Analyzing Github Repositories through Public API Events

Overview

The purpose of this study is to explore possible correlations between repository and event parameters in order to determine a sampling methodology for Github repositories based on public Github event activity.

Github is huge. With over 10 million repositories, the Github repository population is highly variable, thus making analysis incredibly challenging.

The solution explored in this study uses activity data collected by the Github Archive project. The Github Archive Project keeps a regularly up-to-date archive of the public event stream available through the Github API. The data are available as a public data set on Google BigQuery.

Events data allow for time-based stratification of Github repositories. For purposes of this research, only recently active repositories are of interest, however even research exploring the lifespan of repositories could benefit from this stratification.

This study is part of a larger research project to answer the following questions:

- What are the measurable impacts of Continuous Integration?
- What software projects would most likely benefit from Continuous Integration?

Hypotheses

There is a relationship between the number of actors interacting with a repository and the types of events present in a given repository.

Repositories with an overall higher number of total unique actors will have a higher frequency of certain event types, whereas less active repositories will show little to no frequency of that same event type.

Repositories with a smaller number of total actors will show a higher frequency of event types that are less frequent in repositories with a higher number of total unique actors.

Null Hypothesis

There is no clear correlation between event types and the total number of unique actors for a repository. The event types do not provide information about the total unique actors.

Methodology

This research analyzed 6 months of public GitHub activity published in the GitHub Archive, from June 2016 to December 2016. For data set creation, see the accompanying file, “events_analysis_data.Rmd”.

```
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(ggplot2)
library(reshape2)
```

Event Frequency

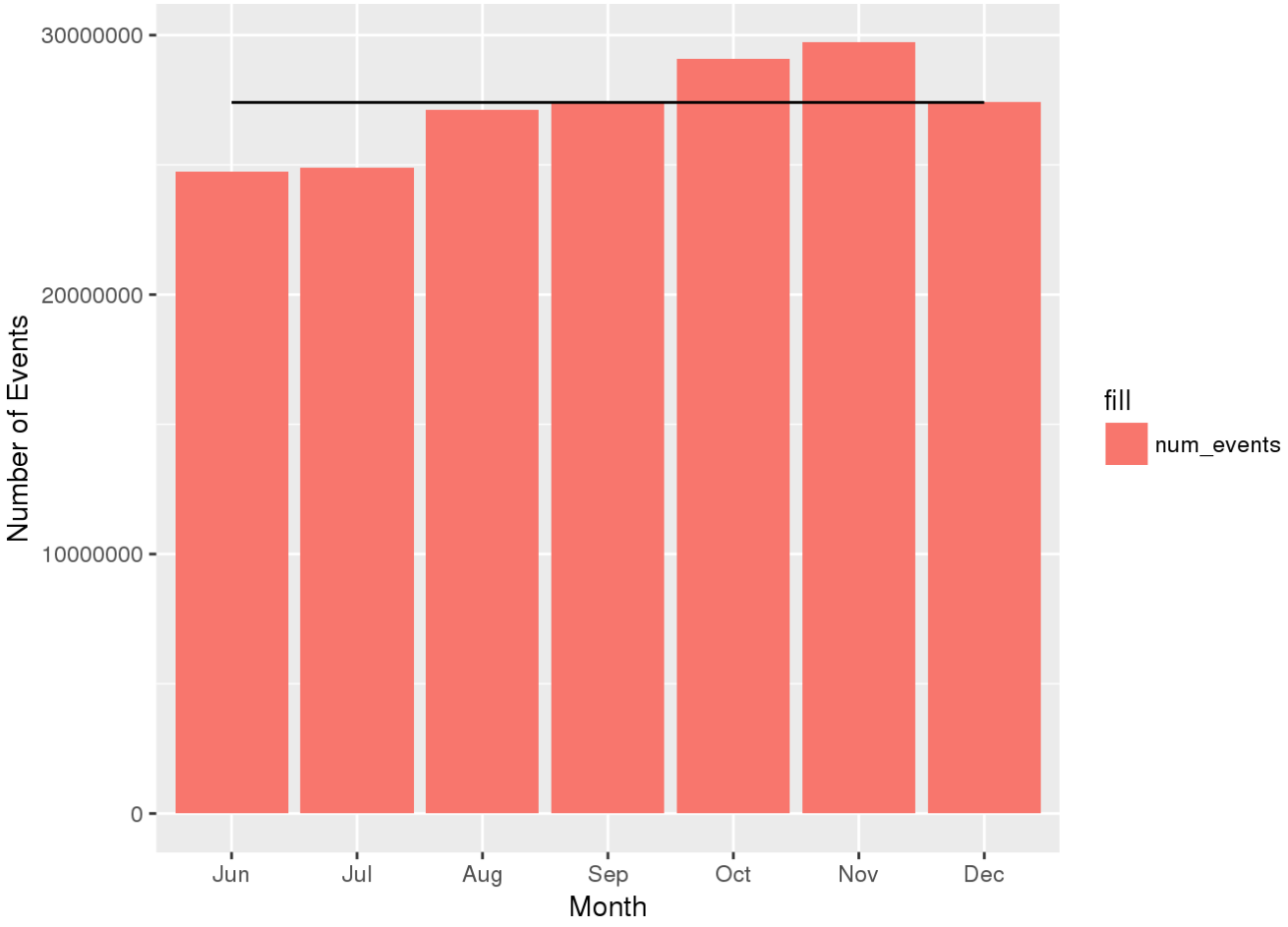
How variable is public GitHub event activity from month to month?

When we examine the total number of events, unique actors, and unique repositories over the 6 month period, there appears to be a lot of variation. However we see less variation when we look at the change from month to month and the deviation from the median. The month-to-month change and the deviation from median have been adjusted to relative proportions to allow for a comparison between the different totals, and to show some sense of the overall scale.

Further analysis needs to be done over a broader span of time for any solid conclusions about overall variability. However, for the purposes of this study, the variability indicated below does not appear to be significant enough to skew further analysis results.

Events per Month

```
event_totals <- readRDS("event_totals.rds")

event_totals$month <- factor(event_totals$month, levels = month.abb)

event_totals_median <- median(event_totals$num_events)

ggplot(data = event_totals, aes(x = month, y = num_events, fill="num_events")) +
  geom_bar(stat="identity", position="dodge") +
  geom_line(stat="identity", group=1, aes(y = event_totals_median)) +
  ylab("Number of Events") +
  xlab("Month") +
  scale_y_continuous(labels=function(n){format(n, scientific = FALSE)})
```

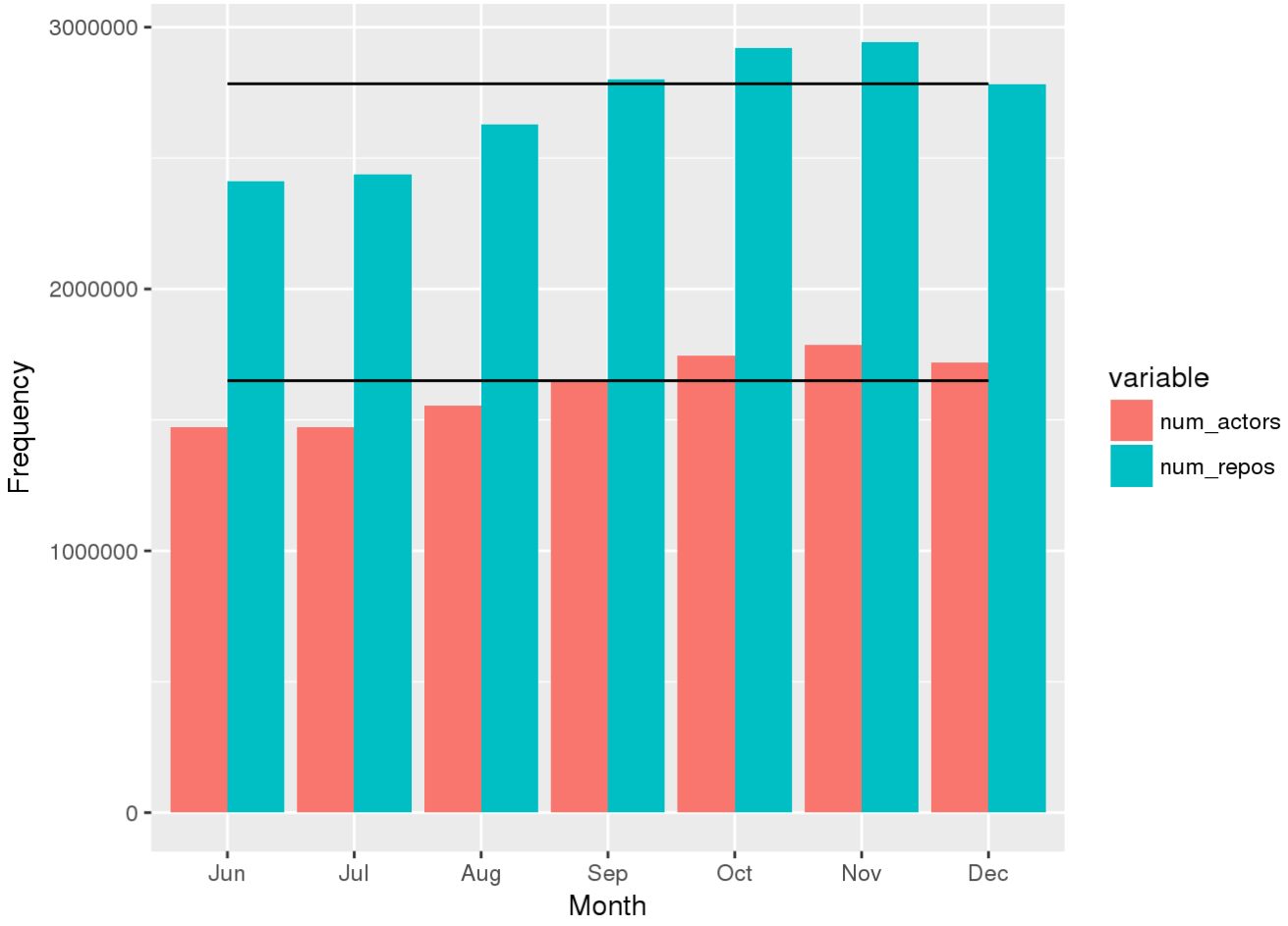


Unique Actors and Repositories per Month

```
event_totals_long <- melt(event_totals) %>% filter(variable == "num_actors" | var
iable == "num_repos")
```

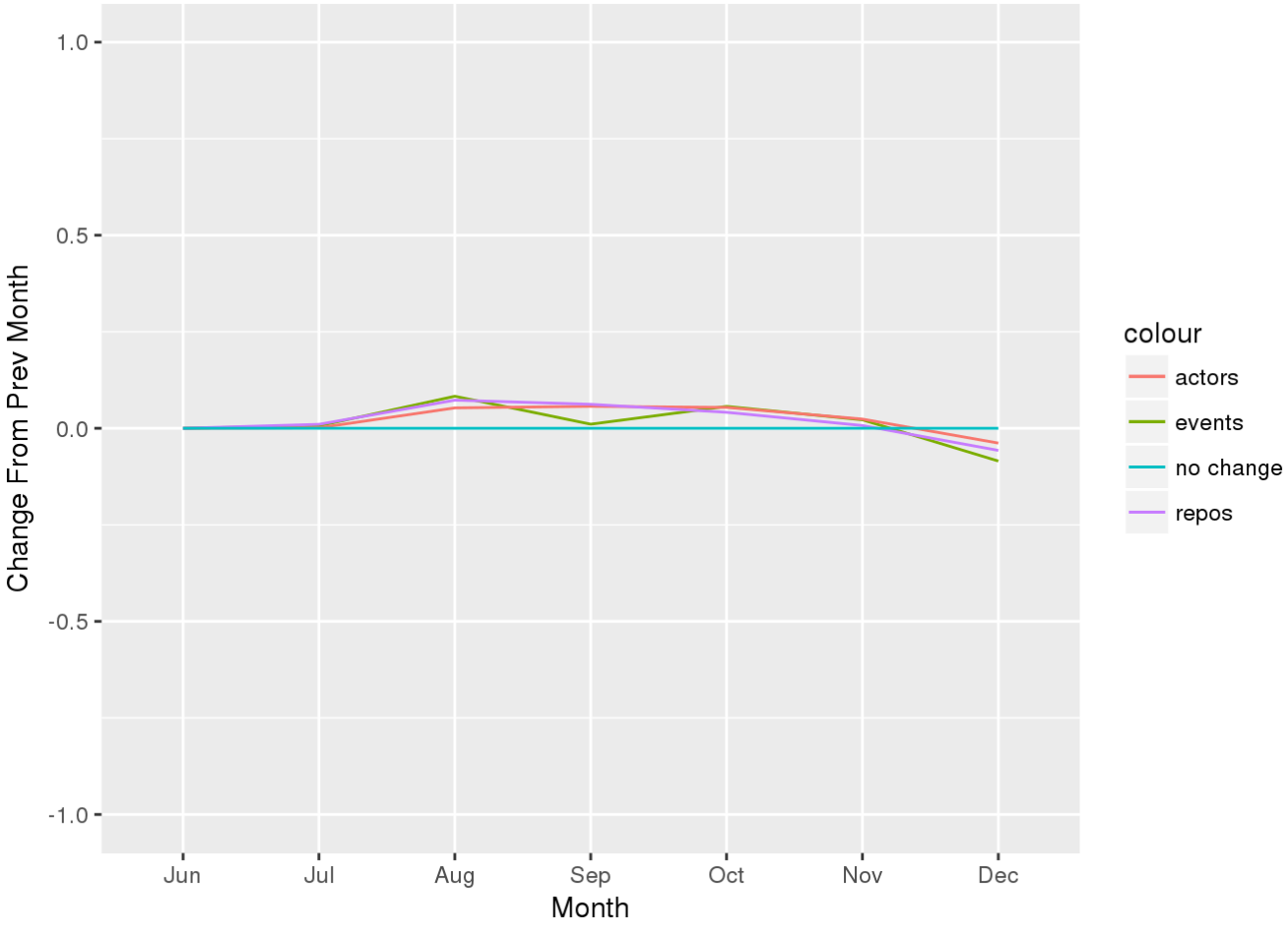
```
## Using month as id variables
```

```
actor_totals_median <- median(event_totals$num_actors)
repo_totals_median <- median(event_totals$num_repos)

ggplot(data = event_totals_long, aes(x = month, y = value, fill = variable)) +
  geom_bar(stat="identity", position="dodge") +
  geom_line(stat="identity", group=1, aes(y = actor_totals_median)) +
  geom_line(stat="identity", group=1, aes(y = repo_totals_median)) +
  ylab("Frequency") +
  xlab("Month") +
  scale_y_continuous(labels=function(n){format(n, scientific = FALSE)})
```



Change From Previous Month

```
event_totals_median <- median(event_totals$num_events)
actor_totals_median <- median(event_totals$num_actors)
repo_totals_median <- median(event_totals$num_repos)

event_totals <- event_totals %>%
  arrange(month) %>%
  mutate(
    events_change = c(0, diff(num_events)),
    events_deviation = num_events - event_totals_median,
    actors_change= c(0, diff(num_actors)),
    actors_deviation = num_actors - actor_totals_median,
    repos_change = c(0,diff(num_repos)),
    repos_deviation = num_repos - repo_totals_median)

ggplot(data = event_totals, aes(x = month)) +
  geom_line(stat="identity", group=1, aes(y = events_change/num_events, color =
"events")) +
  geom_line(stat="identity", group=1, aes(y = actors_change/num_actors, color =
"actors")) +
  geom_line(stat="identity", group=1, aes(y = repos_change/num_repos, color = "re
pos")) +
  geom_line(stat="identity", group=1, aes(y = 0, color = "no change")) +
  ylab("Change From Prev Month") +
  xlab("Month") +
  ylim(-1,1)
```

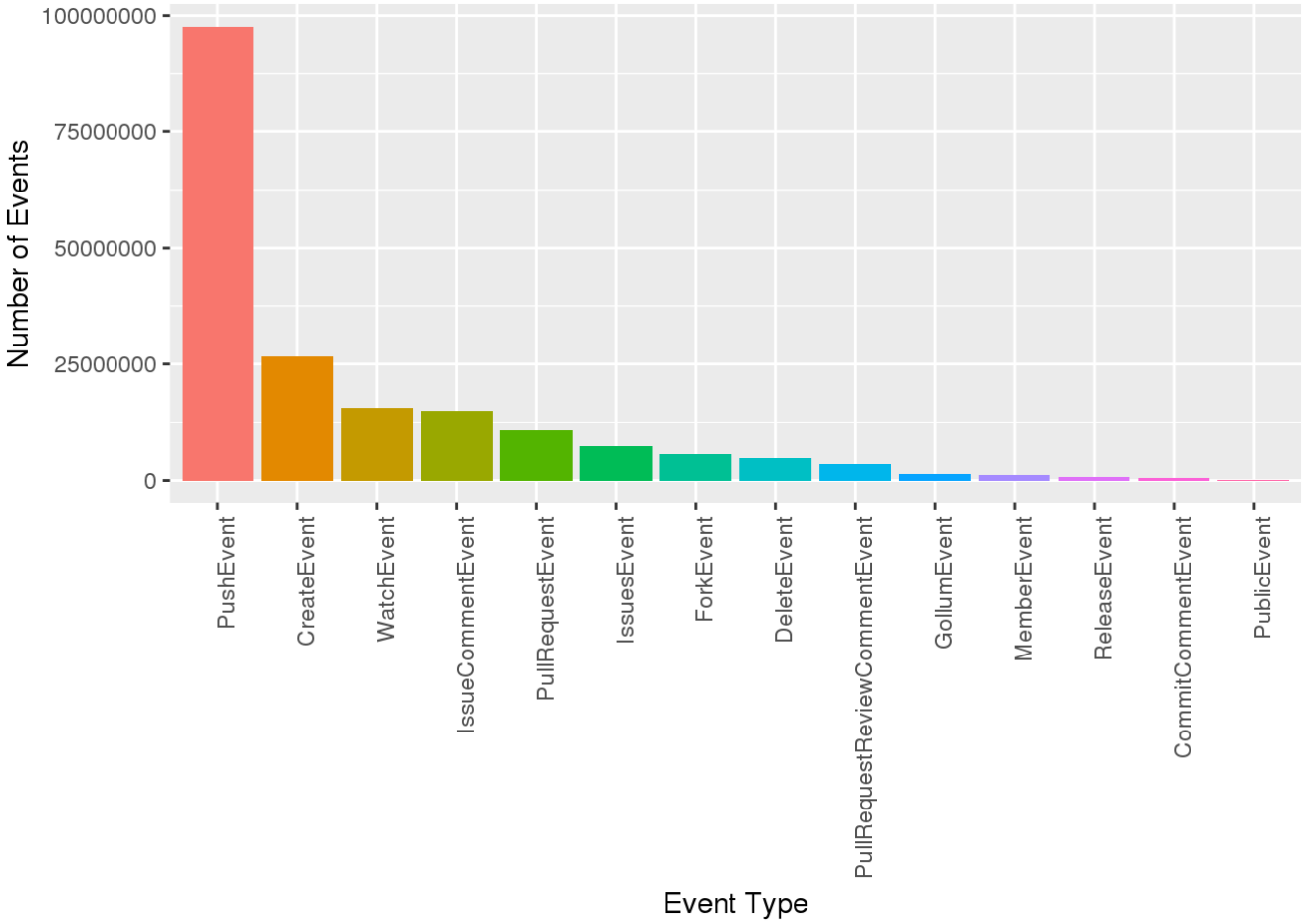


What proportion of the events are represented by each type?

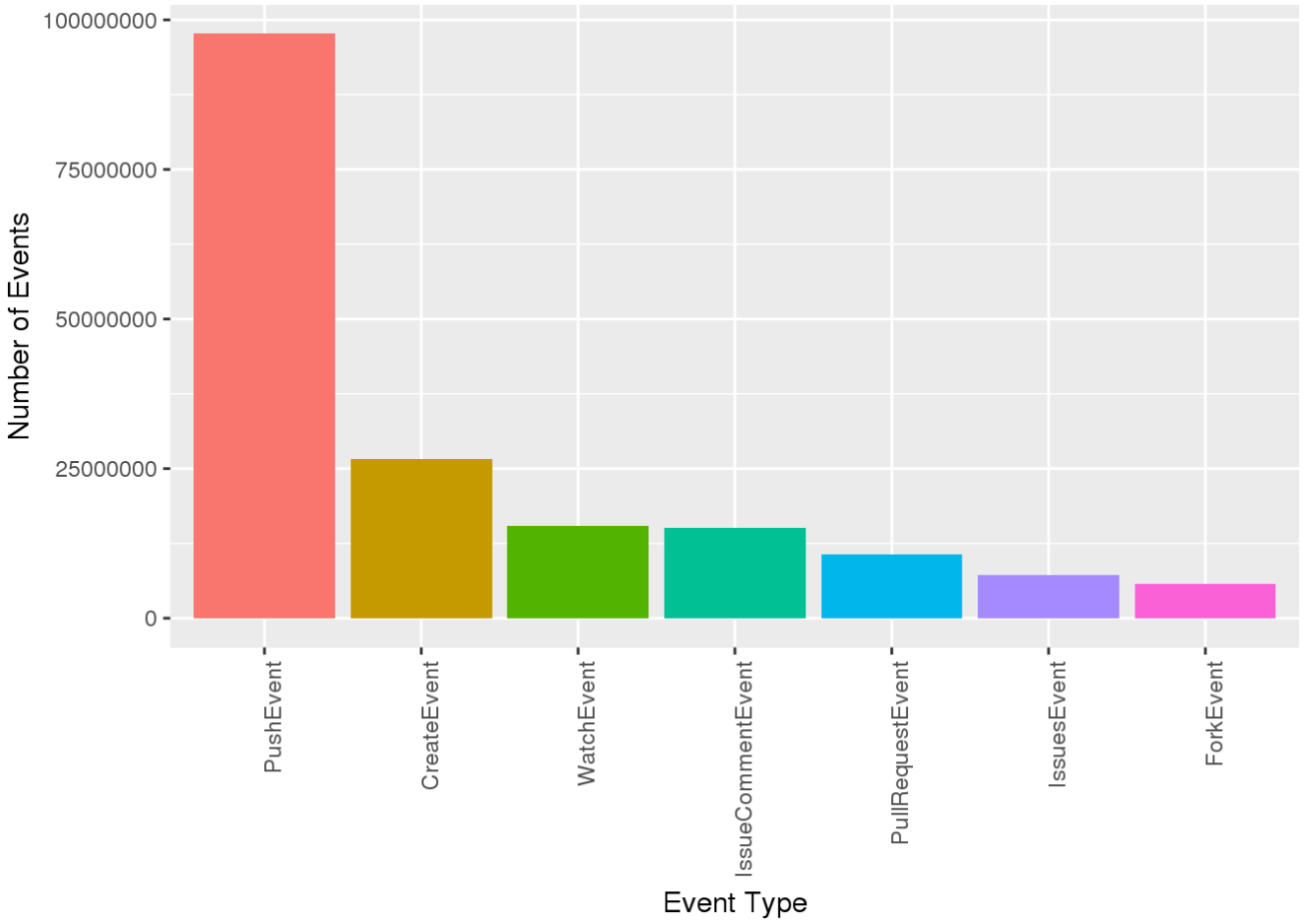
Event types represent different types of activity that occur in each project. Event types that occur more frequently might be distributed across more repositories and therefore may show less correlation to special repository parameters. Samples drawn from more frequent event types might show more variability between repository types than samples drawn from less frequent types. Further analysis needs to be done on each type to see if it is correlated with repository parameters easily available in the events data.

Event Type Frequency

```
event_type_totals_all <- readRDS("event_type_totals.rds")

event_type_totals_all$type <- factor(event_type_totals_all$type,
  levels = unique(event_type_totals_all$type[order(event_type_totals_all$num_even
ts, decreasing=TRUE)]))

ggplot(data = event_type_totals_all, aes(x=type, y=num_events, fill=type)) +
  geom_bar(stat="identity") +
  theme(legend.position="none") +
  ylab("Number of Events") +
  xlab("Event Type") +
  scale_y_continuous(labels=function(n){format(n, scientific = FALSE)}) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```



```
event_type_totals <- event_type_totals_all %>%
  filter(type == "PushEvent" |
           type == "CreateEvent" |
           type == "IssueCommentEvent" |
           type == "WatchEvent" |
           type == "PullRequestEvent" |
           type == "IssuesEvent" |
           type == "ForkEvent")

event_type_totals$type <- factor(event_type_totals$type,
  levels = unique(event_type_totals$type[order(event_type_totals$num_events, decr
easing=TRUE)]))

ggplot(data = event_type_totals, aes(x=type, y=num_events, fill=type)) +
  geom_bar(stat="identity") +
  theme(legend.position="none") +
  ylab("Number of Events") +
  xlab("Event Type") +
  scale_y_continuous(labels=function(n){format(n, scientific = FALSE)}) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```

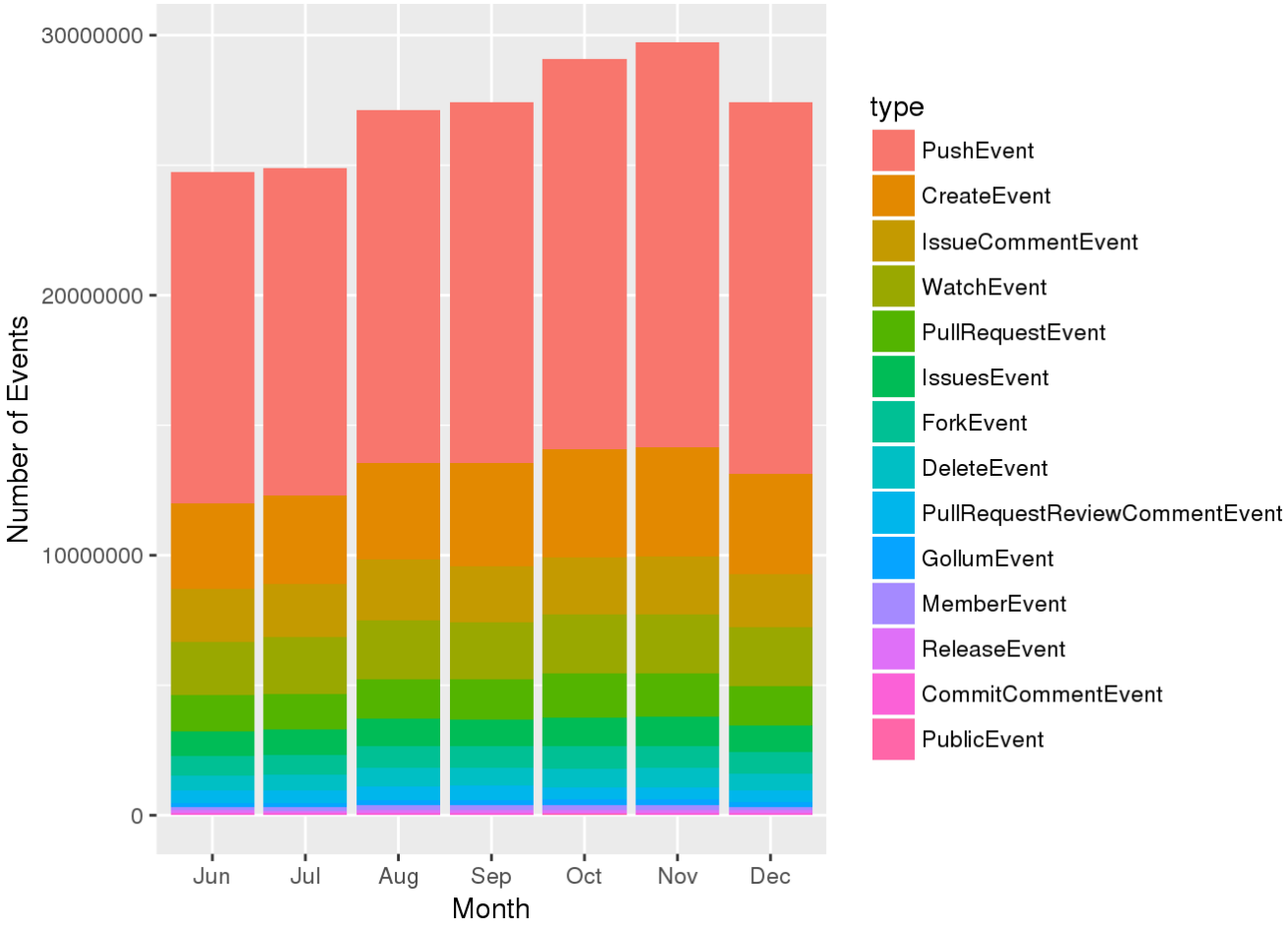


How much does event type frequency vary from month to month?

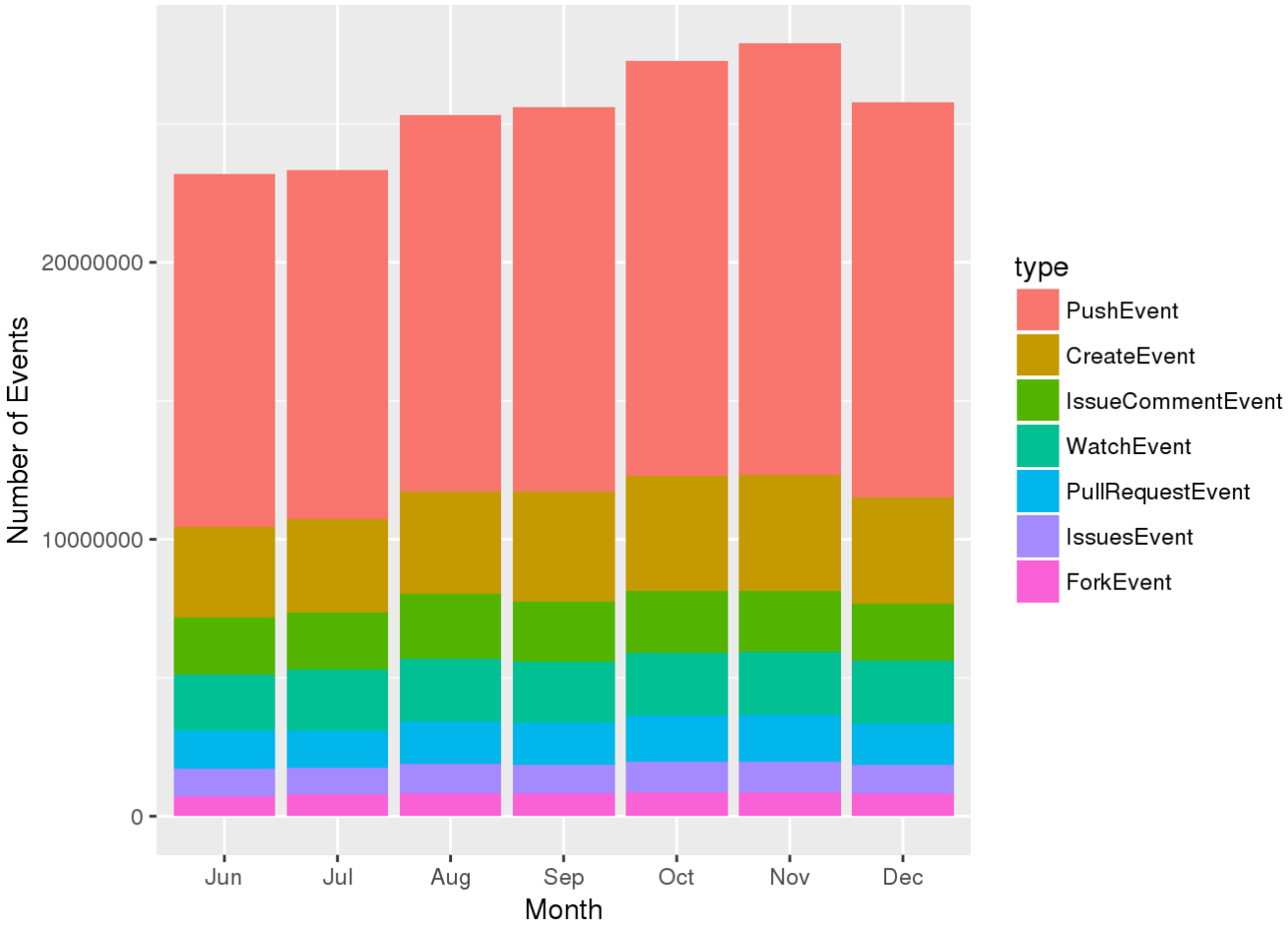
The proportions of events over the 6 month period did not change in terms of rank, except for Issue Comment and Watch Events. For the most part, the events that occurred most frequently in one month occurred most frequently in the subsequent months. The overall change from month to mont does vary. Some events show a lot of change from month to month while others show less. The most frequent events (Push, Create, Issue Comment, Watch, and Issue) appear to show less variability than the least frequent events. The overall shape of the percent change was constant for the different variables explored.

Event Type Frequency Per Month

```
month_event_type_totals_all <- readRDS("month_event_type_totals.rds")

month_event_type_totals_all$type <- factor(month_event_type_totals_all$type,
   levels = unique(month_event_type_totals_all$type[order(month_event_type_totals
_all$num_events, decreasing=TRUE)]))

month_event_type_totals_all$month <- factor(month_event_type_totals_all$month, le
vels = month.abb)

ggplot(data = month_event_type_totals_all,
       aes(x = month,
           y = num_events,
           fill=type)) +
  geom_bar(stat="identity", position="stack") +
  ylab("Number of Events") +
  xlab("Month") +
  scale_y_continuous(labels=function(n){format(n, scientific = FALSE)})
```

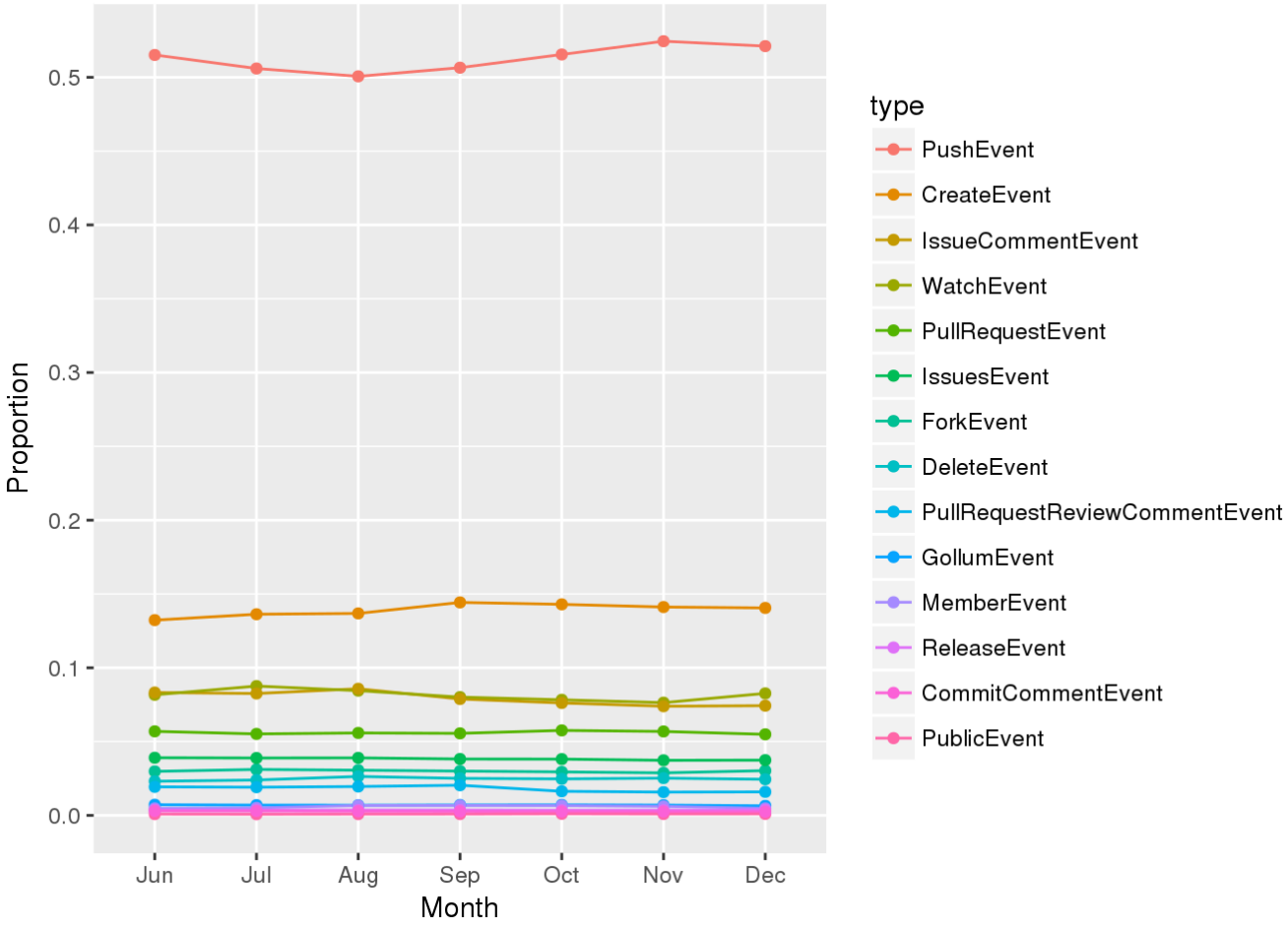


```
month_event_type_totals <- month_event_type_totals_all %>%
  filter(type == "PushEvent" |
           type == "CreateEvent" |
           type == "IssueCommentEvent" |
           type == "WatchEvent" |
           type == "PullRequestEvent" |
           type == "IssuesEvent" |
           type == "ForkEvent")

month_event_type_totals$type <- factor(month_event_type_totals$type,
   levels = unique(month_event_type_totals$type[order(month_event_type_totals$num
_events, decreasing=TRUE)]))

ggplot(data = month_event_type_totals,
       aes(x = month,
           y = num_events,
           fill=type)) +
  geom_bar(stat="identity", position="stack") +
  ylab("Number of Events") +
  xlab("Month") +
  scale_y_continuous(labels=function(n){format(n, scientific = FALSE)})
```



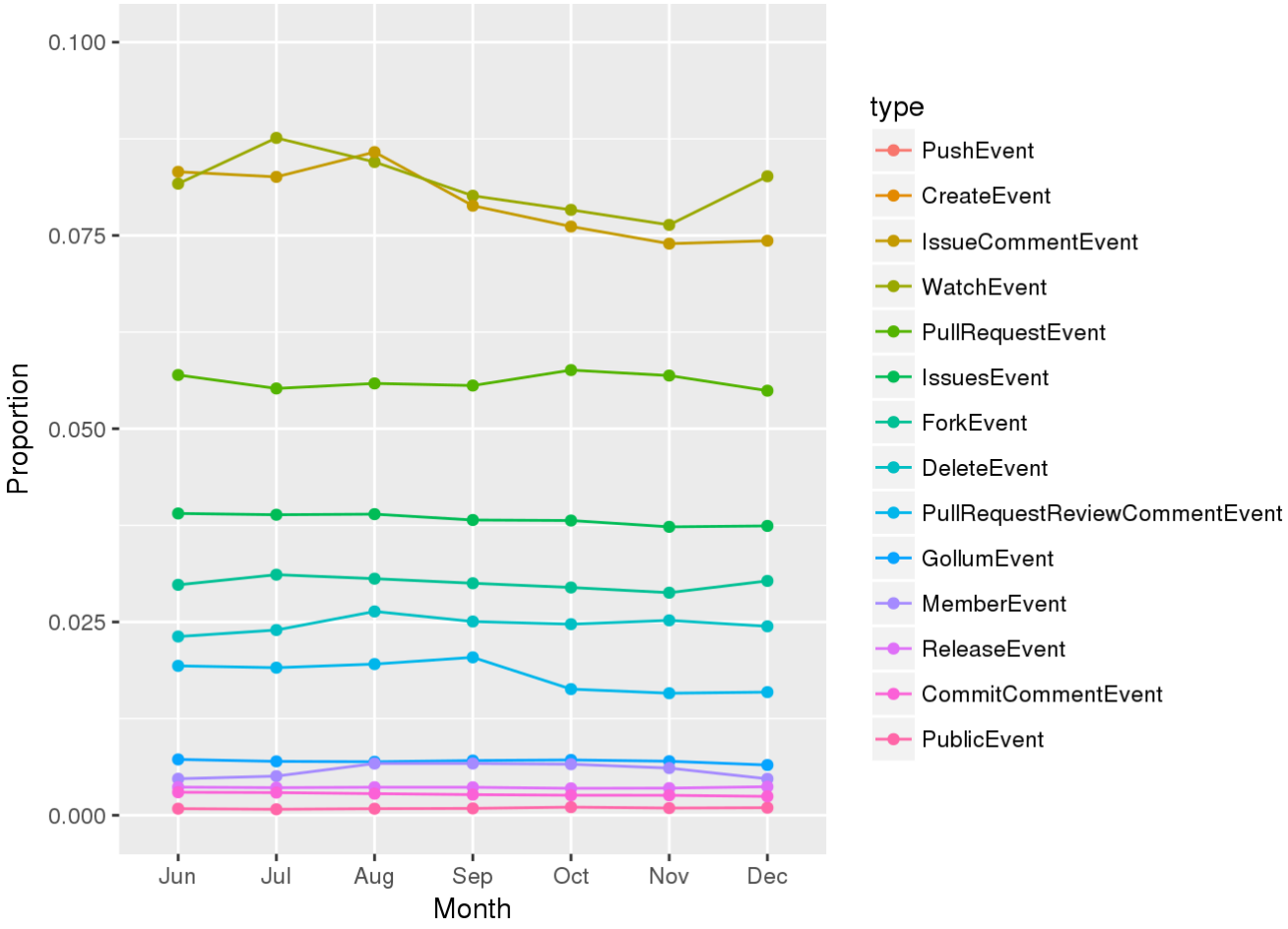
Proportion of Event Types Per Month

```
month_event_type_totals_all$total_events <-
  event_totals$num_events[match(month_event_type_totals_all$month, event_totals$m
onth)]

month_event_type_totals_all <- month_event_type_totals_all %>%
  mutate(events_prop = num_events/total_events)

ggplot(data = month_event_type_totals_all,
       aes(x = month, fill=type, colour=type, group=type)) +
  geom_line(stat="identity", aes(y = events_prop)) +
  geom_point(stat="identity", aes(y = events_prop)) +
  ylab("Proportion") +
  xlab("Month")
```



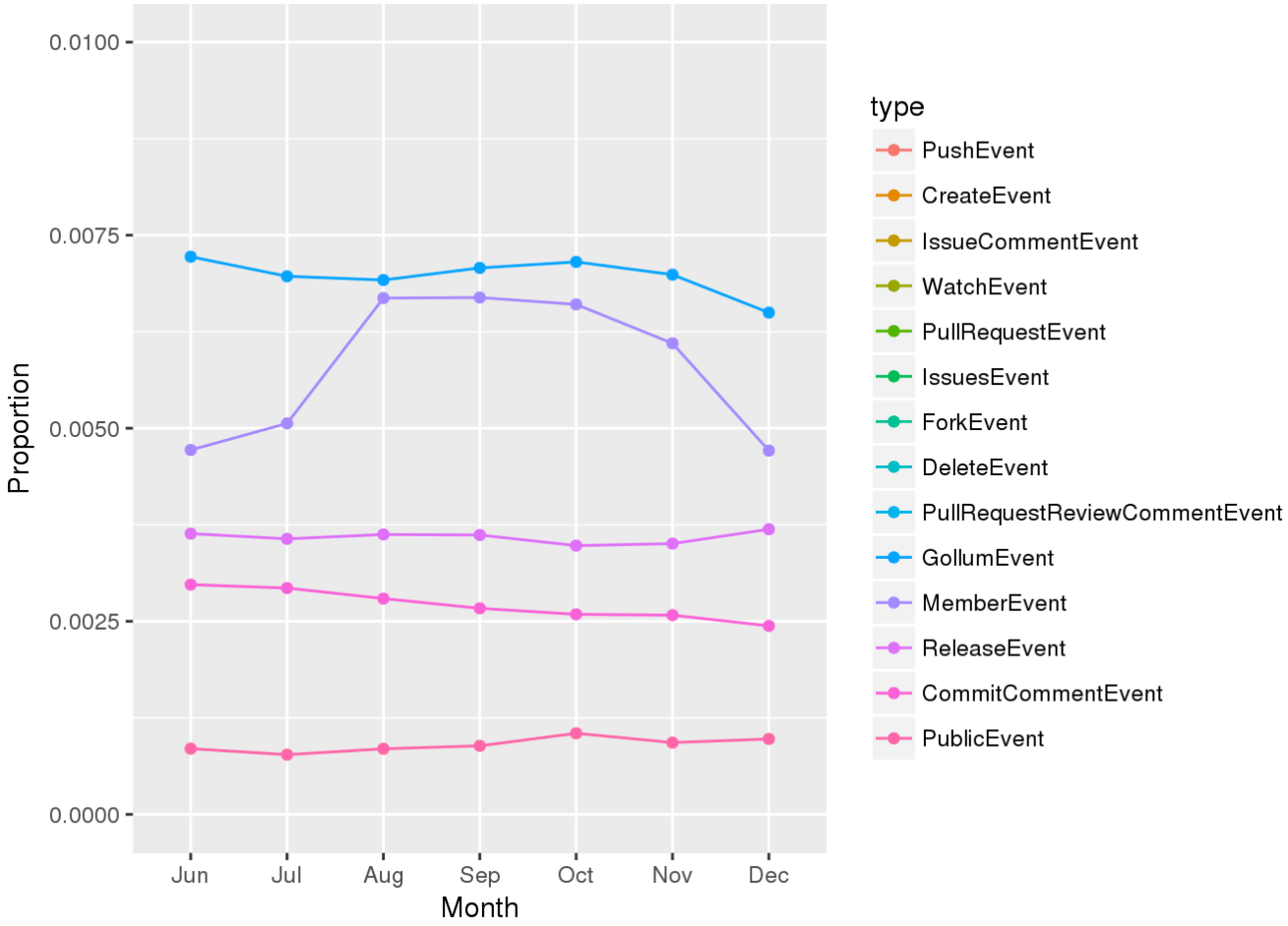

```
ggplot(data = month_event_type_totals_all,
       aes(x = month, fill=type, colour=type, group=type)) +
  geom_line(stat="identity", aes(y = events_prop)) +
  geom_point(stat="identity", aes(y = events_prop)) +
  ylab("Proportion") +
  xlab("Month") +
  ylim(0,.1)
```

```
## Warning: Removed 14 rows containing missing values (geom_point).
```




```
ggplot(data = month_event_type_totals_all,
       aes(x = month, fill=type, colour=type, group=type)) +
  geom_line(stat="identity", aes(y = events_prop)) +
  geom_point(stat="identity", aes(y = events_prop)) +
  ylab("Proportion") +
  xlab("Month") +
  ylim(0,.01)
```

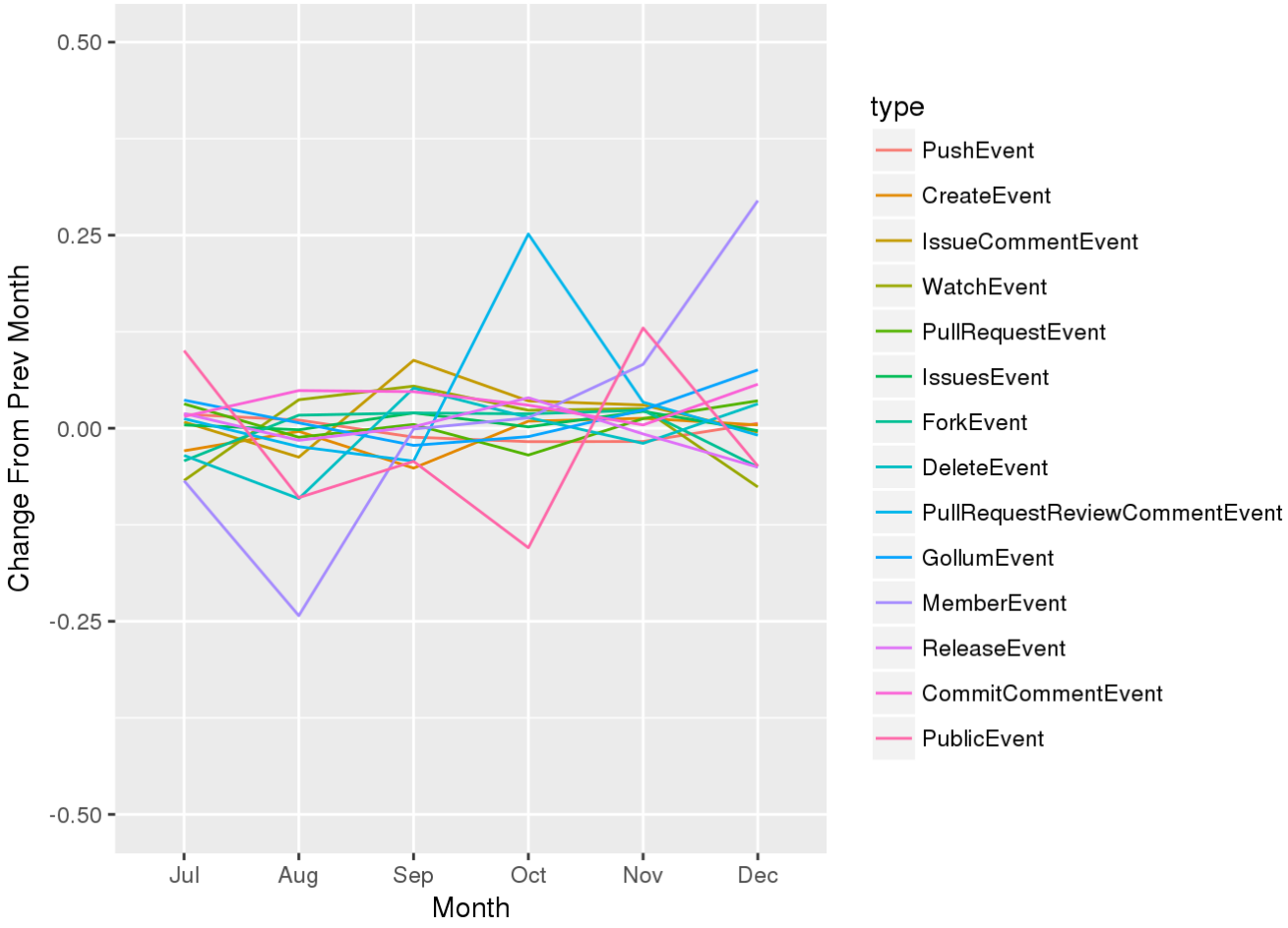
```
## Warning: Removed 63 rows containing missing values (geom_point).
```



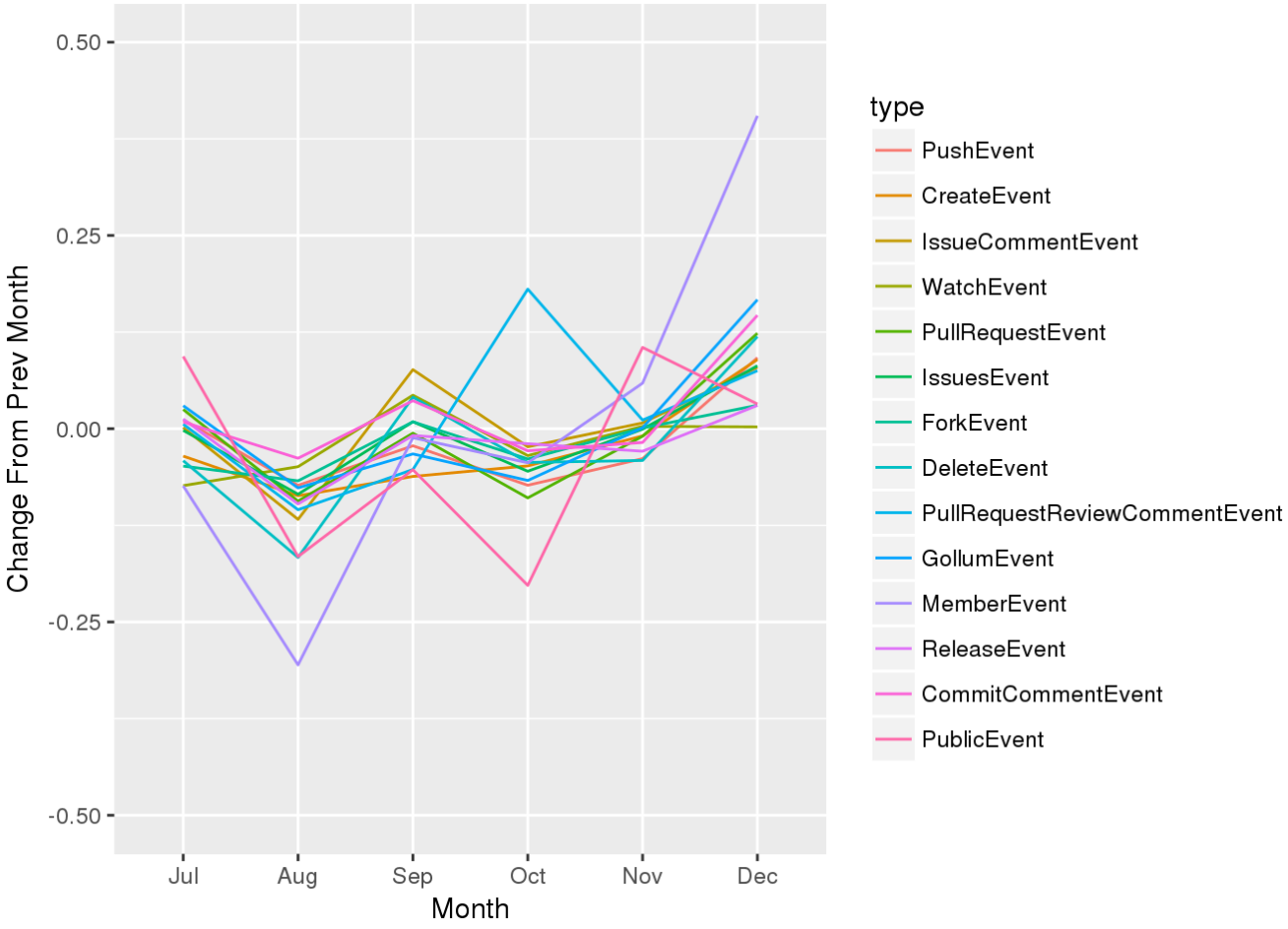
Change From Previous Month

```
event_types <- unique(month_event_type_totals_all$type, nmax = 14)
event_types_change <- data.frame()

for (t in event_types) {
  t_group <- month_event_type_totals_all %>%
    filter(type == t) %>%
    arrange(month) %>%
    mutate(
      prev_events_prop = lag(events_prop),
      events_prop_change = prev_events_prop - events_prop,
      events_prop_change_pct = events_prop_change/events_prop,
      prev_num_events = lag(num_events),
      events_change = prev_num_events - num_events,
      events_change_pct = events_change/num_events,
      prev_num_actors = lag(num_actors),
      actors_change = prev_num_actors - num_actors,
      actors_change_pct = actors_change/num_actors,
      prev_num_repos = lag(num_repos),
      repos_change = prev_num_repos - num_repos,
      repos_change_pct = repos_change/num_repos
    ) %>%
    select(type, month,
           events_prop, events_prop_change, events_prop_change_pct,
           num_events, events_change, events_change_pct,
           num_actors, actors_change, actors_change_pct,
           num_repos, repos_change, repos_change_pct
           ) %>%
    filter(month != "Jun")

   event_types_change <- bind_rows(event_types_change, t_group)
}
```

```
ggplot(data = event_types_change,
       aes(x = month, fill=type, colour=type, group=type)) +
  geom_line(stat="identity", aes(y = events_prop_change_pct)) +
  ylab("Change From Prev Month") +
  xlab("Month") +
  ylim(-.5,.5)
```

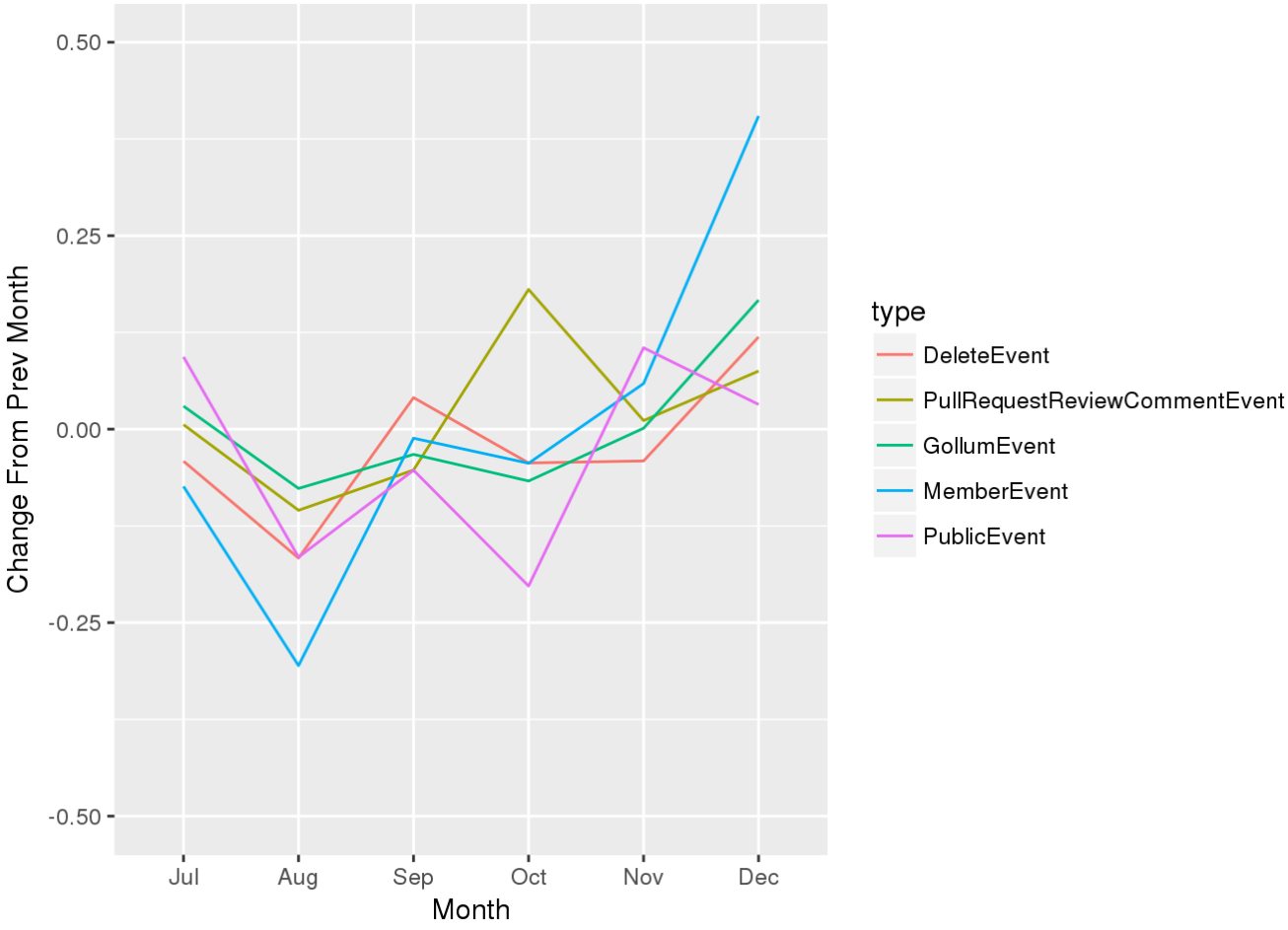



```
ggplot(data = event_types_change,
       aes(x = month, fill=type, colour=type, group=type)) +
  geom_line(stat="identity", aes(y = events_change_pct)) +
  ylab("Change From Prev Month") +
  xlab("Month") +
  ylim(-.5,.5)
```

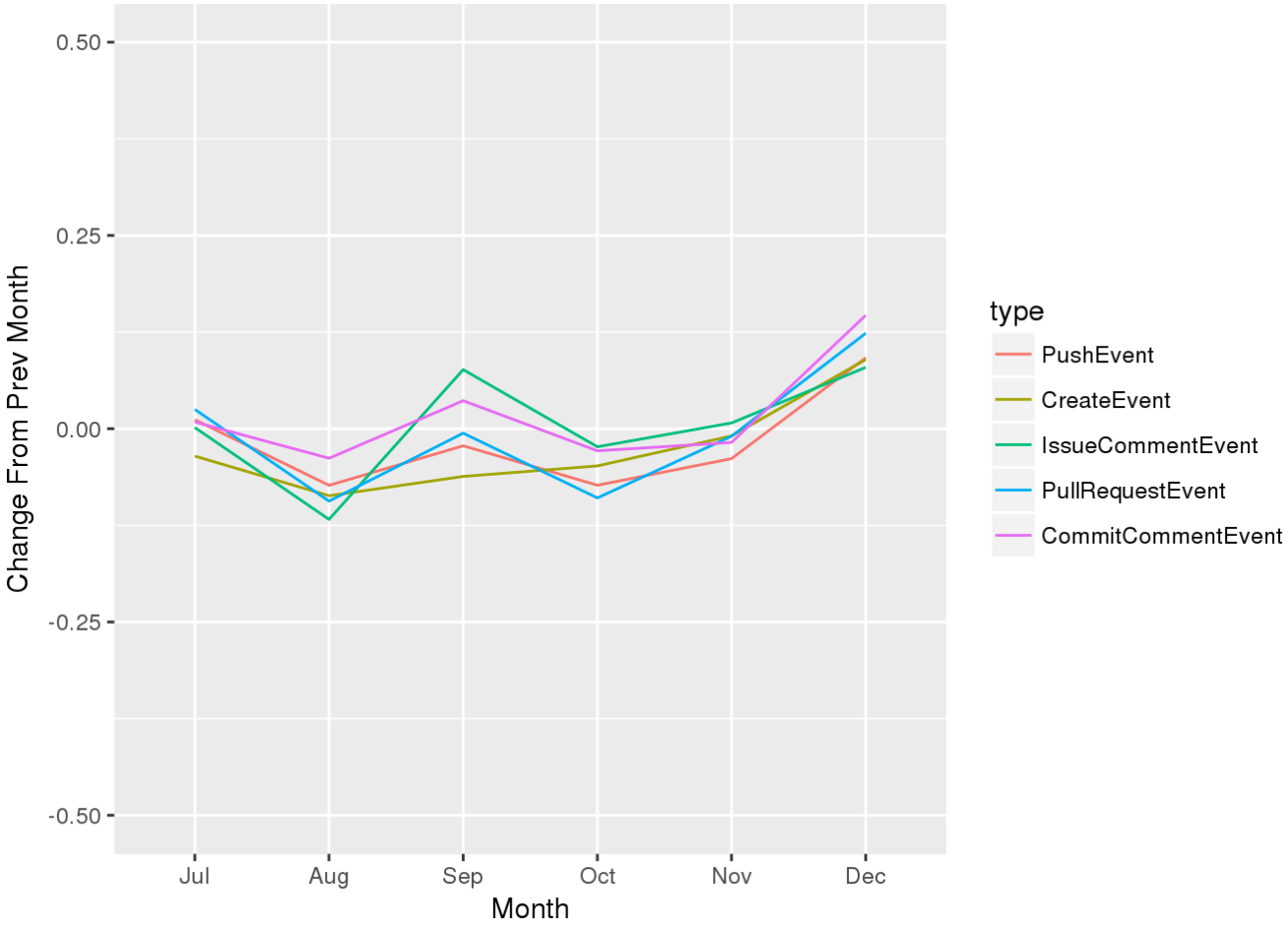



```
event_types_change_summary <- event_types_change %>%
  group_by(type) %>%
  summarise(event_change_sum_pct = sum(abs(events_change_pct)),
            event_change_sum_pct_avg = event_change_sum_pct/6)
```

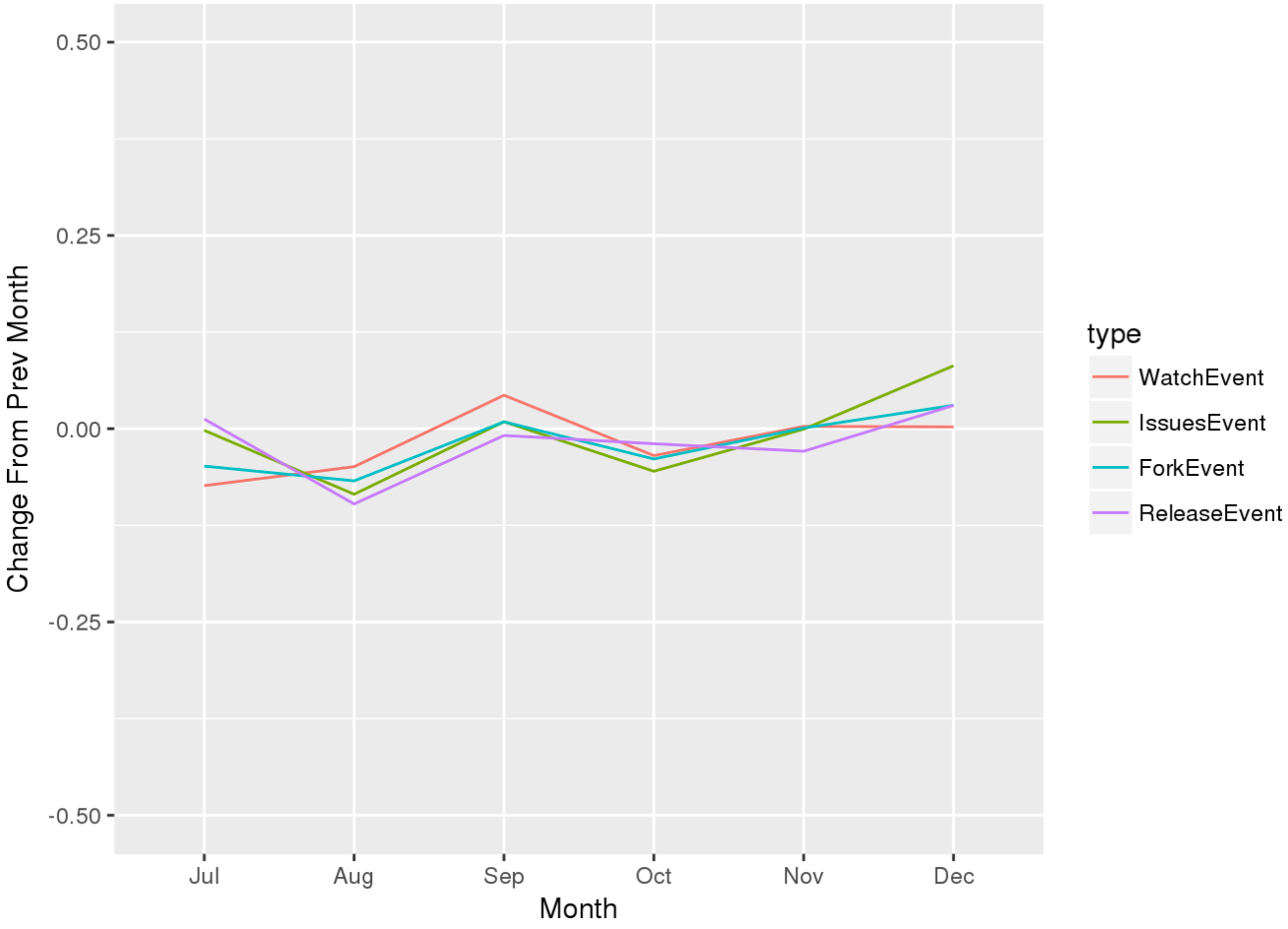
```
types_most_change <- event_types_change_summary %>%
  filter(event_change_sum_pct_avg > .06)

event_types_change_most <- event_types_change %>%
  filter(type %in% types_most_change$type)

ggplot(data = event_types_change_most,
       aes(x = month, fill=type, colour=type, group=type)) +
  geom_line(stat="identity", aes(y = events_change_pct)) +
  ylab("Change From Prev Month") +
  xlab("Month") +
  ylim(-.5, .5)
```




```
types_mid_change <- event_types_change_summary %>%
  filter(event_change_sum_pct_avg <= .06 & event_change_sum_pct_avg > .04)

event_types_change_mid <- event_types_change %>%
  filter(type %in% types_mid_change$type)

ggplot(data = event_types_change_mid,
       aes(x = month, fill=type, colour=type, group=type)) +
  geom_line(stat="identity", aes(y = events_change_pct)) +
  ylab("Change From Prev Month") +
  xlab("Month") +
  ylim(-.5,.5)
```



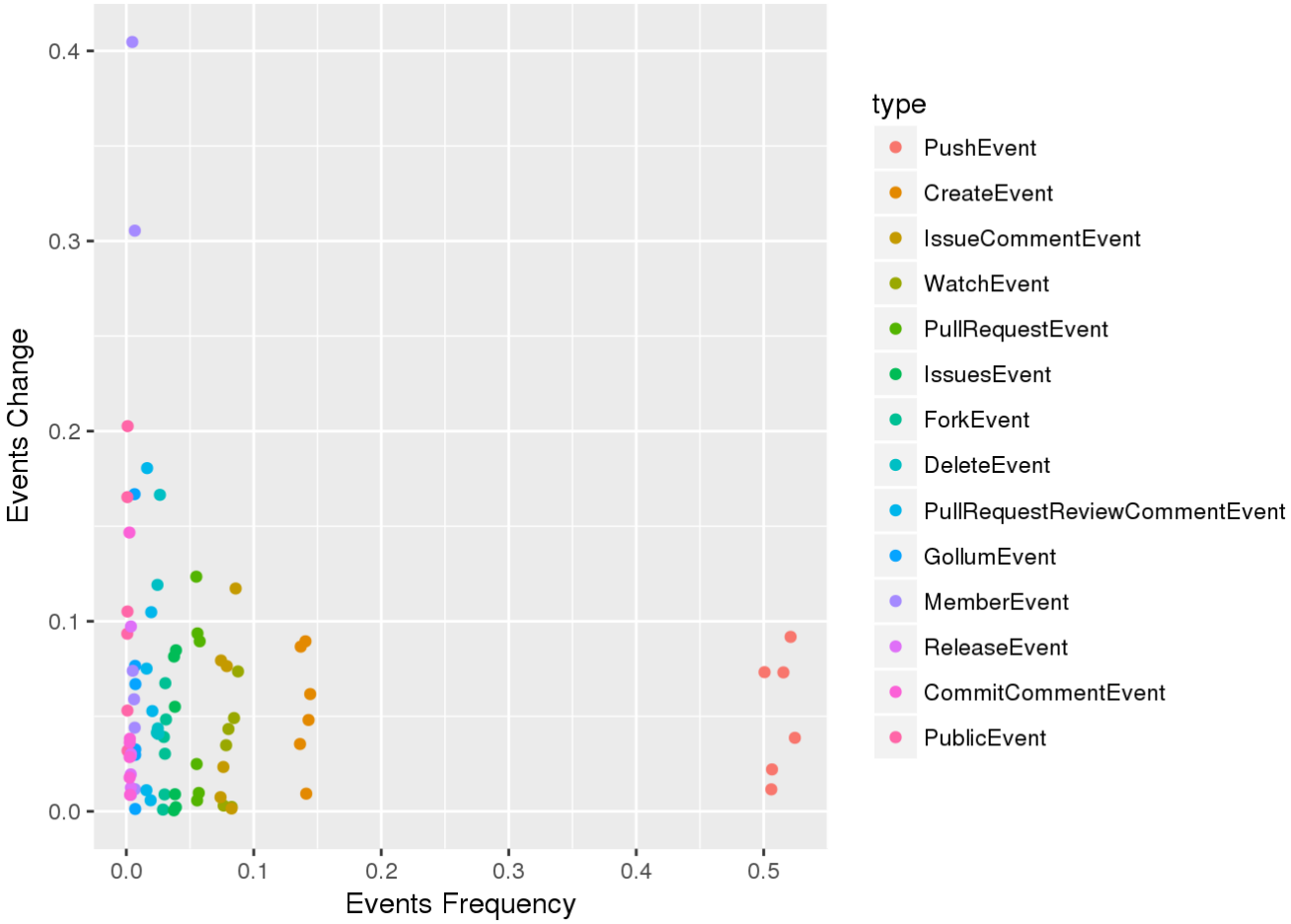

```
types_least_change <- event_types_change_summary %>%
  filter(event_change_sum_pct_avg <= .04)

event_types_change_least <- event_types_change %>%
  filter(type %in% types_least_change$type)

ggplot(data = event_types_change_least,
       aes(x = month, fill=type, colour=type, group=type)) +
  geom_line(stat="identity", aes(y = events_change_pct)) +
  ylab("Change From Prev Month") +
  xlab("Month") +
  ylim(-.5,.5)
```



Is there any relationship between the frequency of an event type and its variability?

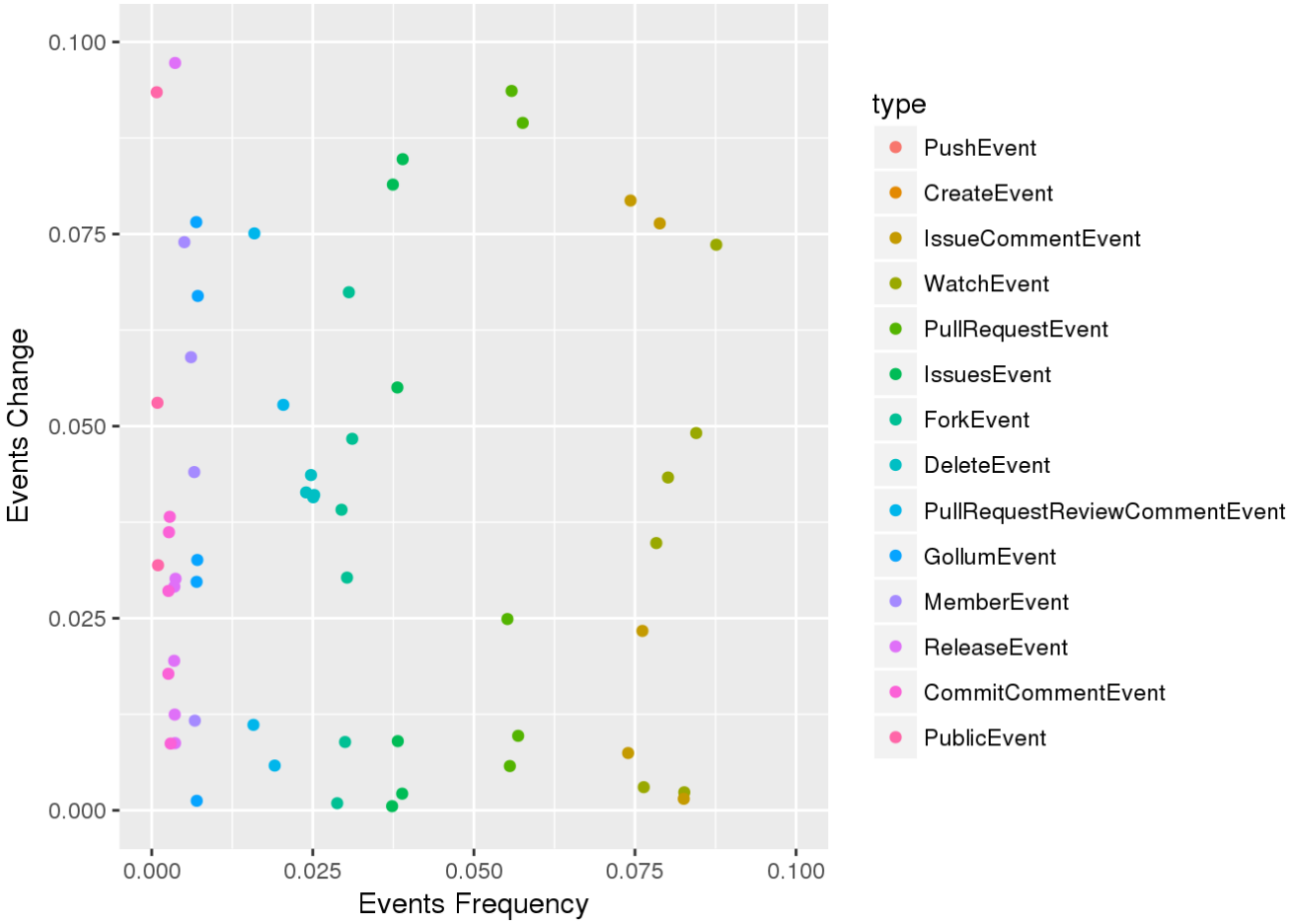
Based on the data available, there doesn't seem to be a huge correlation between frequency and variability. The outliers occur in the least frequent events, suggesting that if there is going to be variability it will be more likely to occur among the lower frequency event types than the higher frequency ones. This suggests that Member, Public, and Gollum events will be the least reliable events to pull samples from. This will be explored in the experiments section with more data points.

```
ggplot(data = event_types_change, aes(y=abs(events_change_pct), x=events_prop, fi
ll=type, colour=type, group=type)) +
  geom_point(stat="identity") +
  ylab("Events Change") +
  xlab("Events Frequency")
```




```
ggplot(data = event_types_change, aes(y=abs(events_change_pct), x=events_prop, fi
ll=type, colour=type, group=type)) +
  geom_point(stat="identity") +
  ylab("Events Change") +
  xlab("Events Frequency") +
  xlim(0,.1) +
  ylim(0,.1)
```

```
## Warning: Removed 25 rows containing missing values (geom_point).
```



How many unique contributors and unique repositories are represented in each type of event?

The charts below are sorted in by event frequency so they are more easily compared with the event type frequency chart above.

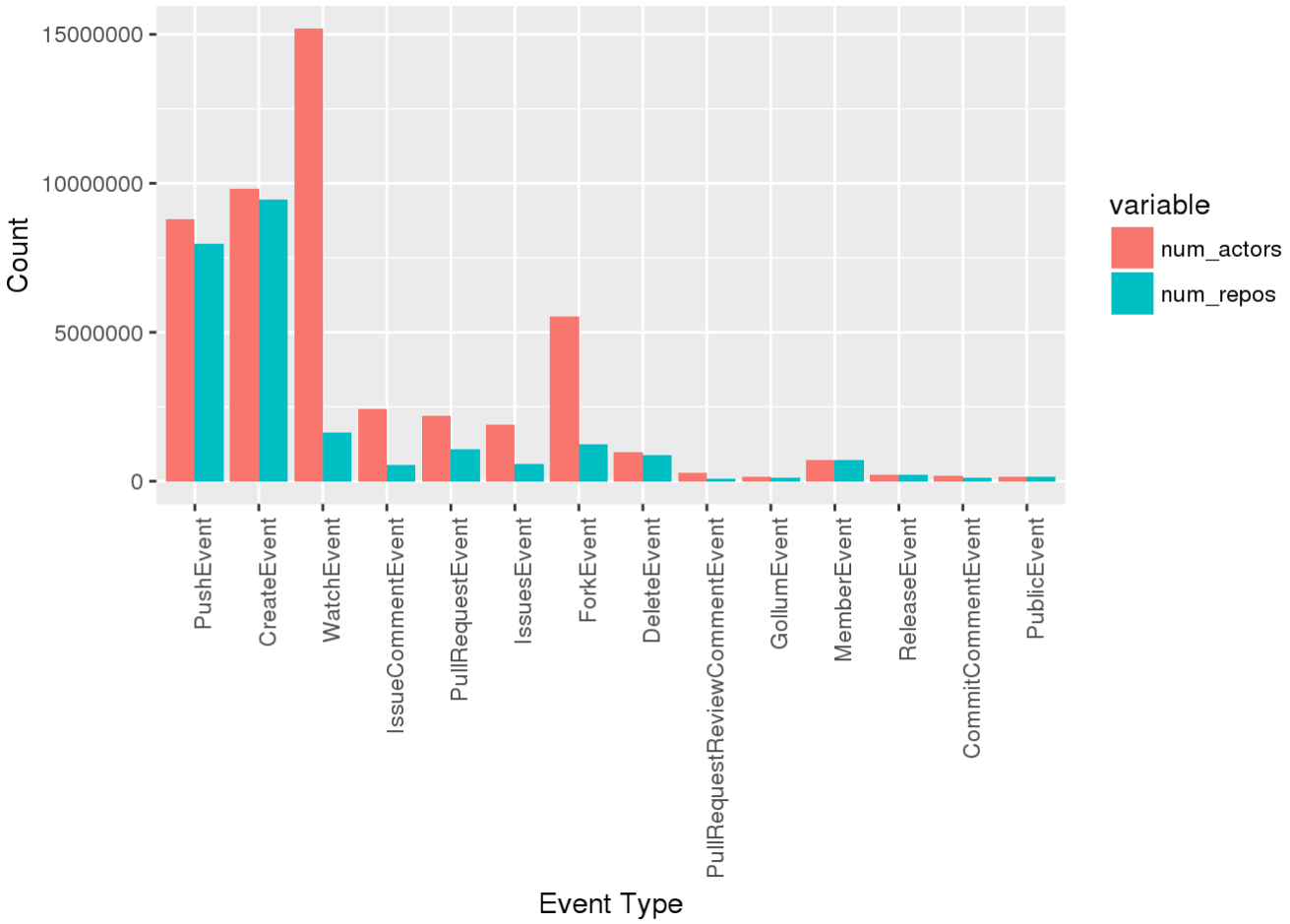
Actors and Repositories Per Event Type

Event types that have a high number of actors and a low number of repositories could indicate many actors interacting with a small number of repositories. Frequencies closer in value indicate a smaller number of actors per repository. This distribution suggests a possible correlation between the type of event and the number of unique repository actors. These values should be plotted for samples further in this analysis to determine if there is a correlation or not.

```
event_type_totals_all$type <- factor(event_type_totals_all$type,
  levels = unique(event_type_totals_all$type[order(event_type_totals_all$num_even
ts, decreasing=TRUE)]))

event_type_totals_long <- melt(event_type_totals_all)
```

```
## Using type as id variables
```

```
event_type_totals_actor_vs_repo <- event_type_totals_long %>%
  filter(variable == "num_actors" | variable == "num_repos")

ggplot(data = event_type_totals_actor_vs_repo, aes(x=type, y=value,
fill=variable))+
  geom_bar(stat="identity", position="dodge") +
  xlab("Event Type") +
  ylab("Count") +
  scale_y_continuous(labels=function(n){format(n, scientific = FALSE)}) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```

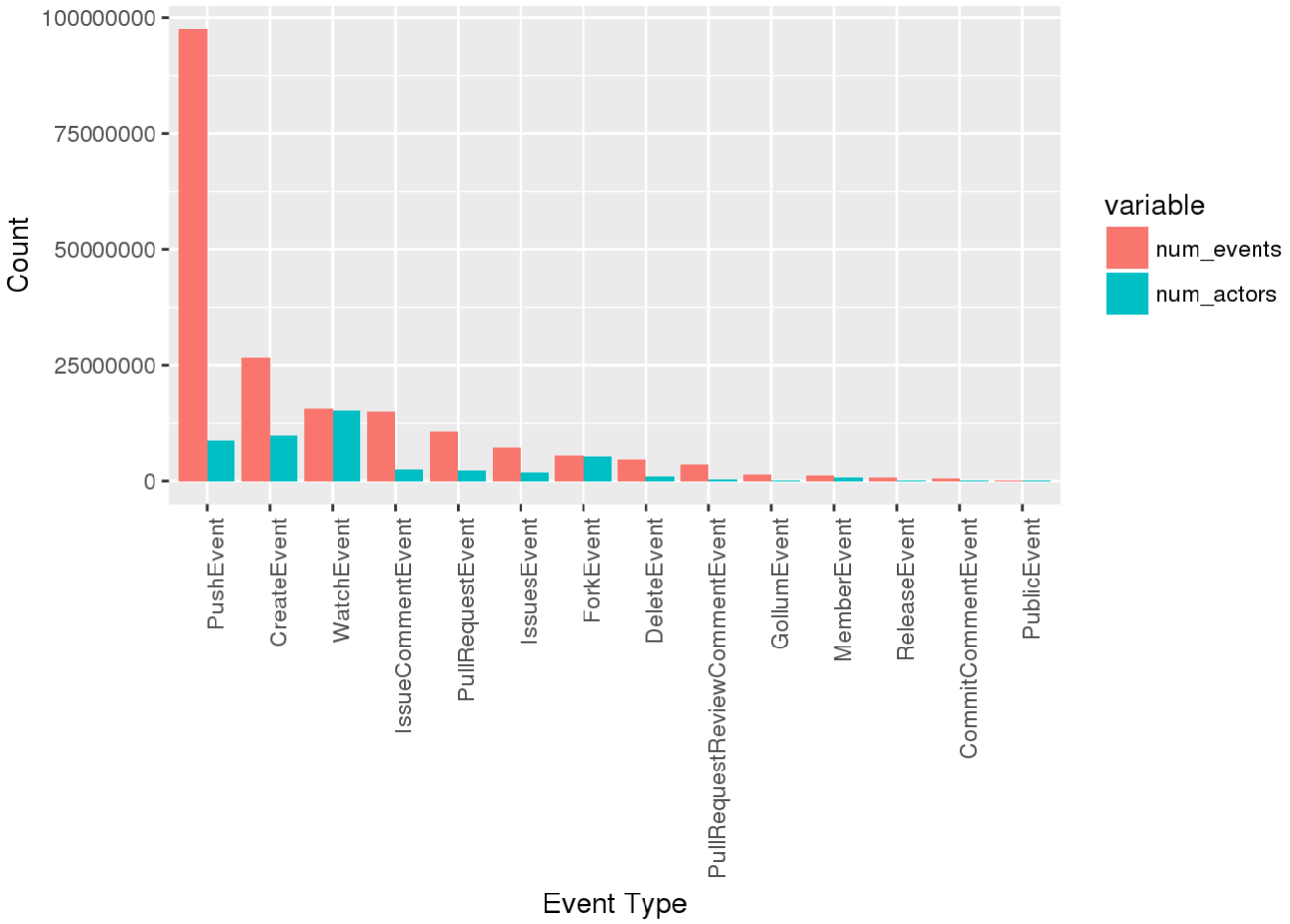


Actors and Events per Event Type

Comparing the number of actors to the number of events show that some activity types are performed multiple times by the same actors while others are less likely to be repeated by the same actor. In particular, WatchEvents seem fairly evenly distributed across the number of actors whereas PushEvents are distributed across around half of the same number of actors associated with WatchEvents.

It's not surprising to see a larger number of actors associated with WatchEvents over PushEvents because the action that triggers the WatchEvent (watching a repository) takes significantly less effort than a PushEvent. What is surprising is how the number of WatchEvents nearly equals the number of unique actors. This suggests that very active Github users may not be using this feature. It also suggests that the users that are using this feature might be less prolific and could be skewing the overall activity level for a repository. Further analysis on the repository demographics is needed to discover what this observation could indicate.

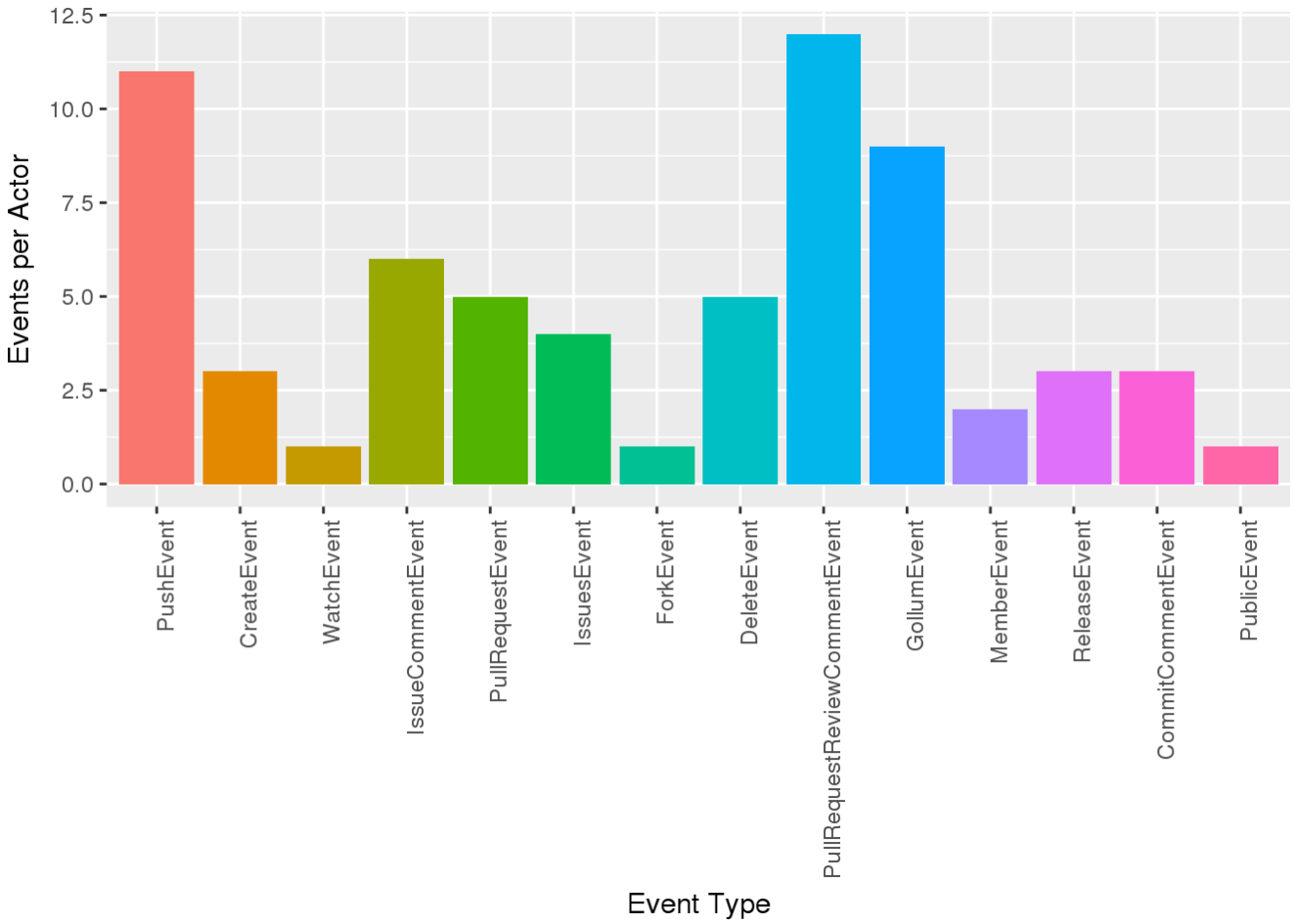
```
event_type_totals_actor_vs_events <- event_type_totals_long %>%
  filter(variable == "num_actors" | variable == "num_events")

ggplot(data = event_type_totals_actor_vs_events, aes(x=type, y=value, fill=variab
le)) +
  geom_bar(stat="identity", position="dodge") +
  xlab("Event Type") +
  ylab("Count") +
  scale_y_continuous(labels=function(n){format(n, scientific = FALSE)}) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```



Events Per Actor

This is a ratio based on the above visualization, the frequency of unique actors per event type compared with the frequency of events of that type. These are sorted by most frequent overall event type. A high events-to-actor ratio suggests the event type only represents the activity of a small group of individuals. A smaller events-to-actor ratio suggests the event type is spread across a larger number of individuals. The most spread out events appear to be Watch, Fork, and Public events.

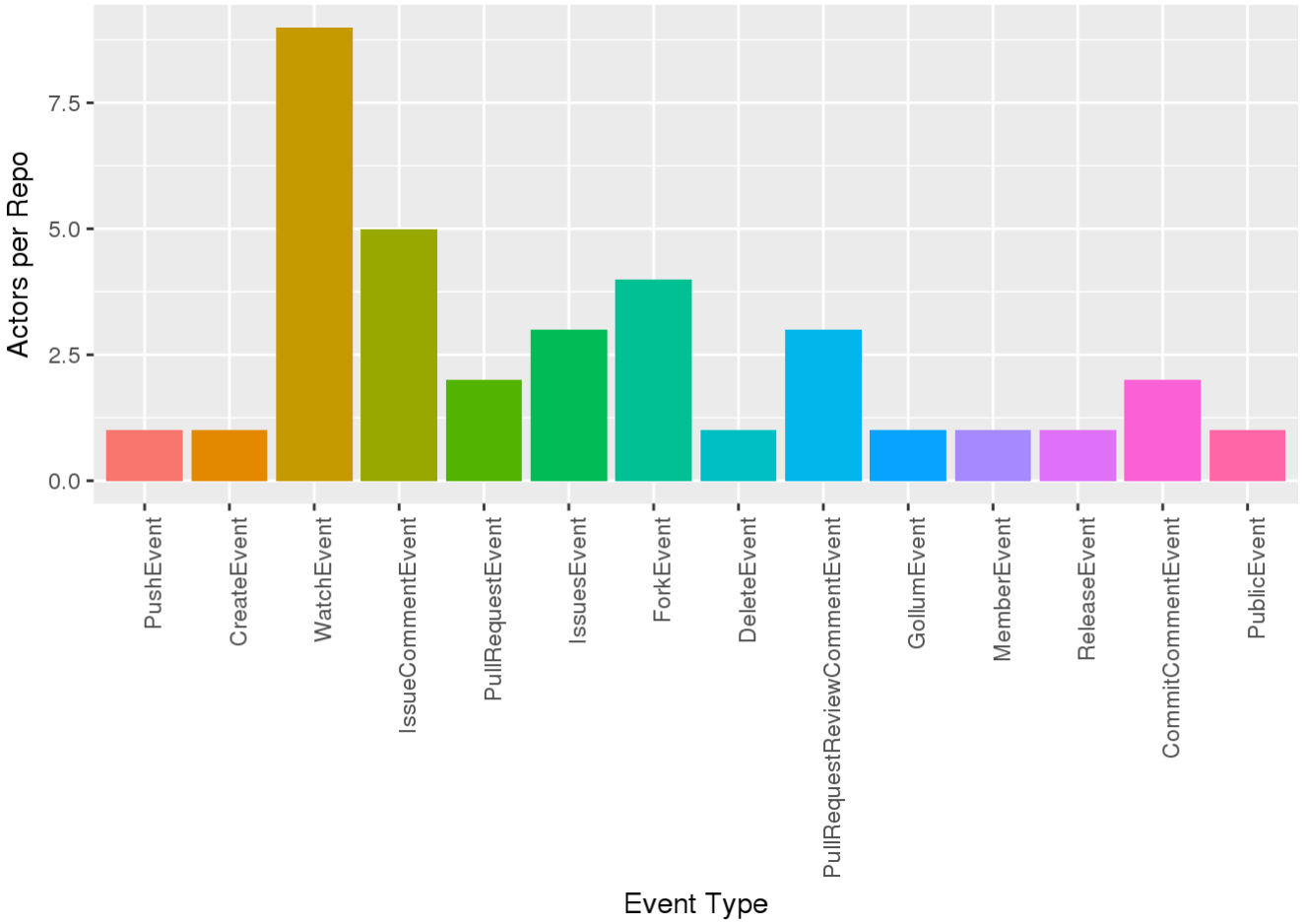
```
ggplot(data = event_type_totals_all, aes(x=type, y=events_to_actor, fill=type)) +
  geom_bar(stat="identity") +
  theme(legend.position="none") +
  ylab("Events per Actor") +
  xlab("Event Type") +
  scale_y_continuous(labels=function(n){format(n, scientific = FALSE)}) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```



Actors per Repository

These are sorted by most frequent overall event type. The actors to repository ratio represents the frequency of unique actors compared frequency of unique repositories within a given event type. A high actors-to-repository ratio suggests the event type represents the activity of a large group of individuals associating with a smaller number of repositories. A smaller actors-to-repository ratio suggests the event type is spread across a smaller number of individuals associating with a larger number of repositories. This ratio could be a clue as to what types of events occur more frequently with projects that have a larger number of contributors. Watch events clearly have the highest Actor-to-Repository ratio. Fork events and the comment events are the next group that have a high ratio.

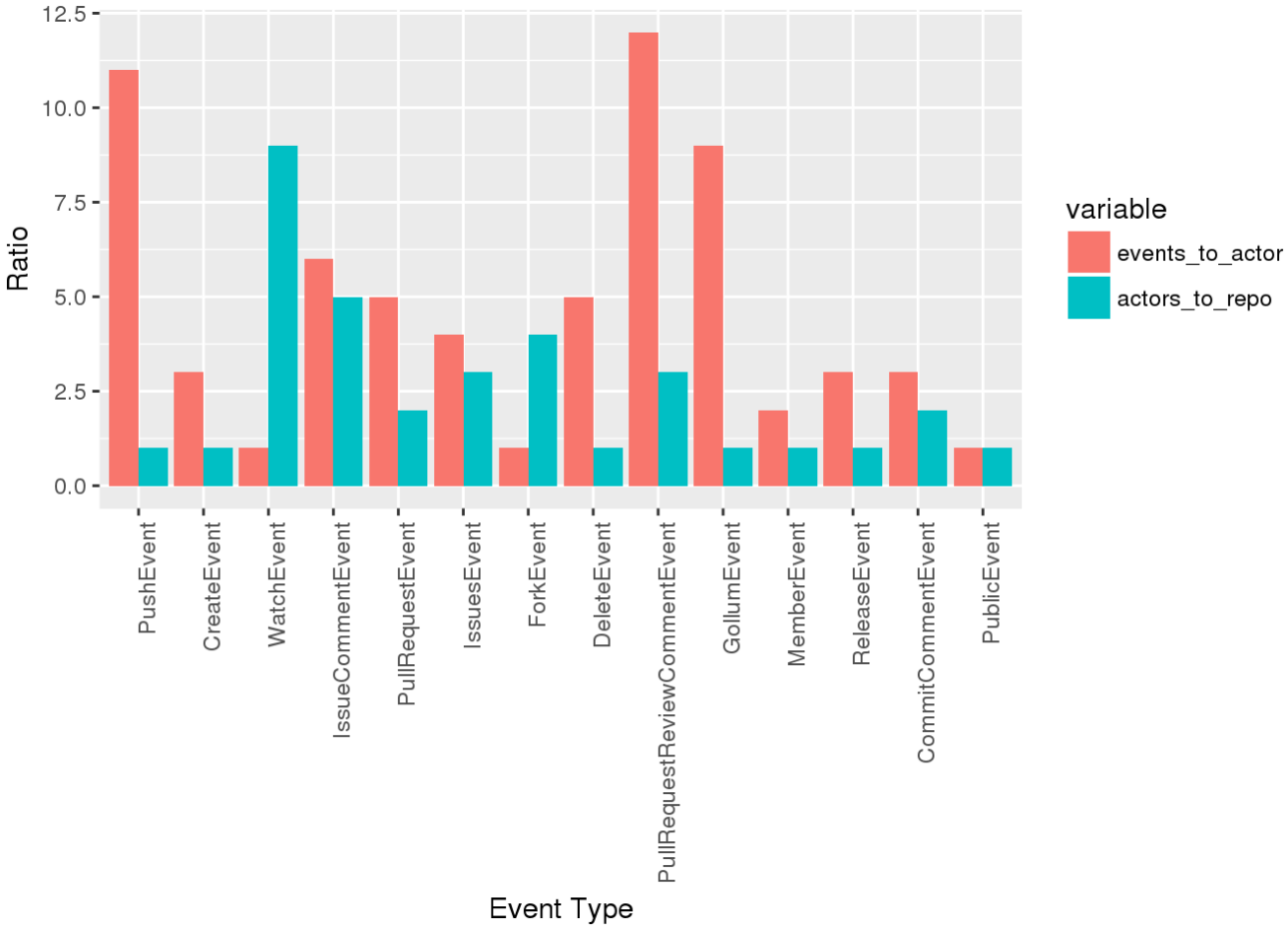
```
ggplot(data = event_type_totals_all, aes(x=type, y=actors_to_repo, fill=type)) +
  geom_bar(stat="identity") +
  theme(legend.position="none") +
  ylab("Actors per Repo") +
  xlab("Event Type") +
  scale_y_continuous(labels=function(n){format(n, scientific = FALSE)}) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```



Events per Actor vs Actors per Repository

This is just an overlay of the two ratios to see how they compare for each event type. If projects with more contributors should have a high actors to repository ratio and a low events to actor ratio, then the event types that would indicate higher participation levels would be the WatchEvent and ForkEvent types. Lower participation levels, meaning projects that have a smaller number of contributors, seem to be indicated by the PushEvent type. The IssueEvent and IssueComment event types are interesting in that they seem to hover right in the middle with their values being pretty close. The distribution of event types and these ratios plotted over the full 6 months should be further examined to see if there is a lot of variation from month to month.

```
event_type_totals_ratios <- event_type_totals_long %>%
  filter(variable == "actors_to_repo" | variable == "events_to_actor")

ggplot(data = event_type_totals_ratios, aes(x=type, y=value, fill=variable)) +
  geom_bar(stat="identity", position="dodge") +
  xlab("Event Type") +
  ylab("Ratio") +
  scale_y_continuous(labels=function(n){format(n, scientific = FALSE)}) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```

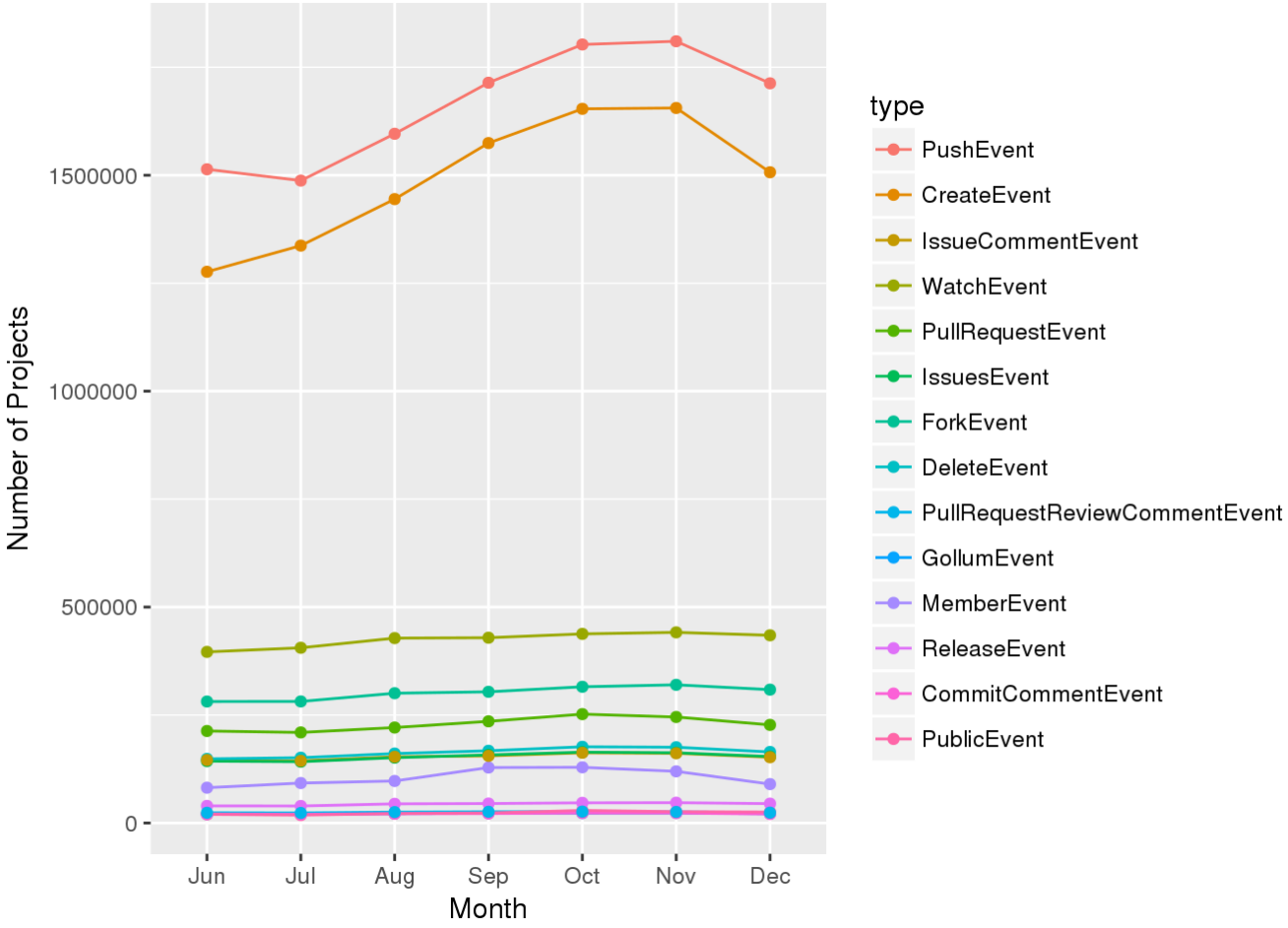


How much variation do we see in the repositories and actors associated with these events each month?

In other words, is the overall pattern for the 6 month period consistent or is there a lot of variation in who is doing what from month to month? It looks to be a similar pattern as the events frequency change with the events with the most actors and the least actors having a higher probability for greater variability from month to month.

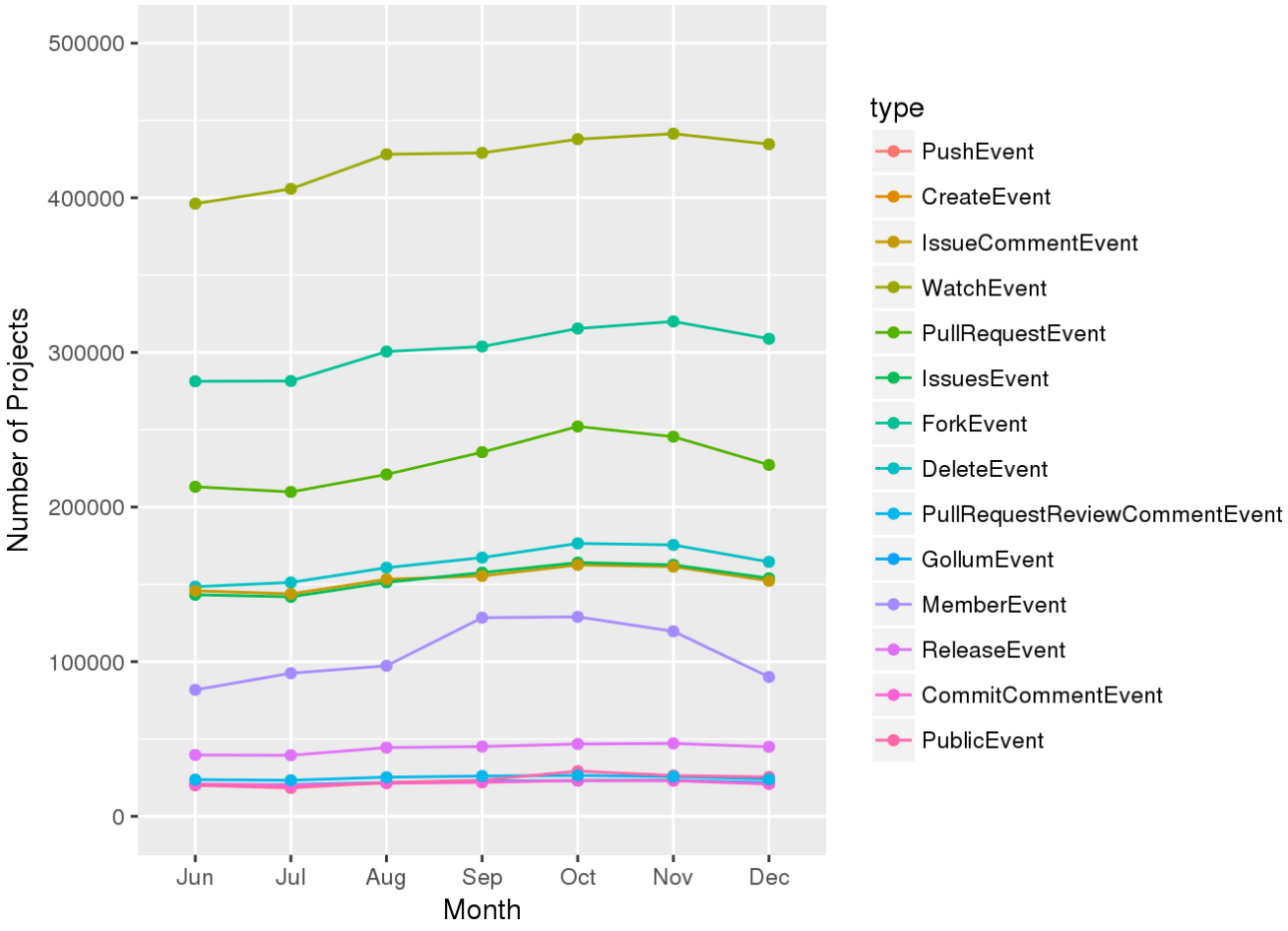
Repositories Per Month

```
ggplot(data = month_event_type_totals_all,
       aes(x=month,
       y=num_repos,
       fill=type,
       colour=type,
       group=type)) +
  geom_line(stat="identity") +
  geom_point(stat="identity") +
  xlab("Month") +
  ylab("Number of Projects") +
  scale_y_continuous(labels=function(n){format(n, scientific = FALSE)})
```



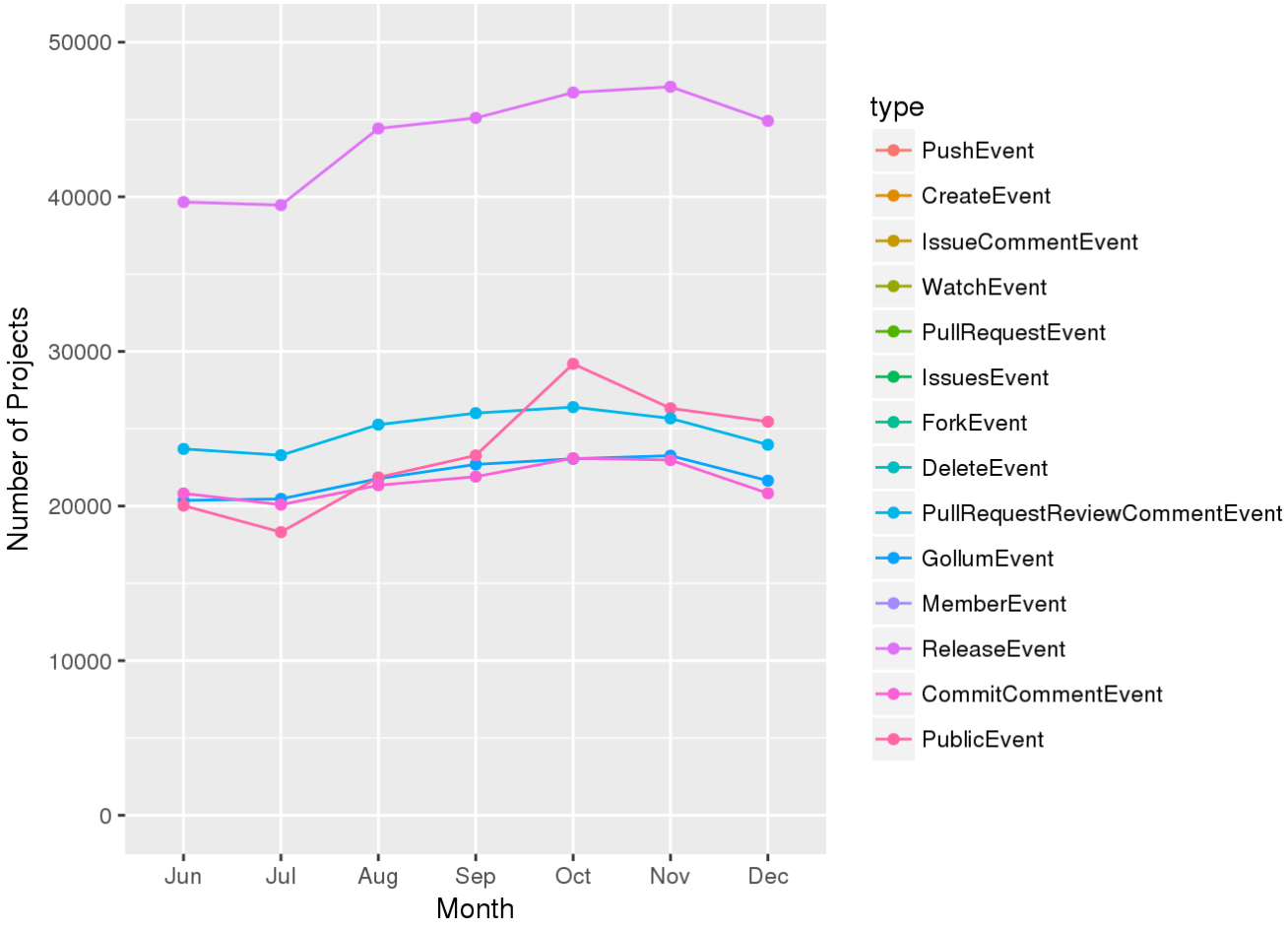

```
ggplot(data = month_event_type_totals_all,
       aes(x=month,
       y=num_repos,
       fill=type,
       colour=type,
       group=type)) +
  geom_line(stat="identity") +
  geom_point(stat="identity") +
  xlab("Month") +
  ylab("Number of Projects") +
  scale_y_continuous(labels=function(n){format(n, scientific = FALSE)}, limits =
c(0,500000) )
```

```
## Warning: Removed 14 rows containing missing values (geom_point).
```



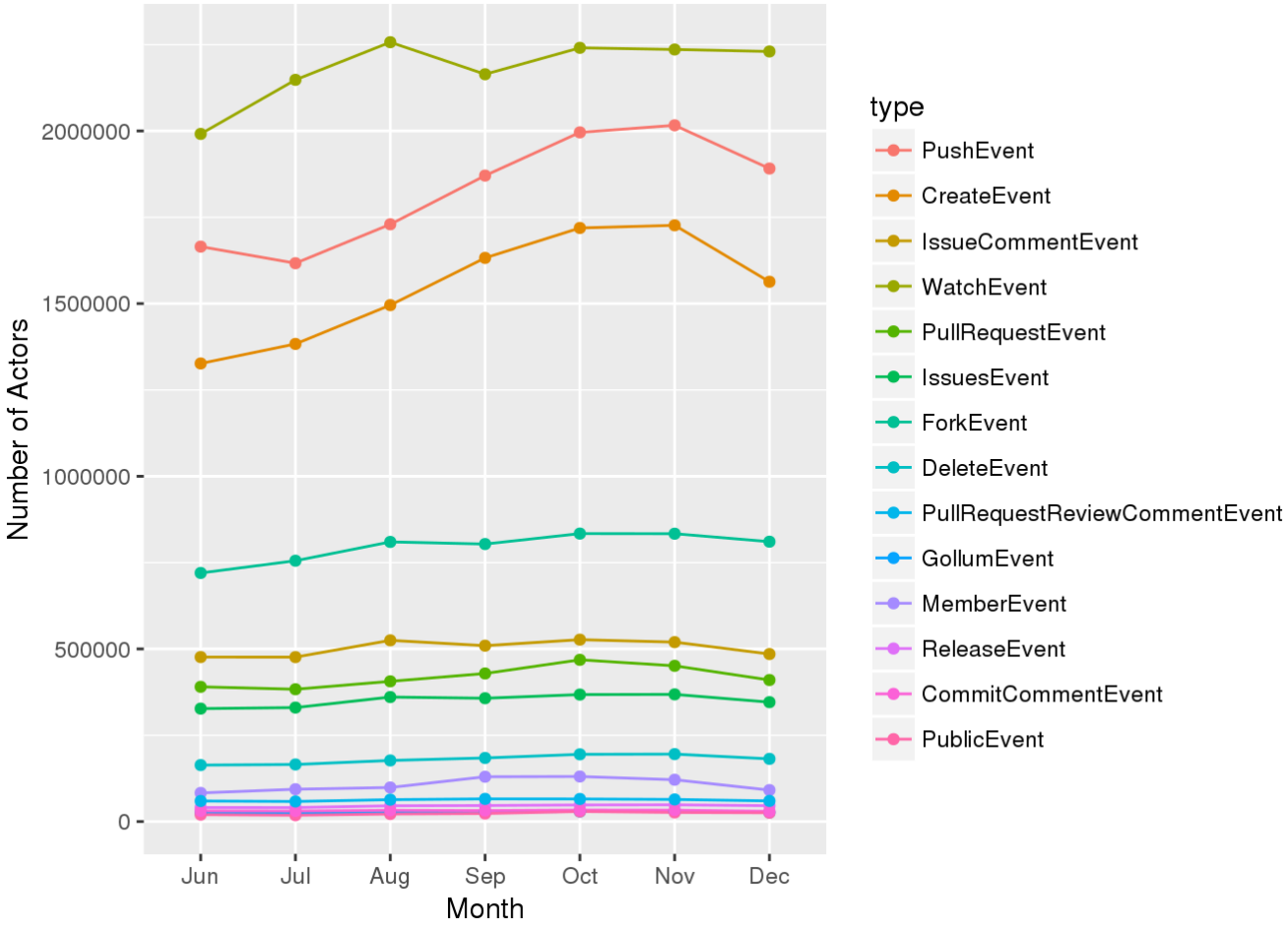

```
ggplot(data = month_event_type_totals_all,
       aes(x=month,
       y=num_repos,
       fill=type,
       colour=type,
       group=type)) +
  geom_line(stat="identity") +
  geom_point(stat="identity") +
  xlab("Month") +
  ylab("Number of Projects") +
  scale_y_continuous(labels=function(n){format(n, scientific = FALSE)}, limits =
c(0,50000) )
```

```
## Warning: Removed 63 rows containing missing values (geom_point).
```



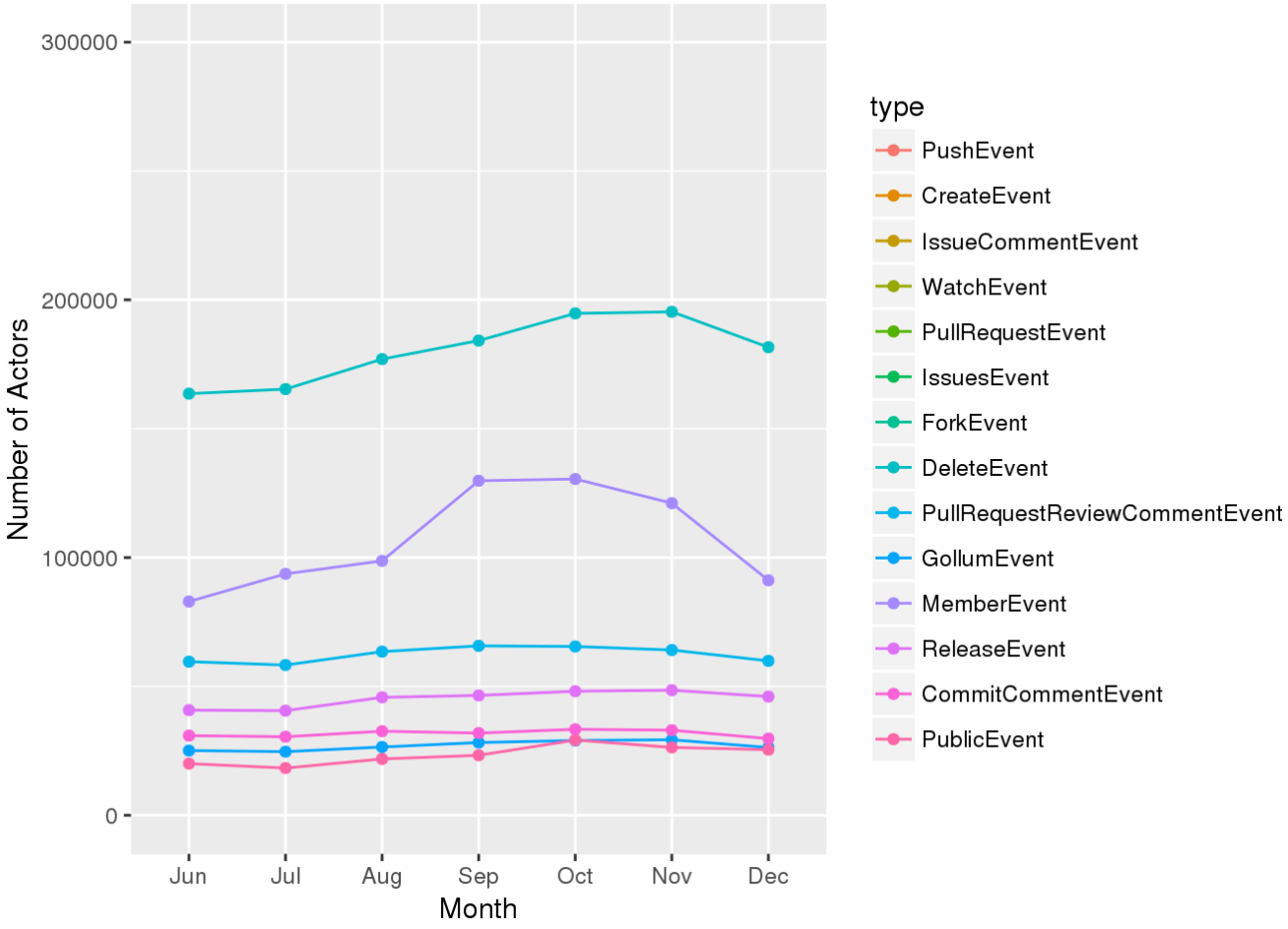
Actors Per Month

```
ggplot(data = month_event_type_totals_all,
       aes(x=month,
       y=num_actors,
       fill=type,
       colour=type,
       group=type)) +
  geom_line(stat="identity") +
  geom_point(stat="identity") +
  xlab("Month") +
  ylab("Number of Actors") +
  scale_y_continuous(labels=function(n){format(n, scientific = FALSE)})
```



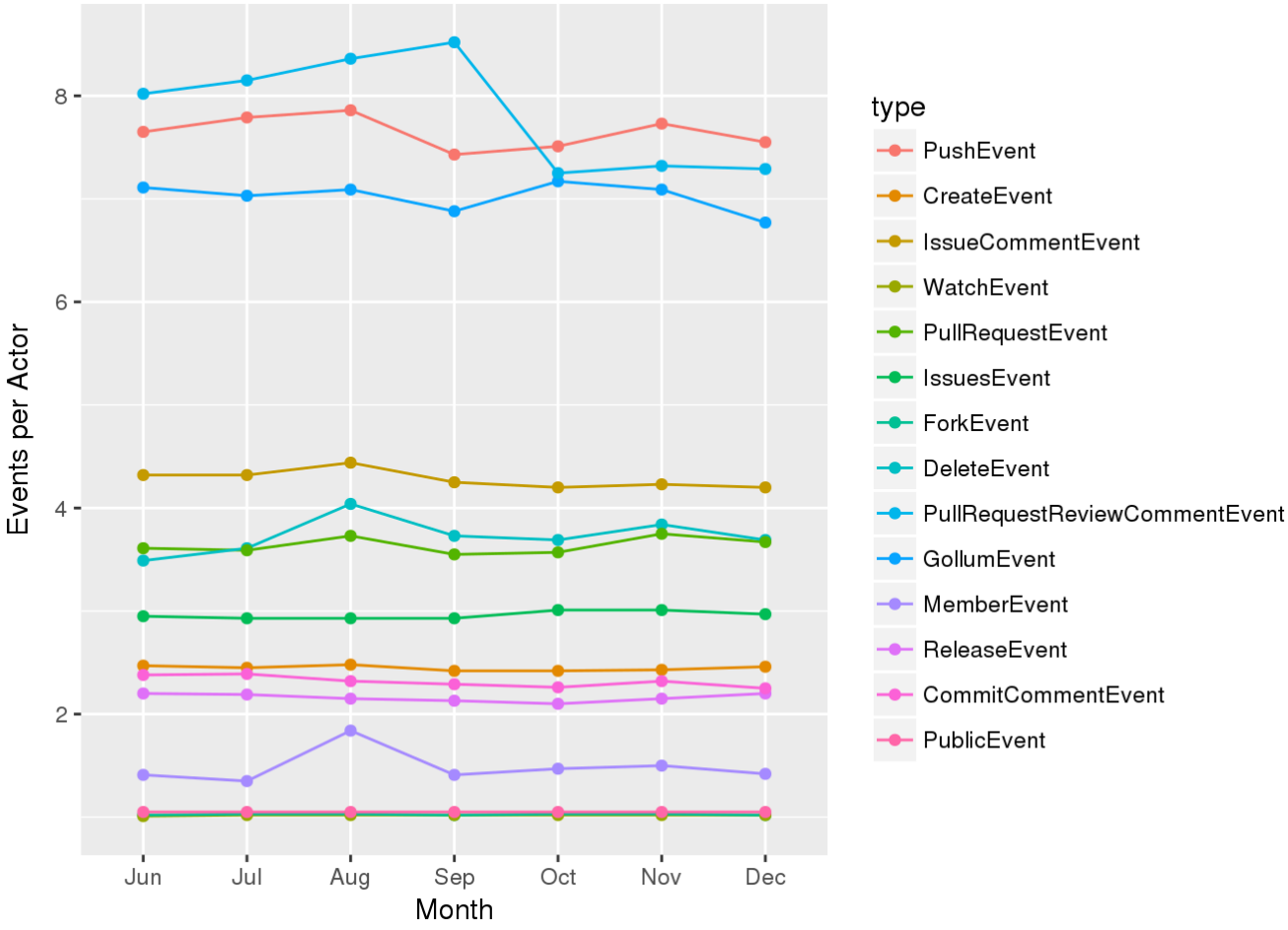

```
ggplot(data = month_event_type_totals_all,
       aes(x=month,
       y=num_actors,
       fill=type,
       colour=type,
       group=type)) +
  geom_line(stat="identity") +
  geom_point(stat="identity") +
  xlab("Month") +
  ylab("Number of Actors") +
  scale_y_continuous(labels=function(n){format(n, scientific = FALSE)},
limit=c(0, 300000))
```

```
## Warning: Removed 49 rows containing missing values (geom_point).
```



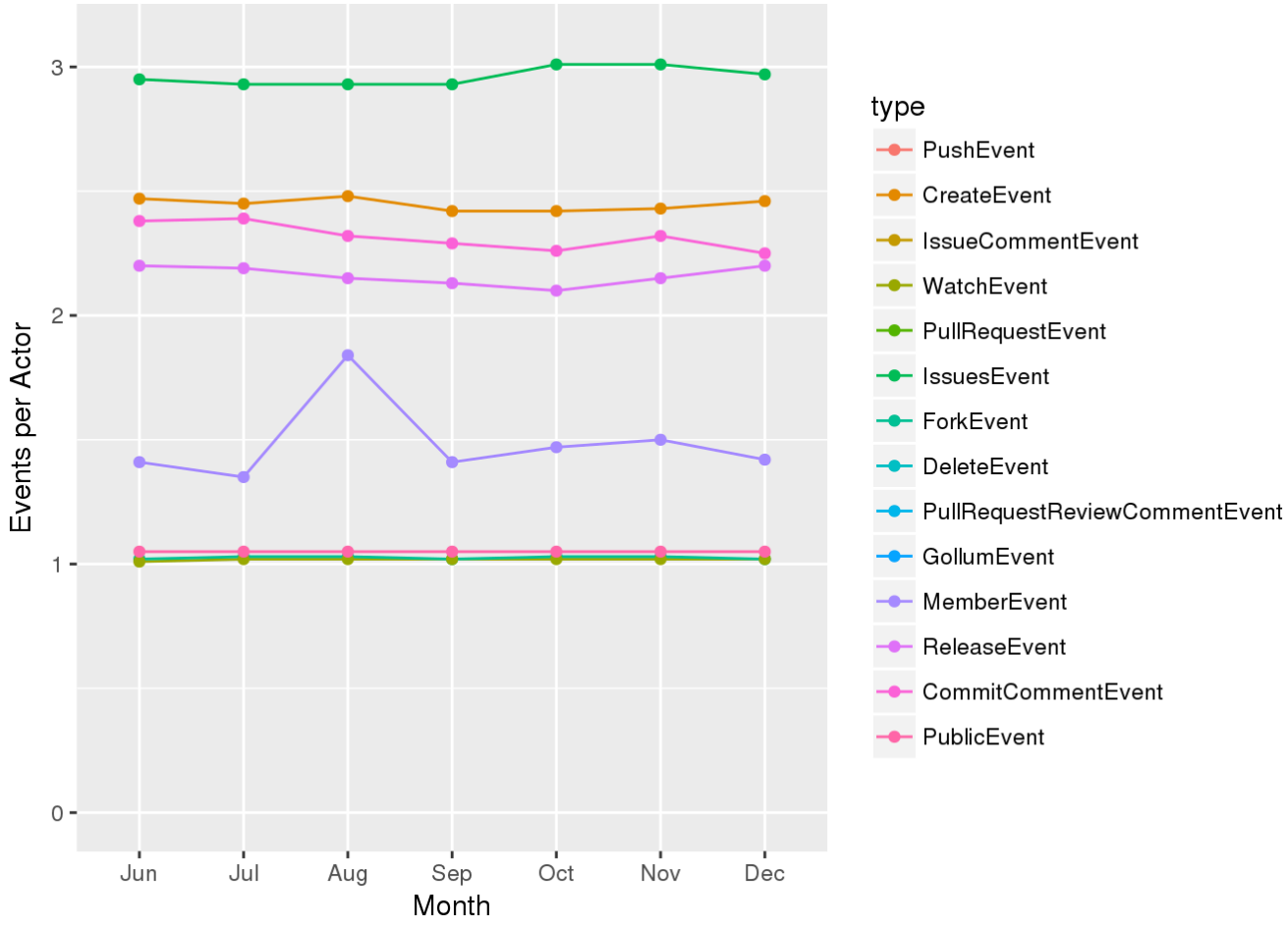
Events per Actor

```
ggplot(data = month_event_type_totals_all,
       aes(x=month,
       y=events_to_actor,
       fill=type,
       colour=type,
       group=type)) +
  geom_line(stat="identity") +
  geom_point(stat="identity") +
  xlab("Month") +
  ylab("Events per Actor") +
  scale_y_continuous(labels=function(n){format(n, scientific = FALSE)})
```



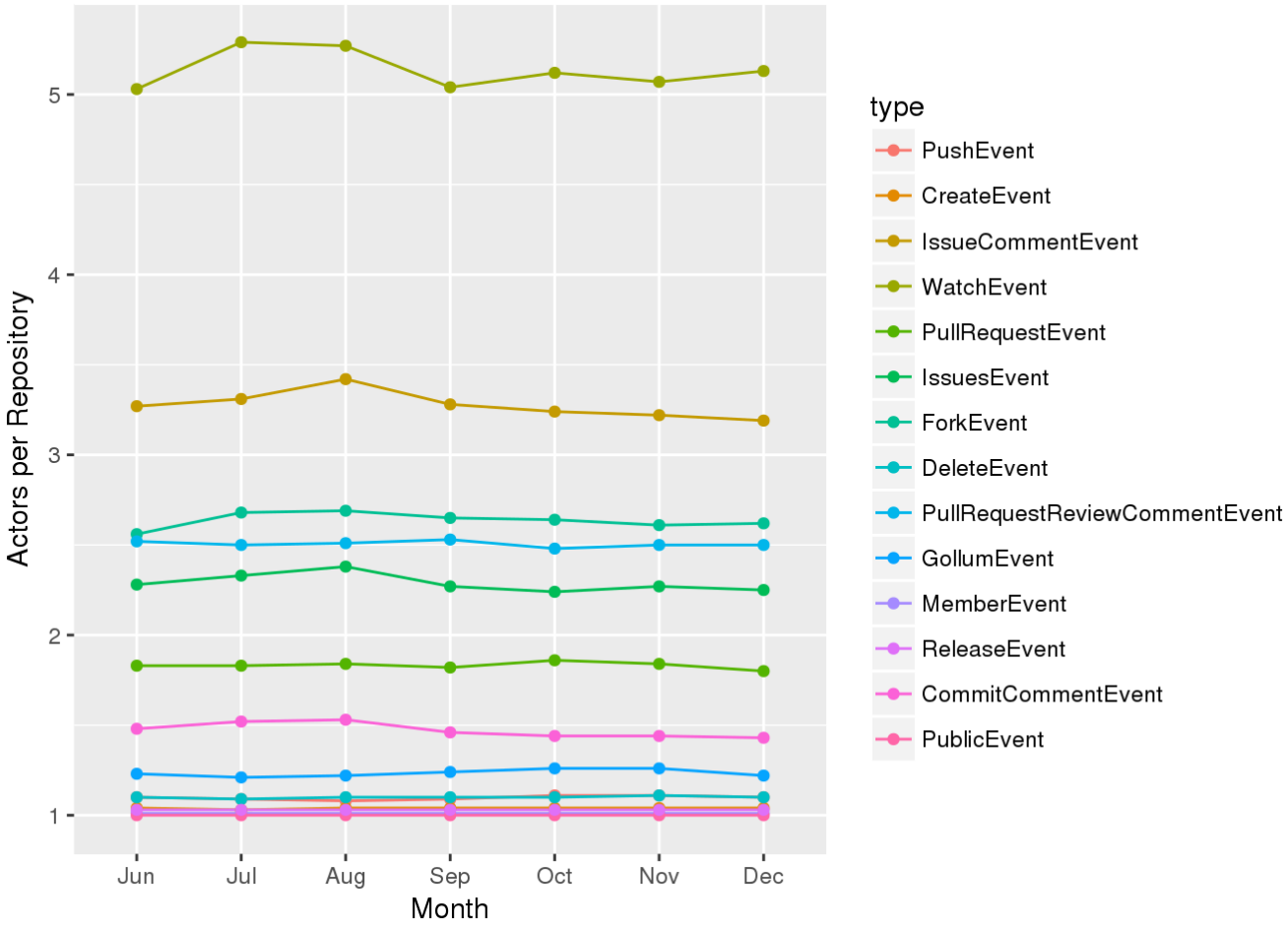

```
ggplot(data = month_event_type_totals_all,
       aes(x=month,
       y=events_to_actor,
       fill=type,
       colour=type,
       group=type)) +
  geom_line(stat="identity") +
  geom_point(stat="identity") +
  xlab("Month") +
  ylab("Events per Actor") +
  scale_y_continuous(labels=function(n){format(n, scientific = FALSE)},
limit=c(0,3.1))
```

```
## Warning: Removed 42 rows containing missing values (geom_point).
```



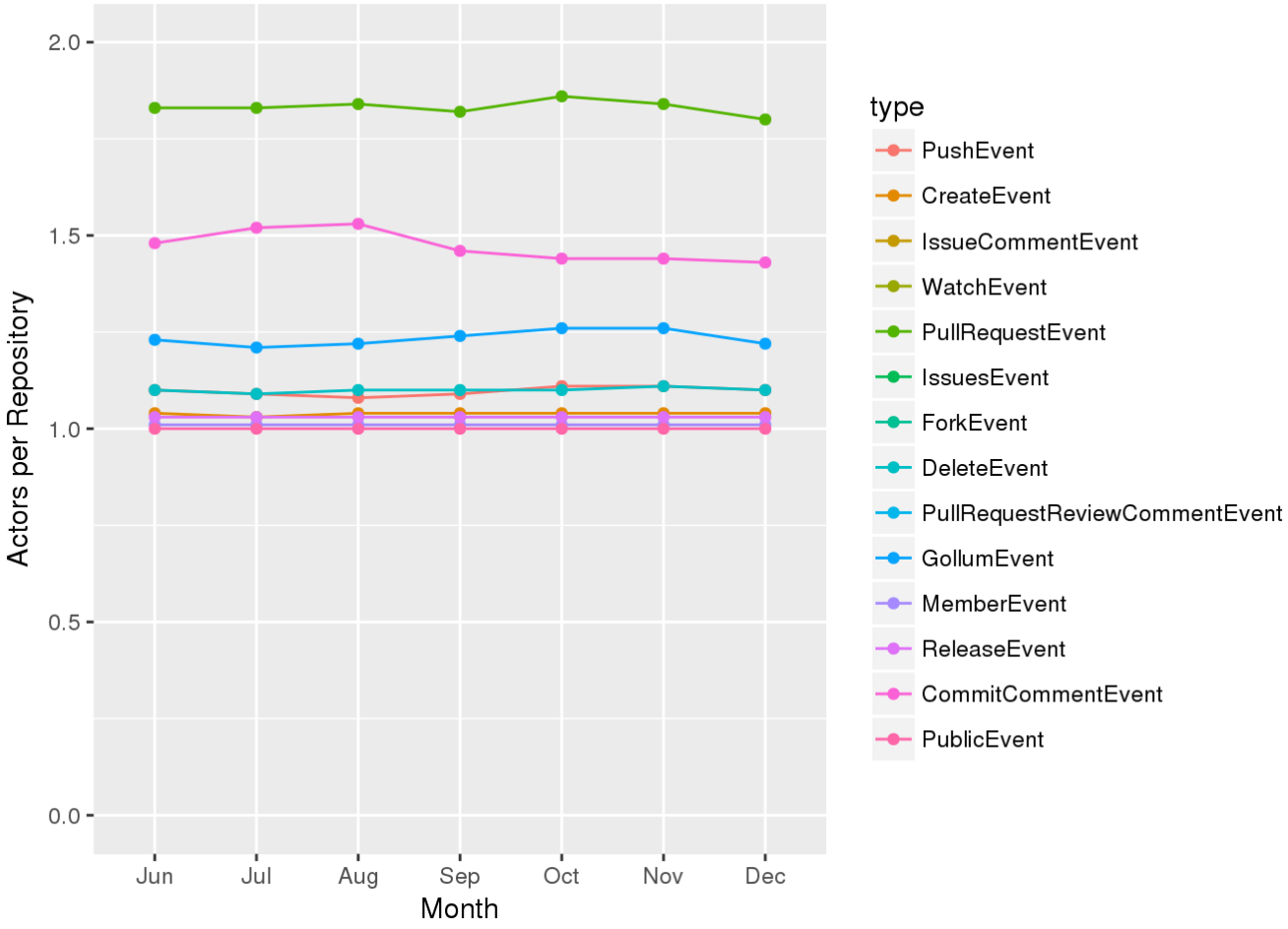
Actors per Repository

```
ggplot(data = month_event_type_totals_all,
       aes(x=month,
       y=actors_to_repo,
       fill=type,
       colour=type,
       group=type)) +
  geom_line(stat="identity") +
  geom_point(stat="identity") +
  xlab("Month") +
  ylab("Actors per Repository") +
  scale_y_continuous(labels=function(n){format(n, scientific = FALSE)})
```




```
ggplot(data = month_event_type_totals_all,
       aes(x=month,
       y=actors_to_repo,
       fill=type,
       colour=type,
       group=type)) +
  geom_line(stat="identity") +
  geom_point(stat="identity") +
  xlab("Month") +
  ylab("Actors per Repository") +
  scale_y_continuous(labels=function(n){format(n, scientific = FALSE)}, limits =
c(0,2))
```

```
## Warning: Removed 35 rows containing missing values (geom_point).
```



Events Sampling Experiment

Given the above information, we should analyze the distributions of events samples. For sampling methodology, see the events_analysis_data.Rmd workbook in this repository.

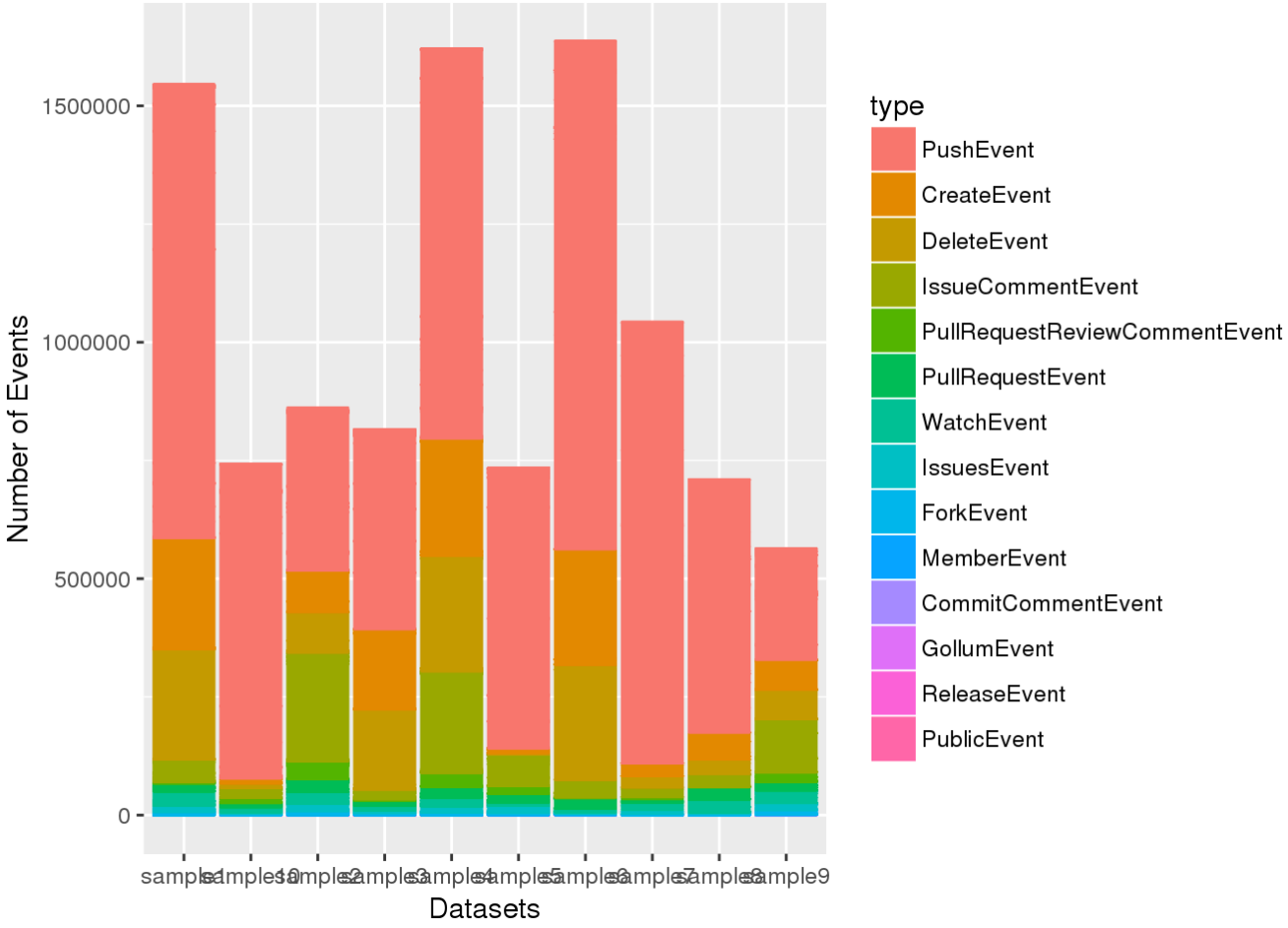
This experiment consists of 10 samples of 100 events taken from all events in the period of June 2016 through December 2016. Additional metrics were computed for each repository based on that repository's total activity over the 6 month period.

```
events_repo_samples <- readRDS("events_repo_samples.rds")

events_repo_samples_by_repo <- events_repo_samples %>%
  group_by(dataset, repo_name) %>%
  summarise(num_repo_actors = max(num_repo_actors),
            num_repo_events = max(num_repo_events))
```

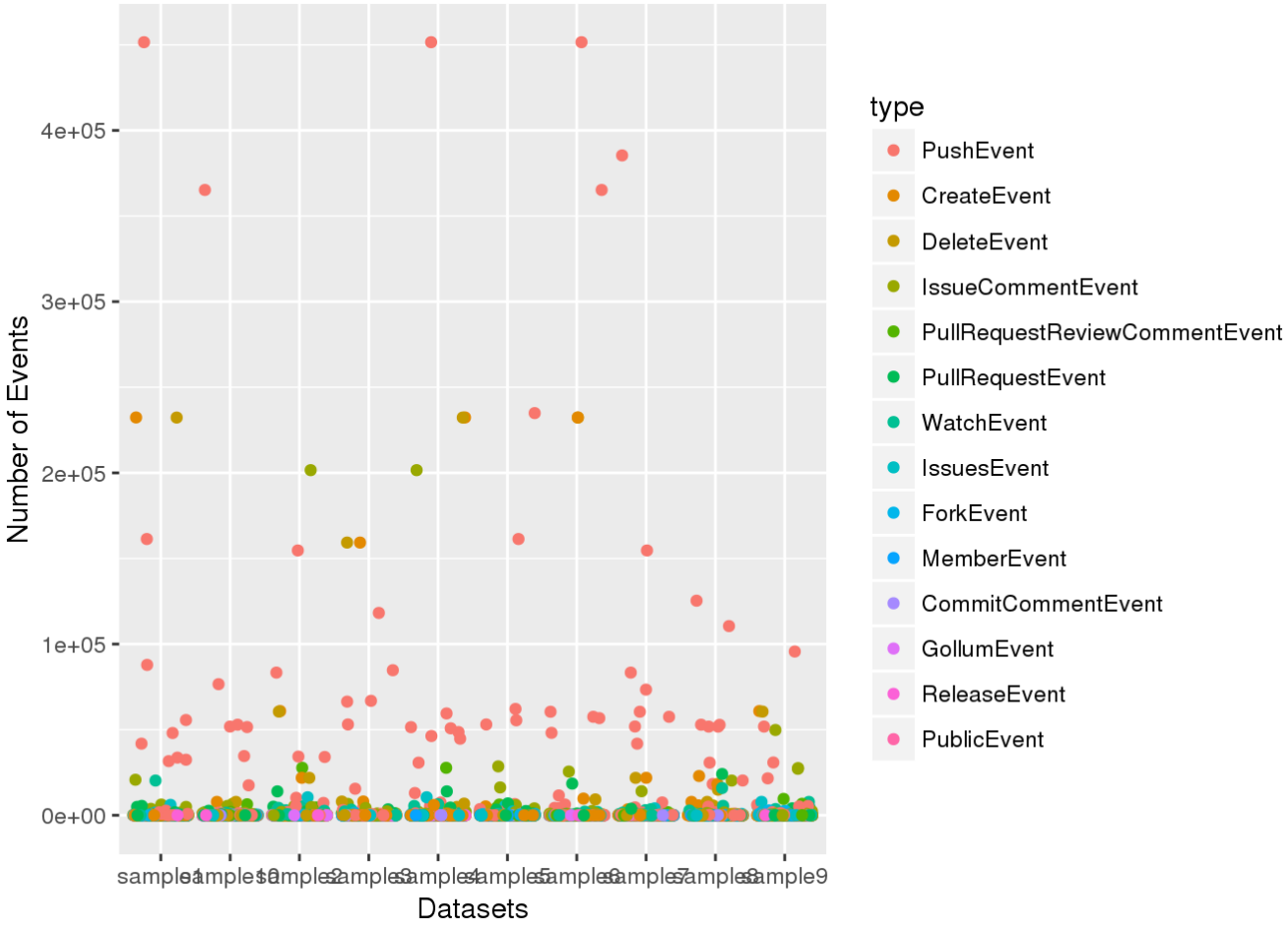
Event Type Distribution

Plotting the total number of events represented by each sample and the event types, we see a lot of variation between the samples. The top represented events appear to be Push, Create, Delete, IssueComment, PullRequestReviewComment, PullRequest, and Watch.

```
events_repo_samples$type <- factor(events_repo_samples$type,
   levels = unique(events_repo_samples$type[order(events_repo_samples$num_events,
 decreasing=TRUE)]))

ggplot(data = events_repo_samples,
       aes(x = dataset,
           y = num_events,
           fill = type, color = type, group = type)) +
  geom_bar(stat="identity") +
  xlab("Datasets") +
  ylab("Number of Events")
```

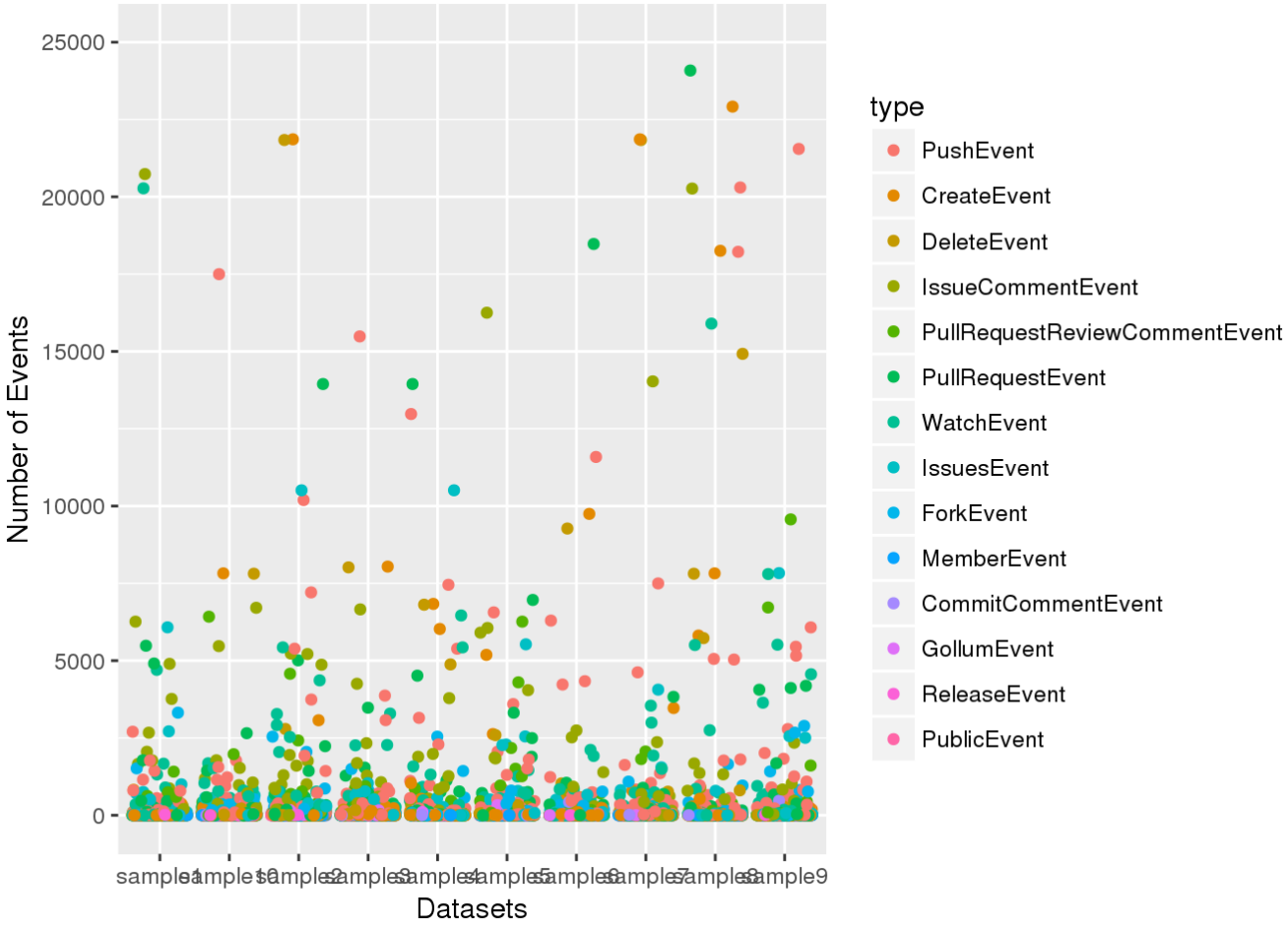



```
ggplot(data = events_repo_samples,
       aes(x = dataset,
           y = num_events,
           fill = type, color = type, group = type)) +
  geom_jitter(stat="identity") +
  xlab("Datasets") +
  ylab("Number of Events")
```



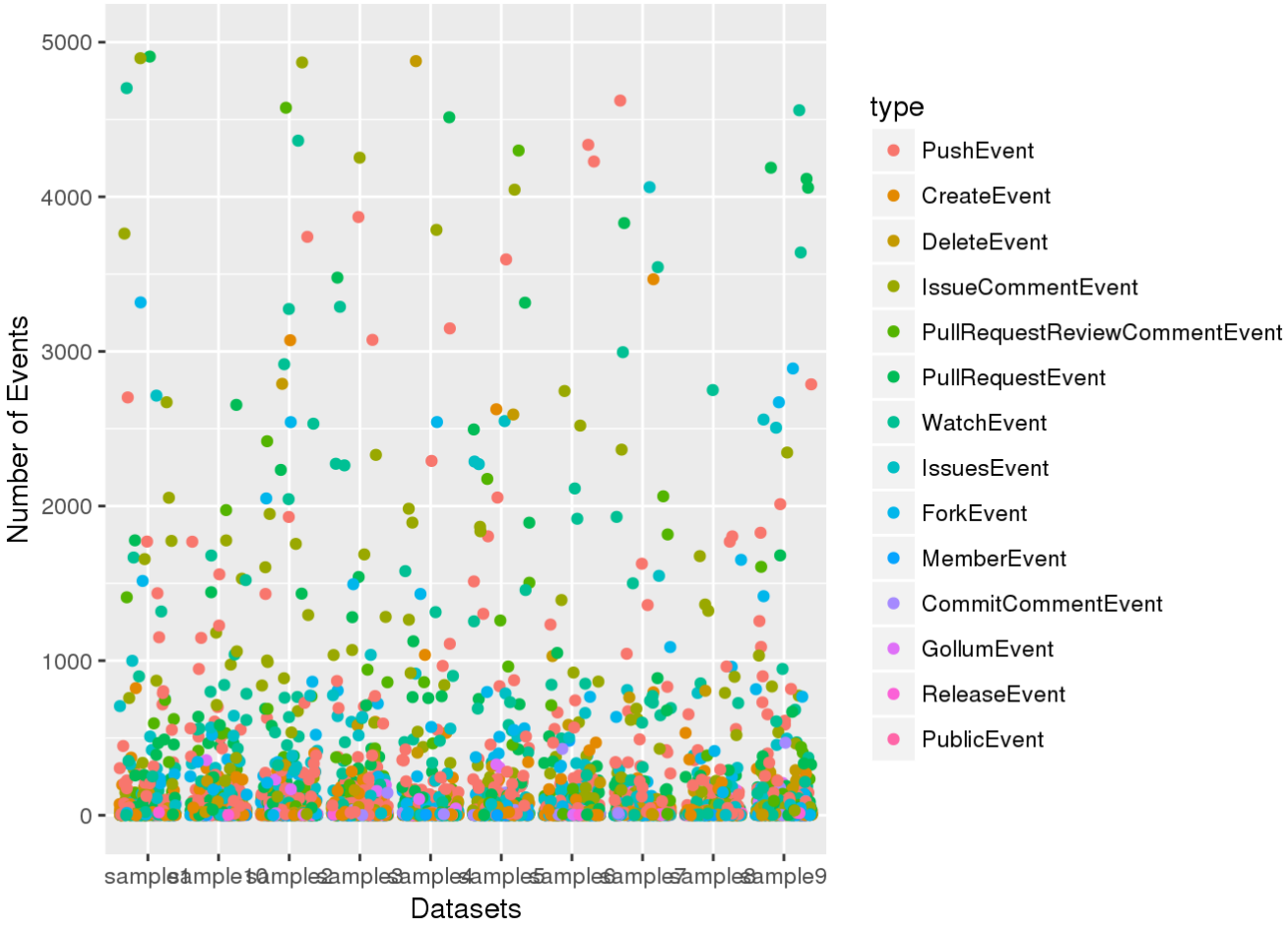

```
ggplot(data = events_repo_samples,
       aes(x = dataset,
           y = num_events,
           fill = type, color = type, group = type)) +
  geom_jitter(stat="identity") +
  xlab("Datasets") +
  ylab("Number of Events") +
  ylim(0, 25000)
```

```
## Warning: Removed 82 rows containing missing values (geom_point).
```



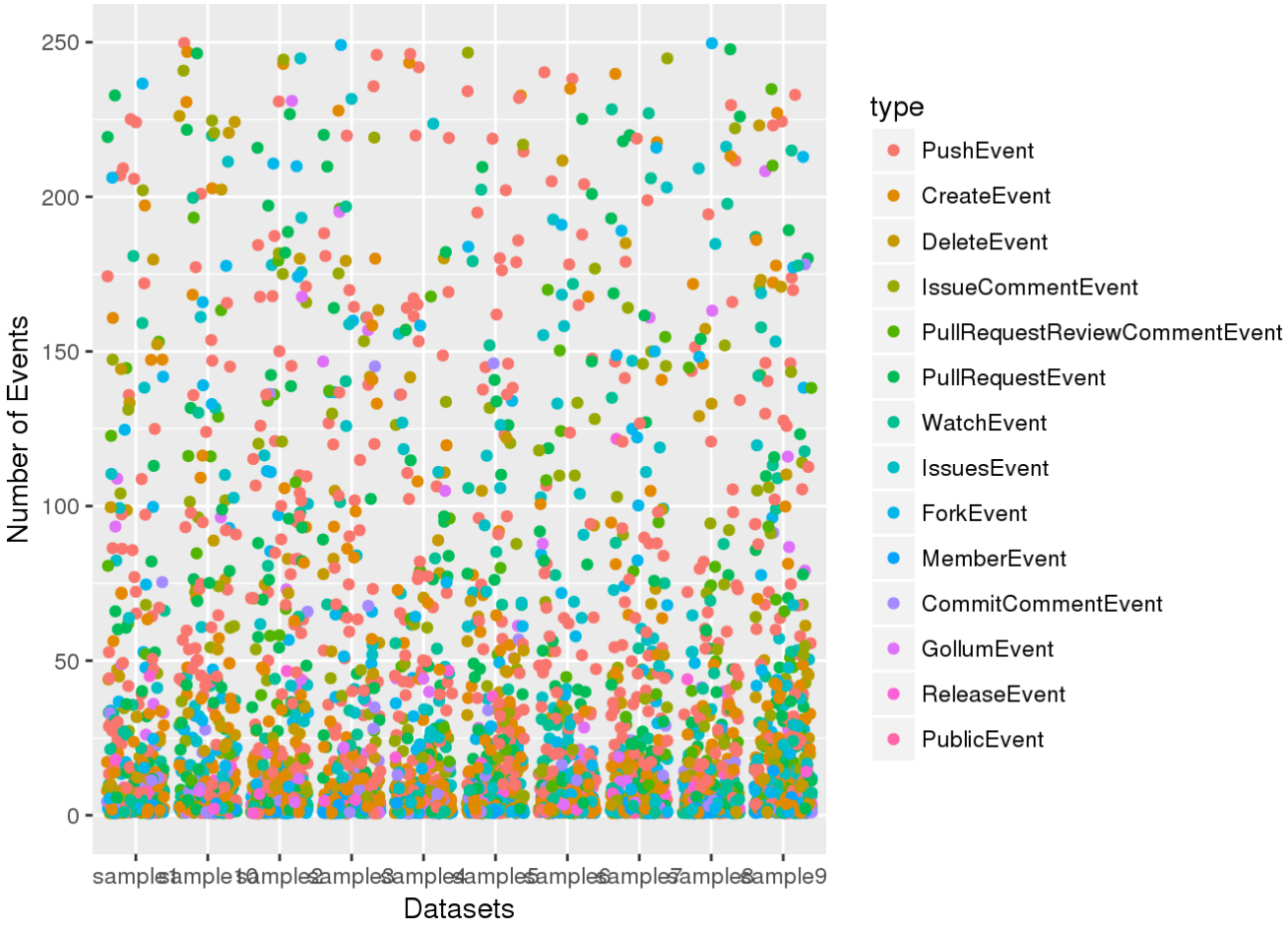

```
ggplot(data = events_repo_samples,
       aes(x = dataset,
           y = num_events,
           fill = type, color = type, group = type)) +
  geom_jitter(stat="identity") +
  xlab("Datasets") +
  ylab("Number of Events") +
  ylim(0, 5000)
```

```
## Warning: Removed 159 rows containing missing values (geom_point).
```



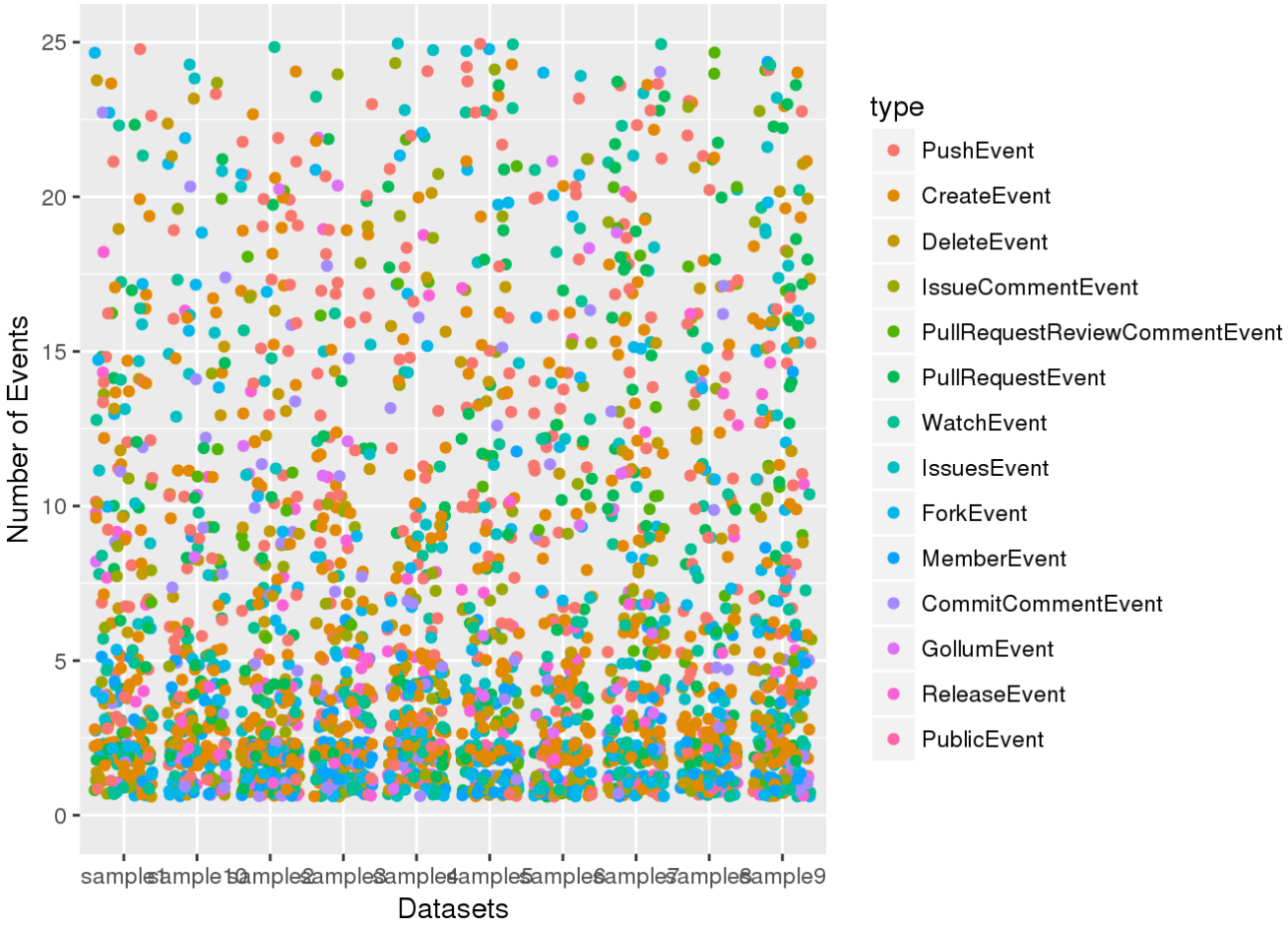

```
ggplot(data = events_repo_samples,
       aes(x = dataset,
           y = num_events,
           fill = type, color = type, group = type)) +
  geom_jitter(stat="identity") +
  xlab("Datasets") +
  ylab("Number of Events") +
  ylim(0, 250)
```

```
## Warning: Removed 685 rows containing missing values (geom_point).
```




```
ggplot(data = events_repo_samples,
       aes(x = dataset,
           y = num_events,
           fill = type, color = type, group = type)) +
  geom_jitter(stat="identity") +
  xlab("Datasets") +
  ylab("Number of Events") +
  ylim(0, 25)
```

```
## Warning: Removed 1889 rows containing missing values (geom_point).
```

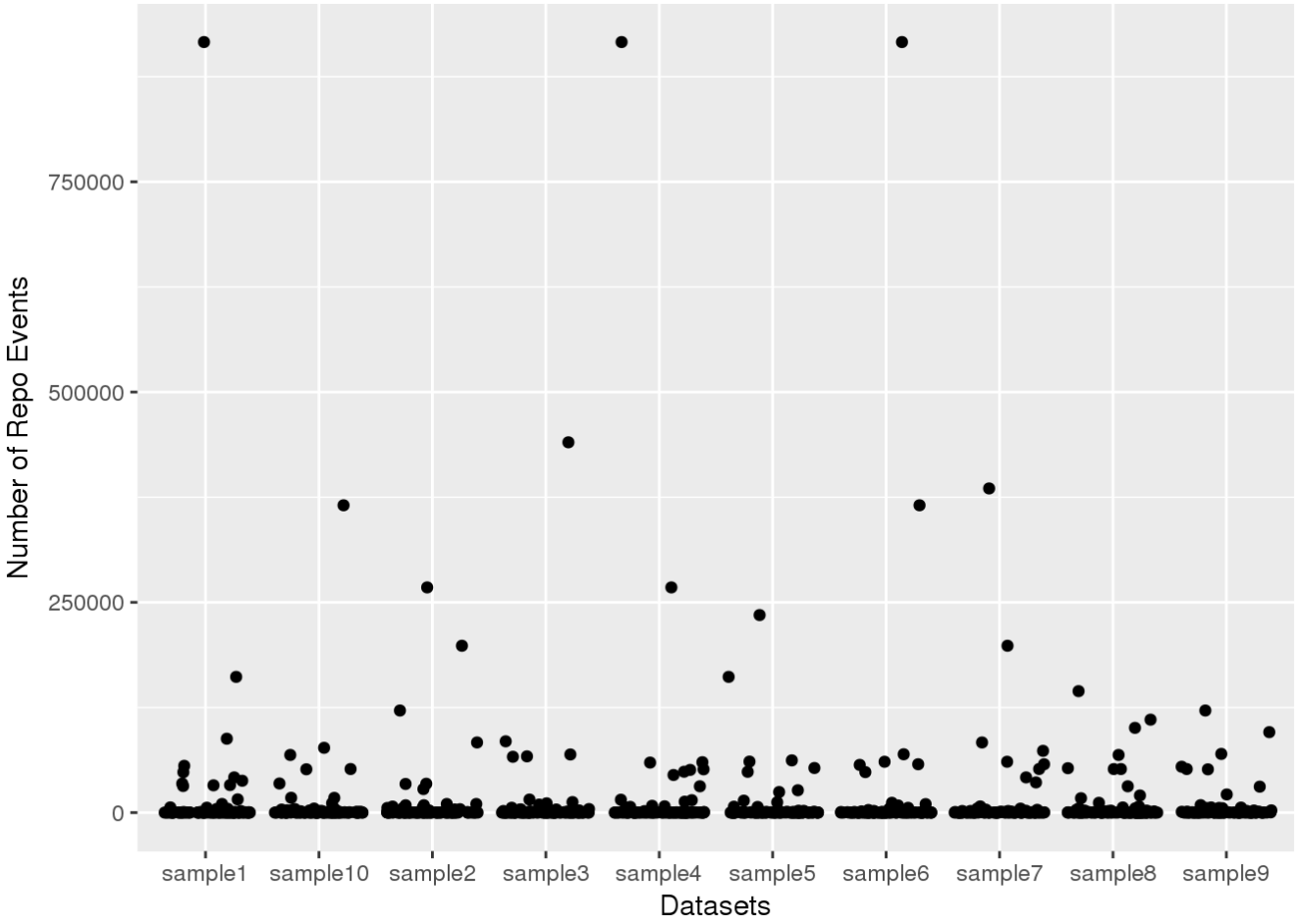


Event Frequency Distribution

The number of events per repository varies widely. The majority of repositories had less than 250 events.

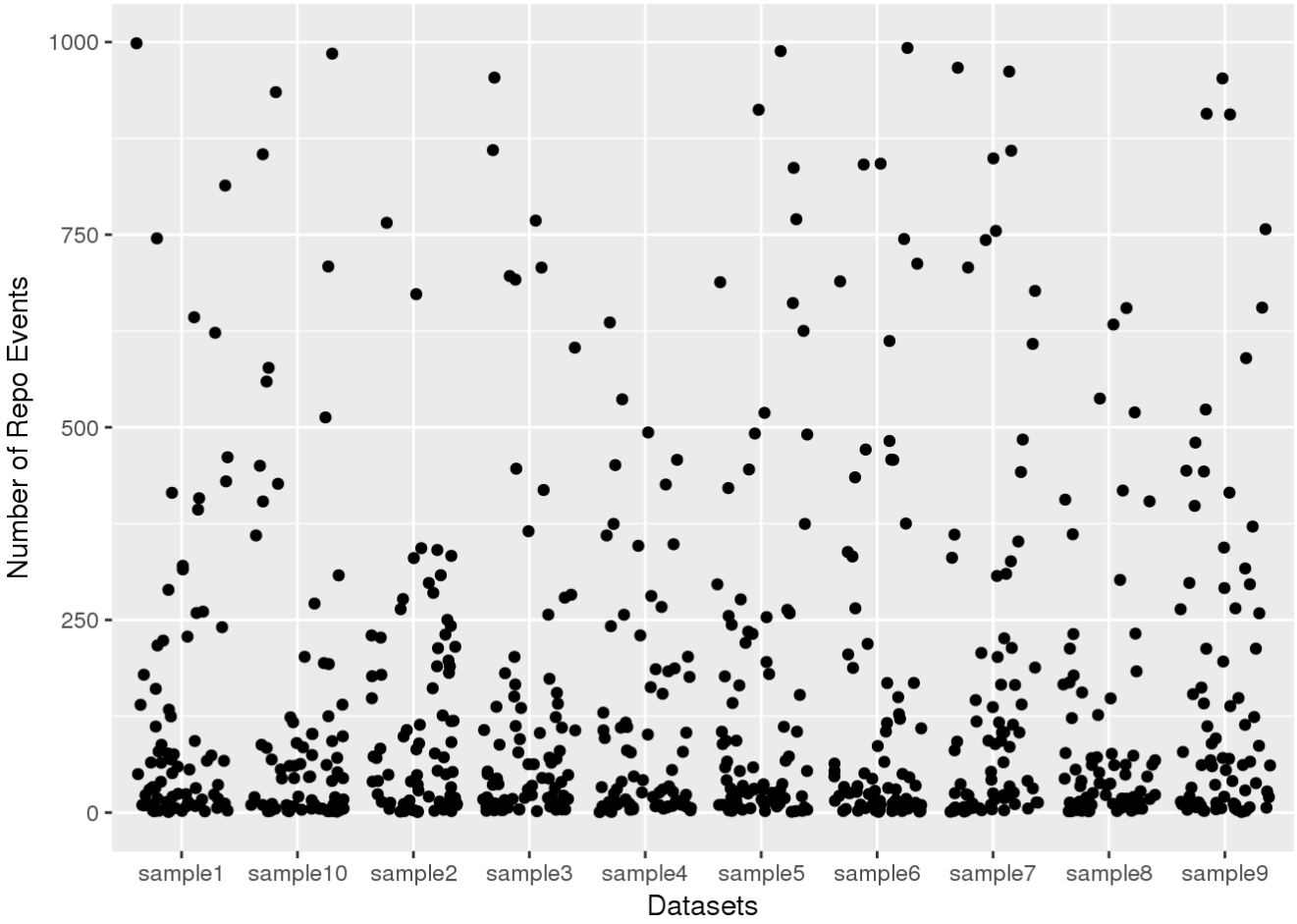
Further analysis would be needed to further assess the variability but for this study the visualizations below should provide sufficient estimation.

```
ggplot(data = events_repo_samples_by_repo,
       aes(x = dataset)) +
  geom_jitter(stat="identity", aes(y = num_repo_events)) +
  xlab("Datasets") +
  ylab("Number of Repo Events")
```




```
ggplot(data = events_repo_samples_by_repo,
       aes(x = dataset,
           y = num_repo_events)) +
  geom_jitter(stat="identity") +
  xlab("Datasets") +
  ylab("Number of Repo Events") +
  ylim(0,1000)
```

```
## Warning: Removed 216 rows containing missing values (geom_point).
```

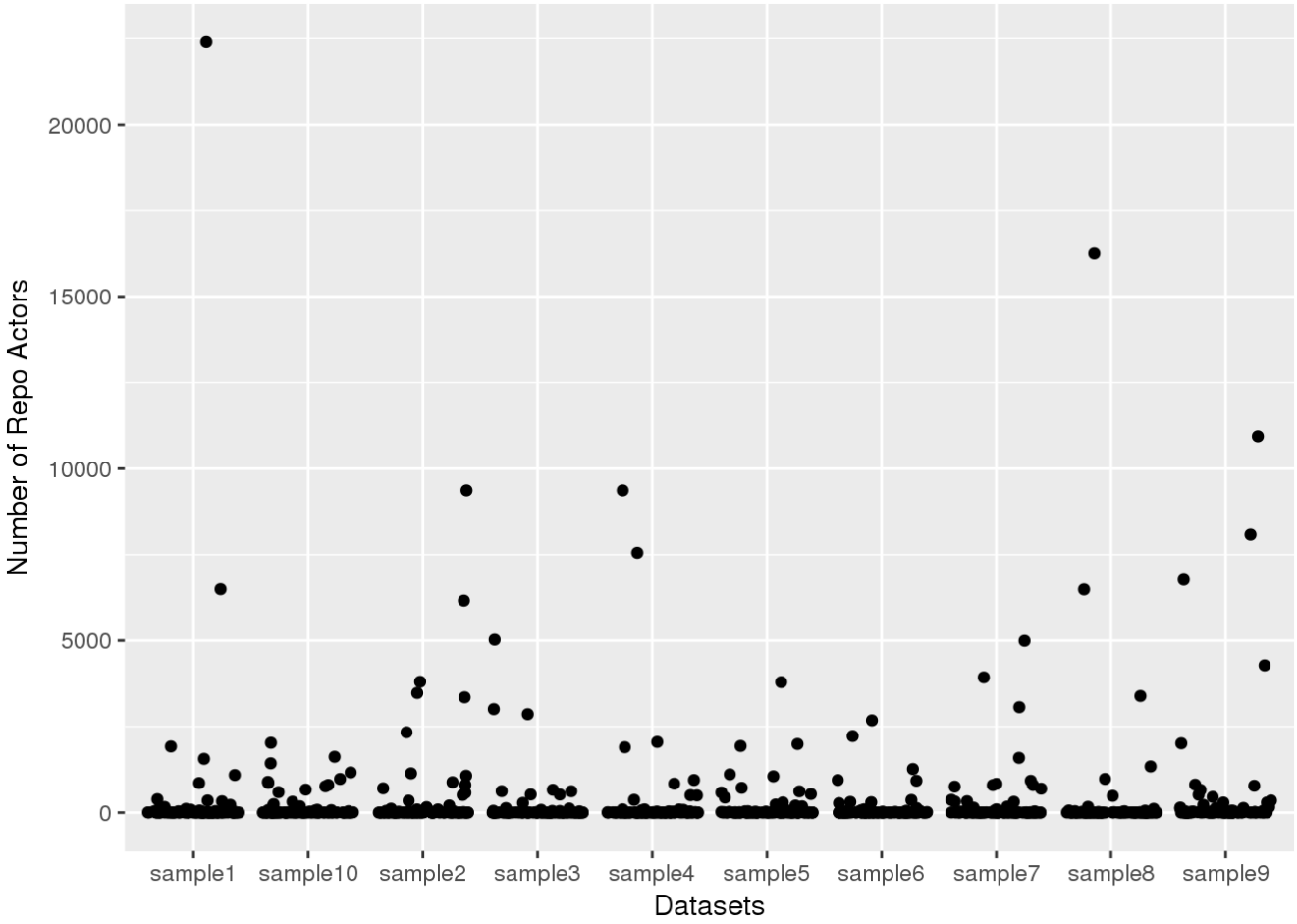


Actors Distribution

The number of actors per repository appears to be less variable than the number of events per repository. The majority of repositories sampled had events generated by less than 5 unique actors.

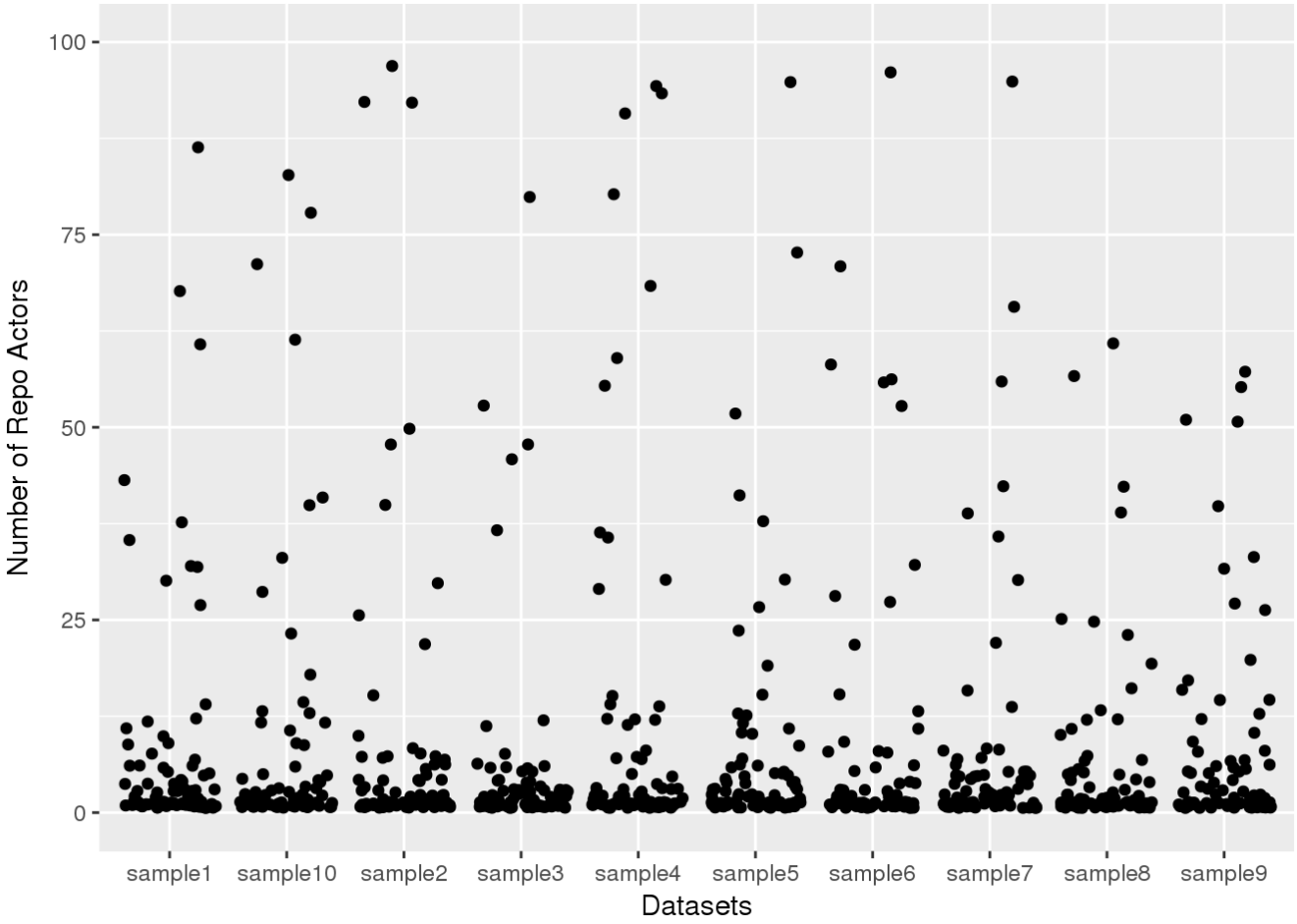
Further analysis would be needed to determine the exact nature of the variability, but for purposes of this study the graphic below is sufficient to get an idea.

```
ggplot(data = events_repo_samples_by_repo,
       aes(x = dataset,
           y = num_repo_actors)) +
  geom_jitter(stat="identity") +
  xlab("Datasets") +
  ylab("Number of Repo Actors")
```



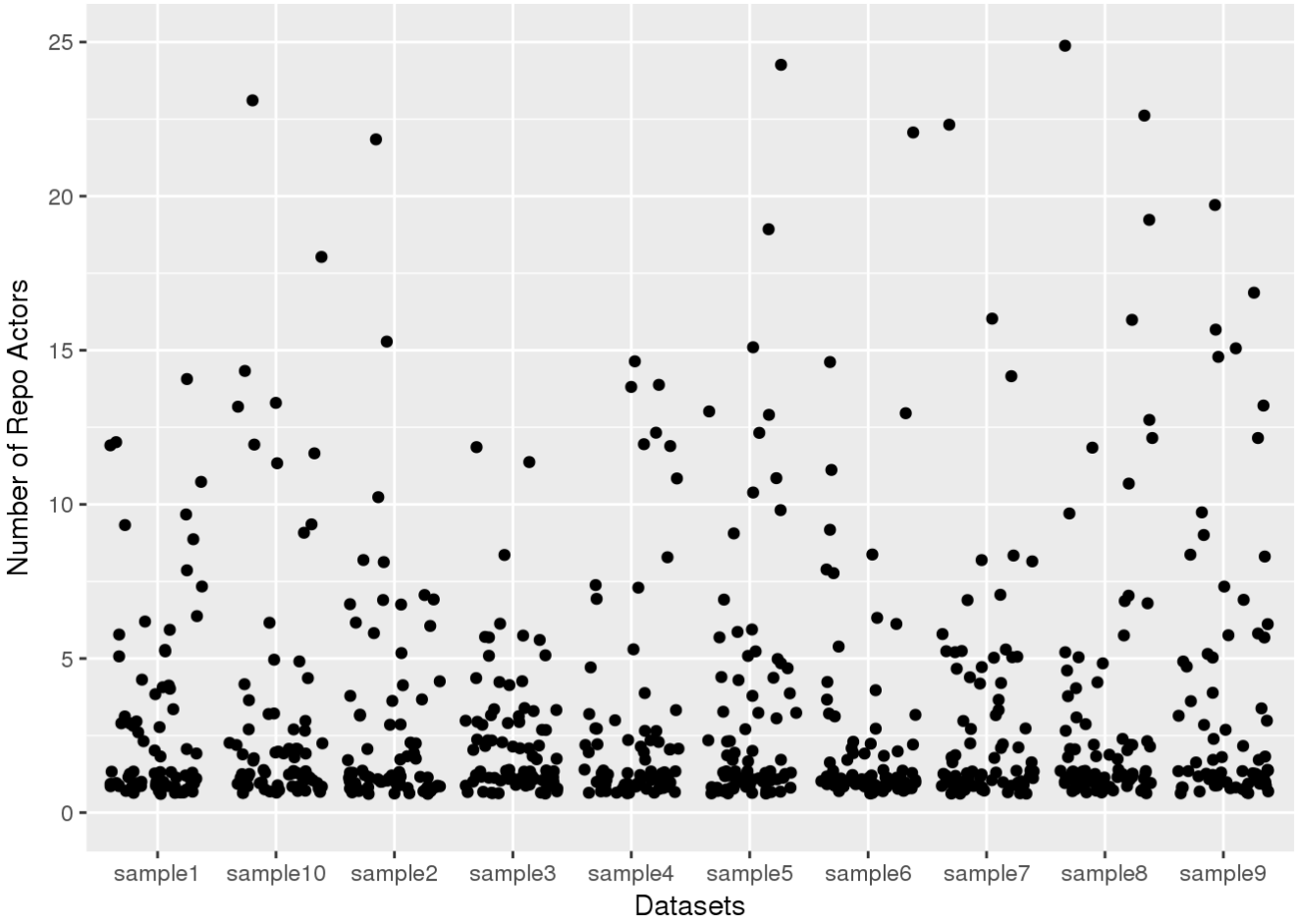

```
ggplot(data = events_repo_samples_by_repo,
       aes(x = dataset,
           y = num_repo_actors)) +
  geom_jitter(stat="identity") +
  xlab("Datasets") +
  ylab("Number of Repo Actors") +
  ylim(0, 100)
```

```
## Warning: Removed 133 rows containing missing values (geom_point).
```




```
ggplot(data = events_repo_samples_by_repo,
       aes(x = dataset,
           y = num_repo_actors)) +
  geom_jitter(stat="identity") +
  xlab("Datasets") +
  ylab("Number of Repo Actors") +
  ylim(0, 25)
```

```
## Warning: Removed 212 rows containing missing values (geom_point).
```



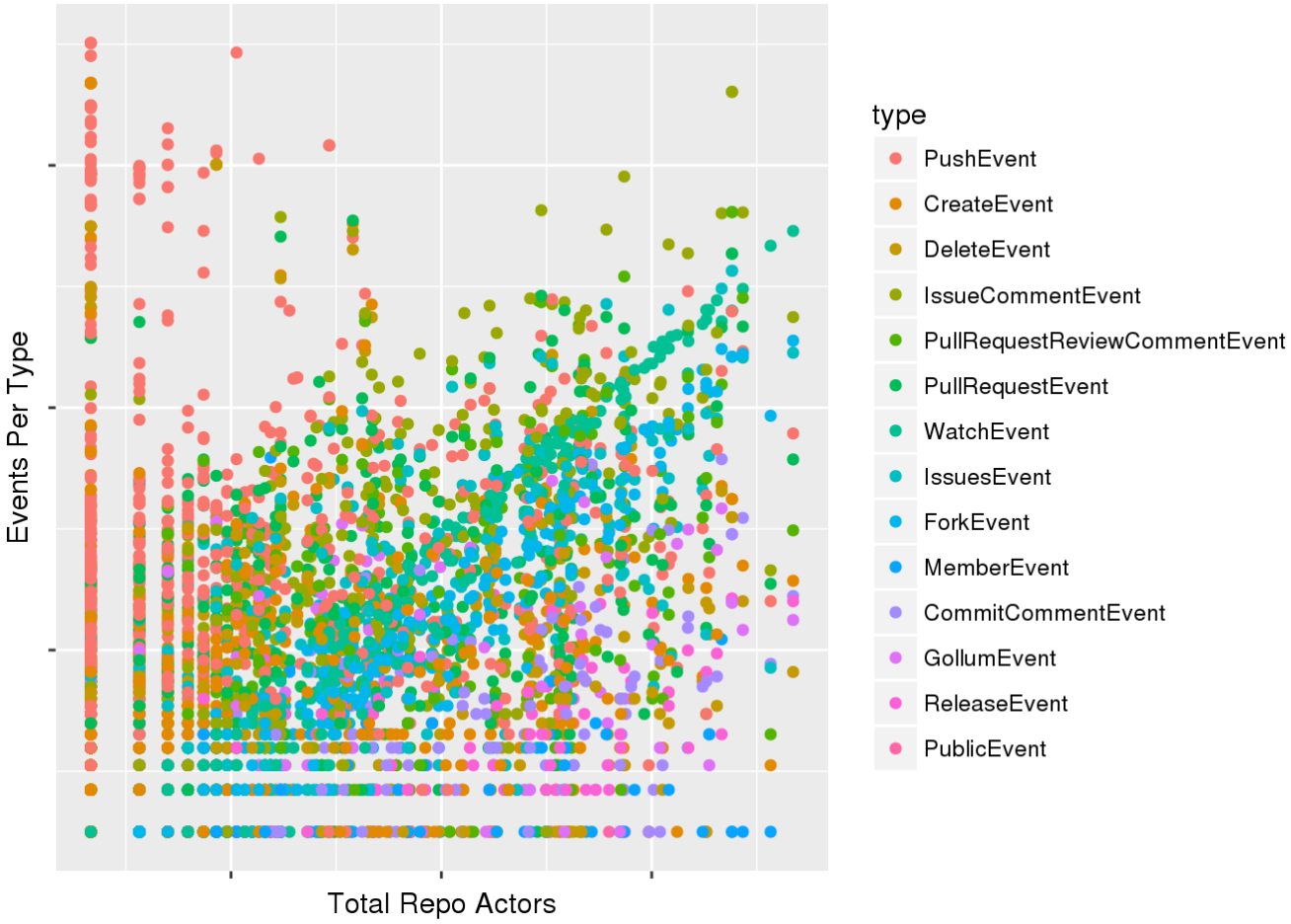
Event Types and Actors

The following plots examine whether there is a correlation between the number of events in each type belonging to a respository in the sample in the sample and the total number of unique repository actors. If a repository has a high frequency of a particular event type, does it follow that it will have a high number of actors? Do low frequencies of certain event types also indicate a low number of actors?

This scatterplot is pretty hard to read, but it does tell us one thing. Some event types seem to have a distinct pattern when plotted with total actors per repo while others do not. The next few scatterplots will focus on single event types to identify which ones might have a correlation. A percentage value has been applied when examining individual event types to indicate the average percent of events per actor. A higher percentage indicates the events are spread across a smaller number of actors. This is to provide insight into the overall shape of the data.

From looking at the more focused scatterplots, it looks like Fork, Issue Comment, and Watch events have the highest correlation to number of Repository Actors.

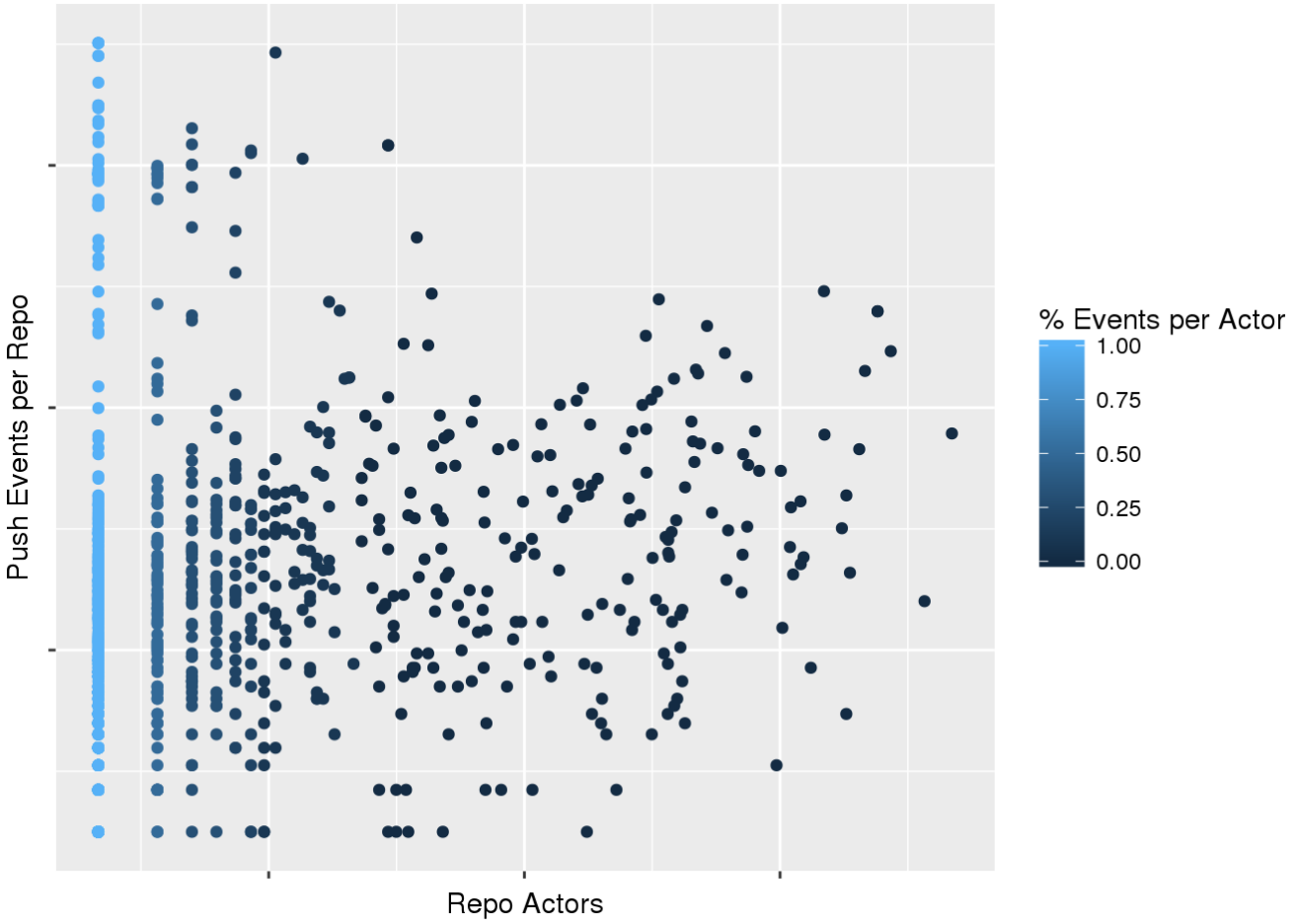
```
ggplot(data = events_repo_samples, aes(y=num_events, x=total_actors, fill=type, c
olour=type, group=type)) +
  geom_point(stat="identity") +
  ylab("Events Per Type") +
  xlab("Total Repo Actors") +
  scale_x_continuous(trans = "log", labels = NULL) +
  scale_y_continuous(trans = "log", labels = NULL)
```



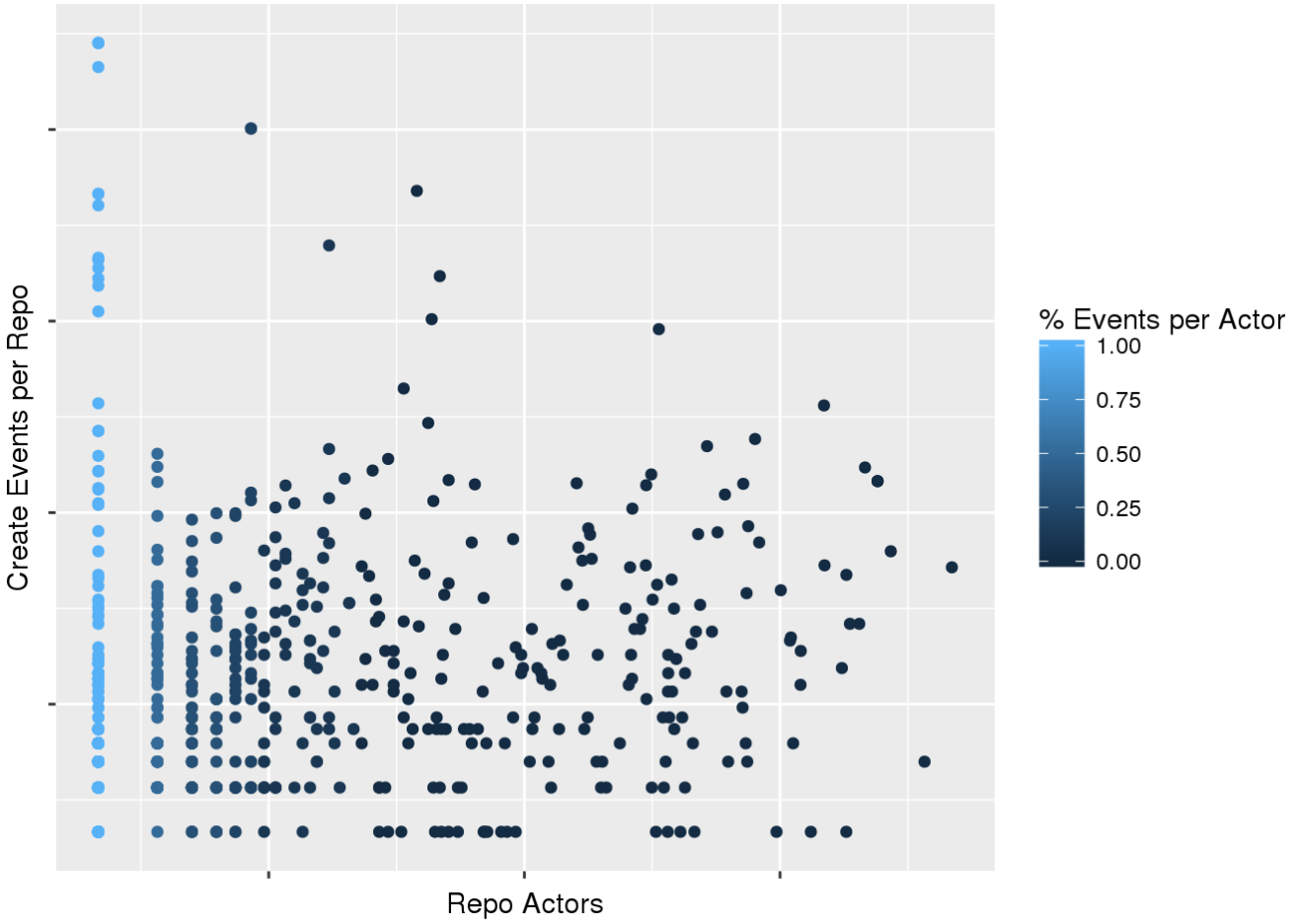
Most Frequent Event Types

Push, Create, Delete, IssueComment, PullRequestReviewComment, PullRequest, and Watch

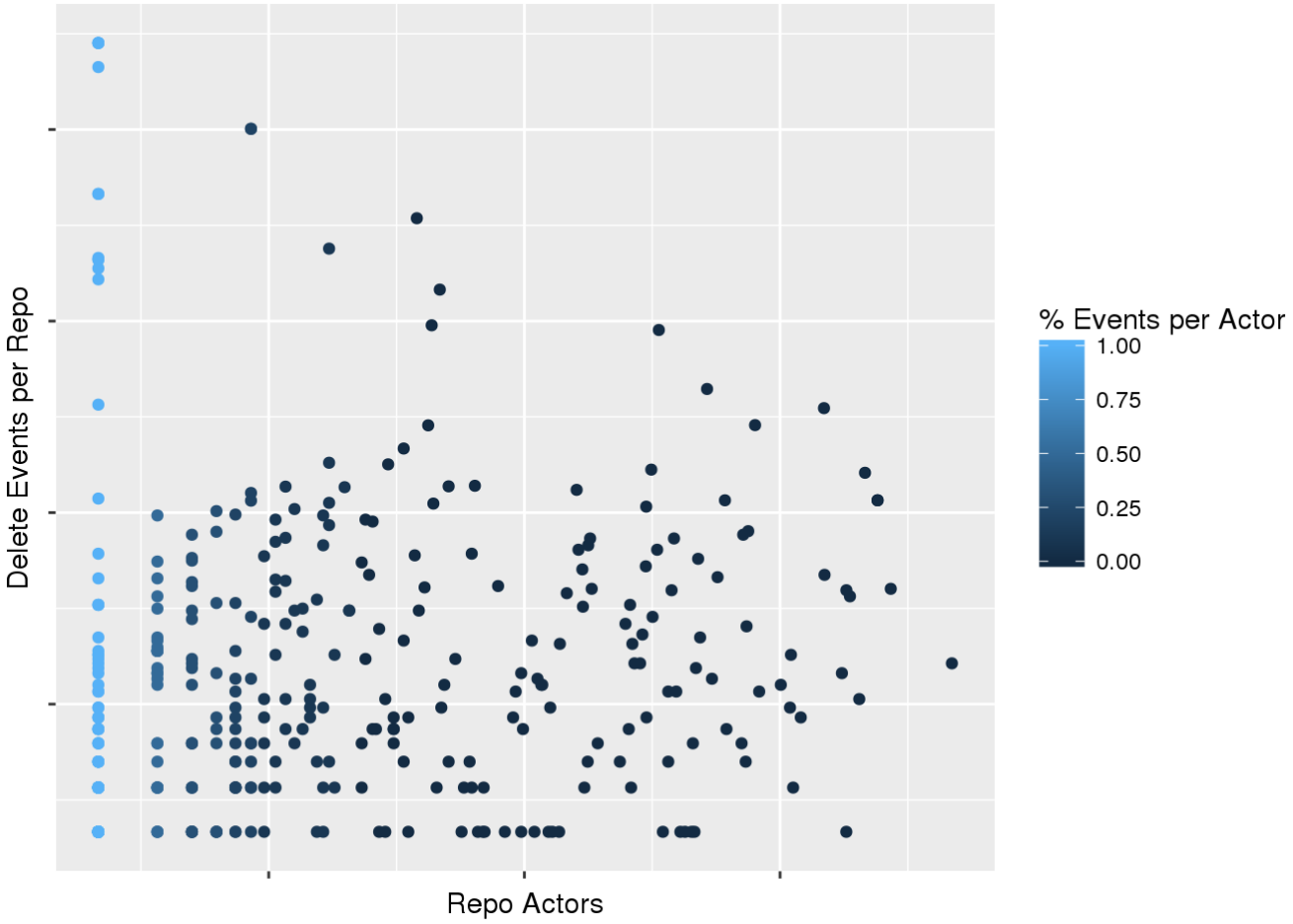
```
events_repo_samples_push <- events_repo_samples %>%
  filter(type == "PushEvent")

ggplot(data = events_repo_samples_push,
       aes(x=total_actors, y=num_events,
           fill=participation_rate, colour=participation_rate, group=participatio
n_rate)) +
  labs(colour = "% Events per Actor", fill = "% Events per Actor") +
  geom_point(stat="identity") +
  ylab("Push Events per Repo") +
  xlab("Repo Actors") +
  scale_x_continuous(trans = "log", labels=NULL) +
  scale_y_continuous(trans = "log", labels=NULL)
```



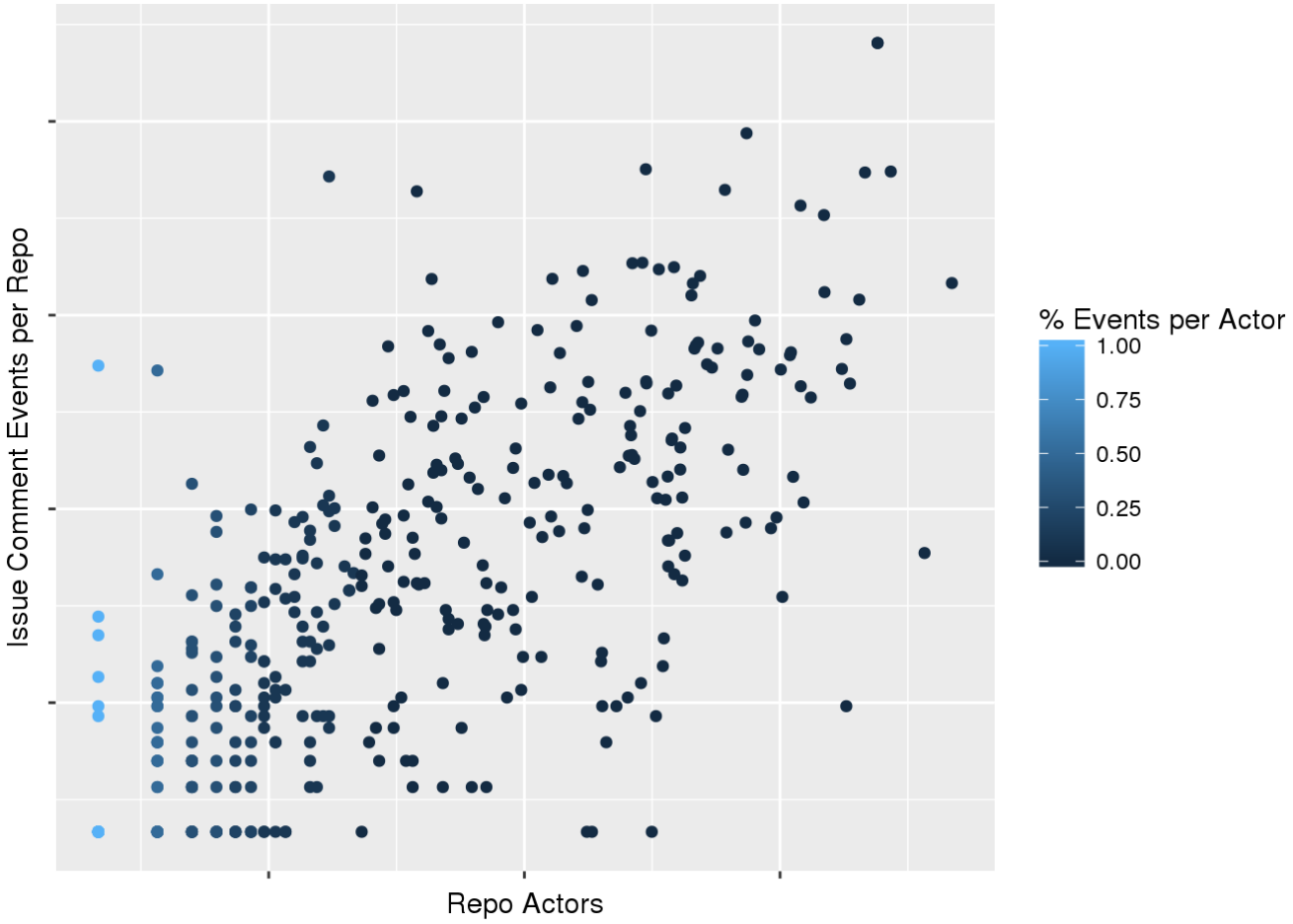

```
events_repo_samples_create <- events_repo_samples %>%
  filter(type == "CreateEvent")

ggplot(data = events_repo_samples_create, aes(x=total_actors, y=num_events,
           fill=participation_rate, colour=participation_rate, group=participatio
n_rate)) +
  labs(colour = "% Events per Actor", fill = "% Events per Actor") +
  geom_point(stat="identity") +
  ylab("Create Events per Repo") +
  xlab("Repo Actors") +
  scale_x_continuous(trans = "log", labels=NULL) +
  scale_y_continuous(trans = "log", labels=NULL)
```



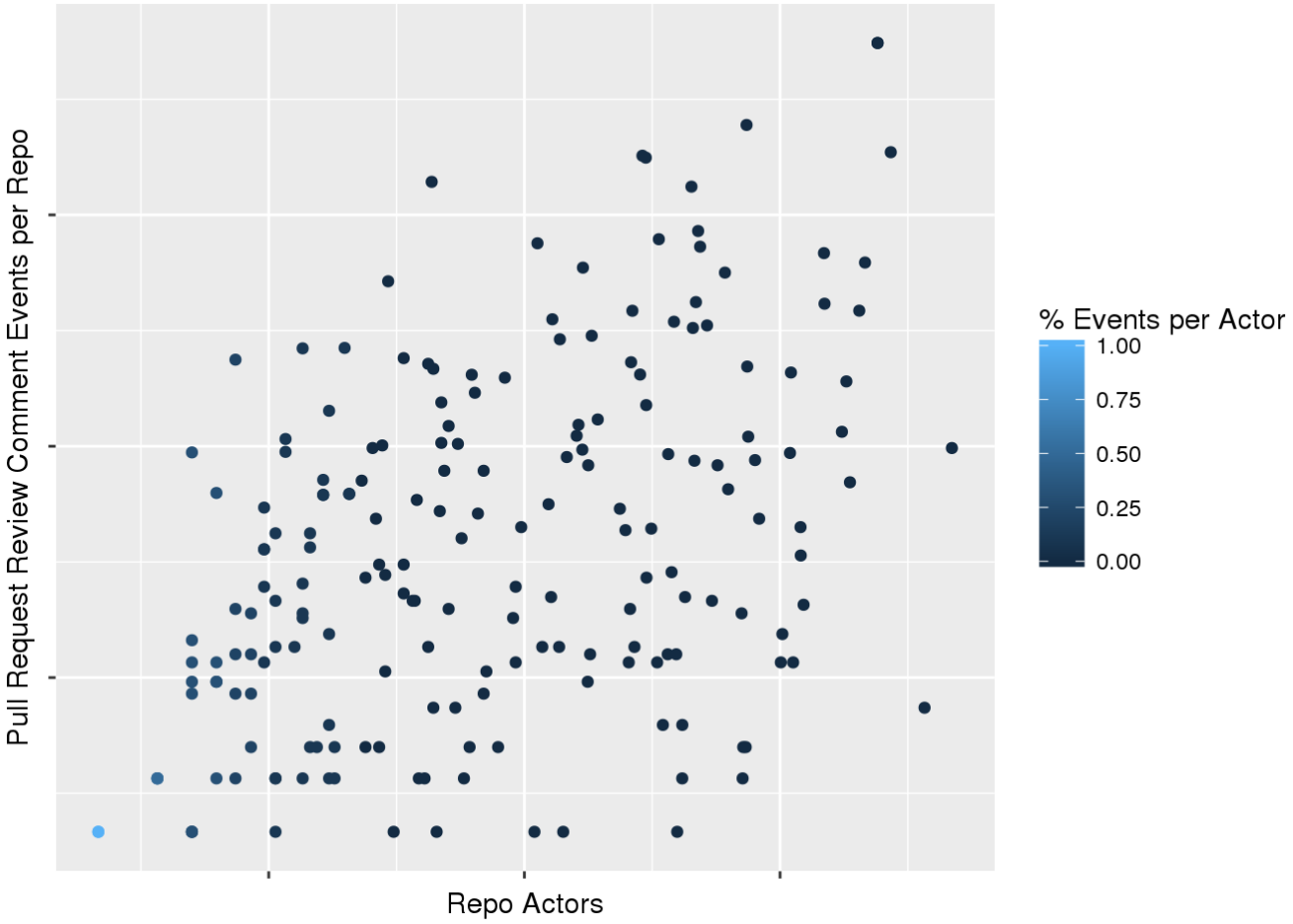

```
events_repo_samples_del <- events_repo_samples %>%
  filter(type == "DeleteEvent")

ggplot(data = events_repo_samples_del, aes(x=total_actors, y=num_events,
           fill=participation_rate, colour=participation_rate, group=participatio
n_rate)) +
  labs(colour = "% Events per Actor", fill = "% Events per Actor") +
  geom_point(stat="identity") +
  ylab("Delete Events per Repo") +
  xlab("Repo Actors") +
  scale_x_continuous(trans = "log", labels=NULL) +
  scale_y_continuous(trans = "log", labels=NULL)
```



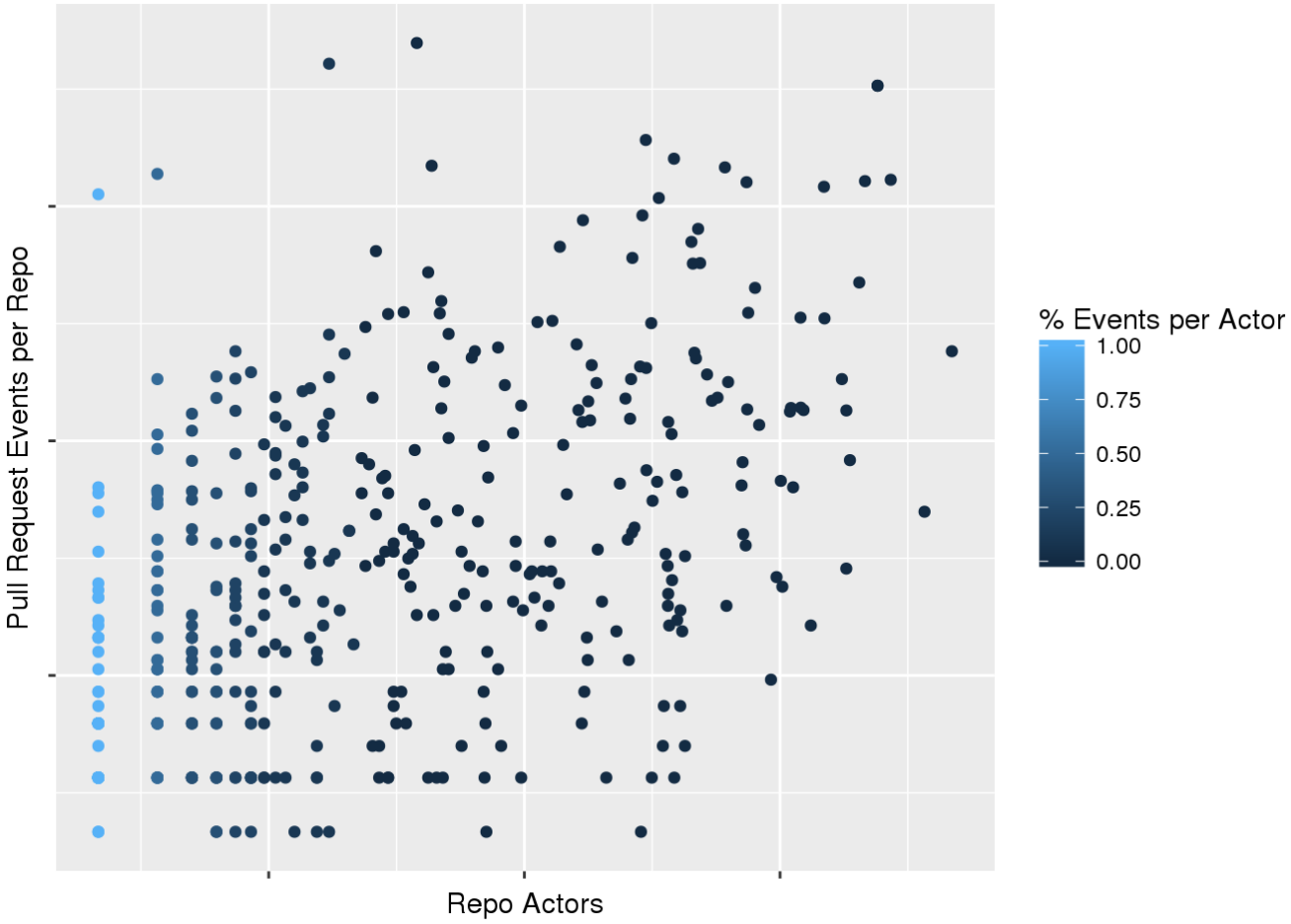

```
events_repo_samples_issuecomment <- events_repo_samples %>%
  filter(type == "IssueCommentEvent")

ggplot(data = events_repo_samples_issuecomment, aes(y=num_events, x=total_actors,
           fill=participation_rate, colour=participation_rate, group=participatio
n_rate)) +
  labs(colour = "% Events per Actor", fill = "% Events per Actor") +
  geom_point(stat="identity") +
  ylab("Issue Comment Events per Repo") +
  xlab("Repo Actors") +
  scale_x_continuous(trans = "log", labels=NULL) +
  scale_y_continuous(trans = "log", labels=NULL)
```



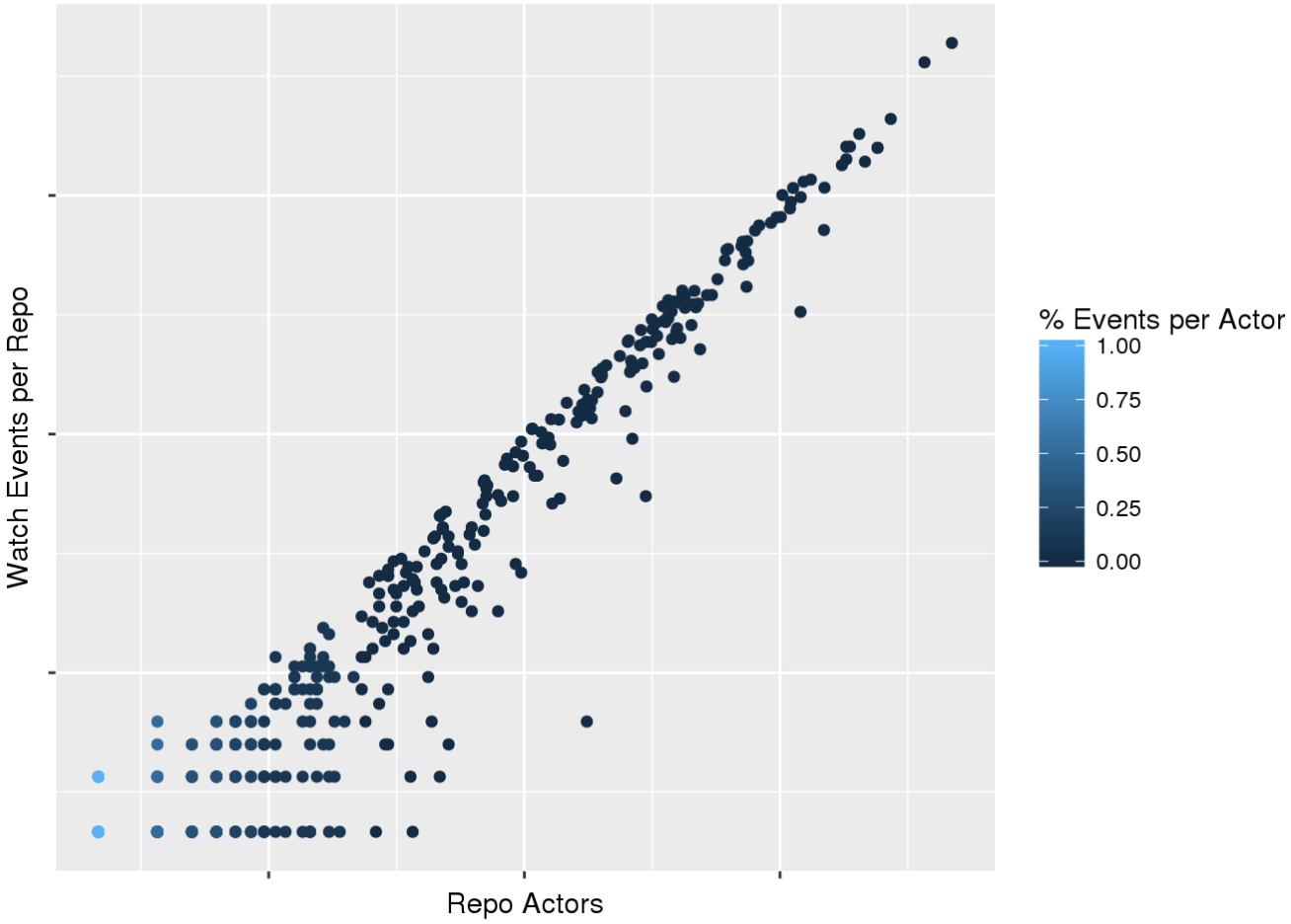

```
events_repo_samples_prc <- events_repo_samples %>%
  filter(type == "PullRequestReviewCommentEvent")

ggplot(data = events_repo_samples_prc, aes(y=num_events, x=total_actors,
           fill=participation_rate, colour=participation_rate, group=participatio
n_rate)) +
  labs(colour = "% Events per Actor", fill = "% Events per Actor") +
  geom_point(stat="identity") +
  ylab("Pull Request Review Comment Events per Repo") +
  xlab("Repo Actors") +
  scale_x_continuous(trans = "log", labels=NULL) +
  scale_y_continuous(trans = "log", labels=NULL)
```




```
events_repo_samples_pr <- events_repo_samples %>%
  filter(type == "PullRequestEvent")

ggplot(data = events_repo_samples_pr, aes(y=num_events, x=total_actors,
           fill=participation_rate, colour=participation_rate, group=participatio
n_rate)) +
  labs(colour = "% Events per Actor", fill = "% Events per Actor") +
  geom_point(stat="identity") +
  ylab("Pull Request Events per Repo") +
  xlab("Repo Actors") +
  scale_x_continuous(trans = "log", labels=NULL) +
  scale_y_continuous(trans = "log", labels=NULL)
```

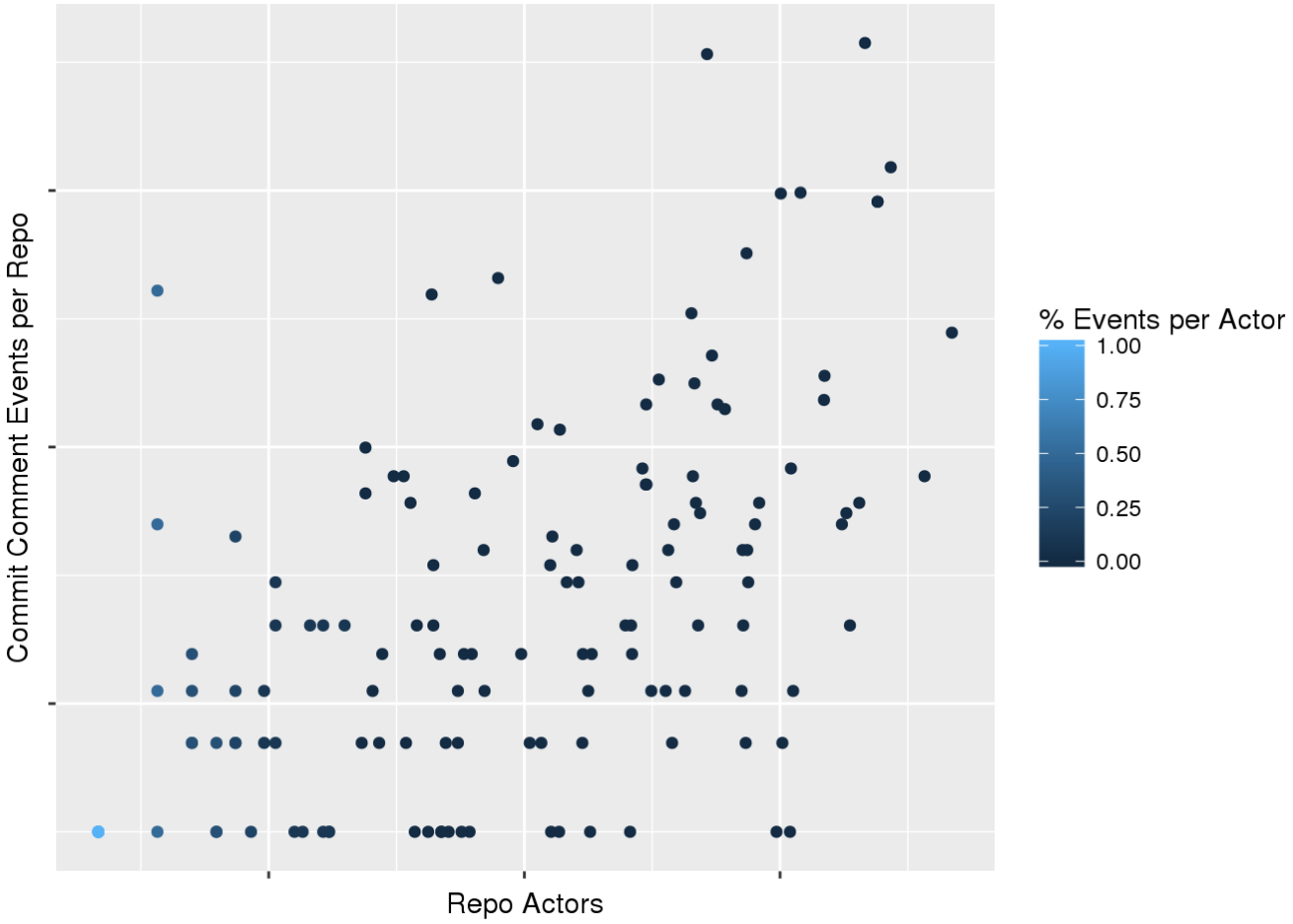



```
events_repo_samples_watch <- events_repo_samples %>%
  filter(type == "WatchEvent")

ggplot(data = events_repo_samples_watch, aes(y=num_events, x=total_actors,
           fill=participation_rate, colour=participation_rate, group=participatio
n_rate)) +
  labs(colour = "% Events per Actor", fill = "% Events per Actor") +
  geom_point(stat="identity") +
  ylab("Watch Events per Repo") +
  xlab("Repo Actors") +
  scale_x_continuous(trans = "log", labels=NULL) +
  scale_y_continuous(trans = "log", labels=NULL)
```

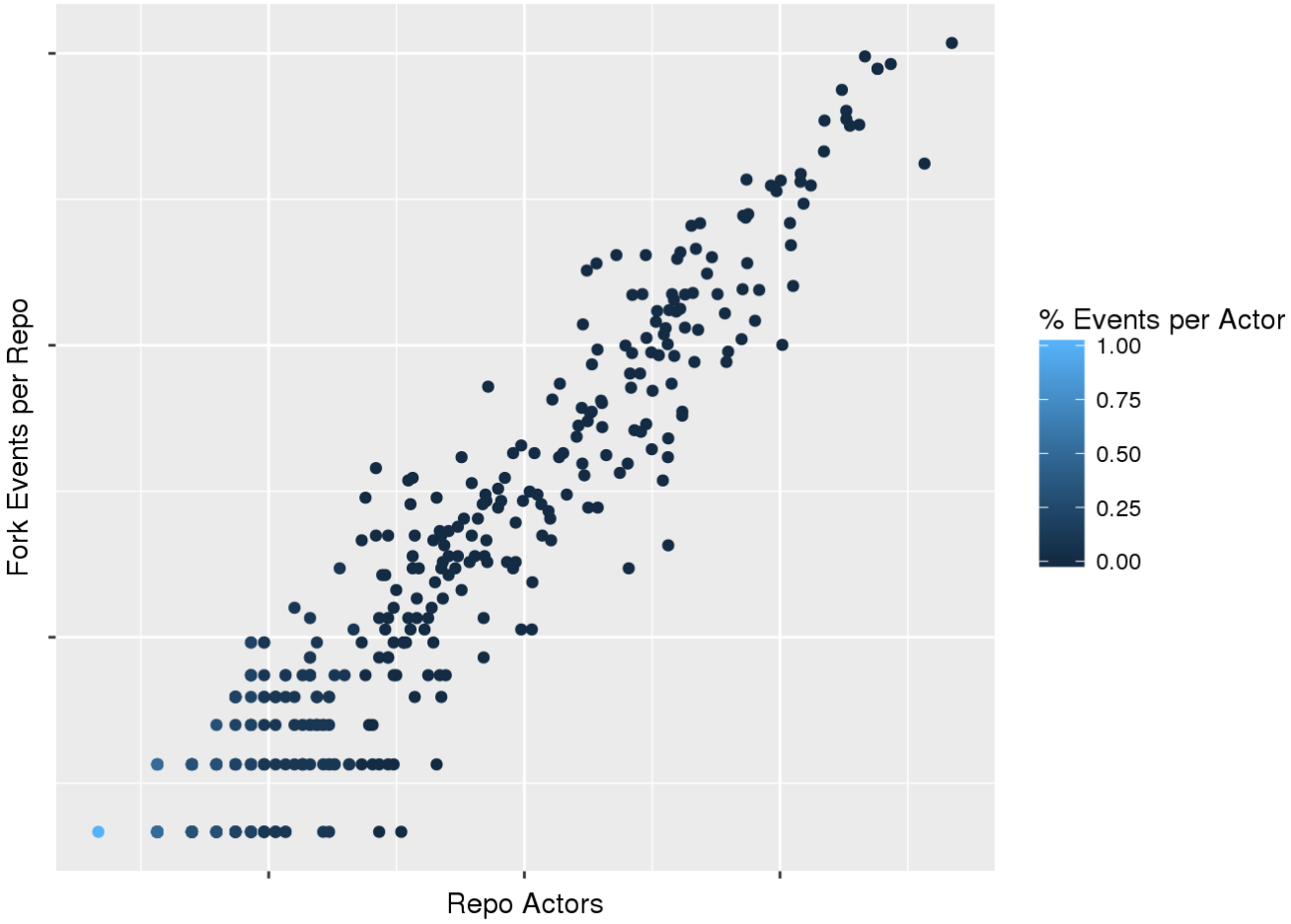


Less Frequent Event Types

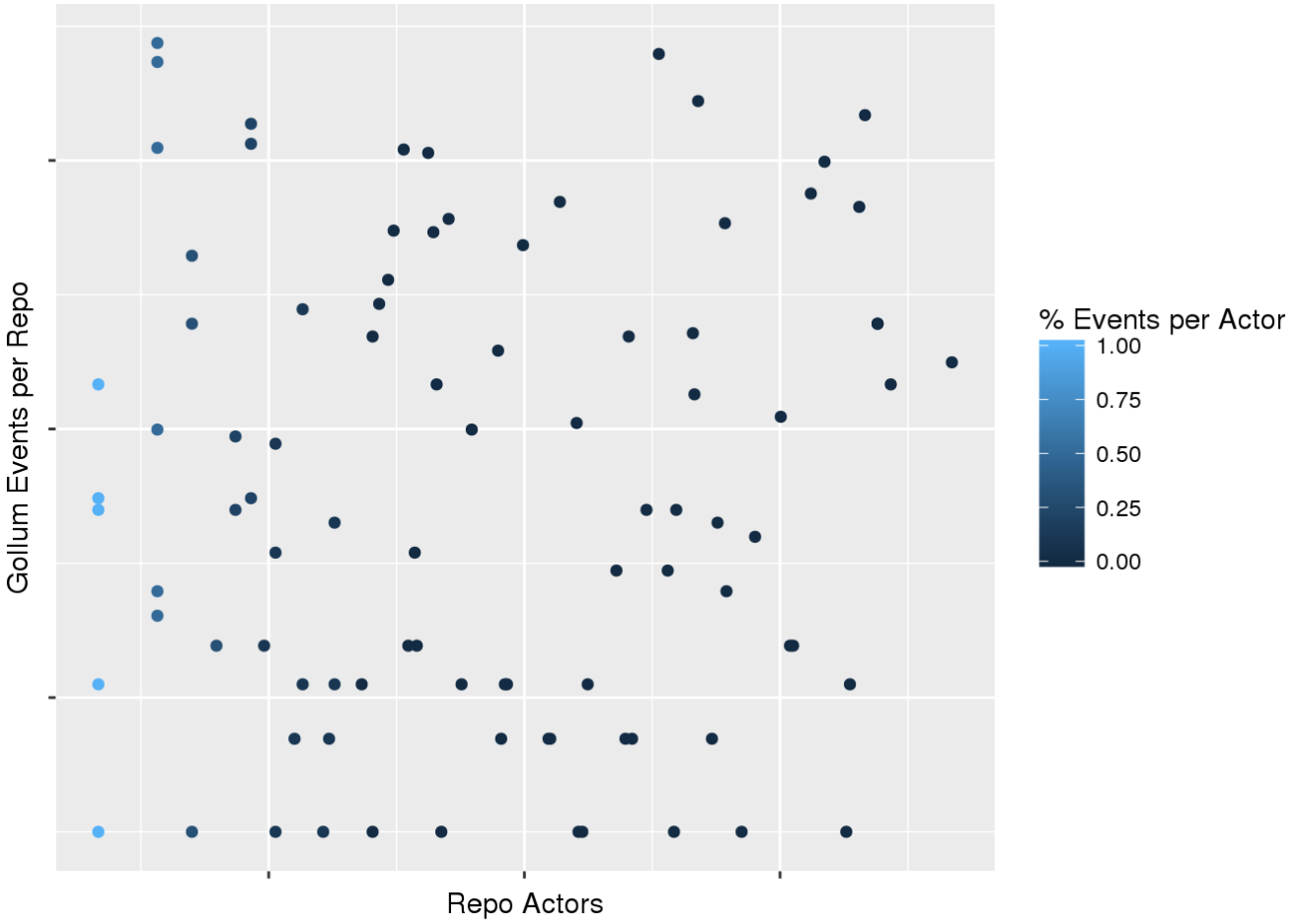
```
events_repo_samples_comcom <- events_repo_samples %>%
  filter(type == "CommitCommentEvent")

ggplot(data = events_repo_samples_comcom, aes(y=num_events, x=total_actors,
           fill=participation_rate, colour=participation_rate, group=participatio
n_rate)) +
  labs(colour = "% Events per Actor", fill = "% Events per Actor") +
  geom_point(stat="identity") +
  ylab("Commit Comment Events per Repo") +
  xlab("Repo Actors") +
  scale_x_continuous(trans = "log", labels=NULL) +
  scale_y_continuous(trans = "log", labels=NULL)
```



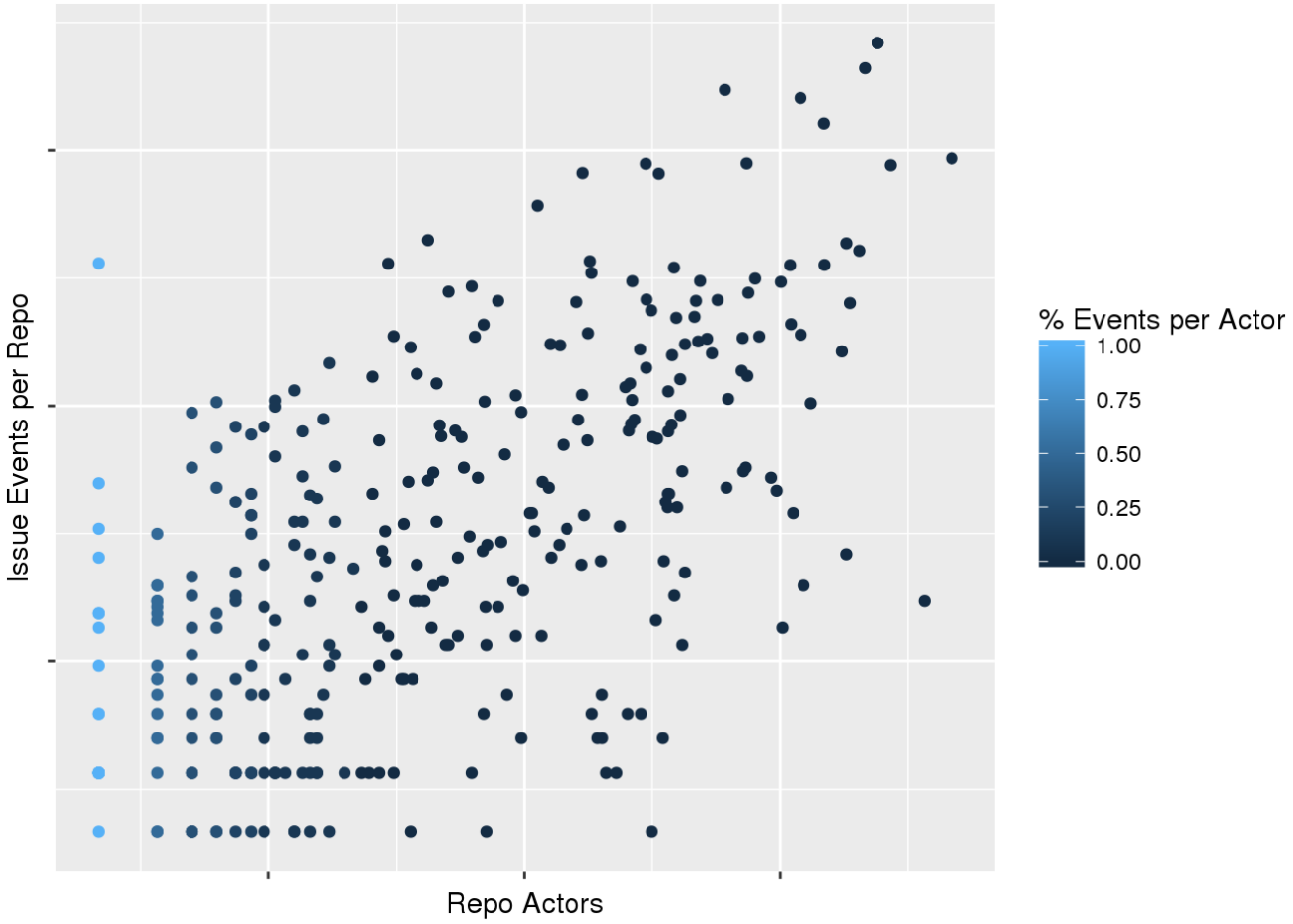

```
events_repo_samples_fork <- events_repo_samples %>%
  filter(type == "ForkEvent")

ggplot(data = events_repo_samples_fork, aes(y=num_events, x=total_actors,
           fill=participation_rate, colour=participation_rate, group=participatio
n_rate)) +
  labs(colour = "% Events per Actor", fill = "% Events per Actor") +
  geom_point(stat="identity") +
  ylab("Fork Events per Repo") +
  xlab("Repo Actors") +
  scale_x_continuous(trans = "log", labels=NULL) +
  scale_y_continuous(trans = "log", labels=NULL)
```



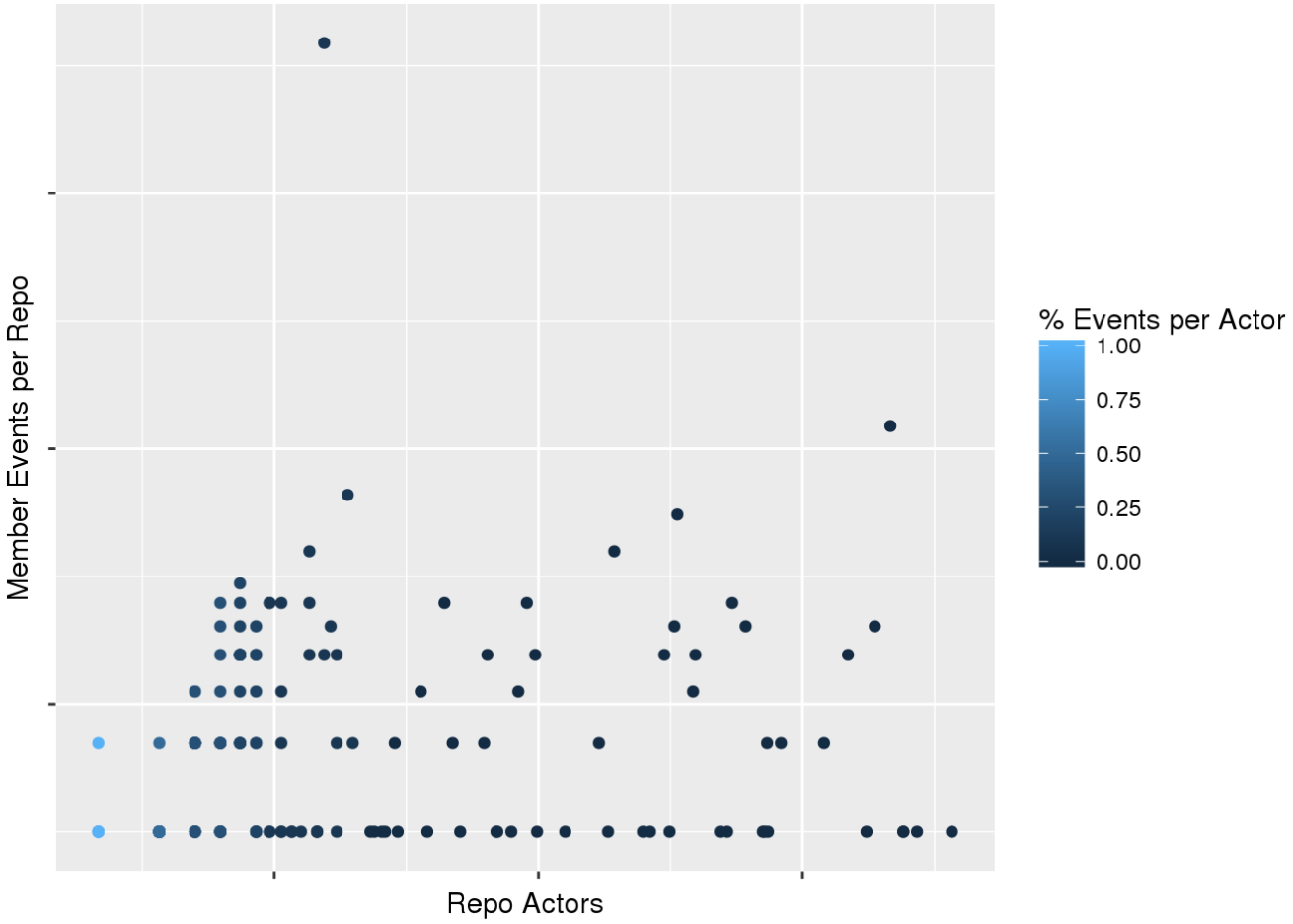

```
events_repo_samples_gollum <- events_repo_samples %>%
  filter(type == "GollumEvent")

ggplot(data = events_repo_samples_gollum, aes(y=num_events, x=total_actors,
           fill=participation_rate, colour=participation_rate, group=participatio
n_rate)) +
  labs(colour = "% Events per Actor", fill = "% Events per Actor") +
  geom_point(stat="identity") +
  ylab("Gollum Events per Repo") +
  xlab("Repo Actors") +
  scale_x_continuous(trans = "log", labels=NULL) +
  scale_y_continuous(trans = "log", labels=NULL)
```



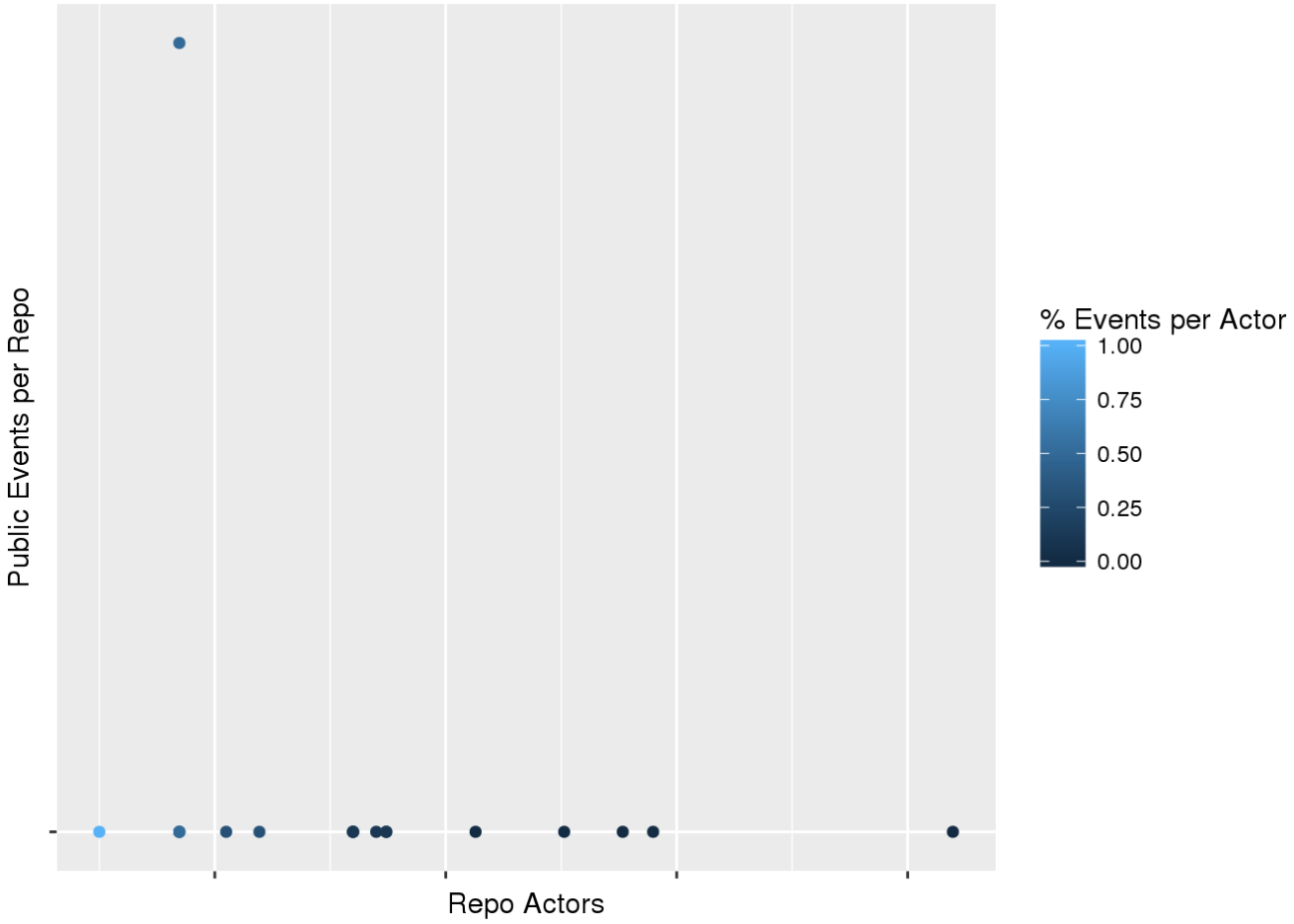

```
events_repo_samples_issue <- events_repo_samples %>%
  filter(type == "IssuesEvent")

ggplot(data = events_repo_samples_issue, aes(y=num_events, x=total_actors,
           fill=participation_rate, colour=participation_rate, group=participatio
n_rate)) +
  labs(colour = "% Events per Actor", fill = "% Events per Actor") +
  geom_point(stat="identity") +
  ylab("Issue Events per Repo") +
  xlab("Repo Actors") +
  scale_x_continuous(trans = "log", labels=NULL) +
  scale_y_continuous(trans = "log", labels=NULL)
```



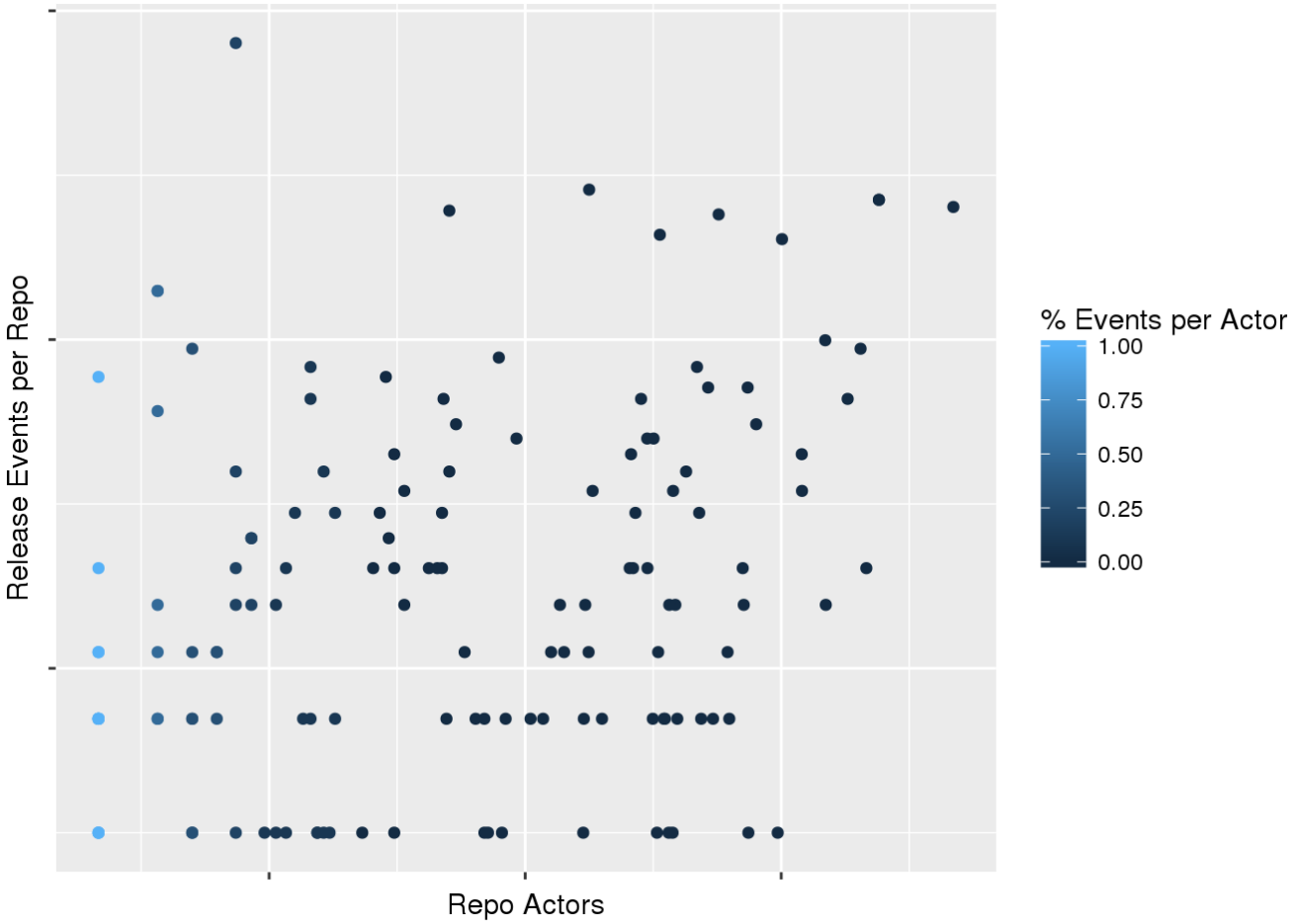

```
events_repo_samples_member <- events_repo_samples %>%
  filter(type == "MemberEvent")

ggplot(data = events_repo_samples_member, aes(y=num_events, x=total_actors,
           fill=participation_rate, colour=participation_rate, group=participatio
n_rate)) +
  labs(colour = "% Events per Actor", fill = "% Events per Actor") +
  geom_point(stat="identity") +
  ylab("Member Events per Repo") +
  xlab("Repo Actors") +
  scale_x_continuous(trans = "log", labels=NULL) +
  scale_y_continuous(trans = "log", labels=NULL)
```




```
events_repo_samples_public <- events_repo_samples %>%
  filter(type == "PublicEvent")

ggplot(data = events_repo_samples_public, aes(y=num_events, x=total_actors,
           fill=participation_rate, colour=participation_rate, group=participatio
n_rate)) +
  labs(colour = "% Events per Actor", fill = "% Events per Actor") +
  geom_point(stat="identity") +
  ylab("Public Events per Repo") +
  xlab("Repo Actors") +
  scale_x_continuous(trans = "log", labels=NULL) +
  scale_y_continuous(trans = "log", labels=NULL)
```




```
events_repo_samples_release <- events_repo_samples %>%
  filter(type == "ReleaseEvent")

ggplot(data = events_repo_samples_release, aes(y=num_events, x=total_actors,
           fill=participation_rate, colour=participation_rate, group=participatio
n_rate)) +
  labs(colour = "% Events per Actor", fill = "% Events per Actor") +
  geom_point(stat="identity") +
  ylab("Release Events per Repo") +
  xlab("Repo Actors") +
  scale_x_continuous(trans = "log", labels=NULL) +
  scale_y_continuous(trans = "log", labels=NULL)
```

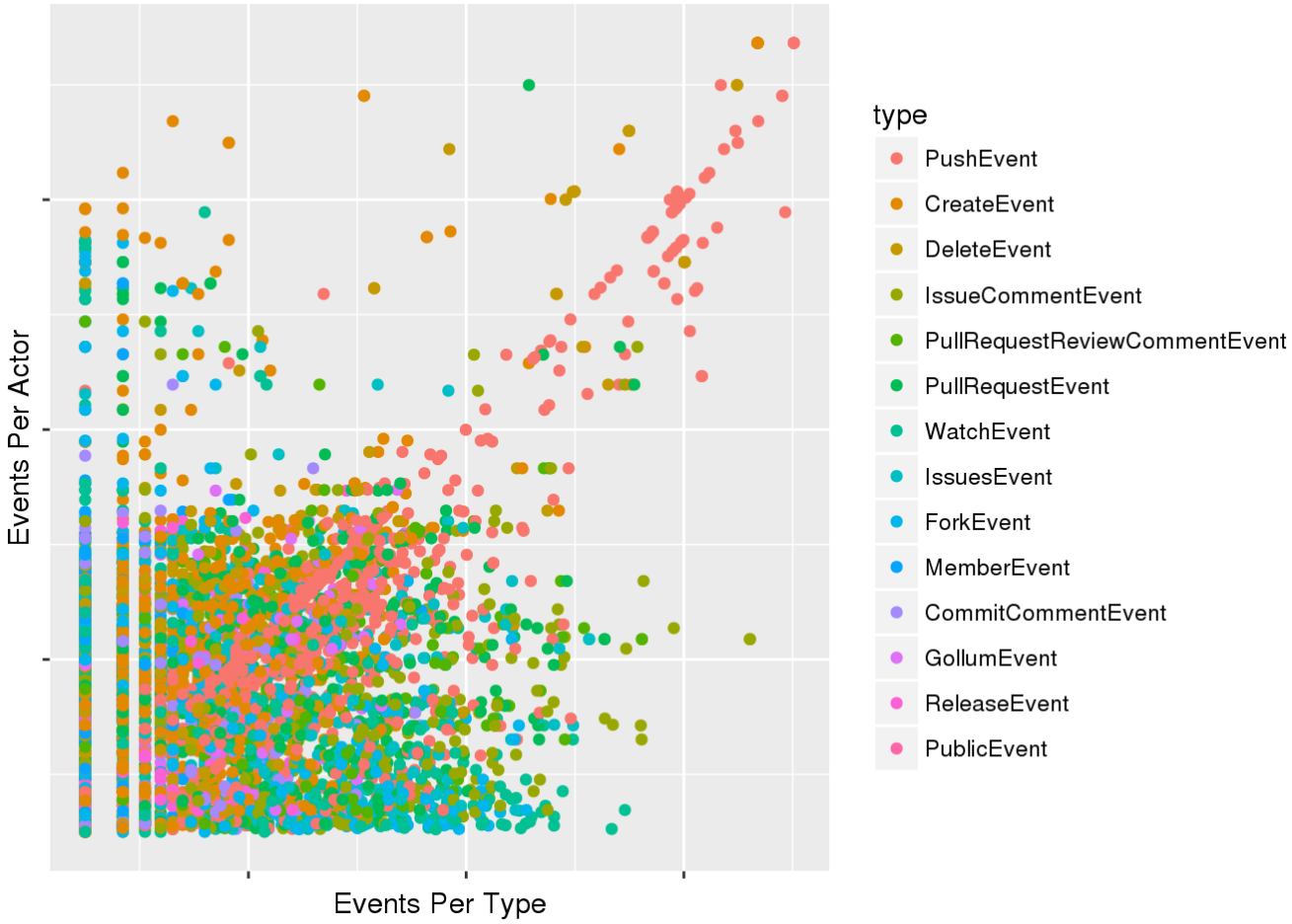


Event Types and Events per Actor

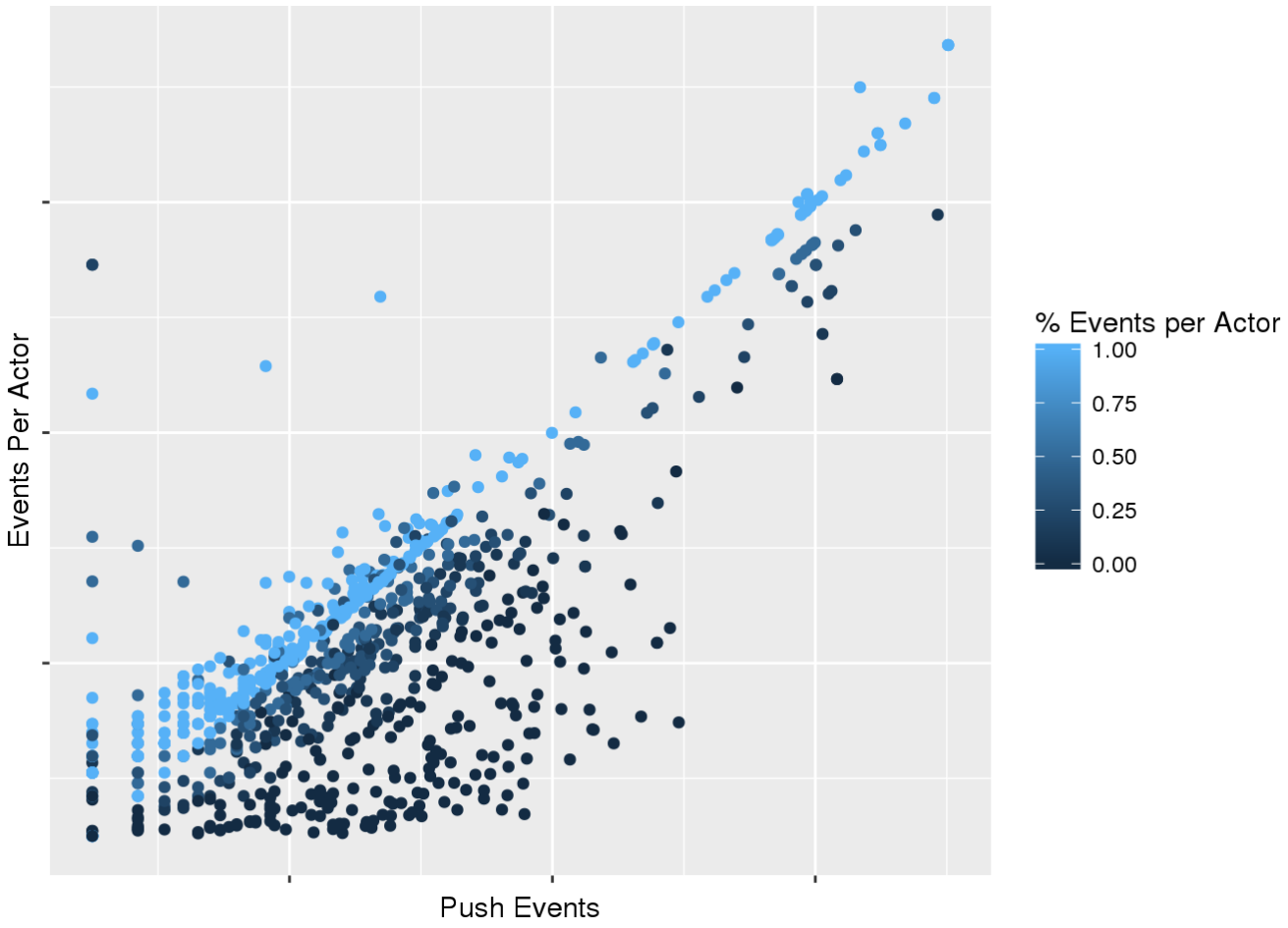
The next series of scatterplots explores the possible correlation between the frequency of an event type and the ratio of events per actor per repository. Ideally a higher events to actor ratio indicates less unique actors per repository. This does not tell us anything about the proportion of those events, so a percentage value has been applied when examining individual event types. The percentage value indicates what percentage of repository events the events per actor ratio represents. A higher percentage indicates the events are spread across a smaller number of actors.

Push events appear to be associated the highest events to actor ratios meaning they are more likely to represent repositories with a smaller number of unique actors. Other event types showed interesting patterns, like Watch and Fork events, that indicated a smaller events to actor ratio overall. For events other than Push events, however there does not appear to be a correlation between the frequency an event type occurs for a respository and the distribution of events overall per actor.

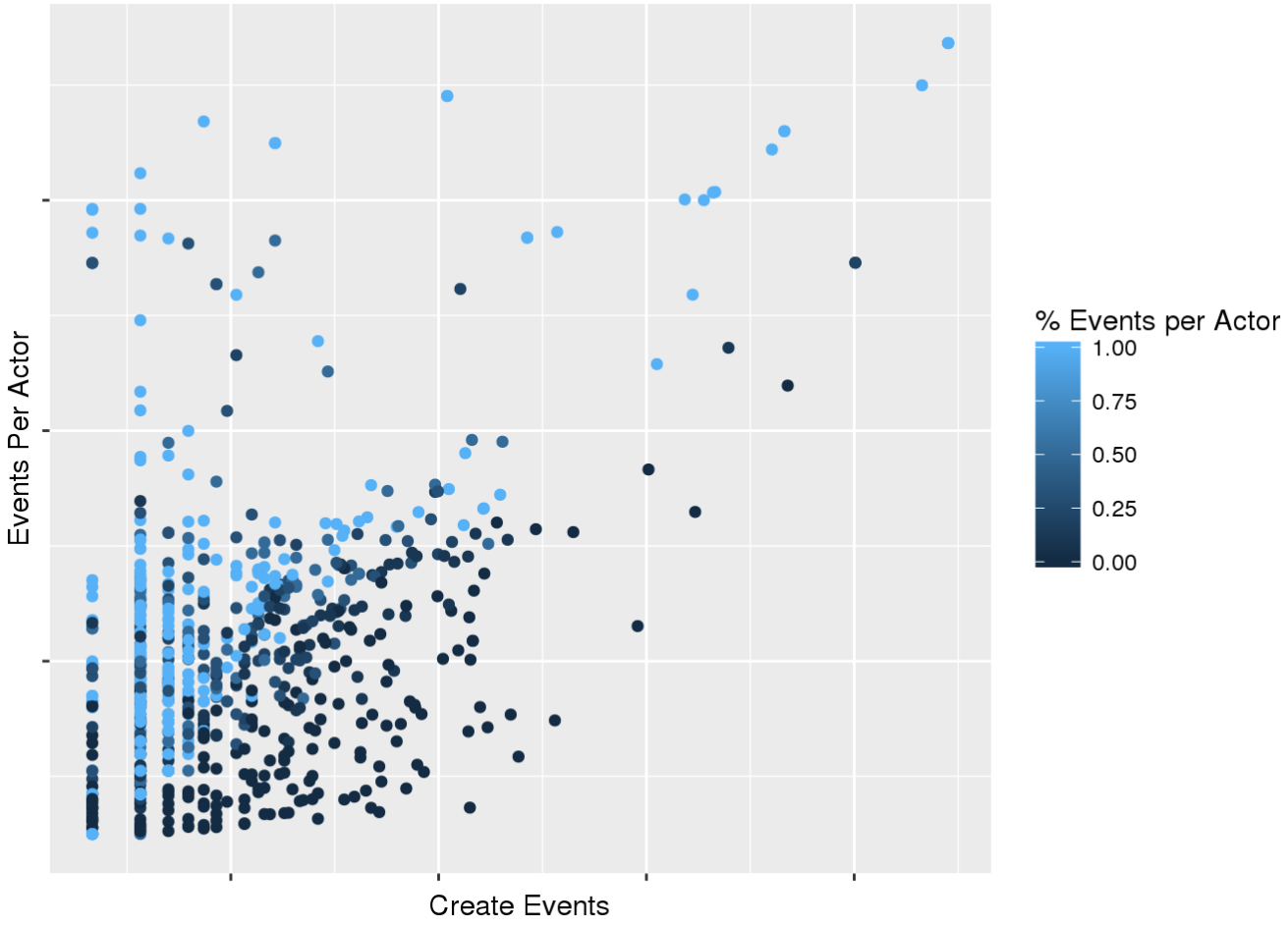
```
ggplot(data = events_repo_samples, aes(x=num_events, y=events_per_actor, fill=typ
e, colour=type, group=type)) +
  geom_point(stat="identity") +
  xlab("Events Per Type") +
  ylab("Events Per Actor") +
  scale_x_continuous(trans = "log", labels = NULL) +
  scale_y_continuous(trans = "log", labels = NULL)
```



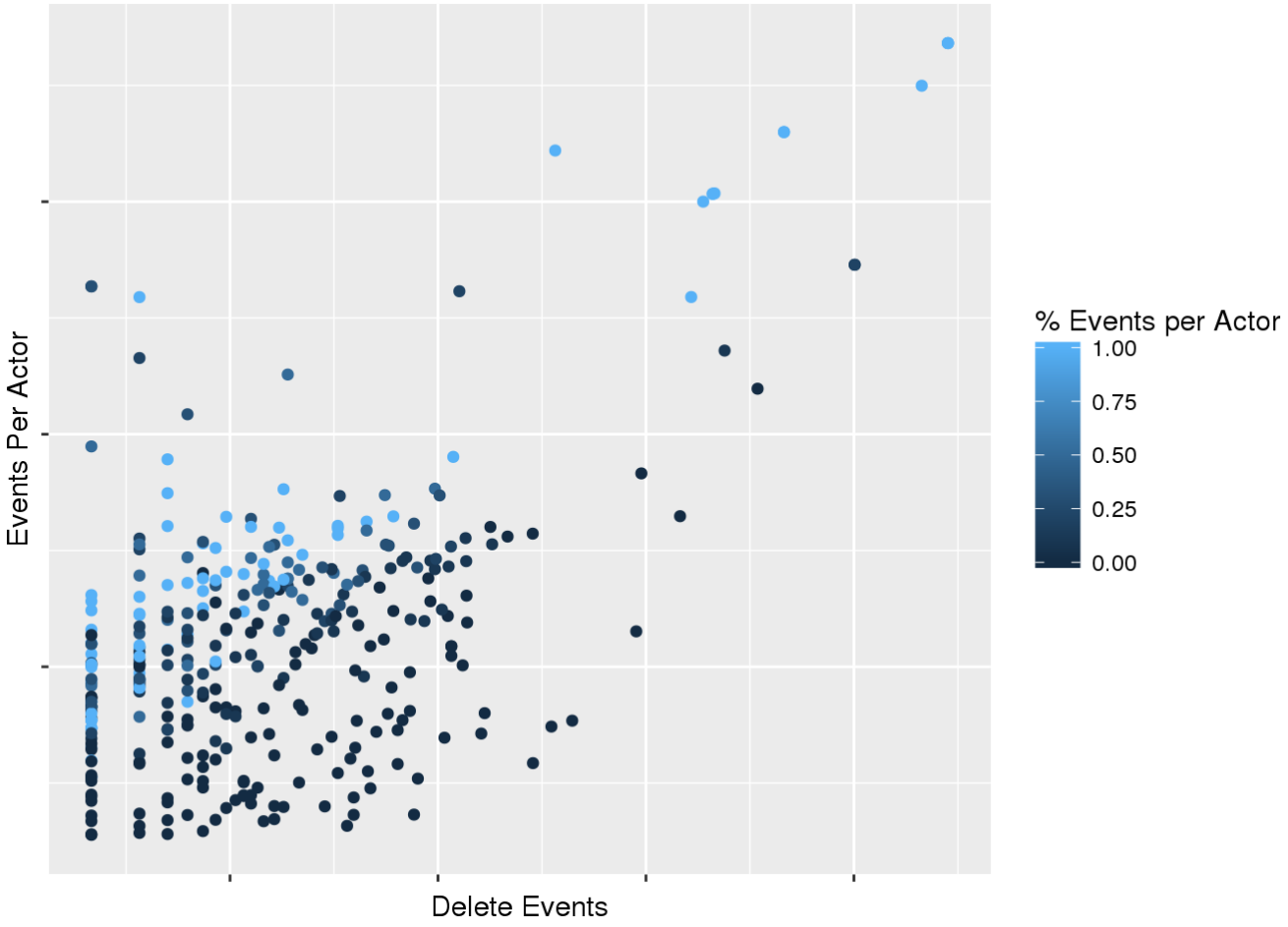

```
ggplot(data = events_repo_samples_push,
       aes(x=num_events, y=events_per_actor,
           fill=participation_rate, colour=participation_rate, group=participatio
n_rate)) +
  labs(colour = "% Events per Actor", fill = "% Events per Actor") +
  geom_point(stat="identity") +
  xlab("Push Events") +
  ylab("Events Per Actor") +
  scale_x_continuous(trans = "log", labels = NULL) +
  scale_y_continuous(trans = "log", labels = NULL)
```



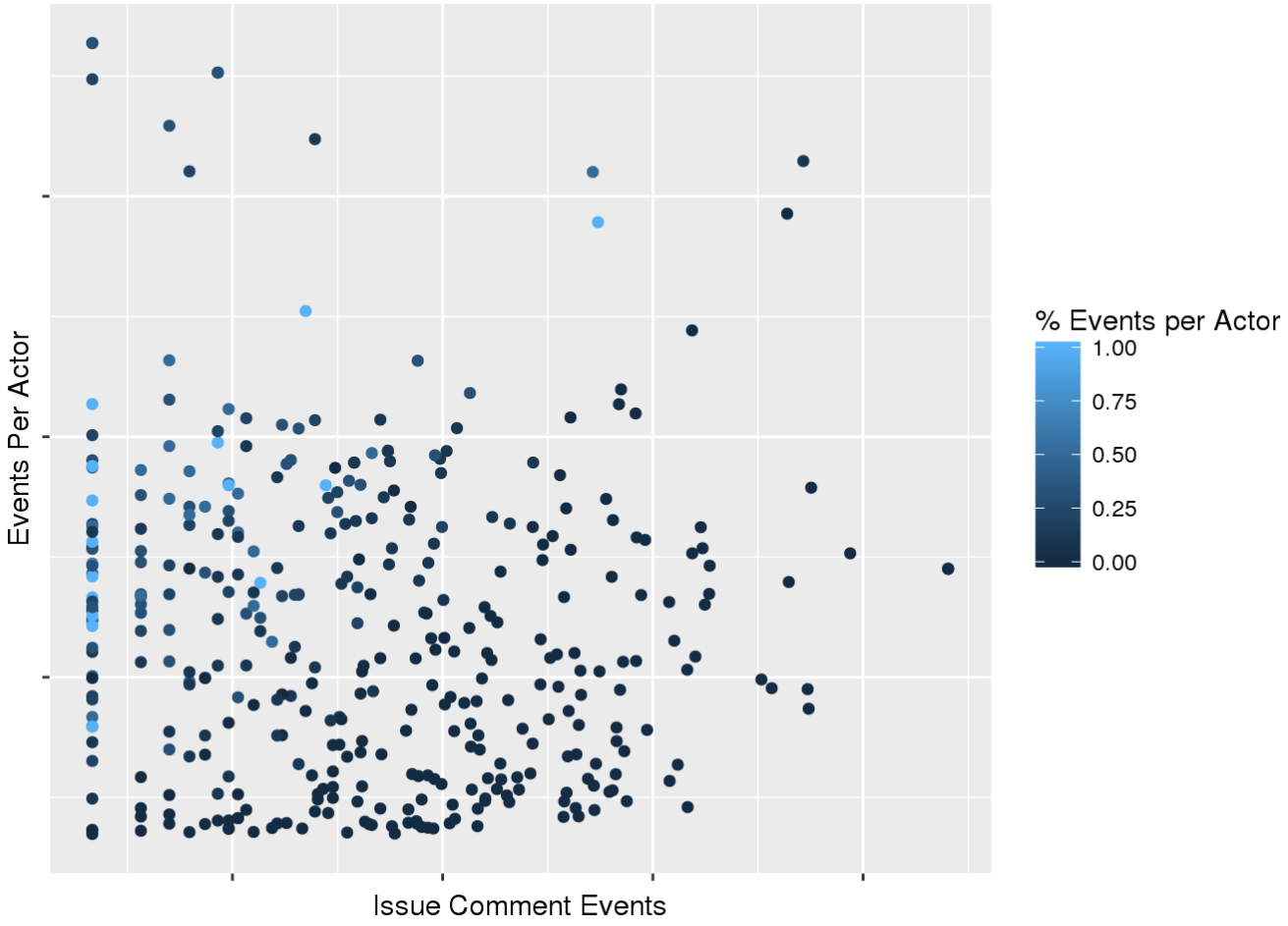

```
ggplot(data = events_repo_samples_create,
       aes(x=num_events, y=events_per_actor,
           fill=participation_rate, colour=participation_rate, group=participatio
n_rate)) +
  labs(colour = "% Events per Actor", fill = "% Events per Actor") +
  geom_point(stat="identity") +
  xlab("Create Events") +
  ylab("Events Per Actor") +
  scale_x_continuous(trans = "log", labels = NULL) +
  scale_y_continuous(trans = "log", labels = NULL)
```



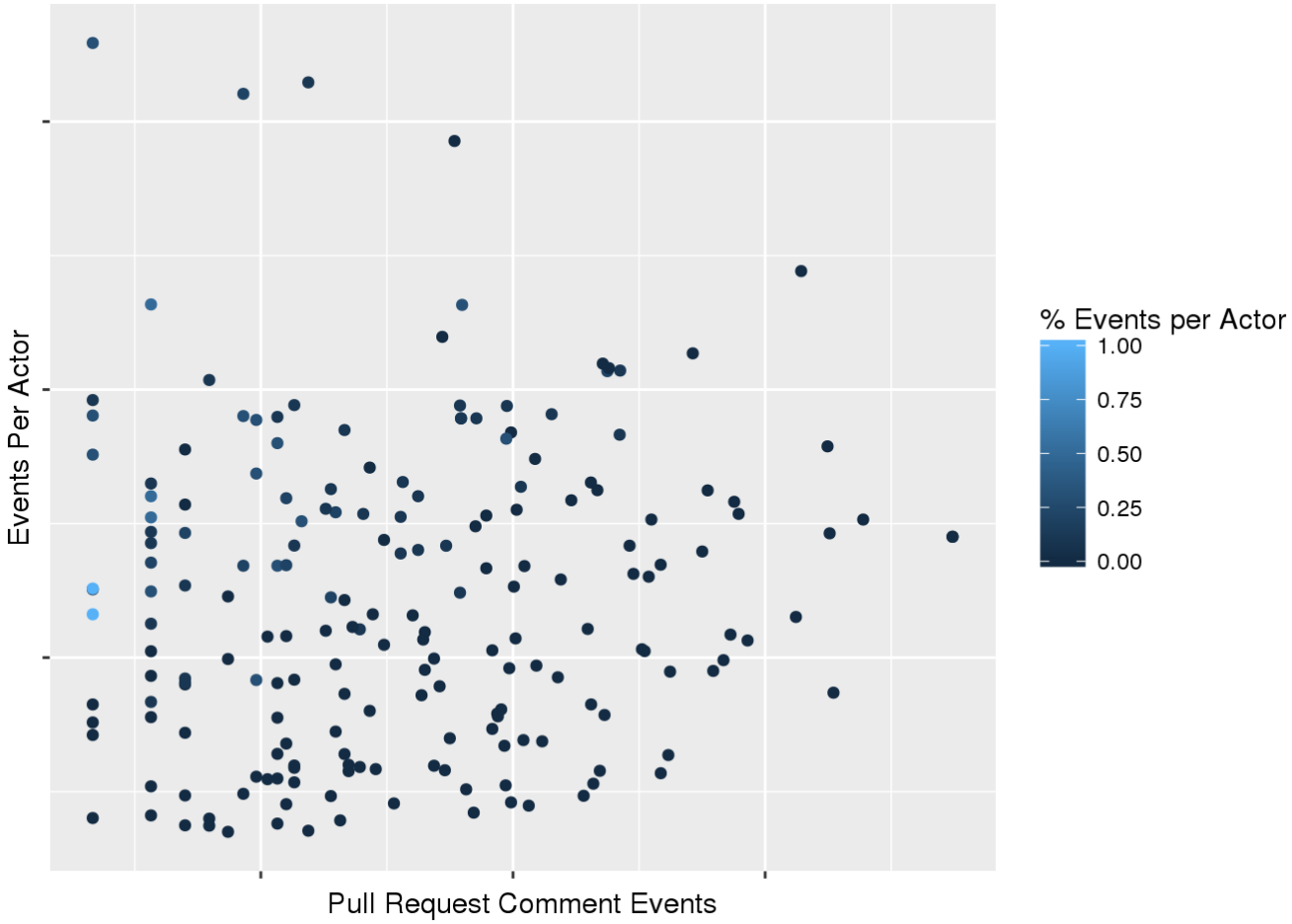

```
ggplot(data = events_repo_samples_del,
       aes(x=num_events, y=events_per_actor,
           fill=participation_rate, colour=participation_rate, group=participatio
n_rate)) +
  labs(colour = "% Events per Actor", fill = "% Events per Actor") +
  geom_point(stat="identity") +
  xlab("Delete Events") +
  ylab("Events Per Actor") +
  scale_x_continuous(trans = "log", labels = NULL) +
  scale_y_continuous(trans = "log", labels = NULL)
```



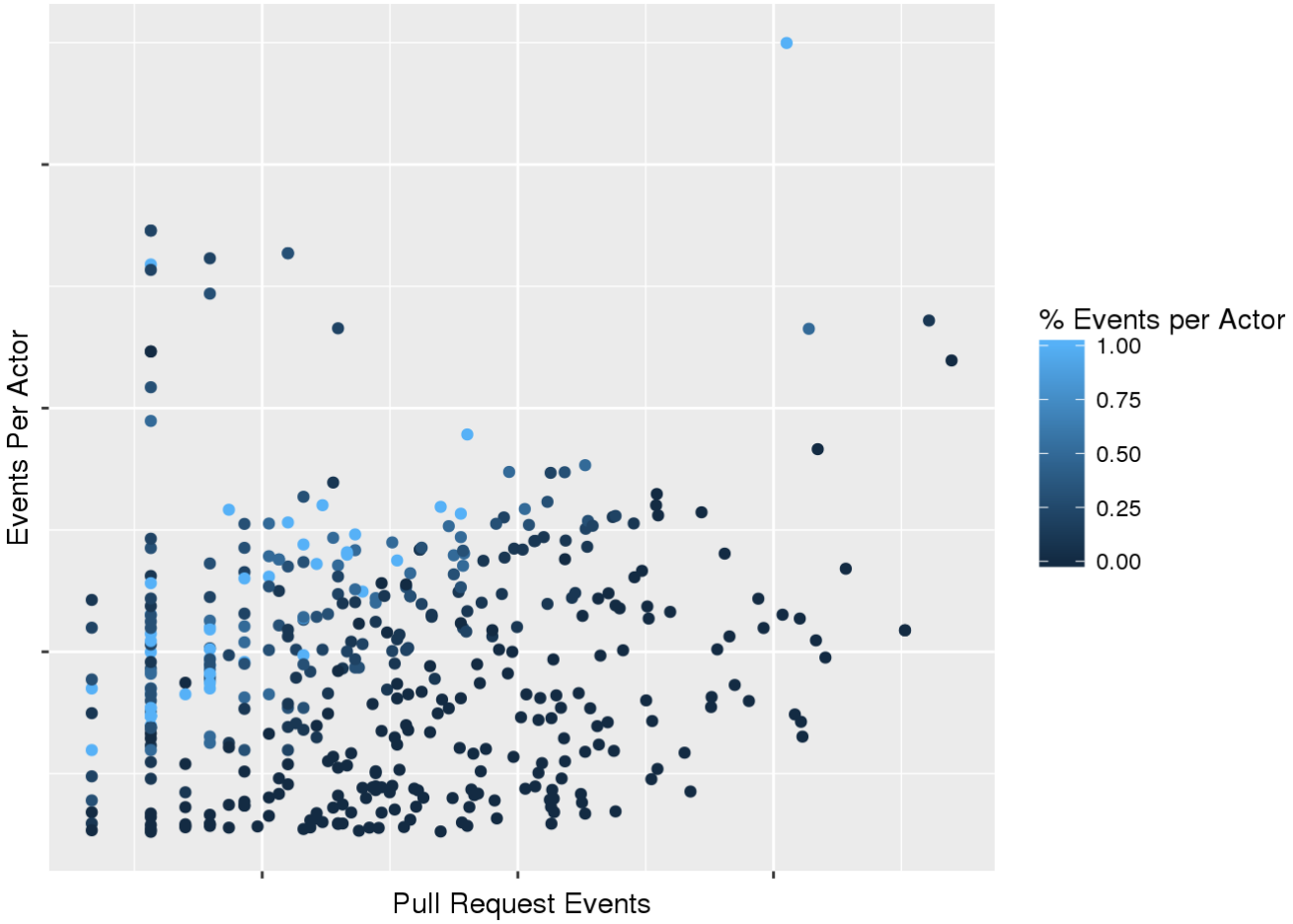

```
ggplot(data = events_repo_samples_issuecomment,
       aes(x=num_events, y=events_per_actor,
           fill=participation_rate, colour=participation_rate, group=participatio
n_rate)) +
  labs(colour = "% Events per Actor", fill = "% Events per Actor") +
  geom_point(stat="identity") +
  xlab("Issue Comment Events") +
  ylab("Events Per Actor") +
  scale_x_continuous(trans = "log", labels = NULL) +
  scale_y_continuous(trans = "log", labels = NULL)
```



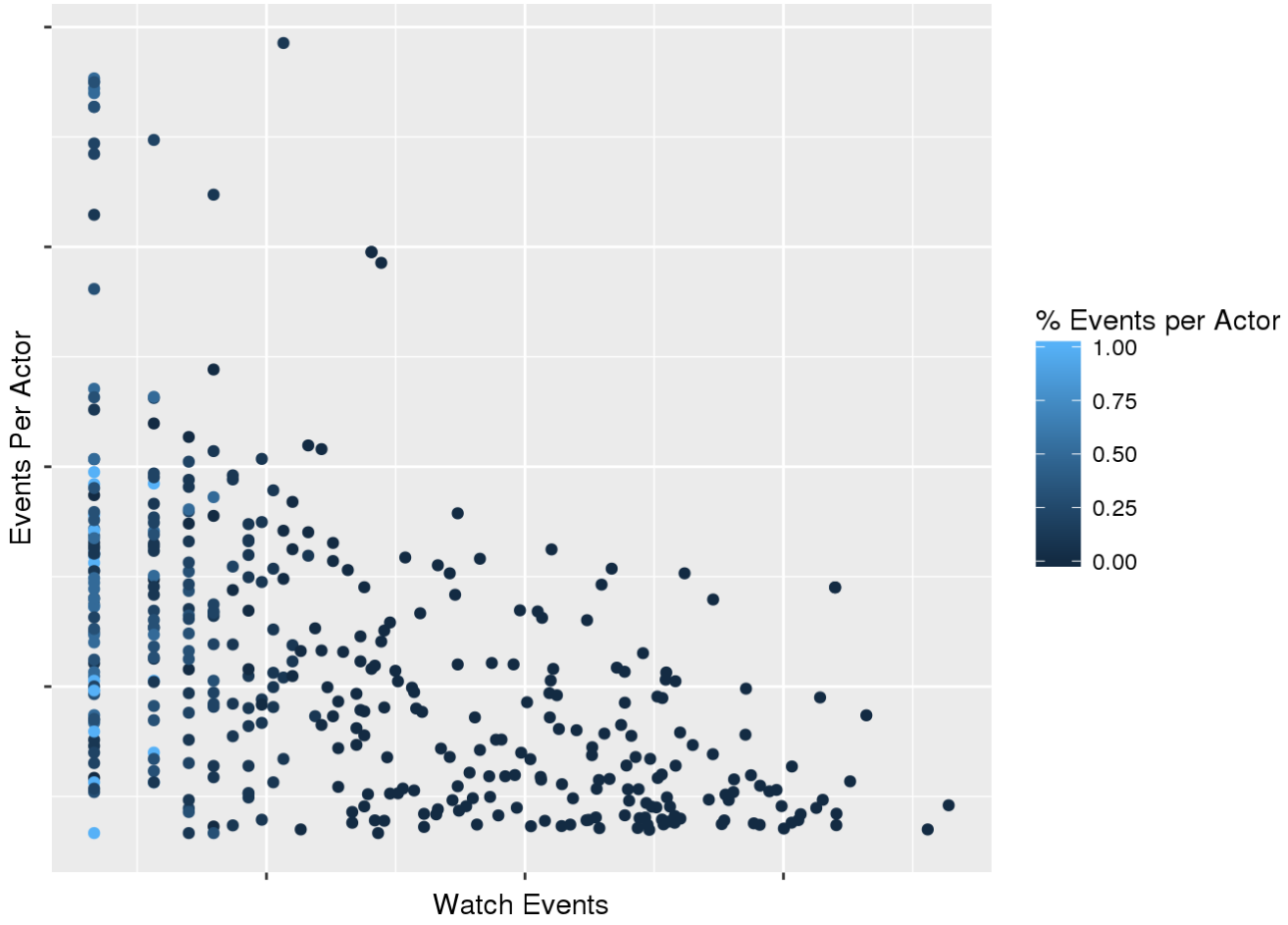

```
ggplot(data = events_repo_samples_prc,
       aes(x=num_events, y=events_per_actor,
           fill=participation_rate, colour=participation_rate, group=participatio
n_rate)) +
  labs(colour = "% Events per Actor", fill = "% Events per Actor") +
  geom_point(stat="identity") +
  xlab("Pull Request Comment Events") +
  ylab("Events Per Actor") +
  scale_x_continuous(trans = "log", labels = NULL) +
  scale_y_continuous(trans = "log", labels = NULL)
```




```
ggplot(data = events_repo_samples_pr,
       aes(x=num_events, y=events_per_actor,
           fill=participation_rate, colour=participation_rate, group=participatio
n_rate)) +
  labs(colour = "% Events per Actor", fill = "% Events per Actor") +
  geom_point(stat="identity") +
  xlab("Pull Request Events") +
  ylab("Events Per Actor") +
  scale_x_continuous(trans = "log", labels = NULL) +
  scale_y_continuous(trans = "log", labels = NULL)
```

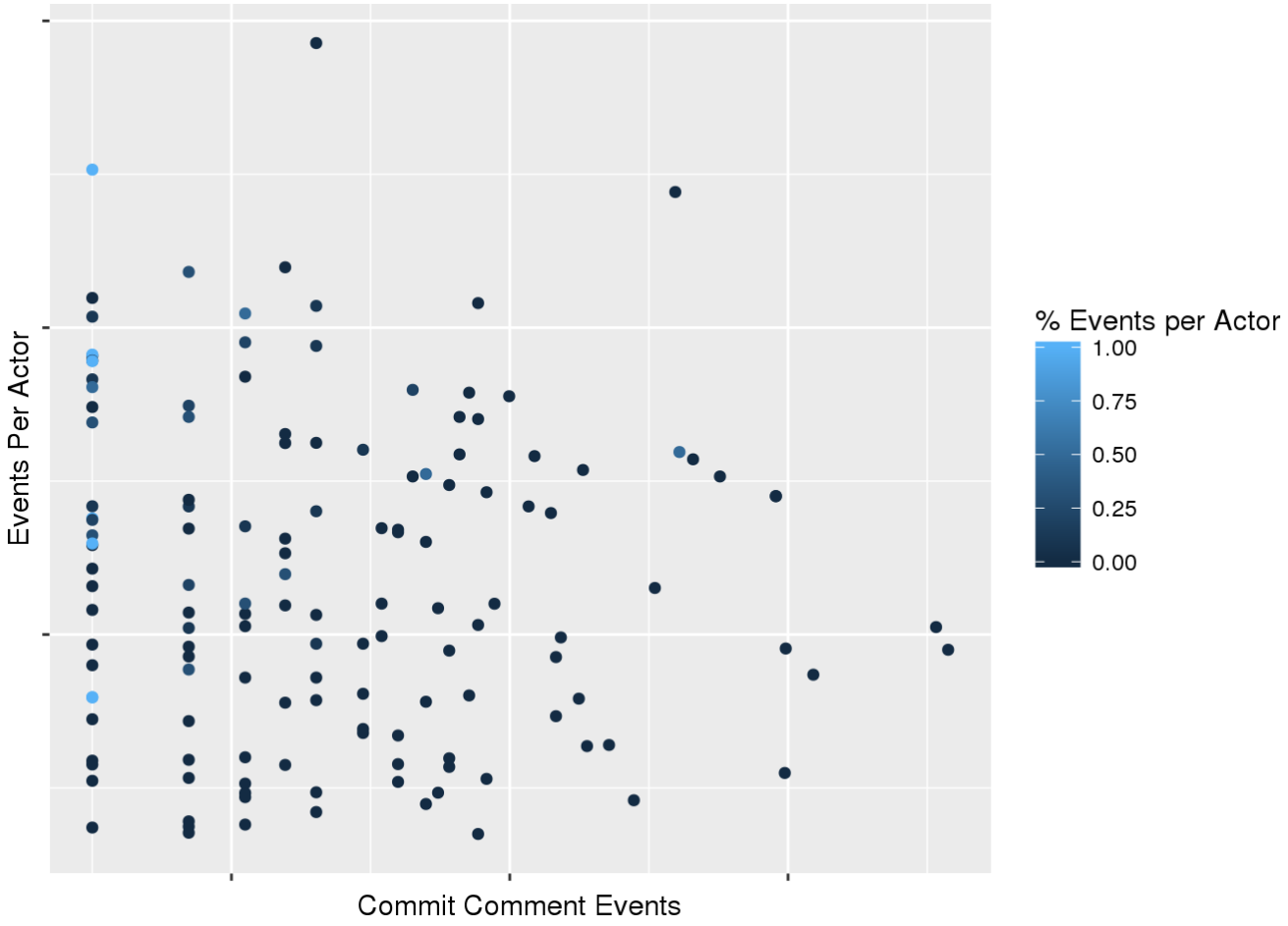



```
ggplot(data = events_repo_samples_watch,
       aes(x=num_events, y=events_per_actor,
           fill=participation_rate, colour=participation_rate, group=participatio
n_rate)) +
  labs(colour = "% Events per Actor", fill = "% Events per Actor") +
  geom_point(stat="identity") +
  xlab("Watch Events") +
  ylab("Events Per Actor") +
  scale_x_continuous(trans = "log", labels = NULL) +
  scale_y_continuous(trans = "log", labels = NULL)
```

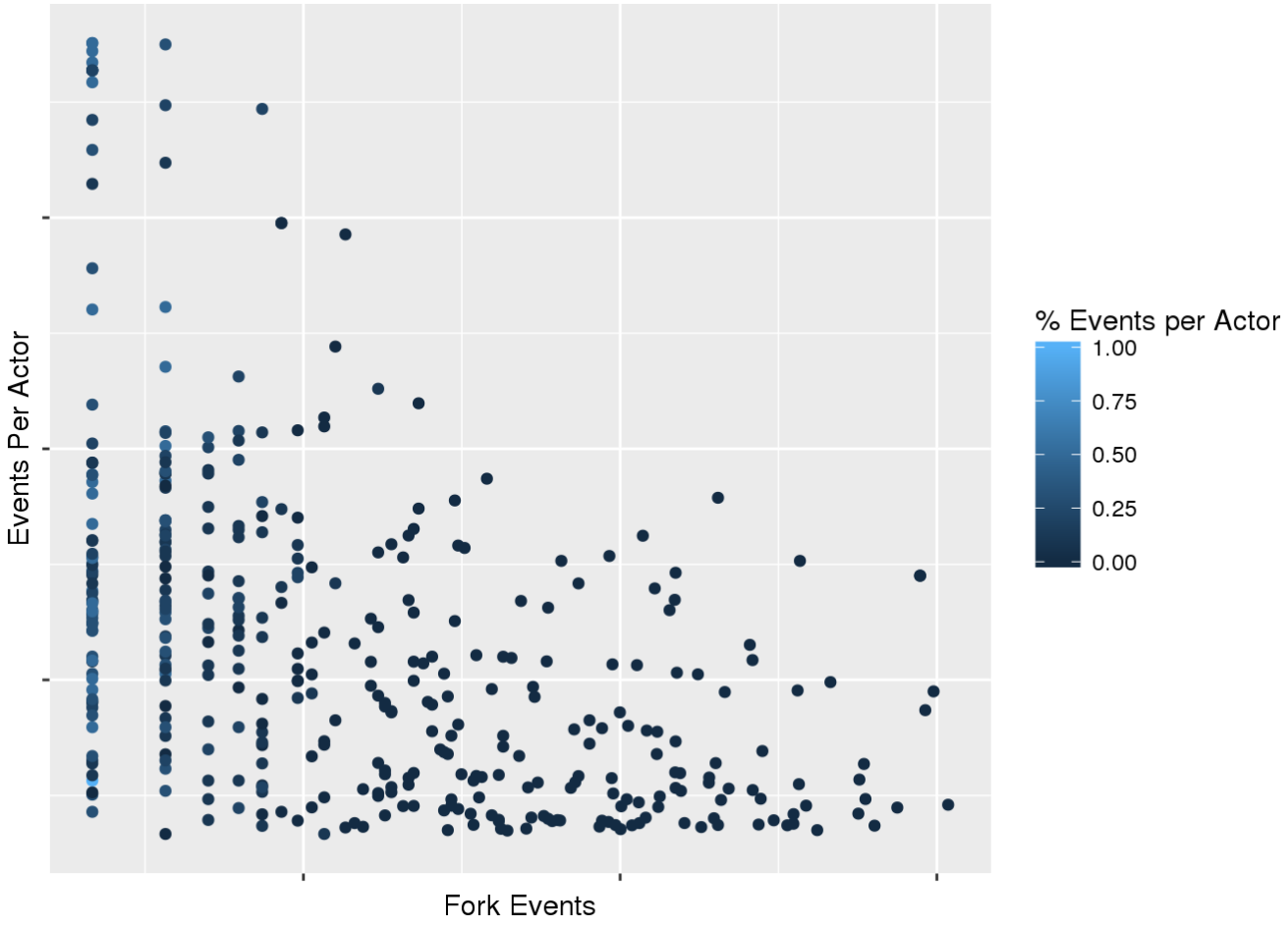


Less Frequent Event Types

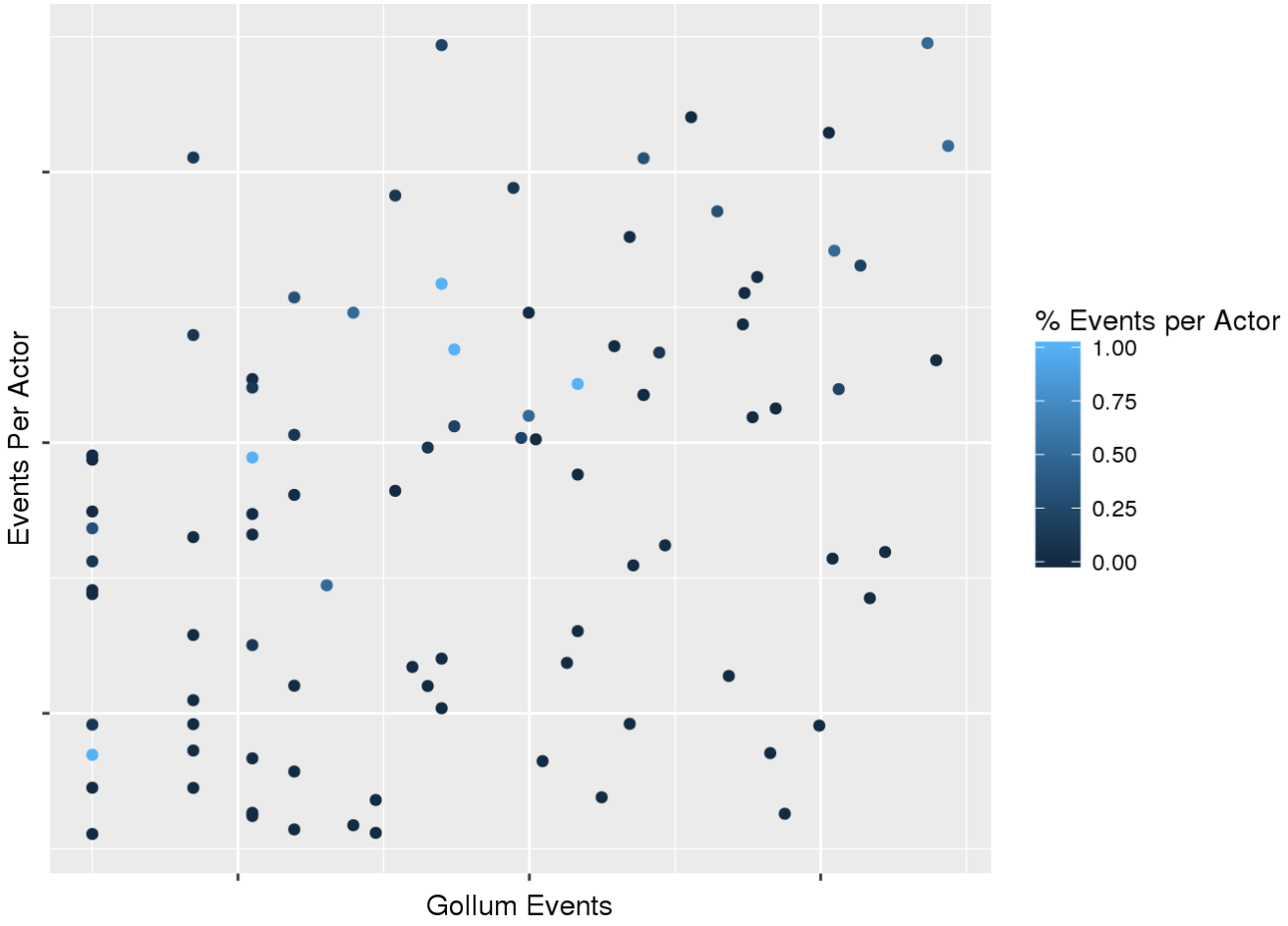
```
ggplot(data = events_repo_samples_comcom,
       aes(x=num_events, y=events_per_actor,
           fill=participation_rate, colour=participation_rate, group=participatio
n_rate)) +
  labs(colour = "% Events per Actor", fill = "% Events per Actor") +
  geom_point(stat="identity") +
  xlab("Commit Comment Events") +
  ylab("Events Per Actor") +
  scale_x_continuous(trans = "log", labels = NULL) +
  scale_y_continuous(trans = "log", labels = NULL)
```



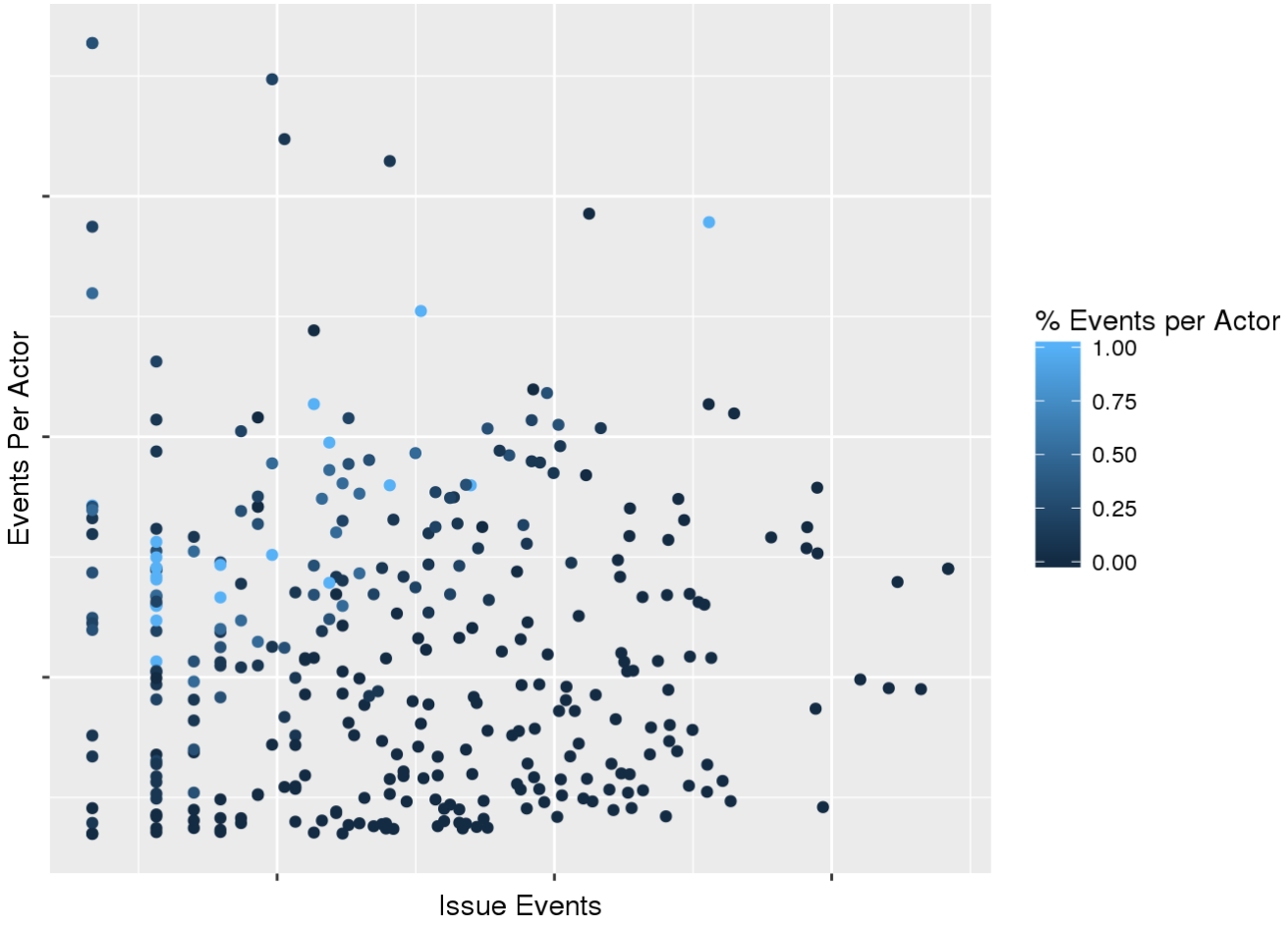

```
ggplot(data = events_repo_samples_fork,
       aes(x=num_events, y=events_per_actor,
           fill=participation_rate, colour=participation_rate, group=participatio
n_rate)) +
  labs(colour = "% Events per Actor", fill = "% Events per Actor") +
  geom_point(stat="identity") +
  xlab("Fork Events") +
  ylab("Events Per Actor") +
  scale_x_continuous(trans = "log", labels = NULL) +
  scale_y_continuous(trans = "log", labels = NULL)
```



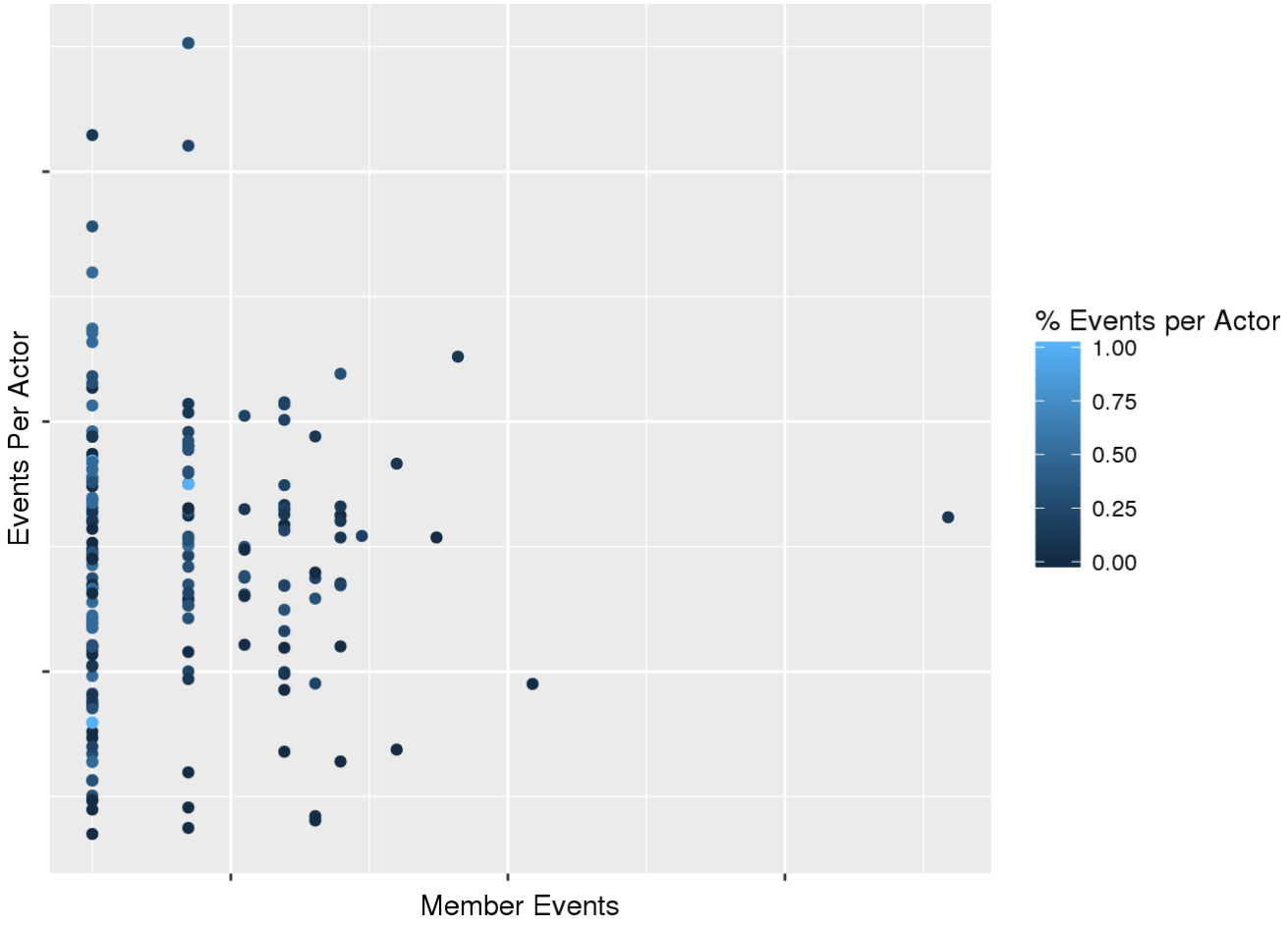

```
ggplot(data = events_repo_samples_gollum,
       aes(x=num_events, y=events_per_actor,
           fill=participation_rate, colour=participation_rate, group=participatio
n_rate)) +
  labs(colour = "% Events per Actor", fill = "% Events per Actor") +
  geom_point(stat="identity") +
  xlab("Gollum Events") +
  ylab("Events Per Actor") +
  scale_x_continuous(trans = "log", labels = NULL) +
  scale_y_continuous(trans = "log", labels = NULL)
```



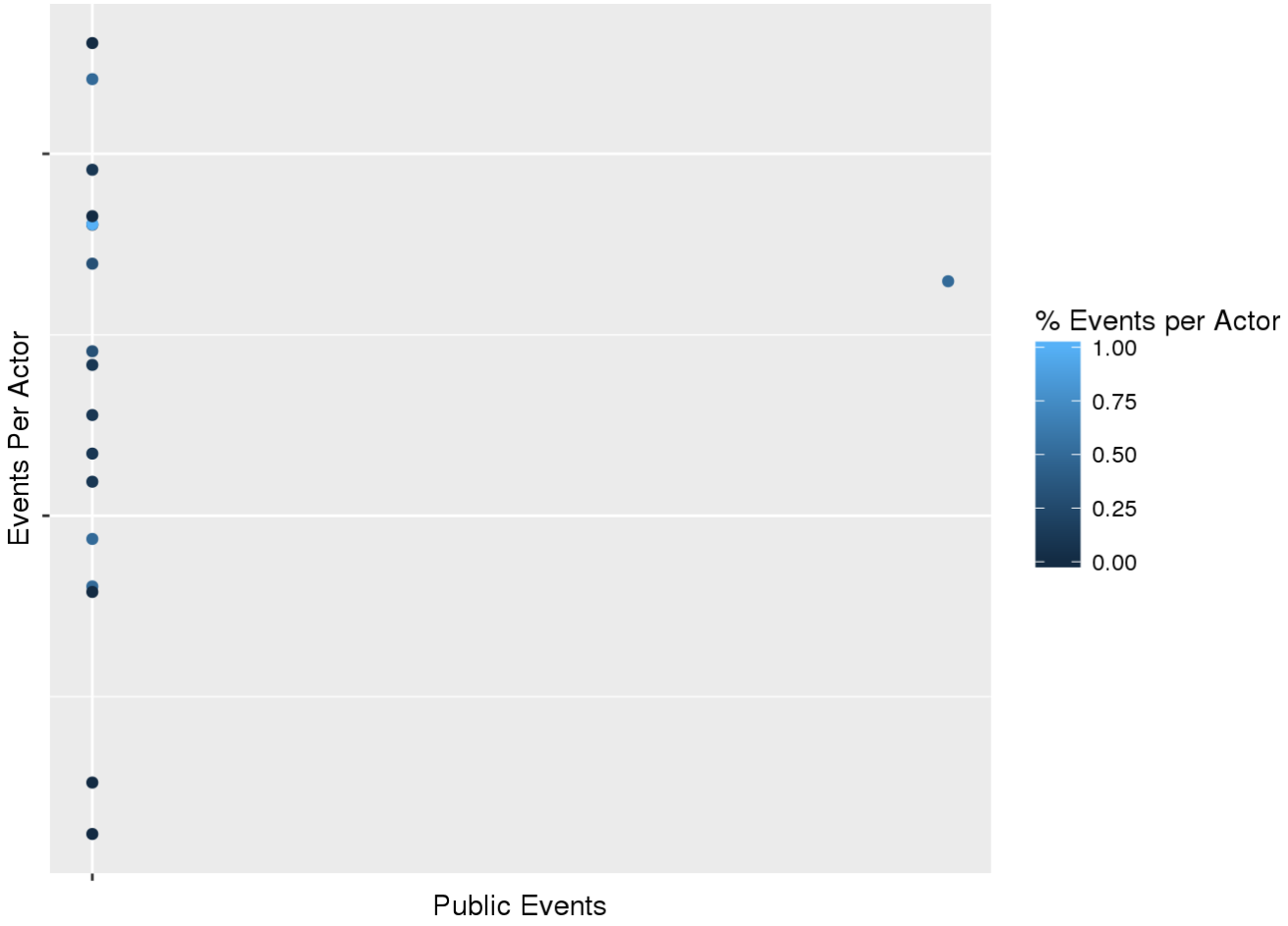

```
ggplot(data = events_repo_samples_issue,
       aes(x=num_events, y=events_per_actor,
           fill=participation_rate, colour=participation_rate, group=participatio
n_rate)) +
  labs(colour = "% Events per Actor", fill = "% Events per Actor") +
  geom_point(stat="identity") +
  xlab("Issue Events") +
  ylab("Events Per Actor") +
  scale_x_continuous(trans = "log", labels = NULL) +
  scale_y_continuous(trans = "log", labels = NULL)
```



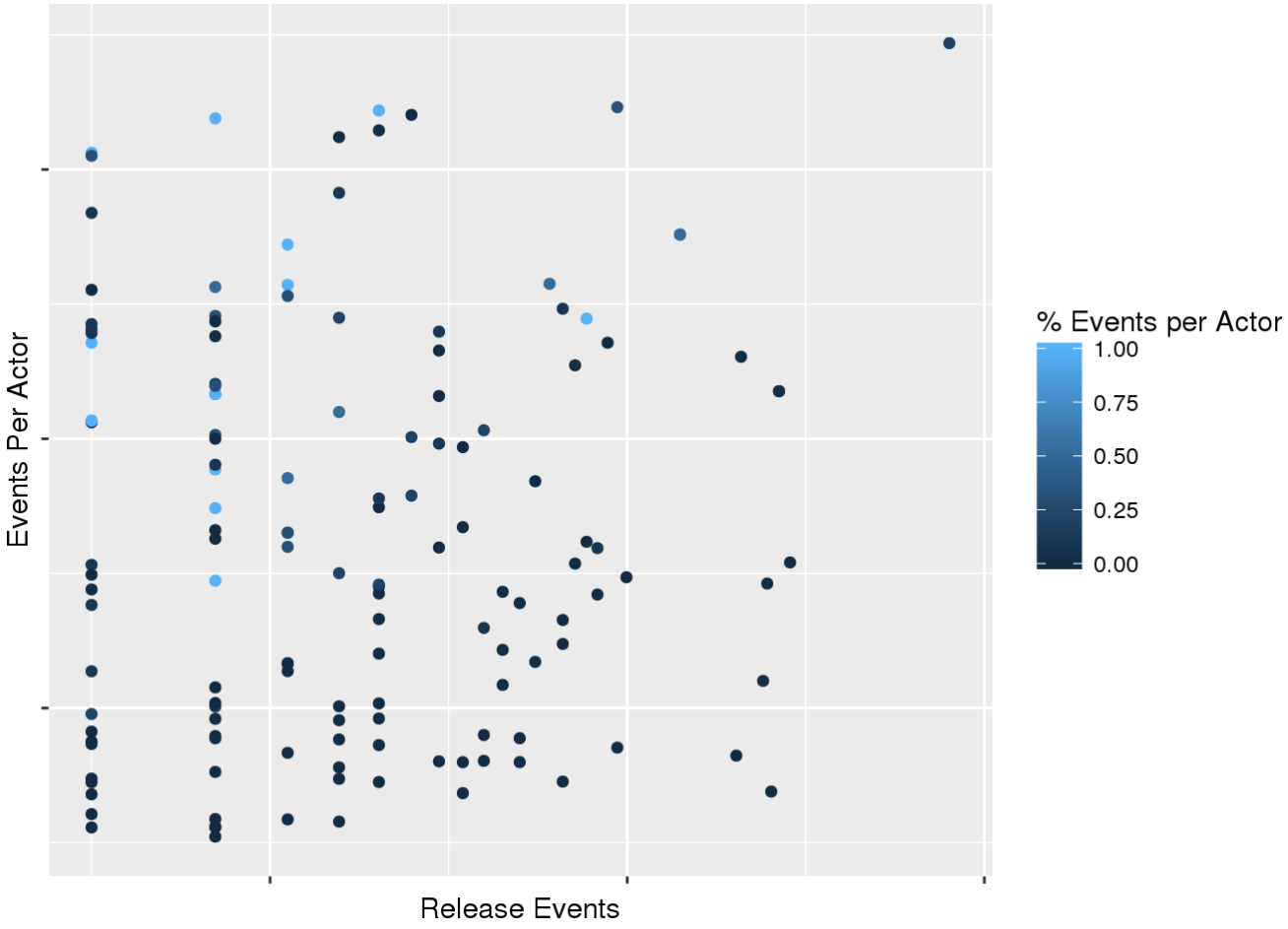

```
ggplot(data = events_repo_samples_member,
       aes(x=num_events, y=events_per_actor,
           fill=participation_rate, colour=participation_rate, group=participatio
n_rate)) +
  labs(colour = "% Events per Actor", fill = "% Events per Actor") +
  geom_point(stat="identity") +
  xlab("Member Events") +
  ylab("Events Per Actor") +
  scale_x_continuous(trans = "log", labels = NULL) +
  scale_y_continuous(trans = "log", labels = NULL)
```




```
ggplot(data = events_repo_samples_public,
       aes(x=num_events, y=events_per_actor,
           fill=participation_rate, colour=participation_rate, group=participatio
n_rate)) +
  labs(colour = "% Events per Actor", fill = "% Events per Actor") +
  geom_point(stat="identity") +
  xlab("Public Events") +
  ylab("Events Per Actor") +
  scale_x_continuous(trans = "log", labels = NULL) +
  scale_y_continuous(trans = "log", labels = NULL)
```




```
ggplot(data = events_repo_samples_release,
       aes(x=num_events, y=events_per_actor,
           fill=participation_rate, colour=participation_rate, group=participatio
n_rate)) +
  labs(colour = "% Events per Actor", fill = "% Events per Actor") +
  geom_point(stat="identity") +
  xlab("Release Events") +
  ylab("Events Per Actor") +
  scale_x_continuous(trans = "log", labels = NULL) +
  scale_y_continuous(trans = "log", labels = NULL)
```



Conclusions

The goal of this study was to determine if there was a relationship between the types of Github events and the number of unique actors that generated those events in each repository. Based on the data analysis presented above, it appears some event types show a possible correlation with overall actor involvement while others do not.

Event types that show the strongest possible correlation with unique actors are: Fork, Issue Comment, Watch, and Push events.

Next Steps

- Samples of repositories should be taken from the event types of interest and then further analyzed.
- The overall repository event distribution per actor metric should be further evaluated. How effective that metric at categorizing GitHub repositories? How does that metric computed from event data compares to a similar metric computed from actual GitHub repository data?