CE888: Data Science and Decision Making
Lecture 2: Summary and resampling statistics

Ana Matran-Fernandez
amatra@essex.ac.uk

Summary statistics and resampling statistics

- Today we are going to learn how to use data to…
  - **estimate** parameters with *confidence*
    - e.g., What's the **average height** for CE888 students?
  - **test theories** about parameters
    - e.g., Are **international** CE888 students **significantly taller** than **home** students?
    - e.g., Do **people who nap perform better** at their job than **people who don't nap**?
- These are some of the ideas behind decision-making

Learning objectives

- Name at least three different summary statistics
- Define a confidence interval
- Calculate confidence intervals for one population parameter
- Communicate statistical ideas clearly and concisely for a potential client
- Know how to formulate a research question
- (Lab) Create confidence intervals in Python
- (Lab) Run hypothesis tests in Python and interpret the output

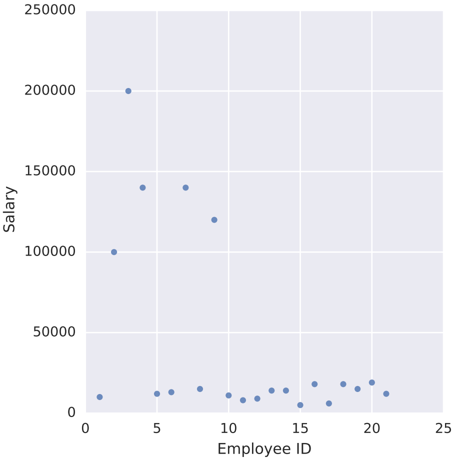
Example: Salaries dataset

	Employee ID	Salary
0	1	10000
1	2	100000
2	3	200000
3	4	140000
4	5	12000
5	6	13000
6	7	140000
7	8	15000
8	9	120000
9	10	11000
10	11	8000
11	12	9000
12	13	14000
13	14	14000

- What's the average salary in this company?
- We only have information about some employees (e.g., through friends and acquaintances)

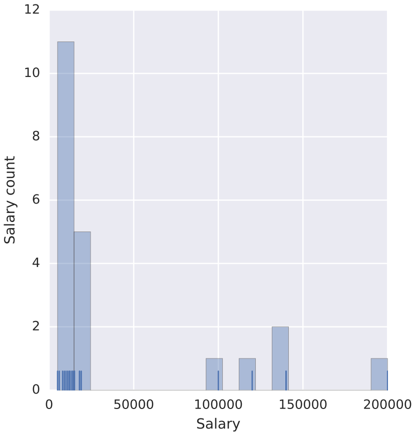
Visualising the data

```python
import pandas as pd
import seaborn as sns

df = pd.read_csv('./salaries.csv')
print(df.columns)
# ['Employee ID', 'Salary']
# Get the values of the second column
# as a NumPy array
data = df['Salary'].values
print(type(data), data)
## An alternative way
data = df.iloc[:, 1].values
print(type(data), data) # NumPy array

sns.lmplot(df.columns[0], df.columns[1],
           data=df, fit_reg=False)
```

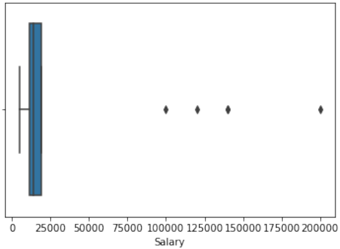


Histogram and boxplot

```
sns.distplot(data, bins=20, kde=False, rug=True)
```




```
sns.boxplot(x='Salary', data=df)
```



Measures of central tendency

- (Sample) Mean
  - $\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i$
- (Sample) Median
  - Rank (i.e., sort) x_i
  - $M = \begin{cases} x_{n/2+1} & \text{if } n \textit{ is odd} \\ (x_{/2} + x_{(n+1)/2})/2 & \text{if } n \textit{ is even} \end{cases}$
- In the salary sample:
  - $\bar{x} = 42809.52$
  - $M = 14000.00$

Measurements of dispersion

- (Sample) Standard deviation
  - $s = \sqrt{\frac{1}{n-1} \sum_{i=1}^{n} (x_i - \bar{x})^2}$
  - Variance is s^2
- In our sample:
  - $s = 56841.15$
  - $s^2 = 3230916099.77$

Note that there are many different summary statistics.

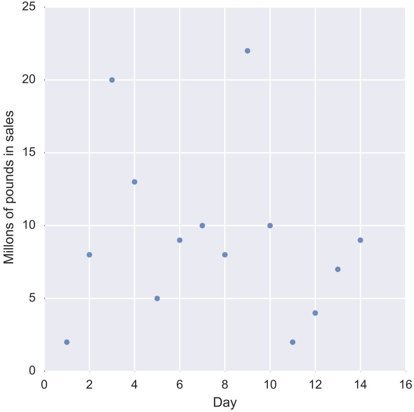
These are just some examples for illustration purposes.

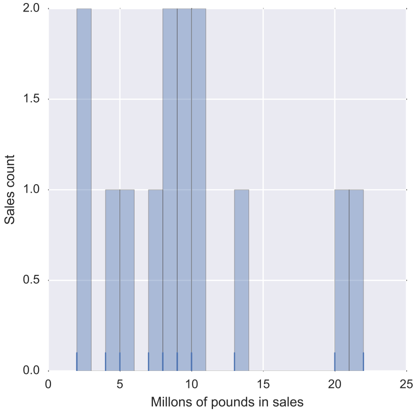
Example: Sales dataset

- A company has recorded their sales for 14 days
- They want to understand their data
- Let's have a look

```
df = pd.read_csv('./customers.csv')
print(df.columns)  # ['Day', 'Millions of pounds in sales']
# Get the values of the second column as a NumPy array
data = df[df.columns[1]].values
sns.lmplot(df.columns[0], df.columns[1], data=df, fit_reg=False)  # Scatterplot
sns.distplot(data, bins=20,  kde=False, rug=True)  # Histogram
```

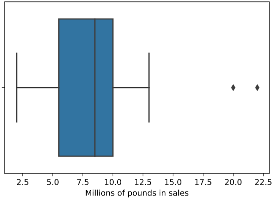
Visualisation of Sales dataset





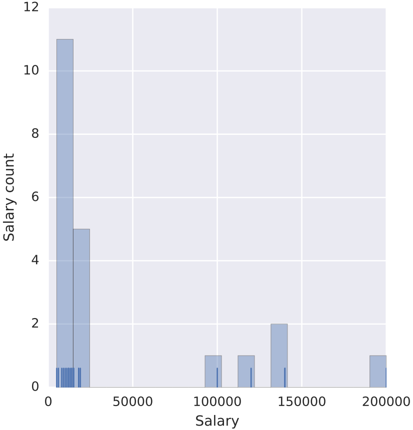
Summary statistics of the sales dataset

```
# Assuming libraries are already imported

df = pd.read_csv('./customers.csv')
print(df.columns)
# ['Day', 'Millions of pounds in sales']
# Get the values of the second column as a NumPy array
data = df[df.columns[1]].values
print(("Mean: %f" % np.mean(data)))  # 9.214286
print(("Median: %f" % np.median(data)))  # 8.500000
print(("Var: %f" % np.var(data)))  # 32.311224
print(("std: %f" % np.std(data)))  # 5.684296

print(df.describe())

#            Day   Millions of pounds in sales
#count   14.0000                     14.000000
#mean     7.5000                      9.214286
#std      4.1833                      5.898873
#min      1.0000                      2.000000
#25%      4.2500                      5.500000
#50%      7.5000                      8.500000
#75%     10.7500                     10.000000
#max     14.0000                     22.000000
```



Back to the Salaries dataset

- Mean: 42,809
- Median: 14,000



Are we confident we got the right mean?

- How confident should the journalist or the analyst be about their summary statistics?
- We would like to build some notion of confidence
  - Get a measure of "If I do this sampling process over and over again, what would I expect to see?"
  - Through Confidence Intervals (CI)
- We are going to take the above statement seriously
  - And introduce the bootstrap!

Confidence interval

- "A range of reasonable values for our parameter"
- Used to give an *interval* estimate for the parameter of interest
- Which we can phrase as:
  - "With 95% confidence, the mean salary in the company is estimated to be between (lower bound) – (upper bound)."
  - "Based on our sample of 21 salaries, we estimate with 95% confidence that the mean salary of the company is between (lower bound) – (upper bound)."
- Note: We don't know if the **real** mean salary in the company falls within this interval. We won't know unless we ask all employees (or HR)
- It is **not** a 95% chance or probability that the real mean salary is in the interval
- It is our confidence in the procedure we used: 95% of the times we run this procedure, the mean will fall in that interval

The bootstrap

- We are going to use a method called the bootstrap to create those confidence intervals
- Very popular, computational method
- You will see this term used quite often in scientific contexts
- Hard to do manually
- DiCiccio, T.J. and Efron, B., 1996. *Bootstrap confidence intervals.* Statistical science, 11(3), pp.189–228.

Bootstrapping (1)

- Ideally, we would sample from the real population
  - i.e., the journalist would go over to a different set of friends
  - and ask them to get more salaries
  - Again and again!
- Once we have a collection of different means (or any other summary we're interested in) we can say that the mean will fall within a certain range with a certain probability
  - But sampling from the real population may be expensive, or infeasible
- However, we can use our sample in a smart way
  - Resample from the sample!

Bootstrapping (2)

- Sample with replacement from the data you have already
  - Create $\{1, ..., B\}$ new samples (i.e., bootstraps) of the same size
  - Let's assume each observation in the initial dataset is x_i, where i is the order in which it appeared

$$x = \{x_1, x_2, x_3, x_4, x_5, ..., x_N\}$$

- Now we create B samples from x:

$$x^1 = \{x_4, x_5, x_3, x_5, x_{N-1}, ...\}$$

$$x^2 = \{x_3, x_7, x_7, x_8, ...\}$$

$$\vdots$$

$$x^B = \{x_8, x_3, x_2, x_4, ...\}$$

Bootstrapping (3)

- Let's do one example
- x = {0, 1, 4, 2, 3}
- Let's draw three bootstraps of size 5

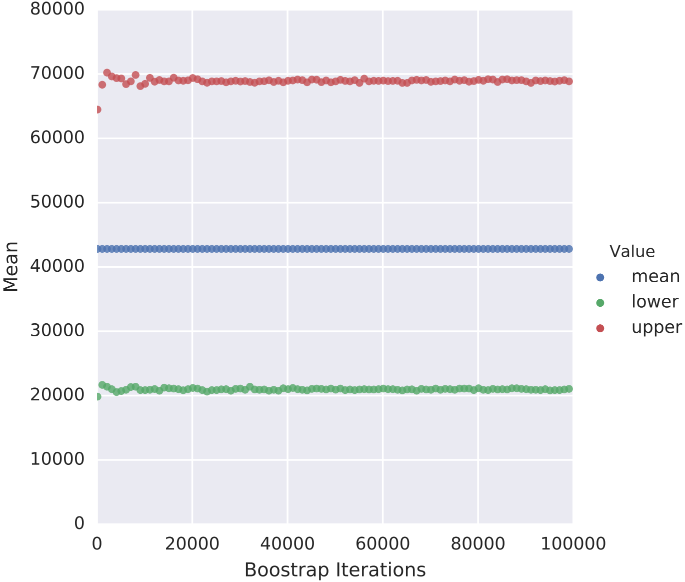
Bootstrapping (4)

- Get the mean for each sample (since we are interested in the mean — but it could be any other measure!)
- We can now sort the means
- We remove the bottom 10% and the top 10% to find $\gamma = 0.80$
- e.g., for the sales data:

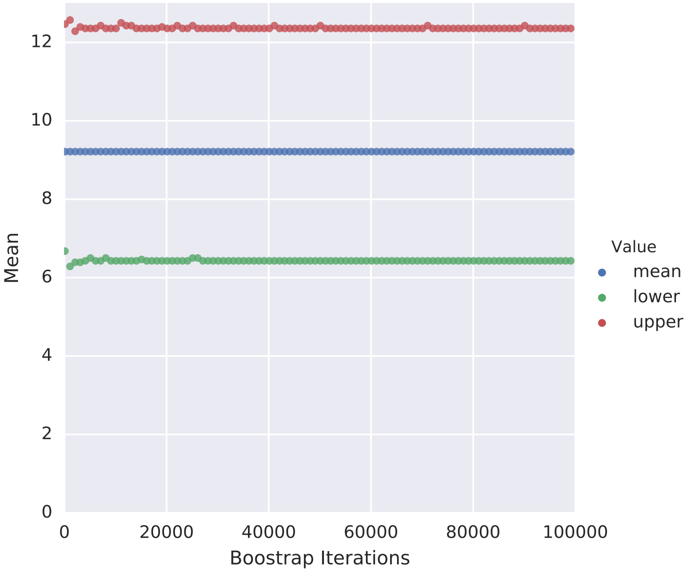
$$x = [6.86, 7.29, 7.86, 8.14, 8.36,$$
$$8.79, 8.86, 9.14, 9.29, 9.5, 9.5,$$
$$9.71, 10.36, 11.14, 11.14, 13.21]$$

- What about if I was interested in $\gamma = 0.90$?
- What about if I was interested in $\gamma = 0.95$?

Salaries


80000
70000
60000
50000
40000
30000
20000
10000
0
Mean
0
20000
40000
60000
80000
100000
Boostrap Iterations
Value
mean
lower
upper

Sales


12
10
8
6
4
2
0
Mean
0
20000
40000
60000
80000
100000
Boostrap Iterations
Value
mean
lower
upper

What can we say about the means now?

- Salaries mean is…
- Sales mean is…
- We can do bootstrap to estimate *any* quantity we want, as long as the distribution has a defined variance and mean
  - i.e., not always
- But for most practical matters, yes

(An Aside) Data bias

- I have described a very biased process of collecting samples
  - The journalist asked her friends
  - All her friends love football
  - What she might actually have learned is the salary of football–loving employees
- How about the sales dataset?
  - Was there anything extraordinary on the days these measurements were taken?
  - Maybe it was Christmas
- Be very careful to randomise properly or at least make sure that you state your bias
  - Think about where your dataset is coming from
  - Try to state it when you answer the question

A/B Testing

- Suppose you had two versions of a website
  - and you would like to check if the newer version is better
- Two versions of an e-mail
  - and you would like to check if the newer, fancier version is better
- A new vaccine
  - and you would like to see if it actually works
- Two groups of academics
  - and you want to know if one of them is more appreciated by students

Hypothesis testing

- Same as A/B testing
- The name people used to call A/B testing when testing for
  - Drug effects
  - Physical effects
  - Quality management

Example problem

- A company sends out e-mails
  - Various promotions and news content
  - They want users to click on the links and get on their website
  - They already have an e-mail format
  - Mark from marketing comes up with a new format with improved content
- Is it better?
  - Without causing too much disruption

Hypothesis testing

- They send 11 e-mails of the usual type (control)
- They also send 11 e-mails of the new design (test)

```
old = np.array([0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0])
new = np.array([1, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0])
```

$\bar{x}_{old} = 0.18$

$\bar{x}_{new} = 0.45$

$t_{obs} = \bar{x}_{new} - \bar{x}_{old} = 0.27$ (observed value of the test statistic)

Should they change to the new design?

Hypothesis forming

H_0: The two e-mails have no difference (their means are equal) - this is called the *null* hypothesis

H_1: The second e-mail is better, and thus has a higher mean - alternative hypothesis (also H_a)

- Set significance level $\alpha = 0.05$, or equivalently, check if the 95% CI of t_{obs} under H_0 does not contain the real t_{obs}
- p value = What is the probability of observing something as extreme as what we just observed by pure chance?

Permutation testing (1)

- Merge all the data into a new array

```
concat = np.concatenate((old, new))
```

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

- Permutate it randomly, i.e. form a new array from the same elements

```
perm = np.random.permutation(concat)
```

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

Permutation testing (2)

- Split again into new and old (first half and second half)

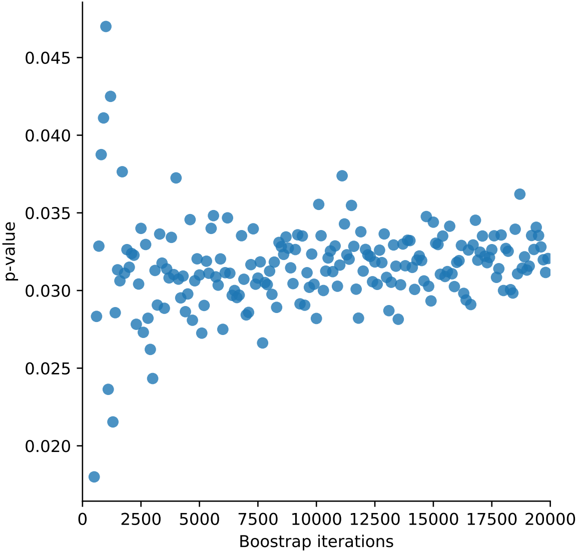
```
pold = perm[:int(len(perm)/2)]
# np.array([0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0])
pnew = perm[int(len(perm)/2):]
# np.array([1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1])
```

- Record whether the value of the test was more extreme than the observed
  - $t_{perm} = \bar{x}_{pnew} - \bar{x}_{pold}$
  - $t_{perm} > t_{obs}$
- Keep on permutating and recording
- Find the number of times $t_{perm} > t_{obs}$
- Divide by the number of permutations you did
- You call that number your *p-value*

Permutation tests (3)

- If you repeat this process 20,000 times you get $p = 0.034$
- Hence we can conclude that 3% of the time you will get a higher difference in means than t_{obs}
- Since this number is smaller than our 5% significance level, we can reject the *null* hypothesis H_0
- So we conclude that the new format is better

Permutation test (4)



Another experiment

- Bob decides that adding a sound to the e-mail should increase user clicking even more
- Thinking that his solution is better for sure, he sends more e-mails with sounds (i.e. the new version)
  - Not exactly A/B testing, but he seems eager…
- Results come back and he has to somehow show that his new e-mail procedure is better

Some data analysis

```
old = np.array([0, 1, 1, 1, 0, 1, 1, 0, 0, 1, 0])
new = np.array([0, 1, 1, 0, 1, 1, 0, 1, 1, 1, 0,
                0, 1, 1, 1, 1, 1, 1, 1])
```

$\bar{x}_{old} = 0.55$

$\bar{x}_{new} = 0.74$

$t_{obs} = \bar{x}_{new} - \bar{x}_{old} = 0.19$

Results

- With 20,000 permutations we get $p = 0.07$
- $p > 0.05$: Thus, we have failed to reject the null hypothesis
- This does **not** mean that the sound does not have any impact
- Just that we can't tell the impact

Gender bias experiment (1)

- This test works with any sort of numbers!
- At the end of term, students are asked to rate every academic that has taught them.
- The School wants to know (as part of the Athena SWAN programme) if there's a difference between male and female academics.
- Hypothesis forming:
  - H_0: Female and male academics are rated similarly (i.e., no difference in means)
  - H_1: Male academics are rated higher than female academics (i.e., there is a difference in the means of the two groups)
- $\alpha = 0.05$
- p value = What is the probability of observing something as extreme as what we just observed by pure chance?

Gender bias experiment (2)

- We have one rating for each academic. We separate them based on gender.

```
female = np.array([random.uniform(1, 3.5) for _ in range(10)])  # 10 females
male = np.array([random.uniform(2, 4) for _ in range(40)])  # 40 males
print(female.mean(), male.mean(), male.mean() - female.mean())
```

$\bar{x}_{female} = 2.63$

$\bar{x}_{male} = 3.04$

$t_{obs} = \bar{x}_{male} - \bar{x}_{female} = 0.41$ (observed value of the test statistic)

Are men rated higher by students?

Gender bias experiment (3)

```
print("Gender bias p value =", permutation_resampling(20000, male, female))
```

Gender bias p value = 0.0243

A way to phrase this outcome could be: "Based on our sample of SAMT scores from 2019/20, we found statistically significant differences at a 5% significance level in the average scores given to male and female academics, with male academics being rated higher ($p = 0.02$)."

We can also estimate confidence intervals for the means of both groups and see if/how they overlap:

- Bootstrap with 5000 iterations:
  - FEMALES: Mean = 2.52; Bootstrap mean = 2.52 (2.07, 2.92)
  - MALES: Mean = 3.06; Bootstrap mean = 3.06 (2.89, 3.22)

Errors

- Type I error: rejecting H_0 even though it is true (a.k.a. false alarm)
- Type II error: failing to reject H_0 even though it is false (there's fire, but the alarm didn't sound)

	H_0 is true	H_0 is false
Reject H_0	Type I error (false positive)	Correct inference
Fail to reject H_0	Correct inference	Type II error (false negative)

Concluding remarks

- Hypothesis testing is used quite extensively
- And abused more often
- Real life problems (usually) have more data and are more noisy than today's examples
    - But you can send e-mails, get clicks, etc. trivially
- If there is one thing to keep from this lecture is **the use of bootstrapping to learn parameter confidence intervals**
    - We will use the bootstrap later on this module in the Modelling lecture
    - Especifically, to estimate the bias in our models