ML WORKSHOP DAY2

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import cv2


from sklearn import datasets,linear_model
from sklearn.linear_model import LinearRegression



#data=pd.read_csv()
iris=datasets.load_iris()
iris
data=pd.DataFrame(iris.data,columns=iris.feature_names)


print(data)
```

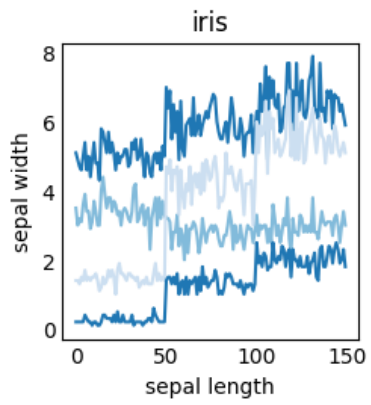
```
     sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)
0                  5.1               3.5                1.4               0.2
1                  4.9               3.0                1.4               0.2
2                  4.7               3.2                1.3               0.2
3                  4.6               3.1                1.5               0.2
4                  5.0               3.6                1.4               0.2
..                 ...               ...                ...               ...
145                6.7               3.0                5.2               2.3
146                6.3               2.5                5.0               1.9
147                6.5               3.0                5.2               2.0
148                6.2               3.4                5.4               2.3
149                5.9               3.0                5.1               1.8

[150 rows x 4 columns]
```

```
data.info()
```

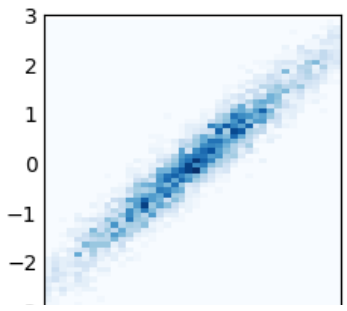
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 4 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   sepal length (cm)  150 non-null    float64
 1   sepal width (cm)   150 non-null    float64
 2   petal length (cm)  150 non-null    float64
 3   petal width (cm)   150 non-null    float64
dtypes: float64(4)
memory usage: 4.8 KB
```

```
plt.xlabel("sepal length")
plt.ylabel("sepal width")
plt.title("iris")
plt.plot(data)
```

```
[<matplotlib.lines.Line2D at 0x7ff937524b20>,
 <matplotlib.lines.Line2D at 0x7ff937524b80>,
 <matplotlib.lines.Line2D at 0x7ff937524bb0>,
 <matplotlib.lines.Line2D at 0x7ff937524ca0>]
```

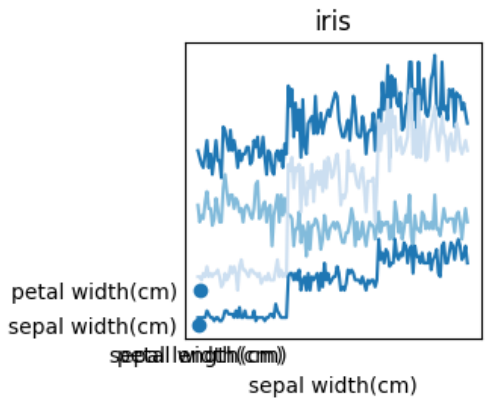



```
np.random.seed(0)
x=4+np.random.normal(0,1.5,200)


import matplotlib.pyplot as plt
import numpy as np

plt.style.use('_mpl-gallery-nogrid')

# make data: correlated + noise
np.random.seed(1)
x = np.random.randn(5000)
y = 1.2 * x + np.random.randn(5000) / 3

# plot:
fig, ax = plt.subplots()

ax.hist2d(x, y, bins=(np.arange(-3, 3, 0.1), np.arange(-3, 3, 0.1)))

ax.set(xlim=(-2, 2), ylim=(-3, 3))

plt.show()
```



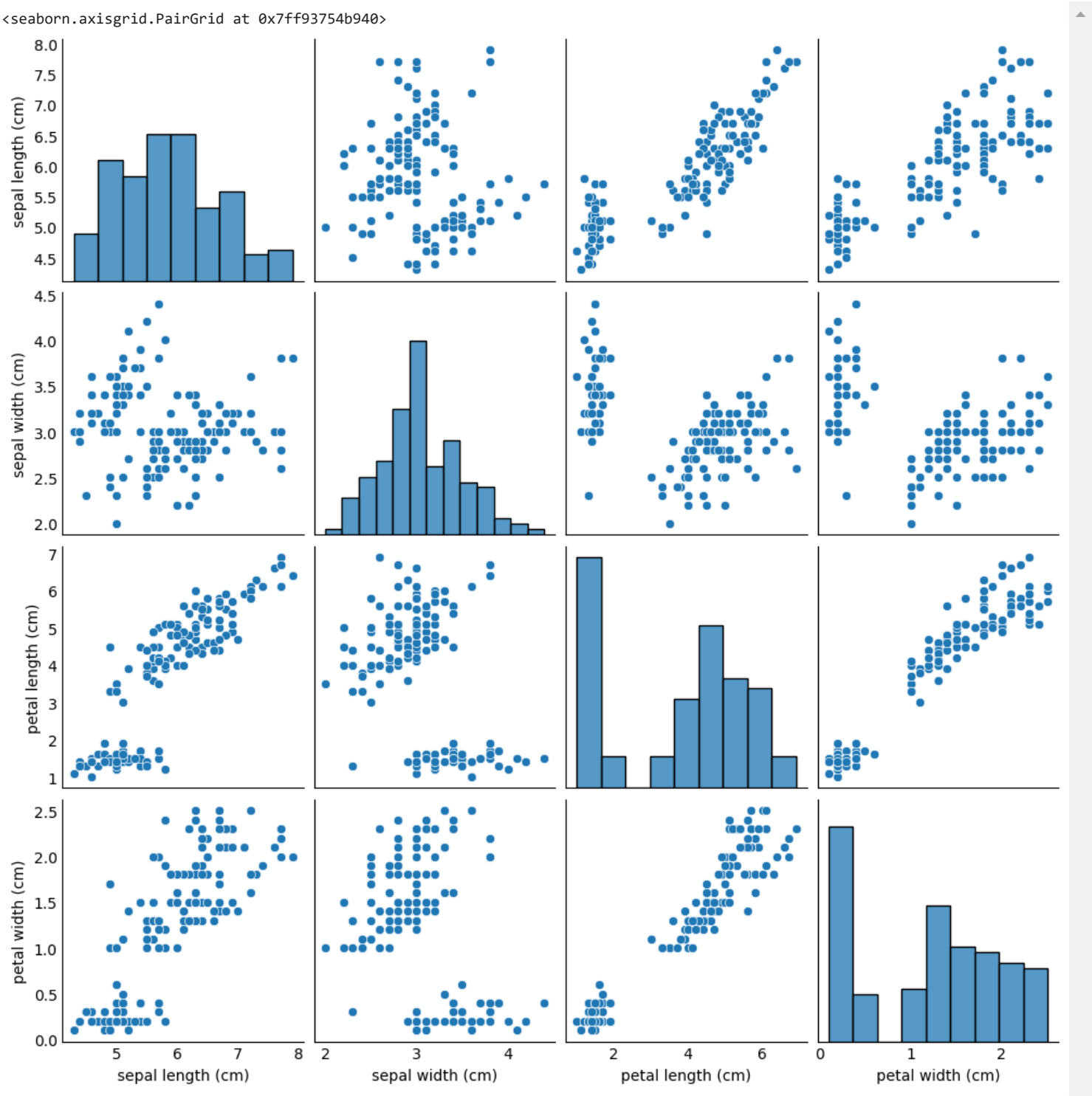

```
df=["sepal length(cm)","sepal width(cm)"]
x="sepal length(cm)","petal width(cm)"
y="sepal width(cm)","petal width(cm)"
plt.xlabel("sepal length(cm)")
plt.xlabel("sepal width(cm)")
plt.title("iris")
plt.plot(data)
plt.scatter(x,y)
```

```
<matplotlib.collections.PathCollection at 0x7ff9373e7be0>
```



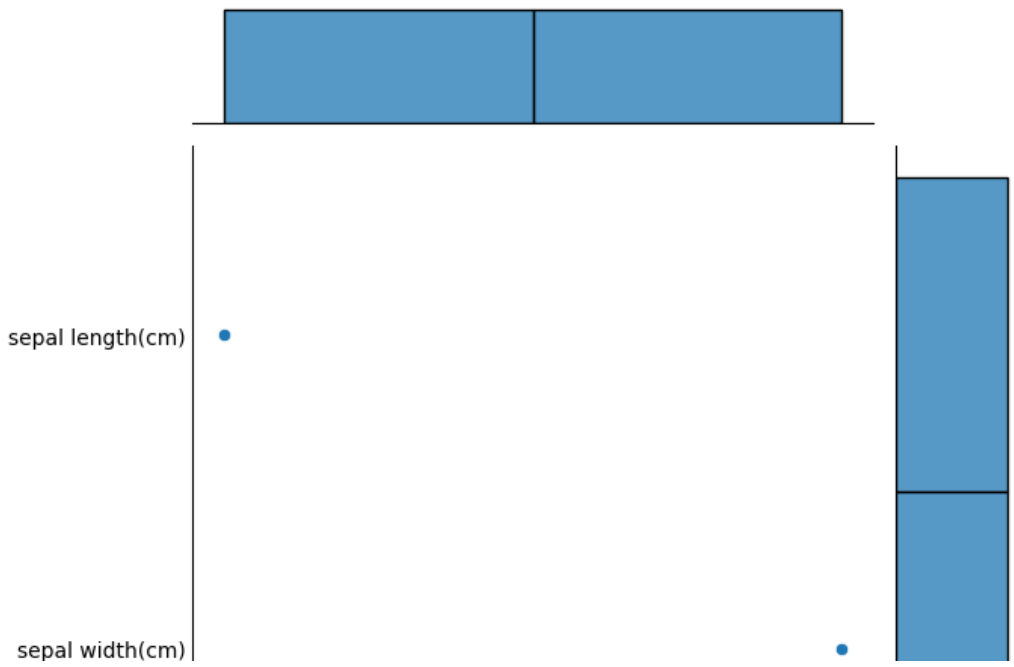

```
import seaborn as sns
sns.pairplot(data)
```

```
<seaborn.axisgrid.PairGrid at 0x7ff93754b940>
```



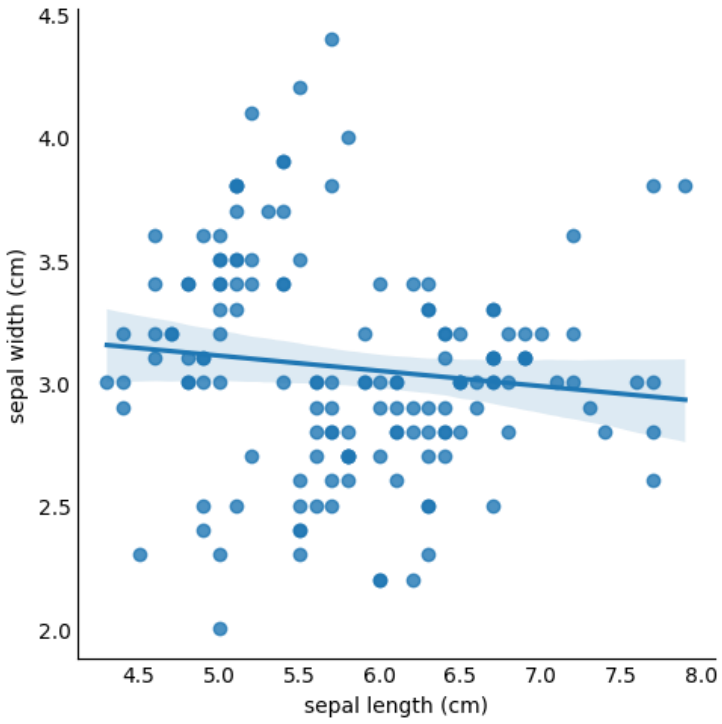

```
sns.jointplot(df)
```

```
<seaborn.axisgrid.JointGrid at 0x7ff936983f40>
```




```
sns.lmplot(data=data,x="sepal length (cm)",y="sepal width (cm)")
```

```
<seaborn.axisgrid.FacetGrid at 0x7ff937a67430>
```




```
x=data["sepal length (cm)"].values
y=data["sepal width (cm)"].values


from sklearn.model_selection import train_test_split
x_train,y_train,x_test,y_test=train_test_split(x,y,test_size=0.25,random_state=4)


x_test

     array([3.1, 3.8, 3. , 3.4, 3.1, 3.7, 2.8, 3. , 4.4, 3. , 3. , 3. , 3.5,
            2.5, 2.6, 3.4, 2.9, 2.9, 3.5, 2.9, 2.2, 2.6, 3.4, 3. , 3.4, 2.9,
            3. , 3. , 3.2, 3.4, 3.1, 2.5, 3.2, 3.4, 2.4, 3.3, 3.6, 3. , 2.8,
            3. , 2.5, 3.6, 2.6, 3.2, 3. , 3.5, 2.7, 2.3, 3.1, 2.7, 2.7, 2.8,
            3.2, 3.4, 2.8, 3. , 2.8, 2.2, 2.7, 3.6, 3.1, 3.3, 2.8, 3.7, 2.7,
            2.8, 2.5, 2.4, 3.5, 3. , 3. , 3. , 2.5, 2.6, 3.1, 3.4, 3.1, 2.8,
            4.2, 3. , 3.2, 3.2, 3.5, 4.1, 2.5, 3.2, 3. , 3.3, 2.9, 3.1, 3.8,
            3.3, 3.7, 3.5, 3.8, 3.1, 2.8, 3. , 3.8, 3. , 2.4, 2.8, 2.7, 3.6,
            2.9, 2.9, 3.1, 3.2, 2.3, 3. , 3. , 2.8])


x_train

     array([6.7, 5.1, 7.7, 5. , 6.7, 5.4, 6.4, 4.3, 5.7, 5.9, 6.1, 6.5, 5.2,
            5.6, 7.7, 6.3, 6.2, 5.7, 5. , 5.6, 6. , 5.5, 4.6, 5.6, 5.1, 6.4,
            6.8, 6.7, 6.5, 6. , 4.9, 4.9, 6.9, 5.4, 5.5, 6.3, 5. , 6.1, 6.5,
            5.9, 6.3, 4.9, 6.1, 6.4, 7.1, 5.5, 6.4, 5.5, 6.9, 5.8, 5.8, 6.1,
            5.9, 6.2, 5.7, 6.6, 5.8, 6. , 5.8, 4.6, 6.7, 5.1, 6.8, 5.3, 5.2,
            6.1, 5.5, 5.5, 5.1, 6.7, 6. , 5.7, 5.1, 5.7, 4.6, 5.2, 6.9, 5.6,
            5.5, 4.8, 4.4, 6.4, 5. , 5.2, 6.3, 6.8, 5.6, 5. , 4.4, 4.8, 7.7,
            6.3, 5.1, 5.1, 7.9, 6.9, 6.2, 4.4, 5.1, 6.5, 4.9, 5.7, 5.6, 7.2,
            6.3, 6.6, 6.4, 7. , 6.3, 6.5, 7.2, 7.7])


y_test

     array([2.8, 3.8, 2.8, 3. , 2.9, 2.9, 2.7, 4. , 3.9, 2.8, 3. , 3.4, 3. ,
            2.9, 2.5, 3.1, 2. , 3.2, 3.4, 2.7, 3.2, 3.3, 2.2, 3.4, 3.2, 3.8,
            2.3, 3.9, 3.4, 2.7, 3. , 3.2, 2.5, 3. , 2.3, 3.3, 3.2, 2.6])


y_train

     array([6.4, 5.7, 7.4, 7.6, 7.3, 6. , 6. , 5.8, 5.4, 6.3, 5. , 4.8, 4.8,
            6.1, 5.7, 4.9, 5. , 4.7, 4.8, 6.3, 5. , 6.7, 6.2, 5. , 4.7, 5.1,
            4.5, 5.4, 5.4, 5.8, 5.4, 4.6, 6.7, 4.9, 5. , 6.7, 7.2, 5.8])


x_train=x_train.reshape(-1,1)
y_test=y_test.reshape(-1,1)
x_test=x_test.reshape(-1,1)
y_train=y_train.reshape(-1,1)
```

```
reg=LinearRegression()
reg.fit(y_train,y_test)
```

```
▾ LinearRegression
LinearRegression()
```

```
y_predict=reg.predict(x_test)
y_predict
```

```
            [3.24322447],
            [3.22874402],
            [3.24322447],
            [3.28666584],
            [3.2504647 ],
            [3.18530265],
            [3.22150379],
            [3.20702334],
            [3.24322447],
            [3.17806242],
            [3.2504647 ],
            [3.24322447],
            [3.26494516],
            [3.27218539],
            [3.19254288],
            [3.22874402],
            [3.22874402],
            [3.22874402],
            [3.26494516],
            [3.25770493],
            [3.22150379],
            [3.19978311],
            [3.22150379],
            [3.24322447],
            [3.14186128],
            [3.22874402],
            [3.21426356],
            [3.21426356],
            [3.19254288],
            [3.14910151],
            [3.26494516],
            [3.21426356],
            [3.22874402],
            [3.20702334],
            [3.23598425],
            [3.22150379],
            [3.1708222 ],
            [3.20702334],
            [3.17806242],
            [3.19254288],
            [3.1708222 ],
            [3.22150379],
            [3.24322447],
            [3.22874402],
            [3.1708222 ],
            [3.22874402],
            [3.27218539],
            [3.24322447],
            [3.2504647 ],
            [3.18530265],
            [3.23598425],
            [3.23598425],
            [3.22150379],
            [3.21426356],
            [3.27942561],
            [3.22874402],
            [3.22874402],
            [3.24322447]])
```

```
reg.score(y_train,y_test)*100
```

```
    1.6890536324479122
```

```
reg.score(x_test,y_predict)*100
```

```
100.0
```

✓ 0s completed at 3:16 PM