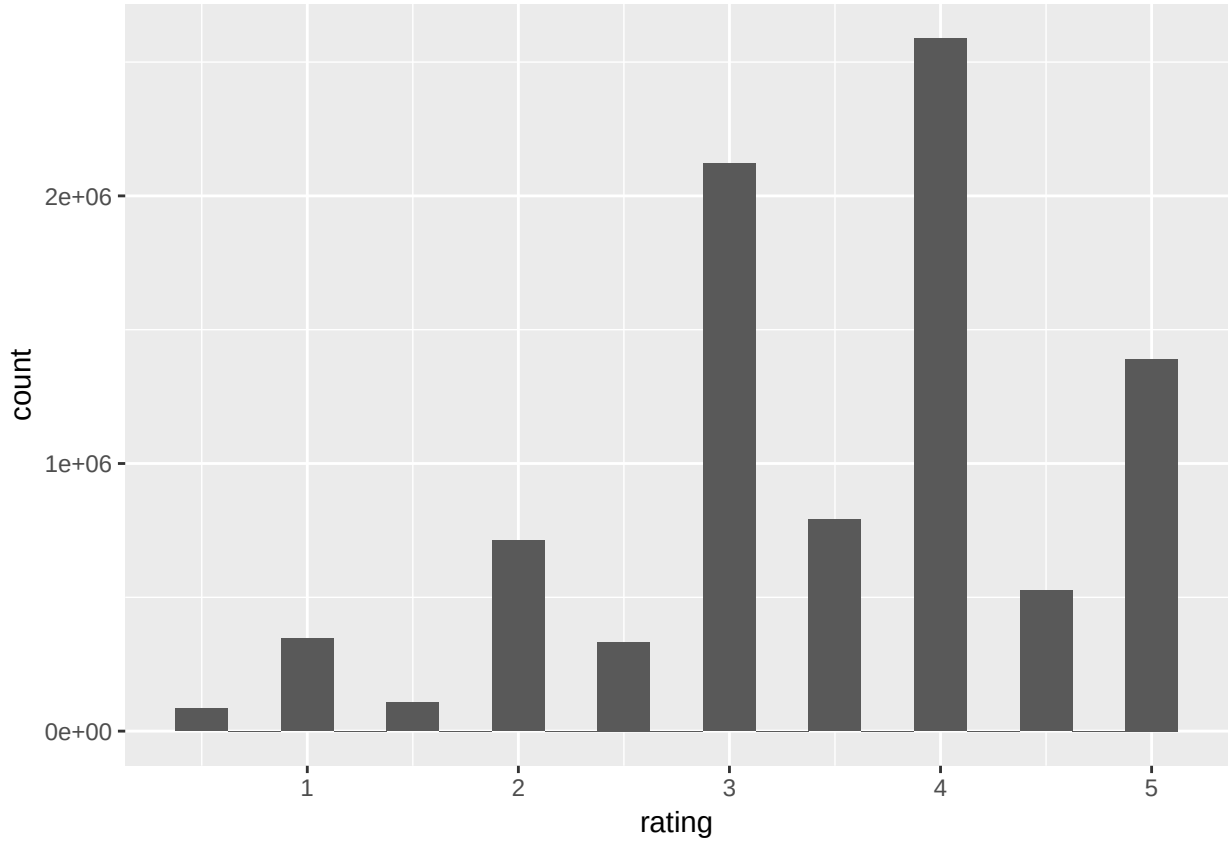
Report

Egar Garcia

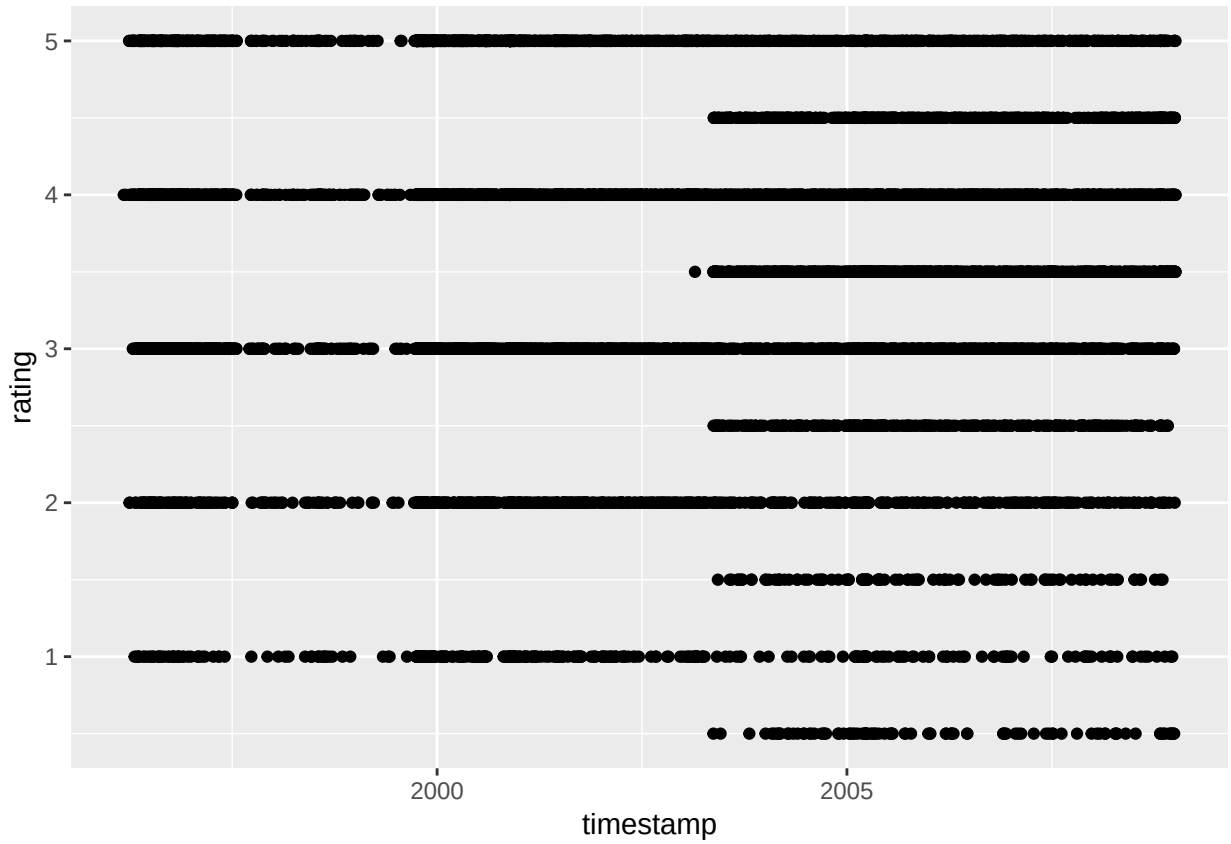
1/22/2019

Overview

```
edx %>%
  ggplot() +
  geom_histogram(aes(x = rating), binwidth = 0.25)
```



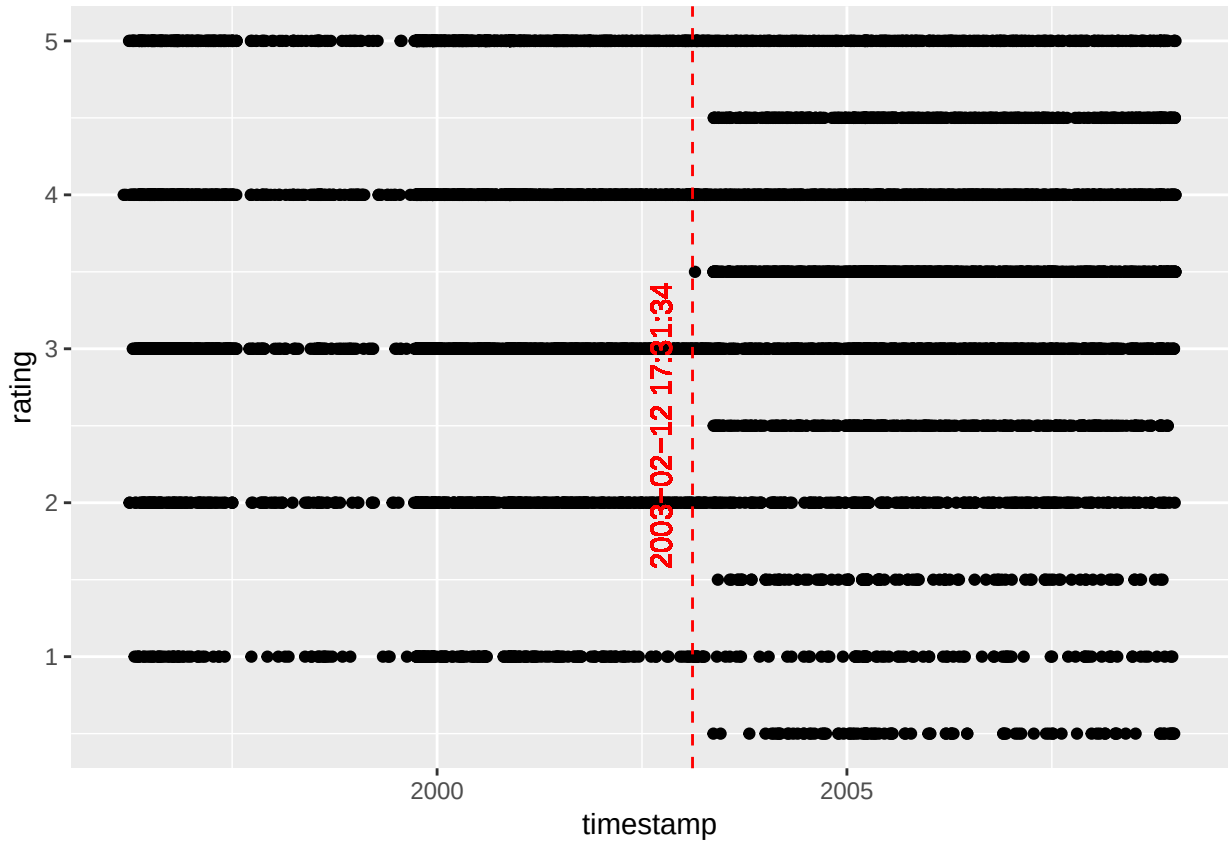

```
edx[createDataPartition(y = edx$rating, times = 1, p = 0.001, list = FALSE),] %>%
  ggplot(aes(x = as_datetime(timestamp), y = rating)) +
  geom_point() +
  labs(x = 'timestamp', y = 'rating')
```




```
half_stars_startpoint <- min(filter(edx, (rating * 2) %% 2 == 1)$timestamp)
```

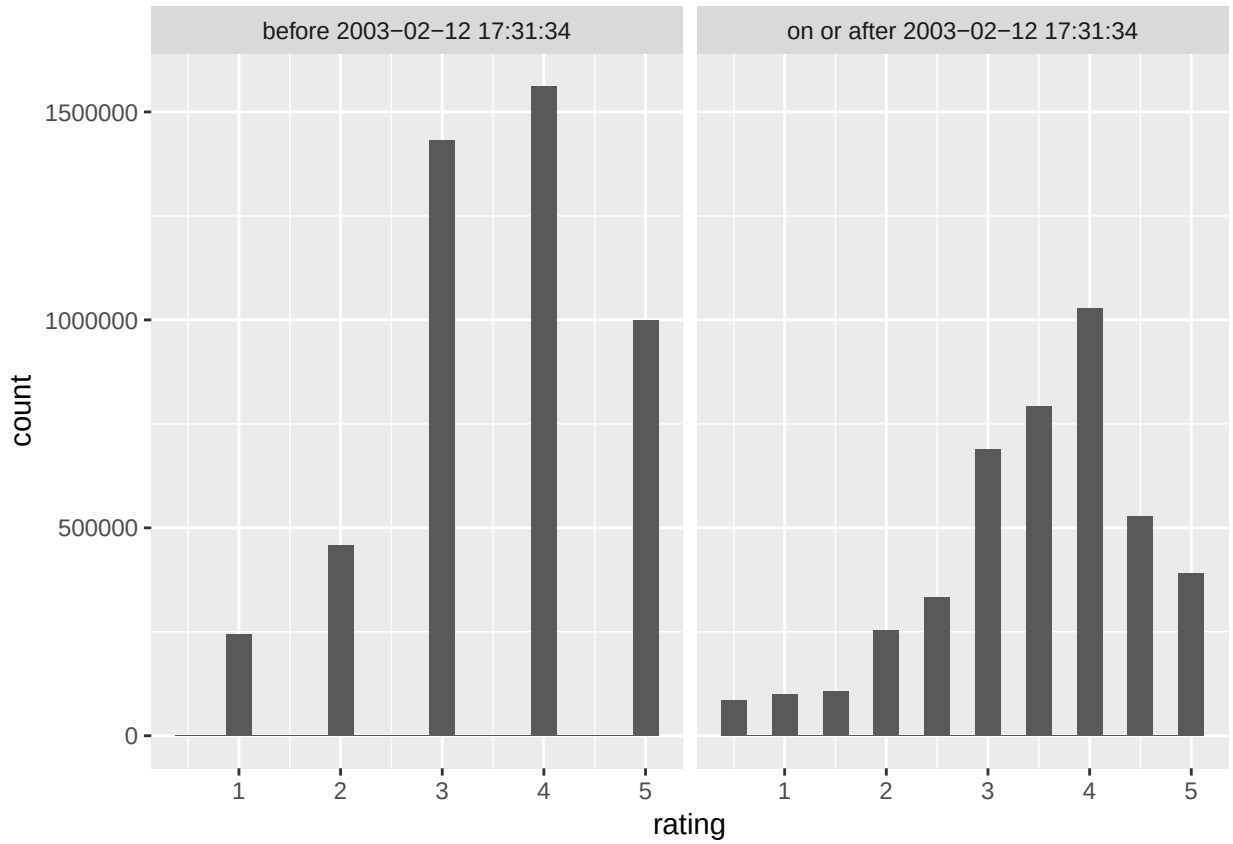
2003-02-12 17:31:34

```
edx[createDataPartition(y = edx$rating, times = 1, p = 0.001, list = FALSE),] %>%
  ggplot(aes(x = as_datetime(timestamp), y = rating)) +
  geom_point() +
  geom_vline(aes(xintercept = as_datetime(half_stars_startpoint)),
             color = "red", linetype = "dashed") +
  geom_text(aes(x = as_datetime(half_stars_startpoint),
                label = as_datetime(half_stars_startpoint),
                y = 2.5),
            color = "red", vjust = -1, angle = 90) +
  labs(x = 'timestamp', y = 'rating')
```




```
partition_names = c(paste('before', as_datetime(half_stars_startpoint)),
                    paste('on or after', as_datetime(half_stars_startpoint)))

edx %>%
  mutate(partition = factor(ifelse(timestamp < half_stars_startpoint,
                                   partition_names[1], partition_names[2]),
                            levels = partition_names)) %>%
  ggplot() +
  geom_histogram(aes(x = rating), binwidth = 0.25) +
  facet_grid(~ partition)
```




```
#' This object-constructor function is used to generate a metamodel
#' that contains two models,
#' one fitted for data before the startpoint when half stars were allowed in the ratings,
#' and the other one fitted for data on or after that startpoint.
#' The predictions are performed by choosing the appropriate model according to the
#' data's timestamp.
#'
#' @param dataset The dataset used to fit both models,
#'    it should contain a column called 'timestamp'
#' @param base_model_generator The function used to generate the base models,
#'    it should receive a dataset to fit the model and have a prediction function
#' @return The created metamodel
PartitionedtModel <- function(dataset, base_model_generator) {
  partitioned_model <- list()

  # Spliting the dataset in 2,
  # one set for data before the startpoint when half stars were allowed
  dataset1 <- dataset %>% filter(timestamp < half_stars_startpoint)
  # and the other one for the data on or after the startpoint when half stars were allowed
  dataset2 <- dataset %>% filter(timestamp >= half_stars_startpoint)

  # Generating a model for each dataset
  partitioned_model$model1 <- base_model_generator(dataset1)
  partitioned_model$model2 <- base_model_generator(dataset2)

  #' Performs a prediction with the combined fitted models,
  #' it tries to do the prediction with the respective model based on the timestamp,
```

```
  #' but if the prediction can not be performed then the other model is attempted.
  #' @param s The dataset used to perform the prediction of
  #' @return A vector containing the prediction for each row of the dataset
  partitioned_model$predict <- function(s) {
    # Performing the predictions on the whole dataset for each one of the models
    pred1 <- partitioned_model$model1$predict(s)
    pred2 <- partitioned_model$model2$predict(s)

    # Selecting the prediction to use according to the data's timestamp,
    # if a prediction is missing the prediction for the other model is used
    s %>%
      mutate(pred1 = pred1, pred2 = pred2) %>%
      mutate(pred = ifelse(timestamp < half_stars_startpoint,
                           ifelse(!is.na(pred1), pred1, pred2),
                           ifelse(!is.na(pred2), pred2, pred1))) %>%
      .$pred
  }

  partitioned_model
}
```

```
#' Converts a prediction (which is a floating point number) to a one used to
#' represent ratings given by stars,
#' i.e. {1, 2, 3, 4, 5} if the timestamp is before the half start startpoint
#' or {1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5} if the timestamp is on or after.
pred2stars <- function(timestamp, pred) {
  # Rounds the prediction either to be full-stars or having a half-star
  # according to the timestamp
  rounded_pred <- ifelse(timestamp < half_stars_startpoint,
                         round(pred),
                         round(pred * 2)/2)

  # Making sure the rating is not smaller that 1 or bigger than 5
  min(max(rounded_pred, 1), 5)
}
```

Methods

Simple Average

```
#' This object-constructor function is used to generate a model
#' that always returns as prediction the average of the rating in the
#' given dataset used to fit the model.
#' @param dataset The dataset used to fit the model
#' @return The model
SimpleAvgModel <- function(dataset) {
  model <- list()

  # The average of ratings
  model$mu <- mean(dataset$rating)

  #' The prediction function
```

```
  #' @param s The dataset used to perform the prediction of
  #' @return A vector containing the prediction
  model$predict <- function(s) {
    model$mu
  }

  model
}
```

```
pred <- SimpleAvgModel(edx)$predict(edx)
```

```
RMSE(pred, edx$rating)
```

```
## [1] 1.060331
```

```
mean(pred2stars(edx$timestamp, pred) == edx$rating)
```

```
## [1] 0.2876016
```

Pseudo Linear Model

See: https://rafalab.github.io/dsbook/recommendation-systems.html

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

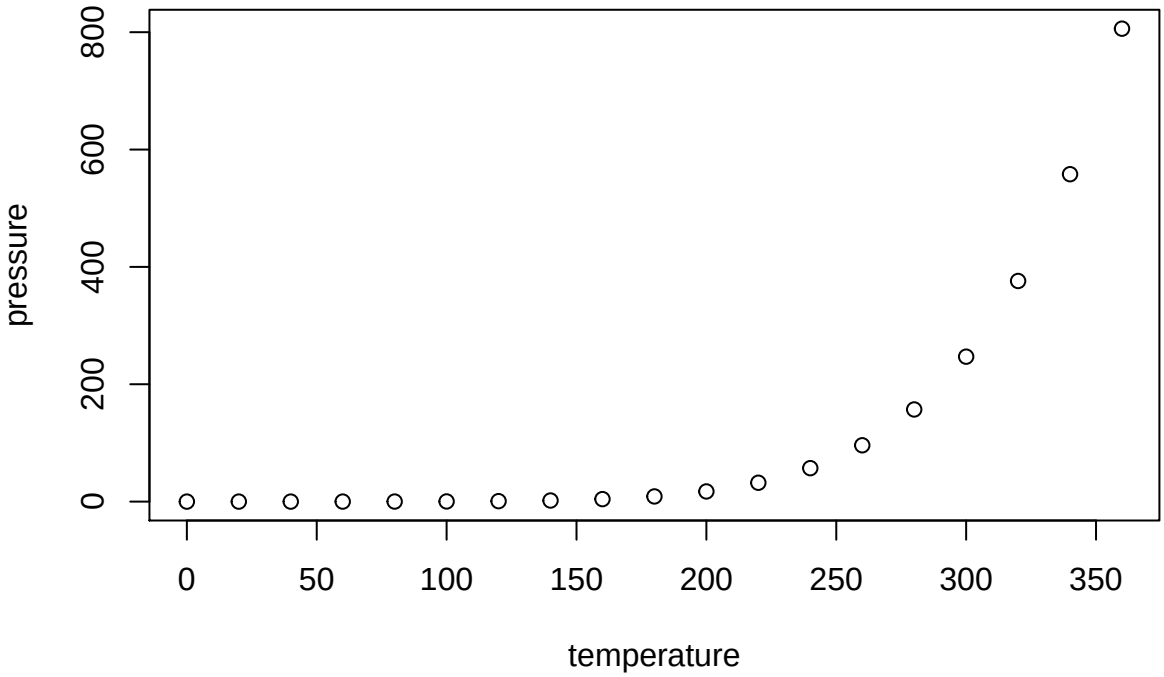
When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
summary(cars)
```

```
##      speed           dist
##  Min.   : 4.0   Min.   :  2.00
##  1st Qu.:12.0   1st Qu.: 26.00
##  Median :15.0   Median : 36.00
##  Mean   :15.4   Mean   : 42.98
##  3rd Qu.:19.0   3rd Qu.: 56.00
##  Max.   :25.0   Max.   :120.00
```

Including Plots

You can also embed plots, for example:



Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.