**Assignment21.1**

**Problem Statement**

**Count the number of blank lines in a text file, by using accumulators**

Sample file :

Hello World

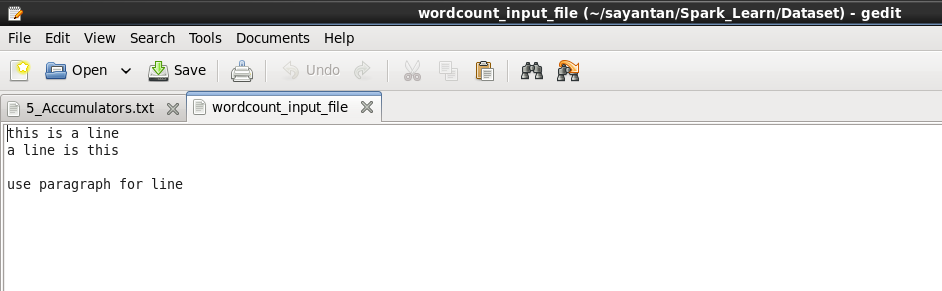
It’s a sunny day

<blank\_line>

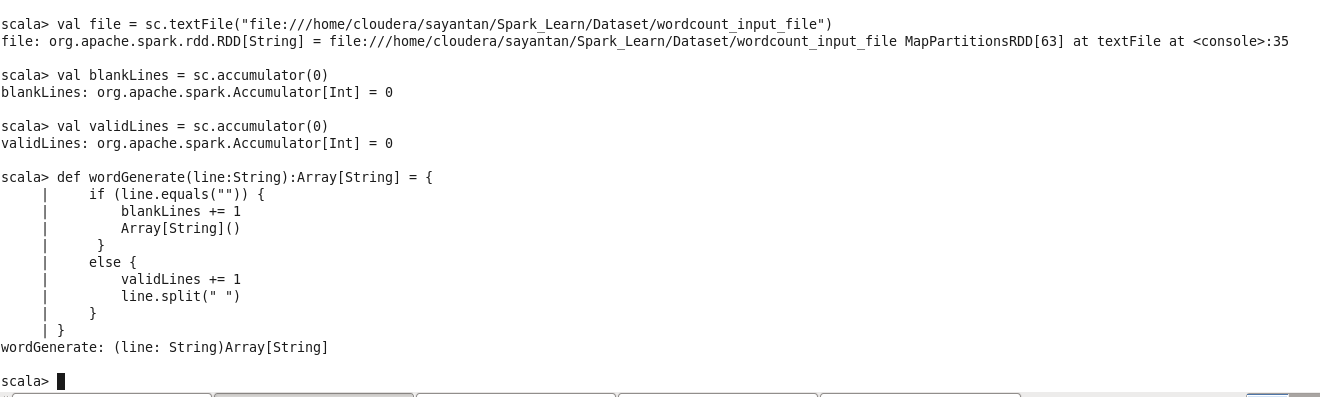
When will it rain?

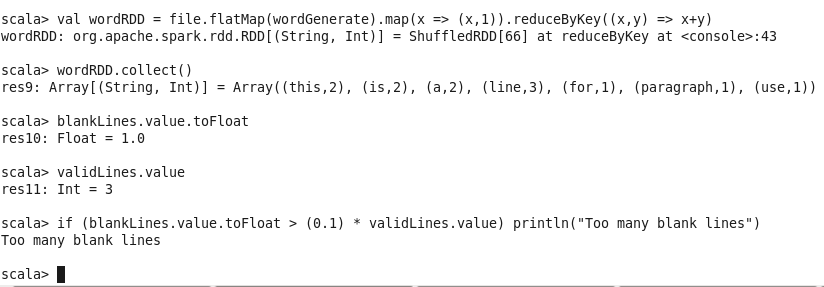
Will it rain today!

The sample input file looks as follows



Now we have to find the number of blank lines between the lines of text using accumulators.





It is possible to aggregate values from an entire RDD back to the driver program using actions like reduce(), but sometimes we need a simple way to aggre‐gate values that, in the process of transforming an RDD, are generated at different scale or granularity than that of the RDD itself.

Accumulators do not change the lazy evaluation model of Spark. If they are being updated within an operation on an RDD, their value is only updated once that RDD is computed as part of an action.

We create them in the driver by calling the SparkContext.accumulator(initial Value) method, which produces an accumulator holding an initial value. The return type is an org.apache.spark.Accumulator[T] object, where T is the type of initialValue. Worker code in Spark closures can add to the accumulator with its += method (or add in Java).The driver program can call the value property on the accumulator to access its value (or call value() and setValue() in Java).Note that tasks on worker nodes cannot access the accumulator’s value() — from the point of view of these tasks, accumulators are write-only variables.Spark automatically deals with failed or slow machines by re-executing failed or slow tasks.

For example, if the node running a partition of a map() operation crashes, Spark will rerun it on another node; and even if the node does not crash but is simply much slower than other nodes, Spark can preemptively launch a “speculative” copy of the task on another node, and take its result if that finishes.

Even if no nodes fail, Spark may have to rerun a task to rebuild a cached value that falls out of memory. The net result is therefore that the same function may run multiple times on the same data depending on what happens on the cluster.For accumulator updates performed inside actions only, Spark guarantees that each task’s update to the accumulator will only be applied once, i.e. restarted tasks will not update the value. In transformations, users should be aware of that each task’s update may be applied more than once if tasks or job stages are re-executed.