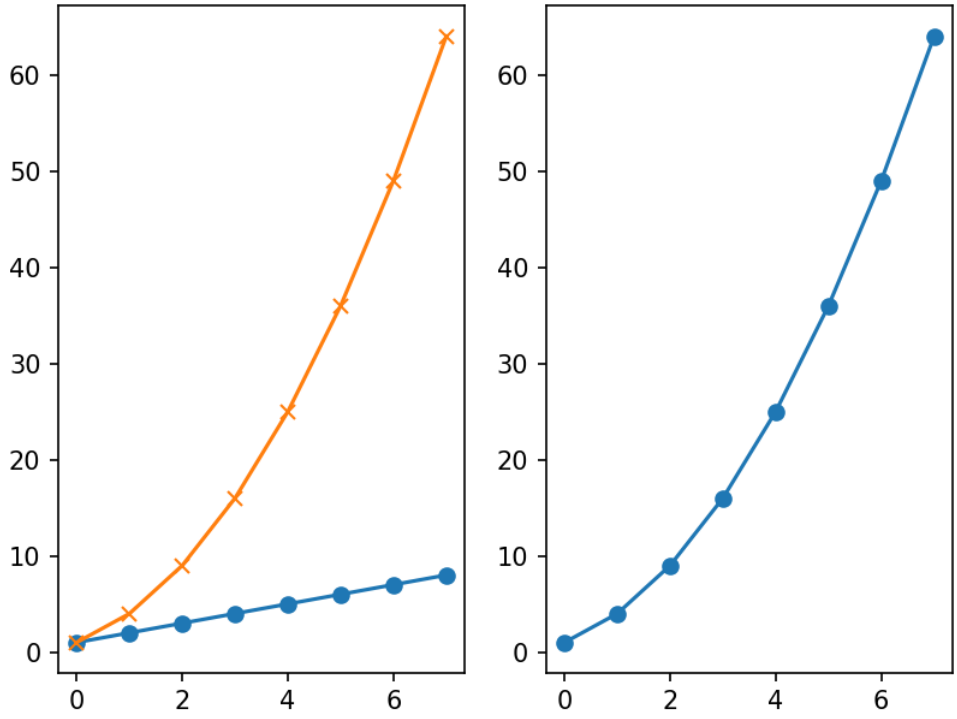
In [1]:

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

In [5]:

```
#SubPlots
plt.figure()
plt.subplot(1,2,1)

linear_data=np.array([1,2,3,4,5,6,7,8])
plt.plot(linear_data,'-o')
```



Out[5]:

```
[<matplotlib.lines.Line2D at 0x17d0bf93508>]
```

In [6]:

```
exponential_data=linear_data**2
plt.subplot(1,2,2)
plt.plot(exponential_data,'-o')
```

Out[6]:

```
[<matplotlib.lines.Line2D at 0x17d0cb146c8>]
```

In [7]:

```
plt.subplot(1,2,1)
plt.plot(exponential_data,'-x')
```

```
C:₩Users₩donghyunkim₩anaconda3₩lib₩site-packages₩ipykernel_launcher.py:1: MatplotlibDeprecationWarning: Adding an axes using the same arguments as a previous axes currently reuses the earlier instance.  In a future version, a new instance will always be created and returned.  Meanwhile, this warning can be suppressed, and the future behavior ensured, by passing a unique label to each axes instance.
  """Entry point for launching an IPython kernel.
```

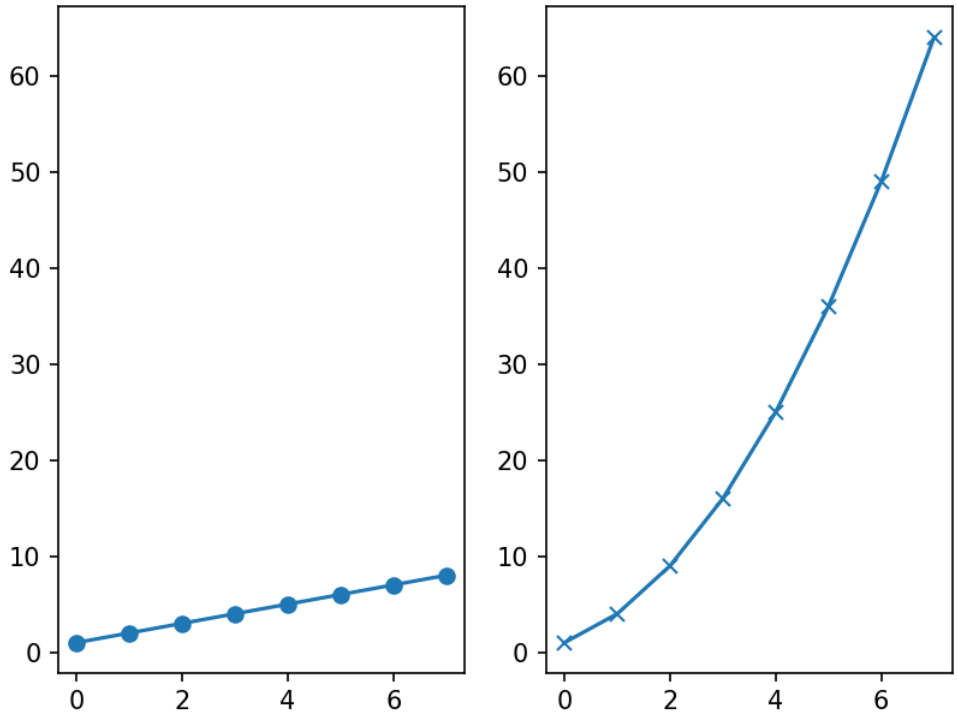
Out[7]:

```
[<matplotlib.lines.Line2D at 0x17d0cb59b48>]
```

In [8]:

```
plt.figure()
ax1=plt.subplot(1,2,1)
plt.plot(linear_data,'-o')

ax2=plt.subplot(1,2,2,sharey=ax1)
plt.plot(exponential_data,'-x')
```

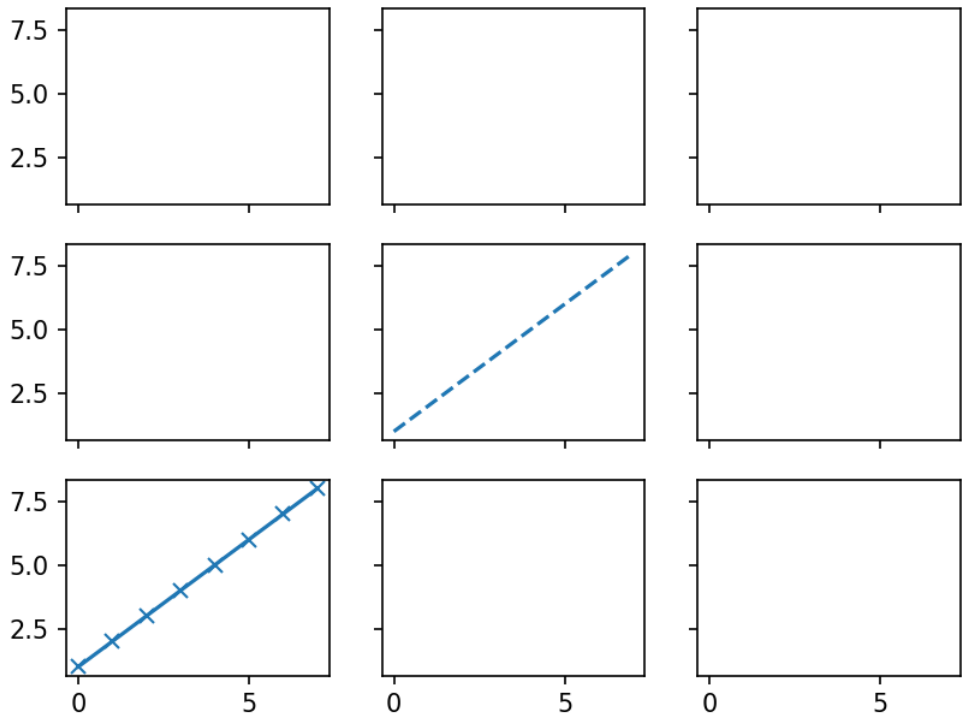


Out[8]:

```
[<matplotlib.lines.Line2D at 0x17d0cb96248>]
```

In [10]:

```
fig, ((ax1,ax2,ax3),(ax4,ax5,ax6),(ax7,ax8,ax9))=plt.subplots(3,3,sharex=True,sharey=True)
```



In [11]:

```
ax5.plot(linear_data,'--')
```

Out[11]:

```
[<matplotlib.lines.Line2D at 0x17d0d9d5548>]
```

In [12]:

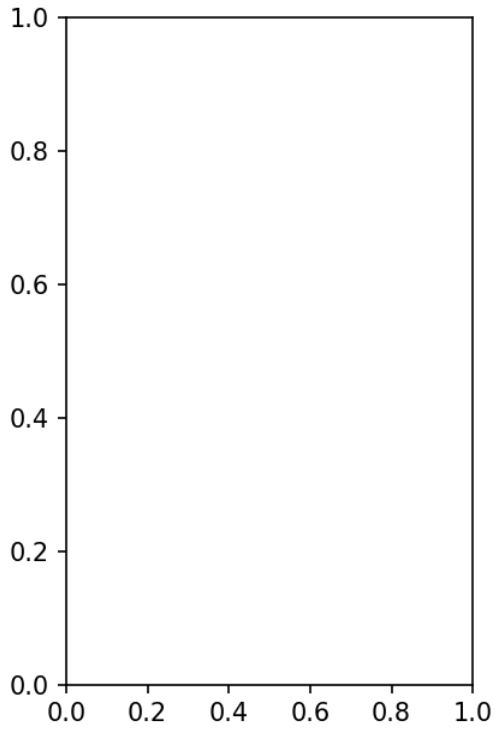
```
ax7.plot(linear_data,'-x')
```

Out[12]:

```
[<matplotlib.lines.Line2D at 0x17d0d9d5608>]
```

In [13]:

```
plt.figure()
plt.subplot(1,2,1)==plt.subplot(121)
```




```
C:₩Users₩donghyunkim₩anaconda3₩lib₩site-packages₩ipykernel_launcher.py:2: MatplotL
ibDeprecationWarning: Adding an axes using the same arguments as a previous axes c
urrently reuses the earlier instance.  In a future version, a new instance will al
ways be created and returned.  Meanwhile, this warning can be suppressed, and the
future behavior ensured, by passing a unique label to each axes instance.
```

Out[13]:

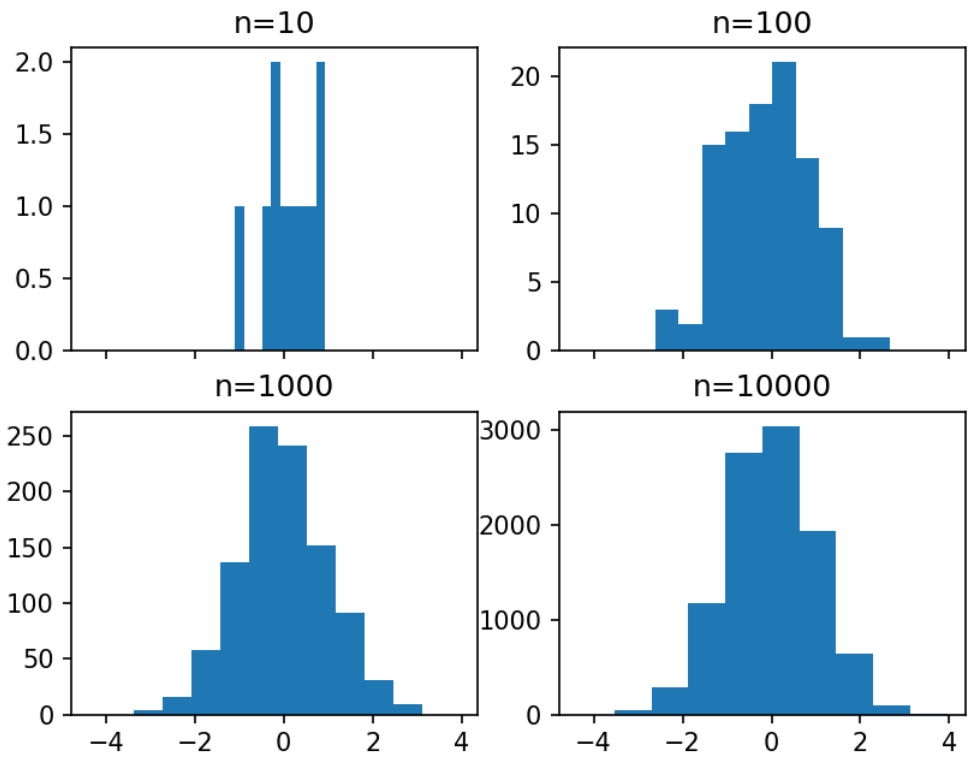
```
True
```

In [ ]:

```
#Histograms
```

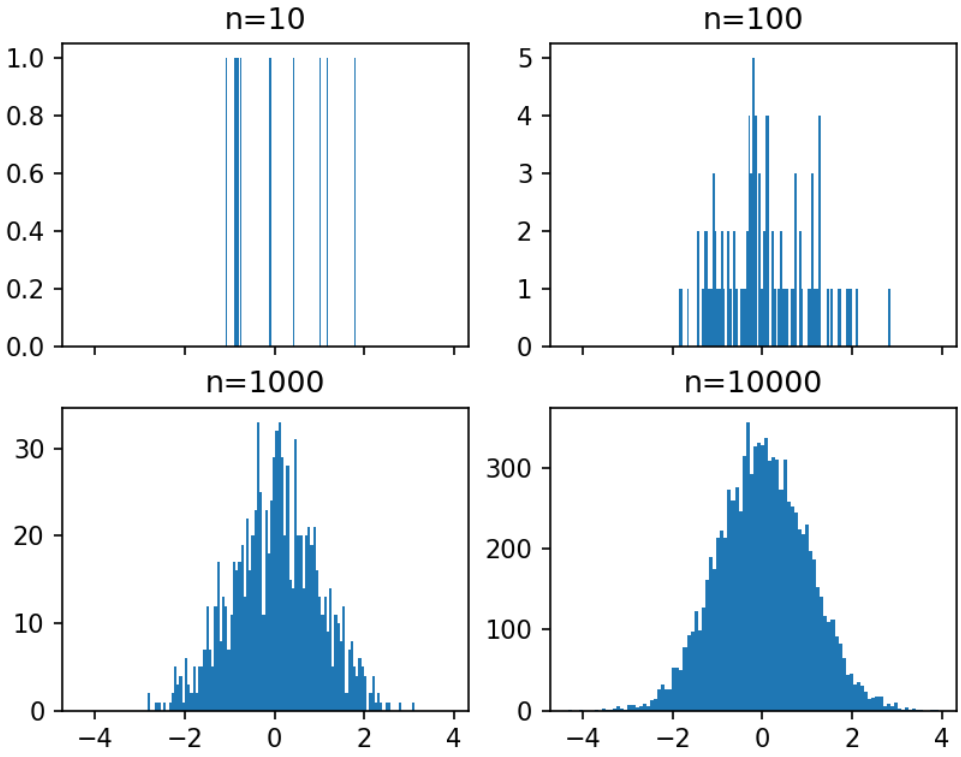
In [14]:

```
fig, ((ax1,ax2),(ax3,ax4))=plt.subplots(2,2,sharex=True)
axs=[ax1,ax2,ax3,ax4]
for n in range(0,len(axs)):
    sample_size=10**(n+1)
    sample=np.random.normal(loc=0.0,scale=1.0,size=sample_size)
    axs[n].hist(sample)
    axs[n].set_title('n={}'.format(sample_size))
```



In [15]:

```
fig, ((ax1,ax2),(ax3,ax4))=plt.subplots(2,2,sharex=True)
axs=[ax1,ax2,ax3,ax4]
for n in range(0,len(axs)):
    sample_size=10**(n+1)
    sample=np.random.normal(loc=0.0,scale=1.0,size=sample_size)
    axs[n].hist(sample,bins=100)#bins는 100개로 쪼갠다(데이터의 밀도를 나타냄)
    axs[n].set_title('n={}'.format(sample_size))
```



In [16]:

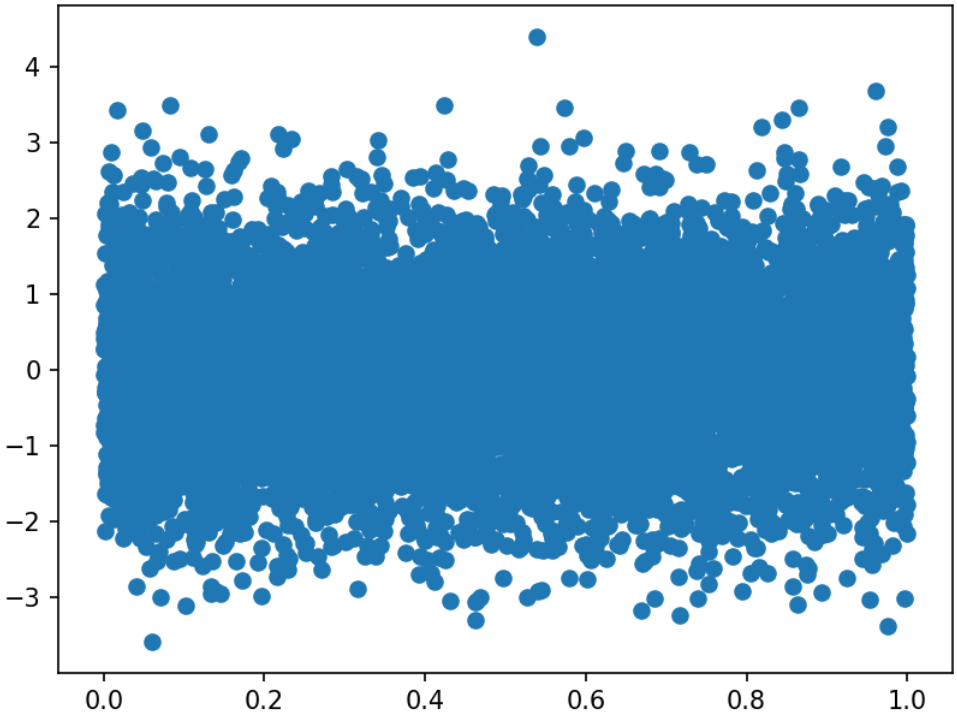
```
sample
```

Out[16]:

```
array([-1.5592241 , -0.18462176, -1.41195328, ..., -1.15630257,
        0.45095411, -1.82453479])
```

In [17]:

```
plt.figure()
Y=np.random.normal(loc=0.0, scale=1.0, size=10000)
X=np.random.random(size=10000)
plt.scatter(X,Y)
```



Out[17]:

```
<matplotlib.collections.PathCollection at 0x17d0e39c548>
```

In [18]:

```
import matplotlib.gridspec as gridspec
```

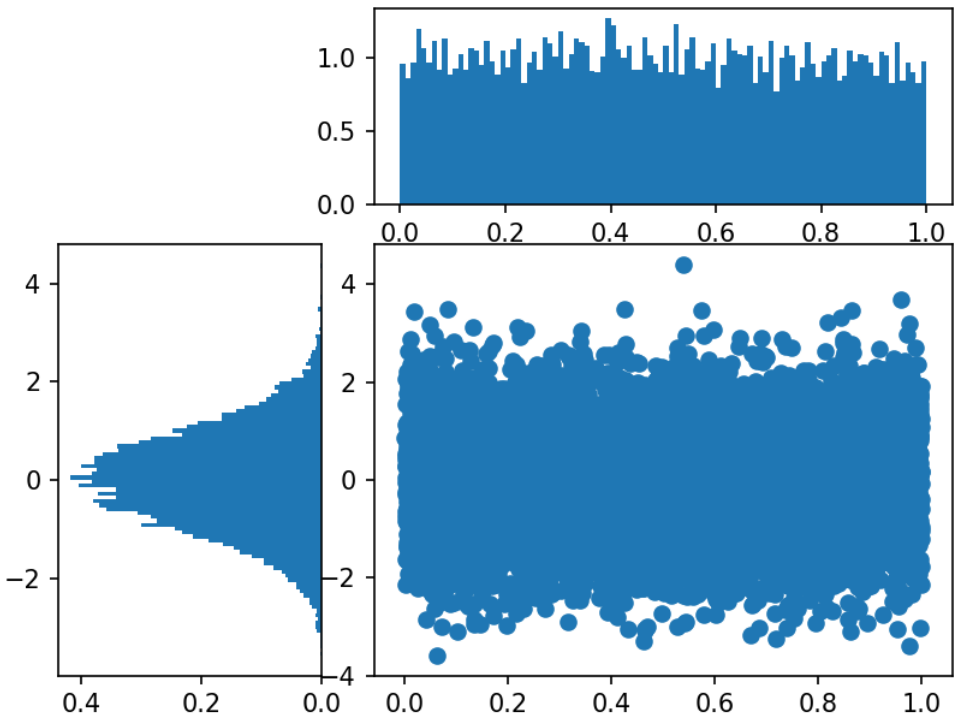
In [19]:

```
plt.figure()
gspec=gridspec.GridSpec(3,3)

top_histogram=plt.subplot(gspec[0,1:])#0~3까지 사용
side_histogram=plt.subplot(gspec[1:,0])
lower_right=plt.subplot(gspec[1:,1:])

lower_right.scatter(X,Y)
top_histogram.hist(X,bins=100)
s=side_histogram.hist(Y,bins=100,orientation='horizontal')

top_histogram.clear()
top_histogram.hist(X,bins=100,density=True)
side_histogram.clear()
side_histogram.hist(Y,bins=100,orientation='horizontal',
                    density=True)
side_histogram.invert_xaxis()
```



In [61]:

In [ ]: