An Introductory Guide to Maximum Likelihood Estimation (with a case study in R)

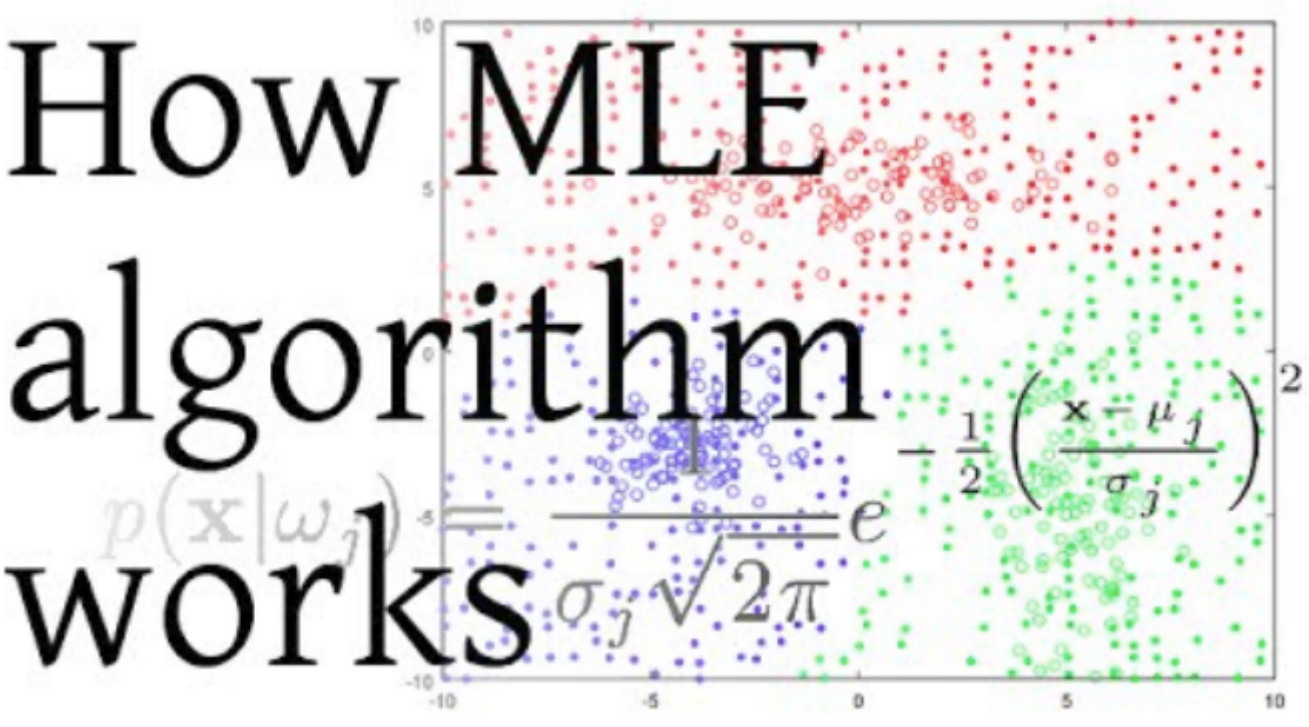
AanishS Singla — Published On July 16, 2018 and Last Modified On May 31st, 2020

Intermediate Machine Learning R Statistics Technique

Introduction

Interpreting how a model works is one of the most basic yet critical aspects of data science. You build a model which is giving you pretty impressive results, but what was the process behind it? As a data scientist, you need to have an answer to this oft-asked question.

For example, let's say you built a model to predict the stock price of a company. You observed that the stock price increased rapidly over night. There could be multiple reasons behind it. Finding the likelihood of the most probable reason is what Maximum Likelihood Estimation is all about. This concept is used in economics, MRIs, satellite imaging, among other things.



Source: YouTube

In this post we will look into how Maximum Likelihood Estimation (referred as MLE hereafter) works and how it can be used to determine coefficients of a model with any kind of distribution. Understanding MLE would involve probability and mathematics, but I will try to make it easier with examples.

*Note: As mentioned, this article assumes that you know the basics of maths and probability. You can refresh your concepts by going through this article first – **6 Common Probability Distributions every data science professional should know**.*

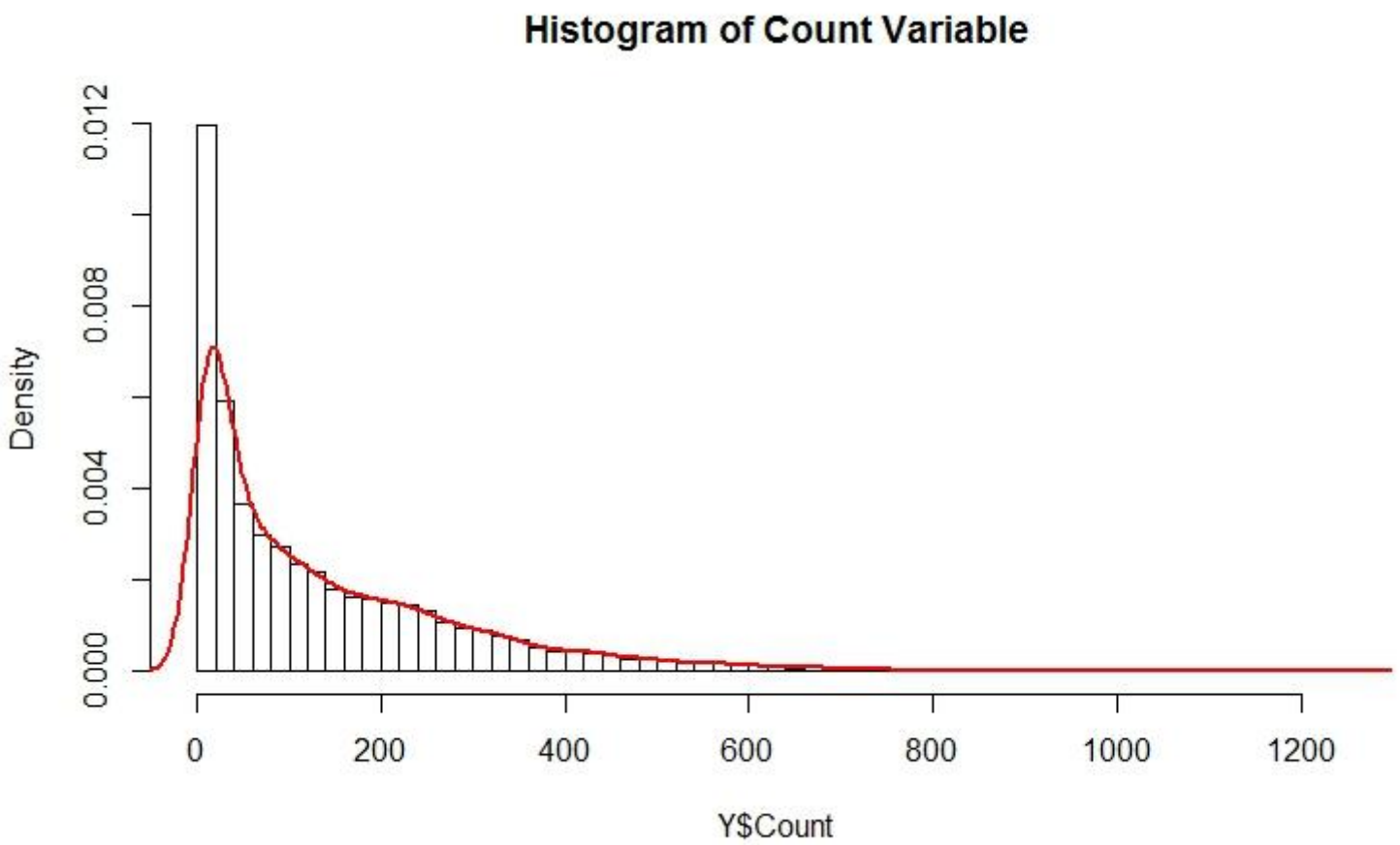
Table of Contents

- Why should I use Maximum Likelihood Estimation (MLE)?

- Maximizing the Likelihood
- Determining model coefficients using MLE
- MLE in R

Why should I use Maximum Likelihood Estimation (MLE)?

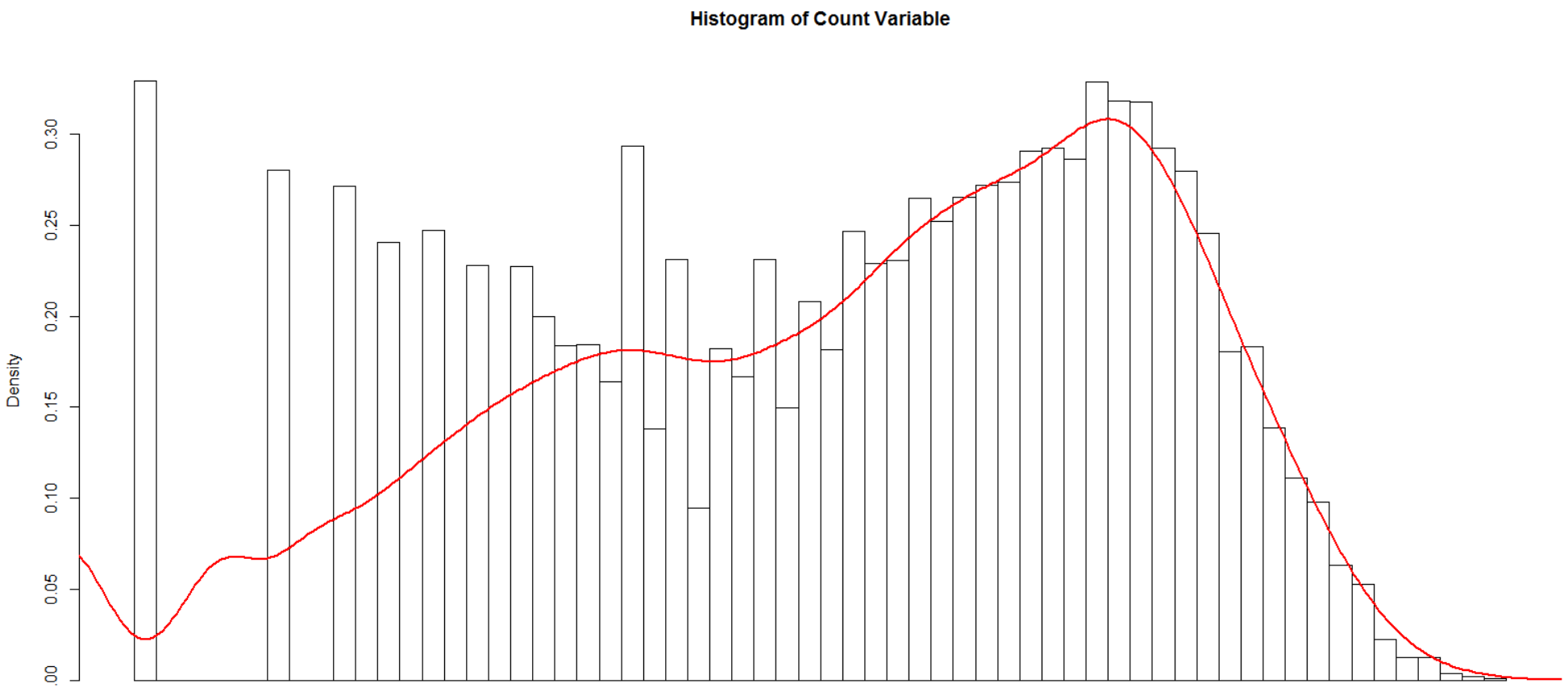
Let us say we want to predict the sale of tickets for an event. The data has the following histogram and density.



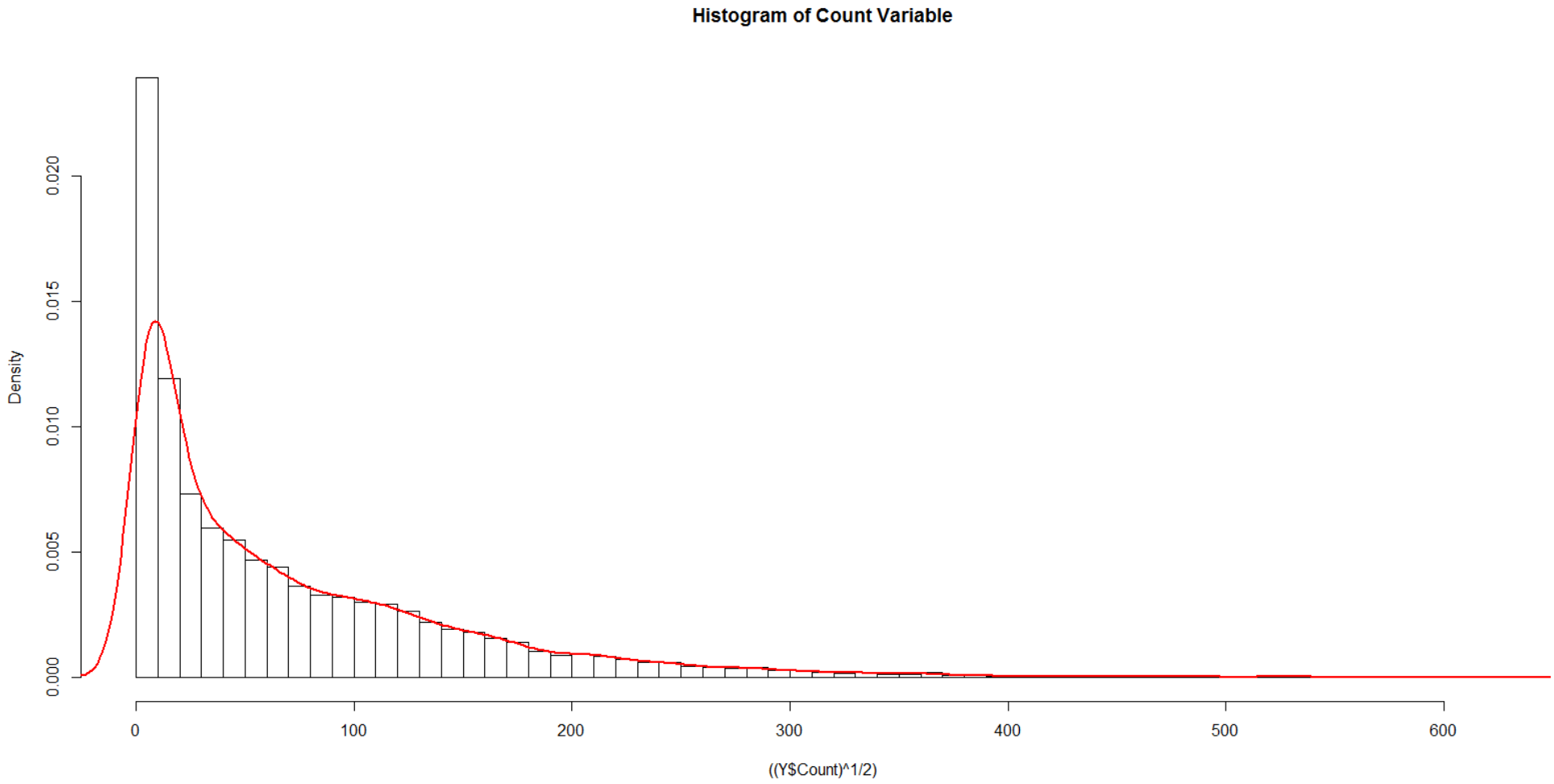
How would you model such a variable? The variable is not normally distributed and is asymmetric and hence it violates the assumptions of linear regression. A popular way is to transform the variable with log, sqrt, reciprocal, etc. so that the transformed variable is normally distributed and can be modelled with linear regression.

Let's try these transformations and see how the results are:

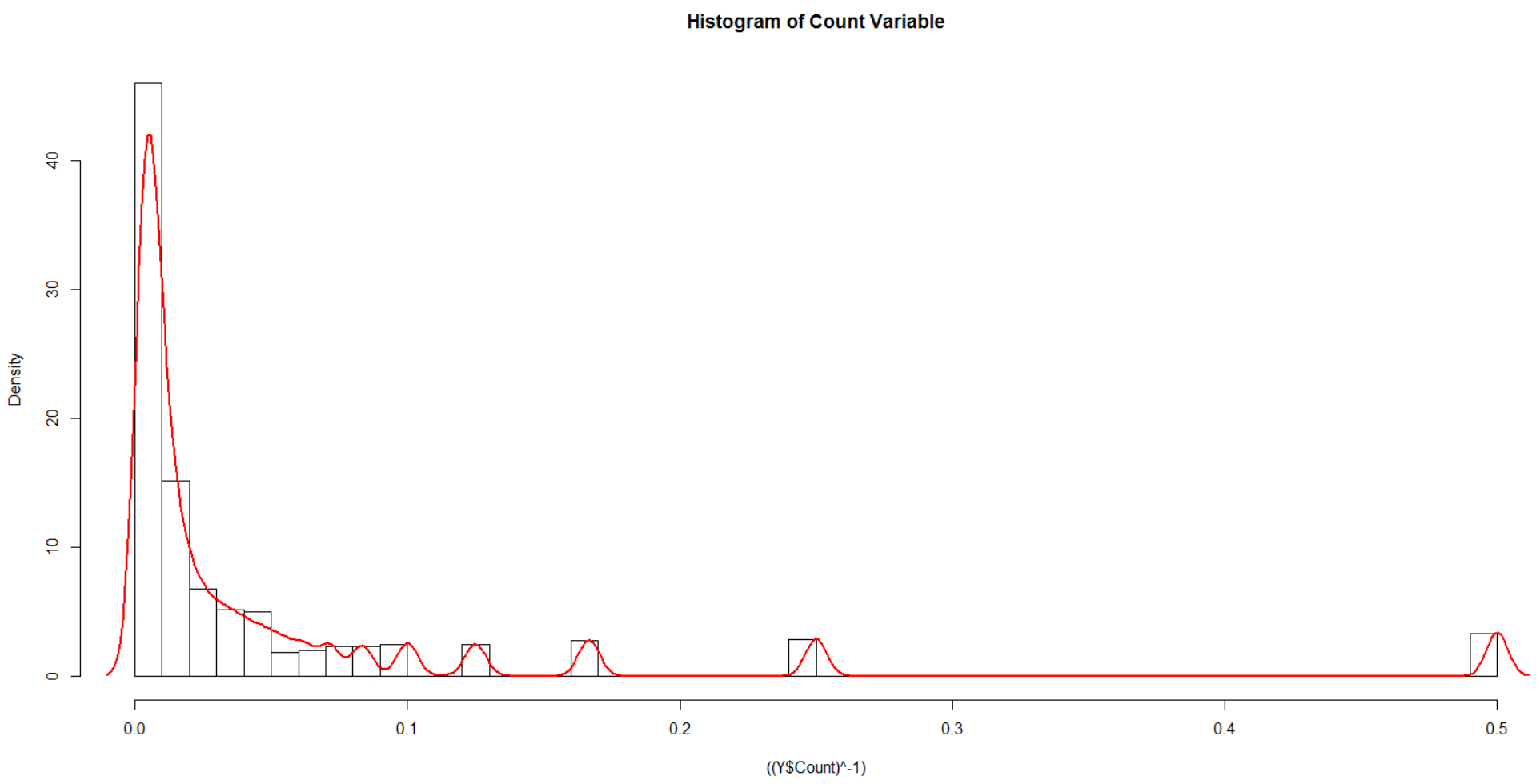
With Log transformation:



An Introductory Guide to Maximum Likelihood Estimation (with a case study in R)



With Reciprocal:



None of these are close to a normal distribution. How should we model such data so that the basic assumptions of the model are not violated? How about modelling this data with a different distribution rather than a normal one? If we do use a different distribution, how will we estimate the coefficients?

This is where **Maximum Likelihood Estimation (MLE)** has such a major advantage.

Understanding MLE with an example

While studying stats and probability, you must have come across problems like – What is the probability of x > 100, given that x

Let's understand this with an example: Suppose we have data points representing the weight (in kgs) of students in a class. The data points are shown in the figure below (the R code that was used to generate the image is provided as well):

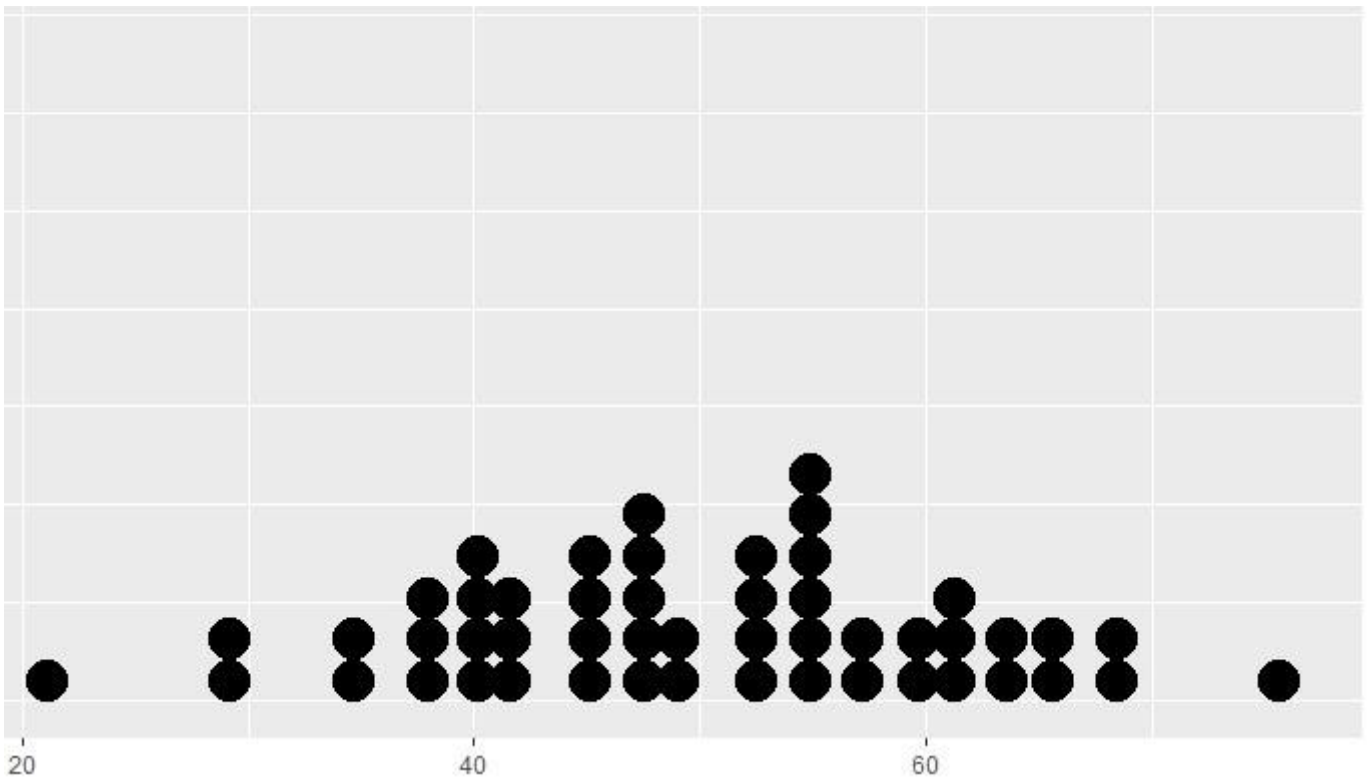


Figure 1

```
x = as.data.frame(rnorm(50,50,10))
ggplot(x, aes(x = x)) + geom_dotplot()
```

This appears to follow a normal distribution. But how do we get the mean and standard deviation (sd) for this distribution? One way is to directly compute the mean and sd of the given data, which comes out to be 49.8 Kg and 11.37 respectively. These values are a good representation of the given data but may not best describe the population.

We can use MLE in order to get more robust parameter estimates. *Thus,* ***MLE can be defined as a method for estimating population parameters (such as the mean and variance for Normal, rate (lambda) for Poisson, etc.) from sample data such that the probability (likelihood) of obtaining the observed data is maximized.***

In order to get an intuition of MLE, try to guess which of the following would maximize the probability of observing the data in the above figure?

1. Mean = 100, SD = 10
2. Mean = 50, SD = 10

Clearly, it is not very likely we'll observe the above data shape if the population mean is 100.

Getting to know the technical details

Now that you got an intuition of what MLE can do, we can get into the details of what actually likelihood is and how it can be maximized. But first, let's start with a quick review of distribution parameters.

Distribution Parameters

Let us first understand distribution parameters. Wikipedia's definition of this term is as follows: "*It is a quantity that indexes a family of probability distributions".* It can be regarded as a numerical characteristic of a population or a statistical model. We can understand it by the following diagram:

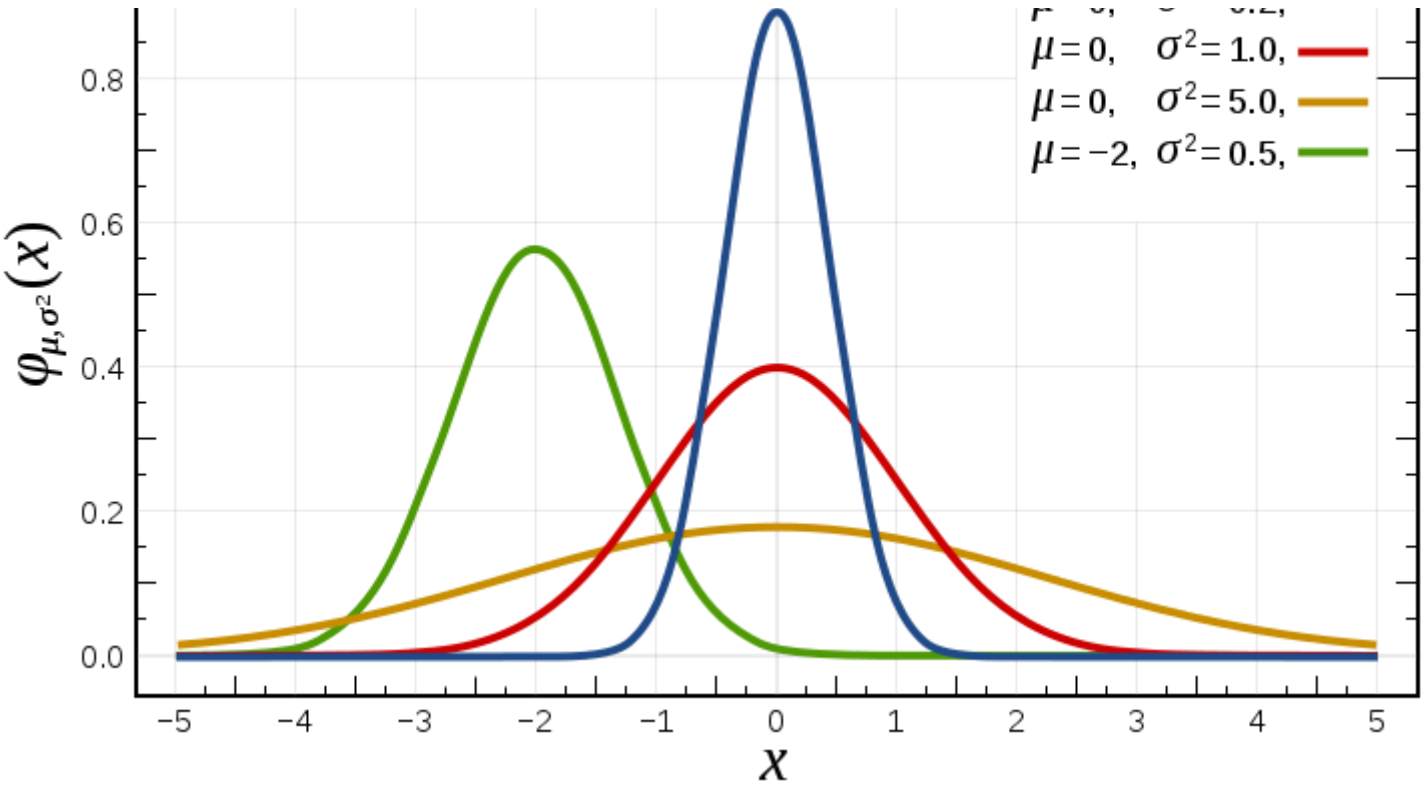


Figure 2, Source: Wikipedia

The width and height of the bell curve is governed by two parameters – mean and variance. These are known as distribution parameters for normal distribution. Similarly, Poisson distribution is governed by one parameter – lambda, which is the number of times an event occurs in an interval of time or space.

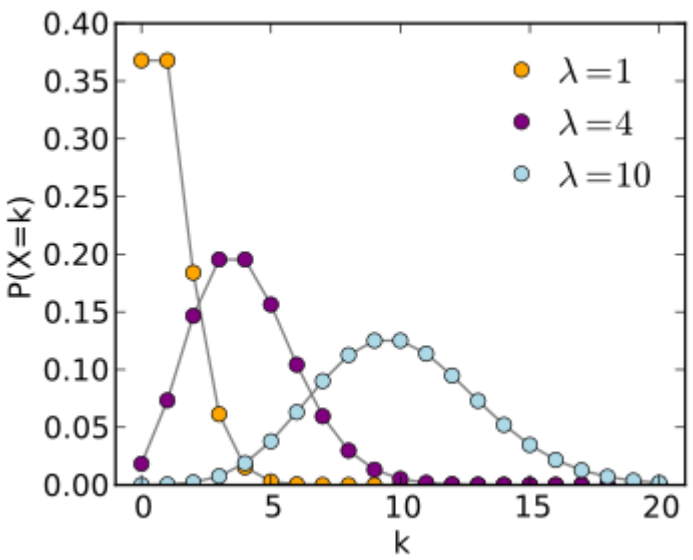


Figure 3, Source: Wikipedia

Most of the distributions have one or two parameters, but some distributions can have up to 4 parameters, like a 4 parameter beta distribution.

Likelihood

From Fig. 2 and 3 we can see that given a set of distribution parameters, some data values are more probable than other data. From Fig. 1, we have seen that the given data is more likely to occur when the mean is 50, rather than 100. In reality however, we have already observed the data. Accordingly, we are faced with an inverse problem: *Given the observed data and a model of interest, we need to find the one Probability Density Function/Probability Mass Function (f(x|θ)), among all the probability densities that are most likely to have produced the data*.

To solve this inverse problem, we define the likelihood function by reversing the roles of the data vector x and the (distribution) parameter vector θ in f(x| θ), i.e.,

Log Likelihood

The mathematical problem at hand becomes simpler if we assume that the observations (xi) are independent and identically distributed random variables drawn from a Probability Distribution, *f0 (where f0 = Normal Distribution for example in Fig.1)*. This reduces the Likelihood function to:

$$L(\theta; x) = f_0(x_1, x_2, x_3, \ldots x_n \mid \theta) = f_0(x_1 \mid \theta) \cdot f_0(x_2 \mid \theta) \cdot f0(x3 \mid \theta) \ldots . f_0(x_n \mid \theta)$$

To find the maxima/minima of this function, we can take the derivative of this function w.r.t θ and equate it to 0 (as zero slope indicates maxima or minima). Since we have terms in product here, we need to apply the chain rule which is quite cumbersome with products. **A clever trick would be to take log of the likelihood function and maximize the same.** This will convert the product to sum and since log is a strictly increasing function, it would not impact the resulting value of θ. So we have:

$$LL(\theta; x) = \log[f_0(x_1 \mid \theta) \cdot f_0(x_2 \mid \theta) \cdot f0(x3 \mid \theta) \ldots . f_0(x_n \mid \theta)]$$

$$= \log(f_0(x_1 \mid \theta) + \log(f_0(x_2 \mid \theta) + \ldots . + \log(f_0(x_n \mid \theta)$$

Maximizing the Likelihood

To find the maxima of the log likelihood function LL(θ; x), we can:

- Take first derivative of LL(θ; x) function w.r.t θ and equate it to 0
- Take second derivative of LL(θ; x) function w.r.t θ and confirm that it is negative

There are many situations where calculus is of no direct help in maximizing a likelihood, but a maximum can still be readily identified. There's nothing that gives setting the first derivative equal to zero any kind of 'primacy' or special place in finding the parameter value(s) that maximize log-likelihood. It's simply a convenient tool when a few parameters need to be estimated.

As a general principle, pretty much any valid approach for identifying the argmax of a function may be suitable to find maxima of the log likelihood function. This is an unconstrained non-linear optimization problem. We seek an optimization algorithm that behaves in the following manner:

1. Reliably converge to a local minimizer from an arbitrary starting point
2. Do it as quickly as possible

It's very common to use optimization techniques to maximize likelihood; there are a large variety of methods (Newton's method, Fisher scoring, various conjugate gradient-based approaches, steepest descent, Nelder-Mead type (simplex) approaches, BFGS and a wide variety of other techniques).

It turns out that when the model is assumed to be Gaussian as in the examples above, the MLE estimates are equivalent to the ordinary least squares method.

You can refer to the proof **here**.

Determining model coefficients using MLE

vector of explanatory variables xi . We could form a simple linear model as follows –

$$\mu_i = x_i' \theta ,$$

where θ is the vector of model coefficients. This model has the disadvantage that the linear predictor on the right-hand side can assume any real value, whereas the Poisson mean on the left-hand side, which represents an expected count, has to be non-negative. A straightforward solution to this problem is to model the logarithm of the mean using a linear model. Thus, we consider a generalized linear model with log link log, which can be written as follows –

$$\log(\mu_i) = x_i' \theta$$

or

$$\mu_i = \exp(x_i' \theta)$$

Our aim is to find θ by using MLE.

Now, Poisson distribution is given by:

$$Pr\{Y = y|\mu\} = (e^{-\mu} \mu^y)/ y!$$

We can apply the log likelihood concept that we learnt in the previous section to find the θ. Taking logs of the above equation and ignoring a constant involving log(y!), we find that the log-likelihood function is –

$$LL(\theta) = \sum \{y_i \log(\mu_i) - \mu_i\}, \qquad \text{-------- Eq. 1}$$

where μi depends on the covariates xi and a vector of θ coefficients. We can substitute μi = exp(xi'θ) and solve the equation to get θ that maximizes the likelihood. Once we have the θ vector, we can then predict the expected value of the mean by multiplying the xi and θ vector.

MLE using R

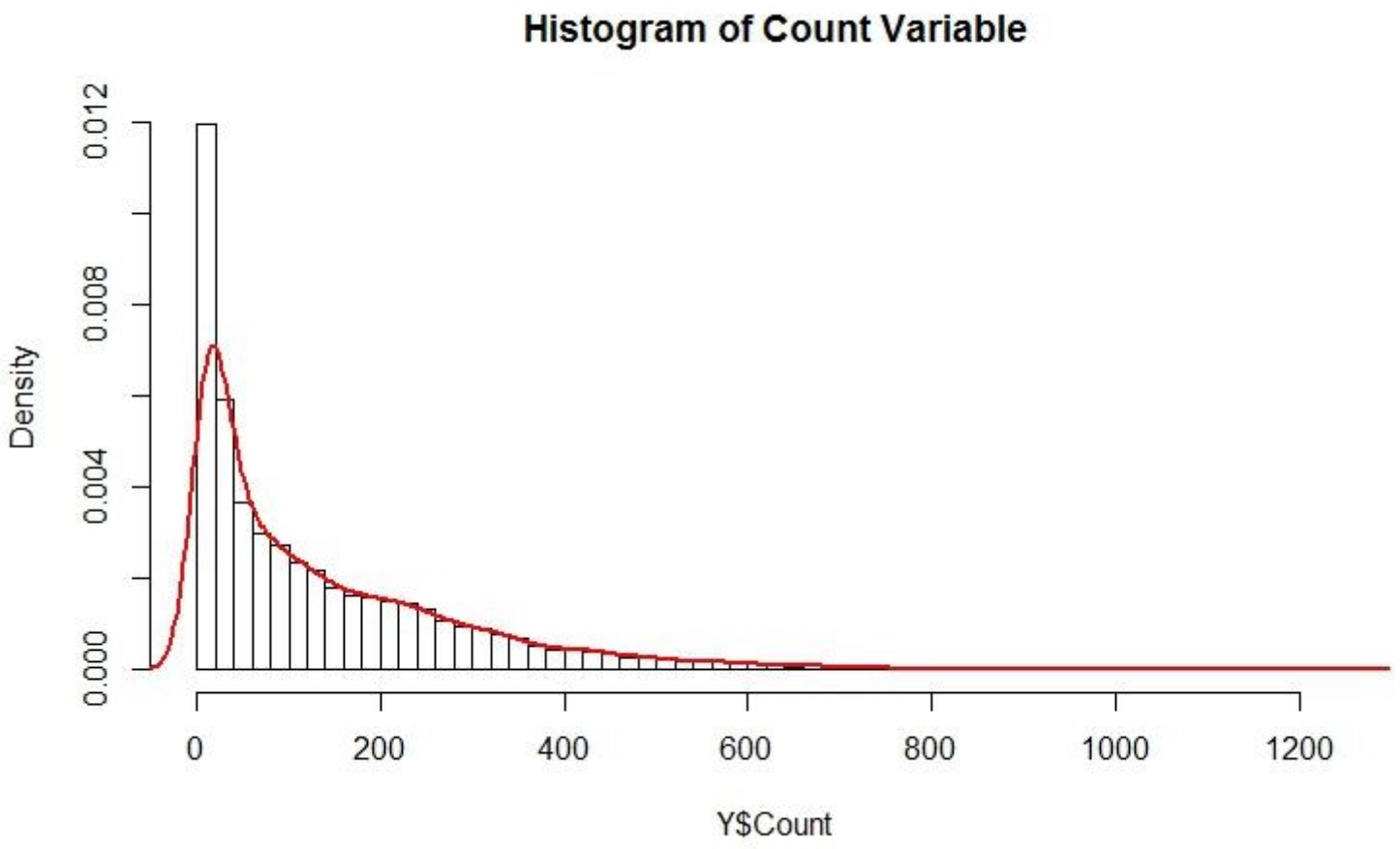
In this section, we will use a real-life dataset to solve a problem using the concepts learnt earlier. ***You can download the dataset from this link.*** A sample from the dataset is as follows:

Datetime	Count of tickets sold
25-08-2012 00:00	8
25-08-2012 01:00	2
25-08-2012 02:00	6
25-08-2012 03:00	2
25-08-2012 04:00	2
25-08-2012 05:00	2

It has the count of tickets sold in each hour from 25th Aug 2012 to 25th Sep 2014 (about 18K records). Our aim is to predict the number of tickets sold in each hour. This is the same dataset which was discussed in the first section of this article.

The problem can be solved using techniques like regression, time series, etc. Here we will use the statistical modeling technique

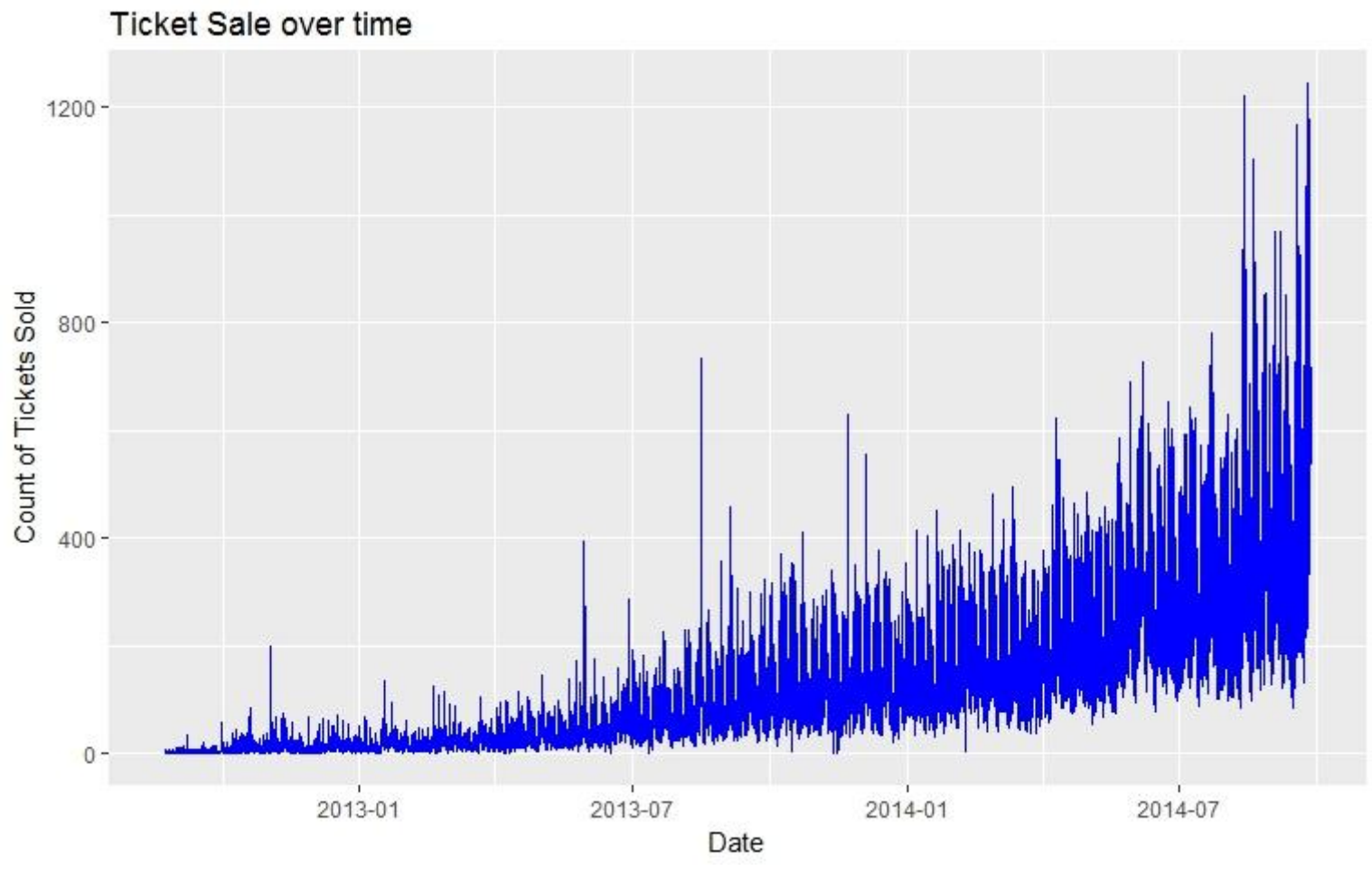
```
hist(Y$Count, breaks = 50,probability = T ,main = "Histogram of Count Variable")
lines(density(Y$Count), col="red", lwd=2)
```



This could be treated as a Poisson distribution or we could even try fitting an exponential distribution.

Since the variable at hand is count of tickets, Poisson is a more suitable model for this. Exponential distribution is generally used to model time interval between events.

Let's plot the count of tickets sold over these 2 years:



Looks like there is a significant increase in sale of tickets over time. In order to keep things simple, let's model the outcome by only using age as a factor, where age is the defined no. of weeks elapsed since 25th Aug 2012. We can write this as:

Combining Eq. 1 and 2, we get the log likelihood function as follows:

$$LL(\theta) = \sum \{y_i.(\theta_0 + age * \theta_1) - \exp(\theta_0 + age * \theta_1)\}$$

We can use the *mle()* function in R *stats4* package to estimate the coefficients θ0 and θ1. It needs the following primary parameters:

1. **Negative Likelihood function which needs to be minimized:** This is same as the one that we have just derived but a negative sign in front [as maximizing the log likelihood is same as minimizing the negative log likelihood]
2. **Starting point for the coefficient vector:** This is the initial guess for the coefficient. Results can vary based on these values as the function can hit local minima. Hence, it's good to verify the results by running the function with different starting points
3. Optionally, the method using which the likelihood function should be optimized. BFGS is the default method

For our example, the negative log likelihood function can be coded as follows:

```
nll <- function(theta0,theta1) {
    x <- Y$age[-idx]
    y <- Y$Count[-idx]
    mu = exp(theta0 + x*theta1)
    -sum(y*(log(mu)) - mu)
}
```

I have divided the data into train and test set so that we can objectively evaluate the performance of the model. *idx* is the indices of the rows which are in test set.

```
set.seed(200)
idx <- createDataPartition(Y$Count, p=0.25,list=FALSE)
```

Next let's call the *mle* function to get the parameters:

```
est <- stats4::mle(minuslog=nll, start=list(theta0=2,theta1=0))
summary(est)

Maximum likelihood estimation
Call:
stats4::mle(minuslogl = nll, start = list(theta0 = 2, theta1 = 0))

Coefficients:
         Estimate  Std. Error
theta0 2.68280754 2.548367e-03
theta1 0.03264451 2.998218e-05

-2 log L: -16594396
```

This gives us the estimate of the coefficients. Let's use RMSE as the evaluation metric for getting results on the test set:

```
pred.ts <- (exp(coef(est)['theta0'] + Y$age[idx]*coef(est)['theta1'] ))
rmse(pred.ts, Y$Count[idx])

86.95227
```

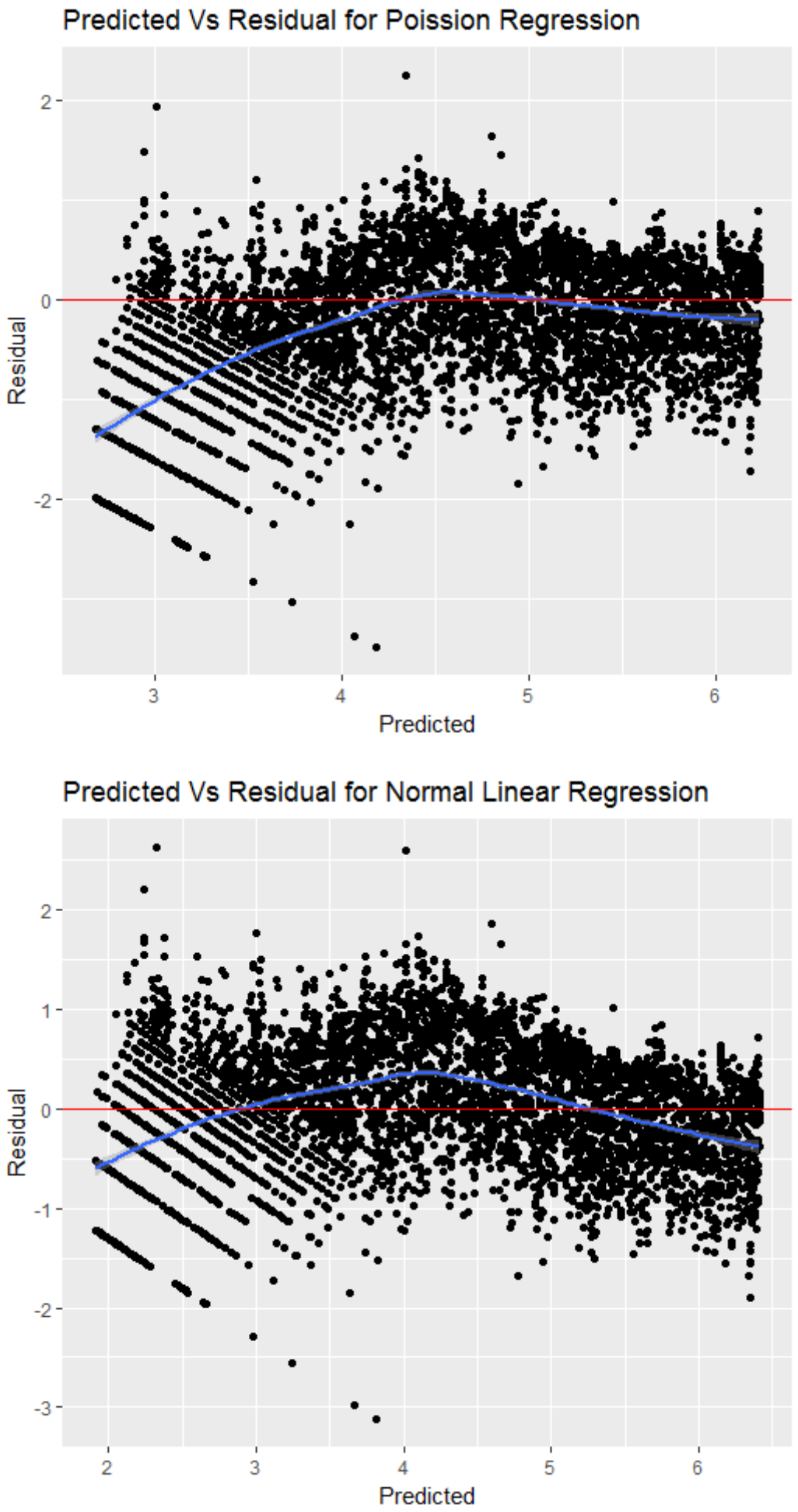
Now let's see how our model fairs against the standard linear model (with errors normally distributed), modelled with log of

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 1.9112992 0.0110972   172.2 <2e-16 ***
age         0.0414107 0.0001768   234.3 <2e-16 ***
```

```
pred.lm <- predict(lm.fit, Y[idx,])
rmse(exp(pred.lm), Y$Count[idx])

93.77393
```

As you can see, RMSE for the standard linear model is higher than our model with Poisson distribution. Let's compare the residual plots for these 2 models on a held out sample to see how the models perform in different regions:



We see that the errors using Poisson regression are much closer to zero when compared to Normal linear regression.

above, you can get the coefficients directly using the below command:

```
glm(Count ~ age, family = "poisson", data = Y)

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) 2.669     2.218e-03    1203 <2e-16 ***
age         0.03278   2.612e-05    1255 <2e-16 ***
```

Same can be done in Python using *pymc.glm()* and setting the family as *pm.glm.families.Poisson()*.

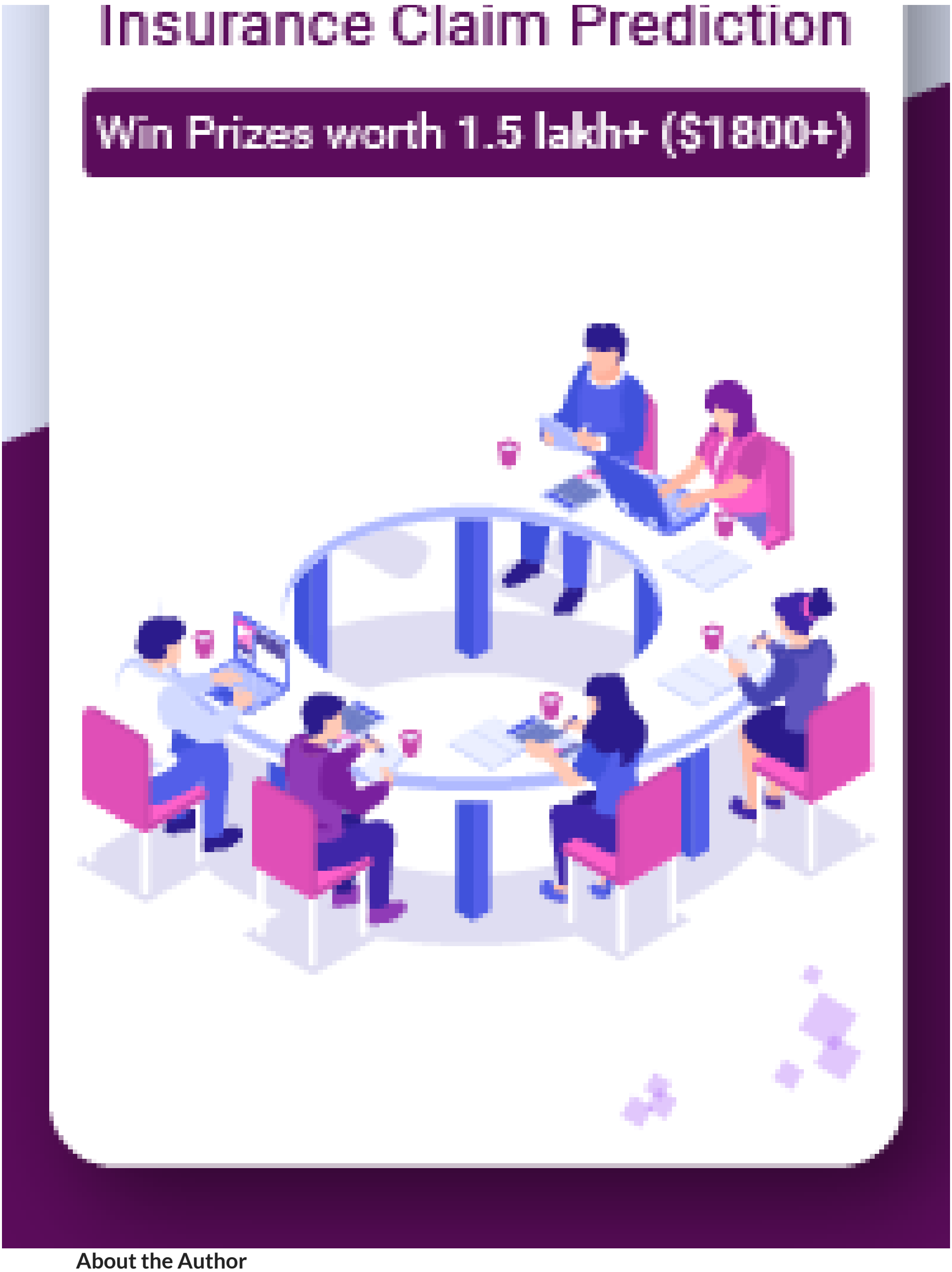
End Notes

One way to think of the above example is that there exist better coefficients in the parameter space than those estimated by a standard linear model. Normal distribution is the default and most widely used form of distribution, but we can obtain better results if the correct distribution is used instead. Maximum likelihood estimation is a technique which can be used to estimate the distribution parameters irrespective of the distribution used. So next time you have a modelling problem at hand, first look at the distribution of data and see if something other than normal makes more sense!

The detailed code and data is present on my Github repository. Refer to the “Modelling single variables.R” file for an example that covers data reading, formatting and modelling using only age variables. I have also modelled using multiple variables, which is present in the “Modelling multiple variables.R” file.

likelihood function maximum likelihood Maximum Likelihood Estimation MLE R

An Introductory Guide to Maximum Likelihood Estimation (with a case study in R)

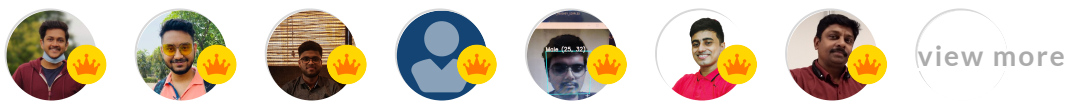


About the Author

AanishS Singla

Aanish is a Data Scientist at Nagarro and has 13+ years of experience in Machine Learning, Developing and

An Introductory Guide to Maximum Likelihood Estimation (with a case study in R)



Download

Analytics Vidhya App for the Latest blog/Article

Previous Post

Microsoft has Released a Collection of Awesome Free Datasets

Next Post

Build your own Computer Vision Model with the Latest TensorFlow Object Detection API Update

2 thoughts on "An Introductory Guide to Maximum Likelihood Estimation (with a case study in R)"

N. says:
August 15, 2018 at 3:10 pm

Thanks, very good introduction and example to mle

Reply

Arefeen Shamsuzzoha says:
October 04, 2022 at 6:10 am

what is formatfile()? I get an error that this function does not exist in r.

Reply

Leave a Reply

Your email address will not be published. Required fields are marked *

Comment

Name*

Email*

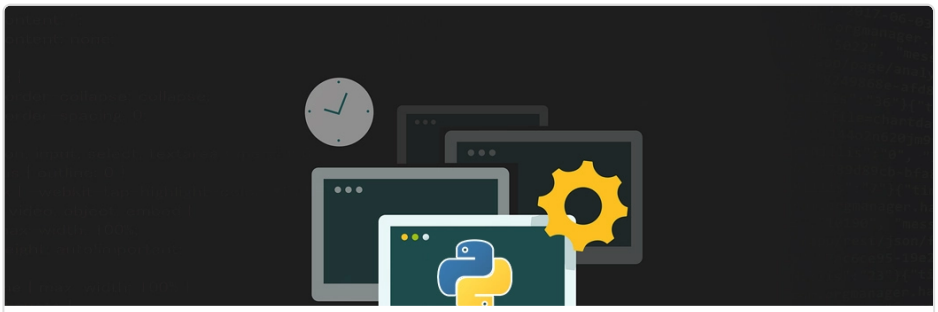
Website

Notify me of follow-up comments by email.

Notify me of new posts by email.

An Introductory Guide to Maximum Likelihood Estimation (with a case study in R)

Top Resources



Python Tutorial: Working with CSV file for Data Science

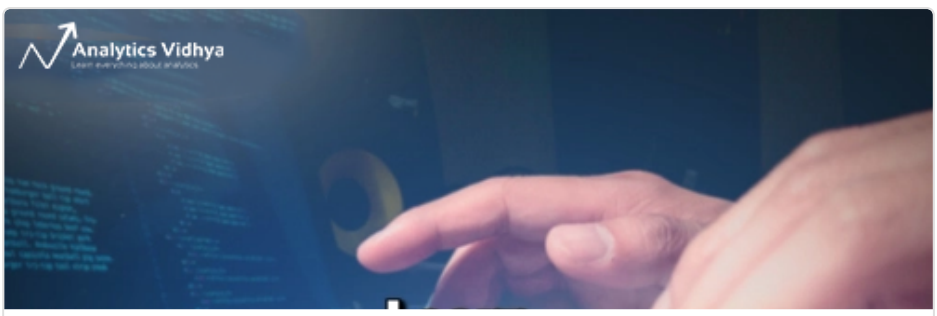
Harika Bonthu - AUG 21, 2021

The Most Comprehensive Guide to K-Means Clustering You'll Ever Need

Pulkit Sharma - AUG 19, 2019

Understanding Random Forest

Sruthi E R - JUN 17, 2021



Understanding Support Vector Machine(SVM) algorithm from examples (along with code)

Sunil Ray - SEP 13, 2017