



Spencer Buckner

Reddit Classification



Why are we here?

Problem: Do Classification models help correctly group Reddit Posts into respective buckets? If so, what works?

Subreddits:

- TalesFromRetail
- TalesFromTechSupport



Why these subreddits?

- Similar enough that topics are not complete opposites of each other

How is Data Collected?

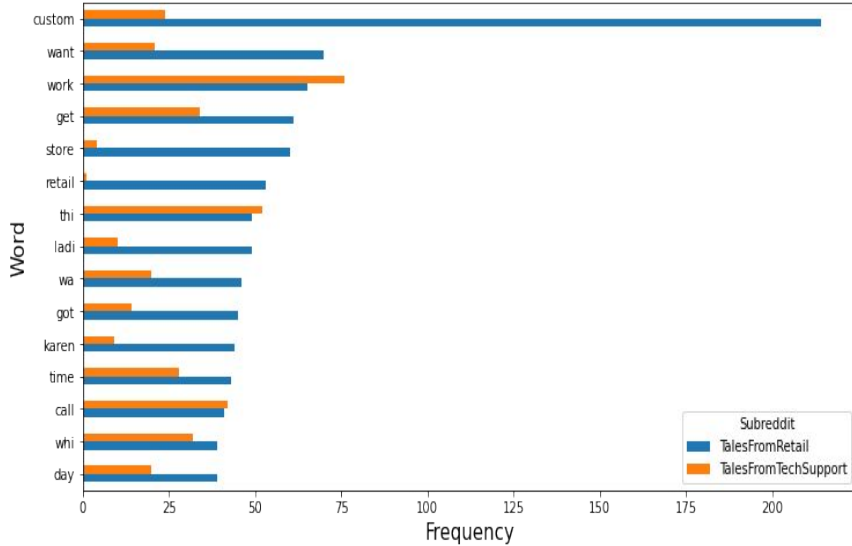
- Web Scraped using Pushshift Reddit API
 - ~1500 posts from each subreddit
 - Data Cleaning cuts ~40 rows from Retail and ~20 rows from Tech
 - Null Hypothesis: 50.18%
-

Exploratory Word Analysis

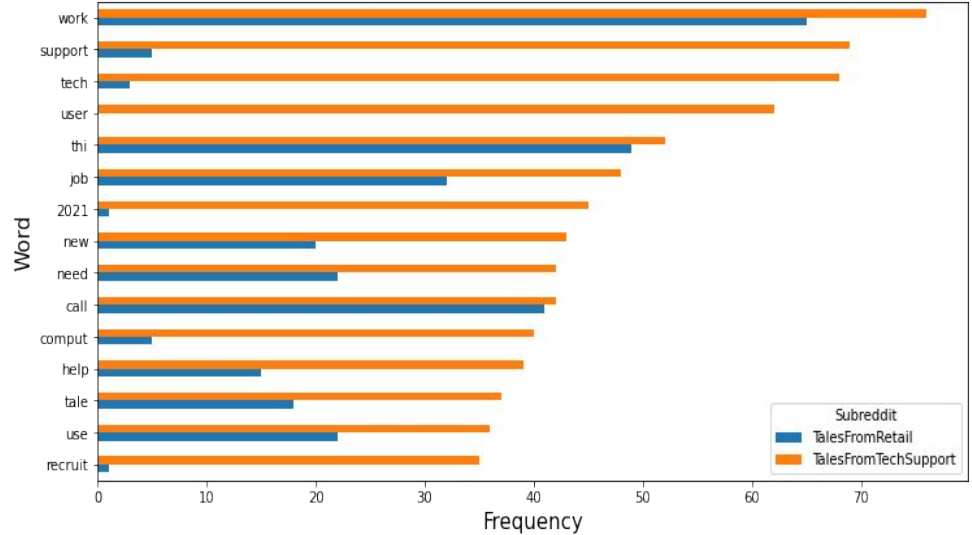
- Tokenize Title and Selftext columns
- Apply a Porter Stemmer to get to root word
- Compare top results sorted by Subreddit

Title Word Analysis

15 Most Common Title Words and their Frequency by Subreddit

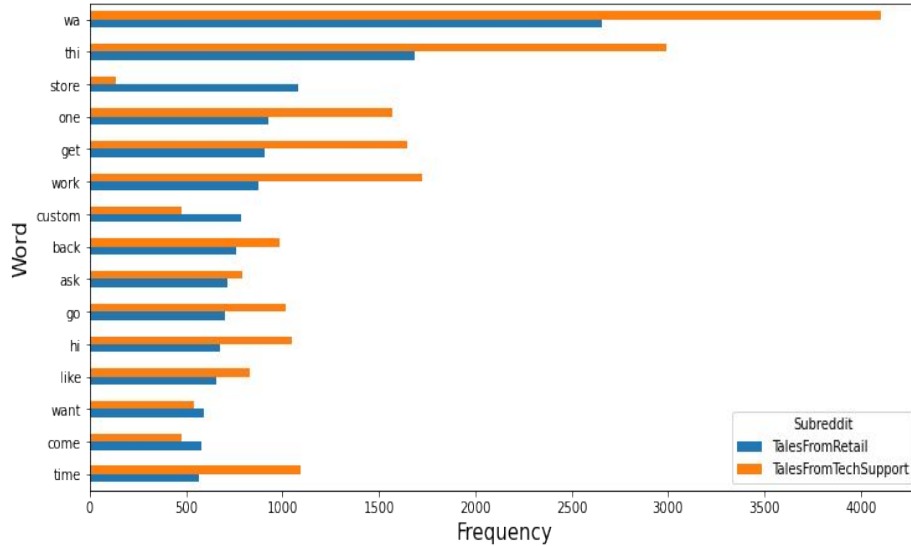


15 Most Common Title Words and their Frequency by Subreddit

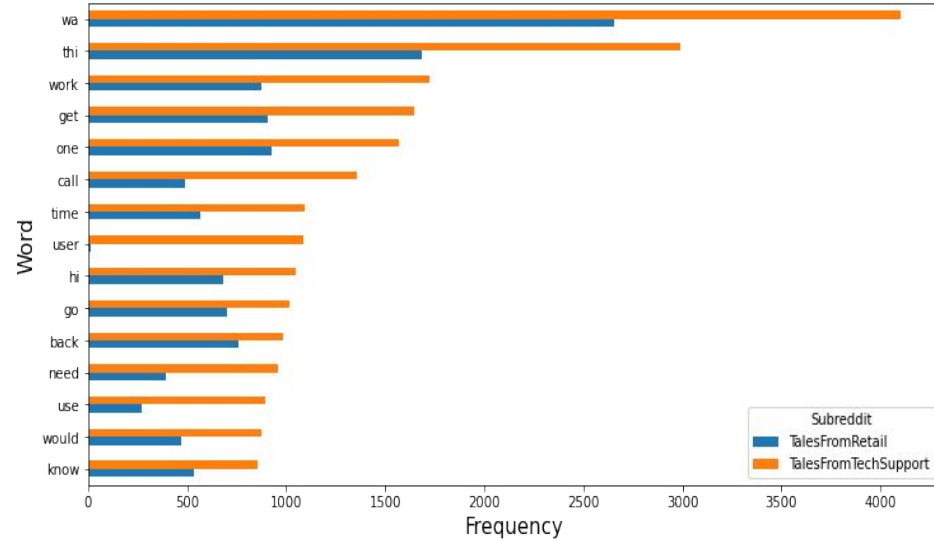


Selftext Word Analysis

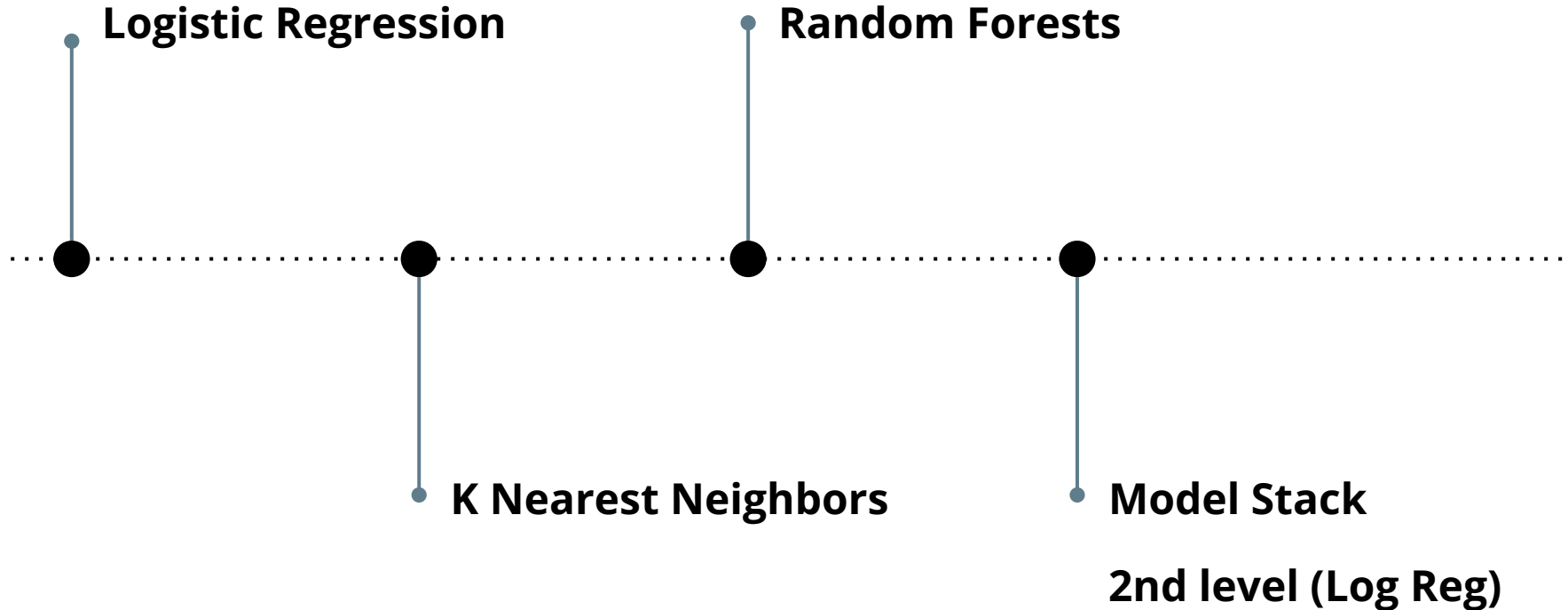
15 Most Common Selftext Words and their Frequency by Subreddit



15 Most Common Selftext Words and their Frequency by Subreddit

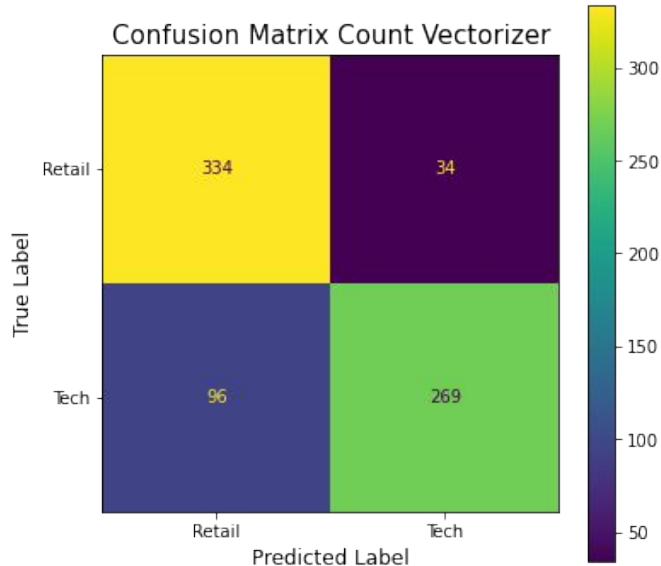


Modeling Process - All use Count/TfidfVectorizer
7 models total - All Created with Pipe/GridSearch

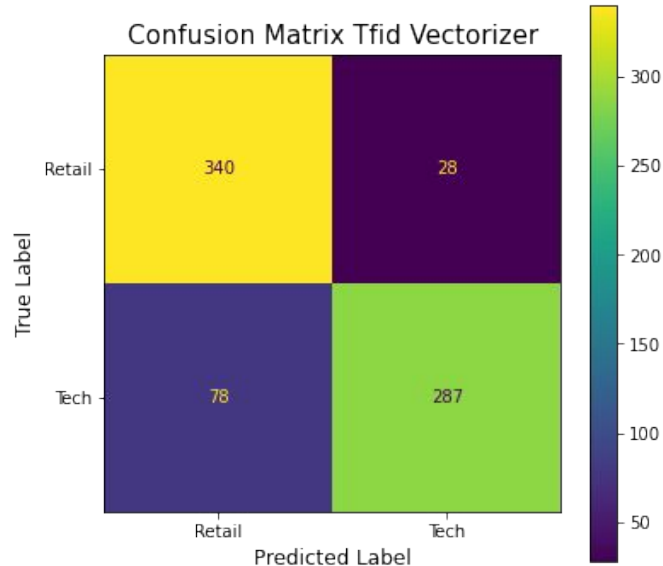


Logistic Regression

Cross Val : 0.8602
Accuracy : 82.26%
Specificity : 90.76%
Sensitivity : 73.70%
Precision : 88.78%

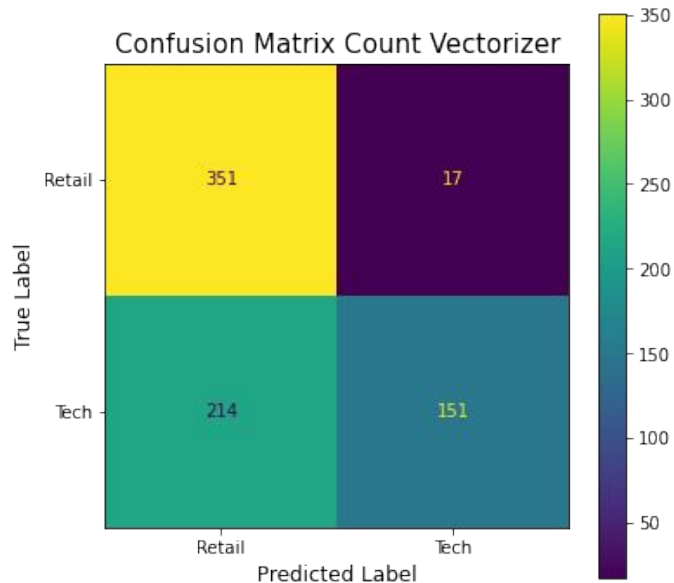


Cross Val : 0.8821
Accuracy : 85.54%
Specificity : 92.39%
Sensitivity : 78.63%
Precision : 91.11%

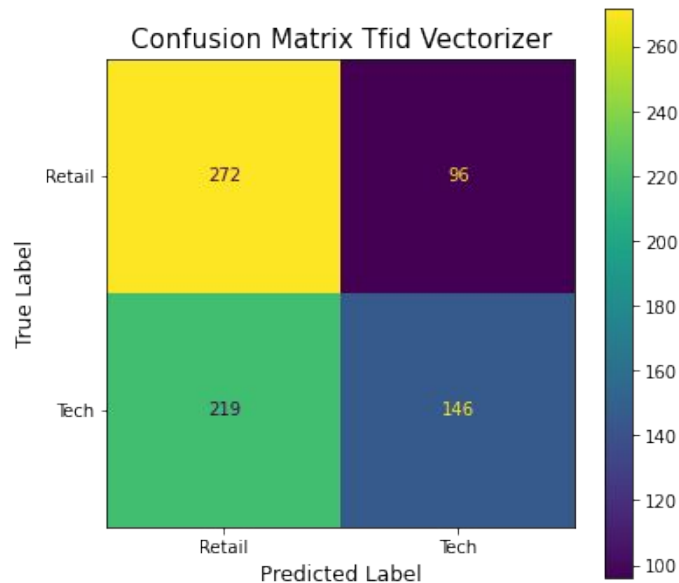


K Nearest Neighbors

Cross Val : .6708
Accuracy : 68.49%
Specificity : 95.38%
Sensitivity : 41.37%
Precision : 89.88%



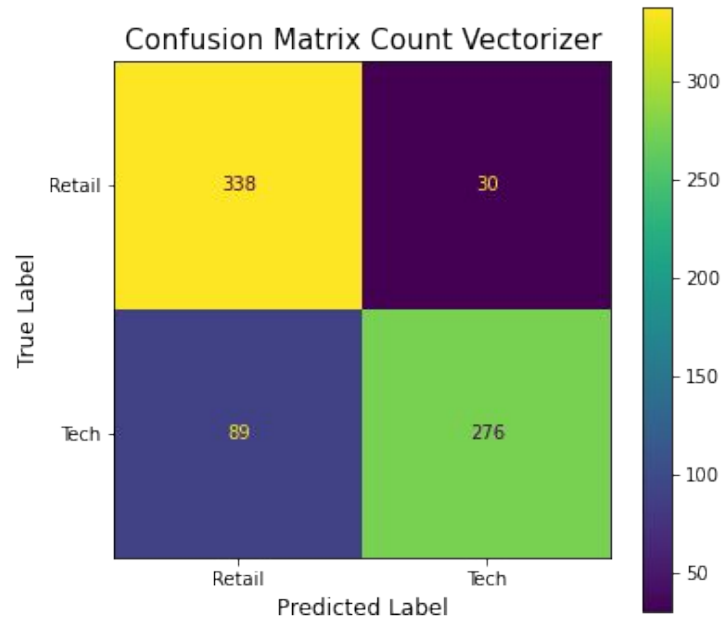
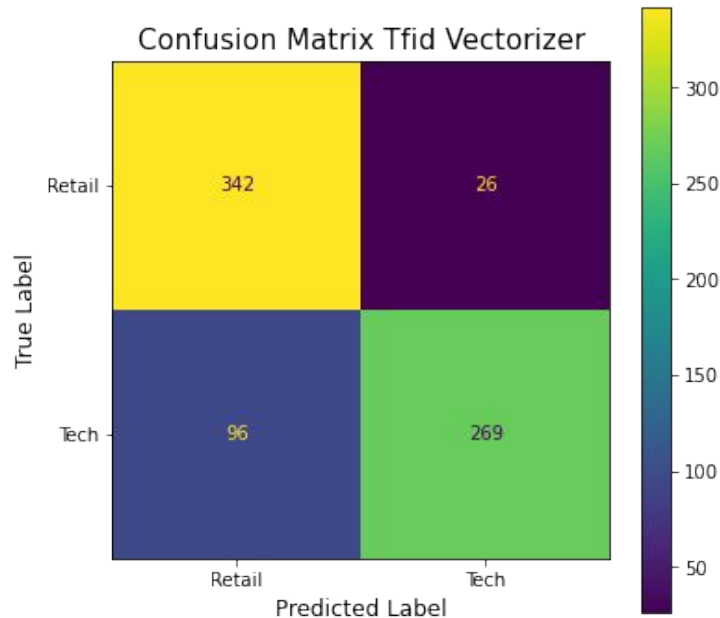
Cross Val : 0.5660
Accuracy : 57.03%
Specificity : 73.91%
Sensitivity : 40.00%
Precision : 60.33%



Random Forest

Cross Val : .8647
Accuracy : 83.77%
Specificity : 91.85%
Sensitivity : 75.62%
Precision : 90.20%

Cross Val : 0.8616
Accuracy : 83.36%
Specificity : 92.93%
Sensitivity : 73.70%
Precision : 91.19%



Model Stack!

- Combining all 6 models and their respective best parameters into level 1
- Use Linear Regression for level 2 since that showed promise as the most accurate/highest cross val score

```
level_1_models = [  
    ('logreg1_pipe', Pipeline([  
        ('cvec', CountVectorizer(stop_words='english', ngram_range=(1,2), max_df=.3, min_df=4, binary=True)),  
        ('logreg', LogisticRegression())  
    ])),  
    ('logreg2_pipe', Pipeline([  
        ('tfidf', TfidfVectorizer(stop_words='english', ngram_range=(1,1), max_df=.3, min_df=3, binary=True)),  
        ('logreg', LogisticRegression(solver='liblinear'))  
    ])),  
    ('knn1_pipe', Pipeline([  
        ('cvec', CountVectorizer(stop_words=stopwords, max_df=.4, min_df=7, binary=True)),  
        ('ss', StandardScaler(with_mean=False)),  
        ('knn', KNeighborsClassifier(n_neighbors=7, weights='distance'))  
    ])),  
    ('knn2_pipe', Pipeline([  
        ('tfidf', TfidfVectorizer(stop_words='english', max_df=.3, min_df=7, binary=False)),  
        ('ss', StandardScaler(with_mean=False)),  
        ('knn', KNeighborsClassifier(weights='distance'))  
    ])),  
    ('rf1_pipe', Pipeline([  
        ('cvec', CountVectorizer(stop_words='english', max_df=.6, min_df=4, binary=False)),  
        ('rf', RandomForestClassifier(n_estimators=200, min_samples_split=10, min_samples_leaf=1))  
    ])),  
    ('rf2_pipe', Pipeline([  
        ('tfidf', TfidfVectorizer(stop_words='english', ngram_range=(1,2), max_df=.6, min_df=4, binary=False)),  
        ('rf', RandomForestClassifier(min_samples_split=30))  
    ]))  
)  
  
stack = StackingClassifier(estimators=level_1_models, final_estimator=LogisticRegression())
```

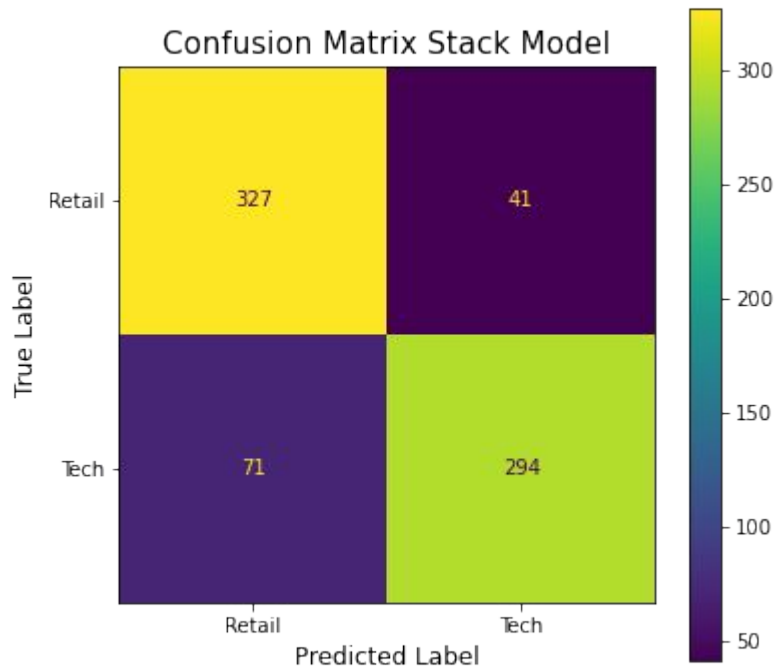
Cross Val : 0.8707

Accuracy : 84.72%

Specificity : 88.86%

Sensitivity : 80.55%

Precision : 87.76%



Statistic Summary

	Logistic Regression		KnearestNeighbors		Random Forest		Stack Model
	CountVec	TfidVec	CountVec	TfidVec	CountVec	TfidVec	
Cross-Val-Score	86.02%	88.21%	67.08%	56.60%	86.47%	86.16%	87.07%
Accuracy	82.26%	85.54%	68.49%	57.03%	83.77%	83.36%	84.72%
Specificity	90.76%	92.39%	95.38%	73.91%	91.85%	92.93%	88.86%
Sensitivity	73.70%	78.63%	41.37%	40.00%	75.62%	73.70%	80.55%
Precision	88.78%	91.11%	89.88%	60.33%	90.20%	91.19%	87.76%
Score STDEV	6.73%	5.46%	21.45%	12.09%	6.40%	7.61%	3.30%
Score MEAN	84.30%	87.18%	72.44%	57.57%	85.58%	85.47%	85.79%

Conclusion

- Highest Score Model: TfidfVectorizer Log Reg
 - Best Score Distribution: Stack Model
 - Future: Investigate further into Type I errors and find out why they were so prevalent in all models, then reevaluate all models
-