Jeffrey Wang

Period 5

Senior Research Final Paper

**Title:** Predicting the Financial Success of Movies

**Abstract:**

*A large portion of sales from movie tickets are generated during the first week of a movie’s release. Alongside the rise of social media, sentiment and word of mouth for movies can spread quickly on the Internet. The project will obtain user ratings and box office data from datasets found on the Internet to determine how well can ratings predict the financial success of movies. The project will apply machine learning algorithms to classify movies as a success or flop. Using machine learning, I predicted the classification of movies with 62% accuracy.*

**Introduction:**

In recent years, many projects and libraries have been developed for the purpose of machine learning. With machine learning gaining popularity, I would like to try to familiarize myself with machine learning tools and explore ways that machine learning can be used for prediction.

Machine learning has applications in a wide variety of fields. Some of the most common applications lie under the categories of image recognition and speech recognition. One example of image recognition is the MNIST character recognition problem, in which a machine is trained to learn how to distinguish between the numerical digits based on handwritten data. In the task of speech recognition, a program takes in spoken words and represents them as a signal. It looks for patterns in the signal to determine what is being said (S. Sharma).

I am most interested in the application of machine learning for statistical analysis. This is useful for businesses wanting to maximize their profits. For instance, machine learning can be used in performing sentiment analysis. In the context of business, sentiment analysis can be effective in measuring brand loyalty and predicting consumer trends (G. Sharma).

As an example, sentiment analysis is useful for the movie industry. In recent years, roughly 46% of total sales from movie tickets are generated in the first week of a movie’s release in theaters (Kuizinas). With the rising popularity of social media, word of mouth on platforms such as Twitter can greatly impact the success of movies. Researchers from the City University of London analyzed micro-blogging word of mouth (MWOM) on Twitter and found that positive messages about movies outnumbered negative ones. However, the negative messages appeared to carry a larger impact in discouraging people from going to the movies than encouragement from positive messages (Hennig-Thurau et al.).

Another study from University of Southern California used sentiment analysis on tweets to predict the box office success of movies. This study was able to classify movies as a hit, average, or flop with 64.4% accuracy (Jain 312). Review aggregators such as Rotten Tomatoes and Metacritic provide Internet users with the opportunity to submit reviews and ratings on movies. As such, user ratings posted online could also be used as a reflection of sentiment towards movies.

As someone who follows box office news, I figure that sentiment analysis can be applied to predict the popularity of movies. The total box office gross is a useful measure of a movie’s popularity, and I want to see if audience reception towards a movie affects how much money it will make. With this in mind, I wrote a Python program that used machine learning tools in an attempt to see if user feedback in the form of ratings can predict whether a movie will make a profit. I also built a website that displayed a graphical overview of my movie dataset with search and filter functions.

**Background:**

Machine learning is a subset of artificial intelligence that involves training computer systems to automatically learn without being explicitly programmed. With machine learning, computer systems can learn to identify concepts and relationships hidden in data and use what it learns to make valuable predictions.

Generally, machine learning algorithms can be broken down into three types: supervised learning, unsupervised learning, and reinforcement learning. For this project, the focus will be on supervised learning. Supervised learning consists of a target output variable that is to be predicted from a given set of input variables. In classification problems, the variables are categorical, and in regression problems, the variables are numerical (Ray). This project will utilize classification, using ratings, budget, and release date as input variables to predict a category of success or flop, the target variable.

Currently, there are many open-source machine learning libraries that can be used. Popular libraries include TensorFlow, Keras, and scikit-learn (Robinson). TensorFlow was originally developed by researchers from Google, and it provides for numerical computation using data flow graphs. Keras is a neural network library useful for performing deep learning in Python. This project uses scikit-learn, which is a machine learning library designed for Python and works with the Python libraries NumPy and SciPy. Scikit-learn is relatively easy to use and provides a variety of classification algorithms which will be used in this project.

Information about movies can be obtained from online APIs. The most popular movie API is provided by The Movie Database (TMDb). Instead of calling a movie API, however, movie ratings were pulled from a dataset provided by GroupLens Research at the University of Minnesota. GroupLens Research built MovieLens, a research website dedicated to providing movie recommendations to users. They used the TMDb API to generate the dataset. Ratings data is stored under a csv file, which contains over 20 million individual ratings applied to movies released before August 2017.

Many websites, such as Facebook and Twitter, provide APIs that allow users to access data in a formatted structure. However, some websites do not have an API and do not provide download links to save their data locally. If a website contains large amounts of data, it would be time consuming to manually copy and paste all the data to local storage. Thus, web scraping can be used instead.

Web scraping is a technique that involves extracting data from websites. Technology is used to automate the data collection process by loading multiple web pages at once, and then the user can choose to store any relevant data from the web pages to their computer. Some companies, such as Web Scraper, have developed software to conduct web scraping through a browser extension. Since this project involves programming with Python, the BeautifulSoup tool will be used to scrape data from a table. The Urllib library will be used to assist BeautifulSoup in fetching URLs, since the table spans over 50 pages. BeautifulSoup can read through the entire HTML structure of a webpage and search for specific HTML tags, which allows the user to easily locate the data relevant for their needs.

Web development for this project was done on Cloud9.io. As the name suggests, the Cloud9 service is a cloud-based integrated development environment (IDE). This means users can write code anywhere as long as they have a computer connected to the Internet. The user does not have to install any files on their local machine; all coding is done on a web browser. Cloud9 comes preconfigured for a large variety of programming languages, including JavaScript, Node.js, Python, and PHP. Although not used for this project, Cloud9 also allows teams to work on code simultaneously.

It is possible to write code in text editors such as Sublime Text and Notepad++. However, working with a cloud-based service like Cloud9 is preferable for web development since the user can load and test their webpages directly on the server. It is also necessary for running server-side scripting languages such as PHP.

Chart.js is an open source JavaScript based library for creating interactive charts in webpages. It is interactive in the sense that users can hover over individual data points to view their values. Many types of charts are supported, such as bar charts, line charts, pie charts, and scatter charts. Chart.js takes advantage of the HTML5 canvas tag, which is used to draw graphics on a webpage via some JavaScript code. It is through the canvas tag, along with the associated Chart.js declarations, that charts are displayed on a webpage. Chart.js is also flexible in the sense that users have a lot of freedom in customizing charts. The axes, labels, colors, and hover text can all be changed by adding in the necessary code.

JavaScript Object Notation (JSON) is a text format used for storing and exchanging data between browsers and servers. Under JSON, data is stored in an array of objects, with each object having a series of names and associated values. The advantages to using JSON are that it is easy for humans to read and easy for machines to parse. Additionally, JSON is language independent, meaning that text specified under the JSON format can be read by any programming language.

**Development / Techniques:**

This project will be deemed a success if I can use machine learning tools to make a prediction about the financial success of movies in a dataset. Specifically, the Python program that I wrote should be able to load a dataset of movies, train a classifier on a portion of the dataset, and predict a classification for movies on the remaining portion of the dataset.

As an overview, I wrote a Python program that uses the scikit-learn module to perform machine learning on a movie dataset that I built. Additionally, I made a web app that allows users to search and filter through the movie dataset. The programming tools I used include: Python, Cloud 9.io for web development, scikit-learn, BeautifulSoup, Regular Expressions, the MovieLens dataset, JSON, Chart.js, HTML, CSS, JavaScript, jQuery.

The first major step is to build the movie dataset. I downloaded the MovieLens dataset, which contains information on ratings for a large list of movies. The MovieLens dataset provides an ID number for each movie so I can easily match a rating to a title.

A table of budgets and worldwide grosses for around 5000 movies is provided by the-numbers website. In order to scrape information from this table, I use the BeautifulSoup library. I read in the URL of the webpage and loop over all instances of the <td> tag. This tag contains information about each row in the table, which will be stored in an array.

Now, I have to match the budgets and worldwide grosses provided by the-numbers website to the movie titles from the MovieLens dataset. The titles provided by MovieLens and the-numbers website don’t exactly match, so I have to rely on looping over the words of a title from one source and checking to see if the other source has the same words. I use a regular expression command to ignore punctuation.

The final dataset includes the budget, average rating, month of release date, and classification of each movie as a success or flop. These parameters will be used later in machine learning. Save the dataset in a text file.

The second major step is to use scikit-learn for machine learning. To do this, we need to first load the dataset. Initialize two lists, called X and y. Loop over each line in the text file containing the dataset. For the best results, we will normalize our data first. We do this by finding the mean and standard deviation of budgets and ratings, which can be easily found by pasting the data in Excel and using Excel functions. Since budget numbers have a very wide range, we take the logarithm (base 10) of each budget. We map the month numbers along a circle, using sine and cosine functions. The month numbers will be represented by month\_x and month\_y as location values on a circle. After processing all of this, we will define each data point as having a budget, rating, month\_x, and month\_y. Load each data point into the X array. Load each classification value into the y array.

Before performing machine learning, we want to randomize both lists X and y so that we do not obtain the same results every time. We do this by using the zip function to bring the two lists together into one list, then we shuffle it. After shuffling, unzip the list back into the original X and y lists. We now need to define which classifier we want to use for machine learning. After choosing a classifier, we choose about 80% of our dataset for training, using the fit function. The remaining 20% of our dataset is used for predicting whether a movie is a success or flop. To get a good idea of the accuracy of prediction, we repeat this process over a large amount of iterations.

The final major step is building the web app to showcase results. We will be building our web app on the Cloud 9.io platform. One aspect of the web app is searching functions. First, we convert our dataset into JSON format and upload the JSON file onto Cloud 9.io. The search functions include: listing movies released in a certain year, how many movies were successes/flops in a certain month, listing movies of a certain genre, and retrieving information on a specific movie.

The second aspect of the web app is a chart displaying movies based on their budget and ratings along with their classification. For this task, we will be using Chart.js. Under the chart declaration, we need to specify a couple of sections. For the chart “type”, we will display a scatterplot. We take the necessary data from the JSON file and put them into an array under the “data” section. We also need to color the data points according to category (green for success, red for flop). Under the “options” section, we insert the title, axis labels, and axis scale. Finally, in the “tooltips” section, we specify what we want to display on the screen when the user hovers their mouse over a data point. We will display the title of the movie, followed by its budget and rating numbers. When the user loads the webpage, the chart should display all the data. We also include the ability to filter the data points by certain qualities, which will be executed using a jQuery click command function when the user clicks on the filter buttons.

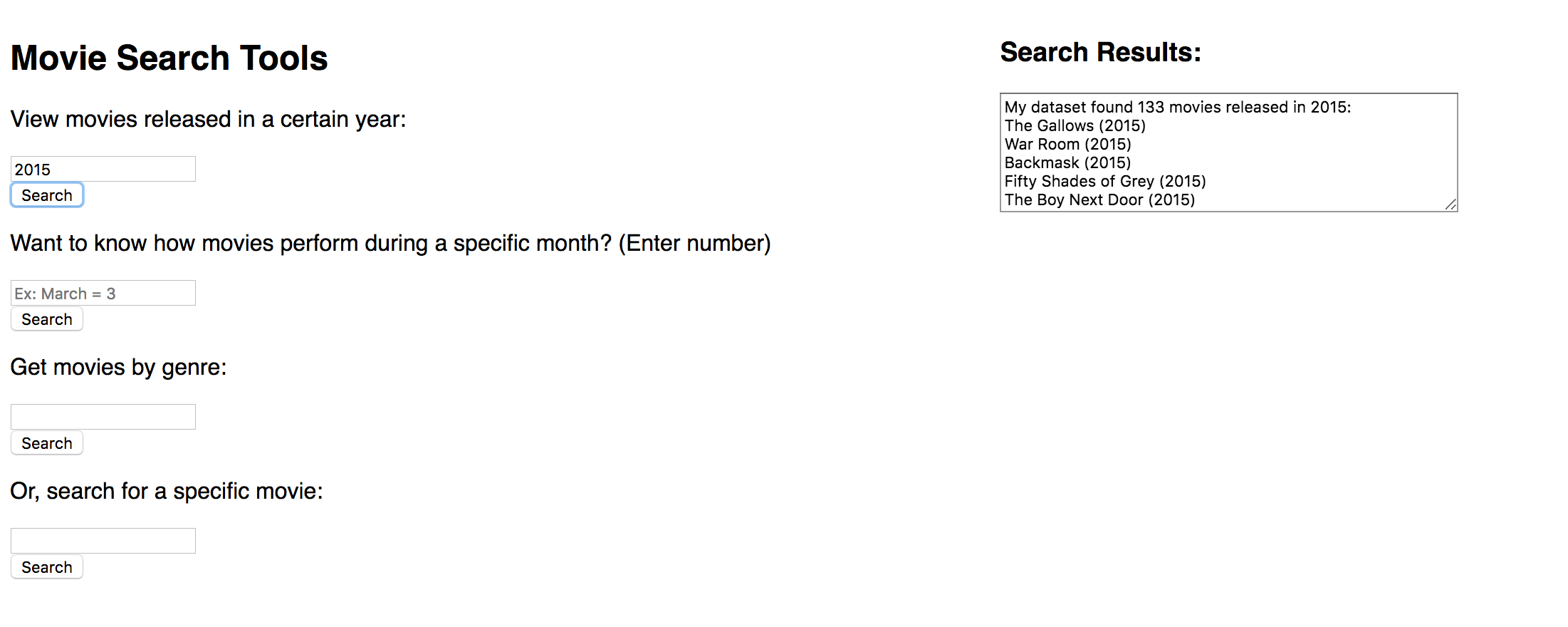
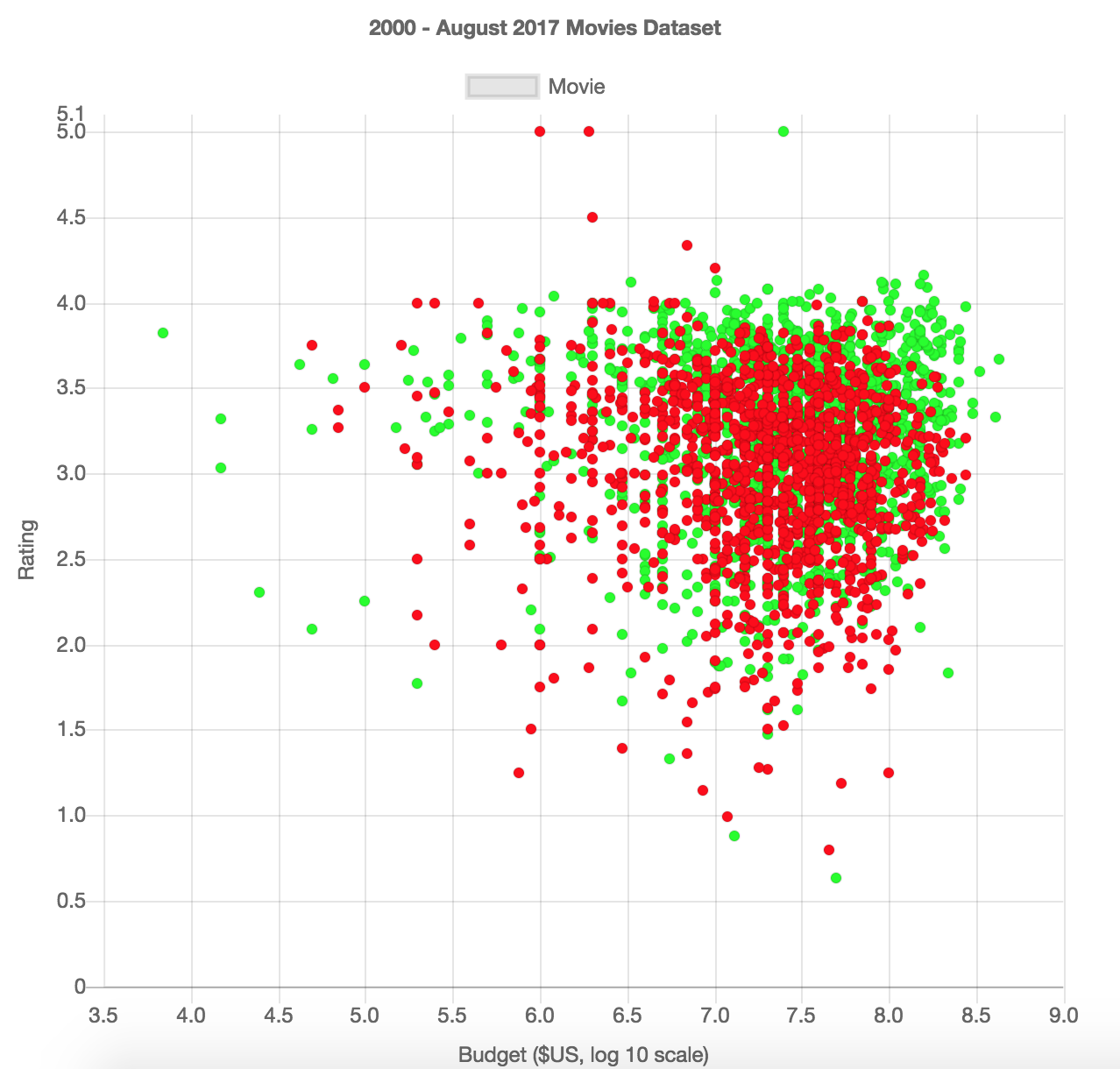
In terms of limitations, I was originally going to use data extracted from tweets using the Tweepy API. However, the Tweepy API only allows users to search for tweets posted in the past 7 days. Premium API services from Twitter requires a large sum of payment, so I decided that using tweets was not a viable option for this project.

While building my movie dataset, I worked with two different sources: the MovieLens dataset and the table of budgets from the-numbers.com. I was able to link together data from these two sources by matching movie titles using regular expressions. However, while testing, I noticed some of the movie titles failed to match up between the two sources. One example is Star Wars: The Force Awakens. Still, I was able to retrieve full data for over 2000 movies, which is a large enough dataset to use for machine learning.

In terms of internal testing, I worked with several different classifiers while conducting machine learning using scikit-learn. To test the effectiveness of each classifier, I ran 30 independent trials for each classifier. In each trial, the classifier was trained on a randomly chosen 80% of the movie dataset. For the other 20% of the dataset, the classifier predicted whether a movie fell under the “success” or “flop” category. The accuracy of predictions was noted for each trial, and I compared the average accuracy of 30 trials across each classifier. The results showed that the SVM (RBF) emerged as the best classifier for this dataset.

For a user validation test, I asked 3 people to play around with my web app. I instructed them to use all the search options, use the filter options for the graph, and toggle between gradient options to color the graph. All 3 people were satisfied with the features presented in my web app, and they did not find any bugs with the features.

Two visuals of my web app are shown below, showing the search options and the graph.



**~~Results~~ / Discussion / Conclusion:**

After selecting the best classifier to use for prediction, I obtained an average accuracy of 62%. I attempted to scale the data points by various factors to determine where I could maximize the prediction accuracy. The best accuracy is obtained with a factor of 1.5x.

The 62% accuracy suggests that one can predict whether a movie will succeed with some degree of accuracy. Our dataset of movies suggests there is an area of budget, rating, and release date combination that usually produces successful movies. However, outside of this area the distribution of movies is noisily dispersed with both successes and flops.

Through this project, I’ve learned how data analytics can lead to useful predictions for industries concerned with the success of their products and services. In the context of the movie industry, movie studios may use such information to better plan for future releases, so that they can minimize the risk of their movies flopping. They can consider controlling the size of their budget on production, how much to spend on advertising, and selecting a suitable release date for a movie.

Further considerations of this project should include sentiment analysis to gauge how healthy the word of mouth is for a particular movie. This project already included ratings to show how much audiences like a movie, so sentiment analysis can serve as a useful supplement. The sentiment analysis can be performed on social media posts that mention specific movie titles, capturing how audiences react to their viewing of a particular movie.

**References:**

Hennig-Thurau, Thorsten, et al. "Does Twitter Matter? The Impact of Microblogging Word of Mouth on Consumers’ Adoption of New Movies." *Journal of the Academy of Marketing Science*, vol. 43, no. 3, 2015, pp. 375-94, openaccess.city.ac.uk/5272/1/Does%20Twitter%20Matter%202015.pdf. Accessed 13 Oct. 2017.

Jain, Vasu. "Prediction of Movie Success Using Sentiment Analysis of Tweets." *The International Journal of Soft Computing and Software Engineering*, vol. 3, no. 3, Mar. 2013, pp. 308-13. Accessed 11 Oct. 2017.

Kuizinas, Gajus. "The Twitter Effect." *Go 2 Cinema*, 27 Sept. 2017, go2cinema.com/stories/the-twitter-effect-8. Accessed 11 Oct. 2017.

Ray, Sunil. "Essentials of Machine Learning Algorithms (with Python and R Codes)." *Analytics Vidhya*, 9 Sept. 2017, www.analyticsvidhya.com/blog/2017/09/common-machine-learning-algorithms/. Accessed 11 Oct. 2017.

Robinson, Scott. "The Best Machine Learning Libraries in Python." *Stack Abuse*, 10 Nov. 2015, stackabuse.com/the-best-machine-learning-libraries-in-python/. Accessed 13 Oct. 2017.

Sharma, Gargi. "Applications of Sentiment Analysis in Business." *ParallelDots*, 24 Dec. 2016, blog.paralleldots.com/technology/deep-learning/applications-of-sentiment-analysis-in-business/. Accessed 11 Oct. 2017.

Sharma, Sheetal. "Top 9 Machine Learning Applications in Real World." *Data Science Central*, 7 Aug. 2017, www.datasciencecentral.com/profiles/blogs/top-9-machine-learning-applications-in-real-world. Accessed 11 Oct. 2017.