Abstract

In this paper are analyzed commit history logs from a git repository. The dataset is created using a python script which saves logs in a csv file.

1. **The Dataset creation**

This paper uses custom dataset created from commit logs of an open source repository AutoFixture [1]. This chapter contains general information about how input csv file was created.

## 1.1 Git log file

To get git logs history following command is used:

git log --numstat > AutoFixture.log

It saves log history in machine friendly format in AutoFixture.log file. An example of commit from AutoFixture.log file:

commit e64ce87243273e13faedb73ff2c1520ef0ed7b06

Author: Alex Povar <user.home.0000@gmail.com>

Date: Tue Aug 23 10:49:52 2016 +0300

Fix code style and add unit test

13 13 Src/AutoNSubstitute/NSubstituteVirtualMethodsCommand.cs

1 1 Src/AutoNSubstitute/SubstituteRequest.cs

100 65 Src/AutoNSubstituteUnitTest/AutoConfiguredFixtureIntegrationTest.cs

It contains commit id, Author full name and email, created date, commit comment and a list of modified files. First number before file name is number of added lines. Second number is number of removed lines. For example

100 65 Src/AutoNSubstituteUnitTest/AutoConfiguredFixtureIntegrationTest.cs

means that 100 new lines have been added and 65 have been removed from file.

**1.2 CSV file creation**

The log file obtained in previous step can’t be used directly in R. The file needs to be transformed to csv format. The data in Autofixture.log file is relational: one commit entry have many file changes. So, the relation between commit and changed file is **one to many**. The csv file will contain one entry for each changed file. In order to transform log file to csv a python script have been created. It can be found in src/python folder of repository [2]. The structure of csv file is following:

id, author, email, date, fileName, added, removed

Several lines of csv file:

id,author,email,date,fileName,added,removed

ab829640ed8e02776e4f4730d0e72ab3cc382339,Mark Seemann,mark@ploeh.dk, Mon Sep 5 15:21:45 2016 +0200,README.md,4,0

52da28891d60ede69d35df8d2040ef64890f17cd,Mark Seemann,mark@ploeh.dk, Wed Aug 31 18:21:21 2016 +0200,Src/AutoFakeItEasy/Properties/AssemblyInfo.cs,2,2

52da28891d60ede69d35df8d2040ef64890f17cd,Mark Seemann,mark@ploeh.dk, Wed Aug 31 18:21:21 2016 +0200,Src/AutoFakeItEasy2.UnitTest/Properties/AssemblyInfo.cs,2,2

Each line from csv file represents information about commit number, author, date and number of lines added and removed.

**2 Data analysis**

This chapter contains the data analysis itself.

## 2.1 Loading data and preparing environment

The csv file contains date-time variable which needs to be parsed. Before loading the csv file the locale is set to *us* and a custom data format is created to parse the date-time.

Sys.setlocale(locale='us')

setClass('git\_date')

setAs("character","git\_date", function(from) strptime(from, " %a %b %e %H:%M:%S %Y %z"))

Now we can load the dataset itself

dataset <- read.csv('commit\_log.csv', colClasses = c('factor', 'factor', 'factor', 'git\_date', 'factor', 'integer', 'integer'))

attach(dataset)

## 2.2 Descriptive statistics analysis

## 2.2.1 Data summary

To get a high overview of data summary command is run.

stats.summary <- summary(dataset)

The resulting summary is big so it was splatted in two parts shown which are shown in figure 2.1 and figure 2.2

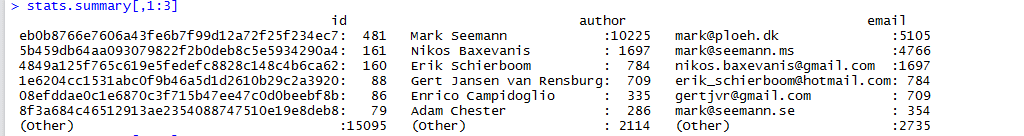


Figure 2.1 – First part of summary

Analyzing this figure 2.1 following conclusion affirmation can be made:

* The larges commit includes 481 files.
* User with name Mark Seemann modified 10225 files.
* User with email <mark@ploeh.gk> modified 5105 files which is much less than 10225. We can suppose that user Mark Seemann uses several emails to commit to github. It is possible because git users often works from different work stations which can be configured to use different user name and email.

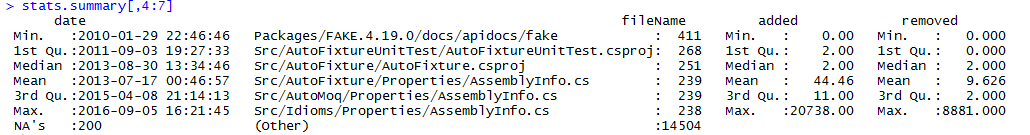


Figure 2.2 – Second part of summary

From second part of summary (Figure 2.2) following affirmation can be made:

* First commit was made on 2010-01-29
* The last commit was made on 2016-09-05
* 200 entries from csv file doesn’t contains commit date. It can be because of bug in R script or because the commit date is provided in wrong format. A quick manual analysis ( subset(dataset, is.na(date))) of csv data discovered that all unparsed dates has +1300 UTC offset, which is valid value.
* File Packages/FAKE.4.19.0/docs/apidocs/fake is included in 411 commits.
* The minimum number of added lines per file in commit is 0 which is obvious.
* The maximum number of added lines in one single commit for a file is 20738. Usually in git it means that a big file has been moved to another directory.
* 1st and 2nd quartile (median) are equals to 2. It means that in most cases only 2 new lines are added to a file. The mean is much bigger than 1st, median and 3rd quartiles – the distribution is ***positively skewed*.**
* The minimum number of removed lines per file in commit is 0 – obvious.
* The maximum number of removed lines in one single commit from a file is 8881.
* The distribution of removed lines if ***positively skewed*** as well, although it is not as much as number of added lines is (mean is closer to median).

Total number of authors is 61 and the total number of commits is 2985.

|  |
| --- |
| > length((unique(dataset$author)))  [1] 61  > length(unique(dataset$id))  [1] 2985 |

**2.2.2 Authors analysis**

To understand better the dataset several plots have been created using ggplot2 library.

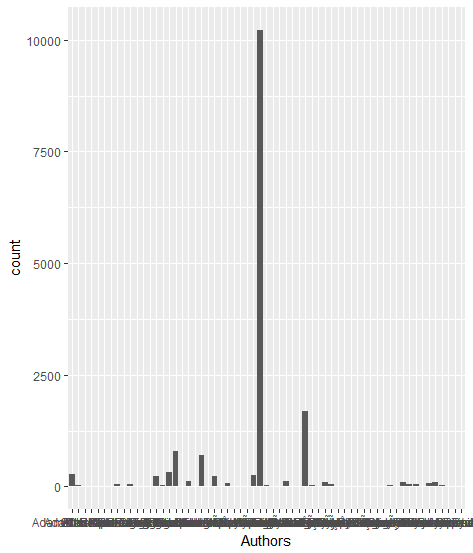


Figure 2.3 – All authors by number of changed files

Variable *author* is categorical variable. In Figure 2.3 is shown frequency distribution of authors. The X axis contains authors and Y axis contains the number of changed files. From figure we can see that there are several authors who made many changes. Source code for the plot is:

qplot(factor(author), data = dataset, geom = "bar", xlab = "Authors")

The X axis contains 61 authors so it’s difficult to see the authors name. Figure 2.4 contains same data on X and Y axis, but only top authors are shown.

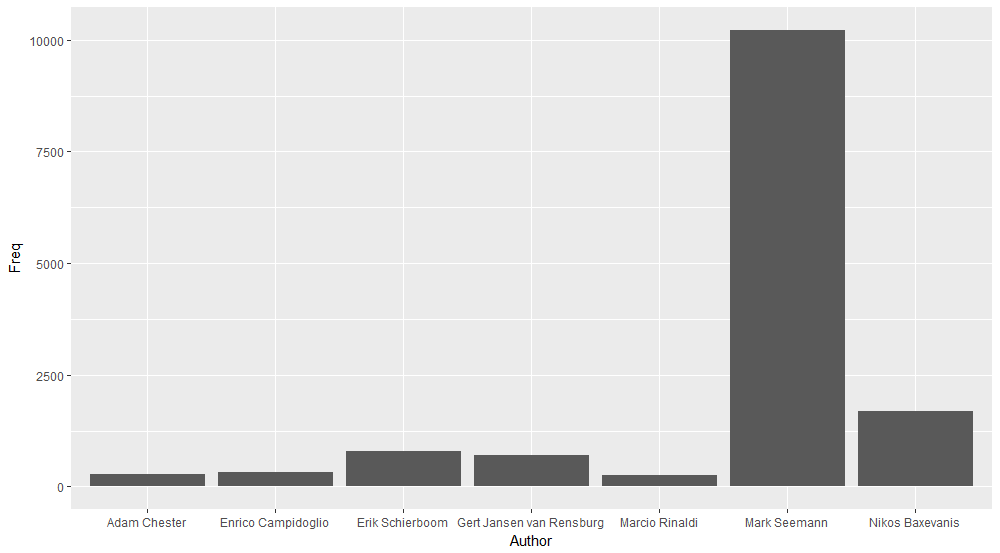


Figure 2.4 – Top contributors

To create plot following script is used:

stats.authors\_freq <- as.data.frame(table(dataset$author))

colnames(stats.authors\_freq) <- c("Author", "Freq")

stats.top\_authors <- subset(stats.authors\_freq, Freq >= mean(stats.authors\_freq$Freq))

ggplot(data = stats.top\_authors, aes(x=Author, y=Freq))+geom\_histogram(stat = "identity")

The script filters authors who changed more than mean (264) files. There are only 7 contributors who changed more than 264 files.

**2.2.3 Added and removed lines analysis**

Another two variables which presents interest are number of added and removed lines.

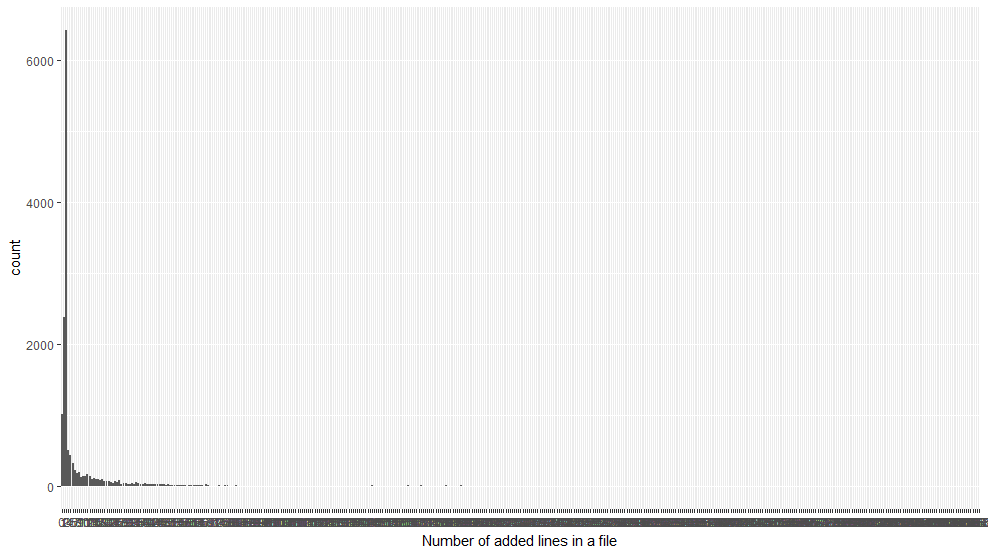


Figure 2.5 – Frequency distribution of Number of added lines

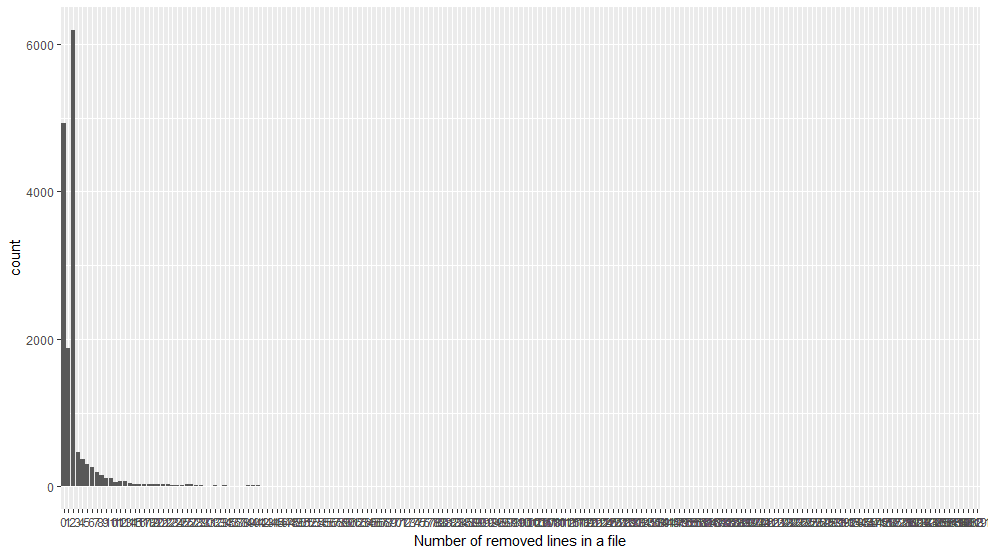


Figure 2.6 – Frequency distribution of Number of removed lines

Figures 2.5 and 2.6 confirms our assumption that the plots are positively skewed. Following scripts are used to create plots:

qplot(factor(added), data = dataset, geom = "bar", xlab = "Number of added lines in a file")

qplot(factor(removed), data = dataset, geom = "bar", xlab = "Number of removed lines in a file")

As in previous example with authors it can be useful to show plots with most frequent values. Figure 2.7 and 2.8 contains this plots. The source code of scripts is:

stats.authors\_freq <- as.data.frame(table(dataset$author))

colnames(stats.authors\_freq) <- c("Author", "Freq")

stats.top\_authors <- subset(stats.authors\_freq, Freq >= mean(stats.authors\_freq$Freq))

ggplot(data = stats.top\_authors, aes(x=Author, y=Freq)) + geom\_histogram(stat = "identity")

stats.added\_freq <- as.data.frame(table(dataset$added))

colnames(stats.added\_freq) <- c("Added", "Freq")

stats.top\_adds <- subset(stats.added\_freq, Freq > mean(dataset$added))

ggplot(data = stats.top\_adds, aes(x=Added, y=Freq)) + geom\_histogram(stat = "identity")

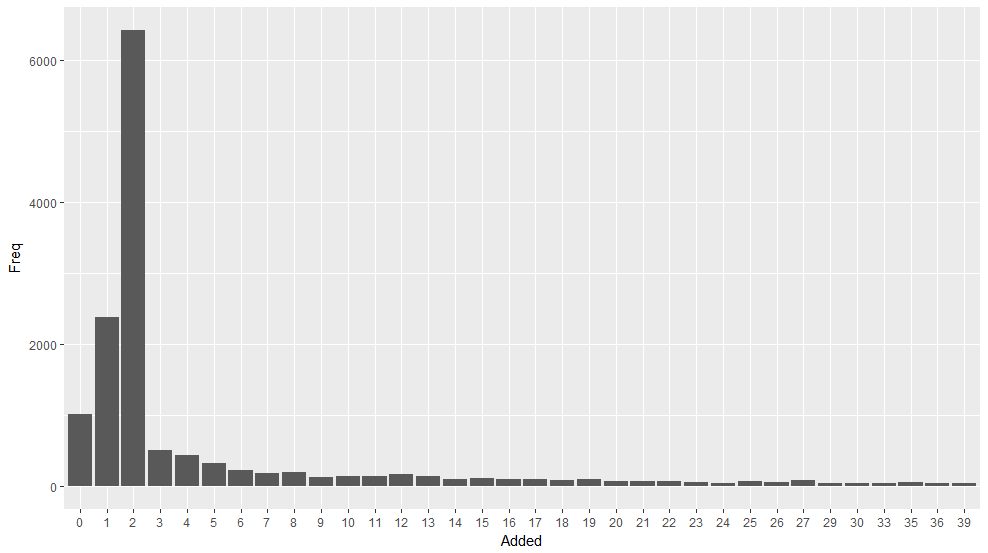


Figure 2.7 – Top frequency distribution of Number of added lines

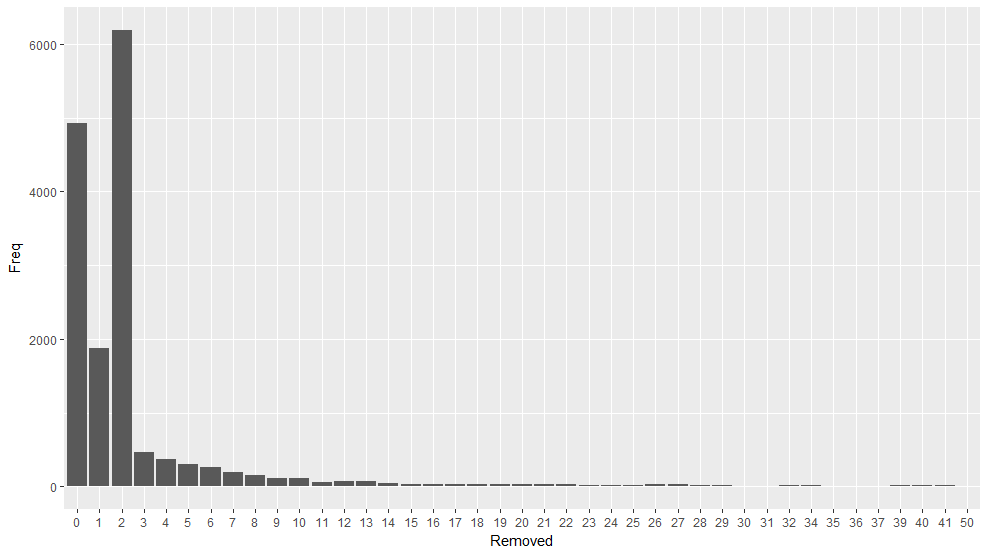


Figure 2.8 – Top frequency distribution of Number of removed lines

**2.2.4 Multivariate data analysis**

In this section, multivariate analysis is done to find relationships between variables.

Authors variable is a categorical variable, while commit date is a continues variable. The relation between two variables denotes contributor’s activity during repository life time. Figure 2.9 represents the contributor’s activity to the repository. The script is:

qplot(author, date, data=dataset, geom="boxplot", fill=author) + theme(axis.text.x = element\_text(angle = 90, hjust = 1))

Analyzing Figure 2.9 following affirmation can be made:

* There are 3 main active contributors – Enrico Campidoglio, Mark Seemann and Nikos Baxevanis.
* Mark Seemann – contributed since repository was created (he’s the repository owner). He still is contributing (because of max value). His contribution activity during 6 years was constant because median is placed in the middle between 1st and 3rd quartiles.
* Enrico Campidoglio – his activity is nearly the same as for Mark. The only difference is that he started contributing in middle of 2011 (the min value). Another small difference is that the median is little bit closer to 3rd quartile which means he’s more active now rather he was at the beginning.
* Nikos Baxevanis – started contributing in the same time as Enrico Campidoglio. He contributed actively very short time because median is very close to min value and 3rd quartile is close to min value as well. He contributed recently (max value) and it keeps him in the list of active contributors.
* The other contributors contributed only once or very short period

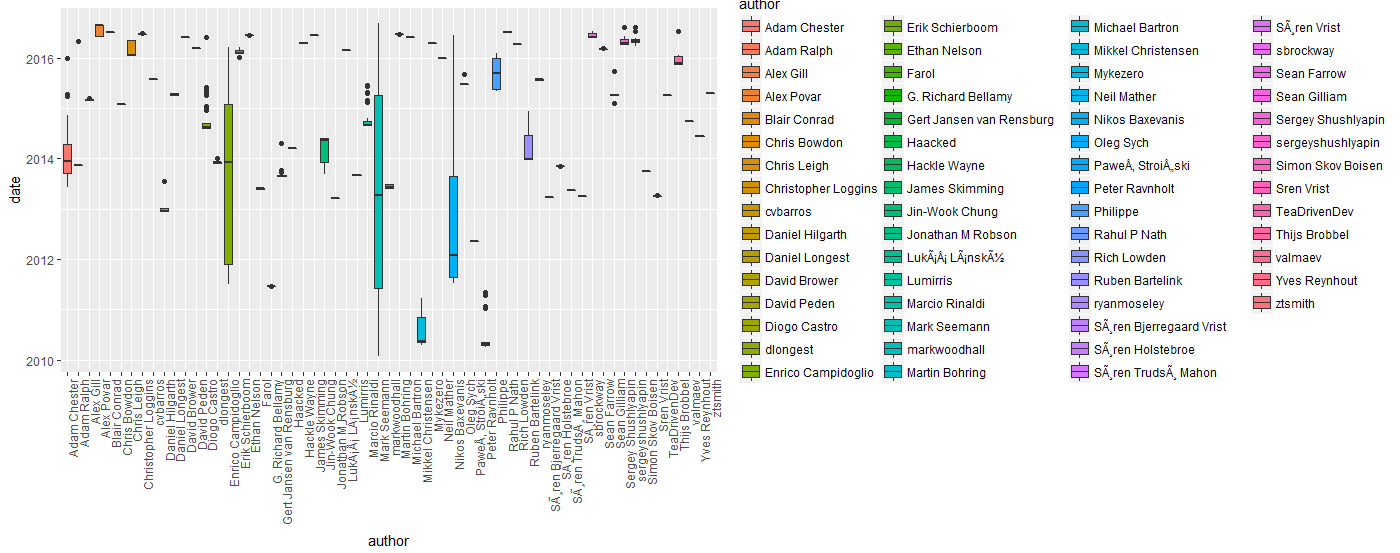


Figure 2.9 – Contributors activity during repository life

Another area of interest represents authors activity during 24 hours interval. In Figure 2.10 is shown this dependency. Minimum activity is from 3:00 AM up to 7:00 AM which means that most contributors don’t work in night hours. Most changes are made in afternoon. Evening hours since 21:00 to 23:00 have also big activity which means that many contributors works on it as a hobby.

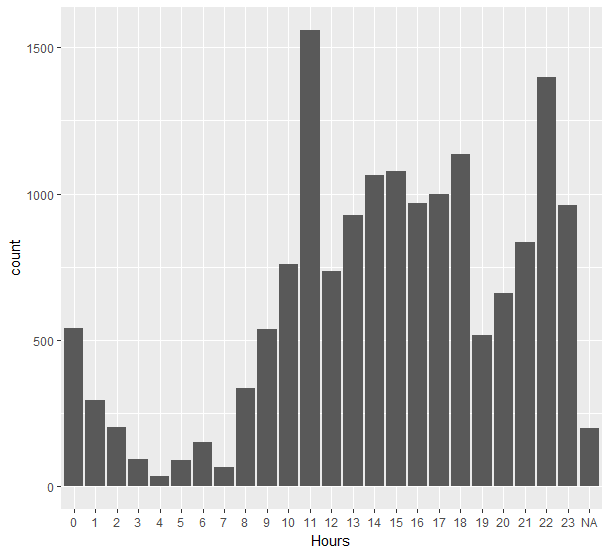


Figure 2.10 – File change activity during a day

The script for Figure 2.10 is:

qplot(factor(date$hour), data = dataset, geom = "bar", xlab = "Hours")

It is interesting to see if different authors prefer different hours to work. Figure 2.11 represents the relation between author and hours when he changed files in repository. Analyzing Figure 2.11 following conclusions can be made:

* Ryanmoseley – contributed all his changes only at night.
* Mark Seemann (the repository owner) – tends to work in evenings.

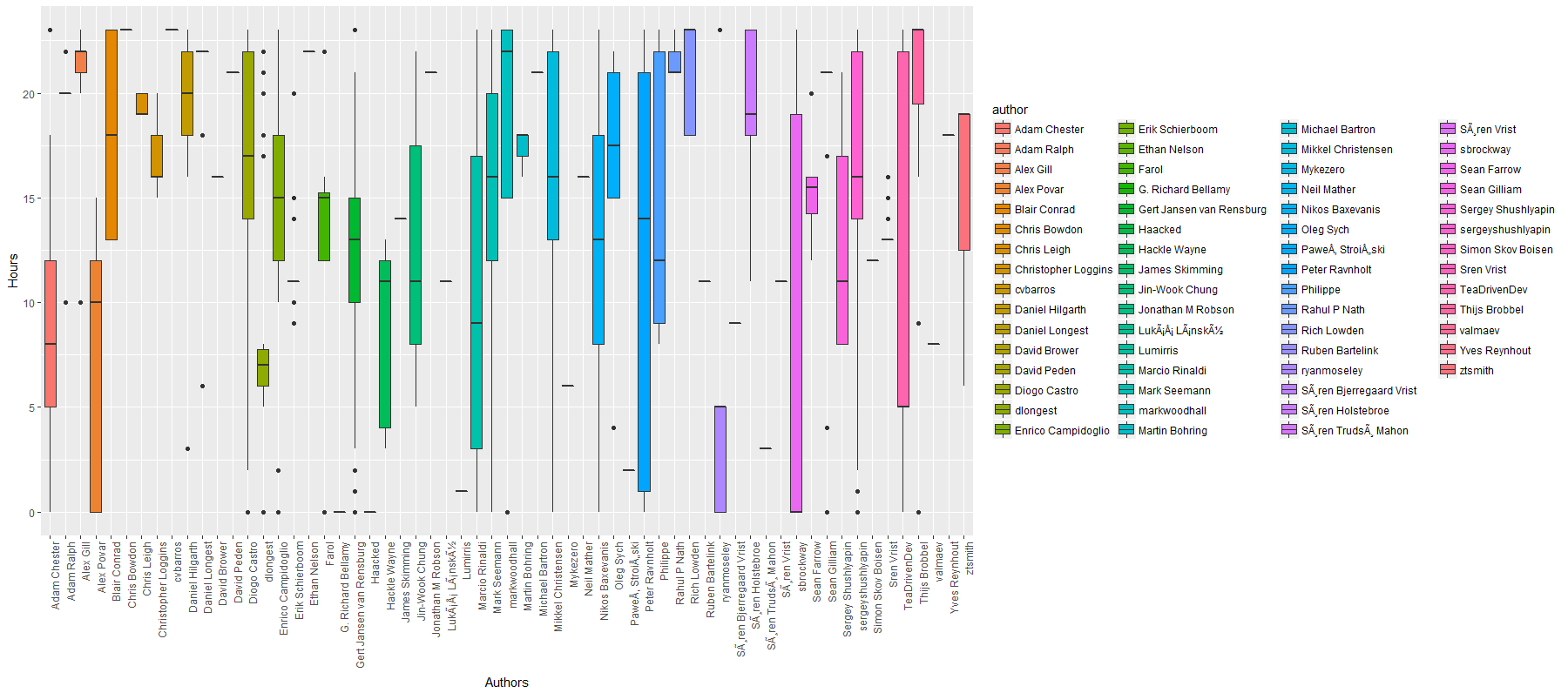


Figure 2.11 – Authors activity during 24 hours interval

The script for figure 2.11 is:

qplot(author, date$hour, data=dataset, geom="boxplot", fill=author, ylab = 'Hours', xlab = 'Authors') + theme(axis.text.x = element\_text(angle = 90, hjust = 1))

**2.4 Testing hypotheses**

In this section statistical analysis is done for different statistical hypotheses.

**2.4.1 Dependency between file change time and author**

Analyzing figure 2.11 following hypothesis can be made: “File change hours depends on author because different authors work in different hours”. It can be tested using NHST. More formal way of expressing it is:

* H0: “File change hours **doesn’t depend** on author (all authors work in same interval of hours)”
* H1: “File change hours **depends** on author (different authors work in different hours)”

Hours variable is a continue variable and it is not normally distributed (figure 2.10). In order to test the hypotheses Kruskal-Wallis test is done (figure 2.12).

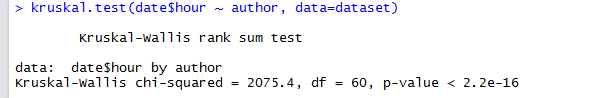


Figure 2.12 – Kruskal-Wallis tests results

The p-value of Kruskal-Wallis test is less than 0.05 which means that H0 hypothesis is rejected and H1 is accepted - File change hours depends on author.

**2.4.2 Authors who work in same hours**

Another hypothesis which can be made from figure 2.11 is that Mark Seemann and Mikkel Christensen work in same hours interval (their box-plots are very similar). If all other authors are ignored two means of this authors can be compared. Because hours variable is not normally distributed T-Student’s test can’t be applied. Instead Mann-Whitney U test can be applied. The hypotheses are:

* H0: Mark Seemann and Mikkel Christensen work in same hours interval.
* H1: Mark Seemann and Mikkel Christensen work in different interval of hours.

From results of test (figure 2.13) can be seen that p-value is greater than 0.05 so we fail to reject H0 and accept it - Mark Seemann and Mikkel Christensen work in same hours interval.

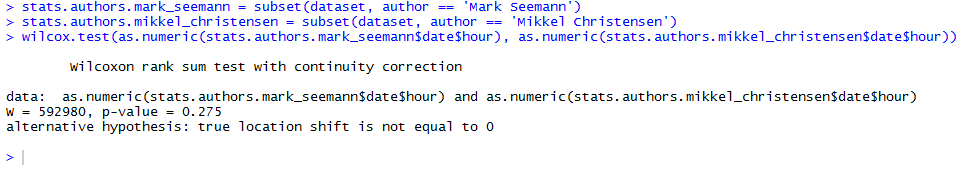


Figure 2.13 – Mann-Whitney U test results

**2.5 Predictive analysis**

Analyzed dataset doesn’t contains any labeled data and is difficult to apply supervised algorithms. However, we can try to create an artificial variable in order to apply supervised algorithms. We can try predict the author based on other variables. In order to simplify the task, we’ll try to predict if given file change belongs to Mark Seemann (the repository owner). The model will use ***date*** variable so all rows where value for date column is missing should be removed.

dataset.valid\_date = subset(dataset, !is.na(dataset$date))

The model uses day of week and hours so they should be added to data set as well.

dataset.valid\_date$hour = dataset.valid\_date$date$hour

dataset.valid\_date$wday = dataset.valid\_date$date$wday

The column ***is\_owner*** will contains Boolean (TRUE if the author is Mark Seemann, FALSE otherwise) value.

dataset.valid\_date$is\_owner = with(dataset.valid\_date, author == 'Mark Seemann')

In order to train the model training and testing sets are required. Following scripts splits data in two sets: 80% for training and 20% for testing.

ntrain <- round(nrow(dataset.valid\_date)\*4/5)

train <- sample(1:nrow(dataset.valid\_date), ntrain)

training <- dataset.valid\_date[train,]

testing <- dataset.valid\_date[-train,]

The model uses **Logistic Regression** [3] algorithm to predict the data. To avoid overfitting model variables should be selected carefully. In previous sections dependency between author and commit hours has been analyzed so it is a good candidate to be added to the model.

model <- glm(is\_owner~date$hour, data=training, family=binomial(logit))

Figure 2.14 contains the summary of obtained model. Analyzing the summary following conclusion can be made:

* The b1 coefficient value can be interpreted as “For every one hour increase in time the odds of having reached Mark Seemann increase by exp(0.09) = 1.094 times”.
* Both variables (b0­ and b1) are statistically significant because Pr(>|z|) is less than 0.05

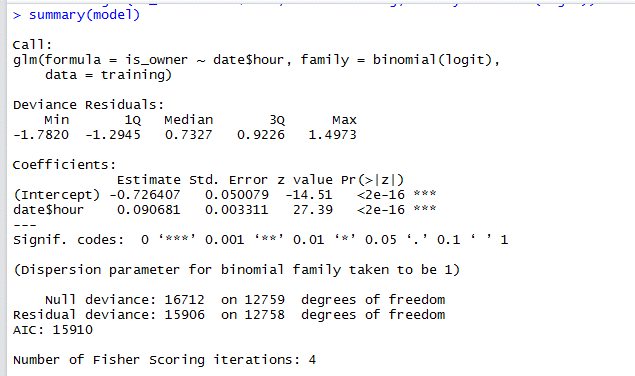


Figure 2.14 – model trained only on hours

Now the model can be tested on testing set. To reduce amount of code accuracy function has been created. It takes a vector of predicted and actual value and returns the accuracy.

accuracy <- function(predictions, answers){

sum((predictions==answers)/(length(answers)))

}

First we evaluate accuracy on training set:

> predictions <- round(predict(model, training, type="response"))

> predictions <- ifelse(predictions == 1, TRUE, FALSE)

> accuracy(predictions, training$is\_owner)

[1] 0.6920846

Next we can evaluate the model on testing set:

> predictions <- round(predict(model, testing, type="response"))

> predictions <- ifelse(predictions == 1, TRUE, FALSE)

> accuracy(predictions, testing$is\_owner)

[1] 0.684953

On training set the accuracy is nearly same which is good. It is good to compare it with probability of random reaching a change which belongs to Mark Seemann:

nrow(dataset.valid\_date[dataset.valid\_date$is\_owner == TRUE, ]) /nrow(dataset.valid\_date)

[1] 0.6410658

It means that if our model would always predict Mark Seemann then the accuracy would be 0.64. We can conclude that our model is performing little bit better than primitive model. The model accuracy can be increased if more variables are added to it. Another significant variable can be the year because Mark is active contributor and he has worked since the repository was created. The script is bellow:

> model <- glm(is\_owner~date$hour + date$year, data=training, family=binomial(logit))

> summary(model)

Call:

glm(formula = is\_owner ~ date$hour + date$year, family = binomial(logit),

data = training)

Deviance Residuals:

Min 1Q Median 3Q Max

-2.0657 -1.1239 0.7040 0.9075 1.7492

Coefficients:

Estimate Std. Error z value Pr(>|z|)

(Intercept) 21.305486 1.136385 18.75 <2e-16 \*\*\*

date$hour 0.092382 0.003353 27.55 <2e-16 \*\*\*

date$year -0.194752 0.010057 -19.36 <2e-16 \*\*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 16649 on 12759 degrees of freedom

Residual deviance: 15478 on 12757 degrees of freedom

AIC: 15484

Number of Fisher Scoring iterations: 4

> predictions <- round(predict(model, training, type="response"))

> predictions <- ifelse(predictions == 1, TRUE, FALSE)

> accuracy(predictions, training$is\_owner)

[1] 0.6905956

>

> predictions <- round(predict(model, testing, type="response"))

> predictions <- ifelse(predictions == 1, TRUE, FALSE)

> accuracy(predictions, testing$is\_owner)

[1] 0.6796238

From summary, we can see that year has statistical significance. The prediction accuracy has increased on training set, and decreased on training set which is sign of overfitting. We can add more variables to see how the model is performing.

model <- glm(is\_owner~date$hour+date$wday+date$year+added+removed, data=training, family=binomial(logit))

Warning message:

glm.fit: fitted probabilities numerically 0 or 1 occurred

> summary(model)

Call:

glm(formula = is\_owner ~ date$hour + date$wday + date$year +

added + removed, family = binomial(logit), data = training)

Deviance Residuals:

Min 1Q Median 3Q Max

-2.1709 -1.1293 0.6780 0.8865 8.4904

Coefficients:

Estimate Std. Error z value Pr(>|z|)

(Intercept) 20.6058868 1.1610187 17.748 <2e-16 \*\*\*

date$hour 0.0909512 0.0033914 26.818 <2e-16 \*\*\*

date$wday 0.0822986 0.0099792 8.247 <2e-16 \*\*\*

date$year -0.1893624 0.0102879 -18.406 <2e-16 \*\*\*

added -0.0068913 0.0004648 -14.827 <2e-16 \*\*\*

removed -0.0009885 0.0004625 -2.137 0.0326 \*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 16649 on 12759 degrees of freedom

Residual deviance: 15036 on 12754 degrees of freedom

AIC: 15048

Number of Fisher Scoring iterations: 8

> predictions <- round(predict(model, training, type="response"))

> predictions <- ifelse(predictions == 1, TRUE, FALSE)

> accuracy(predictions, training$is\_owner)

[1] 0.7121473

>

> predictions <- round(predict(model, testing, type="response"))

> predictions <- ifelse(predictions == 1, TRUE, FALSE)

> accuracy(predictions, testing$is\_owner)

[1] 0.7050157

All variables have statistical significance however *removed*’s Pr(>|z|) is much higher than other. The accuracy has increased for both training and testing sets. The results are not very good but at least the model performs better that primitive model.

**2.6 Clustering**

Because of nature of dataset it’s difficult to apply supervised algorithms to it. Instead unsupervised algorithms can be applied to identify some common patterns. K-Means [4] algorithm can be applied to working hours to split 24 hours in several intervals. In previous section we’ve seen that activity is different during the day, night and evening. Reasonable split is 4 intervals: Morning, Afternoon, Evening and Night.

> km = kmeans(dataset.valid\_date[,c("hour")],centers = 4, nstart = 20)

> km$centers

[,1]

1 1.76258

2 10.89575

3 16.01106

4 21.37243

> plot(dataset.valid\_date$hour, col = km$cluster)

> points(km$centers, cex = 1.5, pch = 11, col = c(1:4))

The algorithm divided 24 intervals in 4 intervals: night (~01:45 AM), morning (~10:50 AM), afternoon (~4:00 PM) and evening (~09:20 PM). For better illustration results are presented in figure 2.15.

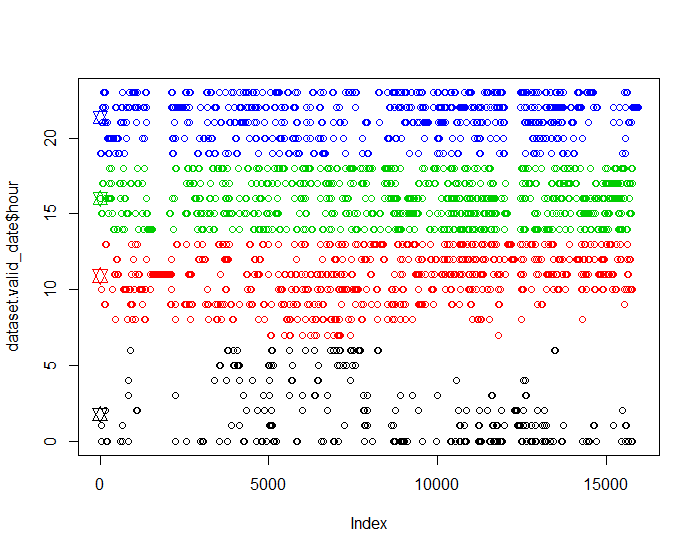


Figure 2.15 – 24 hours clustered in 4 groups

The largest interval is night interval however there is no big difference between the intervals’ size.

**Conclusion**

Using descriptive statistical analysis we identified top contributors of repository (it’s owner – Mark Seemann), their activity during repository life time, and when they work (many of them work in evenings, after work). Areas where Linear Regression can be applied haven’t been identified. An artificial attempt was made to apply Logistic Regression. K-NN could be applied as well on this to compare results, but because the dataset is not very good for this purpose and because of insufficiency of time it was omitted. Clustering algorithm (K-Means) performed well on splitting the day in 4 activity intervals.

Most analysis done in this paper is related to authors activity. Future analysis can be done on data related to files: number of files by extension, number of lines in different file types, number of authors which modified a file, etc. Another area of interest would be commit analysis: number of commits per author, average commit size, etc.

# References

|  |  |
| --- | --- |
| [1] | "AutoFixture," [Online]. Available: https://github.com/AutoFixture/AutoFixture. |
| [2] | S. Grajdean, "GitLogAnalysis\_RLang," [Online]. Available: https://github.com/grajdeanserghei/GitLogAnalysis\_RLang. |
| [3] | A. Fischetti and T. Fischetti, Data Analysis with R, Packt Publishing, 2015. |
| [4] | J. Ledolter, Data Mining and Business Analytics with R, Iowa City: John Wiley & Sons, 2013. |

**Appendix A**

**Source code of R scripts used in paper**