Project1_10M100K

I Zidan

15/04/2020

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

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

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

Mission

- Create a movie recommendation system using a customised datasets.
- The datasets used is based on the 10M version of the MovieLens dataset.
- Code for the dataset supplied by lecturer Rafael Razi.

Given

- edx dataset as described above.
- validation dataset for final prediction.

Requirement

- Split the provided edx dataset into training and test partitions.
- Analyse data.
- Create a recommendation system.
- Train algorithim
- Test algorthim
- Pproduce final predictions on the providedvalidation dataset.
- Provide RMSE figures as a final ouput.

Methodology

When creating the algorithim, both movie and user effect on rating will be considered. Based on regularisation principal, penalising concept will be followed for optimised results.

References

Course lecture notes

1- loading or creating edx and validation data (Refere to the script to see code details)

.RDATA files are provided. to save time, place the file in the same directory as the script and the script will pick them up. I found that if you run the data data extraction routine through RStudio, couple of records are missing in the edx dataset Please download files using the below links. Place them in the same directory as the script.

validation file : https://drive.google.com/open?id=1vJDE-26kh5ioWO7QX88Y-xoeRx4i7u8Q
edx file: https://drive.google.com/open?id=15GqskACjQXbd-xyI-kPnQoSWi1vbZZSF

2 - analyse contents

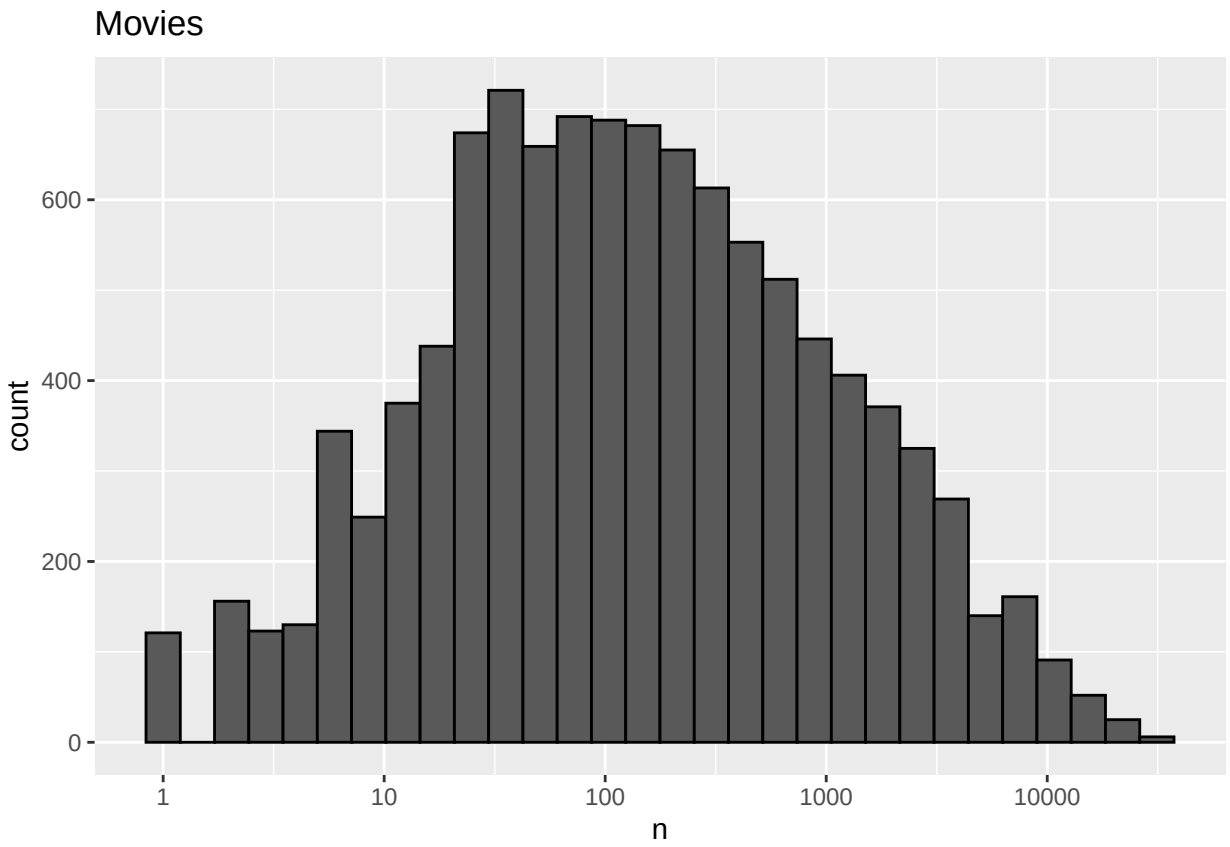
2.1 - data overview in a snaphot

```
summary(edx)
```

```
##      userId         movieId          rating        timestamp
##  Min.   :    1   Min.   :    1   Min.   :0.500   Min.   :7.897e+08
##  1st Qu.:18122   1st Qu.:  648   1st Qu.:3.000   1st Qu.:9.468e+08
##  Median :35743   Median : 1834   Median :4.000   Median :1.035e+09
##  Mean   :35869   Mean   : 4120   Mean   :3.512   Mean   :1.033e+09
##  3rd Qu.:53602   3rd Qu.: 3624   3rd Qu.:4.000   3rd Qu.:1.127e+09
##  Max.   :71567   Max.   :65133   Max.   :5.000   Max.   :1.231e+09
##     title              genres
##  Length:9000061     Length:9000061
##  Class :character   Class :character
##  Mode  :character   Mode  :character
##
##
##
```

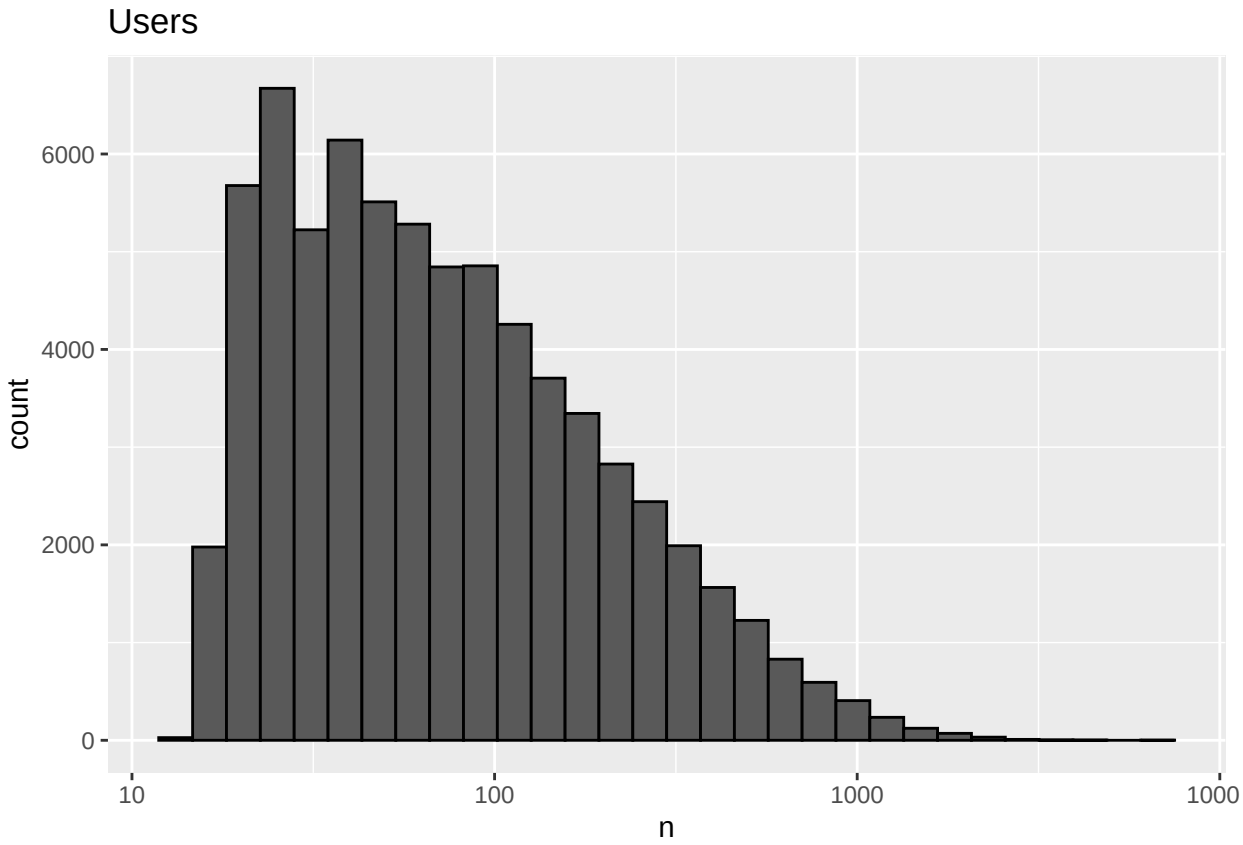
2.2 - Ploting rating by movie.

This will visualise and in turn give an idea of movies' effect on rating. Using histogram and bucketing



2.3 - Ploting rating by users.

This will visualise and in turn give an idea of users' effect on rating.



3 - partition dataset.

30% of the edx data goes to test and the rest for training

```
set.seed(1)
rating_test_index <- createDataPartition(y = edx$rating, times = 1,  p = 0.3, list = FALSE)
rating_train_set <- edx[-rating_test_index,]
rating_test_set <- edx[rating_test_index,]
```

4 - process data.

```
# function to set current dataset to use
WhichDataSet <- function(partition) {

  if (partition == "train") {
     rating_train_set
  } else if (partition == "test") {
      rating_test_set
  } else if (partition == "validation") {
     validation
  }
```

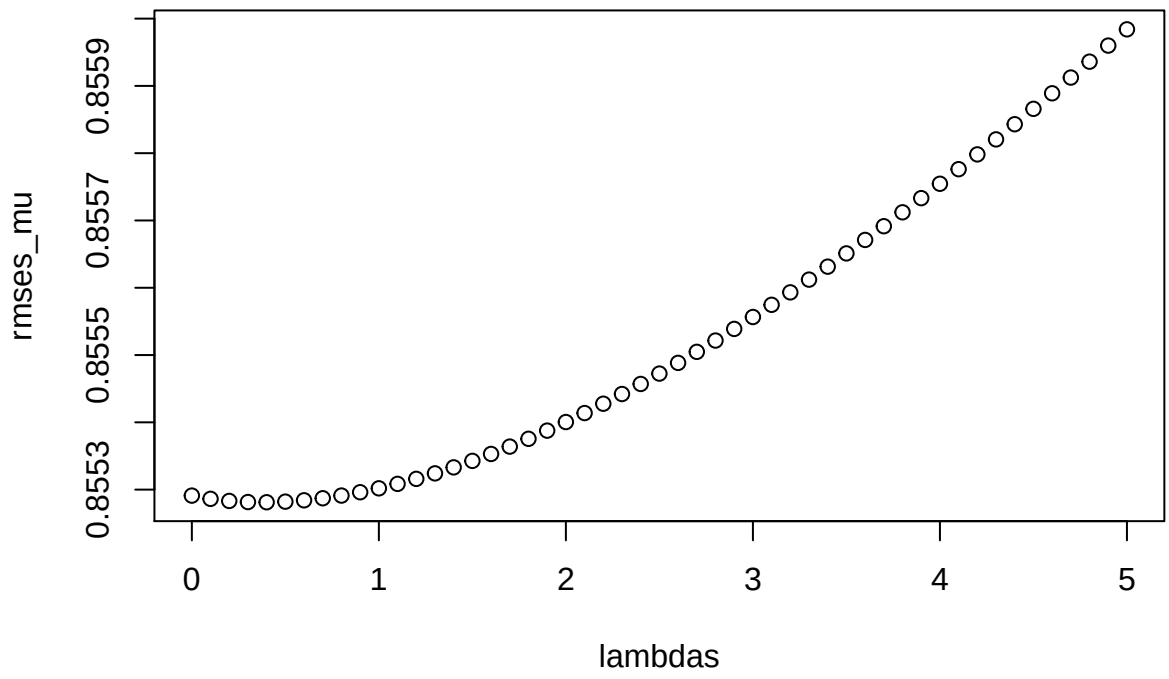
```
}


# Root Mean Square Error
RMSE <- function(true_ratings, predicted_ratings){
  sqrt(mean((true_ratings - predicted_ratings)^2))
}


# Set the dataset to training
dSet <- WhichDataSet("train")

str_c("there are ",nrow(dSet)," records in the training dataset")
```

```
## [1] "there are 6300042 records in the training dataset"
```

```
# The mean of the training data set
mu <- mean(rating_train_set$rating)


# tuning factors sequence vector
lambdas <- seq(0, 5, 0.1)


# run algorithim and produce RMSE for each lamda and store in a vector
rmses_mu <- sapply(lambdas, function(l){


  # Regularisation to estimate movies effect.
  # Penalising large estimates that come from small samples
    low_ratings <- rating_train_set%>%
                   group_by(movieId)%>%
                   summarize(low_ratings = sum(rating - mu)/(n()+l))


  #Regularisation to estimate user effect.
  #Penalising large estimates that come from small samples
  scaled_mean <- rating_train_set %>%
                 left_join(low_ratings, by="movieId") %>%
                 group_by(userId) %>%
                 summarize(scaled_mean = sum(rating - low_ratings - mu)/(n()+l))


  # extract penalty value lambdas
  predicted_ratings <-
    dSet %>%
    left_join(low_ratings, by = "movieId") %>%
    left_join(scaled_mean, by = "userId") %>%
    mutate(pred = mu + low_ratings + scaled_mean) %>%.$pred

  return(RMSE(dSet$rating,predicted_ratings))

})
```

5 - Cross validation of lambda tuning parameter.

```
plot(lambdas, rmses_mu)
```



6 - Output the best lambda

```
min_lambda <- lambdas[which.min(rmses_mu)]

str_c('Smallest RMSE: ', min(rmses_mu), ' by lambda: ', min_lambda)
```

```
## [1] "Smallest RMSE: 0.855281241734575 by lambda: 0.4"
```

7 - Store the result in a tibble

```
rmse_results <- tibble(method = "Result from traing dataset", RMSE = min(rmses_mu))
```

8 - Set the dataset to test dataset.

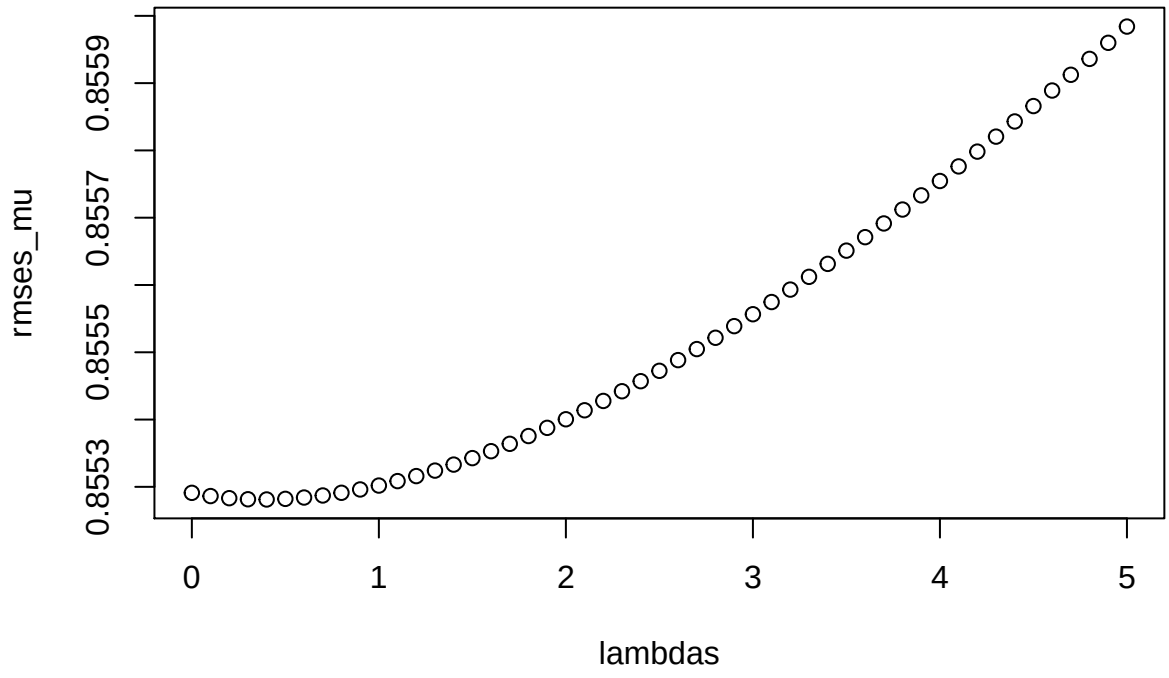
This is to run the algorithim on the test partition.

```
dSet <- WhichDataSet("test")
str_c("there are ",nrow(dSet)," records in the test dataset")
```

```
## [1] "there are 2700019 records in the test dataset"
```

8.1 - plot cross validation with the test dataset

```
plot(lambdas, rmses_mu)
```



8.2 output the best lambda when test dataset is used

```
min_lambda <- lambdas[which.min(rmses_mu)]

str_c('Smallest RMSE: ', min(rmses_mu), ' by lambda: ', min_lambda)
```

```
## [1] "Smallest RMSE: 0.855281241734575 by lambda: 0.4"
```

8.3 - Store the result in a tibble

```
rmse_results <- bind_rows(rmse_results,tibble(method = "Result from test dataset", RMSE = min(rmses_mu))
```

9 - assign lambda to the smalles value found

```
lambdas <- lambdas[which.min(rmses_mu)]
```

10 - Set the dataset to validation dataset.

This is to run the algorithim on the test partition.

```
dSet <- WhichDataSet("validation")
str_c("there are ",nrow(dSet)," records in the validation dataset")
```

```
## [1] "there are 999993 records in the validation dataset"
```

10.1 - run algorithim against the validation dataset

```
val_rmse <- rmses_mu
```

10.2 - Store the result in a tibble

```
rmse_results <- bind_rows(rmse_results,tibble(method = "Result from validation dataset", RMSE = min(val_
```

11 - output the tibble of all results

```
rmse_results
```

```
## # A tibble: 3 x 2
##   method                          RMSE
##   <chr>                          <dbl>
## 1 Result from traing dataset     0.855
## 2 Result from test dataset       0.855
## 3 Result from validation dataset 0.855
```

12 - Conclusion

Regulaisation provides an optimised and explained results. Final output is efficient when running the algorithim with the validation data.

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