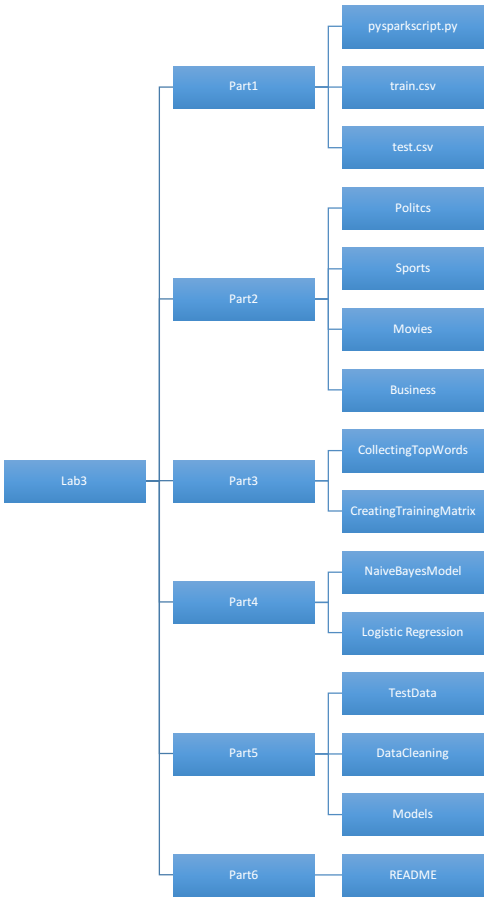
Mahalakshmi Padma Sri Harsha Maddu – 50246769 – mmaddu
Parikshit Sunil Deshmukh – 502476449 - pdeshmuk

LAB3: DATA AGGREGATION, BIG DATA ANALYSIS AND VISUALIZATION: B. RAMAMURTHY

Directory Structure:



Part1: Understanding Apache Spark

- Apache Spark has been downloaded and installed in MacOS and Jupyter environment.
- The titanic data analysis is done using Python.

The accuracy received for the titanic data analysis is:
Logistic Regression: 0.8369
Decision Tree: 0.77238
RandomForest: 0.85812

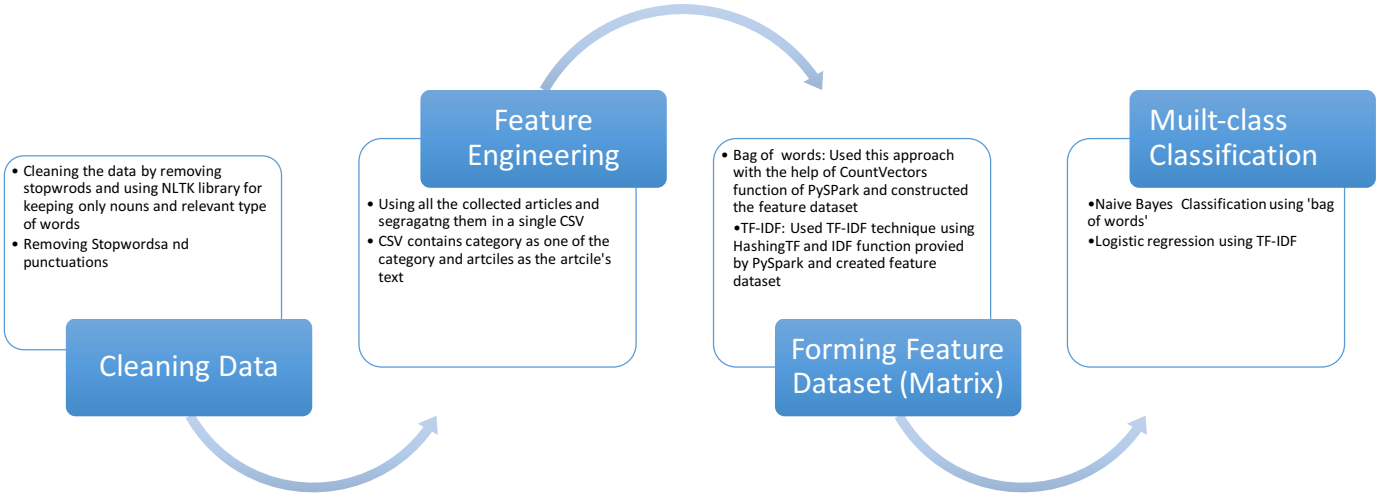
Output:

```
2018-05-11 16:15:29 INFO  MapPartitionsRDD:54 - Removing RDD 328 from persistence list
2018-05-11 16:15:29 INFO  BlockManager:54 - Removing RDD 328
{'LogisticRegression': 0.8369523688232297, 'DecistionTree': 0.7723828323993885, 'RandomForest': 0.8581253183902191}
2018-05-11 16:15:29 INFO  SparkContext:54 - Invoking stop() from shutdown hook
```

Part2: Collecting the data and cleaning

- Articles from four different categories are collected from NYTimes Articles API.
- The four categories are Sports, Politics, Movies and Business.
- Around 200 articles for each category are collected.
- Although web scraping and cleaning to an extent is done in this part, tokenizing and removing stop words is done in this in the next section during tokenizing.
- After collecting the data for the Bag of Words and TF-IDF approach we segregated all the data into single big CSV which can be called as a raw feature set containing rows of labels against their respective articles.

Data Pipeline:



Part3: Feature Engineering

1st Approach: Creating Document Featue Matrix

- We need features to train the model.
- The top 20 most frequently occurring words for each category are taken as features.
- There are a total of 80 features now.
- A training matrix is created by passing each of the article to give frequency of the 80 features.
- Now, we have a matrix of rows-with total number of articles collected and columns – with the number of features.

- The contents of the matrix will give how many times each of the 80 features are occurring in each article.
- After creating the required training matrix, we convert the csv file into txt file of "libsvm" format to apply classification models.

Another Approach:
We first tokenize the articles data and forms list of words for each articles
We then remove stop-words and other irrelevant words using NLTK library and other functions.

- Bag Of Words:
    - In this approach we use the CSV file formed as a raw feature set to use for this step.
    - This techniques gives the count of the words according to their frequency and the StringIndexer encodes a string column of labels to a column of label indices. The indices are ordered by label frequencies and we have most frequent label gets index 0.
    - So this approach gives us a final feature matrix which can be directly used for Machine Learning models for the classification purpose
- TF-IDF:
    - Term Frequency-Inverse Document Frequency- This approach is one of the intelligent way of finding weightage of the words within corpus and to form the feature dataset
    - We used HashingTF and IDF fucntions from Spark ML Library to accomplish this and formt he final feature dataset

Part4: Multiclass Classification:

- For this step, classification methods namely Naïve Baye's Classifier and Logistic Regression are used.
- We divided the dataset into training ad tsting in ration 70:30 and used above two models to classify he artciles
- We used Naïve Bayes classifier with feature dataset that is formed using bag of words tecquniues and for logistic regression we used feature data formed by TF-IDF method.
- After training the models we tested them on unseen test dataset where models classified the articles and accuracies are as below:

Approach1:
Naïve Bayes Classifier Accuracy Received: 99%
Logistic Regression Accuracy Received: 99%

Approach2:
Naïve Bayes Classifier Accuracy Received: 87%
Logistic Regression Accuracy Received: 85%

Part5: Testing the model accuracy with unknown data.

Naïve Bayes Classifier Accuracy Received: 0.58

Logistic Regression Accuracy Received: 0.544