Experiment-2

siju.swamy@saintgits.org

07/07/2021

Probability Distributions Set operations, simulation of various properties. Bayes' rule. Generate and Visualize Discrete and continuous distributions using the statistical environment. Demonstration of CDF and PDF uniform and normal, binomial Poisson distributions. Students are expected to generate artificial data using the chosen statistical environment and explore various distribution and its properties. Various parameter changes may be studied.

Set operations in R

Performs set `union`, `intersection`, (asymmetric!) `difference`, `equality` and `membership` on two vectors.

Syntax: `union(x, y)`,`intersect(x, y)`, `setdiff(x, y)`, `setdiff(y, x)`, `setequal(x, y)`.

Example:

```
x <- c(sort(sample(1:20, 9)), NA)
y <- c(sort(sample(3:23, 7)), NA)
union(x, y)
```

```
##  [1]  1  2  3  4  5  6  7 11 19 NA  8 12 14 20 21 22
```

```
intersect(x, y)
```

```
## [1]  4 NA
```

```
setdiff(x, y)
```

```
## [1]  1  2  3  5  6  7 11 19
```

```
setdiff(y, x)
```

```
## [1]  8 12 14 20 21 22
```

```
setequal(x, y)
```

```
## [1] FALSE
```

```
## True for all possible x & y :
setequal( union(x, y),c(setdiff(x, y), intersect(x, y), setdiff(y, x)))
```

```
## [1] TRUE
```

```
is.element(x, y) # length 10
```

```
##  [1] FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE  TRUE
```

```
is.element(y, x) # length  8
```

```
## [1]  TRUE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE
```

Bayes' rule

```
library(LaplacesDemon)
```

```
## Warning: package 'LaplacesDemon' was built under R version 3.5.3
```

```
PrA <- c(0.75,0.25)
PrBA <- c(6/9, 5/7)
BayesTheorem(PrA, PrBA)
```

```
## [1] 0.7368421 0.2631579
## attr(,"class")
## [1] "bayestheorem"
```

Generate and Visualize Discrete and continuous distributions

Two common probabilities you will calculate in statistics is $P(X = x)$ and $P(X \leq x)$. `R` provides your some simple ways to calculate these probabilities. In this section, we will learn how to calculate probabilities for Discrete Distributions in `R`.

Binomial Distribution

The pdf of the Binomial distribution is given by

$$p(x) = \binom{n}{x} p^x (1-q)^{n-x}$$

.

The probability mass at $X = x$ can be calculated as `dbinom(x, size = 100, p = p)`. As an example $p(x = 7)$ for a binomial distribution with $n = 100, p = 0.5$ is shown bellow.

```
dbinom(7, size = 100, prob = 0.5)
```

```
## [1] 1.262774e-20
```

The cumulative probability density, CDF, $p(X \leq x)$ can be calculated using the syntax

```
pbinom(x, size = 100, p = p)
```

```
pbinom(7, size = 10, prob = 0.5)
```

```
## [1] 0.9453125
```

The Quantile or the Inverse CDF

Next, we can take a look at getting the quantile from a distribution. This is also known as the inverse CDF. To do this, we can use the distribution functions prefixed with a `q`. In our case, we will use `qbinom`.

Example:*

```
qbinom(0.9453125, size = 10, prob = 0.5)
```

```
## [1] 7
```

Generating Random Binomials

We can also generate random samples from any of the distributions. Again, we follow the same pattern of having a short name for the distribution, i.e. binom, preceded by a prefix, `r`. To generate random samples, we can use `rbinom`.

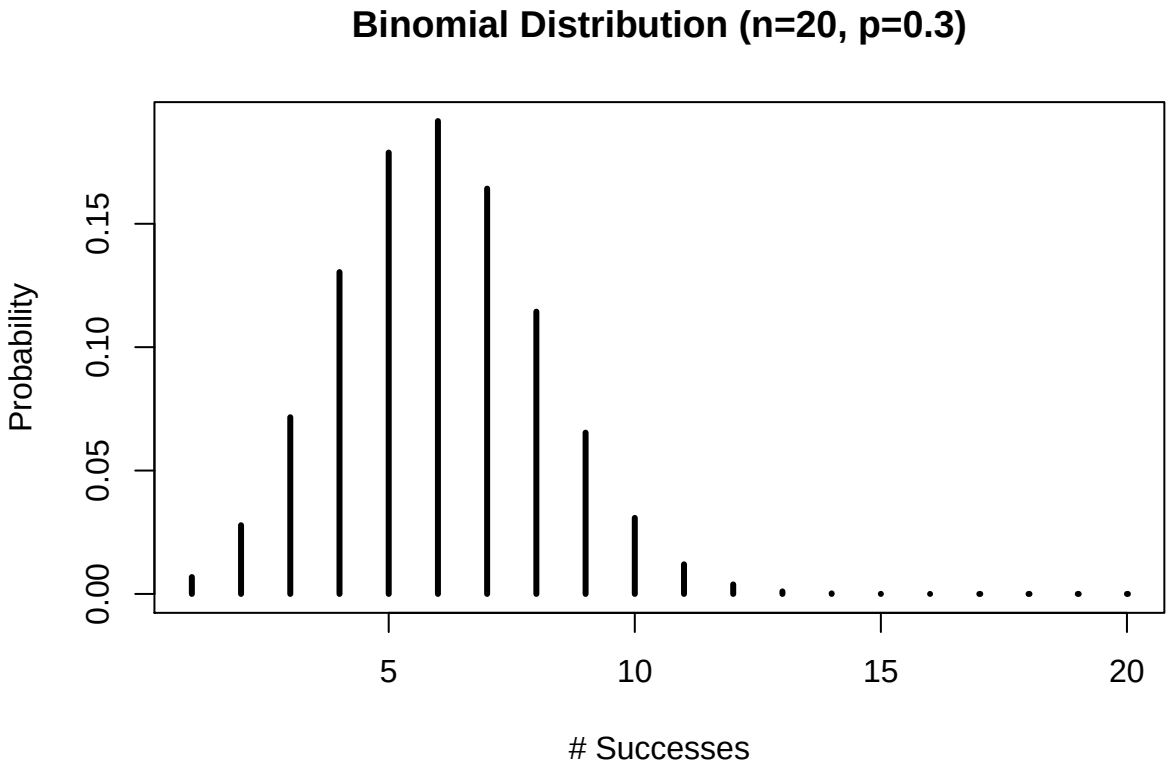
```r
rbinom(10, size =  10, prob = .5)
```

```
##  [1] 6 3 7 4 7 9 3 2 4 5
```

The example above will generate a random sample, which is multiple values of X from a Bin(10, .5) distribution. In our case, we supplied 10 as the first argument, thus our sample size will be 10.

Plotting Binomial variates

```r
success=1:20
plot(success,dbinom(success,size=20,prob=.3),
     type='h',
     main='Binomial Distribution (n=20, p=0.3)',
     ylab='Probability',
     xlab ='# Successes',
     lwd=3)
```



Poisson Distribution

The pdf of the Poisson distribution is given by

$$p(x) = \frac{e^{-\lambda}\lambda^x}{x!}$$

.

The probability mass at $X = x$ can be calculated as `dpois(x, lambda=lambda)`. As an example $p(x = 7)$ for a binomial distribution with $\lambda = 2$ is shown bellow.

```r
dpois(7, lambda= 2)
```

```
## [1] 0.003437087
```

The cumulative probability density, CDF, $p(X \leq x)$ can be calculated using the syntax

```
ppois(x, lambda=lambda)
```

```r
ppois(7, lambda=2)
```

```
## [1] 0.9989033
```

The Quantile or the Inverse CDF

Next, we can take a look at getting the quantile from a distribution. This is also known as the inverse CDF. To do this, we can use the distribution functions prefixed with a `q`. In our case, we will use `qbinom`.

Example:*

```r
qpois(0.9989033, lambda=2)
```

```
## [1] 8
```

Generating Random poisson variates

We can also generate random samples from any of the distributions. Again, we follow the same pattern of having a short name for the distribution, i.e. binom, preceded by a prefix, `r`. To generate random samples, we can use `rpois`.

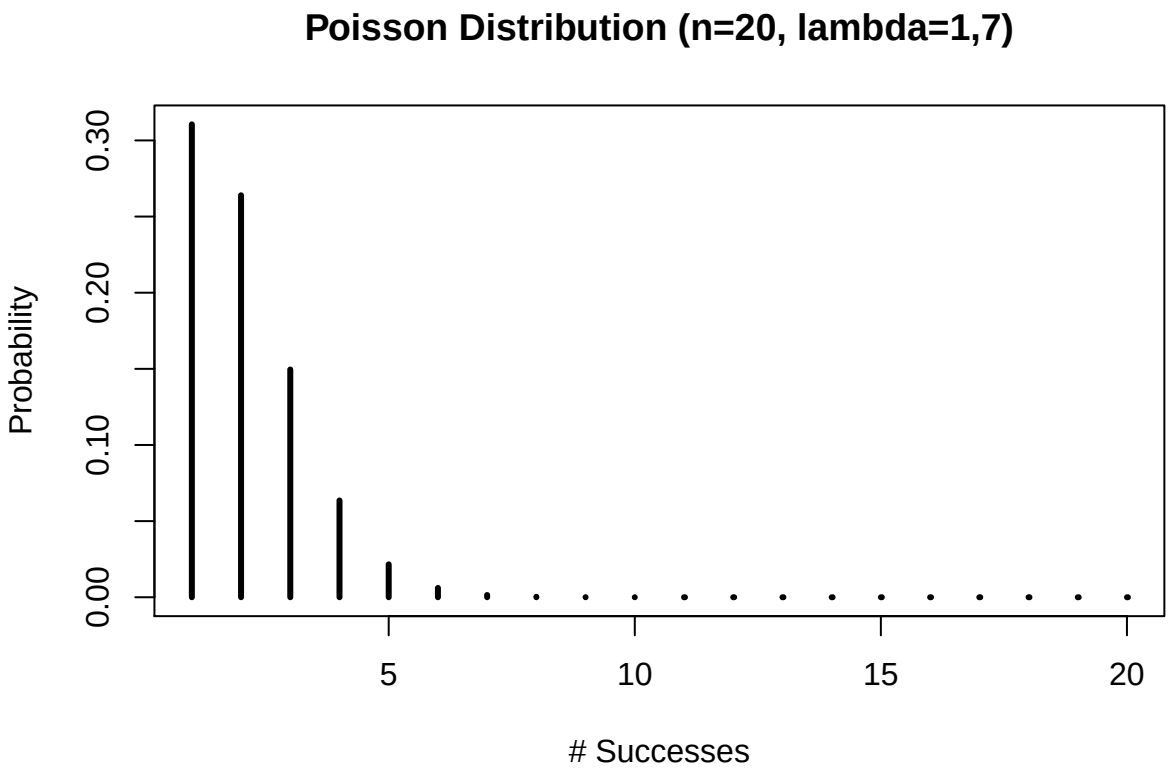
```r
rpois(10, lambda= 2)
```

```
##  [1] 0 5 1 2 2 4 6 1 2 2
```

The example above will generate a random sample, which is multiple values of X from a `Pois(2)` distribution. In our case, we supplied 10 as the first argument, thus our sample size will be 10.

Plotting Poisson distribution

```r
plot(success, dpois(success, lambda=1.7),
     type='h',
     main='Poisson Distribution (n=20, lambda=1,7)',
     ylab='Probability',
     xlab ='# Successes',
     lwd=3)
```




```
success=1:20
```

Continuous Distributions

To calculate probabilities for continuous distributions, you need to use the cumulative distribution function, CDF. Unlike discrete distributions, there is no single point probability, $P(X = x)$, there is a density. In this article, we will learn how to calculate probabilities of continuous distributions in `R`.

Normal Distribution

To calculate these probabilities, we can use function prefixed with `p` for each distribution. For example, to calculate the cdf of the normal distribution, we can use `pnorm`. So $p(X \leq 0.8)$ can be calculated as

```
pnorm(q = .8, mean = 0, sd = 1)
```

```
## [1] 0.7881446
```

The Quantile or the Inverse CDF

Next, we can take a look at getting the quantile from a distribution. This is also known as the inverse CDF. To do this, we can use the distribution functions prefixed with a `q`. In our case, we will use `qnorm`.

```
qnorm(0.7881446, mean = 0, sd = 1)
```

```
## [1] 0.8
```

Notice that our result is .8 which is exactly the value of `q` from the previous section. This shows how the two functions are inversely related.

Generating Random Samples

It is common to generate a random sample of normal variables. These are used in simulations and many other situations you will learn. To do this, we can follow a similar pattern of the above function using the `norm` preceded by a `r` for random.

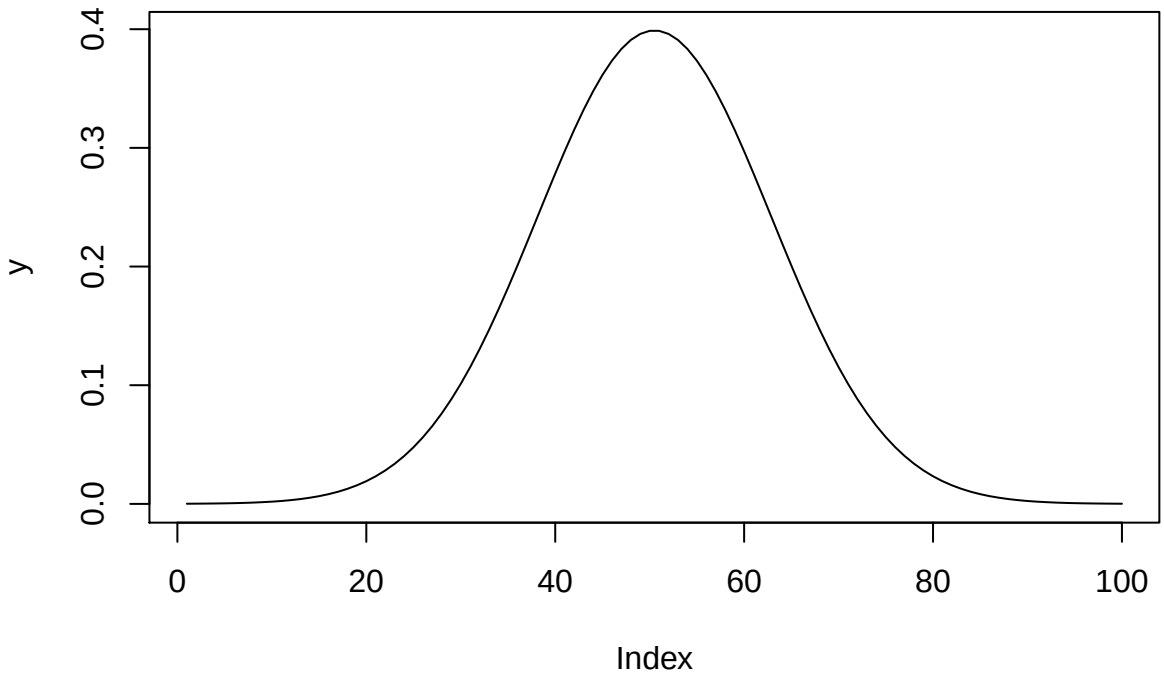
```
rnorm(10)
```

```
##  [1] -0.2022101 -0.3810489 -0.8260696  2.0792971  0.2705131  0.9257613
##  [7] -1.5199320 -0.3028444  1.1448967  1.2316226
```

The PDF

The pdf isn't very useful for continuous variables as you are usually looking for the probability which is the cumulative density. However, you can use the `dnorm` and other functions to generate the density plots.

```
x <- seq(-4, 4, length=100)
y <- dnorm(x)
plot(y, type='l')
```



Unfirom Distribution

In the similar way we can define the pmf at x as `dunif(x,min=a,max=b)`. An example is shown bellow ($p(x = 2)$)

```
dunif(x=2,min=2,max=5)
```

```
## [1] 0.3333333
```

The CDF can be calculated using the function `punfi(x,min=a,max=b)`.

```
punif(q=2.6,min=2,max=5)
```

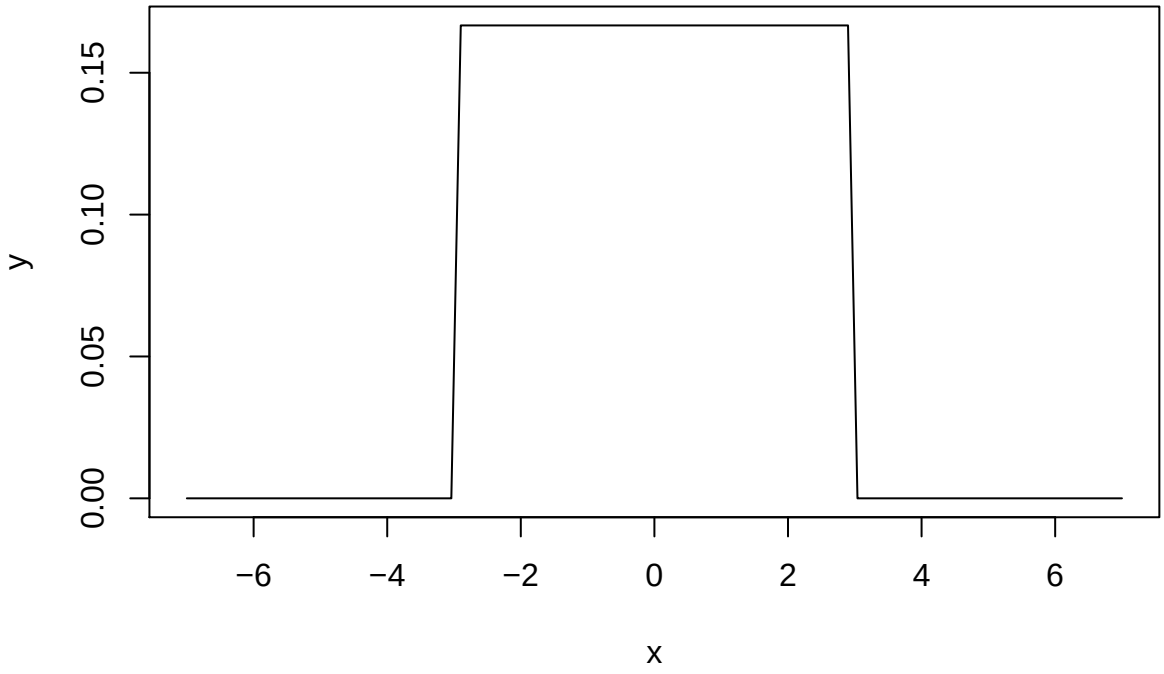
```
## [1] 0.2
```

The quantile for $q = 2.6$ can be found using

```
qunif(p=0.2,min=2,max=5)
```

```
## [1] 2.6
```

```
x <- seq(-7, 7, length=100)
y <- dunif(x,-3,3)
plot(x,y, type='l')
```



Creating Uniform random variate

```
runif(10,-2,2)
```

```
##  [1]  1.75599823 -0.81978116  0.32241904 -0.35738161 -0.90587040
##  [6]  1.79700286  0.88456872 -1.15194004 -0.12740296 -0.03275809
```