EE511
Akshay Kenchappa Manjunath
USC id: 1777012718

PROJECT 1

22 january 2020

Introduction

In probability theory and statistics, the Bernoulli distribution, named after Swiss mathematician Jacob Bernoulli,[1] is the discrete probability distribution of a random variable which takes the value 1 with probability p and the value 0 with probability q=1-p,that is, the probability distribution of any single experiment that asks a yes–no question; the question results in a boolean-valued outcome, a single bit whose value is success/yes/true/one with probability p and failure/no/false/zero with probability q. It can be used to represent a (possibly biased) coin toss where 1 and 0 would represent "heads" and "tails" (or vice versa), respectively, and p would be the probability of the coin landing on heads or tails, respectively. In particular, unfair coins would have $p \neq 1/2.p \neq 1/2.$

The Bernoulli distribution is a special case of the binomial distribution where a single trial is conducted (so n would be 1 for such a binomial distribution). It is also a special case of the two-point distribution, for which the possible outcomes need not be 0 and 1.

Question 1

Q1.Simulate tossing a fair coin (a Bernoulli trial) 50 times. Count the number of heads. Record the longest run of heads.

```
'''q1 - Simulating tossing a fair coin 50 times and count the number of heads.
       Record longest run of heads'''

import numpy as np
import matplotlib.pyplot as plt

simu = np.random.binomial(n=1, p=0.5, size = 50)  #draw samples from a binomial distribution
                                                  #n=1 for Bernoulli trial

num_heads = sum(simu)   #count the number of heads
count = 0
head_runs = 0
for i in simu:
    if i == 1:
        count += 1
        if count >= head_runs:
            head_runs = count
    if i == 0:
        count = 0

print ("Number of heads = ", num_heads)
print ("Longest Head run length = ", head_runs)
```

```
Number of heads =  26
Longest Head run length =  6
```

Figure 1: Q1 code and output

Q1(a).Repeat the above experiment 20, 100, 200, and 1000 times. Generate a histogram for each showing the number of heads in 50 flips. Comment on the limit of the histogram.

```
''' q1(a) - Simulating the experiment 20, 100, 200, 1000 times.
    Generating a Histogram for each showing number of heads in 50 flips.'''

num_repeat = 1000
head_arr = np.empty([num_repeat,1])
for i in range(num_repeat):
    num_heads = 0
    for j in range(50):
        if np.random.sample() < 0.5:
            num_heads += 1
    head_arr[i] = num_heads

bins = np.arange(51+1)-0.5

plt.hist(head_arr, bins, facecolor = 'red', edgecolor = 'black', alpha = 0.75)
plt.xticks(range(51))
plt.xlim([10, 40])
plt.xlabel("Number of heads")
plt.ylabel("count")
plt.title("Histogram showing number of heads in 50 flips")
plt.show
```

Figure 2: code

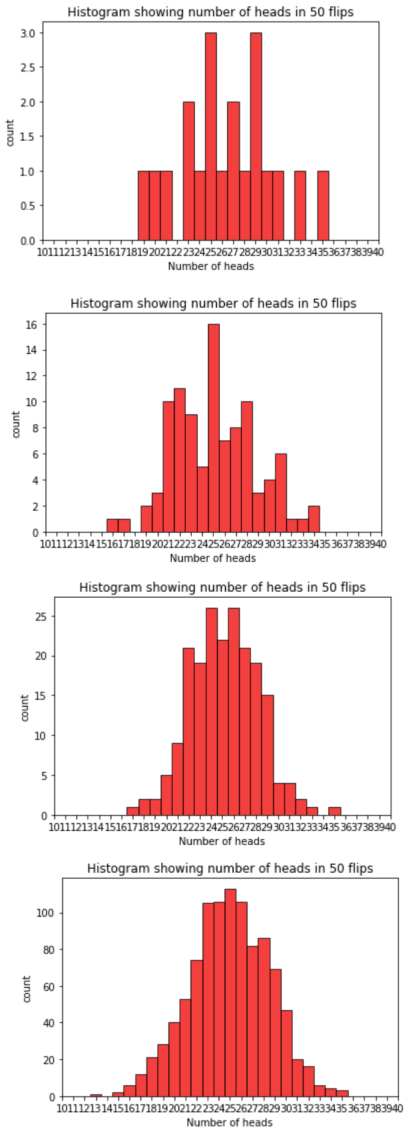

Figure 3: Output Q1(a)

Question 2

Simulate tossing a biased coin 200 times where P[HEAD] = 0.80. Count the number of heads. Record the longest run of heads. Generate a histogram for the Bernoulli outcomes.

```
'''q2 - Simulating tossing a biased coin 200 times with P[head] = 0.8 and counting number of heads.
        Record longest run of heads. To Generate a histogram for the outcomes'''

import numpy as np
import matplotlib.pyplot as plt

tosses = 200
record = np.empty([tosses, 1])
num_heads = 0
count = 0
head_runs = 0

for i in range(tosses):
    if np.random.sample() < 0.8: #heads
        num_heads += 1
        count += 1
        if count >= head_runs:
            head_runs = count
        record[i] = 1

    else:            #tails
        count = 0
        record[i] = 0

print ("Number of heads = ", num_heads)
print ("Longest Head run length = ", head_runs)

binwidth = 1
bins = np.arange(min(record)-0.5,max(record)+0.5+binwidth,binwidth)

plt.hist(record, bins, facecolor = 'red', edgecolor = 'black', alpha = 0.75)
plt.xlabel("Tails = 0, Heads = 1")
plt.ylabel("count")
plt.title("Histogram for Bernoulli Outcomes")
plt.grid(True)
plt.show()
```

```
Number of heads =  162
Longest Head run length =  16
```

Figure 4: Q2 code and output

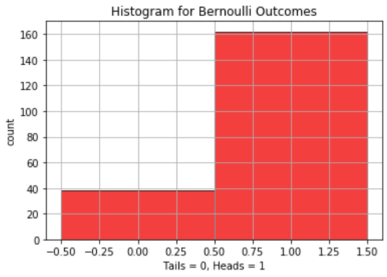

Figure 5: Output Q2

Question 3

Simulate tossing a fair coin 100 times. Generate a histogram showing the heads run lengths.

```
'''q3 - Simulating tossing a fair coin 100 times.
       To generate a histogram showing head run lengths'''

import numpy as np
import matplotlib.pyplot as plt

tosses = 100
record_run = np.empty([tosses, 1])
count = 0
head_runs = 0
times = 0

for i in range(tosses):
    if np.random.sample() >= 0.5:
        if count != 0:
            record_run[times] = count
            times += 1
        count = 0
    else:
        count += 1
        if count >= head_runs:
            head_runs = count

    if i == tosses-1 and count != 0:
        record_run[times] = count

print ("Longest Head run length = ", head_runs)

bins = np.arange(51+1)-0.5

plt.hist(record, bins, facecolor = 'red', edgecolor = 'black', alpha = 0.75)
plt.xlim([-1, 10])
plt.xlabel("Head run lengths")
plt.ylabel("count")
plt.title("Histogram showing the head run lengths")
plt.show()
```

```
Longest Head run length =  4
```

Figure 6: Q3 code and Output

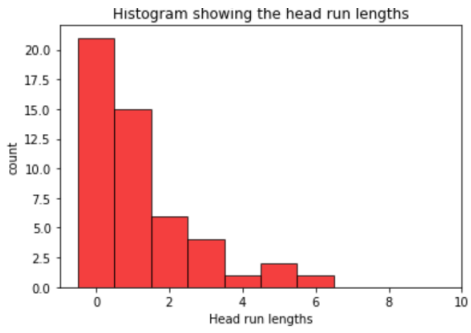

Figure 7: Output Q3

Question 4

Simulate tossing a fair coin and count the number of tosses until reaching a user-specified positive number of heads.

```
'''q4 - Simulating tossing a fair coin and to count the number of tosses
        until reaching user specified number of heads'''

import numpy as np
import matplotlib.pyplot as plt

heads_desired = int(input("Enter the number of heads desired"))
num_heads = 0
tosses = 0

while num_heads < heads_desired:
    if np.random.sample() < 0.5:
        num_heads += 1
        tosses += 1
    else:
        tosses += 1

print("Number of tosses = ", tosses)
```

```
Enter the number of heads desired 7
Number of tosses =  14
```

Figure 8: Q4 code and Output

References

[1] https://en.wikipedia.org/wiki/Bernoulli_distribution

[2] https://docs.scipy.org/doc/numpy-1.14.0/reference/generated/numpy.random.binomial.html

[3] https://alexamarioarei.github.io/Research/docs/LongestHrunReview.pdf

[4] https://www.csun.edu/~hcmth031/tlroh.pdf

[5] https://docs.scipy.org/doc/numpy/reference/generated/numpy.histogram.html

[6] https://docs.scipy.org/doc/numpy/reference/generated/numpy.arange.html