Initial exploration

EXPLORATORY DATA ANALYSIS IN PYTHON

Izzy Weber
Curriculum Manager, DataCamp

datacamp

Exploratory Data Analysis

The process of reviewing and cleaning data to...

- derive insights
- generate hypotheses

A first look with .head()

```
books = pd.read_csv("books.csv")
books.head()
```

name	author	rating	year	genre
10-Day Green Smoothie Cleanse	JJ Smith	4.73	2016	Non Fiction
11/22/63: A Novel	Stephen King	4.62	2011	Fiction
12 Rules for Life	Jordan B. Peterson	4.69	2018	Non Fiction
1984 (Signet Classics)	George Orwell	4.73	2017	Fiction
5,000 Awesome Facts	National Geographic Kids	4.81	2019	Childrens

Gathering more .info()

```
books.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 350 entries, 0 to 349
Data columns (total 5 columns):
 #   Column  Non-Null Count  Dtype
--   ------  --------------  -----
 0   name    350 non-null    object
 1   author  350 non-null    object
 2   rating  350 non-null    float64
 3   year    350 non-null    int64
 4   genre   350 non-null    object
dtypes: float64(1), int64(1), object(3)
memory usage: 13.8+ KB
```

A closer look at categorical columns

```
books.value_counts("genre")
```

```
genre
Non Fiction    179
Fiction        131
Childrens       40
dtype: int64
```

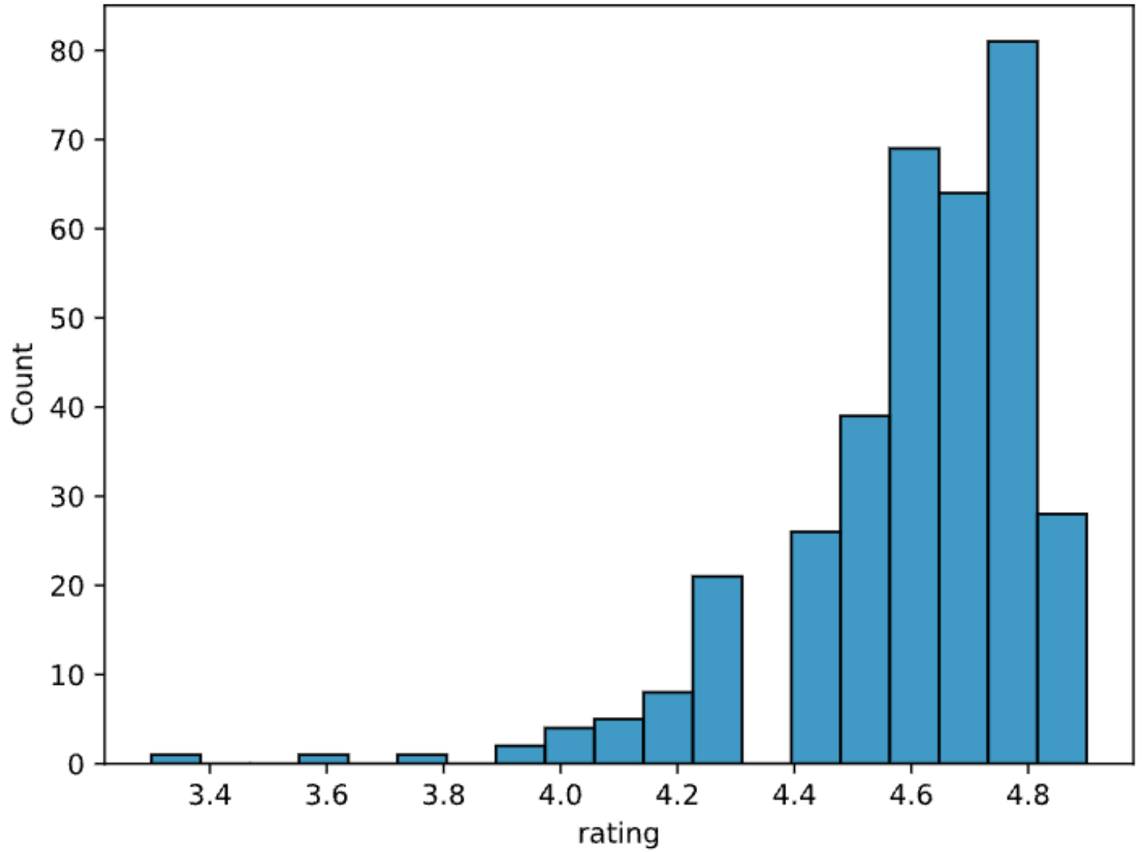
.describe() numerical columns

```
books.describe()
```

```
          rating         year
count    350.000000   350.000000
mean     4.608571    2013.508571
std      0.226941    3.284711
min      3.300000    2009.000000
25%      4.500000    2010.000000
50%      4.600000    2013.000000
75%      4.800000    2016.000000
max      4.900000    2019.000000
```

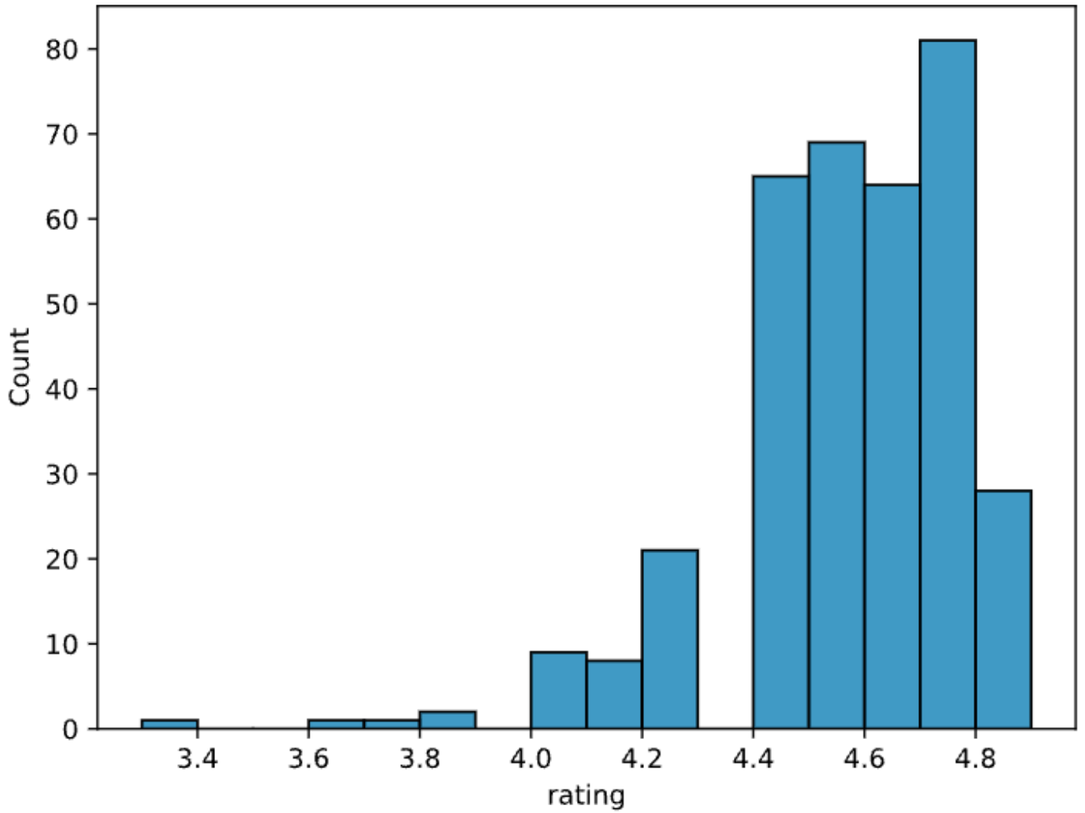
Visualizing numerical data

```
import seaborn as sns
import matplotlib.pyplot as plt
sns.histplot(data=books, x="rating")
plt.show()
```



Adjusting bin width

```
sns.histplot(data=books, x="rating", binwidth=.1)
plt.show()
```



Let's practice!

EXPLORATORY DATA ANALYSIS IN PYTHON

Data validation

EXPLORATORY DATA ANALYSIS IN PYTHON

Izzy Weber
Curriculum Manager, DataCamp

datacamp

Validating data types

```
books.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 350 entries, 0 to 349
Data columns (total 5 columns):
 #   Column  Non-Null Count  Dtype
--   ------  --------------  -----
 0   name    350 non-null    object
 1   author  350 non-null    object
 2   rating  350 non-null    float64
 3   year    350 non-null    float64
 4   genre   350 non-null    object
dtypes: float64(1), int64(1), object(3)
memory usage: 13.8+ KB
```

```
books.dtypes
```

```
name       object
author     object
rating    float64
year      float64
genre      object
dtype: object
```

Updating data types

```
books["year"] = books["year"].astype(int)
books.dtypes
```

```
name        object
author      object
rating     float64
year         int64
genre       object
dtype: object
```

Updating data types

Type	Python Name
String	`str`
Integer	`int`
Float	`float`
Dictionary	`dict`
List	`list`
Boolean	`bool`

Validating categorical data

```
books["genre"].isin(["Fiction", "Non Fiction"])
```

```
0       True
1       True
2       True
3       True
4      False
       ...
345     True
346     True
347     True
348     True
349    False
Name: genre, Length: 350, dtype: bool
```

Validating categorical data

```
~books["genre"].isin(["Fiction", "Non Fiction"])
```

```
0      False
1      False
2      False
3      False
4       True
       ...
345    False
346    False
347    False
348    False
349     True
Name: genre, Length: 350, dtype: bool
```

Validating categorical data

```
books[books["genre"].isin(["Fiction", "Non Fiction"])].head()
```

	name	author	rating	year	genre
0	10-Day Green Smoothie Cleanse	JJ Smith	4.7	2016	Non Fiction
1	11/22/63: A Novel	Stephen King	4.6	2011	Fiction
2	12 Rules for Life	Jordan B. Peterson	4.7	2018	Non Fiction
3	1984 (Signet Classics)	George Orwell	4.7	2017	Fiction
5	A Dance with Dragons	George R. R. Martin	4.4	2011	Fiction

Validating numerical data

```
books.select_dtypes("number").head()
```

```
|    | rating | year |
|---|--------|------|
| 0 |    4.7 | 2016 |
| 1 |    4.6 | 2011 |
| 2 |    4.7 | 2018 |
| 3 |    4.7 | 2017 |
| 4 |    4.8 | 2019 |
```

Validating numerical data

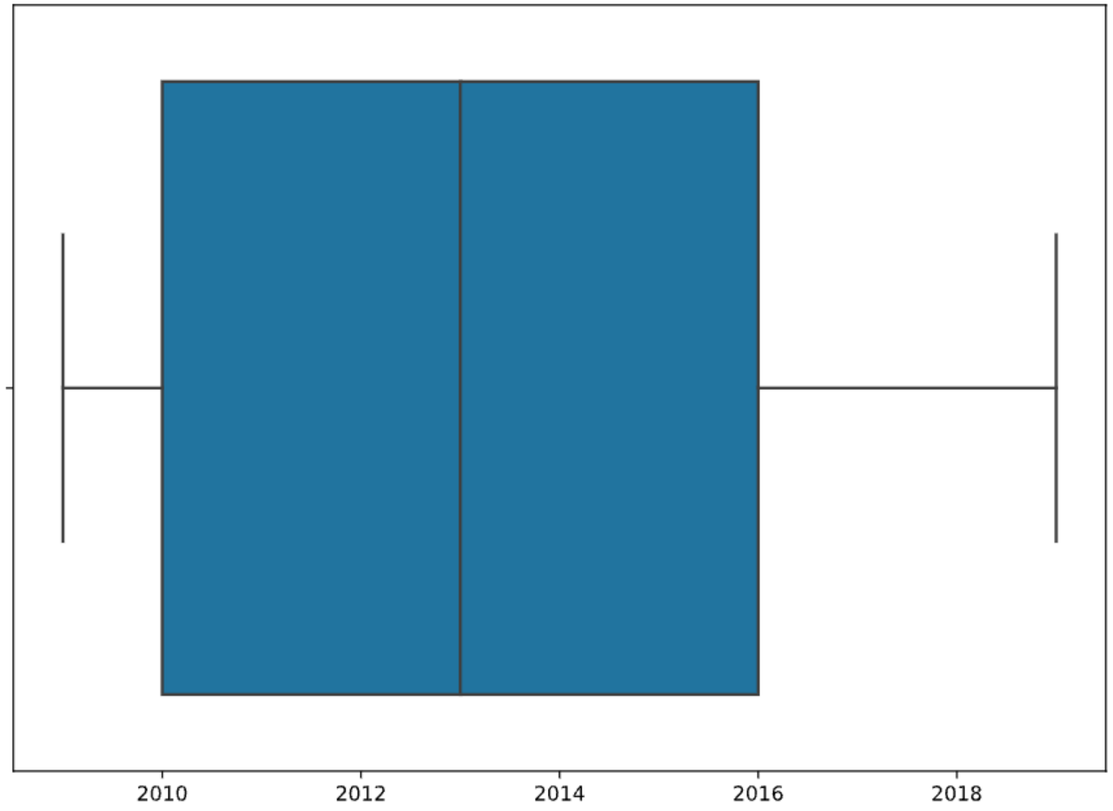
```
books["year"].min()
```

```
2009
```

```
books["year"].max()
```

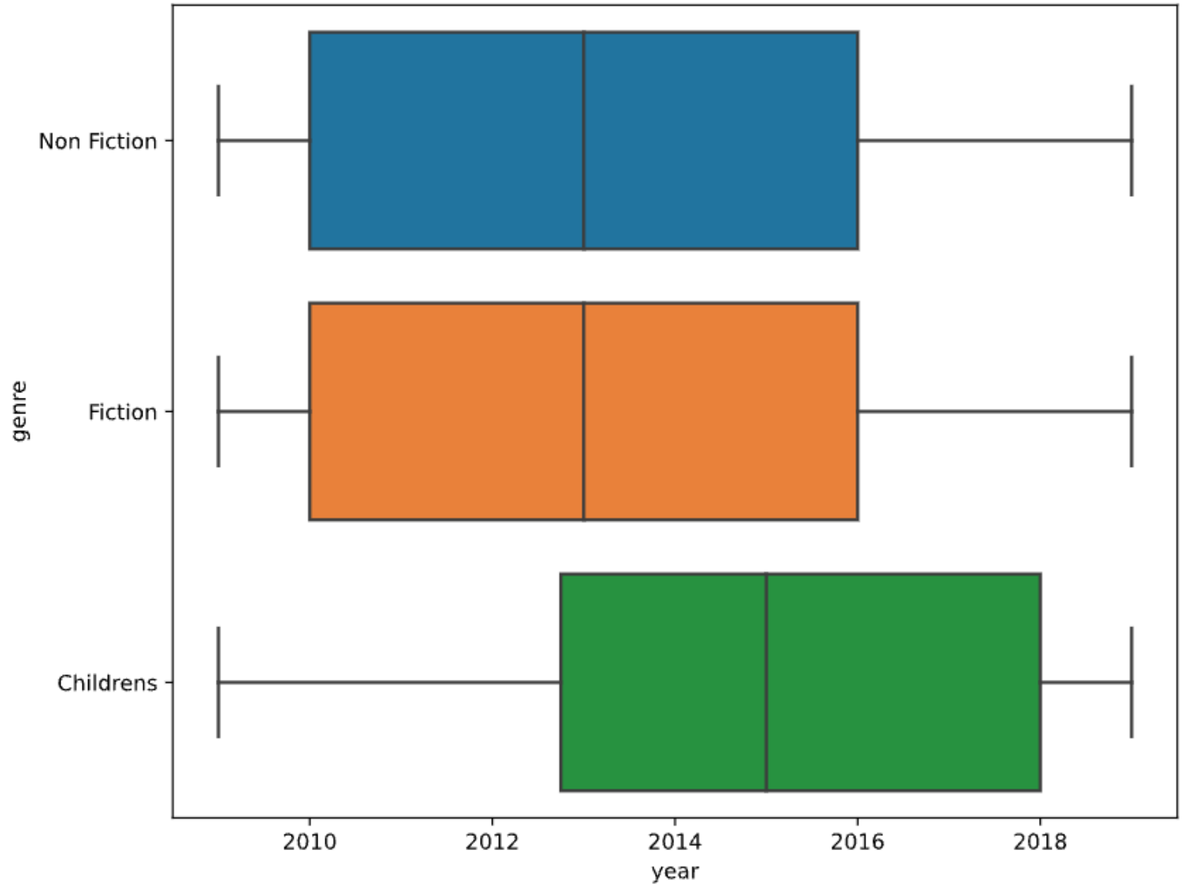
```
2019
```

```
sns.boxplot(data=books, x="year")
plt.show()
```



Validating numerical data

```
sns.boxplot(data=books, x="year", y="genre")
```



Let's practice!

EXPLORATORY DATA ANALYSIS IN PYTHON

Data summarization

EXPLORATORY DATA ANALYSIS IN PYTHON

Izzy Weber
Curriculum Manager, DataCamp

datacamp

Exploring groups of data

- `.groupby()` groups data by category
- Aggregating function indicates how to summarize grouped data

```
books[["genre", "rating", "year"]].groupby("genre").mean()
```

```
|       genre |   rating |        year |
|-------------|----------|-------------|
|   Childrens | 4.780000 | 2015.075000 |
|     Fiction | 4.570229 | 2013.022901 |
| Non Fiction | 4.598324 | 2013.513966 |
```

Aggregating functions

- Sum: `.sum()`
- Count: `.count()`
- Minimum: `.min()`
- Maximum: `.max()`
- Variance: `.var()`
- Standard deviation: `.std()`

Aggregating ungrouped data

- `.agg()` applies aggregating functions across a DataFrame

```
books[["rating", "year"]].agg(["mean", "std"])
```

```
|      |   rating |         year |
|------|----------|--------------|
| mean | 4.608571 | 2013.508571 |
|  std | 0.226941 |     3.28471 |
```

Specifying aggregations for columns

```
books.agg({"rating": ["mean", "std"], "year": ["median"]})
```

```
|        |   rating |   year |
|--------|----------|--------|
|   mean | 4.608571 |    NaN |
|    std | 0.226941 |    NaN |
| median |      NaN | 2013.0 |
```

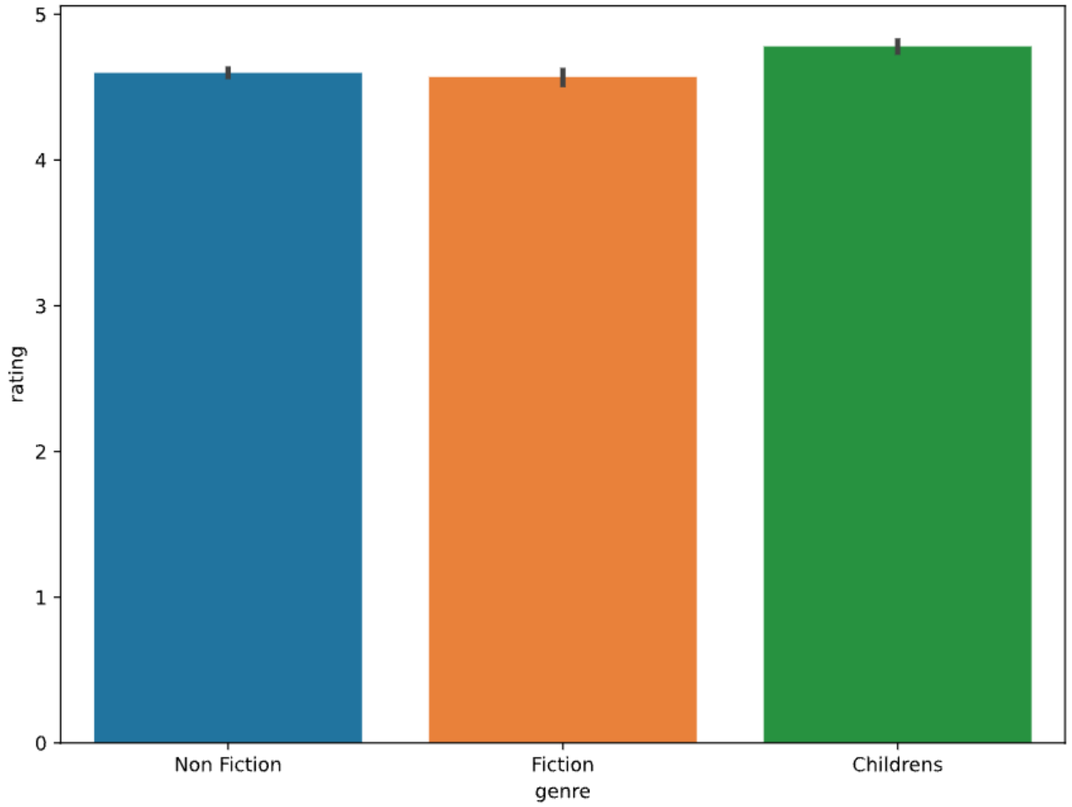
Named summary columns

```
books.groupby("genre").agg(
    mean_rating=("rating", "mean"),
    std_rating=("rating", "std"),
    median_year=("year", "median")
)
```

genre	mean_rating	std_rating	median_year
Childrens	4.780000	0.122370	2015.0
Fiction	4.570229	0.281123	2013.0
Non Fiction	4.598324	0.179411	2013.0

Visualizing categorical summaries

```
sns.barplot(data=books, x="genre", y="rating")
plt.show()
```



Let's practice!

EXPLORATORY DATA ANALYSIS IN PYTHON