Quora Insincere Question Classification

Team VSM (Team 7)

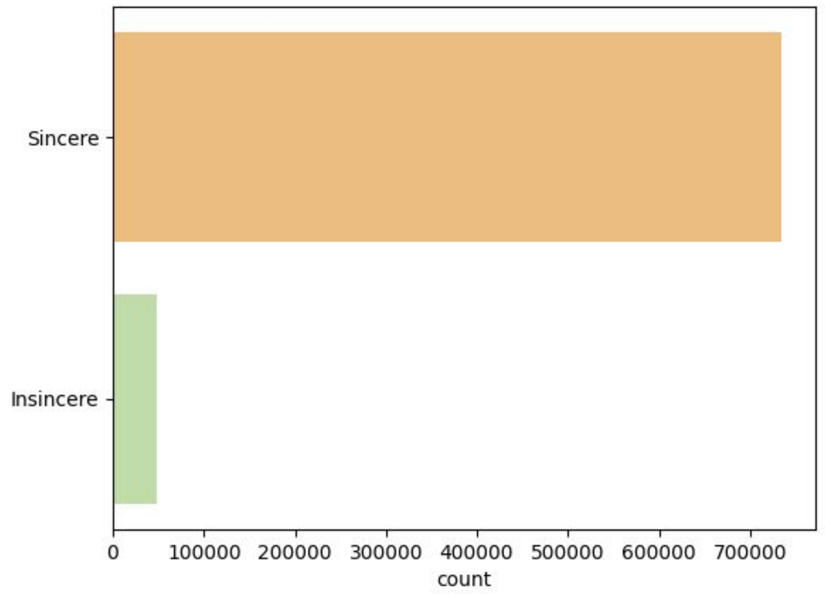
Sai Manish Sasanapuri	(IMT2018520)
Shubhayu Das	(IMT2018523)
Veerendra S Devaraddi	(IMT2018529)

Analysis

Preliminary look

In a classification problem, the first thing to look at is the class distribution. This can be very insightful:

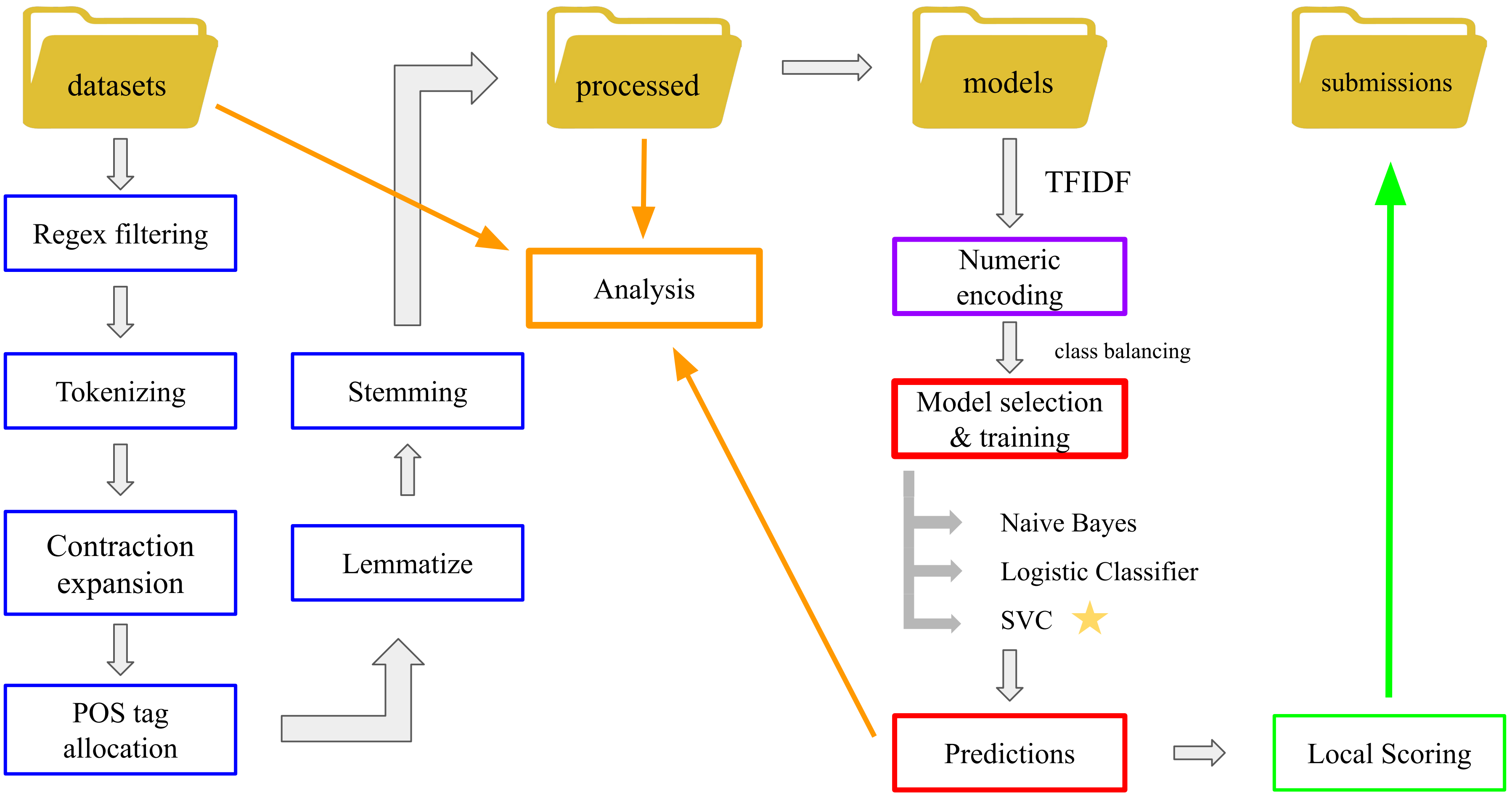
Class distribution in the train dataset



A lot of the other analysis will be explained along with the preprocessing with relevant examples.

Workflow

An overview


datasets
processed
models
submissions
Regex filtering
Tokenizing
Contraction expansion
POS tag allocation
Lemmatize
Stemming
Analysis
TFIDF
Numeric encoding
class balancing
Model selection & training
Naive Bayes
Logistic Classifier
SVC
Predictions
Local Scoring

Preprocessing

Regex filtering

- Utilizing regex to detect some troublesome string sequences.
- It is a working progress, regex expressions are not perfect yet.

qid: ea5c30e04c4c4beff7bb

Who wishes to join ISIS? Will you join ISIS for **5000000000000067986499753345678 97600000000** dollars?

r”\w{0,5}[0-9]{3,30}\w{0,5}” →

Replace with *snumexpr*

qid: ea5c30e04c4c4beff7bb

Who wishes to join ISIS? Will you join ISIS for **snumexpr** *00* dollars?

10kg, 10cm, 10m, 10%, US$10
….

.replace(r'(\d+)', r' \1 ', flags) →

10 kg, 10 cm, 10 m, 10 %, US$ 10
….

Similar manipulations helped remove **chemical formulae, long numbers, website URLs** and **symbols**.

1. Credits to https://regex101.com/ for helping in designing the regex expressions and testing

Tokenizing and Contraction Expansion

Tokenizing is a process of dividing the sentence into words.

qid: d49b3966070b27bf07fc

'What did they use for transportation in Ancient India?'

$\xrightarrow{\text{nltk.word\_tokenize(text)}}$

qid: d49b3966070b27bf07fc

['What', 'did', 'they', 'use', 'for', 'transportation', 'in', 'Ancient', 'India', '?']

Contraction expansion is done using a python module *contractions*. This reduces the number of unique words present in the dataset.

qid:69e9ac8cfacd84eeeb0b

[Why, cant, diabetics, eat, cornbread]

$\xrightarrow{\text{.apply(contractions.fix(text))}}$

qid:69e9ac8cfacd84eeeb0b

why can not diabetics eat cornbread

The number of unique words reduced from 217989 to 185711 words which is reduction of 32278 unique words.

Positional Tag allocation

- Utilizing nltk's positional tagger, which works based on the Penn Treebank tagset.
- The tags are then modified to work with our Lemmatizer.

qid: 6d5faa49380557c8ca7b

['What', 'are', 'the', 'most', 'important', 'provisions', 'of', 'Obamacare?']

$\xrightarrow{\text{nltk.tag.pos\_tag}}$

qid: 6d5faa49380557c8ca7b

[('What', 'WP'), ('are', 'VBP'), ('the', 'DT'), ('most', 'RBS'), ('important', 'JJ'), ('provisions', 'NNS'), ('of', 'IN'), ('Obamacare', 'NNP')]

Problems:

- The vocabulary of the classifier is not extensive, leading to misclassification.
- Too many output classes,which are eventually not utilized

Possible solution is to train our own tagger, but that will require a huge labelled corpus. Other possibility is to utilize rule based taggers in combination with this.

Language translation

Using langdetect library for detecting language and using googletrans library to translate non english texts detected to english.

For example :

Qid: 3a7af5718b866b9994ec Qid: 3a7af5718b866b9994ec

Aaj bhahut thand hai aaj kam par nhi hai? $\xrightarrow{\text{langdetect}}$ Aaj bhahut thand hai aaj kam par nhi hai?**,id**

Qid: 3a7af5718b866b9994ec Qid: 3a7af5718b866b9994ec

Aaj bhahut thand hai aaj kam par nhi hai? $\xrightarrow{\text{googletrans}}$ It is very cold today, not at work today?

Spelling correction

Ideally, correcting spelling can decrease the number of unique words. More often than not, this improves the model's performance.

Problems:

1. Spelling correction module can correct words from one language at a time.
2. It is computationally expensive.
3. Spelling correction module needs a large and diverse corpus to get higher accuracy.

qid: 09fadb0f1951620926e8

['how','do','i','say', 'tumhe','tumhari', 'maa', 'ki','kasam','agar', 'tumne', 'meri', 'baat', 'nahi','maani','toh','in','english']

$$\xrightarrow[\text{Spelling correction for non proper nouns}]{\text{autocorrection.Speller(fast=True)}}$$

qid: 09fadb0f1951620926e8

how do i say tumhe tumhari maa ki kasam agar tune merit boat nazi mani toh in english

Observations:

1. Score of the model reduced.
2. # of unique words got reduced from 185711 to 143764 because of small corpus size.

Removing stopwords

Stopwords is a list of word which might not help the model to learn on the dataset. Stopwords were collected from nltk.corpus.stopwords .

qid: 6f47b0f60633c2056455

['how', 'can',‘i', 'reply', 'to', 'this', 'comment', 'india','is', 'poor', 'it', 'is', 'a', 'fact', 'i', 'do', 'not', 'understand', 'the', 'unnecessary', 'criticism', 'of', 'snapchat', 'ceos', 'statement']

$\xrightarrow{\text{Stopwords removed}}$

qid: 6f47b0f60633c2056455

['reply', 'comment', 'india', 'poor', 'fact', 'understand', 'unnecessary','criticism', 'snapchat','ceos','statement']

Observations:

1. # of unique words reduced from 185711 to 185560 which is a reduction of 151 words.

Stemming and Lemmatization

Stemming and Lemmatization are usually used to normalize the words to their base form. The main difference between stemming and lemmatization is lemmatization considers the context and converts the word to its meaningful base form, whereas stemming just removes the last few characters, often leading to incorrect meanings and spelling errors.

Qid :36f8df01de41c56243ba

Qid :36f8df01de41c56243ba

"How Korean students don't get **tired** or be **energetic** after **studying** long time?" $\xrightarrow{\text{Stemming}}$ ['how', 'korean', 'student', 'do', "n't", 'get', '**tire**', 'or', 'be', **‘energet'**, 'after', **'studi'**, 'long', 'time', '?']

"How Korean students don't get **tired** be **energetic** after **studying** long time?" $\xrightarrow{\text{Lemmatization}}$ ['How', 'Korean', 'student', 'do', "n't", 'get', '**tired**', 'or', 'be’, or **'energetic',** 'after**', 'study',** 'long**',** 'time**', '?']**

Stemming

Snowball Stemmer :Snowball stemmer supports many languages but here we are utilizing English only(so far). It is We compared it to the Porter stemming algorithm, which gives inferior results.

An example: Word **generously** when stemmed using Snowball stemmer it get stemmed to **generous** and when stemmed using original Porter stemmer it gets stemmed to **gener.** This is why we are using snowball stemming.

Problems: Over-stemming and Under-stemming

Observation:

1. When stemming applied with other preprocessing steps (Regex filtering, Tokenization and Contraction expansion) , no of unique words reduce from 185711 to 106827.

Lemmatization

Used Wordnet Lemmatizer with NLTK to lemmatize sentences. Wordnet is an large lexical database for the English. Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept. To make the lemmatization better and context dependent, we would need to find out the POS tag and pass it on to the lemmatizer.

"The **striped** bats **are hanging** on their feet for best" ‘ Lemmatizing without Pos tag $\longrightarrow$ ['The', **'striped'**, 'bat', **'are'**, **'hanging'**, 'on', 'their', 'foot', ' ‘for' , 'best']

"The **striped** bats **are hanging** on their feet for best" Lemmatizing with Pos tag $\longrightarrow$ ['The', **'strip'**, 'bat', **'be'**, **'hang'**, 'on', 'their', 'foot', 'for', 'best']

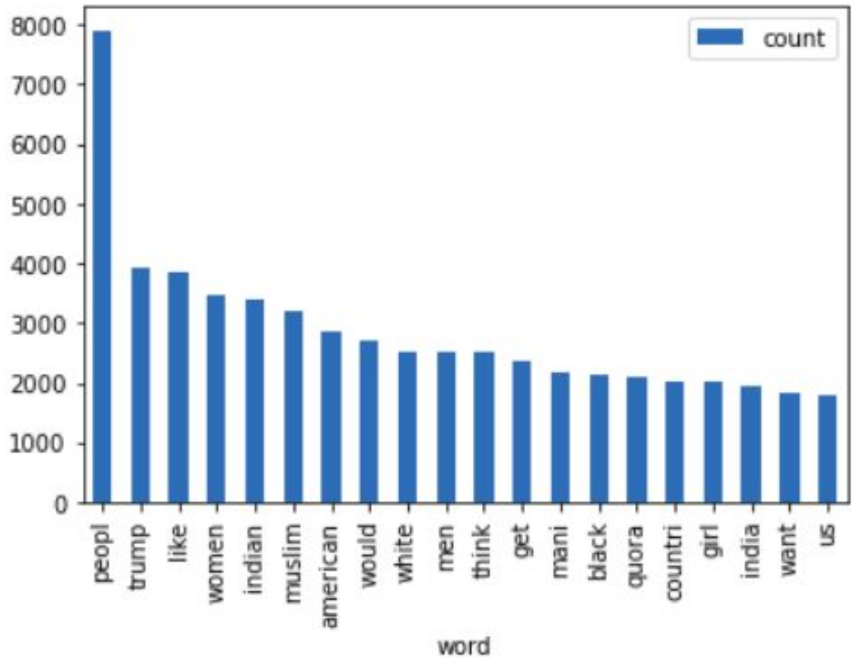
Observation :

1. When Lemmatization applied with other preprocessing steps (Regex filtering, Tokenization and Contraction expansion) , no of unique words reduce from 185711 to 127867
2. When Stemming applied after Lemmatization with other preprocessing steps (Regex filtering, Tokenization and Contraction expansion) , no of unique words reduce from 185711 to 106246

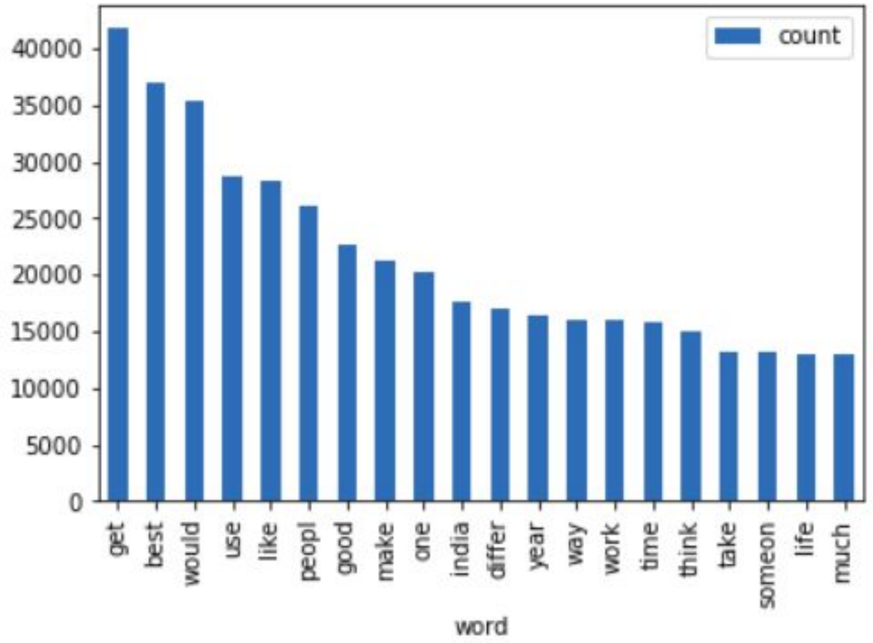
Creation of ngram

- Frequency vs word bar plot

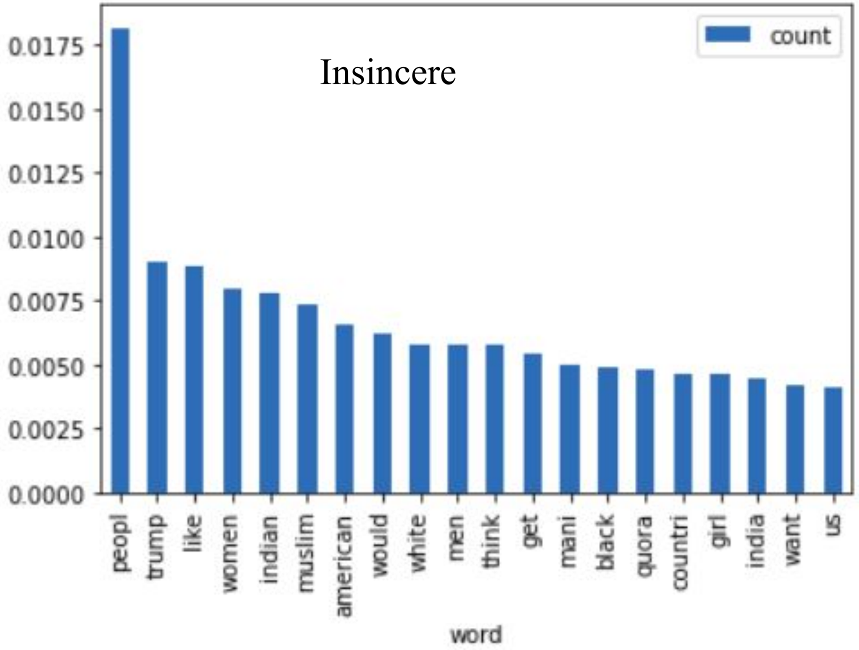
Insincere

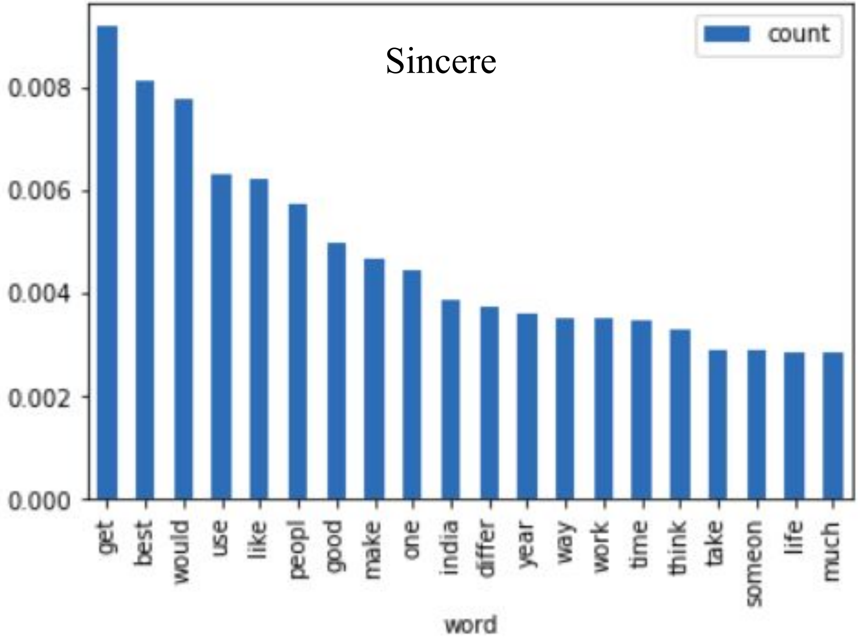


Sincere



- Different type of word cases
  - Word is present in approximately same **proportion**, in both the category. Eg. “people”
  - Word’s **proportion** is higher in only one category. Eg “Trump” (0.009,0.001)

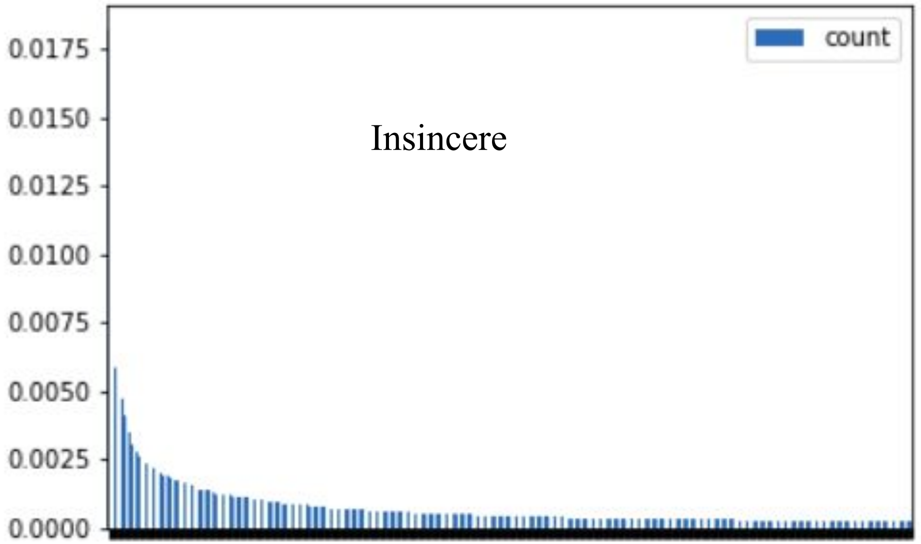


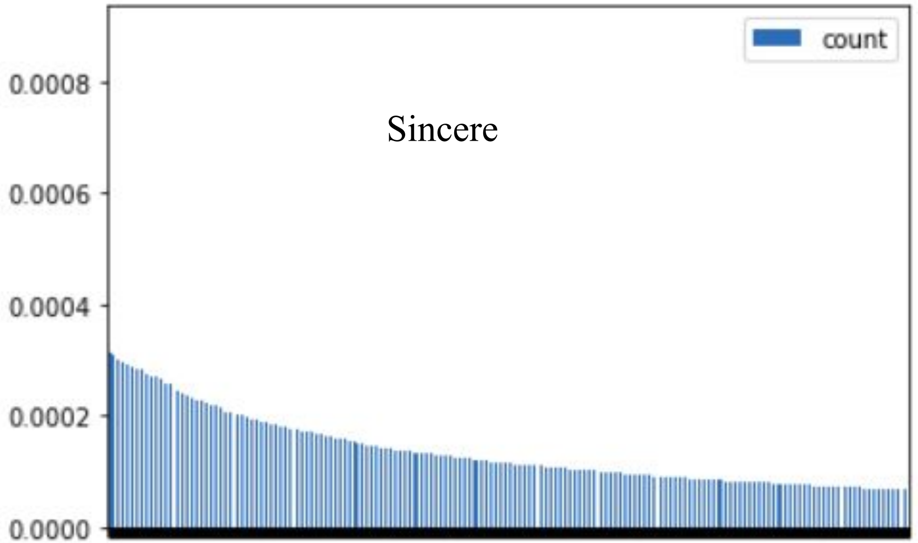


Removing certain words changes model’s intention….

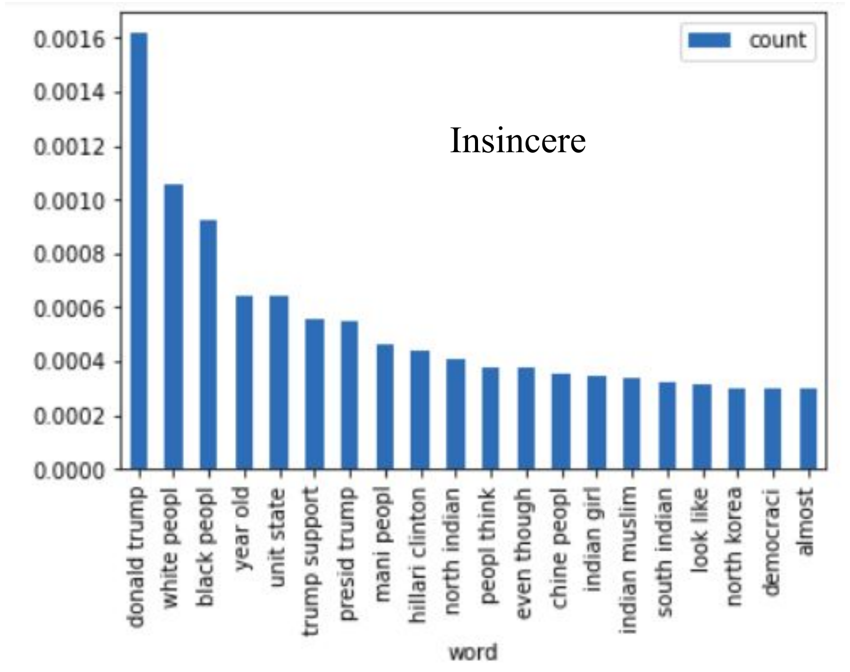
Removing the words is not the solution because….

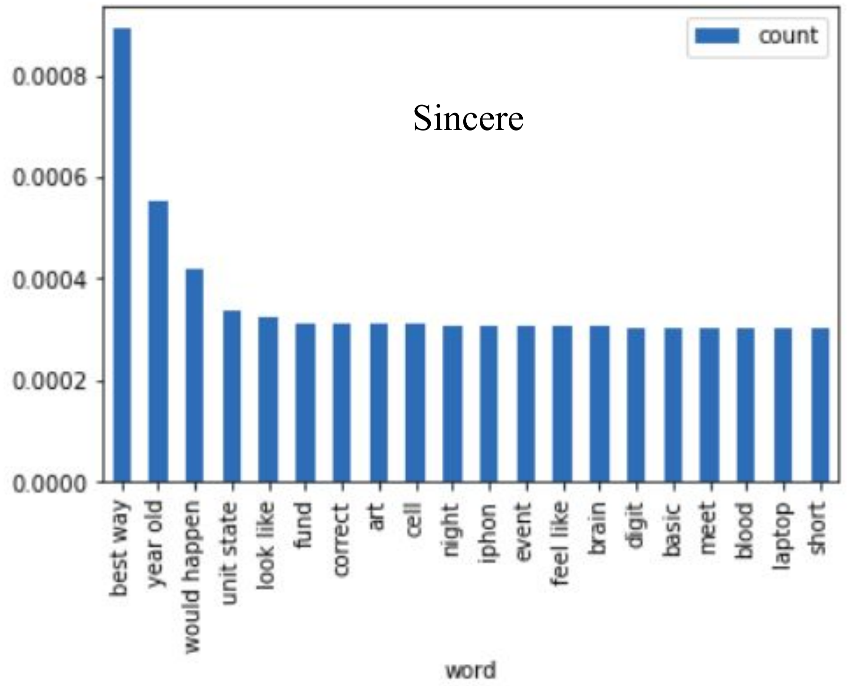
- Top 900 unigram words distribution





- (unigram + bigram) frequency vs word plots





Encoding from text to numbers

Embeddings

A word embedding is a learned representation for text where words that have the same meaning have a similar representation. Each word is represented in the form of a n-dimensional vector(n varies based on model size and implementation). ex: Word2Vec, GloVe

- It can hande new words which it never saw in its training.
- Relation between the words is captured.

References

1. Word Embedding Explained, a comparison and code tutorial *by Duncan Cam-Stei*
2. What Are Word Embeddings for Text? *by Jason Brownlee*

Term frequency-inverse document frequency (TFIDF)

1. Bag of words based construct.
2. Involves the product of **term frequency** and **inverse document frequency**

$$\mathbf{TF}_{\mathbf{i,\,j}} = \frac{N_{i,\,j}}{\Sigma_k N_{i,j}} \quad * \quad \mathbf{IDF(w)} = \log \frac{N}{df_w}$$

Some of the problems include large sparse matrix, loss of ordering of words, no regard for class distribution.

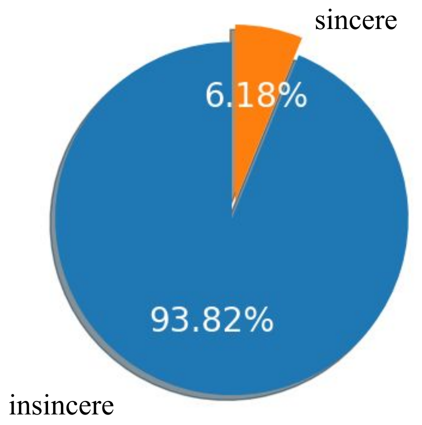
Can be modified to include a other terms, based on a variety of parameters[1]. We are currently experimenting with these variations.

[1] Modified TF-IDF Term Weighting Strategies for Text Categorization

Model Selection

Resampling

Class distribution of training dataset



1. Using SMOTE[1] algorithm to oversample the minority class, until the minority class forms 15% of dataset.
2. Randomly subsampling the majority class, until the minority class forms 20% of dataset.

We are utilizing the imblearn library for this.

Preprocessed dataset

dims: (783673, 250000)

SMOTE →

Oversampled minority in dataset

dims: (845505, 250000)

RandomUnderSampler →

Subsampled majority + oversampled minority

dims: (661698, 250000)

[1] N. V. Chawla, K. W. Bowyer, L. O.Hall, W. P. Kegelmeyer, "SMOTE: synthetic minority over-sampling technique," Journal of artificial intelligence research, 321-357, 2002

Basic requirements of the model

- We need a classifier model.
- Capable of dealing with massive class imbalance.
- Capable of scaling to large dimensional inputs, with reasonable training time.
- Capable of dealing with high sparsity in input vector.

Possible candidates:

Naive Bayes classifier	Logistic Regression
Support Vector Machine Classifier	Trees like Random forests/XGBoost

Ensembling?

Naive-Bayes vs Logistic Classifier vs SVC

BernoulliNB

1. Probabilistic model, based on the Bernoulli distribution.
2. The model doesn't infer anything about the data, just assigns weighting coefficients to each feature.
3. Baseline performance. Highest *F1 score ≈ 0.52*
4. Trains in sub-second durations.

LogisticRegressionCV

1. Large dimensionality and high sparsity of dataset leads to convergence issues.
2. Gives decent scores, but takes too long to fit. # of parameters to learn is high
3. Highest F1 score achieved ≈ 0.62*

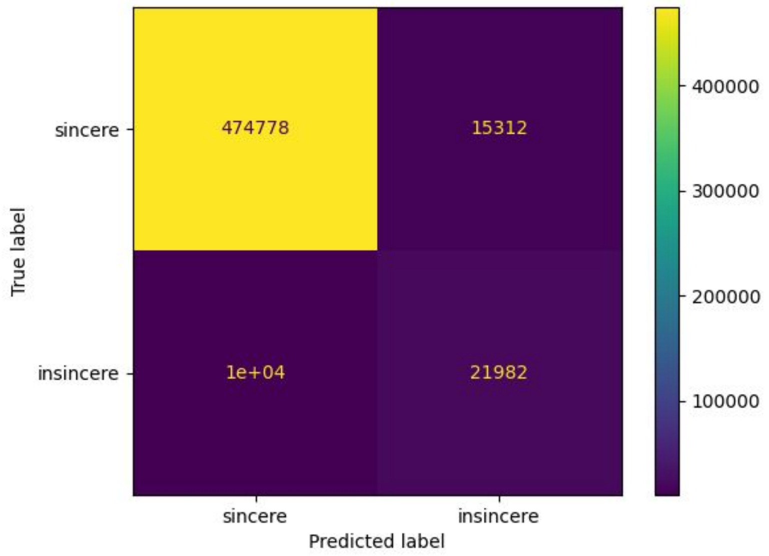
LinearSVC

1. Learns decision boundaries better
2. Forced to use linear kernel in primal formulation.
3. Fits fast(<10 mins)[1], gives very good scores.
4. Highest F1 score = 0.631
5. Handles sparsity well, especially with L1 norm

[1] Liblinear solver

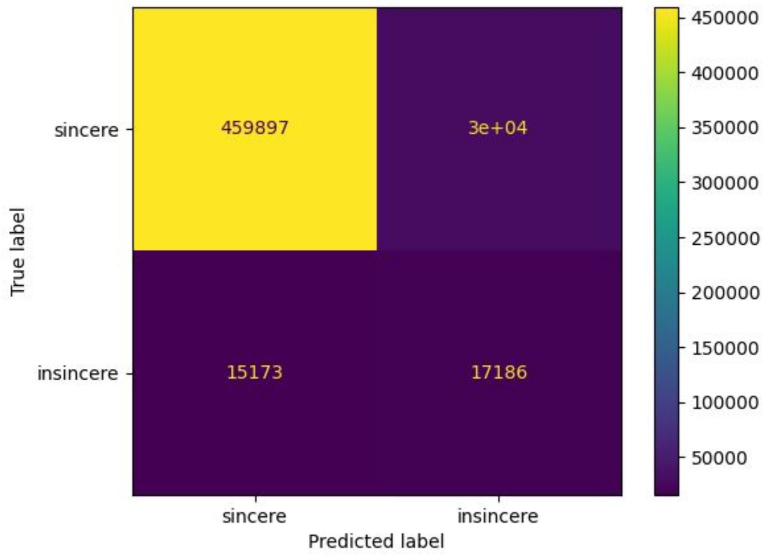
Scoring metrics

Confusion matrix



LinearSVC(C=0.3, penalty='l1', dual=False)

6 minutes to train



LogisticRegressionCV(cv=4, penalty='l1', scorer)

> 20 minutes to train

Precision, Recall and the F1 score

$$\textbf{Precision} = \frac{\text{True positives}}{\text{True positives + false positives}}$$

Out of those predicted to be insincere, how many are actually insincere.

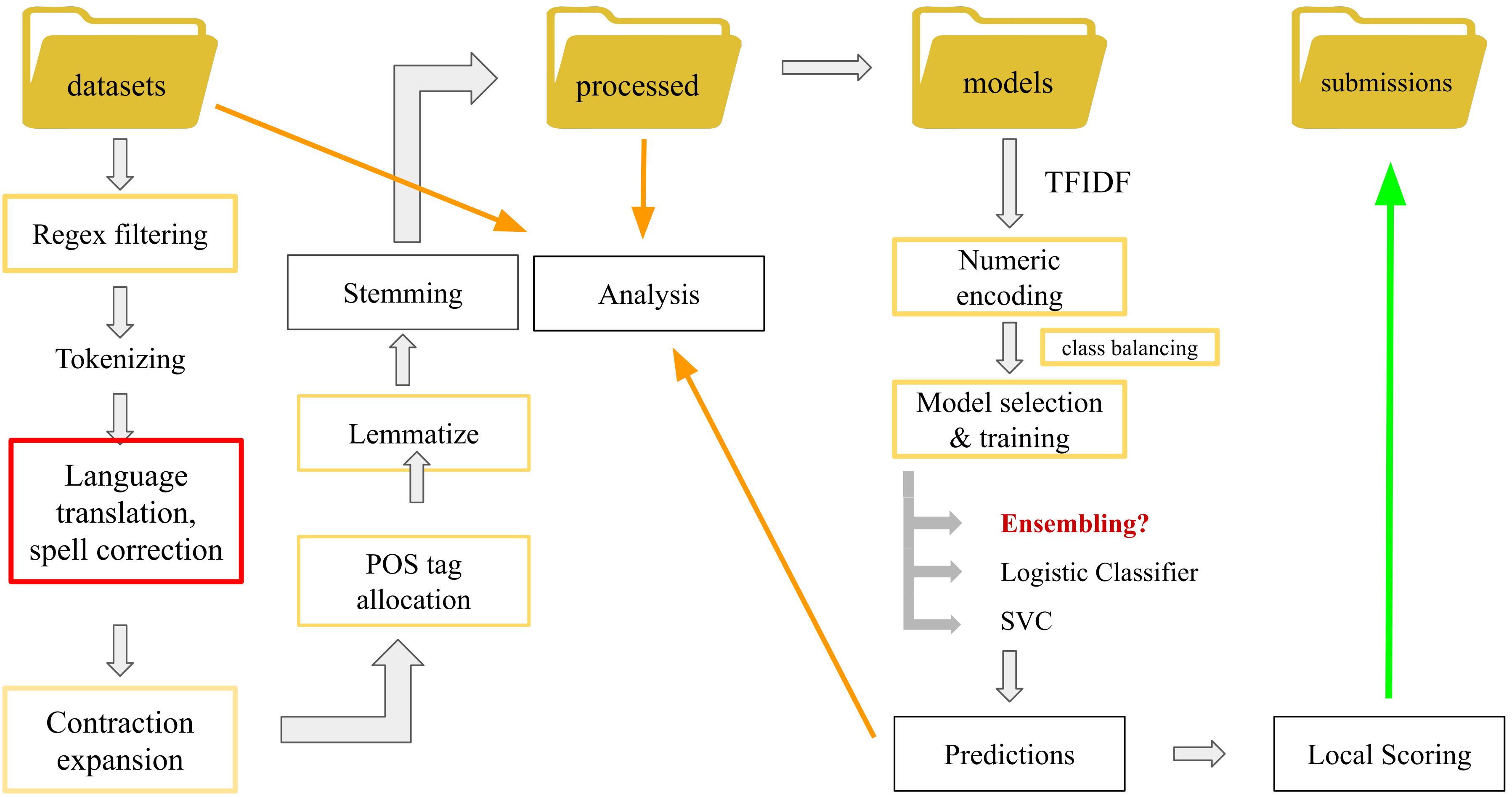
$$\textbf{Recall} = \frac{\text{True positives}}{\text{True positives + false negatives}}$$

Out of the total insincere questions, how many were classified correctly.

$$\textbf{F1 score} = \frac{\text{2(precision + recall)}}{\text{precision*recall}}$$

Harmonic mean of the precision and recall.

Explorations and Improvements


datasets
processed
models
submissions
Regex filtering
Tokenizing
Language translation, spell correction
Contraction expansion
POS tag allocation
Lemmatize
Stemming
Analysis
TFIDF
Numeric encoding
class balancing
Model selection & training
Ensembling?
Logistic Classifier
SVC
Predictions
Local Scoring