# Covid Hospitalization Analysis

A brief report on Covid related ICU cases

Hariharan Sasidharan
BDSA 8773986

# Content:

## Abstract:

Covid19 has rapidly changed the way of human life in just the past two years. It has negatively affected countless lives and continue being a huge threat till date. It is high time to start studying the effects by looking at it from a data enthusiast's point of view. This report is one such effort where the data from Case and Contact Management System (CCM), Toronto, Ontario is used to understand the hospitalization patterns.

## Introduction:

This report aims at creating a predictive model that can predict the number of patients who are going to be admitted in the ICU while affected with covid. The predictive model will be developed with the help of Apache Spark. The algorithm used for this predictive model will be the Random Tree Classifier algorithm.

## Understanding the data:

The dataset contains information related to all the confirmed or probable cases that was reported to the CCM.

The dataset contains the following information in the form of columns,

- **_id** – Unique row identifier for Open Data database
- **Assigned_ID-** A unique ID assigned to cases by Toronto Public Health for the purposes of posting to Open Data, to allow for tracking of specific cases.
- **Outbreak Associated-** Outbreak associated cases are associated with outbreaks of COVID-19 in Toronto healthcare institutions and healthcare settings (e.g. long-term care homes, retirement homes, hospitals, etc.) and other Toronto congregate settings (such as homeless shelters).
- **Age Group-** Age at time of illness. Age groups (in years): ≤19, 20-29, 30-39, 40-49, 50-59, 60-69, 70-79, 80-89, 90+, unknown (blank)

- **Neighbourhood Name-** Name of the neighborhood in Toronto.
- **FSA-** Forward sortation area (i.e. first three characters of postal code).
- **Source of Infection-** The infection vector description.
- **Classification-** Classifying between confirmed and probable cases.
- **Episode Date-** Estimated date of infection
- **Reported Date-** The date on which the case was reported to Toronto Public Health.
- **Client Gender-** Self-reported gender.
- **Outcome-** Cases reported as Fatal, Resolved and Active.
- **Currently Hospitalized-** Cases that are Currently Hospitalized.
- **Currently in ICU-** Cases that are currently in ICU.
- **Currently Intubated-** Cases that are currently intubated.
- **Ever Hospitalized-** Cases that were ever Hospitalized.
- **Ever in ICU-** Cases that were ever in ICU.
- **Ever Intubated-** Cases that were ever intubated.

# Objective:

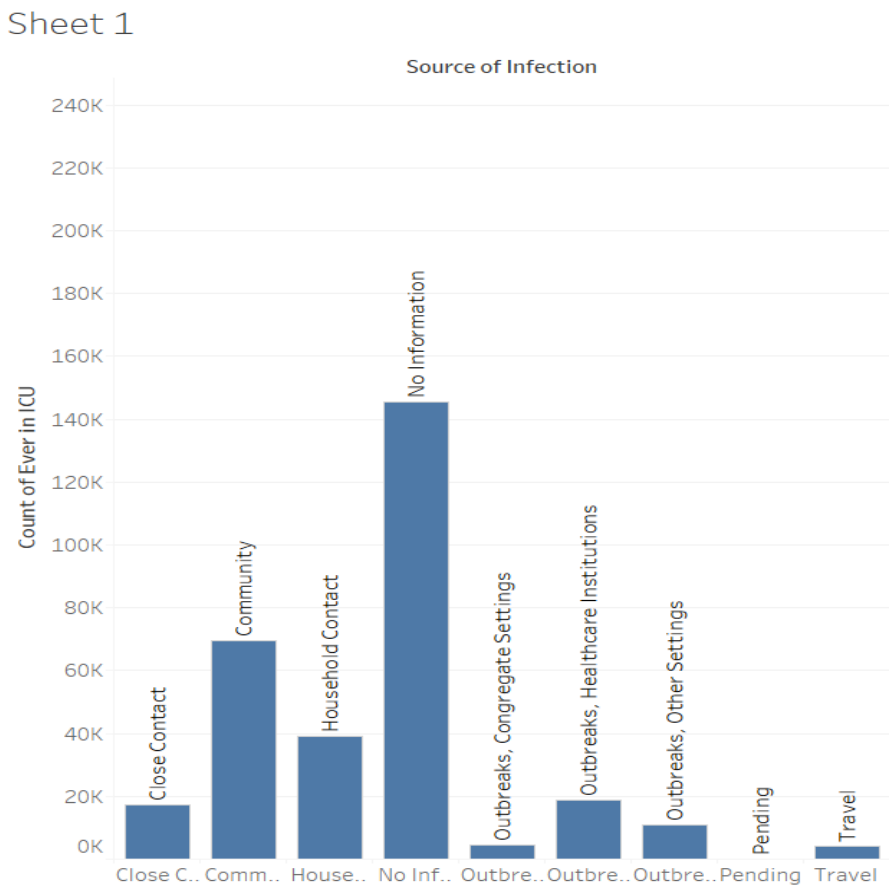
The objective of this report is to find factors contributing to ICU admission rates in patients affected with Covid19. The end goal is to create a predictive model that can predict the ICU admission rates.

# Analysis:

A slew of preliminary analysis is done in order to select the appropriate the feature selection.

## Source of infection and ICU admission:

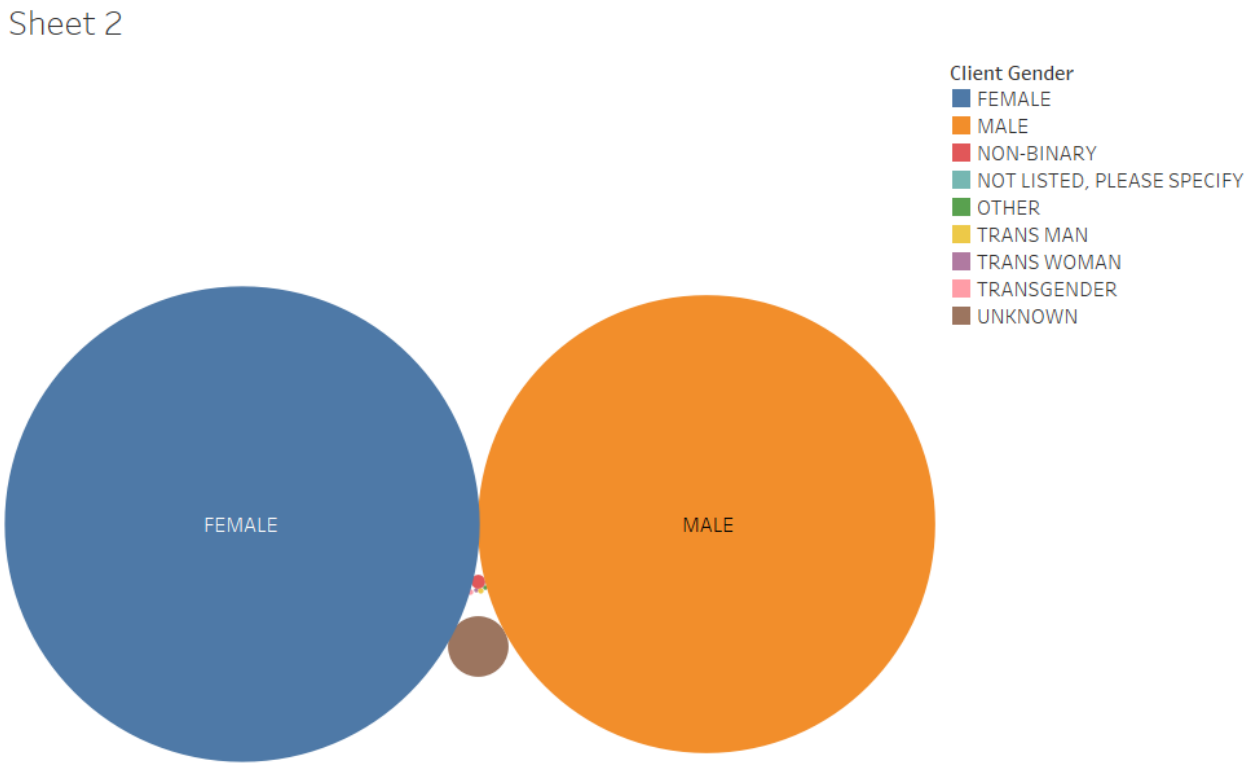
Source of infection can be a good estimator about how the patients react and end up in ICU. The below chart shows how many patients ended up in ICU with different sources of infection.




Count of Ever in ICU for each Source of Infection. The marks are labeled by Source of Infection.

## Gender and ICU rates:

Another predictor is gender of the patient and how it affects the ICU admission rate.

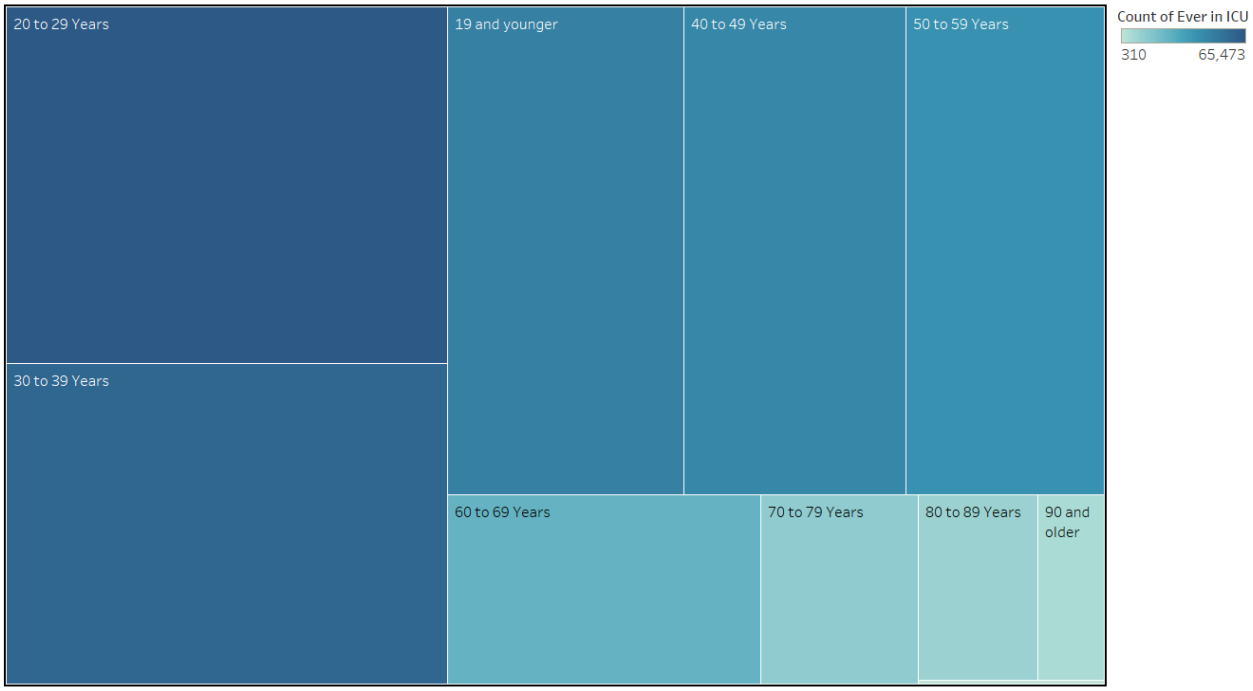



Client Gender. Color shows details about Client Gender. Size shows count of Ever in ICU. The marks are labeled by Client Gender.

## Age group and ICU admission rate:

The age group of the patient is another valuable predictor for the admission rates. An interesting fact is that most people who ended up in ICU were younger people in Toronto.

Sheet 3




Age Group. Color shows count of Ever in ICU. Size shows count of Ever in ICU. The marks are labeled by Age Group.

Therefore, the report will select the age group, gender, source of infection and Outbreak associated as the features for predicting if the patient will end up in ICU.

## Conclusion:

The predictive model built with Random Forest Classifier for the covid dataset was able to produce accuracy of 99%.

# Appendix:

## Loading the data and selecting features:

First the data source is uploaded to GCP HDFS cluster and then loaded into the spark shell.

```
import org.apache.spark.sql.functions._
import org.apache.spark.sql.expressions.Window
import org.apache.spark.ml.feature.{VectorAssembler, StringIndexer}
import org.apache.spark.ml.Pipeline
import org.apache.spark.ml.classification.{RandomForestClassificationModel, RandomForestClassifier}
import org.apache.spark.ml.tuning.{CrossValidator, CrossValidatorModel, ParamGridBuilder}
import org.apache.spark.ml.evaluation.{MulticlassClassificationEvaluator}
import org.apache.spark.ml.param.ParamMap
import org.apache.spark.sql.types.{IntegerType, DoubleType}

val df=spark.read
 .format("csv")
 .option("header","true")
 .load("hdfs://10.128.0.16:8020/BigData/covid.csv")
val cleaned_DF = df.na.drop()

val dataset = cleaned_DF.select(col("Ever in ICU"),
 col("Age Group"),
 col("Source of Infection"),
 col("Outbreak Associated"),
 col("Client Gender"))


dataset.show(10)
```

### Output:

```
+-----------+--------------+-------------------+-------------------+-------------+
|Ever in ICU|     Age Group|Source of Infection|Outbreak Associated|Client Gender|
+-----------+--------------+-------------------+-------------------+-------------+
|         No|50 to 59 Years|             Travel|           Sporadic|       FEMALE|
|         No|50 to 59 Years|             Travel|           Sporadic|         MALE|
|         No|20 to 29 Years|             Travel|           Sporadic|       FEMALE|
|         No|60 to 69 Years|             Travel|           Sporadic|       FEMALE|
|         No|60 to 69 Years|             Travel|           Sporadic|         MALE|
|         No|50 to 59 Years|             Travel|           Sporadic|         MALE|
|         No|80 to 89 Years|             Travel|           Sporadic|         MALE|
|         No|60 to 69 Years|             Travel|           Sporadic|         MALE|
|         No|50 to 59 Years|             Travel|           Sporadic|         MALE|
|         No|60 to 69 Years|             Travel|           Sporadic|         MALE|
+-----------+--------------+-------------------+-------------------+-------------+
only showing top 10 rows
```

## Indexing:

As our dataset contains mostly text values it is better to index them and assign integers for distinct string values. This is done so that the ML algorithm can better understand the dataset and come up with a better model.

```
val inputColumns = Array("Age Group","Source of Infection", "Outbreak Associated","Client Gender")
val outputColumns = Array("Age_index","Source_index","Outbreak_index","Gender_index")

val indexer = new StringIndexer()
indexer.setInputCols(inputColumns)
indexer.setOutputCols(outputColumns)

val stringIndexer = new StringIndexer()
 .setInputCol("Ever in ICU")
 .setOutputCol("ICU_index")

val DF_indexed = indexer.fit(dataset).transform(dataset)
val DF_indexed2 = stringIndexer.fit(DF_indexed).transform(DF_indexed)

val rankDf = DF_indexed2.select(col("ICU_index").cast(IntegerType),
col("Age_index").cast(IntegerType),
col("Source_index").cast(IntegerType),
col("Outbreak_index").cast(IntegerType),
col("Gender_index").cast(IntegerType))
rankDf.show(10)
```

### Output:

```
+---------+---------+------------+--------------+------------+
|ICU_index|Age_index|Source_index|Outbreak_index|Gender_index|
+---------+---------+------------+--------------+------------+
|        0|        4|           7|             0|           0|
|        0|        4|           7|             0|           1|
|        0|        0|           7|             0|           0|
|        0|        5|           7|             0|           0|
|        0|        5|           7|             0|           1|
|        0|        4|           7|             0|           1|
|        0|        7|           7|             0|           1|
|        0|        5|           7|             0|           1|
|        0|        4|           7|             0|           1|
|        0|        5|           7|             0|           1|
+---------+---------+------------+--------------+------------+
only showing top 10 rows
```

## Splitting the data and building model:

The next step is to split the data set into training and test dataset. This model is built on an 80/20 split. We select Random Forest classifier as our ML algorithm and then feed the data set to a newly assigned pipeline.

```
val Array(trainingData, testData) = rankDf.randomSplit(Array(0.8, 0.2), 750)

val assembler = new VectorAssembler()
 .setInputCols(Array("Age_vector","Source_vector","Outbreak_vector","Gender_vector","Age_index","Source_index","Outbreak_index","Gender_index"))
 .setOutputCol("assembled-features")

val rf = new RandomForestClassifier()
 .setFeaturesCol("assembled-features")
 .setLabelCol("ICU_index")
 .setSeed(1234)

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

## Evaluation:

Finally, we run our dataset into the built up model to test its accuracy level.

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

val paramGrid = new ParamGridBuilder()
 .addGrid(rf.maxDepth,Array(3,4))
 .addGrid(rf.impurity, Array("entropy","gini")).build()

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

val cvModel = cross_validator.fit(trainingData)

val predictions = cvModel.transform(testData)

val accuracy = evaluator.evaluate(predictions)
println("Accuracy of the model = "+accuracy)
```

### Output:

```
// Exiting paste mode, now interpreting.

Accuracy of the model = 0.9912615056841825
```

## Saving the model:

The model has a 99% accuracy so now we save it in the HDFS Cluster.

```
predictions
 .select(col("ICU_index"),
col("Age_index"),
col("Source_index"),
col("Outbreak_index"),
col("Gender_index"))
 .write
 .format("csv")
 .save("hdfs://10.128.0.16:8020/BigData/covid/output/")

// Exiting paste mode, now interpreting.
```

## Output:

```
harisasi258@cluster-c3a8-m:~$ hadoop fs -ls /BigData/covid/output/
Found 3 items
-rw-r--r--   1 harisasi258 hadoop          0 2022-04-22 03:17 /BigData/covid/output/_SUCCESS
-rw-r--r--   1 harisasi258 hadoop     332490 2022-04-22 03:17 /BigData/covid/output/part-00000-4b732e07-b42a-480a-b
419-8b95cb5a1e7c-c000.csv
-rw-r--r--   1 harisasi258 hadoop     268300 2022-04-22 03:17 /BigData/covid/output/part-00001-4b732e07-b42a-480a-b
419-8b95cb5a1e7c-c000.csv
```