

Project 3

Reddit API Classification & NLP

Chan Song Yuan
General Assembly DSI14
May 18, 2020

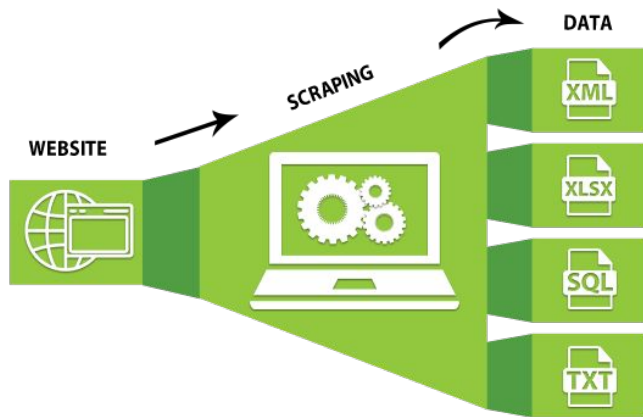
Outline

- **Problem Statement**
- **Data Information**
- **Data Cleaning & EDA**
- **Modeling**
- **Evaluation**
- **Conclusion & Recommendation**

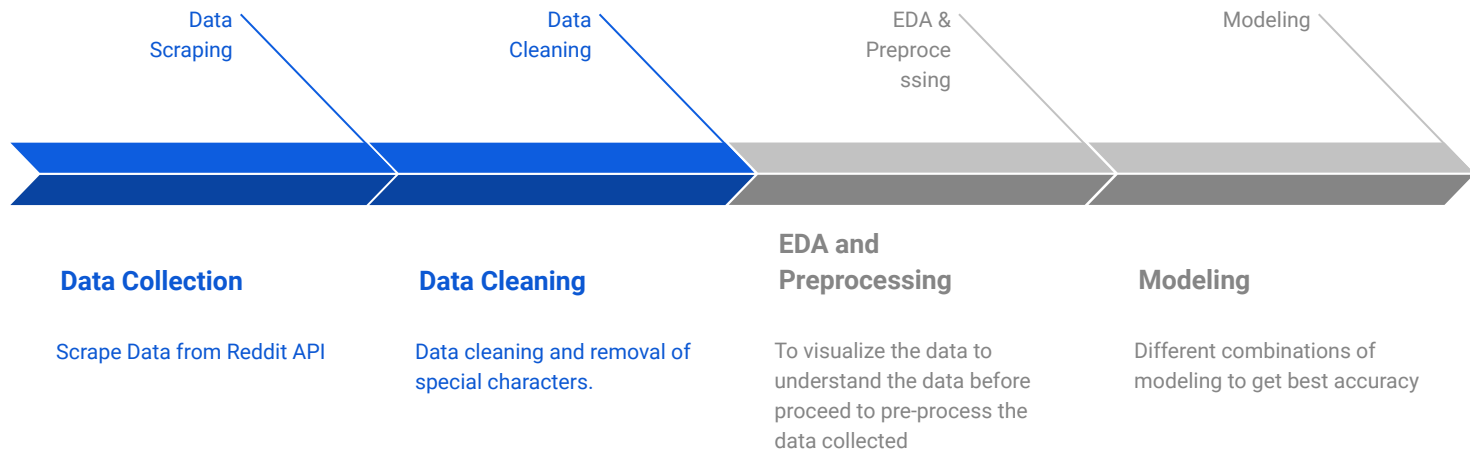


Problem Statement

In this project, we will explore how well Natural Language Processing Model differentiate post content from two similar subreddits which is **/r/Python** & **/r/bigdata**. Which combinations of model and classifier works best? What is the accuracy and how much of the miss-classification will occur between two different subreddit posts?



My Collected Data



- Total of 1837 rows of data collected after drop duplicates
 - 866 from r/Python
 - 971 from r/bigdata

Frequent Words in r/Python and r/bigdata

Word Cloud for /r/Python

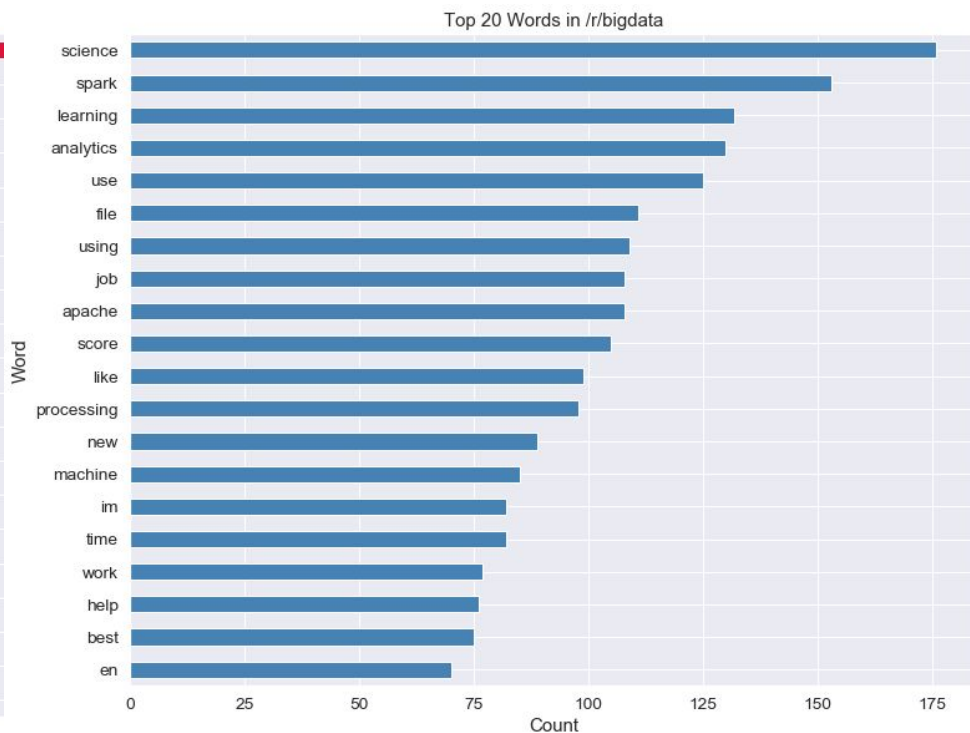
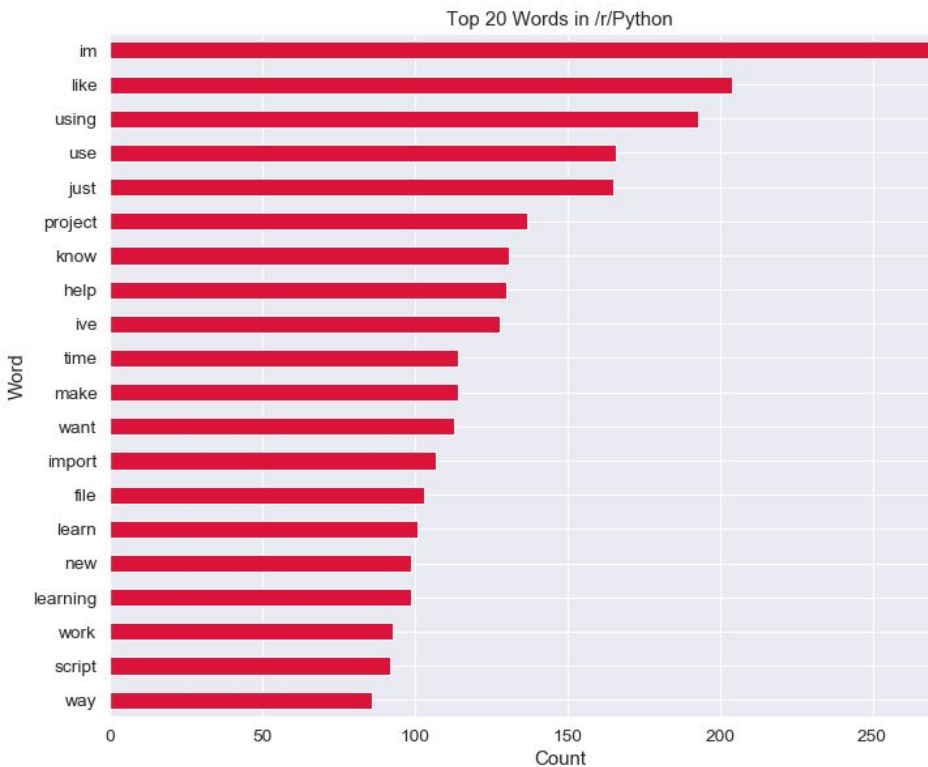


Word Cloud for /r/bigdata



Frequent Words in r/Python & r/bidata

After additional of Stopwords Removed..['did', 'doe', 'don', 'doesn', 'getting', 'going', 'got', 'ha', 'isn', 'wa', 'python', 'big', 'data']



Modeling

Modeling and Selecting the best parameter with GridSearch CV

Model	Train Score (CountVectorizer)	Test Score (CountVectorizer)	Train Score (TfidfVectorizer)	Test Score (TfidfVectorizer)
Logistic Regression	0.9871	0.8179	0.9741	0.8315
Naive Bayes (Multinomial)	0.9244	0.8478	0.9551	0.8668
Random Forest	0.9918	0.7935	0.9925	0.8043
Decision Tree	0.9918	0.75	0.9925	0.7609

Model Parameters

Parameter Set...

```
model_params = {  
    'cv': {  
        'cv__max_features': [2500, 5000],  
        'cv__min_df': [2, 3],  
        'cv__max_df': [.9, .95],  
        'cv__ngram_range': [(1,1), (1,2)]  
    },  
    'tf': {  
        'tf__max_features': [2500, 5000],  
        'tf__min_df': [2, 3],  
        'tf__max_df': [.9, .95],  
        'tf__ngram_range': [(1,1), (1,2)]  
    },  
}
```


Comparing Classification Report

		precision	recall	f1-score	support
CountVectorizer with Naive Bayes (Multinomial)	r/Python	0.83	0.86	0.84	173
	r/bigdata	0.87	0.84	0.85	195
	accuracy			0.85	368
	macro avg	0.85	0.85	0.85	368
	weighted avg	0.85	0.85	0.85	368
		precision	recall	f1-score	support
TF-IDF Vectorizer with Naive Bayes (Multinomial)	r/Python	0.85	0.87	0.86	173
	r/bigdata	0.88	0.87	0.87	195
	accuracy			0.87	368
	macro avg	0.87	0.87	0.87	368
	weighted avg	0.87	0.87	0.87	368

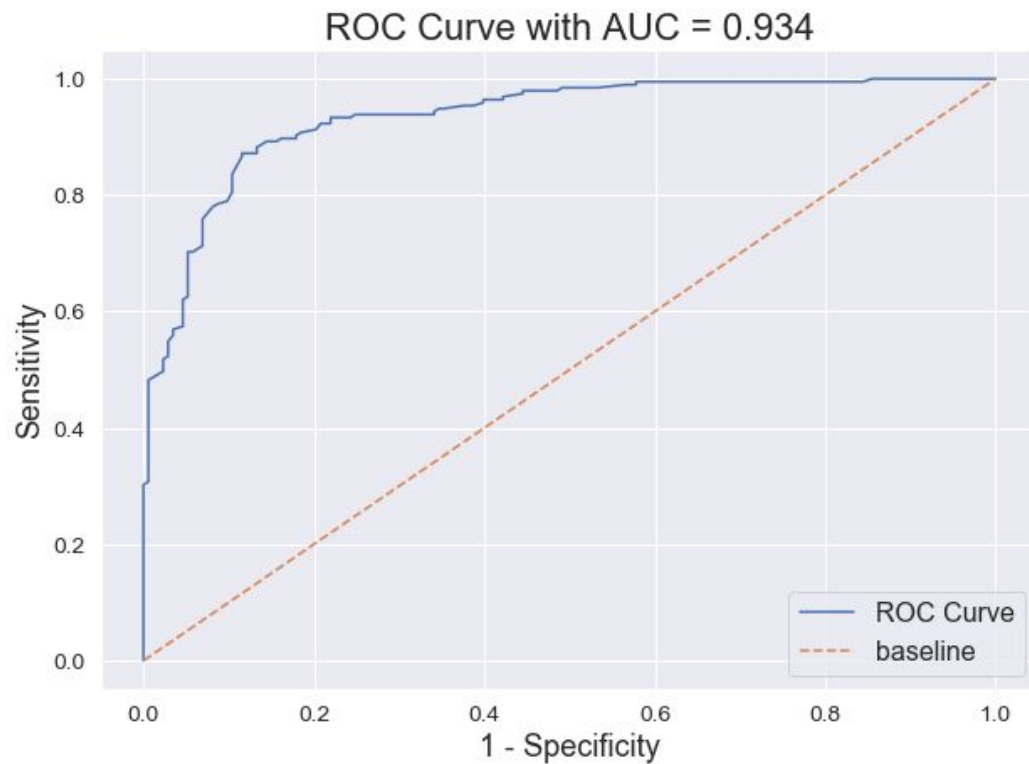
Confusion Matrix

Confusion Matrix based on TF-IDF Vectorizer with Naive Bayes Multinomial

	Predict r/Python	Predict r/bigdata
Actual r/Python	150 / 368	23 / 368
Actual r/bigdata	26 / 368	169 / 368

ROC AUC

ROC AUC SCORE : 0.934



Conclusion

- **Naive Bayes with TF-IDF Vectorizer worked fairly well with accuracy of 87%, with both subreddits were related.**
- **Naive Bayes with Countvectorizer works well too with accuracy of 85%.**
- **Scope can be expanded to improve modes further:**
 - **Collect more subreddit posts**
 - **Tuning parameters for each model to achieve better scores, will take longer time to tune and compile best parameters**
 - **Consider other classifiers, AdaBoost, Gradient Boost, etc.**

THE END