BIT2203

ADVANCED DATABASE MANAGEMENT SYSTEM

GROUP 10

james mwangi nyoro sct221-0086/2021

1. **How data was compiled:**

   We got our data source from this link;
   https://coronavirus.jhu.edu/
   The data came in three excel sheets that included;

      time_series_covid19_recovered_global.csv
      time_series_covid19_deaths_global.csv
      time_series_covid19_confirmed_global.csv

   The data sources included data from all the countries in the world.

1	Province/	Country/R	Lat	Long	########	########	########	########	########	########	########	########	########	########	2/1/2020	2/2/2020	2/3/2020	2/4/2020	2/5/2020	2/6/2020	2/7
2		Afghanist	33.93911	67.70995	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
3		Albania	41.1533	20.1683	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
4		Algeria	28.0339	1.6596	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
5		Andorra	42.5063	1.5218	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
6		Angola	-11.2027	17.8739	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
7		Antarctica	-71.9499	23.347	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
8		Antigua ar	17.0608	-61.7964	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
9		Argentina	-38.4161	-63.6167	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
10		Armenia	40.0691	45.0382	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
11	Australian	Australia	-35.4735	149.0124	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
12	New Sout	Australia	-33.8688	151.2093	0	0	0	0	0	0	0	0	2	2	2	2	2	2	2	2	
13	Northern	Australia	-12.4634	130.8456	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

   We created a separate excel sheets and copied Kenya covid 19 data to it from the three csv files.

1	Province/	Country/R	Lat	Long	1/22/2020	1/23/2020	1/24/2020	1/25/2020	1/26/2020	1/27/2020	1/28/2020	1/29/2020	1/30/2020	1/31/2020	2/1/2020	2/2/2020	2/3/2020	2/4/20
2		Kenya	-0.0236	37.9062	0	0	0	0	0	0	0	0	0	0	0	0	0	
3		Kenya	-0.0236	37.9062	0	0	0	0	0	0	0	0	0	0	0	0	0	
4		Kenya	-0.0236	37.9062	0	0	0	0	0	0	0	0	0	0	0	0	0	
5																		

We then transposed the data cells and added new column heads the transformed the excel sheet to a csv file.

1		Infected	Recovere	Deaths
2	Province/State			
3	Country/R	Kenya	Kenya	Kenya
4	Lat	-0.0236	-0.0236	-0.0236
5	Long	37.9062	37.9062	37.9062
6	1/22/2020	0	0	0
7	1/23/2020	0	0	0
8	1/24/2020	0	0	0
9	1/25/2020	0	0	0
10	1/26/2020	0	0	0
11	1/27/2020	0	0	0
12	1/28/2020	0	0	0
13	1/29/2020	0	0	0

2. **How data was ingested into Hadoop data lake:**

We created a directory using the command **hdfs dfs -mkdir /kenyac19data** and inserted the prepared csv file into it using the command **hdfs dfs -put D:\filepath /kenyac19data**

```
C:\hadoop\hadoop-2.9.0\bin>hdfs dfs -mkdir /kenyac19data

C:\hadoop\hadoop-2.9.0\bin>hdfs dfs -ls /
Found 1 items
drwxr-xr-x   - hp supergroup          0 2023-12-01 21:56 /kenyac19data

C:\hadoop\hadoop-2.9.0\bin>hdfs dfs -put D:\class_units\3.1\AdvDB\TakeAwayCat\kenyac19data.csv /kenyac19data

C:\hadoop\hadoop-2.9.0\bin>hdfs dfs -ls /
Found 1 items
drwxr-xr-x   - hp supergroup          0 2023-12-01 21:58 /kenyac19data

C:\hadoop\hadoop-2.9.0\bin>hdfs dfs -ls /kenyac19data/kenyac19data.csv
-rw-r--r--   1 hp supergroup      28681 2023-12-01 21:58 /kenyac19data/kenyac19data.csv
```

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
drwxr-xr-x	hp	supergroup	0 B	Dec 01 21:58	0	0 B	kenyac19data

/kenyac19data Go!

Show 25 entries Search:

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
-rw-r--r--	hp	supergroup	28.01 KB	Dec 01 21:58	1	128 MB	kenyac19data.csv

3. How data was extracted using pyspark:

We imported the SparkSession library, created a session then loaded our file from our Hadoop data lake into pyspark by specifying the files path. We then displayed the csv file.

```
>>> from pyspark.sql import SparkSession
>>> spark = SparkSession.builder.appName("DataLakeExtract").getOrCreate()
>>> csv_files_path = "hdfs://localhost:9000/kenyac19data/kenyac19data.csv"
>>> data = spark.read.format("csv").option("header", "true").load(csv_files_path)
>>> data.show()
+--------------+--------+---------+-------+
|         Dates|Infected|Recovered| Deaths|
+--------------+--------+---------+-------+
|Province/State|    null|     null|   null|
|Country/Region|   Kenya|    Kenya|  Kenya|
|           Lat| -0.0236|  -0.0236|-0.0236|
|          Long| 37.9062|  37.9062|37.9062|
|      2/1/2020|       0|        0|      0|
|      2/2/2020|       0|        0|      0|
|      2/3/2020|       0|        0|      0|
|      2/4/2020|       0|        0|      0|
|      2/5/2020|       0|        0|      0|
|      2/6/2020|       0|        0|      0|
|      2/7/2020|       0|        0|      0|
|      2/8/2020|       0|        0|      0|
|      2/9/2020|       0|        0|      0|
|      3/1/2020|       0|        0|      0|
|      3/2/2020|       0|        0|      0|
|      3/3/2020|       0|        0|      0|
|      3/4/2020|       0|        0|      0|
|      3/5/2020|       0|        0|      0|
|      3/6/2020|       0|        0|      0|
|      3/7/2020|       0|        0|      0|
+--------------+--------+---------+-------+
only showing top 20 rows
```

4. Pre-processing tasks/techniques used to prepare data.

We removed all the rows with null values, string values and floating point values. We remained with only integer values in the Infected, Recovered and Deaths columns which would be easier to work with.

```
>>> from pyspark.sql.functions import col
>>> data_filtered = data.filter((col("Infected") % 1 == 0) | (col("Recovered") % 1 == 0) | (col("Deaths") % 1 == 0))
>>> data_filtered.show()
+---------+--------+---------+------+
|    Dates|Infected|Recovered|Deaths|
+---------+--------+---------+------+
| 2/1/2020|       0|        0|     0|
| 2/2/2020|       0|        0|     0|
| 2/3/2020|       0|        0|     0|
| 2/4/2020|       0|        0|     0|
| 2/5/2020|       0|        0|     0|
| 2/6/2020|       0|        0|     0|
| 2/7/2020|       0|        0|     0|
| 2/8/2020|       0|        0|     0|
| 2/9/2020|       0|        0|     0|
| 3/1/2020|       0|        0|     0|
| 3/2/2020|       0|        0|     0|
| 3/3/2020|       0|        0|     0|
| 3/4/2020|       0|        0|     0|
| 3/5/2020|       0|        0|     0|
| 3/6/2020|       0|        0|     0|
| 3/7/2020|       0|        0|     0|
| 3/8/2020|       0|        0|     0|
| 3/9/2020|       0|        0|     0|
|3/10/2020|       0|        0|     0|
|3/11/2020|       0|        0|     0|
+---------+--------+---------+------+
only showing top 20 rows
```

Next we removed all the rows that had zero (0) values in all the columns. The zero values in all columns makes the rows irrelevant as we need real numbers to work with.

```
>>> df = data_filtered.filter((col("Infected") != 0) | (col("Recovered") != 0) | (col("Deaths") != 0))
>>> df.show()
+---------+--------+---------+------+
|    Dates|Infected|Recovered|Deaths|
+---------+--------+---------+------+
|3/13/2020|       1|        0|     0|
|3/14/2020|       1|        0|     0|
|3/15/2020|       3|        0|     0|
|3/16/2020|       3|        0|     0|
|3/17/2020|       3|        0|     0|
|3/18/2020|       3|        0|     0|
|3/19/2020|       7|        0|     0|
|3/20/2020|       7|        0|     0|
|3/21/2020|       7|        0|     0|
|3/22/2020|      15|        0|     0|
|3/23/2020|      16|        0|     0|
|3/24/2020|      25|        0|     0|
|3/25/2020|      28|        1|     0|
|3/26/2020|      31|        1|     1|
|3/27/2020|      31|        1|     1|
|3/28/2020|      38|        1|     1|
|3/29/2020|      42|        1|     1|
|3/30/2020|      50|        1|     1|
|3/31/2020|      59|        1|     1|
| 4/1/2020|      81|        3|     1|
+---------+--------+---------+------+
only showing top 20 rows
```

5. Test Results and Interpretation:

We used the Mean Squared Error predictive analytics method to predict the number of deaths. The MSE we found was as 88540, that for all the infections recorded and the rate of recovery, this is the number of people that are expected to perish from the virus.

```
>>> print(df)
          Dates  Infected  Recovered  Deaths
0     3/13/2020         1          0       0
1     3/14/2020         1          0       0
2     3/15/2020         3          0       0
3     3/16/2020         3          0       0
4     3/17/2020         3          0       0
...         ...       ...        ...     ...
1087   3/5/2023    342919          0    5688
1088   3/6/2023    342919          0    5688
1089   3/7/2023    342932          0    5688
1090   3/8/2023    342937          0    5688
1091   3/9/2023    342937          0    5688

[1092 rows x 4 columns]
>>>
>>> from sklearn.model_selection import train_test_split
>>> from sklearn.linear_model import LinearRegression
>>> from sklearn.metrics import mean_squared_error
>>> df = pd.DataFrame(df)
>>> x = df[['Infected','Recovered']]
>>> y = df['Deaths']
>>> x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random
_state=40)
>>> model = LinearRegression()
>>> model.fit(x_train, y_train)
LinearRegression()
>>> predictions = model.predict(x_test)
>>> mse = mean_squared_error(y_test, predictions)
>>> print(f"Mean Squared Error: {mse}")
Mean Squared Error: 88540.72813249362
>>>
```

6. Validations results and interpretations:

We validated the value we got from the Mean Squared Error technique using the Root Mean Squared Error technique. The value we got was the exact root of the value we found in the MSE.

$\sqrt{88540.73} = 297.56$

```
>>>
>>> from sklearn.metrics import mean_squared_error
>>> rmse = mean_squared_error(y_test, predictions, squared=False)
>>> print("Root Mean Squared Error (RMSE):", rmse)
Root Mean Squared Error (RMSE): 297.5579407989201
>>>
>>>
```

7. Potential applications of the interpreted results:

a. ***Public Awareness:*** Communicating anticipated death rates based on infection numbers can raise public awareness about the seriousness of the situation. This information can encourage adherence to preventive measures and vaccination, potentially reducing the spread of the virus.

b. ***Mitigation Strategies:*** Knowing the potential death toll can prompt the implementation of targeted interventions in high-risk areas or among vulnerable populations, such as the elderly or those with preexisting health conditions.
c. ***Policy Making:*** Predictions can inform policymakers about the potential impact of the virus, guiding decisions on lockdowns, social distancing measures, travel restrictions, and vaccination drives. It helps in creating a more targeted and effective response.

Data Visualization:

We imported the python matplotlib library to visualize the model.

```
>>> import matplotlib.pyplot as plt
>>> residuals = y_test - predictions
>>> plt.scatter(predictions, residuals)
<matplotlib.collections.PathCollection object at 0x000001E14192D250>
>>> plt.xlabel('Predicted')
Text(0.5, 0, 'Predicted')
>>> plt.ylabel('Residuals')
Text(0, 0.5, 'Residuals')
>>> plt.axhline(y=0, color='r', linestyle='--')
<matplotlib.lines.Line2D object at 0x000001E1414766D0>
>>> plt.title('Visualize Deaths Residual Plot')
Text(0.5, 1.0, 'Visualize Deaths Residual Plot')
>>> plt.show()
```

The result was;

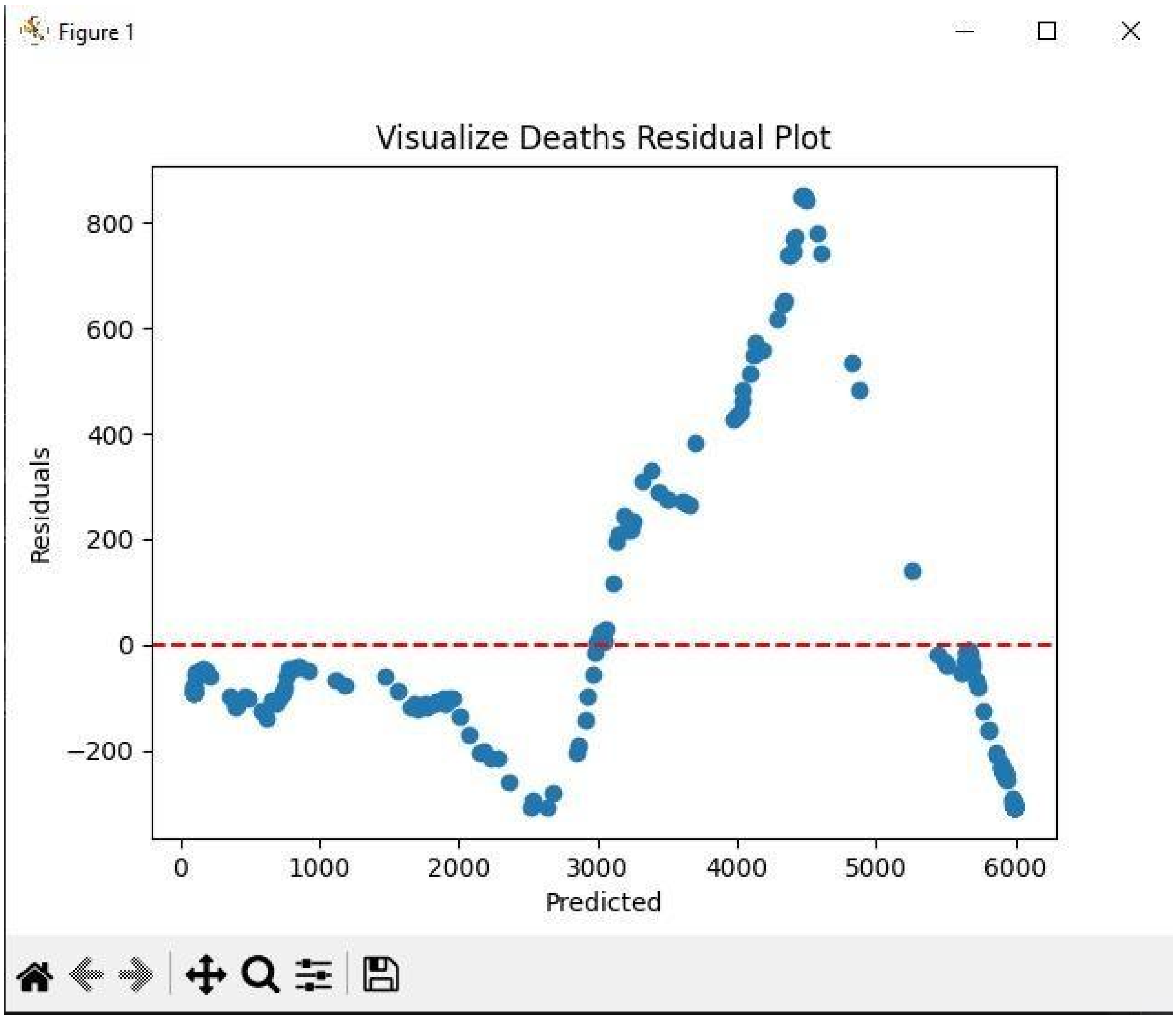
Figure 1
Visualize Deaths Residual Plot
Residuals
800
600
400
200
0
−200
0
1000
2000
3000
4000
5000
6000
Predicted