Capstone Project

TED Talk views prediction using regression

Arvind Krishna, Keshav Sharma, Jayesh Panchal, Sahil Ahuja
Data science students
Cohort- Boston, Alma Better

Introduction

TED stands for Technology, Entertainment and Design.

A TED talk refers to a recorded public-speaking presentation that was originally given at the main TED (technology, entertainment and design) annual event or one of its many satellite events around the world. TED is a nonprofit devoted to spreading ideas, usually in the form of short, powerful talks, often called "TED talks." These could be a motivational video, awareness, technical, Science entertainment, or any other genre.

TED is devoted to spreading powerful ideas on just about any topic. Founded in 1984 by Richard Salman as a nonprofit organization that aimed at bringing experts from the fields of Technology, Entertainment, and Design
together, TED Conferences have gone on to become the Mecca of ideas from virtually all walks of life. As of 2015, TED and its sister TEDx chapters have published more than 2000 talks for free consumption by the masses.

1. Objective

The main objective of this project is to build an optimal predictive model, which could help in predicting the views of the videos uploaded on the TEDx platform.

In the Ted talks dataset, the following attributes/features are available for us to begin with:

- talk_id : Talk identification number provided by TED
- title : Title of the talk
- peaker_1 : First speaker in TED's speaker list.
- speakers : Speakers in the talk
- occupations : Occupations of the speakers
- about_speakers : Blurb about each speaker.
- Views(Output Variable) : Count of views .
- recorded_date : Date the talk was recorded.
- published_date : Date the talk was published to TED.com
- event : Event or medium in which the talk was give

- native_lang : Language the talk was given in
- available_lang : All available languages (lang_code) for a talk
- comments : Count of comments
- duration : Duration in seconds
- topics : Related tags or topics for the talk
- related_talks : Related talks (key='talk_id', value='title')
- url : URL of the talk
- description : Description of the talk
- transcript : Full transcript of the talk

2. Exploratory Data Analysis

In terms of EDA, there is quite a bit of analysis and derivations that we have discovered and worked with.
In the interest of keeping this document precise and on-the-dot, We shall cover the most significant logical findings of EDA under “Feature engineering”. A more detailed approach of the EDA is available in the associated Python Notebook. Kindly refer to that, for more clarity.

3. Feature Engineering

All machine learning algorithms use some input data to create outputs. This input data comprises of various features, which are usually in the form of structured columns. Algorithms require features with some specific characteristics to work properly. Here, the need for feature engineering arises. Feature engineering mainly have two goals:

* Preparing the proper input dataset, compatible with the machine learning algorithm requirements.

* Improving the performance of machine learning models.

We'll try adding and removing some features in this section in order to make a perfect data matrix we can pass to a machine learning model. We will try to interpret categorical features as numeric, so as to be passed to the ML models.

From the **published_date** column, we can easily extract the year of uploading of TED talk.

From the year, we can extract more information and fetch the **recency** and **per annum views** for a video. The recency is the number of years a video has been available on the website for.

We can calculate recency by subtracting the upload year from 2021 (we only have TED talks data upto year 2020, so we are considering videos uploaded in 2020 been available on website for at least a year.

Per annum views of a video can be found by dividing the views in total by the recency of the video (number of years it has been online).

The “**available_lang**” feature holds all languages the video is available in. With that, we can count the total languages for a video and hold that count as a feature. We are going with the intuition that if a video is available in many languages, it'd reach a much wider audience.

Similarly, we can calculate the number of topics covered by a video, by dissecting the “**topics”** feature.

We then give weights to the video based on the topics covered. This can be achieved by calculating the number of videos available with that topic. The popular topics would have a high frequency and we can use that frequency to put as weight of the topic.

For **description** feature, we are doing a word count, with the hypothesis that a video with more elaborated description would cover more tags and topics that can attract more viewers.

There's a column for **related_talks** which holds the dictionary of related videos for any video. Using this information, we derive a feature that will hold the average views out of all the related videos for a particular talk...

4.Model Selection

Now it's time to implement the Machine Learning models and check the accuracy of each model to point out the best one out of all. In this project we have tried and tested 7 machine learning algorithms to predict the target variable, post which we then apply optimization techniques on the one that gives best r2 score out of all.

Following algorithms have been used for predictions:-

- **Linear Regression**
  Linear Regression is a supervised machine learning algorithm where the predicted output is continuous and has a constant slope. It's used to predict values within a continuous range, rather than trying to classify them into categories. There are two main types:

  Simple regression:

  Simple linear regression uses traditional slope-intercept form, where m and b are the variables our algorithm will try to "learn" to produce the most accurate predictions. x represents our input data and y represents our prediction.

  $y=mx+b$

  Multivariable regression:

  A more complex, multi-variable linear equation might look like this, where w represents the coefficients, or weights, our model will try to learn.

  $f(x,y,z)=w1x+w2y+w3z$

- **Lasso Regression**
  Lasso regression stands for Least Absolute Shrinkage and Selection Operator. It adds penalty term to the cost function. This term is the absolute sum of the coefficients. As the value of coefficients increases from 0 this term penalizes, cause model, to decrease the value of coefficients in order to reduce loss. The difference between ridge and lasso regression is that it tends to make coefficients to absolute zero as compared to Ridge which never sets the value of coefficient to absolute zero.

  Limitation of Lasso Regression:
  Lasso sometimes struggles with some types of data. If the number of predictors (p) is greater than the number of observations (n), Lasso will pick at most n predictors as non-zero, even if all predictors are

relevant (or may be used in the test set). If there are two or more highly collinear variables then LASSO regression select one of them randomly which is not good for the interpretation of data

- **Ridge Regression**
  In Ridge regression, we add a penalty term which is equal to the square of the coefficient. The L2 term is equal to the square of the magnitude of the coefficients. We also add a coefficient lambda to control that penalty term. In this case if lambda is zero then the equation is the basic OLS else if lambda > 0 then it will add a constraint to the coefficient. As we increase the value of lambda this constraint causes the value of the coefficient to tend towards zero. This leads to tradeoff of higher bias (dependencies on certain coefficients tend to be 0 and on certain coefficients tend to be very large, making the model less flexible) for lower variance.

  Limitation of Ridge Regression: Ridge regression decreases the complexity of a model but does not reduce the number of variables since it never leads to a coefficient been zero rather only minimizes it. Hence, this model is not good for feature reduction.

- **Elastic Net Regression**
  Sometimes, the lasso regression can cause a small bias in the model where the prediction is too dependent upon a particular variable. In these cases, elastic Net is proved to better it combines the regularization of both lasso and Ridge. The advantage of that it does not easily eliminate the high collinearity coefficient.

- **Decision Tree**
  Decision trees can be used for classification as well as regression problems. The name itself suggests that it uses a flowchart like a tree structure to show the predictions that result from a series of feature-based splits. It starts with a root node and ends with a decision made by leaves.

  Decision trees are upside down which means the root is at the top and then this root is split into various several nodes. Decision trees are nothing but a bunch of if-else statements in layman terms. It checks if the condition is true and if it is then it goes to the next node attached to that decision.

  In a Decision Tree diagram, we have:

  Root Node: The first split which decides the entire population or sample data should further get divided into two or more homogeneous sets. In our case, the Outlook node.

  Splitting: It is a process of dividing a node into two or more sub-nodes.

  Decision Node: This node decides whether/when a sub-node splits into further sub-nodes or not. Here we have, Outlook node, Humidity node, and Windy node.

  Leaf: Terminal Node that predicts the outcome (categorical or continuous value). The coloured

nodes, i.e., Yes and No nodes, are the leaves.

- **Random Forest Regression**
  Random Forest is a technique that uses ensemble learning, that combines many weak classifiers to provide solutions to complex problems.

  As the name suggests random forest consists of many decision trees. Rather than depending on one tree it takes the prediction from each tree and based on the majority votes of predictions, predicts the final output.

  Random forests use the bagging method. It creates a subset of the original dataset, and the final output is based on majority ranking and hence the problem of overfitting is taken care of.

- **XGB Regression**
  XGBoost is a powerful approach for building supervised regression models. The validity of this statement can be inferred by knowing about its (XGBoost) objective function and base learners. The objective function contains loss function and a regularization term. It tells about the difference between actual values and predicted values, i.e how far the model results are from the real values. The most common loss functions in XGBoost for regression problems is reg:linear, and that for binary classification is reg:logistics. Ensemble learning involves training and combining individual models (known as base learners) to get a single prediction, and XGBoost is one of the ensemble learning methods. XGBoost expects to have the base learners which are uniformly bad at the remainder so that when all the predictions are combined, bad predictions cancels out and better one sums up to form final good predictions.

5.Model Optimization

From the Scores we have seen earlier Random forest has given by far the accuracy in predicting the target variable. Hence we have selected that for the optimization section.

We have implemented Gridsearch Cross-Validation upon Random forest, to fine-tune prediction results from our model. To achieve that we'll use Grid Search CV that will help us find best hyperparameters values for our Random Forest model.

Grid Search uses different combinations of all the specified hyperparameters and their values and calculates the performance for each combination and selects the best value for the hyperparameters. This makes the process time-consuming and expensive based on the number of hyperparameters involved.

6. Conclusion:

We were able to see that the linear algorithms were not performing optimally even with Gradient Boosting optimization, and the tree-based algorithms performed significantly better.

Out of the tree-based algorithms, the Random Forest Regressor was providing an

optimal solution towards achieving our Objective. We were able to achieve an R2 score of 0.99 in the train split, and 0.92 in the test split. We also noticed that even in the case of Decision tree, we were able to achieve an R2 score of 0.89 in the test split.

We then implemented Grid Search Cross Validation on the Random Forest Regressor, to further optimize the model, and were able to achieve an R2 score of 0.99 in the train split, and 0.95 in the test split.

Finally, we conclude Random Forest with GridSearchCV to be the best model to achieve our objective.

7. References-

- **Python Pandas Documentation** https://pandas.pydata.org/pandas-docs/stable
- **Python MatPlotLib Documentation** https://matplotlib.org/stable/index.html
- **Python SkLearn Documentation** https://scikit-learn.org/stable
- **TED website** https://ted.com