**COVID cases in TEXAS and Infection Rate**

**- A Data Science Perspective**

**-by**

**Abhijat Goyal**

**Introduction**

We are more than a month into the COVID-19 pandemic in United States, with New York taking the brunt of the outbreak. Sitting in my home, working on this capstone project for IBM data science specialization, I realize that data is everywhere. I hear Dr Birx repeat many a times about the data driven approach to solving this pandemic. I listen to Dr. Fauci, and he is saying that we need to study the data coming in from states and counties, and that is how the vital medical equipment (such as PPE, ventilators, etc) will be deployed from the federal stockpile based on this micro level data coming in from the county level. They also talk about imperfections in data, wherein 30-40% of testing data is never reported into the CDC. Bottomline, data is everywhere.

In this short capstone project, I will try and present the COVID-19 epidemic in Texas from a data science perspective and understand how we can use machine learning to predict the future cases. The reason I chose Texas is because I live in Texas. One of the primary underlying factors in predicting how long will this wave last is to study the "infectiousness" of the disease.

**Methodology**

Scourging the internet for technical details, I found that the primary metric for understanding the spread of this disease and its rate of spread is to understand something called as "R0". Now what is R0? This is called as basic reproduction number. It’s defined as the number of secondary infections produced by a single infection. If R0 is greater than one, the epidemic spreads quickly. If R0 is less than one, there is an inherent damping to the spread, and the infection eventually dies down. The smaller the number is compared to 1, the greater is the rate of die down of the infection in the population. Just to provide the readers with some reference numbers, the flu has an R0 between one and two while measles sits in the high teens. One of the basic flaws with R0 is that it is a static number. We need a better metric, in line with the modern times, in a world where we have data science, wherein we can modify a parameter based on stream of data coming in every few minutes from counties all over the country, and monitor that parameter on a dynamic basis. One such parameter is a modified version of R0, and is called as Rt. Rt is just the time varying, instantaneous version of R0 and provides a more dynamic view of the infectiousness of the disease and helps to predict when the disease will die down.

In their seminal paper in 2008 titled “Real Time Bayesian Estimation of the Epidemic Potential of Emerging Infectious Diseases.” , Bettencourt and Ribeiro proposed a method based on Bayesian statistics to estimate the most likely value of Rt and also return a credible interval for the true value of Rt. Bettencourt & Ribeiro’s original algorithm to estimate Rt is a function of how many new cases appear each day. The relationship between the number of cases yesterday and the number of cases today give us a hint of what Rt might be. However, we can’t rely on any one day too much in trying to guess Rt, as daily case counts are imperfect due to changing testing capacity, lags in data reporting, and random chance. However, using Bayes’ Theorem, we can take the new information we get from each day’s case count to adjust our expectation of what Rt is, getting closer to the true value as more daily data becomes available.

This is Bayes' Theorem as we'll use it:

𝑃(𝑅𝑡|𝑘)=𝑃(𝑘|𝑅𝑡)⋅𝑃(𝑅𝑡) / 𝑃(𝑘)

This says that, having seen 𝑘 new cases, we believe the distribution of 𝑅𝑡 is equal to:

1. The likelihood of seeing 𝑘k new cases given 𝑅𝑡 times ...
2. The prior beliefs of the value of 𝑃(𝑅𝑡) without the data ...
3. divided by the probability of seeing this many cases in general.

This is for a single day. To make it iterative: every day that passes, we use yesterday's prior 𝑃(𝑅𝑡−1) to estimate today's prior 𝑃(𝑅𝑡). We will assume the distribution of 𝑅𝑡 to be a Gaussian centered around 𝑅𝑡−1 so 𝑃(𝑅𝑡|𝑅𝑡−1)=N(𝑅𝑡−1,𝜎), where 𝜎 is a hyperparameter (see below on how we estimate 𝜎). So, on day one:

𝑃(𝑅1|𝑘1) ∝ 𝑃(𝑅1)⋅L(𝑅1|𝑘1)

On day two:

𝑃(𝑅2|𝑘1,𝑘2) ∝ 𝑃(𝑅2)⋅L(𝑅2|𝑘2) = ∑𝑅1𝑃(𝑅1|𝑘1)⋅𝑃(𝑅2|𝑅1)⋅L(𝑅2|𝑘2)

etc.

One drawback of the algorithm proposed by Bettencourt and Ribeiro is that with time, the value of Rt converges asymptotically to 1. However, in the real world, the value of Rt can be below 1, and can go to zero, representing a scenario where there is no infection. This was fixed by Kevin Systrom, in his very good work in March of 2020. The modelling approach and data analysis presented in this case study is based on his work.

The first step is to model the 'arrival' process of infections. A popular choice for the distribution of arrivals amongst statisticians is the Poisson Distribution. Accordingly, if we let λ represent the average rate of infections per day, then the probability that we are likely to see k new cases on a day, is given by

𝑃(𝑘|λ)=[λ𝑘𝑒xp(−λ)] / 𝑘!

Given this setup, we can construct the probability distribution of new cases for a set of λ's.

**Code Setup on Fictitious Data**

To start our analysis, we can start working on some assumed cases per day. One such case can be an arbitrary number of cases per day ranging from 20 to 80.

The steps involved are –

1. **Importing necessary libraries**

*#Load packages*

**import** pandas **as** pd

**import** numpy **as** np

​**from** matplotlib **import** pyplot **as** plt

**from** matplotlib.dates **import** date2num, num2date

**from** matplotlib **import** dates **as** mdates

**from** matplotlib **import** ticker

**from** matplotlib.colors **import** ListedColormap

**from** matplotlib.patches **import** Patch

​**from** scipy **import** stats **as** sps

**from** scipy.interpolate **import** interp1d

​**from** IPython.display **import** clear\_output

1. **Doing some basic set up**

Let's do some basic analysis and plot some graphs to gain an understanding of k and lambda. The first step is to create a column matrix of k, which is infection per day. Let's assume that this can vary from 0 to 7. Just using a "arange" function of numpy to generate the k column vector.

​k **=** np.arange(0, 70)[:, **None**]

​Let's choose different values of lambda to study the effect of the value of lambda on the distribution of k

lambdas **=** [10, 15, 20, 25, 30, 35, 40, 50, 60, 70]

​Now let's use scipy package to evaluate the poisson's Probability Mass Function

y **=** sps.poisson.pmf(k, lambdas)

​Let's look at the resulting matrix and see it's shape

print(y.shape)

Output: (70, 10)

To gain visual understanding of the k and lambda values and its affect on Poisson's equation

​fig, ax **=** plt.subplots()

ax.set(title**=**'Poisson Distribution of Cases\n $p(k|\lambda)$')

​plt.plot(k, y,

marker**=**'.',

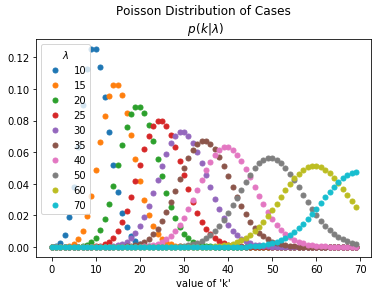
markersize**=**10,

lw**=**0)

​

plt.legend(title**=**"$\lambda$", labels**=**lambdas);

ax.set\_xlabel('value of \'k\'');



Please remember that the value of 𝜆 in the above graph is the number of cases per day. As you can see, there is quite a bit of spread across the value of 𝜆 and that is the inherent probability that there will not be exactly the same cases every day. There will be a significant noise in the data, as plotted on a daily basis or even hourly basis. For example, imagine we observe 𝑘=30k=30 new cases, and we want to know how likely each 𝜆 is:

k **=** 30

​lam **=** np.linspace(1, 60, 90)

​

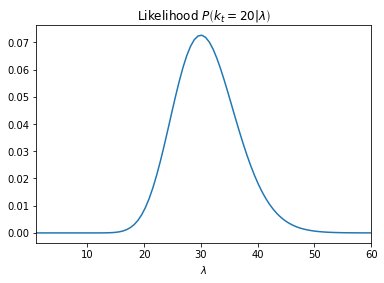
likelihood **=** pd.Series(data**=**sps.poisson.pmf(k, lam),

index**=**pd.Index(lam, name**=**'$\lambda$'),

name**=**'lambda')

​

likelihood.plot(title**=**r'Likelihood $P\left(k\_t=20|\lambda\right)$');



# Connecting 𝜆 and 𝑅𝑡

The key insight to making this work is to realize there's a connection between 𝑅𝑡 and 𝜆. The derivation is beyond the scope of this capstone project, but here it is:

𝜆=𝑘𝑡−1𝑒xp[𝛾(𝑅𝑡−1)]

where 𝛾 is the reciprocal of the serial interval (about 7 days for COVID19). Since we know every new case count on the previous day, we can now reformulate the likelihood function as a Poisson parameterized by fixing 𝑘 and varying 𝑅𝑡.

𝜆=𝑘𝑡−1𝑒xp[𝛾(𝑅𝑡−1)]

𝑃(𝑘|𝑅𝑡)=[𝜆𝑘𝑒xp(−𝜆)] / 𝑘!

Let's define some arbitrary values for k

​k **=** np.array([20, 30, 40, 50, 60, 70, 80, 90])

​Now let's define an array for every possible value of Rt.

R\_T\_MAX **=** 12

r\_t\_range **=** np.linspace(0, R\_T\_MAX, R\_T\_MAX**\***100**+**1)

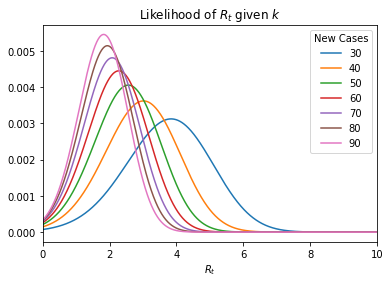
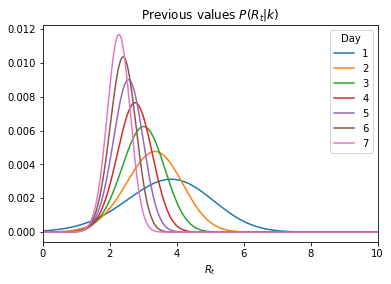
​

Gamma is 1/serial interval and is adequately documented and described in the following websites, <https://wwwnc.cdc.gov/eid/article/26/7/20-0282_article>, and <https://www.nejm.org/doi/full/10.1056/NEJMoa2001316>.

​GAMMA **=** 1**/**7

​As can be seen from the following graph, we can get likelihood of Rt given the values of k. As we can see, there is quite a bit of spread and hence uncertainty with the values of k.

To perform the Bayesian update, we need to multiply the likelihood by the prior (which is just the previous day's likelihood without our Gaussian update) to get the posteriors. Let's do that using the cumulative product of each successive day. The resulting graph is shown below -



Looking at the data, after some processing on a day to day basis, for the fictitious numbers we have, we get that as we go from one day to next, we in table below, the infection rate decreases. Day Infection Rate (Rt)

1 3.84

2 3.36

3 3.01

4 2.76

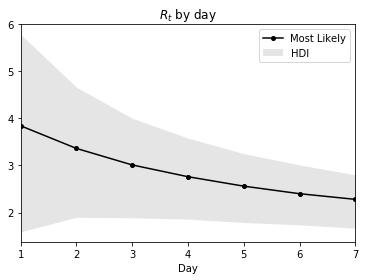
5 2.56

6 2.40

7 2.28

Further, we can calculate the confidence levels on these numbers, which are given in the table below.

|  | **Low\_90** | **High\_90** |
| --- | --- | --- |
| **Day** |  |  |
| **3** | 1.89 | 4.00 |
| **4** | 1.86 | 3.58 |
| **5** | 1.79 | 3.25 |
| **6** | 1.74 | 3.01 |
| **7** | 1.67 | 2.80 |

The same has been drawn as a figure below.

**ACTUAL DATA SKIMMING AND ANALYSIS FOR TEXAS**

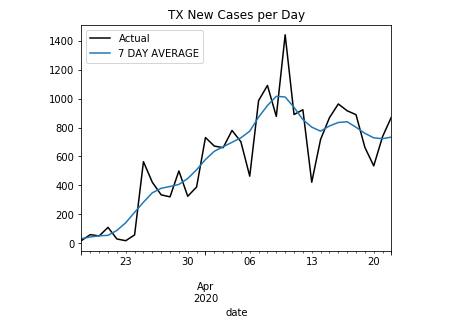
Now that we have understood and implemented the modelling part using a range of data from 20 infections per day to 90 infections per day, let's apply our code to the actual data from Texas which is updated daily

For this, we will be importing data from a website and running our analysis on it.

url **=** 'https://covidtracking.com/api/v1/states/daily.csv'

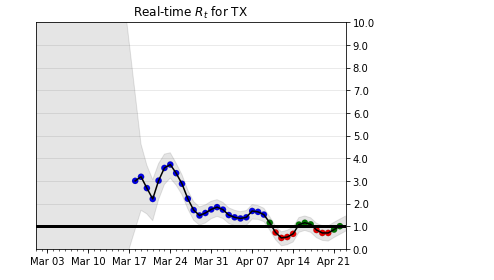
states **=** pd.read\_csv(url, usecols**=**['date', 'state', 'positive'], parse\_dates**=**['date'], index\_col**=**['state', 'date'], squeeze**=True**).sort\_index()

​The data from the website was imported and the curve below shows the number of cases per day. The blue line is the 7 day average smoothing of the data.

Having this data, we can calculate the value of Rt, and the associated confidence levels for each day wherein the data is available for the COVID cases in Texas. The table below shows the table for each day -

|  | **ML** | **Low\_90** | **High\_90** |
| --- | --- | --- | --- |
| **date** |  |  |  |
| **2020-03-18** | 3.00 | 0.86 | 6.84 |
| **2020-03-19** | 3.18 | 1.70 | 4.62 |
| **2020-03-20** | 2.68 | 1.54 | 3.69 |
| **2020-03-21** | 2.20 | 1.24 | 3.05 |
| **2020-03-22** | 3.01 | 2.17 | 3.76 |
| **2020-03-23** | 3.57 | 2.84 | 4.18 |
| **2020-03-24** | 3.72 | 3.12 | 4.24 |
| **2020-03-25** | 3.34 | 2.81 | 3.78 |
| **2020-03-26** | 2.87 | 2.39 | 3.27 |
| **2020-03-27** | 2.21 | 1.76 | 2.59 |
| **2020-03-28** | 1.71 | 1.26 | 2.08 |
| **2020-03-29** | 1.47 | 1.04 | 1.85 |
| **2020-03-30** | 1.58 | 1.14 | 1.94 |
| **2020-03-31** | 1.74 | 1.33 | 2.10 |
| **2020-04-01** | 1.84 | 1.43 | 2.17 |
| **2020-04-02** | 1.73 | 1.34 | 2.05 |
| **2020-04-03** | 1.49 | 1.12 | 1.81 |
| **2020-04-04** | 1.39 | 1.02 | 1.70 |
| **2020-04-05** | 1.34 | 0.97 | 1.64 |
| **2020-04-06** | 1.39 | 1.03 | 1.69 |
| **2020-04-07** | 1.67 | 1.31 | 1.95 |
| **2020-04-08** | 1.63 | 1.30 | 1.91 |
| **2020-04-09** | 1.51 | 1.20 | 1.79 |
| **2020-04-10** | 1.15 | 0.83 | 1.42 |
| **2020-04-11** | 0.72 | 0.40 | 1.00 |
| **2020-04-12** | 0.48 | 0.13 | 0.75 |
| **2020-04-13** | 0.52 | 0.18 | 0.82 |
| **2020-04-14** | 0.66 | 0.30 | 0.96 |
| **2020-04-15** | 1.07 | 0.73 | 1.38 |
| **2020-04-16** | 1.15 | 0.81 | 1.45 |
| **2020-04-17** | 1.08 | 0.73 | 1.37 |
| **2020-04-18** | 0.83 | 0.49 | 1.13 |
| **2020-04-19** | 0.70 | 0.37 | 1.02 |
| **2020-04-20** | 0.70 | 0.34 | 1.01 |

The above table gives us a fair idea of what the infectivity Rt was each day of this COVID-19 pandemic in Texas. But it will be nice to plot it for the people to be able to see the graph. Here is the graphical representation of the above data -



**CONCLUSIONS**

As we can see from the above graph, the dynamic infection rate fell below one sometimes around April 10. It then bounced back and now it is hovering around 1. Since any value of less than one results in the infection eventually dying down with time, this is great news for the state of Texas. By using data analysis of the data present in one of the government websites, we were able to calculate the infectivity of the disease on a day to day basis and provide future predictive nature of the disease and how long it will last. It is very important for the businesses to know this information because it has far reaching implications in terms of social distancing, office closures, customer demand, supply chain constraints, etc.