First, copy input file **textdata.txt** into HDFS directory **/input**. You might want to create /input directory on HDFS if you don't have it.

***$ hdfs dfs -mkdir /input***

Then copy input file,

***$ hdfs dfs -put input/textdata.txt /input***

Execute the word count program.

***$ hadoop jar wordcount.jar WordCount /input/textdata.txt /output/wordcount***

*Glimpses:*

What is inside in wordcount java

package com.mapreduce.wc;  
import java.io.IOException;   
import org.apache.hadoop.conf.Configuration;   
import org.apache.hadoop.fs.Path;   
import org.apache.hadoop.io.IntWritable;   
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;  
import org.apache.hadoop.util.GenericOptionsParser;   
public class WordCount {   
 public static void main(String [] args) throws Exception   
 {   
 Configuration c=new Configuration();   
 String[] files=new GenericOptionsParser(c,args).getRemainingArgs();   
 Path input=new Path(files[0]);   
 Path output=new Path(files[1]);   
 Job j=new Job(c,"wordcount");   
 j.setJarByClass(WordCount.class);   
 j.setMapperClass(MapForWordCount.class);   
 j.setReducerClass(ReduceForWordCount.class);   
 j.setOutputKeyClass(Text.class);   
 j.setOutputValueClass(IntWritable.class);   
 FileInputFormat.addInputPath(j, input);   
 FileOutputFormat.setOutputPath(j, output);   
 System.exit(j.waitForCompletion(true)?0:1);   
 }   
public static class MapForWordCount extends Mapper<LongWritable, Text, Text,   
 IntWritable>{   
 public void map(LongWritable key, Text value, Context con) throws   
IOException, InterruptedException  
 {   
 String line = value.toString();   
 String[] words=line.split(" ");   
 for(String word: words )   
 {   
 Text outputKey = new Text(word.toUpperCase().trim());   
 IntWritable outputValue = new IntWritable(1);   
 con.write(outputKey, outputValue);   
 }   
 }   
}   
public static class ReduceForWordCount extends Reducer<Text, IntWritable, Text,   
IntWritable>   
{   
 public void reduce(Text word, Iterable<IntWritable> values, Context con)   
 throws IOException, InterruptedException   
 {   
 int sum = 0;   
 for(IntWritable value : values)   
 {   
 sum += value.get();  
 }   
 con.write(word, new IntWritable(sum));   
 }   
 }   
}

*If you want alternative jar file then,*

*Sources:*

[*https://github.com/petehouston/hadoop-wordcount*](https://github.com/petehouston/hadoop-wordcount)

[*https://medium.com/@tirthshah100/word-count-in-apache-hadoop-mapreduce-c6ee8e737fb9*](https://medium.com/@tirthshah100/word-count-in-apache-hadoop-mapreduce-c6ee8e737fb9)