Dataset #2: Movie Ratings

This is a dataset about Movies, obtained using data from IMDB. (You can get amazing amounts of IMDB information from ftp://ftp.fu-berlin.de/pub/misc/movies/database/ (ftp://ftp.fu-berlin.de/pub/misc/movies/database/))

Goal: predict the IMDB rating of a move. Specifically: given the attributes of each input movie, predict its IMDB rating.

The Rating column of the training set gives examples. Each rating is a value of the form X.Y, where X and Y are single digits. Your job is to predict these values as closely as possible.

A histogram of rating values is shown below, near the end of this notebook.

Schema of the Dataset

The dataset has the following columns:

Title	movie title
Year	year released
Length	length in minutes
Budget	production budget in US dollars (usually NA)
Rating	average rating of IMDB users
Votes	number of voting IMDB users
R1	approximate percentage of users voting for rating: 1
R2	approximate percentage of users voting for rating: 2
R3	approximate percentage of users voting for rating: 3
R4	approximate percentage of users voting for rating: 4
R5	approximate percentage of users voting for rating: 5
R6	approximate percentage of users voting for rating: 6
R7	approximate percentage of users voting for rating: 7
R8	approximate percentage of users voting for rating: 8
R9	approximate percentage of users voting for rating: 9
R10	approximate percentage of users voting for rating: 10
MPAA	MPAA parental guidance rating (blank, NC-17, PG, PG-13, R)
Action	1 if Action, 0 otherwise
Animation	1 if Animation, 0 otherwise
Comedy	1 if Comedy, 0 otherwise
Drama	1 if Drama, 0 otherwise
Documentary	1 if Documentary, 0 otherwise
Romance	1 if Romance, 0 otherwise
Short	1 if Short Film, 0 otherwise

Caution

The dataset has missing values (such as "NA" Budget values and blank MPAA ratings).

A First Look at the Dataset

```
In [1]: import pandas as pd
        import numpy as np

        Movies = pd.DataFrame.from_csv('Movies.csv')

        Movies.head()
```

Out[1]:

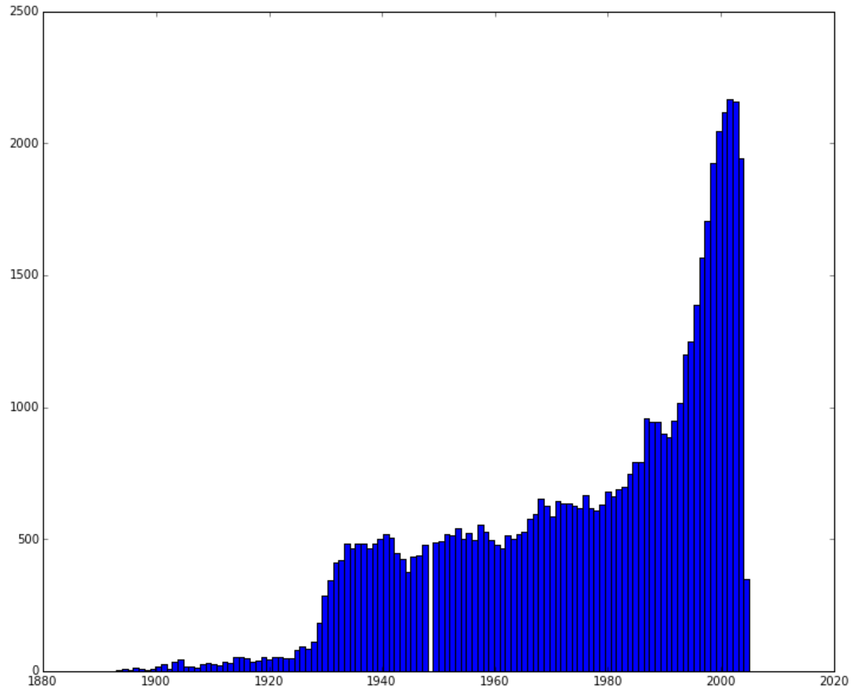
	Year	Length	Budget	Rating	Votes	R1	R2	R3	R4	R5	...	R9	R10	MPAA	Action	Animation	Comedy	Drama	Documentary	Romance	Short
Title																					
$	1971	121	NaN	6.4	348	5	5	5	5	15	...	5	5	NaN	0	0	1	1	0	0	0
$1000 a Touchdown	1939	71	NaN	6.0	20	0	15	5	25	15	...	5	15	NaN	0	0	1	0	0	0	0
$21 a Day Once a Month	1941	7	NaN	8.2	5	0	0	0	0	0	...	25	25	NaN	0	1	0	0	0	0	1
$40,000	1996	70	NaN	8.2	6	15	0	0	0	0	...	35	45	NaN	0	0	1	0	0	0	0
$50,000 Climax Show, The	1975	71	NaN	3.4	17	25	5	0	15	15	...	0	25	NaN	0	0	0	0	0	0	0

5 rows × 23 columns

```
In [6]: movie_years = Movies[['Year']].values

        earliest = np.min(movie_years)
        latest = np.max(movie_years)

        print((earliest, latest))

        (1893, 2005)
```

```
In [7]: import matplotlib.pyplot as plt
        %matplotlib inline
        plt.rcParams['figure.figsize'] = (12.0, 10.0)
```

```
In [8]: plt.hist( movie_years, bins=len(range(earliest,latest))+2 )
        plt.show()
```

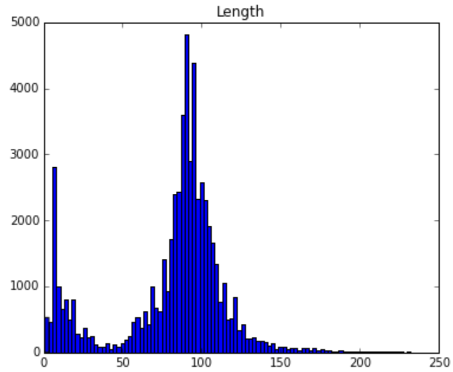


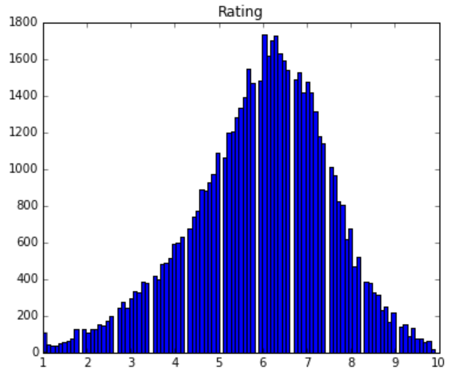
```
In [9]: Movies.columns

Out[9]: Index([u'Year', u'Length', u'Budget', u'Rating', u'Votes', u'R1', u'R2', u'R3', u'R4', u'R5', u'R6', u'R7', u'R8', u'R9', u'R1
        0', u'MPAA', u'Action', u'Animation', u'Comedy', u'Drama', u'Documentary', u'Romance', u'Short'], dtype='object')
```

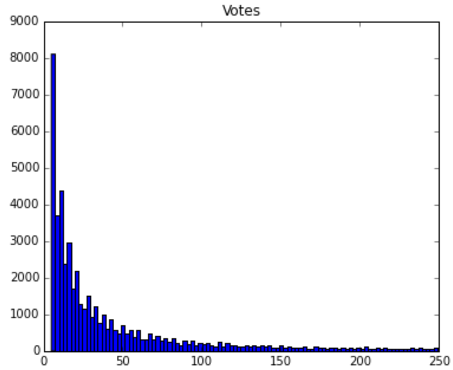
```
In [10]: movie_years[ (movie_years < 1900) ]

Out[10]: array([1897, 1899, 1896, 1899, 1899, 1896, 1896, 1895, 1896, 1897, 1896,
                1893, 1897, 1894, 1894, 1897, 1894, 1897, 1898, 1899, 1899, 1898,
                1895, 1894, 1894, 1894, 1899, 1899, 1896, 1896, 1894, 1899, 1894,
                1898, 1896, 1898, 1897, 1896, 1896, 1897, 1898, 1895, 1897, 1896,
                1897, 1894, 1899, 1896, 1896])
```

```
In [11]: np.count_nonzero( movie_years < 1900 )

Out[11]: 49
```

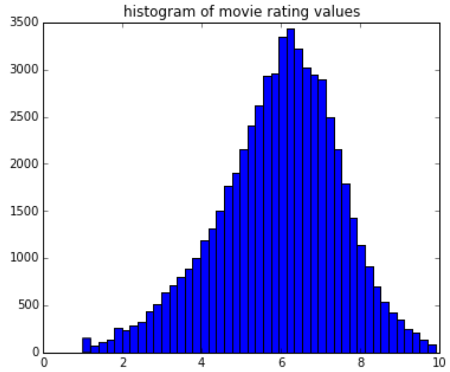
In [12]:
```
plt.rcParams['figure.figsize'] = (6.0, 5.0)

for ColName in ['Length', 'Rating', 'Votes']:  ### Movies.columns:
    ColValues = Movies[[ColName]].values
    if (ColName == 'Length'):
        ColValues = ColValues[ ColValues<240 ]  # ignore any movies over 4 hours
    if (ColName == 'Votes'):
        ColValues = ColValues[ ColValues<250 ]  # ignore any movie with over 250 votes
    plt.hist( ColValues, bins=100 )
    plt.title( ColName )
    plt.show()
```







In [13]:
```
plt.hist( Movies[['Rating']].values, bins=45 )
plt.title('histogram of movie rating values')
plt.show()
```



What the program's output should look like

The program should output lines consisting of predicted rating values for the test set, like:

```
5.6
7.2
9.1
...
8.4
```

Each rating should be numeric value of the form X.Y, where X and Y are single digits.