Forcasting issues

Forcast Padawan 2

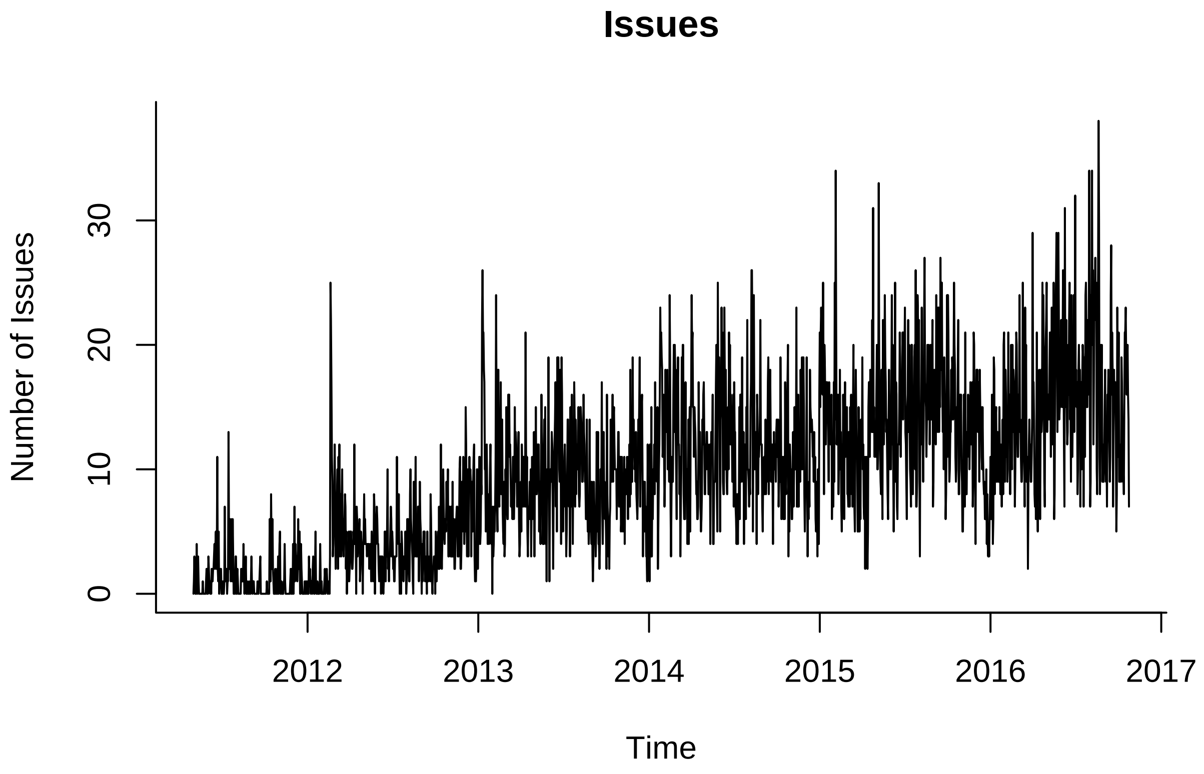
November 17, 2016

The goal of this experiment is to design the best model to forcaste the number of issue in the per day in the comming two weeks. We think that sthis could help Open Source organisation to manage there human ressources.

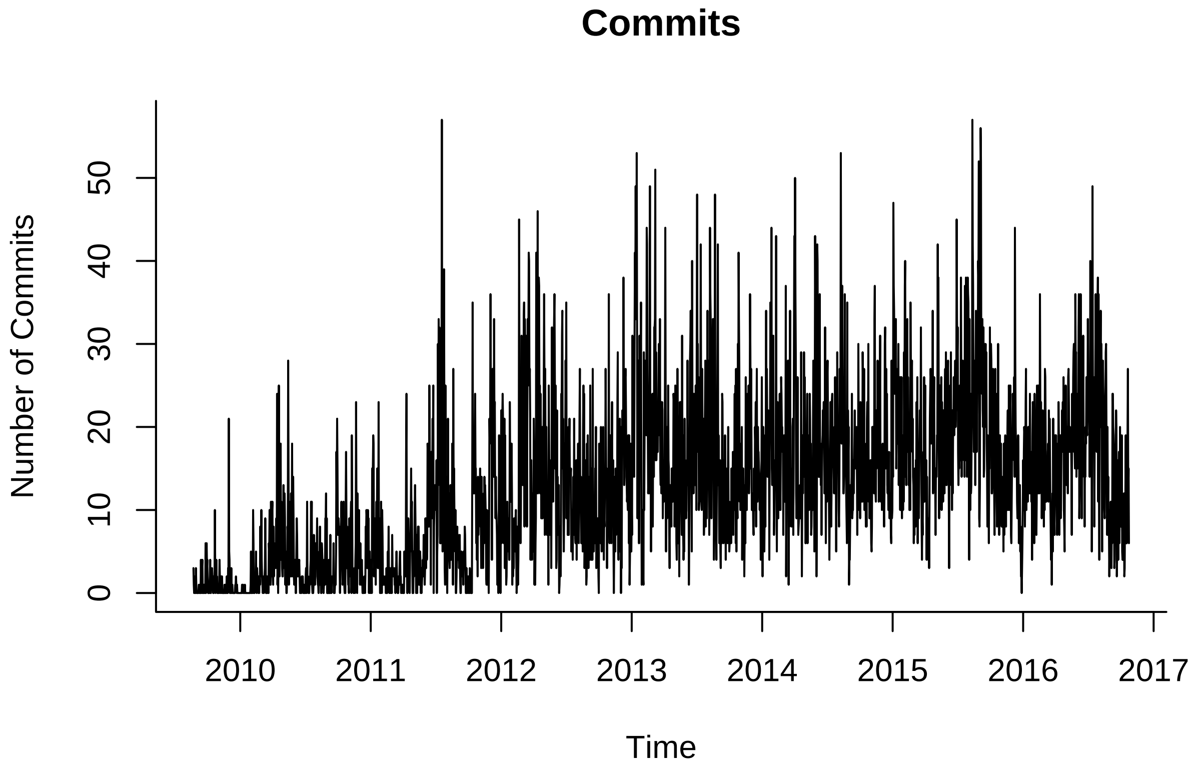
Load the data

```r
#install.packages('forecast')
library('forecast')
#load the data frame
issues.csv <- read.csv("issues/julialang_julia.csv")
commits.csv <- read.csv("commits/julialang_julia.csv")

issues.csv$date = as.Date(issues.csv$date,format='%m/%d/%Y')
commits.csv$date = as.Date(commits.csv$date,format='%m/%d/%Y')
```

```r
#loading issues and commits into a ts object
issues.ts <- ts(issues.csv$number_of_issues,
                start=c(as.numeric(format(issues.csv$date[[1]],'%Y')),
                        as.numeric(format(issues.csv$date[[1]],'%j'))),
                end=c(as.numeric(format(issues.csv$date[[length(issues.csv$date)]],'%Y')),
                      as.numeric(format(issues.csv$date[[length(issues.csv$date)]],'%j'))),
                freq=365)

commits.ts <- ts(commits.csv$number_of_commits,
                 start=c(as.numeric(format(commits.csv$date[[1]],'%Y')),
                         as.numeric(format(commits.csv$date[[1]],'%j'))),
                 end=c(as.numeric(format(commits.csv$date[[length(commits.csv$date)]],'%Y')),
                      as.numeric(format(commits.csv$date[[length(commits.csv$date)]],'%j'))),
                 freq=365)
plot(issues.ts, main = 'Issues', bty = 'l', ylab = 'Number of Issues')
```



```r
plot(commits.ts, main = 'Commits', bty = 'l', ylab = 'Number of Commits')
```



```r
time <- time(issues.ts)
```

```
n.valid <- 21
n.train <- length(issues.ts) - n.valid

train.issues.ts <- window(issues.ts, start=time[1], end=time[n.train])
valid.issues.ts <- window(issues.ts,
                   start=time[n.train+1],
                   end=time[n.train+n.valid])
```

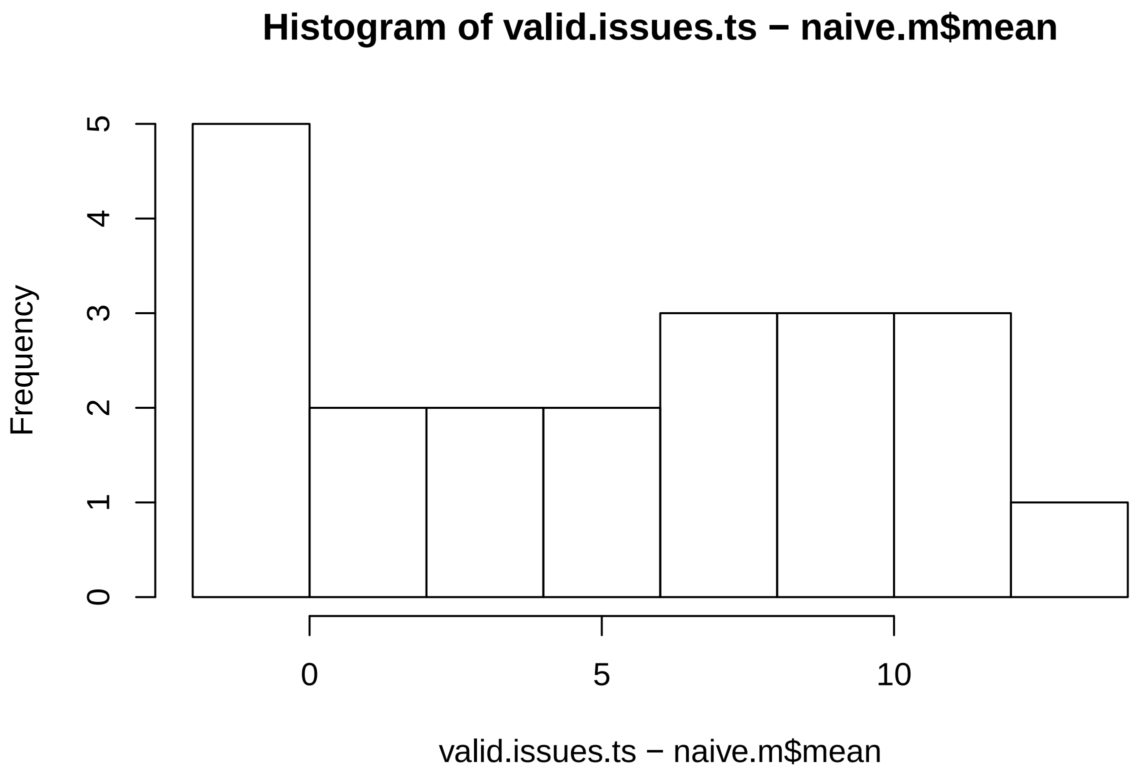
Naive Forecast

Naive

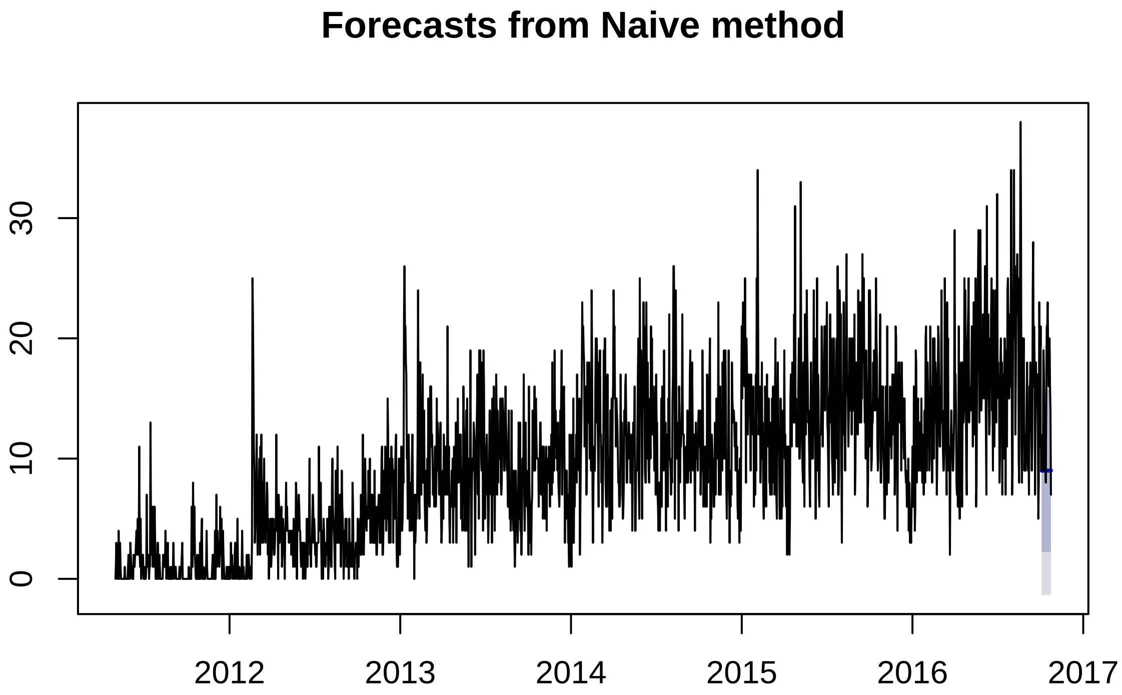
```
naive.m <- naive(train.issues.ts, h=n.valid)
accuracy(naive.m, valid.issues.ts)
```

```
##                       ME     RMSE      MAE      MPE     MAPE      MASE
## Training set 0.004547751 5.283810 3.889338     -Inf      Inf 0.7245681
## Test set     5.571428571 7.358183 5.857143 30.13343 34.04499 1.0911623
##                    ACF1 Theil's U
## Training set -0.3612623        NA
## Test set      0.5502692  1.291006
```

```
hist(valid.issues.ts - naive.m$mean)
```



```r
plot(naive.m)
lines(valid.issues.ts)
```

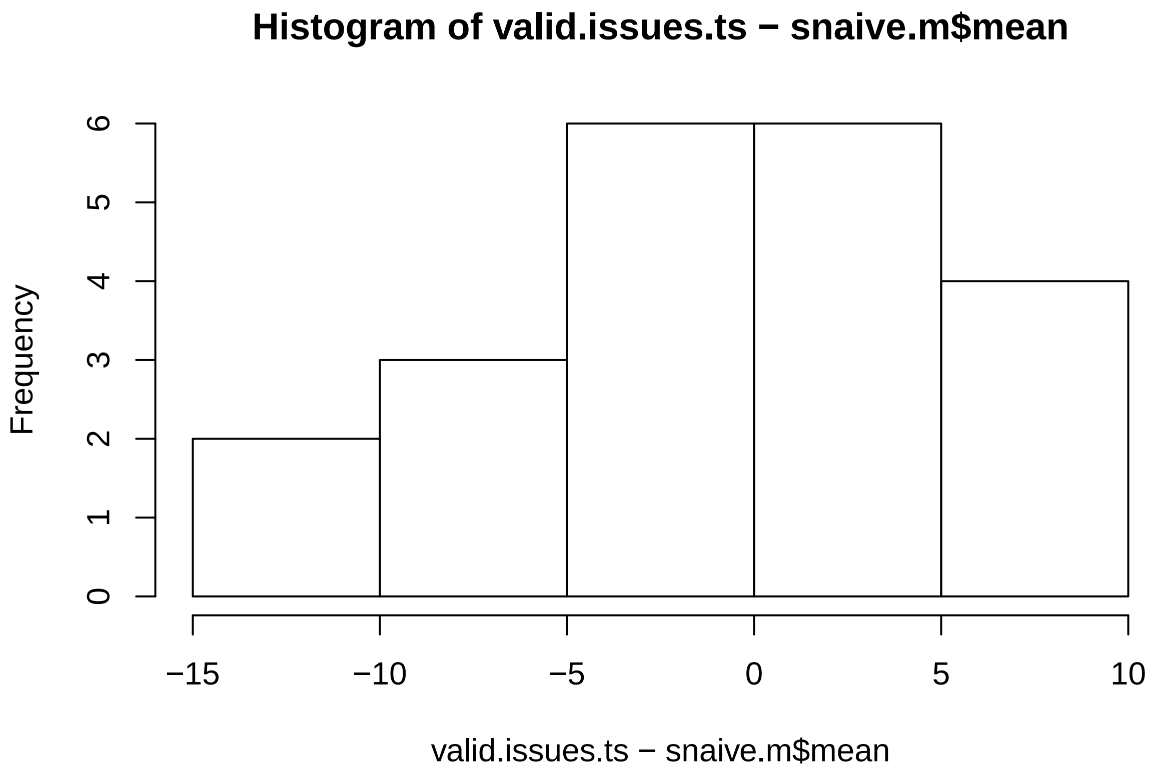


Seasonal Naive

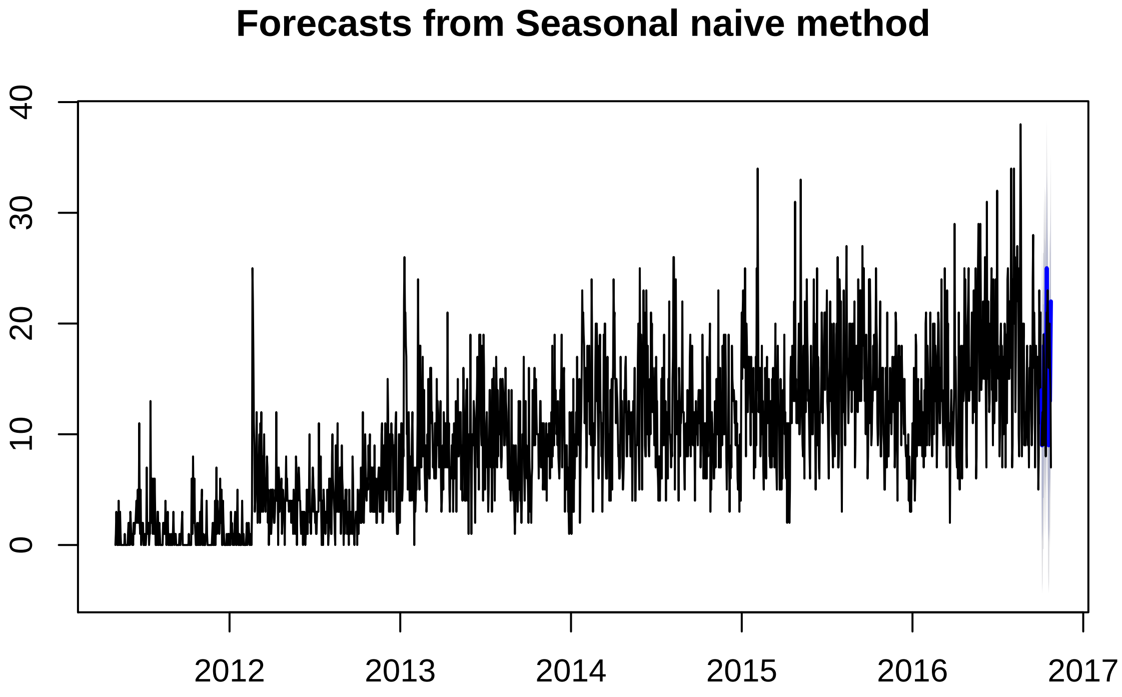
```r
snaive.m <- snaive(train.issues.ts, h=n.valid)
accuracy(snaive.m, valid.issues.ts)
```

```
##                      ME     RMSE      MAE       MPE     MAPE      MASE
## Training set  2.8229102 6.821247 5.367802      -Inf      Inf 1.0000000
## Test set     -0.7142857 6.145072 5.190476 -19.21842 44.64491 0.9669649
##                   ACF1 Theil's U
## Training set 0.3039580        NA
## Test set     0.5050089  1.259666
```

```r
hist(valid.issues.ts - snaive.m$mean)
```



```r
plot(snaive.m)
lines(valid.issues.ts)
```


Forecasts from Seasonal naive method