Analysis of Covid 19 using Apache Spark

Importing Libraries

In [1]:

```
from pyspark.sql import SparkSession
from pyspark.sql.types import StructField,IntegerType,StructType,StringType
import pyspark.sql.functions as f
from pyspark.ml.regression import LinearRegression
from pyspark.ml.linalg import Vectors
from pyspark.ml.feature import VectorAssembler
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.graph_objects as go
import plotly.express as px
from plotly.subplots import make_subplots
from pyspark.ml.classification import DecisionTreeClassifier, GBTClassifier, RandomForestClassifier
from pyspark.ml.regression import RandomForestRegressor, DecisionTreeRegressor, GBTRegressor
from pyspark.ml import Pipeline
from pyspark.ml.evaluation import RegressionEvaluator
from pyspark.ml.feature import HashingTF, Tokenizer
from pyspark.ml.tuning import ParamGridBuilder, TrainValidationSplit
from pyspark.ml.feature import VectorIndexer
```

Loading the dataset

In [2]:

```
spark = SparkSession.builder.appName("Project").getOrCreate()
```

In [3]:

```
df = spark.read.csv("country_wise_latest.csv", inferSchema =True, header=True)
data = pd.read_csv("country_wise_latest.csv", index_col=0)
world_data = pd.read_csv("worldometer_data.csv")
```

Data Exploration

In [4]:

```
df.toPandas().head()
```

Out[4]:

	Country/Region	Confirmed	Deaths	Recovered	Active	New cases	New deaths	New recovered	Deaths / 100 Cases	Recovered / 100 Cases	Deaths / 100 Recovered	Confirmed last week	1 week change	1 inc
0	Afghanistan	36263	1269	25198	9796	106	10	18	3.50	69.49	5.04	35526	737	
1	Albania	4880	144	2745	1991	117	6	63	2.95	56.25	5.25	4171	709	
2	Algeria	27973	1163	18837	7973	616	8	749	4.16	67.34	6.17	23691	4282	
3	Andorra	907	52	803	52	10	0	0	5.73	88.53	6.48	884	23	
4	Angola	950	41	242	667	18	1	0	4.32	25.47	16.94	749	201	

In [5]:

```
world_data.head()
```

Out[5]:

	Country/Region	Continent	Population	TotalCases	NewCases	TotalDeaths	NewDeaths	TotalRecovered	NewRecovered	ActiveCases
0	USA	North America	3.311981e+08	5032179	NaN	162804.0	NaN	2576668.0	NaN	2292707.0
1	Brazil	South America	2.127107e+08	2917562	NaN	98644.0	NaN	2047660.0	NaN	771258.0
2	India	Asia	1.381345e+09	2025409	NaN	41638.0	NaN	1377384.0	NaN	606387.0
3	Russia	Europe	1.459409e+08	871894	NaN	14606.0	NaN	676357.0	NaN	180931.0
4	South Africa	Africa	5.938157e+07	538184	NaN	9604.0	NaN	387316.0	NaN	141264.0

In [6]:

```
#Enables the user to check the column datatypes
df.printSchema()
```

```
root
 |-- Country/Region: string (nullable = true)
 |-- Confirmed: integer (nullable = true)
 |-- Deaths: integer (nullable = true)
 |-- Recovered: integer (nullable = true)
 |-- Active: integer (nullable = true)
 |-- New cases: integer (nullable = true)
 |-- New deaths: integer (nullable = true)
 |-- New recovered: integer (nullable = true)
 |-- Deaths / 100 Cases: double (nullable = true)
 |-- Recovered / 100 Cases: double (nullable = true)
 |-- Deaths / 100 Recovered: string (nullable = true)
 |-- Confirmed last week: integer (nullable = true)
 |-- 1 week change: integer (nullable = true)
 |-- 1 week % increase: double (nullable = true)
 |-- WHO Region: string (nullable = true)
```

In [7]:

```
#Counting the total number of rows and column for primary dataset
print((df.count(), len(df.columns)))
```

```
(187, 15)
```

In [8]:

```
#Printing the rows and column for secondary dataset
world_data.shape
```

Out[8]:

```
(209, 16)
```

Data Pre-processing

In [9]:

```
#Changing the column datatype from string to integer
df = df.withColumn("Deaths / 100 Recovered", df["Deaths / 100 Recovered"].cast(IntegerType()))
```

In [10]:

```
df.printSchema()
```

```
root
 |-- Country/Region: string (nullable = true)
 |-- Confirmed: integer (nullable = true)
 |-- Deaths: integer (nullable = true)
 |-- Recovered: integer (nullable = true)
 |-- Active: integer (nullable = true)
 |-- New cases: integer (nullable = true)
 |-- New deaths: integer (nullable = true)
 |-- New recovered: integer (nullable = true)
 |-- Deaths / 100 Cases: double (nullable = true)
 |-- Recovered / 100 Cases: double (nullable = true)
 |-- Deaths / 100 Recovered: integer (nullable = true)
 |-- Confirmed last week: integer (nullable = true)
 |-- 1 week change: integer (nullable = true)
 |-- 1 week % increase: double (nullable = true)
 |-- WHO Region: string (nullable = true)
```

In [11]:

```
df.describe().toPandas()
```

Out[11]:

	summary	Country/Region	Confirmed	Deaths	Recovered	Active	New cases	I
0	count	187	187	187	187	187	187	
1	mean	None	88130.935828877	3497.51871657754	50631.48128342246	34001.935828877	1222.957219251337	28.957
2	stddev	None	383318.6638306154	14100.00248201848	190188.18964313966	213326.17337142891	5710.374790280563	120.0371
3	min	Afghanistan	10	0	0	0	0	
4	max	Zimbabwe	4290259	148011	1846641	2816444	56336	

In [12]:

```
#Selecting the growth factor column to determine the increase of covid cases per week
df.select("1 week % increase").show()
```

```
+-----------------+
|1 week % increase|
+-----------------+
|             2.07|
|             17.0|
|            18.07|
|              2.6|
|            26.84|
|            13.16|
|            28.02|
|             6.89|
|            23.13|
|             4.13|
|             9.16|
|           119.54|
|             6.89|
|             9.05|
|             3.77|
|             1.57|
|             3.64|
|             20.0|
|            10.49|
|             10.0|
+-----------------+
only showing top 20 rows
```

In [13]:

```
type(df["1 week % increase"])
```

Out[13]:

```
pyspark.sql.column.Column
```

In [14]:

```
#Rename the column since it is considered as label
df = df.withColumnRenamed("1 week % increase", "growth_factor")
```

In [15]:

```
df.show()
```

```
+--------------------+---------+------+---------+------+---------+----------+-------------+-----------------+--------------------+---------------------+---------------
-----+-------------+-------------+--------------------+
|      Country/Region|Confirmed|Deaths|Recovered|Active|New cases|New deaths|New recovered|Deaths / 100 Cases|Recovered / 100 Cases|
Deaths / 100 Recovered|Confirmed last week|1 week change|growth_factor|          WHO Region|
+--------------------+---------+------+---------+------+---------+----------+-------------+-----------------+--------------------+---------------------+---------------
-----+-------------+-------------+--------------------+
|         Afghanistan|    36263|  1269|    25198|  9796|      106|        10|           18|              3.5|               69.49|                    5|            35526|
737|         2.07|Eastern Mediterra...|
|             Albania|     4880|   144|     2745|  1991|      117|         6|           63|             2.95|               56.25|                    5|             4171|
709|         17.0|              Europe|
|             Algeria|    27973|  1163|    18837|  7973|      616|         8|          749|             4.16|               67.34|                    6|            23691|
4282|        18.07|              Africa|
|             Andorra|      907|    52|      803|    52|       10|         0|            0|             5.73|               88.53|                    6|              884|          23|
2.6|              Europe|
|              Angola|      950|    41|      242|   667|       18|         1|            0|             4.32|               25.47|                   16|              749|         201|
26.84|              Africa|
|Antigua and Barbuda|       86|     3|       65|    18|        4|         0|            5|             3.49|               75.58|                    4|               76|          10
|        13.16|            Americas|
|           Argentina|   167416|  3059|    72575| 91782|     4890|       120|         2057|             1.83|               43.35|                    4|           13077
4|        36642|        28.02|            Americas|
|             Armenia|    37390|   711|    26665| 10014|       73|         6|          187|              1.9|               71.32|                    2|            34981|
2409|         6.89|              Europe|
|           Australia|    15303|   167|     9311|  5825|      368|         6|          137|             1.09|               60.84|                    1|            12428|
2875|        23.13|     Western Pacific|
|             Austria|    20558|   713|    18246|  1599|       86|         1|           37|             3.47|               88.75|                    3|            19743|
815|         4.13|              Europe|
|          Azerbaijan|    30446|   423|    23242|  6781|      396|         6|          558|             1.39|               76.34|                    1|            27890|
2556|         9.16|              Europe|
|             Bahamas|      382|    11|       91|   280|       40|         0|            0|             2.88|               23.82|                   12|              174|          20
8|       119.54|            Americas|
|             Bahrain|    39482|   141|    36110|  3231|      351|         1|          421|             0.36|               91.46|                    0|            36936|
2546|         6.89|Eastern Mediterra...|
|          Bangladesh|   226225|  2965|   125683| 97577|     2772|        37|         1801|             1.31|               55.56|                    2|           2074
53|        18772|         9.05|     South-East Asia|
|            Barbados|      110|     7|       94|     9|        0|         0|            0|             6.36|               85.45|                    7|              106|           4|
3.77|            Americas|
|             Belarus|    67251|   538|    60492|  6221|      119|         4|           67|              0.8|               89.95|                    0|            66213|
1038|         1.57|              Europe|
|             Belgium|    66428|  9822|    17452| 39154|      402|         1|           14|            14.79|               26.27|                   56|            64094|
2334|         3.64|              Europe|
|              Belize|       48|     2|       26|    20|        0|         0|            0|             4.17|               54.17|                    7|               40|           8|
20.0|            Americas|
|               Benin|     1770|    35|     1036|   699|        0|         0|            0|             1.98|               58.53|                    3|             1602|         168|
10.49|              Africa|
|              Bhutan|       99|     0|       86|    13|        4|         0|            1|              0.0|               86.87|                    0|               90|           9|
10.0|     South-East Asia|
+--------------------+---------+------+---------+------+---------+----------+-------------+-----------------+--------------------+---------------------+---------------
-----+-------------+-------------+--------------------+
only showing top 20 rows
```

In [16]:

```
#Fill all the null values present in the dataset with 0
df = df.na.fill("0", subset = ['New cases',
 'New deaths',
 'New recovered',
 'Deaths / 100 Cases',
 'Recovered / 100 Cases',
 'Deaths / 100 Recovered'])
```

In [17]:

```
df.select("growth_factor").show()
```

```
+-------------+
|growth_factor|
+-------------+
|         2.07|
|         17.0|
|        18.07|
|          2.6|
|        26.84|
|        13.16|
|        28.02|
|         6.89|
|        23.13|
|         4.13|
|         9.16|
|       119.54|
|         6.89|
|         9.05|
|         3.77|
|         1.57|
|         3.64|
|         20.0|
|        10.49|
|         10.0|
+-------------+
only showing top 20 rows
```

In [18]:

```
#Filtering out the column growth factor
df.filter("growth_factor == 0").select(["Country/Region","New cases","growth_factor" ]).show()
```

```
+--------------------+---------+-------------+
|      Country/Region|New cases|growth_factor|
+--------------------+---------+-------------+
|              Brunei|        0|          0.0|
|            Dominica|        0|          0.0|
|   Equatorial Guinea|        0|          0.0|
|                Fiji|        0|          0.0|
|             Grenada|        0|          0.0|
|            Holy See|        0|          0.0|
|       Liechtenstein|        0|          0.0|
|Saint Kitts and N...|        0|          0.0|
|          San Marino|        0|          0.0|
|            Tanzania|        0|          0.0|
|         Timor-Leste|        0|          0.0|
|      Western Sahara|        0|          0.0|
+--------------------+---------+-------------+
```

In [19]:

```
new_df = df.filter(df["growth_factor"] > 0).sort("growth_factor", ascending = True)
```

In [20]:

```
new_df.select("growth_factor").show()
```

```
+-------------+
|growth_factor|
+-------------+
|         0.13|
|         0.26|
|         0.29|
|         0.49|
|         0.64|
|         0.68|
|          0.7|
|         0.78|
|         0.79|
|         0.82|
|         1.08|
|         1.12|
|         1.18|
|         1.36|
|         1.45|
|         1.54|
|         1.57|
|          1.6|
|         1.73|
|         1.86|
+-------------+
only showing top 20 rows
```

In [21]:

```
new_df.describe().toPandas()
```

Out[21]:

	summary	Country/Region	Confirmed	Deaths	Recovered	Active	New cases	
0	count	174	174	174	174	174	174	
1	mean	None	94682.02298850575	3758.080459770115	54396.84482758621	36527.097701149425	1314.2816091954023	31.12(
2	stddev	None	396678.03033529996	14586.544569931522	196683.77697176125	220987.46339036332	5910.842358130151	124.1!
3	min	Afghanistan	14	0	0	1	0	
4	max	Zimbabwe	4290259	148011	1846641	2816444	56336	

In [22]:

```
new_df.printSchema()
```

```
root
 |-- Country/Region: string (nullable = true)
 |-- Confirmed: integer (nullable = true)
 |-- Deaths: integer (nullable = true)
 |-- Recovered: integer (nullable = true)
 |-- Active: integer (nullable = true)
 |-- New cases: integer (nullable = true)
 |-- New deaths: integer (nullable = true)
 |-- New recovered: integer (nullable = true)
 |-- Deaths / 100 Cases: double (nullable = true)
 |-- Recovered / 100 Cases: double (nullable = true)
 |-- Deaths / 100 Recovered: integer (nullable = true)
 |-- Confirmed last week: integer (nullable = true)
 |-- 1 week change: integer (nullable = true)
 |-- growth_factor: double (nullable = true)
 |-- WHO Region: string (nullable = true)
```

In [23]:

```
new_df.toPandas().head()
```

Out[23]:

	Country/Region	Confirmed	Deaths	Recovered	Active	New cases	New deaths	New recovered	Deaths / 100 Cases	Recovered / 100 Cases	Deaths / 100 Recovered	Confirmed last week	1 week change	gro
0	New Zealand	1557	22	1514	21	1	0	1	1.41	97.24	1.0	1555	2	
1	Guinea-Bissau	1954	26	803	1125	0	0	0	1.33	41.10	3.0	1949	5	
2	Mauritius	344	10	332	2	0	0	0	2.91	96.51	3.0	343	1	
3	Ireland	25892	1764	23364	764	11	0	0	6.81	90.24	7.0	25766	126	
4	Estonia	2034	69	1923	42	0	0	1	3.39	94.54	3.0	2021	13	

Data Visualization

In [24]:

```
part = data.iloc[:10, :5]
part
```

Out[24]:

Country/Region	Confirmed	Deaths	Recovered	Active	New cases
Afghanistan	36263	1269	25198	9796	106
Albania	4880	144	2745	1991	117
Algeria	27973	1163	18837	7973	616
Andorra	907	52	803	52	10
Angola	950	41	242	667	18
Antigua and Barbuda	86	3	65	18	4
Argentina	167416	3059	72575	91782	4890
Armenia	37390	711	26665	10014	73
Australia	15303	167	9311	5825	368
Austria	20558	713	18246	1599	86

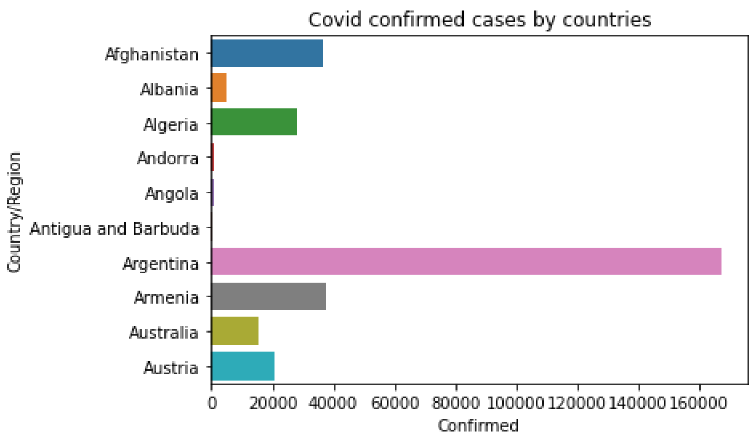
In [25]:

```
#Displaying the number of confirmed cases for the first 10 countries by default
sns.barplot(part['Confirmed'], part.index).set_title('Covid confirmed cases by countries')
```

```
/home/muz/.local/lib/python3.8/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following variables as keyword args: x, y.
From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an e
rror or misinterpretation.
  warnings.warn(
```

Out[25]:

```
Text(0.5, 1.0, 'Covid confirmed cases by countries')
```



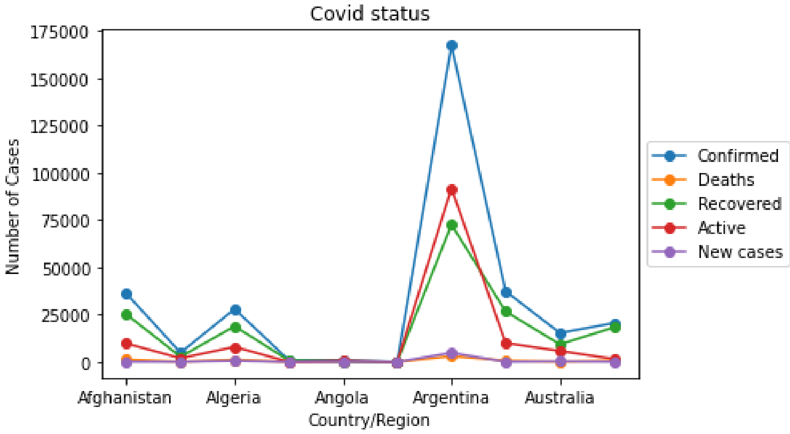
In [26]:

```
# Displaying the Deaths, Recovered and Active Cases across different countries[first 5]
part.plot(style='o-')
plt.legend(loc='center left', bbox_to_anchor=(1.0, 0.5))
plt.title("Covid status")
```

```
plt.ylabel("Number of Cases")
```

Out[26]:

Text(0, 0.5, 'Number of Cases')



In [27]:

```
# Visualization of Total Confirmed cases, Total Deaths and Total Active cases for top 20 countries
fig=px.bar(world_data.iloc[:20,:],y='Country/Region',x='TotalCases',color='TotalCases',text="TotalCases")
fig.update_layout(template="plotly_dark",title_text="<b>Top 20 countries of Total confirmed cases</b>")
fig.show()
fig=px.bar(world_data.sort_values('TotalDeaths',ascending=False).iloc[:20,:],y='Country/Region',x='TotalDeaths',color='TotalDeaths',text="To
fig.update_layout(template="plotly_dark",title_text="<b>Top 20 countries of Total deaths</b>")
fig.show()
fig=px.bar(world_data.sort_values('ActiveCases',ascending=False).iloc[:20,:],y='Country/Region',x='ActiveCases',color='ActiveCases',text='A
fig.update_layout(template="plotly_dark",title_text="<b>Top 20 countries of Total Active cases</b>")
fig.show()
fig=px.bar(world_data.sort_values('TotalRecovered',ascending=False).iloc[:20,:],y='Country/Region',x='TotalRecovered',color='TotalRecovere
fig.update_layout(template="plotly_dark",title_text="<b>Top 20 countries of Total Recovered</b>")
fig.show()
```

In [28]:

```
#Avg of growthfactor used to predict the percentile of week 2
con_sum = new_df.groupBy("Confirmed").sum()
death_sum = new_df.groupBy("Deaths").sum()
rec_sum = new_df.groupBy("Recovered").sum()

avg_growth = new_df.groupBy("growth_factor").sum()
```

In [29]:

```
avg_growth.toPandas().head()
```

Out[29]:

	growth_factor	sum(Confirmed)	sum(Deaths)	sum(Recovered)	sum(Active)	sum(New cases)	sum(New deaths)	sum(New recovered)	sum(Deaths / 100 Cases)	sum(Recovered / 100 Cases)
0	0.13	1557	22	1514	21	1	0	1	1.41	97.24
1	0.26	1954	26	803	1125	0	0	0	1.33	41.10
2	0.29	344	10	332	2	0	0	0	2.91	96.51
3	0.49	25892	1764	23364	764	11	0	0	6.81	90.24
4	0.64	2034	69	1923	42	0	0	1	3.39	94.54

ML algorithms applied using Spark

Determining the label and feature column to split the dataset

In [30]:

```
for item in new_df.head(1)[0]:
    print(item)
```

```
New Zealand
1557
22
1514
21
1
0
1
1.41
97.24
1
1555
2
0.13
Western Pacific
```

In [31]:

```
new_df.columns
```

Out[31]:

```
['Country/Region',
 'Confirmed',
 'Deaths',
 'Recovered',
 'Active',
 'New cases',
 'New deaths',
 'New recovered',
 'Deaths / 100 Cases',
 'Recovered / 100 Cases',
 'Deaths / 100 Recovered',
 'Confirmed last week',
 '1 week change',
 'growth_factor',
 'WHO Region']
```

In [32]:

```
assembler = VectorAssembler(inputCols=['Confirmed','1 week change',
 'Confirmed last week'], outputCol='features')
```

In [33]:

```
output = assembler.transform(new_df)
```

In [34]:

```
output.printSchema()
```

```
root
 |-- Country/Region: string (nullable = true)
 |-- Confirmed: integer (nullable = true)
 |-- Deaths: integer (nullable = true)
 |-- Recovered: integer (nullable = true)
 |-- Active: integer (nullable = true)
 |-- New cases: integer (nullable = true)
 |-- New deaths: integer (nullable = true)
 |-- New recovered: integer (nullable = true)
 |-- Deaths / 100 Cases: double (nullable = true)
 |-- Recovered / 100 Cases: double (nullable = true)
 |-- Deaths / 100 Recovered: integer (nullable = true)
 |-- Confirmed last week: integer (nullable = true)
 |-- 1 week change: integer (nullable = true)
 |-- growth_factor: double (nullable = true)
 |-- WHO Region: string (nullable = true)
 |-- features: vector (nullable = true)
```

In [35]:

```
final_data = output.select("features", "growth_factor")
```

In [36]:

```
final_data.show()
```

```
+--------------------+-------------+
|            features|growth_factor|
+--------------------+-------------+
| [1557.0,2.0,1555.0]|         0.13|
| [1954.0,5.0,1949.0]|         0.26|
|   [344.0,1.0,343.0]|         0.29|
|[25892.0,126.0,25...|         0.49|
|[2034.0,13.0,2021.0]|         0.64|
|[246286.0,1662.0,...|         0.68|
|   [289.0,2.0,287.0]|          0.7|
|[5059.0,39.0,5020.0]|         0.78|
|[7398.0,58.0,7340.0]|         0.79|
|[1854.0,15.0,1839.0]|         0.82|
|[9132.0,98.0,9034.0]|         1.08|
|[4599.0,51.0,4548.0]|         1.12|
|[8904.0,104.0,880...|         1.18|
|[86783.0,1161.0,8...|         1.36|
|[3297.0,47.0,3250.0]|         1.45|
|[2513.0,38.0,2475.0]|         1.54|
|[67251.0,1038.0,6...|         1.57|
|[301708.0,4764.0,...|          1.6|
|[79395.0,1347.0,7...|         1.73|
|[207112.0,3787.0,...|         1.86|
+--------------------+-------------+
only showing top 20 rows
```

In [37]:

```
#Splitting the Dataset
train_data,test_data = final_data.randomSplit([0.8,0.2])
```

Linear Regression

In [38]:

```
#Initialize the model and assign to an object
lr = LinearRegression(labelCol="growth_factor")
```

In [39]:

```
#Training the model using train dataset
lr_model = lr.fit(train_data)
```

In [40]:

```
#Evaluating the model using the test dataset
test_results = lr_model.evaluate(test_data)
```

In [41]:

```
test_results.residuals.show()
```

```
+-------------------+
|          residuals|
+-------------------+
|-10.342908548530055|
|-12.445341418826391|
|-12.310432666992382|
|-12.260911291083795|
| -9.626351699311964|
|-10.381571672499517|
| -11.34272449071033|
| -10.10315486688515|
|-10.668273608717577|
| -10.25259971267764|
|-10.001995004429709|
|  -9.53787584771572|
| -9.526679406361826|
| -6.876238894182514|
| -5.439312032681437|
| -5.709388615431024|
| -5.408952467318873|
| -5.082417514962112|
|-2.6663941275472656|
| -2.558042692802866|
+-------------------+
only showing top 20 rows
```

In [42]:

```
#Rootmeansquared error
print("RMSE for Linear Regression")
print(test_results.rootMeanSquaredError)
```

```
RMSE for Linear Regression
13.743615446702012
```

In [43]:

```
#Returning the coefficient of determination
print("R Squared Value for Linear Regression")
print(test_results.r2)
```

```
R Squared Value for Linear Regression
0.04899621536079357
```

In [44]:

```
final_data.describe().show()
```

```
+-------+------------------+
|summary|     growth_factor|
+-------+------------------+
|  count|               174|
|   mean|14.644827586206901|
| stddev|25.103550073060976|
|    min|              0.13|
|    max|            226.32|
+-------+------------------+
```

In [45]:

```
#Data prediction using the Feature column
unlabled_data = test_data.select("features")
```

In [46]:

```
unlabled_data.show()
```

```
+--------------------+
|            features|
+--------------------+
|[109597.0,2560.0,...|
|[1132.0,27.0,1105.0]|
|[4448.0,109.0,433...|
|   [350.0,9.0,341.0]|
|[116458.0,3533.0,...|
|[59177.0,1984.0,5...|
|[2532.0,86.0,2446.0]|
|[66428.0,2334.0,6...|
|[1783.0,72.0,1711.0]|
|[7235.0,314.0,692...|
|[6208.0,285.0,592...|
|[18752.0,908.0,17...|
|[1455.0,74.0,1381.0]|
|  [148.0,11.0,137.0]|
|[81161.0,6541.0,7...|
|[9764.0,816.0,894...|
|[15655.0,1343.0,1...|
|[23154.0,2039.0,2...|
|  [431.0,47.0,384.0]|
|[3369.0,370.0,299...|
+--------------------+
only showing top 20 rows
```

In [47]:

```
prediction = lr_model.transform(unlabled_data)
```

In [48]:

```
# Growth Factor prediction
prediction.show()
```

```
+--------------------+------------------+
|            features|        prediction|
+--------------------+------------------+
|[109597.0,2560.0,...|12.732908548530055|
|[1132.0,27.0,1105.0]| 14.88534141882639|
|[4448.0,109.0,433...|14.820432666992382|
|   [350.0,9.0,341.0]|14.900911291083796|
|[116458.0,3533.0,...|12.756351699311965|
|[59177.0,1984.0,5...|13.851571672499517|
|[2532.0,86.0,2446.0]| 14.86272449071033|
|[66428.0,2334.0,6...| 13.74315486688515|
|[1783.0,72.0,1711.0]|14.878273608717576|
|[7235.0,314.0,692...|14.792599712677639|
|[6208.0,285.0,592...|14.811995004429708|
|[18752.0,908.0,17...| 14.62787584771572|
|[1455.0,74.0,1381.0]|14.886679406361825|
|  [148.0,11.0,137.0]|14.906238894182513|
|[81161.0,6541.0,7...|14.209312032681437|
|[9764.0,816.0,894...|14.829388615431023|
|[15655.0,1343.0,1...|14.788952467318873|
|[23154.0,2039.0,2...|14.742417514962112|
|  [431.0,47.0,384.0]|14.906394127547266|
|[3369.0,370.0,299...|14.898042692802866|
+--------------------+------------------+
only showing top 20 rows
```

Decision Tree Regressor

In [49]:

```
#Intialize the decision tree regressor
featureIndexer = VectorIndexer(inputCol="features", outputCol="indexedFeatures", maxCategories=4).fit(final_data)
dtr = DecisionTreeRegressor(labelCol= "growth_factor",featuresCol= "features")
```

In [50]:

```
pipeline = Pipeline(stages=[featureIndexer, dtr])
```

In [51]:

```
dtr_model = pipeline.fit(train_data)
```

In [52]:

```
dtr_predictions = dtr_model.transform(test_data)
```

In [53]:

```
dtr_predictions.select("prediction", "growth_factor", "features").show(5)
```

```
+------------------+-------------+--------------------+
|        prediction|growth_factor|            features|
+------------------+-------------+--------------------+
| 6.615172413793103|         2.39|[109597.0,2560.0,...|
|2.5714285714285716|         2.44|[1132.0,27.0,1105.0]|
|1.5219999999999998|         2.51|[4448.0,109.0,433...|
|2.5714285714285716|         2.64|   [350.0,9.0,341.0]|
| 6.615172413793103|         3.13|[116458.0,3533.0,...|
+------------------+-------------+--------------------+
only showing top 5 rows
```

In [58]:

```
#Evaluate the model that has been trianed using the train dataset where label is growth factor
print("Root Mean Squared Error (RMSE) for Decision Tree on test data")
dtr_evaluator = RegressionEvaluator(
    labelCol="growth_factor", predictionCol="prediction", metricName="rmse")
dtr_rmse = dtr_evaluator.evaluate(dtr_predictions)
print(dtr_rmse)
```

```
Root Mean Squared Error (RMSE) for Decision Tree on test data
10.34182489586513
```

In [59]:

```
#Summary of the model
dtr_treeModel = dtr_model.stages[1]
print(dtr_treeModel)
```

```
DecisionTreeRegressionModel: uid=DecisionTreeRegressor_ef24775c252f, depth=5, numNodes=53, numFeatures=3
```

RandomForest Tree regressor

In [60]:

```
#Initalize the Randomforest Regressor
rfr = RandomForestRegressor(numTrees= 100,labelCol= "growth_factor", featuresCol= "features")
rfr_pipeline = Pipeline(stages=[featureIndexer, rfr])
rfr_model = rfr_pipeline.fit(train_data)
rfr_predictions = rfr_model.transform(test_data)
```

In [61]:

```
rfr_predictions.select("prediction", "growth_factor", "features").show(5)
```

```
+------------------+-------------+--------------------+
|        prediction|growth_factor|            features|
+------------------+-------------+--------------------+
|10.191944281248436|         2.39|[109597.0,2560.0,...|
| 4.681050535452821|         2.44|[1132.0,27.0,1105.0]|
| 7.367222753089058|         2.51|[4448.0,109.0,433...|
| 8.590482416711305|         2.64|   [350.0,9.0,341.0]|
|10.992051185772498|         3.13|[116458.0,3533.0,...|
+------------------+-------------+--------------------+
only showing top 5 rows
```

In [64]:

```
#Evaluate the model that has been trianed using the train dataset where label is growth factor
print("Root Mean Squared Error (RMSE) for Random Forest on test data")
rfr_evaluator = RegressionEvaluator(
    labelCol="growth_factor", predictionCol="prediction", metricName="rmse")
rfr_rmse = rfr_evaluator.evaluate(rfr_predictions)
print(rfr_rmse)
```

```
Root Mean Squared Error (RMSE) for Random Forest on test data
12.460239542681974
```

In [65]:

```
#Summary of the model
rfr_treeModel = rfr_model.stages[1]
```

```
print(rfr_treeModel)
```

```
RandomForestRegressionModel: uid=RandomForestRegressor_f22d1d717bdd, numTrees=100, numFeatures=3
```

Gradient Boost Tree Regressor

In [66]:

```
#Initialize Gradient Booster Regressor
gbr = GBTRegressor(labelCol= "growth_factor", featuresCol= "features")
gbr_pipeline = Pipeline(stages=[featureIndexer, gbr])
gbr_model = gbr_pipeline.fit(train_data)
gbr_predictions = gbr_model.transform(test_data)
```

In [67]:

```
gbr_predictions.select("prediction", "growth_factor", "features").show(5)
```

```
+------------------+-------------+--------------------+
|        prediction|growth_factor|            features|
+------------------+-------------+--------------------+
| 5.169092022645198|         2.39|[109597.0,2560.0,...|
| 2.680869028166836|         2.44|[1132.0,27.0,1105.0]|
| 1.806442660961181|         2.51|[4448.0,109.0,433...|
|1.5040683028236084|         2.64|   [350.0,9.0,341.0]|
|5.4472733099287804|         3.13|[116458.0,3533.0,...|
+------------------+-------------+--------------------+
only showing top 5 rows
```

In [69]:

```
#Evaluate the model that has been trianed using the train dataset where label is growth factor
print("Root Mean Squared Error (RMSE) for Gradient Boost on test data")
gbr_evaluator = RegressionEvaluator(
    labelCol="growth_factor", predictionCol="prediction", metricName="rmse")
gbr_rmse = gbr_evaluator.evaluate(gbr_predictions)
print(gbr_rmse)
```

```
Root Mean Squared Error (RMSE) for Gradient Boost on test data
7.833917524401471
```

In [70]:

```
#Summary of the model
gbr_treeModel = gbr_model.stages[1]
print(gbr_treeModel)
```

```
GBTRegressionModel: uid=GBTRegressor_66dd9e38cacb, numTrees=20, numFeatures=3
```

In [71]:

```
#Data prediction using the Feature column[if new entries were added to the current dataset]
unlabled_data = test_data.select("features")
```

In [72]:

```
unlabled_data.show()
```

```
+--------------------+
|            features|
+--------------------+
|[109597.0,2560.0,...|
|[1132.0,27.0,1105.0]|
|[4448.0,109.0,433...|
|   [350.0,9.0,341.0]|
|[116458.0,3533.0,...|
|[59177.0,1984.0,5...|
|[2532.0,86.0,2446.0]|
|[66428.0,2334.0,6...|
|[1783.0,72.0,1711.0]|
|[7235.0,314.0,692...|
|[6208.0,285.0,592...|
|[18752.0,908.0,17...|
|[1455.0,74.0,1381.0]|
|  [148.0,11.0,137.0]|
|[81161.0,6541.0,7...|
|[9764.0,816.0,894...|
|[15655.0,1343.0,1...|
|[23154.0,2039.0,2...|
|  [431.0,47.0,384.0]|
|[3369.0,370.0,299...|
+--------------------+
only showing top 20 rows
```

In [73]:

```
prediction_2 = gbr_model.transform(unlabled_data)
```

In [74]:

```
# Growth Factor prediction
prediction_2.show()
```

```
+--------------------+--------------------+------------------+
|            features|     indexedFeatures|        prediction|
+--------------------+--------------------+------------------+
|[109597.0,2560.0,...|[109597.0,2560.0,...| 5.169092022645198|
|[1132.0,27.0,1105.0]|[1132.0,27.0,1105.0]| 2.680869028166836|
|[4448.0,109.0,433...|[4448.0,109.0,433...| 1.806442660961181|
|   [350.0,9.0,341.0]|   [350.0,9.0,341.0]|1.5040683028236084|
|[116458.0,3533.0,...|[116458.0,3533.0,...|5.4472733099287804|
|[59177.0,1984.0,5...|[59177.0,1984.0,5...| 2.388992576577857|
|[2532.0,86.0,2446.0]|[2532.0,86.0,2446.0]| 5.126350710171436|
|[66428.0,2334.0,6...|[66428.0,2334.0,6...| 6.337634515145746|
|[1783.0,72.0,1711.0]|[1783.0,72.0,1711.0]| 4.924440456738263|
|[7235.0,314.0,692...|[7235.0,314.0,692...| 6.242074332209283|
|[6208.0,285.0,592...|[6208.0,285.0,592...| 6.242074332209283|
|[18752.0,908.0,17...|[18752.0,908.0,17...| 4.424208005309042|
|[1455.0,74.0,1381.0]|[1455.0,74.0,1381.0]| 4.924440456738263|
|  [148.0,11.0,137.0]|  [148.0,11.0,137.0]| 5.818553586633133|
|[81161.0,6541.0,7...|[81161.0,6541.0,7...| 8.745725271437182|
|[9764.0,816.0,894...|[9764.0,816.0,894...| 8.313693109334517|
|[15655.0,1343.0,1...|[15655.0,1343.0,1...| 14.27559875845944|
|[23154.0,2039.0,2...|[23154.0,2039.0,2...| 15.78728864959134|
|  [431.0,47.0,384.0]|  [431.0,47.0,384.0]| 5.515408268537896|
|[3369.0,370.0,299...|[3369.0,370.0,299...| 11.08392957885411|
+--------------------+--------------------+------------------+
only showing top 20 rows
```

Hyperparameter Tuning using Train Validation Split

In [75]:

```
#Renaming the growth_factor column to its default name
final_data = final_data.withColumnRenamed('growth_factor','label')
final_data.show(5)
```

```
+--------------------+-----+
|            features|label|
+--------------------+-----+
| [1557.0,2.0,1555.0]| 0.13|
| [1954.0,5.0,1949.0]| 0.26|
|   [344.0,1.0,343.0]| 0.29|
|[25892.0,126.0,25...| 0.49|
|[2034.0,13.0,2021.0]| 0.64|
+--------------------+-----+
only showing top 5 rows
```

In [76]:

```
#Splitting the dataset into train and test
train, test = final_data.randomSplit([0.8, 0.2], seed=12345)
```

In [77]:

```
#Initialize linear regression model
lr2 = LinearRegression( maxIter=10)
train.show(5)
```

```
+--------------------+-----+
|            features|label|
+--------------------+-----+
| [1557.0,2.0,1555.0]| 0.13|
| [1954.0,5.0,1949.0]| 0.26|
|   [344.0,1.0,343.0]| 0.29|
|[25892.0,126.0,25...| 0.49|
|[2034.0,13.0,2021.0]| 0.64|
+--------------------+-----+
only showing top 5 rows
```

In [78]:

```
test.show(5)
```

```
+--------------------+-----+
|            features|label|
+--------------------+-----+
|[246286.0,1662.0,...| 0.68|
|[9132.0,98.0,9034.0]| 1.08|
|[2513.0,38.0,2475.0]| 1.54|
|[301708.0,4764.0,...|  1.6|
|  [462.0,11.0,451.0]| 2.44|
+--------------------+-----+
only showing top 5 rows
```

In [79]:

```
paramGrid = ParamGridBuilder()\
    .addGrid(lr.regParam, [0.1, 0.01]) \
    .addGrid(lr.fitIntercept, [False, True])\
    .addGrid(lr.elasticNetParam, [0.0, 0.5, 1.0])\
    .build()
```

In [80]:

```
tvs = TrainValidationSplit(estimator=lr2,
                           estimatorParamMaps=paramGrid,
                           evaluator=RegressionEvaluator(),
                           trainRatio=0.8)
```

In [81]:

```
# Run TrainValidationSplit, and choose the best set of parameters.
model = tvs.fit(train)
```

In [82]:

```
model.transform(test)\
    .select("features", "label", "prediction")\
    .show()
```

```
+--------------------+-----+------------------+
|            features|label|        prediction|
+--------------------+-----+------------------+
|[246286.0,1662.0,...| 0.68| 9.241321131457799|
|[9132.0,98.0,9034.0]| 1.08|14.336751651536426|
|[2513.0,38.0,2475.0]| 1.54|14.475933105974457|
|[301708.0,4764.0,...|  1.6| 8.558165394245743|
|  [462.0,11.0,451.0]| 2.44|14.517500925236522|
|[1132.0,27.0,1105.0]| 2.44|14.505253860410901|
|  [907.0,23.0,884.0]|  2.6|14.509621400458085|
|[14203.0,387.0,13...|  2.8|14.275203958824356|
|[227019.0,6447.0,...| 2.92|10.566486739971866|
|[50299.0,1528.0,4...| 3.13|13.667154199699686|
|[116458.0,3533.0,...| 3.13|12.536674567540102|
|  [701.0,24.0,677.0]| 3.55|14.514484979615759|
|[2305.0,94.0,2211.0]| 4.25|14.491044737615399|
|     [24.0,1.0,23.0]| 4.35| 14.52559249112532|
|[1167.0,60.0,1107.0]| 5.42|14.510580622108927|
|  [853.0,44.0,809.0]| 5.44|14.514743243676852|
|   [114.0,6.0,108.0]| 5.56|14.524476183406323|
|  [265.0,14.0,251.0]| 5.58|14.522531127284006|
|[50838.0,2803.0,4...| 5.84|13.891429601715728|
|[39482.0,2546.0,3...| 6.89|14.101640130619082|
+--------------------+-----+------------------+
only showing top 20 rows
```

In [ ]:

In [ ]: