linearRegressionBostonHousing

February 7, 2025

[3]:
```
#boston_housing data and use linear regression
from pyspark.sql import SparkSession
from pyspark.ml.regression import LinearRegression
from pyspark.ml.feature import VectorAssembler
from pyspark.ml.evaluation import RegressionEvaluator
import logging
import warnings

# Suppress PySpark and Py4J warnings
logging.getLogger("py4j").setLevel(logging.ERROR)
logging.getLogger("pyspark").setLevel(logging.ERROR)
logging.getLogger("sparkConf").setLevel(logging.ERROR)

# Suppress Python warnings
warnings.filterwarnings("ignore")

# Step 1: Initialize SparkSession
spark = SparkSession.builder.appName("LinearRegressionBostonHousing").
 ↪master("spark://spark-master:7077").getOrCreate()

# Step 2: Load the dataset from a CSV file
file_path = "/spark/user/boston_housing.csv"  # Replace with the path to your␣
 ↪CSV file
df = spark.read.csv(file_path, header=True, inferSchema=True)
df.show(5)
```

```
+-------+----+-----+----+-----+-----+----+------+---+---+-------+------+-----+--
--+
|   crim|  zn|indus|chas|  nox|   rm| age|   dis|rad|tax|ptratio|
b|lstat|medv|
+-------+----+-----+----+-----+-----+----+------+---+---+-------+------+-----+--
--+
|0.00632|18.0| 2.31|   0|0.538|6.575|65.2|  4.09|  1|296|   15.3| 396.9|
4.98|24.0|
|0.02731| 0.0| 7.07|   0|0.469|6.421|78.9|4.9671|  2|242|   17.8| 396.9|
9.14|21.6|
|0.02729| 0.0| 7.07|   0|0.469|7.185|61.1|4.9671|  2|242|   17.8|392.83|
```

```
4.03|34.7|
|0.03237| 0.0| 2.18|   0|0.458|6.998|45.8|6.0622|  3|222|   18.7|394.63|
2.94|33.4|
|0.06905| 0.0| 2.18|   0|0.458|7.147|54.2|6.0622|  3|222|   18.7| 396.9|
5.33|36.2|
+-------+----+-----+----+-----+-----+----+------+---+---+-------+------+-----+--
--+
only showing top 5 rows
```

[4]:
```
# Step 3: Prepare the data for Linear Regression
# Combine all feature columns into a single vector column
feature_columns = df.columns[:-1]  # All columns except the last one (target)
assembler = VectorAssembler(inputCols=feature_columns, outputCol="features")
df = assembler.transform(df)

# Step 4: Split the data into training and testing sets
train_data, test_data = df.randomSplit([0.8, 0.2], seed=42)

# Step 5: Create and train the Linear Regression model
lr = LinearRegression(featuresCol="features", labelCol="medv")  # 'medv' is the␣
 ↪target column
lr_model = lr.fit(train_data)

# Step 6: Make predictions on the test data
predictions = lr_model.transform(test_data)

# Step 7: Evaluate the model
evaluator = RegressionEvaluator(labelCol="medv", predictionCol="prediction",␣
 ↪metricName="rmse")
rmse = evaluator.evaluate(predictions)
print(f"Root Mean Squared Error (RMSE): {rmse}")
```

```
25/02/04 01:16:58 WARN SparkConf: The configuration key 'spark.executor.port'
has been deprecated as of Spark 2.0.0 and may be removed in the future. Not used
anymore
25/02/04 01:16:58 WARN Instrumentation: [41373008] regParam is zero, which might
cause numerical instability and overfitting.
25/02/04 01:17:00 WARN InstanceBuilder: Failed to load implementation
from:dev.ludovic.netlib.blas.JNIBLAS
25/02/04 01:17:00 WARN InstanceBuilder: Failed to load implementation
from:dev.ludovic.netlib.lapack.JNILAPACK

Root Mean Squared Error (RMSE): 4.671806485171284
```

[5]:
```
# Step 8: Show the predictions
print("Predictions:")
predictions.select("features", "medv", "prediction").show(5)
```

```
# Step 9: Print model coefficients and intercept
print("Model Coefficients:")
for feature, coef in zip(feature_columns, lr_model.coefficients):
    print(f"{feature}: {coef}")
print(f"Intercept: {lr_model.intercept}")

# Step 10: Stop the SparkSession
spark.stop()
```

```
Predictions:

25/02/04 01:17:20 WARN SparkStringUtils: Truncated the string representation of
a plan since it was too large. This behavior can be adjusted by setting
'spark.sql.debug.maxToStringFields'.

+--------------------+----+------------------+
|            features|medv|        prediction|
+--------------------+----+------------------+
|[0.01096,55.0,2.2…|22.0| 27.48227401818613|
|[0.01381,80.0,0.4…|50.0| 40.59821928572494|
|[0.01439,60.0,2.9…|29.1|31.560171030407233|
|[0.01778,95.0,1.4…|32.9|30.504107540914198|
|[0.02177,82.5,2.0…|42.3| 36.71084264945604|
+--------------------+----+------------------+
only showing top 5 rows

Model Coefficients:
crim: -0.11362203729408954
zn: 0.048909186934053925
indus: 0.02379542898673389
chas: 2.801771998735119
nox: -18.4154245411894
rm: 3.5158797633120065
age: 0.0052116821614709204
dis: -1.4163830723539739
rad: 0.3317669315937035
tax: -0.013607893704163878
ptratio: -0.9534143338408072
b: 0.008602677392853256
lstat: -0.519503531247664
Intercept: 38.61699144573437
```

[ ]: