from pyspark.sql import SparkSession

from pyspark.sql.functions import col, StringType

from pyspark.ml.feature import StringIndexer, VectorAssembler

from pyspark.ml.classification import RandomForestClassifier

from pyspark.ml.evaluation import MulticlassClassificationEvaluator

from pyspark.ml import Pipeline

# Initialize Spark Session

spark = SparkSession.builder.appName("ML\_Similarity\_Model").getOrCreate()

# Load the processed CSV file

csv\_filename = "/mnt/data/processed\_data\_fixed\_similarity.csv"

df = spark.read.option("header", "true").csv(csv\_filename, inferSchema=True)

# Selecting relevant feature columns

feature\_cols = ["Fuzz\_Ratio", "Fuzz\_Partial\_Ratio", "Fuzz\_Token\_Sort\_Ratio", "Fuzz\_Token\_Set\_Ratio"]

label\_col = "Similarity\_Label"

# Convert the label column to numerical values using StringIndexer

indexer = StringIndexer(inputCol=label\_col, outputCol="label")

# Assemble features into a single vector column

assembler = VectorAssembler(inputCols=feature\_cols, outputCol="features")

# Initialize the classifier (Random Forest)

rf\_classifier = RandomForestClassifier(labelCol="label", featuresCol="features", numTrees=100, maxDepth=5)

# Create a pipeline

pipeline = Pipeline(stages=[indexer, assembler, rf\_classifier])

# Split dataset into train and test (80% Train, 20% Test)

train\_df, test\_df = df.randomSplit([0.8, 0.2], seed=42)

# Train the model

model = pipeline.fit(train\_df)

# Make predictions on the test dataset

predictions = model.transform(test\_df)

# Evaluate the model using Accuracy

evaluator = MulticlassClassificationEvaluator(labelCol="label", metricName="accuracy")

accuracy = evaluator.evaluate(predictions)

print(f"Model Accuracy: {accuracy:.4f}")

# Show some predictions

predictions.select("Value", "Modified\_Value", "Similarity\_Label", "prediction").show(10, truncate=False)