In [1]:
```
import findspark
findspark.init()
```

In [2]:
```
from pyspark.sql import SparkSession
spark=SparkSession.builder.appName("DfApp").getOrCreate()
```

In [6]:
```
df=spark.read.option("header","true").csv('D:\\tips.csv',inferSchema=True)
df.show()
```

```
+----------+----+------+------+---+------+----+
|total_bill| tip|   sex|smoker|day|  time|size|
+----------+----+------+------+---+------+----+
|     16.99|1.01|Female|    No|Sun|Dinner|   2|
|     10.34|1.66|  Male|    No|Sun|Dinner|   3|
|     21.01| 3.5|  Male|    No|Sun|Dinner|   3|
|     23.68|3.31|  Male|    No|Sun|Dinner|   2|
|     24.59|3.61|Female|    No|Sun|Dinner|   4|
|     25.29|4.71|  Male|    No|Sun|Dinner|   4|
|      8.77| 2.0|  Male|    No|Sun|Dinner|   2|
|     26.88|3.12|  Male|    No|Sun|Dinner|   4|
|     15.04|1.96|  Male|    No|Sun|Dinner|   2|
|     14.78|3.23|  Male|    No|Sun|Dinner|   2|
|     10.27|1.71|  Male|    No|Sun|Dinner|   2|
|     35.26| 5.0|Female|    No|Sun|Dinner|   4|
|     15.42|1.57|  Male|    No|Sun|Dinner|   2|
|     18.43| 3.0|  Male|    No|Sun|Dinner|   4|
|     14.83|3.02|Female|    No|Sun|Dinner|   2|
|     21.58|3.92|  Male|    No|Sun|Dinner|   2|
|     10.33|1.67|Female|    No|Sun|Dinner|   3|
|     16.29|3.71|  Male|    No|Sun|Dinner|   3|
|     16.97| 3.5|Female|    No|Sun|Dinner|   3|
|     20.65|3.35|  Male|    No|Sat|Dinner|   3|
+----------+----+------+------+---+------+----+
only showing top 20 rows
```

In [7]:
```
df.printSchema()
```

```
root
 |-- total_bill: double (nullable = true)
 |-- tip: double (nullable = true)
 |-- sex: string (nullable = true)
 |-- smoker: string (nullable = true)
 |-- day: string (nullable = true)
 |-- time: string (nullable = true)
 |-- size: integer (nullable = true)
```

In [8]:
```
df.count()
```

Out[8]:
```
244
```

In [9]:
```
df.na.drop(how='any')
```

Out[9]:
```
DataFrame[total_bill: double, tip: double, sex: string, smoker: string, day: string, time: string, size: int]
```

In [10]:
```
df.count()
```

Out[10]:
```
244
```

In [11]:
```
from pyspark.ml.feature import VectorAssembler
```

In [12]:
```
featureassembler=VectorAssembler(inputCols=['tip'],outputCol='TipData')
v=featureassembler.transform(df)
v.show()
```

```
+----------+----+------+------+---+------+----+-------+
|total_bill| tip|   sex|smoker|day|  time|size|TipData|
+----------+----+------+------+---+------+----+-------+
|     16.99|1.01|Female|    No|Sun|Dinner|   2| [1.01]|
|     10.34|1.66|  Male|    No|Sun|Dinner|   3| [1.66]|
|     21.01| 3.5|  Male|    No|Sun|Dinner|   3|  [3.5]|
|     23.68|3.31|  Male|    No|Sun|Dinner|   2| [3.31]|
|     24.59|3.61|Female|    No|Sun|Dinner|   4| [3.61]|
|     25.29|4.71|  Male|    No|Sun|Dinner|   4| [4.71]|
|      8.77| 2.0|  Male|    No|Sun|Dinner|   2|  [2.0]|
|     26.88|3.12|  Male|    No|Sun|Dinner|   4| [3.12]|
|     15.04|1.96|  Male|    No|Sun|Dinner|   2| [1.96]|
|     14.78|3.23|  Male|    No|Sun|Dinner|   2| [3.23]|
|     10.27|1.71|  Male|    No|Sun|Dinner|   2| [1.71]|
|     35.26| 5.0|Female|    No|Sun|Dinner|   4|  [5.0]|
|     15.42|1.57|  Male|    No|Sun|Dinner|   2| [1.57]|
|     18.43| 3.0|  Male|    No|Sun|Dinner|   4|  [3.0]|
|     14.83|3.02|Female|    No|Sun|Dinner|   2| [3.02]|
|     21.58|3.92|  Male|    No|Sun|Dinner|   2| [3.92]|
|     10.33|1.67|Female|    No|Sun|Dinner|   3| [1.67]|
|     16.29|3.71|  Male|    No|Sun|Dinner|   3| [3.71]|
|     16.97| 3.5|Female|    No|Sun|Dinner|   3|  [3.5]|
|     20.65|3.35|  Male|    No|Sat|Dinner|   3| [3.35]|
+----------+----+------+------+---+------+----+-------+
only showing top 20 rows
```

In [18]:
```
fd=v.select('TipData','total_bill')
fd.show()
```

```
+-------+----------+
|TipData|total_bill|
+-------+----------+
| [1.01]|     16.99|
| [1.66]|     10.34|
|  [3.5]|     21.01|
| [3.31]|     23.68|
| [3.61]|     24.59|
| [4.71]|     25.29|
|  [2.0]|      8.77|
| [3.12]|     26.88|
| [1.96]|     15.04|
| [3.23]|     14.78|
| [1.71]|     10.27|
|  [5.0]|     35.26|
| [1.57]|     15.42|
|  [3.0]|     18.43|
| [3.02]|     14.83|
| [3.92]|     21.58|
| [1.67]|     10.33|
| [3.71]|     16.29|
|  [3.5]|     16.97|
| [3.35]|     20.65|
+-------+----------+
only showing top 20 rows
```

In [19]:
```
v.printSchema()
```

```
root
 |-- total_bill: double (nullable = true)
 |-- tip: double (nullable = true)
 |-- sex: string (nullable = true)
 |-- smoker: string (nullable = true)
 |-- day: string (nullable = true)
 |-- time: string (nullable = true)
 |-- size: integer (nullable = true)
 |-- TipData: vector (nullable = true)
```

In [20]:
```
from pyspark.ml.regression import LinearRegression
```

In [21]:
```
train,test=fd.randomSplit([.70,.30])
lr=LinearRegression(featuresCol='TipData',labelCol='total_bill')
tm=lr.fit(train)
```

In [22]:
```
tm.intercept
```

Out[22]:
```
7.122562688266986
```

In [23]:
```
tm.coefficients
```

Out[23]:
```
DenseVector([4.2821])
```

In [24]:
```
res=tm.evaluate(test)
res.predictions.show()
```

```
C:\Users\user\anaconda3\lib\site-packages\pyspark\sql\context.py:125: FutureWarning: Deprecated in 3.0.0. Use SparkSession.builder.getOrCreate() instead.
  warnings.warn(
+-------+----------+------------------+
|TipData|total_bill|        prediction|
+-------+----------+------------------+
|  [1.1]|      12.9|11.832862119619367|
| [1.25]|     10.07|12.475175678440145|
| [1.25]|     10.51|12.475175678440145|
| [1.36]|     18.64|12.946205621575384|
| [1.45]|      9.55| 13.33159375686785|
| [1.48]|      8.52|13.460056468632006|
|  [1.5]|     15.69|13.545698276474777|
|  [1.5]|     26.41|13.545698276474777|
| [1.56]|      9.94|13.802623700003089|
| [1.57]|     15.42|13.845444603924474|
| [1.68]|     13.42|14.316474547059713|
| [1.75]|     17.82|14.616220874509409|
| [1.92]|      8.58|15.344176241172956|
| [1.96]|     15.04|15.515459856858499|
|  [2.0]|     10.09|15.686743472544041|
|  [2.0]|     10.34|15.686743472544041|
|  [2.0]|     10.63|15.686743472544041|
|  [2.0]|     11.38|15.686743472544041|
|  [2.0]|     12.26|15.686743472544041|
|  [2.0]|     13.51|15.686743472544041|
+-------+----------+------------------+
only showing top 20 rows
```

In [25]:
```
print("R2",res.r2)
print("Mean Absolute Error is",res.meanAbsoluteError)
print("Root Mean Square Error(RMSE)",res. rootMeanSquaredError)
```

```
R2 0.45728446679935075
Mean Absolute Error is 4.676778193595545
Root Mean Square Error(RMSE) 6.627871945615341
```

In [ ]:
```

```