Multivariate Procedures with R

Prof. Shalabh

Department of Mathematics and Statistics

IIT Kanpur

Week – 05

Lecture – 24

Univariate Normal Distribution: Application in R Software

Hello friends, welcome to the course Multivariate Procedure with R. So, you can recall that in the last lecture we considered the univariate normal distribution and we had discussed different types of properties and one important aspect was that how are you going to compute different types of probabilities. The rule which I had explained you in the last lecture to compute different types of probabilities in the normal distribution, they are going to be useful when we are trying to consider different types of application like as test of hypothesis and other things. So, now we have understood that how one can manipulate different types of probabilities under the normal distribution. What do I mean by manipulate? Manipulate does not mean we are going to do anything wrong. But for example, if I want to know the value of the CDF at -a, then it is going to be computed by 1 minus CDF at a.

So, these types of manipulations and different types of rule we had understood and you can derive them very easily if you try to look at the curve and they can be brought and understood very easily by understanding the rules of integration. So, now in this lecture we are going to show you the implementation of this normal distribution in the R software. So, this can mean we would like to see how we can generate the random numbers, how we can compute different types of probabilities etc. in the R software.

So, and for this thing you will not need any special package. This I am telling you because in the case of univariate you do not need any special type of package, but when we were considering the multivariate then you will need a special package. So, let us begin our lecture and try to understand it. So, you can see here that we had discussed the normal probability density function that random variable X is said to follow normal distribution with parameters μ and σ square if its probability density function is given by this function $f_X(x) = \frac{1}{\sigma\sqrt{2\pi}} exp\left\{-\frac{(x-\mu)^2}{2\sigma^2}\right\}; -\infty < x < \infty$. And we always write it in this way that $X \sim N(\mu, \sigma^2)$ where μ is the mean and σ^2 is the variance.

So, obviously, the σ is going to be the standard deviation right ok. So, now in the R software there are various possibilities. So, we can compute the density distribution function, quantile function as well as we can generate the random generation for the normal distribution. And the two parameters the mean and variance they are specified by the parameters. So, here m e a n, mean that means it is something like here you can write down here mean = μ whatever is the value of μ which you want to specify.

And you have to remember one thing that this command does not use variance, but it uses the standard deviation which is specified by the command sd(). So, you will write down here sd = what would be the value for example, here the value of σ right. So, many times you will see that in different problems you are given the variance. So, then you have to be careful that you use the square root of variance there right. So, now we will consider here four possible commands for example, the first command dnorm(), dnorm() gives the density.

So, X is here the theta vector and then you have to specify here the value of mean and standard deviation. Similarly, there is another function here pnorm(). So, pnorm() gives us the distribution function. So, you have to assign here a value q then mean standard deviation and there is one option here lower.tail = TRUE. Well, I will try to explain you all these things over here.

For example, if you see here q here is the vector of quantiles right. And similarly, if you want to use here this term here qnorm(). So, this qnorm() is going to give you here the quantile function. So, here you have to specify here the p,mean, standard deviation (sd), and lower.tail = TRUE. So, you can see here this p here is the vector of probabilities right.

And then we have here another function here rnorm(). So, rnorm() generates the random numbers from this normal distribution. So, N is the number of random numbers which you want to generate and then mean and standard deviation that you have to specify that from which distribution you want to generate the random numbers right. So, you can see here N is the number of observations, mean is the vector of means, sd is the standard deviation and lower.tail. If this is TRUE, this is the default actually then the probabilities are computed like this probability X <=x which is something like your cumulative density function.

In case if you take lower.tail = FALSE then this probability of X >= x is computed. You see in this normal distribution this part, this part here is called as lower tail and this part here is called as upper tail right. So, if you want to compute here this probability this will be your here something if this is here some 0 then if this will be your here probability of minus Z then this will become here probability of Z. So, you want to specify whether you want to compute the probability on the left-hand side of the normal distribution or on the

right-hand side of the normal distribution. And in case if the values of mean and variance are not specified then μ that is mean is assumed to be 0 and standard deviation is assumed to be 1 that they are the default values right.

So, now let me try to take here some very simple examples to explain you that how you can compute different types of probabilities and how they are useful. So, suppose an apple formal says the apples in the boxes and the weight of the boxes vary and we and are assumed to be normally distributed with mean 20 and variance 4 that means μ = 20 kilogram and the variance is 4 kilogram square. You know that when the farmer is trying to pack the apples in the boxes means the apples have got different weights and you cannot ensure that every box had exactly say they said 20 kilograms of apples sometime there will be some variation maybe some grams. So, this is reflected by the value σ^2 and the farmer wants to avoid customer being unsatisfied because the boxes are those are too low in weight customers will not be happy. So, therefore, the farmer wants to know the probability that a box with the weight of less than 18 kilogram is sold.

That is a very genuine question because the farmer is trying to fill the apples in the box based on the approximate weight and if the weight is less than the then the customers may not feel happy and so the farmer wants to know its probability. So, if you try to translate this problem in the statistical term then I can say that farmer wants to know the probability that a box with the weight of less than 18 kilogram is sold that means if I say that x is the weight of the box or the apple then we want to compute the probability that x less than 18. So, this probability can be computed by the command pnorm and basically if you try to see when you try to compute that P(X<18) this is actually here $F_X(X = 18)$ this is the value of the CDF at x = 18. So, then we try to give here the command here pnorm(q) = 18 then min = 20 and then sd = square root of 4 which is here the variance and if you try to execute it on the R console this will come out to be 0.1586553.

So, you can see here it is very easy to compute such probability right and if I want to show you hit it on the R console so that you can be confident you can see here it is like this I try to copy and paste this command over here and if I give it here in the R console it will give you here the same value. So, now means you can be confident that these values are very easy to obtain right and on the other hand if you want to use here the option here lower.tail = TRUE means you can put it here and then what will happen the same value will be obtained here right because lower.tail = TRUE is the default right you can see here the screenshot that both the values are going to be the same. Now I try to modify this question yeah, I just want to show you how to compute different types of probabilities. Now suppose the farmer wants to know the probability that a box with a weight of more than 22 kg is sold right. So, that means the farmer wants to compute the probability that probability X greater than 22.

Now this is the place where you have to use the rules which I explained you to compute different types of probabilities. It is just like when I shown you that X probability X greater than 22 is less than 1 minus the area under the curve which is probability X <= 22 right. If you try to see on the curve this will be like this if somewhere it is here 22 suppose here and you want to compute here this probability right. So, this is going to be the 1 minus this remaining probability which is here written here like this one. So, and this quantity here is actually F(22).

So, if you try to see the trick, the trick is this I want to convert the given probability in the form of this cdf. Although in case if you try to use here the command lower dot till correctly then also then you do not need to make this transformation. So, both the approaches can be used and they will give the correct answer. So, now if you want to compute this probability that the first option is that you try to use the function here 1 minus pnorm() which is coming from here like this 1 minus pnorm() and then you try to write down here q = 20 which is coming here from this 22. And then you have to specify the mean = 20 and sd = square root of 4 which is and lower.tail = TRUE this is the default.

Even if you do not write there will be no problem and you get here this value or equivalently if you do not want to make it something like 1 minus F 22 then you can directly use this command to compute this probability that x greater than 22 is pnorm() q = 22 mean 20 sd square root of 4 and after that you have to use the command here lower dot tail = FALSE and if you try to use this command you will get here the same values. So, now it depends on you whichever way you feel convenient and comfortable you can use it there is absolutely no issue. Now I try to give you about some idea about different command this is a qnorm(). What is this qnorm() does? It gives you the quintile function and it calculates the quintile which is defined by the smallest value of x such that F(x) >= p. F x is the Sd f of x which is here like this at any point small x.

Well, you will see that when we try to conduct that it has the hypothesis and various other application this type of probability is required to be computed right. So, suppose if you want to know the value of the 90th quintile or say 90 percent quintile suppose the value here is q then it is described by probability that x less than = q is should be greater than or = 0.9. So, this can be obtained by the command here qnorm(). So, you simply have to write down here p = 0.9 which is coming from here and then you have to specify here the mean = 20 and L d = square root of 4 right. And if you try to execute this command here you will see here this is coming out to be 22.5631 right. And in case if you just use here the command the option here lower.tail = TRUE this is the default then again both the values are going to be the same and that you can see here in the screenshot right. So, let me try to show you these values on the R console before I try to move forward right.

So, in this case if you want to see that how I am computing this quantity probability that x greater than = 22 this is here like this and if you want to use here the other command that not by subtracting from 1, but by using the option lower.tail = FALSE then you can see here both are giving you the same value. So, this is about pnorm(). Now, similarly if you try to use here this qnorm() or the new qnorm(). So, you can see here that this is here like this. So, this is the giving you the value of the 90th quantile.

And similarly, if you try to use here the default part then it will be here the same value that lower.tail = TRUE will give you again the same value here right. So, let me come back to our slide. Now, I would like to give you the idea about this random number generation. You see whenever we are trying to compute different types of simulation or we want to experiment with something where we want to have a data which is originating from the normal distribution. So, this rnorm() function helps us in the generation of the random numbers from the normal distribution with the given mean and the given variance.

It is very helpful when we try to conduct different types of simulation studies Monte Carlo simulation etcetera. So, for the commands to generate the random number here is rnorm() r n o r m and inside the parenthesis we write here the n which is the number of random numbers to be generated then the value of the mean and then the value of standard deviation right. For example, if you want to generate 4 random numbers from a normal distribution whose mean is 20 and variance is 4. So, this can be obtained by the command here rnorm() n = 4 mean = 20 and sd = square root of 4. So, it will give you here these 4 values 1, 2, 3 and here 4.

So, these values have been generated from the normal distribution whose mean is 20 and variance is 4. The mean is 20 that is why you can see here most of the values are concentrated around 20 right and this is the screenshot, but when I try to execute this command in the R console now then these values will not be obtained because these are random numbers every time you try to generate the set of random numbers they will be different unless and until you use a command like set seed right. So, that is why I wanted to handle it separately and that is why I have shown you the earlier commands. Now, I will try to show you here these commands on the R console. So, if you try to see here let me clear the screen and you can see here rnorm() will give you here these 4 values and in case if you try to repeat it you can see here that this is going to give you different values and in case if you try to change here the number of observations you can see here now you are getting here 10 values.

And now if you try to suppose if I want to show you that what is the effect of variance suppose I try to make this variance high suppose I make it here 100. Now, if you try to compare the 4 values from this normal distribution you can see that although the mean is here 20, but the values are varying a lot 15, 17, 13, 14 right and in case if you want to

make this standard deviation to be very high, I have made you suppose 1000 that means the SD is square root of 1000. You can see here it is giving you the value like 52, -3.3, 15, 45 lots of variation is there among the values. Now, if you try to make it very small suppose I now make it here say square root of 1 you can see here the values are very close to 20.

So, by conducting this type of this experiment you can very easily means identify and understand different statistical properties of different probability density functions right. So, now let me come to an end to this lecture and now you can see here in this short lecture we have understood that how are we going to generate the random numbers and how we can compute different types of probabilities using the R software. Be it generation of random numbers, computation of the CDF, computation of the quantiles anything that is possible. Well, it depends on the problem that what you want to solve that suppose if you want to compute some probability the probability can be computed as a difference of the CDFs. Now, how to adjust those CDFs and how to take them for example, in the last lecture we had discussed that if X is following $N(\mu, \sigma^2)$.

So, then you can always make a transformation like X minus μ upon σ and then they will be following a $N(0,1)$ that is also possible otherwise if the R software now if you want to do it you do not need to do anything, but you can simply use the same mean μ and σ directly over here you do not need any transformation also. So, this is how you can compute various types of probabilities without any problem and the problem of going into the tables and choosing the value that is also solved. But my request to you all is here that now you try to take different types of examples, try to create different say portions in the curve of normal distribution and try to compute various areas they are simply your probabilities. For example, if the areas on the left-hand side, areas on the right-hand side, areas in the center. So, how can you compute different types of probabilities using the CDF because we have understood that probability that X lying between a and b is the same as F(b) – F(a), F is here the CDF.

So, now all those things you can verify here. Although I have explained you here only about the normal distribution, but these types of activities can be done for all type of probability distributions and probability mass functions. Be it binomial, be it Poisson, be it geometric, be it say gamma distribution, but my objective in the course was to introduce you with the normal distribution because we are going to use them in the forthcoming lectures. So, my request to you is that you please try to take up some problems you can consult the books you can create your own problem, but try to practice them on the R software. So, you try to practice it and I will see you in the next lecture with bivariate and multivariate normal distribution till then goodbye. Thank you.