week-9(Linear Regression)

to import libraries:-

```
scala> import org.apache.spark.mllib.regression.LabeledPoint
import org.apache.spark.mllib.linalg.Vectors
import org.apache.spark.mllib.regression.LinearRegressionWithSGD
import org.apache.spark.mllib.evaluation.RegressionMetrics
import org.apache.spark.rdd.RDD
import org.apache.spark.mllib.regression.LabeledPoint
import org.apache.spark.mllib.linalg.Vectors
import org.apache.spark.mllib.regression.LinearRegressionWithSGD
import org.apache.spark.mllib.evaluation.RegressionMetrics
import org.apache.spark.rdd.RDD
```

to insert values:-

```
scala> val data = sc.parallelize(Seq(
  LabeledPoint(1.0, Vectors.dense(1.0, 2.0)),
  LabeledPoint(2.0, Vectors.dense(1.0, 3.0)),
  LabeledPoint(3.0, Vectors.dense(2.0, 3.0)),
  LabeledPoint(4.0, Vectors.dense(2.0, 4.0)),
  LabeledPoint(5.0, Vectors.dense(3.0, 4.0)),
  LabeledPoint(6.0, Vectors.dense(3.0, 5.0))
))
```

output:-

```
data: org.apache.spark.rdd.RDD[org.apache.spark.mllib.regression.LabeledPoint] = Paral
```

code:-

```
scala> val Array(trainingData, testData) = data.randomSplit(Array(0.7, 0.3))
```

output:-

```
trainingData: org.apache.spark.rdd.RDD[org.apache.spark.mllib.regression.LabeledPoint]
testData: org.apache.spark.rdd.RDD[org.apache.spark.mllib.regression.LabeledPoint] = M
```

code:-

```
scala> val numIterations = 100  // Number of iterations for training
val model = LinearRegressionWithSGD.train(trainingData, numIterations)
```

ouput:-

```
24/12/02 01:19:40 WARN regression.LinearRegressionWithSGD: The input data is not directly cached, whi
24/12/02 01:19:42 WARN regression.LinearRegressionWithSGD: The input data was not directly cached, wh
numIterations: Int = 100
model: org.apache.spark.mllib.regression.LinearRegressionModel = org.apache.spark.mllib.regression.Li
```

code:-

```
scala> val predictions = testData.map { point =>
  val prediction = model.predict(point.features)
  (point.label, prediction)
}
```

output:-

```
predictions: org.apache.spark.rdd.RDD[(Double, Double)] = MapPartitionsRDD[409]
```

code:-

```
scala> val metrics = new RegressionMetrics(predictions)

val rmse = metrics.rootMeanSquaredError
val mse = metrics.meanSquaredError
val r2 = metrics.r2
```

output:-

```
metrics: org.apache.spark.mllib.evaluation.RegressionMetrics = org.apache.spark.mllib
rmse: Double = 8889546.680100044
mse: Double = 7.902404017767773E13
r2: Double = -8.081210336353292
```

code:-

```
scala> println(s"Root Mean Squared Error (RMSE) = $rmse")
println(s"Mean Squared Error (MSE) = $mse")
println(s"R-squared = $r2")
```

output:-

```
Root Mean Squared Error (RMSE) = 8889546.680100044
Mean Squared Error (MSE) = 7.902404017767773E13
R-squared = -8.081210336353292
```