demo2_matplotlib

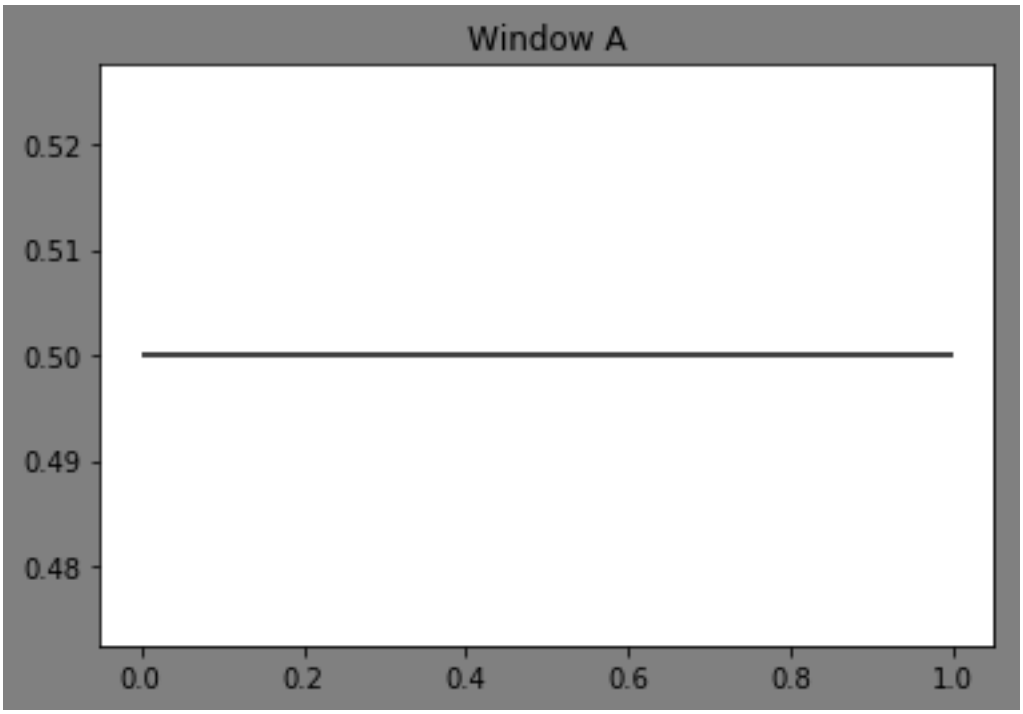
May 15, 2020

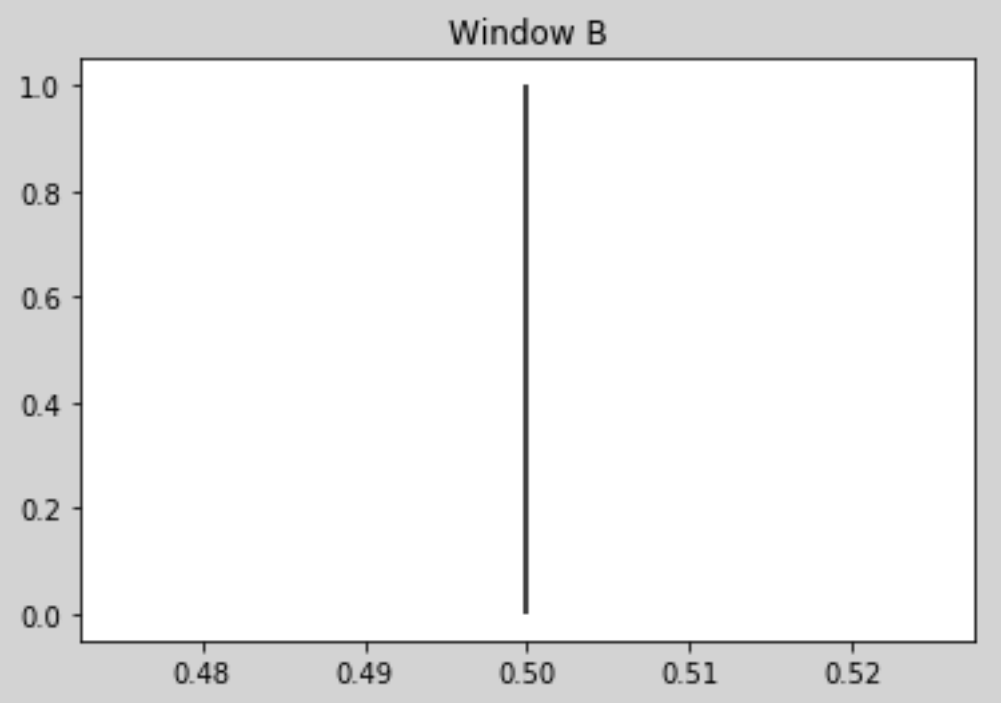
[1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
```

0.1

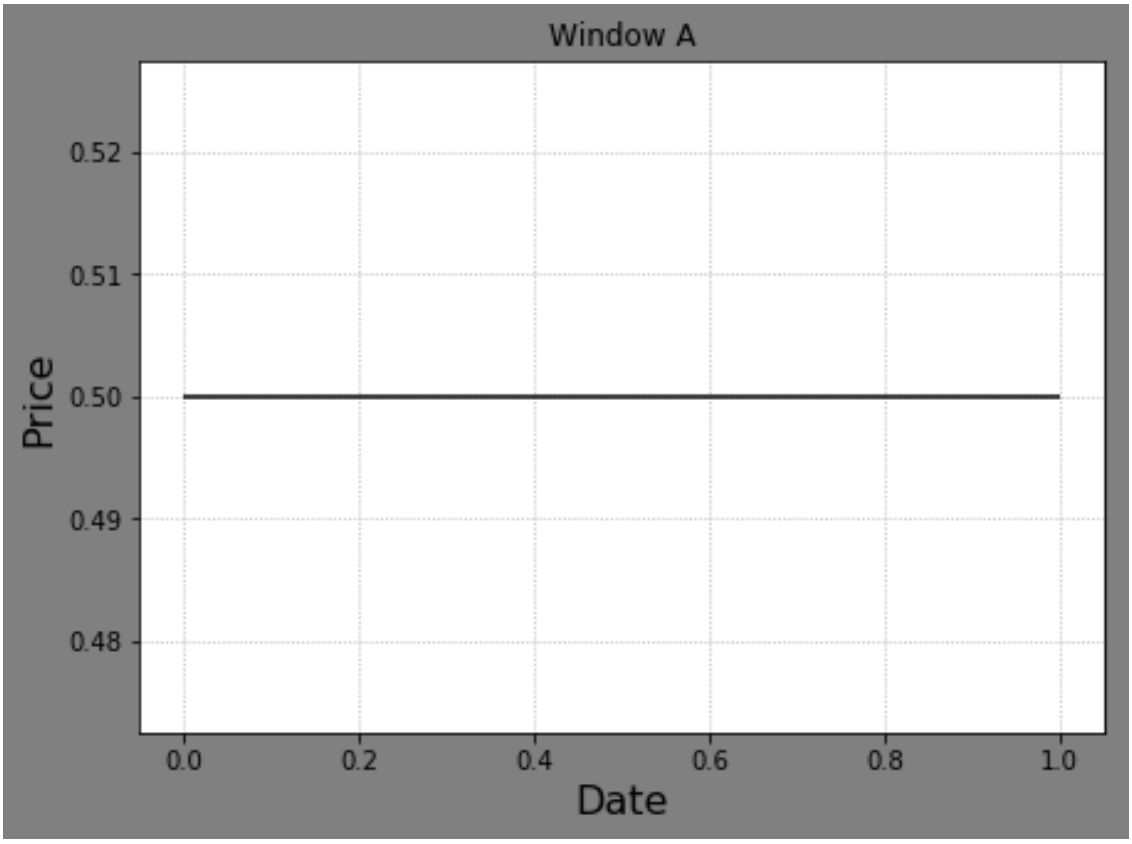
[2]:
```
plt.figure('Window A', facecolor='gray')
plt.title('Window A')
plt.hlines(0.5, 0, 1)
plt.figure('Window B', facecolor='lightgray')
plt.title('Window B')
plt.vlines(0.5, 0, 1)
```

[2]:
```
<matplotlib.collections.LineCollection at 0x264a6f49348>
```



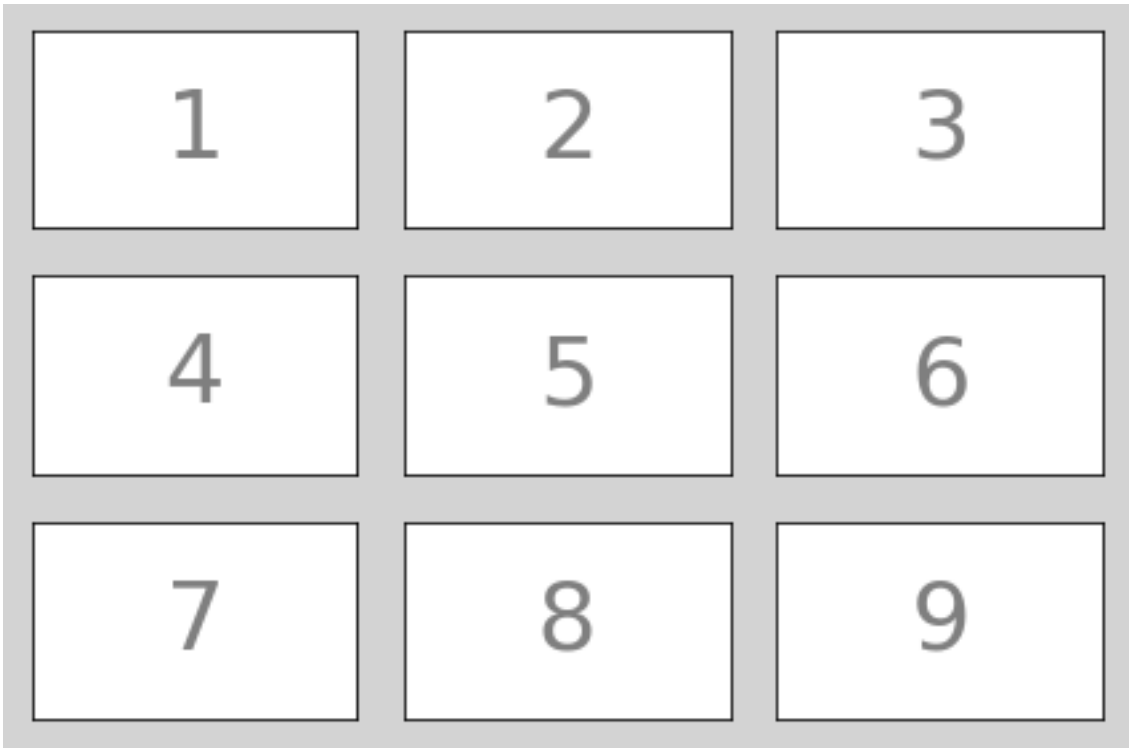


[3]:
```
plt.figure('Window A', facecolor='gray', figsize=((7, 5)))
plt.title('Window A')
plt.hlines(0.5, 0, 1)
plt.xlabel('Date', fontsize=16)
plt.ylabel('Price', fontsize=16)
plt.grid(ls=':')
```



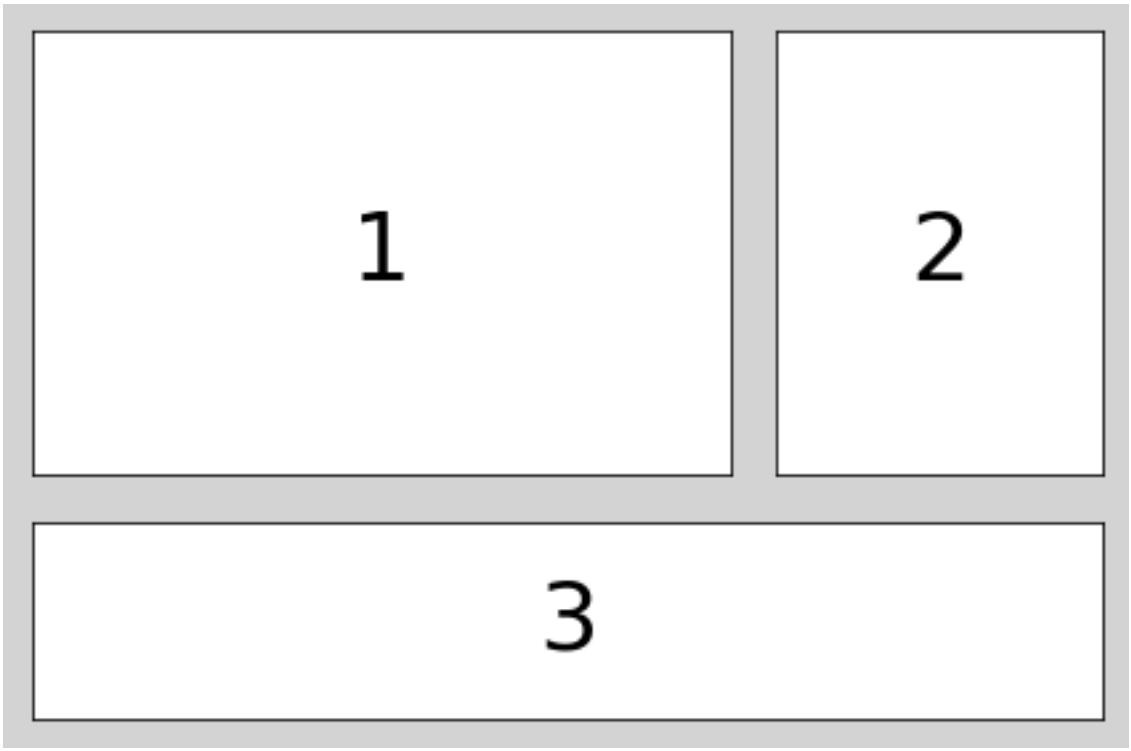
0.2

[4]:
```
plt.figure('Subplot Layout', facecolor='lightgray')
for i in range(1, 10):
    plt.subplot(3, 3, i)
    plt.text(0.5, 0.5, i, ha='center', va='center', size=36, alpha=0.5)
    plt.xticks([])
    plt.yticks([])

plt.tight_layout()
```



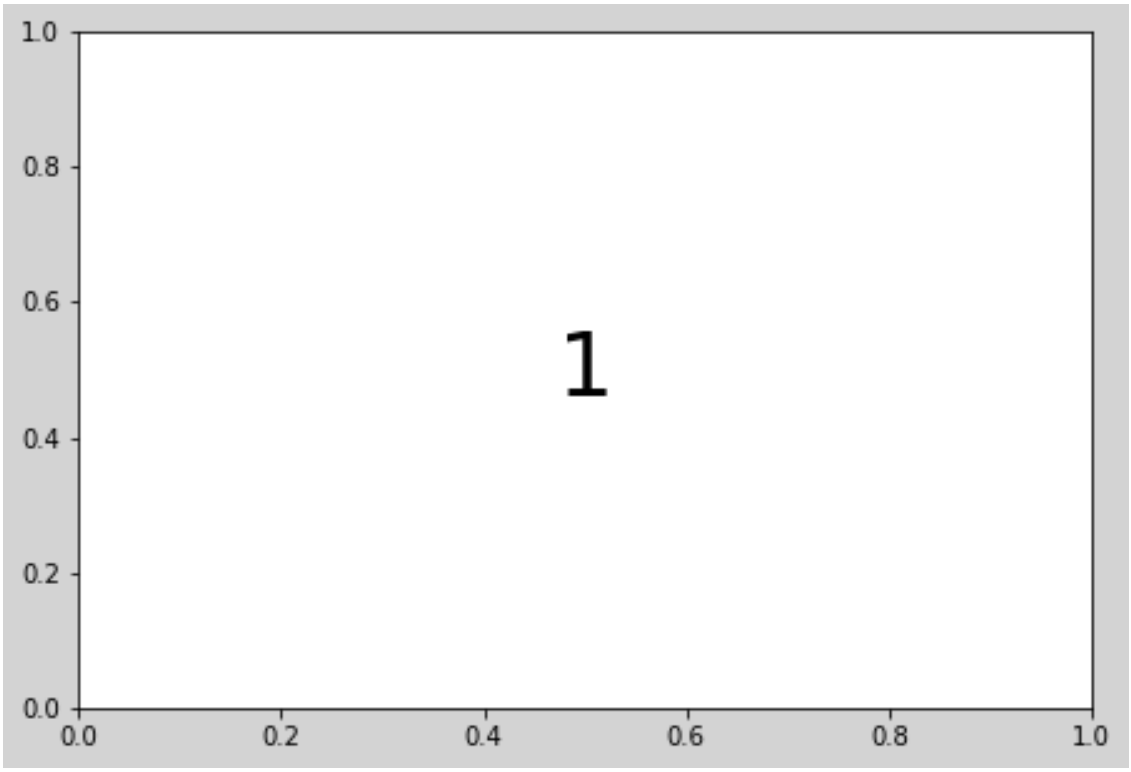
0.3

[5]:
```
import matplotlib.gridspec as mg
plt.figure('Grid Layout', facecolor='lightgray')
gs = mg.GridSpec(3, 3)
plt.subplot(gs[:2, :2])
plt.text(0.5, 0.5, '1', ha='center', va='center', size=36)
plt.xticks([])
plt.yticks([])

plt.subplot(gs[:2, 2])
plt.text(0.5, 0.5, '2', ha='center', va='center', size=36)
plt.xticks([])
plt.yticks([])

plt.subplot(gs[2, :])
plt.text(0.5, 0.5, '3', ha='center', va='center', size=36)
plt.xticks([])
plt.yticks([])

plt.tight_layout()
```



0.4

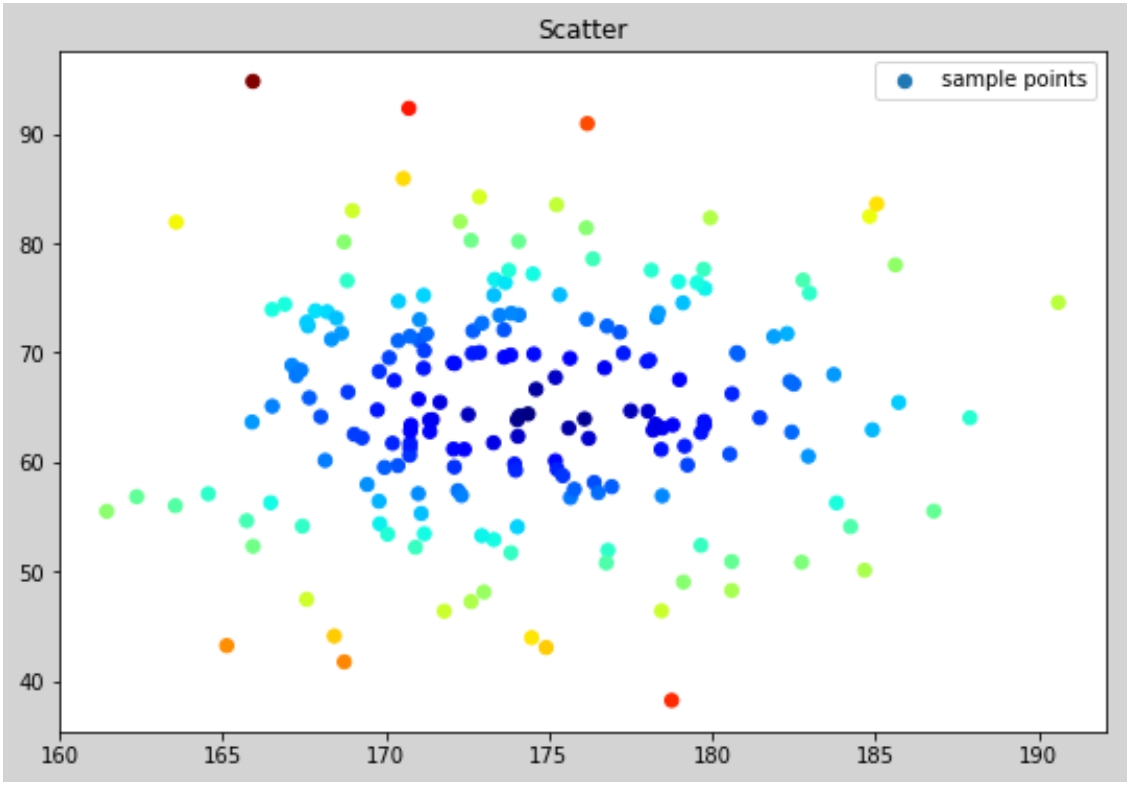
[6]:
```
plt.figure('Flow Layout', facecolor='lightgray')
plt.axes([0.03, 0.03, 0.94, 0.94])
plt.text(0.5, 0.5, '1', ha='center', va='center', size=36)
plt.show()
```



0.5 (Scatterplot)

[7]:
```
n = 200
x = np.random.normal(175, 6, n)
y = np.random.normal(65, 10, n)

plt.figure("scatter", facecolor="lightgray", figsize=(9, 6))
plt.title("Scatter")
d = np.sqrt((x - 175)**2 + (y - 65)**2)
plt.scatter(x, y, marker="o", s=40, label='sample points', c=d, cmap='jet')
plt.legend()
```

[7]:
```
<matplotlib.legend.Legend at 0x264a6a5a508>
```

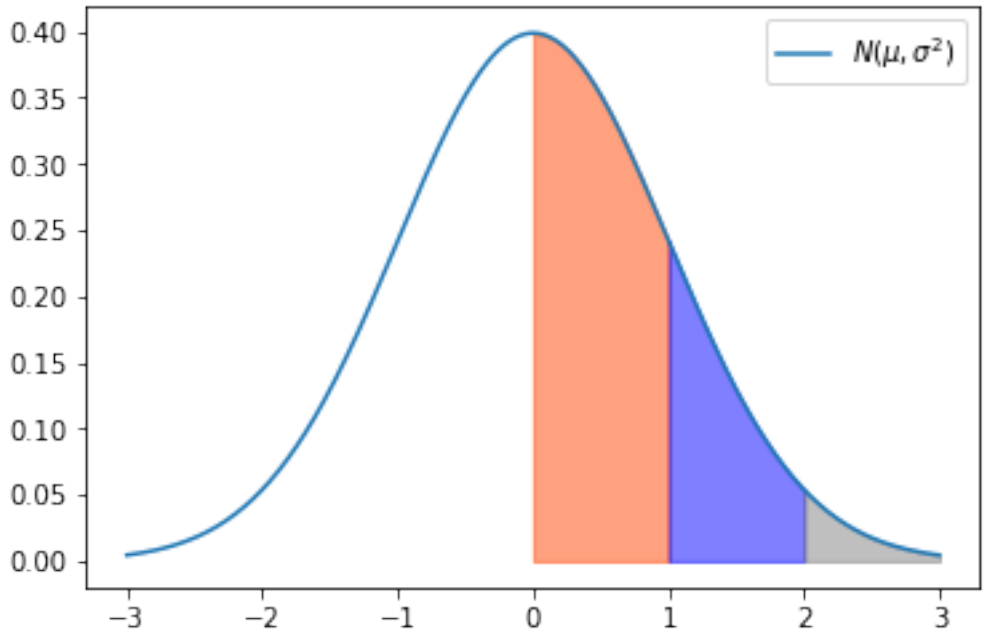


0.6

[8]:
```
def N(x):
    return np.exp(-x**2 / 2) / np.sqrt(2 * np.pi)


x = np.linspace(-3, 3, 1000)
y = N(x)

x2 = np.linspace(0, 1, 200)
y2 = N(x2)
y2_ = np.zeros_like(y2)
plt.fill_between(x2, y2, y2_, y2 > y2_, color='orangered', alpha=0.5)
x3 = np.linspace(1, 2, 200)
y3 = N(x3)
y3_ = np.zeros_like(y3)
plt.fill_between(x3, y3, y3_, y3 > y3_, color='blue', alpha=0.5)

x4 = np.linspace(2, 3, 200)
y4 = N(x4)
y4_ = np.zeros_like(y4)
plt.fill_between(x4, y4_, y4, y4 > y4_, color='gray', alpha=0.5)
```

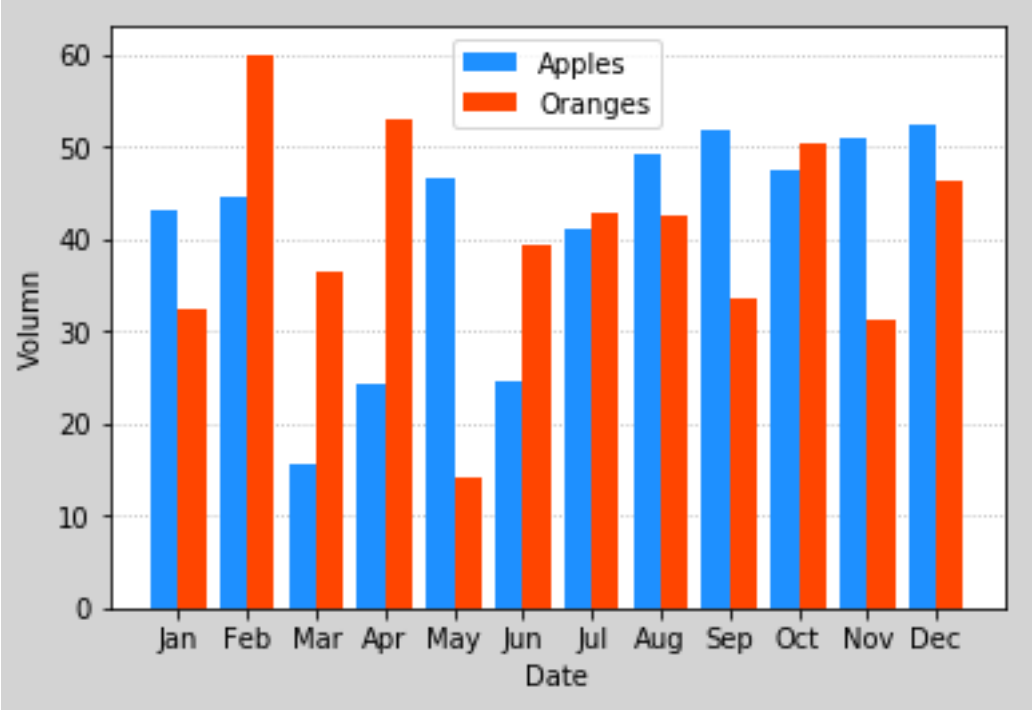
```
plt.plot(x, y, label=r'$N(\mu, \sigma^2)$')
plt.legend()
```

[8]: <matplotlib.legend.Legend at 0x264a7405048>



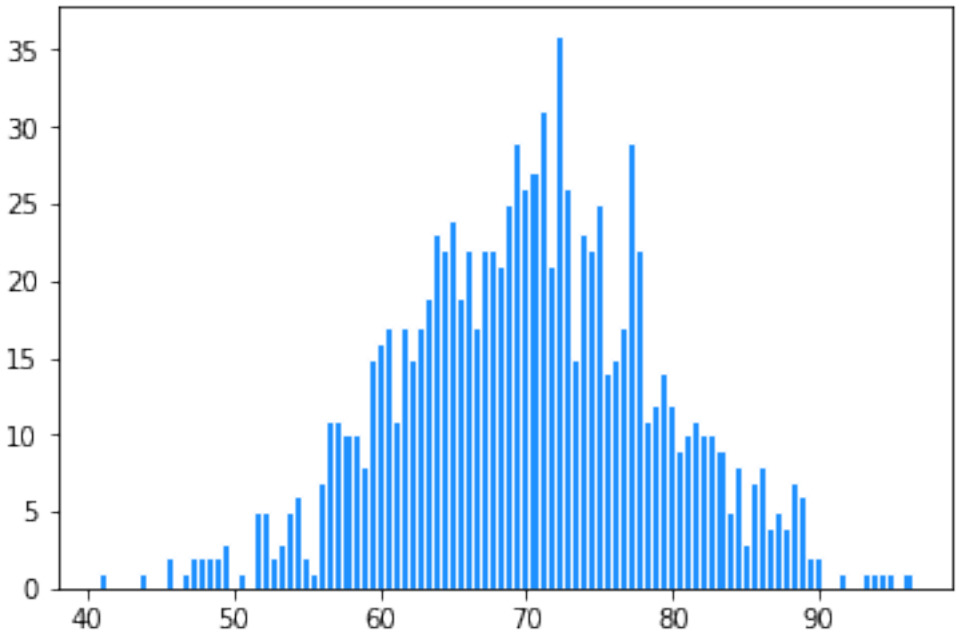
0.7 (Bar Chart)

[9]:
```
apples = np.random.normal(40, 10, 12)
x = np.arange(apples.size)
plt.figure('Bar Chart', facecolor='lightgray')
plt.grid(ls=':', axis='y')
plt.xlabel('Date')
plt.ylabel('Volumn')
plt.bar(x - 0.2, apples, 0.4, color='dodgerblue', label='Apples', zorder=3)

oranges = np.random.normal(40, 10, 12)
plt.bar(x + 0.2, oranges, 0.4, color='orangered', label='Oranges', zorder=3)

plt.xticks(x, [
    'Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun', 'Jul', 'Aug', 'Sep', 'Oct',
    'Nov', 'Dec'
])
plt.legend()
plt.show()
```



0.8 (Histogram)

[24]:
```
x = np.random.normal(70, 9, 1000)
plt.hist(x, bins=100, color='dodgerblue', edgecolor='white')
plt.show()
```



0.9 , ,

[30]:
```
a = np.random.binomial(10, 0.6, 1000000)
for i in range(11):
    print(i, ':', (a == i).sum() / 1000000)
```

```
0 : 0.000103
1 : 0.001537
2 : 0.010685
3 : 0.042442
4 : 0.111557
5 : 0.200312
6 : 0.250567
7 : 0.215434
8 : 0.120737
9 : 0.040521
10 : 0.006105
```

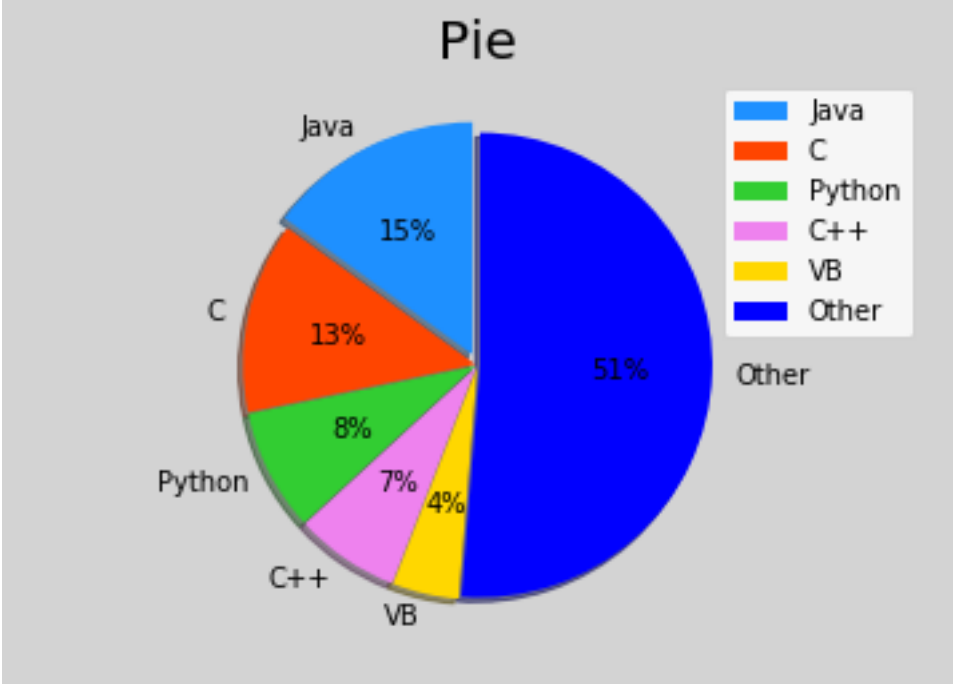
[42]:
```
r = np.random.hypergeometric(4, 6, 3, 1000000)
for i in range(4):
    print(i, ':', (r == i).sum() / 1000000)
```

```
0 : 0.166362
1 : 0.500298
2 : 0.30003
3 : 0.03331
```

0.10 (Pie Chart)

[43]:
```
plt.figure('pie', facecolor='lightgray')
plt.title('Pie', fontsize=20)
#
values = [15, 13.3, 8.5, 7.3, 4.62, 51.28]
spaces = [0.05, 0.01, 0.01, 0.01, 0.01, 0.01]
labels = ['Java', 'C', 'Python', 'C++', 'VB', 'Other']
colors = ['dodgerblue', 'orangered', 'limegreen', 'violet', 'gold', 'blue']
#
plt.axis('equal')
plt.pie(
    values,  #
    spaces,  #
    labels,  #
    colors,  #
    '%d%%',  #
    shadow=True,  #
```

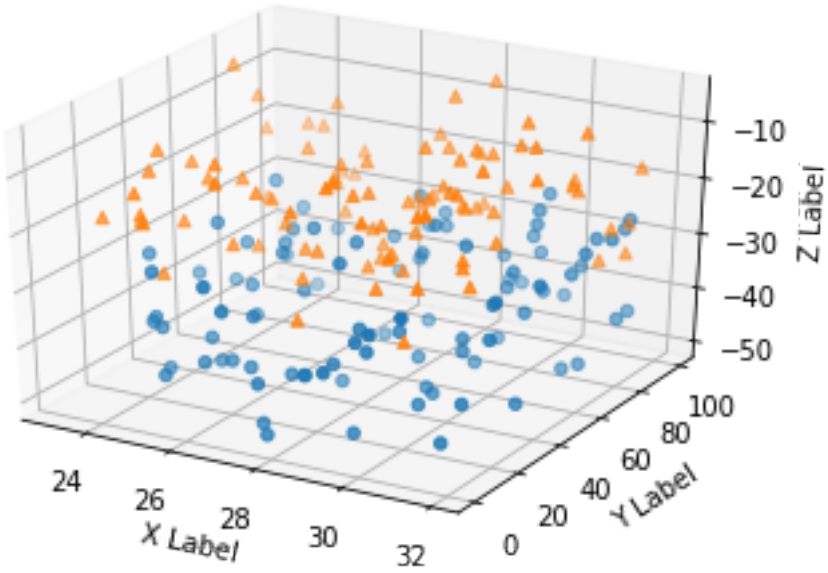
```
    startangle=90,  #
    radius=1  #
)
plt.legend()
plt.show()
```



0.11 3D

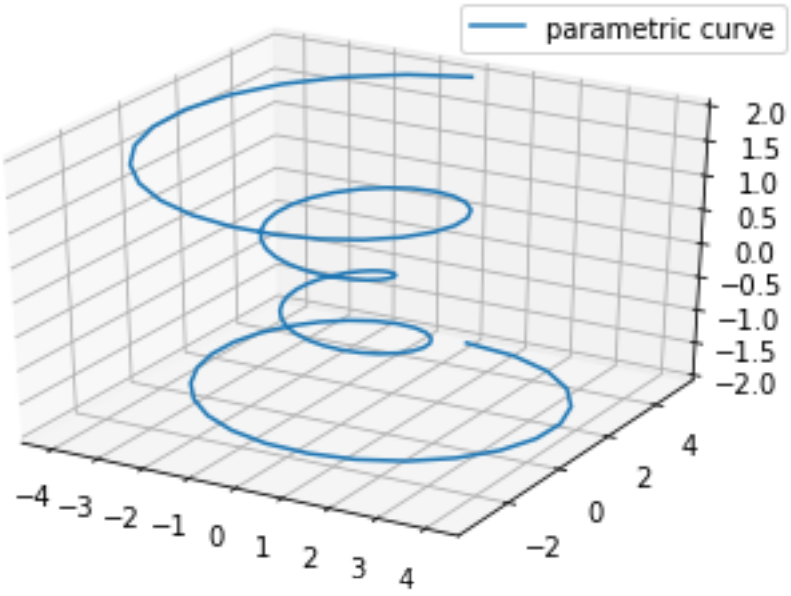
[45]:
```
import matplotlib.pyplot as plt
import numpy as np

# Fixing random state for reproducibility
np.random.seed(19680801)


def randrange(n, vmin, vmax):
    '''
    Helper function to make an array of random numbers having shape (n, )
    with each number distributed Uniform(vmin, vmax).
    '''
    return (vmax - vmin) * np.random.rand(n) + vmin


fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
```

```
n = 100

# For each set of style and range settings, plot n random points in the box
# defined by x in [23, 32], y in [0, 100], z in [zlow, zhigh].
for m, zlow, zhigh in [('o', -50, -25), ('^', -30, -5)]:
    xs = randrange(n, 23, 32)
    ys = randrange(n, 0, 100)
    zs = randrange(n, zlow, zhigh)
    ax.scatter(xs, ys, zs, marker=m)

ax.set_xlabel('X Label')
ax.set_ylabel('Y Label')
ax.set_zlabel('Z Label')

plt.show()
```



[46]:
```
import numpy as np
import matplotlib.pyplot as plt

plt.rcParams['legend.fontsize'] = 10

fig = plt.figure()
ax = fig.gca(projection='3d')

# Prepare arrays x, y, z
```

```
theta = np.linspace(-4 * np.pi, 4 * np.pi, 100)
z = np.linspace(-2, 2, 100)
r = z**2 + 1
x = r * np.sin(theta)
y = r * np.cos(theta)

ax.plot(x, y, z, label='parametric curve')
ax.legend()

plt.show()
```



[ ]: