Jeffrey Wang

Period 5

Background Section Draft

* How to build the 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.
* Using scikit learn for machine learning
  + First, we need to 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 you can easily find 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.
* 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.