**Analyzing Apache access log using Apache Spark and Hadoop**

**and visualization using R**

In this project I am trying to demonstrate how we can leverage the power of apache Spark and Hadoop

to process apache access log file to get insights of what is going on the site we are hosting.

I want to see the following inquiries:-

* Hits by Hour of the day?   
  This will help us to see at what time of the hour our site is handling more requests.
* Hits by Ip Address (Top 10)?  
  To identify potential customers or fraud
* Identify possible broken Url

**Hits by Hour of the day**

**Spark Code:**

def accessByHour(input:RDD[String], sc:SparkContext) = {

val output = input

.map(s => Log(s))

.filter(log => log != null)

.map(log => (log.dateTime.getHour, 1))

.reduceByKey(\_+\_)

.map{ case (k,v) => (v,k)}

.sortByKey(ascending = false)

.map{ case (v,k) => k + "," + v}

val header = sc.parallelize(Array("hour,count"))

val finalOutput = header.union(output)

//Coalesce is used to group together the output from different partition in to one

finalOutput.coalesce(1, true).saveAsTextFile("/bigdata/analytics/output/byHour")

*/\* finalOutput.repartition(1).saveAsTextFile("/bigdata/analytics/output")\*/*

}

finalOutput.coalesce(1, true).saveAsTextFile("/bigdata/analytics/output/byHour")

*/\* finalOutput.repartition(1).saveAsTextFile("/bigdata/analytics/output")\*/*

}

**View result from hadoop**

hadoop fs -cat /bigdata/analytics/output/byHour/part-00000

hour,count

17,19533

15,18690

20,18186

16,17991

12,17512

19,17040

13,16885

18,16849

14,16824

21,16359

22,14779

10,10050

8,9367

9,9190

11,8915

7,8113

23,7467

6,5192

0,4565

1,4478

5,3626

2,3436

3,3265

4,2662

**Visualize using R**

accessByHour <- read.df(sqlContext, "hdfs://localhost:9001/bigdata/analytics/output/byHour/part-00000","com.databricks.spark.csv", header="true")

#convert sparkr data frame in to r data frame

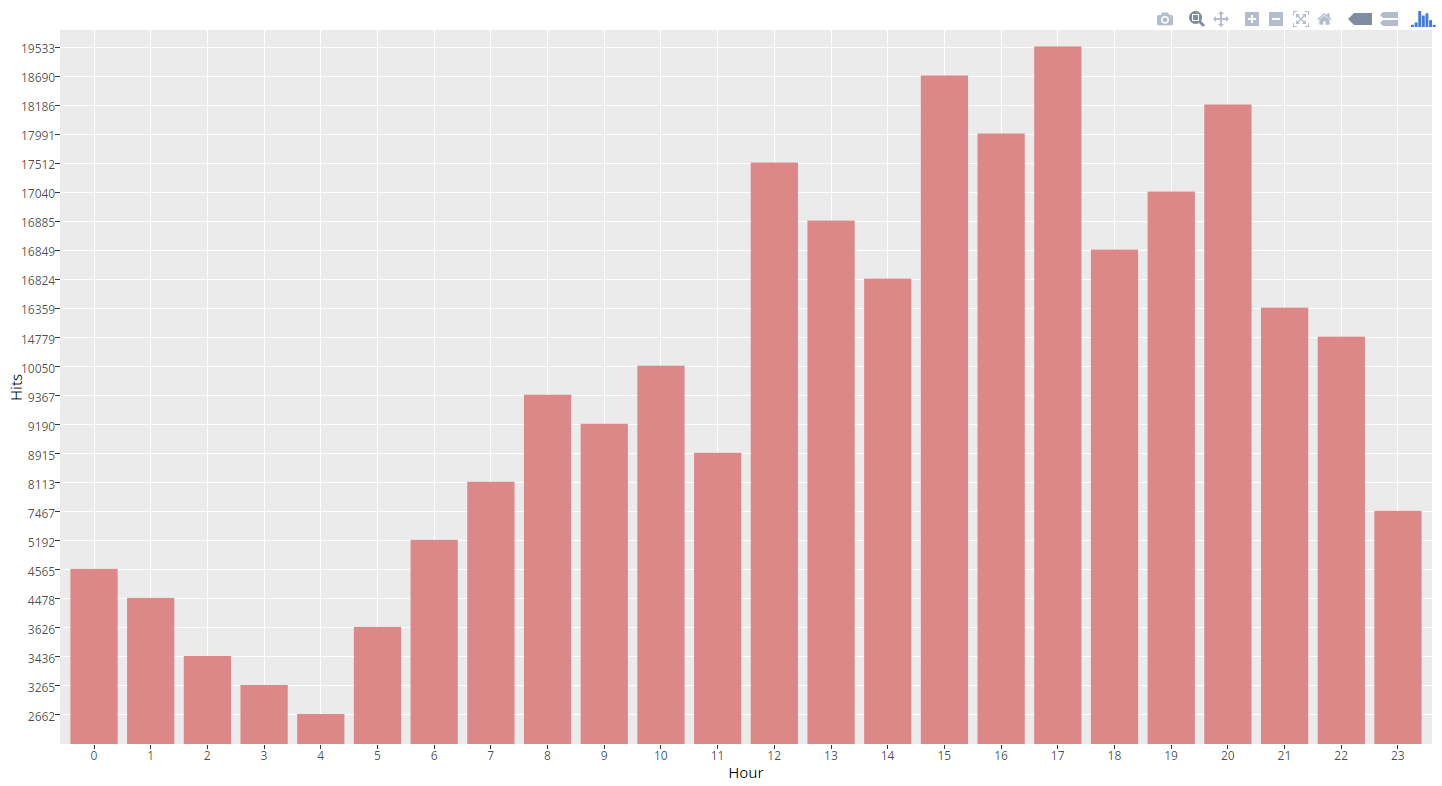
accessByHour <- collect(select(accessByHour, "hour", "count"))

accessByHour$hour <- factor(as.integer(accessByHour$hour))

accessByHour$count <- factor(as.integer(accessByHour$count))

byHourPlot <- ggplot(NULL, aes(x = accessByHour$hour, y=accessByHour$count)) + geom\_bar(colour="#DD8888", fill="#DD8888",stat = "identity") + xlab("Hour") + ylab("Hits")

ggplotly(byHourPlot)



**Hits by Ip Address (Top 10)**

**spark code:**

def accessByIpAddress(input:RDD[String], sc:SparkContext) = {

val output = input

.map(s => Log(s))

.filter(l => l != null)

.map(log => (log.ipAddress, 1))

.reduceByKey(\_+\_)

.map{ case (k,v) => (v,k)}

.sortByKey(ascending = false)

.map{ case (v,k) => k + "," + v}

val header = sc.parallelize(Array("ipAddress,count"))

val finalOutput = header.union(output)

//Coalesce is used to group together the output from different partition in to one

finalOutput.coalesce(1, true).saveAsTextFile("/bigdata/analytics/output/byIpAddress")

*/\* finalOutput.repartition(1).saveAsTextFile("/bigdata/analytics/output")\*/*

}

**View result from hadoop**

hadoop fs -cat /bigdata/analytics/output/byIpAddress/part-00000

205.167.170.15,31844

134.249.53.185,17904

192.227.172.158,13474

37.1.206.196,3622

79.142.95.122,3207

78.186.191.187,2840

200.148.166.42,2829

148.251.50.49,1929

149.56.81.58,1520

177.134.189.192,1353

91.200.12.22,1255

177.180.140.8,843

200.137.2.52,829

52.22.118.215,734

84.112.161.41,712

91.200.12.108,673

46.5.196.77,668

192.111.153.243,586

213.150.254.81,434

212.31.90.192,390

84.58.165.21,381

80.121.16.94,368

188.163.107.190,363

…………….

**Visualize using R**

accessByIp <- read.df(sqlContext, "hdfs://localhost:9001/bigdata/analytics/output/byIpAddress/part-00000","com.databricks.spark.csv", header="true")

accessByIp <- take(accessByIp, 10)

accessByIp$count <- factor(as.integer(accessByIp$count))

byIpPlot <- ggplot(NULL, aes(x = accessByIp$ipAddress, y = accessByIp$count, group = 1)) + geom\_line(color = accessByIp$count) + geom\_point() + xlab("Ip Address") + ylab("Hits")

ggplotly(byIpPlot)

