Stats 102A (Computational Statistics) - Homework 5

Yaxi Lou

July 21, 2016

Data frames for the game board, and the two decks of cards

```r
gameboard <- data.frame(space = 1:40, title = c("Go" , "Mediterranean Avenue" , "Community Chest" , "Ba

chancedeck <- data.frame(index = 1:15, card = c("Advance to Go" , "Advance to Illinois Ave." , "Advance

communitydeck <- data.frame(index = 1:16, card = c("Advance to Go" , "Go to Jail" , "Bank error in your

dice <- function(){
    faces <- sample(1:6, 2, replace=TRUE)
    if(faces[1] == faces[2]) doubles = TRUE
    else doubles = FALSE
    movement = sum(faces)
    return(list(faces=faces, doubles=doubles, movement=movement))
}
```

Main Code

```r
# Store special positions first
community <- c(3, 18, 34)
chance <- c(8, 23, 37)
utilities <- c(13, 29)
railroads <- c(6, 16, 26, 36)

# Set basic status for a player
Player <- setRefClass("Player",
                      fields=list(
                        position="numeric", # current position of player
                        jail="logical",  # player is in jail
                        doubles="numeric",  # number of times the player has rolled double this turn
                        free="numeric"  # turn that the player can get out of jail
                        ))

# Draw a chance card
draw_chance <- function(player) {
  card <- sample(nrow(chancedeck), 1)
  update <- vector(mode="numeric")  # contains indexes that we need to update in table
  # Advance to Go
  if (card == 1) {
    player$position <- 1
    update <- c(update, 1)
  }
  # Advance to Illinois Ave
  if (card == 2) {
    player$position <- 25
    update <- c(update, 25)
  }
  # Advance to St Charles Place
  if (card == 3) {
    player$position <- 12
    update <- c(update, 12)
  }
  # Advance to nearest Utility
  if (card == 4) {
    if (player$position > 29) {
      player$position <- utilities[1]
      update <- c(update, utilities[1])
    } else {
      nearest <- min(which(utilities > player$position))
      player$position <- utilities[nearest]
      update <- c(update, utilities[nearest])
    }
  }
  # Advance to nearest Railroad
  if (card == 5) {
    if (player$position > 36) {
      player$position <- railroads[1]
      update <- c(update, railroads[1])
    } else {
      nearest <- min(which(railroads > player$position))
      player$position <- railroads[nearest]
      update <- c(update, railroads[nearest])
    }
  }
  # Take a ride on Reading Railroad
  if (card == 6) {
    player$position <- 6
    update <- c(update, 6)
  }
  # Take a walk on Boardwalk
  if (card == 7) {
    player$position <- 40
    update <- c(update, 40)
  }
  # Go to Jail
  if (card == 8) {
    player$position <- 11
    player$jail <- TRUE
    update <- c(update, 11)
  }
  # Go back 3 spaces
  if (card == 9) {
    player$position <- player$position - 3
    # Reset player position relative to end of board
    if (player$position < 1) {
      player$position <- player$position + 40
    }
    update <- c(update, player$position)
  }
  #For other cards, remain the same position
  return(update)
}

# Draw a community card
draw_community <- function(player) {
  card <- sample(nrow(communitydeck), 1)
  update <- vector(mode="numeric")  # contains indexes that we need to update in table
  # Advance to Go
  if (card == 1) {
    player$position <- 1
    update <- c(update, 1)
  }
  # Go to jail
  if (card == 2) {
    player$position <- 11
    player$jail <- TRUE
    update <- c(update, 11)
  }
  #For other cards, remain the same position
  return(update)
}

# Check if the space we landed on does anything special
check_space <- function(player, table, turn) {
  new_table <- table
  before <- player$jail
  # Go to Jail
  if (player$position == 31) {
    player$jail <- TRUE
    player$position <- 11
    player$free <- turn + 3
    new_table$count[player$position] <- new_table$count[player$position] + 1
  }
  # Landed on community chest
  if (player$position %in% community) {
    to_update <- draw_community(player)
    # Update the counts within table if we drew a card that moved the player
    for (i in to_update) {
      new_table$count[i] <- new_table$count[i] + 1
    }
    # If the player wasn't in jail before and is now in jail
    if (before == FALSE && player$jail == TRUE) {
      player$free <- turn + 3
    }
  }
  # Landed on Chance
  if (player$position %in% chance) {
    to_update <- draw_chance(player)
    # Update the counts within table if we drew a card that moved the player
    for (i in to_update) {
      new_table$count[i] <- new_table$count[i] + 1
```

```r
    }
    # If the player wasn't in jail before and is now in jail
    if (before == FALSE && player$jail == TRUE) {
      player$free <- turn + 3
    }
  }
  return(new_table)
}

# Runs the simulation
# Assumes 1 turn = player rolling dice
monopoly <- function(n, turns) {
  # Initialize the table that keeps track of where players land
  table <- data.frame(space=gameboard$space, title=gameboard$title, count=rep(0,40))
  for (i in 1:n) {
    # New player for every simulation
    player <- Player$new(position=1, jail=FALSE, doubles=0, free=0)

    for (turn in 1:turns) {
        repeat {
          reroll <- FALSE  # Determine if we need to re-roll
          # Free the player from jail
          current_roll <- dice()  # Roll the dice
          # Got a doubles
          if (current_roll$doubles == TRUE) {
            # Rolled a double to get out of jail
            if (player$jail == TRUE) {
              reroll <- FALSE  # Don't reroll again
              player$jail <- FALSE
            } else {
              # Player already had two doubles and got doubles the 3rd times in a row
              if (player$doubles == 2) {
                reroll <- FALSE
                player$jail <- TRUE
                player$free <- turn + 3
                player$position <- 11
                table$count[11] <- table$count[11] + 1
                break;
              } else {
                reroll <- TRUE   #Player rolls double the second time
                player$doubles <- player$doubles + 1
              }
            }
          }
          # 3rd turn in jail
          if (player$jail == TRUE && player$free == turn) {
            player$jail <- FALSE
          }
          # Update the player's current position only if player is not in jail
          if (player$jail == FALSE) {
            player$position <- player$position + current_roll$movement
          }
          # If the player's position goes over 40, reset relative to beginning of board
          if (player$position > 40) {
            player$position <- player$position %% 40
          }
          # Update the counts in the table
          table$count[player$position] <- table$count[player$position] + 1
          # Update the counts again if we moved the player during the check space
          if (player$jail == FALSE) {
            table <- check_space(player, table, turn)
          }
          # Turn has ended
          if (reroll == FALSE) {
            player$doubles <- 0
            break
          }
        }
      }
  }
  # Zero out the "Go to Jail" space because it doesn't count as "landed upon" per instructions
  table$count[31] <- 0
  return(table)
}

output<-monopoly(2000,100)
newdata<-as.data.frame(output)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
newdata %>% arrange(desc(count)) %>% mutate(ratio=count/sum(count))
```

```
##    space                 title count      ratio
## 1     11                  Jail 27684 0.11230285
## 2     25       Illinois Avenue  7108 0.02883430
## 3      1                    Go  6868 0.02786071
## 4     20       New York Avenue  6740 0.02734147
## 5     21          Free Parking  6648 0.02696826
## 6     19      Tennessee Avenue  6617 0.02684251
## 7     23                Chance  6363 0.02581213
## 8      6      Reading Railroad  6351 0.02576345
## 9     17       St. James Place  6306 0.02558090
## 10    26        B & O Railroad  6304 0.02557279
## 11    13      Electric Company  6267 0.02542270
## 12    18       Community Chest  6130 0.02486694
## 13    22       Kentucky Avenue  6110 0.02478581
## 14    29           Water Works  6103 0.02475742
## 15    12     St. Charles Place  6100 0.02474525
## 16    24        Indiana Avenue  5986 0.02428279
## 17    16 Pennsylvania Railroad  5929 0.02405157
## 18    27       Atlantic Avenue  5908 0.02396638
## 19    28        Ventnor Avenue  5893 0.02390553
## 20    15       Virginia Avenue  5839 0.02368647
## 21    34       Community Chest  5765 0.02338629
## 22    32        Pacific Avenue  5748 0.02331732
## 23    30        Marvin Gardens  5725 0.02322402
## 24    40             Boardwalk  5688 0.02307393
## 25    33 North Carolina Avenue  5681 0.02304553
## 26    35   Pennsylvania Avenue  5459 0.02214497
## 27     9        Vermont Avenue  5416 0.02197053
## 28     8                Chance  5320 0.02158110
## 29    10    Connecticut Avenue  5242 0.02126468
## 30    36   Short Line Railroad  5199 0.02109025
## 31     7       Oriental Avenue  5194 0.02106997
## 32     5            Income Tax  5162 0.02094016
## 33    14         States Avenue  5102 0.02069676
## 34    37                Chance  5070 0.02056695
## 35     4         Baltic Avenue  4736 0.01921205
## 36     3       Community Chest  4714 0.01912280
## 37    38            Park Place  4705 0.01908629
## 38    39            Luxury Tax  4704 0.01908224
## 39     2  Mediterranean Avenue  4628 0.01877393
## 40    31            Go to jail     0 0.00000000
```