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

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

In [3]:
```
df=spark.read.option("header","true").csv('D:\weight-height.csv',inferSchema=True)
df.show()
```

```
+------+-----------+-----------+
|Gender|     Height|     Weight|
+------+-----------+-----------+
|  Male|73.84701702|241.8935632|
|  Male|68.78190405|162.3104725|
|  Male|74.11010539|212.7408556|
|  Male| 71.7309784|220.0424703|
|  Male|69.88179586|206.3498006|
|  Male|67.25301569|152.2121558|
|  Male|  65.865443|183.9278886|
|Female|68.34851551|167.9711105|
|  Male|67.01894966|175.9294404|
|  Male|  56.234555|156.3996764|
|  Male|71.19538228|186.6049256|
|  Male|71.64080512|213.7411695|
|  Male|64.76632913|167.1274611|
|  Male| 65.3456677|189.4461814|
|Female|69.24373223| 186.434168|
|  Male| 67.6456197|172.1869301|
|  Male|72.41831663|196.0285063|
|  Male|63.97432572|172.8834702|
|Female| 69.6400599|185.9839576|
|  Male|  65.654333| 182.426648|
+------+-----------+-----------+
only showing top 20 rows
```

In [4]:
```
df.printSchema()
```

```
root
 |-- Gender: string (nullable = true)
 |-- Height: double (nullable = true)
 |-- Weight: double (nullable = true)
```

In [5]:
```
df.count()
```

Out[5]:
```
10000
```

In [6]:
```
df.na.drop(how='any')
```

Out[6]:
```
DataFrame[Gender: string, Height: double, Weight: double]
```

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

Out[7]:
```
10000
```

In [8]:
```
from pyspark.ml.feature import VectorAssembler
```

In [9]:
```
featureassembler=VectorAssembler(inputCols=['Height'],outputCol='HeightData')
v=featureassembler.transform(df)
v.show()
```

```
+------+-----------+-----------+------------+
|Gender|     Height|     Weight|  HeightData|
+------+-----------+-----------+------------+
|  Male|73.84701702|241.8935632|[73.84701702]|
|  Male|68.78190405|162.3104725|[68.78190405]|
|  Male|74.11010539|212.7408556|[74.11010539]|
|  Male| 71.7309784|220.0424703| [71.7309784]|
|  Male|69.88179586|206.3498006|[69.88179586]|
|  Male|67.25301569|152.2121558|[67.25301569]|
|  Male|  65.865443|183.9278886|  [65.865443]|
|Female|68.34851551|167.9711105|[68.34851551]|
|  Male|67.01894966|175.9294404|[67.01894966]|
|  Male|  56.234555|156.3996764|  [56.234555]|
|  Male|71.19538228|186.6049256|[71.19538228]|
|  Male|71.64080512|213.7411695|[71.64080512]|
|  Male|64.76632913|167.1274611|[64.76632913]|
|  Male| 65.3456677|189.4461814| [65.3456677]|
|Female|69.24373223| 186.434168|[69.24373223]|
|  Male| 67.6456197|172.1869301| [67.6456197]|
|  Male|72.41831663|196.0285063|[72.41831663]|
|  Male|63.97432572|172.8834702|[63.97432572]|
|Female| 69.6400599|185.9839576| [69.6400599]|
|  Male|  65.654333| 182.426648|  [65.654333]|
+------+-----------+-----------+------------+
only showing top 20 rows
```

In [10]:
```
v.select('HeightData','Weight').show()
```

```
+------------+-----------+
|  HeightData|     Weight|
+------------+-----------+
|[73.84701702]|241.8935632|
|[68.78190405]|162.3104725|
|[74.11010539]|212.7408556|
| [71.7309784]|220.0424703|
|[69.88179586]|206.3498006|
|[67.25301569]|152.2121558|
|  [65.865443]|183.9278886|
|[68.34851551]|167.9711105|
|[67.01894966]|175.9294404|
|  [56.234555]|156.3996764|
|[71.19538228]|186.6049256|
|[71.64080512]|213.7411695|
|[64.76632913]|167.1274611|
| [65.3456677]|189.4461814|
|[69.24373223]| 186.434168|
| [67.6456197]|172.1869301|
|[72.41831663]|196.0285063|
|[63.97432572]|172.8834702|
| [69.6400599]|185.9839576|
|  [65.654333]| 182.426648|
+------------+-----------+
only showing top 20 rows
```

In [11]:
```
v.printSchema()
```

```
root
 |-- Gender: string (nullable = true)
 |-- Height: double (nullable = true)
 |-- Weight: double (nullable = true)
 |-- HeightData: vector (nullable = true)
```

In [12]:
```
fd=v.select('HeightData','Weight')
fd.show()
```

```
+------------+-----------+
|  HeightData|     Weight|
+------------+-----------+
|[73.84701702]|241.8935632|
|[68.78190405]|162.3104725|
|[74.11010539]|212.7408556|
| [71.7309784]|220.0424703|
|[69.88179586]|206.3498006|
|[67.25301569]|152.2121558|
|  [65.865443]|183.9278886|
|[68.34851551]|167.9711105|
|[67.01894966]|175.9294404|
|  [56.234555]|156.3996764|
|[71.19538228]|186.6049256|
|[71.64080512]|213.7411695|
|[64.76632913]|167.1274611|
| [65.3456677]|189.4461814|
|[69.24373223]| 186.434168|
| [67.6456197]|172.1869301|
|[72.41831663]|196.0285063|
|[63.97432572]|172.8834702|
| [69.6400599]|185.9839576|
|  [65.654333]| 182.426648|
+------------+-----------+
only showing top 20 rows
```

In [13]:
```
# spliting of data into train and test data
from pyspark.ml.regression import LinearRegression
```

In [14]:
```
train,test=fd.randomSplit([.70,.30])
lr=LinearRegression(featuresCol='HeightData',labelCol='Weight')
tm=lr.fit(train)
```

In [15]:
```
# coefficient and intercept
tm.intercept
```

Out[15]:
```
-345.5104174679869
```

In [16]:
```
tm.coefficients  # correlation bet weight and height
```

Out[16]:
```
DenseVector([7.6367])
```

In [17]:
```
# evaluate the train data
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(
+------------+-----------+-----------------+
|  HeightData|     Weight|       prediction|
+------------+-----------+-----------------+
|[55.97919788]|85.41753362|81.98789608042966|
|[56.06663635]|89.57120474|82.65564052622159|
|[56.10536959]|87.29886913|82.95143602535967|
|[56.63041198]|89.48048027|86.96104575123644|
|  [56.765443]|201.7607769|87.99224190110743|
|  [56.765544]|166.5035316|87.99301321136738|
|[56.78543437]|83.99307747|88.14491070102207|
|[56.78938641]|95.32808768|88.17509138418757|
|[56.85608213]|97.36497833|88.68442893993495|
|[56.94941514]|107.1718559| 89.3971884271096|
|[56.97513323]|89.16984997|89.59359067149785|
|[56.97527896]|90.34178426|89.59470352792543|
|[57.02885744]|101.2025509|90.00386823962157|
|[57.14819808]|91.64547294| 90.9152411115079|
| [57.2330564]|99.37128426|91.56328163485227|
|[57.27014705]|94.49963415|91.84653310904099|
|[57.31302352]| 93.8764374| 92.1739693587565|
|[57.31390274]| 95.1390468| 92.1806837291204|
|[57.35309776]|72.75014469|92.47996753618463|
|[57.37575853]|114.1922086|92.65306002084378|
+------------+-----------+-----------------+
only showing top 20 rows
```

In [18]:
```
print("R2",res.r2)
print("Mean Absolute Error is",res.meanAbsoluteError)
print("Root Mean Square Error(RMSE)",res. rootMeanSquaredError)
```

```
R2 0.8490377509012588
Mean Absolute Error is 9.752730559572562
Root Mean Square Error(RMSE) 12.509674871693
```

In [ ]:
```

```