TED Talk Views Prediction

Aamir Sohail
Data science trainee,
AlmaBetter, Bangalore

Abstract:

TED is devoted to spreading powerful ideas on just about any topic. These datasets contain over 4,000 TED talks including transcripts in many languages. 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 and its speaker list boasts of the likes of Al Gore, Jimmy Wales, Shahrukh Khan, and Bill Gates.

Keywords: machine learning, views prediction, TED Talk, Regression model

1.Problem Statement

The main objective is to build a predictive model, which could help in predicting the views of the videos uploaded on the TEDx website.

Dataset Information

- Number of instances: 4,005
- Number of attributes: 19

Features information:

The dataset contains features like:

- **talk_id**: Talk identification number provided by TED
- **title**: Title of the talk
- **speaker_1**: First speaker in TED's speaker list
- **all_speakers**: Speakers in the talk
- **occupations**: Occupations of the speakers
- **about_speakers**: Blurb about each speaker
- **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 given
- **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

Target Variable :

- **Views**: The number of views for each talk

2. Introduction

Ted Talks are one of the institutions that are constantly using Machine Learning algorithms to optimize the number of views the videos receive. They do this by understanding the dependency of views with other relevant features to increase the viewers satisfaction by recommending videos according to the average views that have been affected by features like topics,comments, etc.

In order to help Ted Talks to increase their views we are helping them find relevant features and their dependencies on the views.

3. Steps involved:

- **Data Collection**

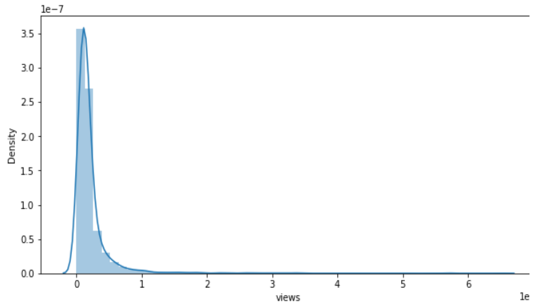
  To proceed with the problem dealing first we will load our dataset that is given to us in .csv file into a dataframe.Mount the drive and load the csv file into a dataframe.

- **Exploratory Data Analysis**

  After loading the dataset we looked for duplicate values in the 'talk_id' column. There were none. So We performed EDA by comparing our target variable that is Views with other independent variables. This process helped us figuring out various aspects and relationships among the target and the independent variables. It gave us a better idea of which feature behaves in which manner compared to the target variable.

1. **Numerical Variables:**
   - Talk_id
   - Views
   - Comments
   - duration
2. **Textual Variables:**
   - Title
   - Speaker_1
   - Recorded_date
   - Published_date
   - Event
   - Native_lang
   - Url
   - Description
3. **Dictionaries:**
   - Speakers
   - Occupations
   - About_speakers
   - Related_talks
4. **List:**
   - topics

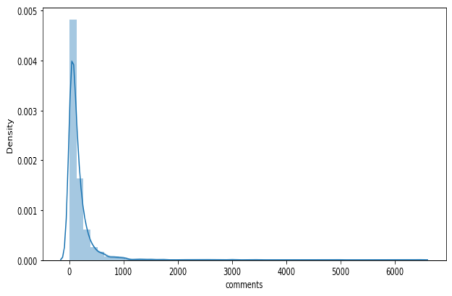
Out of all the continuous variables, 'views' is the target variable.



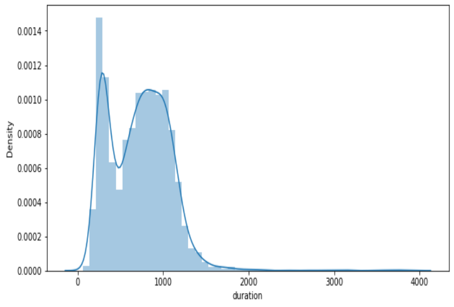
The target variable 'views' was a skewed variable.

The other continuous variables have distributions as:

1. Comments



2. Duration



All of the data had very skewed continuous variable distributions.

- **Null values Treatment**

  Our dataset contains around 400 null values which might tend to disturb our mean absolute score hence we have performed KNN nan value imputer for numerical features and replaced categorical features nan values with the value 'Other'. We chose to impute nan values and not drop them due to the size of the data set

- **Encoding of categorical columns**

  We used Target Encoding for replacing the values of categorical variables with the mean of the views. This was done to not increase the dimensions to the data set while also keeping the relationship of variables with views into consideration.

- **Feature Selection**

  For Feature Selection we have done the following: we have introduced new numerical features from the categorical features,combined features and also we have used f_regression in which we have taken the features with the maximum f-scores.

- **Outlier Treatment**

  We have done outlier treatment on variables like duration and occupation. This was done by replacing outliers with the extreme values at the first and third quartiles. We have done outlier treatment to prevent high errors that were influenced by outliers.

- **Fitting different models**

  For modelling we tried various regression algorithms like:

1. **XGBoost Regressor**

2. **Extra Trees Regressor**
3. **Random Forest Regressor**

- **Tuning the hyperparameters for better accuracy**

  Tuning the hyperparameters of respective algorithms is necessary for less error values,regularization and to avoid overfitting in case of tree based models.

4.1. Algorithms:

We have used only non-parametric models for prediction because two of the hypotheses such as linearity between output and input variables and errors normally distributed were not met.

1. XGBoost Regression:

Gradient boosting refers to a class of ensemble machine learning algorithms that can be used for classification or regression predictive modeling problems.Extreme Gradient Boosting, or XGBoost for short, is an efficient open-source implementation of the gradient boosting algorithm. It is computationally effectively faster with better model performance.

The Taylor approximation is given by:

$$L(y,p_i + O_v) = L(y,p_i) + gO_v + \frac{1}{2}hO_v^2$$

Expand the summation,

$$\sum_{i=1}^{n} L(y_i, p_i^0 + O_v) + \frac{1}{2}\lambda O_v^2$$

$$L(y_1, p_1^0 + O_v) + L(y_2, p_2^0 + O_v) + \cdots + L(y_n, p_n^0 + O_v) + \frac{1}{2}\lambda O_v^2$$

Plug in Taylor Approximation,

$$L(y_1,p_1^0) + g_1 O_v + \frac{1}{2}h_1 O_v^2 + L(y_2,p_2^0) + g_2 O_v + \frac{1}{2}h_2 O_v^2 + \ldots\ldots + L(y_n,p_n^0) + g_n O_v + \frac{1}{2}h_n O_v^2 + \frac{1}{2}\lambda O_v^2$$

This is the approximated equation.

XGboost can be optimized by fixing the number of trees, fixing learning rate,tuning gamma, tuning regularization and various hyper parameter tuning.

When we implemented this model we got the following scores :

MAE train: 211051.556655
MAE test: 228812.768108
R2_Score train: 0.866211
R2_Score test: 0.832566
RMSE_Score train: 403273.960146
RMSE_Score test: 451028.530993

.

2. Extra Trees Regressor:

Extremely Randomized Trees, or Extra Trees for short, is an ensemble machine learning algorithm.

Specifically, it is an ensemble of decision trees and is related to other ensembles of decision trees algorithms such as bootstrap aggregation (bagging) and random forest.

The Extra Trees algorithm works by creating a large number of unpruned decision trees from the training dataset. Predictions are made by averaging the prediction of the decision trees.

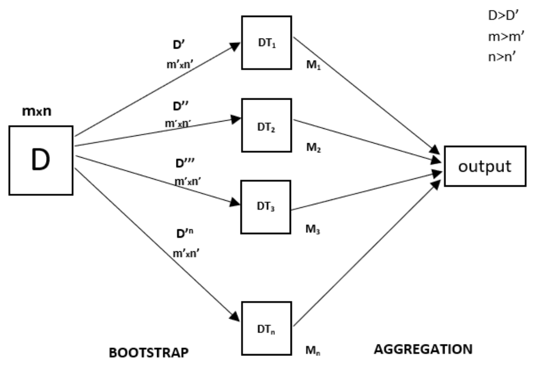
The random selection of split points makes the decision trees in the ensemble less correlated, although this increases the variance of the algorithm. This increase in variance can be countered by increasing the number of trees used in the ensemble.

We use the criterion as 'MAE' as it uses L1 regularization to select the median and selects the best features for reducing the mean absolute error. MAE is used as it is not influenced by outliers.
Scores obtained were:

MAE_train : 197961.601666
MAE_test: 195381.403519
R2_Score_train: 0.796220
R2_Score_test: 0.806408
RMSE_Score_train: 497703.788585
RMSE_Score_test: 484983.269104

3. Random Forest Regressor:

Every decision tree has high variance, but when we combine all of them together in parallel then the resultant variance is low as each decision tree gets perfectly trained on that particular sample data and hence the output doesn't depend on one decision tree but multiple decision trees. In the case of a classification problem, the final output is taken by using the majority voting classifier. In the case of a regression problem, the final output is the mean of all the outputs.



A Random Forest is an ensemble technique capable of performing both regression and classification tasks with the use of multiple decision trees and a technique called Bootstrap and Aggregation, commonly known as **bagging**. The basic idea behind this is to combine multiple decision trees in determining the final output rather than relying on individual decision trees.
Random Forest has multiple decision trees as base learning models. We randomly perform row sampling and feature sampling from the dataset forming sample datasets for every model. This part is called Bootstrap.

MAE train: 186638.460863
MAE test: 192011.316625
R2_Score train: 0.806400
R2_Score test: 0.803162
RMSE_Score_train: 485112.674688
RMSE_Score_test: 489031.640334

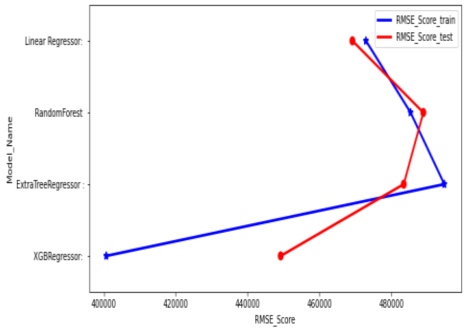
4.2. Model performance:

Model can be evaluated by various metrics such as:

1. **Root Mean Square Error-** Root Mean Square Error (RMSE) is the standard deviation of the residuals (prediction errors). Residuals are a measure of how far from the regression line data points are; RMSE is a measure of how spread out these residuals are. In other words, it tells you how concentrated the data is around the line of best fit.

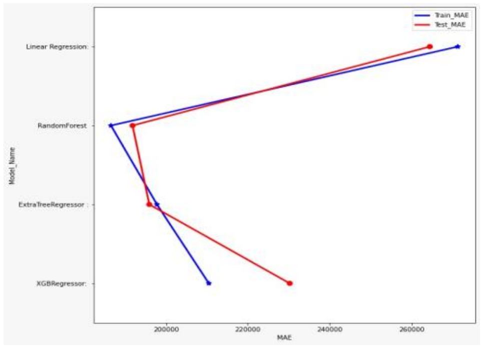
$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} (Predicted_i - Actual_i)^2}{N}}$$

It gets influenced by outliers.



2. **Mean Absolute Error-** *Mean Absolute Error* is a model evaluation metric used with regression models. The mean absolute error of a model with respect to a test set is the mean of the absolute values of the individual prediction errors on all instances in the test set. Each prediction error is the difference between the true value and the predicted value for the instance.

$$MAE = \overset{\text{Divide by the total number of data points}}{\boxed{\frac{1}{n}}} \underset{\text{Sum of}}{\Sigma} \underbrace{\left| \overset{\text{Actual output value}}{\boxed{y}} - \overset{\text{Predicted output value}}{\boxed{\hat{y}}} \right|}_{\text{The absolute value of the residual}}$$



We choose MAE and not RMSE as the deciding factor because of the following reasons:

1. RMSE is heavily influenced by outliers as the higher the values get the more the RMSE increases. MAE doesn't increase with outliers.
2. MAE is linear and RMSE is quadratically increasing.

4.3. Hyper parameter tuning:

Hyperparameters are sets of information that are used to control the way of learning an algorithm. Their definitions impact parameters of the models, seen as a way of learning, change from the new hyperparameters. This set of values affects performance, stability and interpretation of a model. Each algorithm requires a specific hyperparameters grid that can be adjusted according to the business problem. Hyperparameters alter the way a model learns to trigger this training algorithm after parameters to generate outputs.

We used Grid Search CV, Randomized Search CV and Bayesian Optimization for hyperparameter tuning. This also results in cross validation and in our case we divided the dataset into different folds. The best performance improvement among the three was by Bayesian Optimization.

1. **Grid Search CV-**Grid Search combines a selection of hyperparameters established by the scientist and runs through all of them to evaluate the model's performance. Its advantage is that it is a simple technique that will go through all the programmed combinations.
   The common hyperparameters which we extracted were n_estimators, max_depth, verbose=1 and cv = KFold.

2. **Randomized Search CV-** In Random Search, the hyperparameters are chosen at random within a range of values that it can assume. The advantage of this method is that there is a greater chance of finding regions of the cost minimization space with more suitable hyperparameters, since the choice for each iteration is random. The disadvantage of this method is that the combination of hyperparameters is beyond the scientist's control

3. **Bayesian Optimization-** Bayesian Hyperparameter optimization is a very efficient and interesting way to find good hyperparameters. In this approach, in naive interpretation way is to use a support model to find the best hyperparameters.A hyperparameter optimization process based on a probabilistic model, often Gaussian Process, will be used to find data from data observed in the later distribution of the performance of the given models or set of tested hyperparameters.
   As it is a Bayesian process at each iteration, the distribution of the model's performance in relation to the hyperparameters used is evaluated and a new probability distribution is generated. With this distribution it is possible to make a more appropriate choice of the set of values that we will use so that our algorithm learns in the best possible way.

5. Conclusion:

That's it! We reached the end of our exercise.
Starting with loading the data so far we have done EDA , null values treatment, encoding of categorical columns, feature selection and then model building.
In all of these models our errors have been in the range of 2,00,000 which is around 10% of the average views. We have been able to correctly predict views 90% of the time.
After hyper parameter tuning, we have prevented overfitting and decreased errors by regularizing and reducing learning rate. Given that only 10% is errors, our models have performed very well on unseen data due to various factors like feature selection,correct model selection,etc.

<u>Future work:</u>

1. We can do a dynamic regression time series modelling due to the availability of the time features.
2. We can improve the views on the less popular topics by inviting more popular speakers.
3. We can use topic modelling to tackle views in each topic separately.

References-

1. MachineLearningMastery
2. GeeksforGeeks
3. Analytics Vidhya