brainstation

Prince

8/10/2020

```
rm(list = ls())
library(ggplot2)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(stringr)
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

```
## The following object is masked from 'package:base':
##
##     date
```

```
trial = read.csv('table_1.csv', sep = ',', quote = "\"" )    #lets work with this one
trial$Name <- as.character(trial$Name)
trial$Sub_category_id <- as.numeric(as.character(trial$Sub_category_id))
```

```
## Warning: NAs introduced by coercion
```

```
trial <- na.omit(trial)
currency <- read.csv('currency.csv')
countries <- read.csv('countries.csv', sep = ',')
write.csv(trial, 'BS.csv', col.names = TRUE)     #export this to Python...
```

```
## Warning in write.csv(trial, "BS.csv", col.names = TRUE): attempt to set
## 'col.names' ignored
```

```
#convert factors to numbers
n = vector()
numbers <- function(x) {
  for (i in 1:length(x))  {
```

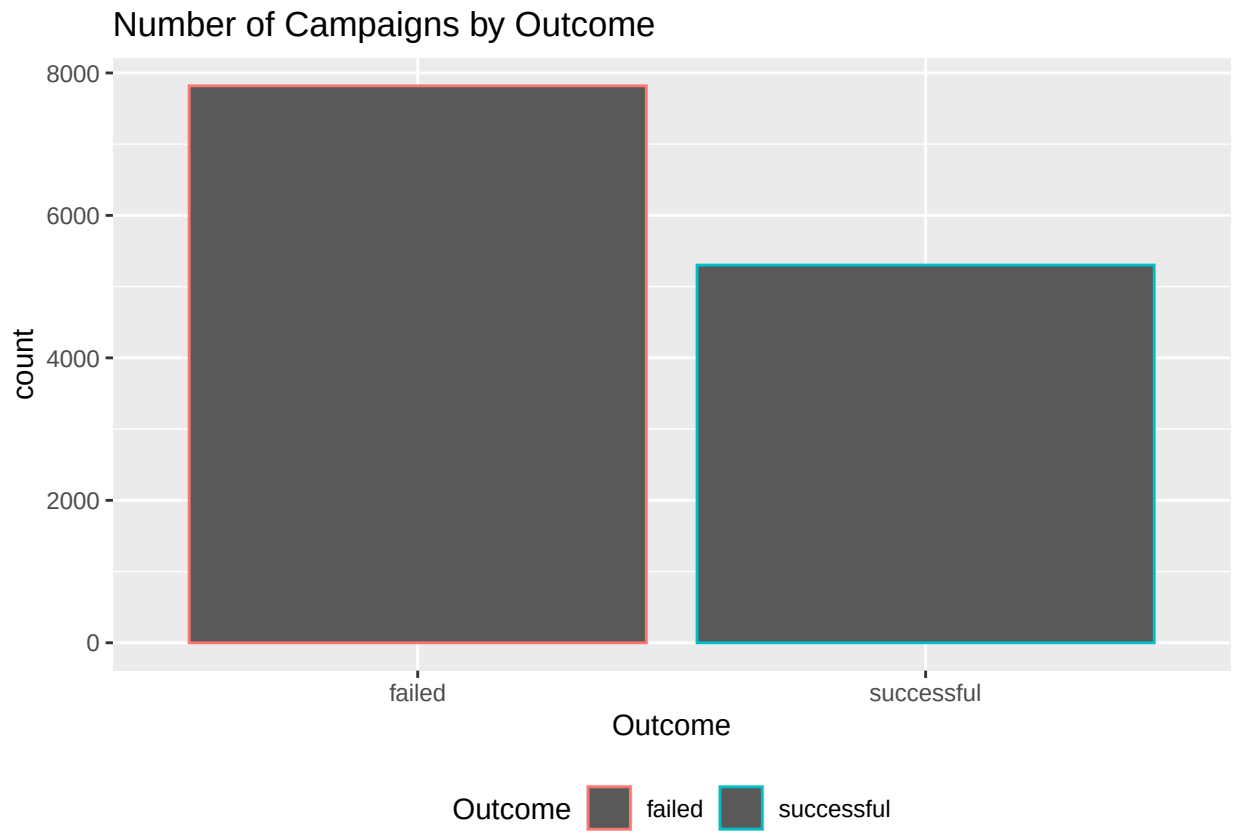
```
 n[[i]] <- as.numeric(as.character(x[i]))
  }
  return(n)
}
```

```
trial[c(7,8,9,12,3)] <- apply(trial[c(7,8,9,12,3)], 2, numbers)
trial$Launched <- mdy_hm(trial$Launched)
trial$Deadline <- mdy_hm(trial$Deadline)
trial$Outcome <- factor(as.character(trial$Outcome))
trial$name <- factor(as.character(trial$name))
trial$Sub_Category_name <- factor(trial$Sub_Category_name)
trial <- trial[-c(16,17)]
```

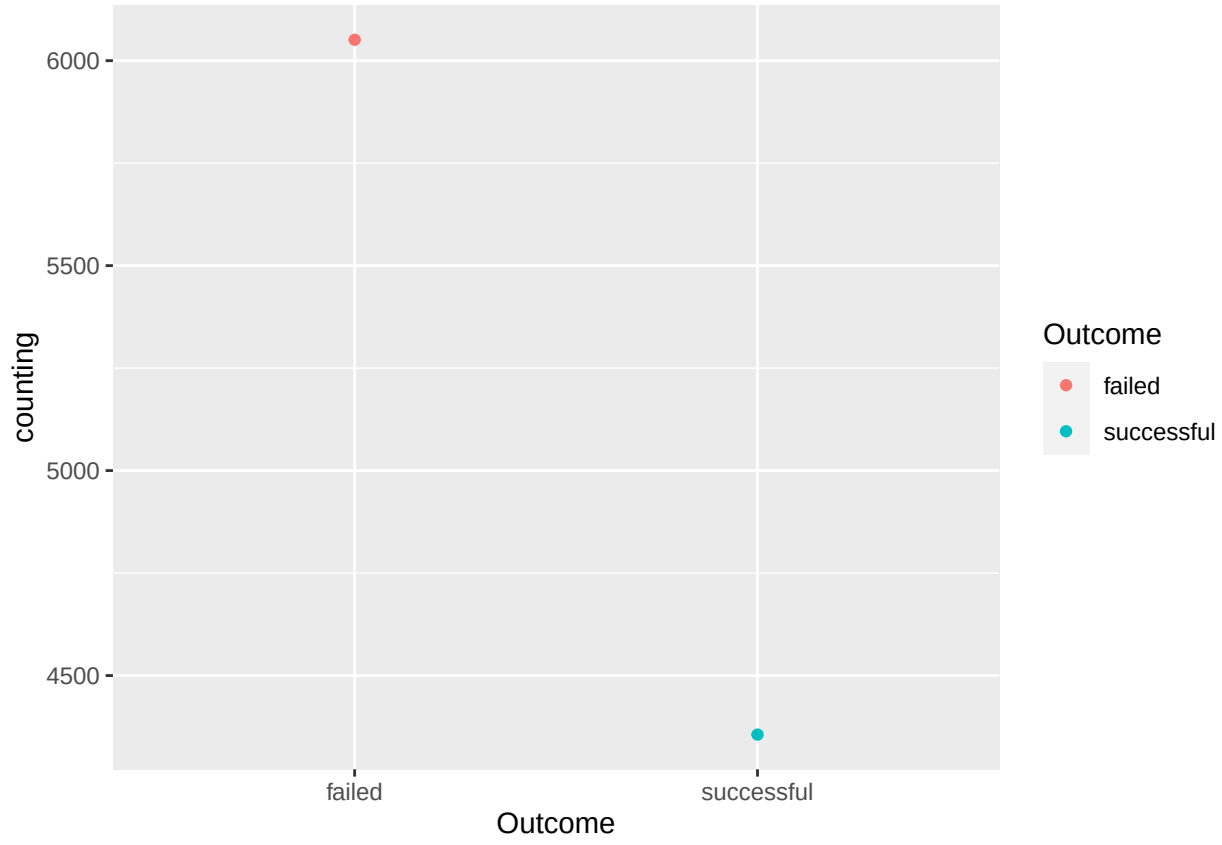
```
trial2 <- filter(trial, trial$Outcome %in% c('successful', 'failed'))
trial2 <- left_join(trial2, currency, by = c('Currency_id' = 'id'))
trial2 <- left_join(trial2, countries, by = c('Country_id' = 'id'))
colnames(trial2) <- c('Name', 'Sub_category_id', 'Country_id', 'currency_id', 'Launched', 'Deadline', '
```

```
trial2 <- trial2 %>% mutate(month_of_launch = month(Launched, label = TRUE))
trial2 <- trial2 %>% mutate(day_of_launch = wday(Launched, label = TRUE))
```

```
ggplot(trial2) + geom_bar(mapping = aes(x = Outcome, color =Outcome)) + scale_fill_brewer(palette = "Blu
```

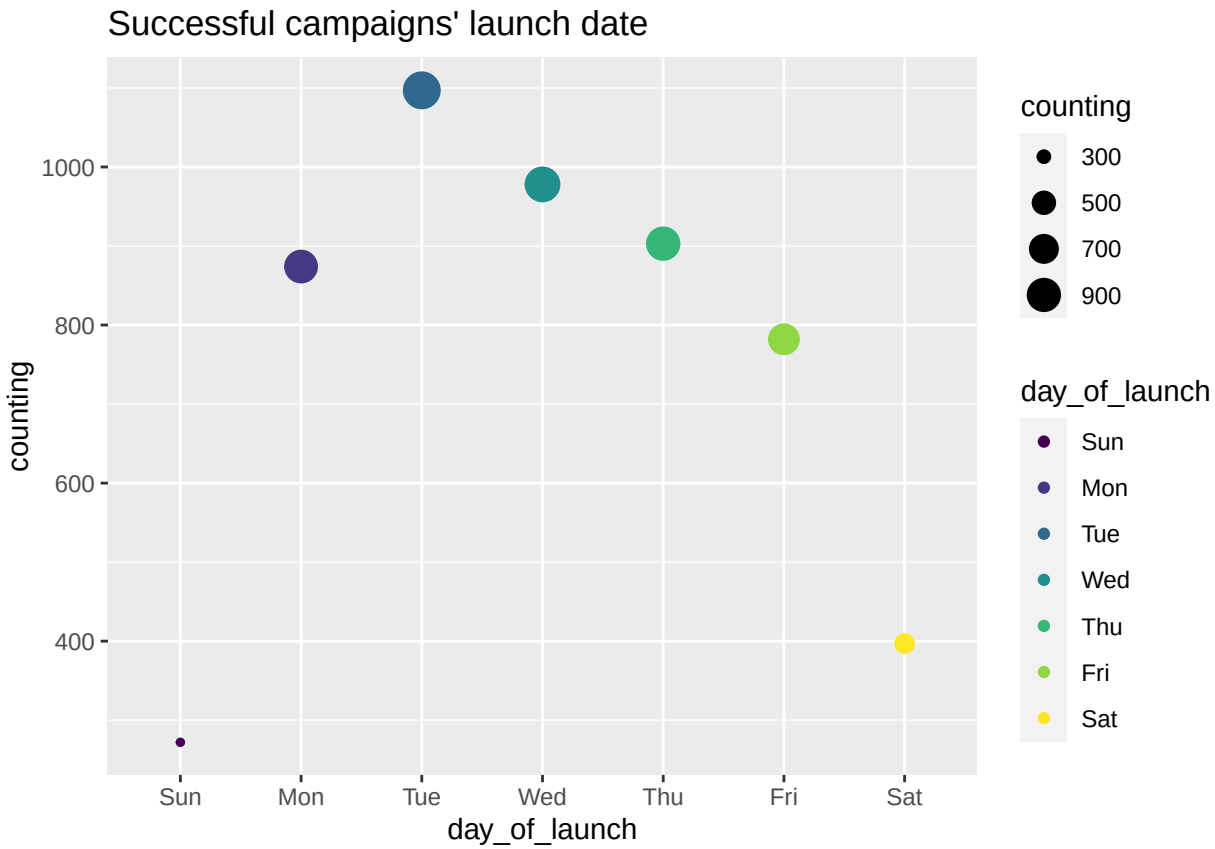


```
trial2 %>% filter(trial2$Currency == 'USD') %>%
  group_by(Outcome) %>%
  summarise(counting =  n()) %>%
ggplot() + geom_point(aes(x = Outcome, y = counting, color = Outcome))
```



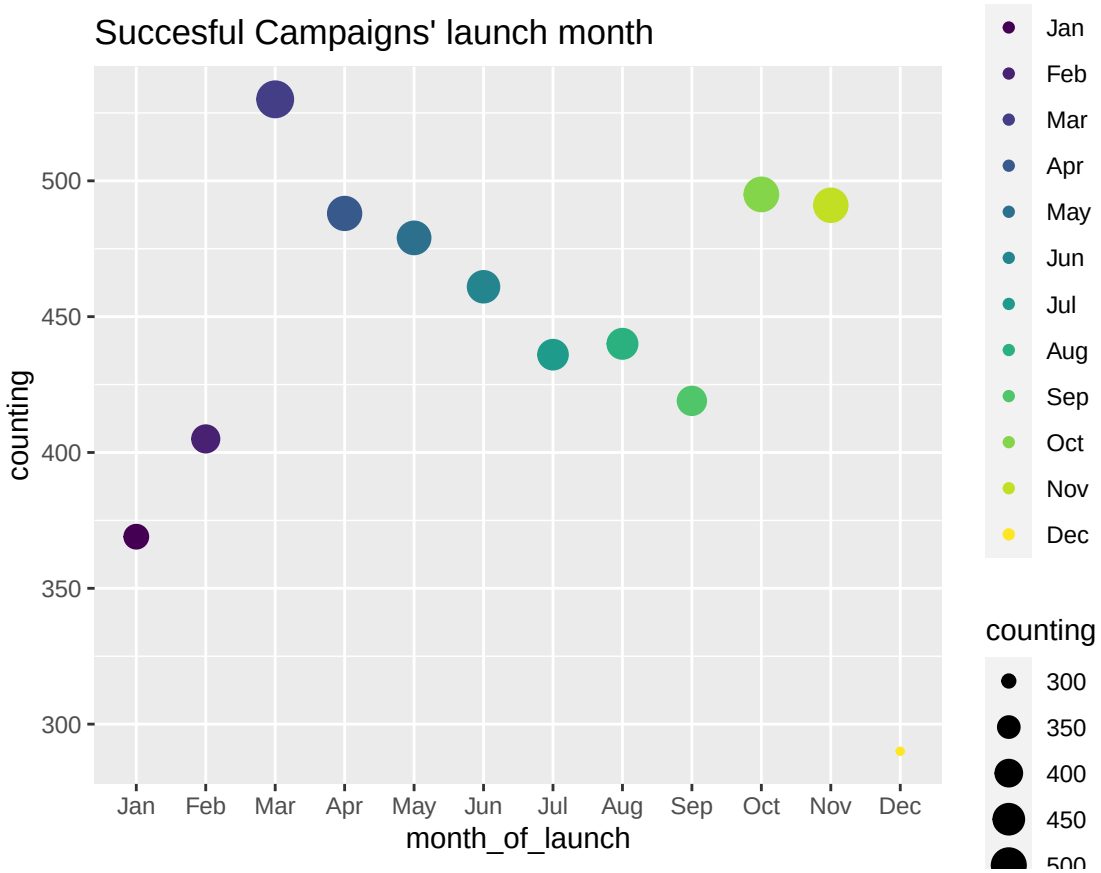

```
#do it early weekday, Tuesday being the peak

trial2 %>% filter((trial2$Outcome == 'successful')) %>%
  group_by(day_of_launch) %>%
  summarise(counting = n()) %>%
  ggplot() + geom_point(aes(x = day_of_launch, y = counting, color = day_of_launch, size = counting)) +
```



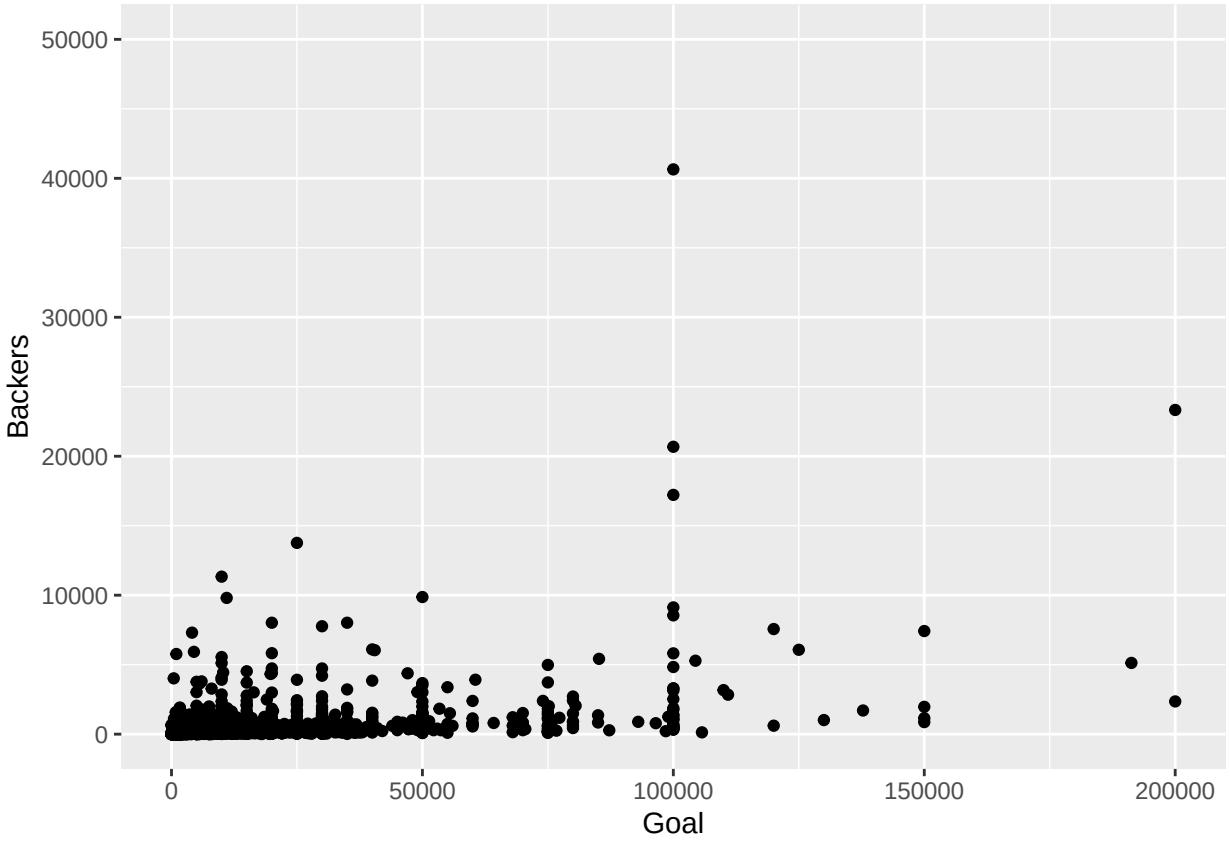

```
#month irrespective of year

trial2 %>% filter((trial2$Outcome == 'successful')) %>%
  group_by(month_of_launch) %>%
  summarise(counting = n()) %>%
  ggplot() + geom_point(aes(x = month_of_launch, y = counting, color = month_of_launch, size = counting)
```



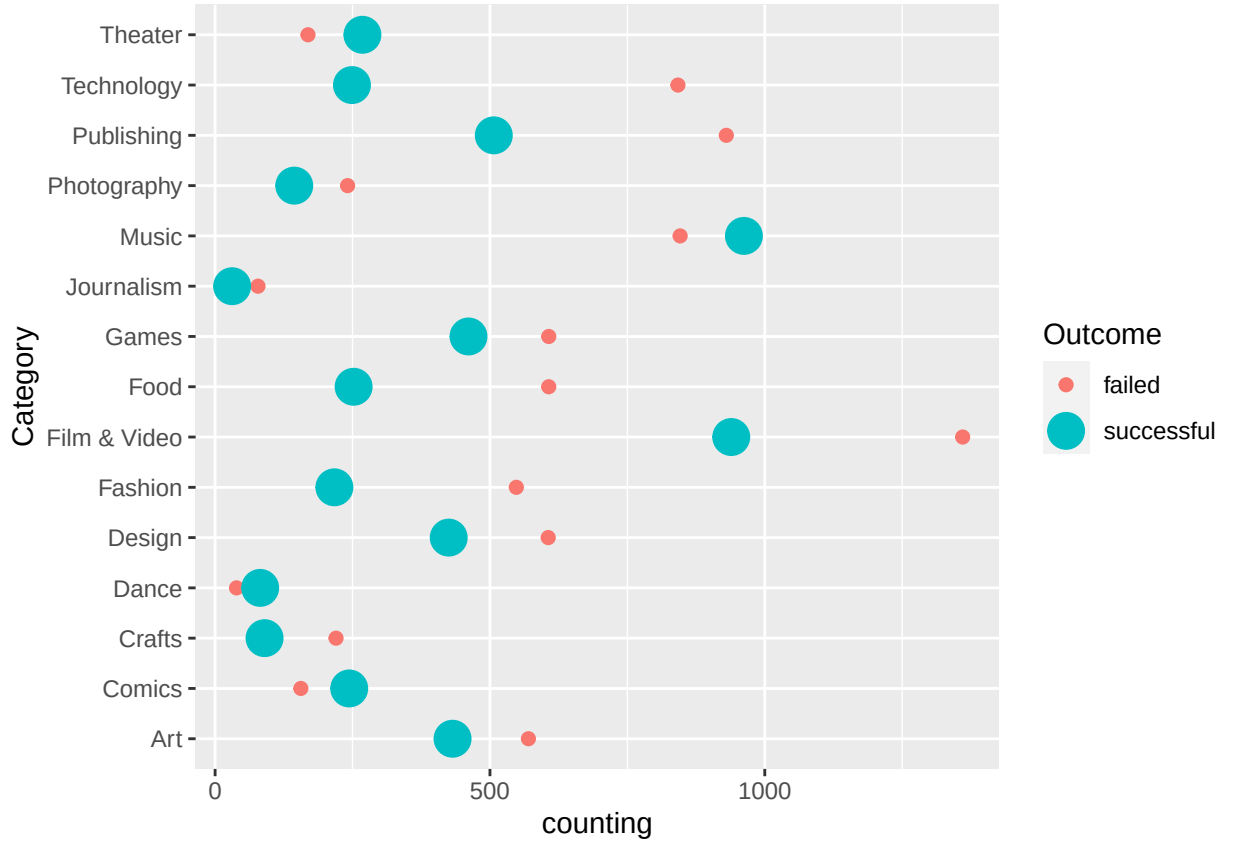

```
trial2 %>% filter(Outcome == 'successful') %>%
  ggplot() + geom_point(aes(x = Goal, y = Backers)) + xlim(0, 200000) + ylim(0, 50000)
```

```
## Warning: Removed 11 rows containing missing values (geom_point).
```




```
trial2 %>% group_by(Category, Outcome) %>%
  summarise(counting = n()) %>%
  ggplot() + geom_point(aes(x = Category, y = counting, color = Outcome, size = Outcome)) + coord_flip()
```

```
## Warning: Using size for a discrete variable is not advised.
```



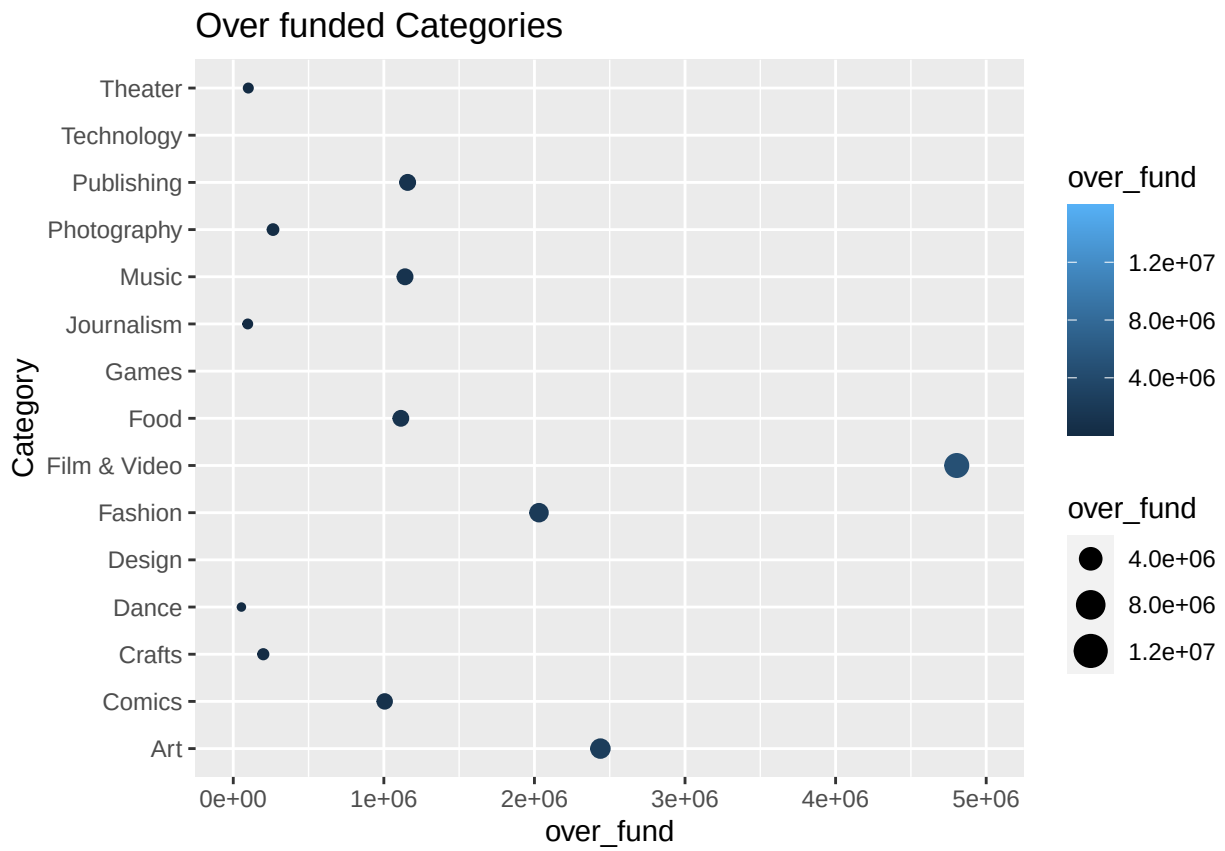

```
#over_funded projects
trial2 <- trial2 %>% mutate(money_overfunded = ifelse(Outcome == 'successful', Pledged - Goal, 0))
trial2 <- trial2 %>% mutate(overfunded = ifelse(money_overfunded > 0, 'yes', 'no'))

trial2 %>%
  filter(Currency == 'USD') %>%
  group_by(overfunded) %>%
  summarise(overfunds = n())
```

```
## # A tibble: 2 x 2
##   overfunded overfunds
##   <chr>          <int>
## 1 no              6171
## 2 yes             4236
```

```
#did a box plot, but many outliers, hence varies greatly, but with total for each, easier to see
trial2 %>% filter(overfunded == 'yes' & Currency == 'USD') %>%
  group_by(Category) %>% summarise(over_fund = sum(money_overfunded)) %>%
ggplot() + geom_point(aes(x = Category, y = over_fund, color = over_fund, size = over_fund)) + coord_fli
```

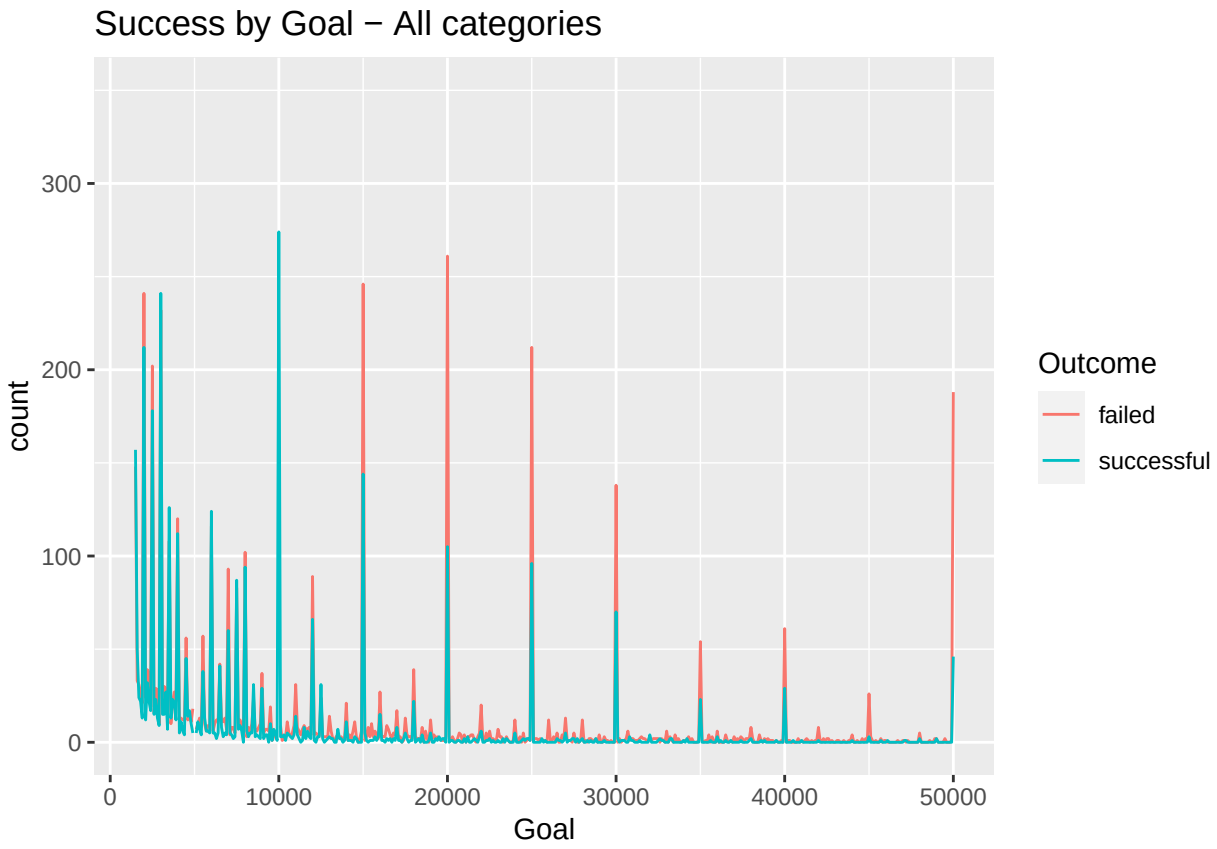
```
## Warning: Removed 3 rows containing missing values (geom_point).
```




```
#most successful ones are ner 15000 goal and less
trial2 %>%
  ggplot() + geom_freqpoly(aes(x = Goal, color = Outcome), binwidth = 100) + xlim(1500,50000) + ylim(0,3
```

```
## Warning: Removed 3398 rows containing non-finite values (stat_bin).
```

```
## Warning: Removed 4 row(s) containing missing values (geom_path).
```



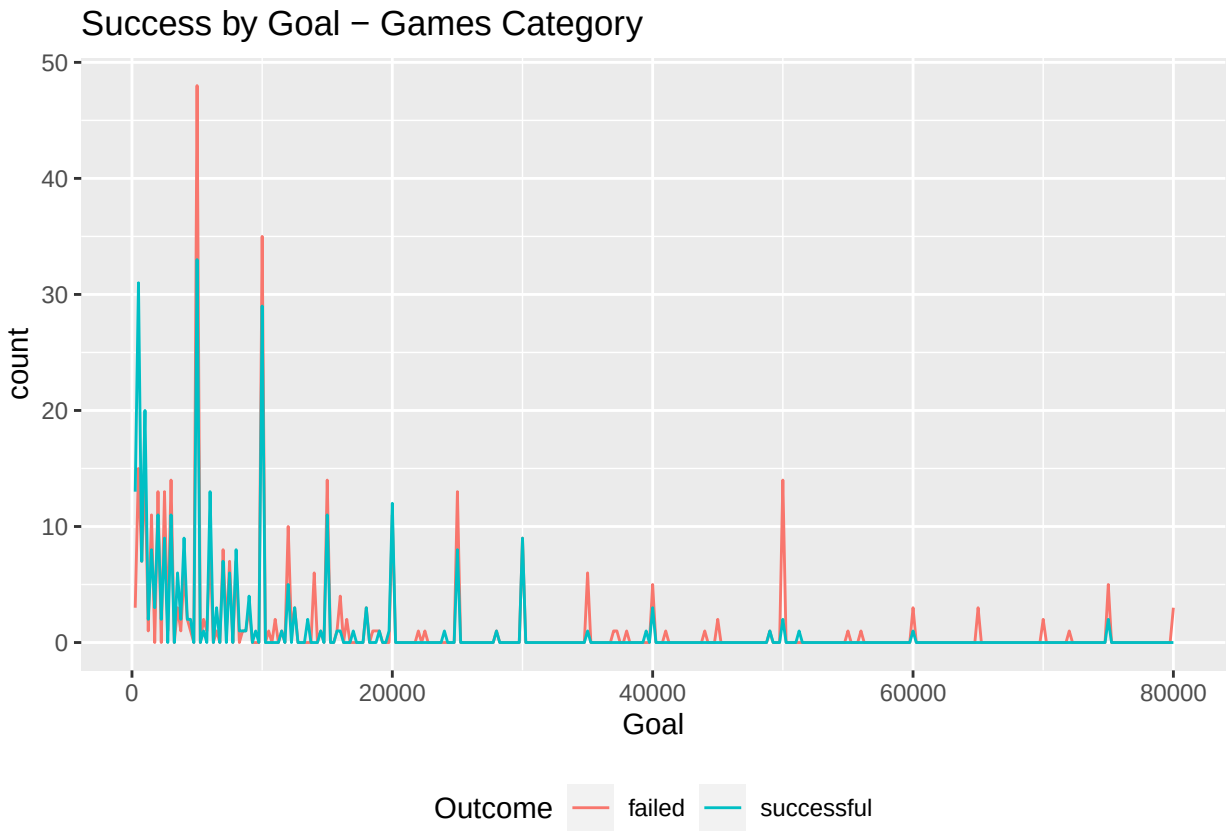

```
#Games



games <- trial2 %>% filter(Sub_category_id %in% c(13,14,44,66,70,113,122,134))
games %>% filter(Currency == 'USD') %>%
  ggplot() + geom_freqpoly(aes(x = Goal, color = Outcome), binwidth = 250) + xlim(100, 80000) + ggtitle
```

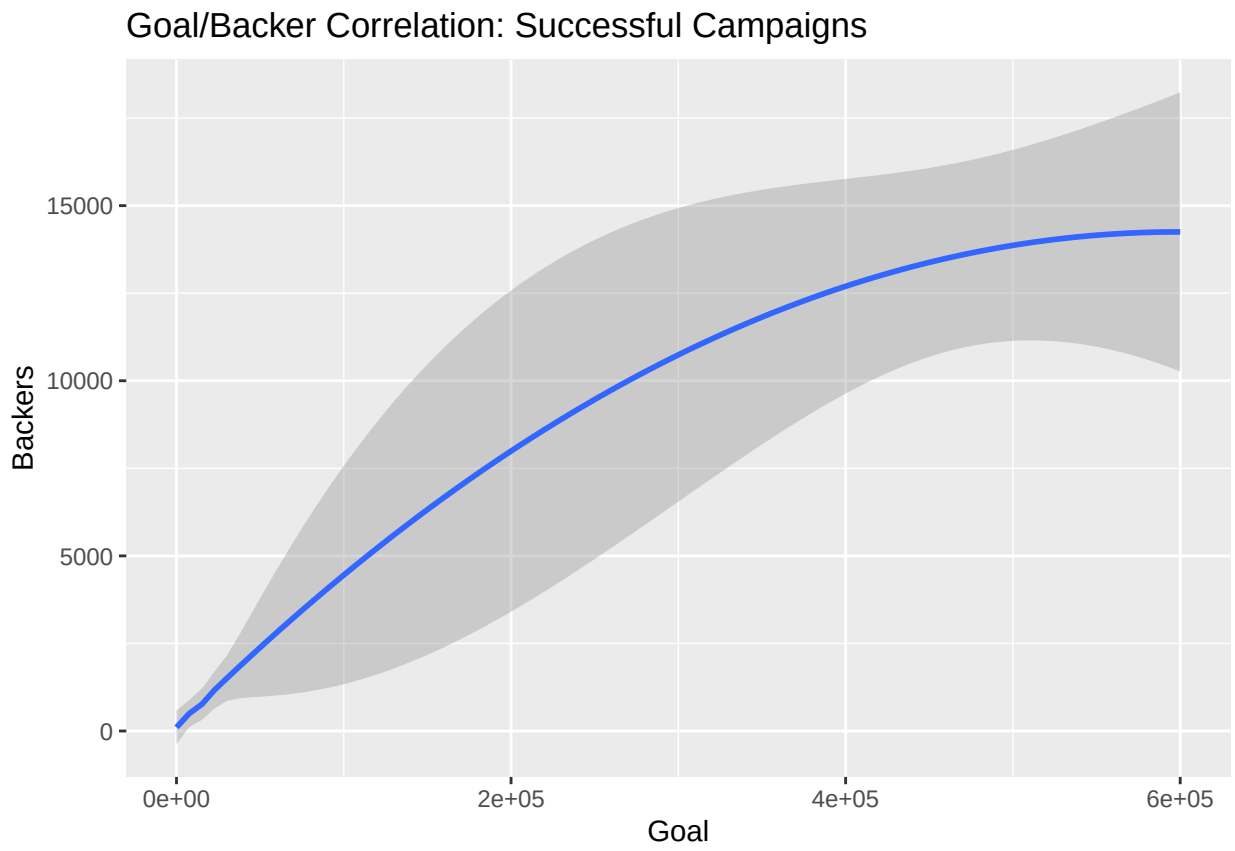
```
## Warning: Removed 53 rows containing non-finite values (stat_bin).
```

```
## Warning: Removed 6 row(s) containing missing values (geom_path).
```




```
games %>% filter(Outcome == 'successful') %>% ggplot() + geom_smooth(aes(x = Goal, y = Backers)) + ggti
```

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




```
data = read.csv('r_doc.csv')
data = data[-c(1)]
```

```
library(caTools)
set.seed(123)
split = sample.split(data$Outcome, SplitRatio = 0.8)
training_set = subset(data, split == TRUE)
test_set = subset(data, split == FALSE)

# Feature Scaling
training_set[c(1,2,3,5)] = scale(training_set[c(1,2,3,5)])
test_set[c(1,2,3,5)] = scale(test_set[c(1,2,3,5)])


library(randomForest)
```

```
## randomForest 4.6-14

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:dplyr':
##
##     combine
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
set.seed(123)
classifier = randomForest(x = training_set[-9],
                          y = training_set$Outcome,
                          ntree = 500,
                          )

# Predicting the Test set results
y_pred = predict(classifier, newdata = test_set[-9])

cm = table(test_set[, 9], y_pred)
importance(classifier)
```

```
##                 MeanDecreaseGini
## Duration               211.37052
## Goal                   741.83830
## Backers               3069.06235
## special_name            41.73268
## length_of_name         171.80487
## month_of_launch        264.53991
## day_of_launch          173.56886
## Category               328.95411
```

```
our_pred = c(30,400000, 1296, factor('no'), 10, factor('Sep'), factor('Wed'), factor('Games')  )
predict(classifier, newdata = our_pred)
```

```
##          1
## successful
## Levels: failed successful
```