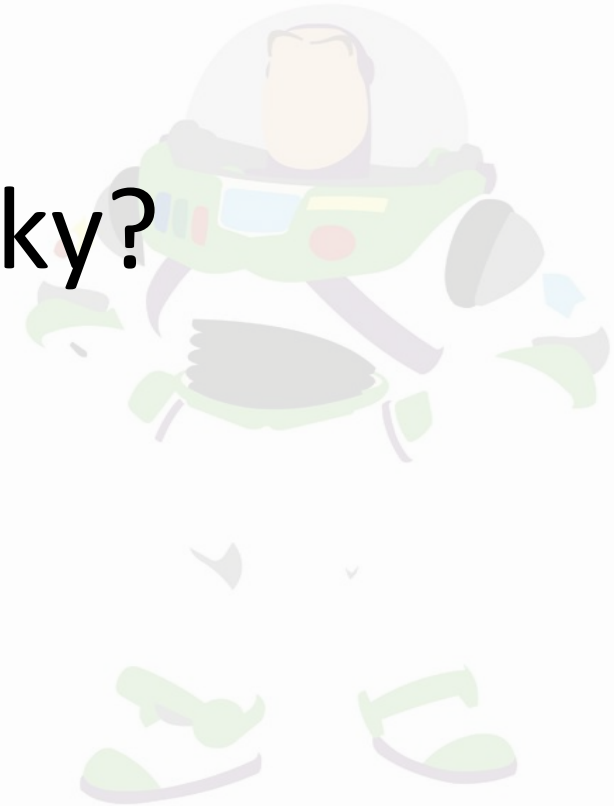
Toy Story 4 : Risky?

Gabriel Teixeira Fonteles



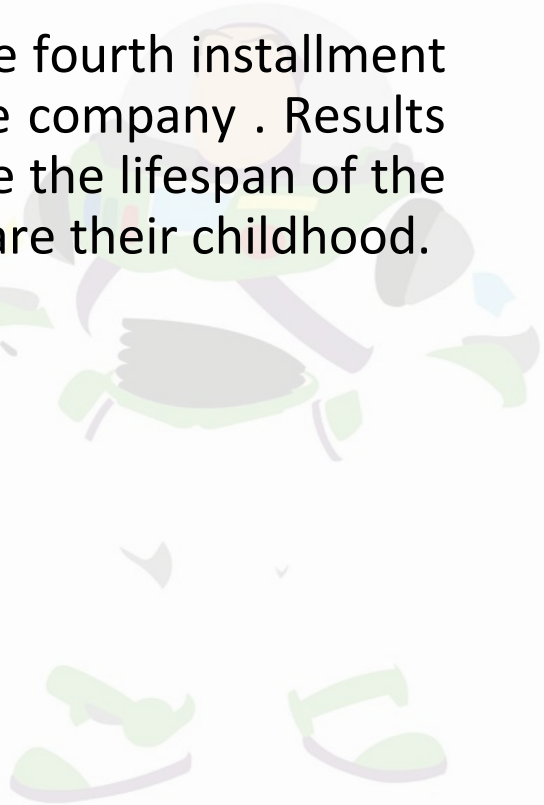
Dataset(s)

The dataset chosen to perform the study:

- IMDB Movie Dataset

Motivation

This work intends to analyze the decision of making the fourth installment of the series, whether or not it’s a secure move to the company . Results are important to decision makers at Disney, to evaluate the lifespan of the franchise, and to fans alike who want to remember/share their childhood.



Research Question(s)

The following study will answer the questions:

- Is Toy Story 4 a risky production?
- Is Toy Story still a strong franchise?

Findings

Initially it was searched on the database the string Toy Story to verify if the films were present in the database and retrieve its movie id's

	movieId	title	genres
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
3027	3114	Toy Story 2 (1999)	Adventure|Animation|Children|Comedy|Fantasy
15401	78499	Toy Story 3 (2010)	Adventure|Animation|Children|Comedy|Fantasy|IMAX
21981	106022	Toy Story of Terror (2013)	Animation|Children|Comedy
24458	115875	Toy Story Toons: Hawaiian Vacation (2011)	Adventure|Animation|Children|Comedy|Fantasy
24460	115879	Toy Story Toons: Small Fry (2011)	Adventure|Animation|Children|Comedy|Fantasy
25461	120468	Toy Story Toons: Partysaurus Rex (2012)	Animation|Children|Comedy
25463	120474	Toy Story That Time Forgot (2014)	Animation|Children

Findings

After getting each film's movie id it was possible to access it's ratings over time. There are a total 8 movies on the database but only the movies below had at least 3 years of ratings:

- Toy Story
- Toy Story 2
- Toy Story 3
- Toy Story of Terror

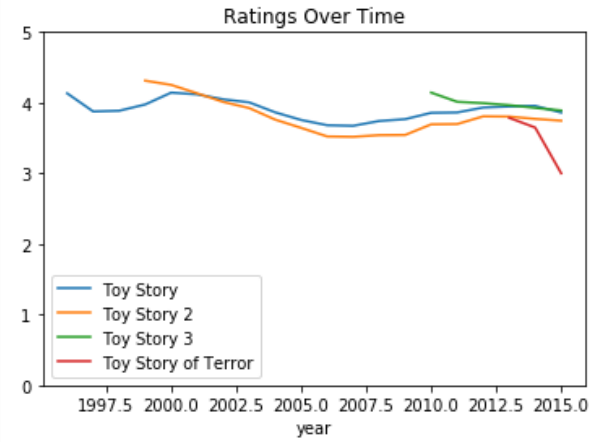
And because of lacking data from the other movies the analysis was focused on these movies.

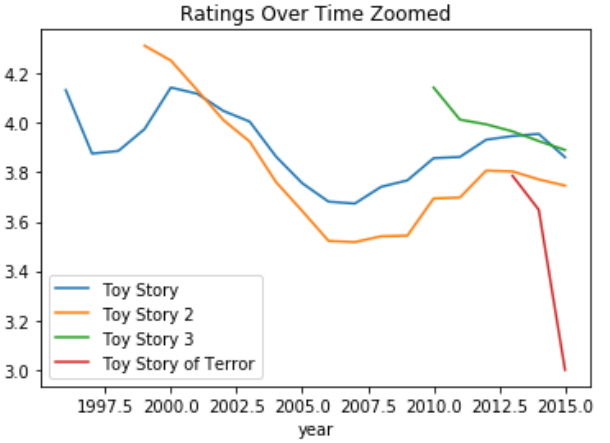
Findings

The Ratings Over Time were plotted two times so the discrepancy between movies were more evident.

From these graphs alone one can conclude that Toy Story franchise maintained a high popularity over time despite the expected fall on ratings from spin off movies.

Assuring it will have a long lifespan since movies rating doesn't seem to be falling.





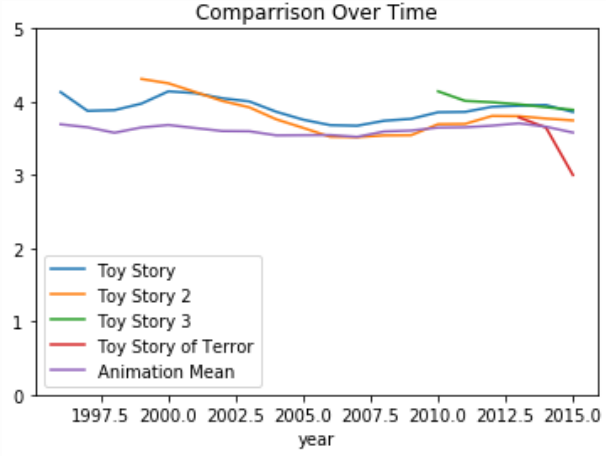
Findings

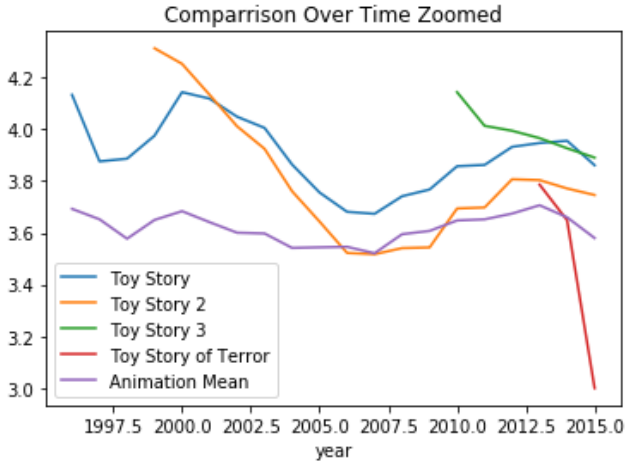
After a initial impression that the franchise is good and a fourth installment of the series would be secure now the ratings will be compared to the whole Animation industry mean over time to confirm that the franchise is worthy of the next movie.

Findings

Again it was plotted two times so the discrepancies are more visible and to maintain the honesty of the visualization.

The author considers the fourth installment a secure investment as, even on the worst performing titles, the ratings were above or near the industry mean most of the time.





Findings

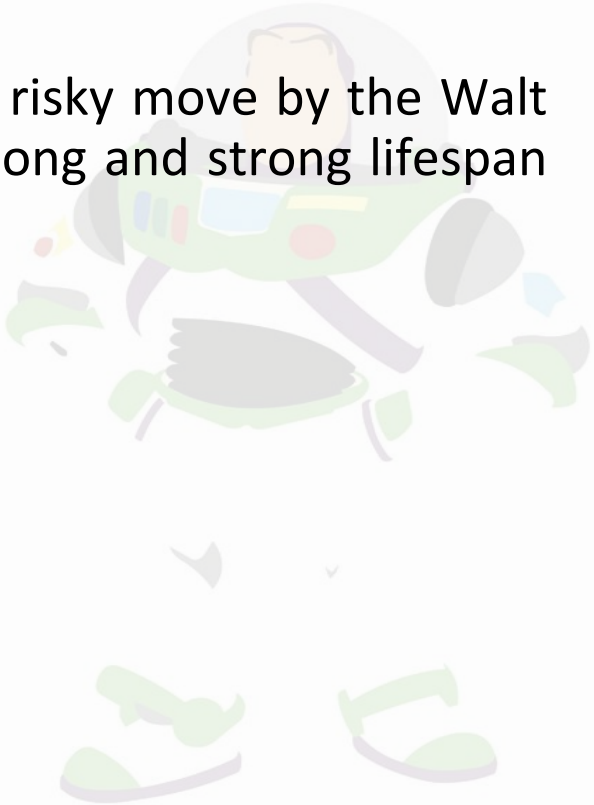
To further prove the consistency of the series the three main movies were compared to the best performing movies in the Animation category resulting:

- Toy Story is at 6.2% of Top Rated Animation Movies
- Toy Story 2 is at 9.65% of Top Rated Animation Movies
- Toy Story 3 is at 3.25% of Top Rated Animation Movies

Therefore the main series is considered consistent and represent a relatively low risk to a sequel

Findings

This work concludes that Toy Story 4 was not a risky move by the Walt Disney company and that the franchise still have a long and strong lifespan since the ratings seem stable through the years.



Acknowledgements

All the work was done inside the Jupyter Notebook using primarily the Pandas library. I have presented this work to family and some friends who are Toy Story fans and adjusted some graphs and the code to ensure a better readability based on their feedback.

References

I did all the work on my own based on a brief previous experience with data on college.

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [2]:
```
movies = pd.read_csv('./movielens/movies.csv', sep=',')
ratings = pd.read_csv('./movielens/ratings.csv', sep=',')
tags = pd.read_csv('./movielens/tags.csv', sep=',')
```

In [3]:
```
movies.head(5)
```

Out[3]:

	movieId	title	genres
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
1	2	Jumanji (1995)	Adventure|Children|Fantasy
2	3	Grumpier Old Men (1995)	Comedy|Romance
3	4	Waiting to Exhale (1995)	Comedy|Drama|Romance
4	5	Father of the Bride Part II (1995)	Comedy

In [4]:
```
tags.head(5)
```

Out[4]:

	userId	movieId	tag	timestamp
0	18	4141	Mark Waters	1240597180
1	65	208	dark hero	1368150078
2	65	353	dark hero	1368150079
3	65	521	noir thriller	1368149983
4	65	592	dark hero	1368150078

In [5]:
```
ratings.head(5)
```

Out[5]:

	userId	movieId	rating	timestamp
0	1	2	3.5	1112486027
1	1	29	3.5	1112484676
2	1	32	3.5	1112484819
3	1	47	3.5	1112484727
4	1	50	3.5	1112484580

In [6]:
```
ratings['parsed_time'] = pd.to_datetime(ratings['timestamp'], unit='s')
ratings['year']=ratings['parsed_time'].dt.year
del ratings['parsed_time']
del ratings['timestamp']

ratings.head(5)
```

Out[6]:

	userId	movieId	rating	year
0	1	2	3.5	2005
1	1	29	3.5	2005
2	1	32	3.5	2005
3	1	47	3.5	2005
4	1	50	3.5	2005

Checking Toy Story

next line will check if all Toy Stories are present in the data

In [7]:
```
movies[movies['title'].str.contains('Toy Story')]
```

Out[7]:

	movieId	title	genres
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
3027	3114	Toy Story 2 (1999)	Adventure|Animation|Children|Comedy|Fantasy
15401	78499	Toy Story 3 (2010)	Adventure|Animation|Children|Comedy|Fantasy|IMAX
21981	106022	Toy Story of Terror (2013)	Animation|Children|Comedy
24458	115875	Toy Story Toons: Hawaiian Vacation (2011)	Adventure|Animation|Children|Comedy|Fantasy
24460	115879	Toy Story Toons: Small Fry (2011)	Adventure|Animation|Children|Comedy|Fantasy
25461	120468	Toy Story Toons: Partysaurus Rex (2012)	Animation|Children|Comedy
25463	120474	Toy Story That Time Forgot (2014)	Animation|Children

Toy Stories movieId's

- Toy Story -- 1
- Toy Story 2 -- 3114
- Toy Story 3 -- 78499
- Toy Story of Terror -- 106022
- Toy Story Toons: Hawaiian Vacation -- 115875
- Toy Story Toons: Small Fry -- 115879
- Toy Story Toons: Partysaurus Rex -- 120468
- Toy Story That Time Forgot -- 120474

In [8]:
```
TS1_ratings=(ratings[ratings['movieId']==1]).groupby('year', as_index=False).mean()
TS1_ratings['Toy Story']=TS1_ratings['rating']
del TS1_ratings['userId']
del TS1_ratings['rating']
TS1_ratings
```

Out[8]:

	year	movieId	Toy Story
0	1996	1.0	4.132270
1	1997	1.0	3.875424
2	1998	1.0	3.885799
3	1999	1.0	3.974688
4	2000	1.0	4.142609
5	2001	1.0	4.117698
6	2002	1.0	4.047855
7	2003	1.0	4.004680
8	2004	1.0	3.863380
9	2005	1.0	3.755229
10	2006	1.0	3.680774
11	2007	1.0	3.672973
12	2008	1.0	3.740781
13	2009	1.0	3.766777
14	2010	1.0	3.856772
15	2011	1.0	3.861689
16	2012	1.0	3.931421
17	2013	1.0	3.946274
18	2014	1.0	3.954945
19	2015	1.0	3.860412

In [9]:
```
TS2_ratings=(ratings[ratings['movieId']==3114]).groupby('year', as_index=False).
mean()
TS2_ratings['Toy Story 2']=TS2_ratings['rating']
del TS2_ratings['userId']
del TS2_ratings['rating']
TS2_ratings
```

Out[9]:

	year	movieId	Toy Story 2
0	1999	3114.0	4.311828
1	2000	3114.0	4.251947
2	2001	3114.0	4.132420
3	2002	3114.0	4.010844
4	2003	3114.0	3.923929
5	2004	3114.0	3.760249
6	2005	3114.0	3.643074
7	2006	3114.0	3.521959
8	2007	3114.0	3.517027
9	2008	3114.0	3.540570
10	2009	3114.0	3.543142
11	2010	3114.0	3.693533
12	2011	3114.0	3.697494
13	2012	3114.0	3.806373
14	2013	3114.0	3.803161
15	2014	3114.0	3.770793
16	2015	3114.0	3.745575

In [10]:
```
TS3_ratings=(ratings[ratings['movieId']==78499]).groupby('year', as_index=False)
.mean()
TS3_ratings['Toy Story 3']=TS3_ratings['rating']
del TS3_ratings['userId']
del TS3_ratings['rating']
TS3_ratings
```

Out[10]:

	year	movieId	Toy Story 3
0	2010	78499.0	4.142544
1	2011	78499.0	4.012645
2	2012	78499.0	3.993615
3	2013	78499.0	3.965144
4	2014	78499.0	3.925974
5	2015	78499.0	3.889976

In [11]:
```
TSterror_ratings=(ratings[ratings['movieId']==106022]).groupby('year', as_index=
False).mean()
TSterror_ratings['Toy Story of Terror']=TSterror_ratings['rating']
del TSterror_ratings['userId']
del TSterror_ratings['rating']
TSterror_ratings
```

Out[11]:

	year	movieId	Toy Story of Terror
0	2013	106022.0	3.785714
1	2014	106022.0	3.648148
2	2015	106022.0	3.000000

In [12]:
```
TShawaii_ratings=(ratings[ratings['movieId']==115875]).groupby('year', as_index=
False).mean()
del TShawaii_ratings['userId']

TShawaii_ratings
```

Out[12]:

	year	movieId	rating
0	2014	115875.0	3.583333
1	2015	115875.0	3.000000

In [13]:
```
TSfry_ratings=(ratings[ratings['movieId']==115879]).groupby('year', as_index=Fal
se).mean()
del TSfry_ratings['userId']

TSfry_ratings
```

Out[13]:

	year	movieId	rating
0	2014	115879.0	3.625000
1	2015	115879.0	2.666667

In [14]:
```
TSrex_ratings=(ratings[ratings['movieId']==120468]).groupby('year', as_index=Fal
se).mean()
del TSrex_ratings['userId']
TSrex_ratings
```

Out[14]:

	year	movieId	rating
0	2015	120468	2.625

In [15]:
```
TStime_ratings=(ratings[ratings['movieId']==120474]).groupby('year', as_index=Fa
lse).mean()
del TStime_ratings['userId']

TStime_ratings
```

Out[15]:

	year	movieId	rating
0	2015	120474.0	3.138889

In [16]:
```
ts_1a2= TS1_ratings.merge(TS2_ratings, on='year', how='outer')

ts_1a2a3= ts_1a2.merge(TS3_ratings, on='year', how='outer')

ts_all= ts_1a2a3.merge(TSterror_ratings, on='year', how='outer')
```

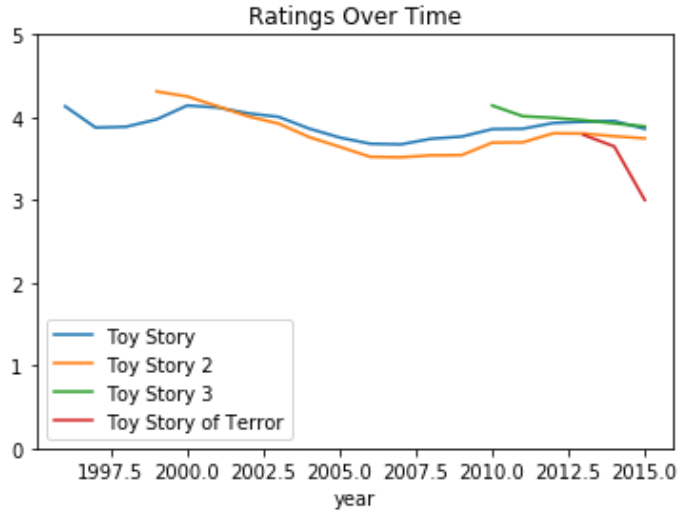
In [17]:
```
del ts_all['movieId_x']
del ts_all['movieId_y']
ts_all.head(5)
```

Out[17]:

	year	Toy Story	Toy Story 2	Toy Story 3	Toy Story of Terror
0	1996	4.132270	NaN	NaN	NaN
1	1997	3.875424	NaN	NaN	NaN
2	1998	3.885799	NaN	NaN	NaN
3	1999	3.974688	4.311828	NaN	NaN
4	2000	4.142609	4.251947	NaN	NaN

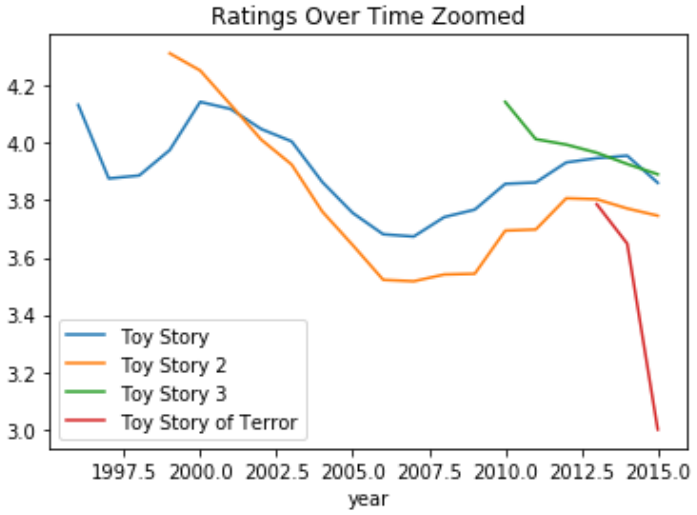
In [18]:
```
ts_all2=ts_all.groupby('year', as_index= True).mean()

ts_all2.plot(ylim=[0,5], title='Ratings Over Time')
```

Out[18]: <matplotlib.axes._subplots.AxesSubplot at 0x2150d33b7f0>



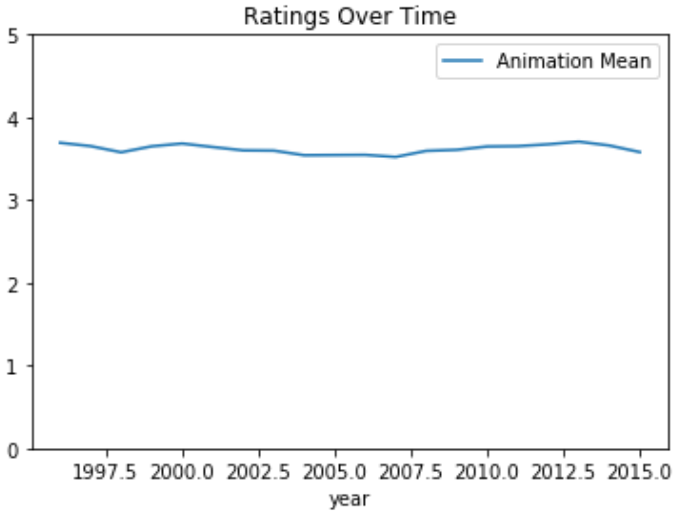
In [19]:
```
ts_all2.plot(title='Ratings Over Time Zoomed')
```

Out[19]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x2150d488780>
```



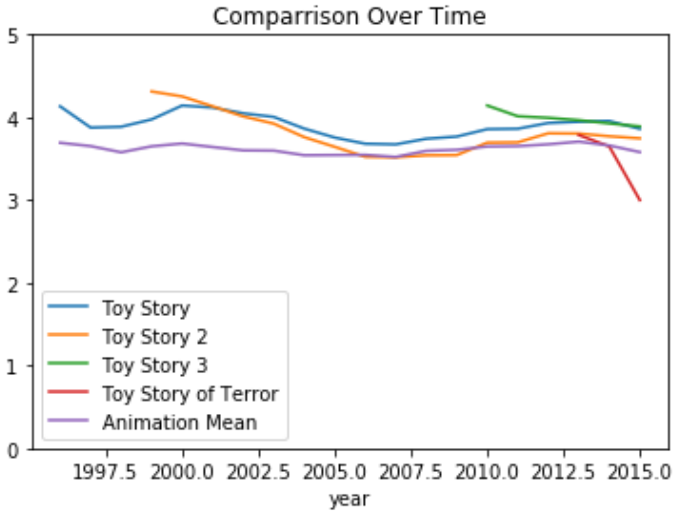
In [20]:
```
is_Animation=movies['genres'].str.contains('Animation')
Ani = movies[is_Animation]

Anima = Ani.merge(ratings, on ='movieId', how ="inner")

Animation= Anima.groupby('year',as_index=True).mean()

Animation["Animation Mean"]= Animation['rating']

del Animation ['movieId']
del Animation ['userId']
del Animation ['rating']

Animation.plot(ylim=[0,5],title='Ratings Over Time')
```

Out[20]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x2150d506c18>
```



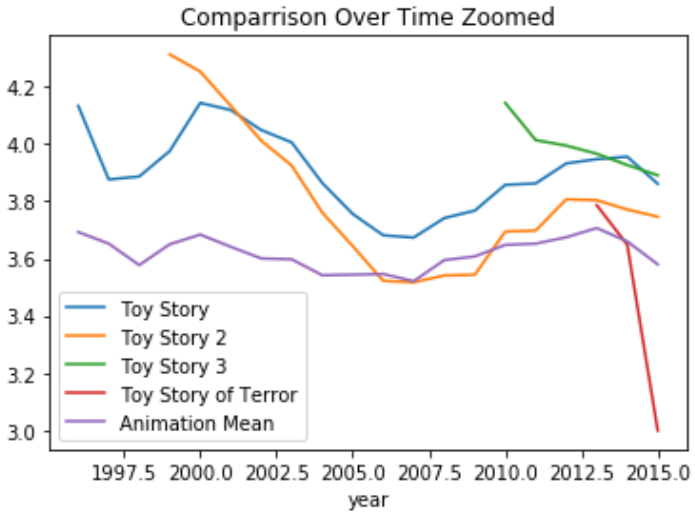
In [21]:
```
Result=ts_all2.merge(Animation, on ='year', how ="inner")
Result.plot(ylim=[0,5], title='Comparrison Over Time')
```

Out[21]: <matplotlib.axes._subplots.AxesSubplot at 0x2150d5b2b70>



In [22]:
```
Result.plot(title='Comparrison Over Time Zoomed')
```

Out[22]: <matplotlib.axes._subplots.AxesSubplot at 0x2150d5c9b00>



In [23]:
```
AnimFilms=Anima.groupby('movieId', as_index=False).mean()
AnimFilmsSortd=AnimFilms.sort_values(by='rating',ascending=False)

AFSR=AnimFilmsSortd.reset_index()
```

In [24]:
```
n={'rank':pd.Series(range(1,1016),index=range(1015))}

nmbr=pd.DataFrame(n)
nmbr.shape
```

Out[24]: (1015, 1)

In [25]:
```
rank = pd.concat([AFSR, nmbr], axis=1, sort=False)
```

In [26]:
```
print(rank.loc[rank['movieId'] == 1])
```

```
    index  movieId        userId   rating         year  rank
62      0        1  69282.396821  3.92124  2002.860549    63
```

In [27]:
```
print(rank.loc[rank['movieId'] == 3114])
```

```
    index  movieId        userId    rating         year  rank
97     88     3114  69059.550856  3.841853  2005.078788    98
```

In [28]:
```
print(rank.loc[rank['movieId'] == 78499])
```

```
    index  movieId        userId    rating         year  rank
32    454    78499  69130.541602  4.012974  2011.909877    33
```

In [29]:
```
print('ToyStory is '+str(round(((63/1016)*100),2))+'% of Top Scores')
```

```
ToyStory is 6.2% of Top Scores
```

In [30]:
```
print('ToyStory2 is '+str(round(((98/1016)*100),2))+'% of Top Scores')
```

```
ToyStory2 is 9.65% of Top Scores
```

In [31]:
```
print('ToyStory3 is '+str(round(((33/1016)*100),2))+'% of Top Scores')
```

```
ToyStory3 is 3.25% of Top Scores
```