Springboard Capstone 1 : TED Talk Text Analysis

Pavan Poosarla; pavanpoosarla01@gmail.com

1 INTRODUCTION

The goal of this capstone project is to automatically assign tags to a TED talk based on its transcript. To do this, we train our model on the existing TED talk dataset. TED talk dataset is taken from the Kaggle repositories.

Kaggle TED talk transcripts : https://www.kaggle.com/rounakbanik/ted-talks/version/3

2 PROBLEM STATEMENT

The goal of this project is to predict the most popular rating of the talk based on the transcript and other metadata associated with the talk (like tags, length of talk, audience reactions, etc.).

On the TED website, there are several possible ratings, some of them with similar connotations (for example, Longwinded and Confusing). In order to get sufficient samples for each rating, we consolidate the ratings into the following categories for the machine learning section

Ratings the models are trained on are : 'Fascinating': 1, 'BadTalk': 2, 'Beautiful':3,'Informative':4, 'Funny':5

1. Fascinating
2. BadTalk (consolidation of all negative ratings: Confusing, Obnoxious, Longwinded, etc.)
3. Beautiful
4. Informative
5. Funny

3 DESCRIPTION OF SOURCE DATA

This data has been scraped from the TED talk official website and is available now by a creative commons license. The database consists of talks scraped till Sep 21st, 2017. It has two files, containing information from about 2550 talks from >350 individual TED and TEDx events across the world. This gives a excellent medium to develop text analysis and NLP techniques to automatically predict sentiments from verbal speech

ted_main.csv : This file contains the metadata associated with each talk. The columns included in this file are shown in the table below

transcripts.csv : It consists of a file consisting of the text of the transcript along with the corresponding url of the talk. The columns on this file are shown in the table below

Table 1 List of columns and their short descriptions from the 'ted_main.csv' file of the dataset

Column	**Description**
comments	Number of first-level comments on the talk
description	A short note about the talk, usually 1-2 sentences
duration	In seconds
event	Event the talk was presented in
film_date	Unix timestamp of the filming
languages	Number of languages talk is available in
main_speaker	Main Speaker
name	Official name of TED talk, including speaker and title
num_speaker	Number of speakers
published	Unix timestamp of when talk appeared on ted.com
ratings	A stringified dictionary of the various ratings given to the talk (inspiring, fascinating, jaw dropping, etc.)
related_tags	A list of dictionaries of recommended talks to watch next
speaker_occupation	The occupation of the main speaker
tags	The themes associated with the talk
title	The title of the talk
url	The URL of the talk. This is common column with transcripts file, used for merging the two files on
views	The number of views on the talk

Table 2 List of columns and brief descriptions from 'transcripts.csv' file in the dataset

Column	Description
transcript	This is the transcript of the talk. Text is not divided into paragraphs. It includes comments and audience reactions in parenthesis. For example *(Applause), (Laughter), etc.*
url	URL link of the talk. This is the common column with the *ted_main.csv* used to merge both files on

4 DATA WRANGLING

The first step in creating a usable dataset to extract statistics from and test machine learning methods is to do data wrangling and output a cleaned dataset. We develop a cleaned dataset, 'df_clean'. To do this, we do the following steps

1. **Join** : The two individual files, 'ted_main.csv' and 'transcripts.csv' are merged on the column url with a inner join. This outputs a dataframe containing information on 2467 talks, each having 18 data columns.
2. **Missing Data** : Looking for missing values in all the columns, we find the dataset to be relatively clean. Only column having missing values is the 'speaker_occupation' column with 6 values missing. We replace missing data with the string 'Unknown'

3. **Remove Outliers :** We also drop the talks with more than 1 speaker. As we have a very few talks with more than 1 speaker, we do not want to contaminate the training data with these talks.
4. **Drop Extra Columns :** Next, we drop the columns which are either redundant or unnecessary for out analysis. The columns dropped and the justification is shown in the table on the next page. Note that some of these columns will provide insights into the data storytelling , but may not be relevant to the machine learning aspects (Categorical with too many categories, etc.)

Format Changes : Another change we do is to convert the date/time fields , i.e, 'film_datestamp' and 'pub_datestamp' into python datetime format. This makes it easier to perform datetime manipulations of these datefields in the future. It also makes these column data human readable.

Table 3 List of columns dropped during data wrangling with justifications/ reasoning for each

Dropped Column	**Reasoning/ Justification**
event	Not relevant for learning. Perhaps relevant to data storytelling
languages	Not relevant as we only consider english transcripts. Can be relevant to data storytelling, where it is included.
name	Redundant data. Has title and speaker name
num_speaker	All are 1. Redundant data
related_talks	Dropped. Not relevant
url	Dropped. Not relevant
views	May be too generic a metric to predict or use for training. But included for possible Data Storytelling Assignment and for qualitative validation later on

Once the above operations are completed, we also extract extra features based on analysis of the transcript text. The features extracted from the text are

1. *sentence_count* : Counts the number of sentences in the transcript.
2. *word_count* : Counts the number of words in the transcript
3. *aud_reaction_dict* **:** Creates a dictionary of audience reactions in the transcript (The transcripts contain these in the form of parenthesis)
4. *ratings_dict :* Use a inbuilt function to convert the ratings to a dictionary.

Finally, the two dictionary columns we created, *aud_reaction_dict* and *ratings_dict* are converted to dataframe with multiple columns, with entries in each.

Due to the large number of unique audience reactions (also includes situational descriptions like background music, video playing, etc.), we keep only most popular reactions. Common variants like *laughter* and *laughs* have also been consolidated. The final list of audience reactions are `'laughter', 'applause', 'music', 'cheering', 'sighs', 'video', 'singing', and 'audio'`

Finally, we also separate out the *reaction_dict* into eight columns with the counts corresponding to the following reactions

```
'Funny', 'Beautiful', 'Ingenious', 'Courageous', 'Longwinded', 'Confusing',
'Informative', 'Fascinating', 'Unconvincing', 'Persuasive', 'Jaw-dropping',
'OK', 'Obnoxious', 'Inspiring'
```

The final dataframe has 2412 rows and 40 columns. It is written out to a csv file named *'After_DataWrang_Out.csv'.* We will use this file as an input to the data storytelling and the machine learning portions of the project.

5 Data Storytelling

For data storytelling, we start off with the output file from the data wrangling portion of the project. This data is read from the file output after data wrangling (*After_DataWrang_Out.csv*). This file has information on 2412 talks. After dropping redundant columns which have been parsed earlier, each of these talks is left with 35 columns.

5.1 Visualizations of Metadata

The talks in the dataset are from multiple TED event types. We look at the distribution of talks in each of the major TED categories. The distributions are shown below

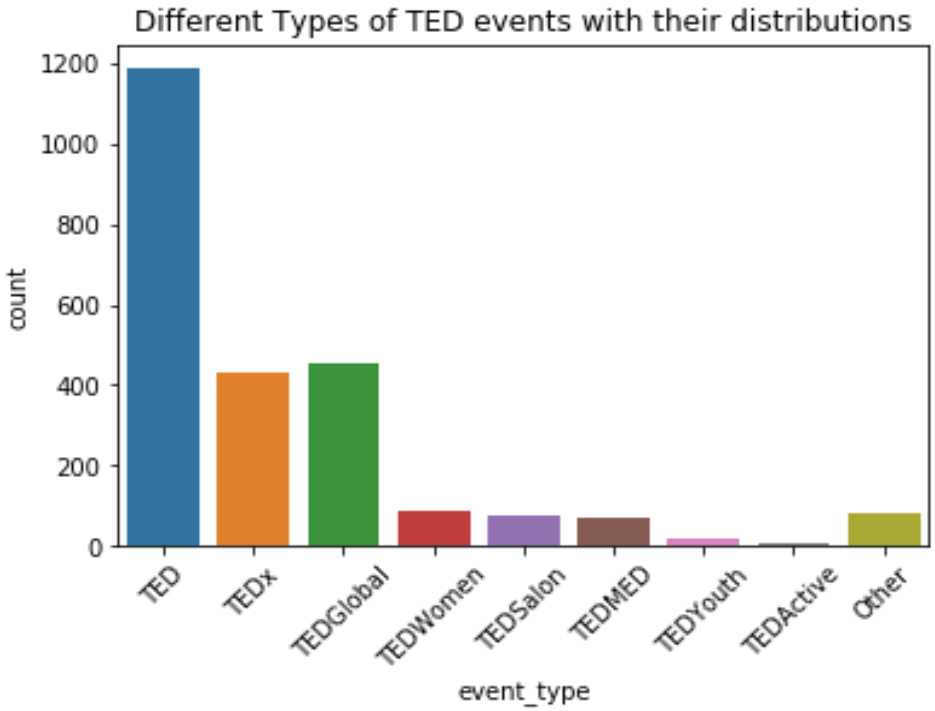

Figure 1 Bar chart showing the counts of various types of TED talks in the dataset

We see that based on the number of talks, traditional TED events are the most common. Amongst the variants, TEDx and TEDGlobal are the next most popular versions. Other events, like TED Women, TEDSalon and TEDMED are much smaller in scale. However, TED Youth and TEDActive have the fewest talks of all the events. All other less common variants are grouped under "Other" on the plot.

Looking at the talks themselves, we see the distribution of talk duration and words per minute as shown below. The mean for the distributions are marked in red.

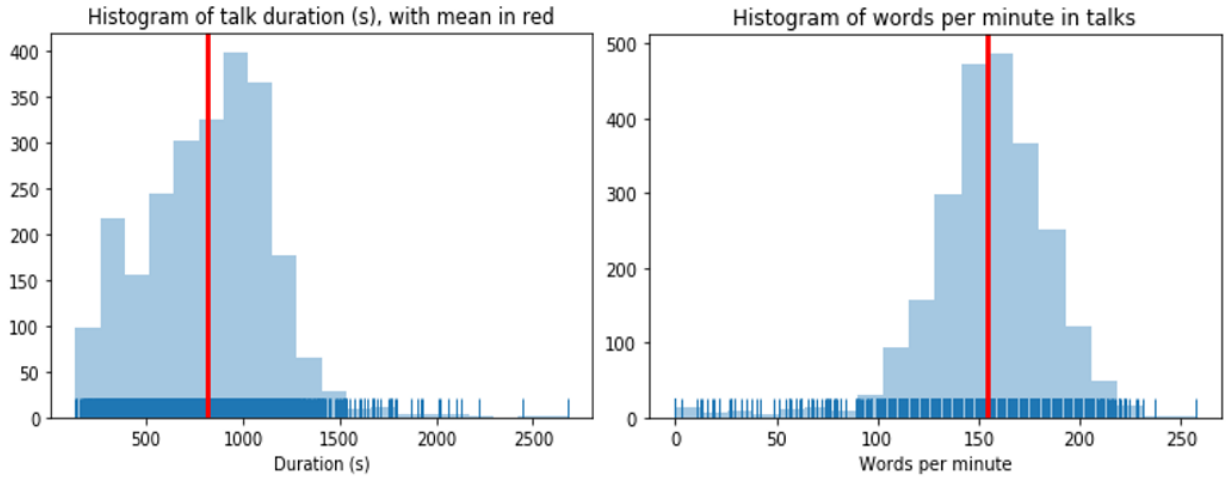

Figure 2 Histograms of (a) Talk duration (in seconds), and (b) Words-per-minute of the talks

While both have a strong gaussian component, we see that they also have few outliers which are much outside the distribution. Looking at a few examples, we see that these are not traditional talks. These are either musical performances (Those with zero words per minute), product launches, interviews, ad-hoc interviews with notable persons and so on. To avoid these contaminating the dataset, we drop the talks with 'duration' and 'words_per_min' lower than 0.01 percentile and greater than 0.99 percentile.

After filtering, we see that the average duration of a TED talk is about 13 minutes. Most of the talks have a average speed of about 156 words per minute. This is higher than the typical speaking speed for an average person. This is most likely due to the words per minute being contaminated with the audience reactions in the transcript.

5.2 Ratings Columns

To understand the impact of various predictors on our target variable, we will look at the various columns related to ratings. We have 14 columns related to the ratings. We would like to reduce these and also finalize the predictor for applying machine learning to predict the rating. To do this, we look at correlations between different ratings to see which can be dropped and which can be consolidated with others.

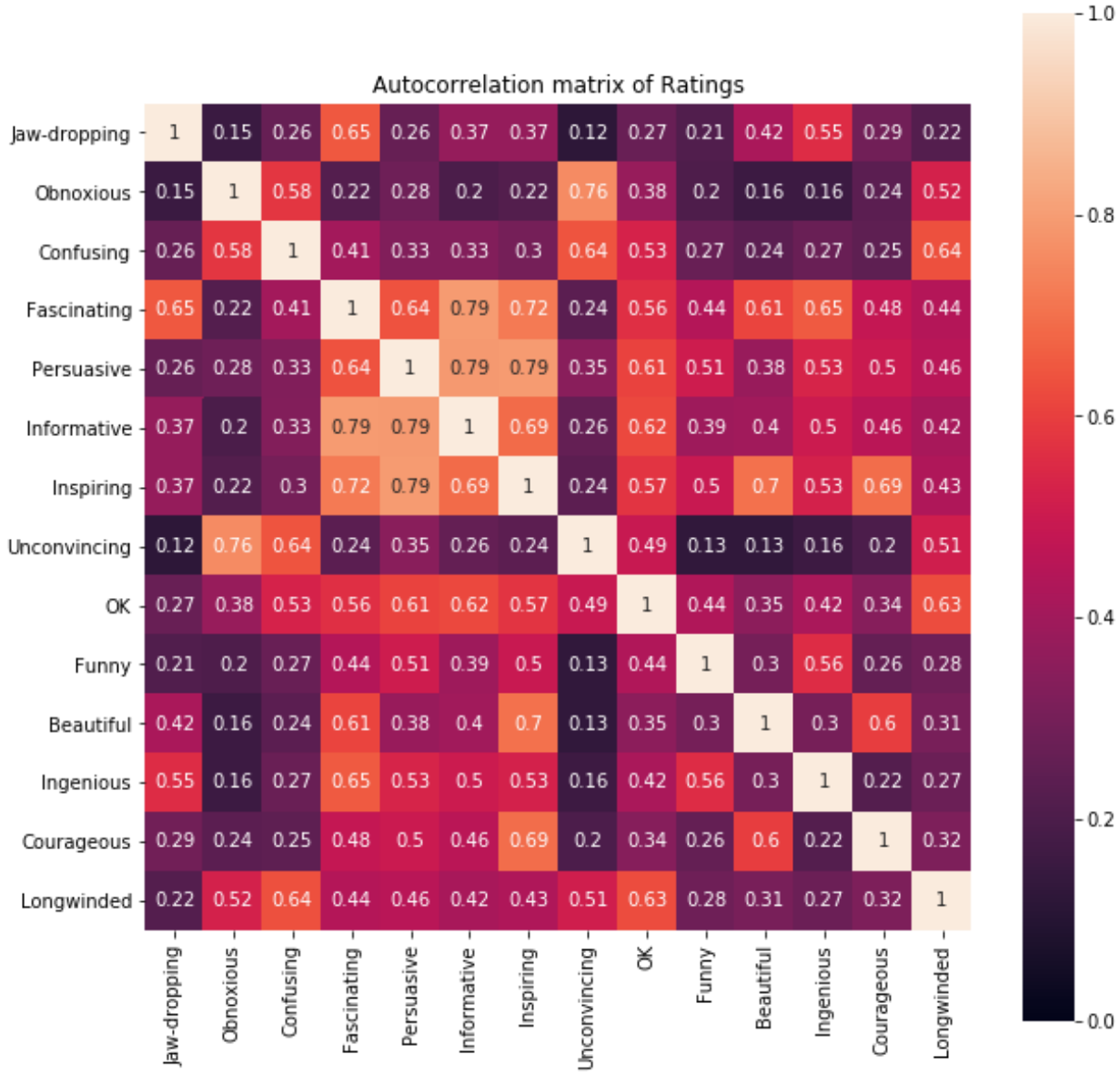

Figure 3 Auto-correlation map of various ratings in the talks. We will consolidate related ratings together to arrive at 5 primary ratings used for classification

From the above autocorrelation matrix, we see clear correlations between few of the ratings. For example, most of negative ratings are well correlated. Some of the positive ratings are also correlated (like Inspiring and Persuasive). However, some of the ratings are ambivalent, which might mean that they can be used either with positive or negative connotation (Jaw=dropping, for example). Finally, some positive ratings are not correlated to others. For example, “Funny” and “Inspiring”, although both positive, are not necessarily correlated.

Here is the summary of observations

1. 'Inspring' is strongly correlated to 'Persuasive'. This is understandable based on the similarity in meanings

2. Negative sentiments are highly correlated. Example, "Obnoxious" and "Unconvincing", "Longwinded" and "Confusing", etc.

2. Some of the ratings seem to have multiple connotations. For example, 'Jaw-dropping' is correlated to 'Fasciating' as well as 'Confusing'. As meaning is unclear, we make a call to drop it, expecting the sentiment to be captured elsewhere

3. 'Funny' is a unique reaction unto itself. So, we do not see any strong correlation to other items. Some talks may be Funny and Persuasive while others may be Funny and Inspiring. However, it is a positive connotation

4. 'Ingenious' is also a independent reaction, not directly correlated to other emotions

5. 'Informative' is strongly correlated to 'Fascinating' and 'Persuasive'

Based on the above insights from auto-correlation matrix, we consolidate the ratings as follows

1. Strongly correlated ratings are merged to the more common rating and rating with lower counts is dropped
2. Ratings which can be used in positive and negative contexts, like 'Jaw-dropping' are dropped.
3. All negative correlations strongly correlated with each other but do not have a common sentiment, "Longwinded", "Obnoxious", "Confusing" and "Uninspiring" are grouped together as "BadTalk"

After consolidating the ratings in this way, we end up with five ratings, which we will train the model on. The final list of ratings selected is: '*Fascinating', 'BadTalk', 'Beautiful', 'Informative' , and 'Funny* . The counts for each rating are shown in bar chart below

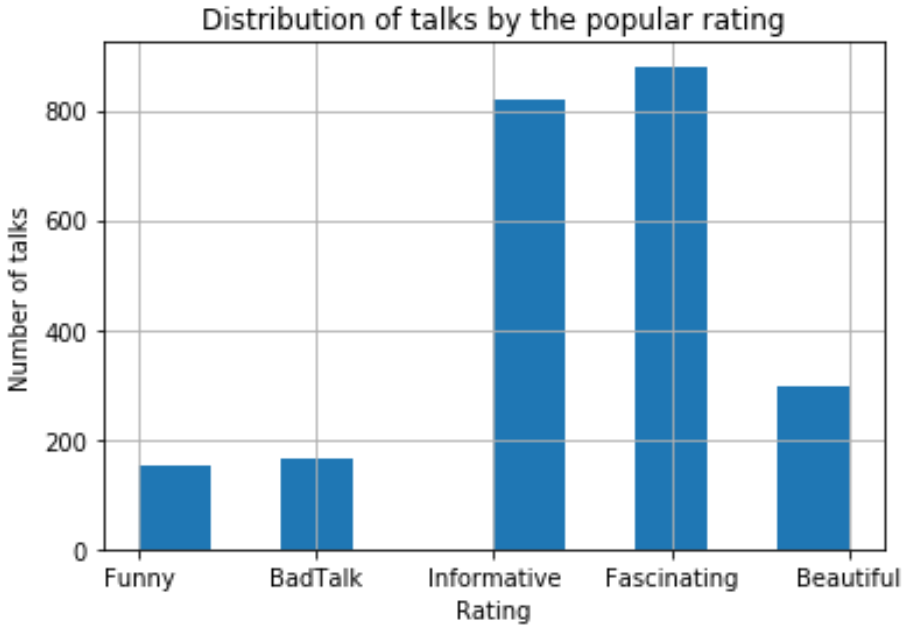

Figure 4 The number of talks by each rating after consolidation

5.3 TEXT-ANALYSIS : PREDICT THE WORDS STRONGLY PREDICTIVE OF SPECIFIC RATINGS

In this section, the transcripts are treated in a bag-of-words manner to predict the words that strongly predictive of a particular rating. The analysis done to do this is as follows

1. Create a cleaned transcript after pre-processing
2. Convert the pre-processed text into bag-of-words vector
3. Use the bag-of-words vector to predict rating using Naïve Bayes model
4. Calculate the probabilities of each rating for each word in vocabulary
5. For each rating, extract the five most common words that are predictive of each rating

In the first section, the exact **pre-processing** steps used on the text are

a. Removal of accented characters (like ï , for example)
b. Convert text to lower case
c. Remove all audience reactions (like applause, laughter, etc.) from the text.
d. Expand contractions. For example, replace “won’t” by “will not”
e. Remove punctuation

After pre-processing is completed, the resulting transcripts are converted into a count vectors using the bag-of-words approach. These count vectors are then used to train a Naïve Bayes model. This is then used to predict relative probabilities of each class for each word in the vocabulary.

Next, for each rating, the words having highest probability for that rating are extracted as best words predictive of a particular rating. Similarly, the words with lowest probability for a particular rating are extracted as worst words. The results for each of the ratings are shown in the tables below

Table 4 Table of best predictive words for each rating

Rating	Fascinating	BadTalk	Beautiful	Informative	Funny
Best words	light (0.65)	god (0.10)	song (0.21)	country (0.61)	Humor(0.11)
	robots (0.65)	tapirs (0.10)	girl (0.21)	government (0.62)	comedy (0.11)
	universe (0.68)	concussion (0.11)	poetry (0.22)	companies (0.62)	flag (0.11)
	cells (0.68)	glamorous (0.11)	compassion (0.22)	women (0.63)	laughter(0.11)
	robot (0.74)	glamour (0.12)	hum (0.22)	global (0.64)	dh (0.13)

Table 5 Table of the worst predictive words for each rating

Rating	Fascinating	BadTalk	Beautiful	Informative	Funny
Worst words	women (0.18)	data (0.02)	data (0.02)	robot (0.16)	percent (0.01)
	governments (0.23)	brain (0.02)	information (0.03)	universe (0.21)	world (0.01)
	rights (0.23)	two (0.02)	percent (0.03)	art (0.22)	years (0.01)
	sector (0.24)	cells (0.02)	brain (0.03)	object (0.22)	us (0.01)
	refugees (0.25)	see (0.02)	technology (0.03)	robots (0.22)	human (0.01)

6 STATISTICAL DATA ANALYSIS

For applying machine learning to the rating of the talk, we will try to predict the majority rating for each talk based on the metadata and transcript of the talk.

For statistical data analysis, we would like to evaluate if there is any impact of tags on the ratings. The most common tags in the dataset are shown below

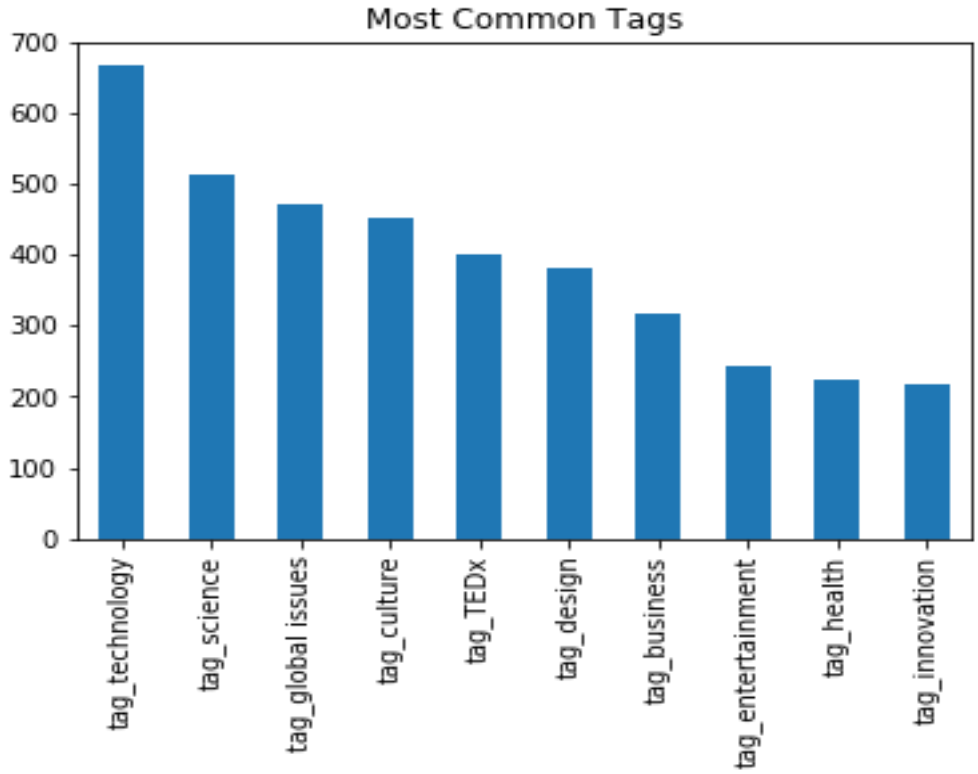

Figure 5 List of most common tags appearing in the dataset

Of these, we use one-hot encoding to represent the presence of most common tags. The common tags selected are: **technology, science, global issues, culture, design, business and entertainment.** Statistical methods are used to get a quantitative measure of correlation and dependence between various factors and ratings

We would be interested in finding whether the appearance of a particular tag would correlate to a particular rating for the talk. This would reveal insights about whether there are particular topics that intrinsically have a particular reaction from the audience. To measure this correlation statistically, we use point-biserial correlation to identify dependence of tags on ratings. The results are shown in the figure.

From the point-biserial correlation heatmap, we see that the tag ‘science’ is highly correlated to the ratings ‘Fascinating’ and ‘Informative’. Similarly, ‘technology’ and ‘design’ are correlated to ‘Ingenious’. ‘Global Issues’ tend to be correlated to ‘Inspiring’. We also notice that ‘Funny’ is correlated to ‘entertainment’ but not to any other tags considered. Similarly, negative ratings like ‘Confusing’, ‘Unconvincing’ and ‘OK’ are also not correlated to any specific tags

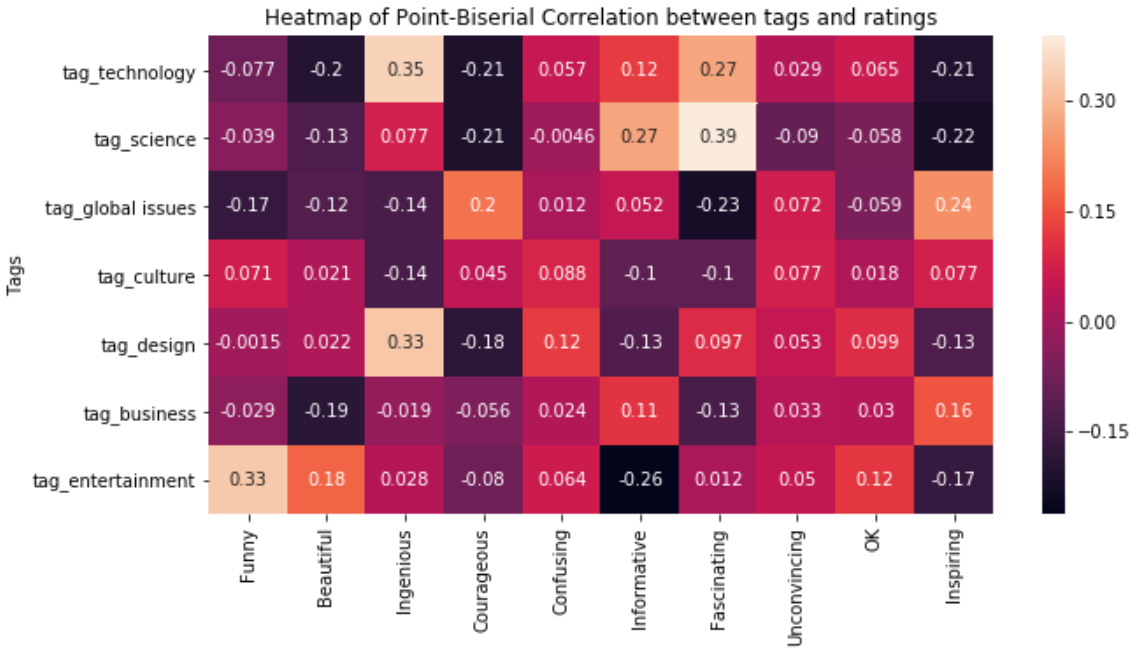

Figure 6 Point biserial correlation between tags and ratings

Finally, we use t-test to test specific relationships in the dataset. Specifically, we answer the following questions based on statistical hypothesis testing

Q1 : Are talks with tag 'technology' score differently compared to talks without this tag on the rating 'Informative'?

To perform the analysis, we postulate the following null and alternate hypothesis

Null Hypotheis : Talks with and without 'technology' tag score similar on 'Informative

Alt Hypotheis : Two groups are different

Performing t-test, the result obtained is

`Ttest_indResult(statistic=6.227420675983781, pvalue=6.344877075326349e-10)`

For a alpha of 5%, we can safely conclude that null hypothesis is disproved. Thus, technology tag has an impact on whether a talk is rated as ‘Informative’. As can be seen from the box plot, talks with technology tag rate higher than those without

Similarly, it is also concluded that ‘design’ tag impacts the rating of a talk as ‘Inspiring’. The test result is shown below

Q2 : Do talks with tag 'design' score differently on 'inspiring' compared to talks without?

Null Hypotheis : Talks with and without tag ‘design’ get rated similarly on ‘Inspiring’

Alt Hypothesis : They rate differently

T-test result

`Ttest_indResult(statistic=-6.615982464579601, pvalue=8.34047782299715e-11)`

We see that talks with tag ‘design’ get rated differently from those without the tag on ‘Inspiring’. Talks with ‘design’ tag score lower

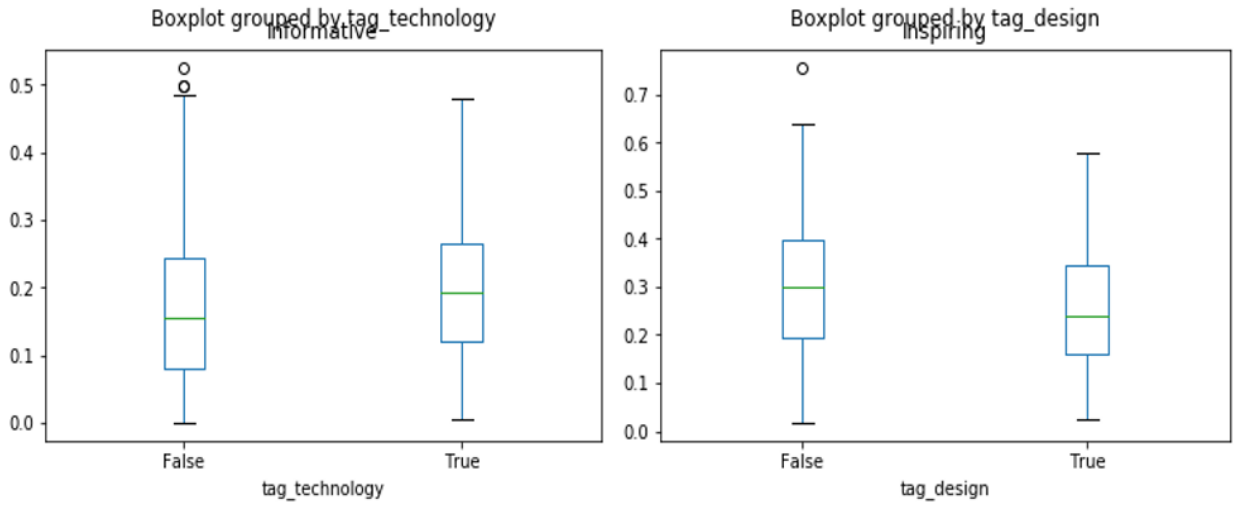

Figure 7 Box plots showing the (a) rating score of informative for talks with and without the tag 'Informative'. (b) Normalized scores on 'Inspiring' for talks with and without the tag 'design'

7 Applying Machine Learning

For this section, we will attempt to use machine learning methods to predict the normalized score obtained by each talk for a particular rating. The ratings of interest are ‘Inspiring’, ‘Informative’ and ‘Funny’.

The model training and tuning part can be considered in three steps. These are

1. **Define features and targets. Split into test and train sets**
   In our current dataset, feature vector is made by combining the matrix obtained from count vectorizer of cleaned transcript with the matrix containing metadata columns. The combined feature vector has 6972 columns(features) and 2314 rows(talks). This matrix is stored as a sparse matrix. Target variable y is the ‘*Max_rating’* column where every class is denoted by a number from 1-5.
   The resulting feature and target data is split into train/ test datasets with 80%/ 20% ratio.
2. **Train Model and define metric to evaluate performance**
   We train three different models (a) Logistic Regression, (b) Naïve Bayes , and (c) Random Forest Classifier on the dataset. Balanced accuracy score is used as the scoring metric. This is chosen to account for unbalanced classes in our dataset.
3. **Optimize hyperparameters and test performance on test set**
   The model hyperparameters are optimized using gridsearchCV (or RandomsearchCV for random forest). The best model performance is plotted as classification report and confusion matrix.

Further information on each of the models trained is discussed in the following sub sections, after a short discussion about the impact of unbalanced classes on our model

7.1 Impact of Unbalanced Classes

As we notice from Figure 4, the number of talks in each rating class not equal. Due to this imbalance in the class, the trained models are likely to be biased toward predicting the more common class, in this classes, in this case, *Informative* and *Fascinating*.

In logistic regression, this can be handled by weighing each class inversely proportional to the class frequencies. However, for other models, the performance while predicting less frequent classes will be impacted when training data is unbalanced.

To overcome this drawback, we use SMOTE oversampling to oversample the minority class. This oversampled data is used to train the classifiers to reduce the bias due to class imbalance in the dataset.

7.2 Comparison of Models

Logistic Regression :

As described above, training data is used to train logistic regression model and hyperparameter, C is optimized using GridSearchCV using 3-fold cross validation.

The classification report is shown below

	precision	recall	f1-score	support
1	0.71	0.68	0.69	161
2	0.16	0.09	0.11	34
3	0.61	0.68	0.64	62
4	0.71	0.73	0.72	179
5	0.59	0.85	0.70	27
accuracy			0.66	463
macro avg	0.56	0.60	0.57	463
weighted avg	0.65	0.66	0.65	463

Naïve Bayes

Similarly, the classification report for Naïve Bayes after model tuning is

	precision	recall	f1-score	support
1	0.62	0.34	0.44	176
2	0.14	0.47	0.22	32
3	0.55	0.36	0.43	64
4	0.62	0.53	0.57	164
5	0.23	0.67	0.34	27
accuracy			0.44	463
macro avg	0.43	0.47	0.40	463
weighted avg	0.55	0.44	0.46	463

Random Forest Classifier

And for the random forest classifier after tuning hyperparameters with RandomsearchCV, we get the report as below

	precision	recall	f1-score	support
1	0.52	0.81	0.63	161
2	0.00	0.00	0.00	34
3	0.88	0.11	0.20	62
4	0.60	0.67	0.63	179
5	1.00	0.19	0.31	27
accuracy			0.57	463
macro avg	0.60	0.36	0.36	463
weighted avg	0.59	0.57	0.51	463

Random forest classifier does not seem to be performing well on the 'BadTalk' class at all.

7.3 Picking the Best Model

Comparing the performance of the three models developed, we see that Logistic Regression performs better than Naïve Bayes. Naïve Bayes seems to have lower recall than Logistic Regression. Random forest, while better than Naïve Bayes, is still not as good as the Logistic Regression Classifier. Hence, we pick the Logistic regression classifier as the best model for this problem. The confusion matrix obtained from logistic regression classifier is shown below

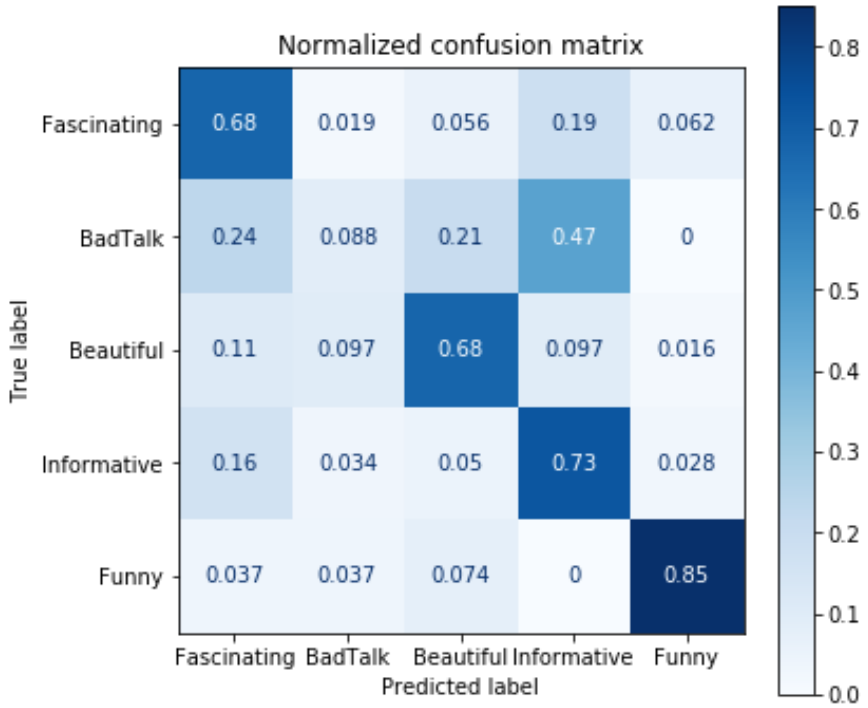

Figure 8 Confusion matrix of the final model selected, Logistic Regression

8 CONCLUSION

In conclusion, we describe the model for predicting the most prominent rating for a TED talk that considered both the transcript as well as the metadata associated with the talk. This can predict, with reasonable accuracy, the prominent audience reaction (rating) simply based on attributes of the talk and the topics of the talk (tags).

Github link for the project is : https://github.com/PavanPoosarla01/Springboard-Capstone1.git