

# Movie Recommendation Using Naive Bayes Text Classification

Dylan McCardle | Corey Myers | Evan Sheehan

# Project Description

---

- Create movie recommendation engine using Naive Bayes for text classification of reviews.
- Input - Prompt user for 3 liked/disliked movies, this is the start of the training/comparison set.
- Algorithm - Iteratively apply different tunings of Naive Bayes
- Output - A list of 1000 movies and the predicted class they belong to (like or dislike).

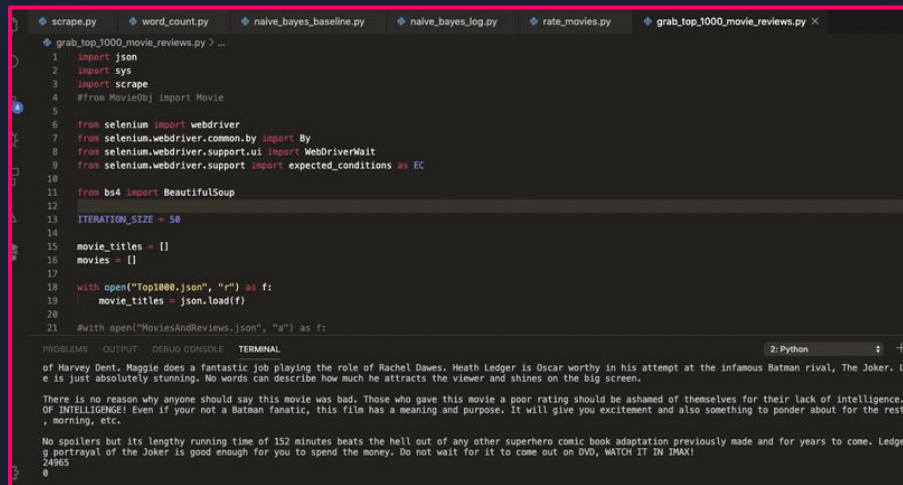
# Constructing a Dataset

---

- Dynamically loading movies to compare proved to be slow and impractical
- Solution: Create a static dataset
  - 1000 Movies
    - Most popular of all time according to Letterboxd.com
  - 100 Reviews for each movie
    - From IMDB. Reviews more useful than Letterboxd

# Constructing a Dataset

- Selenium WebDriver to scrape 100 most helpful reviews from IMDB for each movie.
- Reviews converted to dictionary with word frequency as key value.



```
1 import json
2 import sys
3 import scrape
4 #from MovieObj import Movie
5
6 from selenium import webdriver
7 from selenium.webdriver.common.by import By
8 from selenium.webdriver.support.ui import WebDriverWait
9 from selenium.webdriver.support import expected_conditions as EC
10
11 from bs4 import BeautifulSoup
12
13 ITERATION_SIZE = 50
14
15 movie_titles = []
16 movies = []
17
18 with open("Top1000.json", "r") as f:
19     movie_titles = json.load(f)
20
21 #with open("MoviesAndReviews.json", "a") as f:
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

2: Python

of Harvey Dent. Maggie does a fantastic job playing the role of Rachel Dawes. Heath Ledger is Oscar worthy in his attempt at the infamous Batman rival, The Joker. L  
e is just absolutely stunning. No words can describe how much he attracts the viewer and shines on the big screen.

There is no reason why anyone should say this movie was bad. Those who gave this movie a poor rating should be ashamed of themselves for their lack of intelligence.  
OF INTELLIGENCE! Even if your not a Batman fanatic, this film has a meaning and purpose. It will give you excitement and also something to ponder about for the rest  
, morning, etc.

No spoilers but its lengthy running time of 152 minutes beats the hell out of any other superhero comic book adaptation previously made and for years to come. Ledge  
g portrayal of the Joker is good enough for you to spend the money. Do not wait for it to come out on DVD, WATCH IT IN IMAX!

24965  
0

# Baseline (Multinomial Naive Bayes)

---

- Multinomial Naive Bayes
- First iteration - essentially useless without Laplace smoothing/logical underflow protection
  - Decimals over hundreds of words will underflow to 0 and cause everything to be 0

```
> 00: ['Get Out', 0.0, 0.0, 0.0]
> 01: ['Avengers: Infinity War', 0.0, 0.0, 0.0]
> 02: ['Pulp Fiction', 0.0, 0.0, 0.0]
> 03: ['La La Land', 0.0, 0.0, 0.0]
> 04: ['Spider-Man: Into the Spider-Verse', 0.0, 0.0, 0.0]
> 05: ['Black Panther', 0.0, 0.0, 0.0]
> 06: ['The Dark Knight', 0.0, 0.0, 0.0]
```

**No probabilities!**

# Laplace Smoothing Bayes

---

- Multinomial Naive Bayes but with laplace smoothing
- Smoothing helps account for the fact that not every word will appear in the training vocabulary.
- Still results in nearly all zeros from computational underflow caused by the multiplication of so many very small decimals.

```
> 00: ['Get Out', 0.0, 0.0, 0.0]
> 01: ['Avengers: Infinity War', 0.0, 0.0, 0.0]
> 02: ['Pulp Fiction', 0.0, 0.0, 0.0]
> 03: ['La La Land', 0.0, 0.0, 0.0]
> 04: ['Spider-Man: Into the Spider-Verse', 0.0, 0.0, 0.0]
> 05: ['Black Panther', 0.0, 0.0, 0.0]
> 06: ['The Dark Knight', 0.0, 0.0, 0.0]
```

Still no probabilities!

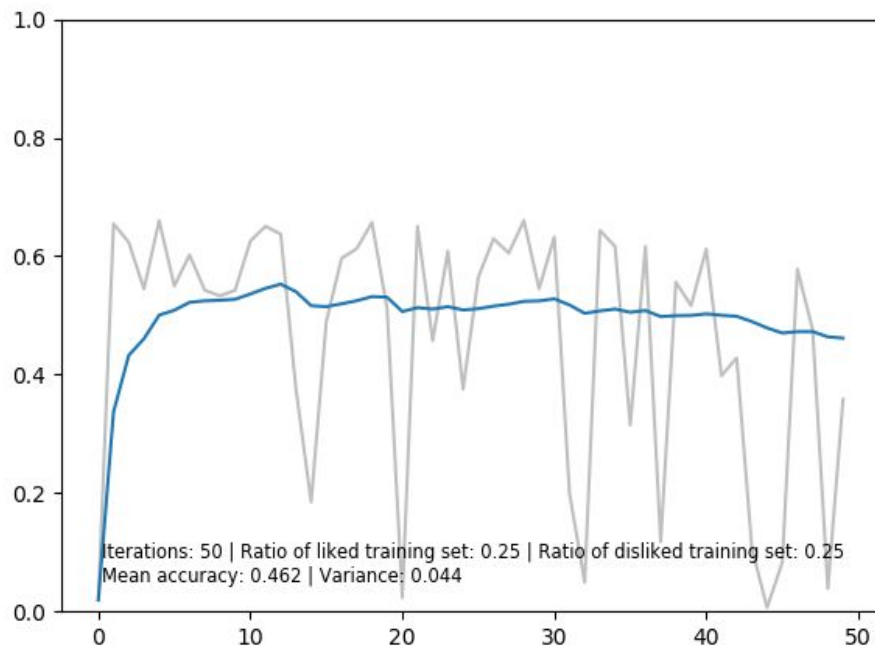
# Logarithmic Summation Bayes

---

- Naive Bayes but in log space
- We kept laplace smoothing and used log space as our way of determining probabilities to eliminate decimal underflow
- At this point we also created a real world data set to begin testing accuracy
  - One of our group members rated all 1000 movies based on liked, disliked and not seen.
- Begin to see actual results, albeit finnickly

# Logarithmic Bayes results

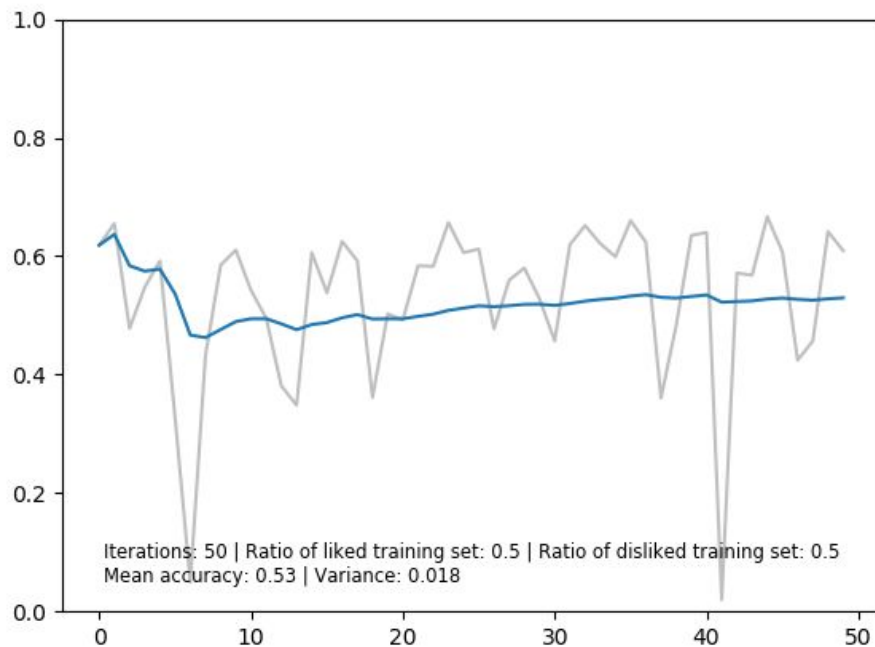
---





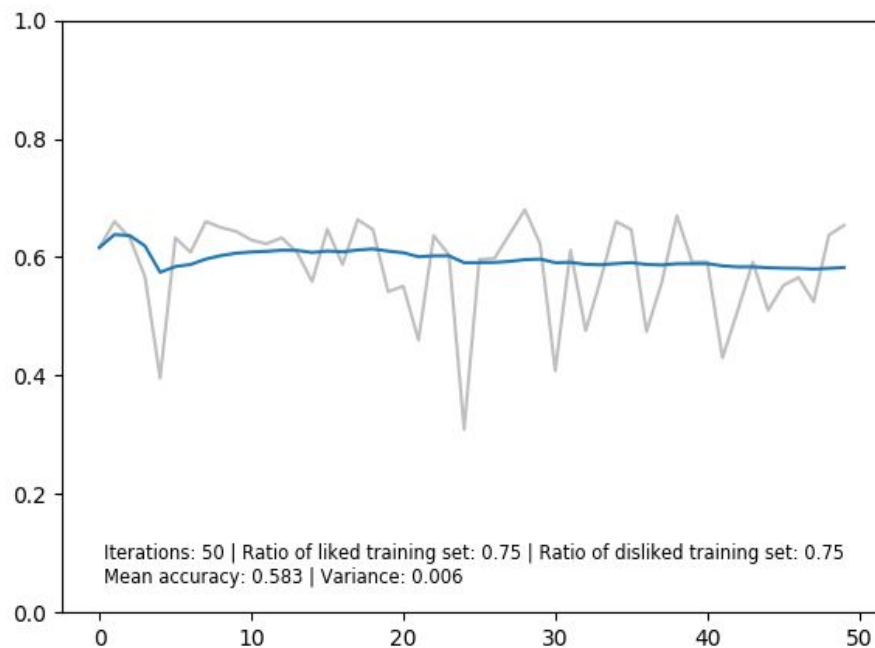
# Logarithmic Bayes results

---



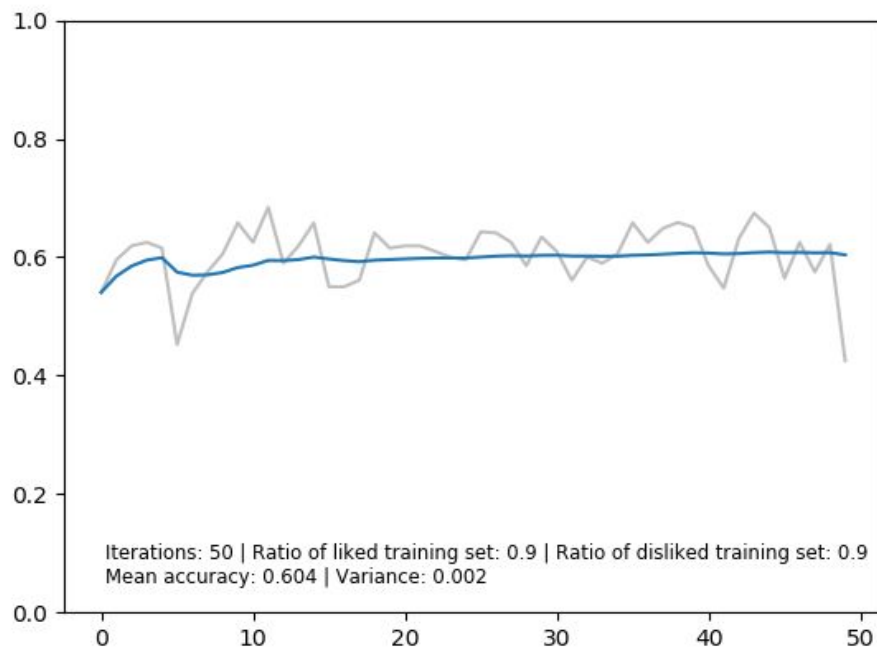
# Logarithmic Bayes results

---



# Logarithmic Bayes results

---



# Other Naive Bayes considerations

---

- Gaussian Naive Bayes
  - Not useful as our data set is not continuous, but instance based
- Complementary Naive Bayes
  - Not useful as our categories are binary, therefore the complement is equal to the other category's probability
- Bernoulli Naive Bayes
  - Potentially useful, yet to implement

# Adjusting Data

---

- Outliers in data found during testing
  - Movies with no reviews at all
  - Movies with a significantly higher number of reviews/words
  - Movies with a significantly lower number of reviews/words
    - Example: The movie “Coco” with only 77 unique words in its review dictionary.
      - A data collection bug likely caused only a single review to be scraped.
- Solution: used standard deviation of the data size to remove outlying values (very small and very large)

# Future progressions

---

- Plan to add more instances of data distribution by polling people to get their likes and dislikes, allowing a different training and testing set
  - Once we have achieved this we can compare several instances of training and testing data from several users to get a mean performance of our algorithm across several people's preferences.
- In progress - create accuracy model (based on number of runs by polling other people)
  - Use random 200 liked/disliked movies as basis for training vocab