Report
Suman Kumar

List of All File attached:

youtube_scrap.ipynb => contain the initial code for scraping youtube
preprocessing.ipynb => contain code for preprocessing part
visualisation.ipynb => contain wordcloud for each category on preprocessed_data
naive_bayes_random_forest.ipynb => naive bayes and random forest techniques applied
CNN.ipynb => CNN applied on the preprocessed data
scrapped_youtube_data.csv => contain the initial scrapped data
preprocessed_data.csv => contain the preprocessed data

1. Scrap part

For youtube scrapping, BeautifulSoup is used. But using BeautifulSoup, only able to scrap only 8 to 9 video links only which is present on the first page of youtube initially.
So further searching over internet, found the youtube pagination and using WebCrawling, and using this, we can move from one page to another as buttons on the youtube page.

Few important code lines below

```
# using beautifulSoup
soup = BeautifulSoup(html, 'lxml')

#to get the button information present over page:
buttons = soup.findAll('a',attrs={'class':"yt-uix-button vve-check yt-uix-sessionlink yt-uix-button-default yt-uix-button-size-default"})

# to get the video id
video_id = soup.findAll('a',attrs={'class':'yt-uix-tile-link'})

# to get the video title
title = soup.h3.a.text

# to get the description
description = soup.find('div', class_="yt-lockup-description yt-ui-ellipsis yt-ui-ellipsis-2").text
```

2. Preprocessing

Initially each category contains 1700 datasets.

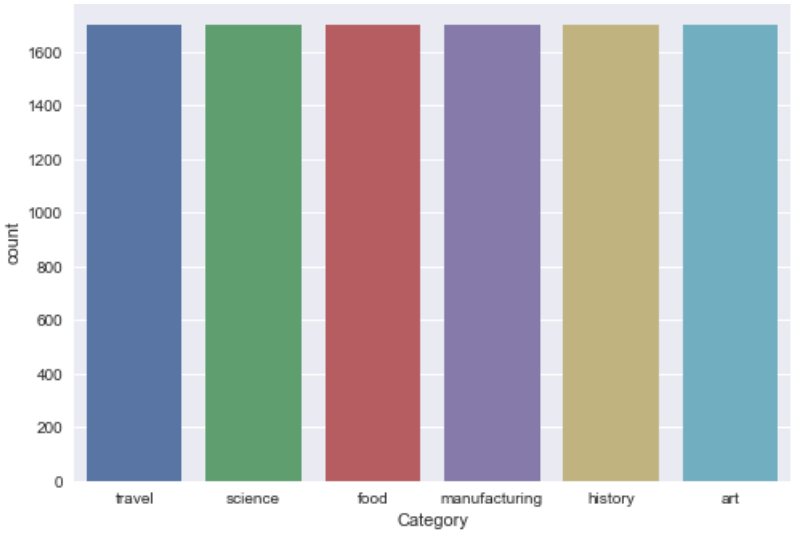


Illustration 1: initial count of each category type

So initial step was to remove the duplicate rows and remove the NAN rows. So after doing these two thing. So now each category contain around 350 images. This shows that my youtube scrapping code must have done pagination and in this case many linkes point to each other only.

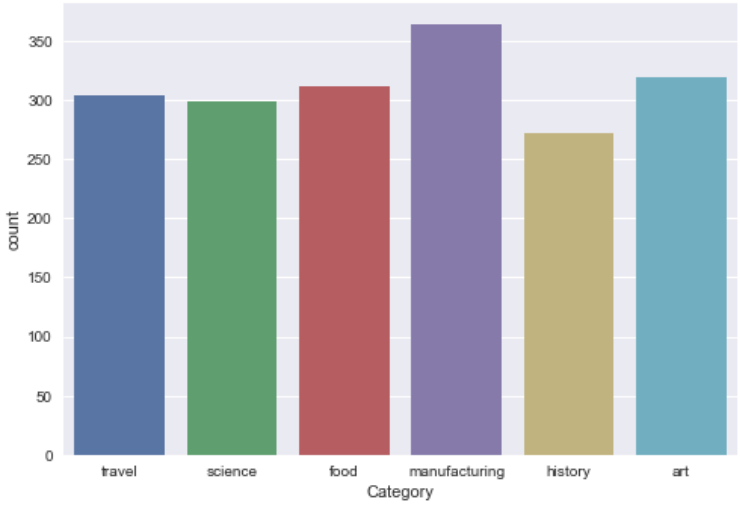


Illustration 2: count of category after duplicate and nan rows removal

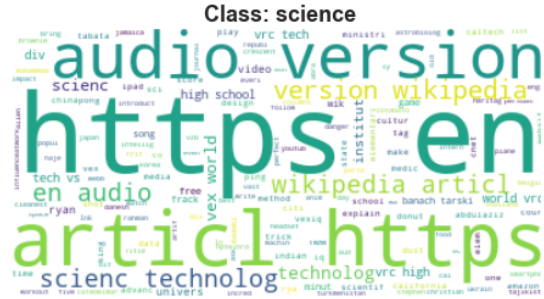
So now further steps follows as below

a. lowercase conversion
b. converting (i'm to i am, can't to can not etc....)
c. removal of character other than [a-zA-Z0-9]
d. removal of digit[0-9]
e. applied the PorterStemmer to bring word to its base form
f. removal of stop(applied the nltk library for this)

After doing the above preprocessing steps, save the data in csv file so that no need to run preprocessing step everytime while running the model.

3. Visualisation

Plot the wordcloud of each category.



4. Methods

a. Naive Bayes

Naive Bayes generally peroform better on textual data as due to to its simplicity and faster training. Also its assumption that each feature is indepenent of each other, this way it easily handle large datasets and quickly trained.

b. Random Forest

As random forest is ensembling methods, so using the many weak classifier can impove the performance of our model.

c. CNN

In recent time, CNN outperform on textual data and leave behind the Recurrent Neural network. So I choose the CNN as CNN is widely used on text data now a days.

5. Result

a. Naive Bayes

split the dataset into 80:20 train: test split
applied the tfidf for feature generation
accuracy: 71.9 %
precision(micro): 72 %
recall(micro): 72 %
f1 score(micro): 72 %

confusion matrix:

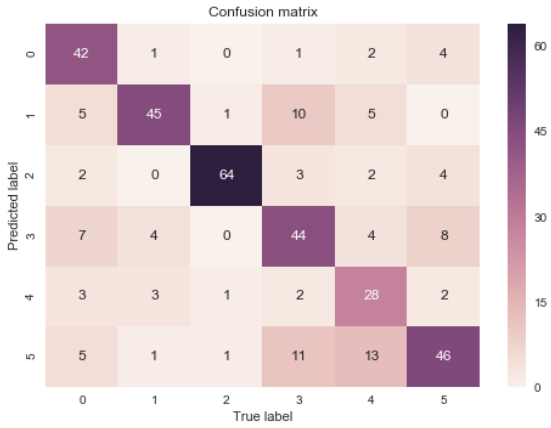


Illustration 3: confusion matrix(Naive Bayes)

The accuracy is around is 70% which is quite good with naive bayes.

b. Random Forest

split the dataset into 80:20 trian:test split
applied the tfidf for feature generation
accuracy: 66 %
precision(micro): 66 %
recall(micro): 66 %
f1 score(micro): 66 %

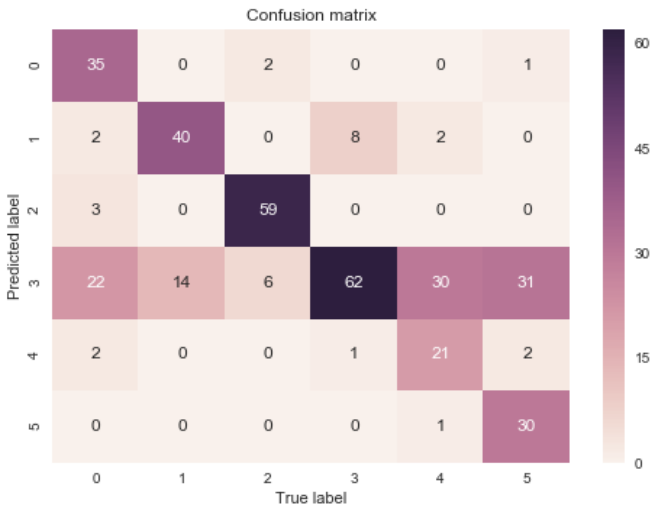


Illustration 4: confusion matrix(Random Forest)

With Random Forest, result is around 66 %, which is less. May be hyperparameter tunning is needed using GridSearchCV which can improve the result.

c. CNN

used the keras preprocessing library and embedding layer to handle the text part
applied the one hot encoder to label.
Each word embedding of size: 250
CNN architecture:
Conv1D(100) -> Conv1D(64) -> Flatten() -> Dense(64) -> Dense(32) -> Dense(6)
In between the above layer, DropOut layer is also used to avoid the overfitting.
Accuray: 16 %
precision: 16 %
recall: 100 %

With CNN accuracy and precision is very low as dataset is very small for such deep learning model. As neural network need lots of data, if datasets will be enough size, then CNN can give quite good result.