MSCI 446 - Project Report
Professor Golab

Group 2:
Calum Hrabi: 20908876
John Kachura: 20875639
Elize Kooij: 20885246
Thomas Kleinknect: 20883814

April 6th, 2023
University of Waterloo

Link to project proposal: MSCI 446 - Project Proposal

Plan:

- Data cleaning
    - Refer to plan in proposal
    - Clean -> transform -> vectorize -> put thru algorithm

- Timeline
    - Next wed 2:30, March 22nd (After ML class)
    - Get running
    - March 28th-30th?? (More meetings after Software is completed)

Steps:

- Import data
- Remove old data
- Remove TV episodes, Shorts and Adult films
- Remove Non-english words
- Trim the amount of genres
- Remove common words like “the” and “a”
- Stemming
- Vectorize words
- ^ EK + CH
- Run through Multiclass Logistic Regression Algo and Naive Bayes
    - **Scikit Learn [TK + JK]**
        - https://machinelearningmastery.com/multinomial-logistic-regression-with-python/
        - Pandas dataset, what to do to apply the algorithm (x2 both)
        - Add steps
        - Tutorial w something more manageable
- Success:
    - The best genre that it fits, or one of the possible ones
- Testing
    - K-fold cross validation
    - Find the determine the optimal K value (elbow)
    - https://satishgunjal.com/kfold/

-

-

# of Movies with 1 Genre - 644532
# of Movies with 2 Genres - 264285
# of Movies with 3 Genres - 158579

- As you said, Logit models only work with numerical inputs. If you want to use text-based analysis, it's better to look into other ML algorithms for such tasks. One of the best algorithms is Naïve Bayes which is used in Spam filtering applications. Besides, you don't just want to classify your movies as 2 genres, so a multiclass logit model should be developed.
- A bit lacking in terms of citations in order to prove the importance of your work. However, only 1 point is deducted.
- A few grammatical mistakes that make the sentences difficult to read.
- Text not properly justified.

Deliverable:

Introduction. - [John] ✅ — NEED SOMEONE TO READ THIS OVER

The goal of our project is to develop a machine learning model which can predict the genre of a movie, based on an inputted string of text representing its title. A movie's genre is a very defining characteristic and primary identifier of any film, old or new. We would like to determine just how accurately we can predict this primary identifier using another, less descriptive and more random identifier, the movie's name. There are obvious cases in which this should be relatively simple, films with "ghost" or "scream" in the name, are most likely horror films, and movies with titles including "love", or "marriage", are most likely romance films. But what happens when things stray from these direct relationships? What happens when a movie title includes combinations, such as "Screaming Laughing at the War Celebration", or single worded titles, such as "Hairspray"? We aim to explore how accurate we can make our predictions and reveal possible tendencies towards certain genres.

This project can provide important insights as movie genres are key factors of how they are categorized, marketed, and consumed by audiences. It grabs the viewer's attention, leads them to quickly grasp the type of movie they are expecting to watch, and helps their decision making process to select a movie. Our model will also allow us to bridge the gap of ambiguous movie titles, and create or discover existing relationships between the title and the genre of a movie. Successfully creating a working model can help movie marketing teams or creative advertising agencies ensure a movie title is emitting the right genre and targeting their intended audience. Some additional real world applications we envision for this model would serve in the context of movie search engineers, discovery platforms, and enhancing user experience/engagement.

To successfully create a working model for our project, we will be using a supervised learning model, applying Naive Bayes and Multinomial Logistic Regression algorithms since we are conducting classification with many genre possibilities. This allows us to train the model to draw patterns from these titles, and given different considerations and possible limitations, we will have to determine which model is best suited for the task. Any model is only as good as its data, so we set out to search for a dataset which gave us the best chances at success, by providing us with trustworthy data.

In summation, with our main objective in mind - to create a model to predict a movie's genre based on an inputted title, there were also a plethora of different factors to take into consideration in order to reach as much accuracy as possible. At many stages of this project, important decision making took place, with regards to data processing, attempts at models and how to best measure accuracy. We will discuss in our report which approaches worked best, as well as the challenges and limitations we faced.

- Restate your business problem and motivation ✅
    - This can be taken from our proposal ✅
    - Add better justification (from proposal feedback) ✅
    - Justification for the importance of the problem is a bit lacking. ✅

Data description. - [EK] - EDITING NEEDED PLZ

- **NEED TO INPUT TABLES/CHARTS**

The raw dataset we chose to work with is made up of 10 million rows of movies, shorts, television series and episodes, sourced from the infamous IMDb (CITE SOURCE). This dataset includes movies from 1874 up to present day, each of which are tagged by up to 3 genres they correspond to. With this large dataset at our disposal, there were many data cleaning and manipulation steps we had to conduct to make the data suitable to then apply machine learning models.

As mentioned in our proposal, we started by removing the titles that go too far back in time, as very old movies were drastically different from what is common today, and we are aiming to capture more modern trends in movie naming trends. We removed all titles that were released prior to 1920. To further narrow our dataset, we removed TV episodes, shorts and adult films, as they provide little contribution to naming and we would rather focus solely on movie titles. These were relatively simple to remove, because the dataset includes a titleType column (identifying shorts and episodes) as well as a boolean characteristic, isAdult. We also decided to trim the list of available genres, as some of them are quite niche (ie. experimental, news, history etc.) and helps our model be more general. This was able to be completed because most titles are still tagged by at least one more genre that applies to it (ie. Action, Comedy, Experimental).

One of the more challenging data cleaning steps we had to conduct was removing the non-english language titles (which are quite common), as there is no existing column that displays the language of the movie. We chose to leverage a Python library called Langdetect to filter our data and only keep the movie titles written in english. Once we ran Langdetect, it significantly reduced our dataset, leaving us with 510,000 data points.

After completing the data cleaning steps mentioned, we used NLTK to tokenize each individual word in the movie titles, which involved breaking down titles into individual words to create a list of tokens. This allowed us to work with individual words rather than the entire title, allowing us to apply ML algorithms with ease.

To continue our text pre-processing, we removed common stop-words like "the" and "a" using the Natural Language Toolkit (NLTK) library. This removes the low-level information and allows us to place more emphasis on words that are likelier to correspond to a genre.

The next step was to complete word stemming. The process involved reducing each word to its root or base form, which normalizes the data and reduces the dimensionality of the dataset. Stemming the words makes it able to group words with similar meanings together and reduces the variations in the dataset, making it easier to find patterns in data. We have specifically removed -s, -ing, -te from the dataset of titles. Finally vectorizing the text data into a numerical format that ML algorithms can understand.

We involved feature engineering to select and create relevant features from the available data to improve the accuracy of our model. Our project's target value is the genres and the features are the words found in the title. However, since many titles are tagged by multiple genres (ADD STATS), decisions are needed to determine how we would handle test entries with multiple genres. One option is to list multiple movies, each with the same name, one for each genre to train the model that the words in this movie's title could correspond to any of these 2 or 3 genres. Another option we have is to run a process which randomly selects one of the multiple associated genres, and then trains the model on the randomized selection. Both options present their own difficulties in training and testing. One other alternative is to limit the number of genres in our model to minimize the number of multi-genre entries and improve the accuracy and functionality of the model.

Once the steps used to clean, transform, and feature engineer data, the dataset was now suitable for applying a number of different ML models to successfully predict the genre(s) of a movie based on a title

- Explain how you collected the data. Explain any data cleaning, data transformation or feature engineering. Present statistics about the data using tables and/or charts.
-

Machine Learning (TK)

- Present and discuss your machine learning results. Point out any surprising or unexpected results.

With the data now prepared for running the models, a new layer of complication has been added to the fold due to the issues encountered with multiple genres being assigned to movies. What this means however, with multiple ways of handling this issue in our arsenal, is that we had more options to consider, and hopefully, better results. In totality, we built and tested eight different models, four using the naive bayes algorithm, and four using multinomial regression. These models were chosen as they are both highly used in classification problems, and their structure works with the fact that our only feature was the title. For each of these supervised learning algorithms, we first attempted the "duplication method" - duplicating titles in the entries in order to have an entry per genre which was listed for the film, and kept the complete list of 27 genres which were in the dataset. This would guide our next steps in the process and hopefully give us some insight.

After running the Naive Bayes model using Scikit Learn, and testing using a test-train split dataset, with the default Scikit Learn setting of 25% of the total dataset used for our testing data, our 27 genre classifier model returned 26.4% accuracy. When run through multinomial regression, it returned a small improvement, but still not a strong result, with 27.2% accuracy. Multinomial regression, as it essentially needs to build a model for each genre, took exponentially longer to train, 3-4 minutes, in comparison with seconds for Naive Bayes. This led to discussions about why and how this occurred. On the testing side, we found a fundamental flaw limiting our maximal possible accuracy. If we list duplicated titles for each genre, given there are two genres, even if we select one of the correct genres, it will not be correct for the other entry and we are limited to 50% accuracy for this title. Another factor however, could be that there truly are too many genres and the model is having trouble distinguishing between them. Some are very limited in the amount of entries they have, and we made the decision to try limiting the genres to 9 from the 27 which were initially listed. This decision was based on not only what were the most common genres, but also eliminating those we thought could be aptly described by one of the others. With two new theories in hand, we set out to test these hypotheses with new models.

Using our still flawed testing criteria, in order to measure just how much this reduction in genres could truly affect our accuracy on its own, we filtered our dataset down to the 9 aforementioned genres. This reduced the total number of entries to 328,187 and reduced the number of entries with two or three genres listed to 90,773. Although fewer entries will hurt the training of the model we still have 64.3% of the data, and the number of available categories for prediction has decreased to 33% of its previous total. When ran, our hypothesis was proven correct, as the multinomial regression model returned 40.9% accuracy, and the Naive Bayes model returned 39.8%, even with the suspectedly flawed testing criteria. This is a substantial increase, and also caused large improvements in training time for the multinomial regression model, as it only took ~2 minutes to train with 9 genres. These factors reinforced our belief that this change was needed. However, how much more could this be improved with an updated testing criteria, or perhaps by randomly selecting the genre of multi-genre entries?

Our solution to the testing issue was to change our criteria for success. We decided to leave the testing data as is, rather than duplicate the titles for each genre as we have done for the training data. This allows us, for the multiple genre entries, to run the entry through the model, return a genre from our model, and then check to see if it is one of the two or three listed. This then would be considered a success. With the same 9 genre data used in the previous model, we made another attempt with the two algorithms, and our new testing criteria improved the accuracy from 40.9% to 56.4% for multinomial regression, and from 39.8%, to 54.2% for Naive Bayes. These two hypothesized changes from our initial models were able to more than double our original accuracy, proving our earlier hypotheses as true.

With these learnings in mind, could the other method of handling multi-genre entries, randomized genre selection, improve this further? This was a method which we were hesitant to start off with, as we were unsure how we could test it accurately. With our original method of testing, we would enact this randomized genre selection on all entries in the dataset, but this caused issues. If the randomly selected genre for a training entry wasn't the one which our model has found to be associated with the words in its title, this would result in a failure, even if this was originally in the list of genres for this entry. With the total number of genres being limited to 9 however, this could be less of a concern, but it would still limit the maximal possible accuracy of this method. With multinomial regression, we returned 55.1% accuracy, which is still much better than some of our previous iterations, but not better than the incumbent method. This pattern was matched by the Naive Bayes algorithm, as it returned 52.2% accuracy, just short of its previous mark.

Conclusions

After all the iteration shown above, the solution which we found most suitable was to classify between nine general genres (drama, documentary, comedy, action, romance, adventure, family, thriller, horror) using the multinomial regression algorithm, which returned an accuracy of 56.4%. Across the board, with every change that was made to the data or to the definition of a successful test, the multinomial regression model performed slightly better than the Naive Bayes model. This makes sense for two core reasons: the amount of training data which we had available, and the distribution of this data across the nine genres. The Naive Bayes model has been shown to train well on smaller amounts of data, but reach its performance asymptote sooner than logistic or multinomial regression, which performs better the more data it has (Ng & Jordan). We had a very large amount of data, with hundreds of thousands of entries and the charts below show the distribution of entries across genres is highly centralized in three genres. The Naive Bayes model is also known to have more bias, and less variance, than logistic or multinomial regression (EDUCBA), and because of this, centralizes its predictions even more towards these three genres. This is shown in the charts below.

There are a few core limitations to our chosen model, and most are related to just how limited the input is to the model. This affects not only the ability to train the model properly, but also its practicality and application. In training, it caused many issues with less common genres, and even with the smaller chosen list of nine genres, it still causes a heavy tendency towards the most popular genres due to the ambiguity which many titles contain. Many inputs may result in the model returning one of the two most popular genres, drama and documentary, due to the titles being too ambiguous (e.g. "This is us"), short (e.g. "Holes"), or containing unique words such as names of people or places (e.g. "Mrs. Robinson"). Given these factors it is easy to see how our accuracy is lower even given the relatively small number of classes.

How does this model project to the real world, given these statistics and limitations? To film-makers, writers, production companies, agents, and anyone else, it allows for a preliminary look into how the name of their movie will be perceived. This applies not only to the average consumer, but also by many search engines or social media platforms which classify content across the internet. With nine possible genres, and 56.4% accuracy, the model is much better than just choosing the most common genre, and depending on the length and ambiguity of the input, the model can give reliable predictions as to how this title will be perceived. Keeping the above listed limitations in mind, and using descriptive words, it should give a result which the average person would have a hard time disagreeing with.

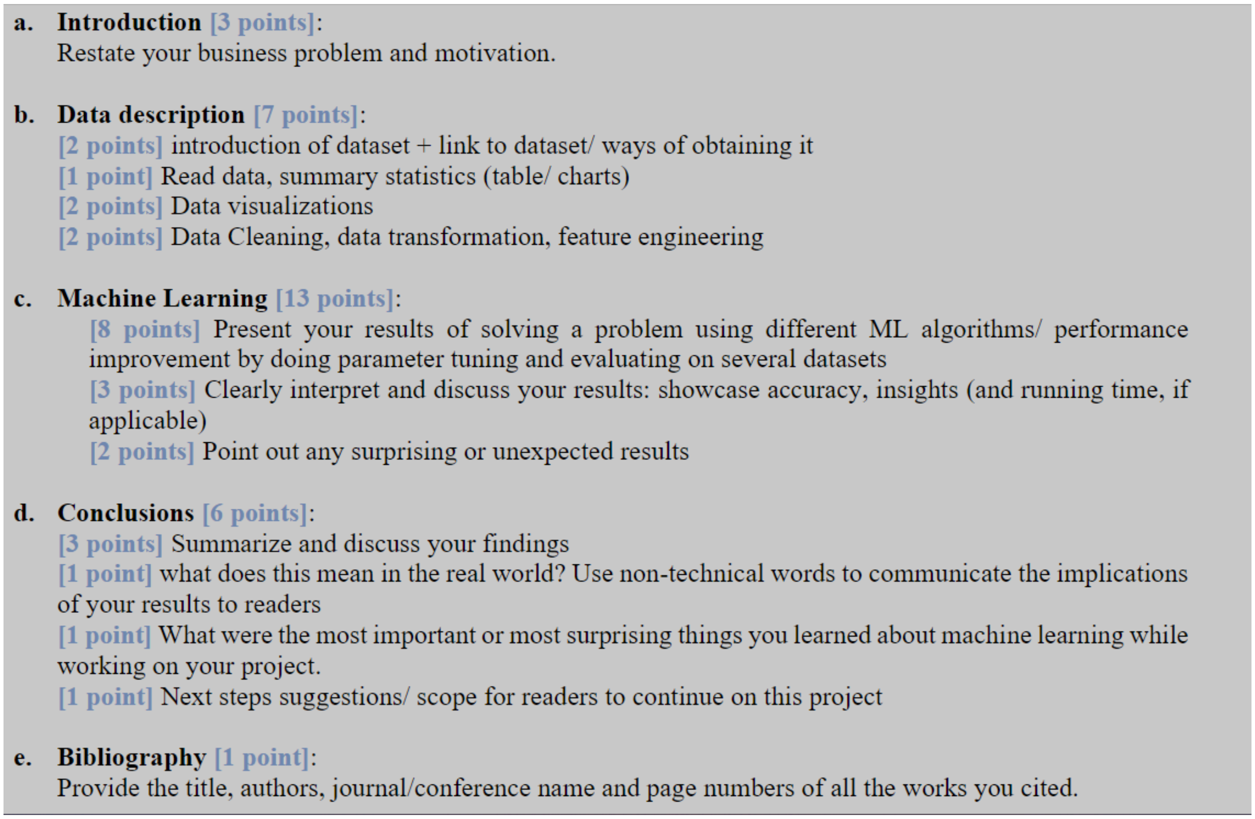
This project taught all of us just how much of the machine learning process truly is about the data. In reflection, the majority of our time was spent making data-centered decisions. We needed to decide what to filter out, how we would handle other languages, ambiguous words, and what to do with the difficulties with our class variable, genre. Determining how to enact these changes took time as well, and even running some of them took hours! We also found that any manipulation of data had a large effect on the way we needed to test when running the model, especially because a lot of the manipulations we needed to make were with our target variable. This relationship was evident as the project progressed. The steps in data preparation were many, as well as in testing, whereas the steps in running the model were relatively few. This makes sense however, as a model that cannot be tested, cannot be used, at least not with any confidence! These are the less glorified aspects of machine learning which made all the difference in our ability to complete this project and improve the accuracy of our model.

Next steps for this project would be to try a pre-built natural language processing model of some kind to perform this classification. This model would preferably be able to take not only the frequency of the words themselves into account, but their meaning, and even better, their meaning in context if possible. This would undoubtedly improve the model significantly, and if trained on the cleaned data which we have, it would be very interesting to see just how much of an improvement could be made. The previously mentioned issues with ambiguity, unique words, and short titles would all still apply, but how much of an effect these things would have on a more sophisticated model would be an interesting experiment to undergo.

Bibliography

EDUCBA. (2023, March 8). *Naive Bayes vs logistic regression: Top 5 differences you should know*. EDUCBA. Retrieved April 6, 2023, from https://www.educba.com/naive-bayes-vs-logistic-regression/

Ng, A. Y., & Jordan, M. I. (2001, January 1). *On discriminative vs. Generative Classifiers: Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and synthetic*. Guide Proceedings. Retrieved April 6, 2023, from https://dl.acm.org/doi/10.5555/2980539.2980648



a. **Introduction** [3 points]:
Restate your business problem and motivation.

b. **Data description** [7 points]:
[2 points] introduction of dataset + link to dataset/ ways of obtaining it
[1 point] Read data, summary statistics (table/ charts)
[2 points] Data visualizations
[2 points] Data Cleaning, data transformation, feature engineering

c. **Machine Learning** [13 points]:
[8 points] Present your results of solving a problem using different ML algorithms/ performance improvement by doing parameter tuning and evaluating on several datasets
[3 points] Clearly interpret and discuss your results: showcase accuracy, insights (and running time, if applicable)
[2 points] Point out any surprising or unexpected results

d. **Conclusions** [6 points]:
[3 points] Summarize and discuss your findings
[1 point] what does this mean in the real world? Use non-technical words to communicate the implications of your results to readers
[1 point] What were the most important or most surprising things you learned about machine learning while working on your project.
[1 point] Next steps suggestions/ scope for readers to continue on this project

e. **Bibliography** [1 point]:
Provide the title, authors, journal/conference name and page numbers of all the works you cited.

Data Cleaning

Genres: currently 27

Status update

- All the text processing done
    - And written up
- Ran Naive bayes algorithm
    - 26% accuracy rn (sucky)
    - Tried with 3 lines
    - Will try to Improve (try with the diff genre options)
    - Try without stemming
    - Try with less classes (trimming genres)
- Algorithms still to do
    - Multi Regression
    - Try 1 deep learning one
    - Aiming to get an algorithm that works at least 50% for now
- Have meeting
    - Monday 3pm?
    - Run thru all the code, calum walkthrough
    - Write report

Game plan Monday April 3:

- Calum to improve accuracy
    - Only use 9 specific Genres
        - Drama
        - Documentary
        - Comedy
        - Action
        - Romance
        - Adventure
        - Family
        - Thriller
        - Horror
    - See how many still have 3
    - Try with the random genre
        - But test will choose random too

  - Maybe build own accuracy class (is it one of the two)
  - Or just cut
- Then
  - Logistic regression
  - Neural network
- Write ab the troubles
- Why it gets confused

April 5, 2023

- Logistic Regression only improves it by 1% (random 9 genre)
  - Takes exponentially longer to train (3-4 minutes to train, compared to the seconds in other algorithm)
  - Although logistic regression improves by a miniscule amount it takes much longer to train
- Random genre chosen:
  - Testing this relies on random selection in testing data reflecting the genre which the trained model believes it will be
- All results were improved with the random 9 genre method
- Not doing neural networks anymore
  - Too much to talk about already

For future improvements

- Use a language model duh
  - To acc get the correlations of the word meanings
  - Silly of not to have considered this in the first place