by Xiaohua Liang and Siyu Yang
{xiaohual, siyu}@stanford.edu

PC game play time estimation based on Steam data and reviews

Summary

The Steam platform is the largest PC game online distributor in the world, and has accumulated a vast amount of player and game data.

In this project, our goal is to predict the average total playtime of a game based on its metadata (e.g. genre, price and publisher) and reviews (votes up and review text).

Our best model was a random forest using the mean absolute error as criterion and logarithm of the playtime as output. This model was able to predict the average total playtime to within an error of 50% for 62.8% of all the games in our test set. Although certain 2-grams and 3-grams were among the most informative features, adding text-based features did not improve the overall performance significantly.

Data

Data Source:
- Official Steam web APIs
- Third-party Steam Spy web APIs
  - Steam Spy gives an estimate of average total playtime by sampling players' playtime information, which can be inaccurate. This was a problem for prediction as our 'ground truth' could be inaccurate.

Processing: Filtered out games
- published after August 2017
- with N/A fields
- that cannot be played in single-player mode

Number of data points: **9334 games**

Features

Feature (256 columns)	Processing
category	one-hot encoded with an 'others' column
publisher	
genre	
initial price	raw input (unscaled)
number of up-votes	
up- to down-votes ratio	
number of owners	
achievements to unlock	
review text features*	weighted frequency

* Text features
- After tokenizing the sentences, lemmatizing and removing stop words in there, we hand-picked 62 popular 1-grams, 2-grams and 3-grams that we thought indicate long playtime.
- These are the count of this n-grams in a game's reviews divided by total count of 1-, 2- and 3- grams in these reviews.

Models

Linear Regression
- Because of the large skew in output feature, negative playtime were predicted for games with short playtime

Gradient Boosting Regression
- Resistance to outliers did not suit the distribution of y

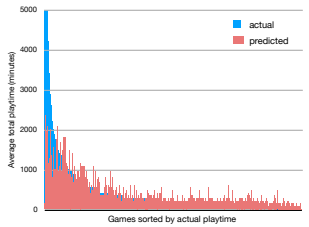
Random Forest Regression
- Used OOB score to tune the hyper parameters
- Most informative features: price, owners, up-votes, achievements to unlock, votes ratio, genre (Indie), 1-gram ('minute'), genre (Strategy), genre (Action), genre (RPG), 1-gram ('waste'), publisher (others), 2-gram ('10 10')

Discussion

- Our model predicts the most accurately when the playtime is around the medium (~250 minutes), and less well around extreme values
- Predicts well for major studio titles since prolific publishers would be included as a column in features

Results

7467 training samples, 1867 test samples



- Because the distribution of playtime is so skewed, we decided to use **log(playtime) as output feature** and the mean absolute loss to prevent few games with very long playtime to dominate the cost function
- On the right, we present the result in terms of percentage of games with less than x% error because a one-number summary of performance will be misleading, with such a skewed output variable

Results - Experimenting with Models and Features

Model	games with more than 100% error		games with more than 50% error		games with more than 30% error	
Data set	test	train	test	train	test	train
Predicting average	55%	47%	80%	75%	87%	85%
Predicting medium	18%	19%	47%	48%	62%	64%
RF (MSE)	26%	22%	48%	38%	63%	55%
RF (MAE)	18%	14%	40%	29%	58%	46%
RF (MSE), log(y)	13%	9%	38%	26%	57%	44%
RF (MAE), log (y)	12%	9%	37%	27%	56%	44%
RF (MAE), log (y), text	12%	9%	36%	26%	56%	44%
GB (least sq), log(y)	14%	15%	40%	40%	59%	59%

Results for Some Well-Known Games

game	actual (minutes)	predicted (minutes)	percentage error
age of empires ii hd	2791	2001	28%
wwe 2k16	1921	1576	18%
watch_dogs® 2	1632	1194	27%
assassin's creed® unity	1613	1452	10%
watch_dogs™	1596	1273	20%
final fantasy ix	1390	1109	20%
far cry® 2: fortune's edition	584	543	7%

Future

Taking log on the output feature improved performance, but our model still under-predicts for games with high playtime - can additional features or adding model flexibility help with this shortcoming?

References

Thomas, N Alexander. PyDate Vocab Analysis. 2017. GitHub repository, https://github.com/alexander-n-thomas/pydata-vocab-analysis