Process_Analysis

Code

This is R Notebook present a process analysis of 746 completed, fully validated, and archived projects in the HOT Tasking Manager (HOT-TM) over the past two years. The process is analised from four perspectives:

1. Control flow
2. Time
3. Organisation
4. Outcome

Process discovery was performed using bupaR, a suite of open-source R packages for business process data analysis.

Hide

```
# Import required libraries
suppressWarnings({
#install.packages("reshape2")
#install.packages("gt")
library(bupaverse)
library(reshape2)
library(gt)
library(scales)
library(readr)
library(dplyr)
library(magrittr)
library(ggplot2)
library(Hmisc)
library(gamlss)
})
```

Read event data

The log containing only initial tasks (“initial_tasks.csv”) is used in 1) Control flow, 2) Time, 3) Organisation sections, where a clean view of the process is required.

Hide

```
event_log_df <- read.csv("initial_tasks.csv", stringsAsFactors = FALSE, sep = ",")
event_log_df <- event_log_df %>%
  convert_timestamps(columns = c("start", "complete"), format = ymd_hms) %>%
  activitylog(case_id = "taskId", activity_id = "action", resource_id = "actionBy", time
stamps = c("start", "complete"))
head(event_log_df)
```

```
# Log of 12 events consisting of:
2 traces
5 cases
6 instances of 2 activities
1 resource
Events occurred from 2021-12-01 04:45:26 until 2021-12-01 04:54:13

# Variables were mapped as follows:
Case identifier:        taskId
Activity identifier:        action
Resource identifier:        actionBy
Timestamps:     start, complete
```

X	taskId	action	actionBy	start	compl
<int>	<chr>	<chr>	<chr>	<S3: POSIXct>	<S3: POSIX
0	11875_33	LOCKED_FOR_MAPPING	fajarramadhana	2021-12-01 04:45:26	2021-12-01 04:46
1	11875_72	LOCKED_FOR_MAPPING	fajarramadhana	2021-12-01 04:46:37	2021-12-01 04:47
2	11875_85	LOCKED_FOR_MAPPING	fajarramadhana	2021-12-01 04:47:24	2021-12-01 04:49
3	11875_42	LOCKED_FOR_MAPPING	fajarramadhana	2021-12-01 04:49:53	2021-12-01 04:53
4	11875_42	MAPPED	fajarramadhana	2021-12-01 04:53:23	2021-12-01 04:53
5	11875_29	LOCKED_FOR_MAPPING	fajarramadhana	2021-12-01 04:53:55	2021-12-01 04:54

6 rows

Hide

```
head(event_log_df)
```

Control flow

Absolute frequency of activities in the eventlog.

Hide

```
event_log_df %>% activity_frequency("activity")
```

action	absolute	relative
<chr>	<int>	<dbl>
LOCKED_FOR_MAPPING	406324	0.305448204
LOCKED_FOR_VALIDATION	308244	0.231717979
MAPPED	284214	0.213653773
VALIDATED	284035	0.213519212

action <chr>	absolute <int>	relative <dbl>
AUTO_UNLOCKED_FOR_MAPPING	20070	0.015087333
INVALIDATED	9429	0.007088115
SPLIT	7680	0.005773329
BADIMAGERY	4502	0.003384314
AUTO_UNLOCKED_FOR_VALIDATION	2956	0.002222130
EXTENDED_FOR_MAPPING	2801	0.002105611

1-10 of 10 rows

Activity presence shows in what percentage of cases an activity is present.

Hide

```
event_log_df %>% activity_presence()
```

action <chr>	absolute <int>	relative <dbl>
LOCKED_FOR_MAPPING	274340	0.987740553
LOCKED_FOR_VALIDATION	271218	0.976500027
MAPPED	271187	0.976388414
VALIDATED	270091	0.972442348
AUTO_UNLOCKED_FOR_MAPPING	14733	0.053045059
INVALIDATED	8201	0.029527084
SPLIT	7680	0.027651263
BADIMAGERY	4124	0.014848152
AUTO_UNLOCKED_FOR_VALIDATION	2796	0.010066788
EXTENDED_FOR_MAPPING	766	0.002757925

1-10 of 10 rows

The start of cases can be described using the start_activities function.

Hide

```
event_log_df %>% start_activities("activity")
```

action <chr>	absolute <int>	relative <dbl>	cum_sum <dbl>
LOCKED_FOR_MAPPING	265857	9.571981e-01	0.9571981

action <chr>	absolute <int>	relative <dbl>	cum_sum <dbl>
AUTO_UNLOCKED_FOR_MAPPING	8598	3.095645e-02	0.9881546
SPLIT	3095	1.114331e-02	0.9992979
MAPPED	79	2.844336e-04	0.9995824
BADIMAGERY	60	2.160255e-04	0.9997984
LOCKED_FOR_VALIDATION	48	1.728204e-04	0.9999712
VALIDATED	4	1.440170e-05	0.9999856
INVALIDATED	3	1.080127e-05	0.9999964
EXTENDED_FOR_MAPPING	1	3.600425e-06	1.0000000

9 rows

The end_activities function describes the end of cases

Hide

```
event_log_df %>% end_activities("activity")
```

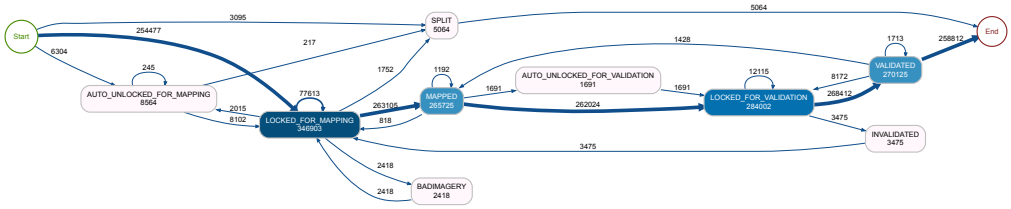
action <chr>	absolute <int>	relative <dbl>	cum_sum <dbl>
VALIDATED	269844	9.715530e-01	0.9715530
SPLIT	7642	2.751445e-02	0.9990675
BADIMAGERY	185	6.660786e-04	0.9997336
AUTO_UNLOCKED_FOR_MAPPING	38	1.368161e-04	0.9998704
MAPPED	30	1.080127e-04	0.9999784
LOCKED_FOR_VALIDATION	4	1.440170e-05	0.9999928
AUTO_UNLOCKED_FOR_VALIDATION	2	7.200850e-06	1.0000000

7 rows

In the frequency process map, nodes represent the absolute number of activity instance executions and edges represent the absolute number of times source and target activities were executed directly following each other. To provide a clear process map, the event log was previously filter using filter_trace_frequency(). Setting percentage = 0.95 selects at least 95% of the cases, starting with those that have the highest frequency.

Hide

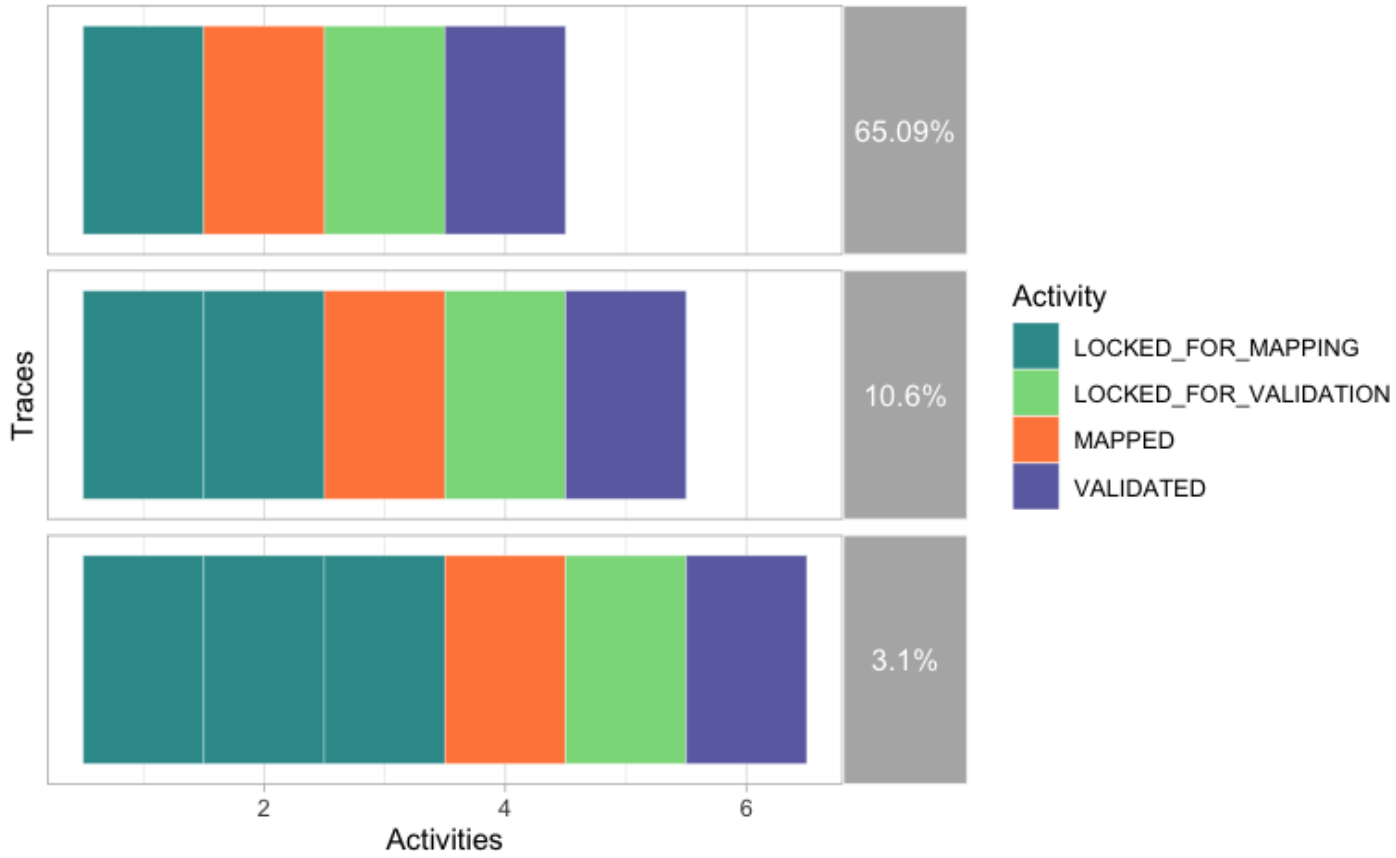
```
tmp <- event_log_df %>% filter_trace_frequency(percentage = 0.95)
tmp %>%  process_map(frequency("absolute"))
```



trace_explorer() with coverage argument n_traces = 3 shows the 3 most frequent in the event log.

Hide

```
event_log_df %>%
    trace_explorer(n_traces = 3, show_labels = FALSE, coverage_labels = c("relative"))
```

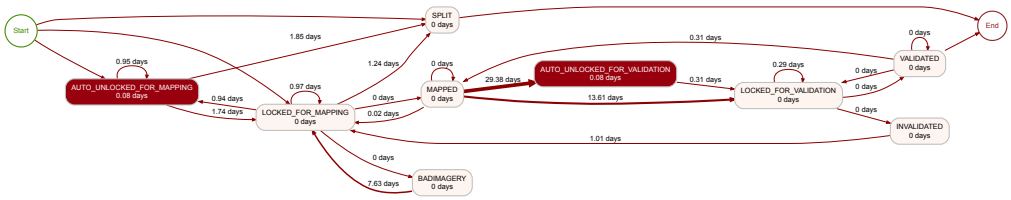


Time

In the temporal process map, the value of nodes and edges represent the median duration in days of activities and waiting times.

Hide

```
tmp %>%  process_map(performance(median, "days"))
```



Timestamps are properly formatted for time calculations

Hide

```
event_log_df <- event_log_df %>% convert_timestamps(columns = c("start", "complete"), fo
rmat = ymd_hms)
event_log_df$time_diff=as.numeric(event_log_df$complete-event_log_df$start)
task_duration <- event_log_df %>%  group_by(taskId) %>%  summarise(min = min(start), max
= max(complete))
task_duration$duration=as.numeric(task_duration$max-task_duration$min)
task_duration <- task_duration[,c("taskId","duration")]
```

We calculated the relative time devoted for mapping activities (‘LOCKED_FOR_MAPPING’, ‘AUTO_UNLOCKED_FOR_MAPPING’) and validation activities (‘LOCKED_FOR_VALIDATION’,‘AUTO_UNLOCKED_FOR_VALIDATION’) per case expressed as percentage of total case duration. The remaining time is considered idle.

Hide

```
mapping_duration <- filter(event_log_df, action == 'LOCKED_FOR_MAPPING' | action == 'AUT
O_UNLOCKED_FOR_MAPPING') %>%  group_by(taskId) %>% summarise(mapping = sum(time_diff))
validation_duration <- filter(event_log_df, action == 'LOCKED_FOR_VALIDATION' | action =
= 'AUTO_UNLOCKED_FOR_VALIDATION') %>%  group_by(taskId) %>% summarise(validation = sum(t
ime_diff))
durations <-  merge(x = merge(x = task_duration, y = mapping_duration, by = "taskId", al
l.x = TRUE), y = validation_duration, by = "taskId", all.x = TRUE)
durations[is.na(durations)] <- 0
durations$mapping_per=durations$mapping/durations$duration*100
durations$validation_per=durations$validation/durations$duration*100
durations$service=durations$mapping_per+durations$validation_per
durations$iddle_per=100-durations$mapping_per-durations$validation_per
```

Median % of iddle time per case

Hide

```
median(durations$iddle_per, na.rm = TRUE )
```

```
[1] 99.93861
```

Median % of mapping and validation time per case

Hide

```
median(durations$service, na.rm = TRUE )
```

```
[1] 0.06139425
```

Median % of mapping time per case

Hide

```
median(durations$mapping_per, na.rm = TRUE )
```

```
[1] 0.01586283
```

Median % of validation time per case

Hide

```
median(durations$validation_per, na.rm = TRUE )
```

```
[1] 0.02293203
```

Organisation

The contributor profile (“contributors.csv”) is added to the event log to calculate the relative frequency with which contributors according to their mapping level execute each type of activity.

Hide

```
contributors <- read.csv("contributors.csv", stringsAsFactors = FALSE, sep = ",")
event_log_df <- merge(event_log_df, contributors, by.x = "actionBy", by.y = "username")
head(event_log_df)
```

	actionBy <chr>	X <int>	taskId <chr>	action <chr>	star <S3: POSIXct>
1	Jean Yves Garinet	897682	13680_57	AUTO_UNLOCKED_FOR_MAPPING	2022-10-30 20:47:58
2	Jean Yves Garinet	1068988	14301_8	LOCKED_FOR_MAPPING	2023-03-23 20:27:36
3	Jean Yves Garinet	1156728	14637_3465	LOCKED_FOR_MAPPING	2023-06-21 11:36:49
4	Jean Yves Garinet	1156729	14637_3284	LOCKED_FOR_MAPPING	2023-06-21 11:36:49
5	Jean Yves Garinet	1156727	14637_3303	LOCKED_FOR_MAPPING	2023-06-21 11:36:49
6	____Melissa____	1046446	14235_1766	AUTO_UNLOCKED_FOR_MAPPING	2023-02-14 16:11:57

6 rows | 1-6 of 8 columns

Composition of the total number of contributors of the analysed projects according to their mapping level.

Hide

```
mappingLevel <- event_log_df %>%  group_by(mappingLevel) %>% summarise(count = n_distinct(actionBy))
mappingLevel$percentage <- round(mappingLevel$count/sum(mappingLevel$count)*100,1)
mappingLevel
```

mappingLevel <chr>	count <int>	percentage <dbl>
ADVANCED	2219	6.7
BEGINNER	29921	90.7
INTERMEDIATE	852	2.6

3 rows

Breakdown of status execution frequency per mapping level.

Hide

```
data_pivot <- dcast(event_log_df, action ~ mappingLevel,value.var = "taskId", length)
data_pivot$sum <-  data_pivot$ADVANCED + data_pivot$BEGINNER + data_pivot$INTERMEDIATE
data_pivot$ADVANCEDper <- round(data_pivot$ADVANCED/data_pivot$sum*100,1)
data_pivot$BEGINNERper <- round(data_pivot$BEGINNER/data_pivot$sum*100,1)
data_pivot$INTERMEDIATEper <- round(data_pivot$INTERMEDIATE/data_pivot$sum*100,1)
data_pivot[c("action","ADVANCEDper","BEGINNERper","INTERMEDIATEper")] %>% gt() %>%  data_color(columns = 2:4, colors = col_numeric(palette = c("white","darkgreen"),domain = c(0,100)))
```

action	ADVANCEDper	BEGINNERper	INTERMEDIATEper
AUTO_UNLOCKED_FOR_MAPPING	19.2	74.4	6.4
AUTO_UNLOCKED_FOR_VALIDATION	96.0	1.8	2.2
BADIMAGERY	28.1	68.0	3.8
EXTENDED_FOR_MAPPING	96.1	3.3	0.6
INVALIDATED	94.3	2.6	3.1
LOCKED_FOR_MAPPING	42.5	50.7	6.7
LOCKED_FOR_VALIDATION	98.6	0.5	0.9
MAPPED	48.7	45.1	6.2
SPLIT	66.9	27.8	5.3
VALIDATED	98.9	0.3	0.8

Outcome

The regression.csv file contains the total list of tasks with columns describing either the absolute frequency of executions of each type of activity (“activityname”) or a binary flag (1/0) indicating whether or not the activity was executed (“activityname1”) on that task.

Hide

```
regression <- read.csv("regression.csv", stringsAsFactors = FALSE, sep = ",")
regression$projId=as.character(regression$projId)
regression$percentage_area_covered_by_building = regression$percentage_area_covered_by_b
uilding / 100
#A few records have insignificant negative values attributed to rounding
regression$percentage_area_covered_by_building[regression$percentage_area_covered_by_bui
lding < 0] <- 0
regression = na.omit(regression)
```

Regression coefficients from the GAMLSS-BEZI (https://www.rdocumentation.org/packages/gamlss.dist/versions/6.1-1/topics/BEZI) model

Hide

```
model <- gamlss(percentage_area_covered_by_building ~ splits1 + invalidations1 + locked_
for_mappings + badimagery1 + area_sqm + difficulty,  family = BEZI, data = regression, t
race = F)
```

```
Warning: Algorithm RS has not yet converged
```

Hide

```
summary(model)
```

```
******************************************************************
Family:  c("BEZI", "Zero Inflated Beta")

Call:  gamlss(formula = percentage_area_covered_by_building ~      splits1 + invalidatio
ns1 + locked_for_mappings +
        badimagery1 + area_sqm + difficulty, family = BEZI,      data = regression, trac
e = F)

Fitting method: RS()

------------------------------------------------------------------
Mu link function:  logit
Mu Coefficients:
                     Estimate Std. Error  t value Pr(>|t|)
(Intercept)         -3.490e+00  6.852e-03 -509.305   <2e-16 ***
splits1              4.414e-01  7.620e-03   57.924   <2e-16 ***
invalidations1       1.595e-01  1.102e-02   14.465   <2e-16 ***
locked_for_mappings  2.475e-02  1.026e-03   24.127   <2e-16 ***
badimagery1          2.744e-01  1.483e-02   18.505   <2e-16 ***
area_sqm            -1.363e-08  5.395e-10  -25.254   <2e-16 ***
difficultyEASY       8.706e-03  4.188e-03    2.079   0.0376 *
difficultyMODERATE  -3.100e-01  6.941e-03  -44.670   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

------------------------------------------------------------------
Sigma link function:  log
Sigma Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.78150    0.00534   333.6   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

------------------------------------------------------------------
Nu link function:  logit
Nu Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.527404   0.003707  -142.3   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

------------------------------------------------------------------
No. of observations in the fit:  311808
Degrees of Freedom for the fit:  10
      Residual Deg. of Freedom:  311798
                      at cycle:  20

Global Deviance:     -1472183
            AIC:     -1472163
```

```
            SBC:     -1472057
********************************************************************
```

Dropping model to single model terms

Hide

```
drop1(model, parallel = "multicore", ncpus = 4)
```

```
Warning: Algorithm RS has not yet convergedWarning: Algorithm RS has not yet convergedWa
rning: Algorithm RS has not yet convergedWarning: Algorithm RS has not yet converged
```

Describe percentage_area_covered_by_building for all projects

Hide

```
regression$percentage_area_covered_by_building = regression$percentage_area_covered_by_b
uilding * 100
describe(regression$percentage_area_covered_by_building)
```

```
regression$percentage_area_covered_by_building
       n  missing distinct     Info     Mean      Gmd      .05      .10      .25      .5
0      .75      .90      .95
  311808        0   195041    0.949    1.909    3.294  0.00000  0.00000  0.00000  0.0649
9  0.92208  4.98023 11.29157

lowest : 0           1.11031e-14 1.11042e-14 1.11055e-14 1.11063e-14, highest: 96.982
98.676      99.1095     99.6358     99.8204
```

Hide

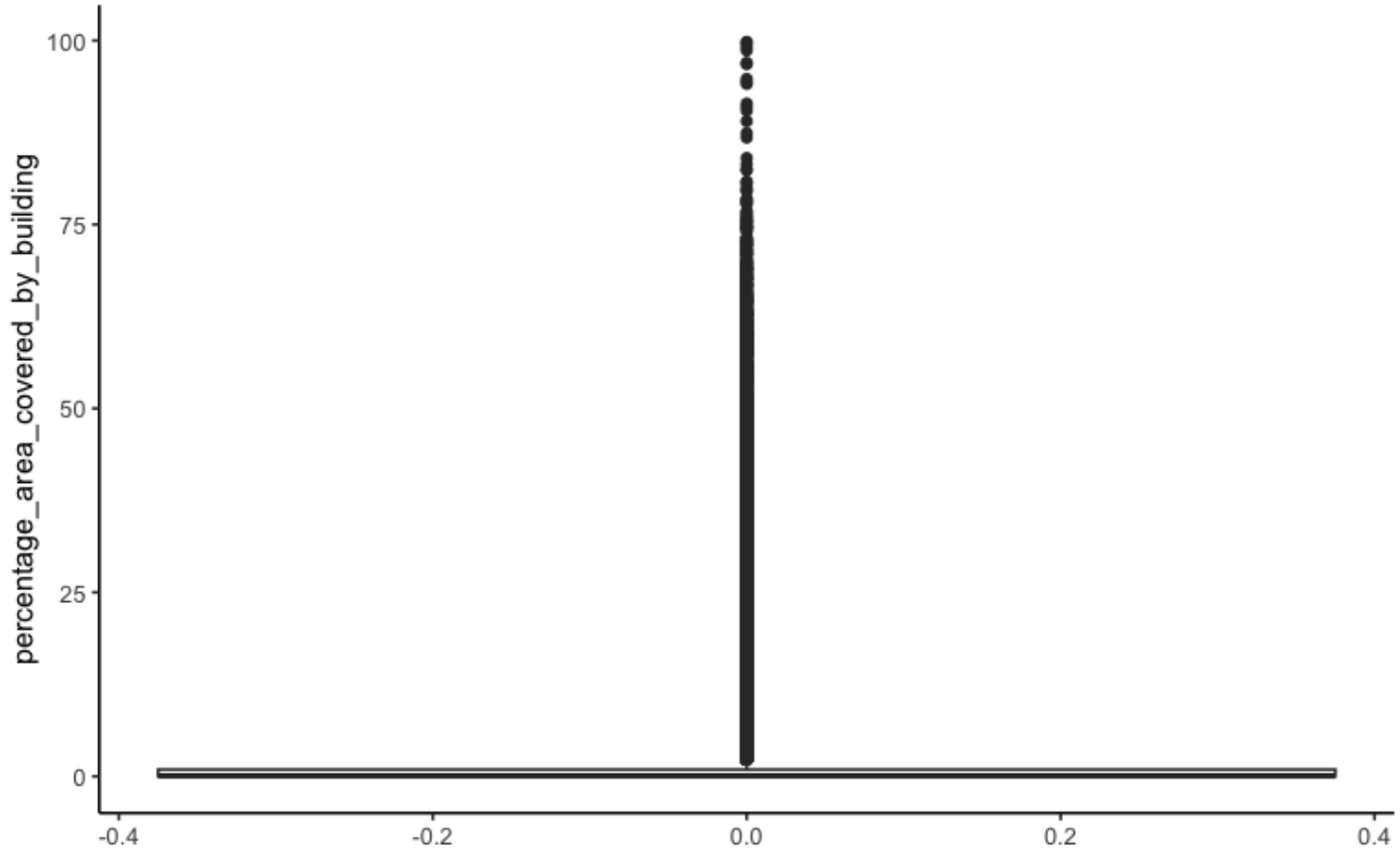
```
sd(regression$percentage_area_covered_by_building, na.rm=TRUE)
```

```
[1] 5.568738
```

Display box-plot of percentage_area_covered_by_building for all projects

Hide

```
p<-ggplot(regression, aes(y=percentage_area_covered_by_building)) +
  geom_boxplot()
p + theme_classic()
```



Describe percentage_area_covered_by_building by split category

Hide

```
regression %>% group_by(splits1) %>% summarise(projId = n())
```

splits1 <int>	projId <int>
0	269775
1	42033

2 rows

Hide

```
tapply(regression$percentage_area_covered_by_building, regression$splits1, summary)
```

```
$`0`
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max. 
 0.00000  0.00000  0.05387  1.43866  0.74144 99.82039 

$`1`
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
 0.0000  0.0000  0.3152  4.9264  5.7893 72.9727 
```

Display box-plot of percentage_area_covered_by_building by split category

Hide

```
regression$splits1=as.character(regression$splits1)
p<-ggplot(regression, aes(x=splits1, y=percentage_area_covered_by_building, color=splits
1)) +
  geom_boxplot()
p + theme_classic()
```