INTRODUCTION TO R

Nihan Acar-Denizli, Pau Fonseca

Probabilities

Working with probabilities in R

Initial stuff

- Load library prob
  - > library(prob)
- Some nice functions
  - tosscoin(n)
  - rolldie(n)
  - cards()

tosscoin

```
> tosscoin
function (times, makespace = FALSE)
{
    temp <- list()
    for (i in 1:times) {
        temp[[i]] <- c("H", "T")
    }
    res <- expand.grid(temp, KEEP.OUT.ATTRS = FALSE)
    names(res) <- c(paste(rep("toss", times), 1:times, sep = ""))
    if (makespace)
        res$probs <- rep(1, 2^times)/2^times
    return(res)
}
<environment: namespace:prob>
```

rolldie

```
> rolldie
function (times, nsides = 6, makespace = FALSE)
{
    temp = list()
    for (i in 1:times) {
        temp[[i]] <- 1:nsides
    }
    res <- expand.grid(temp, KEEP.OUT.ATTRS = FALSE)
    names(res) <- c(paste(rep("X", times), 1:times, sep = ""))
    if (makespace)
        res$probs <- rep(1, nsides^times)/nsides^times
    return(res)
}
<environment: namespace:prob>
```

cards

```
> cards
function (jokers = FALSE, makespace = FALSE)
{
    x <- c(2:10, "J", "Q", "K", "A")
    y <- c("Club", "Diamond", "Heart", "Spade")
    res <- expand.grid(rank = x, suit = y)
    if (jokers) {
        levels(res$rank) <- c(levels(res$rank), "Joker")
        res <- rbind(res, data.frame(rank = c("Joker", "Joker"),
            suit = c(NA, NA)))
    }
    if (makespace) {
        res$probs <- rep(1, dim(res)[1])/dim(res)[1]
    }
    return(res)
}
<environment: namespace:prob>
>
```

Computing probabilities

- > S = cards()
- > A = subset(S, suit == "Heart")
- > B = subset(S, rank %in% 7:9)

Name	Denoted	Defined by elements	Code
Union	$A \cup B$	in A or B or both	`union(A,B)`
Intersection	$A \cap B$	in both A and B	`intersect(A,B)`
Difference	$A \backslash B$	in A but not in B	`setdiff(A,B)`

Defining the Space

- We have 6 outcomes with the same probability

```
> p=rep(1/6, 6)
> p
[1] 0.1666667 0.1666667 0.1666667 0.1666667
0.1666667 0.1666667
```

Defining the Space

- If the outcomes represents a roll of a die we can use the example set.

```
> outcomes=rolldie(1) # a single roll
> outcomes
  X1
1  1
2  2
3  3
4  4
5  5
6  6
```

Defining the Space

- Now we create the probability space

```
> probspace(outcomes, probs=p)
  X1      probs
1  1 0.1666667
2  2 0.1666667
3  3 0.1666667
4  4 0.1666667
5  5 0.1666667
6  6 0.1666667
```

Defining the Space

- probspace(outcomes)

- We obtain here the same probability for each outcome, since in that case we are considering that each roll is equally likely.

Working with probabilities

```
> S <- cards(makespace = TRUE)
> A <- subset(S, suit == "Heart")
> B <- subset(S, rank %in% 7:9)
> prob(A)
```

- [1] 0.25
- Note that we can get the same answer with

```
> prob(S, suit == "Heart")
[1] 0.25
```

Working with probabilities

- The event argument is used to define a subset of x, that is, the only outcomes used in the probability calculation will be those that are elements of x and satisfy event simultaneously.
- In other words, prob(x, event) calculates
  - prob(intersect(x, subset(x, event)))

Working with probabilities

- First we define the space
  - S=rolldie(2,6,TRUE)

- Next we define two subsets

> A=subset(S,X1==X2)

> B=subset(S,X1+X2>=8)

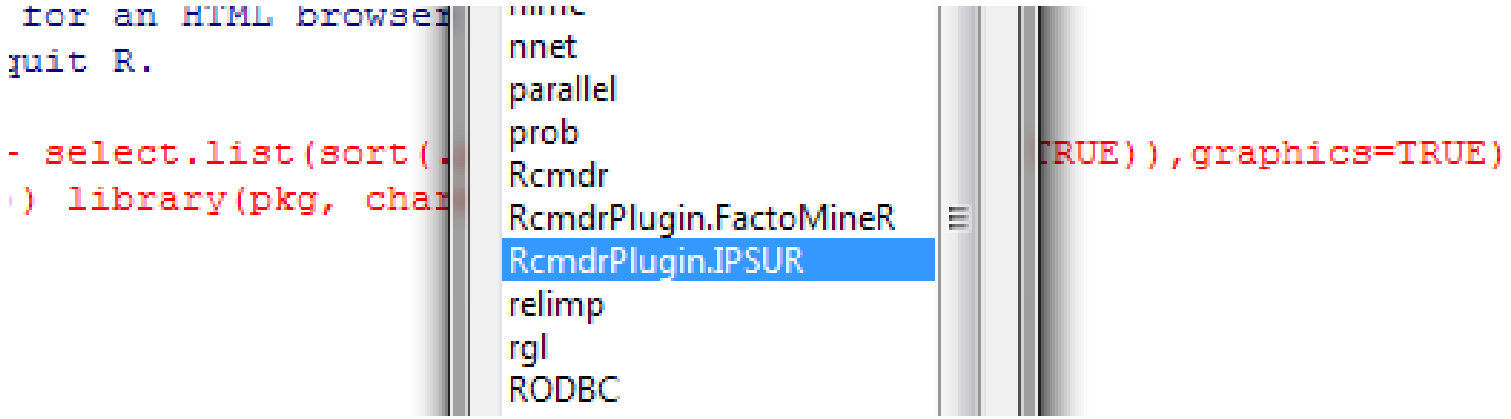
Working with conditional probabilities

- Now we can calculate the probabilities

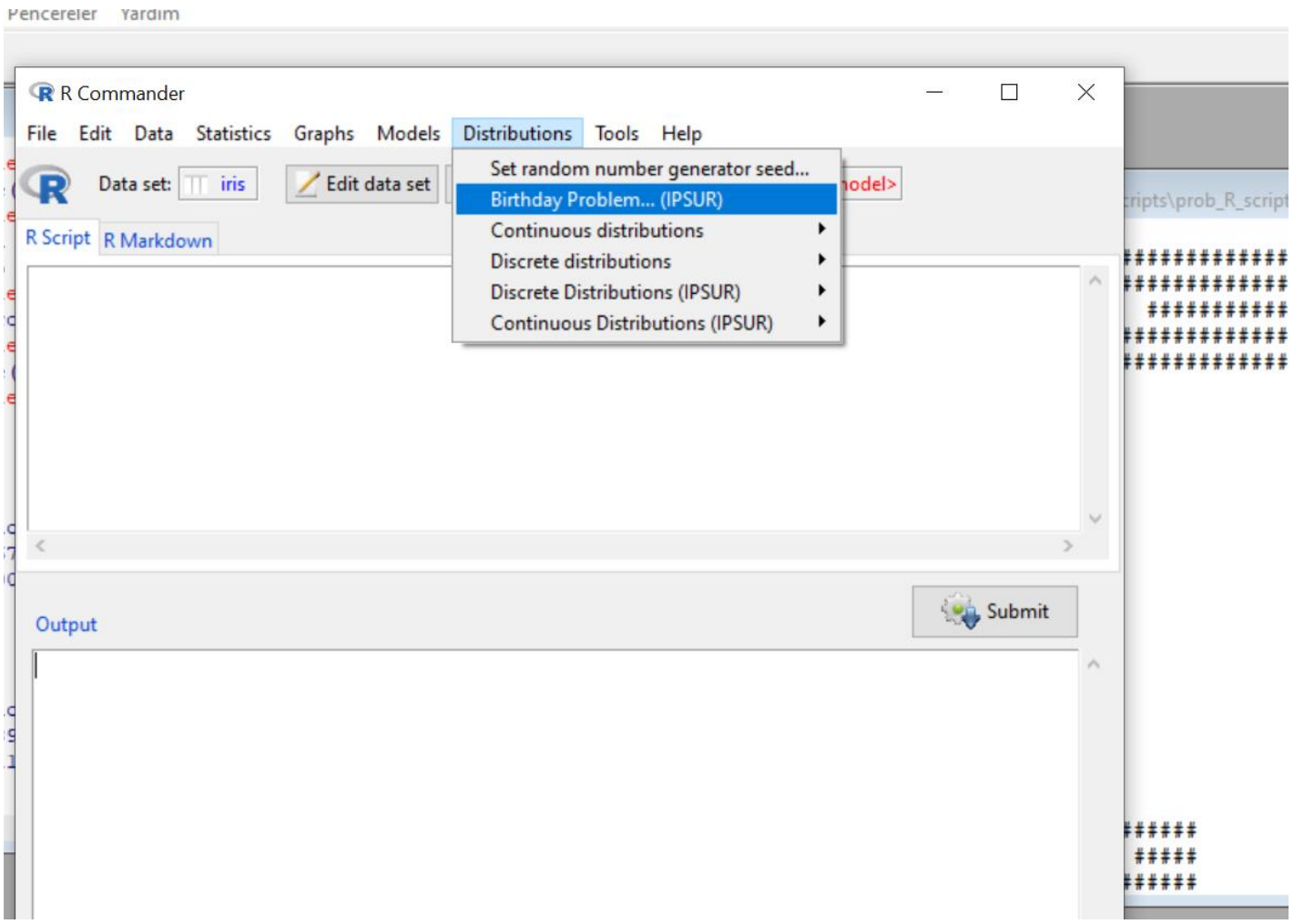
```
> prob(A, given=B)
[1] 0.2
> prob(B,given=A)
[1] 0.5
> prob(S, X1==X2, given = (X1 + X2 >= 8) )
[1] 0.2
> prob(S, X1+X2 >= 8, given = (X1==X2) )
[1] 0.5
```

Birthday Problem

- How many people are needed in a room so that the probability that there are two people whose birthdays are the exactly the same day is roughly ½?



Birthday Problem



Birthday Problem