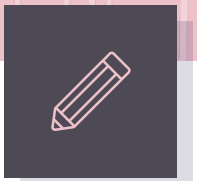


Predicting Subreddit from Posts

By: Yasmine Hays

The Problem

?

A cosmetics company is using subreddit posts to find the best way to advertise new products. When collecting data they forgot to specify which subreddit each post came from, so I have been contracted to design a model that can make that prediction.


CHICKS BE LIKE
HAIR TIED, CHILLIN
WITH NO MAKEUP ON

Gathering the Data

In order to get the necessary data a function was created which used the Pushshift API to contact Reddit and:

1). Gather 700 posts from the subreddit “haircareexchange”

2). Gather 700 posts from the subreddit “makeupexchange”

Once all the data was collected it was stored into a single datafrmae and used for the analysis

Cleaning the Data

- The initial dataset was narrowed down to only the columns containing text

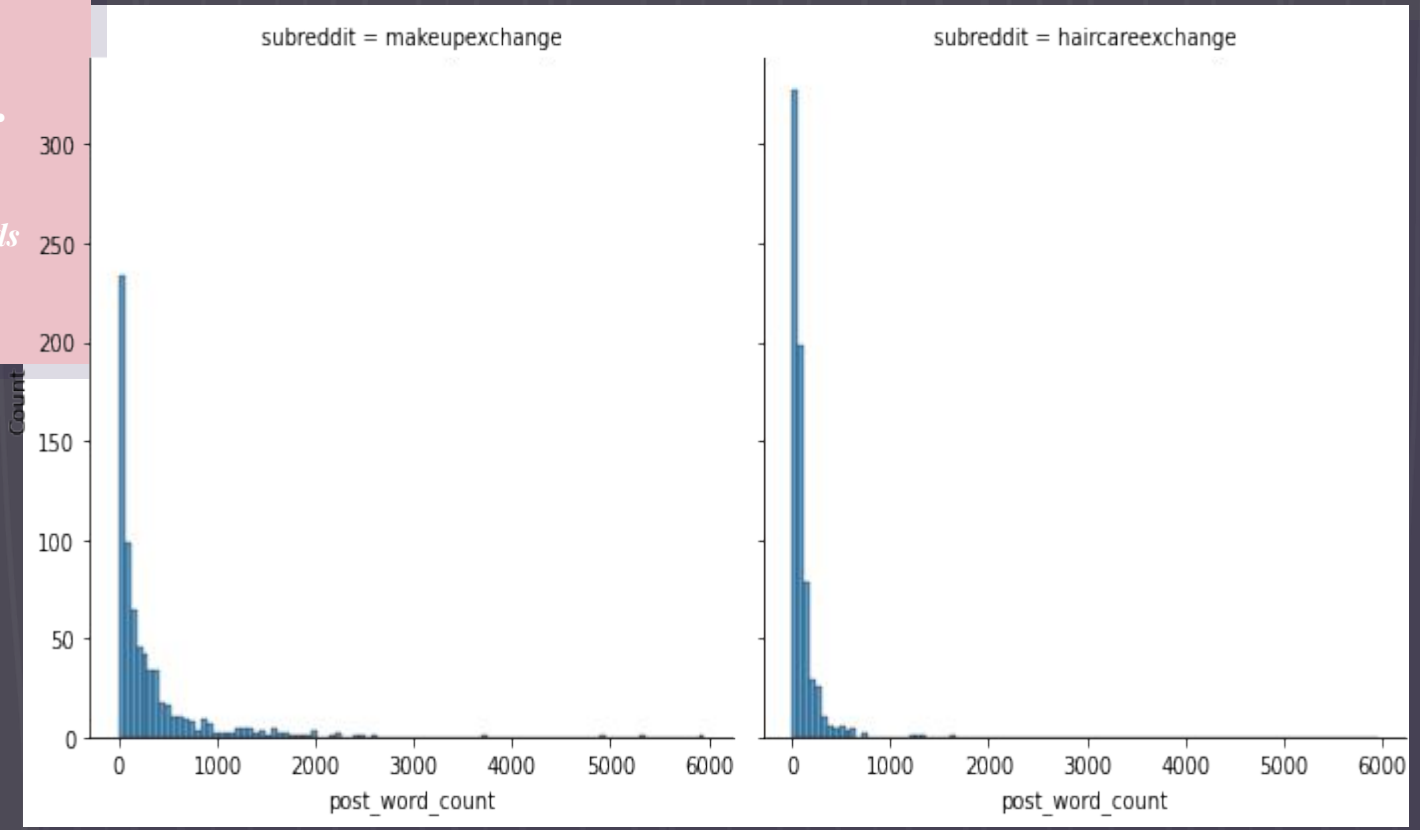
  * Subreddit
  * Author
  *Selftext
  * Title
  *Domain

- All missing values were dropped and columns containing the word count per post and the word count per title were created

- The final dataset consisted of 7 columns and 1396 rows of data

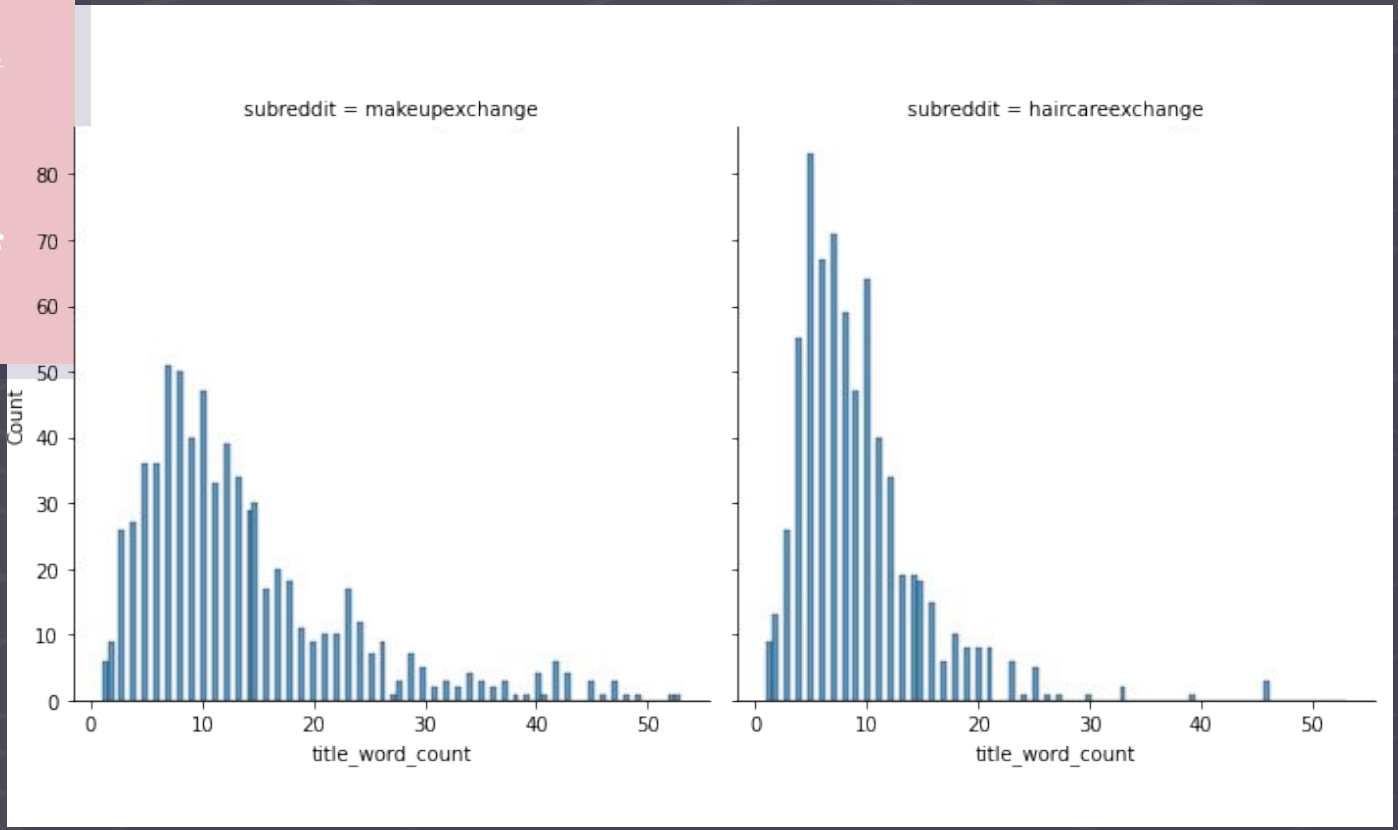
EDA: WORD COUNT BY POST

Range: 0 - 5956 words



EDA: Word Count Per Title

Range: 1 - 53 words



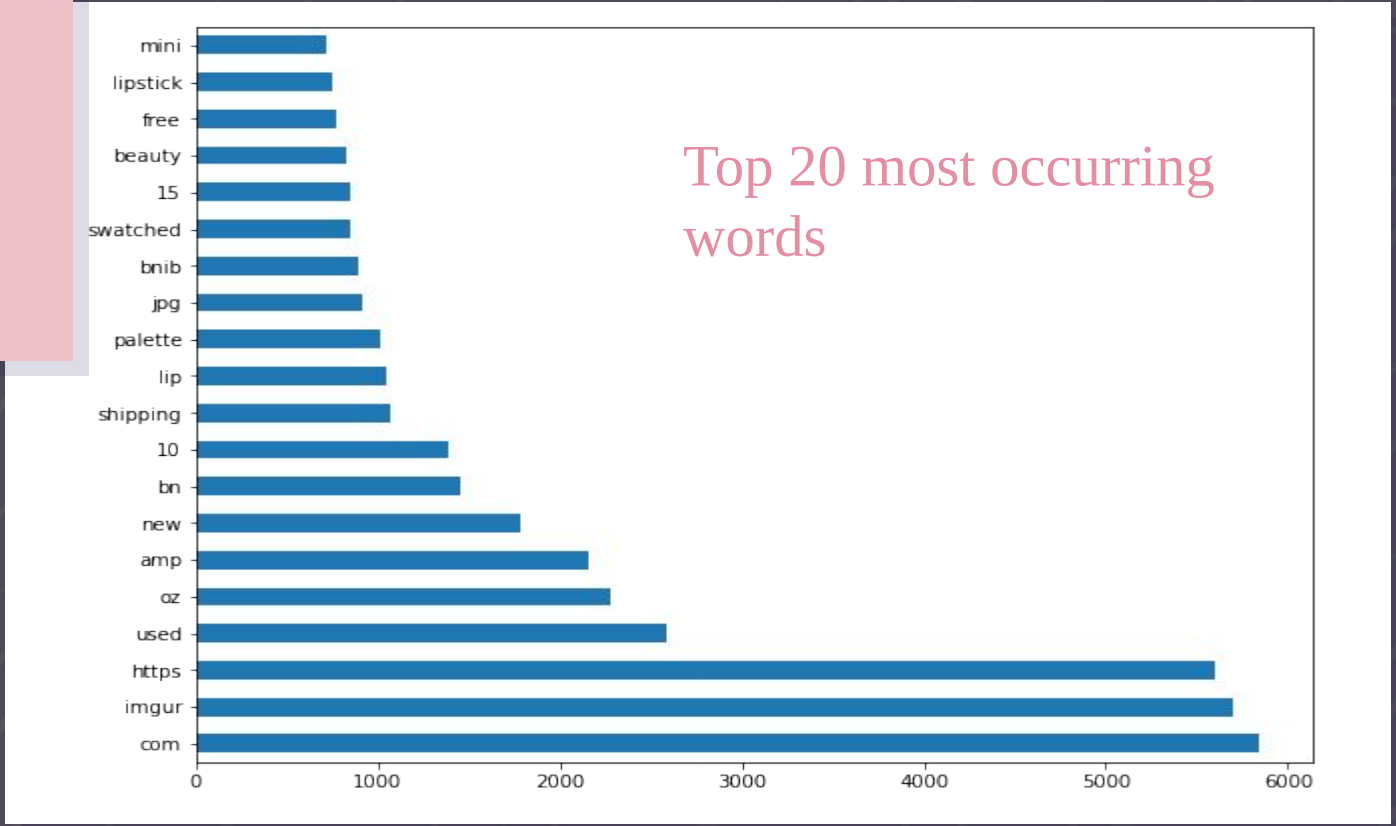
The client was informed that the post titles had a better spread of words and could likely result be a better predictor

But they wanted to base the model on posts only so I proceeded under their direction and made a training and testing set with “Self text” as my X variable

Preprocessing:

COUNT VECTORIZER

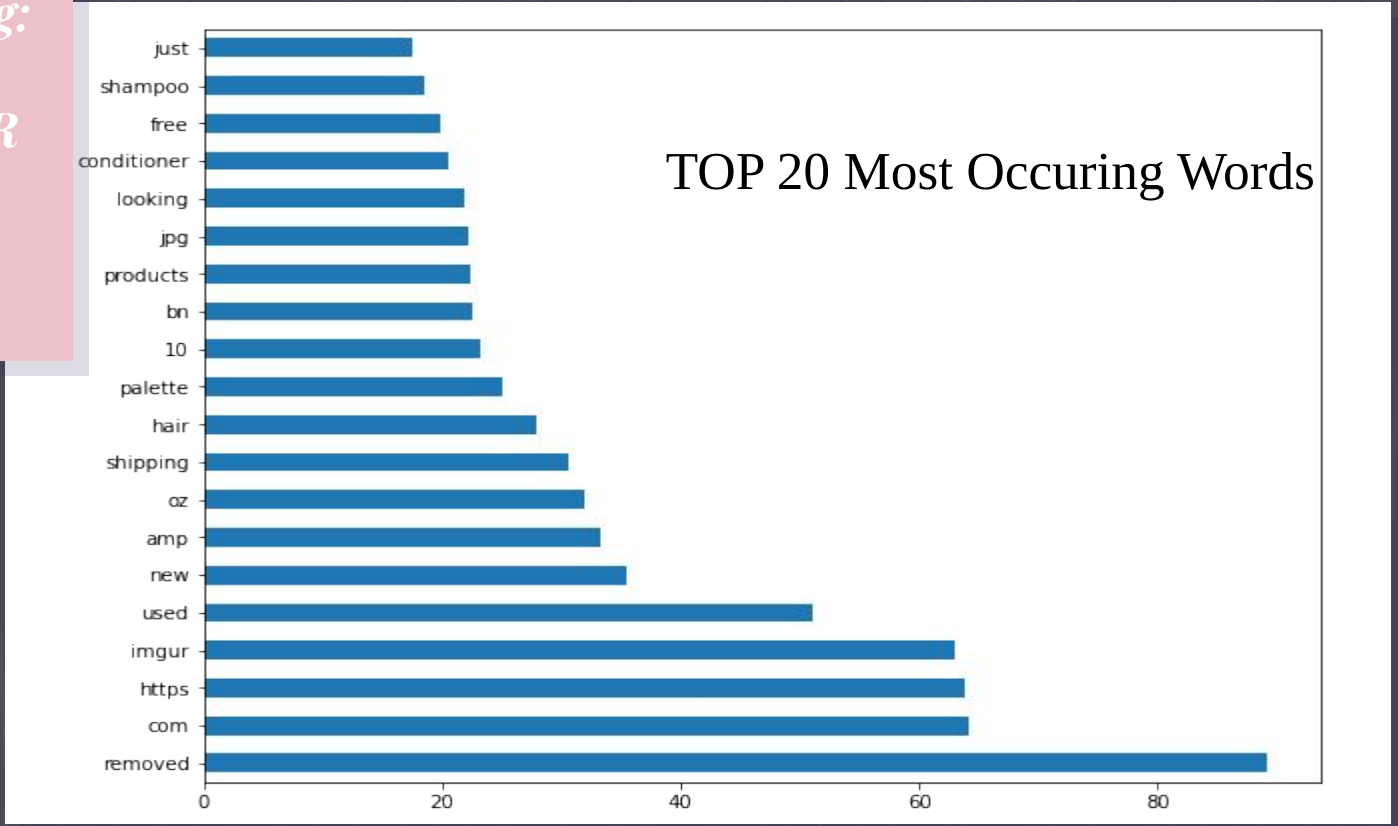
(stop words removed)



Top 20 most occurring words

Preprocessing: TFIDF VECTORIZER

(stop words removed)



Modeling

Pipe 1

Training Score : 0.968
Testing Score : 0.959

CV + LR

Pipe 2

CV + MNB

Training Score : 0.893
Testing Score : 0.888

TF + LR

Training Score : 0.973
Testing Score : 0.959

WINNER!!

Pipe 3

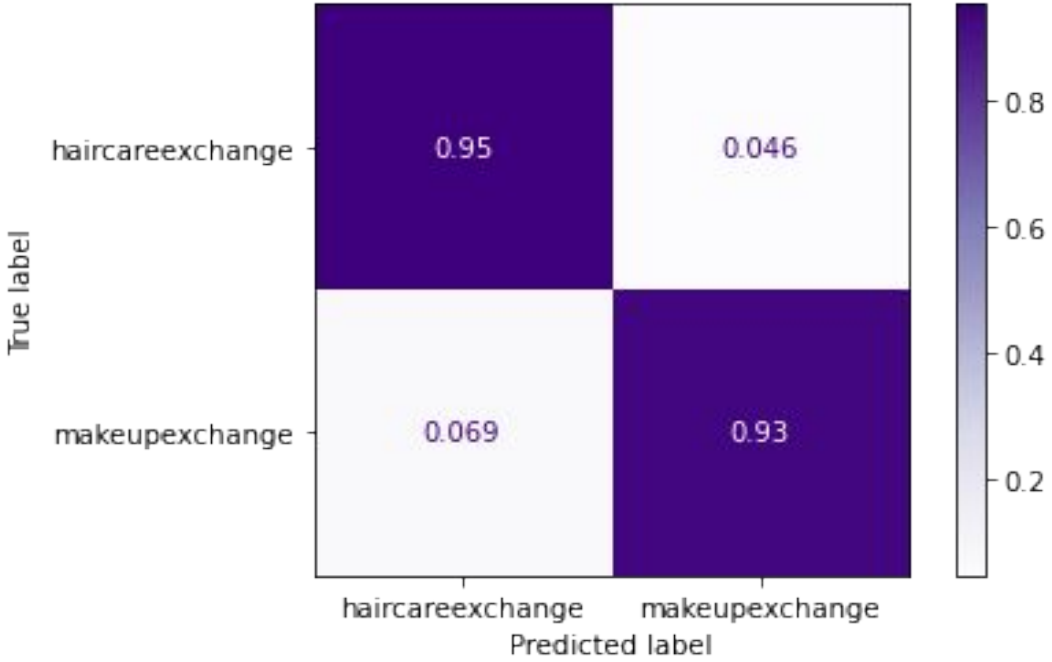
TF + MNB

Training Score : 0.957
Testing Score : 0.959

Pipe 4

.932157666689

The best score after performing a gridsearch on my best model for max_features, n_gram range and stop_words



Specificity (True Negative): 0.954

Sensitivity (True Positive) : 0.931

Baseline Accuracy:

haircareexchange	0.501433
makeupexchange	0.498567

How the model did on the testing data

Conclusions and Recommendations

- The model performed fairly well, but could potentially be improved by
  - Incorporating the “title” column into to predictor
  - Experimenting with more parameters and hyperparameters
  - Pickling the model so that it can be used by the common client

Thanks!

Any questions?