Importing libraries

```
%matplotlib inline
# import matplotlib
import matplotlib.pyplot as plt
```

Importing seaborn and setting the parameters

```
# import seaborn
import seaborn as sns
# settings for seaborn plotting style
sns.set(color_codes=True)
# settings for seaborn plot sizes
sns.set(rc={'figure.figsize':(4.5,3)})
```

```
/usr/local/lib/python3.6/dist-packages/statsmodels/tools/_testing.py:19: FutureWarnir
  import pandas.util.testing as tm
```

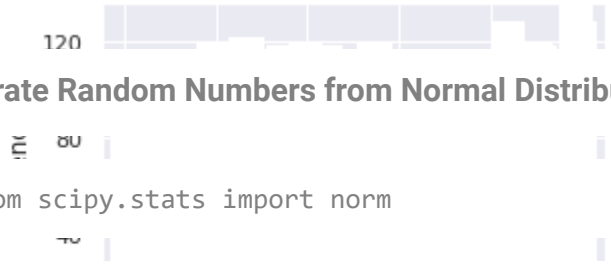
Generating Random Numbers from Uniform Distribution

```
# import uniform distribution
from scipy.stats import uniform
```

```
# random numbers from uniform distribution
# Generate 10 numbers from 0 to 10
n = 10000
a = 0
b = 10
data_uniform = uniform.rvs(size=n, loc = a, scale=b)
```

```
ax = sns.distplot(data_uniform,
                  bins=100,
                  kde=False,
                  color='skyblue',
                  hist_kws={"linewidth": 15,'alpha':1})
ax.set(xlabel='Uniform ', ylabel='Frequency')
```

```
[Text(0, 0.5, 'Frequency'), Text(0.5, 0, 'Uniform ')]
```



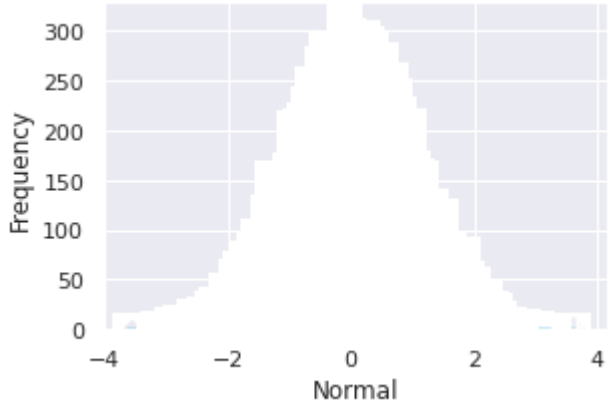
Generate Random Numbers from Normal Distribution

```
from scipy.stats import norm
```

```
# generate random numbersfrom N(0,1)
data_normal = norm.rvs(size=10000,loc=0,scale=1)
```


```
ax = sns.distplot(data_normal,
                  bins=100,
                  kde=False,
                  color='skyblue',
                  hist_kws={"linewidth": 15,'alpha':1})
ax.set(xlabel='Normal', ylabel='Frequency')
```

```
[Text(0, 0.5, 'Frequency'), Text(0.5, 0, 'Normal')]
```

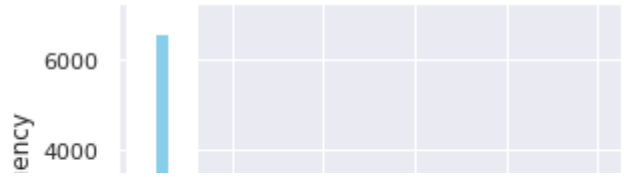


Generate Random Numbers from Bernoulli Distribution

```
# import bernoulli
from scipy.stats import bernoulli
```

```
# generate bernoulli
data_bern = bernoulli.rvs(size=10000,p=0.3)
ax= sns.distplot(data_bern,
                 kde=False,
                 color="skyblue",
                 hist_kws={"linewidth": 15,'alpha':1})
ax.set(xlabel='Bernoulli', ylabel='Frequency')
```

```
[Text(0, 0.5, 'Frequency'), Text(0.5, 0, 'Bernoulli')]
```

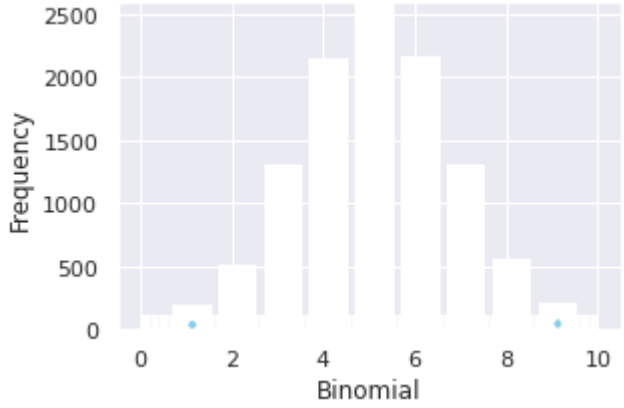


Generate Random Numbers from Binomial Distribution

```
from scipy.stats import binom
```

```
data_binom = binom.rvs(n=10,p=0.5,size=10000)
ax = sns.distplot(data_binom,
                  kde=False,
                  color='skyblue',
                  hist_kws={"linewidth": 15,'alpha':1})
ax.set(xlabel='Binomial', ylabel='Frequency')
```

```
[Text(0, 0.5, 'Frequency'), Text(0.5, 0, 'Binomial')]
```



Generate Random Numbers from Poisson Distribution

```
from scipy.stats import poisson
```

```
#Let us generate 10000 random numbers from Poisson random variable with mu = 0.3 and pl

data_poisson = poisson.rvs(mu=3, size=10000)
ax = sns.distplot(data_poisson,
                  kde=False,
                  color='green',
                  hist_kws={"linewidth": 15,'alpha':1})
ax.set(xlabel='Poisson', ylabel='Frequency')
```

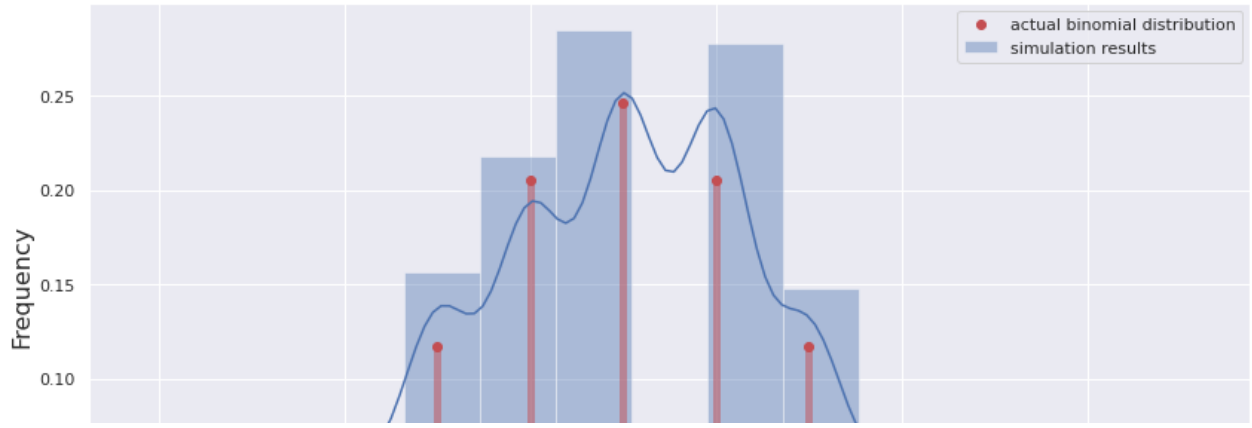
```
[Text(0, 0.5, 'Frequency'), Text(0.5, 0, 'Poisson')]
```


```
# Import libraries
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```
            0     2     4     6     8     10
```

```

# Input variables
# Number of trials
trials = 1000
# Number of independent experiments in each trial
n = 10
# Probability of success for each experiment
p = 0.5
# Function that runs our coin toss trials
# heads is a list of the number of successes from each trial of n experiments
def run_binom(trials, n, p):
    heads = []
    for i in range(trials):
        tosses = [np.random.random() for i in range(n)]
        heads.append(len([i for i in tosses if i>=0.50]))
    return heads
# Run the function
heads = run_binom(trials, n, p)
# Plot the results as a histogram
fig, ax = plt.subplots(figsize=(14,7))
ax = sns.distplot(heads, bins=11, label='simulation results')
ax.set_xlabel("Number of Heads",fontsize=16)
ax.set_ylabel("Frequency",fontsize=16)
# Plot the actual binomial distribution as a sanity check
from scipy.stats import binom
x = range(0,11)
ax.plot(x, binom.pmf(x, n, p), 'ro', label='actual binomial distribution')
ax.vlines(x, 0, binom.pmf(x, n, p), colors='r', lw=5, alpha=0.5)
plt.legend()
plt.show()
```




```
# Probability of getting 6 heads
runs = 10000
prob_6 = sum([1 for i in np.random.binomial(n, p, size=runs) if i==6])/runs
print('The probability of 6 heads is: ' + str(prob_6))
```

```
The probability of 6 heads is: 0.2088
```