Predicting Success of Fiction Literature Film Adaptation with Artificial Neural Networks

Mary Moser
Dept. of Computer Science
CMPSCI 383 (Artificial Intelligence)
UMass Amherst
Amherst, MA
mmoser@umass.edu

Isaac Vawter
Dept. of Computer Science
CMPSCI 383 (Artificial Intelligence)
UMass Amherst
Amherst, MA
ivawter@umass.edu

***Abstract*— Film adaptation from literature is a complex process that would benefit greatly if its success could be estimated before the adaptation is attempted. In this study an Artificial Neural Network (ANN) is used to predict the success of a film adaptation by observing data about the work of literature it is based on. This ANN uses the popularity and subject matter of a work of literature to not only predict the financial success of its film adaptation, but also the artistic success as determined by its audience ratings and award nominations. To accomplish this, the data from hundreds of books and their film adaptations will be gathered and used to train an ANN by supervised learning and the Backpropagation Algorithm. The ANN will then associate certain literature popularity and subject matter characteristics with successful film adaptations. The results of this study show a strong relationship between the success of a film adaptation and the popularity and subject matter of the book it was based upon.**

***Keywords*— Artificial Neural Network (ANN), Multi-Output Classification, Supervised Learning, Backpropagation**

I. Introduction

These days, it seems that Hollywood is turning every popular book into a motion picture. Fiction works, especially, give filmmakers a nice creative template to work from, and often a pre-assembled audience of devoted fans to celebrate (or dread) the chance to see a beloved story and cast of characters on the big screen. But how can filmmakers, and readers, predict whether a book would lend itself well to a film version, or is better left on the page?

Prediction models for forecasting the financial success of films is a common but difficult topic in statistics and artificial intelligence. An even more challenging, and often unaddressed, problem is assessing not only a film's financial performance (that is, how much filmmakers can profit from it), but its artistic performance and ability to resonate with audiences. In our research, we used artificial intelligence techniques to analyze previous trends and correlations between the success and themes of a fiction book, and the financial and cultural success of respective film adaptations. We implemented our own artificial neural network (ANN) to classify book-related input variables to film-related output variables. Given a training data set, the ANN was trained with backpropagation on one or two hidden layers, then its learned classification capabilities were applied to make predictions on a smaller set of testing data.

II. Goals

As previously suggested, our research aimed to accurately predict the quality and financial return of a movie based on information from its original literary source. The original

intention was to take into account multiple types of literature, including prose, poetry, and plays. However, since it was difficult to find a sizeable and usable mapping of all forms of literature to their film versions, we narrowed our focus to only consider books adapted into films. Our dataset was further narrowed down to include only fiction works made into films.

III. Related Work

Our original inspiration for the assignment came from an article in the June 2015 issue of Time Magazine [1]. The article briefly discussed how Big Data was used to predict which Hollywood films would be ideal for a remake. Intrigued by this study, we chose to research a similar topic, but one that related data from two forms of media instead of one: whether or not a book would be ideal for a film adaptation.

Forecasting box office success is a popular research topic in statistics and artificial intelligence, with a variety of prediction models being employed [2-5]. Our study was largely based on two major sources: “Predicting box-office success of motion pictures with neural networks” [2], and “Forecasting box office success with BP neural networks”[3]. Both studies treated the prediction problem as a classification problem [2], training a multi-layered perceptron network to classify input variables to one of multiple classes of an output variable. The former study used a multi-layer perceptron (MLP) neural network, while the latter applied a multi-layer backpropagation (MLBP) neural network. In order to improve the simplicity, speed, and accuracy of their prediction models, variables were presented to and output by the network in binary form. Furthermore, continuous (and thus potentially infinite) values were discretized into a limited number of classes, rather than attempting to pin-point exact monetary values.

For our research, we borrowed the above mentioned use of multi-layered perceptions for our ANN structure. We also implemented supervised learning using backpropagation, since the MLBP model from the second study yielded more accurate results that the MLP method of the first. For simplicity, both independent and dependent variables are represented in binary, converting continuous ranges to discrete partitions as necessary. Unlike the previous research, however, inputs of our network are mapped to multiple categories of outputs, not just box office revenue. Thus, our prediction model used a multioutput-multiclass approach to classification [6].

Keeping with our original theme of book-to-movie adaptations, independent (input) variables were selected from book metadata, while dependent (output) variables were selected from data related to the respective film versions of each book. Moreover, since our goal was analyze both the quantitative and qualitative success of a film, we supplemented box office revenue with three additional target labels for our output layer: the estimated production costs, the average viewer rating, and the number of award nominations.

IV. Data Mining Methods

This study required two major endeavors, training data acquisition followed by data classification through supervised learning by an ANN. In order to gather data on appropriate books and movies, a list of books and their film adaptations was

needed. This list was acquired via web-scraping from four Wikipedia.org web pages[1]. Java web-scrapers[2] were used to pull the HTML source lines from these web pages. These HTML source lines were then scraped by pattern matching to identify table rows, table columns, and the book and movie titles within table columns. Certain character patterns were identified before each book title and each movie title. This allowed those titles to be extracted and associated with each other via the table row identification. These book and movie titles were then written to a CSV file and manually proofread for erroneous or ambiguous data. This Book-to-Movie Index File[3] of books and their film adaptations provided a reference point that enable targeted web scraping for relevant book and movie training data.

A. Data and Variable Definitions - Inputs

The data gathered for books was broken into five categories: Amazon Bestseller, Amazon Rating, Amazon Review Count, Age Group, and Keywords. The Bestseller, Rating, and Review Count were chosen as indicators of popularity, fame, and quality. If a book is well known and well regarded, it follows that a film adaptation of it will be well received and more successful. Age Group and Keywords were chosen as subject matter indicators. If a book possesses particular themes, settings or target audience, it could have a dramatic influence on its film adaptation's production costs and how likely it is to attract viewers.

1) Input Data Acquisition

All input data was gathered from Amazon.com [11] product web pages via web scraping. Unfortunately, there is no Amazon.com API that provides all the desired categories of the input data, so custom Java web scrapers[4] had to be designed to acquire this data.

These web scrapers were designed to find the associated input data for every book listed in the Book-to-Movie Index File. To accomplish this, it reads the Book-to-Movie Index File one line at a time and parses the book title into a String. This String is then used to create an Amazon book search URL. Upon establishing a connection to this URL, the HTML source lines are pulled and the URL from the first search result is parsed out and used to establish a connection to the book's Amazon product webpage associated with the original book title. The web scraper then pulls the product page's HTML source lines and uses String pattern matching to find and parse relevant input data and write it to a CSV file.

Once all the book input data had been scraped from Amazon, it was then filtered. Books that had no bestseller, rating, and review information were removed because that indicated that the book title either had no search results, or no reviews on its Amazon book webpage. It was decided that if a book possessed no review or rating information, it would just confuse our data and not help in training our ANN. The resulting filtered book data file was then compared to the filtered movie data file to

[1] https://en.wikipedia.org/wiki/List_of_fiction_works_made_into_feature_films_%280-9%29_%26_%28A-C%29

https://en.wikipedia.org/wiki/List_of_fiction_works_made_into_feature_films_%28D-J%29

https://en.wikipedia.org/wiki/List_of_fiction_works_made_into_feature_films_%28K-R%29

https://en.wikipedia.org/wiki/List_of_fiction_works_made_into_feature_films_%28S-Z%29

[2] Attachment 1 – WikiScraper.java, Attachment 4 – BasicScraper.java

[3] Attachment 13 - Book-To-Movie Index.csv

[4] Attachment 2 – AmazonBookScraper.java, Attachment 4 – BasicScraper.java

eliminate any books and movies that were present in one list, but not the other. The resulting list of 505 books, movies, and associated data from this final comparison was ultimately used as the Dataset [1] for all ANN supervised training and testing conducted during this project.

2) Input Data Representation

The input data for the ANN was discretized to provide a quantifiable number of input classes. A total of forty-three binary inputs are used to represent a book's data.

Amazon Bestseller data is represented as a single binary input with a 1 representing that the book is an Amazon #1 Bestseller, and a 0 if not.

The Amazon Star Rating for a book is represented with five mutually-exclusive binary inputs representing classes for this category. These five inputs represent ranges of Amazon star-ratings as indicated in *Table 1*.

Binary Index	Amazon Star Rating Range
0	1.0 – 1.9
1	2.0 – 2.9
2	3.0 – 3.9
3	4.0 – 4.9

Table 1: Amazon Rating Binary Index Table

A book's Review Count (RC) on Amazon.com is represented with fifteen mutually-exclusive binary inputs representing classes for this category. These fifteen inputs represent ranges of a logarithmic scale as indicated in *Formula 1*. A book's total review count on Amazon is applied to *Formula 1* to indicate which class it falls into. The largest binary index has no upper limit to include uncommonly massive review counts.

$$RC\ Binary\ Index = Integer\ \log_2(RC)$$

Formula 1

The Age Group for a book is represented with three mutually-exclusive binary inputs representing classes of this category. These three inputs represent one of either Child, Teen, or Adult literature. Amazon's book description and book bestseller categorization is used to determine if a book belongs in Child or Teen literature, otherwise it is considered Adult literature.

Binary Index	Keyword
0	“sci-fi”
1	“crime”
2	“classic”
3	“adventure”
4	“comedy”
5	“mystery”
6	“fantasy”
7	“romance”
8	“tragedy”
9	“wizard”
10	“drama”
11	“satire”
12	“tech”
13	“vampire”
14	“history”
15	“myth”
16	“western”
17	“space”
18	“animal”
19	“voyage”

Table 2: Keyword Binary Index Table

Book Keywords are represented with twenty binary inputs. These twenty inputs represent any of the substrings from *Table 2*

[1] Attachments 9 - BookDataFinal.csv, Attachment 10 - MovieDataFinal.csv

that are present in Amazon's book description and book bestseller categorization.

B. Data and Variable Definitions - Outputs

In the previous studies we analyzed, instances were classified into only one output label -- namely, the box office revenue of a movie. As previously mentioned, however, we wished to consider multiple outputs for measuring the success and quality of a book-based movie. Thus, our methods differed from previous work because we treated the study as a multi-output classification problem, rather than predicting only a single target output [6].

The target labels used were: 1) viewer ratings, 2) box office revenue, 3) estimated production costs (to compare against box office revenue), and 4) nomination count. Gross revenue and production costs were analyzed to measure the financial (quantitative) success of a film adaption, while viewer rating and nomination count helped assess the qualitative success of a film and how well-received it was among audiences. A more detailed explanation of each label is described below.

Viewer Ratings: In addition to financial revenue, a movie's success can also be measured by its popularity among viewers. We chose to represent the degree of popularity by its average user rating on IMDb.

Box Office Revenue: Similar to previous work, we choose to measure the success of a movie based on its gross box office revenue, as listed on IMDb.

Estimated Production Costs: While not a direct measure of a movie's success, we believed that predicting the budget cost of a movie would also be helpful in determining whether or not a book should be adapted into a film. For a filmmaker, predicting gross revenue alone may not be a full indicator of the financial return of a film; if the production costs are too high relative to the box office income, a studio may ultimately lose money on a movie. Thus, the ability to predict production costs is equally important in order to predict the total net profit of a film.

Nomination Count: The number of awards and nominations that a film receives is a popular indicator of its quality and how well it resonated with audiences. Note that "nomination count" hereafter refers to the total number of wins and no-win nominations for film awards. We wished to take all nominations, including no-wins, into account, because we wanted to disregard as much as possible any competition with other films of the same time period.

1) Output Data Acquisition

All movie information was pulled from IMDb.com [12]. To help retrieve data, we employed IMDbPY [7], an open-source Python API for extracting information from the IMDb website. With the exception of nomination count, which required a custom-programmed web scraper to obtain[1], all desired information of a film could be directly pulled by using the pre-packaged methods from the API[2].

Using the Book-to-Movie Index File[3] as a reference, we used IMDbPY to search for each unique movie title associated with a book. Since our goal was to predict a book's potential for movie success, we hoped to analyze only the most "successful" film adaptation. Thus, if a book was adapted to multiple movies, we

[1] Attachment 3 - IMDbScraper.java, Attachment 4 - BasicScraper.java
[2] Attachment 11 - imdbInfo.py, Attachment 12 - writeToCSV.py
[3] Attachment 13 - Book-To-Movie Index.csv

selected the first closest search result for each unique title (IMDb orders search results by closest match, which are then sorted by most popular matches.) Then, from our list of best uniquely-titled movies, we selected the highest rated film. If the information for that film was incomplete -- for instance, many movies returned no gross revenue or budget costs -- we selected the most popular film containing the most complete data. In cases where our program still failed to find a movie with complete information, or otherwise pulled a "poor" example, we either hand-selected a film adaption to represent the output, or discarded the instance from our data set.

Given a specific movie, IMDbPY could easily extract information for gross box office revenue, estimated budget, and rating. If no rating was found, a default value of 0 stars was used. Also, because gross revenue and estimated budget could potentially return multiple values (typically, one or more for each country), particular care had to be taken in selecting which value to use. In general, we aimed to extract the value with the highest monetary amount in USA dollars. In the case of gross revenue, since most films on IMDb list the "Worldwide" gross revenue, this was most often the value used for a particular film. Otherwise, the highest "USA" or "Non-USA" value was pulled.[1]

Unfortunately, the IMDbPY API did not include built-in support for extracting award information for a movie. Instead, we used the API to retrieve the unique IMDb ID for each film, which is included in its respective URL string. From there, we implemented our own Java web scraper[1] to parse the award-nomination listing on a movie's award page. The number of wins and the number of no-win nominations were parsed and summed, yielding the total nomination count. If the specific HTML element containing award information could not be found, or did not include any numeric values, then we assumed that the film received no award nominations, and returned a default nomination count of 0.

2) Output Data Representation

Once all relevant information for each film was obtained, and poor or incomplete instances were eliminated from our data set, we divided each variable into a set of possible classifications, represented by mutually exclusive binary variables.

Viewer Ratings are classified into 9 possible categories, each represented as binary variable. These classes correspond to IMDb's user rating system, whereby a movie's listed rating is the decimal average of all user ratings (ranging from 0 to 10 stars). All shown in *Table 3*, rating classes are equal-length partitions between two whole number star ratings, with the exception of the lowest category (0-2 stars).

Binary Index	0	1	2	3	4	5	6	7	8
IMDb Rating	0-2	2-3	3-4	4-5	5-6	6-7	7-8	8-9	9-10

Table 3: Viewer Rating Binary Index Table

Gross box office revenue is represented with 15 binary variables, indicating ranges of financial revenue. *Formula 2* is used to classify gross box office revenue (GBOR) into its appropriate binary representation. To cover all possible monetary values, the lowest binary index represents revenues

[1] Attachment 3 – IMDbScraper.java, Attachment 4 – Basic Scraper.java

from $0-$100,000, while the highest index represents revenues of $1,638,000,000 or above.

$$GBOR\ Binary\ Index = Integer\ log_2(\frac{GBOR}{100{,}000})$$

Formula 2

A logarithmic scale was chosen over a linear scale in order to avoid overgeneralization of low-revenue movies and to reduce the total number of binary variables required.

Estimated production costs (EPC) are similarly represented with 15 binary variables. The partitions are calculated with *Formula 3*, which is identical to that for classifying box office revenue, except the respective values are decreased by a factor of 10. (For instance, the lowest binary input for production cost only ranges from $0-10,000).

$$EPC\ Binary\ Index = Integer \log_2(\frac{EPC}{10{,}000})$$

Formula 3

Nomination count is classified into 1 of the 5 possible binary variables in *Table 4*, each representing a fixed-sized partition of 20 nominations (with the exception of the highest binary index, which has no upper bound).

Binary Index	0	1	2	3	4
Nomination Count	0-19	20-39	40-59	60-79	≥80

Table 4: Nomination Count Binary Index Table

V. Data Classification Methods

An Artificial Neural Network with a Multi-Layer Perceptron structure is used to classify book input data and predict movie output data. The ANN program used in this study is implemented in four Java classes[1] that read and parse the Dataset[1,2], construct an ANN, train and test the ANN, and print out prediction accuracy statistics. The ANN is made up of an Input Layer, a variable number of Hidden Layers, and an Output Layer all made of perceptrons. This ANN uses discrete binary representations of book and movie data. It also takes in all the inputs and generates all the outputs simultaneously to allow interdependencies of output data to influence classification. The ANN is implemented with adjustable settings to help identify optimal network structure and avoid overfitting the training set.

A. Artificial Neural Network Structure

The ANN used in this study has fixed input and output layers, but variable hidden layers as shown in *Figure 1*. This allowed the internal structure of the ANN to be adjusted between iterations of supervised learning. The input layer of the ANN is comprised of 43 perceptrons which map to an array of 43 binary inputs. The structure of the binary input array is displayed in *Table 5*. The output layer of the ANN consists of 44 perceptrons which produce 44 continuous outputs. During training and testing these outputs are compared to an array of 44 binary outputs. The structure of the binary output array is displayed in *Table 6*.

[1] Attachment 5 – ANNMain.java, Attachment 6 – NeuralNet.java, Attachment 7 – Neuron.java, Attachment 8 – TrainingParser.java

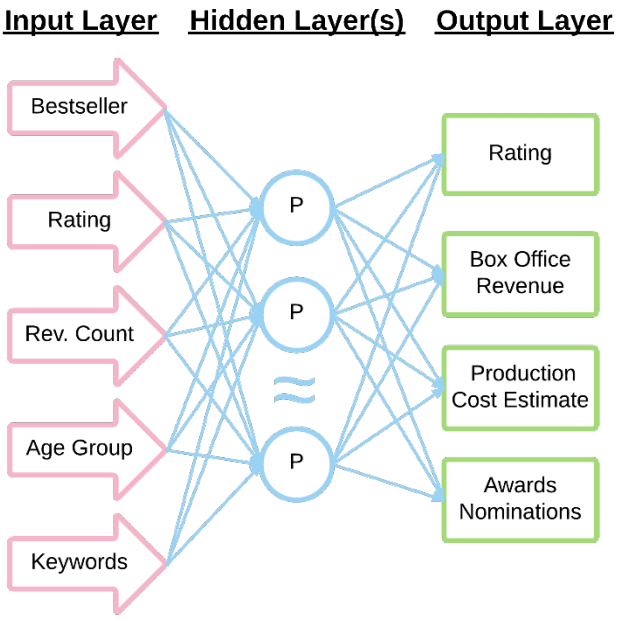

Figure 1: Artificial Neural Network Structure

Input Layer Class	Number of Binary Indices	Indices In Input Array
Bestseller	1	0
Amazon Rating	4	1-4
Review Count	15	5-19
Age Group	3	20-22
Keywords	20	23-42

Table 5: Artificial Neural Network Binary Input Array

Input Layer Class	Number of Binary Indices	Indices In Input Array
IMDb Rating	9	0-8
Box Office Revenue	15	9-23
Production Cost Estimate	15	24-38
Award Nominations	5	39-43

Table 6: Artificial Neural Network Binary Output Array

B. Supervised Learning

Supervised learning by the ANN used in this study was accomplished using the Backpropagation Algorithm. When the Dataset[1] is parsed it is divided into an input data map and an output data map that are both keyed to a specific String representing a book title. After these maps are created, the key-set of book titles is stochastically divided into a training and testing set for supervised learning. This is accomplished by generating a random number between zero and one for each book title, if the number is less than 0.8 it is added to the training set, otherwise it is added to the testing set. This guarantees the entire Dataset[2] is stochastically partitioned with roughly 80% belonging to the training set and 20% belonging to the testing set. The input and output maps along with the training and testing sets are provided to the ANN for supervised learning.

To train the ANN, it iterates repeatedly through the training set and invokes the Backpropagation Algorithm until a specified averaged squared error threshold is reached. This error threshold is adjustable to allow determination of the threshold that achieves maximum accuracy without overfitting. During training each set of book input data indicated by the training set is applied to the Input Layer. The resulting continuous outputs of the Output Layer are then compared to correct outputs, squared error is recorded, and backpropagation is invoked. After the whole training set has been processed by the ANN, the average squared error is calculated to determine if another iteration is required.

The Backpropagation Algorithm used in this study was implemented arithmetically based on a specified learning rate (η). After the input data is processed, each Output Layer

[1] Attachment 9 – BookDataFinal.csv, Attachment 1 – MovieDataFinal.csv

perceptron determines its Squared Error by using the corresponding output data. There are then three basic steps performed by each perceptron for the Backpropagation Algorithm. First the derivative (Δw_i) of the Squared Error (E) with respect to each of a perceptron's input weights (w_i) is determined as shown in *Formula 4*. Second the derivative of the Squared Error (E) with respect to each of a perceptron's inputs (i) is similarly calculated, refer to *Formula 5*, and that adjustment (Δi) is provided to the perceptron that gave that input for its own Squared Error determination. Third all the weights are adjusted as indicated by *Formula 6*. Backpropagation is performed one layer at a time, starting with the Output Layer, and then back through each preceding Hidden Layer.

$$\Delta w_i = \frac{E(w_i + 1 \cdot 10^{-8}) - E(w_i - 1 \cdot 10^{-8})}{2 \cdot 10^{-8}}$$

Formula 4

$$\Delta i = \frac{E(i + 1 \cdot 10^{-8}) - E(i - 1 \cdot 10^{-8})}{2 \cdot 10^{-8}}$$

Formula 5

$$w_i = w_i - \eta \cdot \Delta w_i$$

Formula 6

Testing is performed once the ANN reaches the aforementioned average squared error threshold. To test the ANN each set of book input data from the testing set is applied to the Input Layer and the squared error of the output is recorded as is an output prediction array. The output prediction array is determined by finding the local maximum of partitions of the output that contain mutually exclusive elements. For example as shown in *Table 6*, the first nine indices of the binary output array contain mutually exclusive elements, thus the outputs of the first nine perceptrons of the Output Layer are compared and the index of the largest output is marked with a one in the predicted output array and the rest of the elements in that partition are marked with a zero. This method is applied to each mutually exclusive partition of the predicted output array. After every set of book input data from the testing set has been processed by the ANN the average squared error from testing is calculated.

VI. Performance Metrics

Throughout the supervised learning process, with every iteration of training and testing the average squared error is calculated and recorded. The average squared error was used to determine the optimal threshold to prevent overfitting during training, because it indicated at what point more training increased error during testing.

The predicted output arrays recorded during testing are used to calculate the statistical accuracy of the ANN for each of the four output categories: Viewer Rating, Box Office Revenue, Estimated Production Cost, and Award Nominations. Class offsets are calculated for each output category for every movie in the testing set by comparing the predicted output array and the actual output array. These offsets are recorded in class offset arrays, which are then used to calculate the standard deviation in classes for each output category.

Learning Rate	Hidden Layers	Hidden Layer Sizes	Training Error Threshold	Testing Error	Rating Std. Dev	Box Office Std. Dev.	Cost Est. Std. Dev.	Award Nom. Std. Dev.
0.25	1	12	0.0674	0.0657	1.010	2.324	1.915	1.251
0.1	1	10	0.0674	0.0664	1.163	2.644	1.609	1.441
0.05	1	8	0.0674	0.0670	1.126	2.892	2.068	1.384
0.25	2	10, 8	0.0674	0.0660	0.962	2.934	2.024	1.326
0.25	2	8, 6	0.0674	0.0660	1.196	2.625	2.065	1.427
0.8	2	6, 4	0.0674	0.0698	1.221	2.546	1.896	1.593
Random Guessing					3.782	6.174	6.174	2.186

Table 7: ANN Settings and Prediction Results

VII. Results

Over the course of several testing iterations, it was found that the ideal average squared error threshold for most Hidden Layer structures was 0.0674, which produced a similar or lower average squared error in the testing set and took a reasonable amount of time to train. When the error threshold was dropped to 0.06 or lower the testing error was almost always 0.068 or higher, however, better results were sometimes achieved when using a lower error threshold with a single Hidden Layer. Several different Hidden Layer structures were evaluated and their settings and results are recorded in *Table 8*.

Testing iterations showed that an ANN could predict the success rate of movies from the book input data quite successfully when compared to random guessing. The IMDb Rating, Box Office Revenue, and Estimated Production costs were predicted with the high accuracy, whereas Award Nominations were the most inaccurate and were only marginally better than random guessing.

Our data was determined to be very noisy considering how large the error threshold needed to be to prevent overfitting. An average squared error of 0.0674 is equivalent to an average absolute error of 26.0%, which is means a significant amount of error needs to remain in the ANN in order to accommodate the inconsistencies between book input data and movie output data and suggests a high level of noise.

Additional sources of error in this study could have resulted from bad data collection. Although the book and movie data gathered was proofread and checked for errors, it is still possible the data gathered from Amazon.com may have not been from the correct book or that books and movies with relatively generic names were not properly identified.

The best results we achieved were with a single Hidden Layer with twelve perceptrons. This ANN was determined to be able to predict Box Office Revenue within one order of magnitude with a probability of 0.857. Compared to the more accurate of the two previous works [3], which determined box office revenue within a factor of three with a probability of 0.681, this ANN can predict box office revenue within a factor of three with a probability of 0.497. Their predictions are more accurate, however, their study uses data related to the production of the actual movie for inputs, and ours only used data from the book that a movie was adapted from.

VIII. Conclusions and Future Work

The results show that an artificial neural network can predict the success of a film adaptation by observing data about the book the film is based on. Using the a book's Bestseller, Rating, Review Count, Review Count, and Keyword data from Amazon.com allowed the ANN in this study to predict movie success with high degree of accuracy.

There are several improvement to this study that can be pursued in the future. One of these improvements will be to include the Decade a film was produced in, which could help relate certain Keywords to films made during a specific era. Another improvement would be to correlate the Keywords to the specific book bestseller categories that Amazon.com uses. An additional area of research with this study would be to predict film adaptation success of literature types other than books, such as comics, poems, plays, etc.

Further analysis of the resulting weights after training the ANN could give useful insights into the relationship between specific input and output classes as well as identify subject matter that improves the success of films. Also, increasing the size of the Dataset by finding more sources of movie and book data could create a more even representation of the each input and output class. Continued experimentation will undoubtedly uncover more elements of literature that support successful film adaptation.

References

[1] "What Should Hollywood Remake Next?" *Time Magazine* 25 June 2015: 91. Print.

[2] Sharda, Ramesh, and Dursun Delen. "Predicting box-office success of motion pictures with neural networks." *Expert Systems with Applications* 30.2 (2006): 243-254.

[3] Zhang, Li, Jianhua Luo, and Suying Yang. "Forecasting box office revenue of movies with BP neural network." *Expert Systems with Applications* 36.3 (2009): 6580-6587.

[4] Lee, Kyung Jae, and Woojin Chang. "Bayesian belief network for box-office performance: A case study on Korean movies." *Expert Systems with Applications* 36.1 (2009): 280-291.

[5] Delen, Dursun, Ramesh Sharda, and Prajeeb Kumar. "Movie forecast guru: a web-based DSS for Hollywood managers." *Decision Support Systems* 43.4 (2007): 1151-1170.

[6] "1.12. Multiclass and Multilabel Algorithms." *Scikit Learn*. Scikit-learn Developers, 2014. Web. 14 Dec. 2015.

[7] Malagoli, Alberto. *IMDbPY*. Studio DE&P, n.d.. Web. 28 Nov. 2015.

[8] Rojas, Raul. *Neural Networks: A Systematic Introduction*. Berlin: Springer-Verlag, 1996. Print.

[9] MacLeod, Christopher. "An Introduction to Practical Neural Networks and Genetic Algorithms For Engineers and Scientists." (2010).

Data Sources

[10] *Wikipedia*. Wikimedia Foundation, n.d. Web. 28 Nov. 2015.

[11] *Amazon.com*. Amazon.com, Inc., 2015. Web. 5 Dec. 2015.

[12] *IMDb*. Amazon, 2015. Web. 5 Dec. 2015.