Mod13_Sesion03MLibArbolClasificacion

September 23, 2023

1 Árboles de Clasificación

[1]:

```
from pyspark import SparkContext
from pyspark.sql import SQLContext
```

[6]:

```
#Leer el contenido de una carpeta
#Para leer de HDFS usar
# hdfs:///tmp/dcd/OnTimeDB
#Para leer de local usar
# file:/home/cloudera/dcd/OnTimeDB/

## Descargar archivo zip y subir al cluster
!wget https://github.com/omarmendoza564/datos/raw/main/datos/OnTimeDB.zip -O /
 ↪home/sergio_ibarra1795/OnTimeDB.zip
!unzip -o /home/sergio_ibarra1795/OnTimeDB.zip -d /home/sergio_ibarra1795/
!ls -la /home/sergio_ibarra1795/OnTimeDB
!hdfs dfs -mkdir /tmp/dcd/OnTimeDB
!hdfs dfs -put /home/sergio_ibarra1795/OnTimeDB/ /tmp/dcd/
```

```
--2023-09-23 18:24:40--
https://github.com/omarmendoza564/datos/raw/main/datos/OnTimeDB.zip
Resolving github.com (github.com)… 140.82.112.3
Connecting to github.com (github.com)|140.82.112.3|:443… connected.
HTTP request sent, awaiting response… 302 Found
Location:
https://raw.githubusercontent.com/omarmendoza564/datos/main/datos/OnTimeDB.zip
[following]
--2023-09-23 18:24:40--
https://raw.githubusercontent.com/omarmendoza564/datos/main/datos/OnTimeDB.zip
Resolving raw.githubusercontent.com (raw.githubusercontent.com)…
185.199.111.133, 185.199.108.133, 185.199.109.133, …
Connecting to raw.githubusercontent.com
(raw.githubusercontent.com)|185.199.111.133|:443… connected.
HTTP request sent, awaiting response… 200 OK
Length: 484951 (474K) [application/zip]
Saving to: ‘/home/sergio_ibarra1795/OnTimeDB.zip’

/home/sergio_ibarra 100%[===================>] 473.58K  --.-KB/s    in 0.03s
```

```
2023-09-23 18:24:40 (13.3 MB/s) - ‘/home/sergio_ibarra1795/OnTimeDB.zip’ saved
[484951/484951]

Archive:  /home/sergio_ibarra1795/OnTimeDB.zip
   creating: /home/sergio_ibarra1795/OnTimeDB/
  inflating: /home/sergio_ibarra1795/OnTimeDB/.part-00000.crc
  inflating: /home/sergio_ibarra1795/OnTimeDB/.part-00001.crc
  inflating: /home/sergio_ibarra1795/OnTimeDB/.part-00002.crc
  inflating: /home/sergio_ibarra1795/OnTimeDB/.part-00003.crc
  inflating: /home/sergio_ibarra1795/OnTimeDB/.part-00004.crc
  inflating: /home/sergio_ibarra1795/OnTimeDB/.part-00005.crc
  inflating: /home/sergio_ibarra1795/OnTimeDB/.part-00006.crc
  inflating: /home/sergio_ibarra1795/OnTimeDB/part-00000
  inflating: /home/sergio_ibarra1795/OnTimeDB/part-00001
  inflating: /home/sergio_ibarra1795/OnTimeDB/part-00002
  inflating: /home/sergio_ibarra1795/OnTimeDB/part-00003
  inflating: /home/sergio_ibarra1795/OnTimeDB/part-00004
  inflating: /home/sergio_ibarra1795/OnTimeDB/part-00005
  inflating: /home/sergio_ibarra1795/OnTimeDB/part-00006
  inflating: /home/sergio_ibarra1795/OnTimeDB/_SUCCESS
total 3276
drwxrwxr-x 2 root              root                4096 Nov 23  2021 .
drwxr-xr-x 7 sergio_ibarra1795 sergio_ibarra1795   4096 Sep 23 18:24 ..
-rw-r--r-- 1 root              root                7428 Jul 18  2018
.part-00000.crc
-rw-r--r-- 1 root              root                2256 Jul 18  2018
.part-00001.crc
-rw-r--r-- 1 root              root                3184 Jul 18  2018
.part-00002.crc
-rw-r--r-- 1 root              root                2916 Jul 18  2018
.part-00003.crc
-rw-r--r-- 1 root              root                6948 Jul 18  2018
.part-00004.crc
-rw-r--r-- 1 root              root                2012 Jul 18  2018
.part-00005.crc
-rw-r--r-- 1 root              root                1076 Jul 18  2018
.part-00006.crc
-rw-r--r-- 1 root              root                   0 Jul 18  2018 _SUCCESS
-rw-r--r-- 1 root              root              949404 Jul 18  2018 part-00000
-rw-r--r-- 1 root              root              287594 Jul 18  2018 part-00001
-rw-r--r-- 1 root              root              406415 Jul 18  2018 part-00002
-rw-r--r-- 1 root              root              372196 Jul 18  2018 part-00003
-rw-r--r-- 1 root              root              888266 Jul 18  2018 part-00004
-rw-r--r-- 1 root              root              256021 Jul 18  2018 part-00005
-rw-r--r-- 1 root              root              136582 Jul 18  2018 part-00006
mkdir: `/tmp/dcd/OnTimeDB': File exists
```

[7]:

```
bd = sqlContext.read.csv("hdfs:///tmp/dcd/OnTimeDB/", inferSchema=True,␣
 ↪header=True)
sqlContext.registerDataFrameAsTable(bd, "bd")
bd.count()
```

[7]: 30466

[8]:

```
bd.show(10)
```

```
[Stage 5:>                                                          (0 + 1) / 1]

+----+-----+----------+---------+----------+-------------+-------+--------+-----
---+------+----+--------+---------+--------+------------+------------+--------+-
------------+-----------------+------------------+-------+-----------+-------+
|Year|Month|DayofMonth|DayOfWeek|CRSDepTime|UniqueCarrier|TailNum|ArrDelay|DepDe
lay|Origin|Dest|Distance|Cancelled|Diverted|CarrierDelay|WeatherDelay|NASDelay|S
ecurityDelay|LateAircraftDelay|              LogD|Retraso|RetrasoNeto|Horario|
+----+-----+----------+---------+----------+-------------+-------+--------+-----
---+------+----+--------+---------+--------+------------+------------+--------+-
------------+-----------------+------------------+-------+-----------+-------+
|2016|   12|         1|        4|       845|           AA| N8ARAA|    -7.0|
-5.0|   LAX| DFW|  1235.0|      0.0|     0.0|         0.0|         0.0|     0.0|
0.0|              0.0|3.0916669575956846|      0|       -2.0|      2|
|2016|   12|         2|        5|       845|           AA| N8ARAA|    -3.0|
5.0|   LAX| DFW|  1235.0|      0.0|     0.0|         0.0|         0.0|     0.0|
0.0|              0.0|3.0916669575956846|      0|       -8.0|      2|
|2016|   12|         3|        6|       845|           AA| N8ABAA|    -3.0|
-3.0|   LAX| DFW|  1235.0|      0.0|     0.0|         0.0|         0.0|     0.0|
0.0|              0.0|3.0916669575956846|      0|        0.0|      2|
|2016|   12|         4|        7|       845|           AA| N8ABAA|    -2.0|
-7.0|   LAX| DFW|  1235.0|      0.0|     0.0|         0.0|         0.0|     0.0|
0.0|              0.0|3.0916669575956846|      0|        5.0|      2|
|2016|   12|         5|        1|       845|           AA| N8ACAA|    -2.0|
-6.0|   LAX| DFW|  1235.0|      0.0|     0.0|         0.0|         0.0|     0.0|
0.0|              0.0|3.0916669575956846|      0|        4.0|      2|
|2016|   12|         6|        2|       845|           AA| N867AA|     0.0|
-1.0|   LAX| DFW|  1235.0|      0.0|     0.0|         0.0|         0.0|     0.0|
0.0|              0.0|3.0916669575956846|      0|        1.0|      2|
|2016|   12|         7|        3|       845|           AA| N8ACAA|    -6.0|
0.0|   LAX| DFW|  1235.0|      0.0|     0.0|         0.0|         0.0|     0.0|
0.0|              0.0|3.0916669575956846|      0|       -6.0|      2|
|2016|   12|         8|        4|       845|           AA| N8AKAA|     7.0|
0.0|   LAX| DFW|  1235.0|      0.0|     0.0|         0.0|         0.0|     0.0|
0.0|              0.0|3.0916669575956846|      0|        7.0|      2|
|2016|   12|         9|        5|       845|           AA| N8ARAA|    -9.0|
-1.0|   LAX| DFW|  1235.0|      0.0|     0.0|         0.0|         0.0|     0.0|
0.0|              0.0|3.0916669575956846|      0|       -8.0|      2|
|2016|   12|        10|        6|       845|           AA| N8AKAA|    -2.0|
-1.0|   LAX| DFW|  1235.0|      0.0|     0.0|         0.0|         0.0|     0.0|
0.0|              0.0|3.0916669575956846|      0|       -1.0|      2|
+----+-----+----------+---------+----------+-------------+-------+--------+-----
---+------+----+--------+---------+--------+------------+------------+--------+-
------------+-----------------+------------------+-------+-----------+-------+
only showing top 10 rows
```

[9]:

```
# Ver las variables disponibles
bd.dtypes
```

[9]:

```
[('Year', 'int'),
 ('Month', 'int'),
 ('DayofMonth', 'int'),
 ('DayOfWeek', 'int'),
 ('CRSDepTime', 'int'),
 ('UniqueCarrier', 'string'),
 ('TailNum', 'string'),
 ('ArrDelay', 'double'),
 ('DepDelay', 'double'),
 ('Origin', 'string'),
 ('Dest', 'string'),
 ('Distance', 'double'),
 ('Cancelled', 'double'),
 ('Diverted', 'double'),
 ('CarrierDelay', 'double'),
 ('WeatherDelay', 'double'),
 ('NASDelay', 'double'),
 ('SecurityDelay', 'double'),
 ('LateAircraftDelay', 'double'),
 ('LogD', 'double'),
 ('Retraso', 'int'),
 ('RetrasoNeto', 'double'),
 ('Horario', 'int')]
```

[10]:

```
spark.sql("SELECT year, Retraso, RetrasoNeto, Horario from bd LIMIT 10").show()
```

```
+----+-------+-----------+-------+
|year|Retraso|RetrasoNeto|Horario|
+----+-------+-----------+-------+
|2016|      0|       -2.0|      2|
|2016|      0|       -8.0|      2|
|2016|      0|        0.0|      2|
|2016|      0|        5.0|      2|
|2016|      0|        4.0|      2|
|2016|      0|        1.0|      2|
|2016|      0|       -6.0|      2|
|2016|      0|        7.0|      2|
|2016|      0|       -8.0|      2|
|2016|      0|       -1.0|      2|
+----+-------+-----------+-------+
```

[11]:

```
#Agregar una variable numerica (IndexUniqueCarrier) basada en la variable␣
 ↪alfanumerica
#de la compañia que opera el vuelo (UniqueCarrier)

from pyspark.ml.feature import StringIndexer

indexer =␣
 ↪StringIndexer(inputCol='UniqueCarrier',outputCol='IndexUniqueCarrier') #el␣
 ↪índice empieza en el 0!
bd1=indexer.fit(bd).transform(bd)

#Se muestra el numero de vuelos operados por cada compañia, la variable␣
 ↪IndexUniqueCarrier
#ya se puede utilizar en el modelo

bd1.groupBy('UniqueCarrier','IndexUniqueCarrier').count().
 ↪sort('IndexUniqueCarrier').show()
```

```
[Stage 12:============================>                          (1 + 1) / 2]

+-------------+------------------+-----+
|UniqueCarrier|IndexUniqueCarrier|count|
+-------------+------------------+-----+
|           AA|               0.0| 8853|
|           UA|               1.0| 6112|
|           WN|               2.0| 5395|
|           DL|               3.0| 4239|
|           VX|               4.0| 1703|
|           NK|               5.0| 1581|
|           F9|               6.0| 1295|
|           OO|               7.0| 1166|
|           B6|               8.0|  121|
|           EV|               9.0|    1|
+-------------+------------------+-----+
```

1.1 Ajuste del modelo

[12]:
```
from pyspark.ml.feature import VectorAssembler, StringIndexer
from pyspark.sql.functions import col

#Crear un arreglo de variables predictoras llamdo 'features'
#ArrDelay representa el numero de minutos que un vuelo tiene de retraso
#Si un vuelo llega con mas de 15 minutos de retraso la variable 'Retraso' tiene␣
 ↪valor 1
#Renombrar la variable objetivo (Retraso) como 'label'

#En el caso particular de los árboles de clasificación, la variable objetivo␣
 ↪debe ser de tipo doble.
#Por lo tanto, transformar la variable a tipo doble.
#Además, la variable debe estar convertida a través de la función stringIndexer
#para poder ser analizada por el modelo
#Por lo tanto, la variable de trabajo, en este caso, será 'label2'


a1  = VectorAssembler(
    inputCols=['DepDelay','Distance','DayOfWeek',
               'CRSDepTime','IndexUniqueCarrier'],
    outputCol='features')

bd2 = a1.transform(bd1).select(col("Retraso").cast('double').
 ↪alias("label"),'features')

stringIndexer = StringIndexer(inputCol = 'label', outputCol = 'label2')
sI = stringIndexer.fit(bd2)
bd2 = sI.transform(bd2)
bd2.dtypes
```

[12]:
```
[('label', 'double'), ('features', 'vector'), ('label2', 'double')]
```

[14]:
```
#Mostrar un solo renglon de la BD
bd2.show(5)
```

```
[Stage 19:>                                                         (0 + 1) / 1]

+-----+--------------------+------+
|label|            features|label2|
+-----+--------------------+------+
|  0.0|[-5.0,1235.0,4.0,...|   0.0|
|  0.0|[5.0,1235.0,5.0,8...|   0.0|
|  0.0|[-3.0,1235.0,6.0,...|   0.0|
|  0.0|[-7.0,1235.0,7.0,...|   0.0|
|  0.0|[-6.0,1235.0,1.0,...|   0.0|
+-----+-------------------+------+
only showing top 5 rows
```

1.1.1 Partición Test - Train

```
[15]: #70% Train
      #30% Test
      (bd_train, bd_test) = bd2.randomSplit([0.7, 0.3],seed=123)
      print("Renglones de la BD Train: ", bd_train.count())
      print("Renglones de la BD Test: ",bd_test.count())
```

```
Renglones de la BD Train:  21219

[Stage 23:==============================>                          (1 + 1) / 2]

Renglones de la BD Test:  9247
```

```
[16]: # Utilizamos el modelo DecisionTreeClassifier para generar un prediccion basado␣
        ↪en los features
      # disponibles
      #Con una profundidad (maxDepth) de 5
      #Espeficiar la variable objetivo (labelCol)

      from pyspark.ml.classification import DecisionTreeClassifier as DTC

      rt = DTC(maxDepth=5, labelCol = 'label2')

      model = rt.fit(bd_train)
      pred = model.transform(bd_train)
```

```
[17]: #La columna rawPrediction, está especificando el número de casos negativos y ␣
        ↪positivos
      #en cada uno de los nodos terminales pertinentes para cada observación.
      #[Casos para 0, Casos para 1]

      #El campo probability muestra la probabilidad de ser 0 o 1
      #El valor predicho, estableciendo un punto de corte del 50% se muestra en el␣
        ↪campo prediction

      pred.show()
```

```
+-----+-------------------+------+---------------+-------------------+--------
--+
|label|            features|label2|  rawPrediction|
probability|prediction|
+-----+--------------------+------+---------------+--------------------+--------
--+
|  0.0|[-21.0,868.0,6.0,…|   0.0|[13665.0,985.0]|[0.93276450511945…|
0.0|
|  0.0|[-20.0,1440.0,6.0…|   0.0|[13665.0,985.0]|[0.93276450511945…|
0.0|
|  0.0|[-19.0,1440.0,3.0…|   0.0|[13665.0,985.0]|[0.93276450511945…|
0.0|
|  0.0|[-18.0,602.0,5.0,…|   0.0|[13665.0,985.0]|[0.93276450511945…|
0.0|
|  0.0|[-17.0,888.0,6.0,…|   0.0|[13665.0,985.0]|[0.93276450511945…|
0.0|
|  0.0|[-17.0,1440.0,1.0…|   0.0|[13665.0,985.0]|[0.93276450511945…|
0.0|
|  0.0|[-17.0,1744.0,1.0…|   0.0|[13665.0,985.0]|[0.93276450511945…|
0.0|
|  0.0|[-16.0,641.0,6.0,…|   0.0|[13665.0,985.0]|[0.93276450511945…|
0.0|
|  0.0|[-16.0,868.0,6.0,…|   0.0|[13665.0,985.0]|[0.93276450511945…|
0.0|
|  0.0|[-15.0,731.0,1.0,…|   0.0|[13665.0,985.0]|[0.93276450511945…|
0.0|
|  0.0|[-15.0,868.0,3.0,…|   0.0|[13665.0,985.0]|[0.93276450511945…|
0.0|
|  0.0|[-15.0,888.0,4.0,…|   0.0|[13665.0,985.0]|[0.93276450511945…|
0.0|
|  0.0|[-15.0,888.0,5.0,…|   0.0|[13665.0,985.0]|[0.93276450511945…|
0.0|
|  0.0|[-15.0,1464.0,6.0…|   0.0|[13665.0,985.0]|[0.93276450511945…|
0.0|
|  0.0|[-15.0,1514.0,2.0…|   0.0|[13665.0,985.0]|[0.93276450511945…|
0.0|
|  0.0|[-15.0,1514.0,4.0…|   0.0|[13665.0,985.0]|[0.93276450511945…|
0.0|
|  0.0|[-14.0,236.0,2.0,…|   0.0|[13665.0,985.0]|[0.93276450511945…|
0.0|
|  0.0|[-14.0,236.0,3.0,…|   0.0|[13665.0,985.0]|[0.93276450511945…|
0.0|
|  0.0|[-14.0,236.0,4.0,…|   0.0|[13665.0,985.0]|[0.93276450511945…|
0.0|
|  0.0|[-14.0,255.0,3.0,…|   0.0|[13665.0,985.0]|[0.93276450511945…|
0.0|
+-----+--------------------+------+---------------+--------------------+--------
--+
only showing top 20 rows
```

```
[18]: #Validar el modelo
      #Calcular el Areba bajo la curva
      #El AUC proporciona una medición agregada del rendimiento en todos los umbrales␣
       ↪de clasificación
      #posibles

      from pyspark.ml.evaluation import BinaryClassificationEvaluator as BCE
      print('AUC=',BCE(metricName="areaUnderROC", rawPredictionCol = 'probability').
       ↪evaluate(pred))
```

```
AUC= 0.8966334172544019
```

```
[19]: #Generar una tabla de frecuencias de las distintan probabilidades, es decir de␣
       ↪los distintos
      #nodos terminales

      pred.groupBy('probability').count().sort('count').show(50)
```

```
[Stage 52:=============================>                            (1 + 1) / 2]

+--------------------+-----+
|         probability|count|
+--------------------+-----+
|           [1.0,0.0]|    2|
|[0.65116279069767…|   43|
|[0.35526315789473…|   76|
|[0.25984251968503…|  127|
|[0.66666666666666…|  144|
|[0.50531914893617…|  188|
|[0.09150326797385…|  306|
|[0.46683673469387…|  392|
|[0.27331887201735…|  461|
|[0.70650032829940…| 1523|
|[0.01118838826731…| 3307|
|[0.93276450511945…|14650|
+--------------------+-----+
```

```
[20]: #Generar la matriz de confusion
      pred.groupBy('label','prediction').count().show()
```

```
[Stage 55:=============================>                            (1 + 1) / 2]

+-----+----------+-----+
|label|prediction|count|
+-----+----------+-----+
|  1.0|       1.0| 4235|
|  0.0|       1.0|  434|
|  1.0|       0.0| 1588|
|  0.0|       0.0|14962|
+-----+----------+-----+
```

[21]:

```
#Generar algunas estadísticas para tener una idea de cómo fueron las␣
 ↪predicciones

numSuccesses = pred.where("""(prediction = 0.0 AND label2 = 0.0) OR (prediction␣
 ↪= 1.0 AND label2 = 1.0)""").count()
numInspections = pred.count()

print ("Se realizaron", numInspections, "inspeciones y existen", numSuccesses,␣
 ↪"predicciones existosas")
print ("Esta es una tasa de éxito del", str((float(numSuccesses) /␣
 ↪float(numInspections)) * 100) + "%")
```

```
[Stage 61:=============================>                              (1 + 1) / 2]

Se realizaron 21219 inspeciones y existen 19197 predicciones existosas
Esta es una tasa de éxito del 90.47080446769404%
```

[22]:

```
# DecisionTreeClassifier(featuresCol="features",
#    labelCol="label",
#    predictionCol="prediction",
#    probabilityCol="probability",
#    rawPredictionCol="rawPrediction",
#    maxDepth=5,
#    maxBins=32,
#    minInstancesPerNode=1,
#    minInfoGain=0.0,
#    maxMemoryInMB=256,
#    impurity="gini"  / impurity="entropy" )
```

[23]:

```
#Cambiando las propiedades del ejecutamos un nuevo arbol con una profundidad de␣
 ↪20
rt = DTC(maxDepth=20, labelCol = 'label2')
model = rt.fit(bd_train)
pred = model.transform(bd_train)

#Evaluar el modelo con AUC que ahora ha aumentado
```

```
print('AUC=',BCE(metricName="areaUnderROC", rawPredictionCol = 'probability').
 ↪evaluate(pred))
```

```
AUC= 0.9990185263935085
```

1.1.2 Validación externa

[24]:
```
#Validación externa con la BD test

predtest = model.transform(bd_test)

print('AUC=',BCE(metricName="areaUnderROC",rawPredictionCol = 'probability').
 ↪evaluate(predtest))
```

```
AUC= 0.8193169919620452
```

[ ]:
```
```