In [92]:

```
import pandas as pd
import matplotlib.pyplot as plt
```

Movie Data Analysis

In [93]:

```
movies = pd.read_csv('./movielens/movies.csv', sep=',')
print(type(movies))
movies.head(10)
```

```
<class 'pandas.core.frame.DataFrame'>
```

Out[93]:

	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
5	6	Heat (1995)	Action|Crime|Thriller
6	7	Sabrina (1995)	Comedy|Romance
7	8	Tom and Huck (1995)	Adventure|Children
8	9	Sudden Death (1995)	Action
9	10	GoldenEye (1995)	Action|Adventure|Thriller

In [94]:

```
tags = pd.read_csv('./movielens/tags.csv', sep=',')
tags.head()
```

Out[94]:

	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 [95]:

```
ratings = pd.read_csv('./movielens/ratings.csv', sep=',',parse_dates=['timestamp']
ratings.head(5)
```

Out[95]:

	userId	movieId	rating	timestamp
0	1	31	2.5	1260759144
1	1	1029	3.0	1260759179
2	1	1061	3.0	1260759182
3	1	1129	2.0	1260759185
4	1	1172	4.0	1260759205

In [96]:

```
# Extract row 0, 11, 1500 from DataFrame

tags.iloc[ [0,11,1500] ]
```

Out[96]:

	userId	movieId	tag	timestamp
0	18	4141	Mark Waters	1240597180
11	65	1783	noir thriller	1368149983
1500	619	1197	seen more than once	1184188996

In [72]:

```
tag_counts = tags['tag'].value_counts()
tag_counts[-10:]
```

Out[72]:

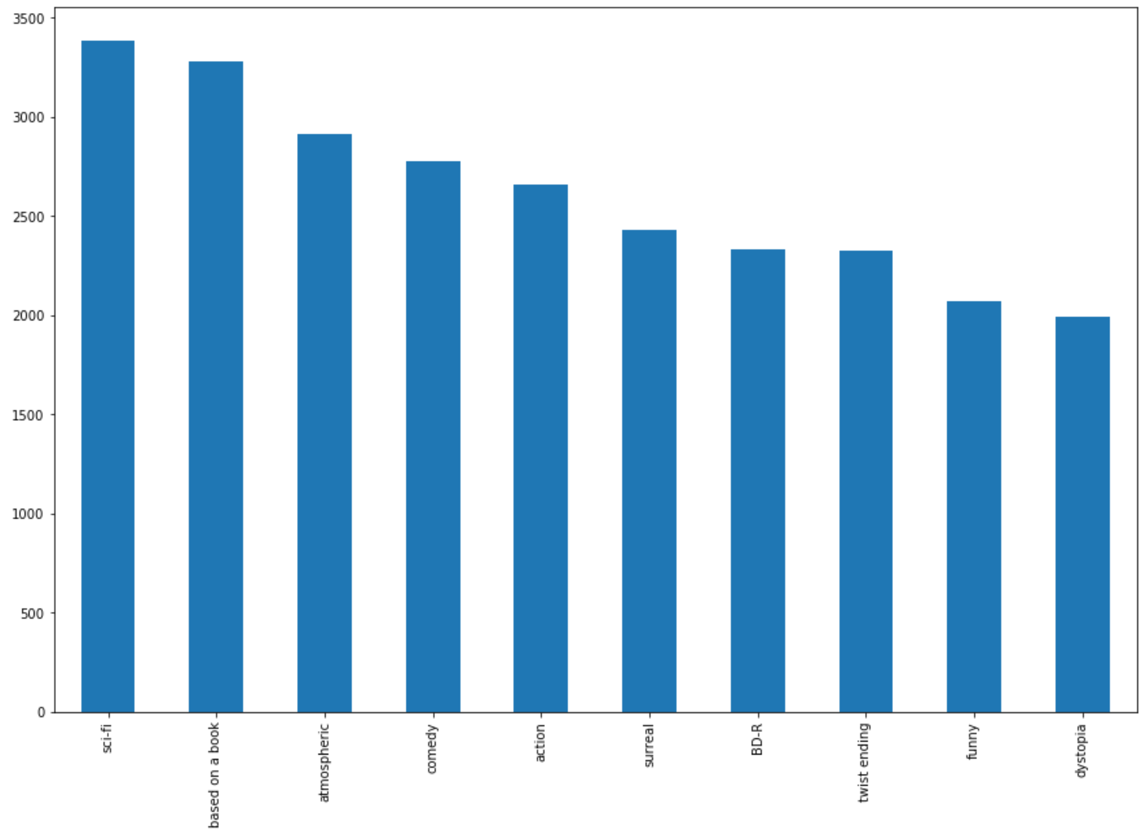
```
Lisa Eilbacher                  1
Tommy Morrison                  1
hollywood witchhunt             1
unique idea                     1
Emily BrontÃ«                   1
very straight forward story     1
Maria Schrader                  1
Teary Eyed                      1
One of Hitcocks best            1
waking up                       1
Name: tag, dtype: int64
```

In [73]:

```
%matplotlib inline
tag_counts[:10].plot(kind='bar', figsize=(15,10))
```

Out[73]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x8ec91d0>
```



Group by and aggregate

In [97]:

```
tags = tags.dropna()
```

In [98]:

```
ratings_count = ratings[['movieId','rating']].groupby('rating').count()
ratings_count
```

Out[98]:

	movieId
rating	
0.5	1101
1.0	3326
1.5	1687
2.0	7271
2.5	4449
3.0	20064
3.5	10538
4.0	28750
4.5	7723
5.0	15095

In [99]:

```
average_rating = ratings[['movieId','rating']].groupby('movieId').mean()
average_rating.head()
```

Out[99]:

	rating
movieId	
1	3.872470
2	3.401869
3	3.161017
4	2.384615
5	3.267857

In [100]:

```
movie_count = ratings[['movieId','rating']].groupby('movieId').count()
movie_count.head()
```

Out[100]:

	rating
movieId	
1	247
2	107
3	59
4	13
5	56

Merge DataFrames

In [101]:

```
t = movies.merge(ratings, on='movieId', how='inner')
t.head()
```

Out[101]:

	movieId	title	genres	userId	rating	timesta
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	7	3.0	8518667
1	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	9	4.0	9386291
2	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	13	5.0	13313800
3	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	15	2.0	9979383
4	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	19	3.0	8551900

In [102]:

```
avg_ratings = ratings.groupby('movieId', as_index=False).mean()
del avg_ratings['userId']
avg_ratings
```

Out[102]:

	movieId	rating
0	1	3.872470
1	2	3.401869
2	3	3.161017
3	4	2.384615
4	5	3.267857
5	6	3.884615
6	7	3.283019
7	8	3.800000
8	9	3.150000
9	10	3.450820
10	11	3.689024
11	12	2.861111
12	13	3.937500
13	14	3.451613
14	15	2.318182
15	16	3.948864
16	17	3.924419
17	18	3.288462
18	19	2.597826
19	20	2.538462
20	21	3.536842
21	22	3.355263
22	23	3.090909
23	24	3.044118
24	25	3.742574
25	26	4.100000
26	27	3.142857
27	28	4.083333
28	29	4.025000
29	30	4.050000

```
 ...     ...       ...
9036  158238  3.750000
9037  158314  4.500000
9038  158528  3.500000
9039  158956  4.000000
9040  159093  2.000000
9041  159462  3.000000
9042  159690  2.000000
9043  159755  1.000000
9044  159858  3.750000
9045  159972  0.500000
9046  160080  1.000000
9047  160271  2.500000
9048  160438  4.250000
9049  160440  1.500000
9050  160563  2.500000
9051  160565  2.000000
9052  160567  4.000000
9053  160590  5.000000
9054  160656  3.500000
9055  160718  4.000000
9056  161084  2.500000
9057  161155  0.500000
9058  161594  3.000000
9059  161830  1.000000
9060  161918  1.500000
9061  161944  5.000000
9062  162376  4.500000
9063  162542  5.000000
9064  162672  3.000000
9065  163949  5.000000

9066 rows × 2 columns
```

In [103]:

```
box_office = movies.merge(avg_ratings, on='movieId', how='inner')
box_office.head()
```

Out[103]:

	movieId	title	genres	rating
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	3.872470
1	2	Jumanji (1995)	Adventure|Children|Fantasy	3.401869
2	3	Grumpier Old Men (1995)	Comedy|Romance	3.161017
3	4	Waiting to Exhale (1995)	Comedy|Drama|Romance	2.384615
4	5	Father of the Bride Part II (1995)	Comedy	3.267857

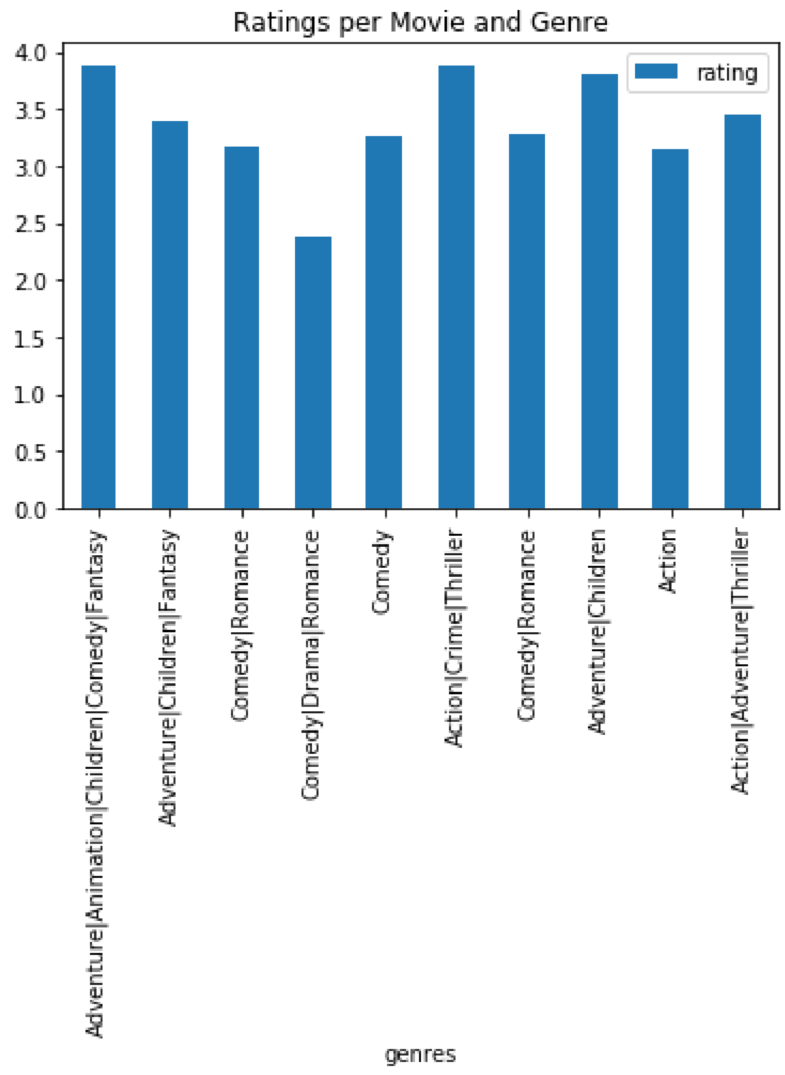
In [ ]:

```
plt.plot(box_office['rating'])
plt.legend(box_office.columns)
dcsummary = pd.DataFrame([box_office.mean(), box_office.sum()],index=['Mean','Tota

plt.table(cellText=dcsummary.values,colWidths = [0.25]*len(box_office.columns),
          rowLabels=dcsummary.index,
          colLabels=dcsummary.columns,
          cellLoc = 'center', rowLoc = 'center',
          loc='top')
fig = plt.gcf()

plt.show()
```

```
In [149]:
```

```
%matplotlib inline
box_office[:10].plot.bar(x='genres', y='rating',title='Ratings per Movie and Genre
plt.box_office.describe()
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent ca
ll last)
<ipython-input-149-117482fde71c> in <module>()
      1 get_ipython().magic('matplotlib inline')
      2 box_office[:10].plot.bar(x='genres', y='rating',title='Rati
ngs per Movie and Genre')
----> 3 plt.box_office.describe()

AttributeError: module 'matplotlib.pyplot' has no attribute 'box_of
fice'
```



is_sci_fi = box_office['genres'].str.contains('Sci-Fi') box_office[is_sci_fi][:5]
box_office[is_sci_fi].describe()

In [105]:

```
boxdesc=box_office[is_sci_fi].sort_values(by='rating', ascending=False)
boxdesc.describe()
```

Out[105]:

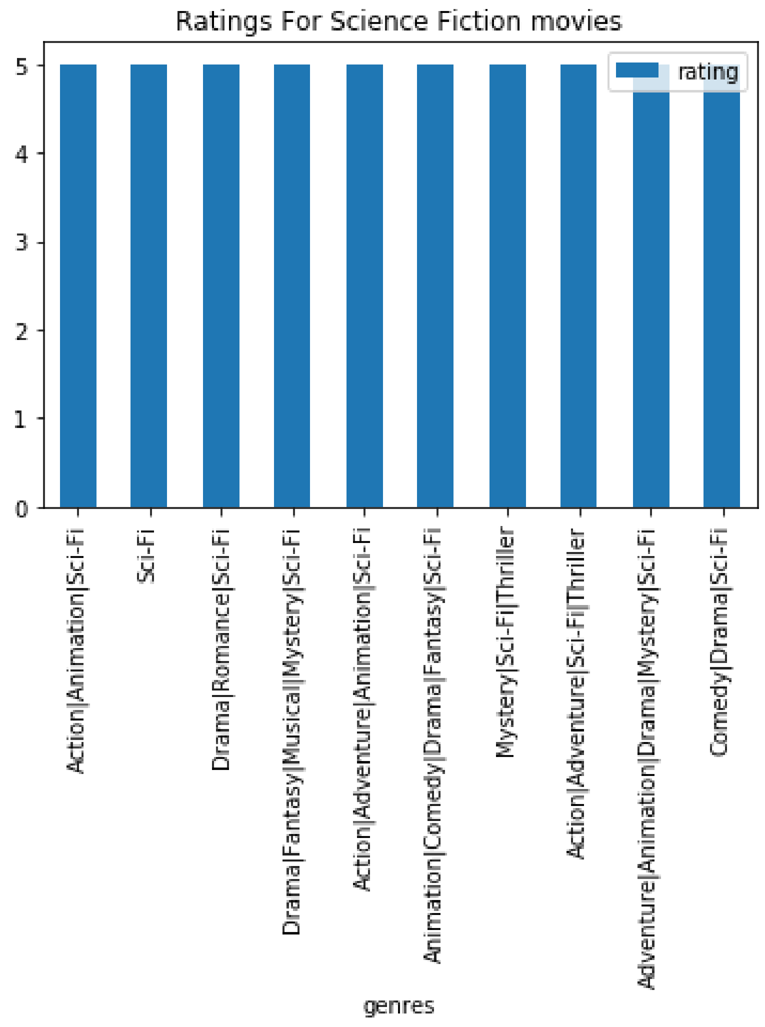
	movieId	rating
count	791.000000	791.000000
mean	36762.238938	3.167052
std	43269.830748	0.813392
min	24.000000	0.500000
25%	2875.500000	2.698214
50%	7003.000000	3.250000
75%	70828.000000	3.750000
max	161918.000000	5.000000

In [106]:

```
%matplotlib inline
boxdesc[:10].plot.bar(x='genres', y='rating', title='Ratings For Science Fiction m
```

Out[106]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x11e5bf90>
```



Descriptive Statistics

In [107]:

```
ratings['rating'].describe()
```

Out[107]:

```
count    100004.000000
mean          3.543608
std           1.058064
min           0.500000
25%           3.000000
50%           4.000000
75%           4.000000
max           5.000000
Name: rating, dtype: float64
```

In [109]:

```
ratings['rating'].mean()
```

Out[109]:

```
3.543608255669773
```

In [14]:

```
ratings.mean()
```

Out[14]:

```
userId       3.470113e+02
movieId      1.254866e+04
rating       3.543608e+00
timestamp             inf
dtype: float64
```

In [110]:

```
ratings['rating'].min()
```

Out[110]:

```
0.5
```

In [111]:

```
ratings['rating'].max()
```

Out[111]:

```
5.0
```

In [112]:

```
ratings['rating'].std()
```

Out[112]:

```
1.0580641091073735
```

In [113]:

```
ratings['rating'].mode()
```

Out[113]:

```
0    4.0
dtype: float64
```

In [114]:

```
ratings.corr()
```

Out[114]:

	userId	movieId	rating
userId	1.000000	0.007126	0.010467
movieId	0.007126	1.000000	-0.028894
rating	0.010467	-0.028894	1.000000

Data Cleaning: Handling Missing Data

In [115]:

```
movies.shape
```

Out[115]:

```
(9125, 3)
```

In [116]:

```
#is any row NULL ?

movies.isnull().any()
```

Out[116]:

```
movieId    False
title      False
genres     False
dtype: bool
```

No NULL values

In [117]:

```
ratings.shape
```

Out[117]:

```
(100004, 4)
```

In [118]:

```
#is any row NULL ?

ratings.isnull().any()
```

Out[118]:

```
userId       False
movieId      False
rating       False
timestamp    False
dtype: bool
```

No NULL values

In [119]:

```
tags.shape
```

Out[119]:

```
(465548, 4)
```

In [120]:

```
#is any row NULL ?

tags.isnull().any()
```

Out[120]:

```
userId       False
movieId      False
tag          False
timestamp    False
dtype: bool
```

In [121]:

```
tags = tags.dropna()
```

In [122]:

```
#Check again: is any row NULL ?

tags.isnull().any()
```

Out[122]:

```
userId       False
movieId      False
tag          False
timestamp    False
dtype: bool
```

In [123]:

```
tags.shape
```

Out[123]:

```
(465548, 4)
```

No NULL values ! Number of lines have reduced.

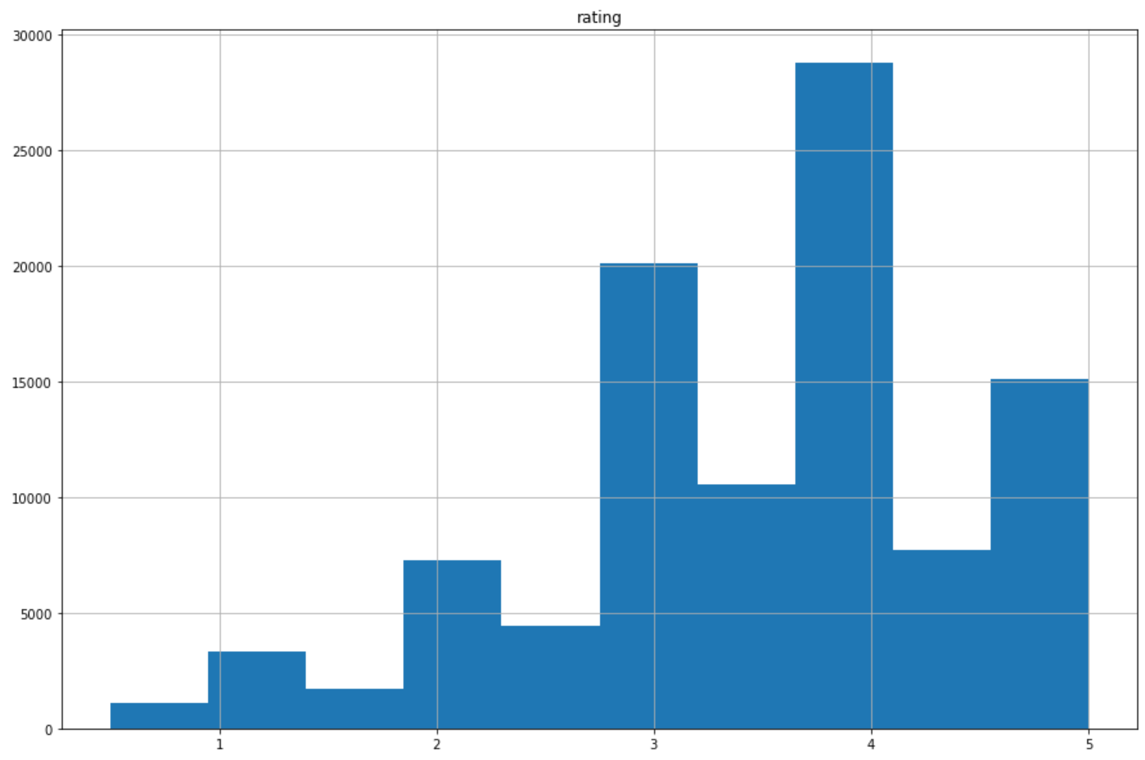
Data Visualisation

In [62]:

```
%matplotlib inline

ratings.hist(column='rating', figsize=(15,10))
```

Out[62]:

```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x0B954F90
>]], dtype=object)
```

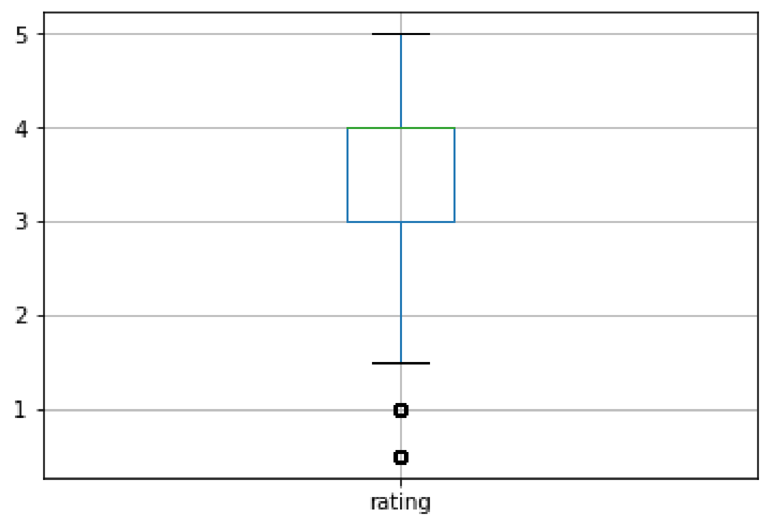


In [125]:

```
ratings.boxplot(column='rating',figsize=(15,20))
```

Out[125]:

```
<matplotlib.axes._subplots.AxesSubplot at 0xea267d0>
```



In [124]:

```
tags['tag'].head()
```

Out[124]:

```
0      Mark Waters
1        dark hero
2        dark hero
3    noir thriller
4        dark hero
Name: tag, dtype: object
```

In [125]:

```
movies[['title','genres']].head()
```

Out[125]:

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

In [126]:

```
ratings[-5:]
```

Out[126]:

	userId	movieId	rating	timestamp
99999	671	6268	2.5	1065579370
100000	671	6269	4.0	1065149201
100001	671	6365	4.0	1070940363
100002	671	6385	2.5	1070979663
100003	671	6565	3.5	1074784724

In [127]:

```
tag_counts = tags['tag'].value_counts()
tag_counts[-10:]
```

Out[127]:

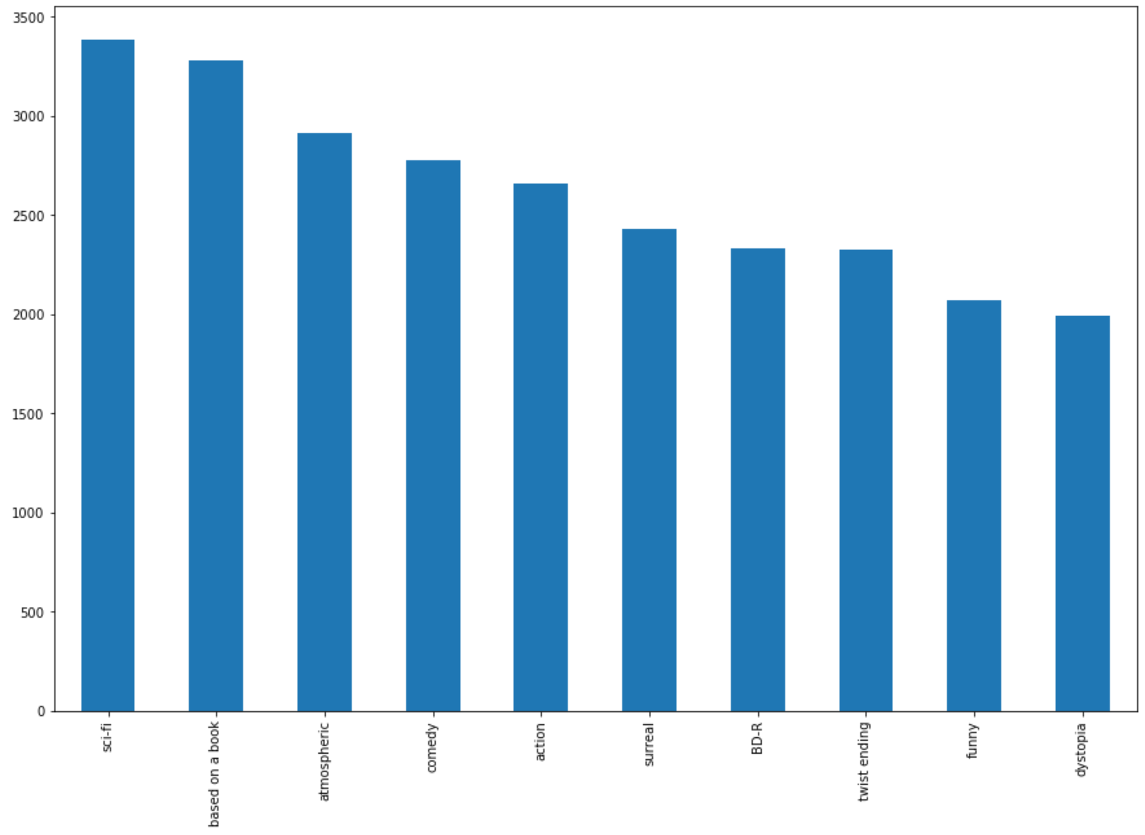
```
moms                               1
satisfying                         1
Wrong colors                       1
James Faulkner                     1
Oscar nominee: Lead Actress        1
bhangra                            1
Till Schauder                      1
David Manners                      1
Irvin S. Yeaworth Jr.              1
set & costume design               1
Name: tag, dtype: int64
```

In [128]:

```
tag_counts[:10].plot(kind='bar', figsize=(15,10))
```

Out[128]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x94fa410>
```



Filters for selecting rows

In [129]:

```
is_highly_rated = ratings['rating'] >= 4.0

ratings[is_highly_rated][30:50]
```

Out[129]:

	userId	movieId	rating	timestamp
83	2	551	5.0	835355767
85	2	585	5.0	835355817
89	2	589	5.0	835355697
90	2	590	5.0	835355395
91	2	592	5.0	835355395
94	2	661	4.0	835356141
95	2	720	4.0	835355978
97	3	110	4.0	1298922049
100	3	296	4.5	1298862418
101	3	318	5.0	1298862121
103	3	356	5.0	1298862167
111	3	778	4.0	1298863157
113	3	1197	5.0	1298932770
115	3	1235	4.0	1298861628
117	3	1378	4.0	1298861658
119	3	1721	4.5	1298923236
120	3	1884	4.0	1298863143
121	3	2028	4.0	1298921862
122	3	2318	4.0	1298861753
128	3	2841	4.0	1298861733

In [130]:

```
is_sc_fi = movies['genres'].str.contains('Sci-Fi')

movies[is_sc_fi][5:15]
```

Out[130]:

	movieId	title	genres
95	103	Unforgettable (1996)	Mystery|Sci-Fi|Thriller
139	160	Congo (1995)	Action|Adventure|Mystery|Sci-Fi
151	172	Johnny Mnemonic (1995)	Action|Sci-Fi|Thriller
152	173	Judge Dredd (1995)	Action|Crime|Sci-Fi
173	196	Species (1995)	Horror|Sci-Fi
174	198	Strange Days (1995)	Action|Crime|Drama|Mystery|Sci-Fi|Thriller
184	208	Waterworld (1995)	Action|Adventure|Sci-Fi
228	256	Junior (1994)	Comedy|Sci-Fi
232	260	Star Wars: Episode IV - A New Hope (1977)	Action|Adventure|Sci-Fi
245	273	Mary Shelley's Frankenstein (Frankenstein) (1994)	Drama|Horror|Sci-Fi

In [131]:

```
movies[is_sc_fi].head(10)
```

Out[131]:

	movieId	title	genres
23	24	Powder (1995)	Drama|Sci-Fi
28	29	City of Lost Children, The (Cité des enfants p...	Adventure|Drama|Fantasy|Mystery|Sci-Fi
31	32	Twelve Monkeys (a.k.a. 12 Monkeys) (1995)	Mystery|Sci-Fi|Thriller
62	66	Lawnmower Man 2: Beyond Cyberspace (1996)	Action|Sci-Fi|Thriller
70	76	Screamers (1995)	Action|Sci-Fi|Thriller
95	103	Unforgettable (1996)	Mystery|Sci-Fi|Thriller
139	160	Congo (1995)	Action|Adventure|Mystery|Sci-Fi
151	172	Johnny Mnemonic (1995)	Action|Sci-Fi|Thriller
152	173	Judge Dredd (1995)	Action|Crime|Sci-Fi
173	196	Species (1995)	Horror|Sci-Fi

Group by and Aggregate

In [132]:

```
ratings_count = ratings[['movieId','rating']].groupby('rating').count()
ratings_count
```

Out[132]:

	movieId
rating	
0.5	1101
1.0	3326
1.5	1687
2.0	7271
2.5	4449
3.0	20064
3.5	10538
4.0	28750
4.5	7723
5.0	15095

In [133]:

```
average_rating = ratings[['movieId','rating']].groupby('movieId').mean()
average_rating.head()
```

Out[133]:

	rating
movieId	
1	3.872470
2	3.401869
3	3.161017
4	2.384615
5	3.267857

In [134]:

```
movie_count = ratings[['movieId','rating']].groupby('movieId').count()
movie_count.head()
```

Out[134]:

```
        rating
movieId
      1    247
      2    107
      3     59
      4     13
      5     56
```

In [135]:

```
movie_count = ratings[['movieId','rating']].groupby('movieId').count()
movie_count.tail()
```

Out[135]:

```
        rating
movieId
 161944      1
 162376      1
 162542      1
 162672      1
 163949      1
```

Merge Dataframes

In [136]:

```
tags.head()
```

Out[136]:

	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 [137]:

```
movies.head()
```

Out[137]:

	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 [138]:

```
t = movies.merge(tags, on='movieId', how='inner')
t.head()
```

Out[138]:

	movieId	title	genres	userId	tag	time
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	1644	Watched	14177
1	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	1741	computer animation	11839
2	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	1741	Disney animated feature	11839
3	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	1741	Pixar animation	11839
4	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	1741	TÃ©a Leoni does not star in this movie	12450

Combine aggreagation, merging, and filters to get useful analytics

In [139]:

```
avg_ratings = ratings.groupby('movieId', as_index=False).mean()
del avg_ratings['userId']
avg_ratings.head()
```

Out[139]:

	movieId	rating
0	1	3.872470
1	2	3.401869
2	3	3.161017
3	4	2.384615
4	5	3.267857

Do science fiction movies tend to be rated more highly than other movie genres? You can pull out the ratings by genre and see how they stack up to one another. You could also see if the distributions of ratings within genre are comparable across genres (e.g., maybe science fiction movies tend to be either highly or poorly rated, with little in between).

In [140]:

```
box_office = movies.merge(avg_ratings, on='movieId', how='inner')
box_office.head()
```

Out[140]:

	movieId	title	genres	rating
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	3.872470
1	2	Jumanji (1995)	Adventure|Children|Fantasy	3.401869
2	3	Grumpier Old Men (1995)	Comedy|Romance	3.161017
3	4	Waiting to Exhale (1995)	Comedy|Drama|Romance	2.384615
4	5	Father of the Bride Part II (1995)	Comedy	3.267857

In [141]:

```
is_highly_rated = box_office['rating'] >= 4.0

box_office[is_highly_rated][-5:]
```

Out[141]:

	movieId	title	genres	rating
9055	160718	Piper (2016)	Animation	4.0
9061	161944	The Last Brickmaker in America (2001)	Drama	5.0
9062	162376	Stranger Things	Drama	4.5
9063	162542	Rustom (2016)	Romance|Thriller	5.0
9065	163949	The Beatles: Eight Days a Week - The Touring Y...	Documentary	5.0

In [142]:

```
is_sci_fi = box_office['genres'].str.contains('Sci-Fi')

box_office[is_sci_fi][:5]
```

Out[142]:

	movieId	title	genres	rating
23	24	Powder (1995)	Drama|Sci-Fi	3.044118
28	29	City of Lost Children, The (Cité des enfants p...	Adventure|Drama|Fantasy|Mystery|Sci-Fi	4.025000
31	32	Twelve Monkeys (a.k.a. 12 Monkeys) (1995)	Mystery|Sci-Fi|Thriller	3.923469
62	66	Lawnmower Man 2: Beyond Cyberspace (1996)	Action|Sci-Fi|Thriller	2.000000
70	76	Screamers (1995)	Action|Sci-Fi|Thriller	3.333333

In [143]:

```
box_office[is_sci_fi & is_highly_rated][-5:]
```

Out[143]:

	movieId	title	genres	rating
8854	134130	The Martian (2015)	Adventure|Drama|Sci-Fi	4.1
8869	135266	Zenon: The Zequel (2001)	Adventure|Children|Comedy|Sci-Fi	4.0
8886	136449	Ghost in the Shell 2.0 (2008)	Action|Animation|Sci-Fi	5.0
8934	140751	Almost Normal (2005)	Comedy|Drama|Sci-Fi	5.0
9039	158956	Kill Command (2016)	Action|Horror|Sci-Fi	4.0

Split genres into multiple columns

In [144]:

```
movie_genres = movies['genres'].str.split('|', expand=True)
```

In [146]:

```
movie_genres[:10]
```

Out[146]:

	0	1	2	3	4	5	6	7	8	9
0	Adventure	Animation	Children	Comedy	Fantasy	None	None	None	None	None
1	Adventure	Children	Fantasy	None	None	None	None	None	None	None
2	Comedy	Romance	None	None	None	None	None	None	None	None
3	Comedy	Drama	Romance	None	None	None	None	None	None	None
4	Comedy	None	None	None	None	None	None	None	None	None
5	Action	Crime	Thriller	None	None	None	None	None	None	None
6	Comedy	Romance	None	None	None	None	None	None	None	None
7	Adventure	Children	None	None	None	None	None	None	None	None
8	Action	None	None	None	None	None	None	None	None	None
9	Action	Adventure	Thriller	None	None	None	None	None	None	None

Add a new column for comedy genre flag

In [147]:

```
movie_genres['is_science_fi'] = movies['genres'].str.contains('Sci-Fi')
```

In [148]:

```
movie_genres[:10]
```

Out[148]:

	0	1	2	3	4	5	6	7	8	9
0	Adventure	Animation	Children	Comedy	Fantasy	None	None	None	None	None
1	Adventure	Children	Fantasy	None	None	None	None	None	None	None
2	Comedy	Romance	None	None	None	None	None	None	None	None
3	Comedy	Drama	Romance	None	None	None	None	None	None	None
4	Comedy	None	None	None	None	None	None	None	None	None
5	Action	Crime	Thriller	None	None	None	None	None	None	None
6	Comedy	Romance	None	None	None	None	None	None	None	None
7	Adventure	Children	None	None	None	None	None	None	None	None
8	Action	None	None	None	None	None	None	None	None	None
9	Action	Adventure	Thriller	None	None	None	None	None	None	None

Extract year from title e.g. (1995)

In [149]:

```
movies['year'] = movies['title'].str.extract('.*\((.*)\).*', expand=True)
```

In [150]:

```
movies.tail()
```

Out[150]:

	movieId	title	genres	year
9120	162672	Mohenjo Daro (2016)	Adventure|Drama|Romance	2016
9121	163056	Shin Godzilla (2016)	Action|Adventure|Fantasy|Sci-Fi	2016
9122	163949	The Beatles: Eight Days a Week - The Touring Y...	Documentary	2016
9123	164977	The Gay Desperado (1936)	Comedy	1936
9124	164979	Women of '69, Unboxed	Documentary	NaN

Parsing Timestamps

In [151]:

```
tags = pd.read_csv('./movielens/tags.csv', sep=',')
```

In [152]:

```
tags.head(5)
```

Out[152]:

	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 [ ]:

```
tags['parsed_time'] = pd.to_datetime(tags['timestamp'], unit='s')
```

In [155]:

```
tags['parsed_time'].dtype
```

Out[155]:

```
dtype('<M8[ns]')
```

In [156]:

```
tags.head(2)
```

Out[156]:

	userId	movieId	tag	timestamp	parsed_time
0	18	4141	Mark Waters	1240597180	2009-04-24 18:19:40
1	65	208	dark hero	1368150078	2013-05-10 01:41:18

Selecting rows based on timestamps

In [160]:

```
greater_than_t = tags['parsed_time'] > '2015-02-01'

selected_rows = tags[greater_than_t]

tags.shape, selected_rows.shape
```

Out[160]:

```
((465564, 5), (12130, 5))
```

Sorting the table using the timestamps

In [ ]:

```
tags.sort_values(by='parsed_time', ascending=True)[:10]
```

Average Movie Ratings over Time

Are Movie ratings related to the year of launch?

In [162]:

```
average_rating = ratings[['movieId','rating']].groupby('movieId', as_index=False).
average_rating.tail()
```

Out[162]:

	movieId	rating
9061	161944	5.0
9062	162376	4.5
9063	162542	5.0
9064	162672	3.0
9065	163949	5.0

In [ ]:

```
joined = movies.merge(average_rating, on='movieId', how='inner')
joined.head()
joined.corr()
```

In [164]:

```
yearly_average = joined[['year','rating']].groupby('year', as_index=False).mean()
yearly_average[:10]
```

Out[164]:

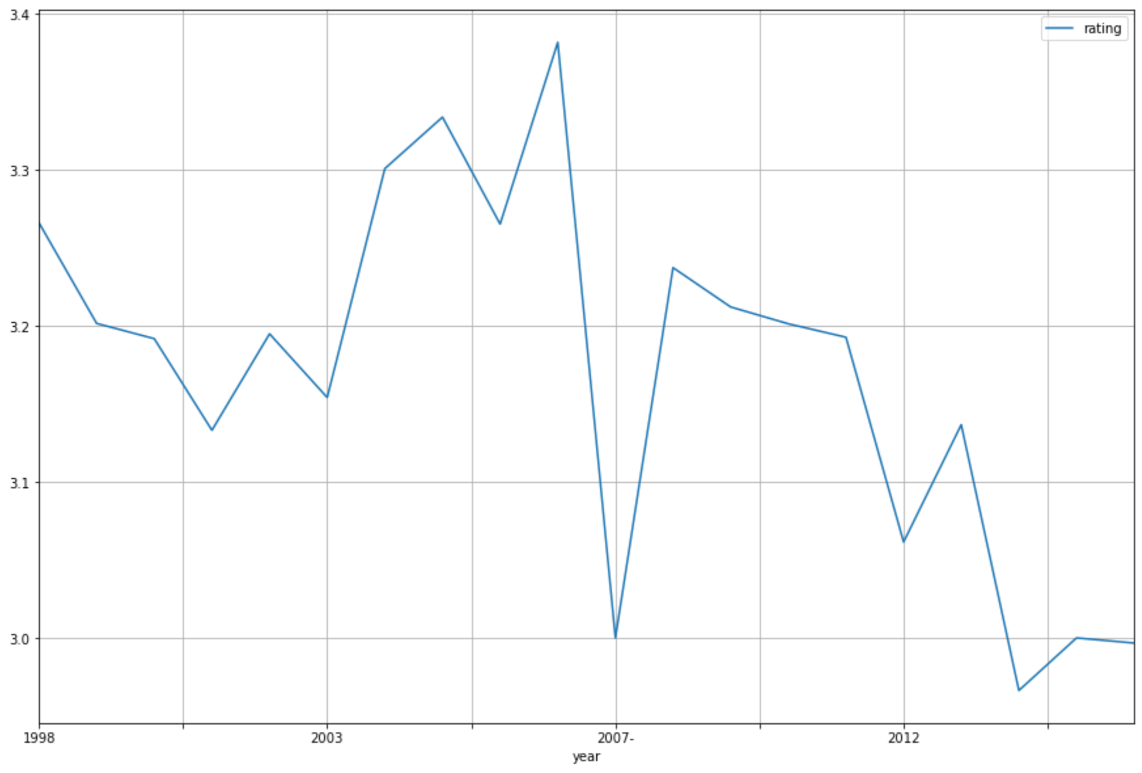
	year	rating
0	1902	4.333333
1	1915	3.000000
2	1916	3.500000
3	1917	4.250000
4	1918	4.250000
5	1919	3.000000
6	1920	2.500000
7	1921	4.387500
8	1922	3.926587
9	1923	4.166667

In [165]:

```
yearly_average[-20:].plot(x='year', y='rating', figsize=(15,10), grid=True)
```

Out[165]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x137f54b0>
```



In [ ]:

<Movie Data Analysis>

<James Karimi>

Dataset(s)

Which dataset did you use of the following:

- IMDB Movie Dataset

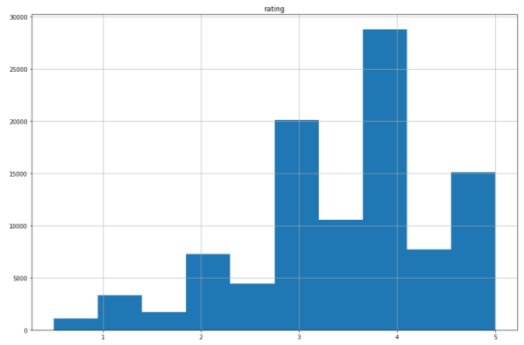
Motivation

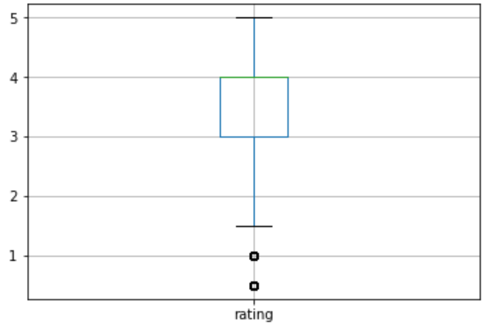
The aim was to analyse and describe the dataset and to evaluate the ratings of science fiction movies and across genres. Are their preferences for movies which are enjoyed most ? What do the ratings tell us ? This could give insight to movie makers.

Research Question(s)

Do science fiction movies tend to be rated more highly than other movie genres?

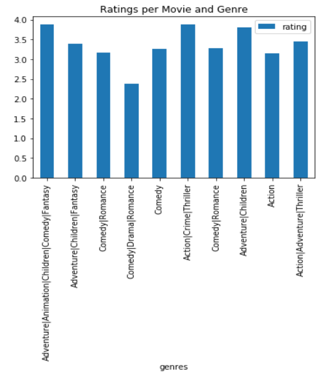
Are the distributions of ratings within genre comparable across genres ?





Findings

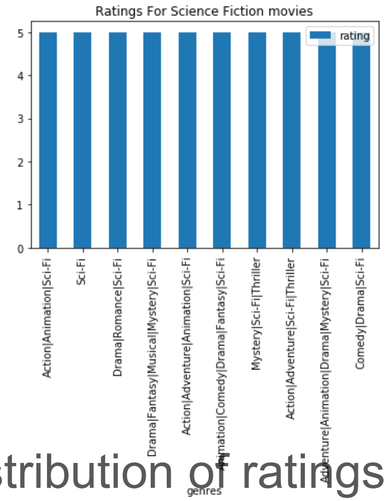
The mean average rating of Science Fiction movies is 3.167 below the mean average of genres 3.543.



The distribution of ratings within genre are comparable within genre. The 25 % and 75% distribution of Science Fiction ratings is between 2.698 and 3.75. Genres have a distribution rating between 3.00 and 4.00.

Findings

The distribution of ratings within genre are comparable within genre. The 25 % and 75% distribution of Science Fiction ratings is between 2.698 and 3.75. Genres have a distribution rating between 3.00 and 4.00.



Science fiction movie are not more highly rated and the distribution of ratings within genres are comparable across genres

Acknowledgements

I looked at the describe function in Python for mean average.

References

I did not use any reference papers.