Nick Videtti

nvidetti@syr.edu

Assignment 5

Analysis on Web-Based Text Data Retrieved Using an API

NewsAPI, Pandas, Word Clouds, and SciKit Learn (Count Vectorizer and Multinomial Naive Bayes)

5/10/2023

| Predicting The Topic of News Articles | |
| --- | --- |
| **Introduction** | While it may not always seem like it given the continuous decline of the once abundant newspaper, there are an overwhelmingly large amount of news sources available to anybody by simply using the internet. In these seemingly countless news sources, there are a wide range of topics, many of which are specialties of the news source such as sports for ESPN or politics for CNN. It is intuitive that news articles from different topics are different by nature, but are they so different that any given article’s topic can be guessed? On a small scale, this paper will look to answer that question.  Samples of 100 news articles each from five different categories, for a total of 500, will be collected. The categories will be “sports”, “data”, “AI”, “food”, and “summer”. These should give a good mix of some topics that are closely related (data and AI, food and summer) and some topics that are not as similar (summer and data). These articles will be sourced from NewsAPI, which is an online collection of news articles from tens of thousands of sources that are both current and historic.  These 500 articles will be used to see whether they are different enough between the 5 topics to guess the topic of any given article. While this is a large sample of articles, it is important to note that this does not consider whether predicting any article that exists outside the five topics of this study can be predicted. Another important disclaimer is that the 100 articles from each topic were the 100 most recent available articles at the time of this study, so there can not be any conclusions about articles written before the last few weeks leading up to May 10th, 2023. |
| **Analysis** | The news articles will be pulled using an API, specifically NewsAPI as stated in the Introduction section. This will require registration for an API key in order to access the API. Once the API key is obtained, the articles will be pulled from NewsAPI one topic at a time. Each of these iterations will return a JSON object, but only the “articles” key will be necessary to use. Within the “articles” key, there is a nested JSON object for each article that contains many attributes such as source, author, date published, description, URL, article content, and more. For each article, the published date, URL, news source, and article content will be stored into a pandas DataFrame. The published date will also be used to store the published month and published year.  The article content will then be split into words using NLTK’s word tokenizer. From there, the words will be converted to lowercase and filtered down using regular expressions to only words with exclusively letters. Stopwords will also be removed using NLTK’s English stopwords. Finally, these filtered words will be converted back into a string of the words and will be saved into a new column in the dataframe called “Cleaned Content”, which will in effect replace the original “Content” column.  With these data, word clouds will be created for each topic and SciKit Learn’s Multinomial Naive Bayes algorithm will be used to train and test a model for predicting article topics. Both of these tasks will require creating a dataframe of word count vectors using the cleaned content. This will be done using SciKit Learn’s Count Vectorizer. The corresponding article topic will also be in each row of this word count vector dataframe.  Both the main dataframe and the count vectors will then be saved to csv files so they are available to view and/or use for other analysis. Those same csv files will then be imported back into the Python session and used for the word cloud generation and the Multinomial Naive Bayes modeling.  The word clouds will be generated using WordCloud and will be visualized using PyPlot. All 5 word clouds, one for each topic, will be shown at the same time.  SciKit Learn’s Multinomial Naive Bayes will be used to do the final analysis of the study and answer the main question of this study. The model will be trained and tested using the holdout method with a training proportion of 0.7. From there, predictions will be made, a confusion matrix will be generated, and the model accuracy will be evaluated. |
| **results** | The results of the data processing and analyses described in the “Analysis” section were as follows.  **Part of Nested JSON for First Article Retrieved**    **Main DataFrame**      **Labeled Word Count Vectors DataFrame**      **First Few Rows of Exported Data - Main DataFrame**    **First Few Rows and First Few Columns of Exported Data - Word Count Vectors**    **Word Clouds**            **Multinomial Naive Bayes Model - Confusion Matrix and Accuracy Score** |
| **conclusions** | After going through all 500 news articles, they were predicted to be the correct category roughly 63% of the time. As stated in the “Introduction” section, there are certainly many disclaimers to the scalability and usefulness of the findings from this one specific study, but 63% is actually a reasonably good accuracy score. While it is close to 50%, the number to really be thinking about for a benchmark is 20%. This is because a random guess between 5 topics has a 20% chance of being correct. All that said, it seems there are definitely ways to make a relatively good guess at a news article’s topic in this example, and it would certainly be interesting to use more news articles and test whether this is also likely to be the case for any given news article from any given topic. |