Training of Document Categorizer using Naive Bayes Algorithm in OpenNLP

Training of Document Categorizer using Naive Bayes Algorithm in OpenNLP

In this Apache OpenNLP Tutorial, we shall learn how to build a model for document classification with the Training of Document Categorizer using Naive Bayes Algorithm in OpenNLP.

Document Categorizing or Classification is requirement based task. Hence there is no pre-built models for this problem of natural language processing in Apache openNLP.

In this tutorial, we shall train the Document Categorizer to classify two categories : Thriller, Romantic. The categories chosen are movie generes. The data for each document is the plot of the movie.

Following are the steps to train Document Categorizer that uses Naive Bayes Algorithm for creating a Model :

- **Step 1** : Prepare the training data.
  The training data file should contain an example for each observation or document with the format : Category followed by data of document, seperated by space.
  For example, consider the below line which is from the training file :

  ```
  Thriller John Hannibal Smith Liam Neeson is held captive in Mexico
  ```

  Here ,
  Category is “Thriller”
  Data of the document is “John Hannibal Smith Liam Neeson is held captive in Mexico”.

  Find the complete training file used in the example, here en-movie-category.

- **Step 2** : Read the training data file.

  ```
  InputStreamFactory dataIn = new Markab
  ObjectStream lineStream = new PlainTex
  ObjectStream sampleStream = new Docume
  ```

- **Step 3** : Define the training parameters.

  ```
  TrainingParameters params = new Trainin
  params.put(TrainingParameters.ITERATIO
  ```

```
params.put(TrainingParameters.CUTOFF_PA
params.put(AbstractTrainer.ALGORITHM_PA
```

- **Step 4** : Train and create a model from the training data and defined training parameters.

```
DoccatModel model = DocumentCategorizer
```

- **Step 5** : Save the newly trained model to a local file, which can be used later for predicting movie genere.

```
BufferedOutputStream modelOut = new Buf
model.serialize(modelOut);
```

- **Step 6** : Test the model for a sample string and print the probabilities for the string to belong to different categories. The method DocumentCategorizer.categorize(String[] wordsOfDoc) takes words of a document as an argument in the form of an array of Strings.

```
DocumentCategorizer doccat = new Docume
double[] aProbs = doccat.categorize("Af
```

The complete program is provided in the following java file:

DocClassificationNai ... Java

```
1  import java.io.BufferedOutputStream;
2  import java.io.File;
3  import java.io.FileOutputStream;
4  import java.io.IOException;
5
```

```
6  import opennlp.tools.doccat.DoccatFactor
7  import opennlp.tools.doccat.DoccatModel;
8  import opennlp.tools.doccat.DocumentCate
9  import opennlp.tools.doccat.DocumentCate
10 import opennlp.tools.doccat.DocumentSamp
11 import opennlp.tools.doccat.DocumentSamp
12 import opennlp.tools.ml.AbstractTrainer;
13 import opennlp.tools.ml.naivebayes.Naive
14 import opennlp.tools.util.InputStreamFac
15 import opennlp.tools.util.MarkableFileIn
16 import opennlp.tools.util.ObjectStream;
17 import opennlp.tools.util.PlainTextByLin
18 import opennlp.tools.util.TrainingParame
19
20 /**
21  * oepnnlp version 1.7.2
22  * Training of Document Categorizer usin
23  * @author www.tutorialkart.com
24  */
25 public class DocClassificationNaiveBayes
26
27     public static void main(String[] arg
28
29         try {
30             // read the training data
31             InputStreamFactory dataIn =
32             ObjectStream lineStream = ne
33             ObjectStream sampleStream =
34
35             // define the training param
36             TrainingParameters params =
37             params.put(TrainingParameter
38             params.put(TrainingParameter
39             params.put(AbstractTrainer.A
40
41             // create a model from trani
42             DoccatModel model = Document
43             System.out.println("\nModel
44
45             // save the model to local
46             BufferedOutputStream modelOu
47             model.serialize(modelOut);
48             System.out.println("\nTraine
49
50             // test the model file by su
51             DocumentCategorizer doccat =
52             String[] docWords = "Afterwa
53             double[] aProbs = doccat.cat
```

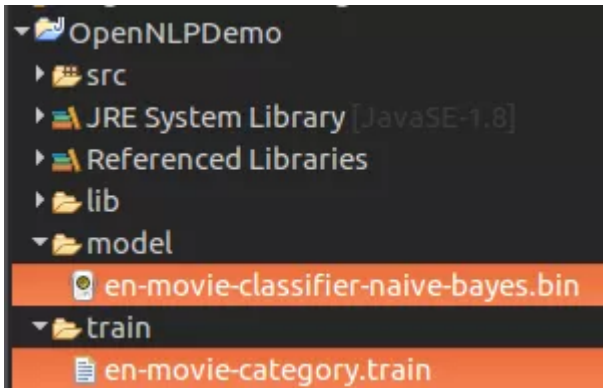
```
54
55              // print the probabilities o
56              System.out.println("\n------
57              for(int i=0;i<doccat.getNumb
58                  System.out.println(docca
59              }
60              System.out.println("--------
61
62              System.out.println("\n"+docc
63          }
64          catch (IOException e) {
65              System.out.println("An excep
66              e.printStackTrace();
67          }
68      }
69  }
```

When the above program is run, the output to the console is as shown below :

```
Program Output
Indexing events using cutoff of 0

    Computing event counts...  done. 66 ever
    Indexing...  done.
Collecting events... Done indexing.
Incorporating indexed data for training...
done.
    Number of Event Tokens: 66
        Number of Outcomes: 2
      Number of Predicates: 6886
Computing model parameters...
Stats: (27/66) 0.4090909090909091
...done.

Model is successfully trained.
Compressed 6886 parameters to 6886
3 outcome patterns

Trained Model is saved locally at : model/er

---------------------------------
Category : Probability
---------------------------------
Thriller : 2.1694037140217655E-14
Romantic : 0.9999999999999782
---------------------------------
```

```
Romantic : is the predicted category for the
```

The location of the training file and the locally saved model file are shown in the following picture :



Location of Training file and Generated Model file

Conclusion :

In this OpenNLP Tutorial, we have learnt briefly the training input requirements for Document Categorizer API of OpenNLP and also learnt the example program for Training of Document Categorizer using Naive Bayes Algorithm in OpenNLP used for document classification.

‹ Previous Next ›

Popular Tutorials

- Salesforce Tutorial
- SAP Tutorials
- R Tutorial
- Kafka Tutorial
- Kotlin Tutorial

Interview Questions

- Salesforce Visualforce Interview Questions
- Salesforce Apex Interview Questions
- HR Interview Questions
- Aptitude Interview Questions
- Kotlin Interview Questions

Tutorial Kart

About Us

Contact Us

Careers - Write for us