Dear students,

I have been looking for a way to help you out on his project, and I managed to put this email together. I hope it will help you make efficient progress in your project.

Here is the basic idea of your algorithm:

1. Give the entire collection of training documents to Balie and produce the bag of words which also produces term frquencies (call it BOW_all that represents all documents). Make sure you remove the stopwords (Balie can do this!) before you extract this bag of words.
* see BOW sample code below..!

2. for each document (j) in the training set and in the testing set, you repeat the above extraction of bag-of-words that produces the BOW_j which represents this particular document (j).

3. Then, write a small piece of code to compute the TF-IDF scores for each word in the bag of words BOW_all (extracted from step 1) using those words found in BOW_j, where (see course slides for these variables!):
- f_ij = frequency of term i in document j, i.e. the term i is in BOW_all and the frquency is in BOW_j
- tf_ij = f_ij / max{f_ij} for all terms in BOW_j
- df_i = number of documents that contain term i, this means for each term of BOW_all, compute the number of documents that contain that term i
- idf_i = log_2(N / df_i) where N = total number of training documents

4. produce a vector for each training and testing documents (keep them split!) to represnet that document. This vectore contains the TF-IDF weights (as computed above) for each word in BOW_all in each document. Now you have 2 options:

Option 1:
=========
- Save these vectors into two files, training and testing respectively.
- use weka to do feature selection on the training data, then
- build your classifier using the training data reduced to those features (words) selected by feature selection, and
- test your classifier on the test set

Option 2:
=========
Do exactly as option 1 but by calling weka from inside your program above becasue Balie has a built-in Weka tester that can interface into weka and calls weka classes from within. THis option is elegant, clean, and allows you to run and re-run many times, which reduces your experimental overhead, however, it requires more programming. See Balie Java documentation below.

Summary:
1. all training documents ==> Balie ==> BoW_all (with frequencies)
2. for each document (j) (training/testing) and BOW_all ==> Balie ==> data set (each document is a vector of TF-IDF weights for each word in BOW_all
3. after processing all training and testing documents, you now can use weka to train and test classifiers...
   Note: for this step, you can write a Java program that calls Balie, then calls Weka to do all this in a single Java program which you can run over and ove again and test to produce high quality experiments, see below. Alternatively, you can save your data and use weka gui manually!

Helpful Links:

* Java Documentation of Balie at:
http://balie.sourceforge.net/doc/

* Here is a simple example of calling Balie:
http://sourceforge.net/docman/display_doc.php?docid=25687&group_id=124581

* Balie Technical report:
http://balie.sourceforge.net/dnadeau05balie.pdf

* A Sample BOW Java code:

```
-------------snip----------------snip------------snip----------
import java.util.Enumeration;
import java.util.Hashtable;
```

```
import ca.uottawa.balie.AccentLookup;
import ca.uottawa.balie.Balie;
import ca.uottawa.balie.Token;
import ca.uottawa.balie.TokenFeature;
import ca.uottawa.balie.TokenList;
import ca.uottawa.balie.TokenListIterator;
import ca.uottawa.balie.Tokenizer;

public class Reuter {

    private static final String REUTER_TEXT_EXAMPLE = "Showers continued
throughout the week in the Bahia cocoa zone, alleviating the drought
since early January and improving prospects for the coming temporao,
although normal humidity levels have not been restored, Comissaria
Smith said in its weekly review.";

    public static void TestBalie() {
        PrintBagOfWord();
        PrintExhaustiveTokenFeatures();
    }

    private static void PrintBagOfWord() {
        Tokenizer t = new Tokenizer(Balie.LANGUAGE_ENGLISH, true, true);
        t.Tokenize(REUTER_TEXT_EXAMPLE);
        TokenList tl = t.GetTokenList();

        System.out.println("[Printing term frequencies]");
        Hashtable<String, Double> tf = tl.TermFrequencyTable();
        Enumeration<String> eCur = tf.keys();
        while (eCur.hasMoreElements()) {
            String strTerm = eCur.nextElement();
            Double fFreq = tf.get(strTerm);
            System.out.println(strTerm+":"+fFreq);
        }
    }
    public static void main(String[] args) {
        Reuter.TestBalie();
    }
}

-------------snip----------------snip------------snip----------
```

I hope this will help.

Good luck.