Problem Definition

Lets discuss on the data repository for the 2019 Novel Coronavirus Visual Dashboard operated by the Johns Hopkins University Center for Systems Science and Engineering (JHU CSSE). Also, Supported by ESRI Living Atlas Team and the Johns Hopkins University Applied Physics Lab (JHU APL).

Coronavirus is a family of viruses that can cause illness, which can vary from *common cold* and *cough* to sometimes more severe disease. **Middle East Respiratory Syndrome (MERS-CoV)** and **Severe Acute Respiratory Syndrome (SARS-CoV)** were such severe cases with the world already has faced. **SARS-CoV-2 (n-coronavirus)** is the new virus of the coronavirus family, which first *discovered* in 2019, which has not been identified in humans before. It is a *contiguous* virus which started from **Wuhan** in **December 2019**. Which later declared as **Pandemic** by **WHO** due to high rate spreads throughout the world. Currently (on the date 20 May 2020), this leads to a total of *300K+ Deaths* across the globe, including *90K+ deaths* alone in USA.The dataset is provided to identify the deaths and recovered cases.

For reference we ae taking the database from

https://github.com/dsrscientist/COVID_19_Datasets/blob/master/csse_covid_19_daily_reports_us.csv

Data Analysis

Database contains 58 rows and 18 columns, where our target variable is Province State we have considered this as a target variable just for to have a analysis of given data on the basis of geography.

Lets have a look on the tables we have in our database.

- **Province_State** - The name of the State within the USA.
- **Country_Region** - The name of the Country (US).
- **Last_Update** - The most recent date the file was pushed.
- **Lat** - Latitude.
- **Long_** - Longitude.
- **Confirmed** - Aggregated confirmed case count for the state.
- **Deaths** - Aggregated Death case count for the state.
- **Recovered** - Aggregated Recovered case count for the state.

- **Active** - Aggregated confirmed cases that have not been resolved (Active = Confirmed - Recovered - Deaths).
- **FIPS** - Federal Information Processing Standards code that uniquely identifies counties within the USA.
- **Incident_Rate** - confirmed cases per 100,000 persons.
- **People_Tested** - Total number of people who have been tested.
- **People_Hospitalized** - Total number of people hospitalized.
- **Mortality_Rate** - Number recorded deaths * 100/ Number confirmed cases.
- **UID** - Unique Identifier for each row entry.
- **ISO3** - Officialy assigned country code identifiers.
- **Testing_Rate** - Total number of people tested per 100,000 persons.
- **Hospitalization_Rate** - Total number of people hospitalized * 100/ Number of confirmed cases.

First we will refine this data by removing/ Dropping some columns for the database.

For our convenience we can drop ('Country_Region','ISO3','FIPS','UID','Last_Update') tables as these will not impact our database largly. After trimming the dataset we are left with 13 columns and on these columns we can perform our EDA process.

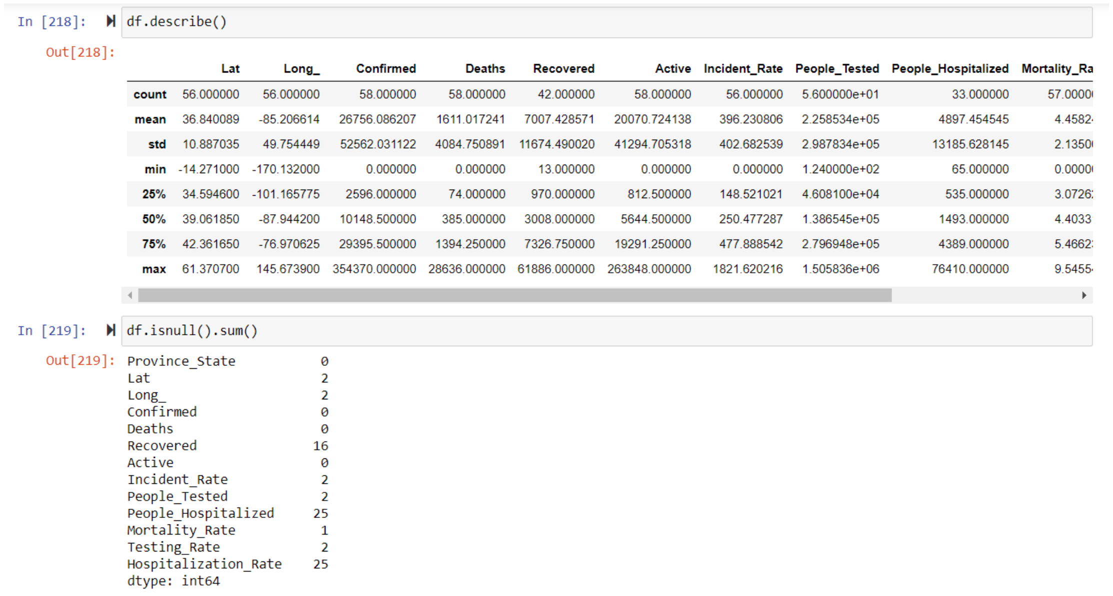
EDA can be performed by univariant analysis and multivariant analysis of various columns.

We can check for the datatypes of our datasets and can check any null values in our data sets.

We have observed that below columns have Null values. In which Recovered , People Hospit alized and hospitalization columns has Maximum Number of Null Values. We have to treat t hese values.

```
Lat                      2
Long_                    2
Recovered               16
Incident_Rate            2
People_Tested            2
People_Hospitalized     25
Mortality_Rate           1
Testing_Rate             2
Hospitalization_Rate    25
```

Following below table shows the result.



```
In [218]:  df.describe()
Out[218]:
```

	Lat	Long_	Confirmed	Deaths	Recovered	Active	Incident_Rate	People_Tested	People_Hospitalized	Mortality_Ra
count	56.000000	56.000000	58.000000	58.000000	42.000000	58.000000	56.000000	5.600000e+01	33.000000	57.0000
mean	36.840089	-85.206614	26756.086207	1611.017241	7007.428571	20070.724138	396.230806	2.258534e+05	4897.454545	4.4582
std	10.887035	49.754449	52562.031122	4084.750891	11674.490020	41294.705318	402.682539	2.987834e+05	13185.628145	2.1350
min	-14.271000	-170.132000	0.000000	0.000000	13.000000	0.000000	0.000000	1.240000e+02	65.000000	0.0000
25%	34.594600	-101.165775	2596.000000	74.000000	970.000000	812.500000	148.521021	4.608100e+04	535.000000	3.0726
50%	39.061850	-87.944200	10148.500000	385.000000	3008.000000	5644.500000	250.477287	1.386545e+05	1493.000000	4.4033
75%	42.361650	-76.970625	29395.500000	1394.250000	7326.750000	19291.250000	477.888542	2.796948e+05	4389.000000	5.4662
max	61.370700	145.673900	354370.000000	28636.000000	61886.000000	263848.000000	1821.620216	1.505836e+06	76410.000000	9.5455

```
In [219]:  df.isnull().sum()
Out[219]:  Province_State           0
           Lat                      2
           Long_                    2
           Confirmed                0
           Deaths                   0
           Recovered               16
           Active                   0
           Incident_Rate            2
           People_Tested            2
           People_Hospitalized     25
           Mortality_Rate           1
           Testing_Rate             2
           Hospitalization_Rate    25
           dtype: int64
```

From above we have observed the below findings.

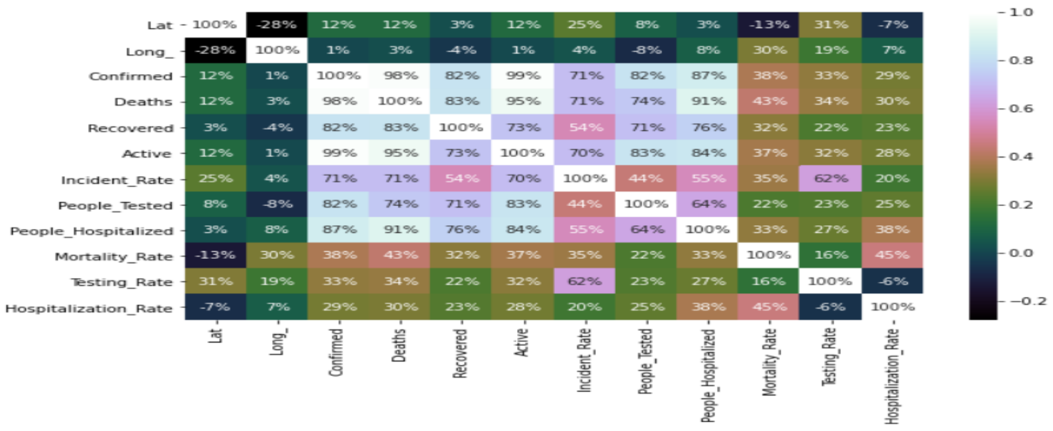
Observations:-

1) As count is Same so no null values in the data set.

2) Min Death Count is 0 and maximum is 28636 and vice versa we can check for Recovered cases.

3) There might be presence of outliers as the difference between maximmum and min is more also the std is more than mean in many columns.

Use of Heat Map to find correlation among various columns.



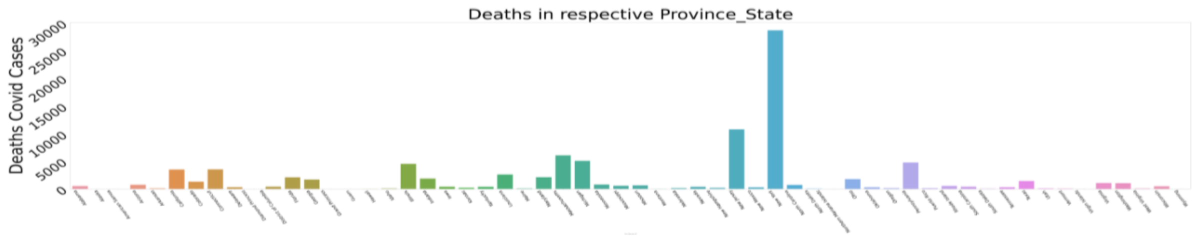
There is negative co relation between Lat and Long ,Testing Rate,Mortality Rate

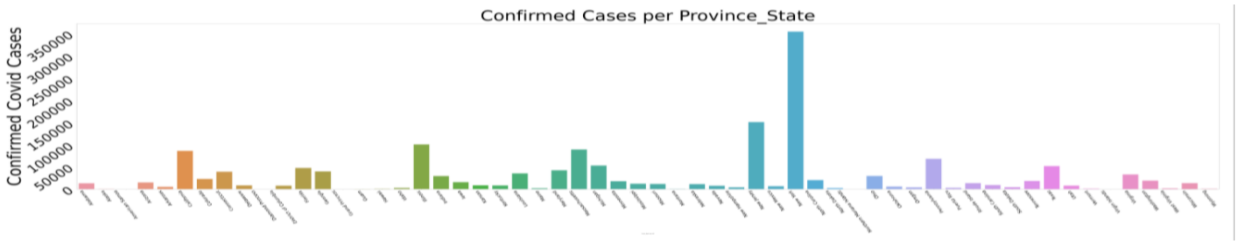
EDA Concluding Remarks

We have to perform Linear regression to perform Data analysis, as we have converted the whole data set into classification type of problem.

By performing Bilinear analysis, we come to No that New York has the highest rate of Confirmed COVID cases. After that New jearsey ,Massachusetts and so on.

Also the cases of Deaths are also more in Newyork.





Pre-processing Pipeline

We can use numpy, pandas, matplotlib. pyplot,, seaborn, sklearn libraries in analysis.

Now, as we can do process for removing outliers using Z score technique and will check skewness in the datasets.

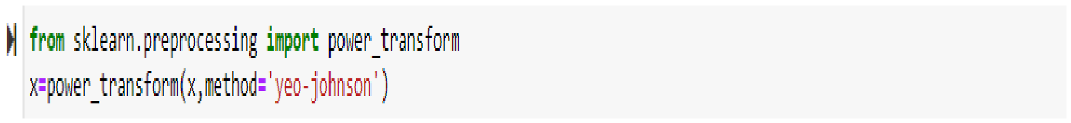
After doing all processing we can perform Label Encoding.

As Label Encoding in Python can be achieved using Sklearn Library. Label encoder encode labels with a value between 0 and n_classes-1 where n is the number of distinct labels. If a label repeats it assigns the same value to as assigned earlier.

So , we can use this on Province State column as that is our target variable.

Here I have tried Power Transformer module from the scikit-learn Python library as it is a quick, often-overlooked way to significantly improve model performance.

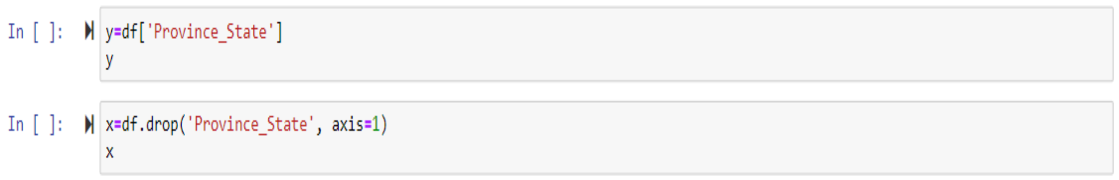
Power Transformer is a way to algorithmically correct problems, we can use yeo- Johnson Transformation it can be used on both positive and negative values as below.



```
from sklearn.preprocessing import power_transform
x=power_transform(x,method='yeo-johnson')
```

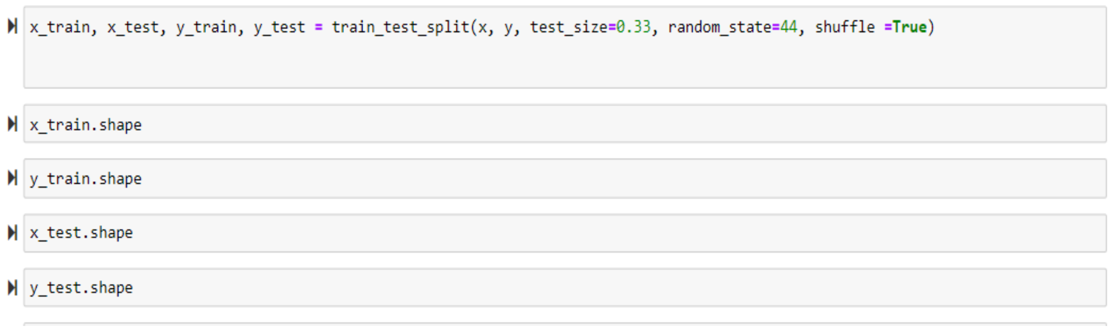
Building Machine Learning Models

We then separate the features and the dependent variable into variables x and y respectively. Because our data is all numbers and there's no text in it.



```
In [ ]:  y=df['Province_State']
         y

In [ ]:  x=df.drop('Province_State', axis=1)
         x
```

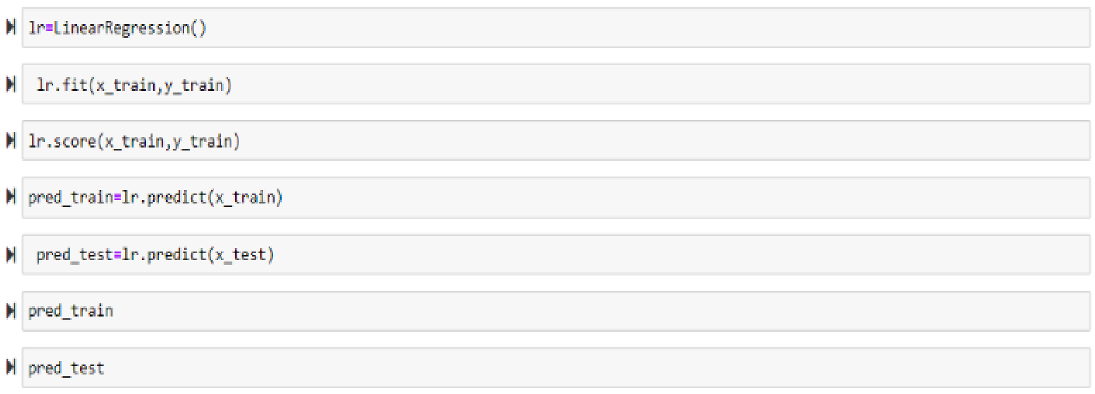
So as soon as we separate the features and the dependent variable, we can split our data into training and test sets. The code for doing this is given below:



```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.33, random_state=44, shuffle =True)

x_train.shape

y_train.shape

x_test.shape

y_test.shape
```

The next step for us is to simply create a linear regression object, fit it to our training set, and start predicting.

We can predict Test data and Train data and then made a model based on observations.



```
lr=LinearRegression()
```

```
 lr.fit(x_train,y_train)
```

```
lr.score(x_train,y_train)
```

```
pred_train=lr.predict(x_train)
```

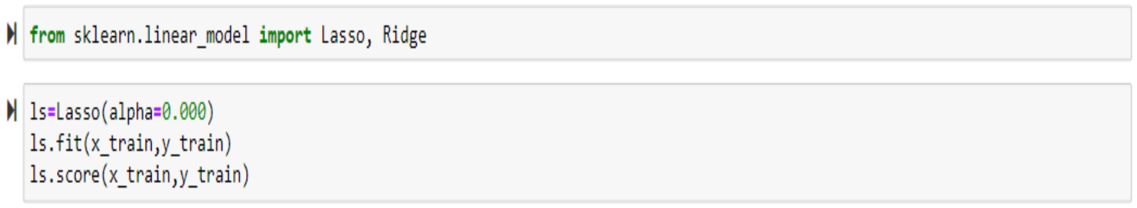
```
 pred_test=lr.predict(x_test)
```

```
pred_train
```

```
pred_test
```

We are using here Lasso regression here, as it has least errors.

Lasso Regression is a popular type of regularized linear regression which includes an L1 penalty. This has the effect of shrinking the coefficients for those input variables that do not contribute much to the prediction task. This penalty allows most coefficient values to go to the value equal to zero, allowing input variables to be removed from the model, providing a type of automatic feature selection.



```
from sklearn.linear_model import Lasso, Ridge
```

```
ls=Lasso(alpha=0.000)
ls.fit(x_train,y_train)
ls.score(x_train,y_train)
```

Concluding Remarks

In the entire data there is a low death rate. We have found spike in the Death rate only where there is a recovered cases are also more. So we can conlude that the provinces in USA has a maximum death and recovered data rate in Newyork, this might be because of population, there were so many parameters which can also be observed i.e like hospitalization rate , active cases and many more can be target variables. But as for simple understanding and using Linear Regression , Power Transformation. We can perform and made model on the required dataset.