SARDAR PATEL UNIVERSITY

DEPARTMENT OF STATISTICS

“MASTER OF APPLIED STATISTICS”

A

PROJECT REPORT

(PS04CAST23)

On

“Artificial Neural Networks for Survival Data”

PROJECT GUIDE

Mr. Agniva Das(Sir)

PRESENTED By

Rushikesh Pawar

Suhas Pawar

2020 – 2021

PREFACE

"Experience is a best teacher" this saying had played a guiding role in including as a part of as a part of curricular of Master of Science in Statistics. Knowledge in itself is a continuous process. Getting practical knowledge is an important thing for an individual in a business concern. In the competition prevailing industry, a total awareness is the first and foremost thing for all aspects. Working smart seems to be as important as working harder and longer. Hence, practical expose at M.Sc Statistics after training in certain aspect has provide a boon for us.

The goal of the project is to give corporate exposure to the student as well as to give him an opportunity to apply theory into practice. The real business problems are drastically different from classroom case solving, project aims at providing little insight into working of an organization.

ACKNOWLEDGEMENT

This year has been an extremely informative journey for, my friend and me. We would like to express our special thanks of gratitude our guidance **Mr. Agniva Das(Sir)** and Head of the department **Prof. Jyoti M. Divecha** who gave us golden opportunity to do this wonderful project on the "Artificial Neural Networks for Survival Data". Documentation is heart of project, so we take opportunity to express our heartfelt thanks to all my dear friends who support and encourage my project partner and me to complete our documentation successfully We come to know about many new things . We want to grab this opportunity to acknowledge our sincere thanks to all of them while submitting.

Mr. Rushikesh Pawar (AS32)

Mr. Suhas Pawar (AS41)

Place: V.V.Nagar, Anand

Date:

CERTIFICATE

This is to certify that **Mr. Rushikesh Rajendra Pawar**, student of “Master of science in Applied Statistics”, Roll No. 32, Exam No. 35 has successfully completed his project entitled **“Artificial Neural Networks for Survival Data”** for M.Sc. (Applied Statistics) semester IV during the Dec 2020 – May 2021.

Place: Vallabh Vidyanagar

Date**:**

Project Guide	Head of Department
Mr. Aginva Das	**Prof. Jyoti M. Divecha**

Masters of Science (Applied Statistics),

Department of Statistics,

Sardar Patel University,

Vallabh Vidyanagar.

CERTIFICATE

This is to certify that **Mr. Suhas Tanaji Pawar**, student of “Master of science in Applied Statistics”, Roll No. 41, Exam No. 43 has successfully completed his project entitled **“Artificial Neural Networks for Survival Data”** for M.Sc. (Applied Statistics) semester IV during the Dec 2020 – May 2021.

Place: Vallabh Vidyanagar

Date**:**

Project Guide	Head of Department
Mr. Aginva Das	**Prof. Jyoti M. Divecha**

Masters of Science (Applied Statistics),
Department of Statistics,
Sardar Patel University,
Vallabh Vidyanagar, Anand.

INDEX

ABSTRACT

There has been increasing interest in modeling survival data using deep learning methods in medical research. The purpose of this study is to demonstrate how a form of neural network analysis can be used to perform survival analysis on survival data, and to compare neural network analysis with the most commonly used technique for this type of analysis, Cox regression. This study predicts the factors which are highly affected on the survival of corona patients using survival models and neural networks. This study forecasts the count of number of patients in next 10 days using LSTM model. It is very helpful to decision makers in planning future responses. The data obtained from the Indian Statistical Institute , Bangalore center. In Karnataka state, first COVID-19 case was detected on March 09, 2020.

Methods:

We analyzed state's COVID – 19 data from March 09, 2020 to July 21, 2020 to describe patient's distribution. We analyzed the data in such a way that, it helps to find out the factors which are mostly affects on survival of CORONA patients in Karnataka state.

INTRODUCTION

Neural networks are a form of artificial intelligence that have found application in a wide range of problems. These systems are presented as a stimulus, a set of input variables, and produce in response , a prediction as to the class or probability of an event associated with that stimulus. Neural networks have been used with comparable and, in some cases, superior results to standard mechanistic or statistical models.

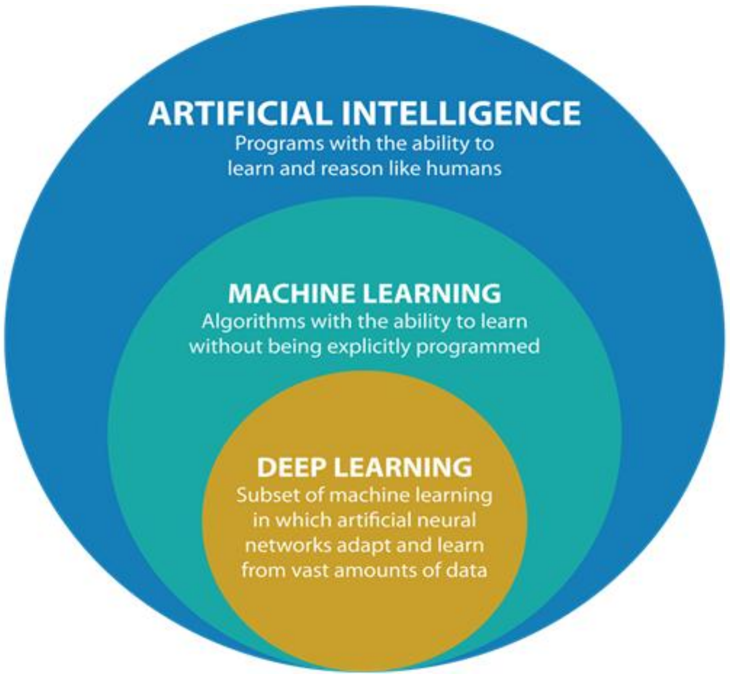
The semi – parametric Cox proportional hazards regression is the most widely used method for analyzing survival data. It studies the effects of the covariate variables on survival by estimating hazard functions. In this study, understanding the epidemiology of reported COVID – 19 cases helps decision makers in planning future responses.

OBJECTIVES

- To establish guideline of Artificial Neural Networks for survival data.
- To check whether Artificial Neural Networks are performing better for Survival data.
- To compare traditional Survival models with Artificial Neural Network models.
- To forecast number of COVID positive patients for next 10 days.

Brief terms

Introduction to Deep Learning


ARTIFICIAL INTELLIGENCE
Programs with the ability to learn and reason like humans
MACHINE LEARNING
Algorithms with the ability to learn without being explicitly programmed
DEEP LEARNING
Subset of machine learning in which artificial neural networks adapt and learn from vast amounts of data

- **Artificial Intelligence:**

  Any technique that enable computer to mimic human behavior .

- **Machine Learning:**

  Ability to learn without explicitly being programmed.

- **Deep Learning:**

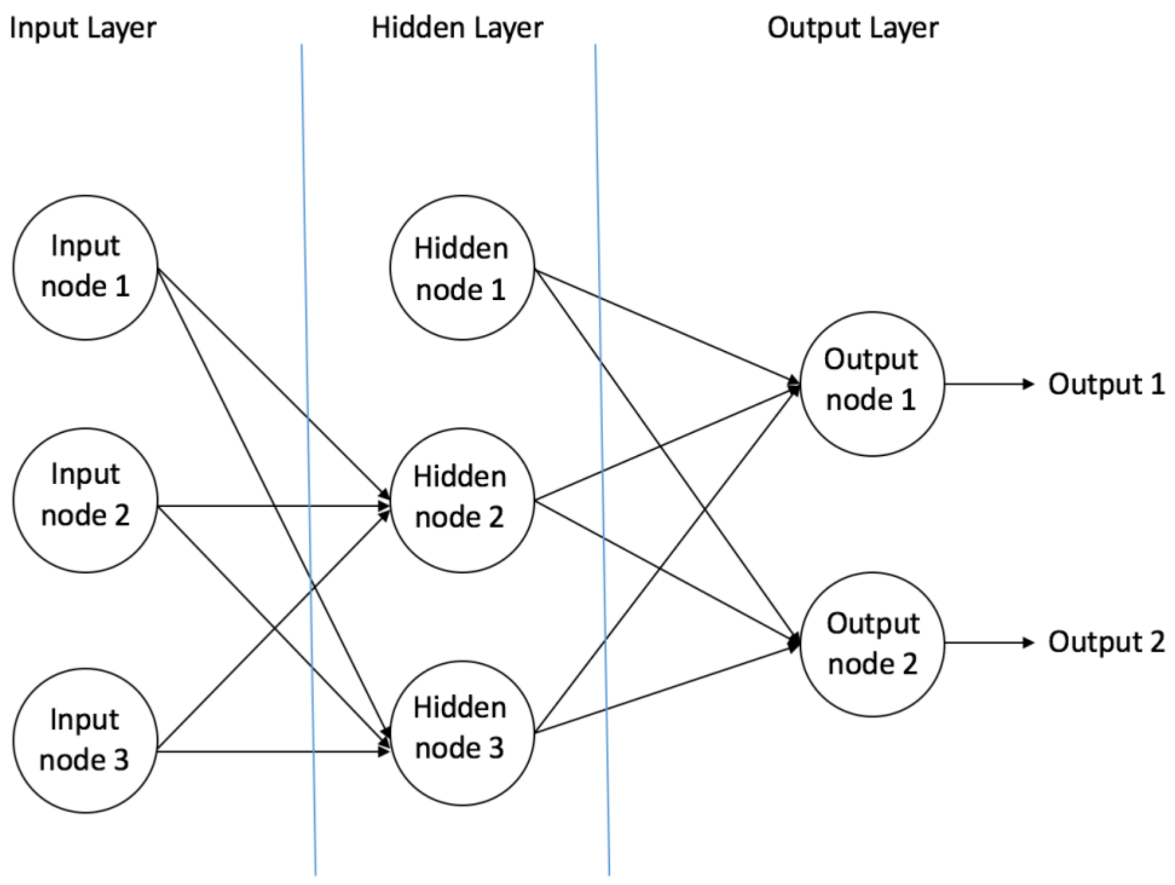
  Extract patterns from data using Neural Networks .

  Why Deep learning ?

  1. Machine learning is unable to process high dimensional data it can only process small set of variables.
  2. Feature extraction is manual in Machine learning .
  3. Machine learning is not ideal for image processing.
     Object detection requires high dimensional data and Machine learning cannot be used for data sets only with restricted number of features .

Neural Networks:

Neural Networks approach problems in a very different way by trying to mimic how neurons in the human brain work. In fact, they learn by example rather than being programmed to perform a specific task. Technically, they are composed of a large number of highly interconnected processing elements (nodes) that work in parallel to solve a specific problem, which is similar to how the human brain works.



- **Input Node –**

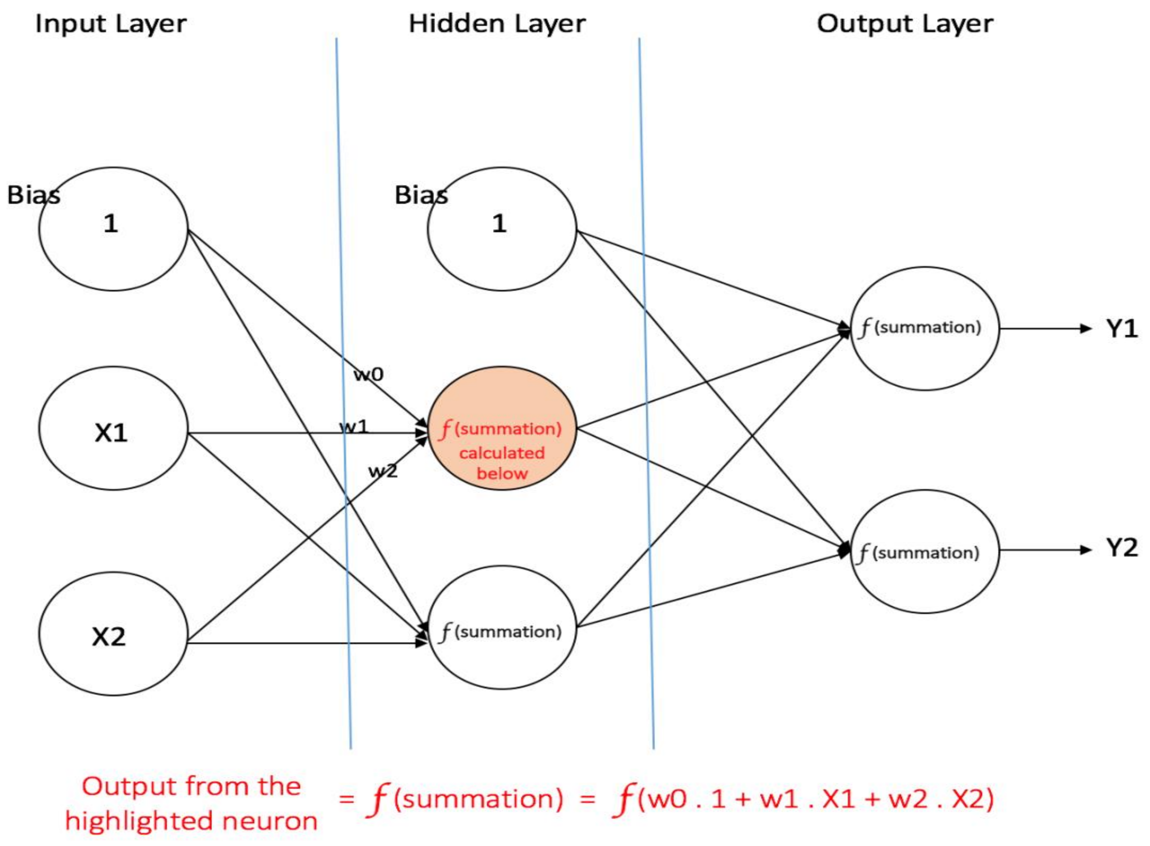
An input node contains the input of the network and this input is always numerical. If the input is not numerical by default, it is always converted. An input node is located within the input layer, which is the first layer of a neural network. Each input node represents a single dimension and is often called a feature, and all features are stored within a vector.

- **Hidden Node** –

A hidden node is a node within a hidden layer. A hidden layer can have many hidden nodes, and there are various theories for implementing the correct amount. A hidden layer is a layer of nodes between the input and output layers. There can be either a single hidden layer or multiple hidden layers in a network, and the more that exist, the "deeper" the learning that a network can perform

- **Output Node** –

An output node is a node within an output layer. There can be a single or multiple output nodes depending on the objective of the network.


Input Layer
Hidden Layer
Output Layer
Bias
1
X1
X2
Bias
1
w0
w1
w2
ƒ(summation) calculated below
ƒ(summation)
ƒ(summation)
ƒ(summation)
Y1
Y2
Output from the highlighted neuron = ƒ(summation) = ƒ(w0 . 1 + w1 . X1 + w2 . X2)

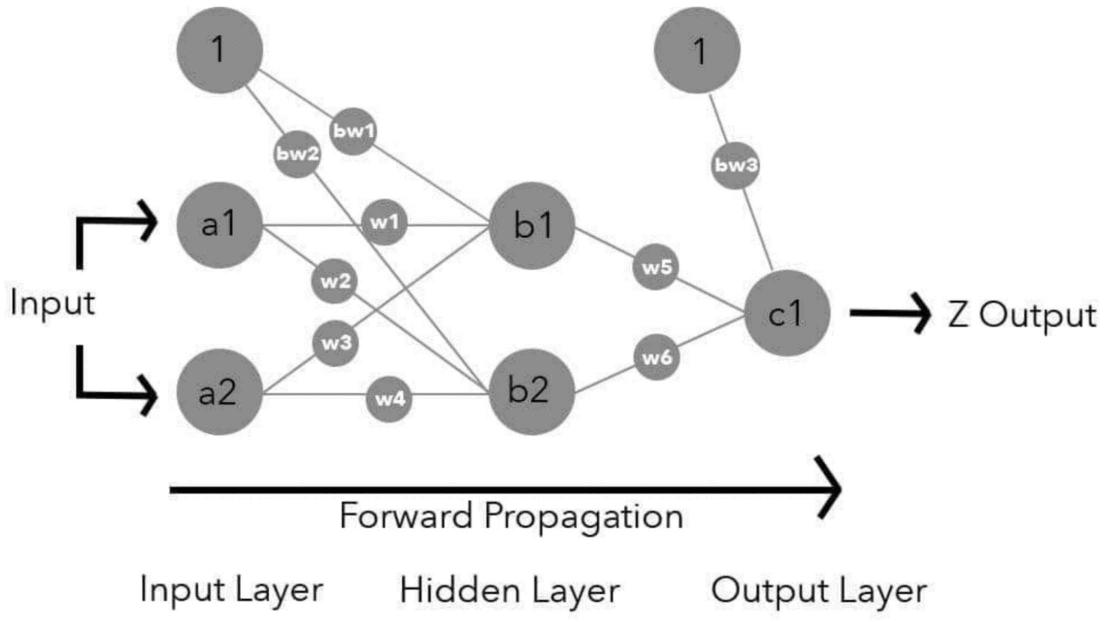
- **Bias Value** –

A bias node is an extra node added to each hidden and output layer, and it connects to every node within each respective layer. A bias is never connected to a previous layer, but is simply added to the input of a layer. In addition, it typically has a constant value of 1 or -1 and has a weight on its connecting edge.

- **Weight Value :**

A weight is a variable that sits on an edge between nodes. The output of every node in a layer is multiplied by a weight, then summed with other weighted nodes in that layer to become the net input of a node in the following layer. Algorithms are typically used to assign random weights for a neural network. A popular range is between -1 and 1.

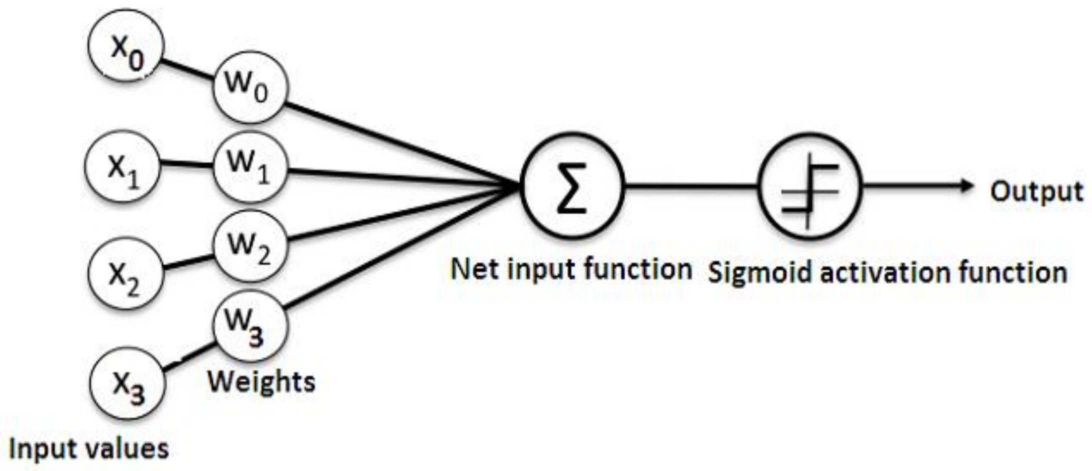
- **Forward propagation :**



We can investigate how input moves through the network to become output. This process is technically called **forward propagation**. When input is passed into the network, it moves from one layer to the next until it passes through the output layer. An **Activation functions** are repeatedly used to make all of this possible.

Activation function :

What is an Activation Function? Within a neural network, an activation function receives the output of the summation operator and transforms it into the final output of a node. On a high level, an activation function essentially "squashes" the input and transforms it into an output value that represents how much a node should contribute (i.e., how much a node should fire).

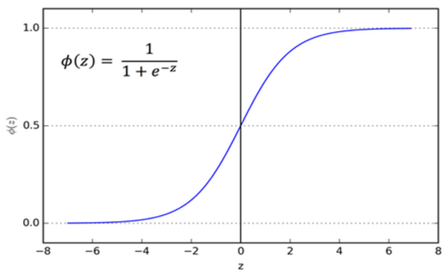


- Purpose of an Activation function is to introduce non linearity into the network . It is an transformation that we do over the input before sending it to the next layer of neurons
- Non linearity allows us to approximate arbitrarily complex function .
- The equation for an activation function differs between functions.

- Some of the types of Activation function are:

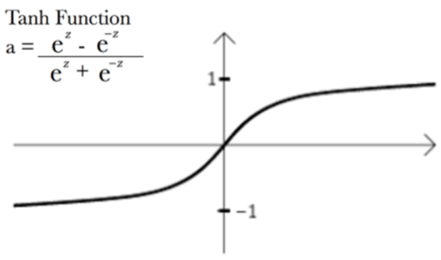
Sigmoid Function

$$\phi(z) = \frac{1}{1 + e^{-z}}$$



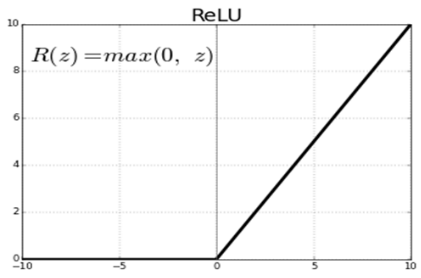
Hyperbolic Tangent

$$x = \frac{e^z - e^z}{e^z + e^z}$$



Rectified Linear Unit (ReLU)

R(z)= max(0,z)



1. Sigmoid Function (also called as logistic activation function)
2. Hyperbolic Tangent function (tanh)
3. Rectified Linear Unit (ReLU)

Loss Function (cost function):

Within a neural network, a cost function transforms everything that occurs within the network into a number that represents the total error of the network. Essentially, it is a measure of how wrong a network is. There are many types of cost functions to choose from. Popular options include the Mean Squared Error, Squared Error, Root Mean Square Error , Binary cross entropy and Sum of Square Errors .

Optimization Algorithm (Loss Optimization) :

Optimization Algorithms are used to update weights and biases i.e. the internal parameters of a model to reduce the error. They can be divided into two categories:

We used three first order optimization functions and studied their effect.

Stochastic Gradient Decent

Adagrad

Adam

Gradient Descent :

Calculates gradient for the whole dataset and updates values in direction opposite to the gradients until we find a local minima. Stochastic Gradient Descent performs a parameter update for each training example unlike normal Gradient Descent which performs only one update. Thus it is much faster. Gradient Decent algorithms can further be improved by tuning important parameters like momentum, learning rate etc.

Adagrad :

Adagrad is more preferable for a sparse data set as it makes big updates for infrequent parameters and small updates for frequent parameters. It uses a different learning Rate for every parameter θ at a time step based on the past gradients which were computed for that parameter. Thus we do not need to manually tune the learning rate.

Adam:

Adam stands for Adaptive Moment Estimation. It also calculates different learning rate. Adam works well in practice, is faster, and outperforms other techniques.

SGD:

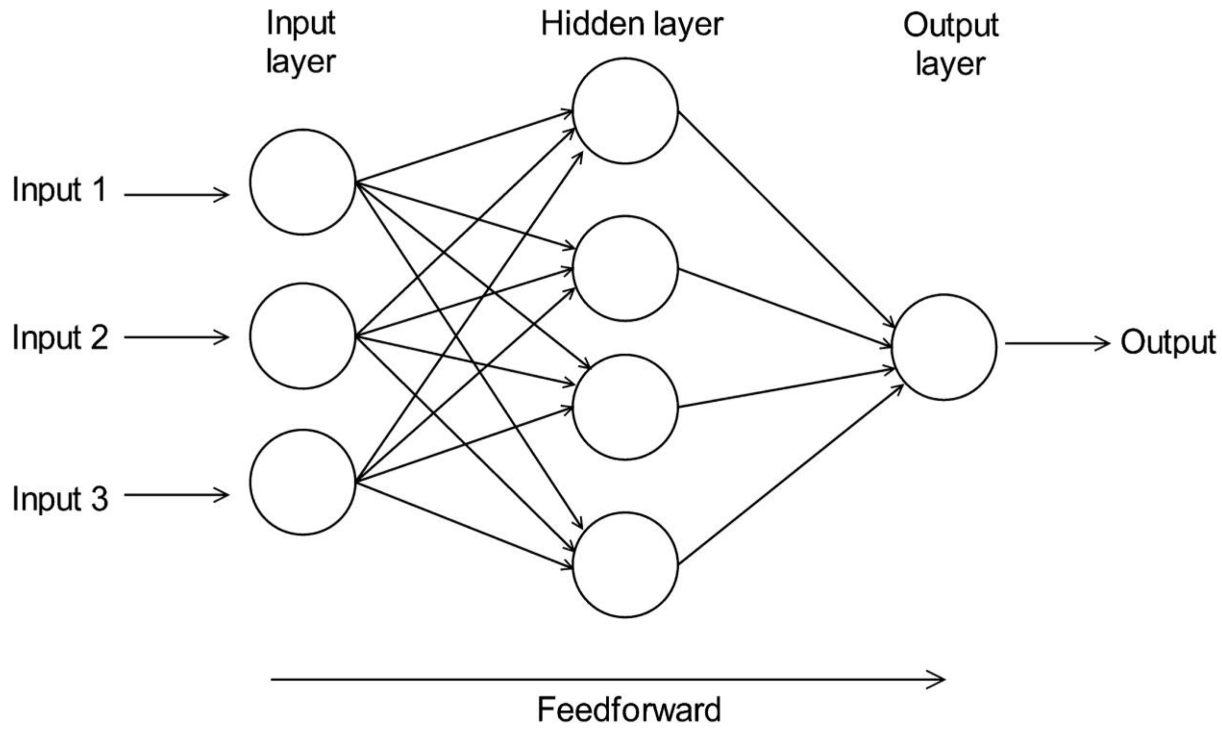
Stochastic Gradient Decent was much faster than the other algorithms but the results produced were far from optimum. Both, Adagrad and Adam produced better results that SGD, but they were computationally extensive. Adam was slightly faster than Adagrad. Thus, while using a particular optimization function, one has to make a trade off between more computation power and more optimum results.

Types of Neural Networks :

- Feed-forward Neural Network
- Recurrent Neural Network
- Convolutional Neural Network

Feed-forward neural networks :

- Feed-forward neural networks are inspired by the information processing of one or more neural cells, called a neuron.



Like the human brain, this process relies on many individual neurons in order to handle and process larger tasks. As the individual networks perform their tasks independently, the results can be combined at the end to produce a synthesized, and cohesive output.

It consists of a (possibly large) number of simple neuron-like processing units, organized in layers. Every unit in a layer is connected with all the units in the previous layer. This is why they are called feed forward neural networks.

In this neural network information is only processed in one direction , hence the connections between nodes does not form a cycle. The opposite of a feed forward neural network is a recurrent neural network, in which certain pathways are cycled.

Working :

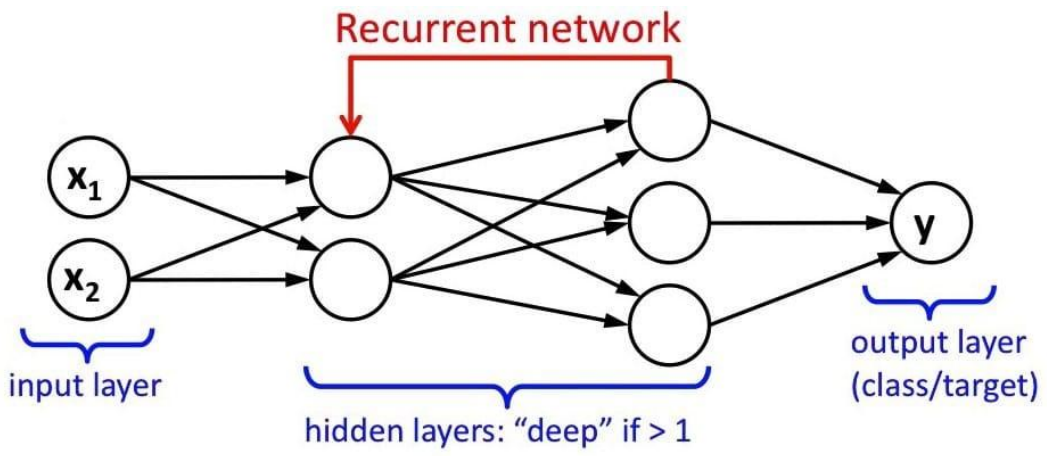
A Feed Forward Neural Network is commonly seen in its simplest form as a single layer perceptrons. In this model, a series of inputs enter the layer and are multiplied by the weights. Each value is then added together to get a sum of the weighted input values. If the sum of the values is above a specific threshold, usually set at zero, the value produced is often 1, whereas if the sum falls below the threshold, the output value is -1.

The single layer perceptron is an important model of feed forward neural networks and is often used in classification tasks. Furthermore, single layer perceptrons can incorporate aspects of machine learning. Using a property known as the delta rule, the neural network can compare the outputs of its nodes with the intended values, thus allowing the network to adjust its weights through training in order to produce more accurate output values.

This process of training and learning produces a form of a gradient descent. In multi-layered perceptrons, the process of updating weights is nearly analogous, however the process is defined more specifically as back-propagation. In such cases, each hidden layer within the network is adjusted according to the output values produced by the final layer.

Recurrent Neural Network

The main and most important feature of RNN is Hidden state, which remembers some information about a sequence. Suppose there is a deeper network with one input layer, three hidden layers and one output layer. Then like other neural networks, each hidden layer will have its own set of weights and biases, let’s say, for hidden layer 1 the weights and biases are (w1, b1), (w2, b2) for second hidden layer and (w3, b3) for third hidden layer. This means that each of these layers are independent of each other i.e they do not memorize the previous outputs.



- Now the RNN will do the following:
- RNN converts the independent activations into dependent activations by providing the same weights and biases to all the layers, thus reducing the complexity of increasing parameters and memorizing each previous outputs by giving each output as input to the next hidden layer.
- Hence these three layers can be joined together such that the weights and bias of all the hidden layers is the same, into a single recurrent layer.

- **Training through RNN**
    - A single time step of the input is provided to the network.
    - Then calculate its current state using set of current input and the previous state.
    - The current h(t) becomes h(t)-1 for the next time step.
    - One can go as many time steps according to the problem and join the information from all the previous states.
    - Once all the time steps are completed the final current state is used to calculate the output.

- The output is then compared to the actual output i.e. the target output and the error is generated.
- The error is then back-propagated to the network to update the weights and hence the network (RNN) is trained.

- **Advantages of Recurrent Neural Network :**

An RNN remembers each and every information through time. It is useful in time series prediction only because of the feature to remember previous inputs as well. This is called Long Short Term Memory.

Two issues of Standard RNN's

- Gradient is partial derivative with respect to its input.
- Gradient measures how much the output of a function changes if you change the inputs a little bit
- Gradient is a slope of function.
- Gradient measures change in all weights with regard to the change in error.

1) Exploding Gradients : Assigns stupidly high importance to weights, this problem is solved by truncating or squashing the gradients.
2) Vanishing Gradients : occurs when values of gradient are too small and model stops learning.

$$Wnew = Wold - \eta \frac{\partial L}{\partial Wold}$$

Where,

η = Learning Rate (1)

W_{new} = Update weight

W_{old} = Old weight

dL/dW_{old} = derivative of loss with respect to old weight

As, sigmoid function lies between 0 and 1.

$$0 < \frac{\partial L}{\partial Wold} < 0.25$$

$W_{new} \sim W_{old}$

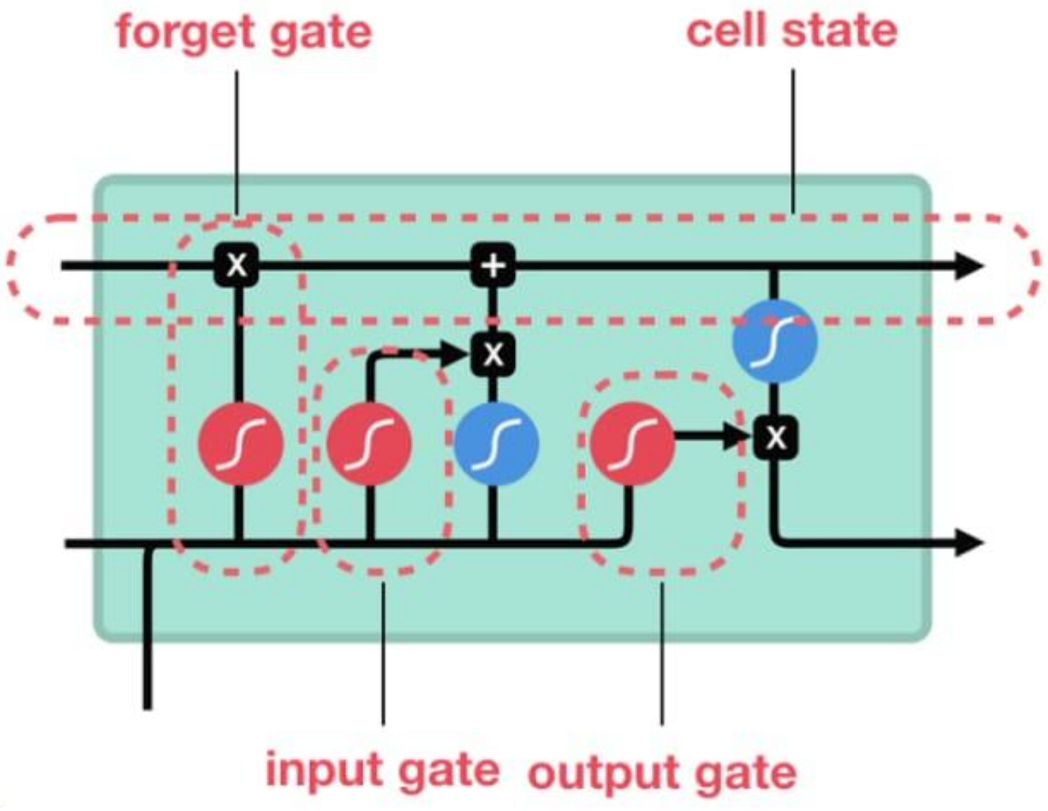
Hence the weights are not update well.

Such problem is solved by Long Short Term Memory (LSTM)

Long Short Term Memory (LSTM) :

- Extends the Memory
- It is well suited to learn from important experiences that have very long time lags in between.
- LSTM contain information in a memory, much like a memory of a computer.
- LSTM can read, write and delete information from its memory.
- Memory can be seen as a gated cell. Based on importance it decides whether or not to store or delete information.

LSTM can choose information which is relevant to remember or forget during sequence processing. LSTM'S internal mechanisms called gates that can regulate the flow of information.These gates can learn which data in a sequence is important to keep or throw away. By doing that, it can pass relevant information down the long chain of sequences to make predictions.



Working of LSTM:

The gates in an LSTM are analog in the form of sigmoid, because they enable to do backpropogation .The problem of vanishing gradients is solved by LSTM because it keeps the gradients steep enough, which keeps the training relatively short and the accuracy high.

- Forget Gate:

This gate decides what information should be thrown away or kept. Information from the previous hidden state and information from the current input is passed through the sigmoid function. Values come out between 0 and 1. The closer to 0 means to forget, and the closer to 1 means to keep.

- Input Gate :

We pass the previous hidden state and current input into a sigmoid function. That decides which values will be updated by transforming the values to be between 0 and 1. 0 means not important, and 1 means important. You also pass the hidden state and current input into the tanh function to squish values between -1 and 1 to help regulate the network. Then you multiply the tanh output with the sigmoid output. The sigmoid output will decide which information is important to keep from the tanh output.

- CELL State :

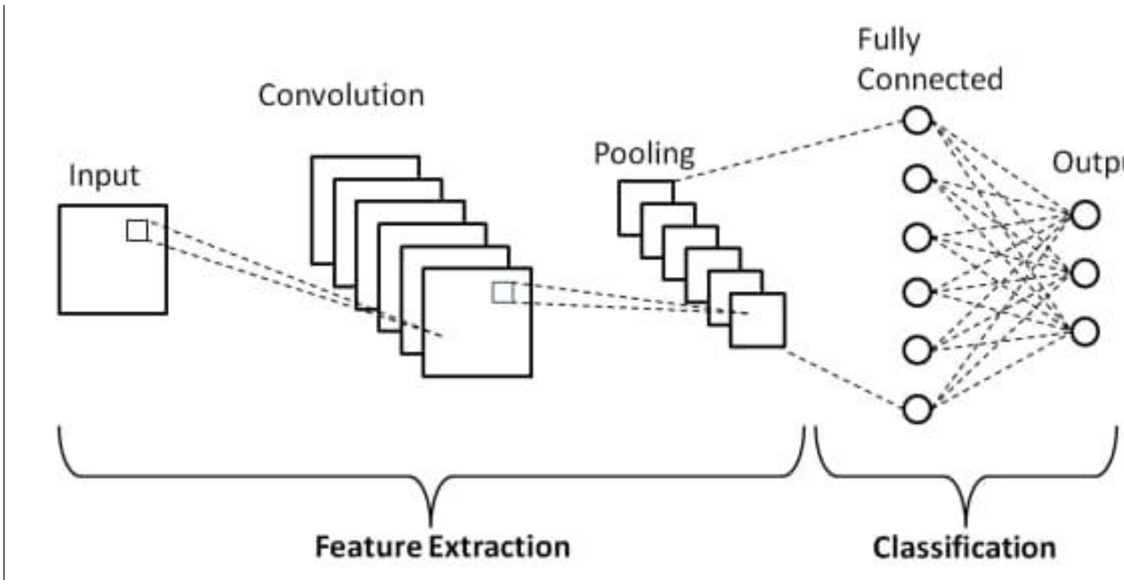
The cell state gets point wise multiplied by the forget vector. This has a possibility of dropping values in the cell state if it gets multiplied by values near 0. Then we take the output from the input gate and do a point wise addition which updates the cell state to new values that the neural network finds relevant. That gives us our new cell state.

- Output Gate:

The output gate decides what the next hidden state should be. Remember that the hidden state contains information on previous inputs. The hidden state is also used for predictions. First, we pass the previous hidden state and the current input into a sigmoid function. Then we pass the newly modified cell state to the tanh function.

Convolutional Neural Network:

(ConvNet/CNN) is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other. The pre-processing required in a ConvNet is much lower as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training, ConvNets have the ability to learn these filters/characteristics.



The architecture of a ConvNet is analogous to that of the connectivity pattern of Neurons in the Human Brain and was inspired by the organization of the Visual Cortex. Individual neurons respond to stimuli only in a restricted region of the visual field known as the Receptive Field.

Survival Analysis :

Survival analysis is a collection of statistical procedures for data analysis where the outcome variable of interest is time until an event occurs. Because of censoring–the nonobservation of the event of interest after a period of follow-up–a proportion of the survival times of interest will often be unknown. It is assumed that those patients who are censored have the same survival prospects as those who continue to be followed, that is, the censoring is uninformative.

Survival data are generally described and modelled in terms of two related functions, the survivor function and the hazard function. The survivor function represents the probability that an individual survives from the time of origin to some time beyond time t. It directly describes the survival experience of a study cohort, and is usually estimated by the KM method. The logrank test may be used to test for differences between survival curves for groups, such as treatment arms.

The hazard function gives the instantaneous potential of having an event at a time, given survival up to that time. It is used primarily as a diagnostic tool or for specifying a mathematical model for survival analysis. In comparing treatments or prognostic groups in terms of survival, it is often necessary to adjust for patient-related factors that could potentially affect the survival time of a patient. Failure to adjust for confounders may result in spurious effects.

Survivor Function :

The survivor function (aka. survival function, reliability function) is denoted as $S(t)$. The survivor function gives the probability that a person survives longer than some specific time t $S(t)=P(T>t)$. All survivor functions follow these same 3 characteristics:

- As t increases, the $S(t)$ should decrease.
- $S(0)=1$. That is at the start of the study, no one has the event and hence the probability of surviving at $t=0$ is 1.

- $S(\infty)=0$. If the study were to go to $S(\infty)$, then everyone will eventually experience the event and hence the survival probability must be 0.

In theory, survival curves should be a "smooth" function with time ranging from 0 to ∞:

Hazard function :

The hazard function specifies the instantaneous rate of failure at T = t given that the individual survived up to time t

$$h(t) = \lim_{\Delta t \to 0} \left(\frac{P\{ t \leq T < t + \Delta t \mid T \geq t \}}{\Delta t} \right)$$

Where,

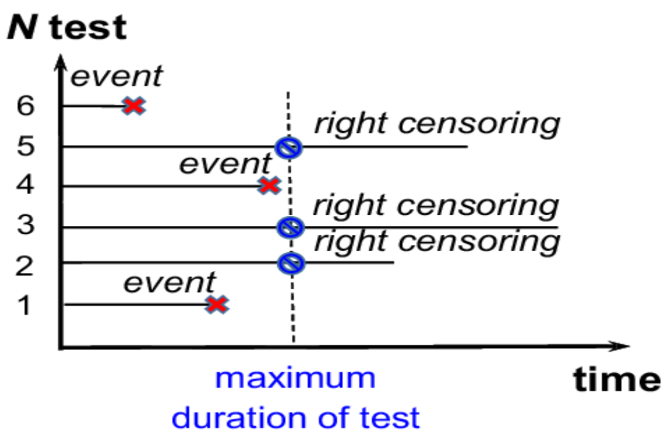
- $\lim \Delta t \to \infty$: This indicates as the time interval approaches 0. This essentially gives us the instantaneous measurement at a particular time.
- $P(t \leq T < t+\Delta t \mid T \geq t)$: Probability that a person's survival time T will be between the time interval t and t+Δt given that they have survived up to t (T≥t).

We can think of it as the probability of failure in an infinitesimally small time period between t and t + Δt given that the subject has survived up till time t. In this sense, the hazard is a measure of risk: the greater the hazard between times t1 and t2, the greater the risk of failure in this time interval.

Censoring:

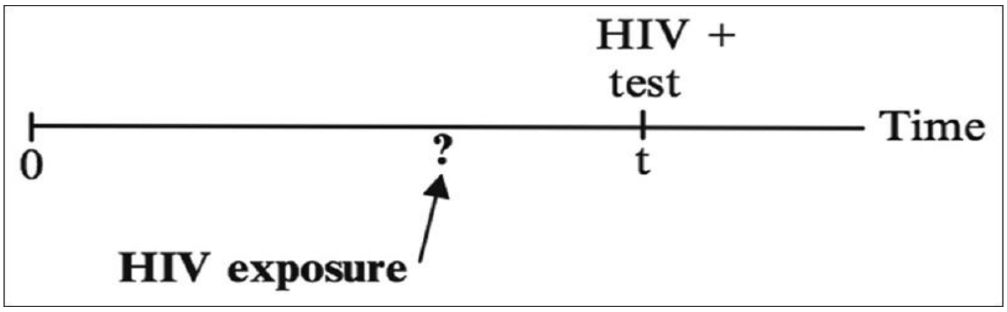
Censoring can arise in different situations and can be classified in several ways:

Right Censoring :



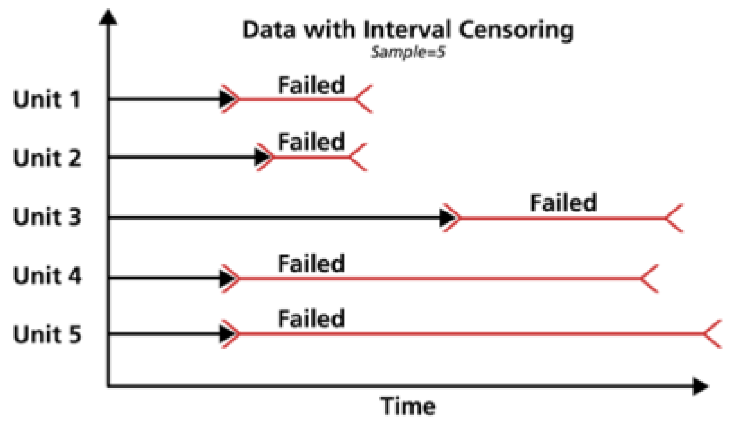
A data point is above a certain value but it is unknown by how much . For example, we consider patients in a clinical trial to study the effect of treatments on stroke occurrence. Those patients who have had no strokes by the end of the year are censored.

Left Censoring:



A data point is below a certain value but it is unknown by how much,e.g. timespent for acquiring a skill in a study.

Interval Censoring :

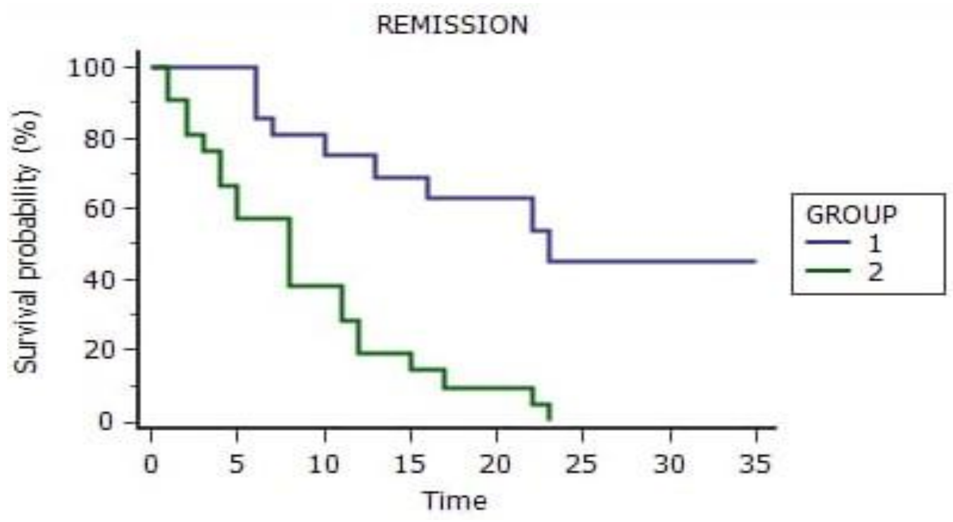


A data point is somewhere in an interval between two values,

e.g., time between visits to a doctor.

➢ **Right Censoring Models :**

- **Kaplan–Meier model** :



The Kaplan–Meier estimator, also known as the product limit estimator, is a non-parametric statistic used to estimate the survival function from lifetime data. K-M model is only useful when predictor variable is categorical . In medical research, it is often used to measure the fraction of patients living for a certain amount of time after treatment. In other fields, Kaplan–Meier estimators may be used to measure the length of time people remain unemployed after a job loss. K-M model cannot estimate Hazard Ratio.

It shows what the probability of an event (for example, survival) is at a certain time interval. If the sample size is large enough, the curve should approach the true survival function for the population under investigation. Observations are called censored when information about their survival time is incomplete.

Assumptions :

• Where possible, left-censoring should be minimized or avoided.

• There should be independence of censoring and the event

• There should be no secular trends .

Formula for Kaplan-Meier is as follows :

$$\widehat{S(t)} = \prod_{t_i<t} \frac{n_i - d_i}{n_i}$$

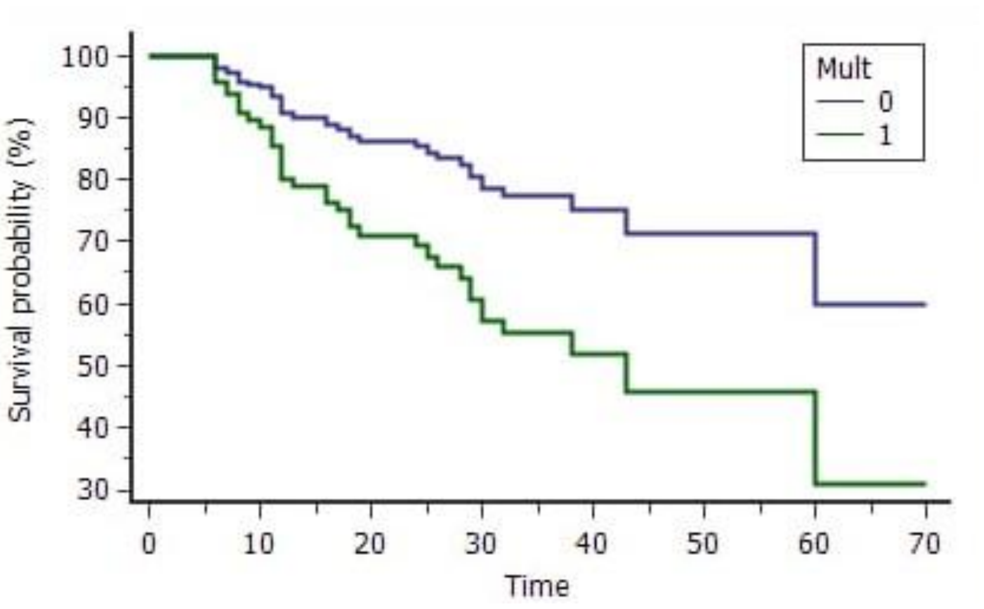
We can also write it as :

$$S(t_i) = S(t_{i-1}) * \left(1 - \frac{d_i}{n_i}\right)$$

Where,

$S(t_{i-1})$	=	the probability of being alive at t_{i-1}
n_i	=	the number of patients alive just before t_i
d_i	=	the number of events at t_i
t_0	=	0
$S(0)$	=	1

Cox proportional-hazards model :



Formula for Cox proportional-hazards model is as follows :

$$h(t) = h_0(t) \times \exp(b_1x_1 + b_2x_2 + \cdots + b_px_p)$$

Where,

- t represents the survival time
- h(t) is the hazard function determined by a set of p covariates (x1,x2,...,xp)
- the coefficients (b1,b2,...,bp) measure the impact (i.e., the effect size) of covariates.
- the term h0 is called the baseline hazard. It corresponds to the value of the hazard if all the xi are equal to zero (the quantity exp(0) equals 1). The 't' in h(t) reminds us that the hazard may vary over time.

The Cox proportional-hazards model is essentially a regression model commonly used statistical in medical research for investigating the association between the survival time of patients and one or more predictor variables. Cox regression is method for investigating the effect of several variables upon the time a specified event takes to happen. In the context of an outcome such as death this is known as Cox regression for survival analysis.

Assumption :

- The ratio of the hazards for any two individuals is constant over time.
- The explanatory variables act multiplicatively on the Hazard function.
- Failure time of individual subjects are independent of each other.

Competing Risks:

Competing Risks in Survival Analysis So far, we've assumed that there is only one survival endpoint of interest, and that censoring is independent of the event of interest. However, in many contexts it is likely that we can have several different types of failure (death, relapse, opportunistic infection, etc) that are of interest to us, and the occurrence of one type of failure may (or may not) prevent us from observing the other types of failures socalled"competing"risks.

Examples:

• In cancer studies, deaths from other causes (such as heart disease, diabetes, etc.) are consideredcompetingrisks.

• After a bone marrow transplantation, patients are followed to evaluate "leukemia-free survival", so the endpoint is time to leukemia relapse or death, whichever occurs first. This endpoint consists of two types of failures (competing risks):

– leukemia relapse

– non-relapse deaths

Risks Data

Competing risks occur when there are at least two possible ways that a person can fail, but only one such failure type can actually occur.

For example,

1. A person can die from lung cancer or from a stroke, but not from both (although he can have both lung cancer and atherosclerosis before he dies);

TIME TO EVENT MODELS :

Exponential Model:

let T= Time until Next event occurs (survival time)

Here , we fix the event (y=1) and check the time required to complete the event .

S(T)=e-(haz)*t

we can use regression

Haz=eb0+b1x1+….+bkxk

ln(Haz)=b0+b1x1+b2x2+…+bkxk

Weibull model :

It can accurately model the time-to-failure of real-world events and is sufficiently flexible despite having only two parameters.

Competing risks : System fails when only one failure occurs .

Bayesian regression models for competing risks :

We present a Bayesian approach for analysis of competing risks survival data with masked causes of failure. This approach is often used to assess the impact of covariates on the hazard functions when the failure time is exactly observed for some subjects but only known to lie in an interval of time for the remaining subjects.
As one typical multi-model method, Bayesian model averaging (BMA) has recently gained popularity in various fields because it can produce more adaptive and reliable predictions than other techniques

Posterior distribution :

Combination of likelihood and prior Bayesian methods sample from this posterior distribution and create summaries of distributions to make parameter inference using summaries allows us to make inferences about a what values parameters might take.

Classical regression models for competing risks :

Competing risks are the single components of such a composite time-to-event endpoint . Competing risks considers time-until-first-event and type-of-first-event . A competing risks analysis therefore provides for more specific results

BinomialRegressionModel:

The Binomial Regression model can be used for predicting the odds of seeing an event, given a vector of regression variables. For e.g. one could use the Binomial Regression model to predict the odds of its starting to rain in the next 2 hours, given the current temperature, humidity, barometric pressure, time of year, geo-location, altitude etc.In a Binomial Regression model, the dependent variable y is a discrete random variable that takes on values such as 0, 1, 5, etc. Each value represents the number of 'successes' observed in m trials. Thus y follows the binomial distribution.

Pseudo Value Regression Model:

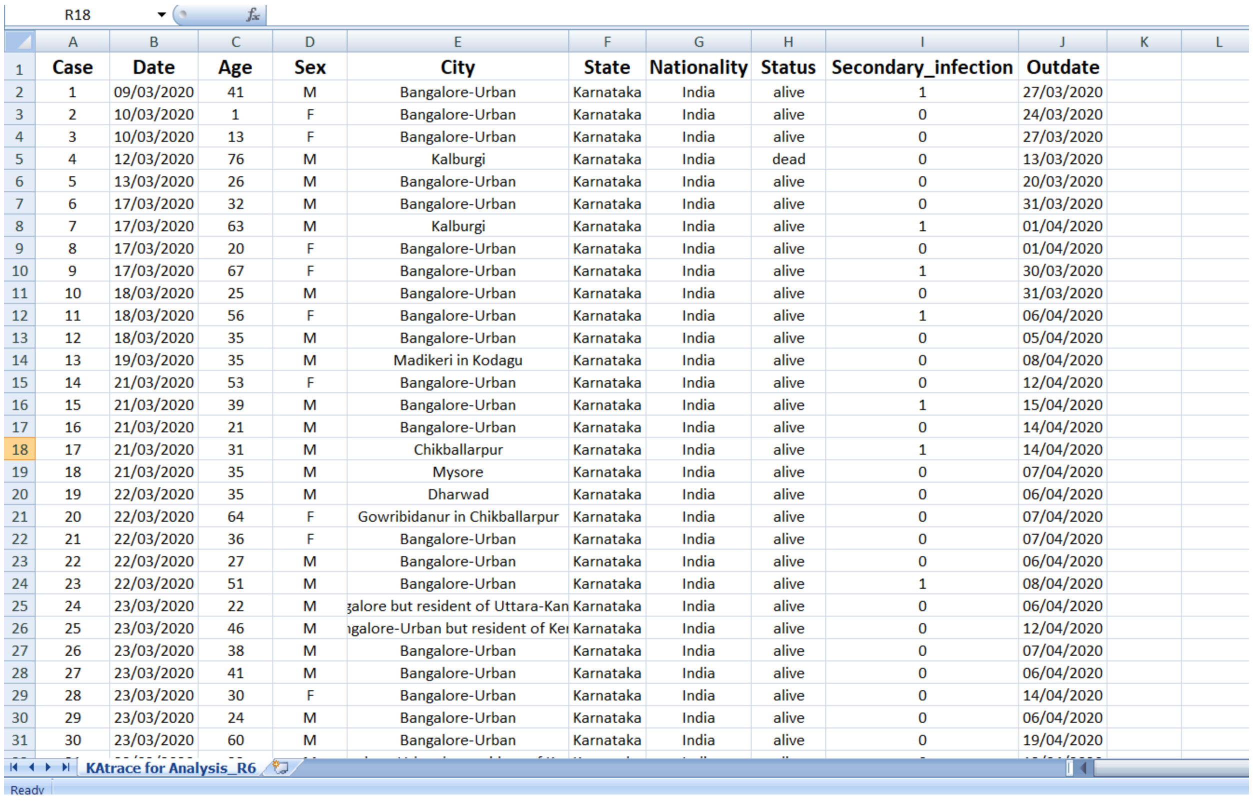
We review an alternative approach to inference with incomplete survival data, based on pseudo-values or pseudo-observations obtained from a jackknife statistic constructed from non-parametric estimators for the quantity of interest. These pseudo-values are then used as outcome variables in a generalized linear model and model parameters are estimated using generalized estimating equations

- **Data Source :** The data of COVID-19 patients of Karnataka state is collected from the website

  https://www.isibang.ac.in/~athreya/incovid19/data.htmlwebsite

  Which is official website of Indian Statistical Institute , Bangalore centre .

Data description :



Case	Date	Age	Sex	City	State	Nationality	Status	Secondary_infection	Outdate
1	09/03/2020	41	M	Bangalore-Urban	Karnataka	India	alive	1	27/03/2020
2	10/03/2020	1	F	Bangalore-Urban	Karnataka	India	alive	0	24/03/2020
3	10/03/2020	13	F	Bangalore-Urban	Karnataka	India	alive	0	27/03/2020
4	12/03/2020	76	M	Kalburgi	Karnataka	India	dead	0	13/03/2020
5	13/03/2020	26	M	Bangalore-Urban	Karnataka	India	alive	0	20/03/2020
6	17/03/2020	32	M	Bangalore-Urban	Karnataka	India	alive	0	31/03/2020
7	17/03/2020	63	M	Kalburgi	Karnataka	India	alive	1	01/04/2020
8	17/03/2020	20	F	Bangalore-Urban	Karnataka	India	alive	0	01/04/2020
9	17/03/2020	67	F	Bangalore-Urban	Karnataka	India	alive	1	30/03/2020
10	18/03/2020	25	M	Bangalore-Urban	Karnataka	India	alive	0	31/03/2020
11	18/03/2020	56	F	Bangalore-Urban	Karnataka	India	alive	1	06/04/2020
12	18/03/2020	35	M	Bangalore-Urban	Karnataka	India	alive	0	05/04/2020
13	19/03/2020	35	M	Madikeri in Kodagu	Karnataka	India	alive	0	08/04/2020
14	21/03/2020	53	F	Bangalore-Urban	Karnataka	India	alive	0	12/04/2020
15	21/03/2020	39	M	Bangalore-Urban	Karnataka	India	alive	1	15/04/2020
16	21/03/2020	21	M	Bangalore-Urban	Karnataka	India	alive	0	14/04/2020
17	21/03/2020	31	M	Chikballarpur	Karnataka	India	alive	1	14/04/2020
18	21/03/2020	35	M	Mysore	Karnataka	India	alive	0	07/04/2020
19	22/03/2020	35	M	Dharwad	Karnataka	India	alive	0	06/04/2020
20	22/03/2020	64	F	Gowribidanur in Chikballarpur	Karnataka	India	alive	0	07/04/2020
21	22/03/2020	36	F	Bangalore-Urban	Karnataka	India	alive	0	07/04/2020
22	22/03/2020	27	M	Bangalore-Urban	Karnataka	India	alive	0	06/04/2020
23	22/03/2020	51	M	Bangalore-Urban	Karnataka	India	alive	1	08/04/2020
24	23/03/2020	22	M	galore but resident of Uttara-Kan	Karnataka	India	alive	0	06/04/2020
25	23/03/2020	46	M	ngalore-Urban but resident of Ke	Karnataka	India	alive	0	12/04/2020
26	23/03/2020	38	M	Bangalore-Urban	Karnataka	India	alive	0	07/04/2020
27	23/03/2020	41	M	Bangalore-Urban	Karnataka	India	alive	0	06/04/2020
28	23/03/2020	30	F	Bangalore-Urban	Karnataka	India	alive	0	14/04/2020
29	23/03/2020	24	M	Bangalore-Urban	Karnataka	India	alive	0	06/04/2020
30	23/03/2020	60	M	Bangalore-Urban	Karnataka	India	alive	0	19/04/2020

- Data set contains 26261 rows and 10 columns one row corresponds to one case in Karnataka state . For each patient the columns contains following information as

  "case number" , "Date" , "Age" , "Sex" , "City" , "State" , "Nationality" , "Status" , "Secondary infection" , "Out date"

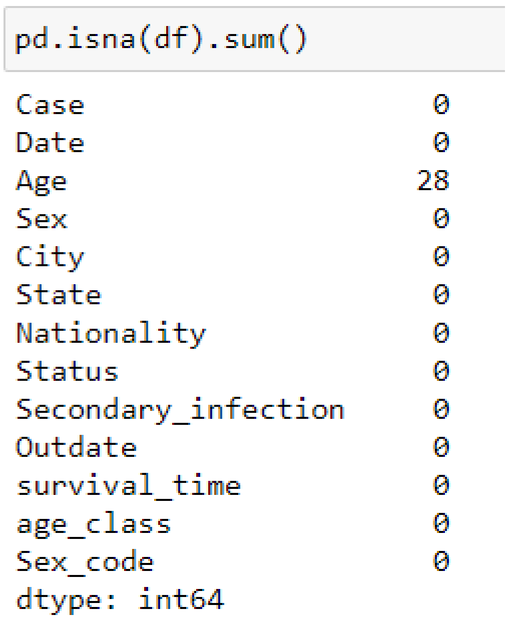
  Secondary infection indicates

  0 : no secondary infection

  1 : secondary infection

➢ Exploratory Data Analysis :

To check whether missing values:

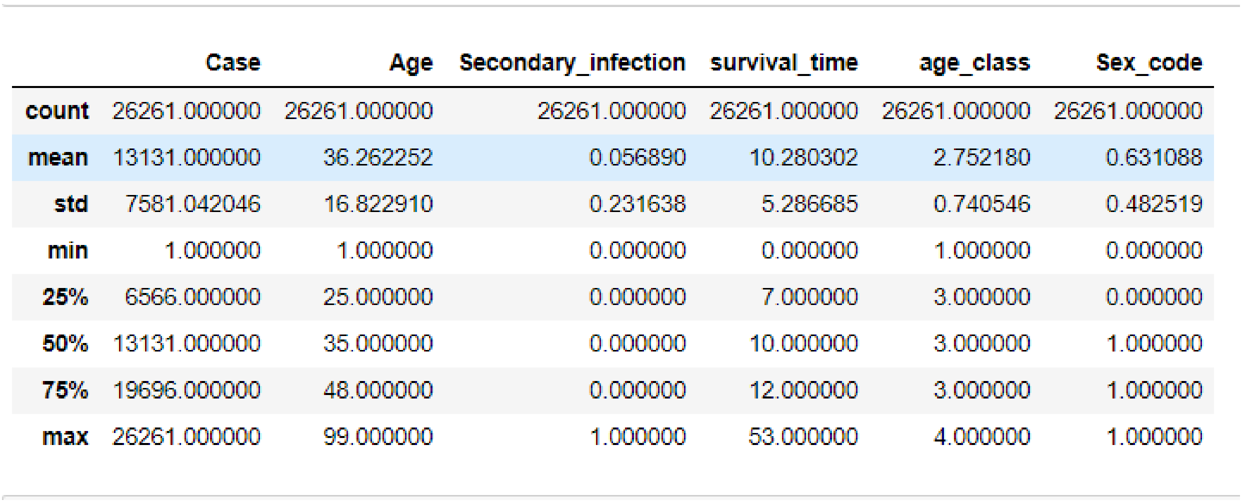


```
pd.isna(df).sum()
```

```
Case                    0
Date                    0
Age                    28
Sex                     0
City                    0
State                   0
Nationality             0
Status                  0
Secondary_infection     0
Outdate                 0
survival_time           0
age_class               0
Sex_code                0
dtype: int64
```

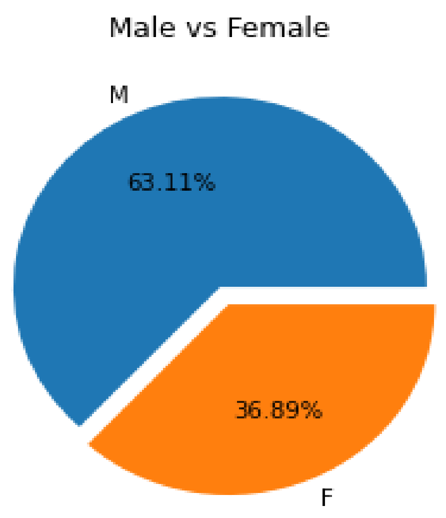
Only Age column has 28 missing values

- **Summary Statistics**



	Case	Age	Secondary_infection	survival_time	age_class	Sex_code
count	26261.000000	26261.000000	26261.000000	26261.000000	26261.000000	26261.000000
mean	13131.000000	36.262252	0.056890	10.280302	2.752180	0.631088
std	7581.042046	16.822910	0.231638	5.286685	0.740546	0.482519
min	1.000000	1.000000	0.000000	0.000000	1.000000	0.000000
25%	6566.000000	25.000000	0.000000	7.000000	3.000000	0.000000
50%	13131.000000	35.000000	0.000000	10.000000	3.000000	1.000000
75%	19696.000000	48.000000	0.000000	12.000000	3.000000	1.000000
max	26261.000000	99.000000	1.000000	53.000000	4.000000	1.000000

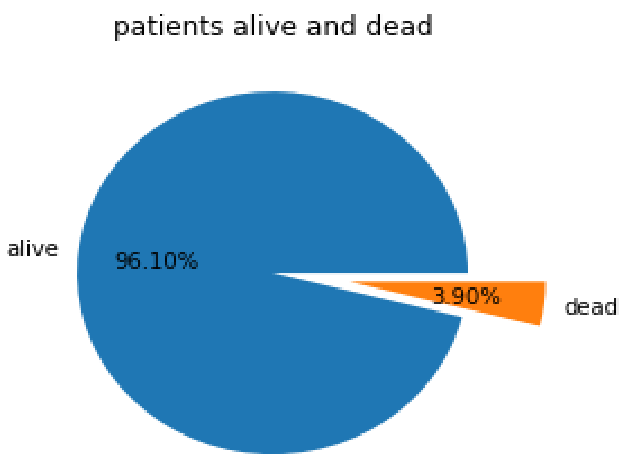
- **Pie chart for Gender**



Data contains 63.11 % Male patients

and 36.89 % of Female patients.

Status of patients

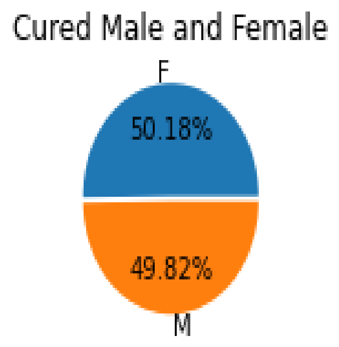


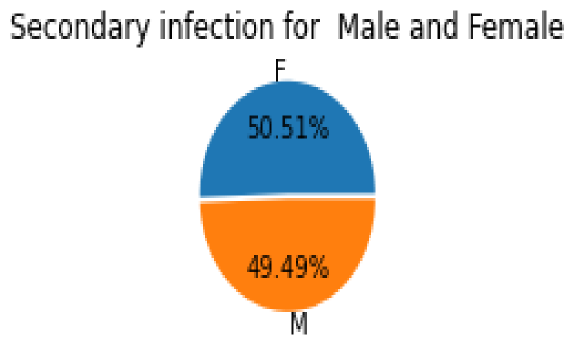
96.10 % patients are alive and 3.90 % patients are dead by corona virus disease.

- **Crosstab of Status vs Gender and Secondary infection vs Gender.**

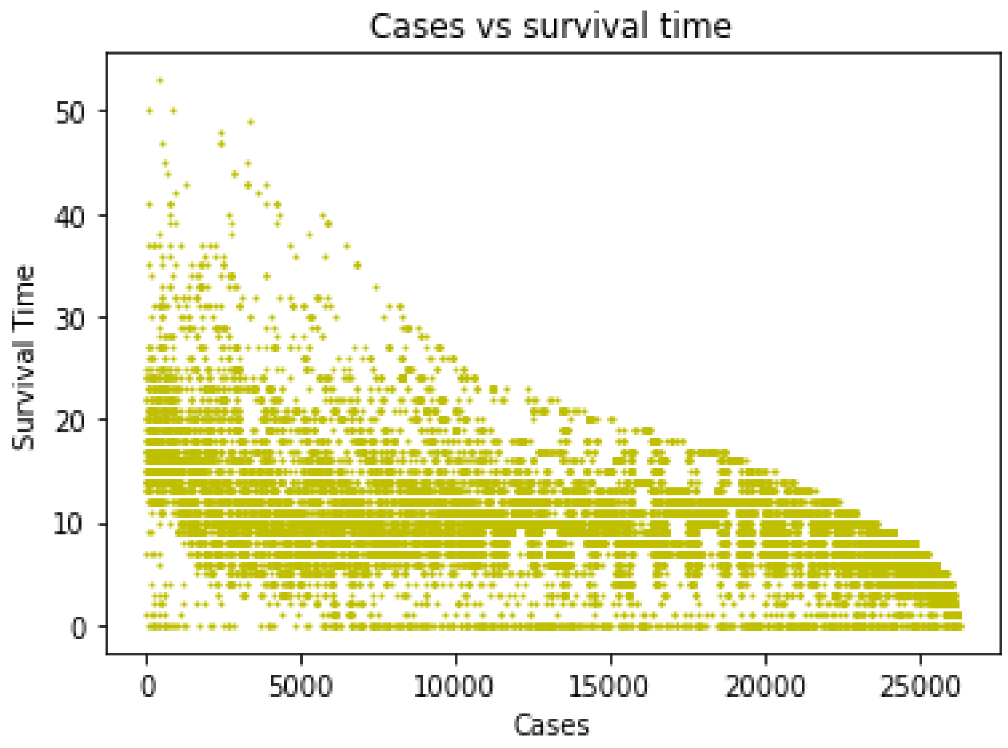
Status	alive	dead	All
Sex			
F	9354	334	9688
M	15884	689	16573
All	25238	1023	26261

Secondary_infection	0	1	All
Sex			
F	9255	433	9688
M	15512	1061	16573
All	24767	1494	26261





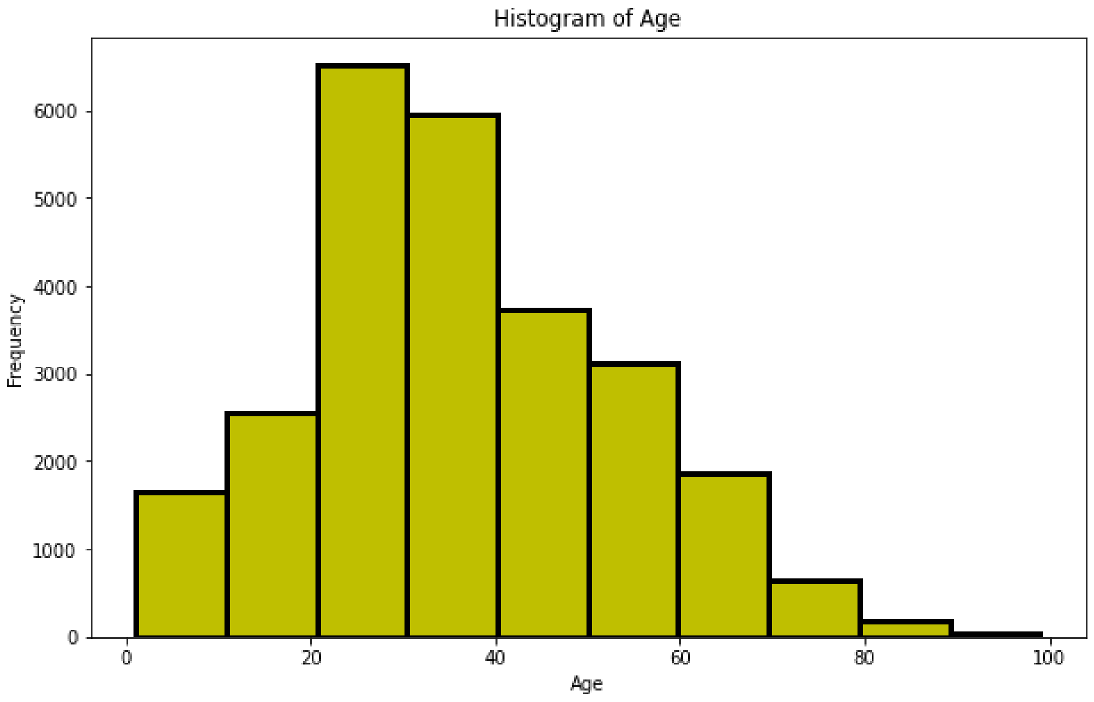
- **Scatter diagram of Cases vs Survival time**



.

Here , we say that as Cases increases (time passes) Survival time decreases .

- **Histogram of Age**



Maximum number of corona virus patients are in 20-40 age group

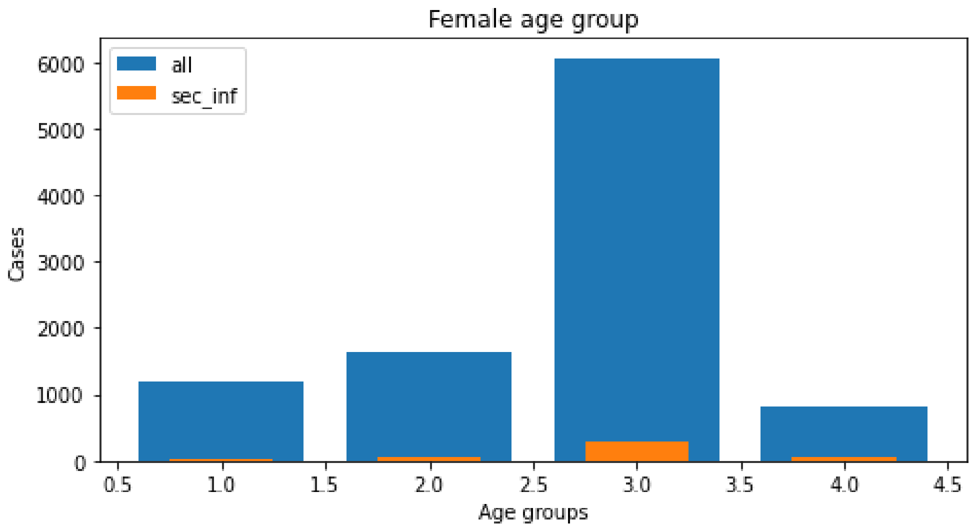
- Bar chart of Age groups in Four categories for Male and Female

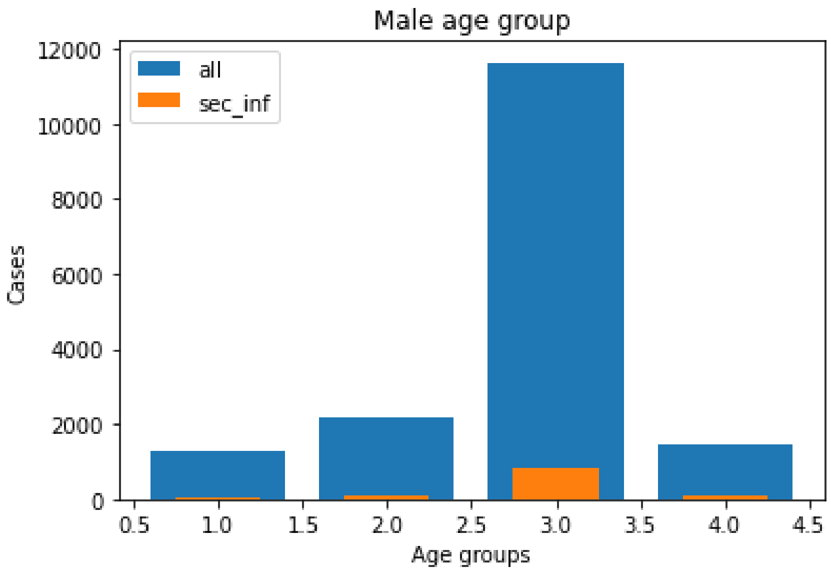
  1 : Children(0-14 years)

  2 : Youth(15-24 years)

  3 : Adult(25-60 years)

  4 : Seniors(60 and over years)

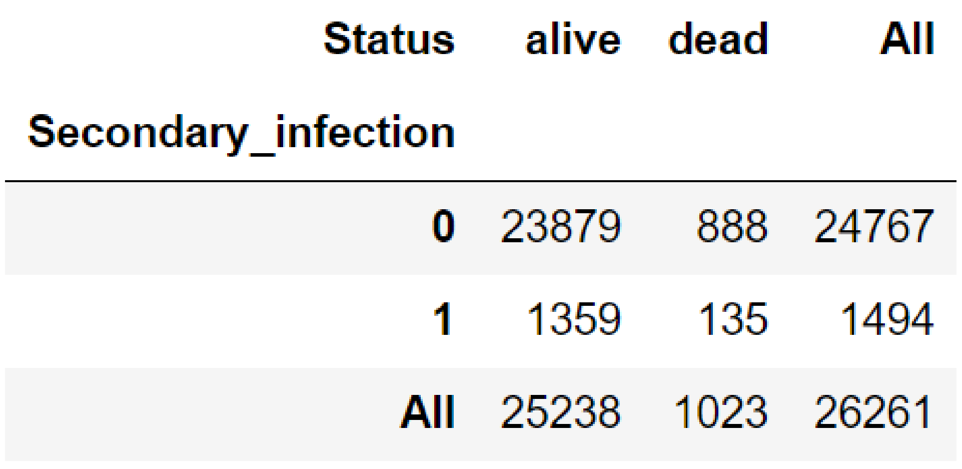




In both cases, majority of patients caused by corona virus are Adults

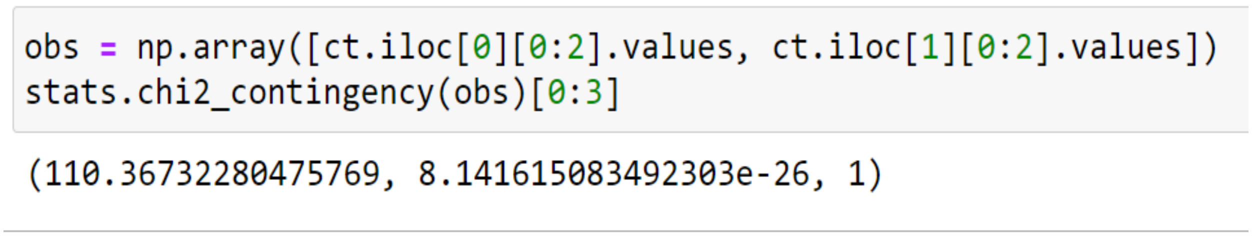
➢ Chi-square test for association :

- The chi-square tes of independence is used to determine there is a significant relationship between two nominal (categorical) variables.
- The frequency of each category for one nominal variable is compared across the categories of the second nominal variable.
- The data can be displayed in a contingency table where each row represents a category for one variable and each column represents a category for the other variable.



Status	**alive**	**dead**	**All**
Secondary_infection			
0	23879	888	24767
1	1359	135	1494
All	25238	1023	26261

- Hypothesis :
  - H_0 : There is no significant relationship between ‘Status of patient’ and ‘Secondary infection’
  - H_1 : There is significant relationship between ‘Status of patient’ and ‘Secondary infection’.



```
obs = np.array([ct.iloc[0][0:2].values, ct.iloc[1][0:2].values])
stats.chi2_contingency(obs)[0:3]

(110.36732280475769, 8.141615083492303e-26, 1)
```

Here , P-value < 0.05 with 1 degrees of freedom ,

Hence we reject H0 at 5% level of significance .

i.e., There is Significant relationship between 'Status of patient' and 'Secondary infection' .

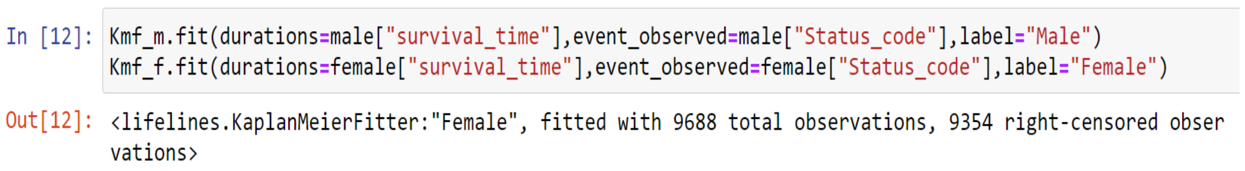
➢ Survival Models :

- Kaplan-Meier estimator :

We require lifelines library

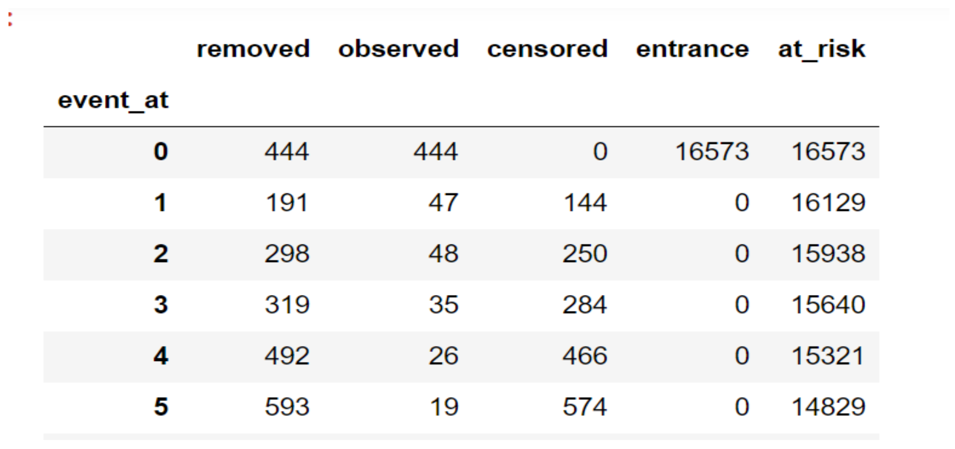
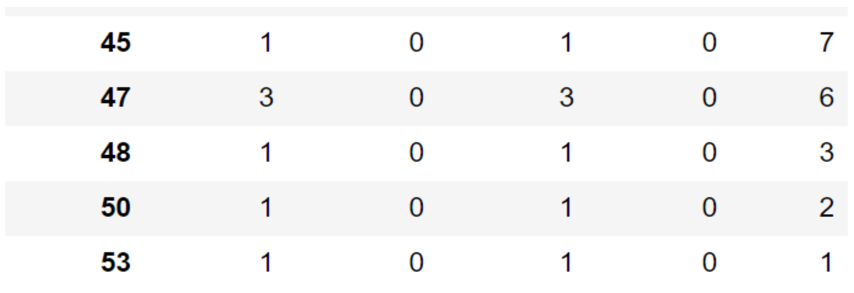
From lifelines import KaplanMeierFitter

Event Table : one most important method of kmf object is event table it gives various information for data fitted .



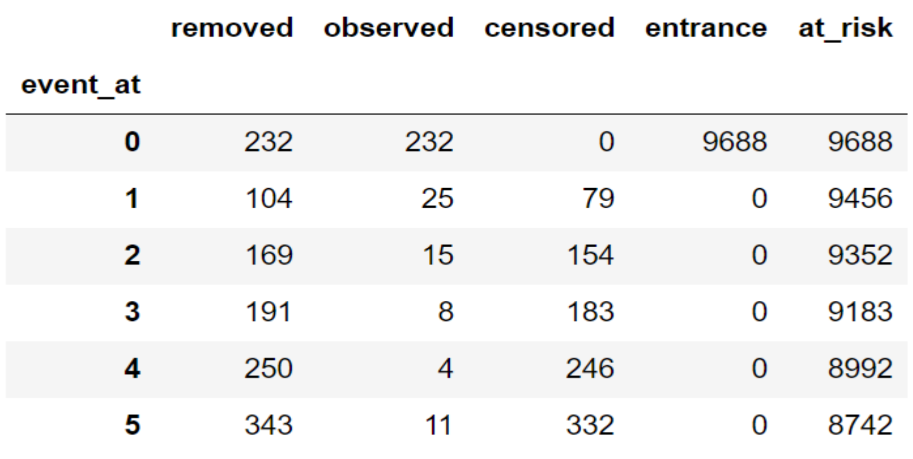
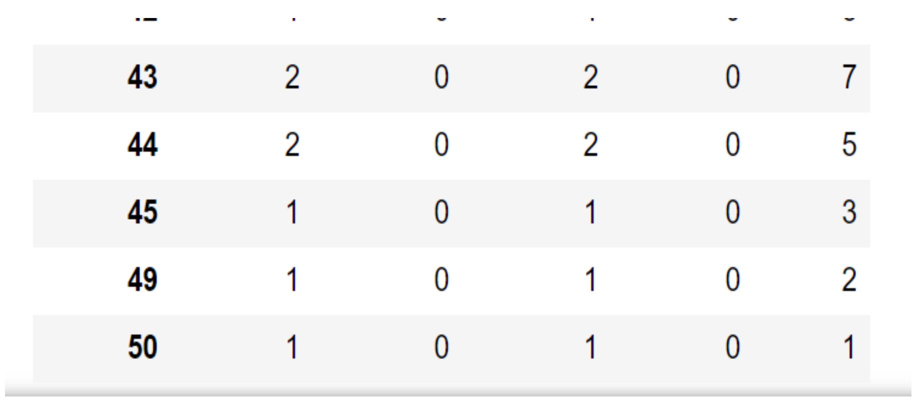
```
In [12]: Kmf_m.fit(durations=male["survival_time"],event_observed=male["Status_code"],label="Male")
         Kmf_f.fit(durations=female["survival_time"],event_observed=female["Status_code"],label="Female")

Out[12]: <lifelines.KaplanMeierFitter:"Female", fitted with 9688 total observations, 9354 right-censored obser
         vations>
```

Event table for male

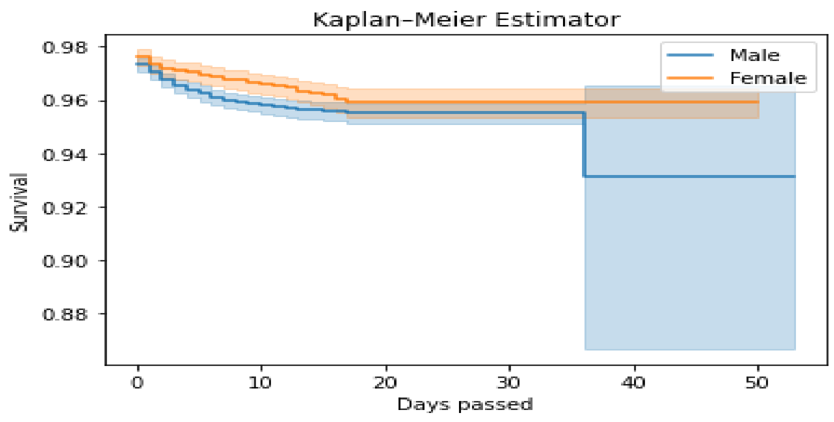
event_at	removed	observed	censored	entrance	at_risk
0	444	444	0	16573	16573
1	191	47	144	0	16129
2	298	48	250	0	15938
3	319	35	284	0	15640
4	492	26	466	0	15321
5	593	19	574	0	14829
45	1	0	1	0	7
47	3	0	3	0	6
48	1	0	1	0	3
50	1	0	1	0	2
53	1	0	1	0	1

Event table for Female

event_at	removed	observed	censored	entrance	at_risk
0	232	232	0	9688	9688
1	104	25	79	0	9456
2	169	15	154	0	9352
3	191	8	183	0	9183
4	250	4	246	0	8992
5	343	11	332	0	8742
43	2	0	2	0	7
44	2	0	2	0	5
45	1	0	1	0	3
49	1	0	1	0	2
50	1	0	1	0	1

KMF Survival Graph (Male , Female) :



Notice that probability of female patient surviving in covid pandemic is higher than male patients

Log-Rank test : The log rank test is a popular test to test the null hypothesis of no difference in survival between two or more independent groups.

H_0 : There is no significant difference between the groups being studied.

H_1 : There is significant difference between the groups being studied.

```
from lifelines.statistics import logrank_test
results = logrank_test(T,T1,event_observed_A=E,event_observed_B=E1)
results.print_summary()
```

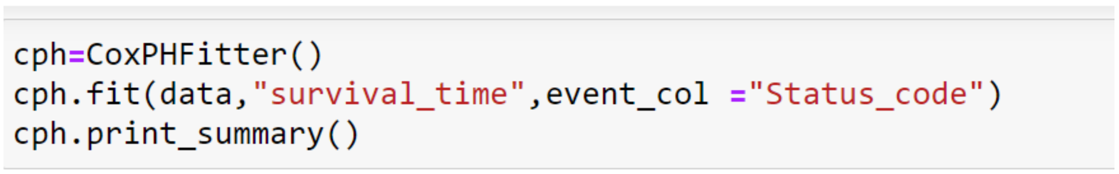
t_0	-1
null_distribution	chi squared
degrees_of_freedom	1
test_name	logrank_test

	test_statistic	p	-log2(p)
0	8.26	<0.005	7.95

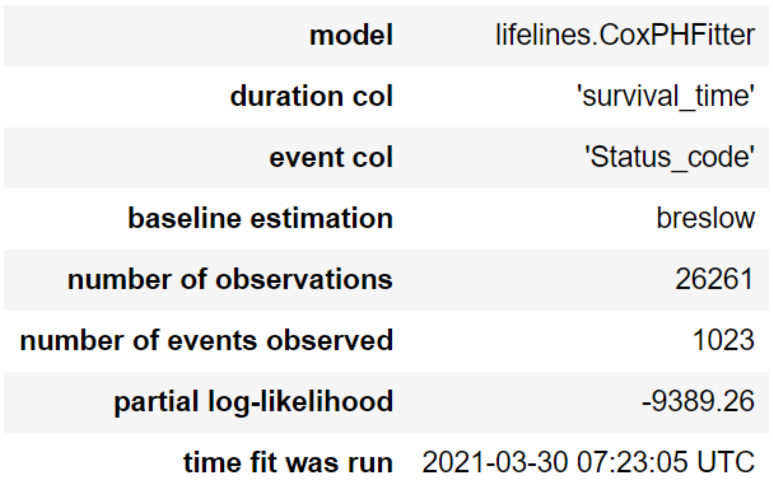
Here, P-value < 0.05 with 1 degrees of freedom ,hence we reject H_0

i.e., Gender is associated with survival days

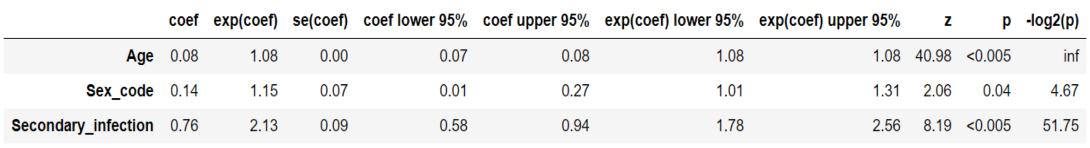
➢ Cox-proportional Hazard Model :



```
cph=CoxPHFitter()
cph.fit(data,"survival_time",event_col ="Status_code")
cph.print_summary()
```



model	lifelines.CoxPHFitter
duration col	'survival_time'
event col	'Status_code'
baseline estimation	breslow
number of observations	26261
number of events observed	1023
partial log-likelihood	-9389.26
time fit was run	2021-03-30 07:23:05 UTC



	coef	exp(coef)	se(coef)	coef lower 95%	coef upper 95%	exp(coef) lower 95%	exp(coef) upper 95%	z	p	-log2(p)
Age	0.08	1.08	0.00	0.07	0.08	1.08	1.08	40.98	<0.005	inf
Sex_code	0.14	1.15	0.07	0.01	0.27	1.01	1.31	2.06	0.04	4.67
Secondary_infection	0.76	2.13	0.09	0.58	0.94	1.78	2.56	8.19	<0.005	51.75

HR = 1 : No Effect.

HR < 1 : Reduction in the Hazard.

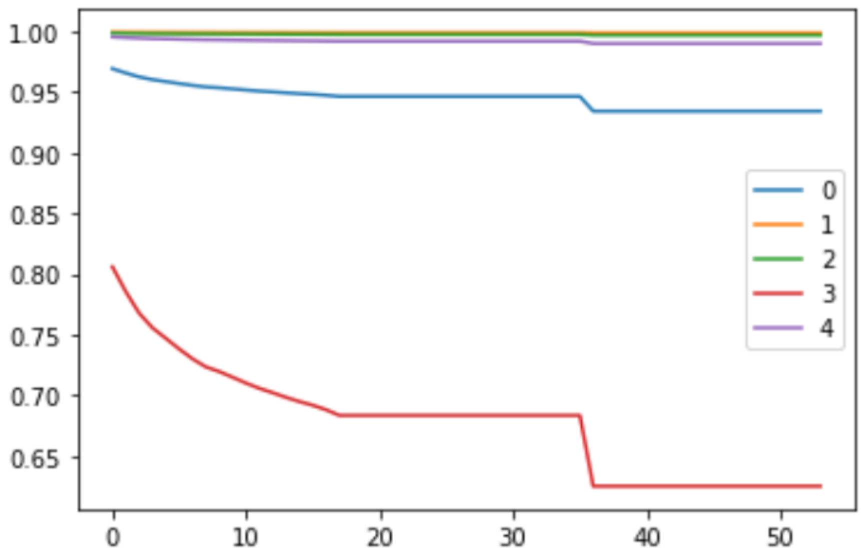
HR > 1 : Increase in Hazard.

Interpretation :

p-value for secondary infection is < 0.05

and the Hazard ratio is 2.13 indicates that strong relationship between secondary infection and increased risk of death .

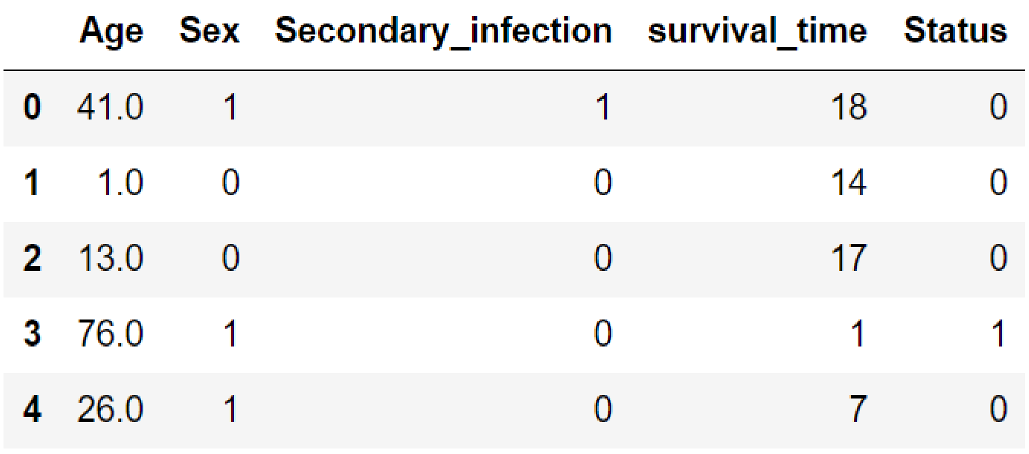
- Survival probability for first 5 patients



Here, survival probability of fourth Patient is very low as compared to other patients .

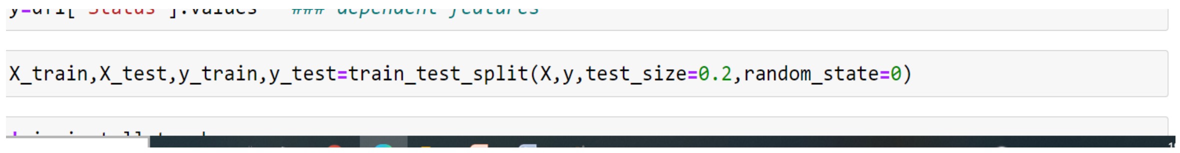
➢ Feed forward Neural Network for Survival Data :

- We Modify data to use in Neural Network



	Age	Sex	Secondary_infection	survival_time	Status
0	41.0	1	1	18	0
1	1.0	0	0	14	0
2	13.0	0	0	17	0
3	76.0	1	0	1	1
4	26.0	1	0	7	0

Splitting data in Train set and Test set

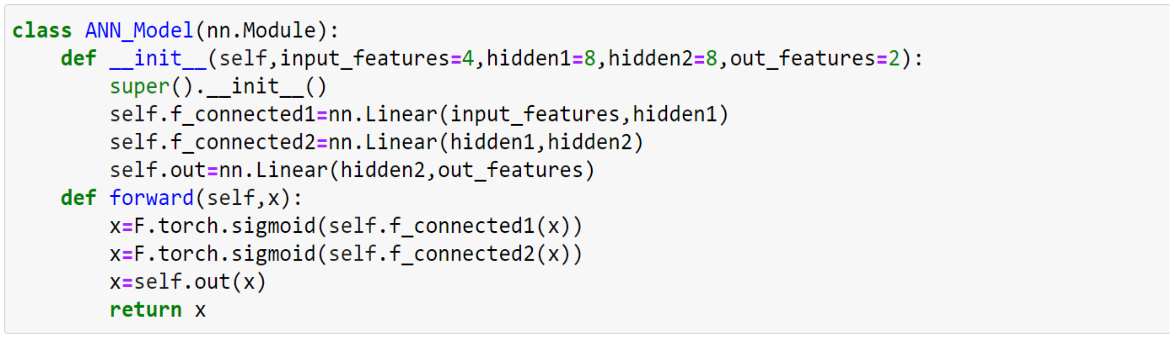


```
y=df['Status'].values   ### dependent features
```

```
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=0)
```

Install torch library to build neural network.

- Build up Neural Network :



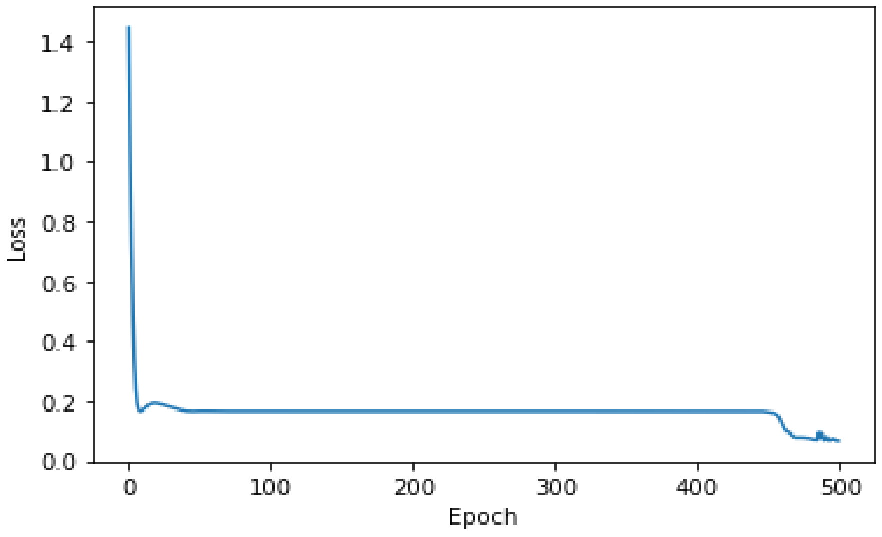
```
class ANN_Model(nn.Module):
    def __init__(self,input_features=4,hidden1=8,hidden2=8,out_features=2):
        super().__init__()
        self.f_connected1=nn.Linear(input_features,hidden1)
        self.f_connected2=nn.Linear(hidden1,hidden2)
        self.out=nn.Linear(hidden2,out_features)
    def forward(self,x):
        x=F.torch.sigmoid(self.f_connected1(x))
        x=F.torch.sigmoid(self.f_connected2(x))
        x=self.out(x)
        return x
```

We construct neural network with Input layer having 4 nodes , two hidden layers each having 8 nodes and output layer having two nodes .

Using Sigmoid Activation function

- Back –Propagation
  - We use Loss function as Cross entropy and loss optimizer function as Adam(Adaptive moment estimation)
  - We choose Epoch = 500
  - (What is an Epoch? In terms of artificial neural networks, an epoch refers to one cycle through the full training dataset.)

➢ Loss Function Plot

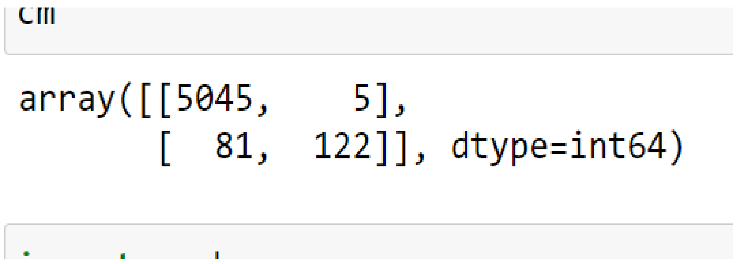


As Epoch increases loss of model decreases.

Epoch : An epoch describes the number of times a network sees an entire dataset.

➢ Prediction on Test data

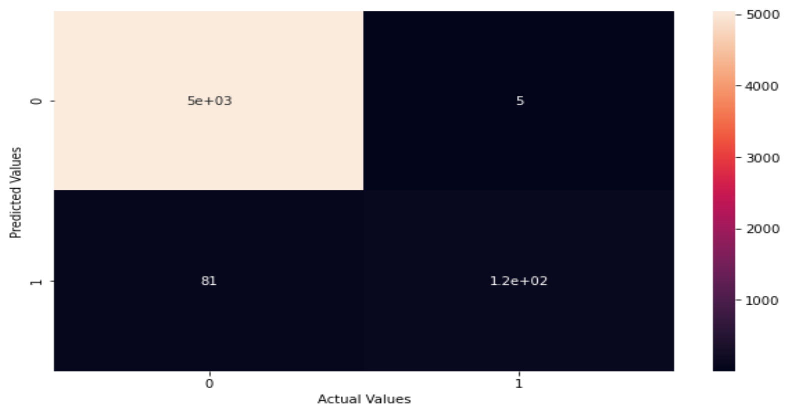
Confusion Matrix :



```
cm
```

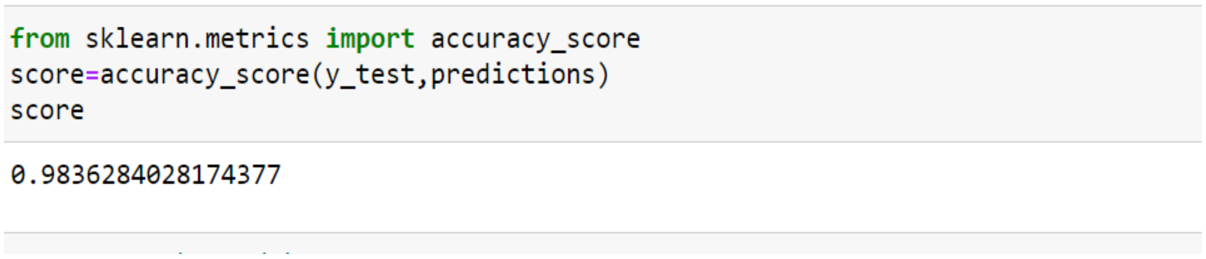
```
array([[5045,    5],
       [  81,  122]], dtype=int64)
```

➢ Heat-map of Confusion matrix



Model predicts 5167 data points correctly where as 86 data points incorrectly .

- Model accuracy



```
from sklearn.metrics import accuracy_score
score=accuracy_score(y_test,predictions)
score
```

```
0.9836284028174377
```

Fitted Model gives 98.36 % accuracy on test data .

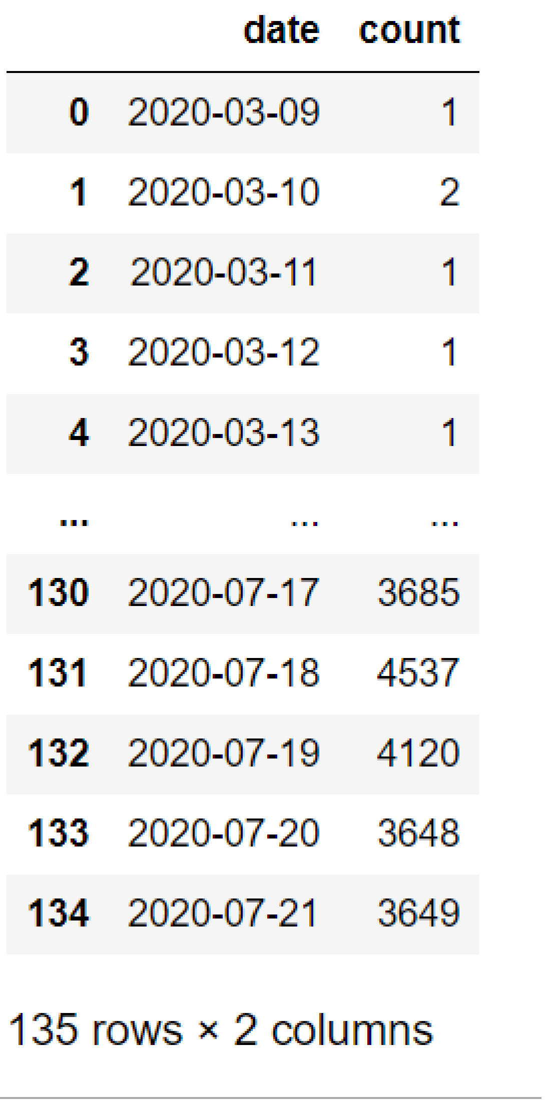
- Conclusion :

– Neural Network Model gives maximum accuracy for Survival data by predicting Status of patients (i.e., dead or alive) .

– Model predicts that Secondary infection affects highly on status of patient rather than Age and Gender .

– As cases increases or days passed time in hospital of patients during treatment decreases.

- **LSTM for forecasting number of patients for next 10 days**
  - Preparing data for further process :



	date	count
0	2020-03-09	1
1	2020-03-10	2
2	2020-03-11	1
3	2020-03-12	1
4	2020-03-13	1
...	...	...
130	2020-07-17	3685
131	2020-07-18	4537
132	2020-07-19	4120
133	2020-07-20	3648
134	2020-07-21	3649

135 rows × 2 columns

- Data set contains 135 dates in sequence
- First 125 dates are used to train model.
- Installed tensorflow and keras libraries to build neural network.
- Import Sequential,LSTM,Dense,Flatten.

```python
import numpy as np
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import Flatten
```

- Preparing function for splitting data

```python
def prepare_data(timeseries_data, n_features):
    X, y =[],[]
    for i in range(len(timeseries_data)):
        # find the end of this pattern
        end_ix = i + n_features
        # check if we are beyond the sequence
        if end_ix > len(timeseries_data)-1:
            break
            # gather input and output parts of the pattern
        seq_x, seq_y = timeseries_data[i:end_ix], timeseries_data[end_ix]
        X.append(seq_x)
        y.append(seq_y)
    return np.array(X), np.array(y)
```

- Splitting data into 115 rows , each row having 10 steps(data points).

```python
timeseries_data = train_data
timeseries_data
n_steps = 10
X, y = prepare_data(timeseries_data, n_steps)
```

```
[[   1    2    1 ...    0    4    3]
 [   2    1    1 ...    4    3    1]
 [   1    1    1 ...    3    1    0]
 ...
 [1105  947 1272 ... 1843 1498 2062]
 [ 947 1272 1502 ... 1498 2062 2228]
 [1272 1502 1694 ... 2062 2228 2313]]
[   1    0    5    5    8    8   10    4    9   12    7    5   13    9
   14    4   16    7   12   12    6   16   10    8   17   15   13   19
   36   44   25    6   18   10    9   18   29   26    3    9   11   11
   31   24   12   13   37   22   20   12   48   41   54   14   63   34
   28   69   36   55   99  149   67  143  138  216  130   93  101  135
  115  248  141  299  187  388  267  257  515  378  239  308  161  120
  204  271  308  176  213  317  204  210  337  416  453  249  322  397
  442  445  918 1267 1105  947 1272 1502 1694 1839 1925 1843 1498 2062
 2228 2313 2798]
```
.

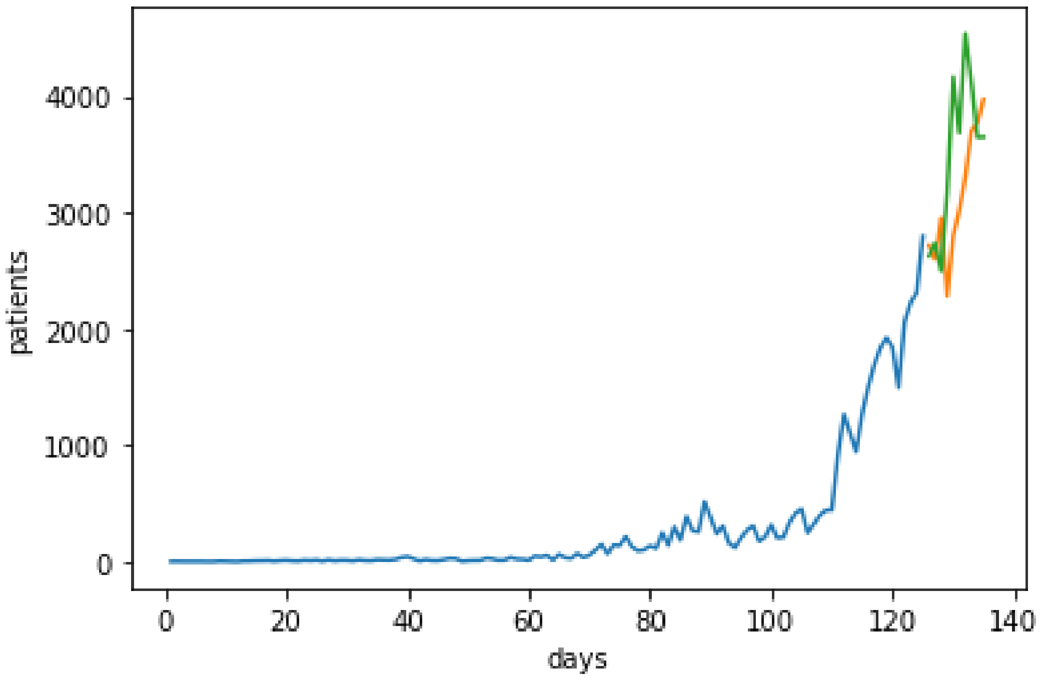
- Building LSTM Model

```
# define model
model = Sequential()
model.add(LSTM(50, activation='relu', return_sequences=True, input_shape=(n_steps, n_features)))
model.add(LSTM(50, activation='relu'))
model.add(Dense(1))
model.compile(optimizer='adam', loss='mse')
# fit model
history = model.fit(X, y, epochs=1000, verbose=1)
pyplot.plot(history.history['loss'])
#pyplot.plot(history.history['val_loss'])
pyplot.xlabel('epoch')
pyplot.ylabel('loss')
```

- Demonstrate prediction for next 10 days.

```
x_input = np.array(last10)
temp_input=list(x_input)
pred_data=[]
i=0
while(i<10):

    if(len(temp_input)>10):
        x_input=np.array(temp_input[1:])
        print("{} day input {}".format(i,x_input))
        #print(x_input)
        x_input = x_input.reshape((1, n_steps, n_features))
        #print(x_input)
        yhat = model.predict(x_input, verbose=0)
        print("{} day output {}".format(i,yhat))
        temp_input.append(yhat[0][0])
        temp_input=temp_input[1:]
        #print(temp_input)
        pred_data.append(yhat[0][0])
        i=i+1
    else:
        x_input = x_input.reshape((1, n_steps, n_features))
        yhat = model.predict(x_input, verbose=0)
        print(yhat[0])
        temp_input.append(yhat[0][0])
        pred_data.append(yhat[0][0])
        i=i+1


print(pred_data)
```

- Visualizing Output .



As days passes number of patients increases .

Green line indicates forecasted number of patients ,where as Orange line indicates actual numbers.

- The Accuracy of Model is calculated by "Mean Absolute Percentage Error"(**MAPE**).

$$M = 1/n \, \Sigma \, |(At - Ft)/At| \; * \; 100$$

M=mean absolute percentage error

N =number of times the summation iteration happens

At = actual value

Ft =forecast value

– The MAPE for the model is **23.19** . Model contains 23.19 % error in forecasted values.

➢ Conclusion :

- From Kaplan–Meier estimator we conclude that probability of female patient surviving in covid pandemic is higher than male patients .
- From cox-proportional model there is strong relationship between secondary infection and increased risk of death .
- Feed-forward Neural Network gives 98.36% Accuracy for survival data by predicting status of patients (i.e, dead or alive) .
- LSTM model gives forecasted number of patients for next 10 days with 23.19% error .

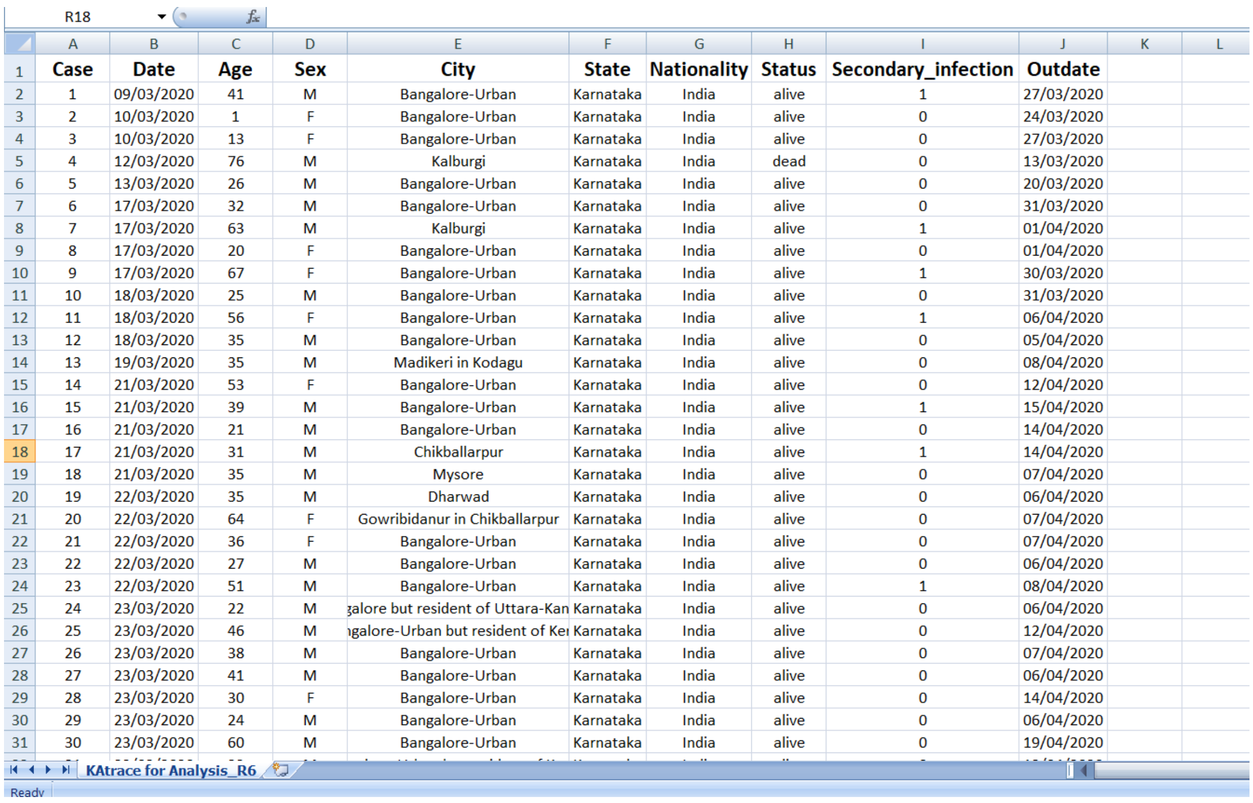
➢ Appendix :

The data of covid -19 patients of Karnataka state is collected from the website

https://www.isibang.ac.in/~athreya/incovid19/data.htmlwebsite

Which is official website of Indian Statistical Institute , Bangalore centre .

Data description :



Case	Date	Age	Sex	City	State	Nationality	Status	Secondary_infection	Outdate
1	09/03/2020	41	M	Bangalore-Urban	Karnataka	India	alive	1	27/03/2020
2	10/03/2020	1	F	Bangalore-Urban	Karnataka	India	alive	0	24/03/2020
3	10/03/2020	13	F	Bangalore-Urban	Karnataka	India	alive	0	27/03/2020
4	12/03/2020	76	M	Kalburgi	Karnataka	India	dead	0	13/03/2020
5	13/03/2020	26	M	Bangalore-Urban	Karnataka	India	alive	0	20/03/2020
6	17/03/2020	32	M	Bangalore-Urban	Karnataka	India	alive	0	31/03/2020
7	17/03/2020	63	M	Kalburgi	Karnataka	India	alive	1	01/04/2020
8	17/03/2020	20	F	Bangalore-Urban	Karnataka	India	alive	0	01/04/2020
9	17/03/2020	67	F	Bangalore-Urban	Karnataka	India	alive	1	30/03/2020
10	18/03/2020	25	M	Bangalore-Urban	Karnataka	India	alive	0	31/03/2020
11	18/03/2020	56	F	Bangalore-Urban	Karnataka	India	alive	1	06/04/2020
12	18/03/2020	35	M	Bangalore-Urban	Karnataka	India	alive	0	05/04/2020
13	19/03/2020	35	M	Madikeri in Kodagu	Karnataka	India	alive	0	08/04/2020
14	21/03/2020	53	F	Bangalore-Urban	Karnataka	India	alive	0	12/04/2020
15	21/03/2020	39	M	Bangalore-Urban	Karnataka	India	alive	1	15/04/2020
16	21/03/2020	21	M	Bangalore-Urban	Karnataka	India	alive	0	14/04/2020
17	21/03/2020	31	M	Chikballarpur	Karnataka	India	alive	1	14/04/2020
18	21/03/2020	35	M	Mysore	Karnataka	India	alive	0	07/04/2020
19	22/03/2020	35	M	Dharwad	Karnataka	India	alive	0	06/04/2020
20	22/03/2020	64	F	Gowribidanur in Chikballarpur	Karnataka	India	alive	0	07/04/2020
21	22/03/2020	36	F	Bangalore-Urban	Karnataka	India	alive	0	07/04/2020
22	22/03/2020	27	M	Bangalore-Urban	Karnataka	India	alive	0	06/04/2020
23	22/03/2020	51	M	Bangalore-Urban	Karnataka	India	alive	1	08/04/2020
24	23/03/2020	22	M	galore but resident of Uttara-Kan	Karnataka	India	alive	0	06/04/2020
25	23/03/2020	46	M	ngalore-Urban but resident of Ke	Karnataka	India	alive	0	12/04/2020
26	23/03/2020	38	M	Bangalore-Urban	Karnataka	India	alive	0	07/04/2020
27	23/03/2020	41	M	Bangalore-Urban	Karnataka	India	alive	0	06/04/2020
28	23/03/2020	30	F	Bangalore-Urban	Karnataka	India	alive	0	14/04/2020
29	23/03/2020	24	M	Bangalore-Urban	Karnataka	India	alive	0	06/04/2020
30	23/03/2020	60	M	Bangalore-Urban	Karnataka	India	alive	0	19/04/2020

KAtrace for Analysis_R6

For the entire data repository,

1) Go to the website and register on Registration page.
2) After completion of registration page , they send an email with username and password to login on their page.
3) Then you can download the dataset from the website.

In this dataset one row corresponds to one case of Coronavirus in Karnataka. For each patient the columns contain the following information.

A : case number : as recognized by the Karnataka government .

B : Date : when they tested positive.

C : Age : Age of patient (in years).

D : Sex : Sex of patient(M and F)

E : city : Contains both the city where a patient is held and the city of their residence.

F : State : Karnataka for all states.

G : Nationality : Indian for all cases.

I : Secondary infection : Indicates that the patient has any [known] secondary infection .The entries of this are either 0 or 1 .

0 for No-secondary infection

1 for secondary infection

J : outdate : This columns indicate that discharge date of patient from hospital.

➢ References :

- John P.Klein , Hans C. van Houwelingen , Joseph G. Ibrahim , Thomas H. Scheike : Handbook of Survival Analysis .
- David W. Hosmer Stanley Lemeshow Susanne may : Applied Survival Analysis
- Miachel Taylour :Make Your Own Neural Network_ An In-Depth Visual Introduction for Beginners .
- Research papers :
- Early triage of critically ill covid 19 patients using deep learning.
- Brown 1997 : on the use of ANN for Analysis of Survival data .
- Feed Forward NN for the Analysis of censoring Survival Data : Partial logistic regression approach .