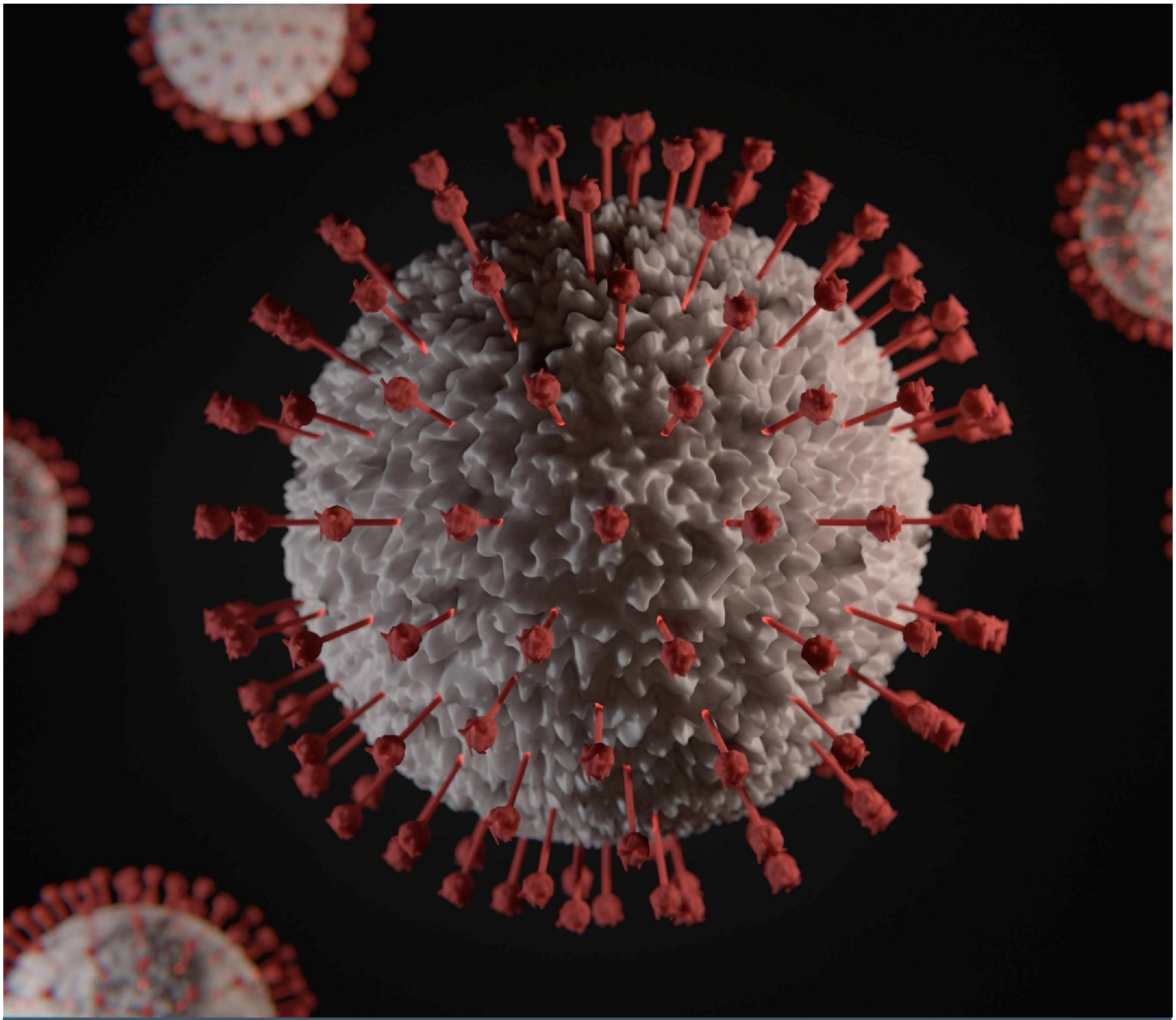


Machine Learning Algorithm Analysis in Spark

City of Toronto – COVID19 cases

Vishal Kaatal 4/23/22 BDAT1008

TABLE OF CONTENTS

INTRODUCTION 2

DATA CLEANING 3

ANALYSIS 4

CONCLUSION 8

REFERENCES 9

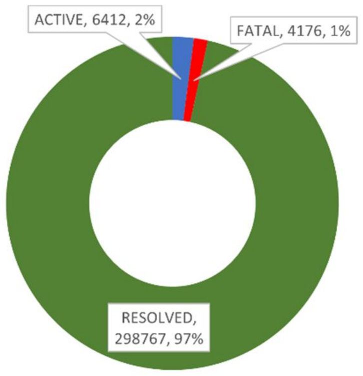
APPENDIX 10

INTRODUCTION

(World Health Organization, 2022) Coronavirus disease, or COVID-19, is an infectious disease that occurs when a person infects another with the SARS-CoV-2 virus. Most people who contract the virus will suffer mild to moderate respiratory symptoms and recover without treatment. COVID-19 is capable of causing severe illness to anyone, even young children. Persons with underlying medical conditions like heart disease, diabetes, chronic respiratory disease, or cancer are more prone to severe illness. Anyone can get sick and die from COVID-19. Stay home and isolate yourself until you feel better if you feel unwell.

(Toronto Public Health, 2022) The COVID19 pandemic has been devastating for hospitals as limited resources can be stretched. Toronto Public Health responds to an ongoing COVID-19 outbreak during a global pandemic. Since the first case was reported in January 2020, this data collection provides demographic, geographic, and severity information for all confirmed and probable cases reported to and managed by Toronto Public Health. Sporadic cases (occurring in the community), as well as outbreak-related, are included. The information is taken from the provincial Case and Contact Management System (CCM). As public health investigations into reported instances and continuous quality improvement measures continue, and as new cases are reported, the information in this analysis is subject to change. The data was extracted on April 14, 2022, for Spark's machine learning algorithm analysis.

The report will cover the machine learning algorithm in Spark that predicts fatalities with its accuracy percentage. It will also cover the cleaning/ balancing of the dataset as 97% of cases have been resolved, as shown in the donut chart.

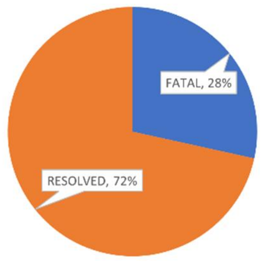


DATA CLEANING

(Toronto Public Health, 2022) The data has been extracted from the provincial Case & Contact Management System by Toronto Public Health. The dataset has 309,355 records, with 18 columns containing various details, out of which only six columns are used for the Machine Learning Algorithm. The selected columns provide the maximum accuracy in various situations: Outbreak Associated, Age Group, Source of Infection, Classification, Client Gender, and Outcome.

The dataset was checked for any null values, and empty records were removed. Since the analysis is regarding the prediction of Outcome as Fatal, all the Active cases were removed. Finally, machine learning only accepts numerical values, which leads to the conversion of the nominal values in the selected columns. The Machine Learning algorithm used in the analysis is Random Forest to overcome the problem of overfitting. A random forest is an ensemble learning algorithm of a decision tree. The report will use this algorithm to build a classification model with Apache Spark using the Google Cloud Platform (GCP).

For the machine-learning algorithm, the dataset was balanced for better results by using the subset of Outcome as Resolved with the subset of Outcome as Fatal as shown in Pie Chart, and then the data was split into 70:30 as training: test. Using the training split data, the prediction was calculated, which was tested against test data for the accuracy of the test. Later, the same predicted training data was tested against an unbalanced dataset to predict the actual accuracy of the Outcome as fatal.



The random forest machine learning code is available in the report's Appendix with the proper steps for cleaning the dataset, balancing, and predicting accuracy.

ANALYSIS

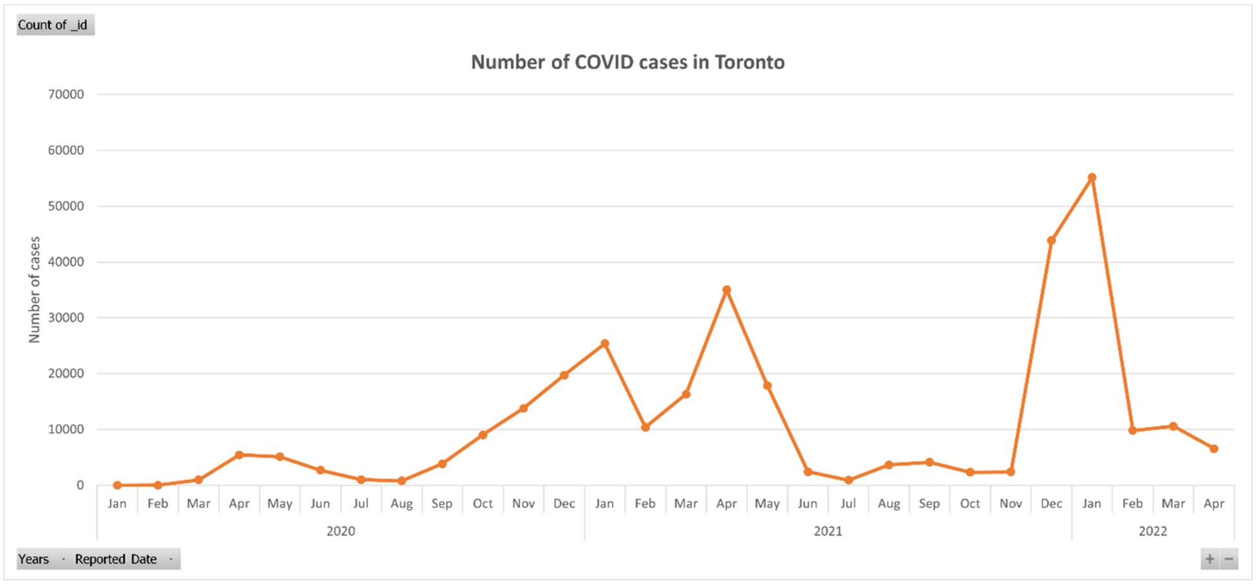


Figure 1: Number of COVID cases in Toronto

The COVID cases in Toronto started in Jan 2020, as shown in figure 1, with a minimal number of cases till March 2020. In April 2020, when the COVID cases started to rise, the provincial government had to lock down to stop the spread throughout the province, which led to a decrease in cases. However, after September 2020, the cases started to rise drastically to 25,000 in January 2021, which dropped in February but again rose in April 2021 to 35,000 cases. During this time, the vaccine started to roll out throughout the world, which led to a decrease in cases. The cases are doing up and down to date, which has yet to end.

Figure 2 depicts the count of cases originating via different types of sources. The community remains the number one source of infection due to the nature of the COVID virus, which spreads massively due to high-touch areas such as Grocery stores, public transport, etc. Then the 2nd place comes as household contact because some people have to go to jobs daily even though it's a pandemic, such as First responders, designated members of the family for groceries, etc., for household activities. Surprisingly, travel-related sources of infection are way less than community transfer. It can be concluded that the virus spreads very quickly through the infected person.

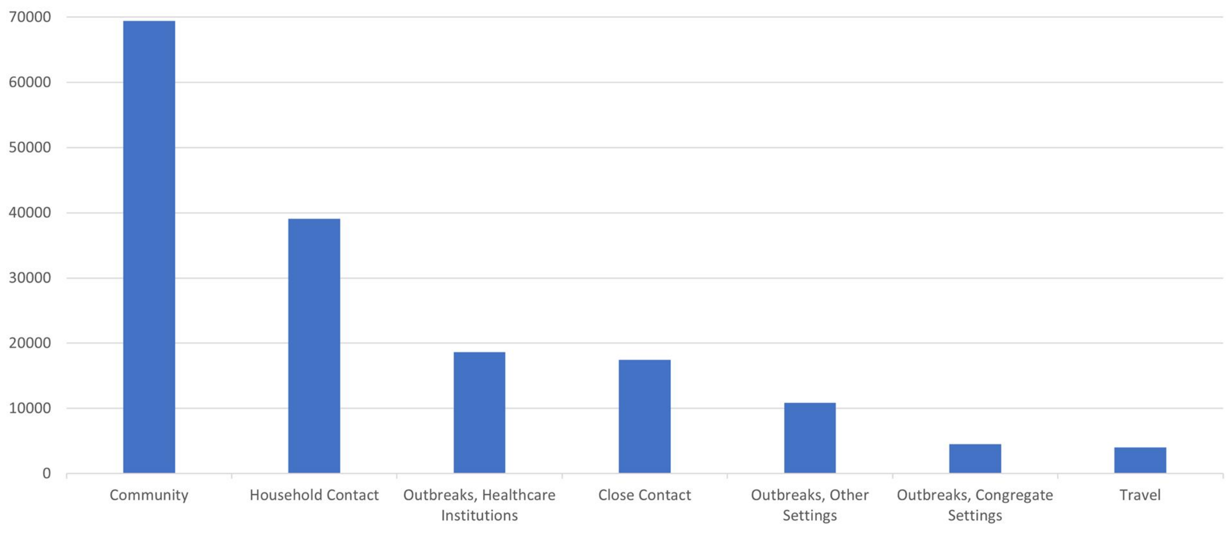


Figure 2: Sources of Infection

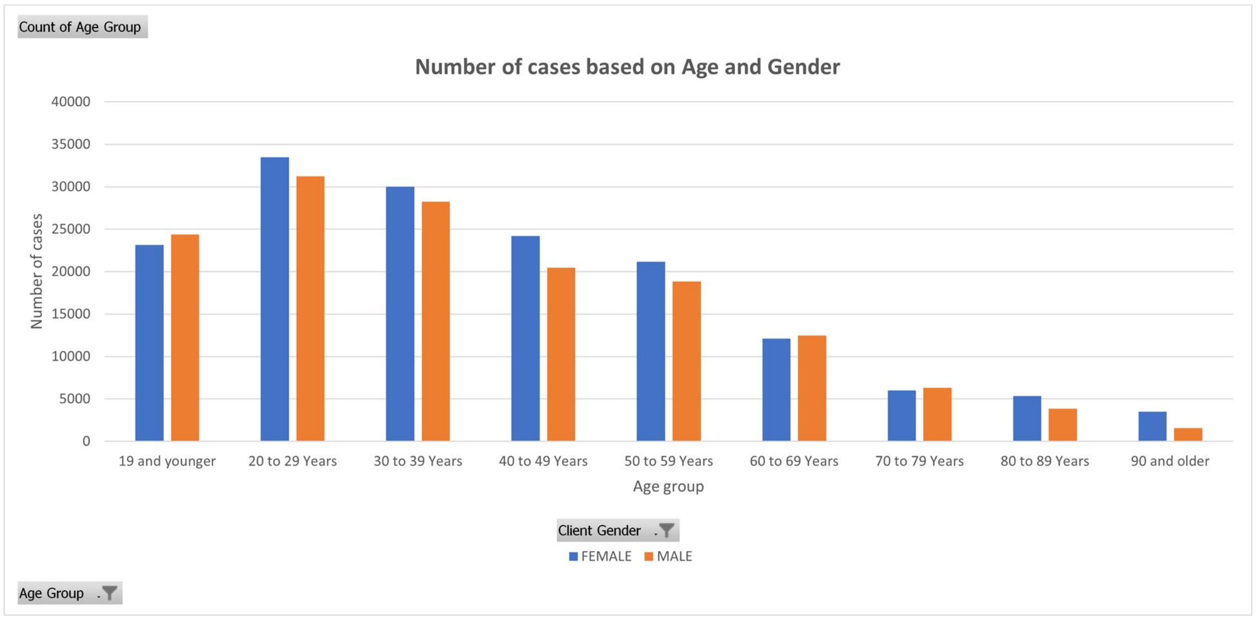


Figure 3: Number of cases in Toronto based on Age and Gender

Figure 3 represents the number of cases in Toronto based on age and gender. The COVID cases cover almost all the age groups starting from 19 years and up. It doesn't matter which gender you belong to; it can affect anyone. A thorough analysis concludes that most of the cases belong to females instead of males throughout the significantly younger age groups.

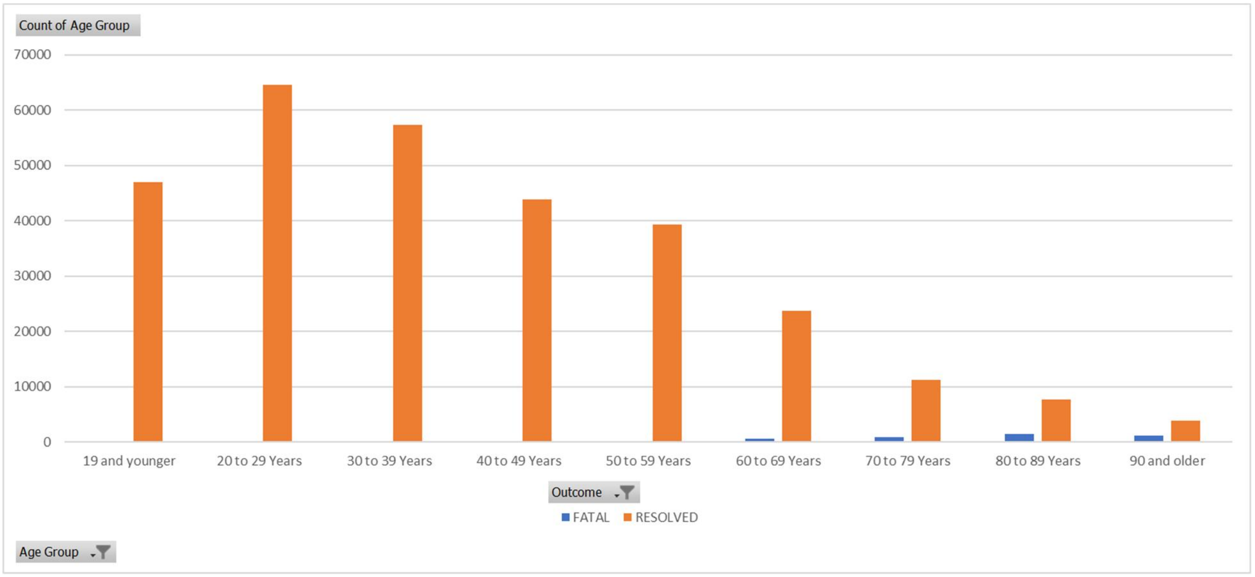
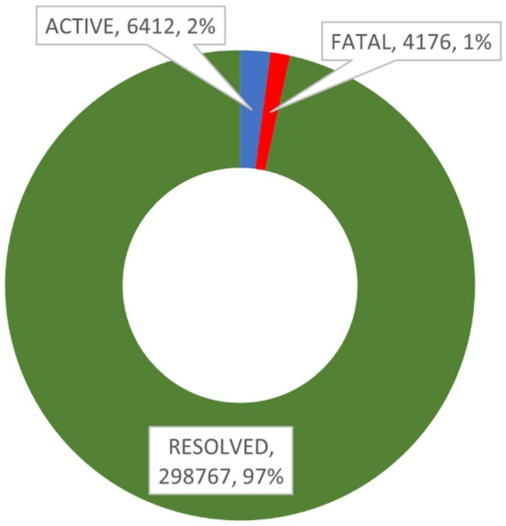


Figure 4: Impact of COVID cases outcome based on Age group



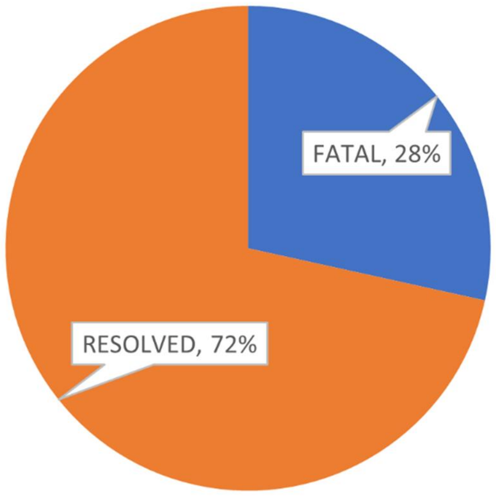


Figure 5: Outcome of the COVID cases: Unbalanced (Left) & Balanced (Right) for ML

Figure 5 depicts the outcome of COVID infection in people. If you catch the COVID virus, it doesn't mean the end of the world, as the data suggests 97% of outcomes resulted as Resolved after two weeks of the period, and only 1% led to fatal. It does not necessarily mean that you should not consider this a pandemic and ignore all the public health safety guidelines. As per figure 4, it can be noticed that most of the Fatal incidence resulted in an older population, of which COVID can be blamed 100% but can be a leading cause due to underlying conditions.

Machine learning can't be directly used on unbalanced cases, as shown in figure 5 (left side). Therefore, the subset of the data is considered, as shown in figure 5 (right side), which is more like a balanced case. The algorithm is available in the Appendix for further details.

CONCLUSION

The coronavirus disease continues to spread across the world, following a trajectory that is difficult to predict. But now, the vaccines are readily available to minimize the impact, and it seems that it can lead to a halt of more spread. Countries' health, humanitarian and socio-economic policies will determine the speed and strength of the recovery/ economy from this pandemic.

The probable causes that would impact the recovery trajectory will be a new & stronger variant and the sense of freedom in people, slowing shifting them towards the side of carelessness and less seriousness about the virus. The government alone cannot bring 100% results, and it is the combined efforts of the government and the people that will result in 100% efforts and results.

Finally, the developed machine learning algorithm can predict any new fatalities based on certain conditions with an accuracy of 91.23%.

REFERENCES

Toronto Public Health. (2022, 04 14). *COVID-19 Cases in Toronto*. Retrieved from Open data - Toronto: https://open.toronto.ca/dataset/covid-19-cases-in-toronto/

World Health Organization. (2022, April 14). Retrieved from https://www.who.int/health-topics/coronavirus#tab=tab_1

APPENDIX

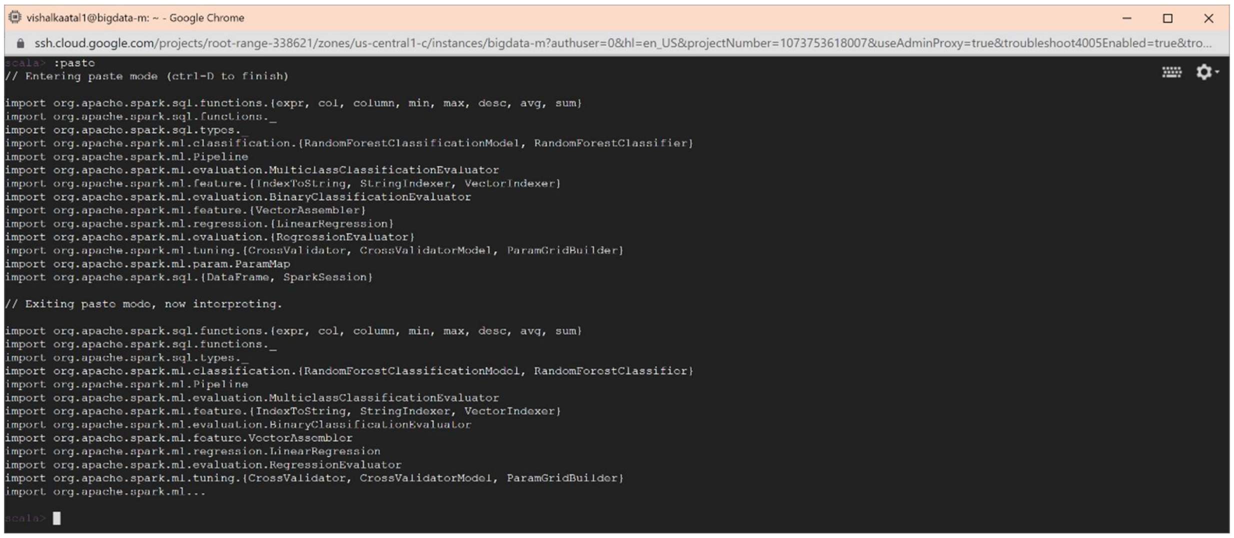
/Loading the Dataset into GCP/
wget https://www.dropbox.com/s/ey2ekrkqe1u1624/COVID19_cases.csv

/Create HDFS directory (if not already done) /
```
hadoop fs -mkdir /BigData
```

/ Load dataset into HDFS /
```
hadoop fs -copyFromLocal COVID19_cases.csv /BigData/.
```

/ Start Spark using the following commands
```
spark-shell --master yarn
```

/Loading the import functions for Scala Spark that are required for Machine Leanning Algorithm & SQL /

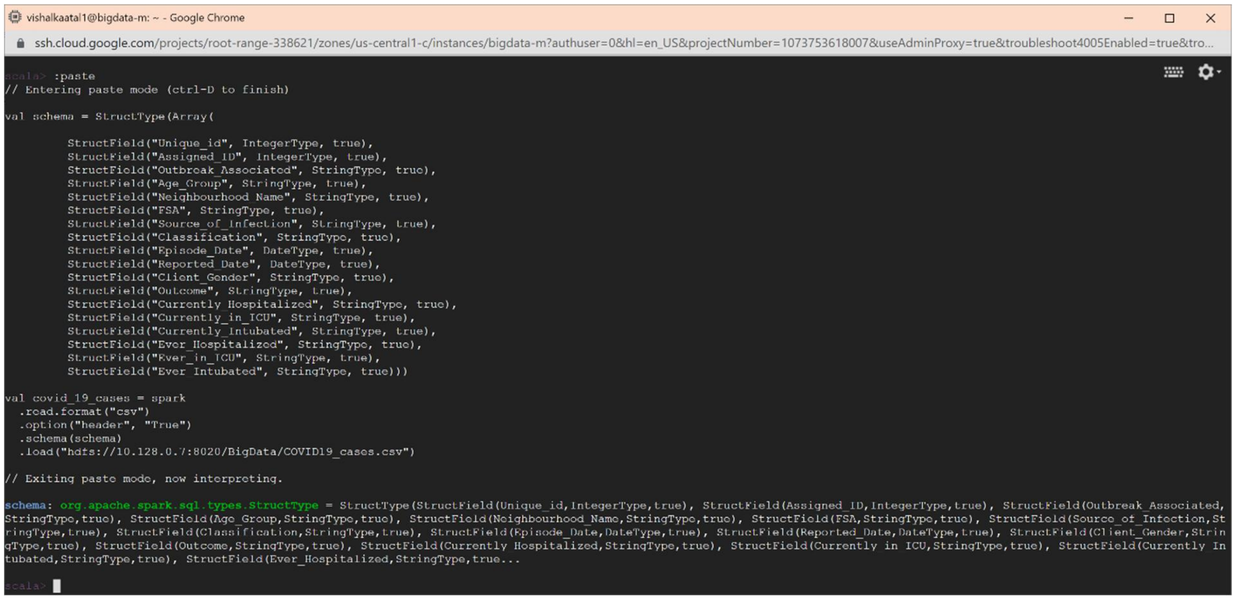


vishalkaatal1@bigdata-m: ~ - Google Chrome

ssh.cloud.google.com/projects/root-range-338621/zones/us-central1-c/instances/bigdata-m?authuser=0&hl=en_US&projectNumber=1073753618007&useAdminProxy=true&troubleshoot4005Enabled=true&tro...

```
scala> :paste
// Entering paste mode (ctrl-D to finish)

import org.apache.spark.sql.functions.{expr, col, column, min, max, desc, avg, sum}
import org.apache.spark.sql.functions._
import org.apache.spark.sql.types._
import org.apache.spark.ml.classification.{RandomForestClassificationModel, RandomForestClassifier}
import org.apache.spark.ml.Pipeline
import org.apache.spark.ml.evaluation.MulticlassClassificationEvaluator
import org.apache.spark.ml.feature.{IndexToString, StringIndexer, VectorIndexer}
import org.apache.spark.ml.evaluation.BinaryClassificationEvaluator
import org.apache.spark.ml.feature.{VectorAssembler}
import org.apache.spark.ml.regression.{LinearRegression}
import org.apache.spark.ml.evaluation.{RegressionEvaluator}
import org.apache.spark.ml.tuning.{CrossValidator, CrossValidatorModel, ParamGridBuilder}
import org.apache.spark.ml.param.ParamMap
import org.apache.spark.sql.{DataFrame, SparkSession}

// Exiting paste mode, now interpreting.

import org.apache.spark.sql.functions.{expr, col, column, min, max, desc, avg, sum}
import org.apache.spark.sql.functions._
import org.apache.spark.sql.types._
import org.apache.spark.ml.classification.{RandomForestClassificationModel, RandomForestClassifier}
import org.apache.spark.ml.Pipeline
import org.apache.spark.ml.evaluation.MulticlassClassificationEvaluator
import org.apache.spark.ml.feature.{IndexToString, StringIndexer, VectorIndexer}
import org.apache.spark.ml.evaluation.BinaryClassificationEvaluator
import org.apache.spark.ml.feature.VectorAssembler
import org.apache.spark.ml.regression.LinearRegression
import org.apache.spark.ml.evaluation.RegressionEvaluator
import org.apache.spark.ml.tuning.{CrossValidator, CrossValidatorModel, ParamGridBuilder}
import org.apache.spark.ml...

scala>
```

/*Creation of a COVID19_cases dataset schema and Load the dataset into Spark from HDFS */

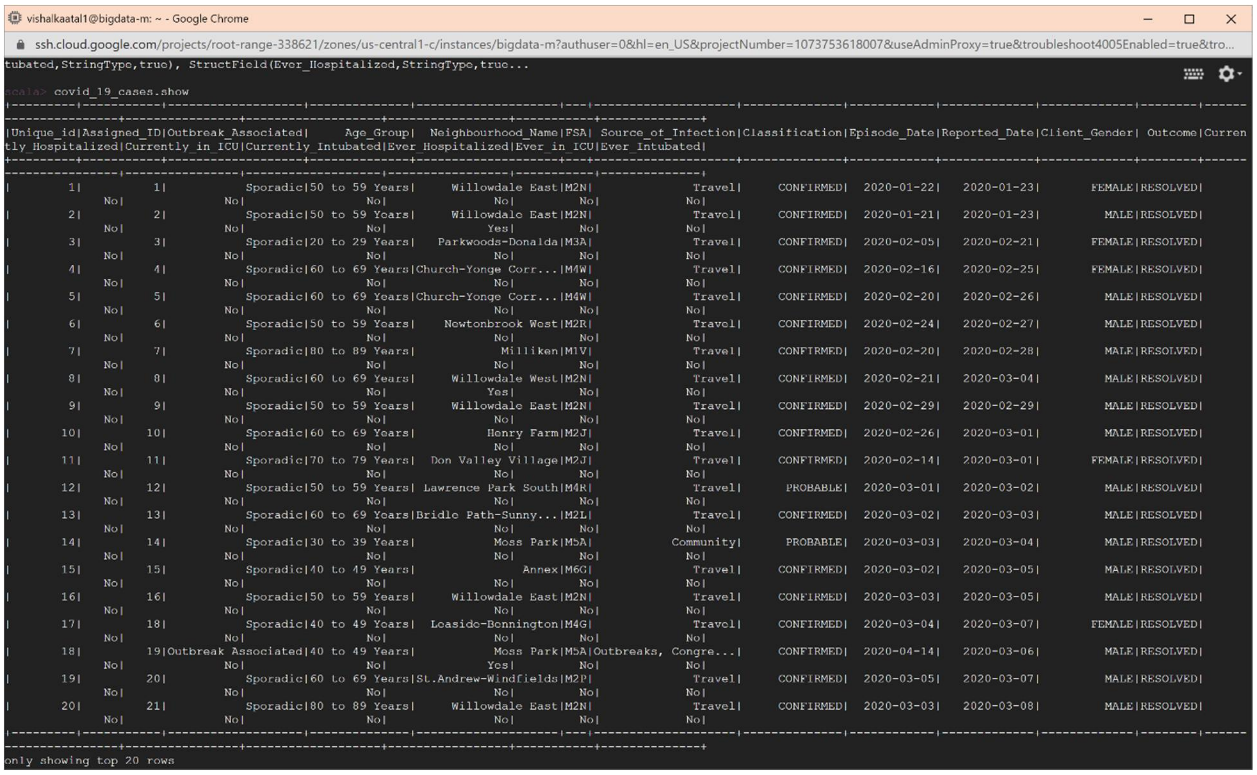


vishalkaatal1@bigdata-m: ~ - Google Chrome

ssh.cloud.google.com/projects/root-range-338621/zones/us-central1-c/instances/bigdata-m?authuser=0&hl=en_US&projectNumber=1073753618007&useAdminProxy=true&troubleshoot4005Enabled=true&tro...

```
scala> :paste
// Entering paste mode (ctrl-D to finish)

val schema = StructType(Array(

         StructField("Unique_id", IntegerType, true),
         StructField("Assigned_ID", IntegerType, true),
         StructField("Outbreak_Associated", StringType, true),
         StructField("Age_Group", StringType, true),
         StructField("Neighbourhood Name", StringType, true),
         StructField("FSA", StringType, true),
         StructField("Source_of_Infection", StringType, true),
         StructField("Classification", StringType, true),
         StructField("Episode_Date", DateType, true),
         StructField("Reported_Date", DateType, true),
         StructField("Client_Gender", StringType, true),
         StructField("Outcome", StringType, true),
         StructField("Currently Hospitalized", StringType, true),
         StructField("Currently_in_ICU", StringType, true),
         StructField("Currently_Intubated", StringType, true),
         StructField("Ever_Hospitalized", StringType, true),
         StructField("Ever_in_ICU", StringType, true),
         StructField("Ever Intubated", StringType, true)))

val covid_19_cases = spark
  .read.format("csv")
  .option("header", "True")
  .schema(schema)
  .load("hdfs://10.128.0.7:8020/BigData/COVID19_cases.csv")

// Exiting paste mode, now interpreting.

schema: org.apache.spark.sql.types.StructType = StructType(StructField(Unique_id,IntegerType,true), StructField(Assigned_ID,IntegerType,true), StructField(Outbreak_Associated,
StringType,true), StructField(Age_Group,StringType,true), StructField(Neighbourhood_Name,StringType,true), StructField(FSA,StringType,true), StructField(Source_of_Infection,St
ringType,true), StructField(Classification,StringType,true), StructField(Episode_Date,DateType,true), StructField(Reported_Date,DateType,true), StructField(Client_Gender,Strin
gType,true), StructField(Outcome,StringType,true), StructField(Currently Hospitalized,StringType,true), StructField(Currently in ICU,StringType,true), StructField(Currently In
tubated,StringType,true), StructField(Ever_Hospitalized,StringType,true...

scala>
```

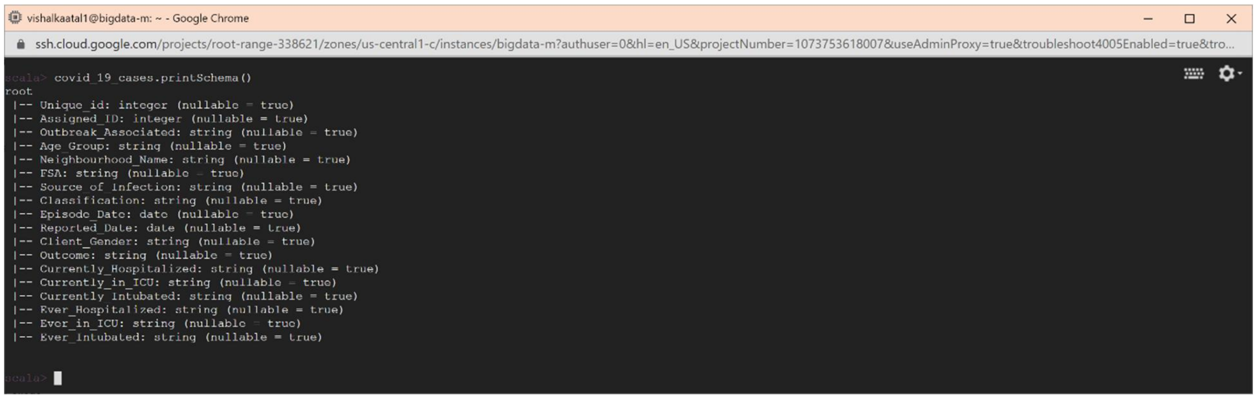
/* Display of top 20 rows in the dataset as per the schema created in the spark */



vishalkaatal1@bigdata-m: ~ - Google Chrome

ssh.cloud.google.com/projects/root-range-338621/zones/us-central1-c/instances/bigdata-m?authuser=0&hl=en_US&projectNumber=1073753618007&useAdminProxy=true&troubleshoot4005Enabled=true&tro...

```
tubated,StringType,true), StructField(Ever_Hospitalized,StringType,true...

scala> covid_19_cases.show
+---------+-----------+------------------+-------------+-------------------+---+-------------------+--------------+------------+-------------+-------------+--------+------
---------------+------------------+-----------------+-----------------+-----------+-------------+
|Unique_id|Assigned_ID|Outbreak_Associated|    Age_Group|  Neighbourhood_Name|FSA| Source_of_Infection|Classification|Episode_Date|Reported_Date|Client_Gender| Outcome|Curren
tly_Hospitalized|Currently_in_ICU|Currently_Intubated|Ever_Hospitalized|Ever_in_ICU|Ever_Intubated|
+---------+-----------+------------------+-------------+-------------------+---+-------------------+--------------+------------+-------------+-------------+--------+------
---------------+------------------+-----------------+-----------------+-----------+-------------+
|        1|          1|          Sporadic|50 to 59 Years|    Willowdale East|M2N|             Travel|     CONFIRMED|  2020-01-22|   2020-01-23|       FEMALE|RESOLVED|
            No|              No|                 No|               No|         No|            No|
|        2|          2|          Sporadic|50 to 59 Years|    Willowdale East|M2N|             Travel|     CONFIRMED|  2020-01-21|   2020-01-23|         MALE|RESOLVED|
            No|              No|                 No|              Yes|         No|            No|
|        3|          3|          Sporadic|20 to 29 Years|   Parkwoods-Donalda|M3A|             Travel|     CONFIRMED|  2020-02-05|   2020-02-21|       FEMALE|RESOLVED|
            No|              No|                 No|               No|         No|            No|
|        4|          4|          Sporadic|60 to 69 Years|Church-Yonge Corr...|M4W|             Travel|     CONFIRMED|  2020-02-16|   2020-02-25|       FEMALE|RESOLVED|
            No|              No|                 No|               No|         No|            No|
|        5|          5|          Sporadic|60 to 69 Years|Church-Yonge Corr...|M4W|             Travel|     CONFIRMED|  2020-02-20|   2020-02-26|         MALE|RESOLVED|
            No|              No|                 No|               No|         No|            No|
|        6|          6|          Sporadic|50 to 59 Years|   Newtonbrook West|M2R|             Travel|     CONFIRMED|  2020-02-24|   2020-02-27|         MALE|RESOLVED|
            No|              No|                 No|               No|         No|            No|
|        7|          7|          Sporadic|80 to 89 Years|           Milliken|M1V|             Travel|     CONFIRMED|  2020-02-20|   2020-02-28|         MALE|RESOLVED|
            No|              No|                 No|               No|         No|            No|
|        8|          8|          Sporadic|60 to 69 Years|    Willowdale West|M2N|             Travel|     CONFIRMED|  2020-02-21|   2020-03-04|         MALE|RESOLVED|
            No|              No|                 No|              Yes|         No|            No|
|        9|          9|          Sporadic|50 to 59 Years|    Willowdale East|M2N|             Travel|     CONFIRMED|  2020-02-29|   2020-02-29|         MALE|RESOLVED|
            No|              No|                 No|               No|         No|            No|
|       10|         10|          Sporadic|60 to 69 Years|         Henry Farm|M2J|             Travel|     CONFIRMED|  2020-02-26|   2020-03-01|         MALE|RESOLVED|
            No|              No|                 No|               No|         No|            No|
|       11|         11|          Sporadic|70 to 79 Years| Don Valley Village|M2J|             Travel|     CONFIRMED|  2020-02-14|   2020-03-01|       FEMALE|RESOLVED|
            No|              No|                 No|               No|         No|            No|
|       12|         12|          Sporadic|50 to 59 Years| Lawrence Park South|M4R|             Travel|      PROBABLE|  2020-03-01|   2020-03-02|         MALE|RESOLVED|
            No|              No|                 No|               No|         No|            No|
|       13|         13|          Sporadic|60 to 69 Years|Bridle Path-Sunny...|M2L|             Travel|     CONFIRMED|  2020-03-02|   2020-03-03|         MALE|RESOLVED|
            No|              No|                 No|               No|         No|            No|
|       14|         14|          Sporadic|30 to 39 Years|          Moss Park|M5A|          Community|      PROBABLE|  2020-03-03|   2020-03-04|         MALE|RESOLVED|
            No|              No|                 No|               No|         No|            No|
|       15|         15|          Sporadic|40 to 49 Years|              Annex|M6G|             Travel|     CONFIRMED|  2020-03-02|   2020-03-05|         MALE|RESOLVED|
            No|              No|                 No|               No|         No|            No|
|       16|         16|          Sporadic|50 to 59 Years|    Willowdale East|M2N|             Travel|     CONFIRMED|  2020-03-03|   2020-03-05|         MALE|RESOLVED|
            No|              No|                 No|               No|         No|            No|
|       17|         18|          Sporadic|40 to 49 Years| Leaside-Bennington|M4G|             Travel|     CONFIRMED|  2020-03-04|   2020-03-07|       FEMALE|RESOLVED|
            No|              No|                 No|               No|         No|            No|
|       18|         19|Outbreak Associated|40 to 49 Years|          Moss Park|M5A|Outbreaks, Congre...|     CONFIRMED|  2020-04-14|   2020-03-06|         MALE|RESOLVED|
            No|              No|                 No|              Yes|         No|            No|
|       19|         20|          Sporadic|60 to 69 Years|St.Andrew-Windfields|M2P|             Travel|     CONFIRMED|  2020-03-05|   2020-03-07|         MALE|RESOLVED|
            No|              No|                 No|               No|         No|            No|
|       20|         21|          Sporadic|80 to 89 Years|    Willowdale East|M2N|             Travel|     CONFIRMED|  2020-03-03|   2020-03-08|         MALE|RESOLVED|
            No|              No|                 No|               No|         No|            No|
+---------+-----------+------------------+-------------+-------------------+---+-------------------+--------------+------------+-------------+-------------+--------+------
---------------+------------------+-----------------+-----------------+-----------+-------------+
only showing top 20 rows
```

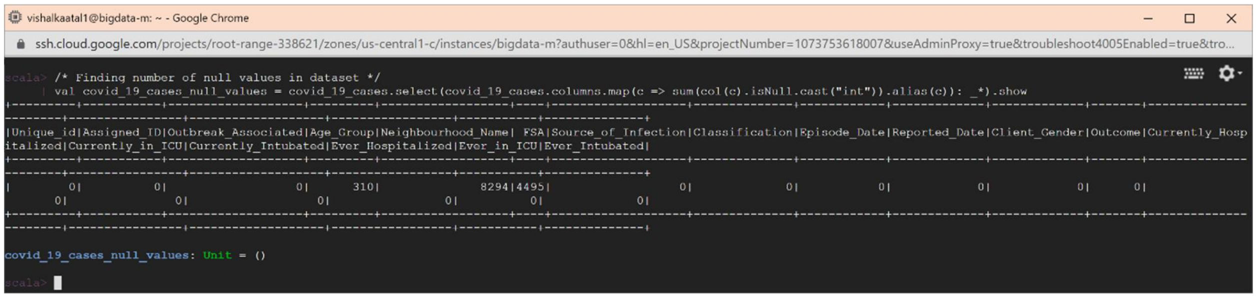


vishalkaatal1@bigdata-m: ~ - Google Chrome

ssh.cloud.google.com/projects/root-range-338621/zones/us-central1-c/instances/bigdata-m?authuser=0&hl=en_US&projectNumber=1073753618007&useAdminProxy=true&troubleshoot4005Enabled=true&tro...

```
scala> covid_19_cases.printSchema()
root
 |-- Unique_id: integer (nullable = true)
 |-- Assigned_ID: integer (nullable = true)
 |-- Outbreak_Associated: string (nullable = true)
 |-- Age_Group: string (nullable = true)
 |-- Neighbourhood_Name: string (nullable = true)
 |-- FSA: string (nullable = true)
 |-- Source_of_Infection: string (nullable = true)
 |-- Classification: string (nullable = true)
 |-- Episode_Date: date (nullable = true)
 |-- Reported_Date: date (nullable = true)
 |-- Client_Gender: string (nullable = true)
 |-- Outcome: string (nullable = true)
 |-- Currently_Hospitalized: string (nullable = true)
 |-- Currently_in_ICU: string (nullable = true)
 |-- Currently_Intubated: string (nullable = true)
 |-- Ever_Hospitalized: string (nullable = true)
 |-- Ever_in_ICU: string (nullable = true)
 |-- Ever_Intubated: string (nullable = true)


scala> 
```

/*Checking for null values in the dataset */

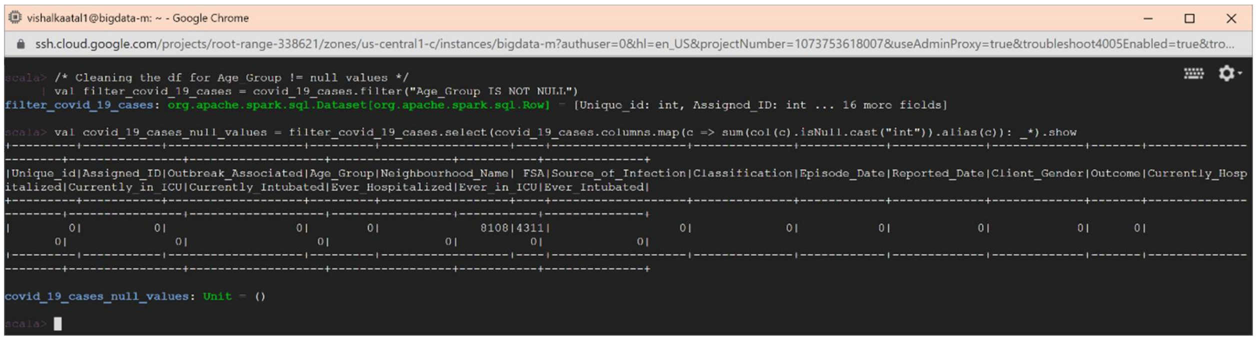


vishalkaatal1@bigdata-m: ~ - Google Chrome

ssh.cloud.google.com/projects/root-range-338621/zones/us-central1-c/instances/bigdata-m?authuser=0&hl=en_US&projectNumber=1073753618007&useAdminProxy=true&troubleshoot4005Enabled=true&tro...

```
scala> /* Finding number of null values in dataset */
     | val covid_19_cases_null_values = covid_19_cases.select(covid_19_cases.columns.map(c => sum(col(c).isNull.cast("int")).alias(c)): _*).show
+---------+-----------+-------------------+---------+------------------+----+-------------------+--------------+------------+-------------+-------------+-------+-------------
--------+----------------+-------------------+-----------------+-----------+--------------+
|Unique_id|Assigned_ID|Outbreak_Associated|Age_Group|Neighbourhood_Name| FSA|Source_of_Infection|Classification|Episode_Date|Reported_Date|Client_Gender|Outcome|Currently_Hosp
italized|Currently_in_ICU|Currently_Intubated|Ever_Hospitalized|Ever_in_ICU|Ever_Intubated|
+---------+-----------+-------------------+---------+------------------+----+-------------------+--------------+------------+-------------+-------------+-------+-------------
--------+----------------+-------------------+-----------------+-----------+--------------+
|        0|          0|                  0|      310|              8294|4495|                  0|             0|           0|            0|            0|      0|
       0|               0|                  0|                0|          0|             0|
+---------+-----------+-------------------+---------+------------------+----+-------------------+--------------+------------+-------------+-------------+-------+-------------
--------+----------------+-------------------+-----------------+-----------+--------------+

covid_19_cases_null_values: Unit = ()

scala> 
```

As it can be noticed from the above picture that Age_Group, Neighbourhood_Name and FSA have some null values. In this report, the machine learning algorithm will consider only the Age_Group column; therefore, only the null values from this particular column will be removed.

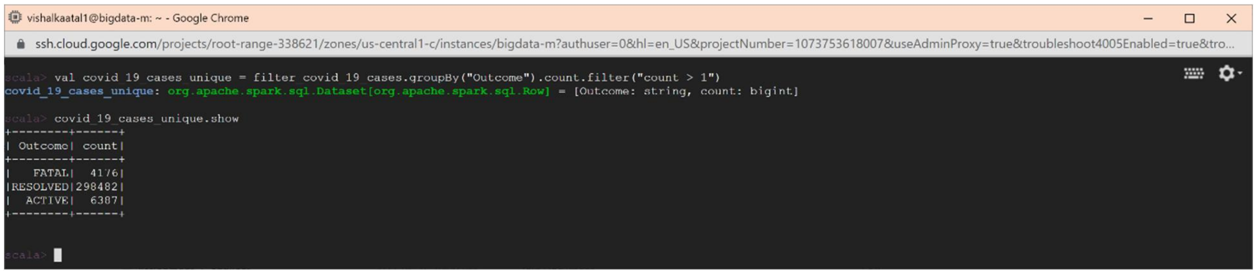


vishalkaatal1@bigdata-m: ~ - Google Chrome

ssh.cloud.google.com/projects/root-range-338621/zones/us-central1-c/instances/bigdata-m?authuser=0&hl=en_US&projectNumber=1073753618007&useAdminProxy=true&troubleshoot4005Enabled=true&tro...

```
scala> /* Cleaning the df for Age Group != null values */
     | val filter_covid_19_cases = covid_19_cases.filter("Age_Group IS NOT NULL")
filter_covid_19_cases: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [Unique_id: int, Assigned_ID: int ... 16 more fields]

scala> val covid_19_cases_null_values = filter_covid_19_cases.select(covid_19_cases.columns.map(c => sum(col(c).isNull.cast("int")).alias(c)): _*).show
+---------+-----------+-------------------+---------+------------------+----+-------------------+--------------+------------+-------------+-------------+-------+--------------
--------+----------------+-------------------+-----------------+-----------+--------------+
|Unique_id|Assigned_ID|Outbreak_Associated|Age_Group|Neighbourhood_Name| FSA|Source_of_Infection|Classification|Episode_Date|Reported_Date|Client_Gender|Outcome|Currently_Hosp
italized|Currently_in_ICU|Currently_Intubated|Ever_Hospitalized|Ever_in_ICU|Ever_Intubated|
+---------+-----------+-------------------+---------+------------------+----+-------------------+--------------+------------+-------------+-------------+-------+--------------
--------+----------------+-------------------+-----------------+-----------+--------------+
|        0|          0|                  0|        0|              8108|4311|                  0|             0|           0|            0|            0|      0|              
       0|               0|                  0|                0|          0|             0|
+---------+-----------+-------------------+---------+------------------+----+-------------------+--------------+------------+-------------+-------------+-------+--------------
--------+----------------+-------------------+-----------------+-----------+--------------+

covid_19_cases_null_values: Unit = ()

scala> 
```

/* Checking the distribution of the Outcome column in the dataset, which will be our Target variable */

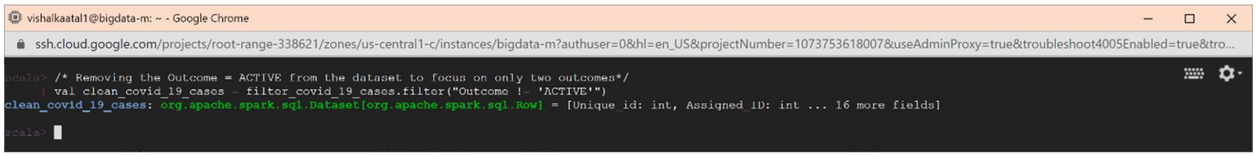


vishalkaatal1@bigdata-m: ~ - Google Chrome

ssh.cloud.google.com/projects/root-range-338621/zones/us-central1-c/instances/bigdata-m?authuser=0&hl=en_US&projectNumber=1073753618007&useAdminProxy=true&troubleshoot4005Enabled=true&tro...

```
scala> val covid_19_cases_unique = filter_covid_19_cases.groupBy("Outcome").count.filter("count > 1")
covid_19_cases_unique: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [Outcome: string, count: bigint]

scala> covid_19_cases_unique.show
+--------+------+
| Outcome| count|
+--------+------+
|   FATAL|  4176|
|RESOLVED|298482|
|  ACTIVE|  6387|
+--------+------+


scala> 
```

Here, it can be noticed that the Outcome consists of ACTIVE cases, RESOLVED cases and FATAL cases. The report is focused on the accurate prediction of Outcomes as FATAL cases; therefore, all the ACTIVE cases will be removed from the dataset.



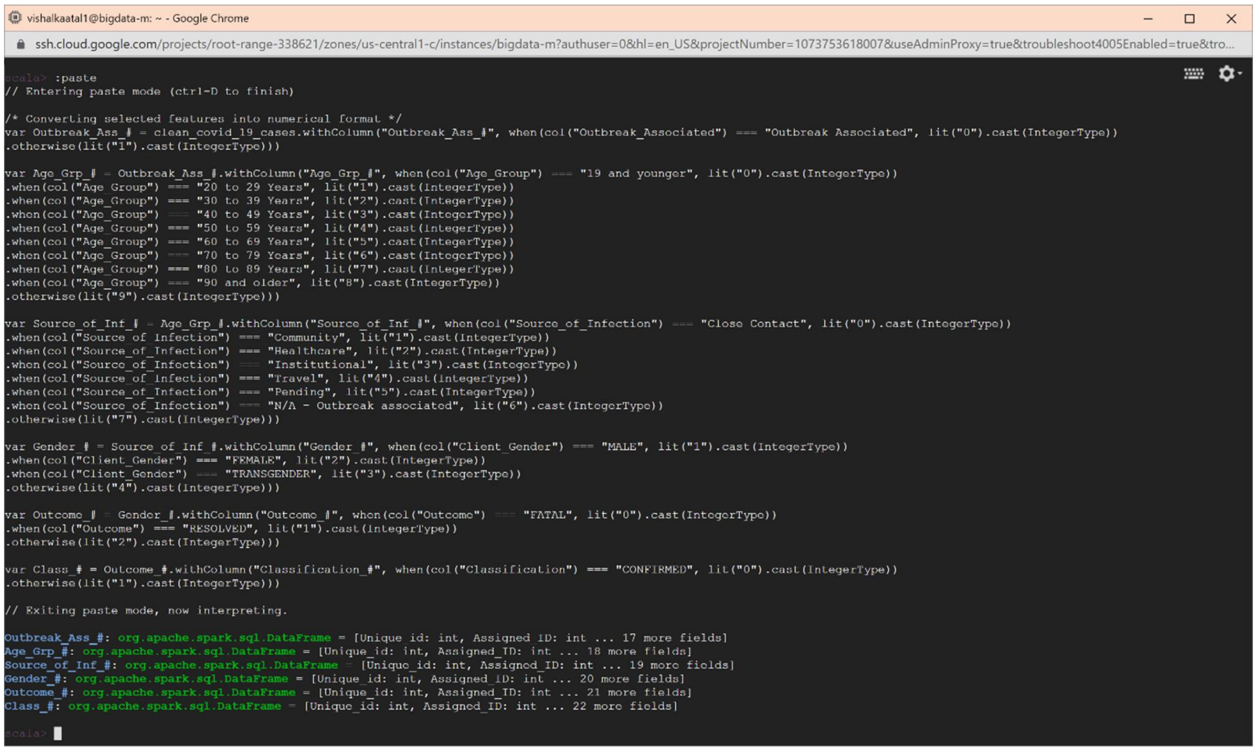
vishalkaatal1@bigdata-m: ~ - Google Chrome

ssh.cloud.google.com/projects/root-range-338621/zones/us-central1-c/instances/bigdata-m?authuser=0&hl=en_US&projectNumber=1073753618007&useAdminProxy=true&troubleshoot4005Enabled=true&tro...

```
scala> /* Removing the Outcome = ACTIVE from the dataset to focus on only two outcomes*/
     | val clean_covid_19_cases = filter_covid_19_cases.filter("Outcome != 'ACTIVE'")
clean_covid_19_cases: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [Unique_id: int, Assigned_ID: int ... 16 more fields]

scala> 
```

/* Now, the selected columns for Machine Learning Algorithm are converted into Numerical */

The ML algorithm, such as Random Forest, can only take numerical values. Therefore, the nominal column values were converted as per the following picture. It is the transformation of columns.

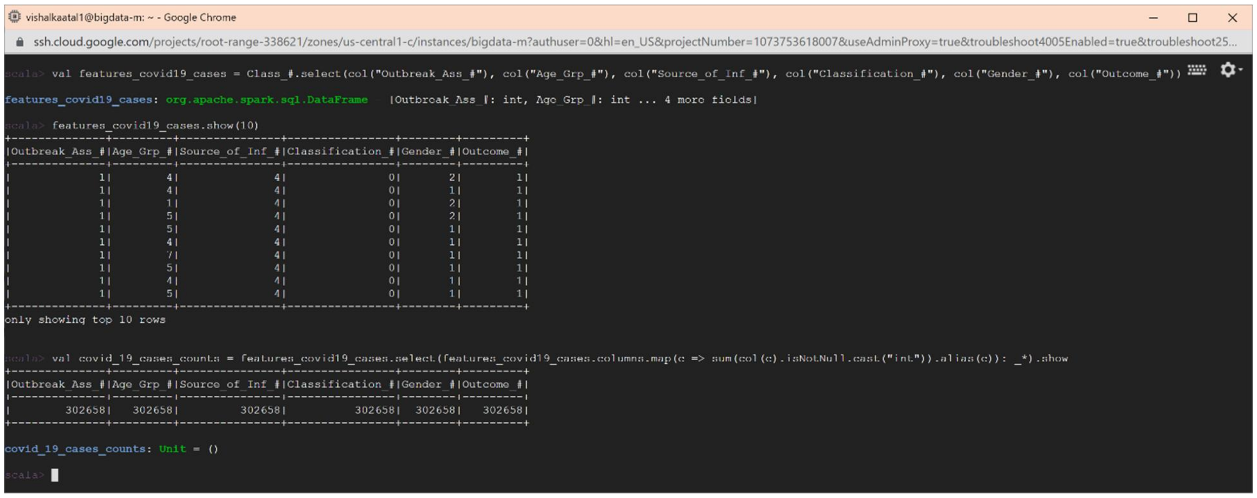


vishalkaatal1@bigdata-m: ~ - Google Chrome

ssh.cloud.google.com/projects/root-range-338621/zones/us-central1-c/instances/bigdata-m?authuser=0&hl=en_US&projectNumber=1073753618007&useAdminProxy=true&troubleshoot4005Enabled=true&tro...

```
scala> :paste
// Entering paste mode (ctrl-D to finish)

/* Converting selected features into numerical format */
var Outbreak_Ass_# = clean_covid_19_cases.withColumn("Outbreak_Ass_#", when(col("Outbreak_Associated") === "Outbreak Associated", lit("0").cast(IntegerType))
.otherwise(lit("1").cast(IntegerType)))

var Age_Grp_# = Outbreak_Ass_#.withColumn("Age_Grp_#", when(col("Age_Group") === "19 and younger", lit("0").cast(IntegerType))
.when(col("Age_Group") === "20 to 29 Years", lit("1").cast(IntegerType))
.when(col("Age_Group") === "30 to 39 Years", lit("2").cast(IntegerType))
.when(col("Age_Group") === "40 to 49 Years", lit("3").cast(IntegerType))
.when(col("Age_Group") === "50 to 59 Years", lit("4").cast(IntegerType))
.when(col("Age_Group") === "60 to 69 Years", lit("5").cast(IntegerType))
.when(col("Age_Group") === "70 to 79 Years", lit("6").cast(IntegerType))
.when(col("Age_Group") === "80 to 89 Years", lit("7").cast(IntegerType))
.when(col("Age_Group") === "90 and older", lit("8").cast(IntegerType))
.otherwise(lit("9").cast(IntegerType)))

var Source_of_Inf_# = Age_Grp_#.withColumn("Source_of_Inf_#", when(col("Source_of_Infection") === "Close Contact", lit("0").cast(IntegerType))
.when(col("Source_of_Infection") === "Community", lit("1").cast(IntegerType))
.when(col("Source_of_Infection") === "Healthcare", lit("2").cast(IntegerType))
.when(col("Source_of_Infection") === "Institutional", lit("3").cast(IntegerType))
.when(col("Source_of_Infection") === "Travel", lit("4").cast(IntegerType))
.when(col("Source_of_Infection") === "Pending", lit("5").cast(IntegerType))
.when(col("Source_of_Infection") === "N/A - Outbreak associated", lit("6").cast(IntegerType))
.otherwise(lit("7").cast(IntegerType)))

var Gender_# = Source_of_Inf_#.withColumn("Gender_#", when(col("Client_Gender") === "MALE", lit("1").cast(IntegerType))
.when(col("Client_Gender") === "FEMALE", lit("2").cast(IntegerType))
.when(col("Client_Gender") === "TRANSGENDER", lit("3").cast(IntegerType))
.otherwise(lit("4").cast(IntegerType)))

var Outcome_# = Gender_#.withColumn("Outcome_#", when(col("Outcome") === "FATAL", lit("0").cast(IntegerType))
.when(col("Outcome") === "RESOLVED", lit("1").cast(IntegerType))
.otherwise(lit("2").cast(IntegerType)))

var Class_# = Outcome_#.withColumn("Classification_#", when(col("Classification") === "CONFIRMED", lit("0").cast(IntegerType))
.otherwise(lit("1").cast(IntegerType)))

// Exiting paste mode, now interpreting.

Outbreak_Ass_#: org.apache.spark.sql.DataFrame = [Unique_id: int, Assigned_ID: int ... 17 more fields]
Age_Grp_#: org.apache.spark.sql.DataFrame = [Unique_id: int, Assigned_ID: int ... 18 more fields]
Source_of_Inf_#: org.apache.spark.sql.DataFrame = [Unique_id: int, Assigned_ID: int ... 19 more fields]
Gender_#: org.apache.spark.sql.DataFrame = [Unique_id: int, Assigned_ID: int ... 20 more fields]
Outcome_#: org.apache.spark.sql.DataFrame = [Unique_id: int, Assigned_ID: int ... 21 more fields]
Class_#: org.apache.spark.sql.DataFrame = [Unique_id: int, Assigned_ID: int ... 22 more fields]

scala>
```

/* New Dataframe with the unbalanced dataset is created for running the ML algorithm with numerical values */



vishalkaatal1@bigdata-m: ~ - Google Chrome

ssh.cloud.google.com/projects/root-range-338621/zones/us-central1-c/instances/bigdata-m?authuser=0&hl=en_US&projectNumber=1073753618007&useAdminProxy=true&troubleshoot4005Enabled=true&troubleshoot25...

```
scala> val features_covid19_cases = Class_#.select(col("Outbreak_Ass_#"), col("Age_Grp_#"), col("Source_of_Inf_#"), col("Classification_#"), col("Gender_#"), col("Outcome_#"))

features_covid19_cases: org.apache.spark.sql.DataFrame = [Outbreak_Ass_#: int, Age_Grp_#: int ... 4 more fields]

scala> features_covid19_cases.show(10)
+--------------+---------+---------------+----------------+--------+---------+
|Outbreak_Ass_#|Age_Grp_#|Source_of_Inf_#|Classification_#|Gender_#|Outcome_#|
+--------------+---------+---------------+----------------+--------+---------+
|             1|        4|              4|               0|       2|        1|
|             1|        4|              4|               0|       1|        1|
|             1|        1|              4|               0|       2|        1|
|             1|        5|              4|               0|       2|        1|
|             1|        5|              4|               0|       1|        1|
|             1|        4|              4|               0|       1|        1|
|             1|        7|              4|               0|       1|        1|
|             1|        5|              4|               0|       1|        1|
|             1|        4|              4|               0|       1|        1|
|             1|        5|              4|               0|       1|        1|
+--------------+---------+---------------+----------------+--------+---------+
only showing top 10 rows


scala> val covid_19_cases_counts = features_covid19_cases.select(features_covid19_cases.columns.map(c => sum(col(c).isNotNull.cast("int")).alias(c)): _*).show
+--------------+---------+---------------+----------------+--------+---------+
|Outbreak_Ass_#|Age_Grp_#|Source_of_Inf_#|Classification_#|Gender_#|Outcome_#|
+--------------+---------+---------------+----------------+--------+---------+
|        302658|   302658|         302658|          302658|  302658|   302658|
+--------------+---------+---------------+----------------+--------+---------+

covid_19_cases_counts: Unit = ()

scala>
```

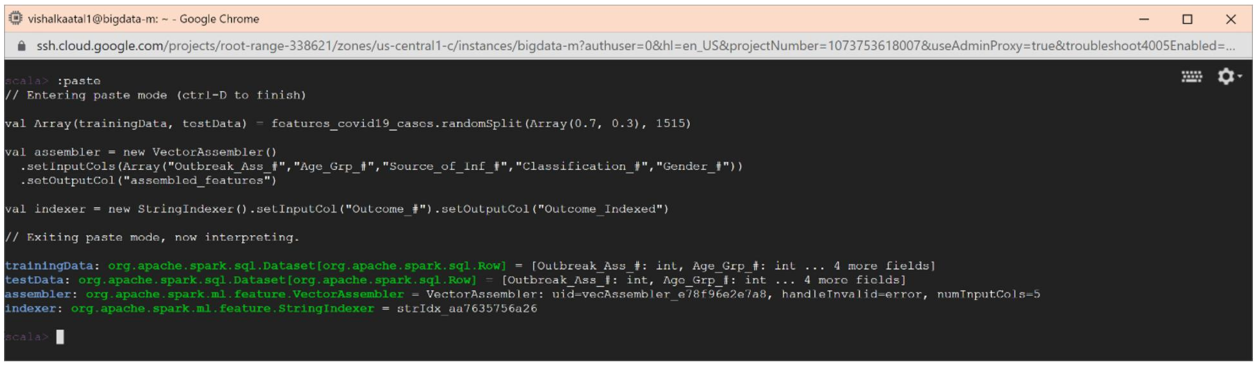
/* To run the Machine Learning Algorithm, there are specific parameters that need to be considered as Assembler */

An array will divide the dataset into training and testing data based on a 70:30 ratio. Training data will be used in the ML process, whereas the testing data will be used to predict the model's accuracy.

```
val Array(trainingData, testData) =
features_covid19_cases.randomSplit(Array(0.7, 0.3), 1515)
```

Assembler is used to select the specific columns needed for the ML analysis.

```
val assembler = new VectorAssembler()
.setInputCols(Array("Outbreak_Ass_#","Age_Grp_#","Source_of_Inf_#","Cl
assification_#","Gender_#"))
.setOutputCol("assembled_features")


val indexer = new
StringIndexer().setInputCol("Outcome_#").setOutputCol("Outcome_Indexed
")
```



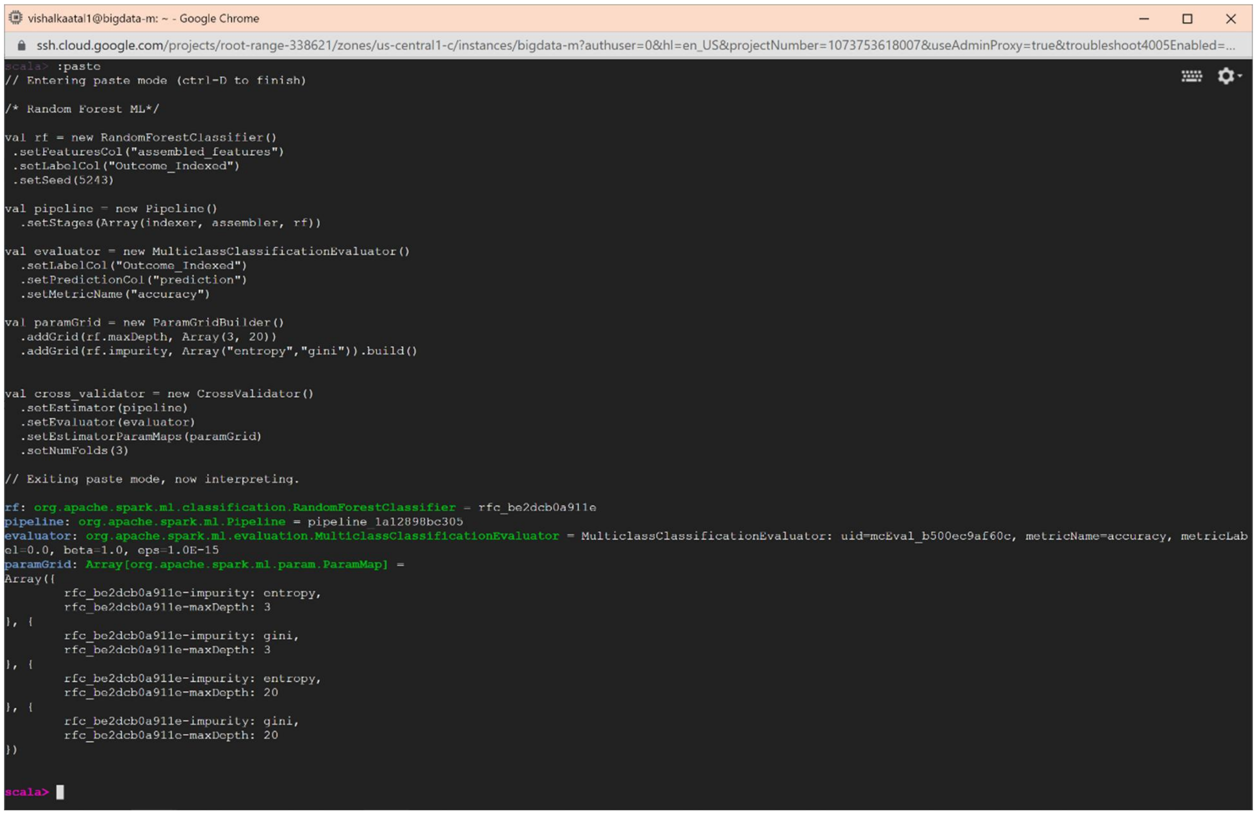
vishalkaatal1@bigdata-m: ~ - Google Chrome

ssh.cloud.google.com/projects/root-range-338621/zones/us-central1-c/instances/bigdata-m?authuser=0&hl=en_US&projectNumber=1073753618007&useAdminProxy=true&troubleshoot4005Enabled=...

```
scala> :paste
// Entering paste mode (ctrl-D to finish)

val Array(trainingData, testData) = features_covid19_cases.randomSplit(Array(0.7, 0.3), 1515)

val assembler = new VectorAssembler()
  .setInputCols(Array("Outbreak_Ass_#","Age_Grp_#","Source_of_Inf_#","Classification_#","Gender_#"))
  .setOutputCol("assembled_features")

val indexer = new StringIndexer().setInputCol("Outcome_#").setOutputCol("Outcome_Indexed")

// Exiting paste mode, now interpreting.

trainingData: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [Outbreak_Ass_#: int, Age_Grp_#: int ... 4 more fields]
testData: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [Outbreak_Ass_#: int, Age_Grp_#: int ... 4 more fields]
assembler: org.apache.spark.ml.feature.VectorAssembler = VectorAssembler: uid=vecAssembler_e78f96e2e7a8, handleInvalid=error, numInputCols=5
indexer: org.apache.spark.ml.feature.StringIndexer = strIdx_aa7635756a26

scala> 
```

/* Random Forest Machine Learning Algorithm */

It will consist of a random forest classifier, pipeline, evaluator, paraGrid, and cross-validator.

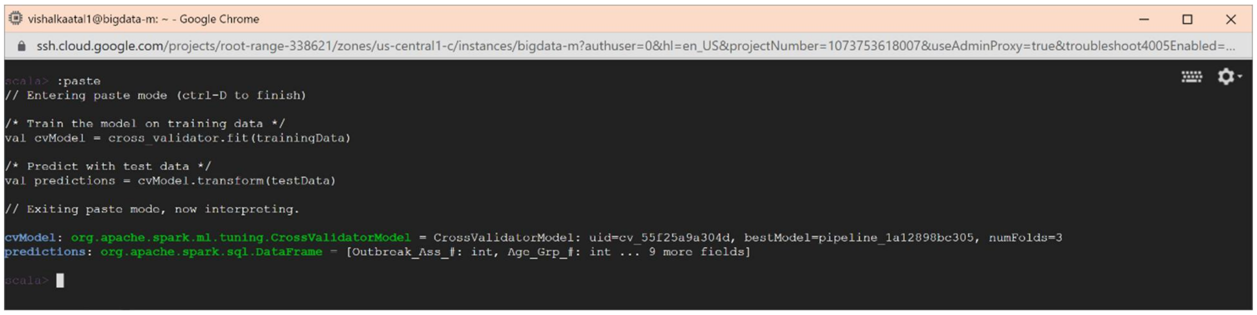


vishalkaatal1@bigdata-m: ~ - Google Chrome

ssh.cloud.google.com/projects/root-range-338621/zones/us-central1-c/instances/bigdata-m?authuser=0&hl=en_US&projectNumber=1073753618007&useAdminProxy=true&troubleshoot4005Enabled=...

```
scala> :paste
// Entering paste mode (ctrl-D to finish)

/* Random Forest ML*/

val rf = new RandomForestClassifier()
 .setFeaturesCol("assembled_features")
 .setLabelCol("Outcome_Indexed")
 .setSeed(5243)

val pipeline = new Pipeline()
  .setStages(Array(indexer, assembler, rf))

val evaluator = new MulticlassClassificationEvaluator()
  .setLabelCol("Outcome_Indexed")
  .setPredictionCol("prediction")
  .setMetricName("accuracy")

val paramGrid = new ParamGridBuilder()
  .addGrid(rf.maxDepth, Array(3, 20))
  .addGrid(rf.impurity, Array("entropy","gini")).build()


val cross_validator = new CrossValidator()
  .setEstimator(pipeline)
  .setEvaluator(evaluator)
  .setEstimatorParamMaps(paramGrid)
  .setNumFolds(3)

// Exiting paste mode, now interpreting.

rf: org.apache.spark.ml.classification.RandomForestClassifier = rfc_be2dcb0a911e
pipeline: org.apache.spark.ml.Pipeline = pipeline_1a12898bc305
evaluator: org.apache.spark.ml.evaluation.MulticlassClassificationEvaluator = MulticlassClassificationEvaluator: uid=mcEval_b500ec9af60c, metricName=accuracy, metricLab
el=0.0, beta=1.0, eps=1.0E-15
paramGrid: Array[org.apache.spark.ml.param.ParamMap] =
Array({
        rfc_be2dcb0a911e-impurity: entropy,
        rfc_be2dcb0a911e-maxDepth: 3
}, {
        rfc_be2dcb0a911e-impurity: gini,
        rfc_be2dcb0a911e-maxDepth: 3
}, {
        rfc_be2dcb0a911e-impurity: entropy,
        rfc_be2dcb0a911e-maxDepth: 20
}, {
        rfc_be2dcb0a911e-impurity: gini,
        rfc_be2dcb0a911e-maxDepth: 20
})


scala> 
```

/* Training and testing of the model using training data */



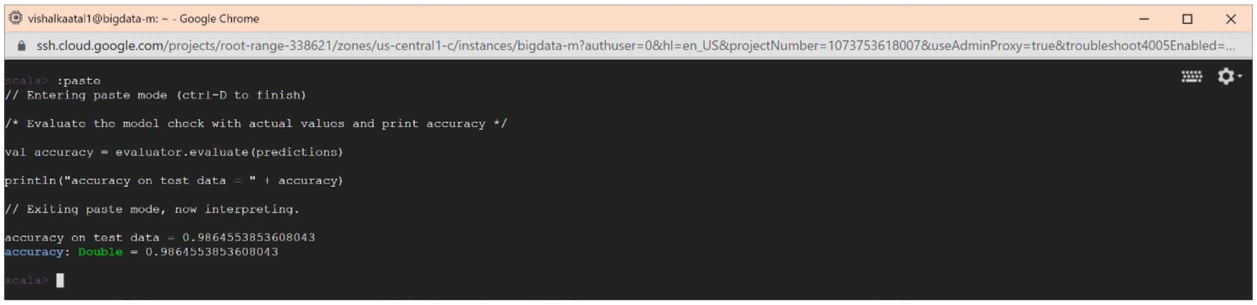
vishalkaatal1@bigdata-m: ~ - Google Chrome

ssh.cloud.google.com/projects/root-range-338621/zones/us-central1-c/instances/bigdata-m?authuser=0&hl=en_US&projectNumber=1073753618007&useAdminProxy=true&troubleshoot4005Enabled=...

```
scala> :paste
// Entering paste mode (ctrl-D to finish)

/* Train the model on training data */
val cvModel = cross_validator.fit(trainingData)

/* Predict with test data */
val predictions = cvModel.transform(testData)

// Exiting paste mode, now interpreting.

cvModel: org.apache.spark.ml.tuning.CrossValidatorModel = CrossValidatorModel: uid=cv_55f25a9a304d, bestModel=pipeline_1a12898bc305, numFolds=3
predictions: org.apache.spark.sql.DataFrame = [Outbreak_Ass_#: int, Age_Grp_#: int ... 9 more fields]

scala>
```

/* Predicting the accuracy of the model using an **Unbalanced dataset** */

Accuracy of the test data = **0.9864553853608043 or 98.65%**



vishalkaatal1@bigdata-m: ~ - Google Chrome

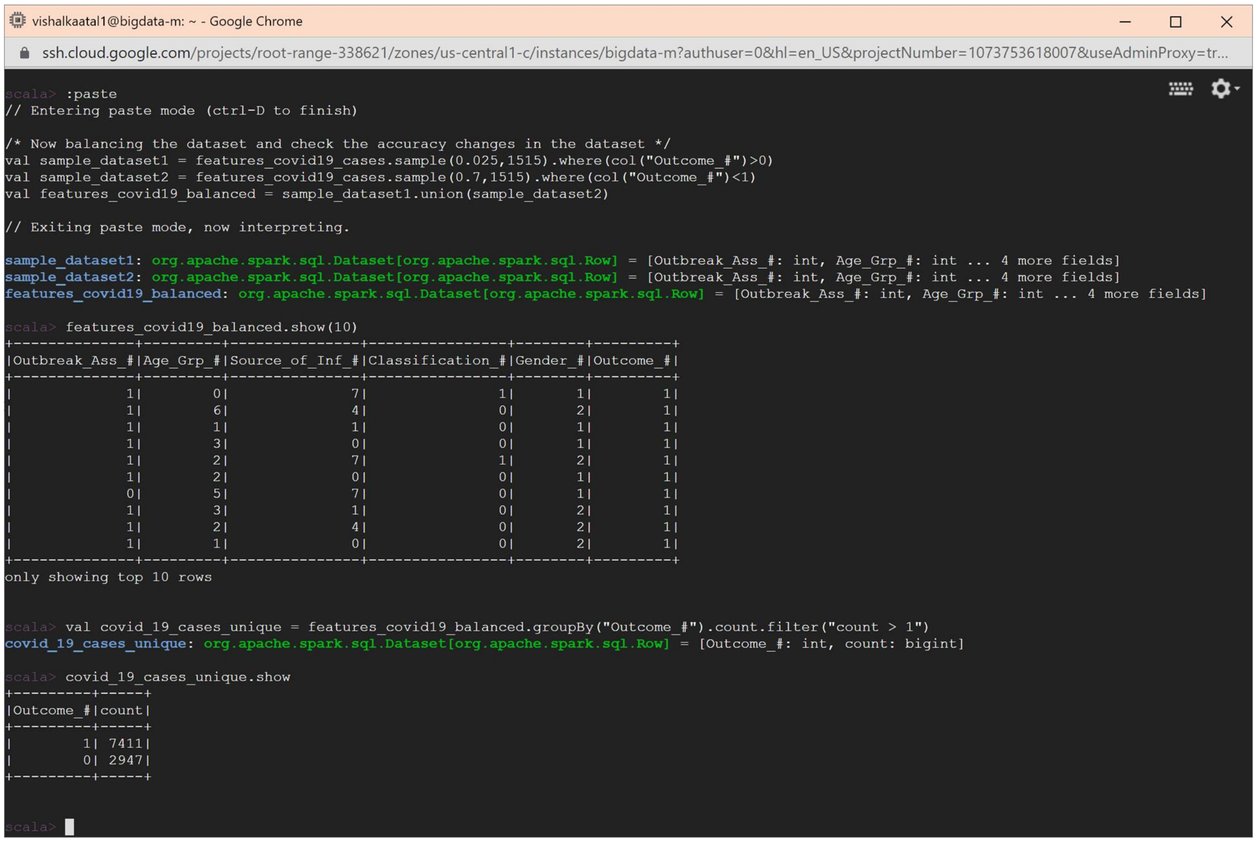
ssh.cloud.google.com/projects/root-range-338621/zones/us-central1-c/instances/bigdata-m?authuser=0&hl=en_US&projectNumber=1073753618007&useAdminProxy=true&troubleshoot4005Enabled=...

```
scala> :paste
// Entering paste mode (ctrl-D to finish)

/* Evaluate the model check with actual values and print accuracy */

val accuracy = evaluator.evaluate(predictions)

println("accuracy on test data = " + accuracy)

// Exiting paste mode, now interpreting.

accuracy on test data = 0.9864553853608043
accuracy: Double = 0.9864553853608043

scala>
```

/* Moving on to Balancing for Dataset */

```
val sample_dataset1 =
features_covid19_cases.sample(0.025,1515).where(col("Outcome_#")>0)

val sample_dataset2 =
features_covid19_cases.sample(0.7,1515).where(col("Outcome_#")<1)

val features_covid19_balanced = sample_dataset1.union(sample_dataset2)

features_covid19_balanced.show
```

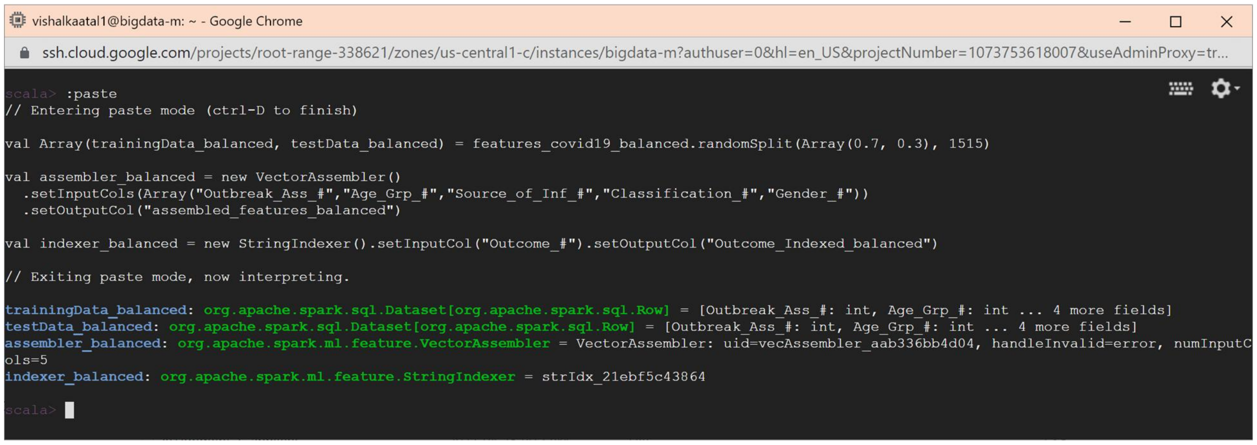
Sample dataset1 consists of Outcome as Resolved, with the maximum number of records. Therefore, the sample1 is taken with 0.025% of 298482 records. However, the sample dataset2 consists of Outcome as Fatal, the target variable. Only 0.7% of 4176 records were considered for sample2. Finally, combining these two will result in a balanced dataset with 7411 records as Resolved (71.55%) and 2947 records as Fatal (28.45%).


vishalkaatal1@bigdata-m: ~ - Google Chrome

ssh.cloud.google.com/projects/root-range-338621/zones/us-central1-c/instances/bigdata-m?authuser=0&hl=en_US&projectNumber=1073753618007&useAdminProxy=tr...

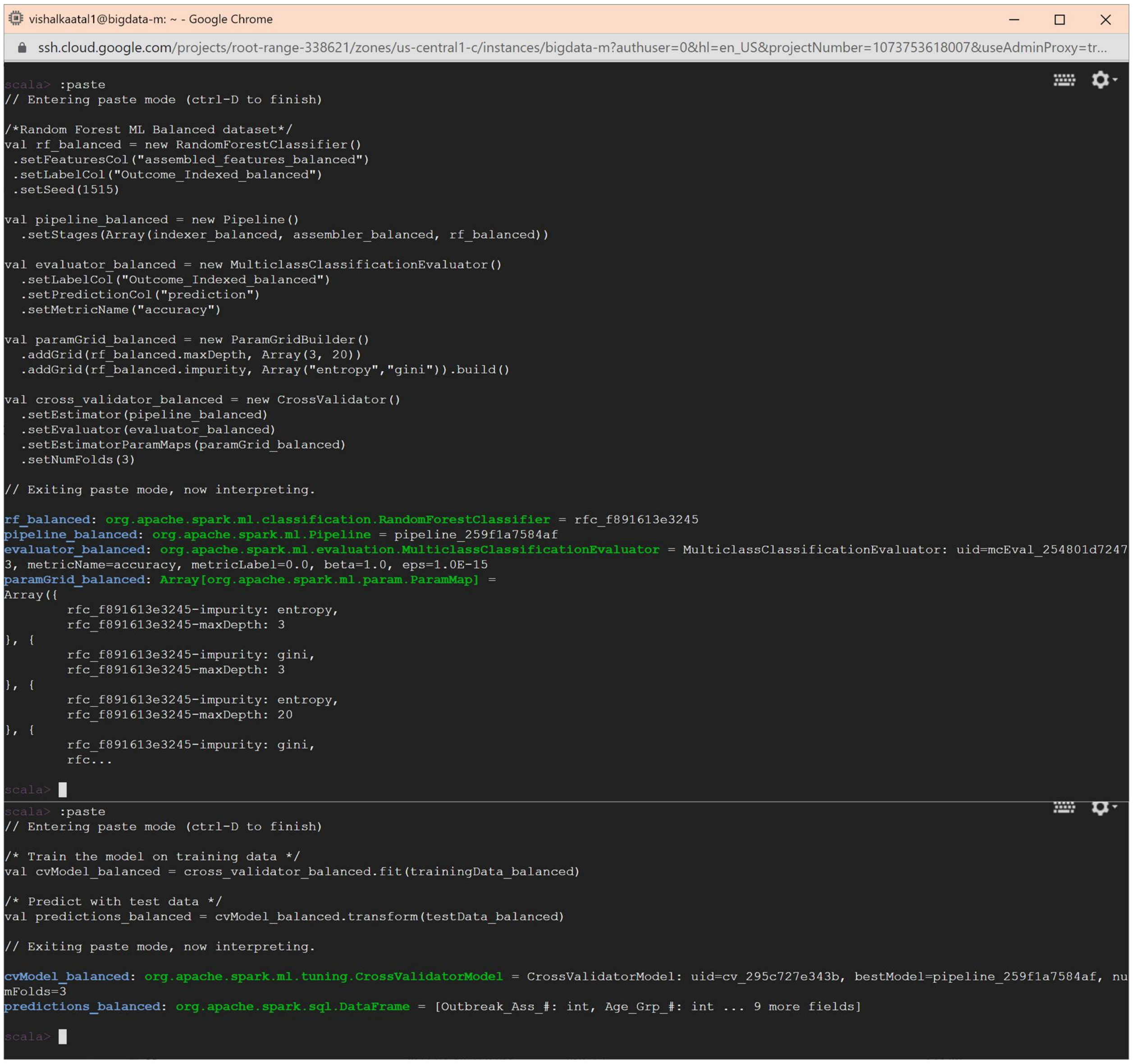
```
scala> :paste
// Entering paste mode (ctrl-D to finish)

/* Now balancing the dataset and check the accuracy changes in the dataset */
val sample_dataset1 = features_covid19_cases.sample(0.025,1515).where(col("Outcome_#")>0)
val sample_dataset2 = features_covid19_cases.sample(0.7,1515).where(col("Outcome_#")<1)
val features_covid19_balanced = sample_dataset1.union(sample_dataset2)

// Exiting paste mode, now interpreting.

sample_dataset1: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [Outbreak_Ass_#: int, Age_Grp_#: int ... 4 more fields]
sample_dataset2: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [Outbreak_Ass_#: int, Age_Grp_#: int ... 4 more fields]
features_covid19_balanced: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [Outbreak_Ass_#: int, Age_Grp_#: int ... 4 more fields]

scala> features_covid19_balanced.show(10)
+--------------+---------+---------------+----------------+--------+---------+
|Outbreak_Ass_#|Age_Grp_#|Source_of_Inf_#|Classification_#|Gender_#|Outcome_#|
+--------------+---------+---------------+----------------+--------+---------+
|             1|        0|              7|               1|       1|        1|
|             1|        6|              4|               0|       2|        1|
|             1|        1|              1|               0|       1|        1|
|             1|        3|              0|               0|       1|        1|
|             1|        2|              7|               1|       2|        1|
|             1|        2|              0|               0|       1|        1|
|             0|        5|              7|               0|       1|        1|
|             1|        3|              1|               0|       2|        1|
|             1|        2|              4|               0|       2|        1|
|             1|        1|              0|               0|       2|        1|
+--------------+---------+---------------+----------------+--------+---------+
only showing top 10 rows


scala> val covid_19_cases_unique = features_covid19_balanced.groupBy("Outcome_#").count.filter("count > 1")
covid_19_cases_unique: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [Outcome_#: int, count: bigint]

scala> covid_19_cases_unique.show
+---------+-----+
|Outcome_#|count|
+---------+-----+
|        1| 7411|
|        0| 2947|
+---------+-----+


scala> 
```

/* Now, again, the ML algorithm is applied by dividing the dataset */


vishalkaatal1@bigdata-m: ~ - Google Chrome

ssh.cloud.google.com/projects/root-range-338621/zones/us-central1-c/instances/bigdata-m?authuser=0&hl=en_US&projectNumber=1073753618007&useAdminProxy=tr...

```
scala> :paste
// Entering paste mode (ctrl-D to finish)

val Array(trainingData_balanced, testData_balanced) = features_covid19_balanced.randomSplit(Array(0.7, 0.3), 1515)

val assembler_balanced = new VectorAssembler()
  .setInputCols(Array("Outbreak_Ass_#","Age_Grp_#","Source_of_Inf_#","Classification_#","Gender_#"))
  .setOutputCol("assembled_features_balanced")

val indexer_balanced = new StringIndexer().setInputCol("Outcome_#").setOutputCol("Outcome_Indexed_balanced")

// Exiting paste mode, now interpreting.

trainingData_balanced: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [Outbreak_Ass_#: int, Age_Grp_#: int ... 4 more fields]
testData_balanced: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [Outbreak_Ass_#: int, Age_Grp_#: int ... 4 more fields]
assembler_balanced: org.apache.spark.ml.feature.VectorAssembler = VectorAssembler: uid=vecAssembler_aab336bb4d04, handleInvalid=error, numInputC
ols=5
indexer_balanced: org.apache.spark.ml.feature.StringIndexer = strIdx_21ebf5c43864

scala> 
```



vishalkaatal1@bigdata-m: ~ - Google Chrome

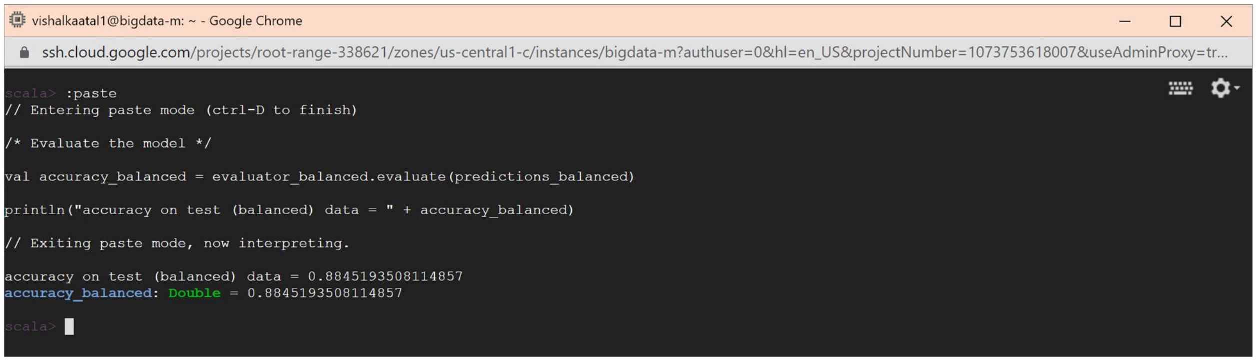
ssh.cloud.google.com/projects/root-range-338621/zones/us-central1-c/instances/bigdata-m?authuser=0&hl=en_US&projectNumber=1073753618007&useAdminProxy=tr...

```
scala> :paste
// Entering paste mode (ctrl-D to finish)

/*Random Forest ML Balanced dataset*/
val rf_balanced = new RandomForestClassifier()
 .setFeaturesCol("assembled_features_balanced")
 .setLabelCol("Outcome_Indexed_balanced")
 .setSeed(1515)

val pipeline_balanced = new Pipeline()
  .setStages(Array(indexer_balanced, assembler_balanced, rf_balanced))

val evaluator_balanced = new MulticlassClassificationEvaluator()
  .setLabelCol("Outcome_Indexed_balanced")
  .setPredictionCol("prediction")
  .setMetricName("accuracy")

val paramGrid_balanced = new ParamGridBuilder()
  .addGrid(rf_balanced.maxDepth, Array(3, 20))
  .addGrid(rf_balanced.impurity, Array("entropy","gini")).build()

val cross_validator_balanced = new CrossValidator()
  .setEstimator(pipeline_balanced)
  .setEvaluator(evaluator_balanced)
  .setEstimatorParamMaps(paramGrid_balanced)
  .setNumFolds(3)

// Exiting paste mode, now interpreting.

rf_balanced: org.apache.spark.ml.classification.RandomForestClassifier = rfc_f891613e3245
pipeline_balanced: org.apache.spark.ml.Pipeline = pipeline_259f1a7584af
evaluator_balanced: org.apache.spark.ml.evaluation.MulticlassClassificationEvaluator = MulticlassClassificationEvaluator: uid=mcEval_254801d7247
3, metricName=accuracy, metricLabel=0.0, beta=1.0, eps=1.0E-15
paramGrid_balanced: Array[org.apache.spark.ml.param.ParamMap] =
Array({
        rfc_f891613e3245-impurity: entropy,
        rfc_f891613e3245-maxDepth: 3
}, {
        rfc_f891613e3245-impurity: gini,
        rfc_f891613e3245-maxDepth: 3
}, {
        rfc_f891613e3245-impurity: entropy,
        rfc_f891613e3245-maxDepth: 20
}, {
        rfc_f891613e3245-impurity: gini,
        rfc...

scala>
```

```
scala> :paste
// Entering paste mode (ctrl-D to finish)

/* Train the model on training data */
val cvModel_balanced = cross_validator_balanced.fit(trainingData_balanced)

/* Predict with test data */
val predictions_balanced = cvModel_balanced.transform(testData_balanced)

// Exiting paste mode, now interpreting.

cvModel_balanced: org.apache.spark.ml.tuning.CrossValidatorModel = CrossValidatorModel: uid=cv_295c727e343b, bestModel=pipeline_259f1a7584af, nu
mFolds=3
predictions_balanced: org.apache.spark.sql.DataFrame = [Outbreak_Ass_#: int, Age_Grp_#: int ... 9 more fields]

scala>
```

/* Finally, the accuracy of the **Balanced dataset** is calculated*/

Accuracy on test (balanced) data **= 0.8845193508114857 or 88.45%**



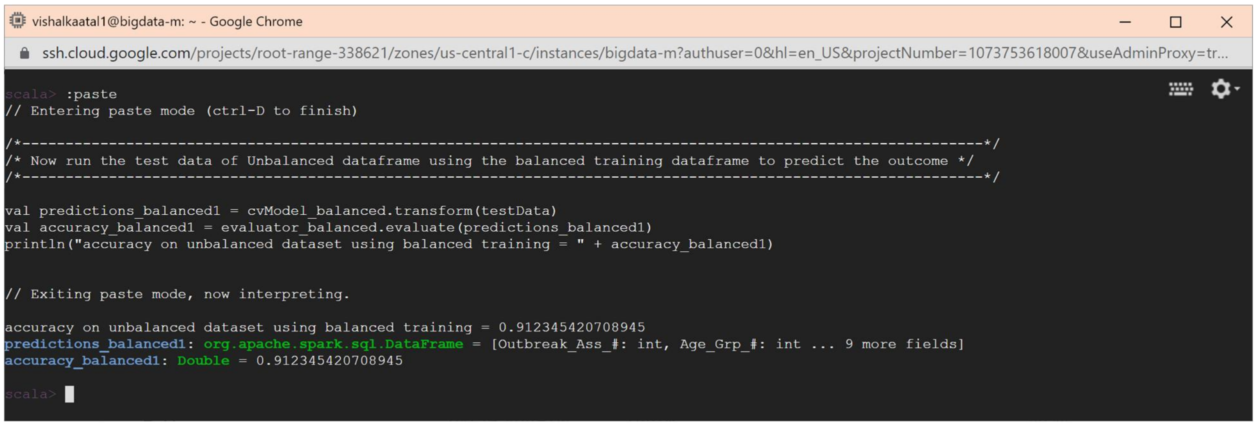
vishalkaatal1@bigdata-m: ~ - Google Chrome

ssh.cloud.google.com/projects/root-range-338621/zones/us-central1-c/instances/bigdata-m?authuser=0&hl=en_US&projectNumber=1073753618007&useAdminProxy=tr...

```
scala> :paste
// Entering paste mode (ctrl-D to finish)

/* Evaluate the model */

val accuracy_balanced = evaluator_balanced.evaluate(predictions_balanced)

println("accuracy on test (balanced) data = " + accuracy_balanced)

// Exiting paste mode, now interpreting.

accuracy on test (balanced) data = 0.8845193508114857
accuracy_balanced: Double = 0.8845193508114857

scala>
```

/* Now run the test data of the Unbalanced data frame using the balanced training data frame to predict the outcome as Fatal */

Accuracy on the unbalanced dataset using balanced training = 0.912345420708945 or 91.23%


vishalkaatal1@bigdata-m: ~ - Google Chrome

ssh.cloud.google.com/projects/root-range-338621/zones/us-central1-c/instances/bigdata-m?authuser=0&hl=en_US&projectNumber=1073753618007&useAdminProxy=tr...

```
scala> :paste
// Entering paste mode (ctrl-D to finish)

/*-----------------------------------------------------------------------------------------------------------------*/
/* Now run the test data of Unbalanced dataframe using the balanced training dataframe to predict the outcome */
/*-----------------------------------------------------------------------------------------------------------------*/

val predictions_balanced1 = cvModel_balanced.transform(testData)
val accuracy_balanced1 = evaluator_balanced.evaluate(predictions_balanced1)
println("accuracy on unbalanced dataset using balanced training = " + accuracy_balanced1)


// Exiting paste mode, now interpreting.

accuracy on unbalanced dataset using balanced training = 0.912345420708945
predictions_balanced1: org.apache.spark.sql.DataFrame = [Outbreak_Ass_#: int, Age_Grp_#: int ... 9 more fields]
accuracy_balanced1: Double = 0.912345420708945

scala>
```

It proves that the selected columns are the best fit in the developed random forest machine-learning algorithm to predict the outcome as fatal with approximately 91.23%.