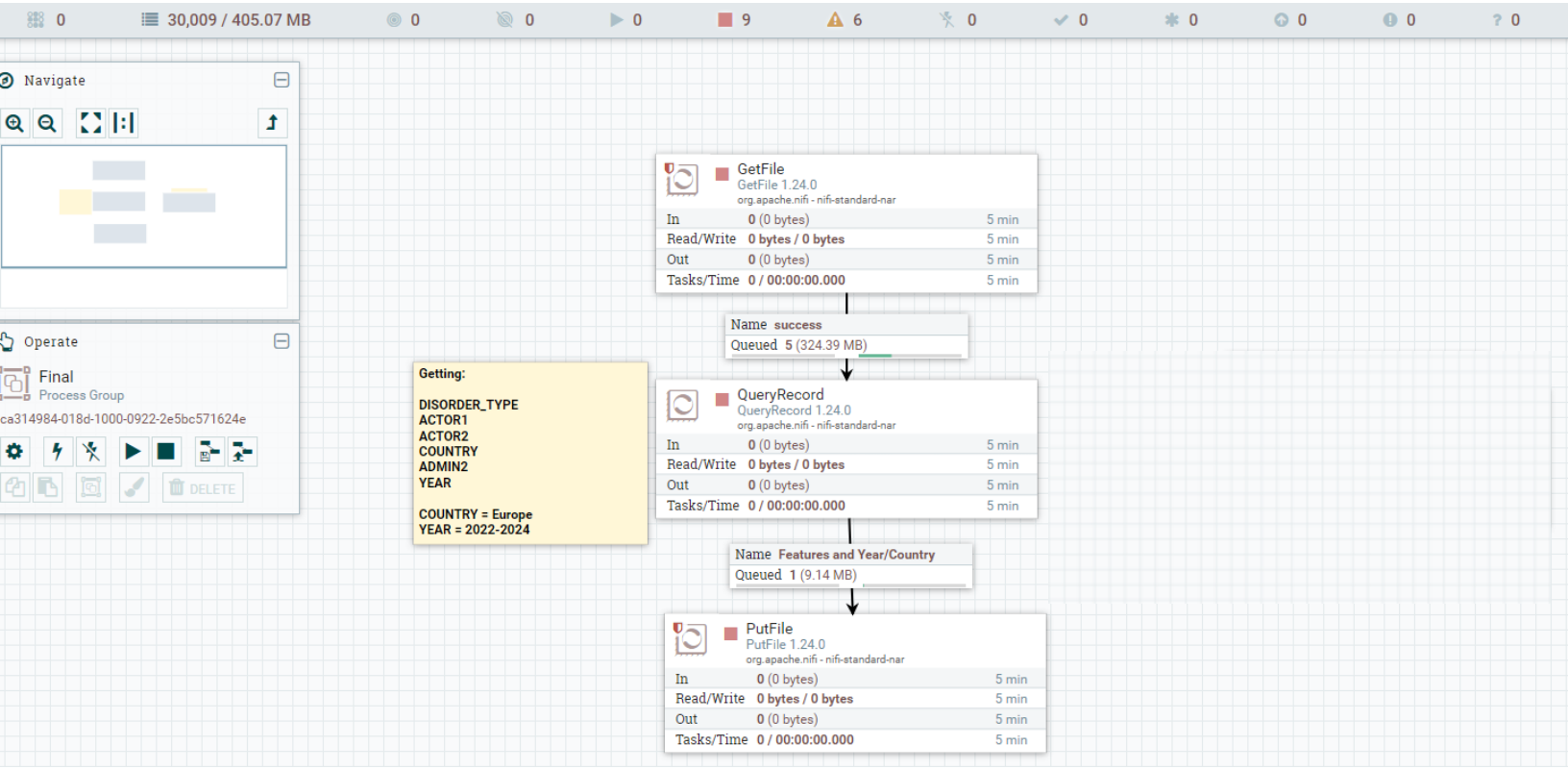
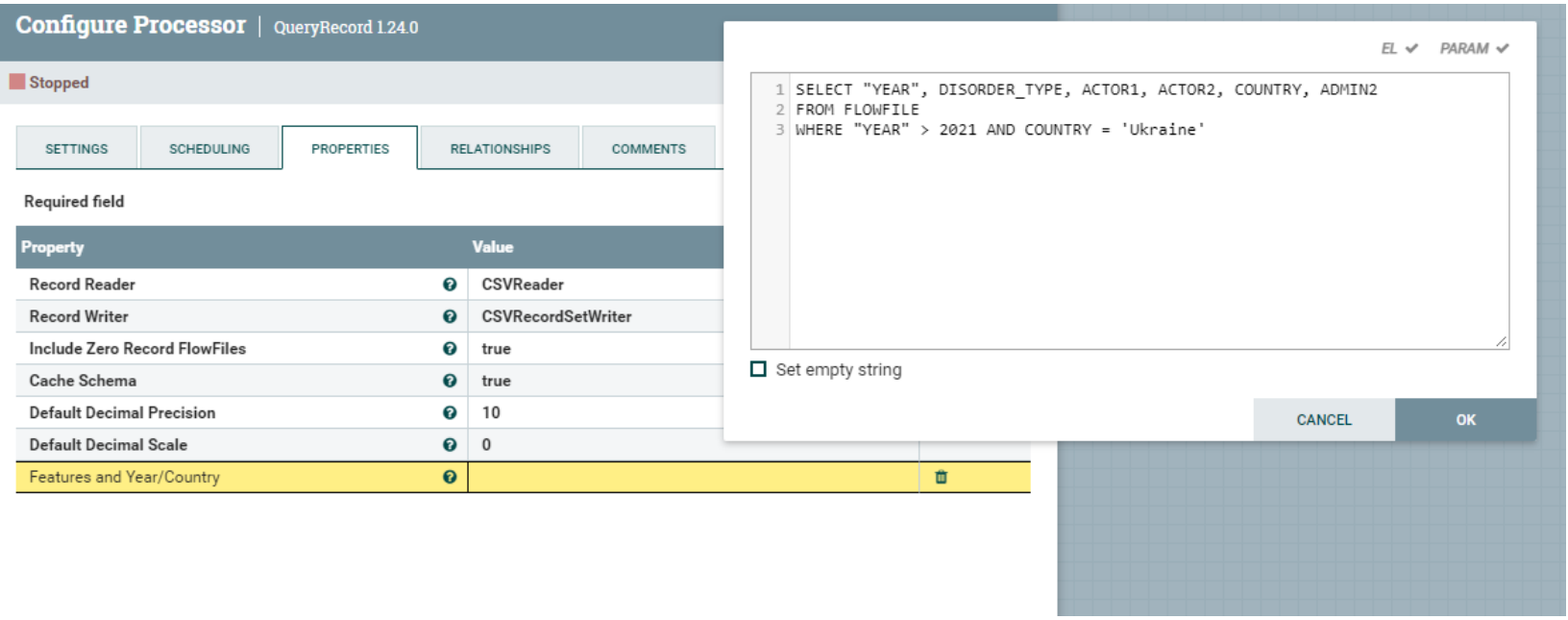
NiFi

Complete Flow



QueryRecord



HDFS

Moving file from local storage to HDFS

```
bash-5.0#  hdfs dfs -put /data/Ukraine_War.csv /data/ukraine_war.csv
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/program/hadoop/share/hadoop/common/lib/slf4j
SLF4J: Found binding in [jar:file:/usr/program/tez/lib/slf4j-log4j12-1.7.10.jar!/or
SLF4J: Found binding in [jar:file:/usr/program/hive/lib/log4j-slf4j-impl-2.10.0.jar
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
2024-03-01 02:44:24,950 WARN util.NativeCodeLoader: Unable to load native-hadoop li
bash-5.0# hdfs dfs -ls /data
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/program/hadoop/share/hadoop/common/lib/slf4j
SLF4J: Found binding in [jar:file:/usr/program/tez/lib/slf4j-log4j12-1.7.10.jar!/or
SLF4J: Found binding in [jar:file:/usr/program/hive/lib/log4j-slf4j-impl-2.10.0.jar
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
2024-03-01 02:44:38,973 WARN util.NativeCodeLoader: Unable to load native-hadoop li
Found 1 items
-rw-r--r--   1 root supergroup    9587962 2024-03-01 02:44 /data/ukraine_war.csv
```

PySpark

Loading of Data, Filling Null Fields, and Selecting Columns

```
>>> ukr_df = spark.read.format('csv').option('header', 'true').load('/ukraine_war')
>>> ukr_df.show(5)
+----+-----------------+-------------------+-------------------+-------+----------------+
|YEAR|     DISORDER_TYPE|             ACTOR1|             ACTOR2|COUNTRY|           ADMIN2|
+----+-----------------+-------------------+-------------------+-------+----------------+
|2024|Political violence|Military Forces o...|Military Forces o...|Ukraine|      Bakhmutskyi|
|2024|Political violence|Military Forces o...|Military Forces o...|Ukraine|       Pokrovskyi|
|2024|Political violence|Military Forces o...|Military Forces o...|Ukraine|       Pokrovskyi|
|2024|Political violence|Military Forces o...|Military Forces o...|Ukraine|Sievierodonetskyi|
|2024|Political violence|Military Forces o...|               null|Ukraine|          Sumskyi|
+----+-----------------+-------------------+-------------------+-------+----------------+
only showing top 5 rows

>>> no_null_df = ukr_df.fillna(value='None')
>>> no_null_df.show(5)
+----+-----------------+-------------------+-------------------+-------+----------------+
|YEAR|     DISORDER_TYPE|             ACTOR1|             ACTOR2|COUNTRY|           ADMIN2|
+----+-----------------+-------------------+-------------------+-------+----------------+
|2024|Political violence|Military Forces o...|Military Forces o...|Ukraine|      Bakhmutskyi|
|2024|Political violence|Military Forces o...|Military Forces o...|Ukraine|       Pokrovskyi|
|2024|Political violence|Military Forces o...|Military Forces o...|Ukraine|       Pokrovskyi|
|2024|Political violence|Military Forces o...|Military Forces o...|Ukraine|Sievierodonetskyi|
|2024|Political violence|Military Forces o...|               None|Ukraine|          Sumskyi|
+----+-----------------+-------------------+-------------------+-------+----------------+
only showing top 5 rows

>>> columns_df = no_null_df.select('DISORDER_TYPE', 'ACTOR1', 'ACTOR2', 'ADMIN2')
>>> columns_df.show(5)
+-----------------+-------------------+-------------------+-----------------+
|     DISORDER_TYPE|             ACTOR1|             ACTOR2|           ADMIN2|
+-----------------+-------------------+-------------------+-----------------+
|Political violence|Military Forces o...|Military Forces o...|      Bakhmutskyi|
|Political violence|Military Forces o...|Military Forces o...|       Pokrovskyi|
|Political violence|Military Forces o...|Military Forces o...|       Pokrovskyi|
|Political violence|Military Forces o...|Military Forces o...|Sievierodonetskyi|
|Political violence|Military Forces o...|               None|          Sumskyi|
+-----------------+-------------------+-------------------+-----------------+
```

PySpark Cont.

Updating Column Names

```
>>> updated_col_names = ['Event_Type','Involved_Party_A','Involved_Party_B','Region']
>>> renamed_df = columns_df.toDF(*updated_col_names)
>>> renamed_df.show(5)
+------------------+--------------------+--------------------+-----------------+
|        Event_Type|    Involved_Party_A|    Involved_Party_B|           Region|
+------------------+--------------------+--------------------+-----------------+
|Political violence|Military Forces o...|Military Forces o...|      Bakhmutskyi|
|Political violence|Military Forces o...|Military Forces o...|       Pokrovskyi|
|Political violence|Military Forces o...|Military Forces o...|       Pokrovskyi|
|Political violence|Military Forces o...|Military Forces o...|Sievierodonetskyi|
|Political violence|Military Forces o...|                None|          Sumskyi|
+------------------+--------------------+--------------------+-----------------+
```

Indexing and Vectorizing Columns (“Independent_Features” is equivalent to “features”, update not shown.)

```
>>> from pyspark.ml.feature import StringIndexer
>>> indexer=StringIndexer(inputCols=['Event_Type','Involved_Party_A','Involved_Party_B','Region'], outputCols=['Indexed_Type','Indexed_Party_A','Indexed_Party_B','Indexed_Region'])
>>> indexed_df = indexer.fit(renamed_df).transform(renamed_df)
>>> indexed_df.show(5)
+------------------+--------------------+--------------------+-----------------+------------+--------------+--------------+--------------+
|        Event_Type|    Involved_Party_A|    Involved_Party_B|           Region|Indexed_Type|Indexed_Party_A|Indexed_Party_B|Indexed_Region|
+------------------+--------------------+--------------------+-----------------+------------+--------------+--------------+--------------+
|Political violence|Military Forces o...|Military Forces o...|      Bakhmutskyi|         0.0|           0.0|           1.0|           0.0|
|Political violence|Military Forces o...|Military Forces o...|       Pokrovskyi|         0.0|           0.0|           1.0|           1.0|
|Political violence|Military Forces o...|Military Forces o...|       Pokrovskyi|         0.0|           0.0|           1.0|           1.0|
|Political violence|Military Forces o...|Military Forces o...|Sievierodonetskyi|         0.0|           0.0|           1.0|           9.0|
|Political violence|Military Forces o...|                None|          Sumskyi|         0.0|           2.0|           0.0|          12.0|
+------------------+--------------------+--------------------+-----------------+------------+--------------+--------------+--------------+
only showing top 5 rows

>>> from pyspark.ml.feature import VectorAssembler
>>> VectorAssembler(inputCols=['Indexed_Party_A','Indexed_Party_B','Indexed_Region'], outputCol='Independent_Features')
VectorAssembler_64b27d0c5ee3
>>> vec_assem = VectorAssembler(inputCols=['Indexed_Party_A','Indexed_Party_B','Indexed_Region'], outputCol='Independent_Features')
>>> output = vec_assem.transform(indexed_df)
>>> output.show()
+------------------+--------------------+--------------------+-----------------+------------+--------------+--------------+--------------+--------------------+
|        Event_Type|    Involved_Party_A|    Involved_Party_B|           Region|Indexed_Type|Indexed_Party_A|Indexed_Party_B|Indexed_Region|Independent_Features|
+------------------+--------------------+--------------------+-----------------+------------+--------------+--------------+--------------+--------------------+
|Political violence|Military Forces o...|Military Forces o...|      Bakhmutskyi|         0.0|           0.0|           1.0|           0.0|       [0.0,1.0,0.0]|
|Political violence|Military Forces o...|Military Forces o...|       Pokrovskyi|         0.0|           0.0|           1.0|           1.0|       [0.0,1.0,1.0]|
|Political violence|Military Forces o...|Military Forces o...|       Pokrovskyi|         0.0|           0.0|           1.0|           1.0|       [0.0,1.0,1.0]|
|Political violence|Military Forces o...|Military Forces o...|Sievierodonetskyi|         0.0|           0.0|           1.0|           9.0|       [0.0,1.0,9.0]|
|Political violence|Military Forces o...|                None|          Sumskyi|         0.0|           2.0|           0.0|          12.0|      [2.0,0.0,12.0]|
|Political violence|Military Forces o...|                None|          Sumskyi|         0.0|           0.0|           0.0|          12.0|      [0.0,0.0,12.0]|
|Political violence|Military Forces o...|                None|Novhorod-Siverskyi|         0.0|           2.0|           0.0|          19.0|      [2.0,0.0,19.0]|
|Political violence|Military Forces o...|Military Forces o...|      Bakhmutskyi|         0.0|           0.0|           1.0|           0.0|       [0.0,1.0,0.0]|
|Political violence|Military Forces o...|                None|      Konotopskyi|         0.0|           0.0|           0.0|          34.0|      [0.0,0.0,34.0]|
|Political violence|Military Forces o...|                None|      Polohivskyi|         0.0|           2.0|           0.0|           2.0|       [2.0,0.0,2.0]|
|Political violence|Military Forces o...|                None|      Kharkivskyi|         0.0|           0.0|           0.0|          10.0|      [0.0,0.0,10.0]|
```

PySpark Cont.

Selecting Features and Splitting the Data

```
>>> ready_df = output.select('Indexed_Type', 'Independent_Features')
>>> ready_df.show(5)
+------------+--------------------+
|Indexed_Type|Independent_Features|
+------------+--------------------+
|         0.0|       [0.0,1.0,0.0]|
|         0.0|       [0.0,1.0,1.0]|
|         0.0|       [0.0,1.0,1.0]|
|         0.0|       [0.0,1.0,9.0]|
|         0.0|      [2.0,0.0,12.0]|
+------------+--------------------+
only showing top 5 rows

>>> train_df, test_df = ready_df.randomSplit([0.75, 0.25])
>>> print(train_df.count())
73513
>>> print(test_df.count())
24536
```

Importing the Decision Tree Classifier and Evaluator

```
>>> from pyspark.ml.classification import DecisionTreeClassifier
>>> from pyspark.ml.evaluation import MulticlassClassificationEvaluator
```

Creating a Decision Tree Classifier Instance and Increasing Bin Size

```
>>> classifier = DecisionTreeClassifier(labelCol="Indexed_Type", maxBins=131).fit(train_df)
```

Creating Predictions and Checking the Accuracy

```
>>> predictor = classifier.transform(test_df)
>>> accuracy = MulticlassClassificationEvaluator(labelCol='Indexed_Type', metricName='accuracy').evaluate(predictor)
>>> accuracy
0.9791327029670688
```