Generate the Gaussian distributed datasets

Assignment requirement:
https://docs.wixstatic.com/ugd/33750e_84919aa1813c45fb89a4250e17edd18f.pdf

1. Define mean vector, we set all the mean=0

```
def zero_mean(dim):
    mean = np.zeros([dim])
    return mean
```

2. Generate the covariance matrix
   - p: unit variance
   - dim: dimensionality

```
def covariance_matrix(p, dim):
    cov = np.zeros([dim, dim])
    for i in range(dim):      #first, generate the upper triangular matri
x
        cnt = 0
        for j in range(i,dim):
            if(i==j):
                    cov[i][j] = 1
              else:
                    cov[i][j] = pow(p,cnt+1)
                    cnt = cnt + 1
     cov = cov + cov.T - np.diag(cov.diagonal()) #upper triangular to sym
metric matrix
    return cov
```

3. Generate gaussian distribution based on the zero mean vector and covariance matrix we created.
   - mean: mean vector
   - cov: covariance matrix
   - sampleNo: #samples
   - dim: dimensionality

```
def gauss_distribution(mean, cov, sampleNo, dim):
      [eigenvalues, eigenvectors] = np.linalg.eig(cov)
      # l = np.matrix(np.diag(np.sqrt(eigenvalues)))
      # print(l)
```

```
        R = cholesky(cov)
        s = np.dot(np.random.randn(sampleNo, dim), R) + mean
        return s
```

4. plot the result of the gaussian distribution

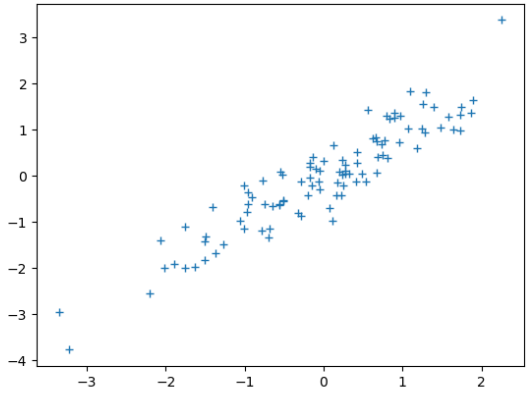
```
def node_plot(s):
    plt.plot()
    plt.plot(s[:,0],s[:,1],'+')
    plt.show()

def histgram(s):
    plt.plot()
    plt.hist(s[:,1], bins='auto')  # arguments are passed to np.histogram
    plt.show()
```

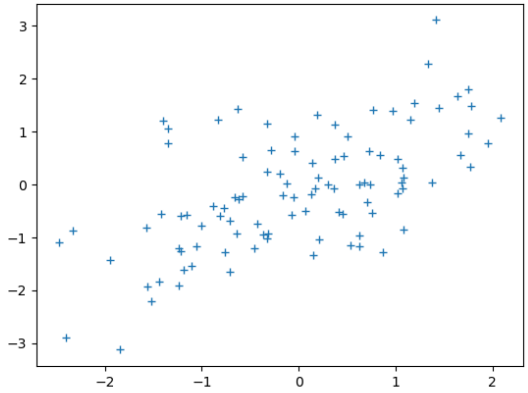
Output:

Two Gaussian distributed classes in which the dimensionality is n = 20 and each class has 100 samples.

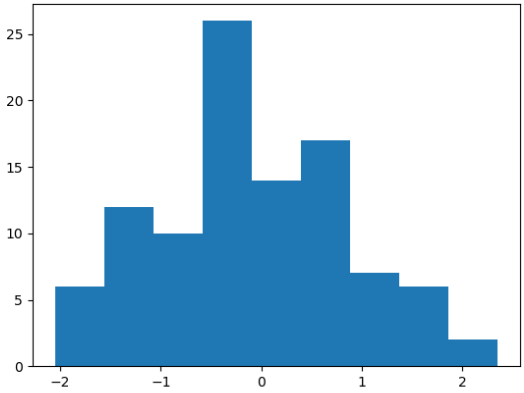
Class 1 : unit variance = 0.9
Class 2 : unit variance = 0.5



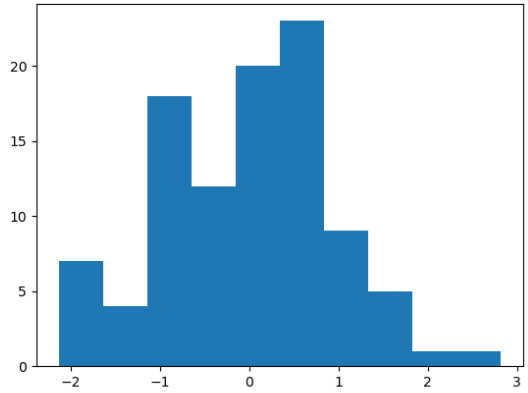
Class 1



Class 2



Class 1



Class 2

The code is on: https://github.com/SnoopyKnight/gaussian_disturbition

Reference:

- https://en.wikipedia.org/wiki/Normal_distribution
- https://docs.scipy.org/
- https://matplotlib.org/