CS 237 Lab 1

Vidya Akavoor

Collaborators: Aaron Heckman, Jesse Fimbres

Sources:

<http://matplotlib.org/api/pyplot_api.html>

<https://docs.python.org/2/library/random.html>

<http://stackoverflow.com/questions/477237/how-do-i-simulate-flip-of-biased-coin-in-python>

Late Days Used: 0

Part 1 – Python

Question 1

from matplotlib import pyplot as plt

import numpy as np

#this is the red dotted line

plt.figure(1)

x\_coord = [1,2,3,4]

y\_coord = [1,2,3,4]

plt.plot(x\_coord, y\_coord, marker = '', linestyle = ':', color = 'r')

plt.show()

#this is the blue triangle line

plt.figure(2)

x\_coord1 = [1,2,3,4]

y\_coord1 = [1,2,3,4]

plt.plot(x\_coord1, y\_coord1, marker = '^', linestyle = '-', color = 'b')

plt.show()

Question 2

#only my function, not the original

def dieroll(m):

sum\_list = [0 for x in range(10000)]

for i in range(10000):

roll\_sum = 0

for j in range(m):

roll = random.choice(range(1,7))

roll\_sum += roll

sum\_list[i] = roll\_sum

plt.figure(1)

plt.hist(sum\_list, bins = 6\*m, rwidth = .7, align = 'mid',\

weights = np.zeros\_like(sum\_list) + 1. / len(sum\_list))

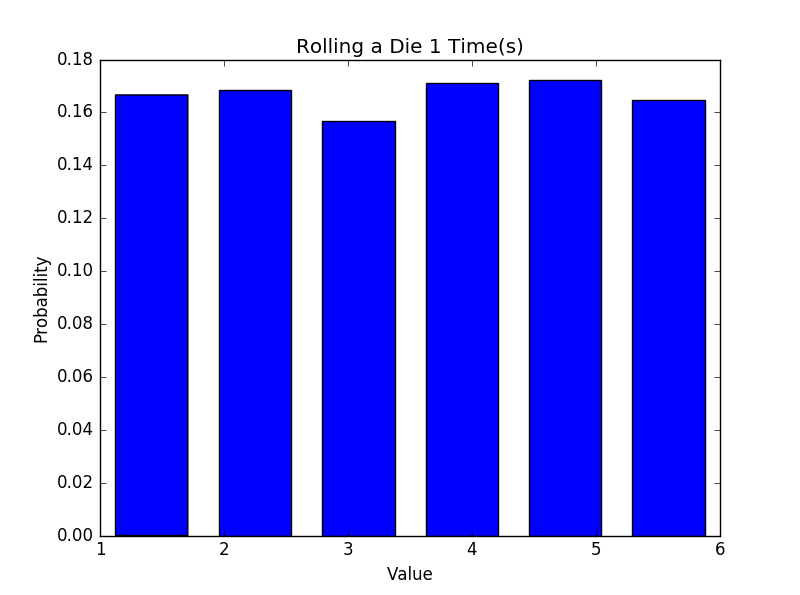
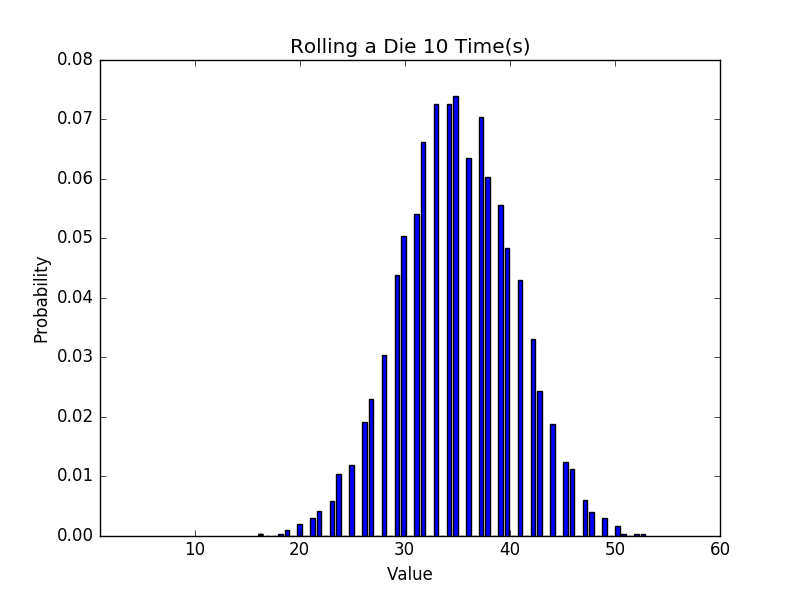
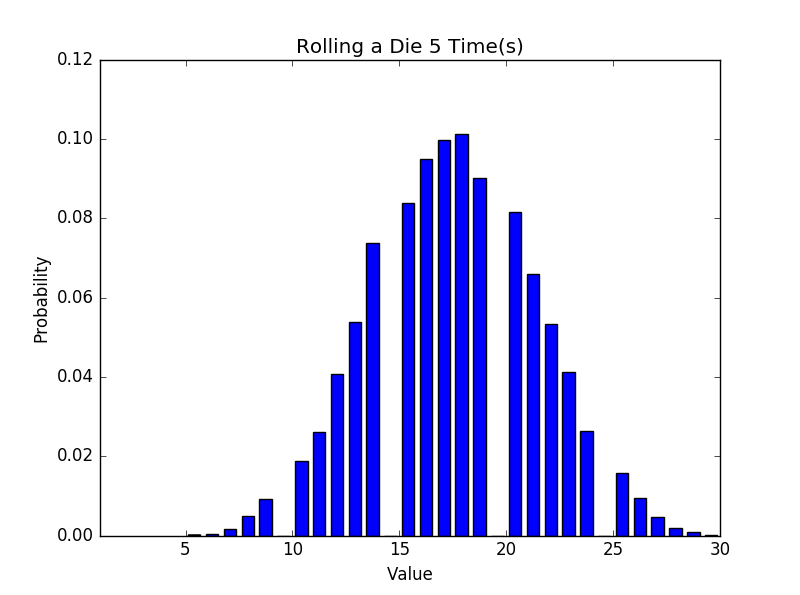
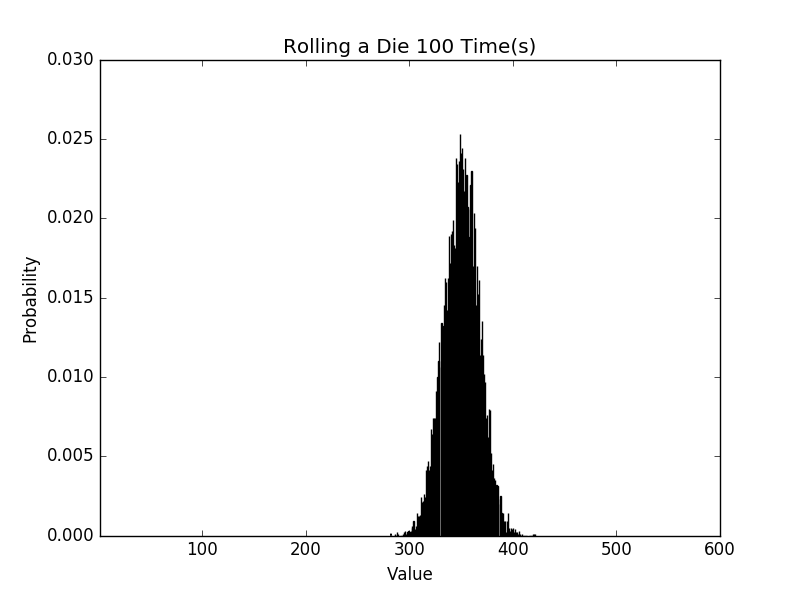
plt.xlim(1,6\*m)

plt.title("Rolling a Die " + str(m) + " Time(s)")

plt.xlabel("Value")

plt.ylabel("Probability")

plt.show()

Question 3

import random

#helper function to determine whether point is in circle

def draw\_point():

x = random.uniform(-1.0, 1.0)

y = random.uniform(-1.0, 1.0)

if ((x\*\*2) + (y\*\*2)) <= 1.0:

return True

else:

return False

def pi\_Estimate(n):

T = 0.0

C = 0.0

for i in range(n):

if draw\_point() == True:

C += 1

T += 1

estimate = 4\*(C/T)

print(C, 'points in the circle out of', T, 'total so pi is estimated to

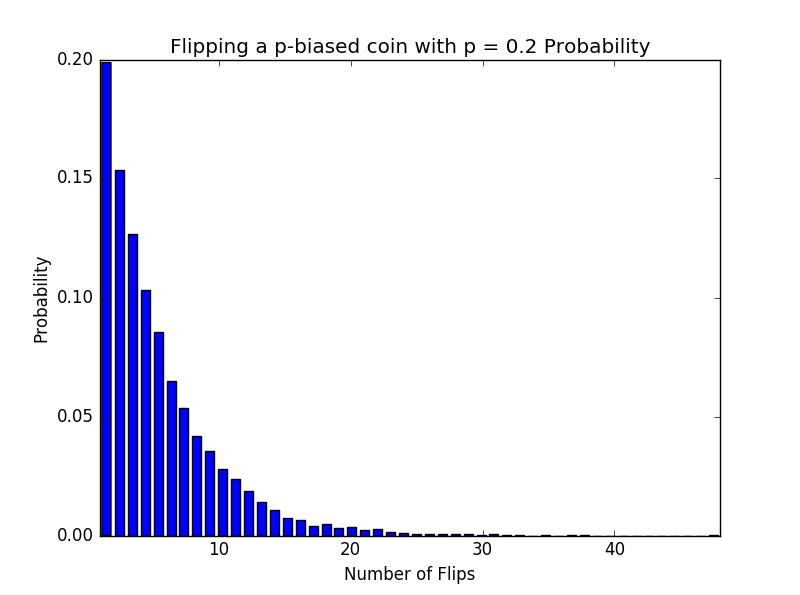
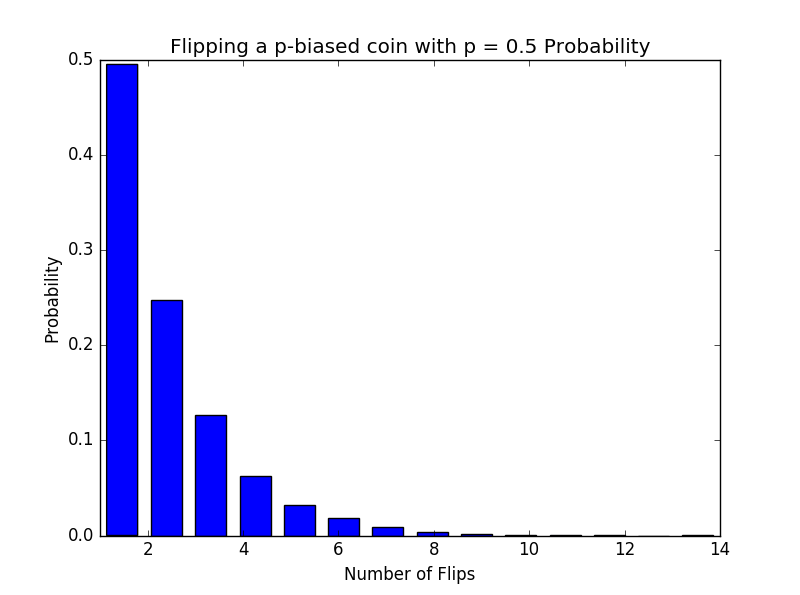
be', estimate)

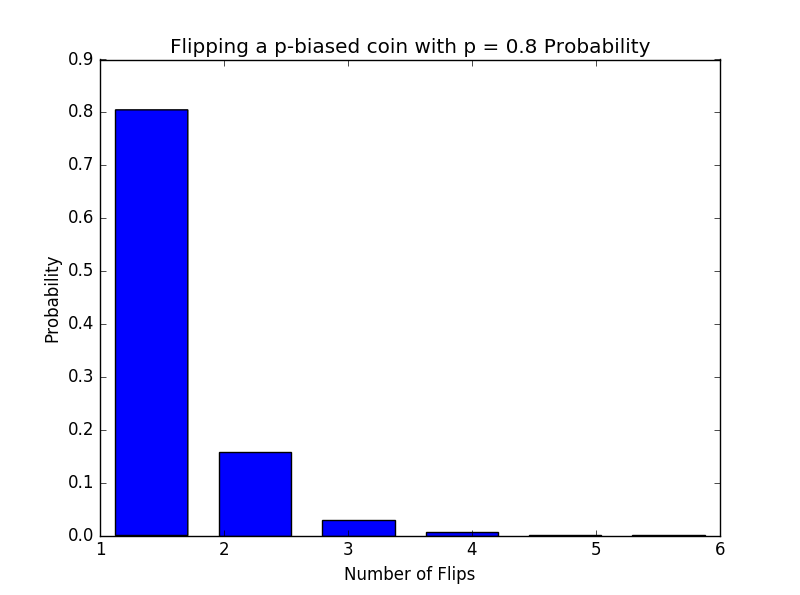
print estimate

pi\_Estimate(100000)

Answer: You need 100,000 points to consistently see 3.14 as the value of pi.

Question 4



Question 5

f(k,p) = p((1-p)^(k-1))

Question 6

import random

from matplotlib import pyplot as plt

import numpy as np

def expt(p):

count = 0

while True:

choice = random.random()

if choice < p:

count+=1

else:

count+=1

break

return count

def Coin\_flip\_test(n,p):

count\_list = []

for x in range(n):

count\_list += [expt(p)]

return count\_list

def plot\_cft\_pmf(n,p):

count\_list = Coin\_flip\_test(n,p)

plt.figure(1)

plt.hist(count\_list, bins = max(count\_list), rwidth = .7, align = 'mid',\

weights = np.zeros\_like(count\_list) + 1. / len(count\_list))

plt.xlim(1,max(count\_list))

plt.title("Flipping a p-biased coin with p = " + str(p) + " Probability")

plt.xlabel("Number of Flips")

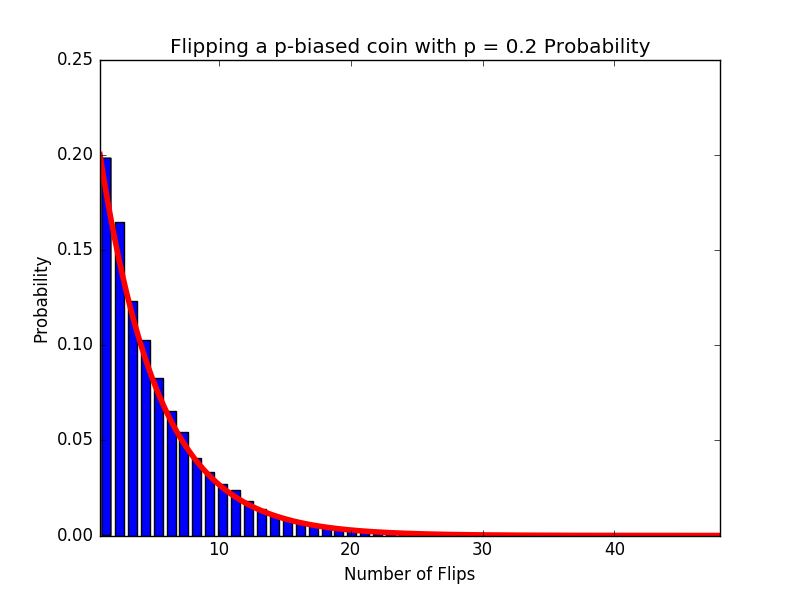
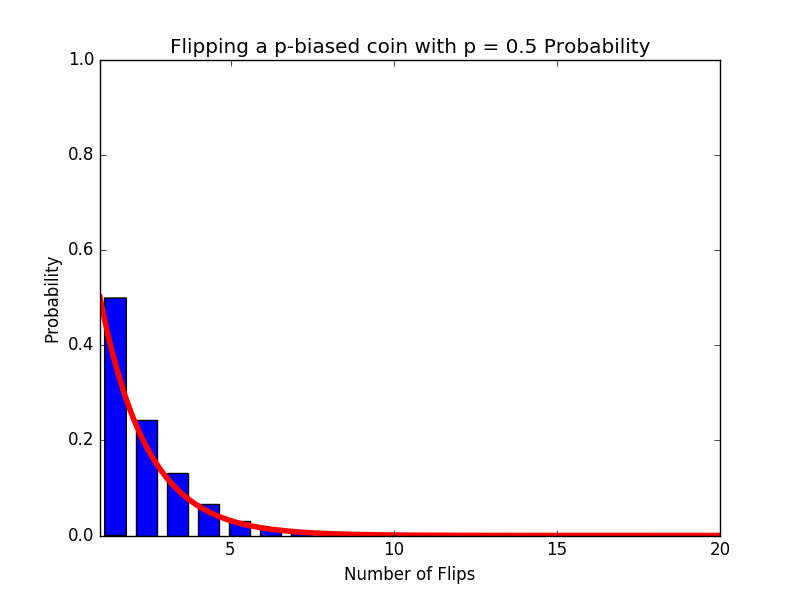
plt.ylabel("Probability")

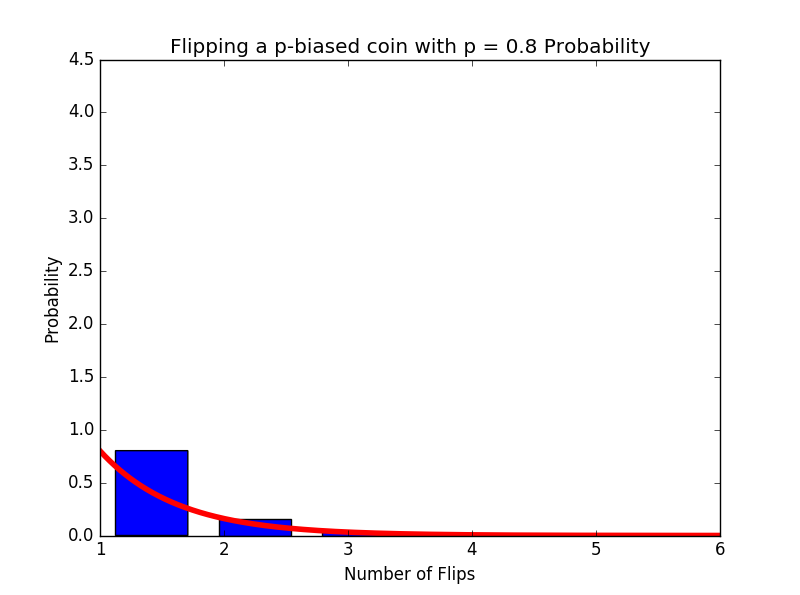
x = np.linspace(0, max(count\_list), 10000)

y = p\*((1-p)\*\*(x-1))

plt.plot(x, y, linestyle = '-', color = 'r')

plt.show()



Part 2 – R

Question 7

> grades <- read.csv('\\Users\\vid82\\OneDrive\\Documents\\CS\\CS237\\grades.csv')

> View(grades)

class1 class2 class3

1 3.3 3.3 2.7

2 2.7 3.7 2.3

3 3.7 3.0 4.0

4 2.3 3.7 2.7

5 4.0 2.3 3.7

6 3.3 3.7 3.3

7 3.3 3.3 2.0

8 3.0 3.0 3.3

9 3.0 2.7 2.3

10 3.7 4.0 3.0

11 4.0 3.7 3.3

12 3.3 3.0 2.3

13 2.7 3.3 3.0

14 2.7 3.7 3.3

15 3.7 3.7 3.7

16 4.0 2.3 3.7

17 3.3 2.7 3.0

18 3.3 3.0 2.3

19 3.0 3.0 2.7

20 3.7 3.3 2.7

21 4.0 3.7 4.0

22 3.3 3.7 2.3

23 3.7 4.0 3.0

24 4.0 3.3 2.0

25 3.7 2.7 4.0

26 4.0 2.0 3.3

27 3.0 2.3 3.0

28 2.3 2.7 3.3

29 2.3 3.0 3.7

30 4.0 3.0 2.0

31 2.7 3.7 2.0

32 3.7 3.0 3.7

33 3.3 3.0 2.3

34 4.0 4.0 2.3

35 3.7 3.0 3.7

36 4.0 2.3 3.3

37 3.3 3.3 2.0

38 3.7 2.7 3.0

39 3.0 3.0 3.3

40 3.7 3.0 3.0

41 3.7 3.0 3.0

42 2.3 3.7 4.0

43 4.0 3.0 2.7

44 4.0 3.3 3.7

45 3.0 3.3 2.7

46 3.0 3.3 3.0

47 4.0 2.3 3.7

48 3.3 2.3 3.3

49 3.0 3.3 4.0

50 3.0 3.7 3.3

51 4.0 4.0 3.7

52 3.7 3.0 3.0

53 3.7 3.3 2.7

54 3.7 3.0 4.0

55 4.0 3.0 2.0

56 2.7 2.3 4.0

57 3.3 3.0 2.0

Among the first ten students:

In class 1, 1 person got a 4.0

In class 2, 1 person got a 4.0

In class 3, 1 person got a 4.0

Question 8

> summary(grades)

class1 class2 class3

Min. :2.3 Min. :2.000 Min. :2.00

1st Qu.:3.0 1st Qu.:3.000 1st Qu.:2.70

Median :3.3 Median :3.000 Median :3.00

Mean :3.4 Mean :3.133 Mean :3.04

3rd Qu.:3.7 3rd Qu.:3.700 3rd Qu.:3.70

Max. :4.0 Max. :4.000 Max. :4.00

The statistics provided are the minimum GPA, the first quartile, the median, the mean, the third quartile, and the maximum GPA.

Question 9

> sd(grades$class1)

[1] 0.5199588

> sd(grades$class2)

[1] 0.5043855

> sd(grades$class3)

[1] 0.6447202

Question 10

> boxplot(grades)

The line inside the box mean.

The line on top of the box is the upper quartile.

The line on the bottom of the box is the lower quartile.

50% of the of the observations are in the box.

Question 11

> table(grades$class1)

2.3 2.7 3 3.3 3.7 4

4 5 9 11 14 14

> table(grades$class2)

2 2.3 2.7 3 3.3 3.7 4

1 7 5 18 11 11 4

> table(grades$class3)

2 2.3 2.7 3 3.3 3.7 4

7 7 7 10 10 9 7

> pmf1 <- table(grades$class1)

> pmf2 <- table(grades$class2)

> pmf3 <- table(grades$class3)

> plot(pmf1/57)

> plot(pmf2/57)

> plot(pmf3/57)