Capstone Project

Diogo Viana

d.cviana@outlook.com

https://www.linkedin.com/in/diogo-viana/

December 2022

BrainStation

Table of Contents

Situation assessment

TED is a non-profit organization created with the intention to share ideas by having regular people talking about powerful subjects. The videos have an average length of 18 minutes or less on average.

The main goal of this Capstone project is to assist the speaker to understand which words are correlated to high number of views. This document has the objective to explain the process and the findings of my analysis.

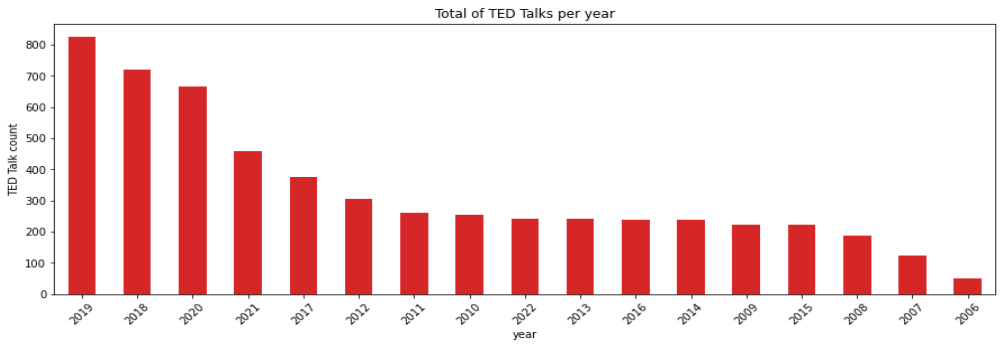
Executive Summary

Sometimes the TED talk speaker could have a good subject and speech, but may not achieve a certain number of views because of the choice of description. The value added with my project is that the speaker will be able to achieve more views and as a consequence, spread their idea widely.

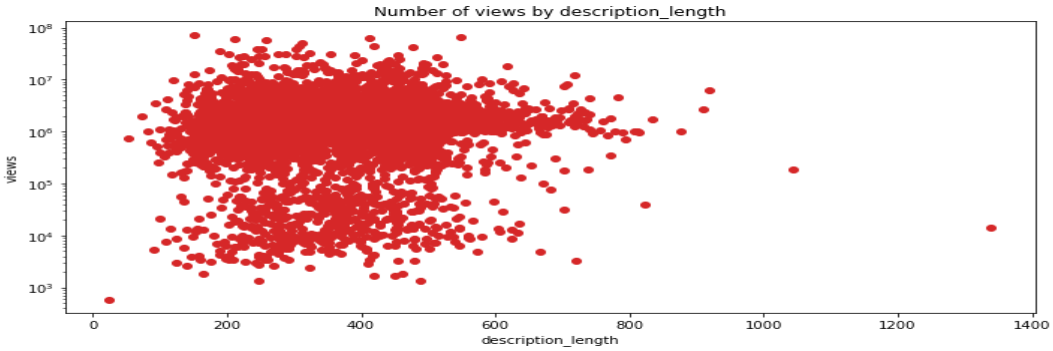
The data was collected from Kaggle where a data scientist made available after completing a web scraping of each page from the official website of TED talk. The original dataset had 5631 rows and 16 columns.

The first part of my analysis was Data Cleaning and Exploratory Data Analysis (EDA). The first step in my notebook was to rename the columns to have an easier reading. I found some missing value where I properly dealt with it. But the most important modifications were counting the length of the columns with a list as information and changing it to numerical.

Moreover, during the EDA, I learned that based on the number of views and likes, the top 10 TED talks are very similar in both cases. In addition, since 2006, the years 2019 and 2018 presented the most number of TED talks.



Furthermore, when I checked the correlation between the features, as expected, views and likes were highly correlated. So, I decided to create a new feature, popularity ratio, by dividing the number of likes by the number of views multiplied by 100. After checking all the correlated columns, I decided to transform the description column into numerical by counting the number of characters. So, I used this new column to see if the length would have any impact on the number of views. After visualizing, I could see that there was no relation between both columns.



The second part was modeling. Therefore, after performing the vectorization using Bag of Words, I was able to use the one-hot-encoding columns and concatenate all the information in one unique data frame. During this process, I trained and tested models to check and compare the performance with Logistic Regression, K-Nearest Neighbor and Decision Tree.

The first model was Logistic Regression and since the beginning, it did not demonstrate a good performance. I began by checking the best C value and then defining the best one. I ran the model again and the delta between Train and Test continued very high. Therefore, I applied 5-fold cross validation and used a pipeline to check for multiple hyperparameters. In conclusion, after applying the best hyperparameters, the F1 score was 73% while the gap between Train and Test was almost 10%, which shows that the model was overfitting.

My second model was KNN and it also did not show good results before applying any hyperparameter optimization. So, I checked for the best k value and found out that the best would be equal to forty-five. The delta between Train and Test was smaller than using Logistic Regression, but the accuracy of this model was very bad, F1 score was equal 59%. So, I tried the next model.

The Decision Tree presented a good accuracy score for the test set without any hyperparameter optimization, however, the train set overfitted and its accuracy was 100%. So, I verified the best max_depth and found out that the best one was equal to five.

When comparing all three models, Decision Tree achieved the best performance with F1 equal to 80%. In addition, the delta between train and test was very small.

	Log. Regression		K-Nearest Neighbor		Decision Trees	
	Plain Model	**Optimized**	**Plain Model**	**Optimized**	**Plain Model**	**Optimized**
Train	95.61%	81.13%	68.10%	64.61%	100%	79.93%
Test (F1 Score)	71.23%	72.28%	51.68%	58.61%	74.77%	79.57%

Therefore, I used the Decision Trees feature importance to extract the words from the description that are correlated to high number of views. The words are the following:

smart | funni(est) | boost | appli(cation) | autism | toma(to) | none | energi(y) | cem(ent) | half

Description based on the words:

“Let’s talk about autism. I would guess that half of the guests in here do not know a great deal about this subject, it does not matter if you consider yourself smart and the funniest, tomato /tomato. I want to present ideas that would boost your knowledge. (...)
(…)So, those are some of the applications you should consider. Your brain is like a truck full of cement, if you stop applying energy to move it, it will get stuck. Therefore, none of your previous efforts will make a difference(…)”

In the third part, I used a Recurrent Neural Network to identify words that could be predicted by using the description information. After completing the data preparation, I used the column `description` to split into characters and then into Train, Validation and Test sets. The next step was performing a single-layer RNN, with an embedding layer that produces 8-dimensional character-vectors, a single LSTM layer with 128 units, a dense hidden layer with 64 units and an output layer.

The objective of using this model was to be able to predict the next character based on the input. During the process, I started by running 20 epochs, but the validation and train did not plateau. I kept trying, next I ran with 50 epochs and still not plateauing. So, I jumped to 100 epochs and then 300 epochs, where finally validation and train leveled-up. The validation accuracy was around 60%, which is considered not bad for predicting the next character.

The following stage was performing deep recurrent neural networks. This time I already started with 300 epochs, but it did not level-up. So, I used 500 epochs which presented a very good result. The sentence generated by the model could not be considered as a good result, however, the set of words given by the model was very good.

Moreover, now I will construct a description based on the word embeddings.

<u>details | living | question | science | ideas | studio | between | modern | funny | go</u>

“If you do not find the idea of living paycheck to paycheck in a not modern studio apartment funny, I have a question for you. Have you ever thought of starting a career as a Data Scientist? So, in today's TED talk, I will talk about the delightful Data Science market in detail. (…)
(...) So, do not be scared and let something between you and a successful life stop you. Just GO!“

Conclusion

In coclusion, the results from the created descriptions appear to be good. However, when comparing both results from the Decision Tree and RNN, we could not notice much similarities.

Therefore, based on the accuracy of 80%, the words generated with Decision Tree feature importance, since they are positively correlated to the high number of views, should be the ones used to structure the description to assist the speaker to achieve a better number of views.

References

"1.10. Decision Trees." *Scikit-learn*, scikit-learn/stable/modules/tree.html. Accessed 10 Dec. 2022.

"Our Organization." *Our Organization | About | TED*, www.ted.com/about/our-organization. Accessed 10 Dec. 2022.

"Recurrent Neural Networks (RNN) With Keras | TensorFlow Core." *TensorFlow*, www.tensorflow.org/guide/keras/rnn. Accessed 10 Dec. 2022.

"Sklearn.Linear_Model.LogisticRegression." *Scikit-learn*, scikit-learn/stable/modules/generated/sklearn.linear_model.LogisticRegression.html. Accessed 10 Dec. 2022.

"Sklearn.Neighbors.KNeighborsClassifier." *Scikit-learn*, scikit-learn/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html. Accessed 10 Dec. 2022.

"TED Talks Web-scraped Dataset." *TED Talks Web-scraped Dataset | Kaggle*, /datasets/jeniagerasimov/ted-talks-info-dataset. Accessed 10 Dec. 2022.