BIG DATA HADOOP & SPARK TRAINING

Case Study III from session 19 on Sensor data

CASE STUDY III

Case study on Sensor data

For this data analysis, you can download the necessary dataset from this link.

In the above link there are two datasets; building.csv contains the details of the top 20 buildings all over the world and HVAC.csv contains the target temperature and the actual temperature along with the building Id.

HVAC (heating, ventilating/ventilation, and air conditioning) is the technology of indoor and vehicular environmental comfort. Its goal is to provide thermal comfort and acceptable indoor air quality. Through the HVAC sensors, we will get the temperature of the buildings.

Here are the columns that are present in the datasets:

Building.csv – BuildingID, BuildingMgr, BuildingAge, HVACproduct,Country

HVAC.csv – Date, Time, TargetTemp, ActualTemp, System, SystemAge, BuildingID

Tasks:

- Load HVAC.csv file into temporary table
- Add a new column, tempchange - set to 1, if there is a change of greater than +/-5 between actual and target temperature
- Load building.csv file into temporary table
- Figure out the number of times, temperature has changed by 5 degrees or more for each country:
    - Join both the tables.
    - Select tempchange and country column
    - Filter the rows where tempchange is 1 and count the number of occurrence for each country

Load HVAC.csv, building.csv file to spark:

To load csv file to spark we have to perform the following steps:

- Create a manual schema for both csv files which would provide the schema while loading both CSV's as shown below

```
val Manual_schema_HVAC = new StructType(Array(new StructField("Date", StringType,
true),
  new StructField("Time", StringType, false),
  new StructField("TargetTemp", LongType, true),
  new StructField("ActualTemp", LongType, false),
  new StructField("System", LongType, false),
  new StructField("SystemAge", LongType, false),
  new StructField("BuildingID", LongType, false)))

val Manual_schema_Building = new StructType(Array(new StructField("BuildingID",
LongType, true),
  new StructField("BuildingMgr", StringType, false),
  new StructField("BuildingAge", LongType, true),
  new StructField("HVACproduct", StringType, false),
  new StructField("Country", StringType, false)))
```

Note:

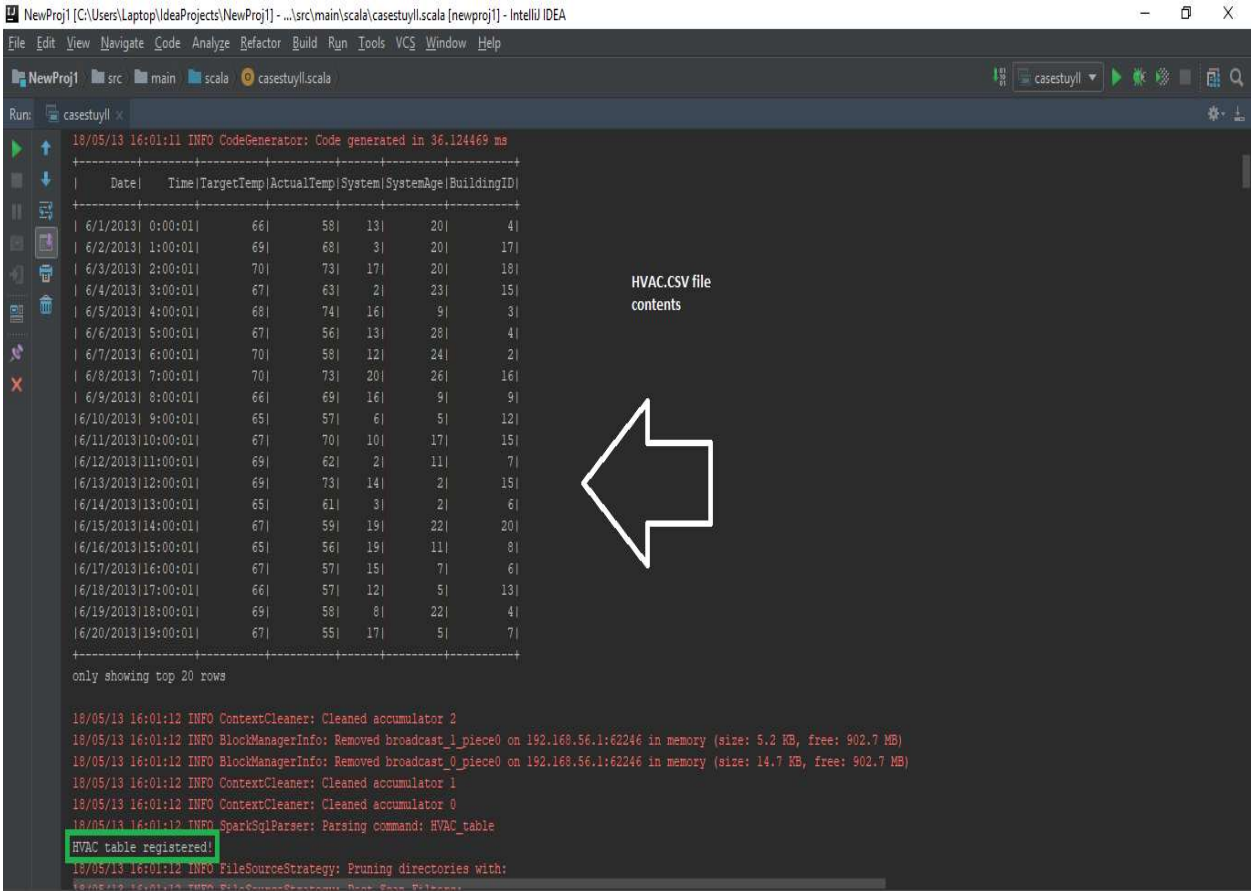
StructType is a built-in data type used for Schema definition in Spark SQL, to represent a collection of StructFields that together define a schema or its part.

<schema-name> = new StructType<array_of_columns><Struct_field>(<column_name>,<data_type_of_column>,<nullable_or_not_nullable(true/false)>)

- Now we load the CSV files from local file system to spark as shown below:

```
    val HVAC = spark.read.format("CSV")
    .option("header",true)
    .schema(Manual_schema_HVAC)
    .load("E:\\casestudies\\sensorcasestudy\\HVAC.csv")

    HVAC.show()
    HVAC.registerTempTable("HVAC_table")
    println("HVAC table registered!")

    val buildings = spark.read.format("CSV")
    .option("header", true)
    .schema(Manual_schema_Building)
    .load("E:\\casestudies\\sensorcasestudy\\buliding.csv")
    buildings.show()

    buildings.registerTempTable("building_table")
    println("buildings table registered!")
```

- We are using the CSV file read format, this provides various options of which we have used a few of them, which are as follows:
    - We have used the option to remove the header from the input file.
    - We have given the manual schema that we have created in the previous step
    - We are giving the path where the CSV file is saved in the local file system.
- Once loading is done we can register the respective tables to TempTable as shown above i.e. HVAC.registerTempTable("HVAC_table").



NewProj1 [C:\Users\Laptop\IdeaProjects\NewProj1] - ...\src\main\scala\casestuyII.scala [newproj1] - IntelliJ IDEA

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help

NewProj1 src main scala casestuyII.scala

Run: casestuyII

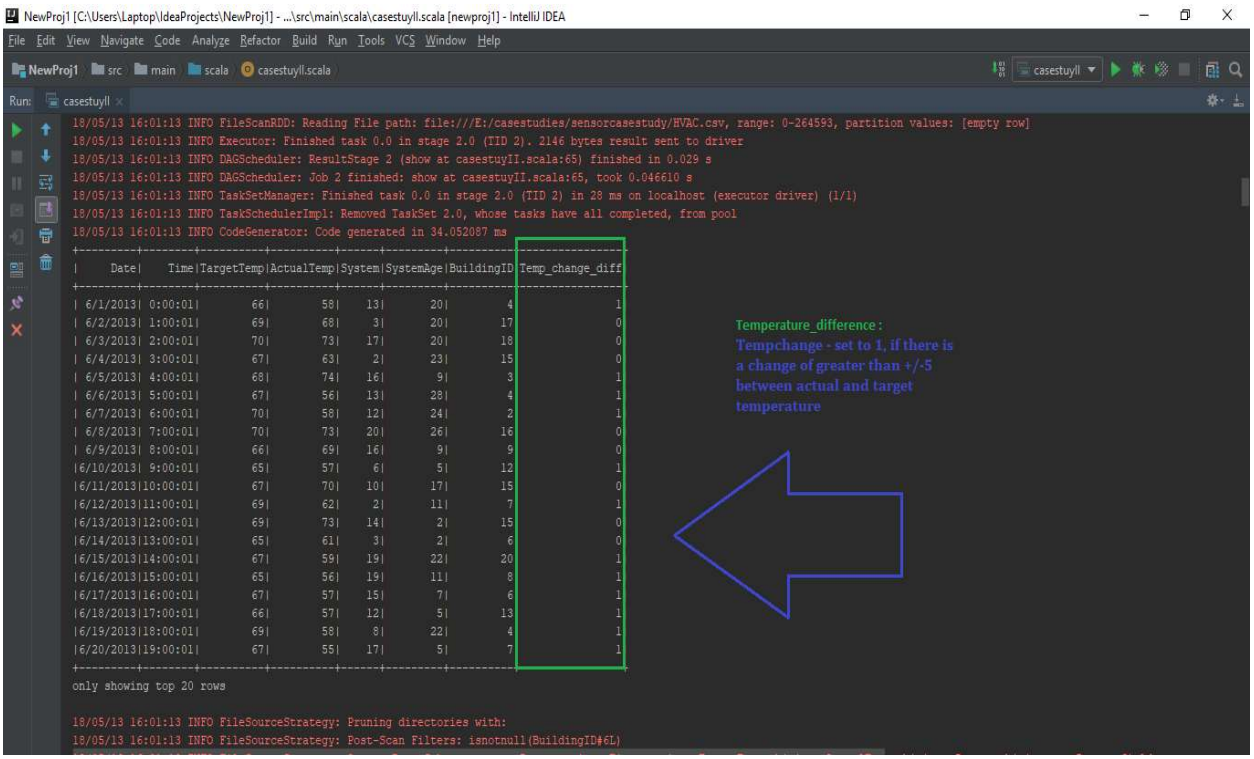
```
18/05/13 16:01:12 INFO DAGScheduler: Job 1 finished: show at casestuyII.scala:57, took 0.068446 s
18/05/13 16:01:12 INFO CodeGenerator: Code generated in 26.117151 ms
+----------+-----------+-----------+-----------+------------+
|BuildingID|BuildingMgr|BuildingAge|HVACproduct|     Country|
+----------+-----------+-----------+-----------+------------+
|         1|         M1|         25|     AC1000|         USA|
|         2|         M2|         27|     FN39TG|      France|
|         3|         M3|         28|     JDNS77|      Brazil|
|         4|         M4|         17|     GG1919|     Finland|
|         5|         M5|          3|    ACMAX22|   Hong Kong|
|         6|         M6|          9|     AC1000|   Singapore|
|         7|         M7|         13|     FN39TG|South Africa|
|         8|         M8|         25|     JDNS77|   Australia|
|         9|         M9|         11|     GG1919|      Mexico|
|        10|        M10|         23|    ACMAX22|       China|
|        11|        M11|         14|     AC1000|     Belgium|
|        12|        M12|         26|     FN39TG|     Finland|
|        13|        M13|         25|     JDNS77|Saudi Arabia|
|        14|        M14|         17|     GG1919|     Germany|
|        15|        M15|         19|    ACMAX22|      Israel|
|        16|        M16|         23|     AC1000|      Turkey|
|        17|        M17|         11|     FN39TG|       Egypt|
|        18|        M18|         25|     JDNS77|   Indonesia|
|        19|        M19|         14|     GG1919|      Canada|
|        20|        M20|         19|    ACMAX22|   Argentina|
+----------+-----------+-----------+-----------+------------+

18/05/13 16:01:12 INFO SparkSqlParser: Parsing command: building_table
buildings table registered!
18/05/13 16:01:12 INFO SparkSqlParser: Parsing command: select *, IF((TargetTemp-ActualTemp)> 5 ,'1', IF((TargetTemp-ActualTemp)< -5 ,'1',0)) as Temp_change_diff from HVAC_table
18/05/13 16:01:13 INFO FileSourceStrategy: Pruning directories with:
18/05/13 16:01:13 INFO FileSourceStrategy: Post-Scan Filters:
18/05/13 16:01:13 INFO FileSourceStrategy: Output Data Schema: struct<Date: string, Time: string, TargetTemp: bigint, ActualTemp: bigint, System: bigint ... 5 more fields>
18/05/13 16:01:13 INFO FileSourceStrategy: Pushed Filters:
18/05/13 16:01:13 INFO CodeGenerator: Code generated in 31.742739 ms
```

Building.csv file contents

- Now we will add a new column, tempchange - set to 1, if there is a change of greater than +/-5 between actual and target temperature

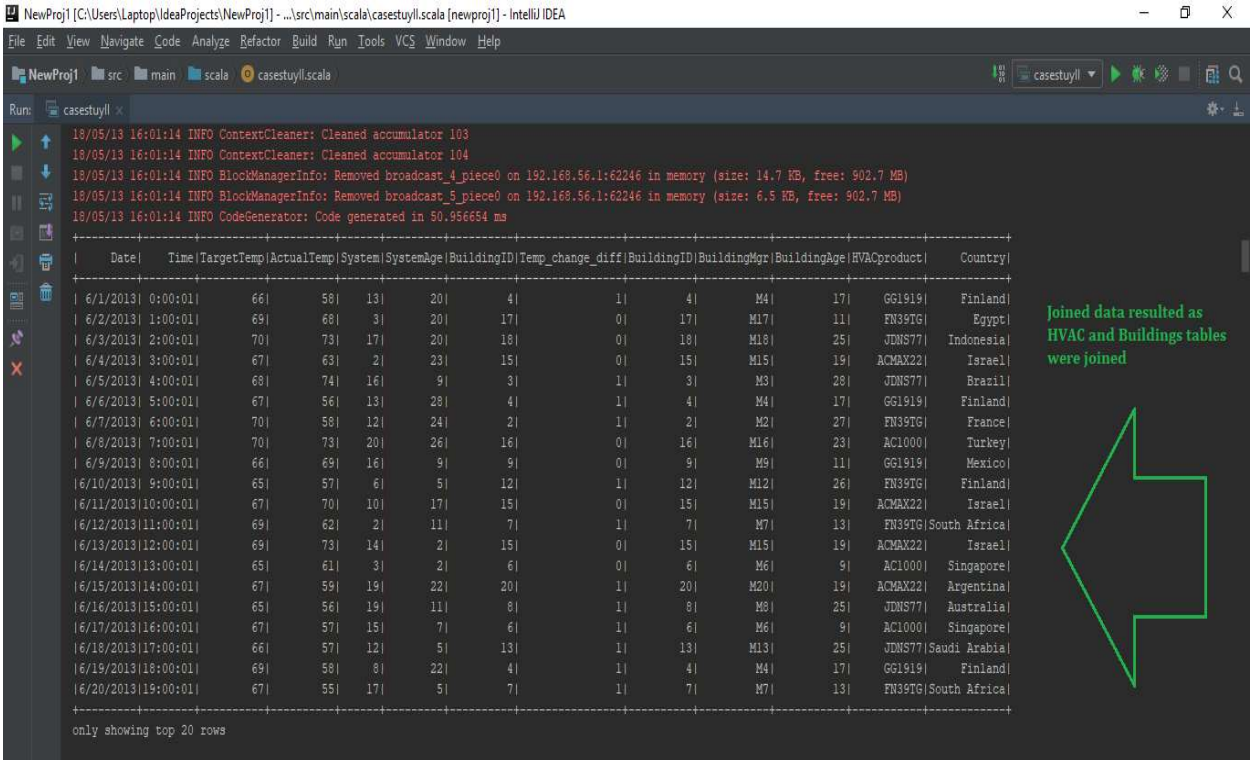
```
val filterHVAC = spark.sql("""select *, IF((TargetTemp-ActualTemp)> 5 ,'1',
IF((TargetTemp-ActualTemp)< -5 ,'1',0)) as Temp_change_diff from HVAC_table""")
filterHVAC.show()
```

  - Here we are filtering based on the difference between the Target temperature and Actual temperature, i.e. if the difference is between +/- 5 then give '1' as output if not give output as '0'.



- Now we will Join both the tables based on BuildingID, i.e. we will join HVAC and Buildings tables and register the output of join to a temptable called "HVACJBUILD"

```
val joinExpression = filterHVAC.col("BuildingID") ===
buildings.toDF().col("BuildingID")
 val HVACJOBUILD = filterHVAC.join(buildings,joinExpression)
 HVACJOBUILD.show()
 HVACJOBUILD.registerTempTable("HVACJBUILD")
```



- Now let's select tempchange and country column, then filter the rows where tempchange is 1 and count the number of occurrence for each country

```
val selective = spark.sql("""select Temp_change_diff, Country from HVACJBUILD WHERE
Temp_change_diff = 1""").toDF()
// we saving the selected fields to variable "selective"
selective.registerTempTable("newselective")
// we will register the above joined table to temptable "newselective"
spark.sql("""select Country, count(Temp_change_diff) from newselective group by
Country""").show()
//we will count the temperature difference across each country
```

NewProj1 [C:\Users\Laptop\IdeaProjects\NewProj1] - ...\src\main\scala\casestuyII.scala [newproj1] - IntelliJ IDEA

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help

NewProj1 src main scala casestuyII.scala

Run: casestuyII

```
18/05/13 16:01:19 INFO ShuffleBlockFetcherIterator: Started 0 remote fetches in 0 ms
18/05/13 16:01:19 INFO Executor: Finished task 59.0 in stage 15.0 (TID 206). 3470 bytes result sent to driver
18/05/13 16:01:19 INFO TaskSetManager: Finished task 59.0 in stage 15.0 (TID 206) in 9 ms on localhost (executor driver) (75/75)
18/05/13 16:01:19 INFO TaskSchedulerImpl: Removed TaskSet 15.0, whose tasks have all completed, from pool
18/05/13 16:01:19 INFO DAGScheduler: ResultStage 15 (show at casestuyII.scala:77) finished in 0.869 s
18/05/13 16:01:19 INFO DAGScheduler: Job 10 finished: show at casestuyII.scala:77, took 0.924789 s
+-----------+----------------------+
|    Country|count(Temp_change_diff)|
+-----------+----------------------+
|  Singapore|                   230|
|     Turkey|                   243|
|    Germany|                   196|
|     France|                   251|
|  Argentina|                   230|
|    Belgium|                   199|
|    Finland|                   473|
|      China|                   241|
|  Hong Kong|                   248|
|     Israel|                   232|
|        USA|                   213|
|     Mexico|                   228|
|  Indonesia|                   243|
|Saudi Arabia|                   233|
|     Canada|                   232|
|     Brazil|                   226|
|  Australia|                   225|
|      Egypt|                   236|
|South Africa|                   237|
+-----------+----------------------+
```

Select tempchange and country column, then filter the rows where tempchange is 1 and count the number of occurrence for each country