COVID-19
Infection Prediction

Course: SC 205
Course Instructor: Dr. Manish K. Gupta

DA-IICT
Discrete Mathematics

Nirmit Ghughu
Student ID: 201901228

7th June 2020

Acknowledgement

I would like to express my sincere gratitude to my course instructor Dr. Manish K. Gupta for giving me opportunity to do this project. I would also like to thank my family and friends, especially Satyadev Patel, for helping me in this project.

Contents

1 Introduction

In the wake of a global pandemic like COVID-19, it is the responsibility and duty of all humans to contribute for the betterment of mankind in any way they can. This project of mine is a humble step towards the same.

1.1 About COVID-19

COVID-19 is the infectious disease caused by the most recently discovered corona virus. This new virus and disease were unknown before the outbreak began in Wuhan, China, in December 2019. This has turned into a global pandemic and governments, scientists, tech-giants and universities all around the world are trying to fight this at their very best.

1.2 About this project

This project is an attempt at trying to predict the chances of a person being infected by COVID-19. For developing this, I have used logistic regression which is a machine learning algorithm.

1.3 What is logistic regression?

Logistic regression is a machine learning algorithm. Like all regression analyses, the logistic regression is a predictive analysis as well.It is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables.

2 Working towards the solution

Step 1. Deciding the features that contribute to person being infected by COVID-19.

Step 2. Formulating the basic equation.

Step 3. Formulating the cost function and mathematically compressing it.

Step 4. Formulating the gradient descent function and calculating the partial derivatives to get a one line function.

Step 5. Using Vectorization for all the calculations done before so that it is easier to code and reduces the code's computational complexity.

Step 6. Load the data set to train our logistic regression function.

Step 7. Run gradient multiple times (100-400) until we get approximately the least possible value of cost function. This will give us the optimum θ values.

Step 8. Input feature(x) values from the used and calculate $100h_\theta(x)$ to get percentage chances of infection.

3 COVID-19 Prediction

3.1 Features

The very first step is to make a list of factors(features) that could contribute to a person being infected by COVID-19. These include symptoms,travel history and age.The features are:-

x_1=age

Most common symptoms:
x_2=fever(in Fahrenheit)
x_3=dry cough(0 or 1)
x_4=tiredness(0 or 1)

Less common symptoms:
x_5=aches and pains(0 or 1)
x_6=sore throat(0 or 1)
x_7=diarrhoea(0 or 1)
x_8=conjunctivitis(0 or 1)
x_9=headache(0 or 1)
x_{10}=loss of taste or smell(0 or 1)
x_{11}=a rash on skin, or discolouration of fingers or toes(0 or 1)

Serious symptoms:
x_{12}=difficulty breathing or shortness of breath(0 or 1)
x_{13}=chest pain or pressure(0 or 1)
x_{14}=loss of speech or movement(0 or 1)

Travel history:
x_{15}=International travel in last 14 days(0 or 1)
x_{16}=Inter-state travel in last 14 days(0 or 1)
x_{17}=Local travel in last 14 days(0 or 1)

3.2 Basic Equation

The basic equation is:-

g_θ(x)= $\theta_0 x_0$+$\theta_1 x_1$+$\theta_2 x_2$+$\theta_3 x_3$...

$where$ x_0 = 1 and x_1,x_2,x_3 etc. are features that would decide if a person is infected by covid-19.
θ_0,θ_1,θ_2,θ_3 are the parameters(coefficients) that decide how much each feature affects the final outcome.
The main objective is to find optimum $_theta$ values so as to get most accurate outcome.
For the purpose of logistic regression, we use the equation:-

h_θ(x)=$\frac{1}{1+e^{g(x)}}$

This equation is used so that the output is between 0 and 1 and the impact that an odd training example has on choosing the values of θ is minimalized.

3.3 Cost Function

We can measure the accuracy of our hypothesis function by using a cost function. Higher the value of the cost function, more is the error.
The cost function we use is as follows:-

J(θ)=$\frac{1}{m}\sum_{i=1}^{m} Cost(h_\theta(x^i), y^i)$
$Cost(h_\theta(x), y) = -\log{(h_\theta(x))}$ if y=1
$Cost(h_\theta(x), y) = -\log{(1 - h_\theta(x))}$ if y=0

Here,
'y=1' for positive Covid-19 cases.
'y=0' for negative Covid-19 cases.
'm' denotes the total number of test cases.
'i' denotes the individual test cases.

On compressing the cost function to a single expression, we get:-

$Cost(h_\theta(x), y) = -y\log{(h_\theta(x))} - (1-y)\log{(1-h_\theta(x))}$

Therefore, the complete cost function can be written as:-

J(θ)=$-\frac{1}{m}\sum_{i=1}^{m}[y^i\log{(h_\theta(x^i))} + (1-y^i)\log{(1-h_\theta(x^i))}]$

3.4 Gradient Descent

Gradient descent is an optimization algorithm used to find the values of parameters (coefficients) of a function (h_θ(x)) that minimizes a cost function (J(θ)). Gradient descent is best used when the parameters cannot be calculated analytically (e.g. using linear algebra) and must be searched for by an optimization algorithm.

The way we do this is by taking the derivative (the tangential line to a function) of our cost function. The slope of the tangent is the derivative at that point and it will give us a direction to move towards. We make steps down the cost function in the direction with the steepest descent. The size of each step is determined by the parameter α, which is called the learning rate.

The gradient descent algorithm is: repeat until convergence:-

$\theta_0 := \theta_0 - \alpha\frac{\delta}{\delta\theta_0}J(\theta)$
$\theta_1 := \theta_1 - \alpha\frac{\delta}{\delta\theta_1}J(\theta)$
$\theta_2 := \theta_2 - \alpha\frac{\delta}{\delta\theta_2}J(\theta)$
$\theta_3 := \theta_3 - \alpha\frac{\delta}{\delta\theta_3}J(\theta)$
$\theta_4 := \theta_4 - \alpha\frac{\delta}{\delta\theta_4}J(\theta)$
.
.
.
$\theta_j := \theta_j - \alpha\frac{\delta}{\delta\theta_j}J(\theta)$
On calculating the partial derivatives of $J(\theta)$ we get:-
$\theta_0 := \theta_0 - \frac{\alpha}{m}\sum_{i=1}^{m}(h_\theta(x^i) - y^i)$
$\theta_1 := \theta_1 - \frac{\alpha}{m}\sum_{i=1}^{m}(h_\theta(x^i) - y^i)x_1$
$\theta_2 := \theta_2 - \frac{\alpha}{m}\sum_{i=1}^{m}(h_\theta(x^i) - y^i)x_2$
$\theta_3 := \theta_3 - \frac{\alpha}{m}\sum_{i=1}^{m}(h_\theta(x^i) - y^i)x_3$
$\theta_4 := \theta_4 - \frac{\alpha}{m}\sum_{i=1}^{m}(h_\theta(x^i) - y^i)x_4$
.
.
.
$\theta_j := \theta_j - \frac{\alpha}{m}\sum_{i=1}^{m}(h_\theta(x^i) - y^i)x_j$

α is the variable which controls the rate at which θ values change. It has to be set manually and it is usually less than 1.

The values of θ are to be updated simultaneously. This is repeated until we get minimum value of J(θ).

3.5 Vectorization

Vectorization is the process of converting an algorithm from operating on a single value at a time to operating on a set of values (vector) at one time. It is used to make the code more clean, faster and to avoid the use of loops.
Let,
x is a vector of all features.

$$x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ . \\ . \\ . \end{bmatrix}$$

θ is a vector of all parameters(coefficients).

$$\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \theta_3 \\ \theta_4 \\ . \\ . \\ . \end{bmatrix}$$

X is matrix of x of all training examples.

$$X = \begin{bmatrix} (x^1)^T \\ (x^2)^T \\ (x^3)^T \\ (x^4)^T \\ . \\ . \\ . \\ (x^m)^T \end{bmatrix}$$

Therefore,

$$\text{X} = \begin{bmatrix} x_0^1 & x_1^1 & x_2^1 & x_3^1 & x_4^1 & . & . & . \\ x_0^2 & x_1^2 & x_2^2 & x_3^2 & x_4^2 & . & . & . \\ x_0^3 & x_1^3 & x_2^3 & x_3^3 & x_4^3 & . & . & . \\ x_0^4 & x_1^4 & x_2^4 & x_3^4 & x_4^4 & . & . & . \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . \\ x_0^m & x_1^m & x_2^m & x_3^m & x_4^m & . & . & . \end{bmatrix}$$

y is a vector of all outcomes of the training examples.

$$y = \begin{bmatrix} y^1 \\ y^2 \\ y^3 \\ y^4 \\ . \\ . \\ . \\ y^m \end{bmatrix}$$

Using this, we can rewrite the equations as follows:-

g_θ(X)=$\theta^T X$
h_θ(X)=$\frac{1}{1+e^{\theta^T X}}$
J(θ)=$-\frac{1}{m}[y^T \log{(h_\theta(X))} + (1-y^T)\log{(1-h_\theta(X))}]$
$\theta := \theta - \frac{\alpha}{m}X^T(h_\theta(X) - y)$

4 Interpretation and significance of this project

4.1 Interpretation of the solution

The set of θ values we obtain after performing logistic regression tells us how much each feature contributes to a person being infected by COVID-19.
After this is done, all a person needs to do is fill in the vector 'x' i.e. the features and calculate $100h_\theta(x)$ to get the percentage chances of him/her being infected. This is for a person to calculate the risk he/she might be at and take precautions accordingly.

4.2 Commercialization and significance of this project

This solution could be turned into an app or website for people to check the risk they face and also for government and medical institutions to get a better estimate of who might be infected and give priority to people facing a higher risk.

5 How could the solution be improved?

5.1 Dependent features

A lot of the original features are dependent on each other due to which new features like $x_{18} = x_1x_2$ or $x_{18} = x_4x_5x_6$ could be formed.
Hence the total number of new features = ${}^{17}C_2$ + ${}^{17}C_3$ + ${}^{17}C_4$ +...+ ${}^{17}C_5$
This is a lot of features which could improve the accuracy of the logistic regression function but it could also lead to overfitting.

To solve the problem of overfitting we would have to use regularization.

Using these many features is not feasible at this point due to the unavailability of a much larger dataset which would be required to train the logistic regression function in this case.

5.2 Alternate Solution

An alternative to using logistic regression could be to use neural networks. This could generate more accurate results.

6 Software

I have used MATLAB for developing this software.

7 Links

Website link: https://sites.google.com/view/covid19infectionprediction
Youtube link: https://www.youtube.com/watch?v=WmQFkCL6Ptkt=22s
Software link: https://github.com/Nirmit22/COVID-19-Infection-Prediction

8 Bibliography

1. https://www.coursera.org/learn/machine-learning/home/week/1
2. https://www.coursera.org/learn/machine-learning/home/week/3