Візуалізація даних

Людське око (і мозок) є дуже потужним інструментом для аналізу даних. Тому візуалізація відіграє важливу роль в аналізі даних. Двовимірна візуалізація використовує дві ортогональні осі координат і представляє кожен вектор ознак як точку в системі координат. Візуалізація лише однієї ознаки називається (простою) діаграмою. Візуалізацію більш ніж однієї функції можна зробити за допомогою діаграм розсіювання. Двовимірна діаграма розсіювання відповідає кожному об’єкту одній із двох осей координат, тому площина об’єкта відповідає площині візуалізації.

Візуалізація даних

Для візуалізації даних використовується бібліотека Matplotlib, яка працює з ndarray. Існують бібліотеки, що надають API для роботи з Matplotlib, які дозволяють розширювати функціональні можливості. Наприклад, бібліотека Seaborn, зокрема, працює з об’єктами бібліотеки Pandas. Також, в самій бібліотеці Pandas є вбудовані функції для побудови основних діаграм.

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

Візуалізація даних

Основною бібліотекою для візуалізації даних в Python є Matplotlib. Графіки, створені за допомогою Matplotlib, вимагають більше рядків коду та налаштування параметрів графіка, порівняно з іншими бібліотеками, такими як Seaborn і Pandas.

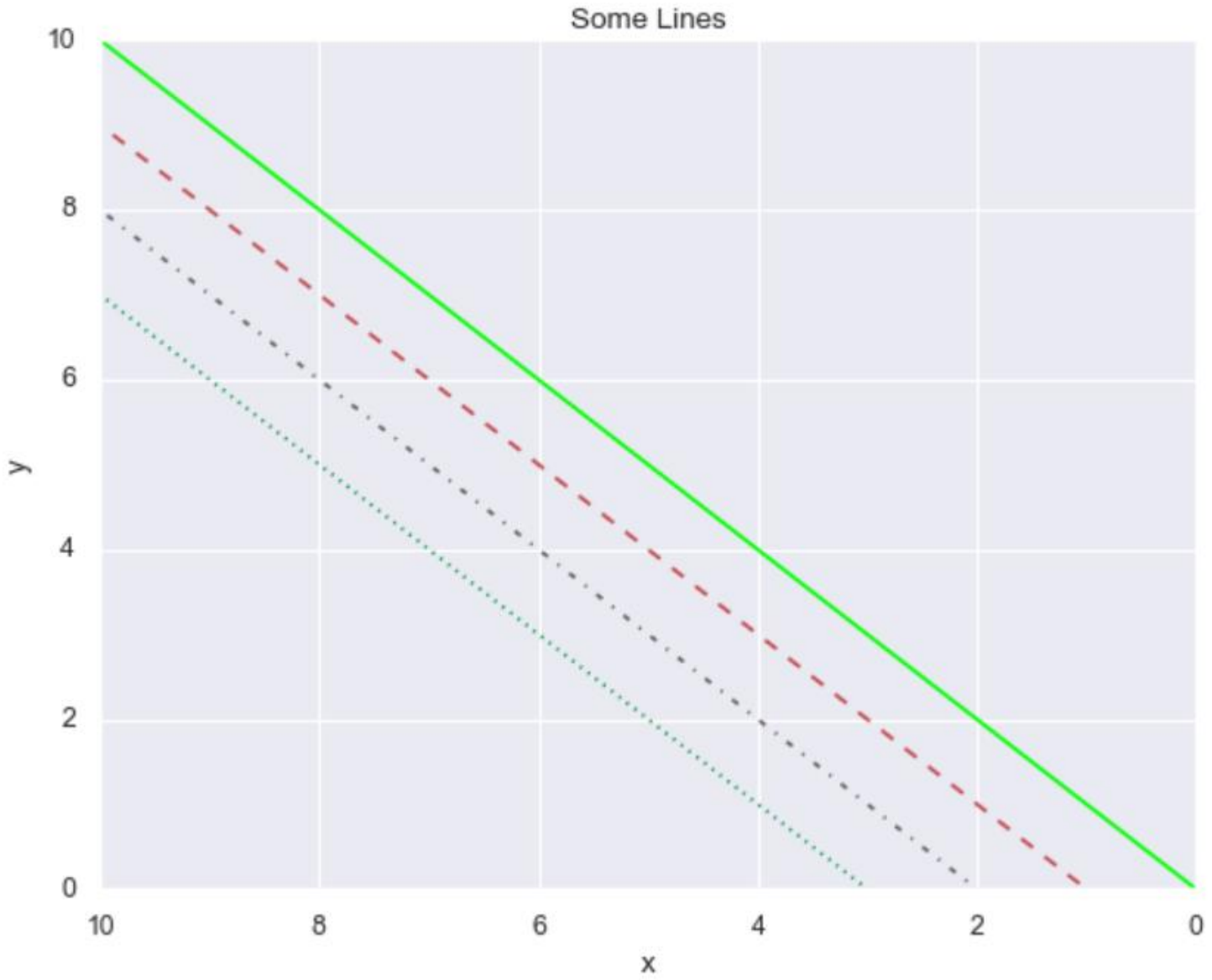
У Matplotlib є два інтерфейси, об’єктно-орієнтований і з збереженням стану. Інтерфейс із збереженням стану використовує клас pyplot. Створюється єдиний об’єкт цього класу, який використовується для всіх задач побудови графіків.

В об’єктно-орієнтованому інтерфейсі використовуються різні об’єкти для різних елементів графіку. Два основних об’єкти, які використовуються в цьому інтерфейсі для побудови графіків:

- Об’єкт figure, який виконує роль контейнера для інших об’єктів.
- Об’єкт axes, який є фактичною областю графіка, що містить вісь x, вісь y, точки, лінії, легенди та мітки.

Приклад побудови графіків в Matplotlib:

```
x = np.linspace(0, 10, 100)
plt.title("Some Lines")
plt.xlabel("x")
plt.ylabel("y");
plt.plot(x, x, color='lime’,
linestyle='solid')
plt.plot(x, x - 1, color='r’,
linestyle='dashed')
plt.plot(x, x - 2, color='0.4’,
linestyle='dashdot')
plt.plot(x, x - 3, color='#099A4D’,
linestyle='dotted');
plt.axis([10, 0, 0, 10]);
```



Візуалізація даних

Бібліотека Pandas використовує бібліотеку Matplotlib поза лаштунками для візуалізації, але побудова графіків за допомогою функцій Pandas набагато інтуїтивніша та зручніша для користувача.

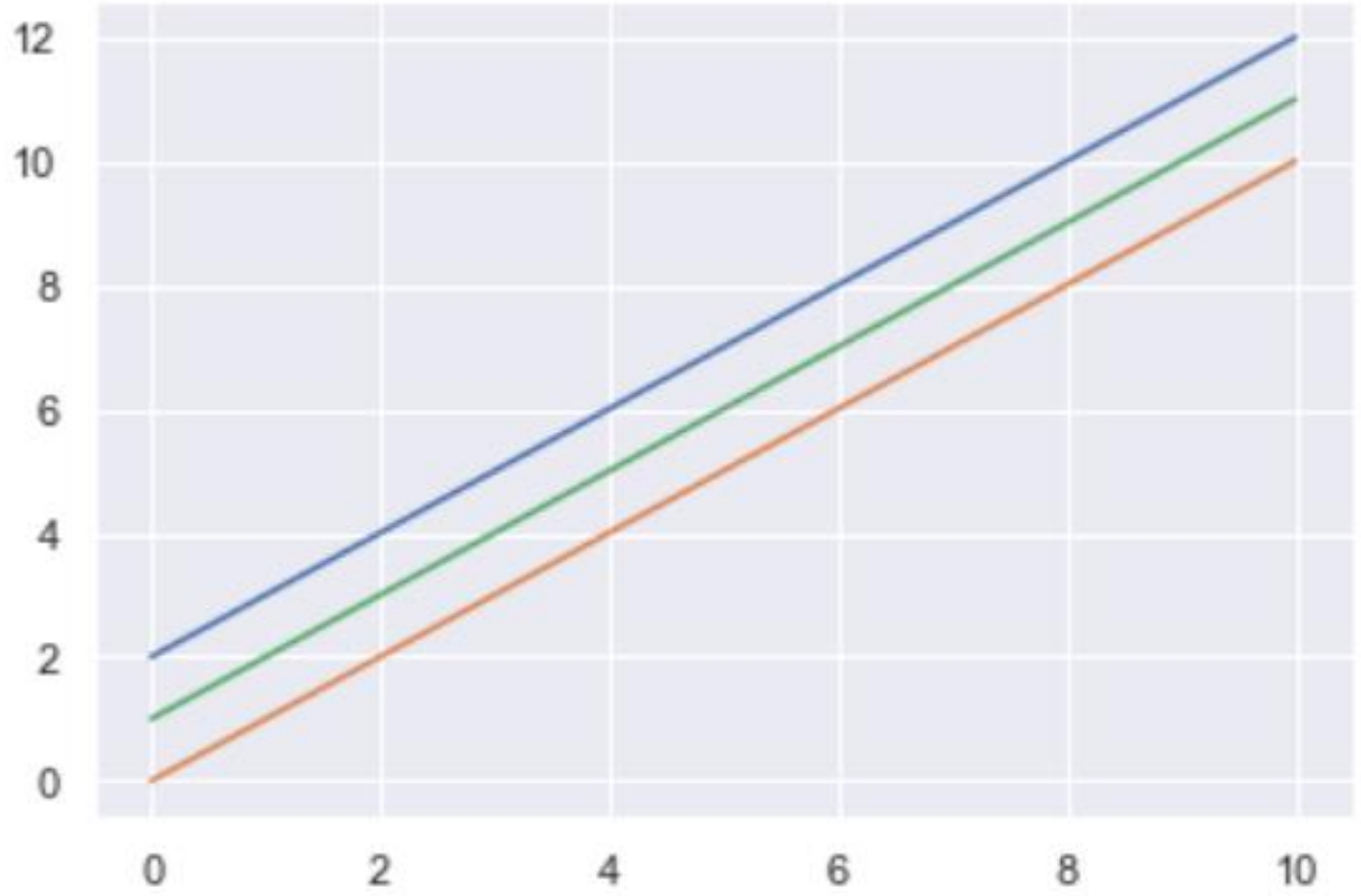
Функція plot (на основі функції Matplotlib plot), що використовується в Pandas, дозволяє створювати різноманітні графіки, просто налаштовуючи значення параметра kind, який визначає тип графіка.

Візуалізація даних

Seaborn — ще одна бібліотека візуалізації даних на основі Python. Seaborn змінює типові властивості Matplotlib, щоб налаштувати палітру кольорів і автоматично виконати агрегацію стовпців. Налаштування за замовчуванням полегшують написання коду, необхідного для створення різних графіків.

Приклад побудови графіків в Seaborn:

```
a = np.linspace(0, 10, 100);
sns.lineplot(x=a, y=a+2);
sns.lineplot(x=a, y=a);
sns.lineplot(x=a, y=a+1);
```



Візуалізація даних

Використаємо декілька наборів даних для прикладів візуалізації:

fish

Species	Weight	Length1	Length2	Length3	Height	Width
Bream	242.0	23.2	25.4	30.0	11.5200	4.0200
Bream	290.0	24.0	26.3	31.2	12.4800	4.3056
Bream	340.0	23.9	26.5	31.1	12.3778	4.6961
Bream	363.0	26.3	29.0	33.5	12.7300	4.4555
Bream	430.0	26.5	29.0	34.0	12.4440	5.1340

cats

	Unnamed: 0	Sex	Bwt	Hwt
0	1	M	2.0	6.5
1	2	M	2.0	6.5
2	3	M	2.1	10.1
3	4	M	2.2	7.2
4	5	M	2.2	7.6

users

	Age	Gender	Region	Occupation	Income	Has Laptop
0	14	male	city	student	0	no
1	34	female	city	teacher	22000	no
2	42	male	countryside	banker	24000	yes
3	30	male	countryside	teacher	25000	no
4	16	male	city	student	0	no

heights

	Family	Father	Mother	Gender	Height	Kids
0	1	199.39	170.18	M	185.928	4
1	1	199.39	170.18	F	175.768	4
2	1	199.39	170.18	F	175.260	4
3	1	199.39	170.18	F	175.260	4
4	2	191.77	168.91	M	186.690	4

Візуалізація даних

Почнемо з візуалізації однієї ознаки.

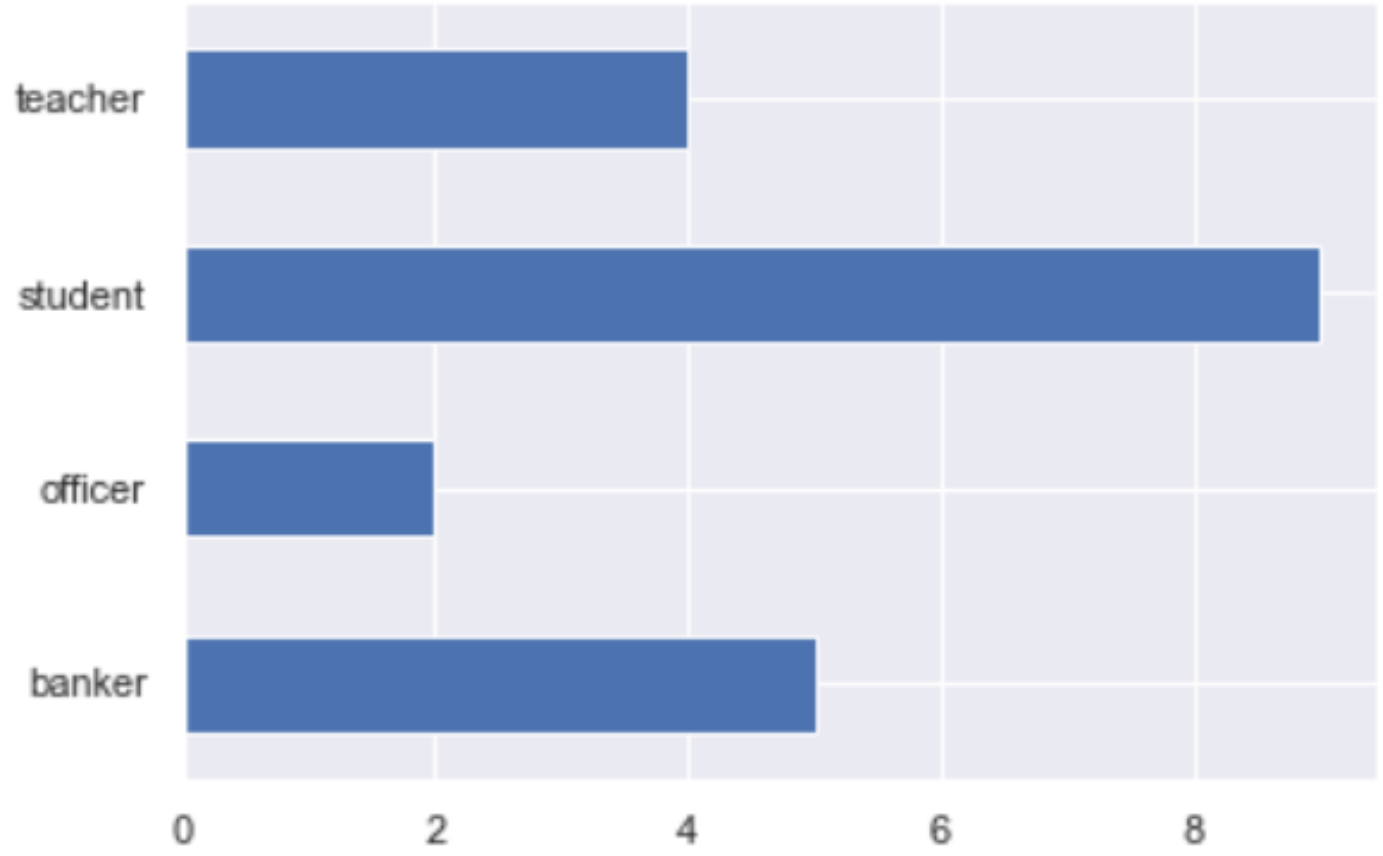
Стовпчикова діаграма створюється шляхом позначення всіх категорій даних на одній осі та частоти кожної категорії даних по іншій осі. Висота стовпчика (якщо діаграма вертикальна) або його довжина (якщо діаграма горизонтальна) показує частоту кожної категорії. Категорії не впорядковані, а між стовпчиками зазвичай присутні проміжки.

Стовпчикова діаграма використовується, коли потрібно порівняти значення показників у різних підгрупах даних.

Часто по осі x відкладається ознака з якісними значеннями. Якісні ознаки набувають фіксованої кількості значень.

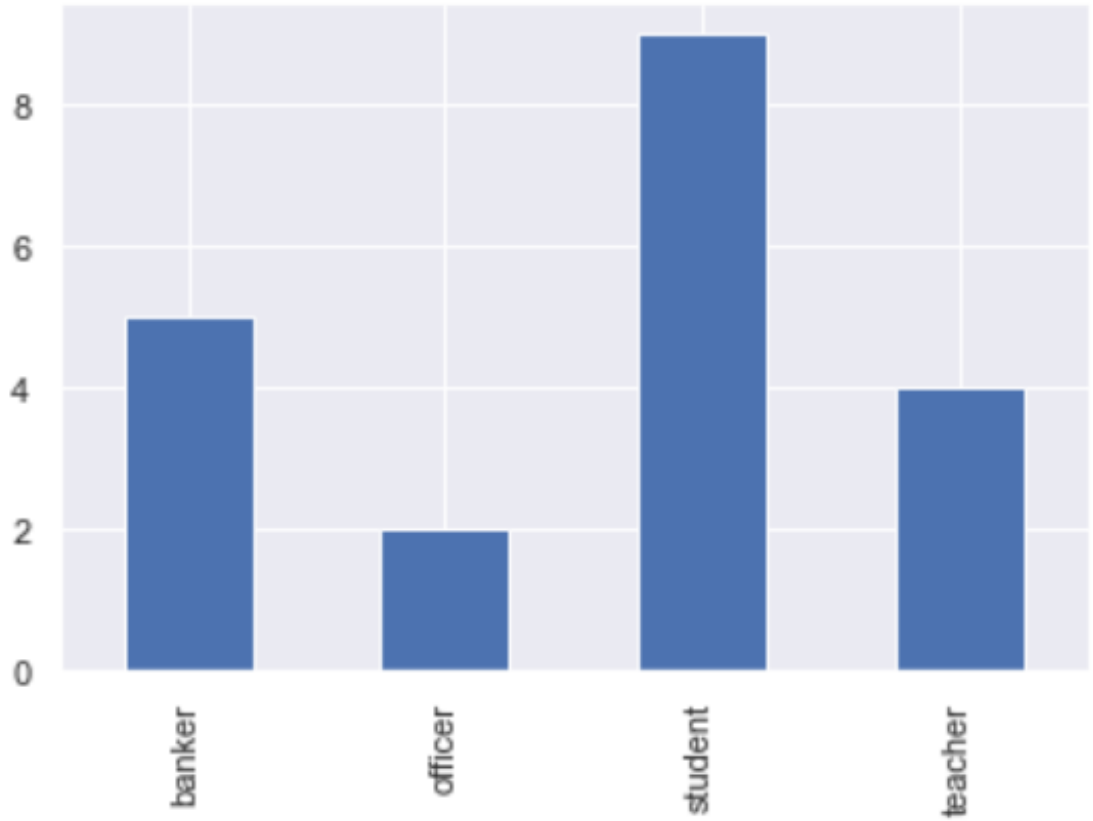
Візуалізація даних

users['Occupation'].value_counts().sort_index().plot.barh();



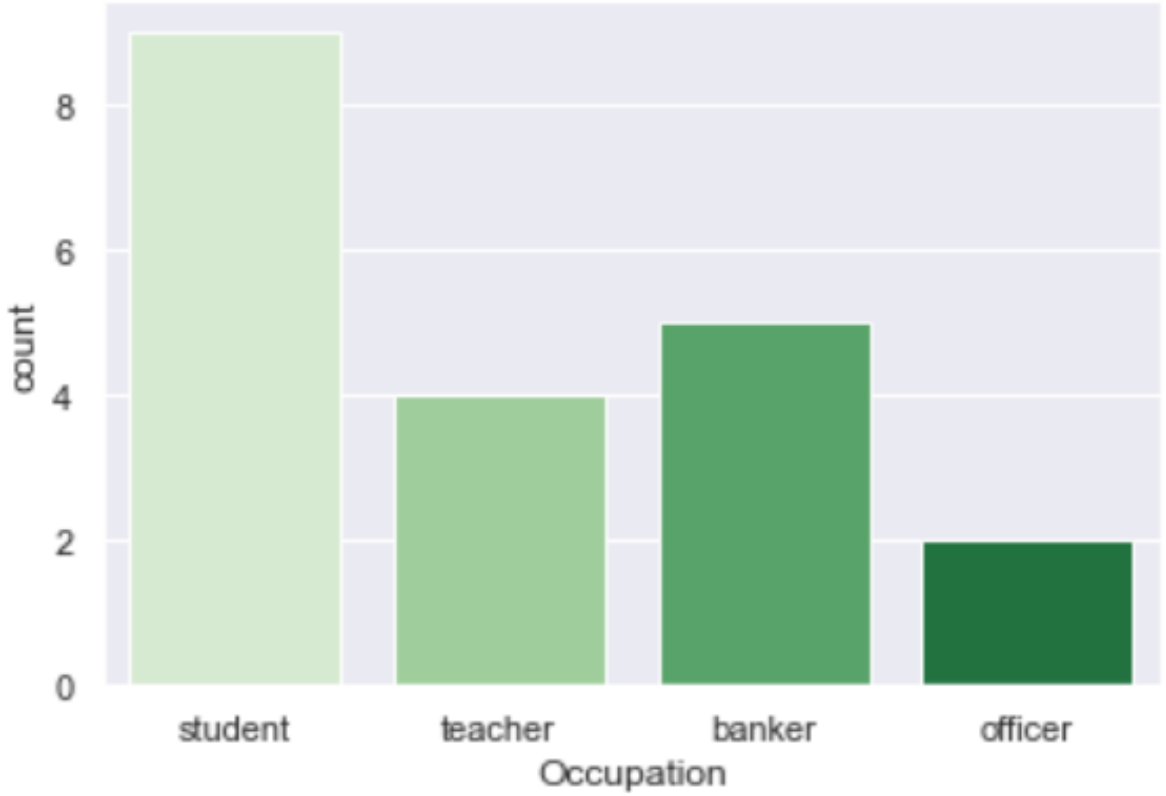
Візуалізація даних

users['Occupation'].value_counts().sort_index().plot.bar();



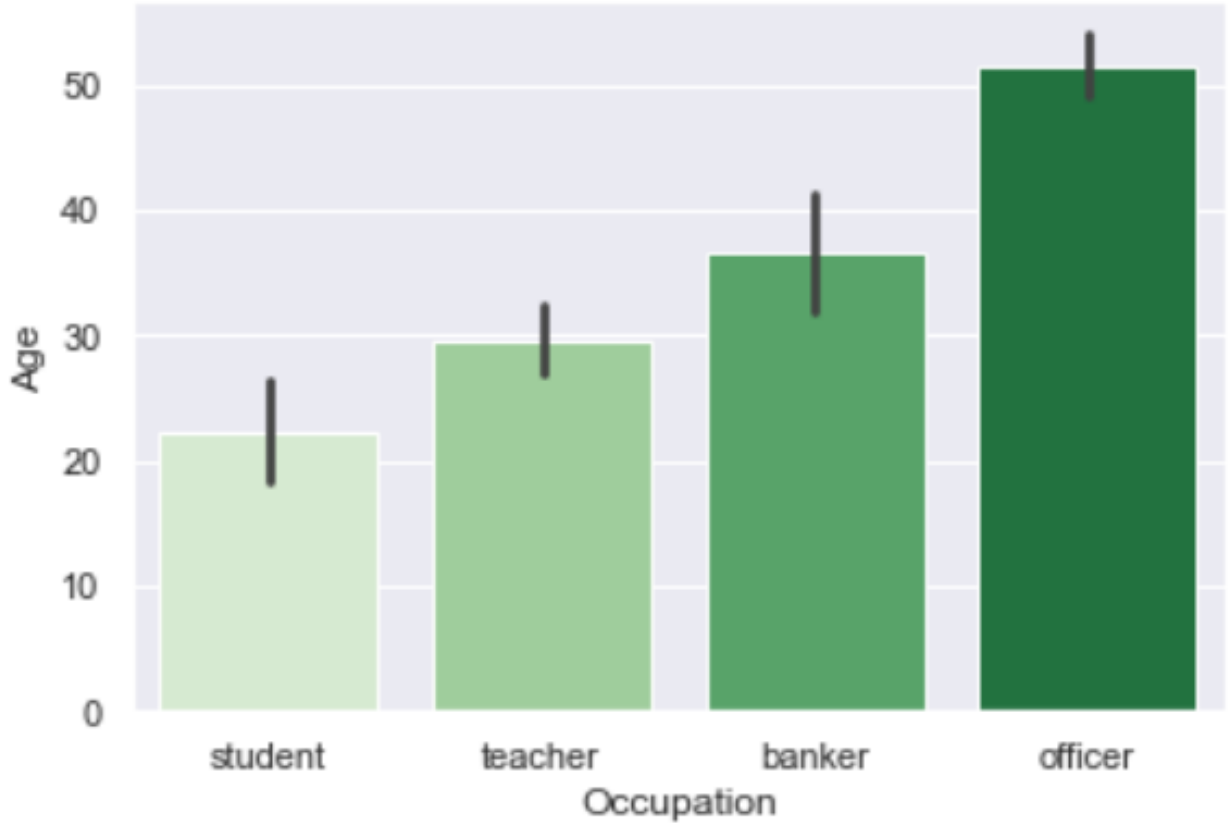
Візуалізація даних

```
sns.countplot(users['Occupation'],palette = "Greens");
```



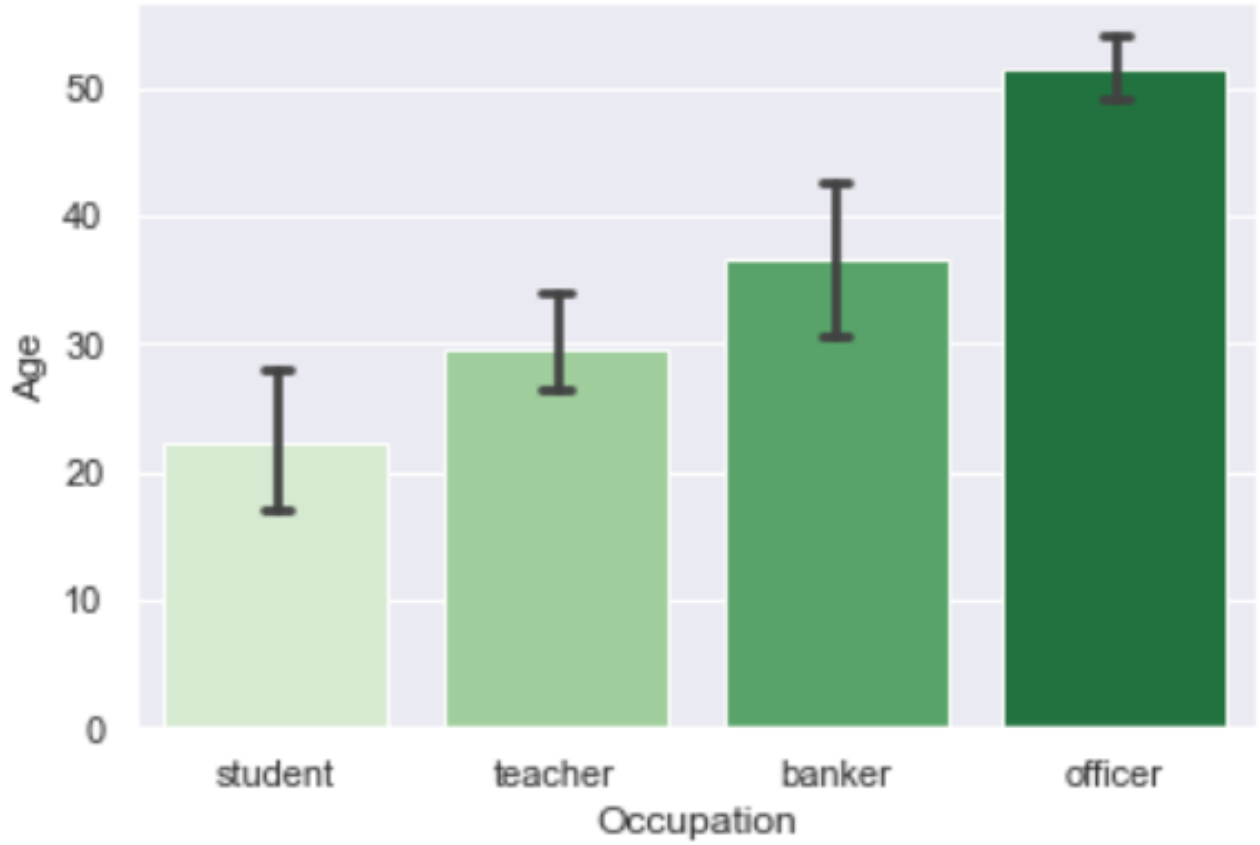
Візуалізація даних

sns.barplot(x='Occupation', y='Age', data=users,palette = "Greens");



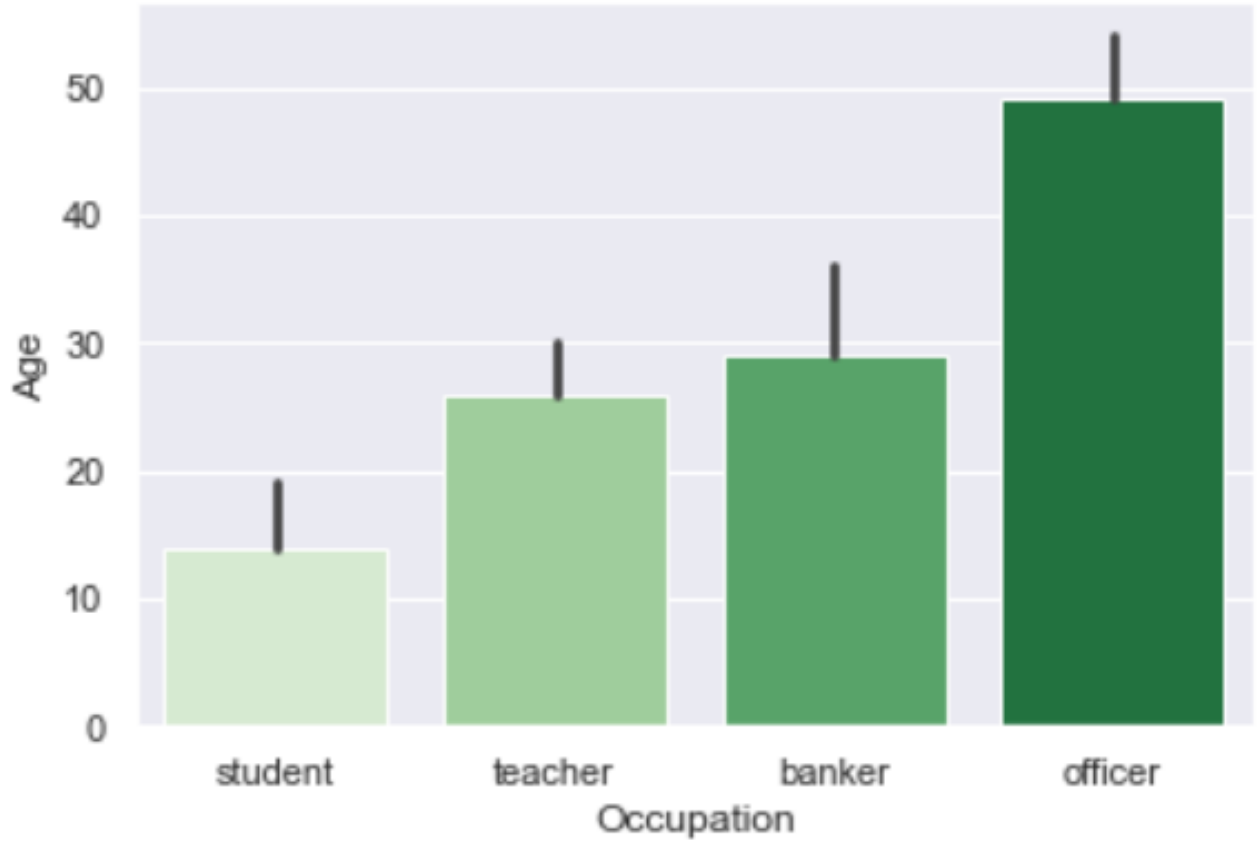
Візуалізація даних

sns.barplot(data=users, x='Occupation', y='Age', capsize=0.1, ci=99, palette = "Greens");



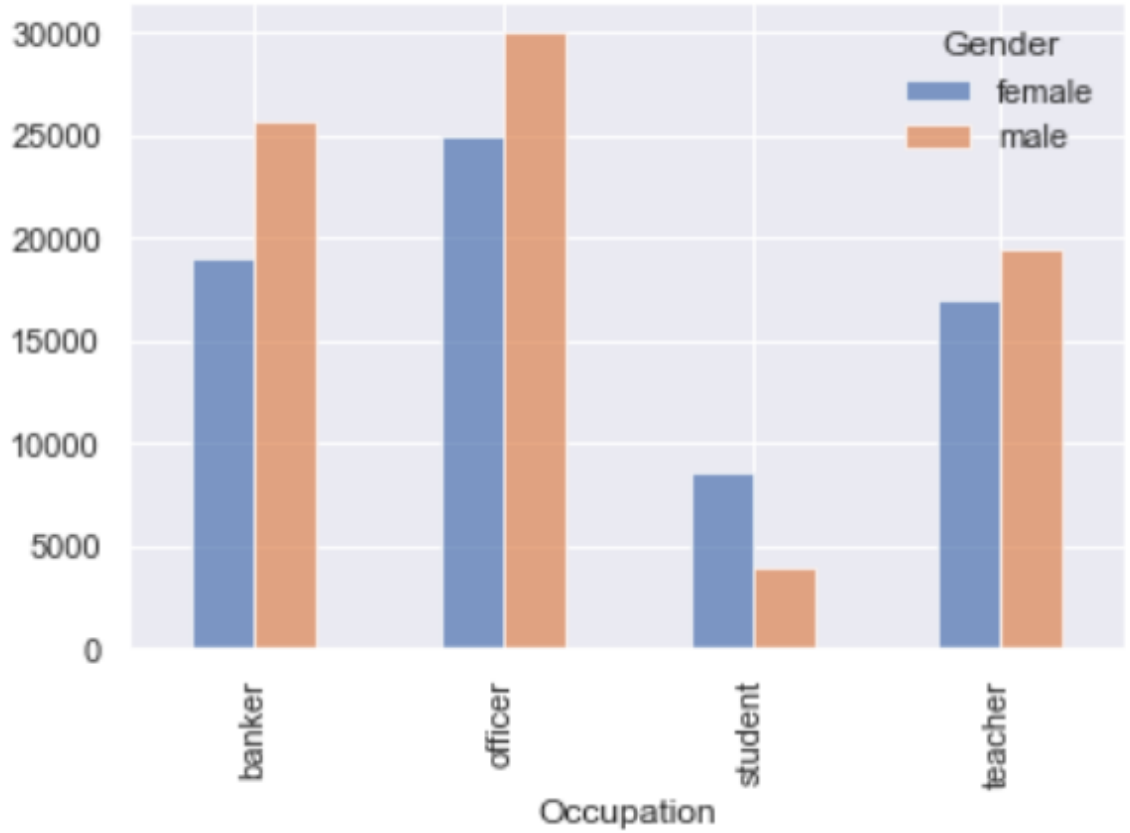
Візуалізація даних

```
sns.barplot(data=users, x='Occupation', y='Age',
estimator=np.min, palette = "Greens");
```



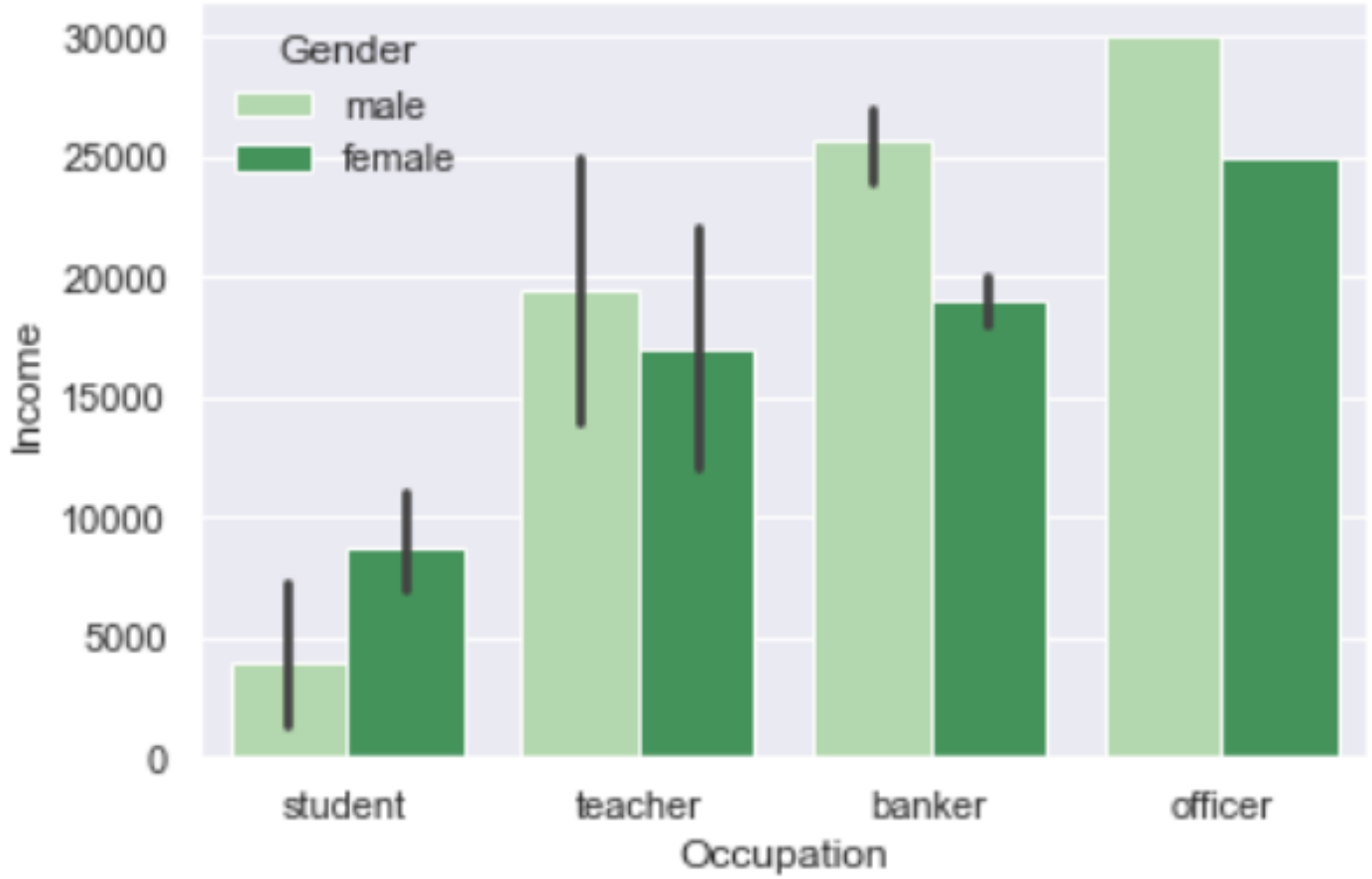
Візуалізація даних

```
users_pivot = pd.pivot_table(users,
values="Income",index="Occupation",columns="Gender", aggfunc=np.mean)
ax = users_pivot.plot(kind="bar",alpha=0.7)
plt.show()
```



Візуалізація даних

sns.barplot(x = 'Occupation',y = 'Income',hue = 'Gender',data = users, palette = "Greens");



Візуалізація даних

Кількісні ознаки набувають впорядкованих числових значень. Ці значення можуть бути дискретними, як цілі числа, або неперервними, як дійсні числа, і зазвичай виражають підрахунок або вимірювання.

Найпростіший спосіб подивитися на розподіл кількісної ознаки - побудувати її гістограму

Візуалізація даних

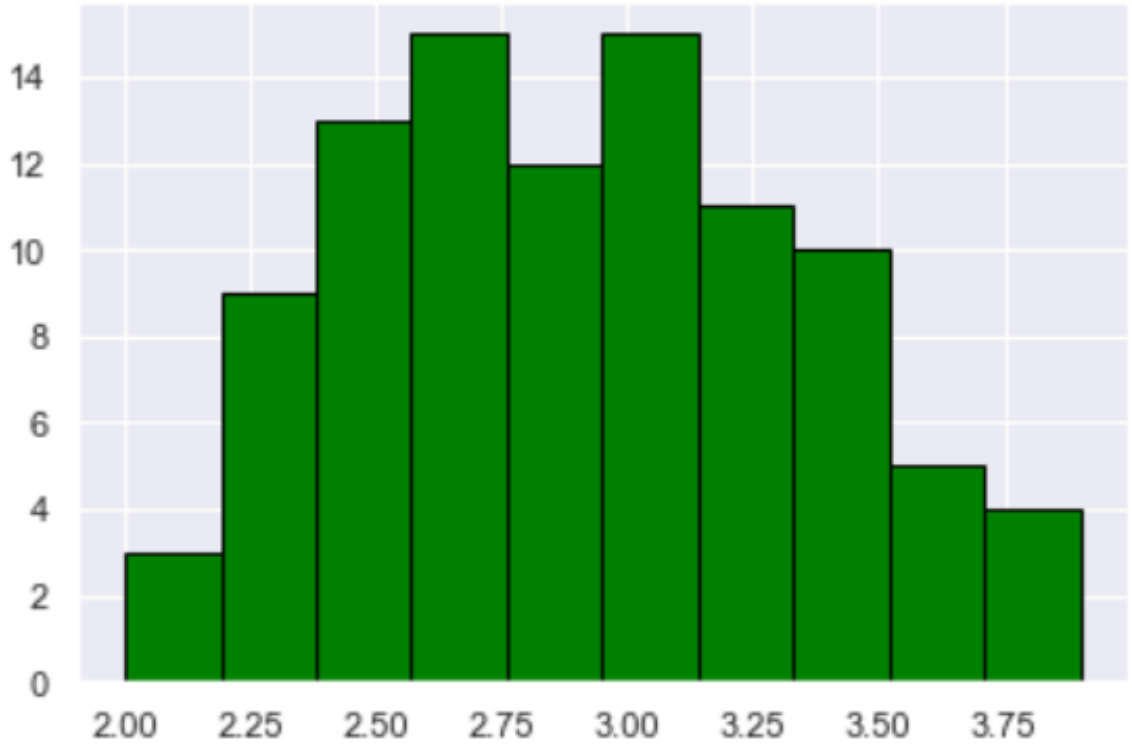
Щоб побудувати гістограму частот, потрібно розділити діапазон даних на інтервали. Якщо це можливо, інтервали повинні бути однакової ширини. Кількість інтервалів залежить від кількості спостережень та від розсіювання даних.

Форма гістограми може містити підказки про базовий тип розподілу: нормальний, експоненціальний тощо.

Візуалізація даних

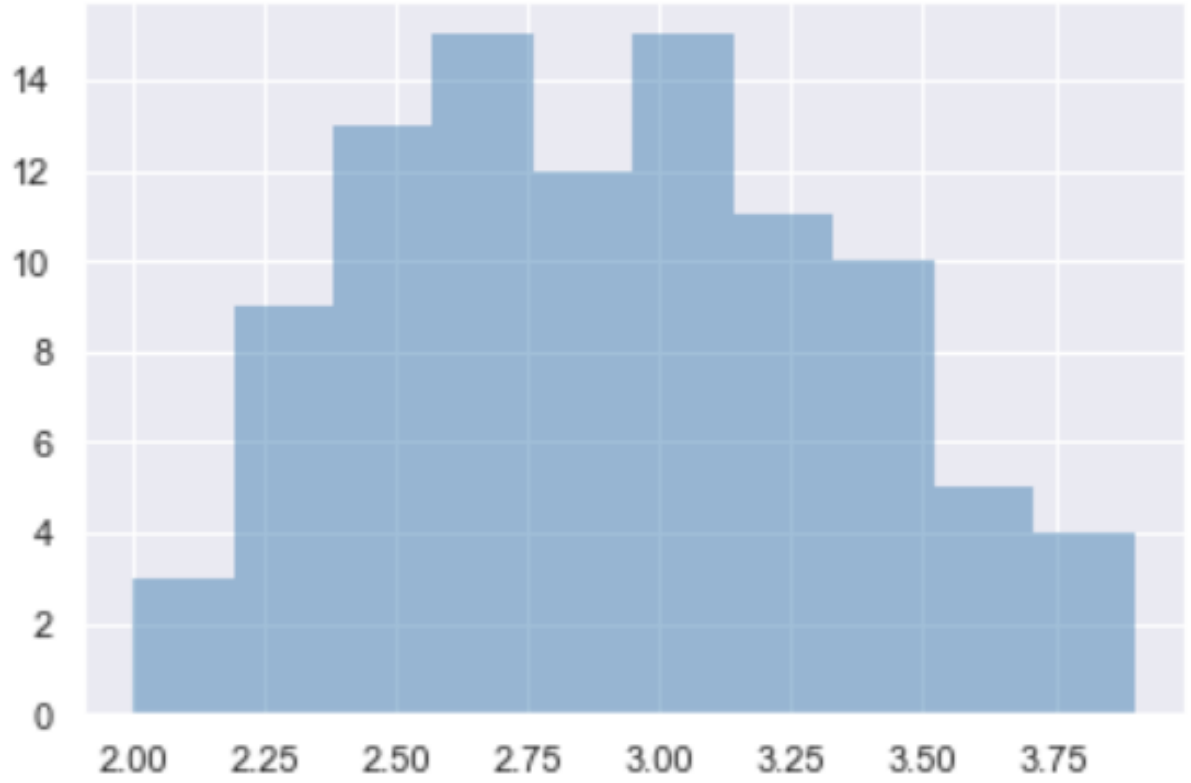
Побудова гістограми:

```
plt.hist(cats['Bwt'],color='green',edgecolor='black');
```

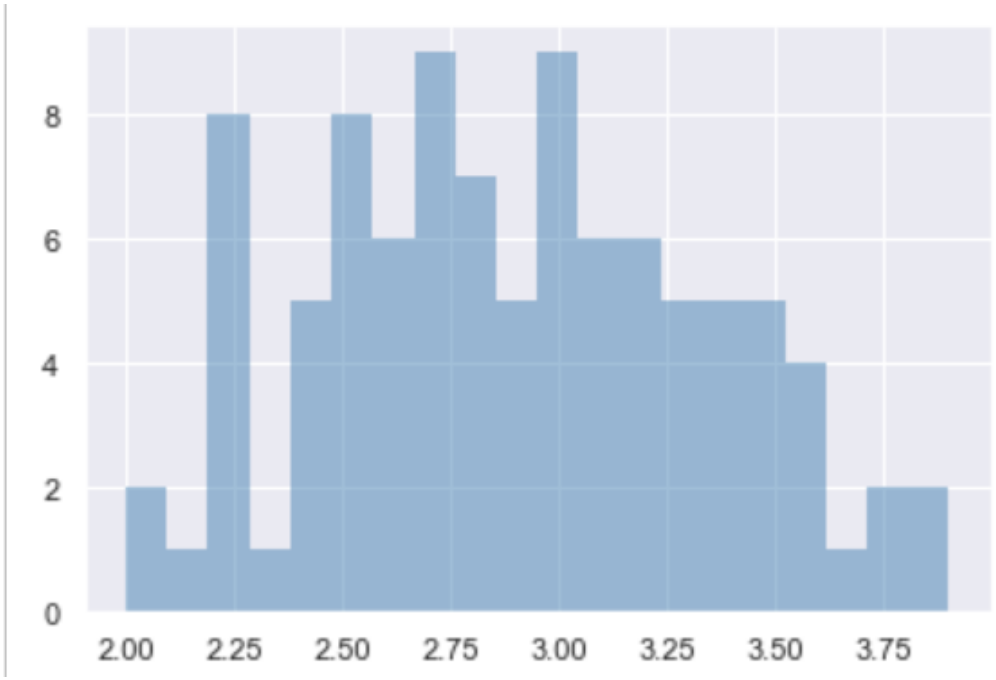


Візуалізація даних

plt.hist(cats['Bwt'],bins=10, alpha=0.5, histtype='stepfilled', color='steelblue', edgecolor='none’);

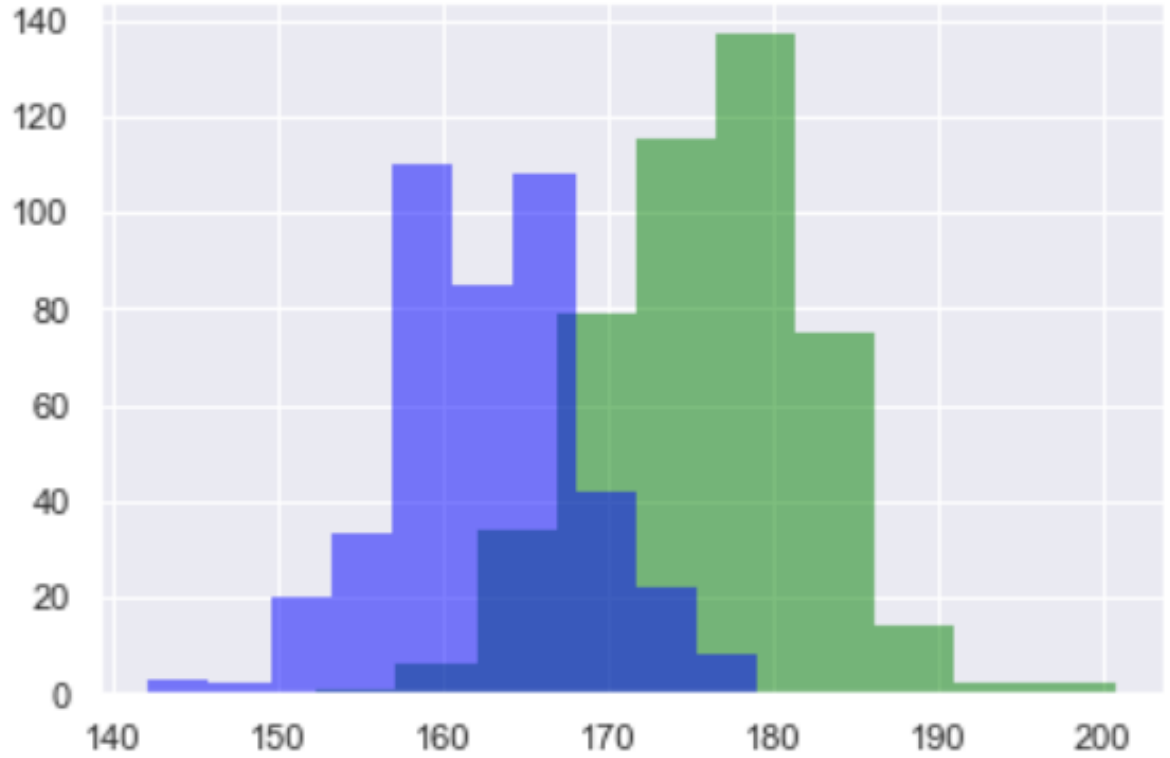


bins=20



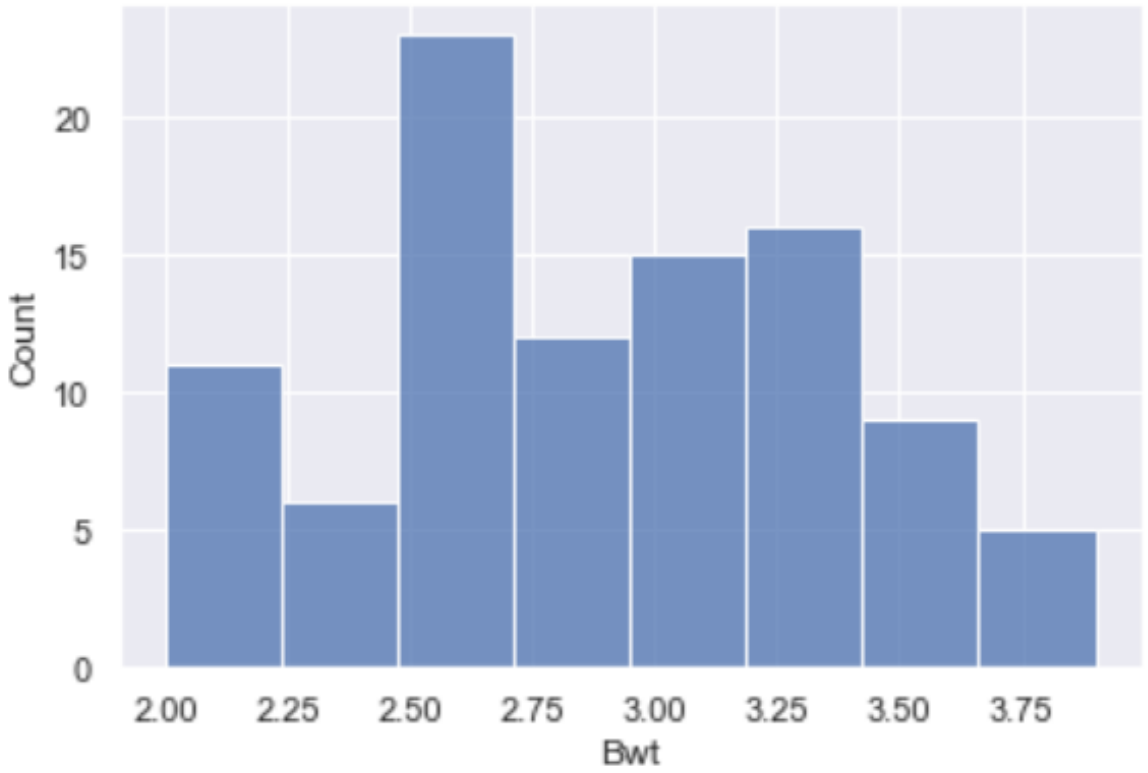
Візуалізація даних

```
heights_m=heights[(heights.Gender=='M')].Height
heights_f=heights[(heights.Gender=='F')].Height
plt.hist(heights_m,alpha=0.5,color='green',edgecolor='none',);
plt.hist(heights_f,alpha=0.5,color='blue',edgecolor='none',);
```

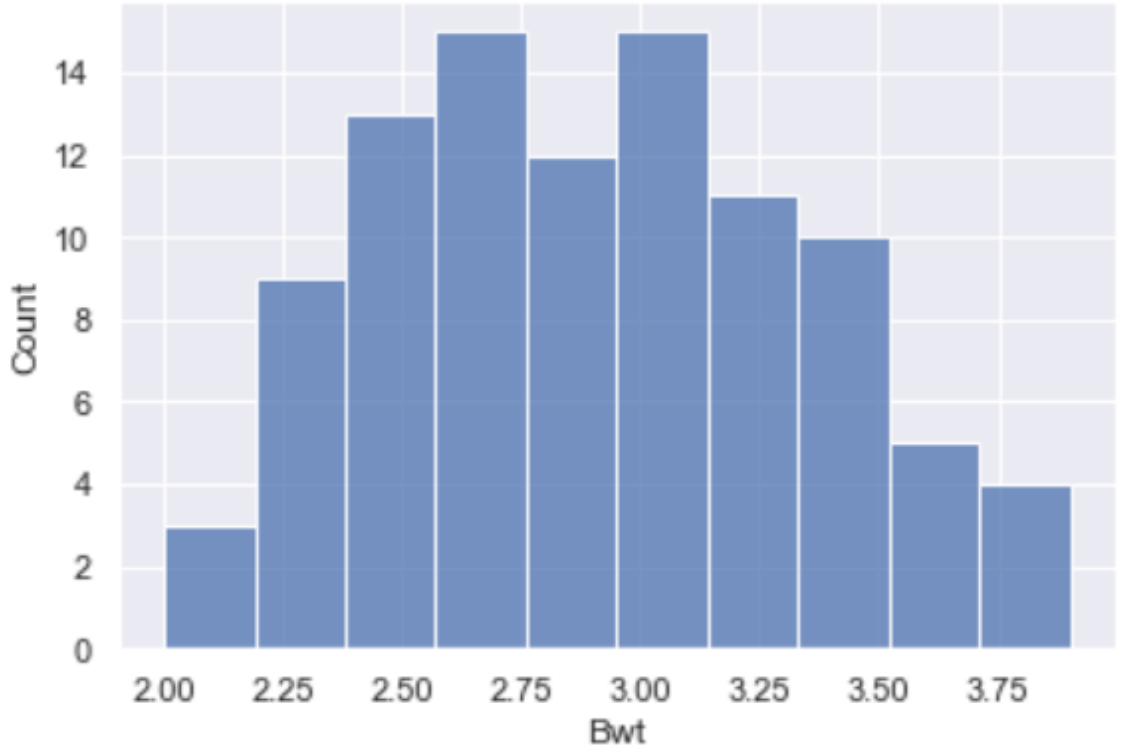


Візуалізація даних

sns.histplot(cats['Bwt']);

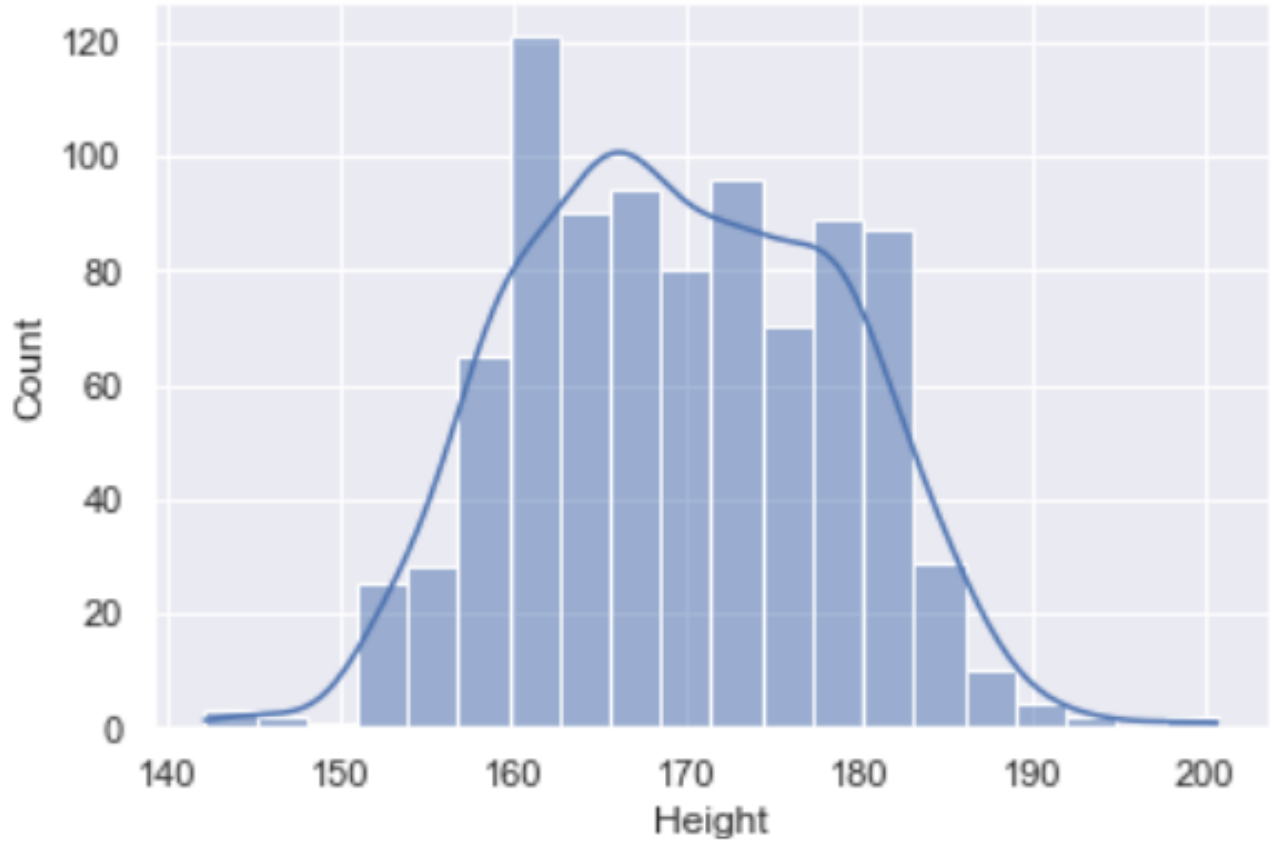


sns.histplot(cats['Bwt'],bins=10);

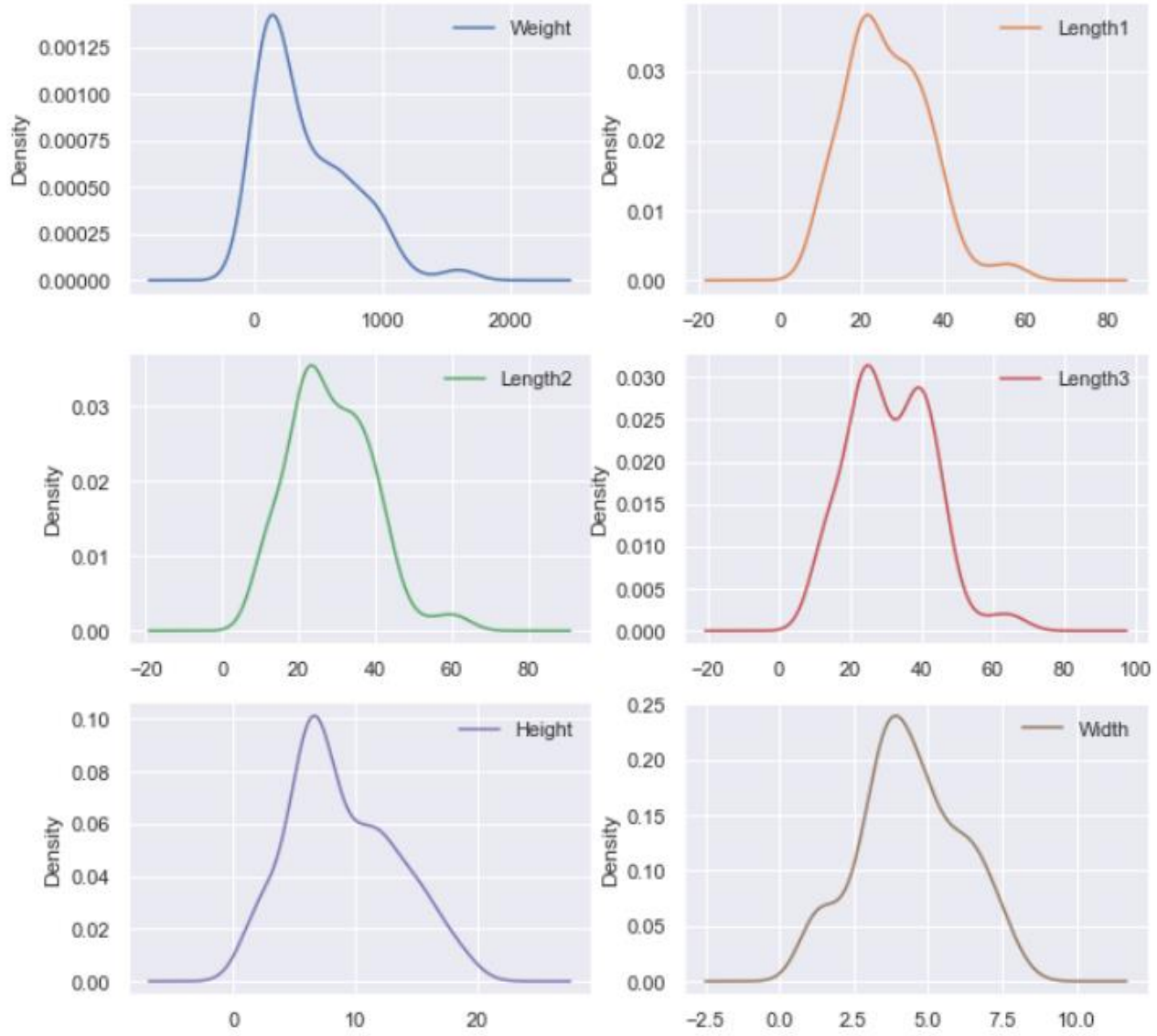


Візуалізація даних

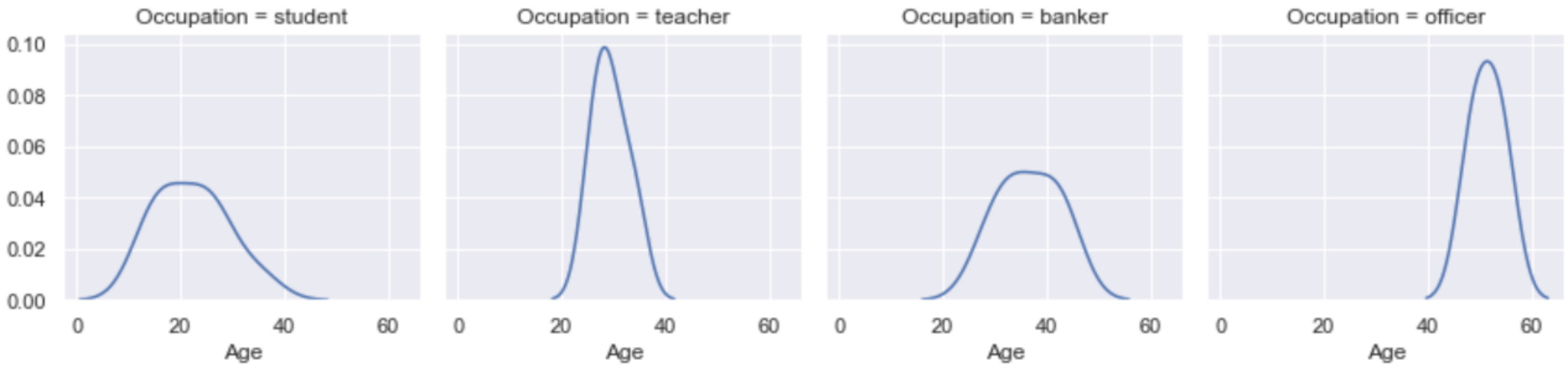
```
sns.histplot(heights['Height'], kde=True);
```



```
fish.plot(kind="density", subplots=True, layout=(3, 2), sharex=False, figsize=(10, 10));
```



```
g = sns.FacetGrid(users, col='Occupation’)
g = g.map(sns.kdeplot, 'Age')
```



Візуалізація даних

Щоб побудувати графік з matplotlib, треба:

1. Підготувати дані
2. Створити основу для графіку. Для цього також можна створити фігуру та осі.

fig, ax = plt.subplots()

3. Побудувати графік

ax.plot(x, y, color='lightblue', linewidth=3)

4. Налаштувати графік.
ax.hist(y) або plt.hist(y); - тип графіку
plt.title("Some Lines"); plt.xlabel("x"); plt.ylabel("y");
або ax.set(title=' Some Lines ', ylabel=' y’, xlabel='x')- заголовок графіку та осей.

Візуалізація даних

ax.set(xlim=[0,10.5],ylim=[-1.5,1.5]) або plt.axis([10, 0, 0, 10]); - границі осей

5. Зберегти графік

plt.savefig(‘graf.png’)

6. Показати графік

plt.show()

Візуалізація даних

Щоб побудувати графік з Seaborn, треба:

1. Підготувати дані

2. Обрати стиль

```
sns.set_style("whitegrid")
```

3. Побудувати графік

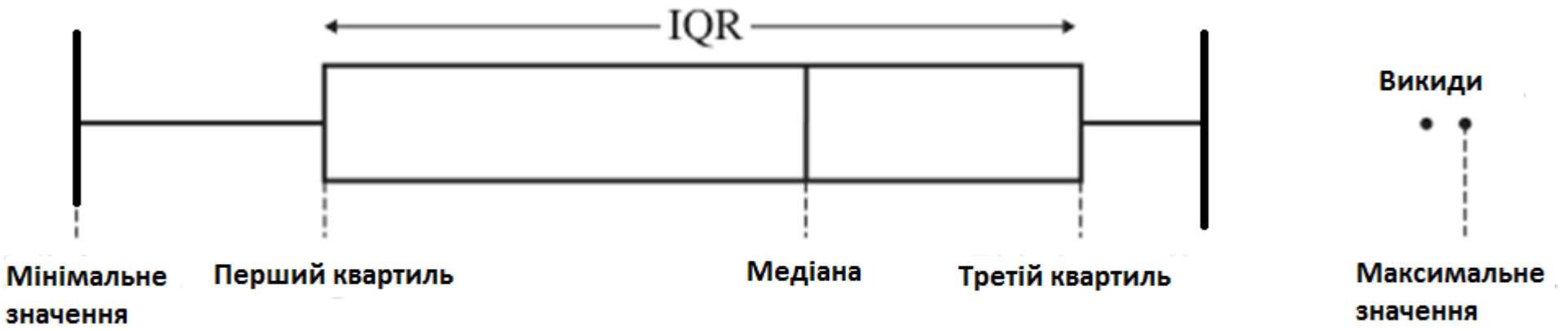
```
g=sns.histplot(cats['Bwt’]);
```

4. Налаштувати графік:

```
g.set(xlim=(1, 8));
g.set(ylim=(-50, 1200));
```

Візуалізація даних

Діаграма розмаху або коробкова діаграма - це схематичне представлення положення даних, включаючи найменші та найбільші значення, нижню та верхню чверть вибірки (нижній та верхній квартилі), медіану та статистичні викиди. Виглядає діаграма наступним чином:



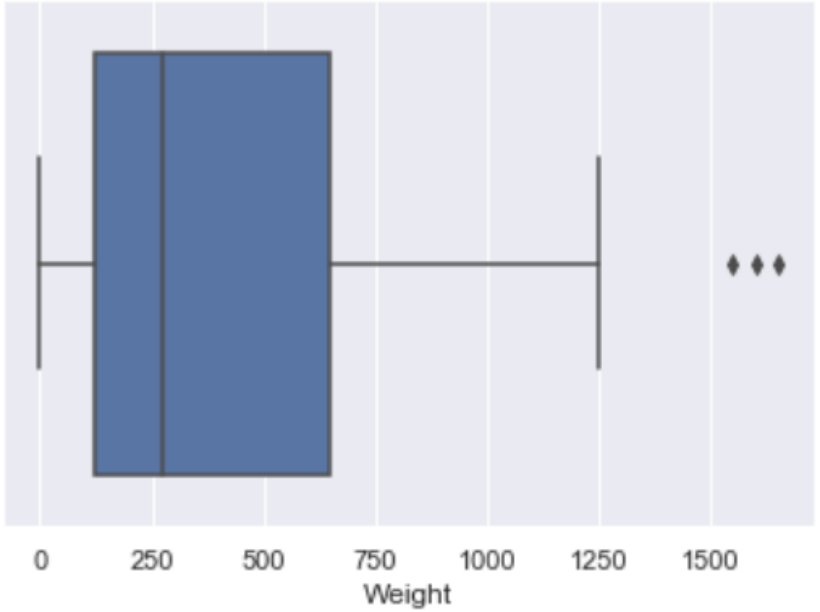
Візуалізація даних

Таким чином для побудови діаграми розмаху потрібно:

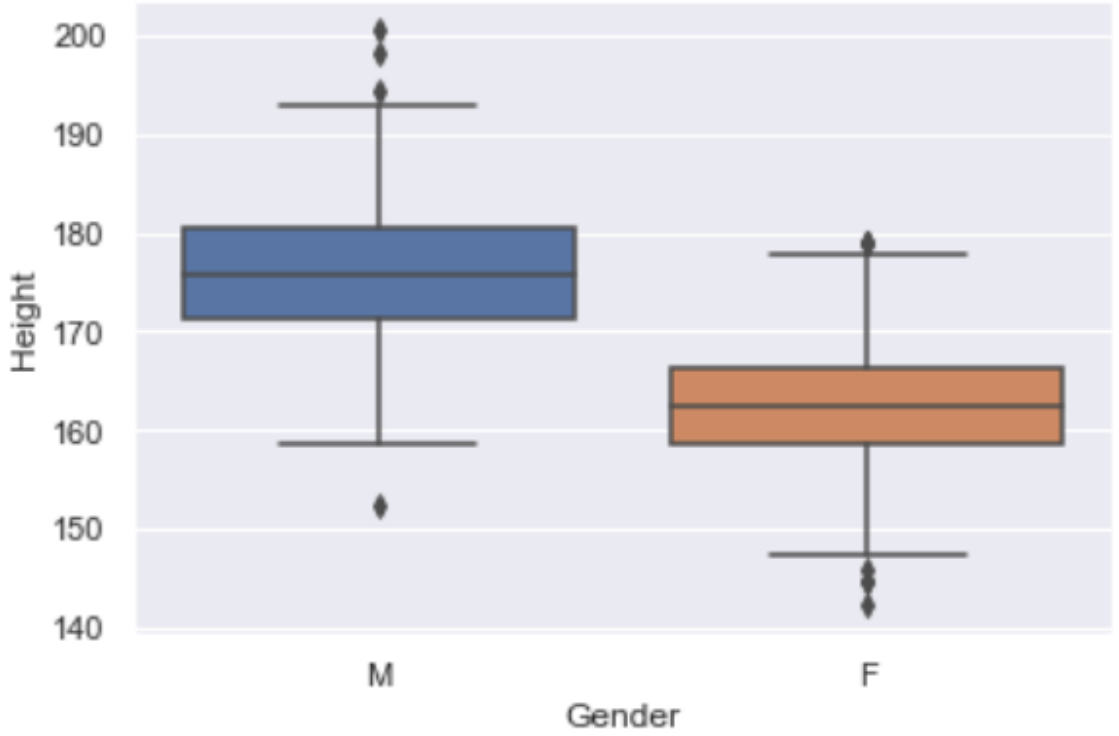
1. Відсортувати дані.
2. Визначити найбільше та найменше значення.
3. Визначити перший, другий (медіану) та третій квартилі.
4. Визначити міжквартильний розмах.
5. Обчислити Q_L –1,5 × IQR та Q_H + 1,5 × IQR. Ці значення округлюються до найближчих значеннь фактичних даних. Таким чином, фактичний початок хвоста діаграми є найнижчим значенням, більшим за (Q_L –1,5 × IQR), а фактичний кінець хвоста - це найвище значення, яке менше за Q_H + 1,5 × IQR.

Візуалізація даних

```
sns.boxplot(x='Weight', data=fish);
```

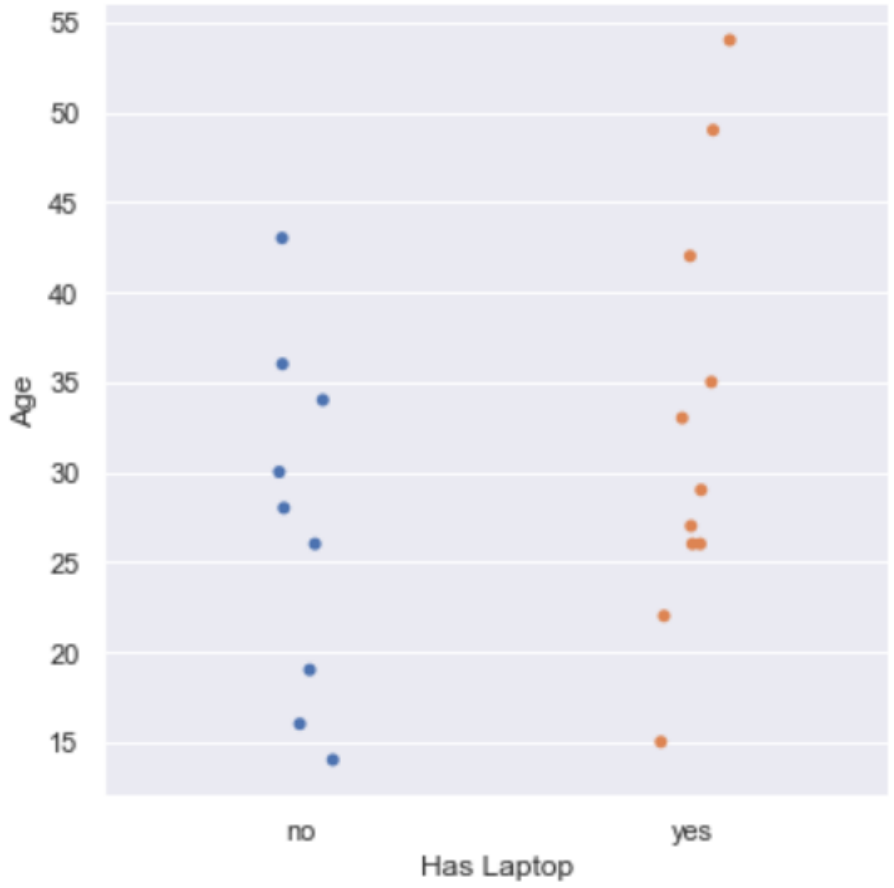



```
sns.boxplot(x='Gender’,
 y='Height',data=heights);
```

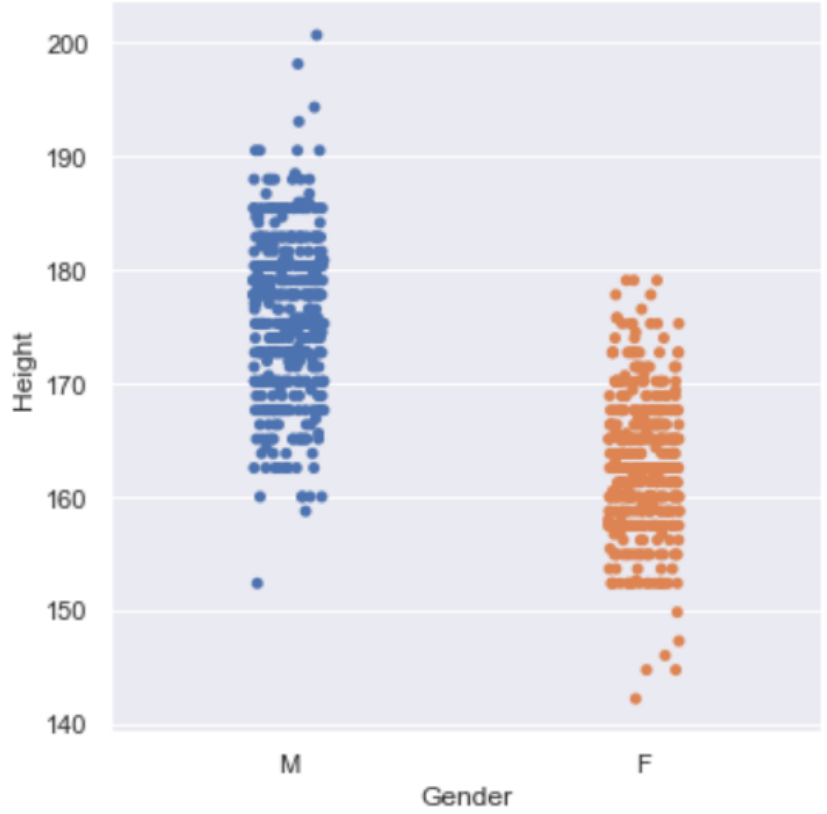


Візуалізація даних

```
sns.catplot(x='Has Laptop', y='Age', data=users);
```

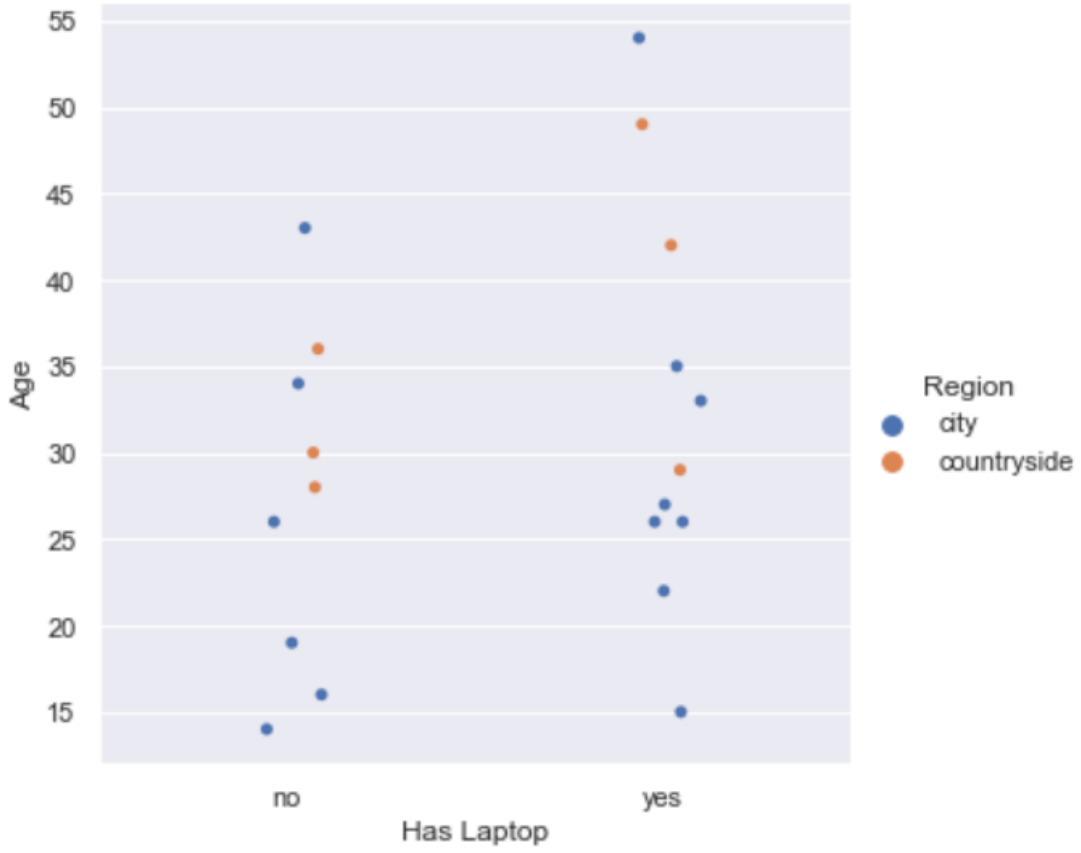



```
sns.catplot(x='Gender’,
 y='Height', data=heights);
```



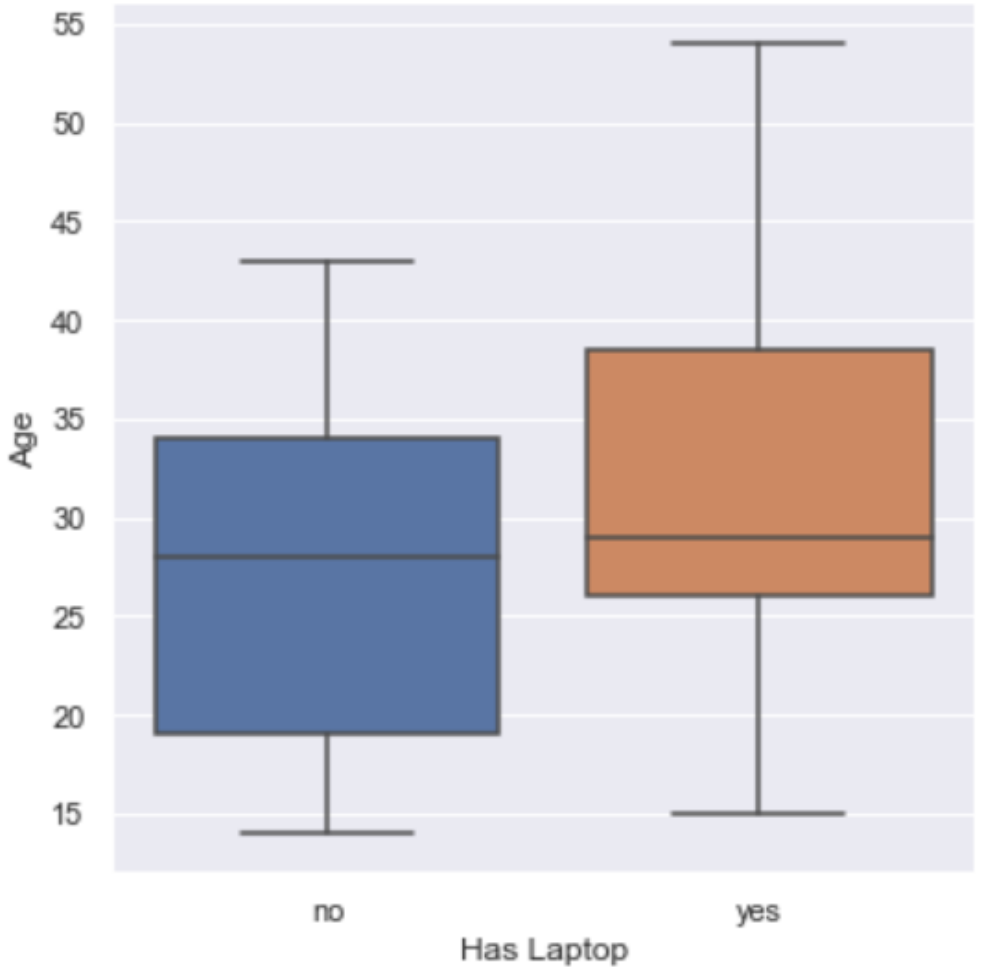
Візуалізація даних

```
sns.catplot(x='Has Laptop', y='Age', hue='Region', data=users);
```



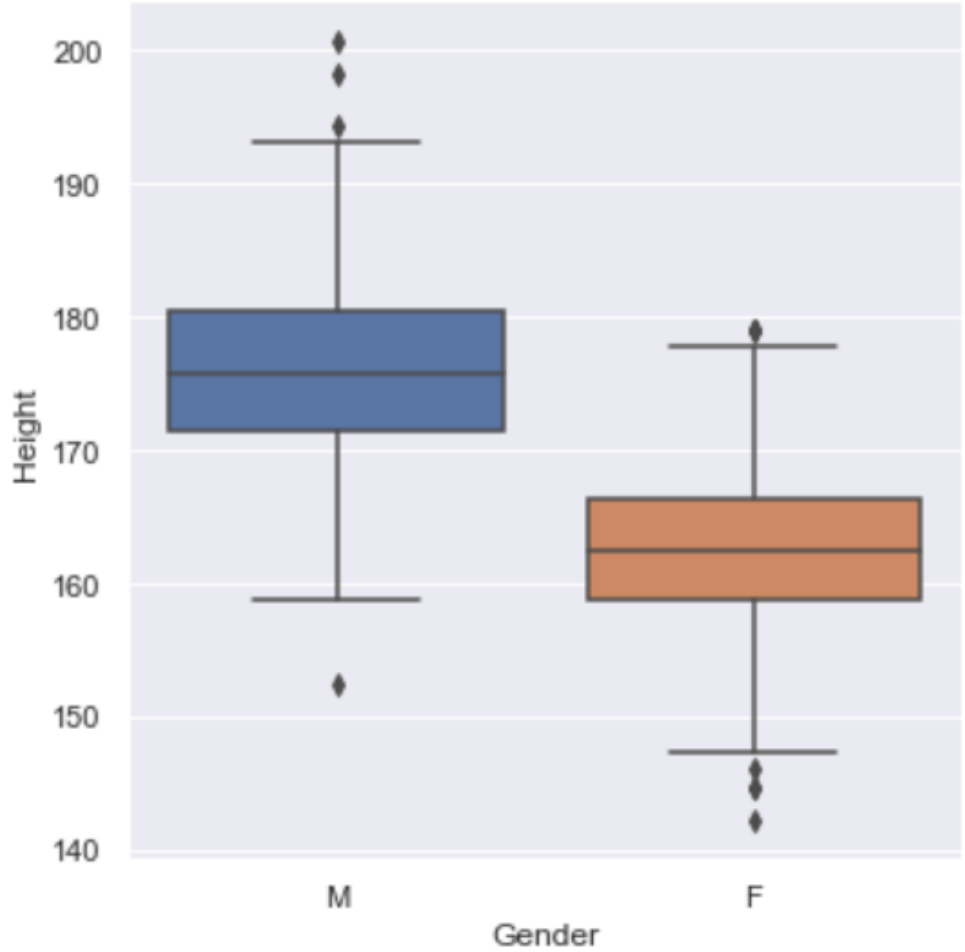
Візуалізація даних

```
sns.catplot(x='Has Laptop', y='Age',kind="box", data=users);
```



Візуалізація даних

```
sns.catplot(x='Gender', y='Height', kind="box", data=heights);
```

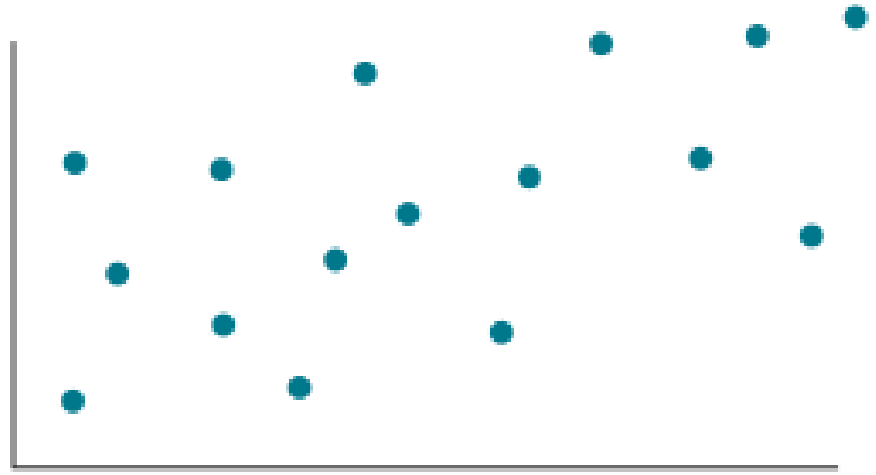


Візуалізація даних

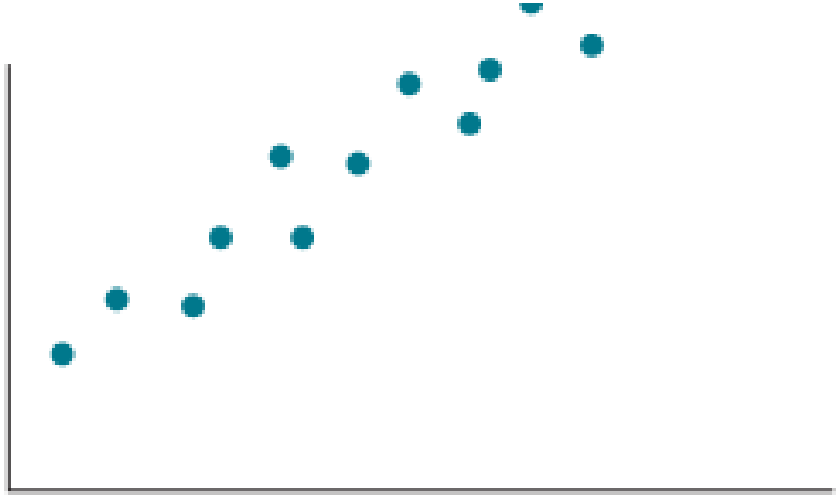
В багатьох випадках доводиться працювати з даними, що мають багатовимірний характер. Тобто кожне спостереження складається з вимірювань декількох змінних.

Діаграма розсіювання або точкова діаграма часто використовується для графічного відображення потенційного зв’язку між парою змінних. Значення однієї змінної відкладаються на осі X, другою – на осі Y. Якщо спостереження має більше двох змінних, то використовується декілька діаграм розсіювання, що зображають зв’язок між кожною парою цих змінних.

Візуалізація даних

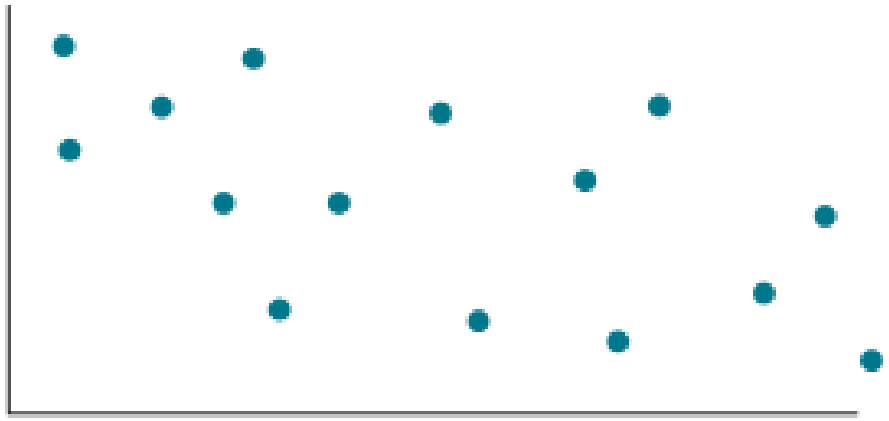


Слабка позитивна залежність

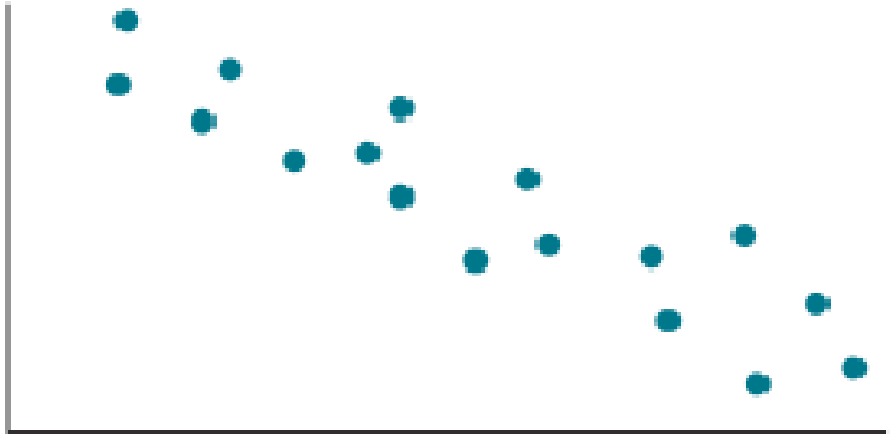


Сильна позитивна залежність

Візуалізація даних

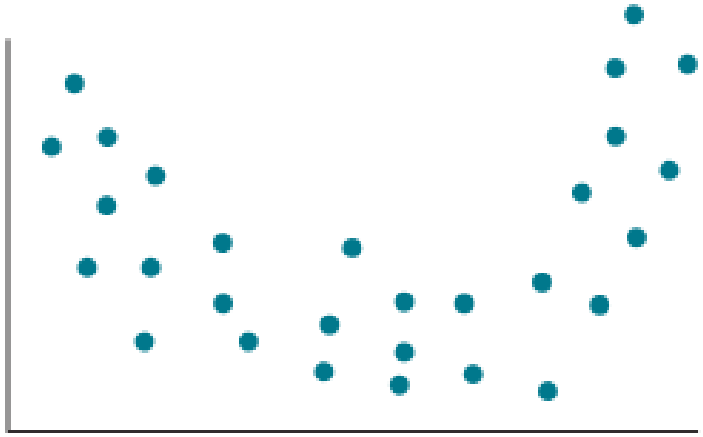


Слабка негативна залежність

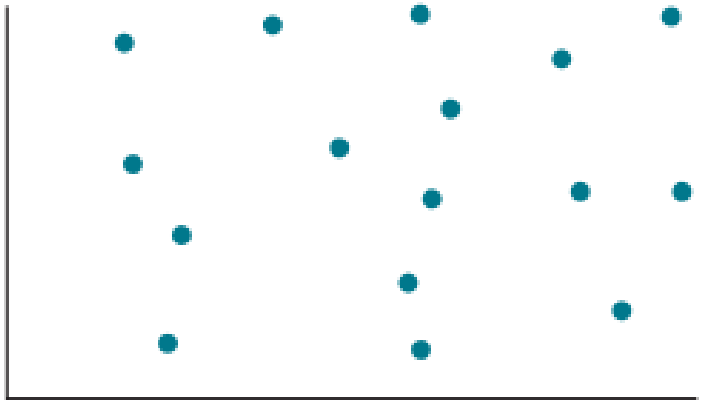


Сильна негативна залежність

Візуалізація даних



Нелінійна квадратична залежність

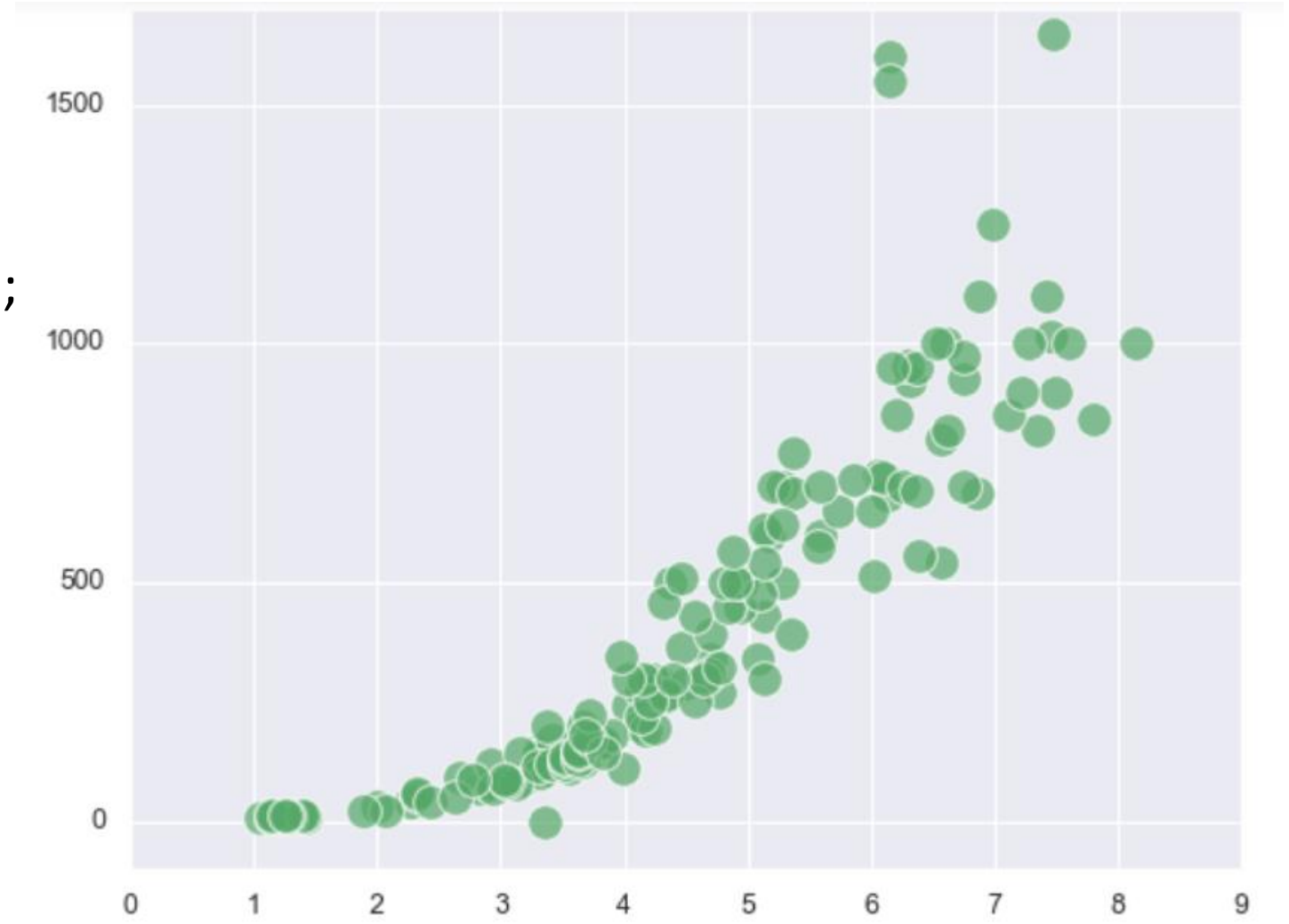


Залежність відсутня

Візуалізація даних

Діаграма розсіювання:

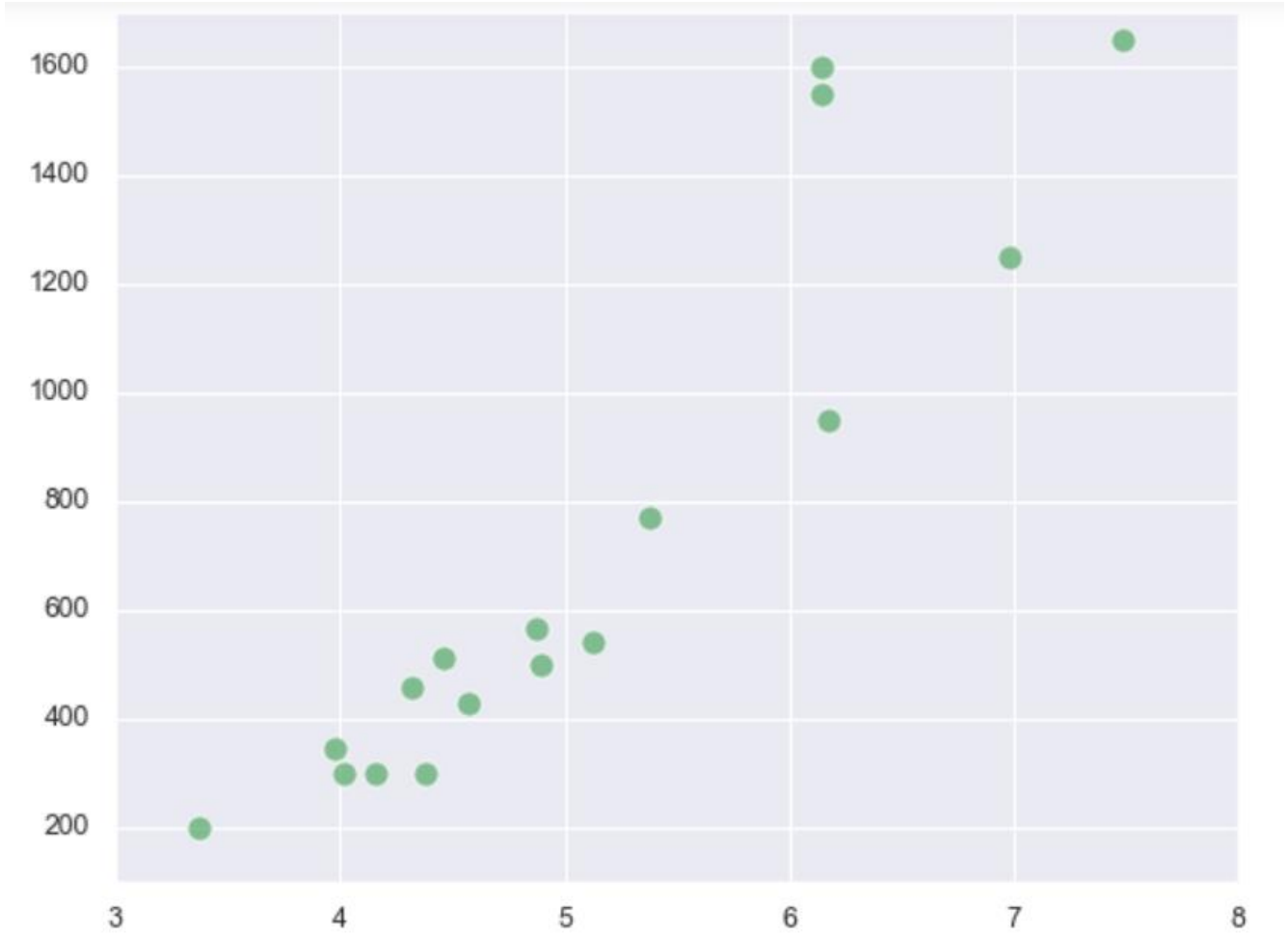
```
fish = pd.read_csv('Fish.csv')
fish=fish.set_index(['Species'])
plt.scatter(fish['Width’],
fish['Weight'],c='g',s=200,alpha=0.7);
plt.axis([0, 9, -100, 1700]);
```



Візуалізація даних

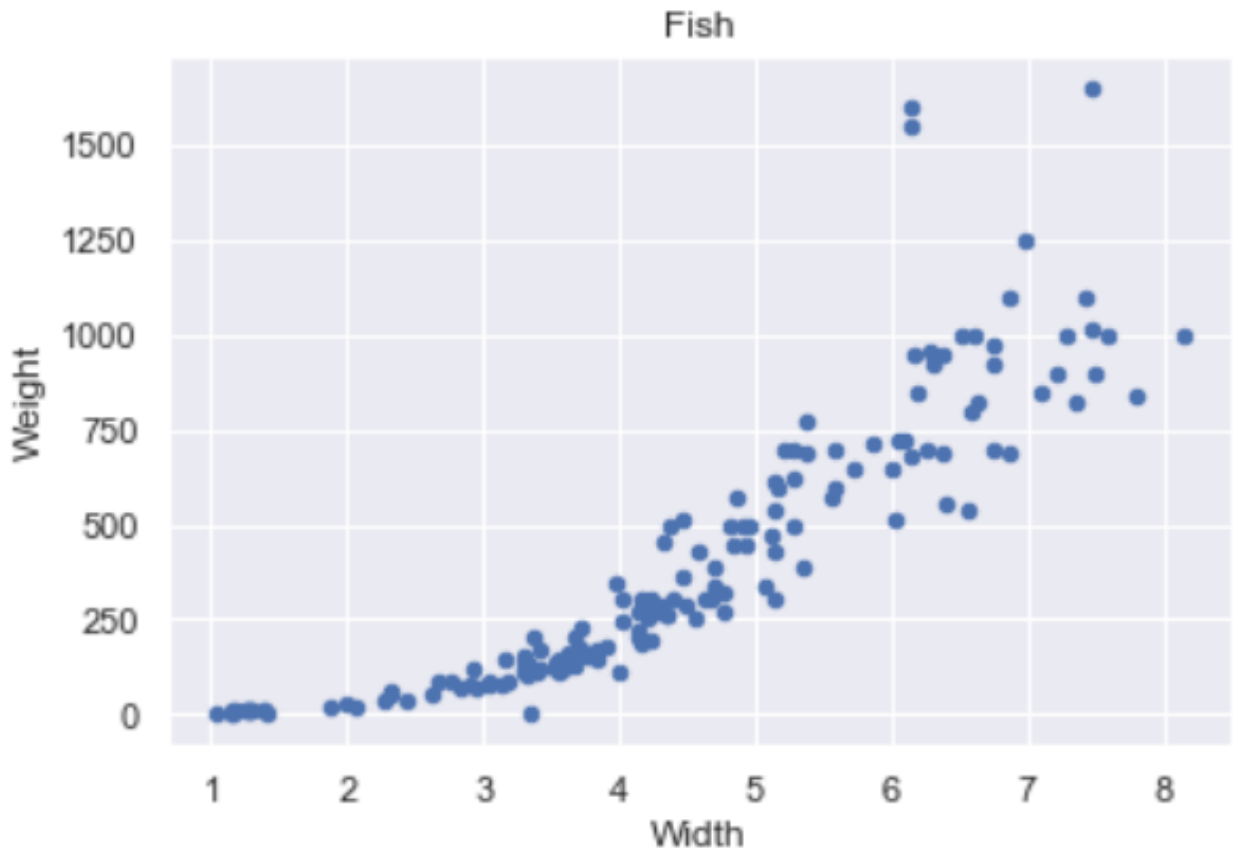
Діаграма розсіювання:

```
plt.scatter(fish.loc['Pike','Width’],
fish.loc['Pike','Weight'],c='g',s=100,alpha=0.7);
plt.axis([3, 8, 100, 1700]);
```



Візуалізація даних

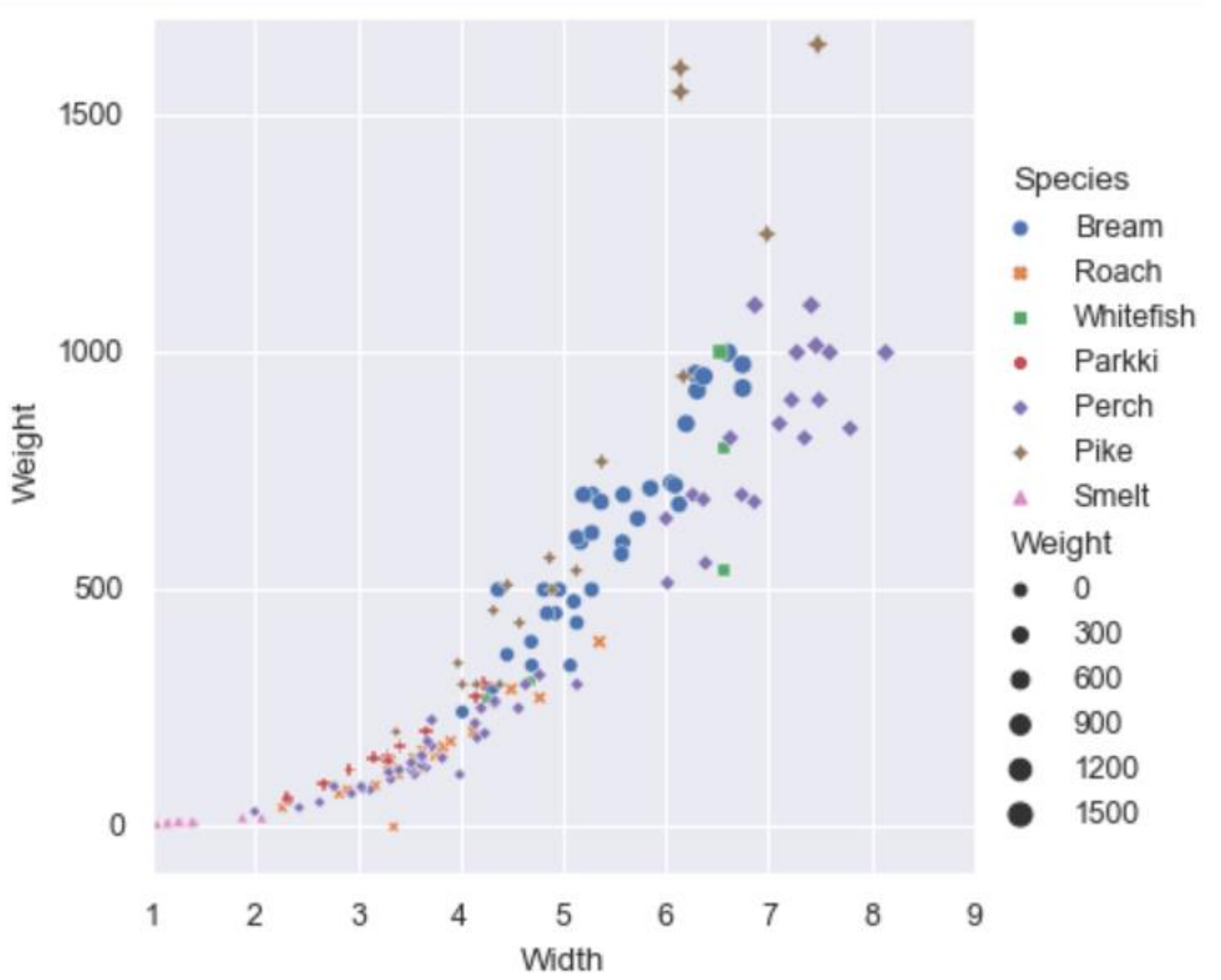
```
fish.plot.scatter(x='Width', y='Weight', title='Fish');
```



Візуалізація даних

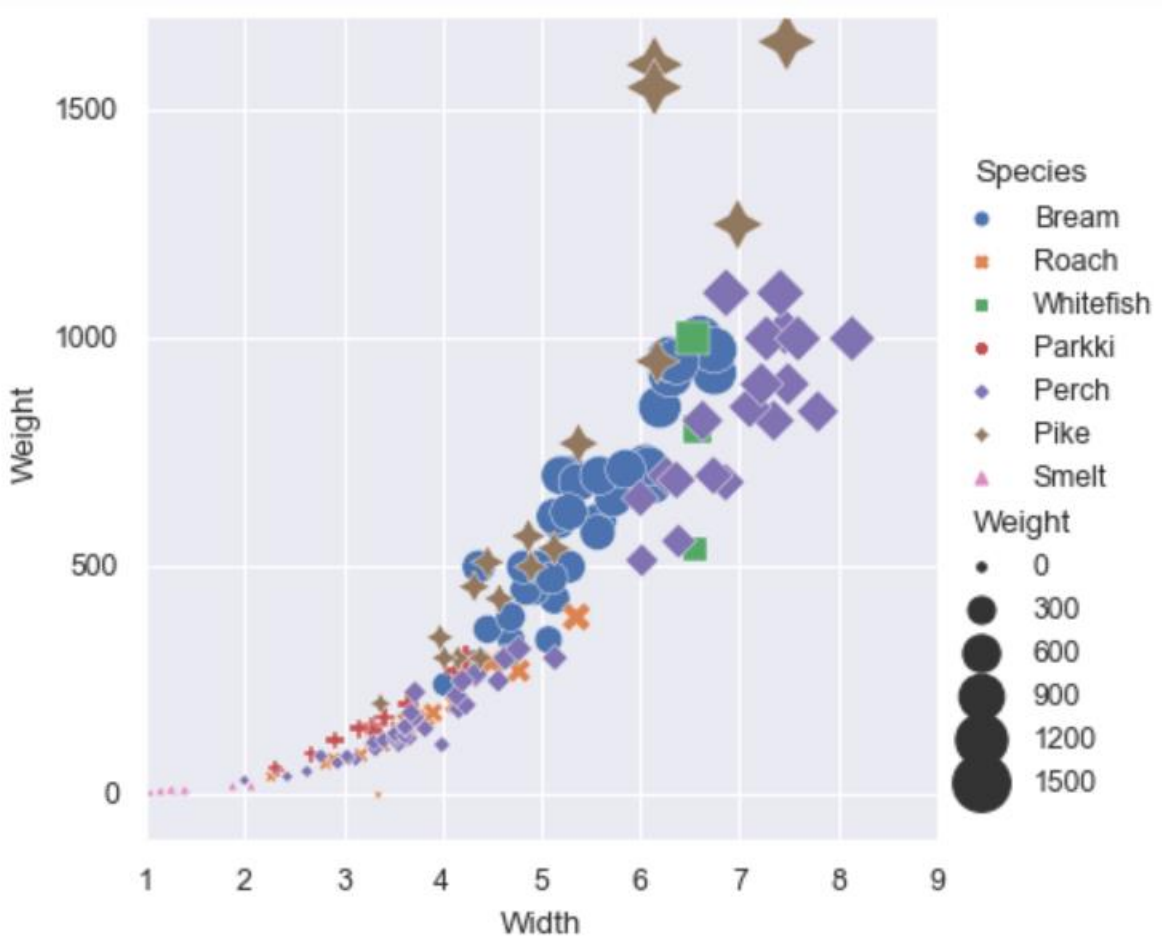
Діаграма розсіювання:

```
fishplot=sns.relplot(data=fish, x="Width", y="Weight", hue="Species",
style="Species", size="Weight");
fishplot.set(xlim=(1, 9));
fishplot.set(ylim=(-100, 1700));
```



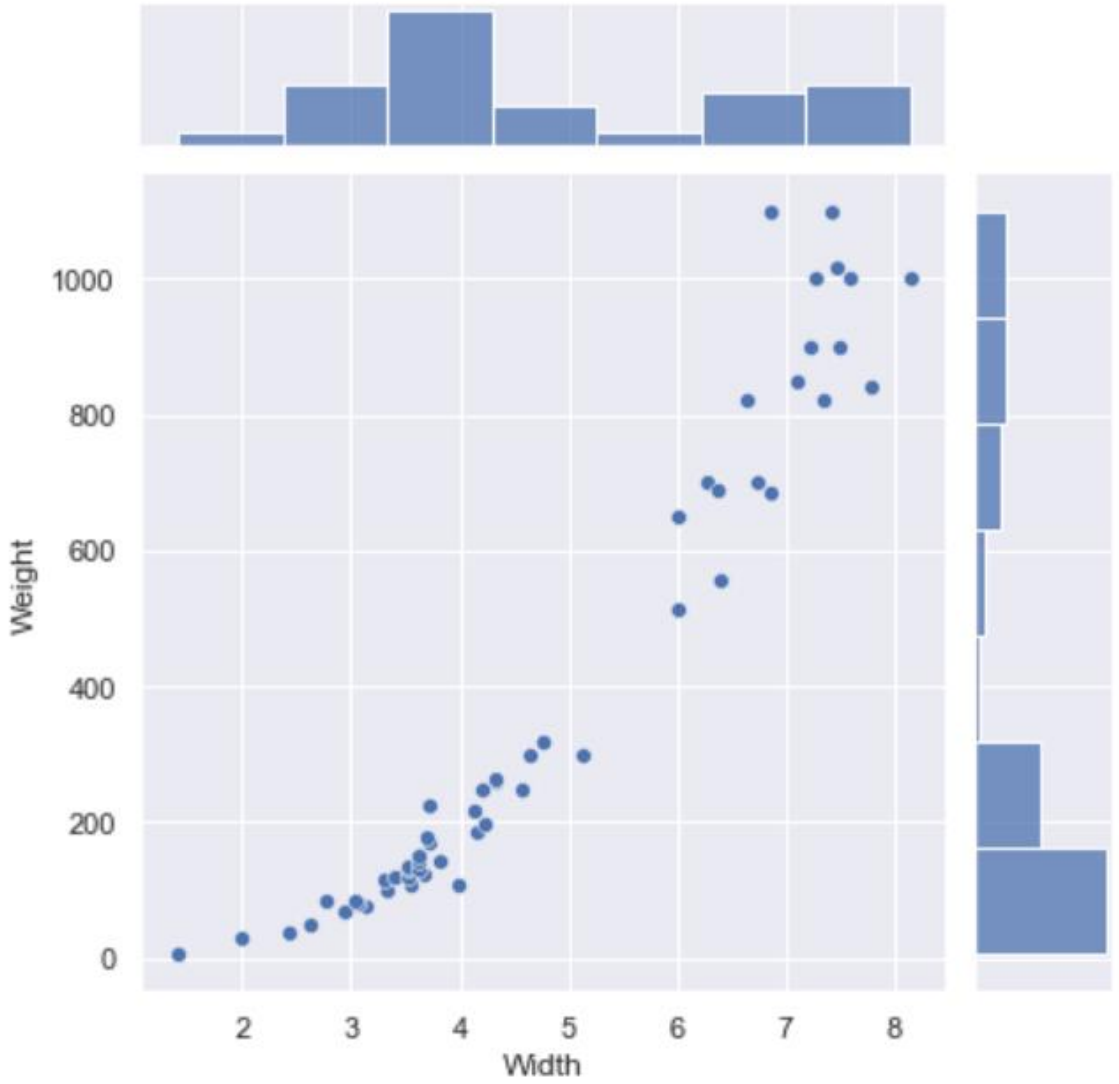
Візуалізація даних

Діаграма розсіювання:
fishplot=sns.relplot(x="Width", y="Weight", hue="Species", style="Species", size="Weight", sizes=(10, 500), data=fish);



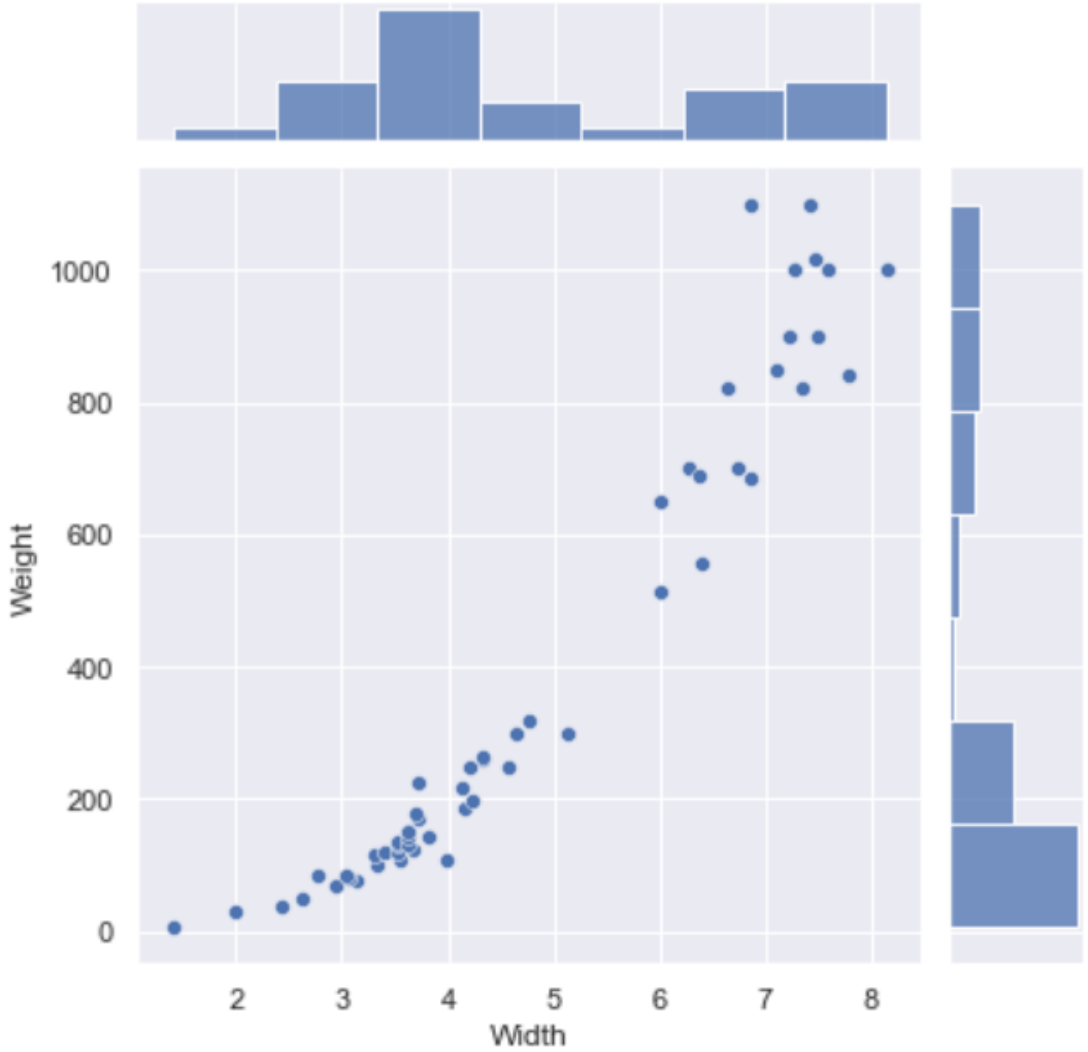
Візуалізація даних

```
sns.jointplot(x='Width', y='Weight', data=fish, kind="scatter");
```



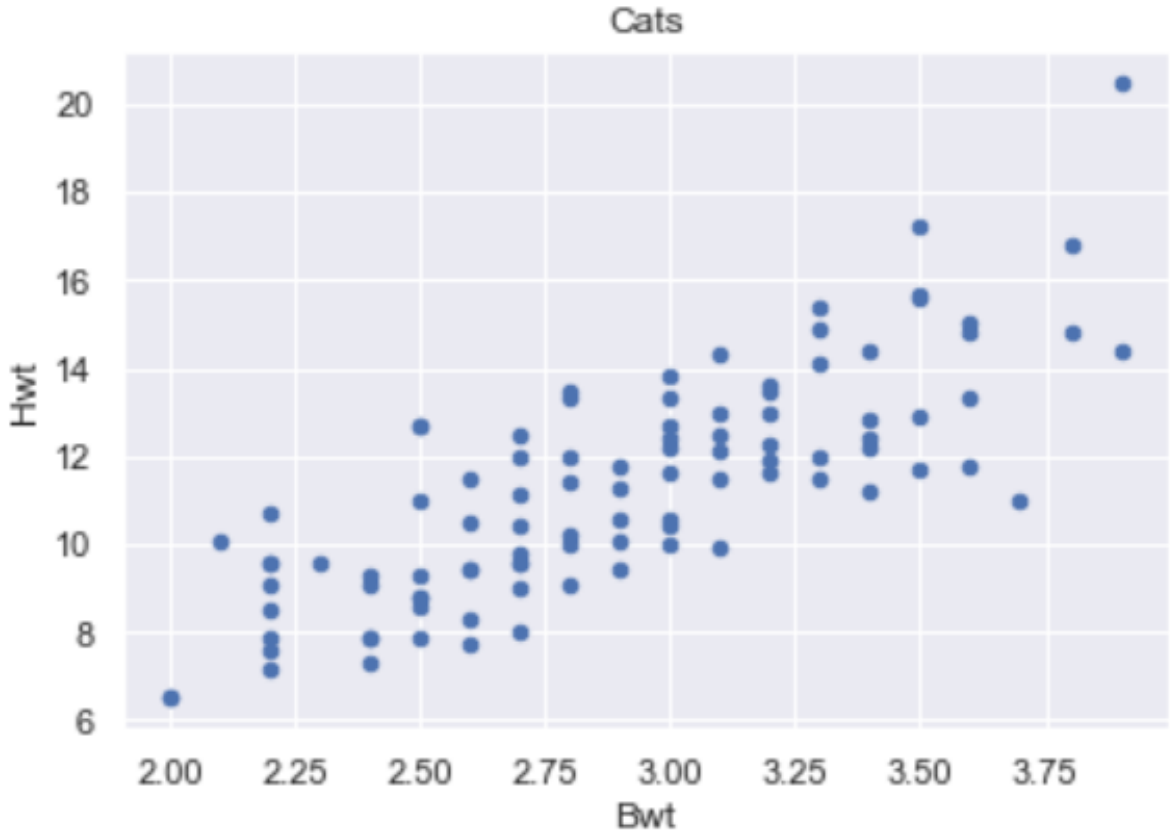
Візуалізація даних

```
sns.jointplot(x='Width', y='Weight', data=fish.loc['Perch'], kind="scatter");
```



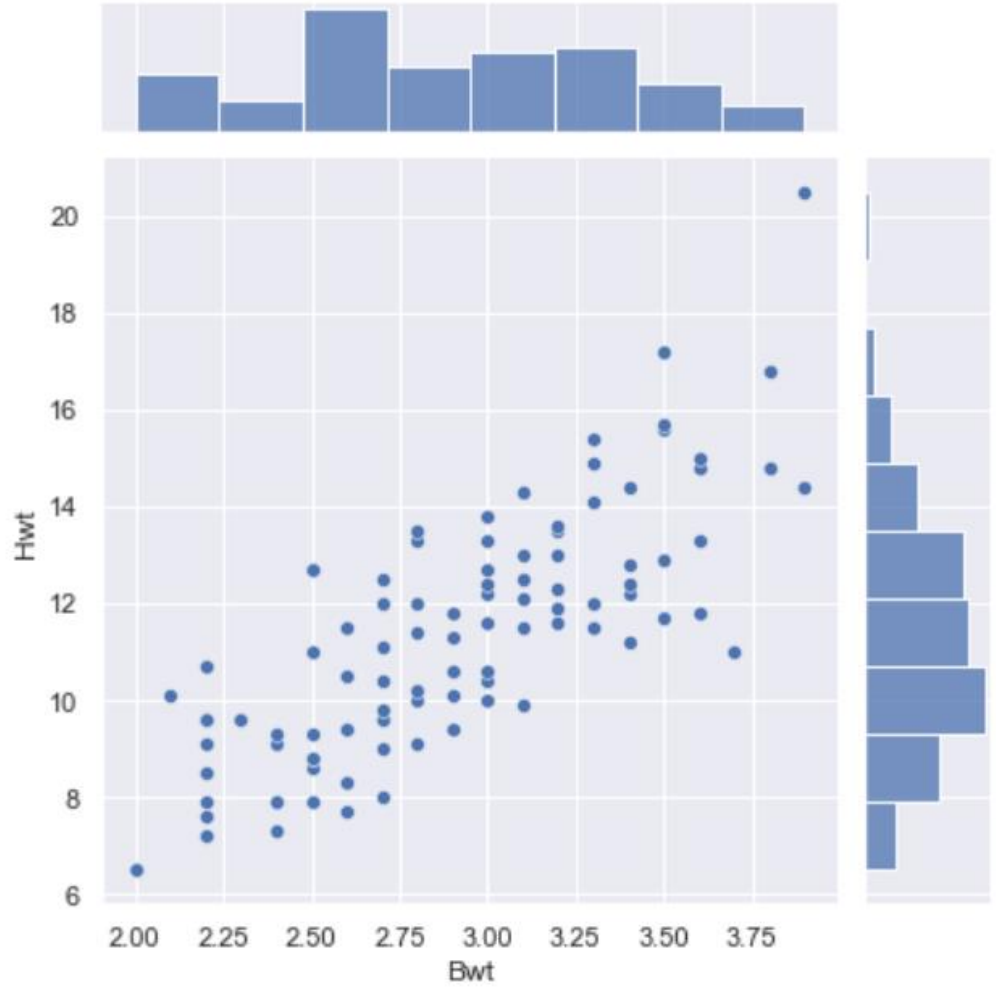
Візуалізація даних

```
cats.plot.scatter(x='Bwt', y='Hwt', title='Cats');
```



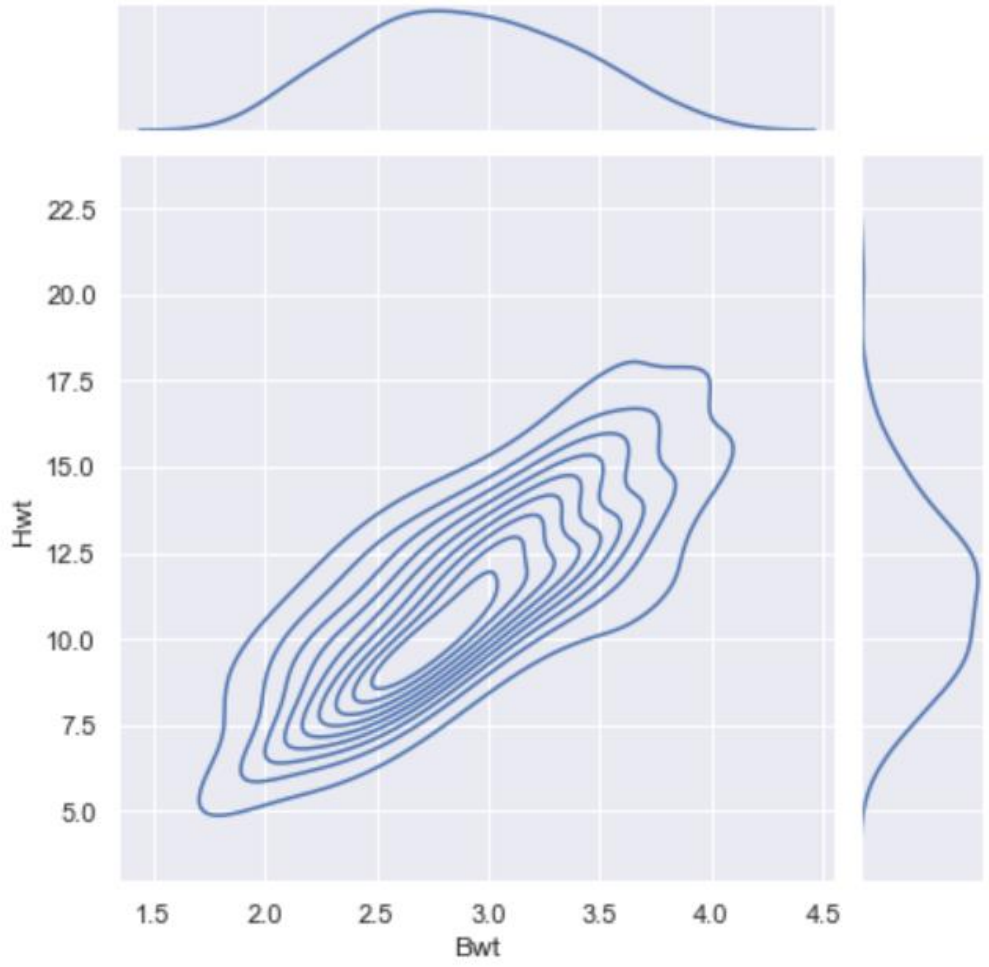
Візуалізація даних

```
sns.jointplot(x='Bwt', y='Hwt', data=cats, kind="scatter");
```



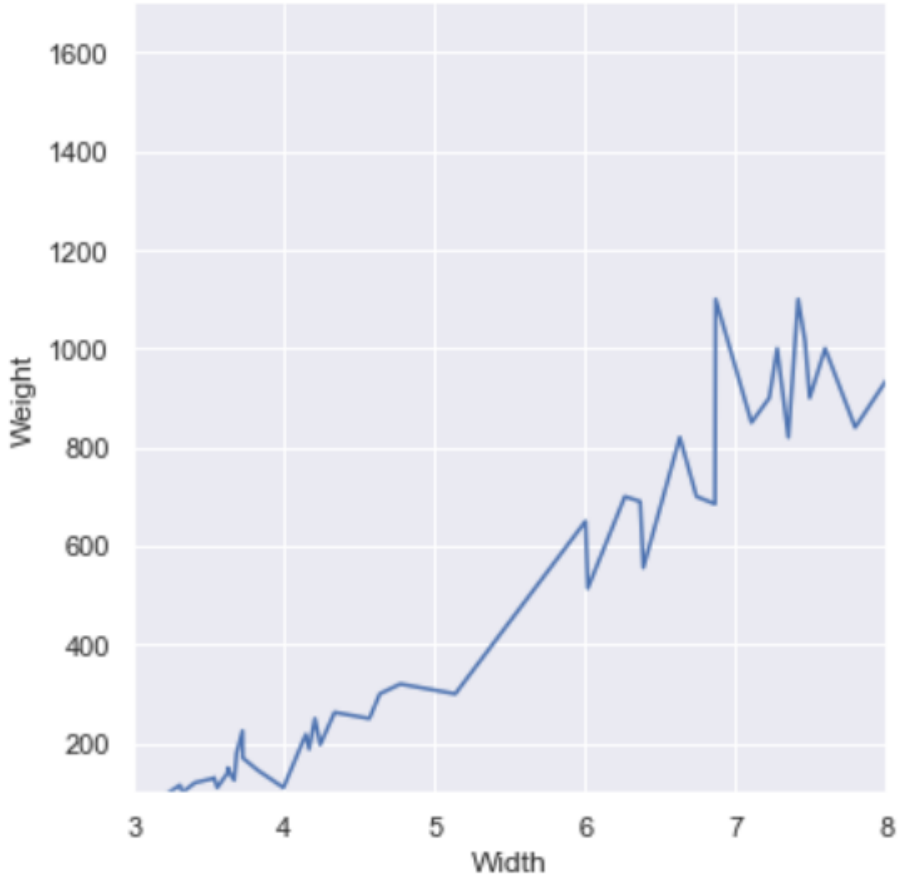
Візуалізація даних

```
sns.jointplot(x='Bwt', y='Hwt', data=cats, kind="kde");
```



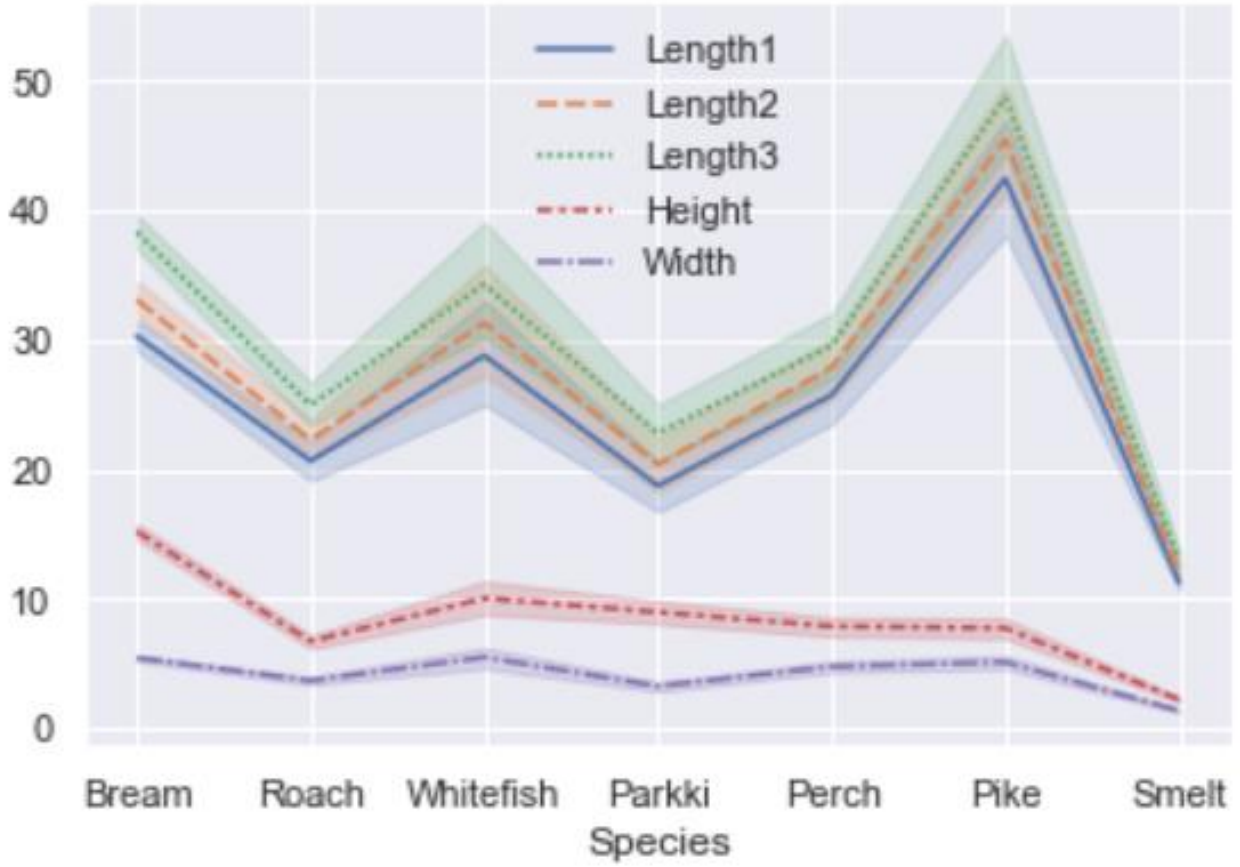
Візуалізація даних

```
fishplot=sns.relplot(data=fish.loc['Perch'], x="Width", y="Weight", kind="line");
fishplot.set(xlim=(3, 8));
fishplot.set(ylim=(100, 1700));
```



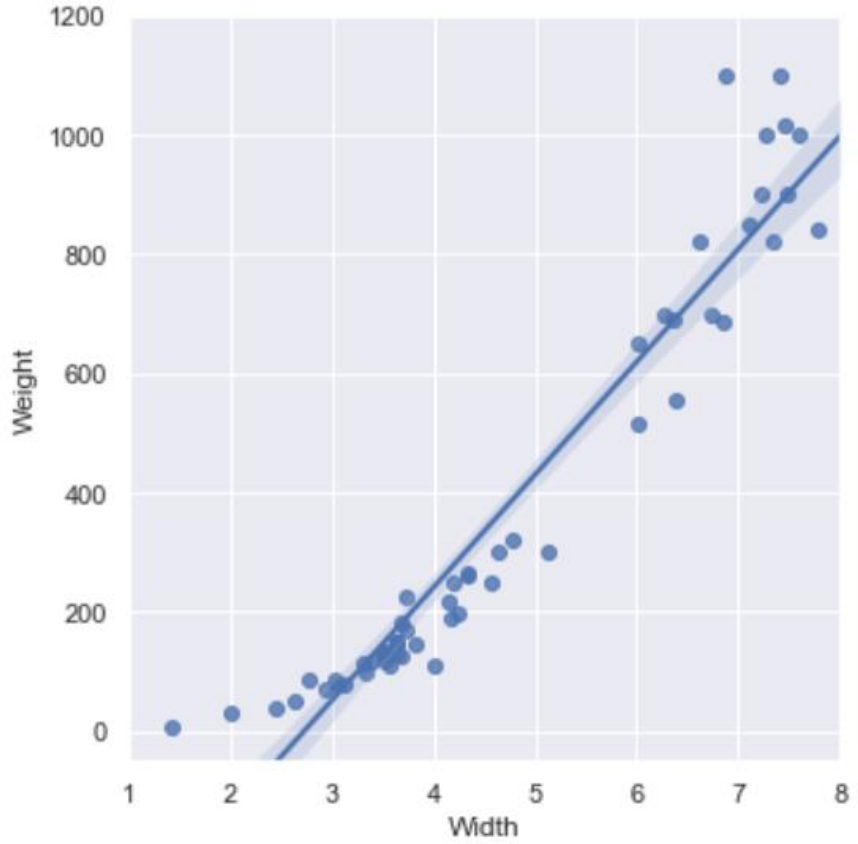
Візуалізація даних

```
sns.lineplot(data=fish.drop(['Weight'],axis=1));
```



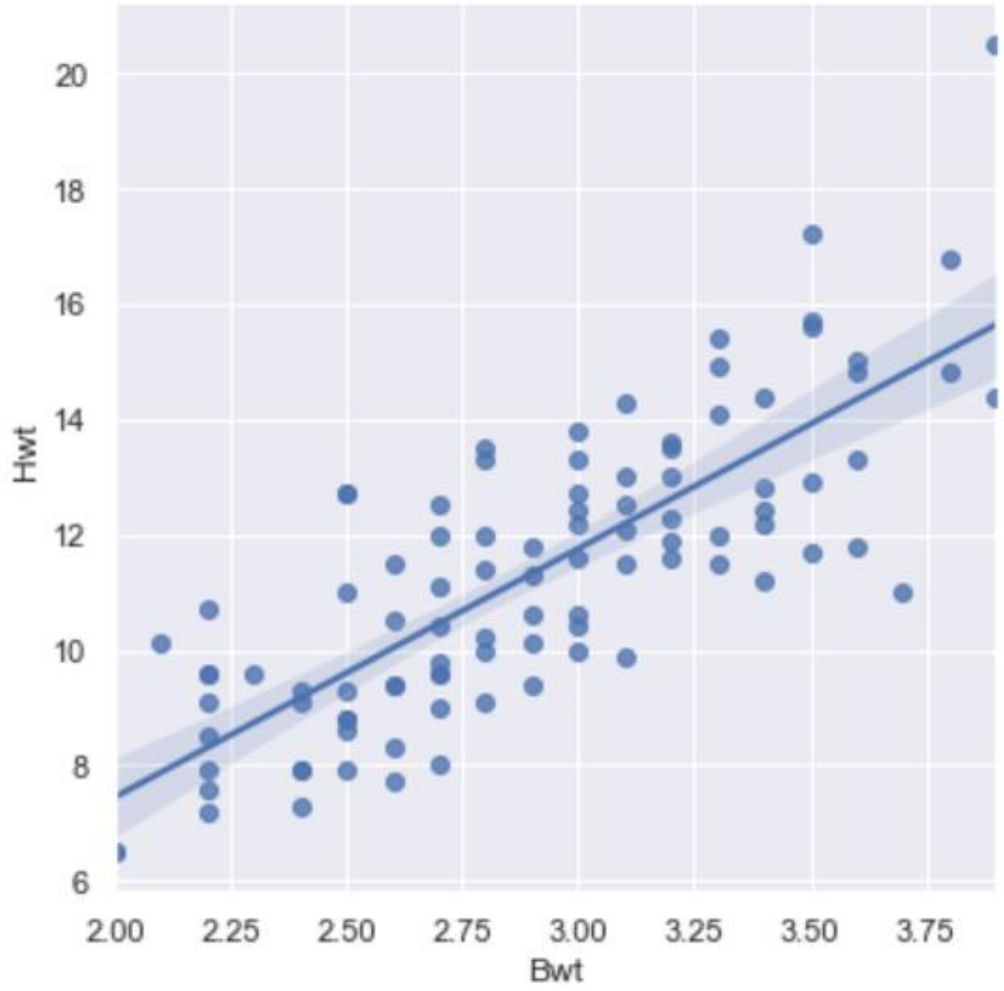
Візуалізація даних

```
fishplot=sns.lmplot(data=fish.loc['Perch'], x="Width", y="Weight");
fishplot.set(xlim=(1, 8));
fishplot.set(ylim=(-50, 1200));
```

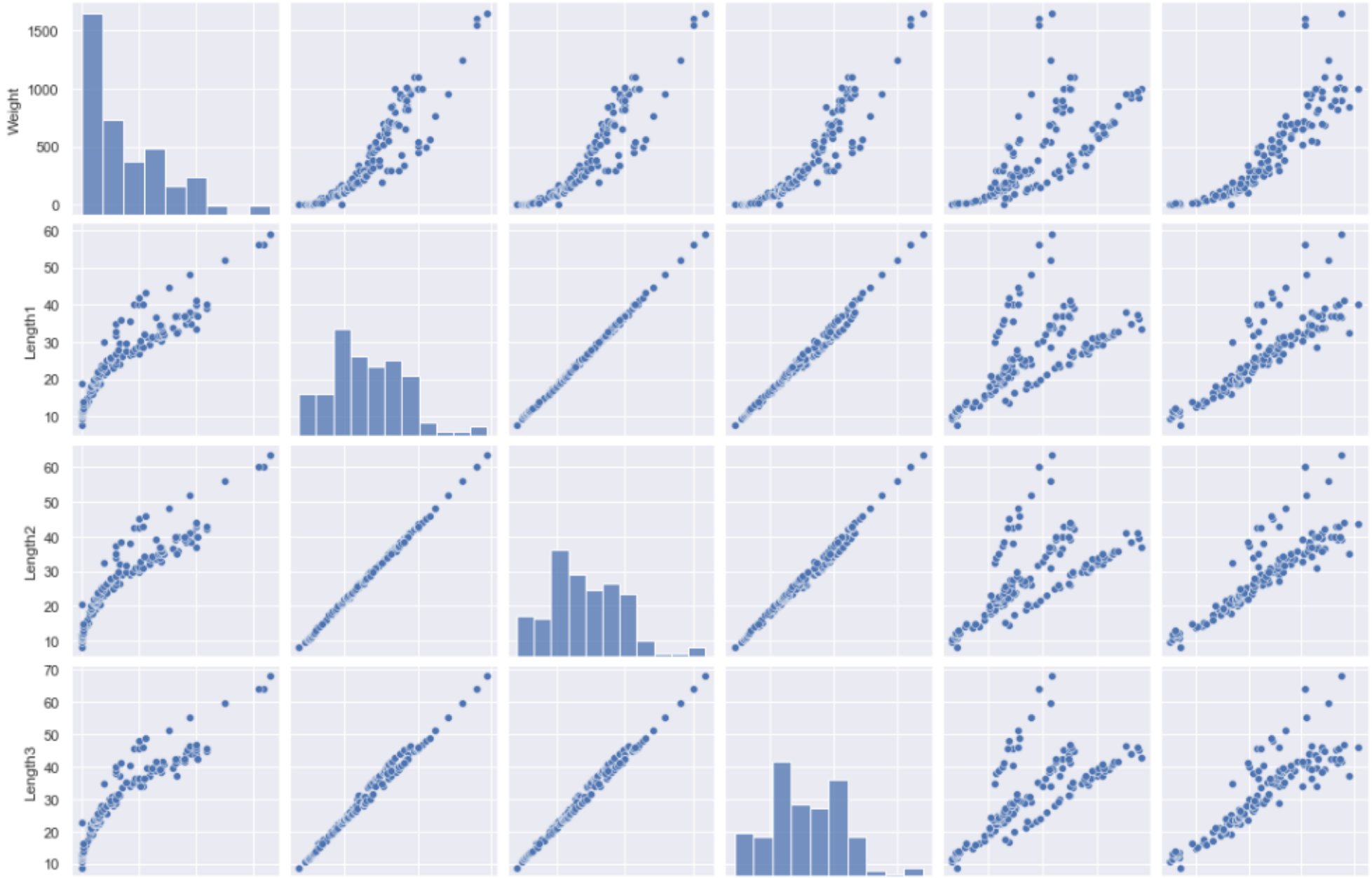


Візуалізація даних

```
sns.lmplot(data=cats, x="Bwt", y="Hwt");
```



sns.pairplot(fish);

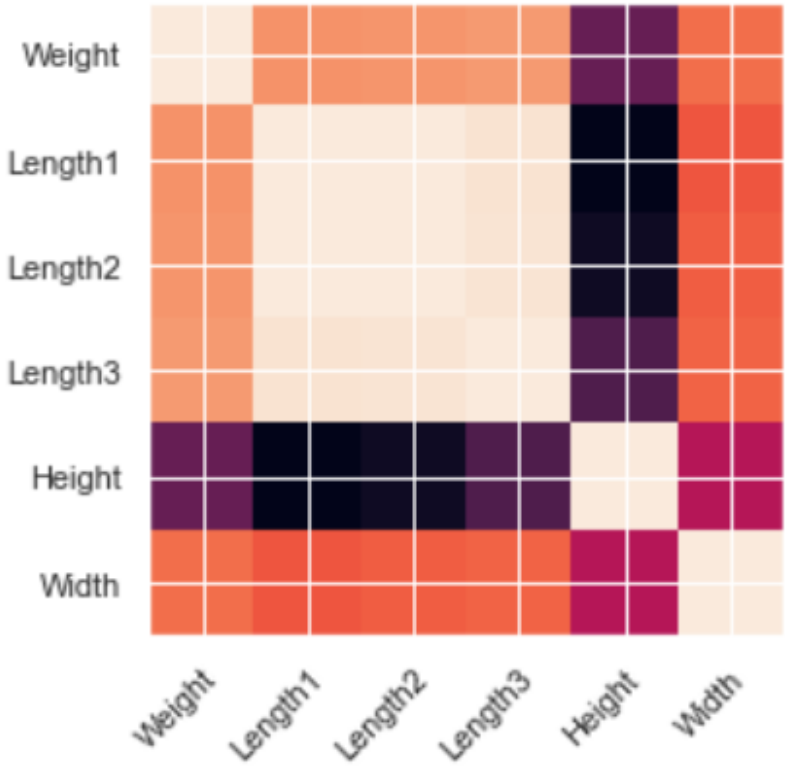


Візуалізація даних

В загальному випадку теплокарти — це графічне представлення даних різними кольорами в залежності від значень. Нерідко їх використовують для представлення кореляційної матриці, візуалізуючи ступінь зв’язку між різними змінними.

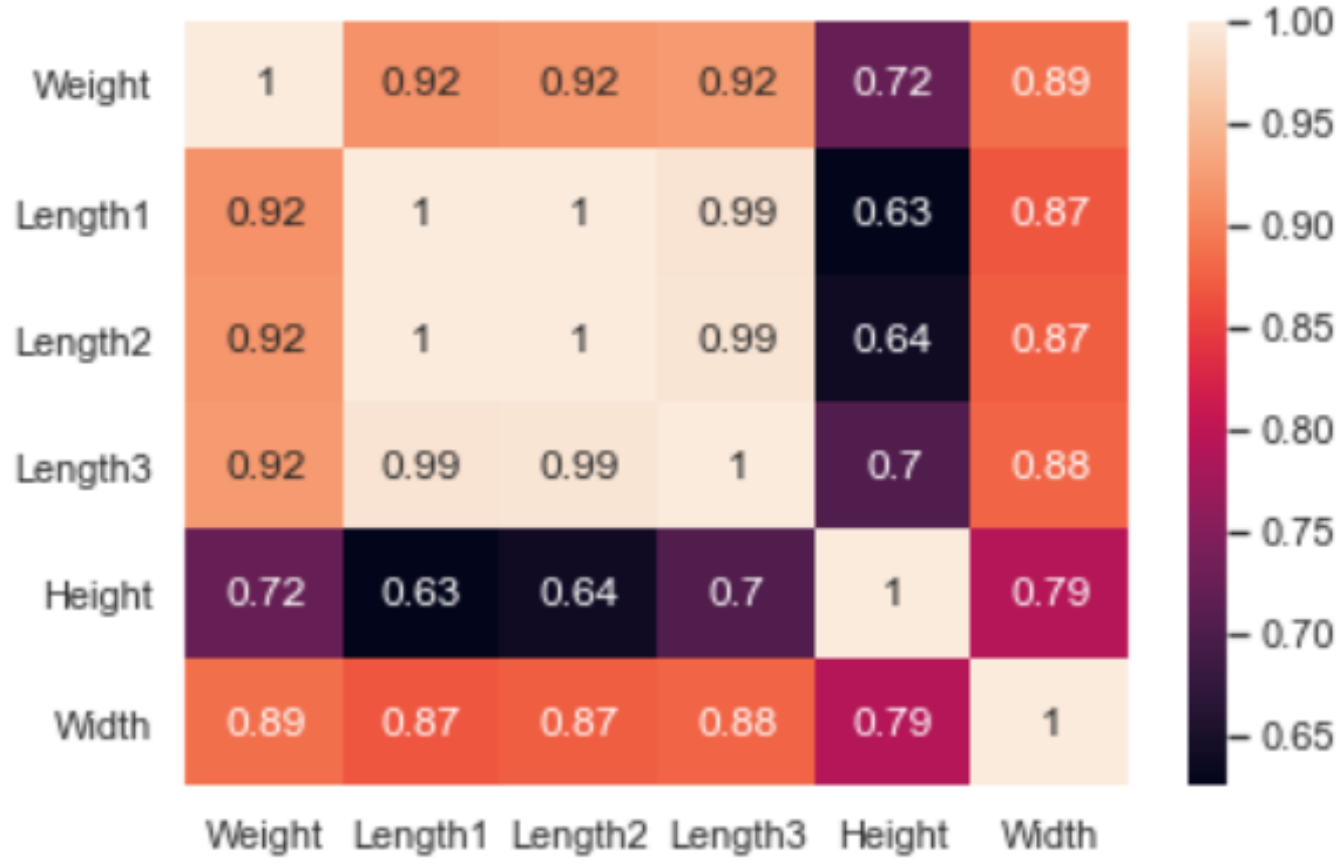
Візуалізація даних

```
corr = fish.corr()
fig, ax = plt.subplots()
im = ax.imshow(corr.values)
ax.set_xticks(np.arange(len(corr.columns)))
ax.set_yticks(np.arange(len(corr.columns)))
ax.set_xticklabels(corr.columns)
ax.set_yticklabels(corr.columns)
plt.setp(ax.get_xticklabels(), rotation=45, ha="right",
        rotation_mode="anchor");
```



Візуалізація даних

sns.heatmap(fish.corr(), annot=True);



Візуалізація даних

Якщо потрібно порівняти кількість, значення, середні значення змінною за різними категоріями, використовується стовпчикова діаграма:

- plt.bar(x, height, width, bottom, align='center', **kwargs)

де x – дані по осі x, height, width – висота і ширина стовпчиків, bottom – y-координати основ стовпчиків, align –розміщення стовпчиків відносно x-координат, **kwargs – інші параметри, наприклад color, linewidth і т.д.

- plt.barh() - горизонтальна діаграма з аналогічними параметрами
- sns.countplot(x, y, hue, data, order, hue_order, orient, color, palette, saturation) – показує кількість об’єктів в кожній катергорії

де x, y, hue – дані для побудови по осі x, y та для позначення відтінком; data –джерело даних; order, hue_order – порядок для категорій; orient – вертикальна (‘’v’’) або горизонтальна (“h”)

- sns.barplot() – за замовчанням показує середнє значення чисельного параметра y для кожної категорії з x. Здебільшого має аналогічні параметри.

параметр estimator вказує статистичну функцію, яка застосовується до y.

Візуалізація даних

Якщо потрібно визначити розподіл однієї кількісної змінної, використовується гістограма:

- plt.hist(x, bins, range, density, weights, cumulative, bottom, histtype, align, orientation, color, label)

де x –дані; bins – кількість стовпчиків (або масив, що визначає краї стовпчиків); range – діапазон даних; density – чи зображати криву густини розподілу; histtype – тип гістограми і т.д.

- sns.histplot(data, x, y, hue, weights, stat, bins, binwidth, binrange, kde, color)

де stat – статистика, що підраховується в кожному стовпчику, за замовчанням 'count'

- sns.displot() – малює діаграму для визначення розподілу, за замовчанням – гістограму.

або діаграма розмаху:

- plt.boxplot()
- sns.boxplot()
- sns.catplot(kind="box")

Візуалізація даних

Якщо потрібно визначити розподіл двовимірної величини, використовується діаграма розсіювання:

- plt.scatter(x,y)
- sns. scatterplot()