SPRINGERBRIEFS IN STATISTICS

Dominique Haughton
Mark-David McLaughlin
Kevin Mentzer
Changan Zhang

Movie Analytics

A Hollywood Introduction to Big Data

Springer

SpringerBriefs in Statistics

More information about this series at http://www.springer.com/series/8921

Dominique Haughton
Mark-David McLaughlin
Kevin Mentzer
Changan Zhang

Movie Analytics

A Hollywood Introduction to Big Data

Springer

Dominique Haughton
Bentley University
Université Paris 1 Panthéon-Sorbonne
(SAMM) and Université
Toulouse 1 (GREMAQ)
Waltham, MA, USA

Kevin Mentzer
Business
Bentley University and Bryant University
Waltham, MA, USA

Mark-David McLaughlin
Information and Process Management
Bentley University and Cisco Systems
Waltham, MA, USA

Changan Zhang
Business
Bentley University and CTrip
Waltham, MA, USA

ISSN 2191-544X ISSN 2191-5458 (electronic)
SpringerBriefs in Statistics
ISBN 978-3-319-09425-0 ISBN 978-3-319-09426-7 (eBook)
DOI 10.1007/978-3-319-09426-7

Library of Congress Control Number: 2015946241

Springer Cham Heidelberg New York Dordrecht London

Printed on acid-free paper

Springer International Publishing AG Switzerland is part of Springer Science+Business Media (www.springer.com)

Acknowledgements

With our most sincere gratitude to Nathalie Villa-Vialaneix for contributing Chap. 4 to this monograph and to Veo (veocinemas.fr) for most kindly providing the data used in Chap. 4. We also wish to warmly thank Ignacio Alvarez-Hamelin for very useful conversations about the LaNet-vi visualization technique.

Contents

What Do We Know About Analyzing Movie Data?

A search in May 2015 for the term “Movie Analytics” yields about 2700 hits (inclusive of the link to this volume …).

Companies such as Cineplan which perform analytics applied to the motion picture industry are emerging and mainstream studios such as Walt Disney and Warner Brothers have their own analytics staff. Many of these efforts have in general focused on predicting the success of films (in popularity and revenue) in terms of a number of features such as budget. Other questions of interest include for example the optimal geographical positioning of movie theaters, or yet the survival time of a given movie in a given movie theater, discussed in work by Darlene Chisholm, an economist who specializes in problems related to the motion picture industry, and her colleagues (Chisholm and Norman 2005). A review of the economics literature devoted to the motion picture industry is given by McKenzie (2012).

Notably, a recent article by Gold et al. (2013) summarized the data science lessons which can be derived from the Academy Awards and the Wall Street Journal reported on the role of data science in Oscar prediction in their Numbers Guy column (Feb 23 2013).

The intent of this monograph is not to exhaustively review all these realms of movie analytics activities, but to instead provide an introduction to a set of analytics techniques which are of particularly current interest via the lens of the motion picture industry.

We begin by providing an overview of Big Data (Chap. 2) in order to distinguish how the data scientist’s (an individual looking to find information from data) view may differ from that of a computer scientist (who is examining better methods of storing and processing data). Next, Chap. 3 provides a discussion of how to visualize very large social networks, in the context of the co-starring network with its over 2.6 million actors, extracted in 2013 from the IMDb database. We demonstrate how computer memory issues arise and how to handle them. The nitty-gritty of how to construct the files needed for the analysis is discussed as well. We then turn our attention to two different approaches to visualizing the sub-networks of most connected, and least connected actors, and how these two approaches differ. We note

D. Haughton et al., *Movie Analytics*, SpringerBriefs in Statistics,
DOI 10.1007/978-3-319-09426-7_1

here that interesting efforts (in the context of the VAST 2013 challenge) at using visualization tools to help predict box-office revenues and ratings include El Assady et al. (2013).

We then move to a discussion of movie theater attendance in Chap. 4, using data from eight movie theaters in France. And in Chap. 5, we explain the role of predictive markets such as Intrade (which has now ceased to operate) or Paddy Power in Oscar prediction.

Finally, in Chap. 6, we demonstrate how to analyze Twitter and text reviews of movies nominated for best picture in the 2013 Academy Awards. While we remain aware of the fact that numeric review scores are likely to be more predictive of movie success than text analytics of reviews or tweets (Gold et al. 2013), we see here an excellent opportunity to demonstrate text analytics techniques. We therefore demonstrate how to employ tools such as the SAS TextMiner and the R package RTemis to extract main themes from movie reviews. In so doing, we give precedence to theme extraction based on a reduction of the term-document matrix, rather than a topic modeling approach. We also provide an introduction to sentiment analysis with Python, applied to movie reviews.

Finally, we allude to the possibility of the impact of "controversy" on Oscar success, and suggest that a first attempt at measuring such controversy might be to look at the variability of numeric review scores for a given movie. We also mention here work by Topsy Labs in 2013 to create a "Twitter Oscars index" (http://oscars.topsy.com/).

What Does “Big Data” Mean? The Data Scientist Point of View

The term “Big Data” means different things to different constituencies. In 2010 (February 27), The Economist published an extensive report titled “The data deluge” which discussed the expected benefits as well as headaches from the overabundance of data in all spheres of human activity.

Data have been defined to be “big” if any of the following three measures *Volume*, *Velocity*, *and Variety* lie beyond levels manageable by a typical computer. These measures are commonly referred to as the “three Vs of big data.” Additional measures such as *Veracity*, *Value* have led to mentions of “five Vs of big data,” but volume, velocity, and variety remain the most central dimensions of what is considered as “big data.” The size of “big” data sets is much larger (two million records and 100 variables would be considered to a be a moderate size), data arrive at a much faster pace (notably via social networks) and come in a wide range of formats, numerical, textual, visual, audio, etc.

Publications about “big data” have proliferated, in the form of monographs and journal articles, as well as in the wider press. Published work ranges from the general to the specifically technical. Notably we refer the reader to books by Davenport (2014) and Franks (2012) for a general focus on analytics and business.

It is interesting to note that the term *Analytics* has existed since the early 1990s and is a term coined by the corporate world, not by academics. Analytics, at its inception, meant “data analysis applied to business problems.” By 2015, the terms *Big Data*, *Analytics*, *Data Science*, *Deep Learning*, *Machine Learning*, etc. have become essentially interchangeable terms. The evolution of the use of these terms over a period ranging from 1970 to 2008 is displayed in Fig. 1.

D. Haughton et al., *Movie Analytics*, SpringerBriefs in Statistics,
DOI 10.1007/978-3-319-09426-7_2

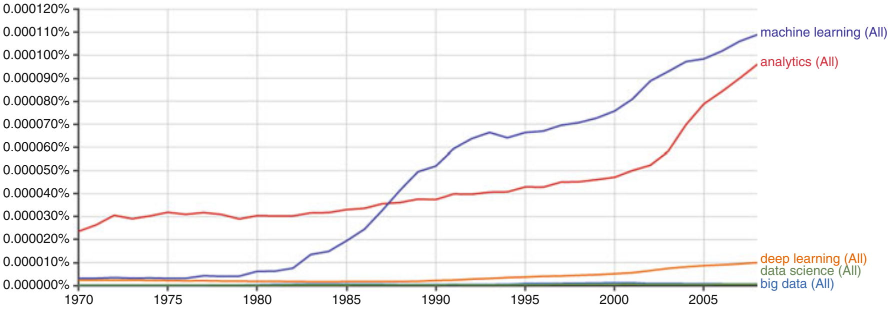


Fig. 1 Number of ngrams for all five terms from 1970 to 2008 on the same graph (Source: Google n-grams)

Visualization of Very Large Networks: The Co-starring Social Network

1 Introduction

Studies and attempts at visualizing the co-starring network, in particular using the Internet Movie Database (IMDb, owned by Amazon) network, as well as various attempts at visualizing other aspects of movies, such as the network of actors featured in the same scene within a movie, story lines etc. abound in published and unpublished reports. We do not attempt here to extensively review this literature, but a simple search of keywords such as visualization, IMDb database, co-starring network yields a representative set of graphs and reports. To the best of our knowledge, no one has found a way of representing the entire network of about 2.6 million actors, in large part because of the difficulties with fitting the co-starring network of these numerous actors into memory. Another challenge is that of making sense of any representations, given that identifying a particular actor in such vast representations is difficult as well.

Because of these memory and processing constraints, we propose to employ a k-core approach to this visualization and analysis challenge. We rely on past work and algorithms to compare the visualizations we obtain via the k-core approach with those obtained by more traditional methods.

2 The IMDb Data

The IMDb database is updated monthly. For demonstration purposes we use here the data from February 2013 up to October 2014. These files are available via anonymous FTP from a variety of locations. The IMDb states that the data are not free but can be used for free in certain situations. See http://www.imdb.com/interfaces for the latest information on the terms of use and FTP locations.

D. Haughton et al., *Movie Analytics*, SpringerBriefs in Statistics,
DOI 10.1007/978-3-319-09426-7_3

Table 1 IMDb titles: 3,196,749 (year range: 1974–2014)

Number	Title type
2,005,173	TV episode
355,382	short
323,161	feature
88,818	TV series
88,781	video
77,057	TV movie
71,345	documentary short
62,543	documentary
28,242	video short
25,909	TV documentary
14,585	video game
11,856	TV series documentary
10,555	video documentary short
10,056	video documentary
9536	TV mini-series
4175	TV short
3937	TV special
1762	TV mini-series documentary
1653	TV documentary short
995	TV series short
880	TV special documentary
129	TV mini-series short
79	TV series documentary short
72	TV special documentary short
38	TV special short
11	video game short
10	video game documentary
9	TV mini-series documentary short

The IMDB is a crowdsourced database. As can be seen in Table 1, there are over three million titles in the database of which approximately two million are TV Show episodes. Feature length films make up only 10.1 % of the titles in the database.

In addition to titles, the other main data classification is related to people (Table 2). There are over six million individual people in the database of which the largest group is that of actors at 25.7 %, followed by actresses at 14.9 %. We can see that a single person can fall into multiple categories, resulting in a total number of people across all categories of 9.4 million. The average actor is credited 7.0 times while the average actress is credited 7.6 times.

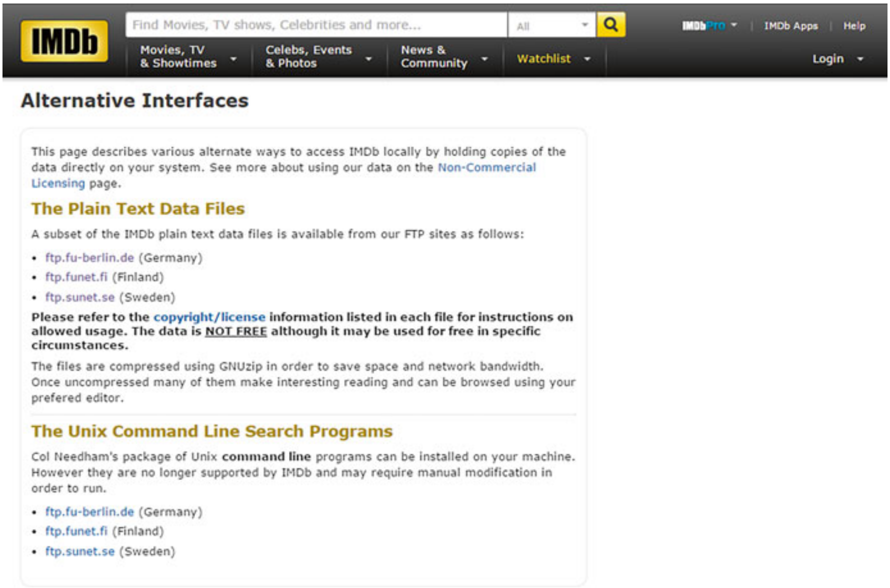
The data sets are available on three different mirror sites located in Germany, Finland, and Sweden (Fig. 1). A partial list of the files available can be seen in Fig. 2. When using this dataset it is important to review any README documentation to confirm understanding the terms of use along with specific requests from the mirror

Table 2 IMDb people: 6,383,724 (Credits: 64,366,326)

# People	# Credits	Category
1,641,391	11,559,944	Actor
951,336	7,198,320	Actress
850,501	5,587,769	Miscellaneous Crew
662,594	5,311,215	Producer
869,686	4,617,656	Self
440,089	3,474,492	Camera and Electrical Department
495,840	3,418,306	Writer
253,729	2,909,456	Sound Department
377,182	2,142,756	Director
257,958	1,900,024	Art Department
253,417	1,441,892	Editor
118,874	1,200,777	Editorial Department
117,932	1,171,991	Make-Up Department
126,684	1,153,327	Music Department
111,496	1,136,065	Animation Department
221,382	1,121,808	Cinematographer
162,112	1,014,043	Second Unit Director or Assistant Director
133,198	1,013,755	Visual Effects
131,859	982,034	Production Manager
169,634	963,476	Composer
94,289	722,312	Soundtrack
83,094	652,762	Costume and Wardrobe Department
54,875	545,199	Stunts
366,150	526,406	Thanks
62,065	385,996	Production Designer
62,328	369,548	Art Director
35,429	357,106	Casting Department
24,952	333,357	Casting Director
48,604	332,990	Costume Designer
55,797	286,892	Transportation Department
49,145	282,022	Special Effects
42,382	252,403	Set Decorator
118	190	Production Department
9	16	Location Management
13	13	Electrical Department
3	8	Script and Continuity Department
89,830	299,550	Archive Footage

sites (some sites will ask to please limit large data transfers to off-hours; it is useful to remember times zones here).

An IMDb community (Fig. 3) is available at https://getsatisfaction.com/imdb.



Alternative Interfaces

This page describes various alternate ways to access IMDb locally by holding copies of the data directly on your system. See more about using our data on the Non-Commercial Licensing page.

The Plain Text Data Files

A subset of the IMDb plain text data files is available from our FTP sites as follows:

- ftp.fu-berlin.de (Germany)
- ftp.funet.fi (Finland)
- ftp.sunet.se (Sweden)

Please refer to the copyright/license information listed in each file for instructions on allowed usage. The data is NOT FREE although it may be used for free in specific circumstances.

The files are compressed using GNUzip in order to save space and network bandwidth. Once uncompressed many of them make interesting reading and can be browsed using your prefered editor.

The Unix Command Line Search Programs

Col Needham's package of Unix **command line** programs can be installed on your machine. However they are no longer supported by IMDb and may require manual modification in order to run.

- ftp.fu-berlin.de (Germany)
- ftp.funet.fi (Finland)
- ftp.sunet.se (Sweden)

Fig. 1 IMDb mirror sites

Table 3 gives summary statistics on the entire IMDb database, and on subsets consisting of the most connected actors, with degrees of 30,000 and above and of “minor actors”, with degrees of 10–50.

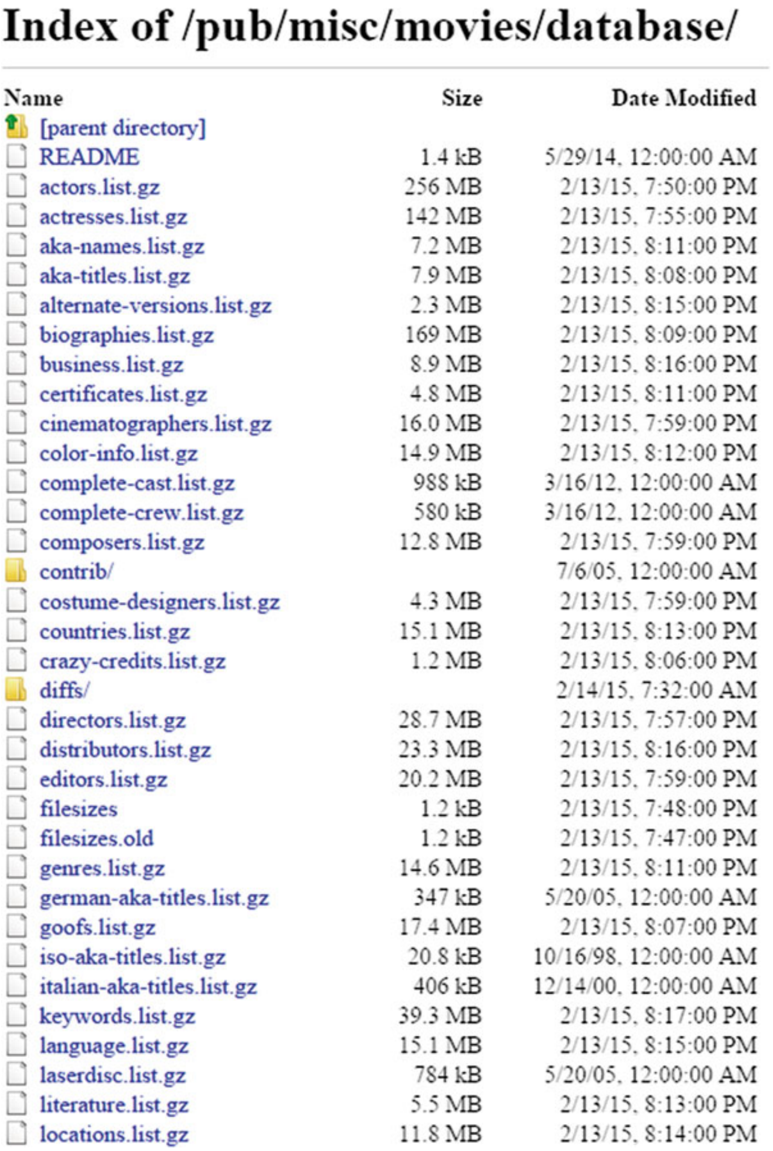
Figure 4 illustrates the degree distribution of the whole network with a histogram of the decimal logarithms of degrees. The histogram reveals the presence of two essential groups of actors, a large group of less connected actors (with degrees centered at about 40) and a smaller group of more connected actors (with degrees centered at about 1600).

3 Issues with Large Datasets

Programs that are used to process the data require a great deal of memory (RAM) in order to successfully load and parse large datasets. For that reason, many of the libraries and applications that are typically installed on a computer may need to be changed to use their 64-bit versions. While 64-bit applications have been around for quite some time, support for many third party packages varies.

Specifically, the 64-bit versions of Python and Java need to be installed. They can be downloaded at the following locations:

http://www.java.com/en/download/manual.jsp
http://python.org/download/



Index of /pub/misc/movies/database/

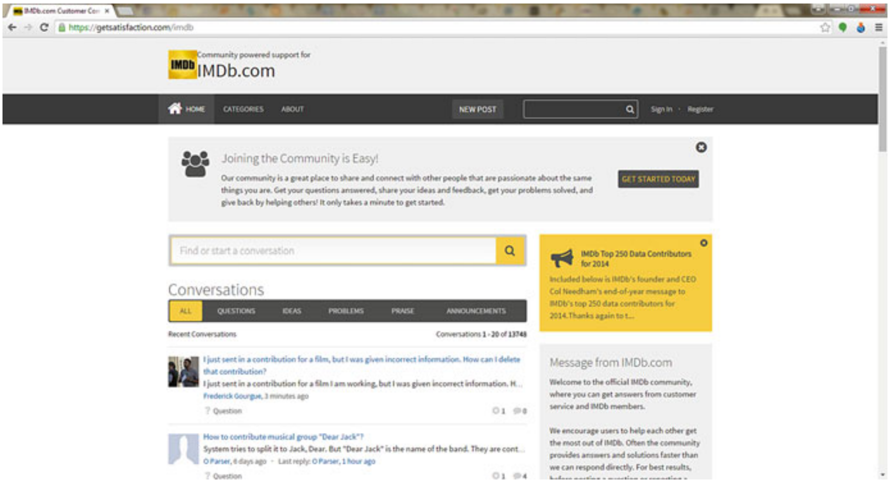
Name	Size	Date Modified
[parent directory]		
README	1.4 kB	5/29/14, 12:00:00 AM
actors.list.gz	256 MB	2/13/15, 7:50:00 PM
actresses.list.gz	142 MB	2/13/15, 7:55:00 PM
aka-names.list.gz	7.2 MB	2/13/15, 8:11:00 PM
aka-titles.list.gz	7.9 MB	2/13/15, 8:08:00 PM
alternate-versions.list.gz	2.3 MB	2/13/15, 8:15:00 PM
biographies.list.gz	169 MB	2/13/15, 8:09:00 PM
business.list.gz	8.9 MB	2/13/15, 8:16:00 PM
certificates.list.gz	4.8 MB	2/13/15, 8:11:00 PM
cinematographers.list.gz	16.0 MB	2/13/15, 7:59:00 PM
color-info.list.gz	14.9 MB	2/13/15, 8:12:00 PM
complete-cast.list.gz	988 kB	3/16/12, 12:00:00 AM
complete-crew.list.gz	580 kB	3/16/12, 12:00:00 AM
composers.list.gz	12.8 MB	2/13/15, 7:59:00 PM
contrib/		7/6/05, 12:00:00 AM
costume-designers.list.gz	4.3 MB	2/13/15, 7:59:00 PM
countries.list.gz	15.1 MB	2/13/15, 8:13:00 PM
crazy-credits.list.gz	1.2 MB	2/13/15, 8:06:00 PM
diffs/		2/14/15, 7:32:00 AM
directors.list.gz	28.7 MB	2/13/15, 7:57:00 PM
distributors.list.gz	23.3 MB	2/13/15, 8:16:00 PM
editors.list.gz	20.2 MB	2/13/15, 7:59:00 PM
filesizes	1.2 kB	2/13/15, 7:48:00 PM
filesizes.old	1.2 kB	2/13/15, 7:47:00 PM
genres.list.gz	14.6 MB	2/13/15, 8:11:00 PM
german-aka-titles.list.gz	347 kB	5/20/05, 12:00:00 AM
goofs.list.gz	17.4 MB	2/13/15, 8:07:00 PM
iso-aka-titles.list.gz	20.8 kB	10/16/98, 12:00:00 AM
italian-aka-titles.list.gz	406 kB	12/14/00, 12:00:00 AM
keywords.list.gz	39.3 MB	2/13/15, 8:17:00 PM
language.list.gz	15.1 MB	2/13/15, 8:15:00 PM
laserdisc.list.gz	784 kB	5/20/05, 12:00:00 AM
literature.list.gz	5.5 MB	2/13/15, 8:13:00 PM
locations.list.gz	11.8 MB	2/13/15, 8:14:00 PM

Fig. 2 IMDb data sets

4 Configuring Python and Gephi to Use the 64-Bit Libraries

4.1 Python

Python (2015) is an open-source programming language which provides a number of packages which can be used for many different applications, somewhat in the same way as R. It is powerful, fast and tricky to install, compared to R.



Joining the Community is Easy!

Our community is a great place to share and connect with other people that are passionate about the same things you are. Get your questions answered, share your ideas and feedback, get your problems solved, and give back by helping others! It only takes a minute to get started.

Conversations

Recent Conversations

Conversations 1 - 20 of 13748

I just sent in a contribution for a film, but I was given incorrect information. How can I delete that contribution?

I just sent in a contribution for a film I am working, but I was given incorrect information. H...

Frederick Gourgue, 3 minutes ago

How to contribute musical group "Dear Jack"?

System tries to split it to Jack, Dear. But "Dear Jack" is the name of the band. They are cont...

O Parser, 6 days ago · Last reply: O Parser, 1 hour ago

IMDb Top 250 Data Contributors for 2014

Included below is IMDb's founder and CEO Col Needham's end-of-year message to IMDb's top 250 data contributors for 2014.Thanks again to t...

Message from IMDb.com

Welcome to the official IMDb community, where you can get answers from customer service and IMDb members.

We encourage users to help each other get the most out of IMDb. Often the community provides answers and solutions faster than we can respond directly. For best results,

Fig. 3 IMDb community

Table 3 Summary statistics for the IMDb database

Network/partition	Number of actors	Number of connections	Average degree
Complete Network	2,614,790	1,083,901,049	834.05
Most Connected (degrees of 30k+)	8432	56,661,389	13,439.61
Minor Actors (degrees of 10–50)	986,031	5,584,885	11.33

We used Python version 2.7.3 in writing this monograph; it can be obtained from http://python.org/download/releases/2.7.3/. The 64-bit version is the *Windows X76-65 MSI Installer*. If both the 32-bit and 64-bit versions are installed, they should be installed in different directories. In this monograph, we installed the library in C:\Python27-64. The Appendix provides sample Python code for parsing the IMDb database. The code generates a variety of output files which can be imported into Pajek or Gephi as will be discussed in the following sections.

Attempting to load large files in the 32-bit version is likely to yield a memory error as shown in Fig. 5.

The Windows PATH should include "C:\Python27-64;C:\Python27-64\Scripts;C:\Python27-64\Tools\Scripts". This is done by editing the system variable "PATH" as shown in Fig. 6a–c.

In order to get the many libraries needed to process various datasets, it may be helpful to install the setup tools. They can be obtained at https://pypi.python.org/pypi/setuptools. In order to use the 64-bit version, the ez_setup.py application must be downloaded and run from the command line with "python ez_setup.py". We note that one needs to open a new command shell after changing any system variables in order to use the new settings.

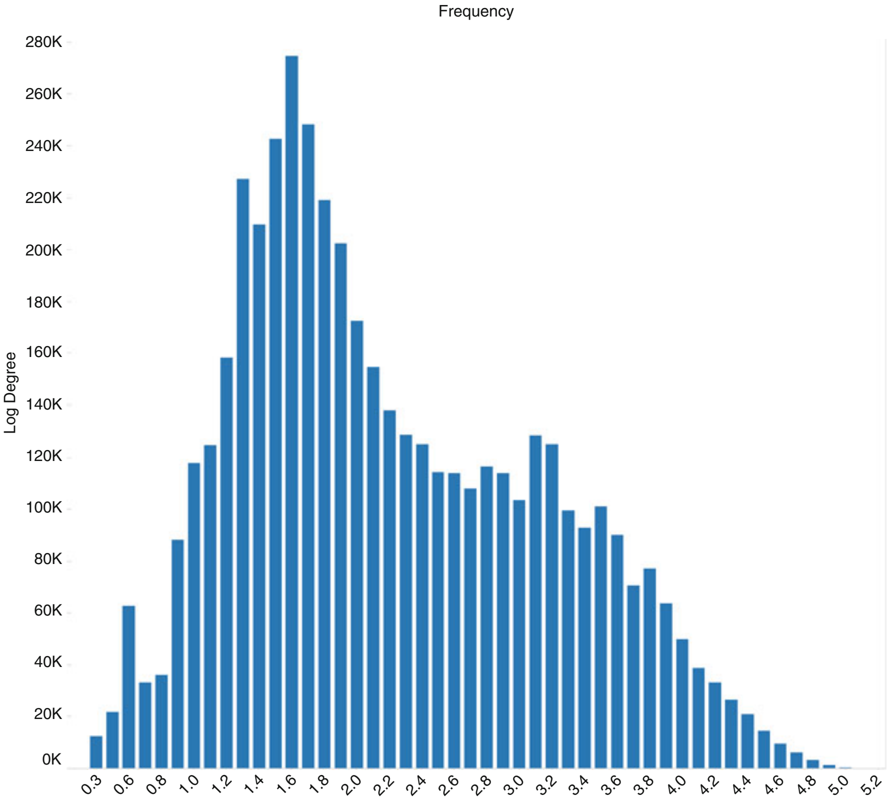
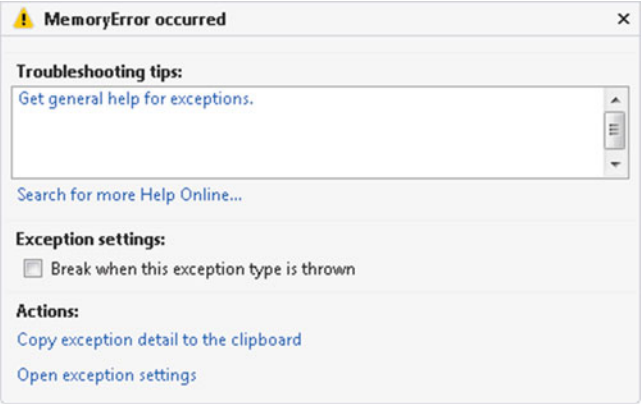


Fig. 4 Histogram of decimal logarithms of degrees in the complete network


MemoryError occurred

Troubleshooting tips:

Get general help for exceptions.

Search for more Help Online...

Exception settings:

□ Break when this exception type is thrown

Actions:

Copy exception detail to the clipboard

Open exception settings

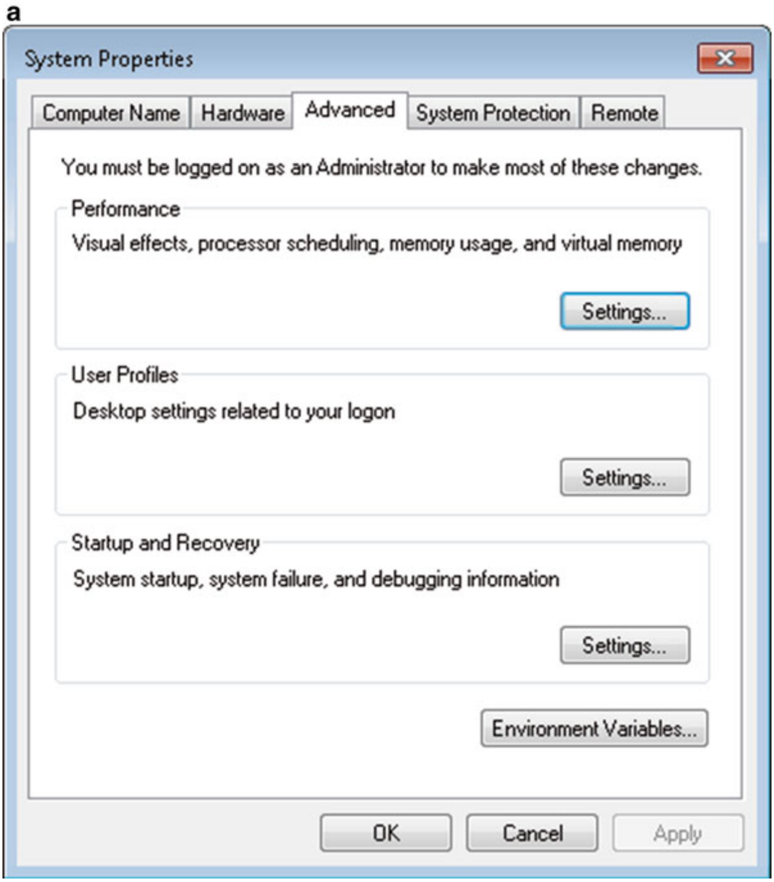
Fig. 5 Python 32 bit memory error

The examples in this monograph use a number of libraries that can now be obtained with easy_install.

However, unfortunately, many of the 64 bit libraries are not available through easy install. For example in order to create a matrix, we use the numPy package which has to be downloaded manually from: http://www.lfd.uci.edu/~gohlke/pythonlibs/#numpy. It is important to match the version with the version of Python installed on the system.

4.2 Gephi

Gephi (2015) is "an interactive visualization and exploration platform for all kinds of networks and complex systems, dynamic and hierarchical graphs".



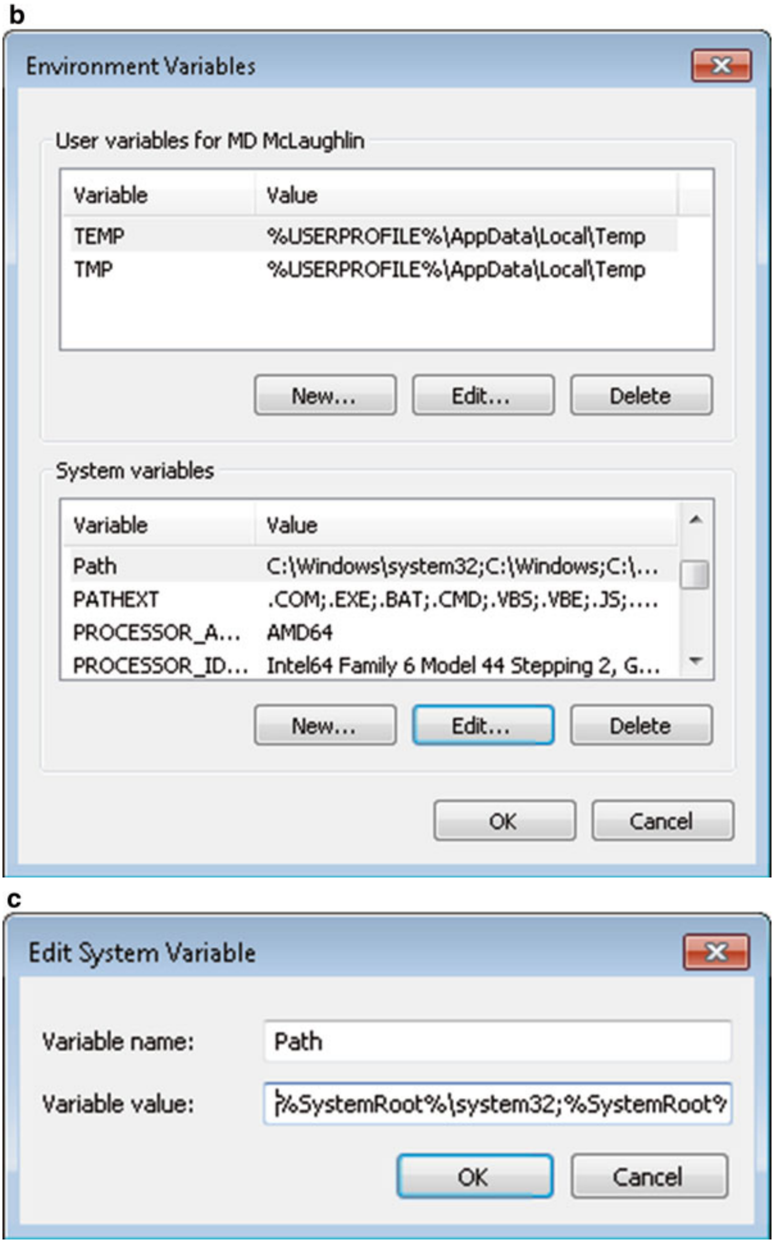
You must be logged on as an Administrator to make most of these changes.

Visual effects, processor scheduling, memory usage, and virtual memory

Desktop settings related to your logon

System startup, system failure, and debugging information

Fig. 6 (**a**) Editing the system variable PATH, first step. (**b**) Editing the system variable PATH, second step. (**c**) Editing the system variable PATH, third step



User variables for MD McLaughlin

Variable	Value
TEMP	%USERPROFILE%\AppData\Local\Temp
TMP	%USERPROFILE%\AppData\Local\Temp

System variables

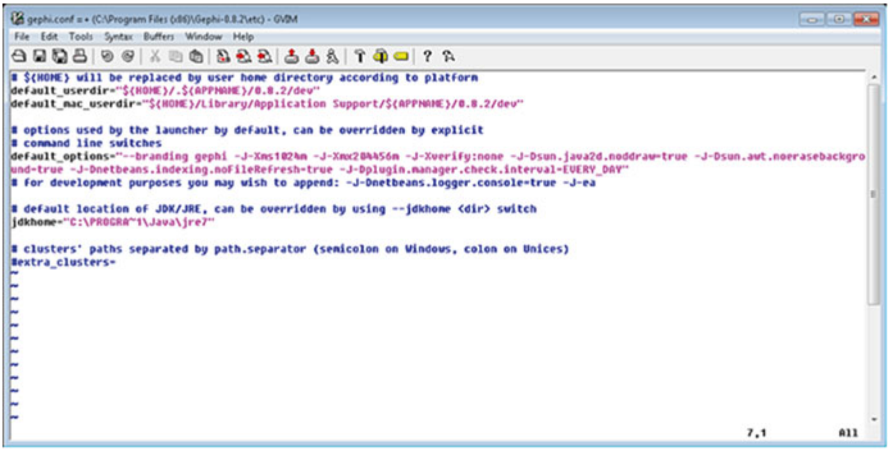
Variable	Value
Path	C:\Windows\system32;C:\Windows;C:\...
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;....
PROCESSOR_A...	AMD64
PROCESSOR_ID...	Intel64 Family 6 Model 44 Stepping 2, G...

Variable name: Path

Variable value: %SystemRoot%\system32;%SystemRoot%

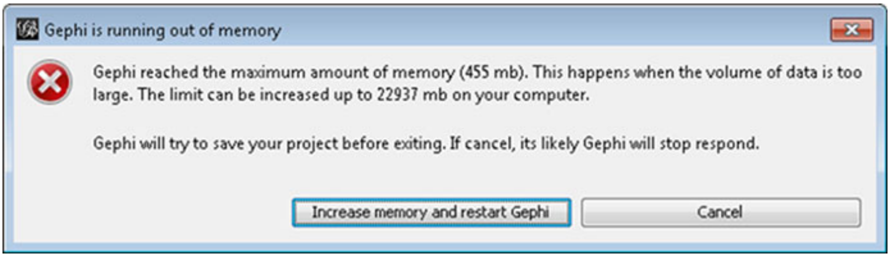
Fig. 6 (continued)

As displayed in Fig. 7, the Gephi configuration file C:\Program Files (x86)\Gephi-0.8.2\etc\gephi.conf should be edited to point the jdkhome variable to "C:\PROGRA~1\Java\jre7" or to the location where the 64-bit version of Java was installed. The default Gephi memory option should be modified to allow for large datasets.


gephi.conf = • (C:\Program Files (x86)\Gephi-0.8.2\etc) - GVIM
File Edit Tools Syntax Buffers Window Help

```
# ${HOME} will be replaced by user home directory according to platform
default_userdir="${HOME}/.${APPNAME}/0.8.2/dev"
default_mac_userdir="${HOME}/Library/Application Support/${APPNAME}/0.8.2/dev"

# options used by the launcher by default, can be overridden by explicit
# command line switches
default_options="--branding gephi -J-Xms1024m -J-Xmx204456m -J-Xverify:none -J-Dsun.java2d.noddraw=true -J-Dsun.awt.noerasebackgro
und=true -J-Dnetbeans.indexing.noFileRefresh=true -J-Dplugin.manager.check.interval=EVERY_DAY"
# for development purposes you may wish to append: -J-Dnetbeans.logger.console=true -J-ea

# default location of JDK/JRE, can be overridden by using --jdkhome <dir> switch
jdkhome="C:\PROGRA~1\Java\jre7"

# clusters' paths separated by path.separator (semicolon on Windows, colon on Unices)
#extra_clusters=
~
~
~
~
~
~
~
~
~
~
~
~
                                                                   7,1           All
```

Fig. 7 Gephi memory option


Gephi is running out of memory

Gephi reached the maximum amount of memory (455 mb). This happens when the volume of data is too large. The limit can be increased up to 22937 mb on your computer.

Gephi will try to save your project before exiting. If cancel, its likely Gephi will stop respond.

Increase memory and restart Gephi
Cancel

Fig. 8 Gephi memory issues

If Gephi runs out of memory, an error such as displayed in Fig. 8 will emerge.

The upper limit will be determined by how much RAM is installed on the system. This upper value will likely need to be changed in the configuration file by altering the parameter –Xmx[value].

5 Visualizations of the Co-starring Network

In the next sections, we first briefly describe the k-core approach to visualizing and finger-printing large networks, such as advocated in Alvarez et al. (2005). We then examine representations of two subsets of the IMDb co-starring network, consisting of very connected actors with degrees of 30,000 and above, and of “minor actors” with degrees between 10 and 50.

6 The Pajek and Gephi Approaches

Pajek (2015) is very effective at processing network files and creating partitions, but visualization is lacking compared to some other tools available such as Gephi. For example, when we attempt to visualize nodes with a large number of edges (as in our IMDb dataset), the Pajek software locks up and displays the image in Fig. 9.

6.1 The Gephi Approach

We therefore turn to another software package mentioned earlier, Gephi, in order to visualize our partitioned data. In Gephi, in order to process the data we perform the following steps:

1. load up the entire dataset
2. partition the data (filter the data by degrees)
3. select the algorithm to display the network
4. analyze the network

6.2 Loading the Data

Loading the data into Gephi is quite simple. Typically a list of edges is entered via the Gephi data laboratory.

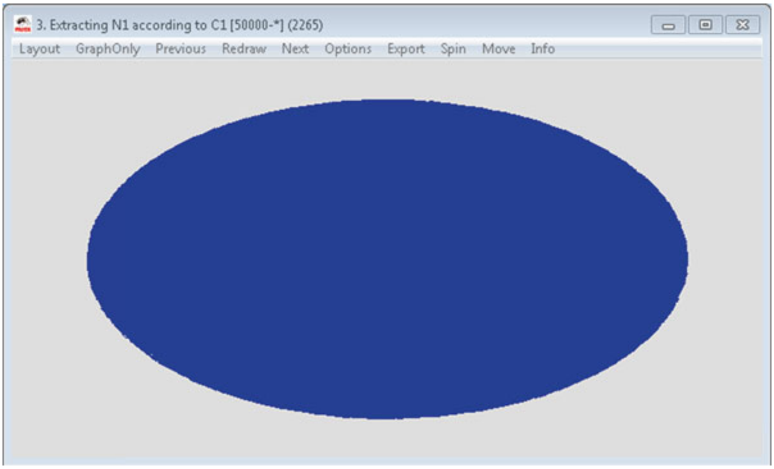

3. Extracting N1 according to C1 [50000-*] (2265)
Layout GraphOnly Previous Redraw Next Options Export Spin Move Info

Fig. 9 Pajek attempt

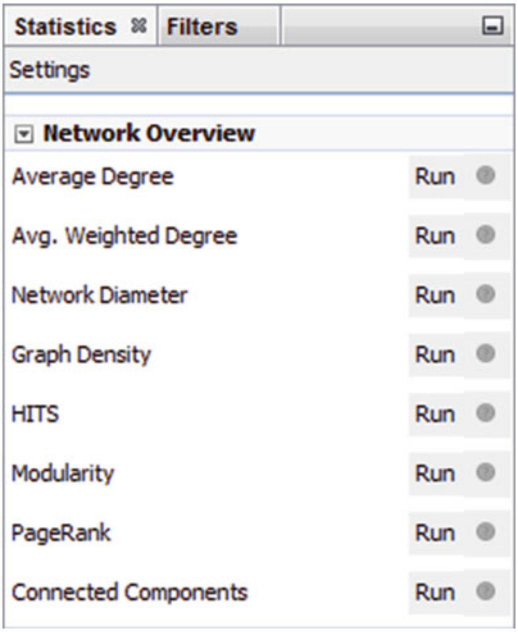


Fig. 10 Data partition in Gephi step 1

6.3 Partitioning the Data

In order to partition the data in Gephi, one must first calculate the variable to be used for partitioning. In our example, we will filter the network by degrees. Therefore, we must run the *Run Average Degree* statistic from the options on the right as shown in Fig. 10.

We must then click on the *Filters* tab, open the *Attributes* menus, and then select the Degree by dragging it from the *Range* menu to the *Queries* as shown in Fig. 11.

A tool to filter the degree similar to the image in Fig. 12 will be displayed below the *Queries* window. One can select the lower and upper ranges by dragging the slider bars on either side of the selection bar, and then click the *Filter* button to partition the data.

6.4 Displaying the Network

After the network is partitioned, a layout algorithm will help us make sense of the data. In our examples, we used the *ForceAtlas 2* algorithm as shown in Fig. 13. As described in Jacomy et al. (2014) *ForceAtlas2* is a force directed layout: it simulates a physical system. Nodes repulse each other (like magnets, with a force which depends on the degree of the two nodes) while edges attract the nodes they connect (like springs, with a force which depends linearly on the distance between the two nodes). These forces create a motion which converges to a balanced state.

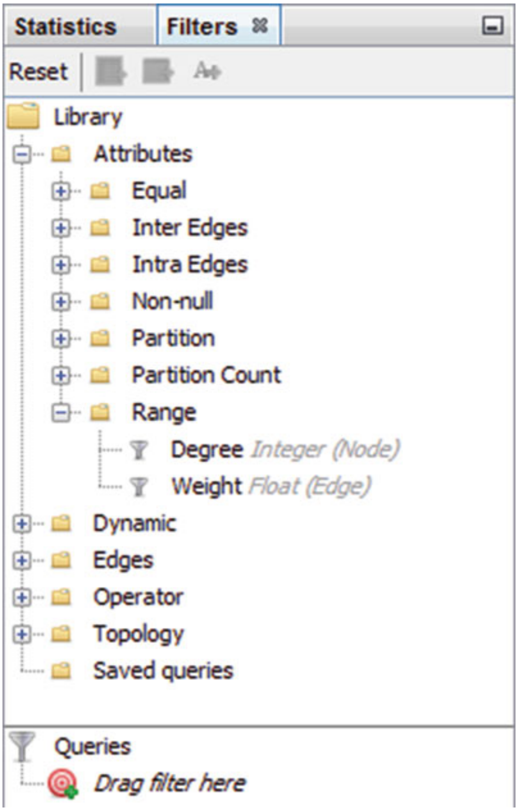


Fig. 11 Data partition in Gephi step 2

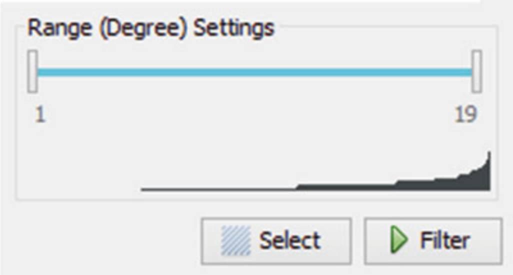
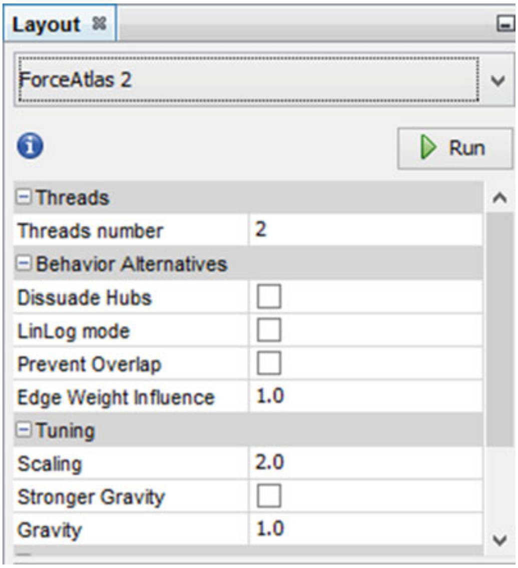


Fig. 12 Data partition in Gephi step 3

6.5 *Interpreting the Data*

We decided to first examine the network of the most connected actors. Therefore we partitioned the network for actors with the highest degrees (30,000 and above) and visualized the network with the *ForceAtlas 2* algorithm. This is shown in Fig. 14 which reveals the presence of clusters within the network. Interpreting the dark areas of the network as a cluster there appears to be about 7 at first sight.

Fig. 13 Layout algorithm



ForceAtlas 2

Threads

Threads number 2

Behavior Alternatives

Dissuade Hubs □

LinLog mode □

Prevent Overlap □

Edge Weight Influence 1.0

Tuning

Scaling 2.0

Stronger Gravity □

Gravity 1.0

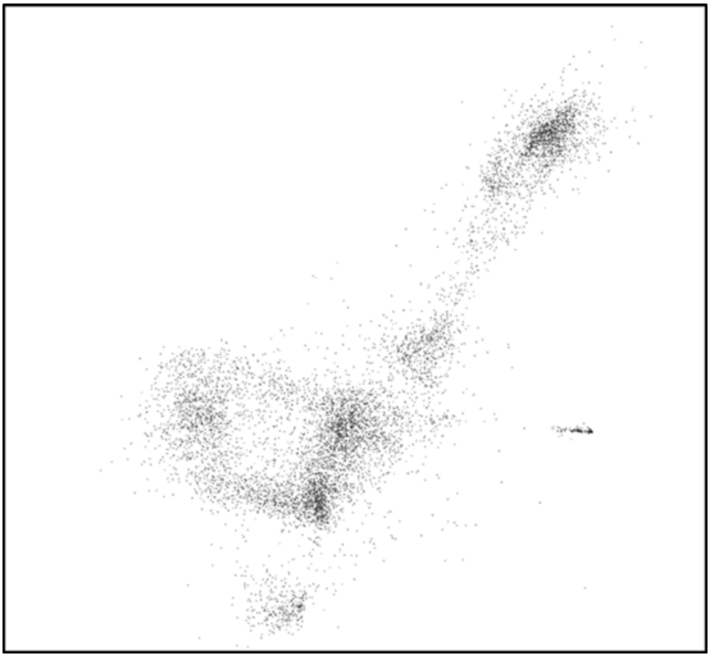


Fig. 14 Representation of the network of most connected actors with Gephi

Figure 15 displays the partitioning of actors into groups with the same degree; each color represents a different value of the degree. It gives a sense of a tight-knit core where actors of different degrees are linked, with several more peripheral groups where actors of different degrees are also linked.

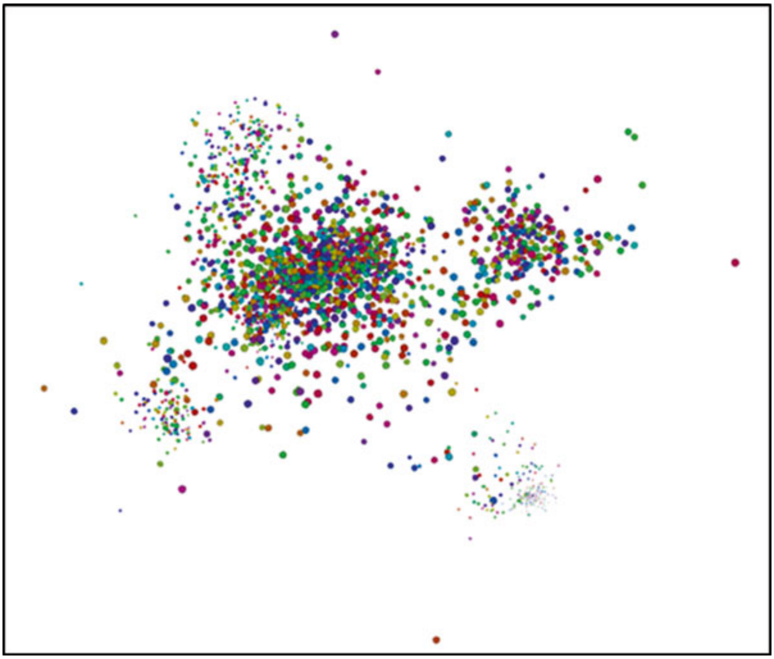


Fig. 15 Partitioning by degree of the network of most connected actors with Gephi

We then partition the data to include the least connected actors (with degrees from 10 to 50) and obtain a very diffused network with many inter- and intra-connected small groups as shown in Fig. 16.

In the next section, we will compare these graphs with those obtained via a k-core approach.

7 K-Core Representations of the IMDb Co-starring Network

7.1 K-Core Decomposition

This brief exposition of the k-core approach to visualizing and finger-printing networks relies on the work by Alvarez et al. (2005). To simplify matters, we consider here undirected networks only in this monograph. The following are some key definitions:

- A sub-graph induced by a graph C is a graph on a subset of vertices of C that includes only edges linking subset vertices. For example, in Fig. 17 the red-colored vertices and the edges that link them form a sub-graph.
- A sub-graph H of C is a k-core if all vertices in H have degree *in H* greater than or equal to k and H is the maximal sub-graph with this property.

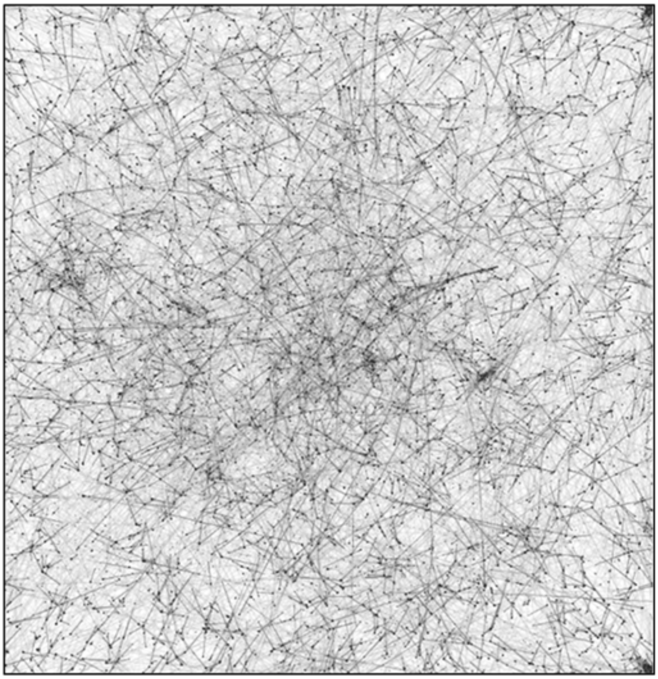


Fig. 16 Representation of the minor actor network with Gephi

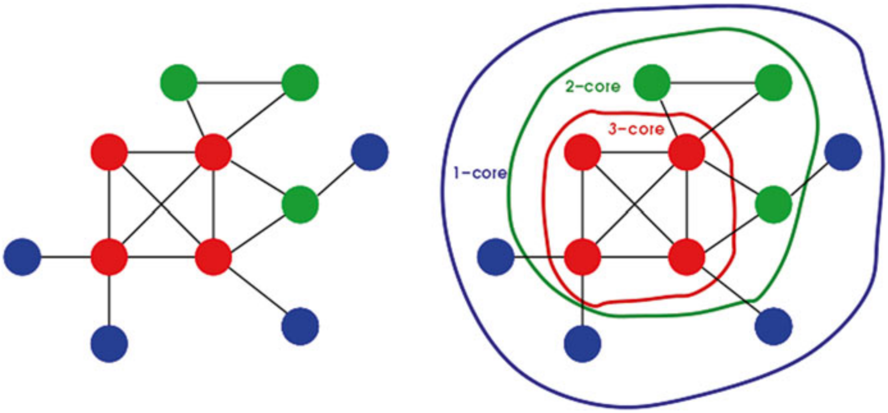


Fig. 17 Representation of k-cores and k-shells, borrowed from Alvarez et al. (2005)

- A vertex has shell index c if it belongs to a c-core but not a (c + 1)-core.
- A shell of order c is composed of all vertices whose shell index is c.
- Each connected (within the original graph) set of vertices with shell index c forms a cluster of order c.

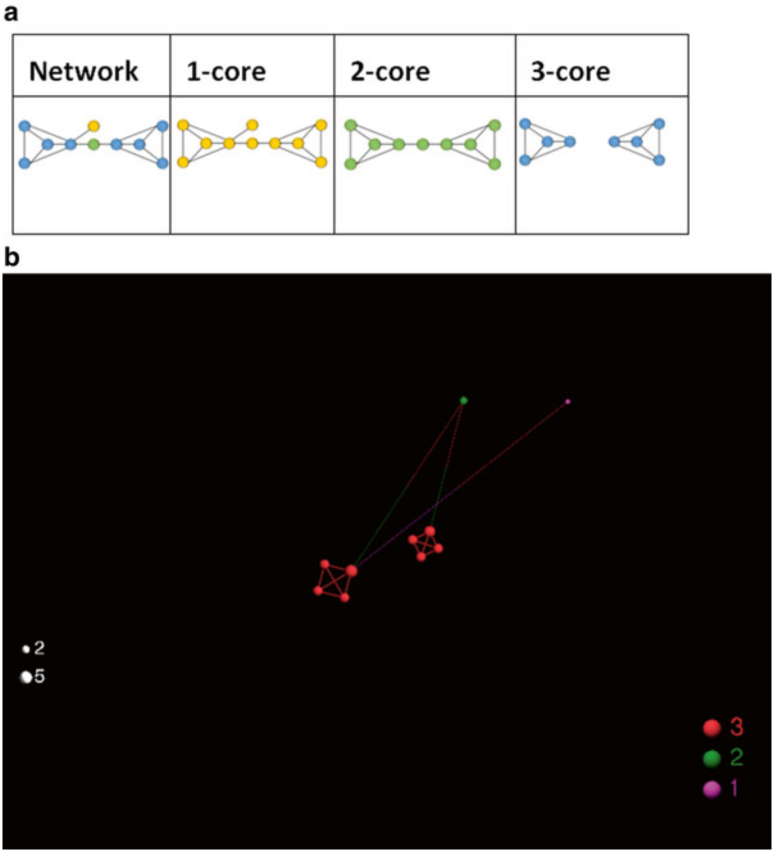


Fig. 18 (**a**) K-cores can be disconnected (adapted from De Nooy et al. 2011). (**b**) LaNet-vi2 rendition of the graph in (**a**)

In Fig. 17, the 1-, 2- and 3-cores are delimited by blue, green and red boundaries respectively, and the colors represent the shells (blue for the 1-shell, green for the 2-shell, red for the 3-shell).

Figure 18a gives another example of the decomposition into k-shells of a simple network due to De Nooy et al. (2011), where it is clear that a k-core can be disconnected, even if the initial graph is connected. Figure 18b gives the LaNet-vi2 rendition of the network in Fig. 18a.

8 K-Core Representations of the IMDb Co-starring Network Using LaNet-vi2

LaNet-vi2 is a "Large Network" Visualization tool created by collaboration between the University of Indiana, Université Paris-Sud, and the Centre National pour la Recherche Scientifique in France (CNRS, National Center for Scientific Research).

The software is available through a web interface and as a download. Due to the multi-gigabyte size of the files, the web interface did not seem to accept the data. The source code is available at: http://sourceforge.net/projects/lanet-vi/. The visualization method uses k-core decompositions discussed in the next section. The software requires a C compiler and the povray package for ray tracing the network. Povray is available at http://www.povray.org/download/. We suggest running LaNet-vi2 under Linux (Ubuntu) since it does not seem to run properly under Windows (there were a lot of warnings compiling the code).

When running on the complete co-starring network, the software crashed (core dump) after a few hours. This is why we consider two subsets of the network here.

Figures 19 and 20a, b display a k-core representation of the network of the most connected actors and that of the network of minor actors respectively.

Before interpreting Figs. 19 and 20a, b, we recall from Alvarez et al. (2005) and Beiro et al. (2008) some aspects of the algorithms which constructed the graphs.

- The color of the nodes represents their shell index
- The size of the nodes represents their degree in the original network (in logarithmic scale)
- If a k-core is made up of several connected components, each such component is drawn in a separate circle with a diameter proportional to the number of vertices in the component

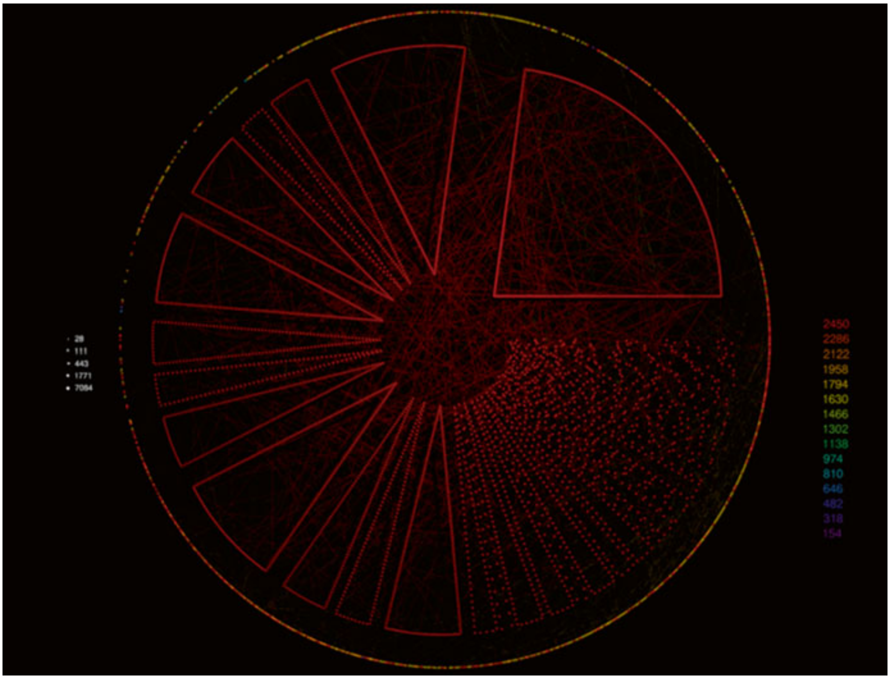


Fig. 19 K-core representation of the network of most connected actors

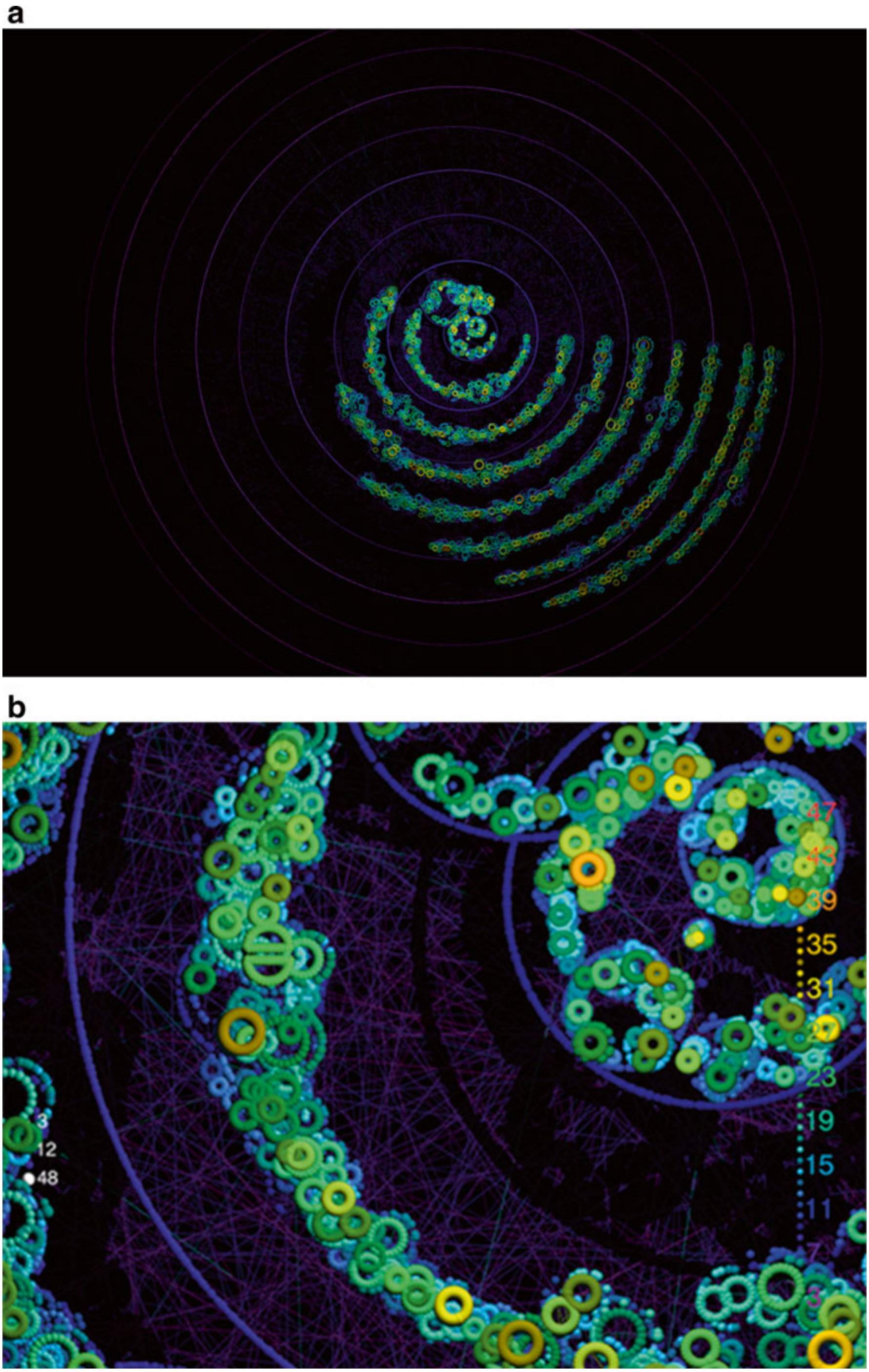


Fig. 20 (**a**) K-core representation of the minor actor network. (**b**) K-core representation of a portion of the minor actor network

- The highest core (the core with the highest value of k) is displayed as a collection of cliques (fully connected sub-graphs of the highest core); each clique is positioned in a separate angular sector with an angle proportional to the number of vertices in the clique; within each clique the vertices are placed uniformly at the periphery of the angular sector
- Cores other than the highest core are represented in equidistant rings, with lower core vertices positioned to their higher core neighbors
- For a given shell diameter, vertices can be placed more internally (if they are connected to vertices with higher shell indices) or externally (if they are connected to vertices with lower shell indices);
- Edges are colored in such a way as to have each extremity colored as the color of the vertex at the opposite end of the edge (see Fig. 18b)

Note that these features imply that "the central cliques organize the deployment of the whole graph" (Beiro et al. 2008)

Figure 19 consists of an outside circle and a central part made up of "slices". The central part is the core with the largest shell index, within which the "slices" are *cliques* (actors are completely connected with each other inside each clique). We can identify about 13 cliques inside the central core.

This central core is very large; smaller components with various shell indices lie on the exterior circle. Note that the maximum degree of 7084 in the legend in Fig. 19 is not as high as 30,000: while the actors in this sub-graph have a degree of at least 30,000 in the full co-starring network, they have a lesser degree in general in the sub-graph.

Next, we examine and discuss a similar representation, but this time for the network of minor actors.

The k-core representation of the network of less connected actors (Fig. 20a, b) is strikingly different from that of the most connected actors (Fig. 20). Numerous circles represent disconnected components within the k-shells. A set of two cliques can be seen as represented by two half slices of a green "pie" in the upper left part of Fig. 20b. Figure 20a displays the whole graph of minor actors and Fig. 20b zooms on a particularly informative portion of the graph. The k-core approach allows for a clearer view (compared to that in Fig. 16) of the many small communities in the network; this is made possible by the successive "peeling" of the k-cores as k increases.

Movie Attendance and Trends

This chapter analyzes longitudinal data on weekly attendance during several years in eight different movie theaters in France, all located in small to medium sized cities in the South West part of France (the movie theaters considered in this study have from 1 to 4 rooms). The data were kindly provided by the French firm Véo http://www.veocine.fr, but the movie theaters were anonymized. In the French cinema industry, weeks start on Wednesday (which is the day of the week when new movies are released) and end on Tuesday. Data were provided in the form of eight separate CSV files as shown on Fig. 1: columns were separated by semi-columns, each column corresponding to a year, and rows contained the number of movie goers for a given week from week 1 of the year (second row) to the last week of the year (last row). Headers contain the years.

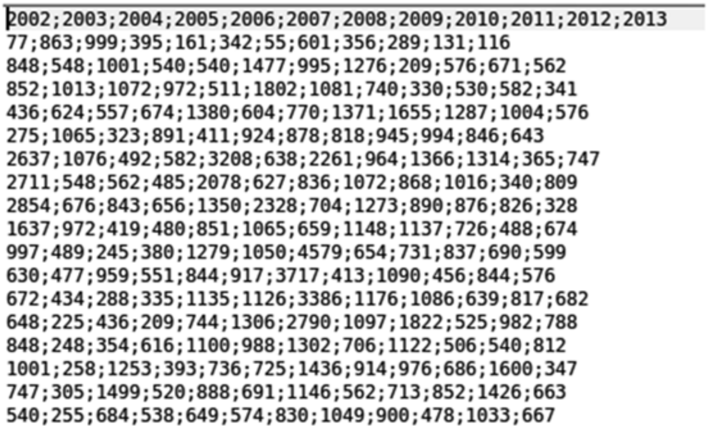
Data were processed using the free statistical software **R**[1] and this section describes how the data can be analyzed step-by-step to extract the main tendency of the time series. This chapter provides some useful commands to manipulate this kind of data and derives insights on particular features that are worth noting.

1 Importing the Data

Using **R**, a list of text files can be retrieved in a given directory (with the function `list.files`) and loaded in the working directory with the function `read.table`. We chose to import the data into a list of eight elements, `all.data`, each element being a `data.frame` (which is one of the fundamental data structures in **R**, used to store several observations, one per row, on several variables, one per column)

[1] http://www.r-project.org

D. Haughton et al., *Movie Analytics*, SpringerBriefs in Statistics,
DOI 10.1007/978-3-319-09426-7_4



```
2002;2003;2004;2005;2006;2007;2008;2009;2010;2011;2012;2013
77;863;999;395;161;342;55;601;356;289;131;116
848;548;1001;540;540;1477;995;1276;209;576;671;562
852;1013;1072;972;511;1802;1081;740;330;530;582;341
436;624;557;674;1380;604;770;1371;1655;1287;1004;576
275;1065;323;891;411;924;878;818;945;994;846;643
2637;1076;492;582;3208;638;2261;964;1366;1314;365;747
2711;548;562;485;2078;627;836;1072;868;1016;340;809
2854;676;843;656;1350;2328;704;1273;890;876;826;328
1637;972;419;480;851;1065;659;1148;1137;726;488;674
997;489;245;380;1279;1050;4579;654;731;837;690;599
630;477;959;551;844;917;3717;413;1090;456;844;576
672;434;288;335;1135;1126;3386;1176;1086;639;817;682
648;225;436;209;744;1306;2790;1097;1822;525;982;788
848;248;354;616;1100;988;1302;706;1122;506;540;812
1001;258;1253;393;736;725;1436;914;976;686;1600;347
747;305;1499;520;888;691;1146;562;713;852;1426;663
540;255;684;538;649;574;830;1049;900;478;1033;667
```

Fig. 1 CSV file containing attendance data for one movie theater

corresponding to a movie theater. For a given movie theater, the rows correspond to the weeks and the columns to the years. Finally, the function `substr` was used to cut the file name so that each element of the list `all.data` was named according to the name of the corresponding CSV file (A.csv was thus stored in the element called “`A`” of the list).

```
# list all files finishing with “.csv” in the directory “csv”
all.files <- list.files("csv", pattern="*.csv")
all.data <- list()
for (ind in seq_along(all.files)) {
  # load every CSV file in one element of the list
  # in CSV file, data are separated by “;”, headers are included in
the file and '1'
  # is used to indicate missing values
 all.data[[ind]] <- read.table(paste0("csv/",all.files[ind]), header=TRUE,
                              sep=";", na.strings="1")
   # remove “X” character that was automatically added to years
(column names)
   colnames(all.data[[ind]]) <- sapply(colnames(all.data[[ind]]),
function(a)
                                      substr(a, 2, nchar(a)))
  # use the CSV file's name to name the current element of the list
    names(all.data)[ind] <- substr(all.files[ind], 1, nchar(all.
files[ind])-4)
}
```

That gives the following imported list (partial listing):

```
> str(all.data)
List of 8
 $ A:'data.frame':      53 obs. of  12 variables:
  ..$ 2002: int [1:53] 77 848 852 436 275 2637 2711 2854 1637 997 …
  ..$ 2003: int [1:53] 863 548 1013 624 1065 1076 548 676 972 489 …
  ..$ 2004: int [1:53] 999 1001 1072 557 323 492 562 843 419 245 …
  ..$ 2005: int [1:53] 395 540 972 674 891 582 485 656 480 380 …
  ..$ 2006: int [1:53] 161 540 511 1380 411 3208 2078 1350 851 1279
…
  ..$ 2007: int [1:53] 342 1477 1802 604 924 638 627 2328 1065 1050
…
  ..$ 2008: int [1:53] 55 995 1081 770 878 2261 836 704 659 4579 …
  ..$ 2009: int [1:53] 601 1276 740 1371 818 964 1072 1273 1148 654
…
  ..$ 2010: int [1:53] 356 209 330 1655 945 1366 868 890 1137 731 …
  ..$ 2011: int [1:53] 289 576 530 1287 994 1314 1016 876 726 837 …
  ..$ 2012: int [1:53] 131 671 582 1004 846 365 340 826 488 690 …
  ..$ 2013: int [1:53] 116 562 341 576 643 747 809 328 674 599 …
 $ B:'data.frame':      53 obs. of  4 variables:
  ..$ 2010: int [1:53] 280 102 165 325 473 591 454 462 576 238 …
  ..$ 2011: int [1:53] 288 302 372 173 705 594 963 283 404 490 …
```

2 Basic Description of the Data

The number of years and weeks available for each movie theater can be checked by applying the functions `ncol` and `nrow` on each element of the list with `lapply`. This provides 53 weeks for all movie theaters (with the first and/or the last week usually not complete weeks) and from 3 to 17 years of data, depending on the movie theater according to Table 1.

The function `is.na` allows for the search of missing values: most years in most theaters had no missing values or, at maximum from 1 to 4 weeks missing, except for year 2008 in movie theater F which had 8 missing weeks in the data. In conclusion, between 0 and 1.5 % of the data were missing depending on the movie theater.

```
# number of columns (years) per movie theater
unlist(lapply(all.data, ncol))
# number of rows (weeks) per movie theater
```

Table 1 Number of years available for each movie theater

Movie theater	A	B	C	D	E	F	G	H
Number of years	12	4	16	3	17	8	17	3

```
unlist(lapply(all.data, nrow))
# number of missing values per movie theater per year
missing.val <- lapply(all.data, function(a.matrix)
                     apply(a.matrix, 2, function(a.column) sum(is.
na(a.column))))
missing.val
# total number of missing values per movie theater
unlist(lapply(missing.val, sum))
# percentage of missing values per movie theater
unlist(lapply(missing.val, sum))/
 unlist(lapply(all.data, function(a.matrix) nrow(a.matrix)*ncol(a.
matrix)))
```

3 Weekly Number of Movie Goers

A first basic comparison of the eight movie theaters can be performed by examining the distribution of the weekly number of movie goers. Using twice the function melt (package reshape), the list can be rearranged into a data frame with the following columns: year, value, and theater in which each row corresponds to a given week in a given year in a given theater and value is the number of movie goers for this week.

```
library(reshape)
# use melt twice to obtain a unique data frame instead of a list
df.all <- lapply(all.data, melt)
df.all <- melt(df.all)
# remove unecessary variable created in the process
df.all$"variable.1" <- NULL
# add names to column
names(df.all) <- c("year", "value", "theater")
# properly order variable year
df.all$year <- ordered(df.all$year, levels=factor(1997:2013))
library(ggplot2)
# weekly attendance distribution per theater
ggplot(df.all, aes(theater, value)) + geom_boxplot() + xlab("movie
theater") +
  ylab("weekly number of movie goers")
# weekly attendance distribution per year and theater
ggplot(df.all, aes(year, value)) + geom_boxplot() + facet_grid
(theater~.) +
  ylab("weekly number of movie goers")
```

This gives the following data frame:

```
> head(df.all)
  year value theater
1 2002    77       A
2 2002   848       A
3 2002   852       A
4 2002   436       A
5 2002   275       A
6 2002  2637       A
```

Then, the ggplot2 package in **R** can be used to display the distribution of the weekly number of movie goers per movie theater (Fig. 2) and the evolution of the weekly number of movie goers per year for each movie theater (Fig. 3). These figures show that the number of movie goers has a skewed distribution for all movie theaters with some weeks having an unexpected high number of movie goers (sometimes more than 4000 movie goers for 1 week). Movie theater G has the largest number of movie goers followed by movie theaters C and A. The evolution through time seems mostly stable, except for movie theater C which shows a visible increase of the number of movie goers after having spent 1 year without spectator (due to a complete renovation of this theater that year). Also, most movie theaters seem to display a slow decrease in attendance during the last years, after some years (until 2007) during which attendance was slowly increasing.

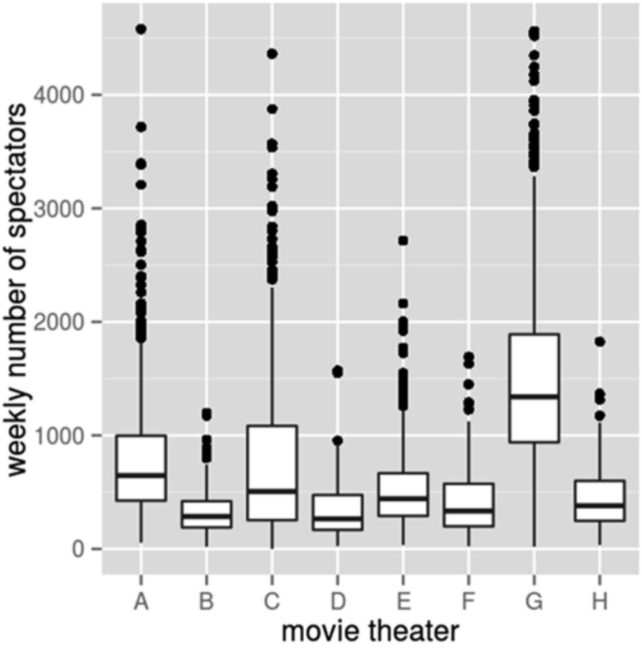


Fig. 2 Weekly attendance distribution per movie theater

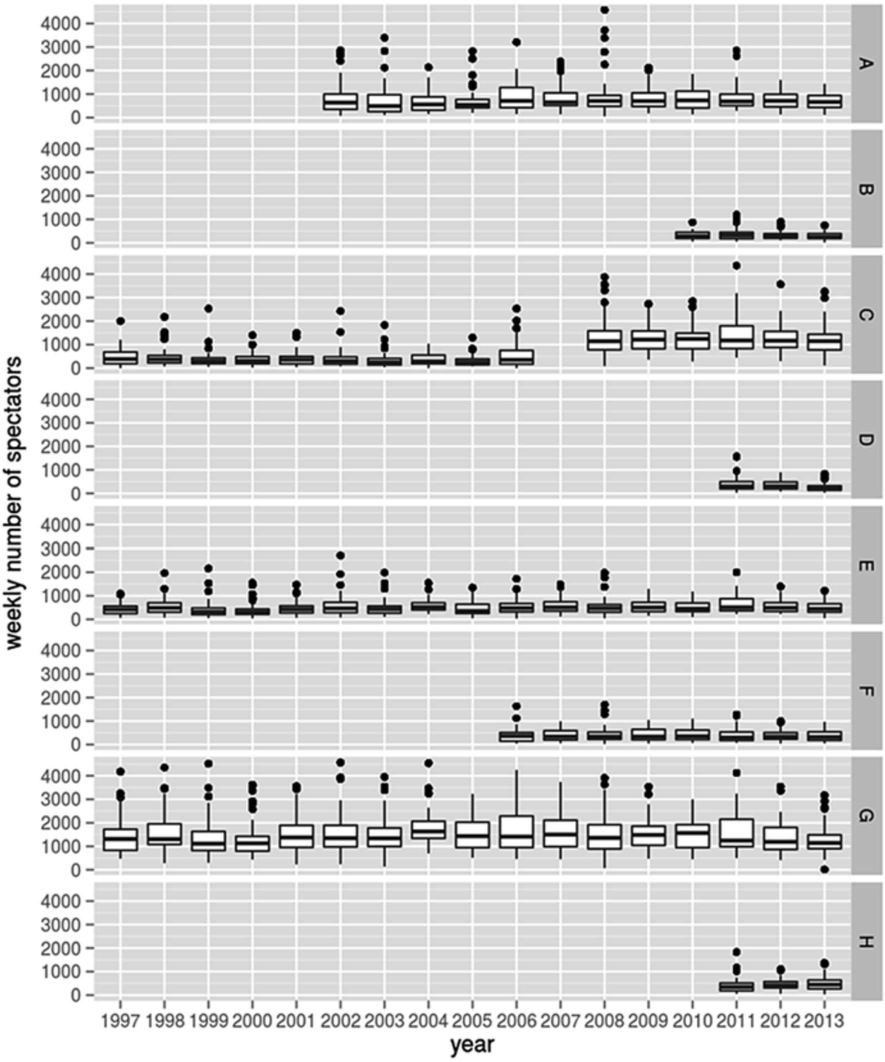


Fig. 3 Weekly attendance distribution per year and theater

4 Evolution of the Weekly Number of Movie Goers

To better visualize the evolution, dates have to be associated to data. To do so, we used the **R** package lubridate which allows for a very easy handling of dates and times. After a number indicating the week of the year has been added to every row in the data frame `df.all`, the functions `ymd` and weeks were used to set up the date of every row: ymd was used to transform the year into a date (the first day of the corresponding year) and weeks to add the proper number of weeks to this starting date.

```
library(lubridate)
library(scales)
# add a number indicating the week of the year (from 1 to 53)
df.all$week <-rep(1:53, nrow(df.all)/53)
# set up dates for every row
df.all$date <- ymd(paste0(as.numeric(as.character(df.all$year)),
"0101")) +
  weeks(df.all$week-1)
# attendance evolution per movie theater
ggplot(df.all, aes(x=date, group=theater, y=value, colour=theater)) +
  geom_line() + xlab("date") + ylab("weekly attendance") +
  scale_x_datetime(breaks = date_breaks("1 year"), labels=date_
format("%Y")) +
  labs(colour="movie theater")
# smoothed attendance evolution per movie theater
span.param <- c(0.05, 0.1, 0.2, 0.5)
p <- list()
for (ind in seq_along(span.param)) {
p[[ind]] <- ggplot(df.all, aes(x=date, group=theater, y=value,
colour=theater)) +
  geom_smooth(span=span.param[[ind]]) + xlab("date") +
  ylab("weekly attendance") + labs(colour="movie theater") +
  ggtitle(paste("smoothing parameter:",span.param[[ind]]))
}
# use the function at
#   http://www.cookbook-r.com/Graphs/Multiple_graphs_on_one_page_
%28ggplot2%29
# to display several ggplot in a single figure
multiplot(plotlist=p, cols=2)
```

The first obtained dates are:

```
> head(df.all$date)
[1] "2002-01-01 UTC" "2002-01-08 UTC" "2002-01-15 UTC" "2002-01-22
UTC" "2002-01-29 UTC" "2002-02-05 UTC"
```

Then, using the **R** package scales for improving the display of dates on the *x* axis, the evolution was drawn with geom_line (package ggplot): the data were grouped by theater, each one associated to a given color as shown in Fig. 4. However, this graph is still not very informative since attendances in movie theaters are highly variable and subject to the release of appealing new movies. Theater G seems to be

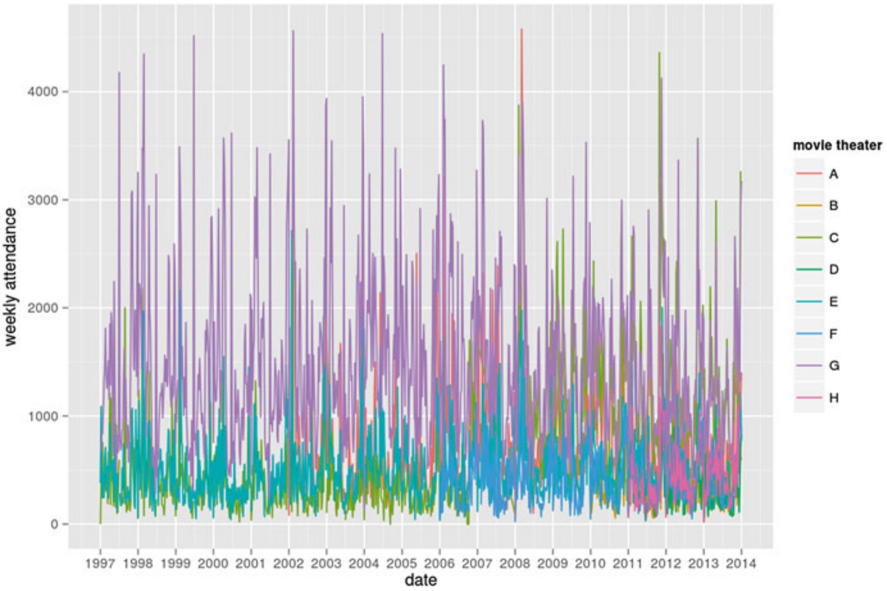


Fig. 4 Evolution of the weekly attendance per movie theater

the theater with the highest number of movie goers but it is also sometimes beaten by theaters A and C.

It is difficult to unravel from Fig. 4 a trend in attendance or to sort the different movie theaters. At this stage, a smoothing is very useful: it can be automatically performed using the function `geom_smooth`, which estimates a LOESS (LOcal regrESSion) non-parametric regression. At each point in the dataset, a 2-degree polynomial is fitted to a subset including a proportion α of the data closest to the estimated point. The polynomial is fitted using weighted least squares, with larger weights for closer points. We use this method to obtain a clearer view of the evolution of weekly attendance in every movie theater, using varying values for the proportion α (the larger α is, the smoother the corresponding estimate is).

The results are shown in Fig. 5. The different parameters provide various insights on the data: first, the smallest values (0.05 and 0.1) clearly emphasize a yearly cycle in the data with a peak at the beginning of the year and the lowest attendance located at approximately the middle of the year. The smoothest values of the parameters enlighten very different behaviors for the eight evolutions: theater G is clearly the one with the best overall results (approximately 1500 movie goers every week) but, after a favorable period (before 2005) during which the number of movie goers was increasing, it has steadily been losing its ability to attract people and has lied below 1400 weekly movie goers in recent years. On the contrary, theater F has a much more steady attendance and seems to even have gained movie goers in recent years. Theater C is a special case since, as we mentioned earlier, it was rebuilt in the middle of the period and displays an interruption in its weekly attendance at that

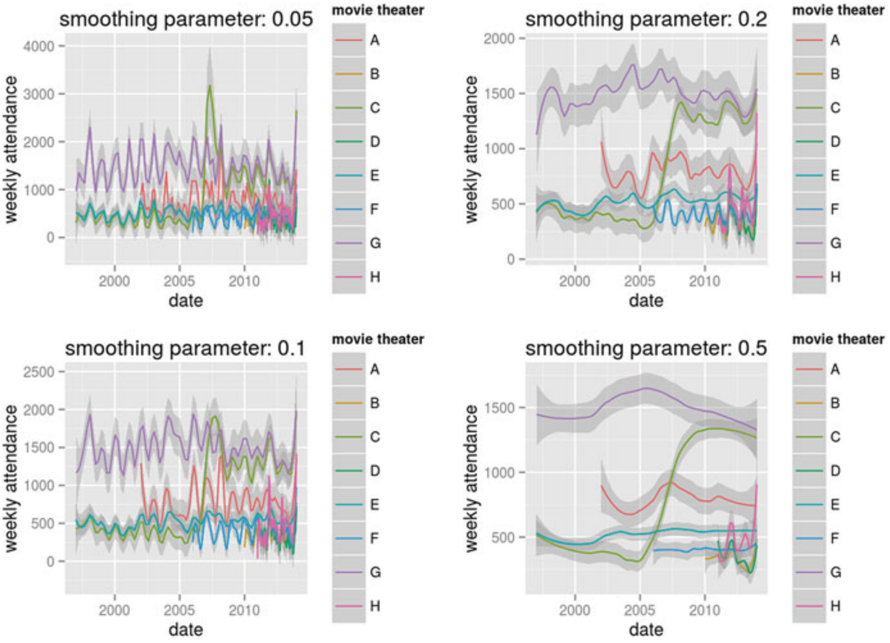


Fig. 5 Smoothed evolution of the weekly attendance per movie theater

time. Since then, its weekly attendance seems to have been slowly decreasing, especially after 2010. Theater A has a rather chaotic attendance, even after smoothing, but also seems to have been losing movie goers in recent years. Finally, the remaining theaters (B, D, F, H, D) were observed on a shorter time period and would thus require a different smoothing parameter to allow for a fair comparison.

5 Trends

This section aims at comparing and describing trends in movie theater attendance in recent years (since 2010). To do so, we restrict our analysis to those movie theaters that were already in use in 2010 (theaters D and H were thus excluded from the analysis) and extract a sub-dataset including only years from 2010 and later. Data were first displayed (using a smoothing approach, as described in the previous section) and a linear model was fitted to the smoothed trend (the one with the largest smoothing parameter, $\alpha = 0.5$) in order to extract the main trend in the evolution. Results are given in Figs. 6 and 7.

Smoothed evolutions present different shapes for the six movie theaters (mostly decreasing for A and approximately stable for B and F for instance) but with a strong border effect: it is hard to say whether the increase observed at the end of the

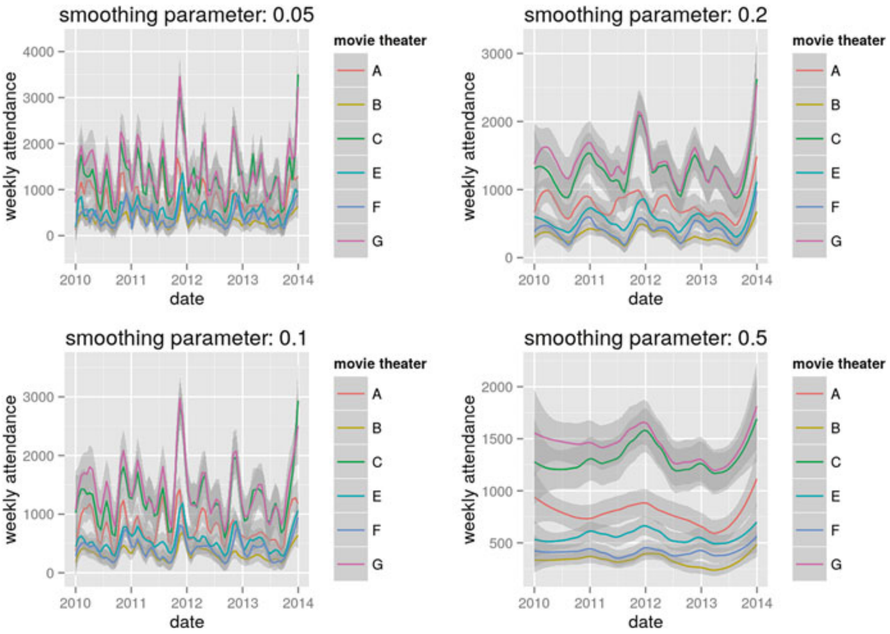


Fig. 6 Weekly attendance for 6 out of 8 movie theaters between 2010 and 2014

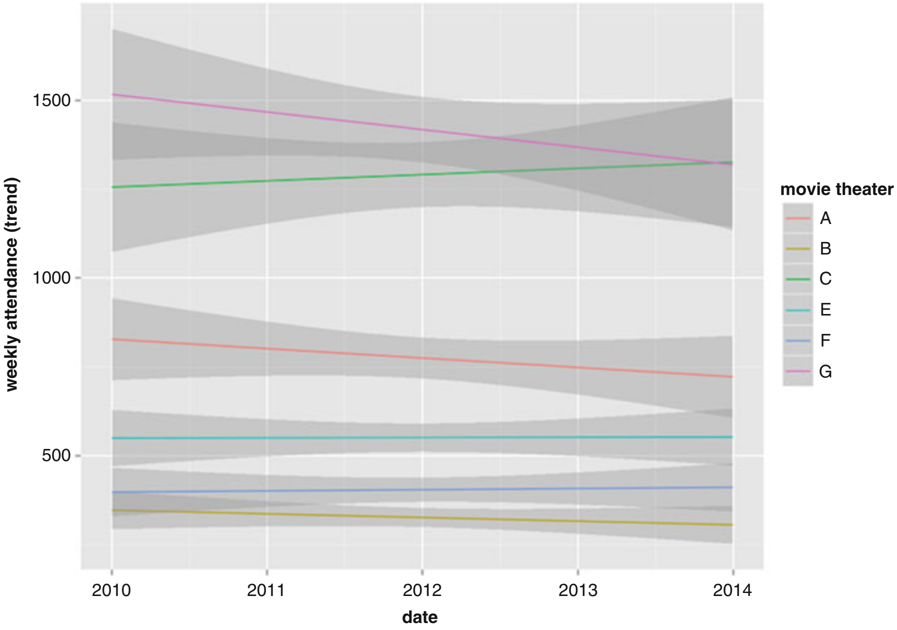


Fig. 7 Linear trends in movie attendance for 6 out of 8 movie theaters between 2010 and 2014

period for all movie theaters is a real trend or an end-year artifact emphasized by the fact that the local averaging uses less data at the boundary of the interval.

Estimated linear trends reveal that movie theater G is the one which has had the strongest decrease during the period (starting from an average weekly attendance equal to 1500 movie goers approximately, it has decreases at the rate of -13.3×10^{-7} movie goers/s). Theater A also has a strong decreasing trend (starting from an average weekly attendance equal to 820 movie goers approximately, it has decreased at the rate of -6.6×10^{-7} movie goers/s), as well as, to a lesser extent, theater B (starting from an average weekly attendance equal to 350 movie goers, it has decreased at the rate of -3.3×10^{-7} movie goers/s). The other three theaters show a global ascending linear trend: the largest increase is for theater C (it started from approximately 1260 movie goers per week and has increased at the rate of 6.8×10^{-7} movie goers/s): at the end of the period, this theater is expected to overcome theater G which was the one with the largest attendance before. Then comes theater F (402 movie goers/week on average at the beginning, rate of increase equal to 1.5×10^{-7} movie goers/s) and finally theater E (560 movie goers/week on average at the beginning, rate of increase equal to 0.3×10^{-7} movie goers/s), which is thus approximately perfectly stable during the later period.

```
# selecting data (6 movie theaters, last 4 years)
df.sub <- df.all[df.all$theater%in%c("A","B","C","E","F","G") &
                   df.all$year%in%as.character(2010:2013),]
# smoothed attendance evolution per movie theater
span.param <- c(0.05, 0.1, 0.2, 0.5)
p <- list()
for (ind in seq_along(span.param)) {
  p[[ind]] <- ggplot(df.sub, aes(x=date, group=theater, y=value,
colour=theater)) +
    geom_smooth(span=span.param[[ind]]) + xlab("date") +
    ylab("weekly attendance") + labs(colour="movie theater") +
    ggtitle(paste("smoothing parameter:",span.param[[ind]]))
}
multiplot(plotlist=p, cols=2)
# fit a linear trend to the smoothed data
model <- NULL
fit <- list()
for (ind in seq_along(unique(df.sub$theater))) {
  print(unique(df.sub$theater)[ind])
  df.sel <- na.omit(df.sub[df.sub$theater==unique(df.sub$theater)
[ind],])
  print(summary(df.sel$date))
  df.sel$smooth <- loess(value~as.numeric(date), data=df.sel,
span=0.5)$fitted
  fit <- lm(smooth ~ date , data=df.sel)
  model <- cbind(model, c(fit$coefficients[2],
                              predict(fit, data.frame(date=ymd("20100
101")))))
}
colnames(model) <- unique(df.sub$theater)
# display the linear trend
ggplot(df.sub, aes(x=date, group=theater, y=value, colour=theater)) +
    geom_smooth(method=lm) + xlab("date") +
        ylab("weekly attendance (trend)") + labs(colour="movie
theater")
> model
                             A             B            C            E
slope                    -6.595601e-07 -3.251354e-07  6.771760e-07
3.168578e-08
starting attendance    8.222594e+02    3.513819e+02  1.264893e+03
5.590331e+02
                                F             G
slope                 1.535095e-07 -1.326393e-06
starting attendance 4.029593e+02  1.515487e+03
```

Data used for this analysis can be downloaded at http://www.nathalievilla.org/movie_analytics with the kind permission from Véo.

Oscar Prediction and Prediction Markets

In this chapter we examine the role of prediction markets in evaluating the probability of a nominated motion picture receiving an Academy award. We illustrate the issue with the best picture award in 2013.

Nine movies were nominated for a Best Picture award at the 2013 Academy Awards. In alphabetical order the movies were:

- Amour
- Argo
- Beasts of the Southern Wild
- Django Unchained
- Les Misérables
- Life of Pi
- Lincoln
- Silver Linings Playbook
- Zero Dark Thirty

Argo won the Best Picture award in 2013.

1 Oscar Prediction and Prediction Markets

The Intrade market (Intrade.com) was an online predictive betting exchange operated by Intrade the Prediction Market Limited. It allowed members to purchase or sell contracts on whether a future event will occur. Popular topics include upcoming elections, movie and music awards, and financial predictions of stock market indexes.

Intrade did not participate in the buying or selling of contracts directly but instead had a flat monthly fee structure for members regardless of the participation level of that member. Trading was done on a per-unit basis with each unit paying $10.00 if the event occurs and $0 if the even does not occur. The contracts traded on a 100

D. Haughton et al., *Movie Analytics*, SpringerBriefs in Statistics,
DOI 10.1007/978-3-319-09426-7_5

point scale with 100 points representing the full $10.00 value. For example, a contract might have stated "Mitt Romney will win the US presidential election in 2012" and the contract might have traded at 25 points. Therefore a member would purchase this contract for the value of $2.50, and if Mitt Romney was elected, then the member would receive $10.00. If Mitt Romney was not elected, then the member would lose the $2.50 to the person who sold the contract.

Intrade received significant media exposure during the 2012 presidential elections with the accurate prediction of nearly all US State electoral contests, but the exposure was overshadowed later in the year with the filing of a civil suit on unregulated trading by the US Commodity Futures Trading Commission. On December 23, 2012, Intrade ceased allowing US members from participating, resulting in a significant drop in overall participation and on March 10, 2013 Intrade ceased all trading. The prediction market PaddyPower (www.paddypower.com) typically hosts best picture bets in Academy Award competitions, as does Bet Victor (www.betvistor.com). But unlike for Intrade, there is no convenient way to get historical pricing information out of Paddy Power or Bet Victor and rapid changes in pricing may be difficult to track without employing some form of screen scraping. Researchers in the USA may also be shut out from even looking at international betting sites (such as www.WilliamHill.com for example).

Scholars have used Intrade data to investigate issues such as participation in the Euro currency (Shambaugh et al. 2012), the probability of a US recession (Leamer 2008), elections (Saxon 2010; Rothschild and Wolfers 2008; Erikson and Wlezien 2008), and entertainment awards such as the Grammy Awards and Oscars (Gold et al. 2013). Prediction market estimates of the probability of a win are considered to be very

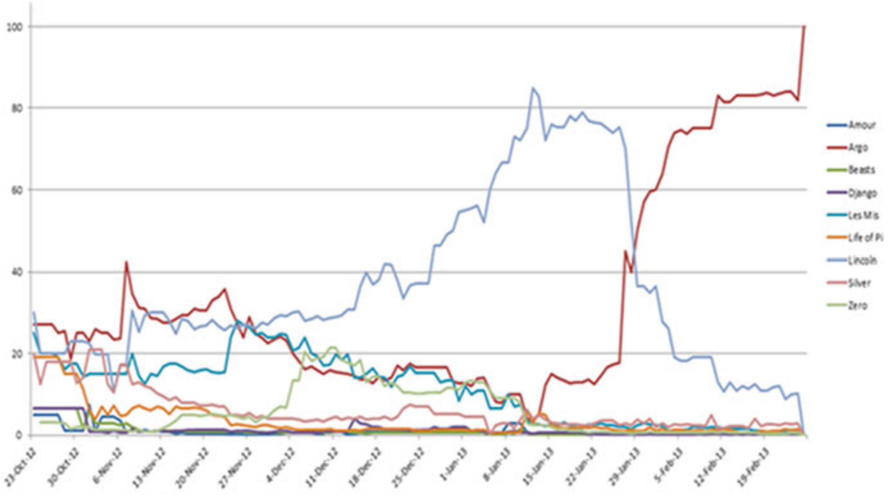


Fig. 1 Close Intrade contract prices for each nominated movie to win the 2013 Best Picture award

accurate, at least for events such as Oscar wins. Prediction markets were reportedly successful again in predicting the 2015 Academy Awards (Leonhardt 2015).

Figure 1 displays the price per contract for each of the nominees winning the Oscar for the 2013 Best Picture award. We can see that up until December there was not a clear front runner. Then beginning in December the film Lincoln emerged as the clear favorite. However, in late January the film Argo began to gain on Lincoln, surpassing Lincoln and holding onto that position until the end.

In analyzing the average contract price for each movie, we see that the five serious contenders for the Best Picture award were in alphabetical order Argo, Les Misérables, Lincoln, Silver Linings Playbook, and Zero Dark Thirty.

What happened over this time period that could have contributed to the perceptions of which film would win the award? Specifically, what occurred around the timeframe of late January that caused such a dramatic change?

On January 26th, 2013, a Los Angeles Times headline read "The Gold Standard; now for real insight into Oscars—by the guilds." The article reported that the guilds' awards, beginning with PGAs, had been fairly reliable predictors (Whipp 2013).

The "PGAs" denote the Producer Guild of America Awards which were announced that evening. The headline coming out of the PGAs that evening was that Argo won the top prize of the night, The Zanuck Award for Outstanding Producer of Theatrical Motion Pictures.

This awards ceremony was followed the next evening with the 19th Annual Screen Actors Guild (SAG) awards. Their top award is the Outstanding Performance by a Cast in a Motion Picture which was awarded to Argo.

So can we simply use those two awards ceremonies to predict the Oscar's Best Picture award? Although we focused only on the 2012 movies, we can take a quick look at the winners over the past decade. Over those 10 years the PGA and SAG have awarded to the same picture five times, and four of those times the Oscars have followed suit and awarded to the same film. In the other 5 years where the PGA and SAG have awarded different films, the Oscars selected one of the two 4 of those 5 years. Only in 2004 did the PGA, SAG, and the Oscars each give the top award to a different film (Table 1).

Table 1 PGA awards, SAG awards, and Oscars

	PGA	SAG	Oscars
2012	Argo	Argo	Argo
2011	The Artist	The Help	The Artist
2010	The King's Speech	The King's Speech	The King's Speech
2009	The Hurt Locker	Inglourious Basterds	The Hurt Locker
2008	Slumdog Millionaire	Slumdog Millionaire	Slumdog Millionaire
2007	No Country for Old Men	No Country for Old Men	No Country for Old Men
2006	Little Miss Sunshine	Little Miss Sunshine	The Departed
2005	Brokeback Mountain	Crash	Crash
2004	The Aviator	Sideways	Million Dollar Baby

Can We Predict Oscars from Twitter and Movie Review Data?

In this chapter, we focus the attention on whether text reviews of movies which are nominated for a Best Picture award carry any sign of the likelihood of a movie winning the award. We suggest that a measure of how controversial the movie is perceived to be, the value of which could be extracted by a text analysis of the reviews, is a potential predictor of a win, aside from other predictors identified in past work.This also is an opportunity to discuss text mining and sentiment analysis techniques.

1 Text Mining

1.1 Twitter Movie Related Data

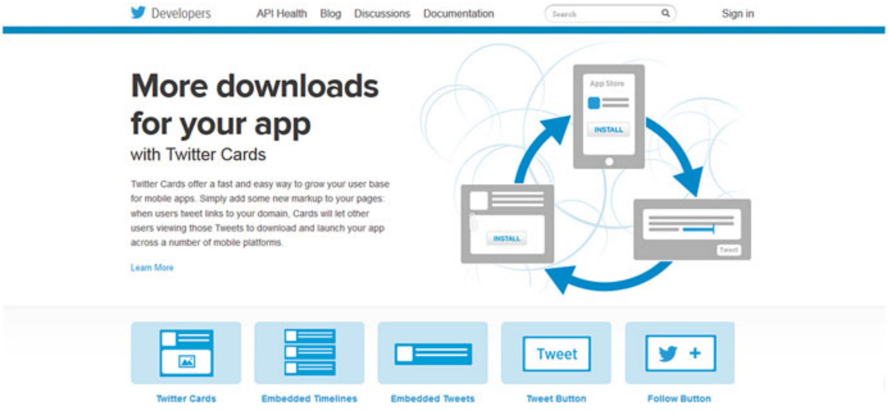
1.1.1 Twitter Streaming API and Developer Account

There are many ways to programmatically obtain and analyze tweets. It is good to start with using the Twitter Streaming API. The free version of the Twitter Streaming API continuously pushes back a small percentage of all the tweets from the time we send it the request. Before running the crawling code, one needs to go to http://dev.twitter.com (Fig. 1), and create a new app. The consumer key and secret key will be generated for the user, who then needs to visit "Your access token" to create an access token.

1.1.2 Python and "Tweepy" Library

In this monograph, we use Python to handle the process of crawling, transforming and loading of tweets. Many Python libraries interface with the Twitter Streaming API, such as tweepy (https://github.com/tweepy/tweepy), Twitter (https://pypi.python.org/pypi/twitter), python-twitter (https://github.com/bear/python-twitter),

D. Haughton et al., *Movie Analytics*, SpringerBriefs in Statistics,
DOI 10.1007/978-3-319-09426-7_6

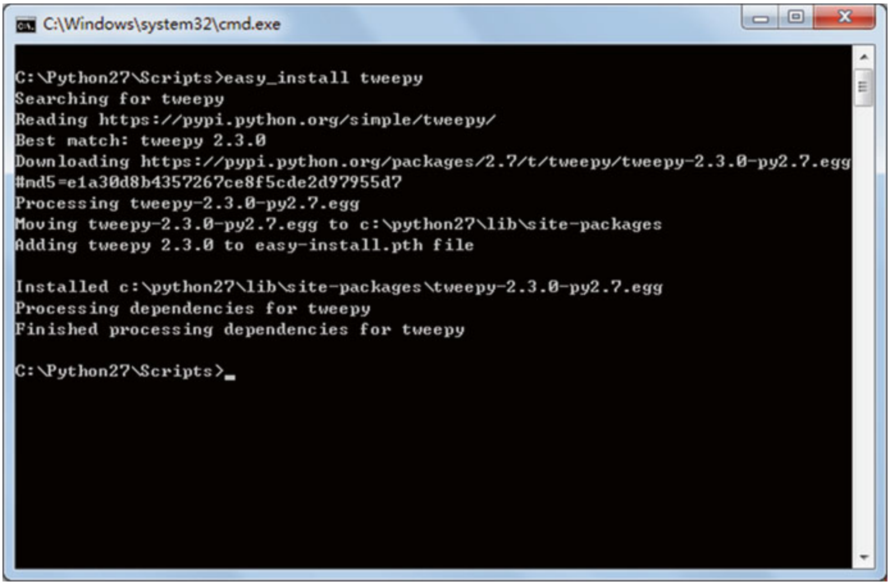


More downloads for your app

with Twitter Cards

Twitter Cards offer a fast and easy way to grow your user base for mobile apps. Simply add some new markup to your pages: when users tweet links to your domain, Cards will let other users viewing those Tweets to download and launch your app across a number of mobile platforms.

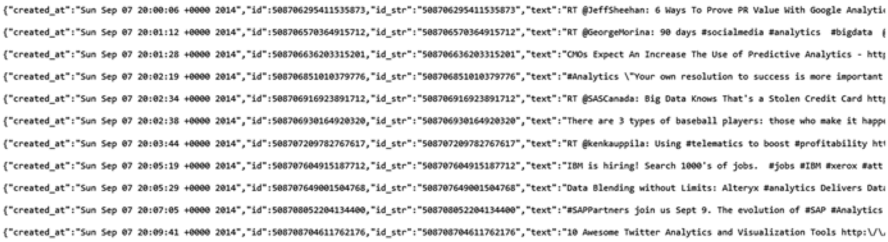
Fig. 1 Twitter developer webpage



```
C:\Python27\Scripts>easy_install tweepy
Searching for tweepy
Reading https://pypi.python.org/simple/tweepy/
Best match: tweepy 2.3.0
Downloading https://pypi.python.org/packages/2.7/t/tweepy/tweepy-2.3.0-py2.7.egg
#md5=e1a30d8b4357267ce8f5cde2d97955d7
Processing tweepy-2.3.0-py2.7.egg
Moving tweepy-2.3.0-py2.7.egg to c:\python27\lib\site-packages
Adding tweepy 2.3.0 to easy-install.pth file

Installed c:\python27\lib\site-packages\tweepy-2.3.0-py2.7.egg
Processing dependencies for tweepy
Finished processing dependencies for tweepy

C:\Python27\Scripts>_
```

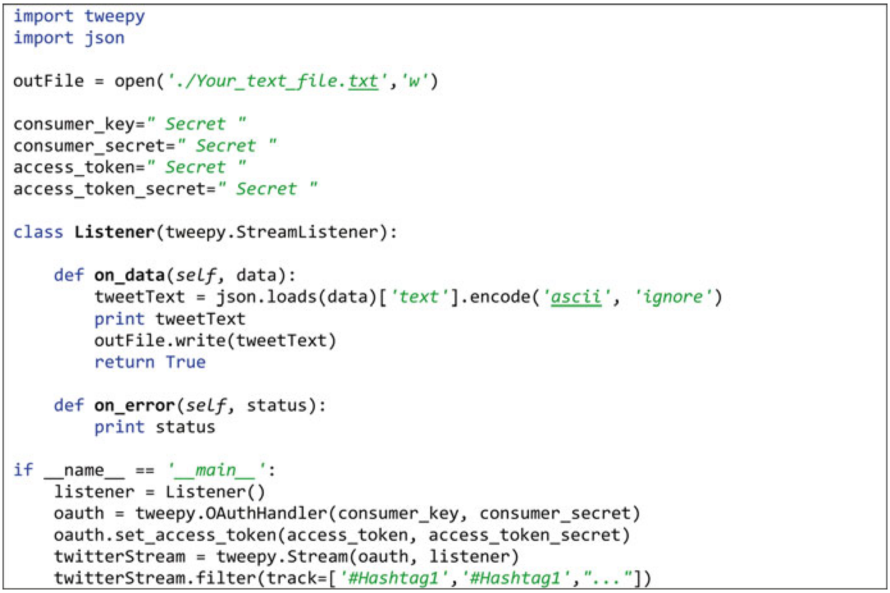
Fig. 2 Installation of the Python twitter package

and TwitterSearch (https://pypi.python.org/pypi/TwitterSearch/). We can choose any of those, but need to remember to check the developer's website for possible library updates. Here we use the Python library tweepy as an example. For the Windows version of Python 2.6, python-twitter can be installed by using "Your Python Path\Scripts\easy_install tweepy" under the Windows command prompt cmd.exe. Once the installation is successful, information such as in Fig. 2 appears. For data storing and loading, Python itself can handle JSON (JavaScript Object Notation) style tweets using the "json" library. The raw JSON style tweet records



```
{"created_at":"Sun Sep 07 20:00:06 +0000 2014","id":508706295411535873,"id_str":"508706295411535873","text":"RT @JeffSheehan: 6 Ways To Prove PR Value With Google Analyti
{"created_at":"Sun Sep 07 20:01:12 +0000 2014","id":508706570364915712,"id_str":"508706570364915712","text":"RT @GeorgeMorina: 90 days #socialmedia #analytics  #bigdata
{"created_at":"Sun Sep 07 20:01:28 +0000 2014","id":508706636203315201,"id_str":"508706636203315201","text":"CMOs Expect An Increase The Use of Predictive Analytics - htt
{"created_at":"Sun Sep 07 20:02:19 +0000 2014","id":508706851010379776,"id_str":"508706851010379776","text":"#Analytics \"Your own resolution to success is more important
{"created_at":"Sun Sep 07 20:02:34 +0000 2014","id":508706916923891712,"id_str":"508706916923891712","text":"RT @SASCanada: Big Data Knows That's a Stolen Credit Card htt
{"created_at":"Sun Sep 07 20:02:38 +0000 2014","id":508706930164920320,"id_str":"508706930164920320","text":"There are 3 types of baseball players: those who make it happ
{"created_at":"Sun Sep 07 20:03:44 +0000 2014","id":508707209782767617,"id_str":"508707209782767617","text":"RT @kenkauppila: Using #telematics to boost #profitability ht
{"created_at":"Sun Sep 07 20:05:19 +0000 2014","id":508707604915187712,"id_str":"508707604915187712","text":"IBM is hiring! Search 1000's of jobs.  #jobs #IBM #xerox #att
{"created_at":"Sun Sep 07 20:05:29 +0000 2014","id":508707649001504768,"id_str":"508707649001504768","text":"Data Blending without Limits: Alteryx #analytics Delivers Dat
{"created_at":"Sun Sep 07 20:07:05 +0000 2014","id":508708052204134400,"id_str":"508708052204134400","text":"#SAPPartners join us Sept 9. The evolution of #SAP #Analytics
{"created_at":"Sun Sep 07 20:09:41 +0000 2014","id":508708704611762176,"id_str":"508708704611762176","text":"10 Awesome Twitter Analytics and Visualization Tools http:\/\
```

Fig. 3 Sample tweets



```python
import tweepy
import json

outFile = open('./Your_text_file.txt','w')

consumer_key=" Secret "
consumer_secret=" Secret "
access_token=" Secret "
access_token_secret=" Secret "

class Listener(tweepy.StreamListener):

    def on_data(self, data):
        tweetText = json.loads(data)['text'].encode('ascii', 'ignore')
        print tweetText
        outFile.write(tweetText)
        return True

    def on_error(self, status):
        print status

if __name__ == '__main__':
    listener = Listener()
    oauth = tweepy.OAuthHandler(consumer_key, consumer_secret)
    oauth.set_access_token(access_token, access_token_secret)
    twitterStream = tweepy.Stream(oauth, listener)
    twitterStream.filter(track=['#Hashtag1','#Hashtag1',"..."])
```

Fig. 4 Python code for extracting tweets

are in a format such as displayed in Fig. 3. For more sophisticated ways to manage the tweets, the light and free MongoDB option is always possible (www.mongodb.org), and is easily driven by Python. Here we use the simplest way to decode JSON style tweets and save the text of tweets into plain text.

1.1.3 Python Code

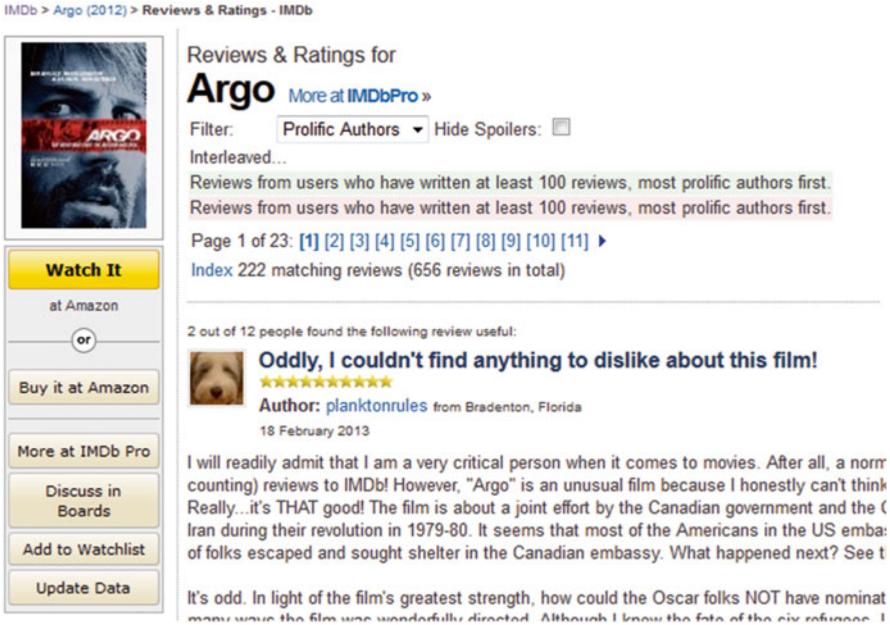
The Python code in Fig. 4 can be run to extract the text part of tweets, which is stored as plain text. The code keeps downloading tweets and parsing the text part, storing the text into the file, and refreshing it. It is important to note that the process does not stop until terminated manually.

To run the code, the values of consumer_key, consumer_secret, access_token, and access_token_secret must be replaced by the user's. The #Hashtag should also be replaced with the hashtag of interest.

1.2 *IMDb Review Data*

1.2.1 IMDb Review

In terms of text mining the opinions of movie watchers, IMDb user reviews have several advantages compared to tweets. First, most user reviews on IMDb are much longer than tweets (which are constrained to a maximum of 140 characters). Therefore, a review can contain richer and more complex thoughts than a tweet. Second, some review writers on IMDb are prolific authors, while the quality of tweets is not guaranteed at all. One can filter out reviews by non-prolific authors by choosing the "Prolific Author" filter on the IMDb review page (Fig. 5). Third, IMDb review readers can vote up or down to a review, as in "2 out of 12 people found the following review useful" in the middle part of Fig. 5. We can use it to measure the quality of a review. However, IMDb does not provide any API or structured database for downloading movie reviews. Therefore, we need to crawl the raw HTML webpage to extract review data.



Reviews & Ratings for

Argo

Filter: Prolific Authors Hide Spoilers: □

Interleaved...

Reviews from users who have written at least 100 reviews, most prolific authors first.

Reviews from users who have written at least 100 reviews, most prolific authors first.

Page 1 of 23: [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] ▸

Index 222 matching reviews (656 reviews in total)

2 out of 12 people found the following review useful:

Oddly, I couldn't find anything to dislike about this film!

★★★★★★★★★★

Author: planktonrules from Bradenton, Florida

18 February 2013

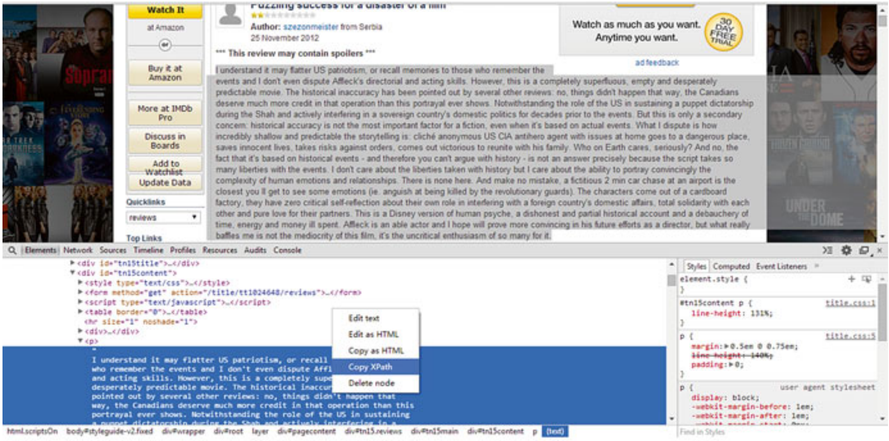
I will readily admit that I am a very critical person when it comes to movies. After all, a norm
counting) reviews to IMDb! However, "Argo" is an unusual film because I honestly can't think
Really...it's THAT good! The film is about a joint effort by the Canadian government and the (
Iran during their revolution in 1979-80. It seems that most of the Americans in the US emba:
of folks escaped and sought shelter in the Canadian embassy. What happened next? See t

It's odd. In light of the film's greatest strength, how could the Oscar folks NOT have nominat
many ways the film was wonderfully directed. Although I know the fate of the six refugees, I

Fig. 5 "Argo" IMDb reviews including prolific authors only

Table 1 Path expressions for XPath, from http://www.w3schools.com/xpath/xpath_syntax.asp

Expression	Description
nodename	Selects all nodes with the name "*nodename*"
/	Selects from the root node
//	Selects nodes in the document from the current node that match the selection no matter where they are
.	Selects the current node
..	Selects the parent of the current node
@	Selects attributes



Puzzling success for a disaster of a film

Author: szezonmeister from Serbia
25 November 2012

*** This review may contain spoilers ***

I understand it may flatter US patriotism, or recall memories to those who remember the events and I don't even dispute Affleck's directorial and acting skills. However, this is a completely superfluous, empty and desperately predictable movie. The historical inaccuracy has been pointed out by several other reviews: no, things didn't happen that way, the Canadians deserve much more credit in that operation than this portrayal ever shows. Notwithstanding the role of the US in sustaining a puppet dictatorship during the Shah and actively interfering in a sovereign country's domestic politics for decades prior to the events. But this is only a secondary concern: historical accuracy is not the most important factor for a fiction, even when it's based on actual events. What I dispute is how incredibly shallow and predictable the storytelling is: cliché anonymous US CIA antihero agent with issues at home goes to a dangerous place, saves innocent lives, takes risks against orders, comes out victorious to reunite with his family. Who on Earth cares, seriously? And no, the fact that it's based on historical events - and therefore you can't argue with history - is not an answer precisely because the script takes so many liberties with the events. I don't care about the liberties taken with history but I care about the ability to portray convincingly the complexity of human emotions and relationships. There is none here. And make no mistake, a fictitious 2 min car chase at an airport is the closest you ll get to see some emotions (ie. anguish at being killed by the revolutionary guards). The characters come out of a cardboard factory, they have zero critical self-reflection about their own role in interfering with a foreign country's domestic affairs, total solidarity with each other and pure love for their partners. This is a Disney version of human psyche, a dishonest and partial historical account and a debauchery of time, energy and money ill spent. Affleck is an able actor and I hope will prove more convincing in his future efforts as a director, but what really baffles me is not the mediocrity of this film, it's the uncritical enthusiasm of so many for it.

```
▶<div id="tn15title">…</div>
▼<div id="tn15content">
  ▶<style type="text/css">…</style>
  ▶<form method="get" action="/title/tt1024648/reviews">…</form>
  ▶<script type="text/javascript">…</script>
  ▶<table border="0">…</table>
   <hr size="1" noshade="1">
  ▶<div>…</div>
  ▼<p>
     I understand it may flatter US patriotism, or recall
     who remember the events and I don't even dispute Affl
     and acting skills. However, this is a completely supe
     desperately predictable movie. The historical inaccur
     pointed out by several other reviews: no, things didn't happen that
     way, the Canadians deserve much more credit in that operation than this
     portrayal ever shows. Notwithstanding the role of the US in sustaining
```

```
html.scriptsOn  body#styleguide-v2.fixed  div#wrapper  div#root  layer  div#pagecontent  div#tn15.reviews  div#tn15main  div#tn15content  p  (text)
```

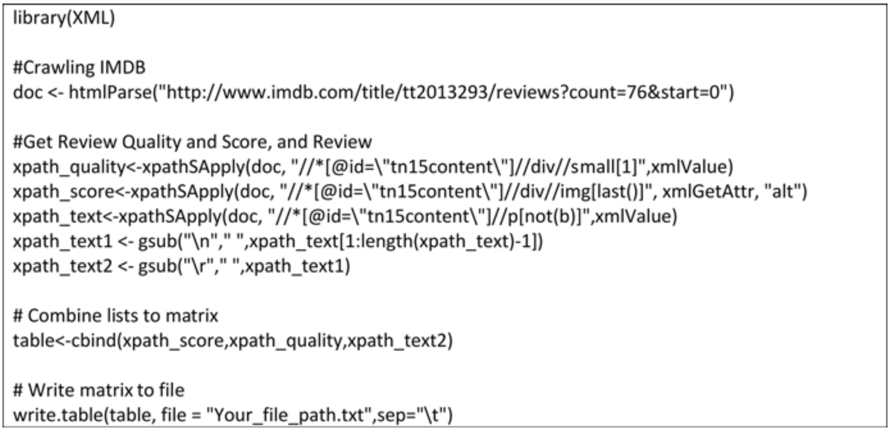
```
element.style {
}
#tn15content p {                     title.css:1
   line-height: 131%;
}
p {                                  title.css:5
   margin:▶0.5em 0 0.75em;
   ~~line-height: 140%;~~
   padding:▶0;
}
p {                    user agent stylesheet
   display: block;
   -webkit-margin-before: 1em;
   -webkit-margin-after: 1em;
```

Fig. 6 Obtaining the XPath of review text using Google Chrome

1.2.2 XPath and R XML Library

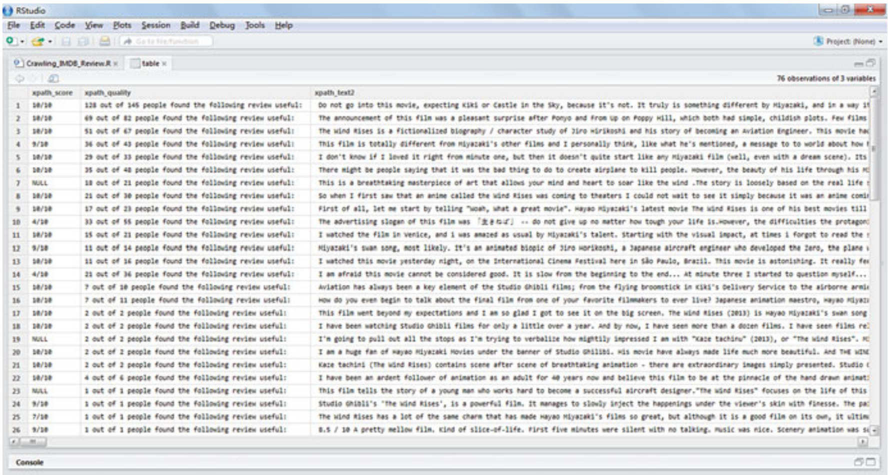
In this chapter, XPath is used to mechanically navigate through elements and attributes in an XML document, such as all IMDb reviews on one webpage. It is easy to read, easy to reuse, and supported by most programming languages and software packages such as Python or R. XPath expressions such as in Table 1 are not difficult to learn. Handy XPath tutorials are available at http://www.w3schools.com/xpath/.

We can get the XPath of the part of a webpage we are interested in by simply using the Google Chrome web browser. First we open an IMDb review webpage in Chrome, choose the part we want to crawl, right click on it, and from the pop-up menu select "Inspect Element." Once that option is selected, we see two windows on the bottom side of the browser, and the part chosen earlier is highlighted in the left-down side HTML code area. We right click on the highlighted area, and select "Copy XPath" from the pop-up menu and then obtain the raw XPath expression for the IMDb review, such as for example "//*[@id="tn15content"]/p[1]/text()" (Fig. 6).



```
library(XML)

#Crawling IMDB
doc <- htmlParse("http://www.imdb.com/title/tt2013293/reviews?count=76&start=0")

#Get Review Quality and Score, and Review
xpath_quality<-xpathSApply(doc, "//*[@id=\"tn15content\"]//div//small[1]",xmlValue)
xpath_score<-xpathSApply(doc, "//*[@id=\"tn15content\"]//div//img[last()]", xmlGetAttr, "alt")
xpath_text<-xpathSApply(doc, "//*[@id=\"tn15content\"]//p[not(b)]",xmlValue)
xpath_text1 <- gsub("\n"," ",xpath_text[1:length(xpath_text)-1])
xpath_text2 <- gsub("\r"," ",xpath_text1)

# Combine lists to matrix
table<-cbind(xpath_score,xpath_quality,xpath_text2)

# Write matrix to file
write.table(table, file = "Your_file_path.txt",sep="\t")
```

Fig. 7 R code to extract IMDb reviews



76 observations of 3 variables

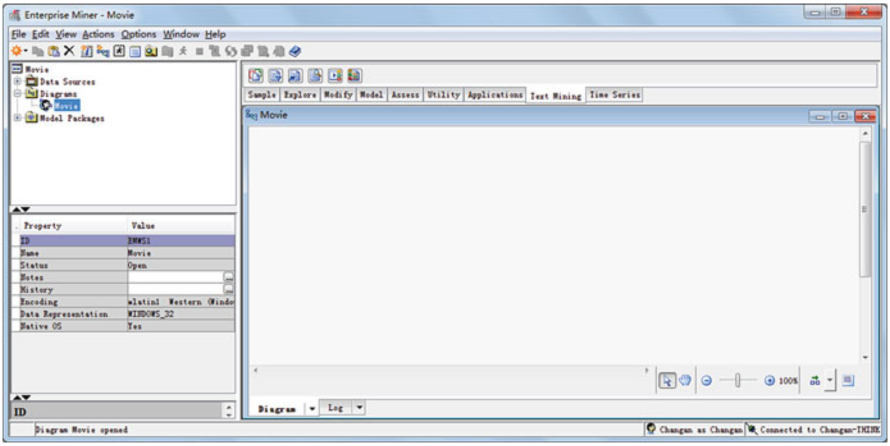
	xpath_score	xpath_quality	xpath_text2
1	10/10	128 out of 145 people found the following review useful:	Do not go into this movie, expecting Kiki or Castle in the Sky, because it's not. It truly is something different by Miyazaki, and in a way it
2	10/10	69 out of 82 people found the following review useful:	The announcement of this film was a pleasant surprise after Ponyo and From up on Poppy Hill, which both had simple, childish plots. Few films
3	10/10	51 out of 67 people found the following review useful:	The Wind Rises is a fictionalized biography / character study of Jiro Horikoshi and his story of becoming an Aviation Engineer. This movie ha
4	9/10	36 out of 43 people found the following review useful:	This film is totally different from Miyazaki's other films and I personally think, like what he's mentioned, a message to to world about how
5	10/10	29 out of 33 people found the following review useful:	I don't know if I loved it right from minute one, but then it doesn't quite start like any Miyazaki film (well, even with a dream scene). Its
6	10/10	35 out of 48 people found the following review useful:	There might be people saying that it was the bad thing to do to create airplane to kill people. However, the beauty of his life through his M
7	NULL	18 out of 21 people found the following review useful:	This is a breathtaking masterpiece of art that allows your mind and heart to soar like the wind .The story is loosely based on the real life
8	10/10	21 out of 30 people found the following review useful:	So when I first saw that an anime called the Wind Rises was coming to theaters I could not wait to see it simply because it was an anime comi
9	10/10	17 out of 23 people found the following review useful:	First of all, let me start by telling "Woah, what a great movie". Hayao Miyazaki's latest movie The Wind Rises is one of his best movies till
10	4/10	33 out of 55 people found the following review useful:	The advertising slogan of this film was 「生きねば」 -- do not give up no matter how tough your life is.However, the difficulties the protagon
11	10/10	15 out of 21 people found the following review useful:	I watched the film in Venice, and i was amazed as usual by Miyazaki's talent. Starting with the visual impact, at times i forgot to read the
12	9/10	11 out of 14 people found the following review useful:	Miyazaki's swan song, most likely. It's an animated biopic of Jiro Horikoshi, a Japanese aircraft engineer who developed the Zero, the plane
13	10/10	11 out of 16 people found the following review useful:	I watched this movie yesterday night, on the International Cinema Festival here in São Paulo, Brazil. This movie is astonishing. It really fe
14	4/10	21 out of 36 people found the following review useful:	I am afraid this movie cannot be considered good. It is slow from the beginning to the end... At minute three I started to question myself...
15	10/10	7 out of 10 people found the following review useful:	Aviation has always been a key element of the Studio Ghibli films; from the flying broomstick in Kiki's Delivery Service to the airborne armi
16	10/10	7 out of 11 people found the following review useful:	How do you even begin to talk about the final film from one of your favorite filmmakers to ever live? Japanese animation maestro, Hayao Miyaz
17	10/10	2 out of 2 people found the following review useful:	This film went beyond my expectations and I am so glad I got to see it on the big screen. The Wind Rises (2013) is Hayao Miyazaki's swan song
18	10/10	2 out of 2 people found the following review useful:	I have been watching Studio Ghibli films for only a little over a year. And by now, I have seen more than a dozen films. I have seen films re
19	NULL	2 out of 2 people found the following review useful:	I'm going to pull out all the stops as I'm trying to verbalize how mightily impressed I am with "Kaze tachinu" (2013), or "The Wind Rises". M
20	10/10	2 out of 2 people found the following review useful:	I am a huge fan of Hayao Miyazaki Movies under the banner of Studio Ghilibi. His movie have always made life much more beautiful. And THE WIN
21	10/10	2 out of 2 people found the following review useful:	Kaze tachini (The Wind Rises) contains scene after scene of breathtaking animation - there are extraordinary images simply presented. Studio
22	10/10	4 out of 6 people found the following review useful:	I have been an ardent follower of animation as an adult for 40 years now and believe this film to be at the pinnacle of the hand drawn animat
23	NULL	1 out of 1 people found the following review useful:	This film tells the story of a young man who works hard to become a successful aircraft designer."The Wind Rises" focuses on the life of this
24	9/10	1 out of 1 people found the following review useful:	Studio Ghibli's 'The Wind Rises', is a powerful film. It manages to slowly inject the happenings under the viewer's skin with finesse. The pa
25	7/10	1 out of 1 people found the following review useful:	The Wind Rises has a lot of the same charm that has made Hayao Miyazaki's films so great, but although it is a good film on its own, it ultim
26	9/10	1 out of 1 people found the following review useful:	8.5 / 10 A pretty mellow film. Kind of slice-of-life. First five minutes were silent with no talking. Music was nice. Scenery animation was s

Fig. 8 Extracted IMDb reviews

In this monograph, we use R to handle the process of crawling, transforming and loading IMDb reviews. To handle the XPath in R, we need to first install the "XML" package. After installation, we can run the R code in Fig. 7 to crawl and parse movie reviews. The result looks like that in Fig. 8.

1.2.3 Text Mining Using SAS Enterprise Miner

In the next step, we handle the textual dataset using SAS Text Miner, which is a plug-in for the SAS Enterprise Miner environment. The Enterprise Miner interface is displayed in Fig. 9, after we have created a New Project and New Diagram. We can then create a SAS data set from the tweets or IMDb review documents, using


Enterprise Miner - Movie
File Edit View Actions Options Window Help
Movie
Data Sources
Diagrams
Movie
Model Packages
Sample Explore Modify Model Assess Utility Applications Text Mining Time Series
Movie
100%
Diagram Log
Changan as Changan
Connected to Changan-THINK

Property	Value
ID	EMWS1
Name	Movie
Status	Open
Notes	
History	
Encoding	wlatin1 Western (Windo
Data Representation	WINDOWS_32
Native OS	Yes

Diagram Movie opened

Fig. 9 SAS Enterprise Miner Diagram

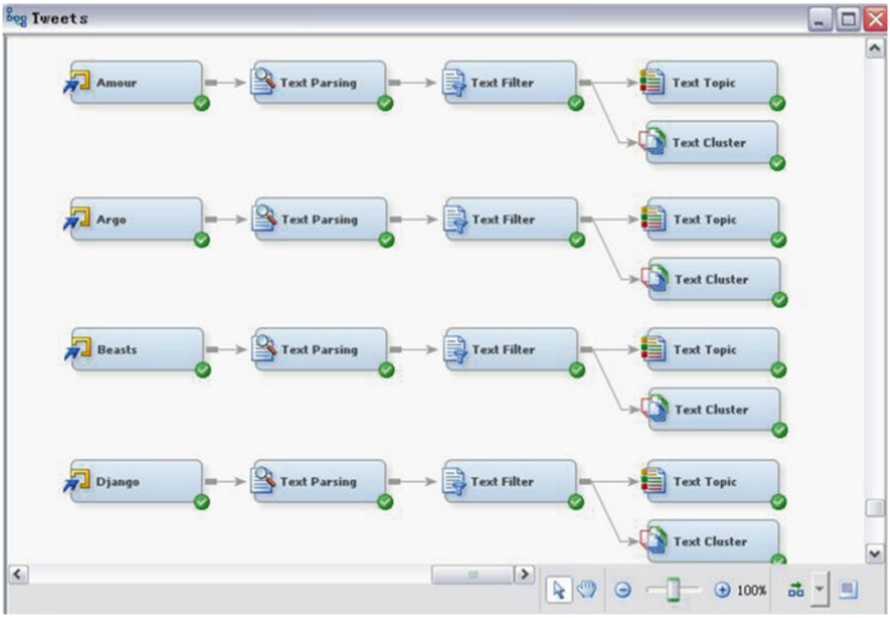


Fig. 10 SAS Enterprise Miner diagram for movie tweets

the Text Import node Text Import, or File Import node File Import. The Text Parsing node Text Parsing in SAS Text Miner decomposes the documents into detailed terms or phrases, and the Text Filter node Text Filter automatically detects misspelling in the data, and transforms the quantitative representation into a compact and informative format. The Text Cluster node Text Cluster clusters documents into disjoint sets of documents, and the Text Topic node Text Topic creates topics for each document, where one document can be associated with more than one topic (Fig. 10).

1.2.4 Review Themes and Oscar Chances

In this section we discuss how a text mining of the IMDb pre-Oscars reviews gives an idea of the numbers of different themes which are perceived by reviewers for each movie, and potentially yields a preliminary measure of perceived controversy. The question is then of how much "controversy" is optimal for Oscar winning purposes.

Measures of controversy and how they are used in marketing are discussed in Zhang and Li (2010). A quote from this article is very pertinent to our discussion: "From a persuasion point of view, our belief is that a convincing argument is not necessarily a mono-color picture, but instead a meaningful "bag" of positive and negative reflections."

This point of view may very well apply to movie chances for Oscars and other measures of success such as profit. Zhang and Li (2010) state that one possible quantitative measure of controversy is simply the standard deviation of the numerical ratings, and we adopt that point of view here as well.

To extract themes from movie reviews, we use the text mining algorithms proposed by SAS Text Miner within the Enterprise Miner platform. Details of the algorithm are published elsewhere, but the algorithms works essentially as follows.

Each review is defined to be a document, and a very large but sparse matrix is constructed with documents as rows and all possible terms (words in documents and their grammatical relatives, such as for example begin, began, beginning etc.) as columns. Singular Matrix Decomposition (SVD) techniques are used to reduce the matrix without losing too much information and a cluster analysis is applied to the reduced matrix, yielding for each set of reviews, a set of clusters of documents. The list of most common terms in these documents is then obtained and gives an idea of the main themes in that cluster.

As an illustration, Table 2 displays the results of this clustering exercise for Argo. For example the main theme in cluster 3 is clearly related to perceived Oscar chances

Table 2 Clusters and main terms for Argo reviews

Cluster	Main terms	Number of documents
1	tony + ambassador + plan + embassy mendez canadian six + hostage + crisis chambers cia fake john goodman arkin	142
2	+movie people watching + good movies great + world first + end characters + fact + country history historical + time	95
3	best + picture acting + great + oscar well + good affleck + actor + director argo alan ben + film + movie	149
4	+feel + seat + edge characters + little especially few films + know + thriller suspense + end + fact + film fake	22
5	canadians shah airport history iranians americans + country canadian people iranian events historical + fact cia american	72
6	chambers bryan + ambassador cranston + plan + crisis john mendez iranian + actor tony fake + thriller alan especially	44

for the movie, director and leading actor (Ben Affleck) and the main theme in cluster 4 is about the thrilling aspects of the movie.

In the case of Argo, the text analysis yielded 6 clusters. Reviews of Amour, a movie by a controversial director (Haneke) on a very complex theme, related to death and euthanasia, yielded 23 clusters. Aside from whether these extracted themes are expressing positive or negative sentiments (and the approach to measuring perceived controversy in Zhang and Li (2010) does use the number of positive and negative sentiments), it is reasonable to surmise that the number of issues such a complex movie rises may be simply too large for a group to rally on.

Our study, based on just nine movies, is still preliminary; we suggest that a very interesting future direction would involve looking at measures of complexity for a much larger number of movies, and investigating how correlated such measure would be to for example the standard deviation of ratings.

Figure 11 displays a scatter plot of the standard deviation of ratings against the number of clusters extracted by the text analysis for the nine movies.

With a few caveats, first that the standard deviation of ratings is fairly small for all nine movies, and that Zero and Amour act as outliers, we can see that that the standard deviation of ratings shows a propensity to increase with the number of clusters.

It is interesting to note that the five serious contenders for the Best Picture award as perceived by the Intrade market (Argo, Les Misérables, Lincoln, Silver Linings Playbook, and Zero Dark Thirty) tend to yield a moderate number of clusters, in other words tend to raise a number of issues which a group can potentially rally on (see Fig. 11 and Table 3).

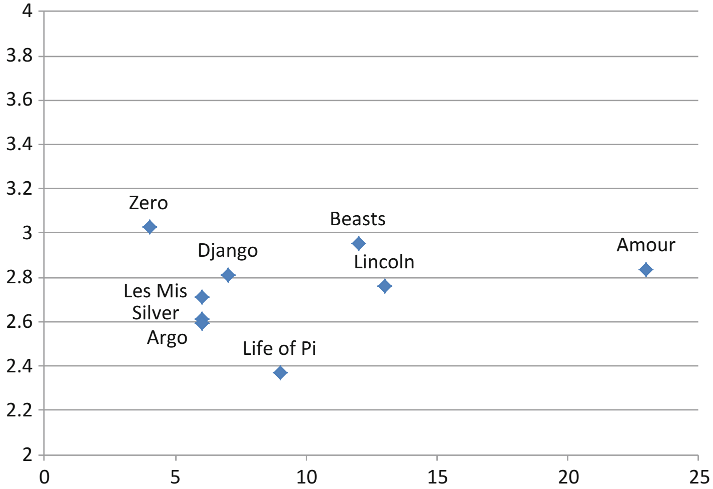
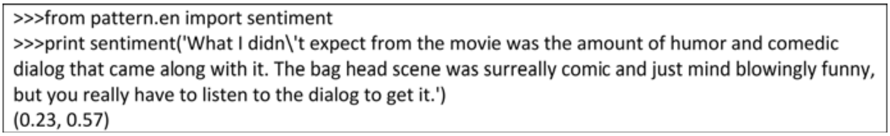


Fig. 11 Standard deviation of ratings and number of clusters for each nominated movie to win the 2013 Best Picture award

Table 3 Number of extracted themes and statistics for the nine movies nominated for a Best Picture award

	Mean rating	Number of themes (of clusters)	Mean Intrade close	Last Intrade close	St dev of ratings	Profit
Amour	7.1	23	0.99	0.4	2.84	$ (2.16)
Beasts	6.8	12	1.07	0.4	2.96	$ 10.98
Django	7.4	7	1.22	0.5	2.81	$ 62.80
Zero	6.3	4	5.53	0.7	3.03	$ 55.72
Les Mis	7.3	6	11.27	1.2	2.71	$ 87.78
Life of Pi	7.8	9	3.43	1.5	2.37	$ 4.98
Silver	7.5	6	6.05	3	2.61	$ 111.09
Lincoln	7.2	13	36.24	10.3	2.76	$ 117.20
Argo	7.4	6	32.27	82	2.60	$ 91.52



```
>>>from pattern.en import sentiment
>>>print sentiment('What I didn\'t expect from the movie was the amount of humor and comedic
dialog that came along with it. The bag head scene was surreally comic and just mind blowingly funny,
but you really have to listen to the dialog to get it.')
(0.23, 0.57)
```

Fig. 12 Sentiment analysis Python code

1.2.5 Sentiment Analysis

In this section, we calculate sentiment and subjectivity scores for each review by using the pattern.en module for Python. The pattern.en module is a natural language processing (NLP) toolkit for English, developed by CLiPS (Computational Linguistics & Psycholinguistics, http://www.clips.ua.ac.be/).

The pattern.en module bundles a lexicon of adjectives (e.g., good, bad, amazing, and irritating) that occur frequently in product reviews, annotated with scores for sentiment polarity (negative ↔ positive) and subjectivity (objective ↔ subjective). According to the website, accuracy is about 75 % for movie reviews. The sentiment() function of pattern.en module returns a (polarity, subjectivity)-tuple for a given word, string, sentence, or text, based on the adjectives it contains, where polarity is a value between −1.0 and +1.0 and subjectivity between 0.0 and 1.0. An example is given in Fig. 12.

It appears that sentiment scores and numerical ratings are positively correlated (Fig. 13) and that the standard deviations of sentiment scores increase fairly linearly with the means of sentiment scores (Fig. 14).

Quite interestingly, numerical ratings, while positively correlated to sentiment scores as we have seen (Fig. 13), behave differently from sentiment scores in that the standard deviations of numerical ratings seem to decrease quadratically with the means of numerical ratings (Fig. 15).

Let us now examine how the standard deviations and means of sentiment and subjectivity scores depend on the number of clusters (themes) extracted from the reviews by the text mining process.

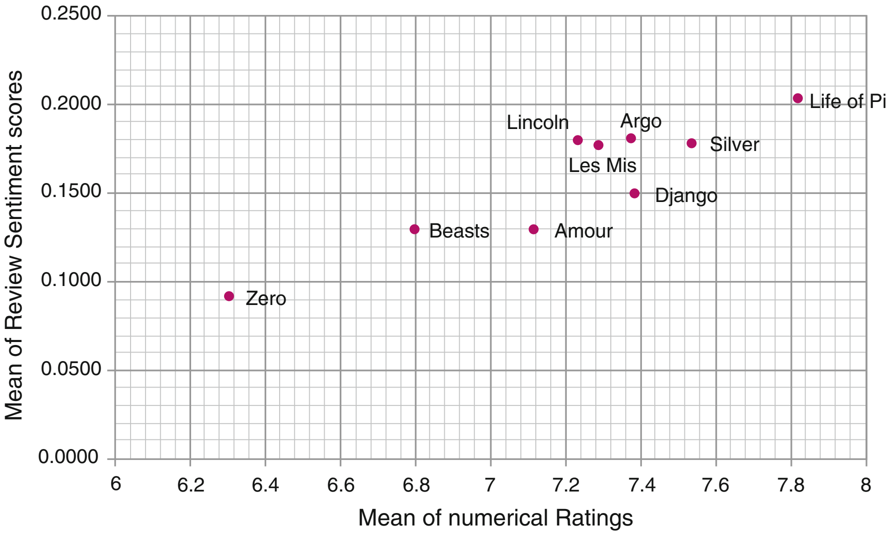


Fig. 13 Mean of sentiment scores and mean of numerical ratings

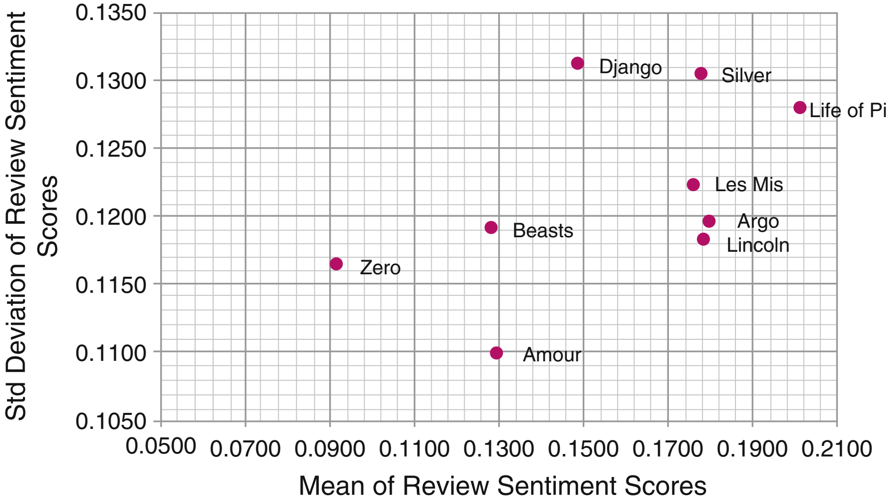


Fig. 14 Standard deviation of sentiment scores and mean of sentiment scores

We observe that wider variations in subjectivity scores (Fig. 16) tend to correspond to more extracted clusters, as one might expect (leaving aside Amour), while mean subjectivity scores tend to depend in a complicated non-linear way on the number of clusters (Fig. 17).

Higher means (and standard deviations) of sentiment scores (Figs. 18 and 19) tend to occur for a moderate number of extracted clusters (themes). This lends credence to our suggestion that moderate controversy might lead to beneficial results.

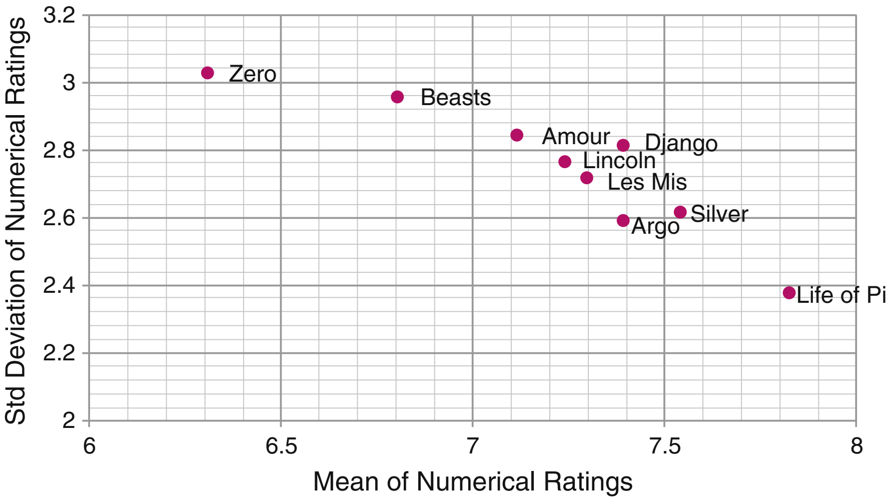


Fig. 15 Standard deviation and mean of numerical ratings

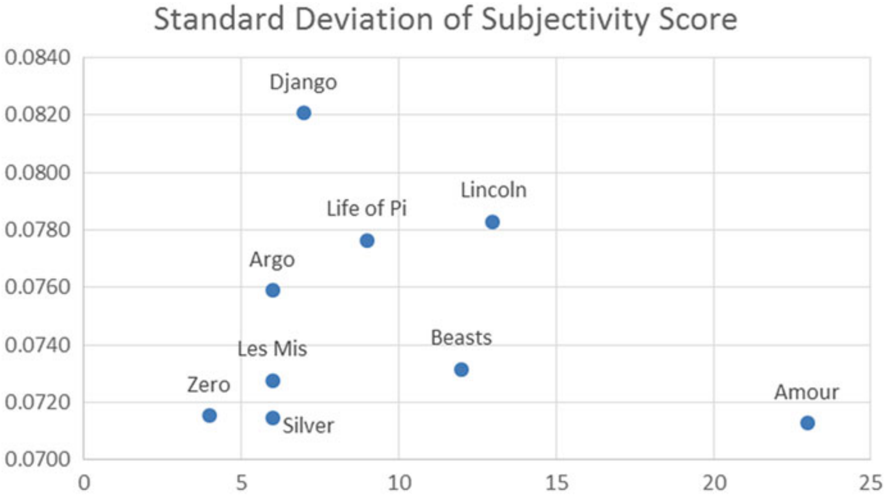


Fig. 16 Standard deviation of subjectivity scores and number of clusters

2 Future Perspectives

Further investigations of controversy indices in movie reviews and their role on measures of success would be very interesting. Controversy is highly correlated with Word-of-mouth (WOM) activity and WOM marketing. WOM is the process of information exchange, involving in particular recommendations about products and

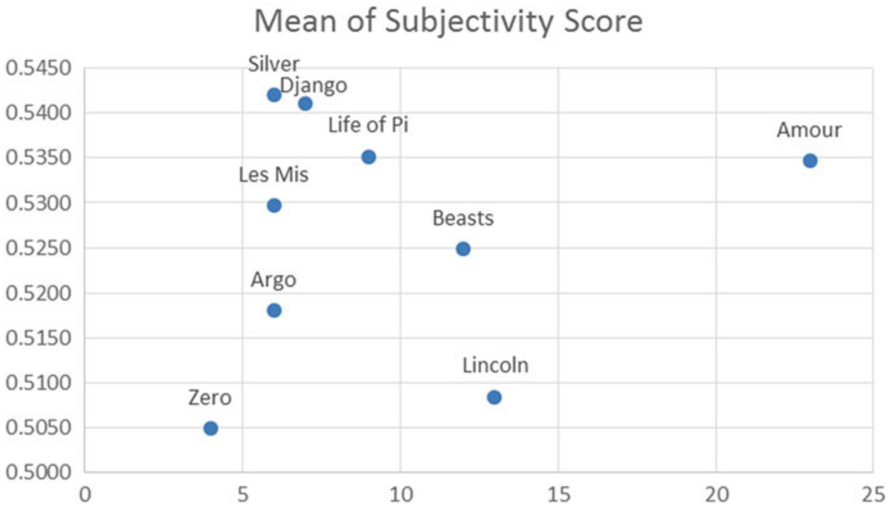


Fig. 17 Mean of subjectivity scores and number of clusters

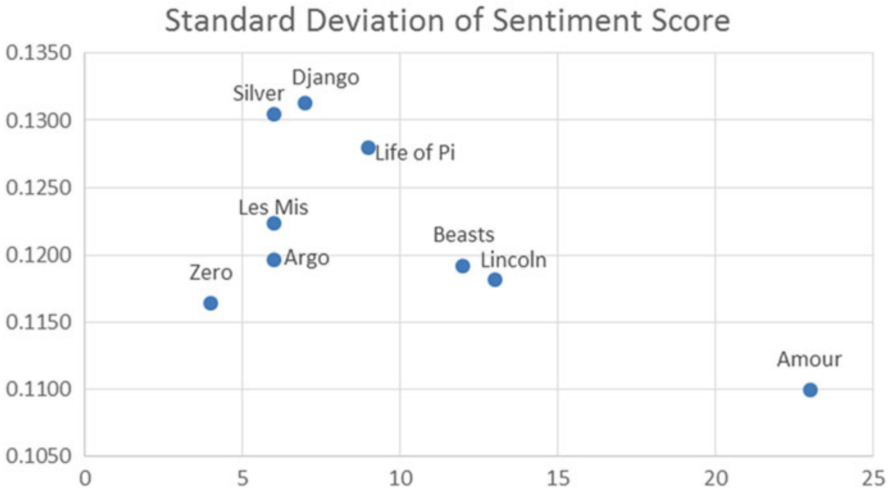


Fig. 18 Standard deviation of sentiment scores and number of clusters

services, between two people in an informal way (O'Leary and Shehan 2008). WOM communication could have a strong influence on consumer short-term and long-term purchasing behavior, influencing both short-term and long-term judgments (Bone 1995). Another advantage of WOM is that cost of WOM marketing is low, for both online and offline channels.

WOM could be positive or negative. The question is, do customers' negative opinions always fall on the bad side of the coin, or is there any advocacy to brand

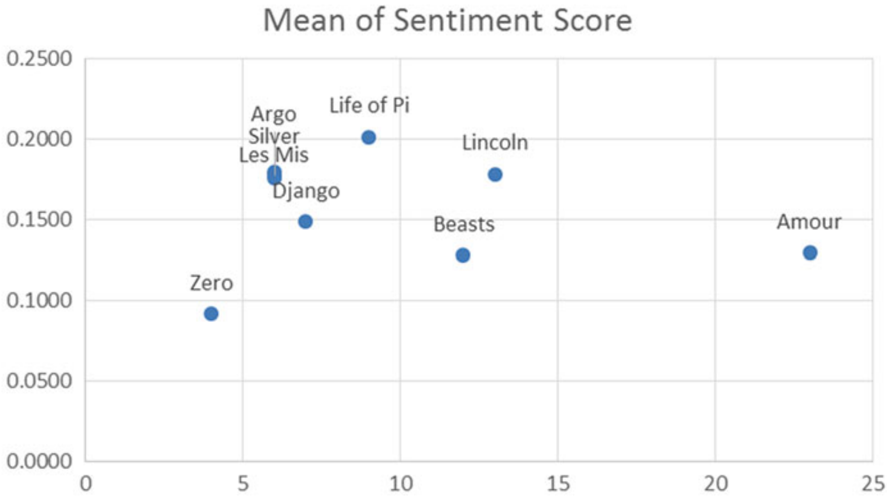


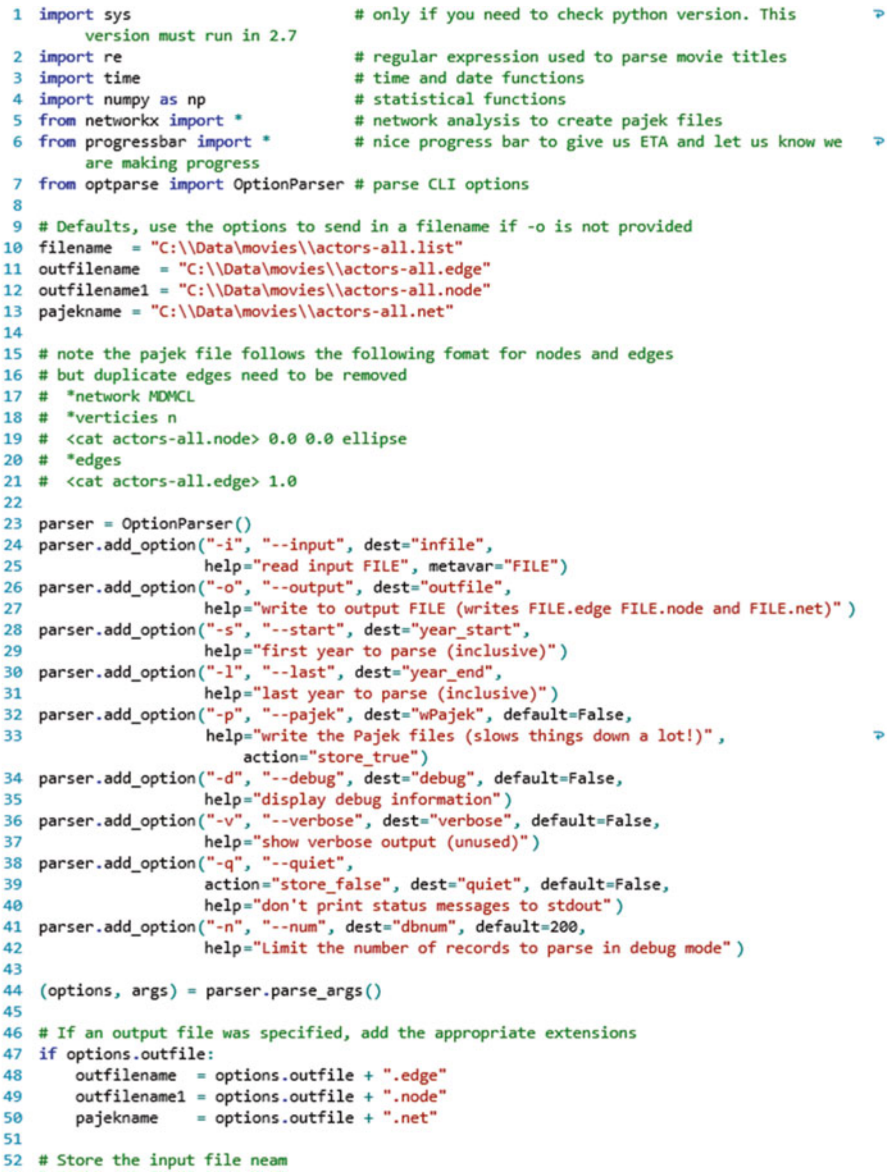
Fig. 19 Mean of sentiment scores and number of clusters

coming from negative WOM or mixed WOM (so-called controversy)? Some research indicates the possibility that controversy arising from consumers' opinion might have a positive impact. Liu (2007) finds that movie box office revenue is correlated with the volume of WOM activity, but NOT correlated with the percentage of negative critical reviews. Zhang and Li (2010) observe that controversy can attract market attention and potentially yield strong sales.

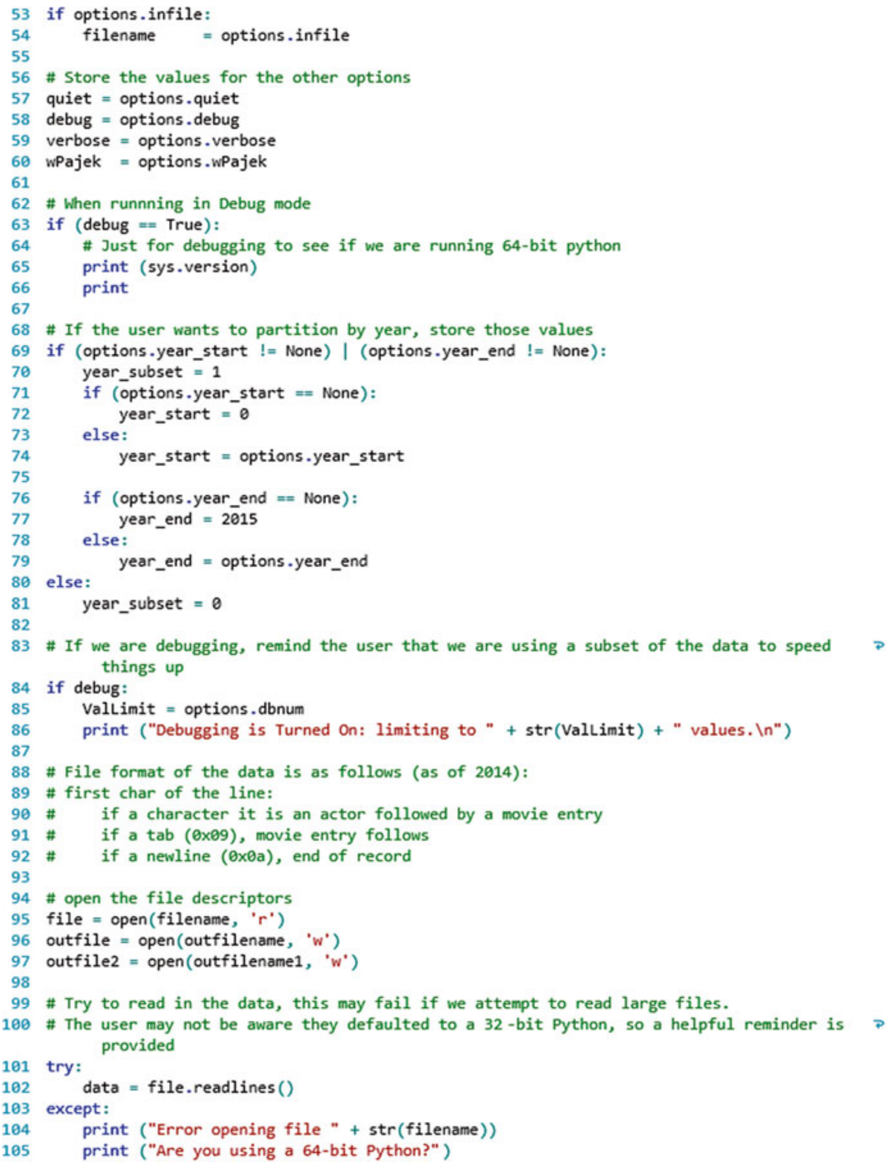
On the other hand, consider the role of controversy in the 2014 Best Animated Feature Award, as pertains to Hayao Miyazaki's "The wind rises." "The wind rises" lost the award in large part because of controversy surrounding the theme in the movie. We suggest that getting a better handle on what constitutes controversy, relying on progress in text mining techniques, is likely to illuminate problems which are to date difficult to apprehend.

Appendix: Code Needed to Perform the Analyses in Chap. 3

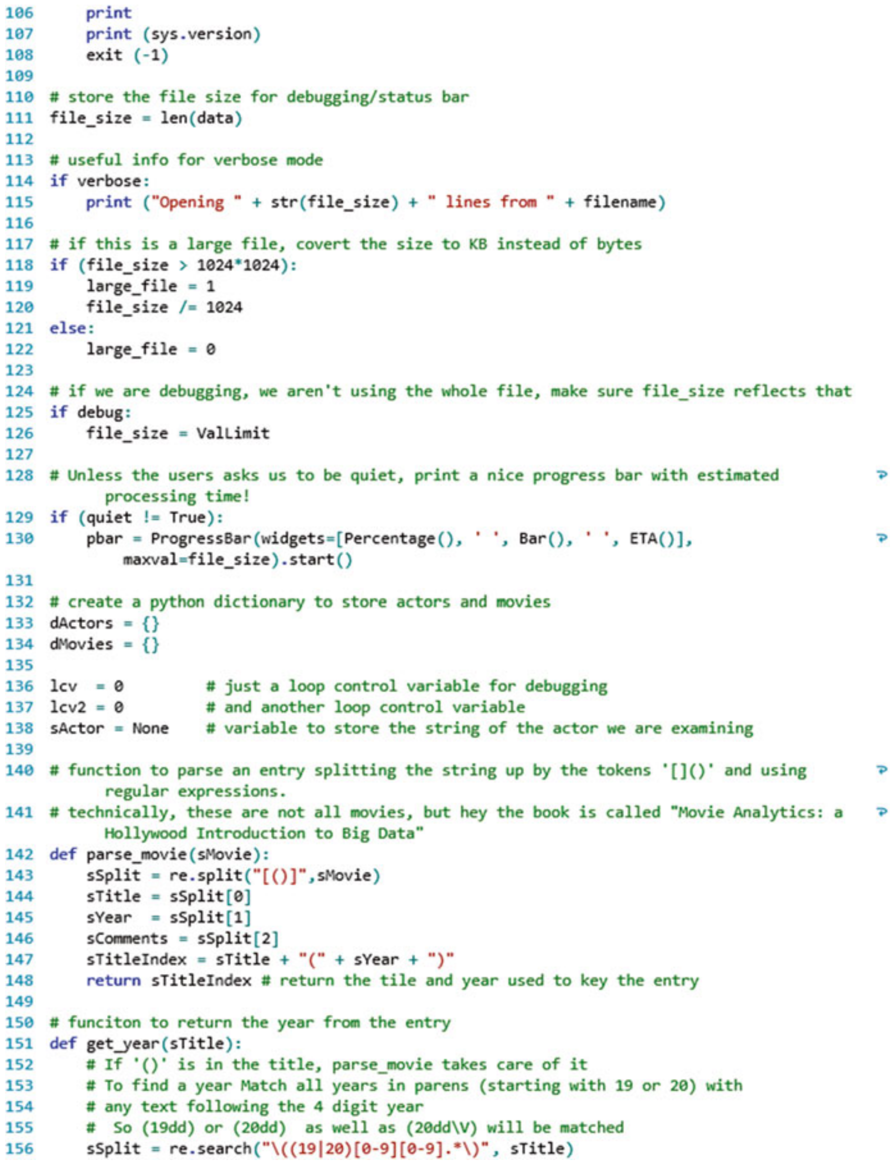
D. Haughton et al., *Movie Analytics*, SpringerBriefs in Statistics,
DOI 10.1007/978-3-319-09426-7



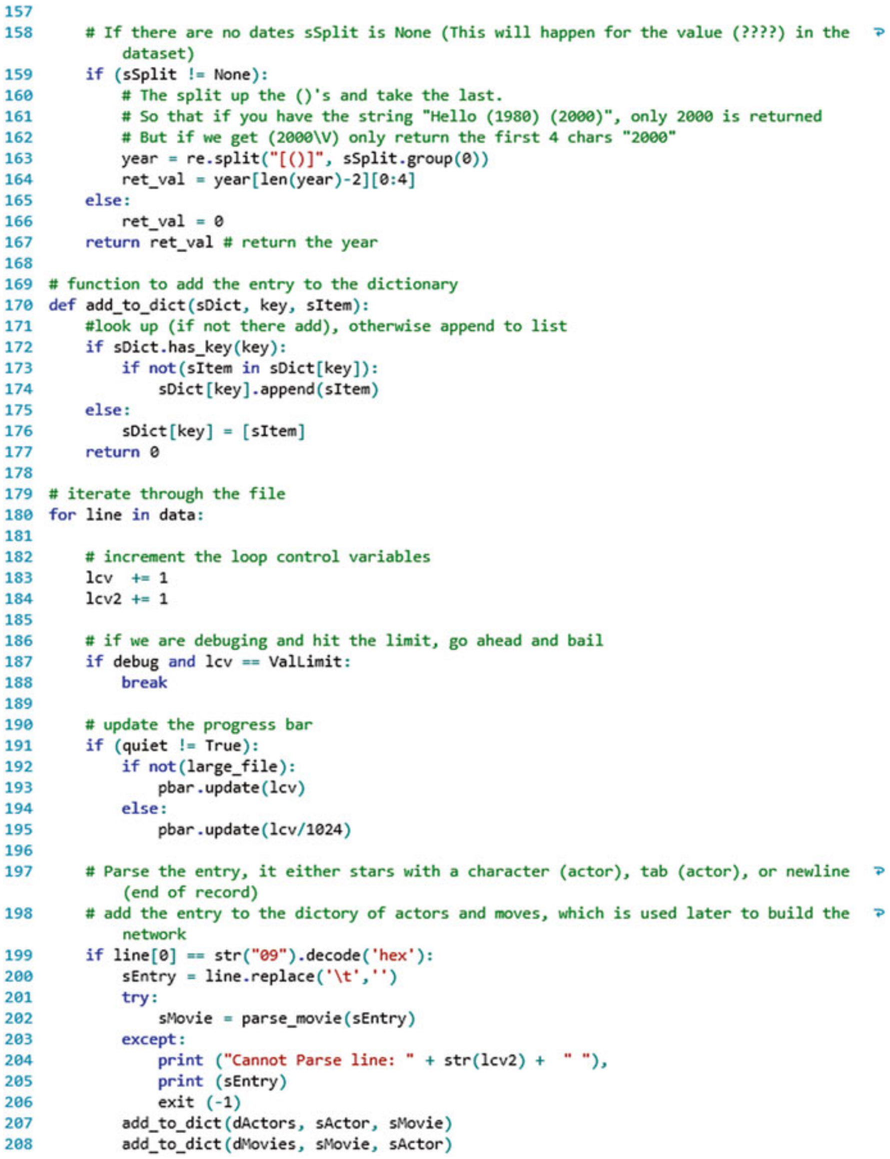
```
import sys                        # only if you need to check python version. This version must run in 2.7
import re                         # regular expression used to parse movie titles
import time                       # time and date functions
import numpy as np                # statistical functions
from networkx import *            # network analysis to create pajek files
from progressbar import *         # nice progress bar to give us ETA and let us know we are making progress
from optparse import OptionParser # parse CLI options

# Defaults, use the options to send in a filename if -o is not provided
filename  = "C:\\Data\movies\\actors-all.list"
outfilename  = "C:\\Data\movies\\actors-all.edge"
outfilename1 = "C:\\Data\movies\\actors-all.node"
pajekname = "C:\\Data\movies\\actors-all.net"

# note the pajek file follows the following fomat for nodes and edges
# but duplicate edges need to be removed
#  *network MDMCL
#  *verticies n
#  <cat actors-all.node> 0.0 0.0 ellipse
#  *edges
#  <cat actors-all.edge> 1.0

parser = OptionParser()
parser.add_option("-i", "--input", dest="infile",
                  help="read input FILE", metavar="FILE")
parser.add_option("-o", "--output", dest="outfile",
                  help="write to output FILE (writes FILE.edge FILE.node and FILE.net)" )
parser.add_option("-s", "--start", dest="year_start",
                  help="first year to parse (inclusive)" )
parser.add_option("-l", "--last", dest="year_end",
                  help="last year to parse (inclusive)" )
parser.add_option("-p", "--pajek", dest="wPajek", default=False,
                  help="write the Pajek files (slows things down a lot!)" , action="store_true")
parser.add_option("-d", "--debug", dest="debug", default=False,
                  help="display debug information" )
parser.add_option("-v", "--verbose", dest="verbose", default=False,
                  help="show verbose output (unused)" )
parser.add_option("-q", "--quiet",
                  action="store_false", dest="quiet", default=False,
                  help="don't print status messages to stdout" )
parser.add_option("-n", "--num", dest="dbnum", default=200,
                  help="Limit the number of records to parse in debug mode" )

(options, args) = parser.parse_args()

# If an output file was specified, add the appropriate extensions
if options.outfile:
    outfilename  = options.outfile + ".edge"
    outfilename1 = options.outfile + ".node"
    pajekname    = options.outfile + ".net"

# Store the input file neam
```



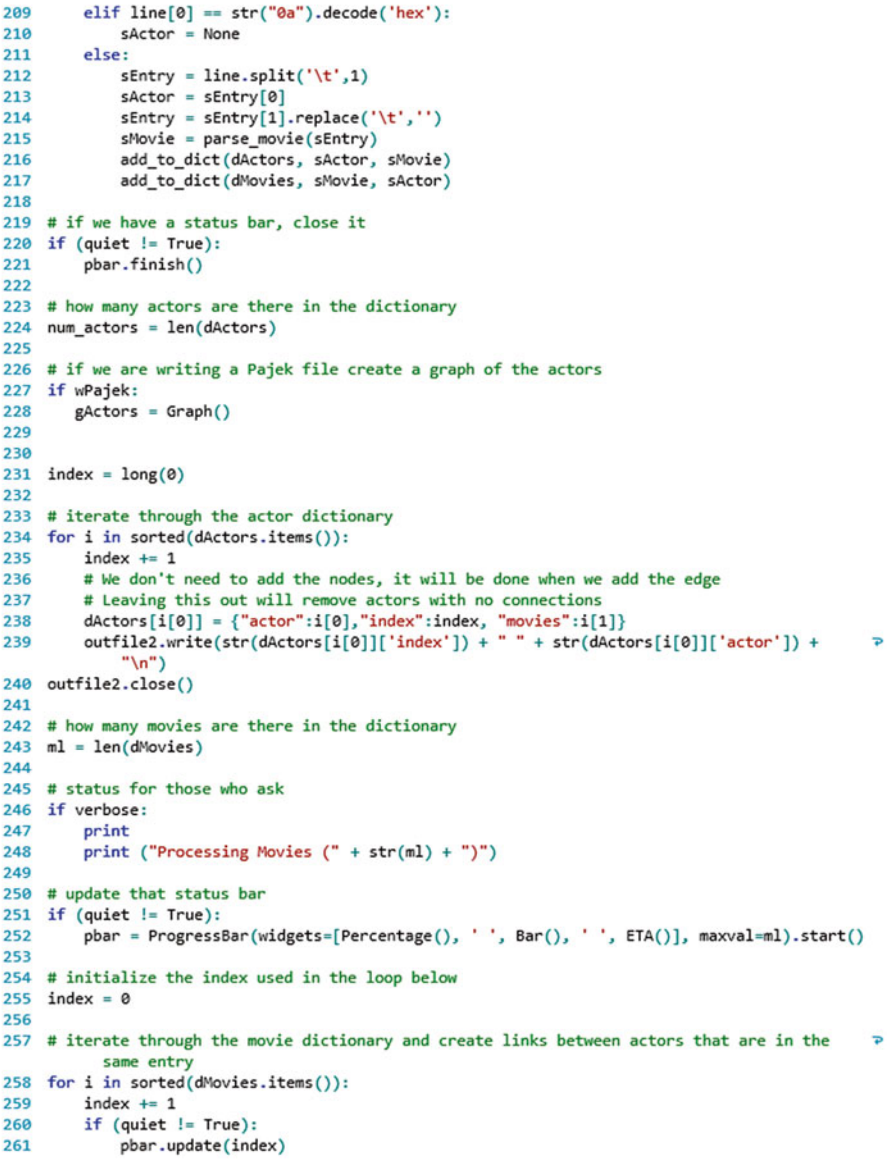
```
if options.infile:
    filename     = options.infile

# Store the values for the other options
quiet = options.quiet
debug = options.debug
verbose = options.verbose
wPajek  = options.wPajek

# When runnning in Debug mode
if (debug == True):
    # Just for debugging to see if we are running 64-bit python
    print (sys.version)
    print

# If the user wants to partition by year, store those values
if (options.year_start != None) | (options.year_end != None):
    year_subset = 1
    if (options.year_start == None):
        year_start = 0
    else:
        year_start = options.year_start

    if (options.year_end == None):
        year_end = 2015
    else:
        year_end = options.year_end
else:
    year_subset = 0

# If we are debugging, remind the user that we are using a subset of the data to speed things up
if debug:
    ValLimit = options.dbnum
    print ("Debugging is Turned On: limiting to " + str(ValLimit) + " values.\n")

# File format of the data is as follows (as of 2014):
# first char of the line:
#     if a character it is an actor followed by a movie entry
#     if a tab (0x09), movie entry follows
#     if a newline (0x0a), end of record

# open the file descriptors
file = open(filename, 'r')
outfile = open(outfilename, 'w')
outfile2 = open(outfilename1, 'w')

# Try to read in the data, this may fail if we attempt to read large files.
# The user may not be aware they defaulted to a 32 -bit Python, so a helpful reminder is provided
try:
    data = file.readlines()
except:
    print ("Error opening file " + str(filename))
    print ("Are you using a 64-bit Python?")
```



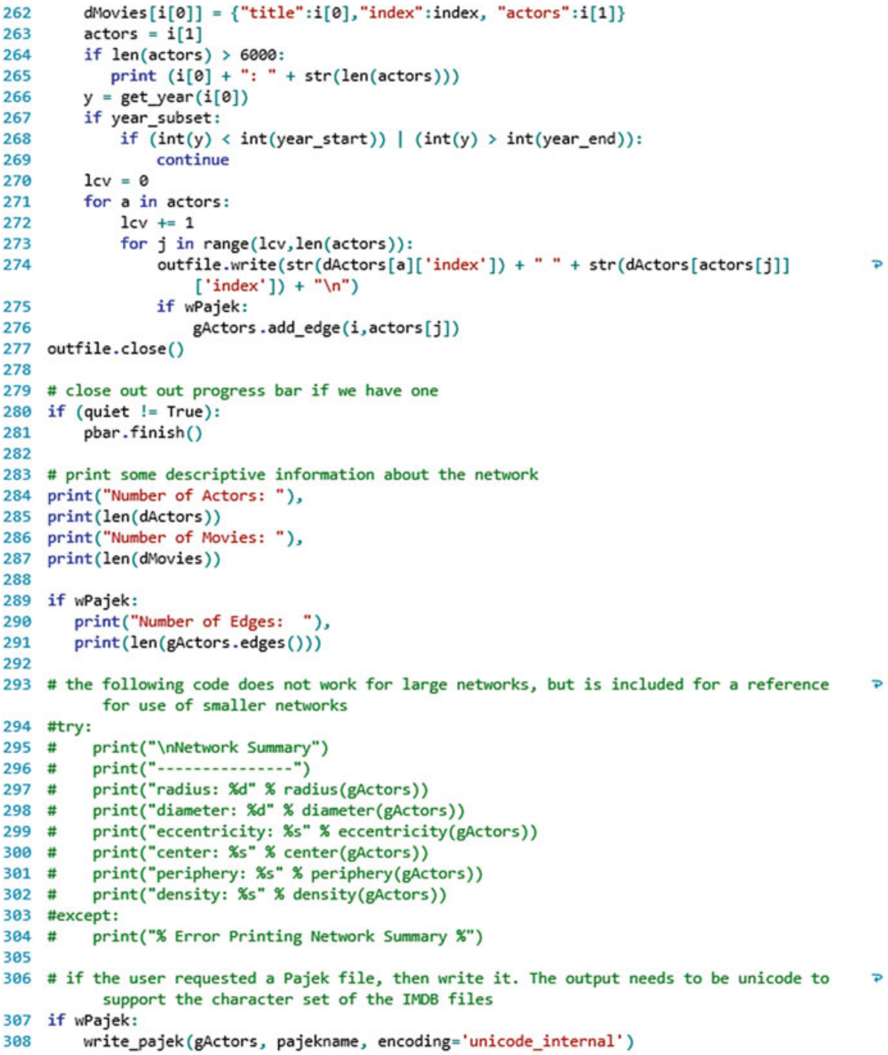
```
    print
    print (sys.version)
    exit (-1)

# store the file size for debugging/status bar
file_size = len(data)

# useful info for verbose mode
if verbose:
    print ("Opening " + str(file_size) + " lines from " + filename)

# if this is a large file, covert the size to KB instead of bytes
if (file_size > 1024*1024):
    large_file = 1
    file_size /= 1024
else:
    large_file = 0

# if we are debugging, we aren't using the whole file, make sure file_size reflects that
if debug:
    file_size = ValLimit

# Unless the users asks us to be quiet, print a nice progress bar with estimated processing time!
if (quiet != True):
    pbar = ProgressBar(widgets=[Percentage(), ' ', Bar(), ' ', ETA()], maxval=file_size).start()

# create a python dictionary to store actors and movies
dActors = {}
dMovies = {}

lcv  = 0         # just a loop control variable for debugging
lcv2 = 0         # and another loop control variable
sActor = None    # variable to store the string of the actor we are examining

# function to parse an entry splitting the string up by the tokens '[]()' and using regular expressions.
# technically, these are not all movies, but hey the book is called "Movie Analytics: a Hollywood Introduction to Big Data"
def parse_movie(sMovie):
    sSplit = re.split("[()]",sMovie)
    sTitle = sSplit[0]
    sYear  = sSplit[1]
    sComments = sSplit[2]
    sTitleIndex = sTitle + "(" + sYear + ")"
    return sTitleIndex # return the tile and year used to key the entry

# funciton to return the year from the entry
def get_year(sTitle):
    # If '()' is in the title, parse_movie takes care of it
    # To find a year Match all years in parens (starting with 19 or 20) with
    # any text following the 4 digit year
    #  So (19dd) or (20dd)  as well as (20dd\V) will be matched
    sSplit = re.search("\((19|20)[0-9][0-9].*\)", sTitle)
```



```

    # If there are no dates sSplit is None (This will happen for the value (????) in the dataset)
    if (sSplit != None):
        # The split up the ()'s and take the last.
        # So that if you have the string "Hello (1980) (2000)", only 2000 is returned
        # But if we get (2000\V) only return the first 4 chars "2000"
        year = re.split("[()]", sSplit.group(0))
        ret_val = year[len(year)-2][0:4]
    else:
        ret_val = 0
    return ret_val # return the year

# function to add the entry to the dictionary
def add_to_dict(sDict, key, sItem):
    #look up (if not there add), otherwise append to list
    if sDict.has_key(key):
        if not(sItem in sDict[key]):
            sDict[key].append(sItem)
    else:
        sDict[key] = [sItem]
    return 0

# iterate through the file
for line in data:

    # increment the loop control variables
    lcv  += 1
    lcv2 += 1

    # if we are debuging and hit the limit, go ahead and bail
    if debug and lcv == ValLimit:
        break

    # update the progress bar
    if (quiet != True):
        if not(large_file):
            pbar.update(lcv)
        else:
            pbar.update(lcv/1024)

    # Parse the entry, it either stars with a character (actor), tab (actor), or newline (end of record)
    # add the entry to the dictory of actors and moves, which is used later to build the network
    if line[0] == str("09").decode('hex'):
        sEntry = line.replace('\t','')
        try:
            sMovie = parse_movie(sEntry)
        except:
            print ("Cannot Parse line: " + str(lcv2) +  " "),
            print (sEntry)
            exit (-1)
        add_to_dict(dActors, sActor, sMovie)
        add_to_dict(dMovies, sMovie, sActor)
```



```
    elif line[0] == str("0a").decode('hex'):
        sActor = None
    else:
        sEntry = line.split('\t',1)
        sActor = sEntry[0]
        sEntry = sEntry[1].replace('\t','')
        sMovie = parse_movie(sEntry)
        add_to_dict(dActors, sActor, sMovie)
        add_to_dict(dMovies, sMovie, sActor)

# if we have a status bar, close it
if (quiet != True):
    pbar.finish()

# how many actors are there in the dictionary
num_actors = len(dActors)

# if we are writing a Pajek file create a graph of the actors
if wPajek:
   gActors = Graph()


index = long(0)

# iterate through the actor dictionary
for i in sorted(dActors.items()):
    index += 1
    # We don't need to add the nodes, it will be done when we add the edge
    # Leaving this out will remove actors with no connections
    dActors[i[0]] = {"actor":i[0],"index":index, "movies":i[1]}
    outfile2.write(str(dActors[i[0]]['index']) + " " + str(dActors[i[0]]['actor']) + "\n")
outfile2.close()

# how many movies are there in the dictionary
ml = len(dMovies)

# status for those who ask
if verbose:
    print
    print ("Processing Movies (" + str(ml) + ")")

# update that status bar
if (quiet != True):
    pbar = ProgressBar(widgets=[Percentage(), ' ', Bar(), ' ', ETA()], maxval=ml).start()

# initialize the index used in the loop below
index = 0

# iterate through the movie dictionary and create links between actors that are in the same entry
for i in sorted(dMovies.items()):
    index += 1
    if (quiet != True):
        pbar.update(index)
```



```
    dMovies[i[0]] = {"title":i[0],"index":index, "actors":i[1]}
    actors = i[1]
    if len(actors) > 6000:
       print (i[0] + ": " + str(len(actors)))
    y = get_year(i[0])
    if year_subset:
        if (int(y) < int(year_start)) | (int(y) > int(year_end)):
            continue
    lcv = 0
    for a in actors:
        lcv += 1
        for j in range(lcv,len(actors)):
            outfile.write(str(dActors[a]['index']) + " " + str(dActors[actors[j]]['index']) + "\n")
            if wPajek:
                gActors.add_edge(i,actors[j])
outfile.close()

# close out out progress bar if we have one
if (quiet != True):
    pbar.finish()

# print some descriptive information about the network
print("Number of Actors: "),
print(len(dActors))
print("Number of Movies: "),
print(len(dMovies))

if wPajek:
   print("Number of Edges:  "),
   print(len(gActors.edges()))

# the following code does not work for large networks, but is included for a reference for use of smaller networks
#try:
#    print("\nNetwork Summary")
#    print("---------------")
#    print("radius: %d" % radius(gActors))
#    print("diameter: %d" % diameter(gActors))
#    print("eccentricity: %s" % eccentricity(gActors))
#    print("center: %s" % center(gActors))
#    print("periphery: %s" % periphery(gActors))
#    print("density: %s" % density(gActors))
#except:
#    print("% Error Printing Network Summary %")

# if the user requested a Pajek file, then write it. The output needs to be unicode to support the character set of the IMDB files
if wPajek:
    write_pajek(gActors, pajekname, encoding='unicode_internal')
```

References

Beiro MG, Alvarez-Hamelin JI, Busch JR. A low complexity visualization tool that helps to perform complex systems analysis. New J Phys. 2008;10:125003. doi:10.1088/1367-2630/10/12/125003.

Bialik C. And the Oscar-pool winner are … the stats dudes. Wall Street Journal, February 23, 2013.

Chisholm D, Norman G. When to exit a product: evidence from the U.S. motion-pictures exhibition market. Available from http://www.researchgate.net/publication/24137962_When_to_Exit_a_Product_Evidence_from_the_U.S._Motion-Pictures_Exhibition_Market. 2005.

Bone PF. Word-of-mouth effects on short-term and long-term product judgments. J Bus Res. 1995;32(3):213–23.

Davenport T. Big data at work: dispelling the myths, uncovering the opportunities. Boston, MA: Harvard Business School Publishing Corporation; 2014.

De Nooy W, Mrvar A, Batagelj V. Exploratory social network analysis with Pajek. Cambridge: Cambridge University Press; 2011.

El Assady M, Hafner D, Hund M, Jager A, Jentner W, Rohrdantz C, Fisher F, Simon S, Schreck T, Keim D. Visual analytics for the prediction of movie rating and box office performance. Available from http://bib.dbvis.de/uploadedFiles/elassady.pdf. 2013.

Erikson RS, Wlezien C. Are political markets really superior to polls as election predictors? Public Opin Q. 2008;72:190–215.

Franks B. Taming the big data tidal wave: finding opportunities in huge data streams with advanced analytics. New York, NY: Wiley; 2012.

Gephi. Network visualization and analysis platform. Available from http://gephi.github.io. 2015.

Gold M, McClarren R, Gaughan C. The lessons Oscar taught us: data science and media & entertainment. Big Data. 2013;1:105–9.

Ignacio Alvarez-Hamelin J, Dall'Asta L, Barrat A, Vespignani A. K-core decomposition: a tool for the visualization of large scale networks. Available from http://arxiv.org/pdf/cs/0504107.pdf. 2005.

Jacomy M, Venturini T, Heymann S, Bastian M. ForceAtlas2, a continuous graph layout algorithm for handy network visualization designed for the gephi software. PLoS One. 2014;9(6):e98679. doi:10.1371/journal.pone.0098679.

Leamer EE. What's a recession, anyway? Cambridge, MA: National Bureau of Economic Research; 2008.

Leonhardt D. Oscars 2015: an excellent night for prediction markets. The New York Times, February 23, 2015.

Liu B. Web data mining: exploring hyperlinks, contents, and usage data. New York, NY: Springer; 2007.

McKenzie J. The economics of movies: a literature survey. J Econ Surv. 2012;26(1):42–70.

D. Haughton et al., *Movie Analytics*, SpringerBriefs in Statistics,
DOI 10.1007/978-3-319-09426-7

O'Leary S, Shehan K. Building buzz to beat the big boys: word of mouth marketing for small businesses. Westport, CT: Praeger Publishers; 2008.

Pajek. Program for large network analysis. Available from http://pajek.imfm.si/doku.php?id=pajek. 2015.

Python. Programming language. Available from www.python.org. 2015.

Rothschild D, Wolfers J. Market manipulation muddies election outlook. The Wall Street Journal, October 2, 2008.

Saxon I. Intrade prediction market accuracy and efficiency: an analysis of the 2004 and 2008 democratic presidential nomination contests. Dissertation. University of Nottingham. 2010.

Shambaugh JC, et al. The Euro's three crises. Brookings papers on economic activity. Washington, DC: Brooking institution; 2012. p. 157–231.

The Economist. The data deluge. Available from http://www.economist.com/printedition/2010-02-27. 2010.

Whipp, G. The Gold Standard; now for real insight into Oscars—by the guilds. Los Angeles Times, January 26, 2013.

Zhang Z, Li X. Controversy in marketing: mining sentiments in social media. In: Proceedings of the 43rd Hawaii international conference on systems sciences. Washington, DC: IEEE; 2010.