Data cleaning

Clean your dataset (remove missing values, sanitize data, etc.). Remove any outliers (except 0s) using the Tukey's rule from class using the default values as in class. Report what you found (number of outliers). Comment on your findings both for data cleaning (what issues you found, how you dealt with them) and outlier detection. MO is Missouri ,MI is Michigan

```
import pandas as pd
import numpy as np
import datetime
import math
from math import *
import matplotlib.pyplot as plt
from scipy.stats import gamma
import seaborn as sns
from scipy.stats import poisson, geom, binom
```

```
[211] from google.colab import drive
      drive.mount('/content/gdrive')
      %cd /content/gdrive/My Drive/Colab Notebooks

      Drive already mounted at /content/gdrive; to attempt to forcibly remount, call drive.mount("/content/gdrive", force_remount=True).
      /content/gdrive/My Drive/Colab Notebooks
```

```
[212]
      df = pd.read_csv("./Cases.csv")
      # Getting total data based on State, MICHIGAN, MISSOURI
      df['conf_cases'] = df['conf_cases'].fillna(0)
      df['prob_cases'] = df['prob_cases'].fillna(0)
      df['pnew_case'] = df['pnew_case'].fillna(0)
      df['conf_death'] = df['conf_death'].fillna(0)
      df['prob_death'] = df['prob_death'].fillna(0)
      df['pnew_death'] = df['pnew_death'].fillna(0)
```

```
[213] df_mo = df[df['state'] == "MO"]
      df_mi = df[df['state'] == "MI"]
      df_mo["submission_date"] = pd.to_datetime(df_mo["submission_date"])
      df_mi["submission_date"] = pd.to_datetime(df_mi["submission_date"])
      df_mi.sort_values(by='submission_date', inplace=True)
      df_mo.sort_values(by='submission_date', inplace=True)
      df_mo = df_mo[['submission_date', 'new_case', 'new_death']]
      df_mi = df_mi[['submission_date', 'new_case', 'new_death']]
```

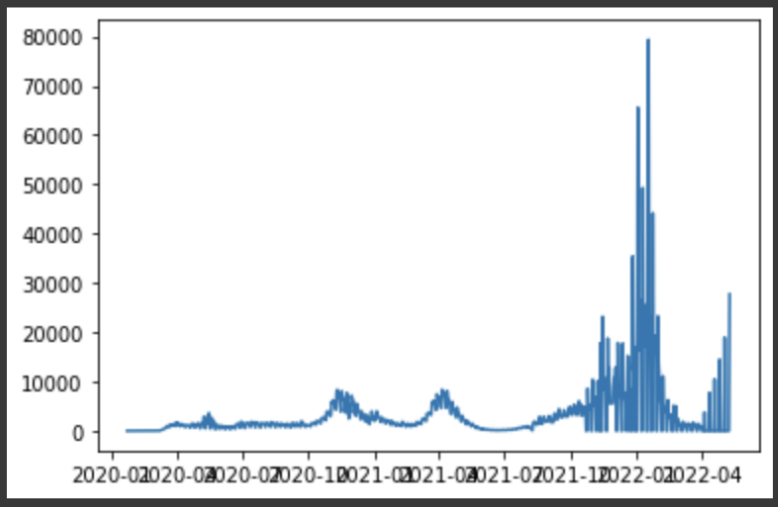
```
frames = [df_mi, df_mo]
pd.merge(df_mi, df_mo, on='submission_date')
total_data.rename(
'submission_date':'Date','new_case_x': 'new_case_MO', 'new_case_y': 'new_case_MI','new_death_x': 'new_death_MO',
'new_death_y': 'new_death_MI'})
'submission_date"] = pd.to_datetime(total_data["submission_date"])
# total_data.sort_values(by=['state', 'submission_date'], inplace=True)
total_data.to_csv('out.csv', encoding='utf-8', index=False)
print(total_data.shape)
# print(total_data.dtypes)
# print("\nGiving a space")
total_data["Year"] = pd.Series(total_data["Date"]).apply(lambda x: x.year)
total_data["Month"] = pd.Series(total_data["Date"]).apply(lambda x: x.month)
print(total_data.info())
total_data = total_data.reindex(columns=['Date','new_case_MO','new_case_MI','new_death_MO','new_death_MI','Year','Month'])
print("Starting and ending date is : \n" + str(total_data['Date'].min()) + "Ending is : " + str(
    total_data['Date'].max()))
print("**************Before Deleting Outlier Data*********************************")
```

Run cell (⌘/Ctrl+Enter)
cell executed since last change

executed by Karthik Vegesana
16:30 (5 minutes ago)
executed in 0.168 s

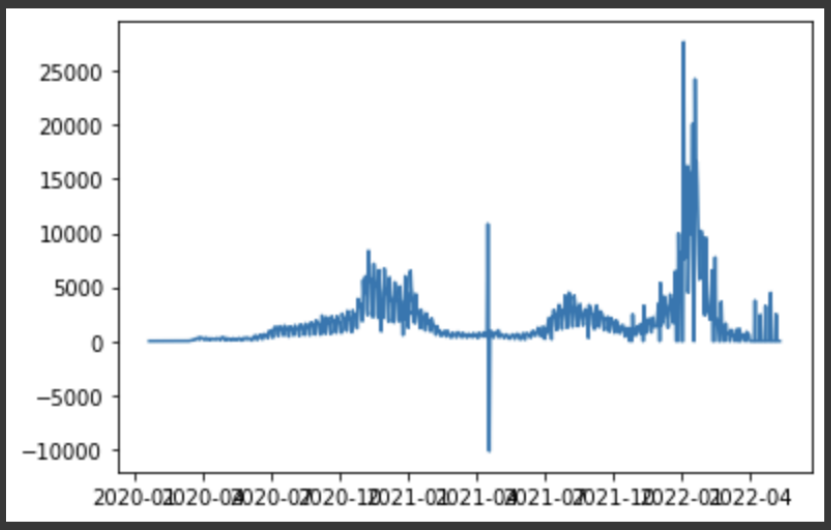
```
(841, 5)
<class 'pandas.core.frame.DataFrame'>
Int64Index: 841 entries, 0 to 840
Data columns (total 7 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Date           841 non-null    datetime64[ns]
 1   new_case_MO    841 non-null    int64
 2   new_death_MO   841 non-null    int64
 3   new_case_MI    841 non-null    int64
 4   new_death_MI   841 non-null    int64
 5   Year           841 non-null    int64
 6   Month          841 non-null    int64
dtypes: datetime64[ns](1), int64(6)
memory usage: 52.6 KB
None
Starting and ending date is :
2020-01-22 00:00:00Ending is : 2022-05-11 00:00:00
**************Before Deleting Outlier Data*********************************
```

```
[215] plt.plot("Date", "new_case_MO", data=total_data)
      plt.show()
```

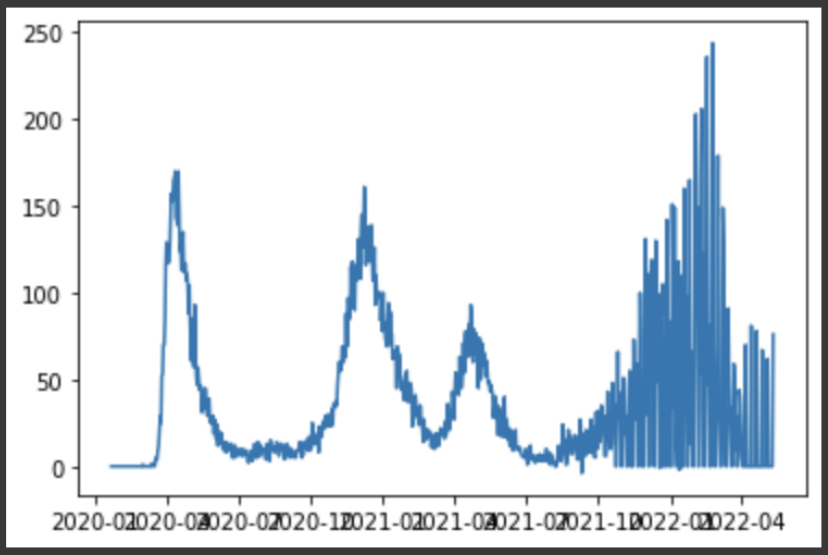


Loading...

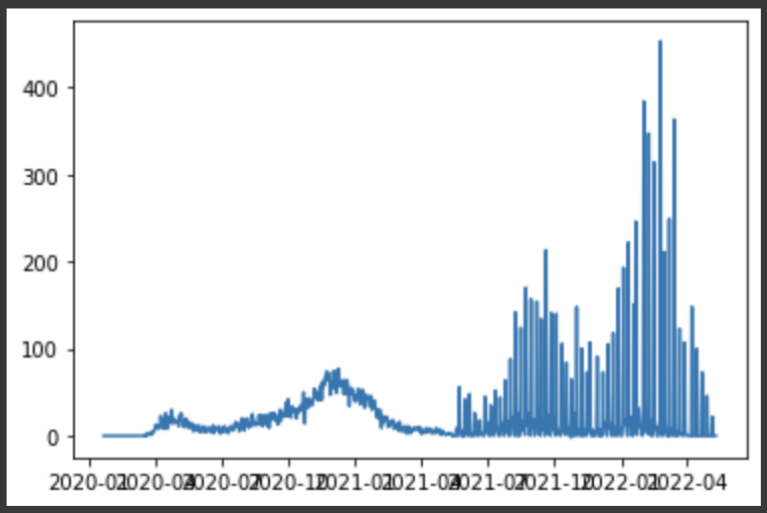
```
plt.plot("Date", "new_case_MI", data=total_data)
plt.show()
```




```
[217] plt.plot("Date", "new_death_MO", data=total_data)
      plt.show()
```



```
plt.plot("Date", "new_death_MI", data=total_data)
plt.show()
```




```
def detect_outliers(data, numerical_attributes):
    outlier_indices = []

    for col in numerical_attributes:
        Q1 = np.percentile(data[col], 25)
        Q3 = np.percentile(data[col], 75)
        IQR = Q3 - Q1
        outlier_step = 1.5 * IQR

        outlier_list_col = sorted(
            data[((data[col] < Q1 - outlier_step) | (data[col] > Q3 + outlier_step)) & data[col] != 0].index)
        # append the found outlier indices for col to the list of outlier indices
        outlier_indices.extend(outlier_list_col)
        print("Outliers in Column " + str(col) + " : " + str(len(outlier_list_col)))
    outlier_indices = list(set(outlier_indices))

    return outlier_indices
```

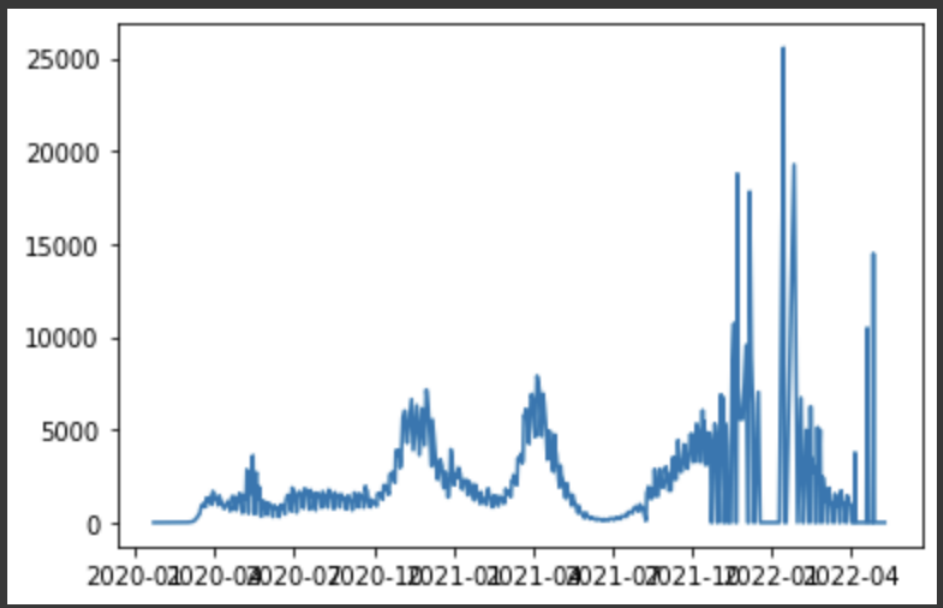
```
[220]

numerical_attributes = ['new_case_MO', 'new_case_MI', 'new_death_MO', 'new_death_MI']
outliers_in_data = detect_outliers(total_data, numerical_attributes)
print("These are the number of rows that have Outliers when 0 is not considered to be an outlier:", len(outliers_in_data))
```

```
Outliers in Column new_case_MO : 41
Outliers in Column new_case_MI : 30
Outliers in Column new_death_MO : 26
Outliers in Column new_death_MI : 37
These are the number of rows that have Outliers when 0 is not considered to be an outlier: 114
```

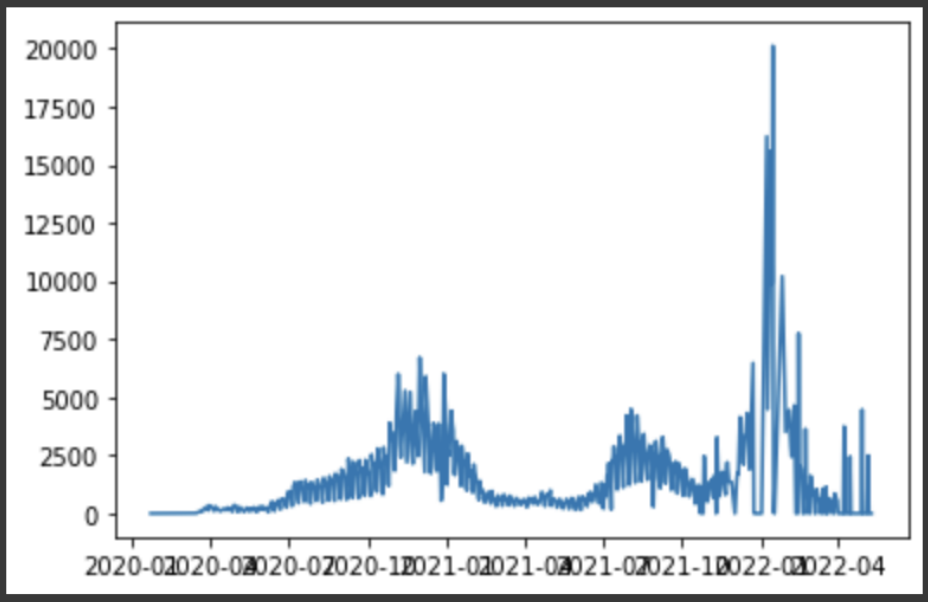
```
new_data = total_data.copy()

total_data= total_data[~total_data.index.isin(outliers_in_data)]

print("Dataset information after removing the outliers: \n")
print(total_data.describe())
print("\n")
print("\n")
print("*************** After Removing Outlier Data**************************")
```

```
Dataset information after removing the outliers:

        new_case_MO   new_case_MI  new_death_MO  new_death_MI         Year  \
count    727.000000    727.000000    727.000000    727.000000   727.000000
mean    1866.863824   1227.821183     28.159560     14.433287  2020.711142
std     2410.623563   1651.348478     35.069775     23.164774     0.681656
min        0.000000      0.000000     -4.000000     -2.000000  2020.000000
25%      340.000000    241.500000      5.000000      1.000000  2020.000000
50%     1284.000000    738.000000     14.000000      8.000000  2021.000000
75%     2333.000000   1745.000000     40.000000     18.000000  2021.000000
max    25560.000000  20116.000000    244.000000    314.000000  2022.000000

            Month
count  727.000000
mean     5.932600
std      3.215342
min      1.000000
25%      3.000000
50%      6.000000
75%      9.000000
max     12.000000



*************** After Removing Outlier Data**************************
```

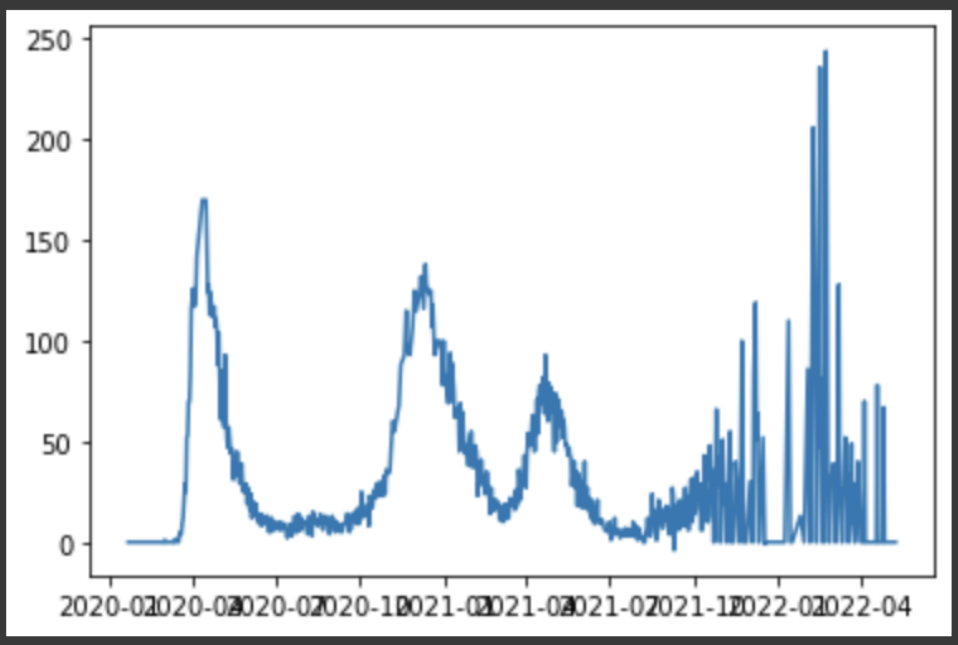
```
plt.plot("Date", "new_case_MO", data=total_data)
plt.show()
```



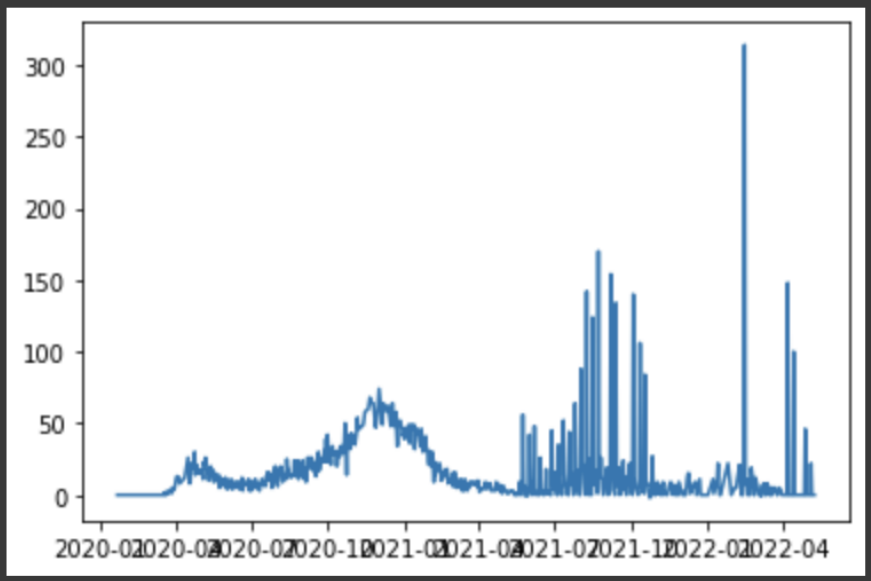
```
[223] plt.plot("Date", "new_case_MI", data=total_data)
      plt.show()
```



```
plt.plot("Date", "new_death_MO", data=total_data)
plt.show()
```




```
plt.plot("Date", "new_death_MI", data=total_data)
plt.show()
```



Mandatory tasks

Mandatory task 2A

```
[226] #Considering data only for February and March months for year 2021
      # total_data = new_data
      feb_cases = total_data[(total_data.Date > '2021-01-31') & (total_data.Date < '2021-03-01')]

      feb_mo_cases = feb_cases['new_case_MO'].to_numpy()
      feb_mo_deaths = feb_cases['new_death_MO'].to_numpy()

      feb_mi_cases = feb_cases['new_case_MI'].to_numpy()
      feb_mi_deaths = feb_cases['new_death_MI'].to_numpy()



      march_cases = total_data[(total_data.Date > '2021-02-28') & (total_data.Date <= '2021-03-31')]
      # print(march_cases)

      march_mo_cases = march_cases['new_case_MO'].to_numpy()
      march_mo_deaths = march_cases['new_death_MO'].to_numpy()

      march_mi_cases = march_cases['new_case_MI'].to_numpy()
      march_mi_deaths = march_cases['new_death_MI'].to_numpy()
      print("Number of data points for February month: "+str(len(feb_cases)))
      print("Number of data points for March month: "+str(len(march_cases)))
      n = total_data.shape[0]
      print(n)

      Number of data points for February month: 28
      Number of data points for March month: 30
      727
```

```
[227] true_std_conf_MO = np.sqrt(
          (1 / (n)) * sum(np.square(total_data["new_case_MO"] - total_data["new_case_MO"].mean()))
      )
      true_std_conf_MI = np.sqrt(
          (1 / (n)) * sum(np.square(total_data["new_case_MO"] - total_data["new_case_MI"].mean()))
      )
      true_std_death_MO = np.sqrt(
          (1 / (n)) * sum(np.square(total_data["new_case_MO"] - total_data["new_death_MO"].mean()))
      )
      true_std_death_MI = np.sqrt(
          (1 / (n)) * sum(np.square(total_data["new_case_MO"] - total_data["new_death_MI"].mean()))
      )

      print(true_std_conf_MO)
      print(true_std_conf_MI)
      print(true_std_death_MO)
      print(true_std_death_MI)

      2408.9650672744865
      2492.2857364982747
      3030.502609496441
      3038.8503727109687
```

Walds one Sample Test

Procedure:

We have calculated the W statistic and compared it with the threshold value of z_alpha/2 = 1.96. The estimator is calculated using MLE of March month's mean(Since for Poisson-distributed data, MLE estimator is lamda_hat which is equal to sample mean). The guess of the estimator is February month's mean. The standard error of the estimator is calculated in below walds function.

Null Hypothesis is (H0):

Mean of February 2021's cases/deaths = Mean of March 2021's cases/deaths.

Alternate Hypothesis is (H1):

Mean of February 2021's cases/deaths is not equal to Mean of March 2021's cases/deaths

```
[228] #Defining Wald's test
      #Assuming poisson's distribution

      def walds_test(feb, march):

          feb_mean = feb.mean()
          march_mean = march.mean()
          print("February Month Mean: "+str(feb_mean))
          print("March Month Mean: "+str(march_mean))
          #Variance and mean are equal for poisson distribution
          standard_error = math.sqrt(march_mean/len(march))
          #computing the W statistic
          W = np.abs((march_mean - feb_mean)/standard_error)
          Z_alpha = 1.96
          print("Walds w-statistics :")
          print(str(W) + "\n")
          if(W <= Z_alpha):
              return "Accept Ho since the value of W statistic "+ str(W)+" is less than threshold value 1.96"
          else:
              return "Reject Ho since the value of W statistic "+ str(W)+" is greater than threshold value 1.96"
```

.

```
0s
print("Procedure:")
print("")
print("Results for every column:")
print("Missouri Confirmed Cases")
print(walds_test(feb_mo_cases,march_mo_cases))
print("\n")
print("Missouri Deaths")
print(walds_test(feb_mo_deaths,march_mo_deaths))
print("\n")
print("Michigan Confirmed Cases")
print(walds_test(feb_mi_cases,march_mi_cases))
print("\n")
print("Michigan Deaths")
print(walds_test(feb_mi_deaths,march_mi_deaths))
print("\n")
```

```
Procedure:

Results for every column:
Missouri Confirmed Cases
February Month Mean: 1247.0714285714287
March Month Mean: 3430.4333333333334
Walds w-statistics :
204.17937212740563

Reject Ho since the value of W statistic 204.17937212740563 is greater than threshold value 1.96


Missouri Deaths
February Month Mean: 29.285714285714285
March Month Mean: 20.933333333333334
Walds w-statistics :
9.998888538258635

Reject Ho since the value of W statistic 9.998888538258635 is greater than threshold value 1.96


Michigan Confirmed Cases
February Month Mean: 881.7857142857143
March Month Mean: 543.7333333333333
Walds w-statistics :
79.40569429496196

Reject Ho since the value of W statistic 79.40569429496196 is greater than threshold value 1.96


Michigan Deaths
February Month Mean: 16.535714285714285
March Month Mean: 7.533333333333333
Walds w-statistics :
17.964884199313513

Reject Ho since the value of W statistic 17.964884199313513 is greater than threshold value 1.96
```

Observation:

The W values returned here are quite high. We cannot conclude that mean of March month is equal to that of February 2021.

Is the Test Applicable?

The Wald's test is not applicable here because it assumes that the estimator is asymptotically normal. And the number of datapoints do not tend to infinity.

Z Test

Procedure:

We have calculated the Z statistic and compared it with the threshold value of z_alpha/2 = 1.96. We have used corrected sample standard deviation of entire dataset as true variance.

Therefore the ddof value for np.var() is set to 1. The MLE for March month's mean is calculated and February month's mean is used as a guess value.

Null Hypothesis is (H0):
Mean of February 2021's cases/deaths = Mean of March 2021's cases/deaths.

Alternate Hypothesis is (H1):
Mean of February 2021's cases/deaths is not equal to Mean of March 2021's cases/deaths.

```
#Defining the Z test

def z_test(feb,march,true_dev):
    feb_mean = feb.mean()
    march_mean = march.mean()
    print("February Month Mean: "+str(feb_mean))
    print("March Month Mean: "+str(march_mean))

    standard_error = true_dev/math.sqrt(len(march))
    #computing the Z statistic
    Z = np.abs((march_mean - feb_mean)/standard_error)
    Z_alpha = 1.96
    if(Z <= Z_alpha):
        return "Accept Ho since Z value is : "+ str(Z)
    else:
        return "Reject Ho since Z value is : "+ str(Z)
```

```
[231] print("Results:\n")
      print("Missouri Confirmed Cases")
      print(z_test(feb_mo_cases,march_mo_cases,true_std_conf_MO))
      print("\n")
      print("Missouri Deaths")
      print(z_test(feb_mo_deaths,march_mo_deaths,true_std_death_MO))
      print("\n")
      print("Michigan Confirmed Cases")
      print(z_test(feb_mi_cases,march_mi_cases,true_std_conf_MI))
      print("\n")
      print("Michigan Deaths")
      print(z_test(feb_mi_deaths,march_mi_deaths,true_std_death_MI))
      print("\n")
```

```
Results:

Missouri Confirmed Cases
February Month Mean: 1247.0714285714287
March Month Mean: 3430.4333333333334
Reject Ho since Z value is : 4.964275251149911


Missouri Deaths
February Month Mean: 29.285714285714285
March Month Mean: 20.933333333333334
Accept Ho since Z value is : 0.015095804379642795


Michigan Confirmed Cases
February Month Mean: 881.7857142857143
March Month Mean: 543.7333333333333
Accept Ho since Z value is : 0.7429281159635485


Michigan Deaths
February Month Mean: 16.535714285714285
March Month Mean: 7.533333333333333
Accept Ho since Z value is : 0.016225896355914024
```

Is the Test Applicable?

The main assumptions of Z-test are either the sample data has to be normally distributed or the sample size should be large. However, both of them are not true in our case. Also, we should know the value of true variance. Therefore the Z-test is not applicable here.

One Sample T-test

Procedure:

We have calculated the T statistic and compared it with the threshold value which we checked in the online table for alpha/2 = 0.025 and degree of freedom as 29 = 2.04523. The estimator is calculated using MLE of March month's mean(Since for Poisson-distributed data, MLE estimator is lamda_hat which is equal to sample mean). The guess of the estimator is February month's mean. We have also calculated sample standard deviation in the below function.

Null Hypothesis is (H0):
Mean of February 2021's cases/deaths = Mean of March 2021's cases/deaths.

Alternate Hypothesis is (H1):
Mean of February 2021's cases/deaths is not equal to Mean of March 2021's cases/deaths.

```
#T test

def t_test(feb,march):
    feb_mean = feb.mean()
    march_mean = march.mean()

    # sample_standard_deviation = np.sqrt(np.sum(np.square(march - march_mean))/(len(march)))
    sample_standard_deviation = np.sqrt((1 / (len(march - 1))) * sum(np.square(march - march_mean)))
    denominator = sample_standard_deviation/math.sqrt(len(march))
    #computing the T statistic
    T = np.abs((march_mean - feb_mean)/denominator)
    T_alpha = 2.04523
    if(T <= T_alpha):
        return "Accept Ho since T value is "+ str(T)
    else:
        return "Reject Ho since T value is "+ str(T)
```

```
[233] print("The number of data points for the March month is : " +str(len(march_cases)))
      print("Therefore the threshold value for T test for degree of freedom 29 and alpha/2 = 0.025 is 2.04523")
      print("Missouri Confirmed Cases")
      print(t_test(feb_mo_cases,march_mo_cases))
      print("\n")
      print("Missouri Deaths")
      print(t_test(feb_mo_deaths,march_mo_deaths))
      print("\n")
      print("Michigan Confirmed Cases")
      print(t_test(feb_mi_cases,march_mi_cases))
      print("\n")
      print("Michigan Deaths")
      print(t_test(feb_mi_deaths,march_mi_deaths))
      print("\n")
```

```
The number of data points for the March month is : 30
Therefore the threshold value for T test for degree of freedom 29 and alpha/2 = 0.025 is 2.04523
Missouri Confirmed Cases
Reject Ho since T value is 6.903600353750057


Missouri Deaths
Reject Ho since T value is 5.763893749279891


Michigan Confirmed Cases
Reject Ho since T value is 13.657101473555631


Michigan Deaths
Reject Ho since T value is 17.23782149635006
```

Is the Test Applicable?

Even though the sample size is small, T test is not applicable here because the data does not follow normal distribution.

2 Sample Wald's Test

Procedure:

Similar to one sampled Wald's test, We have calculated the W statistic and compared it with the threshold value of z_alpha/2 = 1.96. The guess value theta0 is 0. And the difference between mean is difference between the sample means since we know that the data is Poisson distributed. We have also computed standard error for both months as shown below.

Null Hypothesis is (H0):
The difference between mean of February 2021 and March 2021 is 0.

Alternate Hypothesis is (H1):
The difference between mean of February 2021 and March 2021 is not 0.

```
#2 SAMPLE TESTS

#Walds Test
#null hypothesis is that both the means are equal, so theta0 = 0


def walds_test_2(feb,march):
    feb_mean = feb.mean()
    march_mean = march.mean()

    standard_error = np.sqrt((feb_mean/len(feb)) + march_mean/len(march))
    #w value of waltz
    W = np.abs((feb_mean - march_mean)/standard_error)
    Z_alpha = 1.96
    if(W <= Z_alpha):
        return "Accept Ho since the value of W statistic "+ str(W)+" is less than threshold value 1.96"
    else:
        return "Reject Ho since the value of W statistic "+ str(W)+" is greater than threshold value 1.96"
```

```
[235] print("Missouri Confirmed Cases")
    print(walds_test_2(feb_mo_cases,march_mo_cases))
    print("\n")
    print("Missouri Deaths")
    print(walds_test_2(feb_mo_deaths,march_mo_deaths))
    print("\n")
    print("Michigan Confirmed Cases")
    print(walds_test_2(feb_mi_cases,march_mi_cases))
    print("\n")
    print("Michigan Deaths")
    print(walds_test_2(feb_mi_deaths,march_mi_deaths))
    print("\n")
```

```
Missouri Confirmed Cases
Reject Ho since the value of W statistic 173.2139453002994 is greater than threshold value 1.96


Missouri Deaths
Reject Ho since the value of W statistic 6.325209150992754 is greater than threshold value 1.96


Michigan Confirmed Cases
Reject Ho since the value of W statistic 47.99208975283848 is greater than threshold value 1.96


Michigan Deaths
Reject Ho since the value of W statistic 9.812639876251579 is greater than threshold value 1.96
```

Is the Test Applicable?

The two sample Wald's test is not applicable here because it assumes that the estimator is asymptotically normal. Both the estimators here are not asymptotically normal. The number of data samples do not tend to infinity.

Unpaired T test

Procedure:

We have calculated the T statistic and compared it with the threshold value which we checked in the online table for alpha/2 = 0.025 and degree of freedom as (n+m - 2) = 2.30687. We have also calculated sample pooled standard deviation required for unpaired T test.

Null Hypothesis is (H0):
The difference between mean of February 2021 and March 2021 is 0.

Alternate Hypothesis is (H1):
The difference between mean of February 2021 and March 2021 is not 0.

```
#Unpaired T test

def t_test_2(feb,march):
    feb_mean = feb.mean()
    march_mean = march.mean()
    D = feb_mean - march_mean

    standard_deviation_feb_square = (1/(len(feb)-1))*np.sum(np.square(feb-feb_mean))
    standard_deviation_march_square = (1/(len(march)-1))*np.sum(np.square(march-march_mean))

    sample_pooled_standard_deviation = np.sqrt((standard_deviation_feb_square/len(feb)) + (standard_deviation_march_square/len(march)))

    #Calculating the T statistic:
    T = np.abs(D/sample_pooled_standard_deviation)
    T_alpha = 2.30687
    if(T <= T_alpha):
       return "Accept Ho since the value of T statistic "+ str(T)+" is less than threshold value 2.30687"
    else:
        return "Reject Ho since the value of W statistic "+ str(T)+" is greater than threshold value 2.30687"
```

```
[237] print("Results:\n")
    print("Missouri Confirmed Cases")
    print(t_test_2(feb_mo_cases,march_mo_cases))
    print("\n")
    print("Missouri Deaths")
    print(t_test_2(feb_mo_deaths,march_mo_deaths))
    print("\n")
    print("Michigan Confirmed Cases")
    print(t_test_2(feb_mi_cases,march_mi_cases))
    print("\n")
    print("Michigan Deaths")
    print(t_test_2(feb_mi_deaths,march_mi_deaths))
```

```
Results:

Missouri Confirmed Cases
Reject Ho since the value of W statistic 6.713277826100857 is greater than threshold value 2.30687


Missouri Deaths
Reject Ho since the value of W statistic 3.4732331223916315 is greater than threshold value 2.30687


Michigan Confirmed Cases
Reject Ho since the value of W statistic 3.85686572489355 is greater than threshold value 2.30687


Michigan Deaths
Reject Ho since the value of W statistic 7.343079178240679 is greater than threshold value 2.30687
```

Is the Test Applicable?

As discussed for one sample T test, the two sample T test is also not applicable here because the data does not follow a normal distribution.

Task 2B

We have to check the equality of distribution for deaths and number of confirmed cases of MO(Missouri) and MI(Michigan). We have selected the data between Oct. and Dec. For the problem our null hypothesis for deaths and confirmed cases.

H_0 -> MO has same distribution as MI

Results

1)
One sample KS test

For number of confirmed cases

a) In Poisson distribution equality the null hypothesis was accepted.

b) In Geometric distribution equality the null hypothesis was accepted.

c) In Binomial distribution equality the null hypothesis was rejected.

For number of deaths

d) In Poisson distribution equality the null hypothesis was accepted.

e) In Geometric distribution equality the null hypothesis was accepted.

f)In Binomial distribution equality the null hypothesis was accepted.

2) Two sample KS Test

For number of confirmed cases

a)Null hypothesis was rejected.

For number of deaths

b) Null hypothesis was rejected.

3)Permutation test:

For number of confirmed cases

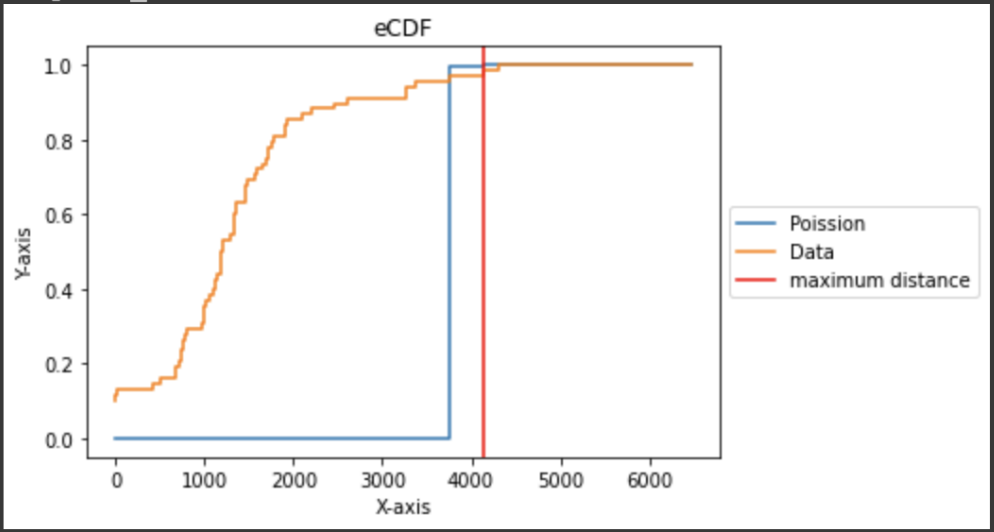
a)Null hypothesis was rejected.

For number of deaths

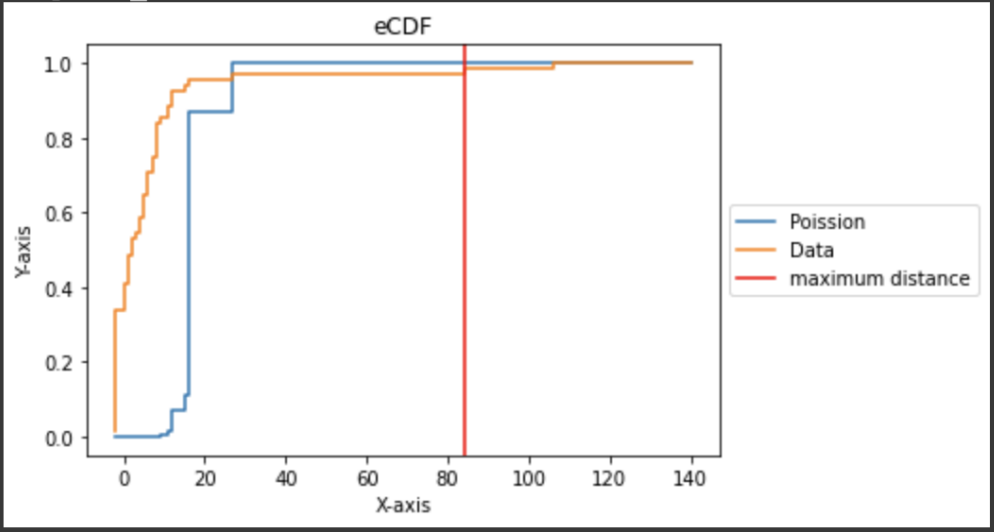
b) Null hypothesis was rejected.

Output

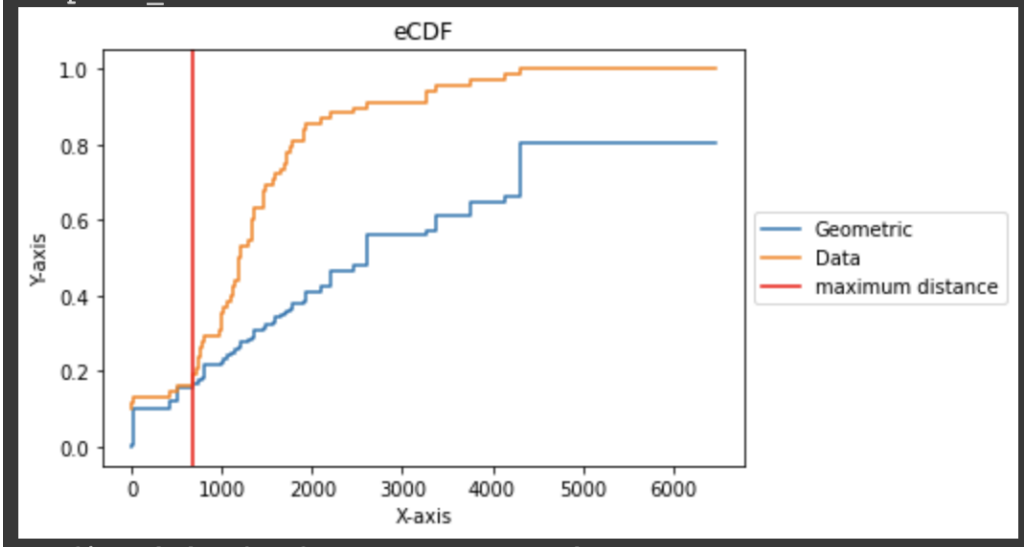
```
#########Poission destribution#########
Confirmed cases for MO(State1) and MI(State2)
lambda is:  3973.205882352941
Maximum distance is:  0.04032134504937168
Accpet H_0
```



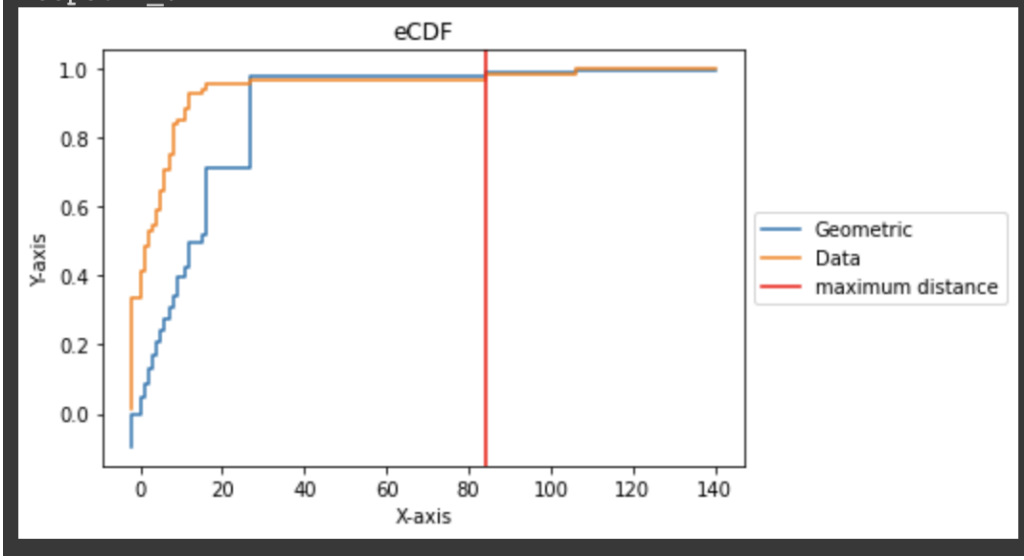

```
Confirmed deaths for MO(State1) and MI(State2)
lambda is:  22.220588235294116
Maximum distance is:  0.04411764705882382
Accpet H_0
```



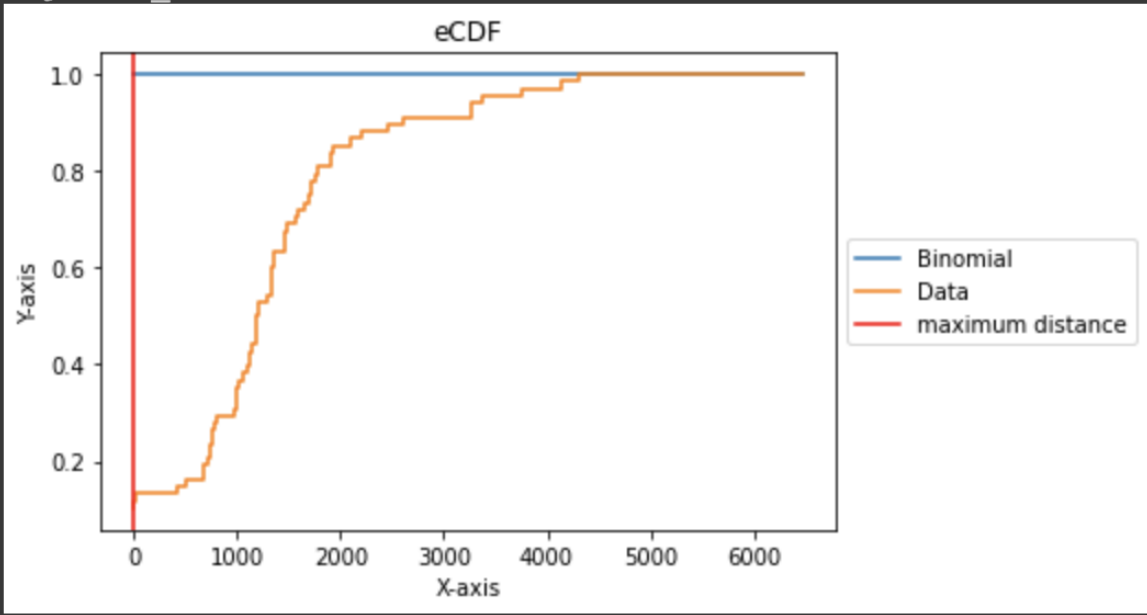
```
#########Geometric destribution#########
Confirmed cases for MO(State1) and MI(State2)
p_mme is  0.0002516859255749913
Maximum distance is:  0.00728627808293783
Accpet H_0
```



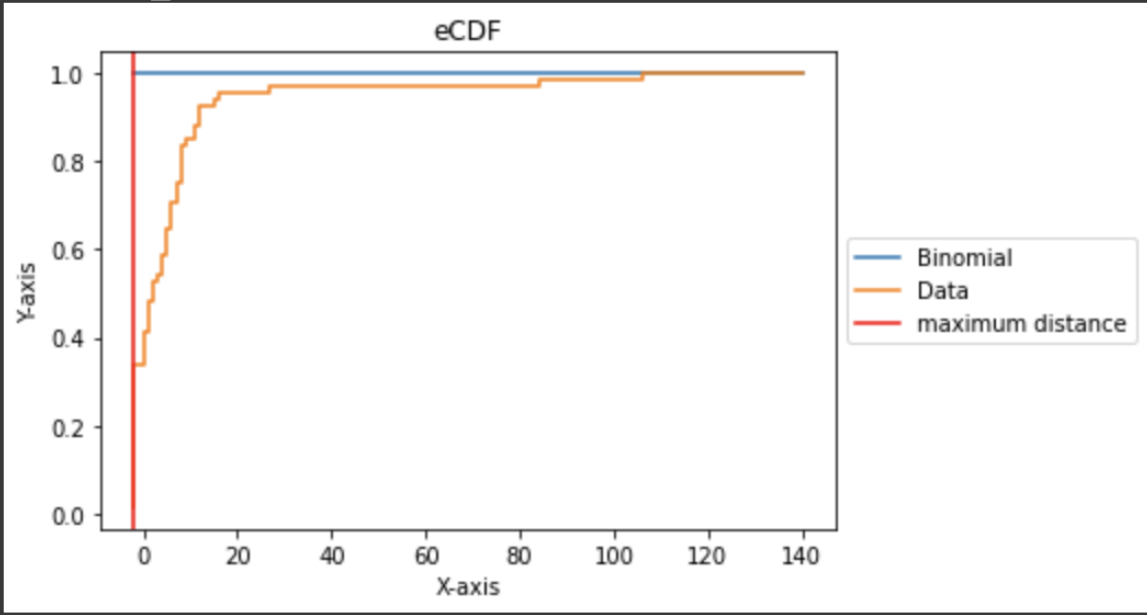

```
Confirmed deaths for MO(State1) and MI(State2)
p_mme is  0.045003309066843154
Maximum distance is:  0.02321714072706682
Accpet H_0
```



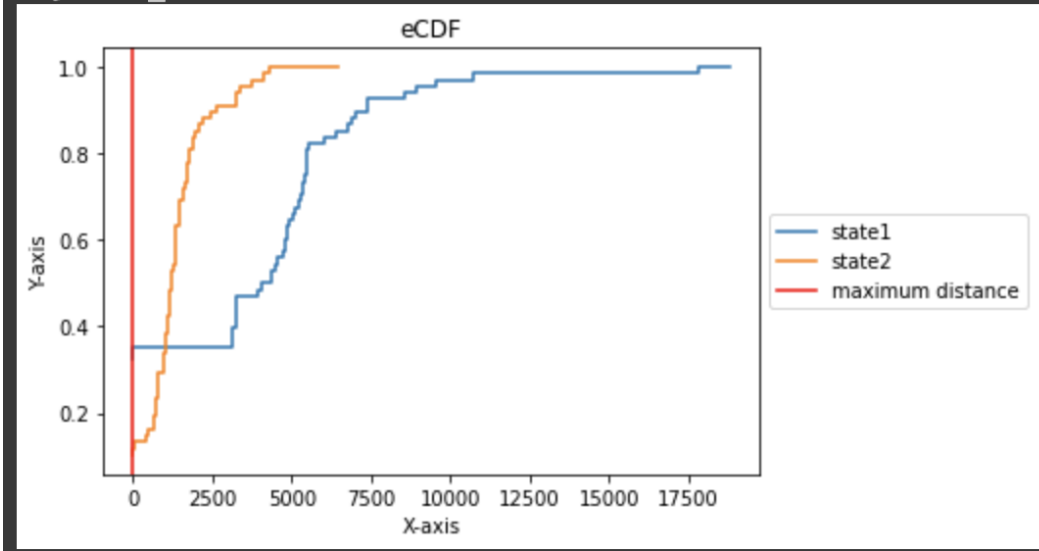
```
#########Binomial destribution#########
Confirmed cases for MO(State1) and MI(State2)
p_mme is  -3601.2680644524985
n_mme is  -1.103279681279985
Maximum distance is:  1.0
Reject H_0
```



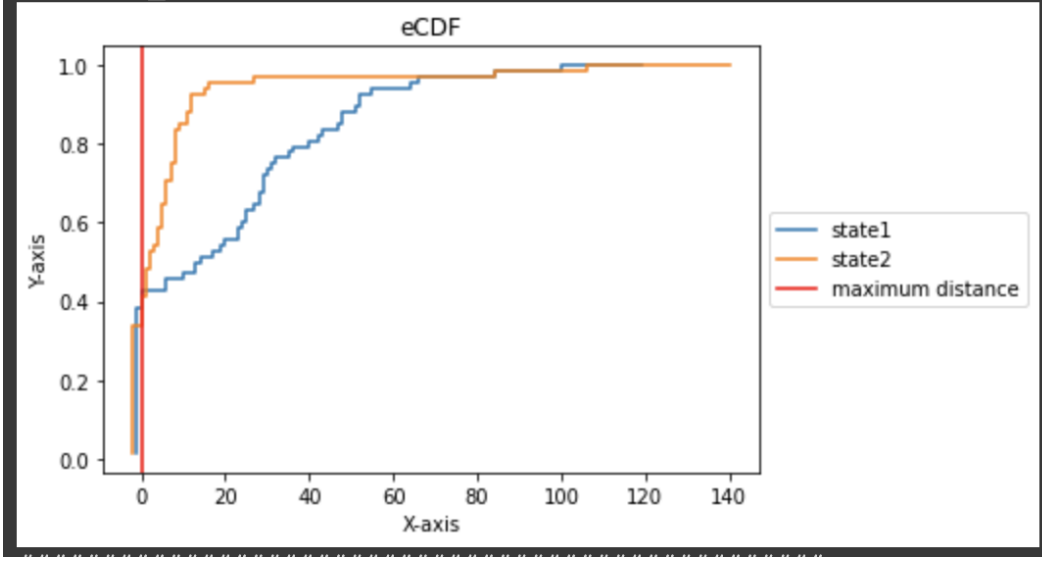

```
Confirmed deaths for MO(State1) and MI(State2)
p_mme is  -28.86875656947093
n_mme is  -0.7697106102170215
Maximum distance is:  nan
Accpet H_0
```



```
###### Two Sample KS Test ######

Confirmed cases for MO(State1) and MI(State2)
Maximum distance is:  0.6617647058823529
Reject H_0
```




```
##################################################
Confirmed deaths for MO(State1) and MI(State2)
Maximum distance is:  0.8088235294117647
Reject H_0
```




```
##################################################
```

```
###### Permutation test ######

Confirmed cases for MO(State1) and MI(State2)
p-value =  0.0
Reject H_0
#####################################################
Confirmed deaths for MO(State1) and MI(State2)
p-value =  0.002
Reject H_0
#####################################################
```

Mandatory Task 2d: Vaccine Prediction

Mandatory Task 2e: Paired T-tests
H_0: $\mu_1 = \mu_2$, H_1: $\mu_1 \neq \mu_2$, test assumed to be applicable per question
$t_{0.05/2,\ 30-1} = 2.045$

Observations: The T statistics obtained in the first two parts were very high due to the large population difference. In any given month, it is likely that Michigan would administer many more vaccines than Missouri. So, to account for this population difference we also performed the paired T-tests in the number of vaccines distributed per 100k people as well. Even then, the T statistics were very high. This is likely due in part to the fact that the percentage of vaccinated people is higher in Michigan compared to Missouri.

Exploratory tasks

For the explanatory part, for our X dataset we have taken the hospitalization data for the states of Michigan and Missouri. Our aim is to check correlation and trends between:

1. Covid Cases increase and Hospitalization demands

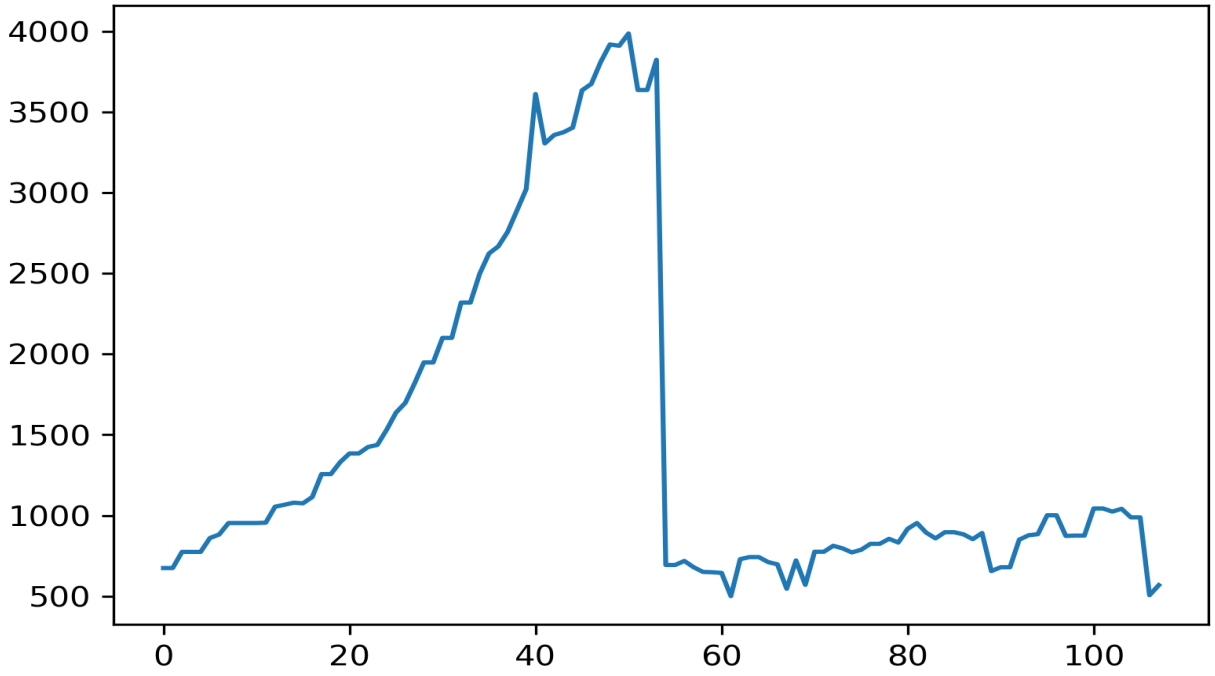


Fig1: Hospitalizations in Michigan and Missouri from 2020-04-10 - 2020-06-02

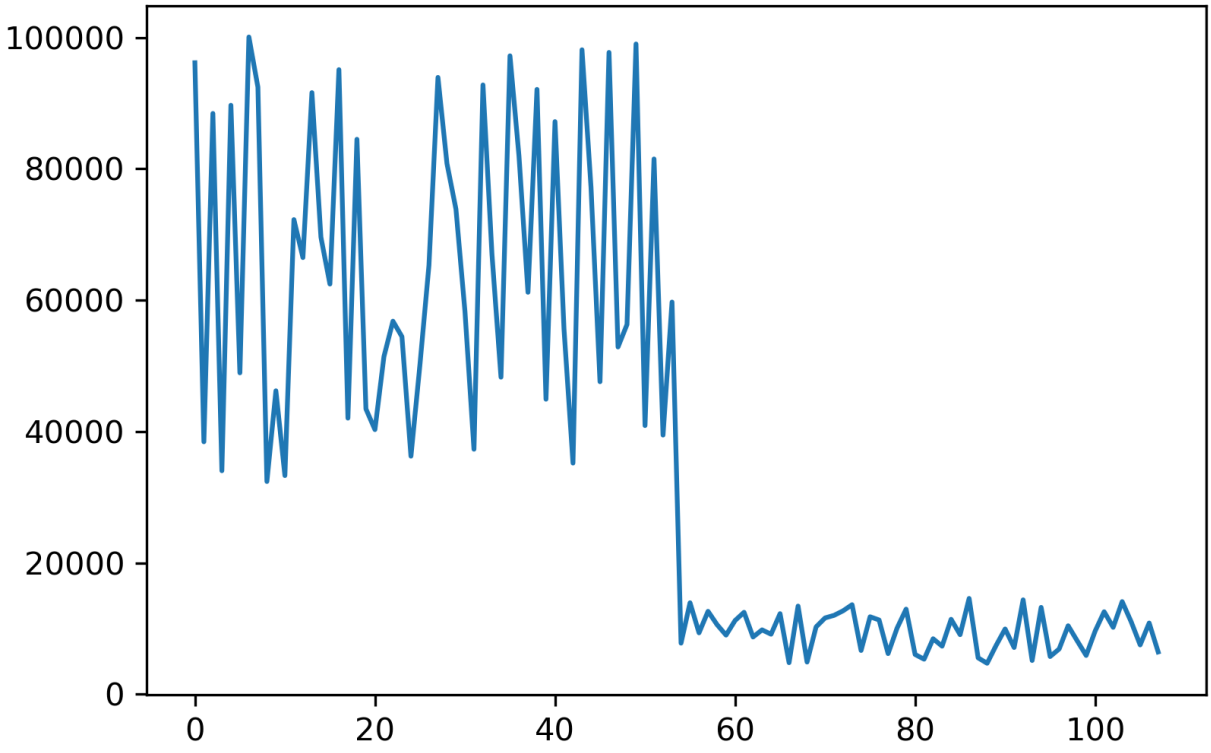


Fig2: COVID-19 Cases in Michigan and Missouri from 2020-04-10 - 2020-06-02

2. Vaccine Distribution and how Hospitalization demands were impacted

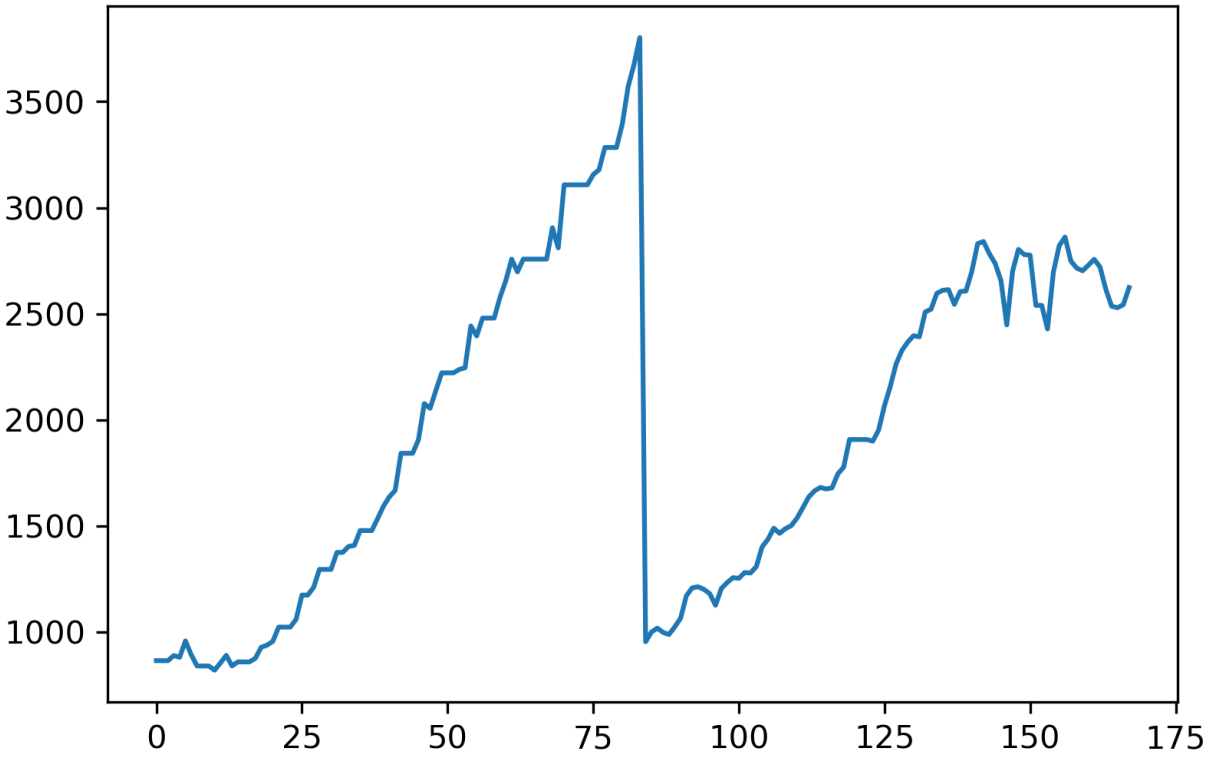


Fig3: Hospitalizations in Michigan and Missouri from 2020-12-14 - 2021-03-7

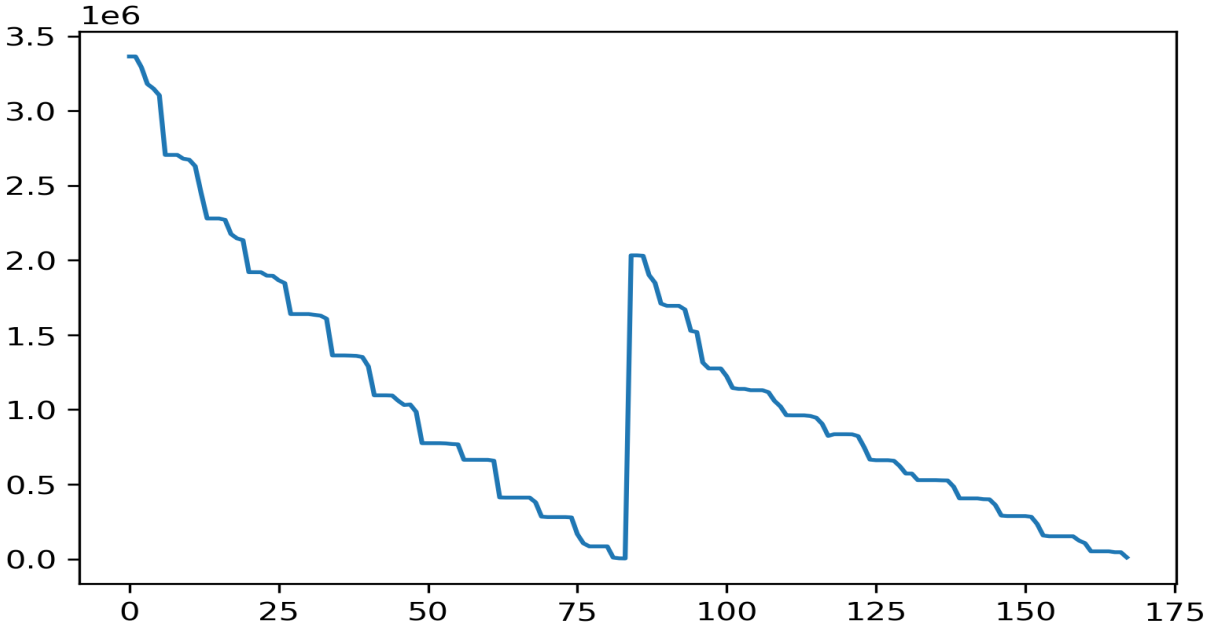


Fig4: COVID-19 Vaccine Distribution in Michigan and Missouri from 2020-12-14 - 2021-03-7

3. Vaccine Distribution and mortality rate due to COVID-19.

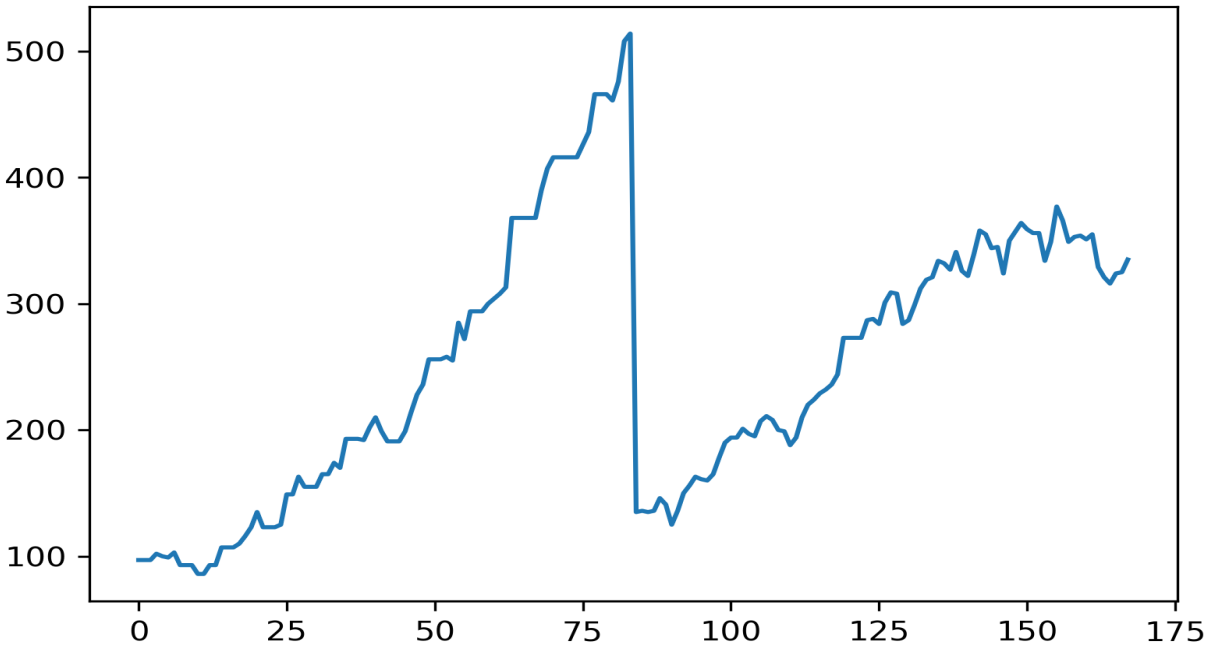


Fig5: Hospitalizations in Michigan and Missouri from 2020-12-14 - 2021-03-7

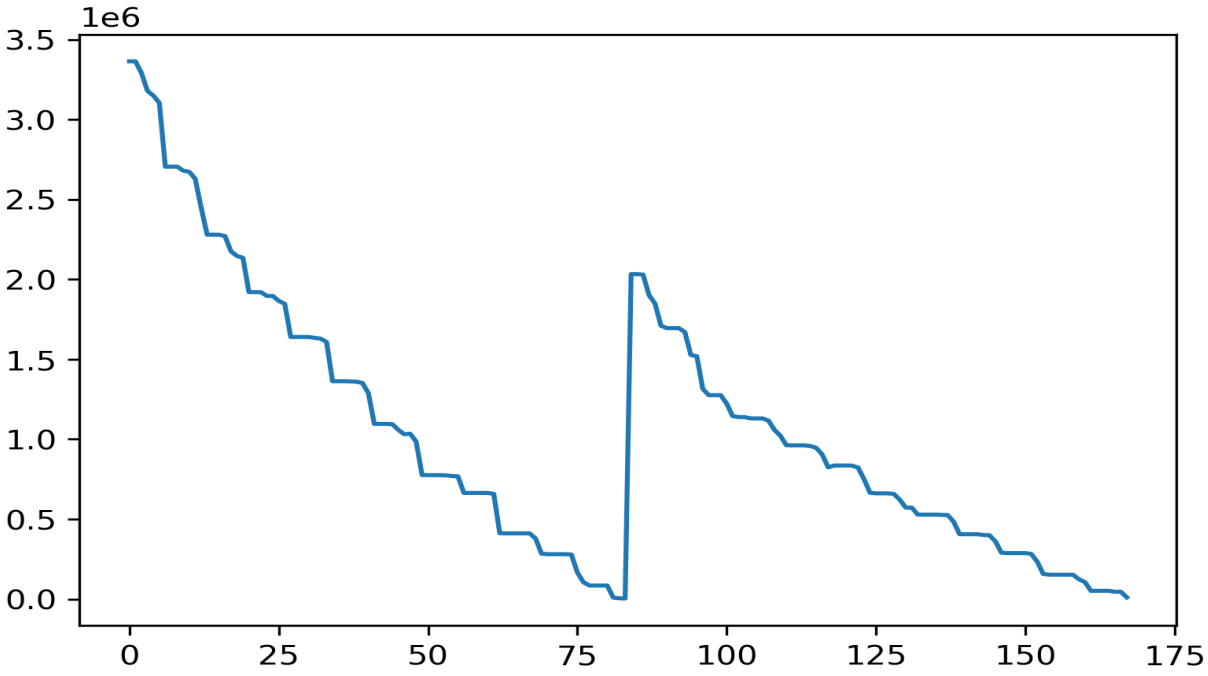


Fig6: Number of patients on ventilator in Michigan and Missouri from 2020-12-14 - 2021-03-7

****
Inference 1:

During time interval: 2020-04-10 - 2020-06-02

Pearson's Correlation Test

H0: Number of cases in Michigan and Missouri NOT LINEARLY CORRELATED with number of hospitalizations
H1: Number of cases in Michigan and Missouri LINEARLY CORRELATED with number of hospitalizations

Pearson's Correlation Coefficient: 0.5666901285910946

Since, |Pearson's Correlation Coefficient| > 0.5, H0 is REJECTED!!
Thus, Number of cases in Michigan and Missouri are POSITIVE LINEARLY CORRELATED with Number of hospitalizations

****
CHI-Square Test
****

H0: Number of cases in Michigan and Missouri are INDEPENDENT of the Number of hospitalizations
H1: Number of cases in Michigan and Missouri are DEPENDENT of the Number of hospitalizations

Q-Observed using Chi-Squared test: 53.55494505494505

DoF: 1
p-value (Pr(CHI_sq_dof > 53.55494505494505) from chi_square table: 0.00001

Since, p-value < 0.05 so H0 will be REJECTED!!
Thus, the number of cases in Michigan and Missouri are DEPENDENT on the Number of Hospitalizations during 2020-04-10 to 2020-06-02
****

Above two tests show the number of cases in Michigan and Missouri and the Number of hospitalizations in the states are DEPENDENT and POSITIVELY CORRELATED complementing the observation that when COVID-19 initially broke out, there weren't any

medicines available in the market. Thus, it required infected people with symptoms to be hospitalized during 2020-04-10 - 2020-06-02!!

****

Inference 2:

During time interval: 2020-12-14 - 2021-03-7

Pearson's Correlation Test

H0: Number of vaccines distributed in Michigan and Missouri are NOT LINEARLY CORRELATED with Number of hospitalizations
H1: Number of vaccines distributed in Michigan and Missouri are LINEARLY CORRELATED with Number of hospitalizations

Pearson's Correlation Coefficient: -0.9025969037470797

Since, |Pearson's Correlation Coefficient| > 0.5, H0 is REJECTED!!
Thus, Number of vaccines distributed in Michigan and Missouri are NEGATIVE LINEARLY CORRELATED with Number of hospitalizations

****

CHI-Square Test

****

H0: Number of vaccines distributed in Michigan and Missouri are INDEPENDENT with Number of hospitalizations
H1: Number of vaccines distributed in Michigan and Missouri are DEPENDENT with Number of hospitalizations

Q-Observed using Chi-Squared test: 141.35413630766766

DoF: 1
p-value (Pr(CHI_sq_dof > 141.35413630766766) from chi_square table: 0.00001

Since, p-value < 0.05 so H0 will be ACCEPTED!!
Thus, Number of vaccines distributed in Michigan and Missouri are DEPENDENT with Number of hospitalizations during 2020-12-14 to 2021-03-7

****

Above two tests show Number of vaccines distributed in Michigan and Missouri and Number of hospitalizations are DEPENDENT and NEGATIVELY CORRELATED. This

reinforces the observation that as vaccine distributions increased, the number of hospitalizations decreased. This gives us an overall indication of the vaccine's efficacy on COVID-19 and suggests that the vaccines had a positive impact in the fight against COVID-19.

****

Inference 3:

During time interval: 2020-12-14 - 2021-03-7

Pearson's Correlation Test

H0: Number of vaccines distributed in Michigan and Missouri are NOT LINEARLY CORRELATED with Number of Patients on Ventilator
H1: Number of vaccines distributed in Michigan and Missouri are LINEARLY CORRELATED with Number of Patients on Ventilator

Pearson's Correlation Coefficient: -0.9085759128664602

Since, |Pearson's Correlation Coefficient| > 0.5, H0 is REJECTED!!
Thus, Number of vaccines distributed in Michigan and Missouri are NEGATIVE LINEARLY CORRELATED with Number of Patients on Ventilator

****
CHI-Square Test
****

H0: Number of vaccines distributed in Michigan and Missouri are INDEPENDENT of Number of Patients on Ventilator
H1: Number of vaccines distributed in Michigan and Missouri are DEPENDENT of Number of Patients on Ventilator

Q-Observed using Chi-Squared test: 156.23642806520198

DoF: 1
p-value (Pr(CHI_sq_dof > 156.23642806520198) from chi_square table: 0.00001

Since, p-value < 0.05 so H0 will be REJECTED!!
Thus, Number of vaccines distributed in Michigan and Missouri are DEPENDENT of Number of Patients on Ventilator during 2020-12-14 to 2021-03-7
****

Above two tests shows Number of vaccines distributed in Michigan and Missouri and Number of Patients on Ventilator are DEPENDENT and NEGATIVE LINEARLY CORRELATED during 2020-12-14 to 2021-03-7 complementing the efficacy results of vaccines. The results clearly show the efficacy of the vaccines are good and thus, the number of patients required to be put on ventilators has reduced after vaccine distribution began.
