r/eagles vs r/cowboys

PREDICTING REDDIT COMMUNITY FROM POST CONTENT USING NATURAL LANGUAGE PROCESSING

BY JON GODIN

The Data Science Problem

USE NLP TO PREDICT SUBREDDIT OF POSTS

- For this project, we were tasked with finding two different subreddits, culling posts from each, and using natural language processing to see how well we can classify posts as belonging to one or the other subreddit
- In honor of this week's NFL Draft, I selected the subreddits r/cowboys and r/eagles, two rival franchises with dedicated fan bases
- Problem I'm trying to solve:

How accurately can I predict posts from the r/cowboys subreddit without using obvious keywords/tokens?

Secondarily, can I uncover which words are more indicative of one or the other fan bases heading into the draft?

Data Collection & Setup

For this project I collected 2000 reddit posts using the Pushshift API (https://github.com/pushshift/api)

1000 posts were culled from the r/cowboys subreddit, and 1000 posts were culled from the r/eagles subreddit

The <title> and <selftext> fields were pulled out from the data and concatenated to form a single string. Some posts had only a title, and occasionally some selftext was deleted (presumably by the subreddit's moderator. Any holdover [removed] or [deleted] tags were replaced with empty strings. Roughly 75% of posts consisted of only a title with no selftext.

Each file was coded as being either r/cowboys posts (=1) or r/eagles posts (=0) based on their sources.

The two files were then merged, shuffled, and randomly split into train and test data files, using a 30% test size, resulting in a train file with 1400 posts, and a test file with 600 posts. Stratification was used to ensure each file consisted of 50% r/cowboys and 50% r/eagles posts.

Data Cleaning/Pre-Processing Steps

Used BeautifulSoup library

Tried this two ways using regular expressions:

- First removing only special characters (excluding the # symbol, which is commonly used to describe roster positions, e.g., #2 WR)
- Removing all numbers and special characters

Optional step, tried fitting models with un-lemmatized tokens, and then tried lemmatizing (which seemed to work better for my data)

See next slide for more details

Stopword list creation

Stopwords are words that are removed from the text string/tokens prior to analysis

Remove common stopwords

- Utilized the nltk.corpus library to remove common English-language stopwords
- These words have been determined to not add much value to NLP analysis as they are extremely common and usually devoid of sentiment
- Additionally, I removed non-meaningful strings like app, http, etc, imgur, www, com, png, reddit and the like

Remove obvious classifiers

- I wanted my model to have to do some work, so I immediately added obvious tells to the stopword list, such as the following:

  dallas, cowboys, cowboy, philadelphia, eagles, philly, eagle, phi, dal, dallascowboys, boys, birds, texas

Remove current and former player, coach, or GM names

- After examining the resulting vocab after the initial stopwords were removed, I pulled out any recognizable names (from either team), again because each fanbase is likely to talk about their own players or front office more frequently

  dak, prescott, ezekiel, elliott, carson, wentz, zach, ertz, jerry, jones, etc.
- A total of 159 “name tokens” were added to the stopword list, including multiple name-forms (e.g., tim, timmy)

Modeling

Vectorization

Two types of vectorization were utilized to create predictor variable word sets:

- **CountVectorization**, where character tokens are simply summed/counted within each post
- **TfidfVectorization**, where character tokens are converted to relative ratios of how many times a word appears in a given post relative to the number of times that word occurs across all posts

Classification Modeling

- Using r/cowboys posts as the target class, each of the vectorization methods were variously and separately tried as predictors for multiple machine learning classifiers, including:
  - Logistic Regression
  - Multinomial Naïve Bayes Classifier
  - Bagging Classifier
  - Random Forests Classifier
  - Extra Trees Classifier
  - Gradient Boosting (including Stochastic Gradient Boosting) Classifier
  - Support Vector Machine Classifier
- Grid search and/or cross-validation was typically utilized for each model to help establish best-performing hyperparameters or ascertain whether either type of vectorization produced better results
- Regularization methods for each ML classifier were tested and/or utilized to reduce overfitting as much as possible
- Models were evaluated based on accuracy, sensitivity, precision, and ROC AUC scores, relative to the default baseline of 50% correct classification, as well as the model performance without removing obvious stopwords

Results

- Naïve baseline: 50% Accuracy
- Models ranked on ROC AUC Score
- Lemmatized tokens outperformed un-lemmatized tokens (though not by much)
- I selected the CVEC Logistic Regression model for further analysis due to its Sensitivity and ROC AUC scores

Stopwords Used	Vectorizer / Classifier	Cross-Val Accuracy	Train Accuracy	Test Accuracy	Test Sensitivity	Test Specificity	Test Precision	ROC AUC Score	
Standard English only	CVEC, Logistic Regression, Lasso Regularization	0.8157	0.9479	0.8033	0.7800	0.8267	0.8182	0.8958	
Standard plus custom	CVEC, Stochastic Gradient Boosting Classifier #	0.6829	0.9707	0.6250	0.5833	0.6670	0.6364	0.7263	
Standard plus custom	CVEC, Logistic Regression, Lasso Regularization	0.6650	0.8957	0.6533	0.7200	0.5867	0.6353	0.7223	
Standard plus custom	TfidfVec, Logistic Regresion, Lasso Regularization	0.6621	0.9207	0.6550	0.7100	0.6000	0.6396	0.7012	*
Standard plus custom	TfidfVec, Support Vector Machine Classifier	0.6364	0.9421	0.6317	0.5733	0.6900	0.6491	0.6999	
Standard plus custom	CVEC, Multinomial Naïve Bayes	0.6800	0.7886	0.6033	0.5900	0.6167	0.6062	0.6979	*
Standard plus custom	CVEC, Support Vector Machine Classifier	0.6379	0.8407	0.6417	0.8000	0.4833	0.6076	0.6919	
Standard plus custom	CVEC, RandomForest Classifier ^	0.6750	0.9771	0.6200	0.5933	0.6467	0.6268	0.6882	
Standard plus custom	TfidfVec, Stochastic Gradient Boosting Classifier #	0.6800	0.9443	0.6050	0.5233	0.6867	0.6255	0.6751	
Standard plus custom	CVEC, Decision Tree Classifier with Bagging	0.6071	0.6414	0.5633	0.2500	0.8767	0.6696	0.5633	
Standard plus custom	CVEC, Decision Tree Classifier	0.5657	0.5964	0.5433	0.1567	0.9300	0.6912	0.5433	

* - ROC AUC Score flipped for easier comparison (r/eagles class more accurately predicted)

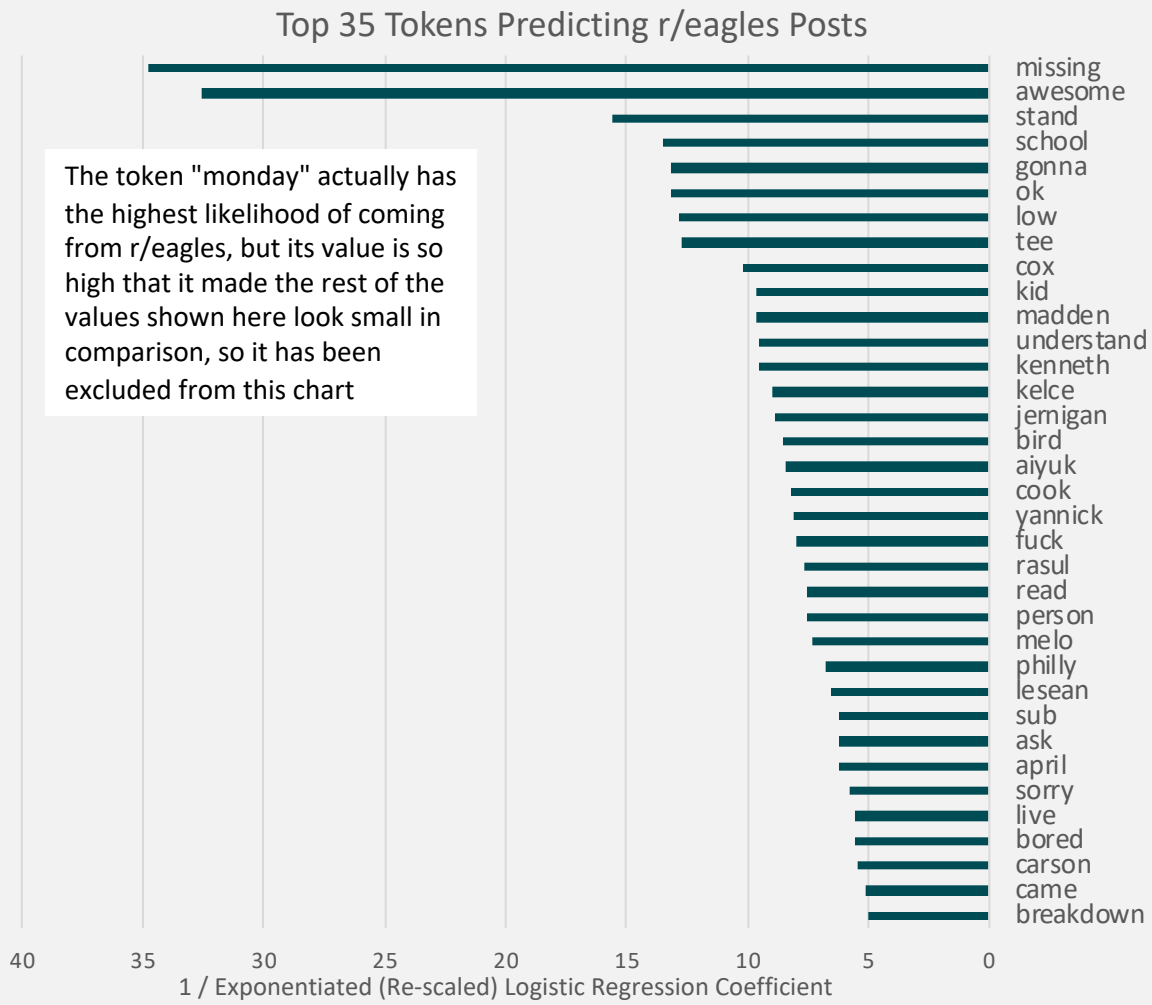
- cross-validation showed gbm outperformed adaboost classifier

^ - cross-validation showed rfc outperformed extratrees classifier

Which words/tokens were more likely from each subreddit?

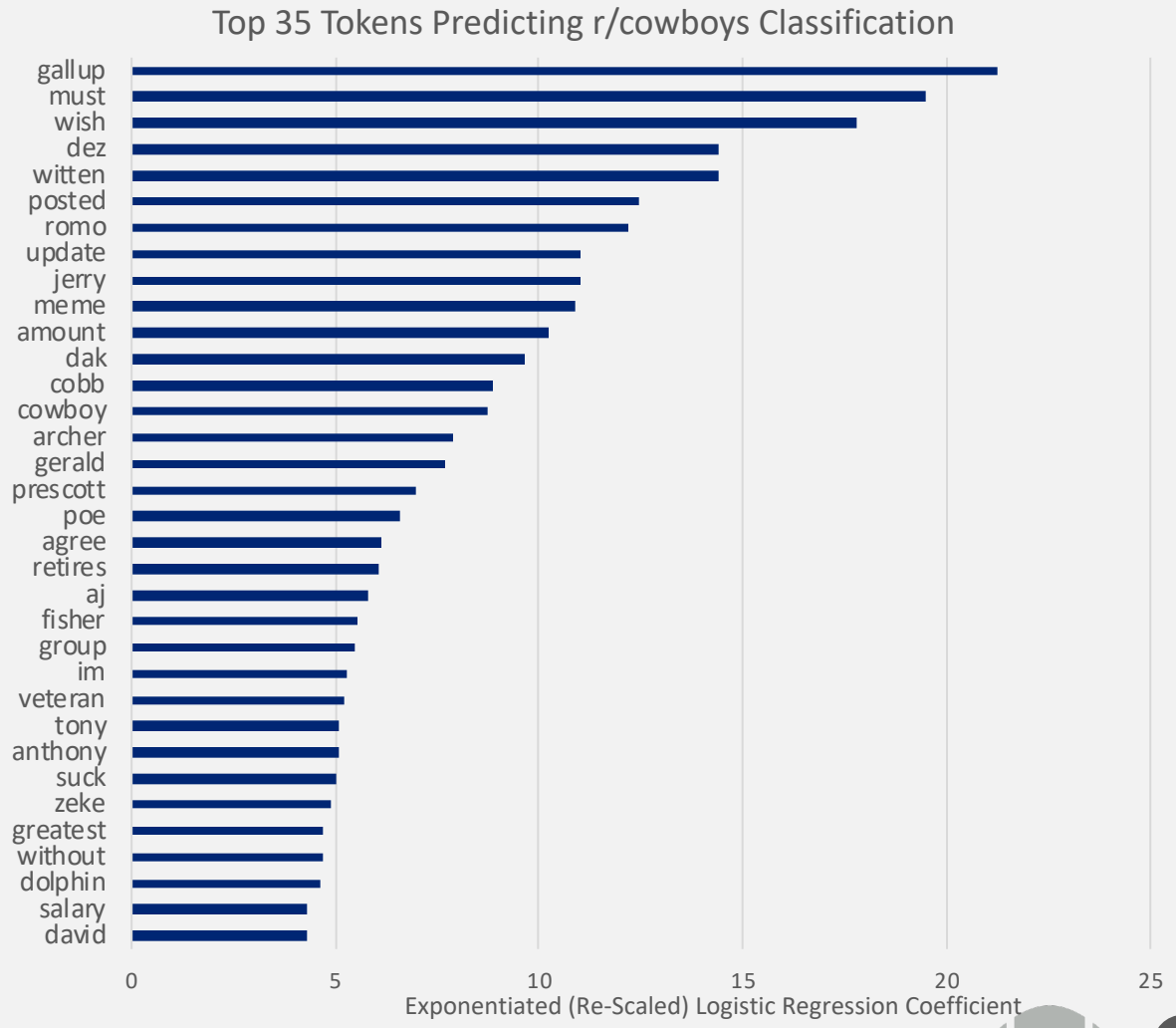
Only standard stopwords removed

Tokens That Predict r/eagles Posts



The token "monday" actually has the highest likelihood of coming from r/eagles, but its value is so high that it made the rest of the values shown here look small in comparison, so it has been excluded from this chart

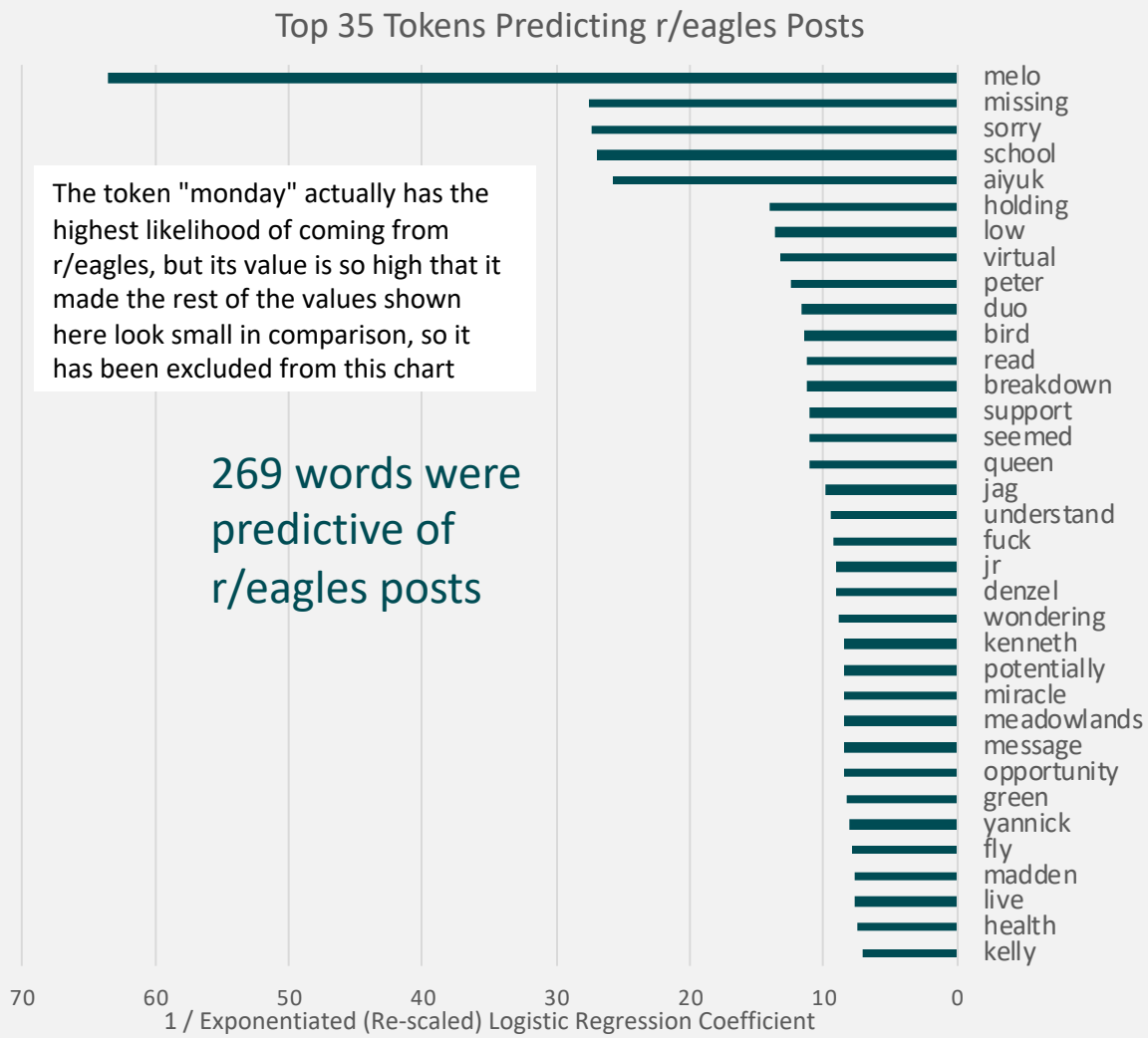
Tokens That Predict r/cowboys Posts



Which words/tokens were more likely from each subreddit?

Standard and custom stopwords removed

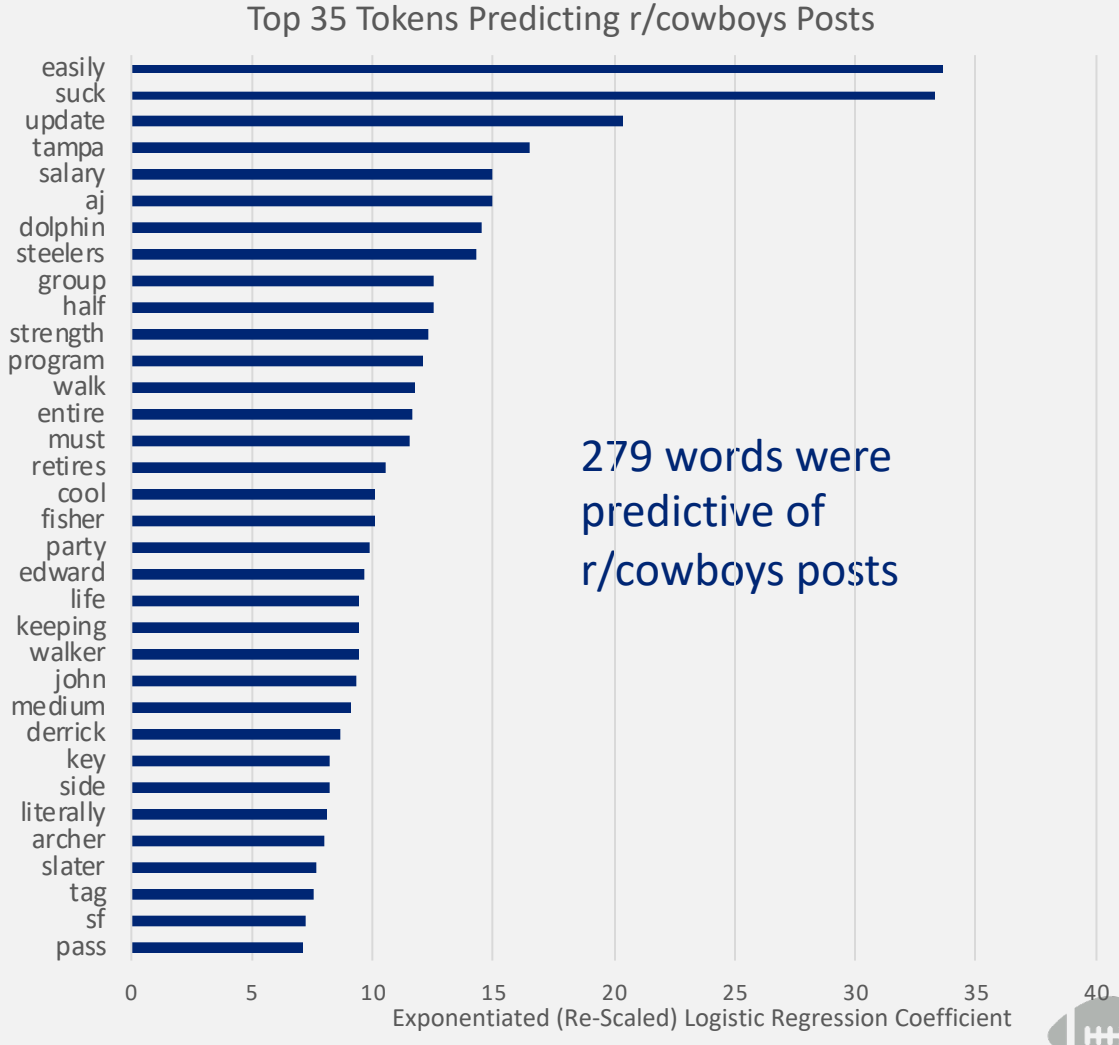
Tokens That Predict r/eagles Posts



The token "monday" actually has the highest likelihood of coming from r/eagles, but its value is so high that it made the rest of the values shown here look small in comparison, so it has been excluded from this chart

269 words were predictive of r/eagles posts

Tokens That Predict r/cowboys Posts



279 words were predictive of r/cowboys posts

Conclusions & Recommendations

- Even without removing obvious-classifying tokens, the best possible model only achieved 80.7% CV accuracy, so trying to predict which subreddit the posts came from proved difficult overall, but **most of the models I estimated were able to achieve a 15-20 percentage point lift in accuracy over baseline**
- Both subreddits consist of football fans, and while they follow one particular team in general, the terminology and/or general language they use may not differ enough for highly-accurate classification
- Based on ROC AUC scores, and my general preference for better sensitivity performance in this case (as a Cowboy fan myself, I want a model that does better at identifying actual Cowboys fans – I don't care how Eagles fans feel!), so the model with Count Vectorization and Logistic Regression with L1 (LASSO) regularization performed the best once obvious stopwords and player/ex-player/coach/GM names were removed
- All models seemed to suffer from fairly strong overfitting, even with regularization
- Differences in word usage did reveal some tendencies in topics of concern, preferences for positions or players the two fan bases wanted their teams to draft, or which media members they tended to follow for NFL news/insights
- **For future research**, larger and longer-term data pulls could be attempted to see whether increasing the overall sample size leads to less overfitting and better model performance
  - Since many posts consisted of only title text, there may also have been less signal in the data than might be optimally desired

Appendix: Custom stopwords

"dallas","cowboys","cowboy","philadelphia","eagles","philly","eagle”,
"phi","dal","dallascowboys","boys","birds","texas","app","http","https",
"etc","imgur","www","com","png","auto","width","reddit","x200b","2qgz89t",
"2z11chhwhq99x5sashj6fb5xi6wir5t","zeke","dak","prescott","romo","tony",
"carson","wentz","dez","bryant","demarco","murray","cole","beasley",
"aldon","smith","amari","cooper","alshon","jeffery","anthony","brown",
"chidobe","awuzie","avonte","maddox","blake","jarwin","brian","dawkins", "byron","jones","cam","fleming","haha","ha",
"ha-ha","clinton","clinton-dix", "dix","rush","darius","slay","david","irving","demarcus","lawrence",
"robinson","desean","jackson","donovan","mcnabb","dontari","poe",
"dorsett","ezekiel","elliott","jake","zach","ertz","leighton","van", "der","esch","fletcher","cox","nick","foles","kai","forbath","travis",
"frederick","jason","garrett","witten","kelce","peters","gerald", "mccoy","howie","roseman","jeff","heath","tim","jernigan","jerry",
"jim","schwartz","joe","looney","jordan","matthews","jp","ladouceur",
"lane","johnson","lesean","mccoy","maliek","collins","mike","mccarthy",
"michael","gallup","miles","sanders","nelson","agholor","nickell",
"robey","coleman","robert","quinn","randall","cobb","randy","gregory",
"sean","lee","roger","staubach","stephen","tank","troy","aikman", "tyron","smith","vick","whiteside","zack","martin","zuerlein","andre",
"dillard","boston","scott","brandon","graham","cameron","chido","cunningham",
"curtis","samuel","darian","thompson","jeremy","maclin","kerry","hyder",
"rasul","douglas","tavon","austin","timmy","malcolm","jenkins","jalen", "mills","greg”, “javon”, “hargrave”, “kellen”, “moore”, “dlaw”


I SET MY TIVO TO
RECORD THE BIGGEST LOSER
IT KEEPS RECORDING
THE PHILADELPHIA EAGLES

THANK YOU