That’s Offensive!

Detecting Insults in Social Commentary

By: Andrew Lin, John Sun, Shoaib Rana, and Eric Jess

Overview

Our Data

Comes from a 2012 Kaggle competition about detecting when a comment from public discussion threads would be considered insulting to another participant in the conversation

Our Aim

Predict whether a specific comment is considered an insult or not.

Get on the top 10 leaderboard for that competition

In ML terms

Create a generalizable binary classifier which would predict a given comment as either an insult or not through text analysis and ML models

Training Data & Goal

Insult	Date	Comment
 0 1	[null] 18% 20120515161425Z 0% Other (3227) 82%	3935 unique values
1	20120618192155Z	"You fuck your dad."
0	20120528192215Z	"i really don't understand your point.\xa0 It seems that you are mixing apples and oranges."
0		"A\\xc2\\xa0majority of Canadians can and has been wrong before now and will be again.\\n\\nUnless y...
0		"listen if you dont wanna get married to a man or a women DONT DO IT. what would it bother you if ga...
0	20120619094753Z	"C\xe1c b\u1ea1n xu\u1ed1ng \u0111\u01b0\u1eddng bi\u1ec3u t\xecnh 2011 c\xf3 \xf4n ho\xe0 kh\xf4ng ...

1	▲28	Vivek Sharma		0.84248	5	8y
2	▲35	tuzzeg		0.83977	5	8y
3	▲17	Andrei Olariu		0.83867	5	8y
4	▲3	joshnk		0.83632	5	8y
5	▲9	Yasser Tabandeh		0.83321	5	8y
6	▲22	Andreas Mueller		0.82987	5	8y
7	▲26	Willie Liao		0.82984	5	8y
8	▲13	book_face		0.82879	2	8y
9	▲26	D'yakonov Alexander		0.82481	2	8y
10	▲36	Vik Paruchuri		0.82307	3	8y

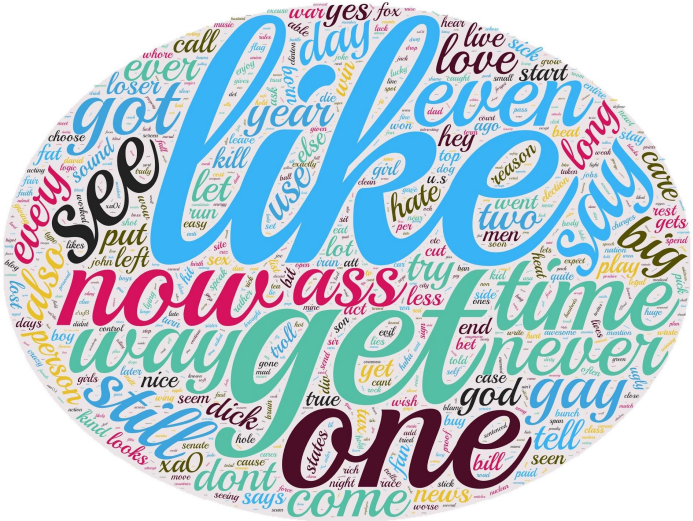
Leaderboard

Our Process

Data Exploration → Feature Engineering → Model Selection

Data Exploration

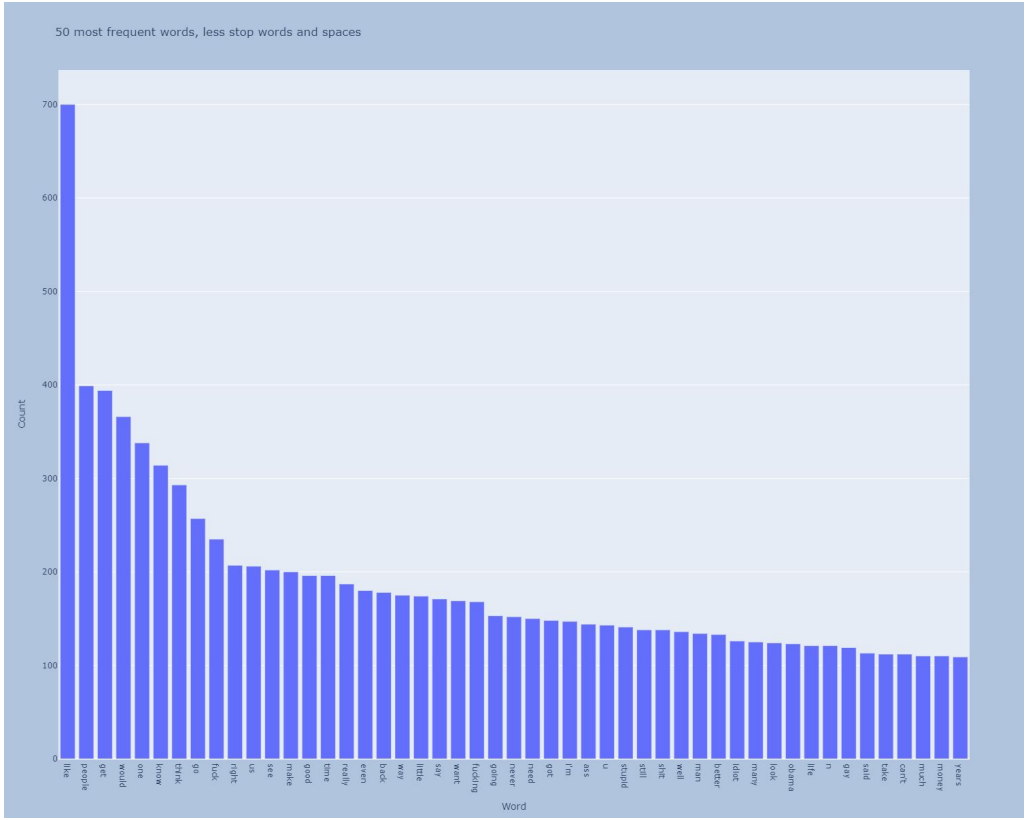
- A Word Cloud was made with all the words in the comments column using wordclouds.com.
- As we can see, the most common words in the comments include like, get, ass, now, and time.
- We also discovered that 27% of the comments were labeled as insults.



Data Visualization

- The 50 most common words are represented in this graph.

- “like”, “people”, “get”, “would”, and “one” make up the top 5.



Feature Engineering

The “5” features we decided to use were:

1. Length of Text
2. Dummy variable for if “You” was present
3. Textblob.sentiment
4. Textblob.subjectivity
5. Bag of words of the 500 most common words in the comments

We decided to use a dummy variable for you as a feature because we figured that if someone was giving an insult, they would use the words “you”, “your”, and “you’re” to throw an insult.

Since we believed that insults would have negative sentiment and would be more subjective than objective, we used Textblob’s sentiment and subjectivity methods.

CountVectorizer just got the most common 500 words in the comments and it really improved our model evaluation metrics.

Feature Engineering Code

This is for the bag of words of the top 500 most common words

```
vectorizer = CountVectorizer(analyzer='word', max_features = 500, stop_words="english")
X2 = vectorizer.fit_transform(text["Comment"])
```

This is the code we used to get every other feature we wanted

```
text["sentiment"] = text.Comment.apply(lambda x: textblob.TextBlob(x).sentiment.polarity)
text["subjectivity"] = text.Comment.apply(lambda x: textblob.TextBlob(x).sentiment.subjectivity)
text["You"] = text["Comment"].str.lower().str.contains("you")
text["You"] = np.where(text["You"], 1, 0)
text["Length"] = text.Comment.apply(lambda x: len(x))
```

Model Selection

We decided to use an ensemble method comprised of:

1. Random Forest Classifier
2. Logistic Regression
3. Bagging Decision Tree Classifier

The Ensemble Model

```
estimators = [('log', log_model), ('rf', forest_model), ('bag', bag_clf)]

voting_total = VotingClassifier(estimators=estimators, voting="hard")

voting_total.fit(X, y)
```

Results

AUC: 0.724
Accuracy: 0.828
Recall: 0.506
Precision: 0.755
F1-Score: 0.606

Confusion Matrix

TN	[[1840 114]	FN
FP	[342 351]]	TP

An AUC of 0.72407 would put us at 41st place

39	▲4	Aleksandra		0.72749	1	8y
40	▲7	Foxtrot		0.72701	1	8y
41	▼7	--- Our Model ----		0.72407	2	8y

Us!

Impact & Ethics

- Detecting insults in public threads with accuracy may be able to identify cyberbullying, which can be beneficial for whatever online community this model is implemented by

- Something to be aware of is that the classifier does not automatically progress with time. Changing of times can lead to certain ideas, values and therefore phrases previously deemed as inappropriate be considered no longer an insult

Future Improvements

- nltk has a SentimentAnalyzer module that will do a lot better job than Textblob
- Textblob doesn’t have an extensive vocabulary, so if it sees a word not in the vocabulary it will just assign it a sentiment of 0
- Extending into n-grams and using neural networks may also improve the predictions
- Some of the comments we observed were objectively insults, but were not classified as such. Going through each comment and finding these mistakes may also help

Challenges

None of us had worked with text data before

Busy with exams and school work. More time would allow for better feature engineering

Making slight modifications required that model be retrained all over again

Learning Curve

Comment Formatting

Time Constraints

Computing Resources

Future Modifications

Some comments were terribly formatted so challenging to classify

Could not connect to a GPU or TPU so finding optimal hyperparameters took a long time

Thank You!