Refactoring Report 001

Refactoring ID: 001
Title: Split functionalities from countFrequency class and optimize data structure
Date: 2014/10/12
Vender: Acumen **Product**: News Classification & Recommendation
Platform: All platform it supports
PIC: FANG Zhou

Situation (Code Smell):

- The functionalities of reading files and building word occurrences dictionary are putted together in one class, thus it is hard to reuse the code for reading files
- Fixed size array makes it hard to utilize memory resources and handle large files.

Refactoring Plan:

- Split out the functionality of reading files to ReadContent class, which takes a file path as input and will return a String array.
- Use ArrayList to replace Fixed Size Array

Diff:

Original

```
public static Classifier BuildDictionary(String category, String [] content, HashMap<String,
Integer> useless_words){
    Hashtable<String, Integer> dict = new Hashtable<String, Integer>();
    for(int i=0; i < content.length; i++) {
        if (content[i] == null) break;

        if(content[i].matches(".*\\d.*") || !content[i].matches("[a-zA-Z-]+"))
            continue;
        //V02s
        if(useless_words.containsKey(content[i]))
            continue;
        //V02e
        if(dict.containsKey(content[i])){
            Integer count = dict.get(content[i]) + 1;
            dict.put(content[i], count);
        }else
            dict.put(content[i], 1);
    }
    Classifier categoryDic = new Classifier(category, dict);
    return categoryDic;
}

private void readAndProcess(File file) {
    final int MAX_WORDS_NUMBER = 20000;
    String[] words = new String[MAX_WORDS_NUMBER];

    try{
        Scanner scanner=new Scanner(new FileReader(file));
        String[] s;
        String line;

        while(scanner.hasNextLine())
        {
            line=scanner.nextLine();
```

```
            s=line.split(",|\\.|\\s+|\t|\"|\'|[illegible]|[illegible]");
            // for(int i=0;i<s.length;i++)
            //     System.out.print(s[i]+"\t");
            // System.out.println();
            copyArray(s);
        }

        scanner.close();
    }catch(Exception e){
        System.out.println(e.getMessage());
    }
}
```

Updated

```
public class ReadContent {

    /** The words. */
    private ArrayList<String> words;

    /** The input path. */
    private String inputPath;

    public ReadContent(String inputPath){
        this.inputPath = inputPath;
        this.words = new ArrayList<String>();
    }


    public String [] processDocument(){
        File inputFile = new File(inputPath);
        if(!inputFile.exists())
            return new String[0];

        if(inputFile.isDirectory()){
            File[] subFiles = inputFile.listFiles();

            for(File singleFile : subFiles){
                readAndFilter(singleFile);
            }
        }else{
            readAndFilter(inputFile);
        }

        return this.words.toArray(new String[words.size()]);
    }

    private void readAndFilter(File srcFile){
        FileReader fr;
        try {
            fr = new FileReader(srcFile);
             BufferedReader br = new BufferedReader(fr);
             String line = null;
             while((line = br.readLine()) != null){
                //TODO: please add this in test case, see if the split works fine
                //split by non a-z, A-Z, -, _ ,0-9 characters
                String tokens[] = line.split("[^a-zA-Z-_0-9]");
                arrayCopy(tokens);
            }
             br.close();
        } catch (FileNotFoundException e) {
```

```
            e.printStackTrace();
        }catch (IOException e){
            e.printStackTrace();
        }
    }
```