Manuel Beltran

Oct. 2, 2019

EE 381

**Project 2**

**Problem 1**

Introduction

A message is generated of being either 0 or 1. When that message, S, is being sent, there is a chance of error that the bit will change. The purpose of this experiment is to see the error probability of the message sent not matching the message received, R, over 100,000 trials.

Methodology

The message generated has a probability of being either 0 (.35) or 1(.65). If the message, S, being sent is 0, then the message received, R, is generated with a probability .96 chance of being 0 or .04 error of being 1. If the message being sent is 1, then the message received is generated like above but with .07/.93 probability. A failure is determined if S is not equal to R. Probability is determined by amount of failures over number of trials.

Results and Conclusion

The results below show that the probability of transmission error for all messages sent over each trial. There’s a low but still very possible chance of an message becoming erroneous after being sent. The error seems pretty close to the average of the 2 “probability of transmission error” of 0 and 1.

|  |  |
| --- | --- |
| Probability of transmission error |  |
| Ans. | p = 0.05961 |

An Appendix

import numpy as np

def nSidedDie(p):

n = len(p)

array = np.array(p)

cs = np.cumsum(array)

cp = np.append(0,cs)

r = np.random.rand()

for k in range(0,n):

if r > cp[k] and r <= cp[k+1]:

d = k + 1

return d

failure = 0

N = 100000

for x in range(N):

p= [0.35, 1-.35]

S = nSidedDie(p) -1

if S == 0:

R = nSidedDie([1-.04, .04]) - 1

else:

R = nSidedDie([.07, 1-.07]) - 1

if S != R:

failure += 1

failureRate = failure / N #.05961

**Problem 2**

Introduction

Like problem 1, the trial is conducted but this time the focus is looking at the success of the message being received, R, is 1 only if the message generated, S, is already 1.

Methodology

The process is almost the same except that only instances where S is equal to 1 is when it’s considered an event. When the event occurs, a success occurs when R is equal to 1. The probability of success is determined by the number of success over the total number of events.

Results and Conclusion

The results below show the conditional probability. The probability closely reflects the probability of a 1 being correctly sent given a error of transmission being .07.

|  |  |
| --- | --- |
| Conditional probability P(R = 1 | S = 1) |  |
| Ans. | p = 0.93024 |

An Appendix

import numpy as np

def nSidedDie(p):

n = len(p)

array = np.array(p)

cs = np.cumsum(array)

cp = np.append(0,cs)

r = np.random.rand()

for k in range(0,n):

if r > cp[k] and r <= cp[k+1]:

d = k + 1

return d

success = 0

total = 0

N = 100000

for x in range(N):

p= [0.35, 1-.35]

S = nSidedDie(p) -1

if S == 1:

total += 1

R = nSidedDie([.07, 1-.07]) - 1

if S == R:

success += 1

successRate = success / total #.93024

**Problem 3**

Introduction

Like problem 1, the trial is conducted but this time the focus is looking at the success of the message being generated, S, is 1 only if the message sent, R, is already 1.

Methodology

The process is almost the same except that only instances where R is equal to 1 is when it’s considered an event. When the event occurs, a success occurs when S is equal to 1. The probability of success is determined by the number of success over the total number of events.

Results and Conclusion

The results below show the conditional probability of S generating 1 given R received is 1.

|  |  |
| --- | --- |
| Conditional probability P(S = 1 | R = 1) |  |
| Ans. | p = 0.97706 |

An Appendix

import numpy as np

def nSidedDie(p):

n = len(p)

array = np.array(p)

cs = np.cumsum(array)

cp = np.append(0,cs)

r = np.random.rand()

for k in range(0,n):

if r > cp[k] and r <= cp[k+1]:

d = k + 1

return d

success = 0

total = 0

N = 100000

for x in range(N):

p= [0.35, 1-.35]

S = nSidedDie(p) -1

if S == 0:

R = nSidedDie([1-.04, .04]) - 1

else:

R = nSidedDie([.07, 1-.07]) - 1

if R == 1:

total += 1

if S == 1:

success += 1

successRate = success / total #.97706

**Problem 4**

Introduction

Like problem 1, the trial is conducted but this time the message generated, S, is sent 3 times. The 3 received messages, R, is considered a failure if the majority does not match S. The trial is conducted 100,000 times.

Methodology

The process is almost the same as problem 1 except that there are 3 received messages instead of 1. The message received is to be considered the majority of the 3 received. If the sum of the messages is less than 2, than the majority is 0. Otherwise the majority is 1. The trial is considered a failure if the S does not match the message received. The probability of error is the number of failures over the number of trials.

Results and Conclusion

The results below show the probability of error is much less if the attempted number of sends are increased and only the majority is considered.

|  |  |
| --- | --- |
| Probability of error with enhanced transmission |  |
| Ans. | p = 0.01064 |

An Appendix

import numpy as np

def nSidedDie(p):

n = len(p)

array = np.array(p)

#

cs = np.cumsum(array)

cp = np.append(0,cs)

r = np.random.rand()

for k in range(0,n):

if r > cp[k] and r <= cp[k+1]:

d = k + 1

return d

failure = 0

N = 100000

p= [0.35, 1-.35]

for x in range(N):

D = 1

S = nSidedDie(p) - 1

R = []

for i in range(3):

if S == 0:

R.append(nSidedDie([1-.04, .04]) - 1)

else:

R.append(nSidedDie([.07, 1-.07]) - 1)

if sum(R) < 2:

D = 0

if S != D:

failure += 1

failureRate = failure / N #.01064