Using probability distributions in R: dnorm, pnorm, qnorm, and rnorm

Basic Statistical Concepts for Data Science 1 October 29, 2018

Distribution functions in R

Every distribution has four associated functions whose prex indicates the type of function and the sux indicates the distribution. To exemplify the use of these functions, I will limit myself to the normal (Gaussian) distribution. The four normal distribution functions are:

dnorm: **density function** of the normal distribution

pnorm: **cumulative density function** of the normal distribution

qnorm: **quantile function** of the normal distribution **r**norm: **random sampling** from the normal distribution

The probability density function: dnorm

The probability density function (PDF, in short: density) indicates the probability of observing a measurement with a specic value and x thus the integral over the density is always 1. For a value , the normal density is dened as

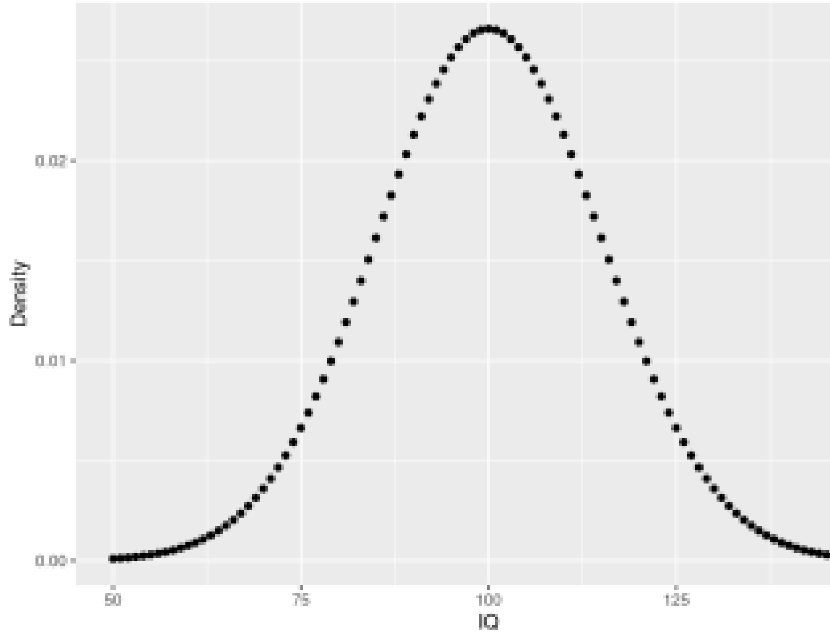
$$f(x \mid \mu, \sigma^2) = \frac{1}{\dots} \exp\left(-\frac{(x-\mu)^2}{\dots}\right)$$

$2\pi\sigma$ $\sqrt{}_2$ Other

$2\sigma^2$

μ σ σ^2
where is the mean, is the standard deviation, and is the variance.

Using the density, it is possible to determine the probabilities of events. For example, you may wonder: *What is the likelihood that a person has an IQ of exactly 140?*. In this case, you would need to retrieve the density of the IQ distribution at value 140. The IQ distribution can be modeled with a mean of 100 and a standard deviation of 15. The corresponding density is:




```
sample.range <- 50:150
iq.mean <- 100
iq.sd <- 15
iq.dist <- dnorm(sample.range, mean =
iq.mean, sd = iq.sd iq.df <-
data.frame("IQ" = sample.range, "Density" =
iq.d library(ggplot2)
ggplot(iq.df, aes(x = IQ, y = Density)) +


geom_point()
```

From these data, we can now answer the initial question as well as additional questions:

Software Engineering

Statistical Signicance Tests **TAGS**

DATA ANALYSIS

BAYESIAN METHODS

DOCUMENTATION HUGO LINEAR PREDICTION MODELS

HANDLING MATCHED DATA

NON-PARAMETRIC
SIGNIFICANCE TESTS

PARAMETRIC SIGNIFICANCE TESTS

PERFORMANCE MEASURES PLOTS PYTHON

R FOR APPLICATIONS IN DATA SCIENCE

```
pp <- function(x) {
 print(paste0(round(x * 100, 3), "%"))
}
# likelihood of IQ == 140?
pp(iq.df$Density[iq.df$IQ == 140])
```

```
## [1] "0.076%"
```

```
# likelihood of IQ >= 140?
pp(sum(iq.df$Density[iq.df$IQ >= 140]))
```

```
## [1] "0.384%"
```

```
# likelihood of 50 < IQ <= 90?
pp(sum(iq.df$Density[iq.df$IQ <= 90]))
```

```
## [1] "26.284%"
```

The cumulative density function: pnorm

The cumulative density (CDF) function is a monotonically increasing function as it integrates over densities via

$$1 \qquad x - \mu$$

$$f(x|\mu, \sigma) = [1 + \mathrm{erf}(-)]$$

$$2 \qquad \sigma\sqrt{2}$$

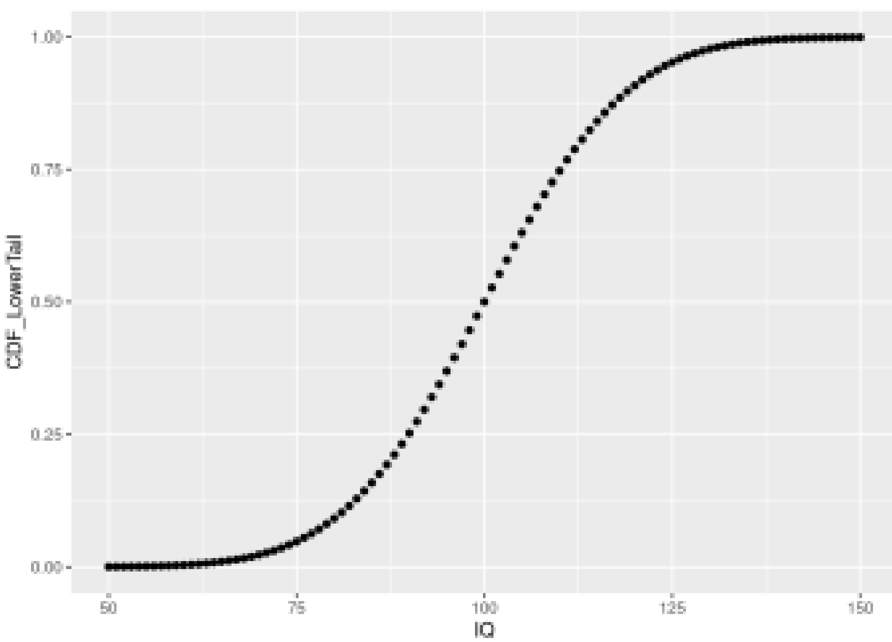
$$\sqrt{\pi}\int_{-x}^{x} e^{-t^2} 1$$

$$\mathrm{erf}(x) = dt$$

where is the error function. To get an intuition of the CDF, let’s

create a plot for the IQ data:

```
cdf <- pnorm(sample.range, iq.mean, iq.sd)
iq.df <- cbind(iq.df, "CDF_LowerTail" = cdf)
ggplot(iq.df, aes(x = IQ, y = CDF_LowerTail)) + geom_poi
```



As we can see, the depicted CDF shows the probability of having an IQ less or equal to a given value. This is because `pnorm` computes

$$P[X <= x]$$

the lower tail by default, i.e. . Using this knowledge, we can obtain answers to some of our previous questions in a slightly dierent manner:

```
# likelihood of 50 < IQ <= 90?
pp(iq.df$CDF_LowerTail[iq.df$IQ == 90])
```

```
## [1] "25.249%"
```

```
# set lower.tail to FALSE to obtain P[X >= x] cdf <-
pnorm(sample.range, iq.mean, iq.sd, lower.tail =  iq.df
<- cbind(iq.df, "CDF_UpperTail" = cdf)
# Probability for IQ >= 140? same value as before using d
pp(iq.df$CDF_UpperTail[iq.df$IQ == 140])
## [1] "0.383%"
```

Note that the results from *pnorm* are the same as those obtained from manually summing up the probabilities obtained via *dnorm*. Moreover, by setting `lower.tail = FALSE` , `dnorm` can be used to directly compute p-values, which measure how the likelihood of an observation that is at least as extreme as the obtained one.

To remember that `pnorm` does not provide the PDF but the CDF, just imagine that the function carries a *p* in its name such that *pnorm* is lexicographically close to *qnorm*, which provides the inverse of the CDF.

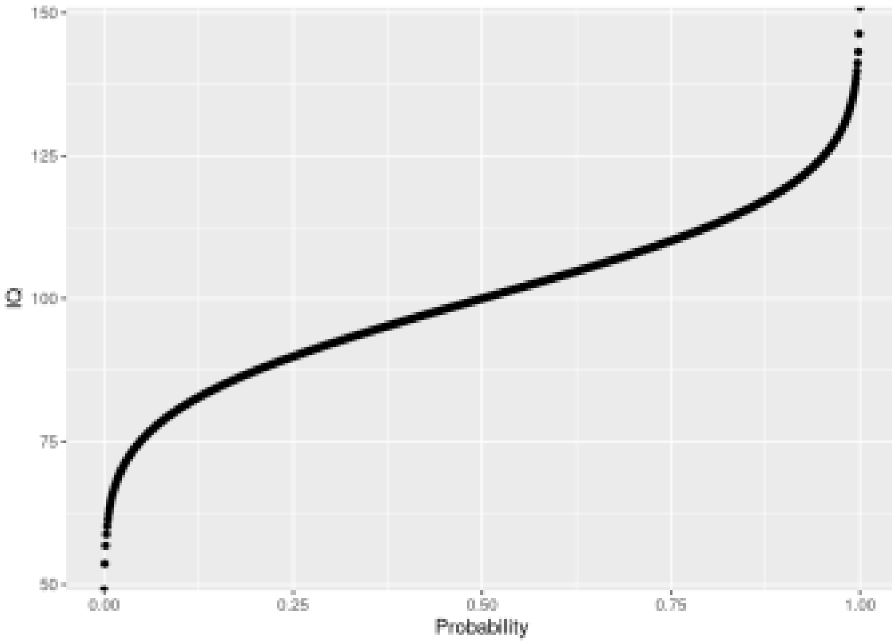
The quantile function: qnorm

The quantile function is simply the inverse of the cumulative density function (iCDF). Thus, the quantile function maps from probabilities

P[X <= x]

to values. Let's take a look at the quantile function for :

```
# input to qnorm is a vector of probabilities
prob.range <- seq(0, 1, 0.001)
icdf.df <- data.frame("Probability" = prob.range, "IQ" =
ggplot(icdf.df, aes(x = Probability, y = IQ)) + geom_poi
```



Using the quantile function, we can answer quantile-related questions:

```
# what is the 25th IQ percentile?
print(icdf.df$IQ[icdf.df$Probability == 0.25])
```

```
## [1] 89.88265
```

```
# what is the 75 IQ percentile?
print(icdf.df$IQ[icdf.df$Probability == 0.75])
```

```
## [1] 110.1173
```

```
# note: this is the same results as from the quantile fu
quantile(icdf.df$IQ)
```

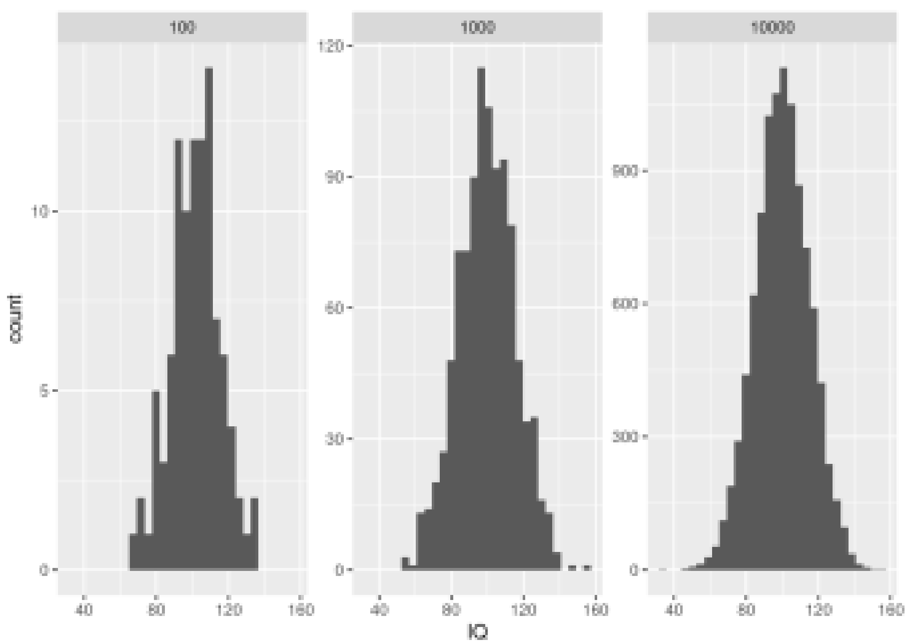
```
## 0% 25% 50% 75% 100%  ## -Inf 89.88265 100.00000
```

```
110.11735 Inf
```

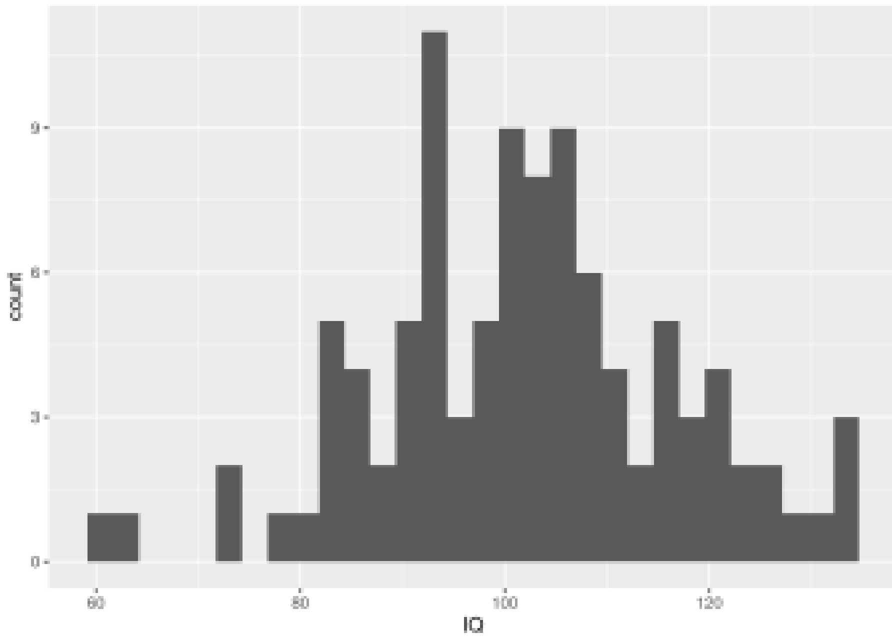
The random sampling function: rnorm

When you want to draw random samples from the normal distribution, you can use `rnorm`. For example, we could use `rnorm` to simulate random samples from the IQ distribution.

```
# fix random seed for reproducibility
set.seed(1)
# law of large numbers: mean will approach expected value
n.samples <- c(100, 1000, 10000)
my.df <- do.call(rbind, lapply(n.samples, function(x) dat
# show one facet per random sample of a given size
ggplot() + geom_histogram(data = my.df, aes(x = IQ)) + fa
```




```
# note: we can also implement our own sampler using the d
my.sample <- sample(iq.df$IQ, 100, prob = iq.df$Density,
my.sample.df <- data.frame("IQ" = my.sample)
ggplot(my.sample.df, aes(x = IQ)) + geom_histogram()
```



Note that we called `set.seed` in order to ensure that the random number generator always generates the same sequence of numbers for reproducibility.

Summary

Of the four functions dealing with distributions, `dnorm` is the most important one. This is because the values from `pnorm` , `qnorm` , and `rnorm` are based on `dnorm` . Still, `pnorm` , `qnorm` , and `rnorm` are very useful convenience functions when dealing with the normal distribution. If you would like to learn about the corresponding functions for the other distributions, you can simply call `?distribtuion` to obtain more information.

R

About Matthias Döring

Matthias Döring is a data scientist and AI architect. He is currently driving the digitization of the German railway system at DB Systel. Previously, he completed a PhD at the Max Planck Institute for Informatics in which he researched computational methods for improving treatment and prevention of viral infections.

Download Markdown

Visualizing Individual Data Points Using Scatter Plots

« PREVIOUS

Copy Code

Comments

Sam

07 Oct 20 07:00 UTC

Comparing Medians and Inter Quartile Ranges Using the Box Plot

NEXT »

Perhaps I'm wrong, but it seems to me that there is a mistake in the **interpretation** of the density function. With continuous random variables, the probability of having an IQ of 140 is **not** the value of the density function at 140. Technically, the probability of having a specic value with a continuous r.v. is always zero. If one wants to makes an approximation, then he would write

```
pnorm(140.1, mean=100,sd=15)-pnorm(139.9,mean=100,s
0.0001519534
```

Which is not identical to dnorm(140,mean=100,sd=15) 0.0007597324

The small interval around 140 which we use to make our calculations with, will depend on the level of precision with which we make our measurements, I believe.

Please enter your information. Your email address will not be published.

Your name
Your email address

Your message. Feel free to use Markdown.

I'm not a robot

reCAPTCHA
Privacy - Terms

SUBMIT