1- Scatter Plot- Dağılım Grafiği:

In [1]:

```
#############################################################################################
#                                                                                           #
#                        DIKKAT                                                             #
#                                                                                           #
# Aşağıdaki kodlar ve açklama yazıları Jupyter Notebook ile hazırlanmıştır.                 #
#                                                                                           #
# Özellikle 3D görüntüleme başka kodlama  arayüzlerinde hata verebilir.                     #
#                                                                                           #
#                                                                                           #
#############################################################################################
```

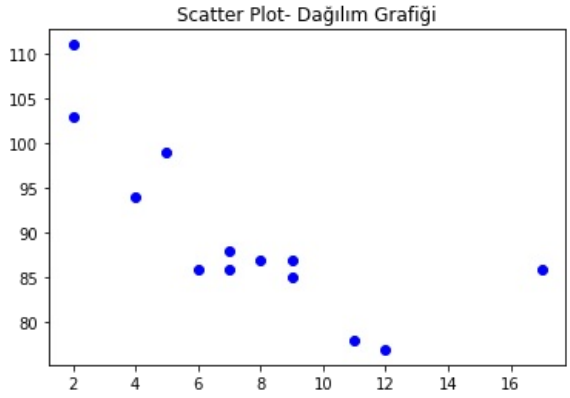
Tanım:

Dağılım grafiği iki sayısal değişken arasındaki ilişkiyi gösterir. Her bir veri noktasında X ve Y değerleri mevcuttur. Böylece iki boyutlu bir grafik oluşturur.

In [ ]:

In [1]:

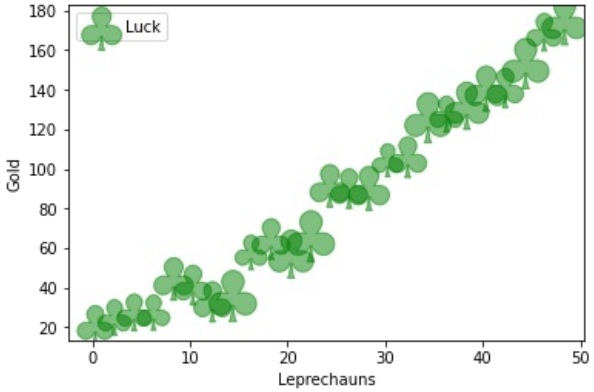
```
import matplotlib.pyplot as plt

x = [5,7,8,7,2,17,2,9,4,11,12,9,6]
y = [99,86,87,88,111,86,103,87,94,78,77,85,86]

a =[17]
b= [86]


plt.title("Scatter Plot- Dağılım Grafiği")

plt.scatter(x, y,marker='o',color='b')

# filled_markers = ('o', 'v', '^', '<', '>', '8', 's', 'p', '*', 'h', 'H', 'D', 'd', 'P', 'X')

#plt.scatter(a,b, marker='o', color='r')

plt.show()
```



In [ ]:

In [2]:

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

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


x = np.arange(0.0, 50.0, 2.0)
y = x ** 1.3 + np.random.rand(*x.shape) * 30.0
s = np.random.rand(*x.shape) * 800 + 500

plt.scatter(x, y, s, c="g", alpha=0.5, marker=r'$\clubsuit$',
            label="Luck")
plt.xlabel("Leprechauns")
plt.ylabel("Gold")
plt.legend(loc='upper left')
plt.show()
```



Ne için kullanlır:

Scatter plot- Dağılım grafiği iki değişken arasındaki bağlantıyı göstermek için kullanılır. Örneğin yıllık gelir ile çocuk sayısı arasındaki ilişki gibi ki Hans Rossling ile ilgili bir iki video izlediyseniz veya yazı okuduysanız mutlaka buna benzer bir iki grafik görmüşsünüzdür.

Farklı data setleri için Lineer bağlantı, kare ilişki(square relationship), sinüsoidal bağlantı veya hiçbir bağı olmayan grafiklere internette sıklıkla karşılaşırsınız.

In [ ]:

```

```

Çeşitleri:

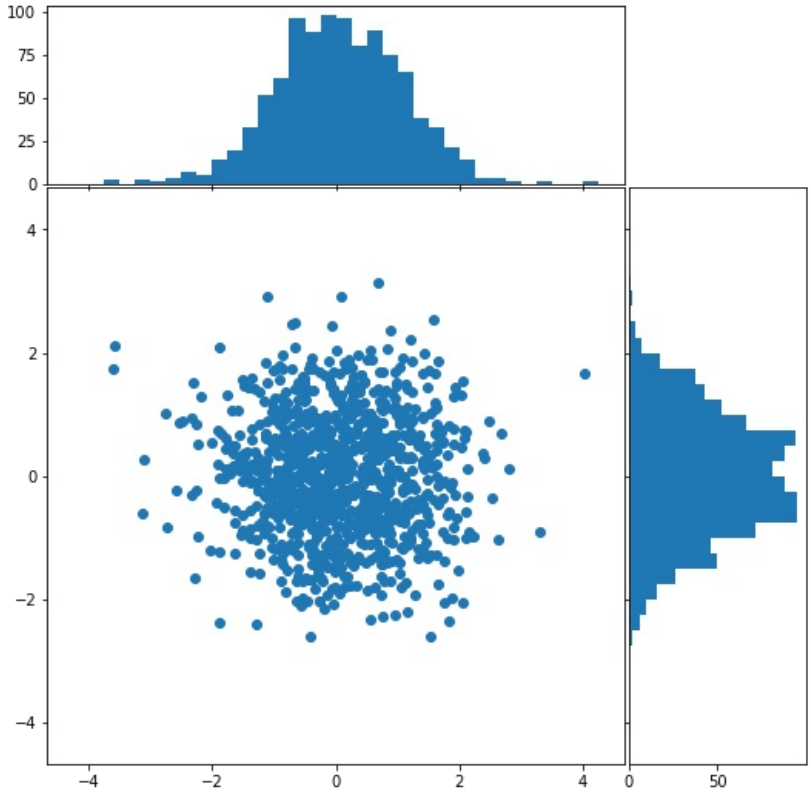
Etkileşimli dağılım grafiği (Interactive Scatter Plot) dağılım grafiğinin bir üst çeşididir. Grafiğin daha belirli kısımlarına zoom yapmanıza izin verir . Ayrıca her bir noktanın üzerine mause ile gittiğinizde size etiket bilgilerini verir. Bunun için internette pekçok python ggplot, R veya başka yazılım dilleri yapılmış örnek kod bulabilirisniz ancak excelde interactivite çok kısıtlıdır.

Interactivite haricinde dağılım grafiklerinin eksenlerine bar - çubuk grafikleri ile grafik detaylandırması yapabilirsiniz. Bu durumda Scatter With Marginal Point- Marjinal Noktalı Dağılım grafiğini çizmiş olursunuz.

Bu grafik türünün görsel benzeri ancak bir üst yapısı Cluster - Kümeleme grafiğidir ki o grafik türünün benzerliği sadece görseldir, zira amacına uygun biçimde o grafik türünde kümeleme amcıyla dağlımdaki her bir nokta farklı renklendirilmiştir. Kümeleme grafiği ileride detaylıca anlatılacaktır.

In [3]:

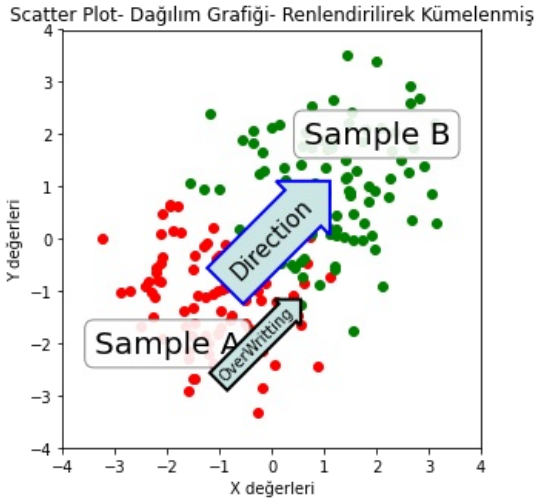
```
# Scatter plot with Marginal histograms


import numpy as np
import matplotlib.pyplot as plt

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

# some random data
x = np.random.randn(1000)
y = np.random.randn(1000)


def scatter_hist(x, y, ax, ax_histx, ax_histy):
    # no labels
    ax_histx.tick_params(axis="x", labelbottom=False)
    ax_histy.tick_params(axis="y", labelleft=False)

    # the scatter plot:
    ax.scatter(x, y)

    # now determine nice limits by hand:
    binwidth = 0.25
    xymax = max(np.max(np.abs(x)), np.max(np.abs(y)))
    lim = (int(xymax/binwidth) + 1) * binwidth

    bins = np.arange(-lim, lim + binwidth, binwidth)
    ax_histx.hist(x, bins=bins)
    ax_histy.hist(y, bins=bins, orientation='horizontal')


# definitions for the axes
left, width = 0.1, 0.65
bottom, height = 0.1, 0.65
spacing = 0.005


rect_scatter = [left, bottom, width, height]
rect_histx = [left, bottom + height + spacing, width, 0.2]
rect_histy = [left + width + spacing, bottom, 0.2, height]

# start with a square Figure
fig = plt.figure(figsize=(8, 8))

ax = fig.add_axes(rect_scatter)
ax_histx = fig.add_axes(rect_histx, sharex=ax)
ax_histy = fig.add_axes(rect_histy, sharey=ax)

# use the previously defined function
scatter_hist(x, y, ax, ax_histx, ax_histy)

plt.show()
```



In [ ]:

In [4]:

```
# Bu grafikte hem renklendirilerek kümeleme hem de anotation yani not ekleme yapılmıştır.

import numpy as np
import matplotlib.pyplot as plt


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

fig, ax = plt.subplots(figsize=(5, 5))
ax.set_aspect(1)

x1 = -1 + np.random.randn(100)
y1 = -1 + np.random.randn(100)
x2 = 1. + np.random.randn(100)
y2 = 1. + np.random.randn(100)

ax.scatter(x1, y1, color="r")
ax.scatter(x2, y2, color="g")

bbox_props = dict(boxstyle="round", fc="w", ec="0.5", alpha=0.9)
ax.text(-2, -2, "Sample A", ha="center", va="center", size=20,
        bbox=bbox_props)
ax.text(2, 2, "Sample B", ha="center", va="center", size=20,
        bbox=bbox_props)


bbox_props = dict(boxstyle="rarrow", fc=(0.8, 0.9, 0.9), ec="b", lw=2)
t = ax.text(0, 0, "Direction", ha="center", va="center", rotation=45,
            size=15,
            bbox=bbox_props)


bb = t.get_bbox_patch()
bb.set_boxstyle("rarrow", pad=0.6)

ax.set_xlim(-4, 4)
ax.set_ylim(-4, 4)

############# Özel Ekleme ############################

plt.title("Scatter Plot- Dağılım Grafiği- Renlendirilirek Kümelenmiş")
ax.set_xlabel("X değerleri")
ax.set_ylabel("Y değerleri")

bbox_props = dict(boxstyle="rarrow", fc=(0.8, 0.9, 0.9), ec="k", lw=2)
t = ax.text(-0.3, -2, "OverWritting", ha="center", va="center", rotation=45,
            size=10,
            bbox=bbox_props)

############################################################


plt.show()
```



In [ ]:

Kullanım Hataları:

Aşırı yoğun datalarda bu grafik türünde noktalar üst üste gösterilerek grafikte görsel hataların yapılmasına neden olurlar.(Over plotting diye aratığınızda bununla ilgili örnekleri ve çeşitli çözümleri bulabilirsiniz

Bu hatayı gidermek için verilerinizi alt gruplara ayırın. Bu durum verileriniz içerindeki farklı paternleri algılamanızı sağlayabilir. Farklı partnlerin veri görselleştime ile tespitini daha iyi anlamak için Wikipedia'dan Simpson Paradox'u aratabilirsiniz.

https://en.wikipedia.org/wiki/Simpson%27s_paradox (https://en.wikipedia.org/wiki/Simpson%27s_paradox)

[Count plot ile üst üste gelen noktaların büyük çizilmesi bu yöntemlerden biridir. (https://www.machinelearningplus.com/plots/top-50-matplotlib-visualizations-the-master-plots-python/#5.-Counts-Plot (https://www.machinelearningplus.com/plots/top-50-matplotlib-visualizations-the-master-plots-python/#5.-Counts-Plot))])

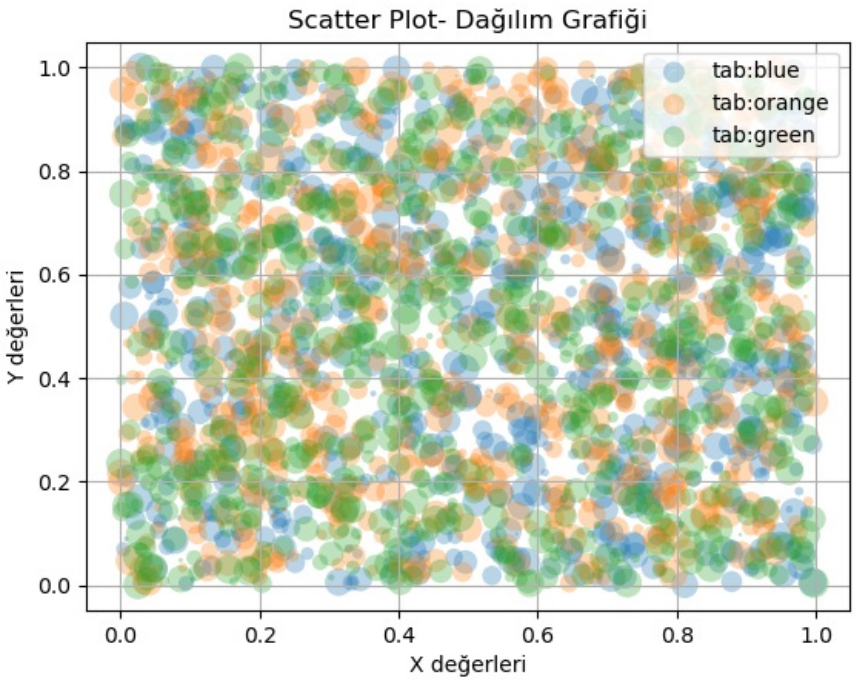
In [ ]:

```
```

Seaborn da stipplot az sayıda overplot noktasını göstermek için kullanılan bir yöntemdir. https://www.machinelearningplus.com/plots/top-50-matplotlib-visualizations-the-master-plots-python/#4.-Jittering-with-stripplot (https://www.machinelearningplus.com/plots/top-50-matplotlib-visualizations-the-master-plots-python/#4.-Jittering-with-stripplot)

In [7]:

```
import numpy as np
np.random.seed(19680801)
import matplotlib.pyplot as plt


fig, ax = plt.subplots()
for color in ['tab:blue', 'tab:orange', 'tab:green']:
    n = 750
    x, y = np.random.rand(2, n)
    scale = 200.0 * np.random.rand(n)
    ax.scatter(x, y, c=color, s=scale, label=color,
               alpha=0.3, edgecolors='none')

# Burada alpha=0.3 değeri ile overwritting farkedilir duruma getirildimiştir.


plt.title("Scatter Plot- Dağılım Grafiği")
ax.set_xlabel("X değerleri")
ax.set_ylabel("Y değerleri")

ax.legend()
ax.grid(True)

plt.show()
```



Grafiğe alpha değeri ekleyerek üst üst yazma durumunun gözlenmesini kolaylaştırabilirsiniz.

In [ ]:

```
```

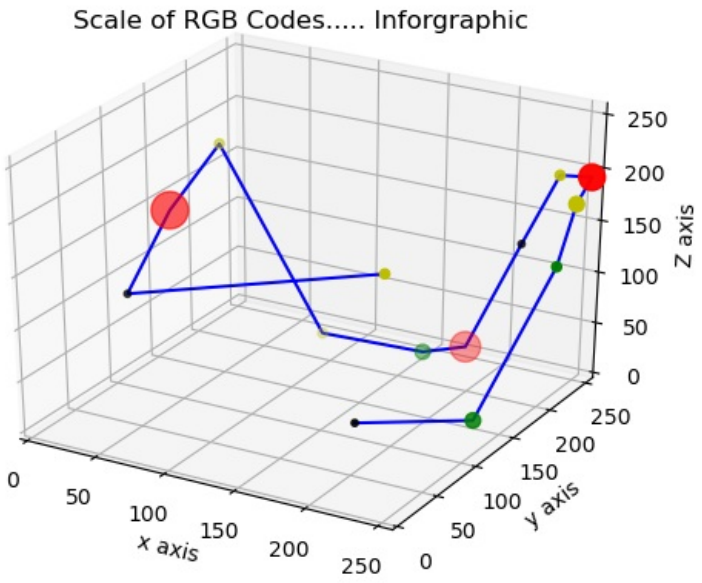
Üst üste yazma durumunu daha net görmek için verilerinizi 3D interaktif bir tabloda inceleyebilirsiniz ancak bu durumda bir raporlamada görüntüleme sıkıntılı olacaktır. Sadece görünütleme amaçlı kullanılması uygundur.

In [ ]:

```
```

In [9]:

```
import warnings

with warnings.catch_warnings():
    warnings.filterwarnings("ignore",category=DeprecationWarning)

%matplotlib notebook


import matplotlib.pyplot as plt

import numpy as np




from mpl_toolkits.mplot3d import Axes3D
import matplotlib.pyplot as plt




fig = plt.figure()
ax=fig.add_subplot(1,1,1,projection='3d')

x=[168,231,255,255,255,233,208,170,153,77,0,0,0,166]
y=[112,151,211,235,255,255,255,255,228,230,230,168,115,152]
z=[1,2,128,176,192,188,115,1,5,0,168,132,76,125]
size=[10,50,20,50,150,20,10,200,50,20,20,300,10,20]
color=['k','g','g','y','r','y','k','r','g','y','y','r','k','y']



ax.plot(x,y,z,c='b')
ax.scatter(x,y,z,c=color,marker='o',s=size)

ax.set_ylabel('y axis')
ax.set_ylim(0,255)
ax.set_xlabel('x axis')
ax.set_xlim(0,255)
ax.set_zlabel('Z axis')
ax.set_zlim(0,255)
ax.set_title('Scale of RGB Codes..... Inforgraphic')




plt.show()
```


Scale of RGB Codes..... Inforgraphic
Z axis
y axis
x axis
0
50
100
150
200
250

In [ ]: