# Project: `MovieLens`

*Ruben R. Kazumov*

## Contents



## The list of variables, acronyms and abbreviations

$i, j, t, q, p$ - iterators.

$R$ - The `ratings` data set.

$r_i$ - Element $i$ of the `ratings` data. Represents a rating, object, observation of the data set.

$N$ - The total number of ratings in the `ratings` data set. Constant.

$N^{u_q}$ - The total number of ratings made by user $u_i$.

$N^{m_p}$ - The total number of ratings of movie $m_i$.

$N^{m_p,u_q} = 1$ - The initial restriction of the rating system.

$b_u$ - The bias of user.

$b_m$ - The bias of movie.

$G$ - The `genres` vector.

$g_t$ - Single genre value, element of the `genres` vector.

$T$ - Total number of unique genres in the data set. Constant.

$U$ - The `users` data set.

$u_q$ - Element of the users data. Represents a user, object, observation of data set.

$Q$ - Total number of unique users in the data set. Constant.

$M$ - The `movies` data set.

$m_p$ - Element of the movies data. Represents a movie, object, observation of data set.

$Q$ - Total number of unique movies in the data set. Constant.

$S$ - The cumulative `sympathy` feature. Vector.

$s^{u,m}$ - The `sympathy` vector. Sympathy of the user $u$ to movie $m$.

$s_{r_i}$ - The `sympathy` vector. Sympathy in the line $i$ of the rating data.

$A$ - Cumulative average sympathy.

OBJECT[PARAMETER] - The value of the parameter in object.

RMSE - Root Mean Square Error parameter.

$J_{1,20}$ - Matrix of ones with dimention $1 \times 20$ (one line and twenty columns).

# The MovieLens Project

The "MovieLens" capstone project is the midterm project of the HarvardX PH125.9x course.

In this project we will try to build a recommendation system for the MovieLens dataset, based on a combined genre, movie and user effects.

The course project overview does not make any recommendations or set restrictions on the type of predictive model, but based on the content of the course PH 125.8x, one can see, the linear regression model is the expected model to be applied.

Due to certain circumstances, we will build regression and classification models of prediction. Later, the reasons for each will be explained in detail.

As stated in the course project description, the quality of model should be graded by RMSE parameter. The target expected value of RMSE is RMSE $\leq 0.87750$.

# The Initial Analysis Of The MovieLens Data

The initial rating data are presented by a tabular dataset (Fig.1)in the form of data frame. The global variables `edx` and `validation` are subsets of the MovieLens dataset.

The `edx` object is the data frame, contains 6 variables and 9 000 055 observations. The `validation` object is the data frame as well, with the same number of variables and 999 999 observations.

For the purpose of analysis of the content and building the model, we will temporarily join them together into the single `movieLens` data frame with 10 000 054 observations.

The dataset `movieLens` represents the rating activity of the users over the movies.

The users $U$, represented by variable `userId`, grade movies $M$, represented by variables `movieId`, `title` and `genres`, with rate value $R$, represented by variable `rating` in a moment in time, represented by variable `timestamp`:

$$R = f(M, U) \tag{1}$$

| **edx** |
|---|
| movieId : Numeric |
| userId : Integer |
| rating : Numeric |
| timestamp : Integer |
| title : Character |
| genres : Character |

| **validation** |
|---|
| movieId : Numeric |
| userId : Integer |
| rating : Numeric |
| timestamp : Integer |
| title : Character |
| genres : Character |

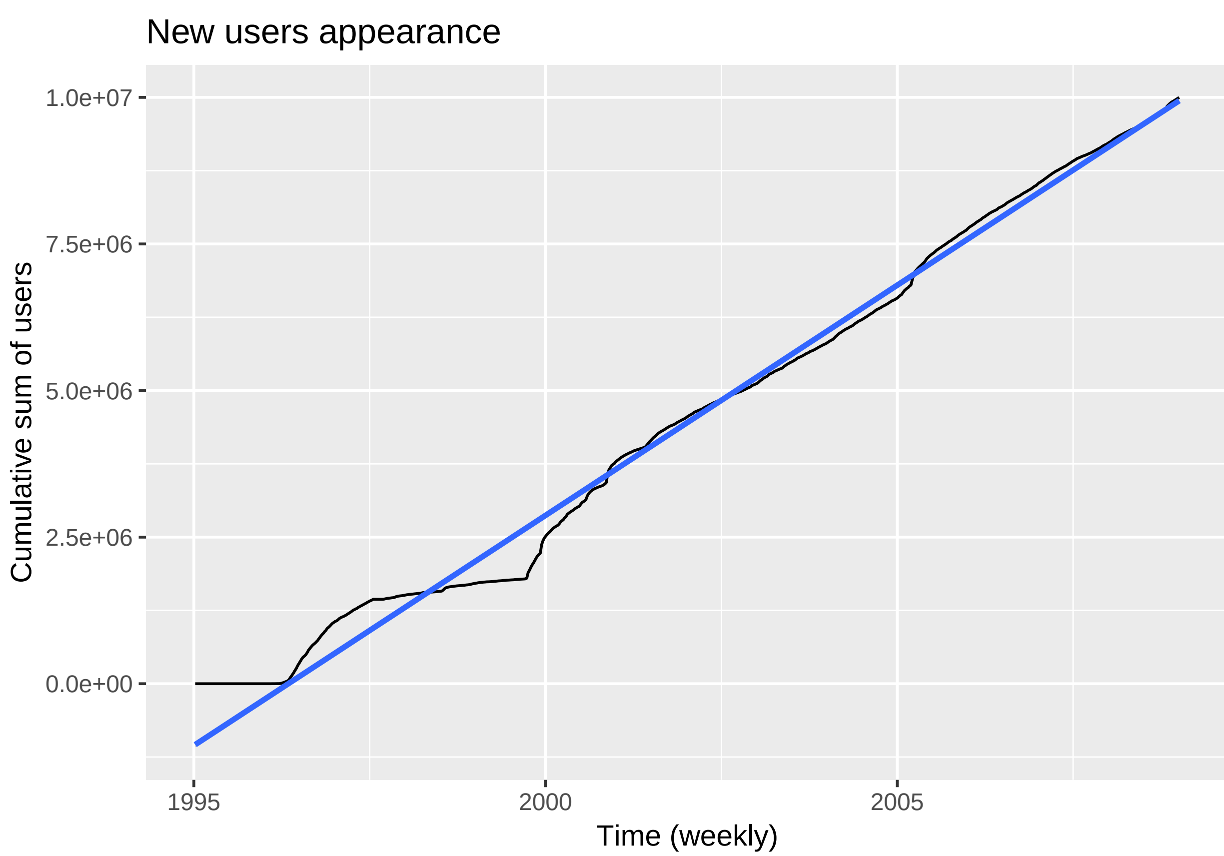
Figure 1: MovieLens data before transformation

Since model (1) describes the finite number of users ($Q = 69\,878$), movies ($P = 10\,677$), and the rating system indexes pair `userId` and `movieId`, i.e. permits the single unique pair of user and movie, the maximum possible size of the ratings matrix R is $dim(R) = 69\,878 \times 10\,677 = 746\,087\,406$:

$$R = f(M, U) = f(M \times U) = f(\begin{pmatrix} m_1 u_1 & \dots & m_1 u_{69878} \\ \vdots & \ddots & \vdots \\ m_{10677} u_1 & \dots & m_{10677} u_{69878} \end{pmatrix}) \quad (2)$$

The model (2) with the 746 087 406 features makes any prediction model technically impossible .

Also, we should take into account the tendency of the stable increase of the users number over time:




This means, the number of features in the matrix model should expand in the power of two just because of the users. The number of the movies in the system will increase as well.

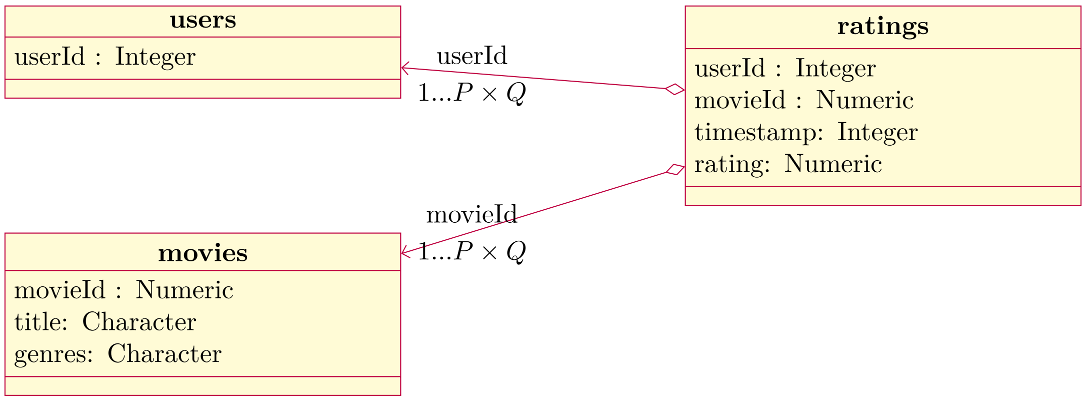



Figure 2: The logical structure of the dataset

## The Users Data

The users of the system are represented in the dataset by `userId` numeric variable (Fig.2). No more data are presented. The relationship of the `userId` to the ratings list is `many-to-many`.

## The Movies Data

The movies in the data set (Fig.2) are represented by tree variables: `movieId`, `title`, and `genres`.

The `movieId` numeric variable must be unique for the movie, just like the `title` one.

The `title` character string contains combined movie title and production year.

The `genres` character string contains the combined list of unique movie genres for a given movie.

## The Ratings Data

The ratings data itself (Fig.2) are presented by numeric variables: `timestamp`, `userId`, `movieId`, and `rating`.

The `timestamp` variable is the numeric representation of date and time of the moment when the rating was recorded.

The `rating` is a numeric variable and represents subjective grading of the movie by the user.

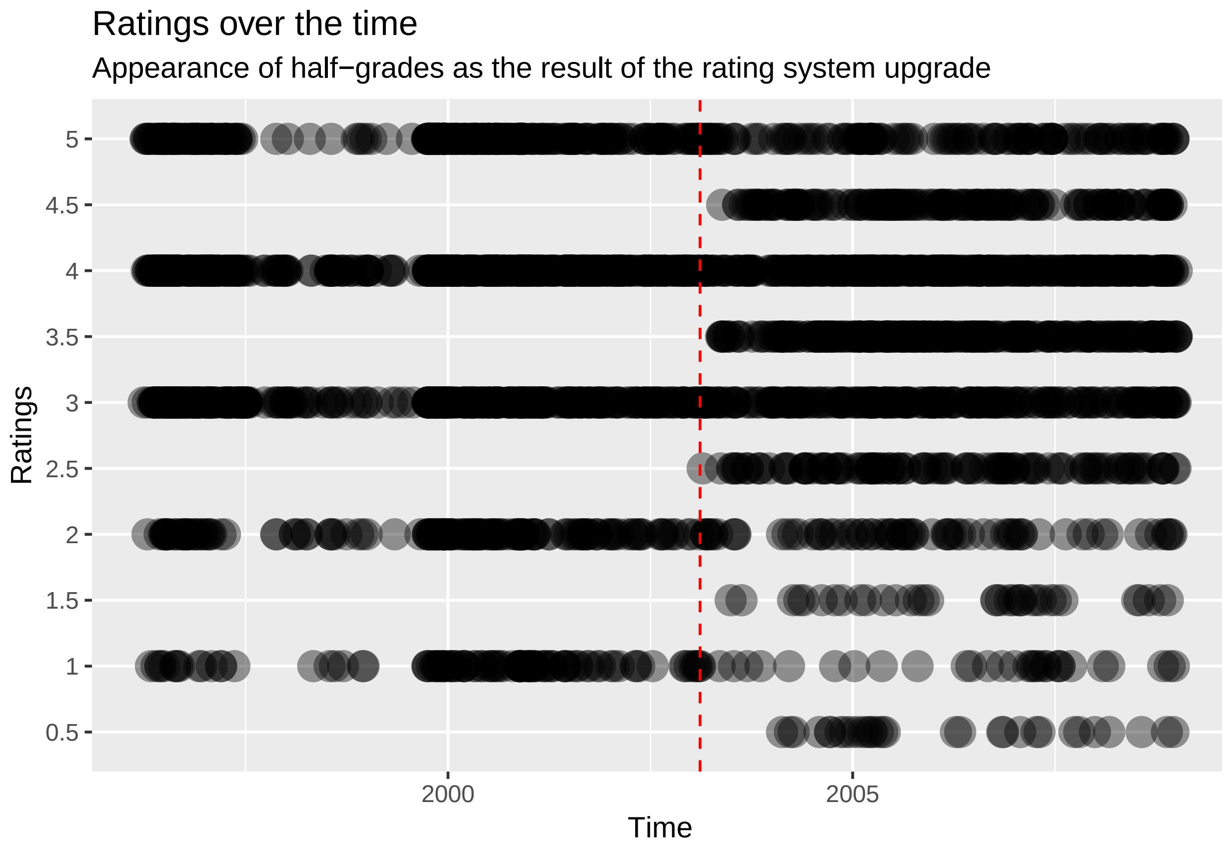
As it was mentioned above, a single user may rate a single movie only once, and the pair `userId` and `movieId` in the `ratings` data is unique.

$$N^{m_p,u_q} = 1 \tag{3}$$

The `rating` vector should be a factor with the levels:

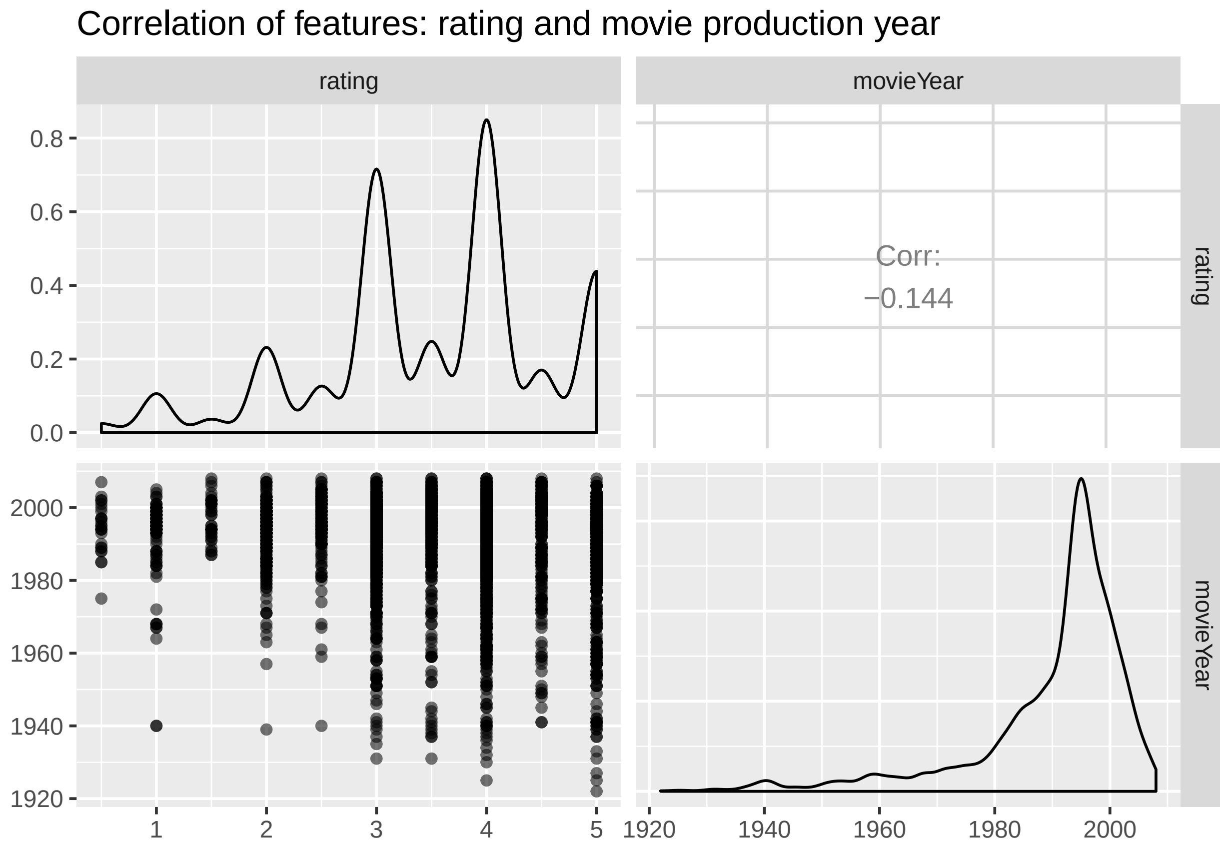
i) 1, 2, 3, 4, and 5 before 2003-02-12 17:31:34, and
ii) 0.5, 1, 1.5, 2, 2.5, 3, 3.4, 4, 4.5, and 5 after 2003-02-12 17:31:34, when the rating system was upgraded.

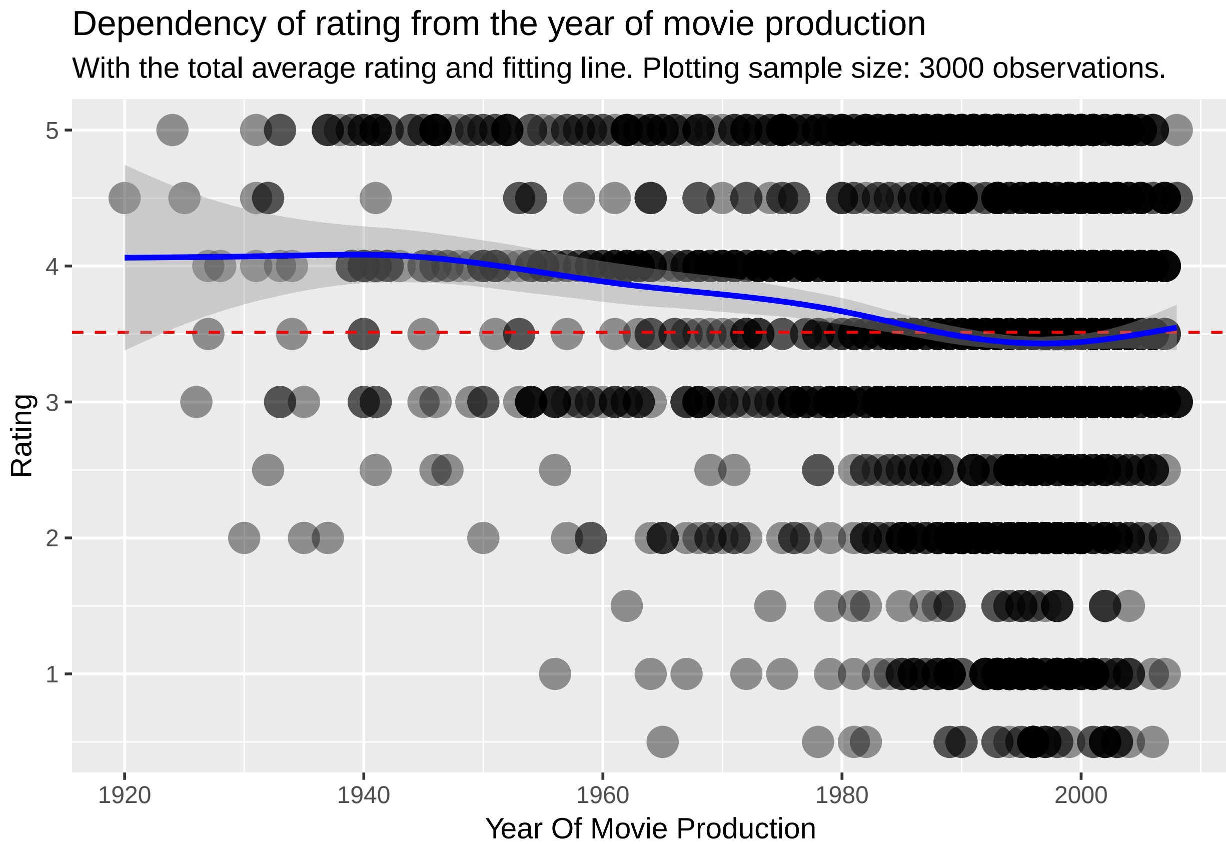
We will use the factor representation during the classification and numeric representation during the regression.



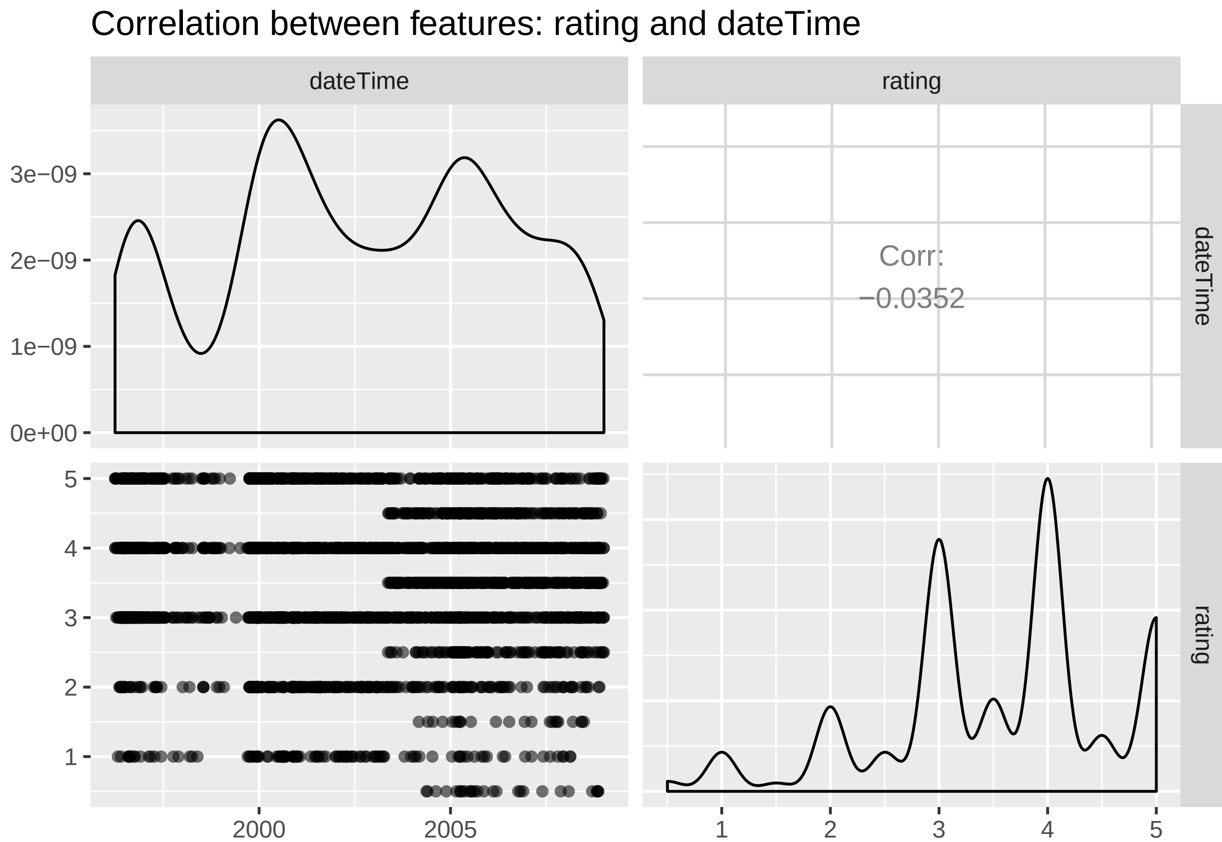

## Initial correlations

The initial analysis demonstrates sufficient correlation between rating and movie production year.

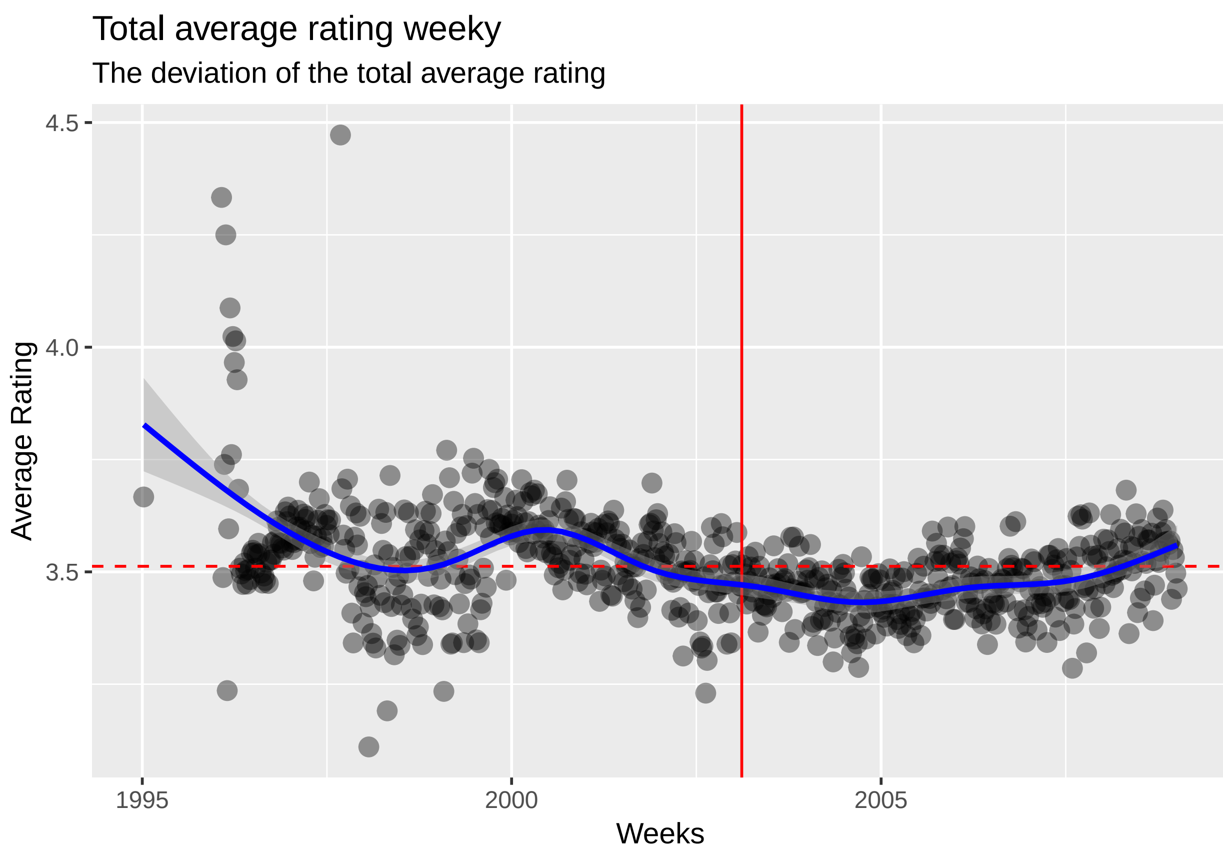

Correlation of features: rating and movie production year
rating
movieYear
Corr:
−0.144
rating
movieYear
0.0
0.2
0.4
0.6
0.8
1920
1940
1960
1980
2000
1
2
3
4
5
1920
1940
1960
1980
2000




Also, the data demonstates sufficient correlation between rating and timeline.



Correlation between features: rating and dateTime
dateTime
rating
3e−09
2e−09
1e−09
0e+00
Corr:
−0.0352
dateTime
5
4
3
2
1
rating
2000
2005
1
2
3
4
5




Initial correlations work well for the regression model. However, after multiple unsuccessful attempts of building a classification prediction system, we should notice, that initial correlations can't be directly applied to the classification. Since the classification system depends on the user behaviour much more than movie averages, we will build a model based on the genres component of the movies.

## The model

As stated above, we will build the prediction model based on the genres $G$, movie bias $b_m$ and user bias $b_u$:

$$R = f(M(G), U(G), b_m, b_u) \tag{4}$$

Since the movie genres can contain any combination of twenty unique genres, one can create the list of features based on the list of possible genres `G`:

$$G = [g_1, \ldots, g_{20}] \tag{5}$$

| id | genre |
|---|---|
| 1 | Comedy |
| 2 | Romance |
| 3 | Action |
| 4 | Crime |
| 5 | Thriller |
| 6 | Drama |

| id | genre |
|---|---|
| 7 | Sci-Fi |
| 8 | Adventure |
| 9 | Children |
| 10 | Fantasy |
| 11 | War |
| 12 | Animation |
| 13 | Musical |
| 14 | Western |
| 15 | Mystery |
| 16 | Film-Noir |
| 17 | Horror |
| 18 | Documentary |
| 19 | IMAX |
| 20 | (no genres listed) |

Since a movie does not have a numerical representation of the "amount of single genre in it", we can represent the movie genre content as a binary vector mask.

For example, the movie with ID 6 contains:

| movieId | genres |
|---|---|
| 6 | Action\|Crime\|Thriller |

We define the sympathy vector $S$ of the movie $m_i$:

$$s^{m_i} = G^{m_i} \tag{6}$$

Therefore, the movie with ID 6 with only three of possible twenty genres will obtain the sympathy:

$$s^{m_6} = [0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0] \tag{7}$$

The users rate movies and therefore make not only qualitative, but also quantitive grading of the genre features. We will call the sympathy of the user $u_i$ to the movie $m_j$ as the scalar multiplication of the rating value $r^{u_i,m_j}$ to vector of movie sympathy:

$$s^{u_i,m_j} = r^{u_i,m_j} \cdot s^{m_j} \tag{8}$$

For example, the user with ID 144 graded the movie with ID 6 by rating value 4.5:

$$\begin{aligned} s^{m_6,u_144} &= 4.5 \cdot [0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0] && (9) \\ &= [0, 0, 4.5, 4.5, 4.5, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0] && (10) \end{aligned}$$

| userId | movieId | rating | timestamp | title | genres |
|---|---|---|---|---|---|
| 144 | 6 | 4.5 | 1084241814 | Heat (1995) | Action\|Crime\|Thriller |

| **db** |
| --- |
| id : Character |
| movieId : Character |
| userId : Character |
| rating : Factor |
| ratingNumeric : Numeric |
| halfRatingPossible : Logical |
| movieAvgRating : Numeric |
| userAvgRating : Numeric |
| sympathy : Numeric |
| train: Logical |

Figure 3: MovieLens data after transformation

The sum of the feature values for all the movies graded by single user, creates the genre profile of the user:

$$S^{u_i} = \sum_{r^{u_i}} s \tag{11}$$

Sympathy vector spreads along twenty features, but describes the affection of a single user to a singe movie. Since all the twenty features describe one rating, we can shrink the sympathy vector to the single cumulative value. Finally, it is important to notice, different users have different number of ratings. To normalize the sympathy value, we have two choices:

i) extend the model with new feature `numberOfRatings`, which represents the total number of ratings made by user $N^{u_i}$, or
ii) combine number of ratings with the sympathy and make the feature with value equal to the average sympathy to the movie.

The second choice decreases the number of features to single one. To have a single feature instead of two, gets us the possibility of a significant increase in the size of training data population during the fitting. In order to decrease the number of indexes in the equation, let's write down the cumulative average sympathy $A$ around the rating index $r_i$:

$$A^{u_i}_{r_i} = \frac{1}{N_{u_i}} S^{u_i} \tag{12}$$

Finally, our model becomes:

$$r_i^{u,m} = A^{u,m} + b_u + b_m + \varepsilon_i \tag{13}$$

# The Data Transformation

The transformation of the `movieLens` dataset is the process of mutation the initial data set to the form of model.

The final data set should consist of the number of features defined by model and will be easily splittable into training and test subsets (Fig.3).

The final data set consists of logic vectors `halfRatingPossible` and `train` for the data subsetting before fitting.

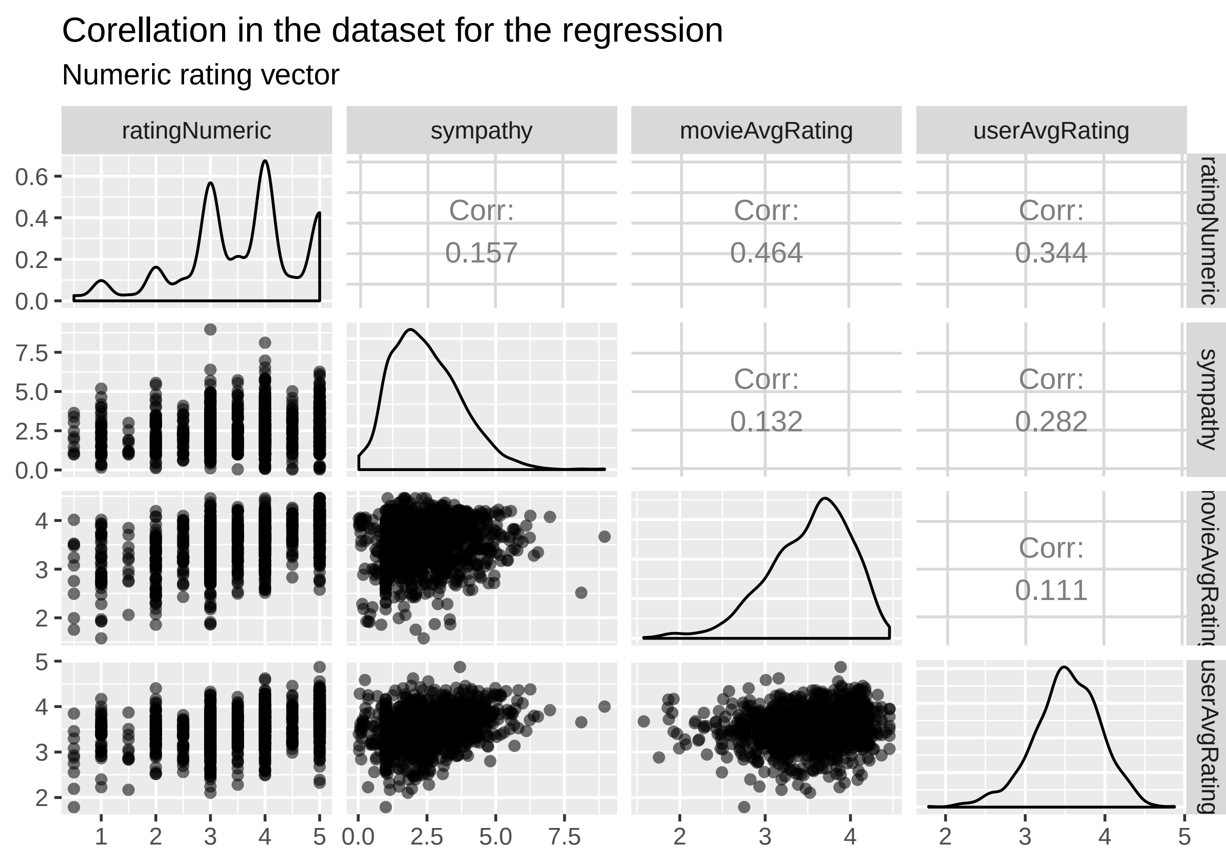
The logic vector `train` is responsible for splitting the data into training and testing sets.

The logic vector `halfRatingPossible` is responsible for splitting the training and testing sets into two additional subsets during the process of classification.

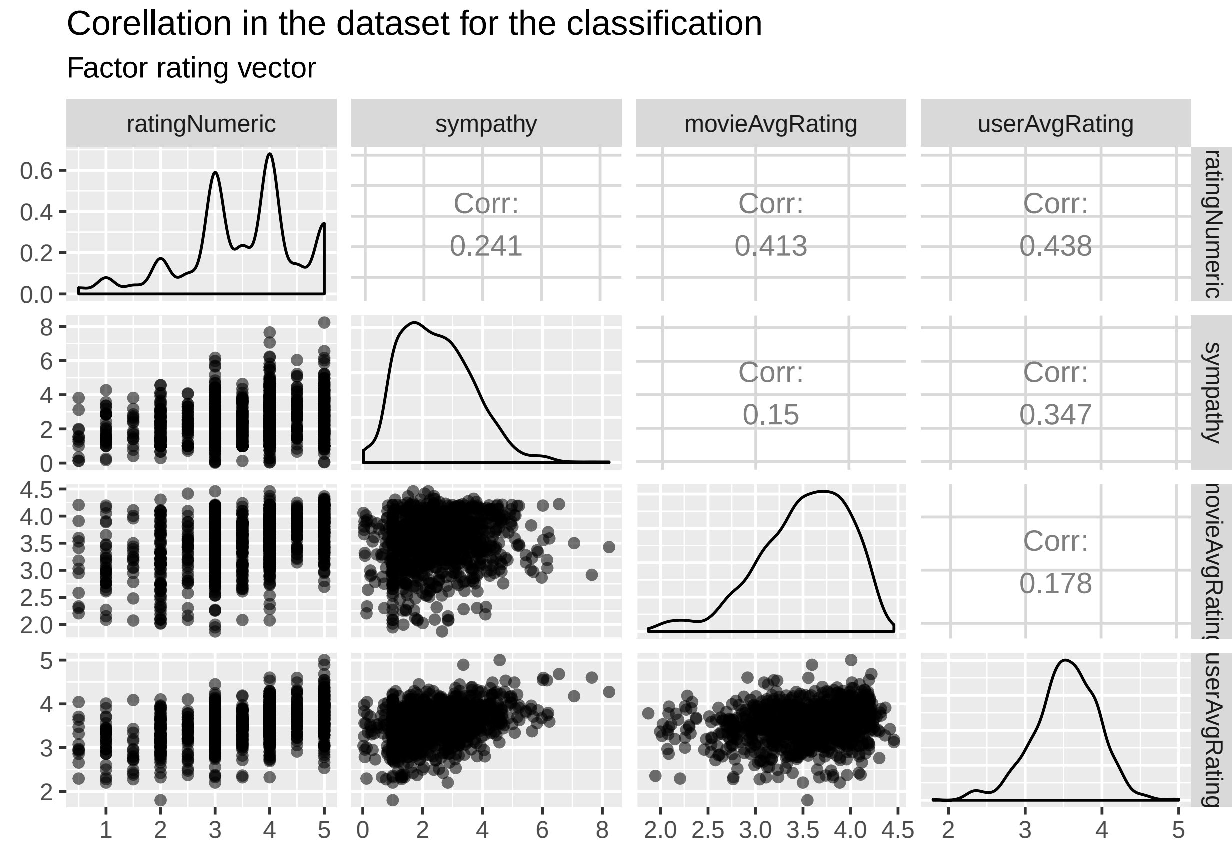
The final data set consists of vectors `rating` and `ratingNumeric`. These vectors have the same values in two different form. The numeric values of `ratingNumeric` is for regression model, but factor `rating` is for the classification.

```
## halfRatingPossible :  logi [1:10000054] FALSE FALSE FALSE TRUE FALSE TRUE ...
## movieAvgRating :  num [1:10000054] 2.86 2.96 3.39 3.5 4.16 ...
## rating :  Factor w/ 10 levels "0.5","1","1.5",..: 10 10 10 7 10 8 8 8 8 8 ...
## ratingNumeric :  num [1:10000054] 5 5 5 3.5 5 4 4 4 4 4 ...
## sympathy :  num [1:10000054] 2.14 3.27 4.34 1 3.21 ...
## train :  logi [1:10000054] TRUE TRUE TRUE TRUE TRUE FALSE ...
## userAvgRating :  num [1:10000054] 5 5 4.03 3.4 3.29 ...
```

The correlations of the data for the regression model:




The correlations of the data for the classification model:




## Classification

The classification should be performed for two types of rating systems:

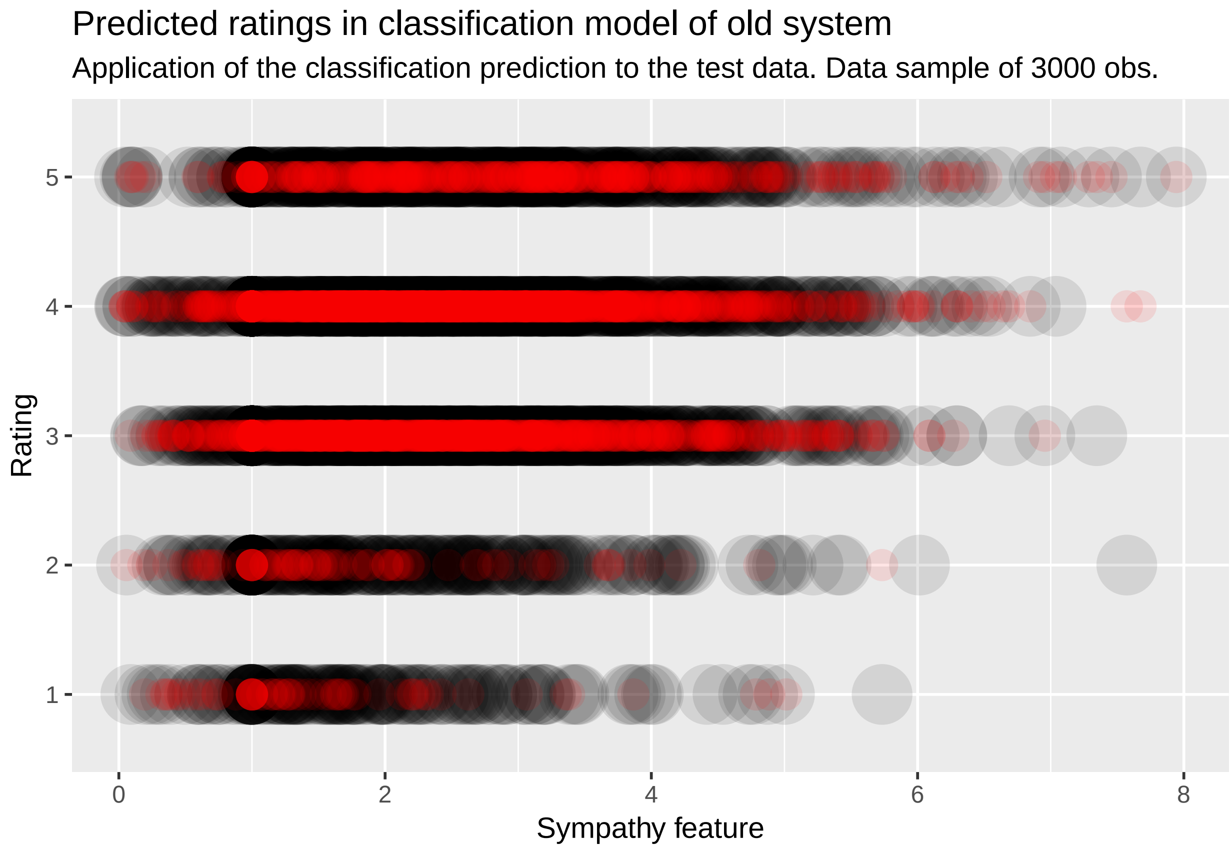
(a) old one, with the rating levels 1, 2, 3, 4, and 5, and
(b) new one, with the levels from 0.5 to 5 with the step 0.5.

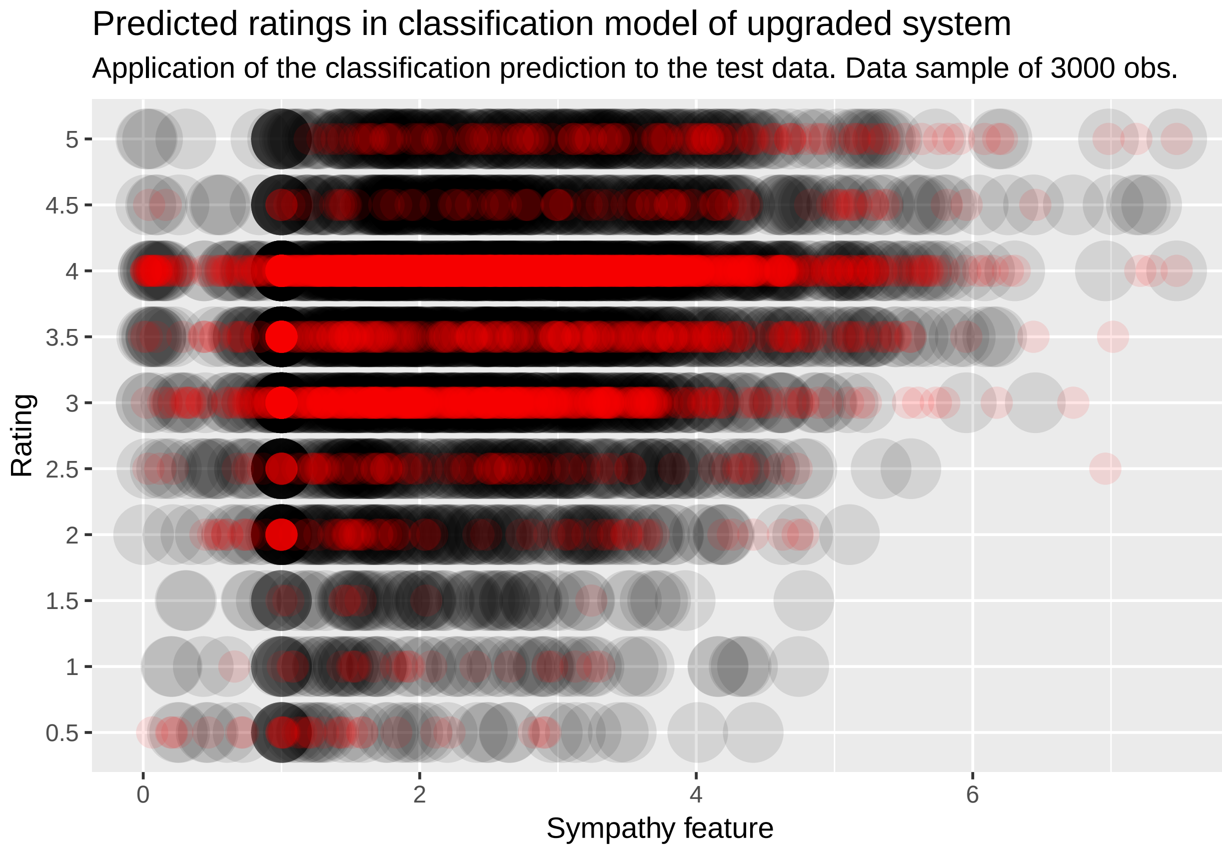
For the purpose of name simplification we will call them "Wholes", and "Halfs".

For the classification we will use two random decision forests models:

(a) "randomForest" v:4.6-14 by Leo Breiman, and
(b) "Rborist" v:0.1-17 by Mark Seligman.

With the Rborist package, on "halfs" subset we achieved minimum value or RMSE = 0.94.


Predicted ratings in classification model of old system
Application of the classification prediction to the test data. Data sample of 3000 obs.
Rating
5
4
3
2
1
0
2
4
6
8
Sympathy feature




## Regression

For the linear regression model there is no difference between the groups of "wholes" and "halfs" in the dataset. The linear regression model interpolates the rating values and produces rating values outside of the system rating possibilities. For example, linear regression model may produce the ratings $< 0.5$ for users who grade movies with small ratings and $> 5.0$ for users with big number of "5" ratings.

However, the linear regression model produces very small value of target paramenter RMSE $= 0.871546$.


## The results of classification and regression

The combined table of results:

| Type | Library | Sample.size | RMSE | Notes |
|---|---|---|---|---|
| classification | randomForest | 500000 | 1.0616242 | 5 ratings set; nTree = 200 |
| classification | randomForest | 500000 | 0.9719761 | 10 ratings set; nTree = 200 |
| classification | Rborist | 1000000 | 1.0281959 | 5 ratings set |
| classification | Rborist | 1000000 | 0.9394816 | 10 ratings set |
| regression | lm | 9000055 | 0.8715465 | full population |

As one can see, the classification models do not produce the $RMSE$ less than 0.9 even for the train population 1 000 000 observations, but we can visually review the accuracy of classification in comparison to regression and notice the difference.

## Conclusion

During the project we applied various methods of data analysis and transformation to the MovieLens dataset.

We found the way to build the list of the most correlated features.

We performed the classification and regression for the data.

Finally, we achieved the main target of the project and built the prediction system with the $RMSE$ less than 0.8775.