Data preparation

Instructions

- You only need to submit the .Rmd of this file, not a PDF.
- You should **comment** your code clearly to show what you've done to prepare the data.
- The purpose of this file is to use the data in the `data-raw` folder to create the data you will use in the report. The data you will use in the report should be saved in the `data` folder. It is good professional practice to make sure you're never directly modifying your raw data, but instead creating new datasets based on merges/manipulations that you need to reuse.
- Make sure you've taken a look at the hints for the web scraping and census API.
- You may find the `write_rds()` function from the `readr` package helpful (it is loaded as part of the `tidyverse`).
- You do not need to keep the structure below.

Set up

```
# Set up any libraries you need
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.8
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(polite)
library(rvest)
```

```
##
## Attaching package: 'rvest'
```

```
## The following object is masked from 'package:readr':
##
##     guess_encoding
```

```
library(haven)
library(lme4)
```

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'Matrix'
```

```
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
```

```
library(eeptools)
library(knitr)
library(sjPlot)
```

```
## Registered S3 method overwritten by 'parameters':
##   method                          from
##   format.parameters_distribution datawizard
```

```
library(cancensus)
```

```
## Census data is currently stored temporarily.
##
##  In order to speed up performance, reduce API quota usage, and reduce unnecessary network calls, plea
##
##  You may add this environment varianble to your .Renviron or add this option, together with your API
```

```
library(janitor)
```

```
##
## Attaching package: 'janitor'
```

```
## The following objects are masked from 'package:stats':
##
##     chisq.test, fisher.test
```

Loading client data

```
customer <- read_rds("data-raw/customer.Rds")
device <- read_rds("data-raw/device.Rds")
cust_sleep <- read_rds("data-raw/cust_sleep.Rds")
cust_dev <- read_rds("data-raw/cust_dev.Rds")
break_glass <- read_rds("data-raw/break_glass_in_case_of_emergency.Rds")
```

Web scraping industry data

```
url <- "https://fitnesstrackerinfohub.netlify.app/"
target <- bow(url,
              user_agent = "alexandertran.tran@mail.utoronto.ca for STA303/1002 project",
              force = TRUE)
target
```

```
## <polite session> https://fitnesstrackerinfohub.netlify.app/
##     User-agent: alexandertran.tran@mail.utoronto.ca for STA303/1002 project
##     robots.txt: 2 rules are defined for 2 bots
##    Crawl delay: 12 sec
##   The path is scrapable for this user-agent
```

```
html <- scrape(target)

device_data <- html %>%
  html_elements("table") %>%
```

```
  html_table() %>%
  pluck(1) # added, in case you're getting a list formats
```

Postcode

```
postcode <- read_sav("pccfNat_fccpNat_082021sav.sav", col_select = c("PC", "CSDuid"))
```

Census API

```
options(cancensus.api_key = "CensusMapper_e044495035736957fbd42a934af93f49",
        cancensus.cache_path = "cache") # this sets a folder for your cache


# get all regions as at the 2016 Census (2020 not up yet)
regions <- list_census_regions(dataset = "CA16")
```

```
## Querying CensusMapper API for regions data...
```

```
regions_filtered <-  regions %>%
  filter(level == "CSD") %>% # Figure out what CSD means in Census data
  as_census_region_list()



# This can take a while
# We want to get household median income
census_data_csd <- get_census(dataset='CA16', regions = regions_filtered,
                          vectors=c("v_CA16_2397"),
                          level='CSD', geo_format = "sf")
```

```
## Reading vectors data from local cache.
```

```
## Reading geo data from local cache.
```

```
#Simplify to only needed variables

median_income <- census_data_csd %>%
  as_tibble() %>%
  select(CSDuid = GeoUID, contains("median"), Population) %>%
  mutate(CSDuid = parse_number(CSDuid)) %>%
  rename(hhld_median_inc = 2)
```

Data cleaning and processing

```
# Rename the column of device_data so that we can combine it with device
device_data_adj <- clean_names(device_data) %>% mutate(released=as.Date(released))

# Set the released column in device into date type
device_adj <- clean_names(device) %>% mutate(released=as.Date(released))

# Join device_adj and device_data_adj
device_full <- device_adj %>% right_join(device_data_adj, by=c("line", "device_name", "released"))

cust_dev_joined <- cust_sleep %>% left_join(customer, by="cust_id") %>% left_join(cust_dev, by="cust_id
```

```r
# Clean the postcode data and create a new dataframe postcode_adj
postcode_adj <- clean_names(postcode)
postcode_adj <- rename(postcode_adj, postcode=pc)

# Clean the median_income data and create a new dataframe median_income_adj
# Combine median_income_adj and postcode_adj
median_income_adj <- clean_names(median_income)
pc_income <- postcode_adj %>% left_join(median_income_adj, by="cs_duid")

# Combining with postal code and income
cust_dev_joined <- cust_dev_joined %>% left_join(pc_income, by="postcode")

# Cleaning variables of interest
cust_dev_joined <- cust_dev_joined %>% mutate(dob=as.Date(dob))
cust_dev_joined <- cust_dev_joined %>% mutate(age=floor(age_calc(dob, units = "years")))
cust_dev_joined <- cust_dev_joined %>% drop_na(cust_id)
cust_dev_joined$emoji_modifier[is.na(cust_dev_joined$emoji_modifier)] <- "Not Set"
cust_dev_joined$sex[is.na(cust_dev_joined$sex)] <- "Not Set"
```

Initial Exploratory Data Analysis

```r
#Data summaries
#Note that means and variances are calculated as a flag rate of per 60 minutes of sleep
cust_dev_joined %>% group_by(emoji_modifier) %>% summarise(n = n(), mean = mean(flags*60/duration), medi
```

```
## # A tibble: 6 x 6
##   emoji_modifier     n  mean median    sd    var
##   <chr>          <int> <dbl>  <dbl> <dbl>  <dbl>
## 1 Not Set         7471 0.393  0.369 0.296 0.0874
## 2 U+1F3FB         5396 0.182  0.152 0.197 0.0387
## 3 U+1F3FC         4577 0.392  0.359 0.296 0.0879
## 4 U+1F3FD         4365 0.598  0.576 0.357 0.127
## 5 U+1F3FE         3561 1.21   1.19  0.504 0.254
## 6 U+1F3FF         3495 2.01   1.98  0.621 0.386
```

```r
cust_dev_joined %>% group_by(sex) %>% summarise(n = n(), mean = mean(flags*60/duration), median = media
```

```
## # A tibble: 4 x 6
##   sex          n  mean median    sd   var
##   <chr>    <int> <dbl>  <dbl> <dbl> <dbl>
## 1 Female   16814 0.686  0.459 0.691 0.478
## 2 Intersex   292 0.645  0.442 0.676 0.457
## 3 Male     11530 0.677  0.463 0.682 0.465
## 4 Not Set    229 0.575  0.439 0.625 0.391
```

```r
cust_dev_joined %>% mutate(age_bracket = cut(age, breaks = seq(15, 95, by=5))) %>%  group_by(age_bracke
```

```
## # A tibble: 16 x 6
##    age_bracket     n  mean median    sd   var
##    <fct>       <int> <dbl>  <dbl> <dbl> <dbl>
##  1 (15,20]      1600 0.886  0.549 0.868 0.753
##  2 (20,25]      2122 0.436  0.411 0.439 0.193
##  3 (25,30]      2852 0.701  0.506 0.686 0.470
##  4 (30,35]      2781 0.793  0.573 0.771 0.594
```

```
##  5 (35,40]      2269 0.699  0.533 0.638 0.407
##  6 (40,45]      2735 0.804  0.475 0.837 0.701
##  7 (45,50]      2601 0.727  0.529 0.693 0.480
##  8 (50,55]      2674 0.625  0.451 0.624 0.390
##  9 (55,60]      2231 0.591  0.367 0.604 0.365
## 10 (60,65]      2389 0.641  0.379 0.632 0.400
## 11 (65,70]      2130 0.687  0.528 0.655 0.428
## 12 (70,75]      1068 0.540  0.377 0.544 0.296
## 13 (75,80]       501 0.564  0.377 0.606 0.367
## 14 (80,85]       272 0.689  0.418 0.704 0.495
## 15 (85,90]       530 0.615  0.424 0.605 0.367
## 16 (90,95]       110 0.687  0.454 0.589 0.347
```

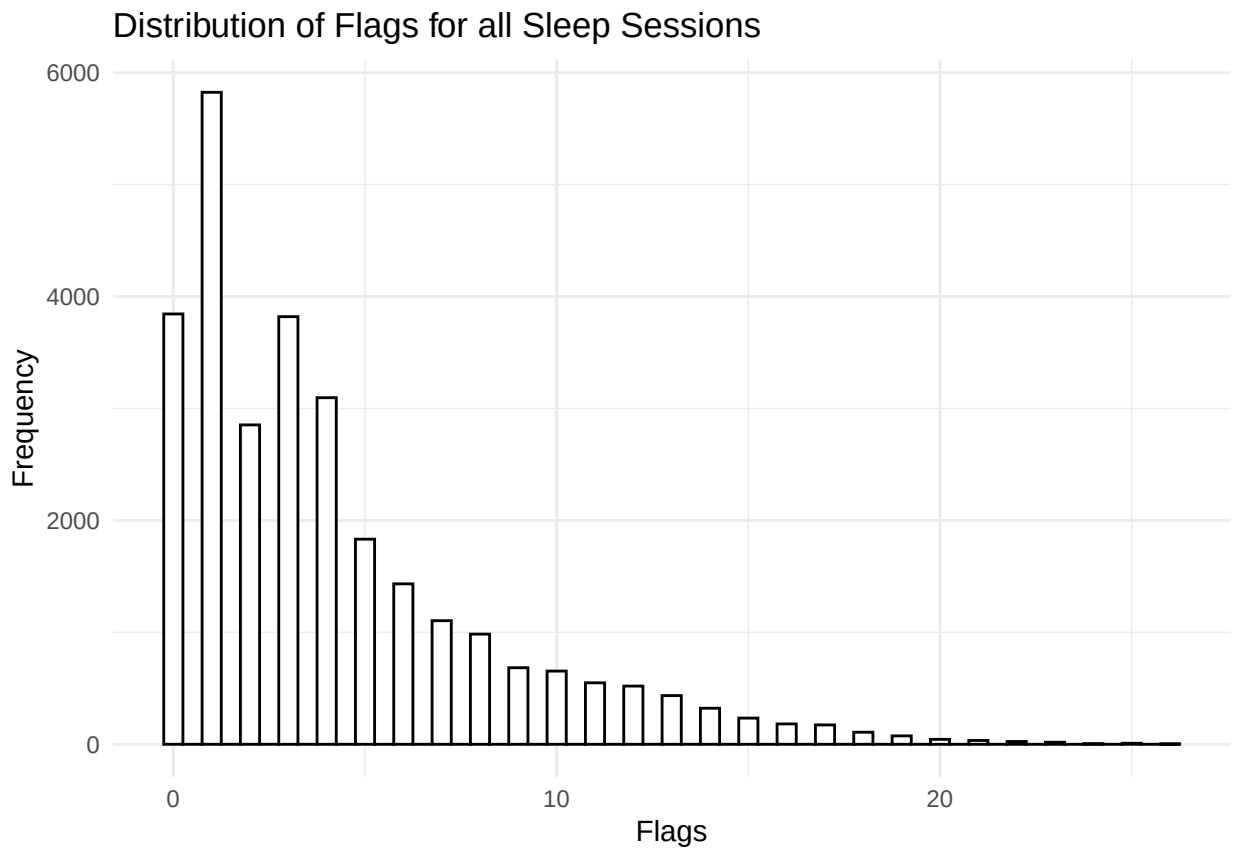
```
cust_dev_joined %>% group_by(device_name) %>% summarise(n = n(), mean = mean(flags*60/duration), median
```

```
## # A tibble: 10 x 6
##    device_name      n  mean median    sd   var
##    <chr>        <int> <dbl>  <dbl> <dbl> <dbl>
##  1 Active Alpha  3296 0.747  0.566 0.673 0.453
##  2 Advance       3005 0.751  0.538 0.732 0.536
##  3 Advance 2     9966 0.702  0.470 0.691 0.477
##  4 iDOL           154 0.473  0.442 0.344 0.118
##  5 Run 7           23 2.11   1.96  0.684 0.468
##  6 Run 7 Plus      45 0.433  0.333 0.364 0.133
##  7 Run 875       1147 0.501  0.326 0.570 0.325
##  8 Run 875 X      179 1.18   0.816 1.05  1.09
##  9 Run BE        5731 0.591  0.429 0.626 0.391
## 10 Run ON        5319 0.682  0.455 0.714 0.510
```

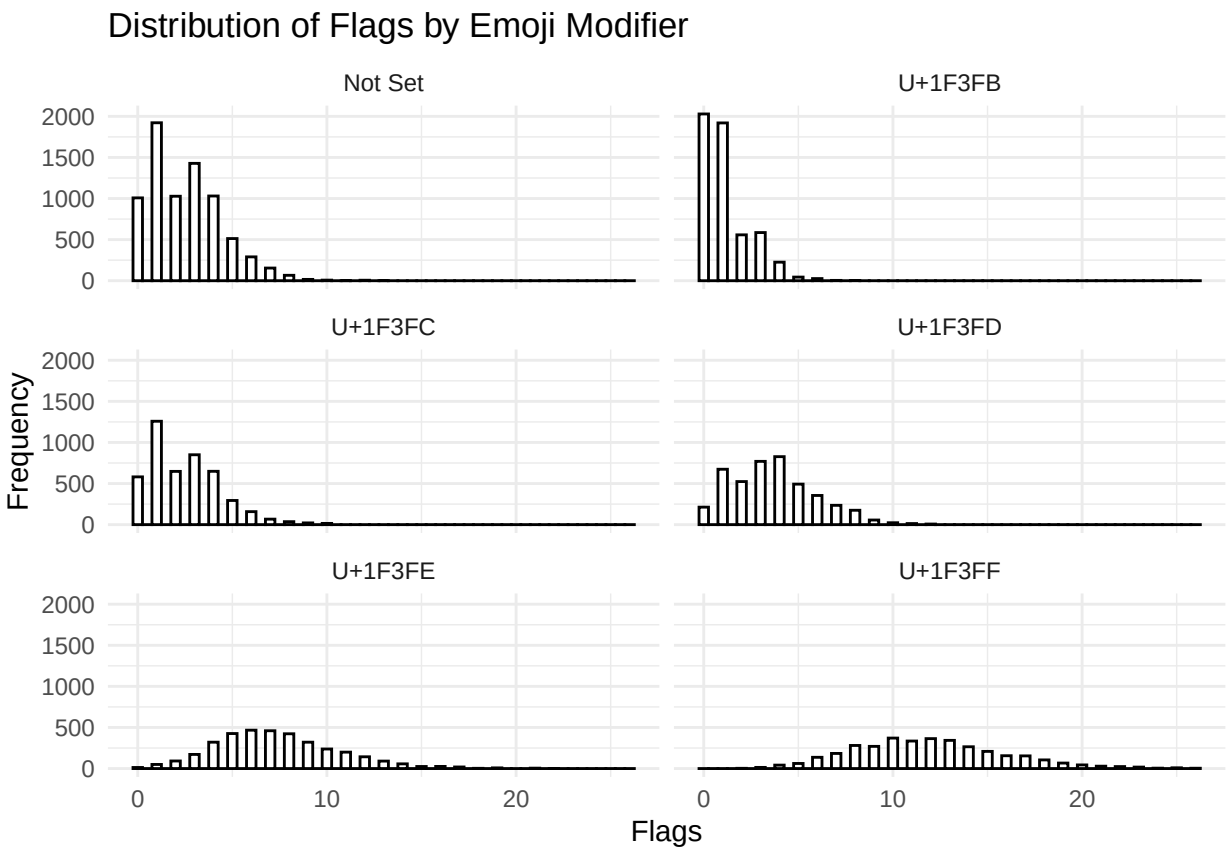
```
cust_dev_joined %>% mutate(income_bracket = cut(hhld_median_inc, breaks = seq(40000, 115000, by=5000)))
```

```
## # A tibble: 15 x 6
##    income_bracket         n  mean median    sd    var
##    <fct>              <int> <dbl>  <dbl> <dbl>  <dbl>
##  1 (4e+04,4.5e+04]       19 0.410  0.338 0.324 0.105
##  2 (4.5e+04,5e+04]       42 1.01   0.952 0.407 0.166
##  3 (5e+04,5.5e+04]     4646 1.00   0.817 0.784 0.615
##  4 (5.5e+04,6e+04]      974 1.14   0.963 0.796 0.634
##  5 (6e+04,6.5e+04]      700 0.685  0.388 0.772 0.596
##  6 (6.5e+04,7e+04]    10939 0.745  0.537 0.716 0.513
##  7 (7e+04,7.5e+04]      819 0.676  0.424 0.686 0.470
##  8 (7.5e+04,8e+04]      910 0.517  0.426 0.489 0.239
##  9 (8e+04,8.5e+04]     1130 0.426  0.336 0.486 0.236
## 10 (8.5e+04,9e+04]     4443 0.424  0.347 0.443 0.197
## 11 (9e+04,9.5e+04]      237 0.440  0.257 0.439 0.193
## 12 (9.5e+04,1e+05]     3468 0.416  0.327 0.441 0.195
## 13 (1e+05,1.05e+05]      69 0.360  0.333 0.313 0.0981
## 14 (1.05e+05,1.1e+05]   351 0.249  0.147 0.285 0.0810
## 15 (1.1e+05,1.15e+05]   118 1.48   1.61  0.914 0.836
```

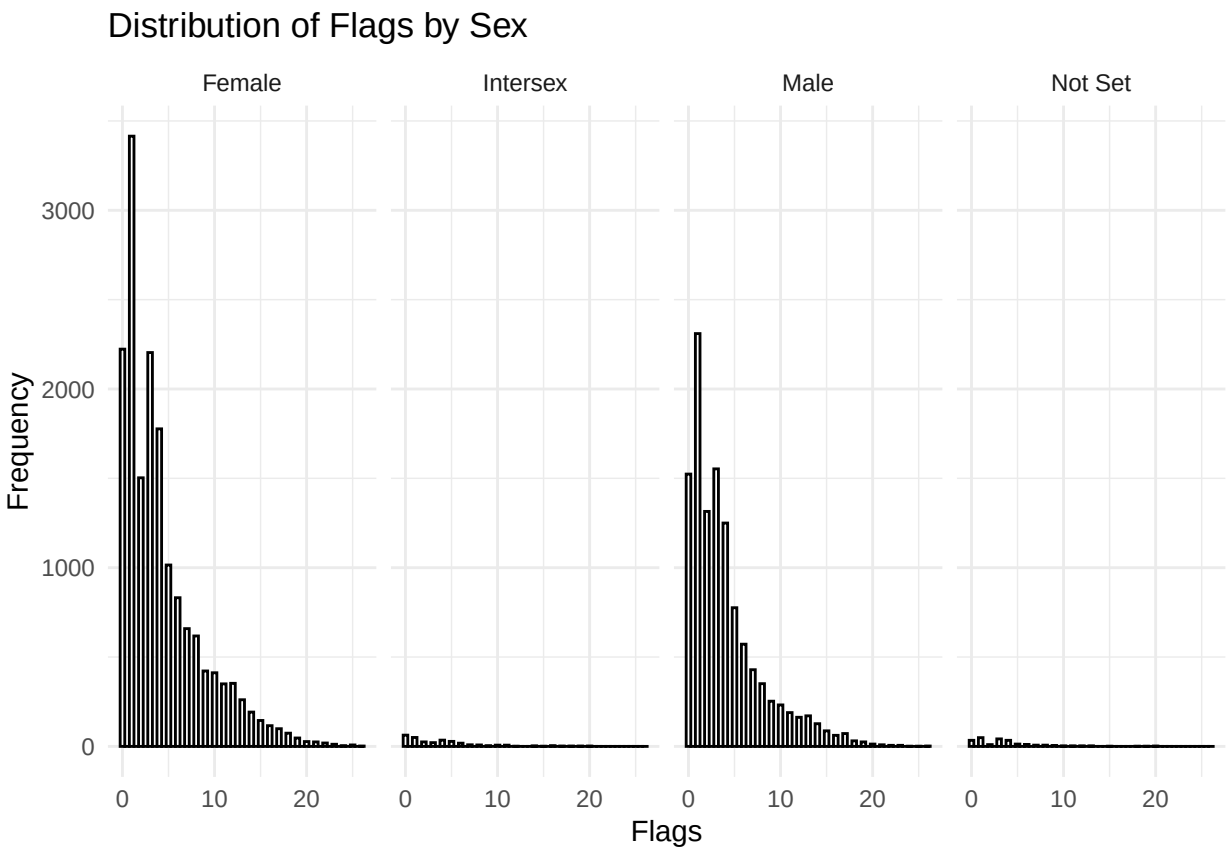
```
#Visualize flags distribution through histograms
cust_dev_joined %>% ggplot(aes(x = flags)) + geom_histogram(colour="black", fill="white", binwidth=0.5)
```



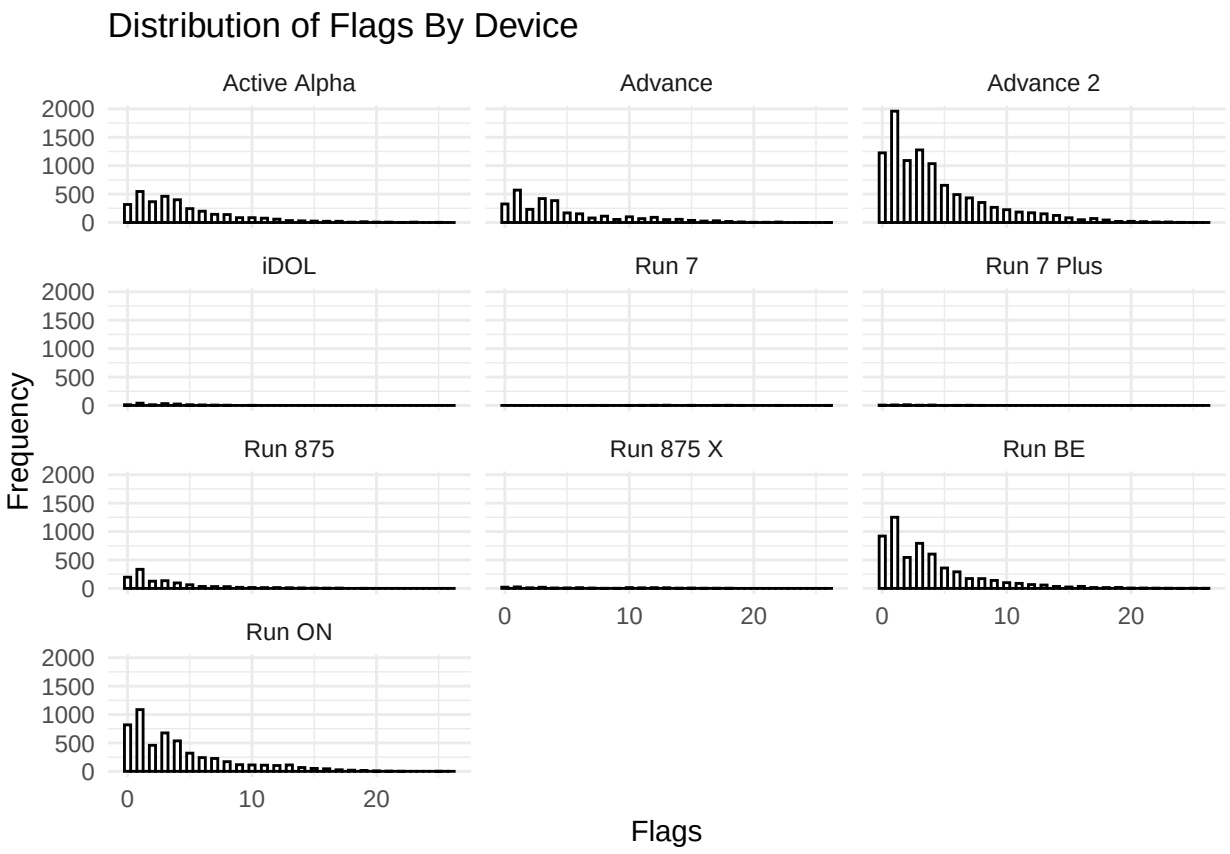

```
#By emoji_modifier
cust_dev_joined %>% ggplot(aes(x = flags)) + geom_histogram(colour="black", fill="white", binwidth=0.5)
```



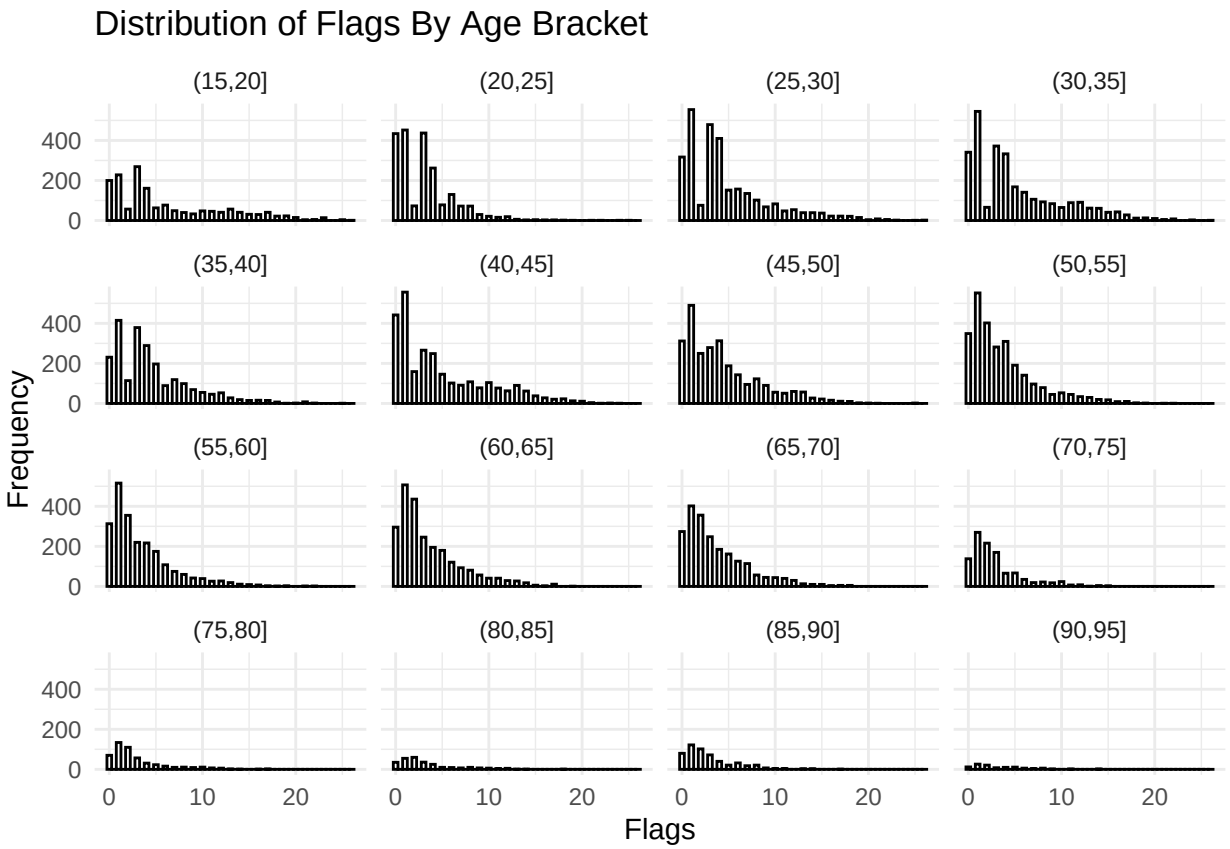

```
#By sex
cust_dev_joined %>% ggplot(aes(x = flags)) + geom_histogram(colour="black", fill="white", binwidth=0.5)
```



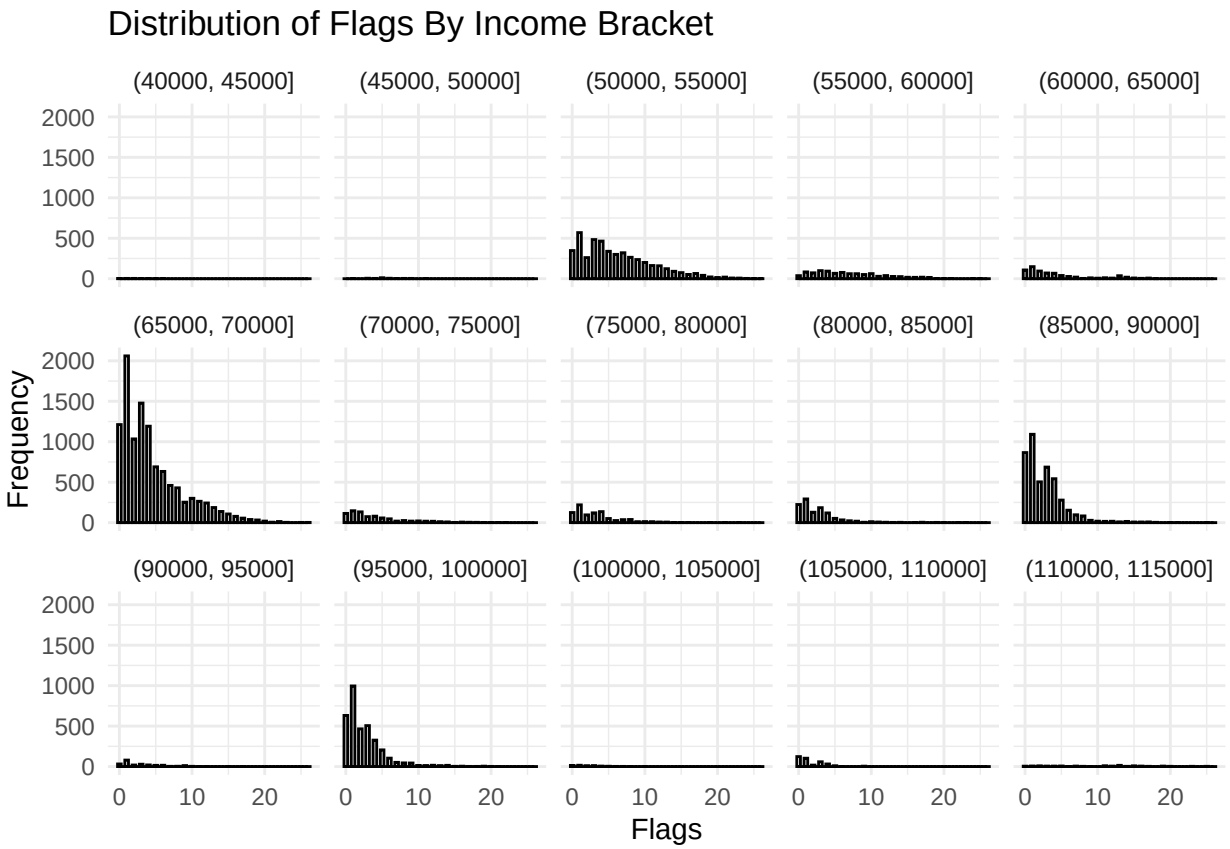

```
#By device
cust_dev_joined %>% ggplot(aes(x = flags)) + geom_histogram(colour="black", fill="white", binwidth=0.5)
```




```
#By age
cust_dev_joined %>% mutate(age_bracket = cut(age, breaks = seq(15, 95, by=5))) %>% ggplot(aes(x = flags
```

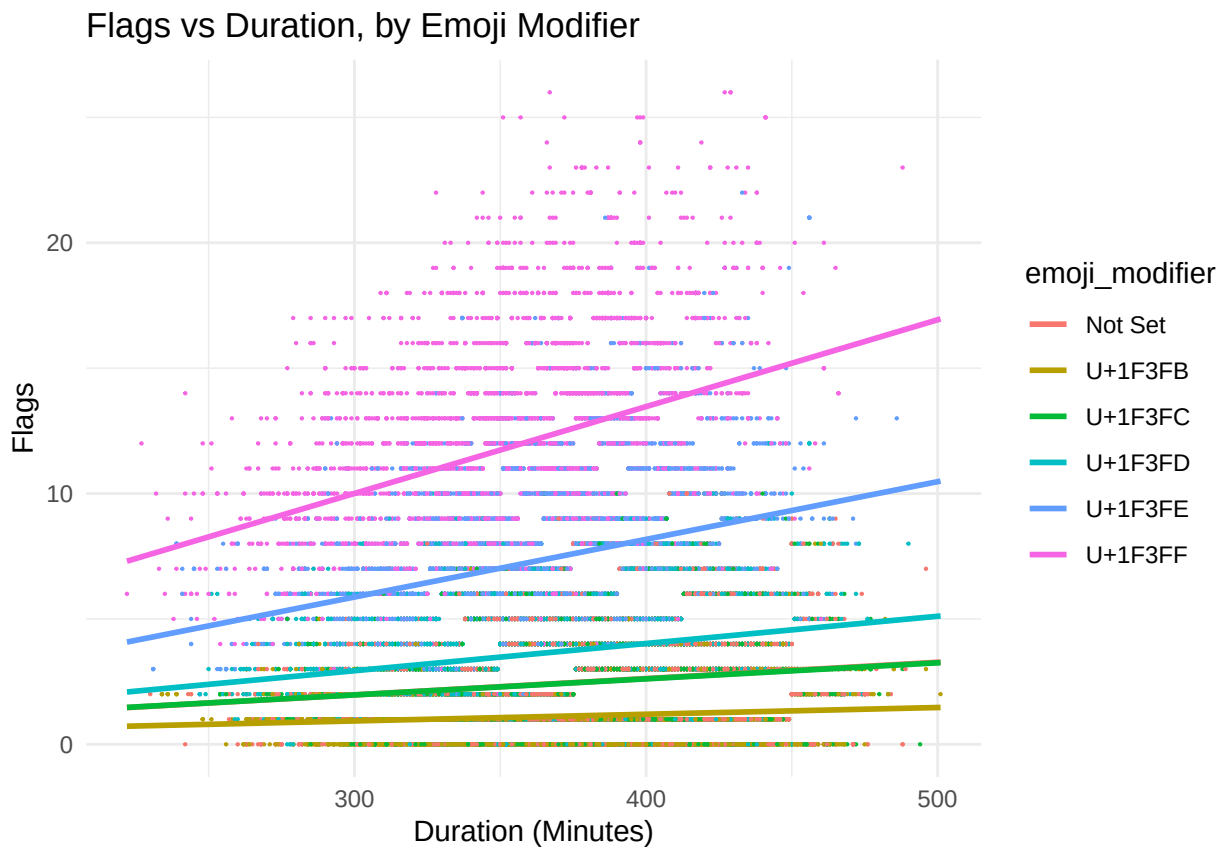



```
#By income_bracket
cust_dev_joined %>% mutate(income_bracket = cut(hhld_median_inc, breaks = seq(40000, 115000, by=5000), l
```



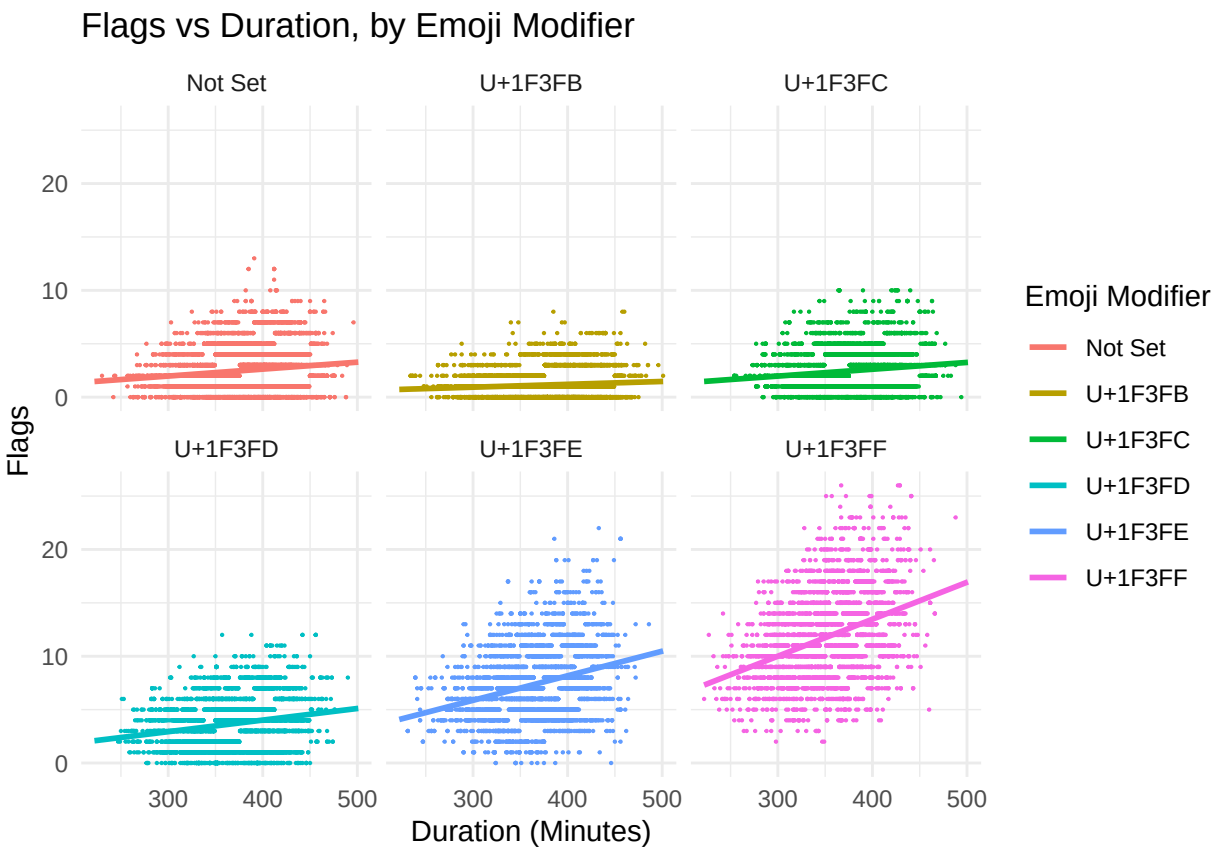

```
#Plot with variables duration and flags to identify any correlation between them and other variables
#Emoji modifiers from B to F is from light to dark skin tone
cust_dev_joined %>% group_by(emoji_modifier) %>% ggplot(aes(x=duration, y=flags, colour = emoji_modifie
```

```
## `geom_smooth()` using formula 'y ~ x'
```



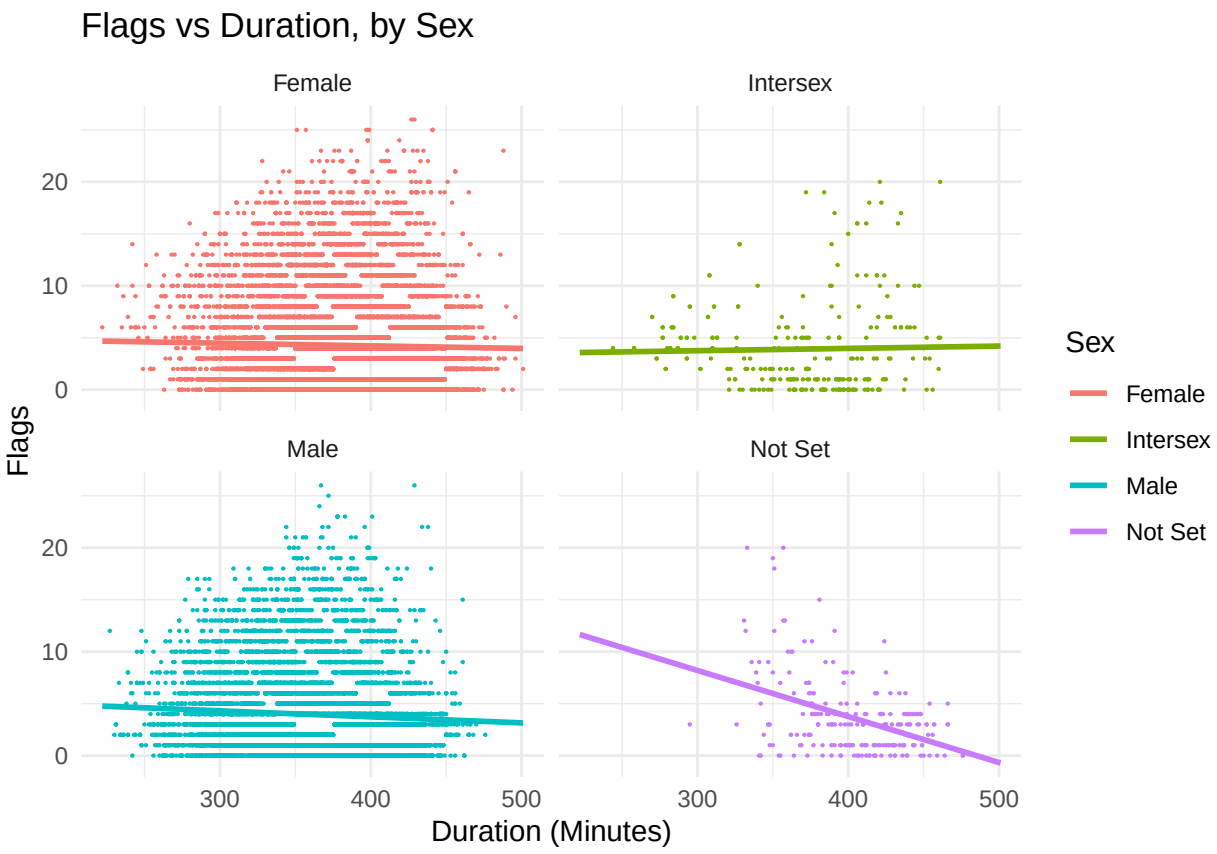

```
#Separated by emoji_modifer
cust_dev_joined %>% group_by(emoji_modifier) %>% ggplot(aes(x=duration, y=flags, colour = emoji_modifie
```

```
## `geom_smooth()` using formula 'y ~ x'
```



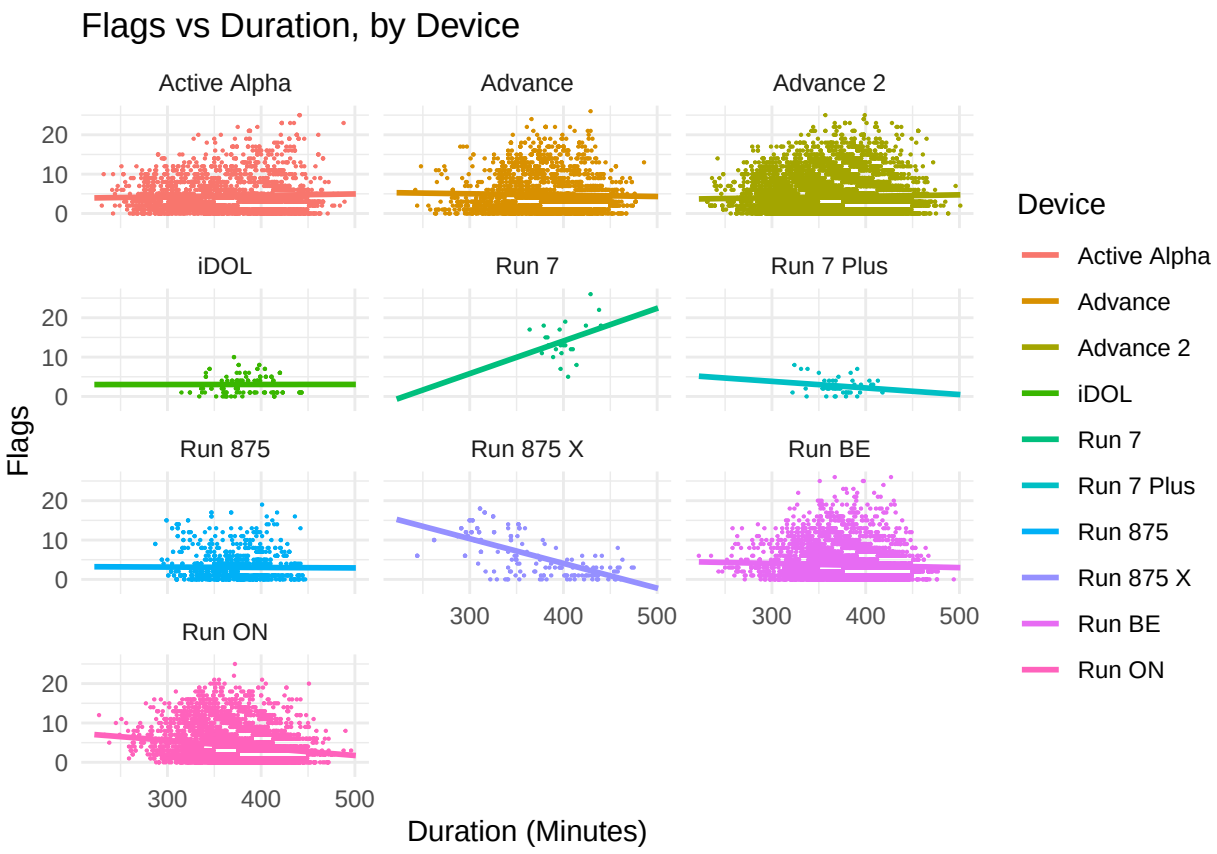

```
#Separated by sex
cust_dev_joined %>% group_by(sex) %>% ggplot(aes(x=duration, y=flags, colour = sex)) + geom_point(size=0
```

```
## `geom_smooth()` using formula 'y ~ x'
```



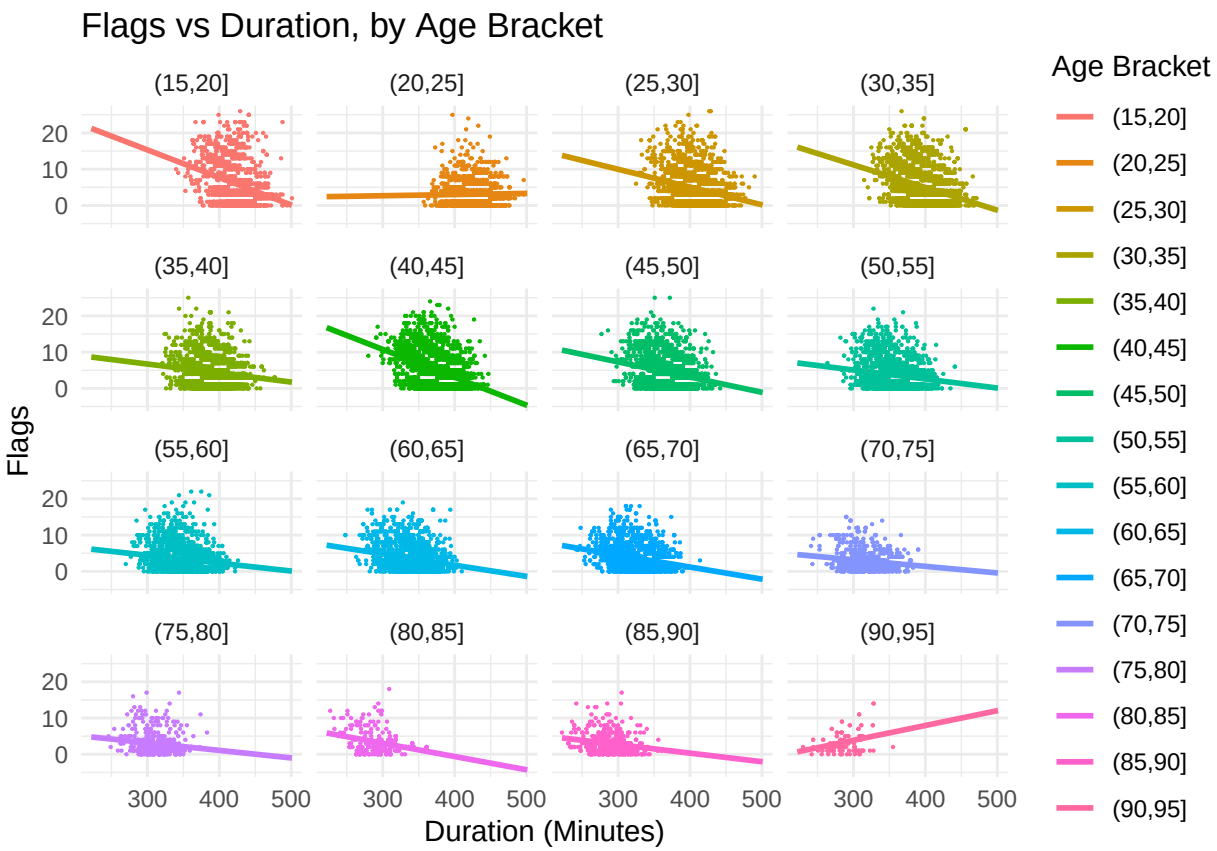

```
#Separated by device_name
cust_dev_joined %>% group_by(device_name) %>% ggplot(aes(x=duration, y=flags, colour = device_name)) + g
```

```
## `geom_smooth()` using formula 'y ~ x'
```



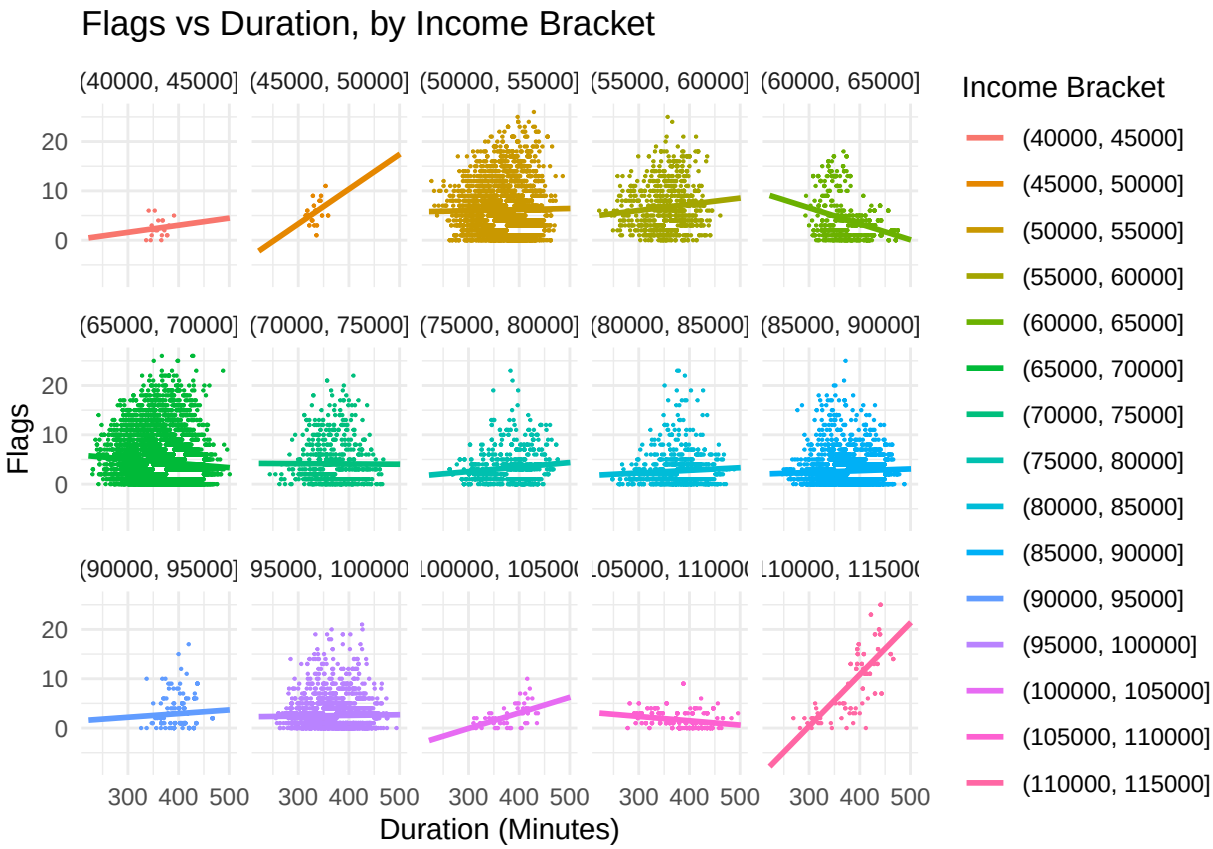

```
#Separated by age
cust_dev_joined %>% mutate(age_bracket = cut(age, breaks = seq(15, 95, by=5))) %>% group_by(age_bracket
```

```
## `geom_smooth()` using formula 'y ~ x'
```



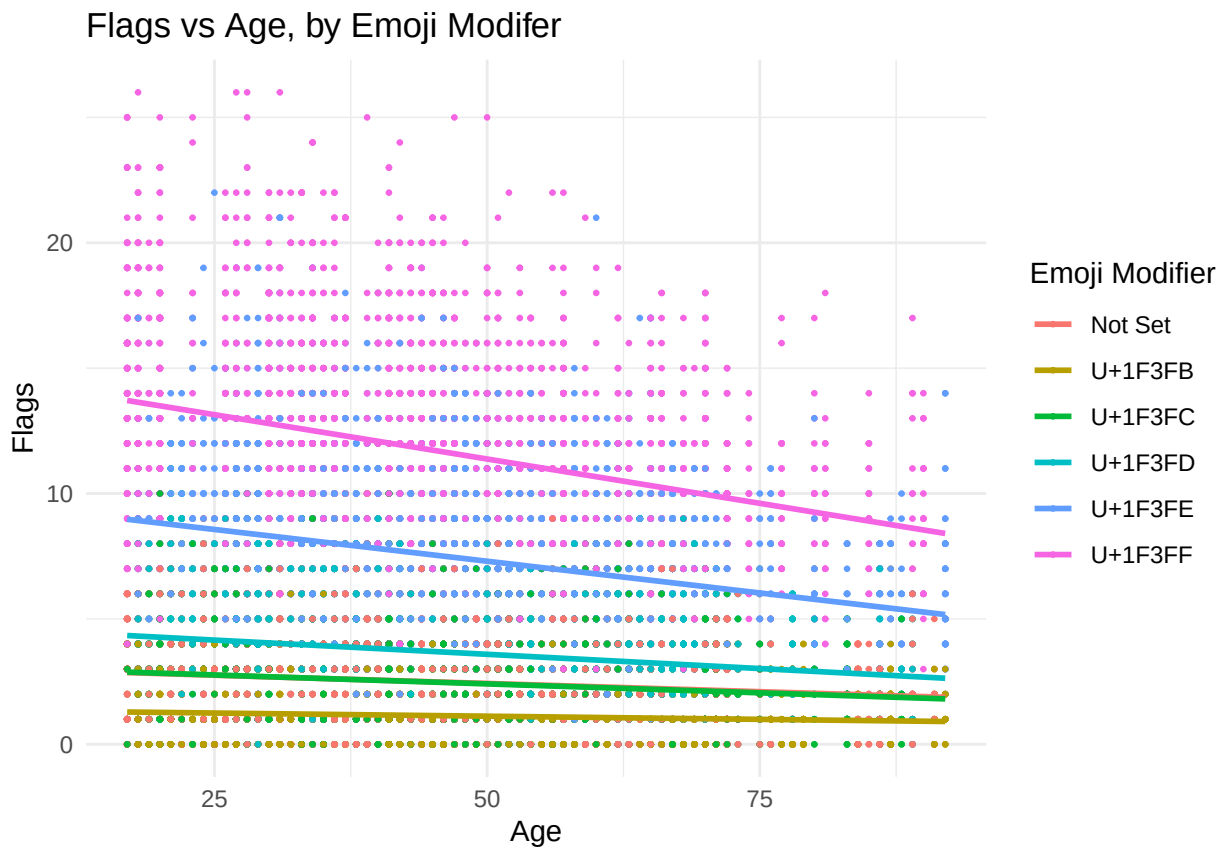

```
#Separated by income
cust_dev_joined %>% mutate(income_bracket = cut(hhld_median_inc, breaks = seq(40000, 115000, by=5000), l
```

```
## `geom_smooth()` using formula 'y ~ x'
```



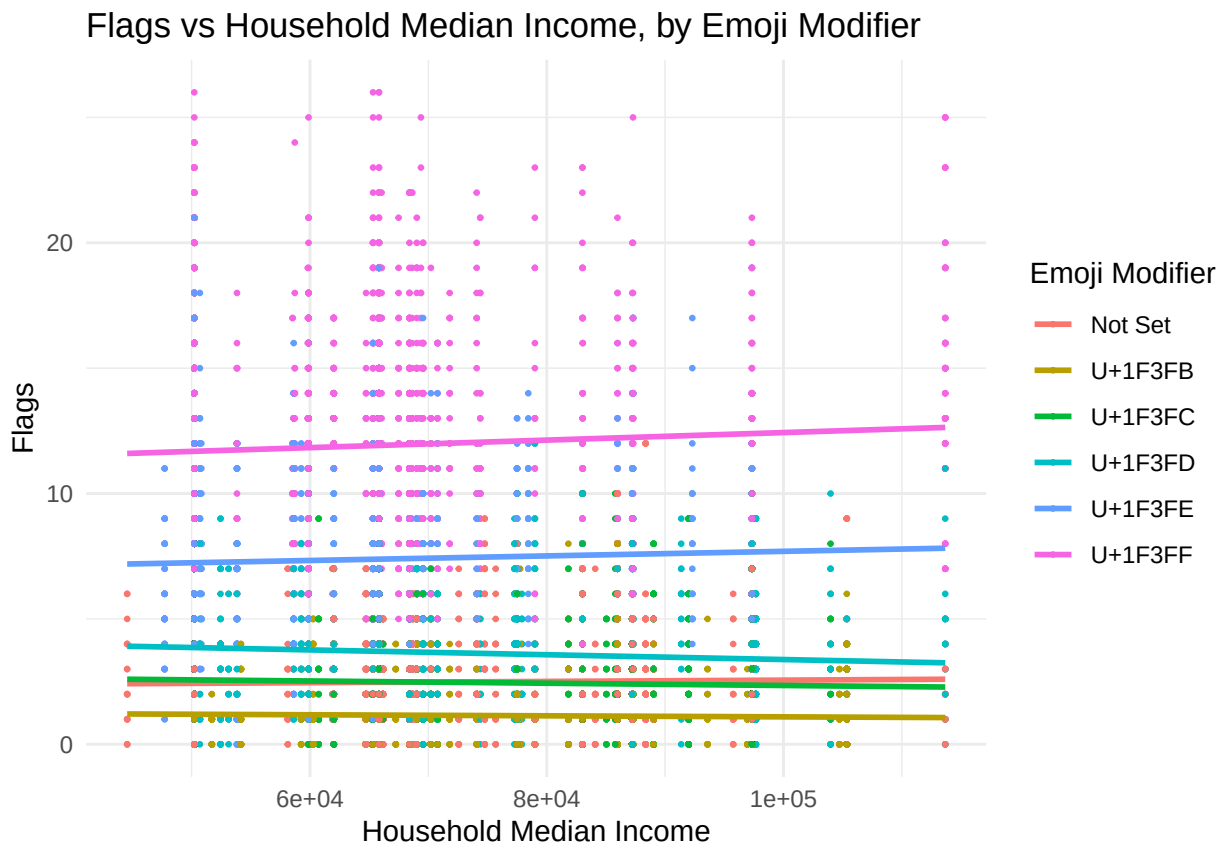

```
#Checking how age affects flags for different emoji_modifiers
cust_dev_joined %>% group_by(emoji_modifier) %>% ggplot(aes(x=age, y=flags, colour = emoji_modifier)) +
```

```
## `geom_smooth()` using formula 'y ~ x'
```



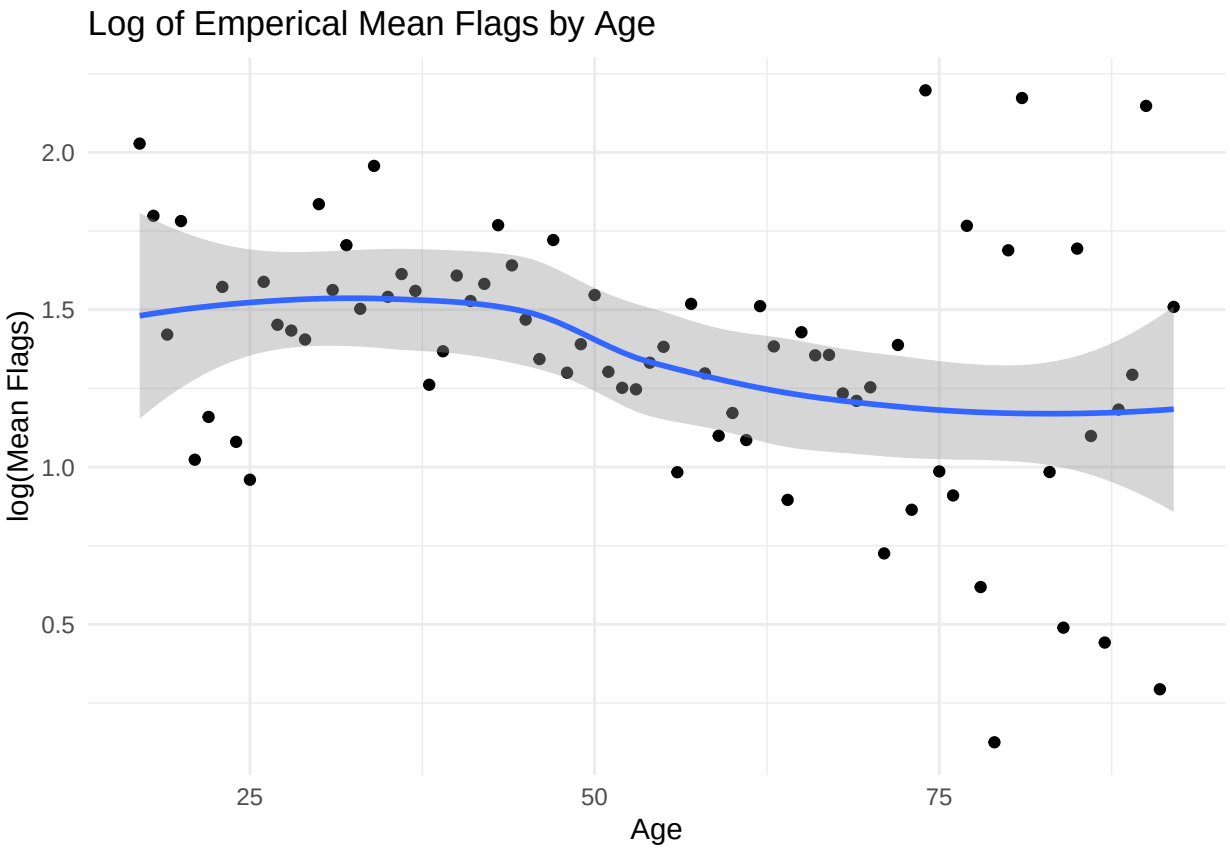

```
#Checking how income affects flags for different emoji_modifiers
cust_dev_joined %>% group_by(emoji_modifier) %>% ggplot(aes(x=hhld_median_inc, y=flags, colour = emoji_m
```

```
## `geom_smooth()` using formula 'y ~ x'
```



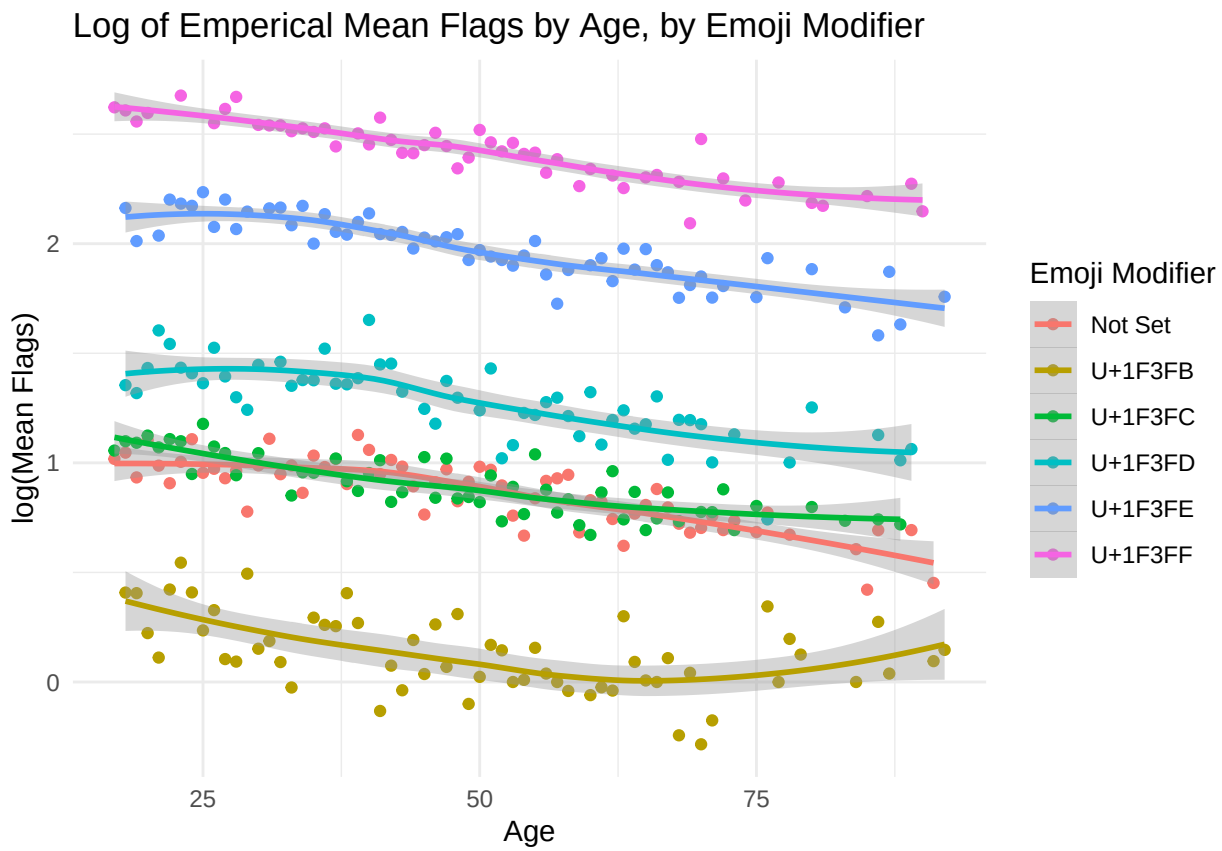

```
#Plotting log means of flags by age to check for linearity
#Interesting behaviour, weakly linear
cust_dev_joined %>% group_by(age) %>% summarise(flags=mean(flags)) %>% ggplot(aes(x=age, y=log(flags)))
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



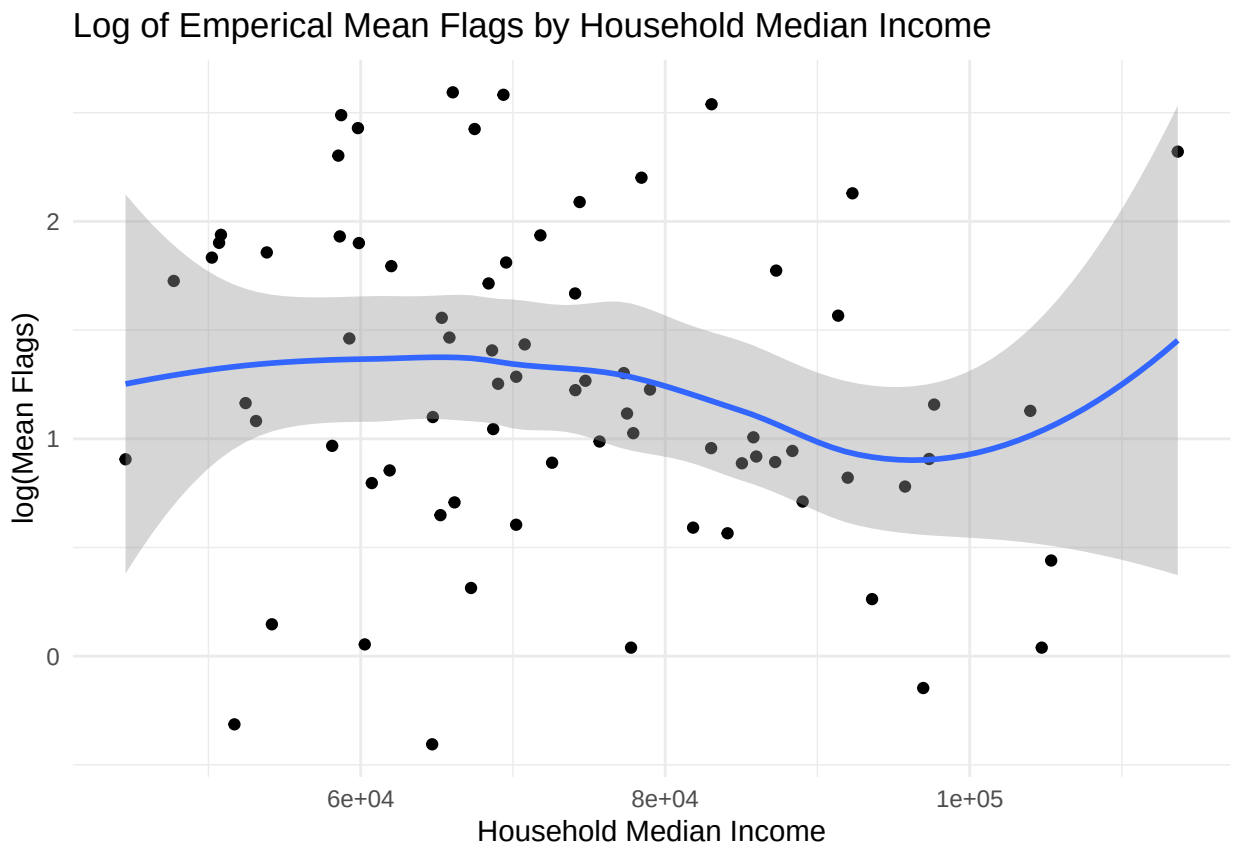

```
cust_dev_joined %>% group_by(age, emoji_modifier) %>% summarise(flags=mean(flags)) %>% ggplot(aes(x=age
```

```
## `summarise()` has grouped output by 'age'. You can override using the `.groups`
## argument.
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



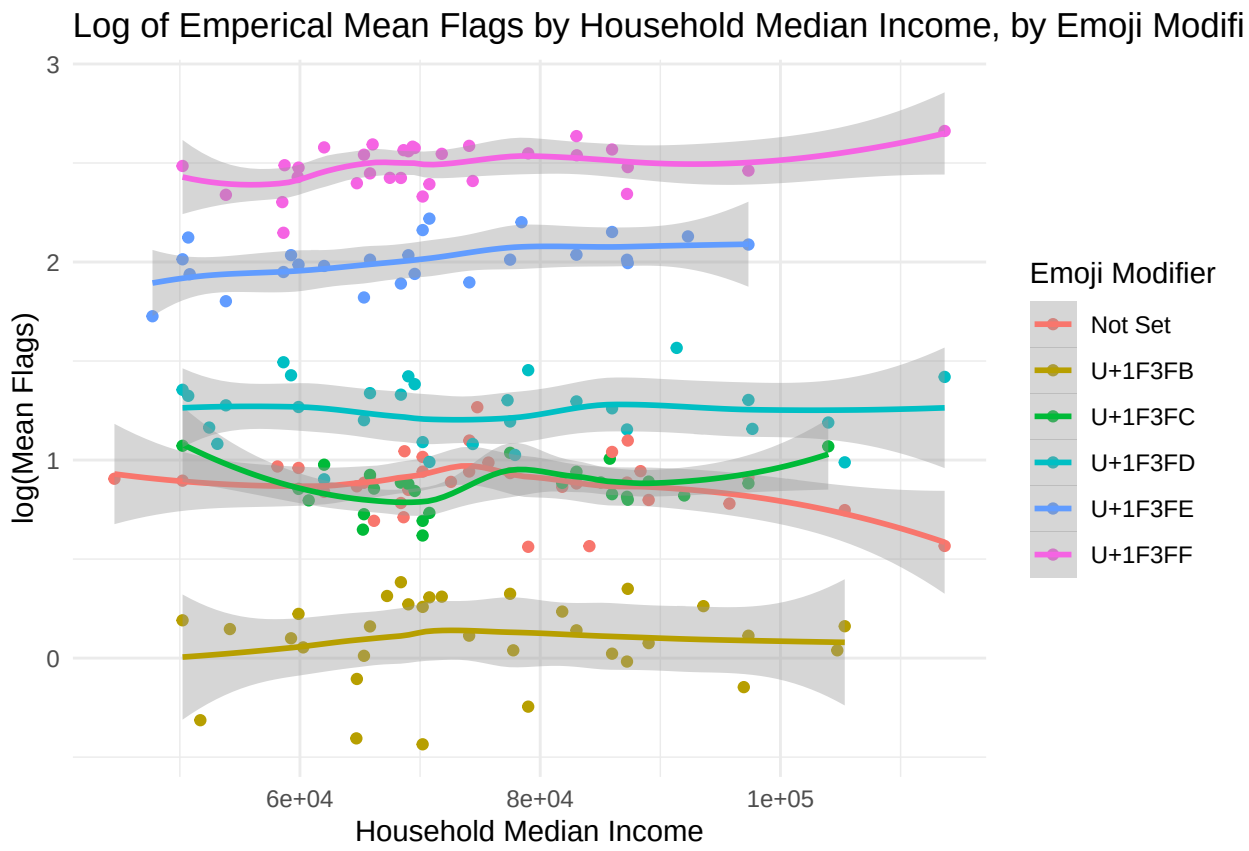

```
#Plotting log means of flags by income to check for linearity
#Interesting behaviour, weakly linear
cust_dev_joined %>% group_by(hhld_median_inc) %>% summarise(flags=mean(flags)) %>% ggplot(aes(x=hhld_me
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```




```
cust_dev_joined %>% group_by(hhld_median_inc, emoji_modifier) %>% summarise(flags=mean(flags)) %>% ggpl
```

```
## `summarise()` has grouped output by 'hhld_median_inc'. You can override using
## the `.groups` argument.
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```




```
#First attempt at a model
#After testing other models later in the code, this one is just better
model_base <- lme4::glmer(flags ~ factor(emoji_modifier) + (1|cust_id), family = poisson, offset = log(
summary(model_base)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: poisson  ( log )
## Formula: flags ~ factor(emoji_modifier) + (1 | cust_id)
##    Data: cust_dev_joined
##  Offset: log(duration/60)
##
##      AIC      BIC   logLik deviance df.resid
## 119508.7 119566.6 -59747.4 119494.7    28858
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -2.8730 -0.9353 -0.1131  0.7183  6.3671
##
## Random effects:
##  Groups  Name        Variance Std.Dev.
##  cust_id (Intercept) 0.007514 0.08668
## Number of obs: 28865, groups:  cust_id, 974
##
## Fixed effects:
##                                Estimate Std. Error z value Pr(>|z|)
```

```
## (Intercept)                    -0.939131   0.009574 -98.087   <2e-16 ***
## factor(emoji_modifier)U+1F3FB -0.764462   0.017727 -43.124   <2e-16 ***
## factor(emoji_modifier)U+1F3FC  0.008169   0.015641   0.522    0.602
## factor(emoji_modifier)U+1F3FD  0.421129   0.014821  28.415   <2e-16 ***
## factor(emoji_modifier)U+1F3FE  1.127640   0.013777  81.851   <2e-16 ***
## factor(emoji_modifier)U+1F3FF  1.632267   0.013396 121.852   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) f(_)U+1F3FB f(_)U+1F3FC f(_)U+1F3FD f(_)U+1F3FE
## f(_)U+1F3FB -0.540
## f(_)U+1F3FC -0.612  0.331
## f(_)U+1F3FD -0.644  0.348       0.394
## f(_)U+1F3FE -0.694  0.375       0.425       0.448
## f(_)U+1F3FF -0.714  0.386       0.437       0.460       0.496
```

```
lme4::VarCorr(model_base)
```

```
##  Groups  Name        Std.Dev.
##  cust_id (Intercept) 0.086682
```

```
#Other models to be considered

#0 coefficient models
model_0a <- lme4::glmer(flags ~ (1|cust_id), family = poisson, offset = log(duration/60), data = cust_d
model_0b <- lme4::glmer(flags ~ (1|device_name), family = poisson, offset = log(duration/60), data = cus
model_0c <- lme4::glmer(flags ~ (1|cust_id) + (1|device_name), family = poisson, offset = log(duration/
```

```
## boundary (singular) fit: see ?isSingular
```

```
#ANOVA shows model with only cust_id random intercept is the best
anova(model_0a, model_base, test = "Chisq")
```

```
## Data: cust_dev_joined
## Models:
## model_0a: flags ~ (1 | cust_id)
## model_base: flags ~ factor(emoji_modifier) + (1 | cust_id)
##            npar    AIC    BIC logLik deviance  Chisq Df Pr(>Chisq)
## model_0a      2 122709 122725 -61352   122705
## model_base    7 119509 119567 -59747   119495 3210.2  5  < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
anova(model_0b, model_base, test = "Chisq")
```

```
## Data: cust_dev_joined
## Models:
## model_0b: flags ~ (1 | device_name)
## model_base: flags ~ factor(emoji_modifier) + (1 | cust_id)
##            npar    AIC    BIC logLik deviance Chisq Df Pr(>Chisq)
## model_0b      2 189214 189231 -94605   189210
## model_base    7 119509 119567 -59747   119495 69716  5  < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
anova(model_0c, model_base, test = "Chisq")
```

```
## Data: cust_dev_joined
## Models:
## model_0c: flags ~ (1 | cust_id) + (1 | device_name)
## model_base: flags ~ factor(emoji_modifier) + (1 | cust_id)
##            npar    AIC    BIC logLik deviance  Chisq Df Pr(>Chisq)
## model_0c      3 122711 122736 -61352   122705
## model_base    7 119509 119567 -59747   119495 3210.2  4  < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
#With device_name as random intercept
model_0d <- lme4::glmer(flags ~ factor(emoji_modifier) + (1|cust_id) + (1|device_name), family = poisso
#ANOVA shows device_name does not significantly improve the model
anova(model_0d, model_base, test = "Chisq")
```

```
## Data: cust_dev_joined
## Models:
## model_base: flags ~ factor(emoji_modifier) + (1 | cust_id)
## model_0d: flags ~ factor(emoji_modifier) + (1 | cust_id) + (1 | device_name)
##            npar    AIC    BIC logLik deviance  Chisq Df Pr(>Chisq)
## model_base    7 119509 119567 -59747   119495
## model_0d      8 119511 119577 -59747   119495 0.0109  1      0.917
```

```r
#1 coefficient models
model_null <- glm(flags ~ 1, family = poisson, offset = log(duration/60), data = cust_dev_joined)
model_1a <- glm(flags ~ age, family = poisson, offset = log(duration/60), data = cust_dev_joined)
model_1b <- glm(flags ~ hhld_median_inc, family = poisson, offset = log(duration/60), data = cust_dev_j
```

```r
#These are significant in the absence of our cust_id random intercept, but since we do have the interce
#So ignore this chunk pretty much
anova(model_null, model_1a, test = "Chisq")
```

```
## Analysis of Deviance Table
##
## Model 1: flags ~ 1
## Model 2: flags ~ age
##   Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
## 1     28864     112800
## 2     28863     112611  1   188.87 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
anova(model_null, model_1b, test = "Chisq")
```

```
## Analysis of Deviance Table
##
## Model 1: flags ~ 1
## Model 2: flags ~ hhld_median_inc
##   Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
## 1     28864     112800
## 2     28863     102509  1    10291 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
model_2a <- lme4::glmer(flags ~ factor(emoji_modifier) + factor(sex) + (1|cust_id), family = poisson, o

anova(model_base, model_2a)
```

```
## Data: cust_dev_joined
## Models:
## model_base: flags ~ factor(emoji_modifier) + (1 | cust_id)
## model_2a: flags ~ factor(emoji_modifier) + factor(sex) + (1 | cust_id)
##            npar    AIC    BIC logLik deviance  Chisq Df Pr(>Chisq)
## model_base    7 119509 119567 -59747   119495
## model_2a     10 119509 119592 -59745   119489 5.5896  3     0.1334
```

```
c_interval <- confint(model_base)
```

```
## Computing profile confidence intervals ...
```

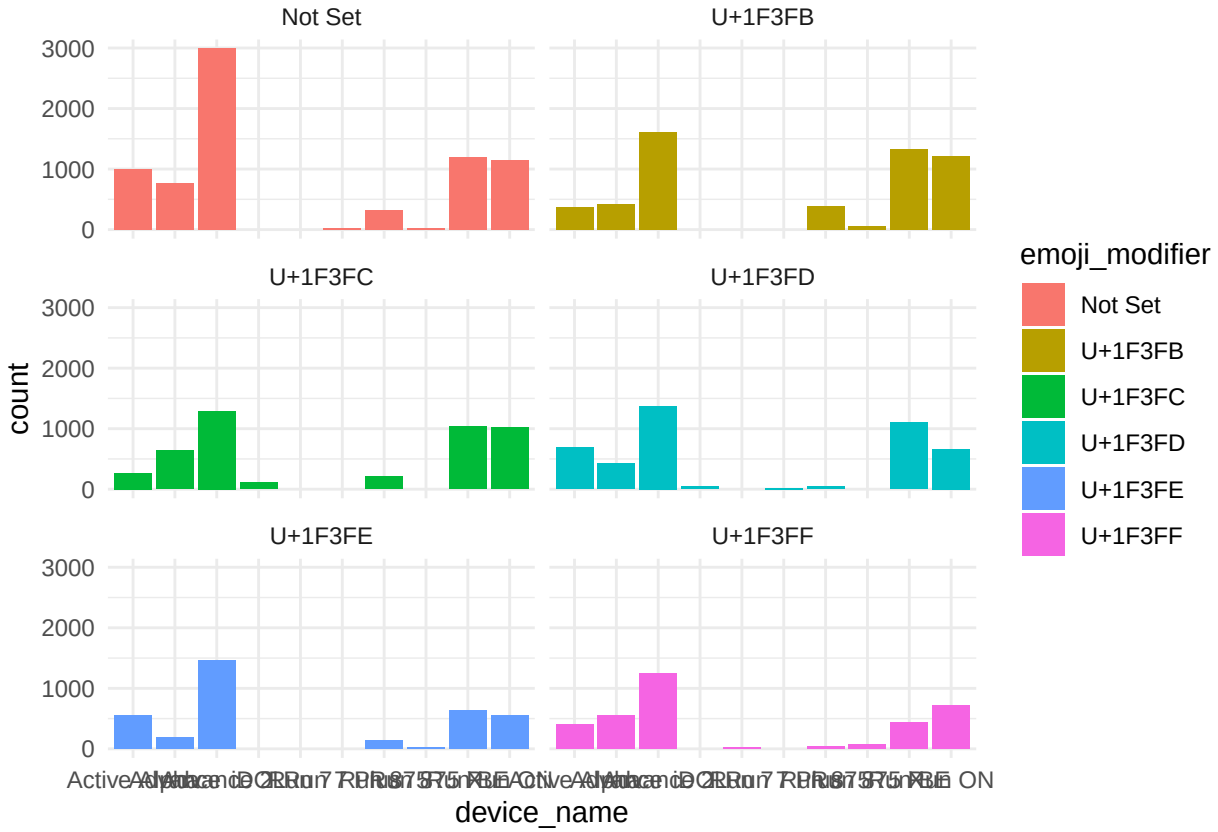
```
exp(c_interval)
```

```
##                                    2.5 %    97.5 %
## .sig01                         1.0808805 1.1007819
## (Intercept)                    0.3836708 0.3983551
## factor(emoji_modifier)U+1F3FB 0.4496671 0.4820390
## factor(emoji_modifier)U+1F3FC 0.9777609 1.0396327
## factor(emoji_modifier)U+1F3FD 1.4799872 1.5686029
## factor(emoji_modifier)U+1F3FE 3.0060933 3.1731116
## factor(emoji_modifier)U+1F3FF 4.9828932 5.2519534
```

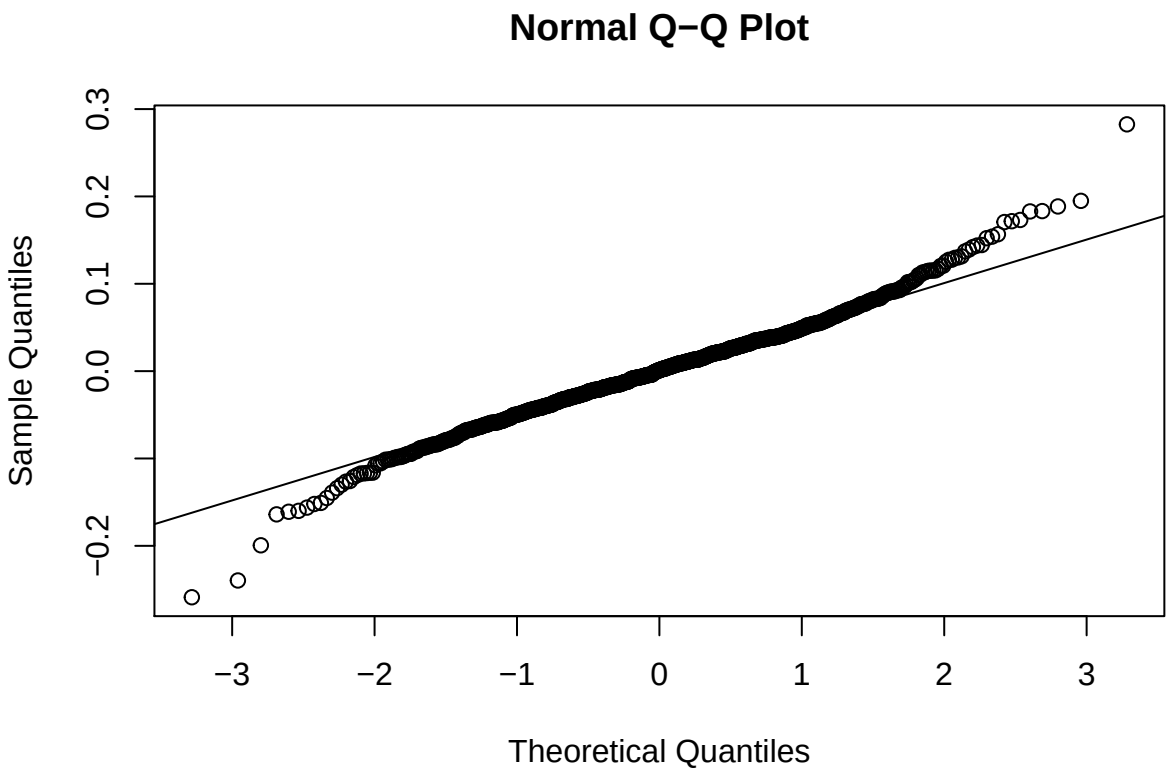
```
ests <- format(round(exp(summary(model_base)$coeff)[,1], 2), nsmall = 2)
cis <- format(round(exp(c_interval),2)[-1,], nsmall = 2)
conf_ints <- str_c("(", trimws(cis[,1]), ", ", cis[,2], ")")
rownames_for_table <- c("Baseline rate", "U+1F3FB", "U+1F3FC", "U+1F3FD", "U+1F3FE", "U+1F3FF")
colnames_for_table <- c("Estimate", "95% CI")
table <- cbind(ests, conf_ints)
rownames(table) <- rownames_for_table
colnames(table) <- colnames_for_table
knitr::kable(table, align = c("r", "r"))
```

	Estimate	95% CI
Baseline rate	0.39	(0.38, 0.40)
U+1F3FB	0.47	(0.45, 0.48)
U+1F3FC	1.01	(0.98, 1.04)
U+1F3FD	1.52	(1.48, 1.57)
U+1F3FE	3.09	(3.01, 3.17)
U+1F3FF	5.12	(4.98, 5.25)

```
cust_dev_joined %>% group_by(emoji_modifier) %>% ggplot(aes(x=device_name, fill=emoji_modifier)) + geom
```



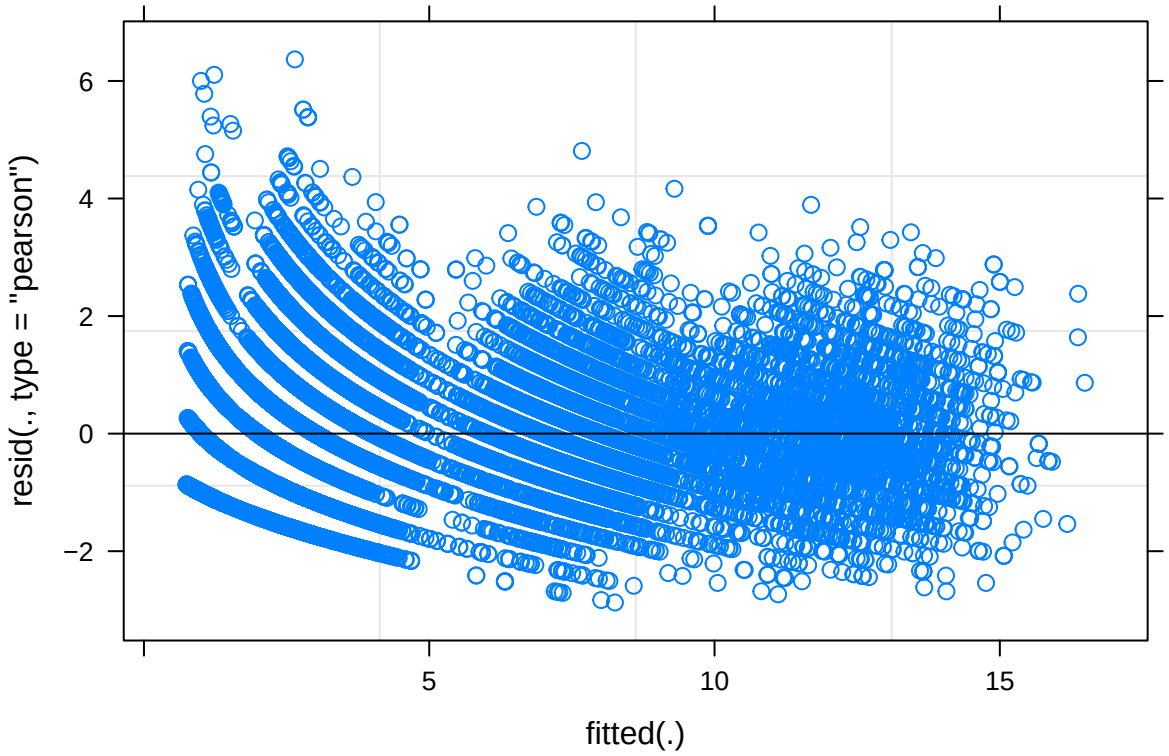

```
res_int<- ranef(model_base)$cust_id$`(Intercept)`
qqnorm(res_int)
qqline(res_int)
```




```
shapiro.test(res_int)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  res_int
## W = 0.98361, p-value = 5.388e-09
```

```
plot(model_base)
```


resid(., type = "pearson")
6
4
2
0
−2
5
10
15
fitted(.)