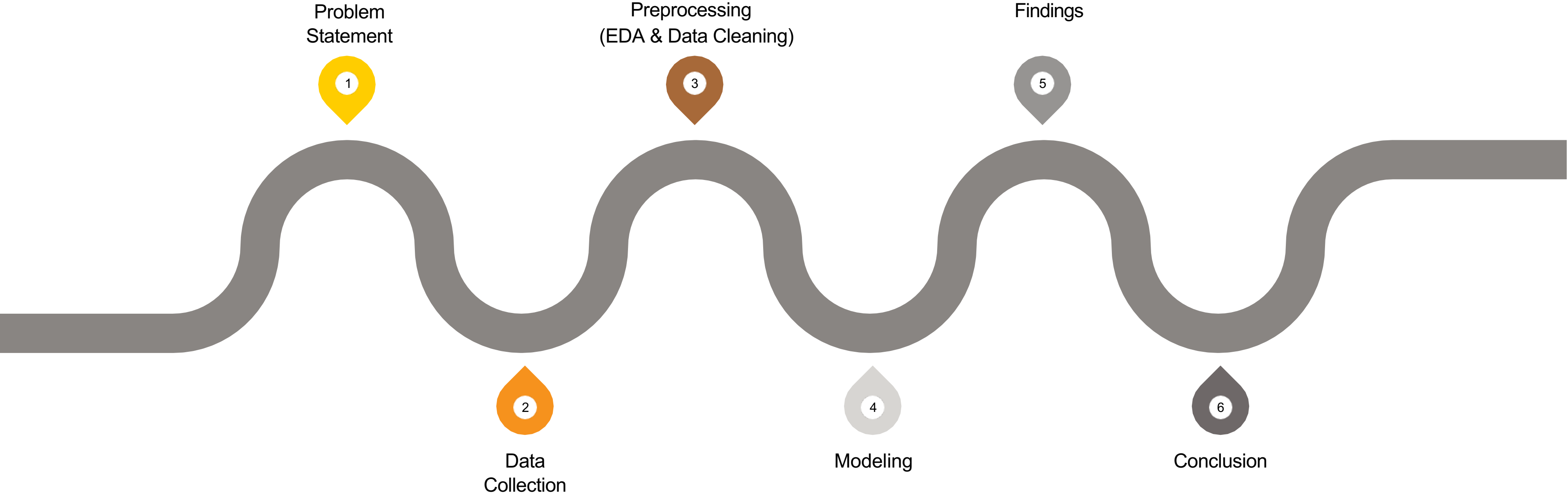
# Subreddit Classification

# Roadmap


Problem Statement
1
Preprocessing (EDA & Data Cleaning)
3
Findings
5
2
Data Collection
4
Modeling
6
Conclusion

# Introduction

## PROBLEM STATEMENT

We aim to answer whether we can build a Classification model to accurately predict if a post belongs to one subreddit or the other. Should we be able to, we next seek to determine which model works best and why.

# Data Collection

Using Pushshift’s API, we performed webscrapping to collect posts from two subreddits over the year 2021.

The subreddits are ‘PremierLeague’ and ‘mma’

**Premier League**

Most popular football league and the most-watched sports league in the world

8290 posts extracted

**MMA**

MMA stands for Mixed Martial Arts, which is a full contact sport based on striking, grappling and ground fighting

6514 posts extracted

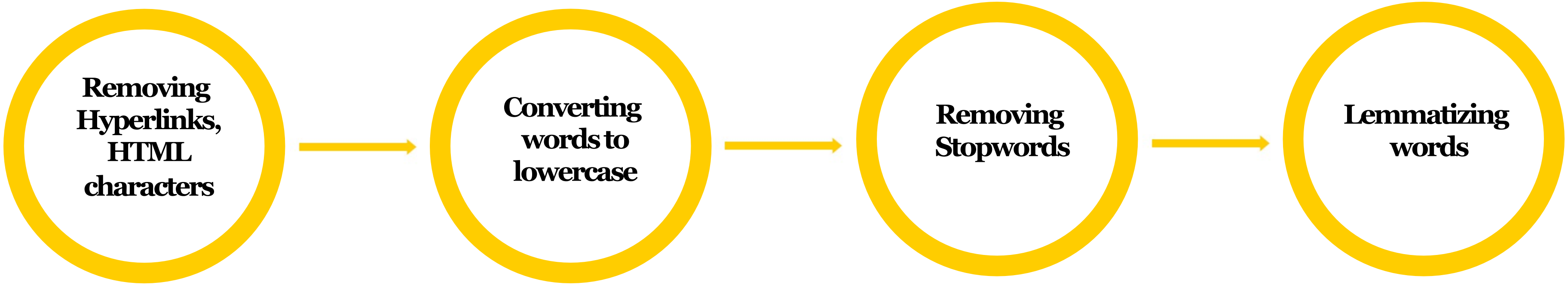
# Data Cleaning

Drop duplicate posts

Fill Null values

Delete ‘[removed]’ ‘[deleted]’

# Preprocessing



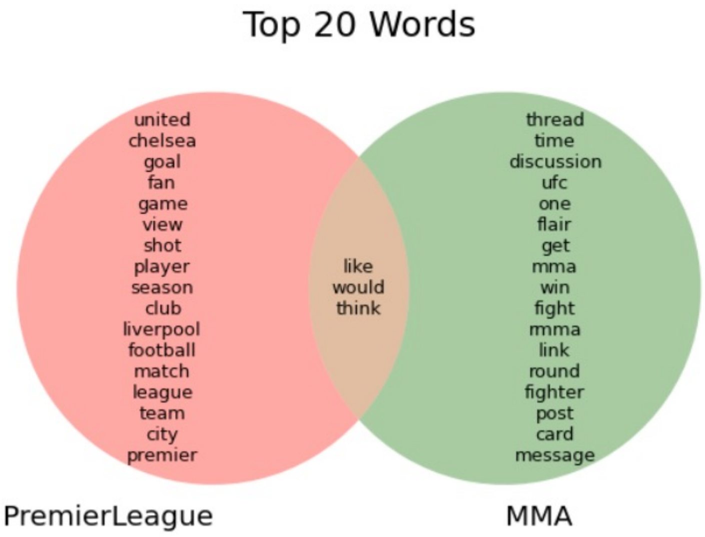
Removing Hyperlinks, HTML characters

Converting words to lowercase

Removing Stopwords

Lemmatizing words

# EDA

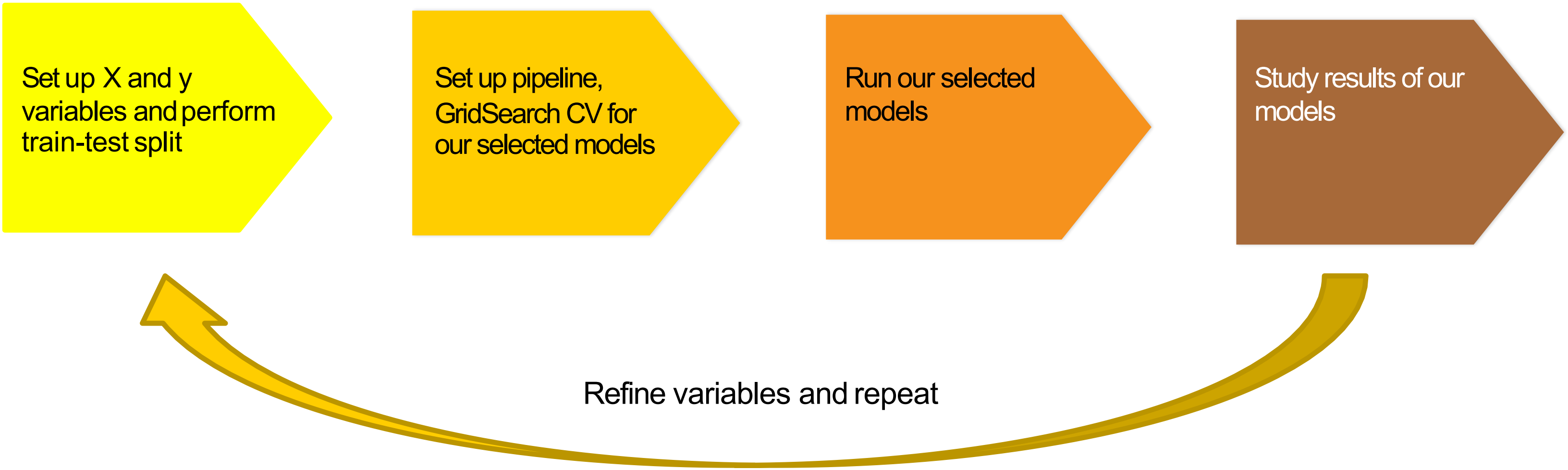


The words ‘like’, ‘would’, ‘think’ appear frequently in both subreddits.

Under MMA, a few words stand out. They are ‘thread’, ‘discussion’, ‘post’, ‘message’. Judging from the nature of such words, these words should appear quite frequently in all the subreddits.

Add these words to StopWords

# Model Workflow



Set up X and y variables and perform train-test split

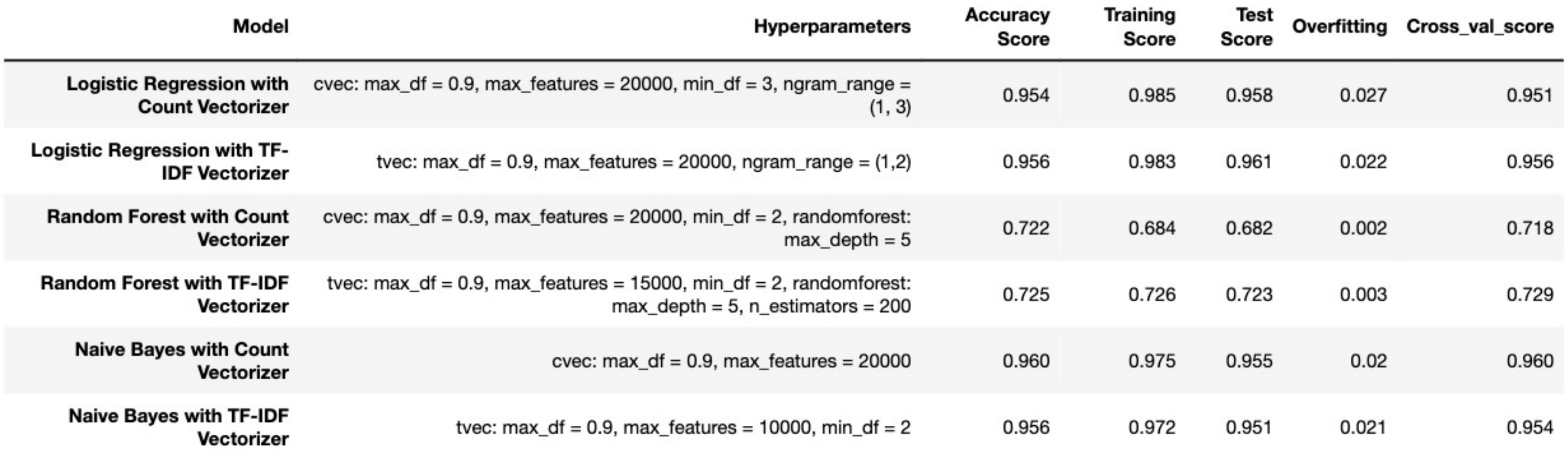
Set up pipeline, GridSearch CV for our selected models

Run our selected models

Study results of our models

Refine variables and repeat

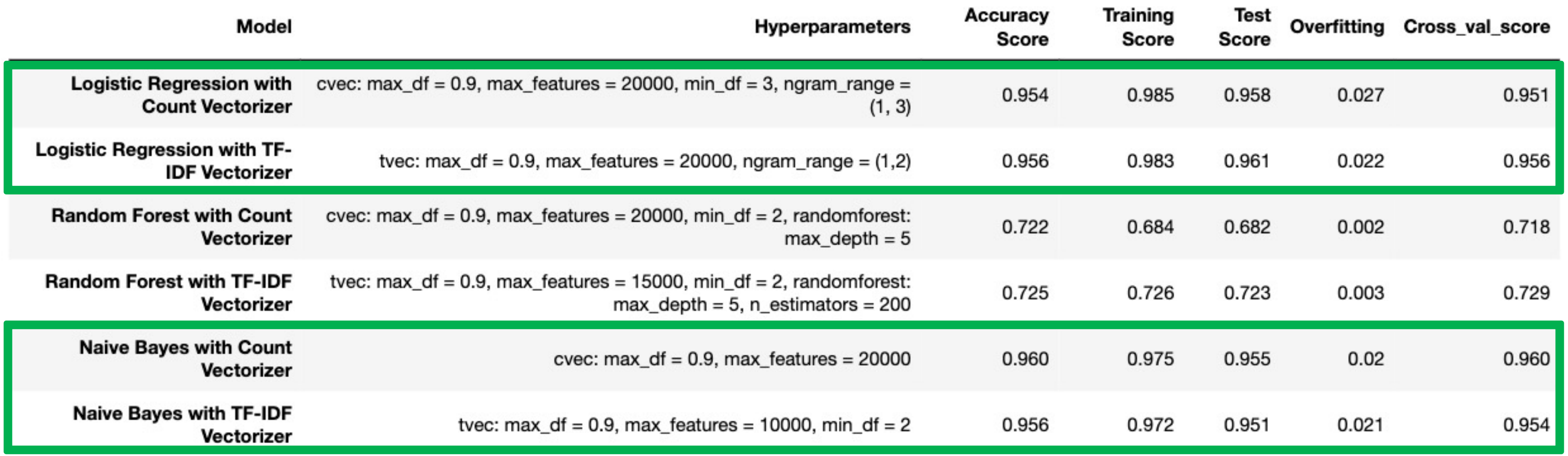
# Findings



| Model | Hyperparameters | Accuracy Score | Training Score | Test Score | Overfitting | Cross_val_score |
|---|---|---|---|---|---|---|
| Logistic Regression with Count Vectorizer | cvec: max_df = 0.9, max_features = 20000, min_df = 3, ngram_range = (1, 3) | 0.954 | 0.985 | 0.958 | 0.027 | 0.951 |
| Logistic Regression with TF-IDF Vectorizer | tvec: max_df = 0.9, max_features = 20000, ngram_range = (1,2) | 0.956 | 0.983 | 0.961 | 0.022 | 0.956 |
| Random Forest with Count Vectorizer | cvec: max_df = 0.9, max_features = 20000, min_df = 2, randomforest: max_depth = 5 | 0.722 | 0.684 | 0.682 | 0.002 | 0.718 |
| Random Forest with TF-IDF Vectorizer | tvec: max_df = 0.9, max_features = 15000, min_df = 2, randomforest: max_depth = 5, n_estimators = 200 | 0.725 | 0.726 | 0.723 | 0.003 | 0.729 |
| Naive Bayes with Count Vectorizer | cvec: max_df = 0.9, max_features = 20000 | 0.960 | 0.975 | 0.955 | 0.02 | 0.960 |
| Naive Bayes with TF-IDF Vectorizer | tvec: max_df = 0.9, max_features = 10000, min_df = 2 | 0.956 | 0.972 | 0.951 | 0.021 | 0.954 |

# Findings

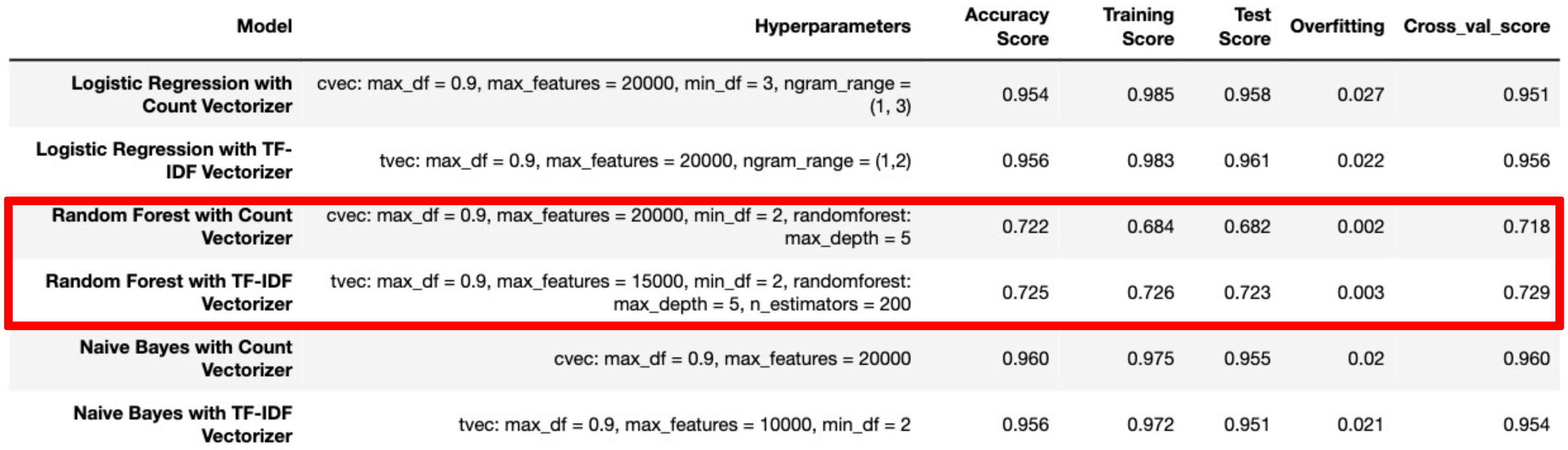
- Using ***Logistic Regression*** and ***Naïve Bayes*** allow us to accurately classify the posts to their respective subreddits



| Model | Hyperparameters | Accuracy Score | Training Score | Test Score | Overfitting | Cross_val_score |
|---|---|---|---|---|---|---|
| **Logistic Regression with Count Vectorizer** | cvec: max_df = 0.9, max_features = 20000, min_df = 3, ngram_range = (1, 3) | 0.954 | 0.985 | 0.958 | 0.027 | 0.951 |
| **Logistic Regression with TF-IDF Vectorizer** | tvec: max_df = 0.9, max_features = 20000, ngram_range = (1,2) | 0.956 | 0.983 | 0.961 | 0.022 | 0.956 |
| **Random Forest with Count Vectorizer** | cvec: max_df = 0.9, max_features = 20000, min_df = 2, randomforest: max_depth = 5 | 0.722 | 0.684 | 0.682 | 0.002 | 0.718 |
| **Random Forest with TF-IDF Vectorizer** | tvec: max_df = 0.9, max_features = 15000, min_df = 2, randomforest: max_depth = 5, n_estimators = 200 | 0.725 | 0.726 | 0.723 | 0.003 | 0.729 |
| **Naive Bayes with Count Vectorizer** | cvec: max_df = 0.9, max_features = 20000 | 0.960 | 0.975 | 0.955 | 0.02 | 0.960 |
| **Naive Bayes with TF-IDF Vectorizer** | tvec: max_df = 0.9, max_features = 10000, min_df = 2 | 0.956 | 0.972 | 0.951 | 0.021 | 0.954 |

# Findings

- ***Random Forest*** performs poorly

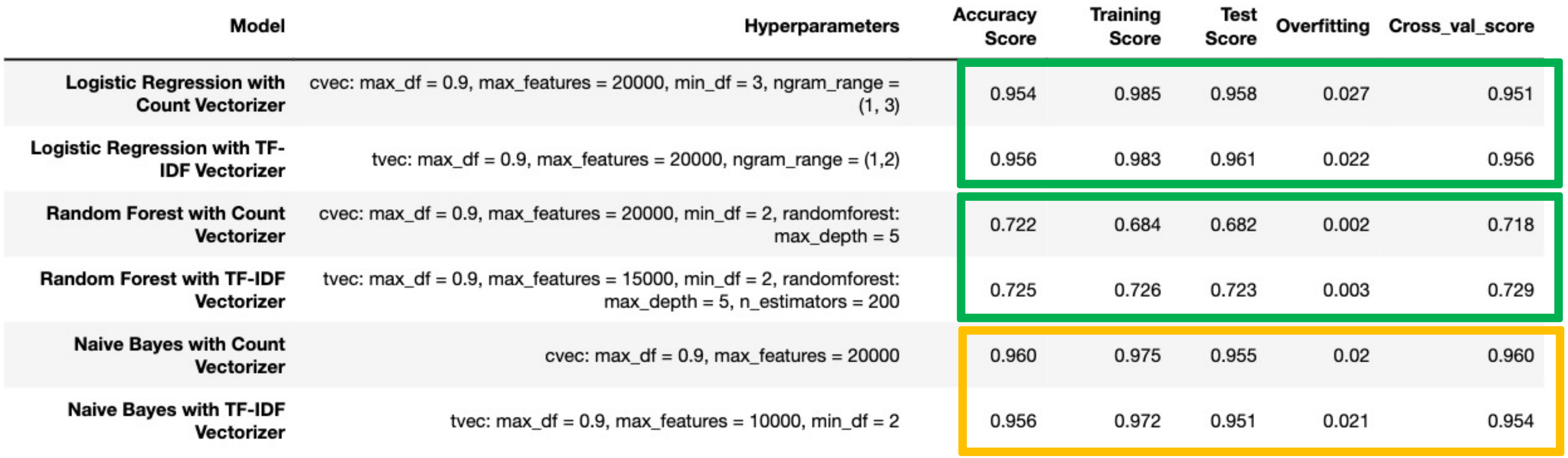


| Model | Hyperparameters | Accuracy Score | Training Score | Test Score | Overfitting | Cross_val_score |
|---|---|---|---|---|---|---|
| Logistic Regression with Count Vectorizer | cvec: max_df = 0.9, max_features = 20000, min_df = 3, ngram_range = (1, 3) | 0.954 | 0.985 | 0.958 | 0.027 | 0.951 |
| Logistic Regression with TF-IDF Vectorizer | tvec: max_df = 0.9, max_features = 20000, ngram_range = (1,2) | 0.956 | 0.983 | 0.961 | 0.022 | 0.956 |
| Random Forest with Count Vectorizer | cvec: max_df = 0.9, max_features = 20000, min_df = 2, randomforest: max_depth = 5 | 0.722 | 0.684 | 0.682 | 0.002 | 0.718 |
| Random Forest with TF-IDF Vectorizer | tvec: max_df = 0.9, max_features = 15000, min_df = 2, randomforest: max_depth = 5, n_estimators = 200 | 0.725 | 0.726 | 0.723 | 0.003 | 0.729 |
| Naive Bayes with Count Vectorizer | cvec: max_df = 0.9, max_features = 20000 | 0.960 | 0.975 | 0.955 | 0.02 | 0.960 |
| Naive Bayes with TF-IDF Vectorizer | tvec: max_df = 0.9, max_features = 10000, min_df = 2 | 0.956 | 0.972 | 0.951 | 0.021 | 0.954 |

- Works best when
  - there is low correlation between the trees, so that the trees protect each other from individual errors
  - there are multi classes

# Findings

- ***TF-IDF*** > Count Vectorizer

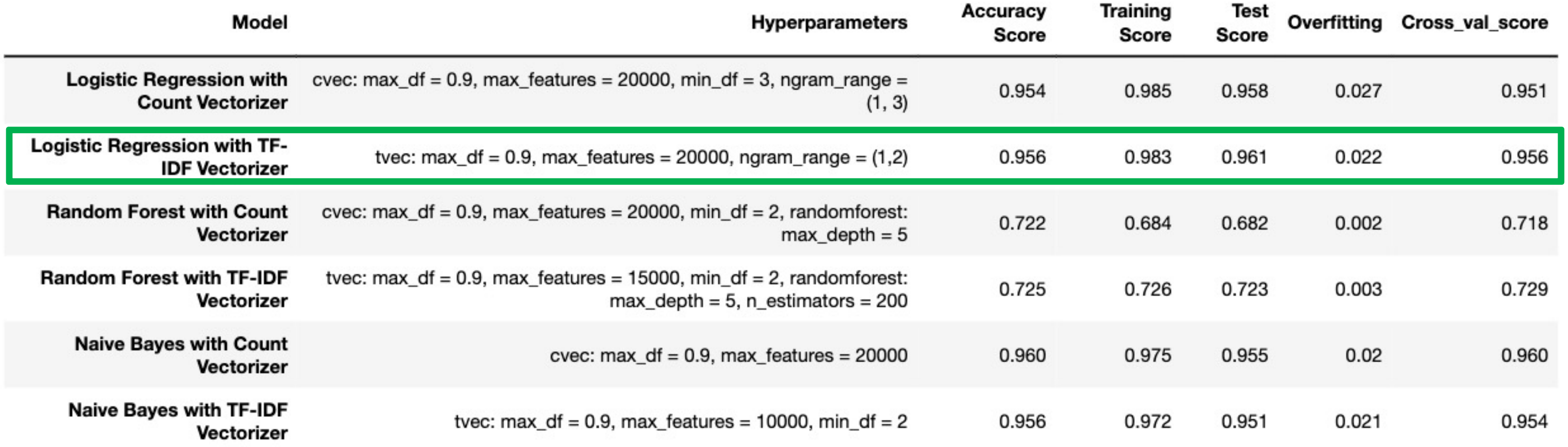


| Model | Hyperparameters | Accuracy Score | Training Score | Test Score | Overfitting | Cross_val_score |
|---|---|---|---|---|---|---|
| **Logistic Regression with Count Vectorizer** | cvec: max_df = 0.9, max_features = 20000, min_df = 3, ngram_range = (1, 3) | 0.954 | 0.985 | 0.958 | 0.027 | 0.951 |
| **Logistic Regression with TF-IDF Vectorizer** | tvec: max_df = 0.9, max_features = 20000, ngram_range = (1,2) | 0.956 | 0.983 | 0.961 | 0.022 | 0.956 |
| **Random Forest with Count Vectorizer** | cvec: max_df = 0.9, max_features = 20000, min_df = 2, randomforest: max_depth = 5 | 0.722 | 0.684 | 0.682 | 0.002 | 0.718 |
| **Random Forest with TF-IDF Vectorizer** | tvec: max_df = 0.9, max_features = 15000, min_df = 2, randomforest: max_depth = 5, n_estimators = 200 | 0.725 | 0.726 | 0.723 | 0.003 | 0.729 |
| **Naive Bayes with Count Vectorizer** | cvec: max_df = 0.9, max_features = 20000 | 0.960 | 0.975 | 0.955 | 0.02 | 0.960 |
| **Naive Bayes with TF-IDF Vectorizer** | tvec: max_df = 0.9, max_features = 10000, min_df = 2 | 0.956 | 0.972 | 0.951 | 0.021 | 0.954 |

- TF-IDF fixes some issues that Count Vectorizer faces
  - ➢ Helps us tease us relevance of words instead of just using word count

# Findings

- ***Logistic Regression*** > Naïve Bayes
- Naïve Bayes makes a conditional independence assumption, which is violated when you have correlated/repetitive features

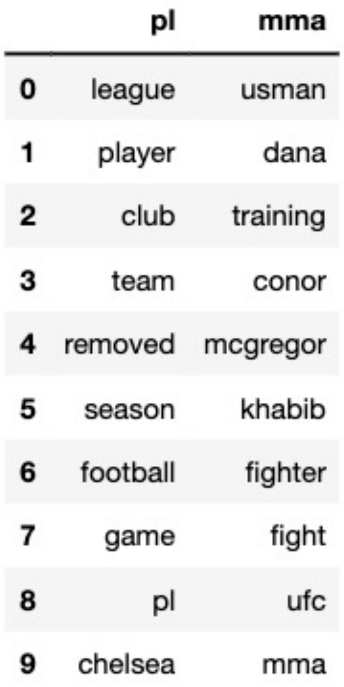


| Model | Hyperparameters | Accuracy Score | Training Score | Test Score | Overfitting | Cross_val_score |
|---|---|---|---|---|---|---|
| Logistic Regression with Count Vectorizer | cvec: max_df = 0.9, max_features = 20000, min_df = 3, ngram_range = (1, 3) | 0.954 | 0.985 | 0.958 | 0.027 | 0.951 |
| Logistic Regression with TF-IDF Vectorizer | tvec: max_df = 0.9, max_features = 20000, ngram_range = (1,2) | 0.956 | 0.983 | 0.961 | 0.022 | 0.956 |
| Random Forest with Count Vectorizer | cvec: max_df = 0.9, max_features = 20000, min_df = 2, randomforest: max_depth = 5 | 0.722 | 0.684 | 0.682 | 0.002 | 0.718 |
| Random Forest with TF-IDF Vectorizer | tvec: max_df = 0.9, max_features = 15000, min_df = 2, randomforest: max_depth = 5, n_estimators = 200 | 0.725 | 0.726 | 0.723 | 0.003 | 0.729 |
| Naive Bayes with Count Vectorizer | cvec: max_df = 0.9, max_features = 20000 | 0.960 | 0.975 | 0.955 | 0.02 | 0.960 |
| Naive Bayes with TF-IDF Vectorizer | tvec: max_df = 0.9, max_features = 10000, min_df = 2 | 0.956 | 0.972 | 0.951 | 0.021 | 0.954 |

- Some words might be correlated to others
  - ➢ Top words for Logistic Regression are the ones with the best predictors with the context of other words

# Findings

***Logistic Regression with TF-IDF Vectorizer***



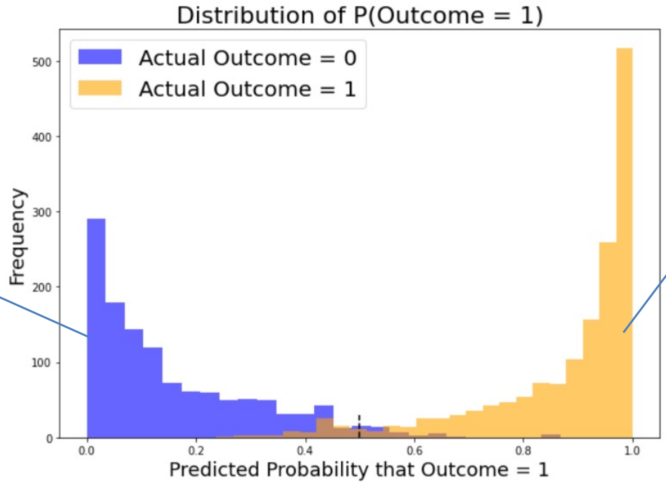
|   | pl | mma |
|---|---|---|
| 0 | league | usman |
| 1 | player | dana |
| 2 | club | training |
| 3 | team | conor |
| 4 | removed | mcgregor |
| 5 | season | khabib |
| 6 | football | fighter |
| 7 | game | fight |
| 8 | pl | ufc |
| 9 | chelsea | mma |

- Top 10 words in each subreddit that contribute the most in predicting whether a post belongs to ‘PremierLeague’ or ‘mma’

# Findings

***Logistic Regression with TF-IDF Vectorizer***

Blue area:
MMA posts that have been classified correctly



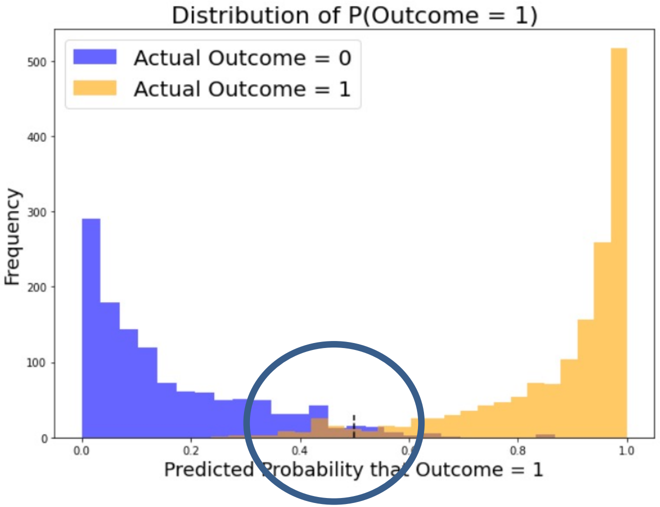
Orange area:
Premier League posts that have been classified correctly

# Findings

## ***Logistic Regression with TF-IDF Vectorizer***

Brown area:
Misclassified posts

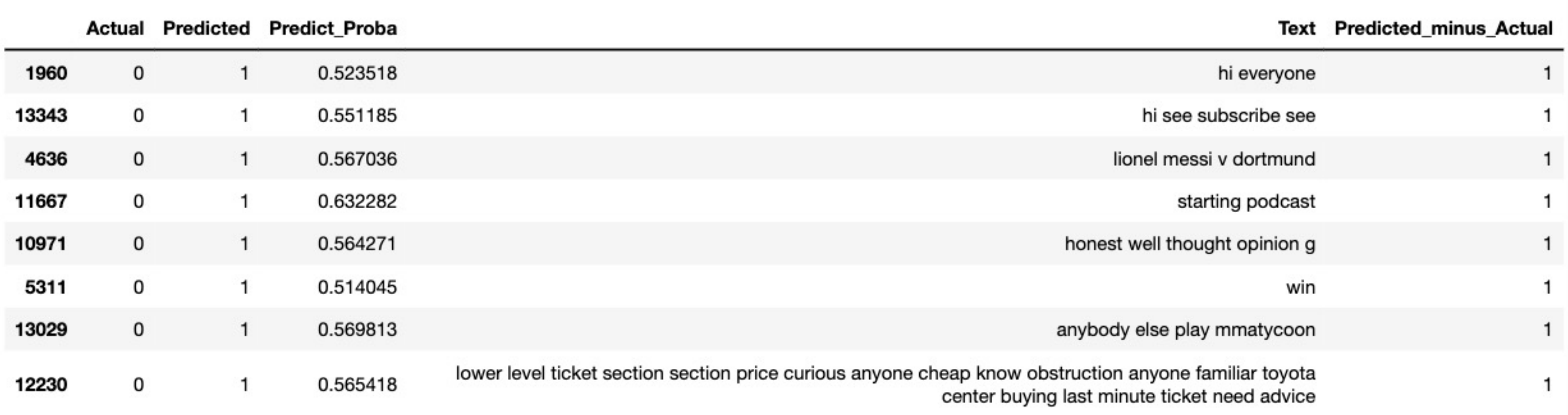
Misclassifications center around 0.5



# Findings

## ***Logistic Regression with TF-IDF Vectorizer***

Misclassified posts (False Positives)



| | Actual | Predicted | Predict_Proba | Text | Predicted_minus_Actual |
|---|---|---|---|---|---|
| **1960** | 0 | 1 | 0.523518 | hi everyone | 1 |
| **13343** | 0 | 1 | 0.551185 | hi see subscribe see | 1 |
| **4636** | 0 | 1 | 0.567036 | lionel messi v dortmund | 1 |
| **11667** | 0 | 1 | 0.632282 | starting podcast | 1 |
| **10971** | 0 | 1 | 0.564271 | honest well thought opinion g | 1 |
| **5311** | 0 | 1 | 0.514045 | win | 1 |
| **13029** | 0 | 1 | 0.569813 | anybody else play mmatycoon | 1 |
| **12230** | 0 | 1 | 0.565418 | lower level ticket section section price curious anyone cheap know obstruction anyone familiar toyota center buying last minute ticket need advice | 1 |

- Misclassified posts tend to contain very few words
  - ➢ Average number of word per post, after cleaning and preprocessing is ~ 30

# Conclusion

- *TF-IDF* > Count Vectorizer
- *Logistic Regression* > Naïve Bayes > Random Forest
- ***Logistic Regression with TF-IDF*** works best for our subreddit classification problem!

Future Steps:

- Test our model with a larger dataset

# Thanks!