P1

Jeremiah Theisen

2024-09-10

```r
loans = read.csv("https://github.com/TienChih/tbil-stats/raw/main/data/loans_full_schema.csv")

names(loans)
```

```
##  [1] "emp_title"                        "emp_length"
##  [3] "state"                            "homeownership"
##  [5] "annual_income"                    "verified_income"
##  [7] "debt_to_income"                   "annual_income_joint"
##  [9] "verification_income_joint"        "debt_to_income_joint"
## [11] "delinq_2y"                        "months_since_last_delinq"
## [13] "earliest_credit_line"             "inquiries_last_12m"
## [15] "total_credit_lines"               "open_credit_lines"
## [17] "total_credit_limit"               "total_credit_utilized"
## [19] "num_collections_last_12m"         "num_historical_failed_to_pay"
## [21] "months_since_90d_late"            "current_accounts_delinq"
## [23] "total_collection_amount_ever"     "current_installment_accounts"
## [25] "accounts_opened_24m"              "months_since_last_credit_inquiry"
## [27] "num_satisfactory_accounts"        "num_accounts_120d_past_due"
## [29] "num_accounts_30d_past_due"        "num_active_debit_accounts"
## [31] "total_debit_limit"                "num_total_cc_accounts"
## [33] "num_open_cc_accounts"             "num_cc_carrying_balance"
## [35] "num_mort_accounts"                "account_never_delinq_percent"
## [37] "tax_liens"                        "public_record_bankrupt"
## [39] "loan_purpose"                     "application_type"
## [41] "loan_amount"                      "term"
## [43] "interest_rate"                    "installment"
## [45] "grade"                            "sub_grade"
## [47] "issue_month"                      "loan_status"
## [49] "initial_listing_status"           "disbursement_method"
## [51] "balance"                          "paid_total"
## [53] "paid_principal"                   "paid_interest"
## [55] "paid_late_fees"
```

2.1.1

a. 1/6
b. 2/6 => 1/3
c. 6/6 => 1/1
d. 5/6

2.1.2

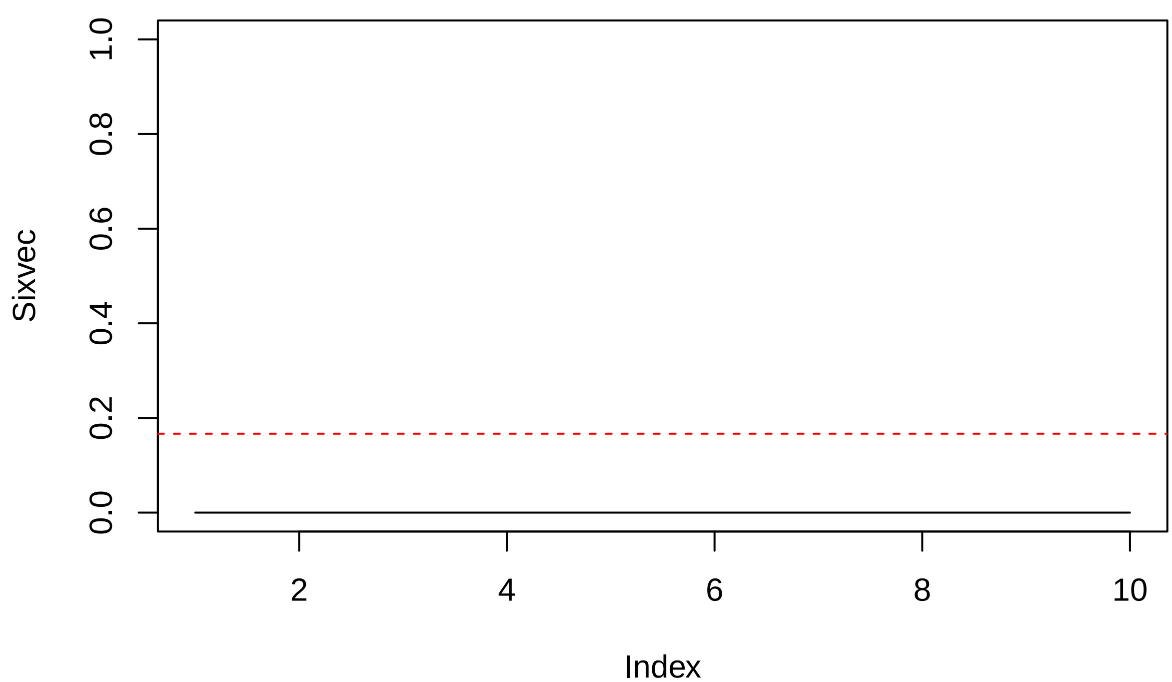
a.

```
n=10                #number of die rolls

sixes=0             # of sixes rolled so far
Sixvec=rep(NA, n) #proportion of sixes rolled

for (i in 1:n){
  roll=sample(1:6,1,replace=TRUE)
  if (roll==6){
    sixes=sixes+1   #increment number of sixes
  }
  Sixvec[i]=sixes/i #records proportion of sixes so far
}

plot(Sixvec, type="l", ylim=c(0,1)) #plots linegraph of proportion of sixes
abline(h=1/6, col="red", lty=2)     #draw y=1/6 line
```



b. The number of 6s changes drastically each time the dice are rolled.

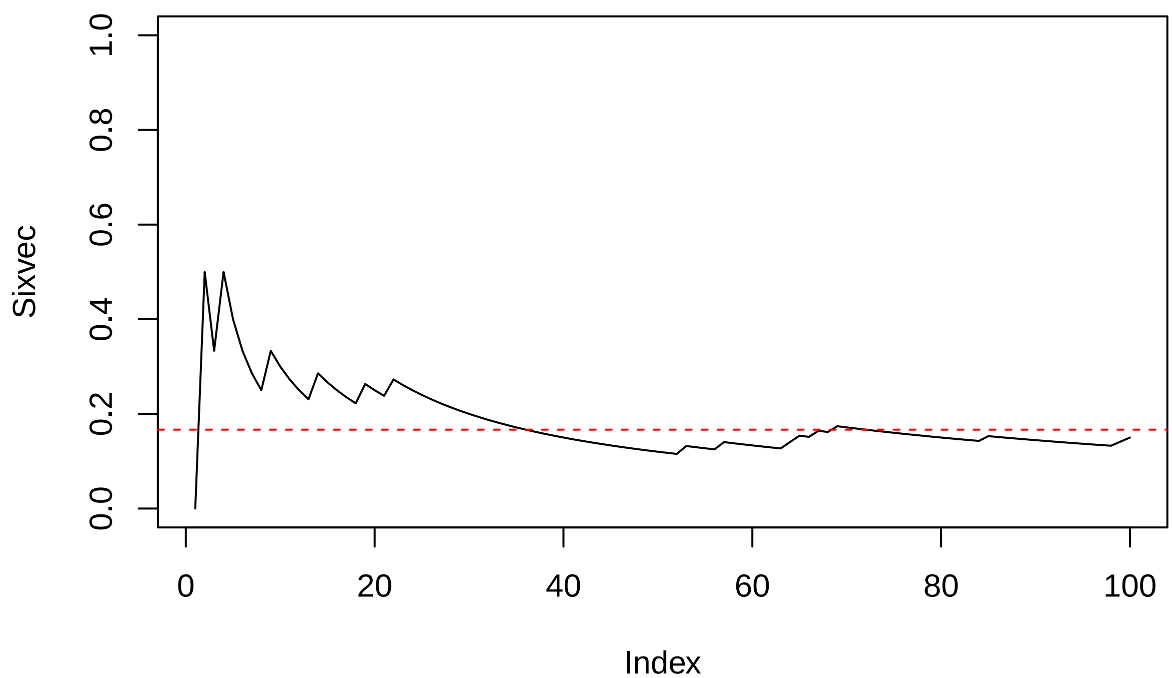
c.

```r
n=100                 #number of die rolls

sixes=0               # of sixes rolled so far
Sixvec=rep(NA, n) #proportion of sixes rolled

for (i in 1:n){
  roll=sample(1:6,1,replace=TRUE)
  if (roll==6){
    sixes=sixes+1   #increment number of sixes
  }
  Sixvec[i]=sixes/i #records proportion of sixes so far
}

plot(Sixvec, type="l", ylim=c(0,1)) #plots linegraph of proportion of sixes
abline(h=1/6, col="red", lty=2)      #draw y=1/6 line
```



d.

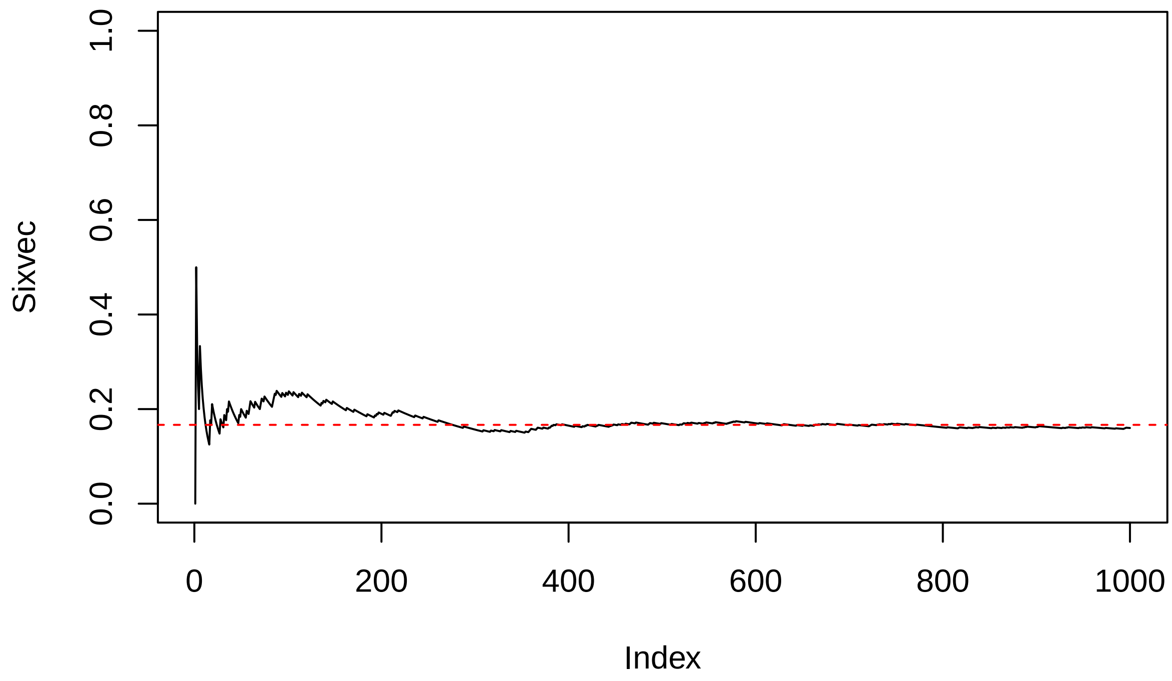
```r
n=1000                 #number of die rolls

sixes=0               # of sixes rolled so far
Sixvec=rep(NA, n) #proportion of sixes rolled

for (i in 1:n){
  roll=sample(1:6,1,replace=TRUE)
```

```
  if (roll==6){
    sixes=sixes+1   #increment number of sixes
  }
  Sixvec[i]=sixes/i #records proportion of sixes so far
}

plot(Sixvec, type="l", ylim=c(0,1)) #plots linegraph of proportion of sixes
abline(h=1/6, col="red", lty=2)      #draw y=1/6 line
```



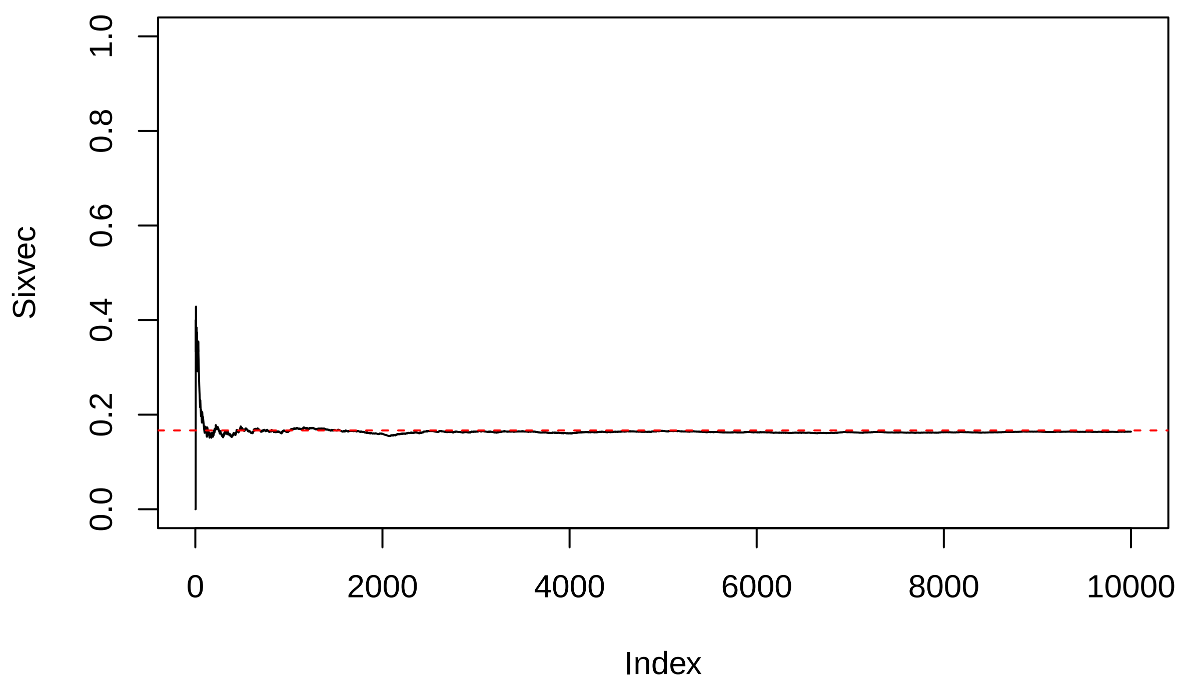
e.

```
n=10000                #number of die rolls

sixes=0               # of sixes rolled so far
Sixvec=rep(NA, n) #proportion of sixes rolled

for (i in 1:n){
  roll=sample(1:6,1,replace=TRUE)
  if (roll==6){
    sixes=sixes+1   #increment number of sixes
  }
  Sixvec[i]=sixes/i #records proportion of sixes so far
}
```

```r
plot(Sixvec, type="l", ylim=c(0,1)) #plots linegraph of proportion of sixes
abline(h=1/6, col="red", lty=2)      #draw y=1/6 line
```



f. The more times the dice is rolled, the number of 6s rolled approaches the expected value.

2.1.3

a. A {6, 7, 8, 9, 10}
b. B {2, 3, 5, 7}
c. A number is chosen which is greater than 5 which is prime
d. {7}
e. A number is either greater than 5 or prime
f. {2, 3, 5, 6, 8, 9, 10}
g. Any number that is not prime
h. {1, 4, 6, 8, 9, 10}
i. A number which is not greater than 5 and prime
j. {2, 3, 5}

2.1.4

a. The event will never happen, and X = 0, or S = infinity Side note: For continuous numbers, the probability of any one number 0.

b. The event will always happen, Y = S

c. P(Z) < 0 and P(Z) > 1 cannot happen because A is a subset of S, or is equal to S. Also, a set cannot have a negative cardinality.

2.1.5

a. 5/10 => 1/2
b. 4/10 => 2/5
c. 1/10 (A and B) = {7}
d. 8/10 (A or B) => {2, 3, 5, 7, 6, 8, 9, 10}
e. P(A or B) < P(A) < P(B) < P(A and B)
f. SKIP - Done in class
g. P(A) + P(B) = 9/10, it is off by 1/10, which is P(A and B)
h. P(A) = 5/10, P(Ac) = 5/10 P(B) = 4/10, P(Bc) = 6/10. The probability of a set and its complement should add up to the total sample size.

2.1.6

a.

```
length(which(loans$application_type=="joint"))
```

```
## [1] 1495
```

b.

```
length(which(loans$homeownership=="MORTGAGE"))
```

```
## [1] 4789
```

c.

```
length(which(loans$application_type=="joint" & loans$homeownership=="MORTGAGE"))
```

```
## [1] 950
```

d. P(J) = 0.1495 P(M) = 0.4795 P(M and J) = 0.095 P(M or J) = P(M) + P(J) - P(M and J) = 0.534

e.

```
index = sample(1:nrow(loans), 1000)
samp=loans[index,]

names(samp)
```

```
##  [1] "emp_title"                        "emp_length"
##  [3] "state"                            "homeownership"
##  [5] "annual_income"                    "verified_income"
##  [7] "debt_to_income"                   "annual_income_joint"
##  [9] "verification_income_joint"        "debt_to_income_joint"
## [11] "delinq_2y"                        "months_since_last_delinq"
## [13] "earliest_credit_line"             "inquiries_last_12m"
## [15] "total_credit_lines"               "open_credit_lines"
## [17] "total_credit_limit"               "total_credit_utilized"
## [19] "num_collections_last_12m"         "num_historical_failed_to_pay"
## [21] "months_since_90d_late"            "current_accounts_delinq"
## [23] "total_collection_amount_ever"     "current_installment_accounts"
## [25] "accounts_opened_24m"              "months_since_last_credit_inquiry"
## [27] "num_satisfactory_accounts"        "num_accounts_120d_past_due"
## [29] "num_accounts_30d_past_due"        "num_active_debit_accounts"
## [31] "total_debit_limit"                "num_total_cc_accounts"
## [33] "num_open_cc_accounts"             "num_cc_carrying_balance"
## [35] "num_mort_accounts"                "account_never_delinq_percent"
## [37] "tax_liens"                        "public_record_bankrupt"
## [39] "loan_purpose"                     "application_type"
## [41] "loan_amount"                      "term"
## [43] "interest_rate"                    "installment"
## [45] "grade"                            "sub_grade"
## [47] "issue_month"                      "loan_status"
## [49] "initial_listing_status"           "disbursement_method"
## [51] "balance"                          "paid_total"
## [53] "paid_principal"                   "paid_interest"
## [55] "paid_late_fees"
```

f.

```r
table(samp$application_type,samp$homeownership)
```

```
##
##              MORTGAGE OWN RENT
##   individual      364 102  373
##   joint            99  26   36
```

```
MORTGAGE OWN RENT
```

individual 365 117 386 joint 76 19 37

g. P(J) = 0.132, P(H) = 0.136 P(J and H) = 0.019. P(M) = 0.441. The proportion of Mortage havers and Joint loans are very similar to the number calculated in d.

2.1.7

a.
b.

```r
length(which(loans$state=="MS"))
```

```
## [1] 72
```

c.

```r
length(which(loans$state=="MS" & loans$issue_month=="Jan-2018"))
```

```
## [1] 29
```

d.

```r
index = sample(1:nrow(loans), 1000)
samp=loans[index,]

table(samp$state,samp$issue_month)
```

```
##
##      Feb-2018 Jan-2018 Mar-2018
##   AK        3        2        4
##   AL        0        5        4
##   AR        0        3        0
##   AZ        9        6        6
##   CA       41       50       44
##   CO       11        5        6
##   CT        3        6        7
##   DC        0        3        0
##   DE        1        0        0
##   FL       12       24       27
##   GA        7       12        8
##   HI        0        1        2
##   ID        4        0        0
##   IL       11        6       17
##   IN        2        2        5
##   KS        2        6        2
##   KY        0        2        1
##   LA        4        4        3
##   MA        5        7       14
##   MD        7       15        8
##   ME        1        2        2
##   MI       11        8        8
##   MN        2        6        3
##   MO        3        3        5
##   MS        2        2        0
##   MT        0        2        2
##   NC       10       20       15
##   NE        3        3        2
##   NH        2        2        3
##   NJ       11       19       13
##   NM        2        2        0
##   NV        5        9        6
```

```
##    NY        16        10        26
##    OH         5        14        13
##    OK         8        10         2
##    OR         4         3         4
##    PA        13        10        13
##    RI         1         4         3
##    SC         6         5         3
##    SD         2         0         1
##    TN         4         8         2
##    TX        27        35        42
##    UT         2         3         2
##    VA         4         3        11
##    VT         1         0         1
##    WA         7        13         7
##    WI         8         3         7
##    WV         3         1         1
##    WY         1         0         0
```

e. MS = 0.006 in sample, 0.0072 in loans. There are 0 loans from MS in Jan-2018 in sample, and 29 in the loans file. The probability for that in the file was 0.0029, which was really low, so it makes sense that in a random sample that never came up.