ASTA

The ASTA team

Contents

1 Software

1.1 Integrated development environments (IDEs)

- We will use Google Colab for running Python code and Jupyter Notebook files online (i.e. no local installation).
- If you prefer you can install a local program for running this, such as VS code.
- In any case, it is a good idea to make a folder on your computer where you want to keep files to use.

1.2 Basics

```
import numpy as np
```

- Ordinary calculations:

```
4.6 * (2 + 3)**4
```

```
## 2875.0
```

- Make a (scalar) object and print it:

```
a = 4
a
```

```
## 4
```

- Make a (vector) object and print it:

```
b = np.array([2, 5, 7])
b
```

```
## array([2, 5, 7])
```

- Make a sequence of numbers and print it:

```
s = np.arange(1, 5)
s
```

```
## array([1, 2, 3, 4])
```

- Note: A more flexible command for sequences:

```
s = np.arange(1, 5, 2)  # same idea
s
```

```
## array([1, 3])
```

- Elementwise calculations:

```
a * b
```

```
## array([ 8, 20, 28])
```

```
a + b
```

```
## array([ 6,  9, 11])
```

```
b ** 2
```

```
## array([ 4, 25, 49])
```

- Sum and product of elements:

```
np.sum(b)
```

```
## np.int64(14)
```

```
np.prod(b)
```

```
## np.int64(70)
```

2 Data

2.1 Data example

Data: Magazine Ads Readability

- Thirty magazines were ranked by educational level of their readers.
- Three magazines were **randomly** selected from each of the following groups:
  - Group 1: highest educational level

  - Group 2: medium educational level
  - Group 3: lowest educational level.
- Six advertisements were **randomly** selected from each of the following nine selected magazines:
  - Group 1: [1] Scientific American, [2] Fortune, [3] The New Yorker
  - Group 2: [4] Sports Illustrated, [5] Newsweek, [6] People
  - Group 3: [7] National Enquirer, [8] Grit, [9] True Confessions
- So, the data contains information about a total of 54 advertisements.

2.2 Data example (continued) - variables and format

- For each advertisement (54 cases), the data below were observed.
- **Variable names**:
  - WDS = number of words in advertisement
  - SEN = number of sentences in advertisement
  - 3SYL = number of 3+ syllable words in advertisement
  - MAG = magazine (1 through 9 as above)
  - GROUP = educational level (1 through 3 as above)
- Take a look at the data:

```
import pandas as pd

magAds = pd.read_csv("https://asta.math.aau.dk/datasets?file=magazineAds.txt", sep='\t')
magAds.head()
```

```
##    WDS  SEN  X3SYL  MAG  GROUP
## 0  205    9     34    1      1
## 1  203   20     21    1      1
## 2  229   18     37    1      1
## 3  208   16     31    1      1
## 4  146    9     10    1      1
```

- Variable names are in the top row. They are not allowed to start with a digit, so an `X` has been prefixed in `X3SYL`.

2.3 Data types

2.3.1 Quantitative variables

- The measurements have numerical values.
- Quantative data often comes about in one of the following ways:
  - **Continuous variables**: measurements of e.g. waiting times in a queue, revenue, share prices, etc.
  - **Discrete variables**: counts of e.g. words in a text, hits on a webpage, number of arrivals to a queue in one hour, etc.
- Measurements like this have a well-defined scale and in **Python** they are stored as the type **float**.
- It is important to be able to distinguish between discrete count variables and continuous variables, since this often determines how we describe the uncertainty of a measurement.

2.3.2 Categorical/qualitative variables

- The measurement is one of a set of given categories, e.g. sex (male/female), social status, satisfaction score (low/medium/high), etc.
- The measurement is usually stored (which is also recommended) as a **categorical** type in **Python**. The possible categories are called **categories**. Sometimes these are also refered to as factors with levels. Example: the categories of the categorical type "sex" is male/female.
- Categorical types have two so-called scales:

  - **Nominal scale**: There is no natural ordering of the categories, e.g. sex and hair color.
  - **Ordinal scale**: There is a natural ordering of the categories, e.g. social status and satisfaction score.

3 Population and sample

3.1 Aim of statistics

- Statistics is all about "saying something" about a population.
- Typically, this is done by taking a random sample from the population.
- The sample is then analysed and a statement about the population can be made.
- The process of making conclusions about a population from analysing a sample is called **statistical inference**.

3.2 Selecting randomly

- For the magazine data:
  - First we select **randomly** 3 magazines from each group.
  - Then we select **randomly** 6 ads from each magazine.
  - An important detail is that the selection is done completely at **random**, i.e.
    - each magazine within a group have an equal chance of being chosen and
    - each ad within a magazine have an equal chance of being chosen.
- In the following it is a fundamental requirement that the data collection respects this principle of randomness and in this case we use the term **sample**.
- More generally:
  - We have a **population** of objects.
  - We choose completely at random n of these objects, and from the jth object we get the measurement y_j, $j = 1, 2, \ldots, n$.
  - The measurements $y_1, y_2, \ldots, y_n$ are then called a **sample**.
- If we e.g. are measuring the water quality 4 times in a year then it is a bad idea to only collect data in fair weather. The chosen sampling time is not allowed to be influenced by something that might influence the measurement itself.

4 Variable grouping and frequency tables

4.1 Binning

- The function `cut` will divide the range of a numeric variable in a number of equally sized intervals, and record which interval each observation belongs to. E.g. for the variable `X3SYL` (the number of words with more than three syllables) in the magazine data:

```
# Before 'cutting':
magAds["X3SYL"].iloc[0:5]
```

```
## 0    34
## 1    21
## 2    37
## 3    31
## 4    10
## Name: X3SYL, dtype: int64
```

```
# After 'cutting' into 4 intervals:
syll = pd.cut(magAds["X3SYL"], bins=4)
```

```
# First 5 values
syll.iloc[0:5]
```

```
## 0      (32.25, 43.0]
## 1      (10.75, 21.5]
## 2      (32.25, 43.0]
## 3      (21.5, 32.25]
## 4    (-0.043, 10.75]
## Name: X3SYL, dtype: category
## Categories (4, interval[float64, right]): [(-0.043, 10.75] < (10.75, 21.5] < (21.5, 32.25] <
##                                            (32.25, 43.0]]
```

- The result is a `category` and the labels are the interval end points by default. Custom ones can be assigned through the `labels` argument:

```
labs = ["few", "some", "many", "lots"]
syll = pd.cut(magAds["X3SYL"], bins = 4, labels = labs)
syll.iloc[0:5]
```

```
## 0    lots
## 1    some
## 2    lots
## 3    many
## 4     few
## Name: X3SYL, dtype: category
## Categories (4, object): ['few' < 'some' < 'many' < 'lots']
```

```
magAds["syll"] = syll
magAds.head()
```

```
##    WDS  SEN  X3SYL  MAG  GROUP  syll
## 0  205    9     34    1      1  lots
## 1  203   20     21    1      1  some
## 2  229   18     37    1      1  lots
## 3  208   16     31    1      1  many
## 4  146    9     10    1      1   few
```

4.2 Tables

- To summarize the results we can use the function `value_counts()` from `pandas` package:

```
magAds["syll"].value_counts()
```

```
## syll
## few     26
## some    14
## many    10
## lots     4
## Name: count, dtype: int64
```

- In percent:

```
magAds["syll"].value_counts(normalize = True) * 100
```

```
## syll
## few     48.148148
## some    25.925926
## many    18.518519
```

```
## lots     7.407407
## Name: proportion, dtype: float64
```

4.3 2 factors: Cross tabulation

- To make a table of all combinations of two factors:

```
pd.crosstab(magAds["syll"], magAds["GROUP"])
```

```
## GROUP  1   2  3
## syll
## few    8  11  7
## some   4   2  8
## many   3   5  2
## lots   3   0  1
```

- Relative frequencies (in percent) columnwise:

```
magAds.groupby("GROUP")["syll"].value_counts(normalize=True).mul(100)
```

```
## GROUP  syll
## 1      few     44.444444
##        some    22.222222
##        many    16.666667
##        lots    16.666667
## 2      few     61.111111
##        many    27.777778
##        some    11.111111
##        lots     0.000000
## 3      some    44.444444
##        few     38.888889
##        many    11.111111
##        lots     5.555556
## Name: proportion, dtype: float64
```

- So, the above table shows e.g. how many percentage of the advertisements in group 1 that have 'few', 'some', 'many' or 'lots' words with more than 3 syllables.

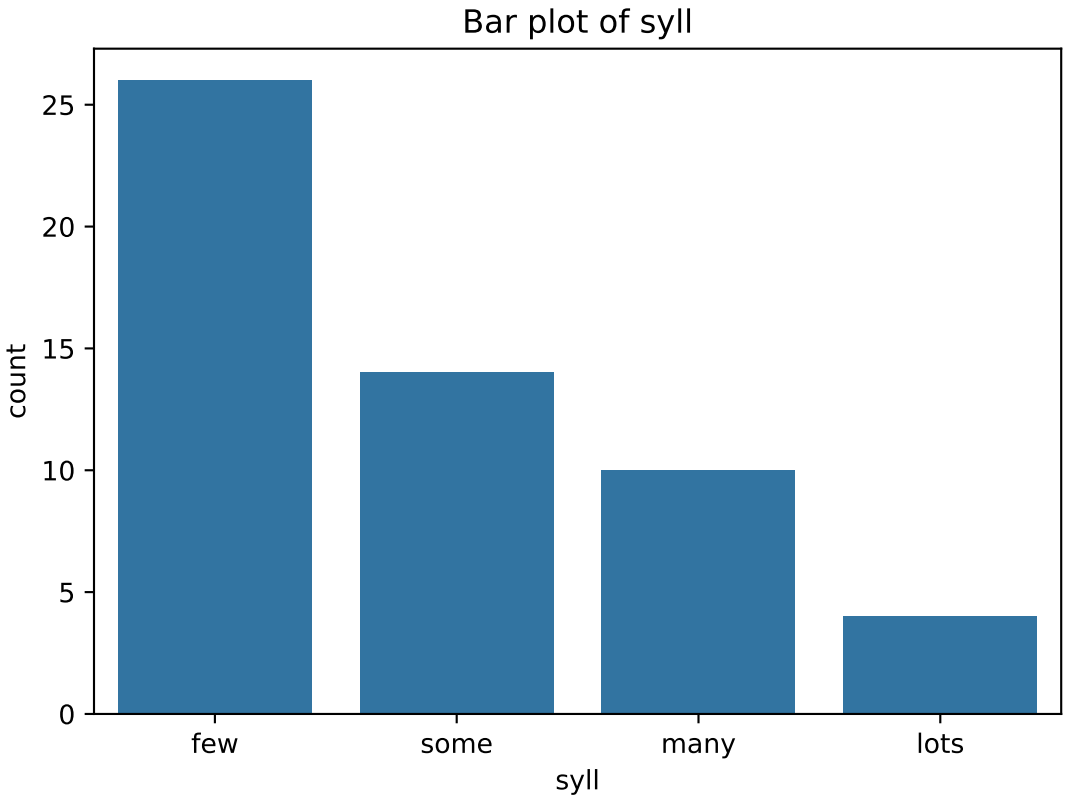
5 Graphics

5.1 Bar graph

- To create a bar graph plot of table data we use the function `countplot` from `seaborn`. For each level of the factor a box is drawn with the height proportional to the frequency (count) of the level.

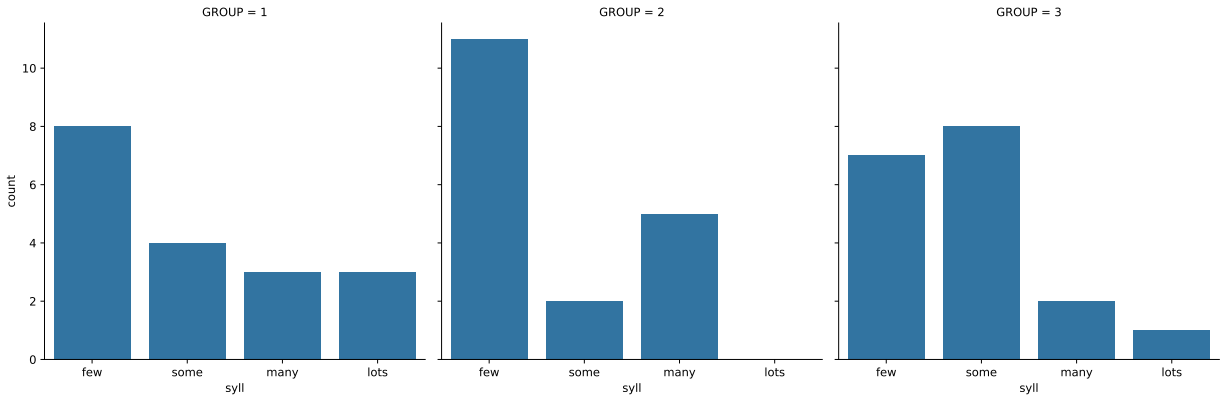
```
import seaborn as sns
import matplotlib.pyplot as plt

g = sns.countplot(data = magAds, x = "syll")
g.set_title("Bar plot of syll")
```



- The bar graph can also be split by group:

```
g = sns.catplot(data = magAds, x = "syll", kind = "count", col = "GROUP")
```



5.2 The Ericksen data

- Description of data: Ericksen 1980 U.S. Census Undercount.
- This data contains the following variables:
  - `minority`: Percentage black or Hispanic.
  - `crime`: Rate of serious crimes per 1000 individuals in the population.
  - `poverty`: Percentage poor.
  - `language`: Percentage having difficulty speaking or writing English.
  - `highschool`: Percentage aged 25 or older who had not finished highschool.

  - `housing`: Percentage of housing in small, multiunit buildings.
  - `city`: A factor with levels: `city` (major city) and `state` (state or state-remainder).
  - `conventional`: Percentage of households counted by conventional personal enumeration.
  - `undercount`: Preliminary estimate of percentage undercount.
- The Ericksen data has 66 rows/observations and 9 columns/variables.
- The observations are measured in 16 large cities, the remaining parts of the states in which these cities are located, and the other U.S. states.

```
Ericksen = pd.read_csv("https://asta.math.aau.dk/datasets?file=Ericksen.txt", sep='\t')
Ericksen.head()
```

```
##            name  minority  crime  poverty  language  highschool  housing   city  conventional  under
## 0       Alabama      26.1     49     18.9       0.2        43.5      7.6  state             0
## 1        Alaska       5.7     62     10.7       1.7        17.5     23.6  state           100
## 2       Arizona      18.9     81     13.2       3.2        27.6      8.1  state            18
## 3      Arkansas      16.9     38     19.0       0.2        44.5      7.0  state             0
## 4  California.R      24.3     73     10.4       5.0        26.0     11.8  state             4
```

```
pd.set_option('display.max_columns', None)  # Show all columns
Ericksen.head()
```

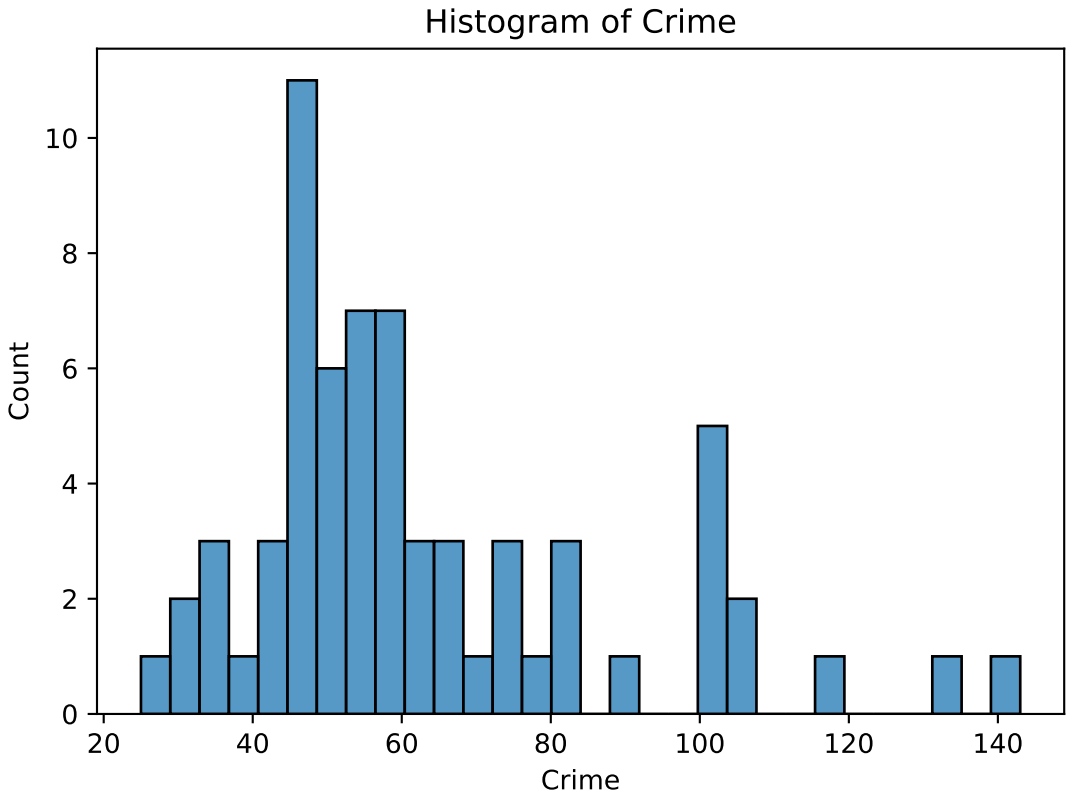
```
##            name  minority  crime  poverty  language  highschool  housing  \
## 0       Alabama      26.1     49     18.9       0.2        43.5      7.6
## 1        Alaska       5.7     62     10.7       1.7        17.5     23.6
## 2       Arizona      18.9     81     13.2       3.2        27.6      8.1
## 3      Arkansas      16.9     38     19.0       0.2        44.5      7.0
## 4  California.R      24.3     73     10.4       5.0        26.0     11.8
##
##     city  conventional  undercount
## 0  state             0       -0.04
## 1  state           100        3.35
## 2  state            18        2.48
## 3  state             0       -0.74
## 4  state             4        3.60
```

- Want to make a histogram for crime rate - how?

5.3 Histogram (quantitative variables)

- How to make a histogram for some variable `x`:
  - Divide the interval from the minimum value of `x` to the maximum value of `x` in an appropriate number of equal sized sub-intervals.
  - Draw a box over each sub-interval with the height being proportional to the number of observations in the sub-interval.
- Histogram of crime rates for the Ericksen data

```
sns.histplot(data = Ericksen, x = "crime", bins = 30, edgecolor = "black")
plt.xlabel("Crime")
plt.ylabel("Count")
plt.title("Histogram of Crime")
plt.show()
```



6 Summary of quantitative variables

6.1 Measures of center of data: Mean and median

- We return to the magazine ads example (`WDS` = number of words in advertisement). A number of numerical summaries for `WDS` can be retrieved using the `favstats` function:

```
col = magAds["WDS"]
summary = {
    "min": col.min(),
    "Q1": col.quantile(0.25),
    "median": col.median(),
    "mean": col.mean(),
    "Q3": col.quantile(0.75),
    "max": col.max(),
    "sd": col.std(),
    "n": col.count(),
    "missing": col.isna().sum()
}
pd.DataFrame([summary])
```

```
##    min    Q1  median       mean     Q3  max         sd   n  missing
## 0   31  69.0    95.5  122.62963  201.5  230  65.877043  54        0
```

```
#or: magAds["WDS"].describe()
```

- The observed values of the variable `WDS` are $y_1 = 205$, $y_2 = 203, \ldots, y_n = 208$, where there are a total of

$n = 54$ values. As previously defined this constitutes a **sample**.

- **mean** = 122.63 is the **average** of the sample, which is calculated by

$$\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$$

We may also call $\bar{y}$ the **(empirical) mean** or the **sample mean**.

- **median** = 95.5 is the 50-percentile, i.e. the value that splits the sample in 2 groups of equal size.
- An important property of the **mean** and the **median** is that they have the same unit as the observations (e.g. meter).

6.2 Measures of variability of data: range, standard deviation and variance

- The **range** is the difference of the largest and smallest observation.
- The **(empirical) variance** is the average of the squared deviations from the mean:

$$s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(y_i - \bar{y})^2.$$

- **sd = standard deviation** $= s = \sqrt{s^2}$.
- Note: If the observations are measured in meter, the **variance** has unit meter^2 which is hard to interpret. The **standard deviation** on the other hand has the same unit as the observations.
- The standard deviation describes how much data varies around the (empirical) mean.

6.3 Calculation of mean, median and standard deviation

The mean, median and standard deviation are just some of the summaries that can be read of the output (shown on previous page). They may also be calculated separately in the following way:

- Mean of `WDS`:

```
magAds["WDS"].mean()
```

```
## np.float64(122.62962962962963)
```

- Median of `WDS`:

```
magAds["WDS"].median()
```

```
## np.float64(95.5)
```

- Standard deviation for `WDS`:

```
magAds["WDS"].std()
```

```
## np.float64(65.87704278349153)
```

We may also calculate the summaries for each group (variable `GROUP`), e.g. for the mean:

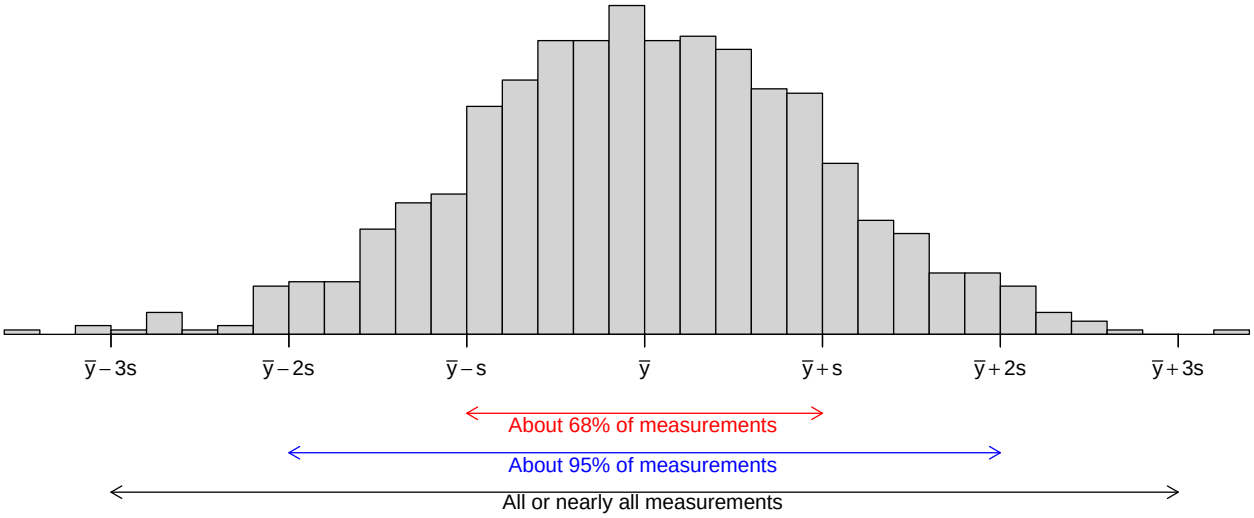
```
magAds.groupby("GROUP")["WDS"].mean().reset_index() # reset_index() resets the grouping again
```

```
##    GROUP         WDS
## 0      1  140.000000
## 1      2  121.388889
## 2      3  106.500000
```

6.4 A word about terminology

- **Standard deviation**: a measure of variability of a population or a sample.
- **Standard error**: a measure of variability of an estimate. For example, a measure of variability of the sample mean.

6.5 The empirical rule



If the histogram of the sample looks like a bell shaped curve, then

- about 68% of the observations lie between $\bar{y} - s$ and $\bar{y} + s$.
- about 95% of the observations lie between $\bar{y} - 2s$ and $\bar{y} + 2s$.
- All or almost all (99.7%) of the observations lie between $\bar{y} - 3s$ and $\bar{y} + 3s$.

6.6 Percentiles

- **The pth percentile** is a value such that about p% of the population (or sample) lies below or at this value and about $(100 - p)$% of the population (or sample) lies above it.

6.6.1 Percentile calculation for a sample:

- First, sort data in increasing order. For the `WDS` variable in the magazine data:

$$y_{(1)} = 31, y_{(2)} = 32, y_{(3)} = 34, \ldots, y_{(n)} = 230.$$

  Here the number of observations is $n = 54$.

- Find the 5th percentile (i. e. $p = 5$):
  - The observation number corresponding to the 5-percentile is $N = \frac{54 \cdot 5}{100} = 2.7$.
  - So the 5-percentile lies between the observations with observation number $k = 2$ and $k + 1 = 3$. That is, its value lies somewhere in the interval between $y_2 = 32$ and $y_3 = 34$
  - One of several methods for estimating the 5-percentile from the value of N is defined as:

$$y_{(k)} + (N - k)(y_{(k+1)} - y_{(k)})$$

    which in this case states

$$y_2 + (2.7 - 2)(y_3 - y_2) = 32 + 0.7 \cdot (34 - 32) = 33.4$$

6.7 Median, quartiles and interquartile range

Recall

```
col = magAds["WDS"]
summary = {
    "min": col.min(),
```

```
    "Q1": col.quantile(0.25),
    "median": col.median(),
    "Q3": col.quantile(0.75),
    "max": col.max(),
    "n": col.count(),
    "missing": col.isna().sum()
}
pd.DataFrame([summary])
```

```
##    min    Q1  median     Q3  max   n  missing
## 0   31  69.0    95.5  201.5  230  54        0
```

```
#or: magAds["WDS"].describe()
```

- 50-percentile = 96 is the **median** and it is a measure of the center of data.
- 0-percentile = 31 is the **minimum** value.
- 25-percentile = 69 is called the **lower quartile** (Q1). Median of lower 50% of data.
- 75-percentile = 202 is called the **upper quartile** (Q3). Median of upper 50% of data.
- 100-percentile = 230 is the **maximum** value.
- **Interquartile Range (IQR)**: a measure of variability given by the difference of the upper and lower quartiles: 202 - 69 = 133.

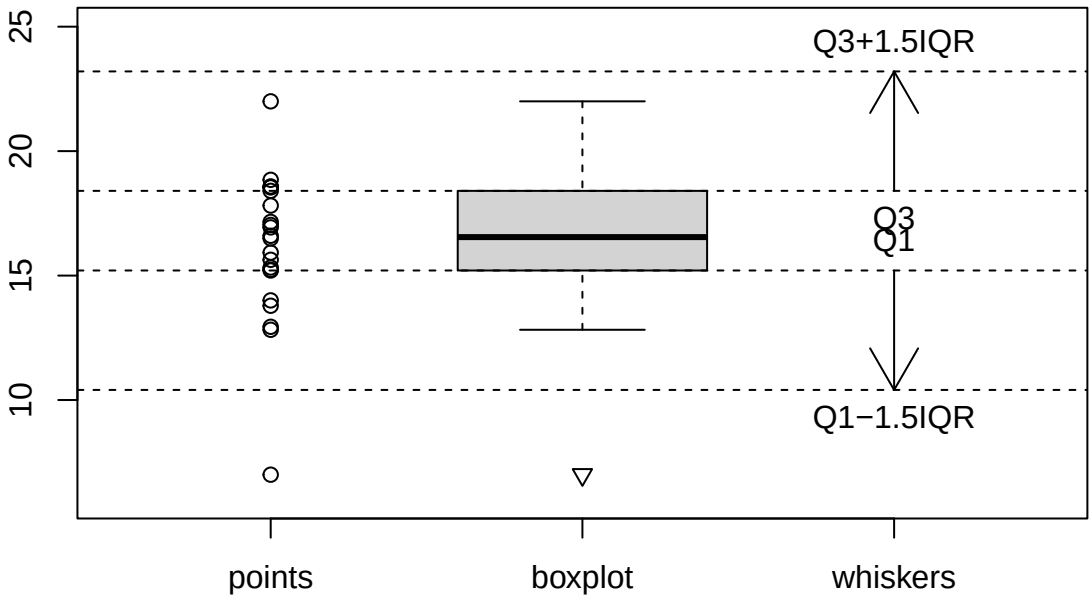
7 More graphics

7.1 Box-and-whiskers plots (or simply box plots)

How to draw a box-and-whiskers plot:

- Box:
  - Calculate the median, lower and upper quartiles.
  - Plot a line by the median and draw a box between the upper and lower quartiles.
- Whiskers:
  - Calculate interquartile range and call it IQR.
  - Calculate the following values:
    - L = lower quartile - 1.5*IQR
    - U = upper quartile + 1.5*IQR
  - Draw a line from lower quartile to the smallest measurement, which is larger than L.
  - Similarly, draw a line from upper quartile to the largest measurement which is smaller than U.
- Outliers: Measurements smaller than L or larger than U are drawn as circles.

Note: Whiskers are minimum and maximum of the observations that are not deemed to be outliers.



7.1.1 Boxplot for Ericksen data

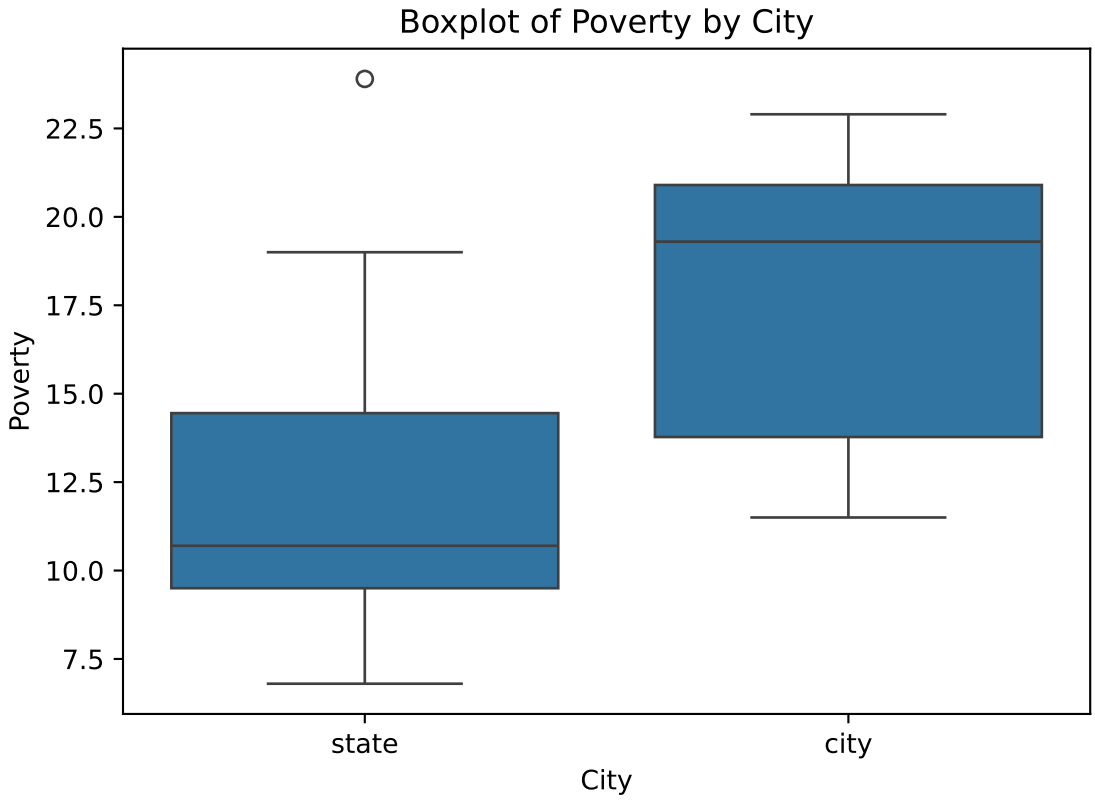
Boxplot of the poverty rates separately for cities and states (variable `city`):

```
summary_by_city = Ericksen.groupby("city")["poverty"].agg(
    ['min',
     lambda x: x.quantile(0.25),  # Q1
     'median',
     'mean',
     lambda x: x.quantile(0.75),  # Q3
     'max',
     'std',
     'count',
     lambda x: x.isna().sum()      # missing values
])

# Rename columns for clarity
summary_by_city.columns = ['min', 'Q1', 'median', 'mean', 'Q3', 'max', 'sd', 'n', 'missing']
summary_by_city
```

```
##         min      Q1  median      mean     Q3   max        sd   n  missing
## city
## city   11.5  13.775    19.3  17.69375  20.90  22.9  4.041859  16        0
## state   6.8   9.500    10.7  12.11600  14.45  23.9  3.733596  50        0
```

```
sns.boxplot(data=Ericksen, x="city", y="poverty")
plt.xlabel("City")
plt.ylabel("Poverty")
plt.title("Boxplot of Poverty by City")
plt.show()
```

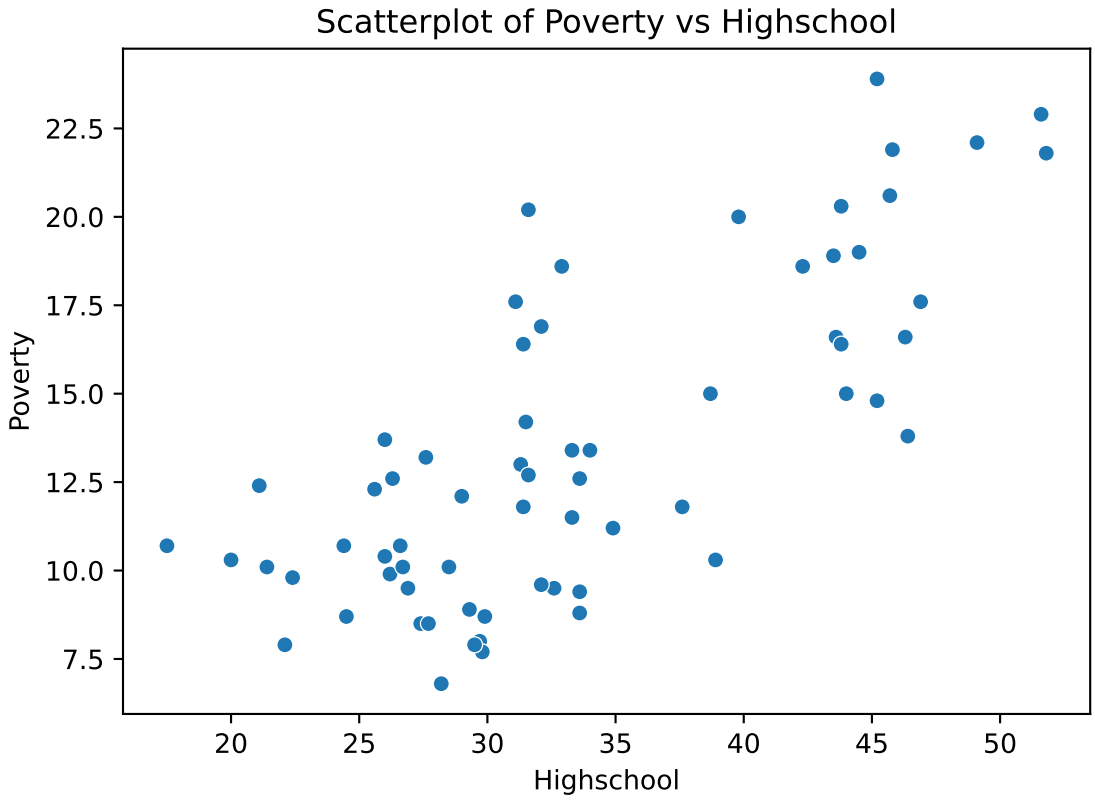


- There seems to be more poverty in the cities.
- A single state differs noticeably from the others with a high poverty rate.

7.2 2 quantitative variables: Scatter plot

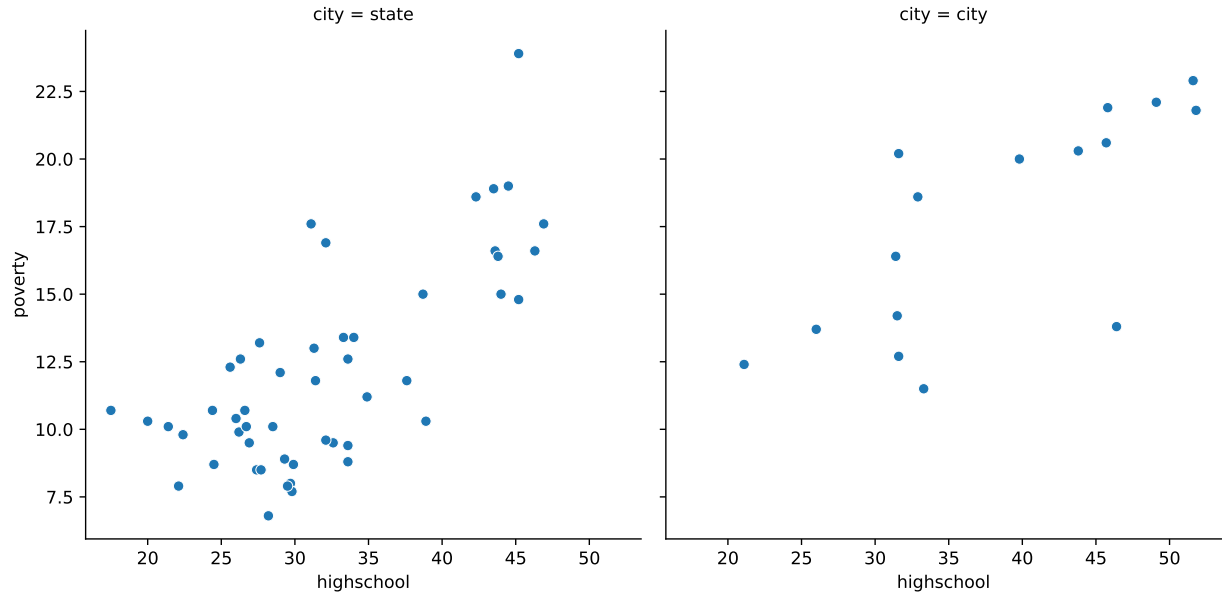
For two quantitative variables the usual graphic is a scatter plot:

```
sns.scatterplot(data=Ericksen, x="highschool", y="poverty")
plt.xlabel("Highschool")
plt.ylabel("Poverty")
plt.title("Scatterplot of Poverty vs Highschool")
plt.show()
```

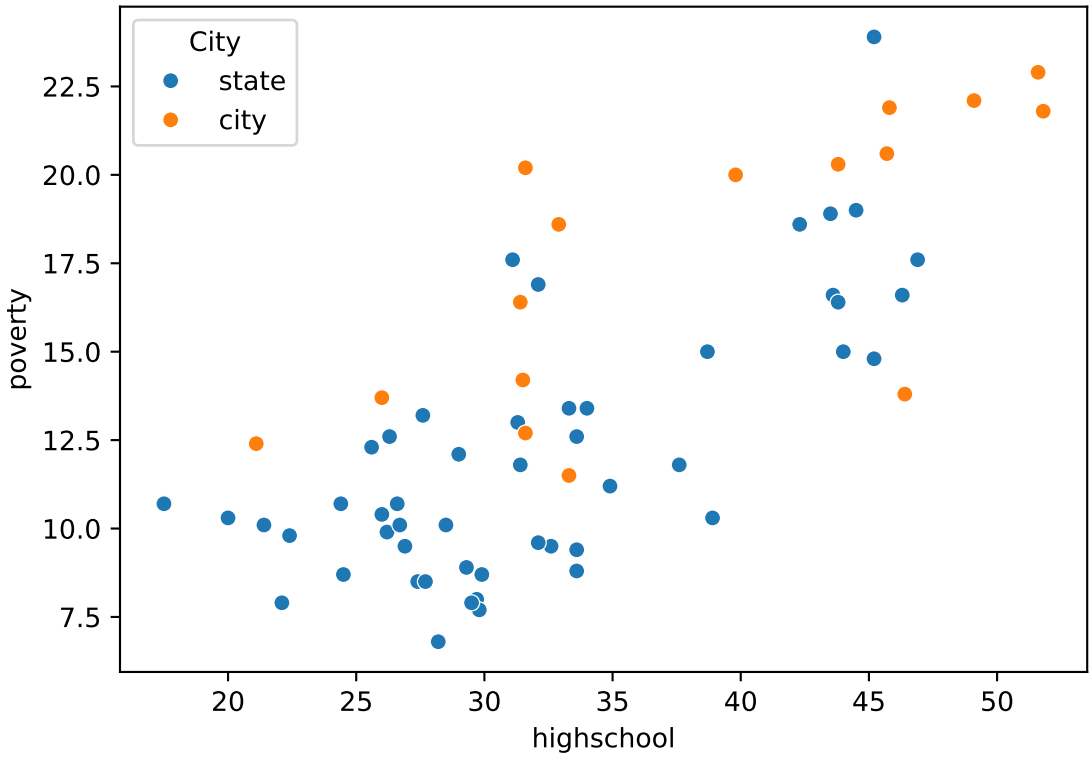


This can be either split or coloured according to the value of `city`:

```
g = sns.relplot(data=Ericksen, x="highschool", y="poverty", col="city", kind="scatter")
```

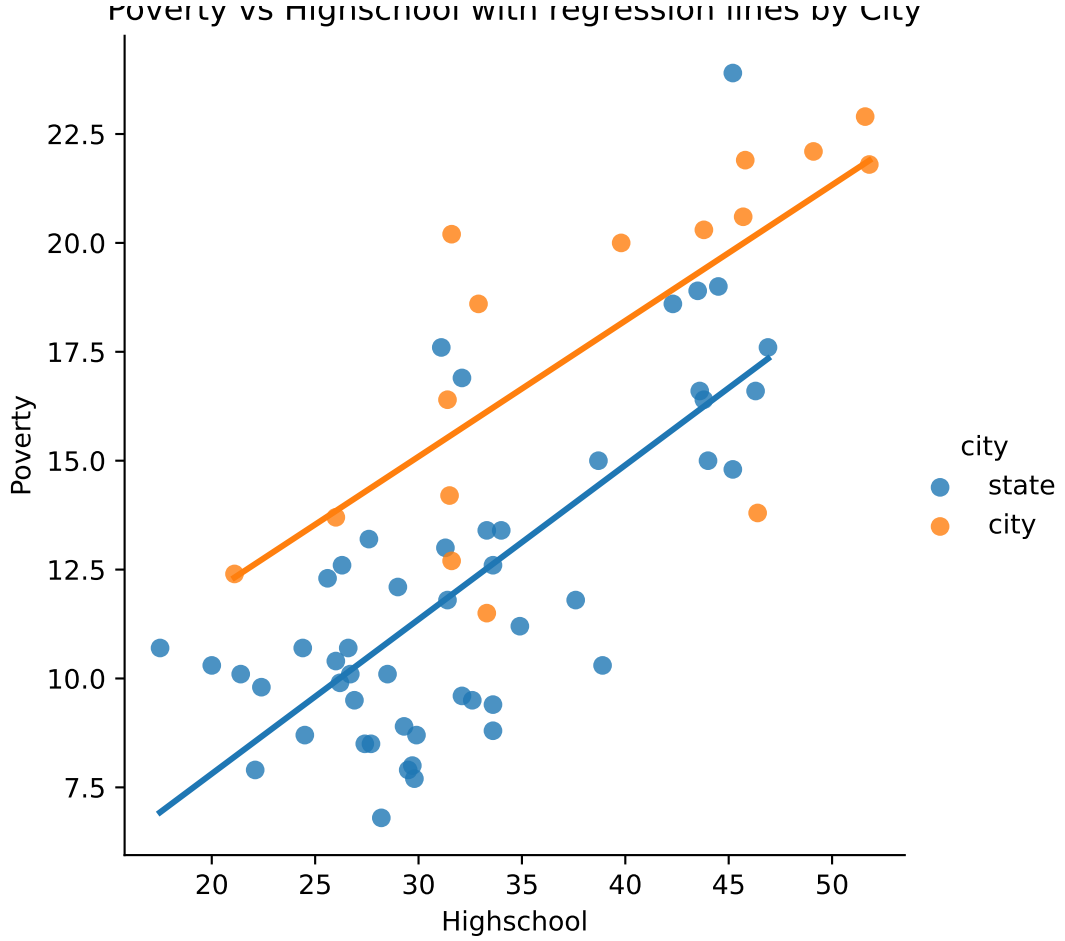



```
g = sns.scatterplot(data=Ericksen, x="highschool", y="poverty", hue="city")
plt.legend(title="City")
```



If we want a regression line along with the points we can do:

```
g = sns.lmplot(data=Ericksen, x="highschool", y="poverty", hue="city", ci=None)
plt.xlabel("Highschool")
plt.ylabel("Poverty")
plt.title("Poverty vs Highschool with regression lines by City")
```



8 Appendix

8.1 Recoding variables

- `astype` can be used to convert a vector of numerical values to be a categorical variable. E.g.:

```
magAds["GROUP"].head()
```

```
## 0    1
## 1    1
## 2    1
## 3    1
## 4    1
## Name: GROUP, dtype: int64
```

```
f = magAds["GROUP"].astype("category")
magAds["GROUP"] = f
magAds["GROUP"].head()
```

```
## 0    1
## 1    1
## 2    1
## 3    1
## 4    1
```

```
## Name: GROUP, dtype: category
## Categories (3, int64): [1, 2, 3]
```

- Custom names for the categories can also be used:

```
f = magAds["GROUP"].astype("category")
f.value_counts()
```

```
## GROUP
## 1    18
## 2    18
## 3    18
## Name: count, dtype: int64
```

```
f = f.cat.rename_categories({
    1: 'high',
    2: 'medium',
    3: 'low'
})
magAds['GROUP'] = f
magAds["GROUP"].head()
```

```
## 0    high
## 1    high
## 2    high
## 3    high
## 4    high
## Name: GROUP, dtype: category
## Categories (3, object): ['high', 'medium', 'low']
```

```
magAds['GROUP'].value_counts()
```

```
## GROUP
## high      18
## medium    18
## low       18
## Name: count, dtype: int64
```

- In this way the numbers are replaced by more informative labels describing the educational level.

9 Probability of events

9.1 The concept of probability

- Experiment: Measure the waiting times in a queue where we note 1, if it exceeds 2 minutes and 0 otherwise.
- The experiment is carried out n times with results $y_1, y_2, \ldots, y_n$. There is **random variation** in the outcome, i.e. sometimes we get a 1 other times a 0.
- **Empirical probability** of exceeding 2 minutes:

$$p_n = \frac{\sum_{i=1}^{n} y_i}{n}.$$

- **Theoretical probability** of exceeding 2 minutes:

$$\pi = \lim_{n \to \infty} p_n.$$

- We try to make inference about π based on a sample, e.g. "Is $\pi > 0.1$?" ("do more than 10% of the customers experience a waiting time in excess of 2 minutes?").

- Statistical inference is concerned with such questions when we only have a finite sample.

9.2 Actual experiment

- On February 23, 2017, a group of students were asked how long time (in minutes) they waited in line last time they went to the canteen at AAU's Copenhagen campus:

```
import numpy as np
y_canteen = np.array([2, 5, 1, 6, 1, 1, 1, 1, 3, 4, 1, 2, 1, 2, 2, 2, 4, 2, 2, 5, 20, 2, 1, 1, 1, 1])
x_canteen = np.where(y_canteen > 2, 1, 0)
x_canteen
```

```
## array([0, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0,
##        0, 0, 0, 0])
```

- Empirical probability of waiting more than 2 minutes:

```
p_canteen = x_canteen.sum() / len(x_canteen)
p_canteen
```

```
## np.float64(0.2692307692307692)
```

- Question: Is the population probability $\pi > 1/3$?
- Notice: One student said he had waited for 20 minutes (we doubt that; he was trying to make himself interesting. Could consider ignoring that observation).

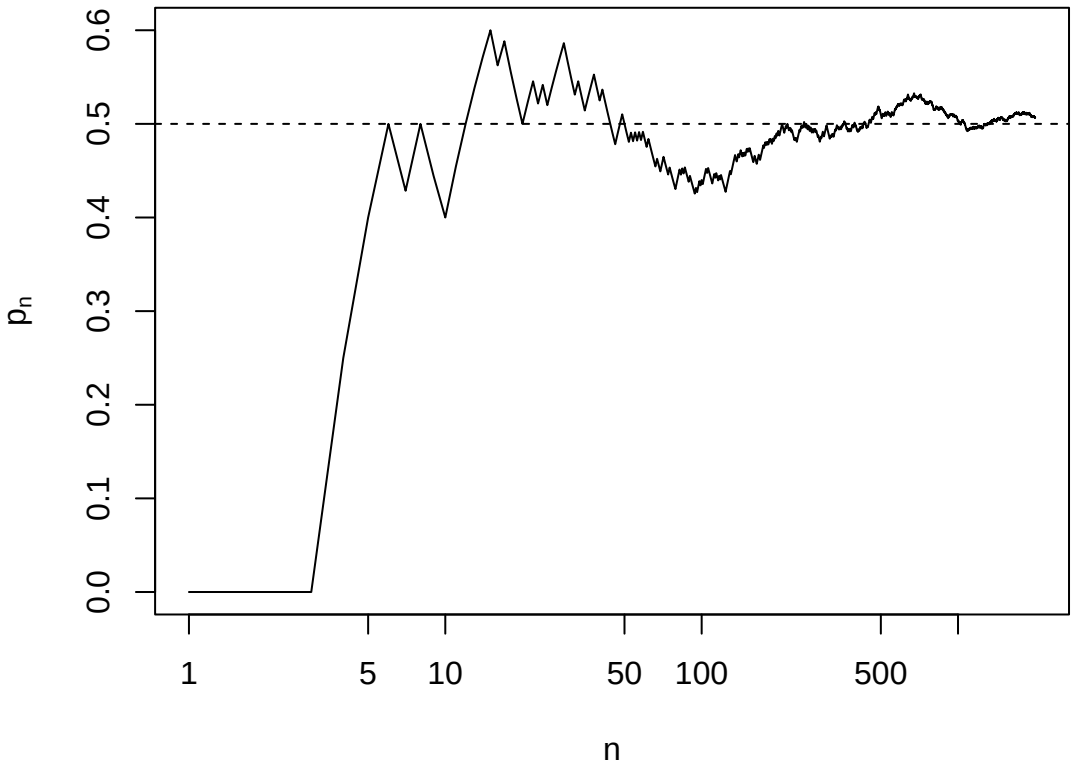
9.3 Another experiment

- John Kerrich, a South African mathematician, was visiting Copenhagen when World War II broke out. Two days before he was scheduled to fly to England, the Germans invaded Denmark. Kerrich spent the rest of the war interned at a camp in Hald Ege near Viborg, Jutland, and to pass the time he carried out a series of experiments in probability theory. In one, he tossed a coin 10,000 times. His results are shown in the following graph.
- Below, `x` is a vector with the first 2,000 outcomes of John Kerrich's experiment (0 = tail, 1 = head):

```
np.array([0, 0, 0, 1, 1, 1, 0, 1, 0, 0])
```

```
## array([0, 0, 0, 1, 1, 1, 0, 1, 0, 0])
```

- Plot of the empirical probability p_n of getting a head against the number of tosses n:



(The horizontal axis is on a log scale).

9.4 Definitions

- **Sample space**: All possible outcomes of the experiment.
- **Event**: A subset of the sample space.

We conduct the experiment n times. Let $\#(A)$ denote how many times we observe the event A.

- **Empirical probability** of the event A:

$$p_n(A) = \frac{\#(A)}{n}.$$

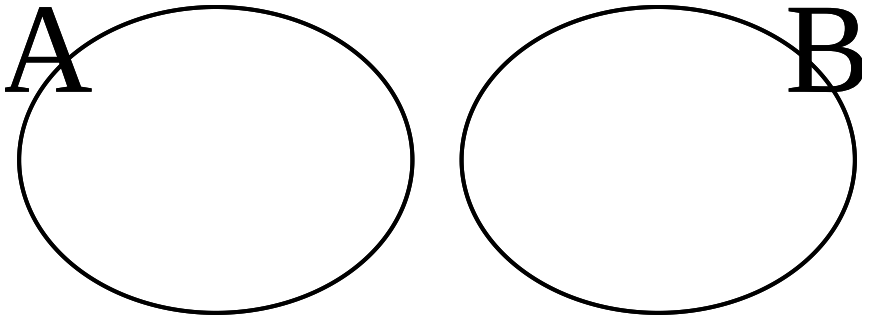
- **Theoretical probability** of the event A:

$$P(A) = \lim_{n\to\infty} p_n(A)$$

- We always have $0 \leq P(A) \leq 1$.
- **Examples:**
  - **Example 1** Tossing a coin once. The sample space is S = {H, T}. E = {H} is an event.
  - **Example 2** Tossing a die. The sample space is S = {1, 2, 3, 4, 5, 6}. E = {2, 4, 6} is an event, which can be described in words as **the number is even**.
  - **Example 3** Tossing a coin twice. The sample space is S = {HH, HT, TH, TT}. E = {HH, HT} is an event, which can be described in words as **the first toss results in a Heads**.
  - **Example 4** Tossing a die twice. The sample space is S = {(i, j) : i, j = 1, 2, ..., 6}, which contains 36 outcomes. **The sum of the results of the two tosses is equal to 10** is an event.

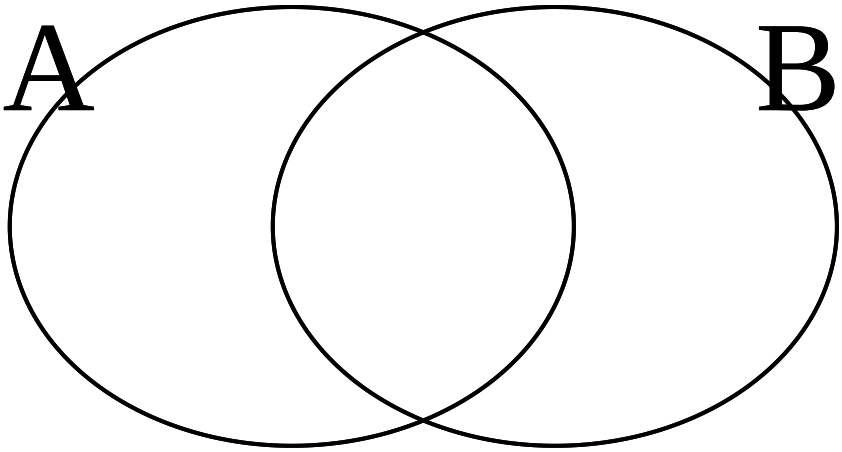
9.5 Theoretical probabilites of two events

- If the two events A and B are **disjoint** (non-overlapping) then
  - $\#(A \text{ and } B) = 0$ implying that $P(A \text{ and } B) = 0$.
  - $\#(A \text{ or } B) = \#(A) + \#(B)$ implying that $P(A \text{ or } B) = P(A) + P(B)$.



- If the two events A and B **are not disjoint** then the more general formula is

$$P(A \text{ or } B) = P(A) + P(B) - P(A \text{ and } B).$$



9.6 Conditional probability

- Say we consider two events A and B. Then the **conditional probability** of A given (or conditional on) the event B is written $P(A \mid B)$ and is defined by

$$P(A \mid B) = \frac{P(A \text{ and } B)}{P(B)}.$$

- The above probability can be understood as: "how probable A is if we know that B has happened".

9.6.1 Example with magazine data:

```
import pandas as pd

magAds = pd.read_csv("https://asta.math.aau.dk/datasets?file=magazineAds.txt", sep = "\t")
magAds['words'] = pd.cut(
    magAds['WDS'],
    bins = [31, 72, 146, 230],
```

```
    include_lowest = True
)
magAds['education'] = pd.Categorical(
    magAds['GROUP'].map({1: 'high', 2: 'medium', 3: 'low'}),
    categories = ['high', 'medium', 'low'],
    ordered = True
)
tab = pd.crosstab(magAds['words'], magAds['education'])
tab
```

```
## education       high  medium  low
## words
## (30.999, 72.0]     4       6    5
## (72.0, 146.0]      5       6    8
## (146.0, 230.0]     9       6    5
```

- The event A={words=(146,230]} (the ad is a "difficult" text) has empirical probability
$$p_n(A) = \frac{9+6+5}{54} = \frac{20}{54} \approx 37\%.$$
- Say we only are interested in the probability of a "difficult" text (event A) for high education magazines, i.e. conditioning on the event $B = \{\text{education=high}\}$. Then the empirical conditional probability can be calculated from the table:
$$p_n(A \mid B) = \frac{9}{4+5+9} = \frac{9}{18} = 0.5 = 50\%.$$
- The conditional probability of A given B may theoretically be expressed as
$$\begin{aligned} P(A \mid B) &= P(\text{words} = (146, 230] \mid \text{education} = \text{high}) \\ &= \frac{P(\text{words} = (146, 230] \text{ and education} = \text{high})}{P(\text{education} = \text{high})}, \end{aligned}$$
which translated to empirical probabilities (substituting P with p_n) will give
$$\begin{aligned} p_n(A \mid B) &= \frac{p_n(\text{words} = (146, 230] \text{ and education} = \text{high})}{p_n(\text{education} = \text{high})} \\ &= \frac{\frac{9}{54}}{\frac{4+5+9}{54}} \\ &= \frac{9}{4+5+9} \\ &= 50\% \end{aligned}$$
as calculated above.

9.7 Conditional probability and independence

- If information about B does not change the probability of A we talk about independence, i.e. A is **independent** of B if
$$P(A \mid B) = P(A) \quad \Leftrightarrow \quad P(A \text{ and } B) = P(A)P(B)$$
The last relation is symmetric in A and B, and we simply say that A and B are **independent events**.
- In general the events $A_1, A_2, ..., A_k$ are independent if
$$P(A_1 \text{ and } A_2 \text{ and } ... \text{ and } A_k) = P(A_1)P(A_2)\cdots P(A_k).$$

9.7.1 Magazine data revisited

- Recall the empirical probabilities calculated above:

$$p_n(A) = 37\% \quad \text{and} \quad p_n(A \mid B) = 50\%.$$

- These indicate (we cannot say for sure as we only consider a finite sample - we will later see how to test for this) that the theoretical probability

$$P(A) \neq P(A \mid B)$$

and hence that knowledge about B (high education level) may convey information about the probability of A (the ad containing a "difficult" text).

9.8 Discrete distribution

9.8.1 Example: Magazine data

```
# Table with the percentage of ads in each combination of the levels of 'words' and 'education'
tab_counts = pd.crosstab(magAds['words'], magAds['education'])
tab_percent = (tab_counts / tab_counts.sum().sum()) * 100
tab_percent.round(2)
```

```
## education        high  medium    low
## words
## (30.999, 72.0]   7.41   11.11   9.26
## (72.0, 146.0]    9.26   11.11  14.81
## (146.0, 230.0]  16.67   11.11   9.26
```

- The 9 disjoint events above (corresponding to combinations of `words` and `education`) make up the whole sample space for the two variables. The empirical probabilities of each event is given in the table.

9.8.2 General discrete distribution

- In general:
  - Let $A_1, A_2, \ldots, A_k$ be a subdivision of the sample space into pairwise disjoint events.
  - The probabilities $P(A_1), P(A_2), ..., P(A_k)$ are called a **discrete distribution** and satisfy

$$\sum_{i=1}^{k} P(A_i) = 1.$$

9.8.3 Example: 3 coin tosses

- **Random/stochastic variable**: A function Y that translates an outcome of the experiment into a number.
- Possible outcomes in an experiment with 3 coin tosses:
  - 0 heads (TTT)
  - 1 head (HTT, THT, TTH)
  - 2 heads (HHT, HTH, THH)
  - 3 heads (HHH)
- The above events are disjoint and make up the whole sample space.
- Let Y be the number of heads in the experiment: $Y(TTT) = 0, Y(HTT) = 1, \ldots$
- Assume that each outcome is equally likely, i.e. probability $1/8$ for each event. Then,

- $P(\text{no heads}) = P(Y = 0) = P(TTT) = 1/8$.
- $P(\text{one head}) = P(Y = 1) = P(HTT \text{ or } THT \text{ or } TTH) = P(HTT) + P(THT) + P(TTH) = 3/8$.
- Similarly for 2 or 3 heads.

- So, the distribution of Y is

Number of heads, Y	0	1	2	3
Probability	1/8	3/8	3/8	1/8

10 Distribution of general random variables

10.1 Probability distribution

- We are conducting an experiment where we make a quantitative measurement Y (a random variable), e.g. the number of words in an ad or the waiting time in a queue.
- In advance there are many possible outcomes of the experiment, i.e. Y's value has an uncertainty, which we quantify by the **probability distribution** of Y.
- For any interval (a, b), the distribution states the probability of observing a value of the random variable Y in this interval:
$$P(a < Y < b), \qquad -\infty < a < b < \infty.$$
- Y is **discrete** if we can enumerate all the possible values of Y, e.g. the number of words in an ad.
- Y is **continuous** if Y can take any value in a interval, e.g. a measurement of waiting time in a queue.

10.1.1 Sample

We conduct an experiment n times, where the outcome of the ith experiment corresponds to a measurement of a random variable Y_i, where we assume

- The experiments are **independent**
- The variables $Y_1, \ldots, Y_n$ have the **same distribution**

10.2 Population parameters

- When the sample size grows, then e.g. the mean of the sample, $\overline{y}$, will stabilize around a fixed value, μ, which is usually unknown. The value μ is called the **population mean**.
- Correspondingly, the standard deviation of the sample, s, will stabilize around a fixed value, σ, which is usually unknown. The value σ is called the **population standard deviation**.
- Notation:
  - μ (mu) denotes the population mean.
  - σ (sigma) denotes the population standard deviation.

Population	Sample
μ	$\overline{y}$
σ	s

10.2.1 Distribution of a discrete random variable

- Possible values for Y: $\{y_1, y_2, \ldots, y_k\}$.
- The **distribution** of Y is the probabilities of each possible value: $p_i = P(Y = y_i), \quad i = 1, 2, \ldots, k$.
- The distribution satisfies: $\sum_{i=1}^{k} p_i = 1$.

10.3 Expected value (mean) for a discrete distribution

- The **expected value** or **(population) mean** of Y is

$$\mu = \sum_{i=1}^{k} y_i p_i$$

- An important property of the expected value is that it has the same unit as the observations (e.g. meter).

10.3.1 Example: number of heads in 3 coin flips

- Recall the distribution of Y (number of heads):

y (number of heads)	0	1	2	3
$P(Y = y)$	1/8	3/8	3/8	1/8

- Then the expected value is

$$\mu = 0\frac{1}{8} + 1\frac{3}{8} + 2\frac{3}{8} + 3\frac{1}{8} = 1.5.$$

Note that the expected value is not a possible outcome of the experiment itself.

10.4 Variance and standard deviation for a discrete distribution

- The **(population) variance** of Y is

$$\sigma^2 = \sum_{i=1}^{k} (y_i - \mu)^2 p_i$$

- The **(population) standard deviation** is $\sigma = \sqrt{\sigma^2}$.
- Note: If the observations have unit meter, the **variance** has unit meter^2 which is hard to interpret. The **standard deviation** on the other hand has the same unit as the observations (e.g. meter).

10.4.1 Example: number of heads in 3 coin flips

The distribution of the random variable 'number of heads in 3 coin flops' has variance

$$\sigma^2 = (0 - 1.5)^2\frac{1}{8} + (1 - 1.5)^2\frac{3}{8} + (2 - 1.5)^2\frac{3}{8} + (3 - 1.5)^2\frac{1}{8} = 0.75.$$

and standard deviation

$$\sigma = \sqrt{\sigma^2} = \sqrt{0.75} = 0.866.$$

10.5 The binomial distribution

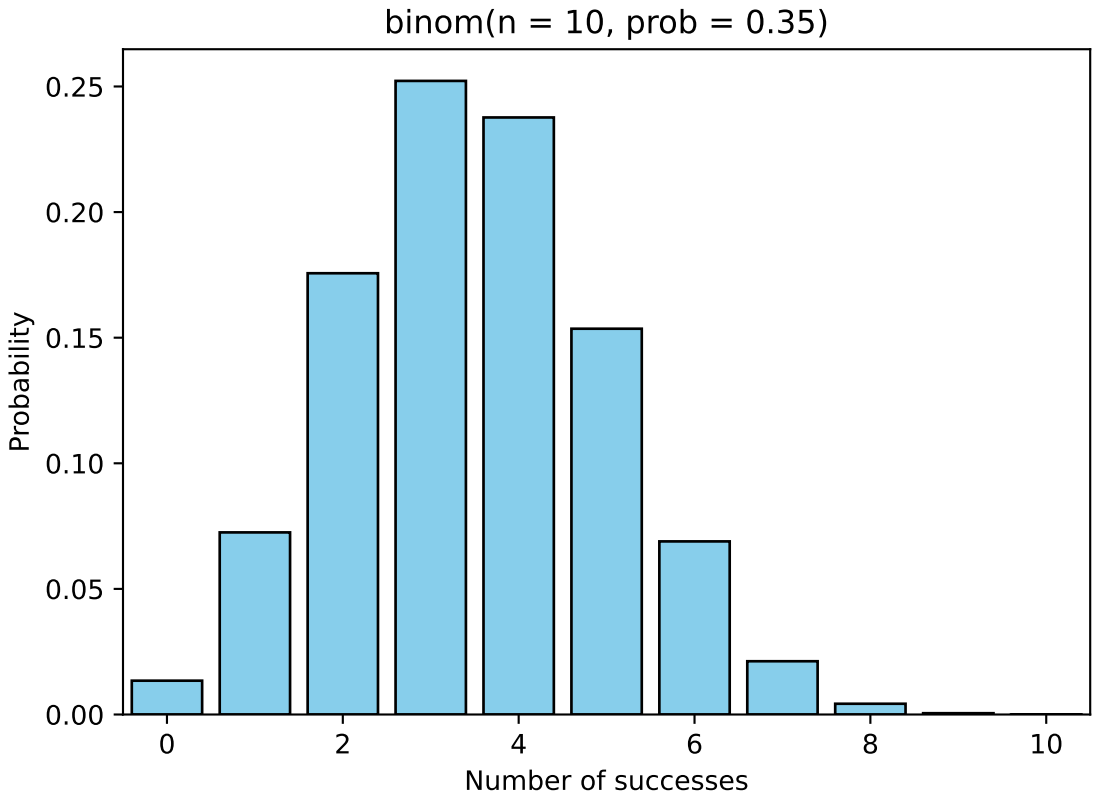
- The **binomial distribution** is a discrete distribution
- The distribution occurs when we conduct a success/failure experiment n times with probability π for success. If Y denotes the number of successes it can be shown that

$$p_Y(y) = P(Y = y) = \binom{n}{y}\pi^y(1 - \pi)^{n-y},$$

where $\binom{n}{y} = \frac{n!}{y!(n-y)!}$ and $m!$ is the product of the first m integers.
- Expected value: $\mu = n\pi$.

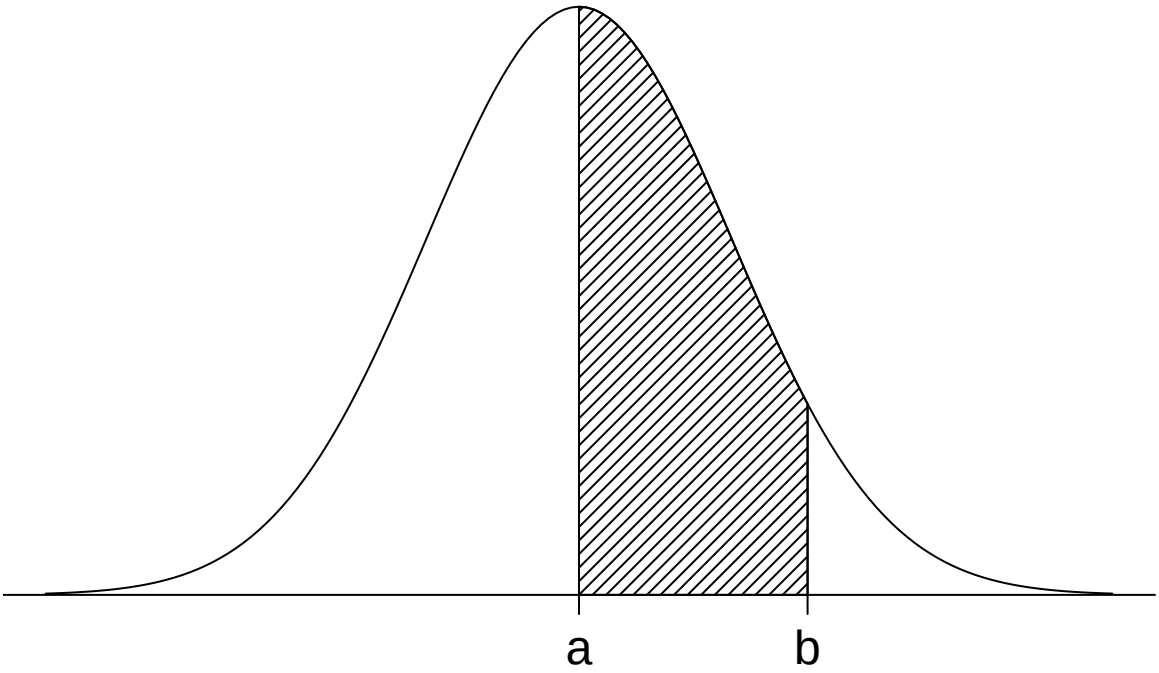
- Variance: $\sigma^2 = n\pi(1-\pi)$.
- Standard deviation: $\sigma = \sqrt{n\pi(1-\pi)}$.

```
import matplotlib.pyplot as plt
from scipy.stats import binom
n = 10
p = 0.35
x = np.arange(0, n + 1)
pmf_values = binom.pmf(x, n, p)
plt.bar(x, pmf_values, width = 0.8, color = "skyblue", edgecolor = "black")
plt.xlim(-0.5, n + 0.5);
plt.xlabel("Number of successes");
plt.ylabel("Probability");
plt.title("binom(n = 10, prob = 0.35)");
```



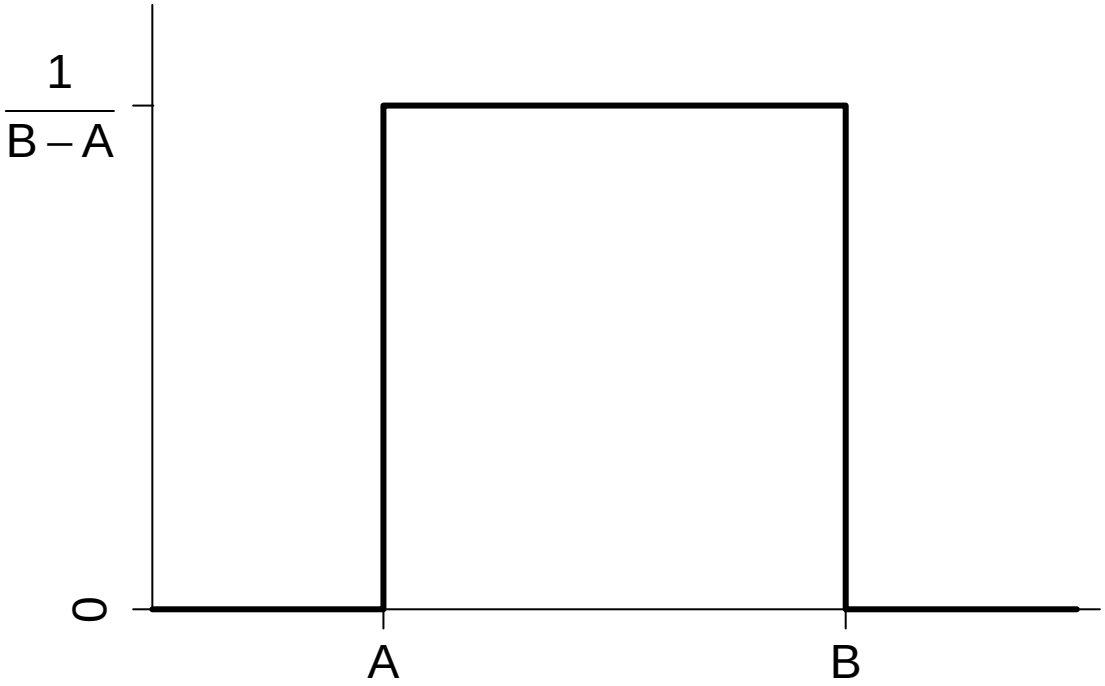
10.6 Distribution of a continuous random variable

- The distribution of a continuous random variable Y is characterized by the so-called probability density function f_Y.



- The area under the graph of the probability density function between a and b is equal to the probability of an observation in this interval.
- $f_Y(y) \geq 0$ for all real numbers y.
- The area under the graph for f_Y is equal to 1.
- For example the **uniform distribution** from A to B:

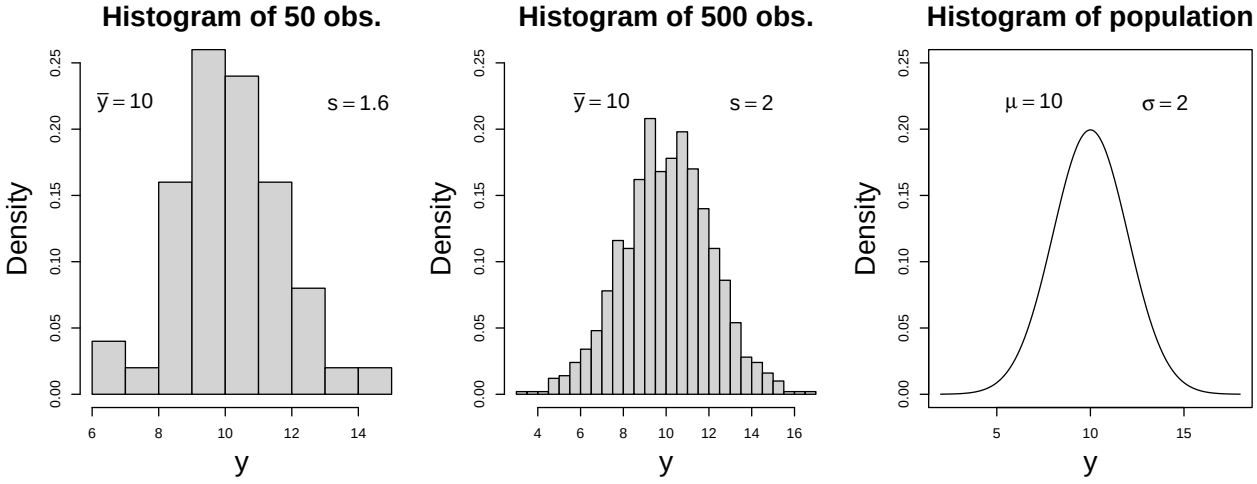
$$f_Y(y) = \begin{cases} \frac{1}{B-A} & A \leq y \leq B \\ 0 & \text{otherwise} \end{cases}$$



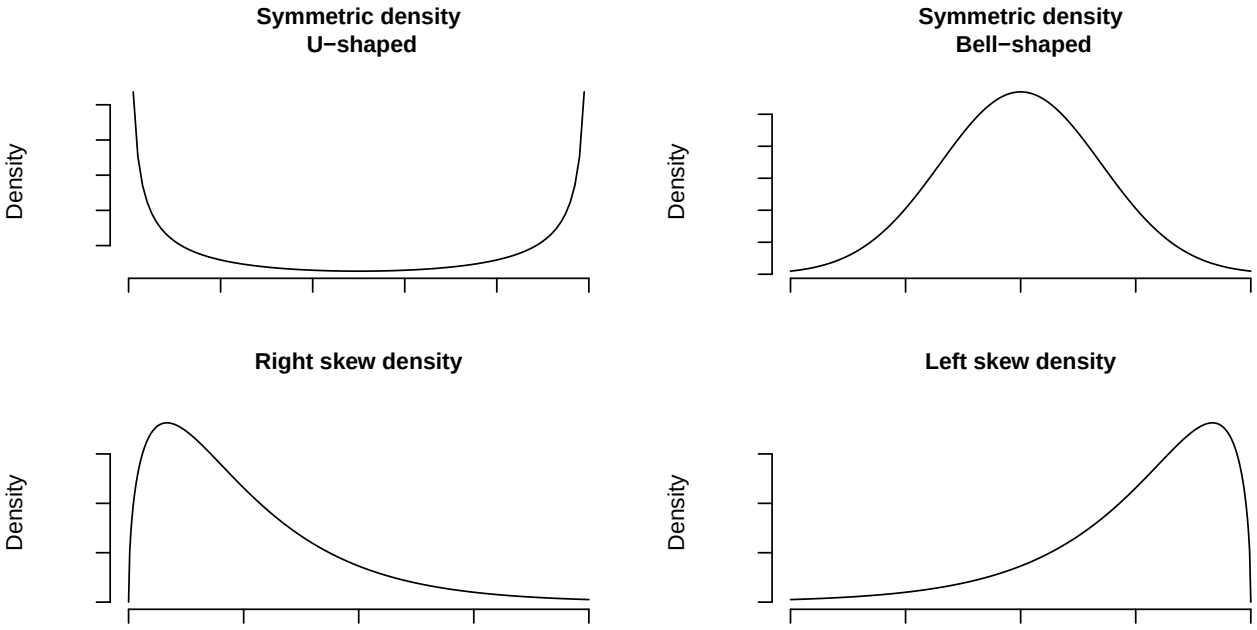
10.7 Density function

10.7.1 Increasing number of observations

- Another way to think about the density is in terms of the histogram.
- If we draw a histogram for a sample where the area of each box corresponds to the relative frequency of each interval, then the total area will be 1.
- When the number of observations (sample size) increase we can make a finer interval division and get a more smooth histogram.
- We can imagine an infinite number of observations, which would produce a nice smooth curve, where the area below the curve is 1. A function derived this way is also what we call the **probability density function**.



10.7.2 Density shapes



10.8 Normal distribution

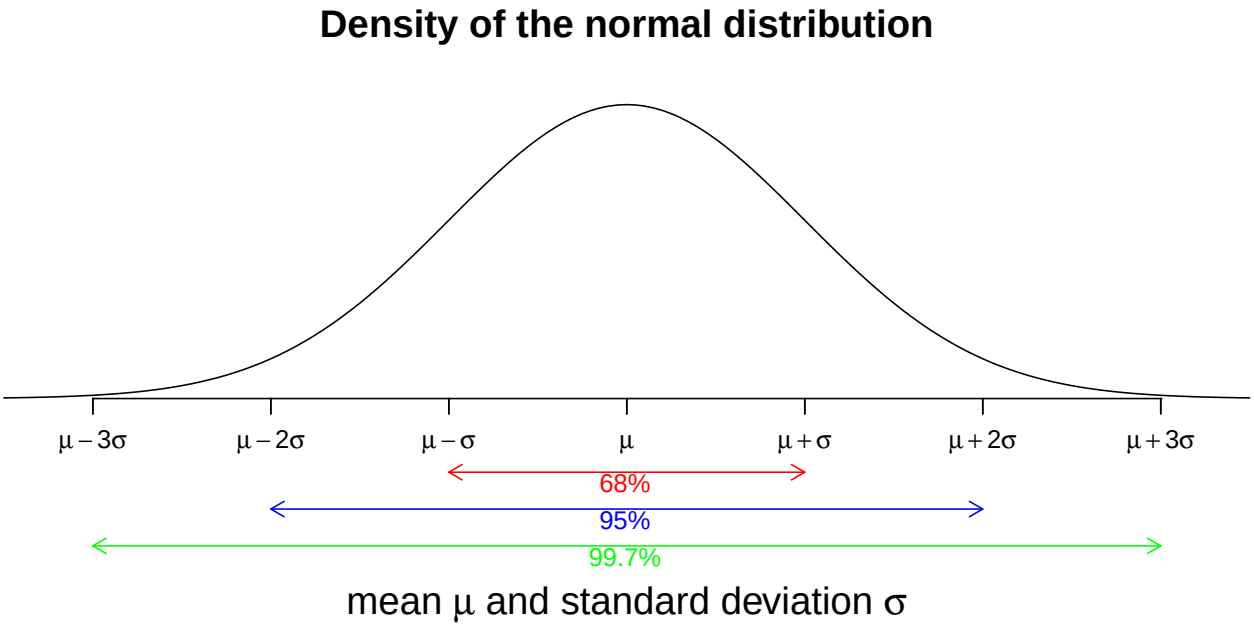
- The normal distribution is a continuous distribution determined by two parameters:

  - μ: the **mean** (expected value), which determines where the distribution is centered.
  - σ: the **standard deviation**, which determines the spread of the distribution about the mean.
- The distribution has a bell-shaped probability density function:

$$f_Y(y;\mu,\sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}(y-\mu)^2\right)$$

- When a random variable Y follows a normal distribution with mean μ and standard deviation σ, then we write $Y \sim \texttt{norm}(\mu, \sigma)$.
- We call $\texttt{norm}(0, 1)$ the **standard normal distribution**.

10.8.1 Reach of the normal distribution



Interpretation of standard deviation:

- $\approx 68\%$ of the population is within 1 standard deviation of the mean.
- $\approx 95\%$ of the population is within 2 standard deviations of the mean.
- $\approx 99.7\%$ of the population is within 3 standard deviations of the mean.

10.8.2 Normal z-score

- If $Y \sim \texttt{norm}(\mu, \sigma)$ then the corresponding so-called z-score is

$$Z = \frac{Y-\mu}{\sigma} = \frac{\texttt{observation} - \texttt{mean}}{\texttt{standard deviation}}$$

- I.e. Z counts the number of standard deviations that the observation lies away from the mean, where a negative value tells that we are below the mean.
- We have that $Z \sim \texttt{norm}(0, 1)$, i.e. Z has zero mean and standard deviation one.
- This implies that
  - Z lies between -1 and 1 with probability 68%
  - Z lies between -2 and 2 with probability 95%
  - Z lies between -3 and 3 with probability 99.7%
- It also implies that:

– The probability of Y being between $\mu - z\sigma$ and $\mu + z\sigma$ is equal to the probability of Z being between $-z$ and z.

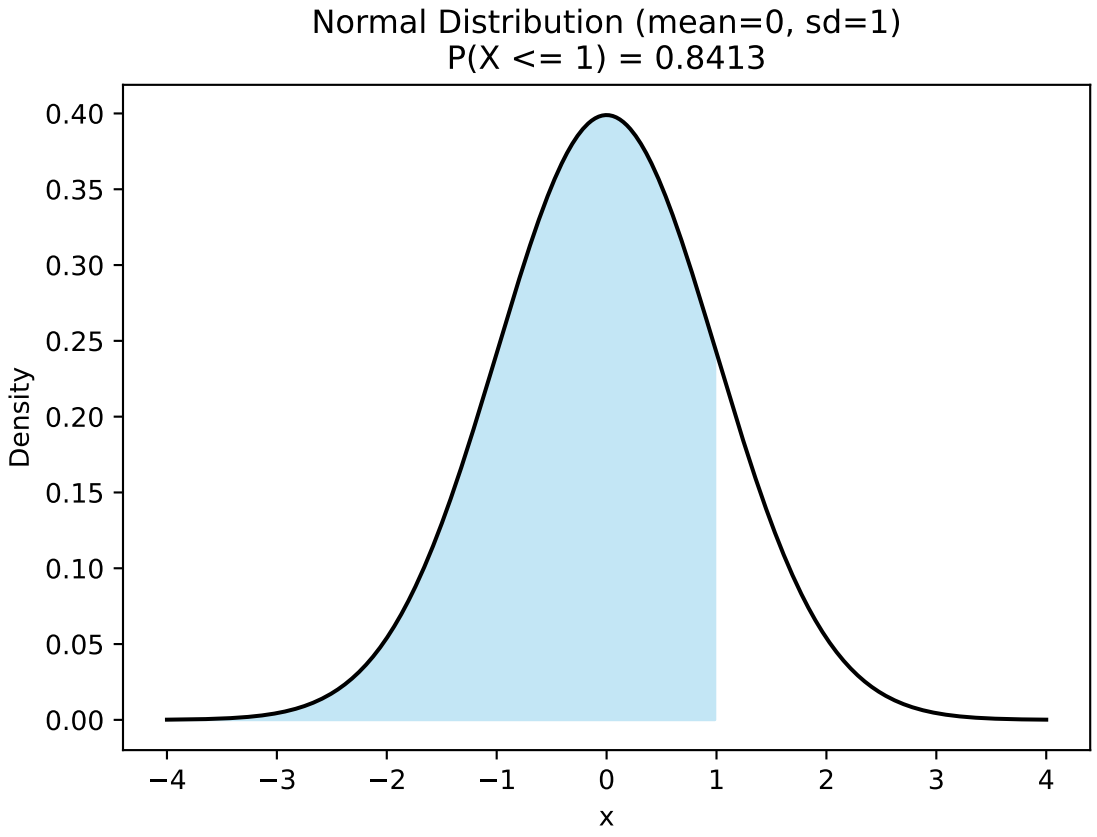
10.8.3 Calculating probabilities in the standard normal distribution

- The function `norm.cdf` always outputs the area to the left of the z-value (quantile/percentile) we give as input (variable `q` in the function), i.e. it outputs the probability of getting a value less than z.

```
# For a standard normal distribution the probability of getting a value less than 1 is:
from scipy.stats import norm
mean = 0
sd = 1
q = 1
left_prob = norm.cdf(q, loc = mean, scale = sd)
left_prob
```

```
## np.float64(0.8413447460685429)
```

```
x = np.linspace(mean - 4*sd, mean + 4*sd, 500)
y = norm.pdf(x, loc = mean, scale = sd)
plt.plot(x, y, color = 'black')
plt.fill_between(x, y, 0, where = (x <= q), color = 'skyblue', alpha = 0.5)
plt.xlabel('x')
plt.ylabel('Density')
plt.title(f'Normal Distribution (mean={mean}, sd={sd})\nP(X <= {q}) = {left_prob:.4f}')
```



So `q=1` corresponds to the 0.84-percentile/quantile for the standard normal distribution

- Here there is a conflict with the textbook, since the book always considers right tail probabilities. Since the total area is 1 and we have the left probability we easily get the right probability:

```
right_prob = 1 - left_prob
right_prob.round(4)
```

```
## np.float64(0.1587)
```

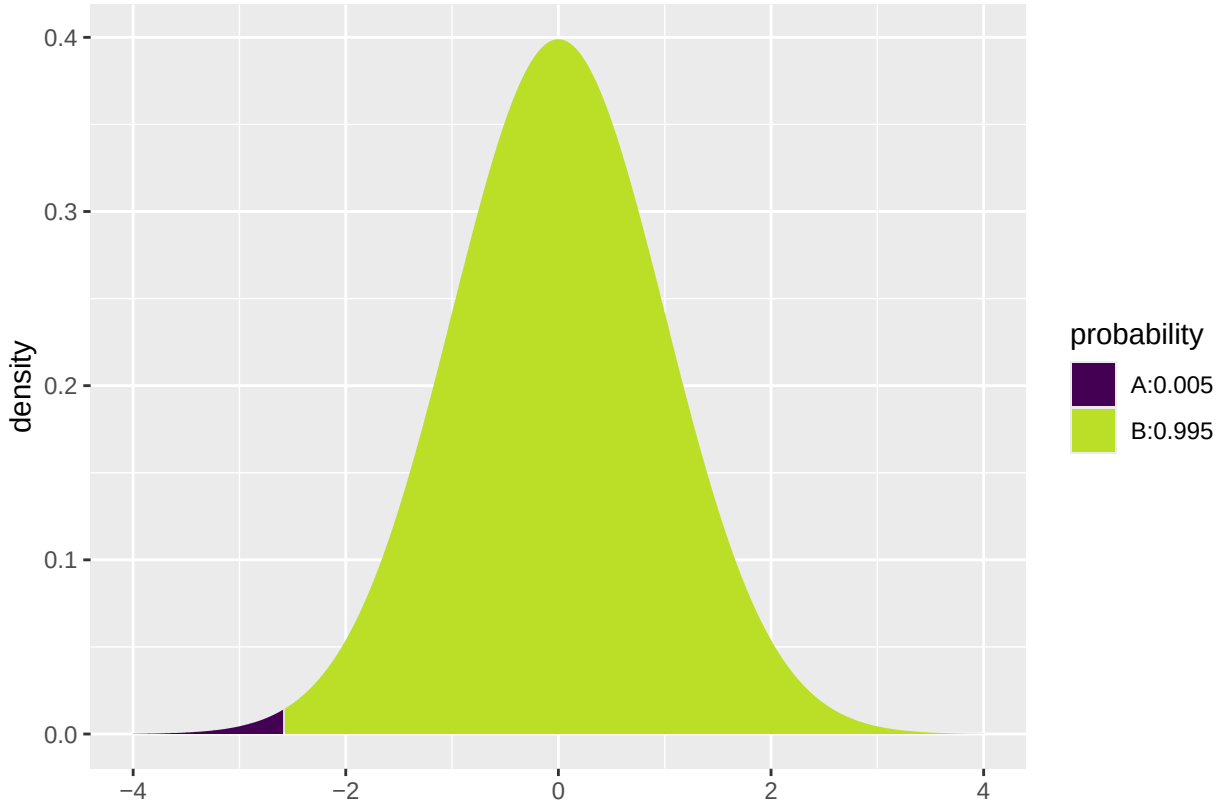
- For $z = 1$ we have a right probability of $p = 0.1587$, so the probability of an observation between -1 and 1 is equal to $1 - 2 \cdot 0.1587 = 0.6826 = 68.26\%$ due to symmetry.

10.8.4 Calculating z-values (quantiles) in the standard normal distribution

- If we have a probability and want to find the corresponding z-value we again need to decide on left/right probability. The default is to find the left probability, so if we want the z-value with e.g. 0.5% probability to the left we get:

```
left_z = norm.ppf(q = 0.005, loc = 0, scale = 1)
left_z
```

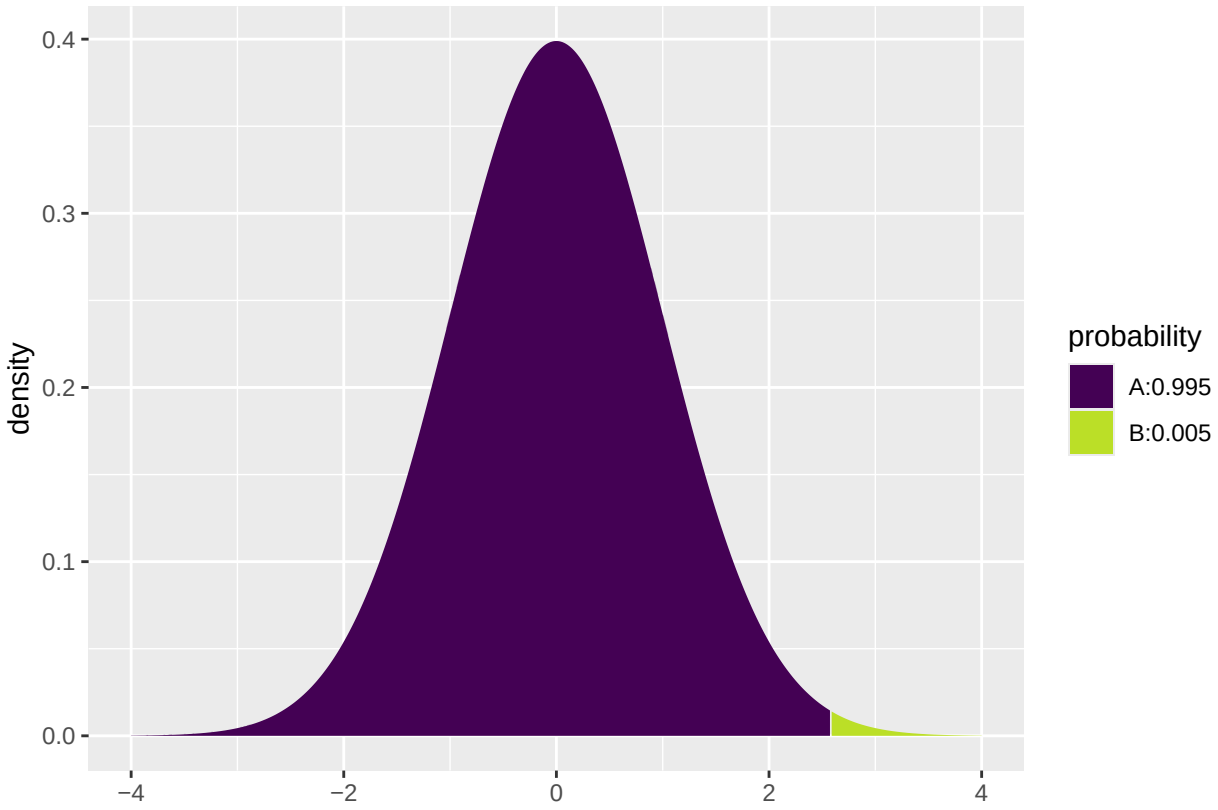
```
## np.float64(-2.575829303548901)
```



- However, in all the formulas in the course we follow the textbook and consider z-values for a given right probability. E.g. with 0.5% probability to the right we get:

```
right_z = norm.ppf(q = 1 - 0.005, loc = 0, scale = 1)
right_z
```

```
## np.float64(2.5758293035489004)
```



- Thus, the probability of an observation between -2.576 and 2.576 equals $1 - 2 \cdot 0.005 = 99\%$.

10.8.5 Example

The Stanford-Binet Intelligence Scale is calibrated to be approximately normal with mean 100 and standard deviation 16.

What is the 99-percentile of IQ scores?

- The corresponding z-score is $Z = \frac{IQ-100}{16}$, which means that $IQ = 16Z + 100$.
- The 99-percentile of z-scores has the value 2.326 (can be calculated using `norm.ppf`).
- Then, the 99-percentile of IQ scores is:

$$IQ = 16 \cdot 2.326 + 100 = 137.2.$$

- So we expect that one out of hundred has an IQ exceeding 137.

11 Distribution of sample statistic

11.1 Estimates and their variability

We are given a sample $y_1, y_2, \ldots, y_n$.

- The sample mean $\bar{y}$ is the most common estimate of the population mean μ.
- The sample standard deviation, s, is the most common estimate of the population standard deviation σ.

  We notice that there is an uncertainty (from sample to sample) connected to these statistics and therefore we are interested in describing their **distribution**.

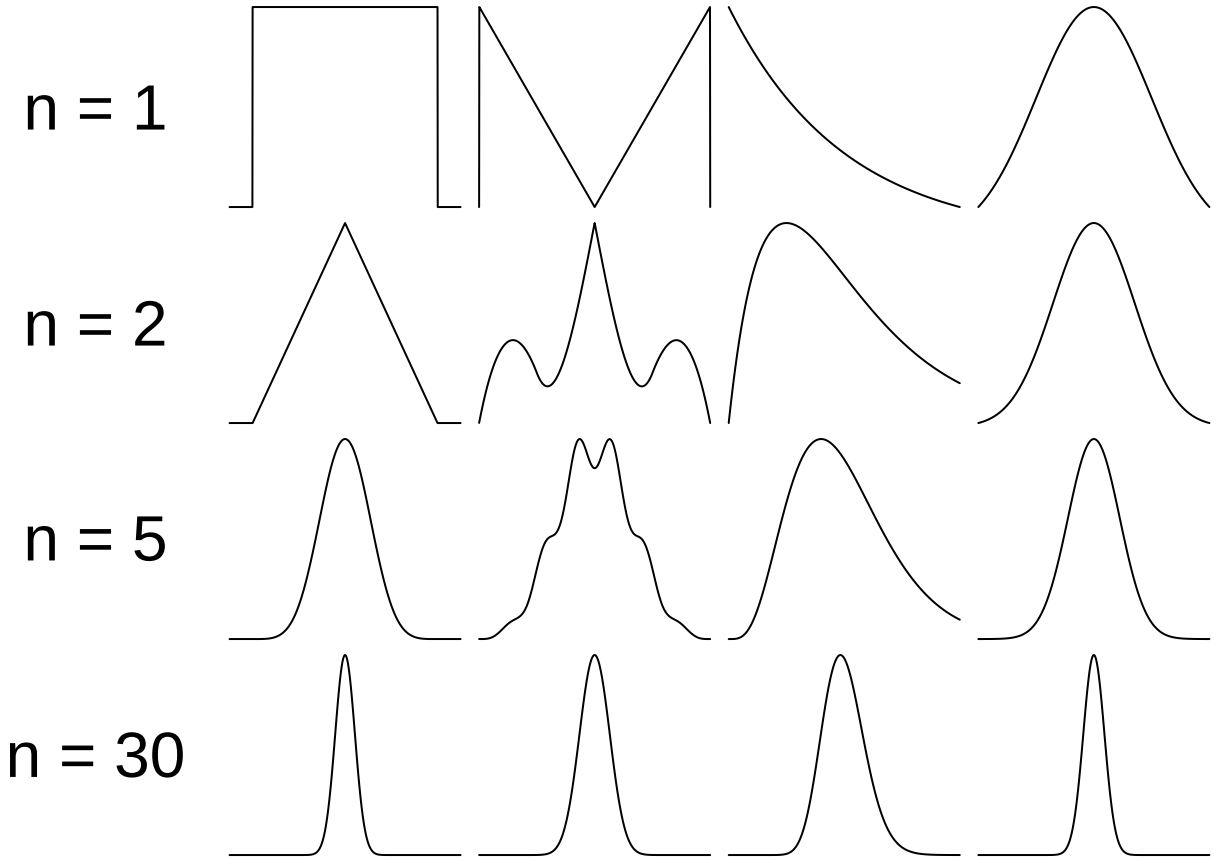
11.2 Distribution of sample mean

- We are given a sample $y_1, y_2, \ldots, y_n$ from a population with mean μ and standard deviation σ.
- The sample mean
$$\bar{y} = \frac{1}{n}(y_1 + y_2 + \ldots + y_n)$$
then has a distribution where
  - the distribution has mean μ,
  - the distribution has standard deviation $\frac{\sigma}{\sqrt{n}}$ (also called the **standard error**), and
  - when n grows, the distribution approaches a normal distribution. This result is called **the central limit theorem**.

11.2.1 Central limit theorem

- The points above can be summarized as
$$\bar{y} \approx \texttt{norm}(\mu, \frac{\sigma}{\sqrt{n}})$$
that is, based on the sample $y_1, \ldots, y_n$ of size n the sample mean $\bar{y}$ is approximately normally distributed with mean μ and standard error $\frac{\sigma}{\sqrt{n}}$.
- When our sample is sufficiently large (such that the above approximation is good) this allows us to make the following observations:
  - We are 95% certain that $\bar{y}$ lies in the interval from $\mu - 2\frac{\sigma}{\sqrt{n}}$ to $\mu + 2\frac{\sigma}{\sqrt{n}}$.
  - We are almost completely certain that $\bar{y}$ lies in the interval from $\mu - 3\frac{\sigma}{\sqrt{n}}$ to $\mu + 3\frac{\sigma}{\sqrt{n}}$.
- This is not useful when μ is unknown, but let us rephrase the first statement to:
  - We are 95% certain that μ lies in the interval from $\bar{y} - 2\frac{\sigma}{\sqrt{n}}$ to $\bar{y} + 2\frac{\sigma}{\sqrt{n}}$, i.e. we are directly talking about the uncertainty of determining μ.

11.2.2 Illustration of CLT



- Four different population distibutions (n=1) of y and corresponding sampling distributions of $\bar{y}$ for different sample sizes. As n increases the sampling distributions become narrower and more bell-shaped.

11.2.3 Example

- Body Mass Index (BMI) of people in Northern Jutland (2010) has mean $\mu = 25.8$ kg/m^2 and standard deviation 4.8 kg/m^2.
- A random sample of $n = 100$ costumers at a burger bar had an average BMI given by $\bar{y} = 27.2$.
- If "burger bar" has "no influence" on BMI (and the sample is representative of the population/people in Northern Jutland), then

$$\bar{y} \approx \texttt{norm}(\mu, \frac{\sigma}{\sqrt{n}}) = \texttt{norm}(25.8, 0.48).$$

- For the actual sample this gives the observed z-score

$$z_{obs} = \frac{27.2 - 25.8}{0.48} = 2.92$$

- Recalling that the z-score is (here approximately) standard normal, the probability of getting a higher z-score is:

```
1 - norm.cdf(2.92, loc=0, scale=1)
```

```
## np.float64(0.0017501569286760832)
```

- Thus, it is highly unlikely to get a random sample with such a high z-score. This indicates that costumers at the burger bar has a mean BMI, which is higher than the population mean.

12 Point and interval estimates

12.1 Point and interval estimates

- We want to study hypotheses for population parameters, e.g. the mean μ and the standard deviation σ.
  - If μ is e.g. the mean waiting time in a queue, then it might be relevant to investigate whether it exceeds 2 minutes.
- Based on a sample we make a **point estimate** which is a guess of the parameter value.
  - For instance we have used $\bar{y}$ as an estimate of μ and s as an estimate of σ.
- We often want to supplement the point estimate with an **interval estimate** (also called a **confidence interval**). This is an interval around the point estimate, in which we are confident (to a certain degree) that the population parameter is located.
- The parameter estimate can then be used to investigate our hypothesis.

12.2 Point estimators: Bias

- If we want to estimate the population mean μ we have several possibilities e.g.
  - the sample mean $\bar{y}$
  - the average y_T of the sample upper and lower quartiles
- Advantage of y_T: Very large/small observations have little effect, i.e. it has practically no effect if there are a few errors in the data set.
- Disadvantage of y_T: If the distribution of the population is skewed, i.e. asymmetrical, then y_T is **biased**, meaning that in the long run this estimator systematically over or under estimates the value of μ.
- Generally we prefer that an estimator is **unbiased**, i.e. its distribution is centered around the true parameter value.
- Recall that for a sample from a population with mean μ, the sample mean $\bar{y}$ also has mean μ. That is, $\bar{y}$ is an unbiased estimate of the population mean μ.

12.3 Point estimators: Consistency

- From previous lectures we know that the standard error of $\bar{y}$ is $\frac{\sigma}{\sqrt{n}}$,
  - i.e. the standard error decrease when the sample size increase.
- In general an estimator with this property is callled **consistent**.
- y_T is also a consistent estimator, but has a variance that is greater than $\bar{y}$.

12.4 Point estimators: Efficiency

- Since the variance of y_T is greater than the variance of $\bar{y}$, $\bar{y}$ is preferred.
- In general we prefer the estimator with the smallest possible variance.
  - This estimator is said to be **efficient**.
- $\bar{y}$ is an efficient estimator.

12.5 Notation

- The symbol $\hat{}$ above a parameter is often used to denote a (point) estimate of the parameter. We have looked at an
  - estimate of the population mean: $\hat{\mu} = \bar{y}$
  - estimate of the population standard deviation: $\hat{\sigma} = s$
- When we observe a 0/1 variable, which e.g. is used to denote yes/no or male/female, then we will use the notation
$$\pi = P(Y = 1)$$
for the proportion of the population with the property $Y = 1$.
- The estimate $\hat{\pi} = (y_1 + y_2 + \ldots + y_n)/n$ is the relative frequency of the property $Y = 1$ in the sample.

12.6 Confidence Interval

- The general definition of a confidence interval for a population parameter is as follows:
  - A **confidence interval** for a parameter is constructed as an interval, where we expect the parameter to be.
  - The probability that this construction yields an interval which includes the parameter is called the **confidence level** and it is typically chosen to be 95%.
  - (1-confidence level) is called the **error probability** (in this case $1 - 0.95 = 0.05$, i.e. 5%).
- In practice the interval is often constructed as a symmetric interval around a point estimate:
  - **point estimate±margin of error**
  - Rule of thumb: With a margin of error of 2 times the standard error you get a confidence interval, where the confidence level is approximately 95%.
  - I.e: **point estimate ± 2 x standard error** has confidence level of approximately 95%.

12.7 Confidence interval for proportion

- Consider a population with a distribution where the probability of having a given characteristic is π and the probability of not having it is $1 - \pi$.
- When *no/yes* to the characteristic is denoted 0/1, i.e. y is 0 or 1, the distribution of y have a standard deviation of:

  $$\sigma = \sqrt{\pi(1-\pi)}.$$

  That is, the standard deviation is not a "free" parameter for a 0/1 variable as it is directly linked to the probability π.
- With a sample size of n the standard error of $\hat{\pi}$ will be (since $\hat{\pi} = \frac{\sum_{i=1}^{n} y_i}{n}$):

  $$\sigma_{\hat{\pi}} = \frac{\sigma}{\sqrt{n}} = \sqrt{\frac{\pi(1-\pi)}{n}}.$$

- We do not know π but we insert the estimate and get the **estimated standard error** of $\hat{\pi}$:

  $$se = \sqrt{\frac{\hat{\pi}(1-\hat{\pi})}{n}}.$$

- The rule of thumb gives that the interval

  $$\hat{\pi} \pm 2\sqrt{\frac{\hat{\pi}(1-\hat{\pi})}{n}}$$

  has confidence level of approximately 95%. I.e., before the data is known the random interval given by the formula above has approximately 95% probability of covering the true value π.

12.7.1 Example: Point and interval estimate for proportion

- Now we will have a look at a data set concerning votes in Chile. Information about the data can be found here.

```
import pandas as pd

Chile = pd.read_csv("https://asta.math.aau.dk/datasets?file=Chile.txt", sep = "\t")
```

- We focus on the variable `sex`, i.e. the gender distribution in the sample.

```
tab_counts = Chile['sex'].value_counts()
tab_counts
```

```
## sex
## F    1379
## M    1321
## Name: count, dtype: int64
```

```
tab_prop = Chile['sex'].value_counts(normalize = True)
tab_prop
```

```
## sex
## F    0.510741
## M    0.489259
## Name: proportion, dtype: float64
```

- Unknown population proportion of females (F), π.
- Estimate of π: $\hat{\pi} = \frac{1379}{1379+1321} = 0.5107$
- Rule of thumb : $\hat{\pi} \pm 2 \times se = 0.5107 \pm 2\sqrt{\frac{0.5107(1-0.5107)}{1379+1321}} = (0.49, 0.53)$ is an approximate 95% confidence interval for π.

12.7.2 Example: Confidence intervals for proportion in R

- We can calculate the confidence interval for the proportion of females when we do a so-called hypothesis test (we will get back to that later):

```
from statsmodels.stats.proportion import proportions_ztest, proportion_confint

counts = Chile['sex'].value_counts()
successes = counts["F"]
nobs = counts.sum()

stat, p_value = proportions_ztest(count = successes, nobs = nobs, value = 0.5)

ci_low, ci_high = proportion_confint(successes, nobs, alpha = 0.05, method = 'normal')

print(f"95% CI: ({ci_low:.4f}, {ci_high:.4f})")
```

```
## 95% CI: (0.4919, 0.5296)
```

```
print(f"sample estimate: {successes/nobs:.4f}")
```

```
## sample estimate: 0.5107
```

```
print(f"p-value: {p_value:.4g}")
```

```
## p-value: 0.2642
```

12.8 General confidence intervals for proportion

- Based on the central limit theorem we have:

$$\hat{\pi} \approx N\left(\pi, \sqrt{\frac{\pi(1-\pi)}{n}}\right)$$

  if $n\hat{\pi}$ and $n(1-\hat{\pi})$ both are at least 15.
- To construct a confidence interval with (approximate) confidence level $1-\alpha$:

1) Find the socalled **critical value** z_{crit} for which the upper tail probability in the standard normal distribution is $\alpha/2$.
2) Calculate $se = \sqrt{\frac{\hat{\pi}(1-\hat{\pi})}{n}}$
3) Then $\hat{\pi} \pm z_{crit} \times se$ is a confidence interval with confidence level $1 - \alpha$.

12.8.1 Example: Chile data

Compute for the `Chile` data set the 99% and 95%-confidence intervals for the probability that a person is female:

- For a 99%-confidence level we have $\alpha = 1\%$ and
  1) z_{crit}=`qdist("norm", 1 - 0.01/2)`=2.576.
  2) We know that $\hat{\pi} = 0.5107$ and $n = 2700$, so $se = \sqrt{\frac{\hat{\pi}(1-\hat{\pi})}{n}} = 0.0096$.
  3) Thereby, a 99%-confidence interval is: $\hat{\pi} \pm z_{crit} \times se = (0.4859, 0.5355)$.
- For a 95%-confidence level we have $\alpha = 5\%$ and
  1) z_{crit}=`qdist("norm", 1 - 0.05/2)`=1.96.
  2) Again, $\hat{\pi} = 0.5107$ and $n = 2700$ and so $se = 0.0096$.
  3) Thereby, we find as 95%-confidence interval $\hat{\pi} \pm z_{crit} \times se = (0.4918, 0.5295)$ (as the result of `prop.test`).

12.9 Confidence Interval for mean - normally distributed sample

- When it is reasonable to assume that the population distribution is normal we have the **exact** result

$$\bar{y} \sim \texttt{norm}(\mu, \frac{\sigma}{\sqrt{n}}),$$

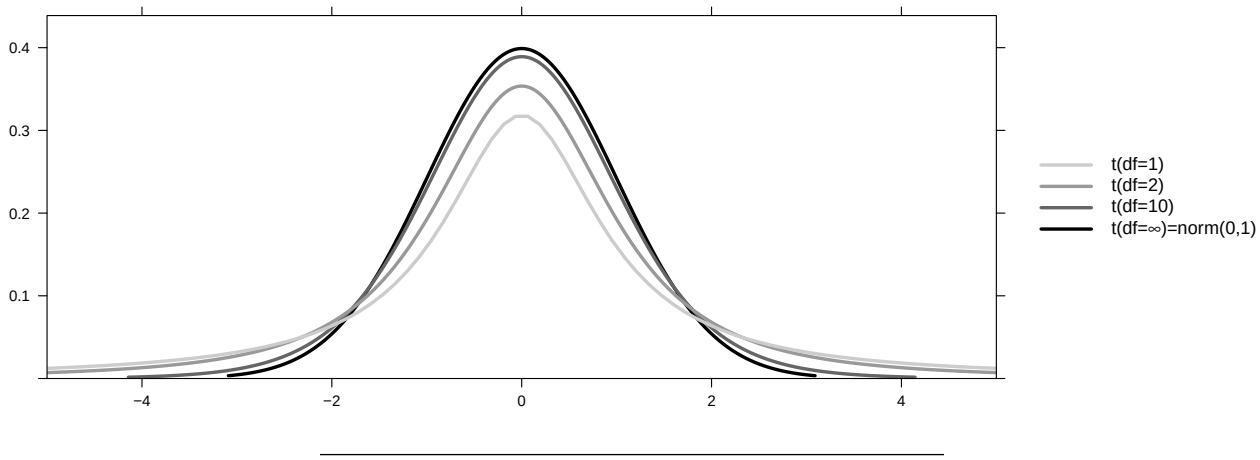
  i.e. $\bar{y} \pm z_{crit} \times \frac{\sigma}{\sqrt{n}}$ is not only an approximate but rather an exact confidence interval for the population mean, μ.
- In practice we **do not know** σ and instead we are forced to apply the sample standard deviation s to find the **estimated standard error** $se = \frac{s}{\sqrt{n}}$.
- This extra uncertainty, however, implies that an exact confidence interval for the population mean μ cannot be constructed using the z-score.
- Luckily, an exact interval can still be constructed by using the so-called t**-score**, which apart from the confidence level depends on the **degrees of freedom**, which are $df = n - 1$. That is the confidence interval now takes the form

$$\bar{y} \pm t_{crit} \times se.$$

12.10 t-distribution and t-score

- Calculation of t-score is based on the t**-distribution**, which is very similar to the standard normal z-distribution:
  - it is symmetric around zero and bell shaped, but
  - it has "heavier" tails and thereby
  - a slightly larger standard deviation than the standard normal distribution.
  - Further, the t-distribution's standard deviation decays as a function of its **degrees of freedom**, which we denote df.
  - and when df grows the t-distribution approaches the standard normal distribution.

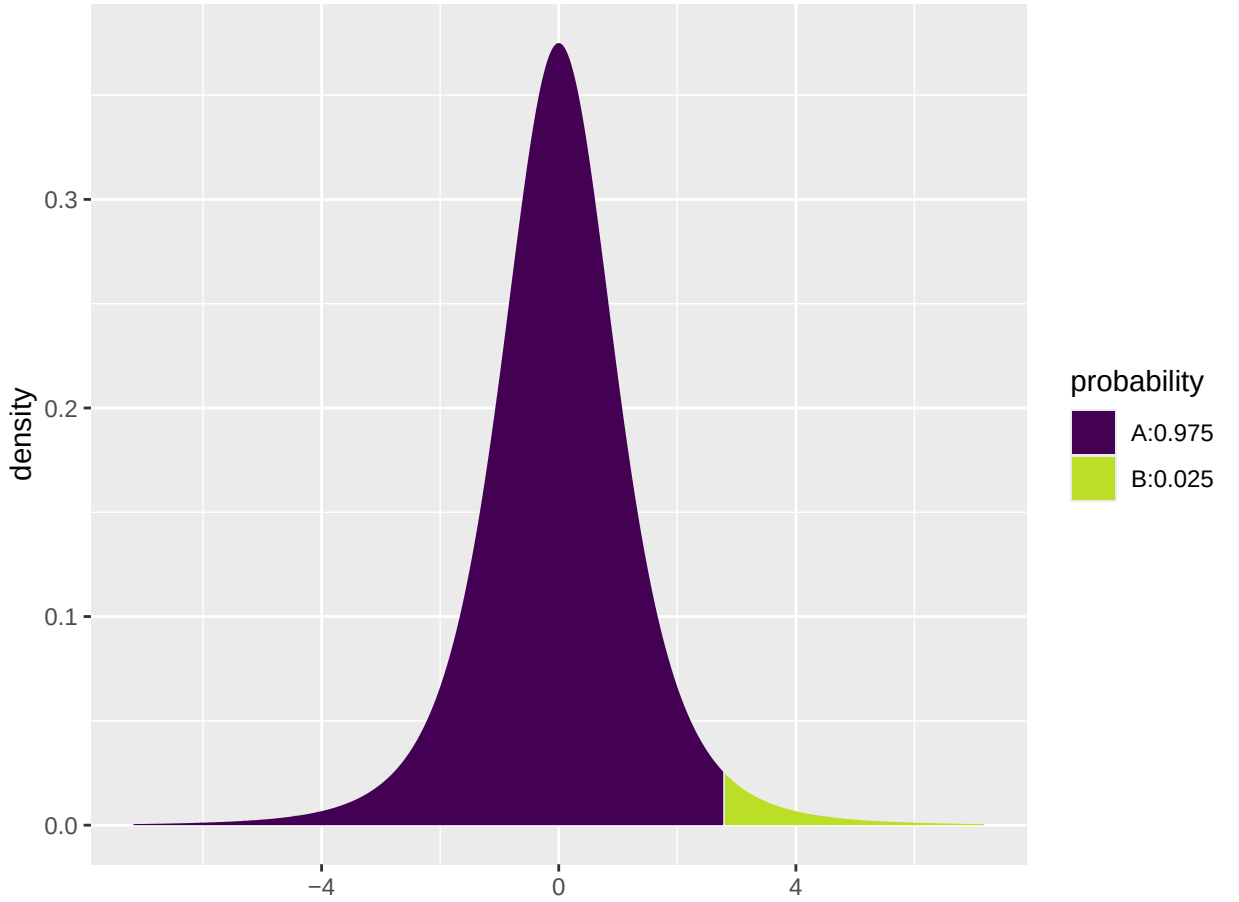
The expression of the density function is of slightly complicated form and will not be stated here, instead the t-distribution is plotted below for $df = 1, 2, 10$ and ∞.



12.11 Calculation of t-score

```
from scipy.stats import t
t.ppf(1 - 0.025, df = 4)
```

```
## np.float64(2.7764451051977987)
```



- We seek the quantile (i.e. value on the x-axis) such that we have a given **right tail** probability. This is the critical t-score associated with our desired level of confidence.
- To get e.g. the t-score corresponding to a right tail probability of 2.5 % we have to look up the 97.5 %

quantile using `ppf` with `p = 1 - 0.025` since `ppf` looks at the area to the **left hand side**.

- The degrees of freedom are determined by the sample size; in this example we just used df $= 4$ for illustration.
- As the t-score giving a right probability of 2.5 % is 2.776 and the t-distribution is symmetric around 0, we have that an observation falls between -2.776 and 2.776 with probability 1 - 2 $\cdot$ 0.025 $=$ 95 % for a t-distribution with 4 degrees of freedom.

12.12 Example: Confidence interval for mean

- We return to the dataset `Ericksen` and want to construct a 95% confidence interval for the population mean μ of the variable `crime`.

```
import numpy as np

Ericksen = pd.read_csv("https://asta.math.aau.dk/datasets?file=Ericksen.txt", sep = "\t")

stats = Ericksen['crime'].agg(
    mean = 'mean',
    std = 'std',
    n = 'count'
)
stats
```

```
## mean    63.060606
## std     24.891073
## n       66.000000
## Name: crime, dtype: float64
```

```
df = stats["n"] - 1
t_crit = t.ppf(1 - 0.025, df = df)
t_crit
```

```
## np.float64(1.9971379083920033)
```

- I.e. we have
  - $\bar{y} = 63.06$
  - $s = 24.89$
  - $n = 66$
  - $df = n - 1 = 65$
  - $t_{crit} = 2.$
- The confidence interval is $\bar{y} \pm t_{crit} \frac{s}{\sqrt{n}} = (56.94, 69.18)$
- All these calculations can be done automatically:

```
import pingouin as pg
res = pg.ttest(x = Ericksen['crime'], y = 0, confidence = 0.95)
res["mean"] = Ericksen['crime'].mean()
res[["mean", "T", "dof", "alternative", "p-val", "CI95%"]]
```

```
##              mean          T  dof alternative         p-val           CI95%
## T-test  63.060606  20.581949   65   two-sided  3.564824e-30  [56.94, 69.18]
```

12.13 Example: Plotting several confidence intervals

- We shall look at a dataset `chickwts`:

An experiment was conducted to measure and compare the effectiveness of various feed supplements on the growth rate of chickens. Newly hatched chicks were randomly allocated into six groups, and each group was given a different feed supplement. Their weights in grams after six weeks are given along with feed types.

- `chickwts` is a data frame with 71 observations on 2 variables:
  - `weight`: a numeric variable giving the chick weight.
  - `feed`: a factor giving the feed type.
- Calculate a confidence interval for the mean weight for each feed separately; the confidence interval is from `lower` to `upper` given by `mean`$\pm$`tscore * se`:

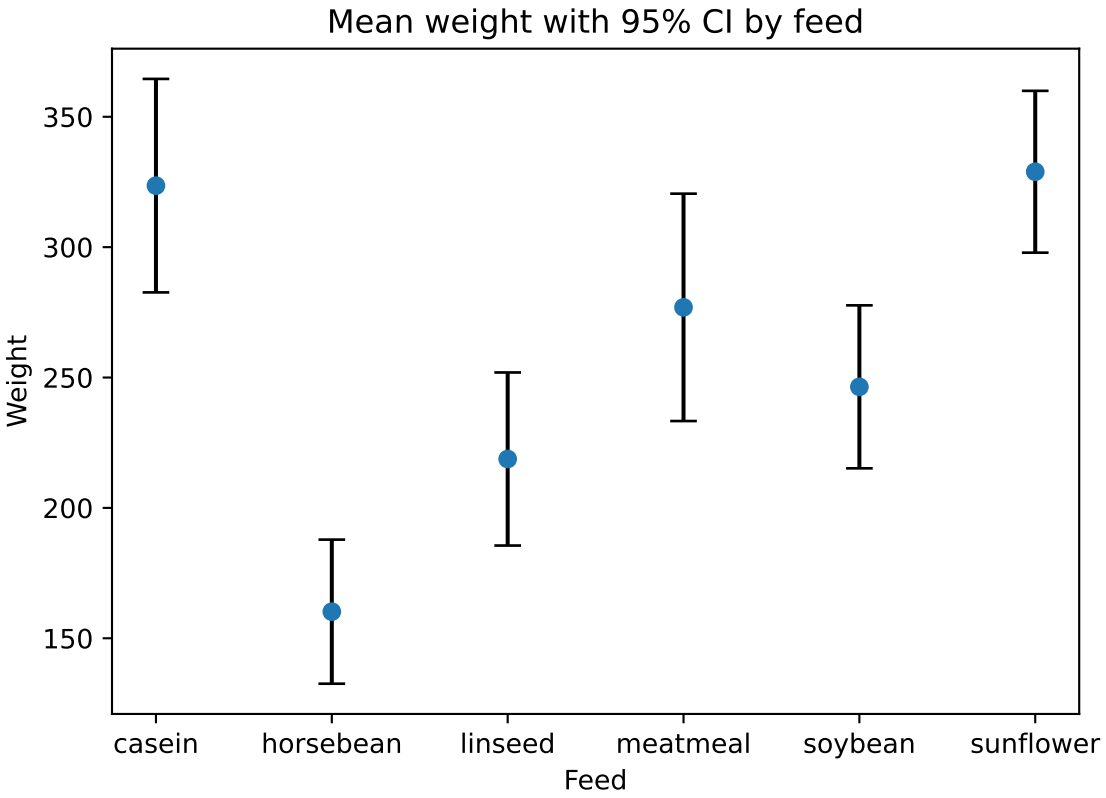
```
chickwts = pd.read_csv("https://asta.math.aau.dk/datasets?file=chickwts.txt", sep = "\t")
cwei = chickwts.groupby("feed")["weight"].agg(
    ['mean',
     'std',
     'count'
])
cwei["se"] = cwei["std"] / (cwei["count"])**(1/2)
cwei["tscore"] = t.ppf(1 - 0.025, df = cwei["count"] - 1)
cwei["lower"] = cwei["mean"] - cwei["tscore"] * cwei["se"]
cwei["upper"] = cwei["mean"] + cwei["tscore"] * cwei["se"]
cwei[["mean", "lower", "upper"]]
```

```
##                  mean       lower       upper
## feed
## casein     323.583333  282.644025  364.522642
## horsebean  160.200000  132.568738  187.831262
## linseed    218.750000  185.561021  251.938979
## meatmeal   276.909091  233.308259  320.509923
## soybean    246.428571  215.175378  277.681765
## sunflower  328.916667  297.887508  359.945825
```

- We can plot the confidence intervals as line segments:

```
import matplotlib.pyplot as plt

g = plt.errorbar(
    x = cwei.index,                # feed categories
    y = cwei["mean"],               # means
    yerr = cwei["mean"] - cwei["lower"],  # half-width of CI
    fmt = 'o',                      # point marker
    ecolor = 'black',               # error bar color
    capsize = 5                     # end cap length
)
plt.xlabel("Feed")
plt.ylabel("Weight")
plt.title("Mean weight with 95% CI by feed")
```



13 Determining sample size

13.1 Sample size for proportion

- The confidence interval is of the form point estimate±estimated margin of error.
- When we estimate a proportion the margin of error is

$$M = z_{crit}\sqrt{\frac{\pi(1-\pi)}{n}},$$

  where the critical z-score, z_{crit}, is determined by the specified confidence level.
- Imagine that we want to plan an experiment, where we **want to achieve a certain margin of error** M (and thus a specific width of the associated confidence interval).
- If we solve the equation above we see:
  - If we choose sample size $n = \pi(1-\pi)(\frac{z_{crit}}{M})^2$, then we obtain an estimate of π with margin of error M.
- If we do not have a good guess for the value of π we can use the worst case value $\pi = 50\%$. The corresponding sample size $n = (\frac{z_{crit}}{2M})^2$ ensures that we obtain an estimate with a margin of error, which is at the *most* M.

13.1.1 Example

- Let us choose $z_{crit} = 1.96$, i.e the confidence level is 95%.
- How many voters should we ask to get a margin of error, which equals 1%?

- Worst case is $\pi = 0.5$, yielding:

$$n = \pi(1-\pi)\left(\frac{z_{crit}}{M}\right)^2 = \frac{1}{4}\left(\frac{1.96}{0.01}\right)^2 = 9604.$$

- If we are interested in the proportion of voters that vote for "socialdemokratiet'' a good guess is $\pi = 0.23$, yielding

$$n = \pi(1-\pi)\left(\frac{z_{crit}}{M}\right)^2 = 0.23(1-0.23)\left(\frac{1.96}{0.01}\right)^2 = 6804.$$

- If we instead are interested in "liberal alliance'' a good guess is $\pi = 0.05$, yielding

$$n = \pi(1-\pi)\left(\frac{z_{crit}}{M}\right)^2 = 0.05(1-0.05)\left(\frac{1.96}{0.01}\right)^2 = 1825.$$

13.2 Sample size for mean

- The confidence interval is of the form point estimate±estimated margin of error.
- When we estimate a mean the margin of error is

$$M = z_{crit}\frac{\sigma}{\sqrt{n}},$$

  where the critical z-score, z_{crit}, is determined by the specified confidence level.
- Imagine that we want to plan an experiment, where we **want to achieve a certain margin of error** M.
- If we solve the equation above we see:
  - If we choose sample size $n = (\frac{z_{crit}\sigma}{M})^2$, then we obtain an estimate with margin of error M.
- Problem: We usually do not know σ. Possible solutions:
  - Based on similar studies conducted previously, we make a qualified guess at σ.
  - Based on a pilot study a value of σ is estimated.