import matplotlib.pyplot as plt

%matplotlib inline

import numpy as np

x = np.linspace(0,5,11) y = x \*\* 2

x

array([0. , 0.5, 1. , 1.5, 2. , 2.5, 3. , 3.5, 4. , 4.5, 5. ])

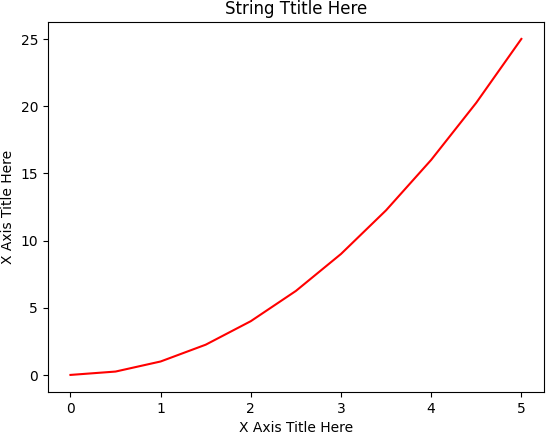
y

array([ 0. , 0.25, 1. , 2.25, 4. , 6.25, 9. , 12.25, 16. ,

20.25, 25. ])

plt.plot(x, y, 'r') #'r' is the color red plt.xlabel('X Axis Title Here')

plt.ylabel('X Axis Title Here') plt.title('String Ttitle Here') plt.show()

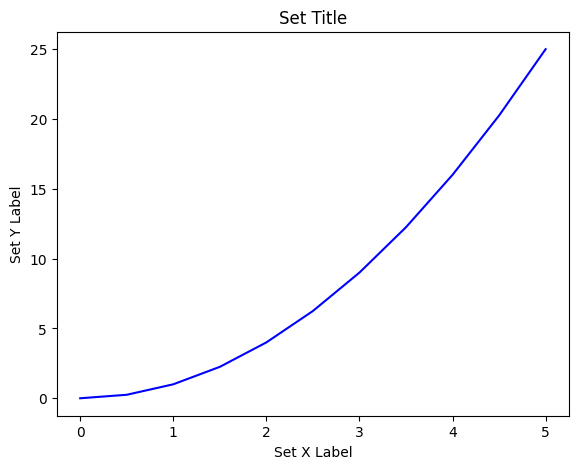
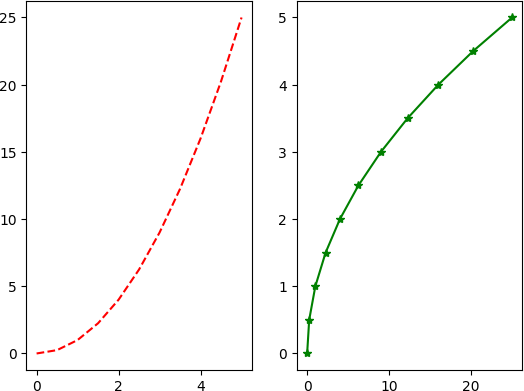


#plt.subplot(nrows, ncols, plot\_number) plt.subplot(1,2,1)

plt.plot(x ,y, 'r--') #More on color options later plt.subplot(1,2,2)

plt.plot(y, x, 'g\*-');

#Creat Figure (empty canvas) fig = plt.figure()



#Add set of axes to figure

axes = fig.add\_axes([0.1,0.1,0.8,0.8]) #Left,bottom,width,height

#Flot on that set of axes axes.plot(x, y, 'b')

axes.set\_xlabel('Set X Label') #Notice the use of set\_ to begin methods axes.set\_ylabel('Set Y Label')

axes.set\_title('Set Title')

Text(0.5, 1.0, 'Set Title')

#Creat blank canvas fig=plt.figure()

#Add set of axes to figure

axes1=fig.add\_axes([0.1,0.1,0.8,0.8]) #Main axes axes2=fig.add\_axes([0.2,0.5,0.4,0.3]) #Inset axes

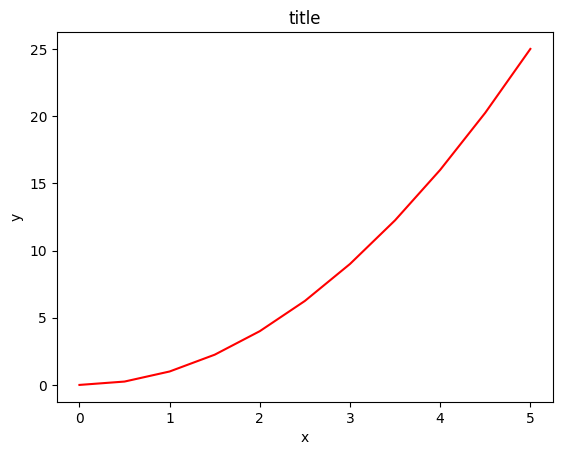
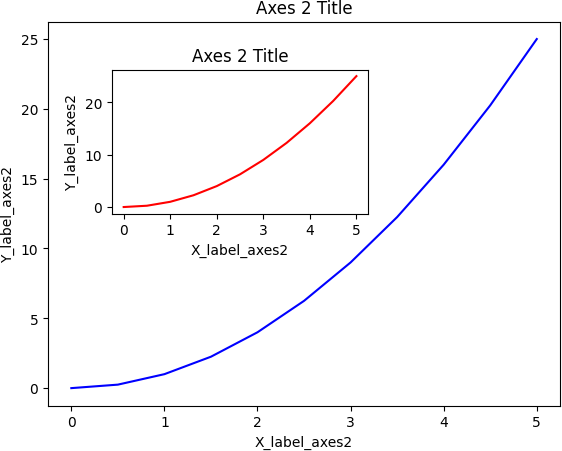
#Larger Figure Axes 1 axes1.plot(x, y, 'b')

axes1.set\_xlabel('X\_label\_axes2') axes1.set\_ylabel('Y\_label\_axes2') axes1.set\_title('Axes 2 Title')

#Larger Figure Axes 2 axes2.plot(x, y, 'r')

axes2.set\_xlabel('X\_label\_axes2') axes2.set\_ylabel('Y\_label\_axes2') axes2.set\_title('Axes 2 Title')

Text(0.5, 1.0, 'Axes 2 Title')



#Use similar to plt.figure() except use tuple unpacking to grab fig and axes fig, axes = plt.subplots()

#Now use the axes object ot add stuff to plot axes.plot(x ,y, 'r')

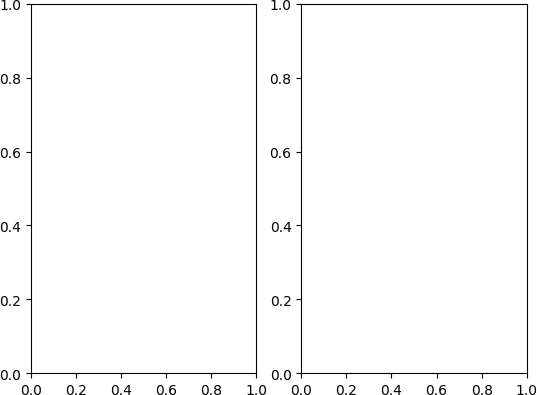
axes.set\_xlabel('x') axes.set\_ylabel('y')

axes.set\_title('title')

Text(0.5, 1.0, 'title')

#Empty canvas of 1 by 2 subplots

fig, axes = plt.subplots(nrows=1, ncols=2)



#Axes is an array of axes to plot on axes

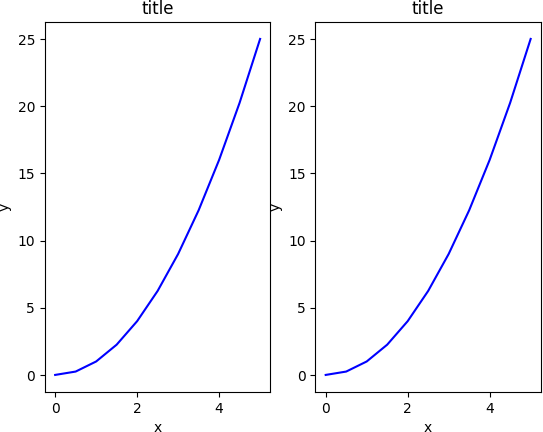
array([<Axes: >, <Axes: >], dtype=object)

for ax in axes:

ax.plot(x, y, 'b') ax.set\_xlabel('x') ax.set\_ylabel('y')

ax.set\_title('title')

#Display the figure object fig



fig, axe = plt.subplots(nrows=1, ncols=2) for ax in axes:

ax.plot(x, y, 'b') ax.set\_xlabel('x') ax.set\_ylabel('y')

ax.set\_title('title')

fig

plt.tight\_layout()

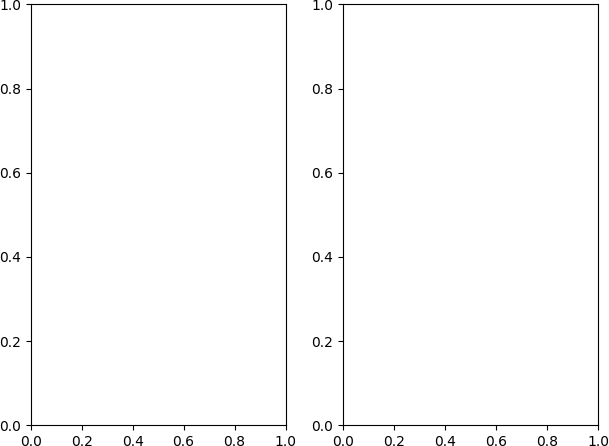


fig = plt.figure(figsize=(8, 4), dpi = 100)

<Figure size 800x400 with 0 Axes>

fig, axes = plt.subplots(figsize = (12,3))

axes.plot(x, y, 'r') axes.set\_xlabel('x') axes.set\_ylabel('y')

axes.set\_title('title');

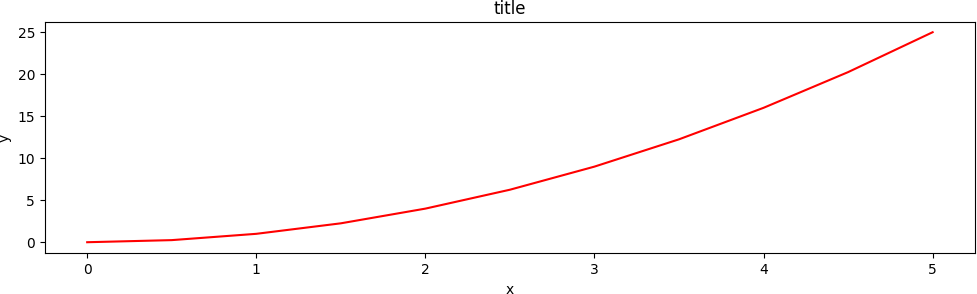


fig.savefig("filename.png")

fig.savefig("filename.png", dpi = 200)

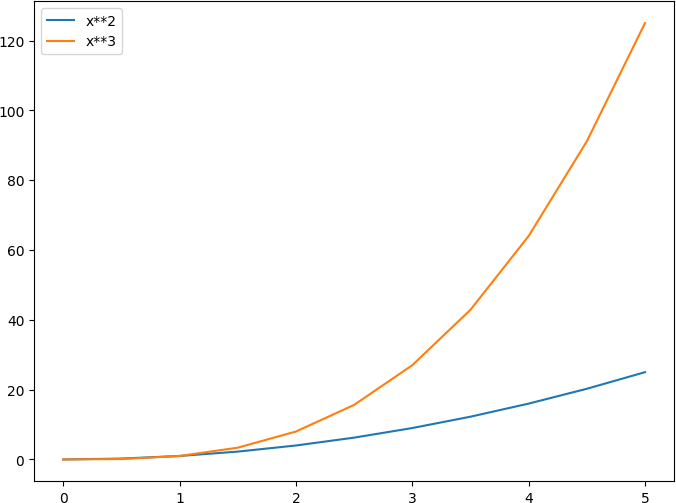
ax.set\_title("title");

ax.set\_xlabel("x") ax.set\_ylabel("y");

fig = plt.figure()

ax = fig.add\_axes([0, 0, 1, 1])

ax.plot(x, x\*\*2, label = "x\*\*2") ax.plot(x, x\*\*3, label = "x\*\*3") ax.legend()

<matplotlib.legend.Legend at 0x7fa14c696e30>

fig, axes = plt.subplots(1, 3, figsize = (12, 4))

axes[0].plot(x, x\*\*2, x, x\*\*3)

axes[0].set\_title("default axes ranges")

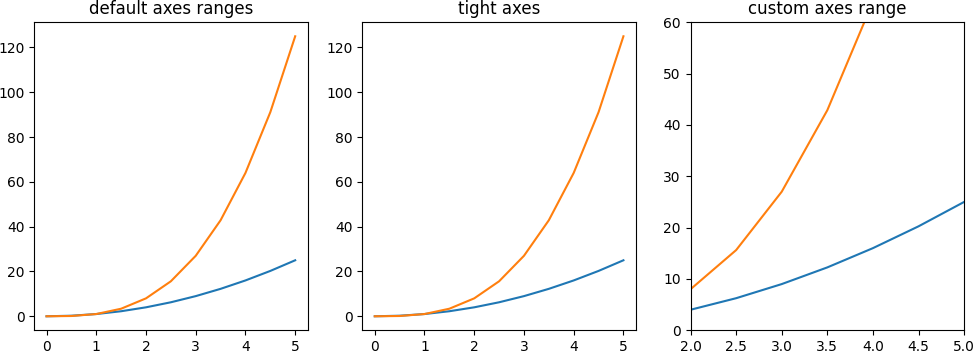
axes[1].plot(x, x\*\*2, x, x\*\*3) axes[1].axis('tight')

axes[1].set\_title("tight axes")

axes[2].plot(x, x\*\*2, x, x\*\*3) axes[2].set\_ylim([0, 60])

axes[2].set\_xlim([2, 5])

axes[2].set\_title("custom axes range");



import pandas as pd

import matplotlib.pyplot as plt import matplotlib.image as mpimg import seaborn as sns

%matplotlib inline

sns.get\_dataset\_names() ['anagrams',

'anscombe', 'attention',

'brain\_networks',

'car\_crashes', 'diamonds',

'dots',

'dowjones', 'exercise', 'flights', 'fmri',

'geyser',

'glue',

'healthexp', 'iris',

'mpg',

'penguins', 'planets', 'seaice',

'taxis',

'tips',

'titanic']

tips = sns.load\_dataset("tips") tips.head()

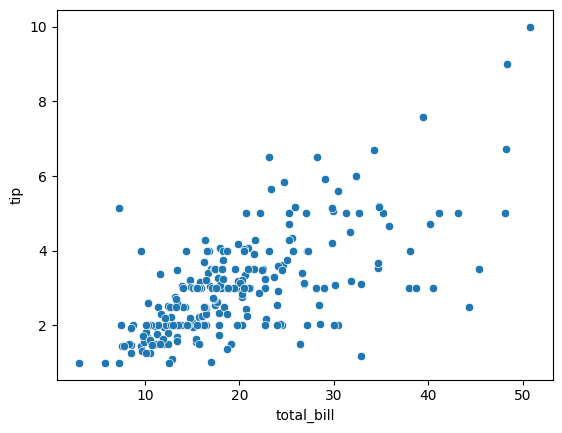
**0** 16.99 1.01 Female No Sun Dinner 2

**1** 10.34 1.66 Male No Sun Dinner 3

**2** 21.01 3.50 Male No Sun Dinner 3

**3** 23.68 3.31 Male No Sun Dinner 2

**4** 24.59 3.61 Female No Sun Dinner 4

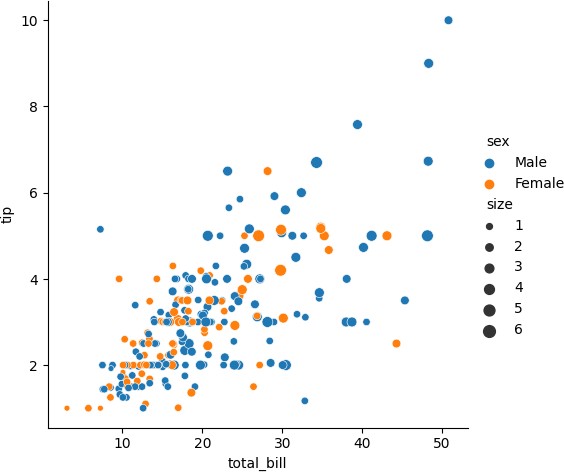


**total\_bill tip**

**sex smoker day time size**

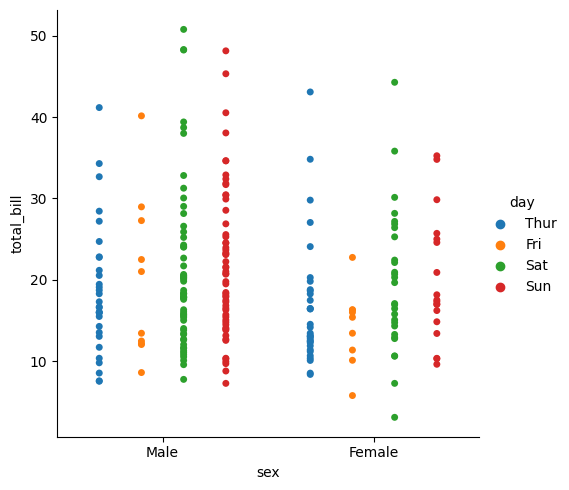
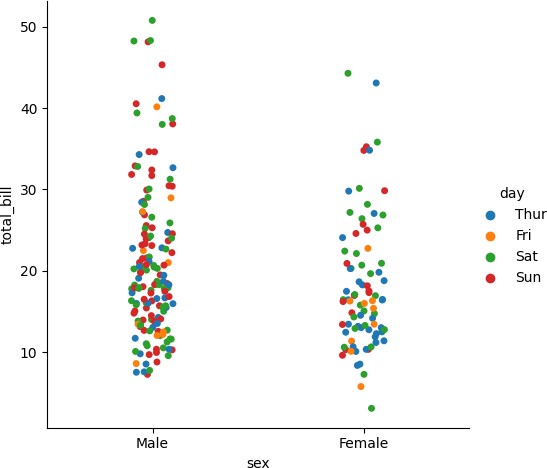
ax = sns.scatterplot(x = "total\_bill", y = "tip", data = tips)

sns.relplot(x = "total\_bill", y = "tip", data = tips, kind = "scatter", hue = "sex", size = "size",)

<seaborn.axisgrid.FacetGrid at 0x7fa1206238e0>

sns.catplot(x = "sex", y = "total\_bill", hue = "day", data = tips, kind = "strip")

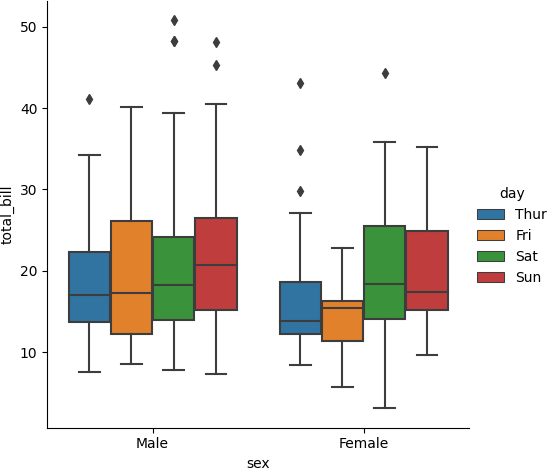
<seaborn.axisgrid.FacetGrid at 0x7fa12050ce50>



sns.catplot(x = "sex", y = "total\_bill", hue = "day", data = tips, kind = "strip", jitter = False, dodge = True)

<seaborn.axisgrid.FacetGrid at 0x7fa12073fca0>

sns.catplot(x ="sex", y = "total\_bill", hue = "day", data = tips, kind = "box")

<seaborn.axisgrid.FacetGrid at 0x7fa1204a2410>

import pandas as pd

import matplotlib.pyplot as plt import numpy as np

data = pd.read\_csv('job-market.csv')

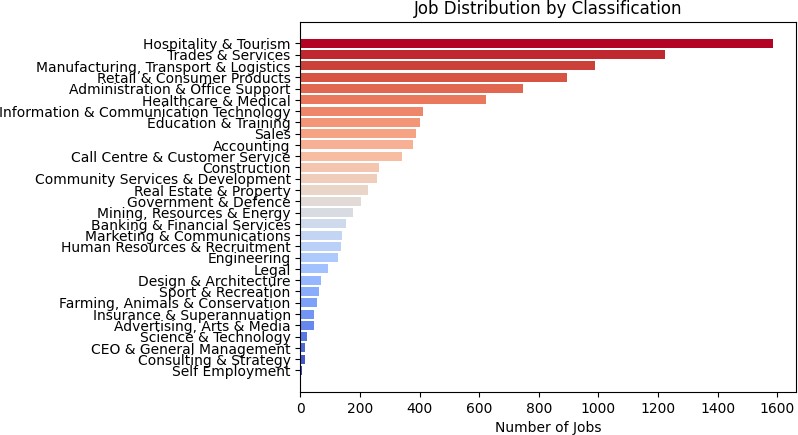
job\_counts = data.groupby('Classification').count()['Id'] job\_counts = job\_counts.sort\_values(ascending=False)

colors = plt.cm.coolwarm\_r(np.linspace(0, 1, len(job\_counts)))

fig, ax = plt.subplots()

ax.barh(job\_counts.index, job\_counts.values, color=colors) ax.invert\_yaxis()

ax.set\_xlabel('Number of Jobs')

ax.set\_title('Job Distribution by Classification') plt.show()

data['salary\_range'] = pd.cut(data['HighestSalary'], bins=[0,30,40,50,60]) data['mean\_salary'] = (data['LowestSalary'] + data['HighestSalary']) / 2

counts = data.groupby('salary\_range').size().reset\_index(name='count') fig, ax = plt.subplots(figsize=(8, 8))

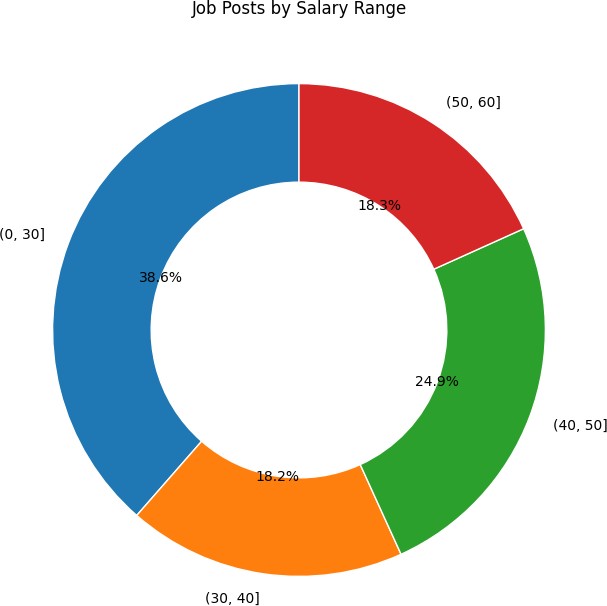
wedgeprops = {'width': 0.4, 'edgecolor': 'w'}

inner\_circle = plt.Circle((0, 0), 0.6, color='white') ax.add\_artist(inner\_circle)

sns.set\_palette('pastel')

ax.pie(counts['count'], labels=counts['salary\_range'], autopct='%1.1f%%', wedgeprops=wedgeprops, startangle=90)

ax.set\_title('Job Posts by Salary Range') plt.show()



import pandas as pd

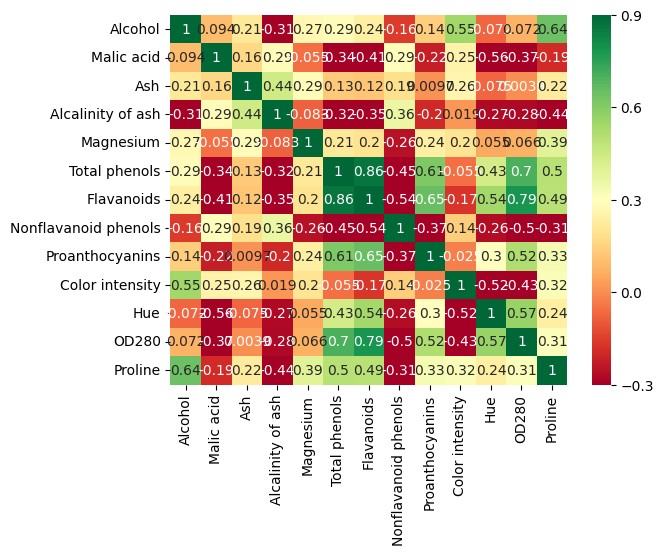
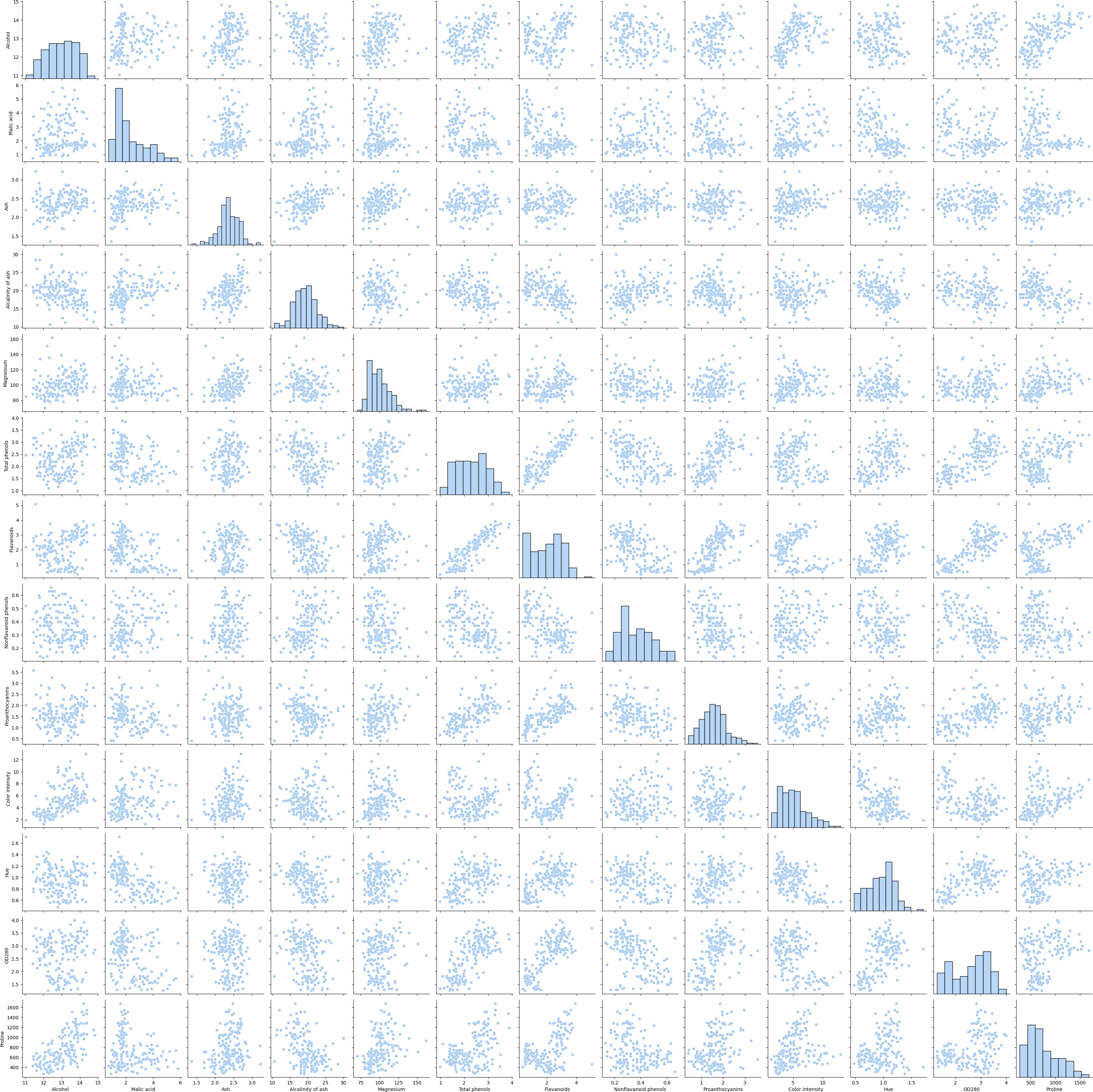
import matplotlib.pyplot as plt import seaborn as sns

data = pd.read\_csv('wine.data.csv')

data.drop('Label', axis=1, inplace=True) sns.pairplot(data)

<seaborn.axisgrid.PairGrid at 0x7fa11f376830>

import seaborn as sns



import matplotlib.pyplot as plt

heatmap = sns.heatmap(data.corr(), cmap='RdYlGn', vmin=-0.3, vmax=0.9, annot=True) heatmap.collections[0].colorbar.set\_ticks([-0.3, 0, 0.3, 0.6, 0.9])

plt.show()

from sklearn.cluster import KMeans

from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

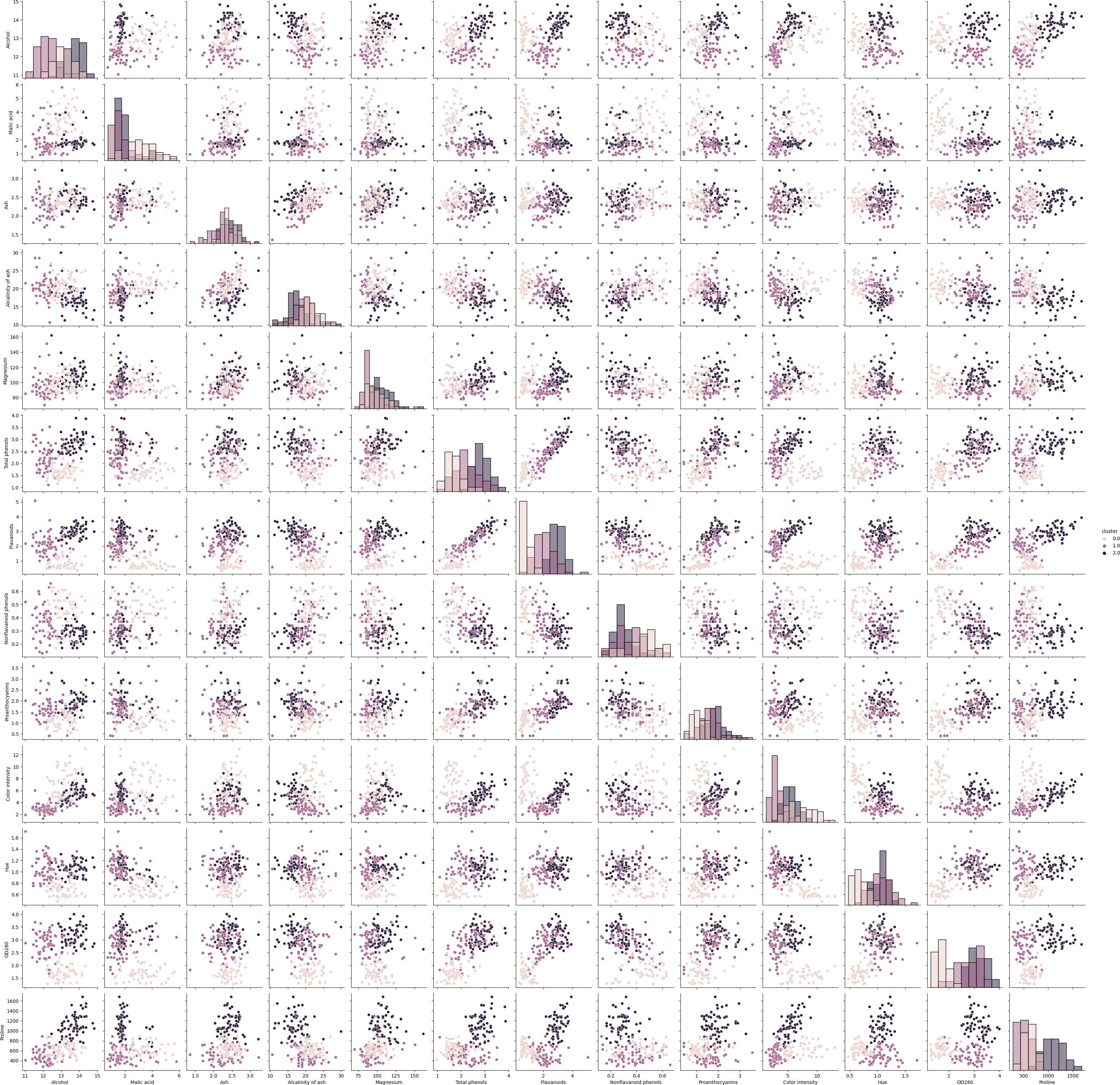
normalized\_data = scaler.fit\_transform(data)

kMeansClustering = KMeans(n\_clusters=3, random\_state=0)

label\_pred\_KM = kMeansClustering.fit\_predict(normalized\_data)

data['cluster'] = label\_pred\_KM.astype('float64')

sns\_plot = sns.pairplot(data, hue='cluster', diag\_kind='hist') plt.show()

/usr/local/lib/python3.10/dist-packages/sklearn/cluster/\_kmeans.py:870: FutureWarn warnings.warn(



 0 giây hoàn thành lúc 20:48

 