Chapter 6: Linear Regression

Demo

Basically what we do here is examine a dataset with Ecommerce Customer Data for a company's website and mobile app. Then we want to see if we can build a regression model that will predict the customer's yearly spend on the company's product.

First thing to do is start a Spark Session

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

In [2]:
```
import pyspark
```

In [3]:
```
from pyspark import SparkContext
from pyspark.conf import SparkConf
from pyspark.sql import SparkSession
```

In [4]:
```
spark = SparkSession.builder.appName('lr_example').getOrCreate()
```

In [5]:
```
from pyspark.ml.regression import LinearRegression
```

In [6]:
```
# Use Spark to read in the Ecommerce Customers csv file.
data = spark.read.csv("Ecommerce_Customers.csv",inferSchema=True,header=True)
```

In [7]:
```
# Print the Schema of the DataFrame
data.printSchema()
```

```
root
 |-- Email: string (nullable = true)
 |-- Address: string (nullable = true)
 |-- Avatar: string (nullable = true)
 |-- Avg Session Length: double (nullable = true)
 |-- Time on App: double (nullable = true)
 |-- Time on Website: double (nullable = true)
 |-- Length of Membership: double (nullable = true)
 |-- Yearly Amount Spent: double (nullable = true)
```

In [8]:
```
data.show(5)
```

```
+--------------------+--------------------+---------------+------------------+
------------------+------------------+--------------------+-------------------+
|               Email|             Address|         Avatar|Avg Session Length|
Time on App|   Time on Website|Length of Membership|Yearly Amount Spent|
+--------------------+--------------------+---------------+------------------+
------------------+------------------+--------------------+-------------------+
|mstephenson@ferna...|835 Frank TunnelW...|         Violet| 34.49726772511229|
12.65565114916675| 39.57766801952616|  4.0826206329529615|  587.9510539684005|
|   hduke@hotmail.com|4547 Archer Commo...|       DarkGreen| 31.92627202636016|
11.109460728682564|37.268958868297744|    2.66403418213262|  392.2049334443264|
|    pallen@yahoo.com|24645 Valerie Uni...|         Bisque|33.000914755642675|
11.330278057777512|37.110597442120856|   4.104543202376424| 487.54750486747207|
|riverarebecca@gma...|1414 David Throug...|     SaddleBrown| 34.30555662975554|
13.717513665142507| 36.72128267790313|   3.120178782748092|  581.8523440352177|
|mstephens@davidso...|14023 Rodriguez P...|MediumAquaMarine| 33.33067252364639|
12.795188551078114| 37.53665330059473|   4.446308318351434|  599.4060920457634|
+--------------------+--------------------+---------------+------------------+
------------------+------------------+--------------------+-------------------+
only showing top 5 rows
```

In [9]:
```
data.head()
```

Out[9]:
```
Row(Email='mstephenson@fernandez.com', Address='835 Frank TunnelWrightmouth, MI
82180-9605', Avatar='Violet', Avg Session Length=34.49726772511229, Time on App
=12.65565114916675, Time on Website=39.57766801952616, Length of Membership=4.0
826206329529615, Yearly Amount Spent=587.9510539684005)
```

In [10]:
```
for item in data.head():
    print(item)
```

```
mstephenson@fernandez.com
835 Frank TunnelWrightmouth, MI 82180-9605
Violet
34.49726772511229
12.65565114916675
39.57766801952616
4.0826206329529615
587.9510539684005
```

Setting Up DataFrame for Machine Learning

In [11]:
```
# It needs to be in the form of two columns
# ("label","features")

# Import VectorAssembler and Vectors
from pyspark.ml.linalg import Vectors
from pyspark.ml.feature import VectorAssembler
```

In [12]:
```
data.columns
```

Out[12]:
```
['Email',
 'Address',
 'Avatar',
 'Avg Session Length',
 'Time on App',
 'Time on Website',
 'Length of Membership',
 'Yearly Amount Spent']
```

In [13]:
```
assembler = VectorAssembler(
    inputCols=["Avg Session Length", "Time on App",
               "Time on Website",'Length of Membership'],
    outputCol="features") # inputs
```

In [14]:
```
data_pre = assembler.transform(data)
```

In [15]:
```
data_pre.select("features").show(2, False)
```

```
+-----------------------------------------------------------------------+
|features                                                               |
+-----------------------------------------------------------------------+
|[34.49726772511229,12.65565114916675,39.57766801952616,4.0826206329529615]|
|[31.92627202636016,11.109460728682564,37.268958868297744,2.66403418213262]|
+-----------------------------------------------------------------------+
only showing top 2 rows
```

In [16]:
```
data_pre.show(2)
```

```
+--------------------+--------------------+---------+------------------+-------
-----------+------------------+--------------------+------------------+-------
-------------+
|               Email|             Address|   Avatar|Avg Session Length|
Time on App|   Time on Website|Length of Membership|Yearly Amount Spent|
features|
+--------------------+--------------------+---------+------------------+-------
-----------+------------------+--------------------+------------------+-------
-------------+
|mstephenson@ferna...|835 Frank TunnelW...|   Violet| 34.49726772511229| 12.655
65114916675| 39.57766801952616|  4.0826206329529615|  587.9510539684005|[34.497
2677251122...|
|   hduke@hotmail.com|4547 Archer Commo...|DarkGreen| 31.92627202636016|11.1094
60728682564|37.268958868297744|    2.66403418213262|  392.2049334443264|[31.926
2720263601...|
+--------------------+--------------------+---------+------------------+-------
-----------+------------------+--------------------+------------------+-------
-------------+
only showing top 2 rows
```

In [17]:
```
final_data = data_pre.select("features",'Yearly Amount Spent')
```

In [18]:
```
train_data,test_data = final_data.randomSplit([0.7,0.3])
```

In [19]:
```
train_data.describe().show()
```

```
+-------+------------------+
|summary|Yearly Amount Spent|
+-------+------------------+
|  count|               343|
|   mean|  500.9283755502173|
| stddev|  83.87024231143093|
|    min| 256.67058229005585|
|    max|  765.5184619388373|
+-------+------------------+
```

In [20]:
```
test_data.describe().show()
```

```
+-------+------------------+
|summary|Yearly Amount Spent|
+-------+------------------+
|  count|               157|
|   mean| 495.78717398452744|
| stddev|  68.43381964871429|
|    min| 302.18954780965197|
|    max|  725.5848140556806|
+-------+------------------+
```

In [21]:
```
# Create a Linear Regression Model object
lr = LinearRegression(featuresCol="features",
                      labelCol='Yearly Amount Spent',
                      predictionCol='Predict_Yearly Amount Spent')
```

In [22]:
```
# Fit the model to the data and call this model lrModel
lrModel = lr.fit(train_data,)
```

In [23]:
```
# Print the coefficients and intercept for linear regression
print("Coefficients: {} Intercept: {}".format(lrModel.coefficients,
                                              lrModel.intercept))
```

```
Coefficients: [25.919298522549543,39.05259649768098,0.7963446754170292,61.58826
630305803] Intercept: -1075.2099532383284
```

In [24]:
```
test_results = lrModel.evaluate(test_data)
```

In [25]:
```
# Interesting results....
test_results.residuals.show(5)
```

```
+-------------------+
|          residuals|
+-------------------+
|-10.826809207839347|
|  0.808945904875543|
| 4.5681283726872834|
|  9.952012731730065|
|-3.0124996202594048|
+-------------------+
only showing top 5 rows
```

In [26]:
```
# Check test dataset
test_model = lrModel.transform(test_data)
```

In [27]:
```
# Inspect results
test_model.select("Predict_Yearly Amount Spent",
                  "Yearly Amount Spent").show(5)
```

```
+--------------------------+-------------------+
|Predict_Yearly Amount Spent|Yearly Amount Spent|
+--------------------------+-------------------+
|         330.75567901103295|  319.9288698031936|
|          441.2554678531901| 442.06441375806565|
|          387.9292708163341|  392.4973991890214|
|         417.40451807056274|  427.3565308022928|
|         426.48303279408333|  423.4705331738239|
+--------------------------+-------------------+
only showing top 5 rows
```

In [28]:
```
print("RMSE: {}".format(test_results.rootMeanSquaredError))
print("MSE: {}".format(test_results.meanSquaredError))
print("r2: {}".format(test_results.r2))
```

```
RMSE: 10.080953257096533
MSE: 101.6256185717652
r2: 0.9781608015705986
```

Excellent results!

In [30]:
```
# Save model
lrModel.save('lrModel_Ecommerce_Customers')
```

In [ ]:
```
from pyspark.ml.regression import LinearRegressionModel
# Load model from
lrModel2 = LinearRegressionModel.load('lrModel_Ecommerce_Customers')
```

In [ ]:
```
# Predict new values (Assuming select test_data)
unlabeled_data = test_data.select('features')
```

In [ ]:
```
predictions = lrModel2.transform(unlabeled_data)
```

In [ ]:
```
predictions.show(5)
```