Quora Insincere Questions Classification
Capstone 2 - Milestone Report

Main Objective

The dataset contains about 1.3 million quora questions with question id, test and classification (sincere vs insincere) provided. The aim is to build a classification model based on the provided dataset to flag the insincere questions. This would be helpful in detecting toxic and misleading content. The slide deck for overview of the project can be found here.

Business Value

Quora is a social platform where people can ask questions and connect with others who contribute unique insights and quality answers. And existential problem is detecting/handling toxic and divisive content. This project is a way to tackle the problem, where we will develop models that identify and flag insincere questions. Quora has a policy of "Be Nice, Be Respectful". This project is a way to combat the problem and uphold Quora's status as a place for sharing and growing the world's knowledge.

Dataset Details

Available at : https://www.kaggle.com/c/quora-insincere-questions-classification/data
Description Provided:
"*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*
- *Is disparaging or inflammatory*
- *Based on false information, or contains absurd assumptions*
- *Uses sexual content (incest, bestiality, pedophilia) for shock value, and not to seek genuine answers*

The training data includes the question that was asked, and whether it was identified as insincere (target = 1).

Data fields

- *qid - unique question identifier*
- *question_text - Quora question text*
- *target - a question labeled "insincere" has a value of 1, otherwise 0"*

Data Wrangling:

The data wrangling notebook is available at: Data Wrangling and Cleanup Notebook

The dataset comprised of ~1.3 million rows (questions) and 3 columns - the question ID, question text and the target variable (1-Insincere and 0-Sincere). The most vital information that we will need to solve the classification problem lies in the question text column. The first step would be to extract useful demographic information from the text. The features we extracted from the original data include:

- Length of the string
- Number of capital letters
- Ratio of number of capital letters to the length of the question
- Number of words used
- Number of unique words
- Ratio of unique words to total number of words
- Number of exclamation marks, question marks and other punctuations

- Number of symbols / special characters used

Once we had extracted the useful information, the next step was to process the text to prepare it for eventual analysis. We started by turning the text into lowercase and stripping any extra white spaces. The preprocessing includes steps to normalize the data. The step taken in order to achieve this:

- Removing accented characters
- Expanding contractions
- Removing special characters
- Lemmatizing the text to retain only the base words
- Removing stop words which add unnecessary noise

Finally, we were left with 14 columns with the previous demographic features we extracted from the original text column and the normalized text column. The processed text column had a few values which were null - this could be attributed to the normalization process- maybe all the words in the text were removed during the preprocessing. We just filled in the null values with an empty string so that we don't lose any data points.

Having the processed data with 1306122 rows and 14 columns, we can add in a few more demographic features comparing the numeric features before and after processing. The features we added were:

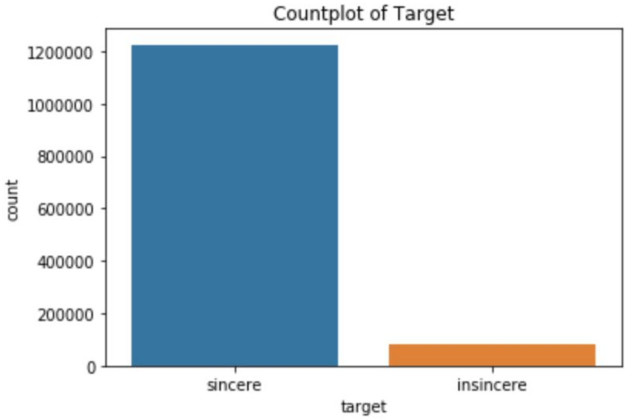
- Length of text after processing
- Number of words after processing
- Ratio of length before vs after
- Ratio of number of words before vs after

And once we have all the important features, we can drop the question id and original question text columns since those don't provide anymore useful information that can be used for further analysis.

Exploratory Data Analysis:

The EDA notebook is available at: Exploratory Data Analysis Notebook

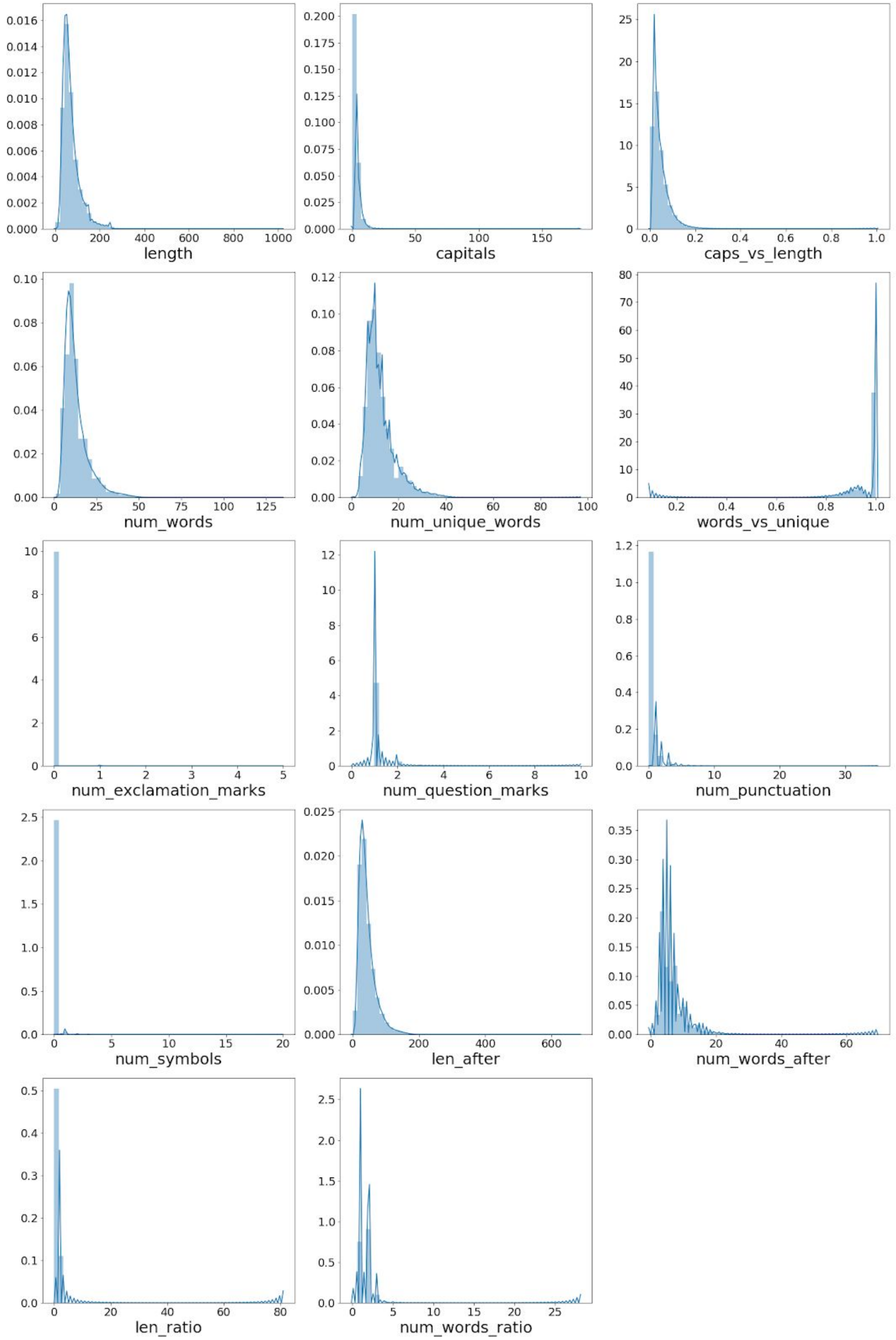
The most important feature for us to look at first is the target - which is 1 for insincere questions and 0 for sincere questions.

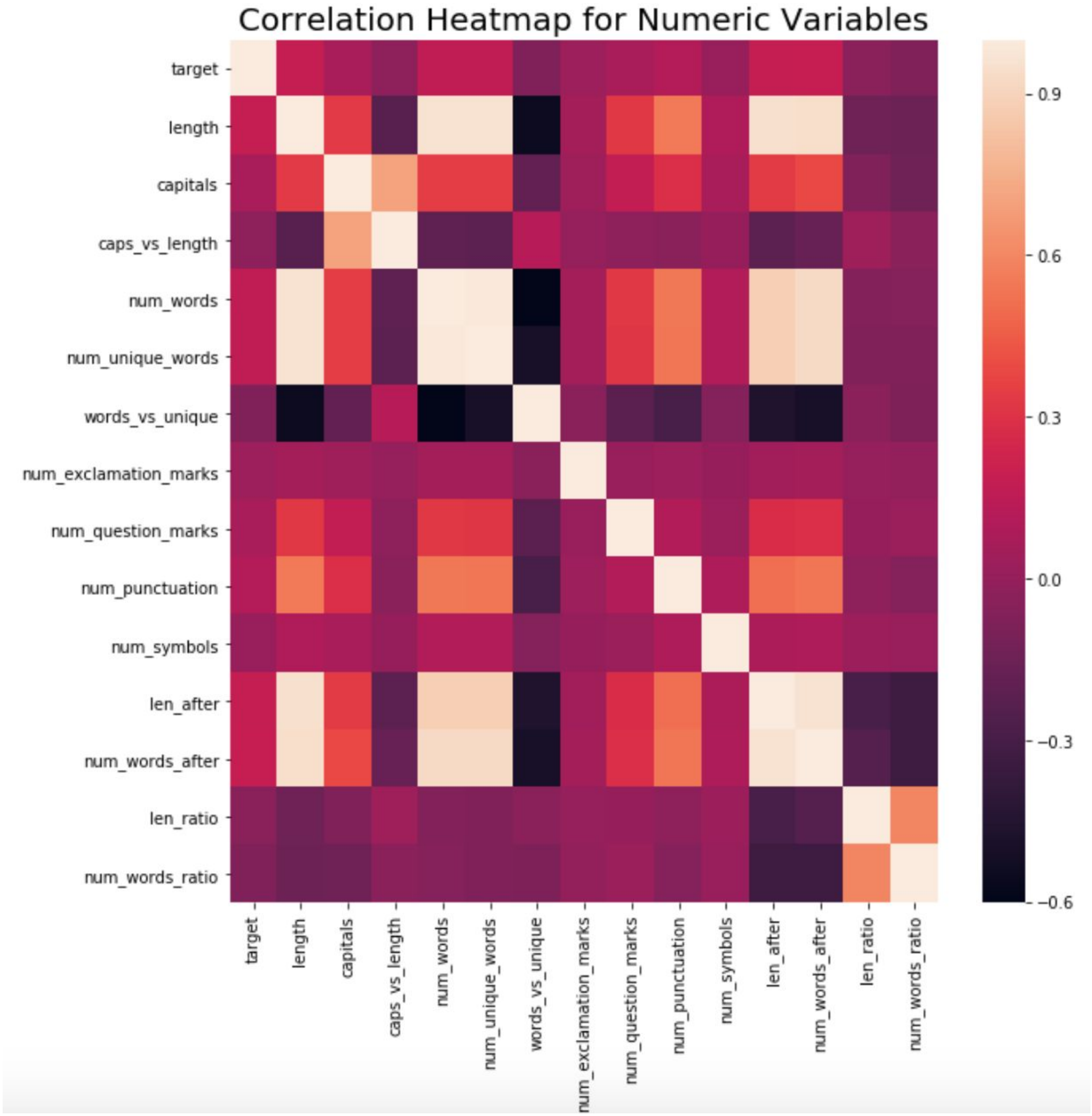


The above countplot shows that the dataset is pretty skewed. There are many sincere questions as compared to insincere questions. There are about 1200000 questions classified as sincere whereas less than 10000 are classified as insincere. This makes the classification problem a bit more complicated as the features defining insincere questions would be hard to spot when the data is so skewed.

Next, we look at the distribution of all other numeric features.

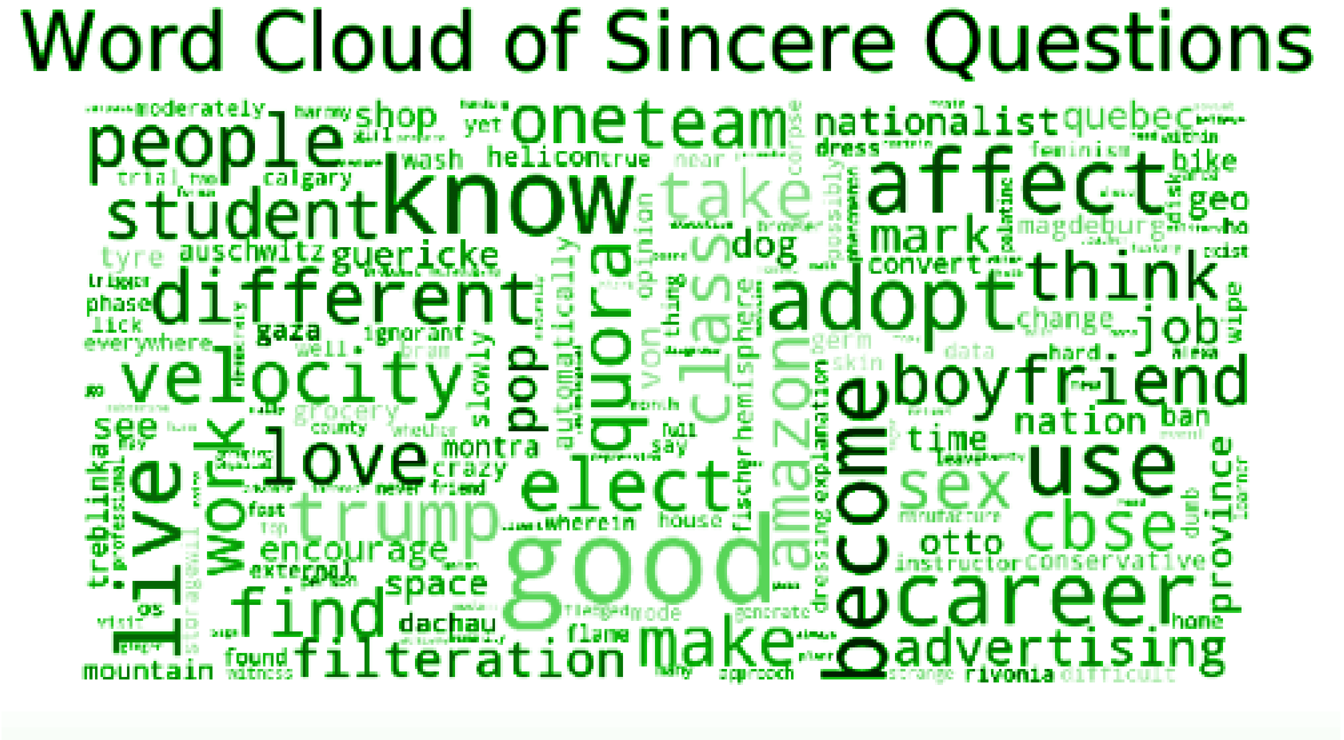
Distribution plot of Numeric features

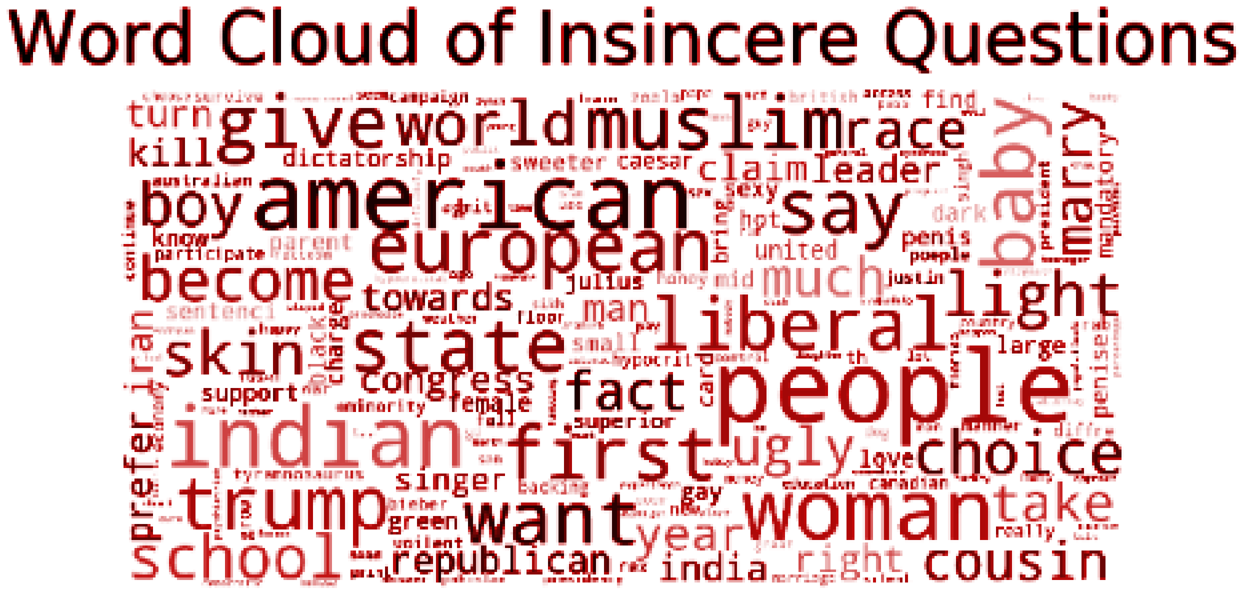




Based on the correlation heatmap, we can conclude that there is strong positive correlation between length related and number of words features. This makes sense intuitively, because longer the question, more words would be expected.

Moving on from the numerical features of the dataset, next focus was on the actual textual data. We started with plotting word cloud for most commonly found words in sincere questions vs in insincere questions.





These word clouds give great insight into the most commonly found words in insincere vs sincere questions. The common words in sincere questions are mostly positive / neutral words:

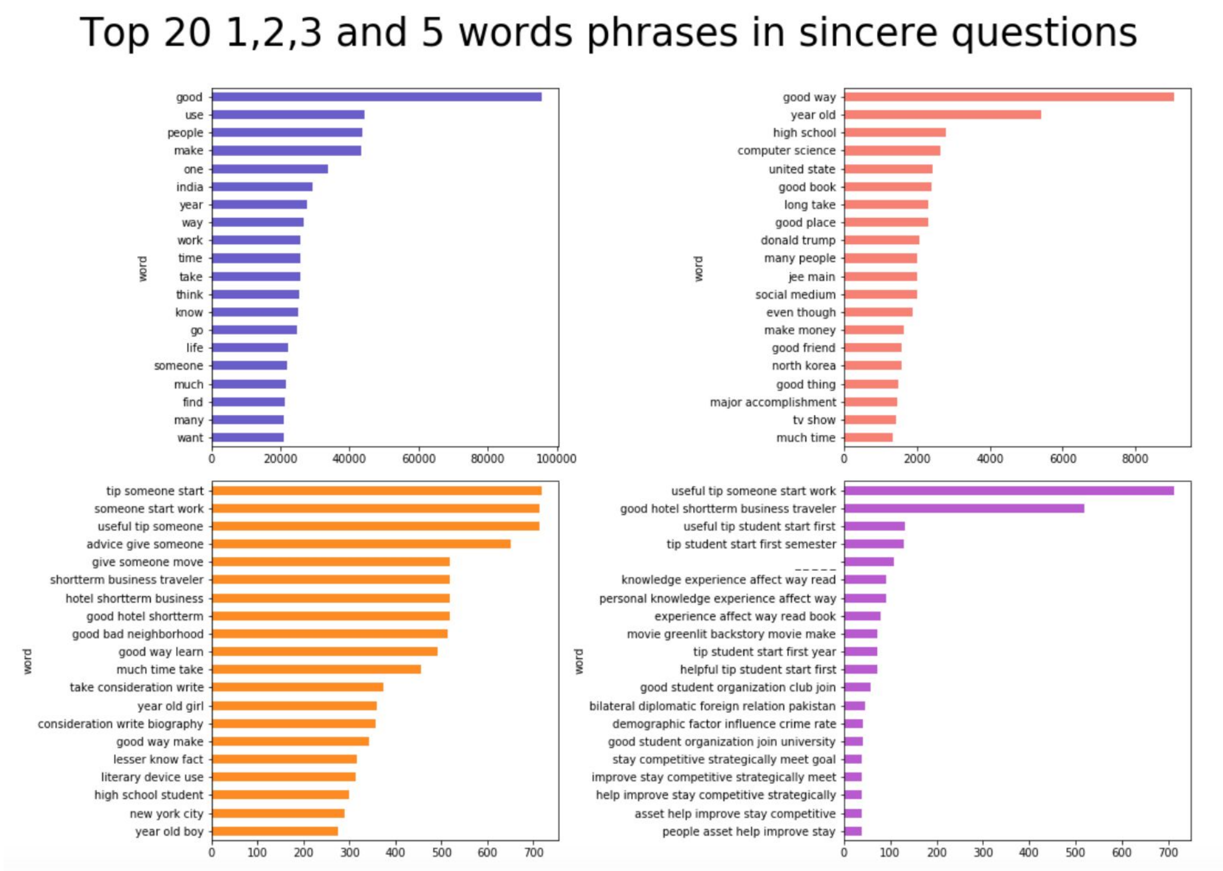
- good
- affect
- career
- love
- become

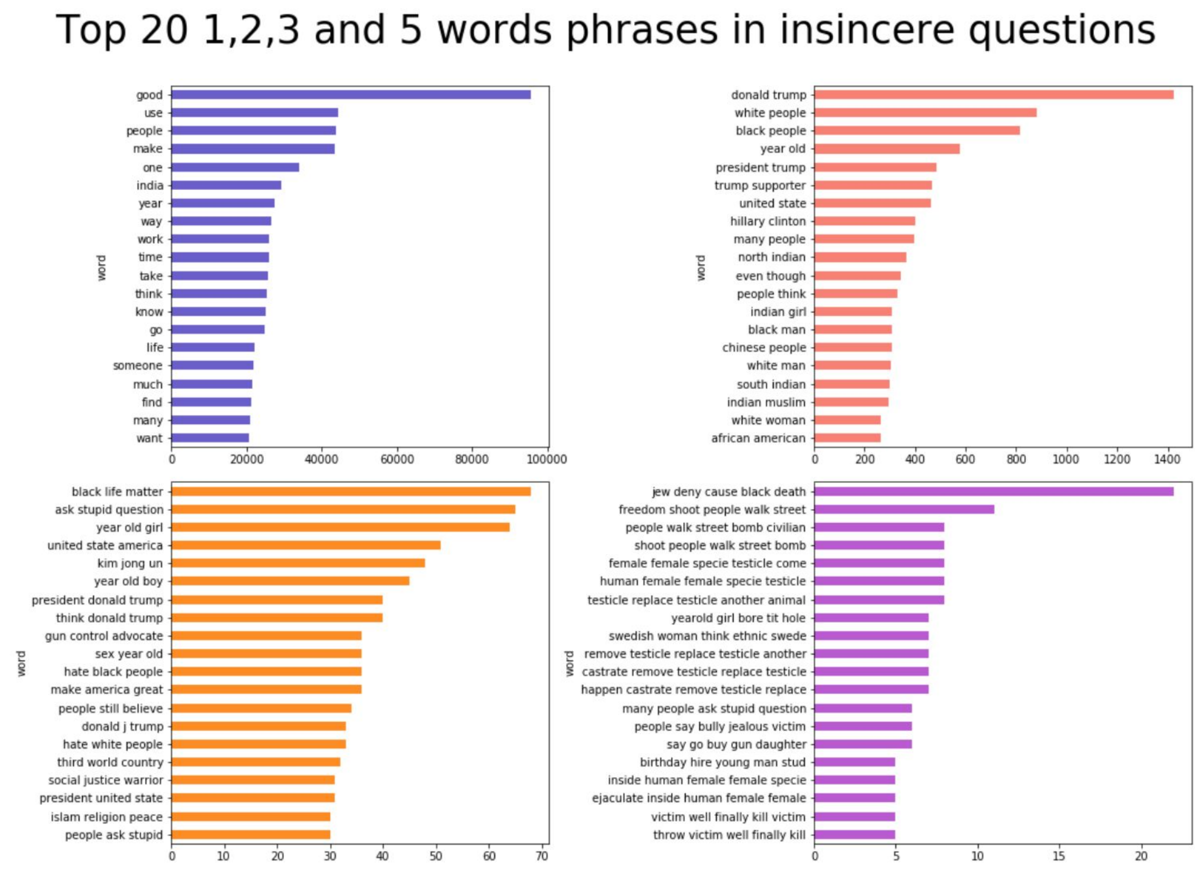
The common words in insincere questions are mostly negative / racial or provocative words:

- American
- European
- Muslim
- people
- woman

This helped us get insight into the theme and kinds of words used in each of the different category of question classification.

And further, we wanted to focus on phrases commonly found in each category of questions - for this purpose we focussed on 1,2,3 and 5 word phrases, determining the top 20 phrases in each category along with their frequencies and plotted them as below.

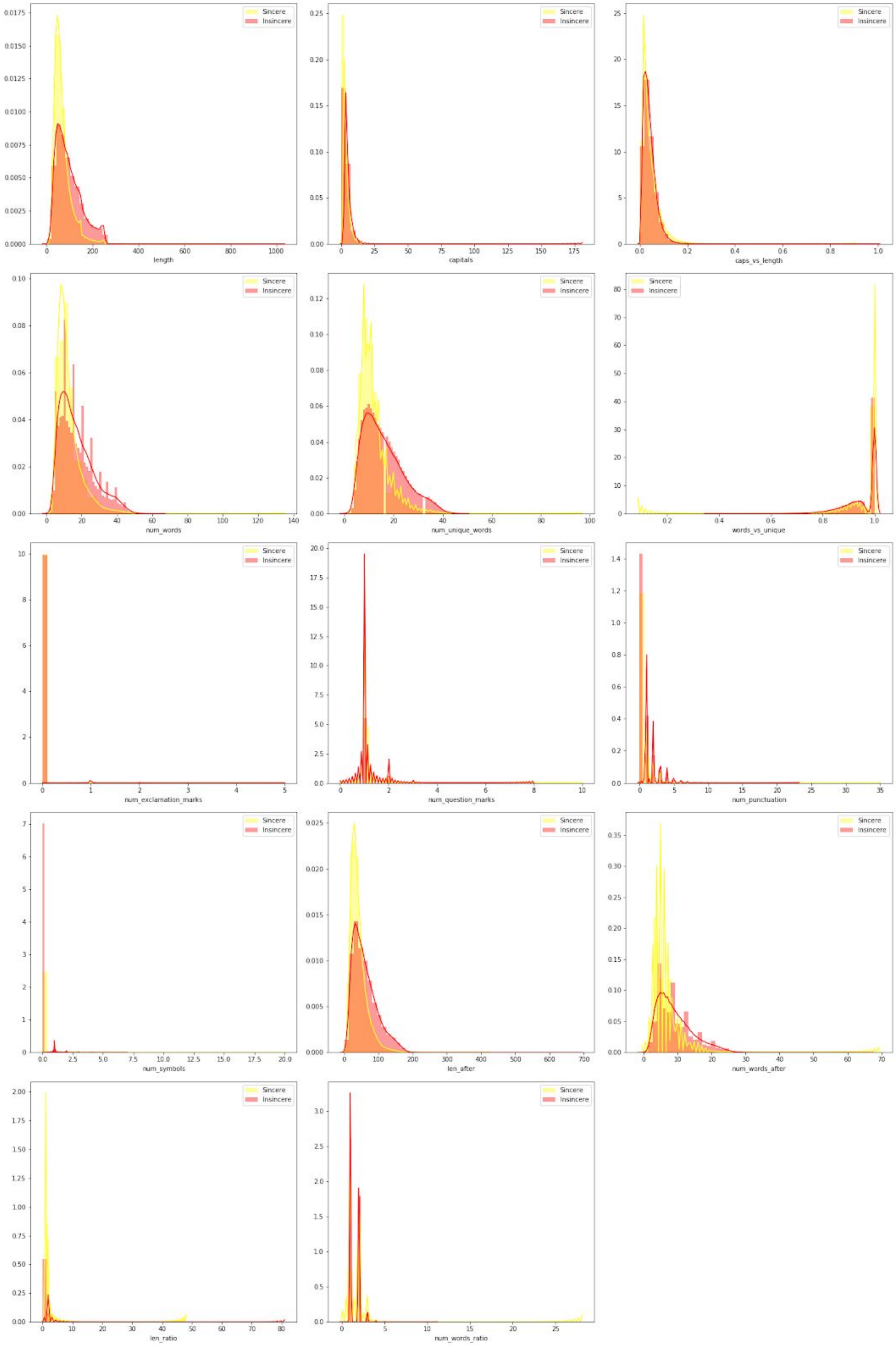




We start seeing some patterns in the topmost phrases in each of the category - sincere vs insincere. This is going to be useful for the further analysis.

Lastly, comparing the distribution of different numerical features in each category of questions separately would also be interesting. So, once the data was divided in two groups according to the target variable, we plotted distribution of each variable for both categories together to help us better visualize them and compare.

Distribution of numeric variables in Sincere vs Insincere questions



Having visualized the numerical and textual data, we have a better understanding of the different underlying trends in the data. We also learnt about the correlations between different variables. And also gained insight in the textual features - including the most commonly used words/phrases. With our better equipped knowledge about the data, we can move on to running statistical analysis.

Inferential Statistics:

Inferential statistics will help us dive deeper into the trends of the data and statistically draw conclusions. We performed different hypothesis testing on different features in the dataset to find statistical relevance. The alpha value fixed at 0.5 for all the tests and the p-values were calculated. The following table provides a summary of the tests performed and the outcomes of those.

Hypotheses	**Final Verdict**
H_0 → Mean number of words in sincere questions = Mean number of words in insincere questions H_a → Mean number of words in sincere questions ≠ Mean number of words in insincere questions	P value ~0. The null hypothesis was rejected, indicating that we have enough evidence to conclude that the number of words in sincere and insincere questions differ significantly, insincere questions seem to have more number of words than the sincere questions on average.
H_0 → Mean number of punctuations in sincere questions = Mean number of punctuations in insincere questions H_a → Mean number of punctuations in sincere questions ≠ Mean number of punctuations in insincere questions	P value ~0. The null hypothesis was rejected, indicating that we have enough evidence to conclude that the number of punctuations in sincere and insincere questions differ significantly, insincere questions seem to have more punctuations than the sincere questions on average.

In conclusion, we have a cleaned up dataset and we have analyzed the data visually as well as statistically. This helped us understand different underlying patterns in the data and uncover correlations. The data is ready for further feature engineering. We will be classifying the questions as sincere / insincere using different classification algorithms.

Machine Learning:

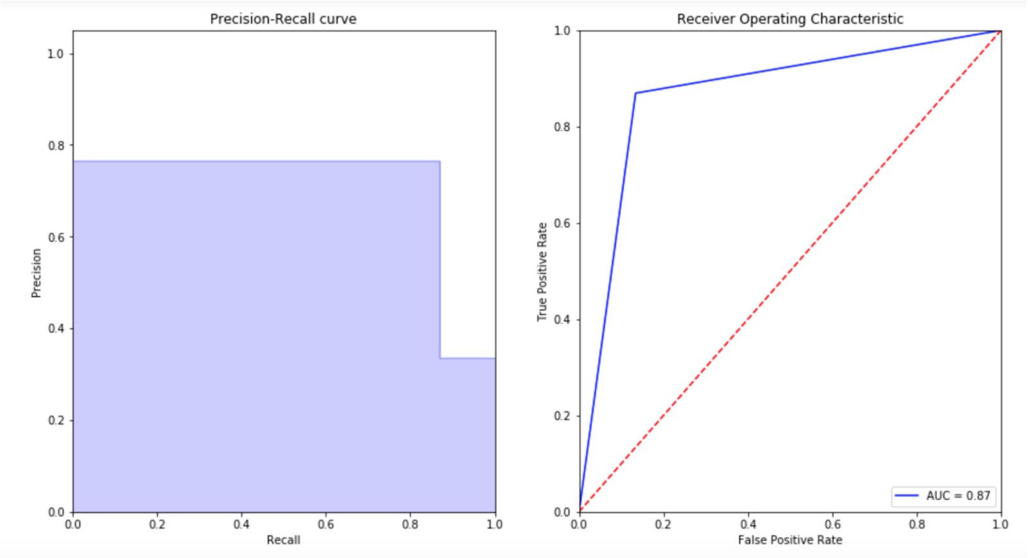
The Machine Learning + Deep Learning notebook is available at: ML+DL Notebook

Now that we had cleaned up our data and gained meaningful insights, we moved on to solving the actual classification problem. For this, we can use different machine learning classification algorithms and then compare them to see which performed the best for our given dataset.
But, one major problem is that we have ~1.3 million data points, and using the whole dataset would be very computationally expensive. Also, the dataset is very imbalance because we have a much higher proportion of sincere questions than insincere questions. To overcome these potential pitfalls, we can take a subset of the dataset. We will retain all the insincere question as the main problem is to be able to correctly classify them. We then randomly chose twice the amount of sincere questions. For the predictive analysis, we used a subset of the original dataset which had ~250k rows and the sincere vs insincere question ratio was 2:1.
To begin, we fit the classification models including:

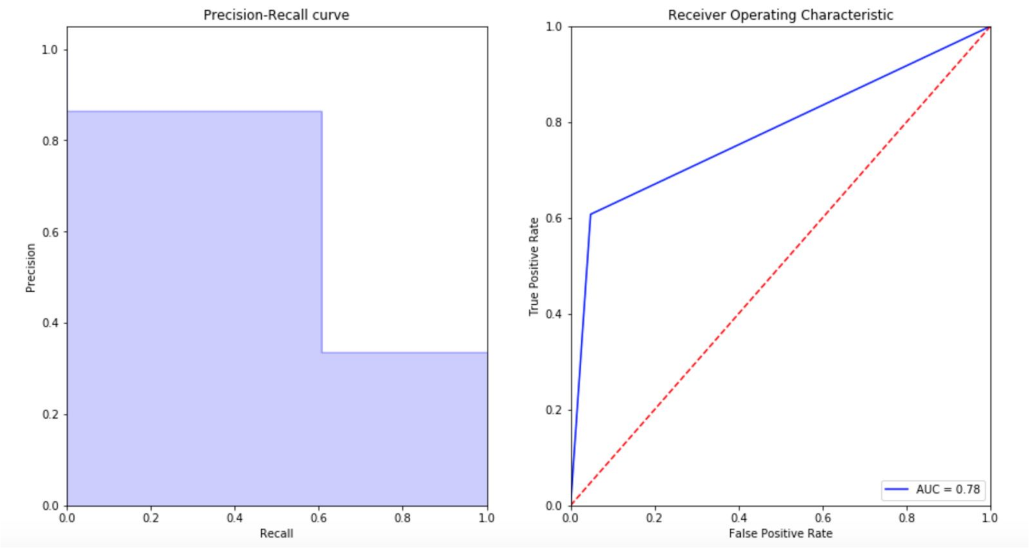
- Logistic Regression
- Naive Bayes
- KNN Classifier
- XG Boost Classifier

We used the countvectorizer and TF-IDF vectorizer to vectorize the text data. We compared the models using the precision vs recall curve and area under the ROC curve.

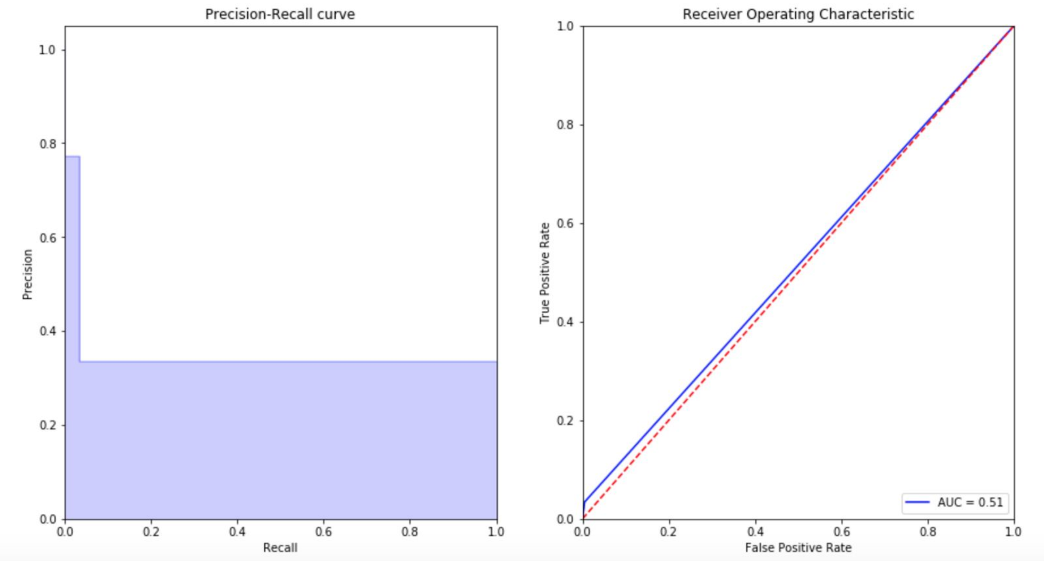
Logistic Regression



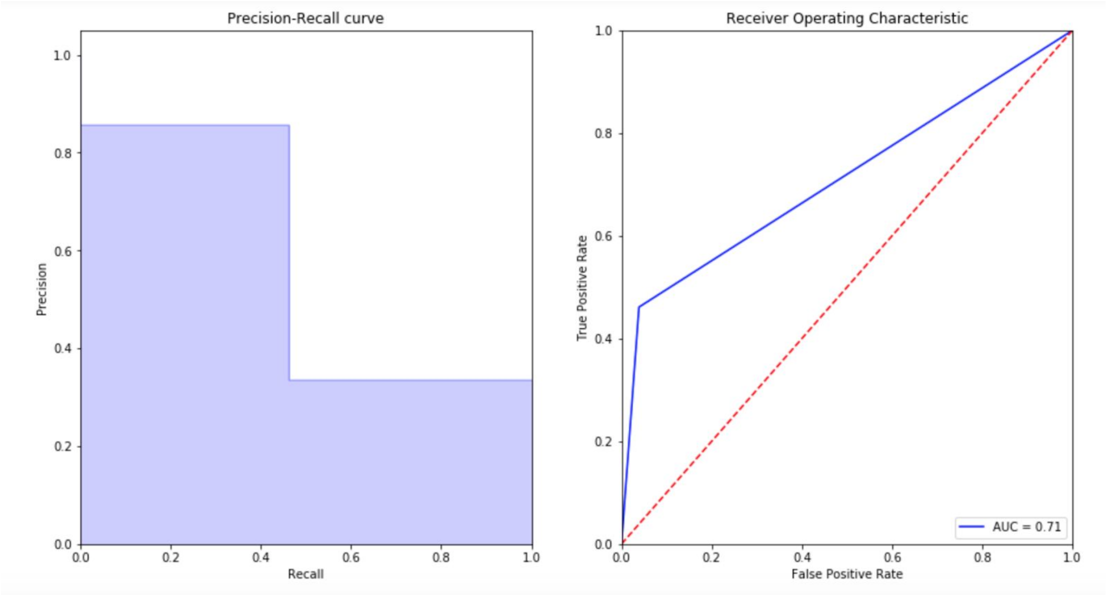
Naive Bayes



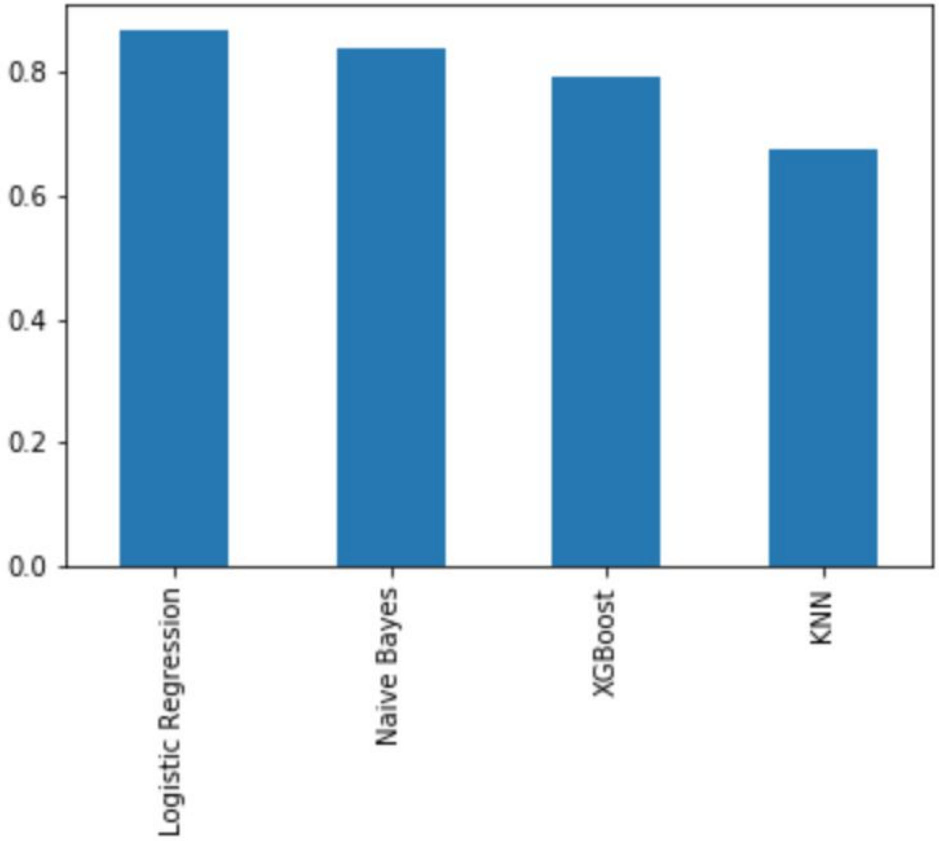
KNN Classifier



XGBoost Classifier

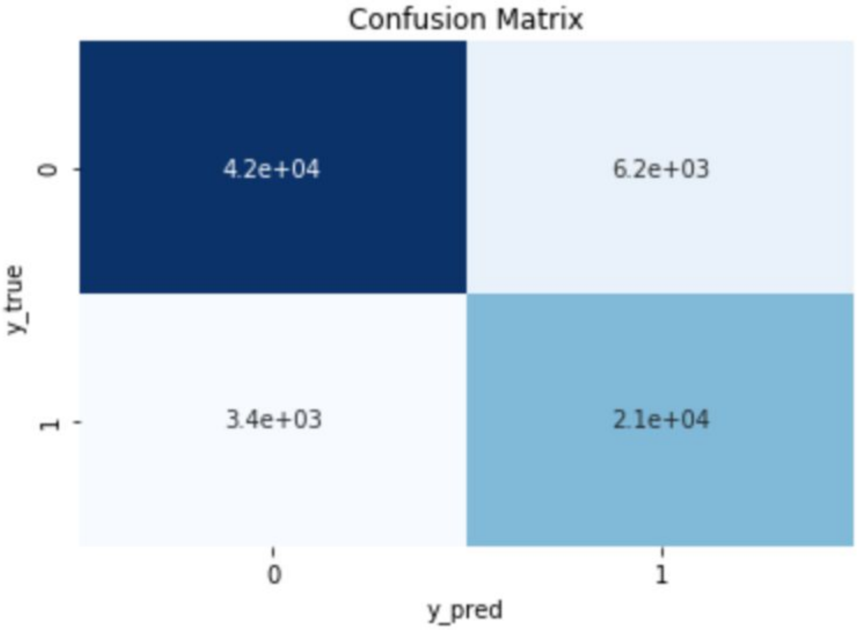


Comparison of different accuracy scores for Classification models



From the analysis, we concluded that Logistic regression was the best model for classifying our dataset. The confusion matrix for logistic regression below helps us get a better understanding about how well the questions were classified.

Confusion Matrix for Logistic Regression Model

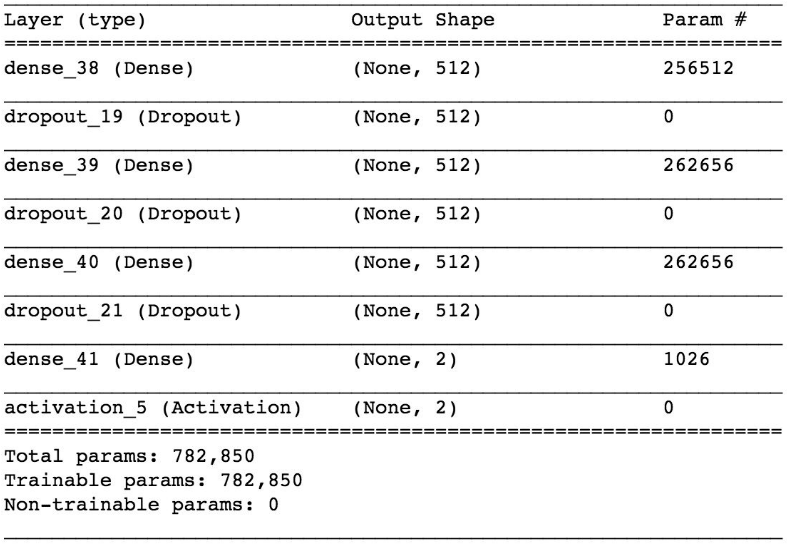


Previously, we also obtained other demographic features pertaining to the data which included the length, number of words, punctuation, etc. We want to try and include them in the analysis while fitting the model to test whether these demographic features are important features for our classification problem. We included these features while fitting the data to logistic regression model, and visualize the feature importances. From the analysis, we concluded that the different demographic features do not hold much importance in our classification problem. They can be disregarded and we can focus solely on the actual text of the question to be able to effectively classify it as sincere vs insincere.

Deep Learning:

We implemented different classifiers to classify question as sincere vs insincere and we got good accuracy results. But, to enhance our classification algorithm, we can also try to implement deep learning in the form of neural networks which would make the model more robust and result in better performance. First we will try a simple deep neural network with multiple layers and in the following network we will also try to implement LSTM and assess the classification performance.

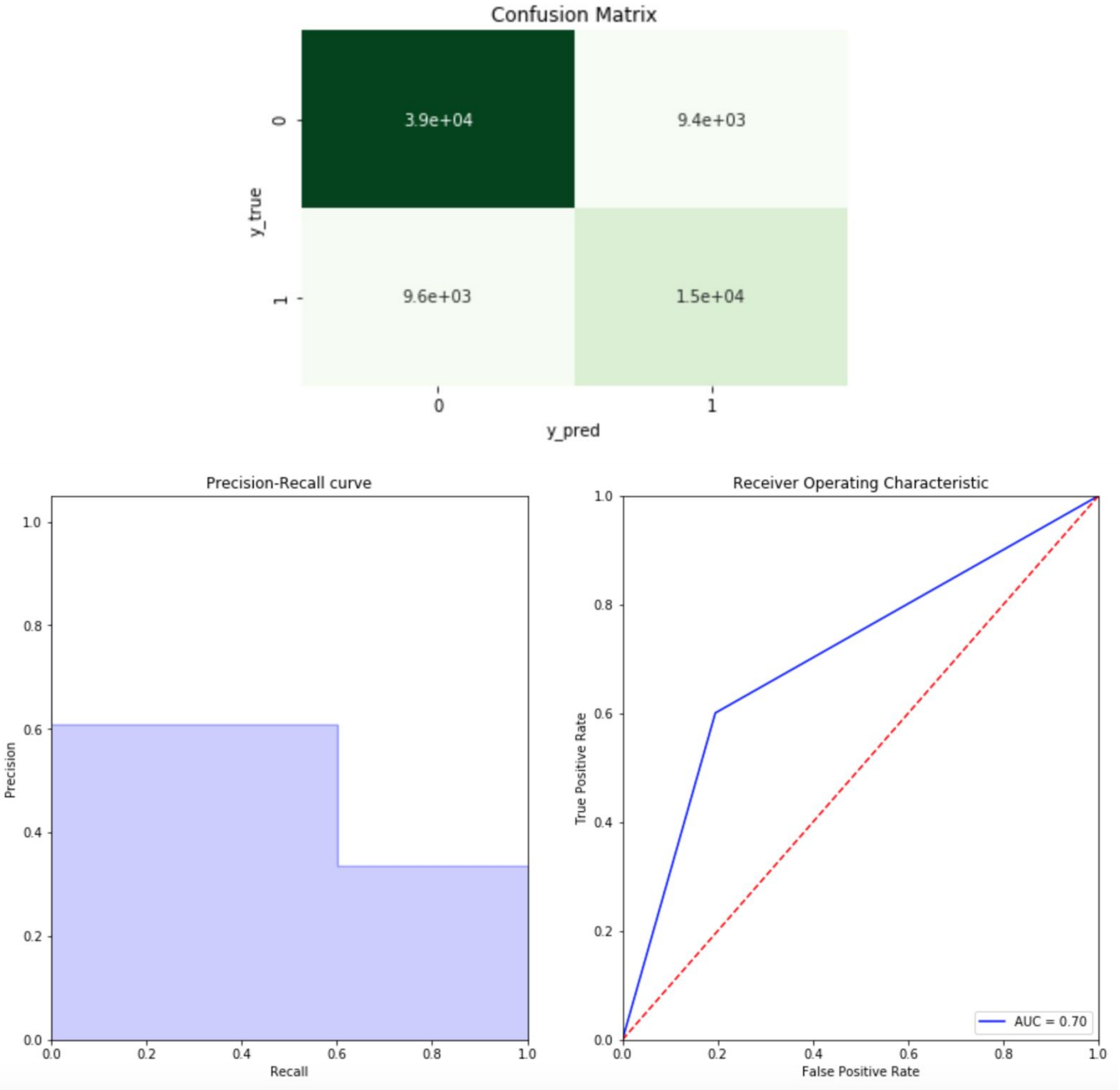
We used the word2vec vectorizer to vectorize the question text for deep learning. The first model built was a simple deep neural network. The summary of the network can be see in the following table:



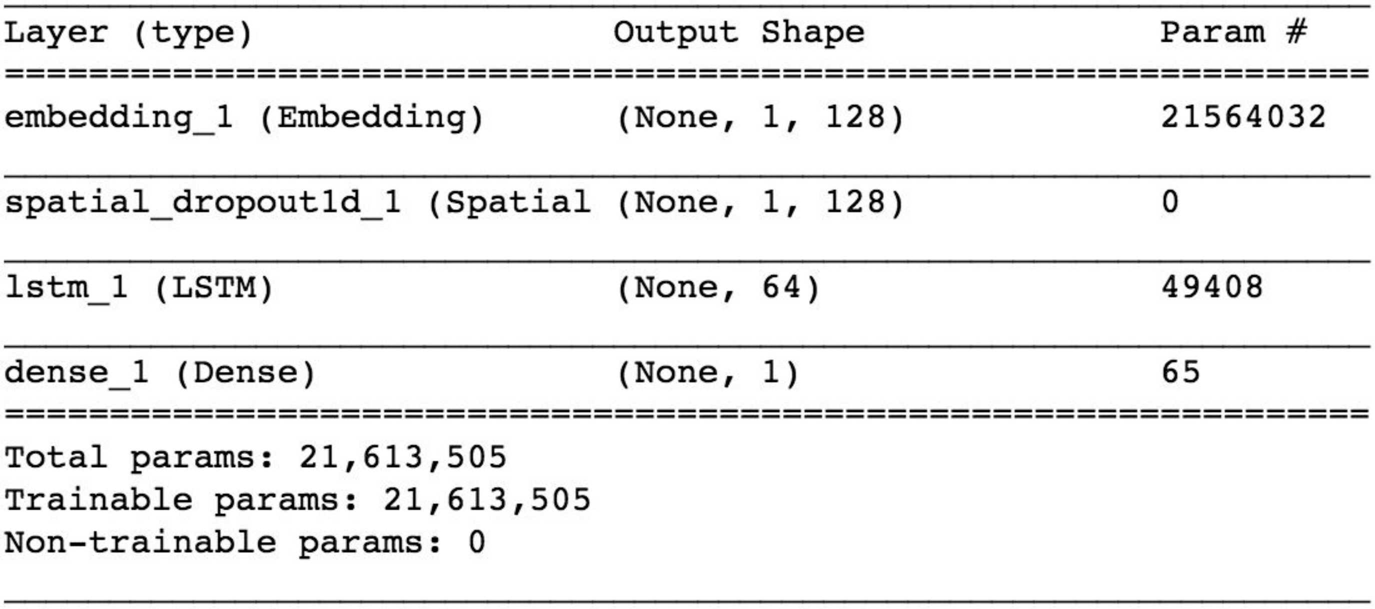
```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
dense_38 (Dense)             (None, 512)               256512    
_________________________________________________________________
dropout_19 (Dropout)         (None, 512)               0         
_________________________________________________________________
dense_39 (Dense)             (None, 512)               262656    
_________________________________________________________________
dropout_20 (Dropout)         (None, 512)               0         
_________________________________________________________________
dense_40 (Dense)             (None, 512)               262656    
_________________________________________________________________
dropout_21 (Dropout)         (None, 512)               0         
_________________________________________________________________
dense_41 (Dense)             (None, 2)                 1026      
_________________________________________________________________
activation_5 (Activation)    (None, 2)                 0         
=================================================================
Total params: 782,850
Trainable params: 782,850
Non-trainable params: 0
_________________________________________________________________
```

Within 5 epochs, we got an accuracy score of approximately 73% using this neural network. Other score metrics were also assessed.

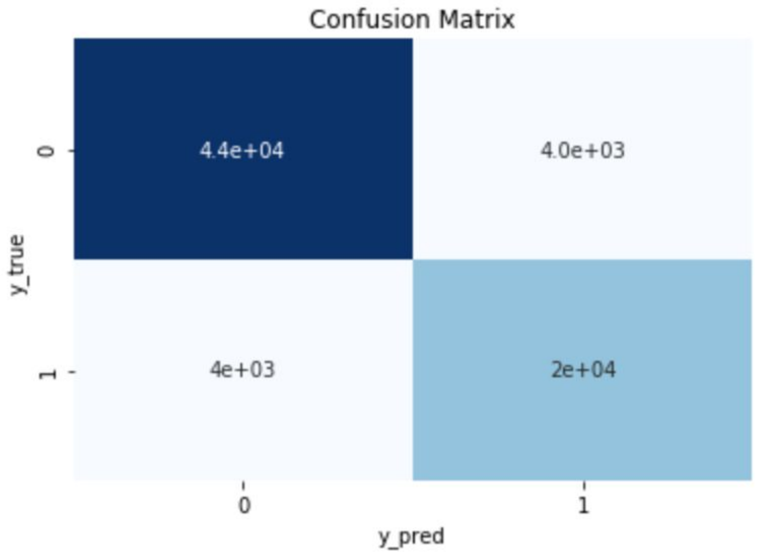
Confusion Matrix, Precision-Recall Curve and ROC curve for Deep Neural Network

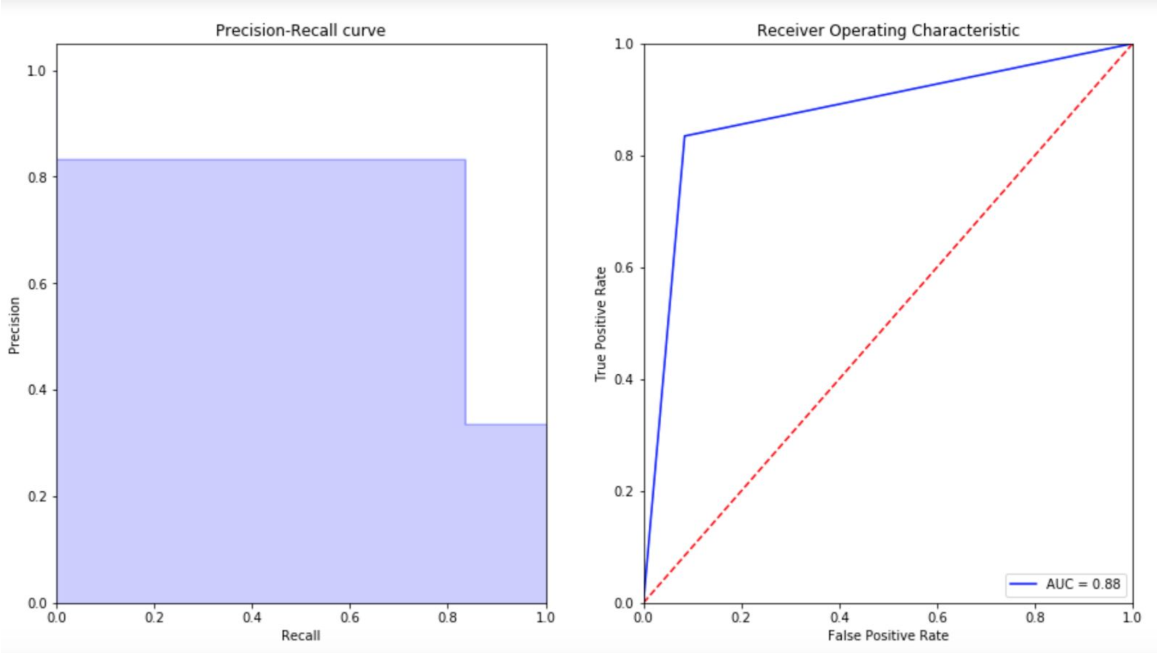


Next, we want to use LSTM - Long Short-Term Memory Units which help preserve the error that can be back propagated through time and layers. By maintaining a more constant error, they allow recurrent nets to continue to learn over many time steps. The neural network with LSTM incorporated can be summarized as below:



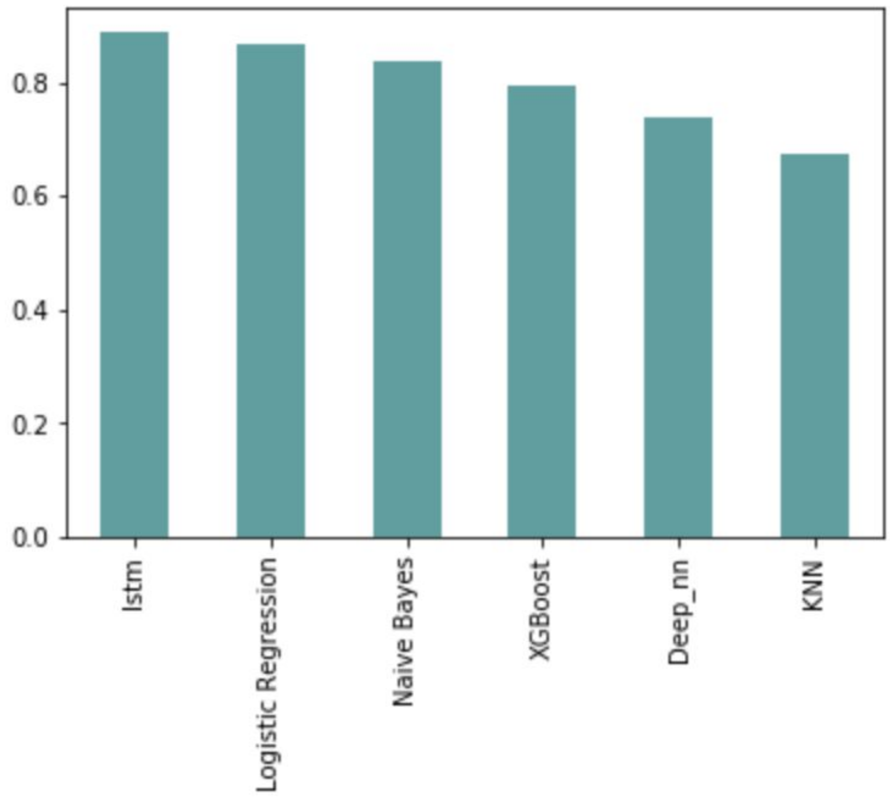
```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_1 (Embedding)      (None, 1, 128)            21564032
_________________________________________________________________
spatial_dropout1d_1 (Spatial (None, 1, 128)            0
_________________________________________________________________
lstm_1 (LSTM)                (None, 64)                49408
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 65
=================================================================
Total params: 21,613,505
Trainable params: 21,613,505
Non-trainable params: 0
_________________________________________________________________
```





Neural network with LSTM performed the best out of all the machine learning and deep learning models. We got a training accuracy of 94% and testing accuracy of 89%.

Comparing testing accuracy of different models



Conclusion:

We can conclude that the neural network with LSTM gave the best accuracy and was the best performing model for our classification data. Since we were evaluating questions for insincere/tox content it was important to look at the questions as a whole rather than just evaluating parts of it separately. LSTM allowed us to do exactly that and thus became the best performing model.

This model would perform great in detecting insincere questions on Quora, and thus help solve the problem to a certain extent. Being able to detect toxic and insincere content in text is in general very important, so this model can also be generalized.

Future directions:

I used a subset of the data to run my analysis since I was training the models on a CPU. But to include all the data for model training, we can run future analysis using a GPU which would be able to handle more intensive tasks.
As for improving the model, we can try to increase the number of epochs to 10 or 20 to keep improving the model accuracy and decrease the loss. This was because each epoch took a long time and CPU wouldn’t be able to handle such heavy processes. This problem can also be solved by using a GPU.
Different models can be combined together as well to increase efficiency of the model - using bidirectional LSTM, convolutional neural networks, etc. We can aim to increase the accuracy for identifying insincere text and train a robust model which can be generalized to all kinds of text data.