Springboard Data Science Career Track Capstone Project 1:

Recognizing Relevant Messages from a Fantasy Football Twitter Feed

By Logan Larson

July 2019

Contents

1 Introduction

Information is wealth in fantasy football. Without a comprehensive understanding of the current state of the NFL, participants could be at a significant disadvantage relative to the rest of the participants in his or her fantasy league. However, sometimes it's not enough to simply know what's going on; there are times when you need to know what's going on *before* the rest of your league finds out. For instance, when the Cleveland Browns traded starting running back Carlos Hyde to the Jacksonville Jaguars midway through the 2018 season, the then-backup Nick Chubb (who was available as a free agent in the vast majority of leagues) became arguably the most crucial free-agent pickup of the year. So crucial, in fact, that 61.4 percent of teams in ESPN leagues who rostered Chubb went on to make the playoffs.[1] In this scenario, having instant knowledge that Hyde was traded could have helped many people win a championship.

To address issues of this sort, I created a classification model that could instantly sort incoming news as either newsworthy or irrelevant in terms of fantasy football. Compared to something like Matthew Berry's Fantasy Life app (which pushes top-level notifications to mobile phones with a multi-minute delay), the end goal of my model would aim to provide notifications -- for all relevant news, not only the most important -- in real-time.

2 Approach

2.1 Data Acquisition and Cleaning

Conveniently, there exists a preeminent NFL reporter who has a reputation for regularly breaking the most important NFL-related news before anyone else: the award-winning Adam Schefter. A testament to just how integral he is, a large collection of Tweets from Schefter's Twitter account could serve as a reasonable approximation of all fantasy-football-relevant news from recent years.

Using the Twitter API, I collected over 33,000 of his Tweets spanning from January 2010 to April 2019. The only reason I didn't collect more was due to limitations imposed by Twitter.

I extracted the actual Tweets from the retrieved data and manually labeled them as follows:

- **T :** tweet contains transactional news, including but not limited to trades and free-agent signings

[1] Cockroft, T. H. (2018, December 28). James Conner, Todd Gurley II among most common players on fantasy playoff teams. Retrieved from https://www.espn.com/fantasy/football/story/_/id/25450523/fantasy-football-james-conner-todd-gurley-ii-most-common-players-fantasy-playoff-teams

- **I :** tweet contains updates on workload impediments, including but not limited to injury updates and role changes
- **X :** Anything else

I then mapped T and I Tweets to the positive binary class (1) and X Tweets to the negative class (0). The positive class effectively represents all relevant Tweets, i.e., newsworthy, while the negative class represents all irrelevant Tweets.

Next, I pre-processed each tweet as follows:

1. Converted all letters to lower case
2. Removed English stopwords using NLTK[2]
3. Removed frequently-appearing stopwords
   → Including 'NFL', 'today' and 'tomorrow'
4. Removed the 50,000 most rare words that appeared in two or fewer Tweets
   → Many of these were player names
5. Word correction with TextBlob[3]
6. Lemmatization using TextBlob

What I didn't do:

1. Remove numbers
   → Schefter uses numbers for injury and contract purposes
2. Remove punctuation and special characters (such as @ symbols)
   → Any tweet that contains these characters likely isn't newsworthy
3. Remove Retweets
   → Like special characters, Retweets are almost never newsworthy
4. Tokenize
   → Included in vectorization step

In the end, I prepared a dataset, a portion of which is shown in the figure below:

Class	Tweet
0	rt rank best kicker history top maybe top 3. f...
0	april least one trade made (but one waiting
1	giant re-signed restricted free agent guard kevin
0	@profootballtalk aaron retiring least trying g...
0	jimmy clausen scheduled fly washington spend s...

Figure 1

[2] Documentation: https://www.nltk.org/api/nltk.corpus.html
[3] Documentation: https://textblob.readthedocs.io/en/dev/

2.2 Storytelling and Inferential Statistics

Most of us are left to experience the world of professional sports from the outside, looking in, with no alternative but to compete in imaginary realms absent of any athletic barriers to entry. The baseball community realized this in the 1980s and ultimately devised a game within a game that could allow the average fan to get a taste of professional baseball. How it worked was generally straightforward: Participants took turns claiming individual athletes within the MLB before the season started, and as the season went on, whoever put together the most statistically productive team usually won.

Known today as "fantasy baseball," this imaginary game only gained popularity with time and nowadays there exists a version of it for practically every major American sport. Based on 2017 year-end data from the Fantasy Sports and Gaming Association, roughly 60 million (roughly 21 percent) people in the American/Canadian population (age 12-plus) were estimated to have participated in some capacity during 2017.[4] Figure 2 shows the growing trend of people participating in fantasy sports from the late 80's to right after 2015.

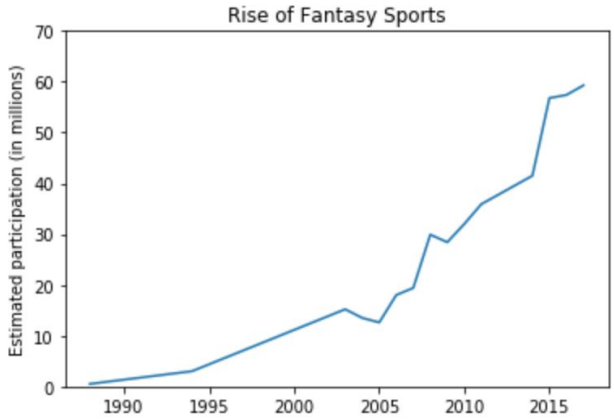


Figure 2

Regardless of sport, the statistical production of any athlete always depends greatest on one factor: availability. The logic is intuitive in that an athlete must be available to compete in order to produce.

Therefore, paying attention to injuries is important for anyone participating in a fantasy sport. And there may be no sport with less certainty around its players' game-by-game availability than football in the National Football League, whose players, according to NFL Injury Analytics, have historically been about four times as likely to get injured in a practice or game than players in the NBA, MLB or NHL.

[4] Fantasy Sports and Gaming Association. (n.d.). Industry Demographics. Retrieved from https://thefsga.org/industry-demographics/

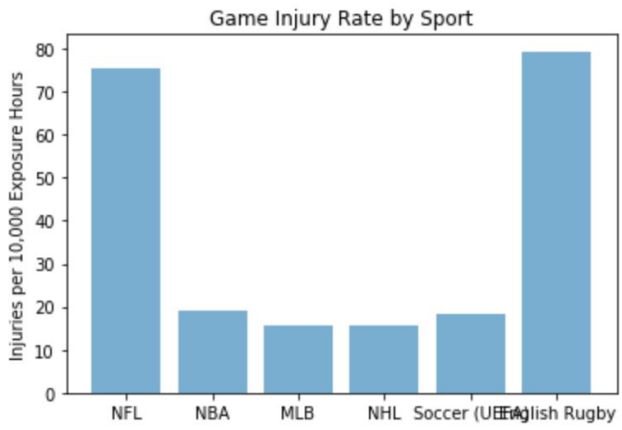


Figure 3. Source: Binney, Z. (2017, June 7). Just How Dangerous IS the NFL vs. Other Sports? Retrieved from https://nflinjuryanalytics.com/2017/06/06/just-how-dangerous-is-the-nfl-vs-other-sports/

Recognizing the individual players that aren't healthy enough to play can pose a significant advantage for the average fantasy football participants. Knowing which players are slated to benefit from that teammate's absence is even more advantageous. Yet in either case, it's always better to learn the information sooner than later — especially in high-stakes leagues with prize money at stake. The problem is that NFL teams have little incentive to provide the general public with perfectly accurate information on a real-time basis since that information could also provide a competitive advantage to its opponent. This often forces fantasy football participants to rely upon the media for injury-related information.

That's where Schefter comes into play. As mentioned above, Schefter is an incredibly reliable reporter who frequently uses Twitter to break news every day. Figure 4 shows the breakdown of his activity level in terms of Tweets per day.

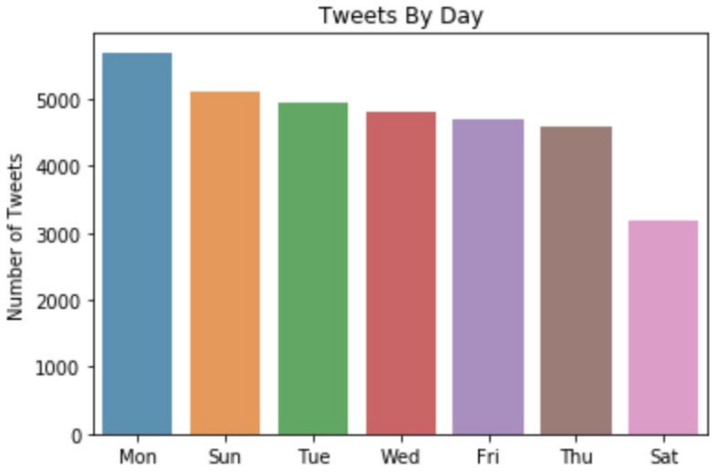


Figure 4

While it's a slight surprise that he wasn't most active on Sundays (when the vast majority of NFL games are played), it's possible he's most active Monday because of extensive follow-ups on injuries that happened in the games the day before. Figure 5 shows the breakdown of Schefter's activity level by Tweets per month.

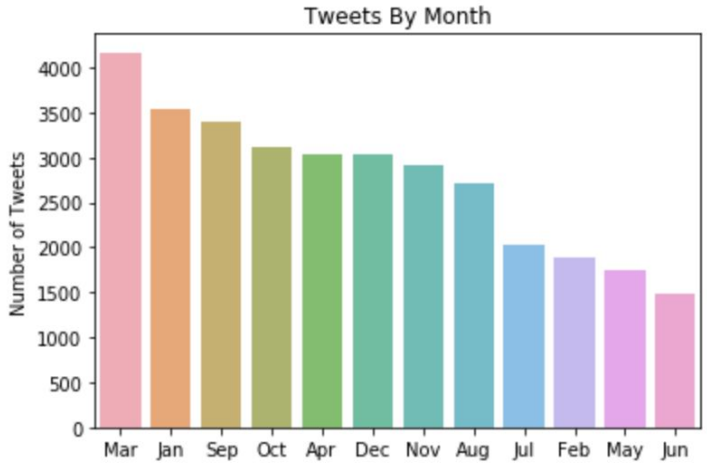


Figure 5

It's also surprising he was most active in March because the NFL's regular season runs mid-September through December and early January. With that said, it's not surprising that March is an active month for him overall since the start of the new league year, and thus the opening days of free agency, occur during this month.

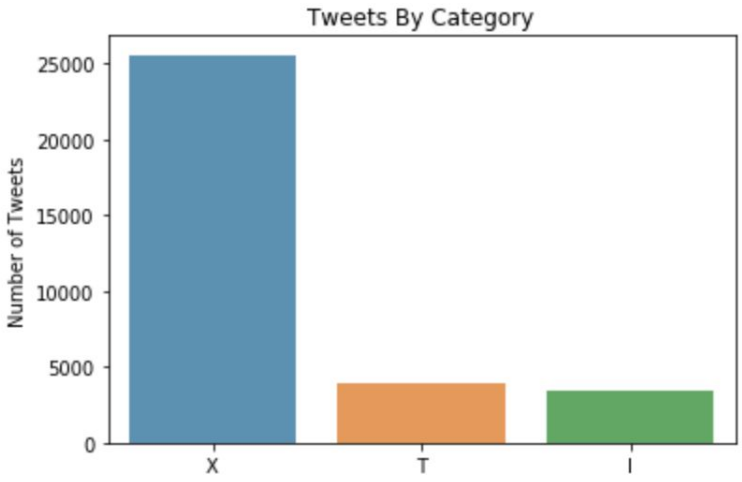


Figure 6

As previously mentioned, an initial goal of my project was to take Schefter's Tweets and categorize them into three categories: X, T and I. The 'T' label designates a tweet is transactionary or, in other words, any tweet that contains information about any player transactions that impact their contract status or playing eligibility. The 'I' label designates any

workload impediments such as injury updates or role changes. The 'X' label designates everything else.

We care about 'T' Tweets because we need to know which players are on which teams -- and within those teams, which of the players are eligible to compete in the first place. Then we get more specific with the 'I' Tweets to gain further information on which eligible players are healthy to play and, subsequently, how much playing time they can be expected to receive. The 'X' Tweets are then entirely irrelevant for our purposes since they don't offer us any further, actionable information.

For the purposes of fantasy football, the 'I' Tweets are generally most important to us because, while the most relevant players don't often change teams, injuries can strike anyone.

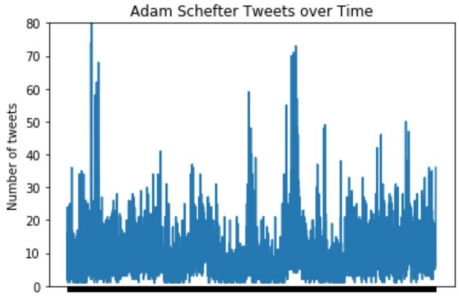


Figure 7

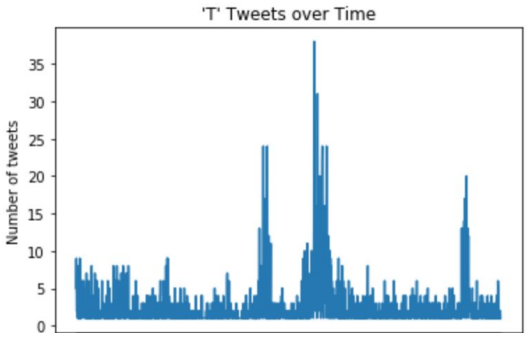
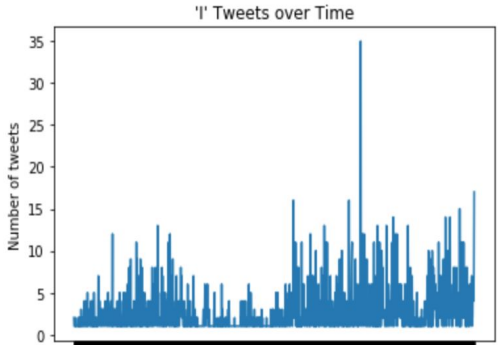


Figure 8

Figure 9



In Figure 9, it seems Schefter has become more involved with injury updates over time. Meanwhile, Figure 8 suggests some offseasons may have had more active free-agent markets than other years. But most importantly, when you compare the trend of 'I' Tweets to all of his Tweets, his total volume seems cyclically stable - but if it holds that he truly has been more active in the injury department in recent years - then it follows that, on the whole, his average tweet is more likely to be relevant to us than it was in year's past (based on how I defined ‘relevant’).

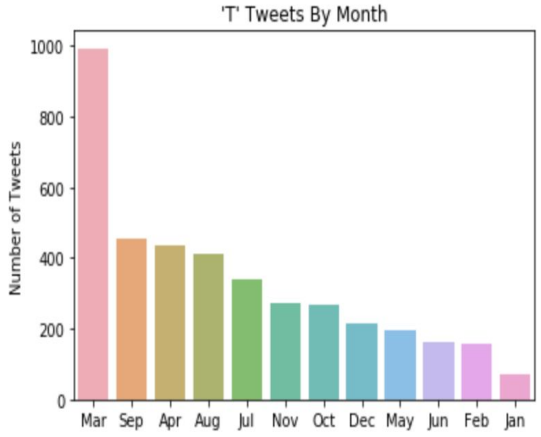


Figure 10

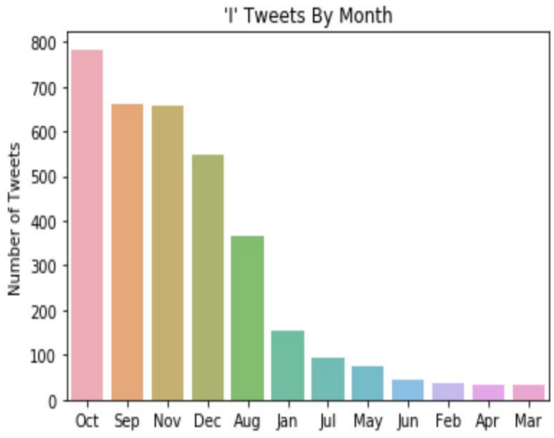


Figure 11

Here is the frequency of Tweets per month, grouped by 'I' Tweets and 'T' Tweets. Again, there's nothing surprising here since it's intuitive that the most injury notes are posted when football is being played (an NFL season runs September through January, preceded by two months of training camp and the preseason in July and August). Meanwhile, free agency opens in March and the NFL Draft is in April, and most roster cuts happen before the regular season -- largely in August and early September -- so we would expect those four months to have the highest frequencies of transactionary Tweets.

2.3 Baseline Modeling

In this section, I'm going to discuss the simplest of the classification models I created to determine which of his Tweets are newsworthy or not. As seen in this notebook, these models were meant to serve as a baseline for future models.

Tweets, after being pre-processed, were vectorized using scikit-learn's count vectorizer. Compared to the more advanced TF-IDF vectorizer, the count vectorizer simply converts a collection of text documents to a matrix of token counts.

Once vectorized, I fed the data into three different classifiers:

1. A Bernoulli naive Bayes classifier (BNB)
2. A multinomial naive Bayes classifier (MNB)
3. A logistic regression classifier (LR)

	BNB	BNB (tuned)	MNB	MNB (tuned)	LR	LR (tuned)	Ridge	Lasso
Training Accuracy	0.89	0.891	0.886	0.89	0.939	0.938	0.938	0.932
Test Accuracy	0.879	0.882	0.874	0.881	0.911	0.911	0.911	0.912
Training Precision (0)	0.963	0.947	0.965	0.947	0.949	0.949	0.949	0.944
Test Precision (0)	0.952	0.941	0.956	0.941	0.929	0.929	0.929	0.929
Training Precision (1)	0.705	0.726	0.692	0.726	0.896	0.896	0.896	0.883
Test Precision (1)	0.69	0.709	0.674	0.709	0.839	0.839	0.839	0.841
Training Recall (0)	0.892	0.909	0.884	0.909	0.972	0.972	0.972	0.969
Test Recall (0)	0.889	0.903	0.878	0.903	0.958	0.958	0.958	0.959
Training Recall (1)	0.882	0.825	0.89	0.825	0.822	0.822	0.822	0.804
Test Recall (1)	0.848	0.807	0.861	0.807	0.749	0.749	0.749	0.751
Training F1 score (0)	0.926	0.928	0.923	0.928	0.961	0.961	0.961	0.956
Test F1 score (0)	0.919	0.922	0.915	0.91	0.922	0.943	0.943	0.943
Training F1 score (1)	0.783	0.772	0.778	0.772	0.857	0.857	0.857	0.842
Test F1 score (1)	0.761	0.755	0.756	0.755	0.791	0.791	0.791	0.793
Count - Training (0)	19185							
Count - Training (1)	5595							
Count - Test (0)	6391							
Count - Test (1)	1870							

Figure 12

The results for the baseline models can be seen in Figure 12. Each attained an accuracy of at least 87 percent on the test set, but the precision-recall tradeoff was different for the naive Bayes classifiers compared to the LR classifier. For example, both the BNB and MNB classifiers attained a recall of at least 84 percent on test data but a precision no higher than 69 percent. Meanwhile, the LR classifier attained a higher precision (0.839) and a lower recall (0.749).

If we define overfitting as a discrepancy of 10 points or more between test and training scores (as percentages) for any of four metrics (accuracy, precision, recall and F1 score), I have little evidence of overfitting in any of the three models. It follows that I therefore have little incentive to perform regularization on any of them. However, there are still improvements that can be made, particularly

in terms of both recall (7.3-point gap) and precision (5.7) for my logistic regression model, so I proceeded with regularization out of curiosity.

The regularization largely did what it's supposed to do for the naive Bayes models. That is to say, it reduced overfitting in nearly every metric even though little overfitting was present to begin with. In turn, the overall accuracy of each model slightly improved. However, the optimal regularization parameter (C) for my LR model was the same as the default value, so I observed zero difference compared to the non-regularized model. I nonetheless proceeded to model an L1 penalty (Lasso classifier) and L2 penalty (Ridge classifier), but neither produced any change in any metric calculated from the test set.

2.4 Extended Modeling

I next replicated the baseline models using a TF-IDF vectorizer, as seen in this notebook. More advanced than the count vectorizer, the TF-IDF vectorizer is equivalent to using the count vectorizer followed by the TF-IDF transformer. In any case, this vectorizer not only accounts for the frequency of a term, but also the frequency that term appears across documents. In the end, each feature is weighted by the frequency it appears multiplied by the inverse of the frequency it appears across all documents.

	BNB	BNB (tuned)	MNB	MNB (tuned)	LR	LR (tuned)
Training Accuracy	0.893	0.892	0.903	0.905	0.923	0.929
Test Accuracy	0.882	0.885	0.891	0.89	0.909	0.911
Training Precision (0)	0.961	0.936	0.921	0.93	0.931	0.939
Test Precision (0)	0.952	0.93	0.912	0.916	0.919	0.924
Training Precision (1)	0.716	0.749	0.828	0.811	0.886	0.888
Test Precision (1)	0.697	0.738	0.805	0.788	0.864	0.855
Training Recall (0)	0.899	0.923	0.957	0.949	0.972	0.971
Test Recall (0)	0.893	0.921	0.951	0.945	0.967	0.964
Training Recall (1)	0.875	0.783	0.718	0.757	0.755	0.785
Test Recall (1)	0.845	0.763	0.684	0.702	0.709	0.728
Training F1 score (0)	0.929	0.93	0.938	0.939	0.951	0.955
Test F1 score (0)	0.921	0.925	0.931	0.93	0.943	0.943
Training F1 score (1)	0.787	0.766	0.769	0.783	0.815	0.833
Test F1 score (1)	0.764	0.75	0.74	0.743	0.779	0.786
Count - Training (0)	19185					
Count - Training (1)	5595					
Count - Test (0)	6391					
Count - Test (1)	1870					

Figure 13

I ultimately attained different results using this approach, as seen in Figure 13. Interestingly, both naive Bayes models increased in overall accuracy relative to the baseline. Meanwhile, the accuracy of the logistic regression model slightly decreased, but it was less overfit in terms of precision, recall and accuracy.

While we again have little evidence of overfitting, I again proceeded with regularization out of curiosity. Doing so had similar results to that from the count vectorizer, as they struck a better balance between recall and precision and slightly increased in overall accuracy.

However, thus far each vectorizer has implicitly used n-grams of length 1. Given the common phrases Schefter uses in newsworthy Tweets (for instance, “so and so **is questionable to play**”, “so and so **has been traded to**…”, etc.), I’m optimistic that using longer n-grams will improve my results.

In practice, the usage of n-grams up to length 5 accentuated the tradeoff between precision and recall. Both the naive Bayes models were an extreme case of this, as the precision skyrocketed to the upper 90s on test data while recall and overall accuracy dropped significantly. The results for each of the BNB, MNB and LR models -- using both a count vectorizer and TF-IDF vectorizer -- can be seen below in Figure 14.

	BNB (TF-IDF)	MNB (TF-IDF)	LR (TF-IDF)	BNB (Count)	MNB (Count)	LR (Count)
Training Accuracy	0.814	0.88	0.923	0.814	0.994	0.999
Test Accuracy	0.784	0.804	0.874	0.784	0.86	0.912
Training Precision (0)	0.806	0.866	0.913	0.806	0.999	0.999
Test Precision (0)	0.782	0.798	0.868	0.782	0.965	0.927
Training Precision (1)	1	0.988	0.976	1	0.977	1
Test Precision (1)	0.978	0.984	0.918	0.978	0.636	0.853
Training Recall (0)	1	1	0.995	1	0.993	1
Test Recall (0)	1	0.999	0.987	1	0.85	0.963
Training Recall (1)	0.177	0.467	0.676	0.177	0.996	0.996
Test Recall (1)	0.047	0.135	0.487	0.047	0.894	0.739
Training F1 score (0)	0.893	0.928	0.952	0.893	0.996	0.999
Test F1 score (0)	0.877	0.887	0.924	0.877	0.877	0.904
Training F1 score (1)	0.301	0.637	0.799	0.301	0.986	0.998
Test F1 score (1)	0.09	0.238	0.636	0.09	0.743	0.792
Count - Training (0)	19185					
Count - Training (1)	5595					
Count - Test (0)	6391					
Count - Test (1)	1870					

Figure 14

3 Findings and Analysis

Model evaluation depends on the business problem and whether - in this scenario - we should care most about overall accuracy, precision, recall, or a combination of precision and recall, such as the f1 score.

To come to a decision, let's start from the beginning. We defined a positive case (labeled 1) as a newsworthy tweet while a negative case (labeled 0) as an irrelevant tweet. It follows that a false positive can be considered an irrelevant tweet that was incorrectly classified as newsworthy. In contrast, a false negative could be considered a newsworthy tweet that was incorrectly classified as irrelevant.

Since I'd rather have an irrelevant tweet classified as newsworthy (false positive) than have a newsworthy tweet classified as irrelevant (false negative) - and risk users missing potentially important information - I primarily want to minimize the number of false negatives. Thus, I should pick the model that performs best in terms of recall.

The primary alternative would be to prioritize precision, and that is less preferable because it wouldn't be the worst thing in the world for an irrelevant tweet to be considered newsworthy since I suspect that potential users of my model could easily recognize false positives and simply ignore them. However, I would lose trust quickly if I failed to recognize the news that matters. Put this way, it doesn't make sense to prioritize either precision or any kind of F score if it means a trade-off in terms of recall.

So, my goal is to maximize recall at the expense of a lesser precision. Thus far, my best performing model by this metric turns out to be the non-regularized, baseline multinomial naive Bayes model. However, I didn’t use n-grams longer than 1 in this model and was initially curious to do so. When applied, I interestingly attained the near-opposite of what I did in the extended modeling notebook: Using n-grams of length 2 through 5, I got a recall of nearly 96 percent at the expense of a low precision (0.310). The exact results can be seen in Figure 15.

	MNB (1,3)	MNB (1,4)	MNB (2,5)
Training Accuracy	0.978	0.99	0.996
Test Accuracy	0.879	0.87	0.508
Training Precision (0)	0.995	0.999	0.999
Test Precision (0)	0.951	0.959	0.969
Training Precision (1)	0.926	0.961	0.986
Test Precision (1)	0.691	0.661	0.31
Training Recall (0)	0.977	0.988	0.996
Test Recall (0)	0.889	0.869	0.376
Training Recall (1)	0.983	0.995	0.997
Test Recall (1)	0.845	0.874	0.959
Training F1 score (0)	0.986	0.993	0.997
Test F1 score (0)	0.919	0.912	0.542
Training F1 score (1)	0.954	0.978	0.991
Test F1 score (1)	0.76	0.753	0.469
Count - Training (0)	19185		
Count - Training (1)	5595		
Count - Test (0)	6391		
Count - Test (1)	1870		

Figure 15

4 Conclusions and Future Work

This latest multinomial Naive Bayes model that uses a count vectorizer and n-grams of range 2 through 5 achieves the highest recall rate yet. The most considerable alternative is a similar model that expands to use n-grams of size 1, which appears to help strike a much better balance between precision and recall. However, as stated earlier, I believe potential users of my model won't have nearly as big of a problem with frequent false positives than they would with only a few false negatives. Therefore, for my business problem, I'm inclined to maximize recall rate and I thus choose the multinomial naive Bayes model using a count vectorizer and n-grams of range 2 through 5. While a low precision means I should expect frequent false positives, this model will make up for it by capturing the vast majority of newsworthy Tweets. After all, I'm confident that anyone who cares enough about fantasy football to use my model has the capability of quickly recognizing any false positives that may slip through the cracks.

```
Row 123 has been classified as  1 and should be  0
Row 126 has been classified as  1 and should be  0
Row 130 has been classified as  1 and should be  0
Row 131 has been classified as  1 and should be  0
Row 132 has been classified as  1 and should be  0
Row 134 has been classified as  0 and should be  1
Row 136 has been classified as  1 and should be  0
Row 137 has been classified as  1 and should be  0
```

Figure 16

```
xtest.iloc[132]
```

```
"cardinal hired terry mcdonough eastern brother ryan new gm, making arizo
na's first family."
```

```
xtest.iloc[134]
```

```
'sammy watkins officially active.'
```

Figure 17

For example, let's take a look at an example of a false positive and a false negative. Figure 17 shows that Row 132 is an example of the former and Row 134 is an example of the latter. When we take a look to see which Tweets these rows correspond to, it's easy to see that the false negative is not something we'd want to miss because Sammy Watkins is a high-profile receiver and it could be highly important to know that he was available to suit up for a game. On the other hand, the false positive clearly is not practically relevant and its reasonable to assume the average fantasy football participant would have the wherewithal to ignore it. In this specific example, it's reasonable to assume that team staffing has practically zero relevance to any actions a fantasy football participant could conceivably take.

Future work could include experimenting with ensemble methods. It could also include working on an additional feature that recognizes when newsworthy Tweets relate to high-profile players. Once that is worked out, the focus would be to create mobile applications that could push notifications to users in real time. Future work could also include making a separate classification model for the

Tweets of other reporters such as Mike Garafolo, in case other reputable reporters happen to beat Schefter to the punch on certain occasions.

5 Recommendations for the Client

I recommend using this model as a starting point for a mobile application. Given the ubiquity of cell phones in America in 2019, email and other means of communication aren't practically valuable for the purposes of fantasy football. After all, sometimes decisions need to be made within seconds or minutes of the breaking of significant news.

However, it could be important for some users to only be notified about certain players. In this case, it would be beneficial to cross-reference positive-predicted observations against some sort of database before pushing the notification to mobile phones