Comparing Text with Supervised Machine Learning

Christopher Caldarella, Data Scientist

2021-05-04

Can Machine Learning tell where text is from?

- I pulled data from the popular(?) "social news aggregation" website (per Wikipedia) "***Reddit.com***"
- Data is exclusively comments pulled from 2 similar "subreddits"
- I want to know if a trained Model can predict which "subreddit" each comment came from

- Subreddits include: "AMA", and "AskReddit"
  - These are 2 popular subreddits within the community where people post answers to questions asked.
  - Subreddit names were removed as part of EDA/Cleaning

(Pulling Data,) EDA and Cleaning

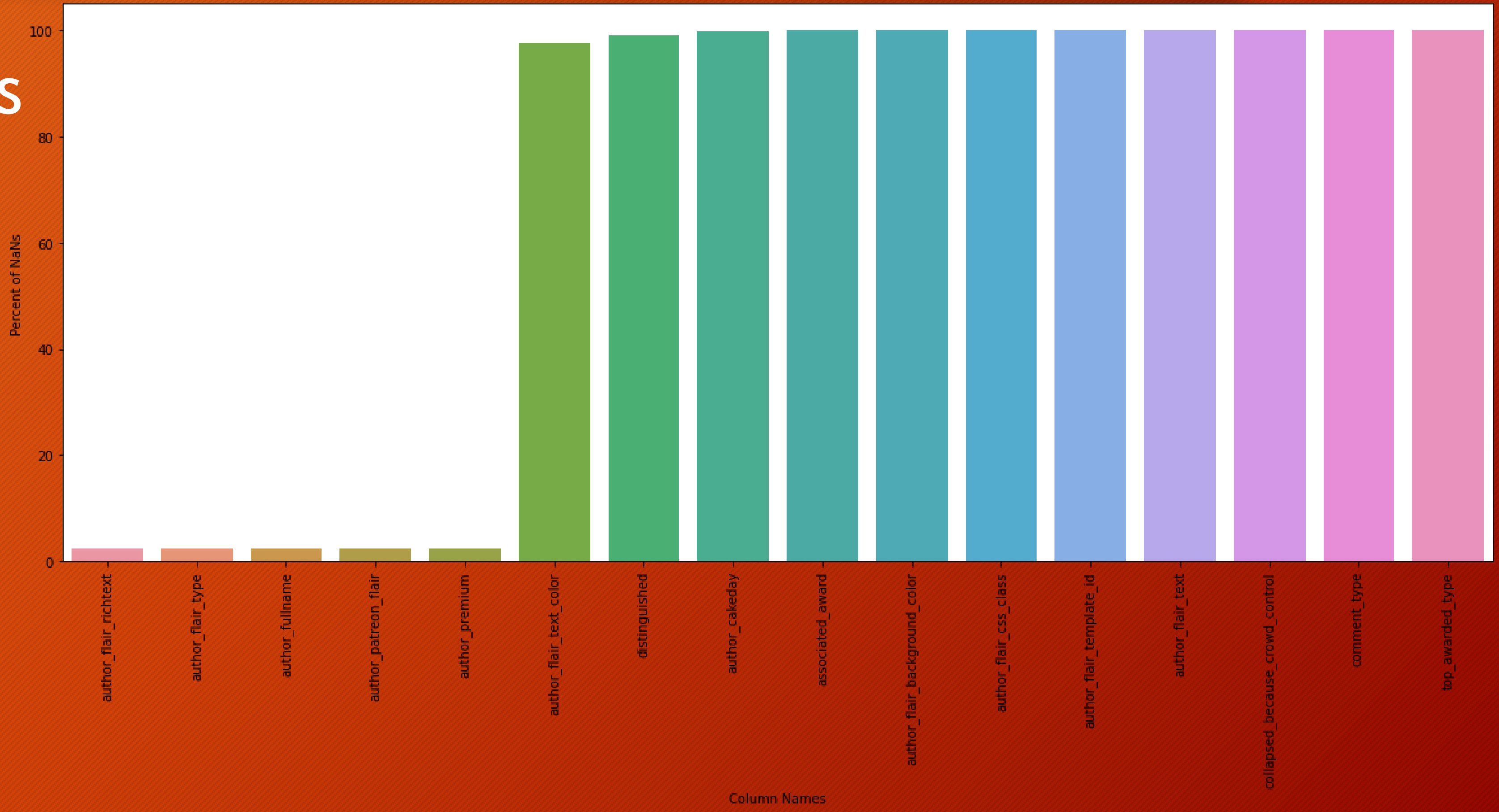
- ***Webscraping***: created a Scraper using Python's "requests" library.
  - Pulled about 12,000 comments total; 10,000 for TTS
- Analysis and Cleaning
  - I added a column 'post_length', a character count
  - I dropped a <u>lot</u> of columns; close to 30 (out of 37)
    - There just was not a lot of data in those columns
    - A lot of it was '[]' or NaNs, or easily identifiable features
  - From the 10,000 entries, over 9,700 were usable, which is fine
    - Removed '[deleted]', and some lengthy 'outlier' comments

(Pulling Data,) EDA and Cleaning

- ***Webscraping***: created a Scraper using Python's "requests" library.
  - Pulled about 12,000 comments total; 10,000 for TTS
- Analysis and Cleaning
  - I added a column 'post_length', a character count
  - I dropped a <u>lot</u> of columns; close to 30 (out of 37)
    - There just was not a lot of data in those columns
    - A lot of it was '[]' or NaNs, or easily identifiable features
  - From the 10,000 entries, over 9,700 were usable, which is fine
    - Removed '[deleted]', and some lengthy 'outlier' comments

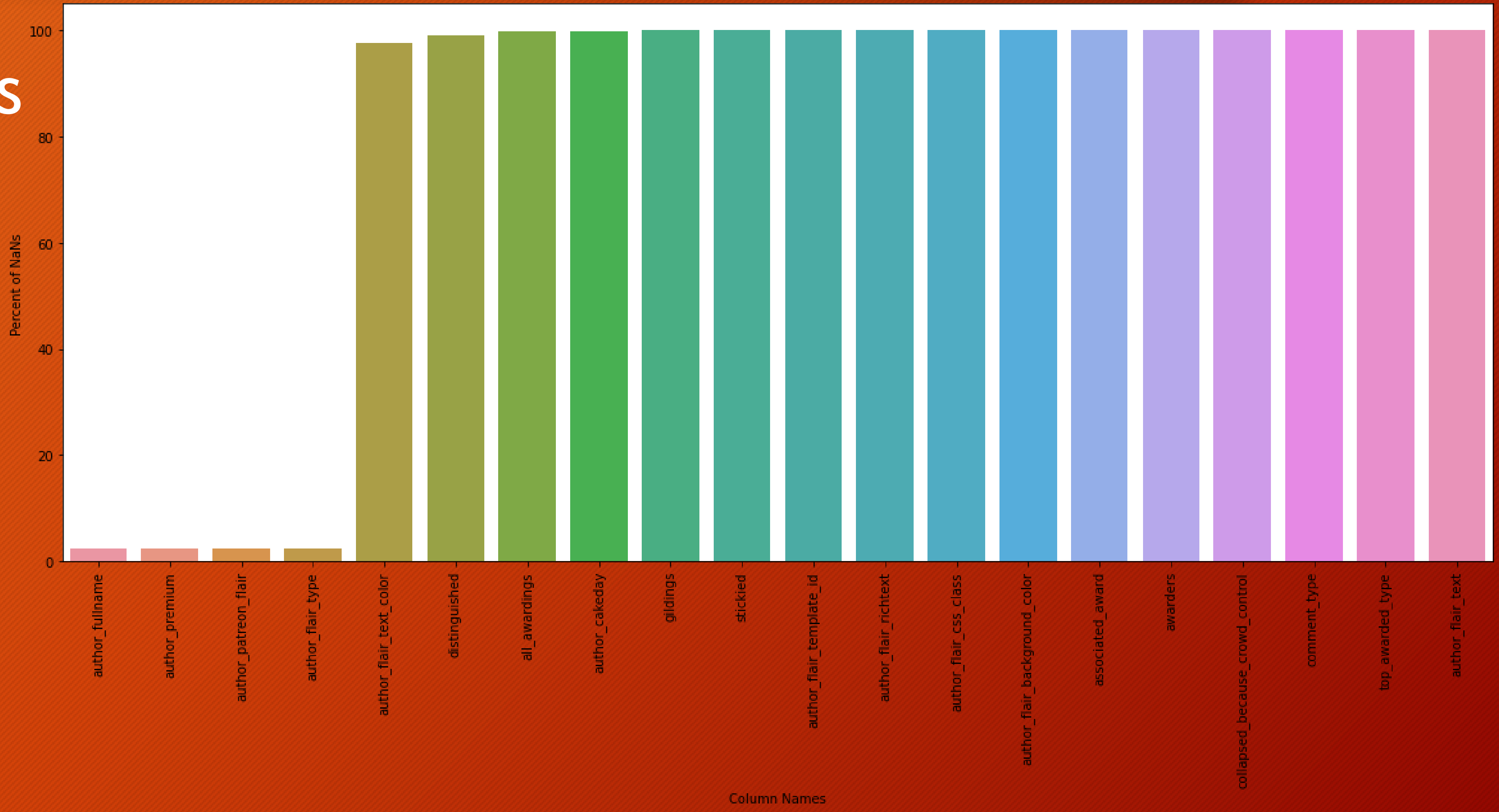
(Pulling Data,) EDA and Cleaning (cont.)

- Dropped Columns
  - (11 here)



(Pulling Data,) EDA and Cleaning (cont.)

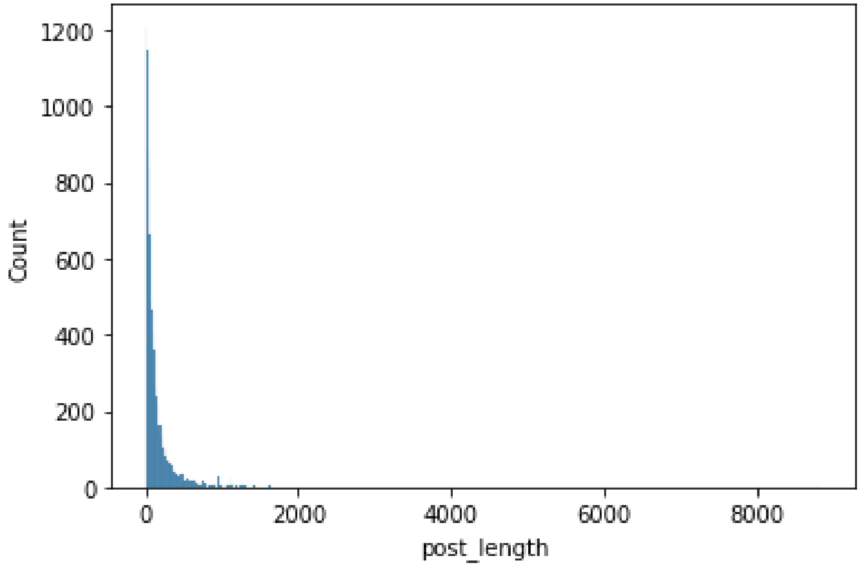
- Dropped Columns
  - (16 here)
  - After changing empty brackets to NaNs
  - (Almost half)



(Pulling Data,) EDA and Cleaning (cont.)

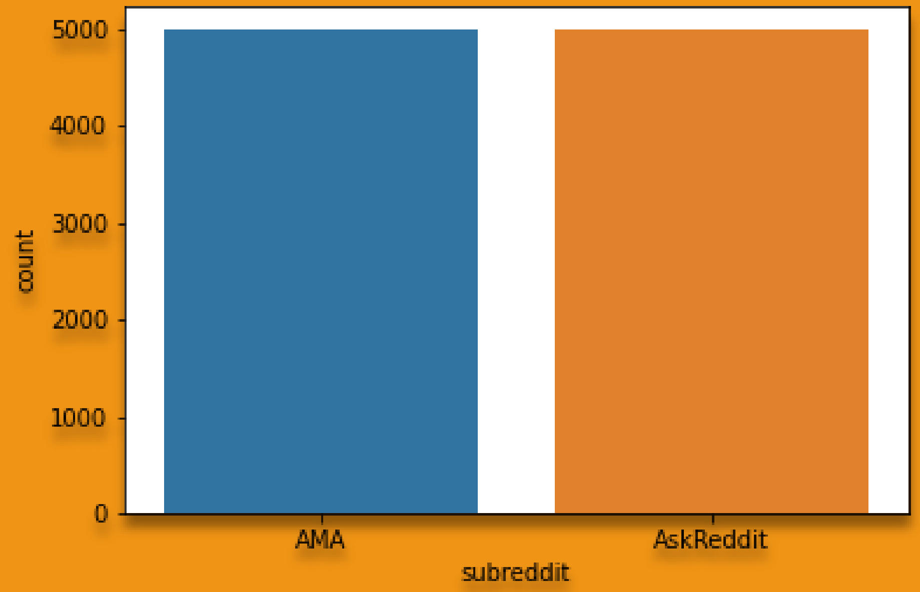
- Added "Post Length feature
- Dropped Rows

- Comments with letters greater than…
  - 6000: 4
  - 4000: 11
  - 2000: 32
  - 1000: 149
- (More from “AMA”)



Null Model

Base model was pretty simple: 50/50



General Observations

What are we looking for?

- A model that performs as well or better than a Naïve Bayes model.

All models used a Vectorizer of some sort:

- Count Vectorizer
- TF-IDF Vectorizer (Term-Frequency-Inverse Document Frequency)

Most Models I tested seemed to do better with TFIDF

- (One of few that performed better with Count Vectorization was Ada Boost)

Most models preferred no Stop-Words

Scoring

- Looked mostly at F1 Score since it is balanced
- Kept Accuracy and Recall in mind as well
- Also looked at TP, FN, FP, and TN and Confusion Matrices

Techniques

- Pipeline and GridSearchCV, TF-IDF
  - Most results: Max_features: 5000, stop words: None
- Later created Functions/Classes to handle and organize my data.
  - Stored scores in DataFrames
  - Ccreated Markup Headers as Labels
- Comparisons were mainly made with tables and some confusion Matrices

Models (F1 Scores)

• Logistic Regression	-	0.7047 *
• Naïve Bayes	-	0.6398 *
• K Neighbors	-	0.6800 ‡
• Random Forrest	-	0.6642
• Decision Tree	-	0.6359
• Bagging	-	0.6146
• Ada Boosting	-	0.6679 ‡
• Gradient Boosting	-	0.6720

‡ CountVectorizer

* From new data

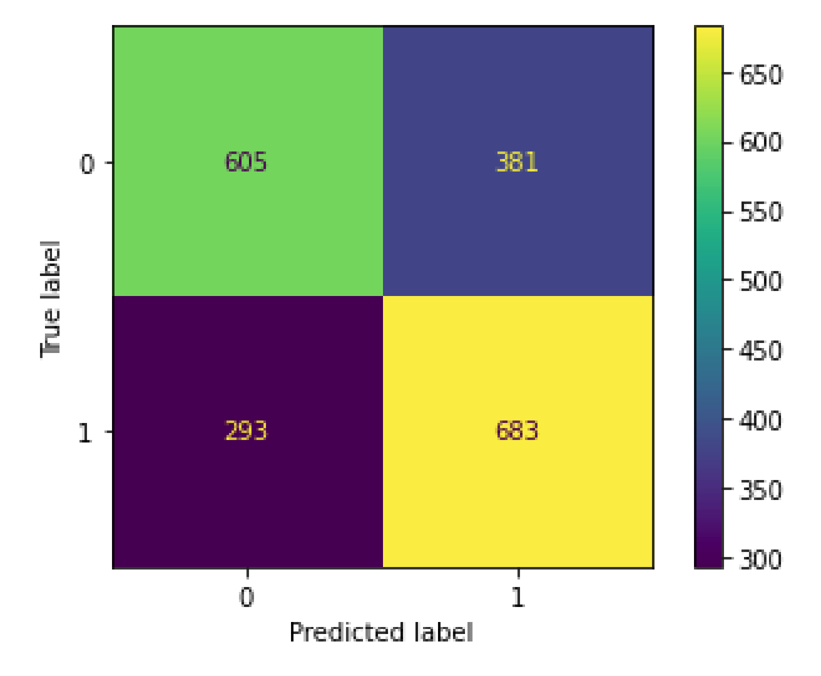
Models without ‘body’, and Voting

- What are Models without ‘body’
  - Removed ‘body’ feature and used the other features.
  - Most other features I was left with were numerical or Boolean
  - I was not able to figure out how to combine them
- ElasticNet (Ridge) and Linear Regression Models
  - With the features that had numerical data – manipulated results by rounding
  - Used OHE, KNNImputer, and Standard Scaler which helped a little
- Random Forest and Logistic Regression had high Recall scores (95%) but were severely biased towards ‘AskReddit’
- Tried Voting (w/ body) with Boosting and Logistic Regression
  - I thought the Boosting overfitting would help, but just brought the scores down

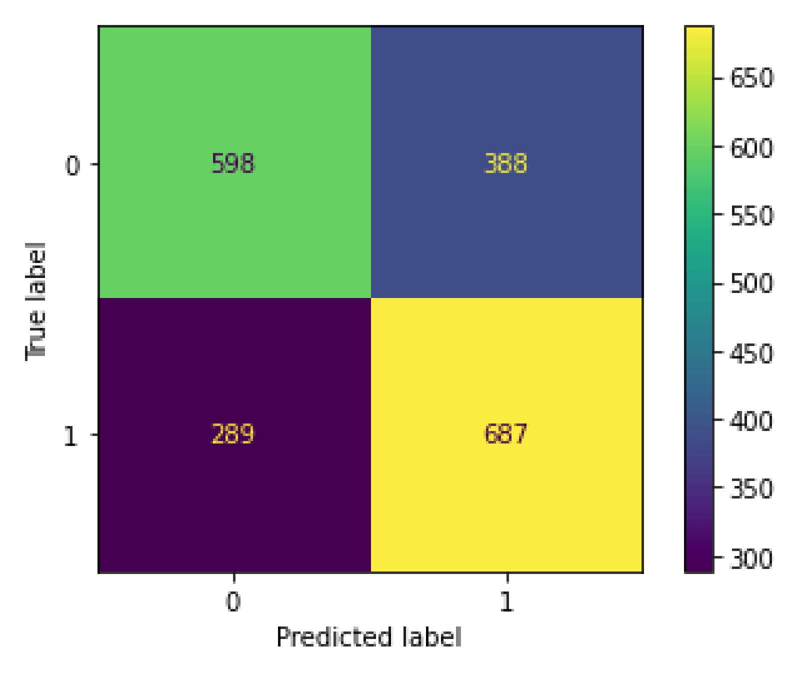
Voting (cont.)

- Maximized values by setting Logistic Regression to 0.6 minimum

Ada, Gradient, Logistic Regression

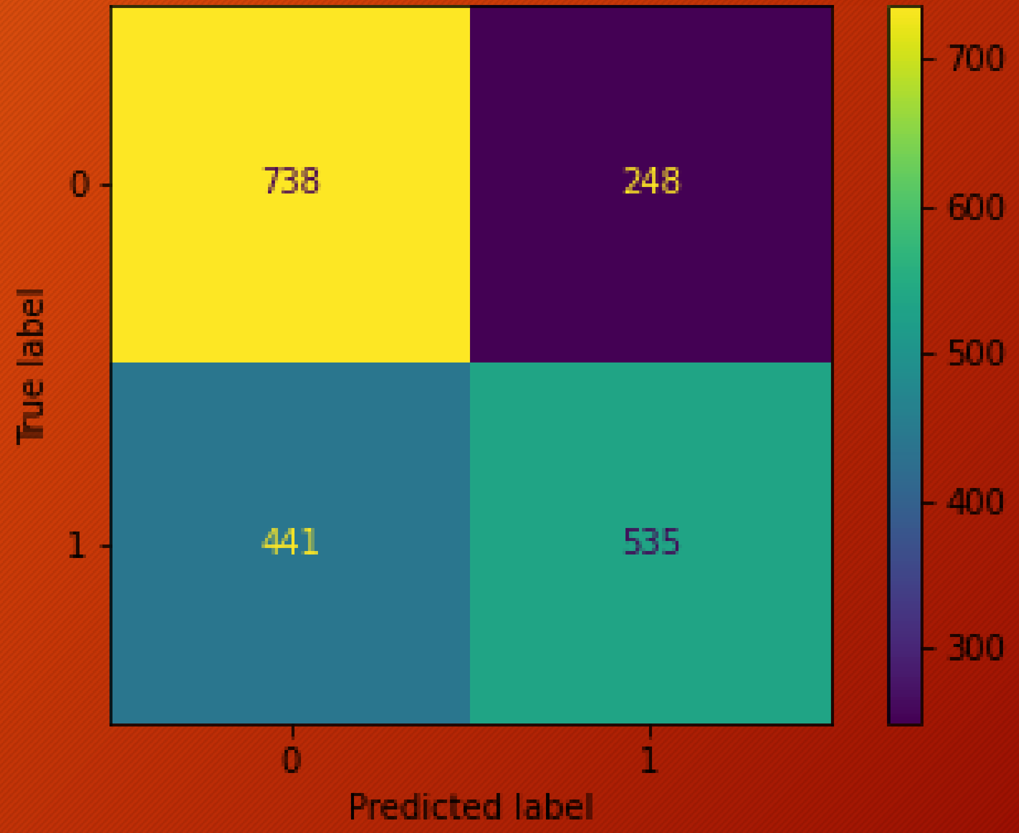


Random Forest, Logistic Regression



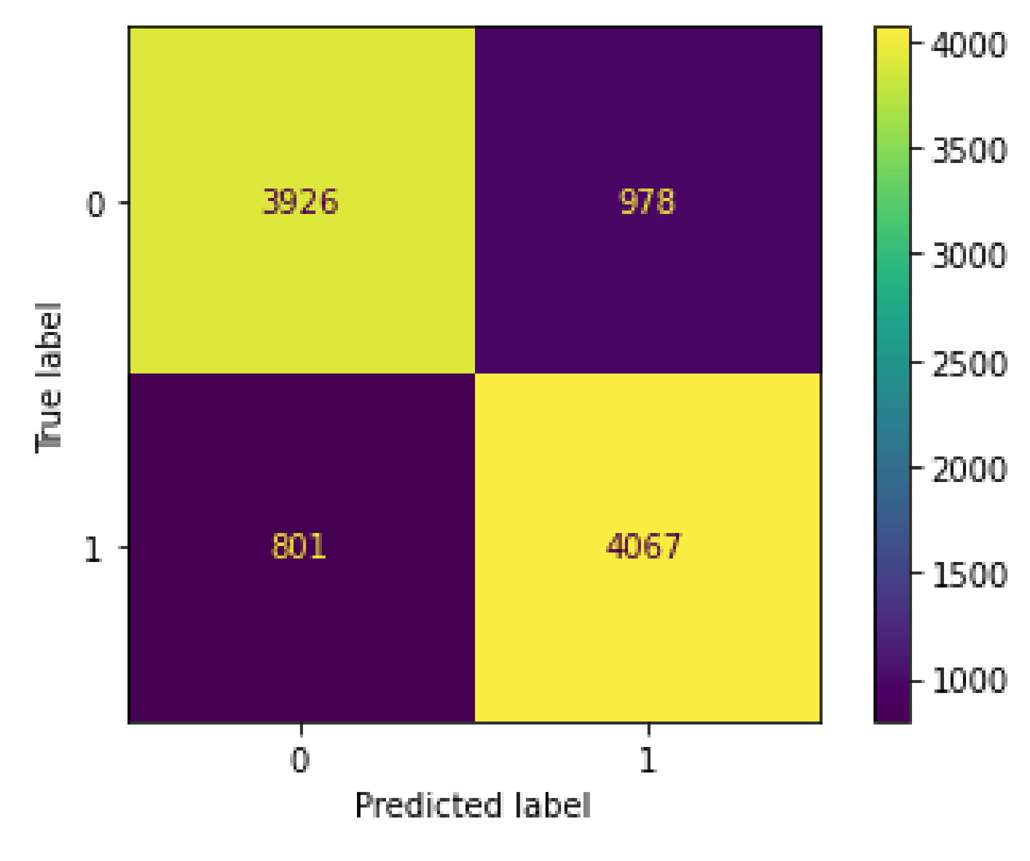
Lemmatized Bayes

- Like most other Models, Bayes with Lemmatization was Biased
- F1 Score: 0.6083
  - (Compared to 0.6398 w/o Lemma)
- It looks like the Bias skews towards ‘AMA’ (0)

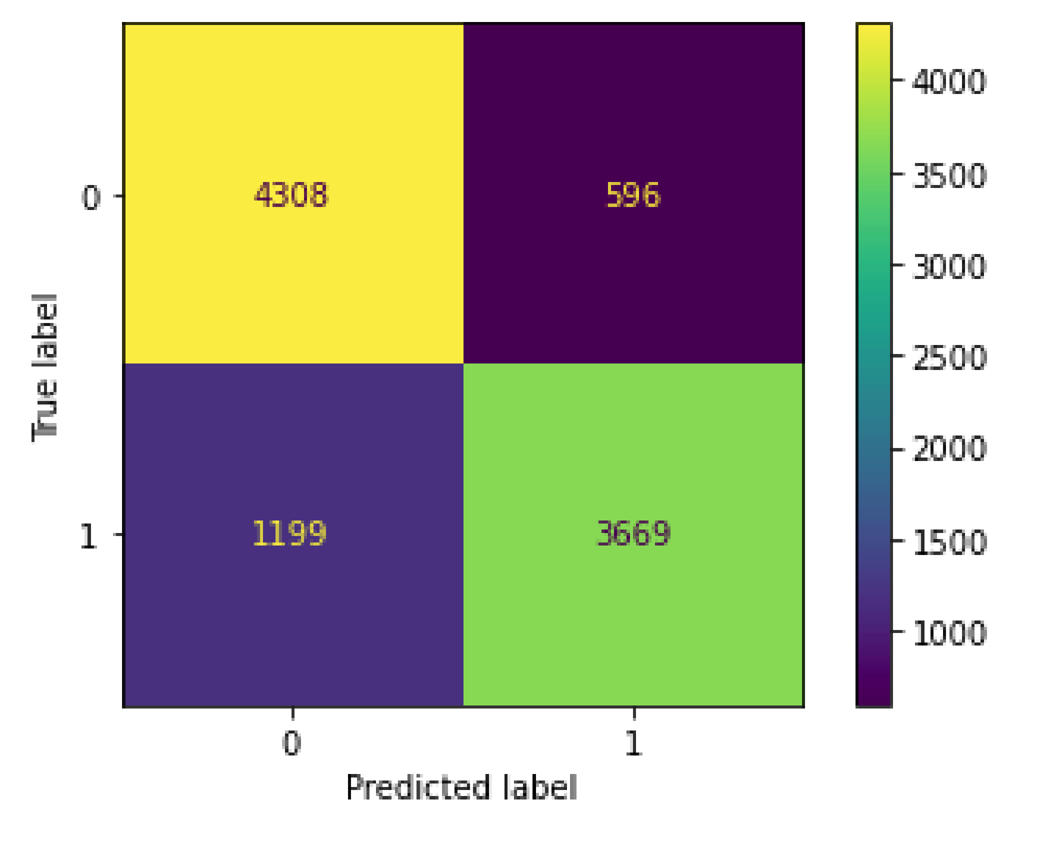


Logistic Regression vs Naïve Bayes

Logistic Regression

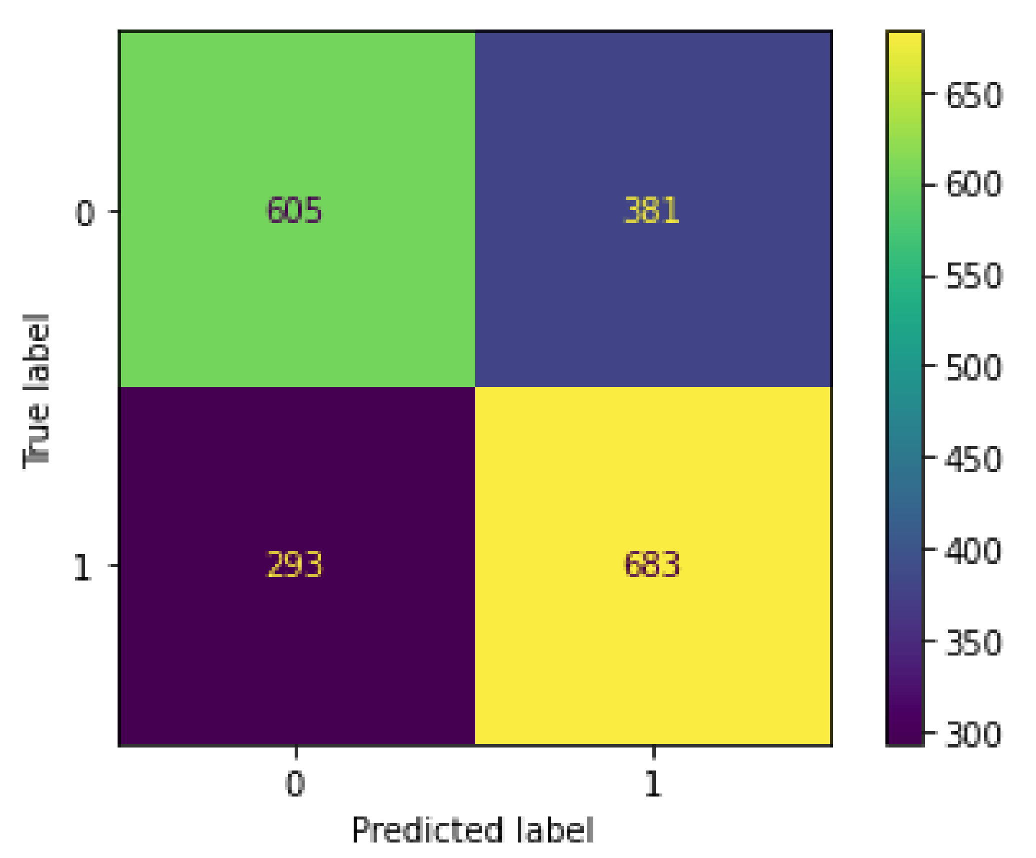


Naïve Bayes



Logistic Regression vs Naïve Bayes

- Why not both?
- It can work! …
  - But, F1 Score of 0.6696
- Logistic Regression on its own still might be better



Conclusion: What is better?

It Depends

- When trying to figure out what category something falls in, we need to find what has the least detriment if any at all.
- In this case, we want balance: **And Logistic Regression delivers**

Questions?