COMP 4442

Dillon Dugan

03/12/2021

Predicting User Score using Random Forest Methods

ABSTRACT

This project explores the use of Random Tree Methods to estimate the user score (0.0-10.0) rating of major PC video games from the online distributor, Steam. The Steam database as many observations of all games on its platform, including the names, price, number of owners, average play time, etc. These variables are going to be the main driving force for the regression models created by the Random Tree Methods. This project also explores the importance of each variable for the decision trees that make the final model, most importantly might be the actual reviews on Steam, which are encoded as number of positive/negative rather than the 0-10 scale that user score has. Running the randomForest and ranger function from the randomForest package, with user score as the outcome variable and all other variables as explanatory variables, both models seem to accurately guess the user score within the mean squared error of 0.89. The most important variables were the Price, number of owners, and the average playtime of the game. Overall, the random forest methods seem appropriate to predicting user score through regression decision trees, however, with more accurate predictive variables, the estimate could yield better results.

DATA SOURCES

The data for this project was sourced from public records of Steam data uploaded to Kaggle and GitHub, however it may be traced directly back to Steam databases available publicly through

website API. The first dataset downloaded directly from the GitHub link provided much of the details of every game on Steam while a second database was used to obtain the percentage of positive/negative reviews for each game on Steam. The final dataset is a Metacritic dataset, also publicly available through website API, that gives the user score and meta score of a large sum of the games featured on Steam.

Ultimately, all three data sources were inner joined on each other by the name of the game. This final dataset was clean-up for duplicate variables and missing content as the randomForest package works best with complete data, resulting in 1150 rows with 12 variables: game, release_date, price, owners, developer and publisher, average_playtime, metascore, genre, percentage of positive and negative reviews, and the actual user score of each game. The price, owners, average_playtime, metascore, and percentage of positive/negative users were continuous explanatory variables and the rest were categorical, however, random forest methods can't handle more than 56 factors so there was not a conversion of character to factor. There were multiple instances of Warner Brothers published games being listed as WB or Warner Bros., so the data needed to combine the both.

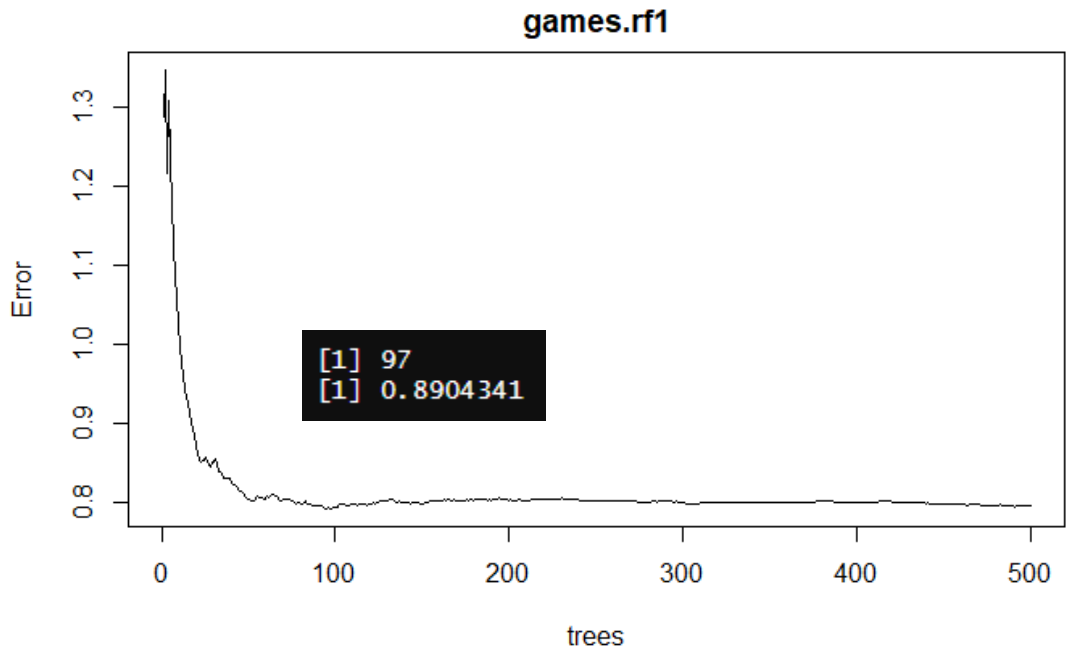
EXPLORATORY DATA ANALYSIS

There wasn't much exploratory data analysis performed other than getting the dataset in a usable form for the randomForest methods. There was a creation of a split 70%/30% training and testing dataset to test the randomForest methods with. However, a validation set, while optional for randomForest methods, was not created as the Out-of-Bag sets would give the cross-validation necessary to test the models' predictive accuracy.

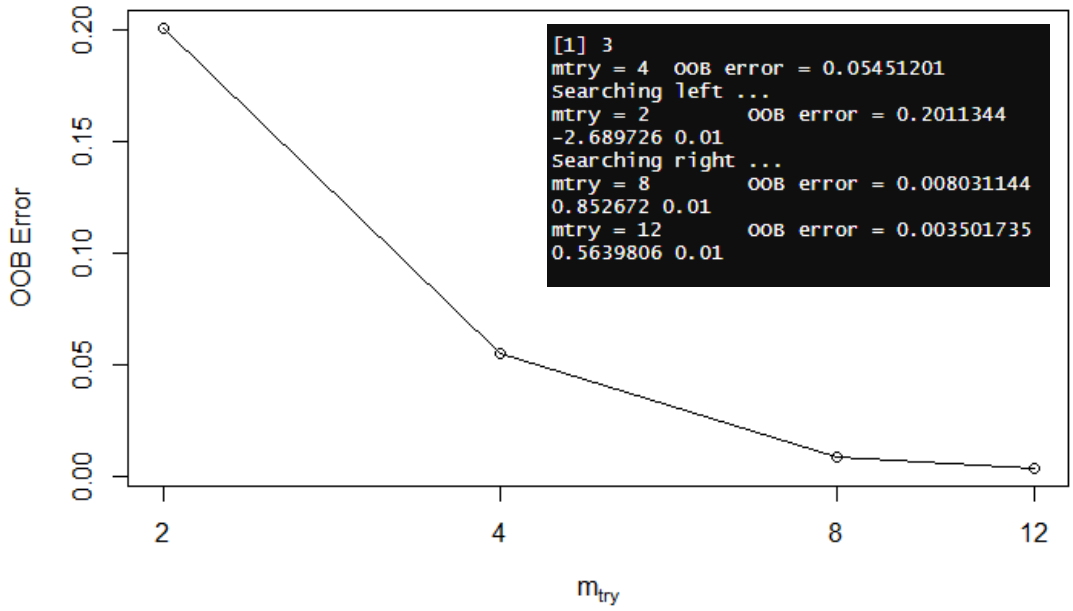
RANDOM FOREST METHODS

The Random Forest methods used for this project were randomForest and ranger. Even though both can output the same predictive power, ranger tends to perform much quicker on the dataset because it takes random selections from the given training set directly rather than taking from a set created by the weights of the original training set. Both functions still create the model by testing the training dataset against a selective number of given variables to create a certain number of decision trees to average into a final model. By default, both functions will use 500 decision trees and will use the number of given variables divided by 3 as the "mtry" parameter. The mtry parameter is how many variables will be tested at each split of the tree. However, using the plot of a basic randomForest model and the TuneRF function from the randomForest package, this dataset seemed to perform best at 97 decision trees and as mtry increased, the error rate continued to decrease. So, for the ranger and randomForest function, the number of trees was changed to 100, as anything above it wouldn't increase the predictive power of the model, and mtry was set to the number of variables in the dataset. These models were then tested with the testing dataset in the predict function. Comparing the actual user scores to the predicted user scores, both models were accurate at predicting the user score based of the given variables. Both models demonstrated that the price, the number of owners, and the average play time of the game was most important for determining a regression model to predict user score.

VISUALIZATIONS

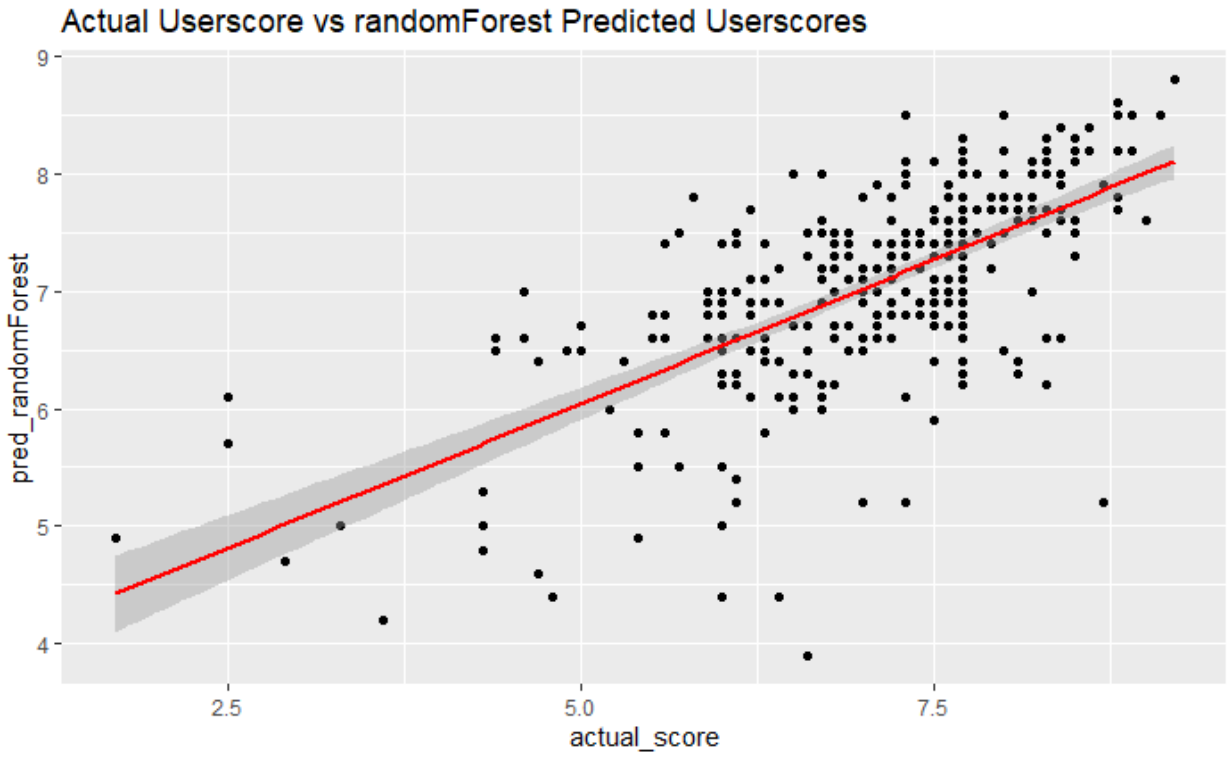


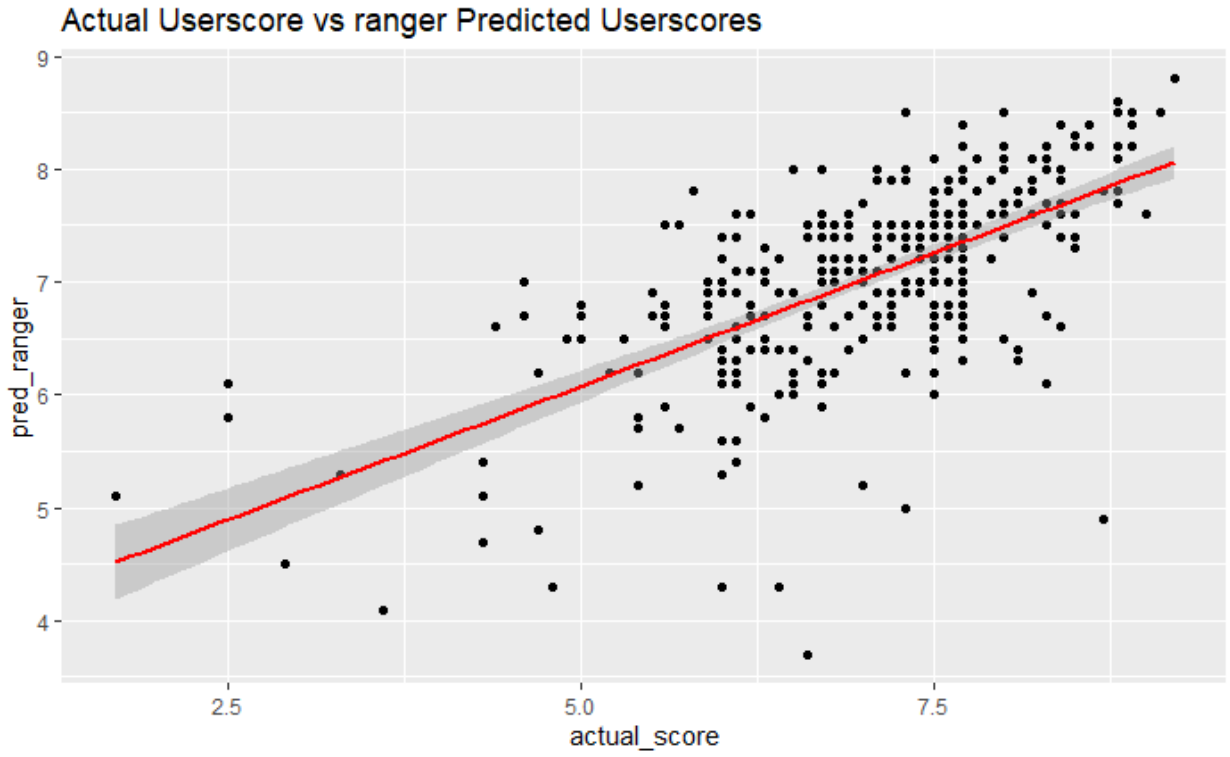
Plot of Basic randomForest model



Plot of TuneRF for tuning of mtry

VISUALIZATIONS (cont.)





SOURCES

- **https://towardsdatascience.com/analyzing-video-games-data-in-r-1afad7122aab**
- **https://raw.githubusercontent.com/rfordatascience/tidytuesday/master/data/2019/2019-07-30/video_games.csv**
- **https://www.kaggle.com/trolukovich/steam-games-complete-dataset**
- **https://www.kaggle.com/skateddu/metacritic-games-stats-20112019**
- **https://towardsdatascience.com/random-forest-explained-7eae084f3ebe**
- **https://uc-r.github.io/random_forests**
- **http://uc-r.github.io/regression_trees#bag**
- **https://miamioh.instructure.com/courses/38953/pages/random-forest**
- **https://www.stat.berkeley.edu/~breiman/randomforest2001.pdf**