Khoa Nguyen

204329993

CS170A - Analysis of independent video games on Steam platform

I. Introduction

Steam as an online store has become ubiquitous with PC gaming in recent years. With 8000 game titles and counting, its successful business model has allowed more video game companies and publishers to reach a wider audience, and independent game developers also benefit from it. In fact, the store has seen an impressive growth for indie games year by year. It goes without saying that this growth comes with additional pressure on indie developers. Marketing a video game is not an easy task, especially with increasingly steep competition, so it would help to know the profile of a successful Steam indie game. For this reason, I would like to study how these games perform and find out which factors may decide the level of support from players.

II. Collecting data

Half of my dataset was retrieved from http://steamspy.com/genre/Indie on February 27, 2016. The Steam Spy website is not affiliated with Steam. It just collects daily data directly from Steam and, according to the site, "extrapolates data from limited number of user profiles and thus isn't 100% correct." Despite this disclaimer, Steam Spy is one of the few public databases for thousands of games on Steam platform, so I decided to use it for my project. Visitors of the site can download csv files of this format:

#	Game	Release date	Price	Score rank(Userscore	Owners	Playtime (Median)
1451	Unturned	7-Jul-14	Free	84% (92%)	20,631,598 ±101,103	14:35 (02:58)
2417	Garry's Mod	29-Nov-06	$9.99	93% (95%)	11,103,515 ±75,498	88:05 (15:42)
1903	Robocraft	8-Jul-14	Free	52% (81%)	9,297,959 ±69,316	17:45 (02:20)
1621	Heroes & Generals	11-Jul-14	Free	30% (69%)	9,066,155 ±68,475	11:09 (00:56)
85	Terraria	16-May-11	$9.99	95% (96%/83%)	7,309,481 ±61,680	70:31 (21:05)

Unfortunately, this valuable data does not have all the information I want to analyze. So I tested the site's API to see if I can get data such as game genres for a more complete dataset. This proves to be difficult because Steam Spy does not have any function like that. I contacted the site owner, Sergey Galyonkin, for advice and was told to look into Steam's API. The bad news is that Steam also lacks this feature. I spent some more time on websites that claim to extend Steam's API. They either use old data or contain only a few hundred games. My last option was to extract the data directly from Steam, but all the web crawlers I have tried just crashed after a few minutes. In the end I decided to write some scripts myself to complete this task.

Since the numbers in the # column above are just Steam Spy's ranking, not the game's appid on Steam store, they cannot be used to get game info. First I use the API http://steamspy.com/api.php?request=genre&genre=Indie to get indie games listed in the csv file I downloaded from Steam Spy. Here is part of the result:

{"304930":{"appid":304930,"name":"Unturned","developer":"Smartly Dressed Games","publisher":"Smartly Dressed Games","score_rank":84,"owners":21033062,"owners_variance":101714,"players_forever":15871716,"players_forever_variance":89201,"players_2weeks":1403177,"players_2weeks_variance":27213,"average_forever":863,"average_2weeks":294,"median_forever":173,"median_2weeks":74,"ccu":29596},
"4000":{"appid":4000,"name":"Garry's Mod","developer":"Facepunch Studios","publisher":"Valve","score_rank":93,"owners":11061616,"owners_variance":75117,"players_forever":10201995,"players_forever_variance":72251,"players_2weeks":1363409,"pl

I extracted the games' appids and names from this JSON array into a text file. These names will be used for counting the number of characters and quick fact checking. Then I wrote a bash script to download all game webpages from Steam using the appids. Each webpage is first retrieved with HTTP GET and then passed through several sed and grep commands to give me the game's genres. After several hours of waiting, my ad hoc web crawler finished without my IP being blacklisted by Steam. Finally, I added these new variables to my dataset and formatted it for Matlab:

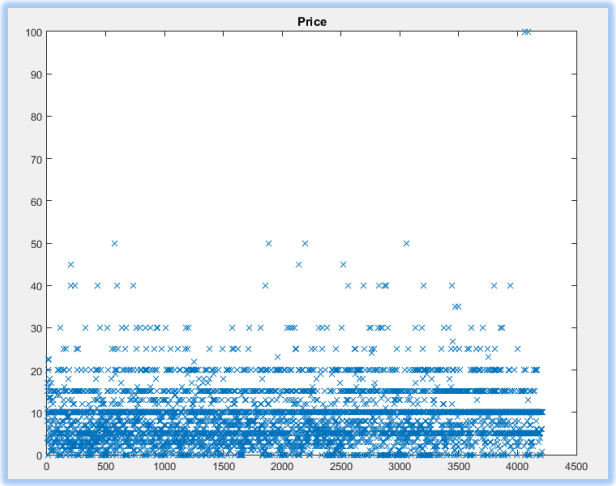
ID	Days on market	Price	Userscore	Metascor	Owners	Playtime	Median playtime	Gen1	Gen2	Gen3	Gen4	Gen5	Gen6	Gen7	Gen8	Gen9	Gen10	Genre Count	Sales/Day	Char. Count	Spec. Char. Count	Release Month	Release Year
378810	8	12.99	0	0	553	72	72	2	0	0	0	0	0	0	0	0	0	1	69	9	0	2	2016
430190	8	12.99	0	0	415	13	13	0	0	0	0	0	0	0	0	0	0	0	52	10	0	2	2016
439720	8	11.99	0	0	3734	177	206	0	9	11	0	0	0	0	0	0	0	2	467	14	0	2	2016
420110	8	9.99	0.98	0	43562	217	135	0	0	0	0	0	0	0	0	0	0	0	5445	13	1	2	2016

- ID: the game's appid as listed on Steam store.
- Days on market: the number of days since the game's release date on Steam to February 27, 2016.
- Price: MSRP, i.e. listed price of the game on Steam store, not current sales price. In US dollars.
- Userscore: average score Steam users give for the game.
- Metascore: average metacritic score for the game.
- Owners: total game copies sold, downloaded for free and given out. Same as Sales.
- Playtime: average lifetime playtime that Steam users spend on the game. Converted to minutes.
- Median playtime: median lifetime playtime that Steam users spend on the game. Converted to minutes.
- Gen1 to Gen10: other genres of indie game, formatted for Matlab. Some games have 10 genres.
- Genre count: total number of genres of the game listed on Steam minus 1 (all games are indie games).
- Sales/day: number of owners over days on market.
- Char. Count: number of alphanumeric characters in the game titles.
- Spec. Char. Count: number of special characters _+-.,!@#$%^&*()[];\/|<>'": in the game titles.
- Release month: the month the game appeared on Steam store
- Release year: the year the game appeared on Steam store.

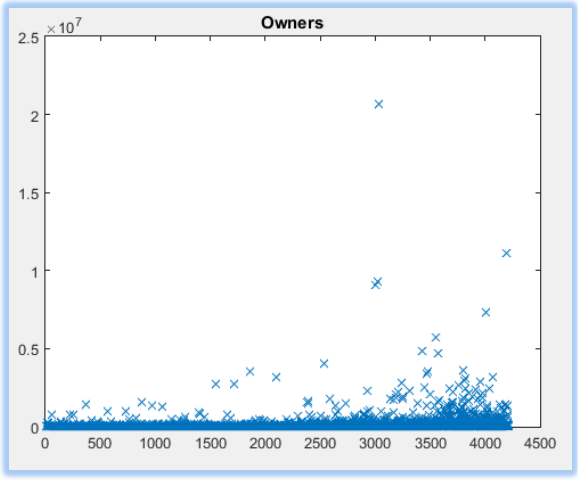
In total, there are 4211 entries in this dataset. Missing data (in Userscore, Metascore, Playtime and Median Playtime) is replaced with 0 so that Matlab can read it.

III. Removing outliers

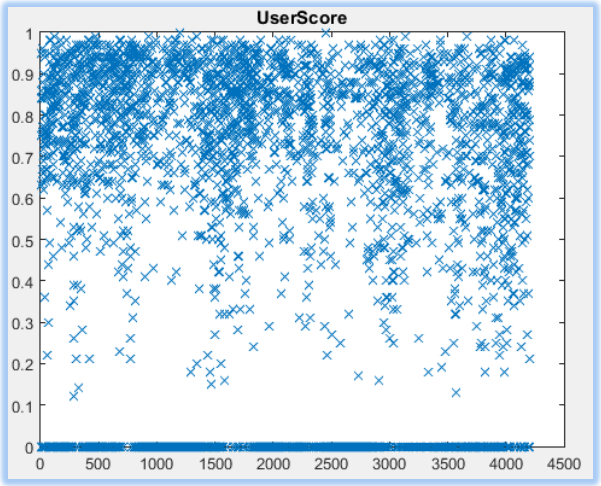
Scatter plot of each variable suggests that the dataset has outliers which may skew the analysis. I have tried to eliminate them in the following way:



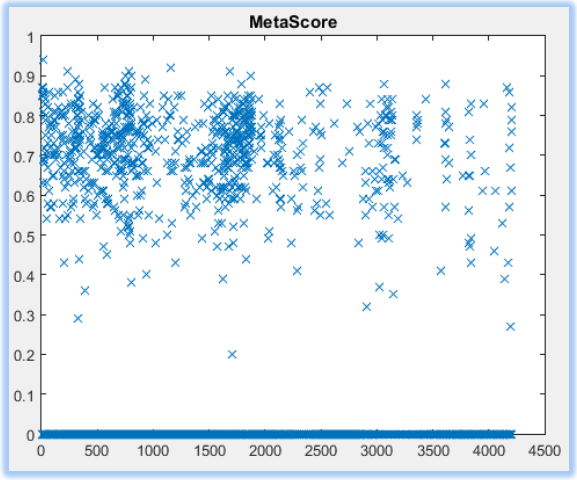
Ignore data points at $100 (2 titles)



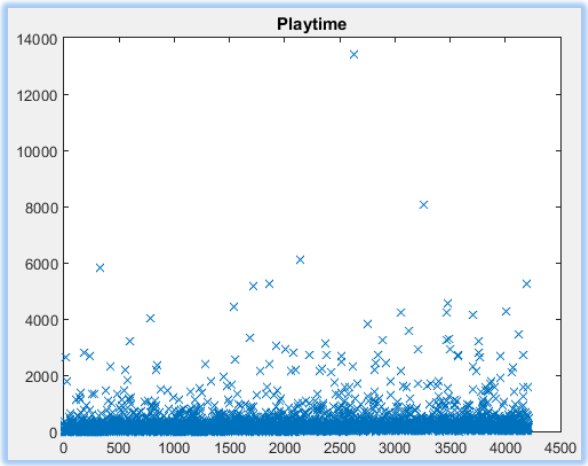
Ignore data points over 10^7 (2 titles)



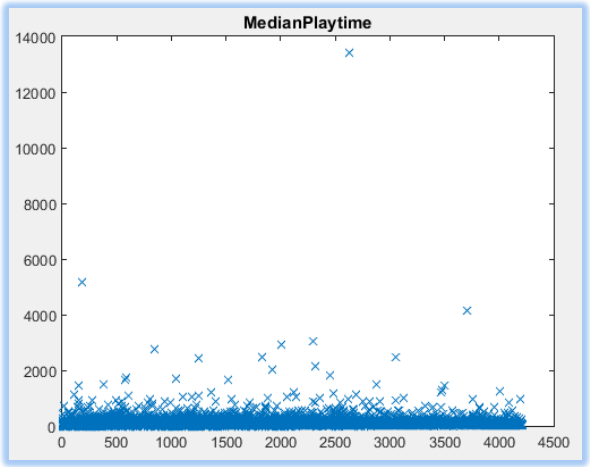
Keep all data points. Data points at 0% score are where data is not available



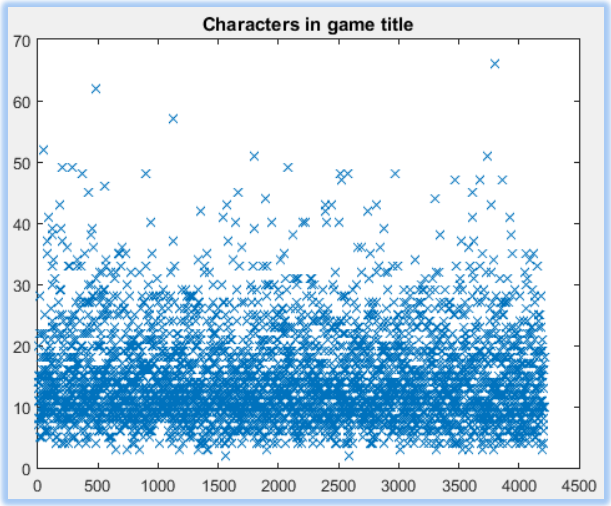
Keep all data points. Data data points at 0% score are where data is not available



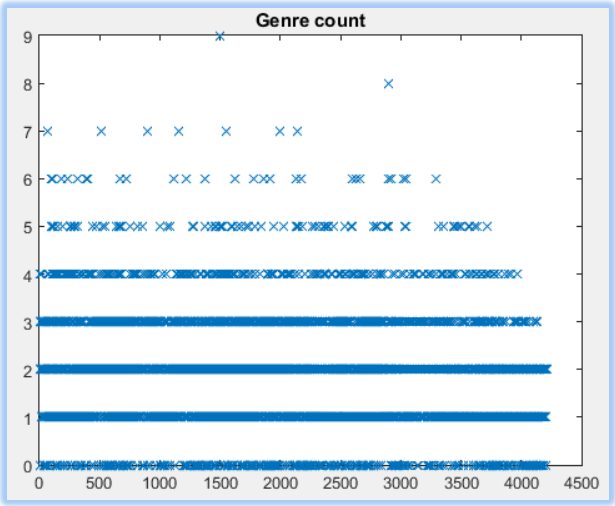
Ignore data points over 8000 minutes (2 titles)



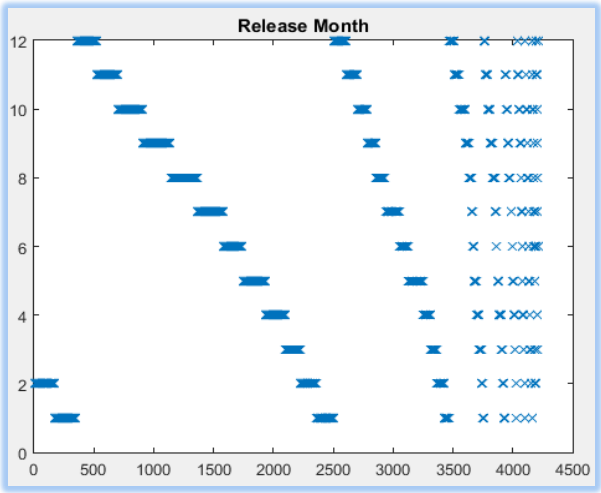
Ignore data points over 6000 minutes (1 title)



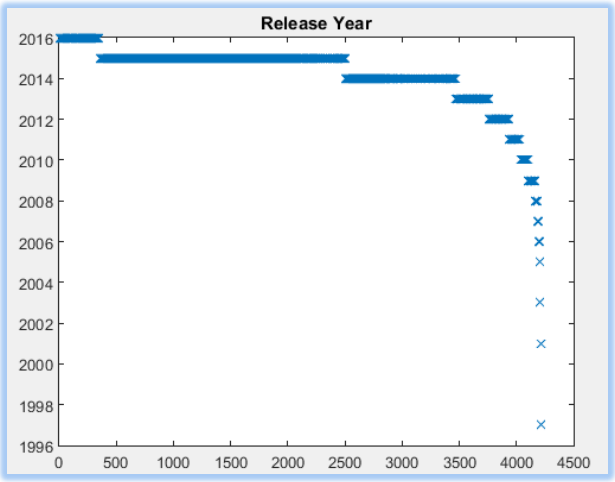
Keep all data points



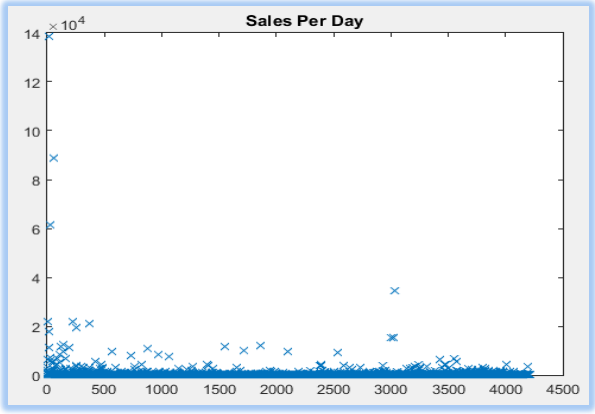
Keep all data points



Keep all data points



Ignore data points before 2008 because some months had no new releases (27 titles)



Ignore data points over 40000 sales per day (3 titles) and games released within 7 days of Feb. 27, 2016 (41 titles) because Steam Spy needs several days to refine data for new releases. (Total 42 titles)

In the end, the dataset has 4137 entries. After removing outliers, the percentage of missing data for Userscore, Metascore, Playtime and Median Playtime are 38%, 80%, 3%, 3% respectively.

IV. Data analysis

a. General statistics

Matlab code for this analysis is at the end of the report. Each statistic is calculated without the missing data, i.e. if Userscore is 0% then that entry is not counted in the average and standard deviation of Userscore. As highlighted in Figure 1, EarlyAccess games do not have a metascore because they are still in development. Also, two games labeled as FreetoPlay actually have prices. I looked up their info on Steam store, and this is not a mistake.

As of this writing, an average indie game on Steam would have been released on October 13, 2014 and has a userscore of 77% along with a metacritic score of 71%. Priced at $9.16, this game sells 253 copies per day with an accumulated sale of 112460 copies. It also belongs to 2 game genres, is played 4.5 hours on average, and its name is 14 character long.

Other interesting facts at a glance:

- Steam users enjoy FreetoPlay indie games, as evident by this genre having the most owners with a relatively small amount of titles.
- Action genre has the most games at 1967 titles, which is 23 times more than the genre with the fewest games, MassivelyMultiplayer, at 84.
- MassivelyMultiplayer games on average have the highest price yet lowest userscore. These games also belong to three other genres.
- Casual games by far have the cheapest average price and the least number of sales.
- Indie games, other than being Indie genre, tend to have 2 more genres.
- There are not a lot of racing games, and they were released quite long ago compared to other games.

Facts regarding indie game titles:

- Developers have a vocabulary of roughly 5000 words.
- 555 games use the colon symbol, most of which for subtitle, some for stylization.
- 589 games use article “the”, 93 use “a” and only 13 use “an”.
- Prepositions appear 561 times. “Of” is used 415 times. Second most used at 51 times is “to”.
- Longest title: “Hamlet or the last game without MMORPG features, shaders and product placement”. Second longest: “Dr. Langeskov, The Tiger, and The Terribly Cursed Emerald: A Whirlwind Heist”.
- 1 in 3 game titles will have 1 of these most common words or their variations:

Word	War	Death	Space	World	Star	Night	Dark	Game	Super	Hero	Dead
Count	109	74	72	68	66	59	56	55	53	51	50
Word	Adventure	Dungeon	Quest	Battle	Life	Time	Pixel	Simulator	Zombie	Legend	Sky
Count	47	45	40	39	34	32	32	30	30	30	28
Word	Black	One	Lost	Last	Planet	HD	Defense	Story	RPG	Park	Escape
Count	26	26	25	25	23	22	21	21	21	20	19

	Stats	General	Action	Adventure	Casual	E. Access	FreetoPlay	Multiplaye	Racing	RPG	Simulation	Sports	Strategy
	Total games	4137	1967	1457	1105	539	168	84	121	829	573	114	889
Price ($) non free games	Average	9.16	8.80	9.52	6.85	12.10	14.99	12.89	9.59	10.45	11.95	11.24	10.92
	Standard deviation	6.26	5.57	6.06	4.66	6.86	7.07	9.08	6.85	6.78	7.93	8.48	7.00
	Number of sample:	3910	1868	1401	1046	505	2	46	118	773	547	106	834
Userscore	Average	77%	76%	77%	76%	77%	72%	64%	78%	77%	73%	77%	75%
	Standard deviation	17%	18%	17%	17%	17%	18%	17%	16%	16%	17%	15%	17%
	Number of sample:	2572	1218	914	587	302	166	74	71	550	388	70	569
Metascore	Average	71%	71%	71%	71%	NaN	66%	70%	74%	72%	69%	74%	71%
	Standard deviation	11%	11%	11%	9%	NaN	16%	13%	9%	10%	10%	12%	11%
	Number of sample:	820	370	332	116	0	9	6	17	140	75	18	188
Days on market	Average	502	472	426	506	293	415	361	533	448	405	514	510
	Standard deviation	510	470	394	557	221	396	276	582	434	382	603	492
	Number of sample:	4137	1967	1457	1105	539	168	84	121	829	573	114	889
Owners	Average	112460	129050	104060	76149	104730	578580	486440	82063	134760	132880	103110	109920
	Standard deviation	402370	494030	382680	244170	628940	1240500	1516800	280080	545380	525680	304960	295800
	Number of sample:	4137	1967	1457	1105	539	168	84	121	829	573	114	889
Playtime (minute)	Average	274	229	244	228	306	275	598	213	402	400	387	425
	Standard deviation	440	375	382	354	525	585	961	370	563	638	814	584
	Number of sample:	4017	1907	1413	1059	511	168	84	119	815	564	109	866
Median playtime (minute)	Average	153	131	155	148	167	35	142	130	201	187	173	191
	Standard deviation	221	181	223	232	271	52	178	205	229	234	292	243
	Number of sample:	4003	1903	1409	1055	509	168	84	119	811	564	109	865
Genre count	Average	2.1	2.3	2.5	2.5	3.2	3.2	4.0	2.9	2.9	3.0	3.2	2.7
	Standard deviation	1.1	1.2	1.2	1.2	1.2	1.3	1.5	1.3	1.3	1.3	1.3	1.2
	Number of sample:	3807	1967	1457	1105	539	168	84	121	829	573	114	889
Sales per day	Average	253	267	233	232	292	1858	1197	259	297	447	374	276
	Standard deviation	1020	1089	813	1188	1317	3321	2732	1289	1193	1680	1406	1083
	Number of sample:	4134	1967	1457	1103	539	168	84	121	829	573	114	889
Character count	Average	14.3	13.6	15.0	14.5	13.1	14.9	14.3	12.5	15.3	14.9	15.6	14.4
	Standard deviation	7.4	6.6	8.3	7.3	6.4	8.2	6.8	5.2	7.8	6.8	6.8	6.6
	Number of sample:	4137	1967	1457	1105	539	168	84	121	829	573	114	889
Special character count	Average	1.2	1.3	1.2	1.2	1.3	1.3	1.1	1.2	1.2	1.2	1.4	1.2
	Standard deviation	0.6	0.7	0.6	0.5	0.9	0.8	0.2	0.5	0.7	0.6	1.0	0.7
	Number of sample:	921	390	368	247	101	38	20	18	206	136	26	214

Figure 1: General statistics of Steam indie games from February 2008 to February 2016

b. Analysis of total games by month throughout all years

	Stats	General	Action	Adventure	Casual	E. Access	FreetoPlay	Multiplayer	Racing	RPG	Simulation	Sports	Strategy
	Total games	4137	1967	1457	1105	539	168	84	121	829	573	114	889
	Month												
Total games	1	402	196	144	116	63	19	8	11	81	59	14	88
	2	356	164	129	103	39	18	13	10	74	48	10	74
	3	232	107	82	70	27	9	5	6	47	32	6	50
	4	290	141	107	80	25	10	6	7	55	40	4	66
	5	374	152	131	106	36	18	4	16	74	54	7	82
	6	266	140	78	77	36	7	5	11	50	45	3	49
	7	375	197	132	100	57	12	6	13	88	48	10	81
	8	363	164	135	90	37	26	14	5	57	51	15	92
	9	373	196	135	93	42	10	4	9	63	49	13	77
	10	405	186	141	88	60	15	6	10	79	52	9	81
	11	361	165	115	96	63	12	5	11	78	48	10	78
	12	340	159	128	86	54	12	8	12	83	47	13	71

Figure 2: Number of Steam indie games by month from February 2008 to February 2016

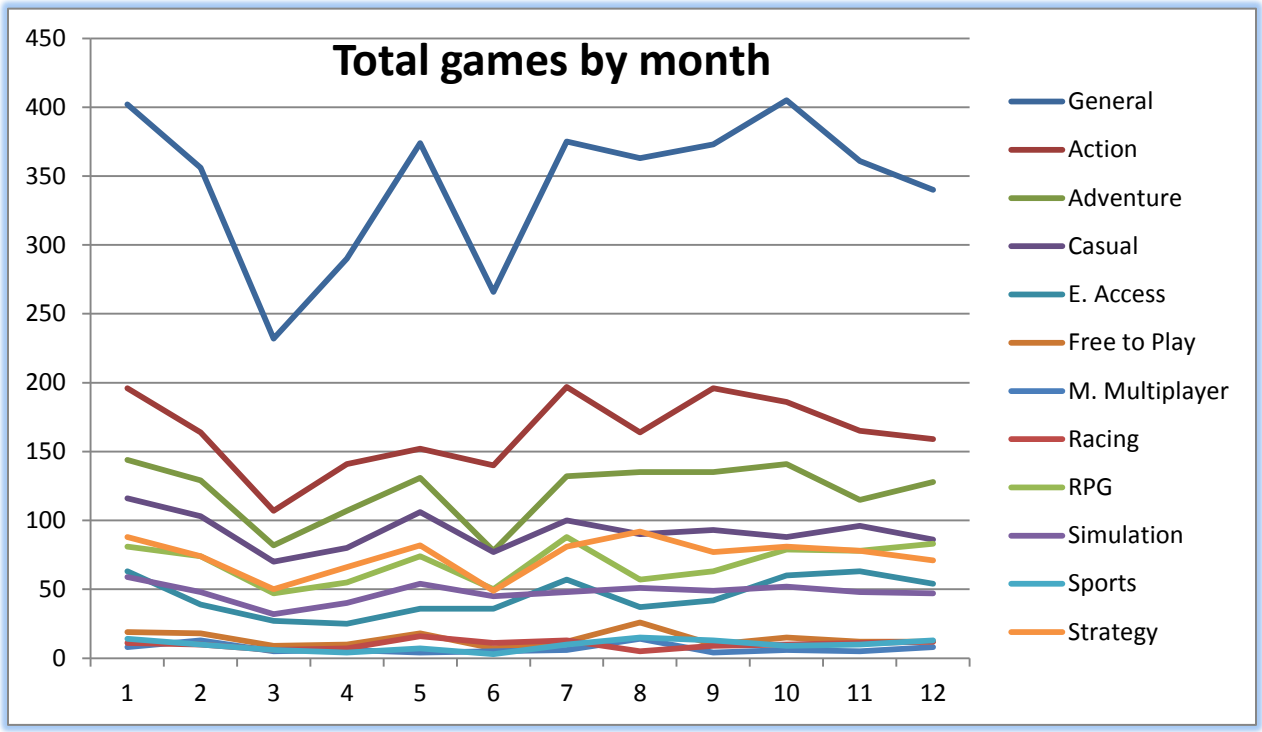


Figure 3: Number of new indie games by month from February 2008 to February 2016

Figures 2 and 3 suggest that the peak months for indie game releases are January and October. The period from July to November also has a steady output of new games while there is a decline in June and December. So my suspicion is that the amount of new releases throughout the year corresponds to Steam sale events. Specifically, perhaps indie game developers tend to release games before a big sale event on Steam store to avoid competition. I tried to verify this hypothesis by plotting new releases by year and looking for pattern. The following results are calculated with Matlab and then plotted in Excel for better visualization:

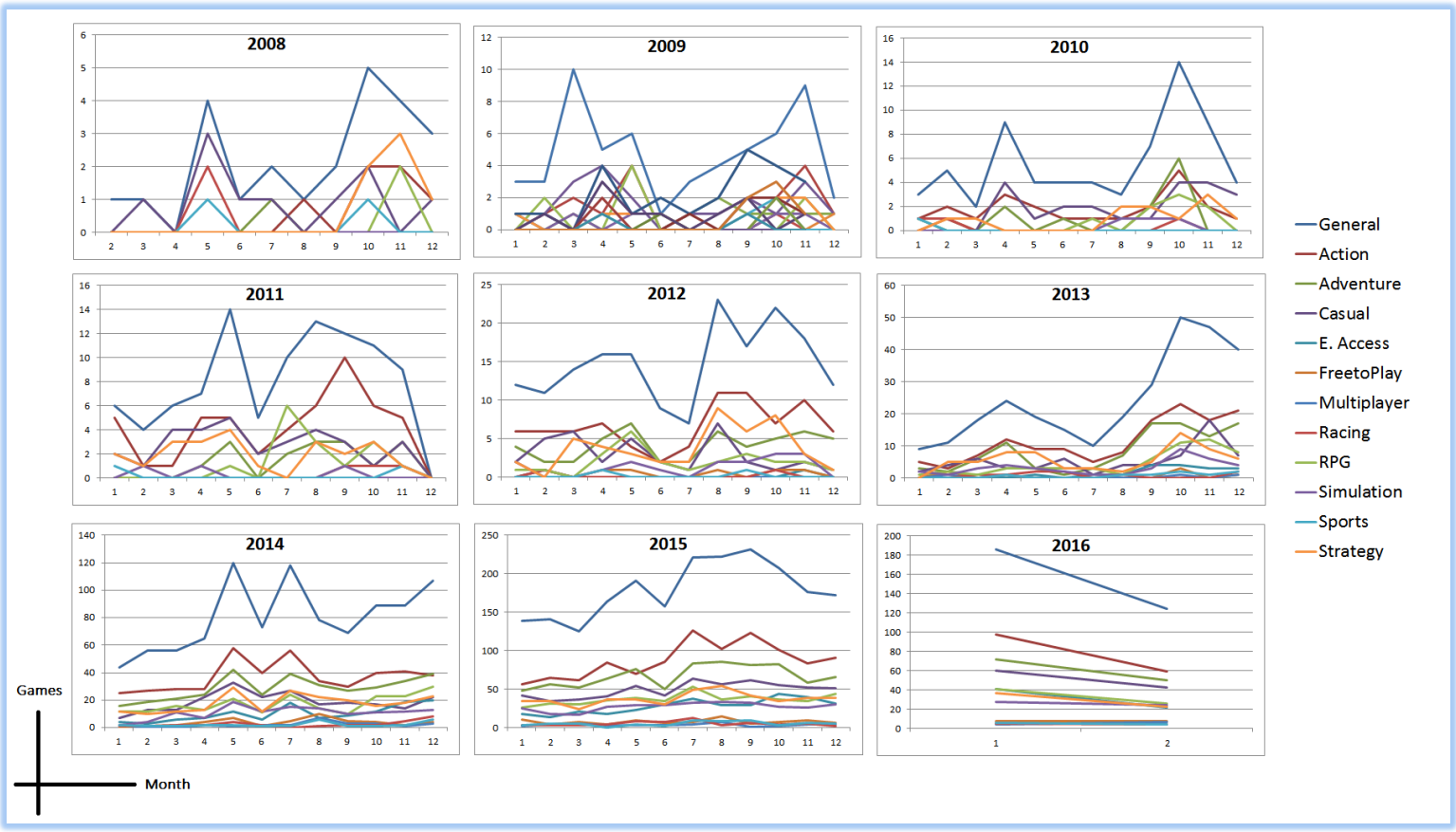


Figure 4: Number of indie game releases by month in each year

There seems to be a strong pattern in the charts above: the amount of new releases decreases sharply around June and December when Steam sale events take place. I have collected the following sale dates from a list of Steam announcements for reference:

			2009 - Dec 23 - Jan 3 - Holiday Sale
2010 - Jun 24 - Jul 4 - Summer Sale	2010 - Oct 28 - Nov 1 - Halloween Sale		2010 - Dec 20 - Jan 2 - Holiday Sale
2011 - Jun 30 - Jul 10 - Summer Sale	2011 - Oct 27 - Oct 31 - Halloween Sale	2011 - Nov 23 - Nov 27 - Autumn Sale	2011 - Dec 19 - Jan 1 - Holiday Sale
2012 - Jul 13 - Jul 23 - Summer Sale	2012 - Oct 29 - Nov 1 - Halloween Sale	2012 - Nov 21 - Nov 26 - Autumn Sale	2012 - Dec 20 - Jan 5 - Holiday Sale
2013 - Jul 11 - Jul 22 - Summer Sale	2013 - Oct 29 - Nov 1 - Halloween Sale	2013 - Nov 27 - Dec 3 - Autumn Sale	2013 - Dec 19 - Jan 2 - Holiday Sale
2014 - June 19 - June 30 - Summer Sale	2014 - Oct 30 - Nov 3 - Halloween Sale	2014 - Nov 26 - Dec 2 - Exploration Sale	2014 - Dec 18 - Jan 2 - Holiday Sale
2015 - Jun 11 - Jun 30 - Summer Sale		2015 - Nov 25 - Nov 30 - Exploration Sale	2015 - Dec 22 - Jan 2 - Winter Sale
2016 - Feb 5 - Feb 7 - Lunar Sale			

Sale events on Steam store http://store.steampowered.com/news/?feed=steam_announce&headlines=1

These dates match the low points on Figure 4. For example, 10 new games were released in July 2013, which is lower than in other months. This decline may be related to the summer sale that ran from

July 11 to July 22 the same year. Similarly, 50 new games were released in October 2013. Then the number gradually drops off as the holiday sale in December approached.

This result is interesting because of its implication. It suggests that indie developers in general avoid releasing their games in the month of a Steam sale event. One explanation may be that their new games are in direct competition with older, more well-known games which in addition have discounted prices. Moreover, Steam advertises these sales on their front page, so this visibility which other games enjoy may hurt the selling power of newer games.

On the other hand, developers seem to release games 1 or 2 months before in anticipation of sale events. Peaks in the charts before sale months indicate a possibility of this practice among indie developers. Usually a video game sells the most copies during its first few weeks, so this is the period where sales will drop sharply. My speculation is that developers intend to participate in Steam sale events to reignite customers' interest in their games. However, whether developers want to reduce prices that early to gain more sales cannot be determined with my dataset which only includes lifetime sales. I think a thorough business analysis is more suitable to answer this question. Free databases such as Steam Spy and Steamdb.info have price history for each game, but until I can find a method to retrieve this information from their graphs and charts, the speculation is left unverified.

The result above agrees with my hypothesis but can neither confirm nor deny it. There are many unknown factors such as bugs in codes, players' initial reception to demos or betas, and marketing strategy which will affect a game's release dates. Even though Figure 4 may show a trend, my dataset does not reflect those nuances. Moreover, the numbers of new games between 2008 and 2015 differ drastically, so there is no certainty that indie developers have this practice for releasing games. To put it into perspective, here is the growth of indie games on Steam month by month:

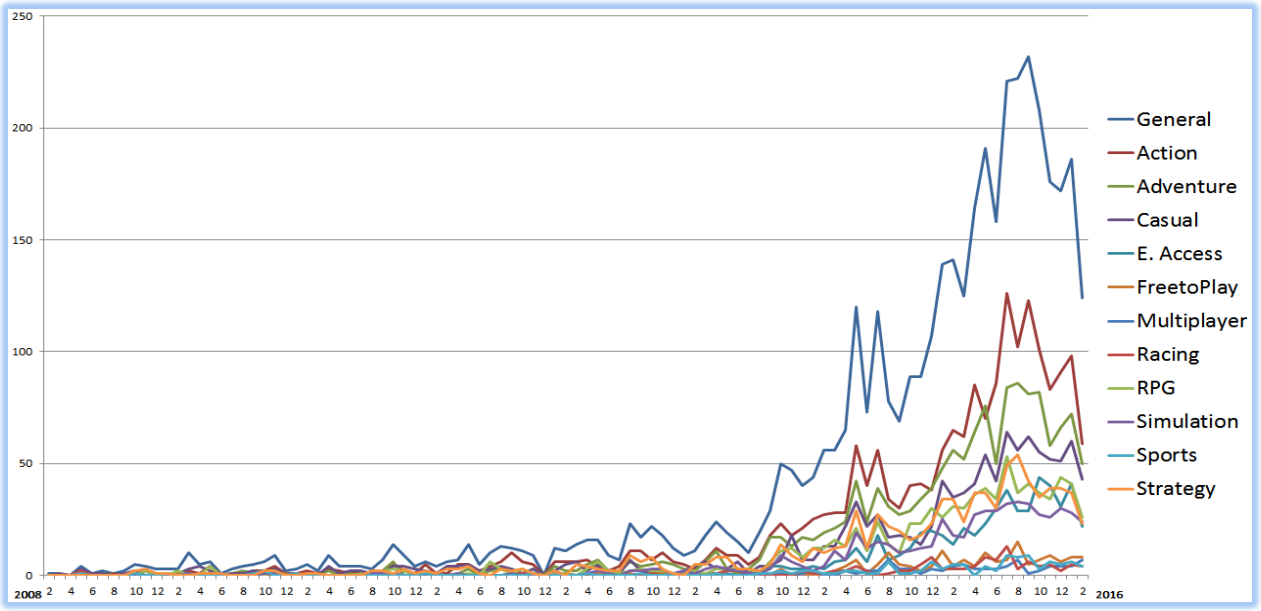


Figure 5: Number of new indie games month by month from February 2008 to February 2016

More data from the following years are needed before I can make a conclusion.

c. Analysis of total games by genre throughout all years

1. Distribution of genres

From the data above, it is quite clear that Action and Adventure are the most popular genres for indie games on Steam. In any average month, the majority of new indie games are labeled with these two genres. This may tell us about the preference of indie game developers and also the market, so I attempted to find the distribution of genres. The following figure shows how game genres appear in a month:

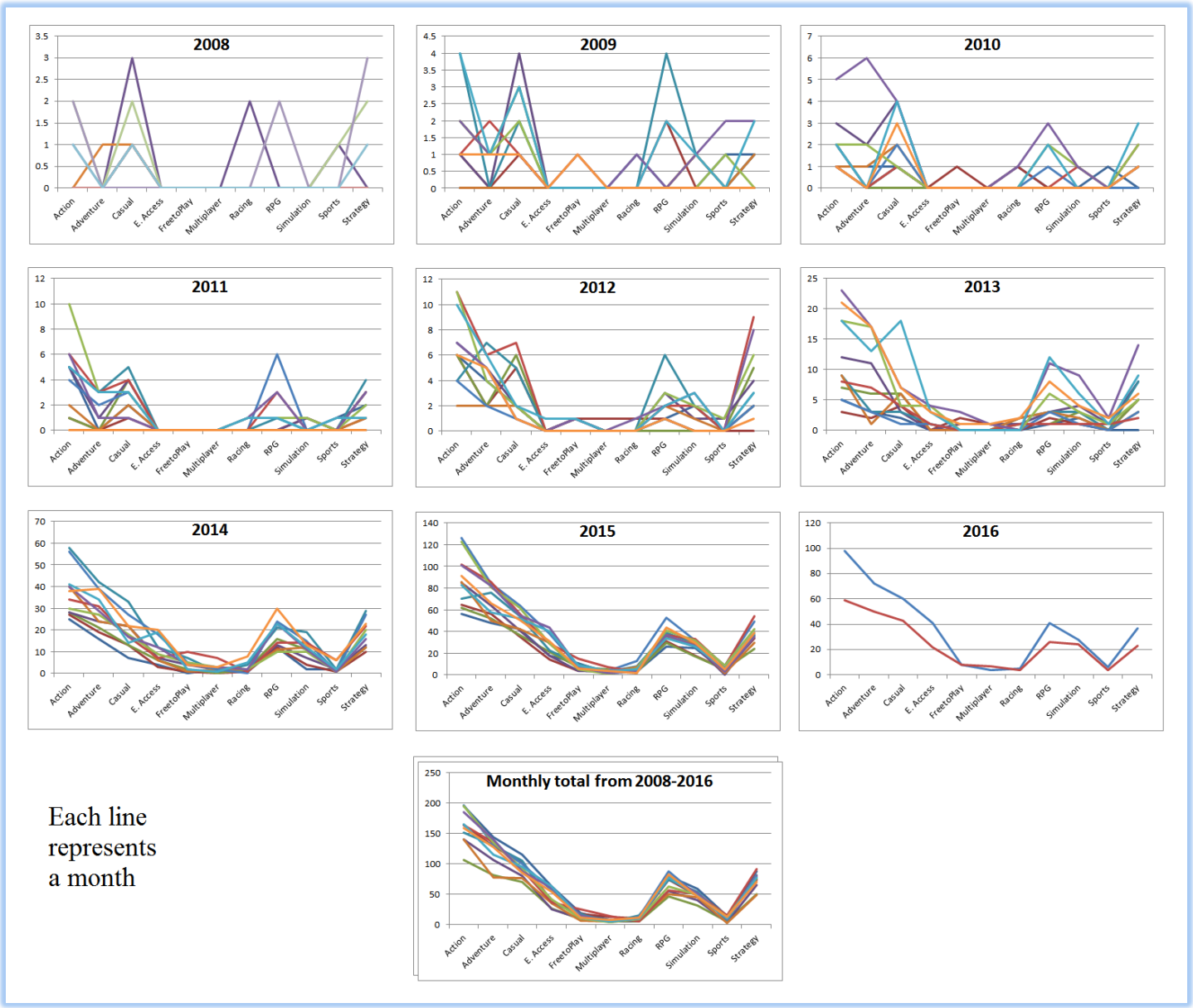


Figure 6: Number of new indie games by genres from February 2008 to February 2016.

As seen above, the distributions of game genres converge to a defined shape as the number of new indie games increases through the years. This trend is observed even within the first two months of 2016, so it cannot be a coincidence. Two different conclusions can be made here. First, Steam may have a

quota for each game genre they will allow in a month. Since developers have to plan their release dates with Steam, it is possible that the store enforces a genre distribution on new indie releases. Second, indie developers may have a clear-cut preference for what types of games they want to create. Conversely, the market for each game genre has stabilized even though Steam indie games are still growing rapidly. Either way, from this result we can create a model for the distribution of game genres in a month. Below is a 3D view of the same data in Matlab:

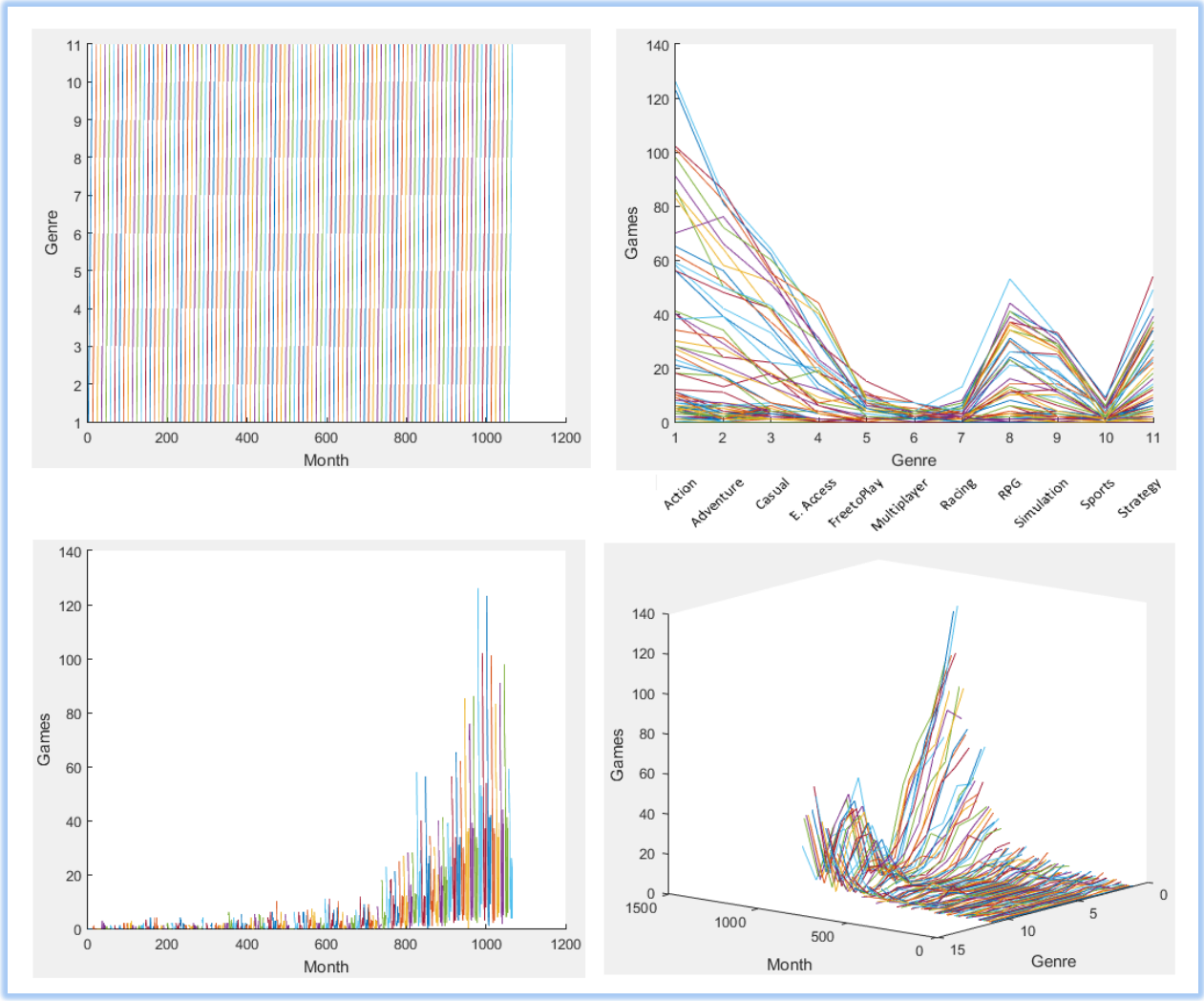


Figure 7: Number of new indie games by genres from February 2008 to February 2016.

I tried least square curve fitting for this data, seeing how it converges to a degree 4 or larger polynomial function. Looking at the pictures above, I think it is best to give less weight to the earlier years in the dataset because of the significant growth in later years. Therefore I used the sum of all monthly amounts throughout 9 years for this calculation. Since each total number is larger than each month by itself, yet still retains the same converging shape, the least square fit for it will be a good estimate for future distribution too.

The data we will look at then has the x-axis as the 11 number-coded game genres and the y-axis as the sum of all monthly amounts of new releases. So there will be 12 data points corresponding to each game genre for a total of 132 observations in this dataset. I attempted curve fitting by formatting the data as described above to find the coefficients of the polynomial using the pseudoinverse. The error of curve fitting is also calculated. I automated this process in Matlab to get the following results:

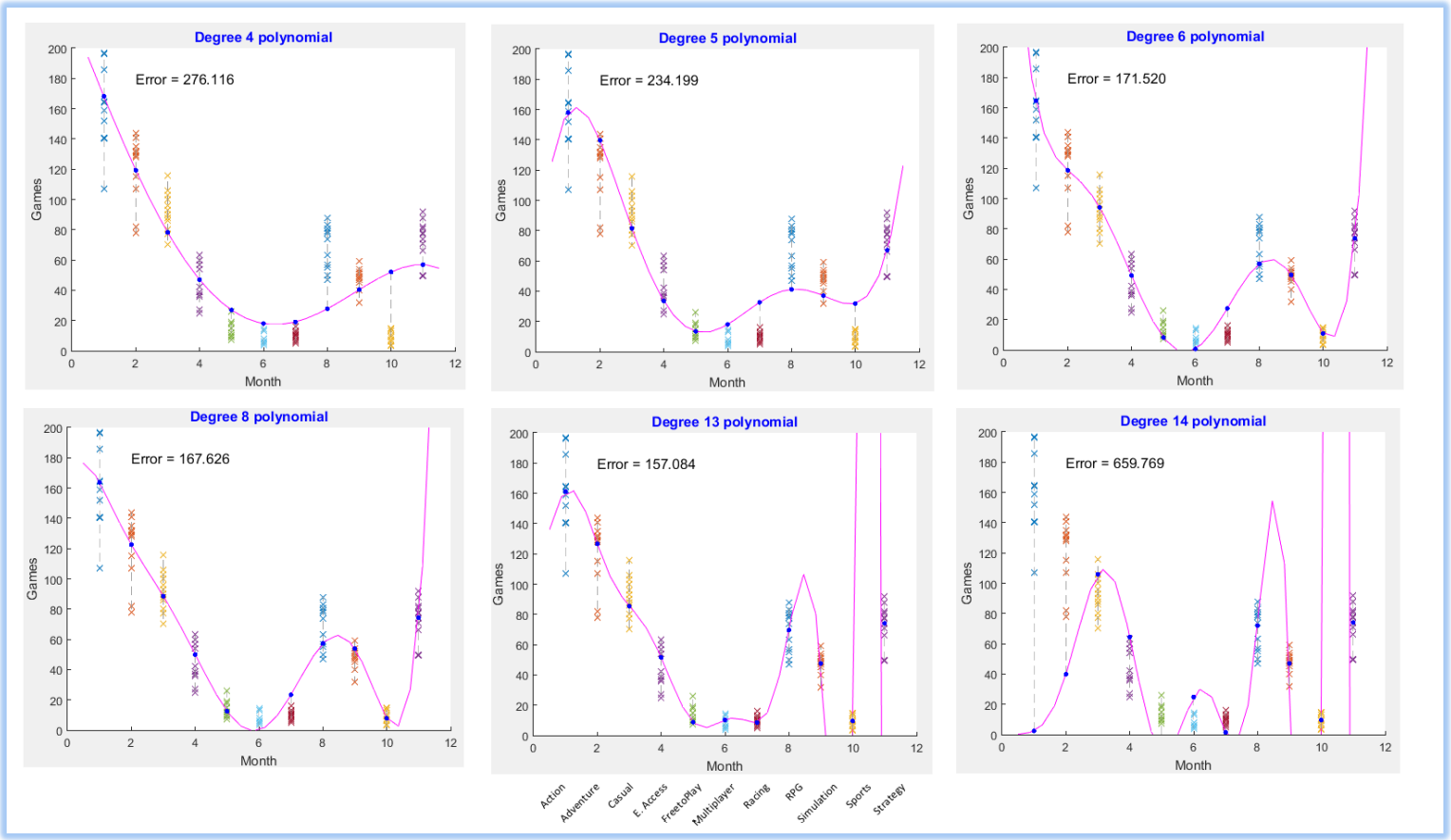


Figure 8: Polynomial curve fitting for total monthly releases from Feb. 2008 to Feb. 2016.

The error decreases as expected when we raise the degree of the polynomial. Going beyond degree 6 is not practical because of diminishing returns, and after degree 14 there is even a surge in error. So the distribution of genres among indie games can be fit reasonably well with the degree 6 polynomial:

$$y = 0.0398x^6 - 1.3497x^5 + 17.4241x^4 - 107.3785x^3 + 328.6871x^2 - 502.1723x + 429.3091$$

2. Predicting distribution of genres

The data for fitting is the total number of games per genre for 9 years, so the curve in the top right plot of Figure 8 may be the distribution for some month in the future. Since every month follows the same trend, we can adjust the constant in the above equation to shift the curve up or down to get the rough estimate of each genre in any month. Then we can guess the whole genre distribution for any month given the count of 1 genre in that month. For example, if Steam introduces 81 new Adventure games in some month, we then shift the curve down 38 units. The formula from curve fitting tells us that in the same

month, there would be 126 Action games, 56 Casual games, and so on. Comparing this to September 2015 where new indie games consisted of 123 Action games, 81 Adventure games, 62 Casual games, etc., we can see that the equation is quite a good approximation.

From this result, we can also extrapolate that with the current growth rate, we will see roughly this distribution for Steam indie games in some month:

Action	Adventure	Casual	E. Access	FreetoPlay	Multiplayer	Racing	RPG	Simulation	Sports	Strategy
165	119	94	49	8	1	27	57	50	11	74

Figure 9: Estimated distribution of indie game genres for some month

To find out when this distribution happens, I attempted to curve fit the growth of indie games by their genres. The assumption is that all genres should have the same growth rate to achieve a converging shape observed in Figure 6. With the projected growth in Figure 9, we may observe this genre distribution within a few months. At that time I can evaluate the accuracy of the modeling method and results in my report.

The procedure in the previous section can be applied here. However, I decided to use Matlab's curve fitting tool because it has better, more interactive visualization and because we do not care about the exact coefficients of the polynomials. Consequently, this section has no Matlab code. The following results should be easy to replicate. I have also recorded the error measurements in Figure 10 below.

It is quite nice that each of the game genres can be fitted with a degree 3 polynomial. However, I suspected that there may be errors with fitting FreetoPlay and MassivelyMultiplayer genres. These two genres went for a long period without a new release and then recently had an uneven boost each month. This is because FreetoPlay was first introduced by Steam in 2013 while my dataset includes data from 2008. For MassivelyMultiplayer genre, perhaps the scopes and challenges of developing online multiplayer games make it daunting for small indie teams, or they did not consider Steam a profitable platform up until recently.

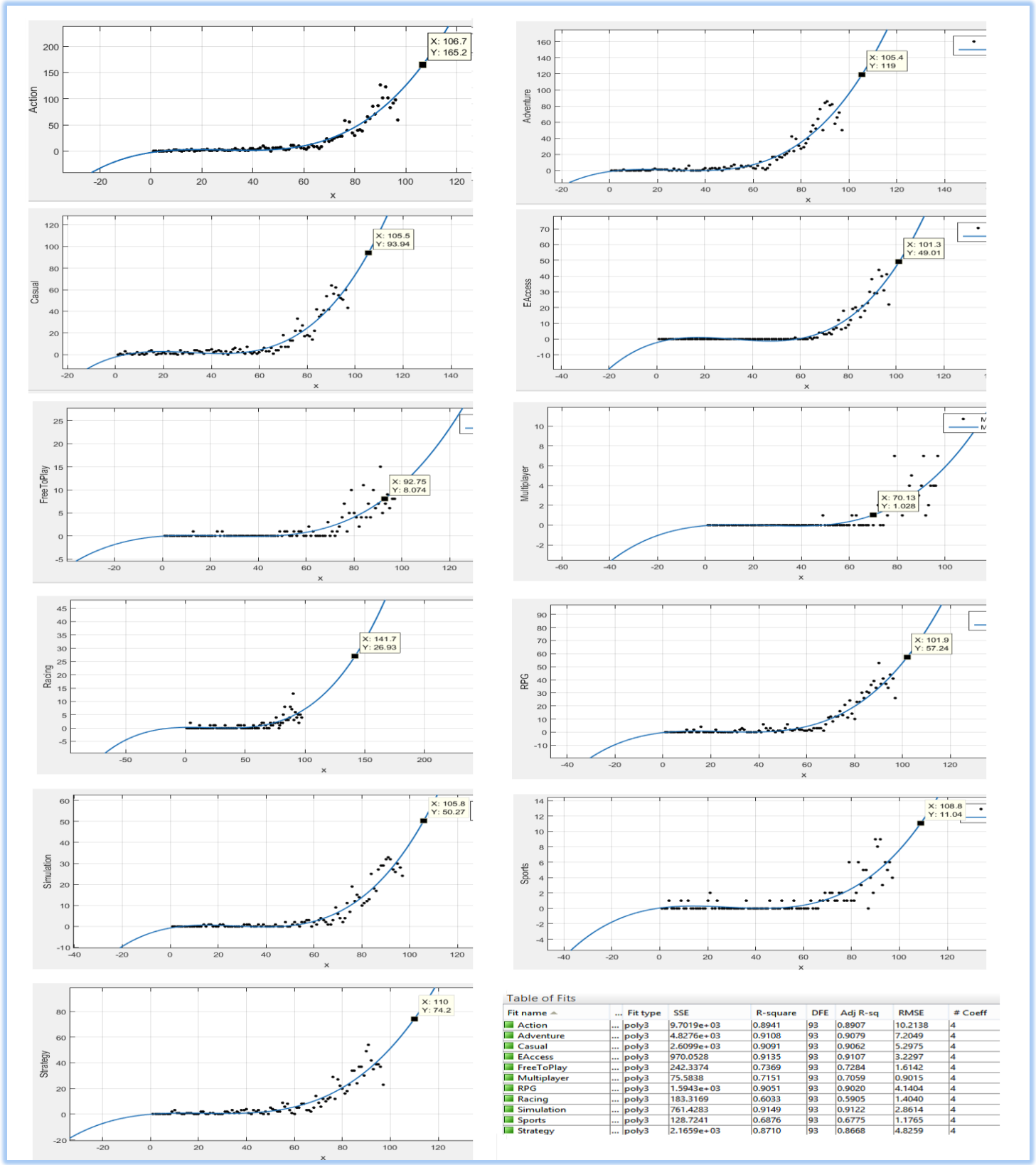


Table of Fits

Fit name	...	Fit type	SSE	R-square	DFE	Adj R-sq	RMSE	# Coeff
Action	...	poly3	9.7019e+03	0.8941	93	0.8907	10.2138	4
Adventure	...	poly3	4.8276e+03	0.9108	93	0.9079	7.2049	4
Casual	...	poly3	2.6099e+03	0.9091	93	0.9062	5.2975	4
EAccess	...	poly3	970.0528	0.9135	93	0.9107	3.2297	4
FreeToPlay	...	poly3	242.3374	0.7369	93	0.7284	1.6142	4
Multiplayer	...	poly3	75.5838	0.7151	93	0.7059	0.9015	4
RPG	...	poly3	1.5943e+03	0.9051	93	0.9020	4.1404	4
Racing	...	poly3	183.3169	0.6033	93	0.5905	1.4040	4
Simulation	...	poly3	761.4283	0.9149	93	0.9122	2.8614	4
Sports	...	poly3	128.7241	0.6876	93	0.6775	1.1765	4
Strategy	...	poly3	2.1659e+03	0.8710	93	0.8668	4.8259	4

Figure 10: Fitting cubic polynomial to the number of indie games in each genre

By looking at the data point with y-value in Figure 10, we can find the number of months when this occurs, which is the x-value. Here are the rounded numbers:

	Action	Adventure	Casual	E. Access	FreetoPlay	Multiplayer	Racing	RPG	Simulation	Sports	Strategy
Distribution	165	119	94	49	8	1	27	57	50	11	74
Months	107	105	106	101	93	70	142	102	106	109	110
Date	Nov, 2016	Sep, 2016	Oct, 2016	May, 2016	Sep, 2015	Oct, 2013	Oct, 2019	Jun, 2016	Oct, 2016	Jan, 2017	Feb, 2017
Ave. Date	Sep, 2016										

Figure 11: Projected dates when distribution in Figure 9 will happen

Results in the Months row are with respect to February 2008. As I thought, the numbers of month for FreetoPlay and MassivelyMultiplayer suggest that the distribution already happened a while ago. They are in fact quite far from the results of other genres. What I did not expect was the large result of Racing. Perhaps the growth of this genre is relatively small compared to the others. Nevertheless, the rest of the results seem consistent. The average date points to September 2016. I concluded that we may observe the distribution in Figure 9 by the end of this year, considering the growth of indie games in general and errors which may occur in projecting dates for FreetoPlay, MassivelyMultiplayer, and Racing genres.

d. Principal component analysis

1. Dealing with missing data

The variables in my dataset are not completely independent from each other. For example, ID and ReleaseYear should vary together. So I attempted PCA to see the correlation between variables. Since each variable has different units and magnitudes, I chose to use the correlation matrix. However, doing a PCA for this project was rather problematic. My dataset has missing data in 4 columns, but the PCA procedure we have studied in class does not cover cases like this.

I tried the analysis with Matlab's ppca function, which does probabilistic PCA to account for missing data. Unfortunately I got an error message saying that the matrix is not positive definite. This is Matlab's way of saying that the result is unusable, according to the documentation. Trying with option 'Rows','pairwise' for the pca function also got me the same error message. So I tried several approaches to get over this problem.

	ID	Days	Price	UserScore	MetaScore	Owners	Playtime	MedianPl	GenreCou	SalesPerD	CharCount	SpecChar	Rel. Mont	Rel. Year	Action	Adventure	Casual	Early Acce	Free 2 pla	Mmulti	Racing	RPG	Simulatio	Sports	Strategy
ID	1.00	-0.91	0.15	0.02	-0.13	-0.27	0.00	0.17	0.09	0.06	-0.02	-0.09	0.04	0.90	0.02	0.19	-0.18	NaN	-0.01	0.03	-0.02	0.06	0.10	0.00	-0.02
Days	-0.91	1.00	-0.27	-0.03	0.12	0.26	-0.06	-0.24	-0.14	-0.13	0.03	0.11	-0.04	-0.99	-0.06	-0.18	0.18	NaN	0.02	-0.02	0.00	-0.09	-0.11	-0.04	-0.01
Price	0.15	-0.27	1.00	0.07	0.13	0.00	0.37	0.44	0.04	0.19	0.06	0.03	0.06	0.26	-0.05	0.01	-0.14	NaN	-0.20	-0.08	0.02	0.06	0.15	0.05	0.13
UserScore	0.02	-0.03	0.07	1.00	0.60	0.23	0.11	0.21	-0.04	0.16	0.01	0.01	0.04	0.02	0.06	0.06	0.02	NaN	-0.06	-0.11	0.04	0.02	-0.09	-0.04	-0.15
MetaScore	-0.13	0.12	0.13	0.60	1.00	0.27	0.17	0.21	0.05	0.21	0.07	0.06	-0.02	-0.12	0.04	0.01	0.03	NaN	-0.06	-0.01	0.04	0.05	-0.06	0.04	0.00
Owners	-0.27	0.26	0.00	0.23	0.27	1.00	0.39	0.30	0.09	0.46	-0.02	-0.02	0.03	-0.27	0.08	-0.04	-0.02	NaN	0.16	0.15	0.01	0.08	0.03	0.01	0.00
Playtime	0.00	-0.06	0.37	0.11	0.17	0.39	1.00	0.81	0.22	0.31	0.11	0.02	0.06	0.05	-0.08	-0.13	-0.10	NaN	0.07	0.11	0.03	0.27	0.23	0.16	0.28
MedianPlaytin	0.17	-0.24	0.44	0.21	0.21	0.30	0.81	1.00	0.19	0.38	0.08	0.03	0.05	0.23	-0.11	-0.01	-0.10	NaN	-0.06	-0.02	0.01	0.25	0.20	0.11	0.24
GenreCount	0.09	-0.14	0.04	-0.04	0.05	0.09	0.22	0.19	1.00	0.08	0.04	-0.04	-0.06	0.15	0.39	0.30	0.27	NaN	0.22	0.17	0.12	0.45	0.27	0.15	0.37
SalesPerDay	0.06	-0.13	0.19	0.16	0.21	0.46	0.31	0.38	0.08	1.00	0.01	-0.03	-0.04	0.13	-0.02	-0.03	-0.05	NaN	0.08	0.07	0.05	0.10	0.10	0.06	0.05
CharCount	-0.02	0.03	0.06	0.01	0.07	-0.02	0.11	0.08	0.04	0.01	1.00	0.54	0.01	-0.03	0.00	0.00	-0.06	NaN	0.01	-0.03	-0.06	0.07	0.03	0.06	0.04
SpecCharCoun	-0.09	0.11	0.03	0.01	0.06	-0.02	0.02	0.03	-0.04	-0.03	0.54	1.00	0.02	-0.11	-0.01	0.03	-0.07	NaN	-0.01	-0.04	-0.05	0.03	-0.06	0.02	-0.01
ReleaseMonth	0.04	-0.04	0.06	0.04	-0.02	0.03	0.06	0.05	-0.06	-0.04	0.01	0.02	1.00	-0.12	-0.04	-0.01	-0.08	NaN	0.00	-0.02	0.04	0.04	-0.01	-0.02	-0.05
ReleaseYear	0.90	-0.99	0.26	0.02	-0.12	-0.27	0.05	0.23	0.15	0.13	-0.03	-0.11	-0.12	1.00	0.06	0.18	-0.16	NaN	-0.02	0.02	0.00	0.08	0.11	0.04	0.02
Action	0.02	-0.06	-0.05	0.06	0.04	0.08	-0.08	-0.11	0.39	-0.02	0.00	-0.01	-0.04	0.06	1.00	-0.08	-0.12	NaN	0.05	0.01	0.07	0.03	-0.09	0.02	-0.13
Adventure	0.19	-0.18	0.01	0.06	0.01	-0.04	-0.13	-0.01	0.30	-0.03	0.00	0.03	-0.01	0.18	-0.08	1.00	0.04	NaN	-0.04	-0.01	-0.10	0.03	-0.17	-0.13	-0.27
Casual	-0.18	0.18	-0.14	0.02	0.03	-0.02	-0.10	-0.10	0.27	-0.05	-0.06	-0.07	-0.08	-0.16	-0.12	0.04	1.00	NaN	-0.01	-0.04	0.00	-0.10	0.03	-0.06	-0.02
Early Access	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Free 2 play	-0.01	0.02	-0.20	-0.06	-0.06	0.16	0.07	-0.06	0.22	0.08	0.01	-0.01	0.00	-0.02	0.05	-0.04	-0.01	NaN	1.00	0.40	-0.02	0.11	0.01	0.06	0.02
Mmulti	0.03	-0.02	-0.08	-0.11	-0.01	0.15	0.11	-0.02	0.17	0.07	-0.03	-0.04	-0.02	0.02	0.01	-0.01	-0.04	NaN	0.40	1.00	-0.01	0.07	0.02	-0.01	0.02
Racing	-0.02	0.00	0.02	0.04	0.04	0.01	0.03	0.01	0.12	0.05	-0.06	-0.05	0.04	0.00	0.07	-0.10	0.00	NaN	-0.02	-0.01	1.00	-0.07	0.01	0.22	-0.06
RPG	0.06	-0.09	0.06	0.02	0.05	0.08	0.27	0.25	0.45	0.10	0.07	0.03	0.04	0.08	0.03	0.03	-0.10	NaN	0.11	0.07	-0.07	1.00	-0.08	-0.07	0.14
Simulation	0.10	-0.11	0.15	-0.09	-0.06	0.03	0.23	0.20	0.27	0.10	0.03	-0.06	-0.01	0.11	-0.09	-0.17	0.03	NaN	0.01	0.02	0.01	-0.08	1.00	0.10	0.20
Sports	0.00	-0.04	0.05	-0.04	0.04	0.01	0.16	0.11	0.15	0.06	0.06	0.02	-0.02	0.04	0.02	-0.13	-0.06	NaN	0.06	-0.01	0.22	-0.07	0.10	1.00	0.06
Strategy	-0.02	-0.01	0.13	-0.15	0.00	0.00	0.28	0.24	0.37	0.05	0.04	-0.01	-0.05	0.02	-0.13	-0.27	-0.02	NaN	0.02	0.02	-0.06	0.14	0.20	0.06	1.00

Figure 12: Correlation matrix of dataset with only valid entries

- The first approach is to consider only valid data. Valid entries are non-zero only. This shrinks the dataset to 757 rows, which is 20% of the original observations. I proceeded to find the correlation matrix of it, which is listed in Figure 12.

There seems to be something wrong with the EarlyAccess row and column because they have NaN values. It turns out that this result is normal. EarlyAccess is just Steam's way of saying games with this label are still being developed, but developers choose to put them on the store for feedbacks and funding to complete the games. Consequently, all EarlyAccess games are not technically released yet and therefore do not have a metacritic score. Since there is no information about them in the valid dataset, it follows that the correlation matrix will have NaN values. The other entries seem fine. We can quickly see that ID and ReleaseYear are indeed positively correlated. I decided to try other approaches to see the differences.

- The second approach is to fill the missing data with zeros. With this, all 4137 observations in the dataset will be used to calculate the correlation matrix:

	ID	Days	Price	UserScore	MetaScor	Owners	Playtime	MedianPl	GenreCou	SalesPerD	CharCoun	SpecChar	Rel. Mont	Rel. Year	Action	Adventure	Casual	Early Acce	Free 2 pla	Mmulti	Racing	RPG	Simulatio	Sports	Strategy
ID	1.00	-0.92	-0.16	-0.36	-0.40	-0.26	-0.11	0.08	0.14	-0.03	0.02	-0.03	0.02	0.90	0.03	0.10	0.04	0.15	0.03	0.03	-0.02	0.04	0.06	-0.01	-0.03
Days	-0.92	1.00	0.04	0.29	0.34	0.23	0.06	-0.10	-0.16	-0.03	-0.01	0.03	-0.03	-0.98	-0.06	-0.11	0.00	-0.16	-0.04	-0.04	0.01	-0.05	-0.08	0.00	0.01
Price	-0.16	0.04	1.00	0.18	0.30	0.04	0.35	0.26	0.06	0.02	0.03	0.00	0.04	-0.05	-0.04	0.06	-0.20	0.16	-0.27	-0.04	0.02	0.09	0.17	0.05	0.13
UserScore	-0.36	0.29	0.18	1.00	0.36	0.24	0.20	0.01	-0.01	0.18	-0.04	0.00	0.01	-0.29	-0.01	0.01	-0.11	-0.05	0.12	0.03	-0.01	0.04	0.02	0.00	0.00
MetaScore	-0.40	0.34	0.30	0.36	1.00	0.23	0.13	0.02	-0.14	0.11	-0.05	-0.01	-0.01	-0.33	-0.02	0.06	-0.14	-0.19	-0.08	-0.05	-0.02	-0.03	-0.07	-0.01	0.02
Owners	-0.26	0.23	0.04	0.24	0.23	1.00	0.29	0.08	0.06	0.60	-0.03	-0.01	0.02	-0.23	0.04	-0.02	-0.05	-0.01	0.24	0.13	-0.01	0.03	0.02	0.00	0.00
Playtime	-0.11	0.06	0.35	0.20	0.13	0.29	1.00	0.61	0.08	0.22	0.04	0.01	0.02	-0.07	-0.10	-0.05	-0.07	0.02	0.00	0.11	-0.02	0.15	0.12	0.04	0.18
MedianPlaytin	0.08	-0.10	0.26	0.01	0.02	0.08	0.61	1.00	0.03	0.06	0.03	-0.01	0.01	0.10	-0.09	0.01	-0.02	0.02	-0.11	0.00	-0.02	0.11	0.07	0.01	0.09
GenreCount	0.14	-0.16	0.06	-0.01	-0.14	0.06	0.08	0.03	1.00	0.11	0.01	0.02	-0.01	0.16	0.31	0.35	0.29	0.44	0.23	0.26	0.14	0.42	0.38	0.19	0.34
SalesPerDay	-0.03	-0.03	0.02	0.18	0.11	0.60	0.22	0.06	0.11	1.00	-0.01	0.00	-0.02	0.03	0.01	-0.01	-0.01	0.02	0.32	0.13	0.00	0.02	0.08	0.02	0.01
CharCount	0.02	-0.01	0.03	-0.04	-0.05	-0.03	0.04	0.03	0.01	-0.01	1.00	0.51	0.01	0.01	-0.09	0.07	0.01	-0.06	0.02	0.00	-0.04	0.07	0.03	0.03	0.01
SpecCharCoun	-0.03	0.03	0.00	0.00	-0.01	-0.01	0.01	-0.01	0.02	0.00	0.51	1.00	0.01	-0.04	-0.03	0.05	-0.01	-0.01	0.01	-0.01	-0.03	0.03	0.01	0.02	0.02
ReleaseMonth	0.02	-0.03	0.04	0.01	-0.01	0.02	0.02	0.01	-0.01	-0.02	0.01	0.01	1.00	-0.18	0.00	0.00	-0.04	0.03	-0.02	-0.02	0.00	0.01	-0.01	0.01	0.00
ReleaseYear	0.90	-0.98	-0.05	-0.29	-0.33	-0.23	-0.07	0.10	0.16	0.03	0.01	-0.04	-0.18	1.00	0.05	0.11	0.00	0.15	0.04	0.04	-0.01	0.05	0.08	-0.01	-0.01
Action	0.03	-0.06	-0.04	-0.01	-0.02	0.04	-0.10	-0.09	0.31	0.01	-0.09	-0.03	0.00	0.05	1.00	-0.03	-0.14	0.13	0.00	0.03	0.05	-0.04	-0.11	0.00	-0.15
Adventure	0.10	-0.11	0.06	0.01	0.06	-0.02	-0.05	0.01	0.35	-0.01	0.07	0.05	0.00	0.11	-0.03	1.00	-0.01	0.02	-0.04	0.01	-0.08	0.17	-0.07	-0.09	-0.16
Casual	0.04	0.00	-0.20	-0.11	-0.14	-0.05	-0.07	-0.02	0.29	-0.01	0.01	-0.01	-0.04	0.00	-0.14	-0.01	1.00	-0.09	0.02	-0.02	0.00	-0.08	0.07	0.02	0.03
Early Access	0.15	-0.16	0.16	-0.05	-0.19	-0.01	0.02	0.02	0.44	0.02	-0.06	-0.01	0.03	0.15	0.13	0.02	-0.09	1.00	0.02	0.11	0.04	0.09	0.14	0.06	0.07
Free 2 play	0.03	-0.04	-0.27	0.12	-0.08	0.24	0.00	-0.11	0.23	0.32	0.02	0.01	-0.02	0.04	0.00	-0.04	0.02	0.02	1.00	0.29	-0.02	0.03	0.01	0.03	0.04
Mmulti	0.03	-0.04	-0.04	0.03	-0.05	0.13	0.11	0.00	0.26	0.13	0.00	-0.01	-0.02	0.04	0.03	0.01	-0.02	0.11	0.29	1.00	0.01	0.11	0.03	0.01	0.02
Racing	-0.02	0.01	0.02	-0.01	-0.02	-0.01	-0.02	-0.02	0.14	0.00	-0.04	-0.03	0.00	-0.01	0.05	-0.08	0.00	0.04	-0.02	0.01	1.00	-0.07	0.06	0.26	-0.07
RPG	0.04	-0.05	0.09	0.04	-0.03	0.03	0.15	0.11	0.42	0.02	0.07	0.03	0.01	0.05	-0.04	0.17	-0.08	0.09	0.03	0.11	-0.07	1.00	-0.01	-0.05	0.09
Simulation	0.06	-0.08	0.17	0.02	-0.07	0.02	0.12	0.07	0.38	0.08	0.03	0.01	-0.01	0.08	-0.11	-0.07	0.07	0.14	0.01	0.03	0.06	-0.01	1.00	0.14	0.19
Sports	-0.01	0.00	0.05	0.00	-0.01	0.00	0.04	0.01	0.19	0.02	0.03	0.02	0.01	-0.01	0.00	-0.09	0.02	0.06	0.03	0.01	0.26	-0.05	0.14	1.00	0.00
Strategy	-0.03	0.01	0.13	0.00	0.02	0.00	0.18	0.09	0.34	0.01	0.01	0.02	0.00	-0.01	-0.15	-0.16	0.03	0.07	0.04	0.02	-0.07	0.09	0.19	0.00	1.00

Figure 13: Correlation matrix of dataset with zeros for missing data

The values for this matrix are a little different from the values in Figure 12. Nevertheless, the existing correlations between variables are still intact, and we do not have NaN for EarlyAccess row and column, which may be considered an improvement.

- The last approach is to adjust for missing data. I find the average of the valid entries of Userscore, multiply it with 4137, subtract it with the sum of all valid entries, and then divide it by the number of observations with missing data in Userscore. This result is then copied to all missing data to ensure that the average of all observations is the same as the average of valid ones. This is acceptable if we assume that the valid observations are representative of the whole dataset. I did the same for the other columns. The percentage of missing data for Userscore, Metascore, Playtime and Median Playtime are 38%, 80%, 3%, 3% respectively.

	ID	Days	Price	UserScore	MetaScor	Owners	Playtime	MedianPl	GenreCou	SalesPerD	CharCount	SpecChar	Rel. Mont	Rel. Year	Action	Adventure	Casual	Early Acce	Free 2 pla	Mmulti	Racing	RPG	Simulatio	Sports	Strategy
ID	1.00	-0.92	-0.16	-0.01	-0.06	-0.26	-0.09	0.10	0.14	-0.03	0.02	-0.03	0.02	0.90	0.03	0.10	0.04	0.15	0.03	0.03	-0.02	0.04	0.06	-0.01	-0.03
Days	-0.92	1.00	0.04	0.00	0.06	0.23	0.05	-0.12	-0.16	-0.03	-0.01	0.03	-0.03	-0.98	-0.06	-0.11	0.00	-0.16	-0.04	-0.04	0.01	-0.05	-0.08	0.00	0.01
Price	-0.16	0.04	1.00	0.11	0.07	0.04	0.35	0.25	0.06	0.02	0.03	0.00	0.04	-0.05	-0.04	0.06	-0.20	0.16	-0.27	-0.04	0.02	0.09	0.17	0.05	0.13
UserScore	-0.01	0.00	0.11	1.00	0.26	0.13	0.13	0.10	-0.06	0.06	0.01	0.01	0.01	0.00	-0.02	0.02	-0.01	0.00	-0.08	-0.12	0.01	0.01	-0.08	0.00	-0.04
MetaScore	-0.06	0.06	0.07	0.26	1.00	0.17	0.09	0.07	0.02	0.12	0.03	0.02	0.00	-0.06	0.01	0.01	0.01	0.00	-0.01	0.00	0.02	0.03	-0.02	0.02	0.00
Owners	-0.26	0.23	0.04	0.13	0.17	1.00	0.29	0.07	0.06	0.60	-0.03	-0.01	0.02	-0.23	0.04	-0.02	-0.05	-0.01	0.24	0.13	-0.01	0.03	0.02	0.00	0.00
Playtime	-0.09	0.05	0.35	0.13	0.09	0.29	1.00	0.60	0.09	0.22	0.04	0.01	0.02	-0.05	-0.10	-0.05	-0.06	0.03	0.00	0.11	-0.02	0.15	0.12	0.04	0.18
MedianPlaytin	0.10	-0.12	0.25	0.10	0.07	0.07	0.60	1.00	0.03	0.06	0.04	-0.01	0.01	0.11	-0.09	0.01	-0.01	0.02	-0.11	-0.01	-0.02	0.11	0.06	0.01	0.09
GenreCount	0.14	-0.16	0.06	-0.06	0.02	0.06	0.09	0.03	1.00	0.11	0.01	0.02	-0.01	0.16	0.31	0.35	0.29	0.44	0.23	0.26	0.14	0.42	0.38	0.19	0.34
SalesPerDay	-0.03	-0.03	0.02	0.06	0.12	0.60	0.22	0.06	0.11	1.00	-0.01	0.00	-0.02	0.03	0.01	-0.01	-0.01	0.02	0.32	0.13	0.00	0.02	0.08	0.02	0.01
CharCount	0.02	-0.01	0.03	0.01	0.03	-0.03	0.04	0.04	0.01	-0.01	1.00	0.51	0.01	0.01	-0.09	0.07	0.01	-0.06	0.02	0.00	-0.04	0.07	0.03	0.03	0.01
SpecCharCoun	-0.03	0.03	0.00	0.01	0.02	-0.01	0.01	-0.01	0.02	0.00	0.51	1.00	0.01	-0.04	-0.03	0.05	-0.01	-0.01	0.01	-0.01	-0.03	0.03	0.01	0.02	0.02
ReleaseMonth	0.02	-0.03	0.04	0.01	0.00	0.02	0.02	0.01	-0.01	-0.02	0.01	0.01	1.00	-0.18	0.00	0.00	-0.04	0.03	-0.02	-0.02	0.00	0.01	-0.01	0.01	0.00
ReleaseYear	0.90	-0.98	-0.05	0.00	-0.06	-0.23	-0.05	0.11	0.16	0.03	0.01	-0.04	-0.18	1.00	0.05	0.11	0.00	0.15	0.04	0.04	-0.01	0.05	0.08	-0.01	-0.01
Action	0.03	-0.06	-0.04	-0.02	0.01	0.04	-0.10	-0.09	0.31	0.01	-0.09	-0.03	0.00	0.05	1.00	-0.03	-0.14	0.13	0.00	0.03	0.05	-0.04	-0.11	0.00	-0.15
Adventure	0.10	-0.11	0.06	0.02	0.01	-0.02	-0.05	0.01	0.35	-0.01	0.07	0.05	0.00	0.11	-0.03	1.00	-0.01	0.02	-0.04	0.01	-0.08	0.17	-0.07	-0.09	-0.16
Casual	0.04	0.00	-0.20	-0.01	0.01	-0.05	-0.06	-0.01	0.29	-0.01	0.01	-0.01	-0.04	0.00	-0.14	-0.01	1.00	-0.09	0.02	-0.02	0.00	-0.08	0.07	0.02	0.03
Early Access	0.15	-0.16	0.16	0.00	0.00	-0.01	0.03	0.02	0.44	0.02	-0.06	-0.01	0.03	0.15	0.13	0.02	-0.09	1.00	0.02	0.11	0.04	0.09	0.14	0.06	0.07
Free 2 play	0.03	-0.04	-0.27	-0.08	-0.01	0.24	0.00	-0.11	0.23	0.32	0.02	0.01	-0.02	0.04	0.00	-0.04	0.02	0.02	1.00	0.29	-0.02	0.03	0.01	0.03	0.04
Mmulti	0.03	-0.04	-0.04	-0.12	0.00	0.13	0.11	-0.01	0.26	0.13	0.00	-0.01	-0.02	0.04	0.03	0.01	-0.02	0.11	0.29	1.00	0.01	0.11	0.03	0.01	0.02
Racing	-0.02	0.01	0.02	0.01	0.02	-0.01	-0.02	-0.02	0.14	0.00	-0.04	-0.03	0.00	-0.01	0.05	-0.08	0.00	0.04	-0.02	0.01	1.00	-0.07	0.06	0.26	-0.07
RPG	0.04	-0.05	0.09	0.01	0.03	0.03	0.15	0.11	0.42	0.02	0.07	0.03	0.01	0.05	-0.04	0.17	-0.08	0.09	0.03	0.11	-0.07	1.00	-0.01	-0.05	0.09
Simulation	0.06	-0.08	0.17	-0.08	-0.02	0.02	0.12	0.06	0.38	0.08	0.03	0.01	-0.01	0.08	-0.11	-0.07	0.07	0.14	0.01	0.03	0.06	-0.01	1.00	0.14	0.19
Sports	-0.01	0.00	0.05	0.00	0.02	0.00	0.04	0.01	0.19	0.02	0.03	0.02	0.01	-0.01	0.00	-0.09	0.02	0.06	0.03	0.01	0.26	-0.05	0.14	1.00	0.00
Strategy	-0.03	0.01	0.13	-0.04	0.00	0.00	0.18	0.09	0.34	0.01	0.01	0.02	0.00	-0.01	-0.15	-0.16	0.03	0.07	0.04	0.02	-0.07	0.09	0.19	0.00	1.00

Figure 14: Correlation matrix of dataset with adjusted values for missing data

The entries for this correlation matrix are only different from Figure 13 where adjusted values are involved. Any correlations between variables are still intact. I decided to continue the PCA with this matrix because it seems like an improvement from the first 2 methods.

As seen in the color coded matrix, darker shades of blue represent negative correlation, and brighter shades represent positive correlation. Obvious correlations are (ID 0.90 ReleaseYear): indie games which were released years ago would have smaller IDs on Steam store and vice versa; (Days -0.98 ReleaseYear): if the games were on the market for a short time then their release years must be recent and vice versa. Here are the notable results:

- (Days 0.23 Owners): recent releases have fewer sales, but this is just a weak correlation.
- (ID -0.16 Price): games with higher Steam ID number have lower price; weak correlation.
- (Price 0.11 Userscore): higher priced games tend to have higher userscore; weak.
- (Userscore 0.26 Metascore): games with high userscore tend to have high metacritic score; weak.
- (Userscore 0.13 Owners) and (Metascore 0.17 Owners): acclaimed games have higher sales; weak.
- (Price 0.35 Playtime) and (Price 0.25 MedianPlaytime): players spend more time on higher priced games; moderately weak.
- (Userscore 0.13 Playtime): higher userscore may lead to higher playtime; weak.
- (Owners 0.29 Playtime): games with many owners tend to have higher playtime; moderately weak.
- (Playtime 0.60 MedianPlaytime): higher playtime may mean higher median playtime; moderately strong.
- (ID 0.14 GenreCount): games with higher Steam ID number tend to be labeled with many genres; weak.
- (Days -0.16 GenreCount): older games have fewer genre labels; weak.
- (Metascore 0.12 SalesPerDay): higher metacritic score means higher sales per day; weak.
- (Owners 0.60 SalesPerDay): games with many owners have higher sales per day; moderately strong.
- (Playtime 0.22 SalesPerday): higher playtime means higher sales per day; weak.
- (GenreCount 0.11 SalesPerDay): games that have many genres have higher sales per day; weak.
- (CharCount 0.51 SpecCharCount): games that have long titles tend to have special characters in titles; moderately strong.
- (Owners -0.23 ReleaseYear): older games have sold more copies; weak.
- (MedianPlaytime 0.11 ReleaseYear): recent games have higher median playtime; weak.
- (GenreCount 0.16 ReleaseYear): recent games have more genres; weak.
- (ReleaseMonth -0.18 ReleaseYear): games released at the end of the year tend to be older releases; weak.

- (Days -0.11 Adventure) and (ReleaseYear 0.11 Adventure): more recent releases are in the adventure genre; weak.
- (Price -0.20 Casual): casual games have lower prices; weak.
- (ID 0.15 EarlyAccess): games with higher ID tend to be on Early Access; weak.
- (Days -0.16 EarlyAccess) and (ReleaseYear 0.15 EarlyAccess): Early Access games are newer releases; weak.
- (Price 0.16 EarlyAccess): Early Access games tend to have higher prices; weak.
- (Action 0.13 EarlyAccess): Action games tend to be on Early Access; weak.
- (Price -0.27 FreetoPlay): free to play games have lower prices; moderately weak. Result debatable as explained above.
- (Owners 0.24 FreetoPlay): free to play games have more owners; weak.
- (MedianPlaytime -0.11 FreetoPlay): free to play games have lower median playtime; weak.
- (SalesPerDay 0.32 FreetoPlay): free to play games have higher sales per day; moderately weak.
- (Userscore -0.12 MassiveMultiplayer): multiplayer games have lower userscore; weak.
- (Owners 0.13 MassiveMultiplayer): multiplayer games tend to have more owners; weak.
- (Playtime 0.11 MassiveMultiplayer): multiplayer games have higher playtime; weak.
- (SalesPerDay 0.13 MassiveMultiplayer): multiplayer games have higher sales per day; weak.
- (EarlyAccess 0.11 MassiveMultiplayer): multiplayer games tend to be on Early Access; weak.
- (FreetoPlay 0.29 MassiveMultiplayer): multiplayer games tend to be free to play; moderately weak.
- (Playtime 0.15 RPG) and (MedianPlaytime 0.11 RPG): players spend more time on RPG games; weak.
- (Adventure 0.17 RPG): RPG games are also labeled as adventure games; weak.
- (MassiveMultiplayer 0.11 RPG): RPG games are also labeled as massive multiplayer games; weak.
- (Price 0.17 Simulation): simulation games have higher prices; weak.
- (Playtime 0.12 Simulation): simulation games have higher playtime; weak.
- (Action -0.11 Simulation): simulation games are not likely to be action games; weak.
- (EarlyAccess 0.14 Simulation): simulation games tend to be on Early Access; weak.
- (Racing 0.26 Sports): sports games are also labeled as racing games; weak.
- (Simulation 0.14 Sports): sports games are also labeled as simulation games; weak.
- (Price 0.13 Strategy): strategy games tend to have higher prices; weak.
- (Playtime 0.18 Strategy): strategy games have higher playtime; weak.
- (Action -0.15 Strategy): strategy games are not likely to be action games; weak.
- (Adventure -0.16 Strategy): strategy games are not likely to be adventure games; weak.
- (Simulation 0.19 Strategy): strategy games are also labeled as simulation games; weak.
- GenreCount positively correlates with all game genres. Weak correlation for FreetoPlay, Racing and Sports games. Moderately weak correlation for other genres. Strongest correlation for EarlyAccess and then RPG.

Questions I have before starting this project are also answered:

- Apparently there is no correlation between the MSRP price of games and number of sales presumably because Steam games go on sale quite often.
- Correlation between price and genres are only seen with Casual, EarlyAccess, Simulation, and Strategy games. Other game genres do not follow a rule for setting their prices.
- Number of owners is only likely to be higher with FreetoPlay and MassiveMultiplayer games.
- It does not matter what month an indie game is released in, whether it is during the holiday season or summer. In the long run the total number of sales and average sales per day do not depend on the release month.
- Lengthy game titles or unusual symbols in titles do not affect sales. Titles indicating that a game is a sequel will neither increase or decrease sales. Perhaps it is only important for a game to have a memorable name, but this cannot be determined with my dataset.
- Games which are well-received do not necessarily have higher sales. Positive Steam userscores may increase sales and playtime, but it is not guaranteed.

2. PCA projection

With this correlation matrix, I used the SVD decomposition to get its principal components. The matrix has 24 eigenvalues and the 25th eigenvalue is 0. It turns out that the two largest eigenvalues only account for 23% of the variance. To capture 80% variance, we need to preserve the first 13 eigenvectors, which is not very efficient.

The first two principal components are:

	PC1	PC2
ID	**-0.52**	-0.09
Days	**0.53**	0.05
Price	0.05	**0.25**
UserScore	0.04	0.08
MetaScore	0.07	0.14
Owners	0.18	**0.34**
Playtime	0.05	**0.41**
MedianPlaytime	-0.06	**0.27**
GenreCount	-0.22	**0.38**
SalesPerDay	0.03	**0.32**
CharCount	-0.01	0.04
SpecCharCount	0.02	0.04
ReleaseMonth	0.03	0.02
ReleaseYear	**-0.53**	-0.05
Action	-0.06	0.01
Adventure	-0.12	0.05
Casual	-0.04	-0.01
Early Access	-0.17	0.20
Free 2 play	-0.04	0.16
Mmulti	-0.06	0.21
Racing	-0.01	0.04
RPG	-0.09	0.23
Simulation	-0.10	0.23
Sports	-0.02	0.10
Strategy	-0.03	0.22

Figure 15: First two principal components of the dataset

We can see that PC1 emphasizes the time when the game was released. It is easy to understand that this would be the maximum spread because different games have different ID and release dates. I suppose if we hash the game titles to unique values, then the hashed titles would also be in PC1, behaving like another set of game ID. PC2 emphasizes sales info and playtime, which can be interpreted as the players' perceived value of a game.

Projecting the dataset onto these two principal components reveals interesting facts about Steam indie games. Figure 16 shows the scatter plot of all games after we perform dimension reduction. Blue vectors represent variables of the dataset, and their directions and magnitudes indicate how each variable contributes to the principal components.

There appears to be a few outliers in the upper part of Figure 16. They outperformed other games in terms of value/time of release. Here are the 3 indie games with the highest ratios in decreasing order:

- Robocraft: a free to play game, released on July 8, 2014, has userscore of 81% with 9.2 million owners who have played on average 17.5 hours. Genre: Action, Early Access, FreetoPlay, MassivelyMultiplayer, RPG, Simulation.
- Heroes & Generals: a free to play game, released on July 11, 2014, has userscore of 69% with 9 million owners who have played on average 11 hours. Genre: Action, EarlyAccess, FreetoPlay, MassivelyMultiplayer.
- Terraria: original price is $9.99, released on May 16, 2014, has userscore of 96%, metascore of 83% with 7.3 million owners who have played on average 71 hours. Genre: Action, Adventure, RPG.

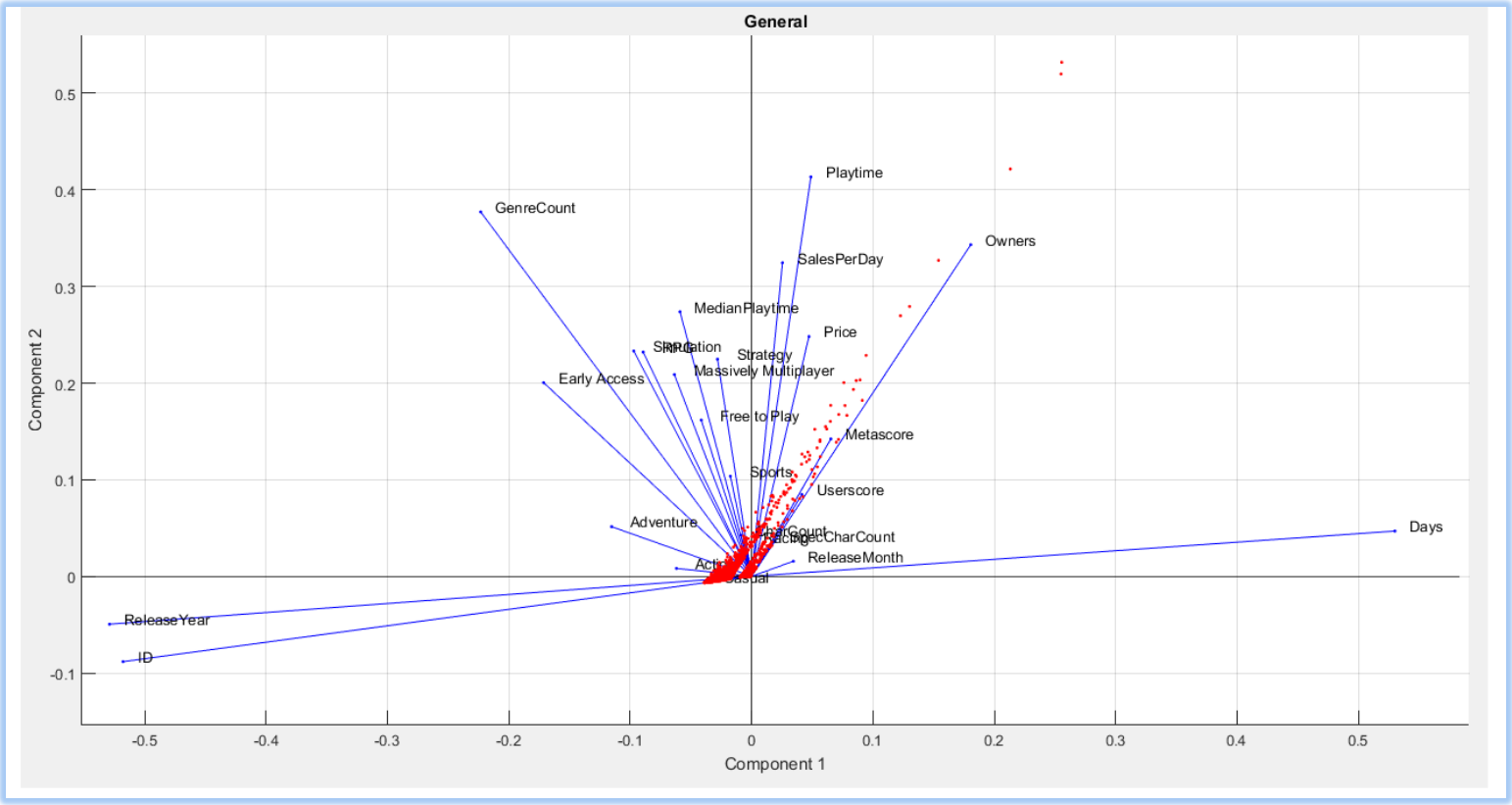


Figure 16: PCA projection of the dataset with corresponding variables

3. Logistic regression

The projected data is not spread out like a cloud seen in random distribution. Instead, the red dots form a thin triangle in this 2D plot. The same shape is observed when we draw the projection for each game genre. I did not include those pictures here, but the code at the end of the report already has a loop to plot all genres. If we take a closer look at the projection, we can see that indie games on Steam are actually distributed into two patches of red dots. There is a clear separation between them, and incidentally, this projected data seems to run along the Owners vector.

I thought this was worth looking into, so I did some PCA runs without the release time variables. My reasoning is that if the projected data in Figure 16 is indeed parallel to the Owners vector, then when we remove some of the horizontal spread, which are variables that dominate in PC1, the new projected

data will be distributed around this vector. Visual results confirm this. Removing variables will give different spreads because they correspond to different PCs, however, the distribution looks mostly the same and in some cases projected data lies directly on the Owners vector. This can be seen in Figure 17:

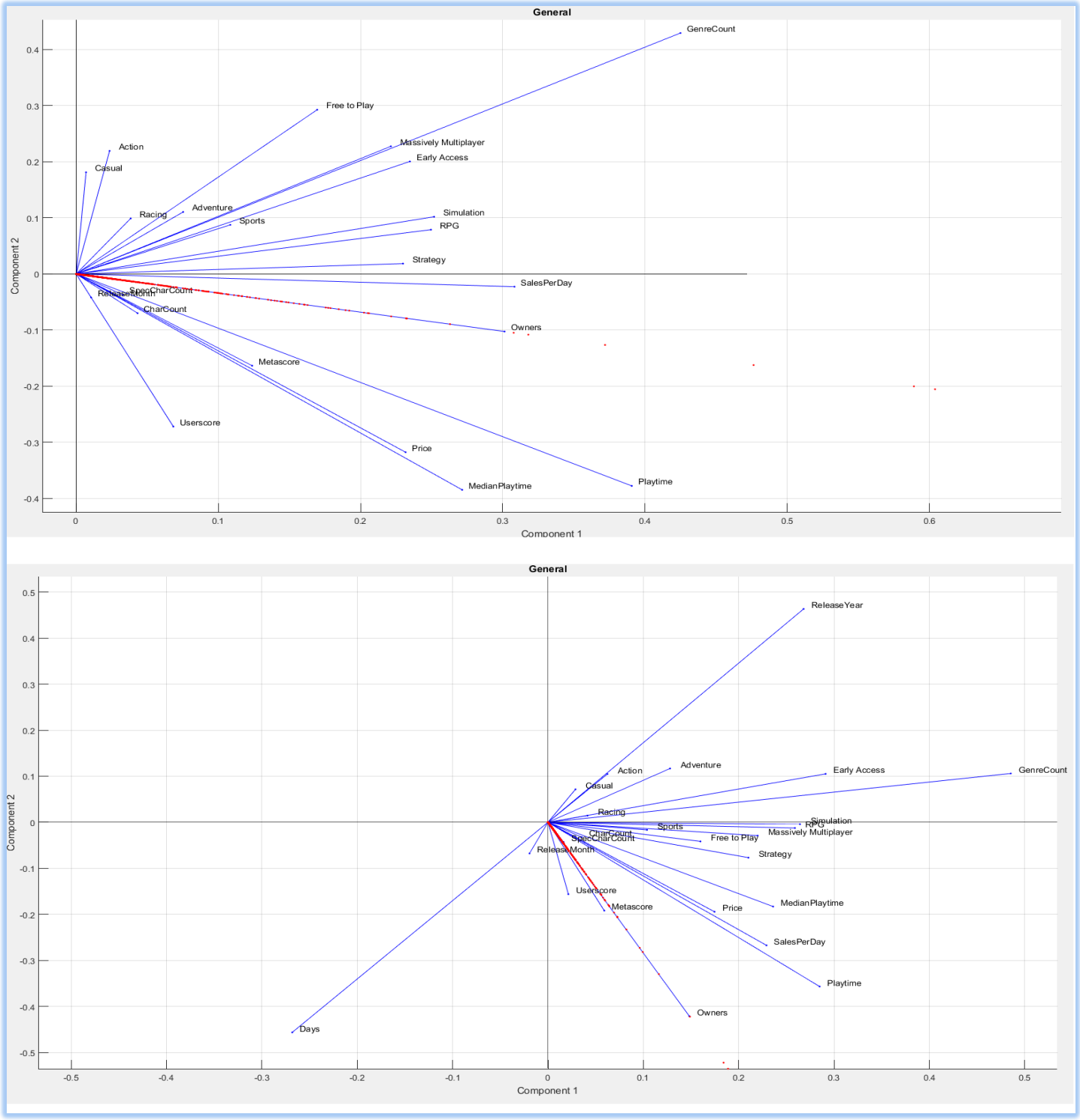


Figure 17: Some PCA projections of the dataset without release time

I am not quite sure how to interpret this. Perhaps the amount of variance captured by projection onto PC1 and PC2 for this particular dataset is closely related to the Owners vector. Nevertheless, there is now a basis to believe that a division line exists for the two patches of projected data. My guess is that the line would have the same slope as the Owners vector, which would be (0.34 value / 0.18 release time). By visually inspecting the graph, I can choose a point roughly in the middle of the gap between two patches. The y-value of this point and the slope of the Owners vector give a straight line that neatly divides the projected data: y = 1.9x + 140000

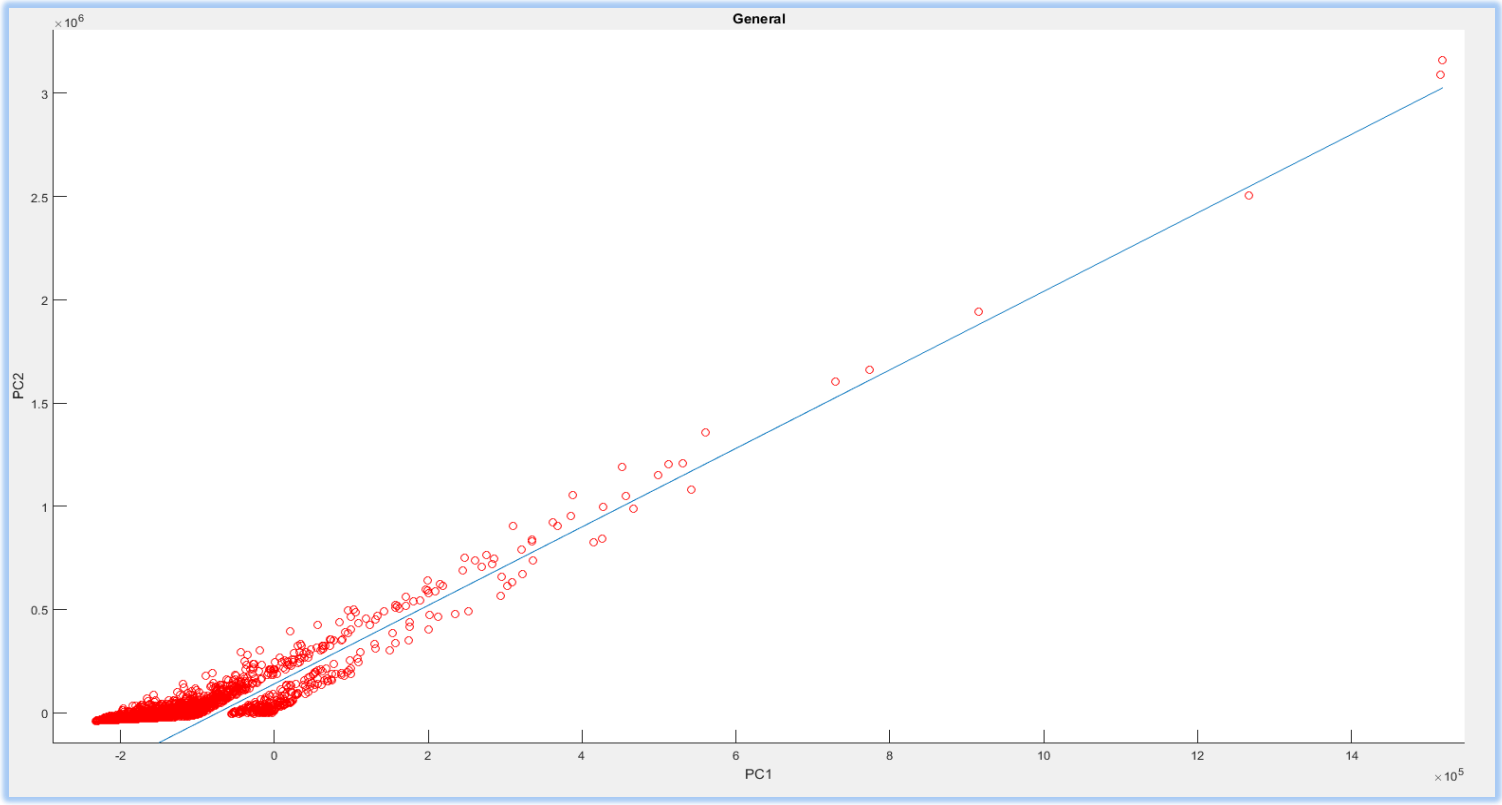


Figure 18: PCA projection of the full dataset with division line

In short, data can be separated into two groups based on their projected values onto the principal components in Figure 15. It is clear from this picture that the higher the projected point is on the y-axis, the higher value it has to the buyers. Therefore, we can build a fast classifier using this crude division line. The group above this line may be called the high performance group, and the low performance group is below the line. The implications here is that given two indie games with sufficiently different measurements but same release window, our classifier can tell which game clearly has more value to players. The code for this classifier is at the end of the report. I picked these two games at random to test the classifier:

```
    game1 = 1.0e+05 *[2.0148 0.0159 0.0000 0.0000 0.0000 3.9565 0.0004 0.0001
0.0000 0.0025 0.0003 0.0000 0.0001 0.0201 0.0000 0 0 0 0 0 0 0.0000 0 0 0];
%released on 24-Oct-11
    game2 = 1.0e+05 *[1.1140 0.0162 0.0000 0.0000 0 1.8061 0.0008 0.0004
0.0000 0.0011 0.0001 0 0.0001 0.0201 0.0000 0 0 0 0 0 0 0 0 0 0]; %released
on 21-Sep-11
```

```
classifier(game1, game2);
```

> Game1 has more value to player

Visual inspection confirms that game1 is above the division line while game2 is below it. I did not include the game titles here out of respect for the developers, but we can verify this information easily on Steam using their appids. This classifier can be a fair method to compare games from the same release window because conditions of the economy and the amount of Steam users are equivalent. The performance metric here also takes into account of genres, so games of different genres can be compared against each other.

The next question is how to increase the value of a game in the low performance group, or of any game at all, under these metrics. Figure 16 shows that almost all vectors point upward, so the answer is to add more to variables left of the Owners vector which are also positive in the y direction and to subtract from the other variables. Unfortunately, not all of them are under the developers' control. So unless developers can add new genres to their games at will, the only easy way to increase value to players is to set a high MSRP, perhaps later reducing price on sales events, and choose a short, memorable name for their game. The solution from this dataset makes sense, albeit only inflating value artificially.

V. Conclusion

I chose this topic because I am interested in video games. After following video game sales for the past 2 years, I want to know more about indie game sales in particular. Indie games used to be overlooked compared to video games from big companies, so this analysis is an effort to understand how they perform and also the implications of developing indie games on PC.

I got half of my dataset from a csv file created by Steam Spy. For the other half, I used the site's API to get indie games' appids to scrape data directly from Steam store and then extracted what I need with Unix commands. Collecting data, testing the correctness of this process together with formatting data for Matlab was the second most time consuming task.

Analysis required even more effort. The dataset has a lot of outliers and missing data. Huge growth of indie games on Steam also made it challenging to remove bias from later years. Processing facts about game titles required heavy use of Unix commands. Linear regression and least square curve fitting were employed. Figuring out how to deal with missing data and to perform PCA on the dataset took quite a bit of time. Aside from computing average sales and play time, I learned a few interesting facts about Steam inde games regarding release dates, genre distribution, and game performance. I was also able to come up with some models for them using the methods we studied in class.

Finishing the report was a little bit stressful because I was short on time. For the figures, I copied results between Matlab and Excel to choose the best way to clearly plot the data, and it was a good lesson in terms of presentation and time management. Writing meaningful discussion was not an easy task either. Fortunately I was able to complete everything in time. After this project, I understood the benefits of PCA and curve fitting more. I also learned that looking at data in different ways can yield different insights. Most importantly, none of my questions before starting the project were left unanswered. All goals in my proposal were completed with correlation between variables described in previous sections. These statistics are not well known, so I hope this analysis will point out a model for success in developing and selling indie games. It would be interesting to see if these results hold true for mainstream PC games on Steam. The techniques I learned here will be helpful for tackling that in a new research.

VI. Matlab code

```
function project()

    GenreStr = {' Action', ' Adventure', ' Casual', 'Early Access', ...
        ' Free to Play', ' Massively Multiplayer', ' Racing', ' RPG', ...
        ' Simulation', ' Sports', ' Strategy'};
    categories = {'ID', 'Days', 'Price', 'Userscore', 'Metascore',...
            'Owners', 'Playtime', 'MedianPlaytime', 'GenreCount',...
            'SalesPerDay', 'CharCount', 'SpecCharCount', 'ReleaseMonth',...
            'ReleaseYear', ' Action', ' Adventure', ' Casual', 'Early Access', ...
        ' Free to Play', ' Massively Multiplayer', ' Racing', ' RPG', ...
        ' Simulation', ' Sports', ' Strategy'};
    names;
    data = csvread('data.csv',1,0);

    ID = data(:,1);
    Days = data(:,2);
    Price = data(:,3);
    UserScore = data(:,4);
    MetaScore = data(:,5);
    Owners = data(:,6);
    Playtime = data(:,7);
    MedianPlaytime = data(:,8);
    Genres = data(:, 9:18);
    GenreCount = data(:, 19);
    SalesPerDay = data(:, 20);
    CharCount = data(:, 21);
    SpecCharCount = data(:, 22);
    ReleaseMonth = data(:,23);
    ReleaseYear = data(:, 24);
    GameGenres = GenreMatrix(Genres, GenreStr);

    NonFreeIndex = Price > 0;
    FreeIndex = ~NonFreeIndex;

    %%%%%%%%%%%%%%%%%%%%

    generalStats(Days, Price, UserScore, MetaScore, ...
        Owners, Playtime, MedianPlaytime, NonFreeIndex, FreeIndex,...
        GenreCount, SalesPerDay, CharCount, SpecCharCount,...
        ReleaseMonth, ReleaseYear);
   countByGenre(Genres, GenreStr);
    statsByGenres(Days,Price, UserScore, MetaScore, ...
        Owners, Playtime, MedianPlaytime, Genres, GenreStr,...
        GenreCount, SalesPerDay, CharCount, SpecCharCount,...
        ReleaseMonth, ReleaseYear);

    UserScore = adjustMissing(UserScore);
    MetaScore = adjustMissing(MetaScore);
    Playtime = adjustMissing(Playtime);
    MedianPlaytime = adjustMissing(MedianPlaytime);

    dataMat = [ ID, Days, Price, UserScore, MetaScore, Owners,  ...
         Playtime, MedianPlaytime, GenreCount, SalesPerDay, CharCount, SpecCharCount,...
        ReleaseMonth, ReleaseYear, GameGenres];

    figure;
    PCAProj(dataMat, categories);
    title('General');
    PCAByGenre(dataMat, Genres, GenreStr, categories);

    releaseMonthByYear(ReleaseYear, ReleaseMonth, GameGenres, GenreStr);
    fitMonthTotal();

    game1 = 1.0e+05 *[2.0148 0.0159 0.0000 0.0000 0.0000 3.9565 0.0004 0.0001 0.0000 0.0025 0.0003 0.0000 0.0001 0.0201 0.0000 0
0 0 0 0 0 0.0000 0 0 0];
    game2 = 1.0e+05 *[1.1140 0.0162 0.0000 0.0000 0 1.8061 0.0008 0.0004 0.0000 0.0011 0.0001 0 0.0001 0.0201 0.0000 0 0 0 0 0 0
0 0 0 0];

    classifier(game1, game2);
end

%%%%classifier for determines which game belongs to which performance group
function classifier(game1, game2)
    PC1 = [-0.5178  0.5300  0.0473  0.0415  0.0653  0.1806  0.0489  -0.0590 -0.2232 0.0255  -0.0089 0.0193  0.0344  -0.5290 -
0.0618 -0.1153 -0.0385 -0.1713 -0.0414 -0.0636 -0.0062 -0.0893 -0.0971 -0.0175 -0.0280]';
    PC2 = [-0.0882  0.0469  0.2482  0.0848  0.1423  0.3433  0.4134  0.2739  0.3772  0.3246  0.0428  0.0362  0.0157  -0.0495
0.0083  0.0515  -0.0059 0.2005  0.1618  0.2089  0.0350  0.2322  0.2333  0.1037  0.2248]';
    x1 = game1 * PC1;
    x2 = game2 * PC1;
    if abs(x1 - x2) > 20000
        disp('Not same release time');
    else
        y = 1.9 * (game1 * PC1) + 140000;
        y1 = game1 * PC2;
        y2 = game2 * PC2;
        if y1 > y && y2 < y
            disp('Game1 has more value to player');
        elseif y1 < y && y2 > y
            disp('Game2 has more value to player');
        else
            disp('Both games belong to the same group');
        end
    end
end
```

```
%%%%game releases by month throughout all years
function releaseMonthByYear(ReleaseYear, ReleaseMonth, GameGenres, GenreStr)
    disp('*****ALL YEARS*****');

    releaseMonth(ReleaseMonth, GameGenres, GenreStr);
    year = unique(ReleaseYear);
    for i = 1:max(size(year))
        index = (ReleaseYear == year(i));
        disp('*****YEAR BY YEAR*****');
        disp(year(i));
        releaseMonth(ReleaseMonth(index), GameGenres(index,:), GenreStr);
    end
end

%%%%game releases by month in a certain year
function releaseMonth(ReleaseMonth, GameGenres, GenreStr)
    disp('*****COUNT BY GENRES*****');

    for i = 1:max(size(GenreStr))
        result = zeros(12,1);
        for j = 1:12
            allMonth(j,1) = sum(ReleaseMonth==j);
            tempGameGenres = GameGenres(ReleaseMonth==j,:);
            result(j) = sum(sum(tempGameGenres(:,i)==1));
        end
        allMonth(:,i+1) = result;
    end
    allMonth
end

%%%%fill zero entries with average value
function result = adjustMissing(data)
    result = data;
    sizeAllData = max(size(data));
    index = data>0;
    meanValidData = mean(data(index));
    sizeValidData = sum(index);
    total = sizeAllData * meanValidData;
    leftover = total - sum(data(index));
    result(~index) = leftover / (sizeAllData - sizeValidData);
end

%%%%construct boolean matrix for 11 game genres
function result = GenreMatrix(Genres, GenreStr)
    M = max(size(Genres));
    N = min(size(Genres));
    result = zeros(M, max(size(GenreStr)));
    for i = 1:M
        for j = 1:N
            if Genres(i, j) > 0
                result(i, Genres(i, j)) = 1;
            end
        end
    end
end

%%%%PCA projection for all game genres
function PCAByGenre(dataMat, Genres, GenreStr, categories)
    for i = 1:max(size(GenreStr))
        tempData = dataMat;
        tempData(:,14+i) = [];
        tempCategories = categories;
        tempCategories(14+i) = [];
        index = indexOfGenre(Genres, i);
        figure;
        PCAProj(tempData(index,:), tempCategories);
        title(GenreStr(i));
    end
end

%%%%PCA projection
function PCAProj(dataMat, categories)
    corrMat = corr(dataMat);
    [U S V] = svd(corrMat);
    PC1 = V(:,1)
    PC2 = V(:,2)
    diag(S);
    X = dataMat * PC1;
    Y = dataMat * PC2;
    xlabel('PC1');
    ylabel('PC2');
    biplot([PC1 PC2],'scores',[X Y],'varlabels',categories);
end

%%%%general statistics for all games by genre
function statsByGenres(Days,Price, UserScore, MetaScore, ...
    Owners, Playtime, MedianPlaytime, Genres, GenreStr,...
        GenreCount, SalesPerDay, CharCount, SpecCharCount,...
        ReleaseMonth, ReleaseYear)
    disp('*****STATS BY GENRES*****');
    for i = 1:max(size(GenreStr))
        statsByGenre(Days,Price, UserScore, MetaScore, ...
            Owners, Playtime, MedianPlaytime, Genres, GenreStr, i,...
        GenreCount, SalesPerDay, CharCount, SpecCharCount,...
        ReleaseMonth, ReleaseYear);
```

```
    end
end

%%%%general statistics for 1 game genre
function statsByGenre(Days,Price, UserScore, MetaScore, ...
    Owners, Playtime, MedianPlaytime, Genres, GenreStr, genreNum,...
        GenreCount, SalesPerDay, CharCount, SpecCharCount,...
        ReleaseMonth, ReleaseYear)
    disp(strcat('*****Stats of ', GenreStr(genreNum), ' games*****'));
    index = indexOfGenre(Genres, genreNum);

    NonFreeIndex = Price(index) > 0;
    FreeIndex = ~NonFreeIndex;

    generalStats(Days(index), Price(index), ...
        UserScore(index), MetaScore(index), Owners(index), ...
        Playtime(index), MedianPlaytime(index), ...
        NonFreeIndex, FreeIndex,...
        GenreCount(index), SalesPerDay(index), CharCount(index),...
        SpecCharCount(index), ReleaseMonth(index), ReleaseYear(index));
end

%%%%filter out other game genres
function result = indexOfGenre(Genres, genreNum)
    result = logical(sum(Genres == genreNum, 2));
end

%%%%count number of games in each genre
function countByGenre(Genres, GenreStr)
    disp('*****COUNT BY GENRES*****');
    for i = 1:max(size(GenreStr))
        disp(strcat('Number of ', GenreStr(i), ' games:'));
        sum(sum(Genres==i))
    end
end

%%%%general statistics for games
function generalStats(Days, Price, UserScore, MetaScore, ...
    Owners, Playtime, MedianPlaytime, NonFreeIndex, FreeIndex,...
        GenreCount, SalesPerDay, CharCount, SpecCharCount,...
        ReleaseMonth, ReleaseYear)
    disp('*****GENREAL STATS*****');
    disp('Number of free games:');
    sum(FreeIndex)
    disp('Number of non free games:');
    sum(NonFreeIndex)

    disp('Average price + std dev of non free games:');
    [m,sig] = average(Price);

    disp('Average userscore + std dev:');
    [m,sig] = average(UserScore);

    disp('Average metascore + std dev:');
    [m,sig] = average(MetaScore);

    disp('Average days on market + std dev:');
    [m,sig] = average(Days);

    disp('Average number of owners + std dev:');
    [m,sig] = average(Owners);

    disp('Average playtime + std dev:');
    [m,sig] = average(Playtime);

    disp('Average median playtime + std dev:');
    [m,sig] = average(MedianPlaytime);

    disp('Average genre count + std dev:');
    [m,sig] = average(GenreCount);

    disp('Average sales per day + std dev:');
    [m,sig] = average(SalesPerDay);

    disp('Average character count + std dev:');
    [m,sig] = average(CharCount);

    disp('Average special character count + std dev:');
    [m,sig] = average(SpecCharCount);

    disp('Average release month + std dev:');
    [m,sig] = average(ReleaseMonth);

    disp('Average release year + std dev:');
    [m,sig] = average(ReleaseYear);
end

%%%%mean and standard deviation for games
function [m,sig] = average(M)
    Z = M(M>0);
    m = mean(Z)
    sig = std(Z)
    fprintf('total count: %d\n', max(size(Z)));
end

%%%%plot games by month
function plotByMonth()
    month = csvread('month.csv',1,0);
```

```
    figure;
    hold on;
    z=1;
    for i = 1:max(size(month))
        plot3(z:(z+10),1:11, month(i,2:12), '-');
        z = z + 11;
    end
    xlabel('Month');
    ylabel('Genre');
    zlabel('Games');
    hold off;
end

%%%%loop for curve fitting
function fitMonthTotal()
    monthData = csvread('monthlytotal.csv',0,0);
    for i = 4:14
        fitPolynomialDegree(monthData, i);
    end
end

%%%%curve fitting and plot
function fitPolynomialDegree(monthData, deg)
    figure;
    hold on;

    for i = 1:11
        plot(monthData(12*i-11:12*i,1), monthData(12*i-11:12*i,2), 'x');
    end
    axis([0 12 0 200]);
    xlabel('Month');
    ylabel('Games');
    x = monthData(:,1);
    y = monthData(:,2);
    n = max( size(monthData) );
    for i=1:(deg+1)
        A(:,i) =  x.^(deg-i+1);
    end
    c = pinv(A) * y;
    disp(deg);
    c
    xCurve = linspace( 0.5, 11.5, 30 )';
    for i=1:(deg+1)
    ACurve(:,i) = xCurve.^(deg-i+1);
    end
    yCurve = ACurve * c;
    plot( xCurve, yCurve, 'm-' );
    title( sprintf('Degree %d polynomial', deg ),...
        'Color', 'b', 'FontSize', 12 );

    yNew = A * c;
    for I = 1:n
        plot( [x(I) x(I)], [yNew(I) y(I)], '--', 'Color', [0.6,0.6,0.6] );
    end;
    plot( x, yNew, 'b.', 'MarkerSize', 10 );
    yNew(1:12:n)
    error = norm( yNew - y );
    text( 2, 180, sprintf( 'Error = %.3f', error ), 'Color', 'k', 'FontSize', 12 );

    hold off;
end
```