In [1]:
```
from pyspark.sql import SparkSession
from pyspark.sql.functions import when, col
from pyspark.ml import PipelineModel
from pyspark.ml.tuning import CrossValidatorModel

# Initialize Spark Session
spark = SparkSession.builder \
    .appName("COVID-19 Model Testing") \
    .config("spark.sql.shuffle.partitions", "100") \
    .config("spark.driver.memory", "8g") \
    .config("spark.executor.memory", "8g") \
    .config("spark.driver.maxResultSize", "2g") \
    .getOrCreate()
```

```
VBox()
Starting Spark application
```

ID	Kind	State	Spark UI	Driver log	User	Current session?
8	pyspark	idle	Link		None	✔

```
FloatProgress(value=0.0, bar_style='info', description='Progress:', layout=Layout(height='25px', w
idth='50%'),…
SparkSession available as 'spark'.
FloatProgress(value=0.0, bar_style='info', description='Progress:', layout=Layout(height='25px', w
idth='50%'),…
```

In [2]:
```
# Load the saved model from S3
model_path = "s3://covid-data-project-final/model/"
loaded_model = CrossValidatorModel.load(model_path)

# Load new data from S3
new_data_path = "s3://covid-data-project-final/new_test_data.csv"
new_data = spark.read.csv(new_data_path, header=True, inferSchema=True)

# Convert CLASIFFICATION_FINAL to binary
new_data = new_data.withColumn(
    "CLASIFFICATION_FINAL",
    when(col("CLASIFFICATION_FINAL") <= 3, 0).otherwise(1)
)

# Handle missing values as per previous processing
new_data_cleaned = new_data.na.drop()

# Create AGE_GROUP feature (as done before)
new_data_cleaned = new_data_cleaned.withColumn(
    'AGE_GROUP',
    when(col('AGE') < 20, 'Under 20')
    .when((col('AGE') >= 20) & (col('AGE') < 40), '20-39')
    .when((col('AGE') >= 40) & (col('AGE') < 60), '40-59')
    .when(col('AGE') >= 60, '60 and above')
)
```

```
VBox()
FloatProgress(value=0.0, bar_style='info', description='Progress:', layout=Layout(height='25px', w
idth='50%'),…
```

In [4]:
```
# Transform the new data with the saved model's pipeline and predict
predictions = loaded_model.transform(new_data_cleaned)
```

```
# Show predictions
predictions.select("features", "prediction").show(20)
```

```
VBox()
FloatProgress(value=0.0, bar_style='info', description='Progress:', layout=Layout(height='25px', w
idth='50%'),…
+--------------------+----------+
|            features|prediction|
+--------------------+----------+
|(16,[0,3,7,10,11,...|       1.0|
|(16,[1,2,3,9,12,1...|       0.0|
|(16,[0,4,7,12,14,...|       1.0|
|(16,[2,9,11,13,14...|       0.0|
|(16,[1,4,5,7,15],...|       0.0|
|(16,[0,1,2,7,8,13...|       0.0|
|(16,[4,5,7,9,11,1...|       1.0|
|(16,[0,1,5,7,12,1...|       0.0|
|(16,[0,2,3,4,10,1...|       0.0|
|(16,[1,2,5,14,15]...|       0.0|
|(16,[0,4,6,7,9,12...|       1.0|
|(16,[7,8,10,13,14...|       1.0|
|(16,[0,1,2,6,7,12...|       0.0|
|(16,[4,5,7,9,11,1...|       1.0|
|(16,[0,1,2,3,9,11...|       0.0|
|(16,[0,3,4,6,11,1...|       1.0|
|(16,[1,2,7,9,13,1...|       0.0|
|(16,[0,2,10,12,14...|       0.0|
|(16,[1,3,4,7,9,14...|       0.0|
|(16,[2,5,6,8,10,1...|       0.0|
+--------------------+----------+
```