Classification and Decision Tree with RPART and RandomForest

Sang Kim

May 1, 2019

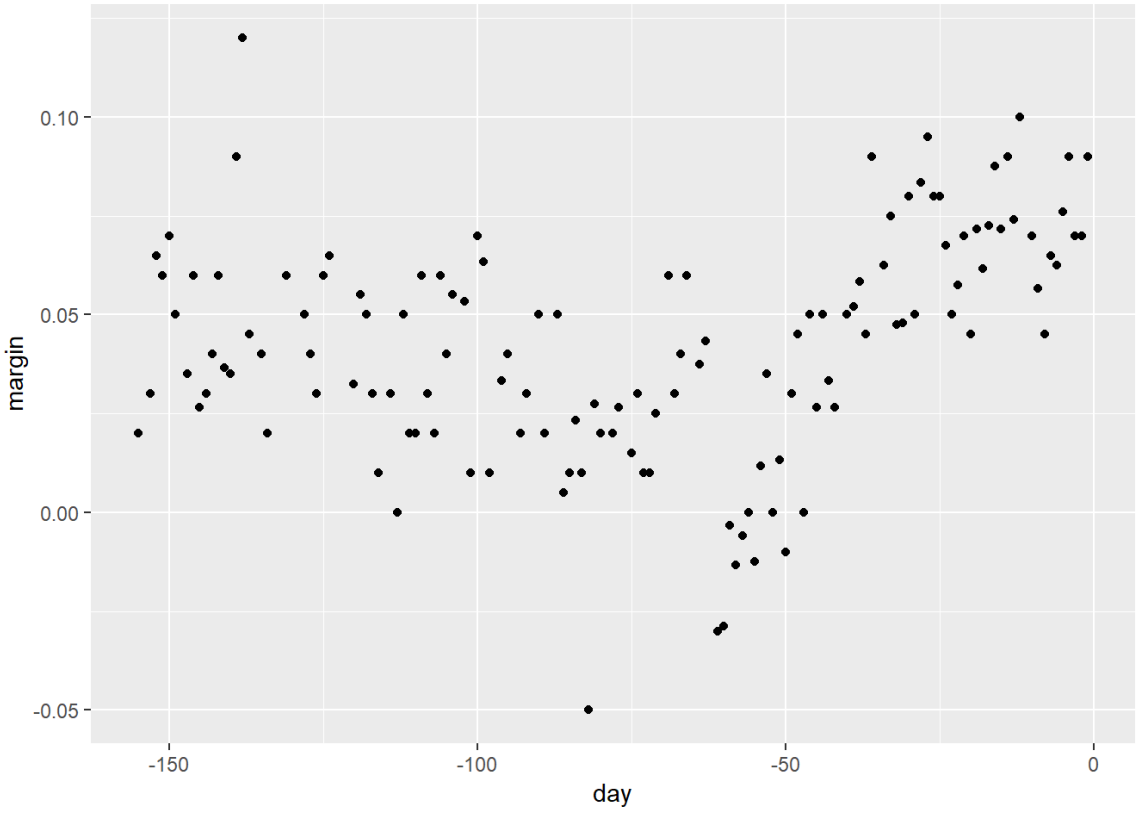
```
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------------------------------
---------------------- tidyverse 1.2.1 --
```

```
## v ggplot2 3.1.0     v purrr   0.2.5
## v tibble  1.4.2     v dplyr   0.7.8
## v tidyr   0.8.2     v stringr 1.3.1
## v readr   1.3.1     v forcats 0.3.0
```

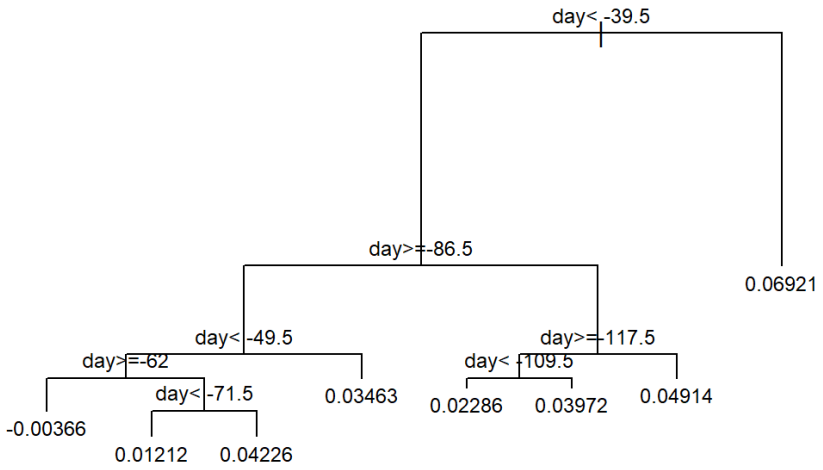
```
## -- Conflicts ------------------------------------------------------------------------------------------------
----------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(dslabs)
library(rpart)
# Regression Tree
data("polls_2008")
qplot(day, margin, data = polls_2008)
```



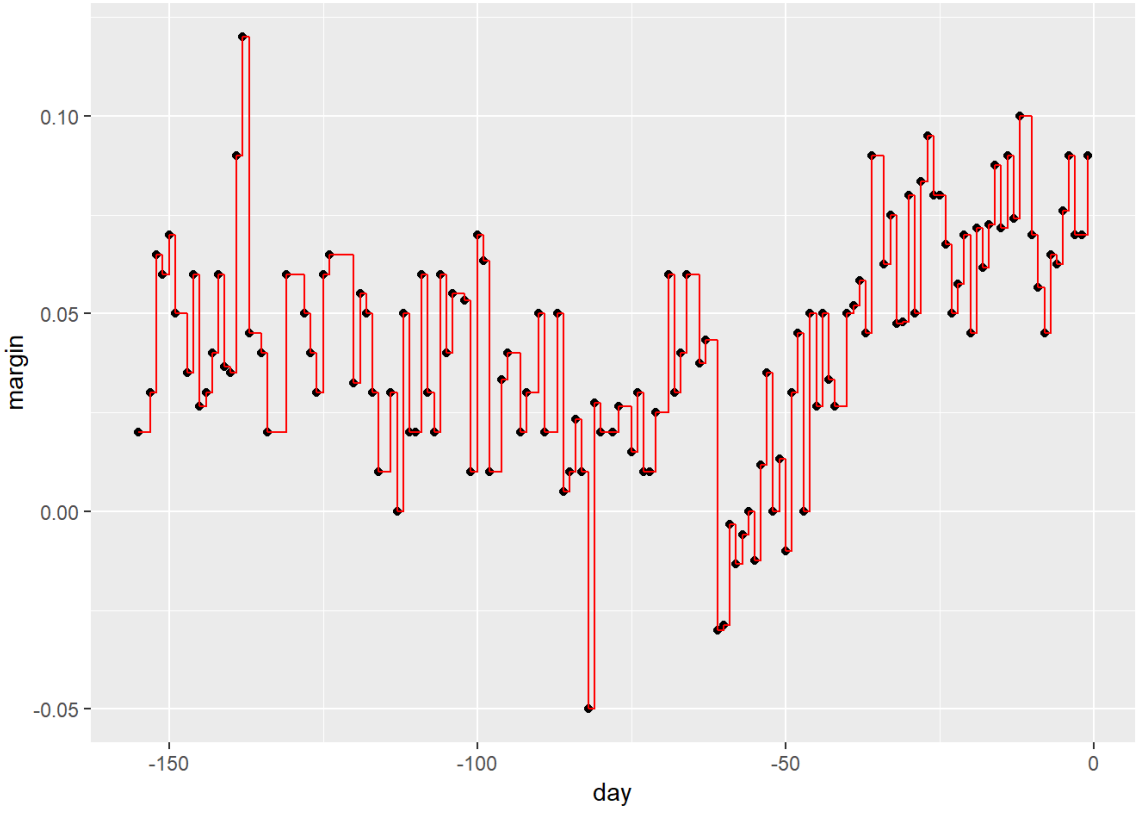
```
# Partitions feature space into  J non-overlappin,
# For every observation that falls within region
# predict with the average of the training observations in the region:
fit <- rpart(margin ~ ., data = polls_2008)

plot(fit, margin = 0.1)
text(fit, cex = 0.75)
```




```
polls_2008 %>%
  mutate(y_hat = predict(fit)) %>%
  ggplot() +
  geom_point(aes(day, margin)) +
  geom_step(aes(day, y_hat), col="red")
```

```
# rpart(y ~ ,. data= , control=rpart.control(cp=0,minsplit=20 minbucket=round(minsplit/3)))
fit <- rpart(margin ~ ., data = polls_2008, control = rpart.control(cp = 0, minsplit = 2))
polls_2008 %>%
  mutate(y_hat = predict(fit)) %>%
  ggplot() +
  geom_point(aes(day, margin)) +
  geom_step(aes(day, y_hat), col="red")
```



```
pruned_fit <- prune(fit, cp = 0.01)
polls_2008 %>%
  mutate(y_hat = predict(pruned_fit)) %>%
  ggplot() +
  geom_point(aes(day, margin)) +
  geom_step(aes(day, y_hat), col="red")
```

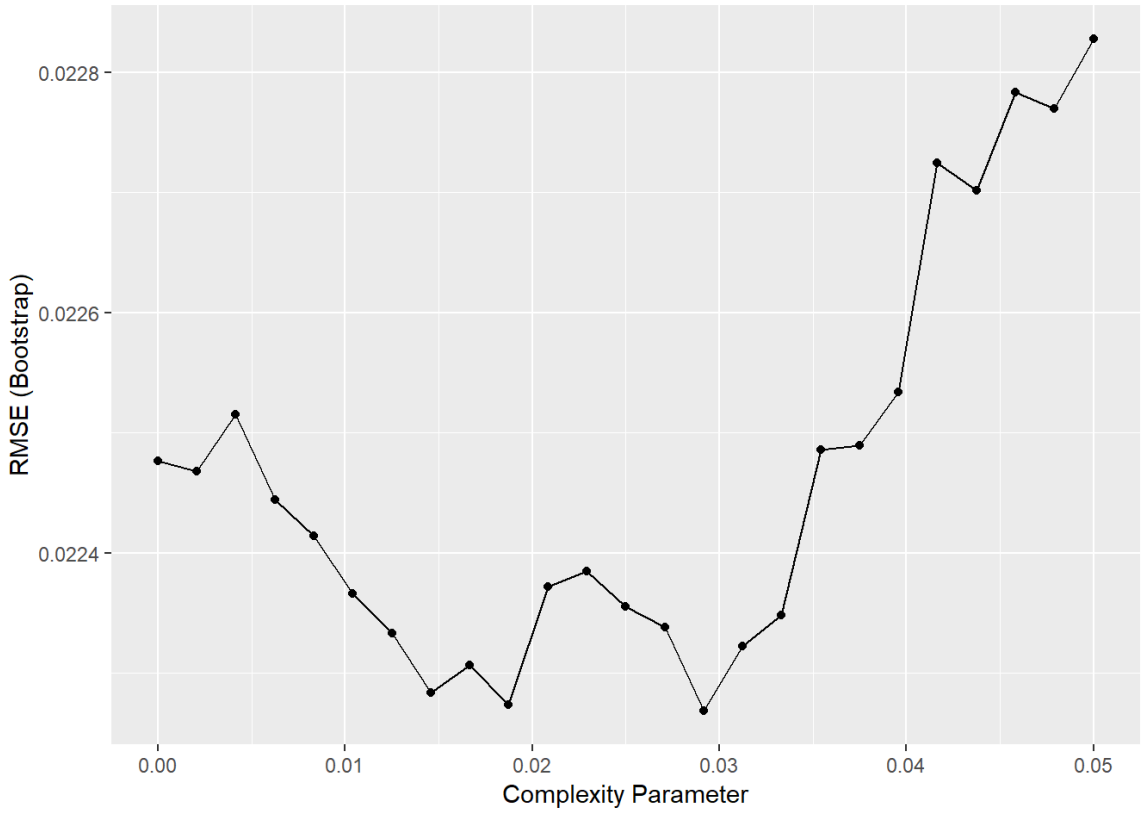
```
# How to pick CP
library(caret)
```

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

```
##
## Attaching package: 'caret'
```

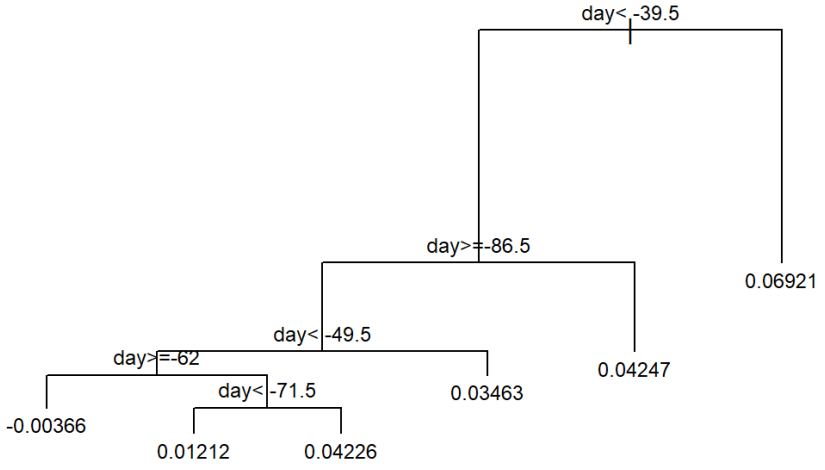
```
## The following object is masked from 'package:purrr':
##
##     lift
```

```
train_rpart <- train(margin ~ .,
                     method = "rpart",
                     tuneGrid = data.frame(cp = seq(0, 0.05, len = 25)),
                     data = polls_2008)
ggplot(train_rpart)
```




```
# Resulting Tree

plot(train_rpart$finalModel, margin = 0.1)
text(train_rpart$finalModel, cex = 0.75)
```



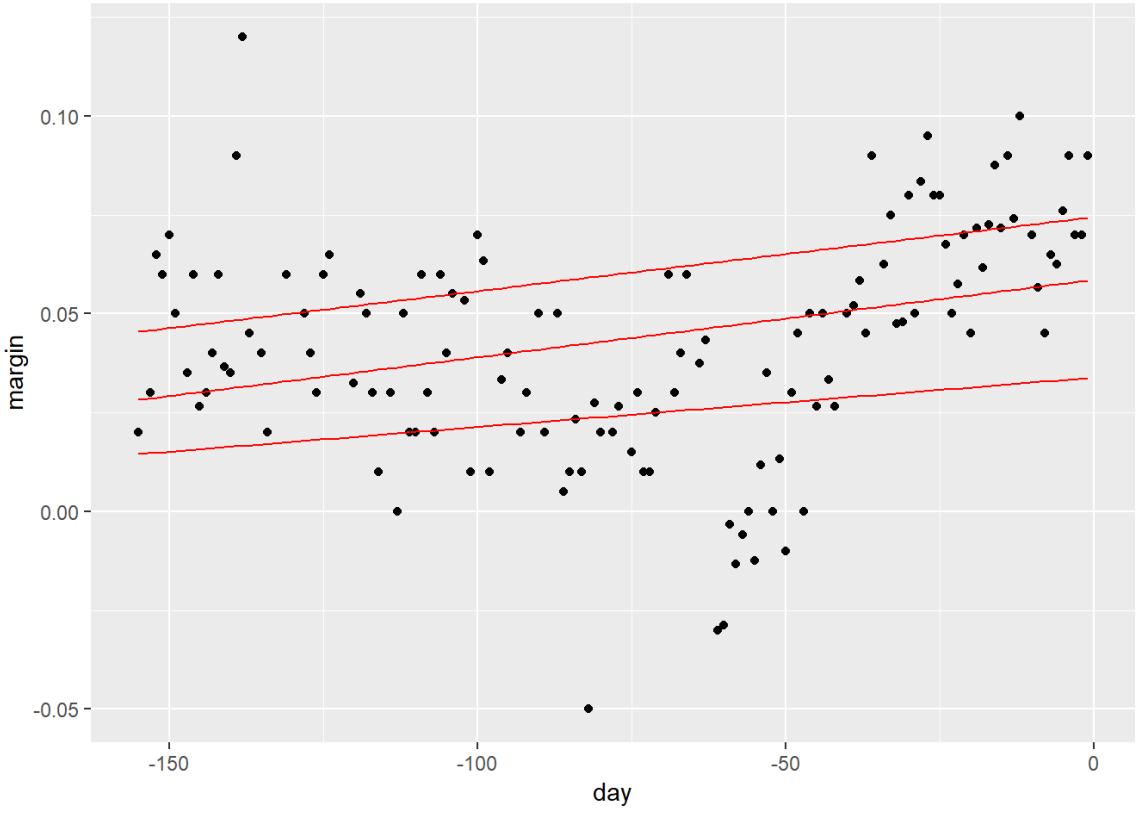
```
polls_2008 %>%
  mutate(y_hat = predict(train_rpart)) %>%
  ggplot() +
  geom_point(aes(day, margin)) +
  #geom_step(aes(day, margin), col="red") +
  #geom_smooth(aes(day, margin), col=2) +
  geom_quantile(aes(day, margin), col=2)
```

```
## Loading required package: SparseM
```

```
##
## Attaching package: 'SparseM'
```

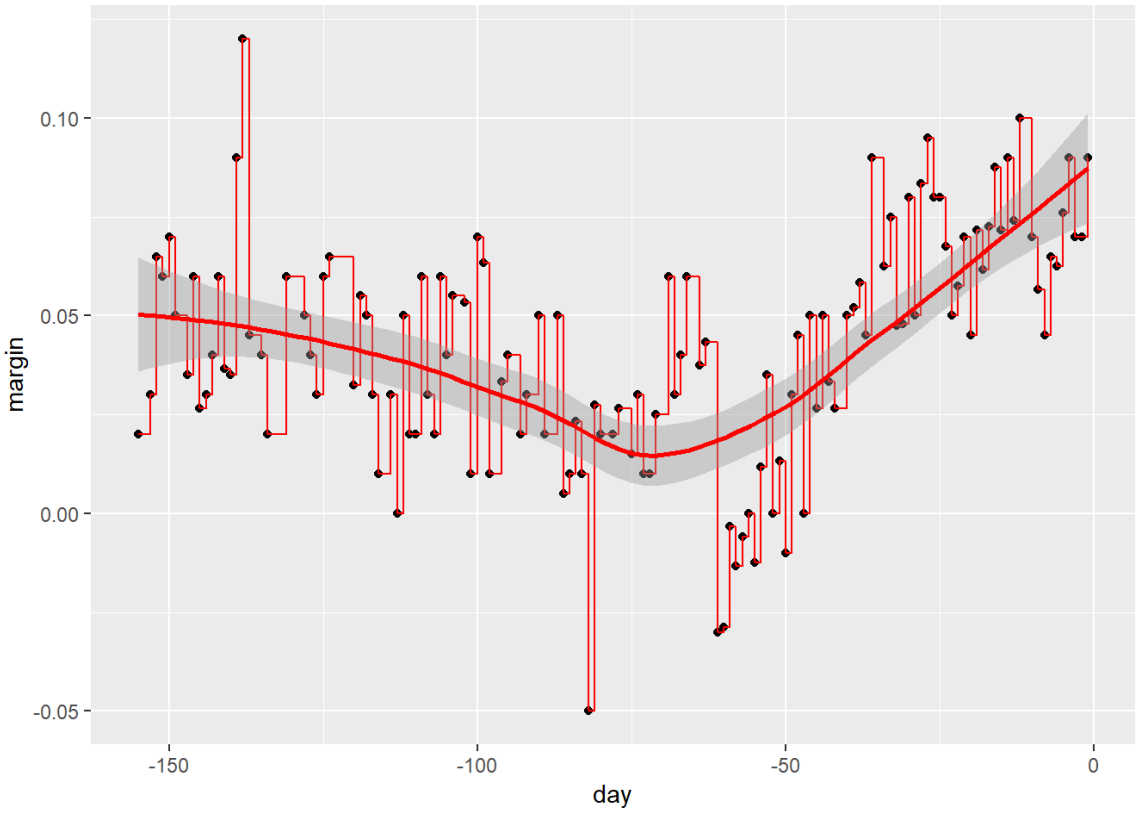
```
## The following object is masked from 'package:base':
##
##     backsolve
```

```
## Smoothing formula not specified. Using: y ~ x
```




```
polls_2008 %>%
  mutate(y_hat = predict(train_rpart)) %>%
  ggplot() +
  geom_point(aes(day, margin)) +
  geom_step(aes(day, margin), col="red") +
  geom_smooth(aes(day, margin), col=2)
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```




```
  #geom_quantile(aes(day, margin), col=2)


# Random Forests - improve prediction performance and reduce instability by averaging multiple decision trees (a
 forest of trees constructed with randomness)
# bootstrap aggregation or bagging
# Build many decision trees  using the training set
#   Create a bootstrap training set by sampling
#   N observations from the training set with replacement.
#   Build a decision tree from bootstrap training set.
# For every observation in the test set, form a prediction using Tree
# For continuous outcomes, predict
# For categorical data classification predict  with majority vote (most frequent class

library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 3.5.3
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

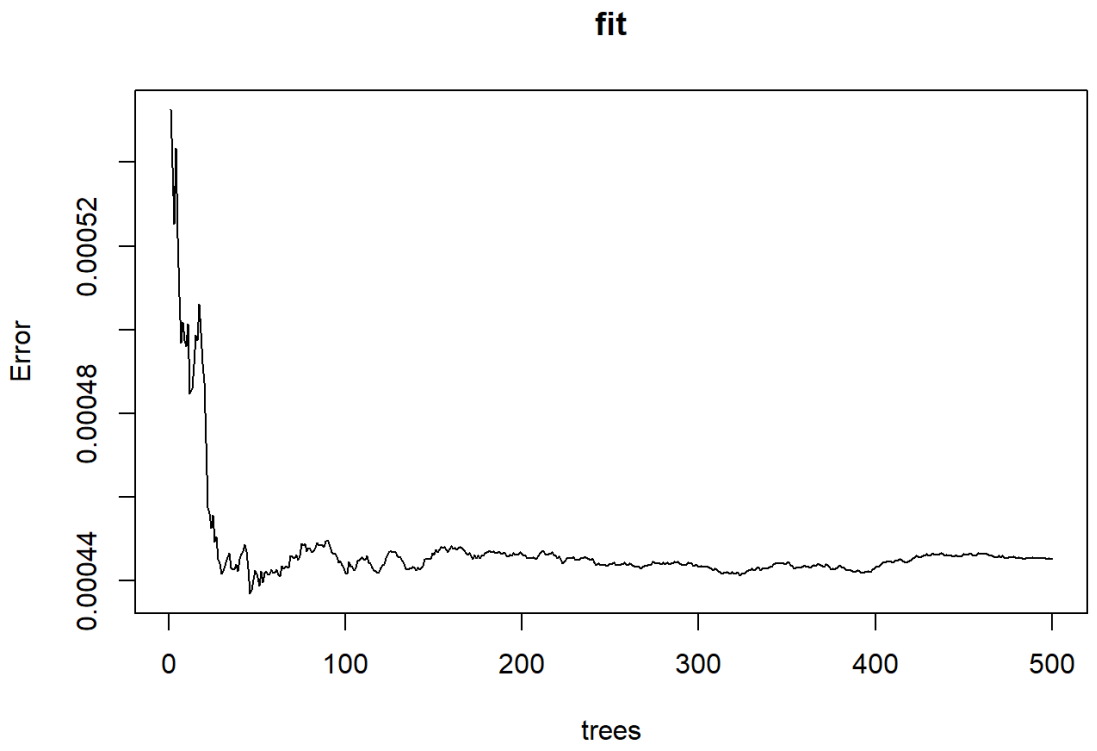
```
##
## Attaching package: 'randomForest'
```

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

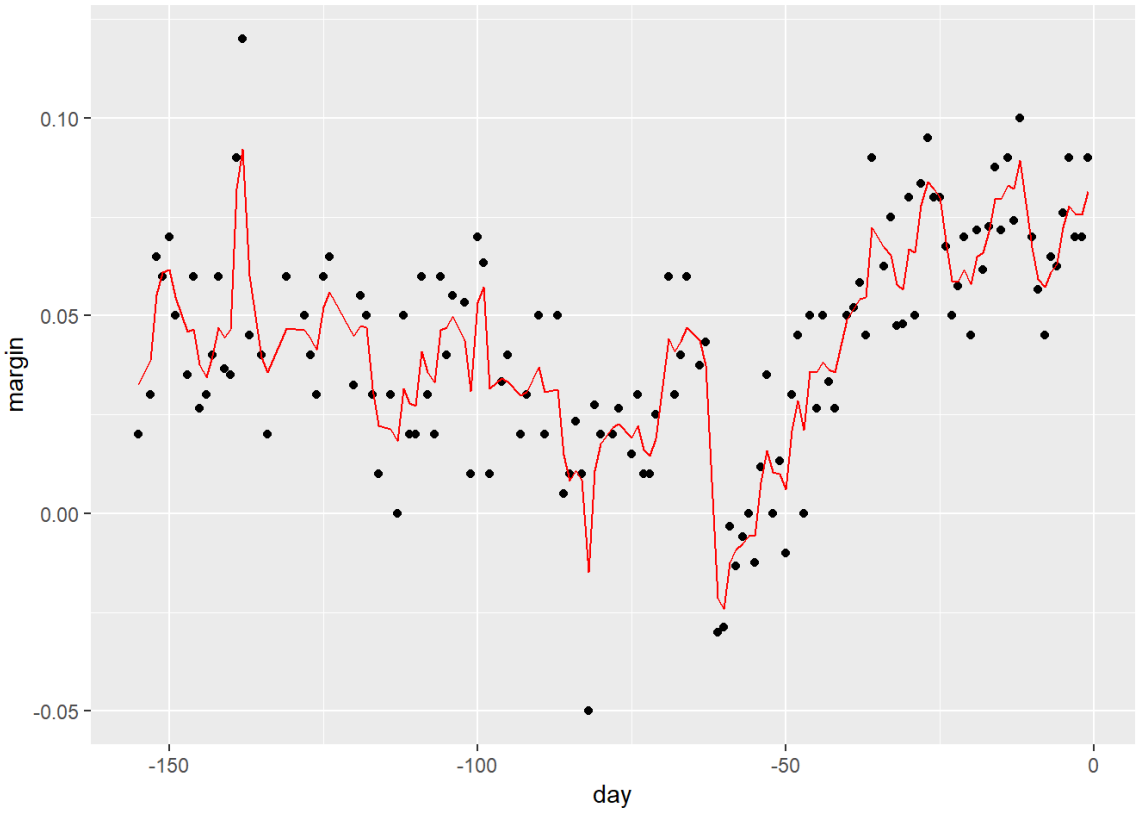
```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
fit <- randomForest(margin~., data = polls_2008)

plot(fit)
```




```
polls_2008 %>%
  mutate(y_hat = predict(fit, newdata = polls_2008)) %>%
  ggplot() +
  geom_point(aes(day, margin)) +
  geom_line(aes(day, y_hat), col="red")
```


0.10
0.05
0.00
-0.05
margin
-150
-100
-50
0
day