```
import org.apache.spark.ml.feature.(Normalizer, PCA, VectorAssembler)
import org.apache.spark.mlile.evaluation.RegressionMetrics
import org.apache.spark.mlile.evaluation.RegressionMetrics
import org.apache.spark.ml.regression.
import org.apache.spark.ml.regression.
import org.apache.spark.ml.regression.
import org.apache.spark.ml.pipeline

// Lecture ded archive CSV on un objeto DataFrame

val data = spark.read
.format("cav")
.option("delimiter", ";")
.option("delimiter", ";")
.option("delimiter", ";")
.option("delimiter", ",")
.option("delimiter delimiter delimiter delimiter delimiter delimiter delimiter delimiter delimiter delimiter
```

```
.format("csv")
.option("delimiter", ";")
.option("delimiter", ";")
.option("delimiter", ";")
.option("northivatue", "Mx")
.option("northivatue", "Mx")
.load("hdfs://Juser/hdfs/demos/machine_learning/ventas_helados.csv")

// Se construye una tabla temporal con los datos de las ventas
data.createOrReplaceTempView("Ventas")

val DF = data.na.drop

val lista_heladerias = List(3409,3859,3859,3859,5280,6021,6228,7190,7357,9015,9240,11344,12080,12337,12540,13828,14148,14679,15499,15532,15566,15604,16565,16916,17522
.j1568,17943,18227,18362,21227,18837,20759)

for (heladeria < lista_heladerias){
    val data = DF.filter("Id_heladeria" === heladeria)

    val split = dataf.randomSplit(Array(0.7, 0.3)) //Se utiliza 70% para entrenamiento y 30% de evaluación
    val training = split(0)

    val essembler = new VectorAssembler()
    .setinputCols(Array("Id_heladeria","Consumo_In_Situ","Productos_Premium","Anyo","Mes","D$a","Tipo","Temperatura","HorasAbierto"))
    .setinputCols(Array("Id_heladeria","Consumo_In_Situ","Productos_Premium","Anyo","Mes","D$a","Tipo","Temperatura","HorasAbierto"))

val pt = new GBTRegressor()
    .setinputCols("features")

val pipeline = new Pipeline()
    .setitages(Array(assembler, gbt))

// Generar el modelo para una heladería concreta
vala model = pipeline.fit(crasining)

// Almacar en disco cada modelo construído
model.write.overwrite.save("RegressionModelHeladeria"+heladeria)
val predictions = model.transform(test)
predictions.createOrnefapaceTempView("Predicciones")
```

```
%spark2.spark

// Lectura del archivo CSV en un objeto DataFrame
val data = spark.read
    .format("csv")
    .option("delimiter", ";")
    .option("header", "true")
    .option("inferSchema", "true")
    .option("nullValue", "NA")
    .load("hdfs:///user/hdfs/demos/machine_learning/venters_helados.csv")

// Se construye una tabla temporal con los datos de las ventas
data.createOrReplaceTempView("Ventas")
```

Anyo	▼ Mes	▼ Dia	▼ Ventas	Prediccion	¥
2013	5	9	903	1467.3070545683204	
2013	5	12	1770	1867.003761537395	
2013	5	14	708	1288.7788047880301	
2013	5	15	948	1040.8762049213692	
2013	5	19	1616	1403.219292983964	
2013	5	23	996	1489.9226567412406	
2013	5	24	1501	1489.9226567412406	
2013	5	27	816	1268.085985849021	

ld_heladeria 🔻	Des_Heladeria 🔻	Comunidad_Autonoma	Consumo_In_Situ	Productos_Premium	HorasAbierto v	Fecha	Anyo	Mes	Dia ^
3409	PLAYA POSTIGUET	Comunidad Valenciana	0	0	17	2013-05-09 00:00:00.0	2013	5	9
3409	PLAYA POSTIGUET	Comunidad Valenciana	0	0	17	2013-05-10 00:00:00.0	2013	5	10
3409	PLAYA POSTIGUET	Comunidad Valenciana	0	0	17	2013-05-11 00:00:00.0	2013	5	11
3409	PLAYA POSTIGUET	Comunidad Valenciana	0	0	17	2013-05-12 00:00:00.0	2013	5	12
3409	PLAYA POSTIGUET	Comunidad Valenciana	0	0	17	2013-05-13 00:00:00.0	2013	5	13
3409	PLAYA POSTIGUET	Comunidad Valenciana	0	0	17	2013-05-14 00:00:00.0	2013	5	14
3409	PLAYA POSTIGUET	Comunidad Valenciana	0	0	17	2013-05-15 00:00:00.0	2013	5	15
3409	PLAYA POSTIGUET	Comunidad Valenciana	0	0	17	2013-05-16 00:00:00.0	2013	5	16
<									>

Heladeria:

ALICANTE - PLAYA POSTIGUET

Temperatura:

20

Horas abierto:

10

Consumo in situ:

No

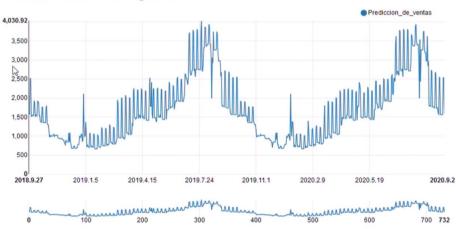
Productos premium:

No

Rango de fechas para predicción:

September 28, 2018 September 28, 2020 
September 28, 2020

Predicción de ventas basada en regresión



Predecir