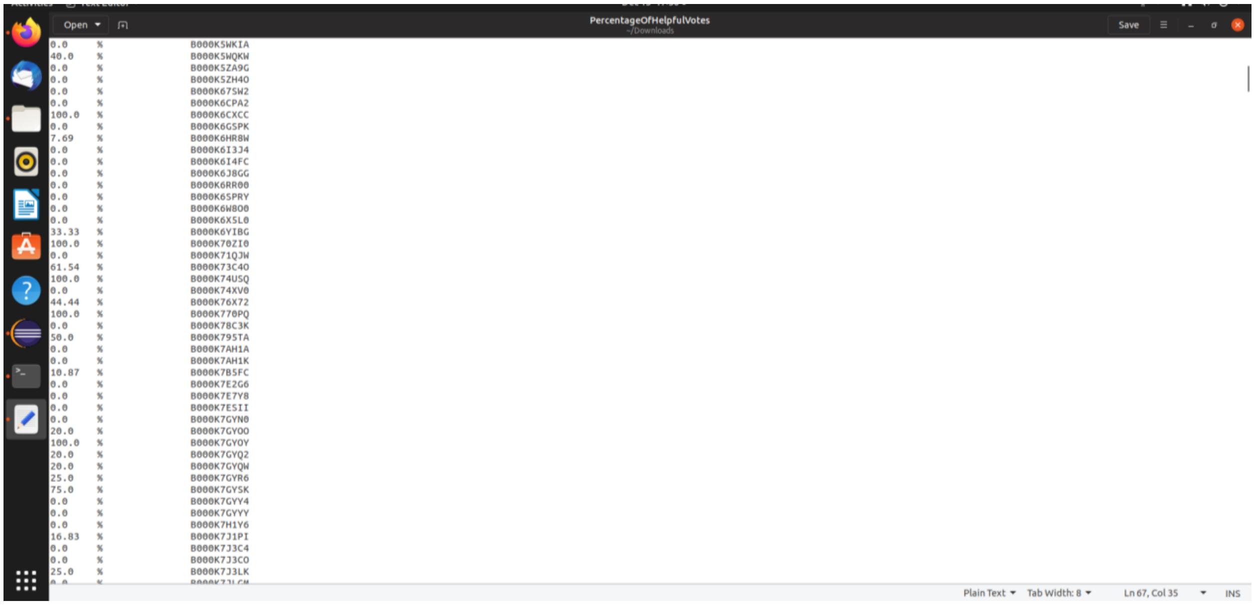
Helpful Review percentage

(Numerical Summarization Pattern)

There is an attribute in the dataset called helpfulness. It indicates the usefulness of the review in buying the product and not the product itself. There is another column called Total Votes. So, in this map reduce I am going to calculate ratio of Helpful_votes/ Total_votes and then converting it into percentage. So this map reduce will give what percent of votes that customer found helpful for a particular product.

OUTPUT:

```
./hadoop jar /home/sayali/Desktop/PercentHelpfulReview.jar
sayali.AmazonReviews9.PercentHelpfulReview  /AmazonReviews/AmazonReviews.tsv
/AmazonPercentHelpfulVotes
```



```
0.0     %               B000K5WKIA
40.0    %               B000K5WQKW
0.0     %               B000K5ZA9G
0.0     %               B000K5ZH4O
0.0     %               B000K67SW2
0.0     %               B000K6CPA2
100.0   %               B000K6CXCC
0.0     %               B000K6GSPK
7.69    %               B000K6HR8W
0.0     %               B000K6I3J4
0.0     %               B000K6I4FC
0.0     %               B000K6J8GG
0.0     %               B000K6RR00
0.0     %               B000K6SPRY
0.0     %               B000K6W8O0
0.0     %               B000K6X5L0
33.33   %               B000K6YIBG
100.0   %               B000K70ZI0
0.0     %               B000K71QJW
61.54   %               B000K73C4O
100.0   %               B000K74USQ
0.0     %               B000K74XV0
44.44   %               B000K76X72
100.0   %               B000K770PQ
0.0     %               B000K78C3K
50.0    %               B000K795TA
0.0     %               B000K7AH1A
0.0     %               B000K7AH1K
10.87   %               B000K7B5FC
0.0     %               B000K7E2G6
0.0     %               B000K7E7Y8
0.0     %               B000K7ESII
0.0     %               B000K7GYN0
20.0    %               B000K7GYOO
100.0   %               B000K7GYOY
20.0    %               B000K7GYQ2
20.0    %               B000K7GYQW
25.0    %               B000K7GYR6
75.0    %               B000K7GYSK
0.0     %               B000K7GYY4
0.0     %               B000K7GYYY
0.0     %               B000K7H1Y6
16.83   %               B000K7J1PI
0.0     %               B000K7J3C4
0.0     %               B000K7J3CO
25.0    %               B000K7J3LK
0.0     %               B000K7JLCM
```

Map Reduce Code

```
package sayali.AmazonReviews9;

import java.io.IOException;
import java.text.DecimalFormat;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.FloatWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
```

Mapper:

```
public class PercentHelpfulReviewMapper extends Mapper<LongWritable, Text, Text, FloatWritable> {

    private Text text = new Text();
    private FloatWritable score = new FloatWritable();

    @Override
    protected void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException {

        if (key.get() == 0) {
            return;
        }
        else {

            String[] line = value.toString().split("\\t");
            String productId = line[3].trim();
            int helpnum = Integer.parseInt(line[8].trim());
            int helpden = Integer.parseInt(line[9].trim());
            float ratio;
            if (helpden != 0) {
                ratio = helpnum / helpden;
            } else {
                ratio = (float) 0.0;
            }
            text.set(productId);
            score.set(ratio);

            context.write(text, score);
        }
    }
}
```

Reducer:

```
public class PercentHelpfulReviewReducer extends Reducer<Text, FloatWritable, FloatWritable, Text> {
    private FloatWritable result = new FloatWritable();
    @Override
    protected void reduce(Text key, Iterable<FloatWritable> values, Context context)
                throws IOException, InterruptedException {
        float sum = 0;
        int count = 0;
        for (FloatWritable val : values) {
            sum += val.get();
            count = count + 1;
        }
        float average = (sum / count) * 100;
        DecimalFormat df = new DecimalFormat();
        df.setMaximumFractionDigits(2);
        float avg = Float.parseFloat(df.format(average));

        key.set("%" + "            " + key);
        result.set(avg);
        context.write(result, key);
    }
}
```

Driver:

```
public class PercentHelpfulReview {
public static void main(String[] args) throws IOException, InterruptedException,
ClassNotFoundException {
        Configuration conf = new Configuration();
        Job job = Job.getInstance(conf, "ProductId and Ratings");
        job.setJarByClass(PercentHelpfulReview.class);
        job.setMapperClass(PercentHelpfulReviewMapper.class);
        job.setMapOutputKeyClass(Text.class);
        job.setMapOutputValueClass(FloatWritable.class);
        job.setReducerClass(PercentHelpfulReviewReducer.class);
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(FloatWritable.class);
        FileInputFormat.addInputPath(job, new Path(args[0]));
        FileOutputFormat.setOutputPath(job, new Path(args[1]));
        System.exit(job.waitForCompletion(true) ? 0 : 1);

    }
}
```