E-M classification

Zongyan Wang

March 28, 2016

Read data and packages

```r
# Upload packages for data clean and analysis
require(plyr)  #data clean
```

```
## Loading required package: plyr
```

```r
require(dplyr) #data clean
```

```
## Loading required package: dplyr
```

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

```
## The following objects are masked from 'package:plyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize
```

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

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

```r
require(tidyr) #data clean
```

```
## Loading required package: tidyr
```

```r
require(ggplot2)  # visualzation
```

```
## Loading required package: ggplot2
```

```r
require(XML)   # web scriping
```

```
## Loading required package: XML
```

```r
require(testthat) #test model
```

```
## Loading required package: testthat
```

```r
require(kernlab) # kernel and SVM
```

```
## Loading required package: kernlab
```

```
##
## Attaching package: 'kernlab'
```

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

```r
require(reshape2) # data clean
```

```
## Loading required package: reshape2
```

```r
require(datasets) # State data
require(caret) # Cross - Validation k fold
```

```
## Loading required package: caret
```

```
## Loading required package: lattice
```

```r
require(RCurl) # load the website link
```

```
## Loading required package: RCurl
```

```
## Loading required package: bitops
```

```
##
## Attaching package: 'RCurl'
```

```
## The following object is masked from 'package:tidyr':
##
##     complete
```

```r
require(maps) # heatmap: map_data
```

```
## Loading required package: maps
```

```
##
##  # maps v3.1: updated 'world': all lakes moved to separate new #
##  # 'lakes' database. Type '?world' or 'news(package="maps")'.  #
```

```
##
## Attaching package: 'maps'

## The following object is masked from 'package:plyr':
##
##     ozone
```

```r
require(ggmap) # for heatmap
```

```
## Loading required package: ggmap
```

```r
require(gridExtra)
```

```
## Loading required package: gridExtra

##
## Attaching package: 'gridExtra'

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

Read file

```r
# Read Data and function files
LC <- read.csv(file.choose(), header = T) #Please read LC_biz_all.csv
source("clean_f2.R") # functions for cleaning the data
source("plot_f2.R") # functions for plot(heatmap)
source("analysis_f2.R") # functions for plot and data analysis
```

Prepare for E-M algorithm

```r
# Data for 36 months term
lc <- perToN(LC)  # transfer percentage to numeric
lc <- replaceBlank_all(lc) # replace the blank
# subset to 36 months
lc36 <- lc %>% subset(term == "_36_months") %>% droplevels()
# Transform earliest_cr_line, issue_d, last_pymnt_d from str to numeric
lc36 <- lc36 %>% dateToNum(., "earliest_cr_line") %>%
  dateToNum(., "issue_d") %>% dateToNum(., "last_pymnt_d")
lc36 <- lc36 %>% mutate(y1 = last_pymnt_d - issue_d, y2 = !(loan_status == "Fully_Paid")) %>% select(-c
# Select the important variables with ANOVA
lc1 <- lc36[, !(colnames(lc36) %in% c(names(n.factor_all(lc36))))]
lc1$loan_status <- lc36$loan_status
anova.p <- data.frame(var = names(anova(lc1)), p.value = anova(lc1))
sig.names <- as.character((anova.p %>% subset(p.value < 0.01))$var)
var.names <- c(sig.names, names(n.factor_all(lc36)))
```

```
lc36 <- lc36[,c(var.names)] %>% select(-emp_title,-last_pymnt_d,-issue_d,
                                       -next_pymnt_d,-last_pymnt_amnt, -last_credit_pull_d, -zip_code)
# Set the loan_status "Charged_Off" with a large y value(100)
lc36$y1[lc36$loan_status == "Charged_Off"] = 61
lc36 <- lc36 %>% select(-loan_status)
lc.categorical <- lc36[,names(n.factor_all(lc36))]
lc.numeric <- lc36[,!names(lc36) %in% names(n.factor_all(lc36))]
# Transform the categorical column to multiple numeric columns
categorical.list <- apply(lc.categorical, 2, function(x) model.matrix(~ x + 0))
lc36 <- as.data.frame(cbind(categorical.list, lc.numeric))

# delete all the na
hasNA_all(lc36)
```

```
## revol_util         y1
##          3          1
```

```
lc36 <- lc36[!is.na(lc36$revol_util),]
lc36 <- lc36[!is.na(lc36$y1),] # One data missing y1
```

Use the EM algorithm to classify current and potential delinquency events(linear regression)

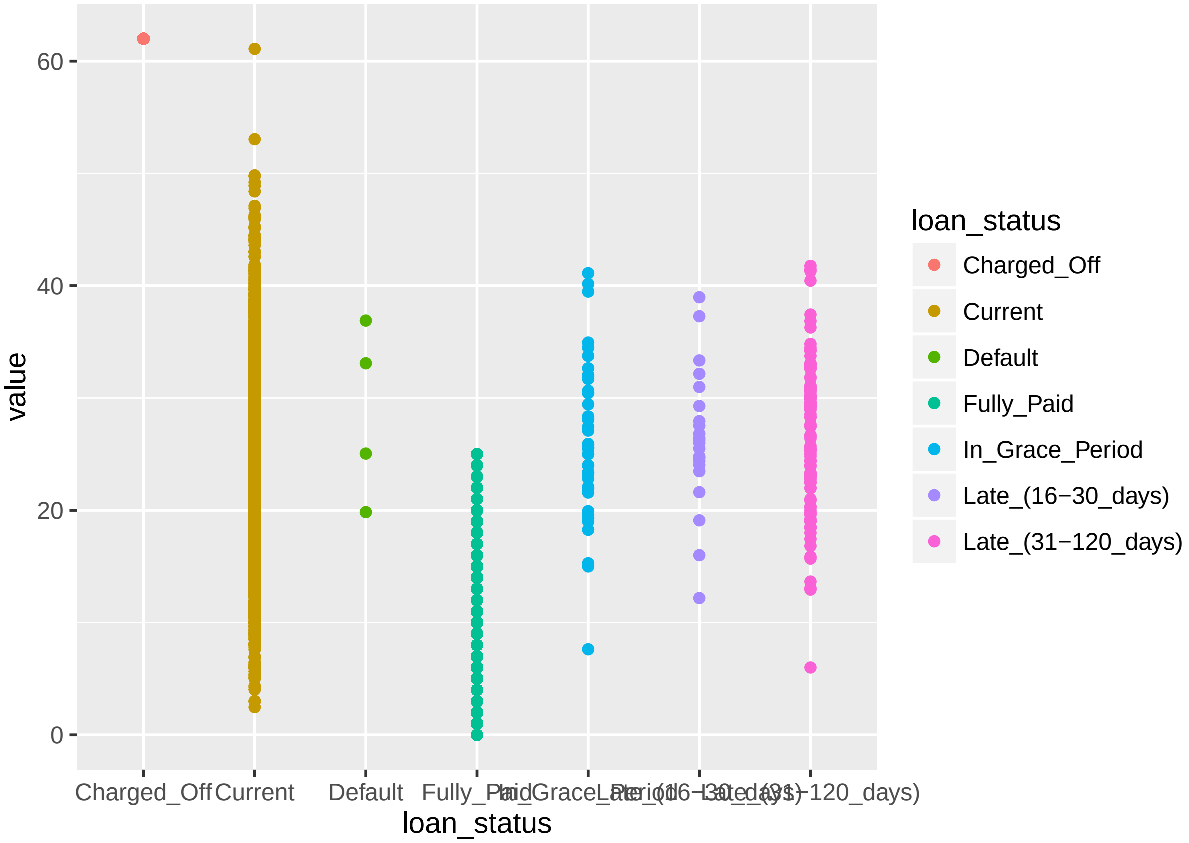
```
lc_em1 <- lc36
for (i in 1:10){
  k <- E_step(lc_em1)
  lc_em1 <- M_step(k,lc_em1)
}
```

```
## Warning in predict.lm(k, newdata): prediction from a rank-deficient fit may
## be misleading

## Warning in predict.lm(k, newdata): prediction from a rank-deficient fit may
## be misleading

## Warning in predict.lm(k, newdata): prediction from a rank-deficient fit may
## be misleading

## Warning in predict.lm(k, newdata): prediction from a rank-deficient fit may
## be misleading

## Warning in predict.lm(k, newdata): prediction from a rank-deficient fit may
## be misleading

## Warning in predict.lm(k, newdata): prediction from a rank-deficient fit may
## be misleading

## Warning in predict.lm(k, newdata): prediction from a rank-deficient fit may
## be misleading
## Warning in predict.lm(k, newdata): prediction from a rank-deficient fit may
## be misleading

## Warning in predict.lm(k, newdata): prediction from a rank-deficient fit may
## be misleading

## Warning in predict.lm(k, newdata): prediction from a rank-deficient fit may
## be misleading
```

```
# performance
true <- lc %>% subset(id %in% lc_em1$id) %>% select(loan_status)
compare.data <- data.frame(y1 = lc_em1$y1, y2 = lc_em1$y2, true = true) %>%
  mutate(y = y1 + y2)
plotdata <- compare.data %>%  select(-y1,-y2) %>%
  melt()
```

```
## Using loan_status as id variables
```

```
totallm_plot <- plotdata %>% ggplot() +
  geom_point(aes(x = loan_status, y = value, color = loan_status))
print(totallm_plot)
```

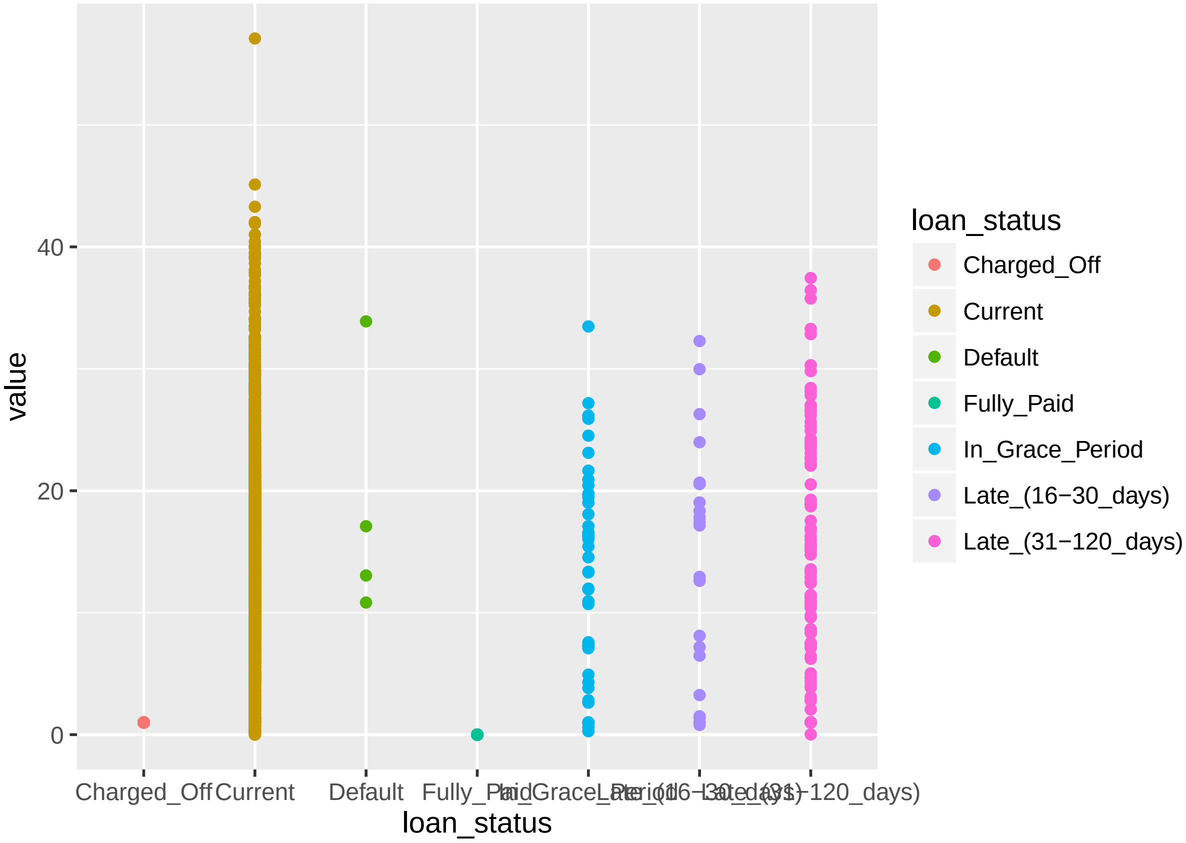


```
ggsave(filename = "totallm.png", plot = totallm_plot, path = ".",
       width = 10, height = 6, dpi = 400)
```

```
plotdata <- compare.data %>%  select(-y1,-y) %>%
  melt()
```

```
## Using loan_status as id variables
```

```
censorlm_plot <- plotdata %>% ggplot() +
  geom_point(aes(x = loan_status, y = value, color = loan_status))
print(censorlm_plot)
```



```
ggsave(filename = "censorlm.png", plot = censorlm_plot, path = ".",
       width = 10, height = 6, dpi = 400)
```

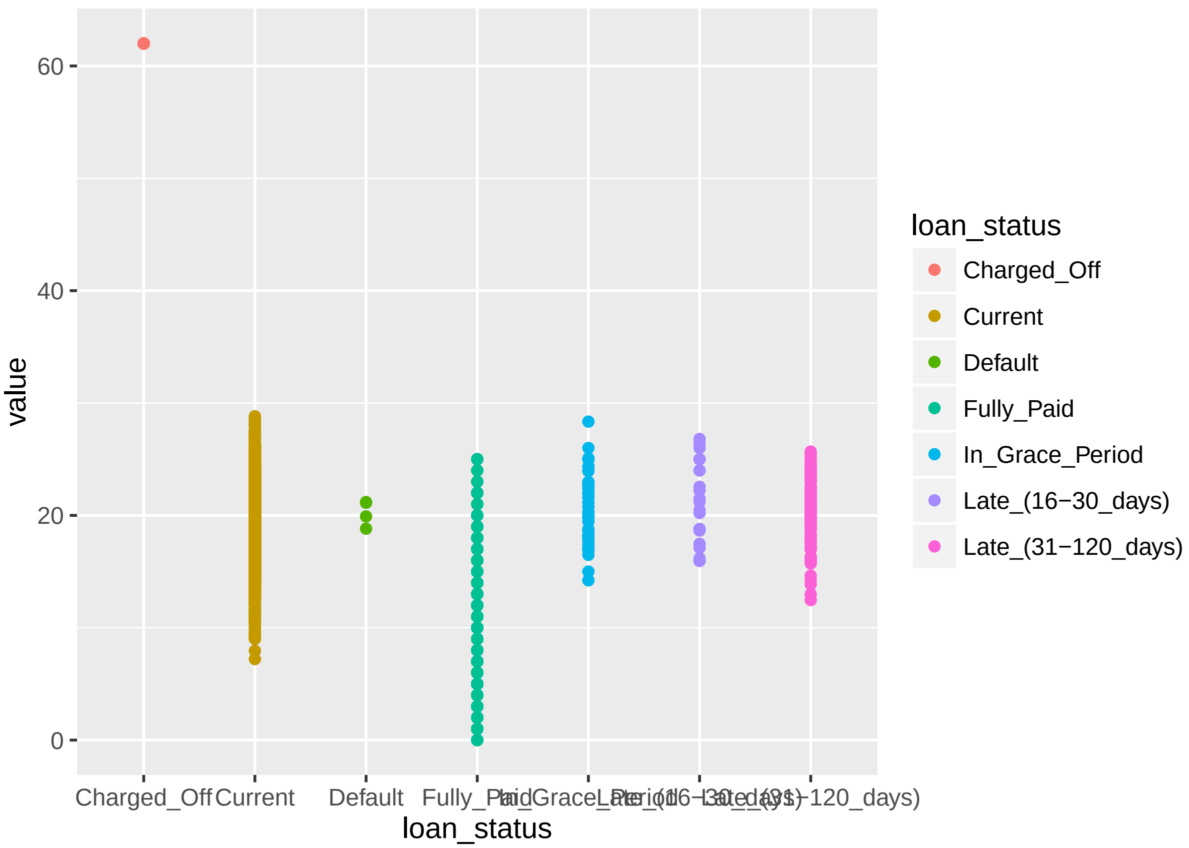
Use the EM algorithm to classify current and potential delinquency events(kernel SVM)

```
lc_em2 <- lc36
for (i in 1:10){
  k <- E_step_kern(lc_em2, kernel = "rbfdot")
  lc_em2 <- M_step_kern(k,lc_em2)
}
# performance
```

```
true <- lc %>% subset(id %in% lc_em2$id) %>% select(loan_status)
compare.data <- data.frame(y1 = lc_em2$y1, y2 = lc_em2$y2, true = true) %>%
  mutate(y = y1 + y2)
plotdata <- compare.data %>%  select(-y1,-y2) %>%
  melt()
```

```
## Using loan_status as id variables
```

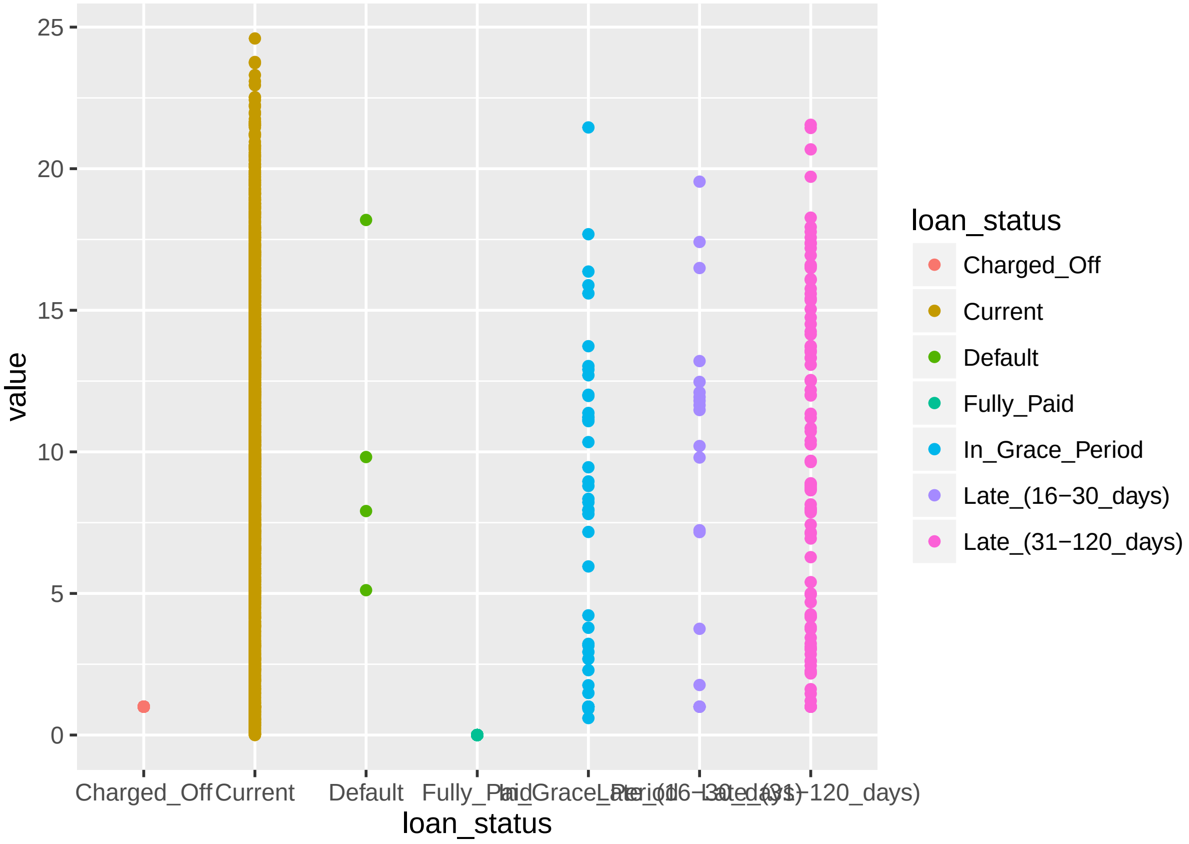
```
totalsvm_plot <- plotdata %>% ggplot() +
  geom_point(aes(x = loan_status, y = value, color = loan_status))
print(totalsvm_plot)
```



```
ggsave(filename = "totalsvm.png", plot = totalsvm_plot, path = ".",
       width = 10, height = 6, dpi = 400)

plotdata <- compare.data %>%  select(-y1,-y) %>%
  melt()
```

```
## Using loan_status as id variables
```

```
censorsvm_plot <- plotdata %>% ggplot() +
  geom_point(aes(x = loan_status, y = value, color = loan_status))
print(censorsvm_plot)
```



```
ggsave(filename = "censorsvm.png", plot = censorsvm_plot, path = ".",
       width = 10, height = 6, dpi = 400)
```