tarea_7

FINISHED

```
%spark.pyspark

##########
# cargo los paquetes

from pyspark.sql import SparkSession
from pyspark.sql.types import DoubleType
from pyspark.sql.functions import *
from pyspark.ml.feature import OneHotEncoder, StringIndexer
from pyspark.ml.feature import VectorAssembler
from pyspark.ml.regression import LinearRegression
from pyspark.ml.regression import GBTRegressor
from pyspark.ml import Pipeline
from pyspark.ml.tuning import CrossValidator, ParamGridBuilder
from pyspark.ml.evaluation import RegressionEvaluator

#spark = SparkSession.builder.master("local[3]").getOrCreate()

##########
# cargo los datos

data = spark.read.csv('s3a://audiracmichelle/flights/flights.csv', header =True)


# selecciono las columnas que voy a utilizar
data = data.withColumn("HOUR_DEPARTURE", floor(data.SCHEDULED_DEPARTURE / 100))
data = data.select("DAY_OF_WEEK", "AIRLINE", "HOUR_DEPARTURE",
data["DEPARTURE_DELAY"].cast(DoubleType())).where("CANCELLED = 0")

data = data.withColumnRenamed('DEPARTURE_DELAY',
                              'label')

data = data.dropna()

##########
# separo en train y test sets

train_data, test_data = data.randomSplit([0.7, 0.3])

##########
# transformacion de variables para el pipeline

day_of_week_indexer = StringIndexer(inputCol="DAY_OF_WEEK", outputCol="DAY_OF_WEEK_CATEGORICAI
airline_indexer = StringIndexer(inputCol="AIRLINE", outputCol="AIRLINE_CATEGORICAL")
hour_departure_indexer = StringIndexer(inputCol="HOUR_DEPARTURE", outputCol="HOUR_DEPARTURE_CA

day_of_week_encoder = OneHotEncoder(inputCol="DAY_OF_WEEK_CATEGORICAL", outputCol="DAY_OF_WEEI
airline_encoder = OneHotEncoder(inputCol="AIRLINE_CATEGORICAL", outputCol="AIRLINE_DUMMY")
hour_departure_encoder = OneHotEncoder(inputCol="HOUR_DEPARTURE_CATEGORICAL", outputCol="HOUR_

assembler = VectorAssembler(inputCols = ["DAY_OF_WEEK_DUMMY", "AIRLINE_DUMMY", "HOUR_DEPARTURI
                            outputCol = "features")
```

```
##########
# modelos

# modelo 1

lr = LinearRegression(maxIter=10)

# modelo 2

gbt = GBTRegressor(maxIter=20)


modelos = [lr, gbt]

############
# grids

# defino el grid de parametros a evaluar del modelo 1

paramGrid_lr = ParamGridBuilder() \
     .addGrid(lr.regParam, [0.1, 0.01, 0.001]) \
     .addGrid(lr.elasticNetParam, [0.0, 0.5, 1.0]) \
     .build()

# defino el grid de parametros a evaluar del modelo 2

paramGrid_gbt = ParamGridBuilder() \
     .addGrid(gbt.maxDepth, [1, 2, 3]) \
     .addGrid(gbt.subsamplingRate, [0.33, 0.66, 1.0]) \
     .build()

grids = [paramGrid_lr, paramGrid_gbt]
```

Took 50 sec. Last updated by anonymous at April 25 2018, 3:16:47 PM.

```
%spark.pyspark
```

FINISHED

```
##########
# magic loop

bestModel = list()
eval = list()

for i in range(2):
    # defino el pipeline
    pipeline = Pipeline(stages=[day_of_week_indexer,
                            airline_indexer,
                            hour_departure_indexer,
                            day_of_week_encoder,
                            airline_encoder,
                            hour_departure_encoder,
                            assembler,
                            modelos[i]])

    # defino el cross validator
    crossval = CrossValidator(estimator=pipeline,
                          estimatorParamMaps=grids[i],
                          evaluator=RegressionEvaluator(),
```

```
                            numFolds = 10)

    # genero los modelos
    cvModel = crossval.fit(train_data)

    eval.append(RegressionEvaluator().evaluate(cvModel.transform(test_data)))
    bestModel.append(cvModel.bestModel.stages[-1])
```

Took 2 hrs 3 min 24 sec. Last updated by anonymous at April 25 2018, 6:10:35 PM.

```
%spark.pyspark
# vemos la evaluacion del modelo 1
print(eval[0])
```

FINISHED

```
36.4852688851
```

Took 0 sec. Last updated by anonymous at April 25 2018, 6:23:50 PM.

```
%spark.pyspark
# obtengo mejores parametros del modelo 1
print(bestModel[0].extractParamMap().get(bestModel[0].getParam('regParam')))
print(bestModel[0].extractParamMap().get(bestModel[0].getParam('elasticNetParam')))
```

FINISHED

```
0.01
0.0
```

Took 0 sec. Last updated by anonymous at April 25 2018, 6:23:55 PM.

```
%spark.pyspark
# vemos la evaluacion del modelo 2
print(eval[1])
```

FINISHED

```
36.52107874
```

Took 0 sec. Last updated by anonymous at April 25 2018, 6:24:06 PM.

```
%spark.pyspark
# obtengo mejores parametros del modelo 2
print(bestModel[1].extractParamMap().get(bestModel[1].getParam('maxDepth')))
print(bestModel[1].extractParamMap().get(bestModel[1].getParam('subsamplingRate')))
```

FINISHED

```
3
0.66
```

Took 0 sec. Last updated by anonymous at April 25 2018, 6:24:20 PM.

```
%spark.pyspark
```

READY