```

```
##
## 1  2  3  4  5 16
## 430 38 9 16 10 16
```

3.1.3 Unfinished Surveys

We have 70 responses that were not finished. We'll create a separate flag variable to determine if they are a cause for attrition concern:

```
## =====
##
## Summary of survey's progress:
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      53.00  100.00  100.00   94.24  100.00  100.00
## =====
##
## Number of responses that are not finished:
## [1] 70
## =====
##
## Primary outcome distribution for unfinished responses:
## < table of extent 0 >
## =====
##
## Source of unfinished responses:
##
##      Mturk Masters Mturk Regulars      F&F
##              0              0          70
## =====
##
## Treatment assignment for unfinished responses:
##
##      Introduction:1Star Introduction:4.5Stars Introduction:Control
##              17              28              25
```

3.1.4 Putting It All Together

```
responses$valid <- "Valid"
responses[responses$Status %in% c("Spam", "Survey Preview"), ]$valid <- "Preview/Spam"
responses[responses$duplicate_ip_xfirst==1,]$valid <- "Duplicate"
responses[responses$Progress < 100,]$valid <- "Not Finished"
table(responses$valid)
```

```
##
##      Duplicate Not Finished Preview/Spam      Valid
##              23              70              3      423
## =====
##
## Source of valid responses:
##
##      Mturk Masters Mturk Regulars      F&F
##              39              291          93
##
## Source of invalid responses:
```

```
##
##   Mturk Masters Mturk Regulars      F&F
##           3           7           86
## =====
##
## Treatment assignment for valid responses:
##
##   Introduction:1Star Introduction:4.5Stars Introduction:Control
##           135           141           147
##
## Treatment assignment for invalid responses:
##
##   Introduction:1Star Introduction:4.5Stars Introduction:Control
##           27           38           31
table(responses[responses$valid == "Valid",]$FL_2_D0,
      responses[responses$valid == "Valid",]$source_group)

##
##           Mturk Masters Mturk Regulars F&F
## Introduction:1Star           10           90 35
## Introduction:4.5Stars          12           101 28
## Introduction:Control          17           100 30
```

3.2 Are Valid Responses Actually Valid?

```
short_story_word_count = 990
duration_secs_minimum = 30
correct_answers_minimum = 1
```

This experiment relies entirely on the assumption that the subjects read the short story. To ensure this, we added several validation checks to the survey.

First, a timer to track how long each participant spends on the short story page itself. The short story is 990 words, so any subject with less than 30, a reading speed of 0.55 will be dropped. Given that the adult average reading speed is about 200 wpm), we believe that this is more than justified.

Second, the survey contains three reading comprehension questions to test the reader's understanding of the story. These questions are designed to be extremely basic and high-level. In fact, the questions were made easier after the pilot as those were deemed to be too difficult. If the subject read the story, then they should be able to answer these questions. Since no one is perfect, we are electing to keep all subjects that answered at least 1 answer correctly. We drop the rest.

Finally, we will be dropping any subject that failed to answer all questions in the survey.

```
valid_responses = subset(responses, valid == "Valid")

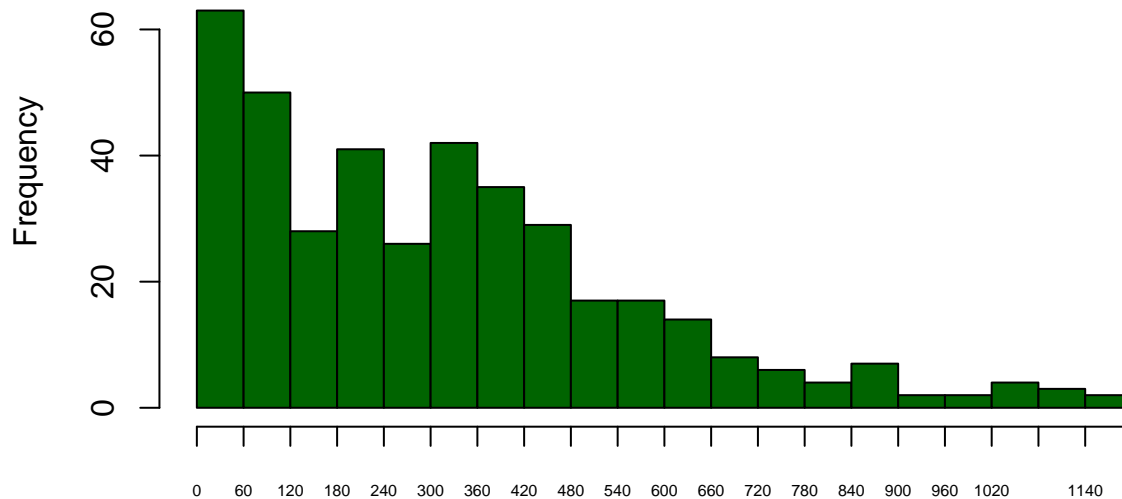
summary(valid_responses$Duration..in.seconds.)

##   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    9.0  104.5   305.0   744.5   494.0 78097.0
sum(valid_responses$Duration..in.seconds. < 1200)

## [1] 400
```

```
hist(valid_responses[valid_responses$Duration..in.seconds. < 1200,]$Duration..in.seconds.,
     col = "darkgreen",
     breaks = seq(0,1200,60),
     ylim = c(0, 75),
     xaxt = 'n',
     xlab = "Time spent on the whole survey",
     main = "Histogram of time spent on survey")
axis(side = 1, at = seq(0,1200,60), labels = seq(0,1200,60), cex.axis = 0.5)
```

Histogram of time spent on survey



Time spent on the whole survey

```
summary(valid_responses$Q2_Page.Submit)
```

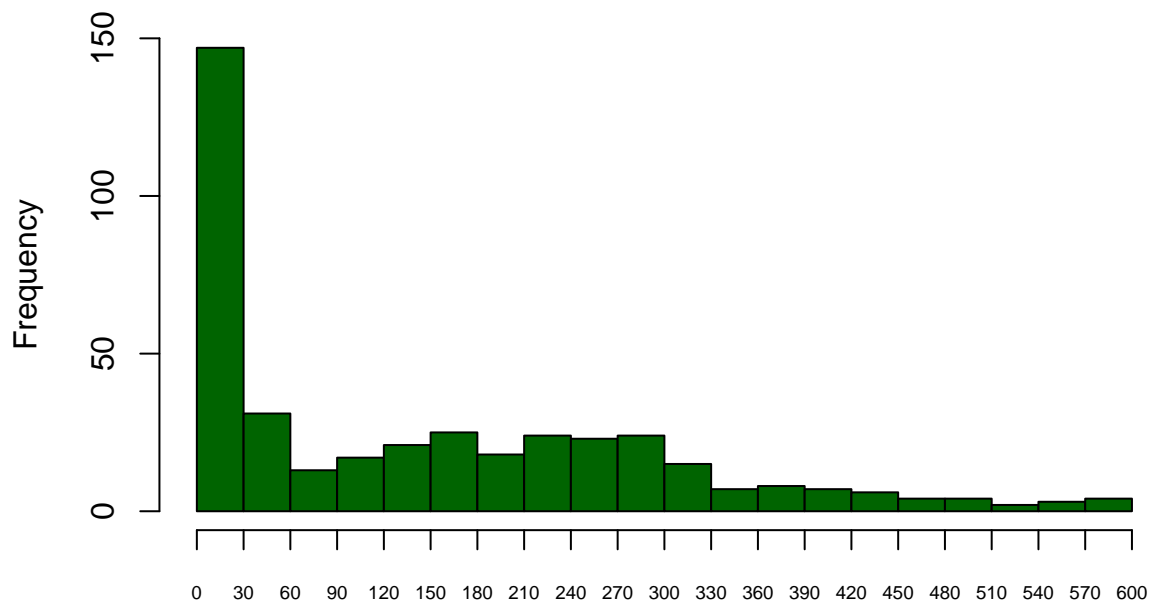
```
##      Min.   1st Qu.   Median     Mean   3rd Qu.     Max.
##  1.511    9.493   126.567   204.099   265.954  7691.925
```

```
sum(valid_responses$Q2_Page.Submit < 600)
```

```
## [1] 403
```

```
hist(valid_responses[valid_responses$Q2_Page.Submit < 600,]$Q2_Page.Submit,
     col = "darkgreen",
     breaks = seq(0,600,30),
     ylim = c(0, 150),
     xaxt = 'n',
     xlab = "Time spent on the story page",
     main = "Histogram of time spent reading")
axis(side = 1, at = seq(0,600,30), labels = seq(0,600,30), cex.axis = 0.6)
```

Histogram of time spent reading



Time spent on the story page

```
# reading time check
#valid_responses = subset(valid_responses, Q2_Page.Submit >= duration_secs_minimum)
valid_responses[valid_responses$Q2_Page.Submit < duration_secs_minimum,]$valid <- "Undertime"
table(valid_responses$valid)

##
## Undertime      Valid
##      147        276

table(valid_responses$valid, valid_responses$FL_2_D0)

##
##           Introduction:1Star Introduction:4.5Stars Introduction:Control
## Undertime           45           53           49
## Valid             90           88           98

table(valid_responses$valid, valid_responses$source_group)

##
##           Mturk Masters Mturk Regulars F&F
## Undertime           5          128   14
## Valid             34          163   79

# reading questions check
Q12_correct_ans = 'Beautiful'
Q14_correct_ans = 'Rome'
Q16_correct_ans = 'Gaius was killed'
valid_responses['Q12_correct'] = ifelse(valid_responses$Q12 == Q12_correct_ans, 1, 0)
```

```

valid_responses['Q14_correct'] = ifelse(valid_responses$Q14 == Q14_correct_ans, 1, 0)
valid_responses['Q16_correct'] = ifelse(valid_responses$Q12 == Q16_correct_ans, 1, 0)
valid_responses['correct_answers'] = valid_responses$Q12_correct +
  valid_responses$Q14_correct + valid_responses$Q16_correct
table(valid_responses$correct_answers)

##
##      0      1      2
## 63 121 239

valid_responses[valid_responses$correct_answers < correct_answers_minimum,]$valid <- "Very incorrect"
table(valid_responses$valid)

##
##      Undertime      Valid Very incorrect
##           95          265          63

sum(valid_responses$Q4 == -99)

## [1] 8

valid_responses[valid_responses$Q4 == -99,]$valid <- "Missing outcome"
table(valid_responses$valid)

##
## Missing outcome      Undertime      Valid Very incorrect
##           8          95          265          55

table(valid_responses$Q25)

##
##      -99 No Yes
##      4   5 292 122

valid_responses$issues <- "No issues"
#valid_responses[valid_responses$Q25 == 'Yes',]$issues <- "Know the story"
valid_responses[valid_responses$Q5 == '-99',]$issues <- "Missing text review"
valid_responses[valid_responses$Q25 == '-99',]$issues <- "Missing answer"
valid_responses[valid_responses$Q12 == '-99',]$issues <- "Missing answer"
valid_responses[valid_responses$Q14 == '-99',]$issues <- "Missing answer"
valid_responses[valid_responses$Q16 == '-99',]$issues <- "Missing answer"
table(valid_responses$issues)

##
##      Missing answer Missing text review      No issues
##           16          22          385

table(valid_responses$valid, valid_responses$issues)

##
##      Missing answer Missing text review No issues
## Missing outcome      8          0          0
## Undertime          1          8         86
## Valid            1          8        256
## Very incorrect    6          6         43

nrows_raw <- nrow(responses)
excl_dups_unfinished <- nrow(valid_responses)

```



```

excl_dur_undertime <- excl_dups_unfinished - sum(valid_responses$valid == 'Undertime')
excl_incorrect_answ <- excl_dur_undertime - sum(valid_responses$valid == 'Very incorrect')
excl_missing_outcome <- excl_incorrect_answ - sum(valid_responses$valid == 'Missing outcome')
nrows_final <- sum(valid_responses$valid == 'Valid')

labels = c('Raw Num\nResponses',
           'Duplicate IPs/Unfinished',
           sprintf('Reading Duration\n>= %s secs', duration_secs_minimum),
           sprintf('Num Correct\nAnswers < %s', correct_answers_minimum),
           'Missing outcome answers',
           'Final Count')

desc = factor(labels, levels=labels)
type = c('in', 'out', 'maybe out', 'maybe out', 'out', 'in')
type = factor(type, levels = c("in", "out", "maybe out"))

start = c(0, nrows_raw,
          excl_dups_unfinished,
          excl_dur_undertime,
          excl_incorrect_answ,
          excl_missing_outcome)
end = c(nrows_raw,
        excl_dups_unfinished,
        excl_dur_undertime,
        excl_incorrect_answ,
        nrows_final, 0)
amount = c(nrows_raw,
           -excl_dups_unfinished,
           -excl_dur_undertime,
           -excl_incorrect_answ,
           -excl_missing_outcome,
           -nrows_final)
id = seq_along(amount)
waterfall_data = data.frame(id, desc, type, start, end, amount)

label_names <- c(nrows_raw,
                 nrows_raw - excl_dups_unfinished,
                 excl_dups_unfinished - excl_dur_undertime,
                 excl_dur_undertime - excl_incorrect_answ,
                 excl_incorrect_answ - excl_missing_outcome,
                 nrows_final)

adj <- 1.075
label_pos <- c(nrows_raw * adj,
               excl_dups_unfinished * adj,
               excl_dur_undertime * adj,
               excl_incorrect_answ * adj,
               excl_missing_outcome * adj,
               nrows_final * adj)

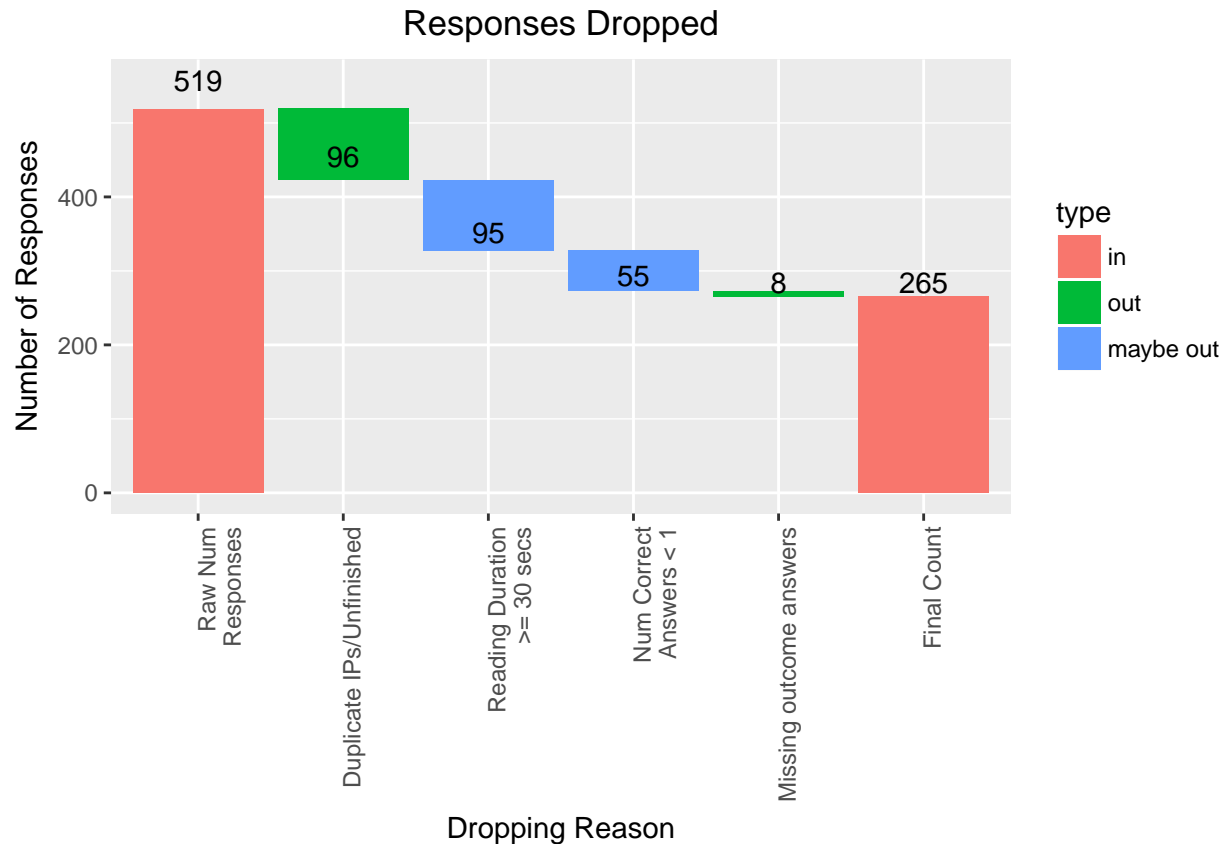
waterfall_plot =
  ggplot(waterfall_data, aes(desc, fill = type,
                             xmin = id - 0.45, xmax = id + 0.45, ymin = end, ymax = start,
                             label = label_names)) +

```

```

geom_rect()
waterfall_plot +
  ggtitle("Responses Dropped") +
  theme(plot.title = element_text(hjust = 0.5),
        axis.text.x = element_text(angle = 90, hjust = 1)) +
  labs(y="Number of Responses", x="Dropping Reason") +
  geom_text(aes(y = label_pos, label = label_names))

```



Identify Treatment and Control Groups

There are three distinct groups in this experiment:

1. Control
2. Treatment - Low Rating
3. Treatment - High Rating

```

valid_responses['experiment_group'] =
  ifelse(valid_responses$FL_2_DO == 'Introduction:Control', 1,
        ifelse(valid_responses$FL_2_DO == 'Introduction:1Star', 2,
              ifelse(valid_responses$FL_2_DO == 'Introduction:4.5Stars', 3, -1)))
valid_responses['treated'] = ifelse(valid_responses$experiment_group == 2, 1,
                                   ifelse(valid_responses$experiment_group == 3, 1, 0))
valid_responses['treatment_rating'] = ifelse(valid_responses$experiment_group == 2, 2,
                                             ifelse(valid_responses$experiment_group == 3, 5, NA))
valid_responses['experiment_group_chr'] =
  ifelse(valid_responses$FL_2_DO == 'Introduction:Control', "Control",

```

```

    ifelse(valid_responses$FL_2_D0 == 'Introduction:1Star', "Treat: Low",
           ifelse(valid_responses$FL_2_D0 == 'Introduction:4.5Stars', "Treat: High", "NA")))

table(valid_responses$experiment_group, valid_responses$experiment_group_chr)

##
##      Control Treat: High Treat: Low
##  1      147          0          0
##  2         0          0        135
##  3         0        141          0

summary(valid_responses$Q4)

##      Min. 1st Qu.  Median      Mean 3rd Qu.      Max.
## -99.000   3.000   4.000   2.135   5.000   6.000

summary(valid_responses[valid_responses$valid == "Valid",]$Q4)

##      Min. 1st Qu.  Median      Mean 3rd Qu.      Max.
##   1.000   3.000   4.000   4.125   5.000   6.000

valid_responses %>%
  filter(valid != "Missing outcome") %>%
  group_by(valid, FL_2_D0) %>%
  summarize(responseCount = n(),
            avgRating = round(mean(Q4),1))

## # A tibble: 9 x 4
## # Groups:   valid [?]
##   valid      FL_2_D0      responseCount avgRating
##   <chr>      <chr>             <int>      <dbl>
## 1 Undertime Introduction:1Star         33         4.5
## 2 Undertime Introduction:4.5Stars      33         4.4
## 3 Undertime Introduction:Control       29         3.8
## 4 Valid      Introduction:1Star         86         3.7
## 5 Valid      Introduction:4.5Stars      84         4.3
## 6 Valid      Introduction:Control       95         4.4
## 7 Very incorrect Introduction:1Star         12         2.9
## 8 Very incorrect Introduction:4.5Stars      22         3.7
## 9 Very incorrect Introduction:Control       21         4

```

Calculate ATE

```

calc_exp_group_avg_rating = function(data, group) {
  avg_rating = mean(subset(data, experiment_group == group)$Q4)
  return(avg_rating)
}

get_nrow_of_group = function(data, group) {
  rows = nrow(subset(data, experiment_group == group))
  return(rows)
}

get_pct_subjects_of_group = function(data, group) {
  count = get_nrow_of_group(data, group)
  pct = round(count / nrow(data) * 100, 2)
  return(pct)
}

```

```

}
control_avg_rating = calc_exp_group_avg_rating(valid_responses[valid_responses$valid == "Valid",], 1)
treatment_low_avg_rating = calc_exp_group_avg_rating(valid_responses[valid_responses$valid == "Valid",], 2)
treatment_high_avg_rating = calc_exp_group_avg_rating(valid_responses[valid_responses$valid == "Valid",], 3)
# todo: is this actually how you calculate ATE?
ate_high = treatment_high_avg_rating - control_avg_rating
ate_low = treatment_low_avg_rating - control_avg_rating
groups = c(
  'Control',
  'Treatment - Low Rating',
  'Treatment - High Rating')
counts = c(
  get_nrow_of_group(valid_responses[valid_responses$valid == "Valid",], 1),
  get_nrow_of_group(valid_responses[valid_responses$valid == "Valid",], 2),
  get_nrow_of_group(valid_responses[valid_responses$valid == "Valid",], 3))
pcts = c(
  get_pct_subjects_of_group(valid_responses[valid_responses$valid == "Valid",], 1),
  get_pct_subjects_of_group(valid_responses[valid_responses$valid == "Valid",], 2),
  get_pct_subjects_of_group(valid_responses[valid_responses$valid == "Valid",], 3))
avg_ratings = c(
  round(control_avg_rating, 2),
  round(treatment_low_avg_rating, 2),
  round(treatment_high_avg_rating, 2))
ates = c(
  0,
  round(ate_low, 2),
  round(ate_high, 2))
treated_ratings = c('na', '2/6 Stars', '5/6 Stars')
outcome_table = data.frame(groups, counts, pcts, treated_ratings, avg_ratings, ates)
kable(outcome_table, col.names =
      c('Group', '# of Subjects', '% of Total Subjects', 'Treated Rating', 'AVG Rating', 'ATE'))

```

Group	# of Subjects	% of Total Subjects	Treated Rating	AVG Rating	ATE
Control	95	35.85	na	4.38	0.00
Treatment - Low Rating	86	32.45	2/6 Stars	3.71	-0.67
Treatment - High Rating	84	31.70	5/6 Stars	4.26	-0.12

Model

```

model = lm(Q4 ~ experiment_group_chr,
           data = valid_responses[valid_responses$valid == "Valid",])
summary(model)

##
## Call:
## lm(formula = Q4 ~ experiment_group_chr, data = valid_responses[valid_responses$valid ==
## "Valid", ])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -3.2619 -0.7093  0.2907  0.7381  2.2907

```

```
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)         4.3789     0.1042  42.018 < 2e-16 ***
## experiment_group_chrTreat: High -0.1170     0.1521  -0.769    0.442
## experiment_group_chrTreat: Low  -0.6696     0.1512  -4.429 1.39e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.016 on 262 degrees of freedom
## Multiple R-squared:  0.07703,    Adjusted R-squared:  0.06999
## F-statistic: 10.93 on 2 and 262 DF,  p-value: 2.751e-05

coefci(model)

##               2.5 %    97.5 %
## (Intercept)         4.1737403  4.5841545
## experiment_group_chrTreat: High -0.4165995  0.1825143
## experiment_group_chrTreat: Low  -0.9673475 -0.3719426

model = lm(Q4 ~ experiment_group_chr,
           data = valid_responses[valid_responses$valid != "Missing outcome",])
summary(model)

##
## Call:
## lm(formula = Q4 ~ experiment_group_chr, data = valid_responses[valid_responses$valid !=
## "Missing outcome", ])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -3.2014 -0.8321 -0.2000  0.7986  2.1679
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)         4.200000    0.092202  45.552 < 2e-16 ***
## experiment_group_chrTreat: High  0.001439    0.131793   0.011  0.99129
## experiment_group_chrTreat: Low  -0.367939    0.133832  -2.749  0.00624 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.11 on 412 degrees of freedom
## Multiple R-squared:  0.02343,    Adjusted R-squared:  0.01869
## F-statistic: 4.942 on 2 and 412 DF,  p-value: 0.007571

coefci(model)

##               2.5 %    97.5 %
## (Intercept)         4.0187545  4.3812455
## experiment_group_chrTreat: High -0.2576323  0.2605100
## experiment_group_chrTreat: Low  -0.6310179 -0.1048599

valid_responses$undertime <- ifelse(valid_responses$valid == "Undertime", 1, 0)
valid_responses$incorrect <- ifelse(valid_responses$valid == "Very incorrect", 1, 0)

model = lm(Q4 ~ experiment_group_chr + undertime + incorrect,
           data = valid_responses[valid_responses$valid != "Missing outcome",])
```

```
summary(model)
```

```
##
## Call:
## lm(formula = Q4 ~ experiment_group_chr + undertime + incorrect,
##     data = valid_responses[valid_responses$valid != "Missing outcome",
##     ])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -3.3698 -0.8515  0.1485  0.7463  2.2776
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)      4.253724   0.098899  43.011 < 2e-16 ***
## experiment_group_chrTreat: High  0.004239   0.130159   0.033  0.97404
## experiment_group_chrTreat: Low -0.402244   0.132407  -3.038  0.00253 **
## undertime         0.116109   0.131130   0.885  0.37643
## incorrect        -0.531294   0.162830  -3.263  0.00120 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.095 on 410 degrees of freedom
## Multiple R-squared:  0.05399,    Adjusted R-squared:  0.04476
## F-statistic:  5.85 on 4 and 410 DF,  p-value: 0.0001381
```

```
coefci(model)
```

```
##              2.5 %      97.5 %
## (Intercept)      4.0593120  4.4481364
## experiment_group_chrTreat: High -0.2516234  0.2601012
## experiment_group_chrTreat: Low -0.6625253 -0.1419624
## undertime        -0.1416626  0.3738808
## incorrect        -0.8513802 -0.2112074
```

```
table(valid_responses$valid, valid_responses$source_group)
```

```
##
##           Mturk Masters Mturk Regulars F&F
## Missing outcome           0           0    8
## Undertime                4           86    5
## Valid                   33          155   77
## Very incorrect           2           50    3
```

```
model = lm(Q4 ~ experiment_group_chr*source_group,
           data = valid_responses[valid_responses$valid == "Valid",])
summary(model)
```

```
##
## Call:
## lm(formula = Q4 ~ experiment_group_chr * source_group, data = valid_responses[valid_responses$valid ==
##     "Valid", ])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -3.3208 -0.6304  0.0333  0.6792  2.3696
```

```

##
## Coefficients:
##
## (Intercept) Estimate
## experiment_group_chrTreat: High 0.37302
## experiment_group_chrTreat: Low -0.77143
## source_groupMturk Regulars 0.50000
## source_groupF&F 0.04857
## experiment_group_chrTreat: High:source_groupMturk Regulars -0.62369
## experiment_group_chrTreat: Low:source_groupMturk Regulars -0.16957
## experiment_group_chrTreat: High:source_groupF&F -0.44756
## experiment_group_chrTreat: Low:source_groupF&F 0.61810
## Std. Error
## (Intercept) 0.26923
## experiment_group_chrTreat: High 0.43040
## experiment_group_chrTreat: Low 0.41709
## source_groupMturk Regulars 0.30101
## source_groupF&F 0.33627
## experiment_group_chrTreat: High:source_groupMturk Regulars 0.47171
## experiment_group_chrTreat: Low:source_groupMturk Regulars 0.46276
## experiment_group_chrTreat: High:source_groupF&F 0.52150
## experiment_group_chrTreat: Low:source_groupF&F 0.49838
## t value
## (Intercept) 15.122
## experiment_group_chrTreat: High 0.867
## experiment_group_chrTreat: Low -1.850
## source_groupMturk Regulars 1.661
## source_groupF&F 0.144
## experiment_group_chrTreat: High:source_groupMturk Regulars -1.322
## experiment_group_chrTreat: Low:source_groupMturk Regulars -0.366
## experiment_group_chrTreat: High:source_groupF&F -0.858
## experiment_group_chrTreat: Low:source_groupF&F 1.240
## Pr(>|t|)
## (Intercept) <2e-16 ***
## experiment_group_chrTreat: High 0.3869
## experiment_group_chrTreat: Low 0.0655 .
## source_groupMturk Regulars 0.0979 .
## source_groupF&F 0.8853
## experiment_group_chrTreat: High:source_groupMturk Regulars 0.1873
## experiment_group_chrTreat: Low:source_groupMturk Regulars 0.7144
## experiment_group_chrTreat: High:source_groupF&F 0.3916
## experiment_group_chrTreat: Low:source_groupF&F 0.2160
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.007 on 256 degrees of freedom
## Multiple R-squared:  0.113, Adjusted R-squared:  0.08529
## F-statistic: 4.077 on 8 and 256 DF, p-value: 0.0001374
coefci(model)

## 2.5 %
## (Intercept) 3.54123550
## experiment_group_chrTreat: High -0.47455635
## experiment_group_chrTreat: Low -1.59280015

```

```
## source_groupMturk Regulars -0.09277388
## source_groupF&F -0.61363951
## experiment_group_chrTreat: High:source_groupMturk Regulars -1.55261863
## experiment_group_chrTreat: Low:source_groupMturk Regulars -1.08087254
## experiment_group_chrTreat: High:source_groupF&F -1.47453849
## experiment_group_chrTreat: Low:source_groupF&F -0.36335883
## 97.5 %
## (Intercept) 4.60162164
## experiment_group_chrTreat: High 1.22058809
## experiment_group_chrTreat: Low 0.04994301
## source_groupMturk Regulars 1.09277388
## source_groupF&F 0.71078237
## experiment_group_chrTreat: High:source_groupMturk Regulars 0.30523918
## experiment_group_chrTreat: Low:source_groupMturk Regulars 0.74174211
## experiment_group_chrTreat: High:source_groupF&F 0.57941584
## experiment_group_chrTreat: Low:source_groupF&F 1.59954931
```

```
model = lm(Q4 ~ experiment_group_chr,
            data = valid_responses[valid_responses$valid == "Valid" & valid_responses$source_group == "Mturk Masters",
            summary(model)
```

```
##
## Call:
## lm(formula = Q4 ~ experiment_group_chr, data = valid_responses[valid_responses$valid ==
## "Valid" & valid_responses$source_group == "Mturk Masters",
## ])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.44444 -0.30000 -0.07143  0.70000  1.55556
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)      4.0714     0.2814  14.470 4.54e-15 ***
## experiment_group_chrTreat: High  0.3730     0.4498   0.829  0.4135
## experiment_group_chrTreat: Low -0.7714     0.4359  -1.770  0.0869 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.053 on 30 degrees of freedom
## Multiple R-squared:  0.1662, Adjusted R-squared:  0.1106
## F-statistic:  2.99 on 2 and 30 DF,  p-value: 0.06545
```

```
coefci(model)
```

```
##              2.5 %   97.5 %
## (Intercept)      3.4967963 4.646061
## experiment_group_chrTreat: High -0.5455974 1.291629
## experiment_group_chrTreat: Low -1.6616451 0.118788
```

```
model = lm(Q4 ~ experiment_group_chr,
            data = valid_responses[valid_responses$valid == "Valid" & valid_responses$source_group == "Mturk Masters",
            summary(model)
```

```
##
## Call:
```



```
## lm(formula = Q4 ~ experiment_group_chr, data = valid_responses[valid_responses$valid ==
## "Valid" & valid_responses$source_group == "Mturk Regulars",
## ])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -3.3208 -0.6304  0.3696  0.6792  2.3696
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)      4.5714     0.1388  32.936 < 2e-16 ***
## experiment_group_chrTreat: High -0.2507     0.1990  -1.259    0.21
## experiment_group_chrTreat: Low  -0.9410     0.2067  -4.553 1.08e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.039 on 152 degrees of freedom
## Multiple R-squared:  0.1249, Adjusted R-squared:  0.1134
## F-statistic: 10.85 on 2 and 152 DF, p-value: 3.931e-05
```

```
coefci(model)
```

```
##              2.5 %      97.5 %
## (Intercept)      4.2972094  4.8456477
## experiment_group_chrTreat: High -0.6439278  0.1425801
## experiment_group_chrTreat: Low  -1.3493310 -0.5326566
```

```
model = lm(Q4 ~ experiment_group_chr,
            data = valid_responses[valid_responses$valid == "Valid" &
            valid_responses$source_group == "F&F",])
```

```
summary(model)
```

```
##
## Call:
## lm(formula = Q4 ~ experiment_group_chr, data = valid_responses[valid_responses$valid ==
## "Valid" & valid_responses$source_group == "F&F", ])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.12000 -0.96667 -0.04545  0.03333  1.95455
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)      4.12000     0.18389  22.404 <2e-16 ***
## experiment_group_chrTreat: High -0.07455     0.26878  -0.277    0.782
## experiment_group_chrTreat: Low  -0.15333     0.24899  -0.616    0.540
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9195 on 74 degrees of freedom
## Multiple R-squared:  0.005119, Adjusted R-squared:  -0.02177
## F-statistic: 0.1904 on 2 and 74 DF, p-value: 0.827
```

```
coefci(model)
```

```
##              2.5 %      97.5 %
```

```
## (Intercept)          3.7535839 4.4864161
## experiment_group_chrTreat: High -0.6101104 0.4610195
## experiment_group_chrTreat: Low  -0.6494631 0.3427964
```