Spark ML

Spark ML (also known as MLlib) is a machine learning library for Apache Spark, an open-source distributed computing system that can process large-scale data sets in parallel across a cluster of computers.

Spark ML provides a set of high-level APIs for common machine learning algorithms such as classification, regression, clustering, collaborative filtering, and dimensionality reduction. It also includes tools for feature extraction, transformation, and selection.

Spark ML's main advantages are its scalability, distributed processing capabilities, and ability to integrate seamlessly with other Spark components such as Spark SQL, DataFrame API, and streaming. This makes it a popular choice for large-scale machine learning projects where data processing speed and efficiency are important considerations.

Steps to train a model with Spark ML. We will be training a basic Iris Flowers Classification model.

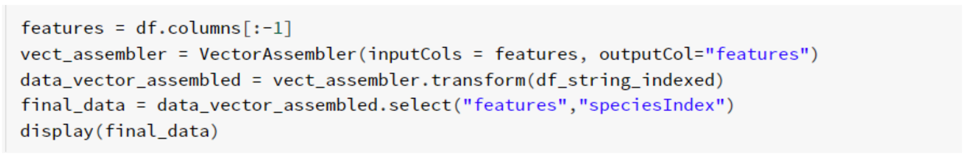
1. Import your dataset

```
df = spark.read.csv("dbfs:/FileStore/IRIS.csv",header=True,inferSchema=True)
display(df)
```

2. Use String Indexer to convert the different names of flowers in the species column to numbers.

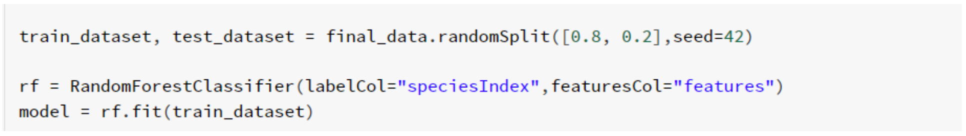
```
species_indexer = StringIndexer(inputCol="species", outputCol="speciesIndex")
df_string_indexed = species_indexer.fit(df).transform(df)
```

3. Use Vector Assembler to combine all the features expect the Species column.



```python
features = df.columns[:-1]
vect_assembler = VectorAssembler(inputCols = features, outputCol="features")
data_vector_assembled = vect_assembler.transform(df_string_indexed)
final_data = data_vector_assembled.select("features","speciesIndex")
display(final_data)
```

4. Split the data into training and testing dataset and train the model.



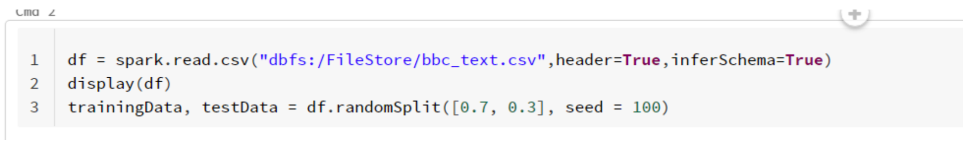
```python
train_dataset, test_dataset = final_data.randomSplit([0.8, 0.2],seed=42)

rf = RandomForestClassifier(labelCol="speciesIndex",featuresCol="features")
model = rf.fit(train_dataset)
```

Spark NLP

Steps to implement a sentiment analysis pipeline with Spark NLP.

1. Import csv file and split into training and test data.



```
df = spark.read.csv("dbfs:/FileStore/bbc_text.csv",header=True,inferSchema=True)
display(df)
trainingData, testData = df.randomSplit([0.7, 0.3], seed = 100)
```

2. Generate word embeddings using BERT , use a deep learning classification model and make a pipeline.

```
document = DocumentAssembler()\
    .setInputCol("text")\
    .setOutputCol("document")


bert = BertSentenceEmbeddings.pretrained("sent_small_bert_L8_512")\
 .setInputCols(["document"])\
 .setOutputCol("sentence_embeddings")

classsifierdl = ClassifierDLApproach()\
  .setInputCols(["sentence_embeddings"])\
  .setOutputCol("class")\
  .setLabelColumn("category")\
  .setMaxEpochs(5)\
  .setEnableOutputLogs(True)

use_clf_pipeline = Pipeline(
    stages = [
        document,
        bert,
        classsifierdl
    ])
```

3. Train the model with training data.

```
use_pipelineModel = use_clf_pipeline.fit(trainingData)
```