Task 10 - Random Forest Classification

In this task, we build an end to end pipeline that reads in data in parquet format, converts it to CSV and loads it into a dataframe, trains a model and perform hyperparameter tuning.

In [18]:
```
# Spark Pipeline Initialization
```

In [2]:
```
%%bash
export version=`python --version |awk '{print $2}' |awk -F"." '{print $1$2}'`

echo $version

if [ $version == '36' ] || [ $version == '37' ]; then
    echo 'Starting installation...'
    pip3 install pyspark==2.4.8 wget==3.2 pyspark2pmml==0.5.1 > install.log 2> install.l
    if [ $? == 0 ]; then
        echo 'Please <<RESTART YOUR KERNEL>> (Kernel->Restart Kernel and Clear All Outpu
    else
        echo 'Installation failed, please check log:'
        cat install.log
    fi
elif [ $version == '38' ] || [ $version == '39' ]; then
    pip3 install pyspark==3.1.2 wget==3.2 pyspark2pmml==0.5.1 > install.log 2> install.l
    if [ $? == 0 ]; then
        echo 'Please <<RESTART YOUR KERNEL>> (Kernel->Restart Kernel and Clear All Outpu
    else
        echo 'Installation failed, please check log:'
        cat install.log
    fi
else
    echo 'Currently only python 3.6, 3.7 , 3.8 and 3.9 are supported, in case you need a
    exit -1
fi
```

```
37
Starting installation...
Please <<RESTART YOUR KERNEL>> (Kernel->Restart Kernel and Clear All Outputs)
```

In [18]:
```
from pyspark import SparkContext, SparkConf #, SQLContext
from pyspark.sql import SparkSession
import os
from pyspark.ml.classification import RandomForestClassifier
from pyspark.sql.types import DoubleType
from pyspark.ml.evaluation import MulticlassClassificationEvaluator
from pyspark.ml import Pipeline
# from pyspark2pmml import PMMLBuilder
from pyspark.ml.feature import StringIndexer
from pyspark.ml.feature import VectorAssembler
from pyspark.ml.feature import MinMaxScaler
import logging
import shutil
import site
import sys
import wget
import re
import pandas as pd
import glob
```

In [6]:
```
if sys.version[0:3] == '3.9':
    url = ('https://github.com/jpmml/jpmml-sparkml/releases/download/1.7.2/'
           'jpmml-sparkml-executable-1.7.2.jar')
    wget.download(url)
    shutil.copy('jpmml-sparkml-executable-1.7.2.jar',
                site.getsitepackages()[0] + '/pyspark/jars/')
elif sys.version[0:3] == '3.8':
    url = ('https://github.com/jpmml/jpmml-sparkml/releases/download/1.7.2/'
           'jpmml-sparkml-executable-1.7.2.jar')
    wget.download(url)
    shutil.copy('jpmml-sparkml-executable-1.7.2.jar',
                site.getsitepackages()[0] + '/pyspark/jars/')
elif sys.version[0:3] == '3.7':
    url = ('https://github.com/jpmml/jpmml-sparkml/releases/download/1.5.12/'
           'jpmml-sparkml-executable-1.5.12.jar')
    wget.download(url)
elif sys.version[0:3] == '3.6':
    url = ('https://github.com/jpmml/jpmml-sparkml/releases/download/1.5.12/'
           'jpmml-sparkml-executable-1.5.12.jar')
    wget.download(url)
else:
    raise Exception('Currently only python 3.6 , 3.7, 3,8 and 3.9 is supported, in case
                    'you need a different version please open an issue at '
                    'https://github.com/IBM/claimed/issues')
```

In [7]:
```
# Creating a spark context class
sc = SparkContext()

# Creating a spark session
spark = SparkSession \
    .builder \
    .appName("Python Spark Random Forest Classification") \
    .getOrCreate()
    # .config("spark.some.config.option", "some-value") \
```

```
22/11/18 17:39:11 WARN util.NativeCodeLoader: Unable to load native-hadoop library for y
our platform... using builtin-java classes where applicable
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLev
el).
```

This notebook does the following:

1. Read in the parquet file we created as part of Task 3.

In [8]:
```
data_parquet = 'data.parquet'
data_csv = 'randomforest.csv'
data_dir = './claimed-component-library/data/'
df = spark.read.parquet(data_dir + data_parquet)
```

2. Convert the parquet file to CSV format.

In [11]:
```
if os.path.exists(data_dir + data_csv):
    os.remove(data_dir + data_csv)
df.coalesce(1).write.option("header", "true").csv(data_dir + data_csv)
file = glob.glob(data_dir + data_csv + '/part-*')
shutil.move(file[0], data_dir + data_csv + '.tmp')
shutil.rmtree(data_dir + data_csv)
shutil.move(data_dir + data_csv + '.tmp', data_dir + data_csv)
```

Out[11]:
```
'./claimed-component-library/data/randomforest.csv'
```

3. Load the CSV file into a dataframe

In [14]:
```
# Read the file using `read_csv` function in pandas
pd_df_csv = pd.read_csv('./claimed-component-library/data/randomforest.csv')
# pd_df_csv.head()

# We use the `createDataFrame` function to load the data into a spark dataframe
sdf = spark.createDataFrame(pd_df_csv)
```

4. Create a 80-20 training and test split with seed=1.

In [15]:
```
# casting feature columns to double
sdf = sdf.withColumn("x", sdf.x.cast(DoubleType()))
sdf = sdf.withColumn("y", sdf.y.cast(DoubleType()))
sdf = sdf.withColumn("z", sdf.z.cast(DoubleType()))

# spliting dataframe into training and testing subsets
splits = sdf.randomSplit([0.8, 0.2], seed=1)
df_train = splits[0]
df_test = splits[1]
```

5. Train a Random Forest model with different hyperparameters listed below and report the best performing hyperparameter combinations:

- number of trees : {10, 20}
- maximum depth : {5, 7}
- use random seed = 1 wherever needed

In [20]:
```
# indexing classes
indexer = StringIndexer(inputCol="class", outputCol="label")
input_columns = ['x', 'y', 'z']
# aggregating feature columns into vector
vectorAssembler = VectorAssembler(inputCols=input_columns,
                                  outputCol="features")
# normalizing features
normalizer = MinMaxScaler(inputCol="features", outputCol="features_norm")

# creating pandas dataframe to keep predictions accuracy
pd_df = pd.DataFrame(columns = ['n_trees', 'max_depth', 'accuracy'])

# hyperparameter testing
for n_trees in [10, 20]:
    for max_depth in [5, 7]:
        rf = RandomForestClassifier(numTrees=n_trees, maxDepth=max_depth, featuresCol="f

        pipeline = Pipeline(stages=[indexer, vectorAssembler, normalizer, rf])

        rf_model = pipeline.fit(df_train)
        predictions = rf_model.transform(df_test)

        # evaluate predictions
        evaluator = MulticlassClassificationEvaluator(labelCol="label", predictionCol="p
        accuracy = evaluator.evaluate(predictions)

        # print accuracy
        print("# Trees = %s" % (n_trees))
        print("Max Depth = %s" % (max_depth))
        print("Accuracy = %s" % (accuracy))

        # add entry to pandas dataframe
        pd_df = pd_df.append({'n_trees' : n_trees, 'max_depth' : max_depth, 'accuracy' :
                              ignore_index = True)
```

```
22/11/18 18:06:31 WARN scheduler.TaskSetManager: Stage 8 contains a task of very large s
ize (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:06:34 WARN scheduler.TaskSetManager: Stage 10 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:06:37 WARN scheduler.TaskSetManager: Stage 12 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:06:37 WARN scheduler.TaskSetManager: Stage 13 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:06:44 WARN scheduler.TaskSetManager: Stage 14 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:06:49 WARN scheduler.TaskSetManager: Stage 16 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:06:56 WARN scheduler.TaskSetManager: Stage 18 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:06:57 WARN scheduler.TaskSetManager: Stage 20 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:06:59 WARN scheduler.TaskSetManager: Stage 22 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:07:00 WARN scheduler.TaskSetManager: Stage 24 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:07:02 WARN scheduler.TaskSetManager: Stage 26 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:07:03 WARN scheduler.TaskSetManager: Stage 27 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:07:08 WARN scheduler.TaskSetManager: Stage 29 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:07:12 WARN scheduler.TaskSetManager: Stage 31 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:07:15 WARN scheduler.TaskSetManager: Stage 33 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
# Trees = 10
Max Depth = 5
Accuracy = 0.3649548599704961
22/11/18 18:07:17 WARN scheduler.TaskSetManager: Stage 35 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:07:19 WARN scheduler.TaskSetManager: Stage 37 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:07:20 WARN scheduler.TaskSetManager: Stage 38 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
```

```
22/11/18 18:07:22 WARN scheduler.TaskSetManager: Stage 39 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:07:25 WARN scheduler.TaskSetManager: Stage 41 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:07:30 WARN scheduler.TaskSetManager: Stage 43 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:07:31 WARN scheduler.TaskSetManager: Stage 45 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:07:32 WARN scheduler.TaskSetManager: Stage 47 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:07:33 WARN scheduler.TaskSetManager: Stage 49 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:07:34 WARN scheduler.TaskSetManager: Stage 51 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:07:36 WARN scheduler.TaskSetManager: Stage 53 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:07:39 WARN scheduler.TaskSetManager: Stage 55 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:07:40 WARN scheduler.TaskSetManager: Stage 56 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:07:42 WARN scheduler.TaskSetManager: Stage 58 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:07:44 WARN scheduler.TaskSetManager: Stage 60 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
```

```
# Trees = 10
Max Depth = 7
Accuracy = 0.38371934759627974
```

```
22/11/18 18:07:48 WARN scheduler.TaskSetManager: Stage 62 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:07:50 WARN scheduler.TaskSetManager: Stage 64 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:07:52 WARN scheduler.TaskSetManager: Stage 66 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:07:53 WARN scheduler.TaskSetManager: Stage 67 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:07:55 WARN scheduler.TaskSetManager: Stage 68 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:07:58 WARN scheduler.TaskSetManager: Stage 70 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:08:03 WARN scheduler.TaskSetManager: Stage 72 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:08:04 WARN scheduler.TaskSetManager: Stage 74 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:08:06 WARN scheduler.TaskSetManager: Stage 76 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:08:08 WARN scheduler.TaskSetManager: Stage 78 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:08:10 WARN scheduler.TaskSetManager: Stage 80 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:08:10 WARN scheduler.TaskSetManager: Stage 81 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:08:13 WARN scheduler.TaskSetManager: Stage 83 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:08:16 WARN scheduler.TaskSetManager: Stage 85 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
```

```
# Trees = 20
Max Depth = 5
```

```
Accuracy = 0.3594295797968959
```

```
22/11/18 18:08:19 WARN scheduler.TaskSetManager: Stage 87 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:08:21 WARN scheduler.TaskSetManager: Stage 89 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:08:23 WARN scheduler.TaskSetManager: Stage 91 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:08:24 WARN scheduler.TaskSetManager: Stage 92 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:08:26 WARN scheduler.TaskSetManager: Stage 93 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:08:29 WARN scheduler.TaskSetManager: Stage 95 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:08:34 WARN scheduler.TaskSetManager: Stage 97 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:08:36 WARN scheduler.TaskSetManager: Stage 99 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:08:37 WARN scheduler.TaskSetManager: Stage 101 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:08:39 WARN scheduler.TaskSetManager: Stage 103 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:08:41 WARN scheduler.TaskSetManager: Stage 105 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:08:44 WARN scheduler.TaskSetManager: Stage 107 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:08:48 WARN scheduler.TaskSetManager: Stage 109 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:08:48 WARN scheduler.TaskSetManager: Stage 110 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:08:52 WARN scheduler.TaskSetManager: Stage 112 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
22/11/18 18:08:55 WARN scheduler.TaskSetManager: Stage 114 contains a task of very large
size (908 KB). The maximum recommended task size is 100 KB.
[Stage 114:==========================================>              (6 + 2) / 8]
```

```
# Trees = 20
Max Depth = 7
Accuracy = 0.38959535022463476
```

6. Use the accuracy metric when evaluating the model with different hyperparameters

In [21]:

```
# print parameters with highest accuracy
pd_df[pd_df['accuracy'] == pd_df['accuracy'].max()]
```

Out[21]:

	n_trees	max_depth	accuracy
3	20.0	7.0	0.389595