Statistical methods for Data Science
Introduction

UNIVERSITY OF
GOTHENBURG

CHALMERS

Richard Johansson

November 5, 2018

today

- overview of the course and practicalities
- analysing numerical data with Python
- random numbers in Python, and basic simulation

why statistics in data science? exploring and modeling

- what is the intensity of incoming HTTP requests to a web server in the day and in the night?
- does the use of clickers have an effect on student satisfaction? on the probability of passing the exam?
- do speakers affected by Alzheimer's disease exhibit a significantly smaller vocabulary?

why statistics in data science? in the models we use

- what is the probability of observing the word *lottery* in a spam email? in a normal email?
- if we assume that our data was generated by a *Gaussian Mixture Model*, how do we find the parameters of the distributions?

why statistics in data science? experimental evaluations

- a search engine S_1 is tested on a sample of 1000 queries and gets a Mean Average Precision score of 0.76. How precise is this measurement?
- another search engine S_2 is tested on the same sample and gets a MAP score of 0.82. Is the second system significantly better than the first one?

is "data science" a rebranding of statistics?

- examples by David Donoho
  - *Aren't* **we** *Data Science?*
    Column of ASA President Marie Davidian in AmStat News, July, 2013[4]
  - *A grand debate: is data science just a 'rebranding' of statistics?*
    Martin Goodson, co-organizer of the Royal Statistical Society meeting May 11, 2015 on the relation of Statistics and Data Science, in internet postings promoting that event.
  - *Let* **us** *own Data Science.*
    IMS Presidential address of Bin Yu, reprinted in IMS bulletin October 2014[5]
- see his paper 50 years of Data Science
- see also the comments by Andrew Gelman and Yann LeCun

difference between "statistics" and "machine learning"?

- "glossary" by Rob Tibshirani

machine learning	**statistics**
network, graphs	model
weights	parameters
learning	fitting
generalization	test set performance
supervised learning	regression, classification
unsupervised learning	density estimation, clustering
large grant = \$1,000,000	large grant = \$50,000
nice place to have a meeting: Snowbird, Utah; French Alps	nice place to have a meeting: Las Vegas in August

the contents of this course

- practical **data analysis**:
  - summary statistics, plotting, practical libraries
- methodology for **experiments**:
  - good and bad practices
  - margins of error, "significance"
  - a small selection of statistical tests
- statistical **models** that are common in data science:
  - basic distributions
  - regression models (linear and more)
  - Naive Bayes classifiers
  - Gaussian mixture models for clustering
  - Hidden Markov models for sequences
  - topic models for text analysis

overview

practical matters about the course

basic data exploration and Python libraries

random numbers and simulation

Canvas website

`https://gu.instructure.com/courses/11025`

- you should be able to see the site without logging in
- but if you can't log in, please let me know

teachers

Richard – course responsible, teaches first half of the course

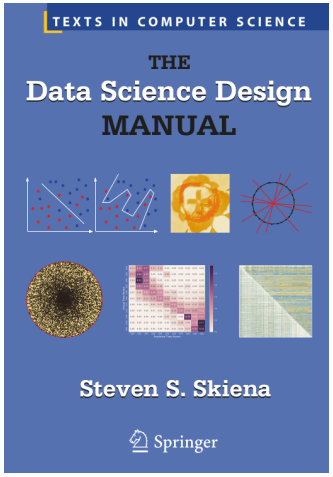
Morteza – teaches second half of the course

Jin – teaching assistant

Fionn – teaching assistant

literature

- we'll reuse some of the literature used in the course *Introduction to Data Science*



- plus links to online resources, will be posted during the course

examination

- **programming assignments** done in groups of 2 or 3
  - please read what the Canvas site says about the assignments
- individual **take-home exam** in January
- I will send out a doodle to find the best time for the take-home exam
- the grades: fail, pass, pass with distinction

Richard's office hours

- I will reserve 2 hours each week when you can come to ask questions about the lectures or assignment, or any discussion about the course
- my office is room 6115 in this building
- this week: Friday, 13-15
- please check Canvas site for the rest of the period

course representatives

- please send me an email if you're interested!

overview

practical matters about the course

basic data exploration and Python libraries

random numbers and simulation

Python libraries we'll use in this course

- **SciPy**: a Python library for statistics and math in general
- **NumPy**: efficient mathematical functions
- **Pandas**: practical data processing

- **Matplotlib**: drawing diagrams
- **Seaborn**: additional statistical plotting functions

- **StatsModels**: some more advanced statistical models
- **scikit-learn**: predictive models and clustering

- if you have a standard Anaconda installation, they are already installed

side note: Bokeh, library for interactive visualization

`https://bokeh.pydata.org/`

side note: plot gallery for Python

https://python-graph-gallery.com/

typical start of a Python file for statistical tasks

- note that there are some standard shorthand imports

```
import pandas as pd
import scipy
import scipy.stats as stats
import numpy as np

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

an example dataset

```
182 86 m
193 112 m
172 72 m
170 61 f
167 58 f
160 66 f
152 53 f
162 54 f
167 79 f
165 63 m
177 67 m
165 66 f
167 62 m
165 61 f
175 87 m
...
```

```
import pandas as pd

# read the data
data = pd.read_csv('bodies.txt',
                   sep=' ',
                   header=None,
                   names=['height', 'weight', 'gender'])

data.head()
```

	height	weight	gender
0	182	86	m
1	193	112	m
2	172	72	m
3	170	61	f
4	167	58	f

summarizing the data

- **summary statistics**:
  - centrality: mean, median, min, max, quantiles
  - dispersion: variance / standard deviation
  - relations: covariance, correlation
- **plots**:
  - distributions: histogram, distribution plot, box plot
  - relations: scatterplot

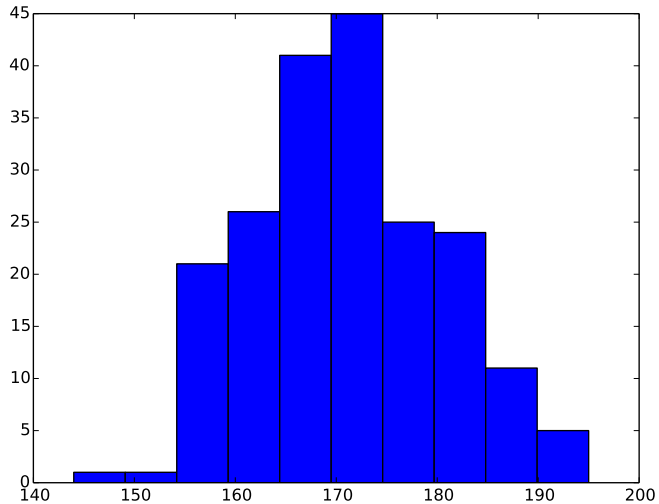
summary statistics (Pandas)

```
data.describe()
```

	height	weight
count	22.000000	22.000000
mean	169.590909	69.636364
std	10.140250	15.941994
min	152.000000	47.000000
25%	165.000000	58.750000
50%	167.000000	66.000000
75%	177.000000	82.000000
max	193.000000	112.000000

plotting histograms

- a **histogram** is a diagram that shows how the data points are distributed
- the x axis shows "bins", e.g. 165–170 cm, and the y axis shows the number of data points in that bin



some basic data analysis

- **maximal** and **minimal** values
- **sample mean** (average) and **median**

measures of dispersion: variance and standard deviation

- recall that the mean $\bar{x}$ of a dataset x is defined

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

- the **sample variance** $V(x)$ of a dataset x measures how much x is concentrated to the mean
  - it is the mean of the squares of the offsets from the mean

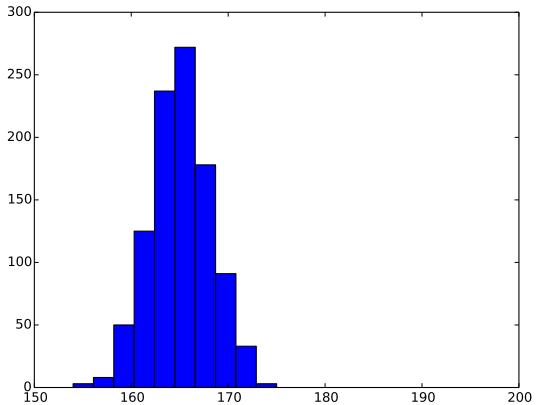
$$V(x) = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

- the **sample standard deviation** $\sigma(x)$ is the square root of the variance

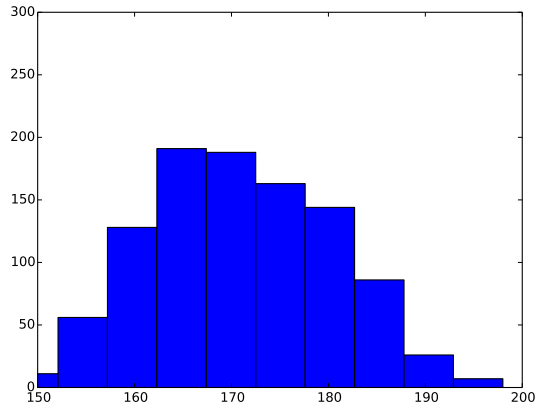
$$\sigma(x) = \sqrt{V(x)}$$

example

- low variance: data concentrated near the mean
  - in the extreme case: all values are identical
- high variance: data spread out



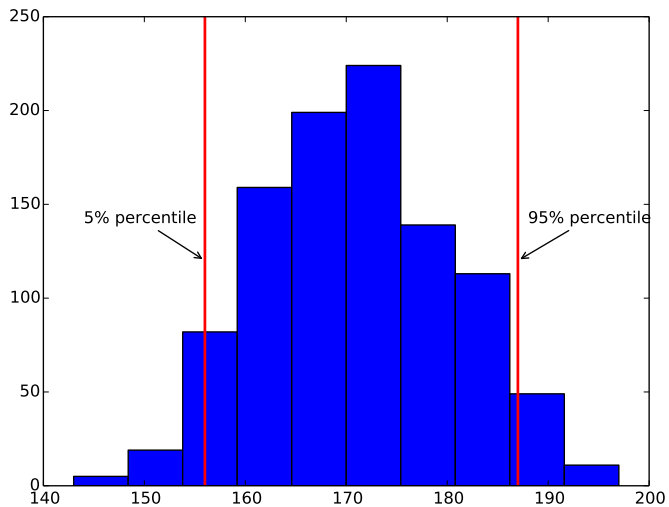
$\sigma = 2.9$



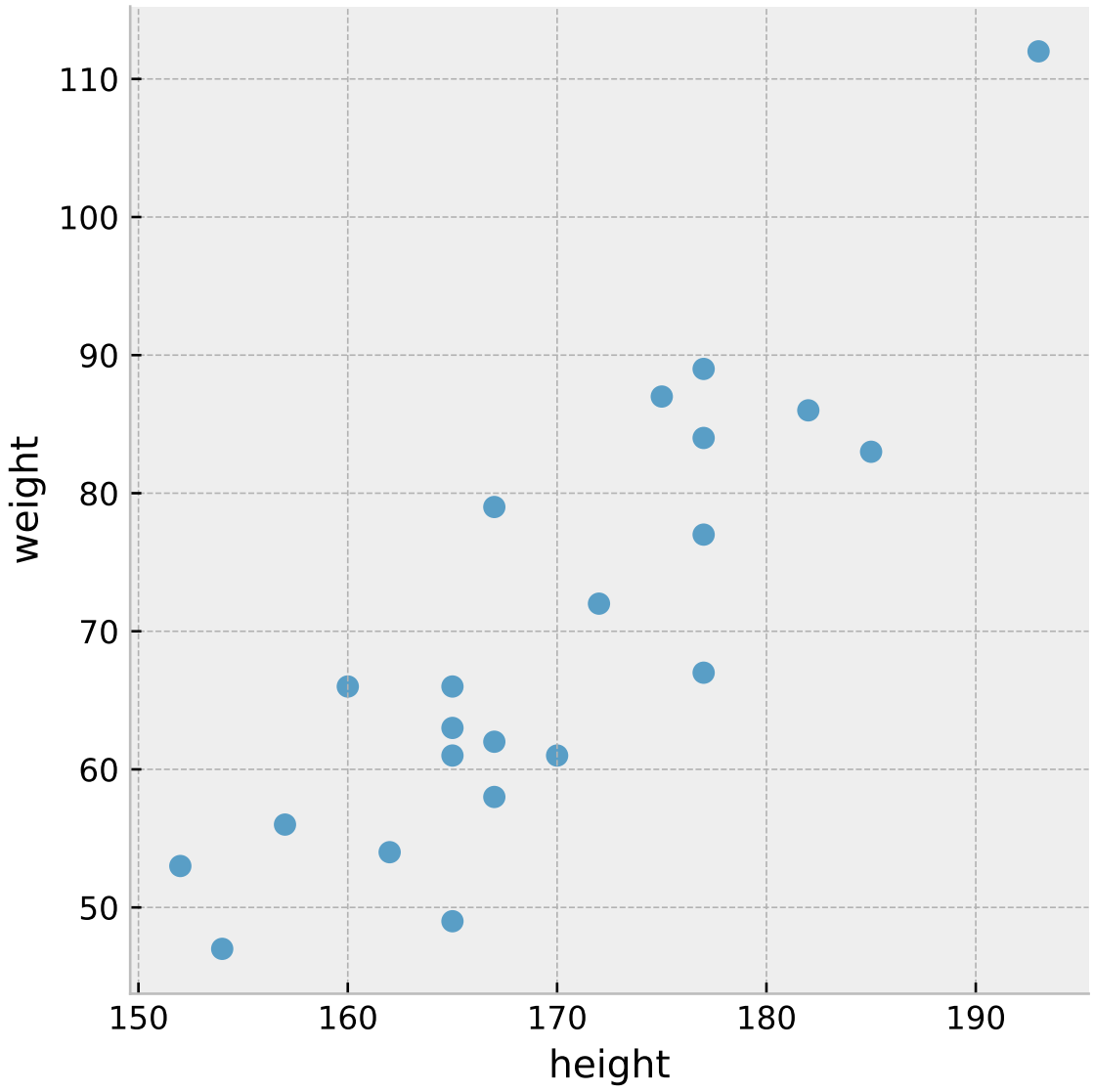
$\sigma = 9.1$

percentiles / quantiles

- how tall are the shortest 5% of the people in the dataset?
  - formally: what is the x such that 5% of the data is less than x?
- this number is called the **5% percentile** (or the 0.05 **quantile**)



relations between two variables: scatterplot



relations between two variables: correlation

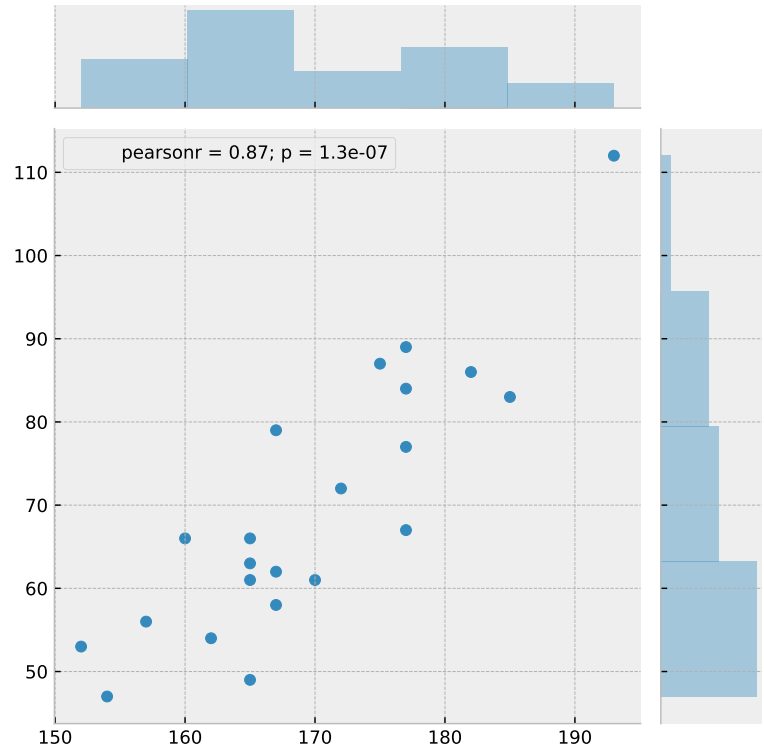
- the **correlation coefficient** or the **Pearson** r measures how close the data is to a linear relationship
- it is a number that ranges between -1 and +1
  - example [Wikipedia]:

- it is defined

$$r(x,y) = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sigma(x)\sigma(y)}$$

correlation example

- for the height–weight data, $r = 0.87$



examples: basic descriptive statistics (NumPy and SciPy)

(assuming we have loaded the data into a lists or arrays called `heights` and `weights`)

- mean, min, max:

  ```
  np.mean(heights); np.min(heights); np.max(heights)
  ```

- median, quantiles/percentiles:

  ```
  np.median(heights); np.percentile(heights, 5)
  ```

- variance and standard deviation:

  ```
  np.var(heights); np.std(heights)
  ```

- correlation:

  ```
  scipy.stats.pearsonr(heights, weights)
  ```

examples: basic descriptive statistics (Pandas)

(assuming we have loaded the data into a `DataFrame` called `data`)

- overall summary:

  ```
  data.describe()
  ```

- mean, min, max:

  ```
  data['heights'].mean(); data['heights'].min(); data['heights'].max()
  ```

- median, quantiles/percentiles:

  ```
  data['heights'].median(); data['heights'].quantile(0.05)
  ```

- variance and standard deviation:

  ```
  data['heights'].var(); data['heights'].std()
  ```

- correlation:

  ```
  data['heights'].corr(data['weights'])
  ```

see the Pandas reference documentation

examples: basic plotting (matplotlib)

- histogram:

```
plt.hist(data['height'], bins=10)
```

- scatterplot:

```
plt.scatter(data['height'], data['weight'])
```

examples: basic plotting (Pandas)

- histogram:

```
data['height'].plot(kind='hist')
```

- scatterplot:

```
data.plot.scatter('height', 'weight');
```

examples: basic plotting (Seaborn)

```
import seaborn as sns
```

- histogram (plus KDE, optionally):

  ```
  sns.distplot(data['height'])
  ```

- scatterplot, plus regression line:

  ```
  sns.lmplot(x='height', y='weight', data=data)
  ```

- scatterplot, no regression line, separate by gender:

  ```
  sns.lmplot(x='height', y='weight', data=data,
             hue='gender', fit_reg=False)
  ```

- two-dimensional density plot using KDE:

  ```
  sns.kdeplot(data['height'], data['weight'])
  ```

overview

practical matters about the course

basic data exploration and Python libraries

random numbers and simulation

why random numbers?

- in this course, mainly for didactic purposes
  - simulating models
  - generating synthetic data
- also used in some statistical methods and models
  - bootstrapping
  - Bayesian inference
  - stochastic gradient descent

a brief note on "random" numbers

- **pseudorandom numbers**: generating "random" numbers in a computer using a deterministic process
  - usually fast
  - we use a starting point called the **seed**
  - if we use the same seed, we'll get the same sequence
    - good for replicable experiments
  - might be a security risk in some situations
- hardware random numbers
  - sample noise from hardware devices
  - Linux: `/dev/random`

basic functions for random numbers: `np.random`

- reset the random number generator:

```
np.random.seed(0)
```

- generate a random floating-point number between 0 and 1:

```
np.random.random()
```

- generate a random integer between 1 and 6

```
die_roll = np.random.randint(1, 7)     # NB 6+1
```

- shuffle the items of a list or array `x`

```
np.random.shuffle(x)
```

- pick a random item from a list or array `x`

```
selection = np.random.choice(x)
```

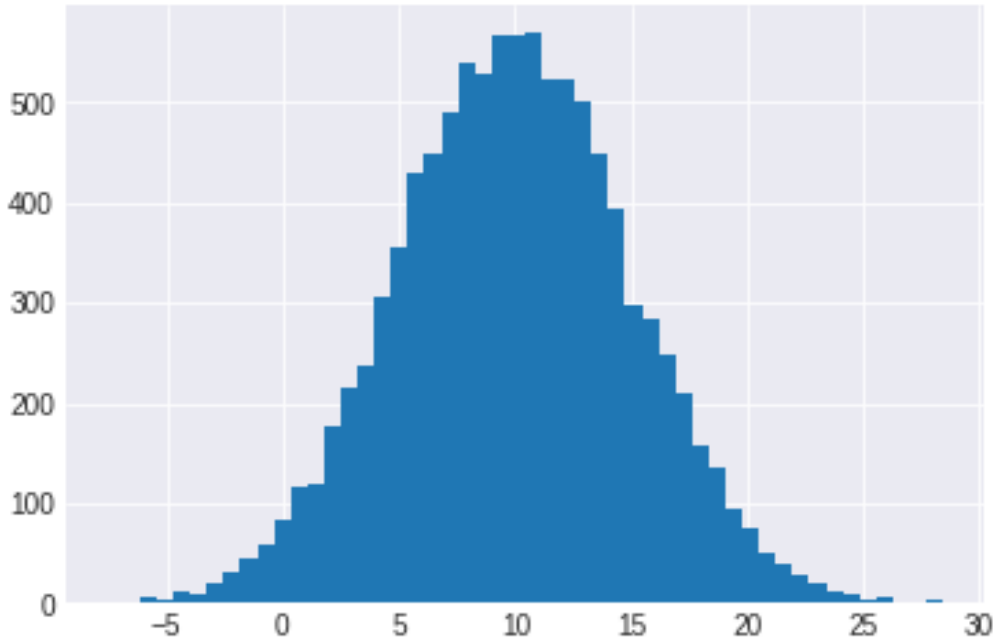
- and much more in the documentation

a note on the random number generators

- the two random number generating functions are examples of **random variables** with **uniform distributions**
  - this means that all outcomes are equally probable
  - if we generate a lot of random numbers, the histogram will be flat
- `np.random.randint(1, 7)` is a **discrete** uniform random variable
  - it generates 1, 2, 3, 4, 5, or 6 with equal probability $\frac{1}{6}$
- `np.random.random()` is a **continuous** uniform random variable
  - it generates any float between 0 and 1 with equal probability
- we'll come back to the notion of random variables and their distributions in later lectures

normally distributed random numbers

```
gauss_numbers = np.random.normal(loc=10, scale=5, size=10000)
plt.hist(gauss_numbers, bins=50);
```



- and there are many other distributions in NumPy, see the documentation

simulation case study: student takes an exam

- the model:
  - there are 5 questions
  - each question awards up to 6 points: we model this as a uniform random variable
  - the scores are **independent**
  - we're interested in the total score
- is this model realistic?

simulation case study: generating product reviews

- the model:
  - there are two categories of reviews: book and camera
  - the document length is a uniform random number between 1 and 100
  - each review category has its own distribution over words
  - we draw the words independently of each other
- this is an example of a **hierarchical** model: we'll talk more about this type of model later in the course
  - in fact, this is a **Naive Bayes** model

the rest of this week

- Wednesday, lab session: first assignment
- Thursday, lecture: introduction to probabilistic models, Naive Bayes classification