Assignment 1

Untitled Untitled Untitled Untitled

FINISHED

```
// Imports
import org.apache.spark.sql.functions._
import org.joda.time.format.DateTimeFormat
```

```
import org.apache.spark.sql.functions._
import org.joda.time.format.DateTimeFormat
```

Took 1 sec. Last updated by anonymous at February 02 2017, 8:49:01 PM.

```
import org.apache.spark.SparkConf
```

```
import org.apache.spark.SparkConf
```

Took 2 sec. Last updated by anonymous at February 02 2017, 7:58:53 PM.

FINISHED r Y M

```
%spark
```

FINISHED r Y M

```
import org.apache.spark.rdd._
import scala.collection.JavaConverters._
import au.com.bytecode.opencsv.CSVReader

import java.io._ import
org.joda.time._ import
org.joda.time.format._
import org.joda.time.format.DateTimeFormat
import org.joda.time.DateTime import
org.joda.time.Days
```

```
import org.apache.spark.rdd._ import
scala.collection.JavaConverters._ import
au.com.bytecode.opencsv.CSVReader import
java.io._ import org.joda.time._ import
org.joda.time.format._
import org.joda.time.format.DateTimeFormat
import org.joda.time.DateTime import
org.joda.time.Days
```

Took 5 sec. Last updated by anonymous at February 02 2017, 8:49:16 PM.

FINISHED r Y M

```
case class DelayRec(year: String,                                  month: String,
dayOfMonth: String,                    dayOfWeek: String,                crsDepTime:
String,                    depDelay: String,                   origin: String,
                    distance: String,
cancelled: String) {

    val holidays = List("01/01/2007", "01/15/2007", "02/19/2007", "05/28/2007", "06/07/2007",
      "09/03/2007", "10/08/2007" ,"11/11/2007", "11/22/2007", "12/25/2007",
      "01/01/2008", "01/21/2008", "02/18/2008", "05/22/2008", "05/26/2008", "07/04/2008",
      "09/01/2008", "10/13/2008" ,"11/11/2008", "11/27/2008", "12/25/2008")
```

```
    def gen_features: (String, Array[Double]) = {
  val values = Array(         depDelay.toDouble,
  month.toDouble,         dayOfMonth.toDouble,
  dayOfWeek.toDouble,
  get_hour(crsDepTime).toDouble,
  distance.toDouble,
        days_from_nearest_holiday(year.toInt, month.toInt, dayOfMonth.toInt)
      )
      new Tuple2(to_date(year.toInt, month.toInt, dayOfMonth.toInt), values)
    }

    def get_hour(depTime: String) : String = "%04d".format(depTime.toInt).take(2)     def
  to_date(year: Int, month: Int, day: Int) = "%04d%02d%02d".format(year, month, day)

    def days_from_nearest_holiday(year:Int, month:Int, day:Int): Int = {
  val sampleDate = new org.joda.time.DateTime(year, month, day, 0, 0)

      holidays.foldLeft(3000) { (r, c) =>
        val holiday = org.joda.time.format.DateTimeFormat.forPattern("MM/dd/yyyy").parseDateTi
    val distance = Math.abs(org.joda.time.Days.daysBetween(holiday, sampleDate).getDays)
```

Ass i g n m mate h.nmitn(r1, Udinstitalendce U)ntitled Untitled Untitled Untitled Untitled Untitled Z }eppelin

```
    }
}
```

Adsefsinigd nclamss eDenlatyR e1c

r Y M U default

Took 1 sec. Last updated by anonymous at February 02 2017, 8:49:20 PM.

FINISHED r Y M def

```
// function to do a preprocessing step for a given file
prepFlightDelays(infile: String): RDD[DelayRec] = {     val data = sc.textFile(infile)

    data.map { line =>
      val reader = new CSVReader(new StringReader(line))
      reader.readAll().asScala.toList.map(rec => DelayRec(rec(0),rec(1),rec(2),rec(3),rec(5),r
}.map(list => list(0))
    .filter(rec => rec.year != "Year")
    .filter(rec => rec.cancelled == "0")
    .filter(rec => rec.origin == "ORD")
}

val data_2007tmp = prepFlightDelays("/Users/ppoornima/Downloads/2007.csv") val data_2007 =
data_2007tmp.map(rec => rec.gen_features._2) val data_2008 =
prepFlightDelays("/Users/ppoornima/Downloads/2008.csv").map(rec => rec.gen_
data_2007tmp.toDF().registerTempTable("data_2007tmp") data_2007.take(5).map(x => x mkString
",").foreach(println)
```

```
prepFlightDelays: (infile: String)org.apache.spark.rdd.RDD[DelayRec] data_2007tmp:
org.apache.spark.rdd.RDD[DelayRec] = MapPartitionsRDD[6] at filter at <console>:
73
data_2007: org.apache.spark.rdd.RDD[Array[Double]] = MapPartitionsRDD[7] at map at <console>:6
7
data_2008: org.apache.spark.rdd.RDD[Array[Double]] = MapPartitionsRDD[15] at map at <console>:
65
warning: there was one deprecation warning; re-run with -deprecation for details -
8.0,1.0,25.0,4.0,11.0,719.0,10.0
```

```
41.0,1.0,28.0,7.0,15.0,925.0,13.0
45.0,1.0,29.0,1.0,20.0,316.0,14.0
-9.0,1.0,17.0,3.0,19.0,719.0,2.0
180.0,1.0,12.0,5.0,17.0,316.0,3.0
```

Took 21 sec. Last updated by anonymous at February 02 2017, 8:50:20 PM.

FINISHED

```
// Imports
import org.apache.spark.sql.functions._
import org.joda.time.format.DateTimeFormat

import org.apache.spark.sql.functions._
```

```
import org.apache.spark.sql.functions._
import org.joda.time.format.DateTimeFormat
import org.apache.spark.sql.functions._
```

Took 2 sec. Last updated by anonymous at February 02 2017, 8:47:40 PM.

Zeppelin

READY

Assignment 1


default