YOUTUBE COMMENT SPAM PREDICTION

IMLA Mini Project

ABSTRACT

- Classifying YouTube comments as it attracts malicious users and is a challenge since messages are short and rife with slangs, symbols and abbreviations.
- Using Text classification and prediction model to predict YouTube comments are spam or not.
- Using bag-of-words and different machine learning algorithms for prediction
- Preprocessing the data to remove punctuations, plurals, to remove stop words, to record binary counts.
- Comparing various classification techniques and statistical analysis of results.

DATASET DESCRIPTION

Source:
This corpus has been collected using the YouTube Data API v3.

Data Set Information:
The table below lists the datasets:

Dataset --- YouTube ID -- # Spam - # Ham - Total
Psy ------- 9bZkp7q19f0 --- 175 --- 175 --- 350
KatyPerry - CevxZvSJLk8 --- 175 --- 175 --- 350
LMFAO ----- KQ6zr6kCPj8 --- 236 --- 202 --- 438
Eminem ---- uelHwf8o7_U --- 245 --- 203 --- 448
Shakira --- pRpeEdMmmQ0 --- 174 --- 196 --- 370

Attribute Information:
The collection is composed by one CSV file per dataset, where each line has the following attributes:

COMMENT_ID,AUTHOR,DATE,CONTENT,TAG

One example bellow:

z12oglnpoq3gjh4om04cfdlbgp2uepyytpw0k,Francisco Nora,2013-11-28T19:52:35,please like :D [Web Link],1

CONCLUSIONS

- The Random Forest and Bagging algorithms had the best performance in comparison to other classifier.
- The KNN classifier had a very poor performance.
- In comparison to CountVectorizer and TfidfVectorizer, CountVectorizer performed better in almost all algorithms
- Even with Cross Validation there was no significant improvement in the accuracy.

PREPROCESSING

The content column was preprocessed and converted into standard form for further processes. The following steps were performed :

STEPS	LIBRARY USED
1. Converting the comments to lowercase	String
2. Removing Punctuation	String
3. Removing stop words	Nltk
4. Convert raw comments to matrix of token counts	Sklearn
5. Convert raw comments to matrix of TF-IDF features	Sklearn
6. Splitting training and testing data	Sklearn

Before preprocessing :
Huh, anyway check out this you[tube] channel: kobyoshi02

After preprocessing :
huh anyway check youtube channel kobyoshi02

MACHINE LEARNING MODELS

The data was converted into numeric form using both CountVectorizer and TfidfVectorizer. The models were analyzed using data obtained by train test split method and cross-validation. The following list of models were used for classification and clustering.

Classification techniques used:
1. MultinomialNB Classifier
2. BernoulliNB Classifier
3. Decision Tree Classifier (criterion='entropy')
4. K-nearest Neighbor Classifier (n_neighbors=5)
5. Random Forest Classifier
6. Voting Classifier
7. Gradient Boosting Classifier

Clustering technique used:
1. KMeans Clustering (n_clusters=5)

MLA Name	Accuracy	
	Train Test Split	Cross Validation
MultinomialNB	0.88	0.85
BernoulliNB	0.84	0.80
Decision Tree	0.89	0.87
Random Forest	0.89	0.88
KNN	0.79	0.77
Bagging	0.89	0.88
Boosting	0.89	0.86

Table 1. Accuracy of Train Test Split data and Cross Validation data for CountVectorizer

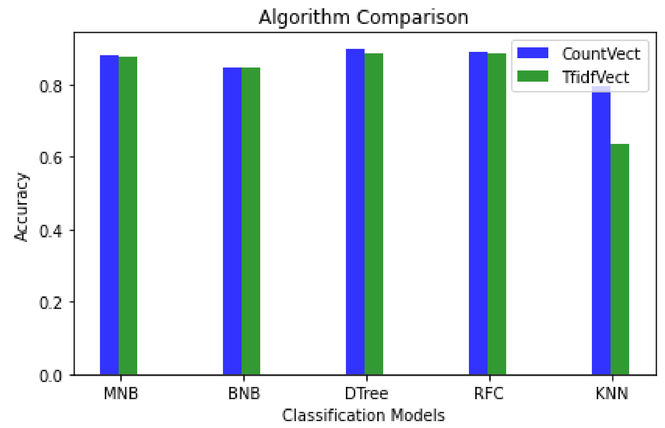

Fig 1. Comparison Graph of Accuracy vs Classification Models for CountVectorizer and TfidfVectorizer

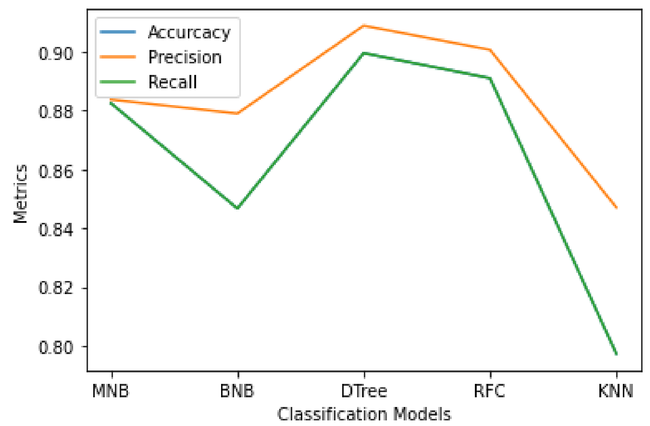

Fig 2. Accuracy, Precision and Recall for CountVectorizer and Models

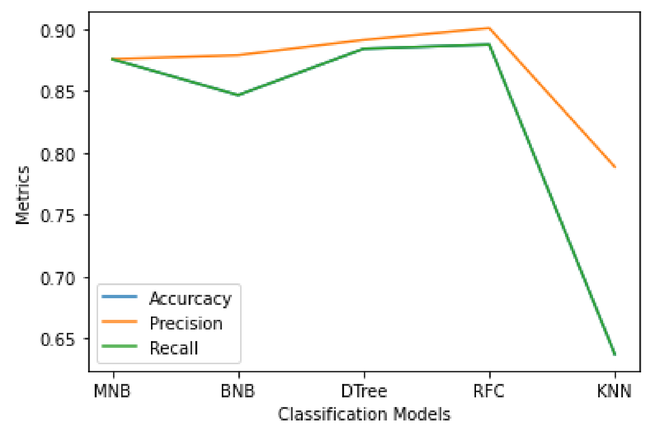

Fig 3. Accuracy, Precision and Recall for TfidfVectorizer and Models

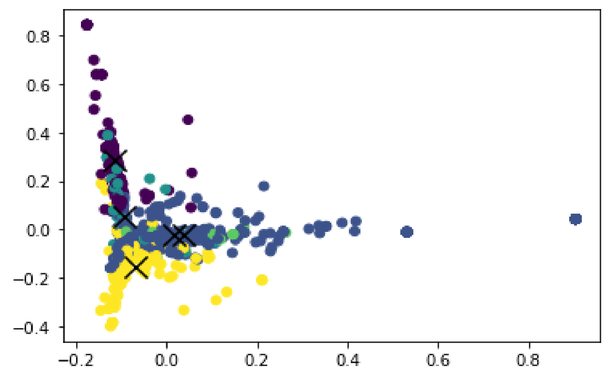

Fig 4. Visualization for clusters obtained by KMeans Clustering Algorithm

Created By:

Name : Karthik M Jain
Panel : B
Roll No.: PB46

Reference links:

1. https://stackoverflow.com
2. https://scikit-learn.org
3. https://sanjayasubedi.com.np/nlp/nlp-with-python-document-clustering/
4. https://matplotlib.org/