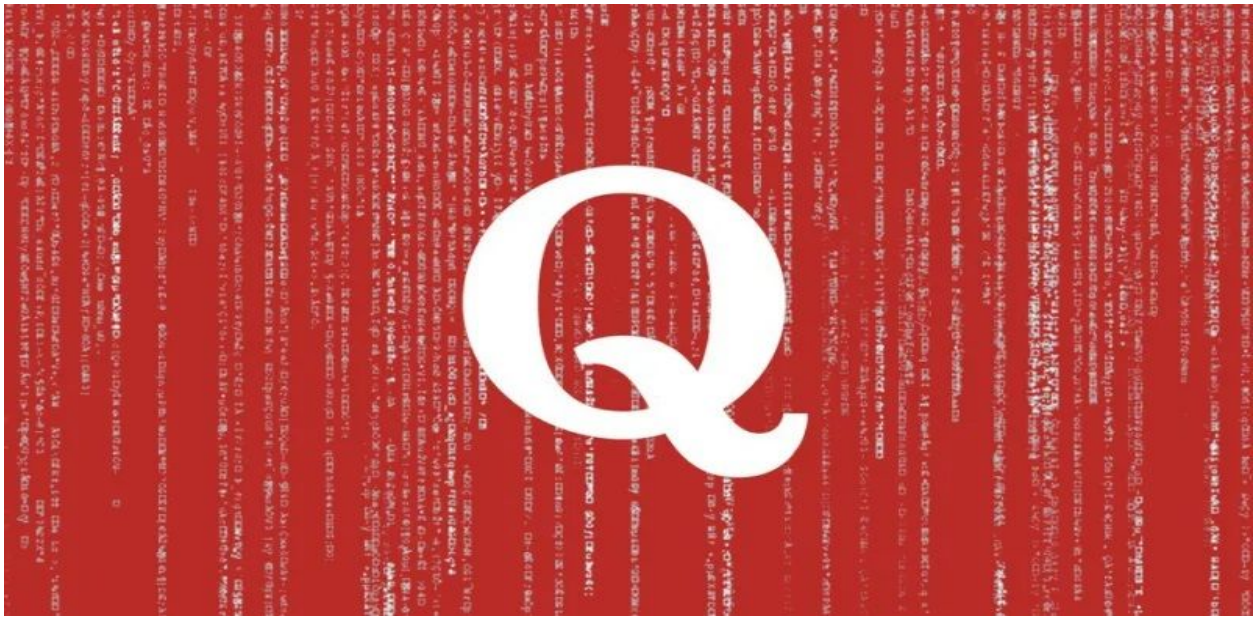


Quora Insincere Questions Classification

Springboard Data Science Career Track Program
Capstone Project # 2

Author : Ashish Mohan Sharma
Reviewer: Kenneth

Publish Data: 07/070/2019

Table of Contents

Introduction:

Quora is a platform that empowers people to learn from each other. On Quora, people can ask questions and connect with others who contribute unique insights and quality answers. Sincere questions are inquiries about which individuals genuinely want to know an answer or gain information from rather than argue a point or make a statement. For instance, “What are some sleep hacks and tips?” is deemed as sincere since it is a candid question which is most likely asked by an individual who would like some advice on falling asleep. On the other hand, insincere questions intend to make some sort of a statement and are usually asked not with the intention of receiving a helpful answer/comment. It is often the case that insincere questions target religion, gender, politics, etc. and are constructed in a non-neutral tone, are exaggerated, or use words that attack various groups. For example, "If blacks support school choice and mandatory sentencing for criminals why don't they vote Republican?" is classified as insincere since it targets the Black community and is phrased in a demeaning manner. A key challenge is to weed out insincere questions -- those founded upon false premises, or that intend to make a statement rather than look for helpful answers.

Here I will try to develop models that identify and flag insincere questions. To date, Quora has employed both machine learning and manual review to address this problem. I will try to help Quora to develop more scalable methods to detect toxic and misleading content and try to combat online trolls at scale. I will try to help Quora uphold their policy of “Be Nice, Be Respectful” and continue to be a place for sharing and growing the world’s knowledge.

Problem Statement:

The goal is to handle unethical comments, in order to improve online conversation on Quora. This is done by identifying them and flagging them as insincere question. An insincere question is defined as a question intended to make a statement rather than look for helpful answers.

The process for solving such problem would be to use a supervised machine learning algorithm (many of them will be tested in this project) that will learn the patterns from

the labelled comments. The final model is supposed to, given a random comment, be able to classify it as ethical (good) or unethical (insincere).

Potential Clients:

Though our model will be specific to Quora but this specific work can be utilized in other Social Media platforms as well to make sure the comments are Ethical and Sincere to be posted on the platform. The model can be integrated as first filter level even before submitting the comment. Those questions that are flagged as insincere wouldn't be allowed to be submitted at all.

It will help Social Media Platforms like Facebook, Instagram, WhatsApp etc to make sure that their users are not bullied or terrified over their system by anyone. That will also help them in creating their own credibility.

Data Acquisition and Understanding:

The data I have used comes from a Kaggle competition and is separated into two datasets, training and testing. The training set consists of a list of questions along with their target values (0 for sincere and 1 for insincere). However, the testing data only contains the sample questions without the target values.

Data can be downloaded from the below link given below.

https://www.kaggle.com/c/quora-insincere-questions-classification/data

An insincere question is defined as a question intended to make a statement rather than look for helpful answers. Some characteristics that can signify that a question is insincere:

- Has a non-neutral tone
    - Has an exaggerated tone to underscore a point about a group of people
    - Is rhetorical and meant to imply a statement about a group of people
- Is disparaging or inflammatory
    - Suggests a discriminatory idea against a protected class of people, or seeks confirmation of a stereotype

    - Makes disparaging attacks/insults against a specific person or group of people
    - Based on an outlandish premise about a group of people
    - Disparages against a characteristic that is not fixable and not measurable
- Isn't grounded in reality
    - Based on false information, or contains absurd assumptions
- Uses sexual content (incest, bestiality, pedophilia) for shock value, and not to seek genuine answers

In addition to this training and testing data, word embedding files (.txt) are used in the final model of this project. This file represents a collection of words with their corresponding embed vector, i.e., the number of features that represent similarities between words. Note: These vectors were obtained by training a neural network (using onehot encoded vectors as inputs), and taking the weights between the input and first hidden layer of the network after training. The embedding matrix here has a size of 2,196,017 by 300. This means that it contains 2,196,017 words, each having a vector of size 300.

I am interested in this specific dataset and the classification of questions as sincere or insincere because we are all active users of Quora. Additionally, I understand that reducing unproductive and harmful content online is a challenge for many tech companies including Quora. This motivated me to gain perspective on the process of moderating online speech.

Metrics:

The F1 score is used for this project to evaluate the performance of the models. While accuracy is a good metric, it will not show a real indication of how the model is performing. Especially since the labels, as we will see in the next section, are highly imbalanced. The model could be saying everything is insincere and still get a high accuracy. F1 score considers both the precision and the recall of the test to compute the score: precision is the number of correct positive results divided by the number of all positive results returned by the model, and recall is the number of correct positive results

divided by the number of all samples that should have been identified as positive. The F1 score is the harmonic average of the precision and recall, where the best score is 1 and worst is 0.

Formula: 2*((precision*recall)/(precision+recall)).

Exploratory Data Analysis:

It is a good practice to understand the data first and try to gather as many insights from it. EDA is all about making sense of data in hand,before getting them dirty with it.

Exploratory Data Analysis (EDA) is an approach/philosophy for data analysis that employs a variety of techniques (mostly graphical) to

1. maximize insight into a data set;
2. uncover underlying structure;
3. extract important variables;
4. detect outliers and anomalies;
5. test underlying assumptions;
6. develop parsimonious models; and
7. determine optimal factor settings.

It is an approach to data analysis that postpones the usual assumptions about what kind of model the data follow with the more direct approach of allowing the data itself to reveal its underlying structure and model.

Data Exploration:

As I explained earlier, we have 2 datasets of which one is Training data while other is Test data. Train data is labeled whether the question is Sincere or Insincere while Test data is not labeled.

Sample of Training Dataset

qid	question_text	target
00002165364db923c7e6	How did Quebec nationalists see their province...	**0**

000032939017120e6e44	Do you have an adopted dog, how would you enco...	**0**
0000412ca6e4628ce2cf	Why does velocity affect time? Does velocity a...	**0**
000042bf85aa498cd78e	How did Otto von Guericke used the Magdeburg h...	**0**
0000455dfa3e01eae3af	Can I convert montra helicon D to a mountain b...	**0**

Looking at the training data I found that there is not data missing in the whole dataset. Also noticed that it has **1225312** records of Sincere data while **80810** of Insincere data. Below are the details of it.

Number of Sincere Questions	**1225312**
Number of InSincere Questions	**80810**
% of Sincere Questions in Train Dataset	**0.938**
% of InSincere Questions in Train Dataset	**0.0619**

Let's explore some of the Sincere and Insincere Questions and try to infer the underlying pattern of both classes intuitively.

Sincere Questions:

1. How did Quebec nationalists see their province as a nation in the 1960s?
2. Do you have an adopted dog, how would you encourage people to adopt and not shop?
3. Why does velocity affect time? Does velocity affect space geometry?
4. How did Otto von Guericke used the Magdeburg hemispheres?
5. Can I convert montra helicon D to a mountain bike by just changing the tyres?

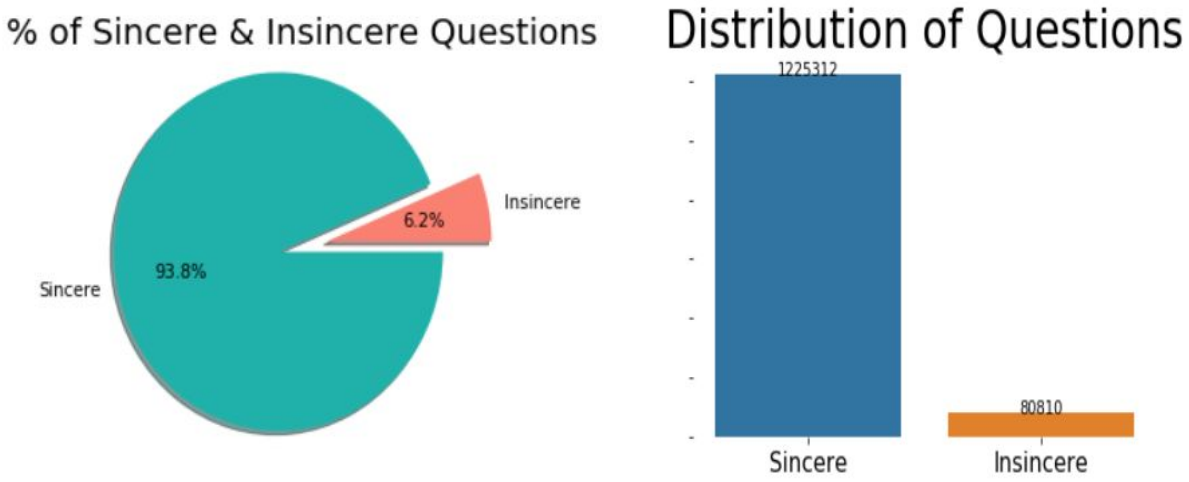
Insincere Questions:

1. Has the United States become the largest dictatorship in the world?
2. Which babies are more sweeter to their parents? Dark skin babies or light skin babies?
3. If blacks support school choice and mandatory sentencing for criminals why don't they vote Republican?
4. I am gay boy and I love my cousin (boy). He is sexy, but I dont know what to do. He is hot, and I want to see his di**. What should I do?
5. Which races have the smallest penis?

From the above insincere questions analysis, we can infer that :

- **They have a non-neutral tone**
  - They have an exaggerated tone to underscore a point about a group of people
  - They are rhetorical and meant to imply a statement about a group of people
- **They are disparaging or inflammatory**
  - They suggests a discriminatory idea against a protected class of people, or seeks confirmation of a stereotype
  - They make disparaging attacks/insults against a specific person or group of people
  - They are based on an outlandish premise about a group of people
  - They make disparages against a characteristic that is not fixable and not measurable
- **They aren't grounded in reality.**
- **They are based on false information, or contains absurd assumptions.**
- **They use sexual content (incest, bestiality, pedophilia) for shock value, and not to seek genuine answers.**

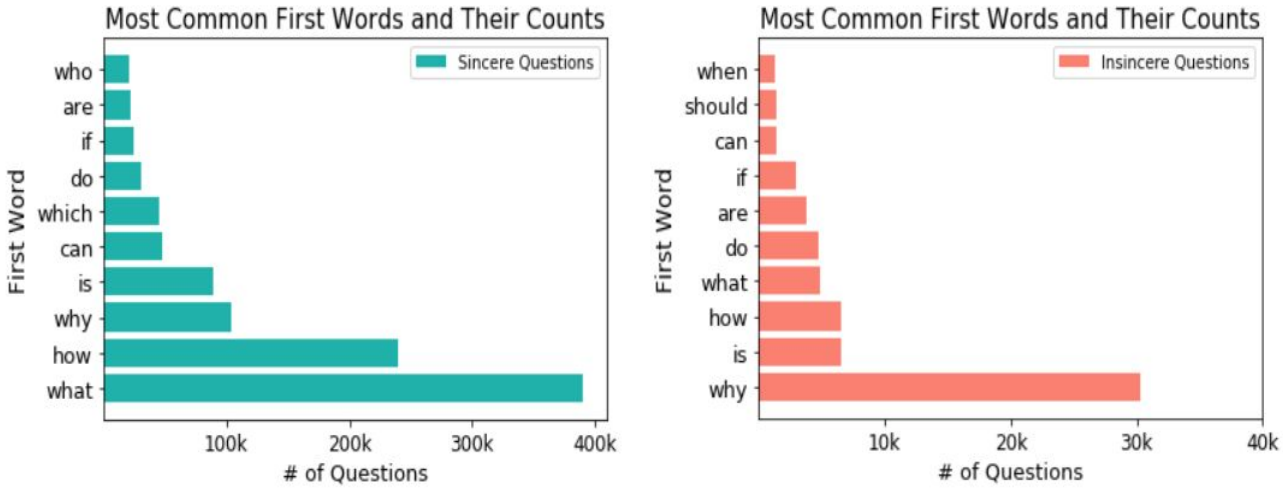
Class Imbalance:



Imbalanced classes are a common problem in machine learning classification where there is a disproportionate ratio of observations in each class. With just **6.6%** of our dataset belonging to the target class, we can definitely have an imbalanced class!

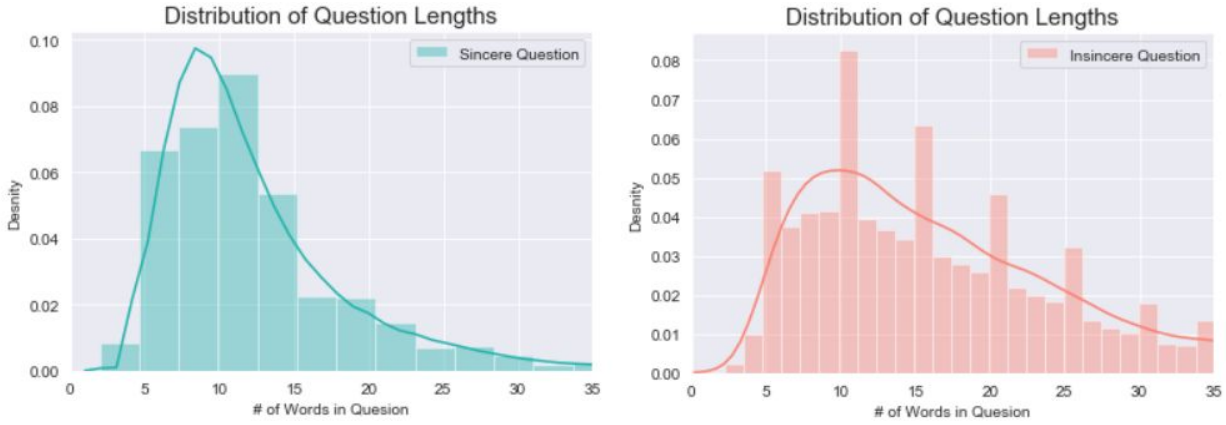
This is a problem because many machine learning models are designed to maximize overall accuracy, which especially with imbalanced classes may not be the best metric to use. Classification accuracy is defined as the number of correct predictions divided by total predictions times 100. For example, if we simply predicted that all questions are sincere, we would get a classification accuracy score of **93%!**

Most Common First Word



There seem to be not much of a difference between the Sincere and Insincere Questions if we analyse most common first words . 10 most frequent words are almost similar like *What, How, Are, Why.* This is intuitive as well, as all are questions so these words should be the part of this result.

Question Length Distribution:



Insincere questions seem to be longer on average and have a larger variance on their length. Sincere questions seem to clump up around word length as 9 while insincere have a less pronounced peek at around 10 words.
When an individual is hoping to assert an opinion, it generally takes more words to do this than asking a question. We noticed that among insincere questions, users were often attempting to state an opinion or make discriminatory remarks about a group. Perhaps, this is why we see the above observation.

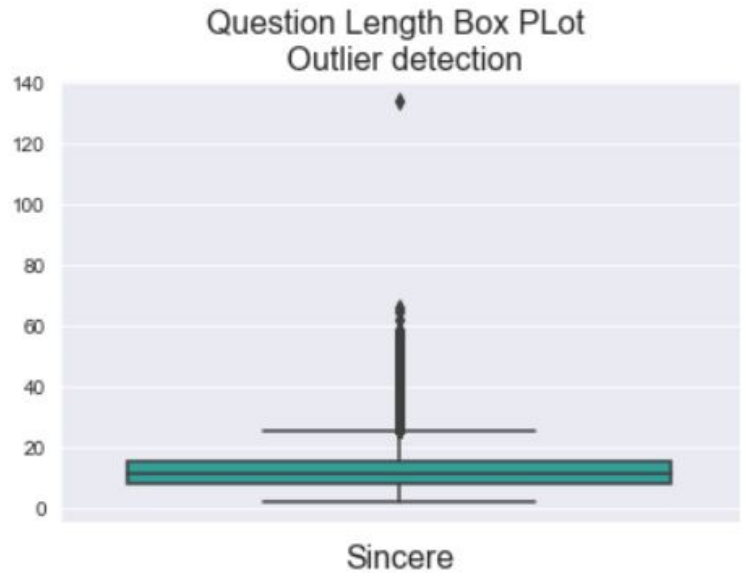
Outlier Detection:

There is an outlier for question length in the Sincere Question group. This question is 134 words long and this user asked 7 questions in one post which resulted in the long length.

```
Longest Sincere Question is:  134  words
```

in "star trek 2013" why did they : *spoilers* *spoilers* *spoilers* *spoilers* 1)make warping look quite a bit like an hyperspace jump 2)what in the world were those bright particles as soon as they jumped. 3)why in the world did they make it possible for two entities to react in warp space in separate jumps. 4)why did spock get emotions for this movie. 5)what was the point of hiding the "enterprise" underwater. 6)when they were intercepted by the dark ship, how come they reached earth when they were far away from her. (i don't seem to remember the scene where they warp to earth). 7)how did the ship enter earth's atmosphere when it wasnt even in orbit. 8)when scotty opened the door of the black ship , how come pike and khan didn't slow down?

Sincere Questions length summary and Box Plot:



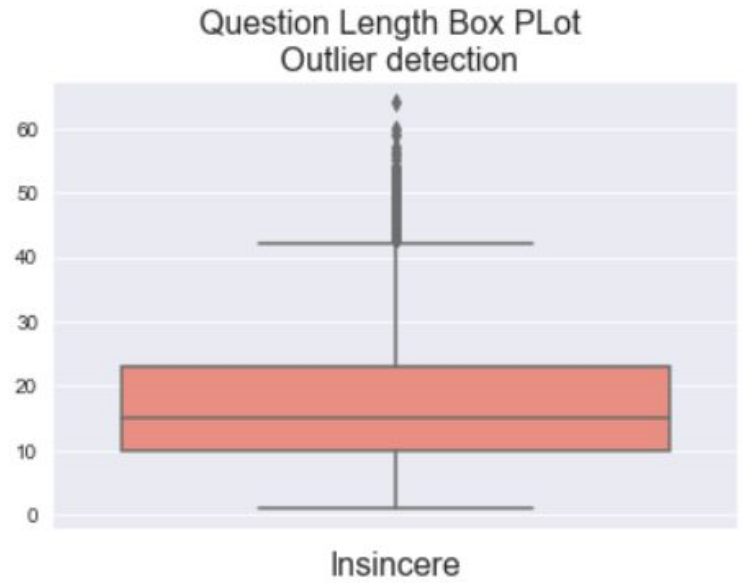
count	1225312
mean	12.5085
std	6.75069
min	2
25%	8
50%	11
75%	15
max	134

Outliers do not seem significant in insincere questions. The longest question length for this type is roughly half as long as the longest sincere question outlier.

`Longest Sincere Question is:  64  words`

to you, does being a christian mean inviting in the spirit of jesus into you and suppressing our own spirit? 'thy will not mine' and all that? do you like living as a zombie of someone else's spirit - however perfect it may be? don't you want to experience and improve your own will and spirit and live your life as you, not jesus?

InSincere Questions length summary and Box Plot:



count	80810
mean	17.2778
std	6.75069
min	1
25%	10
50%	15
75%	23
max	64

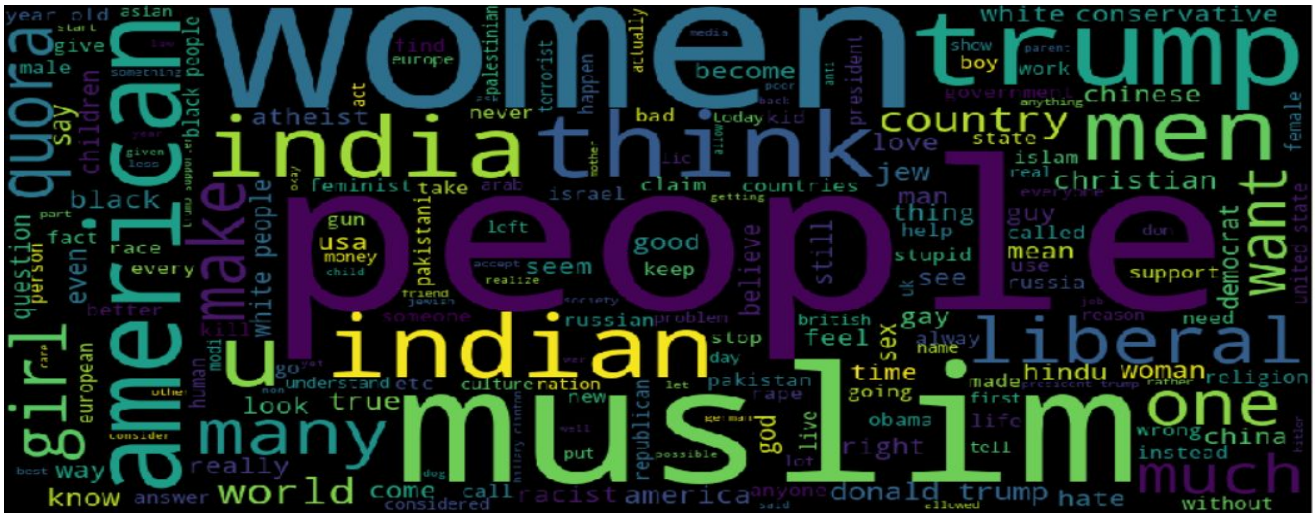
Word-Cloud Visualization:

Word-Cloud Visualization highlights important textual data points, it can make dull data sizzle and immediately convey crucial information.

Sincere Word-Cloud:

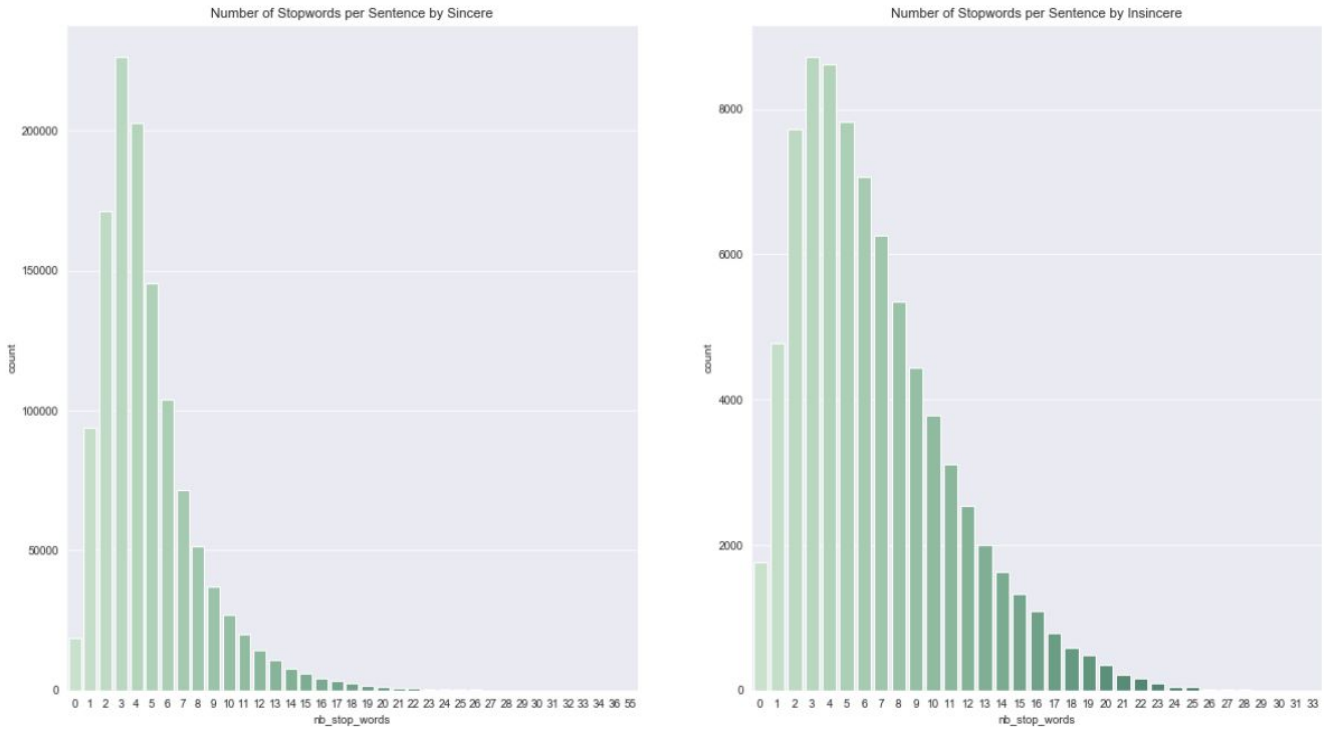
Best , One, Good, Think are really positive words and therefore are categorized as Sincere Questions

Insincere Word-Cloud:



People, Muslim, Trump , Women, girl, hate, sex, racist, gay, lie are the major constituents of the Insincere questions.

StopWords Distribution



Number of stopwords present in the sentence can be a great feature for classifying the Question as Insincere. The above distribution suggests that in Insincere questions, there are a lot more stopwords present. It might be due to more explanation about the negative point they want to convey.

Bigram/Trigram Plots:

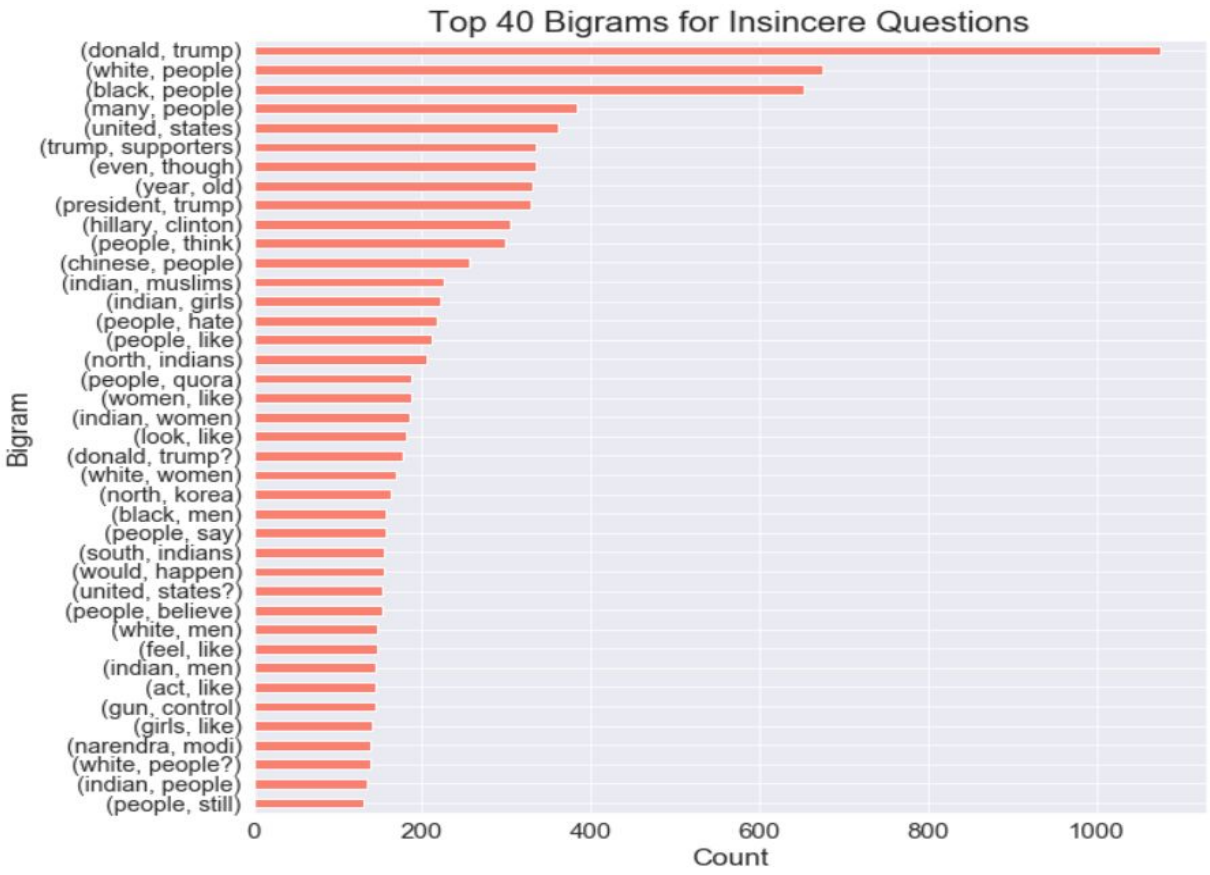
The plots for top bigrams/trigrams in insincere questions can help us determine what kind of questions Quora would like to limit or ban. People are always finding creative new ways to ask insincere questions, and it is important for Quora to be aware of any new forms of insincere questions so as to make changes to its policies and guidelines if need be. The top bigram/trigram plots are one way to group insincere questions in a more intelligible format by pinpointing frequent topics of insincere questions.

There are many potential uses for such findings including this non-exhaustive list:

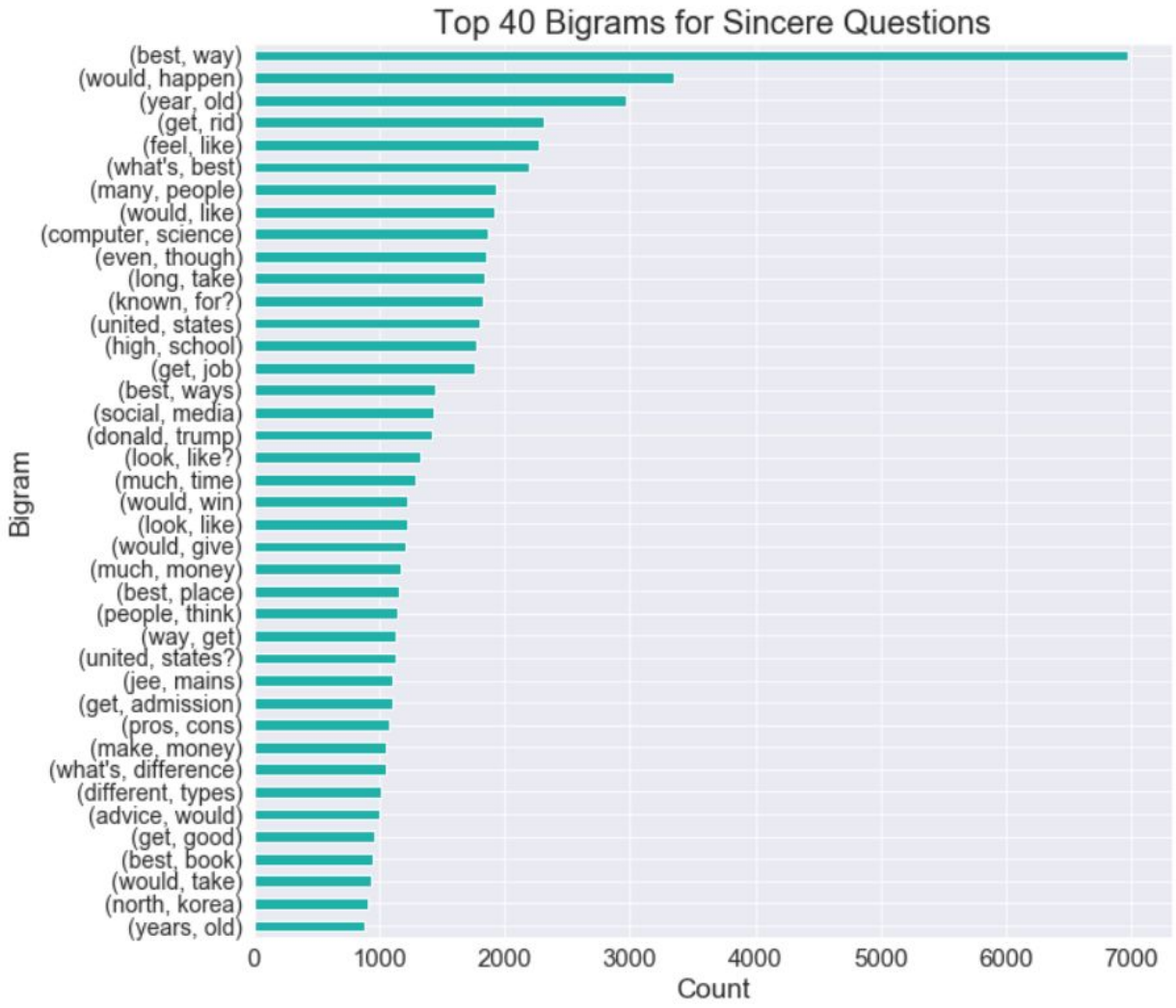
- They can be examined repeatedly to regularly refine Quora's definition of an insincere question.

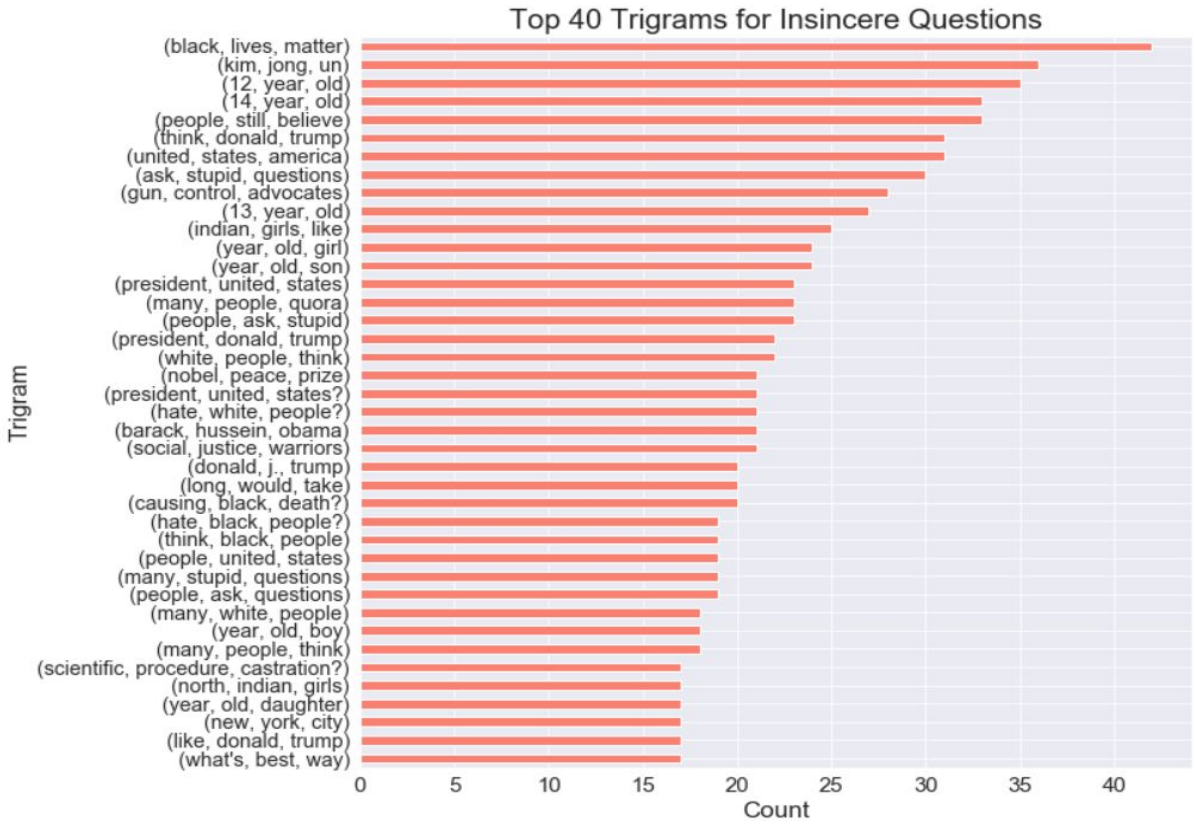
- They can give insight into the level of racism, sexism, and other discriminatory thoughts that are prevalent in the world. Note that although the group of users on Quora is a sample of the population of the world, the approximation for population parameters is not necessarily holistic since the set of all Quora questions is a nonrandom sample. Nonetheless, if our goal is to limit all forms of existing discriminatory thoughts, then using Quora for such an agenda is a step in our intended direction.

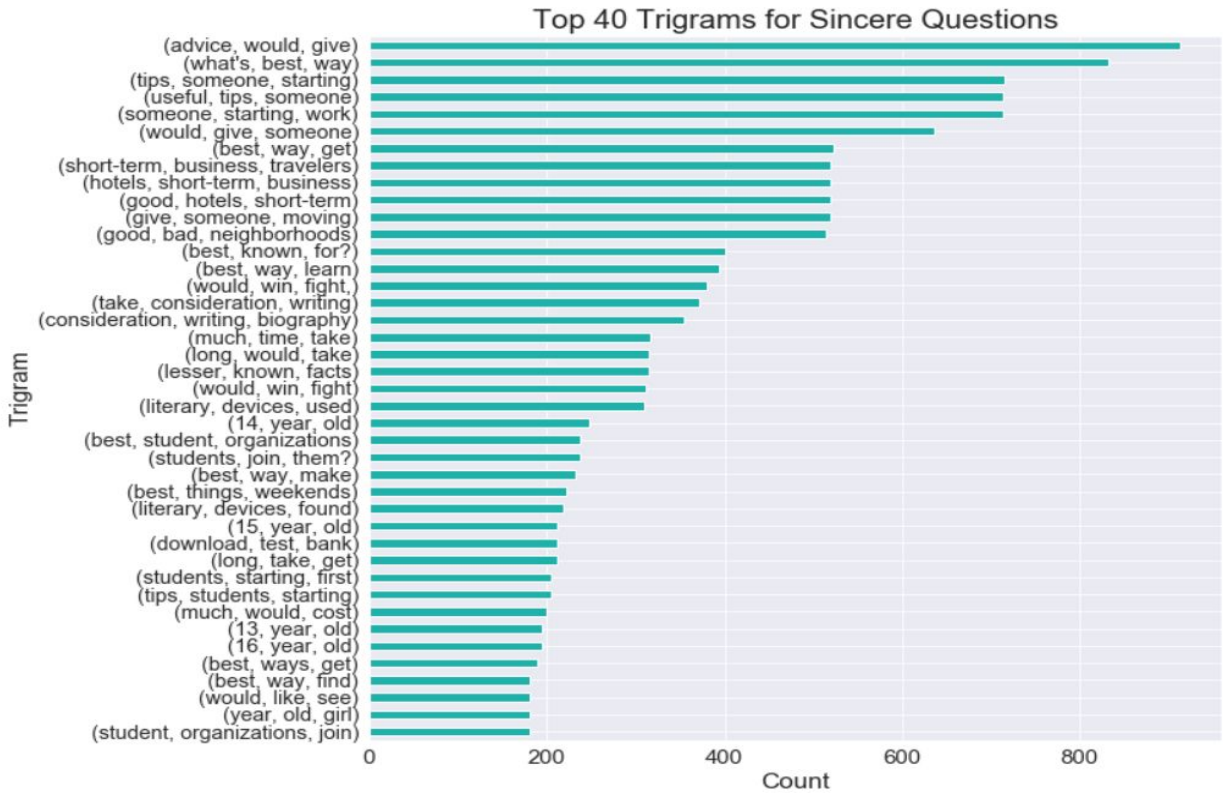
- Given that questions on Quora can be written anonymously, a grouping of the most frequent topics in anonymously proposed insincere questions can illustrate what some of our deepest and/or darkest thoughts are. This demonstrates that anonymity is a key factor to confession since we are often hesitant towards admitting dishonorable thoughts.



Donald Trump, Black people, white people,Indian Muslims are among the most targeted people/Community for insincere comments




Top 40 Trigrams for Insincere Questions
Trigram
(black, lives, matter)
(kim, jong, un)
(12, year, old)
(14, year, old)
(people, still, believe)
(think, donald, trump)
(united, states, america)
(ask, stupid, questions)
(gun, control, advocates)
(13, year, old)
(indian, girls, like)
(year, old, girl)
(year, old, son)
(president, united, states)
(many, people, quora)
(people, ask, stupid)
(president, donald, trump)
(white, people, think)
(nobel, peace, prize)
(president, united, states?)
(hate, white, people?)
(barack, hussein, obama)
(social, justice, warriors)
(donald, j., trump)
(long, would, take)
(causing, black, death?)
(hate, black, people?)
(think, black, people)
(people, united, states)
(many, stupid, questions)
(people, ask, questions)
(many, white, people)
(year, old, boy)
(many, people, think)
(scientific, procedure, castration?)
(north, indian, girls)
(year, old, daughter)
(new, york, city)
(like, donald, trump)
(what's, best, way)
0
5
10
15
20
25
30
35
40
Count


Top 40 Trigrams for Sincere Questions
Trigram
(advice, would, give)
(what's, best, way)
(tips, someone, starting)
(useful, tips, someone)
(someone, starting, work)
(would, give, someone)
(best, way, get)
(short-term, business, travelers)
(hotels, short-term, business)
(good, hotels, short-term)
(give, someone, moving)
(good, bad, neighborhoods)
(best, known, for?)
(best, way, learn)
(would, win, fight,)
(take, consideration, writing)
(consideration, writing, biography)
(much, time, take)
(long, would, take)
(lesser, known, facts)
(would, win, fight)
(literary, devices, used)
(14, year, old)
(best, student, organizations)
(students, join, them?)
(best, way, make)
(best, things, weekends)
(literary, devices, found)
(15, year, old)
(download, test, bank)
(long, take, get)
(students, starting, first)
(tips, students, starting)
(much, would, cost)
(13, year, old)
(16, year, old)
(best, ways, get)
(best, way, find)
(would, like, see)
(year, old, girl)
(student, organizations, join)
0
200
400
600
800
Count

Pre-Processing:

Above analysis of data suggests that Questions contain contraction and should be cleaned. Also it has many references to other languages other than English, so need to remove those words for better results. The pre trained stopword dictionary is not the best approach as well as all stopwords not be removed. It may harm the actual meaning of the sentence.

Text can come in a variety of forms from a list of individual words, to sentences to multiple paragraphs with special characters (like tweets for example). Like any data science problem, understand the questions that are being asked will inform what steps may be employed to transform words into numerical features that work with machine learning algorithms.
Data preprocessing consists of a number of steps, any number of which may or may not apply to a given task, but generally fall under the broad categories of:

- **Tokenization** : Tokenization is a step which splits longer strings of text into smaller pieces, or tokens. Larger chunks of text can be tokenized into sentences, sentences can be tokenized into words, etc. Further processing is generally performed after a piece of text has been appropriately tokenized. Tokenization is also referred to as text segmentation or lexical analysis. Sometimes segmentation is used to refer to the breakdown of a large chunk of text into pieces larger than words (e.g. paragraphs or sentences), while tokenization is reserved for the breakdown process which results exclusively in words.
- **Normalization:**

  Before further processing, text needs to be normalized. Normalization generally refers to a series of related tasks meant to put all text on a level playing field: converting all text to the same case (upper or lower), removing punctuation,

converting numbers to their word equivalents, and so on. Normalization puts all words on equal footing, and allows processing to proceed uniformly.

- **Replace contractions :**

  This replace all contractions with the full meaning. (Contractions scrapped from Wikipedia: https://en.wikipedia.org/wiki/Wikipedia:List_of_English_contractions)

  Contractions are shortened version of words or syllables. They often exist in either written or spoken forms in the English language. These shortened versions or contractions of words are created by removing specific letters and sounds. In case of English contractions, they are often created by removing one of the vowels from the word. Examples would be, ***do not*** to ***don't*** and ***I would*** to ***I'd***. Converting each contraction to its expanded, original form helps with text standardization.

- **Stemming/Lemmatization:**

  Stemming and lemmatization are text normalization techniques that help reduce the number of features generated by reducing the inflectional forms of each word into a common base or root. Stemming and lemmatization differ in that stemming cuts off the end or the beginning of the word, while lemmatization takes into consideration the morphological analysis of the words. For example, 'studies' and 'studying' would stem to 'studi' and 'study' (respectively), but would lemmatize to 'study'

- **Lowering Case:**

This avoids having multiple copies of the same words. For example, while calculating the word count, 'Analytics' and 'analytics' will be taken as different words.

- **Remove Numbers:**

  Replacing all number with the string '#'. This step was done because the total vocabulary of numbers alone was too large and most of them are non-meaningful.

- **Remove Punctuation:**

  Punctuation doesn't add any extra information while treating text data. Therefore removing all instances of it will help us reduce the size of the training data.

- **Spelling Correction:**

  Spelling correction is a useful pre-processing step because this will also help us in reducing multiple copies of words. For example, "*Analytics*" and "*analytcs*" will be treated as different words even if they are used in the same sense. Common misspellings scrapped from oxford dictionaries: https://en.oxforddictionaries.com/spelling/common-misspellings)

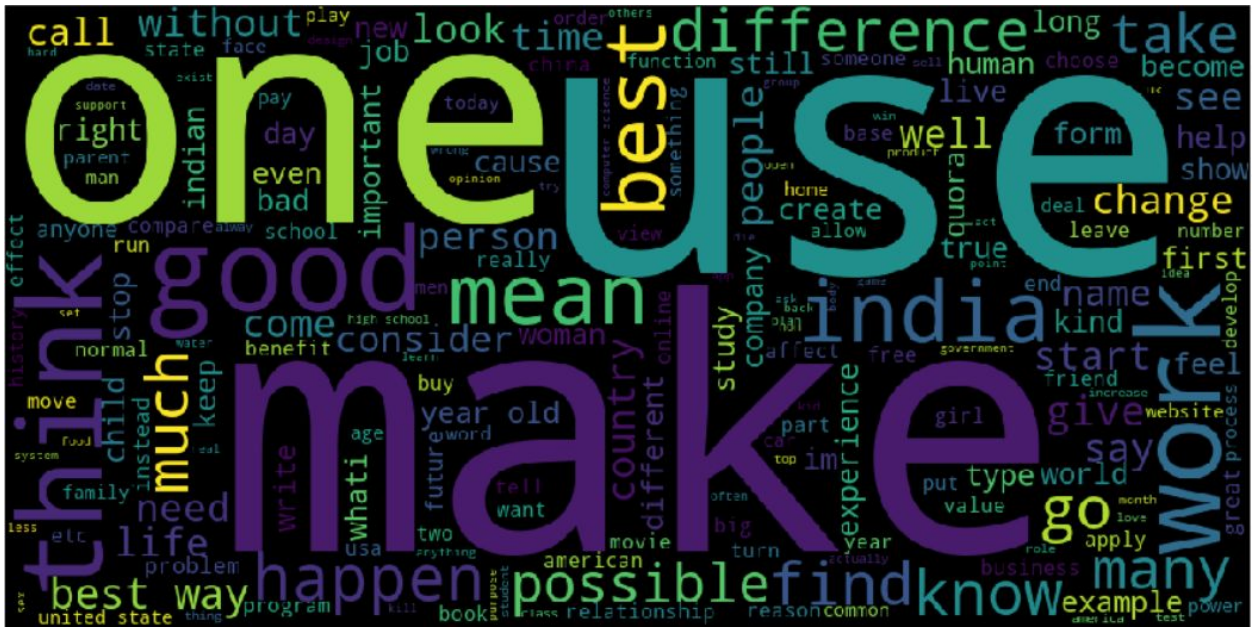
- **Strip white space**
- **Remove Stop Words:**

  Stop words (or commonly occurring words) should be removed from the text data. For this purpose, we can either create a list of stopwords ourselves or we can use predefined libraries. We have to be careful while removing the Stopwords as they can change the meaning of the sentence as well.

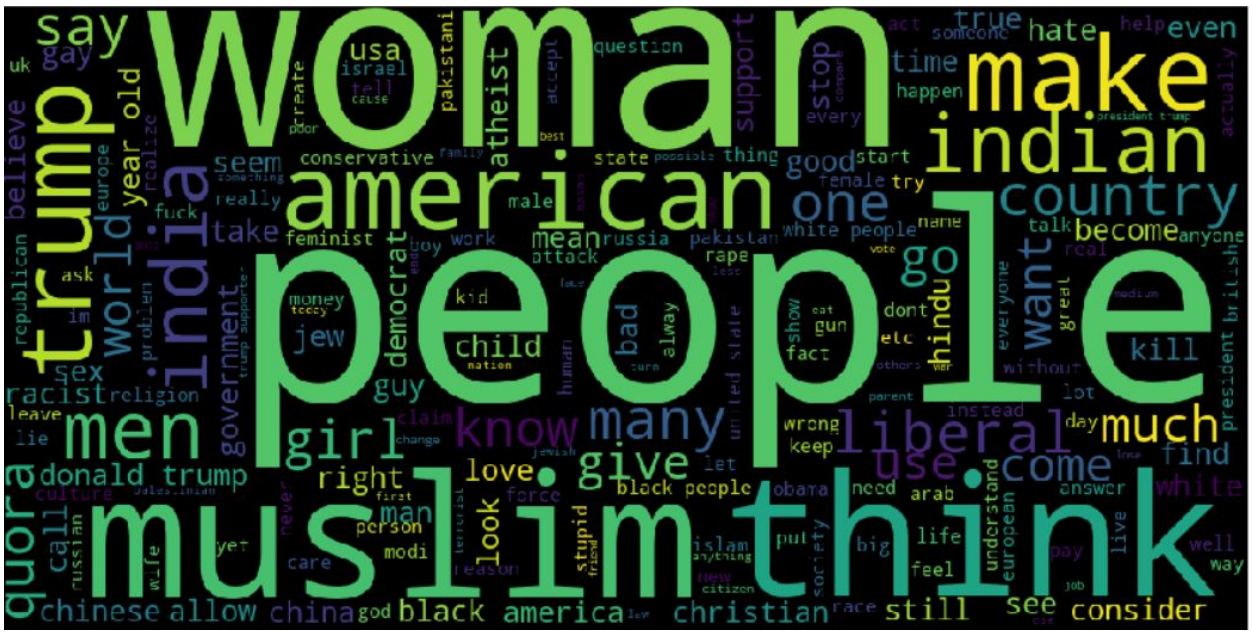
- **Remove 25 commonly occurring words and 25 most rare words from corpus**

WordCloud After Data Cleaning:

World Cloud of Sincere Questions:



.World Cloud of Sincere Questions:



Vectorization

The process of converting NLP text into numbers is called **vectorization** in ML.
Different ways to convert text into vectors are:

- Counting the number of times each word appears in a document.
- Calculating the frequency that each word appears in a document out of all the words in the document.

CountVectorization:

The CountVectorizer provides a simple way to both tokenize a collection of text documents and build a vocabulary of known words, but also to encode new documents using that vocabulary.
You can use it as follows:

1. Create an instance of the CountVectorizer class.
2. Call the fit() function in order to learn a vocabulary from one or more documents.
3. Call the transform() function on one or more documents as needed to encode each as a vector.

The main parameters of CountVectorizer are below which can be tuned for better results:

1. **Ngram_range:**
   An n-gram is just a string of n words in a row. E.g. the sentence 'I am Groot' contains the 2-grams 'I am' and 'am Groot'. The sentence is itself a 3-gram. Set the parameter ngram_range=(a,b) where a is the minimum and b is the maximum size of ngrams you want to include in your features. The default ngram_range is (1,1).
2. **min_df, max_df:**
   These are the minimum and maximum document frequencies words/n-grams must have to be used as features. If either of these parameters are set to integers, they will be used as bounds on the number of documents each feature must be in to be considered as a feature. If either is set to a float, that number will be interpreted as a frequency rather than a numerical limit. min_df defaults to 1 (int) and max_df defaults to 1.0 (float)**.**
3. **max_features:**
   The CountVectorizer will choose the words/features that occur most frequently to be in its' vocabulary and drop everything else.

TFIDF:

TF-IDF stands for term frequency-inverse document frequency. TF-IDF weight is a statistical measure used to evaluate how important a word is to a document in a collection or corpus. The importance increases proportionally to the number of times a word appears in the document but is offset by the frequency of the word in the corpus.

Term Frequency (TF): is a scoring of the frequency of the word in the current document. Since every document is different in length, it is possible that a term would appear much more times in long documents than shorter ones. The term frequency is often divided by the document length to normalize**.**

$$TF(t) = \frac{Number\ of\ times\ term\ t\ appears\ in\ a\ document}{Total\ number\ of\ terms\ in\ the\ document}$$

Inverse Document Frequency (IDF): is a scoring of how rare the word is across documents. IDF is a measure of how rare a term is. Rarer the term, more is the IDF score.

$$IDF(t) = log_e(\frac{Total\ number\ of\ documents}{Number\ of\ documents\ with\ term\ t\ in\ it})$$

Thus, TFIDF score is :

$$TF - IDF score = TF * IDF$$

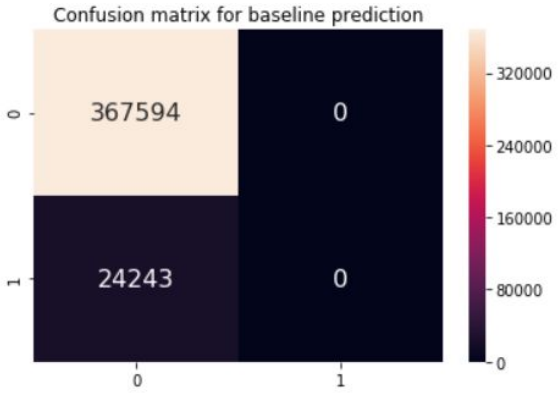
Model Evaluation

As I have iterated and decided before, for my classification task I will be focussing on Naive Bayes, Logistic Regression and Convolution Neural Network. Our goal is to achieve the best F1 score, so our models are compared based on their F score. There is a limitation to the number of submits to the Kaggle to receive the F-Score for the test set, I evaluated my model based on F-scores predicting the validation set. I assumed that model performing little better on validation dataset also do good on Test dataset as well.

Baseline Model:

The counter intuitive model for getting the best score is to label all of them as sincere question and see its F1-Score. The accuracy will be almost 94% but the F1-score, the harmonic mean will be 0 as the True Positives is 0

	precision	recall	f1-score	support
0	0.94	1.00	0.97	367594
1	0.00	0.00	0.00	24243
micro avg	0.94	0.94	0.94	391837
macro avg	0.47	0.50	0.48	391837
weighted avg	0.88	0.94	0.91	391837



CountVectorizer & Logistic Regression Classifier:

Logistic Regression is one of the simplest classification models in terms of fitting and analysis. I did cross validation of the train dataset by vectorizing it using CountVectorizer and used Logistic Regression as classifier.

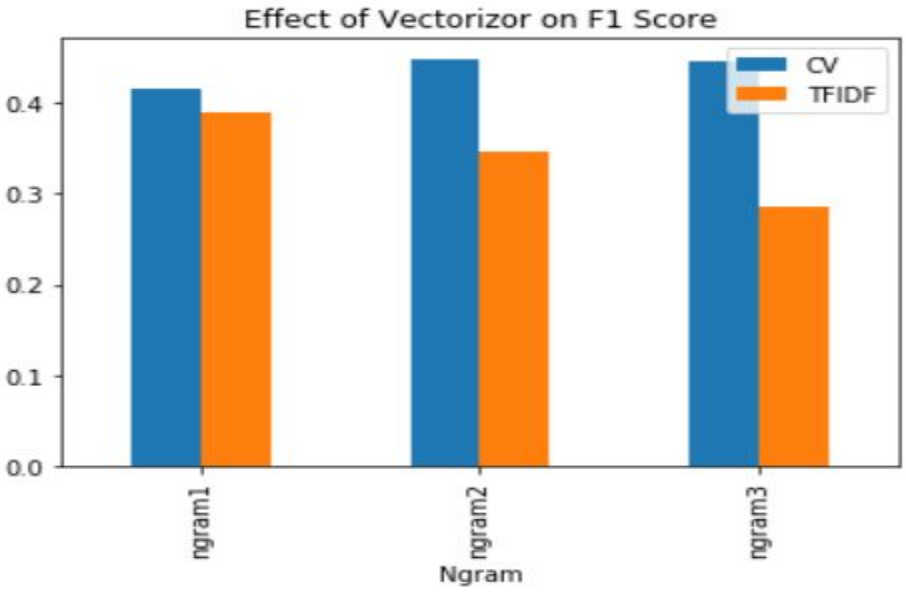
Ngram	Accuracy	Precision	Recall	F_Score	ROC AUC
1,1	0.946	0.642	0.308	0.416	0.914
1,2	0.948	0.651	0.343	0.449	0.917
1,3	0.948	0.657	0.338	0.446	0.916

TFIDF & Logistic Regression Classifier

I did cross validation of the train dataset by vectorizing it using TFIDF and used Logistic Regression as classifier. Below are the results that I achieved

Ngram	Accuracy	Precision	Recall	F_Score	ROC AUC
1,1	0.946	0.648	0.279	0.390	0.919
1,2	0.945	0.69	0.233	0.346	0.922
1,3	0.943	0.648	0.182	0.284	0.918

Comparison Traditional Classifiers:



CountVectorizer is giving better results than TFIDF vectorizor. It gives best score when Ngram is (1,3).

Train-Test Split:

Before fitting classification models, I split Train dataset into two unequal parts, namely training and validation sets. The training set was randomly chosen from the dataset and contains 70% of the original set, while the test contains the rest 30%.

Data Pipeline:

Pipelines are a way to streamline a lot of the routine processes, encapsulating little pieces of logic into one function call, which makes it easier to actually do modeling instead just writing a bunch of code. Pipelines allow for experiments, and for a dataset like this that only has the text as a feature, you're going to need to do a lot of experiments. Plus, when your modeling gets really complicated, it's sometimes hard to see if you have any data leakage hiding somewhere. Pipelines are set up with the fit/transform/predict functionality, so you can fit a whole pipeline to the training data and transform to the test data, without having to do it individually for each thing you do.

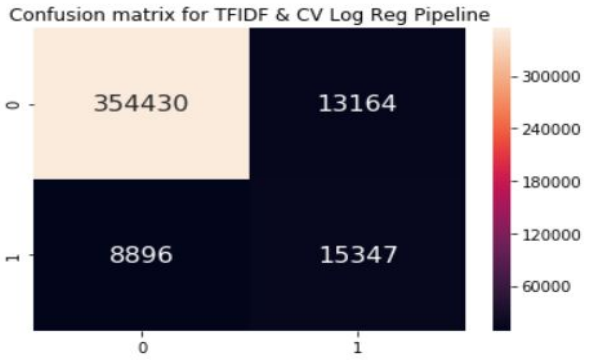
Advantages of pipelines

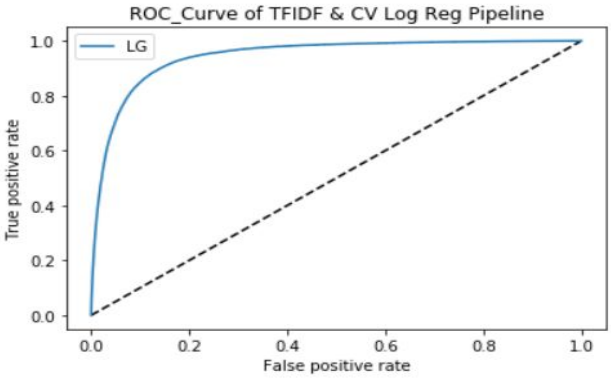
- Use of pipelines gives you a kind of meta-language to describe your model and abstract from some implementation details.
- With pipelines, you don't need to carry test dataset transformation along with your train features - this is taken care of automatically.
- Hyperparameter tuning made easy - set new parameters on any estimator in the pipeline, and refit - in 1 line. Or use GridSearchCV on the pipeline.

TFIDF & CountVectorizer Logistic Regression model

In this pipeline, input data comes first to CountVectorizer, which creates a sparse matrix of word counts in each sentence. This matrix then serves as input to TfidfTransformer which massages the data and handles it to the LogisticRegression estimator for training and prediction.

Accuracy	Precision	Recall	F_Score	Threshold
0.950	0.538	0.633	0.582	0.237



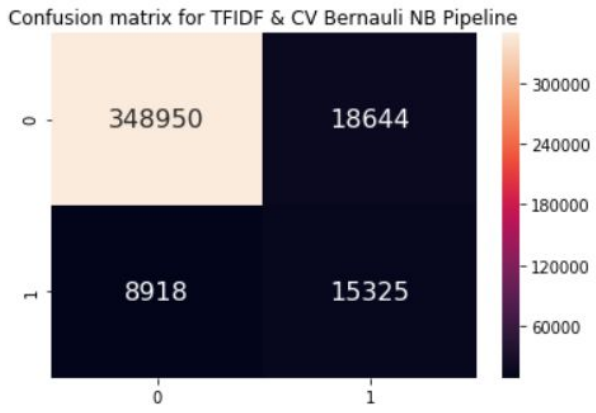


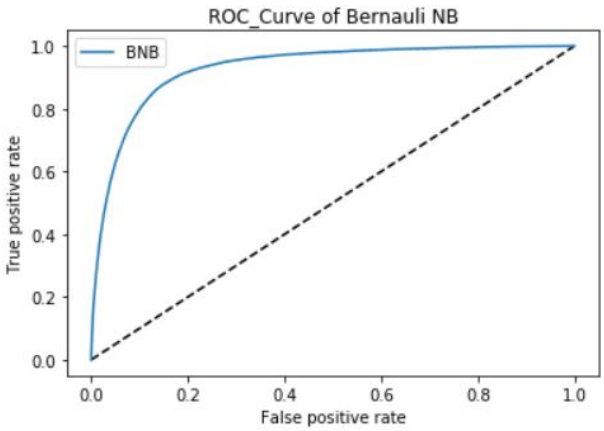
TFIDF & CountVectorizer Naive Bayes model

Bernoulli Naive Bayes classifier is a relatively simple algorithm. It assumes features to describe the data points are independent and boolean. In this case, the features would be words, the instances are the questions and the boolean determines whether a word in the corpus is present in the question.

The result of this model is below :

Accuracy	Precision	Recall	F_Score	Threshold
0.929	0.451	0.632	0.526	0.150

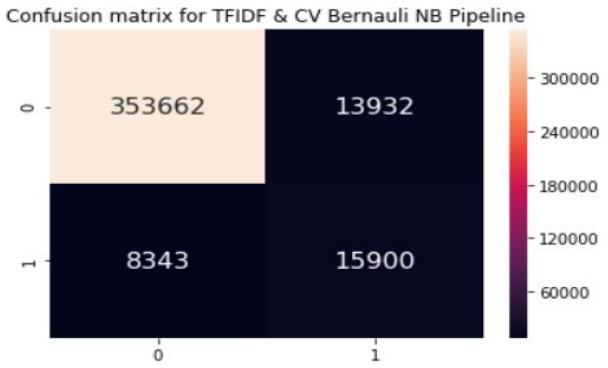


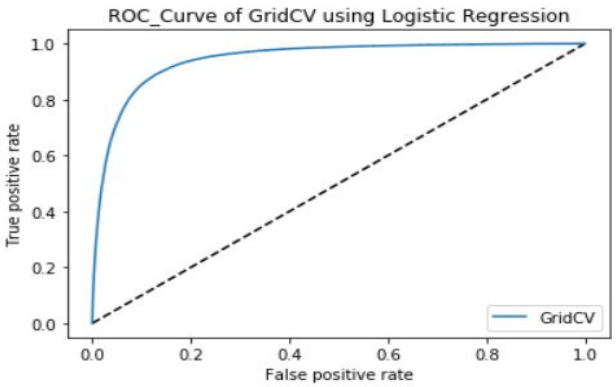


Classifier	Accuracy	Precision	Recall	F_Score	Threshold
Log Reg	0.950	0.538	0.633	0.582	0.237
Naive Bayes	0.929	0.451	0.632	0.526	0.150

Logistic Regression has outplayed the Naive Bayes classifier. One thing to be noted is using both the vectorizer has considerably increased the F_Score of our prediction.
So for further study, I used Logistic Regression. I did Hyperparameter tuning using data pipeline, as its very easy to manipulate each parameter using Pipeline

Classifier	Accuracy	Precision	Recall	F_Score	Threshold
Log Reg Hyperparameter tuning	0.943	0.533	0.656	0.588	0.233





Within each model, I conducted an exhaustive search for the classification threshold value that would optimize model performance. A probability score below the classification threshold was classified as sincere, while a probability score above the threshold was classified as insincere. Adjusting threshold values from the default 0.5 greatly improved F-scores. Threshold values that maximized F-scores varied between models.

Convolution Neural Network

Convolutional neural networks are deep artificial neural networks that are used primarily to classify images, cluster them by similarity, and perform object recognition within scenes. They are algorithms that can identify faces, individuals, street signs, tumors, platypuses and many other aspects of visual data. Convolutional Neural Networks have a different architecture than regular Neural Networks. Convolution is one of the main building blocks of a CNN. The term convolution refers to the mathematical combination of two functions to produce a third function. It merges two sets of information. In the case of a CNN, the convolution is performed on the input data with the use of a filter or kernel to then produce a feature map. Convolution is performed by sliding the filter over the input. At every location, a matrix multiplication is performed and sums the result onto the feature map. After a convolution layer, pooling layer is added to continuously reduce the dimensionality to reduce the number of parameters and computation in the network. This shortens the training time and controls overfitting

Feature Extraction - Word Embeddings:

After optimizing the performance using traditional text classification algorithms and feature extraction techniques, I set about exploring the potential benefits of using distributed word representations, commonly referred to as embeddings. In a typical tokenized document representation, a given word w is mapped to a single integer i, and its presence or absence in a document is denoted by a 1 or 0 in the document vector at index i. The main problem with this format is that it gives no indication of word relationships or semantics. Other than looking for correlations of features across the corpus of documents, the model has no way of inferring whether two features represent synonyms, antonyms, or words with some more nuanced relationship such as superlatives.

Original Word Coverage:

In this competition we have 3 types of embeddings i.e. Glove , Paragram and Google.I have checked the Out of Vocabulary for all 3 embeddings and found Glove has the best word Coverage and least out of vocabulary (OOV) rate. So for CNN classifier, I have used Glove Embedding to get the best result .

Paragram

```
Found embeddings for 87.63% of vocab
Found embeddings for  99.89% of all text
 paragram oov rate: 0.12073093887838689
```

Glove

```
Found embeddings for 87.93% of vocab
Found embeddings for  99.90% of all text
   glove oov rate: 0.12371350556605755
```

Google

```
Found embeddings for 80.53% of vocab
Found embeddings for  99.79% of all text
   google oov rate: 0.194664986347406
```

All the words from all the questions are then label encoded by fitting on the vocabulary and then transforming every question separately. (Of course before this is done all the questions are concatenated, from both training and testing datasets, into one string that contains the whole vocabulary of words before encoding). This step is necessary because the embedding layer requires the input data to be integer encoded. In this step, the max length of words is set to 70 for all questions.

After doing all pre processing, now we need to create the embedding matrix. It was created as follows:

- First the maximum number of words (40000) to be considered is set
- The embedding matrix will contain the vectors of these words selected from the embedding file if they exist, and empty vectors if not, both of size 300.
- So the final embedding matrix size will be 50,000 by 300. This step reduces the dimensionality space, especially when comparing with one-hot encoding.

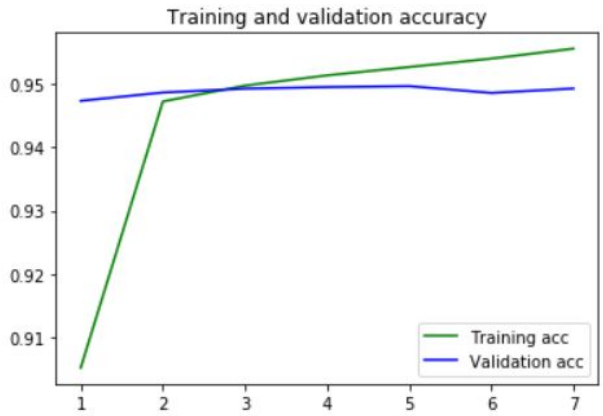
Implementation Of CNN

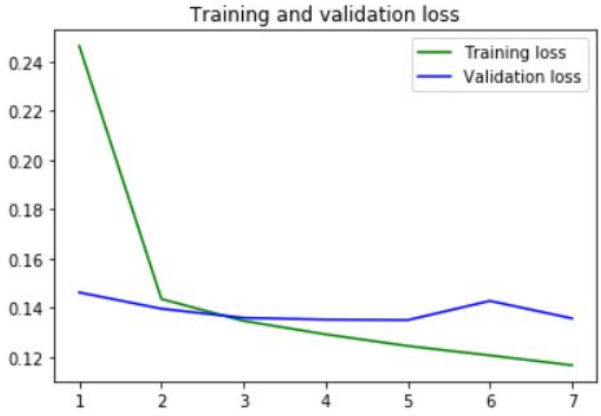
The architecture of the network is built as follows:

- Input layer equal to the max length set (70).

- Embedding layer. For this layer, at least three arguments must be specified: the size of the vocabulary (50,000), the size of the embedding vector (300), and the input length already set to 70. In addition, since the embedding vectors are pre-trained weights representation, this layer's weights are set equal to the embedding matrix weights. These weights are also set to 'non-trainable'.
- The embedding layer contains 70 vectors of 300 dimensions each, one for each word. We could flatten this to a 21000 element vector to pass it to the dense output layer, or we could add more convolutional layers before doing so.
- Following the embedding layer are three 1D convolutional layers. The reason why these layers are 1D and not 2D is because we are working with text data and not images.
- And the last layer is a fully connected layer with one node (binary classification) and a 'sigmoid' activation function.
- The loss function, optimizer and evaluation metric defined for this network are 'binary_crossentropy', 'adam' and F1-score respectively.

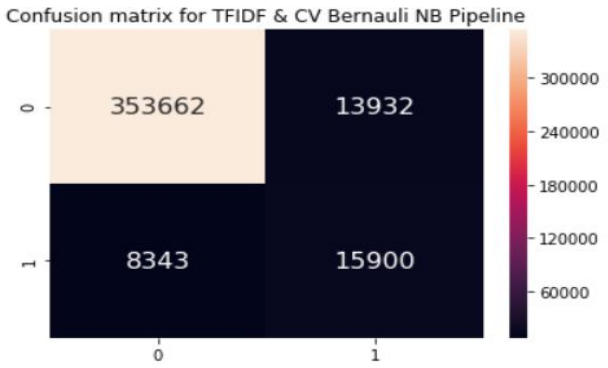
The CNN is a classifier with many tunable parameters. One of them is the number of epochs. And while more epochs can decrease the error and enhance the performance, too many epochs may cause the model to overfit the training data. Overfitting can be detected and visualized by plotting the training and validation score (or error) along with their corresponding number of epochs.

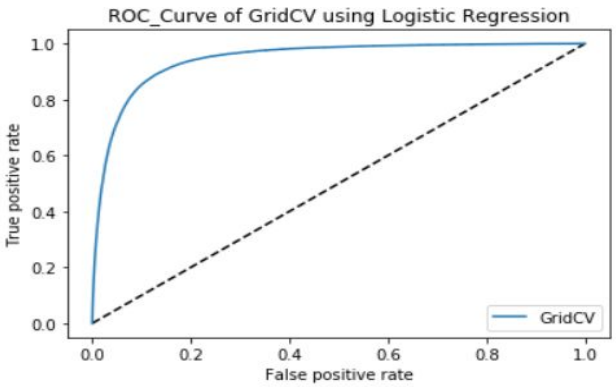




We can clearly see how the validation score starts to decrease while the training data is increasing as the model is trained with more epochs. This is a clear sign of overfitting. This issue however was handled by adding 'early stop' approach, where it will stop training as soon as the model starts overfitting, in this case, around epoch 8.

Let's explore the confusion matrix and other scores of CNN.





Classifier	Accuracy	Precision	Recall	F_Score	Threshold
CNN	0.943	0.633	0.656	0.602	0.230

Conclusion

As expected the CNN model did really well on the given dataset. So I chose CNN as my final classifier as it outperformed our traditional Naive Bayes and Logistic Regression models. The final f1 score for this model is a maximum of 0.629 on the validation set and 0.648 on the test set (dataset size equal to 56370) which is a good one especially since the dataset tested on has an uneven class distribution. This means that this model's predictions have a low percentage of 'False Positives' and 'False Negatives' and a high percentage of 'True Positives'. Thus we can conclude that this model can make predictions reliably. The CNN model is also capable of classifying reliably on unseen data with uneven class distribution.

Improvement

There is a huge room for improvement to all the classifiers to achieve better results. For our traditional model, we could have tune more hyperparameter, add more features like

length of sentence & number of stopwords or use few more classifiers like SVN or RandomForestClassifier.

Neural network classifiers can also be improved by expanding the embedding matrix by using additional pre-trained embedding files.Also we can use Recurrent Neural Network (RNN) instead of CNN.Further work could attack the problem with an ensemble method incorporating the LDA to the Logistic Regression model or including it as another layer in the Neural Network. A future iteration of this project would explore how the other embeddings like word2vec or Paragram could improve our model, perhaps using a combination of the four.