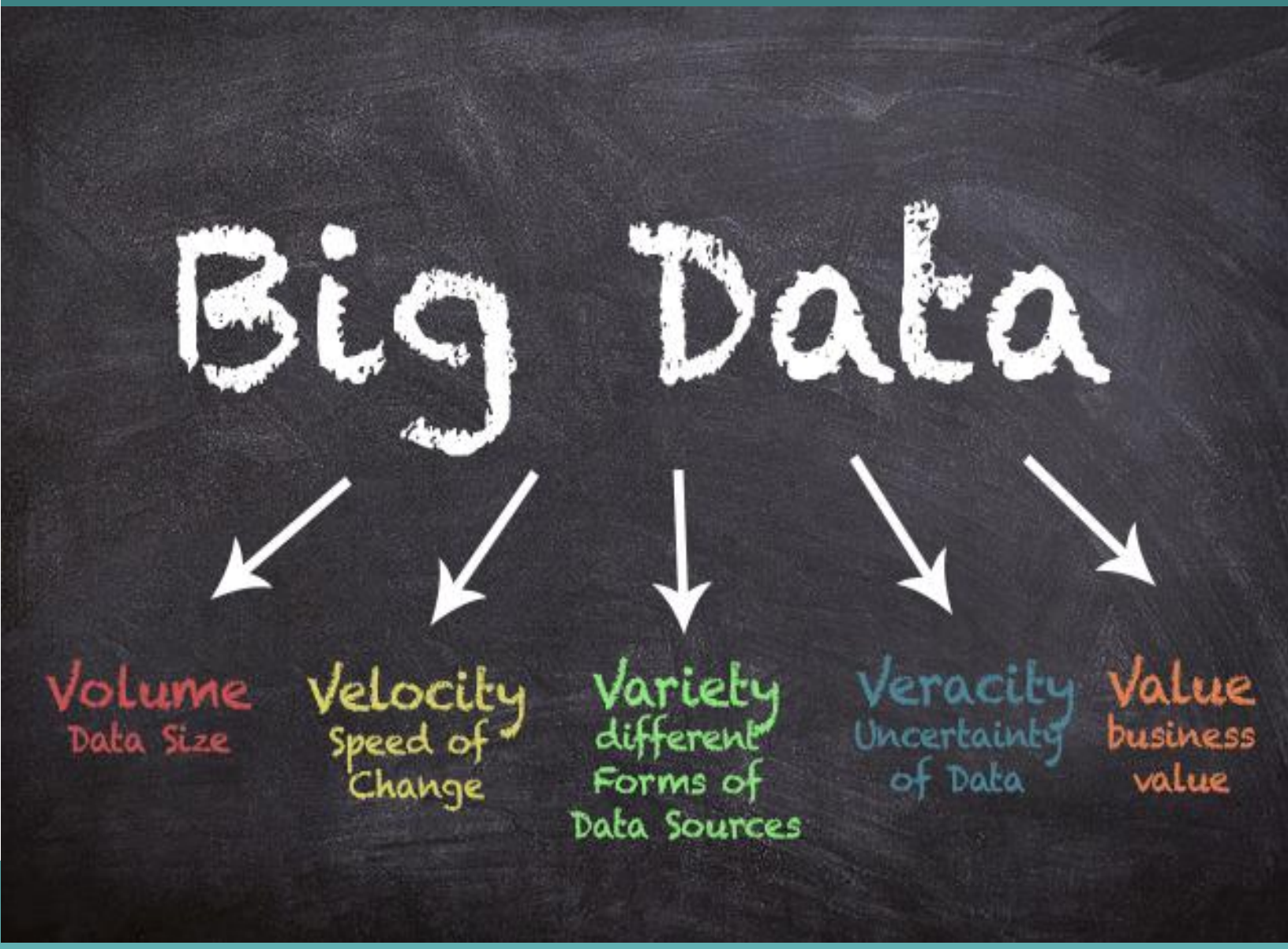
09/12/2018 | By: Ajit K Prasad



Big Data Engineering with Hadoop & Spark

US Hospital Data Analysis
Case Study V

Acadgild

Hospital Data Analysis in US

Case Study V

This Case Study assignment is aimed at consolidating the concepts that was learnt during the various Scala and Apache Spark, Spark SQL session of the course.

Associated Data Files:

Datasets can be downloaded from this ***link***.

Dataset Description:

- ***DRG Definition:*** The code and description identifying the MS-DRG. MS-DRGs are a classification system that groups similar clinical conditions (diagnoses) and procedures furnished by the hospital during their stay.
- ***Provider Id:*** The CMS Certification Number (CCN) assigned to the Medicare-certified hospital facility.
- ***Provider Name:*** The name of the provider.
- ***Provider Street Address:*** The provider's street address.
- ***Provider City:*** The city where the provider is located.
- ***Provider State:*** The state where the provider is located.
- ***Provider Zip Code:*** The provider's zip code.
- ***Provider HRR:*** The Hospital Referral Region (HRR) where the provider is located.
- ***Total Discharges:*** The number of discharges billed by the provider for inpatient hospital services.
- ***Average Covered Charges:*** The provider's average charge for services covered by Medicare for all discharges in the MS-DRG. These will vary from hospital to hospital because of the differences in hospital charge structures.
- ***Average Total Payments:*** The average total payments to all providers for the MS-DRG including the MSDRG amount, teaching, disproportionate share, capital, and outlier payments for all cases. Also included in the average total payments are co-payment and deductible amounts that the patient is responsible for and any additional payments by third parties for coordination of benefits.
- ***Average Medicare Payments:*** The average amount that Medicare pays to the provider for Medicare's share of the MS-DRG. Average Medicare payment amounts include the MS-DRG amount, teaching, disproportionate share, capital, and outlier payments for all cases. Medicare payments DO NOT include beneficiary co-payments and deductible amounts nor any additional payments from third parties for coordination of benefits.

Objectives:

1. Load file into Spark
2. What is the average amount of AverageCoveredCharges per State
    i. Find out the AverageTotalPayments charges per State
    ii. Find out the AverageMedicarePayments charges per State
3. Find out the total number of Discharges per state and for each disease
    i. Sort the output in descending order of totalDischarges

Solution:

1. To load data from ***inpatientCharges.csv*** to Apache Spark, create a manual schema for the files, which would provide the schema while loading data from CSV file, as shown below

```
val ManualSchemaHospital = new StructType(Array(new StructField("DRGDefinition",
StringType,true),
  new StructField("ProviderId", LongType, false),
  new StructField("ProviderName", StringType, true),
  new StructField("ProviderStreetAddress", StringType, false),
  new StructField("ProviderCity", StringType, false),
  new StructField("ProviderState", StringType, false),
  new StructField("ProviderZipCode", LongType, false),
  new StructField("HospitalReferralRegionDescription", StringType, true),
  new StructField("TotalDischarges", LongType, false),
  new StructField("AverageCoveredCharges", DoubleType, false),
  new StructField("AverageTotalPayments", DoubleType, false),
  new StructField("AverageMedicarePayments", DoubleType, 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)>)

2. Now, load the CSV files from local file system to Spark as shown below

```
//Load data from the CSV file
val HospitalData = spark.read.format("csv")
  .option("header", "true")
  .schema(ManualSchemaHospital)
  .load("D:\\AcadGild\\ScalaCaseStudies\\Datasets\\Hospital\\inpatientCharges.csv")
  .toDF()
println("\nInpatient Hospital Data ->> "+HospitalData.count())
println("Hospital Data Loaded and Displayed!")
HospitalData.show()
```

3. The CSV file read format provides various options of which few have been used as follows:
   - Remove the header from the input file
   - Provided the manual schema that we have created in the previous step
   - Provided the path where the CSV file is saved in the local file system

4. Contents of the input file that is loaded into spark is as shown below

Run: HospitalSQL

Spark Session Object Created

Inpatient Hospital Data ->> 163065

Hospital Data Loaded and Displayed!

DRGDefinition	ProviderId	ProviderName	ProviderStreetAddress	ProviderCity	ProviderState	ProviderZipCode	HospitalReferralRegionDescription	TotalDischarges	AverageCoveredCharges	AverageTotalPayments	AverageMedicarePayments
039 - EXTRACRANIA...	10001	SOUTHEAST ALABAMA...	1108 ROSS CLARK C...	DOTHAN	AL	36301	AL - Dothan	91	32963.07	5777.24	4763.73
039 - EXTRACRANIA...	10005	MARSHALL MEDICAL ...	2505 U S HIGHWAY ...	BOAZ	AL	35957	AL - Birmingham	14	15131.85	5787.57	4976.71
039 - EXTRACRANIA...	10006	ELIZA COFFEE MEMO...	205 MARENGO STREET	FLORENCE	AL	35631	AL - Birmingham	24	37560.37	5434.95	4453.79
039 - EXTRACRANIA...	10011	ST VINCENT'S EAST	50 MEDICAL PARK E...	BIRMINGHAM	AL	35235	AL - Birmingham	25	13998.28	5417.56	4129.16
039 - EXTRACRANIA...	10016	SHELBY BAPTIST ME...	1000 FIRST STREET...	ALABASTER	AL	35007	AL - Birmingham	18	31633.27	5658.33	4851.44
039 - EXTRACRANIA...	10023	BAPTIST MEDICAL C...	2105 EAST SOUTH B...	MONTGOMERY	AL	36116	AL - Montgomery	67	16920.79	6653.8	5374.14
039 - EXTRACRANIA...	10029	EAST ALABAMA MEDI...	2000 PEPPERELL PA...	OPELIKA	AL	36801	AL - Birmingham	51	11977.13	5834.74	4761.41
039 - EXTRACRANIA...	10033	UNIVERSITY OF ALA...	619 SOUTH 19TH ST...	BIRMINGHAM	AL	35233	AL - Birmingham	32	35841.09	8031.12	5858.5
039 - EXTRACRANIA...	10039	HUNTSVILLE HOSPITAL	101 SIVLEY RD	HUNTSVILLE	AL	35801	AL - Huntsville	135	28523.39	6113.38	5228.4
039 - EXTRACRANIA...	10040	GADSDEN REGIONAL ...	1007 GOODYEAR AVENUE	GADSDEN	AL	35903	AL - Birmingham	34	75233.38	5541.05	4386.94
039 - EXTRACRANIA...	10046	RIVERVIEW REGIONA...	600 SOUTH THIRD S...	GADSDEN	AL	35901	AL - Birmingham	14	67327.92	5461.57	4493.57
039 - EXTRACRANIA...	10055	FLOWERS HOSPITAL	4370 WEST MAIN ST...	DOTHAN	AL	36305	AL - Dothan	45	39607.28	5356.28	4408.2
039 - EXTRACRANIA...	10056	ST VINCENT'S BIRM...	810 ST VINCENT'S ...	BIRMINGHAM	AL	35205	AL - Birmingham	43	22862.23	5374.65	4186.02
039 - EXTRACRANIA...	10078	NORTHEAST ALABAMA...	400 EAST 10TH STREET	ANNISTON	AL	36207	AL - Birmingham	21	31110.85	5366.23	4376.23
039 - EXTRACRANIA...	10083	SOUTH BALDWIN REG...	1613 NORTH MCKENZ...	FOLEY	AL	36535	AL - Mobile	15	25411.33	5282.93	4383.73
039 - EXTRACRANIA...	10085	DECATUR GENERAL H...	1201 7TH STREET SE	DECATUR	AL	35609	AL - Huntsville	27	9234.51	5676.55	4509.11
039 - EXTRACRANIA...	10090	PROVIDENCE HOSPITAL	6801 AIRPORT BOUL...	MOBILE	AL	36608	AL - Mobile	27	15895.85	5930.11	3972.85
039 - EXTRACRANIA...	10092	D C H REGIONAL ME...	809 UNIVERSITY BO...	TUSCALOOSA	AL	35401	AL - Tuscaloosa	31	19721.16	6192.54	5179.38
039 - EXTRACRANIA...	10100	THOMAS HOSPITAL	750 MORPHY AVENUE	FAIRHOPE	AL	36532	AL - Mobile	18	10710.88	4968.0	3898.88
039 - EXTRACRANIA...	10103	BAPTIST MEDICAL C...	701 PRINCETON AVE...	BIRMINGHAM	AL	35211	AL - Birmingham	33	51343.75	5996.0	4962.45

only showing top 20 rows

5. To calculate the average amount of ***AverageCoveredCharges per State***

i. First create a temporary view named ***"HospitalView"***

ii. Write sql query on the view created to obtain the average amount of ***AverageCoveredCharges***

- In the query we are rounding the average values to 2 decimal points.

```
//Create a view
HospitalData.createOrReplaceTempView("HospitalView")
spark.sql("""select ProviderState, round(avg(AverageCoveredCharges),2)
    |as AvgCoverageCharges_State from HospitalView
    |Group by ProviderState""".stripMargin)
  .show()
println("HospitalView Created and Displayed!\n\n")
```

Run: HospitalSQL

ProviderState	AvgCoverageCharges_State
AZ	41200.06
SC	35862.49
LA	33085.37
MN	27894.36
NJ	66125.69
DC	40116.66
OR	27390.11
VA	29222.0
RI	29942.7
KY	24523.81
WY	28700.6
NH	27059.02
MI	24124.25
NV	61047.12
WI	26149.33
ID	25565.55
CA	67508.62
CT	31318.41
NE	31736.43
MT	22670.02

only showing top 20 rows

HospitalView Created and Displayed!

6. To calculate the ***AverageTotalPayments*** charges ***per State***

- **i.** Using the view created previously, write sql query to obtain the total amount of ***AverageTotalPayments per state***
  - In the query we are rounding the average values to 2 decimal points and we are casting to decimal data type.

```
//Average Total Payments State wise
spark.sql("""select ProviderState, round(sum(cast(AverageTotalPayments as
decimal)/cast(pow(10,2) as decimal)),2)
    |as AvgTotPayment_State from HospitalView
    |Group by ProviderState""".stripMargin)
  .show()
println("State wise Average Total Payments Calculated and Displayed!\n\n")
```

```
Run:  HospitalSQL
+-------------+-------------------+
|ProviderState|AvgTotPayment_State|
+-------------+-------------------+
|           AZ|          289506.28|
|           SC|          260000.46|
|           LA|          261492.69|
|           MN|          224034.81|
|           NJ|          515368.48|
|           DC|           60051.02|
|           OR|          135566.46|
|           VA|          385017.99|
|           RI|           61796.34|
|           KY|          267315.91|
|           WY|           28154.28|
|           NH|           76454.02|
|           MI|          528592.49|
|           NV|          123706.69|
|           WI|          262732.20|
|           ID|           54147.78|
|           CA|         1649942.14|
|           CT|          228559.40|
|           NE|           99102.66|
|           MT|           46819.20|
+-------------+-------------------+
only showing top 20 rows

State wise Average Total Payments Calculated and Displayed!
```

7. To calculate the ***AverageMedicarePayments*** charges ***per State***.

i. Using the view created previously, write sql query to obtain the total amount of ***AverageMedicarePayments per state***

- In the query we are rounding the average values to 2 decimal points and we are casting to decimal data type.

```
//Average Medicare Payments State wise
spark.sql("""select ProviderState, round(sum(cast(AverageMedicarePayments as
decimal)/cast(pow(10,2) as decimal)),2)
    |as AvgMediPayment_State from HospitalView
    |Group by ProviderState""".stripMargin)
  .show()
println("State wise Average Medicare Payments Calculated and Displayed!\n\n")
```

```
Run:  HospitalSQL
+-------------+--------------------+
|ProviderState|AvgMediPayment_State|
+-------------+--------------------+
|           AZ|           251621.67|
|           SC|           224239.61|
|           LA|           223626.26|
|           MN|           194104.94|
|           NJ|           462666.61|
|           DC|            54571.33|
|           OR|           117368.35|
|           VA|           326583.58|
|           RI|            54789.50|
|           KY|           232011.54|
|           WY|            23562.37|
|           NH|            66864.97|
|           MI|           469403.22|
|           NV|           105146.54|
|           WI|           226794.06|
|           ID|            46625.52|
|           CA|          1501628.09|
|           CT|           203203.55|
|           NE|            84881.92|
|           MT|            40384.33|
+-------------+--------------------+
only showing top 20 rows

State wise Average Medicare Payments Calculated and Displayed!
```

8. To calculate the total number of ***Discharges per State*** and for each disease, sort the output in descending order of ***totalDischarges***

i. Using the view created previously, write sql query on the view previously created to obtain the total amount of ***TotalDischares per State*** and ***per disease***.

```
//Total number of Discharges per State and for each disease
//Sort the output in descending order of totalDischarges
spark.sql("""select DRGDefinition,ProviderState, sum(TotalDischarges) as
TotalDischarged from HospitalView
    |Group by DRGDefinition, ProviderState
    |Order by TotalDischarged desc""".stripMargin)
  .show()
println("State wise Total Discharges made for each disease calculated and displayed in
descending!\n\n")
```

Run: HospitalSQL

```
+--------------------+------------+--------------+
|       DRGDefinition|ProviderState|TotalDischarged|
+--------------------+------------+--------------+
|871 - SEPTICEMIA ...|          CA|         34284|
|470 - MAJOR JOINT...|          TX|         30095|
|470 - MAJOR JOINT...|          FL|         29985|
|470 - MAJOR JOINT...|          CA|         29731|
|871 - SEPTICEMIA ...|          TX|         23144|
|871 - SEPTICEMIA ...|          NY|         21970|
|392 - ESOPHAGITIS...|          FL|         21298|
|470 - MAJOR JOINT...|          IL|         20095|
|470 - MAJOR JOINT...|          NY|         19371|
|871 - SEPTICEMIA ...|          FL|         18660|
|690 - KIDNEY & UR...|          TX|         17384|
|392 - ESOPHAGITIS...|          NY|         17337|
|470 - MAJOR JOINT...|          MI|         16847|
|470 - MAJOR JOINT...|          PA|         16712|
|292 - HEART FAILU...|          FL|         16639|
|690 - KIDNEY & UR...|          FL|         16405|
|470 - MAJOR JOINT...|          OH|         16062|
|470 - MAJOR JOINT...|          NC|         15820|
|871 - SEPTICEMIA ...|          IL|         15610|
|871 - SEPTICEMIA ...|          MI|         15548|
+--------------------+------------+--------------+
only showing top 20 rows

State wise Total Discharges made for each disease calculated and displayed in descending!
```