**from pyspark.sql import SparkSession**

**spark = SparkSession.builder.appName('NLP').getOrCreate()**

**df = spark.read.csv('/home/nhom10/spark/python/13\_SMSSpamCollection', inferSchema=True, sep='\t')**

**df = df.withColumnRenamed('\_c0', 'class').withColumnRenamed('\_c1', 'text')**

**df.show()**

**from pyspark.sql.functions import length**

**df = df.withColumn('length', length(df['text']))**

**df.show()**

**df.groupBy('class').mean().show()**

**from pyspark.ml.feature import StringIndexer, Tokenizer, StopWordsRemover, CountVectorizer, IDF, VectorAssembler**

**ham\_spam\_to\_numeric = StringIndexer(inputCol='class', outputCol='label')**

**df\_1 = ham\_spam\_to\_numeric.fit(df).transform(df)**

**df\_1.show()**

**tokenizer = Tokenizer(inputCol='text', outputCol='token\_text')**

**df\_2 = tokenizer.transform(df\_1)**

**df\_2.show()**

**stop\_remove = StopWordsRemover(inputCol='token\_text', outputCol='token\_stop')**

**df\_3 = stop\_remove.transform(df\_2)**

**df\_3.show()**

**count\_vec = CountVectorizer(inputCol='token\_stop', outputCol='count\_vec')**

**df\_4 = count\_vec.fit(df\_3).transform(df\_3)**

**df\_4.show()**

**idf = IDF(inputCol='count\_vec', outputCol='tf-idf')**

**df\_5 = idf.fit(df\_4).transform(df\_4)**

**df\_5.show()**

**transformed\_df = VectorAssembler(inputCols=['length', 'tf-idf'], outputCol='features')**

**df\_6 = transformed\_df.transform(df\_5)**

**df\_6.show()**

**from pyspark.ml import Pipeline**

**df\_pipe = Pipeline(stages=[ham\_spam\_to\_numeric,**

**tokenizer,**

**stop\_remove,**

**count\_vec,**

**idf,**

**transformed\_df])**

**final\_df = df\_pipe.fit(df).transform(df).select('label', 'features')**

**final\_df.show()**

**train\_data, test\_data = final\_df.randomSplit([0.7, 0.3])**

**from pyspark.ml.classification import NaiveBayes**

**classifier = NaiveBayes(featuresCol='features', labelCol='label', predictionCol='prediction')**

**fittied\_classifer = classifier.fit(train\_data)**

**preds = fittied\_classifer.transform(test\_data)**

**preds.show()**

**from pyspark.ml.evaluation import BinaryClassificationEvaluator, MulticlassClassificationEvaluator**

**evaluator = BinaryClassificationEvaluator(rawPredictionCol='prediction', labelCol='label')**

**area\_under\_curve = evaluator.evaluate(preds)**

**accuracy = MulticlassClassificationEvaluator(metricName='accuracy', labelCol='label')**

**accuracy = accuracy.evaluate(preds)**

**print(area\_under\_curve)**

**print(accuracy)**