* RDDs do not need to incur the overhead of checkpointing, as they can be recovered using lineage.
* Only the lost partitions of an RDD need to be recomputed upon failure, and they can be recomputed in parallel on different nodes, without having to roll back the whole program.
* RDDs are best suited for batch applications that apply the same operation to all elements of a dataset; RDDs would be less suitable for applications that make asynchronous finegrained updates to shared state, such as a storage system for a web application or an incremental web crawler.
* When implementing two iterative machine learning applications, logistic regression and k-means, to compare the performance of Hadoop and Spark, we found the key difference between the two applications is the amount of computation they perform per byte of data.
* The iteration time of kmeans is dominated by computation, while logistic regression is less compute-intensive and thus more sensitive to time spent in deserialization and I/O.
* Spark outperformed Hadoop due to several factors: Minimum overhead of the Hadoop software stack, Overhead of HDFS while serving data, and Deserialization cost to convert binary records to usable in-memory Java objects.
* MLlib consists of fast and scalable implementations of standard learning algorithms for common learning settings including classification, regression, collaborative filtering, clustering, and dimensionality reduction. It also provides a variety of underlying statistics, linear algebra, and optimization primitives within spark.