IFT 512 ADVANCED BIG DATA ANALYTICS/AI (2023 Fall)

INFORMATION TECHNOLOGY

A3.2SparkScala

Instructor: Dr. Robert Rucker

Aishwarya Devi Akkim

ASU ID: 1226589665

03 September 2023

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**Code:**

import org.apache.spark.sql.SparkSession

import org.apache.spark.sql.functions.\_

import org.apache.spark.sql.types.\_

val CSVFileStoreLocation = "/FileStore/tables/Trainingsetdata-5.csv"

val sparkSessionBuilder = SparkSession.builder().appName("RegressionAnalysis").getOrCreate()

val sparkSessionDF = sparkSessionBuilder.read.option("header", "true").csv(CSVFileStoreLocation)

val typedDF = sparkSessionDF.select(col("age").cast("double"), col("yearly visits to parks").cast("double"))

// Calculating the mean for the age and visits

val ageOfMean = typedDF.select(avg("age")).first()(0).asInstanceOf[Double]

val visitsOfMean = typedDF.select(avg("yearly visits to parks")).first()(0).asInstanceOf[Double]

// Calculating the regression coefficient and intercept

val numerator = typedDF.rdd.map { case row =>

val xi = row.getDouble(0)

val yi = row.getDouble(1)

(xi - ageOfMean) \* (yi - visitsOfMean)

}.sum()

val denominator = typedDF.rdd.map { case row =>

val xi = row.getDouble(0)

(xi - ageOfMean) \* (xi - ageOfMean)

}.sum()

val reg\_coffecient\_slope = numerator / denominator

val intercept = visitsOfMean - reg\_coffecient\_slope \* ageOfMean

// Calculateing SST, SSR, SSE

val sst = typedDF.rdd.map { case row =>

val yi = row.getDouble(1)

(yi - visitsOfMean) \* (yi - visitsOfMean)

}.sum()

val ssr = typedDF.rdd.map { case row =>

val xi = row.getDouble(0)

val yi = row.getDouble(1)

(xi - ageOfMean) \* (yi - visitsOfMean)

}.sum()

val sse = sst - ssr

// Calculating the correlation coefficient(r)

val cor\_cfct\_r = ssr / math.sqrt(sst \* sse)

// Calculating the R squared(r^2)

val rSqr = cor\_cfct\_r \* cor\_cfct\_r

// Calculating the angle between X and Y

val angleInRds = math.atan(reg\_coffecient\_slope)

println(s"Calculated Regression Coefficient(Slope): $reg\_coffecient\_slope")

println(s"Calculated Intercept: $intercept")

println(s"Calculated Total Sum of Squares (SST): $sst")

println(s"Calculated Regression Sum of Squares (SSR): $ssr")

println(s"Calculated Residual Sum of Squares (SSE): $sse")

println(s"Calculated Correlation Coefficient (r): $cor\_cfct\_r")

println(s"Calculated R-squared (r^2): $rSqr")

println(s"Calculated Angle between X and Y (in radians): $angleInRds")

**Output:**

Calculated Regression Coefficient(Slope): -0.47368421052631576

Calculated Intercept: 6.068421052631579

Calculated Total Sum of Squares (SST): 22.1

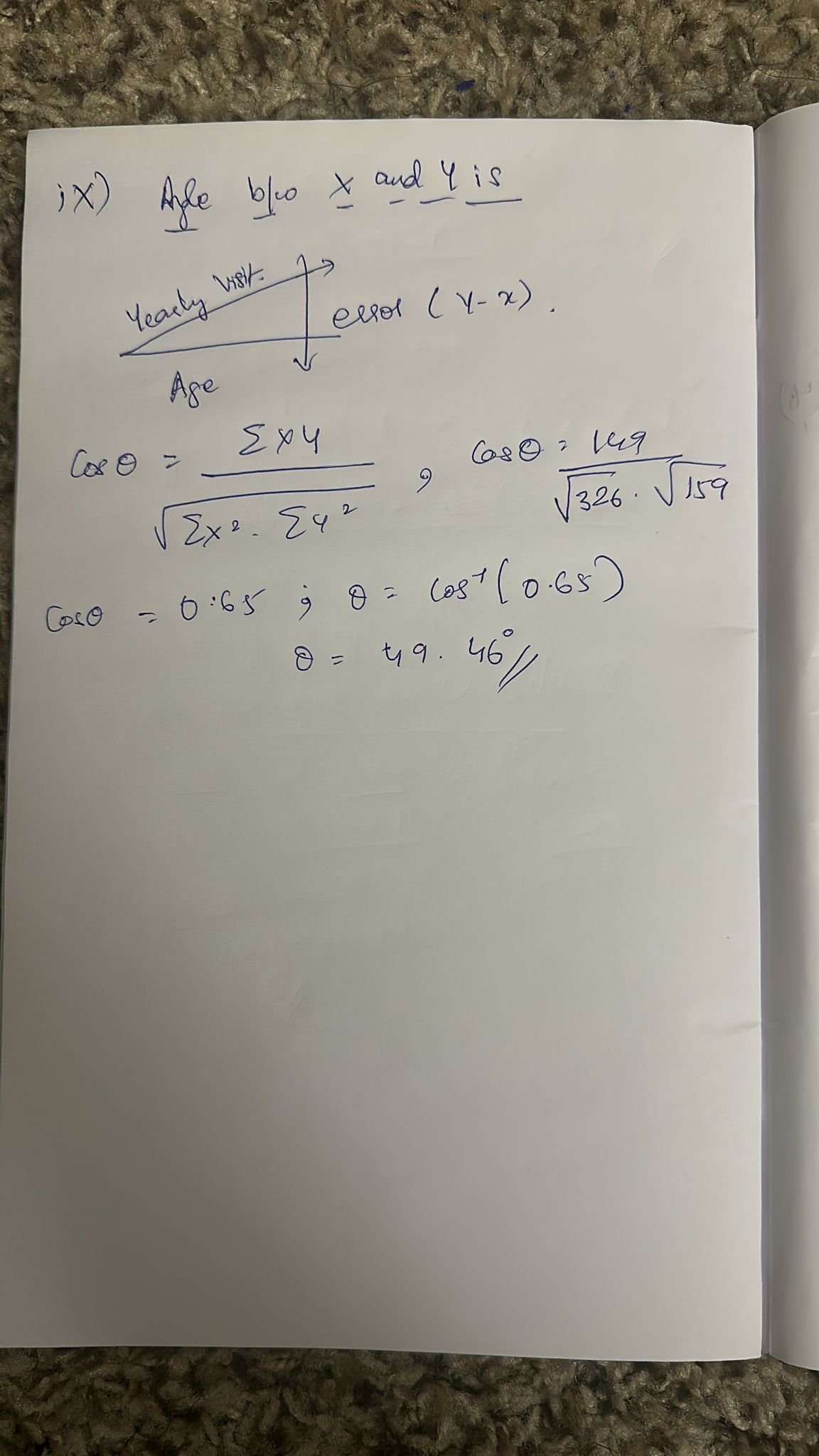
Calculated Regression Sum of Squares (SSR): -36.0

Calculated Residual Sum of Squares (SSE): 58.1

Calculated Correlation Coefficient (r): -1.0046581177222382

Calculated R-squared (r^2): 1.0093379335051906

Calculated Angle between X and Y (in radians): -0.4423742229767449



**Part II**

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**Code:**

import org.apache.spark.sql.SparkSession

import org.apache.spark.sql.functions.\_

import org.apache.spark.ml.feature.VectorAssembler

import org.apache.spark.ml.regression.LinearRegression

// Creating a Spark session

val sparkSessionBuilder = SparkSession.builder()

.appName("Analysis")

.getOrCreate()

// 1. Read in your file (which will give you a DataFrame (DF).

val CSVFileStoreLocation = "/FileStore/tables/Trainingsetdata-5.csv"

val sparkSessionDF = sparkSessionBuilder .read.option("header", "true").csv(CSVFileStoreLocation )

import spark.implicits.\_

val sparkSessionDataSet = sparkSessionDF .select(col("age").cast("double"), col("yearly visits to parks").cast("double")).as[(Double, Double)]

// 3. Construct a Vector Assembler to convert the "age" column to a features vector manually

val vectorAssembler = new VectorAssembler().setInputCols(Array("age")).setOutputCol("features")

val assembledDataset = vectorAssembler.transform(sparkSessionDataSet).select("features", "yearly visits to parks")

// 4. Make sure the "visits" column is called "label" (that is, get your DS ready for regression)!

val labeledDataset = assembledDataset.withColumnRenamed("yearly visits to parks", "label")

// 5. Do a Spark regression

val linearSparkRegression = new LinearRegression()

val sparkRegressionModel = linearSparkRegression.fit(labeledDataset)

val slope = sparkRegressionModel.coefficients

val intercept = sparkRegressionModel.intercept

val RSquared = sparkRegressionModel.summary.r2

// Output the model coefficients

println(s"Calculated Regression Coefficients (Slope): $slope")

println(s"Calculated Intercept: $intercept")

println(s" Calculated Rsquared (R^2): $RSquared")

**Output:**

Calculated Regression Coefficients (Slope): [-0.473684210526316]

Calculated Intercept: 6.068421052631581

Calculated Rsquared (R^2): 0.7716122886401524