Homework 1

- Kai-Ju (Carolyn) Yu
- Brian Smith-Eitches

Programming Questions

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from math import sqrt
from scipy import stats
```

1. Create 1000 samples from a Gaussian distribution with mean -10 and standard deviation 5. Create another 1000 samples from another independent Gaussian with mean 10 and standard deviation 5.

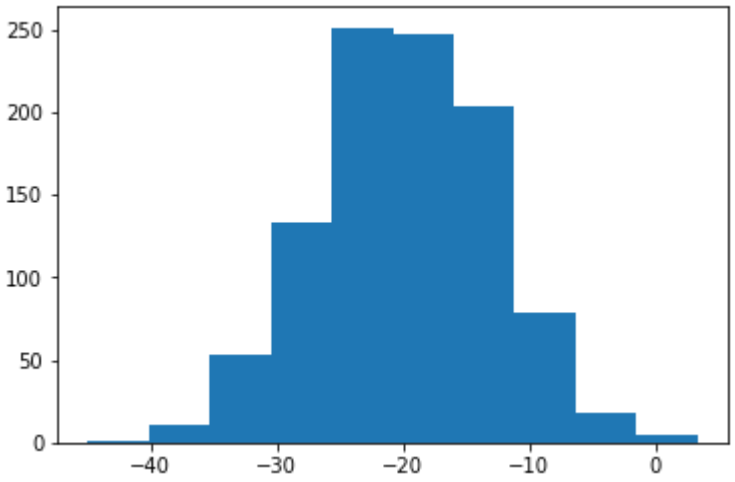
(a) Take the sum of 2 these Gaussians by adding the two sets of 1000 points, point by point, and plot the histogram of the resulting 1000 points. What do you observe?

We observe that there is a relatively normal distribution where the mean appears to be the sum of the two means (-20) and the distribution is wider meaning the standard deviation increased (by a factor of $\sqrt{2}$)

In [2]:
```
mu = -10
sigma = 5
sample1 = np.random.normal(mu, sigma, 1000)
sample2 = np.random.normal(mu, sigma, 1000)
sample_sum = sample1 + sample2
```

In [3]:
```
plt.hist(sample_sum)
```

Out[3]:
```
(array([  1.,  11.,  53., 133., 251., 247., 203.,  79.,  18.,   4.]),
 array([-44.9847842 , -40.15227596, -35.31976773, -30.4872595 ,
        -25.65475127, -20.82224303, -15.9897348 , -11.15722657,
         -6.32471834,  -1.4922101 ,   3.34029813]),
 <a list of 10 Patch objects>)
```



(b) Estimate the mean and the variance of the sum.

In [4]:
```
sample_sum.mean()
```

Out[4]:
```
-19.831074045783456
```

In [5]:
```
sample_sum.var()
```

Out[5]:
```
48.23404480979114
```

In [6]:
```
mean = np.sum(sample_sum) / np.size(sample_sum)
variance = sum([(x - mean) ** 2 for x in sample_sum]) / (len(sample_sum))
print('Mean estimate is\t {}\nVariance estimate is\t  {}'.format(mean, varianc
e))
```

```
Mean estimate is         -19.831074045783456
Variance estimate is      48.23404480979114
```

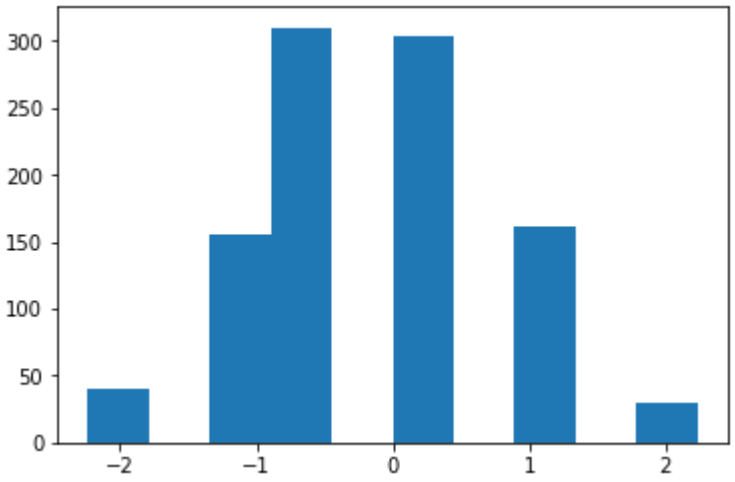
2. Central Limit Theorem Let X_i be an iid Bernoulli random variable with value {-1,1}. Look at the random variable $Z_n = \frac{1}{\sqrt{n}} \sum X_i$

By taking 1000 draws from Z_n, plot its histogram. Check that for small n (say, 5-10) Z_n does not look that much like a Gaussian, but when n is bigger (already by the time n = 30 or 50) it looks much more like a Gaussian. Check also for much bigger n: n = 250, to see that at this point, one can really see the bell curve

In [7]:
```
def getSumXi(n): return sum(np.random.choice((-1,1),n))
def getZn(n): return (1/(sqrt(n)) * getSumXi(n))
def getZnArray(n):
    values = []
    for i in range(1000):
        values.append(getZn(n))
    return np.array(values)
```

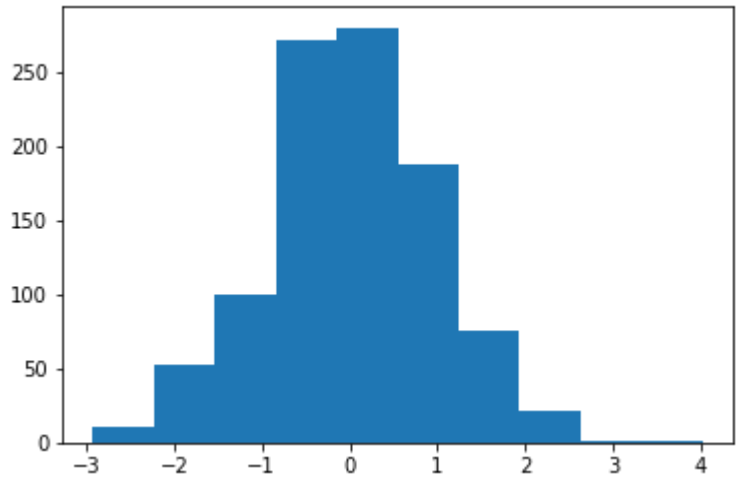
In [8]:
```
values = getZnArray(5)
plt.hist(values)
```

Out[8]:
```
(array([ 40.,   0., 156., 310.,   0., 304.,   0., 161.,   0.,  29.]),
 array([-2.23606798, -1.78885438, -1.34164079, -0.89442719, -0.4472136 ,
         0.        ,  0.4472136 ,  0.89442719,  1.34164079,  1.78885438,
         2.23606798]),
 <a list of 10 Patch objects>)
```



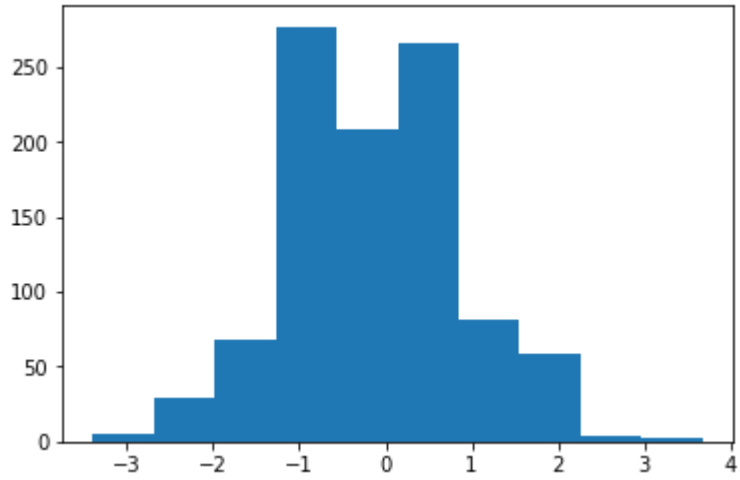
In [9]:
```
values = getZnArray(30)
plt.hist(values)
```

Out[9]:
```
(array([ 11.,  52., 100., 271., 280., 188.,  75.,  21.,   1.,   1.]),
 array([-2.92118697, -2.22740507, -1.53362316, -0.83984125, -0.14605935,
         0.54772256,  1.24150446,  1.93528637,  2.62906828,  3.32285018,
         4.01663209]),
 <a list of 10 Patch objects>)
```



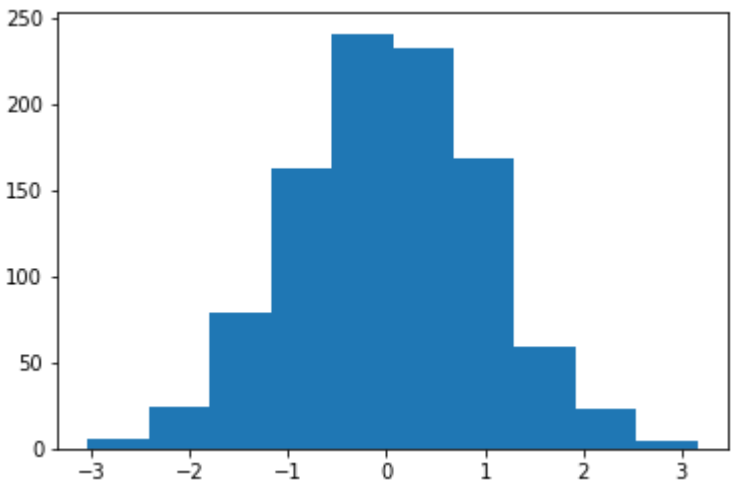
In [10]:
```
values = getZnArray(50)
plt.hist(values)
```

Out[10]:
```
(array([  5.,  29.,  68., 277., 209., 266.,  81.,  58.,   4.,   3.]),
 array([-3.39411255, -2.68700577, -1.97989899, -1.27279221, -0.56568542,
         0.14142136,  0.84852814,  1.55563492,  2.2627417 ,  2.96984848,
         3.67695526]),
 <a list of 10 Patch objects>)
```



In [11]:
```
values = getZnArray(250)
plt.hist(values)
```

Out[11]:
```
(array([  6.,  24.,  79., 163., 241., 232., 169.,  59.,  23.,   4.]),
 array([-3.03578655, -2.41598013, -1.79617371, -1.17636729, -0.55656087,
        0.06324555,  0.68305197,  1.3028584 ,  1.92266482,  2.54247124,
        3.16227766]),
 <a list of 10 Patch objects>)
```



3. Estimate the mean and standard deviation from 1 dimensional data: generate 25,000 samples from a Gaussian distribution with mean 0 and standard deviation 5. Then estimate the mean and standard deviation of this gaussian using elementary numpy commands, i.e., addition,multiplication, division (do not use a command that takes data and returns the mean or standard deviation).

In [12]:
```
mu = 0
sigma = 5
sample = np.random.normal(mu, sigma, 25000)
```

In [13]:
```
mean = np.sum(sample) / np.size(sample)
print(mean)
```

```
-0.010738368815876042
```

In [14]:
```
std_deviation = sqrt(sum([(i - mean) ** 2 for i in sample]) / (len(sample)-1))
print(std_deviation)
```

```
5.014242613345592
```

4. Estimate the mean and covariance matrix for multi-dimensional data: generate 10,000 samples of 2 dimensional data from the Gaussian distribution

$$\begin{pmatrix} X_i \\ Y_i \end{pmatrix} \sim N\left(\begin{pmatrix} -5 \\ 5 \end{pmatrix} \begin{pmatrix} 20 & .8 \\ .8 & 30 \end{pmatrix} \right)$$

Then, estimate the mean and covariance matrix for this multi-dimensional data using elementary numpy commands, i.e., addition, multiplication, division (do not use a command that takes data and returns the mean or standard deviation). (Recall: this notation is a compact way of specifying the distribution of two Gaussian random variables that are correlated. Specifically, the notation indicates that X has mean −5 and variance 20, Y has mean 5 and variance 30, and the covariance of X and Y is 0.8, i.e., $E[(X+5)(Y-5)] = 0.8$.)

In [15]:
```
mean = [-5,5]
cov = [[20,.8],[.8,30]]
matrix = np.random.multivariate_normal(mean, cov, 10000)

n = len(matrix)
```

In [16]:
```
meanMatrix = [x/n for x in np.sum(matrix, axis=0)]
meanMatrix
```

Out[16]:
```
[-5.014323451747807, 5.032672352835161]
```

In [17]:
```
def getCovMatrix(mtx, avmtx):
    variances = []
    for i in range(len(mtx)):
        variance = [(mtx[i][0]-avmtx[0])**2, ##Variance of X
                    (mtx[i][0]-avmtx[0])*(mtx[i][1]-avmtx[1]), ##Covariance of
x and y
                    (mtx[i][1]-avmtx[1])*(mtx[i][0]-avmtx[0]), ##Covariance of
y and x
                    (mtx[i][1]-avmtx[1])**2 ##Variance of Y
                   ]
        variances.append(variance)
    varianceSummation = np.sum(variances, axis=0)
    xx,yx,xy,yy = 0,1,2,3
    newN = n-1
    covMtx = [
                [varianceSummation[xx]/(newN), varianceSummation[yx]/(newN)],
                [varianceSummation[xy]/(newN), varianceSummation[yy]/(newN)]
             ]
    return covMtx
getCovMatrix(matrix,meanMatrix)
```

Out[17]:
```
[[19.751391492160415, 0.7350915980275197],
 [0.7350915980275197, 30.11840288593433]]
```

5. Download from Canvas/Files the dataset PatientData.csv.

Each row is a patient and the last column is the condition that the patient has. Do data exploration using Pandas and other visualization tools to understand what you can about the dataset. For example:

(a) How many patients and how many features are there?

There are 452 patients and 280 features.

In [18]:
```
# Part (a)
data = pd.read_csv("PatientData.csv", header = None)
df = pd.DataFrame(data)
df.shape
```

Out[18]:
```
(452, 280)
```

(b) What is the meaning of the first 4 features? See if you can understand what they mean.

The first feature is age.

The second feature is sex.

The third feature is height (cm).

The fourth feature is weight (kg).

In [19]:
```
df.iloc[:15,:4]
```

Out[19]:

	0	1	2	3
0	75	0	190	80
1	56	1	165	64
2	54	0	172	95
3	55	0	175	94
4	75	0	190	80
5	13	0	169	51
6	40	1	160	52
7	49	1	162	54
8	44	0	168	56
9	50	1	167	67
10	62	0	170	72
11	45	1	165	86
12	54	1	172	58
13	30	0	170	73
14	44	1	160	88

(c) Are there missing values? Replace them with the average of the corresponding feature column

Yes, they are denoted by "?"

In [20]:
```
# Part C
df.replace('?',np.nan, inplace=True)
```

In [21]:
```
# Part C continued
df_missing = df.dropna(how = 'any')
df_mean = df_missing.mean(axis = 0)
df.fillna(df_mean, inplace = True)
```

(d) How could you test which features strongly influence the patient condition and which do not? List what you think are the three most important features

We can test which features have the highest correlation with patient outcome, the last column in the dataframe. The top three most important features are columns 90, 4 and 92.

In [22]:
```
# Part D
corr_matrix = df.corr()
corr_matrix[279].sort_values(ascending=False).head(4)
```

Out[22]:
```
279    1.000000
90     0.368876
4      0.323879
92     0.313982
Name: 279, dtype: float64
```

Written Questions

1. Consider two random variables X,Y that are not independent. Their probabilities of are given by the following table:

(a) What is the probability that X = 1?

P(X=1) = 1/4 + 1/3 = 3/12 + 4/12 = 7/12

(b) What is the probability that X = 1 conditioned on Y = 1?

P(X=1|Y=1) = 1/3 / (1/6 + 1/3) = 1/3 / 1/2 = 2/3

(c) What is the variance of the random variable X?

$Var(X) = E(X^2) - \mu^2$

μ = mean of X = E(X) = 0(1/4 + 1/6) + 1(1/4 + 1/3) = 7/12

$E(X^2)$ = 0^2(1/4 + 1/6) + 1^2 (1/4 + 1/3) = 7/12

$Var(X)$ = 7/12 - (7/12)^2 = 84/144 - 49/144 = 35/144

(d) What is the variance of the random variable X conditioned that Y = 1?

$Var(X|Y = 1) = E(X|Y = 1^2) - \mu^2$

μ = mean of X|Y=1 = E(X|Y=1) = 0(1/3) + 1(2/3) = 2/3

$E(X|Y = 1^2)$ = 0^2(1/3) + 1^2 (2/3) = 2/3

$Var(X)$ = 2/3 - (2/3)^2 = 6/9 - 4/9 = 2/9

(e) What is E[X3+X2+3Y7|Y =1|]?

E[X3+X2+3Y7|Y =1|]

= E(X^3|Y=1) + E(X^2|Y=1) + 3E(Y^7|Y=1)

= 0^3*(1/3)* + *1^3*(2/3) + 0^2*(1/3)* + *1^2*(2/3) + 3*1

= 2/3 + 2/3 + 3 = 13/3

2. Consider the vectors v1 = [1, 1, 1] and v2 = [1, 0, 0]. These two vectors define a 2-dimensional subspace of R_3. Project the points P1 = [3,3,3],P2 = [1,2,3],P3 = [0,0,1] on this subspace. Write down the coordinates of the three projected points. (You can use numpy or a calculator to do arithmetic if you want).

R_3 = span(v_1, v_2)

Formula = $A(A^TA)^{-1}A^TP$

A = $\begin{pmatrix} 1 & 1 \\ 1 & 0 \\ 1 & 0 \end{pmatrix}$

R = $A(A^TA)^{-1}A^T$

R= $\begin{pmatrix} 1 & 1 \\ 1 & 0 \\ 1 & 0 \end{pmatrix} * \left(\begin{pmatrix} 1 & 1 & 1 \\ 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 1 & 0 \\ 1 & 0 \end{pmatrix} \right)^{-1} * \begin{pmatrix} 1 & 1 & 1 \\ 1 & 0 & 0 \end{pmatrix}$

R= $\begin{pmatrix} 1 & 1 \\ 1 & 0 \\ 1 & 0 \end{pmatrix} * \begin{pmatrix} 3 & 1 \\ 1 & 1 \end{pmatrix}^{-1} * \begin{pmatrix} 1 & 1 & 1 \\ 1 & 0 & 0 \end{pmatrix}$

R= $\begin{pmatrix} 1 & 1 \\ 1 & 0 \\ 1 & 0 \end{pmatrix} * \begin{pmatrix} 1/2 & -1/2 \\ -1/2 & 3/2 \end{pmatrix} * \begin{pmatrix} 1 & 1 & 1 \\ 1 & 0 & 0 \end{pmatrix}$

R= $\begin{pmatrix} 0 & 1 \\ 1/2 & -1/2 \\ 1/2 & -1/2 \end{pmatrix} * \begin{pmatrix} 1 & 1 & 1 \\ 1 & 0 & 0 \end{pmatrix}$

R= $\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1/2 & 1/2 \\ 0 & 1/2 & 1/2 \end{pmatrix}$

$p_1|R_3$ = R*P1 = $\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1/2 & 1/2 \\ 0 & 1/2 & 1/2 \end{pmatrix} \begin{pmatrix} 3 \\ 3 \\ 3 \end{pmatrix} = \begin{pmatrix} 3 \\ 3 \\ 3 \end{pmatrix}$

$p_2|R_3$ = R*P2 = $\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1/2 & 1/2 \\ 0 & 1/2 & 1/2 \end{pmatrix} \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} = \begin{pmatrix} 1 \\ 5/2 \\ 5/2 \end{pmatrix}$

$p_3|R_3$ = R*P2 = $\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1/2 & 1/2 \\ 0 & 1/2 & 1/2 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1/2 \\ 1/2 \end{pmatrix}$

3. Consider a coin such that probability of heads is 2/3. Suppose you toss the coin 100 times. Estimate the probability of getting 50 or fewer heads. You can do this in a variety of ways. One way is to use the Central Limit Theorem. Be explicit in your calculations and tell us what tools you are using in these.

Heads(1)	Tails(0)
P = 2/3	P = 1/3

The Bernoulli variable where Heads/success = 1 and Tails/failure = 0 has mean $\mu = 2/3 * 1 + 1/3 * 0 = .667$. This number $(\frac{2}{3})$ represents the proportion of heads we expect to see per coin flip. We therefore expect to see, on average, about 67 Heads per 100 Coin Flips.

We want to know the likelihood of flipping 50 or fewer heads in 100 flips. The next step is to calculate Variance.

Variance is calculated as $variance = n * p * q$ where $p =$ Probability of success/heads and $q =$ Probability of failure/tails or $(1 - p)$ and n is the number of trials $n = 100$. In this case,
$variance = 100 * (\frac{2}{3}) * (\frac{1}{3}) = 100 * \frac{2}{9} = \frac{200}{9}$

We can then take the square root of variance to find the standard deviation.
$\sigma = \sqrt{variance} = \sqrt{\frac{200}{9}} = 4.714$

With $\sigma = 4.714$ and $\mu = .667$, in our sample size $n = 100$, we want to find the corresponding Z Score for an observed average success rate of $X = .5$ or less. Knowing that $Z = \frac{X-\mu}{\sigma}$, we can calculate the Z score is $Z = \frac{.5-.667}{4.714} = -3.535$ for this scenario. Therefore, to determine the likelihood of observing $Pr(\bar{X} \leq .5) = Pr(Z \leq -3.535) =$ `stats.norm.cdf(-3.535)` $= .0002$ meaning the likelihood of flipping 50 heads or fewer in 100 flips is about .02%

```
In [23]: stats.norm.cdf(-3.535)
```

```
Out[23]: 0.0002038875792247969
```