Classification Problem
Learn Python and Learn Programming Subreddit

Ling Chong Gold

Agenda

- Background
- Problem Statement
- Obtaining Data and Cleaning Data
- Exploring Data
- Feature Engineering
- Modelling
- Comparing Top 2 Model
- Model Selection
- Evaluating Model
- Conclusion

Background

- Reddit is a forum consisting different subreddits

- Each subreddits are of a topic of interest

- Users can participate in subreddits which interests them

- Amongst the subreddits there are overlapping topics

- There are also posts which could be placed in more relevant subreddits

Problem Statement

Business Problem:
- We have quite a number of overlapping subreddits. How can we ensure that posts in the subreddits are in the right place?

Data Science Problem
- Use Natural Language Processing
- Identify 2 overlapping subreddits with adequate text
- Create classification models to classify posts to their respective subreddit
- Evaluate the models and select the best model to answer the business problem

Obtaining Data and Cleaning Data

- Subreddit Learn Python and Learn Programming was selected

- Data from both subreddit was scraped using the Requests library

- Column selftext which contains the content of the posts is determined to contain features which will answer our problem

- Data was cleansed using the following libraries

  - BeautifulSoup

  - NLTK (Stopwords)

  - Regex
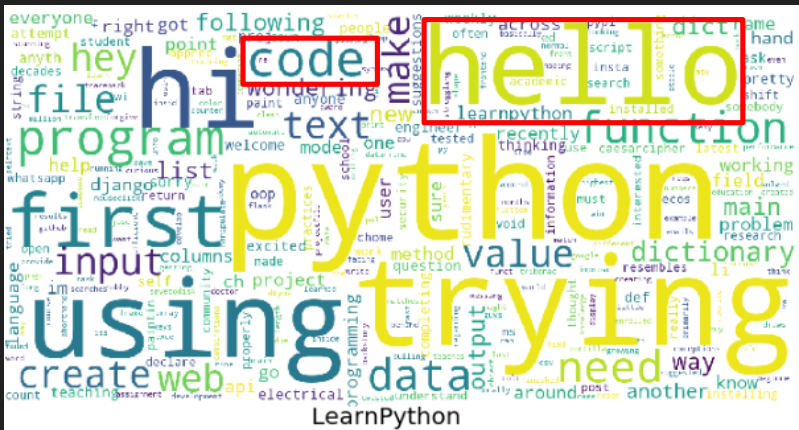
  - Python's string manipulation

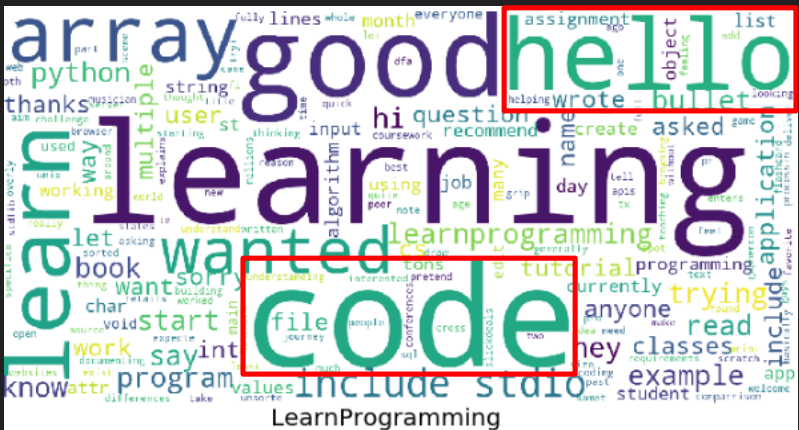# Exploring Data

After Obtaining and Cleaning Data
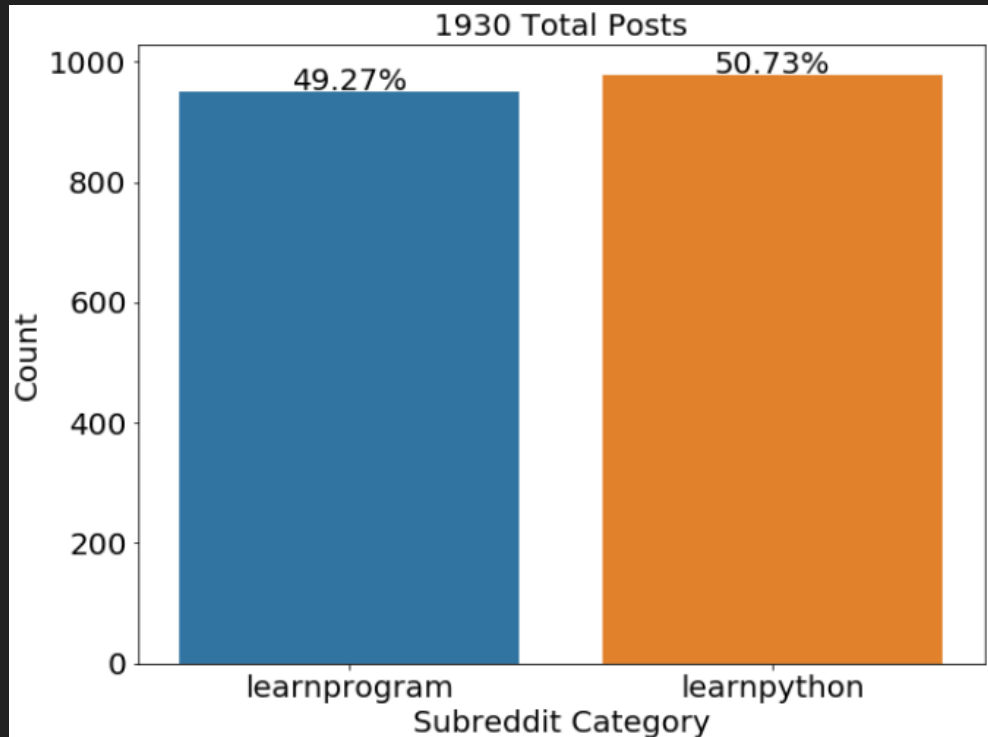
- We have a total of 1930 posts

- A good ratio between the 2 subreddits
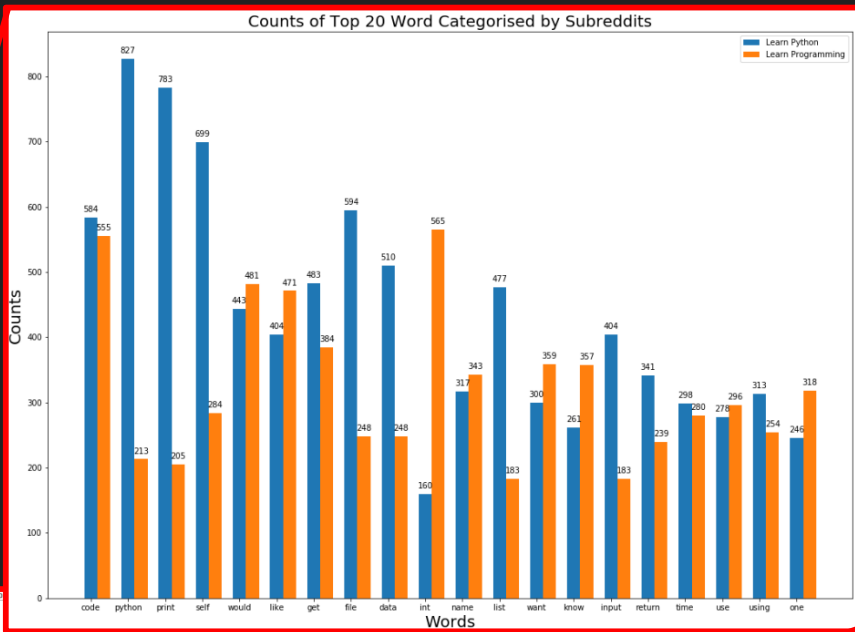
- Baseline Accuracy of 50.73%

Exploring Data

- "hello" and "code" are some common words which appears frequently in both subreddits

- We can explore more on such words and put them in a customized stop words

Exploring Data

Counts of Top 20 Word Categorised by Subreddits

- Top 20 words appearing in the corpus

- Code is a common word at both subreddit

- Python appeared in both but more in learn python

- Words like code should be included in our customized stopwords
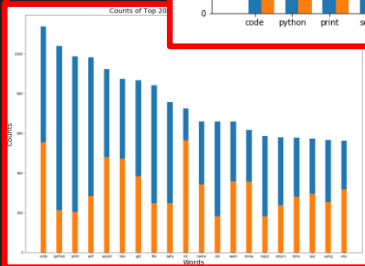
Feature Engineering

- Lemmatizing

  - Reduce words to their root keeping its context e.g. better → good

- Stemming

  - Removes suffixes and prefixes and may lose context e.g. care → car

- Customized Stopwords

```python
common_cv = common_cv.groupby('subreddit').sum()
common_words = []

for words in common_cv:
    if (abs(common_cv[words].loc[0] - common_cv[words].loc[1]) < 30):
        common_words.append(words)
```

```python
len(common_words)
```

646

Modelling

- 4 Models Was Built

    - Model 1 - Count Vectorizer with Naive Bayesian's MultinomialNB

    - Model 2 - Tfidf Vectorizer with Naive Bayesian's MultinomialNB

    - Model 3 - Count Vectorizer with Logistics Regression

    - Model 4 - Tfidf Vectorizer with Logistics Regression

- Model 2 and Model 3 were overfitted

- We will take a closer look into Model 1 and Model 4

Comparing Model 1 and Model 4

Model 1

```
56489 features were selected
Cross Validated Mean Train Score is: 0.7712720437726344
Test Score is: 0.7805383022774327
```

	pred learn programming	pred learn python
actual learn programming	192	46
actual learn python	60	185

```
              precision    recall  f1-score   support

           0       0.76      0.81      0.78       238
           1       0.80      0.76      0.78       245

    accuracy                           0.78       483
   macro avg       0.78      0.78      0.78       483
weighted avg       0.78      0.78      0.78       483
```

Model 4

```
56489 features were selected
Cross Validated Mean Train Score is: 0.7851084474885845
Test Score is: 0.7743271221532091
```

	pred learn programming	pred learn python
actual learn programming	183	55
actual learn python	54	191

```
              precision    recall  f1-score   support

           0       0.77      0.77      0.77       238
           1       0.78      0.78      0.78       245

    accuracy                           0.77       483
   macro avg       0.77      0.77      0.77       483
weighted avg       0.77      0.77      0.77       483
```

- Model 4 has a higher mean score between train and test
- Model 1 has higher precision

Model Selection

- We are looking to classify a post and move them to the correct subreddit.

- A post from learn python appearing in learn programming subreddit is reasonable as python is a programming language while the vice verse is not

- From this, we can see that the cost of false positive is higher than the cost of false negative.

- When the cost of false positive is high, precision is a good metrics to follow

Evaluating - Model 1

```
56489 features were selected
Cross Validated Mean Train Score is: 0.7712720437726344
Test Score is: 0.7805383022774327
```

	pred learn programming	pred learn python
actual learn programming	192	46
actual learn python	60	185

|	precision	recall	f1-score	support
0	0.76	0.81	0.78	238
1	0.80	0.76	0.78	245
accuracy			0.78	483
macro avg	0.78	0.78	0.78	483
weighted avg	0.78	0.78	0.78	483

- 192 posts predicted to be from learn programming are from learn programming
- 46 posts predicted to be from learn python are from learn programming
- 60 posts predicted to be from learn programming are from learn python
- 185 posts predicted to be from learn python are indeed from learn python
- This model's precision is 80% and the recall is 76%.

Evaluating - Model 1

Distribution of P(Outcome = Learn Python)
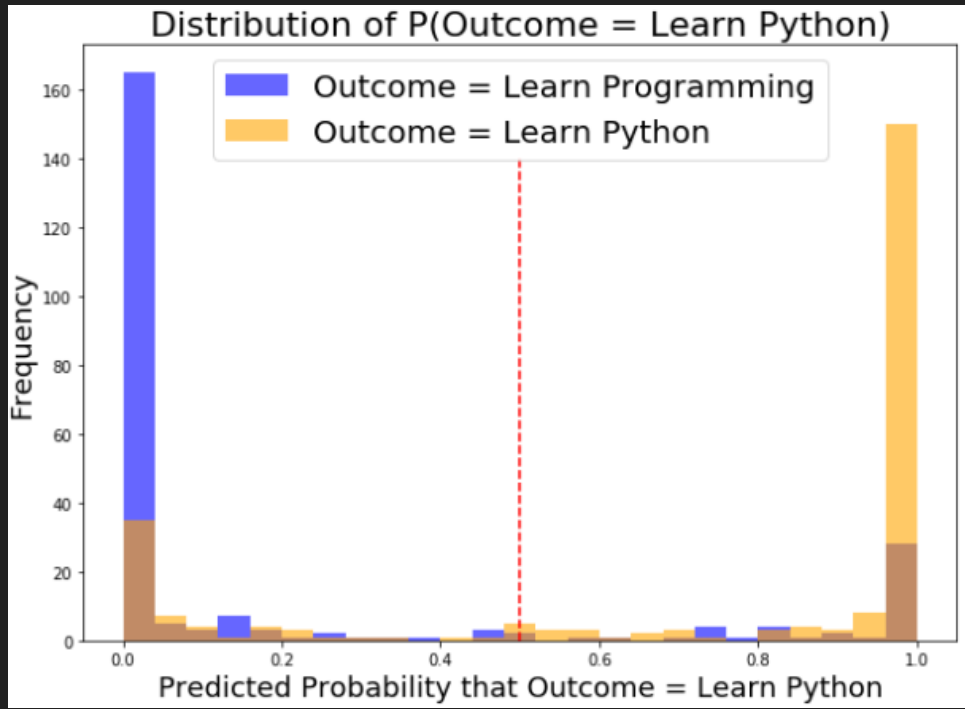
- Predicted probabilities distribution

- Bimodal distribution

- Misclassifications

  - Blues appearing at the right of the red line

  - Orange appearing at the left of the red line

Evaluating - Model 1

ROC Curve with AUC = 0.843
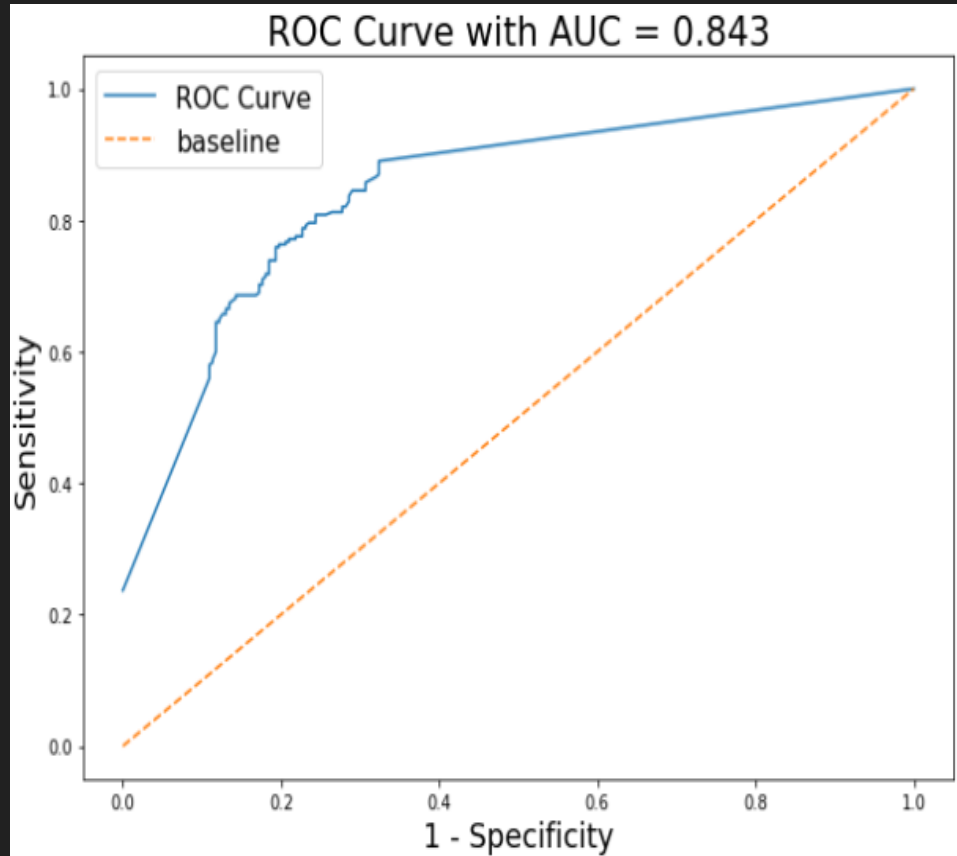
- A good model's ROC AUC is above 0.5 and close to 1

- A ROC AUC of 0.5, that means that our positive and negative population overlaps perfectly, inferring that the model is bad

- If the ROC AUC is below 0.5, it means that our model inversely classified the observations

- Our ROC AUC is 0.843

Conclusion and Recommendation

- A model with the lower score was selected as it answers the business needs

- For this instance a higher precision and lower false positive

- Business can use this model and move posts for the 2 subreddits

- With lesser overlapping posts we can then run the next iteration and determine if the model improves

- We can perform an analysis on the title, number of comments and author to see if these are good features to be included for the next iteration

- We can explore auto-tagging or subreddit recommender system with improved model