# Final Project

## Problem statement and hypothesis

Reddit is a community message board of sorts where users communicate and organize over just about any topic. One such subreddit community is called Random Acts of Pizza. Reddit users post requests for free pizza in this subreddit explaining their situation and why they need pizza. Some user requests result in free pizaa, while others dont. The goal of this project is to build a model to predict what kind of requests will result in the requester receiving free pizza. The hypothesis is that users who write a detailed request, have been Reddit users for a long time, who make a request on a weekday, and who have given pizza away before are more likely to receive a free pizza.

## Data

This dataset includes 5671 requests collected from the Reddit community Random Acts of Pizza between December 8, 2010 and September 29, 2013 (retrieved on September 30, 2013). All requests ask for the same thing: a free pizza. The outcome of each request -- whether its author received a pizza or not -- is known. Meta-data includes information such as: time of the request, activity of the requester, community-age of the requester, etc.

Each JSON entry corresponds to one request (the first and only request by the requester on Random Acts of Pizza). We have removed fields from the test set which would not be available at the time of posting.

### Relevant Data Fields

"request\_id": Identifier of the post on Reddit, e.g. "t3\_w5491".

"request\_text": Full text of the request.

"request\_title": Title of the request.

"requester\_account\_age\_in\_days\_at\_request": Account age of requester in days at time of request.

"requester\_days\_since\_first\_post\_on\_raop\_at\_request": Number of days between requesters first post on RAOP and this request (zero if requester has never posted before on RAOP).

"requester\_number\_of\_comments\_at\_request": Total number of comments on Reddit by requester at time of request.

"requester\_number\_of\_posts\_at\_request": Total number of posts on Reddit by requester at time of request.

"requester\_received\_pizza": Boolean indicating the success of the request, i.e., whether the requester received pizza.

"requester\_upvotes\_minus\_downvotes\_at\_request": Difference of total upvotes and total downvotes of requester at time of request.

"requester\_user\_flair": Users on RAOP receive badges (Reddit calls them flairs) which is a small picture next to their username. In our data set the user flair is either None (neither given nor received pizza, N=4282), "shroom" (received pizza, but not given, N=1306), or "PIF" (pizza given after having received, N=83).

"unix\_timestamp\_of\_request\_utc": Unit timestamp of request in UTC.

## Data Conditioning

I read the Kaggle JSON data into a Pandas DataFrame and created new features using a Python’s Natural Language Tool Kit and regular expressions. This included tokenizing the words in the request\_text and request\_title data fields to help build features, such as word count and count of words that are greater than 10 characters long. I defined a function to check for the lexical diversity of each request (number of unique words divided by the total). I used regex to create categorical features for “please” and profanity in requests. Lastly, I parsed the timestamp to create a feature for request made on weekdays.

I plotted histograms for each feature to identify the distribution of the data. This dataset suffered from class imbalance in almost all of the data fields. For example, most of the user accounts were only a day old, and a vast majority had never posted nor commented before. These variables were less helpful than others such as word count. I also ran a scatterplot matrix for all of the data, however, it was not as helpful as most of my data was categorical and not continuous.

I wanted to experiment with topic modeling so I used GENSIM. I fit a Latent Dirichlet Allocation (LDA) model and generated 10 different topics from the corpus of all the requests. This is a form of unsupervised learning. I could have chosen any number of topics, but I chose 10 for simplicity. The topics aren’t labeled. They are just a collection of words, each with its own probability of occurring in that topic. I then assigned a topic number to each request using this probability distribution. I did not use the results in my model because the probability of getting pizza for each topic was not significantly different.

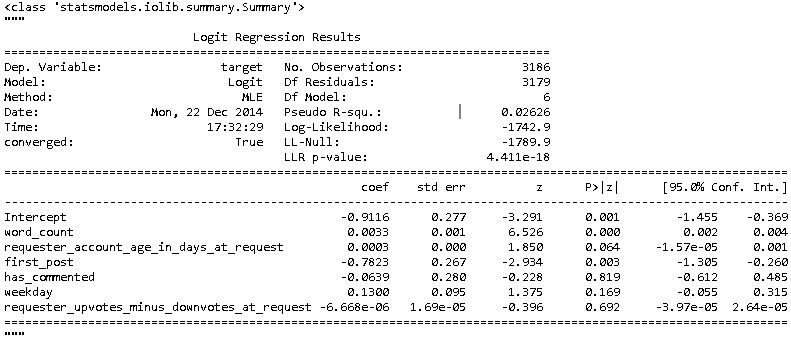
Though LDA is generally regarded as the most appropriate algorithm to use for topic modeling, I was most interested in learning all the variations, so I actually ran four different algorithms, including Latent Semantic Indexing (LSI), Random Projections (RP), and Heirarchical Dirichlet Process (HDP). The results were not ground breaking, but it satisfied my curiosity.

## Model

I split the data into training and test sets, then fit the data to a logistic regression model using the following features: word count of the request, the account age of the requester, whether it was the requesters first post on reddit, whether the requester had commented on reddit before, whether the request was made on a weekday, and the requester’s overall karma score (upvotes minus downvotes). Word count, account age, and whether it was the requester's first post all have a statistically significant effect on whether the requester received pizza. I classified the target as 1 if the probability of receiving pizza was greater than or equal to .5, and 0 if it was less than .5. Since again, this data had a huge class imbalance, I also tried setting the threshold to .3.

Here are the results of my regression:

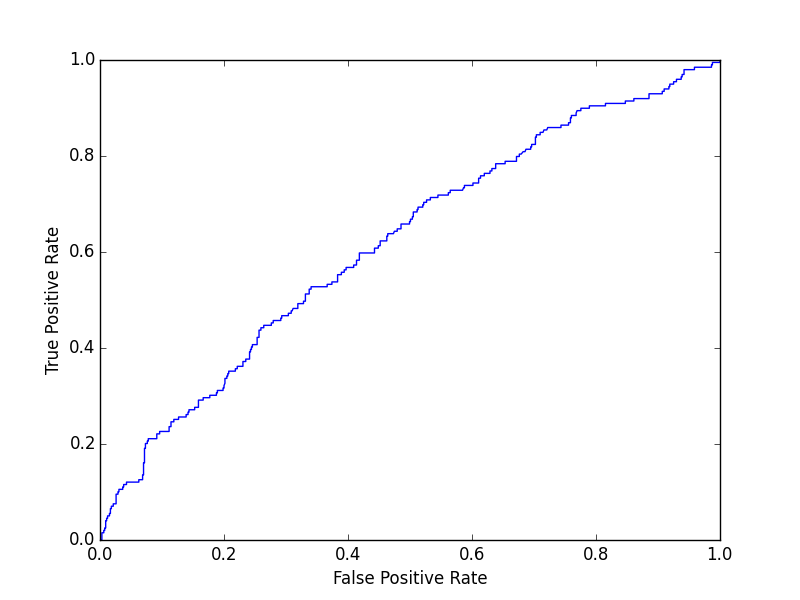
*pizza = smf.logit('target ~ word\_count + requester\_account\_age\_in\_days\_at\_request + first\_post + has\_commented + weekday + requester\_upvotes\_minus\_downvotes\_at\_request', data = train).fit()*

**

Word count and whether it was the user’s first post have a statistically significant effect on whether the request results in free pizza with a p-value < .05. The requester’s account age is also significant at the 6% level.

## Conclusions

I was glad to see that the accuracy on my model was 77%. I was also excited to see that the specificity of my model was about 99%. However, I was quickly humbled when I calculated the sensitivity at about 5%. As it turns out, very few request result in free pizza. The ROC curve for my predictions is plotted below:



While I found a few statistically significant features, such as word count, first post, and account age, this model does not accurately classify requests that result in free pizza. Further experimentation is required using better features or a different modeling algorithm. I’m currently sitting pretty at 258 on the Kaggle leaderboard:

