Prediction and Analysis on YouTube Trending Videos

Sheetal Padale, MSc Big Data Analytics, L00169921

***Abstract*— Social media is rapidly growing nowadays as it is simple and easy to use for creating video, pics and reels too. Amongst many social media accessible, YouTube and Netflix is the most commonly used platform as it is an international medium connected to various countries of the world. In the project with the help of YouTube data, we tried to implement two models using regression and ensemble methods. The aim of the project is to compare YouTube data of different countries and find videos which are trending nowadays, their likes, dislikes and many more with the help of different models. For Sklearn we used Elastic Net Regression and Adaboost Ensemble method and for Spark we have used Gradient-boosted tree regression and as a classification method, Multilayer Perceptron Classifier is used.**

***Index Terms*— YouTube, Elastic Net Regression, Adaboost, Gradient-boosted tree and Multilayer Perceptron Classifier**

I. Introduction

YOuTube is a video based social media which has been in existence having more than one million users worldwide which is nearly half of the internet users. YouTube forecasts range from Vlogs, Sports, Music, News and Information, Lifestyles, Movies, Gaming and many more. It has a facility for interaction socially so that views and opinions of the videos are shared. These views and opinions are based on sharing, comments, ratings, favourites, watch later and many more such features which either may be good or bad. In the project we have conducted analysis with the help of various models for prediction purposes. Some functions carried such as, • Comparing views, likes, dislikes against categories. • Distribution of dislikes against categories, Using boxplot and dislikes on log scale. • Calculate the number of videos published per day of the week. Which day of the week do people publish the most videos?

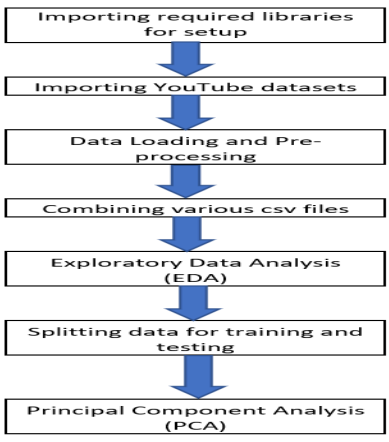

Fig 1: Pipeline Process for YouTube dataset.

II. Datasets

The focus of this assessment is to explore the popularity of the videos popular on platforms. The assessment will focus on YouTube 's data. There are different csv files in the dataset which are categorized by countries such as California, France, India and USA. The dataset has attributes listed as video id, trending date, title, channel title, category id, publish time, tags views, likes, dislikes, comment count, thumbnail link, comments disabled and description video error or removed ratings disabled. Further we work on Data loading and pre-processing for the further execution.

III. Data loading and Pre-processing

Data used should be identified as useful later should be recognised, cleaned, extracted into accessible format. Also unwanted components such as noise, replacing or removing vacant records, replacing vacant records with valuable information and so on must be performed. Then the outliers are also eliminated or reduced in order to obtain clean data. Data cleaning and transformation improves data processing efficiency which increases the possibility of the data analyst to provide exact information as demanded by the client. A new column country is added in the data frame and the ultimate data frame is named as 'combined$_data'$.

Further in order to eliminate the null rows we have used the data cleaning and for this we use the code.

combine.data = combined.data.dropna() combined.data.info ()

For removing the non-numeric columns, the code is combined.data.drop(['trending.date', 'title', 'channel.title', 'category.id', 'publish.time', 'tags', 'views', 'likes', 'dislikes', 'comment.count', 'thumbnail.link', 'description', 'publish.date'], axis = 1, inplace = True)

Further, Exploratory Data Analysis(EDA) is done which is the main step of fine tuning the dataset for the data

analyst to have the insights which in turn help for making productive decisions. In this stage the various entities of data such as columns and rows which are key characteristics are understood for purposes for accurate model selection. Further Principal Component Analysis (PCA) is performed which is an orthogonal transformation used to reduce number of dimensions of the dataset, as a first step fit a PCA model on your train set and then plot the explained.variance.ratio against the number of components to decide the number of components you should keep. Further Splitting the data into train and test by the use of sklearn's train.test.split library and split data into train and test sets, the split should be 80-20 meaning 80

IV. Methodology

A. Analysis with Sklearn (Colab)

1) Using Elastic Net Regression: We will implement regularization to avoid overfitting. you can try different regularization parameters, e.g., try LASSO (L1), Ridge (L2) and elastic net (combination of L1 and L2). ElasticNet uses linear regression model which uses both 1and 2 norm regularization of the coefficients. The combination allows for learning a sparse model where few of the weights are non-zero like Lasso, while still maintaining the regularization properties of Ridge. It is useful when we have multiple features having correlation amongst. It provides stability under rotation. Following figure 2 and 3 shows the execution of ElasticNet

```
from sklearn.linear_model import ElasticNet
from sklearn.metrics import mean_squared_error
# Add your code here
from sklearn.metrics import accuracy_score, mean_absolute_error, mean_squared_error

# Create linear regression object
enet = ElasticNet()

# Train the model using the training sets
enet.fit(x_train, y_train)

# Make predictions

y_predict_test = enet.predict(x_test)
y_predict_train= enet.predict(x_train)

# The coefficients
#print('Coefficients: \n', lin_R.coef_)

print('r Squared: %.2f'
      % enet.score(x_test, y_test))

print('mse_value of Test=',mean_squared_error(y_test, y_predict_test))
print('mse_value of Train=',mean_squared_error(y_train, y_predict_train))
mse_test= mean_squared_error(y_test, y_predict_test)
```

```
r Squared: 0.61
mse_value of Test= 1.3011642534643864
mse_value of Train= 1.2986992445255168
```

```
[ ] print('check_enet', (np.sqrt(mean_squared_error(y_test, y_predict_test))))

check_enet 1.1406858697574833
```

Fig 2: Importing Libraries and finding accuracy$_score, mean_absolute_error, mean_squared_error$.

```
from sklearn.model_selection import GridSearchCV

# Use grid search to tune the parameters:

parametersGrid = {"max_iter": [1000],
                  "alpha": [1.0],
                  "l1_ratio": np.arange(0.5)}

enet = ElasticNet()
grid = GridSearchCV(enet, parametersGrid, scoring='r2', cv=10)
grid.fit(x_train_Trans, y_train)
Y_pred = grid.predict(x_test_Trans)

# indx = np.isnan(x).any(axis=1)
# x_train = x_train[indx]
# y_train = y_train[indx]
```

```
[ ] print('check_enet', (np.sqrt(mean_squared_error(y_test, y_predict_test))))

check_enet 1.1406858697574833
```

Fig 3: Tuning the model.

2) Using Adaboost Regression: An AdaBoost [1] classifier falls as a meta-estimator beginning classifier fitting the original dataset and followed by additional copies on the similar dataset. Its weight is unfairly classified and adjusted such that followed classifiers aim towards complicated cases. Figure 4 shows the Adaboost classifier. Since we have continuous data which does not function with the help of classifiers so we had used regressor in the code.

Task 1.2 AdaBboost

```
from sklearn.model_selection import cross_val_score

from sklearn.ensemble import AdaBoostRegressor

clf = AdaBoostRegressor(n_estimators=100)
scores = cross_val_score(clf, x_train  , y_train, cv=5)
scores.mean()
```

```
0.8072663514655183
```

Fig 4 :Implementation of the Adaboost classifier.

B. Analysis with MLib (Pyspark)

1) Gradient-boosted tree regression (GBT): GBT utilizes an ensemble method to appraise the performance of the system. It is an iterative algorithm, combining simple parameterized functions having low performance (high prediction error) for the production of high and accurate prediction. GBT utilizes an ensemble of weak learners to boost performance. Figure 6. Shows the implementation of the Gradient-boosted tree regression (GBT).

Fig 5 :The implementation of the Gradient-boosted tree regression (GBT)

2) Multilayer Perceptron Classifier: It is a Machine Learning Tools used for tracking, Visualizing Experiments and Comparing purpose. It is Better Models Faster with Experiment Tracking, Dataset Versioning Model Management. Figure 7 shows the implementation of the Multilayer Perceptron Classifier

Fig 6 :The implementation of the Multilayer Perceptron Classifier.

V. Results and Conclusion

We deployed a YouTube dataset and implemented various models and their outcomes are displayed below.

1) Elastic Net Regression the output is: r Squared: 0.61 mse value of Test= 1.3011642534643864 mse value of Test= 1.3011642534643864 check lr 0.6679368826214105

2) Adaboost Regression : Mean = 0.8072663514655183

3) Gradient-boosted tree regression : Root Mean Squared Error (RMSE) on test data = 0.651205

4) Multilayer Perceptron Classifier: And also implemented Multilayer Perceptron Classifier

This assessment will enable us to achieve essential experience of machine learning using sklearn and scalable approach to machine learning based on park ML.

VI. References

[1] https://www.irjet.net/archives/V7/i8/IRJET-V7I8732.pdf

[2] https://scikit-learn.org/stable/supervised$_learning.html$

[3]https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=arnumber=9648486