Dark Souls? CS GO? What's the big deal?

Trinity Ung, Audrey Wei, Cameron Morgan

2022-03-13

Section 1 - Introduction

In the past decade, the video game industry has been under huge development with the advancement of programming and design technology. As a result, it is important to also review the development in consumption of video game products to understand the factors that have led to changes in consumerism related to video games. In our project, we will be analyzing data on video game characteristics and popularity for games featured on the international video game distribution service, Steam, in order to answer the question: What factors make a video game popular?

In analyzing our data and exploring our question of interest, we will be defining our variables as indicated in the codebook provided below. To define our main response variable of "popularity", we will be quantifying it as the proportion of positive ratings for a game, where a proportion close to 1 indicates higher popularity and one close to 0 indicates low popularity. In order to test potential factors that could explain trends in our response variable, we will analyze the following possible explanatory variables: price, genre, category, and number of in-game achievements of the games. The price is indicated as the price of a game in US dollars. The genre will be split into seven main genres which are action, role playing, simulation, adventure & casual, sports & racing, strategy, and multi-genre where multi-genre is any combination of any number of the other main genres. The category is split into three main categories being multi-player, single-player, or multi-category in which the game supports both multi-player and single-player. The number of in-game achievements are split into three categories being that the game either has no achievements, has 1 - 100 achievements or has more than 100 achievements. In doing this, we hope to be able to identify trends in each variable that point to specific factors that should be included or considered when trying to produce a popular video game.

To carry out this analysis, we will be using data from a data set called, "Steam Store Games (Clean Dataset)", published on the data archive website, Kaggle, and created by user, Nik Davis. In his data set, Nik Davis, through his meticulous collection and organization of data on individual game statistics available in Steam from the Steam Store and SteamSpy APIs provided by Valve, Steam's parent company, started collecting the data from scratch as a part of his first large data science project. With his project, he also intended to formulate an understanding of factors that affect video game sales, play-time, and ratings similar to what we will be analyzing in our project. The original resulting data set contains 18 variables with each observation representing a single video game distributed on Steam.

Before starting our modeling process, we had to modify our data set to fit within more reasonable margins of comparison. Since our data set included over 27,000 observations, we needed to figure out a way to cut down on the number of observations. We decided on filtering the data set by the number of owners and total ratings for each game.

We wanted to filter the data frame in order to only include games that have a minimum number of owners starting from the range of 5000-10000 so that the game had a reasonable amount of potential for fair analysis of ratings among a large population. Since our model is based on the proportion of positive ratings, we would have to do another filter based on the number of ratings to ensure that the proportion of positive ratings in each observation would not be based on a small collection of total ratings that could skew the proportion. We decided to filter out any observation that has less than 500 ratings. Eventually, we are left with 1068 observations in the data frame to do the analysis, which would be a pretty reasonable size of the data frame to analyze and build our models on.

Since there were a wide variety of combinations of genres, categories, and achievements, we decided to condense these combinations of variables. For genres, we condensed them into six main genres and for categories and achievements, we condensed them each into three categories.

Codebook

Name of Variable	Description of Variable
appid	A categorical variable that presents a unique numerical ID for each game.
name	A categorical variable that shows the name of each game.
release_date	A categorical variable that represents the date in the form YYYY-MM-DD that the game was released.
english	A categorical variable that takes 1 if the game supports English.
developer	A categorical variable that shows the names of the developers. A semicolon represents multiple developers.
publisher	A categorical variable that shows the names of the publishers. A semicolon represents multiple publishers.
platforms	A categorical variable that shows which platforms the game can run on. A semicolon represents multiple platforms.
required_age	A categorical variable that takes a numeric value and shows the minimum age needed to access the game. A value of 0 represents an unrated game.
categories	A categorical variable that shows the different categories of a game. A semicolon separates multiple categories.
genres	A categorical variable that shows the different genres of a game. A semicolon separates multiple genres.
simplified_genre	A categorical variable that rewrites the longer list of genres that includes combinations of genres into six overarching genres.
steamspy_tags	A categorical variable that is similar to genres and represents a certain aspect of the game but are created by the community instead.
achievements	A numerical variable that represents the amount of achievements that can be obtained by the player.
positive_ratings	A numerical variable that represents the amount of positive ratings a game has received.
negative_ratings	A numerical variable that represents the amount of negative ratings a game has received.

Name of Variable	Description of Variable
total_ratings	A numerical variable gives the sum of the total number of ratings for a game based on adding the number of positive and negative ratings.
prop_pos_ratings	A numerical variable that shows the proportion of positive ratings among the total ratings for a game.
average_playtime	A numerical variable that shows the average playtime of a game in hours.
median_playtime	A numerical variable that shows the median playtime of a game in hours.
owners	A categorical variable that represents an estimated range of the number of owners a game has.
price	A numerical variable that represents the price of a game in USD.
age_requirement	A categorical variable that gives a yes or no output for if the game has an age requirement or not
simplified_category	A categorical variable that rewrites the longer list and combinations of categories into four simple categories
simplified_achievements	A categircal variable that rewrites the numerical variable of achievements to give an output of "None" for 0 in-game achievements, "<= 100" for equal to or less than 100 in-game achievements, and "> 100" for greater than 100 achievements.

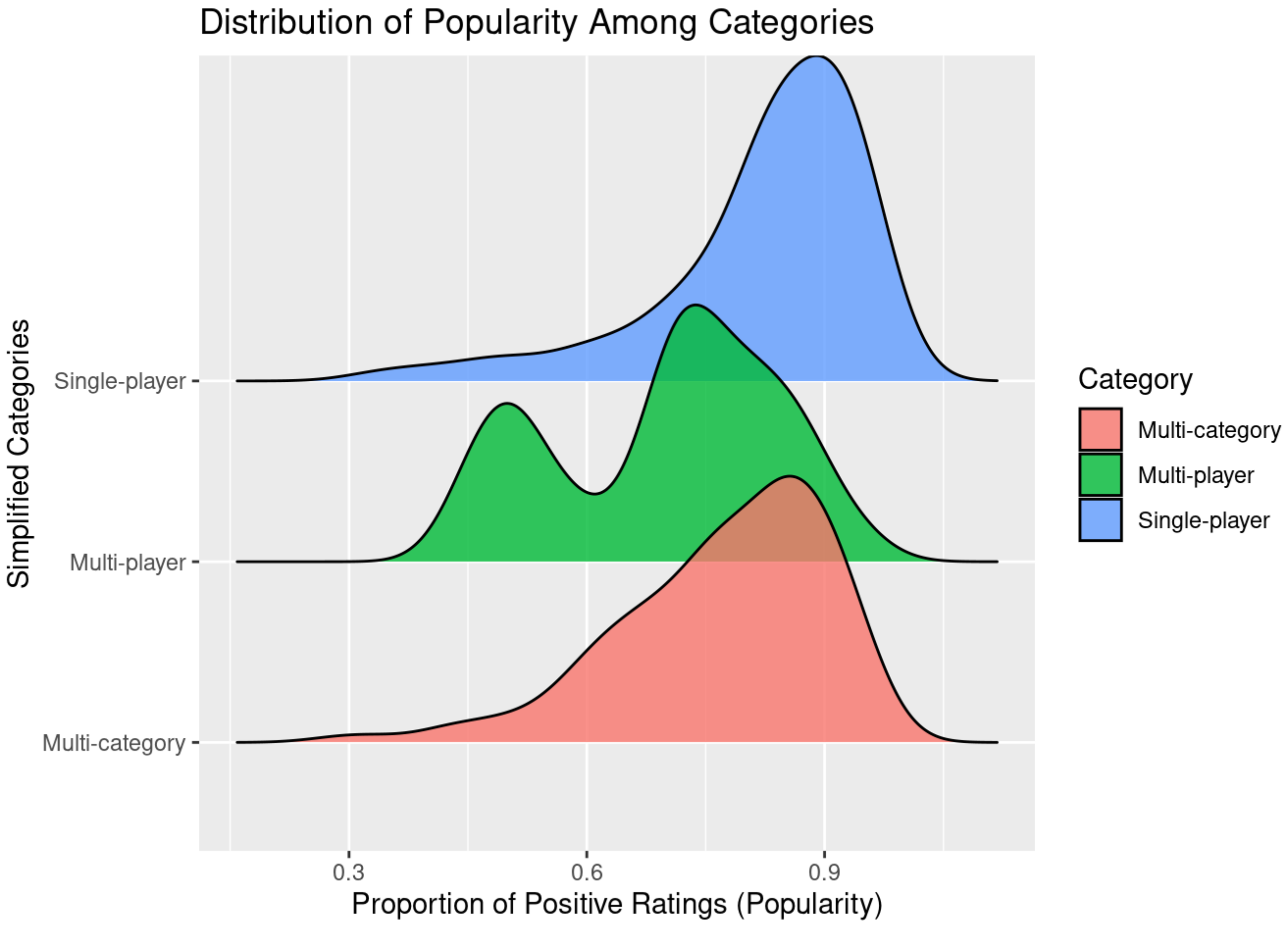
Section 2 - Model Building

In this section, we will be proposing three separate models in an attempt to explain the proportion of positive ratings of games. One model will use the explanatory variables `price` , `simplified_categories` , and `simplified_genres` . The second model will be a main fit model using `price` , `simplfied_categories` , `simplified_achievements` . The last model will be an interactive fit model using `price` , `simplified_categories` , and `simplified_achievements` . For each of these models, we created a testing split and a training split using a 0.50 proportion.

Model proposed by Trinity

When trying to decide if there is a difference between the distribution of "popularity" between different game categories, I used a density ridges plot in order to visualize the spread for each. Upon first plotting the distribution, all of the density ridges plots had a mostly left-skew. In order to rearrange these distributions to lessen the skew without altering the order of data points, I took a log of base 10 for the proportion of positive ratings variable. The result still came out as fairly left-skewed and held its general pattern for each category, so I decided not to include the log transformation in my analysis in order to make reading the x-variable easier to comprehend.

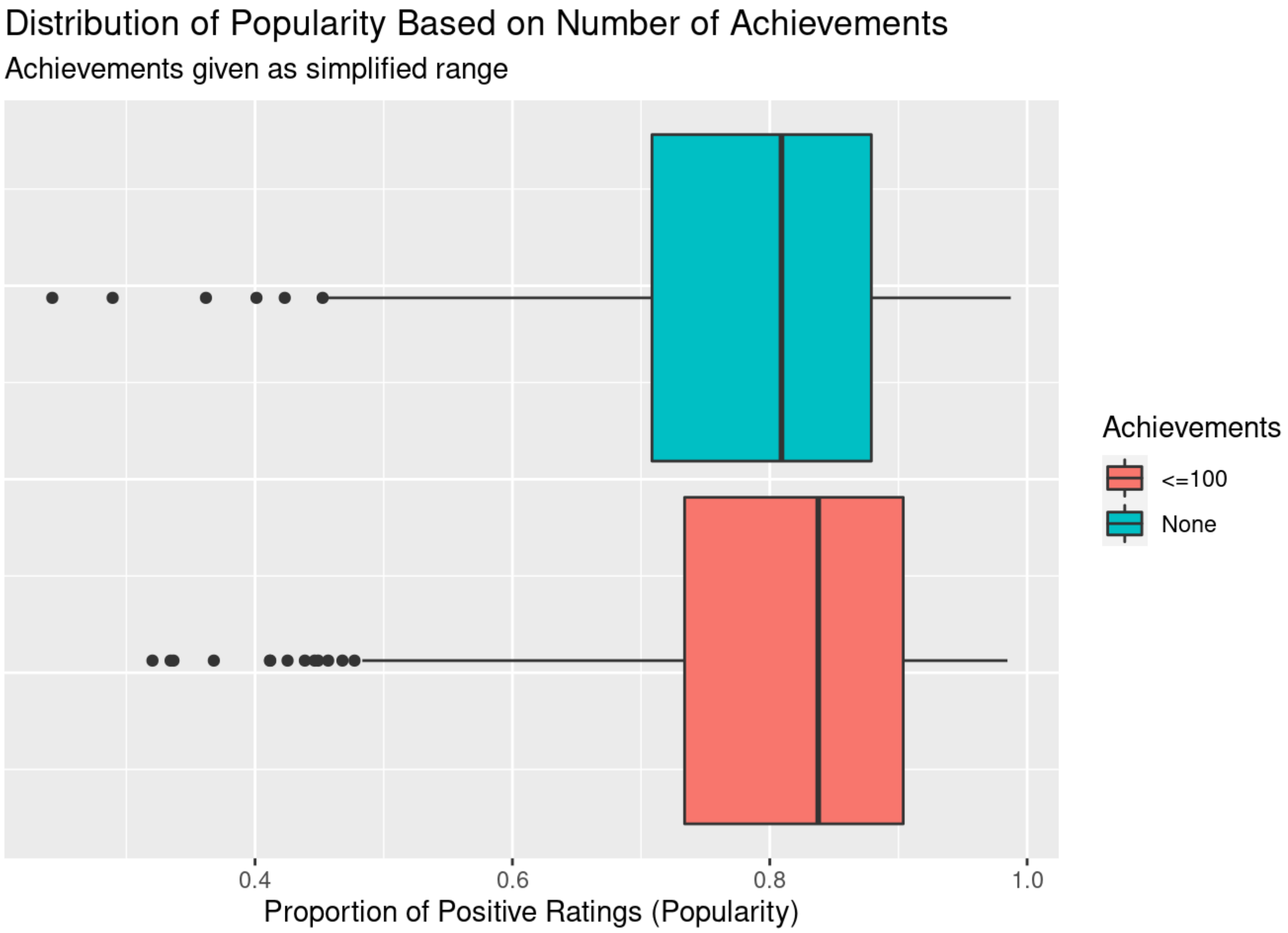
Comparing each density ridges plot for the different `simplified_categories` , all of the categories had a majority of their proportion of positive ratings above 0.50 or 50%, indicating an overall majority of positive ratings for all categories. The single-player category had the highest distribution of games with a 75% or greater proportion of positive ratings. The multi-category category was similar but with a little more spread into lower proportions as shown by the increased width of this plot towards the lower values when compared to the single-player category. However, one plot that stuck out to me was the bimodal and least skewed distribution of the multi-player category. Both the single-player and multi-category categories had a fairly strong unimodal, left-skew, but the multi-player category had a bimodal distribution that was centered further toward the lower ends of “popularity” where its higher peak was around 0.75 (75%) and its lower peak around 0.50 (50%). This was one indication that there may be a difference in popularity when comparing multi-player games against other categories of games.



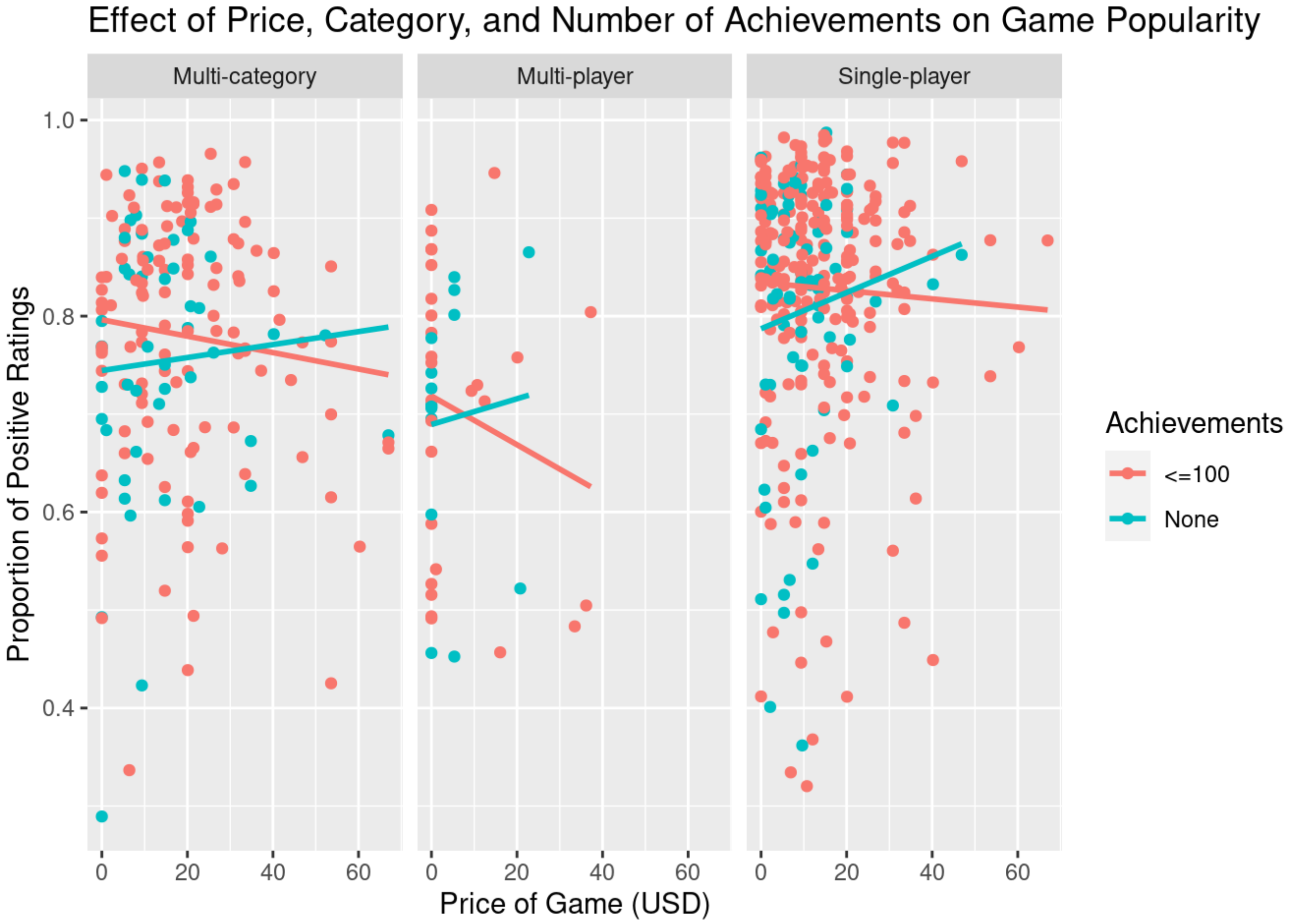
Next, I wanted to include another categorical variable that could help specify the types of in-game characteristics that a video game should include to increase its probability for higher popularity. Upon consideration and various plotting of distributions for a variety of possible categorical variables including age requirement and release month, I decided upon using the categorical variable, `simplified_achievements` , in order to try to further specify the type of game that would likely bring about more popularity. There are many types of players that care about completing in-game achievements such as speedrunners, “completionist” gamers, and any hard-core fans of a series. Many types of achievements also tap into the psychological need to “check the box” off a goal list in order to feel a sense of accomplishment and status within a gaming community. Other more casual achievements allow players to win rewards or prizes that they would otherwise need to pay for through in-game microtransactions. It is through these mechanisms of reeling consumers’ attentions and drive towards a game that make in-game achievements a potentially beneficial characteristic to improving popularity.

In general, single-player games tend to have more in-game achievements than multi-player games. Because of this, I hypothesized that many single-player games that included in-game achievements might show a greater distribution of higher popularity than, say, multi-player games with no achievements. Especially since my earlier exploration plot above demonstrated single-player games having the highest and narrowest distribution of popularity with a center that was greater than any other category, I further expected single-player games that had in-game achievements to have a greater proportion of popularity than any other combination of category and in-game achievements.

Upon plotting this relationship, it seemed like the overall median and 50 percentile of both data sets for games with less than or equal to 100 in-game achievements vs. games with no in-game achievements had a small difference between them. The majority of both of their 50 percentile boxes in the boxplot showed overlapped with each other. Their median values were slightly different in which games with less than or equal to 100 in-game achievements had a higher median proportion of positive ratings than games without achievements. Both distributions had their 50 percentile box well above 50% for the proportion of positive ratings, which should be expected according to the skew of the distributions shown for the game categories in the density ridges plot. Overall, there did not seem to be an overwhelmingly large difference between games that did or did not have achievements in terms of popularity, but it was the comparison, among the other potential categorical variables (as stated above), that had the greatest difference in popularity between the possible categories, which is why I decided to use this as my other categorical explanatory variable.



Based on the these choices of explanatory variables, I decided to create a model relating these categorical variables with `price` in order to predict the `prop_pos_ratings` for a game. Upon first plotting the relationships, there seemed to be quite a large spread of data points with a very unclear trend. Using the geom_smooth function yielded some linear trend lines that showed potential relationships between the data for each `simlified_category` and the `simplified_achievement` type. Upon preliminary analysis (before forming a fitted linear model), the trend lines seemed to show some consistency in direction when comparing the `simplified_achievements` types. For games that had less than or equal to 100 achievements, the overall direction of all trend lines across the same range of `simplified_achievements` was in the negative y-direction. This seemed to indicate that as any game, no matter the `simplified_category` , with less than or equal to 100 achievements increased in price, the overall popularity of the game decreased as a result. For games that had no achievements, the trend lines seemed to generally increase across all `simplified_categories` as the price increased.



After my preliminary data analysis, I decided to try an interactions effect model in order to model the relationships I was observing above. While the relative steepness of the slopes compared across the different categories seemed to be somewhat similar when holding the achievement category constant, the direction of each trend line within each simplified category was opposite of the other when also considering the `simplified_achievements` distinction Thus, I thought that when calculating different slopes for different ranges of achievements for each category, it may be inappropriate to assume the same slope for each trend line as there was a clear opposition in the direction for each. Using an interactions effect model might, in addition, accommodate some of the slight differences in slope steepness when comparing against categories, for example, incorporating the greater negative slope in the multi-player category for achievements less than or equal to 100. For these reasons, I decided to test an interactions fit model as shown below:

```
## # A tibble: 12 × 5
##    term                                 estimate std.error statistic   p.value
##    <chr>                                   <dbl>     <dbl>     <dbl>     <dbl>
##  1 (Intercept)                           7.96e-1  0.0195      40.9   8.10e-163
##  2 price                                -8.39e-4  0.000781    -1.07  2.84e-  1
##  3 simplified_categoriesMulti-player    -7.78e-2  0.0350      -2.22  2.67e-  2
##  4 simplified_categoriesSingle-player    3.81e-2  0.0237       1.61  1.09e-  1
##  5 simplified_achievementsNone          -5.18e-2  0.0345      -1.50  1.33e-  1
##  6 price:simplified_categoriesMulti-play… -1.65e-3  0.00230    -0.716 4.74e-  1
##  7 price:simplified_categoriesSingle-pla…  4.19e-4  0.00108     0.389 6.97e-  1
##  8 price:simplified_achievementsNone     1.50e-3  0.00163     0.922 3.57e-  1
##  9 simplified_categoriesMulti-player:sim…  2.27e-2  0.0602      0.377 7.06e-  1
## 10 simplified_categoriesSingle-player:si…  4.55e-3  0.0438      0.104 9.17e-  1
## 11 price:simplified_categoriesMulti-play…  2.30e-3  0.00546     0.422 6.73e-  1
## 12 price:simplified_categoriesSingle-pla…  7.67e-4  0.00251     0.306 7.60e-  1
```

Fitting the data into the proposed model yielded this equation:

`prop_pos_ratings_hat` = 0.796 - (8.385e-04)(price) - (7.780e-02)(multi-player,yes) + (3.810e-2)(single-player,yes) - (5.184e-02)(achievements,none) - (1.649e-03)(price)(multi-player,yes) + (4.189e-04)(price)(single-player,yes) + (1.501e-03)(price)(achievements,none) + (2.272e-02)(multi-player,yes)(achievements,none) + (4.547e-03)(single-player,yes)(achievements,none) + (2.302e-03)(price)(multi-player,yes)(achievements,none) + (7.669e-04)(price)(single-player,yes)(achievements,none)

This equation states that when the price is zero and the game has no specified amount of achievements or category, the `prop_pos_ratings` is expected to be 0.796, or 79.6%. When the category of the game is multi-category and the game has less than or equal to 100 achievements, the `prop_pos_ratings` is expected to decrease by 8.385E-04 as the price of the game increases by $1.00. When the game takes on any other category and achievements combination, we can use the appropriate dummy variables and their corresponding coefficients in order to predict the rate of change of `prop_pos_ratings` as `price` changes along with the baseline (y-intercept) popularity and the predicted `prop_pos_ratings` for any specific combination.

For example, if we wanted to use our model to predict the popularity of a multi-player game with less than or equal to 100 achievements that costs $20.00, we could carry out the calculations as such:

`prop_pos_ratings_hat` = 0.796 - (8.385E-04)(20.00) - (7.780E-02)(1) + (3.810E-2)(0) - (5.184E-02)(0) - (1.649E-03)(20.00)(1) + (4.189E-04)(20.00)(0) + (1.501E-03)(20.00)(0) + (2.272E-02)(1)(0) + (4.547E-03)(0)(0) + (2.302E-03)(20.00)(1)(0) + (7.669E-04)(20.00)(0)(0)

which simplifies to…

`prop_pos_ratings_hat` = 0.796 - (8.385e-04)(20.00) - (7.780e-02)(1) - (1.649e-03)(20.00)(1) = 0.668

This indicates that my model predicts, on average, a $20.00 multi-player game with less than or equal to 100 achievements to have around a 66.8% positive ratings.

Now that I have a model to predict the `prop_pos_ratings` based on the `simplified_category`, `simplified_achievements`, and `price` of a game, I use the r-squared and adjusted-r-squared values to evaluate the quality of my model in explaining any variation of the popularity based on these explanatory variables.

```
## # A tibble: 1 × 2
##   r.squared adj.r.squared
##       <dbl>         <dbl>
## 1    0.0864        0.0666
```

By calculating both r-squared values, I noticed that both, especially the adjusted-r-squared which should fit my interactions model better, were very low, both being less than 0.100. This indicates that my model does not very accurately predict the trends in the data based on these explanatory variables to a considerable amount. Especially since these r-squared values are significantly less than 0.500, I cannot say that my model supports any correlation between `prop_pos_ratings` and `price`, `simplified_categories`, and `simplified_achievements`.

Even with such a low adjusted-r-squared value for my interactions model, I decided to calculate the RMSE to understand how well the model predicted the `prop_pos_ratings` for any data points that did fall into my model's prediction fairly well. This RMSE calculation is shown here:

```
## # A tibble: 1 × 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rmse    standard       0.130
```

The RMSE for my model gives a value of 0.130 which means that when predicting the popularity for games that fall more or less within an accurate prediction from my interactions model, the model is only off by 0.130, on average, in the prediction of its `prop_pos_ratings` . To see if this is a significant margin of error, I compared this value with the range of the testing data’s `prop_pos_ratings` .

```
##       range
## 1 0.8314379
```

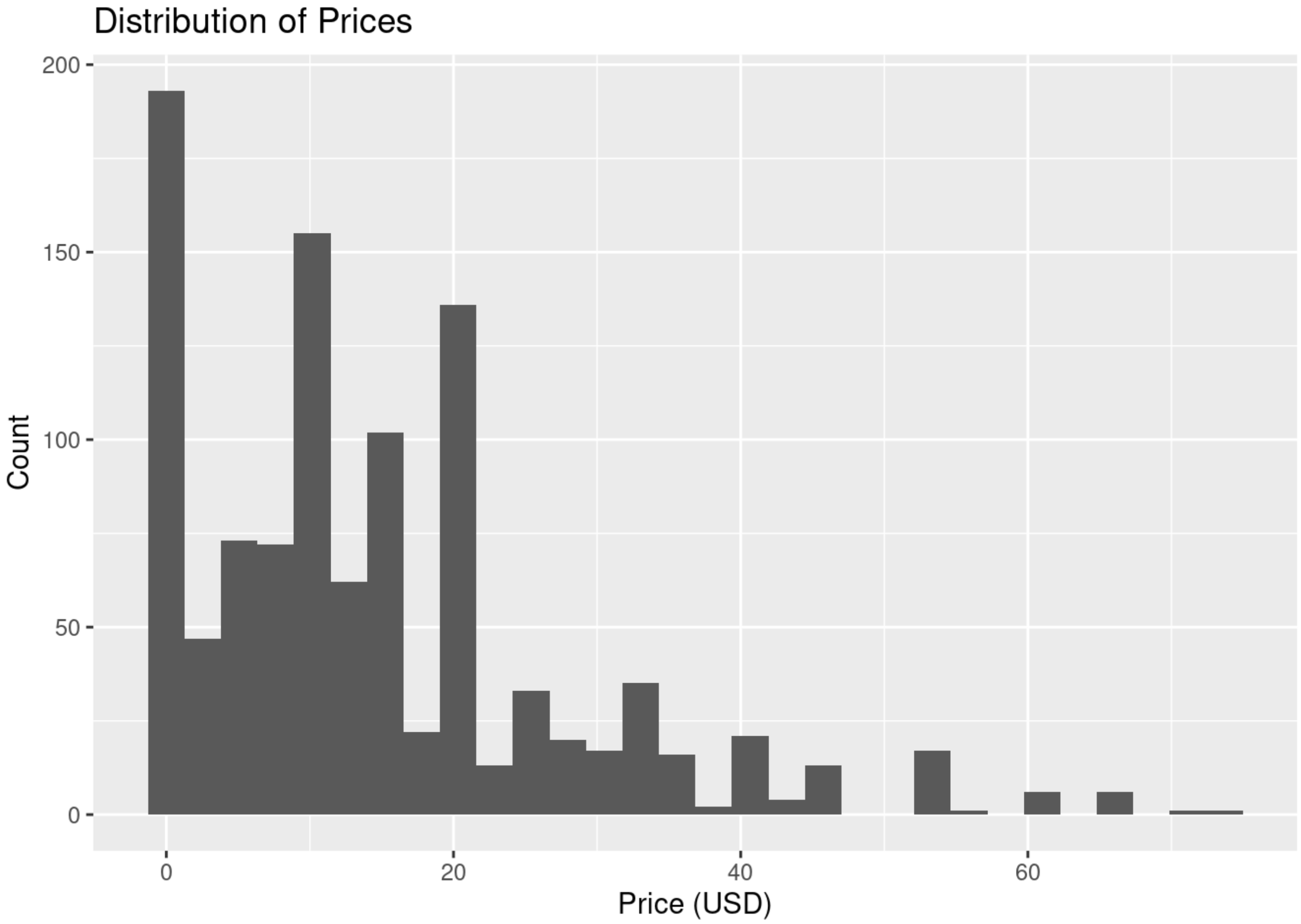
```
## [1] 0.156438
```

This calculation indicated that my RMSE value for my margin of error is only about 15.6% of the range of the `prop_pos_ratings` for the testing data which is fairly small. This means that for data points that are more or less accurately represented by my model, the margin of error by the RMSE is only about 0.130 off for the `prop_pos_ratings` calculation which is fairly small compared to the range of the `prop_pos_ratings` response variable.

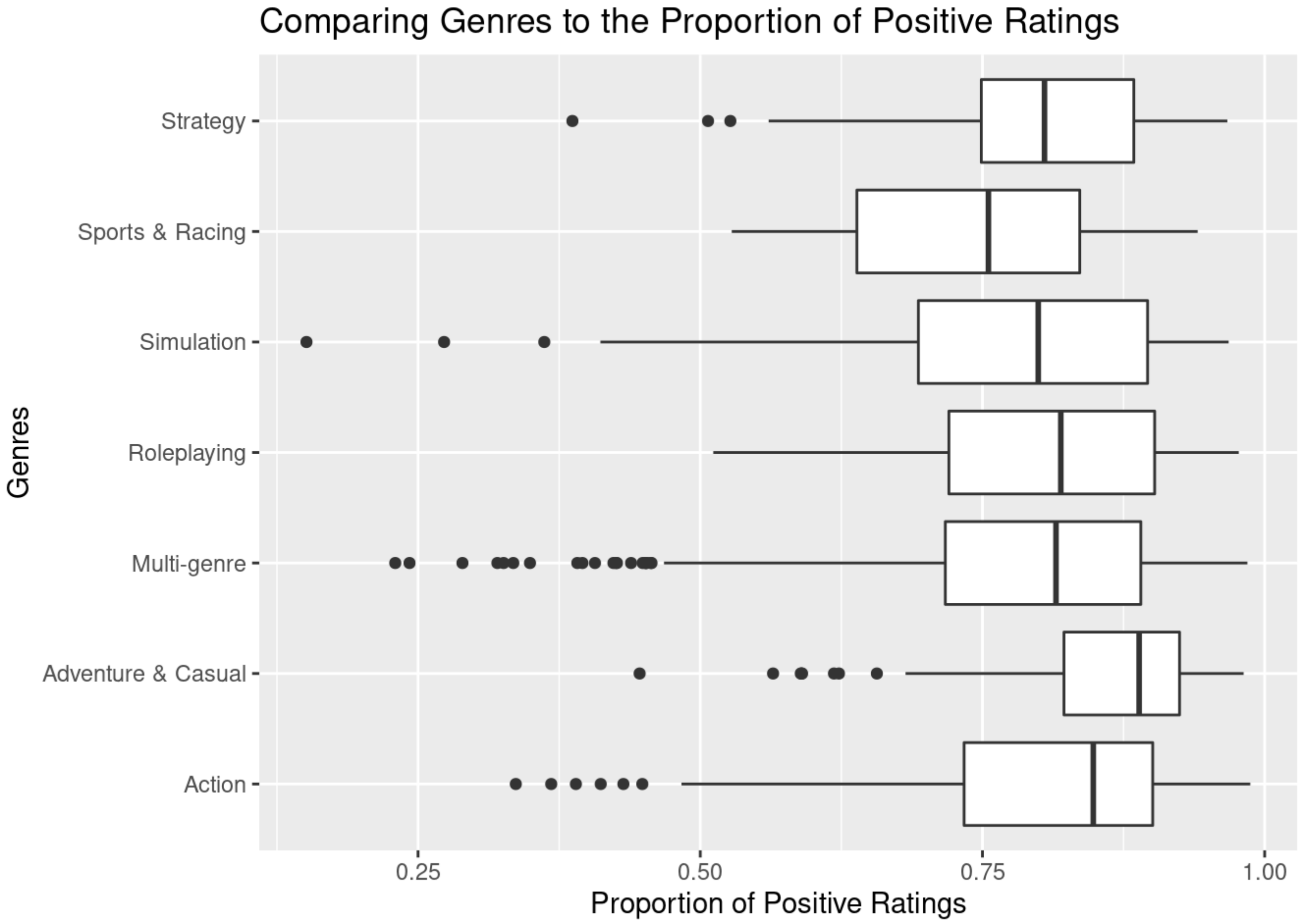
Model Proposed by Cameron

The model that I will be presenting will be the main fit model that uses the explanatory variables `price` , `simplified_categories` , and `simplified_genres` .

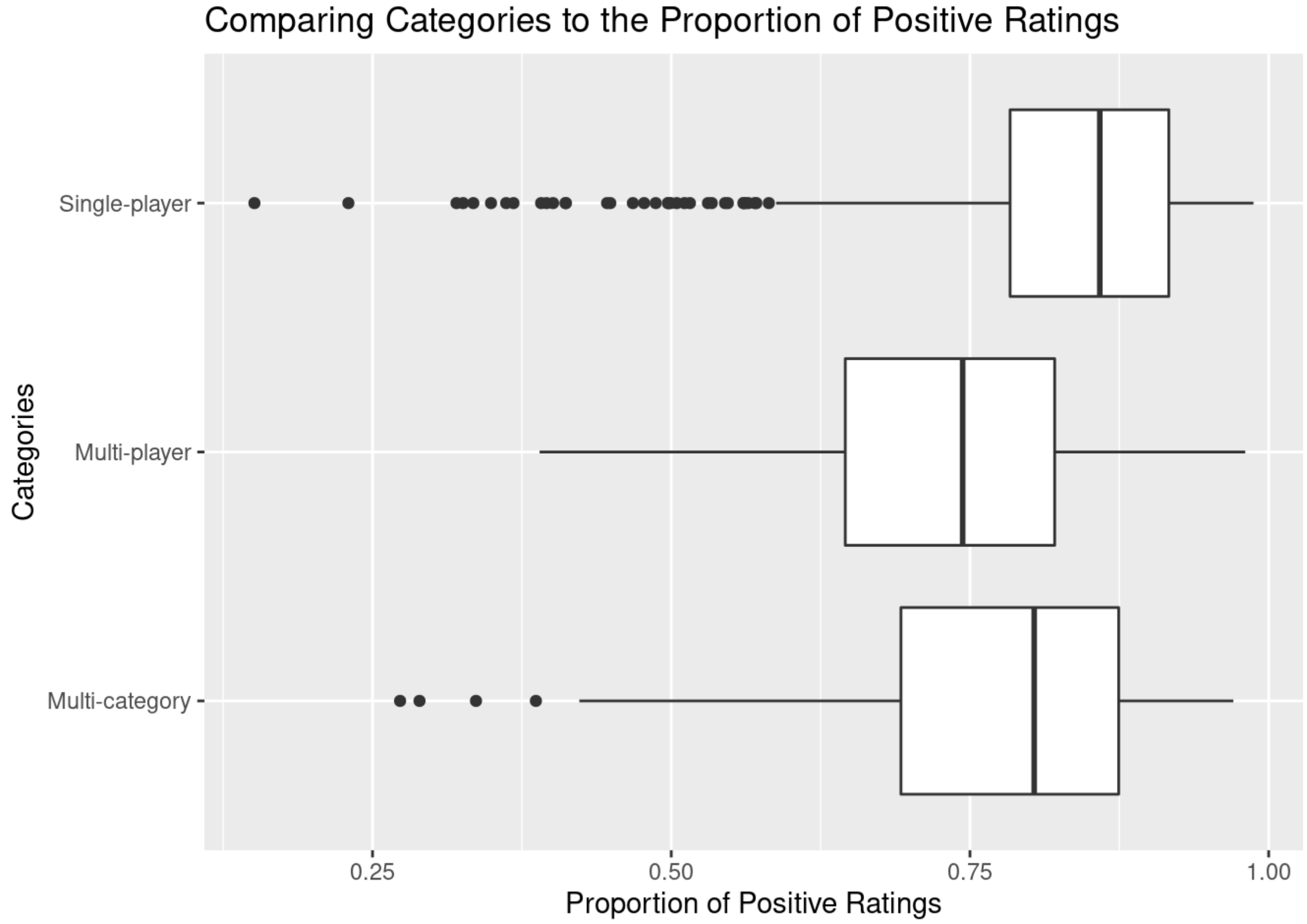
Price was chosen as an explanatory variable due to it being the only numerical variable that had a decent spread of prices. It was also interesting to observe how many games cost $0 and are free. I also kept the default binwidth of 30 because when using the rule of thumb, it produced a histogram with very skinny and rigid columns which made it difficult to interpret.



I chose genres as another explanatory variable because of how genre is an important component to a video game as it lays a foundation for the player and gives the player information on what’s to be expected from a game. When observing these boxplots, we can see that while there isn’t a drastic variation in the distribution of the proportions of positive rating between genres, we do see how some genres are concentrated more towards the higher end of the spectrum while some aren’t as concentrated and are placed more towards the lower end.



Similar to genres, categories are another variable that is important when developing a game as it represents how a player will proceed through the game. I also chose this variable due to the boxplot that I observed which shows a decent amount of variation with the median for each type of category ranging from 0.75 to just under 0.87.



To first create the model, I had to test the main fit model and the interactive fit model. I first made the main fit model which produced the following equation: prop_pos_ratings-hat = 0.801 - (price) * 0.000215 + (Adventure & Casual) * 0.00312 - (Multi-genre) * 0.0363 - (Roleplaying) * 0.00524 - (Simulation) * 0.0402 - (Sports & Racing) * 0.0366 - (Strategy) * 0.0155 - (Multi-player) * 0.0785 + (Single-player) * 0.0463. This equation shows that if a game was free and wasn’t categorized into any category or genre, it would have a proportion of positive ratings of 0.801. It also shows that with every dollar that the game costs, the ratings would fall by 0.000215 on average. For the rest of the variables, if the game is categorized as one of those categories or genres, a 1 would take the place of the variable type name to include the coefficient in the final calculation.

```
## # A tibble: 10 × 5
##    term                                 estimate std.error statistic   p.value
##    <chr>                                   <dbl>     <dbl>     <dbl>     <dbl>
##  1 (Intercept)                          0.801     0.0171      46.8   6.88e-185
##  2 price                               -0.000215  0.000468    -0.459 6.46e-  1
##  3 simplified_genresAdventure & Casual  0.00312   0.0230       0.136 8.92e-  1
##  4 simplified_genresMulti-genre        -0.0363    0.0152      -2.39  1.71e-  2
##  5 simplified_genresRoleplaying        -0.00524   0.0332      -0.158 8.75e-  1
##  6 simplified_genresSimulation         -0.0402    0.0364      -1.10  2.71e-  1
##  7 simplified_genresSports & Racing    -0.0366    0.0604      -0.605 5.45e-  1
##  8 simplified_genresStrategy           -0.0155    0.0272      -0.571 5.68e-  1
##  9 simplified_categoriesMulti-player   -0.0785    0.0235      -3.34  8.88e-  4
## 10 simplified_categoriesSingle-player   0.0463    0.0134       3.44  6.28e-  4
```

After creating the main fit model, I calculated the adjusted r-squared value to see how much of the data this model could account for. However, it yielded a small adjusted r-squared value of 0.0739 which means that this model would account for 7.39% of the total data.

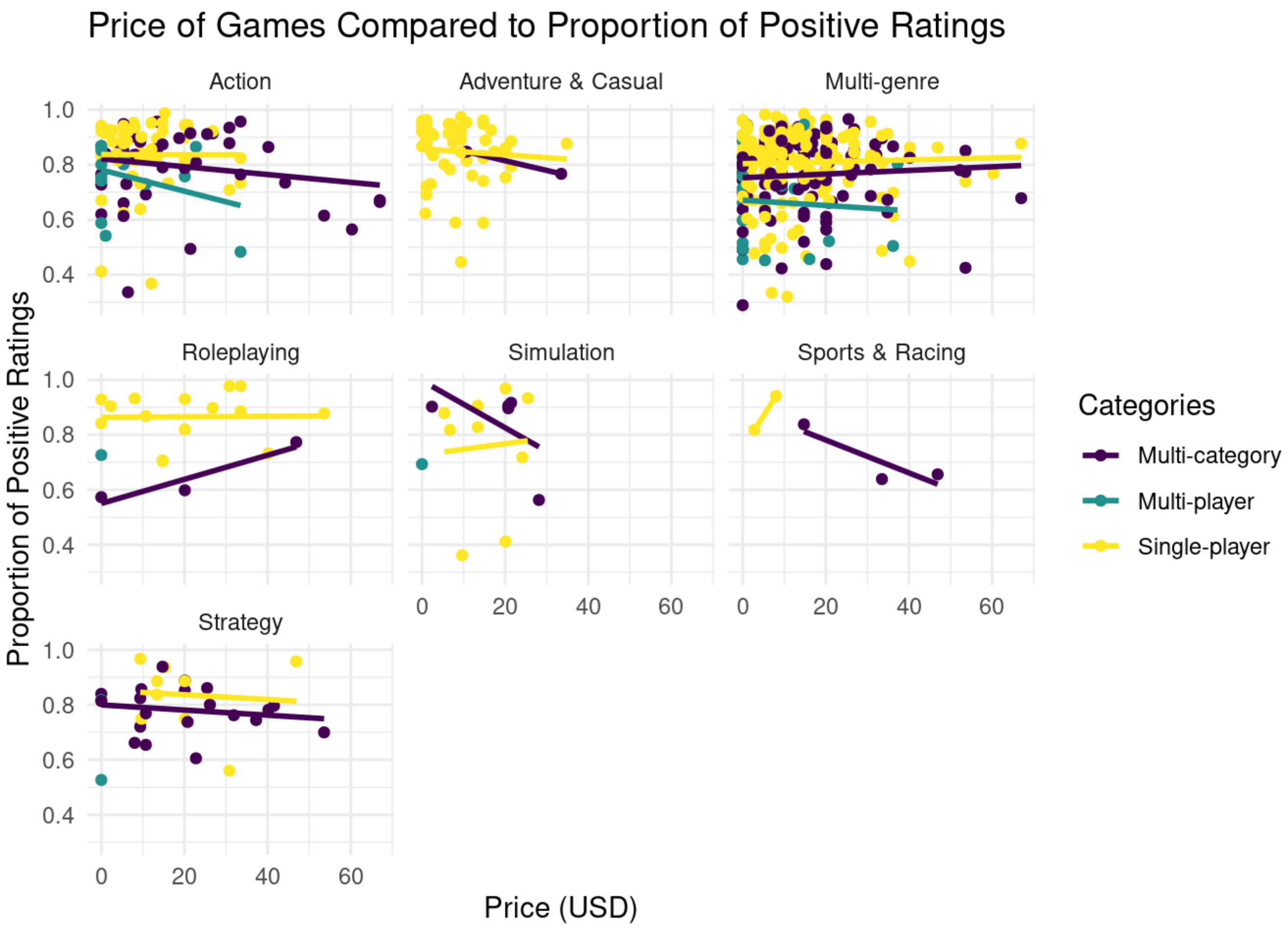
```
## # A tibble: 1 × 2
##   r.squared adj.r.squared
##       <dbl>         <dbl>
## 1    0.0902        0.0739
```

After doing these two things, I proceeded to create the interactive fit model to see if it would be a better model to utilize than the main fit.

After creating the interactive fit model, I immediately calculated the adjusted r-squared value for this model in order to see if the interactive fit model could explain more of the data. Instead, it yielded an adjusted r-squared value of 0.0613 or, in other words, it accounts for 6.13% of the data. Not only is this adjusted r-squared smaller than the main fit model but it is also more complex. Based on these two things, my decision was then to use the main fit model.

```
## # A tibble: 1 × 2
##   r.squared adj.r.squared
##       <dbl>         <dbl>
## 1     0.124        0.0613
```

In order to fully understand what the main fit model is telling us about these variables, I created a scatterplot that was facet wrapped using genres. This scatterplot really showed why the adjusted r-squared was low. Where there are a lot of data points, such as the single-player games in the multi-genre genre, the line of best fit appears to be almost horizontal showing no relation between the price of a game and the proportion of positive ratings. As for the lines of best fit that do appear to have a slope other than zero, there aren’t as many data points which further shows how our model doesn’t explain much of the data.



Once I had a main fit model, I decided to test it on the testing split to see how well it could predict that data set.

By creating an RMSE, I was able to truly analyze how well my model was able to predict the proportion of positive ratings of the testing split. The RMSE that I found was 0.128 which means that on average, the model was off by a proportion of 0.128. This shows that while my model doesn't account for much of the data, for the data that it does try to explain, it predicts them decently well.

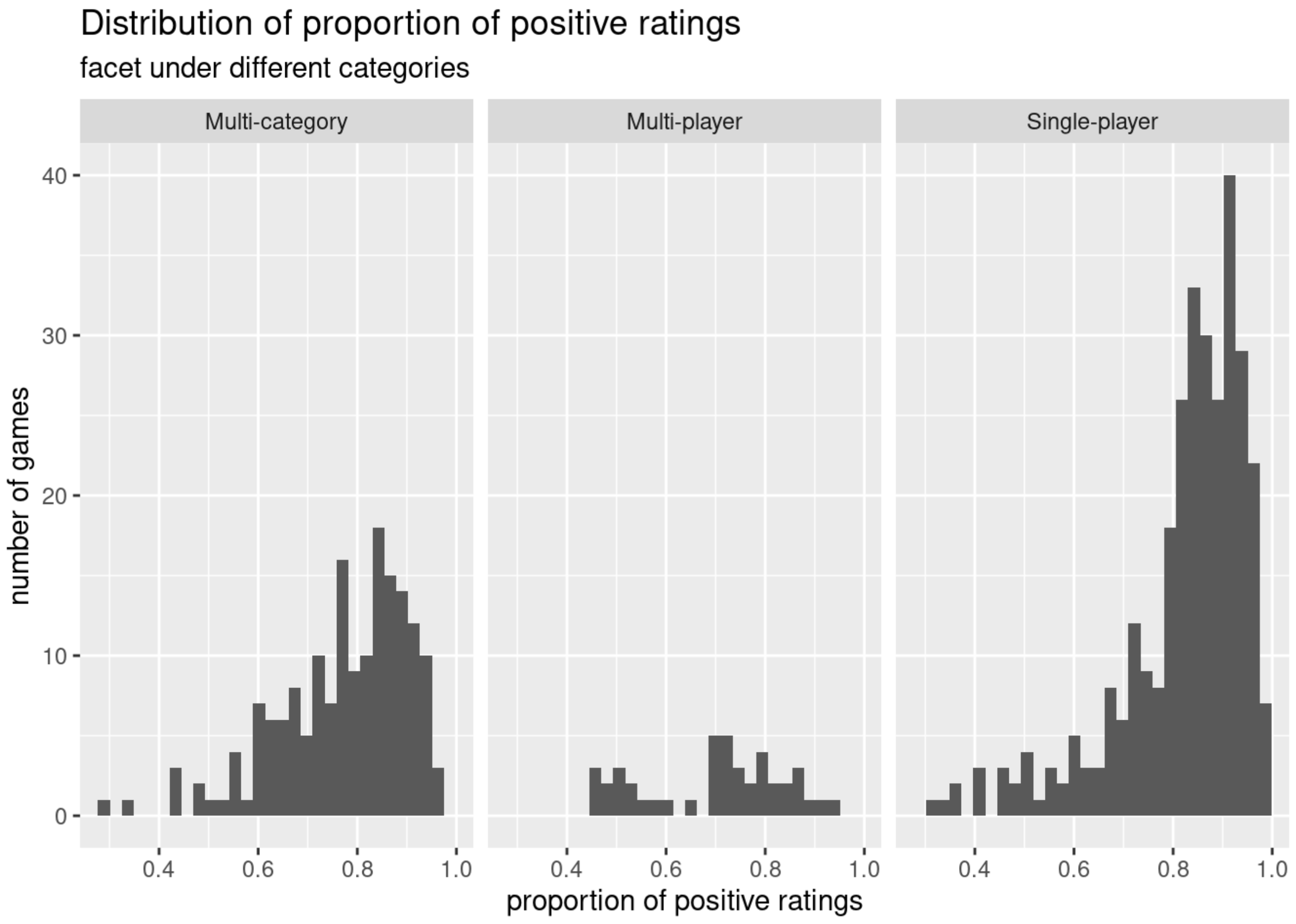
```
## # A tibble: 1 × 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rmse    standard       0.128
```

Model Proposed by Audrey

The three explanatory variables I use to set up my model are price, simplified categories, and simplified categories.

When I think of games, there are a couple of criteria that I would consider before I look into the game. I would like to see the game's categories and genres. Games in different categories (single-players, multi-player, co-op, etc.) would make the player's experience vary by a huge extent. So categories would be one of the explanatory variables I would consider in building my model. To make sure games in different categories would make a difference in their distribution, I made a facet histogram to look into the relationship between price and proportion of positive ratings under different categories.

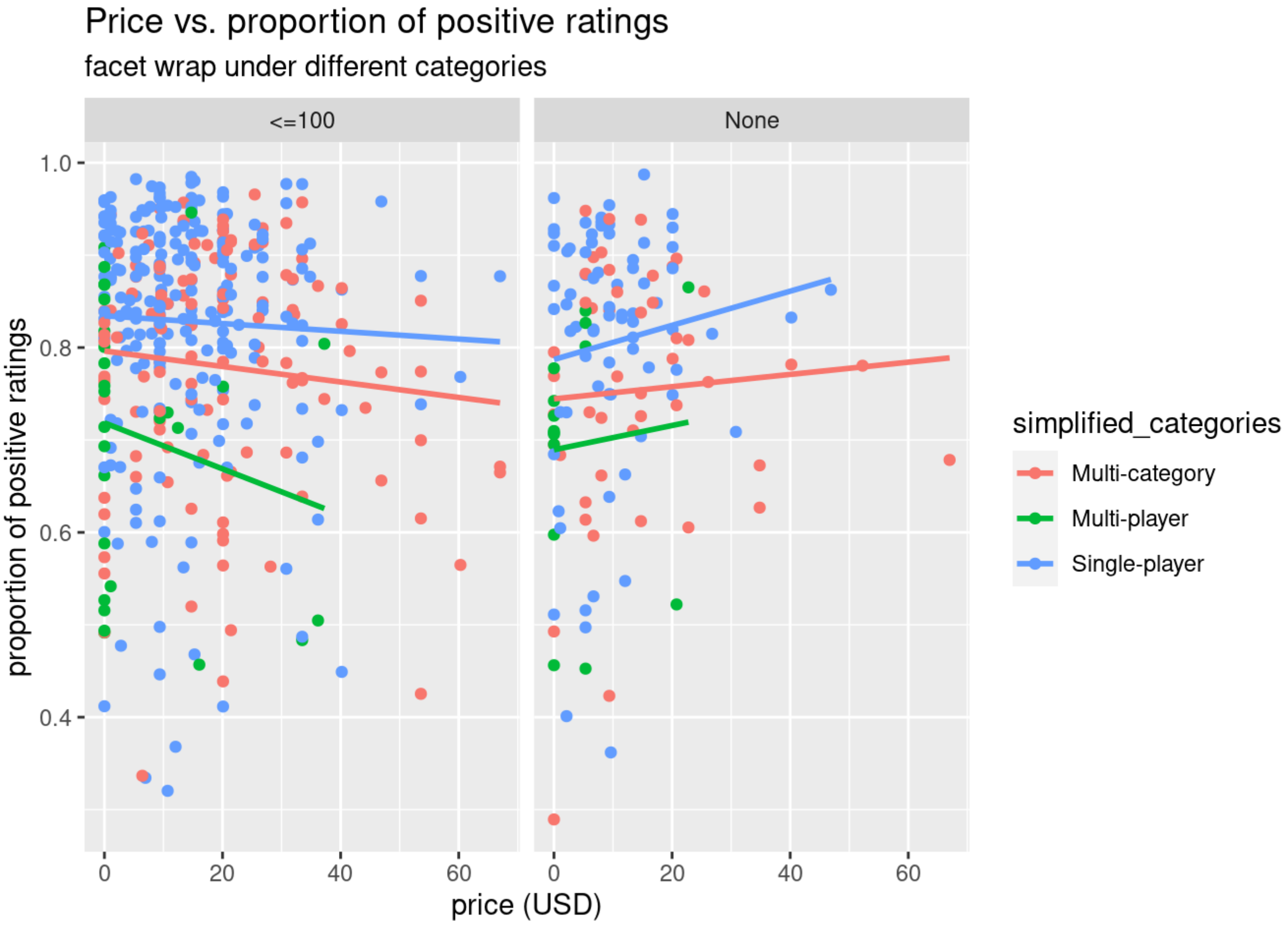
Looking at the graph, we can see that games that are single-player and multi-categories are both left-skewed where single players have a higher count at the very right, which means it is more left-skewed than multi-player games. Games from multi-player seem to have a shape of the distribution of bimodal and the count is much lower than the other two categories. From there, I can tell that different categories of games might affect the proportion of positive ratings.



The second category variable I considered was the number of achievements set in the game. For most single-player and co-op games, there are plenty of achievements preset in the games and some players do get obsessed with getting these achievements. So I did a facet scatter plot under different types of the number of achievements.

After visualizing the relationship using the scatter plot, I see that the correlation direction of games that have less and equal to 100 achievements is negative while the games that have no achievement have the direction of correlation as positive. Also, under different categories, the slope of correlation between the price and proportion of positive ratings changed.

From there, I can be sure that simplified_achievements can be put into my model as one of the explanatory variables.



I did the main fit model because the two category explanatory variables do not have much to interact with each other. However, both of these variables may bring some difference in the correlation between price and proportion of positive ratings.

Based on the result, my model can be explained by the following equation: prop_pos_ratings-hat = -0.000367 * price -0.0237 * (simplified_achievementsNone) -0.0763 * (simplified_categoriesMulti-player) +0.0476 * (simplified_categoriesSingle-player) + 0.786.

This equation means that generally for games that have no achievements, the proportion of positive ratings tends to go down. And for single-player games, the proportion of positive ratings tends to go up.

Although the model does make sense to me, the adjusted r-square is pretty low for my model. this could be caused by that most players would be more obsessed with the concept and the themes of the games instead of putting too much attention on the categories and the number of achievements of a game.

```
## # A tibble: 5 × 5
##   term                              estimate std.error statistic   p.value
##   <chr>                                <dbl>     <dbl>     <dbl>     <dbl>
## 1 (Intercept)                       0.786     0.0141      55.7   2.82e-220
## 2 price                            -0.000367  0.000468    -0.784 4.33e-  1
## 3 simplified_achievementsNone      -0.0237    0.0134      -1.77  7.69e-  2
## 4 simplified_categoriesMulti-player  -0.0763    0.0233      -3.28  1.12e-  3
## 5 simplified_categoriesSingle-player  0.0476    0.0129       3.69  2.46e-  4
```

```
## # A tibble: 1 × 2
##   r.squared adj.r.squared
##       <dbl>         <dbl>
## 1    0.0796        0.0725
```

Based on the RMSE value, 0.127, I can tell that on average, my model would predict by being off with a proportion of 0.127. And this is a fairly small value of RMSE, which means that although the model shows a very small direct correlation between the explanatory and response variables, it does predict the data precisely.

```
## # A tibble: 1 × 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rmse    standard       0.127
```

Deciding on Our Final Model

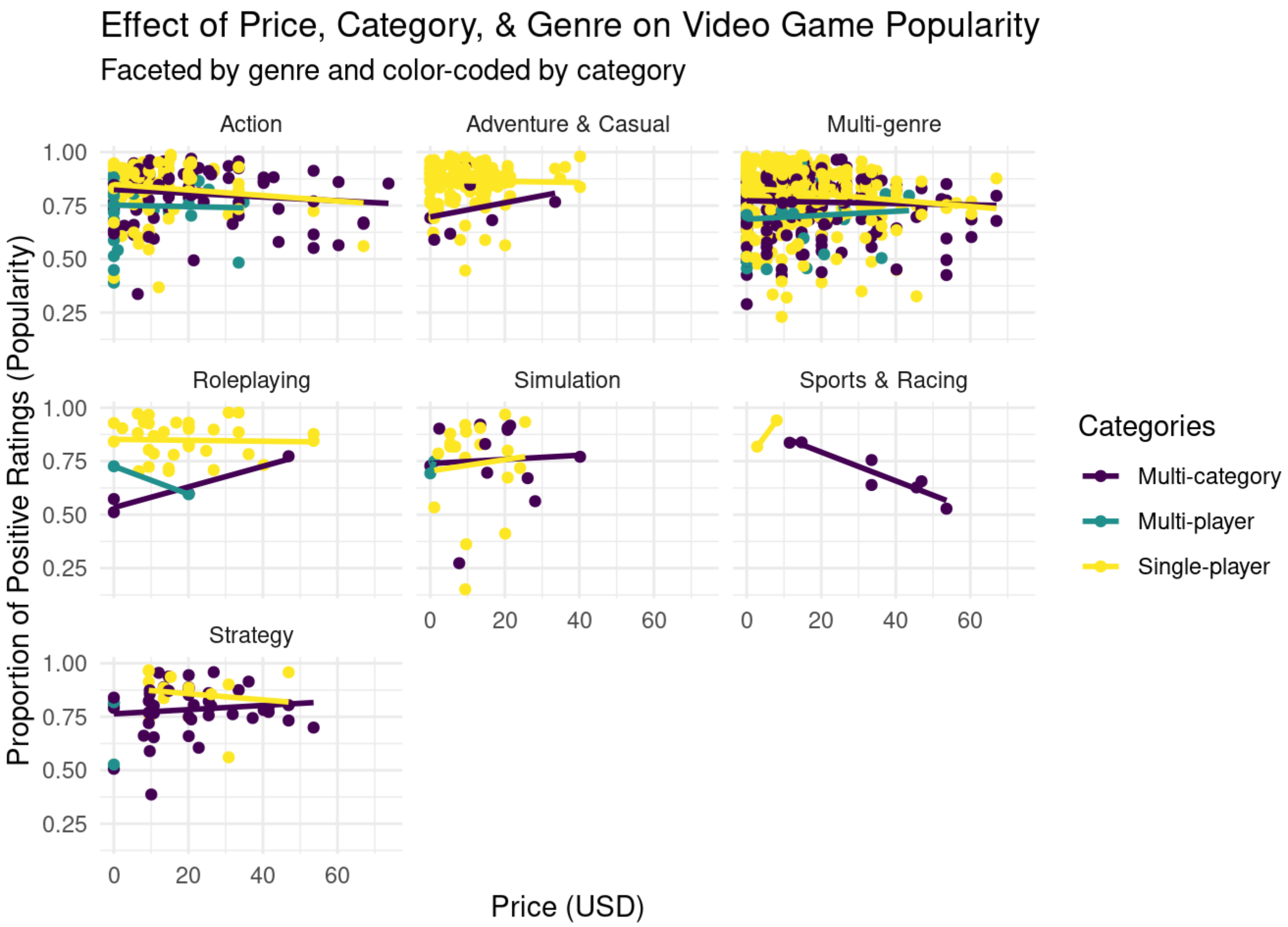
Person’s Model	Adjusted R-Squared	RMSE
Cameron	0.0739	0.128
Audrey	0.0724	0.127
Trinity	0.0666	0.130

Comparing our adjusted r-squared and RMSE values, we decided to focus on further analyzing and testing Cameron’s model since he had the highest adjusted r-squared value and lower RMSE value, very close to Audrey’s which was the lowest. Based on this reasoning, we will be using Cameron’s model to test our full truncated data set.

Section 3 - Results

Final Model

First, we started off by visualizing his model among the entire `steam_revised` data set. We plotted his chosen explanatory variables of `price` , `simplified_genres` , and `simplified_categories` against the `prop_pos_ratings` response variable in faceted scatter plots.



After visualizing these relationships, we used his main effects model in order to reformat his model to predict the `prop_pos_ratings` for our full revised data set.

```
## # A tibble: 10 × 5
##    term                                 estimate std.error statistic    p.value
##    <chr>                                   <dbl>     <dbl>     <dbl>      <dbl>
##  1 (Intercept)                          0.810     0.0120      67.3   0
##  2 price                               -0.000639  0.000315    -2.03  0.0429
##  3 simplified_genresAdventure & Casual  0.0118    0.0154       0.772 0.440
##  4 simplified_genresMulti-genre        -0.0337    0.0107      -3.14  0.00174
##  5 simplified_genresRoleplaying        -0.0183    0.0227      -0.807 0.420
##  6 simplified_genresSimulation         -0.0808    0.0243      -3.33  0.000898
##  7 simplified_genresSports & Racing    -0.0649    0.0442      -1.47  0.143
##  8 simplified_genresStrategy           -0.00882   0.0194      -0.455 0.649
##  9 simplified_categoriesMulti-player   -0.0677    0.0168      -4.02  0.0000636
## 10 simplified_categoriesSingle-player   0.0452    0.00939      4.81  0.00000172
```

The main effects model equation that we found was the following:

Prop_pos_ratings-hat = 0.810 - (price) * 0.000639 + (Adventure & Casual) * 0.0118 - (Multi-genre) * 0.0337 - (Roleplaying) * 0.0183 - (Simulation) * 0.0808 - (Sports & Racing) * 0.0649 - (Strategy) * 0.00882 - (Multi-player) * 0.0677 + (Single-player) * 0.0452

This equation shows that if a game didn't cost any money and wasn't included in any genre or category, it would have a proportion of positive ratings of 0.810. It also shows that, on average, for every dollar that the price increases by, the ratings would fall by 0.000639 on average. The rest of the variables are dummy variables which means that the coefficient to each variable is only included if the game that's being analyzed falls into that category or genre. Otherwise, the coefficient is not taken into account and the model is calculated on the assumption of the genre being action and the category as multi-category.

For example, if the game we were trying to predict the `prop_pos_ratings` for was a single-player, adventure & casual game that costs $50.00, we would use the proposed equation like so:

Prop_pos_ratings-hat = 0.810 - ((50.00) * 0.000639) + ((1) * 0.0118) - ((0) * 0.0337) - ((0) * 0.0183) - ((0) * 0.0808) - ((0) * 0.0649) - ((0) * 0.00882) - ((0) * 0.0677) + ((1) * 0.0452)

Prop_pos_ratings_hat = 0.835

This indicates that our model predicts a single-player, adventure & casual game that costs $50.00 to have a proportion of positive ratings of 83.5% on average.

Now that we have our fitted linear model, we can use other analytical techniques in order to check the quality of our model in accurately predicting the `prop_pos_ratings` based on our given explanatory variables. For this, we will use r-squared, adjusted-r-squared, and RMSE calculations.

We started by calculating our r-squared and adjusted-r-squared values:

```
## # A tibble: 1 × 2
##   r.squared adj.r.squared
##       <dbl>         <dbl>
## 1    0.0928        0.0847
```

Based on the r-squared and adj-r-squared values we got, it appears that our values for both increased a bit from 0.0902 to 0.0928 for the r-squared value and from 0.0739 to 0.0847 for the adjusted-r-squared. However, these r-squared and adj-r-squared values are still very low, below 0.500, such that we still cannot use this model to support our hypothesis that genre, price, and category are clear indicators of the popularity of a game.

Even though our r-squared and adj-r-squared values did not show promise, we still wanted to compare how well our model did in predicting the data points that was more accurate in explaining. To do this, we calculated the RMSE for our model.

```
## # A tibble: 1 × 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rmse    standard       0.128
```

```
##   (max = max(prop_pos_ratings) - (min = min(prop_pos_ratings)))
## 1                                                     0.8361182
```

```
## [1] 0.15311
```

Looking at our RMSE, we can infer that, on average, our model would be off by 0.128 in terms of the `prop_pos_ratings` for data points that are accurately accounted for in our model. This is a relatively low RMSE value considering that the range of `prop_pos_ratings` is 0.836, and our RMSE value only makes up about 15.3% of the total range.

Discussion & Conclusion

After creating several models with the use of different variables, we haven’t found a model that has a relatively high adjusted r-squared value which means that each of our models were unable to explain the majority of our data points. Since our main question asked about what factors made a game popular, it is possible that there aren’t any unique variables within our data set that would have a substantial impact on how a game would be received by the players. However, it is also possible that there are factors that influence a game’s predicted popularity that we did not consider as they were not included in our data set. When looking at the scatterplots, the majority of the lines of best fit appear to be nearly horizontal which shows a slope of nearly 0. In other words, it's showing that there is no relationship between price and the proportion of positive ratings for each category. In addition to this, to analyze how genre relates to the proportion of positive ratings, we would look at how high along the y-axis the lines are placed for each scatterplot (or each genre). Again, there appears to be no difference between genres since the three lines of best fit for each are usually placed on or above the 0.75 mark. This analysis can also be applied to categories such that each line of best fit, no matter the category, is also on or above the 0.75 mark. While there are some lines that have an obvious non-zero slope, the issue with these are how there aren't as many data points that create these lines of best fit, making it difficult to conclude the relationship between these variables in the actual population. Overall, since there doesn't appear to be any variables that greatly impact the popularity of a game, what can be said is that game developers likely shouldn't be too focused on making a game with one particular genre, category, or price point. It appears that positive ratings and popularity extend across all of these variables which shows that any one game has the chance to be well-received by people.

Bibliography

Davis, N. (2019). Steam Store Games (Clean dataset) Combined data of 27,000 games scraped from Steam and SteamSpy APIs, (Version 3) [steam.csv], https://www.kaggle.com/nikdavis/steam-store-games (https://www.kaggle.com/nikdavis/steam-store-games).

…