# Return To Normality: A Statistical Analysis of the Netflix Training Dataset

Gainey, Seth Rylan
Southern Methodist University;
Space and Naval Warfare Systems Center Charleston

The movie ratings in the Netflix Prize training dataset represent the largest data release provided to the public from the internal records of a for-profit company; it contains the anonymized rating activities of 480,189 customers over 7 years (October 1998–December 2005). An analysis of this data for patterns following a normal probability distribution indicates that the majority of customer activies follow power law or mixture model distributions, but the average movie rating per customer and the average rating per movie conform to Gaussian curves. In addition, the mean of this customer rating curve increases for each year, implying systemic changes in Netflix customer base or customer behavior.

Categories and Subject Descriptors: G.3 [**Probability and Statistics**]: Distribution functions—*Normal Distribution*

Additional Key Words and Phrases: Kolmogorov-Smirnov, Non-Parametric, Long Tail, Netflix

## 1. MOTIVATION & BACKGROUND

> "[N]ature has established patterns originating in the return of events,
> . . . but only for the most part."
>
> —*Leibniz in response to Bernoulli, 1703*

Choice of dataset for normality tests was based on 1) preference for "non-natural" data and 2) preference for large datasets (>1,000,000 data points). While normal probability distributions are typical throughout nature [**?**], they are primarily associated with physical attributes and, in some cases, social structures. The chests sizes of Scottish soldiers (cf. Quetelet) will conform to random—rather than systemic—variation, but their incomes, personal library sizes and film predilections will tend towards non-normal distributions.

A large dataset was chosen to not only validate the Law of Large Numbers, but also to test the limits of data manipulation on the author's computing systems.

The Netflix dataset has been used for academic and competitive projects related to predication and data-mining methods, but never to analyze the distribution of data provided by processes suspect to human reliability. Answers to these questions can lead to insight on the probabilistic nature of rating systems and the vagaries of human opinion.

## 2. DATASET INTRODUCTION

In 2006, Netflix announced the $1-million Netflix Prize, an open competition for improving their rating prediction algorithm. In an unprecedented act of data-sharing, the company released 100,480,507 movie ratings, created by 480,189 customers. This data—when converted to a normalized relational format (see steps in §4)—amounts to 5GB of data. Shortly after release, researchers found by combining the anonymized Netflix data with public IMDB profiles, one could interpolate the real world identities of the Netflix customers and information related to their movie rentals [**?**] including information such as political affiliation and sexual orientation [**?**]. The dataset has since been withdrawn due to legal liability, but is still available from third parties [**?**].

## 3. QUESTIONS & RESULT SUMMARY

For each metric in table I, this paper presents a Kolmogorov-Smirnov test of its conformation to a normal distribution, along with a calculation of the mean, standard deviation; skew and kurtosis are provided for normal distributions, and the median is provided for non-normal distributions.

## 4. METHODOLOGY

### 4.1 Data Preprocessing

The steps for preparing the environment and importing the Netflix dataset into a relational format require 8–12 hours, depending on the speed of the computer and network connections. The compressed dataset [**?**] accounts for ~2GB. After extraction, there are four separate data types, but only two are needed for the purposes of the research objectives: movie titles (n = 17770, stored in a single text file with title and release year) and the training set (one file per movie, listing customer ID's, ratings and date of rating). Dates are in format YYYY-MM-DD, ratings range from integer values 1

---

This work is submitted in partial fulfillment of the requirements of EMIS7300, taught in Fall 2012 at Southern Methodist University under the guidance of Dr. Thomas Siems.
Author's addresses: S. Gainey, Systems Engineering Department, Bobby Lyle School of Engineering, Southern Methodist University; Enterprise Systems Engineering, Space and Naval Warfare Systems Center Charleston.
Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies show this notice on the first page or initial screen of a display along with the full citation. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers, to redistribute to lists, or to use any component of this work in other works requires prior specific permission and/or a fee. Permissions may be requested from Publications Dept., ACM, Inc., 2 Penn Plaza, Suite 701, New York, NY 10121-0701 USA, fax +1 (212) 869-0481, or permissions@acm.org.
© 2009 ACM 0730-0301/2009/12-ART106 $15.00
DOI 10.1145/1559755.1559763
http://doi.acm.org/10.1145/1559755.1559763

| METRIC | EXPLANATION | RESULT |
|---|---|---|
| Average Rating by Customer | Probability of a user with an average given rating of r. **What is an average customer?** | Normal, with $\mu = 3.674$ and $\sigma = 0.473$. |
| Average Rating by Movie | Probability of a movie with an average given rating of r. **What is the threshold for a "good" movie?** | Near-normal (bimodal), with a single Gaussian approximate at $\mu = 3.228$ and $\sigma = 0.526$. |
| Number of Ratings per Customer | Probability of a customer rating r number of movies. **What is a "prolific" customer?** | Non-normal with long tail; $\mu = 210$ and median $= 96$. |
| Number of Ratings per Movie | Probability of a movie having r number ratings. **What is "a lot" of ratings?** | Non-normal with long tail; $\mu = 5655$ and median $= 561$ |

RESEARCH OBJECTIVES

Table I.

```
CREATE DATABASE `netflix`;
/*!40100 DEFAULT CHARACTER SET latin1 */
USE `netflix`;
CREATE TABLE `movies` (
  `id` int(5) NOT NULL DEFAULT '0',
  `year` int(4) DEFAULT '0',
  `title` varchar(255) NOT NULL DEFAULT '',
  PRIMARY KEY  (`id`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1;
CREATE TABLE `probe` (
  `movie_id` int(5) NOT NULL DEFAULT '0',
  `customer_id` int(6) NOT NULL DEFAULT '0',
  KEY `movie_id` (`movie_id`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1;
CREATE TABLE `qualifying` (
  `customer_id` int(6) NOT NULL DEFAULT '0',
  `date` date NOT NULL DEFAULT '0000-00-00',
  `movie_id` int(5) NOT NULL DEFAULT '0',
  KEY `movie_id` (`movie_id`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1;
CREATE TABLE `ratings` (
  `movie_id` int(5) NOT NULL DEFAULT '0',
  `customer_id` int(6) NOT NULL DEFAULT '0',
  `rating` int(1) NOT NULL DEFAULT '0',
  `date` date NOT NULL DEFAULT '0000-00-00',
  PRIMARY KEY  (`movie_id`,`customer_id`),
  KEY `date` (`date`),
  KEY `movie_id` (`movie_id`),
  KEY `customer_id` (`customer_id`),
  KEY `rating` (`rating`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1;
```

Fig. 1. SQL DDL for schema creation

to 5, and customer ID's range from from 1 to 2649429, with gaps (480,189 total customers).

4.1.1 *Software Versions Used.* Available without restriction on most platforms.

—Python 2.7.3 (32-bit)
—Ruby 1.93 (32-bit)
—MySQL 5.5 (32-bit)
—NumPy 1.6.2 (32-bit)
—matplotlib 1.2.0 (32-bit)
—SciPy 0.11.0 (32-bit)

4.1.2 *MySQL Configuration.* Default values shown where exists.

```
import MySQLdb

db = MySQLdb.connect("127.0.0.1","root","pass","netflix" )
movieCount, errorCount, cursor = 0, 0, db.cursor()

for n in range(1, 17771):
    try:
        result = cursor.execute("LOAD DATA INFILE '" + '/path/ratings.'\
                + str(n).zfill(7) +  "' INTO TABLE ratings FIELDS TERMINATED\
                BY ',' LINES TERMINATED BY '\n'")
        movieCount += 1
    except MySQLdb.IntegrityError:
        errorCount += 1
        print 'Error #' + str(errorCount)
db.commit()
db.close()
```

Fig. 2. Python code for final database load

—query_cache_size = 0 − > 100M
—query_cache_limit = 3M
—myisam_sort_buffer_size = 69M -> 200M
—bulk_insert_buffer_size = 100M
—join_buffer_size = 100M
—key_buffer_size = 55M -> 600M
—read_buffer_size = 64K -> 200M
—sort_buffer_size = 256K -> 500M
—disable InnoDB/transaction support by using myisam storage engine.

4.1.3 *Schema Defintion.* The schema used for storing the normalized dataset is based on versions used by other researchers [**?**]; the specific code used for this example can be seen in Figure 1.

4.1.4 *Import Optimization.* The 17770 ratings files must be flattened and converted into raw SQL format and then imported using a raw DATA LOAD. The final import can be reduced from over 28 hours to ~3 hours using this method (see code in Figure 2).

## 4.2 Aggregate Data Queries

Despite the work done in the previous steps to optimize the dataset for querying, each SQL aggregate functions used in this section can take 45–150 minutes to complete.

(1) Average Rating by Customer

*select avg*(rating) *from* ratings *group by* customer_id;

(2) Average Rating by Movie

*select avg*(rating), title *from* ratings *inner join* movies *on* movies.id=ratings.movie_id *group by* movie_id;

(3) Number of Ratings per Customer

*select count*(*), customer_id *from* ratings *group by* customer_id;

(4) Number of Ratings per Movie

*select count*(*), title *from* ratings *inner join* movies *on* movies.id= ratings.movie_id *group by* movie_id;

```
#!/usr/bin/env python
import csv
import numpy as np
import matplotlib.mlab as mlab
import matplotlib.pyplot as plt
import scipy.stats as stats
from array import array

values = array('f', [])
for row in csv.reader(open('avg_ratings.csv','rb')):
    values.append(float(row[0]))

mu, sigma, numbins = np.mean(values), np.std(values), 100
normed_data=(values-mu)/sigma
print(stats.kstest(normed_data,'norm'))

# histogram of the data
n, bins, patches = plt.hist(values, numbins, normed=True,\
                            facecolor='blue', alpha=0.75)

# add a 'best fit' line
y = mlab.normpdf(bins, mu, sigma)
l = plt.plot(bins, y, 'r--', linewidth=1)

...

plt.show()
```

Fig. 3. Code Sample for Data Manipulation

### 4.3 Processing, Analysis, Statistics & Graphics

Normalization of the data array for the Kolmogorov-Smirnov (K-S test is performed by mapping the function

$$x_{i,norm} = (x_i - \mu)/\sigma \qquad (1$$

to the set of values used to create the probability density function.

The one-sample Kolmogorov-Smirnov test is used in eacl distribution to test the goodness-of-fit of the normalized data to a Gaussian curve of $\mu = 0, \sigma = 1$, using the equation:

$$D = max_{i \leq i \leq n} \left( F(Y_i) - \frac{i-1}{N}, \frac{i}{N} - F(Y_i)) \right) \qquad (2)$$

where $F() = \phi()$, the standard normal distribution.

The null hypothesis $H_0$ is that the distribution came from a normally distributed population; $H_a$ is that the data does not follow the normal distribution. All tests use significance level $\alpha =$ .05 [**?**].

Functions from the NumPy, SciPy and matplotlib libraries produce the calculated values and graphics in later sections. A simplified version is shown in Figure 3.

## 5. RESULTS

5.0.1 *Note on histogram normalization.* Where not otherwise stated, the number of histogram bins are calculated based on the code in Algorithm 1; when the number of bins is greater than the domain, this approach can produce histogram bars with a height greater than 1.0, however the reader can verify through trapezoidal integration that the area of the PDF curve totals to 1.0. Each PDF of a normal distribution has an overlayed plot of an ideal Gaussian curve (red dashed line) from the calculated $\mu$ and $\sigma$ of the dataset.

**ALGORITHM 1:** Histogram Normalization Algorithm

**Input**: Non-normalized values
**Output**: Weight value such that integrated area of bins equals normalized area
normalizedarea, minval, maxval = 1, min(values), max(values;
binweight = normalizedarea * numbins / (maxval-minval) / values.size;

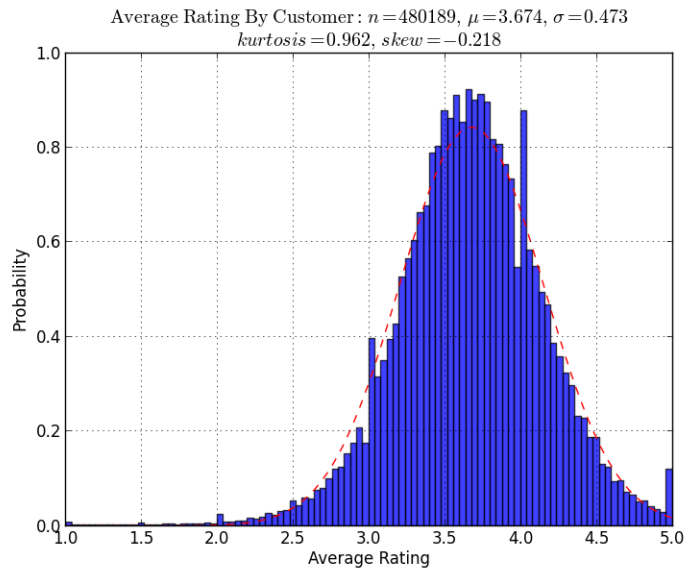


Fig. 4. Probability Density Function (PDF) of the Average Rating Score Given by Customer

### 5.1 Average Rating by Customer

The average of all given ratings by each customer (Figure 4) provides the closest approximation to a normal probability distribution found in the Netflix dataset. Peaks exist at each integer value ([1,2,3,4,5] $\epsilon \mathbb{Z}$), preceeded by dips at values immediately below; the result of customer habits to assign uniform ratings. The distribution is slightly leptokurtic (i.e., more "peaky" than true normal), indicating a central tendency.

From the K-S test, D = 0.0200 ($<$ $\alpha$ = .05); $H_0$, that the data is normally distributed, is not rejected.

### 5.2 Average Rating by Movie

The average of all given ratings to each movie (Figure 5) comes close to a normal distribution, with slight negative skew (i.e., right "leaning"). There is lower than normal probability for a movie to have a rating between 3.2 and 3.4, indicating a bimodal distribution. The author recommends the application of Gaussian Mixture models for future work, but there are two possible explanations presented.

(1) A cursory analysis suggests that customers tend to rate movies as exclusively "good" or "bad"; there are many movies rated as 4's and 5's, and many movies rated as 1's, 2's, and 3's; but relatively few movies rated as "ok" at 3's and 4's. In other words,

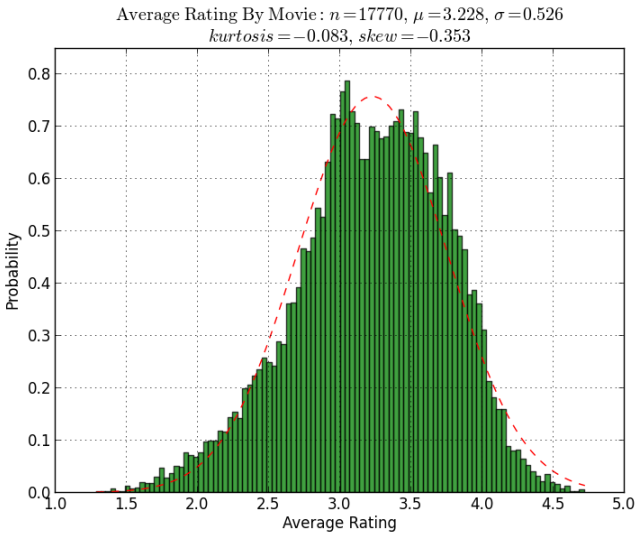


Fig. 5. Probability Density Function (PDF) of the Average Rating Score by Movie

| YEAR | RATINGS ($n$) | $\mu$ | $\sigma$ | SKEW | KURTOSIS |
|---|---|---|---|---|---|
| 1999 | 2,178 | 3.337 | 1.161 | -0.305 | -0.646 |
| 2000 | 924,443 | 3.365 | 1.124 | -0.316 | -0.570 |
| 2001 | 1,769,031 | 3.391 | 1.104 | -0.336 | -0.521 |
| 2002 | 4,342,871 | 3.382 | 1.103 | -0.351 | -0.502 |
| 2003 | 9,985,337 | 3.406 | 1.100 | -0.377 | -0.478 |
| 2004 | 30,206,574 | 3.595 | 1.079 | -0.501 | -0.313 |
| 2005 | 53,250,073 | 3.676 | 1.074 | -0.554 | -0.256 |

TIME-SERIES METRICS

Table II.

customers tend to rate movies for which they hold strong opinions.

(2) When analyzing the time-series data (table II), there is a distinct yearly trend towards higher ratings with less variation. Systemic changes in customer-base or behavior likely cause this pattern, and the collapse of these time-domain stratifications, combined with a increase in number of customers or rating frequency, may produce the bimodal distribution.

Despite this deviation, from the K-S test, D = 0.0295 ($< \alpha$ = .05); $H_0$, that the data is normally distributed, is not rejected.

## 5.3 Number of Ratings per Customer

The number of ratings per customer (Figure 6) has a arithmetic mean of 210 and a median of 96. The Long Tail of the graph (values over 1000 are not shown, but the maximum number of ratings is 17653) indicates a more exotic distribution. Lognormal and rayleigh (Weibull special case, with $\delta = 2$) PDFs—shown in blue and red respectively—were fitted using the SciPy library but fail to approximate the distribution. The fitting method used by SciPy weighs long tail values and peak values equally, and therefore the curve fits appear non-optimal in Figure 6.

From the K-S test, D = 0.245 ($> \alpha$ = .05); $H_0$, that the data is normally distributed, is rejected.

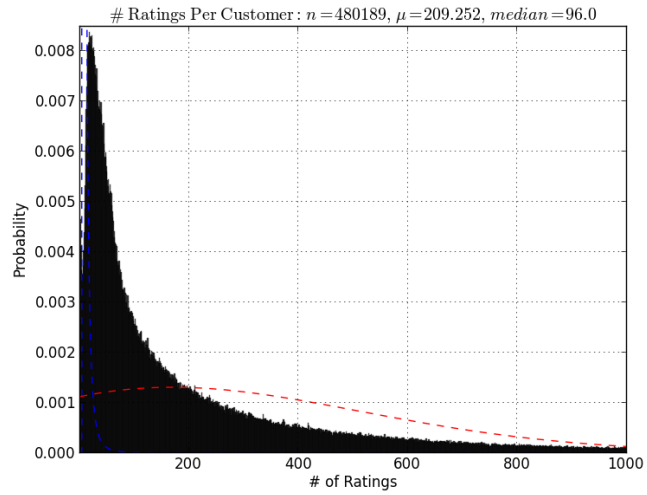


Fig. 6. Probability Density Function (PDF) of the Number of Ratings per Customer. Note: this figure uses a unity number of bins (# of bins equal to x-axis domain and merged); therefore the y-axis probabilities are normalized to 1 for each integer value.

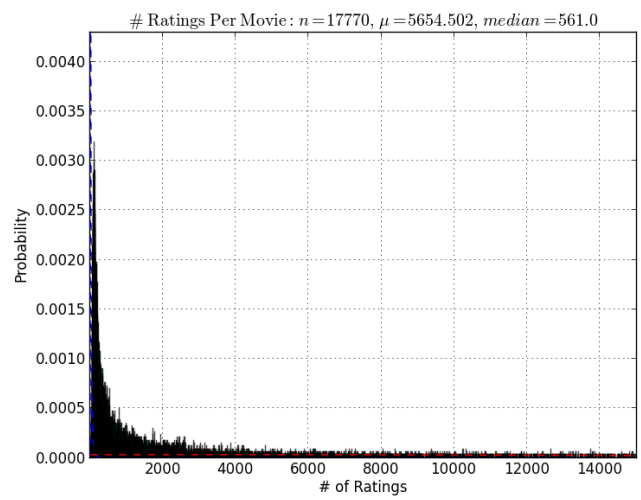


Fig. 7. Probability Density Function (PDF) of the Number of Ratings per Movie. Note: this figure uses a unity number of bins (# of bins equal to x-axis domain); therefore the y-axis probabilities are normalized to 1 for each integer value.

## 5.4 Number of Ratings per Movie

The number of ratings each movie receives (Figure 7) does not follow a normal distribution; the arithmetic mean is 5654 and the median is 561. The lognormal and rayleigh PDF curves of best fit are shown in blue and red. Values above 15,000 are not displayed, but the Long Tail extends into 100,000's: *The Patriot* (200,832), *Independence Day* (216,596) and *Miss Congeniality* (232,944) are the most rated movies from the dataset.

From the K-S test, D = 0.369 ($> \alpha$ = .05); $H_0$, that the data is normally distributed, is rejected.

Received September 2008; accepted March 2009

## 6. CONCLUSION & REFLECTION

### 6.1 Overview

Analysis of the Netflix dataset has shown that most derived attributes related to movie ratings will follow power law distributions, and discovering normal distributions in the data has been a challenging problem to solve, especially given the scale of the data. The investigation presented in this paper omits several failed attempts at modelling the dataset.

In general, distributions of data with no fixed minimum or maximum values will produce non-normal distributions. This explains why physical limits produce normal distributions of body size/weight, but not income, and why only the distributions of ratings (limited in range from 1.0 to 5.0) followed a normal curve. There exist inherent limits to the height of the world's tallest person, but not to the wealth of the world's richest; nor, apparently, to the number of films he or she is capable of rating.

### 6.2 Comments on Distribution Mixture Models

For the distributions which did not conform to normal, lognormal or rayleigh distributions, it is suspected that there is a mixture of power law and normal-like distributions to account for the initial peak and very long tail of the data.

### 6.3 Comments on Outliers

Some of the outliers in the plot of highly rated movies (not shown) include the extended editions of *Lord of the Rings* trilogy, at respective average ratings of 4.7026, 4.7166, and 4.7233. While the standard editions of these films did nearly as well, a customer is unlikely to watch extended version sequels without first enjoying the standard edition prequels, and is therefore more likely than to enjoy the movie. Notably, the *Star Wars* releases do not show the same pattern, for reasons left as an exercise for the reader.

### 6.4 Reflection on Technical Challenges

Data analysis proved very challenging; the more robust Anderson-Darling and Shapiro-Wilk normality tests were discarded due to difficulty in implementation. Methods for visualizing time-series distributions also failed to produce adequate results.

In many cases, it was found that a data structure useful for quickly producing graphs of 17,770 values would dissapoint at 100,000+, because the time to completion grew with an exponential relationship to $n$. The time to re-write and re-run the algorithm with an optimized data structure required far less time than the original program would have needed to complete. The lessons of Knuth come to mind:

> "We should forget about small efficiencies, say about 97% of the time: **premature optimization is the root of all evil**. Yet we should not pass up our opportunities in that critical 3%. A good programmer will not be lulled into complacency by such reasoning, he will be wise to look carefully at the critical code; but only after that code has been identified."
>
> —*Donald Knuth*, emphasis added [**?**]