CPSC 4420 Assignment 4 - Jack Carson

1.

a. The outcome space, or the set of all possible outcomes is **S = {A, B, C, D, F}** for taking the class once
b. The size of the event space contains all subsets of the outcome space, including the empty set, this space is defined as: F = {{}, {A}, {B}, {C}, {D}, {F}, {A,B,C,D,F}... and so on. We can define this space as the sum of (N choose m) where N = 5 and m = (size of subsets from 0 to 5). The size is given as (5 choose 5) + (5 choose 4) + (5 choose 3) + (5 choose 2) + (5 choose 1) + (5 choose 0) = 1 + 5 + 10 + 10 + 5 + 1 = **32**

2.

a. P(b) = 2A → **A = 2 / P(b)**
b. Mean, Median, and Mode when b = 5 are…
    i. Mean = **3.5** (The function is symmetric, so the mean is the middle value)
    ii. Median = **3.5** (Again, for symmetrical pdfs, Mean = Median)
    iii. Mode = **2, 5** (2 local maxima)
c. Mean, Median, and Mode when b = 6 are…
    i. If b = 6, and the function is asymmetrical, then the slope from **4 to b** will not be the same as the slope from **2 to 3**. The slope from (**2 to 3 = -1**), and the slope from (**4 to b is ½**)
        1. Mean = ∫ **From -INF to INF xf(x) dx**
            a. μ will be the integral of the product of x and the piecewise function f(x) from -INF to INF (Components are (**y = 4 - x** from 2 to 3), (**y = 1** from 3 to 4), (**y = 1/2x - 1** from 4 to 6)
        2. Median = **~4.236** (Point where area from (-INF to m) = (m to INF)
            a. Def. Integral from **2 to 3** = ∫ **4 - x dx** = 1.5
            b. Def. Integral from **3 to 4** = ∫ **1 dx** = 1
            c. Def. Integral from **4 to 6** = ∫ **1/2x - 1 dx** = 3
            d. ∑ Integrals = 5.5 → 5.5 * .5 = **2.75** → 2.75 - (1.5 - 1) = **.25**
                i. The Upper Bound (Z) of the definite integral from 4 to Z (∫1/2x - 1 dx) which is equal to **.25** is the median of the PDF
                    1. ∫ 1/2x - 1 dx from (**4 to 4.23605**) = .25
        3. Mode = **2, 6** (2 local maxima)

3.

a. $\sum$ **(0 and 1)p(x)** = μ + (1 - μ) = μ - μ + 1 = 1
b. **E[X]** = P(X = 1) * 1 + P(X = 0) * 0 = p * 1 + q * 0 = p
c. **Var[X]** = E[X^2] - E[X]^2 = E[X] - E[X]^2 = p - p^2 = p(1 - p) = μ(1 - μ)
d. d/dx(−μ log μ − (1 − μ) log(1 − μ)) = ln(-μ + 1) - ln(μ) → ½ is the value at which the first derivative **ln(-μ + 1) - ln(μ) = 0**, and therefore is the value which gives the most entropy to the Bernoulli distribution

4.

a. Mean Proof

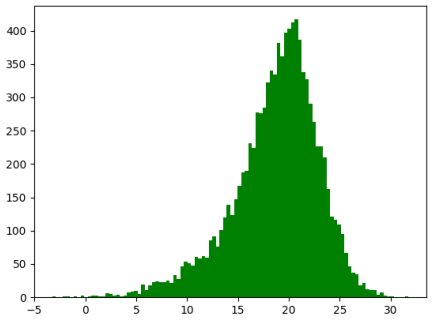
i. E[μ] = $\int$ x * **f(x; a, b)** dx → Def. Integral from 0 to 1 → Substitute f(x; a, b)
ii. = $\int$ x * **$(x^{a-1} * (1 - x)^{b-1} / B(a, b))$** dx → Def. Integral from 0 to 1 → Integrate
iii. = **a / a + b**

b. Variance Proof

i. Var(μ) = E[$(X - μ)^2$ = **$aB / (a + B)^2 * (a + b + 1)$**

5.

```
import matplotlib.pyplot as plt
import numpy as np

pine = np.random.normal(20, 3, 10000)
ponderosa = np.random.normal(15, 5, 10000)

binomial = np.random.binomial(1, 0.75, 10000)
trees = np.where(binomial == 1, pine, ponderosa)

plt.hist(trees, bins=100, color='green')
plt.show()
```



a. The above code and output **give an approximate PDF for the mixture model**
b. Given the distributions of both Pine N(20,3) and Ponderosa N(15,5) trees, and that there are 3x more Pine trees than Ponderosa, we can write the PDF for a randomly chosen tree as: **$f_x(x) = (.75) * N(20, 3) + (.25) * N(15,5)$**
c. Given the scalars (.75) and (.25) representing the probability of selecting each tree, the probability sum is scaled down to 1. For example, selecting 1000 variables from the distribution will yield ~250 variables with characteristics matching ponderosa trees, and ~750 with characteristics matching pine trees. **The def integral of the PDF from -INF to INF = 1**

d. P(x > 16) = Integrate the PDF from 16 to INF, or alternatively, use the law of large numbers, and sample 1000000 trees, adding the below line of code to receive an approximate probability of being over 16 meters. Running this for sufficiently large N gives a probability **~78.6%** of the tree being >16 meters.

```
print("The probability of a tree being over 16 feet tall is", len(trees[trees > 16])/len(trees))
```

e. E[X] = .75 * 20 + .25 * 15 = 15 + 3.75 = **18.75**

i. The expected height of a tree in this mixture model is **18.75 meters**

6.

a. f''(x) = 8 = **Convex**

b. f''(x) = $-1 / \ln(10)x^2$ = **Concave**

c. f''(x) = e^{-x} = **Convex**

d. f''(x) = -sin(x) = **Neither**

e. f''(x) = $\sec^2(x)$ = **Concave** for Interval

f. f''(x) = 2 = **Convex**

g. f''(x) = $-e^{\cos(x)} * \sin(x)$ = **Concave**

h. f''(x) = -cos(x) = **Concave**

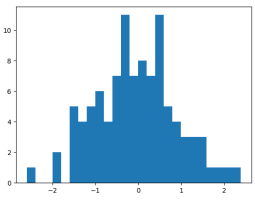
i. f''(x) = 18x = Concave (-INF to 0], Convex [0 to INF) = **Neither**

j. f''(x) = $2(-x^2 + 1) / \ln(10) * (x^2 + 1)^2$ = Concave except [-1, 1] = **Neither**

7. Not Applicable

8.

```
import numpy as np
import matplotlib.pyplot as plt

nums = np.random.normal(0,1,100)

plt.hist(nums, bins=25)
plt.show()
```



a. **Sample code** and output given **above** (25 bins for better visualization)

```
print("Mean: ", np.mean(nums))
print("Variance: ", np.var(nums))
```

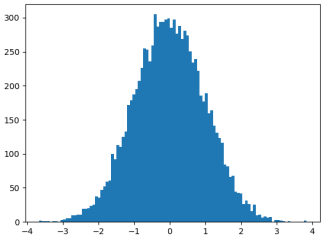
Mean: -0.04111020096294901

Variance: 0.9843968254814078

b. **Sample code** and output given **above**

Mean: -0.005096297472287832

Variance: 0.9797635158127598



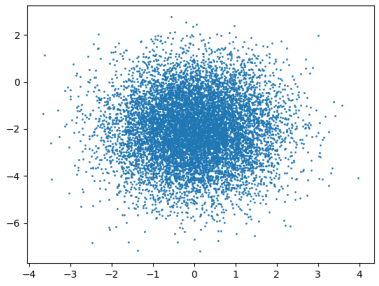
c. Code is reused from above with N = 10000 and bins = 100, with outputs given **above**. With a **larger N, variance and the mean generally converge to the distribution variance and mean**, both following the law of large numbers.

d. The distributions for N = 100, and N = 1000 are given above, and both follow the normal distribution and law of large numbers sufficiently well. **The larger sample size converges to the distribution function, which is expected here.**

9.

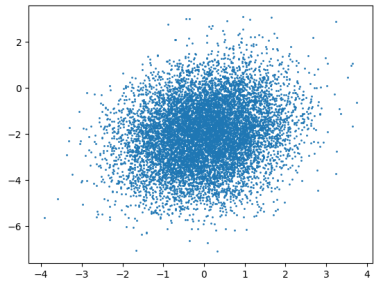
a.

```
import numpy as np
import matplotlib.pyplot as plt

μ = [0, -2]
Σ = [[1, 0], [0, 2]]

vectors = np.random.multivariate_normal(μ, Σ, 10000)

plt.scatter(vectors[:,0], vectors[:,1], s=1)
plt.show()
```



b.

```
import numpy as np
import matplotlib.pyplot as plt

μ = [0, -2]
Σ = [[1, 0.3], [0.3, 2]]

vectors = np.random.multivariate_normal(μ, Σ, 10000)

plt.scatter(vectors[:,0], vectors[:,1], s=1)
plt.show()
```

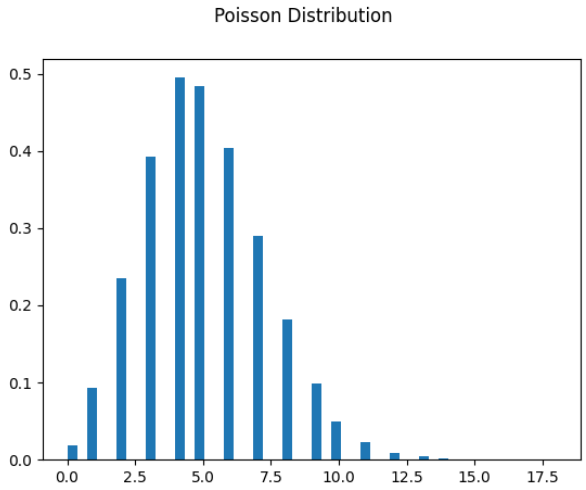


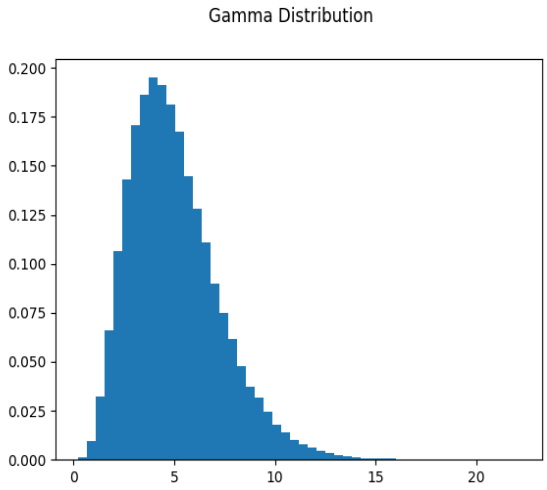
- Given the matrix values of (0.3) in place of the previous weights (0), **it appears that the whole distribution has undergone a linear transformation**, which can be seen by a skew in the data that is only visible at large N.

10.

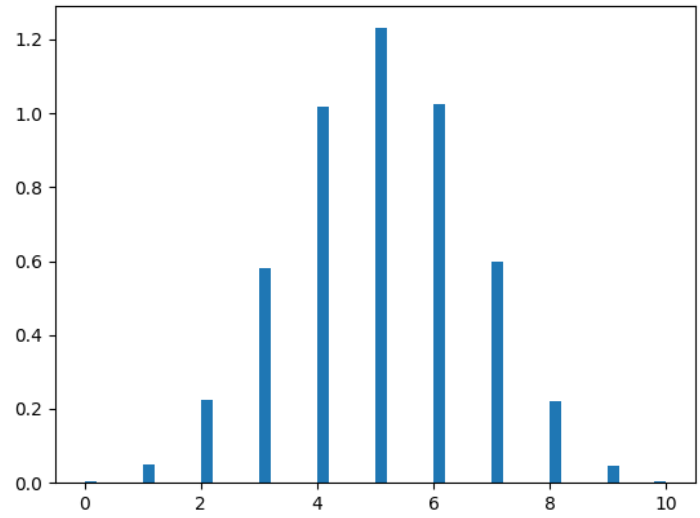
a. Below is code used to generate 5 different probability distributions, and the outputs. Enjoy!

```
import numpy as np
import matplotlib.pyplot as plt

poisson = np.random.poisson(5, 100000) # lambda = 5, 100000 samples
gamma = np.random.gamma(5, 1, 100000) # shape = 5, scale = 1, 100000 samples
binomial = np.random.binomial(10, 0.5, 100000) # n = 10, p = 0.5, 100000 samples
beta = np.random.beta(0.5, 0.5, 100000) # alpha = 0.5, beta = 0.5, 100000 samples
normal = np.random.normal(0, 1, 100000) # mean = 0, standard deviation = 1, 100000 samples

# Define the Histograms
poisson_hist = plt.figure(1)
gamma_hist = plt.figure(2)
binomial_hist = plt.figure(3)
beta_hist = plt.figure(4)
normal_hist = plt.figure(5)

# Name the Histograms
poisson_hist.suptitle('Poisson Distribution')
gamma_hist.suptitle('Gamma Distribution')
binomial_hist.suptitle('Binomial Distribution')
beta_hist.suptitle('Beta Distribution')
normal_hist.suptitle('Normal Distribution')

# Plot the Histograms
poisson_hist.add_subplot(111).hist(poisson, bins = 50, density = True)
gamma_hist.add_subplot(111).hist(gamma, bins = 50, density = True)
binomial_hist.add_subplot(111).hist(binomial, bins = 50, density = True)
beta_hist.add_subplot(111).hist(beta, bins = 50, density = True)
normal_hist.add_subplot(111).hist(normal, bins = 50, density = True)

plt.show()
```

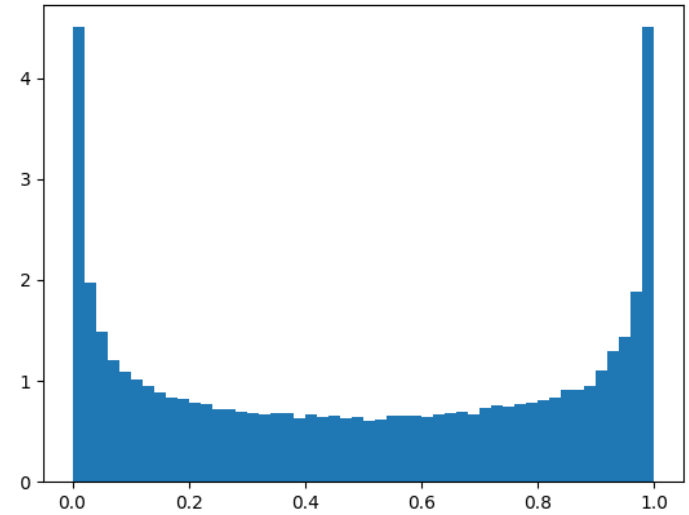




Binomial Distribution



Beta Distribution



Normal Distribution