**NYTIMES articles and Logistic Regression:**

Environment:

1. OS - MacOS
2. Spark installation for MacOS
3. Language used - Python

Commands:

1. Navigate to the folder with the file myNaive.py
2. Run the following code on terminal/cmd:

**spark-submit myNaive.py**

**Data Collection:**

To collect data, we used the NYTimes api.

We stored the articles for the following categories in their respective directories -

1. Media
2. Business
3. Sports
4. Politics

Each of these directories have each article stored in separate .txt files.

Used jupyter to collect data files.

**Data consolidation:**

To enable further computations on data, all the articles are consolidated into one csv file. The structure of the file is the following:

Column 1 - The article text

Column 2 - Label of the article [1 - Sports, 2 - Business, 3 - Media, 4 - Politics]

The file is called final.csv

**Data cleaning and Spark processing for Logistic Regression:**

Input - final.csv

The input is first loaded in an RDD. This RDD is then converted to a Spark dataframe, with two columns, ‘FileContent’ and ‘label’.

Then, the following steps are done -

1. The FileContent data is Tokenised using RegexTokenizer. This will help in filtering stop words in the next step.
2. The Stop words are removed, using nltk package.
3. The term frequence (TF) of the FileContent are calculated, using the Spark ML package. The number of features are set to 20. This means that each token is hashed to one of the 20 hashcodes. And TF for each hash is calculated.
4. The Inverse Document Frequency is calculated, using the Spark ML package.
5. Each word then has a TF-IDF score.
6. The data is split into train and test data
7. The logistic regression model is made and training data
8. To evaluate the accuracy of the model, the test data is fit on the model. And the accuracy is calculated using the evaluator.

The following part is done for testing the model on unseen/new articles:

1. The steps 1 to 5 are followed as before.
2. The data is then fit in the model from the previous part and the accuracy is calculated.

Following are the results:

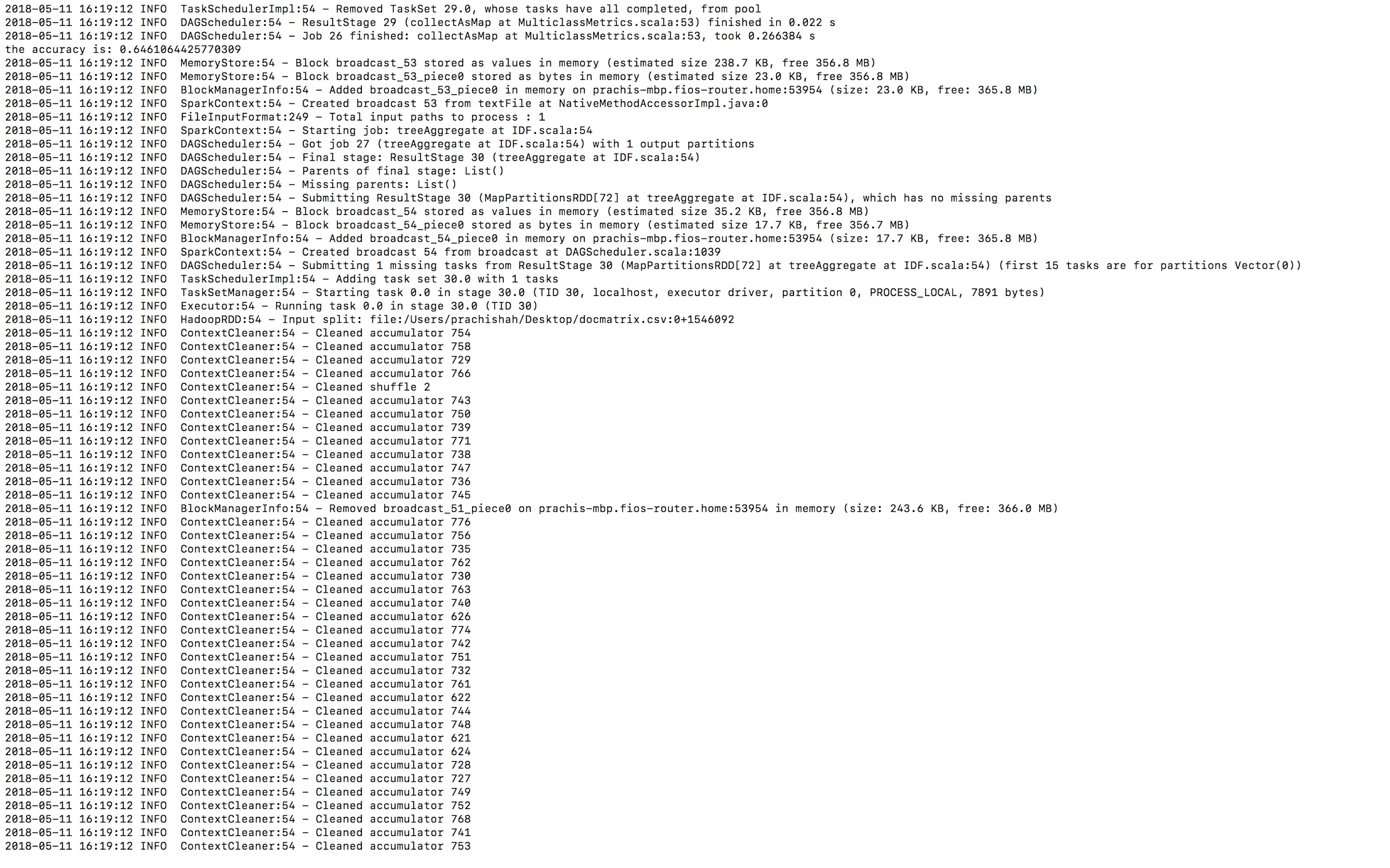


Fig1 - Accuracy for testing data

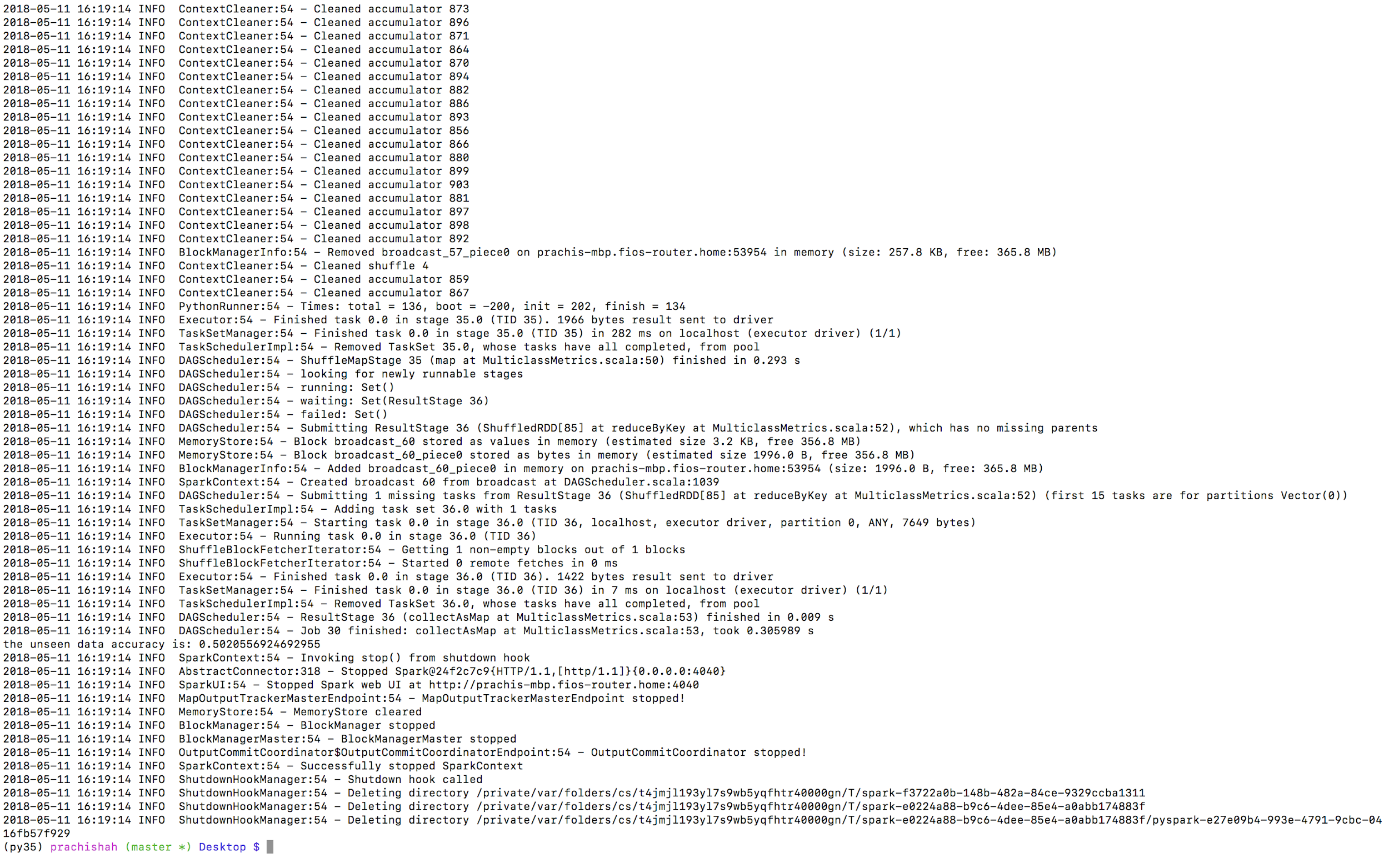


Fig2 - Accuracy for Unseen data