

EEN1083/EEN1085 Data Analysis and Machine Learning I

Release Date = 14-10-2024

Due Date: 29-11-2024 (5pm)

Total Marks: 25

This assignment is designed for students to demonstrate their ability to perform data analysis and machine learning with an aim to gain hands-on experience.

There are three datasets provided for this assignment.

Dataset 1: Energy (Gas and Electricity) Consumption Dataset

The Energy Consumption Dataset contains information related to the number of units of (electricity or gas) consumed at different time intervals throughout the day. The dataset consists of a full 12 months period in 2023 and 2024. The dataset can be accessed at: [Shared Google Drive Folder](#)

Dataset 2: Dublin Air Quality Dataset

This data was collected by Google and Dublin City Council as part of Project Air View Dublin. Google's first electric Street View car equipped with Aclima's mobile air sensing platform drove through the roads of Dublin City measuring street by street air quality. Driving predominantly took place Monday–Friday between 9:00 am and 5:00 pm from May 2021 through August 2022, so the dataset primarily represents typical daytime, weekday air quality. The car measured pollution on each street and highway at 1-second intervals, driving with the flow of traffic at normal speeds. The pollutants determined are: Carbon Monoxide(CO), Carbon Dioxide(CO₂), Nitrogen Dioxide (NO₂), NO (nitric oxide), Ozone (O₃), and Particulate Matter PM2.5 (including size resolved particle counts from 0.3 - 2.5 μm). Further details of the dataset can be found at: <https://data.gov.ie/dataset/google-airview-data-dublin-city>

Dataset 3: Mars surface image (Curiosity Rover) Dataset

This is an image dataset containing 6691 images of the Mars surface collected by the Mars Science Laboratory (MSL, Curiosity). The dataset has labels and spans 24 classes. The provided dataset has low resolution images of roughly 256 X 256 pixels each. There is a high resolution version of the dataset available, however for this assignment you should only use this low resolution image dataset. You can get further details of the dataset at their website: <https://zenodo.org/records/1049137>

<https://data.nasa.gov/Space-Science/Mars-surface-image-Curiosity-rover-labeled-data-set/cjex-ucks>

Assignment Tasks:

Students enrolled in the 5 credit version of this module are required to attempt any 2 of the following 3 questions, students enrolled in the 7.5 credit module are required to attempt all 3 questions.

Q1: Data Analysis [50 Marks]

This question relates to Dataset 1. Using Dataset 1, you are required to;

- 1.1. Briefly describe the dataset. (Report) **[5 Marks]**
- 1.2. Formulate a real-world problem that could be solved by applying data analysis over the given dataset. (Report) **[5 Marks]**
- 1.3. Apply different data pre-processing techniques e.g. data cleansing, data reshaping, and data preparation. (Report + Code) **[10 Marks]**
- 1.4. Perform exploratory data analysis by generating relevant summary statistics. (Report + Code) **[10 Marks]**
- 1.5. Use a set of visualization techniques to convey the outcomes of the data analysis and convey a message for the general audience in order to address your formulated problem (Task 1.2).
(Report + Code) **[20 Marks]**

Q2: Machine Learning on Sensor Dataset**[50 Marks]**

This question relates to Dataset 2. Using Dataset 2, you are required to;

2.1. Briefly describe the dataset. (Report) **[5 Marks]**

2.2. Formulate two different real-world problems (Regression and Classification) that could be solved by applying data analysis and machine learning over the given dataset. (Report) **[5 Marks]**

2.3. Apply different data pre-processing techniques e.g. data cleansing, reshaping, resize and preparation. (Report + Code) **[5 Marks]**

2.4. Divide the dataset into training, testing and validation datasets and explain your approach. (Report + Code) **[5 Marks]**

2.5. Perform exploratory data analysis by generating relevant summary statistics and visualization. (Report + Code) **[5 Marks]**

2.6. Train a machine learning model over the given dataset to solve a regression based problem formulated in Task 2.2.

(Report + Code) **[10 Marks]**

2.7. Train a machine learning model over the given dataset to solve a classification based problem formulated in Task 2.2.

(Report + Code) **[10 Marks]**

2.8. Evaluate performance of your trained models using different error detection or accuracy measurement techniques.

(Report + Code) **[5 Marks]**

Q3: Machine Learning on Images Dataset**[50 Marks]**

This question relates to Dataset 3. Using Dataset 3, you are required to;

- 3.1. Briefly describe the dataset. (Report) **[5 Marks]**
- 3.2. Formulate two different real-world problems (Clustering and Classification) that could be solved by applying data analysis and machine learning over the given dataset. (Report) **[5 Marks]**
- 3.3. Apply different data pre-processing techniques e.g. data cleansing, reshaping, resize and preparation. (Report + Code) **[5 Marks]**
- 3.4. Divide the dataset into training, testing and validation datasets and explain your approach. (Report + Code) **[5 Marks]**
- 3.5. Perform exploratory data analysis by generating relevant summary statistics and visualization. (Report + Code) **[5 Marks]**
- 3.6. Train a machine learning model over the given dataset to solve a clustering problem formulated in Task 3.2. (Report + Code) **[10 Marks]**
- 3.7. Train a machine learning model over the given dataset to solve a classification based problem formulated in Task 3.2.
(Report + Code) **[10 Marks]**
- 3.8. Evaluate performance of your trained models using different error detection or accuracy measurement techniques.
(Report + Code) **[5 Marks]**

Submission Guidelines

The final submission should be submitted on the loop before the due deadline. A final submission should contain;

1. A report (PDF or Word document) documenting all assumptions, design decisions, and findings. Include visualizations, plots, and tables. You should strive to make your work completely reproducible using only the report document: include details on everything you tested and all results. Document and justify all design decisions.
2. Two or more separate notebooks (at least one for each question) containing all of your relevant code. In case you have tested multiple versions of the datasets (e.g. after cleansing) or trained different models for the same task, do not overwrite the previous code. You can either submit multiple versions of the notebook or separate code sections for each iteration/repetition of any task.
3. Any additional resources (e.g. datasets or installation, preparation or execution guidelines) bundled as archive file(s) e.g. zip. If your zip folder size is greater than 250MB, you have to break it down into multiple zip files ensuring each file is less than 250MB so it can be uploaded on the loop.

Important: Do **NOT** include code as images (e.g. screenshots of code) in your report. Include code snippets as text.

Plagiarism

Please read and strictly adhere to the DCU Academic Integrity and Plagiarism Policy. Note that reports are automatically checked against each other and against external web sources for plagiarism. Any suspected plagiarism will be treated seriously and may result in penalties and a zero grade (see Sec 6.2 of the DCU Academic Integrity and Plagiarism Policy). You are not allowed to copy any code (in full or parts) from jupyter notebooks provided by the dataset owner or third party for their own analysis. In case of suspected code plagiarism, you will be asked for an interview to demonstrate your programming skills and ability to code independently.

Grading

The assignment is worth 25% of the overall mark for the module. Marks will be awarded based on the quality of the resulting report. In particular, I will be checking to see if you are handling data correctly, carrying out exploratory analysis to gain insights, correctly performing model selection, and critically, documenting everything in a clear and concise way. The submitted code will also be checked to ensure that the work is your own.

Late Submission:

Please note that any last minute requests to extend assignment submission deadlines will not be entertained. Late submission will incur a **penalty of 30%** on the awarded grades. Loop will remain open for late submission until **02-12-2024 (5pm)**. No submissions will be accepted after the late submission deadline.

EEN1083

Data Analysis and Machine Learning I

Ali Intizar

Semester 1
2024/2025



Ollscoil Chathair
Bhaile Átha Cliath
Dublin City University



Mathematics for Machine Learning

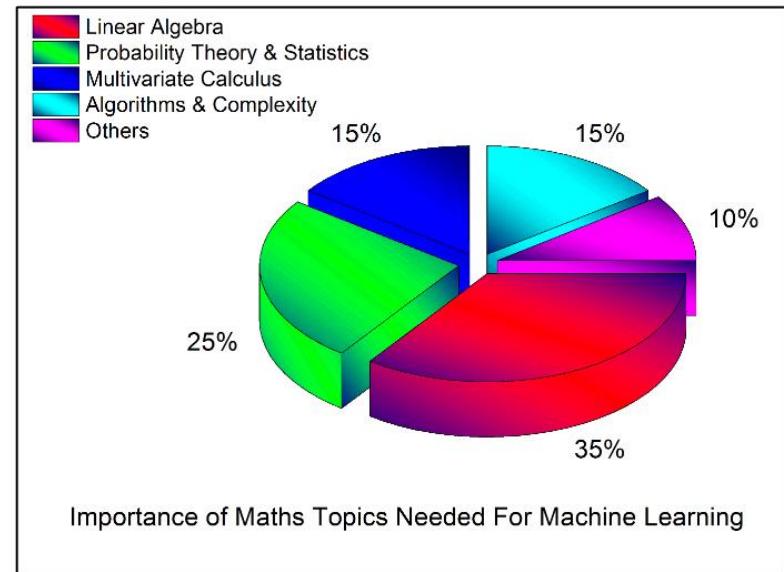


Ollscoil Chathair
Bhaile Átha Cliath
Dublin City University



Why we need mathematics for machine learning?

- Mathematics is one of the core foundational skills
- Algorithms Selection
- Parameters Selection
- Fine Tuning Trained Algorithms
- Optimisation
- Training and Testing Algorithms



What are core concepts I need to be aware of?

- Linear Algebra

- Vectors and Matrixes
- Transpose and Inverse
- Dot products
- Eigenvalues
- Euclidean distance

- Calculus

- Multi-variables
- Limit, continuity, derivatives
- Differential and Integral

- Statistics and Probability

- Sampling
- Regression
- Conditional probability

Sets

A set is a well-defined collection of **distinct** objects (possibly infinite or uncountable).

Some examples:

- ▶ $\{1, 2, 3\}$, $\{a, e, i, o, u\}$, $\{\pi, e\}$
- ▶ Integers $\mathbb{Z} = \{\dots, -3, -2, -1, 0, 1, 2, 3, \dots\}$
- ▶ Positive integers $\mathbb{Z}_{++} = \{1, 2, 3, \dots\}$
- ▶ Real numbers \mathbb{R}

If x is an element of set Z we write $x \in Z$. E.g. $x \in \mathbb{R}$ means x is a real number.

Set builder notation:

- ▶ Positive reals $\mathbb{R}_{++} = \{x \in \mathbb{R} : x > 0\}$

Sets: empty set, cardinality, intersection, union

The empty set is the set with no elements $\emptyset = \{\}$.

The cardinality of a set is the number of elements in the set. E.g. $X = \{1, 2, 3\}$, $|X| = \#X = 3$.

The intersection of two sets is the set containing all common elements. If $A = \{1, 2, 3\}$ and $B = \{3, 4, 5\}$, then the intersection $A \cap B = \{3\}$.

- ▶ $\mathbb{Z} \cap \mathbb{R} = \mathbb{Z}$

The union of two sets is the set containing all elements that occur in either set. If $A = \{1, 2, 3\}$ and $B = \{3, 4, 5\}$, then the union $A \cup B = \{1, 2, 3, 4, 5\}$.

- ▶ $\mathbb{Z} \cup \mathbb{R} = \mathbb{R}$

Sets: subsets

A is a subset of B if all the elements of A are also contained in B . Written as $A \subset B$.

- ▶ $\{1, 2\} \subset \{1, 2, 3\}$
- ▶ $\mathbb{Z} \subset \mathbb{R}$
- ▶ $\mathbb{Z}_{++} \subset \mathbb{Z}_+ \subset \mathbb{Z} \subset \mathbb{R}$
- ▶ $\mathbb{N} \subset \mathbb{Z} \subset \mathbb{Q} \subset \mathbb{R}$

Vectors

Vectors. In general, an abstract mathematical notation, but for purpose of this module can be thought of as an ordered list of numbers.

E.g. **column vectors**:

$$\mathbf{x} = \begin{bmatrix} 1 \\ 3 \end{bmatrix} \quad \mathbf{y} = \begin{bmatrix} 3 \\ 1 \\ 8 \end{bmatrix}$$

To say that a vector \mathbf{x} is real valued with D dimensions, we write $\mathbf{x} \in \mathbb{R}^D$. E.g. $\mathbf{x} \in \mathbb{R}^2$, $\mathbf{y} \in \mathbb{R}^3$.

Can write column vectors more compactly using parentheses $\mathbf{x} = (1 \ 3)$

The elements of a vector are usually denoted using subscripts. E.g. if $\mathbf{x} = (1 \ 4 \ 5)$ then $x_1 = 1, x_2 = 4, x_3 = 5$.

The transpose of a column vector is a **row vector**.

$$\mathbf{x} = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} \quad \mathbf{x}^T = [1 \ 2 \ 3]$$

This gives us another way to write column vectors compactly:

$$\mathbf{x} = [x_1 \ x_2 \ x_3]^T \in \mathbb{R}^3$$

Adding and scaling vectors

To add two row or column vectors of the same dimension, just add their components.

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_D \end{bmatrix} \quad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_D \end{bmatrix} \quad \mathbf{x} + \mathbf{y} = \begin{bmatrix} x_1 + y_1 \\ x_2 + y_2 \\ \vdots \\ x_D + y_D \end{bmatrix}$$

You **cannot** add a row vector to a column vector or vice versa (unless the dimension is 1). You cannot add vectors of different dimensions.

To multiply a vector $\mathbf{x} \in \mathbb{R}^D$ by a scalar $\alpha \in \mathbb{R}$, just multiply each component.

$$\alpha \mathbf{x} = [\alpha x_1 \quad \alpha x_2 \quad \cdots \quad \alpha x_D]^T$$

Dot product

The **dot product** between two vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^D$ is computed by multiplying the corresponding components of \mathbf{x} and \mathbf{y} and adding up the products.

$$\begin{aligned}\mathbf{x} \cdot \mathbf{y} &= x_1 y_1 + x_2 y_2 + \cdots + x_D y_D \\ &= \sum_{i=1}^N x_i y_i\end{aligned}$$

The dot product is also called the **inner product** or **scalar product**.

Alternative notations:

- ▶ $\mathbf{x}^T \mathbf{y}$ (dot product as a matrix multiplication)
- ▶ (\mathbf{x}, \mathbf{y}) or $\langle \mathbf{x} | \mathbf{y} \rangle$ (braket notation, common in Physics)
- ▶ $\mathbf{x} \mathbf{y}$ implicit notation

Matrices

A rectangular array of numbers. E.g.:

$$A = \begin{bmatrix} 1 & 3 & 2 \\ 2 & 1 & 6 \end{bmatrix} \in \mathbb{R}^{2 \times 3}$$

A can be thought of as a horizontal stack of M row vectors or a vertical stack of N column vectors.

$$A = \begin{bmatrix} | & | & & | \\ \mathbf{a}_1 & \mathbf{a}_2 & \cdots & \mathbf{a}_N \\ | & | & & | \end{bmatrix} \quad A = \begin{bmatrix} \text{---} & \mathbf{a}_1^T & \text{---} \\ \text{---} & \mathbf{a}_2^T & \text{---} \\ \vdots & & \\ \text{---} & \mathbf{a}_M^T & \text{---} \end{bmatrix}$$

Beware of the overloaded notation! \mathbf{a}_i^T usually means the i th row rather than the i th column transposed.

Matrices: transpose

The transpose of a matrix A , denoted A^T is the same matrix with rows and columns interchanged. E.g.

$$A = \begin{bmatrix} 1 & 3 & 2 \\ 2 & 1 & 6 \end{bmatrix} \in \mathbb{R}^{2 \times 3} \quad A^T = \begin{bmatrix} 1 & 2 \\ 3 & 1 \\ 2 & 6 \end{bmatrix} \in \mathbb{R}^{3 \times 2}$$

Note:

- ▶ $(A + B)^T = A^T + B^T$ (transpose is linear)
- ▶ $(\alpha A)^T = \alpha(A^T)$ for $\alpha \in \mathbb{R}$ (transpose is linear)
- ▶ $\alpha^T = \alpha$ for $\alpha \in \mathbb{R}$ (transpose a scalar does nothing)

Adding and scaling matrices

Matrices of the same dimensions can be added together in the same way as with vectors: just add the corresponding components.

Matrices can be scaled by scalars by just multiplying each element by the scalar.

Matrix multiplication

Two matrices can be multiplied if the number of columns in the first matrix equals the number of rows in the second matrix.

E.g. it is possible to multiply A and B if $A \in \mathbb{R}^{M \times N}$ and $B \in \mathbb{R}^{N \times D}$. The resulting matrix AB will have dimension $M \times D$. **Rule:** inner dimension must match, outer dimension gives dimension of result.

The ij element of the product AB is equal to the inner product of row \mathbf{a}_i^T from A and column \mathbf{b}_j from B :

$$(AB)_{ij} = \mathbf{a}_i^T \mathbf{b}_j = \sum_{k=1}^M a_{ik} b_{kj}$$

Multiplying two matrices requires performing $M \times D$ inner products.

Matrix-vector multiplication

A matrix-vector multiplication $A\mathbf{x}$ can be performed if the number of columns in A equals the number of rows (entries) in \mathbf{x} .

A matrix-vector product can be interpreted as a linear combination of the columns of A .

$$A = \begin{bmatrix} | & | & & | \\ \mathbf{a}_1 & \mathbf{a}_2 & \cdots & \mathbf{a}_N \\ | & | & & | \end{bmatrix} \quad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix} \quad A\mathbf{x} = x_1\mathbf{a}_1 + x_2\mathbf{a}_2 + \dots + x_N\mathbf{a}_N$$

I.e. each element of \mathbf{x} scales a column of A and the result is the sum of these scaled columns.

It can also be interpreted as a set of M inner products between rows of A and \mathbf{x} with each output being put in the resulting vector.

Matrix vector multiplication

$$\begin{bmatrix} 1 & 6 & 3 & 2 \\ 2 & 3 & 5 & 7 \\ 9 & 2 & 1 & 1 \\ 7 & 8 & 1 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} = ?$$

$$\begin{bmatrix} 1 & 6 & 3 & 2 \\ 2 & 3 & 5 & 7 \\ 9 & 2 & 1 & 1 \\ 7 & 8 & 1 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} = ?$$

Matrix-matrix multiplication

Matrix-matrix multiplication AB can be interpreted as a series of matrix vector multiplications, one for each of the columns of B , with the results being stacked side-by-side in a matrix.

$$AB = [A\mathbf{b}_1 \quad A\mathbf{b}_2 \quad \cdots \quad A\mathbf{b}_D]$$

Rules of matrix multiplication:

1. Not commutative: in general $AB \neq BA$
2. Distributive: $A(B + C) = AB + AC$
3. Associative: $A(BC) = (AB)C = ABC$
4. Transposes: $(AB)^T = B^T A^T$

Square matrices, identities

A matrix is square if it has the same number of rows and columns. E.g. $A \in \mathbb{R}^{3 \times 3}$.

The matrix I is called the identity matrix. The 3×3 identity matrix is:

$$I = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Multiplication by the identity matrix gives back the original matrix.

- ▶ $AI = A$
- ▶ $IA = A$

Inverses

The square matrix B is said to be an inverse of the square matrix A if $AB = I$.

The inverse of A (if it exists) is denoted A^{-1}

- ▶ $AA^{-1} = I$
- ▶ $A^{-1}A = I$
- ▶ $(AB)^{-1} = B^{-1}A^{-1}$

Terminology:

- ▶ A square matrix that has an inverse is called **nonsingular** or **invertible** (also **nondegenerate**)
- ▶ A square matrix that has no inverse is called **singular** or **degenerate**

Computing inverses

Most numerical computation packages include functions for computing inverses. The algorithm used is usually LU decomposition or Gauss-Jordan elimination.

E.g. Python `numpy.linalg.inv(A)`. MATLAB `inv(A)`.

You can invert 2×2 matrices by hand with:

$$A = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \quad A^{-1} = \frac{1}{ad - bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

The matrix has no inverse if the determinant $\det(A) = 0$. In the 2×2 case the determinant is given by $\det(A) = ad - bc$

Systems of linear equations

A system of linear equations can be solved by using the inverse. E.g. consider the linear system:

$$3x + 2y - z = 1$$

$$2x - 2y + 4z = -2$$

$$-x + 0.5y - z = 0$$

This can be written in matrix form as $A\mathbf{x} = \mathbf{b}$ with:

$$A = \begin{bmatrix} 3 & 2 & -1 \\ 2 & -2 & 4 \\ -1 & 0.5 & -1 \end{bmatrix} \quad \mathbf{x} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \quad \mathbf{b} = \begin{bmatrix} 1 \\ -2 \\ 0 \end{bmatrix}$$

And can be solved by finding the inverse of A and multiplying by \mathbf{b} :

$$\mathbf{x} = A^{-1}\mathbf{b}$$

Solving systems of linear eqns numerically

You could do the calculation in Python as follows:

```
import numpy as np
A = np.array([[3,2,-1],[2,-2,4],[-1, 0.5, -1]])
b = np.array([1, -2, 0])
x = np.dot(np.linalg.inv(A), b)
print(x)
```

Which gives the answer $(1, -2, -2)$.

Note that computing inverses is usually not the most efficient way of doing this. Just call `numpy.linalg.solve` directly.

```
x = np.linalg.solve(A, b)
```

EE514

Data analysis and machine learning

Ali Intizar

Semester 1
2024/2025



Data Interpretation, Management, Storage, Wrangling and Cleansing



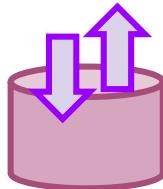
Ollscoil Chathair
Bhaile Átha Cliath
Dublin City University

Overview



Thinking about and interpreting data

- Datasets
- Data types
- Data as vectors and matrices



Data management and storage

- Storing data in files
- Storing data in databases



Data wrangling and cleaning

- Filtering and transforming
- Imputing missing values
- Fusing multiple data sources

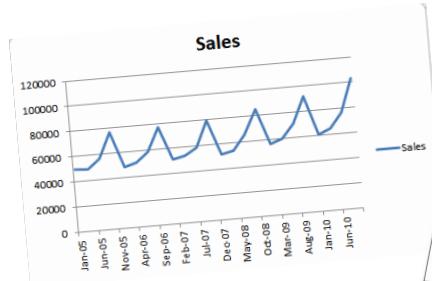


Interpreting data

Datasets

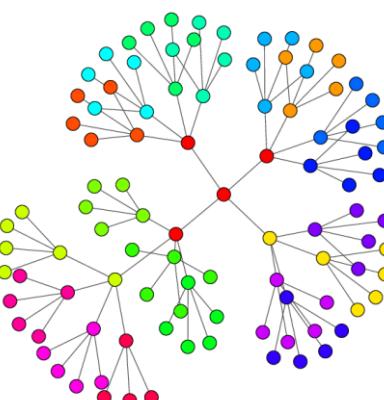
Data can come in a variety of forms

1. Series (1D)
2. Tables
3. Trees
4. Graphs
5. Text
6. Multimedia



This table provides detailed land area information for different forest types, categorized by Peat-Forming, Non-Peatlands, and Peatlands. The columns include Central Minwood (Cm), Lower Boreal Highlands (LBH), and Total LSA, along with percentages of LSA for each region.

			Central Minwood (Cm) Cm Area (ha)	Cm % of LSA	Lower Boreal Highlands (LBH) LBH Area (ha)	LBH % of LSA	Total LSA All	Total Area % of LSA
Peat-Forming	PBL_AWI	Description						
Non-Peatlands	SONS	Shrubby swamps						
	STNN	Treed swamps	58 395644	0.652220	437 908600	0.880993	496 305612	5 543212
	BPNN	Forested bog without permafrost or patterning, no internal lains	4 565384	0.050938	5 000000	0.000000	4 560384	0.050938
	BONS	Nonpatterned, open, shrub-dominated bog	0.000000	0.000000	28 920700	0.323014	28 920717	0.323014
	BTNN	Wooded bog with internal lains	4 470549	0.648951	914 008500	0.10 208516	918 479021	10 258446
	BTNN	Wooded bog without internal lains	0.000000	0.000000	3 371300	0.037654	3 371316	0.037654
	FONG	Nonpatterned, open, graminoid-dominated fens	25 385420	0.263520	1874 809500	0.20 056448	2000 185938	22 339976
	FONS	Nonpatterned, open, graminoid-dominated rich fens	4 162039	0.046491	168 712000	1.044238	172 674302	1 030830
	PTNN	Nonpatterned, open, shrub-dominated rich fens with islands of internal lains	1 681416	0.016780	516 960600	0.574302	518 677375	5 793082
	FTNN	Nonpatterned, wooded fens with islands of internal lains	0.000000	0.000000	31 725200	0.354337	31 725198	0.364337
All			98 656212	1 101986	4730 091442	7 300554	653 647287	7 300554



Examining datasets

Two types of information in datasets

- **Metadata:** data about data. Usually contains semantics (meaning)
- **Data:** the values. AKA attributes, features, measurements, variables, variates.

In tables:

- Rows contain **items** (AKA examples, instances, data points)
- Columns contain **attributes**
- Header contains semantics (metadata)

The titanic dataset

name	survived	sex	age	sibsp	parch	ticket
Allen, Miss. Elisabeth Walton	1	female	29	0	0	24160
Allison, Master. Hudson Trevor	1	male	0.9167	1	2	113781
Allison, Miss. Helen Loraine	0	female	2	1	2	113781
Allison, Mr. Hudson Joshua Creighton	0	male	30	1	2	113781
Allison, Mrs. Hudson J C (Bessie Wa	0	female	25	1	2	113781
Anderson, Mr. Harry	1	male	48	0	0	19952
Andrews, Miss. Kornelia Theodosia	1	female	63	1	0	13502
Andrews, Mr. Thomas Jr	0	male	39	0	0	112050
Appleton, Mrs. Edward Dale (Charlotte)	1	female	53	2	0	11769
Artagaveytia, Mr. Ramon	0	male	71	0	0	PC 17609
Astor, Col. John Jacob	0	male	47	1	0	PC 17757
Astor, Mrs. John Jacob (Madeleine Astor)	1	female	18	1	0	PC 17757
Aubart, Mme. Leontine Pauline	1	female	24	0	0	PC 17477
Barber, Miss. Ellen "Nellie"	1	female	26	0	0	19877
Barkworth, Mr. Algernon Henry Wilsden	1	male	80	0	0	27042
Baumann, Mr. John D	0	male		0	0	PC 17318
Baxter, Mr. Quigg Edmond	0	male	24	0	1	PC 17558
Baxter, Mrs. James (Helene DeLaude)	1	female	50	0	1	PC 17558

Attribute types

Many different ways we can classify the types of attributes in a dataset

Programmer types:

- String
- Integer
- Float
- Boolean

Mathematical sets:

- Real
- Complex
- Rational
- Integer

Much more useful in data analytics to classify them **according to which operations can be performed on them.**

The titanic dataset

name	survived	sex	age	sibsp	parch	ticket
Allen, Miss. Elisabeth Walton	1	female	29	0	0	24160
Allison, Master. Hudson Trevor	1	male	0.9167	1	2	113781
Allison, Miss. Helen Loraine	0	female	2	1	2	113781
Allison, Mr. Hudson Joshua Creighton	0	male	30	1	2	113781
Allison, Mrs. Hudson J C (Bessie Wa	0	female	25	1	2	113781
Anderson, Mr. Harry	1	male	48	0	0	19952
Andrews, Miss. Kornelia Theodosia	1	female	63	1	0	13502
Andrews, Mr. Thomas Jr	0	male	39	0	0	112050
Appleton, Mrs. Edward Dale (Charlotte)	1	female	53	2	0	11769
Artagaveytia, Mr. Ramon	0	male	71	0	0	PC 17609
Astor, Col. John Jacob	0	male	47	1	0	PC 17757
Astor, Mrs. John Jacob (Madeleine Astor)	1	female	18	1	0	PC 17757
Aubart, Mme. Leontine Pauline	1	female	24	0	0	PC 17477
Barber, Miss. Ellen "Nellie"	1	female	26	0	0	19877
Barkworth, Mr. Algernon Henry Wils	1	male	80	0	0	27042
Baumann, Mr. John D	0	male		0	0	PC 17318
Baxter, Mr. Quigg Edmond	0	male	24	0	1	PC 17558
Baxter, Mrs. James (Helene DeLaude)	1	female	50	0	1	PC 17558

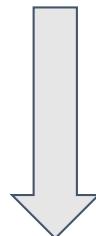
Scales of measurement

One of the most useful methods for classifying attribute types is **levels/scales of measurement**

- Defines what operations you are allowed to perform on the variables
 - (<, +, etc)

Stevens' four scales of measurement [Stevens, 1946]

1. Nominal
2. Ordinal
3. Interval
4. Ratio



Stronger assumptions!

Note: values also referred to as **variables**, **attributes**, and **measurements**

Nominal variables

In nominal measurement the values just name the attribute uniquely. No ordering of is implied.

Also called **categorical variables**

Examples:

- { male, female }
- { yes, no }
- { Ireland, UK, France, Spain, Italy, ... }
- { Ford, Nissan, Mercedes, ... }
- Vocabulary { a, the, person, hat ... }

Valid operations:

- $=, \neq$

Permissible statistics:

- Counts
- Modes
- Contingency correlation

It **never** make sense to compare nominal variables for order:

- Ireland > France ? 

It **never** makes sense to perform arithmetic on nominal variables:

- Yes + No = ? 
- AVERAGE([male, male, female]) 

Ordinal variables

Ordinal attributes (variables) can be **rank-ordered**.

Examples:

- Exam grade { A+, A, A-, B+, B, B-, ... }
- Clothing sizes { XS, S, M, L, XL }
- Position in a race { 1st, 2nd, 3rd }

Valid operations:



- $=, \neq, <, >, (\leq, \geq)$

Permissible statistics:

- Median
- Percentiles
- Spearman correlation

Ordered, so makes sense to compare:

- Large > Medium

Does **not make sense** to find mean, standard deviation, etc.

Quantitative variables

Quantities. Real numbers.

Two subtypes:

- **Interval:** distance between attributes **does** have meaning but there is no absolute zero.
- **Ratio:** same as interval but with a meaningful absolute zero.

Interval:

- Date (1 Jan)
- Temperature in degrees F.
- Geometric point

Ratio:

- Length, mass, temperature (in Kelvin)
- Age, height, weight

For both, we can do arithmetic

- Interval: $=$, \neq , $<$, $>$, $+$, $-$
- Ratio: $=$, \neq , $<$, $>$, $+$, $-$, \times , \div

Permissible statistics:

- Mean
- Standard deviation
- Pearson correlation

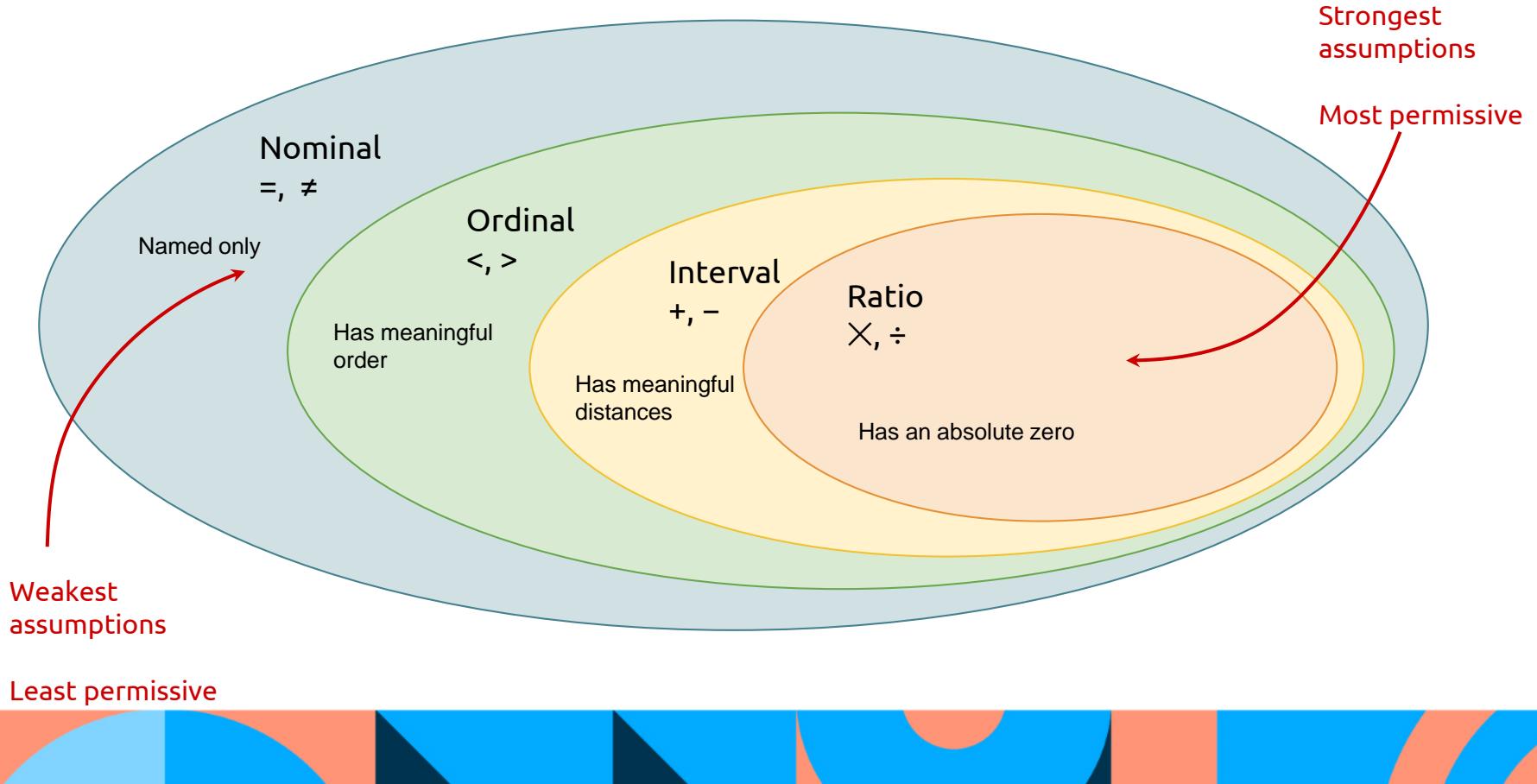
For ratio variables, **ratios make sense**:

- 10 meters is twice 5 meters

For interval variables, **only intervals make sense**

- 6 of Feb is ? to Jan 1?
- Interval: 6 of feb is 37 days after Jan 1

Hierarchy in levels of measurement



name	pclass	survived	sex	age	sibsp	parch	ticket	fare	cabin	embarked	boat	body	home.dest
Allen, Miss. Elisabeth Walton	1	1	female	29	0	0	24160	211.3375 B5	S	2			St Louis, MO
Allison, Master. Hudson Trevor	1	1	male	0.9167	1	2	113781	151.5500 C22 C26	S	11			Montreal, PQ / Chesterville
Allison, Miss. Helen Loraine	1	0	female	2	1	2	113781	151.5500 C22 C26	S				Montreal, PQ / Chesterville
Allison, Mr. Hudson Joshua Creighton	1	0	male	30	1	2	113781	151.5500 C22 C26	S		135		Montreal, PQ / Chesterville
Allison, Mrs. Hudson J C (Bessie Wa	1	0	female	25	1	2	113781	151.5500 C22 C26	S				Montreal, PQ / Chesterville
Anderson, Mr. Harry	1	1	male	48	0	0	19952	26.5500 E12	S	3			New York, NY
Andrews, Miss. Kornelia Theodosia	1	1	female	63	1	0	13502	77.9583 D7	S	10			Hudson, NY
Andrews, Mr. Thomas Jr	1	0	male	39	0	0	112050	0.0000 A36	S				Belfast, NI
Appleton, Mrs. Edward Dale (Charlotte)	1	1	female	53	2	0	11769	51.4792 C101	S	D			Bayside, Queens, NY
Artagaveytia, Mr. Ramon	1	0	male	71	0	0	PC 17609	49.5042	C		22		Montevideo, Uruguay
Astor, Col. John Jacob	1	0	male	47	1	0	PC 17757	227.5250 C62 C64	C		124		New York, NY
Astor, Mrs. John Jacob (Madeleine Astor)	1	1	female	18	1	0	PC 17757	227.5250 C62 C64	C	4			New York, NY
Aubart, Mme. Leontine Pauline	1	1	female	24	0	0	PC 17477	69.3000 B35	C	9			Paris, France
Barber, Miss. Ellen "Nellie"	1	1	female	26	0	0	19877	78.8500	S	6			
Barkworth, Mr. Algernon Henry Wils	1	1	male	80	0	0	27042	30.0000 A23	S	B			Hessle, Yorks
Baumann, Mr. John D	1	0	male	0	0	0	PC 17318	25.9250	S				New York, NY
Baxter, Mr. Quigg Edmond	1	0	male	24	0	1	PC 17558	247.5208 B58 B60	C				Montreal, PQ
Baxter, Mrs. James (Helene DeLaude)	1	1	female	50	0	1	PC 17558	247.5208 B58 B60	C	6			Montreal, PQ
Bazzani, Miss. Albina	1	1	female	32	0	0	11813	76.2917 D15	C	8			
Beattie, Mr. Thomson	1	0	male	36	0	0	13050	75.2417 C6	C	A			Winnipeg, MN
Beckwith, Mr. Richard Leonard	1	1	male	37	1	1	11751	52.5542 D35	S	5			New York, NY
Beckwith, Mrs. Richard Leonard (Sarah)	1	1	female	47	1	1	11751	52.5542 D35	S	5			New York, NY
Behr, Mr. Karl Howell	1	1	male	26	0	0	111369	30.0000 C148	C	5			New York, NY
Bidois, Miss. Rosalie	1	1	female	42	0	0	PC 17757	227.5250	C	4			
Bird, Miss. Ellen	1	1	female	29	0	0	PC 17483	221.7792 C97	S	8			
Birnbaum, Mr. Jakob	1	0	male	25	0	0	13905	26.0000	C		148		San Francisco, CA
Bishop, Mr. Dickinson H	1	1	male	25	1	0	11967	91.0792 B49	C	7			Dowagiac, MI
Bishop, Mrs. Dickinson H (Helen Wa	1	1	female	19	1	0	11967	91.0792 B49	C	7			Dowagiac, MI
Bissette, Miss. Amelia	1	1	female	35	0	0	PC 17760	135.6333 C99	S	8			
Bjornstrom-Steffansson, Mr. Mauritz	1	1	male	28	0	0	110564	26.5500 C52	S	D			Stockholm, Sweden / Wasl
Blackwell, Mr. Stephen Weart	1	0	male	45	0	0	113784	35.5000 T	S				Trenton, NJ
Blank, Mr. Henry	1	1	male	40	0	0	112277	31.0000 A31	C	7			Glen Ridge, NJ
Bonnell, Miss. Caroline	1	1	female	30	0	0	36928	164.8667 C7	S	8			Youngstown, OH
Bonnell, Miss. Elizabeth	1	1	female	58	0	0	113783	26.5500 C103	S	8			Birkdale, England Clevelar
Borebank, Mr. John James	1	0	male	42	0	0	110489	26.5500 D22	S				London / Winnipeg, MB
Bowen, Miss. Grace Scott	1	1	female	45	0	0	PC 17608	262.3750	C	4			Cooperstown, NY
Bowerman, Miss. Elsie Edith	1	1	female	22	0	1	113505	55.0000 E33	S	6			St Leonards-on-Sea, Engla

name	pclass	survived	sex	age	sibsp	parch	ticket	fare	cabin	embarked	boat	body	home.dest
Allen, Miss. Elisabeth Walton	1	0	Ordinal		0	0	24160	211.3375 B5	S	2			St Louis, MO
Allison, Master. Hudson Trevor	1	0			1	2	113781	151.5500 C22 C26	S	11			Montreal, PQ / Chesterville
Allison, Miss. Helen Loraine	1	0	female	2	1	2	113781	151.5500 C22 C26	S				Montreal, PQ / Chesterville
Allison, Mr. Hudson Joshua Creighton	1	0	male	30	1	2	113781	151.5500 C22 C26	S		135		Montreal, PQ / Chesterville
Allison, Mrs. Hudson J C (Bessie Wa	1	0	female	25	1	2	113781	151.5500 C22 C26	S				Montreal, PQ / Chesterville
Anderson, Mr. Harry	1	1	male	48	0	0	19952	26.5500 E12	S	3			New York, NY
Andrews, Miss. Kornelia Theodosia	1	1	female	63	1	0	13502	77.9583 D7	S	10			Hudson, NY
Andrews, Mr. Thomas Jr	1	0	male	39	0	0	112050	0.0000 A36	S				Belfast, NI
Appleton, Mrs. Edward Dale (Charlotte)	1	1	female	53	2	0	11769	51.4792 C101	S	D			Bayside, Queens, NY
Artagaveytia, Mr. Ramon	1	0	male	71	0	0	PC 17609	49.5042	C		22		Montevideo, Uruguay
Astor, Col. John Jacob	1	0	male	47	1	0	PC 17757	227.5250 C62 C64	C		124		New York, NY
Astor, Mrs. John Jacob (Madeleine T.	1	1	female	18	1	0	PC 17757	227.5250 C62 C64	C	4			New York, NY
Aubart, Mme. Leontine Pauline	1	1	female	24	0	0	PC 17477	69.3000 B35	C	9			Paris, France
Barber, Miss. Ellen "Nellie"	1	1	female	26	0	0	19877	78.8500	S	6			
Barkworth, Mr. Algernon Henry Wils	1	1	male	80	0	0	27042	30.0000 A23	S	B			Hessle, Yorks
Baumann, Mr. John D	1	0	male		0	0	PC 17318	25.9250	S				New York, NY
Baxter, Mr. Quigg Edmond	1	0	male	24	0	1	PC 17558	247.5208 B58 B60	C				Montreal, PQ
Baxter, Mrs. James (Helene DeLaude)	1	1	female	50	0	1	PC 17558	247.5208 B58 B60	C	6			Montreal, PQ
Bazzani, Miss. Albina	1	1	female	32	0	0	11813	76.2917 D15	C	8			
Beattie, Mr. Thomson	1	0	male	36	0	0	13050	75.2417 C6	C	A			Winnipeg, MN
Beckwith, Mr. Richard Leonard	1	1	male	37	1	1	11751	52.5542 D35	S	5			New York, NY
Beckwith, Mrs. Richard Leonard (Sarah)	1	1	female	47	1	1	11751	52.5542 D35	S	5			New York, NY
Behr, Mr. Karl Howell	1	1	male	26	0	0	111369	30.0000 C148	C	5			New York, NY
Bidois, Miss. Rosalie	1	1	female	42	0	0	PC 17757	227.5250	C	4			
Bird, Miss. Ellen	1	1	female	29	0	0	PC 17483	221.7792 C97	S	8			
Birnbaum, Mr. Jakob	1	0	male	25	0	0	13905	26.0000	C		148		San Francisco, CA
Bishop, Mr. Dickinson H	1	1	male	25	1	0	11967	91.0792 B49	C	7			Dowagiac, MI
Bishop, Mrs. Dickinson H (Helen Wa	1	1	female	19	1	0	11967	91.0792 B49	C	7			Dowagiac, MI
Bissette, Miss. Amelia	1	1	female	35	0	0	PC 17760	135.6333 C99	S	8			
Bjornstrom-Steffansson, Mr. Mauritz	1	1	male	28	0	0	110564	26.5500 C52	S	D			Stockholm, Sweden / Wash
Blackwell, Mr. Stephen Weart	1	0	male	45	0	0	113784	35.5000 T	S				Trenton, NJ
Blank, Mr. Henry	1	1	male	40	0	0	112277	31.0000 A31	C	7			Glen Ridge, NJ
Bonnell, Miss. Caroline	1	1	female	30	0	0	36928	164.8667 C7	S	8			Youngstown, OH
Bonnell, Miss. Elizabeth	1	1	female	58	0	0	113783	26.5500 C103	S	8			Birkdale, England Clevelar
Borebank, Mr. John James	1	0	male	42	0	0	110489	26.5500 D22	S				London / Winnipeg, MB
Bowen, Miss. Grace Scott	1	1	female	45	0	0	PC 17608	262.3750	C	4			Cooperstown, NY
Bowerman, Miss. Elsie Edith	1	1	female	22	0	1	113505	55.0000 E33	S	6			St Leonards-on-Sea, Engla

name	pclass	survived	sex	age	sibsp	parch	ticket	fare	cabin	embarked	boat	body	home.dest
Allen, Miss. Elisabeth Walton	1	1	female	29	0	0	24160	211.3375 B5	S	2			St Louis, MO
Allison, Master. Hudson Trevor	1	1	male	0.9167	1	2	113781	151.5500 C22 C26	S	11			Montreal, PQ / Chesterville
Allison, Miss. Helen Loraine	1	0	female	2	1	2	113781	151.5500 C22 C26	S				Montreal, PQ / Chesterville
Allison, Mr. Hudson Joshua Creighton	1	0	male	30	1	2	113781	151.5500 C22 C26	S		135		Montreal, PQ / Chesterville
Allison, Mrs. Hudson J C (Bessie Wa	1	0	female	25	1	2	113781	151.5500 C22 C26	S				Montreal, PQ / Chesterville
Anderson, Mr. Harry	1	1	male	48	0	0	19952	26.5500 E12	S	3			New York, NY
Andrews, Miss. Kornelia Theodosia	1	1	female	63	1	0	13502	77.9583 D7	S	10			Hudson, NY
Andrews, Mr. Thomas Jr	1	0	male	39	0	0	112050	0.0000 A36	S				Belfast, NI
Appleton, Mrs. Edward Dale (Charlot	1	1	female	53	2	0	11769	51.4792 C101	S	D			Bayside, Queens, NY
Artagaveytia, Mr. Ramon	1	0	male	71	0	0	PC 17609	49.5042	C		22		Montevideo, Uruguay
Astor, Col. John Jacob	1	0	male	47	1	0	PC 17757	227.5250 C62 C64	C		124		New York, NY
Astor, Mrs. John Jacob (Madeleine T:	1	1	female	18	1	0	PC 17757	227.5250 C62 C64	C	4			New York, NY
Aubart, Mme. Leontine Pauline	1	1	female	24	0	0	PC 17477	69.3000 B35	C	9			Paris, France
Barber, Miss. Ellen "Nellie"	1	1	female	26	0	0	19877	78.8500	S	6			
Barkworth, Mr. Algernon Henry Wils	1	1	male	80	0	0	27042	30.0000 A23	S	B			Hessle, Yorks
Baumann, Mr. John D	1	0	male		0	0	PC 17318	25.9250	S				New York, NY
Baxter, Mr. Quigg Edmond	1	0	male	24	0	1	PC 17558	247.5208 B58 B60	C				Montreal, PQ
Baxter, Mrs. James (Helene DeLaude	1	1	female	50	0	1	PC 17558	247.5208 B58 B60	C	6			Montreal, PQ
Bazzani, Miss. Albina	1	1	female	32	0	0	11813	76.2917 D15	C	8			
Beattie, Mr. Thomson	1	0	male	36	0	0	13050	75.2417 C6	C	A			Winnipeg, MN
Beckwith, Mr. Richard Leonard	1	1	male	37	1	1	11751	52.5542 D35	S	5			New York, NY
Beckwith, Mrs. Richard Leonard (Sal	1	1	female	47	1	1	11751	52.5542 D35	S	5			New York, NY
Behr, Mr. Karl Howell	1	1	male	26	0	0	111369	30.0000 C148	C	5			New York, NY
Bidois, Miss. Rosalie	1	1	female	42	0	0	PC 17757	227.5250	C	4			
Bird, Miss. Ellen	1	1	female	29	0	0	PC 17483	221.7792 C97	S	8			
Birnbaum, Mr. Jakob	1	0	male	25	0	0	13905	26.0000	C		148		San Francisco, CA
Bishop, Mr. Dickinson H	1	1	male	25	1	0	11967	91.0792 B49	C	7			Dowagiac, MI
Bishop, Mrs. Dickinson H (Helen Wa	1	1	female	19	1	0	11967	91.0792 B49	C	7			Dowagiac, MI
Bissette, Miss. Amelia	1	1	female	35	0	0	PC 17760	135.6333 C99	S	8			
Bjornstrom-Steffansson, Mr. Mauritz	1	1	male	28	0	0	110564	26.5500 C52	S	D			Stockholm, Sweden / Wash
Blackwell, Mr. Stephen Weart	1	0	male	45	0	0	113784	35.5000 T	S				Trenton, NJ
Blank, Mr. Henry	1	1	male	40	0	0	112277	31.0000 A31	C	7			Glen Ridge, NJ
Bonnell, Miss. Caroline	1	1	female	30	0	0	36928	164.8667 C7	S	8			Youngstown, OH
Bonnell, Miss. Elizabeth	1	1	female	58	0	0	113783	26.5500 C103	S	8			Birkdale, England Clevelar
Borebank, Mr. John James	1	0	male	42	0	0	110489	26.5500 D22	S				London / Winnipeg, MB
Bowen, Miss. Grace Scott	1	1	female	45	0	0	PC 17608	262.3750	C	4			Cooperstown, NY
Bowerman, Miss. Elsie Edith	1	1	female	22	0	1	113505	55.0000 E33	S	6			St Leonards-on-Sea, Engla

name	pclass	survived	sex	age	sibsp	parch	ticket	fare	cabin	embarked	boat	body	home.dest
Allen, Miss. Elisabeth Walton	1	1	Nominal							S	2		St Louis, MO
Allison, Master. Hudson Trevor	1	1				2	24160	211.3375 B5					Montreal, PQ / Chesterville
Allison, Miss. Helen Loraine	1	0	female	2	1	2	113781	151.5500 C22 C26					Montreal, PQ / Chesterville
Allison, Mr. Hudson Joshua Creighto	1	0	male	30	1	2	113781	151.5500 C22 C26					Montreal, PQ / Chesterville
Allison, Mrs. Hudson J C (Bessie Wa	1	0	female	25	1	2	113781	151.5500 C22 C26					Montreal, PQ / Chesterville
Anderson, Mr. Harry	1	1	male	48	0	0	19952	26.5500 E12			3		New York, NY
Andrews, Miss. Kornelia Theodosia	1	1	female	63	1	0	13502	77.9583 D7			10		Hudson, NY
Andrews, Mr. Thomas Jr	1	0	male	39	0	0	112050	0.0000 A36					Belfast, NI
Appleton, Mrs. Edward Dale (Charlot	1	1	female	53	2	0	11769	51.4792 C101			D		Bayside, Queens, NY
Artagaveytia, Mr. Ramon	1	0	male	71	0	0	PC 17609	49.5042	C				22 Montevideo, Uruguay
Astor, Col. John Jacob	1	0	male	47	1	0	PC 17757	227.5250 C62 C64					124 New York, NY
Astor, Mrs. John Jacob (Madeleine T:	1	1	female	18	1	0	PC 17757	227.5250 C62 C64			4		New York, NY
Aubart, Mme. Leontine Pauline	1	1	female	24	0	0	PC 17477	69.3000 B35			9		Paris, France
Barber, Miss. Ellen "Nellie"	1	1	female	26	0	0	19877	78.8500			6		
Barkworth, Mr. Algernon Henry Wils	1	1	male	80	0	0	27042	30.0000 A23			B		Hessle, Yorks
Baumann, Mr. John D	1	0	male		0	0	PC 17318	25.9250					New York, NY
Baxter, Mr. Quigg Edmond	1	0	male	24	0	1	PC 17558	247.5208 B58 B60					Montreal, PQ
Baxter, Mrs. James (Helene DeLaude	1	1	female	50	0	1	PC 17558	247.5208 B58 B60			6		Montreal, PQ
Bazzani, Miss. Albina	1	1	female	32	0	0	11813	76.2917 D15			8		
Beattie, Mr. Thomson	1	0	male	36	0	0	13050	75.2417 C6			A		Winnipeg, MN
Beckwith, Mr. Richard Leonard	1	1	male	37	1	1	11751	52.5542 D35			5		New York, NY
Beckwith, Mrs. Richard Leonard (Sal	1	1	female	47	1	1	11751	52.5542 D35			5		New York, NY
Behr, Mr. Karl Howell	1	1	male	26	0	0	111369	30.0000 C148			5		New York, NY
Bidois, Miss. Rosalie	1	1	female	42	0	0	PC 17757	227.5250			4		
Bird, Miss. Ellen	1	1	female	29	0	0	PC 17483	221.7792 C97			8		
Birnbaum, Mr. Jakob	1	0	male	25	0	0	13905	26.0000					148 San Francisco, CA
Bishop, Mr. Dickinson H	1	1	male	25	1	0	11967	91.0792 B49			7		Dowagiac, MI
Bishop, Mrs. Dickinson H (Helen Wa	1	1	female	19	1	0	11967	91.0792 B49			7		Dowagiac, MI
Bissette, Miss. Amelia	1	1	female	35	0	0	PC 17760	135.6333 C99			8		
Bjornstrom-Steffansson, Mr. Mauritz	1	1	male	28	0	0	110564	26.5500 C52			D		Stockholm, Sweden / Wash
Blackwell, Mr. Stephen Weart	1	0	male	45	0	0	113784	35.5000 T					Trenton, NJ
Blank, Mr. Henry	1	1	male	40	0	0	112277	31.0000 A31			7		Glen Ridge, NJ
Bonnell, Miss. Caroline	1	1	female	30	0	0	36928	164.8667 C7			8		Youngstown, OH
Bonnell, Miss. Elizabeth	1	1	female	58	0	0	113783	26.5500 C103			8		Birkdale, England Clevelan
Borebank, Mr. John James	1	0	male	42	0	0	110489	26.5500 D22					London / Winnipeg, MB
Bowen, Miss. Grace Scott	1	1	female	45	0	0	PC 17608	262.3750			4		Cooperstown, NY
Bowerman, Miss. Elsie Edith	1	1	female	22	0	1	113505	55.0000 E33			6		St Leonards-on-Sea, Engla

name	pclass	survived	sex	age	sibsp	parch	ticket	fare	cabin	embarked	boat	body	home.dest
Allen, Miss. Elisabeth Walton	1	1	female	29	0	0	24160	211.3375 B5	S	2			St Louis, MO
Allison, Master. Hudson Trevor	1	1	male	0.9167	1	2	113781	151.5500 C22 C26	S	11			Montreal, PQ / Chesterville
Allison, Miss. Helen Loraine	1	0	female	2	1	2	113781	151.5500 C22 C26	S				Montreal, PQ / Chesterville
Allison, Mr. Hudson Joshua Creighton	1	0	male	30	1	2	113781	151.5500 C22 C26	S		135		Montreal, PQ / Chesterville
Allison, Mrs. Hudson J C (Bessie Wa	1	0	female	25	1	2	113781	151.5500 C22 C26	S				Montreal, PQ / Chesterville
Anderson, Mr. Harry	1	1	male	48	0	0	19952	26.5500 E12	S	3			New York, NY
Andrews, Miss. Kornelia Theodosia	1	1	female	63	1	0	13502	77.9583 D7	S	10			Hudson, NY
Andrews, Mr. Thomas Jr	1	0	male	39	0	0	112050	0.0000 A36	S				Belfast, NI
Appleton, Mrs. Edward Dale (Charlotte)	1	1	female	53	2	0	11769	51.4792 C101	S	D			Bayside, Queens, NY
Artagaveytia, Mr. Ramon	1	0	male	71	0	0	PC 17609	49.5042	C		22		Montevideo, Uruguay
Astor, Col. John Jacob	1	0	male	47	1	0	PC 17757	227.5250 C62 C64	C		124		New York, NY
Astor, Mrs. John Jacob (Madeleine T.)	1	1	female	18	1	0	PC 17757	227.5250 C62 C64	C	4			New York, NY
Aubart, Mme. Leontine Pauline	1	1	female	24	0	0	PC 17477	69.3000 B35	C	9			Paris, France
Barber, Miss. Ellen "Nellie"	1	1	female	26	0	0	19877	78.8500	S	6			
Barkworth, Mr. Algernon Henry Wils	1	1	male	80	0	0	27042	30.0000 A23	S	B			Hessle, Yorke
Baumann, Mr. John D	1	0	male		0	0	PC 17318	25.9250	S				New York, NY
Baxter, Mr. Quigg Edmond	1	0	male	24	0	1	PC 17558	247.5208 B58 B60	C				Montreal, PQ
Baxter, Mrs. James (Helene DeLaude)	1	1	female	50	0	1	PC 17558	247.5208 B58 B60	C	6			Montreal, PQ
Bazzani, Miss. Albina	1	1	female	32	0	0	11813	76.2917 D15	C	8			
Beattie, Mr. Thomson	1	0	male	36	0	0	13050	75.2417 C6	C	A			Winnipeg, MN
Beckwith, Mr. Richard Leonard	1	1	male	37	1	1	11751	52.5542 D35	S	5			New York, NY
Beckwith, Mrs. Richard Leonard (Sal)	1	1	female	47	1	1	11751	52.5542 D35	S	5			New York, NY
Behr, Mr. Karl Howell	1	1	male	26	0	0	111369	30.0000 C148	C	5			New York, NY
Bidois, Miss. Rosalie	1	1	female	42	0	0	PC 17757	227.5250	C	4			
Bird, Miss. Ellen	1	1	female	29	0	0	PC 17483	221.7792 C97	S	8			
Birnbaum, Mr. Jakob	1	0	male	25	0	0	13905	26.0000	C		148		San Francisco, CA
Bishop, Mr. Dickinson H	1	1	male	25	1	0	11967	91.0792 B49	C	7			Dowagiac, MI
Bishop, Mrs. Dickinson H (Helen Wa	1	1	female	19	1	0	11967	91.0792 B49	C	7			Dowagiac, MI
Bissette, Miss. Amelia	1	1	female	35	0	0	PC 17760	135.6333 C99	S	8			
Bjornstrom-Steffansson, Mr. Mauritz	1	1	male	28	0	0	110564	26.5500 C52	S	D			Stockholm, Sweden / Wash
Blackwell, Mr. Stephen Weart	1	0	male	45	0	0	113784	35.5000 T	S				Trenton, NJ
Blank, Mr. Henry	1	1	male	40	0	0	112277	31.0000 A31	C	7			Glen Ridge, NJ
Bonnell, Miss. Caroline	1	1	female	30	0	0	36928	164.8667 C7	S	8			Youngstown, OH
Bonnell, Miss. Elizabeth	1	1	female	58	0	0	113783	26.5500 C103	S	8			Birkdale, England Clevelar
Borebank, Mr. John James	1	0	male	42	0	0	110489	26.5500 D22	S				London / Winnipeg, MB
Bowen, Miss. Grace Scott	1	1	female	45	0	0	PC 17608	262.3750	C	4			Cooperstown, NY
Bowerman, Miss. Elsie Edith	1	1	female	22	0	1	113505	55.0000 E33	S	6			St Leonards-on-Sea, Engla

name	pclass	survived	sex	age	sibsp	parch	ticket	fare	cabin	embarked	boat	body	home.dest
Allen, Miss. Elisabeth Walton	1	1	female	29	0	0	24160	211.3375 B5	S	2		St Louis, MO	
Allison, Master. Hudson Trevor	1	1	male	0.9167	1	2	113781	151.5500 C22 C26	S	11		Montreal, PQ / Chesterville	
Allison, Miss. Helen Loraine	1	0	female	2	1	2	113781	151.5500 C22 C26	S			Montreal, PQ / Chesterville	
Allison, Mr. Hudson Joshua Creighton	1	0	male				113781	151.5500 C22 C26	S		135	Montreal, PQ / Chesterville	
Allison, Mrs. Hudson J C (Bessie Wa	1	0	female	25	1	2	113781	151.5500 C22 C26	S			Montreal, PQ / Chesterville	
Anderson, Mr. Harry	1	1	male	48	0	0	19952	26.5500 E12	S	3		New York, NY	
Andrews, Miss. Kornelia Theodosia	1	1	female	63	1	0	13502	77.9583 D7	S	10		Hudson, NY	
Andrews, Mr. Thomas Jr	1	0	male	39	0	0	112050	0.0000 A36	S			Belfast, NI	
Appleton, Mrs. Edward Dale (Charlotte)	1	1	female	53	2	0	11769	51.4792 C101	S	D		Bayside, Queens, NY	
Artagaveytia, Mr. Ramon	1	0	male	71	0	0	PC 17609	49.5042	C		22	Montevideo, Uruguay	
Astor, Col. John Jacob	1	0	male	47	1	0	PC 17757	227.5250 C62 C64	C		124	New York, NY	
Astor, Mrs. John Jacob (Madeleine T.	1	1	female	18	1	0	PC 17757	227.5250 C62 C64	C	4		New York, NY	
Aubart, Mme. Leontine Pauline	1	1	female	24	0	0	PC 17477	69.3000 B35	C	9		Paris, France	
Barber, Miss. Ellen "Nellie"	1	1	female	26	0	0	19877	78.8500	S	6			
Barkworth, Mr. Algernon Henry Wils	1	1	male	80	0	0	27042	30.0000 A23	S	B		Hessle, Yorks	
Baumann, Mr. John D	1	0	male		0	0	PC 17318	25.9250	S			New York, NY	
Baxter, Mr. Quigg Edmond	1	0	male	24	0	1	PC 17558	247.5208 B58 B60	C			Montreal, PQ	
Baxter, Mrs. James (Helene DeLaude)	1	1	female	50	0	1	PC 17558	247.5208 B58 B60	C	6		Montreal, PQ	
Bazzani, Miss. Albina	1	1	female	32	0	0	11813	76.2917 D15	C	8			
Beattie, Mr. Thomson	1	0	male	36	0	0	13050	75.2417 C6	C	A		Winnipeg, MN	
Beckwith, Mr. Richard Leonard	1	1	male	37	1	1	11751	52.5542 D35	S	5		New York, NY	
Beckwith, Mrs. Richard Leonard (Sarah)	1	1	female	47	1	1	11751	52.5542 D35	S	5		New York, NY	
Behr, Mr. Karl Howell	1	1	male	26	0	0	111369	30.0000 C148	C	5		New York, NY	
Bidois, Miss. Rosalie	1	1	female	42	0	0	PC 17757	227.5250	C	4			
Bird, Miss. Ellen	1	1	female	29	0	0	PC 17483	221.7792 C97	S	8			
Birnbaum, Mr. Jakob	1	0	male	25	0	0	13905	26.0000	C		148	San Francisco, CA	
Bishop, Mr. Dickinson H	1	1	male	25	1	0	11967	91.0792 B49	C	7		Dowagiac, MI	
Bishop, Mrs. Dickinson H (Helen Wa	1	1	female	19	1	0	11967	91.0792 B49	C	7		Dowagiac, MI	
Bissette, Miss. Amelia	1	1	female	35	0	0	PC 17760	135.6333 C99	S	8			
Bjornstrom-Steffansson, Mr. Mauritz	1	1	male	28	0	0	110564	26.5500 C52	S	D		Stockholm, Sweden / Wash	
Blackwell, Mr. Stephen Weart	1	0	male	45	0	0	113784	35.5000 T	S			Trenton, NJ	
Blank, Mr. Henry	1	1	male	40	0	0	112277	31.0000 A31	C	7		Glen Ridge, NJ	
Bonnell, Miss. Caroline	1	1	female	30	0	0	36928	164.8667 C7	S	8		Youngstown, OH	
Bonnell, Miss. Elizabeth	1	1	female	58	0	0	113783	26.5500 C103	S	8		Birkdale, England Clevelar	
Borebank, Mr. John James	1	0	male	42	0	0	110489	26.5500 D22	S			London / Winnipeg, MB	
Bowen, Miss. Grace Scott	1	1	female	45	0	0	PC 17608	262.3750	C	4		Cooperstown, NY	
Bowerman, Miss. Elsie Edith	1	1	female	22	0	1	113505	55.0000 E33	S	6		St Leonards-on-Sea, Engla	

Nominal

name	pclass	survived	sex	age	sibsp	parch	ticket	fare	cabin	embarked	boat	body	home.dest
Allen, Miss. Elisabeth Walton	1	1	female	29	0	0	24160	211.3375 B5	S	2			St Louis, MO
Allison, Master. Hudson Trevor	1	1	male	0.9167	1	2	113781	151.5500 C22 C26	S	11			Montreal, PQ / Chesterville
Allison, Miss. Helen Loraine	1	0	female	2	1	2	113781	151.5500 C22 C26	S				Montreal, PQ / Chesterville
Allison, Mr. Hudson Joshua Creighton	1	0	male	30	1	2	113781	151.5500 C22 C26	S		135		Montreal, PQ / Chesterville
Allison, Mrs. Hudson J C (Bessie Wa	1	0	female	25	1	2	113781	151.5500 C22 C26	S				Montreal, PQ / Chesterville
Anderson, Mr. Harry	1	1	male	48	0	0	19952	26.5500 E12	S	3			New York, NY
Andrews, Miss. Kornelia Theodosia	1	1	female	63	1	0	13502	77.9583 D7	S	10			Hudson, NY
Andrews, Mr. Thomas Jr	1	0	male	39	0	0	112050	0.0000 A36	S				Belfast, NI
Appleton, Mrs. Edward Dale (Charlotte)	1	1	female	53	2	0	11769	51.4792 C101	S	D			Bayside, Queens, NY
Artagaveytia, Mr. Ramon	1	0	male	71	0	0	PC 17609	49.5042	C		22		Montevideo, Uruguay
Astor, Col. John Jacob	1	0	male	47	1	0	PC 17757	227.5250 C62 C64	C		124		New York, NY
Astor, Mrs. John Jacob (Madeleine Travers)	1	1	female	18	1	0	PC 17757	227.5250 C62 C64	C	4			New York, NY
Aubart, Mme. Leontine Pauline	1	1	female	24	0	0	PC 17477	69.3000 B35	C	9			Paris, France
Barber, Miss. Ellen "Nellie"	1	1	female	26	0	0	19877	78.8500	S	6			
Barkworth, Mr. Algernon Henry Wils	1	1	male	80	0	0	27042	30.0000 A23	S	B			Hessle, Yorks
Baumann, Mr. John D	1	0	male	0	0	0	PC 17318	25.9250	S				New York, NY
Baxter, Mr. Quigg Edmond	1	0	male	24	0	1	PC 17558	247.5208 B58 B60	C				Montreal, PQ
Baxter, Mrs. James (Helene DeLaude)	1	1	female	50	0	1	PC 17558	247.5208 B58 B60	C	6			Montreal, PQ
Bazzani, Miss. Albina	1	1	female	32	0	0	11813	76.2917 D15	C	8			
Beattie, Mr. Thomson	1	0	male	36	0	0	13050	75.2417 C6	C	A			Winnipeg, MN
Beckwith, Mr. Richard Leonard	1	1	male	37	1	1	11751	52.5542 D35	S	5			New York, NY
Beckwith, Mrs. Richard Leonard (Salome)	1	1	female	47	1	1	11751	52.5542 D35	S	5			New York, NY
Behr, Mr. Karl Howell	1	1	male	26	0	0	111369	30.0000 C148	C	5			New York, NY
Bidois, Miss. Rosalie	1	1	female	42	0	0	PC 17757	227.5250	C	4			
Bird, Miss. Ellen	1	1	female	29	0	0	PC 17483	221.7792 C97	S	8			
Birnbaum, Mr. Jakob	1	0	male	25	0	0	13905	26.0000	C		148		San Francisco, CA
Bishop, Mr. Dickinson H	1	1	male	25	1	0	11967	91.0792 B49	C	7			Dowagiac, MI
Bishop, Mrs. Dickinson H (Helen Wallace)	1	1	female	19	1	0	11967	91.0792 B49	C	7			Dowagiac, MI
Bissette, Miss. Amelia	1	1	female	35	0	0	PC 17760	135.6333 C99	S	8			
Bjornstrom-Steffansson, Mr. Mauritz	1	1	male	28	0	0	110564	26.5500 C52	S	D			Stockholm, Sweden / Washtenaw, MI
Blackwell, Mr. Stephen Weart	1	0	male	45	0	0	113784	35.5000 T	S				Trenton, NJ
Blank, Mr. Henry	1	1	male	40	0	0	112277	31.0000 A31	C	7			Glen Ridge, NJ
Bonnell, Miss. Caroline	1	1	female	30	0	0	36928	164.8667 C7	S	8			Youngstown, OH
Bonnell, Miss. Elizabeth	1	1	female	58	0	0	113783	26.5500 C103	S	8			Birkdale, England Cleveland
Borebank, Mr. John James	1	0	male	42	0	0	110489	26.5500 D22	S				London / Winnipeg, MB
Bowen, Miss. Grace Scott	1	1	female	45	0	0	PC 17608	262.3750	C	4			Cooperstown, NY
Bowerman, Miss. Elsie Edith	1	1	female	22	0	1	113505	55.0000 E33	S	6			St Leonards-on-Sea, England

name	pclass	survived	sex	age	sibsp	parch	ticket	fare	cabin	embarked	boat	body	home.dest
Allen, Miss. Elisabeth Walton	1	1	female	29	0	0	24160	211.3375 B5	S	2			St Louis, MO
Allison, Master. Hudson Trevor	1	1	male	0.9167	1	2	113781	151.5500 C22 C26	S	11			Montreal, PQ / Chesterville
Allison, Miss. Helen Loraine	1	0	female	2	1	2	113781	151.5500 C22 C26	S				Montreal, PQ / Chesterville
Allison, Mr. Hudson Joshua Creighton	1	0	male	30	1	2	113781	151.5500 C22 C26	S		135		Montreal, PQ / Chesterville
Allison, Mrs. Hudson J C (Bessie Wa	1	0	female	25			81	151.5500 C22 C26	S				Montreal, PQ / Chesterville
Anderson, Mr. Harry	1	1	male	48			52	26.5500 E12	S	3			New York, NY
Andrews, Miss. Kornelia Theodosia	1	1	female	63	1	0	13502	77.9583 D7	S	10			Hudson, NY
Andrews, Mr. Thomas Jr	1	0	male	39	0	0	112050	0.0000 A36	S				Belfast, NI
Appleton, Mrs. Edward Dale (Charlotte)	1	1	female	53	2	0	11769	51.4792 C101	S		D		Bayside, Queens, NY
Artagaveytia, Mr. Ramon	1	0	male	71	0	0	PC 17609	49.5042	C			22	Montevideo, Uruguay
Astor, Col. John Jacob	1	0	male	47	1	0	PC 17757	227.5250 C62 C64	C			124	New York, NY
Astor, Mrs. John Jacob (Madeleine Travers)	1	1	female	18	1	0	PC 17757	227.5250 C62 C64	C	4			New York, NY
Aubart, Mme. Leontine Pauline	1	1	female	24	0	0	PC 17477	69.3000 B35	C	9			Paris, France
Barber, Miss. Ellen "Nellie"	1	1	female	26	0	0	19877	78.8500	S	6			
Barkworth, Mr. Algernon Henry Wilsden	1	1	male	80	0	0	27042	30.0000 A23	S		B		Hessle, Yorks
Baumann, Mr. John D	1	0	male		0	0	PC 17318	25.9250	S				New York, NY
Baxter, Mr. Quigg Edmond	1	0	male	24	0	1	PC 17558	247.5208 B58 B60	C				Montreal, PQ
Baxter, Mrs. James (Helene DeLauder)	1	1	female	50	0	1	PC 17558	247.5208 B58 B60	C	6			Montreal, PQ
Bazzani, Miss. Albina	1	1	female	32	0	0	11813	76.2917 D15	C	8			
Beattie, Mr. Thomson	1	0	male	36	0	0	13050	75.2417 C6	C		A		Winnipeg, MN
Beckwith, Mr. Richard Leonard	1	1	male	37	1	1	11751	52.5542 D35	S	5			New York, NY
Beckwith, Mrs. Richard Leonard (Sarah)	1	1	female	47	1	1	11751	52.5542 D35	S	5			New York, NY
Behr, Mr. Karl Howell	1	1	male	26	0	0	111369	30.0000 C148	C	5			New York, NY
Bidois, Miss. Rosalie	1	1	female	42	0	0	PC 17757	227.5250	C	4			
Bird, Miss. Ellen	1	1	female	29	0	0	PC 17483	221.7792 C97	S	8			
Birnbaum, Mr. Jakob	1	0	male	25	0	0	13905	26.0000	C			148	San Francisco, CA
Bishop, Mr. Dickinson H	1	1	male	25	1	0	11967	91.0792 B49	C	7			Dowagiac, MI
Bishop, Mrs. Dickinson H (Helen Waite)	1	1	female	19	1	0	11967	91.0792 B49	C	7			Dowagiac, MI
Bissette, Miss. Amelia	1	1	female	35	0	0	PC 17760	135.6333 C99	S	8			
Bjornstrom-Steffansson, Mr. Mauritz	1	1	male	28	0	0	110564	26.5500 C52	S		D		Stockholm, Sweden / Washington, DC
Blackwell, Mr. Stephen Weart	1	0	male	45	0	0	113784	35.5000 T	S				Trenton, NJ
Blank, Mr. Henry	1	1	male	40	0	0	112277	31.0000 A31	C	7			Glen Ridge, NJ
Bonnell, Miss. Caroline	1	1	female	30	0	0	36928	164.8667 C7	S	8			Youngstown, OH
Bonnell, Miss. Elizabeth	1	1	female	58	0	0	113783	26.5500 C103	S	8			Birkdale, England/Cleveland, OH
Borebank, Mr. John James	1	0	male	42	0	0	110489	26.5500 D22	S				London / Winnipeg, MB
Bowen, Miss. Grace Scott	1	1	female	45	0	0	PC 17608	262.3750	C	4			Cooperstown, NY
Bowerman, Miss. Elsie Edith	1	1	female	22	0	1	113505	55.0000 E33	S	6			St Leonards-on-Sea, England

Numbers of variables

Univariate data

- You only have one attribute. E.g. time series

Velocity	35	50	60	40	20

Bivariate data

- You have two attributes. E.g. a table of longitude and latitude pairs

lon	lat
-6.258854	53.385381
-6.274475	53.360158
-71.094664	42.359980

Multivariate data

- You have $N > 1$ attributes

name	survived	sex	age	sibsp	parch	ticket	fare	c
Allen, Miss. Elisabeth Walton	1	female	29	0	0	24160	211.3375	B5
Allison, Master. Hudson Trevor	1	male	0.9167	1	2	113781	151.5500	C22 C
Allison, Miss. Helen Loraine	0	female	2	1	2	113781	151.5500	C22 C
Allison, Mr. Hudson Joshua Creighton	0	male	30	1	2	113781	151.5500	C22 C
Allison, Mrs. Hudson J C (Bessie Wa	0	female	25	1	2	113781	151.5500	C22 C
Anderson, Mr. Harry	1	male	48	0	0	19952	26.5500	E12
Andrews, Miss. Kornelia Theodosia	1	female	63	1	0	13502	77.9583	D7
Andrews, Mr. Thomas Jr	0	male	39	0	0	112050	0.0000	A36
Appleton, Mrs. Edward Dale (Charlotte)	1	female	53	2	0	11769	51.4792	C101
Artagaveytia, Mr. Ramon	0	male	71	0	0	PC 17609	49.5042	
Astor, Col. John Jacob	0	male	47	1	0	PC 17757	227.5250	C62 C

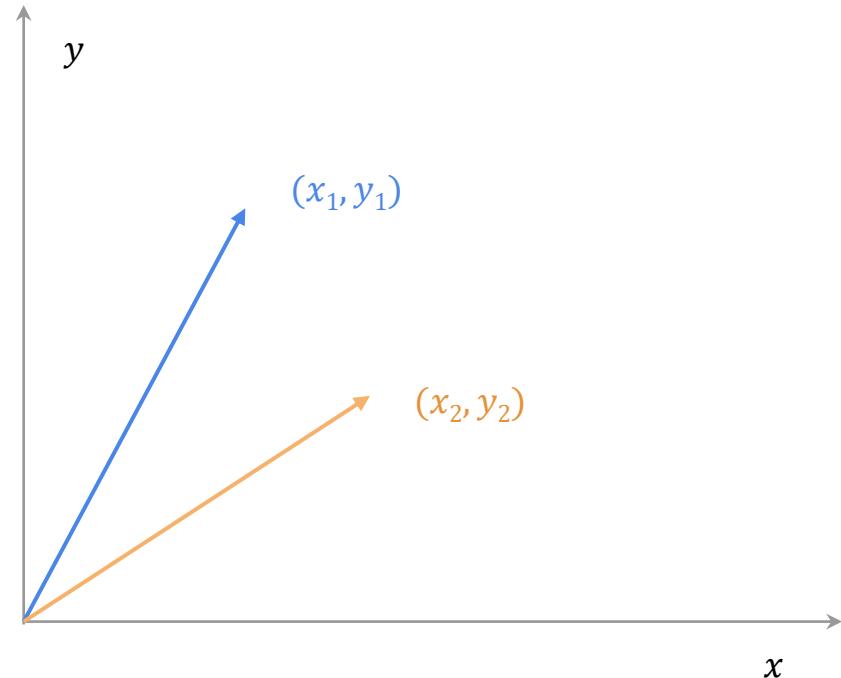
Data as vectors

Bivariate data points with **quantitative** variables can be described using vectors in 2D space.

$$\begin{bmatrix} 1 & 3 \end{bmatrix}^T \begin{bmatrix} 1 \\ 2 \end{bmatrix} \begin{bmatrix} 3 \\ 2 \end{bmatrix} \in R^2$$

Multivariate data points can be described using vectors in D -dimensional space

$$\begin{bmatrix} 2 \\ 3 \\ 1 \\ 8 \end{bmatrix} \in R^D$$



This abstraction is very useful, since it allows us to use linear algebra theory to manipulate data

Datasets as matrices

We can stack quantitative data vectors into matrices.

Usually we stack the items (data points, examples) in the rows, and the attributes (features) in the columns.

- NB: some books/papers do the opposite!

For a dataset X with N items and D attributes, we have a $(N \times D)$ matrix

$$X \in R^{N \times D}$$

Data points

$$\begin{bmatrix} x_1^T \\ x_2^T \\ \vdots \\ x_N^T \end{bmatrix}$$

N items

0.255	0.123	0.127
0.649	0.057	0.476
0.379	0.184	0.943
0.471	0.511	0.092
0.647	0.866	0.759
0.475	0.345	0.858

D attributes

One-hot encoding

Sometimes we want a vector encoding for a **nominal** variable

Solution: one-hot encoding

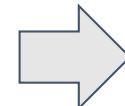
Nominal attribute with K possible values becomes a K -dimensional vector

Transform a categorical variable with K categories into K binary variables

Very useful for encoding text

Categorical attribute with 5 categories

2
3
1
4
5



0	1	0	0	0
0	0	1	0	0
1	0	0	0	0
0	0	0	1	0
0	0	0	0	1

One hot-encoding: 5 binary attributes (sparse matrix)

One-hot encoding of text

Can use one-hot encoding of words.

Set of all possible words is called the **vocabulary**.

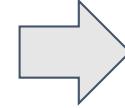
The **codebook** assigns each word to an integer

Codebook

The	1
cat	2
sat	3
on	4
mat	5

The	1
cat	2
sat	3
on	4
the	1
mat	5

"The cat sat on the mat"



1	0	0	0	0
0	1	0	0	0
0	0	1	0	0
0	0	0	1	0
1	0	0	0	0
0	0	0	0	1

Bag of words model of text

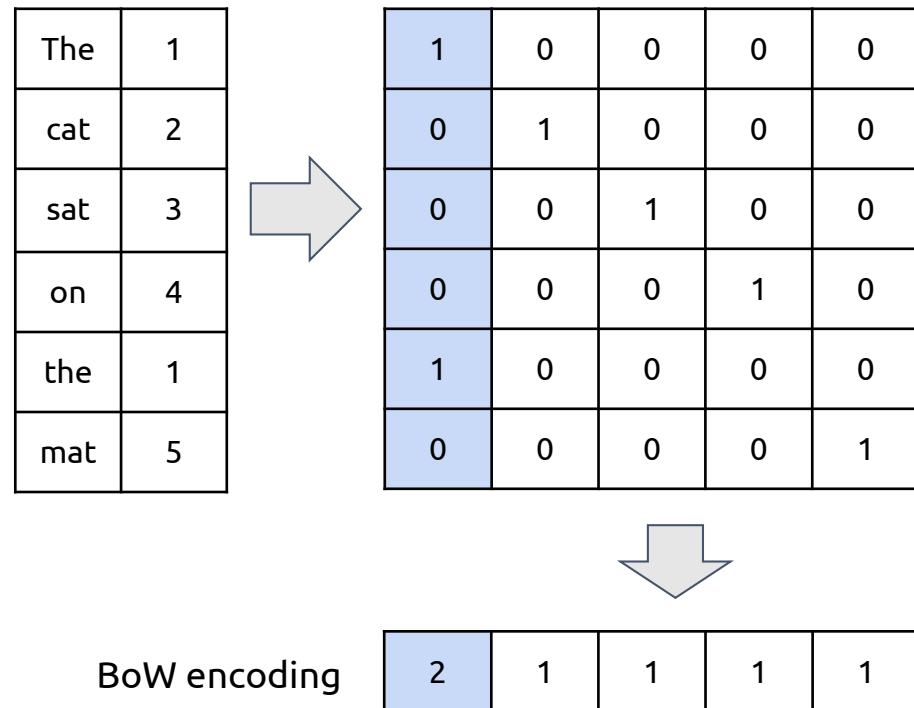
One way we represent an entire passage of text is called the bag of words model.

Add up the rows of the one-hot encodings.

Note: throws away all information about the ordering of words!

This representation is widely used in search engines and text analysis.

Note: may be useful to exclude uninteresting "stop words" like *a*, *the*, *an*, etc., from vocabulary



Why is this useful?

Summary statistics

- Most used words
- Least used words
- Average use of a word

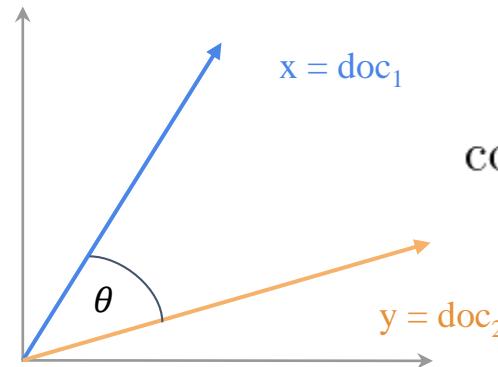
Information retrieval and search

- Encode query using BoW
- Encode all documents using BoW
- Compare query with docs using similarity metric (often cosine similarity)

Similarity

- Measure how similar documents are
- Cluster similar documents
- Topic modelling (PCA/LSI)
- Visualization (t-SNE)

Unsupervised Learning



$$\cos \theta = \frac{\mathbf{x}^T \mathbf{y}}{\|\mathbf{x}\| \|\mathbf{y}\|}$$

Multimedia data



Multimedia documents:

- Text, hypertext
- Images
- Video
- Audio

Types of analysis you might want to do:

- Natural language analysis
- Audio transcription, speaker identification
- Face detection, recognition
- Vehicle tracking
- Automatic image tagging
- ...

Fields of study:

- **Computer vision (CV)** is the field concerned with using computer models to understand visual content
- **Natural language processing (NLP)** is concerned with parsing, disambiguating, and understanding language
- **Automatic speech recognition (ASR)** is concerned with using computer models to transform speech to text.

Representing multimedia data



Many different representations used in practice.

Images:

- 3D tensors: $I \in \mathbb{R}^{H \times W \times 3}$
- As functions: $f(x, y) \rightarrow \mathbb{R}^3$
- As vectors: $I \in \mathbb{R}^{3WH}$
- Compressed (JPEG, PNG)
- Using automatically extracted features

Video: similar to images but with **time**. E.g.

$$f(x, y, t) \rightarrow \mathbb{R}^3$$

Digital audio:

- Sampled (44.1 KHz) quantized sound wave
- Single or multi channel
- Single channel sample of length N can be represented as a vector in \mathbb{R}^N
- Function representation
- Compressed (MP3)

Note: typically we do not store this kind of multimedia data in tables. **Why?**

Summary

Data

- metadata, semantics
- measurements, attributes, values

Stevens' four scales of measurement

- Nominal
- Ordinal
- Interval
- Ratio

Define what operations make sense

Number of variables/attributes

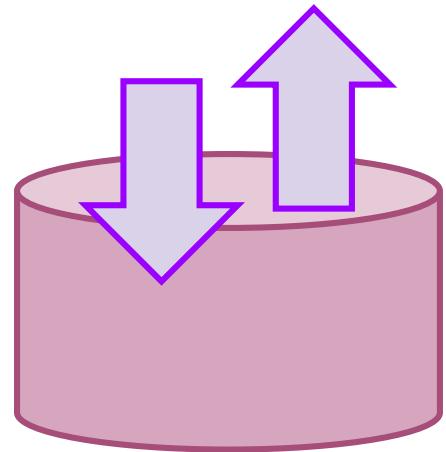
- Univariate
- Bivariate
- Multivariate

Interpreting data as vectors

Datasets as matrices

One-hot encoding

Multimedia



Data storage and I/O

Data storage and I/O

Data can come from files, databases, to be read from streams

Files:

- Can be structured or unstructured
- Good for distribution
- Can be high performance

Databases:

- Good for centralized information and network access
- Can enforce structure via **schemas**
- Multiple readers and writers
- Query languages to filter and search

Streams:

- Real-time processing of live data (e.g. twitter)

Data can be structured, unstructured, or semi-structured

Structured data: includes information on semantics such as relationships and data types. E.g:

- Tables
- Graphs
- Hierarchies
- Relational databases

Semi-structured and unstructured data: semantic information missing or not machine readable. E.g:

- Natural language plain text
- HTML files
- Word docs

File formats

Binary formats

Numeric values stored encoded in binary representation

- 32 bit float (4 bytes)
- 16 bit integer (2 bytes)

Properties:

- Compact
- High performance I/O
- Not human readable
- Need to worry about integer sizes, endiness, signed/unsigned

Plain text formats (ASCII and Unicode)

Numeric values encoded as ASCII or Unicode strings

- float -> "3.1415926"
- int -> "44"

Properties:

- Human readable
- (Somewhat) self-documenting
- Slower I/O
- Less compact

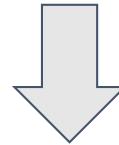
ASCII formats for tabular data

Tables are a common structure with special formats available.

Two very common formats:

- **CSV**: comma separated values
- **FWF**: fixed width format

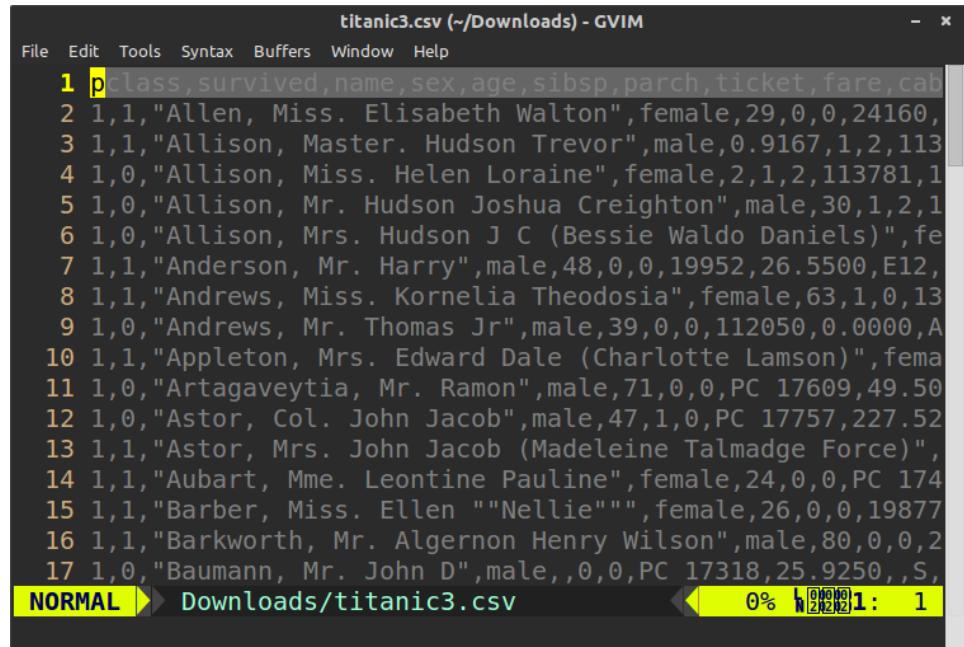
Preferred cola	Age							
	18 to 24	25 to 29	30 to 39	40 to 49	50 to 54	55 to 64	65 or more	
	%	%	%	%	%	%	%	%
Coca-Cola	65	41	55	28	46	36	36	36
Diet Coke	2	10	13	15	8	12	23	
Coke Zero	9	23	19	22	28	16	14	
Pepsi Light	0	3	0	3	3	6	9	
PepsiMax	16	18	6	10	13	24	14	
Pepsi	7	5	7	22	3	6	5	
NET	100	100	100	100	100	100	100	100



CSV

Comma separated values

- Plain text format
- Rows on lines
- Columns separated by commas
- Actually not only commas. Tabs, etc. (TSV)
- Strings containing commas are quoted



A screenshot of the GVIM text editor displaying the contents of the 'titanic3.csv' file. The window title is 'titanic3.csv (~/Downloads) - GVIM'. The file contains 17 rows of comma-separated data. The first row is a header with the following columns: pclass, survived, name, sex, age, sibsp, parch, ticket, fare, cabin, embarked, and class. The data rows follow, with some entries containing commas and being enclosed in double quotes. The status bar at the bottom shows 'NORMAL > Downloads/titanic3.csv' and '0% [000]1: 1'.

pclass	survived	name	sex	age	sibsp	parch	ticket	fare	cabin	embarked	class
1,1,"Allen, Miss. Elisabeth Walton",female,29,0,0,24160,											
3,1,1,"Allison, Master. Hudson Trevor",male,0.9167,1,2,113,											
4,1,0,"Allison, Miss. Helen Loraine",female,2,1,2,113781,1											
5,1,0,"Allison, Mr. Hudson Joshua Creighton",male,30,1,2,1											
6,1,0,"Allison, Mrs. Hudson J C (Bessie Waldo Daniels)",fe											
7,1,1,"Anderson, Mr. Harry",male,48,0,0,19952,26.5500,E12,											
8,1,1,"Andrews, Miss. Kornelia Theodosia",female,63,1,0,13											
9,1,0,"Andrews, Mr. Thomas Jr",male,39,0,0,112050,0.0000,A											
10,1,1,"Appleton, Mrs. Edward Dale (Charlotte Lamson)",fema											
11,1,0,"Artagaveytia, Mr. Ramon",male,71,0,0,PC 17609,49.50											
12,1,0,"Astor, Col. John Jacob",male,47,1,0,PC 17757,227.52											
13,1,1,"Astor, Mrs. John Jacob (Madeleine Talmadge Force)",											
14,1,1,"Aubart, Mme. Leontine Pauline",female,24,0,0,PC 174											
15,1,1,"Barber, Miss. Ellen ""Nellie""",female,26,0,0,19877											
16,1,1,"Barkworth, Mr. Algernon Henry Wilson",male,80,0,0,2											
17,1,0,"Baumann, Mr. John D",male,,0,0,PC 17318,25.9250,,S											

CSV

Advantages:

- Can be read by Excel, OpenOffice, Google Sheets, etc.
- Fast to parse/generate
- Can be compressed (.csv.gz)
- Do not need to load all into memory (streamable)

Disadvantages:

- Not standardised: many variations
- Bulkier than binary formats esp. when uncompressed
- No types

I/O in Python:

Python has a [built in library for parsing CSV](#)

```
import csv

with open('titanic3.csv') as f:
    reader = csv.reader(f)
    for row in reader:
        print(row[0], row[3])

with open('output.csv', 'w') as f:
    writer = csv.writer(f)
    writer.writerow(['A', 1, 'female', 'red'])
```

[Pandas](#) can also read/write CSV

```
import pandas as pd
df = pd.read_csv('titanic3.csv')
df.to_csv('output.csv')
```

FWF

Fixed width format

- Plain text format
- Rows on lines
- Columns of fixed width (fixed number of characters)
- Columns padded using padding character (usually spaces)

output.fwf (~/Downloads) - GVIM1

52	1	1	Cardenza, Mrs. James Warburton Martinez (Charlotte Wardle Drake)	female	58	0	1	PC 17755	512.329	B5			
53	1	0	Carlsson, Mr. Frans Olof	male	33	0	0	695	5	B5			
54	1	0	Carrau, Mr. Francisco M	male	28	0	0	113059	47.1	na			
55	1	0	Carrau, Mr. Jose Pedro	male	17	0	0	113059	47.1	na			
56	1	1	Carter, Master. William Thornton II	male	11	1	2	113760	120	B9			
57	1	1	Carter, Miss. Lucile Polk	female	14	1	2	113760	120	B9			
58	1	1	Carter, Mr. William Ernest	male	36	1	2	113760	120	B9			
59	1	1	Carter, Mrs. William Ernest (Lucile Polk)	female	36	1	2	113760	120	B9			
60	1	0	Case, Mr. Howard Brown	male	49	0	0	19924	26	na			
61	1	1	Cassebeer, Mrs. Henry Arthur Jr (Eleanor Genevieve Fosdick)	female	nan	0	0	17770	27.7208	na			
62	1	0	Cavendish, Mr. Tyrell William	male	36	1	0	19877	78.85	C4			
63	1	1	Cavendish, Mrs. Tyrell William (Julia Florence Siegel)	female	76	1	0	19877	78.85	C4			
64	1	0	Chaffee, Mr. Herbert Fuller	male	46	1	0	W.E.P. 5734	61.175	E3			
65	1	1	Chaffee, Mrs. Herbert Fuller (Carrie Constance Toogood)	female	47	1	0	W.E.P. 5734	61.175	E3			
66	1	1	Chambers, Mr. Norman Campbell	male	27	1	0	113806	53.1	E8			
67	1	1	Chambers, Mrs. Norman Campbell (Bertha Griggs)	female	33	1	0	113806	53.1	E8			
68	1	1	Chaudanson, Miss. Victorine	female	36	0	0	PC 17608	262.375	B6			
69	1	1	Cherry, Miss. Gladys	female	30	0	0	110152	86.5	B7			
70	1	1	Chevre, Mr. Paul Romaine	male	45	0	0	PC 17594	29.7	A9			

NORMAL ▶ Downloads/output.fwf utf-8[unix] 4% ↻ 62: 74 37

FWF

Advantages:

- Easier to read in plain text than CSV
- Can be read by major spreadsheet programs
- Fast to parse
- Can be compressed (.fwf.gz)
- Do not need to load all into memory (streamable)

Disadvantages:

- Not standardised
- Bulkier than CSV (padding characters)
- Need to establish field width before you can write first row
- No types

I/O in Python:

Pandas can also read FWF

```
import pandas as pd  
df = pd.read_fwf('titanic3.fwf')
```

To write, you need the **tabulate** package

```
from tabulate import tabulate  
content = tabulate(  
    df.values.tolist(),  
    list(df.columns),  
    tablefmt="plain")  
  
open('output.fwf', 'w').write(content)
```

Binary formats for tabular data

Binary formats are more compact and performant.

Not human readable.

Common binary formats:

- [HDF5](#)
- MATLAB
- NumPy
- Apache Arrow and Feather
- Excel

```
00000000 0000 0001 0001 1010 0010 0001 0004 0128
00000010 0000 0016 0000 0028 0000 0010 0000 0020
00000020 0000 0001 0004 0000 0000 0000 0000 0000
00000030 0000 0000 0000 0010 0000 0000 0000 0204
00000040 0004 8384 0084 c7c8 00c8 4748 0048 e8e9
00000050 00e9 6a69 0069 a8a9 00a9 2828 0028 fdfe
00000060 00fc 1819 0019 9898 0098 d9d8 00d8 5857
00000070 0057 7b7a 007a bab9 00b9 3a3c 003c 8888
00000080 8888 8888 8888 8888 288e be88 8888 8888
00000090 3b83 5788 8888 8888 7667 778e 8828 8888
000000a0 d61f 7abd 8818 8888 467c 585f 8814 8188
000000b0 8b06 e8f7 88aa 8388 8b3b 88f3 88bd e988
000000c0 8a18 880c e841 c988 b328 6871 688e 958b
000000d0 a948 5862 5884 7e81 3788 1ab4 5a84 3eec
000000e0 3d86 dcbb 5cbb 8888 8888 8888 8888 8888
000000f0 8888 8888 8888 8888 8888 8888 8888 0000
00001000 0000 0000 0000 0000 0000 0000 0000 0000
*
00001300 0000 0000 0000 0000 0000 0000 0000 0000
000013e0
```

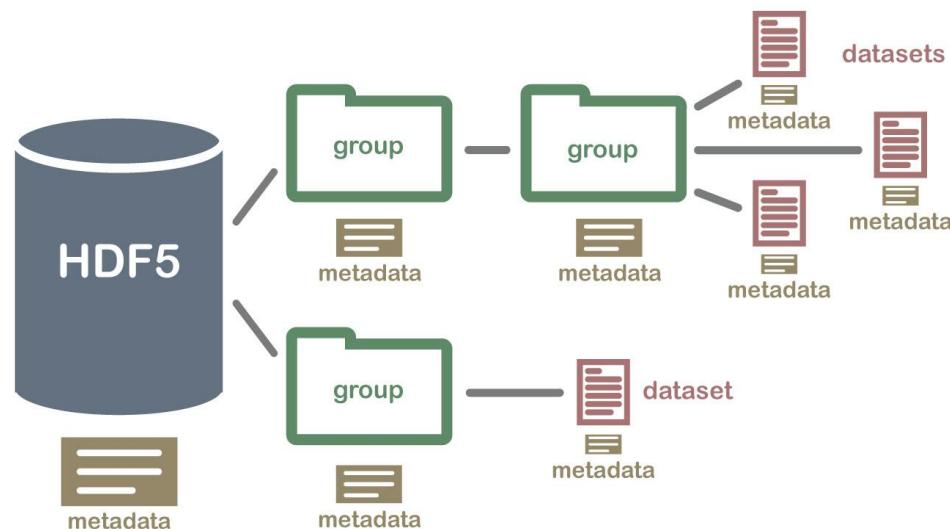


HDF5

Hierarchical data format version 5 (.h5)

Industry standard for storing numeric structured tabular data. Widely used in scientific community.

- Can store **multiple datasets** in a single file.
- Organize datasets into a **hierarchical structure**, just like a filesystem
- Include arbitrary metadata in the file using **metadata "attributes"**
- Uses **memory mapping** so entire dataset does not need to be loaded into memory. Memory can be **shared between processes**.
- Library support in many languages.





HDF5

Advantages:

- High performance
- Compact
- Multiple datasets in one file
- Type information
- Metadata
- Parallel reads, shared memory
- Optional compression

Disadvantages:

- Not human readable
- Can be cumbersome when table size is not known in advance

I/O in Python:

Can use [h5py](#) library

```
import h5py
f = h5py.File("mytestfile.hdf5", "w")
dset = f.create_dataset(
    "mydataset", shape=(100,), dtype='i')
```

Pandas can also read certain types of .h5 files
(using [PyTables](#))

```
pd.read_hdf('store_t1.h5', 'table')
df.to_hdf('store_t1.h5', 'table', append=True)

store = pd.HDFStore('store.h5')
store['mydataset']
```

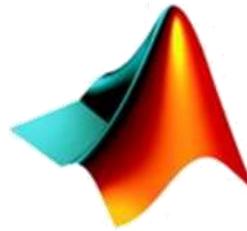
MATLAB files

Proprietary format used by [MATLAB](#) software (.mat)

Common for data exchange in industry and academia.

Can store multiple named arrays and various other structures.

Newer versions of MATLAB (R2006b or later) now store data in HDF5-based files (.mat v7.3)



MATLAB®

I/O in Python:

Scipy library includes [functions](#) for loading and saving .mat files in the `scipy.io` package

- `scipy.io.loadmat`
- `scipy.io.savemat`
- `scipy.io.whosmat`

NumPy

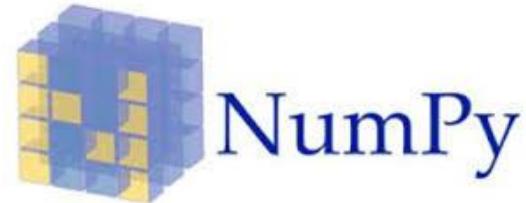
NumPy has built in support for two lightweight binary formats for storing NumPy arrays:

- `.npy` files contain single numpy arrays
- `.npz` files contain multiple arrays

`.npz` files are actually zipped archives of `.npy` files. Arrays in `.npz` files can be named. The format also supports compression.

Advantages and Disadvantages

- Fast, compact
- Supports memory mapping
- Built in support in NumPy
- No metadata
- Not standardized, less portable than H5



I/O in Python:

`load(file[, mmap_mode, allow_pickle, ...])`
Load arrays or pickled objects from `.npy`, `.npz` or pickled files.

`save(file, arr[, allow_pickle, fix_imports])`
Save an array to a binary file in NumPy `.npy` format.

`savez(file, *args, **kwds)`
Save several arrays into a single file in uncompressed `.npz` format.

`savez_compressed(file, *args, **kwds)`
Save several arrays into a single file in compressed `.npz` format.

Excel format

Very common proprietary format used in Microsoft Excel

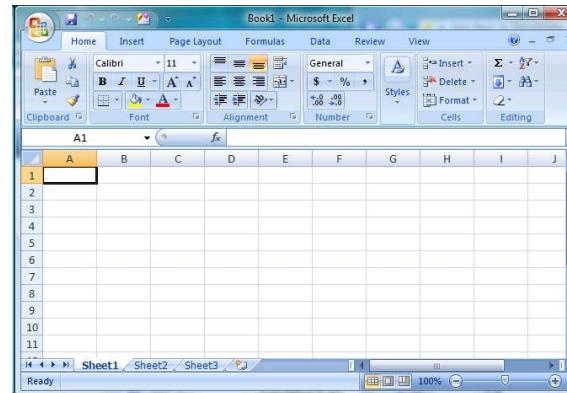
- XLS files, XLSX files

Advantages:

- Works with MS products
- Keeps “formatting” (colors, etc.)

Disadvantages:

- Keeps “formatting”
- Not portable
- Proprietary



I/O in Python:

Pandas has a function to read (`read_excel`) and method to write (`DataFrame.to_excel`)

Also see <http://www.python-excel.org/> for info on other libraries for working with Excel

- `xlrd`, `xlwt`

Common formats for structured data

More complex structures than plain tables (although can be reduced to tables)

Structure: encode data types, attributes, relationships, hierarchies

Common **plain-text formats** for structured data:

- [XML](#)
- [JSON](#)
- [YAML](#)



Binary formats:

- [MessagePack](#)
- [Google Protobuf](#)



XML

Extensible Markup Language ([XML](#))

Format for structured data designed to be both human and machine readable

Designed to be a self-describing format: no additional documentation needed

XML document is a hierarchy made up of:

- **Elements:** <book>
- **Attributes:** ISBN="0553212419"
- **Content:** Sherlock Holmes...

```
<Books>
  <Book ISBN="0553212419">
    <title>Sherlock Holmes: Complete Novels...
    <author>Sir Arthur Conan Doyle</author>
  </Book>
  <Book ISBN="0743273567">
    <title>The Great Gatsby</title>
    <author>F. Scott Fitzgerald</author>
  </Book>
  <Book ISBN="0684826976">
    <title>Undaunted Courage</title>
    <author>Stephen E. Ambrose</author>
  </Book>
  <Book ISBN="0743203178">
    <title>Nothing Like It In the World</title>
    <author>Stephen E. Ambrose</author>
  </Book>
</Books>
```



XML

Advantages:

- Self-describing
- "Human readable" (ish)

Disadvantages:

- High overhead (extremely bulky)
- Verbose
- Slow parsers
- Not well suited to big data and large scientific datasets
- XML namespaces are a nightmare
- Separate validation and DTD
- Painful to type manually
- No built-in types for scientific data

I/O in Python:

ElementTree in standard library. Stores entire XML structure in memory

```
import xml.etree.ElementTree as ET
tree = ET.parse('input.xml')
root = tree.getroot()
for child in root:
    print(child.tag, child.attrib)
```

Several other ways in Python

- SAX and Expat: event-driven parsing (callbacks)

See: <https://docs.python.org/2/library/xml.html>



JSON

JavaScript Object Notation

Subset of JavaScript for describing data

Value types:

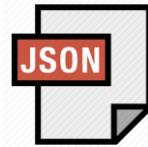
- String
- Number
- Boolean
- Object
- Array
- Null

Type is implicit in syntax

- 35 is a number
- "35" is a string
- { "a" : 1 } is an object
- [1, 2, 3] is an array

```
{  
  "employees": [  
    {"firstName": "John", "lastName": "Doe"},  
    {"firstName": "Anna", "lastName": "Smith"},  
    {"firstName": "Peter", "lastName": "Jones"}  
  ]  
}
```

JSON



Advantages:

- Has types
- Hierarchical
- Human-readable
- Self-describing
- Easy to write by hand
- More compact than XML
- Faster to parse than XML
- Most languages have parsers

Disadvantages:

- Some overhead (plain text)
- Not well suited to big data and large scientific datasets

I/O in Python:

Super simple using built-in json module

```
import json
data = json.load(open('input.json'))
with open('output.json', 'w') as f:
    json.dump(data, f)
```

JSON now used in place of XML in many applications, especially web apps.

YAML

YAML Ain't Markup Language!

- Like JSON, but easier to write by hand.
- Very useful for configuration files and metadata files.
- Slower than JSON

I/O in Python: Using pyyaml

```
import yaml  
data = yaml.safe_load(open('input.yaml'))
```

YAML: YAML Ain't Markup Language

What It Is: YAML is a human friendly data serialization standard for all programming languages.

YAML Resources:

YAML 1.2 (3rd Edition): <http://yaml.org/spec/1.2/spec.html>
YAML 1.1 (2nd Edition): <http://yaml.org/spec/1.1/>
YAML 1.0 (1st Edition): <http://yaml.org/spec/1.0/>
YAML Issues Page: <https://github.com/yaml/yaml/issues>
YAML Mailing List: yaml-core@lists.sourceforge.net
YAML IRC Channel: "#yaml on irc.freenode.net"
YAML Cookbook (Ruby): <http://yaml4r.sourceforge.net/cookbook/> ([local](#))
YAML Reference Parser: <http://ben-kiki.org/ypaste/>

Projects:

C/C++ Libraries:

- [libyaml](#) # "C" Fast YAML 1.1
- [Syck](#) # (dated) "C" YAML 1.0
- [yaml-cpp](#) # C++ YAML 1.2 implementation

Python:

- [PyYAML](#) # YAML 1.1, pure python and libyaml binding
- [ruamel.yaml](#) # YAML 1.2, update of PyYAML with ...

MessagePack

[MessagePack](#): binary encoded JSON

Much more compact than JSON

Advantages:

- Very compact
- High I/O performance
- Good for network I/O and databases
- Good for bigger data
- Streamable

Disadvantages:

- Not human readable (but easy to decode)
- Less suited to large numerical arrays

MessagePack

I/O in Python:

[Msgpack](#) library

```
>>> import msgpack  
>>> msgpack.packb([1, 2, 3])  
'\x93\x01\x02\x03'  
>>> msgpack.unpackb(_)  
[1, 2, 3]
```



Google Protobuf

"[Protocol buffers](#) are Google's language-neutral, platform-neutral, extensible mechanism for serializing structured data – think XML, but smaller, faster, and simpler."

Primarily designed for serializing data over the wire, but can also be used as a file format

Protobuf defines a protocol specification language (prototxt) and a wire format.

Notably used for model specification and distribution by the well-known deep learning library developed by Berkeley: [Caffe](#)

Prototxt example:

```
message Person {  
    required string name = 1;  
    required int32 id = 2;  
    optional string email = 3;  
}
```

I/O in Python:

Use Google's protoc compiler to generate "header" files. Then load and save with

```
ParseFromString(data)  
SerializeToString()
```

Things to consider when choosing a data format

- I/O performance
- Structure support (hierarchies, graphs, etc.)
- Streamable
- Scalability
- Appendable
- Portability
- Compactness
- Metadata
- Type support
- Readability
- Write by hand

Databases

Databases are also common for data acquisition and storage.

Advantages of databases include network access, concurrency, enforced consistency, fast indexes, query languages.

Not necessarily efficient for large binary data (e.g. images, video, audio, sensor data, astro).

Two important types:

- **Relational databases (SQL)**
- **NoSQL databases**



Summary of data formats and I/O

Files:

- Plain text formats
- Binary formats
- Structured or unstructured
- Semantics and metadata
- Types

Databases:

- SQL
- NoSQL

Files are good for:

- Data distribution
- Ephemeral or intermediate results
- High-speed processing
- Multimedia data

Databases are good for:

- Centralized network data access
- Concurrent access
- Enforcing consistency
- Subsetting and querying
- (Caching)



Data wrangling

Real-world datasets

Real world datasets are often “dirty”: messy, complicated, inconsistent

Sources of problems:

- Use (or lack of) incomplete standards
- Manual entry errors (typos)
- Measurement errors (equipment and noise)
- Inconsistent notations (naming)
- Redundancies and duplicates
- Missing values (NAs)

Section1All_csv.csv - LibreOffice Calc

File Edit View Insert Format Tools Data Window Help

A	B	C	D	E	F	G	H	I	J	K
1	Table 1.1.1. Percent Change From Preceding Period in Real Gross Domestic Product									
2	[Percent]									
3	Annual data from 1969 To 2015									
4	Bureau of Economic Analysis									
5	Data published September 29, 2016									
6	File created 9/28/2016 11:41:15 AM									
7										
8	Line		1969	1970	1971	1972	1973	1974	1975	19
9	1 Gross domestic product	A191RL1	3.1	0.2	3.3	5.2	5.6	-0.5	-0.2	5
10	2 Personal consumption expenditures	DPCERL1	3.7	2.4	3.8	6.1	5	-0.8	2.3	5
11	3 Goods	DGDSRL1	3.1	0.8	4.2	6.5	5.2	-3.6	0.7	
12	4 Durable goods	DDURRL1	3.7	-2.7	10	12.4	10.5	-6.4	0.2	12
13	5 Nondurable goods	DNDGRL1	2.8	2.2	1.9	4	2.9	-2.4	0.9	4
14	6 Services	DSERRL1	4.4	3.9	3.5	5.8	4.7	1.9	3.8	4
15	7 Gross private domestic investment	A006RL1	5.6	-6.1	10.3	11.3	10.9	-6.6	-16.2	19
16	8 Fixed investment	A007RL1	5.9	-2.1	6.9	11.4	8.6	-5.6	-9.8	9
17	9 Nonresidential	A008RL1	7	-0.9	0	8.7	13.2	0.8	-9	5
18	10 Structures	A009RL1	5.4	0.3	-1.6	3.1	8.2	-2.2	-10.5	2
19	11 Equipment	Y033RL1	8.3	-1.8	0.8	12.7	18.5	2.1	-10.5	6
20	12 Intellectual property products	Y001RL1	5.4	-0.1	0.4	7	5	2.9	0.9	10
21	13 Residential	A011RL1	3.1	-5.2	26.6	17.4	-0.6	-19.6	-12.1	22
22	14 Change in private inventories	ZZZZZ1
23	15 Net exports of goods and services	ZZZZZ1
24	16 Exports	A020RL1	4.9	10.7	1.7	7.8	18.8	7.9	-0.6	4
25	17 Goods	A253RL1	5.2	11.2	-0.1	10.9	24.5	8.5	-2.1	5
26	18 Services	A646RL1	3.9	9	7.3	-0.4	1.7	5.4	5.9	1
27	19 Imports	A021RL1	5.7	4.3	5.3	11.3	4.6	-2.3	-11.1	19
28	20 Goods	A255RL1	5.5	3.9	8.4	13.6	7.1	-2.8	-12.6	22
29	21 Services	A656RL1	6.3	5.2	-2.8	4.2	-3.4	-0.1	-4.3	58

1 Table 1.1.1. Percent Change From Preceding Period in Real Gross Domestic Product,,,

2 [Percent],.....

3 Annual data from 1969 To 2015,,,

4 Bureau of Economic Analysis,,,

5 "Data published September 29, 2016",

6 File created 9/28/2016 11:41:15 AM,

7

8 Line,,,1969,1970,1971,1972,1973,1974,1975,1976,1977,1978,1979,1980,1981,1982,1983,1984,1985,1986,1987,1988,1989,1990,1991,1992,1993,19

9 1, Gross domestic product,A191RL1,3.1,0.2,3.3,5.2,5.6,-0.5,-0.2,5.4,4.6,5.6,3.2,-0.2,2.6,-1.9,4.6,7.3,4.2,3.5,3.5,4.2,3.7,1.9,-0.1,

10 2,Personal consumption expenditures,DPCERL1,3.7,2.4,3.8,6.1,5.7,5.3,5.3,4.2,3.4,4.2,2.9,2.1,

11 3, Goods,DGDSRL1,3.1,0.8,4.2,6.5,5.2,-3.6,0.7,7.0,4.3,4.1,0.6,-2.0,3.2,4.2,5.3,3.0,4.5,4

12 4, Durable goods,DDURRL1,3.7,-2.7,10.0,12.4,10.5,-6.4,0,0.0,9.6,2.0,5.7,2.2,-0.4,-5.4,

13 5, Nondurable goods,DNDGRL1,2.8,2.2,1.9,4.0,2.9,-2.4,0.9,1.7,2.6,2.7,1.2,-0.3,1.9,2.5,

14 6, Services,DSERRL1,4.4,3.9,3.5,5.8,4.7,1.9,3.8,4.3,4.1,4.6,3.2,3.0,1.6,4.0,3.1,3.1,3.0,2.9,3

15 7,Gross private domestic investment,A006RL1,5.6,-6.1,10.3,11.3,10.9,-6.6,-16.2,19.1,14.3,11.6,3.5,-10.1,8.8,-13.0,9.3,27.3,-0.1,0.2,2.

16 8, Fixed investment,A007RL1,5.9,-2.1,6.9,11.4,8.6,-5.6,-9.8,9.8,13.6,11.6,5.8,-5.9,2.7,-6.7,7.5,16.2,5.5,1.8,0.6,3.3,3.2,-1.4,-5.1,5.

17 9, Nonresidential,A008RL1,7.0,-0.9,0.0,8.7,13.2,0.8,-9.0,5.7,10.8,13.8,10.0,0.0,6.1,-3.6,-0.4,16.7,6.6,-1.7,0.1,5.0,5.7,1.1,-3.9,2.

18 10, Structures,A009RL1,5.4,0.3,-1.6,3.1,8.2,-2.2,-10.5,2.4,4.1,14.4,12.7,5.9,8.0,-1.6,-10.8,13.9,7.1,-11.0,-2.9,0.7,2.0,1.5,-11.1

19 11, Equipment,Y033RL1,8.3,-1.8,0.8,12.7,18.5,2.1,-10.5,6.1,15.5,15.1,8.2,-4.4,3.7,-7.6,4.6,19.4,5.5,1.1,0.4,6.6,5.3,-2.1,-4.6,5.9

20 12, Intellectual property products,Y001RL1,5.4,-0.1,0.4,7.0,5.0,2.9,0.9,10.9,6.6,7.1,11.7,5.0,10.9,6.2,7.9,13.7,9.0,7.0,3.9,7.1,1

21 13, Residential,A011RL1,3.1,-5.2,26.6,17.4,-0.6,-19.6,-12.1,22.1,20.5,6.7,-3.7,-20.9,-8.2,-18.1,42.0,14.8,2.3,12.4,2.0,-0.9,-3.2,-8

22 14, Change in private inventories,ZZZZZZ1,.....

23 15,Net exports of goods and services,ZZZZZZ1,.....

24 16, Exports,A020RL1,4.9,10.7,1.7,7.8,18.8,7.9,-0.6,4.4,2.4,10.5,9.9,10.8,1.2,-7.6,-2.6,8.2,3.3,7.7,10.9,16.2,11.6,8.8,6.6,6.9,3.3,8.8

25 17, Goods,A253RL1,5.2,11.2,-0.1,10.9,24.5,8.5,-2.1,5.1,1.9,10.4,10.6,12.3,-0.6,-8.5,-3.2,7.1,3.5,5.4,12.2,17.8,11.4,8.6,6.7,7.5,3.2

26 18, Services,A646RL1,3.9,9.0,7.3,-0.4,1.7,5.4,5.9,1.1,4.5,11.2,7.1,4.2,9.8,-4.4,-0.2,11.8,2.8,14.4,7.6,11.9,12.0,9.5,6.4,5.4,3.3,7.

27 19, Imports,A021RL1,5.7,4.3,5.3,11.3,4.6,-2.3,-11.1,19.5,10.9,8.7,1.7,-6.6,2.6,-1.3,12.6,24.3,6.5,8.5,5.9,3.9,4.4,3.6,-0.1,7.0,8.6,11

28 20, Goods,A255RL1,5.5,3.9,8.4,13.6,7.1,-2.8,-12.6,22.6,12.2,9.0,1.7,-7.4,2.1,-2.5,13.6,24.2,6.3,10.3,4.6,4.1,4.3,2.9,0.5,9.4,10.0,1

29 21, Services,A656RL1,6.3,5.2,-2.8,4.2,-3.4,-0.1,-4.3,6.9,5.0,7.1,1.4,-2.2,5.9,5.3,8.1,25.1,7.6,1.1,11.8,3.4,4.8,6.5,-2.6,-2.7,2.7,5

30 22,Government consumption expenditures and gross investment,A822RL1,0.2,-2.0,-1.8,-0.5,-0.3,2.3,2.2,0.5,1.2,2.9,1.9,1.9,1.0,1.8,3.8,3.

31 23, Federal,A823RL1,-2.4,-6.1,-6.4,-3.1,-3.6,0.7,0.5,0.2,2.2,2.5,2.3,4.4,4.5,3.7,6.5,3.3,7.9,5.9,3.8,-1.3,1.7,2.1,0.0,-1.5,-3.5,-3.5,

32 24, National defense,A824RL1,-4.1,-8.2,-10.2,-6.9,-5.1,-1.0,-1.0,-0.5,1.0,0.8,2.7,3.9,6.2,7.2,7.3,5.2,8.8,6.9,5.1,-0.2,-0.2,0.3,-1

33 25, Nondefense,A825RL1,3.9,1.0,5.6,7.2,0.2,4.6,3.9,1.6,4.7,6.0,1.7,5.4,1.0,-3.6,4.7,-1.4,5.7,3.1,0.2,-4.3,7.2,7.3,2.4,5.9,0.0,-0.8,

34 26, State and local,A829RL1,3.5,2.9,3.1,2.2,2.8,3.7,3.6,0.8,0.4,3.3,1.5,-0.2,-2.0,0.1,1.3,3.8,5.7,5.0,2.2,3.9,4.0,4.1,2.2,2.1,1.2,60

Headers need to be removed

1 Table 1.1.1. Percent Change From Preceding Period in Real Gross Domestic Product,,,
2 [Percent],.....
3 Annual data from 1969 To 2015.....
4 Bureau of Economic Analysis,.....
5 "Data published September 29, 2016",
6 File created 9/28/2016 11:41:15 AM,
7,
8 Line,,,1969,1970,1971,1972,1973,1974,1975,1976,1977,1978,1979,1980,1981,1982,1983,1984,1985,1986,1987,1988,1989,1990,1991,1992,1993,19
9 1, Gross domestic product,A191RL1,3.1,0.2,3.3,5.2,5.6,-0.5,-0.2,5.4,4.6,5.6,3.2,-0.2,2.6,-1.9,4.6,7.3,4.2,3.5,3.5,4.2,3.7,1.9,-0.1,
10 2,Personal consumption expenditures,DPCERL1,3.7,2.4,3.8,6.1,5,5.7,5.3,5.3,4.2,3.4,4.2,2.9,2.1,
11 3, Goods,DGDSRL1,3.1,0.8,4.2,6.5,5.2,-3.6,0.7,7.0,4.3,4.1,0.6,-2.0,3.2,4.2,5.3,3.0,4.5,4
12 4, Durable goods,DDURRL1,3.7,-2.7,10.0,12.4,10.5,-6.4,0,0.0,9.6,2.0,5.7,2.2,-0.4,-5.4,
13 5, Nondurable goods,DNDGRL1,2.8,2.2,1.9,4.0,2.9,-2.4,0.9,1.7,2.6,2.7,1.2,-0.3,1.9,2.5,
14 6, Services,DSERRL1,4.4,3.9,3.5,5.8,4.7,1.9,3.8,4.3,4.1,4.6,3.2,3.0,1.6,4.0,3.1,3.1,3.0,2.9,3
15 7,Gross private domestic investment,A006RL1,5.6,-6.1,10.3,11.3,10.9,-6.6,-16.2,19.1,14.3,11.6,3.5,-10.1,8.8,-13.0,9.3,27.3,-0.1,0.2,2.
16 8, Fixed investment,A007RL1,5.9,-2.1,6.9,11.4,8.6,-5.6,-9.8,9.8,13.6,11.6,5.8,-5.9,2.7,-6.7,7.5,16.2,5.5,1.8,0.6,3.3,3.2,-1.4,-5.1,5.
17 9, Nonresidential,A008RL1,7.0,-0.9,0.0,8.7,13.2,0.8,-9.0,5.7,10.8,13.8,10.0,0.0,6.1,-3.6,-0.4,16.7,6.6,-1.7,0.1,5.0,5.7,1.1,-3.9,2.
18 10, Structures,A009RL1,5.4,0.3,-1.6,3.1,8.2,-2.2,-10.5,2.4,4.1,14.4,12.7,5.9,8.0,-1.6,-10.8,13.9,7.1,-11.0,-2.9,0.7,2.0,1.5,-11.1
19 11, Equipment,Y033RL1,8.3,-1.8,0.8,12.7,18.5,2.1,-10.5,6.1,15.5,15.1,8.2,-4.4,3.7,-7.6,4.6,19.4,5.5,1.1,0.4,6.6,5.3,-2.1,-4.6,5.9
20 12, Intellectual property products,Y001RL1,5.4,-0.1,0.4,7.0,5.0,2.9,0.9,10.9,6.6,7.1,11.7,5.0,10.9,6.2,7.9,13.7,9.0,7.0,3.9,7.1,1
21 13, Residential,A011RL1,3.1,-5.2,26.6,17.4,-0.6,-19.6,-12.1,22.1,20.5,6.7,-3.7,-20.9,-8.2,-18.1,42.0,14.8,2.3,12.4,2.0,-0.9,-3.2,-8
22 14, Change in private inventories,ZZZZZZ1,.....
23 15,Net exports of goods and services,ZZZZZZ1,.....
24 16, Exports,A020RL1,4.9,10.7,1.7,7.8,18.8,7.9,-0.6,4.4,2.4,10.5,9.9,10.8,1.2,-7.6,-2.6,8.2,3.3,7.7,10.9,16.2,11.6,8.8,6.6,6.9,3.3,8.8
25 17, Goods,A253RL1,5.2,11.2,-0.1,10.9,24.5,8.5,-2.1,5.1,1.9,10.4,10.6,12.3,-0.6,-8.5,-3.2,7.1,3.5,5.4,12.2,17.8,11.4,8.6,6.7,7.5,3.2
26 18, Services,A646RL1,3.9,9.0,7.3,-0.4,1.7,5.4,5.9,1.1,4.5,11.2,7.1,4.2,9.8,-4.4,-0.2,11.8,2.8,14.4,7.6,11.9,12.0,9.5,6.4,5.4,3.3,7.
27 19, Imports,A021RL1,5.7,4.3,5.3,11.3,4.6,-2.3,-11.1,19.5,10.9,8.7,1.7,-6.6,2.6,-1.3,12.6,24.3,6.5,8.5,5.9,3.9,4.4,3.6,-0.1,7.0,8.6,11
28 20, Goods,A255RL1,5.5,3.9,8.4,13.6,7.1,-2.8,-12.6,22.6,12.2,9.0,1.7,-7.4,2.1,-2.5,13.6,24.2,6.3,10.3,4.6,4.1,4.3,2.9,0.5,9.4,10.0,1
29 21, Services,A656RL1,6.3,5.2,-2.8,4.2,-3.4,-0.1,-4.3,6.9,5.0,7.1,1.4,-2.2,5.9,5.3,8.1,25.1,7.6,1.1,11.8,3.4,4.8,6.5,-2.6,-2.7,2.7,5
30 22,Government consumption expenditures and gross investment,A822RL1,0.2,-2.0,-1.8,-0.5,-0.3,2.3,2.2,0.5,1.2,2.9,1.9,1.9,1.0,1.8,3.8,3.
31 23, Federal,A823RL1,-2.4,-6.1,-6.4,-3.1,-3.6,0.7,0.5,0.2,2.2,2.5,2.3,4.4,4.5,3.7,6.5,3.3,7.9,5.9,3.8,-1.3,1.7,2.1,0.0,-1.5,-3.5,-3.5,
32 24, National defense,A824RL1,-4.1,-8.2,-10.2,-6.9,-5.1,-1.0,-1.0,-0.5,1.0,0.8,2.7,3.9,6.2,7.2,7.3,5.2,8.8,6.9,5.1,-0.2,-0.2,0.3,-1.
33 25, Nondefense,A825RL1,3.9,1.0,5.6,7.2,0.2,4.6,3.9,1.6,4.7,6.0,1.7,5.4,1.0,-3.6,4.7,-1.4,5.7,3.1,0.2,-4.3,7.2,7.3,2.4,5.9,0.0,-0.8,
34 26, State and local,A829RL1,3.5,2.9,3.1,2.2,2.8,3.7,3.6,0.8,0.4,3.3,1.5,-0.2,-2.0,0.1,1.3,3.8,5.7,5.0,2.2,3.9,4.0,4.1,2.2,2.1,1.2,62

1 Table 1.1.1. Percent Change From Preceding Period in Real Gross Domestic Product,,,
2 [Percent],.....
3 Annual data from 1969 To 2015,,,
4 Bureau of Economic Analysis,,,
5 "Data published September 29, 2016",
6 File created 9/28/2016 11:41:15 AM,
7,1969,1970,1971,1972,1973,1974,1975,1976,1977,1978,1979,1980,1981,1982,1983,1984,1985,1986,1987,1988,1989,1990,1991,1992,1993,19
8 1, Gross domestic product,A191RL1,3.1,0.2,3.3,5.2,5.6,-0.5,-0.2,5.4,4.6,5.6,3.2,-0.2,2.6,-1.9,4.6,7.3,4.2,3.5,3.5,4.2,3.7,1.9,-0.1,
9 2,Personal consumption expenditures,DPCERL1,3.7,2.4,3.8,6.1,5.7,5.3,5.3,4.2,3.4,4.2,2.9,2.1,
10 3, Goods,DGDSRL1,3.1,0.8,4.2,6.5,5.2,-3.6,0.7,7.0,4.3,4.1,0.6,-2.0,3.2,4.2,5.3,3.0,4.5,4
11 4, Durable goods,DDURRL1,3.7,-2.7,10.0,12.4,10.5,-6.4,0,0.9,6.2,0.5,7.2,2,-0.4,-5.4,
12 5, Nondurable goods,DNDGRL1,2.8,2.2,1.9,4.0,2.9,-2.4,0.9,1.7,2.6,2.7,1.2,-0.3,1.9,2.5,
13 6, Services,DSERRL1,4.4,3.9,3.5,5.8,4.7,1.9,3.8,4.3,4.1,4.6,3.2,3.0,1.6,4.0,3.1,3.1,3.0,2.9,3
14 7,Gross private domestic investment,A006RL1,5.6,-6.1,10.3,11.3,10.9,-6.6,-16.2,19.1,14.3,11.6,3.5,-10.1,8.8,-13.0,9.3,27.3,-0.1,0.2,2.
15 8, Fixed investment,A007RL1,5.9,-2.1,6.9,11.4,8.6,-5.6,-9.8,9.8,13.6,11.6,5.8,-5.9,2.7,-6.7,7.5,16.2,5.5,1.8,0.6,3.3,3.2,-1.4,-5.1,5.
16 9, Nonresidential,A008RL1,7.0,-0.9,0.0,8.7,13.2,0.8,-9.0,5.7,10.8,13.8,10.0,0.0,6.1,-3.6,-0.4,16.7,6.6,-1.7,0.1,5.0,5.7,1.1,-3.9,2.
17 10, Structures,A009RL1,5.4,0.3,-1.6,3.1,8.2,-2.2,-10.5,2.4,4.1,14.4,12.7,5.9,8.0,-1.6,-10.8,13.9,7.1,-11.0,-2.9,0.7,2.0,1.5,-11.1
18 11, Equipment,Y033RL1,8.3,-1.8,0.8,12.7,18.5,2.1,-10.5,6.1,15.5,15.1,8.2,-4.4,3.7,-7.6,4.6,19.4,5.5,1.1,0.4,6.6,5.3,-2.1,-4.6,5.9
19 12, Intellectual property products,Y001RL1,5.4,-0.1,0.4,4.7,0.5,0.2,9.0,9.1,10.9,6.6,7.1,11.7,5.0,10.9,6.2,7.9,13.7,9.0,7.0,3.9,7.1,1
20 13, Residential,A011RL1,3.1,-5.2,26.6,17.4,-0.6,-19.6,-12.1,22.1,20.5,6.7,-3.7,-20.9,-8.2,-18.1,42.0,14.8,2.3,12.4,2.0,-0.9,-3.2,-8
21 14, Change in private inventories,ZZZZZZ1,.....
22 15,Net exports of goods and services,ZZZZZZ1,.....
23 16, Exports,A020RL1,4.9,10.7,1.7,7.8,18.8,7.9,-0.6,4.4,2.4,10.5,9.9,10.8,1.2,-7.6,-2.6,8.2,3.3,7.7,10.9,16.2,11.6,8.8,6.6,6.9,3.3,8.8
24 17, Goods,A253RL1,5.2,11.2,-0.1,10.9,24.5,8.5,-2.1,5.1,1.9,10.4,10.6,12.3,-0.6,-8.5,-3.2,7.1,3.5,5.4,12.2,17.8,11.4,8.6,6.7,7.5,3.2
25 18, Services,A646RL1,3.9,9.0,7.3,-0.4,1.7,5.4,5.9,1.1,4.5,11.2,7.1,4.2,9.8,-4.4,-0.2,11.8,2.8,14.4,7.6,11.9,12.0,9.5,6.4,5.4,3.3,7.
26 19, Imports,A021RL1,5.7,4.3,5.3,11.3,4.6,-2.3,-11.1,19.5,10.9,8.7,1.7,-6.6,2.6,-1.3,12.6,24.3,6.5,8.5,5.9,3.9,4.4,3.6,-0.1,7.0,8.6,11
27 20, Goods,A255RL1,5.5,3.9,8.4,13.6,7.1,-2.8,-12.6,22.6,12.2,9.0,1.7,-7.4,2.1,-2.5,13.6,24.2,6.3,10.3,4.6,4.1,4.3,2.9,0.5,9.4,10.0,1
28 21, Services,A656RL1,6.3,5.2,-2.8,4.2,-3.4,-0.1,-4.3,6.9,5.0,7.1,1.4,-2.2,5.9,5.3,8.1,25.1,7.6,1.1,11.8,3.4,4.8,6.5,-2.6,-2.7,2.7,5
29 22,Government consumption expenditures and gross investment,A822RL1,0.2,-2.0,-1.8,-0.5,-0.3,2.3,2.2,0.5,1.2,2.9,1.9,1.9,1.0,1.8,3.8,3.
30 23, Federal,A823RL1,-2.4,-6.1,-6.4,-3.1,-3.6,0.7,0.5,0.2,2.2,2.5,2.3,4.4,4.5,3.7,6.5,3.3,7.9,5.9,3.8,-1.3,1.7,2.1,0.0,-1.5,-3.5,-3.5,
31 24, National defense,A824RL1,-4.1,-8.2,-10.2,-6.9,-5.1,-1.0,-1.0,-0.5,1.0,0.8,2.7,3.9,6.2,7.2,7.3,5.2,8.8,6.9,5.1,-0.2,-0.2,0.3,-1.
32 25, Nondefense,A825RL1,3.9,1.0,5.6,7.2,0.2,4.6,3.9,1.6,4.7,6.0,1.7,5.4,1.0,-3.6,4.7,-1.4,5.7,3.1,0.2,-4.3,7.2,7.3,2.4,5.9,0.0,-0.8,
33 26, State and local,A829RL1,3.5,2.9,3.1,2.2,2.8,3.7,3.6,0.8,0.4,3.3,1.5,-0.2,-2.0,0.1,1.3,3.8,5.7,5.0,2.2,3.9,4.0,4.1,2.2,2.1,1.2,63

Data wrangling

Process of transforming “raw” data into data that can be analysed to generate actionable insights

AKA:

- Preprocessing
- Munging
- Cleaning
- Scrubbing
- Preparation
- Transformation

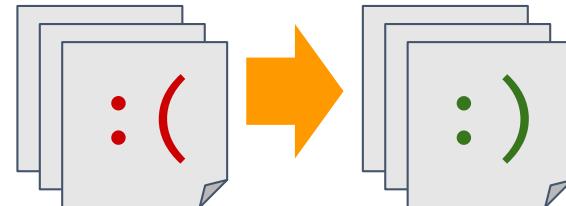
Typical data wrangling tasks

- Fixing ugly and broken formats
- Handling missing values
- Removing redundant attributes and records
- Fixing inconsistencies
- Shaping data
- Fusing data sources
- Scraping and gathering data from external sources
- Extracting information from unstructured sources

Ugly and broken formats

Examples:

- Badly formatted tables
- Broken XML/JSON (syntax errors)
- Hand entered data with syntax errors
- Log files with strange formatting



First step: transform to more machine friendly parsable format

Toolbox: Python [csv](#) module, text editor like [vim](#), custom parsing scripts, regular expressions ([re](#) module), [tabular](#), [pandas](#)

Missing values

Missing values are common and can have significant effects on analysis and conclusions

Causes:

- Non-response
- Unobserved or unknown values
- Sensor or measurement errors
- Censorship
- Errors in data collection or data entry

Often show up in datasets as:

- Special `NA` values
- `NAN`
- `null` or `None`
- Sentinel values (e.g `age == -1`)
- Blanks

GENDER	AGE	RELIG.	Q1
M	18	CR	...
M	-1	ATH	null
F	22	CR	...
F	36	N/A	...

Important to try understand the reasons for missing values in order to appropriately deal with them...

Missing values

Three types of missing values:

1. **Missing completely at random (MCAR):** missing values are randomly distributed for all observations
2. **Missing at random (MAR):** probability of value being missing depends on other observed variables
3. **Missing not at random (MNAR):** probability of value being missing depends on value of missing variable or another unobserved variable.

GENDER	AGE	RELIG.	Q1
M	18	CR	...
M	-1	ATH	null
F	22	CR	...
F	36	N/A	...

MCAR and MAR assumption is common.

If assumptions are made when dealing with missing values, make them explicit!

Strategies for handling missing values

Three common approaches:

1. Ignore
2. Remove
3. Impute

May introduce bias for MNAR values!

Ignore

Drop missing values when computing summary statistics (e.g. mean, variance).

Remove

If plenty of data is available, may be possible to simply ignore rows that have missing values

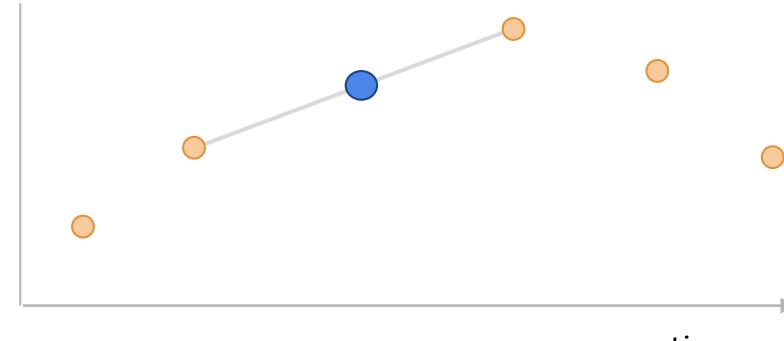
Impute

Try to “fill in the blanks”

Common imputation techniques:

- Mean/mode substitution
- Predict from other attributes
- Interpolation (e.g. time series)

Note: good idea to include **indicator variable** to state if value was imputed



Redundant attributes

For example:

- Useless attributes
- Duplicated attributes
- Attributes easily derived from other attributes

Can cause problems for some statistical analysis (e.g. regression models).

Eliminate redundancy where possible.

HUMAN	SEX	GENDER	HEIGHT (ft)	HEIGHT (cm)
Y	M	male	5'9	175
Y	M	male	6'4	193
Y	F	female	5'10	178
Y	F	female	6'1	185

Inconsistent categories (nominal attributes)

Ask 10 different people to do the same task and they will do it 11 different ways!

For example:

- Misspellings
- Inconsistent spellings
- Hyphenation
- Inconsistent case
- Inconsistent abbreviations

Techniques:

- Print unique vals and try to detect outliers and splits
- Normalize case and spelling

GENDER	STATE
Male	NY
male	New York
F	"New-york"
fem.	Califrnia, USA

Tools:

- Unix sort | uniq
- Python sort, set()
- str.lower, str.replace,
- Regular expressions (re)

Dates and times

Huge variation in ways dates, times, and timestamps can be represented.

Data cleaning should **standardize** to a single format.

Preferably include **timezone** information.

Standard plain text format: [ISO8601](#)

- 2016-10-10T16:04:07+00:00

	A	B
1	Sunday, January 03, 1988	
2		1/3/1988
3		1/3/88
4		01/03/88
5		3-Jan-88
6		03-Jan-88
7		Jan-88
8		January-88
9		January 3, 1988
10		1/3/1988
11		3-Jan-1988
12		

Parsing dates and times

Python standard library:

```
datetime.strptime(str, fmt)
```

The [parsedatetime](#) library:

Will parse almost anything!

```
import parsedatetime  
cal = parsedatetime.Calendar()  
cal.parse("tomorrow")
```

Also consider using [arrow](#) library if you do a lot of date and time manipulation

Input	Parse
19 November 1975	Wed Nov 19 08:41:38 1975
19 November 75	Wed Nov 19 08:41:38 1975
19 Nov 75	Wed Nov 19 08:41:38 1975
tomorrow	Tue Jun 21 09:00:00 2016
yesterday	Sun Jun 19 09:00:00 2016
10 minutes from now	Mon Jun 20 08:51:38 2016
the first of January, 2001	Mon Jan 1 08:41:38 2001
3 days ago	Fri Jun 17 08:41:38 2016

Outliers

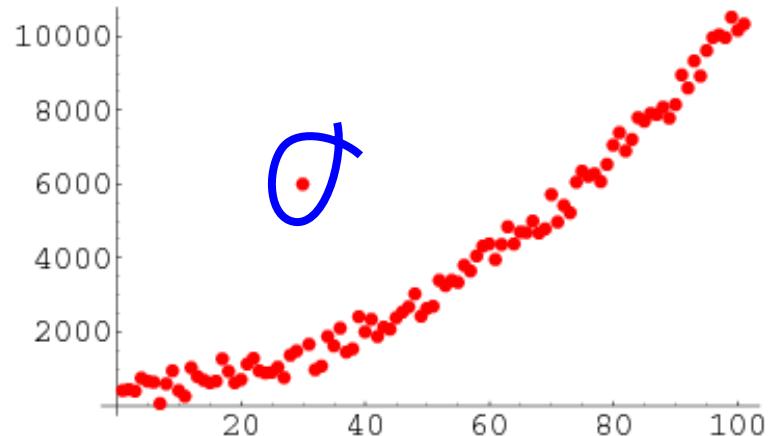
Data points that are extremely unlikely under the data distribution (far from other data points).

Causes:

- Measurement error
- Recording error
- Statistical anomalies (may be interesting!)

Often you'll want to identify outliers prior to further analysis.

- Measure quantity of outliers
- Label outliers
- Completely remove outliers

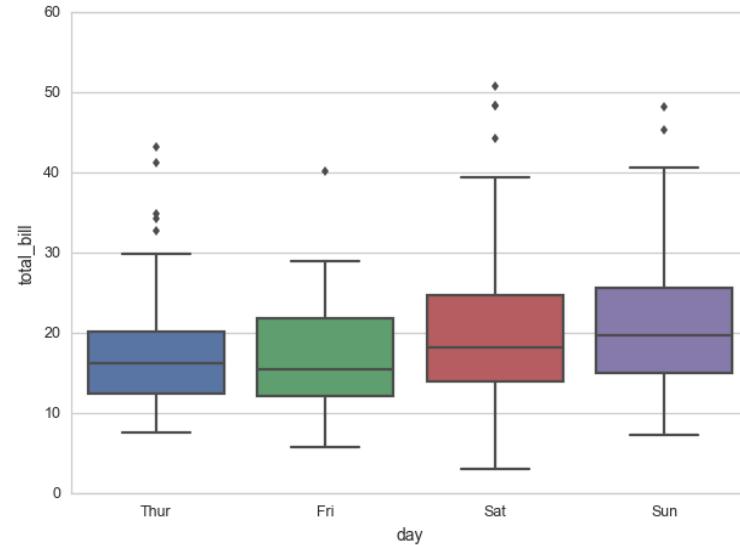


Detecting outliers

Often you can detect outliers by plotting your data and doing some visual inspection.

- Boxplots, jitter plots, histograms
- More details next lecture!

Alternatively, you can make some assumptions about the distribution of the attribute and find items that are unlikely under this distribution.



Detecting outliers

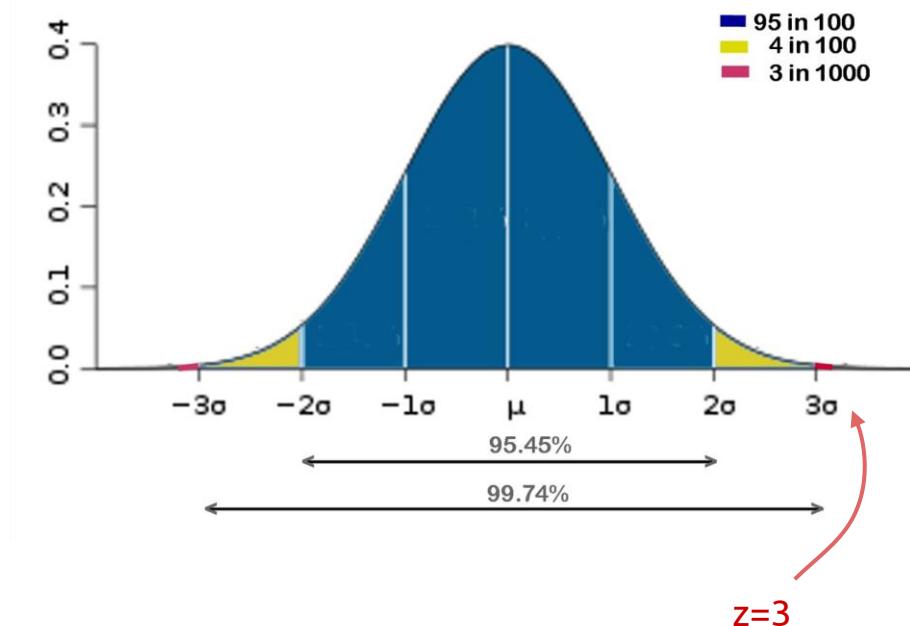
E.g. assume data is **normally (Gaussian) distributed.**

Estimate sample mean and standard deviation from data and compute Z scores.

$$z = \frac{x - \mu}{\sigma}$$

sample mean

sample standard deviation



Data shaping

Data often stored in "stacked" or record format

date	variable	value
2000-01-03	A	0.469112
2000-01-04	A	-0.282863
2000-01-05	A	-1.509059
2000-01-03	B	-1.135632
2000-01-04	B	1.212112
2000-01-05	B	-0.173215
2000-01-03	C	0.119209
2000-01-04	C	-1.044236
2000-01-05	C	-0.861849
2000-01-03	D	-2.104569
2000-01-04	D	-0.494929
2000-01-05	D	1.071804



Sometimes more convenient to have one "observation" per row with multiple attributes

date	A	B	C	D
2000-01-03	0.469112	-1.135632	0.119209	-2.104569
2000-01-04	-0.282863	1.212112	-1.044236	-0.494929
2000-01-05	-1.509059	-0.173215	-0.861849	1.071804

This can be achieved by reshaping or pivot operations. In pandas:

```
df.pivot(index='date', columns='variable',  
values='value')
```

Fusion of data sources

ID	GENDER	LOC	AGE
1	F	COR	32
2	F	DUB	25
3	F	DUB	19
4	M	LIM	43

ID	Q1	Q2	Q3
1	N	Y	1
2	Y	Y	3
3	Y	N	2
4	N	Y	1

JOIN ON ID

ID	GENDER	LOC	Q1	Q2	Q3
1	F	COR	N	Y	1
2	F	DUB	Y	Y	3
3	F	DUB	Y	N	2
4	M	LIM	N	Y	1

Fusion of data sources

Pandas

```
result = pd.merge(left, right, on='key')
```

Lots more options for merging and concatenation.

See [docs](#).

SQL Inner Joins

SELECT

```
    left.A, left.B, left.key,  
    right.C, right.D  
FROM left  
INNER JOIN right  
ON left.key = right.key;
```

	left			right		
	A	B	key	C	D	key
0	A0	B0	K0	C0	D0	K0
1	A1	B1	K1	C1	D1	K1
2	A2	B2	K2	C2	D2	K2
3	A3	B3	K3	C3	D3	K3

	Result				
	A	B	key	C	D
0	A0	B0	K0	C0	D0
1	A1	B1	K1	C1	D1
2	A2	B2	K2	C2	D2
3	A3	B3	K3	C3	D3

Dealing with unstructured data

So far we've talked mostly about cleaning structured data (even if structure is awful!)

What about semi-structured and unstructured data?

- Natural language plain text
- HTML files

Usually we'll want to **extract some features** from this type of data for analysis

We've already seen one way of encoding text features:
bag-of-words

```
links%2Cteahouse%7Cext_tmh.thumbnail.styles%7Cext_uls.interlanguage%7Cext_visualEditor_desk
opArticleTarget_noscript%7Cext_wikimediaBadges%7Cmediawiki_legacy.commonPrint%2Cshared%7Cme
iwiki_sectionAnchor%7Cmediawiki_skinning.interface%only=scripts&skin=vector%7Cskins.vector.styles%7Cwikibase.client
init&amp;style&skin=vector" />
12 <script async="" src="/w/load.php?debug=false&lang=en&modules=startup&only=scripts&skin=vector"></script>
13 <meta name="ResourceLoaderDynamicStyles" content="" />
14 <link rel="stylesheet" href="/w/load.php?debug=false&lang=en&modules=site.styles&only=styles&skin=vector"/>
15 <meta name="generator" content="MediaWiki 1.28.0-wmf.21"/>
16 <meta name="referrer" content="origin-when-cross-origin"/>
17 <link rel="alternate" href="android-
app://org.wikipedia/http/en.m.wikipedia.org/wiki/Web_scraping"/>
18 <link rel="alternate" type="application/x-wiki" title="Edit this page" href="/w/index.php?
title=Web_scraping&action=edit"/>
19 <link rel="edit" title="Edit this page" href="/w/index.php?
title=Web_scraping&action=edit"/>
20 <link rel="apple-touch-icon" href="/static/apple-touch/wikipedia.png"/>
21 <link rel="shortcut icon" href="/static/favicon/wikipedia.ico"/>
22 <link rel="search" type="application/opensearchdescription+xml"
href="/w/opensearch_desc.php" title="Wikipedia (en)"/>
23 <link rel="EditURI" type="application/rsd+xml" href="https://en.wikipedia.org/w/api.php?
action=rsd"/>
24 <link rel="copyright" href="https://creativecommons.org/licenses/by-sa/3.0/"/>
25 <link rel="canonical" href="https://en.wikipedia.org/wiki/Web_scraping"/>
26 <link rel="dns-prefetch" href="https://login.wikimedia.org"/>
27 <link rel="dns-prefetch" href="https://meta.wikimedia.org"/>
28 </head>
29 <body class="mediawiki ltr sitedir-ltr mw-hide-empty-elt ns-0 ns-subject page-Web_scraping
rootpage-Web_scraping skin-vector action-view feature-footer-v2"> <div id="mw-page-
base" class="noprint"></div>
30 <div id="mw-head-base" class="noprint"></div>
31 <div id="content" class="mw-body" role="main">
32 <a id="top"></a>
33
34 <div id="siteNotice"><!-- CentralNotice --></div>
35 <div class="mw-indicators">
36 </div>
37 <h1 id="firstHeading" class="firstHeading" lang="en">Web scraping</h1>
38 <div id="bodyContent" class="mw-body-content">
39 <div id="sitesub">From Wikipedia, the free
encyclopedia</div>
40 <div id="contentSub"></div>
41 <div id="jump-to-nav" class="mw-jump">
42 <a href="#mw-head">navigation</a>,<br/>
<a href="#p-search">search</a>
```

Web scraping: getting data from webpages

Idea: extract structured information from unstructured web pages by automatic downloading and parsing them.

Task	Issues	Toolbox
Getting the data	HTTP requests, cookies, headers, downloads, JavaScript, timing	wget , curl requests , mechanize , selenium
Figuring out which data to get	Link crawling and spidering	scrapy
Extracting structured information	Robust parsing, DOM querying	PyQuery , BeautifulSoup , lxml

Web scraping

wget, curl

Convenient command line tools.

```
$ curl https://curl.haxx.se >> page.html
```

requests

Easily make HTTP requests directly in Python

```
>>> r = requests.get(  
...     'https://api.github.com/user', auth=('user', 'pass'))  
>>> r.status_code  
200  
>>> r.headers['content-type']  
'application/json; charset=utf8'  
>>> r.text  
u'{"type": "User"...'  
>>> r.json()  
{u'private_gists': 419, u'total_private_repos': 77, ...}
```

mechanize

Stateful programmatic web browsing. Pretend to be a browser (cookies and headers and all)

```
browser = mechanize.Browser()  
browser.open("http://www.example.com/")  
response = br.follow_link(text_regex=r"cheese\s*shop", nr=1)  
print(browser.title())  
print(response.geturl())  
print(response.info()) # headers  
print(response.read()) # body
```

selenium

Remote control an actual browser!

```
driver = webdriver.Firefox()  
driver.get("http://www.python.org")  
elem = driver.find_element_by_name("q")  
elem.clear()  
elem.send_keys("pycon")  
elem.send_keys(Keys.RETURN)
```

PyQuery

Parsing and querying HTML with [PyQuery](#):

```
>>> from pyquery import PyQuery
>>> html = open("index.html", 'r').read()
>>> pq = PyQuery(html)

>>> pq("title").text()
'PyQuery Test!'

>>> pq("li").eq(1).text()
'DOM Manipulation is EASY!'

>>> for x in pq("a"):
...     print pq(x).text()
...
PyQuery
jQuery

>>> pq("ul").children().eq(0).html()
u'It makes parsing files a <strong>SNAP</strong>!'
```

```
<!DOCTYPE html>
<html>
  <head>
    <title>PyQuery Test!</title>
  </head>

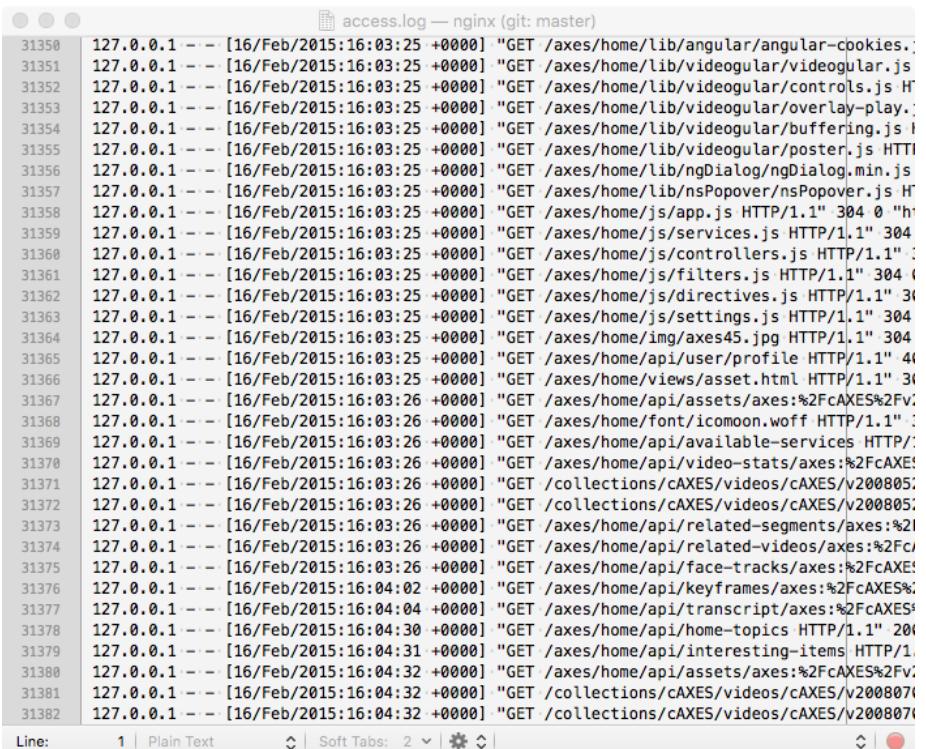
  <body>
    <h1>PyQuery is AWESOME!</h1>
    <p><a href="http://pypi.python.org/pypi/pyquery">PyQuery</a>
       is a Python port of the famous
       <a href="http://jquery.com">jQuery</a>
       JavaScript library.
    <h2>What is it Good For?</h2>
    <ul id="pitch">
      <li>It makes parsing files a <strong>SNAP</strong>!</li>
      <li>DOM Manipulation is EASY!</li>
      <li>You <em>never</em>
          have to worry about confusing syntax</li>
    </ul>
  </body>
</html>
```

Processing log files

Log files are a good example of semi-structured data. **Log analysis:** making sense and extracting information from log files.

Regular expressions and string partitioning are useful tools:

- Built-in Python [re](#) module
- Python string functions:
 - `str.split`, `str.rsplit`
 - `str.partition`, `str.rpartition`
 - `str.find`, `str.rfind`, `str.replace`
 - `str.strip`, `str.upper`, `str.lower`
 - `str[a:b:c]`
- Unix tools: [sed](#), [awk](#), [grep](#), [vim](#)



```
access.log — nginx (git: master)
31350 127.0.0.1 -- [16/Feb/2015:16:03:25 +0000] "GET /axes/home/lib/angular/angular-cookies.js HTTP/1.1" 304 638
31351 127.0.0.1 -- [16/Feb/2015:16:03:25 +0000] "GET /axes/home/lib/videoangular/videoangular.js HTTP/1.1" 304 103
31352 127.0.0.1 -- [16/Feb/2015:16:03:25 +0000] "GET /axes/home/lib/videoangular/controls.js HTTP/1.1" 304 103
31353 127.0.0.1 -- [16/Feb/2015:16:03:25 +0000] "GET /axes/home/lib/videoangular/overlay-play.js HTTP/1.1" 304 103
31354 127.0.0.1 -- [16/Feb/2015:16:03:25 +0000] "GET /axes/home/lib/videoangular/buffering.js HTTP/1.1" 304 103
31355 127.0.0.1 -- [16/Feb/2015:16:03:25 +0000] "GET /axes/home/lib/videoangular/poster.js HTTP/1.1" 304 103
31356 127.0.0.1 -- [16/Feb/2015:16:03:25 +0000] "GET /axes/home/lib/ngDialog/ngDialog.min.js HTTP/1.1" 304 103
31357 127.0.0.1 -- [16/Feb/2015:16:03:25 +0000] "GET /axes/home/lib/nsPopover/nsPopover.js HTTP/1.1" 304 103
31358 127.0.0.1 -- [16/Feb/2015:16:03:25 +0000] "GET /axes/home/js/app.js HTTP/1.1" 304 0 "http://127.0.0.1:8080"
31359 127.0.0.1 -- [16/Feb/2015:16:03:25 +0000] "GET /axes/home/js/services.js HTTP/1.1" 304 103
31360 127.0.0.1 -- [16/Feb/2015:16:03:25 +0000] "GET /axes/home/js/controllers.js HTTP/1.1" 304 103
31361 127.0.0.1 -- [16/Feb/2015:16:03:25 +0000] "GET /axes/home/js/filters.js HTTP/1.1" 304 103
31362 127.0.0.1 -- [16/Feb/2015:16:03:25 +0000] "GET /axes/home/js/directives.js HTTP/1.1" 304 103
31363 127.0.0.1 -- [16/Feb/2015:16:03:25 +0000] "GET /axes/home/js/settings.js HTTP/1.1" 304 103
31364 127.0.0.1 -- [16/Feb/2015:16:03:25 +0000] "GET /axes/home/img/axes45.jpg HTTP/1.1" 304 103
31365 127.0.0.1 -- [16/Feb/2015:16:03:25 +0000] "GET /axes/home/api/user/profile HTTP/1.1" 404 103
31366 127.0.0.1 -- [16/Feb/2015:16:03:25 +0000] "GET /axes/home/views/assets.html HTTP/1.1" 304 103
31367 127.0.0.1 -- [16/Feb/2015:16:03:26 +0000] "GET /axes/home/api/assets/axes:%2FcAXES%2Fv00805" 200 103
31368 127.0.0.1 -- [16/Feb/2015:16:03:26 +0000] "GET /axes/home/font/iconoon.woff HTTP/1.1" 304 103
31369 127.0.0.1 -- [16/Feb/2015:16:03:26 +0000] "GET /axes/home/api/available-services HTTP/1.1" 304 103
31370 127.0.0.1 -- [16/Feb/2015:16:03:26 +0000] "GET /axes/home/api/video-stats/axes:%2FcAXES%2Fv00805" 200 103
31371 127.0.0.1 -- [16/Feb/2015:16:03:26 +0000] "GET /collections/cAXES/videos/cAXES/v200805" 200 103
31372 127.0.0.1 -- [16/Feb/2015:16:03:26 +0000] "GET /collections/cAXES/videos/cAXES/v200805" 200 103
31373 127.0.0.1 -- [16/Feb/2015:16:03:26 +0000] "GET /axes/home/api/related-segments/axes:%2FcAXES%2Fv00805" 200 103
31374 127.0.0.1 -- [16/Feb/2015:16:03:26 +0000] "GET /axes/home/api/related-videos/axes:%2FcAXES%2Fv00805" 200 103
31375 127.0.0.1 -- [16/Feb/2015:16:03:26 +0000] "GET /axes/home/api/face-tracks/axes:%2FcAXES%2Fv00805" 200 103
31376 127.0.0.1 -- [16/Feb/2015:16:04:02 +0000] "GET /axes/home/api/keyframes/axes:%2FcAXES%2Fv00805" 200 103
31377 127.0.0.1 -- [16/Feb/2015:16:04:04 +0000] "GET /axes/home/api/transcript/axes:%2FcAXES%2Fv00805" 200 103
31378 127.0.0.1 -- [16/Feb/2015:16:04:30 +0000] "GET /axes/home/api/home-topics HTTP/1.1" 200 103
31379 127.0.0.1 -- [16/Feb/2015:16:04:31 +0000] "GET /axes/home/api/interesting-items HTTP/1.1" 200 103
31380 127.0.0.1 -- [16/Feb/2015:16:04:32 +0000] "GET /axes/home/api/assets/axes:%2FcAXES%2Fv00805" 200 103
31381 127.0.0.1 -- [16/Feb/2015:16:04:32 +0000] "GET /collections/cAXES/videos/cAXES/v200805" 200 103
31382 127.0.0.1 -- [16/Feb/2015:16:04:32 +0000] "GET /collections/cAXES/videos/cAXES/v200805" 200 103
```

Line:

1 | Plain Text

2

3

4

5

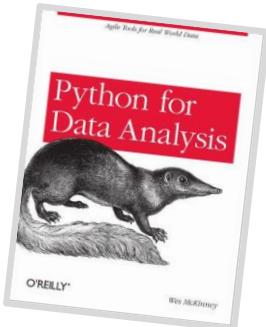


Data wrangling best practices

- **Keep raw data separate from cleaned data.** Never overwrite raw data. You may need it again. Keep backups!
- **Script data wrangling steps as much as possible.** If you need to change something later on, it's far easier if the steps are scripted.
- **Document all the transforms carried out and assumptions made.** Distribute this documentation with the cleaned dataset. Try make the wrangling process as reproducible as possible.
- **For large datasets, start with a small random sample.** Iterate faster: once you have perfected your cleaning steps on the sample, apply to full dataset.

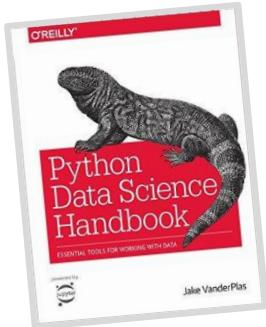
Questions?

Further reading



Python for Data Analysis, Wes McKinney ([DCU library](#))

- Chapter 5: Getting Started with Pandas
- Chapter 6: Data Loading, Storage, and File Formats
- Chapter 7: Data Wrangling: Clean, Transform, Merge, Reshape



Python for Data Science Handbook, Jake Vanderplas

([Available online](#))

- Chapter 3: Data Manipulation with Pandas

EEN1085/EEN1083

Data analysis and machine learning I

Ali Intizar

Semester 1
2024/2025



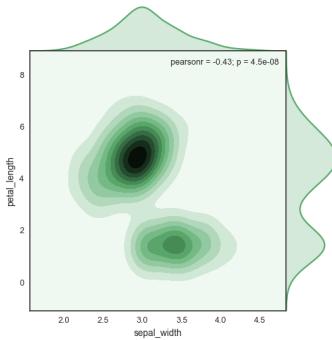
Ollscoil Chathair
Bhaile Átha Cliath
Dublin City University

Data summarization and visualization



Ollscoil Chathair
Bhaile Átha Cliath
Dublin City University

Overview



Data summarization

- Summary statistics
- Measures of central tendency
- Measures of variation
- Measures of correlation

Data visualization

- Why visualization is important
- Visualizing distributions
- Visualizing relationships
- Comparing quantities
- Visualizing composition
- Designing visualizations
- Non-statistical visualizations

Σ

Data summarization

Overview

Data distributions

Measures of **centrality**

- Mean, median, mode
- Sample mean vs population mean

Measures of **variation**

- Variance, standard deviation
- Range and Interquartile range

Sufficient statistics

Skewness and kurtosis

Conditioning on attributes

Measures of association

- Pearson
- Spearman

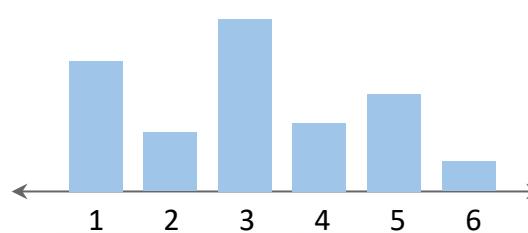
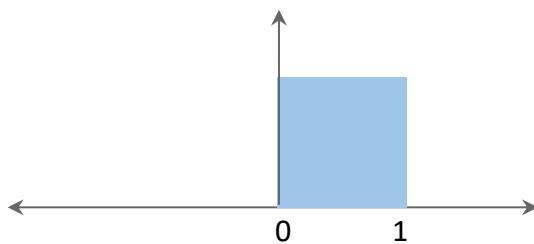
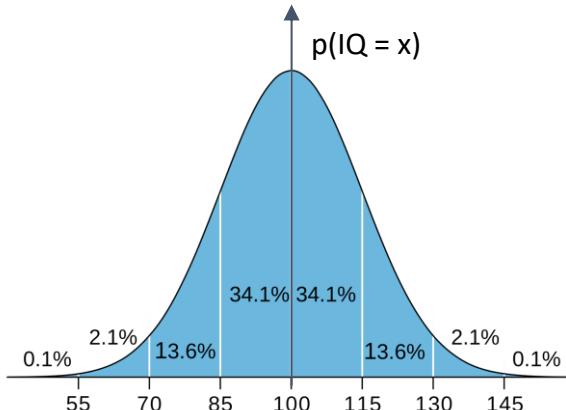
Distributions

The data distribution describes the probability of a random variable taking a particular value.

Example: IQ follows a **Normal (Gaussian) distribution**. ~70% of people have an IQ between 85 and 115, 95% between 70 and 130.

A random number generator (`np.random.rand`) will generate random numbers that follow a **uniform distribution** in $[0, 1]$

A roll of a biased dice has a **categorical distribution** (multinomial) over the values $\{1, 2, 3, 4, 5, 6\}$



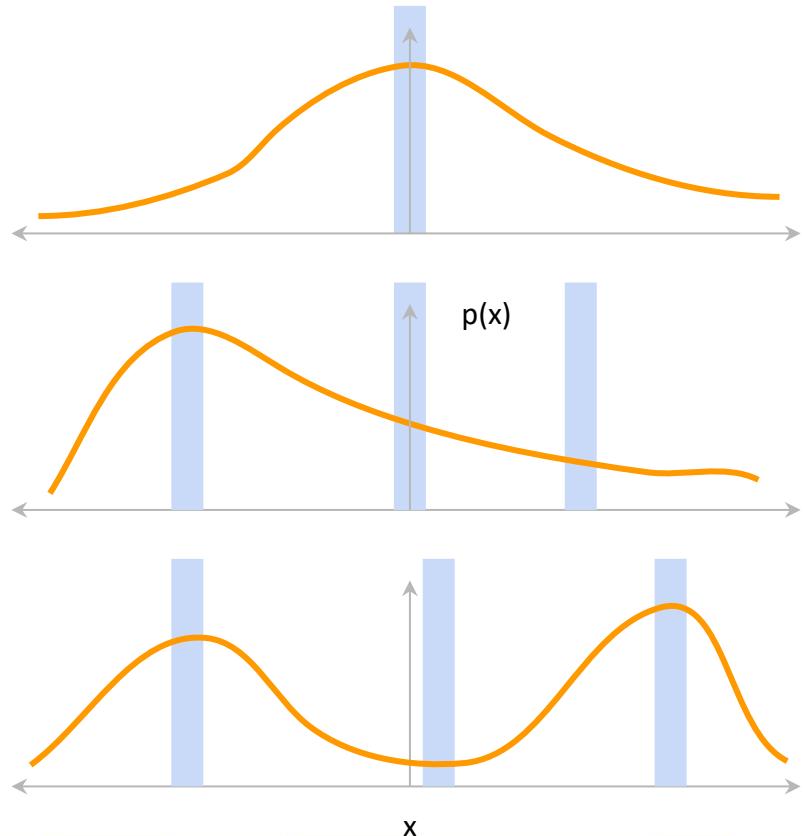
Measures of central tendency

A **sample statistic** is a measurement on a sample from a distribution, calculated by applying a function to the sample.

Measures of **central tendency** are sample statistics that attempt to capture where the middle of a distribution is.

Many ways of measuring central tendency.
Three common ones:

- Mean
- Median
- Mode



Mean, median, mode

Arithmetic mean: sum of observations divided by number of observations

$$\bar{x} = \frac{1}{N} \sum_{n=1}^N x_n$$

Median: the middle value that separates the higher half from the lower half of the data set

Sort data and take middle value.

Even number of observations: take arithmetic mean of middle two values

The median is a **robust statistic** (less sensitive to outliers)

Mode: the most frequent value in the dataset. Suitable measure of central tendency for nominal variables.

Easy to compute for discrete values: most frequent value.

Not straightforward for continuous distributions (some have no well-defined mode).

Mean, median, mode

1, 2, 2, 3, 4, 7, 9

Mean

$$(1 + 2 + 2 + 3 + 4 + 7 + 9) / 7 = 4$$

Median

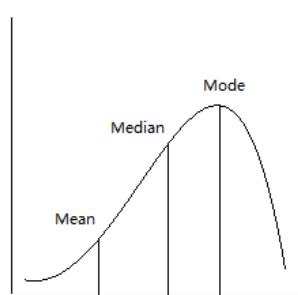
$$1, 2, 2, \mathbf{3}, 4, 7, 9 \\ = 3$$

Mode

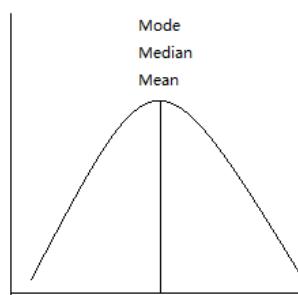
$$1, \mathbf{2}, \mathbf{2}, 3, 4, 7, 9 \\ = 2$$

Mean, median, mode

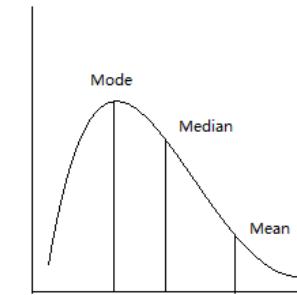
	Dimension	Type	Calculate
Mean	N	Quantitative	<code>np.mean</code>
Median	1	Ordinal	<code>np.median</code>
Mode	N	Nominal	From histogram



Left skew



Normal Distribution



Right skew

Sample mean vs population mean

The **population mean** is the true mean for the entire population.

$$\mu$$

The **sample mean** (average) is the calculated mean of a sample from a population.

$$\bar{x}$$

E.g. the population mean human height would require measuring the height of every person on earth. A sample mean 100 **randomly chosen** people.

As sample size increases the sample mean approaches the population mean.

$$\lim_{n \rightarrow \infty} \bar{x}_n = \mu$$

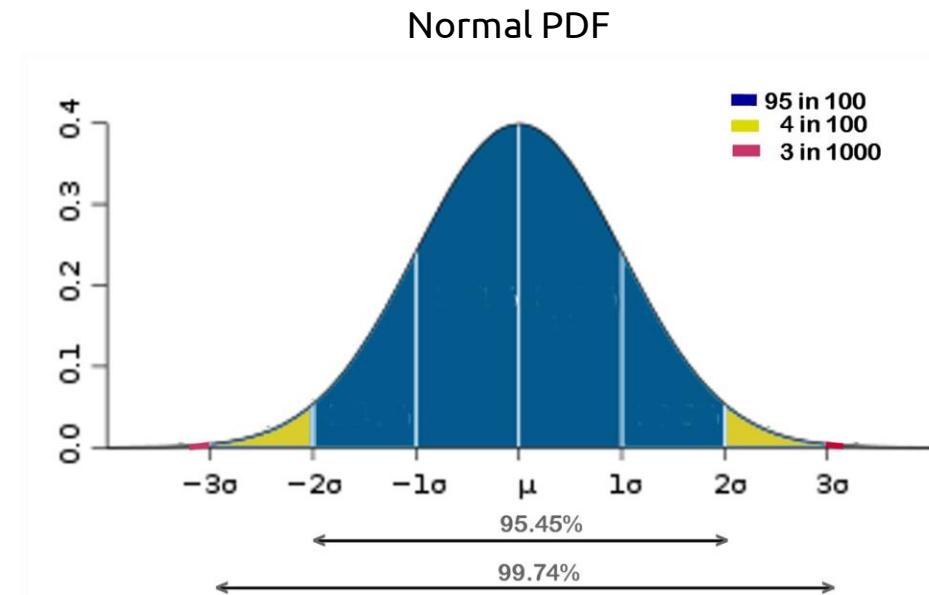
Measures of statistical dispersion

Measures of how stretched or squeezed a distribution is.

Most common methods are:

- Variance (σ^2)
- Standard deviation (σ)
- Interquartile range (IQR)

Again, can be defined for sample or population.



Variance, SD

Variance: the expected squared deviation from the mean.

$$\begin{aligned}\sigma^2 &= \mathbf{E}[(X - \mathbf{E}X)^2] \\ &= \mathbf{E}[(X - \mu)^2]\end{aligned}$$

$$\hat{\sigma}^2 = \frac{1}{N} \sum_{i=1}^N (x_i - \mu)^2$$

Usually we don't know the population mean μ , so we just use the sample mean:

$$\hat{\sigma}_N^2 = \frac{1}{N} \sum_{i=1}^N (x_i - \bar{x})^2$$

For small samples, this is biased, so may want to use the bias corrected estimator:

$$\hat{\sigma}_{N-1}^2 = \frac{1}{N-1} \sum_{i=1}^N (x_i - \bar{x})^2$$

The above is known as [**Bessel's correction**](#).

The **standard deviation** is just the square root of the (possibly bias corrected) variance.

$$\sigma = \sqrt{\sigma^2}$$

Range and interquartile range

The **range** of a sample is the difference between its minimum and maximum values.

The range is not a robust statistic!

The IQR is the range in which 50% of the data lies.

$$\text{IQR} = Q_3 - Q_1 = 119 - 31 = 88$$

```
q1, q3 = np.percentile(x, [25, 75])  
iqr = q1 - q3
```

Pandas: DataFrame.quantile()

i	x _i	Quartile
1	7	
2	7	
3	21	
4	31	Q ₁ (25% percentile)
5	47	
6	75	
7	87	Q ₂ (median)
8	115	
9	116	
10	119	Q ₃ (75% percentile)
11	119	
12	155	
13	177	

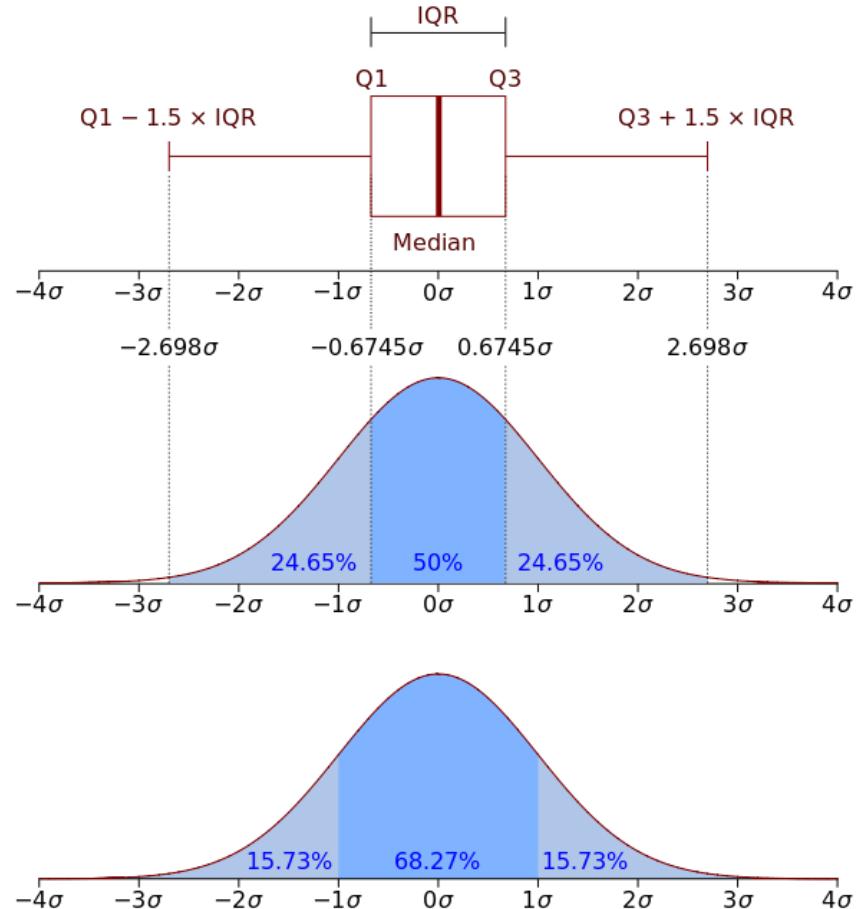
25% {

IQR }

25% {

Interquartile range

IQM can be used on ordinal variables and often makes more sense than the standard deviation.



Sufficient statistics

In statistics, a statistic is **sufficient** with respect to a statistical model and its associated unknown parameter when no other statistic that can be calculated from the same sample provides any additional information as to the value of the parameter.

For the Normal distribution, sufficient statistics are the mean and standard deviation.

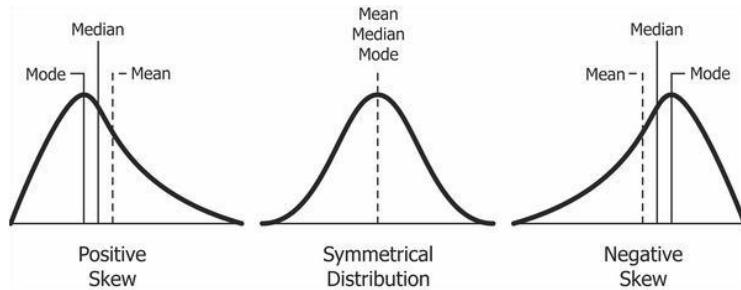
$$\theta = \{\mu, \sigma\}$$

Once you know the sufficient statistics, you know everything there is to know about the distribution of a variable.

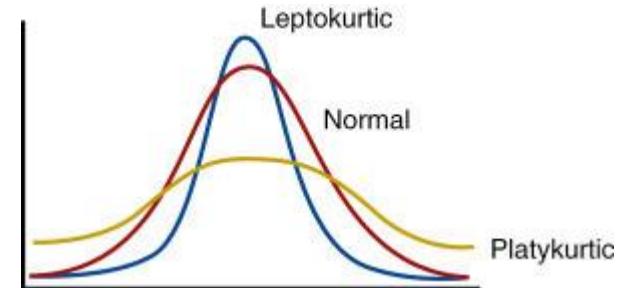
Skewness and kurtosis

Most data are not normally distributed.

Skewness is a measure of the asymmetry of the probability distribution of a real-valued random variable about its mean.



Kurtosis is a measure of how heavy the tails are of a probability distribution of a real-valued random variable.



Skewness and kurtosis

Pearson's moment coefficient of skewness

Third standardized moment:

$$\gamma_1 = E \left[\left(\frac{X - \mu}{\sigma} \right)^3 \right]$$

Pearson's moment coefficient of kurtosis

Fourth standardized moment:

$$\beta_2 = E \left[\left(\frac{X - \mu}{\sigma} \right)^4 \right]$$

Replace mu and sigma with sample statistics to calculate for a sample

[scipy.stats.skew\(x\)](#)

[scipy.stats.kurtosis\(x\)](#)

Pandas describe

```
import seaborn as sns
```

```
tips = sns.load_dataset("tips")
tips.head()
```

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4

```
tips.describe()
```

	total_bill	tip	size
count	244.000000	244.000000	244.000000
mean	19.785943	2.998279	2.569672
std	8.902412	1.383638	0.951100
min	3.070000	1.000000	1.000000
25%	13.347500	2.000000	2.000000
50%	17.795000	2.900000	2.000000
75%	24.127500	3.562500	3.000000
max	50.810000	10.000000	6.000000

Conditioning on categorical variables

Compute statistics based on subsets of the sample.

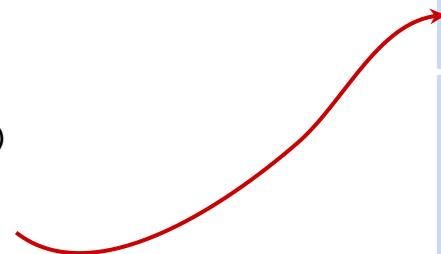
Subsets selected by grouping on categorical variables.

These are statistics on the conditional distributions. E.g. **P(size | time = Lunch)**

Pandas:

```
tips.groupby('time').describe()
```

```
tips.groupby('smoker').mean()
```

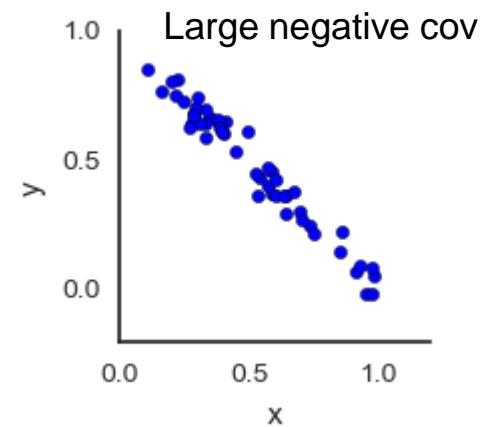
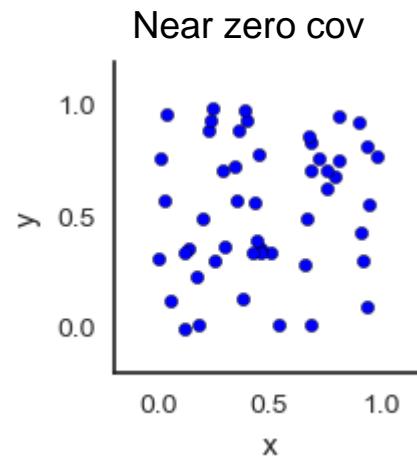
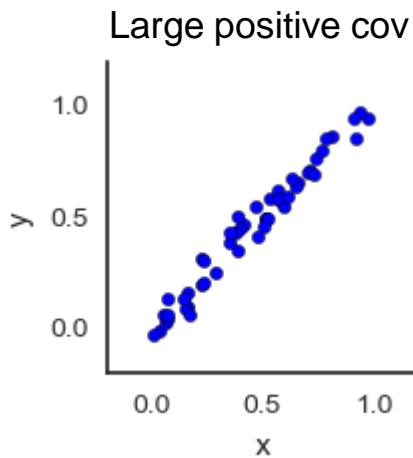


	size	tip	total_bill
Lunch	count	68.000000	68.000000
	mean	2.411765	2.728088
	std	1.040024	1.205345
	min	1.000000	1.250000
	25%	2.000000	2.000000
	50%	2.000000	2.250000
	75%	2.000000	3.287500
	max	6.000000	43.110000
Dinner	count	176.000000	176.000000
	mean	2.630682	3.102670
	std	0.910241	1.436243
	min	1.000000	1.000000
	25%	2.000000	2.000000
	50%	2.000000	3.000000
	75%	3.000000	3.687500
	max	6.000000	10.000000

Statistics of association between attributes

Covariance: measures how much two variables vary together.

$$\text{cov}(X, Y) = E[(X - \mu_X)(Y - \mu_Y)]$$



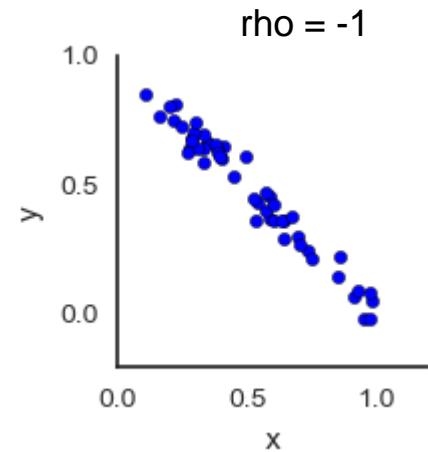
Statistics of association between attributes

Pearson product-moment correlation coefficient (Pearson's rho): measure of the linear relationship between quantitative variables. Normalized version of the covariance:

$$\rho_{X,Y} = \frac{\text{cov}(X, Y)}{\sigma_X \sigma_Y}$$

- Ranges from -1 to 1.
- > 0 : positive correlation
- < 0 : negative correlation
- +1 indicates a perfect linear relationship between X and Y
- -1 is perfect negative correlation

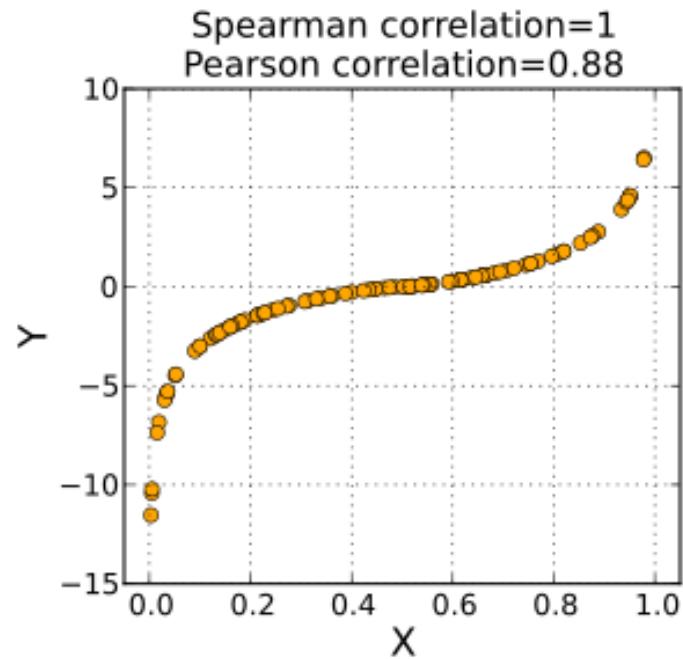
Not suitable for ordinal variables!



Spearman correlation

Spearman's rank correlation coefficient (Spearman's rho): the Pearson correlation between the rank values of those two variables.

Pearson's correlation assesses linear relationships, Spearman's correlation assesses monotonic relationships (whether linear or not).



Computing measures of association

Covariance

Numpy (covariance matrix)
`np.cov(x, y)`

Spearman

Scipy
`scipy.stats.spearmanr(a, b)`

Pandas

`df.corr(method='spearman')`

Pearson

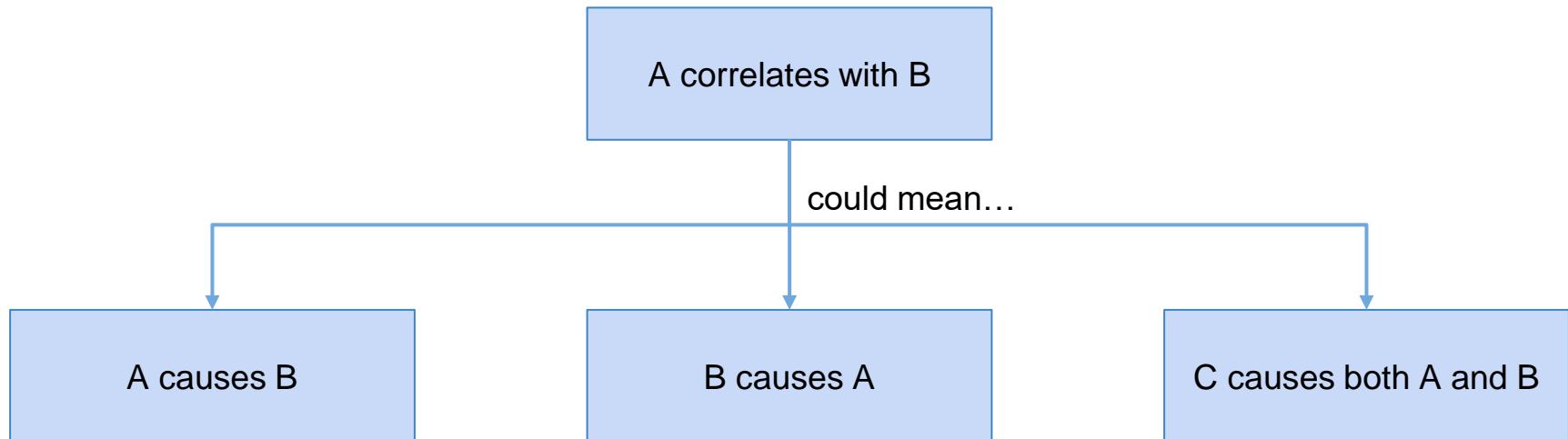
Scipy
`scipy.stats.pearsonr(x, y)`

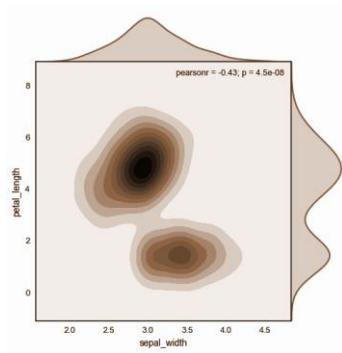
Pandas

`df.corr(method='pearson')`

On correlation vs causation

Correlation does NOT imply causality!





Data visualization

Why visualize data?

Explore

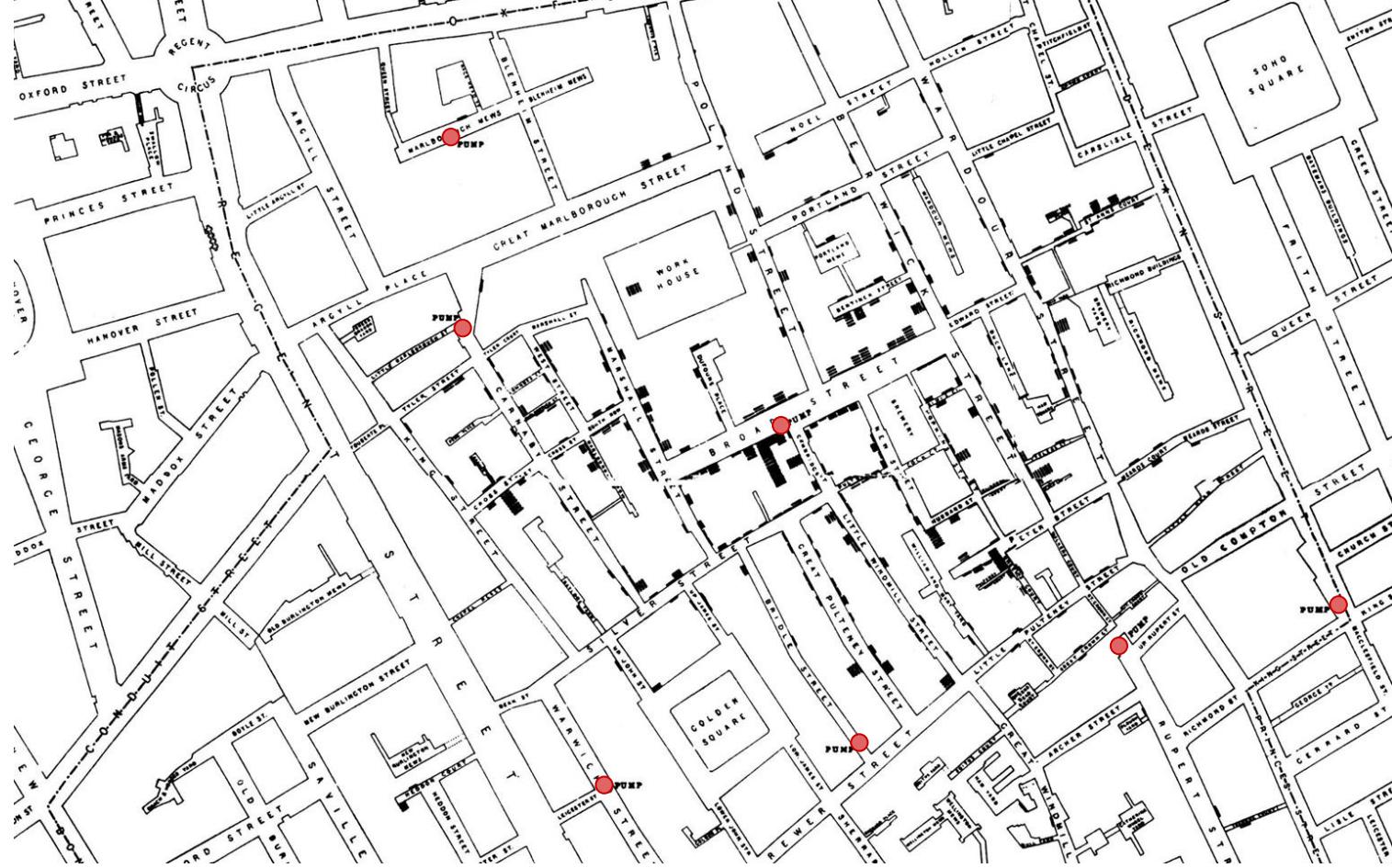
- Understand data and relationships between attributes
- Generate new hypotheses
- Detect errors and anomalies

Confirm

- Hypotheses already generated
- Check assumptions
- Verify conclusions

Communicate

- Analysis has been completed
- Conclusions verified
- Task: Present and communicate results
- Inform, persuade, educate, entertain



Explore:

John Snow's visualization of the London cholera outbreak of 1854



Explore:

John Snow's visualization of the London cholera outbreak of 1854

Carte Figurative des pertes successives en hommes de l'Armée Française dans la Campagne de Russie 1812-1813.

Dessinée par M. Minard, Inspecteur Général des Ponts et Chaussées en retraite.

Paris, le 20 Novembre 1869.

Les nombres d'hommes présents sont représentés par les largeurs des zones colorées à raison d'un millimètre pour dix mille hommes; ils sont de plus écrits en travers des zones. Le rouge désigne les hommes qui entrent en Russie, le noir ceux qui en sortent. Les renseignements qui ont servi à dresser la carte ont été puisés dans les ouvrages de M. M. Chier, de Segur, de Fezensac, de Chambray et le journal inédit de Jacob, pharmacien de l'Armée depuis le 28 Octobre.

Pour mieux faire juger à l'œil la diminution de l'armée, j'ai supposé que les corps du Prince Néronne et du Maréchal Davout, qui avaient été détachés sur Minsk et Mohilow et qui rejoignirent vers Orscha et Wiltsk, avaient toujours marché avec l'armée.

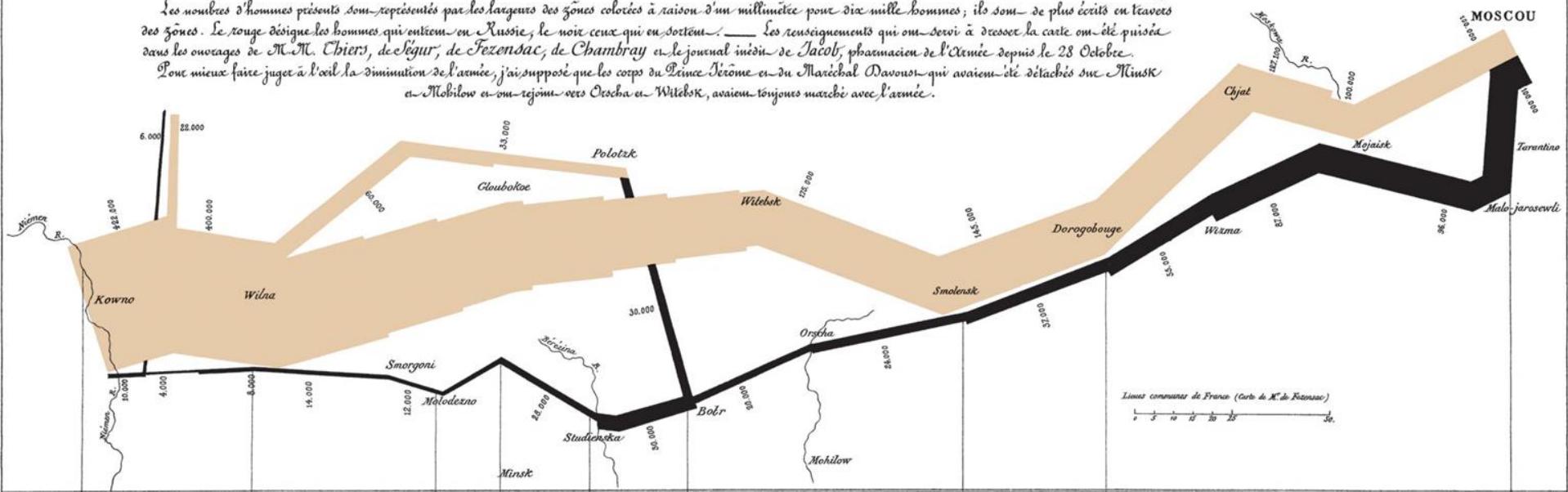


TABLEAU CRAPHIQUE de la température en degrés du thermomètre de Réaumur au dessous de zéro.

Les Cosaques passent au galop
le Niemen gelé.

-26° le 7 X.^{bre}
 -30° le 6 X.^{bre}
 -24° le 1^{er} X.^{bre}
 -20° le 28 9.^{bre}
 -11°
 -21° le 14 9.^{bre}
 -5° le 9 9.^{bre}

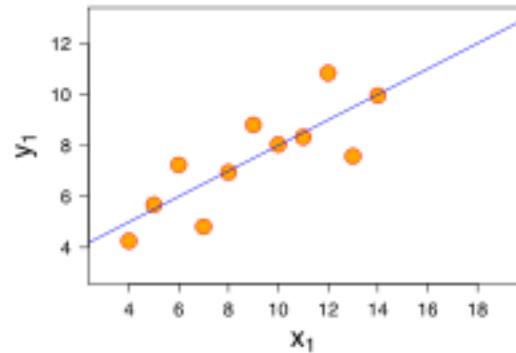
Zéro le 18 8.^{bre}
Pluie 24 8.^{bre}
5
10
15
20
25
30 degrés

Present and communicate:

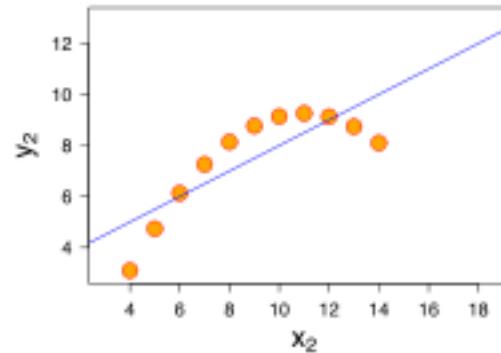
Charles Minard's map of Napoleon's disastrous Russian campaign of 1812.

Importance of visualization: Anscombe's quartet

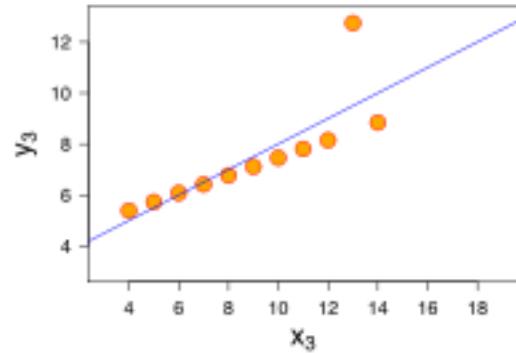
Linear + Gaussian Noise



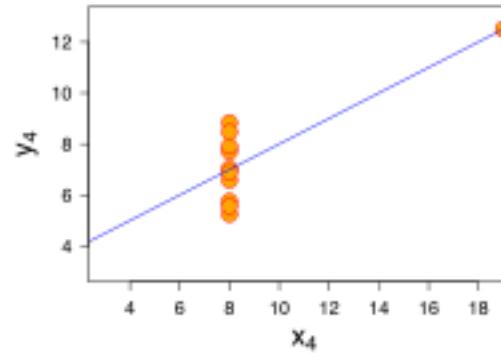
Nonlinear relationship



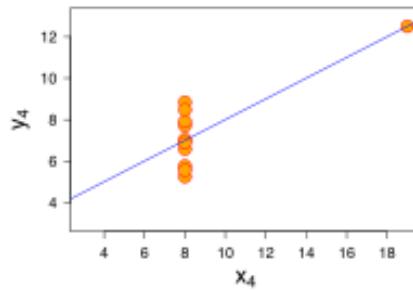
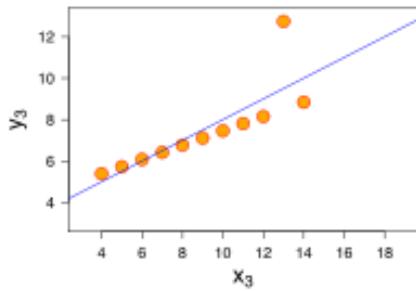
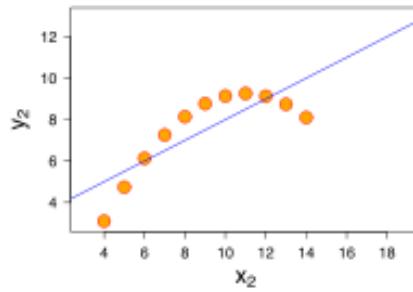
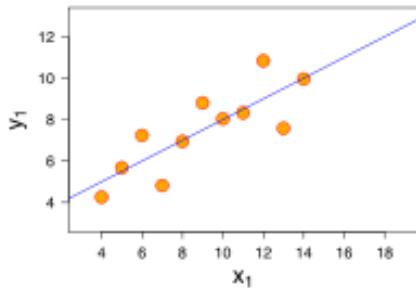
Linear + outlier



No relationship + outlier



Importance of visualization: Anscombe's quartet



Mean of x in each case	9
Sample variance of x in each case	11
Mean of y in each case	7.50
Sample variance of y in each case	4.122 - 4.127
Correlation between x and y in each case	0.816
Linear regression line in each case	$y = 3.00 + 0.500x$

Statistical data visualization

Visualizations like the ones just shown are extremely informative, but it takes **significant time** to conceptualize and create.

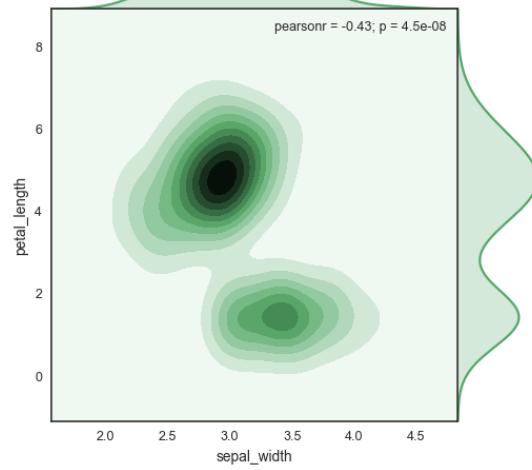
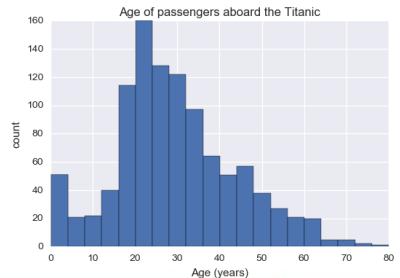
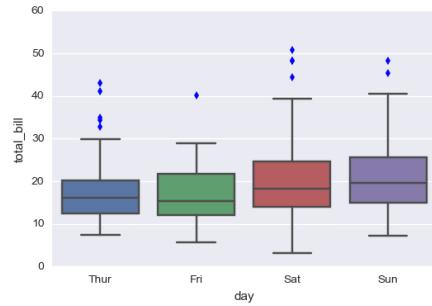
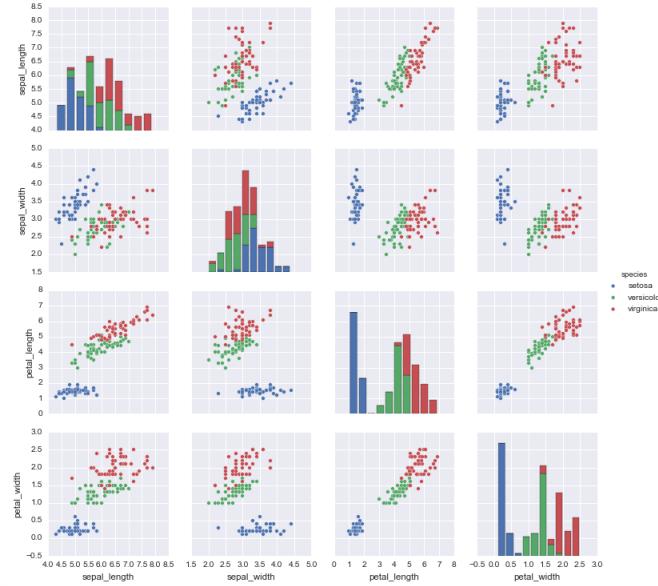
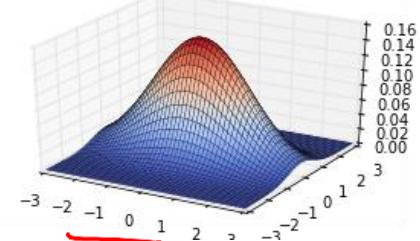
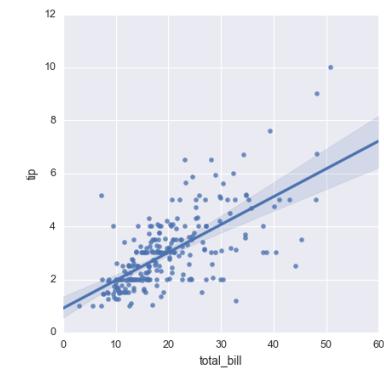
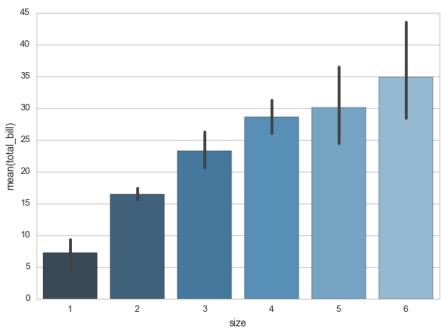
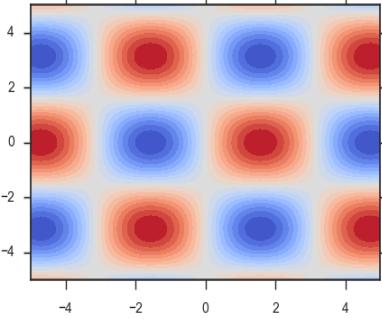
There are many “off the shelf” visualizations that we can use for routine exploratory and visualization work.

Statistical graphics are used to explore the statistical properties of data in a visual way

Statistical data visualizations provide a set of standard visual tools for:

- Understanding the shape and distribution of data
- Understanding the relationships between attributes
- Understanding composition and part-whole relationships
- Comparing quantities





Software packages for visualization

Python

- [Matplotlib](#): 2D plotting library with **simple** MATLAB style API
- [Seaborn](#): Built on Matplotlib and provides a high-level interface for drawing attractive **statistical** graphics
- [Bokeh](#): **interactive** visualization library that targets modern web browsers

Others

- [ggplot2](#) (R language)
- [d3js](#) (Javascript)
- [Processing](#) (Java)



Topics

- Visualizing the shape and distribution of data
- Visualizing relationships between attributes
- Comparing quantities
- Visualizing composition and part-whole relationships
- Design principles and visualization faux pas

Visualizing the data distribution

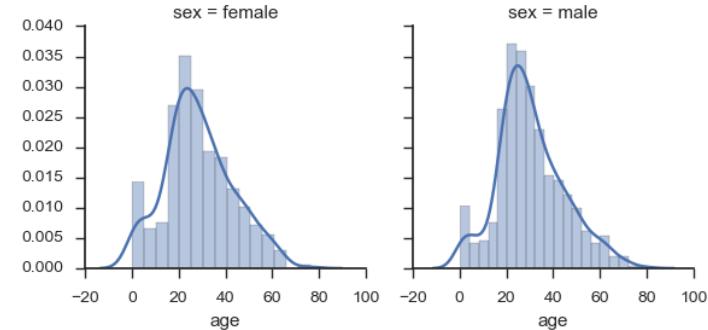
Data distribution: where are the areas in which most of the data lives.

Shape

- Is a variable normally distributed?
- Is it multi-modal?
- Are there long tails? Fat tails?
- Is the distribution skewed?

Modes

- Where are the modes of the distribution?
- Which values occur most frequently?



Comparison

- Are two distributions the same? Different?
- Does conditioning on a variable affect the distribution?

Outliers: are there obvious outliers that need to be removed/imputed?

Histograms

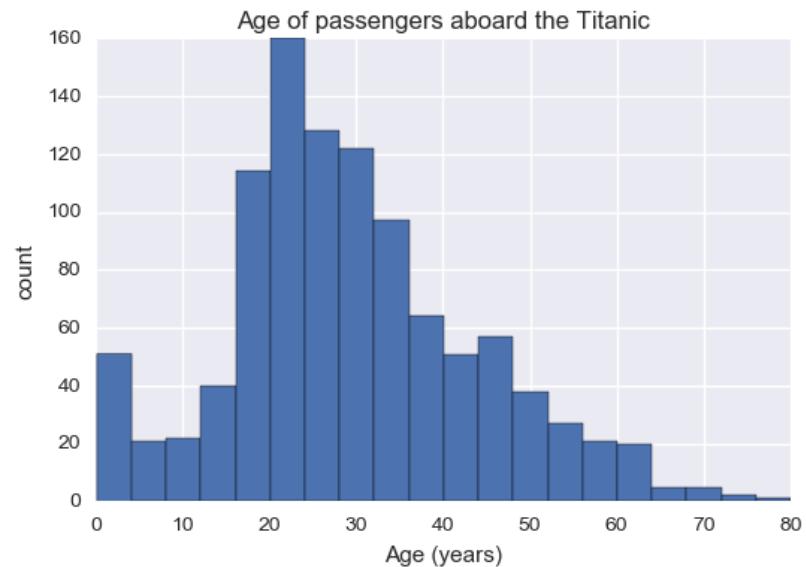
A histogram is a graphical representation of the distribution of numerical data.

Histograms allow you to look at the shape of the distribution of a single quantitative variable

Construction: divide the range of values into intervals (**bins**) and count how many values fall in each bin

Matplotlib:

```
ages = df['age'].dropna()  
hist(ages, bins=20)  
xlabel('Age (years)'); ylabel('count')  
title('Age of passengers aboard the Titanic')
```



Histograms

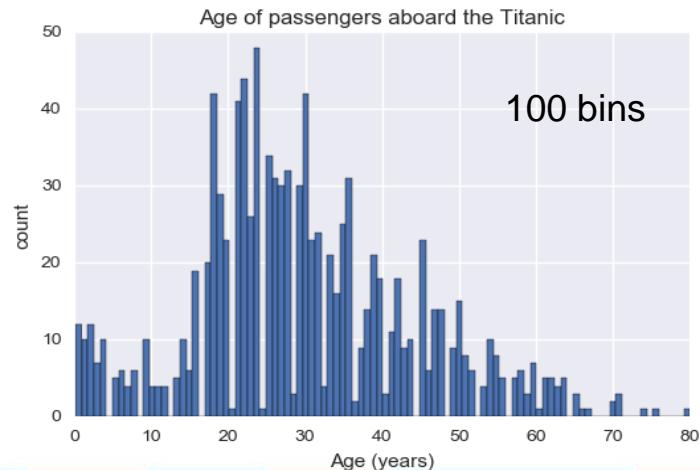
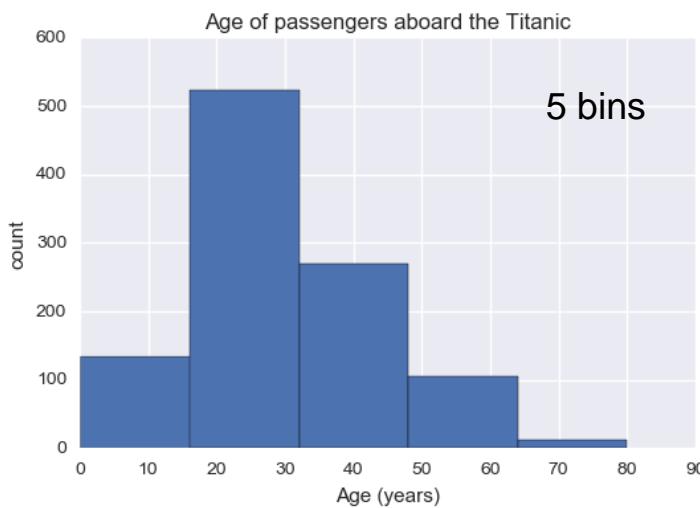
Uses:

- Get an idea for the shape of a distribution
- Determine if attribute is unimodal or multimodal
- See maximums and minimums
- Spot outliers

Choosing the right number of bins is important!

- Too few fails to capture details in shape
- Too many: lots of empty bins

Usually bins are of fixed width, but not necessarily.



Kernel density estimators

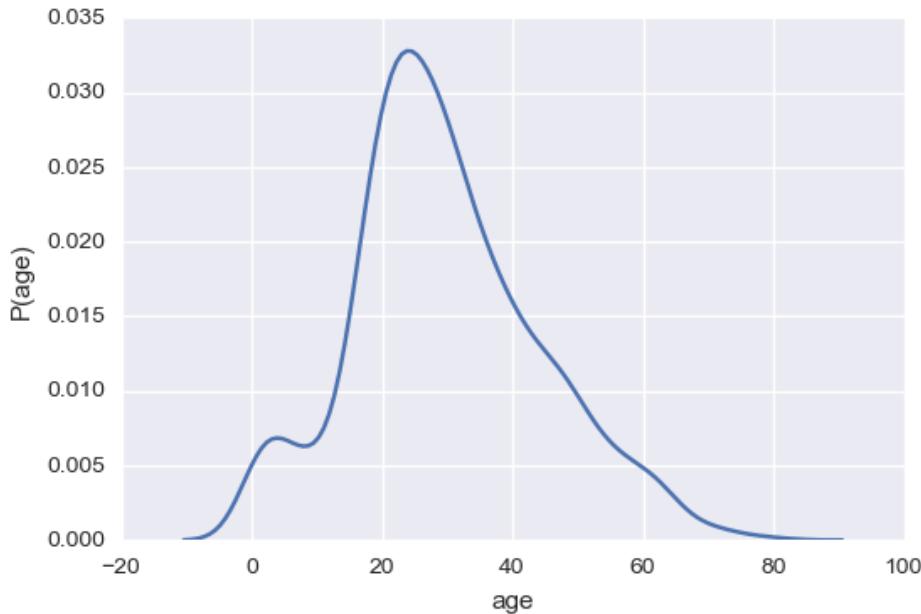
Basic idea: estimate the probability density at a point by counting points in local neighbourhood

$$\hat{f}(x) = \frac{1}{N} \sum_{i=1}^N K\left(\frac{x - x_i}{h}\right)$$

K(.) is a kernel function, h is a bandwidth parameter. K function is smaller for large values and integrates to 1.

Seaborn

```
sns.distplot(ages, hist=False)
xlabel('age'); ylabel('P(age)')
```

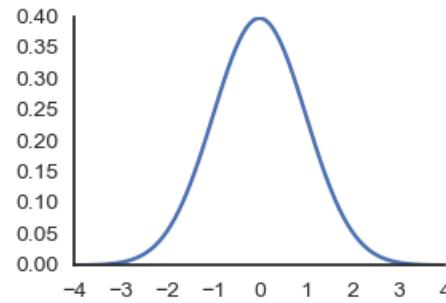


Kernel density estimators

Gaussian kernel:

$$K(x) = \frac{1}{\sqrt{2\pi}} \exp(-0.5x^2)$$

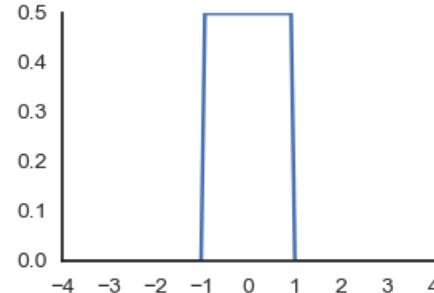
Produces smooth density plots



Uniform kernel:

$$K(x) = \frac{1}{2} I(|x| \leq 1)$$

Produces histograms!



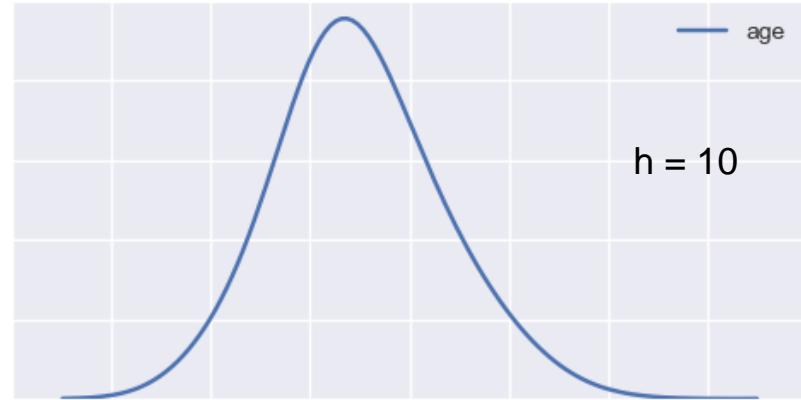
Kernel density estimators

Histograms: important to choose number of bins appropriately

Density plots: important to choose bandwidth (h) properly

- Too large: over smoothing
- Too small: not enough smoothing

```
sns.kdeplot(ages, bw=10)  
sns.kdeplot(ages, bw=0.1)
```



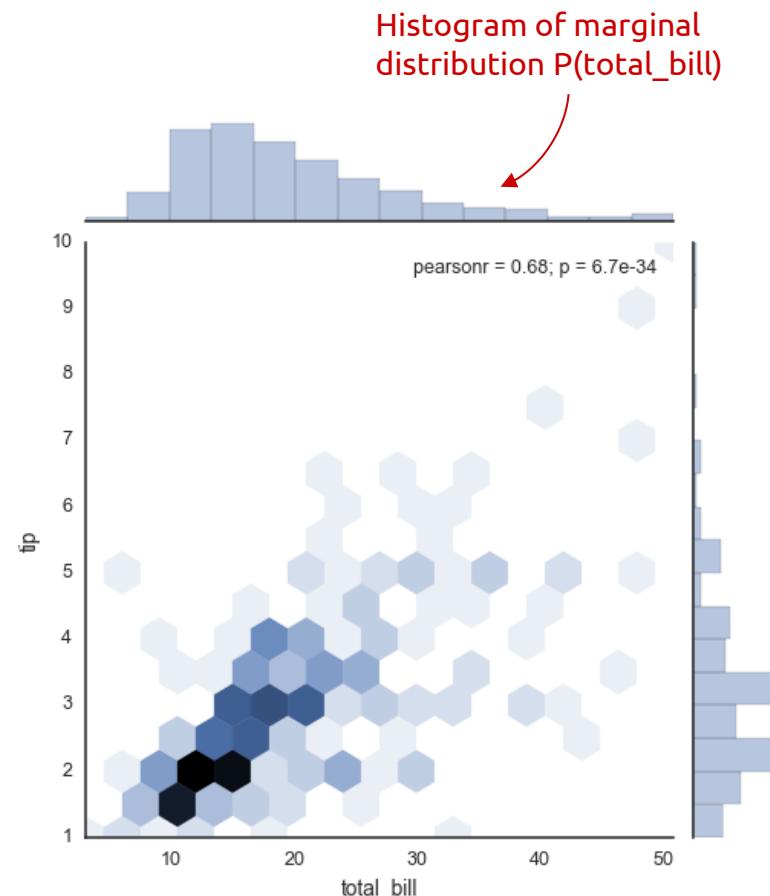
Joint distributions

Standard histograms and density plots are fine for plotting distribution of single variable.

What if we want to visualize the **joint distribution** of two variables $P(x, y)$

Hex plots are 2D analogies to histograms.

```
tips = sns.load_dataset("tips")
sns.jointplot(
    "total_bill", "tip",
    data=tips, kind="hex")
```

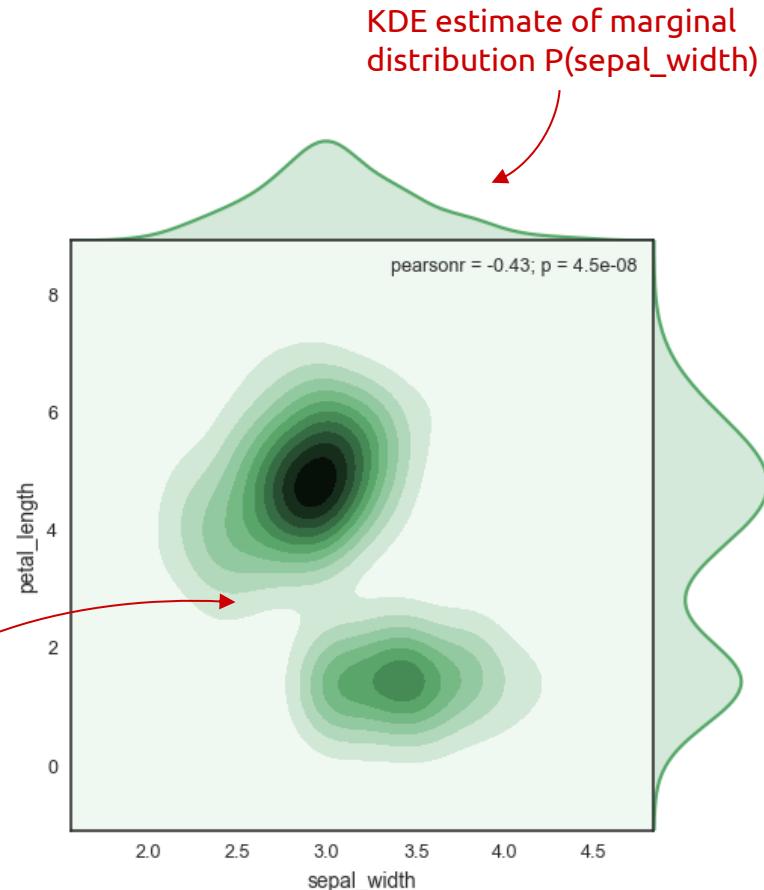


Joint distributions

Kernel density estimators also possible in 2D

```
iris = sns.load_dataset("iris")  
  
sns.jointplot(  
    "sepal_width", "petal_length",  
    data=iris, kind="kde",  
    space=0, color="g")
```

This part is called a
contour plot of $P(x,y)$



Contour plots

Incidentally, contour plots are not only for probability densities, but can be used to visualize any 2D function $f(x, y) \rightarrow R$

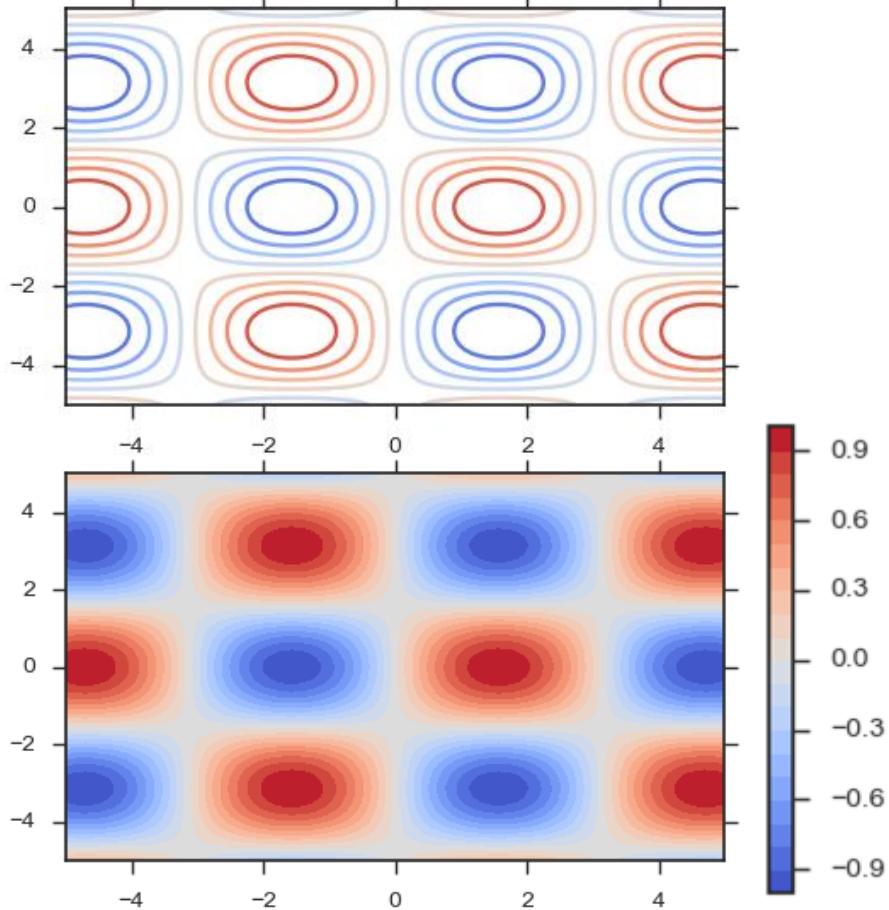
E.g.

$$f(x, y) = \sin(x) \cos(y)$$

```
# create a grid of x, y values
xs = linspace(-5, 5, 100)
ys = linspace(-5, 5, 100)
X, Y = meshgrid(xs, ys)

# define function and evaluate on grid
f = lambda x, y: np.sin(x) * np.cos(y)
Z = f(X, Y)

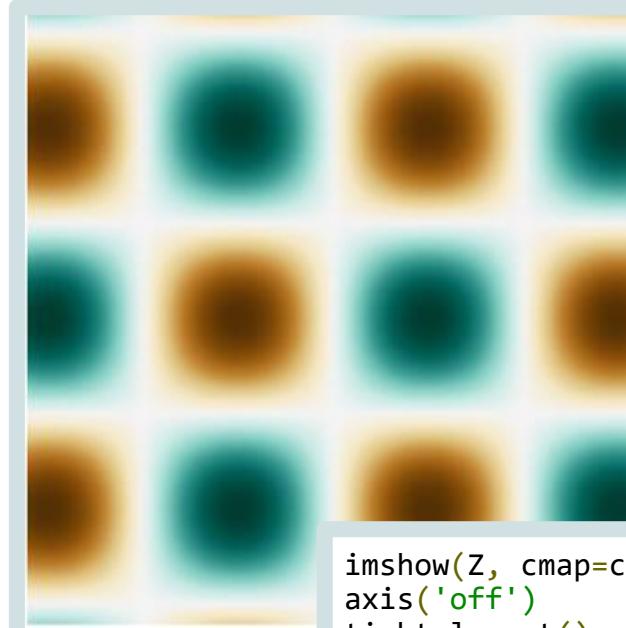
# plot filled contours
contourf(X, Y, Z, levels=linspace(-1, 1, 20),
          cmap=cm.coolwarm)
```



Using images to visualize functions of two variables



```
imshow(Z, cmap=cm.gray)  
axis('off')  
tight_layout()
```



```
imshow(Z, cmap=cm.BrBG)  
axis('off')  
tight_layout()
```

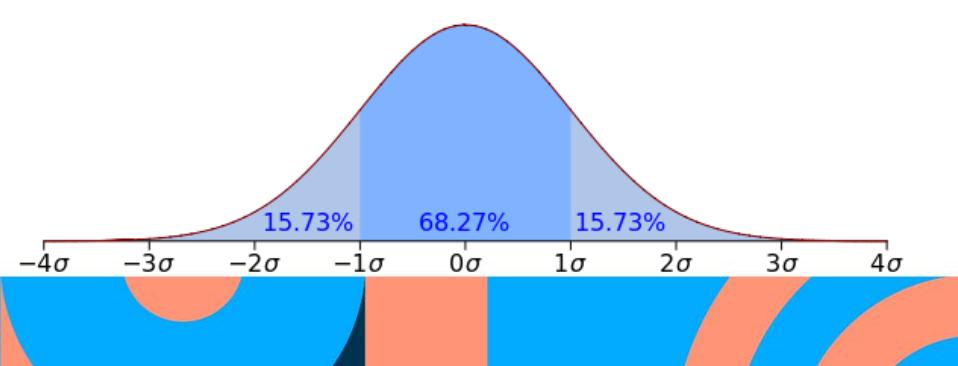
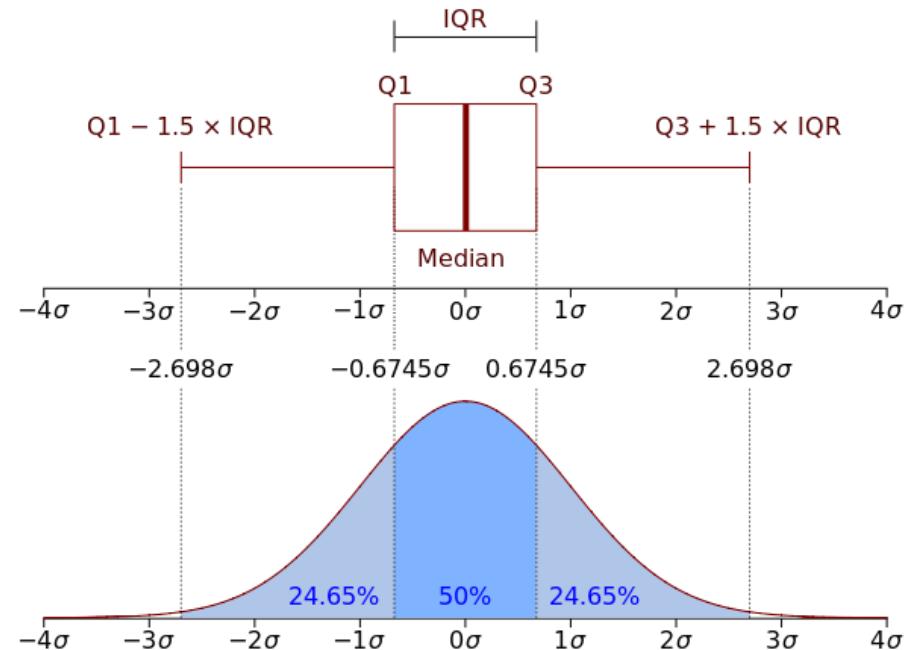
Boxplots

A **boxplot** (or box-and-whisker plot) shows the distribution of quantitative data in a way that facilitates comparisons between variables or across levels of a categorical variable.

The box shows the quartiles of the dataset

The whiskers extend to show the rest of the distribution, except for points that are determined to be outliers

The median is shown in the center of the box.



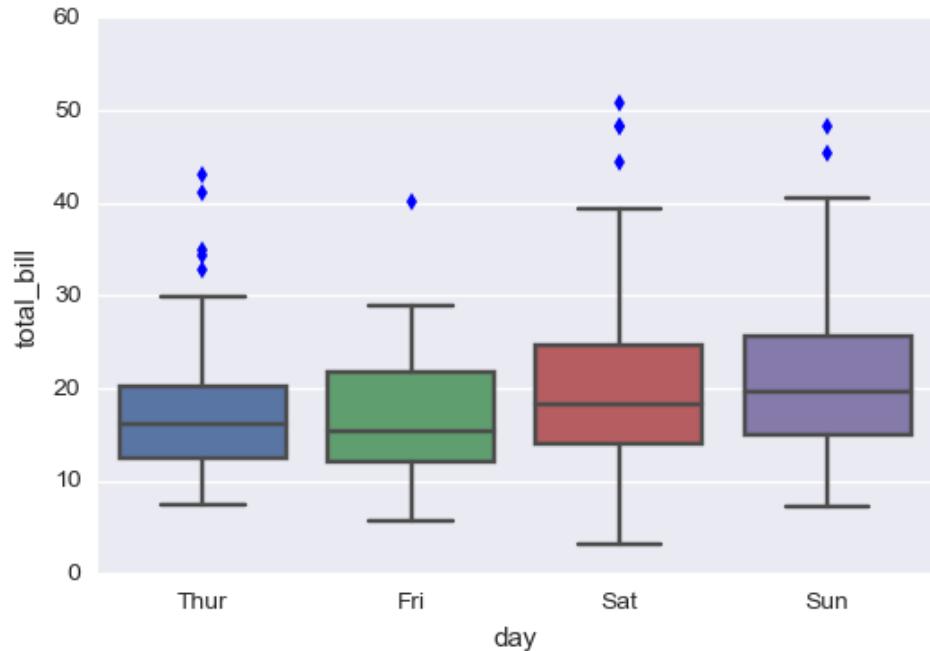
Boxplots

Especially useful for

1. detecting **outliers**
2. **comparing** distributions conditioned on a categorical (nominal) variable.

```
tips = sns.load_dataset("tips")
```

```
sns.boxplot(  
    x="day",  
    y="total_bill",  
    data=tips)
```



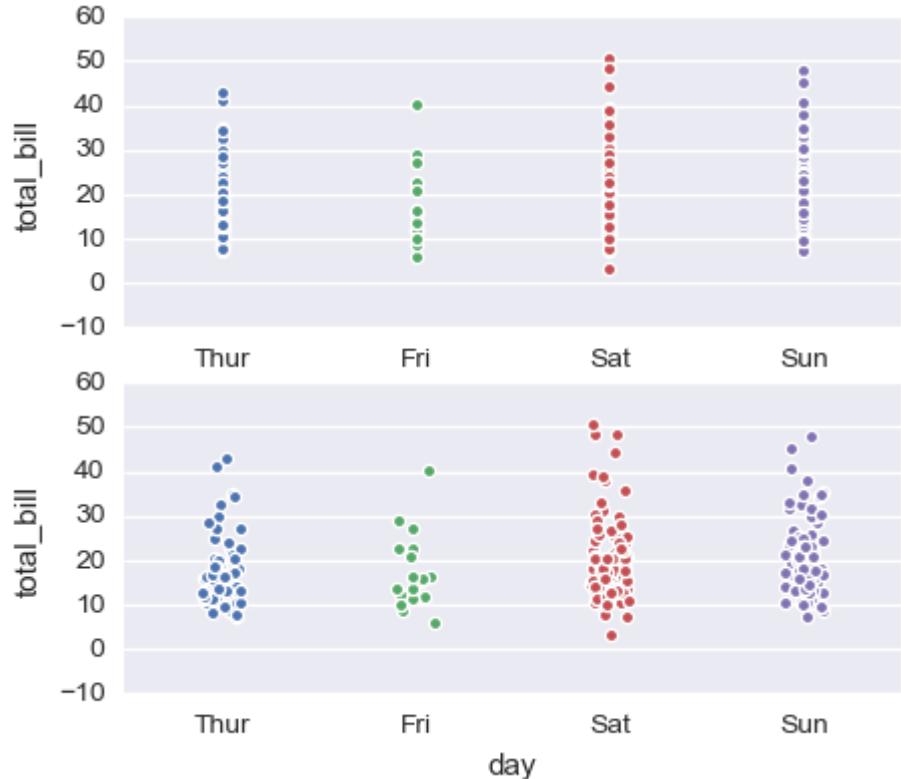
Strip plots and jitter plots

It is often good to plot the raw data points to ensure that you are not being misled by factors like the kernel density estimator or skewed distributions.

Plotting the data points on a 1D line often results in **overplotting**.

One technique to alleviate overplotting is to add some noise to the x-coordinate. This is called **jitter**.

```
sns.stripplot(  
    x="day", y="total_bill",  
    data=tips, jitter=True)
```



Topics

- Visualizing the shape and distribution of data
- Visualizing relationships between attributes
- Comparing quantities
- Visualizing composition and part-whole relationships
- Design principles and visualization faux pas

Visualizing relationships between variables

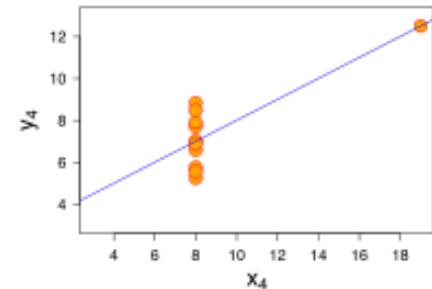
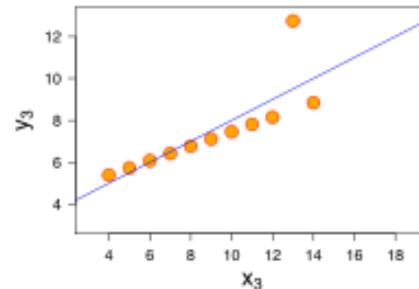
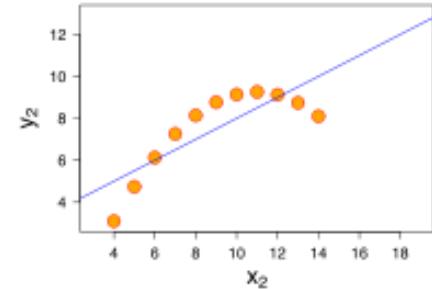
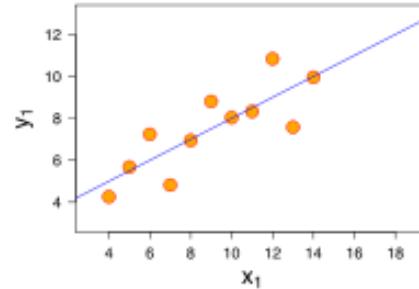
Pearson's and spearman's rho can be used to numerically measure linear and rank correlation.

Visualization is an even more powerful way to examine the relationship between variables:

- Non-linear relationships
- Non-rank order relationships

Some statistical graphics:

1. Scatter plots
2. Regression plots
3. Line plots



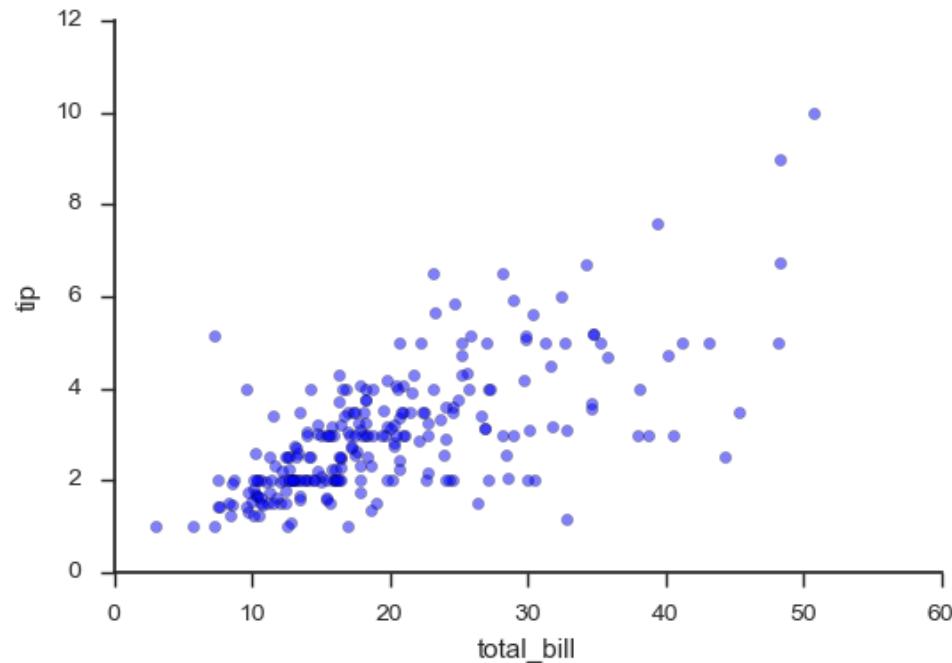
Scatter plots

Scatter plots (**scattergrams**) show the relationship between two variables by plotting the values of one variable against the other.

If variables are **linearly correlated**, they should appear to cluster around a diagonal line.

If variables are **uncorrelated**, or statistically **independent**, they will appear randomly scattered across the graph.

Non-linear relationships may also be apparent in a scatterplot.



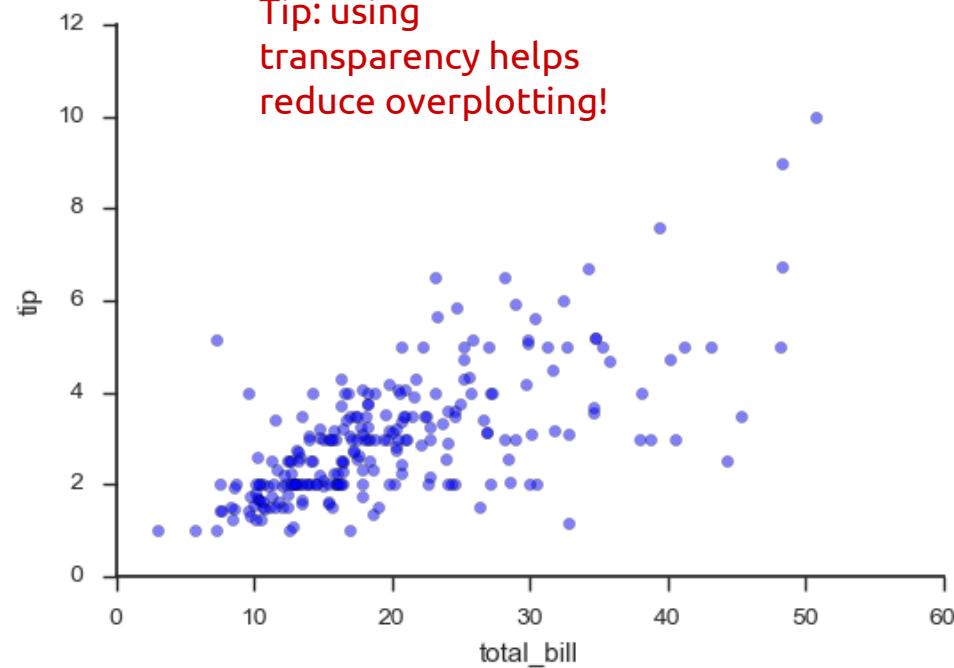
Scatter plots

Generating scatter plots with matplotlib (and seaborn for style):

```
sns.set_style('ticks')
scatter(
    tips['total_bill'],
    tips['tip'], alpha=0.5)

xlabel('total_bill'); ylabel('tip')

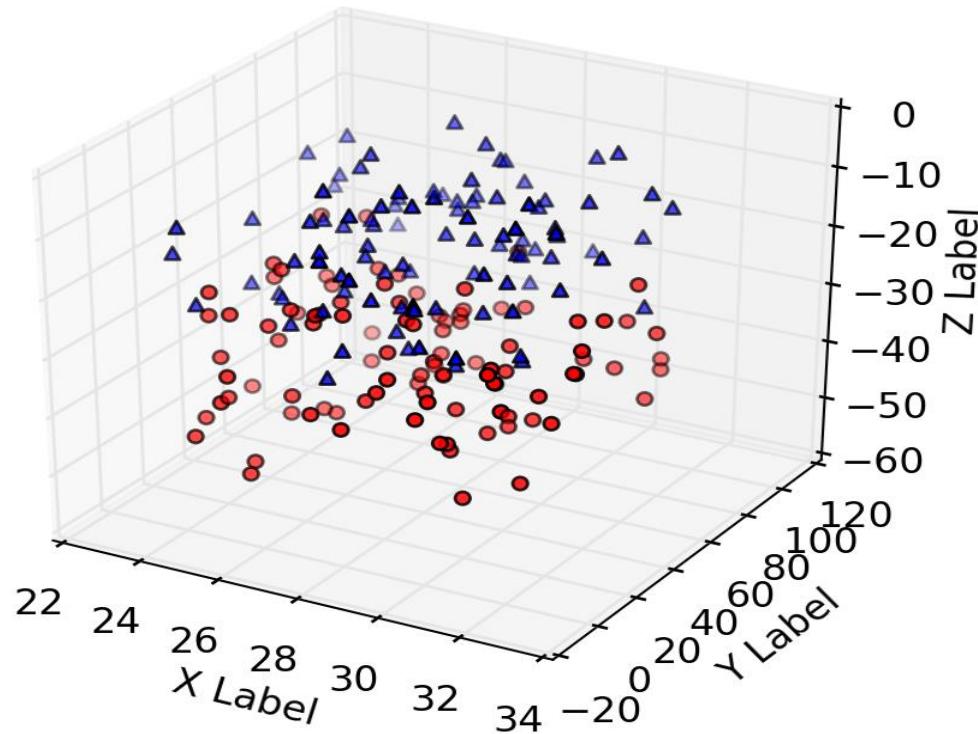
# remove lines from box
sns.despine()
```



3D scatterplots

Usually not very useful.

Avoid.

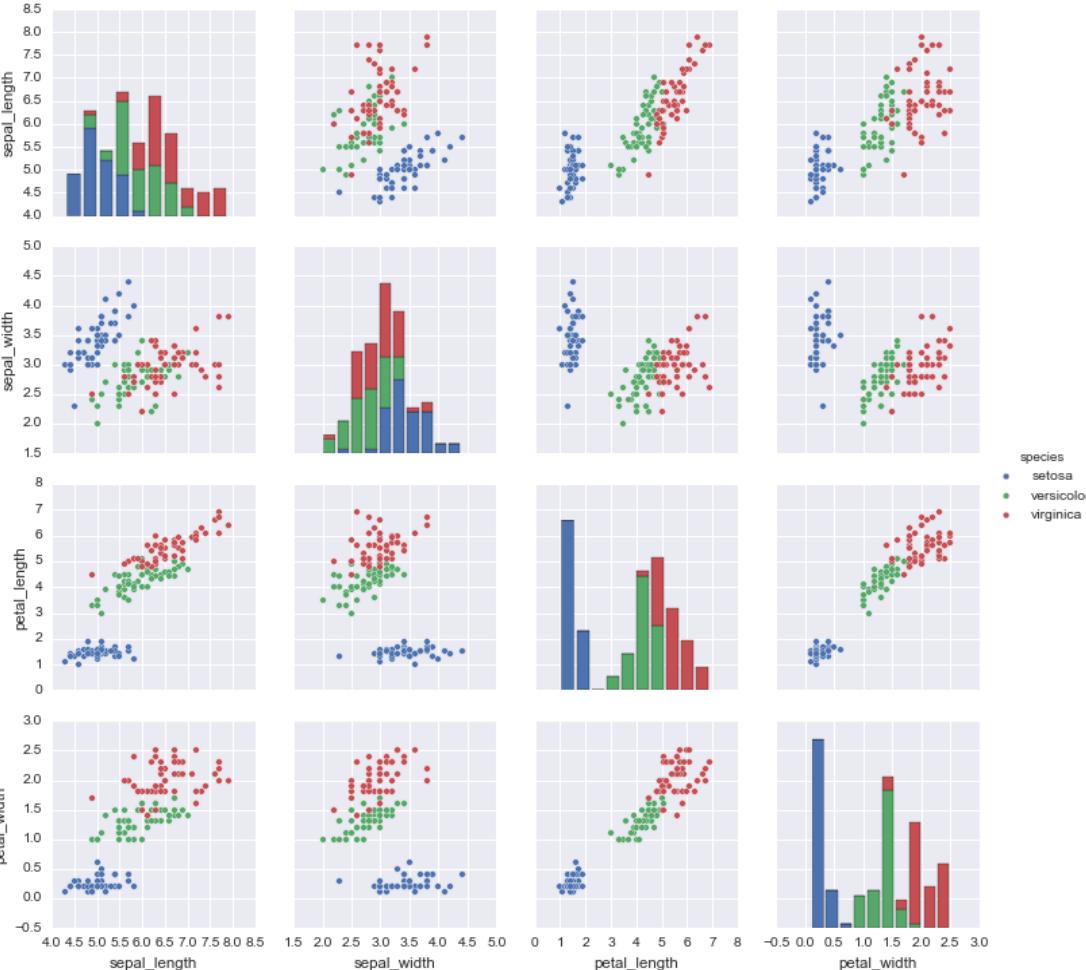


Scatterplot matrix

If you want to look at the relationships between several pairs of attributes simultaneously, you can use a scatterplot matrix.

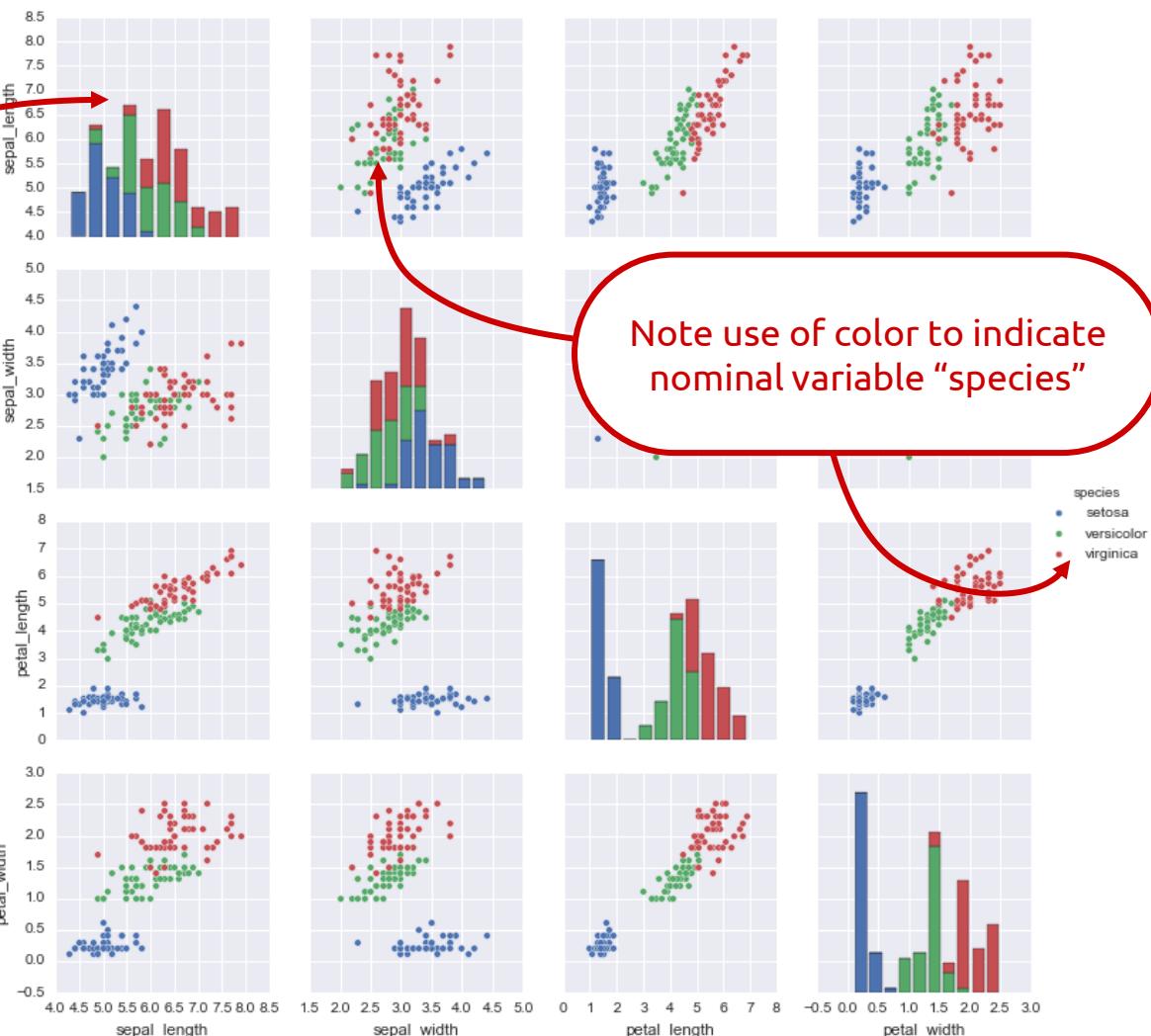
The relationship between variable 1 and 2 is shown in (1, 2) in the matrix. Its transpose is in (2, 1).

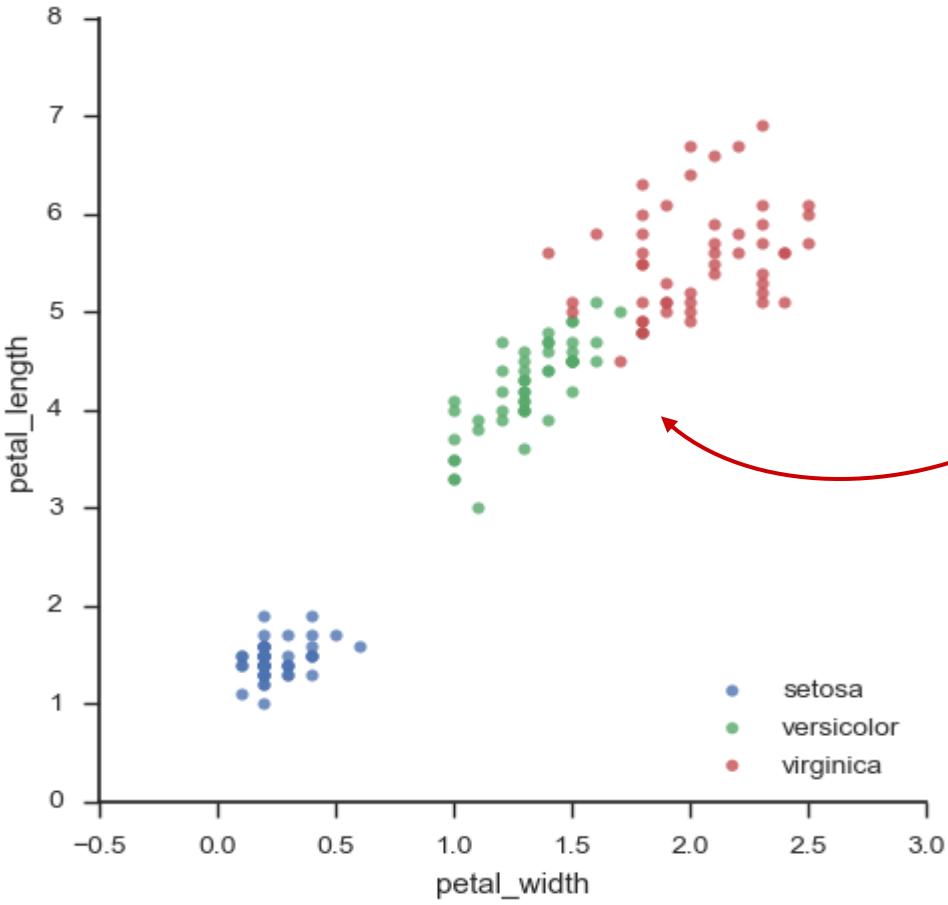
The correlation between any variable and itself is always 1. Plotting packages often use the diagonal of the scatterplot matrix to show additional info.



Seaborn shows univariate histograms of variables on the diagonal by default

```
iris =  
sns.load_dataset('iris')  
sns.pairplot(  
    iris, hue='species')
```





Notice anything strange about the data in this graph?

Values are quantized!

```
iris = sns.load_dataset('iris')
sns.lmplot(
    "petal_width", "petal_length", data=iris,
    hue='species', fit_reg=False, legend=False)
legend(loc='lower right')
```

On the use of color

Color can be an incredibly useful tool for including more dimensions in a 2D plot.

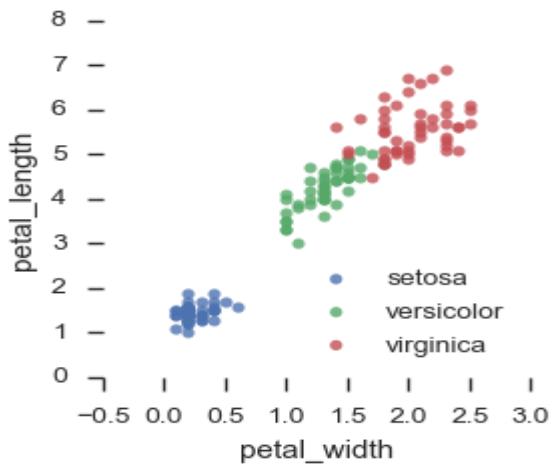
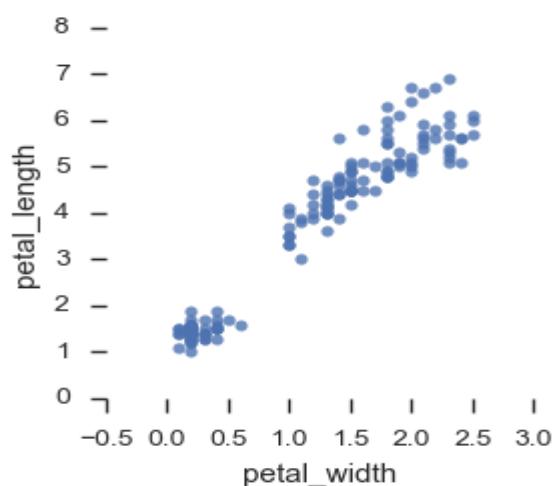
Tufte: color can be your greatest ally or worst enemy if used improperly

Example on right uses different colors to overlay the nominal variable species on the scatter plot.

The choice of color palette used is important, and should reflect the type of variable being indicated and what you want to show.

Recall Stephen's scales of measurement:

1. Nominal (categorical)
2. Ordinal (ordered)
3. Quantitative (ratio or interval)



Choosing a color palette



Qualitative palettes

Do not imply an inherent ordering. Suitable for nominal variables. E.g. species, gender, category, etc.



Sequential palettes

Imply an ordering. Appropriate when data range from relatively **low or uninteresting** values to relatively **high or interesting** values. Suitable for ordinal and quantitative variables. E.g. risk, crime rate



Diverging palettes

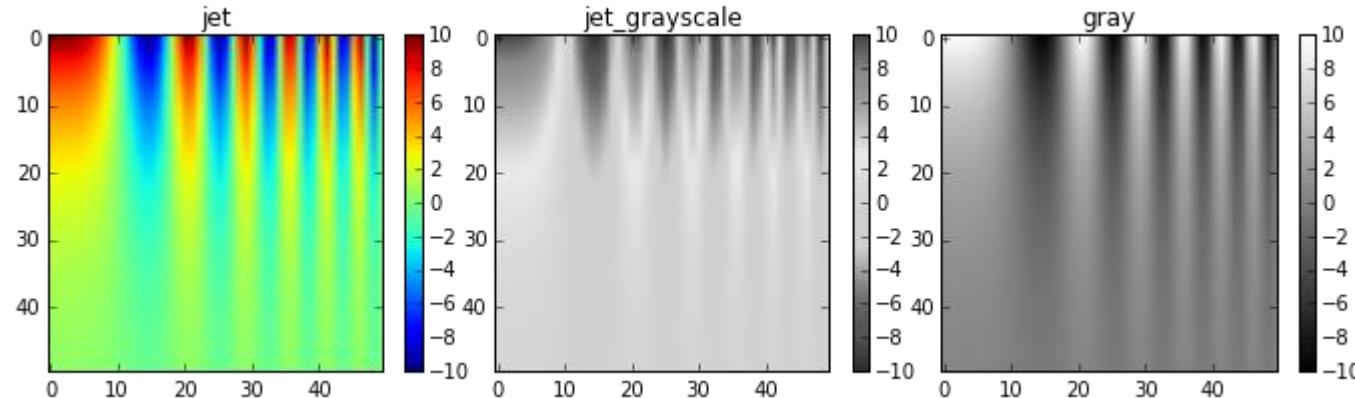
Also imply an ordering and suitable for ordinal and quantitative variables. Used for data where both **large low and high values are interesting**. Interpretable **midpoint** in the data. E.g. temperature, rainfall

Avoid “jet” and “rainbow”



Jet is (or was) the default color bar originally used by MATLAB.

Complex luminance profile tends to introduce artifacts that are not really there! Highlights features that may not exist.

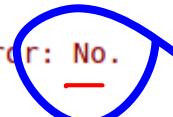


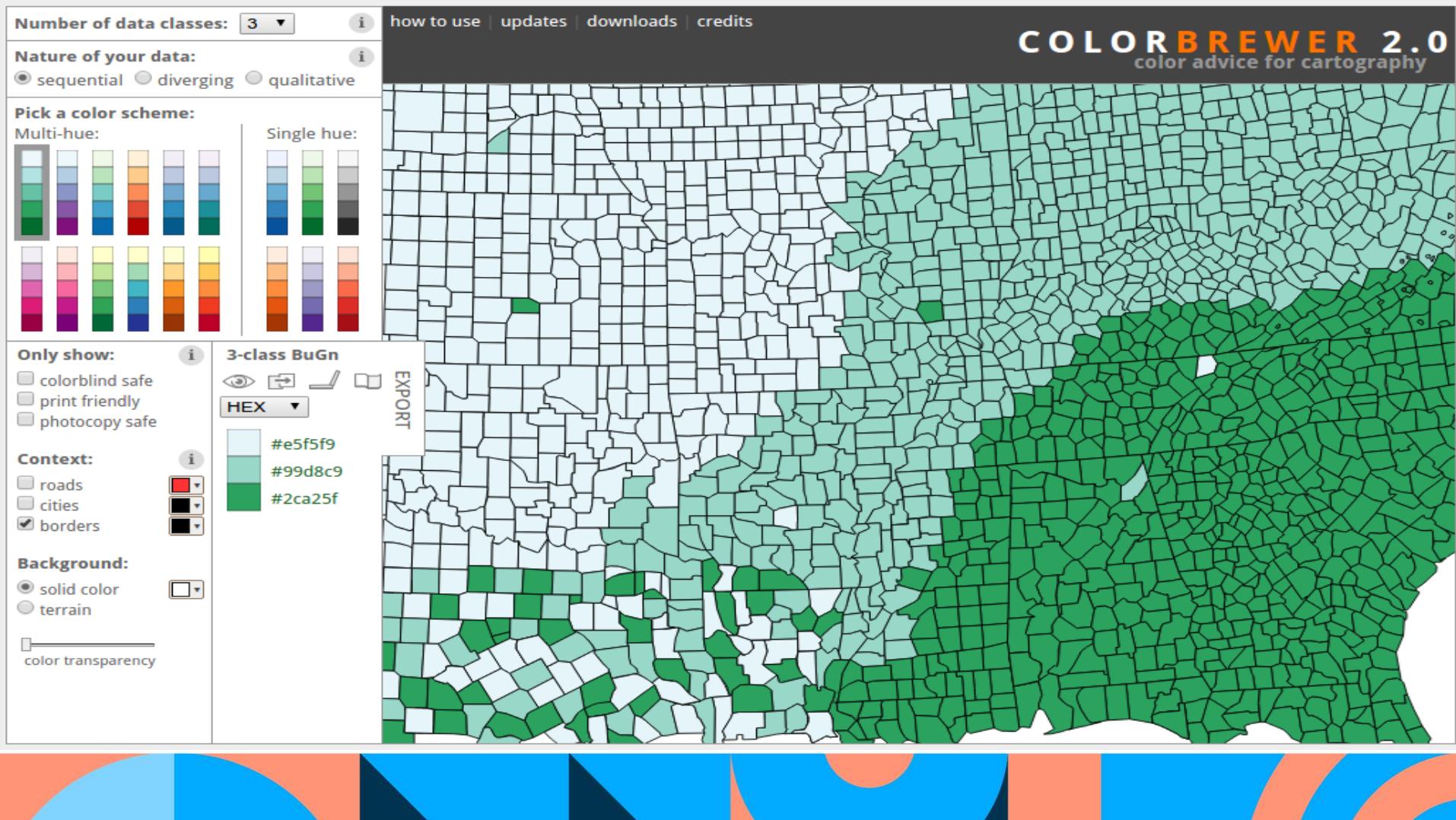
Avoid “jet” and “rainbow”

```
In [217]: sns.color_palette('jet', 7)
-----
-----
ValueError                                Traceback (most recent call last)
ast)
<ipython-input-217-aa7828b71ac8> in <module>()
----> 1 sns.color_palette('jet', 7)

/usr/local/lib/python2.7/dist-packages/seaborn/palettes.pyc in color_pa
lette(palette, n_colors, desat)
    167         palette = husl_palette(n_colors)
    168     elif palette.lower() == "jet":
--> 169         raise ValueError("No. ")
    170     elif palette in SEABORN_PALETTE:
    171         palette = SEABORN_PALETTE[palette]

ValueError: No.
```





Regression lines

When we think there is a linear relationship between two variables (x, y), we may want to plot a regression line to illustrate it.

$$y = (ax + b) + \epsilon$$

- y is the **target** variable (aka. dependant variable)
- x is the **predictor** (feature, independent variable)
- a, b are **parameters**
- ϵ is **noise** (assume iid Gaussian)

To do this we can find the parameters (a, b) that minimize the sum of squared errors:

$$\sum_{i=1}^N (y_i - (ax_i + b))^2$$

The parameters can be found using a little calculus and algebra. Compute deriv wrt. (a, b) and set to zero and solve linear system.

We'll see how to do this in more detail when we study **supervised learning** later...

Regression lines

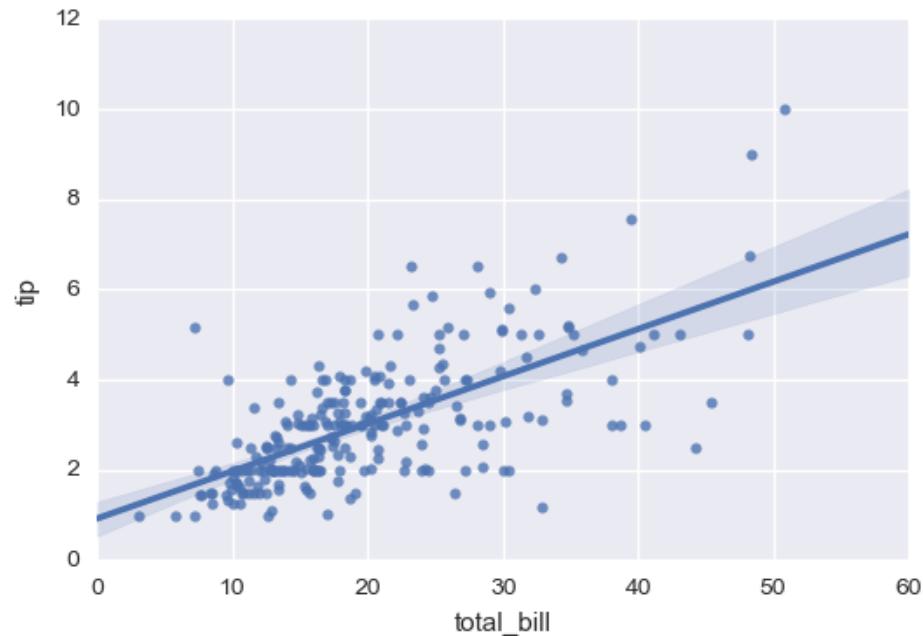
Seaborn can be used to plot regression lines.

```
sns.regplot('total_bill', 'tip', data=tips)
```

By default, it will also plot **confidence intervals** too (estimated using bootstrap)

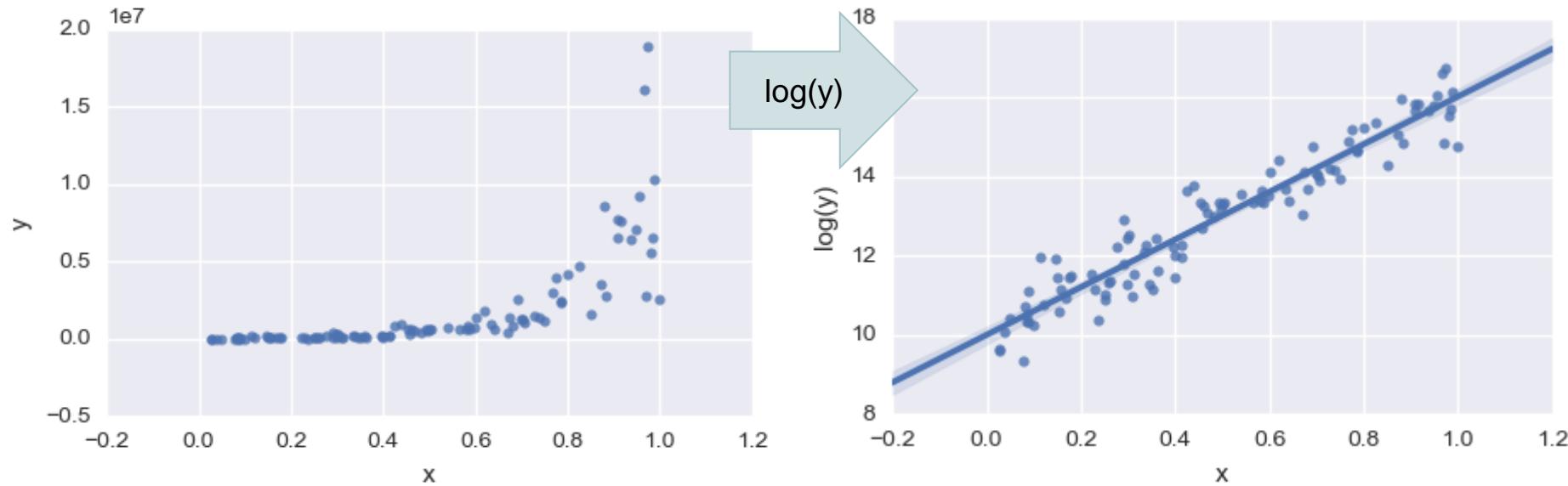
The **bootstrap** is a non-parametric way of estimating confidence intervals.

Idea: sample data with replacement 100 times. Each time fit a regression line. Look at variation in fits

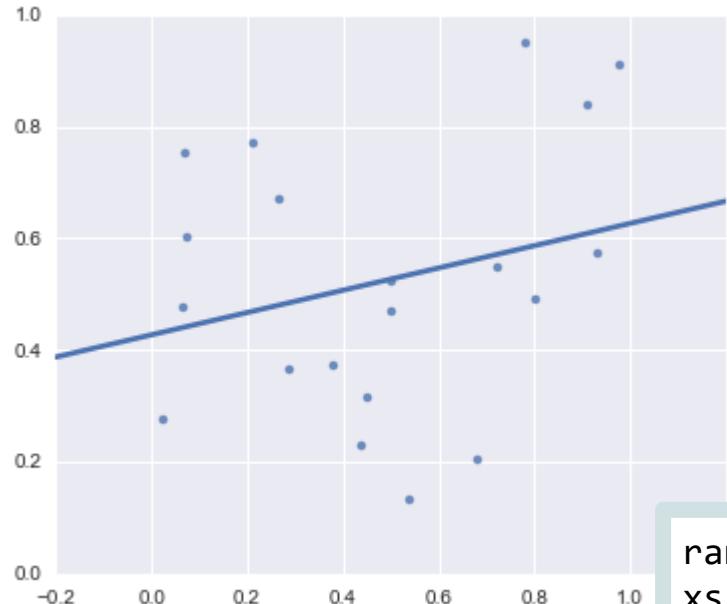


Axes scales

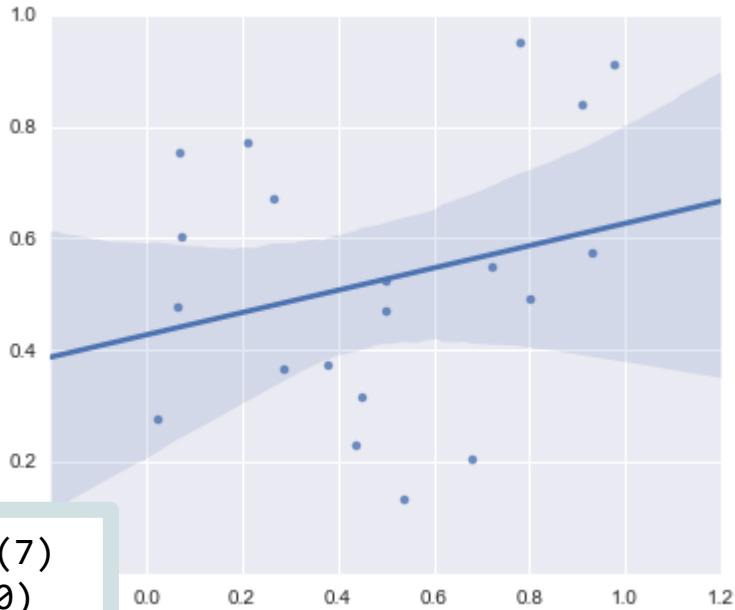
Sometimes data appears correlated, but not linearly. If $y = \exp(ax + b) + \text{noise}$, then changing the axes scale from linear to logarithmic allows a linear fit.



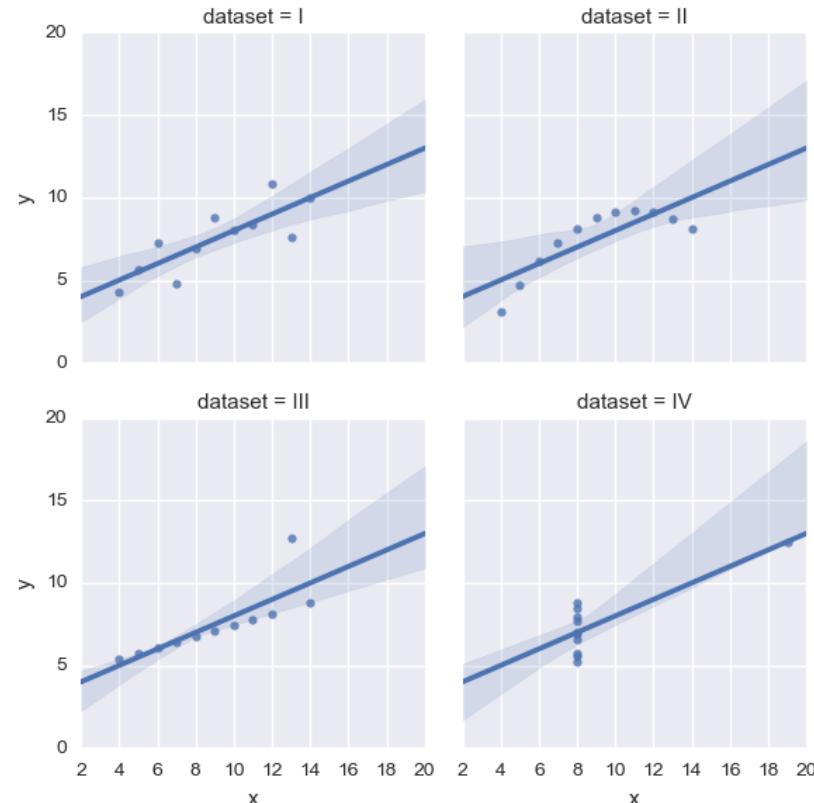
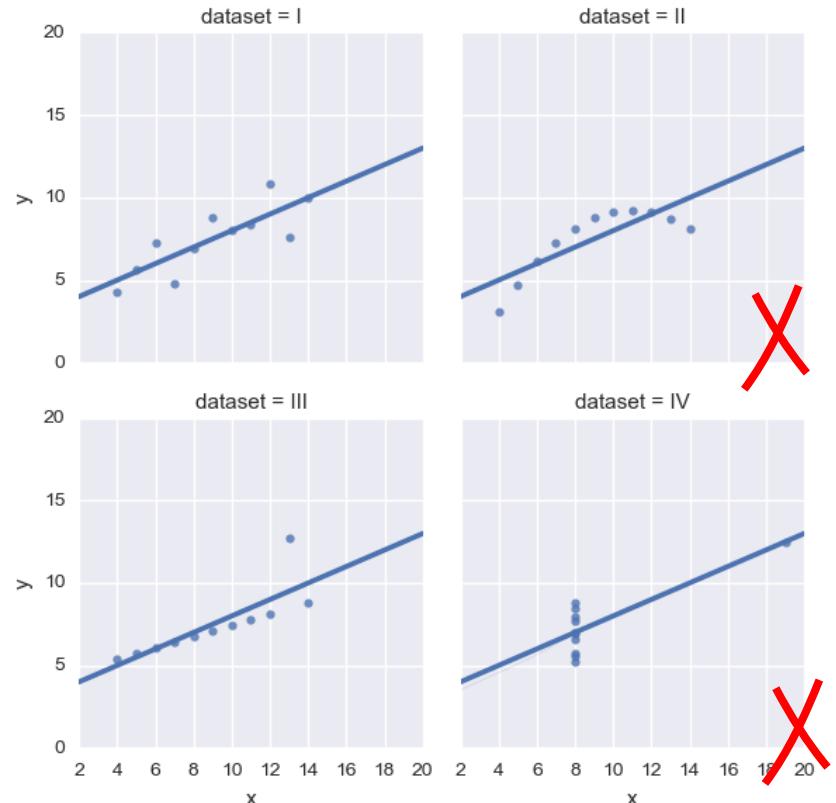
Inappropriate uses of regression plots



```
random.seed(7)  
xs = rand(20)  
ys = rand(20)
```



Anscombe's quartet



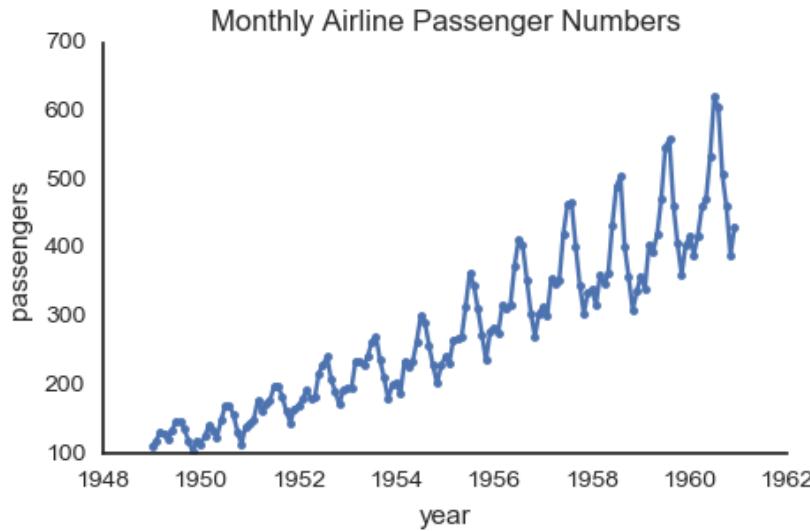
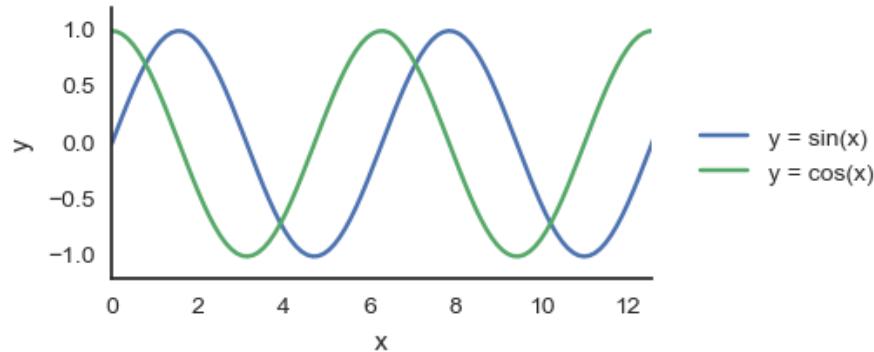
Line graphs

Line graphs are appropriate for plotting one variable as a function of another **continuous ordered** variable.

Often x is **time** and y is some quantity we wish to study over time. This is called a **time series**.

Data points are connected by lines. It's good practice to include **markers** for the data points.

```
air = pd.read_csv('data/AirPassengers.csv')
figure(figsize=(5,3))
plot(air.time, air.AirPassengers, marker='.')
sns.despine()
xlabel('year')
ylabel('passengers')
title('Monthly Airline Passenger Numbers')
```



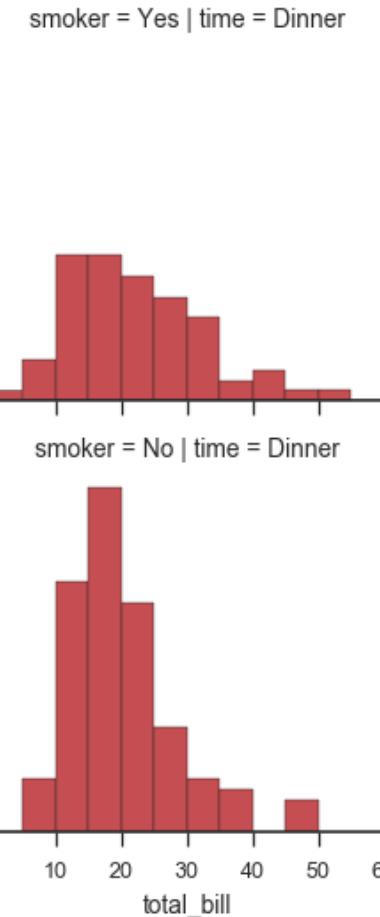
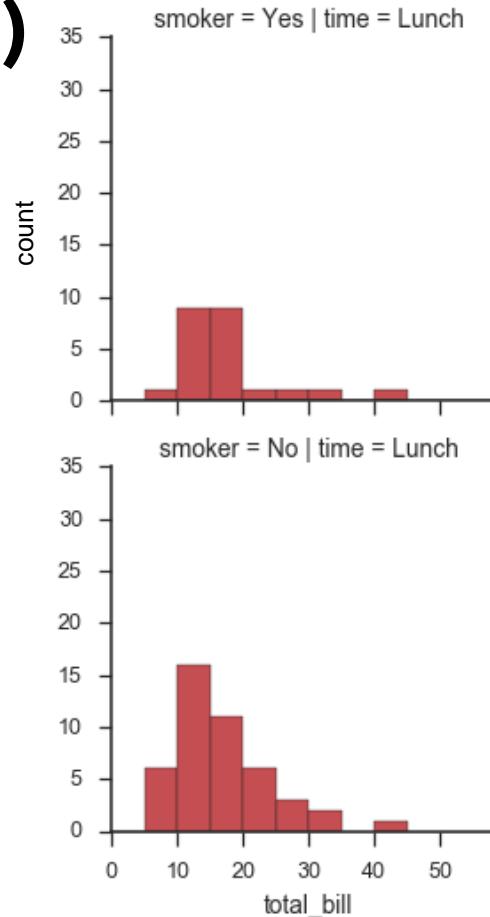
Facet grids (lattice plots)

Visualize **multivariate relationships** by displaying relationship between one or two variables **conditioned on other nominal variables**.

I.e. group data by facets (nominal variables) and plot groups on a grid.

If variable being conditioned upon is not nominal, it can be quantized into intervals.

```
g = sns.FacetGrid(  
    tips, col="time", row="smoker")  
g.map(plt.hist, "total_bill", bins=bins)
```



Topics

- Visualizing the shape and distribution of data
- Visualizing relationships between attributes
- Comparing quantities
- Visualizing composition and part-whole relationships
- Design principles and visualization faux pas

Comparing quantities

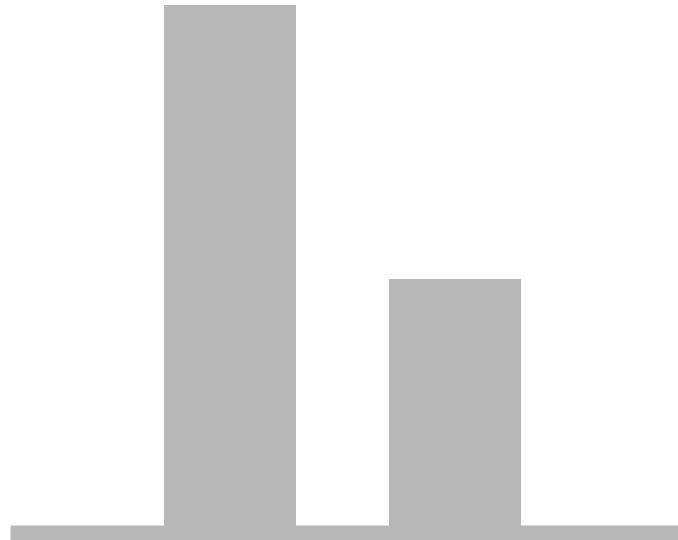
Compare quantities conditioned on another variable

Typical quantities:

- Mean
- Variance

E.g. accuracy given algorithm, height given gender, etc.

Time series: quantity given time.



Bar charts

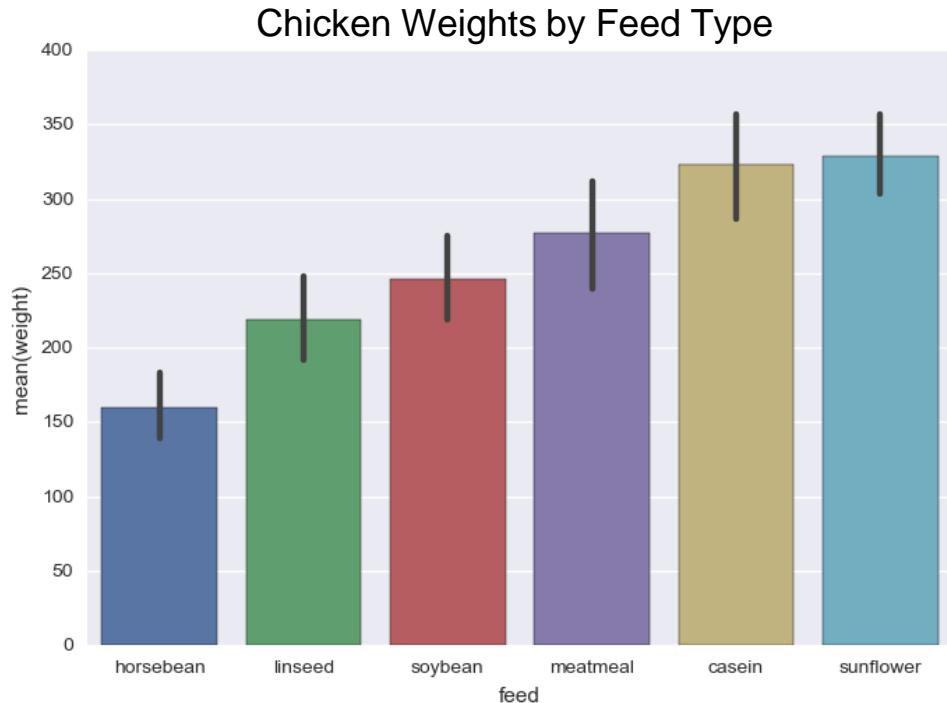
Use for comparing quantities conditioned on a categorical variable.

When quantities represent averages, confidence intervals can be also be plotted.

Seaborn uses bootstrapping to estimate confidence intervals.

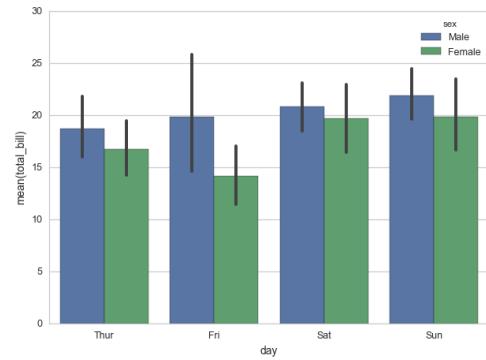
Bars be plotted horizontally or vertically.
Horizontal bars better when there are many categories.

```
sns.barplot('feed', 'weight', data=chicken)
```



Bars chart tips

- If it makes sense to **interpolate** between the bars (quantitative variables), use a line chart. If not, use a bar chart.
- Since categorical variables have no inherent order, **sort** categories by bar height.
- When there are only a few variables and confidence intervals don't matter, prefer a small table to a bar chart.
- When conditioning on another variable, color can be used and bars can be grouped



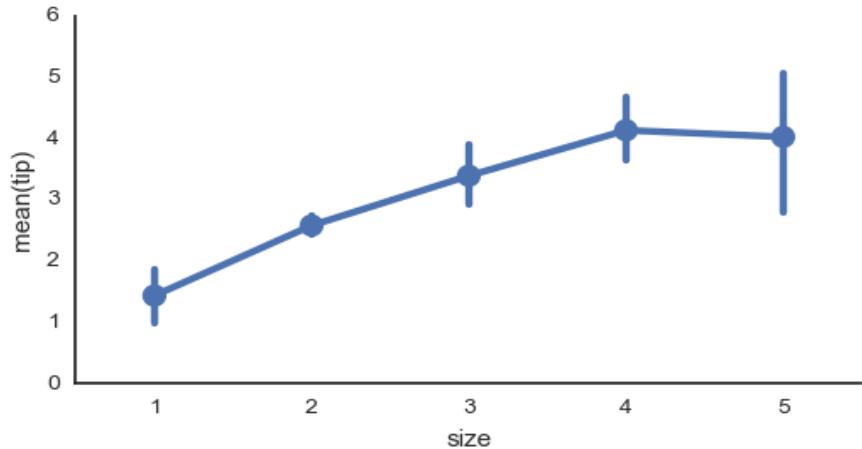
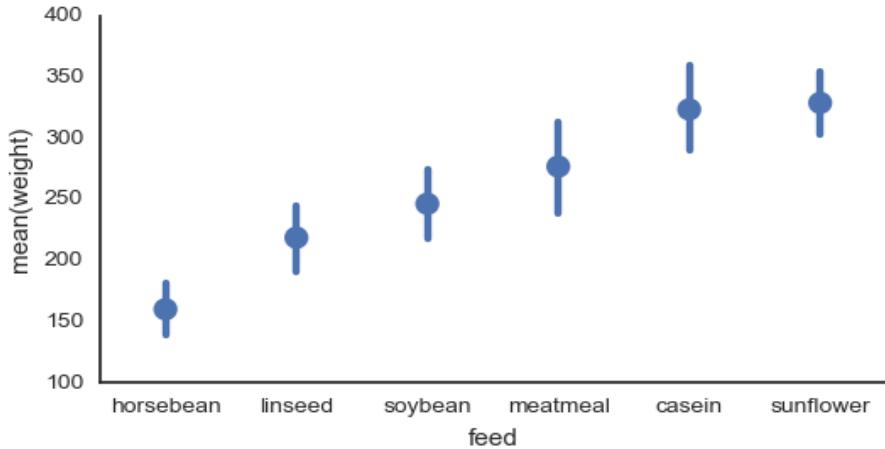
Dot/point plots

Similar use to bar charts. Save ink!

More appropriate when there is no meaningful zero (interval measurements)

Can include a line when it makes sense to interpolate between values on x-axis.

```
sns.pointplot(  
    'size', 'tip',  
    data=tips, order=range(1,6))
```



Topics

- Visualizing the shape and distribution of data
- Visualizing relationships between attributes
- Comparing quantities
- Visualizing composition and part-whole relationships
- Design principles and visualization faux pas

Visualizing composition

Visualizing composition of some "whole" as a set of parts.

Very often **percentages**.

Several widely used types of plots, but often it's better just to use a bar plot!

Pie charts

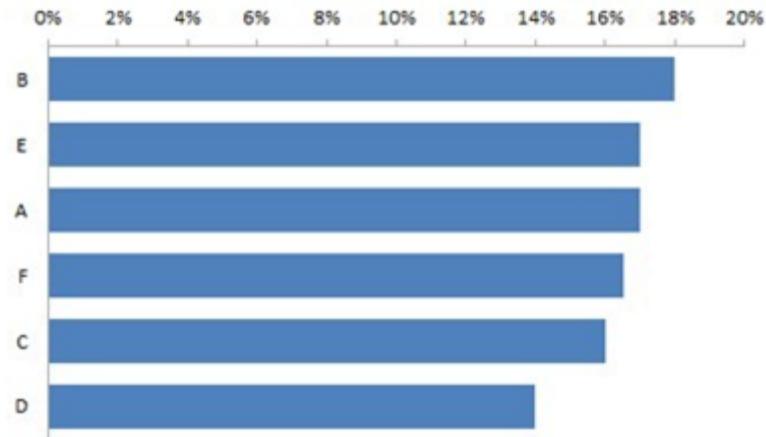
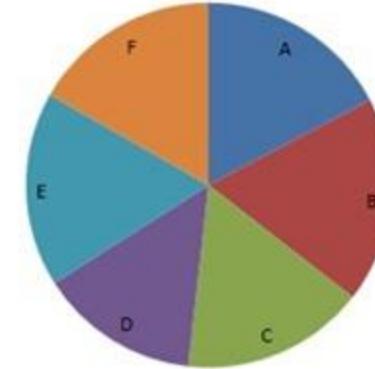
A pie chart is a statistical graphic that illustrates numerical proportion using the arc lengths of the slices.

Widely used in business world and mass media.

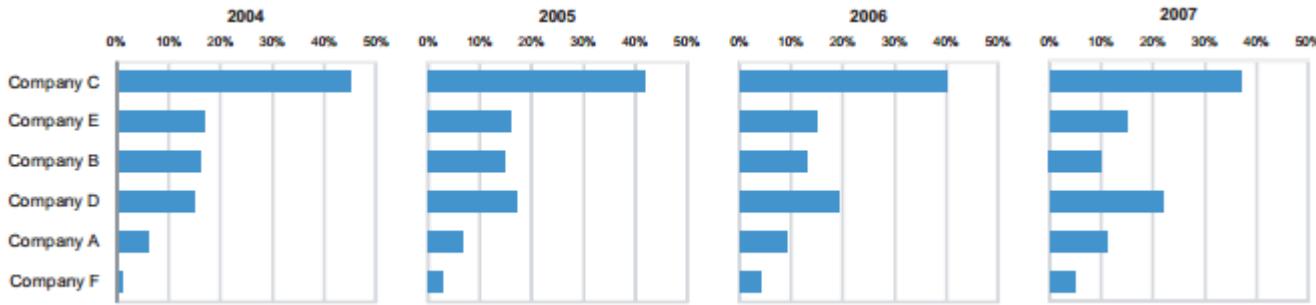
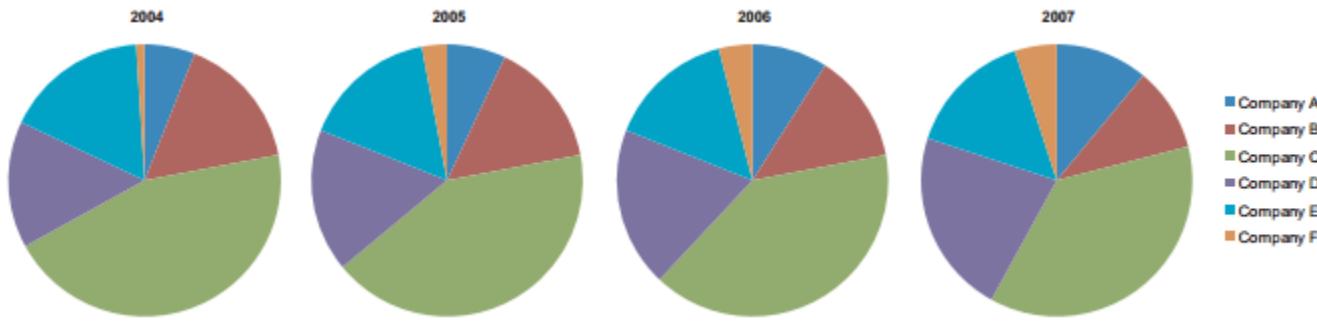
Disadvantages:

- Can only show small number of values
- Take up more space than bar charts
- Humans are poor at comparing arc lengths and slice areas

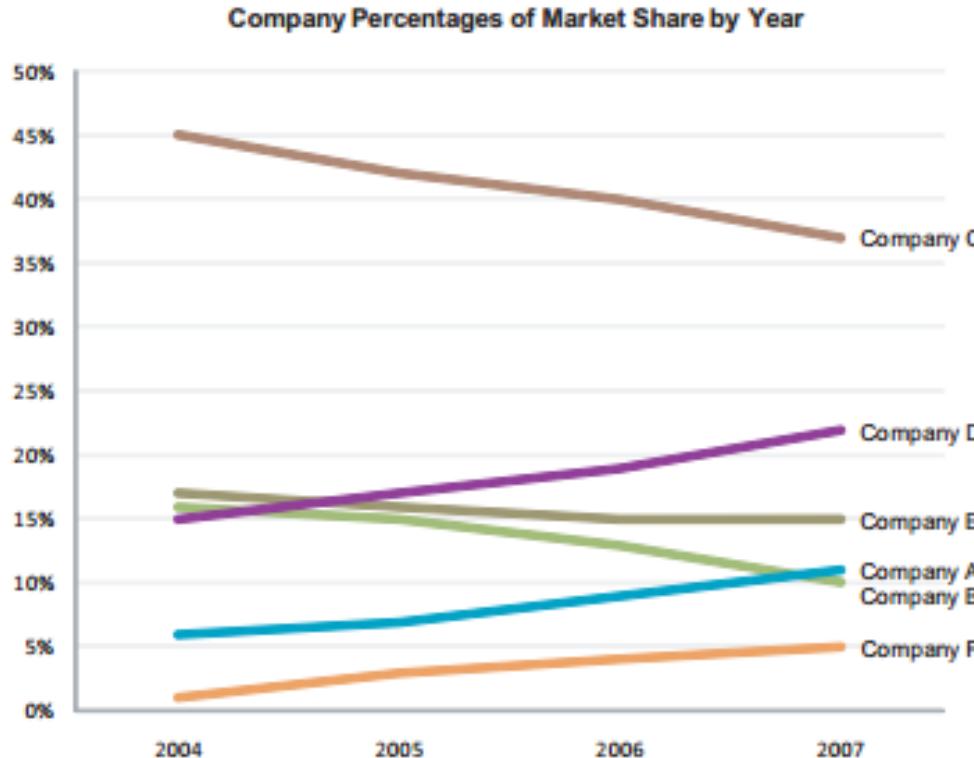
Prefer a sorted bar charts to pie charts!



Avoid pie charts



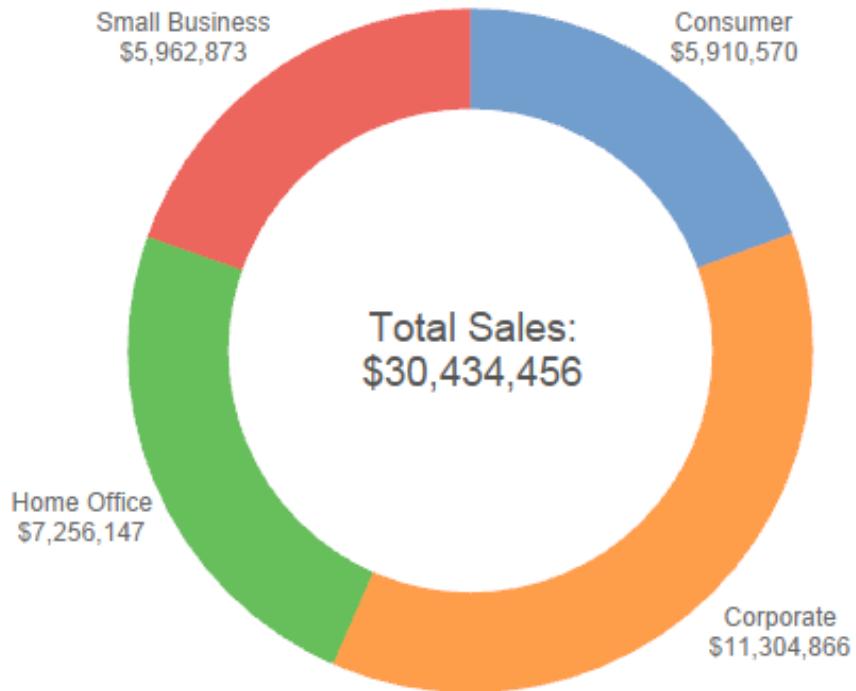
Avoid pie charts



What about donuts?

Look cooler, but have the same problems as pie charts. Avoid.

Prefer bar charts to pies and donuts!



Stacked area charts

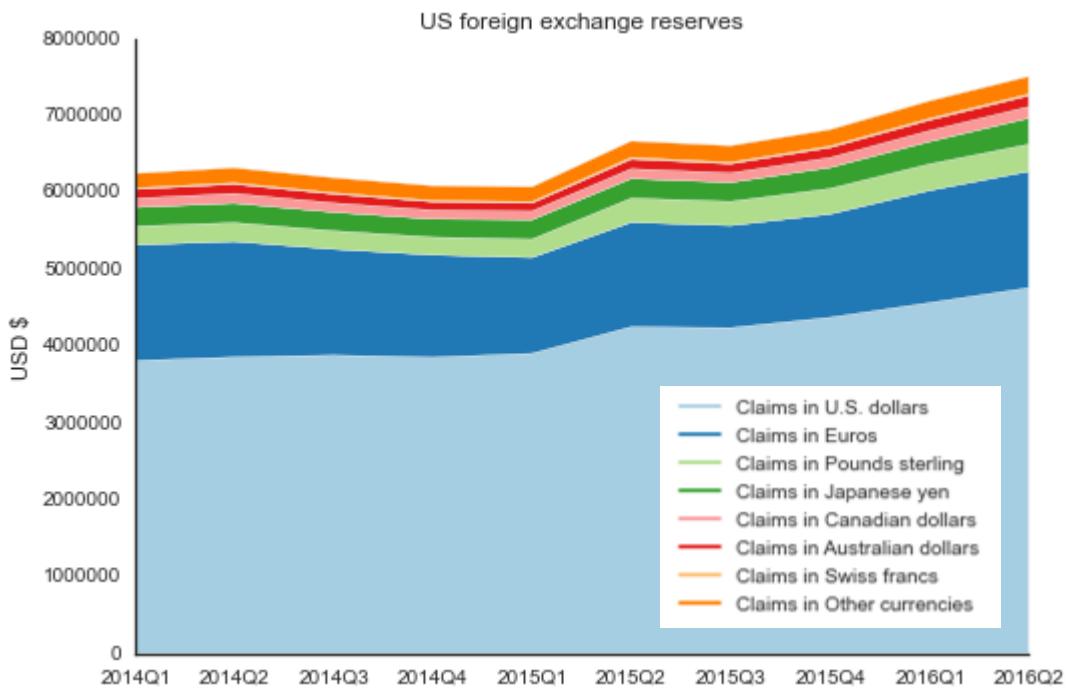
Useful for displaying composition conditioned on time (time series composition)

```
sns.set_style('white')

# trick to get the legend to work
colors = sns.color_palette(
    "Paired", len(fields))
for field, color in zip(fields, colors):
    plot([], [], label=field, color=color)

stackplot(range(10), data, colors=colors)
legend()

title('US foreign exchange reserves')
xticks(range(10), currency.index[1:])
ylabel('USD $')
sns.despine()
```



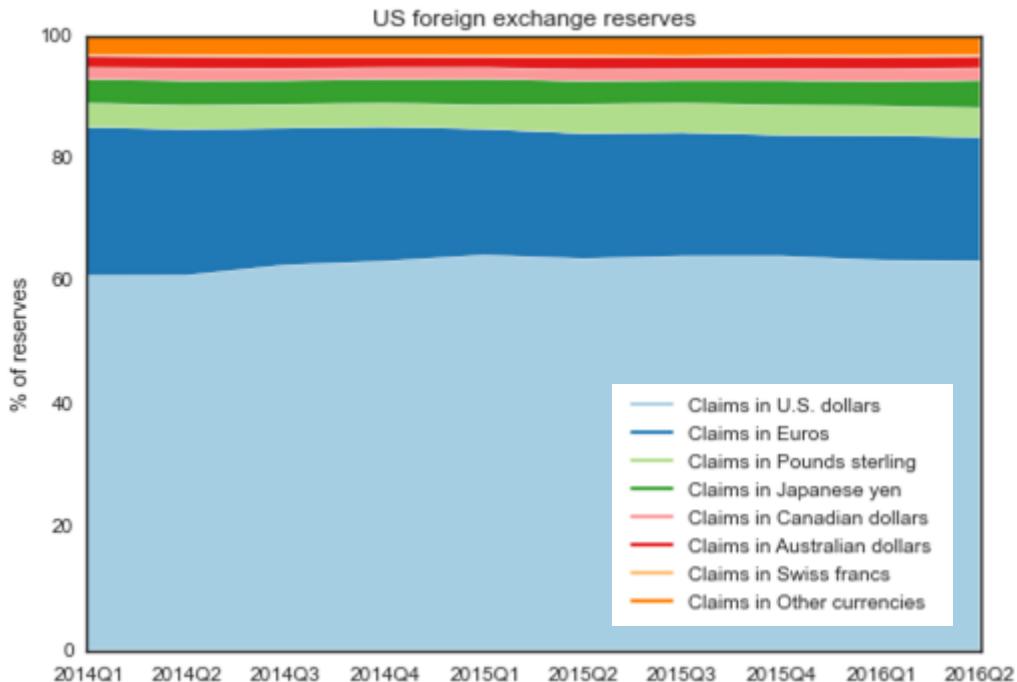
Stacked area charts

To display percentages, simply convert the absolute values to percentages

```
percent = data / data.sum(axis=0) * 100
```

Which version is best to use depends on what you are trying to show.

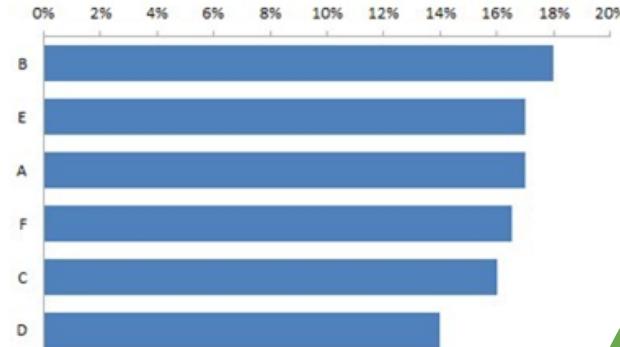
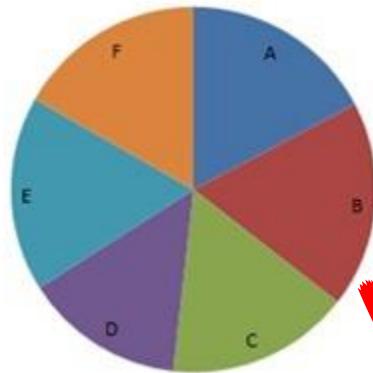
Percentages show less information but highlight changes in composition.



Summary of visualizing composition

Studies show human beings are not very good at judging areas.

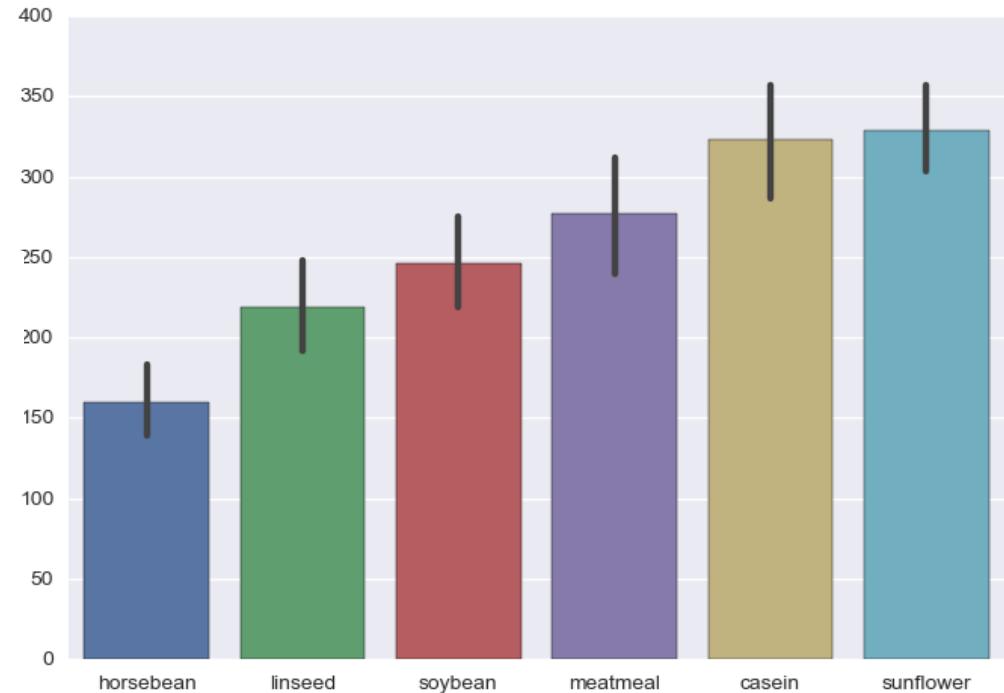
In most cases, just use a bar or line chart to show composition!



Topics

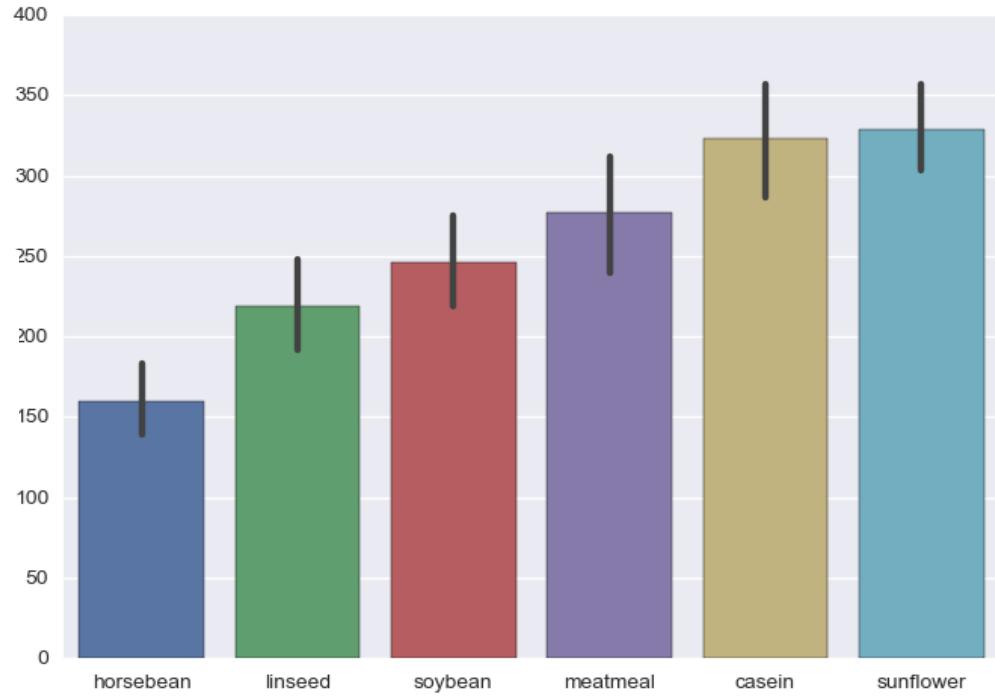
- Visualizing the shape and distribution of data
- Visualizing relationships between attributes
- Comparing quantities
- Visualizing composition and part-whole relationships
- Design principles and visualization faux pas

Faux pas

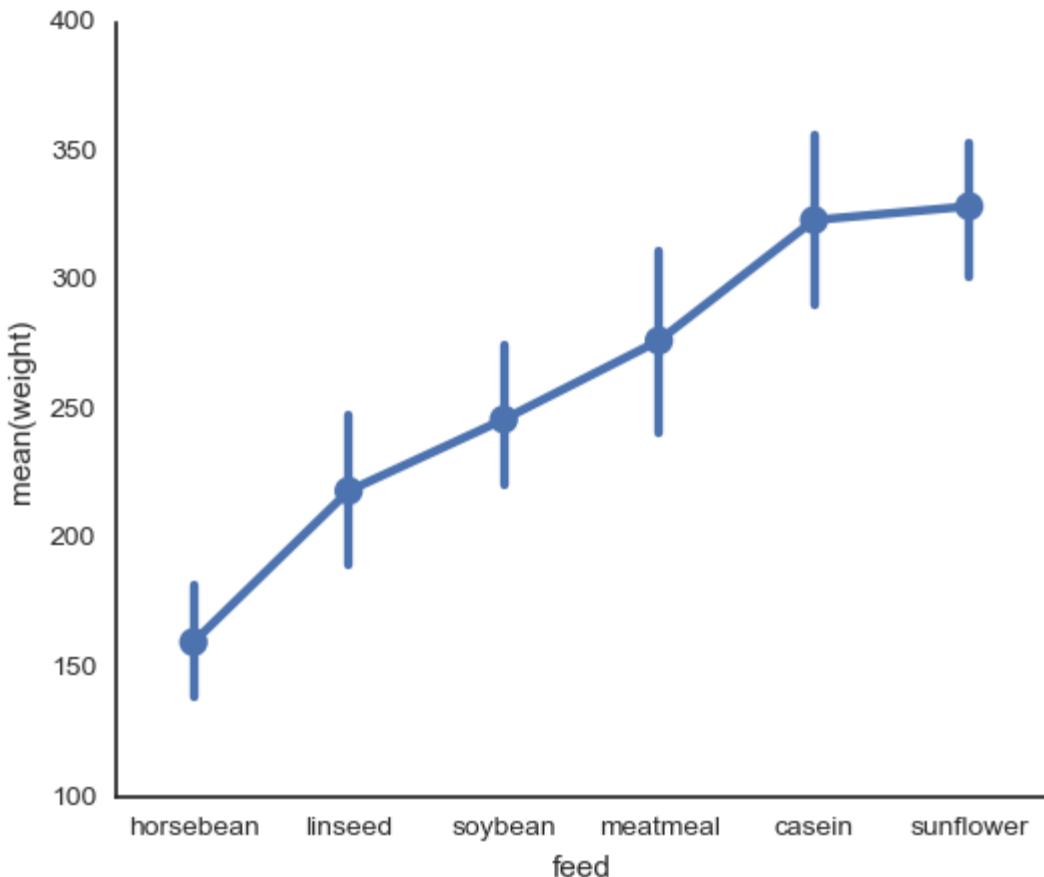


Faux pas

Forgetting to label the axes (when it's important)

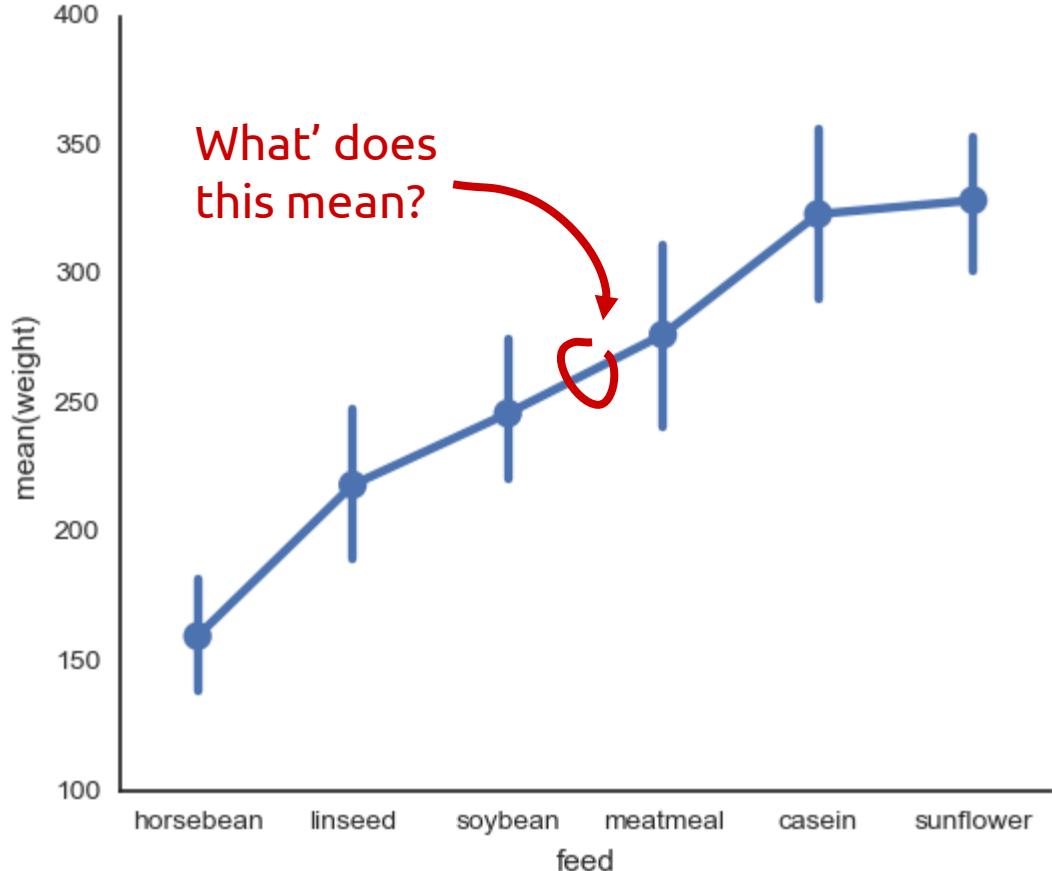


Faux pas



Faux pas

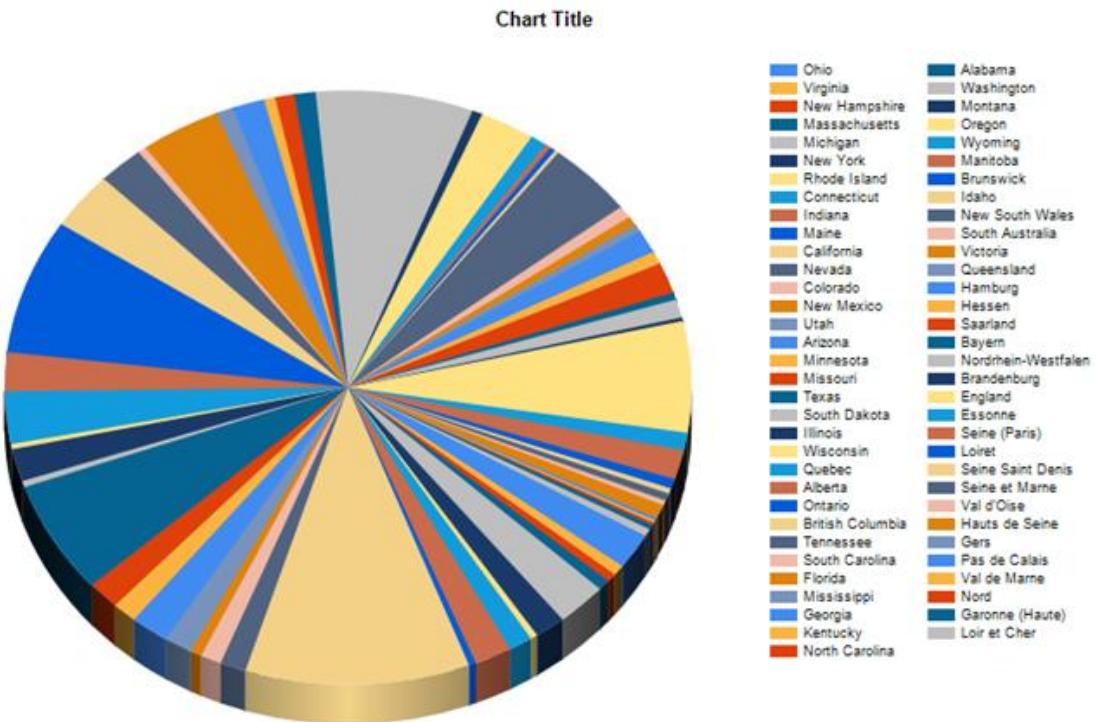
Line plot where bar is more appropriate



Faux pas

Pie charts with too many sectors

- 2-4 sectors max, only when differences are large.
- Or avoid altogether!



General design principles

Seen lots of statistical graphics plots

Easy to design ugly, misleading visualizations even using standard plots

Several useful principles for designing visualizations:

- Tufte principles
 - Graphical excellence
 - Graphical integrity
 - Data-ink ratio
 - Chartjunk
- Methods for representing attributes visually
- Avoiding overplotting

Tufte's principles

Graphical excellence

Excellence in statistical graphics consists of complex ideas communicated with clarity, precision, and efficiency

- **show the data**
- induce viewer to think about substance, rather than form
- **avoid distorting the data**
- present many numbers in small space
- make large datasets coherent
- reveal data on several levels of detail



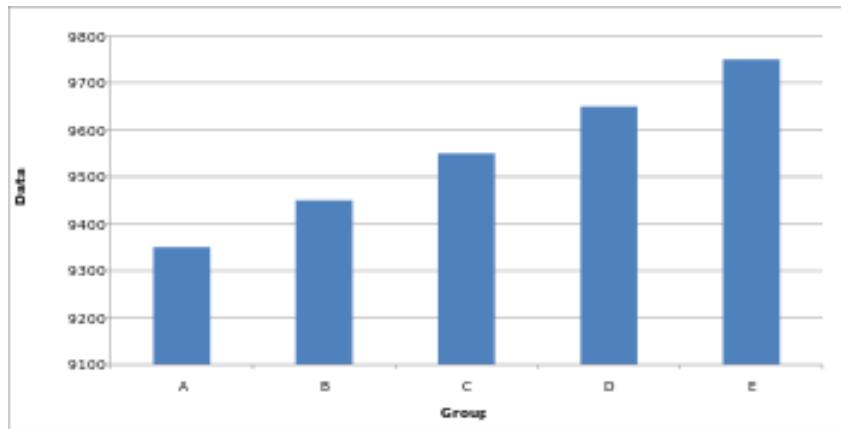
Graphical integrity

Data graphics are no different to words: any means of communication can be used to deceive

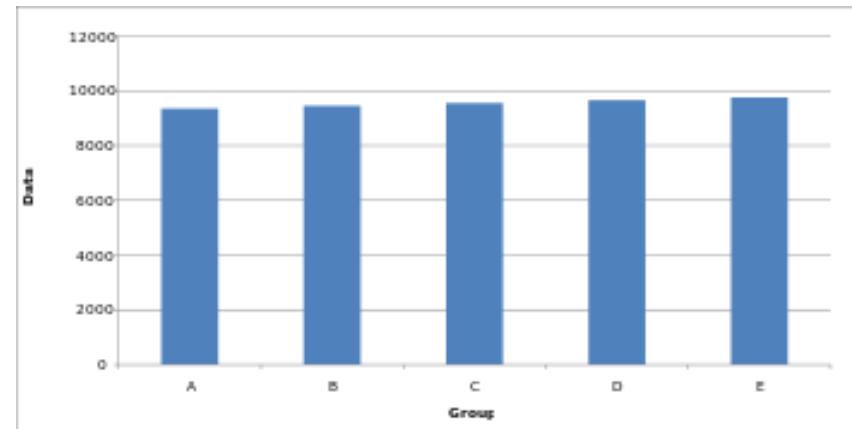
- visual representation should be consistent with numerical

Integrity: scale distortions

Amazing improvement!

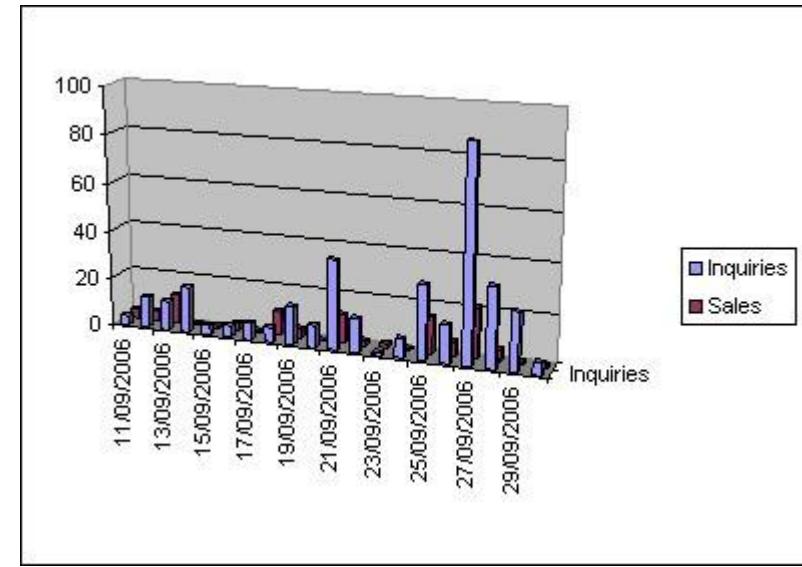
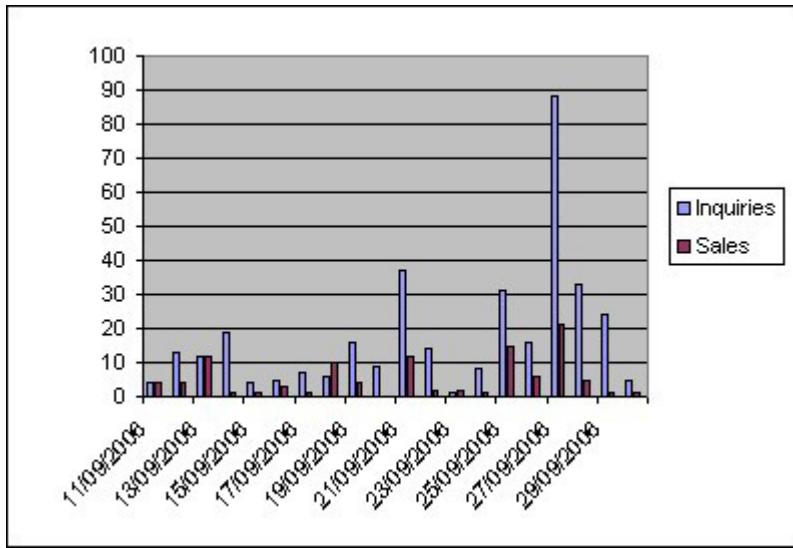


Not so much...

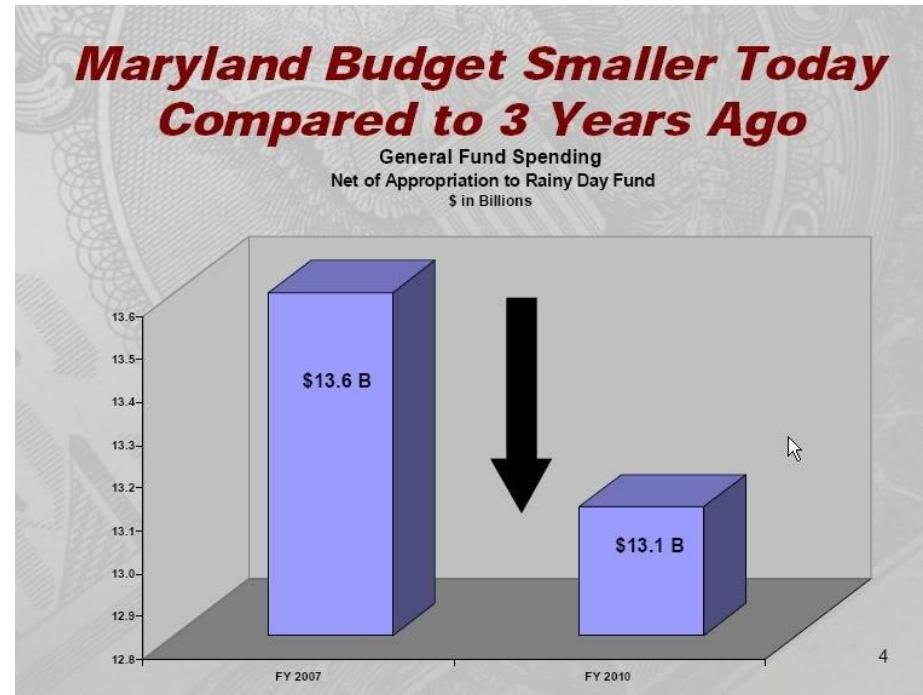
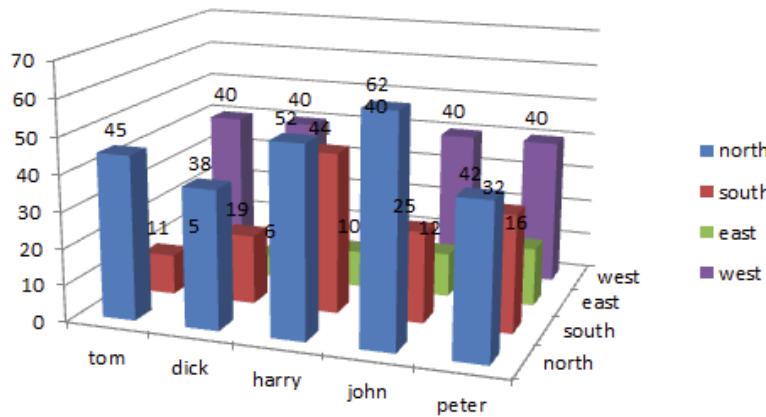


Many more examples:
https://en.wikipedia.org/wiki/Misleading_graph

Chartjunk



Chartjunk



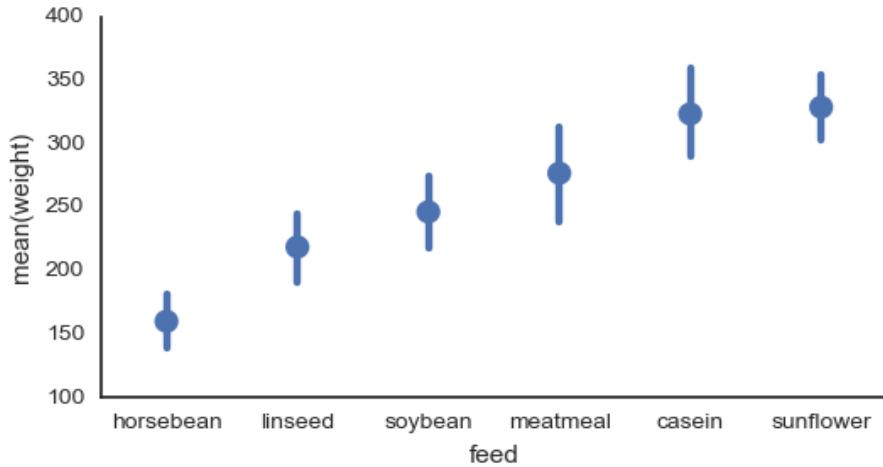
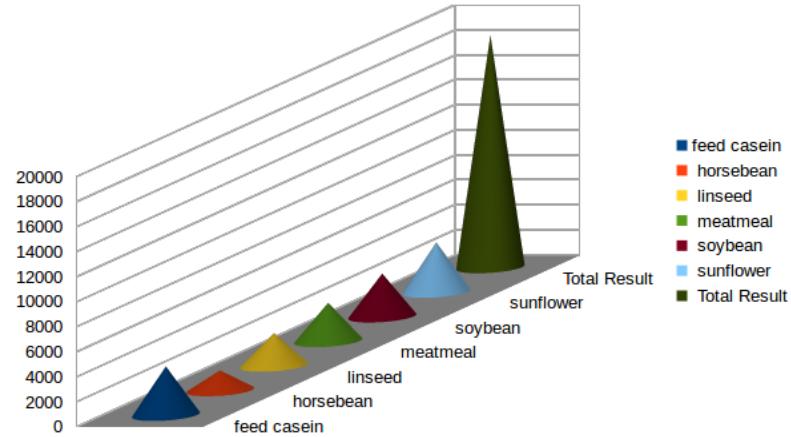
Tufte's data-ink ratio

The **data-ink**: the non-erasable core of the graphic

Maximize the data-ink ratio (within reason).

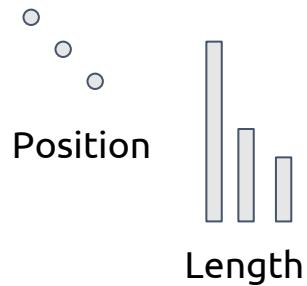
Two principles:

1. Increase data ink (add more data)
2. Erase non-data ink



Representing attributes visually

Quantitative and ordinal variables



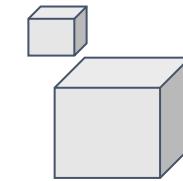
Area



Luminance



Saturation



Volume



Curvature



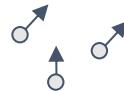
Most effective

Least effective



Spatial region
(e.g. x-axis)

Hue



Motion



Shape

Categorical variables

Overplotting

Reduce size of points

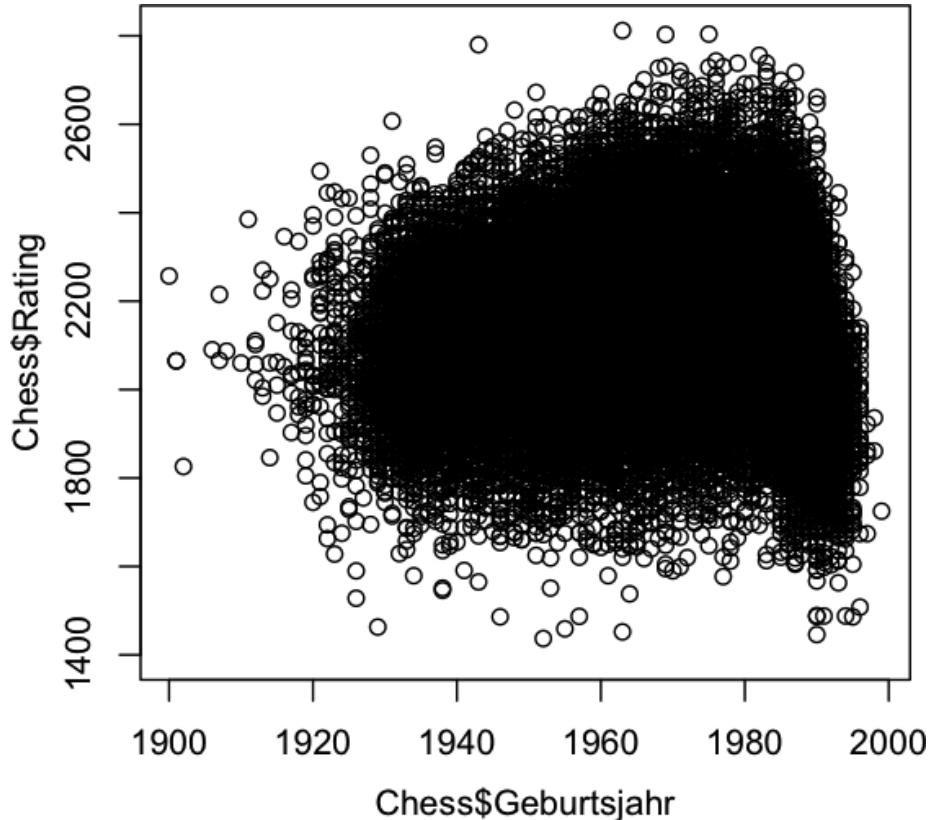
Increase transparency

Jitter

Use a sample

Split data by facets

https://www.perceptualedge.com/articles/visual_business_intelligence/overplotting_in_graphs.pdf



Non-statistical visualizations

Statistical graphics great out of the box tools for fast data exploration

Various other types of visualizations to consider for communication and presentation of results:

- Custom graphics
- Animations
- Infographics
- Interactive visualizations

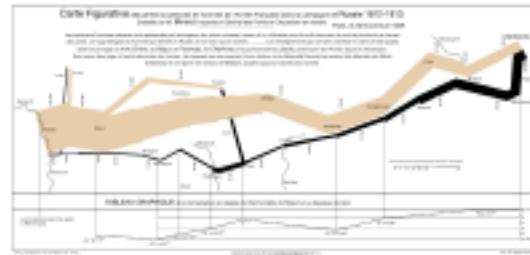


Infographics

Graphic visual representations of information, data or knowledge intended to present information quickly and clearly.

They can improve cognition by using graphics to enhance the human visual system's ability to see patterns and trends.

Can be extremely effective but be wary of distorting the data and use of excessive design junk.



Interactive visualizations

Interactive statistical graphics

[Bokeh](#) can be used to create interactive statistical graphics:

- Uses browser technology backend (Javascript, SVG, canvas)
- Interaction is a good tool for data exploration
- Publication export more problematic

Interactive custom graphics

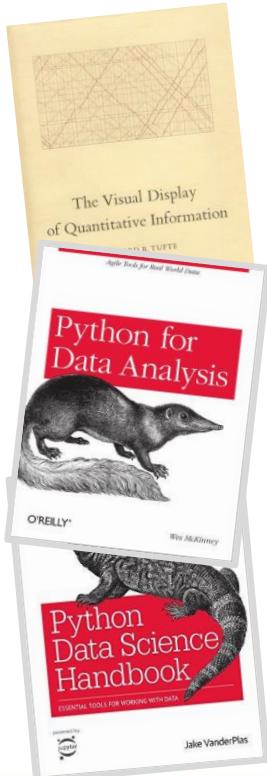
Create powerful novel visualizations

- [d3js](#) (Javascript)
- [Processing](#) (Java)

Can be time consuming to design these visualizations.

Questions?

Further reading



Edward R. Tufte, **The Visual Display of Quantitative Information** ([DCU library](#))

Wes McKinney, **Python for Data Analysis** ([DCU library](#))

- Chapter 8: Plotting and visualization
- Chapter 9: Data aggregation and group operations

Jake Vanderplas, **Python for Data Science Handbook**

- Chapter 4: Visualization with Matplotlib

Seaborn documentation:

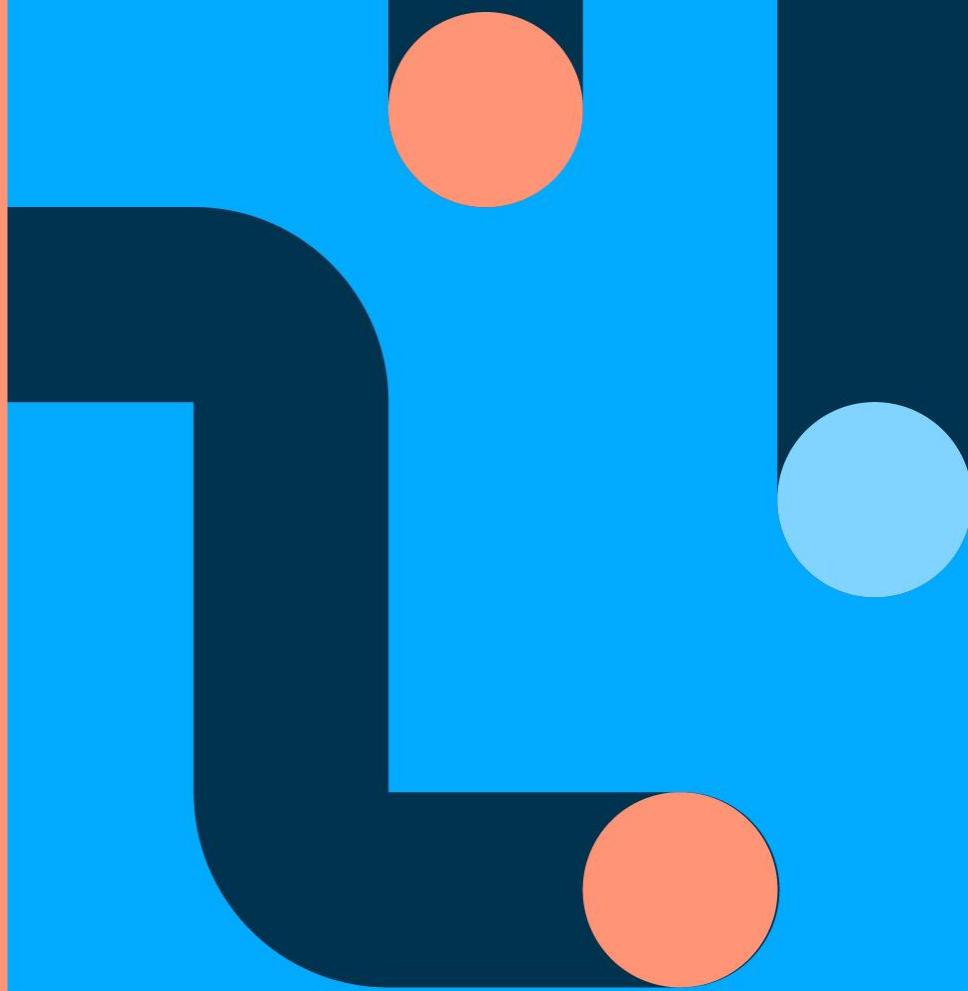
<http://seaborn.pydata.org/index.html>

EEN1083/EE N1085

Data analysis and machine learning I

Ali Intizar

Semester 1
2024/2025



Supervised Learning



Ollscoil Chathair
Bhaile Átha Cliath
Dublin City University

Outline

1. Introduction
2. Nearest neighbours
3. Bias and variance
4. Model selection
5. Linear and quadratic discriminant analysis



Introduction

Machine Learning

What is Machine Learning?

“Machine Learning is a subfield of artificial intelligence which develops algorithms that can learn from training data and then make predictions on new unseen data.”

Why Machine Learning?

- Many real-world problems can't be solved analytical.
- Rule based programming is not scalable for large scale unseen data.
- Large amount of data, can we train machines to learn from data?

Types of Machine Learning

Supervised Machine Learning

Supervised learning is a machine learning approach that's defined by its use of labelled data sets. These data sets are designed to train or "supervise" algorithms into classifying data or predicting outcomes accurately. Using labelled inputs and outputs, the model can measure its accuracy and learn over time[1].

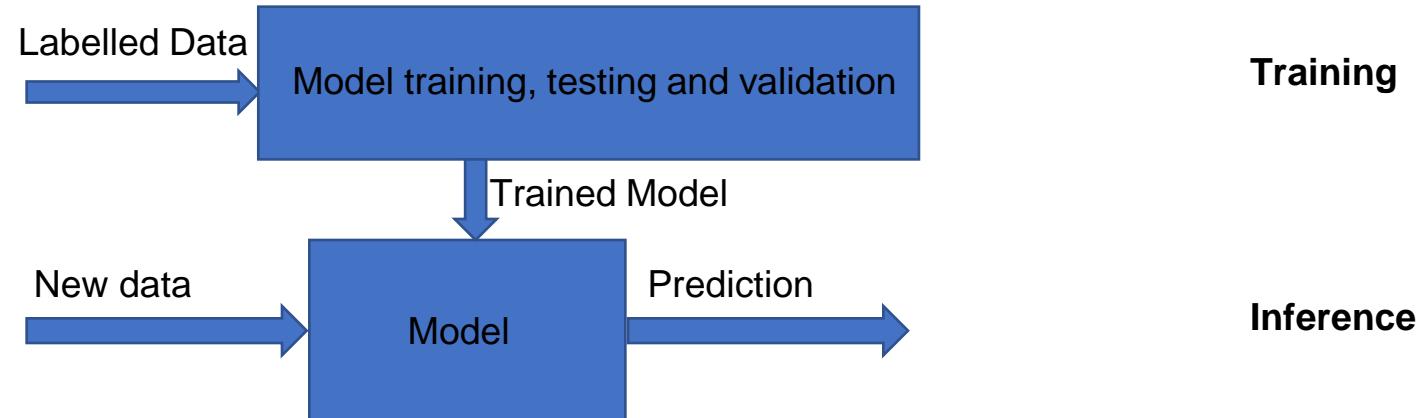
Un-supervised Machine Learning

Unsupervised learning uses machine learning algorithms to analyse and cluster unlabelled data sets. These algorithms discover hidden patterns in data without the need for human intervention (hence, they are "unsupervised").[1]

<https://www.ibm.com/think/topics/supervised-vs-unsupervised-learning>

Machine Learning

- Machine Learning develops a model that takes in a new(unseen) data points and predicts a value for output.
- Model learns from training data to be accurate in its predictions.
- Models can be also seen as parametrised functions and machine learning finds best value for parameters.



The supervised learning problem

Given pairs of training examples $\{(\mathbf{x}_i, y_i)\}_{i=1}^N$ produce a function $\hat{y} = f(\mathbf{x})$ such that **$f(\mathbf{x})$ generalizes well** to previously unseen data.

- The \mathbf{x}_i 's are the features or inputs. Often $\mathbf{x}_i \in \mathbb{R}^D$
- The y_i 's are the targets
- The function $f(\mathbf{x})$ is the decision function
- \hat{y} is the prediction of $f(\mathbf{x})$

Will write \mathbf{x}_i to mean training example i and x_j to mean feature j of some training example. x_{ij} is feature j of example i .

Example 1

Training set consists of emails marked as spam or not spam.

- \mathbf{x}_i is the feature vector corresponding to email i .
- x_{ij} is number of occurrences of word j in email i in the training set. E.g. with a vocabulary size of 1000, the $\mathbf{x}_i \in \mathbb{R}^{1000}$.
- The target variable $y_i \in \{0, 1\}$ depending on whether the email is spam or not spam.

Our objective is to use the examples from training set $\{(\mathbf{x}_i, y_i)\}_{i=1}^N$ to come up with a function $f(\mathbf{x})$ that can map a feature representation of an email to $\{0, 1\}$.

We would also like our function to work well on data outside the training set (unseen data).

Example 1

Training set:

doc	class	meeting	data	casino	won	viagra	join	...	spam?
	x_1	x_2	x_3	x_4	x_5	x_6	x_7	...	y
\mathbf{x}_1	3	0	3	0	0	0	0	...	0
\mathbf{x}_2	0	2	1	0	0	0	0	...	0
\mathbf{x}_3	0	0	3	0	0	5	0	...	1
\mathbf{x}_4	2	0	0	0	0	0	0	...	0
\mathbf{x}_5	1	0	3	2	2	0	2	...	1

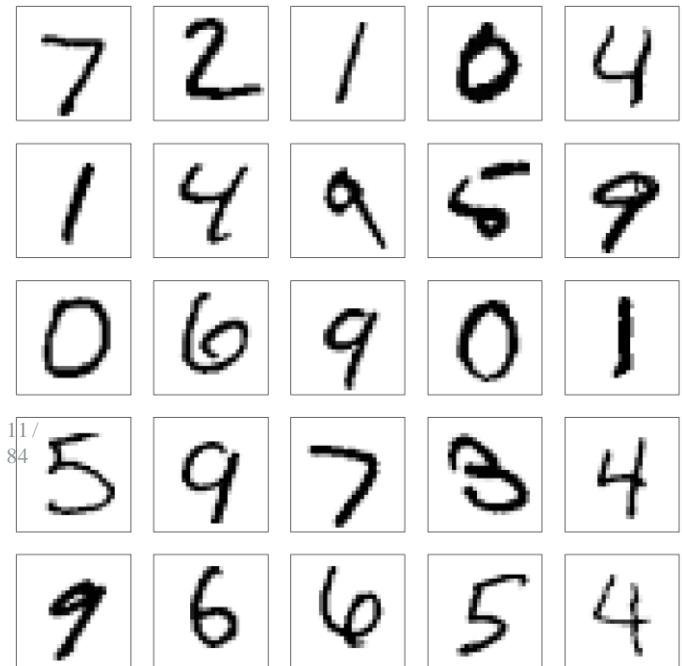
This is an example of a **binary classification** task.

Example 2

Produce a classifier to map from pixels to the digit.

- If images are grayscale and 28×28 pixels in size, then $\mathbf{x}_i \in \mathbb{R}^{784}$
- $y_i \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$

Example of a **multi-class classification** task.



Example 3

Features \mathbf{x} could be:

- Age
- Level of education (secondary school, degree, Ph.D)
- Years of experience
- Industry

Target variable y is salary. E.g. $y_i = 50000$ EUR

Regression problem: target y value is continuous.

$$f : \mathbb{R}^D \rightarrow \mathbb{R}$$

More examples

- feature is time of day, target is light level
- feature is light level, target is time of day
- features are measurements from sensors (temp, humidity, brightness, etc.), target is {rain, no rain}
- features are vectors of image pixels, target is {cat, dog, car, person, ...}
- features are recorded audio fragments, targets are words

Terminology

The training set: $T = \{(\mathbf{x}_i, y_i) : \mathbf{x}_i \in X, y_i \in Y\}$

- \mathbf{x}_i are the inputs. Often vectors of D -dimensional real numbers $\mathbf{x}_i \in \mathbb{R}^D$
- y_i is the output or target variable.
- (\mathbf{x}_i, y_i) is a training example.
- X is the domain (set)
- Y is the range (set)

E.g.

$$\mathbf{x}_1 = [1 \ 3 \ 0 \ 2 \ 8 \ 6]^T$$

$$y_1 = 1$$

Terminology

$$\mathbf{x}_1 = [\begin{array}{cccccc} 1 & 3 & 0 & 2 & 8 & 6 \end{array}]^T$$

$$y_1 = 1$$

The x_j values are known as:

- Features
- Variates
- Attributes
- Predictors

The y values are known as

- Targets
- Outputs
- Covariates

Terminology

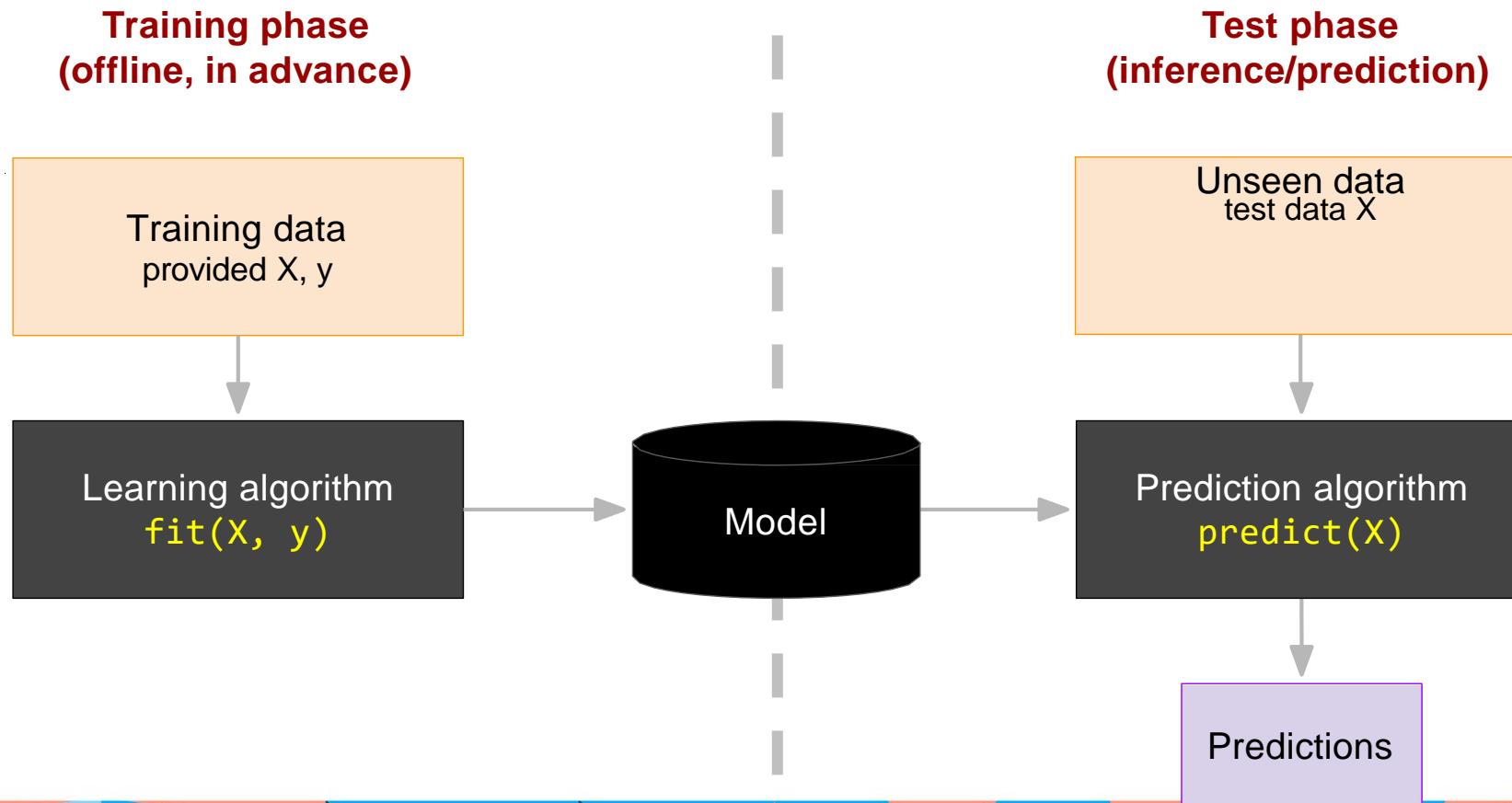
$$\{(\mathbf{x}_i, y_i)\}_{i=1}^N$$

Can stack the \mathbf{x}_i values into a matrix X and y_i values into a vector \mathbf{y} . E.g.

$$X = \begin{bmatrix} \mathbf{x}_1^T \\ \mathbf{x}_2^T \\ \mathbf{x}_3^T \\ \mathbf{x}_4^T \\ \mathbf{x}_5^T \end{bmatrix} = \begin{bmatrix} 2.1 & 3.2 & 4.8 & 0.1 & 0.0 & 2.6 \\ 3.1 & 1.4 & 2.5 & 0.2 & 1.0 & 2.0 \\ 1.0 & 2.3 & 3.2 & 9.3 & 6.4 & 0.3 \\ 2.0 & 5.0 & 3.2 & 1.0 & 6.9 & 9.1 \\ 9.0 & 3.5 & 5.4 & 5.5 & 3.2 & 1.0 \end{bmatrix} \in \mathbb{R}^{N \times D} \quad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \end{bmatrix} \in \{0, 1\}^N$$

- X has N rows, one per example
- X has D columns, one per feature
- \mathbf{y} has N rows

Black box abstraction of supervised learning



Two types of problems

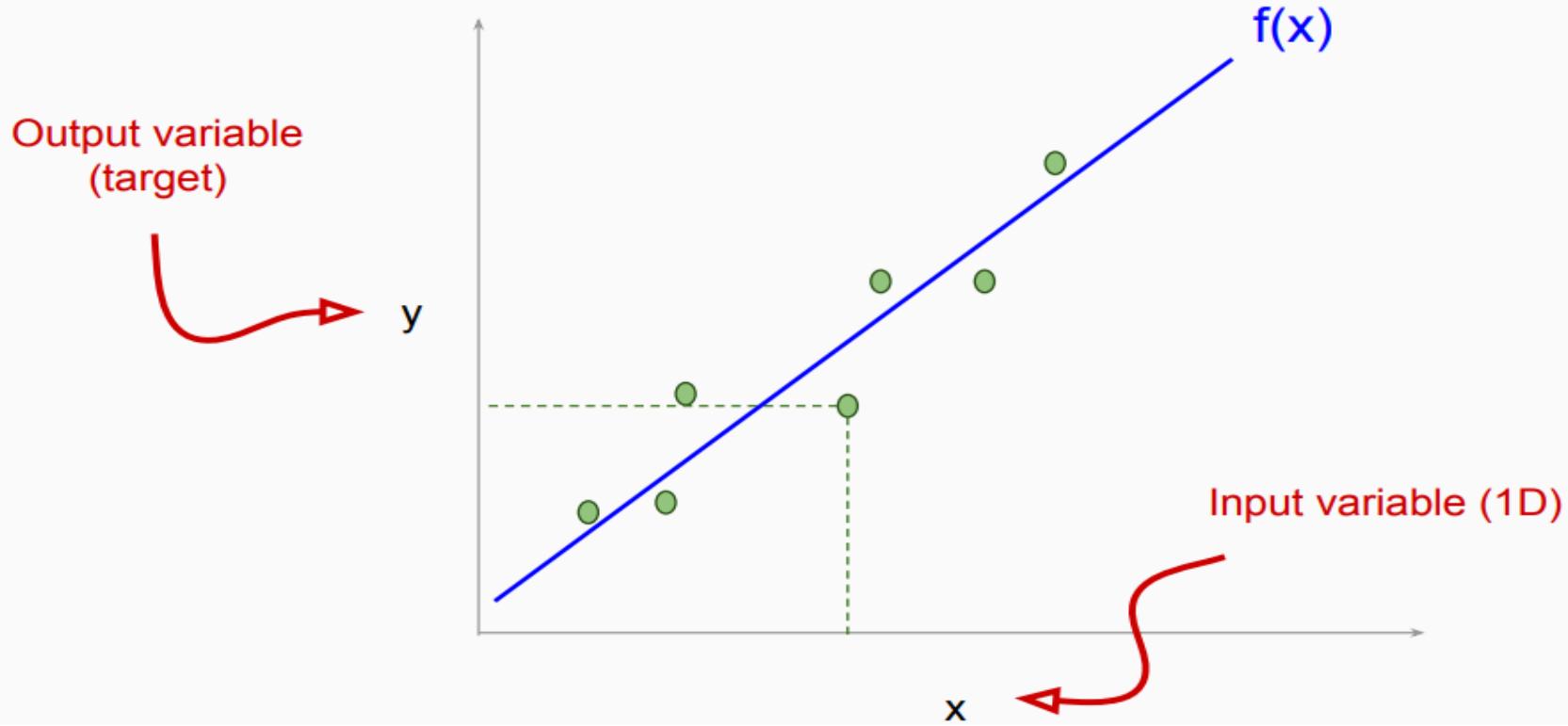
Regression: when the output variable y is a real number ($y \in \mathbb{R}$)

- Predicting house price from location, number of bedrooms, etc.
- Predicting movie earnings from user reviews
- Predicting multiple real numbers (multivariate regression $\mathbf{y} \in \mathbb{R}^N$)

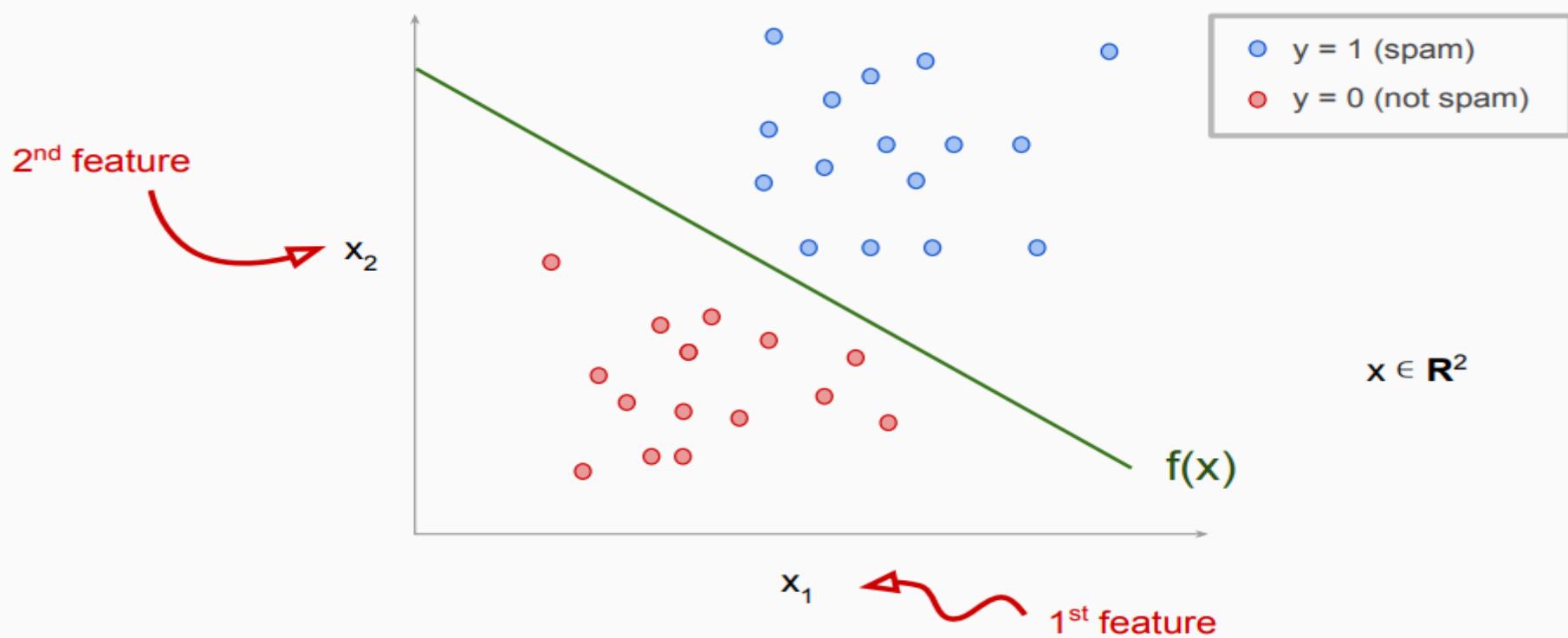
Classification: when the output variable y is a categorical variable

- Is an email spam or non spam (binary classification $y \in \{0, 1\}$)
- Is this an image of a $\{car, house, cat, dog, person\}$ (multiclass classification)
- Is the sentiment of this text $\{positive, negative\}$

Regression



Classification



Measuring error

Given some trained black box predictor $f(\mathbf{x})$ (classifier or regressor) trained on T , we would like some way to measure how accurate it is.

Two things we need:

- An error or accuracy **metric**
- A **dataset** to measure error against.

There are many metrics we can use. Some suitable for regression and others for classification.

Measuring error

Want to measure the distance between the prediction $\hat{y} = f(\mathbf{x})$ and the target y .

Standard metric is **mean squared error** (MSE) (aka Euclidean error).

Squared error for example i :

$$(y - f(\mathbf{x}))^2 = (y - \hat{y})^2$$

MSE for a dataset $D = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$

$$E = \frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2$$

Measuring error

MSE not appropriate for classification, since y is a categorical variable.

One way to measure error is the **misclassification rate**:

$$E = \frac{1}{N} \sum_{i=1}^N \mathbb{1}(y_i \neq \hat{y}_i)$$

where $\mathbb{1}(.)$ is the indicator function.

Note: accuracy is $(1 - \text{misclassification rate})$.

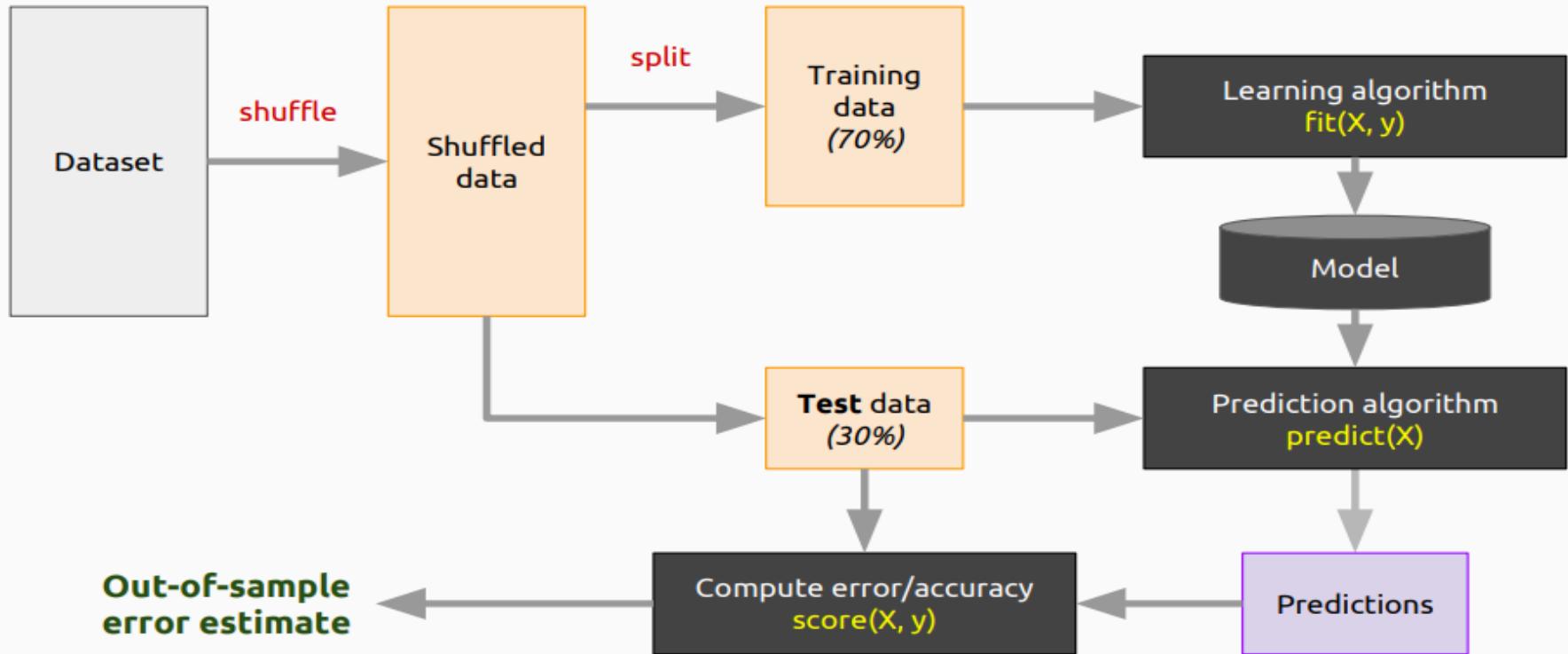
Measuring error

1. **Training error:** error on training set T
2. **Out-of-sample (OOS) error:** expected error on unseen data
3. **Test error:** error on a sample of unseen data (estimate of out of sample error)
4. **Generalization error:** (OOS error - training error)

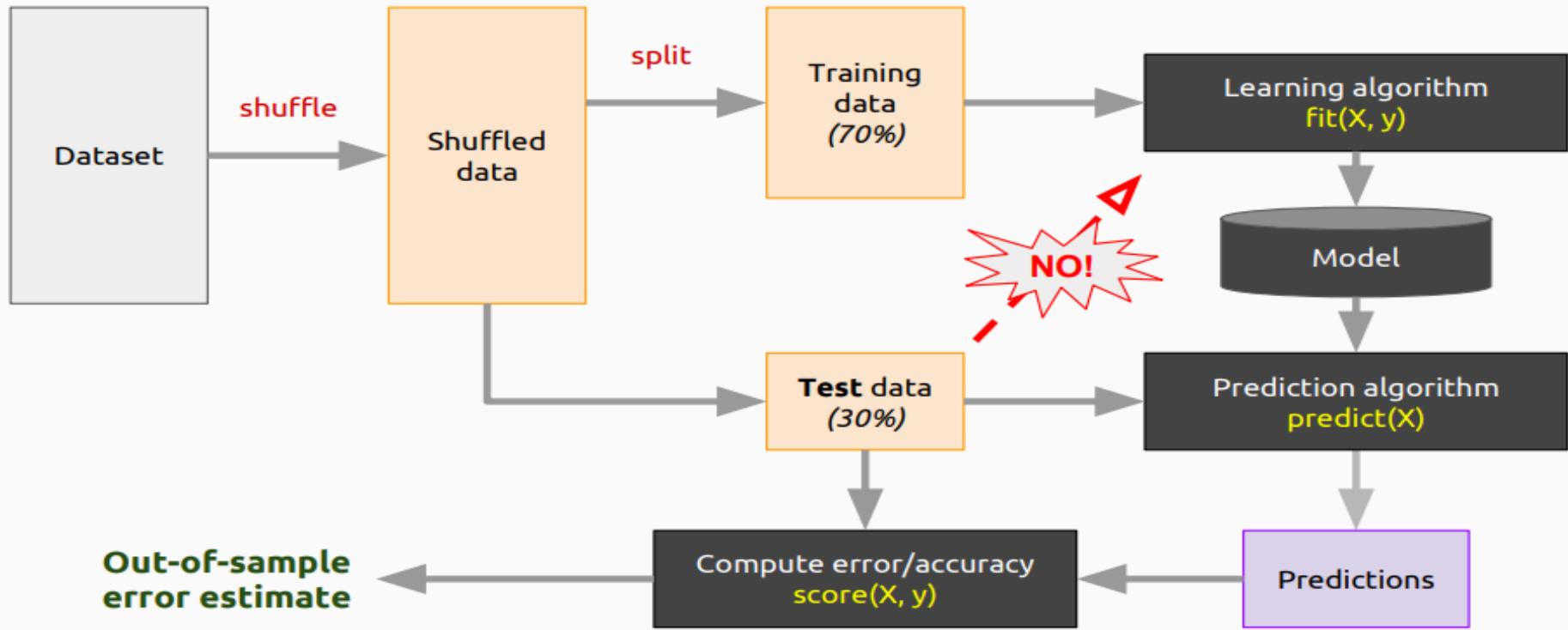
The objective of supervised learning is to **minimize OOS error**. Cannot measure this directly.

Essential to keep aside some test data to estimate the accuracy of your model on unseen data. Can use test error as a proxy for OOS error.

Train test splits



Train test splits



Train test splits

Why can we not just estimate accuracy on using the training set?

- It is easy to get 0 error on the training set.
- How?

Train test splits

Why can we not just estimate accuracy on using the training set?

- It is easy to get 0 error on the training set.
- How? **Just memorize the whole training set.**

Why is it important to shuffle the training data before splitting?

- Ensure the training set is **unbiased**
- Exception: time series data. (Why?)

How much test data do we need?

- Enough so that the estimate of OOS error is accurate.
- Central limit theorem. Variance of estimate goes to zero as $n \rightarrow \infty$
- Trade-off: want to keep as much data for training but still have enough to have low variance estimate of OOS.
- Rule of thumb: keep 10 – 30% of data for test.

Why not use some of the test data to train?

- You have no way to estimate OOS error if you do
- Your model could easily **overfit** the test data and have poor generalization, you have no way of knowing without test data
- Model may fail in production

Best practices

Split your dataset into train and test at the very start

- Usually good practice to shuffle data (exception: time series)

Do not look at test data (data snooping)!

- Lock it away at the start to prevent contamination

NB: Never ever train on the test data!

- You have no way to estimate error if you do
- Model may fail in production

Nearest neighbours

Nearest neighbours

Simplest possible algorithm for “learning” from data. Learning by rote memorization.

Idea: simply memorize all the data.

Prediction: Find the **nearest** example in the dataset and simply output the memorized target value for that example.

Nearest neighbour classification

Learning algorithm (.fit)

Memorize the entire training set

$$T = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$$

Prediction algorithm (.predict)

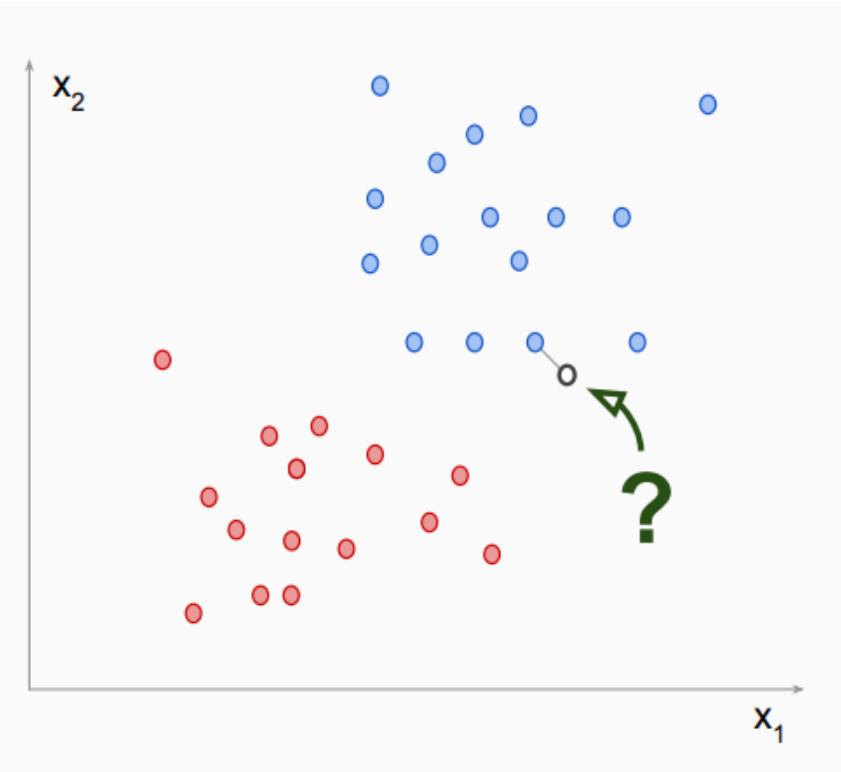
Find nearest \mathbf{x}_i in training set and output
the corresponding y value

$$f(\mathbf{x}) = \arg \min_{y_i \in T} D(\mathbf{x}, \mathbf{x}_i)$$

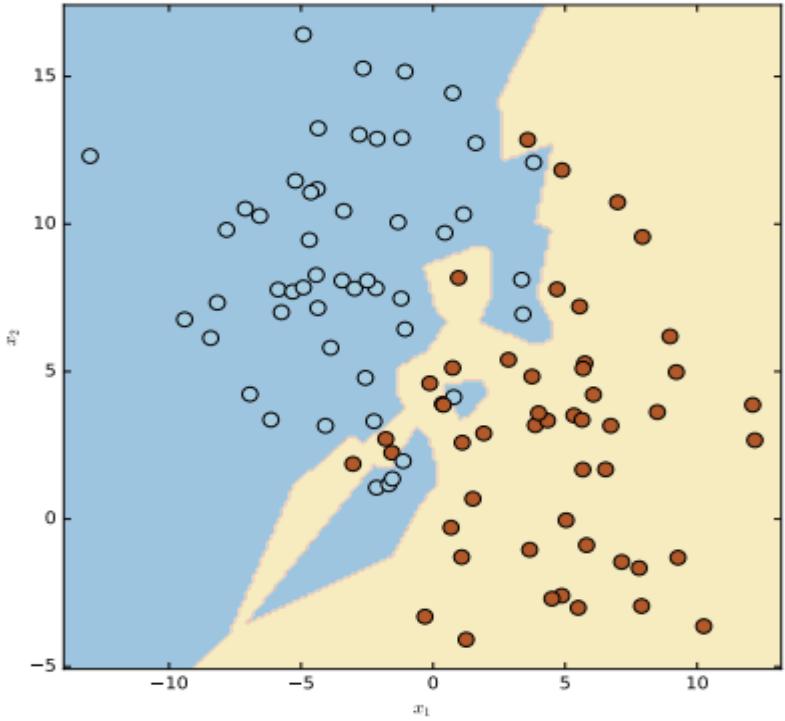
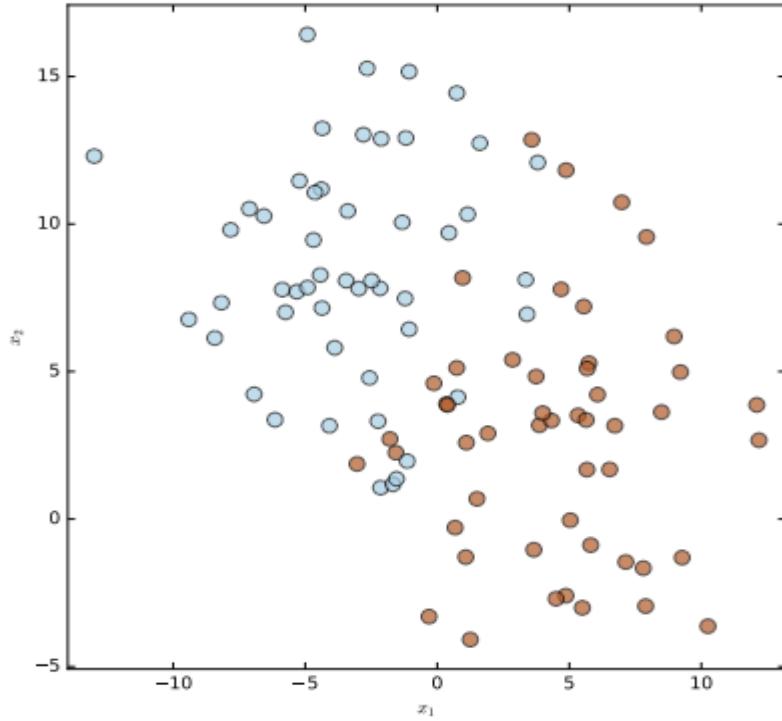
where D is a distance measure. E.g.

Euclidean distance:

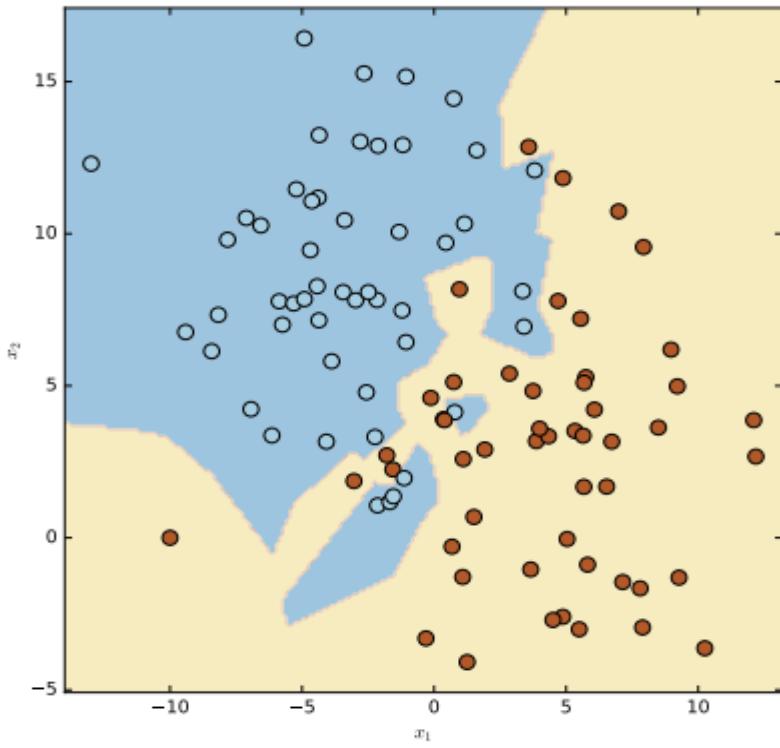
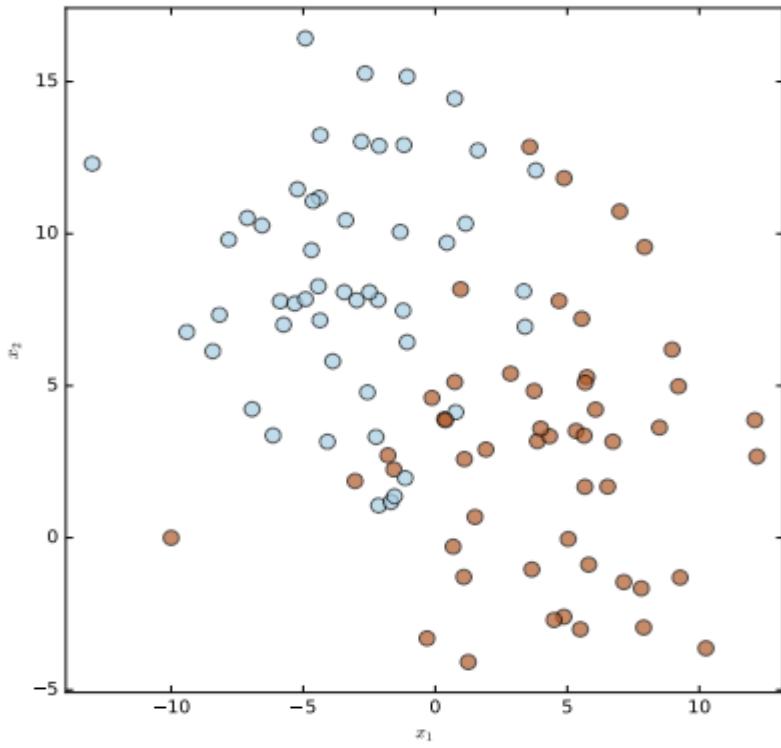
$$D(\mathbf{x}, \mathbf{z}) = \|\mathbf{x} - \mathbf{z}\|_2$$



Nearest neighbour classification

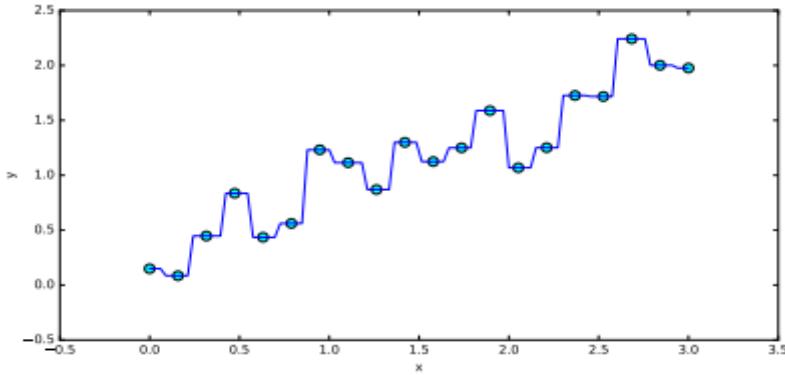
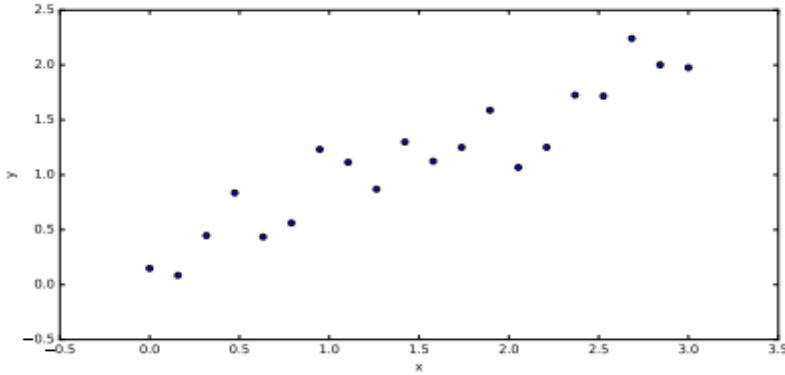


Nearest neighbour classification



Nearest neighbour regression

- Data generated from $y = 0.8x + \text{noise}$
- Nearest neighbour regression fits data exactly.
- Fit function is very jagged
- Q: generalization outside of range of training data?



Advantages of nearest neighbours

Very fast learning algorithm!

- Just store all the training examples.
- Implementation is trivial.

Low classifier bias.

- Is able to always fit the training data exactly.

Can use any distance (or similarity) function. E.g. for bag of words representations of text documents, we can use *cosine similarity*.

$$\cos \theta = \frac{\mathbf{x}^T \mathbf{z}}{\|\mathbf{x}\| \|\mathbf{z}\|}$$

Problems with nearest neighbours

High variance

- Prediction function not robust to noise
- Small amount of noise can cause big changes to prediction function

Prone to **overfitting**:

- Always fits training examples exactly
- Fits any errors in the training set

Need to choose an appropriate distance measure:

- How to choose a good one for natural images? Audio?

Problems with nearest neighbours

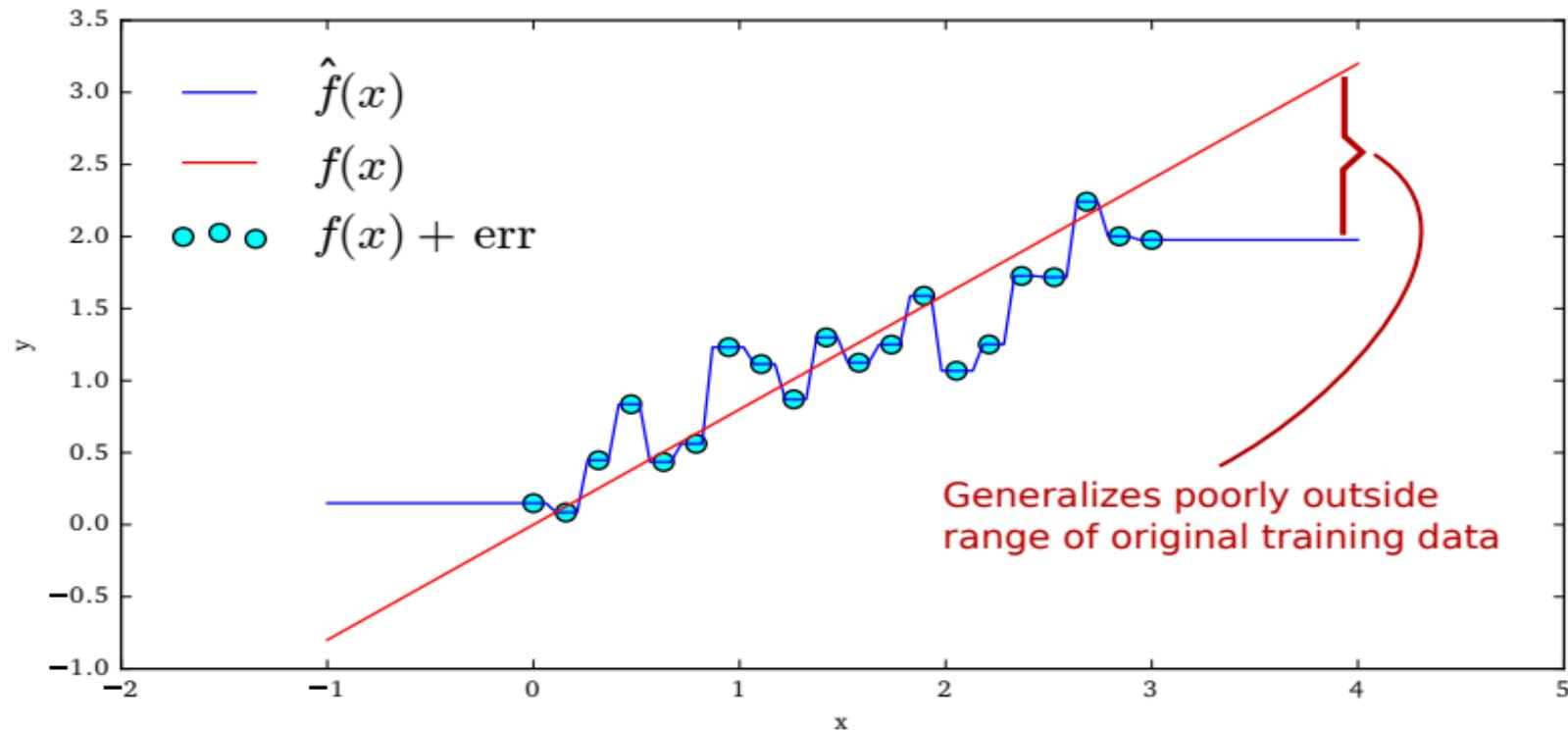
Need to memorize entire training set:

- Memory requirements grow with number of training examples.
- Predict time grows with number of training examples.

Problems with **generalization**

- How to classify examples that are far from what you have seen?
- In high dimensions, your nearest neighbour may be very different! (*curse of dimensionality*)

Nearest neighbours generalisation



Bias and variance

Sources of error

We've seen that the nearest neighbour predictors have **low bias but high variance**.

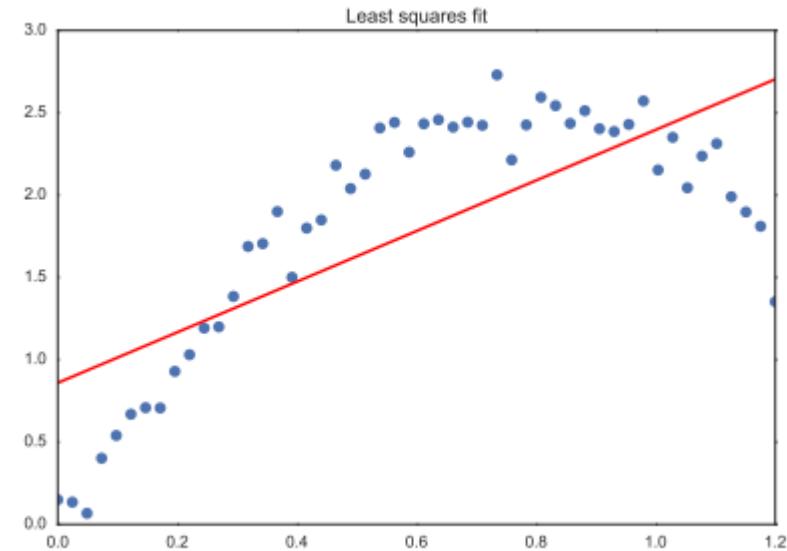
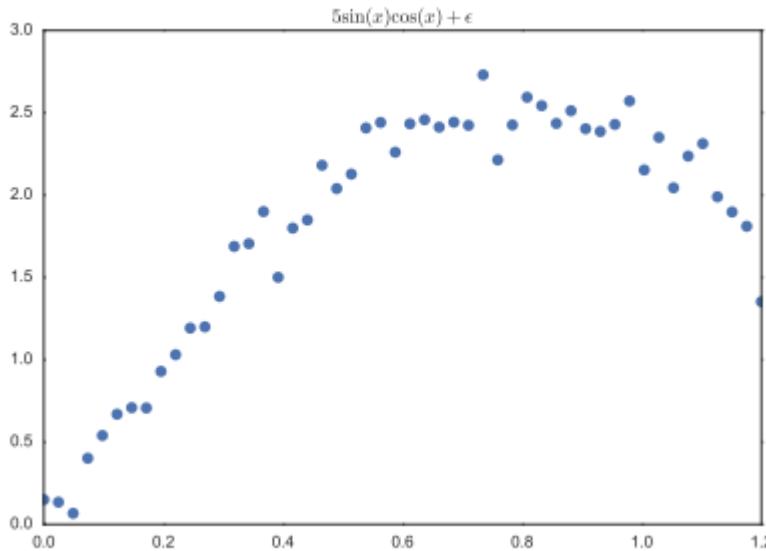
Also seen that nearest neighbours are susceptible to **overfitting**: fitting the **noise**.

We will show that it is possible to **decompose** all sources of error into three components: bias, variance, and unavoidable noise.

This is called the **bias-variance** decomposition.

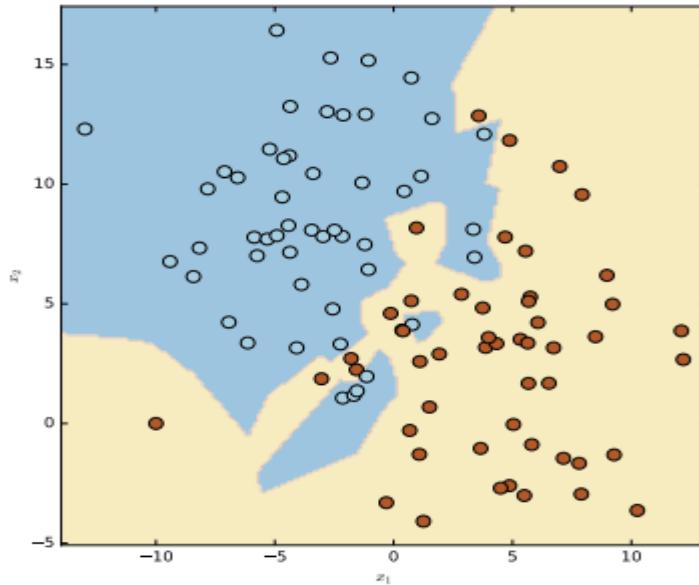
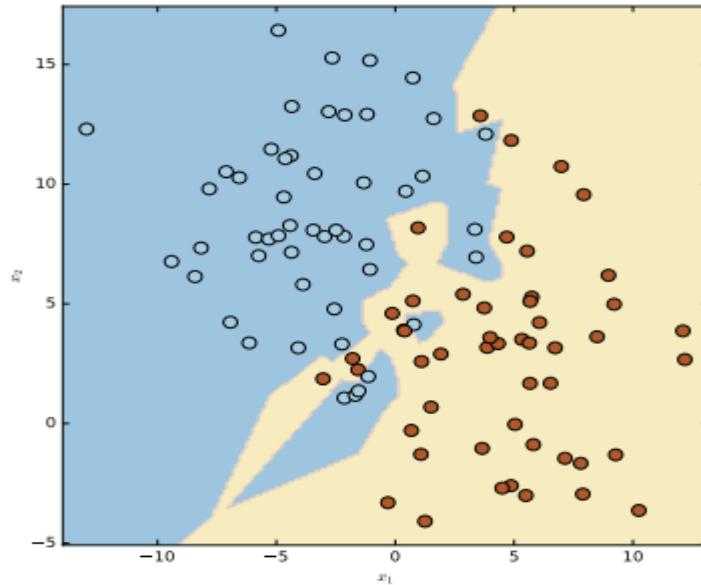
What is bias?

Error due to model rigidity: model not flexible enough to account for observations (underfitting). Often due to incorrect assumptions.



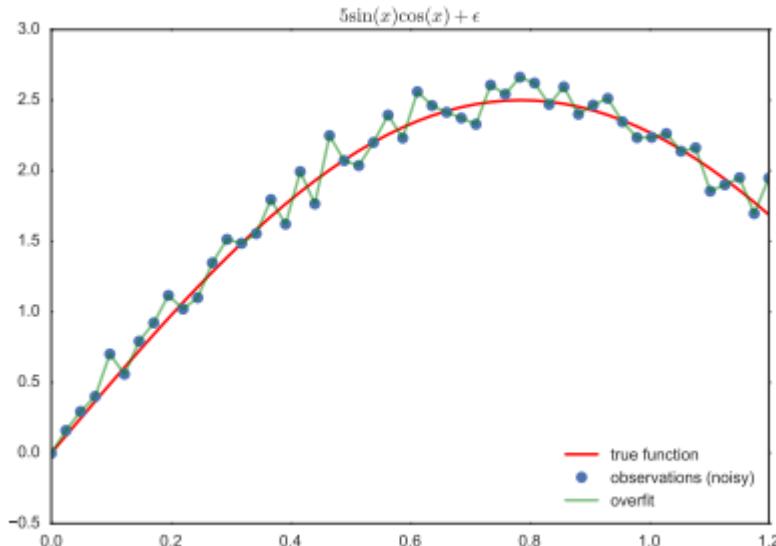
What is variance?

Error due to our model being sensitive to small perturbations in the training data (overfitting). Small changes in training data produce large changes in decision function.



What is noise?

Unavoidable error. Noise: measurement error, unobserved factors that influence target variable. Errors in target variable measurement.



Fitting this kind of unavoidable error is also overfitting.

The bias-variance decomposition

$$\text{Error} = \text{Bias} + \text{Variance} + \text{Noise}$$

Error due to model rigidity: model not flexible enough to account for observations (Underfitting). Often due to incorrect assumptions.

Error due to our model being sensitive to small perturbations in the training data (Overfitting).

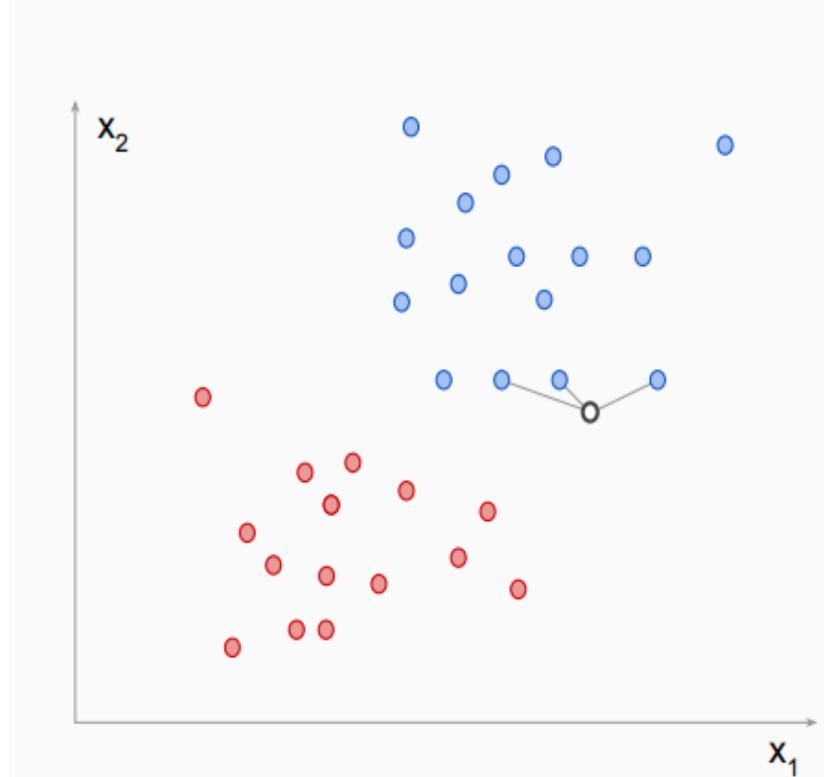
Unavoidable error due to noise. Noise: measurement error, unobserved factors that influence target variable. Errors in target variable measurement

K nearest neighbours

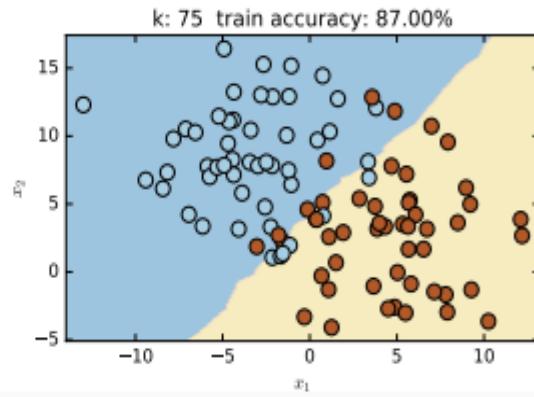
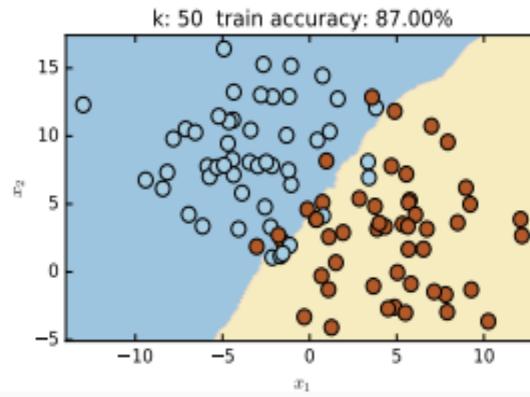
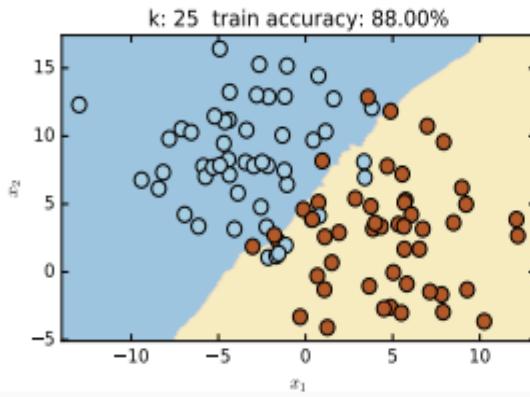
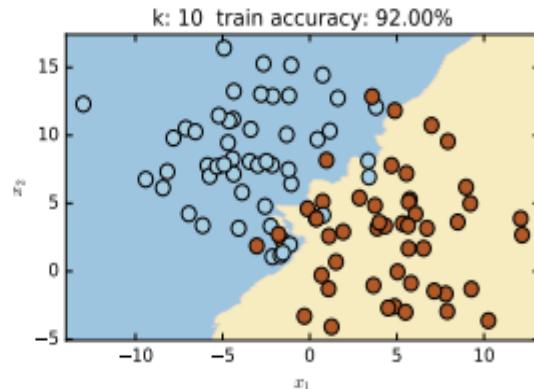
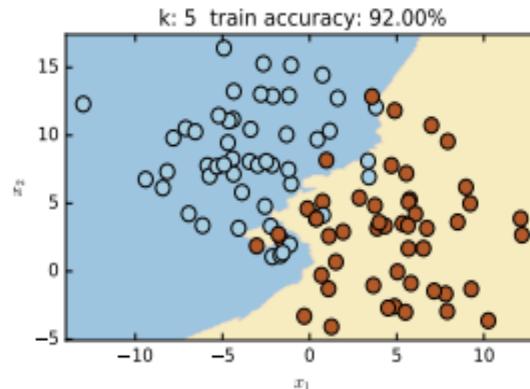
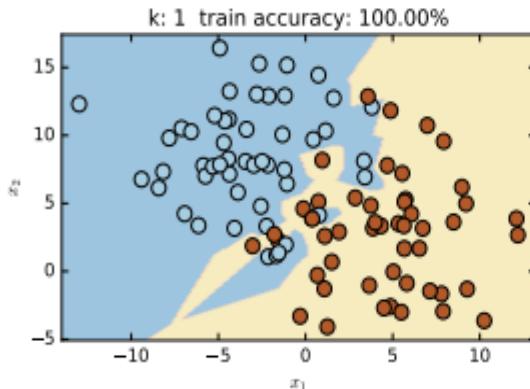
Regression: average y values over the K nearest neighbours

Classification: vote for the correct class with the K nearest neighbours

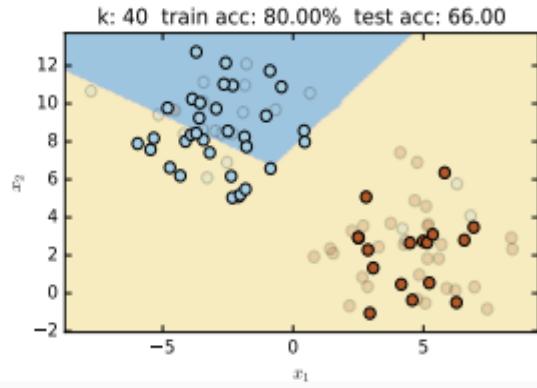
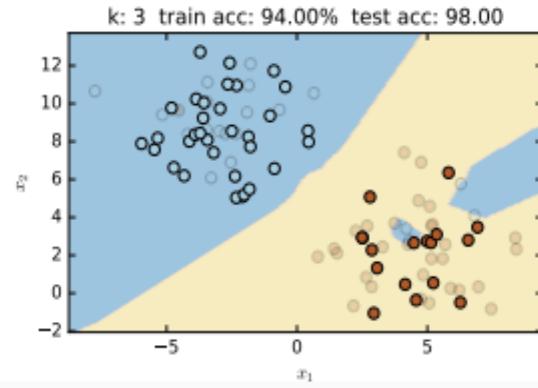
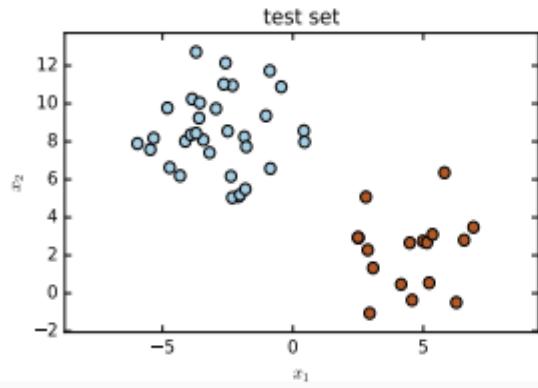
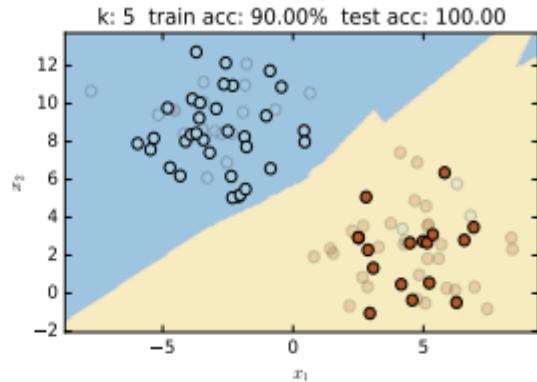
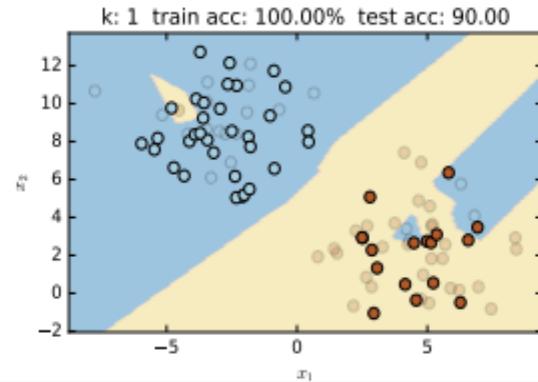
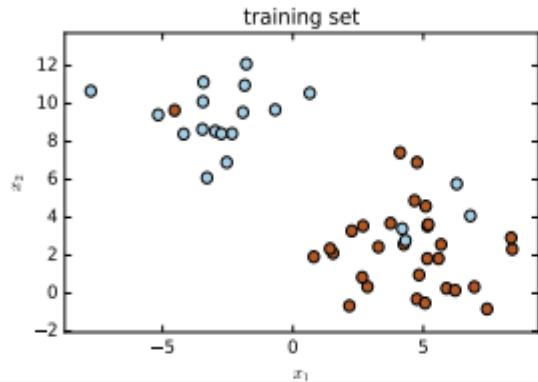
Individual training examples (and noise) now have a much smaller influence on the final model. Noise more likely to be “averaged out” by other neighbours.



K nearest neighbours

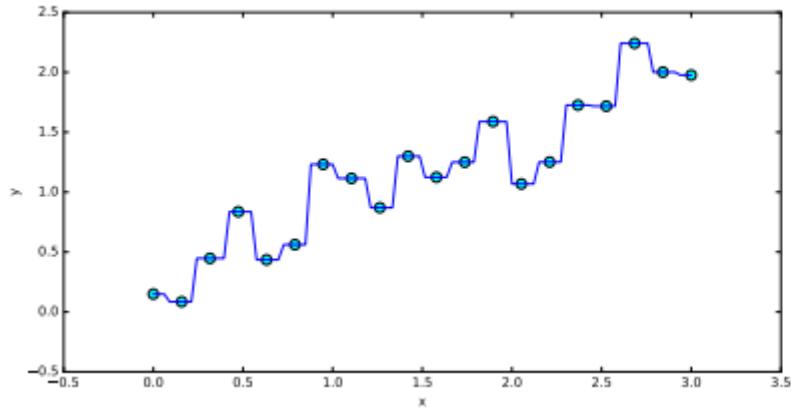


K nearest neighbours

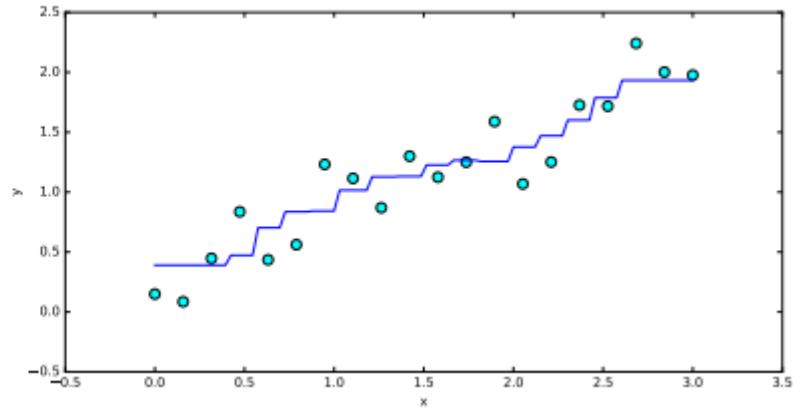


K nearest neighbours

$k = 1$



$k = 5$



K nearest neighbours

Larger K reduces the effect of noise on the final predictor: it **reduces the variance** of the predictor.

However, larger values of K **increase the bias** of the predictor.

The bias-variance tradeoff

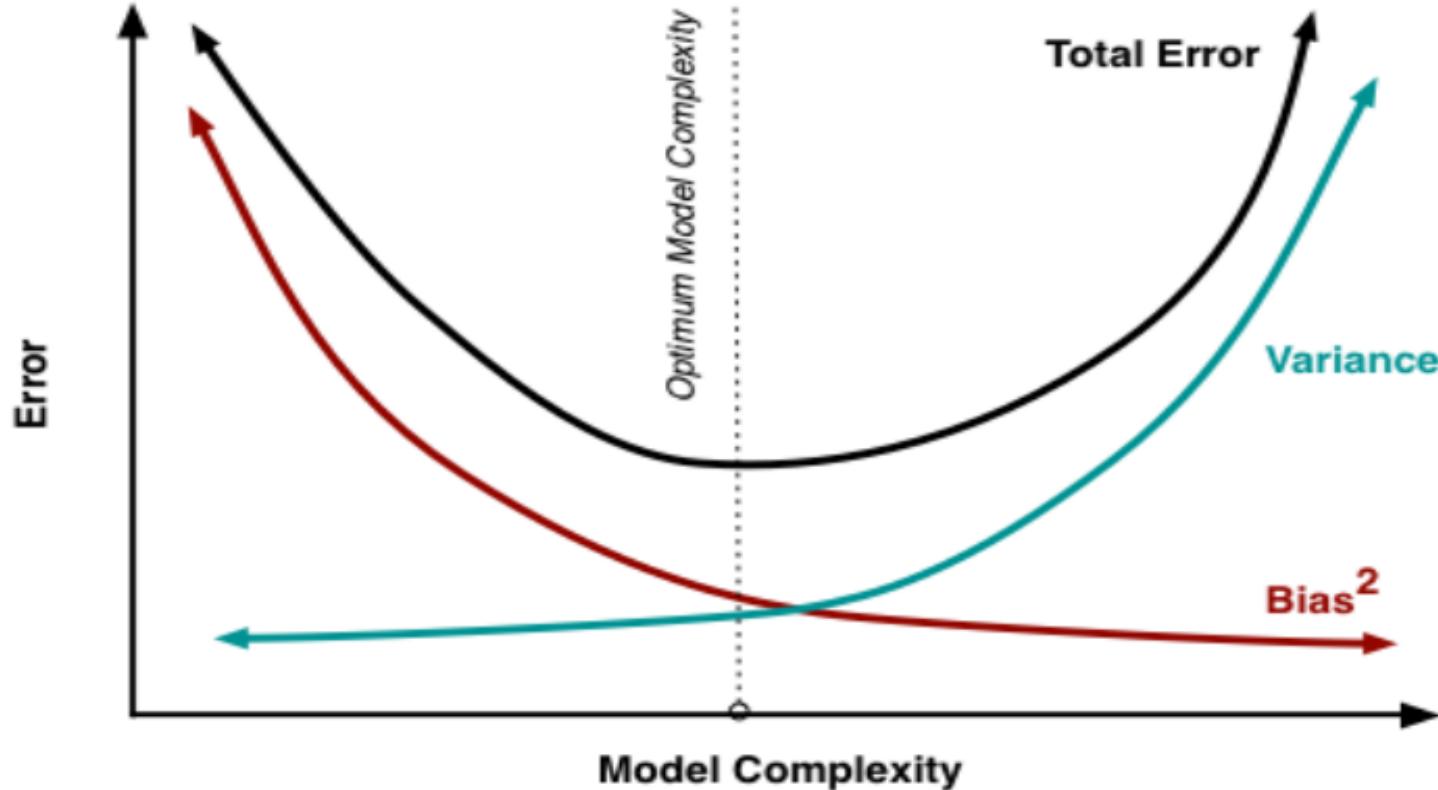
So there is a tradeoff:

- Too large K : out of sample error high due to bias
- Too small K : out of sample error high due to variance

This situation is typical in machine learning and is called the **bias-variance tradeoff**. In general:

- More flexible models (**more degrees of freedom**) have higher variance, but lower bias.
- Less flexible models (**fewer degrees of freedom**) have higher bias, but lower variance.

The bias-variance tradeoff



Choosing K

K nearest neighbours has a parameter (K) that must be selected somehow

K is known as a **hyperparameter** (to distinguish it from learned parameters, as we will see later in linear models)

Other models also have hyperparameters.

How should we select good values for such hyperparameters?

More generally, given a choice of models, how should we choose the one that we expect will generalize best?

Model selection

Model selection

- Choosing appropriate hyperparameters is called model selection.
- More generally: model selection is choosing between a set of candidate models (e.g. linear models, nearest neighbours). Hyperparameters give rise to new models.
- **Idea:** why don't we just try out a bunch of different hyperparameters, train models, test them on the test set, compute error, and use the hyperparameter that gives the lowest error?

Model selection

- Choosing appropriate hyperparameters is called model selection.
- More generally: model selection is choosing between a set of candidate models (e.g. linear models, nearest neighbours). Hyperparameters give rise to new models.
- **Idea:** why don't we just try out a bunch of different hyperparameters, train models, test them on the test set, compute error, and use the hyperparameter that gives the lowest error?
- **THIS IS A BAD IDEA!!!!!**

Model selection

- Hyperparameters are just like any other value: selecting them to minimize error on the test set means you can **overfit the test set!**
- Form of data snooping.
- Means that test error is now a **biased estimate** of OOS error.
- You now need a new test set!
- What to do instead?

Validation sets

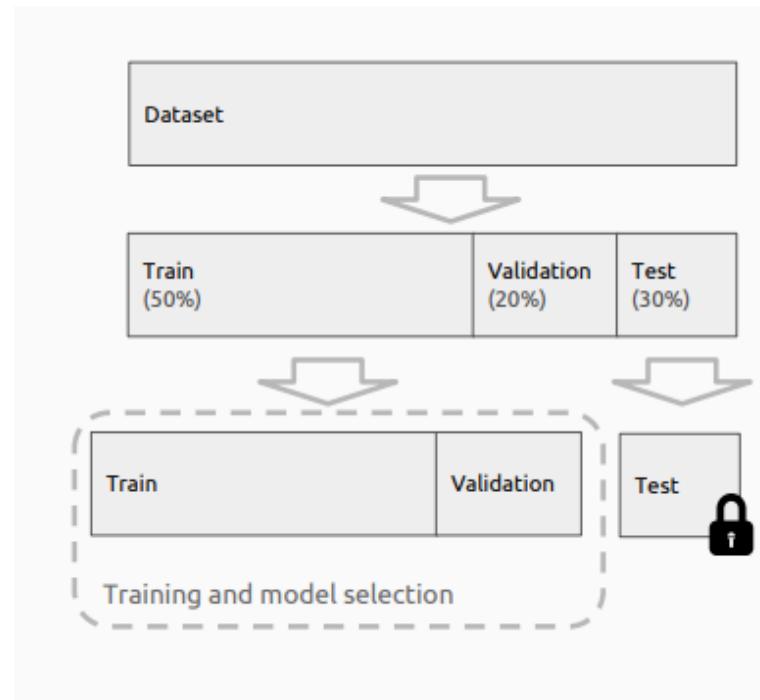
Split the data into 3 sets: training, validation, and test

Use the validation set to choose your model:

- Train with different hyperparameter values
- Compute error on validation set
- Select hyperparameters that give smallest error on validation set

Optional: when no more hyperparameters need to be tuned, retrain on (training + validation)

Finally: estimate OOS error on test set



Cross validation

Issue: using a validation set means you have less training data!

This may be problematic, especially when data is scarce.

We would like to use more of the training data for training, without having to split off a big chunk for validation.

Answer: cross-validation!

Cross validation

- Idea:

1. Divide training data up into five chunks (folds).
2. Train on chunk 1, 2, 3, 4. Compute validation error on 5.
3. Now train on 1, 2, 3, 5. Compute validation error on 4.
4. Now train on 1, 2, 4, 5. Compute validation error on 3.
- 5. ...
- 6. Average together errors.
- Choose model that minimizes average validation error.

K-fold cross validation

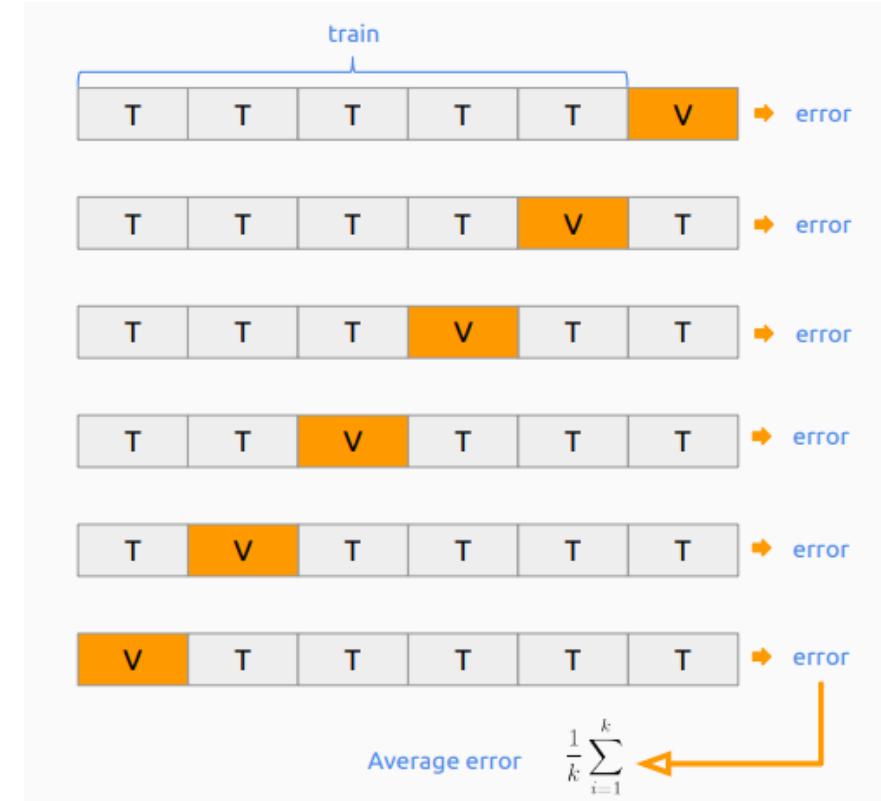
Divide training set up into k folds.

- K can be as small or large as you like
- Large k (smaller folds): more training data, slower
- Small k (larger folds): less training data, faster

Train on $(k - 1)$ folds, validate on remaining fold

Repeat until all folds have been validated on

Average validation error over folds



Leave one out cross validation

Make folds as small as possible: 1 single training example per fold

Train on all data bar one example

Compute error on single “left out” example

Repeat $(N - 1)$ times, leaving out a different example each time

Compute validation error as average of error on all N left out examples.

- **Advantages**

- Make maximum use of the training data
- Appropriate when there is very little training data available

- **Disadvantages:**

- Need to run learning algorithm N times: may be very time consuming if learning is algorithm is slow

Model selection in practice

When you have **lots of data**: use a dedicated validation set.

When you have **less data** and more computing resources: use cross validation.

When you have **very little data** and want to make maximum use of it: use leave one out cross validation.

The no free lunch theorem

We've already introduced several machine learning algorithms.

Remainder of the module will introduce even more.

Why so many? Is there not one algorithm that will work reasonably well on all types of tasks?

No free lunch: **bias free learning is futile.**

You can't do inference without making assumptions!

Different assumptions needed for different tasks.

The no free lunch theorem

No Free Lunch theorem (Wolpert, 1996): there is no one model that works best for every problem.

Assumptions of a great model for one problem may not hold for another; it is common in ML to try multiple models and select one that works best for a particular problem.

You typically cannot tell in advance which algorithm will work best: there is no **a-priori** distinction between them.

This is why **model selection** is so important: test several models and choose one with least validation error.

Summary

- Black box abstraction of supervised learning
- Regression and classification problems
- Types of error
- Training and test data splits
- Nearest neighbours
- Bias and variance
- K nearest neighbours
- Model selection
- The no free lunch theorem

Further reading

The elements of statistical learning:

- Intro to supervised learning, nearest neighbours: Chapter 1, 2
- Model selection, bias, variance: Chapter 7
- Linear and quadratic discriminant analysis: Section 4.3

Questions?

Resources

Alternate derivation of the bias-variance decomposition by Yasser Abu-Mostafa from Caltech. Video lecture on YouTube (**Highly recommended**):

<https://www.youtube.com/watch?v=zrEyxfI2-a8>. Starts approx 9 mins in.

EEN1083/EEN1085

Data analysis and machine learning I

Ali Intizar

Semester 1
2024/2025



Ollscoil Chathair
Bhaile Átha Cliath
Dublin City University

Course material is provided under the Creative Commons Attribution-NonCommercial-ShareAlike license. You are free to copy it, adapt it, and redistribute it, provided you do not sell it in any form.

Linear Models



Ollscoil Chathair
Bhaile Átha Cliath
Dublin City University

Outline

Introduction

Linear Regression

Gradient descent
Stochastic
gradient descent

Probabilistic view

Linear classification

Logistic regression

Features

Overfitting

Regularization

L2 regularization

L1 regularization

Multiple outputs

Multiple regression

One-vs-rest

Softmax regression

Summary

Outline

1. Introduction
2. Linear Regression
3. Linear Classification
4. Features
5. Multi-class Regression

Introduction

What are linear models?

Linear models are **parametric models** that are characterized by having prediction functions of the form:

$$\hat{y} = f(\mathbf{x}) = g(\mathbf{w}^T \mathbf{x} + b)$$

- \mathbf{w}, b are parameters. These need to be learned from the training data
- \mathbf{w} is called the **weight vector**
- b is a **scalar bias**
- $g()$ is an activation function, which may be the identity (i.e. $g(x) = x$).⁶
- if $\mathbf{x} \in \mathbb{R}^D$ then $\mathbf{w} \in \mathbb{R}^D$ and there are $(D + 1)$ parameters.

Example

Assume we have two features, $\mathbf{x} = [x_1 \quad x_2]^T \in \mathbb{R}^2$ and we are using the identity activation function $g(\mathbf{x}) = \mathbf{x}$.

The decision function is:

$$\begin{aligned}f(\mathbf{x}) &= \mathbf{w}^T \mathbf{x} + b \\&= \sum_{i=1}^D w_i x_i + b \\&= w_1 x_1 + w_2 x_2 + b\end{aligned}$$

There are three free parameters to be learned: $\theta = \{w_1, w_2, b\}$

Note that it is also possible to prepend a 1 to the feature vector, e.g.

$\mathbf{x} = [1 \ x_1 \ x_2]^T \in \mathbb{R}^3$. We can then write the decision function more succinctly as:

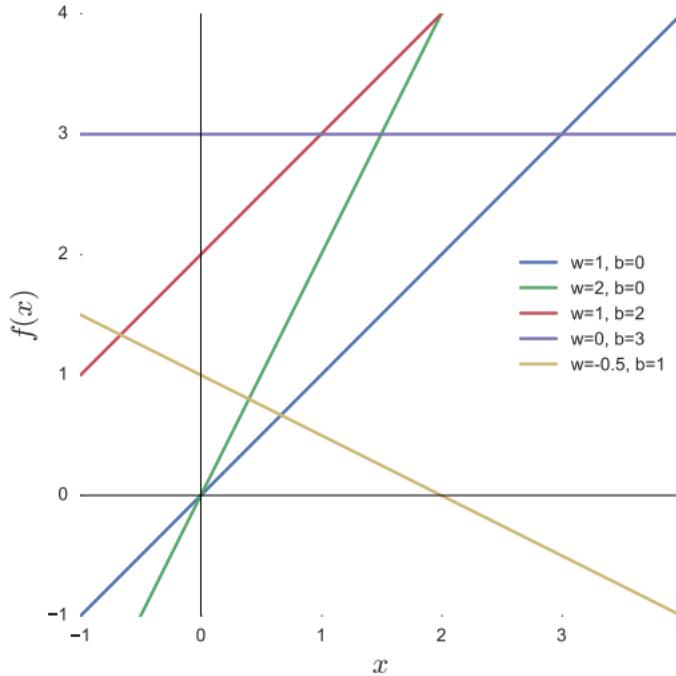
$$f(\mathbf{x}) = g(\mathbf{w}^T \mathbf{x})$$

And the free parameters are now $\theta = \{w_0, w_1, w_2\}$, with w_0 being the implicit bias.

Models like this are called linear models because $\mathbf{w}^T \mathbf{x} + b$ (or $\mathbf{w}^T \mathbf{x} + b = 0$) defines a line.

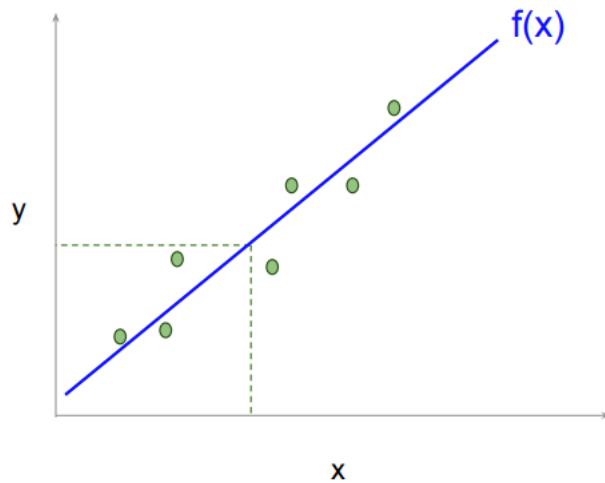
- Plane or **hyperplane** in higher dimension

The slope (normal) of the line is given by \mathbf{w} . The y -intercept is given by b in regression.

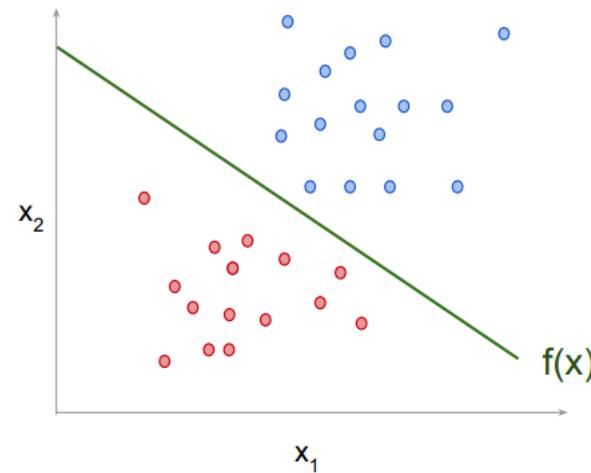


Classification and regression with linear models

When using a linear model for regression, we simply output the value of $\hat{y} = \mathbf{w}^T \mathbf{x} + b$



When using a linear model for classification, we are interested in which side of the line $f(\mathbf{x}) = c$ the target appears on.



Why linear models?

The linear assumption seems extremely rigid. It appears at first glance that linear models would be of very limited use.

Turns out that linear models are extremely useful!

Reasons:

1. Even if there is no linear relationship (or decision boundary), often possible to **project** data to a space in which relationship is linear.
2. If you have very little data, you need to make strong assumptions to fit a model.
Low variance.
3. **Occam's razor** (principle of parsimony). Simpler models that explain the data are better than more complicated ones.
4. Possible to build much more complex models using linear models as a building block. This is the basis of **neural networks** and **deep learning**.

What can you do with linear models?

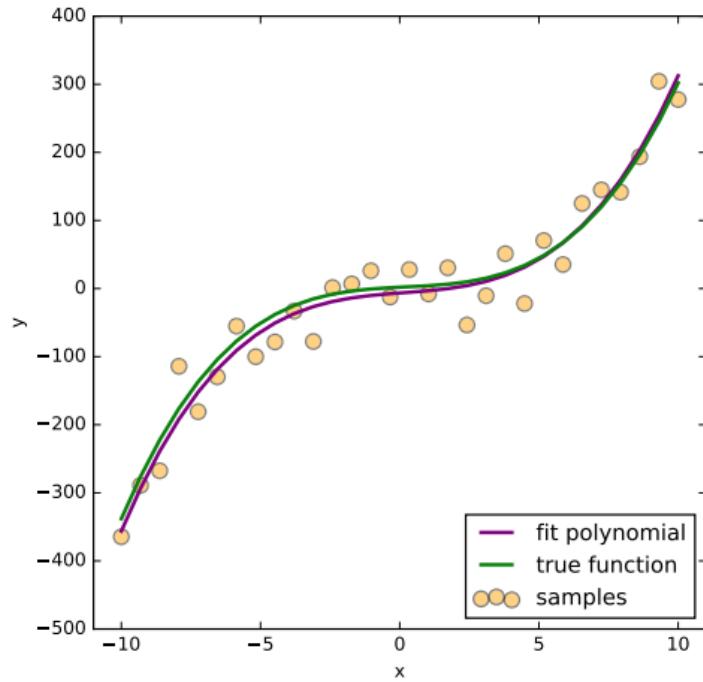
Obviously, fit lines.

But also:

- Polynomials
- Circles

Anything that is **linear in the weights**.

Will see how to do this later.



Linear Regression

Linear regression

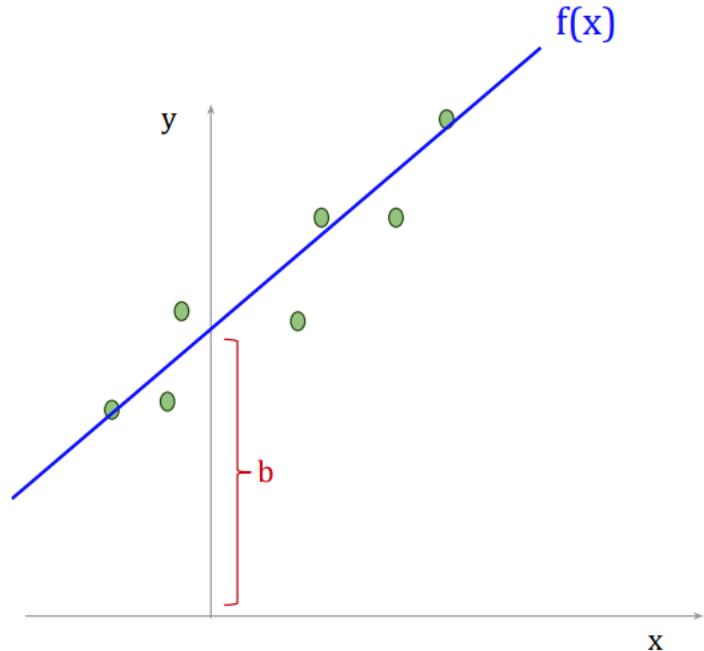
Regression: output variable y is continuous real values $y \in \mathbb{R}$.

Decision function:

$$f(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b$$

In 1D, f is a line. In 2D a plane. In N -D a hyperplane.

\mathbf{w} defines the slope of the line, b defines its offset from the origin on the y -axis.



Linear regression

In linear regression (aka **ordinary least squares**), we would like to find parameters $\{\mathbf{w}, b\}$ such that the resulting line $f(\mathbf{x})$ fits the training data well.

Need to define what it means to fit the data well.

A natural way to do this is using squared error. The squared error for the prediction $\hat{y} = f(\mathbf{x})$ is:

$$(y - \hat{y})^2$$

Total squared error for the training set $T = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$ is the sum over the individual squared errors:

$$\frac{1}{2} \sum_{i=1}^N (y_i - \hat{y}_i)^2$$

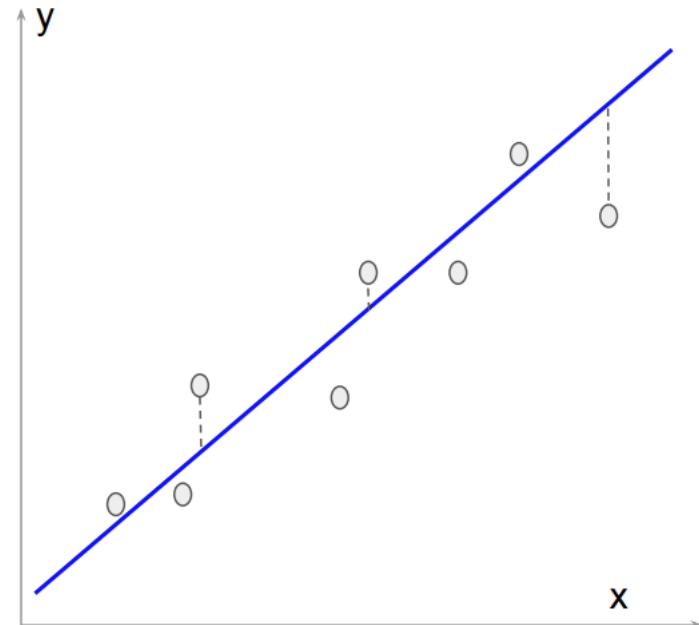
Linear regression

Define the **loss** (cost) function as:

$$\mathcal{L}(\mathbf{w}, b) = \frac{1}{2} \sum_{i=1}^N (y_i - \hat{y}_i)^2$$

In linear regression, we want to find the $\theta = \{\mathbf{w}, b\}$ that **minimize** this squared error loss.

$$\hat{\theta} = \arg \min_{\mathbf{w}, b} L(\mathbf{w}, b)$$



Linear regression

How do we minimize the loss?

From calculus, we know that the minimum value of a function is at a **critical point**: a place where the gradient is zero:

$$\frac{\partial L}{\partial w_i} = 0 \quad \text{and} \quad \frac{\partial L}{\partial b} = 0$$

Approach: take gradients and set them to zero, then solve for \mathbf{w} and b in closed form.

This is possible and leads to what are called the **normal equations** for **ordinary least squares**.

Normal equations

Start by expressing the difference between the target and predictions on the training set X, \mathbf{y} in vector form:

$$\mathbf{e} = \mathbf{y} - X\mathbf{w}$$

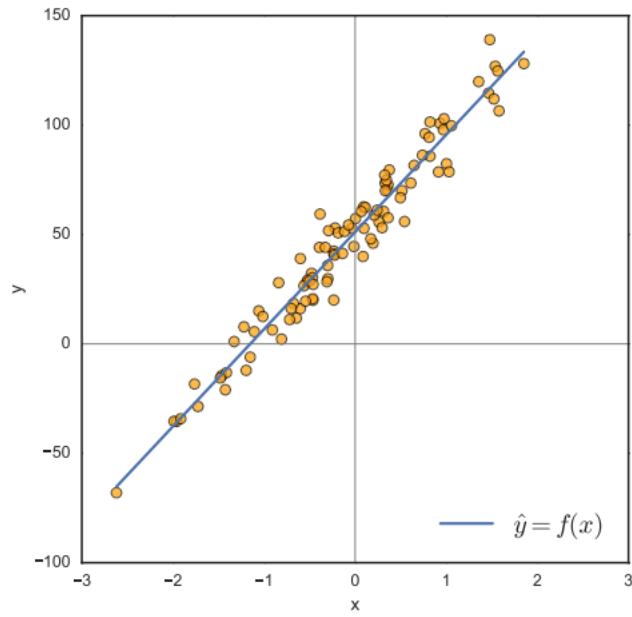
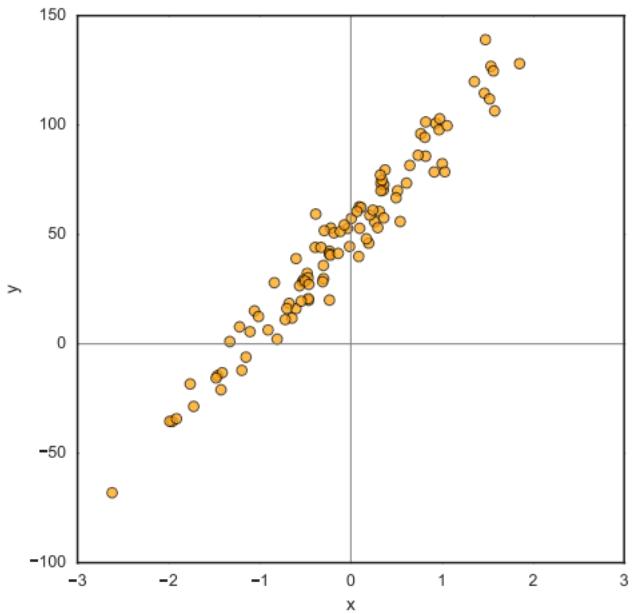
with X being our training examples with a one prepended to each row. This allows the bias parameter to be treated like any other weight.

$$X = \begin{bmatrix} 1 & \mathbf{x}_1^T \\ 1 & \mathbf{x}_2^T \\ \vdots & \vdots \\ 1 & \mathbf{x}_N^T \end{bmatrix} \quad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}$$

Normal Equation

$$\mathbf{w} = (X^T X)^{-1} X^T \mathbf{y}$$

Example

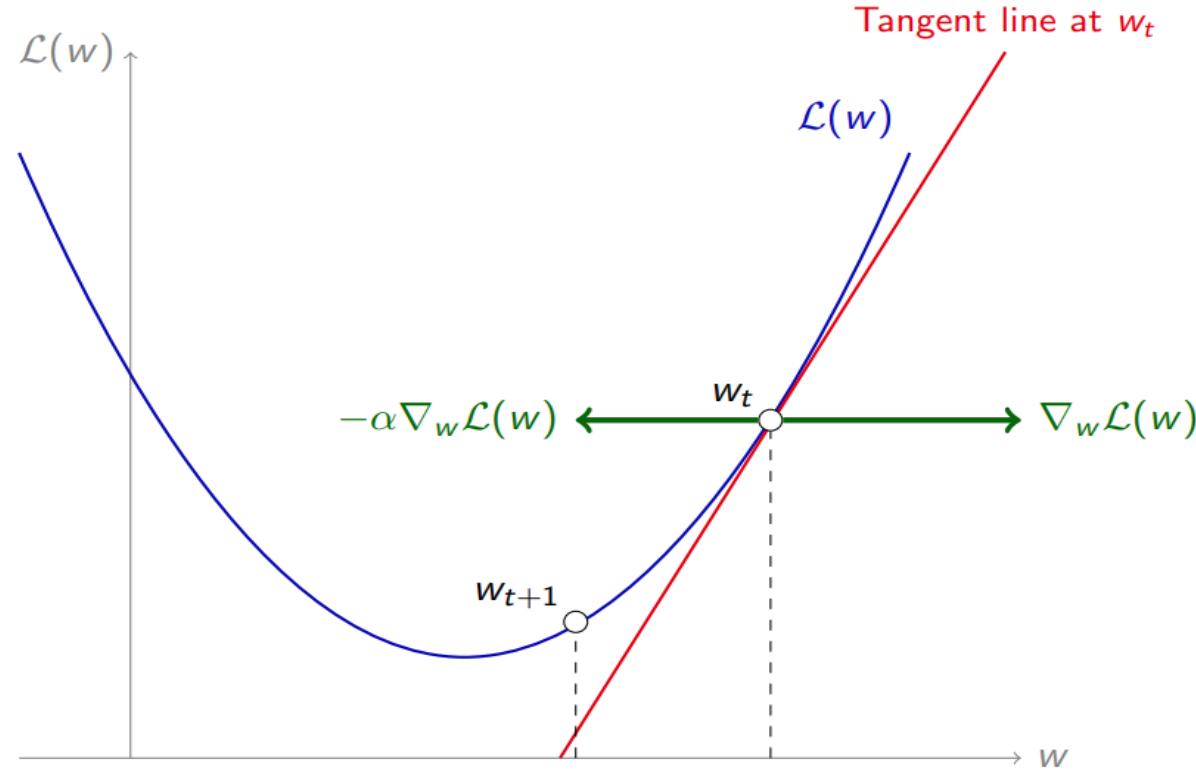


$$w = 44, b = 51$$

Gradient descent

- Often in machine learning we will take a similar approach to producing an algorithm as we did for linear regression.
- That is, specify a loss function and then figure out how to minimize it.
- Usually, however, the approach of setting the gradients equal to zero and solving for the parameters closed form won't work.
- An alternative, more general approach, is to start with a random guess for the parameters, and then **iteratively improve** this guess.
- One very common way of doing this is **gradient descent** (steepest descent).

Gradient descent: the idea



Gradient descent: the algorithm

Algorithm

1. Start with a random guess for the initial parameters θ_0
2. While not converged:
 - 2.1 Set $\theta_{t+1} = \theta_t - \alpha_t \nabla_{\theta} L(\theta_t)$

The value of α_t is called the **learning rate**. Decide convergence

when:

- The change in θ is small $|\theta_{t+1} - \theta_t| < T$
- Maximum number of iterations are reached.
- Loss stops improving.

Gradient descent for linear regression

Loss function:

$$\mathcal{L}(\mathbf{w}, b) = \frac{1}{2} \sum_{i=1}^N (y_i - f(\mathbf{x}_i))^2$$

Gradient of the loss function wrt. \mathbf{w} :

$$\begin{aligned}\nabla_{\mathbf{w}} \mathcal{L}(\mathbf{w}) &= \frac{1}{2} \sum_{i=1}^N \nabla_{\mathbf{w}} (y_i - f(\mathbf{x}_i))^2 \\ &= \sum_{i=1}^N (f(\mathbf{x}_i) - y_i) \mathbf{x}_i\end{aligned}$$

Gradient of the loss function wrt. b :

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial b} &= \frac{1}{2} \sum_{i=1}^N \partial_b (y_i - f(\mathbf{x}_i))^2 \\ &= \sum_{i=1}^N (f(\mathbf{x}_i) - y_i)\end{aligned}$$

Gradient descent for linear regression

Update rules:

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \alpha_t \sum_{i=1}^N (f(\mathbf{x}_i) - y_i) \mathbf{x}_i$$

$$b_{t+1} = b_t - \alpha_t \sum_{i=1}^N (f(\mathbf{x}_i) - y_i)$$

Note: may want to divide learning rate α_t by N to make learning independent of number of data points.

Properties of gradient descent

The learning rate α_t :

- Gradient descent will converge on convex functions as long as the learning rate α_t keeps getting smaller over time.
- Gradient descent may be very slow if α_t is too small.
- Gradient descent may diverge if α_t is too large.

Gradient descent requires a full pass over the dataset each time it updates the parameters (weights and biases). Slow for large datasets!

Stochastic gradient descent

Idea: Instead of computing the gradient of the loss for the full dataset, estimate it using a **single randomly chosen example** and then take step in that direction.

Total loss:

$$\mathcal{L}(\mathbf{w}, b) = \frac{1}{2} \sum_{i=1}^N (y_i - f(\mathbf{x}_i))^2$$

Loss for a single example \mathbf{x}_i :

$$\frac{1}{2}(y_i - f(\mathbf{x}_i))^2$$

SGD update rules:

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \alpha_t (f(\mathbf{x}_i) - y_i) \mathbf{x}_i \quad b_{t+1} = b_t - \alpha_t (f(\mathbf{x}_i) - y_i)$$

Properties of SGD

Usually much faster than batch gradient descent, since we update the weights every iteration instead of every epoch.

Can operate on large datasets, since we only need to process a single example at a time.

Important to shuffle training data to ensure the gradient estimates are unbiased (i.i.d.)

But: gradient updates are noisy, since it is only an estimate of the full gradient.

Convergence may be slower as you near the optimum solution.

Linear regression

Assumptions of linear regression:

- Linear relationship between variates and covariate
- All other variation can be modelled as zero mean Gaussian noise

If these assumptions are violated, then linear regression may be a bad model.

Implications:

1. Linear regression is sensitive to extreme outliers.
2. Linear regression may not be appropriate when the noise is not Gaussian.
3. Linear regression may not give good results if the relationship between the variates and covariates is not approximately linear.

Noise: any unmeasured quantities, including random variation and measurement error.

Linear classification

Linear classification

Recall the binary classification setup:

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1^T \\ \mathbf{x}_2^T \\ \vdots \\ \mathbf{x}_N^T \end{bmatrix} \in \mathbb{R}^{N \times D} \quad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix} \in \{0, 1\}^N$$

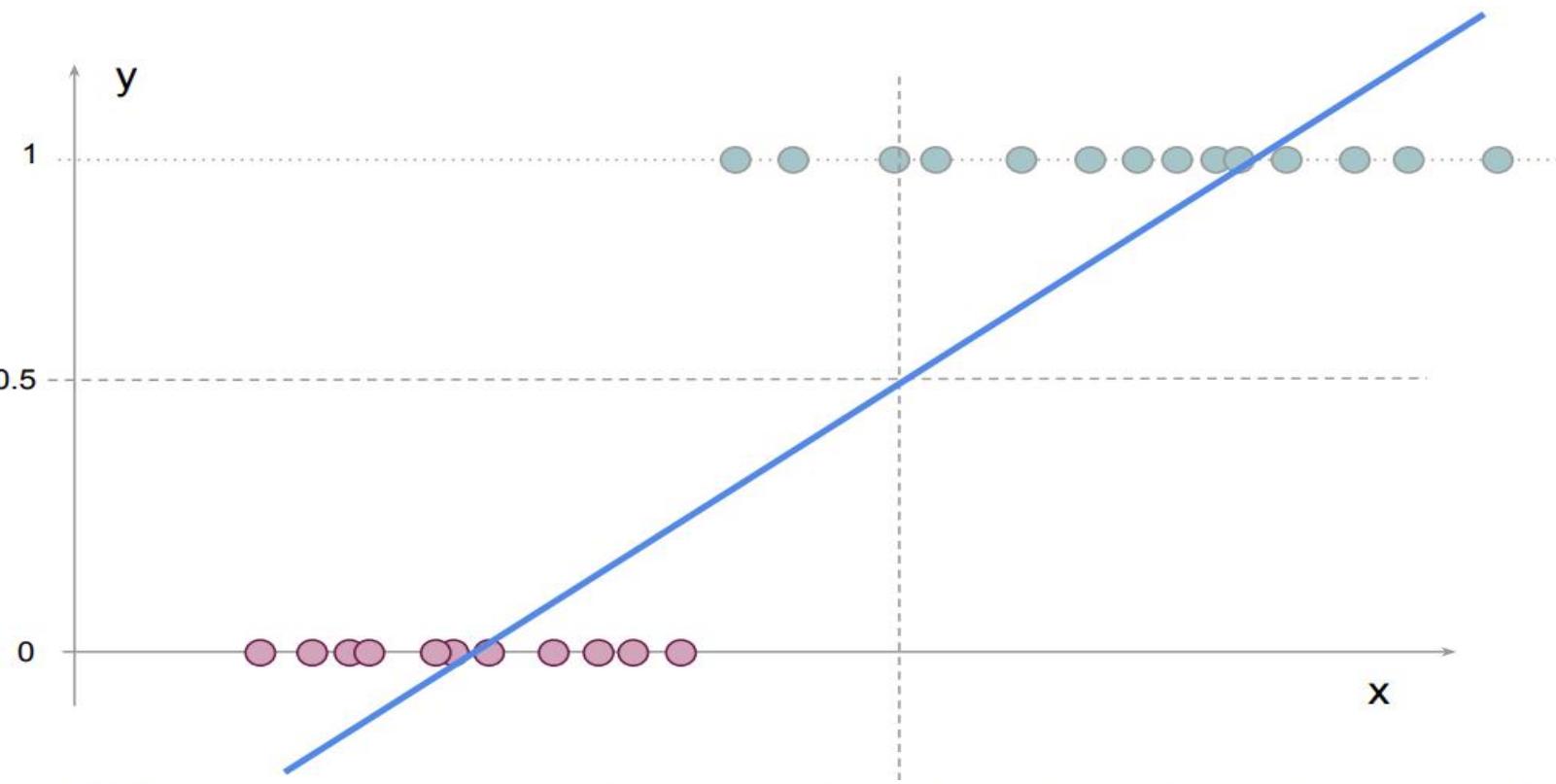
Q: Is it possible to use our linear regression model to do classification?

Linear classification

A: Sure, but it probably won't work very well.

1. The relationship between the variates and covariates is not directly linear
2. We would like our outputs to be binary variables in $\{0, 1\}$ (or probabilities in $[0 \dots 1]$), but linear regression produces arbitrary real numbers. We would need to do some post processing to map the values to the desired range.
3. Assumption of Gaussian noise is not true for binary outputs. So square error is inappropriate.

Linear classification with least squares



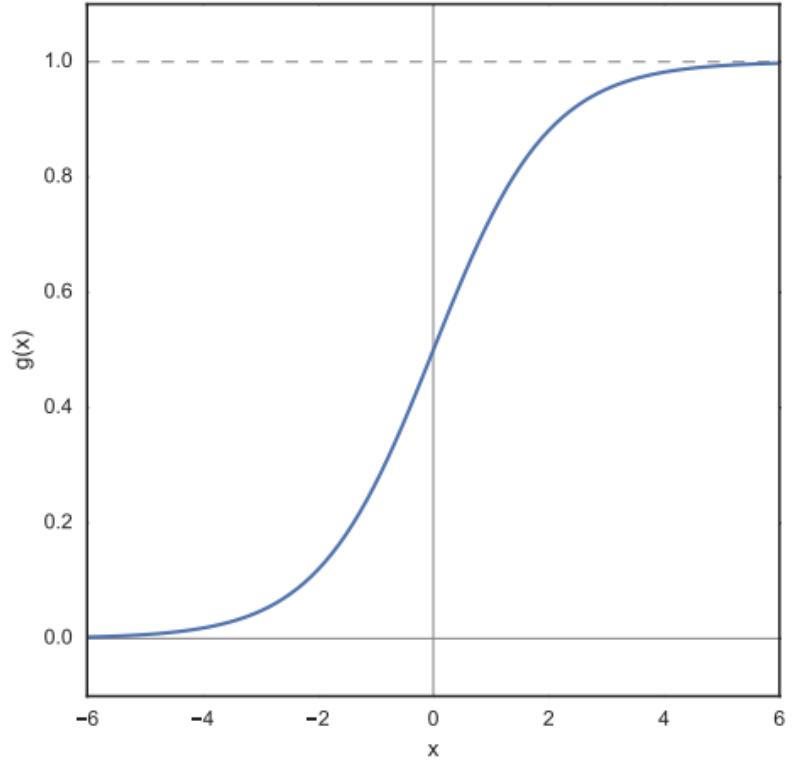
Logistic regression

Let's assume that the relationship between the input and the outputs is:

$$f(\mathbf{x}) = g(\mathbf{w}^T \mathbf{x} + b)$$

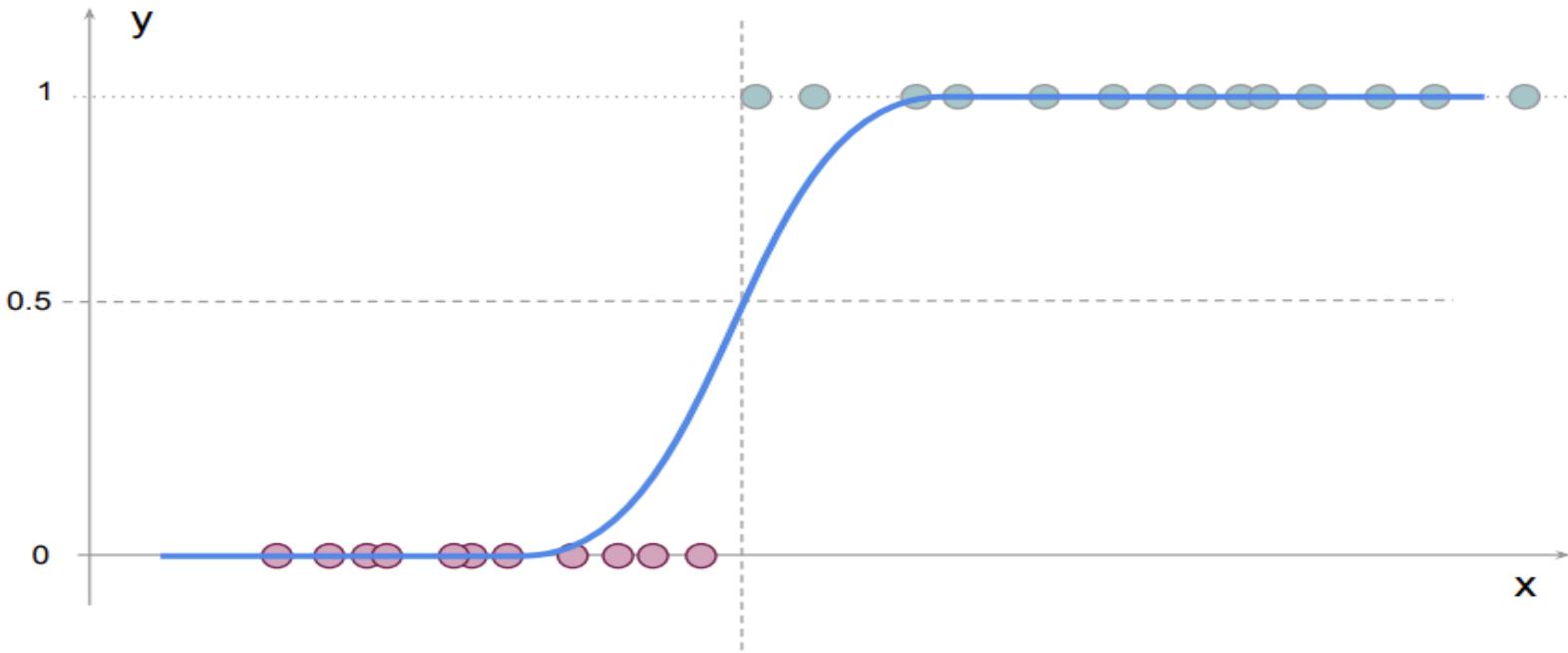
with $g(\cdot)$ being the sigmoid (logistic) function, which “squashes” its input to the range (0, 1):

$$g(x) = \frac{1}{1 + e^{-x}}$$



Logistic regression

$$f(\mathbf{x}) = g(\mathbf{w}^T \mathbf{x} + b)$$



Logistic regression: Loss Functions

To fit the parameters we would also like to use a **more appropriate loss function than square loss** (which is only appropriate for normally distributed error).

Probability Distribution

$f(\mathbf{x})$ can now be interpreted as a probability distribution $P(\mathbf{y} | X, \theta)$. Following the maximum likelihood approach, we want to find:

$$\begin{aligned}\hat{\theta}_{\text{ML}} &= \arg \max_{\theta} P(X, \mathbf{y} | \theta) \\ &= \arg \max_{\theta} P(\mathbf{y} | X, \theta)P(X) \\ &= \arg \max_{\theta} P(\mathbf{y} | X, \theta)\end{aligned}$$

Binary cross entropy loss

a loss function to minimize loss

$$\mathcal{L} = - \sum_{i=1}^N y_i \log f(\mathbf{x}_i) + (1 - y_i) \log(1 - f(\mathbf{x}_i))$$

We now have all the components needed to specify the logistic regression algorithm:

1. An activation function (transfer function): the logistic sigmoid
2. An appropriate loss function (cross-entropy) derived using maximum likelihood

Logistic regression

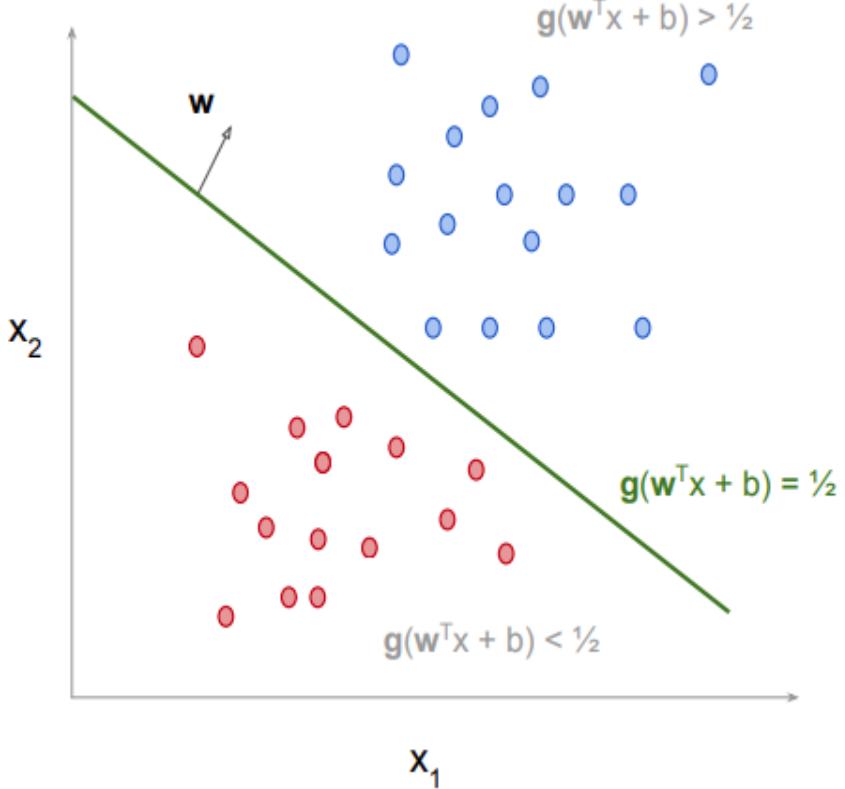
$$f(\mathbf{x}) = g(\mathbf{w}^T \mathbf{x} + b)$$

Activation function: sigmoid

$$g(x) = \frac{1}{1 + e^{-x}}$$

Loss function: cross entropy

$$\mathcal{L} = - \sum_{i=1}^N y_i \log f(\mathbf{x}_i) + (1-y_i) \log(1-f(\mathbf{x}_i))$$



Gradient descent for logistic regression

If we try to find the gradient of the loss with respect to the parameters and set this to zero and try find a closed form solution like we did with linear regression, we will fail.

Instead we turn again to an iterative solution: **gradient descent**.

We need to first find the gradient (vector) of the loss function wrt. the parameters

$$\nabla_{\theta} L = \frac{\partial L}{\partial \theta_i}$$

Once we have this, we can perform gradient descent (or SGD) using the usual update rule to find good parameters $\hat{\theta} = \{\mathbf{w}, b\}$:

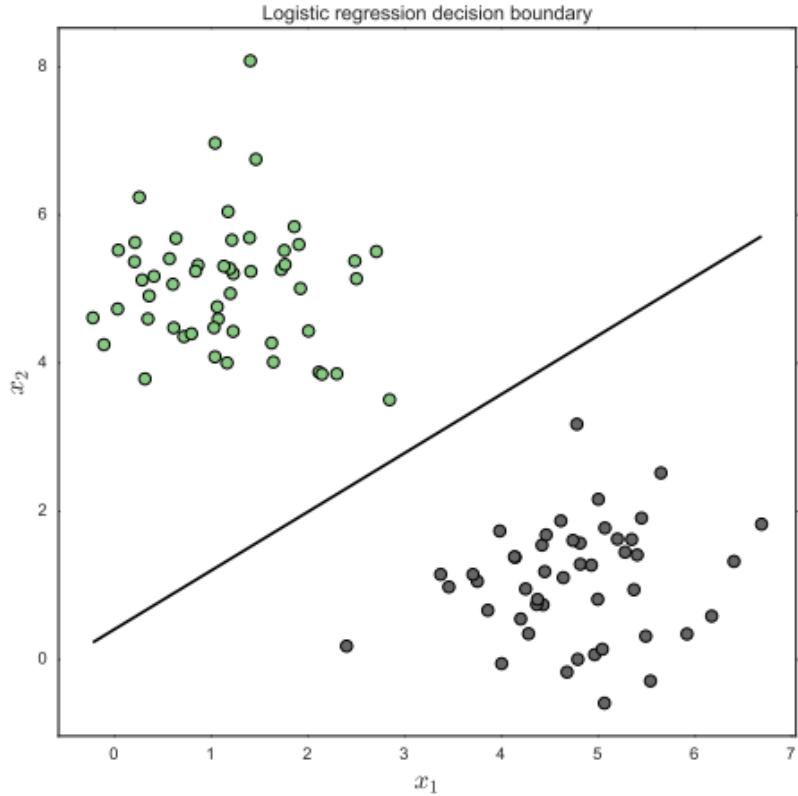
$$\theta_{t+1} = \theta_t - \alpha \nabla_{\theta} L(\theta_t)$$

Example

Training data: 100 points sampled from two 2D Gaussians with centers at $(1, 5)$ and $(5, 1)$.

Gradient descent:

- Initial $\mathbf{w}_0 = (0, 0)$ and $b_0 = 1$.
- Learning rate $\alpha = 0.001$.
- Gradient descent for 1000 iterations.
- Initial loss: 0.84. Final loss: 0.00489.
- Fit parameters $\hat{\mathbf{w}} = (1.75, -2.2)$ and $\hat{b} = 0.926$



Linear and logistic regression in scikit-learn

1. `sklearn.linear_model.LogisticRegression`
2. `sklearn.linear_model.LinearRegression`

Usual methods: `fit(X, y)`, `predict(X)`, and `score(X, y)`.

Fit parameters $\hat{\mathbf{w}}$, \hat{b} accessible via `.coef_` and `.intercept_` attributes.

```
from sklearn.linear_model import LogisticRegression
clf = LogisticRegression()
clf.fit(X, y)
y_hat = clf.predict(X)
```

Summary

- Logistic regression is a binary **classification** algorithm.
- Decision function:

$$f(\mathbf{x}) = g(\mathbf{w}^T \mathbf{x} + b)$$

with

$$g(z) = \frac{1}{1 + \exp(-z)}$$

- Loss function is **binary cross entropy**:

$$\mathcal{L} = - \sum_{i=1}^N y_i \log f(\mathbf{x}_i) + (1 - y_i) \log(1 - f(\mathbf{x}_i))$$

- Optimize using gradient descent (or SGD)

$$\nabla_{\mathbf{w}} \mathcal{L} = \sum_{i=1}^N (f(\mathbf{x}_i) - y_i) \mathbf{x}_i$$

Features



More on linear regression: fitting polynomials

Possible to use linear regression for more than just fitting lines!

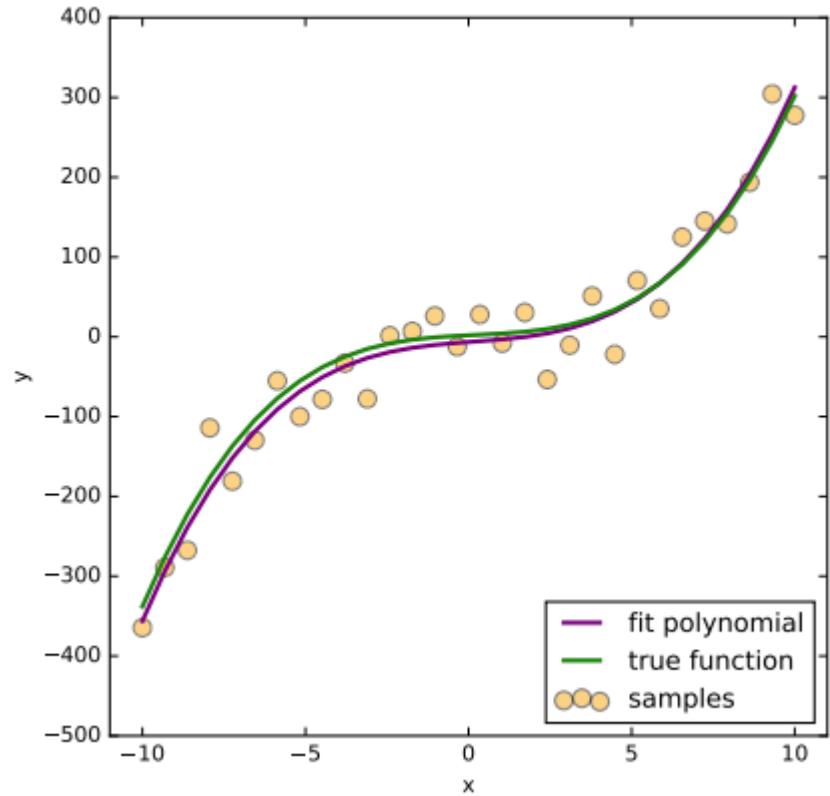
Only needs for function to be linear in the parameters

E.g. can fit polynomials:

$$f(x) = w_1x + w_2x^2 + w_3x^3 + b$$

Figure on right:

$$f(x) = 2x + 0.2x^2 + 0.3x + 2 \quad y = f(x) + \epsilon$$



Fitting polynomials

To fit a polynomial with degree D we just need to construct the X matrix so that contains the relevant powers of x : x, x^2, \dots, x^D . E.g. to fit a degree 3 polynomial, construct X as follows:

$$X = \begin{bmatrix} 1 & x_1 & x_1^2 & x_1^3 \\ 1 & x_2 & x_2^2 & x_2^3 \\ 1 & x_3 & x_3^2 & x_3^3 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_N & x_N^2 & x_N^3 \end{bmatrix}$$

Can then just use linear regression as usual.

To fit polynomials in higher dimensions, construct X which contain powers of all input variables and cross terms (if necessary).

Nonlinear mappings

More generally, we can use any (nonlinear) function to map \mathbf{x} to a different space of features. The resulting function is still linear in \mathbf{w} .

$$f(\mathbf{x}) = \mathbf{w}^T \varphi(\mathbf{x}) + b$$

E.g.

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \quad \phi(\mathbf{x}) = \begin{bmatrix} x_1 \\ x_2 \\ x_1^2 \\ x_2^2 \\ x_1 x_2 \end{bmatrix}$$

Nonlinear mappings

Note: we are still fitting a hyperplane in the new space defined by the projection function. This turns out to be nonlinear when back projected to the original space of \mathbf{x} .

Classification: we can do the same for logistic regression. In this case, the decision boundary is nonlinear when back projected into the original space.

Of course, we must somehow design the function $\varphi(\mathbf{x})$.

Feature engineering

Designing a good feature mapping function $\varphi(\mathbf{x})$ can be difficult!

- **Low dimensional:** less likely for data to be linearly separable
- **High dimensional:** possibly more prone to overfitting

Process of designing such functions by hand is called **feature engineering**.

Process of choosing features from a candidate set is called **feature selection**.

Good features are often the key to good generalization performance.

Feature engineering

$\varphi(\mathbf{x})$ can be (and often must be) a very complicated function of the data.

Examples

- Computer vision: $\varphi(\mathbf{x})$ could be the scale invariant feature transform (SIFT), followed by a bag-of-words encoding
- Speech recognition: $\varphi(\mathbf{x})$ could be mel frequency cepstral coefficients (MFCC)
- Text classification: $\varphi(\mathbf{x})$ could be a TF-IDF representation of the text
- $\varphi(\mathbf{x}) = \text{PCA transform}$
- $\varphi(\mathbf{x}) = \text{histogram}$
- $\varphi(\mathbf{x}) = \text{random projections}$

Feature learning

Feature engineering is notoriously difficult!

Alternative: learn features from the data directly.

Known as **representation learning**: $\varphi(\mathbf{x})$ produces a representation of the data in which it is easier to solve the relevant problem (classification, regression, etc.)

Can be done either:

- Unsupervised: from X alone with no labels, e.g.⁴⁸ PCA, clustering.
- Supervised: using both X and y .
- Semi-supervised: some labeled data, some unlabeled.

Deep learning is a very successful method for representation learning. Later...

Overfitting

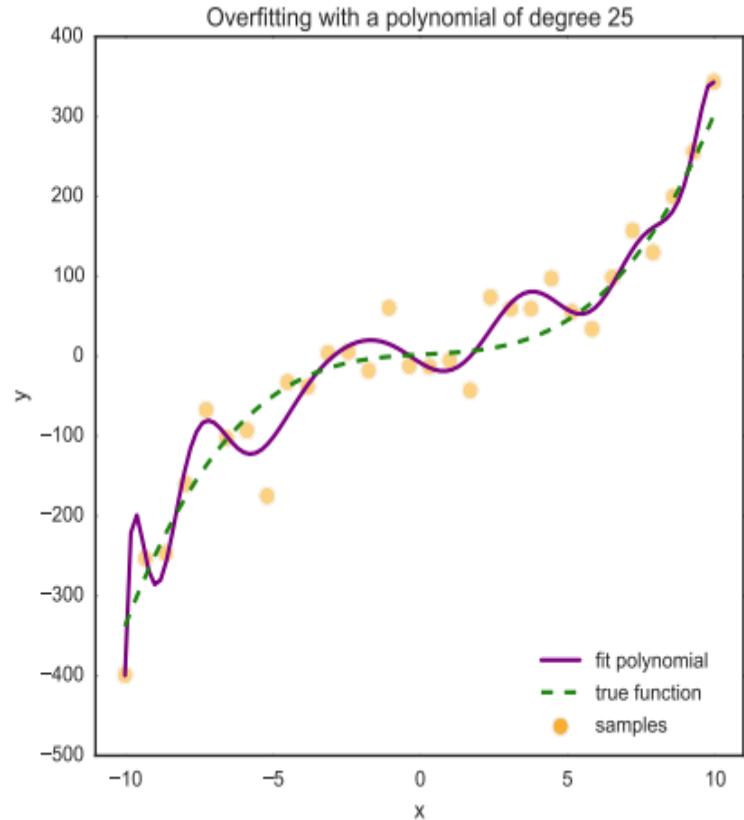
If the model has too many degrees of freedom, you can end up fitting not only the patterns of interest, but also the **noise**.

$$E[(y - \hat{f})^2] = \sigma^2 + \text{var}[\hat{f}] + \text{bias}[\hat{f}]$$

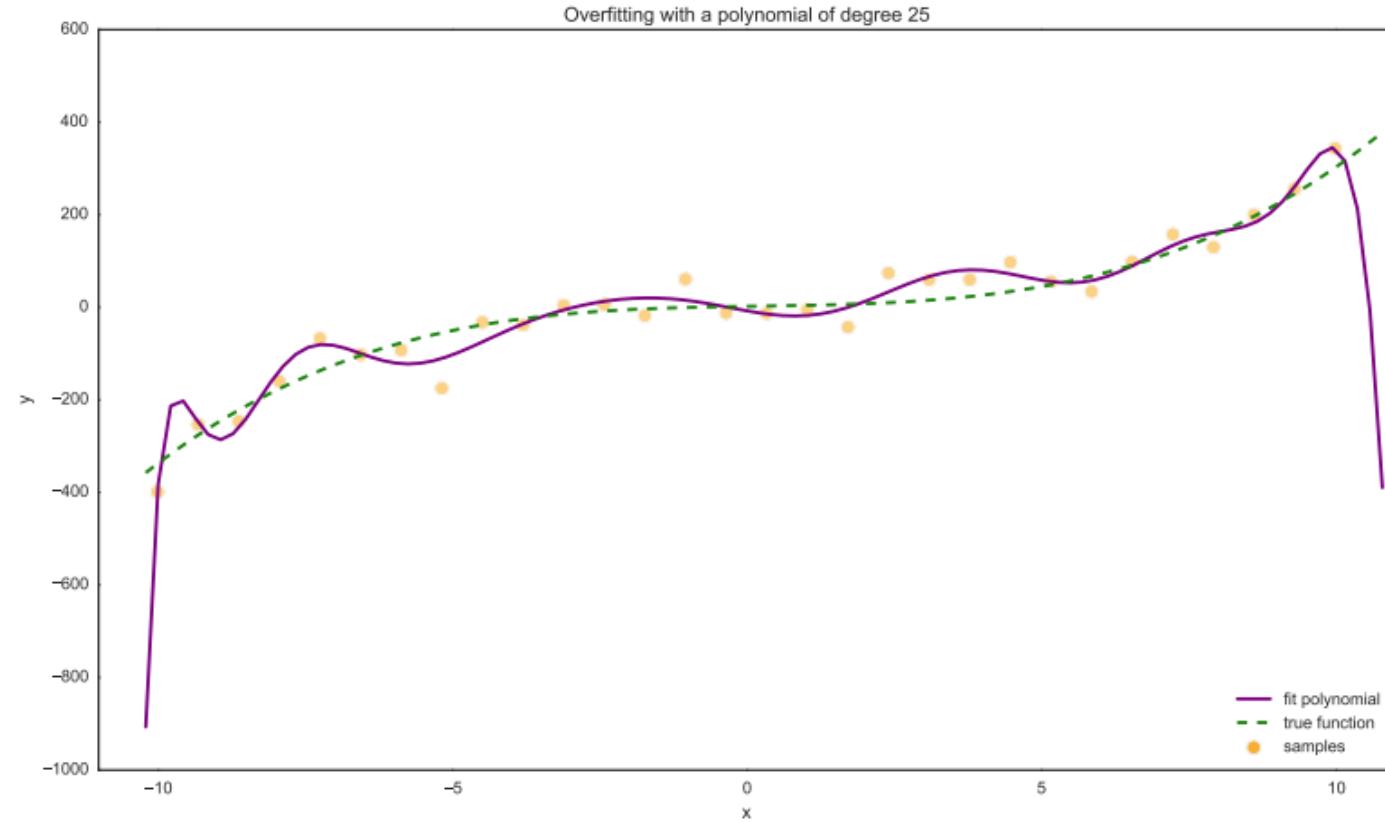
This leads to poor generalization: model fits the training data well but does poor on unseen data

Can happen when the model has too many parameters (and too little data to train on).

Remember **model selection**: use validation data (or cross validation) to check for overfitting!



Overfitting



Curse of dimensionality

Curses of dimensionality (large D):

1. **Estimation:** more parameters to estimate (risk of overfitting).
2. **Sampling:** exponential increase in volume of space.
3. **Optimization:** slower, larger space to search.
4. **Distances:** everything is far away.
5. Harder to model data distribution $P(x_1, x_2, \dots, x_D)$
6. Exponentially harder to compute integrals.
7. Geometric intuitions break down.
8. Difficult to visualize.

Blessings of dimensionality:

- **Linear separability:** easier to separate things in high- D using hyperplanes

Multiple outputs

Multiple regression

What if the target is not a scalar $y \in \mathbb{R}$ but actually a vector $\mathbf{y} \in \mathbb{R}^K$?

Easy. Just train K separate models, one for each target y_k .

Can solve the least squares problem in one shot by stacking the y_k values into a matrix:

$$\mathbf{Y} = \begin{bmatrix} | & | & & | \\ \mathbf{y}_1 & \mathbf{y}_2 & \dots & \mathbf{y}_K \\ | & | & & | \end{bmatrix} = \begin{bmatrix} y_{11} & y_{12} & y_{13} & \dots & y_{1K} \\ y_{21} & y_{22} & y_{23} & \dots & y_{2K} \\ \vdots & \vdots & \vdots & & \vdots \\ y_{N1} & y_{N2} & y_{N3} & \dots & y_{NK} \end{bmatrix}$$

and then solving the normal equations:

$$(\mathbf{X}^T \mathbf{X}) \mathbf{W} = \mathbf{X}^T \mathbf{Y}$$

$$\mathbf{W} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{Y}$$

Multi-class classification

Logistic regression is limited to **binary outputs** $\{0, 1\}$

Often if we want to do **multi-class** classification:

- Output is one of K classes $\{1, 2, \dots, K\}$

Examples:

- Handwritten digit recognition $\{0, 1, 2, 3, 4, \dots, 9\}$
- Audio: word/phoneme recognition
- Text classification: document is about $\{\text{politics}, \text{religion}, \text{sports}, \text{fashion}, \dots\}$
- Action classification: person is $\{\text{walking}, \text{running}, \text{sitting}, \text{standing}, \text{jumping}, \dots\}$
- Sign language recognition

Multi-class classification

How do we model this setting?

Two approaches:

- 1. One-vs-rest (OVR): aka one-vs-all**
- 2. Softmax regression**

One-vs-rest (OVR)

Want to train a logistic regression classifier for K classes.

Idea: train K separate binary classifiers.

E.g. classes are: $\{1, 2, 3\}$:

1. Train first classifier $f_1(\mathbf{x})$ with $y = 1$ for class 1 and $y = 0$ for classes $\{2, 3\}$.
2. Train second classifier $f_2(\mathbf{x})$ with $y = 1$ for class 2 and $y = 0$ for classes $\{1, 3\}$.
3. Train third classifier $f_3(\mathbf{x})$ with $y = 1$ for class 3 and $y = 0$ for classes $\{1, 2\}$.

At predict time, output class with **highest probability**:

$$\hat{y} = \arg \max_k f_k(\mathbf{x})$$

Softmax regression

In OVR classification the resulting probabilities do not sum to one (are not a distribution).

This corresponds to independent random output variables.

Sometimes this is what you want:

- multi-label classification: target may be more than one class
- target may be some other unseen class.

Sometimes it is not:

- Digit classification: target must be one of $\{0, 1, \dots, 9\}$. Never both. Never none.
- Target is a distribution: probabilities should sum to one $\sum_{i=1}^K y_i = 1$

Softmax regression

The latter can be achieved using **softmax regression**, which is the direct extension of logistic regression to the multi-class case.

Encode target values \mathbf{y} as a one-hot vector. E.g. $\mathbf{y} = (0, 0, 1, 0)$

The **softmax** activation is the extension of the sigmoid to the multi-class setting.

$$\hat{\mathbf{y}} = f(\mathbf{x}) = \text{softmax}(\mathbf{z})$$

with

$$\mathbf{z} = \begin{bmatrix} \mathbf{w}_1^T \mathbf{x} + b_1 \\ \mathbf{w}_2^T \mathbf{x} + b_2 \\ \vdots \\ \mathbf{w}_K^T \mathbf{x} + b_K \end{bmatrix} = \mathbf{W}\mathbf{x} + \mathbf{b}$$

Softmax regression

The softmax activation function is the analogue of the sigmoid for more than two classes:

$$\text{softmax}(\mathbf{x}) = \frac{1}{\sum_{j=1}^K \exp(x_j)} \begin{bmatrix} \exp(x_1) \\ \exp(x_2) \\ \vdots \\ \exp(x_K) \end{bmatrix}$$

Softmax regression

Loss function for softmax regression is the **categorical cross entropy**, which is the extension of binary cross entropy to categorical distributions:

$$\begin{aligned}\mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}) &= -\mathbf{y}^T \log \hat{\mathbf{y}} \\ &= -\sum_{j=1}^K y_i \log \hat{y}_i\end{aligned}$$

Unlike logistic regression, softmax regression is **overparameterized**:

- probabilities must sum to one,
- one extra set of weights and biases than needed,
- L_2 regularization is important to ensure unique minimizer.

Summary

Properties of linear models

Require $(D + 1)$ parameters for D features.

Strong assumptions mean they have **high bias**.

Have fairly **low variance**, but still possible to overfit when D is large relative to N .

Regularization can be used to reduce size of hypotheses space and help prevent overfitting.

Very fast at predict time: just a linear function.

Interpretable: weights specify feature importance (careful with this).

Further reading

The elements of statistical learning:

- Chapter 3: Linear methods for regression
- Chapter 4: Linear methods for classification

Resources

Stanford machine learning lectures (Andrew Ng):

- Lecture 3: linear and logistic regression
<http://www.youtube.com/watch?v=HZ4cvaztQEs>
- Lecture 4: Generalized linear models
<http://www.youtube.com/watch?v=nLKOQfKLUks>

Caltech machine learning lectures (Yaser Abu-Mostafa):

- Lecture 3: The linear model 1
[http://www.youtube.com/watch?v=FIbVs5GbBlQ⁶⁴](http://www.youtube.com/watch?v=FIbVs5GbBlQ)
- Lecture 9: The linear model 2
<http://www.youtube.com/watch?v=qSTHZvN8hzs>
- Lecture 12: Regularization
<http://www.youtube.com/watch?v=I-VfYXzC5ro>

Resources

Oxford deep learning lectures (Nando de Freitas):

- Lecture 2: Linear models

<http://www.youtube.com/watch?v=DHspIG64CVM>

- Lecture 3: Maximum likelihood

<http://www.youtube.com/watch?v=kPrHqQzCkg0>

- Lecture 4: Regularization 1

http://www.youtube.com/watch?v=VR0W_PNwLGw

- Lecture 5: Regularization 2

http://www.youtube.com/watch?v=VR0W_PNwLGw

Measuring accuracy

To evaluate our model (evaluation, test) or to choose a model (model selection, validation) we need a metric to measure how well our model is performing.

Depending on the task, different metrics may be appropriate

- Binary classification
- Multiclass classification
- Regression
- Detection

Evaluating classification accuracy

Types of errors for binary classifiers

- **Type 1 error:** predict true when false
- **Type 2 error:** predict false when true

Can categorize predictions wrt ground truth as:

1. True positives (TP): $y_{\text{true}} = 1, y_{\text{pred}} = 1$
2. True negatives (TN): $y_{\text{true}} = 0, y_{\text{pred}} = 0$
3. False positives (FP): $y_{\text{true}} = 0, y_{\text{pred}} = 1$
4. False negatives (FN): $y_{\text{true}} = 1, y_{\text{pred}} = 0$

		y_{PRED}	
		T	F
y_{TRUE}	T	TP	FN
	F	FP	TN

Metrics for classification

Classification accuracy: proportion of correctly classified results

```
np.mean(y_true == y_pred)
```

Can also be written in terms of TP, FP, TN, FN

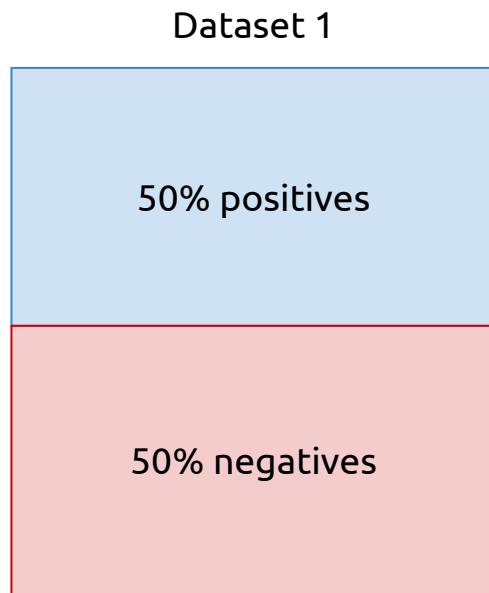
$$\text{ACC} = (\text{TP} + \text{TN}) / (\text{TP} + \text{TN} + \text{FP} + \text{FN})$$

Also known as **zero-one accuracy**. Zero-one error = 1 - ACC

		y _{PRED}	
		T	F
y _{TRUE}	T	TP	FN
	F	FP	TN

Considerations for using accuracy

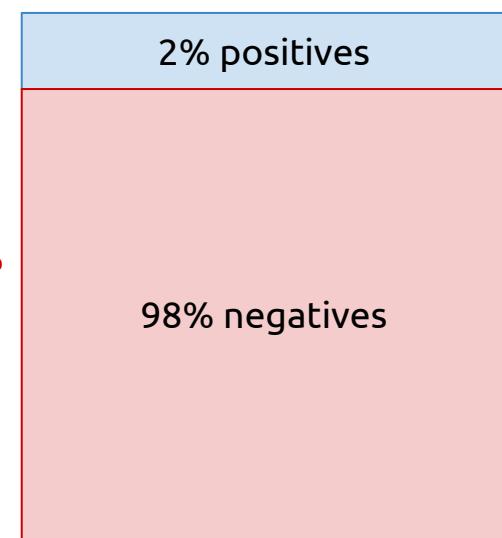
If the dataset is not balanced, then the value of accuracy may be misleading!



Constant classifier

$$f(\mathbf{x}) = 0$$

ACC: 50%



ACC: 98%

Balanced accuracy

Can be beneficial to use balanced accuracy when you have many more of one class than the other.

$$\text{Balanced Accuracy} = \frac{1}{2} \left(\frac{TP}{P} + \frac{TN}{N} \right)$$

Proportion of total negatives marked as negative

Proportion of total positives marked as positive

Same as accuracy when dataset is balanced

		y _{PRED}	
		T	F
y _{TRUE}	T	TP	FN
	F	FP	TN

Precision and recall

Consider a **search engine** that classifies documents as {relevant, non-relevant} based on a query

In addition to accuracy, would also be useful to know:

- What proportion of returned results are actually relevant
- What proportion of all relevant results are returned

Two metrics you can use for this are **precision** and **recall**

Precision

The proportion of elements marked as positive that are correct.

$$\text{Precision} = \frac{TP}{P} = \frac{TP}{TP + FP}$$

Information retrieval: proportion of documents returned that are relevant

Also known as PPV: positive predictive value

		y_{PRED}	
		T	F
y_{TRUE}	T	TP	FN
	F	FP	TN

Recall

The proportion all true events that are marked as true

$$\text{Recall} = \frac{TP}{T} = \frac{TP}{TP + FN}$$

Information retrieval: proportion of all relevant documents that were found by the system

AKA: true positive rate (Sensitivity)

		y_{PRED}	
		T	F
y_{TRUE}	T	TP	FN
	F	FP	TN

Precision recall curve

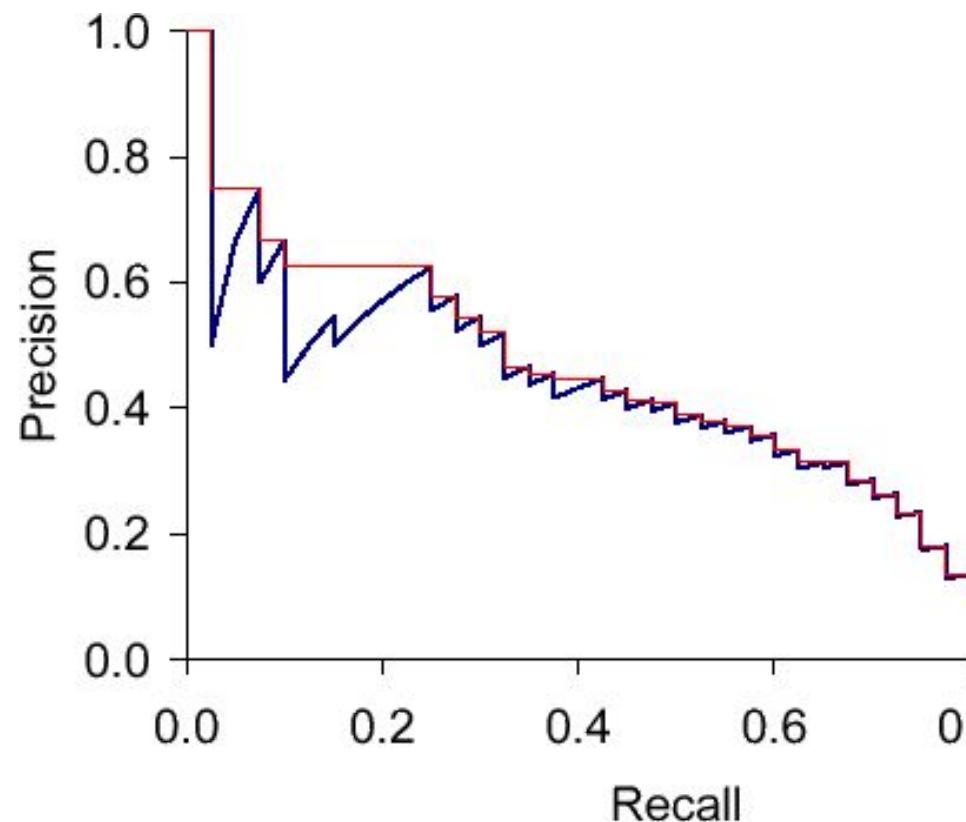
Easy to get perfect precision. $f(x) = 0$ (return no document)

Easy to get perfect recall. $f(x) = 1$ (return every document)

Often our classifier produces a score. Thresholding the score at different values will give different precision and recall values.

The curve traced out by all possible thresholds is called the **precision recall curve**

Precision recall curve



F₁ scores

Sometimes it is useful to have a single metric that combines precision and recall.
The F₁ score is the harmonic mean of precision and recall.

$$F_1 = \frac{2(\text{Precision} \times \text{Recall})}{\text{Precision} + \text{Recall}}$$

Can also be calculated at many points on the PR curve (i.e. as a function of threshold)

Average precision

Precision is a function of the threshold you set: $p(t)$

Integrating over all possible thresholds [0..1] gives the area under the precision curve, known as average precision.

$$\text{AveP} = \int_0^1 p(t)dt$$

Can be computed by ranking by classifier score and computing the average value of precision@k

Notes on precision and recall based metrics

Precision and recall both use **true positives** proportions.

Assumes there is something special about the +1 class. Often true, e.g. for:

- Document retrieval
- Face detection
- Intrusion detection

But may not always be true. E.g. classify image as {dog, cat}. Which is the true case?

$$\text{Precision} = \frac{\textcircled{TP}}{P} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{\textcircled{TP}}{T} = \frac{TP}{TP + FN}$$

True and false positive rates

True positive rate (same as Recall):

$$\text{TPR} = \text{TP} / (\text{TP} + \text{FN})$$

False positive rate:

$$\text{FPR} = \text{FP} / (\text{FP} + \text{TN})$$

Using these to draw a curve is not biased toward the +1 class...

		y_{PRED}	
		T	F
y_{TRUE}	T	TP	FN
	F	FP	TN

True and false positive rates

Example: intrusion detection system

- **True positive rate** tells you how often your system will raise the alarm when there is indeed a burglar present $P(\text{Alarm}|\text{Burglar})$
- **False positive rate** tells you how often your system will raise the alarm when there is no burglar present $P(\text{Alarm}|\text{No burglar})$

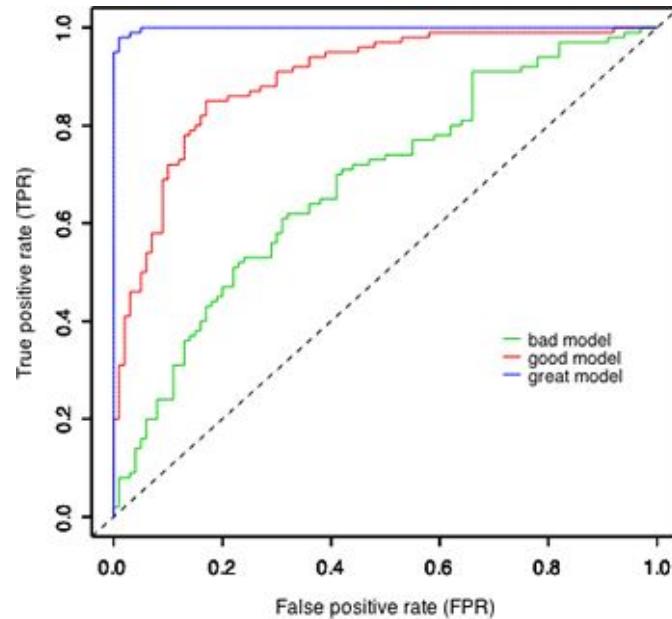
The receiver operator characteristic

ROC curve. Graph of TPR vs FPR as you vary the threshold.

- Sensitivity vs. 1-Specificity
- $P(\text{true}|\text{true})$ vs. $P(\text{true}|\text{false})$

Called ROC for historical reasons (radar)

Different applications may prefer different points on the ROC



AUC

To reduce the ROC curve to a single metric, you can measure the **area under the curve**

This is called the AUC metric (or ROC-AUC)

AUC represents performance averaged over all possible cost ratios

AUC	Interpretation
1.0	perfect prediction
0.9	excellent prediction
0.8	good
0.7	mediocre
0.6	poor
0.5	random
<0.5	bug in code!

Multiclass classification and error analysis

For multiclass problems we can compute overall classification accuracy.

Can also compute binary metrics for each class, treating all other classes as negatives

In this case, the positive class is special, so it makes sense to use precision and recall

```
import sklearn.metrics as metrics  
  
metrics.classification_report(y_true, y_pred)
```

class	precision	recall	f1-score	support
0	1	1	1	6451190
1	0.9	0.91	0.9	616014
2	0.94	0.95	0.94	89966
3	0.93	0.93	0.93	609634
4	0.62	0.52	0.57	6956
5	0.85	0.88	0.87	24057
6	0.95	0.96	0.96	122025
7	0.92	0.91	0.92	36384
8	0.93	0.9	0.91	456774
avg/tot	0.98	0.98	0.98	8413000

Multiclass classification and error analysis

The **confusion matrix** can be used to analyse which classes are most often mixed up by the classifier

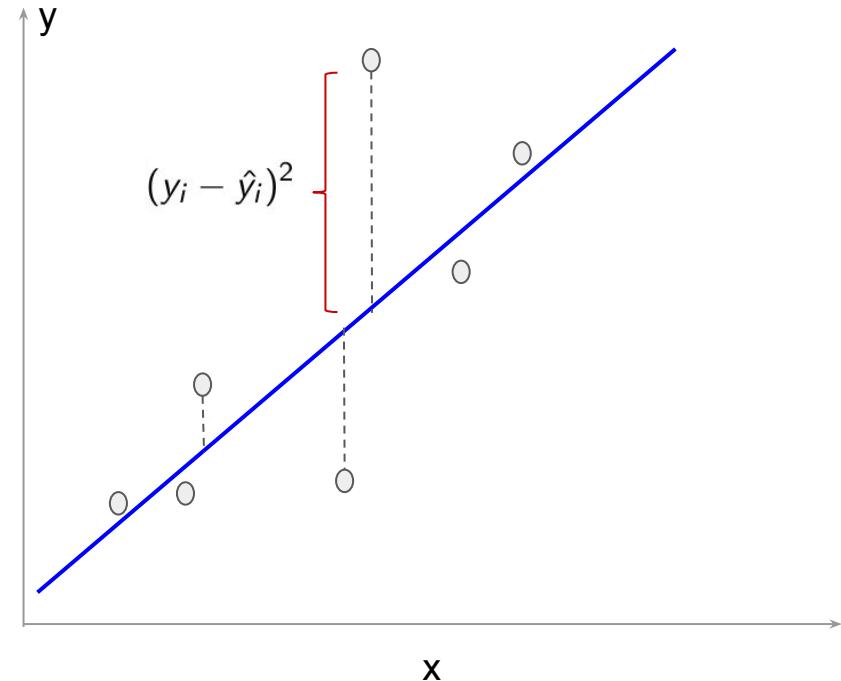
Off diagonal elements represent errors.

	0	1	2	3	4
0	1476	34	184	4	0
1	490	47	235	17	0
2	268	32	590	90	0
3	6	6	132	499	53
4	0	0	0	52	185

Metrics for regression

Mean squared error (MSE) is standard for regression problems.

$$E = \frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2$$

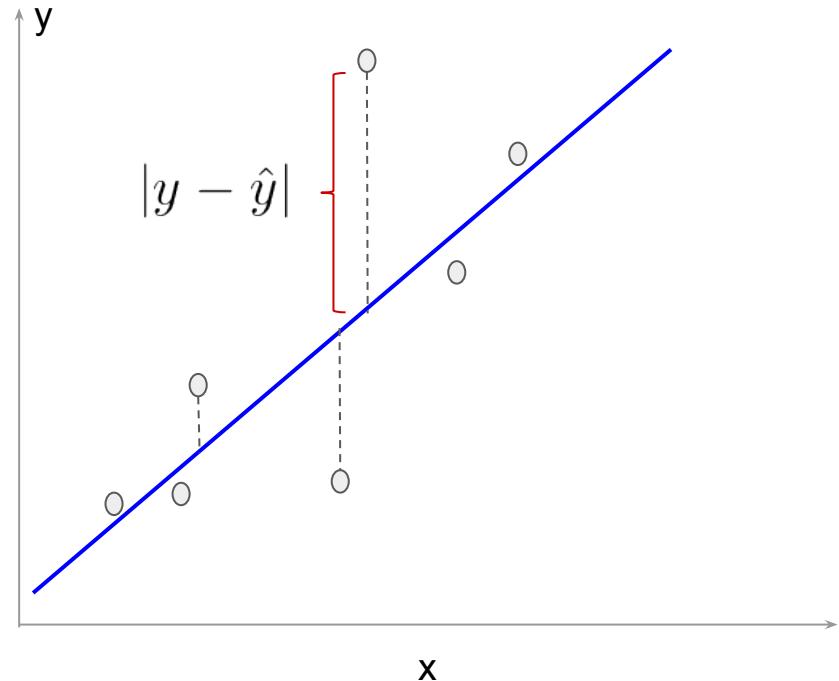


Mean absolute error

Mean squared error heavily penalizes points that are far from the prediction

Mean absolute error (MAE) should be used in situations where you want to be more robust to outliers (e.g. noisy ground truth)

$$E = \frac{1}{N} \sum_{i=1}^N |y_i - \hat{y}_i|$$



EE1083/EEN1085

Data analysis and machine learning

Ali Intizar



Ollscoil Chathair
Bhaile Átha Cliath
Dublin City University

Unsupervised Learning



Ollscoil Chathair
Bhaile Átha Cliath
Dublin City University



Overview

Motivation and goals of unsupervised learning

Clustering

- K-Means
- Agglomerative clustering

Motivation

- Vast amounts of unlabelled data
- Most data has structure; we would like to discover hidden structure
- Modelling the **probability density** of the data $P(X)$
- Fighting the **curse of dimensionality**
- **Visualizing** high-dimensional data
- Supervised learning tasks: learning from fewer training examples

Assumptions

It is necessary to make some assumptions to learn structure from data.

“You can’t do inference without making assumptions”

-- David MacKay, Information Theory, Inference, and Learning Algorithms

Typical assumptions:

- Smoothness assumption
 - Points which are close to each other are more likely to share semantics.
- Cluster assumption
 - The data form discrete clusters; points in the same cluster are likely to share semantics
- Manifold assumption
 - The data lie approximately on a manifold of much lower dimension than the input space.

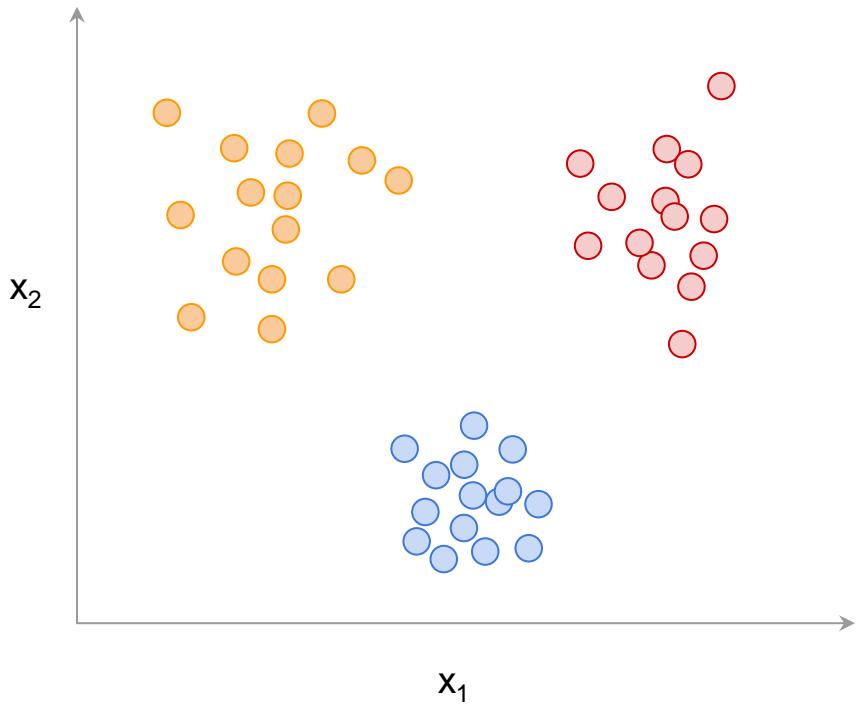
Clustering

Cluster assumption: data form discrete clusters.

We would like to come up with an algorithm to automatically discover these clusters from data.

Will look at two approaches:

1. **K-Means:** formulate as an optimization problem. Find approximate solution using iterative algorithm.
2. **Agglomerative clustering:** iterative greedy bottom-up algorithms that produce a hierarchical clustering



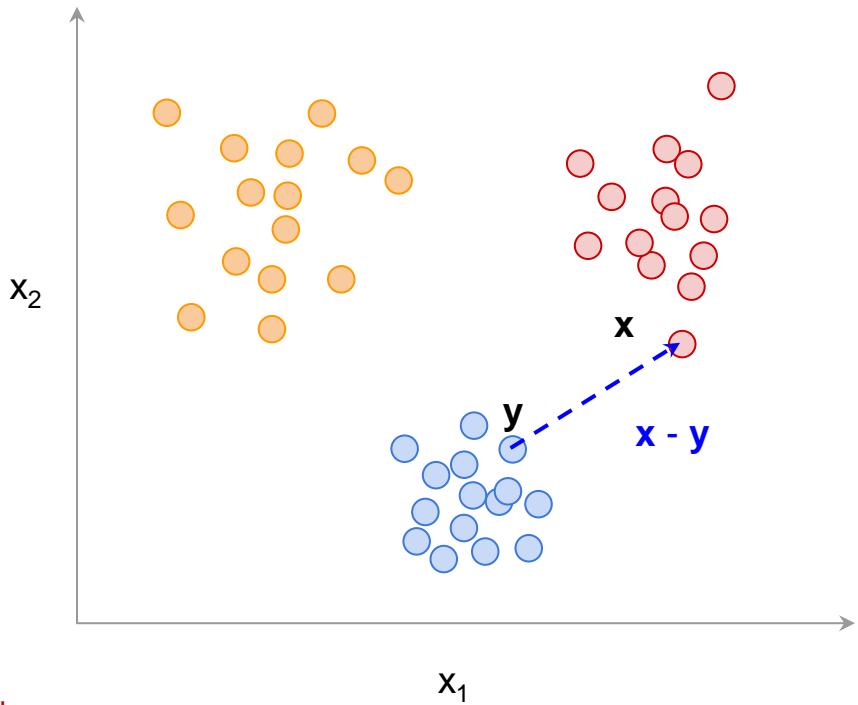
K-Means

Simple but very popular clustering algorithm

Aims to find a fixed number (k) of discrete clusters such that the average distance from a point to the center of its cluster is minimized.

Distance is taken to be the square Euclidean distance.

$$\begin{aligned} d(\mathbf{x}, \mathbf{y}) &= \|\mathbf{x} - \mathbf{y}\|_2^2 \\ &= (\mathbf{x} - \mathbf{y})^T (\mathbf{x} - \mathbf{y}) \\ &= (x_1 - y_1)^2 + (x_2 - y_2)^2 + \dots \end{aligned}$$



K-Means

k-means objective:

$$\arg \min_{\mathbf{S}} \sum_{i=1}^k \sum_{\mathbf{x} \in S_i} \|\mathbf{x} - \boldsymbol{\mu}_i\|^2$$

where:

- $\mathbf{S} = \{\mathbf{S}_1, \mathbf{S}_2, \dots, \mathbf{S}_k\}$ is the set of non-overlapping clusters assignments
- \mathbf{S}_i is set of all points in cluster i ,
- $\boldsymbol{\mu}_i$ is centroid of cluster i (mean of all points in \mathbf{S}_i)

Discrete optimization problem: objective function is non-smooth and **non-convex**.

Objective is **NP-hard** even in 2D: impossible to solve in polynomial time.

K-Means algorithms attempt to find an **approximate solution** (local minimum of the objective function) in polynomial time.

Lloyd's algorithm

Iterative approach for finding a local minimum of the k -means objective.

Idea: start with randomly cluster centres. At each iteration, move them to reduce the cost.

Algorithm

Start with k random chosen cluster centers

While not converged:

1. **Assign** each point (descriptor) to nearest cluster center
2. **Update** cluster center to centroid of all points assigned to it

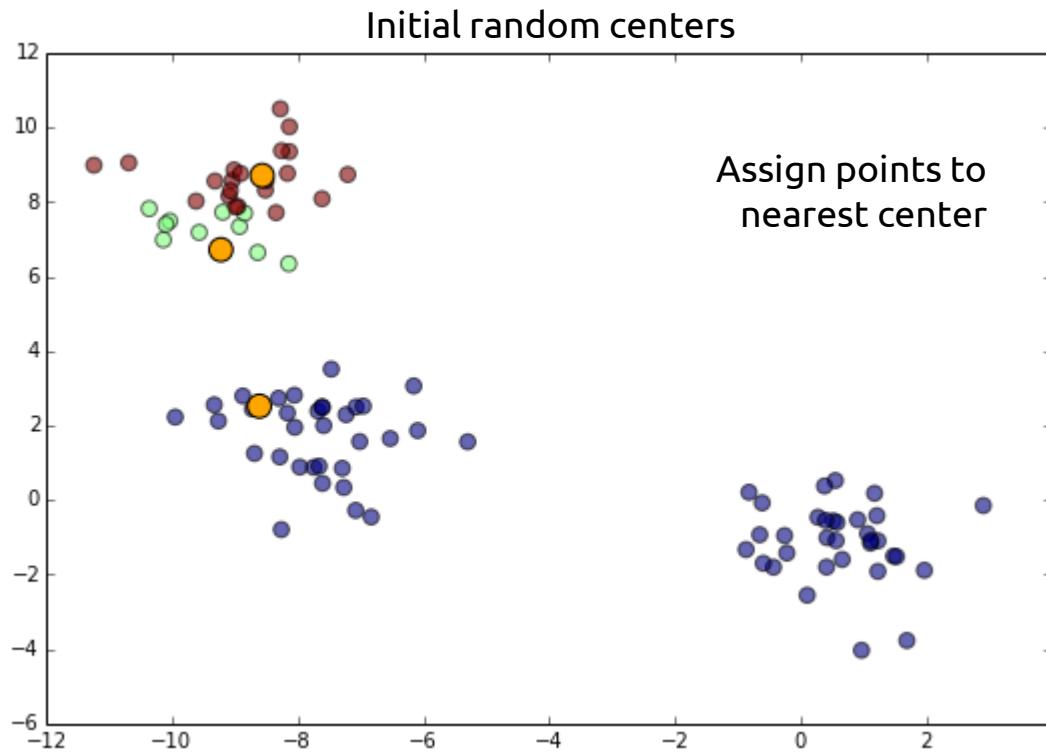
Convergence is when assignments don't change or change in position of cluster centers is not significant

Example of a two-step **coordinate descent** algorithm:

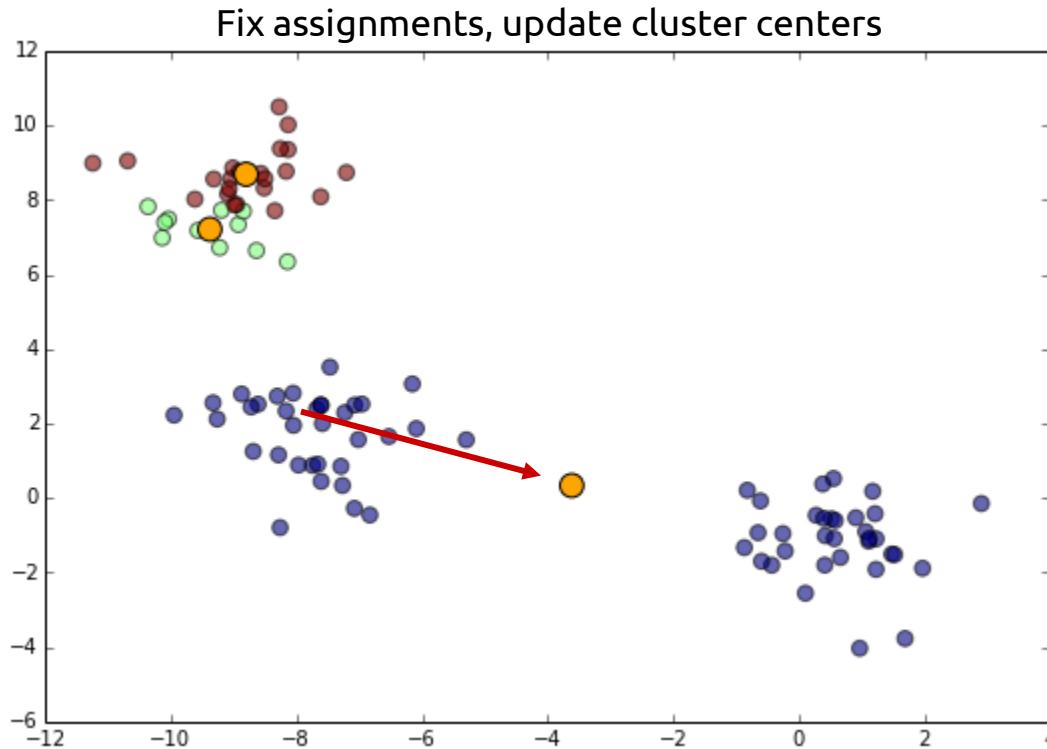
- **Step 1:** fix the cluster centers, find the optimal assignments
- **Step 2:** fix the assignments, find the optimal cluster centers

Coordinate descent: fix A, minimize B, fix B, minimize A, ...

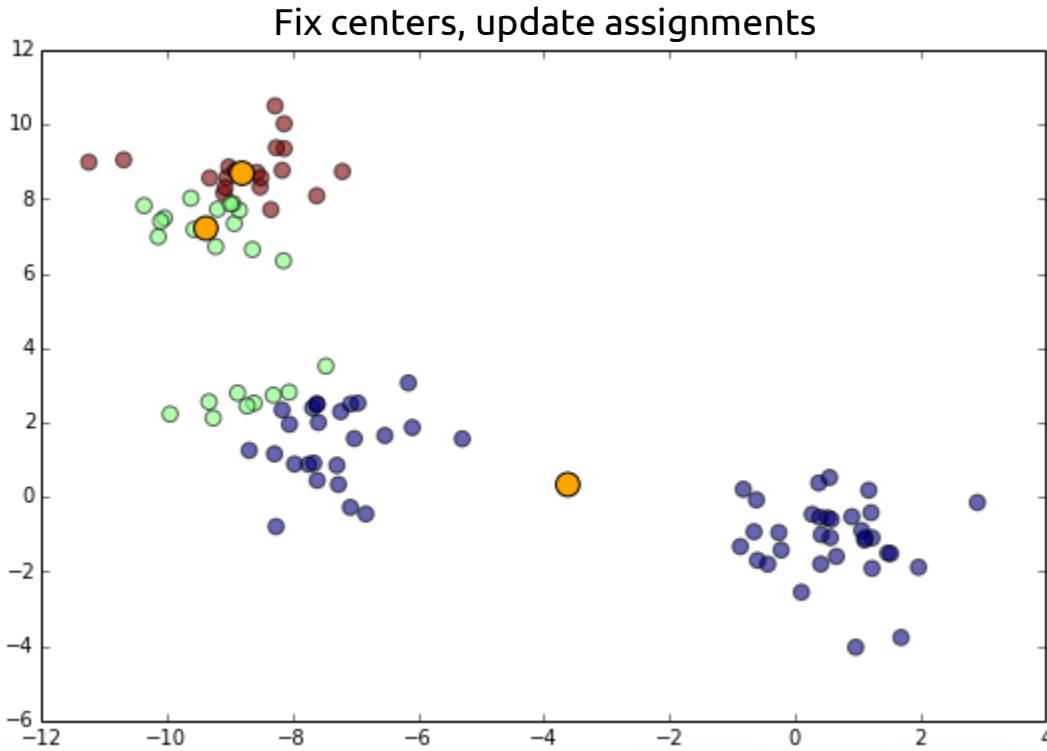
Demo of Lloyd's algorithm



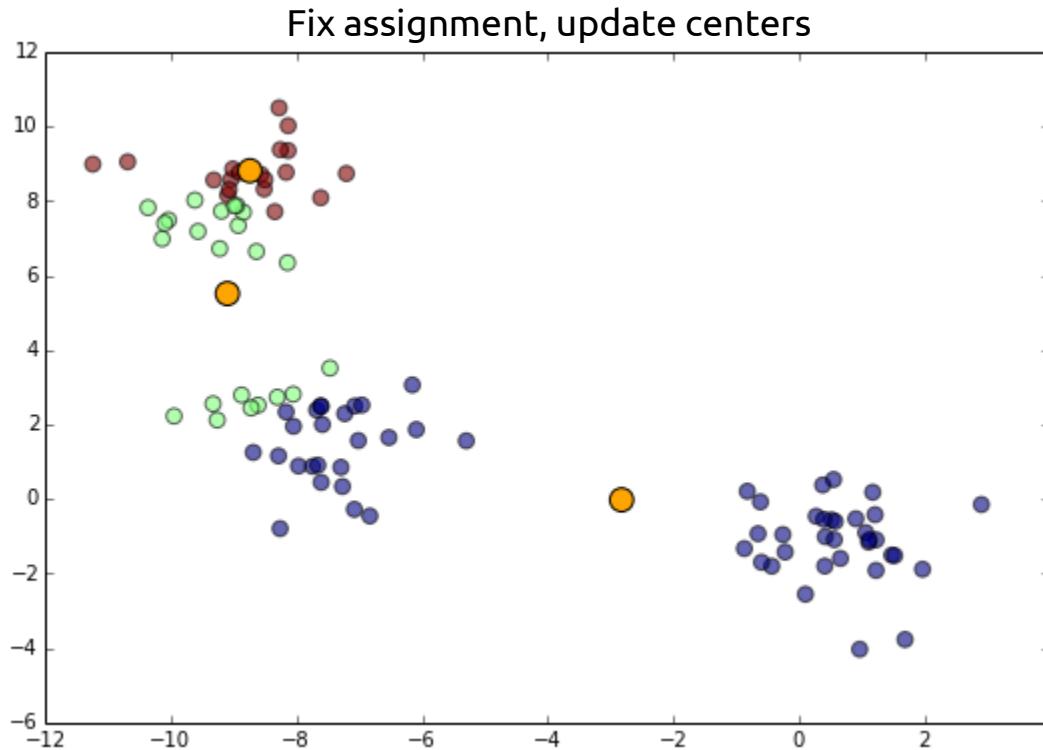
Demo of Lloyd's algorithm



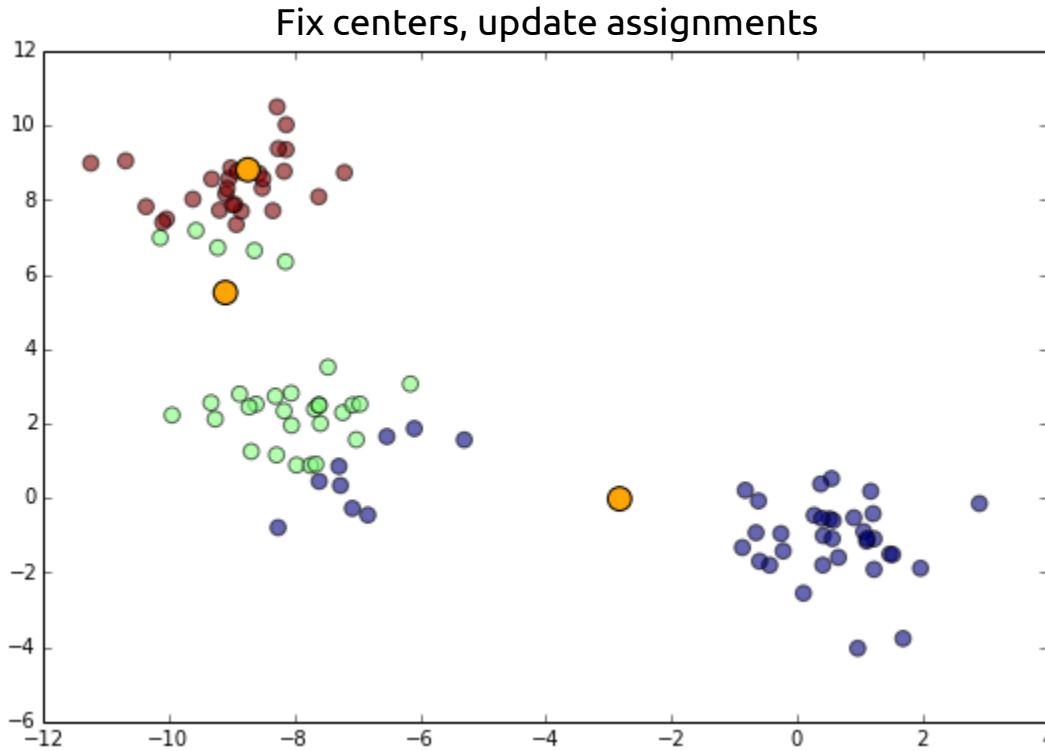
Demo of Lloyd's algorithm



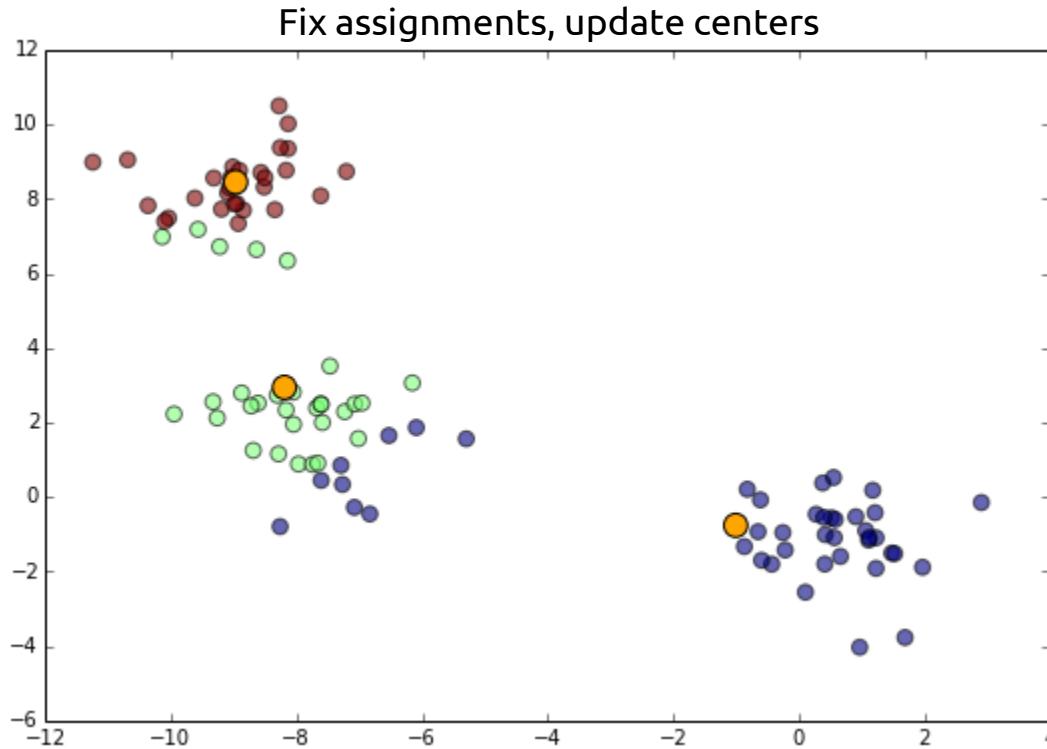
Demo of Lloyd's algorithm



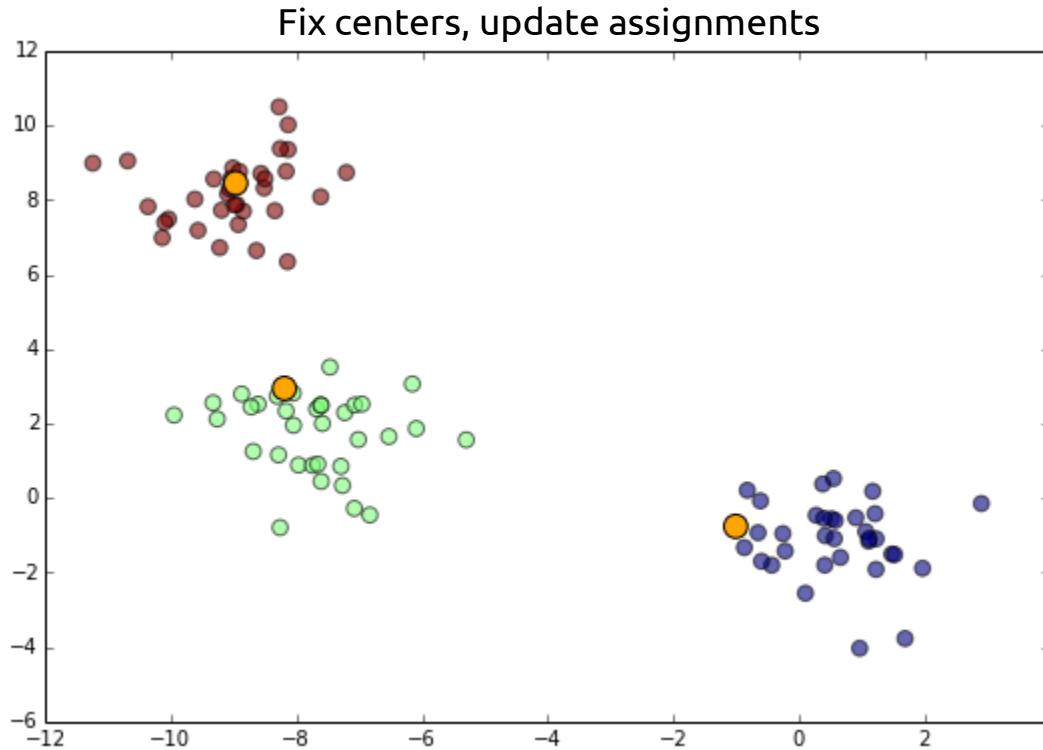
Demo of Lloyd's algorithm



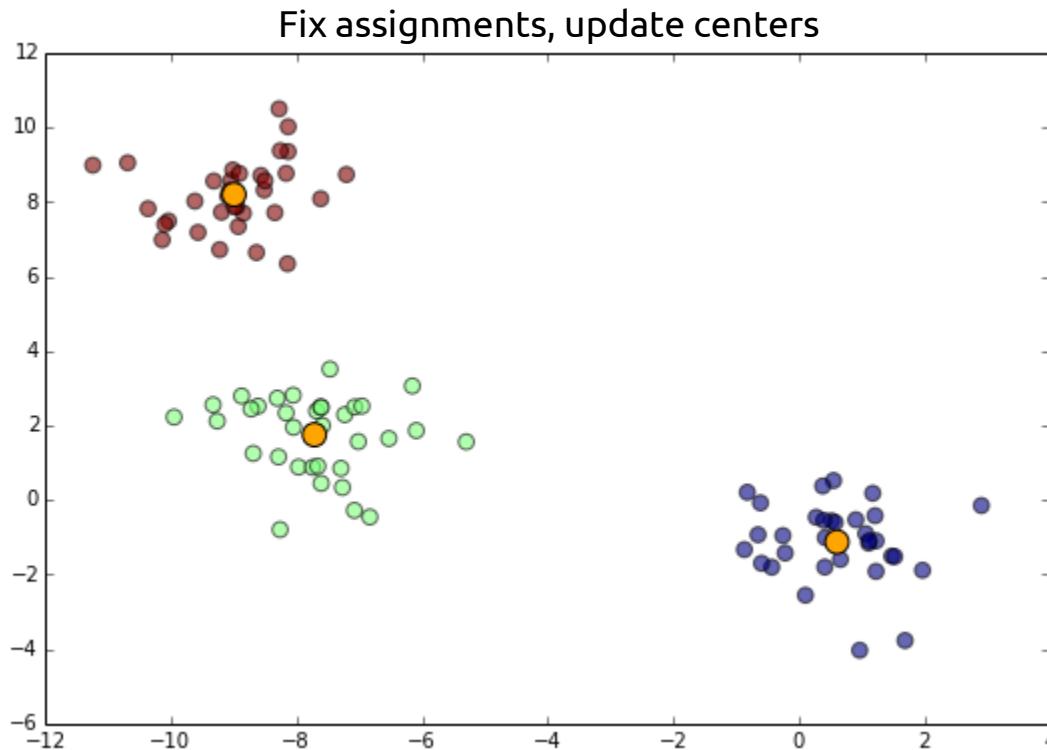
Demo of Lloyd's algorithm



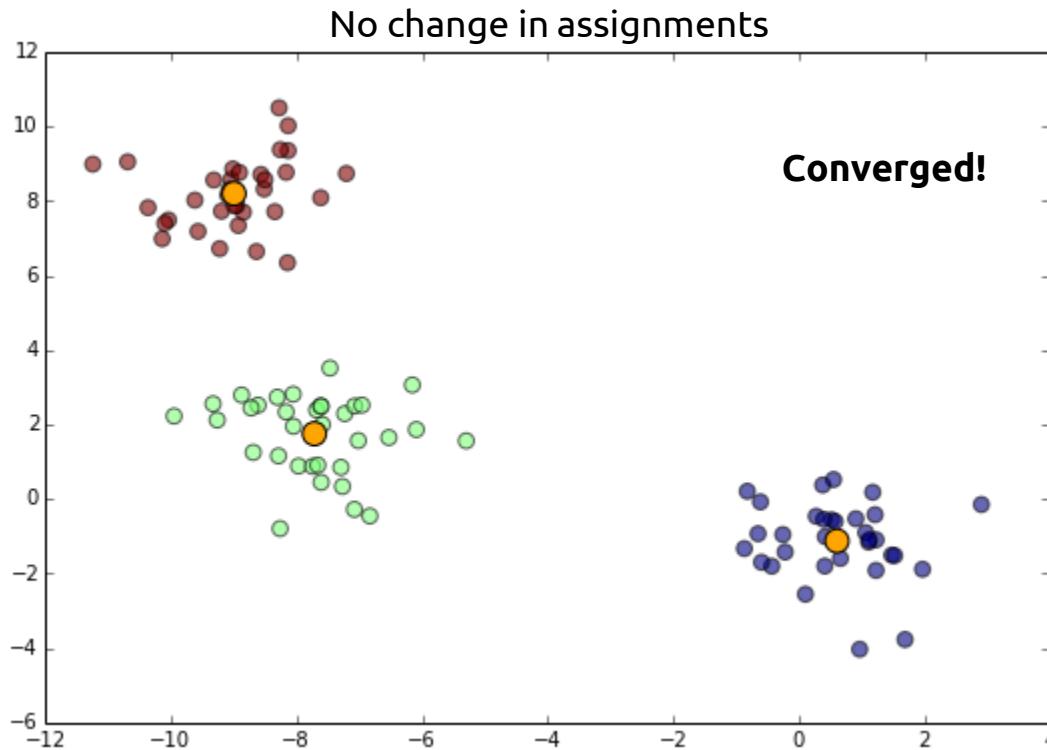
Demo of Lloyd's algorithm



Demo of Lloyd's algorithm



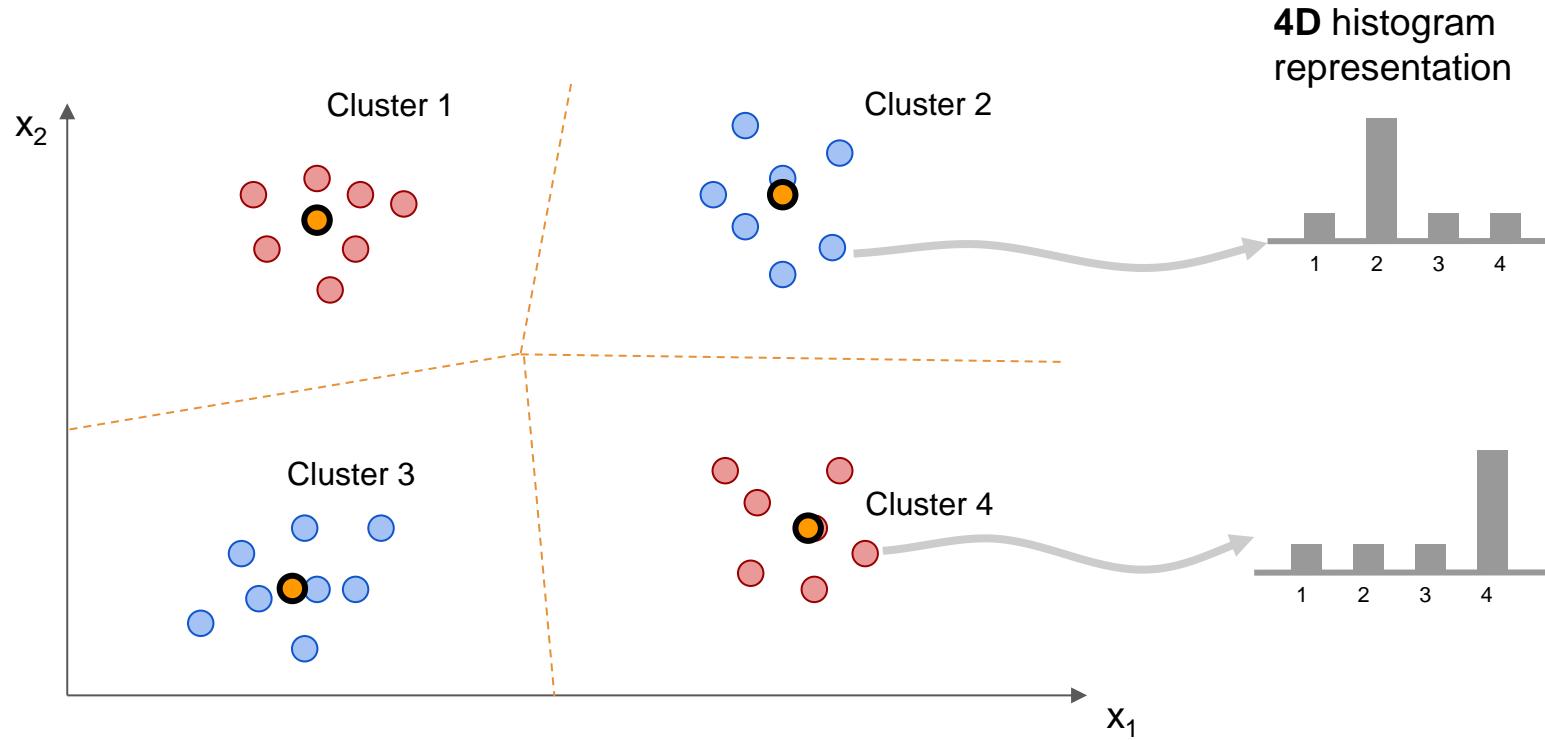
Demo of Lloyd's algorithm



Codebook representation

- Possible now to represent a data point using the index of nearest cluster center. This is called **vector quantization**.
- Distances to nearest cluster center induces a **Voronoi tessellation** of the space.
- In this context, the set of cluster centers is often called the **codebook**.
- Easy to compute a histogram of the counts of points assigned to each cluster center.

Vector quantization



Speeding up distance computations

At each step in k -means we need to compute the distance from every data point to every cluster center.

$$d(\mathbf{x}_i, \mu_j) = (\mathbf{x}_i - \mu_j)^T (\mathbf{x}_i - \mu_j)$$

Possible to use binomial expansion to speed this up:

Can be computed very quickly
for all centers and data points
using a matrix multiplication

$$d(\mathbf{x}_i, \mu_j) = \mathbf{x}_i^T \mathbf{x}_i + \mu_j^T \mu_j - 2\mathbf{x}_i^T \mu_j$$

Doesn't change when centers
change: compute once for all
data points at start

Quick to compute when number
of cluster centers is relatively
small

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1^T \\ \vdots \\ \mathbf{x}_N^T \end{bmatrix}, \quad \mathbf{C} = [\mu_1 \quad \cdots \quad \mu_K], \quad \langle \mathbf{x}_i, \mu_j \rangle = (\mathbf{X}\mathbf{C})_{ij}.$$

$$[2\mathbf{x}_i^T \mu_j] = 2\mathbf{X}\mathbf{C}^T$$

K-Means code

```
def kmeans(data, k=3, max_iters=100):  
  
    # precompute norms for fast distance computations  
    data_norms = np.sum(data**2, axis=1)  
  
    def get_distances_to_centers(centers):  
  
        # binomial trick for fast distances  
        center_norms = np.sum(centers ** 2, axis=1)  
        dists = -2 * np.dot(data, centers.T)  
        dists += center_norms[:,np.newaxis,:]  
        dists += data_norms[:,np.newaxis]  
        return dists  
  
    def get_assignments(centers):  
        dists = get_distances_to_centers(centers)  
        assignments = np.argmin(dists, 1)  
        return dists, assignments  
  
    def get_updated_centers(assignments):  
        centers = [data[assignments==i,:].mean(axis=0)  
                  for i in xrange(k)]  
        return np.array(centers)
```

```
def get_initial_centers():  
    N = data.shape[0]  
    indices = np.random.randint(0, N, k)  
    return data[indices, :]  
  
# start off with initial random centers  
centers = get_initial_centers()  
prev_assignments = None  
for i in xrange(max_iters):  
  
    # fix centers and update assignments  
    dists, assignments = get_assignments(centers)  
  
    # fix assignments and update centers  
    centers = get_updated_centers(assignments)  
  
    # if nothing changes, we have converged  
    if prev_assignments is not None:  
        if np.sum(assignments != prev_assignments) == 0:  
            break  
  
    prev_assignments = assignments  
  
return assignments, centers
```

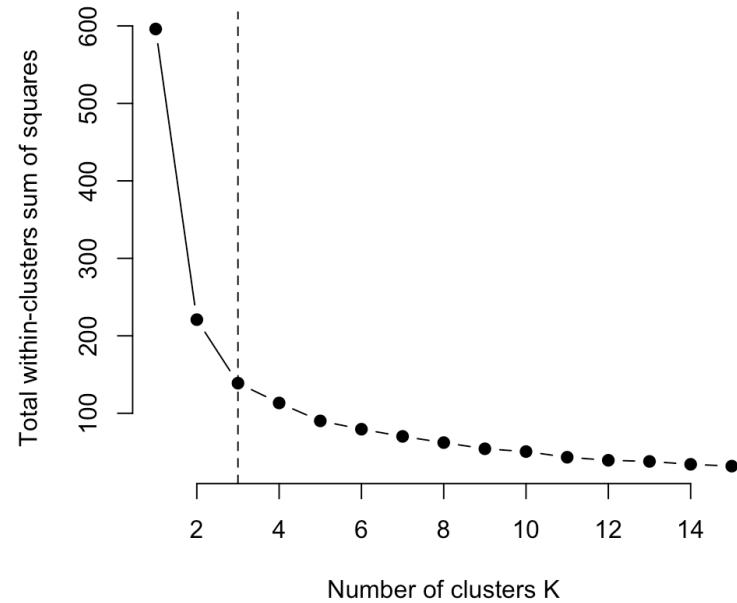
Choosing k

Typical heuristic:

- Plot within cluster sum of squares for different values of k
- Choose k at the “elbow” in the plot

The gap statistic

- More formal way of finding the elbow algorithmically
- **Idea:** generate random datasets and compare within cluster sum of squares of clusterings on random data with real data as k increases.



The gap statistic algorithm

1. Generate B randomly distributed datasets that are uniformly distributed over the ranges of the original attributes.
2. For increasing values of k :
 - 2.1 Using k centers, cluster original and reference datasets.
 - 2.2 Calculate the log point scatter $\log(W_k)$ for clusterings on the original dataset
 - 2.3 Calculate the log point scatter $\log(W_{kb}^*)$ for clusterings on the reference datasets
 - 2.4 Calculate the gap statistic: $G_k = \mathbf{E}_B[\log(W_{kb}^*)] - \log(W_k)$
3. Choose $\hat{k} = \text{smallest } k \text{ such that } G_k \geq G_{k+1} - s_{k+1}$, where $s_k = \sigma_k \sqrt{1 + 1/B}$, with σ_k being the standard deviation of the log point scatter on the reference datasets for k clusters.

Robert Tibshirani, Guenther Walther, Trevor Hastie, *Estimating the number of clusters in a data set via the gap statistic*, Journal of the Royal Statistical Society: Series B (Statistical Methodology), vol. 63 (2), pp. 411-423, 2001

K-Means notes

Sensitive to initialization: different initializations can produce different clusterings

- Can run several times with different initializations
- Use clustering with minimum cost function value

Alternative initialization strategies: [kmeans++](#)

1. Choose first seed at random
2. Find distance to all other points
3. Choose next seed randomly with probability proportional to distance
4. Repeat step 2 and 3 until all seeds chosen

Approximate k-means: possible to make k -means practical for very large datasets ([#bigdata](#)) by using a **fast approximate nearest neighbour algorithm (e.g. FAISS)**

- When computing cluster assignments we need to find the nearest cluster centre (nearest neighbour)

Overview

Motivation and goals of unsupervised learning

Clustering

- K-Means
- Agglomerative clustering

Agglomerative clustering

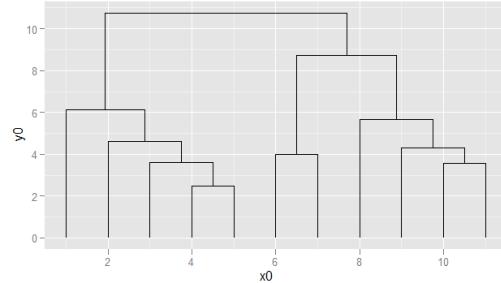
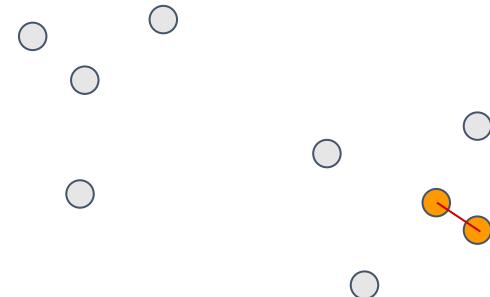
Bottom-up **recursive hierarchical clustering** of points.

Need to specify two things:

1. Distance measure (e.g. Euclidean)
2. Linkage type

Linkage types

1. Single linkage (min)
2. Complete linkage (max)
3. Average linkage (mean)
4. Centroid linkage
5. Ward linkage



Dendrogram

Linkage

Single linkage: distance between two clusters is the **minimum** distance between any single data point in the first cluster and any single data point in the second cluster

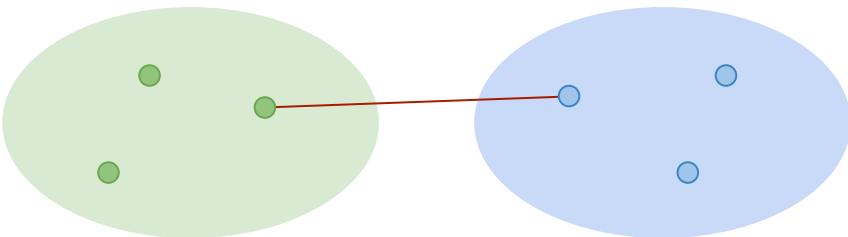
Complete linkage: the distance between two clusters is the **maximum** distance between any single data point in the first cluster and any single data point in the second cluster

Average linkage: the distance between two clusters is the **average** distance between data points in the first cluster and data points in the second cluster.

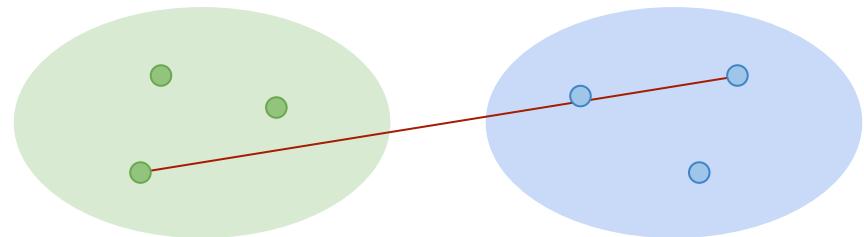
Centroid method: the distance between two clusters is the distance between the two mean vectors (centroids) of the clusters.

Ward's method: merge the clusters that create the smallest increase in cluster variance.

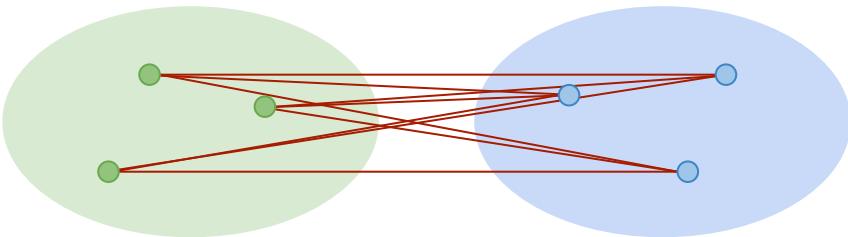
Single linkage



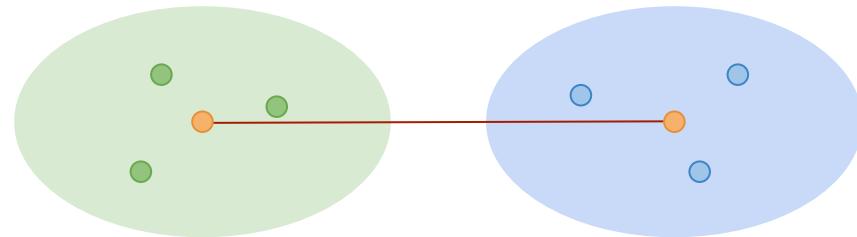
Complete linkage



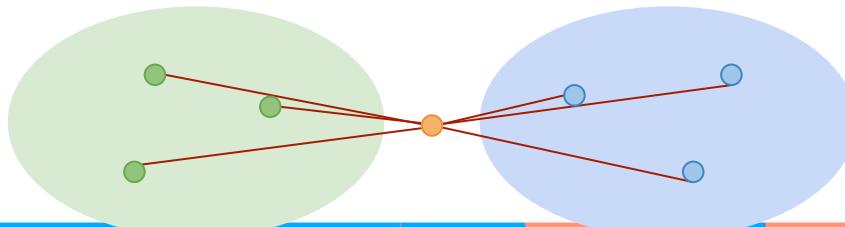
Average linkage



Centroid linkage



Ward linkage



Overview

Motivation and goals of unsupervised learning

Clustering

- K-Means
- Agglomerative clustering

Dimensionality reduction

- PCA
- t-SNE
- Other methods

Dimensionality Reduction

Assumption: the data lie close to a lower dimensional **manifold** embedded in a high dimensional space.

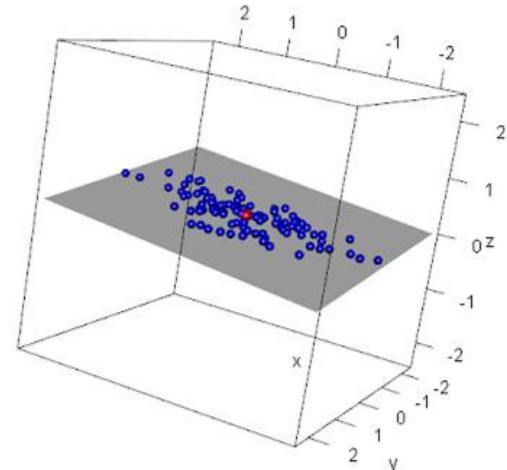
Linear dimensionality reduction: further assume that this manifold is a hyperplane.

- 1D: a line
- 2D: a plane
- N-D: hyperplane

We would like to recover the coordinates of the data on this manifold.

Why?

- Visualization
- Compression
- Summarization
- Supervised learning

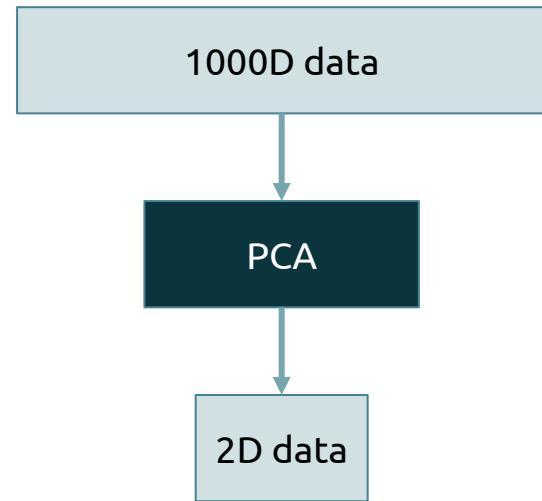


Principal component analysis (PCA)

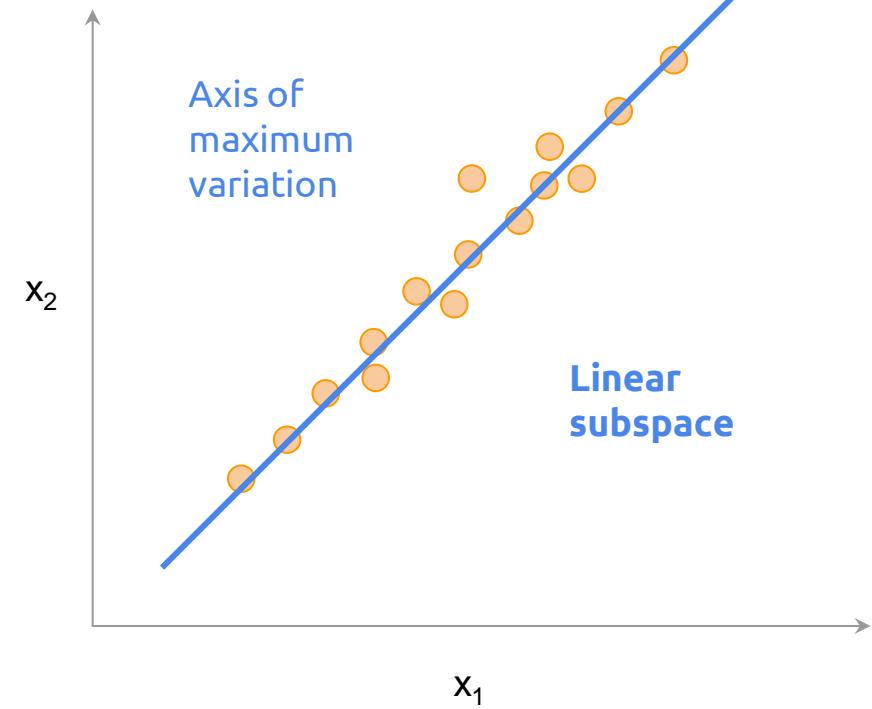
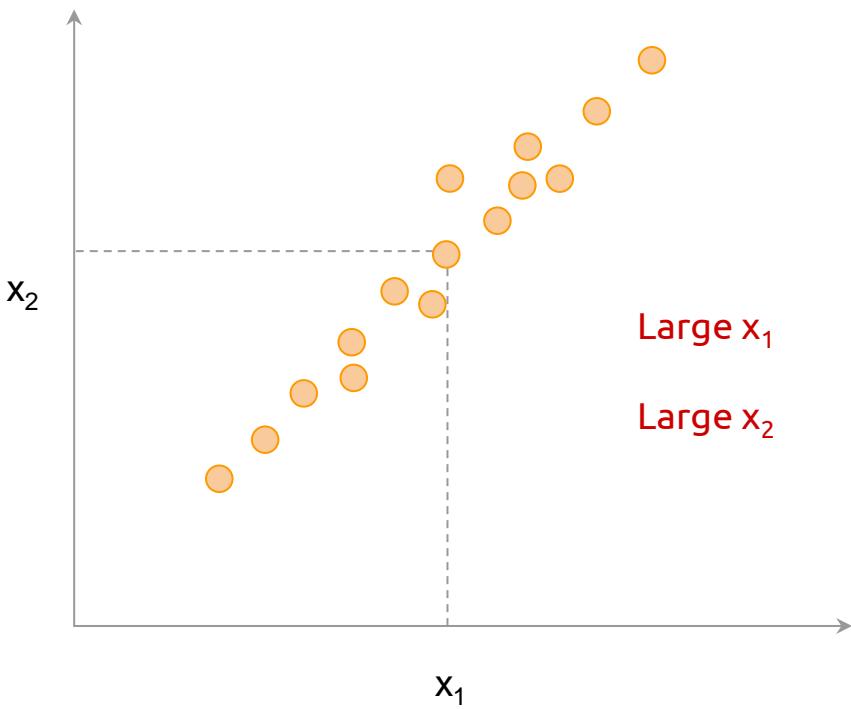
Reduce the dimensionality of the data by **projecting it onto a linear subspace** and **discarding axes of least variation**.

Very widely used:

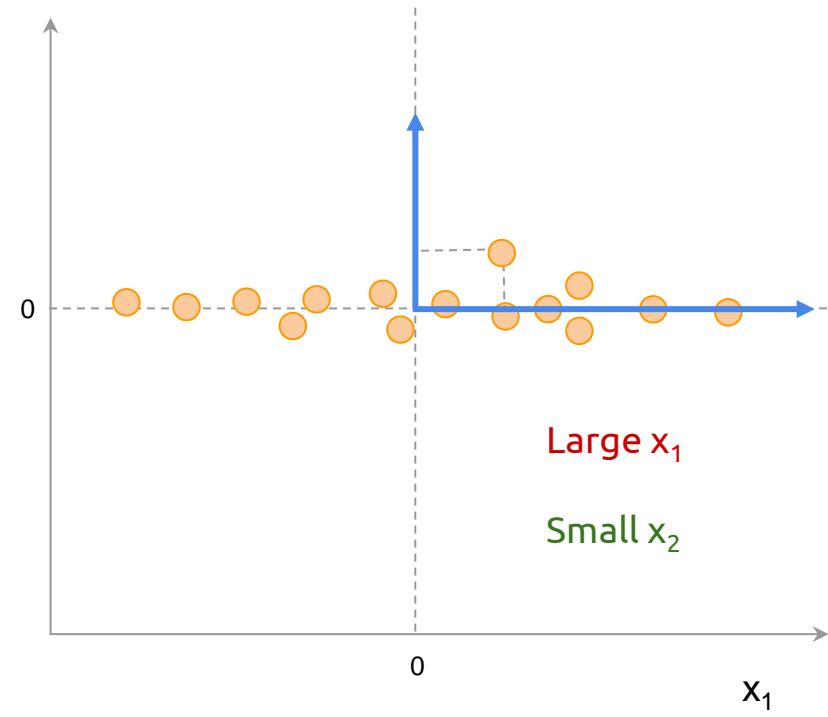
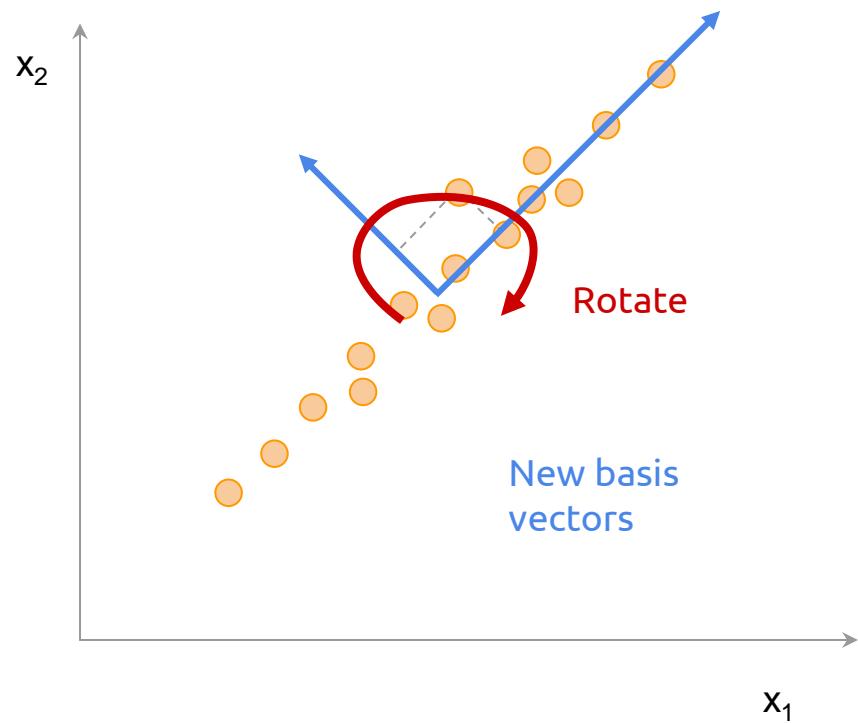
- 2D/3D visualization of high-dimensional data
- Compression
- Decorrelating data, **whitening**
- Eliminating noise
- Supervised learning



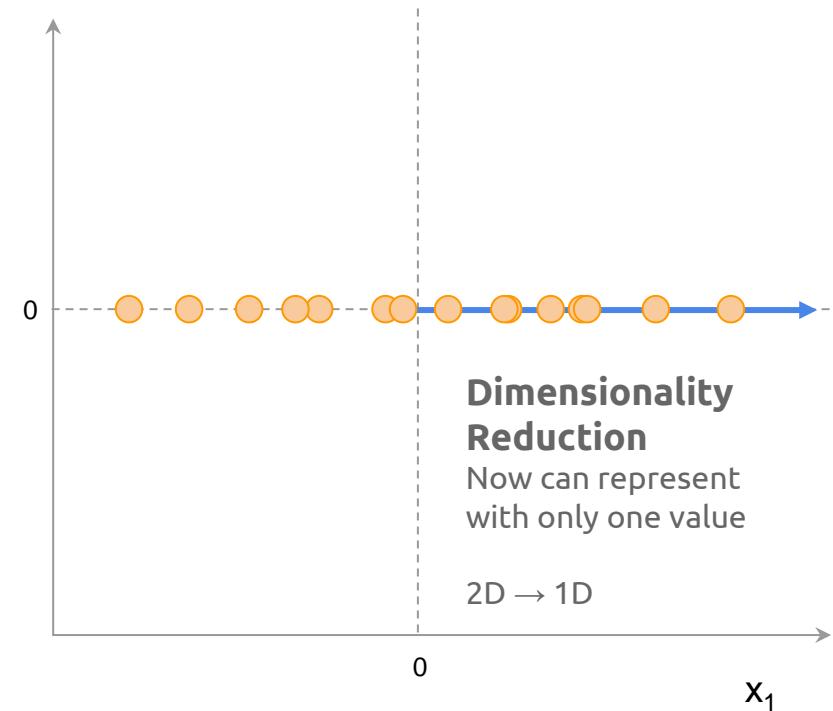
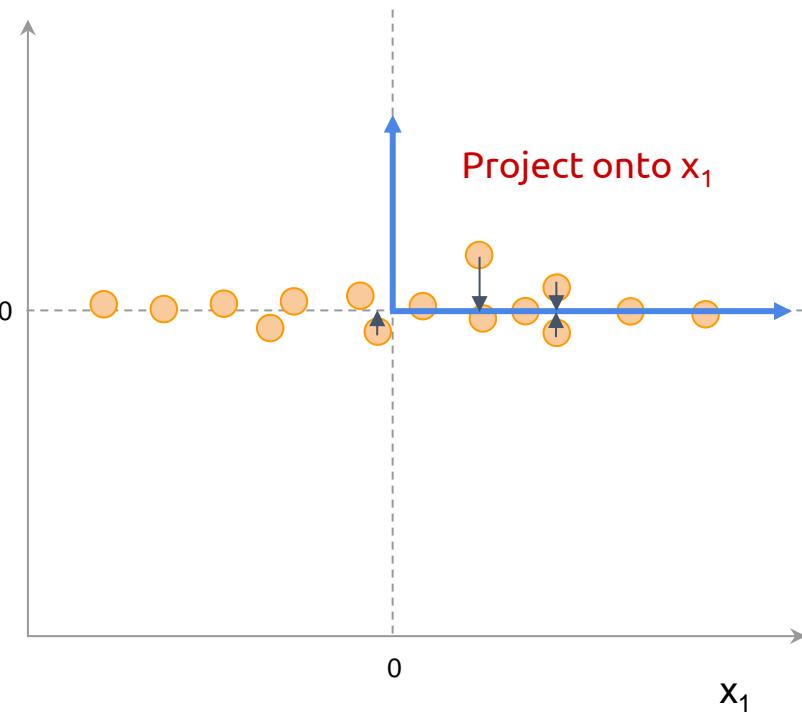
Intuition



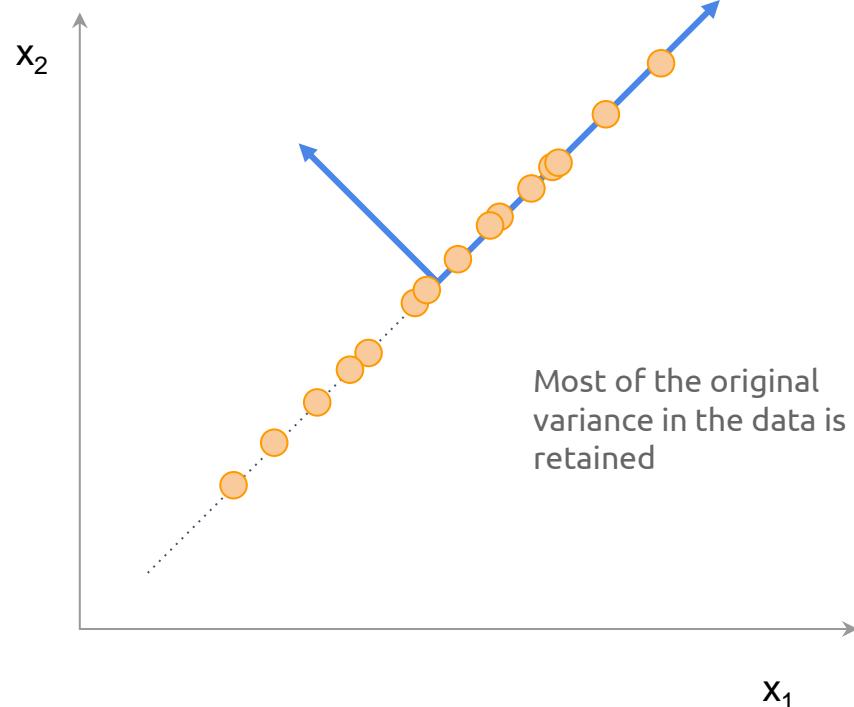
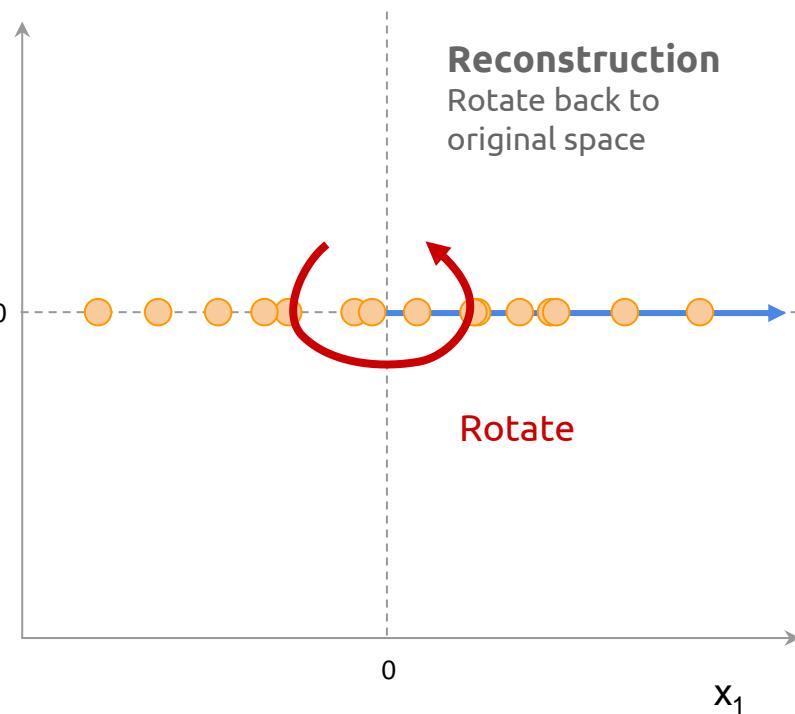
Intuition



Intuition



Intuition



PCA

PCA finds a **linear transform** (matrix) that **rotates** the data so that the directions of maximum variation in the data are aligned with the (x_1, x_2, \dots, x_n) axes.

The largest direction of variation is aligned with the x_1 axis, the second largest with the x_2 axis, and so on.

Truncating the dimension of the data after this rotation eliminates directions small variation and **reduces the dimensionality** of the data.

Reconstruction can be done by rotating the dimensionally reduced data back to its original coordinate system (basis).

Method

The PCA algorithm is based on **Eigenvalue decomposition** of the **empirical covariance matrix** of the data.

This matrix captures how the different dimensions of the data (e.g. x_1 and x_2) covary

PCA in code

```
class PCA(object):
    def __init__(self, n=1):
        self.n = n

    def fit(self, X):

        # compute mean of the data and store
        self.mean = X.mean(axis=0)

        # subtract the mean
        X = X - self.mean

        # compute the empirical covariance matrix
        covariance = np.dot(X.T, X)

        # compute eigenvalues and eigenvectors
        vals, vecs = np.linalg.eig(covariance)
```

```
# sort eigenvalues and vectors by eigenvalue
indices = np.argsort(-vals)
vals = vals[indices]
vecs = vecs[:, indices]

# store eigenvalues and principal components
self.eigenvalues = vals
self.components = vecs[:, :self.n]
return self

def transform(self, X):
    return np.dot(X - self.mean, self.components)

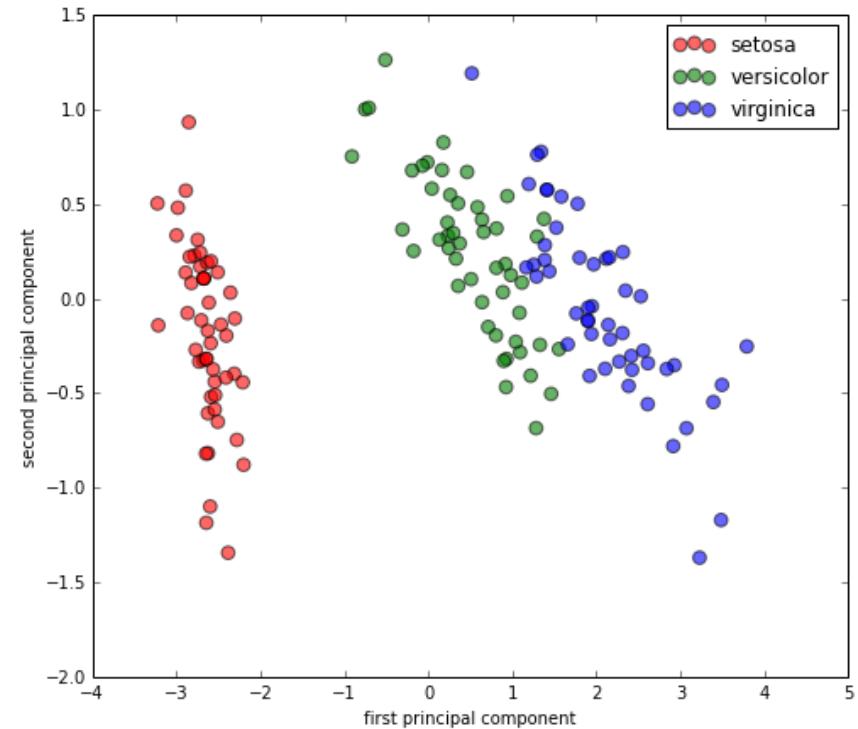
def inverse_transform(self, X):
    return np.dot(X, self.components.T) + self.mean
```

Example: visualization of 4D data

The famous Iris plants dataset (Fisher, 1950)

150 samples, 4 variables, 3 classes of plant

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	class
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa
5	5.4	3.9	1.7	0.4	setosa
6	4.6	3.4	1.4	0.3	setosa



Application of PCA

Eigenfaces (1987)

Recognition of identities from images of people's faces

Images are very high dimensional data!

- 100×100 image = 10,000D

We would like a low-dimensional representation that captures most of the variation: PCA

We can then compare different images (using distance or similarity metric) in this low-dimensional space

(Much better ways to do this nowadays)



Latent semantic analysis (LSA)

PCA-like algorithms applied to document analysis

Documents can be represented as **bags of words**. Histograms of word counts in the document. These are very high-dimensional representations (1 dimension per word in vocabulary).

Applying PCA to this representation gives a lower-dimensional representation for a document that retains the main directions of variation

Assumption: documents about similar topics have similar terms (covariance).

LSA is used to create a low-dimensional **latent** representation that captures the main topics.

Can be used with k-means or agglomerative clustering to cluster documents with similar topics.

How many principal components?

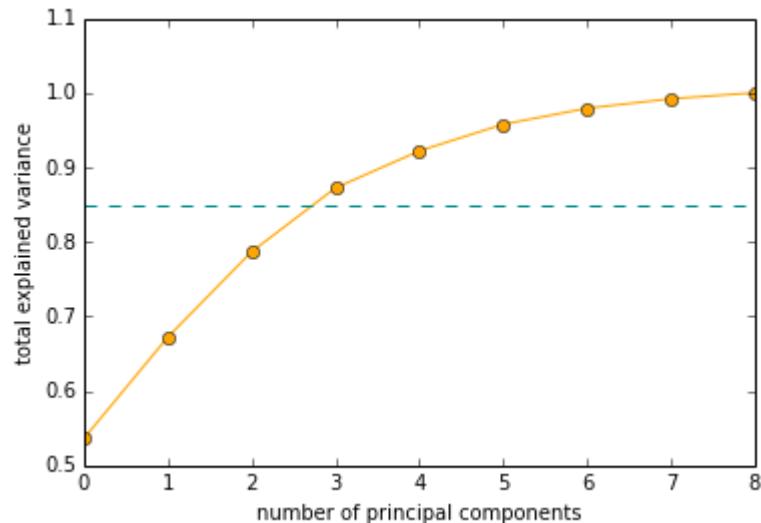
Depends on the task

- 2D visualization: 2 (obviously)

Typical heuristic:

- Take first k components that explain 85-90% of variance

More components: more variation explained, but possibly more noise



Overview

Motivation and goals of unsupervised learning

Clustering

- K-Means
- Agglomerative clustering

Dimensionality reduction

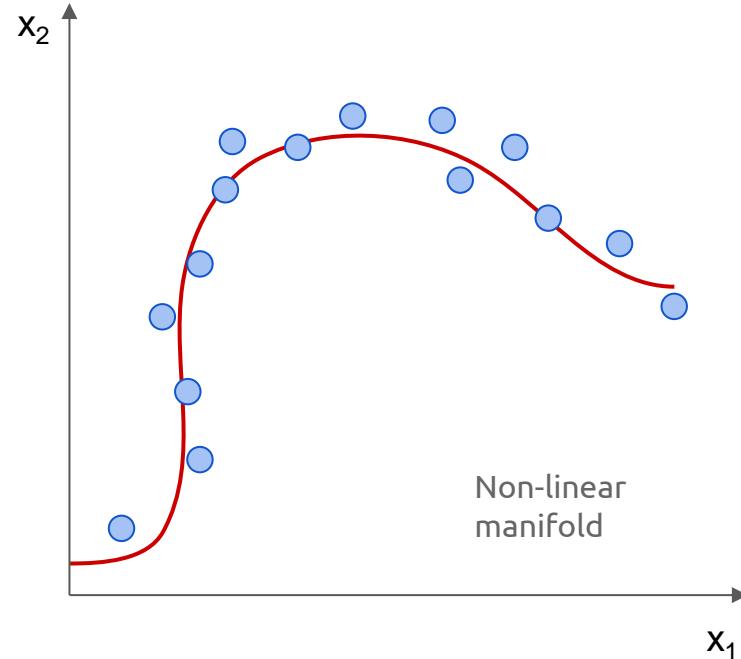
- PCA
- t-SNE
- Other methods

Nonlinear dimensionality reduction

PCA assumes that the data is near a linear manifold (hyperplane)

There are also techniques called **manifold learning** algorithms that can model nonlinear manifolds.

- Nonlinear autoencoders
- t-SNE
- Isomap
- Multidimensional scaling (MDS)



t-SNE

t-distributed stochastic neighbor embedding

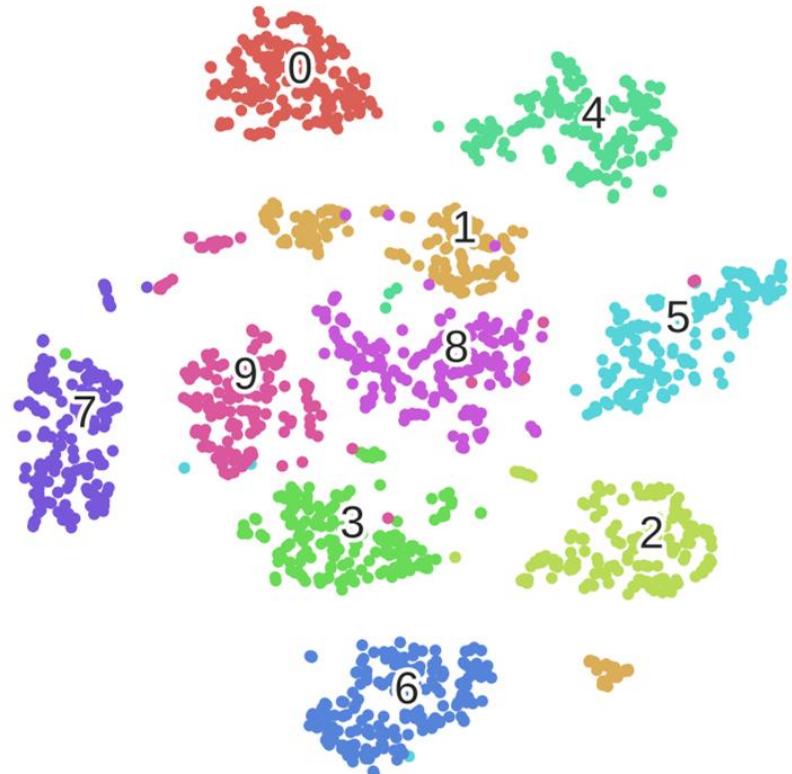
Mainly used as a data visualization technique.

Takes points in D dimensions and **embeds** them in a 2D or 3D space (map points).

Starts off with a random (**stochastic**) embedding and iteratively improves it.

Attempts to preserve distances between nearby **neighbors**.

Uses the **Student's t distribution** to measure the similarity between points



t-SNE



t-SNE algorithm

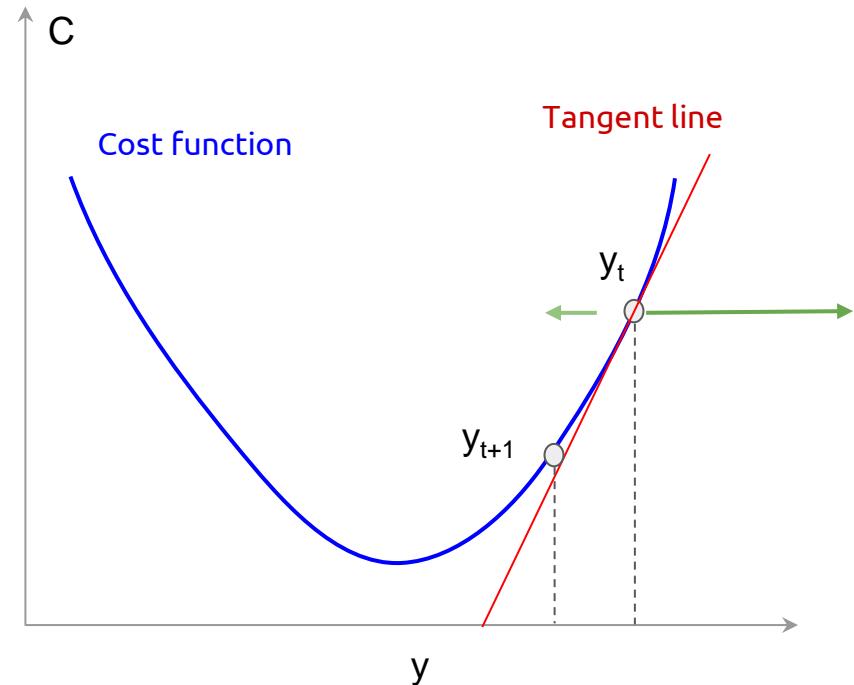
Method: **gradient descent**

Start with randomly chosen y_i

Iteratively improve y_i by computing gradient of cost function wrt. y_i and updating y by taking a step in the direction opposite to the gradient.

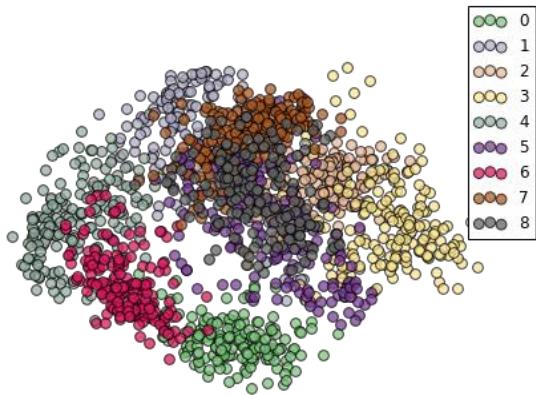
$$\frac{\partial C}{\partial y_i} = 4 \sum_j (p_{ij} - q_{ij})(y_i - y_j)(1 + \|y_i - y_j\|^2)^{-1}$$

$$\mathbf{y}_{t+1} = \mathbf{y}_t - \alpha \nabla_{\mathbf{y}} C$$

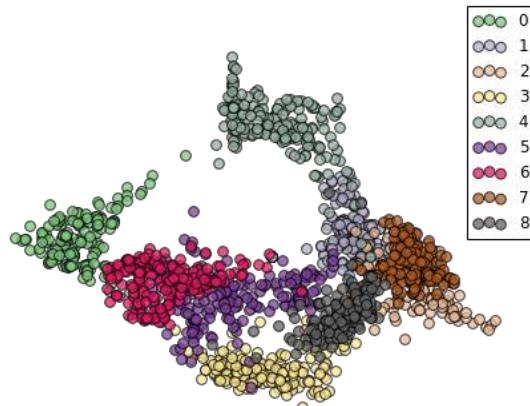


Visualization

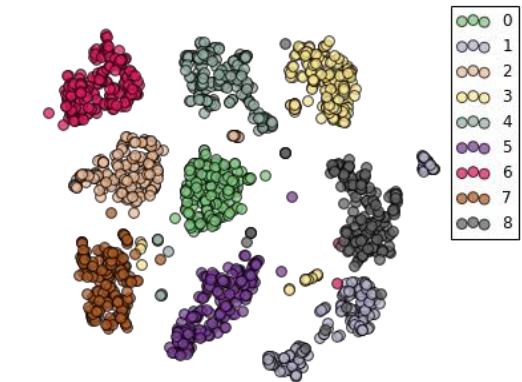
PCA



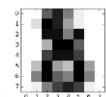
Isomap



t-SNE



Manifold learning for 8x8 (64D) images of handwritten digits (1800 samples)



t-SNE notes

Tends to work better than PCA for data visualization.

t-SNE is **stochastic**: you'll get a different result every time you run it (unless you seed the random number generator)

Only transforms given data: cannot be used to transform new unseen data (unlike K-means and PCA). Mostly used as a visualization tool.

Standard algorithm can be computationally expensive. Faster approximations like Barnes-Hut exist.

Overview

Motivation and goals of unsupervised learning

Clustering

- K-Means
- Agglomerative clustering

Dimensionality reduction

- PCA
- t-SNE
- Other methods

Other approaches

Multidimensional scaling (MDS)

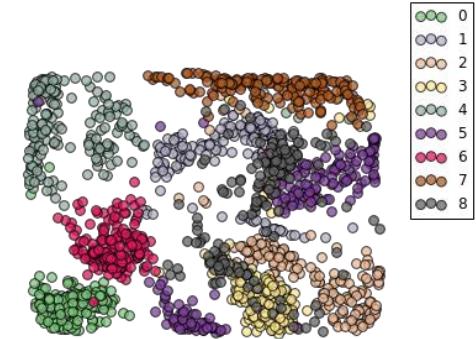
- Given **pairwise distance matrix**, find an embedding that preserves distances
- Based on Eigenvalue analysis of the **Gram matrix** (XX^T). Closely related to PCA

Isomap

- Extension of MDS: use neighbourhood graph and geodesic distances to create pairwise distance matrix.
- Solve with MDS

Autoencoders

- Minimize reconstruction error using gradient descent
- Linear autoencoders \sim PCA
- Lots of variants (sparse, denoising, deep)
- Unlike MDS/Isomap/TSNE, can be used to transform unseen data.
- Will see more about these in lecture on deep learning



Unsupervised learning with scikit-learn

Classes

- `sklearn.cluster.KMeans`
- `sklearn.cluster.AgglomerativeClustering`
- `sklearn.decomposition.PCA`
- `sklearn.manifold.TSNE`
- `sklearn.manifold.MDS`
- `sklearn.manifold.Isomap`
- `sklearn.mixture.GaussianMixture`

```
from sklearn.datasets import load_digits
from sklearn.manifold import TSNE

digits = load_digits()
X, y = digits.data, digits.target

tsne = TSNE()
X_tsne = tsne.fit_transform(X)
```

Methods

- `fit(X)`
- `fit_transform(X)`
- `transform(X)`
- `predict(X)`

Not all algorithms implement all these methods!

Further reading

The Elements of Statistical Learning, Chapter 14

- Introduction: 14.1
- Clustering: 14.3
 - K-Means: 14.3.6
 - Gap statistic: 14.3.11
 - Agglomerative clustering: 14.3.12
- PCA: 14.5.1

EEN1085/EEN1083

Data analysis and machine learning I

Ali Intizar



Ollscoil Chathair
Bhaile Átha Cliath
Dublin City University

Decisions Trees



Ollscoil Chathair
Bhaile Átha Cliath
Dublin City University

Outline

Decision Trees

Top-down greedy search

CART

Advantages and limitations of tree models

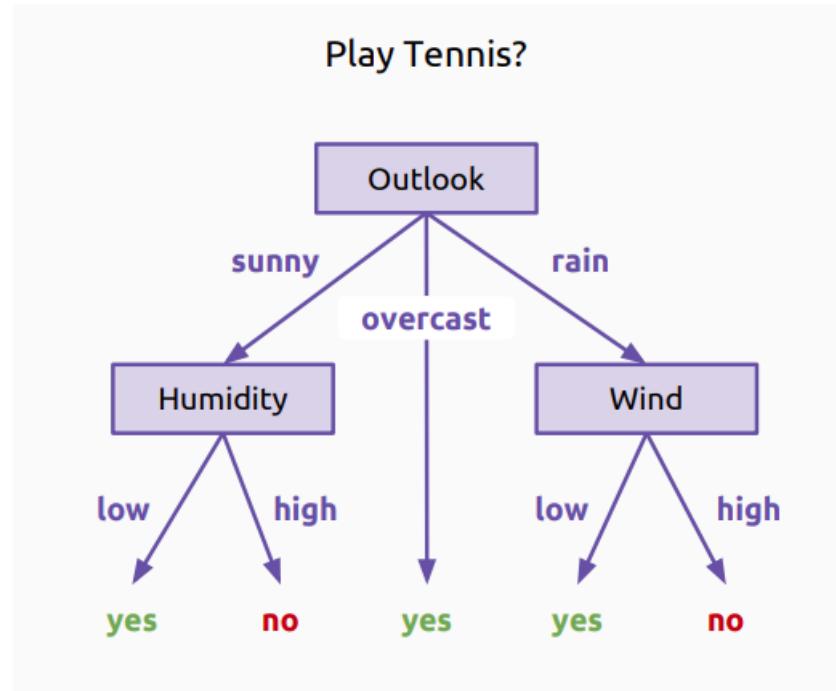
Decision Trees

Decision Trees

Hierarchical models for prediction.

Terminology:

- **Nodes** represent attributes of data (predictors, features, variates)
- **Branches** connect to other nodes based on value of attributes
- **Leaf** nodes assign outputs (e.g. classification)



Decision Trees

Classification trees: target variable is discrete and categorical. E.g.
 $y \in \{\text{red, green, blue}\}$

Regression trees: target variable is a real number. $y \in \mathbf{R}^D$

Fitting classification tree models

Given a training dataset, we would like an algorithm to produce a decision tree model that predicts a target variable from a set of attributes.

For now, let's simplify the problem a little:

- Assume ordinal attributes (e.g. $\mathbf{x} \in \mathbb{R}^D$)
- Assume binary classification $y \in \{0, 1\}$
- Consider only **binary** trees.

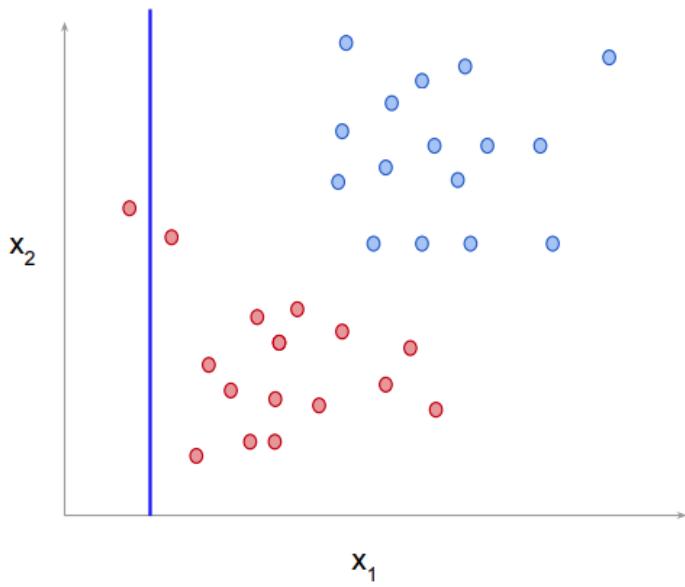
Idea: At each node in our tree we can split the data based on one of the attributes.
Build the tree by recursively splitting the data.

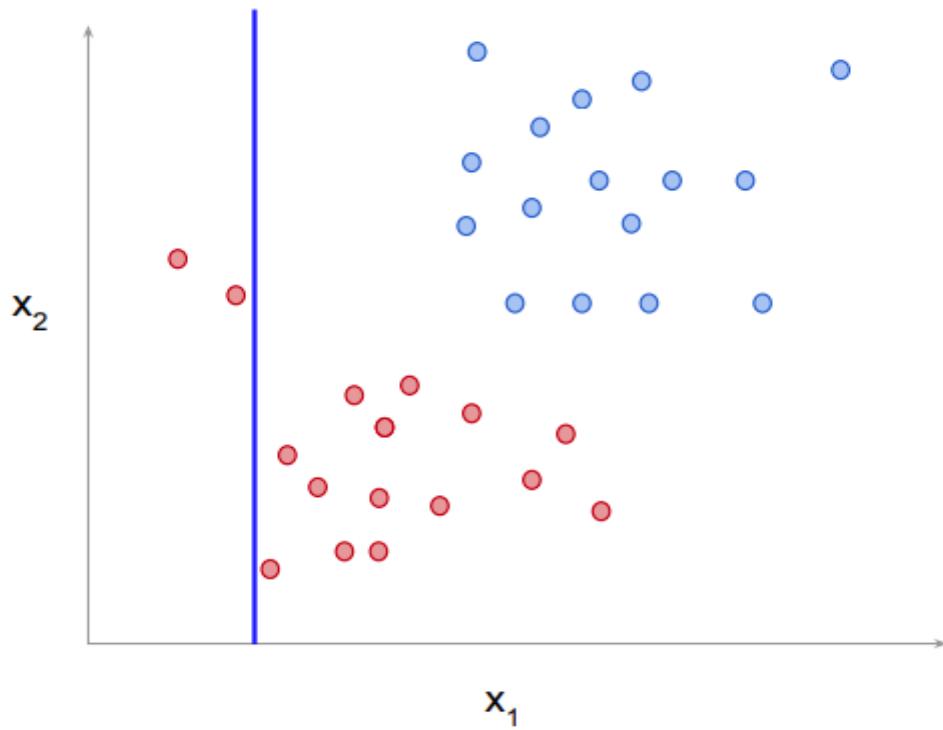
Top-down greedy search

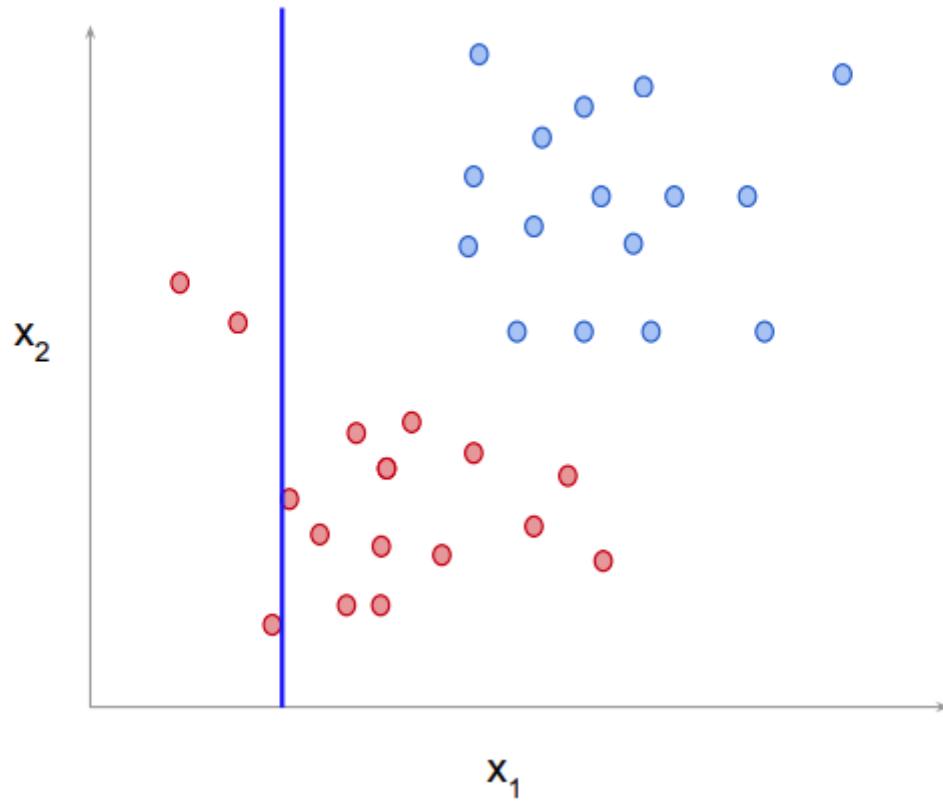
Starting with the root node, recursively:

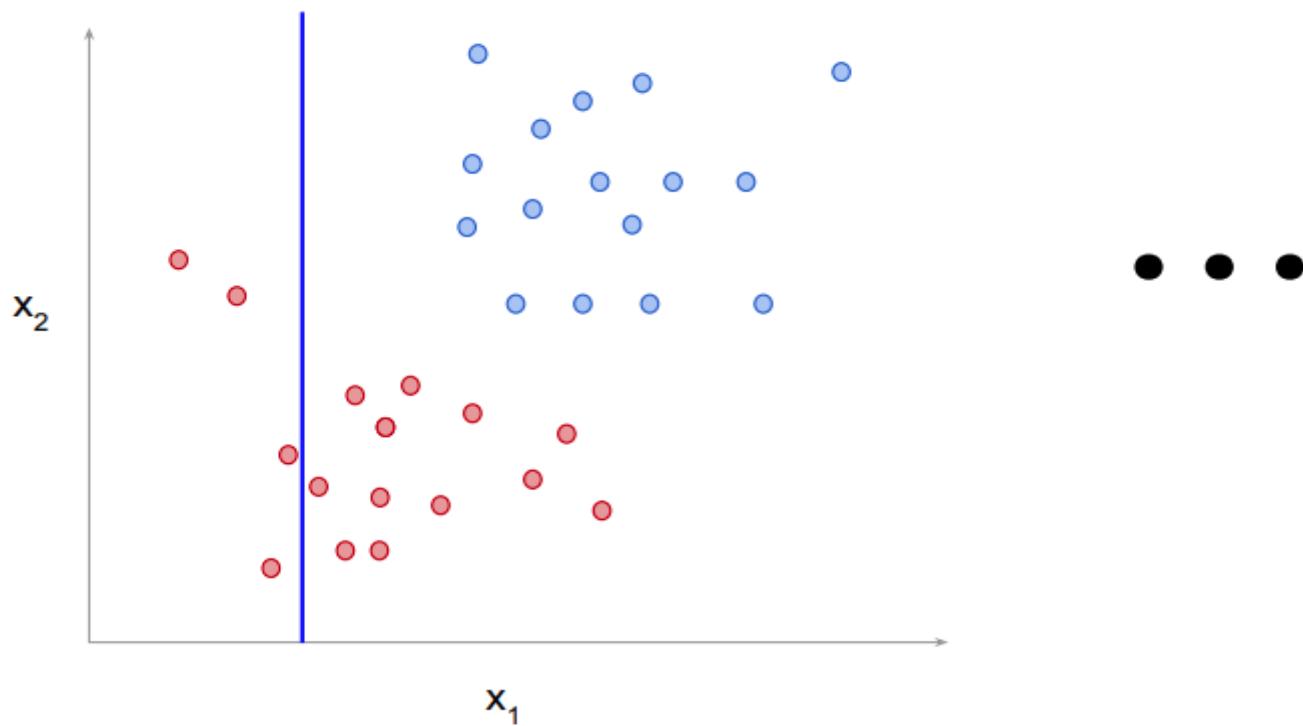
1. Find the attribute (dimension) and split point (threshold) that maximize some **measure of quality** (e.g. accuracy)
2. If the **stopping criteria** is not met:
 - Create a tree node to split on this attribute.
 - Perform the same procedure recursively for the left and right children.

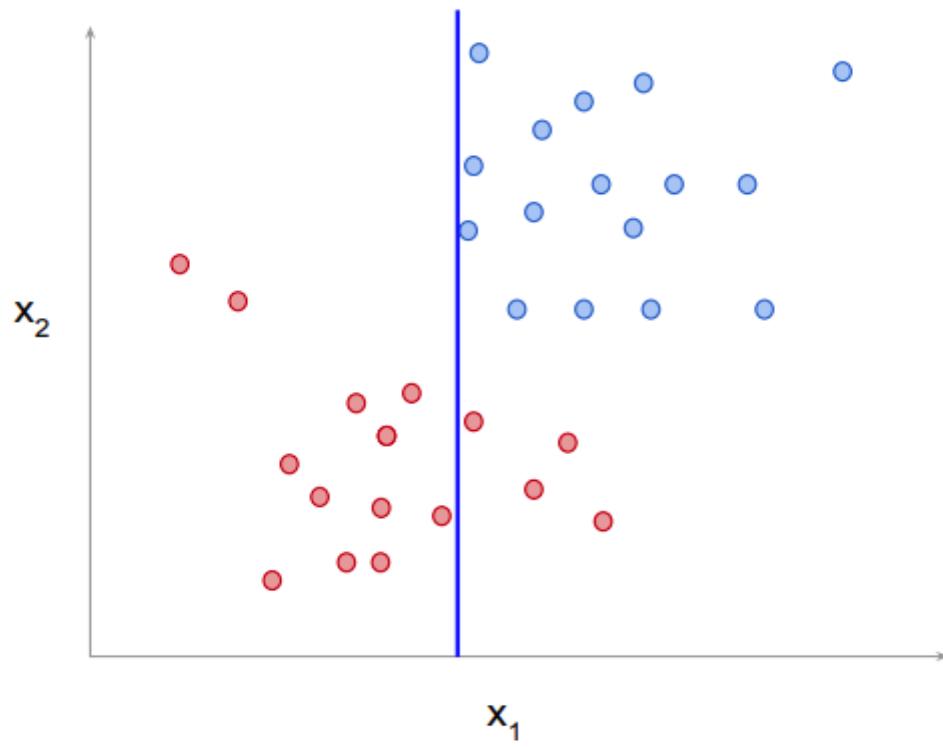
Example...

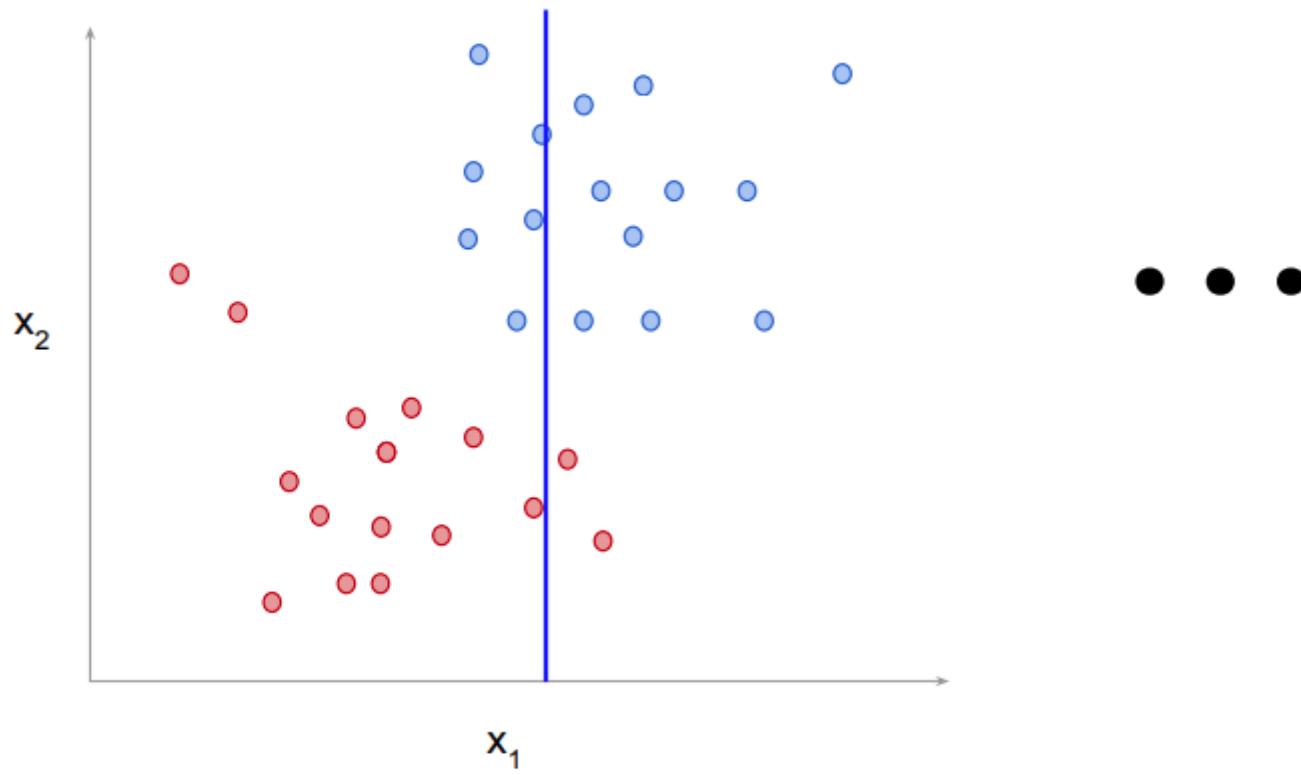


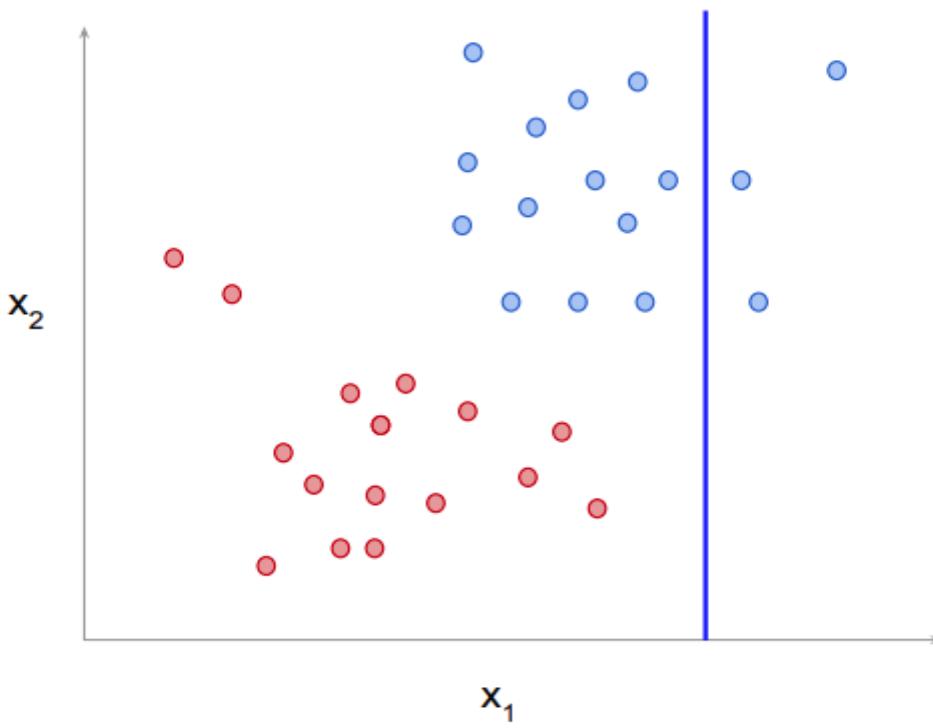


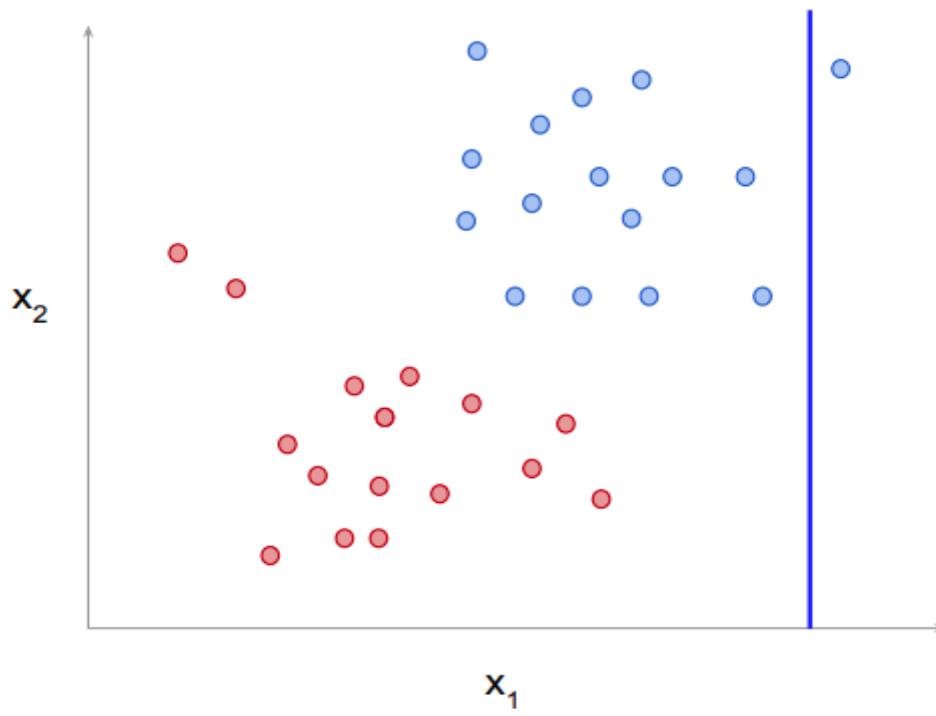


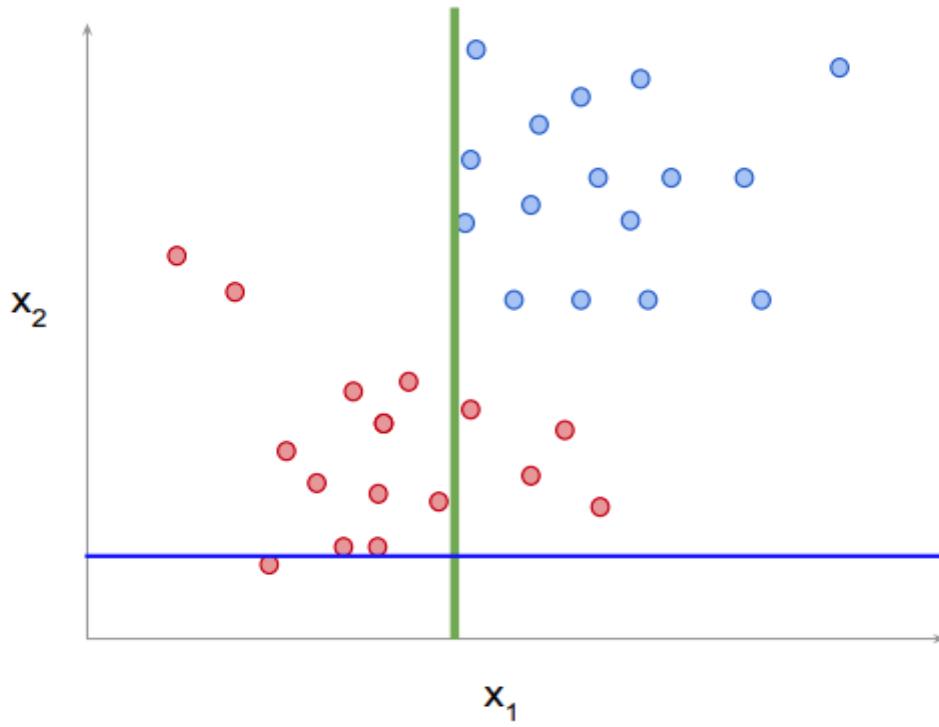


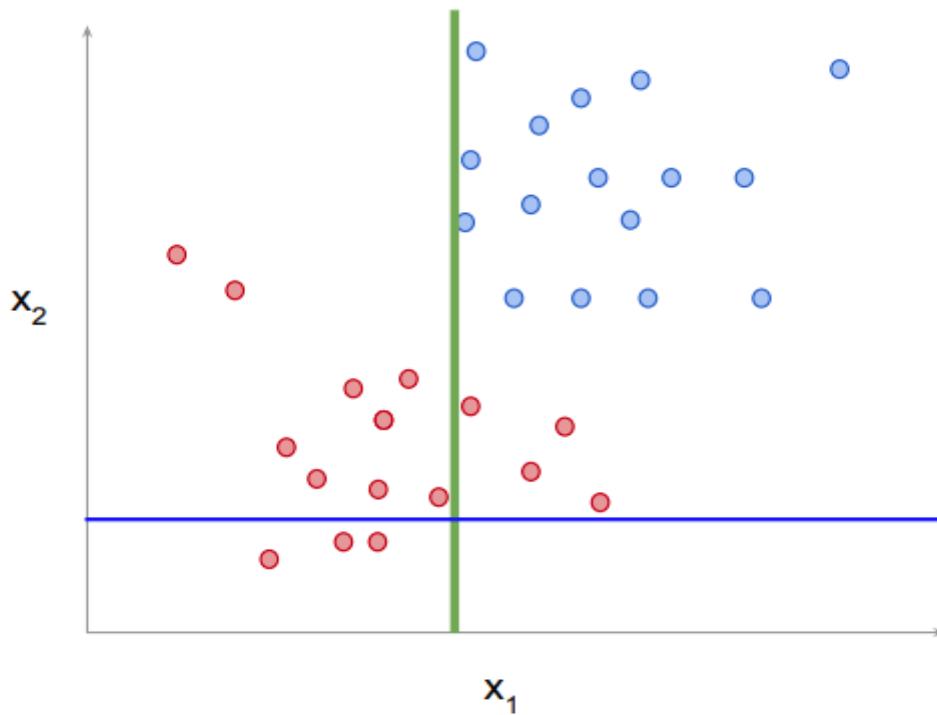


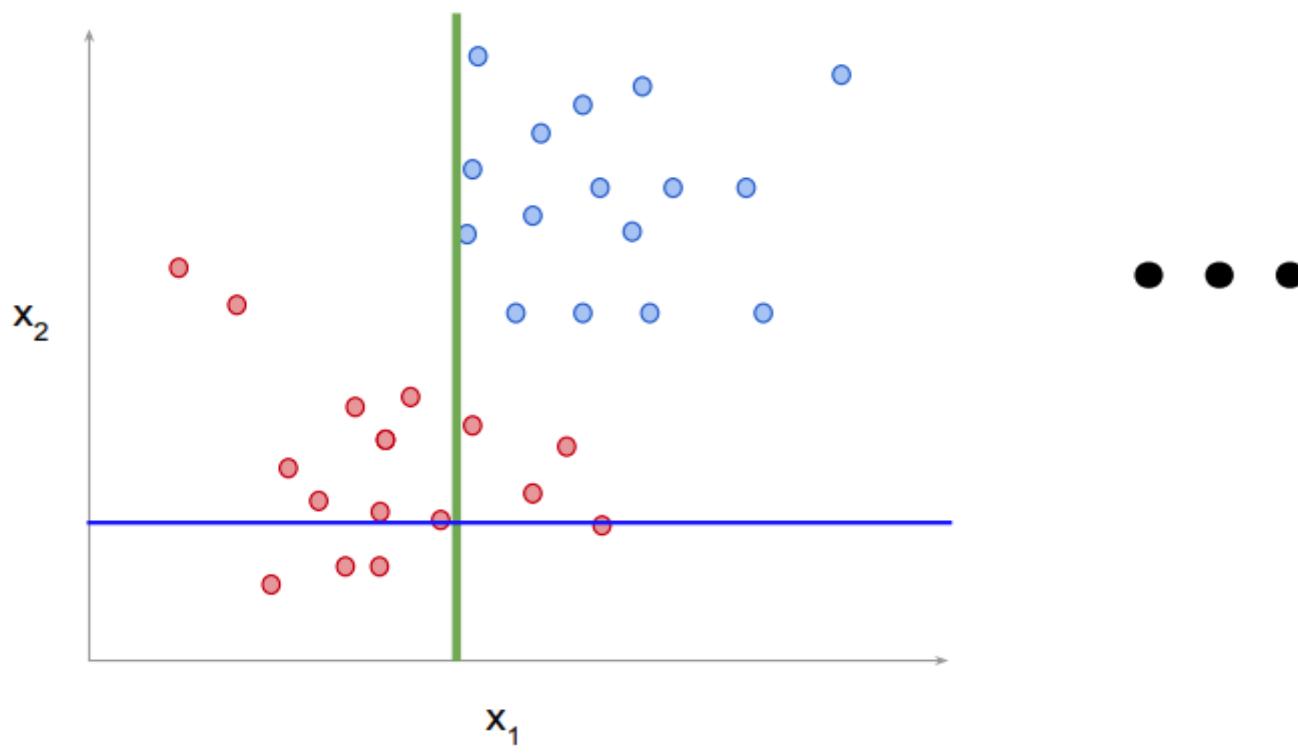


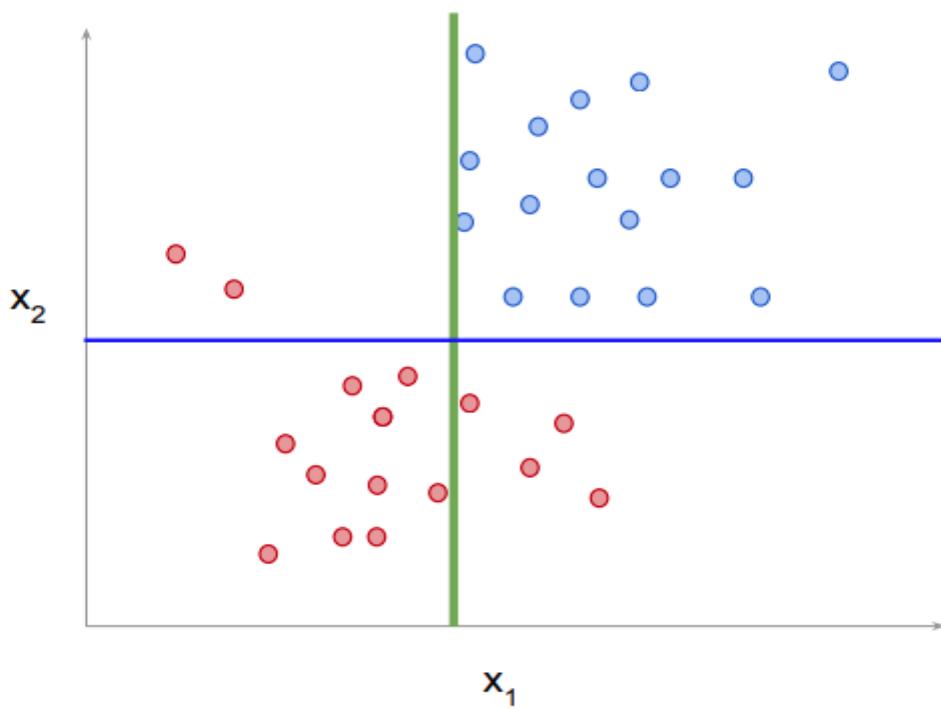


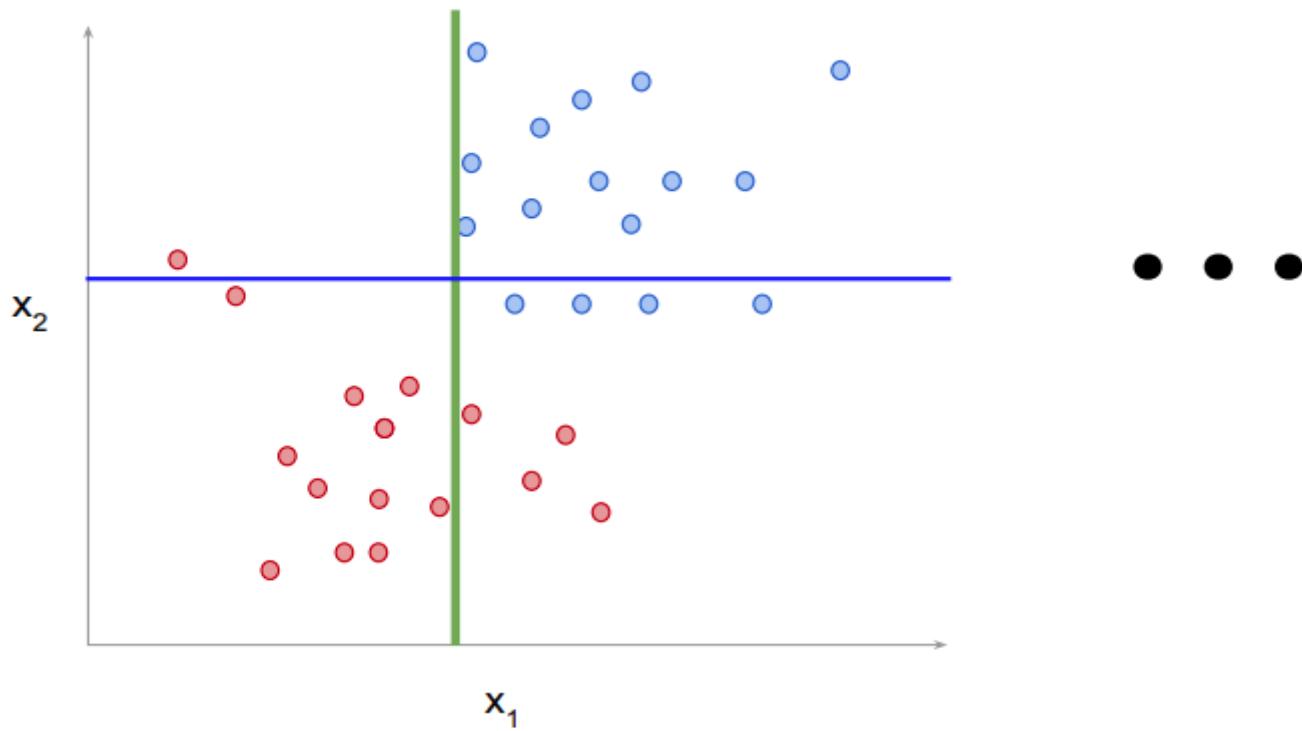


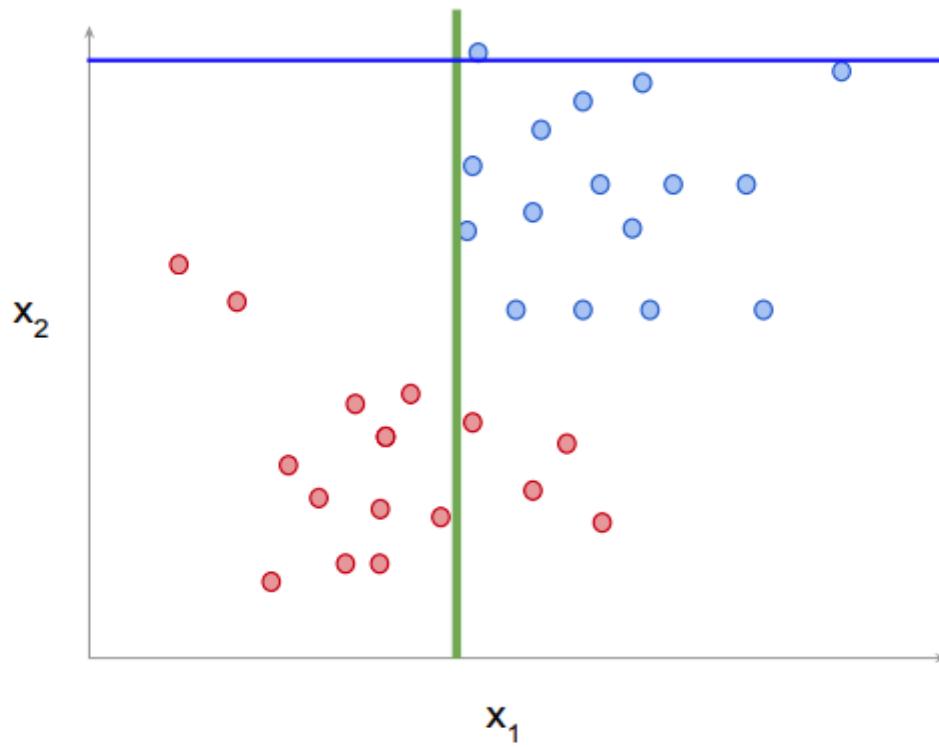


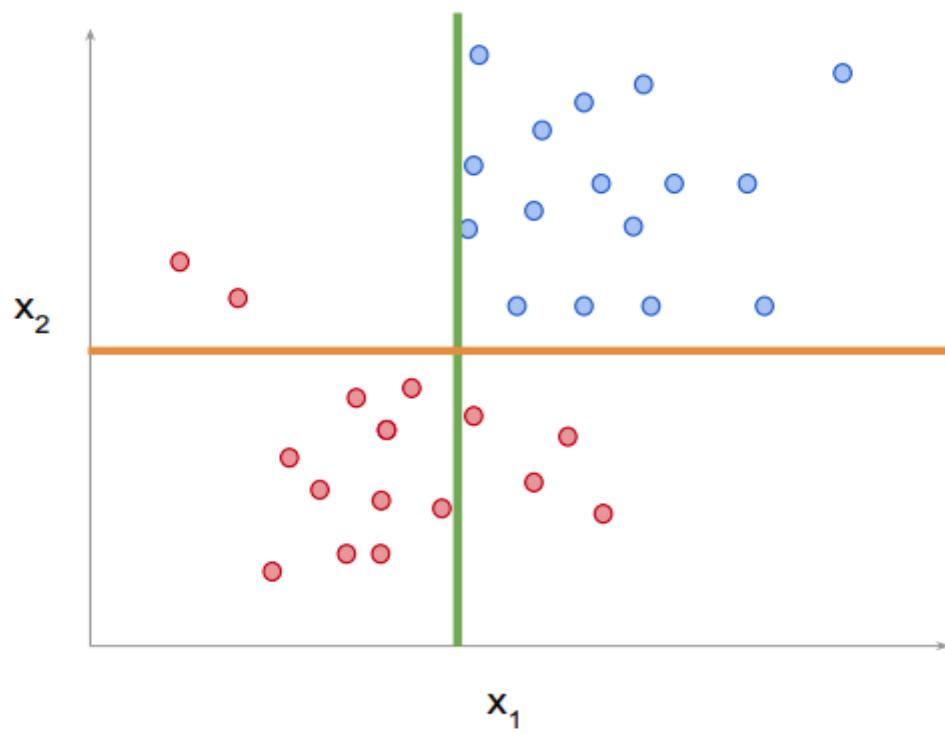


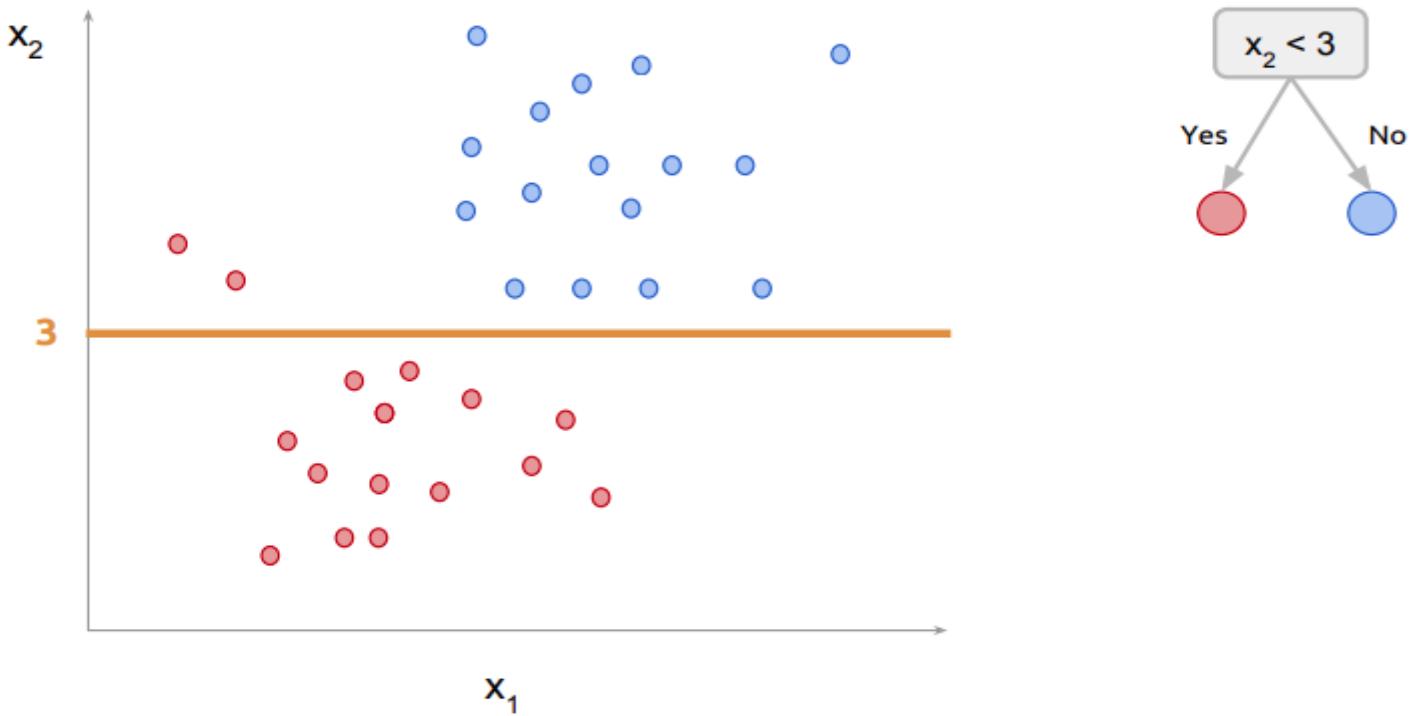


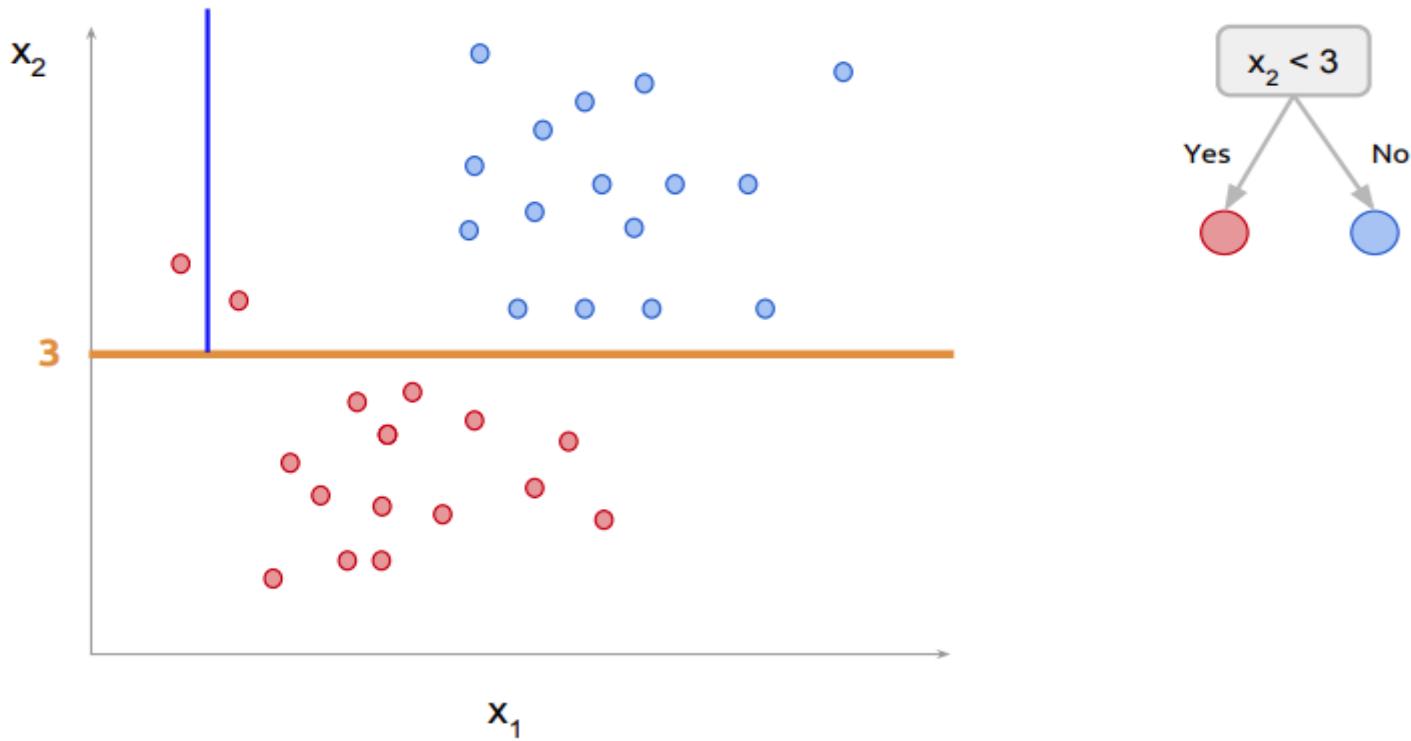


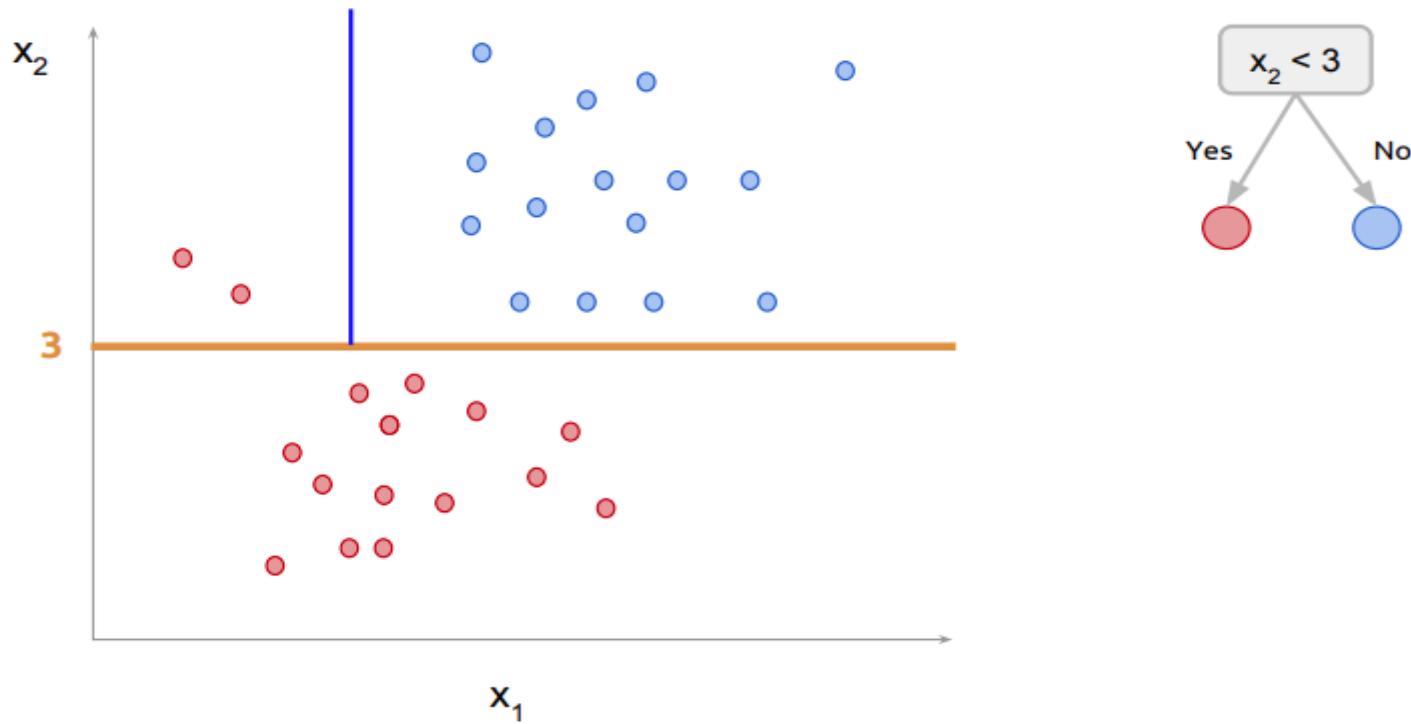


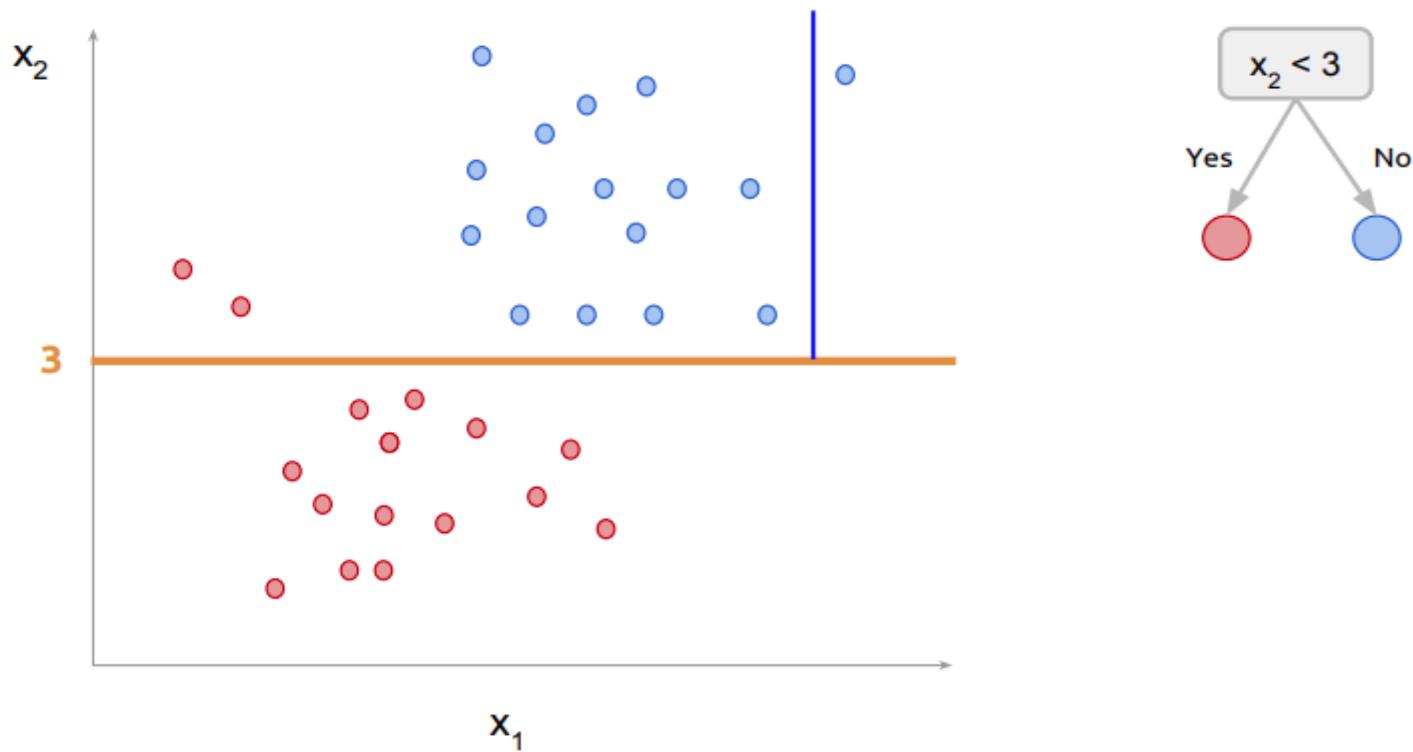






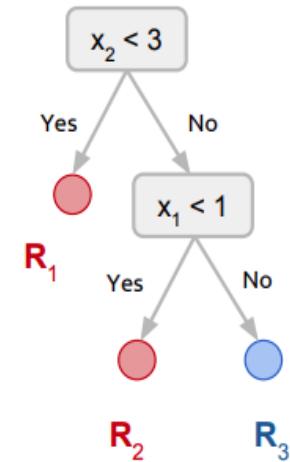
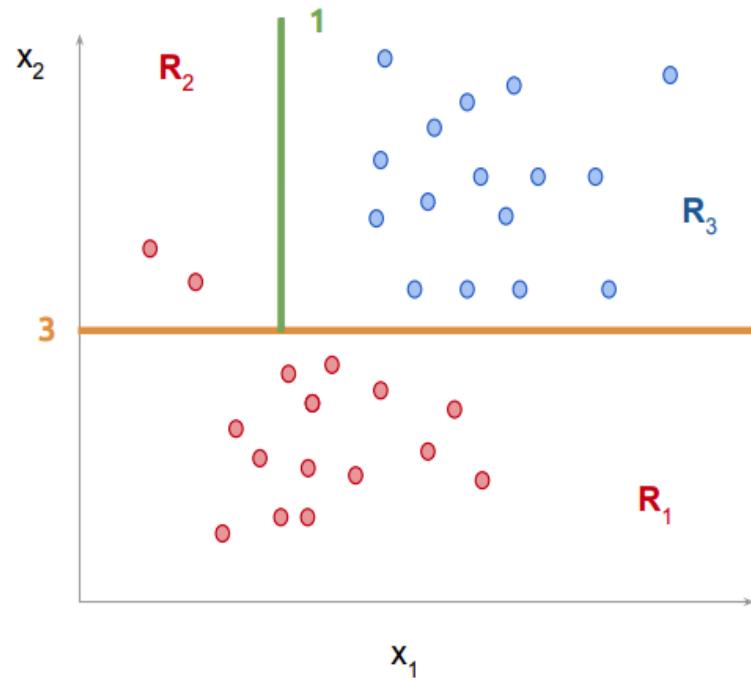




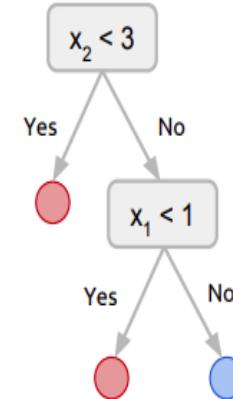
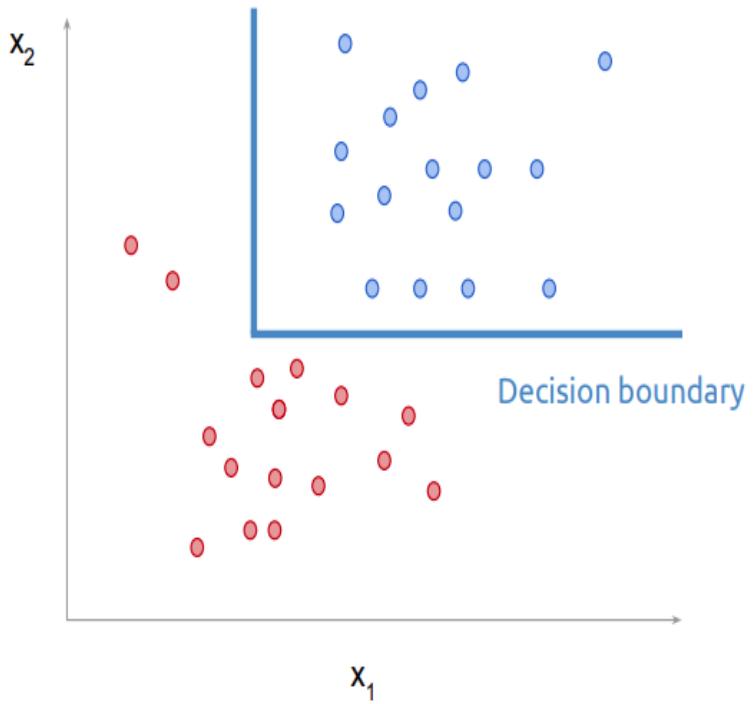


$$f(x) = \sum_{m=1}^M c_m [x \in R_m]$$

Most common category in
the region



$$f(x) = \sum_{m=1}^M c_m [x \in R_m]$$

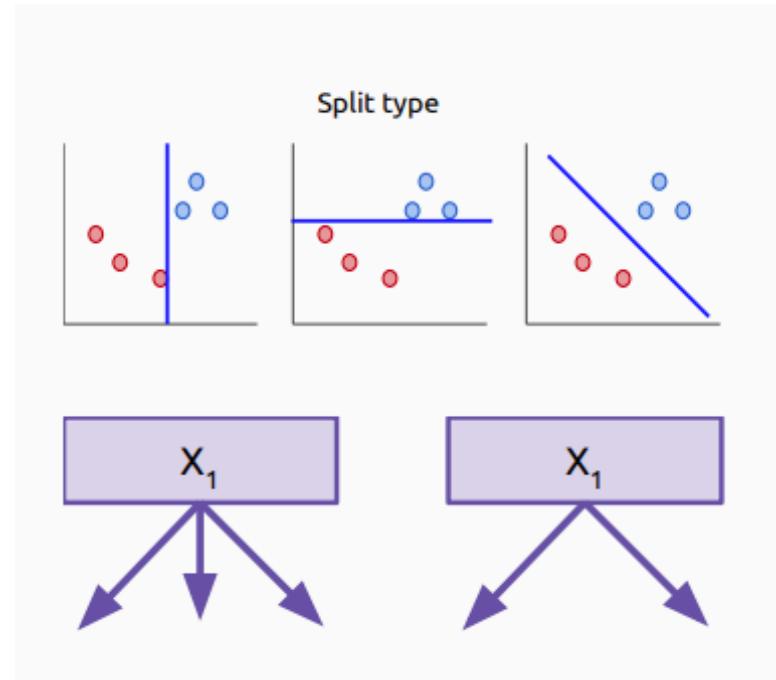


Decision tree algorithms

Many algorithms proposed for fitting (growing) decision trees

Variations:

- **Split type:** axes aligned (univariate) or linear
- **Branches per split:** binary or multinomial
- **Splitting criteria:** e.g. misclassification error, entropy, Gini index
- **Stopping criteria**
- Handling missing values



CART

- Classification and regression trees
- **Classification trees:** Predict a categorical output.
- Prediction for a region is the category that occurs most frequently in the region.
- **Regression trees:** Predict a real valued output.
- Instead of voting inside a region for the correct category, just average all y values in the region.

CART

- CART generates a **binary tree**.
- Top-down greedy splits nodes to minimize **impurity**:
 - Classification: Impurity of a node measured by impurity metric like the Gini index.
 - Regression: Impurity of a node measured by mean squared error.
- **Stopping criteria:** grow tree out to maximum size and prune it back. Use cross validation to find the optimal tree.

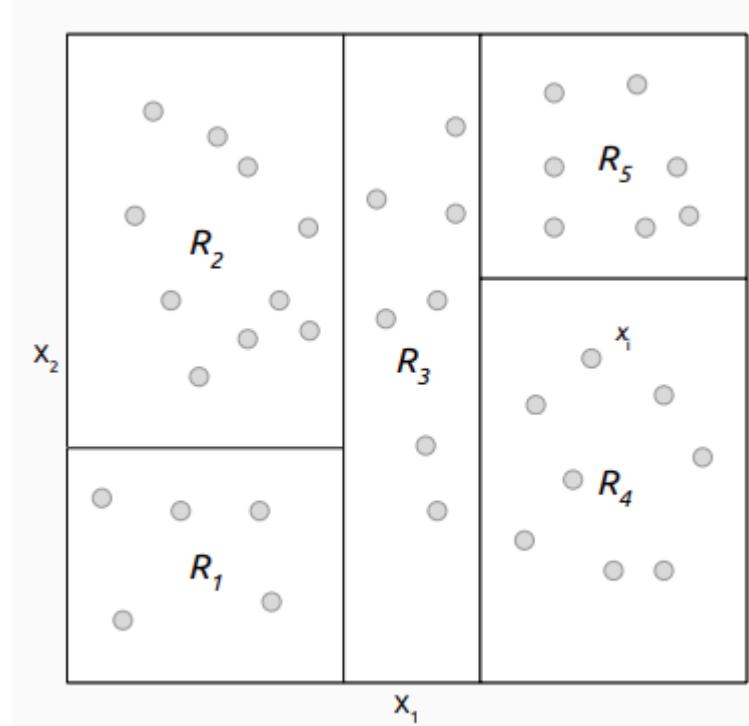
CART for regression

Notation:

- Training data: $\{(\mathbf{x}_i, y_i) : \mathbf{x}_i \in \mathbb{R}^D, y_i \in \mathbb{R}\}_{i=1}^N$
- Features: X_j
- Regions: $\{R_1, \dots, R_M\}$
- Region size: $N_m = |R_m|$

Prediction for region R_m is the average y value in the region:

$$c_m = \frac{1}{N_m} \sum_{i=1}^N y_i \mathbb{1}(\mathbf{x}_i \in R_m)$$



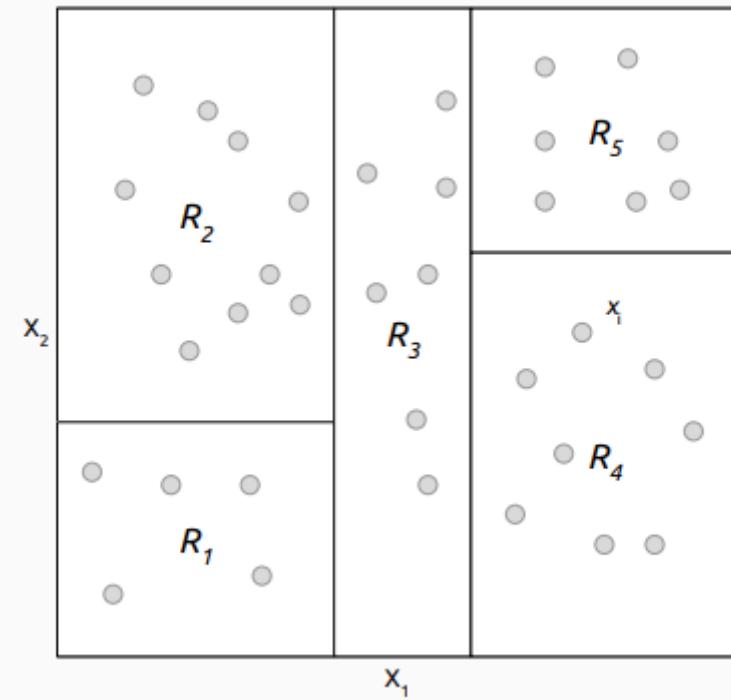
CART for regression

The decision function is:

$$f(\mathbf{x}) = \sum_{m=1}^M c_m \mathbb{1}(\mathbf{x} \in R_m)$$

Define the **impurity** of region R_m as:

$$E_m = \sum_{x_i \in R_m} (y_i - c_m)^2$$



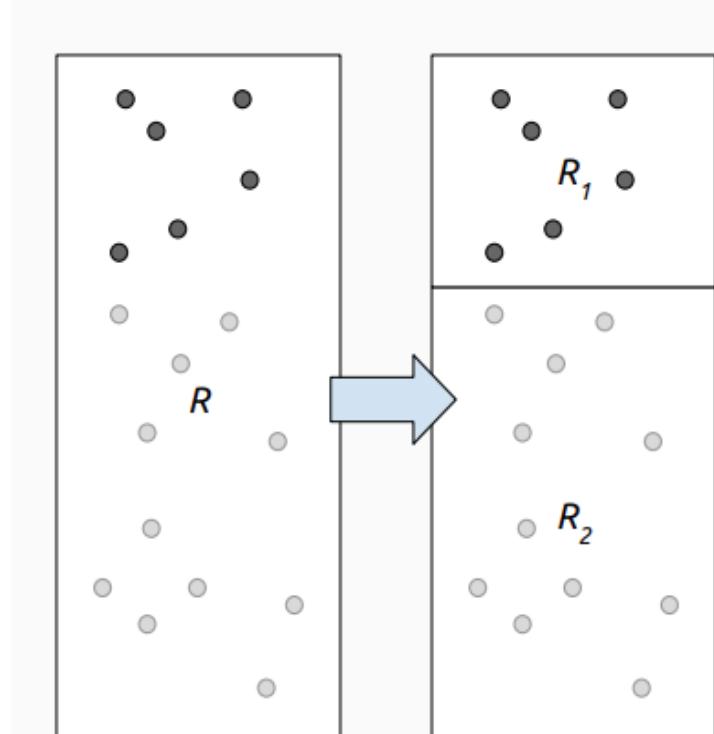
CART for regression

Greedy top-down algorithm.

Starting with a single region, recursively split region R into two subregions R_1 and R_2 .

Split: Search all features X_j and split points s to minimize the impurity of the resulting regions:

$$\begin{aligned} L(j, s) &= E_1 + E_2 \\ &= \sum_{x_i \in R_1} (y_i - c_1)^2 + \sum_{x_i \in R_2} (y_i - c_2)^2 \end{aligned}$$



CART for regression

Stopping criteria

Several strategies can be used:

1. Only split a node if the decrease in impurity exceeds a threshold.
2. Stop when each region contains only one point (grow tree to maximum depth).
3. Stop when each region contains fewer than K points.
4. Stop when the tree reaches a certain maximum depth.
5. Grow tree out to maximum depth and prune it back.

(1) is too shortsighted: poor split now could lead to great ones later. (2) can lead to overfitting. (3-4) are commonly used. (5) requires a **pruning algorithm**.

CART for regression

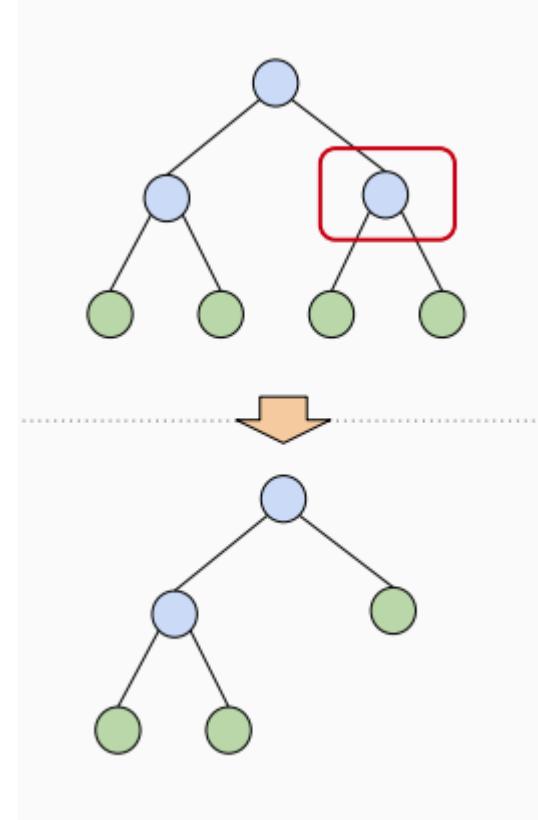
Cost complexity pruning

Idea: try to simultaneously minimize impurity (cost) and tree complexity.

Let T be the set of terminal (leaf) nodes in the tree. The *cost complexity criteria* trades off impurity for tree size $|T|$:

$$C_\alpha(T) = \sum_{m=1}^{|T|} E_m + \alpha |T|$$

Can find subtree T_α that minimizes cost complexity criteria for a given α by **weakest link pruning**.

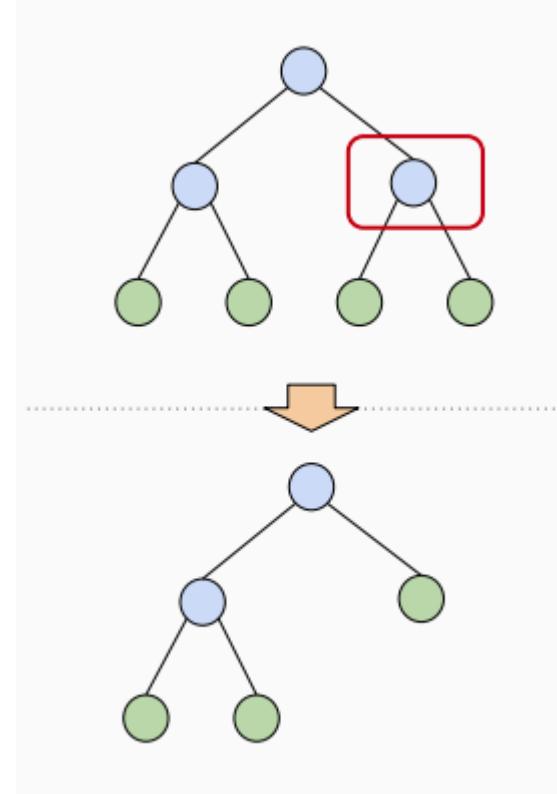


CART for regression

Weakest link pruning

1. Successively collapse internal nodes that produces the smallest increase in impurity until only the root node remains.
2. Compute C_α on each successive subtree and choose one with minimum value.

Need to choose hyperparameter α . Can be done using a validation set or by *cross validation*.

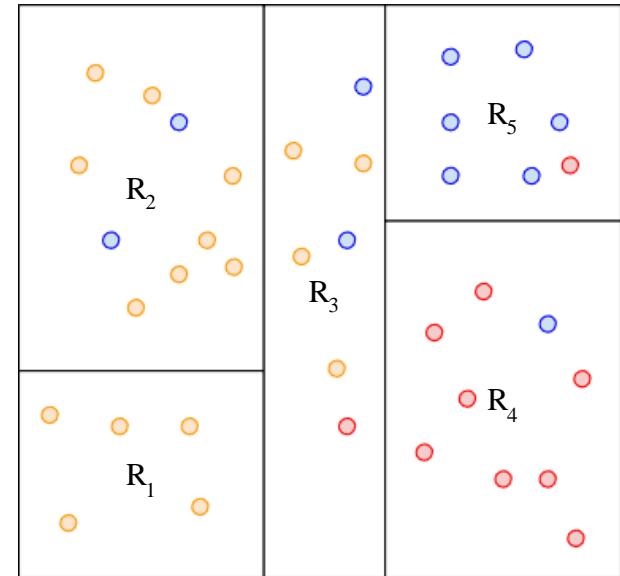


CART for classification

We can use almost the same algorithm for classification as we used for regression.

Two changes:

- Prediction function (c_m):** the prediction for a region is done by voting instead of averaging.
- Impurity measure (E_m):** Mean squared error is not appropriate for categorical variables.



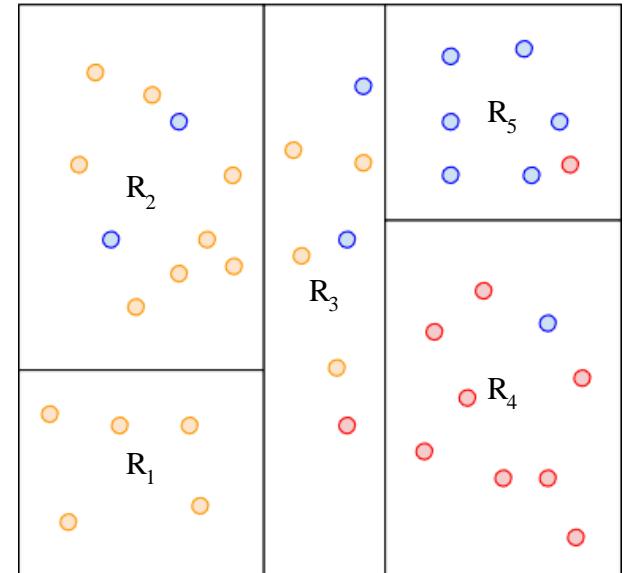
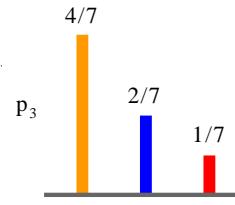
CART for classification

Region composition

Let \hat{p}_{mk} be the proportion of y values in region R_m that take the value k :

$$\hat{p}_{mk} = \frac{1}{N_m} \sum_{x_i \in R_m} \mathbb{1}(y_i = k)$$

I.e. \hat{p}_{mk} is a normalized **histogram** of the values of y in region R_m . E.g. region R_3 , \hat{p}_{3k} looks like:



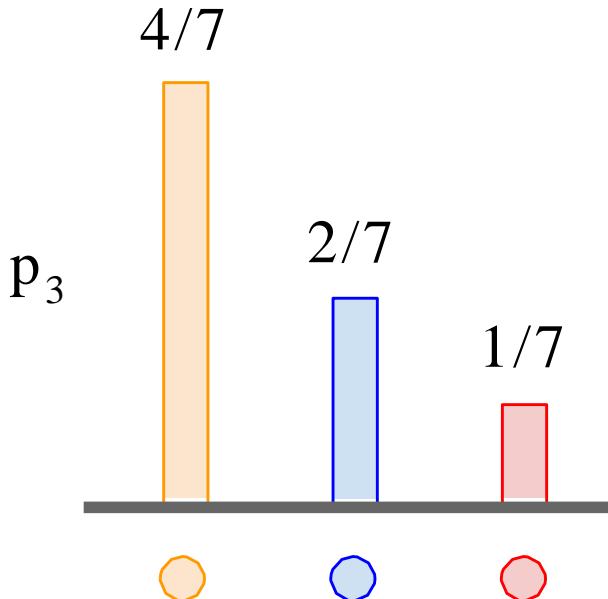
CART for classification

Prediction for region: y value with the most votes.

$$c_m = \max_k \hat{p}_{mk}$$

Region impurity Three commonly used metrics:

1. Misclassification error.
2. Entropy.
3. Gini index.



CART for classification

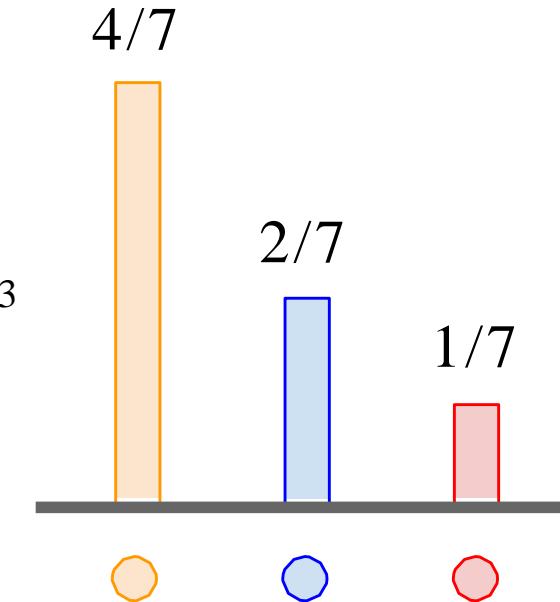
Misclassification error: number of misclassified elements in region R_m :

$$E_m = 1.0 - c_m$$

Entropy: information theoretical measure of *disorder* of a region.

$$H_m = - \sum_{k=1}^K \hat{p}_{mk} \log_2 \hat{p}_{mk}$$

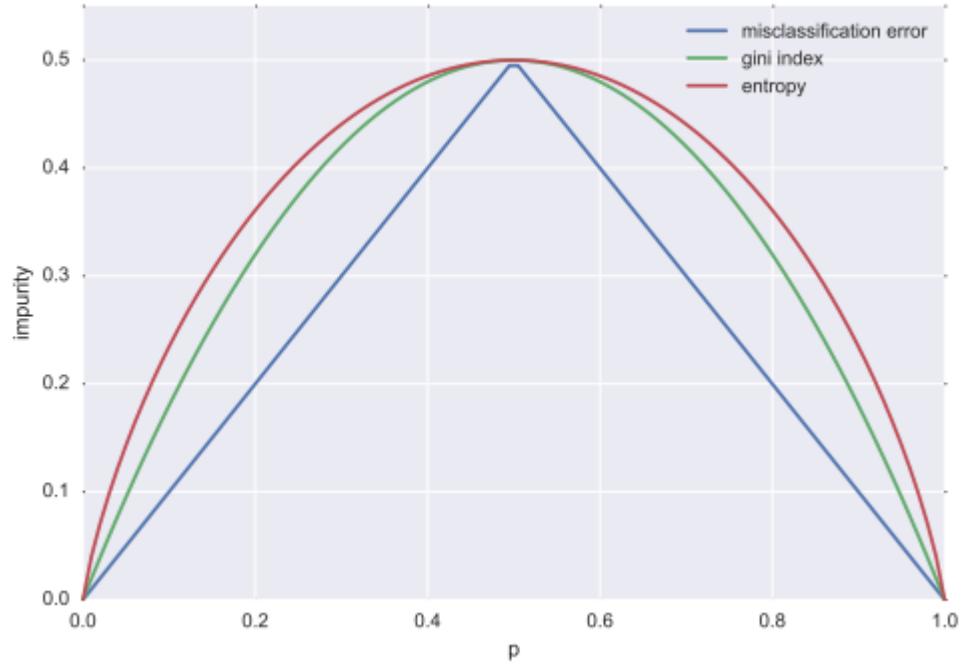
How many bits you would need to encode random draws from the region. Pure region has entropy 0.



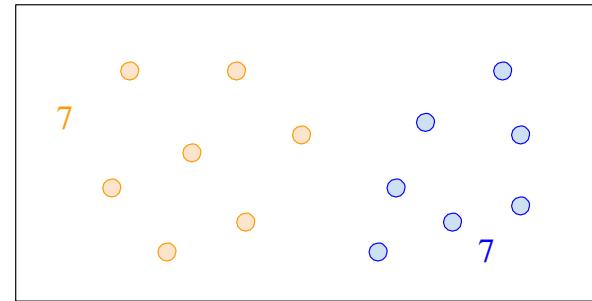
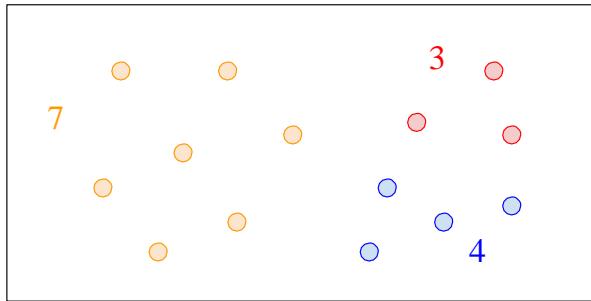
CART for classification

Gini index: most commonly used metric. Approximates entropy but is faster to compute and more numerically stable.

$$G_m = \sum_{k=1}^K \hat{p}_{mk}(1 - \hat{p}_{mk})$$



Why use Entropy or Gini over misclassification rate?



$$E_m = 0.5$$

$$H_m = 1.49$$

$$G_m = 0.62$$

$$E_m = 0.5$$

$$H_m = 1.0$$

$$G_m = 0.5$$

Other issues

Categorical variables: encode categorical variables with K possible values using a one-hot encoding $\rightarrow K$ binary variables.

Why binary splits? Multiway splits fragment the data too quickly, leaving less data next level down. Series of binary splits can achieve same as multiway ones.

Missing values: Can use usual techniques like imputing. Can also only introduce new categorical variables for when a value is missing.

Linear combination splits: Instead of using axis aligned splits, we could use linear predictors $\mathbf{w}^T \mathbf{x} \leq s$. Empirically, this doesn't work much better and it is more difficult to optimize.

Summary of CART

	Regression	Classification
Prediction for region R_m	$c_m = \frac{1}{N_m} \sum_{i=1}^N y_i \mathbb{1}(\mathbf{x}_i \in R_m)$	$c_m = \max_k \hat{p}_{mk}$
Decision function	$f(\mathbf{x}) = \sum_{m=1}^M c_m \mathbb{1}(\mathbf{x} \in R_m)$	$f(\mathbf{x}) = \sum_{m=1}^M c_m \mathbb{1}(\mathbf{x} \in R_m)$
Impurity measures	SE: $E_m = \sum_{x_i \in R_m} (y_i - c_m)^2$	Gini: $G_m = \sum_{k=1}^K \hat{p}_{mk}(1 - \hat{p}_{mk})$
Search		Greedy top down
Stopping		Depth threshold, region size threshold, pruning

Advantages of decision trees

Fast to train: Splitting the root node requires searching through D dimensions and N possible values. However, if each split roughly divides data in half, then N reduces logarithmically with successive splits.

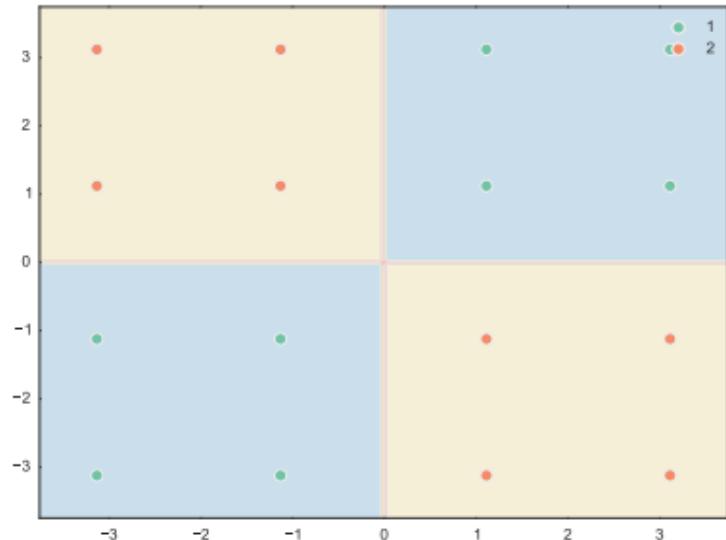
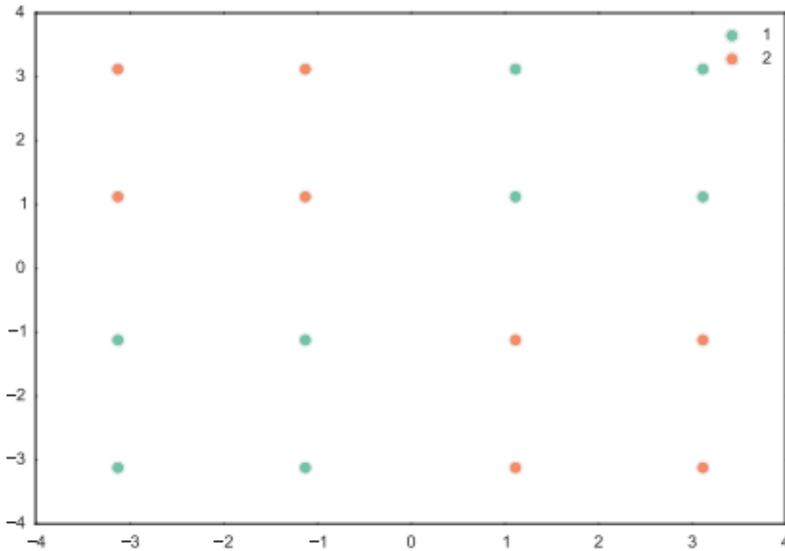
Fast to predict: Prediction simply involves traversing the tree from the root to a leaf node, following the rules as you descent the tree. Each rule is a comparison with a single feature.

Interpretable: Easy to visualize and interpret model. Produces a successive set of rules. Features near root of tree can be thought of as being more predictive.

Limitations of decision trees

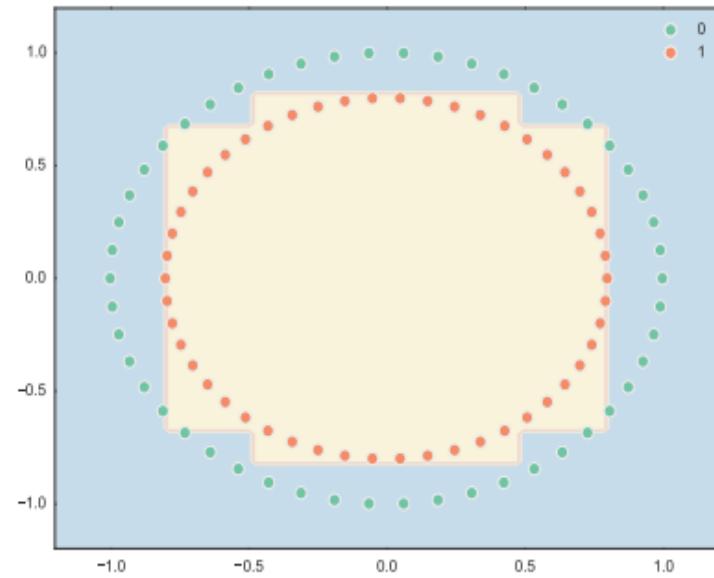
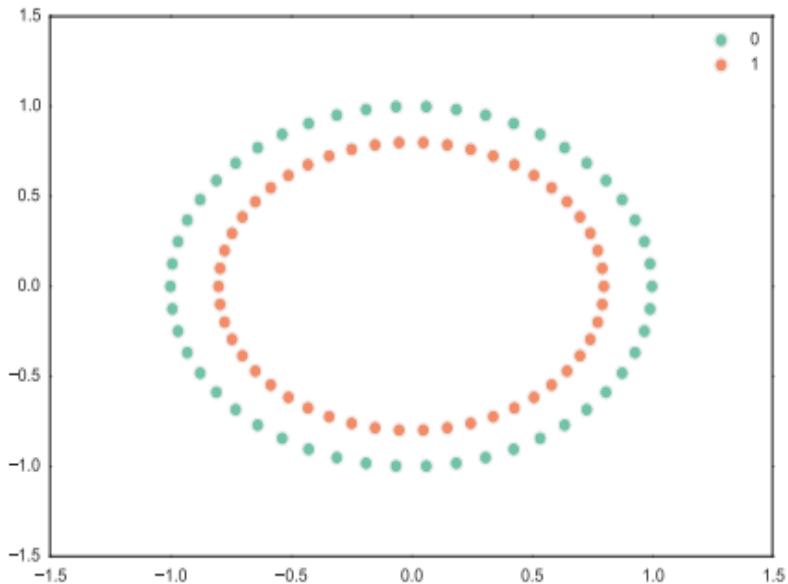
- Difficulty in breaking symmetry
- Difficulty producing smooth/simple decision boundaries
- Difficulty in capturing additive structure
- High variance

Limitations of decision trees: symmetry breaking

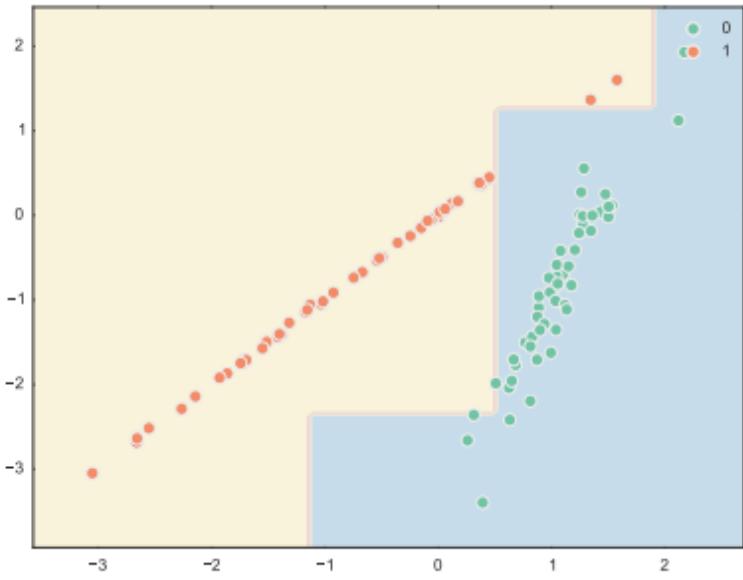
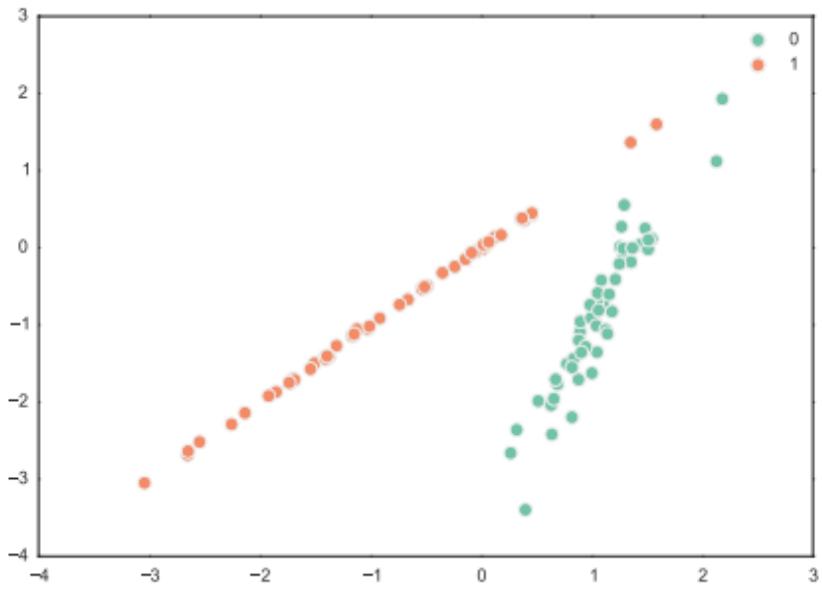


The above is also a good example of why rules to stop splitting if impurity reduction is too small are shortsighted.

Limitations of decision trees: complex decision boundaries



Limitations of decision trees: complex decision boundaries



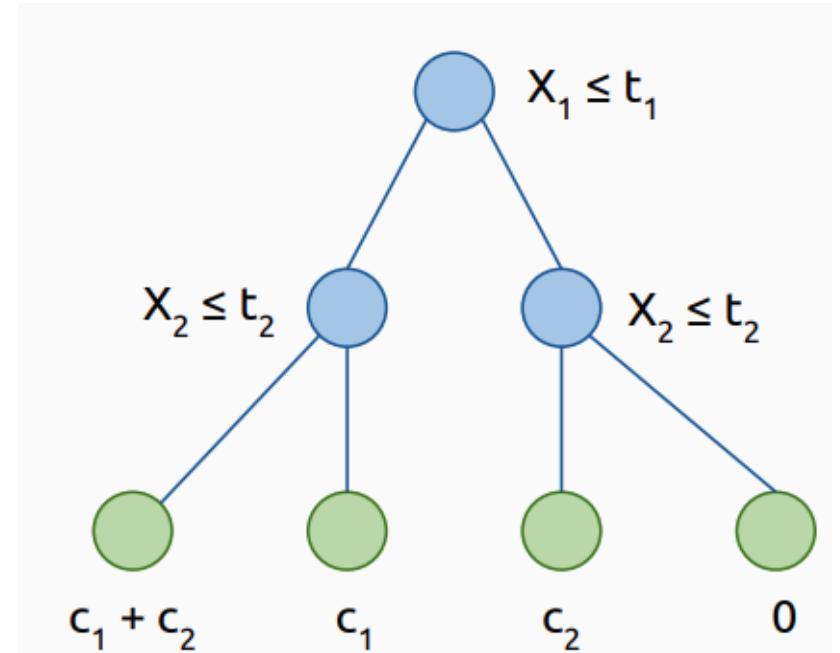
Limitations of decision trees: additive structure

$$y = c_1 \mathbb{1}(X_1 < t_1) + c_2 \mathbb{1}(X_2 < t_2) + c_3 \mathbb{1}(X_3 < t_3) + \dots + \epsilon$$

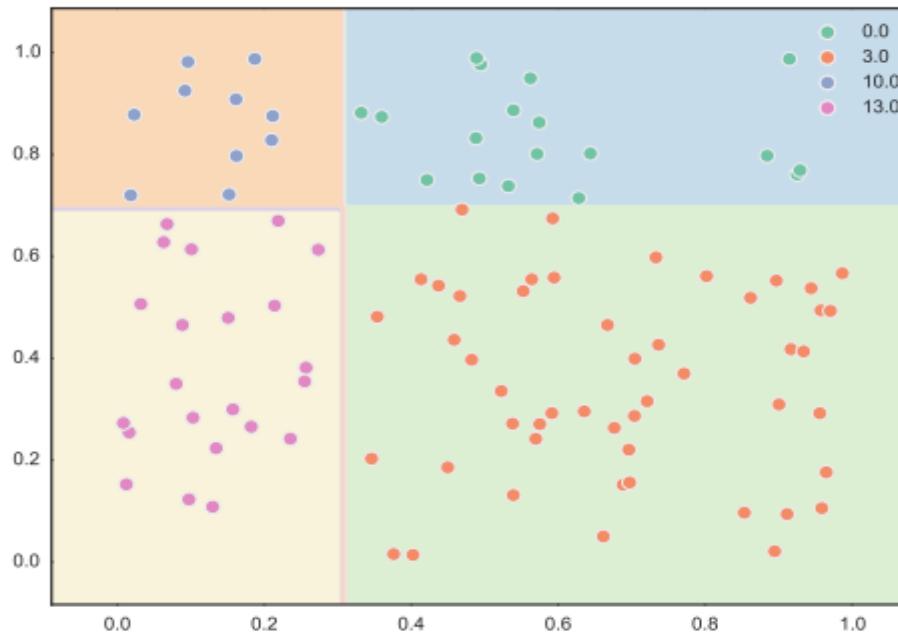
Assume tree first splits near t_1 . Then:

- Need 2 splits to capture $c_2 \mathbb{1}(X_2 < t_2)$
- Need 4 splits to capture $c_3 \mathbb{1}(X_3 < t_3)$
- Need 8 splits to capture $c_4 \mathbb{1}(X_4 < t_4)$
- ...

Need **exponentially more parameters** to capture such structure than a classifier that could model the additive structure directly.



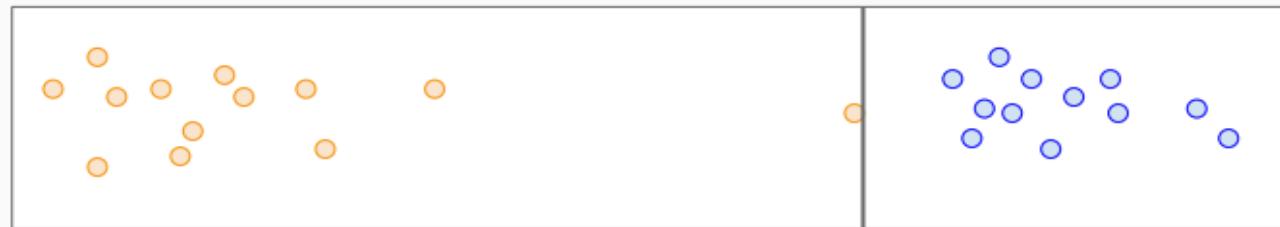
$$y = c_1 \mathbb{1}(X_1 < t_1) + c_2 \mathbb{1}(X_2 < t_2) + \epsilon$$



Limitations of decision trees: high variance

- The main drawback of using decision tree models is **high variance**.
- **Small changes in the data can produce very different trees.** These trees can make very different predictions on data outside of the training set.
- Makes trees very **sensitive to noise**. Trees are **not a robust classifier**.
- Also affects **interpretability** of the model. If small changes to the data cause very different trees, how interpretable are the rules in a given tree?

Limitations of decision trees: high variance



Further reading

The elements of statistical learning:

- Classification and regression trees: Section 9.2

Other resources

Jeff Miller's (mathematicalmonk) on CART: ML 2.1 :

<https://www.youtube.com/playlist?list=PLD0F06AA0D2E8FFBA>

Nando de Freitas' lectures

- Decision Trees

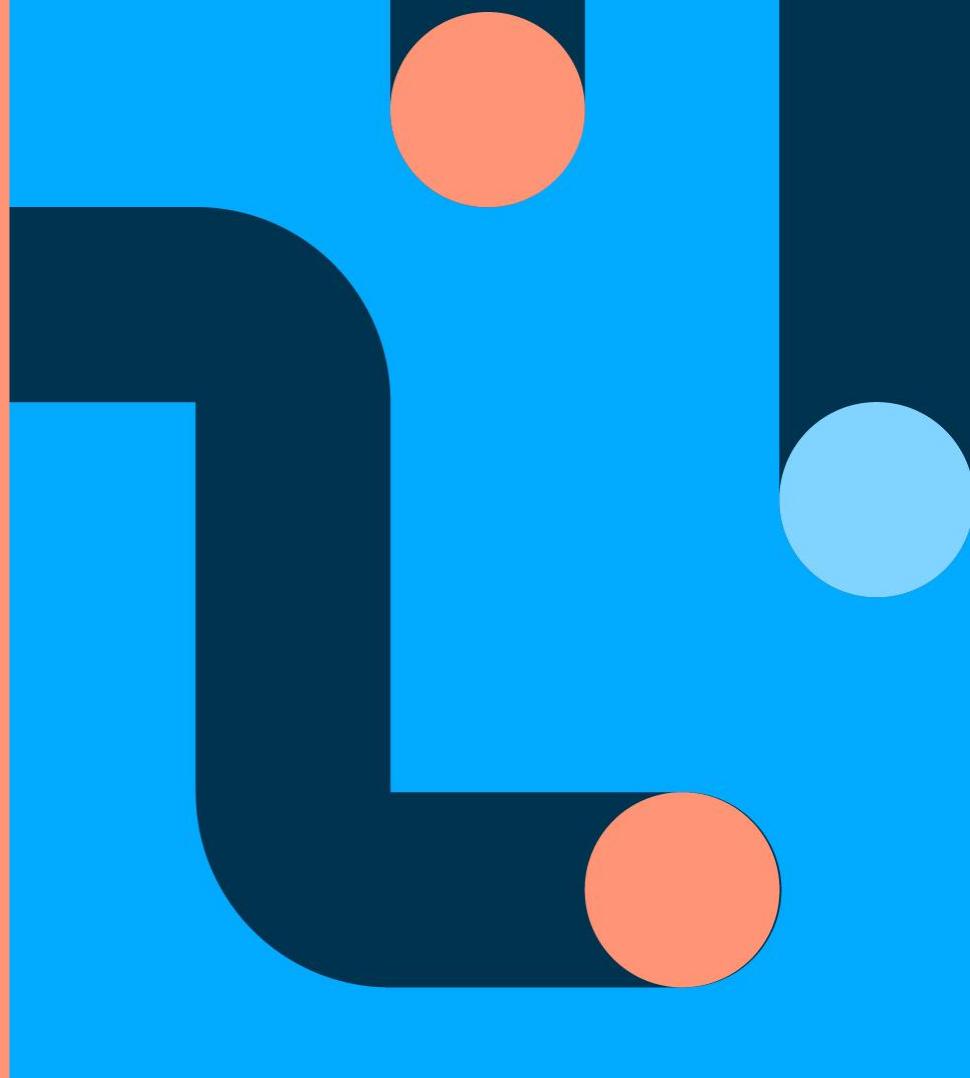
EEN1083

Data analysis and machine learning I

Ali Intizar



Ollscoil Chathair
Bhaile Átha Cliath
Dublin City University



Regularization

DCU

Ollscoil Chathair
Bhaile Átha Cliath
Dublin City University

Outline

Norms

Overfitting
Regularization

L2 regularization
L1 regularization

Norms

In mathematics, a norm is a function from a real or complex vector space to the non-negative real numbers that behaves in certain ways like the distance from the origin: it commutes with scaling, obeys a form of the triangle inequality, and is zero only at the origin. (Wikipedia)

In machine learning norms are a way to measure the size of a vector, matrix or tensor

Norms

Most common norm is the Euclidean norm or L_2 norm, denoted $\|\mathbf{x}\|_2$ or just $\|\mathbf{x}\|$. Gives the length of the vector.

$$\|\mathbf{x}\| = \sqrt{\mathbf{x}^T \mathbf{x}} = \sqrt{\sum_i x_i^2}$$

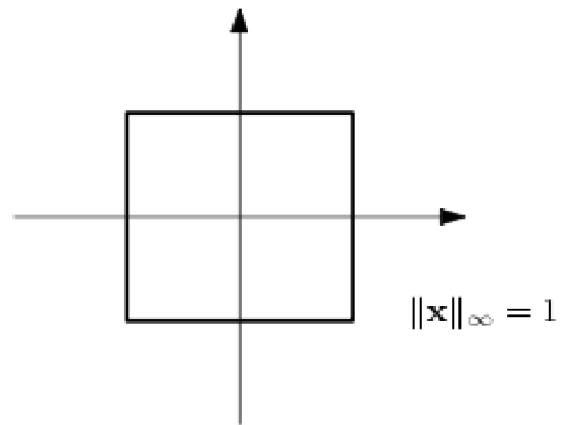
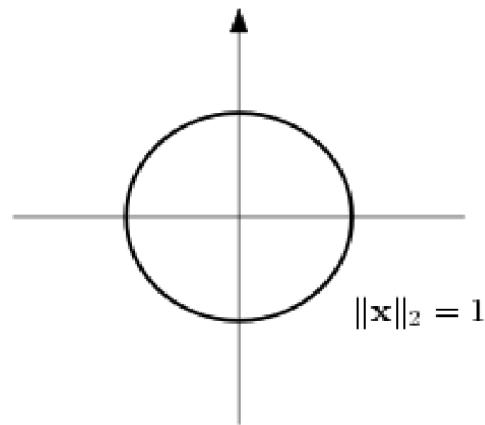
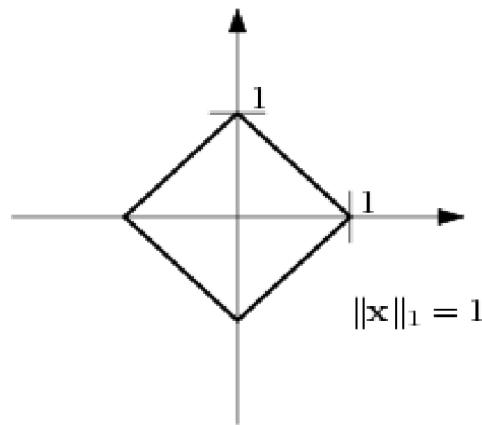
The squared Euclidean norm $\|\mathbf{x}\|_2^2$ is often useful:

$$\|\mathbf{x}\|^2 = \mathbf{x}^T \mathbf{x} = \mathbf{x} \cdot \mathbf{x}$$

Another common norm is the L_1 norm, which is the sum of absolute values of \mathbf{x}

$$\|\mathbf{x}\|_1 = \sum_i |x_i|$$

Norms



Properties of norms

A norm is any function p from vectors to \mathbb{R} that satisfies:

1. $p(\alpha \mathbf{x}) = |\alpha| p(\mathbf{x})$ for $\alpha \in \mathbb{R}$
2. $p(\mathbf{x} + \mathbf{y}) \leq p(\mathbf{x}) + p(\mathbf{y})$ (triangle inequality)
3. $p(\mathbf{x}) = 0 \iff \mathbf{x} = \mathbf{0}$

E.g. $\|5\mathbf{x}\| = 5\|\mathbf{x}\|$ for any norm $\|\cdot\|$.

We also have $p(-\mathbf{x}) = p(\mathbf{x})$

Norms and distance metrics

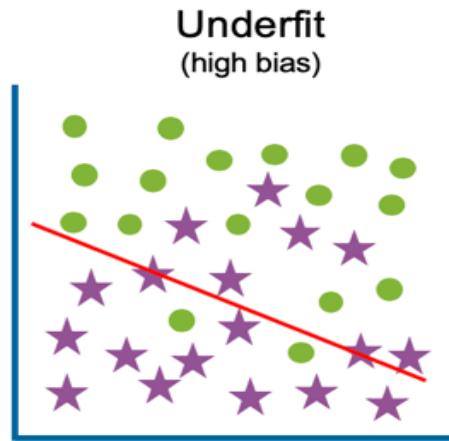
Any vector space V and norm p can be used to induce a distance metric (define a **metric space**) by defining the distance metric to be $d(\mathbf{x}, \mathbf{y}) = p(\mathbf{x} - \mathbf{y})$

E.g. The space of D dimensional real valued vectors \mathbb{R}^D and the L_2 norm give the Euclidean space with metric $d(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|_2$.

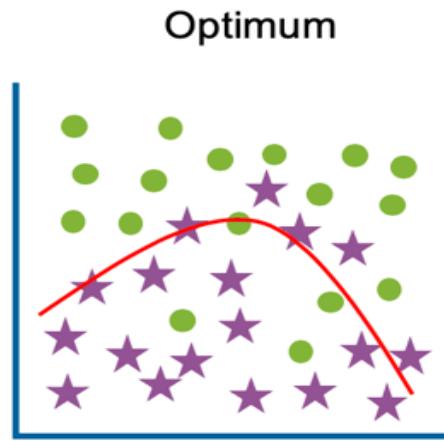
In 2D this gives the familiar Euclidean distance between two points \mathbf{x} and \mathbf{y} :

$$d(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|_2 = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2}$$

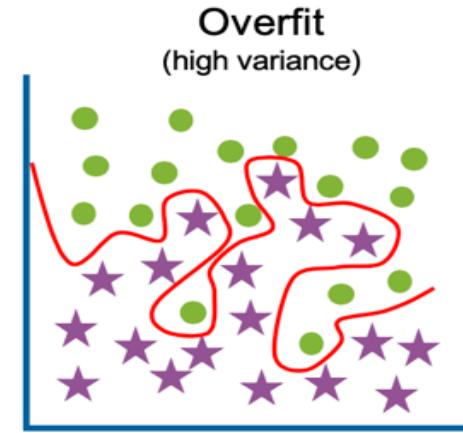
Overfitting



High training error
High test error

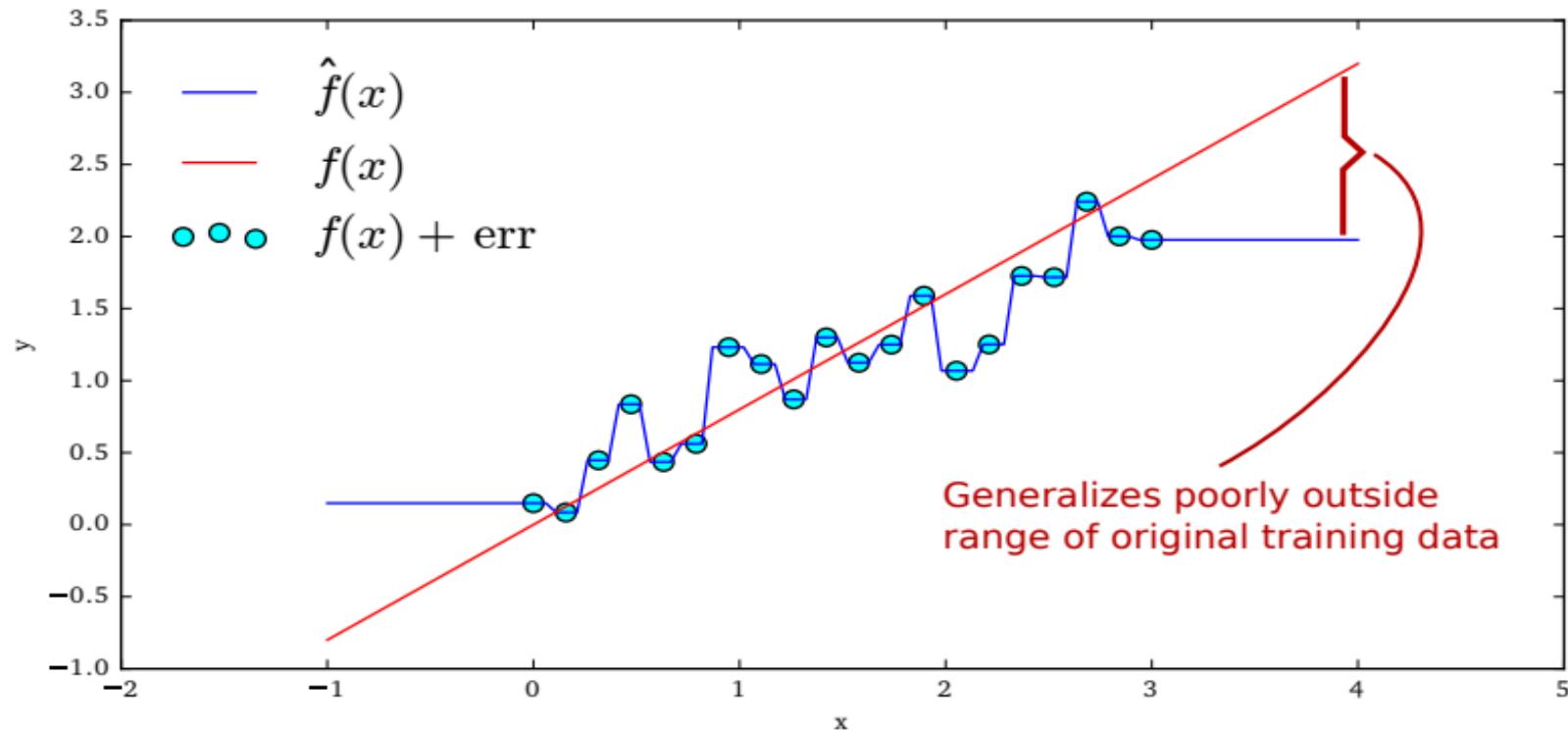


Low training error
Low test error



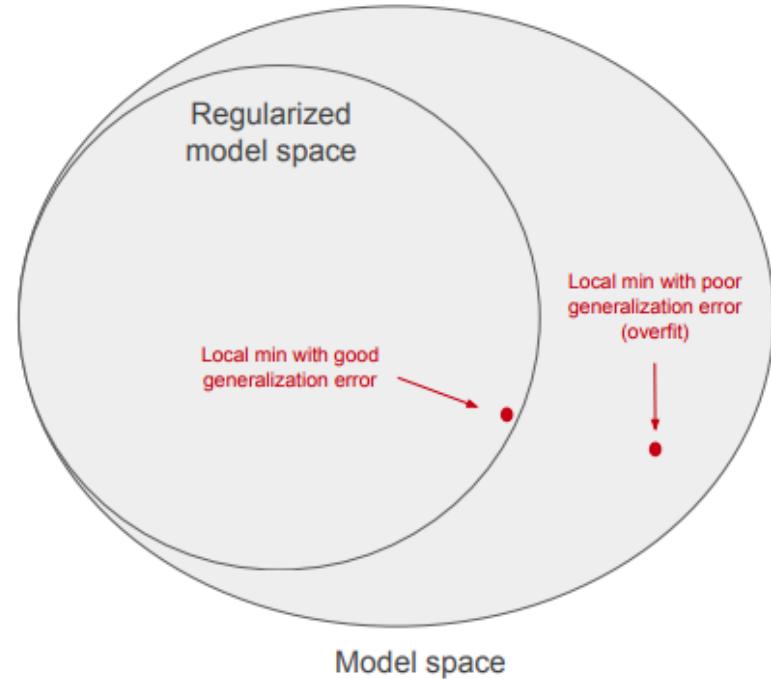
Low training error
High test error

Nearest neighbours generalisation



Structural risk minimization

- **Structural risk minimization:** prevent overfitting by balancing model complexity with success at fitting training data.
- Idea: Given two models (hypotheses) with similar training error, prefer ones with lower complexity.
- **Regularization:** add additional information to solve an ill-posed problem or prevent overfitting. Additional information is often a **penalty** included in the loss function.



L2 regularization

Add a penalty to the loss function for large weights.

Penalty is on the L_2 norm of the weights

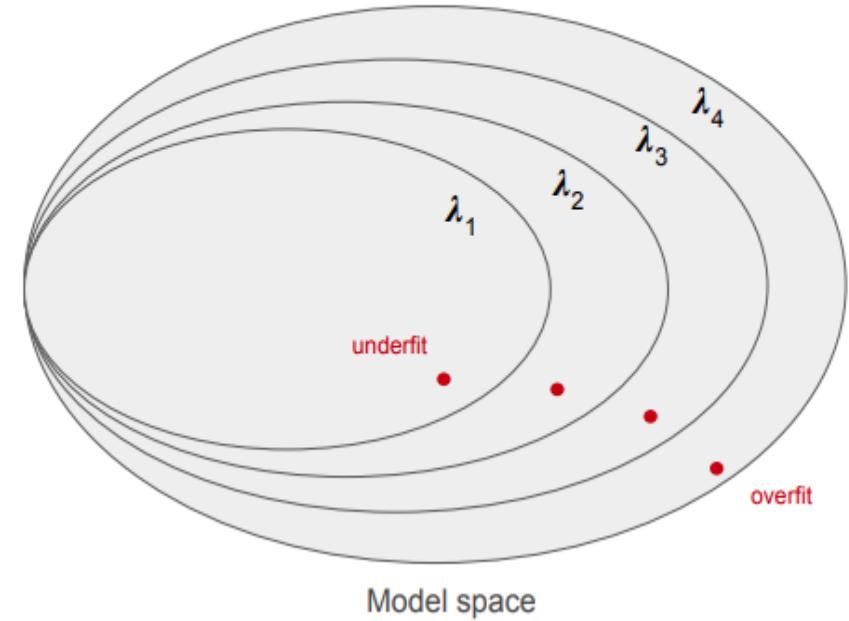
$$\|\mathbf{w}\|_2^2 = \mathbf{w}^T \mathbf{w} = \sum_{i=1}^D w_i^2$$

The modified loss is:

$$\mathcal{L} = \mathcal{L}_{\text{data}} + \frac{\lambda}{2} \|\mathbf{w}\|_2^2$$

where λ is the **regularization parameter**, which controls the strength of the regularization

$$\lambda_1 > \lambda_2 > \lambda_3 > \lambda_4$$



L2 regularization and gradient descent

L_2 regularized loss:

$$\mathcal{L} = \mathcal{L}_{\text{data}} + \frac{\lambda}{2} \|\mathbf{w}\|_2^2$$

Taking the gradient wrt. \mathbf{w} we get

$$\nabla_{\mathbf{w}} \mathcal{L} = \nabla_{\mathbf{w}} \mathcal{L}_{\text{data}} + \lambda \mathbf{w}$$

which gives the gradient descent update rule:

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \alpha \nabla_{\mathbf{w}} \mathcal{L}_{\text{data}} - \alpha \lambda \mathbf{w}_t$$

This can be understood as slowly *decaying* the weights toward zero with each iteration. In the neural networks literature L_2 regularization is known as **weight decay**.

Why the L2 penalty is reasonable

Imagine we had a training set for which the first two features are always equal $x_1 = x_2$. E.g. this could happen due to a broken sensor, or an image with a watermark.

	x_1	x_2	x_3	x_4	\dots	x_N	y
\mathbf{x}_1	1	1	4	7	\dots	2	0
\mathbf{x}_2	1	1	2	1	\dots	6	1
\mathbf{x}_3	1	1	9	2	\dots	5	1
:	:	:	:	:	:	14	:

If the decision function $f(x) = \mathbf{w}^T \mathbf{x} + b = w_1x_1 + w_2x_2 + \dots + w_Nx_N + b$, then we can set $w_1 = -w_2$ to any constant we want without changing the outcome or the loss.

Why the L2 penalty is reasonable

Lets say we choose $w_1 = 1000, w_2 = -1000$.

We observe a test point in which $x_2 = 0$. Now we have $f(x) = 1000x_1 + 1000(0) + \dots$, which changes the value of the decision function by +1000!

With large weights, small changes in input can produce large changes in output!

If we chose $w_1 = w_2 = 0$, the error on the training set would be the same and there would have been no such problem with the test point.

The L_2 penalty encourages finding solutions with **smaller weights**.

Ridge regression

Applying the L_2 penalty to linear regression gives a model called **ridge regression**.
The ridge regression loss is:

$$\mathcal{L} = \frac{1}{2} \sum_{i=1}^N (y - f(\mathbf{x}))^2 + \frac{\lambda}{2} \|\mathbf{w}\|_2^2$$

The optimal parameters are the solution to

$$\arg \min_{\mathbf{w}} L$$

As with linear regression, this can be solved in closed form by taking derivatives and setting to zero. The solution turns out to be:

$$\hat{\mathbf{w}} = (X^T X + \lambda I)^{-1} X^T \mathbf{y}$$

Choosing the regularization parameter

The regularization parameter λ controls the tradeoff between having small weights and fitting the training data.

- Smaller values for λ place more emphasis on fitting the training data and may be more suitable in low-noise settings or when there are few parameters to fit.
- Larger values for λ place more emphasis on making the values of w small, and are more suitable when there is a lot of noise or many parameters to fit.

As usual, the best way to choose an appropriate value for λ is using model selection procedures like a hold-out validation set or cross-validation.

Notes on L2 regularization

L_2 regularization is also known as **Tikhonov regularization**.

L_2 regularization in ridge regression corresponds to a Gaussian prior on the weights with zero mean and variance proportional to $1/\lambda$.

Can be used to find least squares solutions to ill-posed problems:

- more free variables than constraints,
- $(X^T X)$ is singular,
- $(X^T X + \lambda I)$ is invertible.

Essential for overparameterized problems like softmax regression.

L1 regularization

L_2 regularization encourages solutions with small weights.

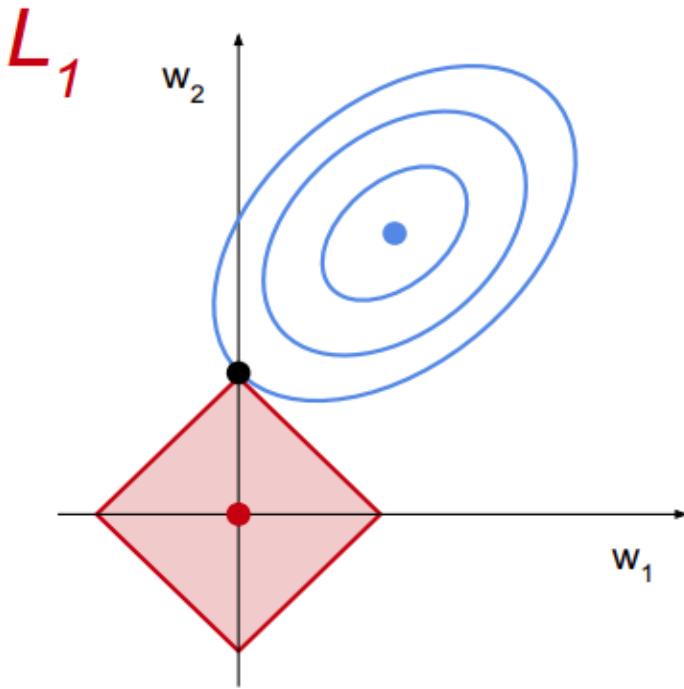
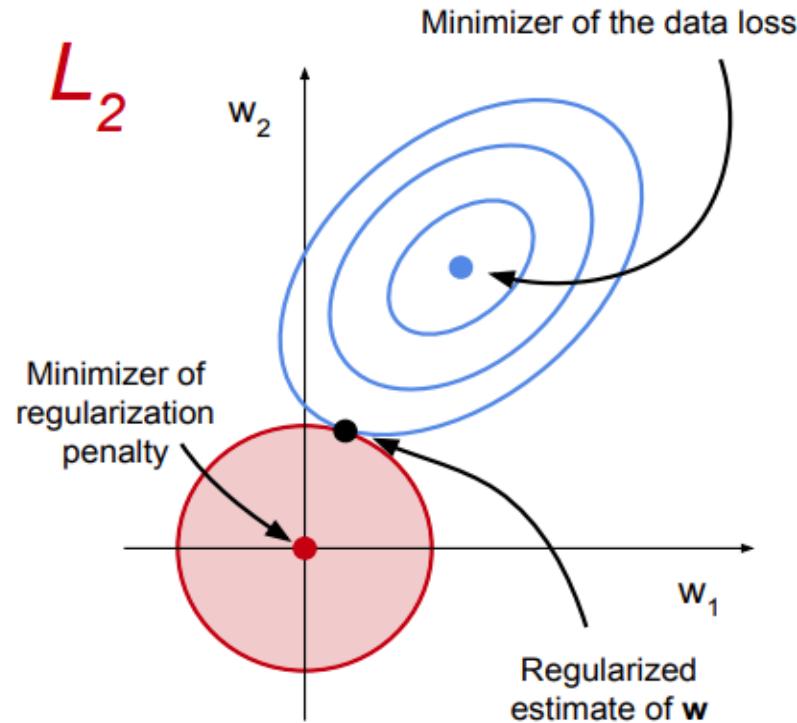
Sometimes you want to find solutions in which most of the weights are exactly zero. These are called **sparse** solutions.

E.g. **feature selection**: you have many features, and you want your decision function to only use a subset. Any feature with weight zero is not used.

L_1 regularization encourages sparse solutions by using penalizing the L_1 -norm of the weights.

$$\|\mathbf{w}\|_1 = \sum_{i=1}^D |w_i|$$

How does L1 regularization encourage sparsity?



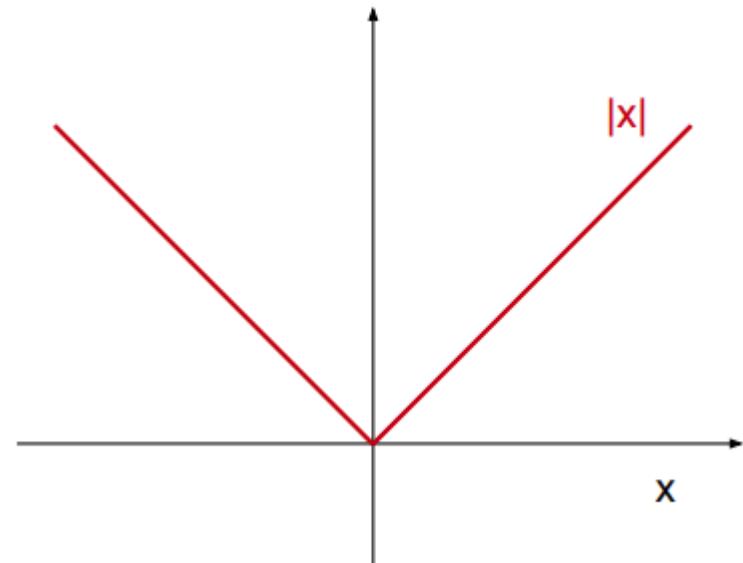
(Sub)-gradient descent for L1 regularization

L_1 regularized loss:

$$\begin{aligned}\mathcal{L} &= \mathcal{L}_{\text{data}} + \lambda \|\mathbf{w}\|_1 \\ &= \mathcal{L}_{\text{data}} + \lambda \sum_{i=1}^D |w_i|\end{aligned}$$

The absolute value has a discontinuity at zero so gradient is not defined everywhere.

For gradient descent we can use a **subgradient**.



(Sub)-gradient descent for L1 regularization

A subgradient for $|w|$ is:

$$\nabla_w |w| = \begin{cases} 1 & w > 0 \\ -1 & w < 0 \\ 0 & w = 0 \end{cases}$$

which can be written more succinctly using the sign function $\nabla_w |w| = \text{sign}(w)$.

This gives us the subgradient descent update rule for L_1 regularization:

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \alpha \nabla_{\mathbf{w}} \mathcal{L} - \lambda \sum_{i=0}^D \text{sign}(w_i)$$

The LASSO and Elastic Net

Combining ordinary least squares regression with the L_1 penalty gives rise to the so called **LASSO** (least absolute shrinkage and selection operator).

Can solve using subgradient descent as shown. Generally preferable to use a faster algorithm like **least angle regression** (LARS).

It is also possible to use **both** L_1 and L_2 regularizers together. This model is called the **elastic net**.

In scikit-learn

- `sklearn.linear_model.Ridge`
- `sklearn.linear_model.Lasso`
- `sklearn.linear_model.LassoLars`
- `sklearn.linear_model.ElasticNet`

Logistic regression in scikit-learn can also take an L_1 or L_2 regularization penalty:
`LogisticRegression(penalty='l2', C=1.0)`, where $C > 0$ is the **inverse** of λ
(smaller values specify stronger regularization).

Scikit-learn also has classes that do automatic (and optimized) cross validation to figure out the best value for the regularization parameter(s). E.g.:

- `sklearn.linear_model.RidgeCV`
- `sklearn.linear_model.LassoCV`

Resources

Stanford machine learning lectures (Andrew Ng):

- Lecture 3: linear and logistic regression
<http://www.youtube.com/watch?v=HZ4cvaztQEs>
- Lecture 4: Generalized linear models
<http://www.youtube.com/watch?v=nLKOQfKLUks>

Caltech machine learning lectures (Yaser Abu-Mostafa):

- Lecture 12: Regularization
[http://www.youtube.com/watch?v=I-VfYXzC5ro²⁵](http://www.youtube.com/watch?v=I-VfYXzC5ro)

Resources

Oxford deep learning lectures (Nando de Freitas):

- Lecture 4: Regularization 1

http://www.youtube.com/watch?v=VR0W_PNwLGw

- Lecture 5: Regularization 2

http://www.youtube.com/watch?v=VR0W_PNwLGw

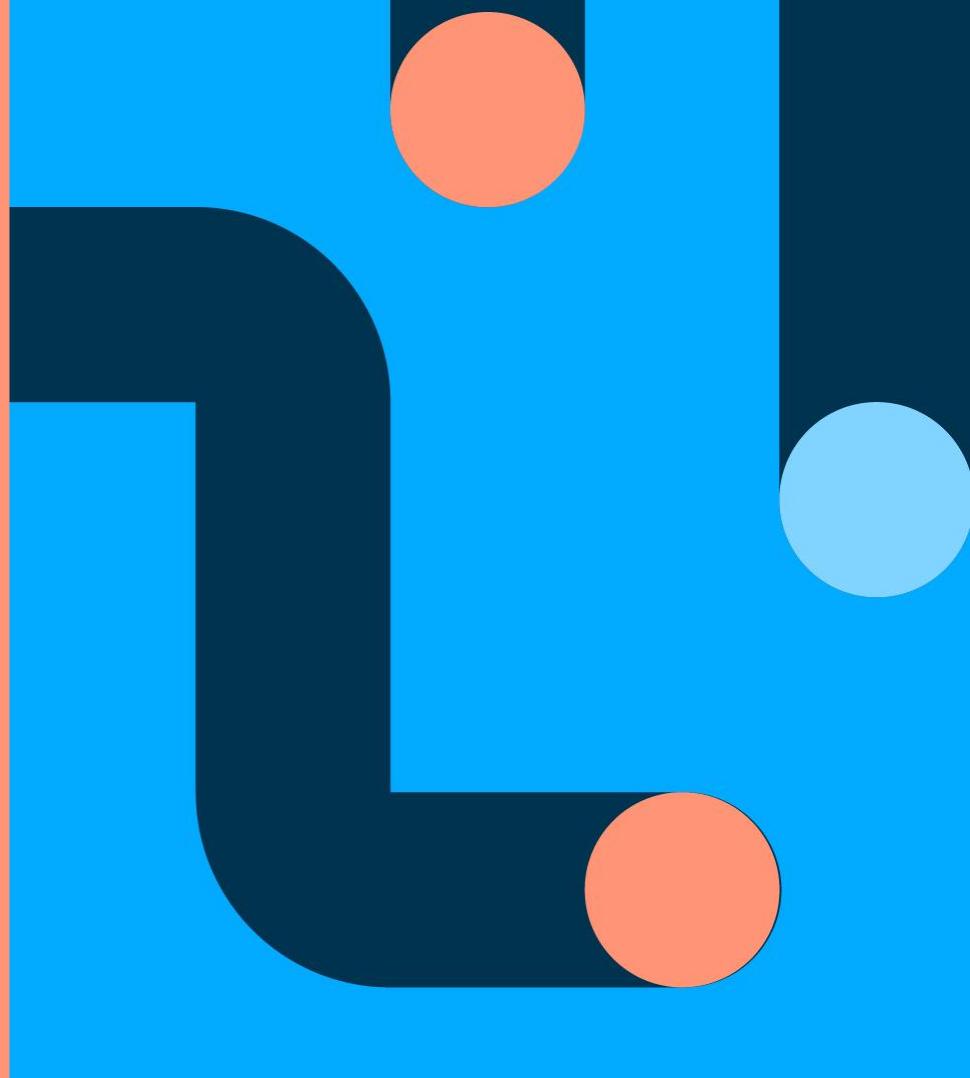
EEN1083

Data analysis and machine learning I

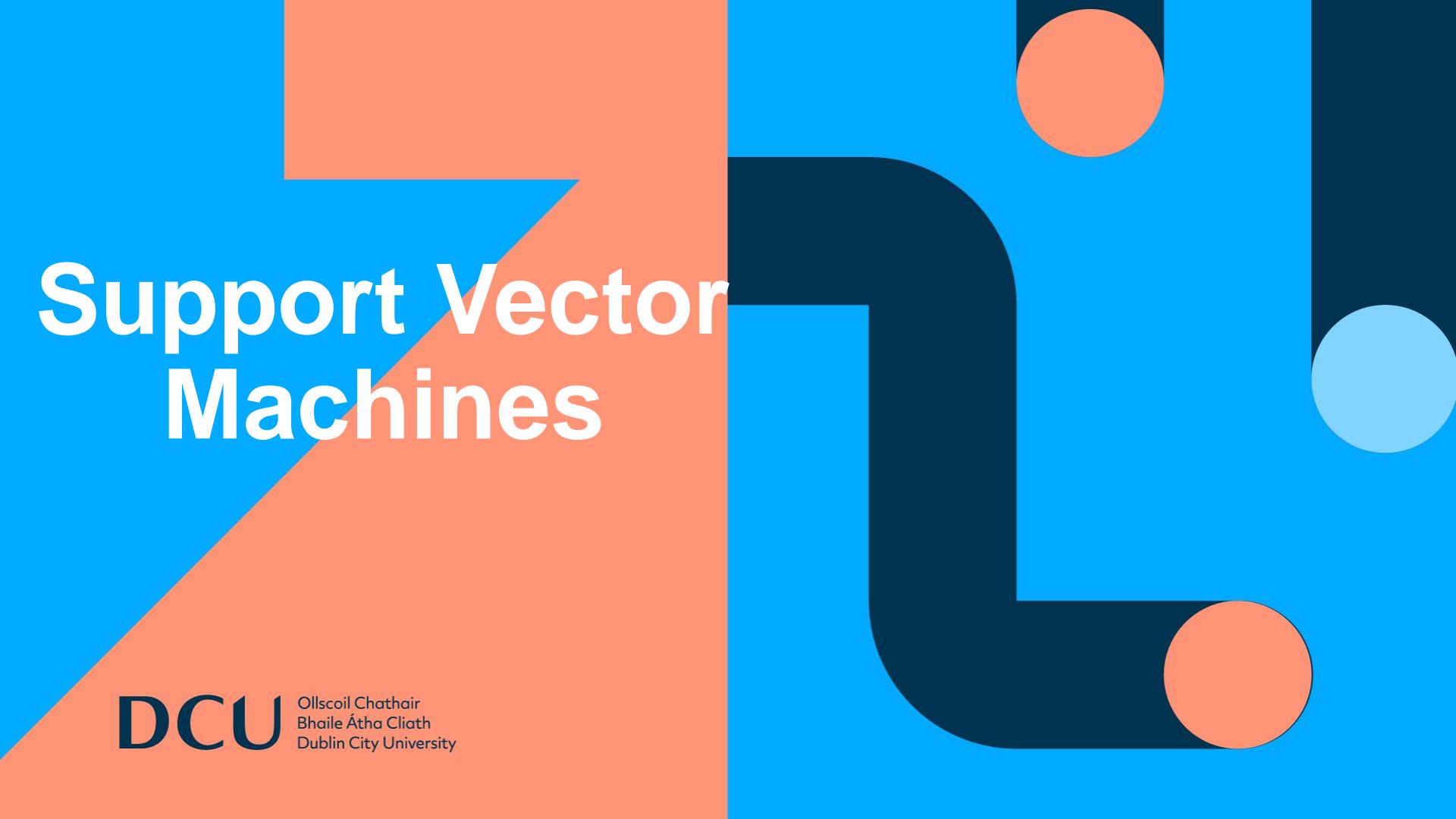
Ali Intizar



Ollscoil Chathair
Bhaile Átha Cliath
Dublin City University



Support Vector Machines

The background features a large blue rectangle on the left and a large orange rectangle on the right. Overlaid on these are several circles: a large orange circle at the top center, a smaller blue circle on the right side, and two smaller orange circles near the bottom right corner. A dark blue curved shape also borders the orange area on the right.

DCU

Ollscoil Chathair
Bhaile Átha Cliath
Dublin City University

Outline

Hard-margin linear SVM

Soft-margin linear SVM

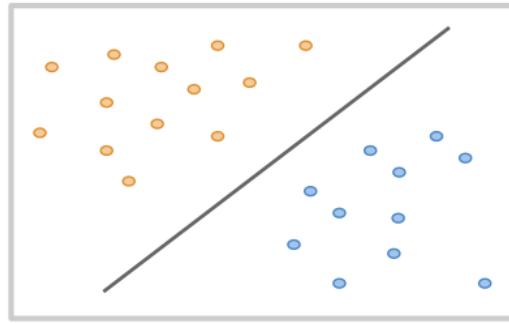
Kernel SVMs

3

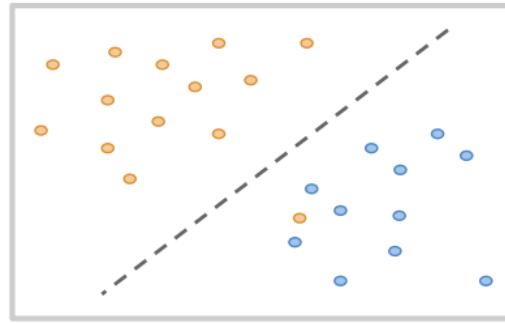
Tips for use

Hard-margin linear SVM

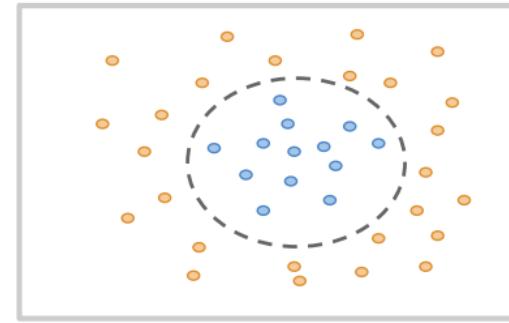
Three problems



Linearly separable



Almost linearly separable



Not linearly separable

The binary classification setup

Training set:

$$\{(\mathbf{x}_i, y_i)\}_{i=1}^N \quad \mathbf{x}_i \in \mathbb{R}^D, \quad y_i \in \{-1, +1\}$$

Linear prediction function:

$$f(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b$$

And we would like that:

$$f(\mathbf{x}_i) > 0 \quad \text{if } y_i = +1$$

$$f(\mathbf{x}_i) < 0 \quad \text{if } y_i = -1$$

Which we can also write as:

$$y_i f(\mathbf{x}_i) > 0$$

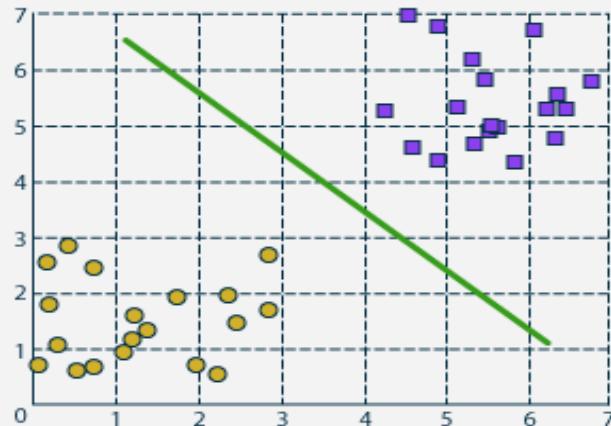
$$y_i (\mathbf{w}^T \mathbf{x}_i + b) > 0$$

Hyperplane geometry

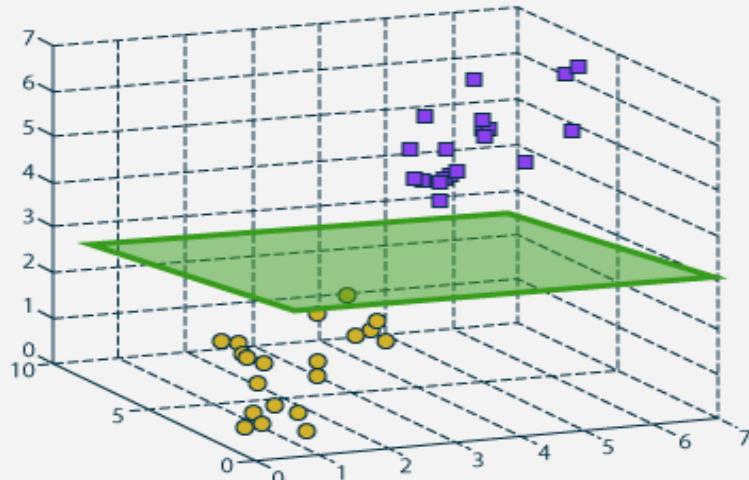
Hyperplane



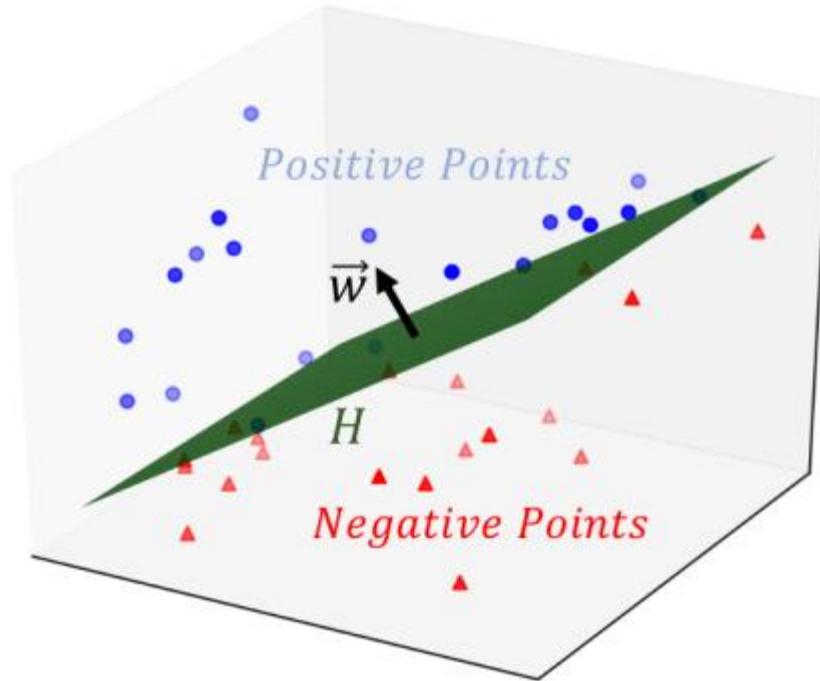
A Hyperplane in R^2 is a line



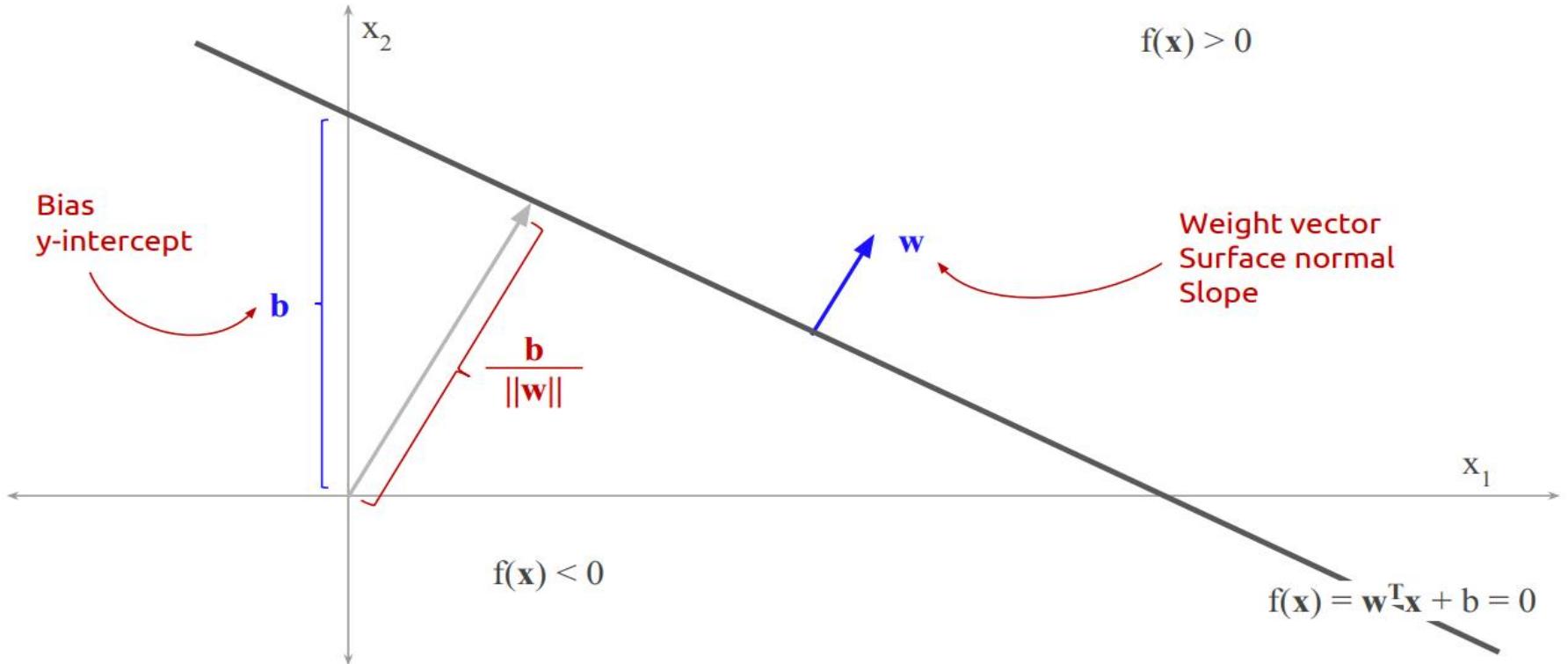
A Hyperplane in R^3 is a plane



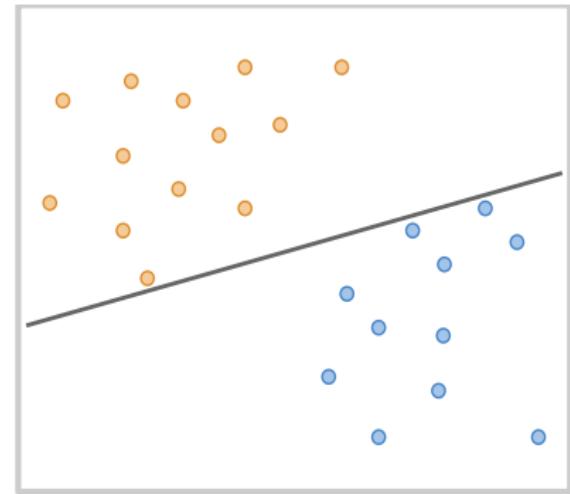
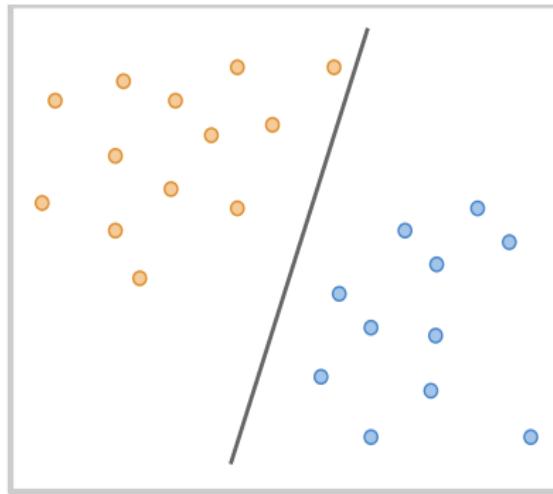
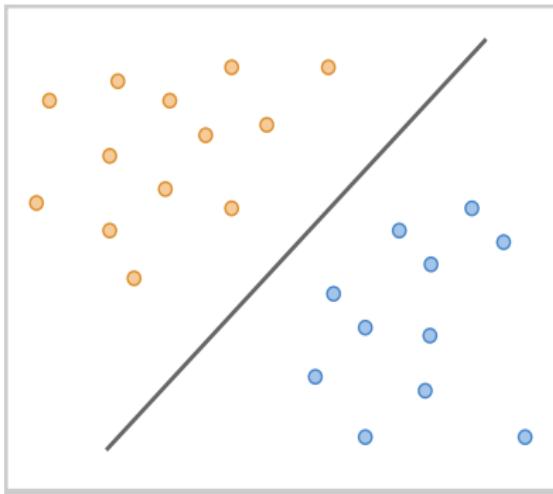
Hyperplane geometry



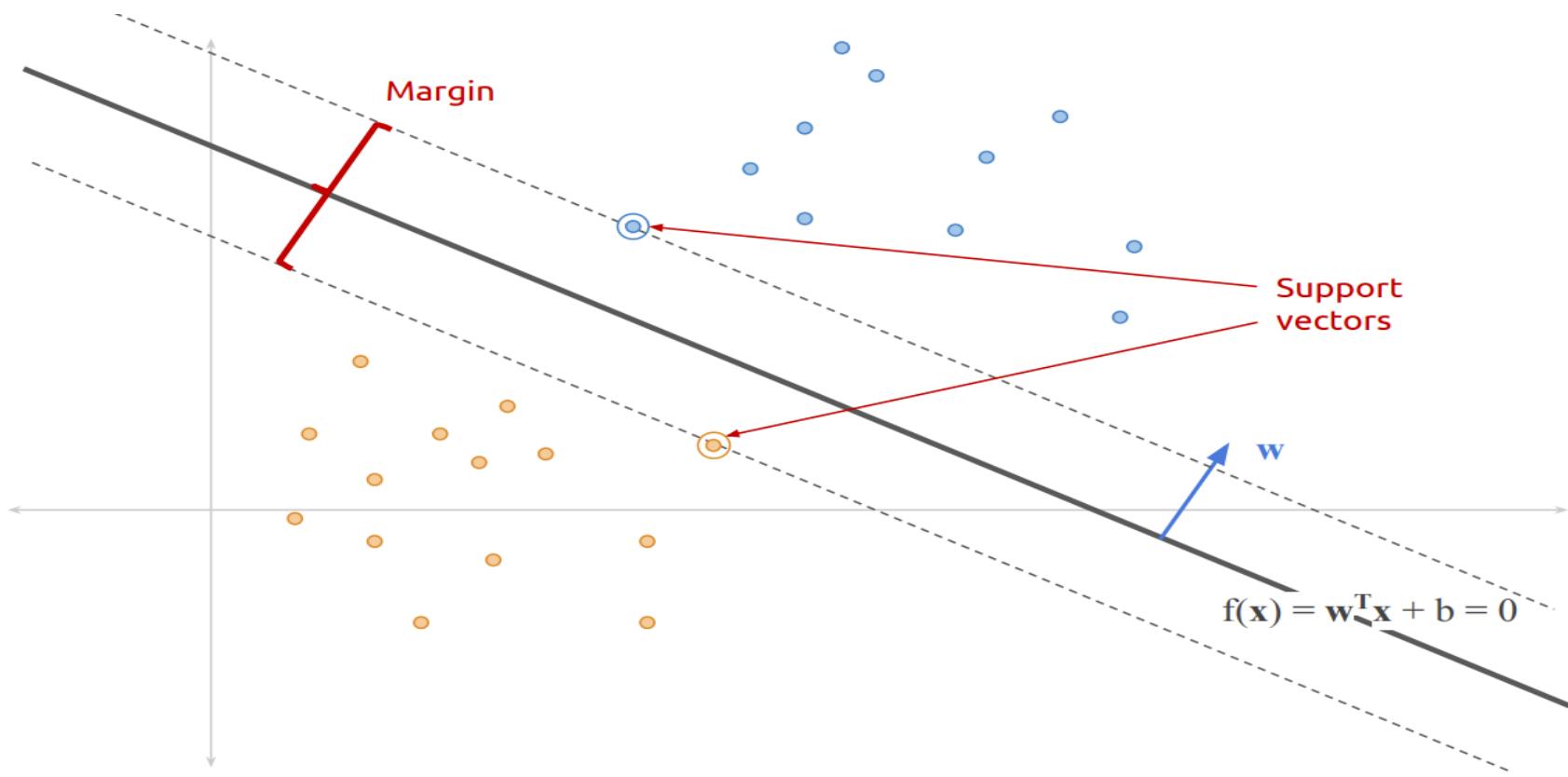
Hyperplane geometry



Which separating hyperplane is the best?



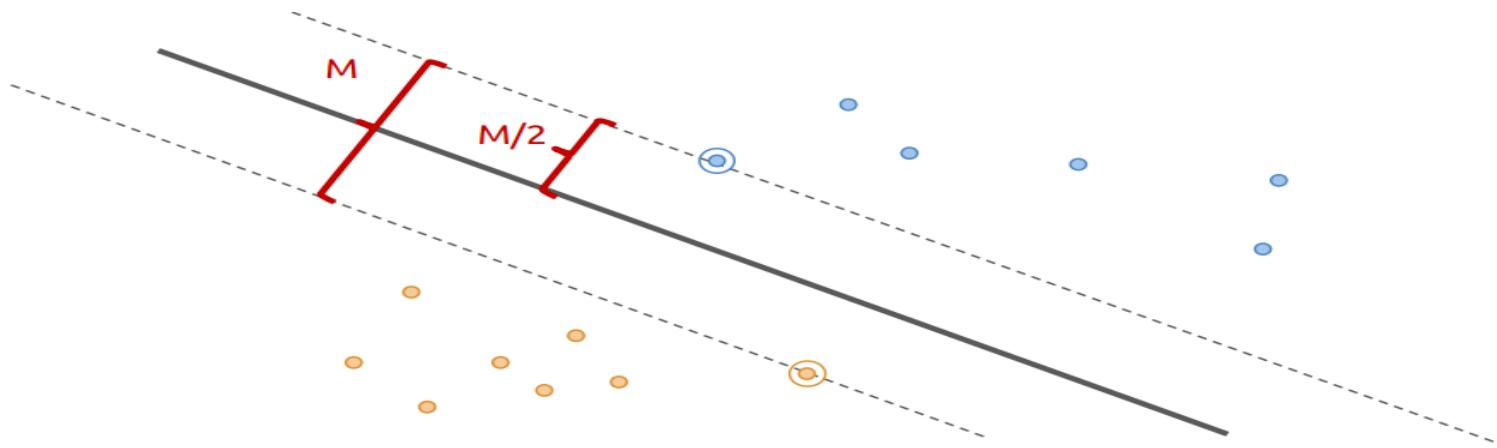
Max-margin principle



Max-margin principle

maximize M
 \mathbf{w}, b

subject to $\frac{y_i}{\|\mathbf{w}\|} (\mathbf{w}^T \mathbf{x}_i + b) \geq \frac{M}{2}, \quad i = 1, \dots, N.$



Python code to solve hard margin SVM with Quadratic Optimizer

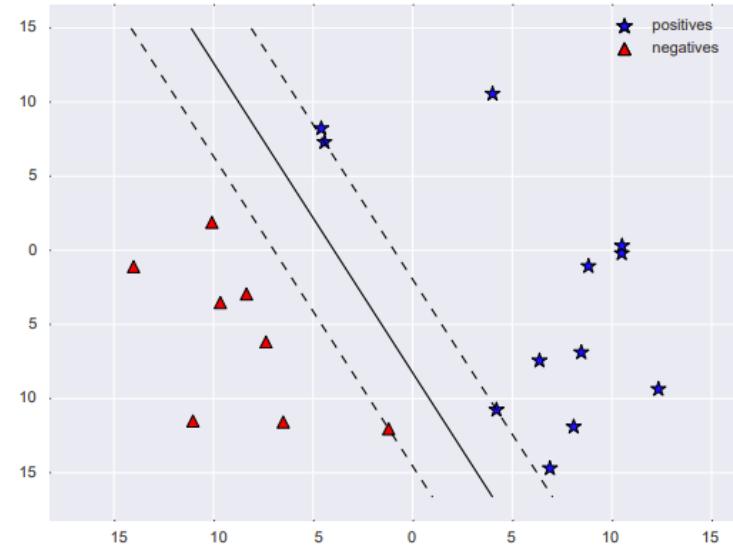
```
class HardMarginSVM(object):
    def fit(self, X, y):
        n, m = X.shape
        mat = cvxopt.matrix

        # Formulate QP
        P = mat(np.eye(m+1))
        P[-1,-1] = 0
        q = mat(0.0, (m+1, 1))
        G = mat(np.c_[-y[:,np.newaxis] * X, -y])
        h = mat(-1.0, (n, 1))

        # Solve QP
        solution = cvxopt.solvers.qp(P, q, G, h)

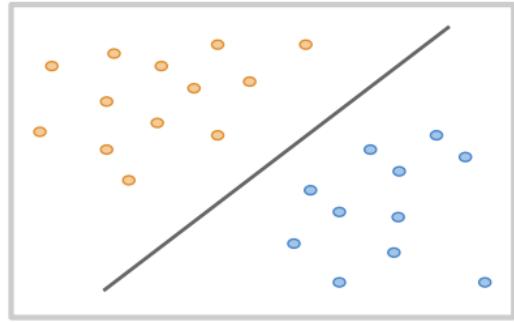
        # Make sure we have the optimal solution
        if solution['status'] != 'optimal':
            raise ValueError('infeasible')

        # Assign model parameters w, b
        x = np.array(solution['x'])
        self.w = x[:-1].ravel()
        self.b = x[-1][0]
```

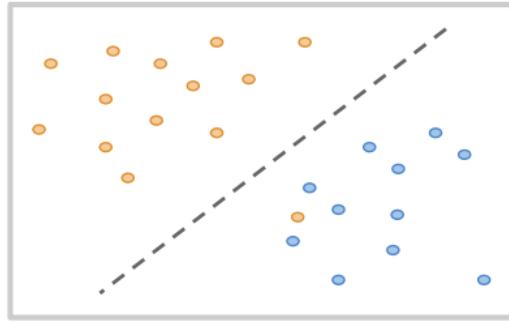


Soft-margin linear SVM

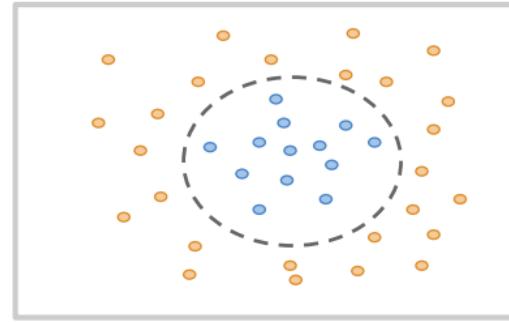
Three problems



Linearly separable

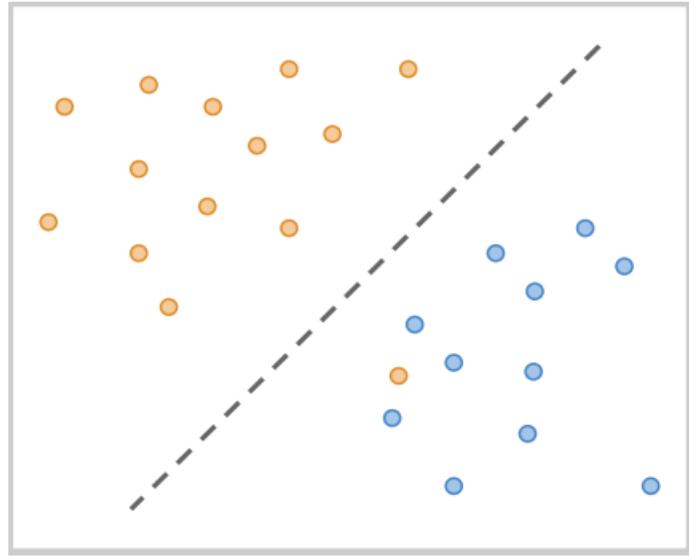


Almost linearly separable

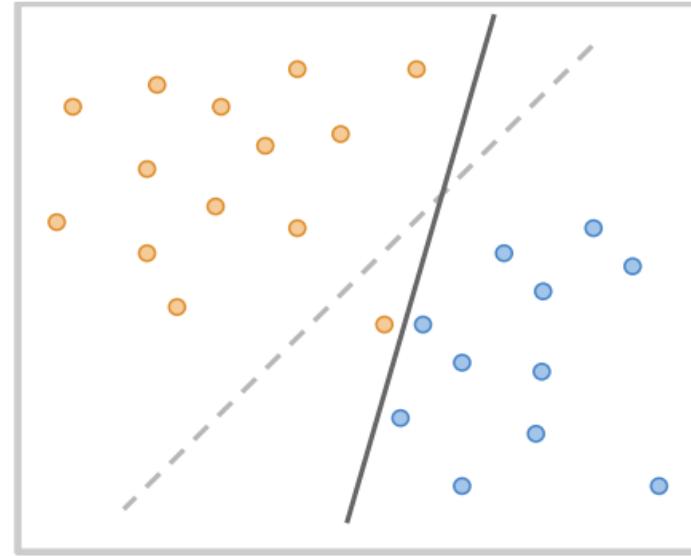


Not linearly separable

Motivating examples



Almost linearly separable



Linearly separable

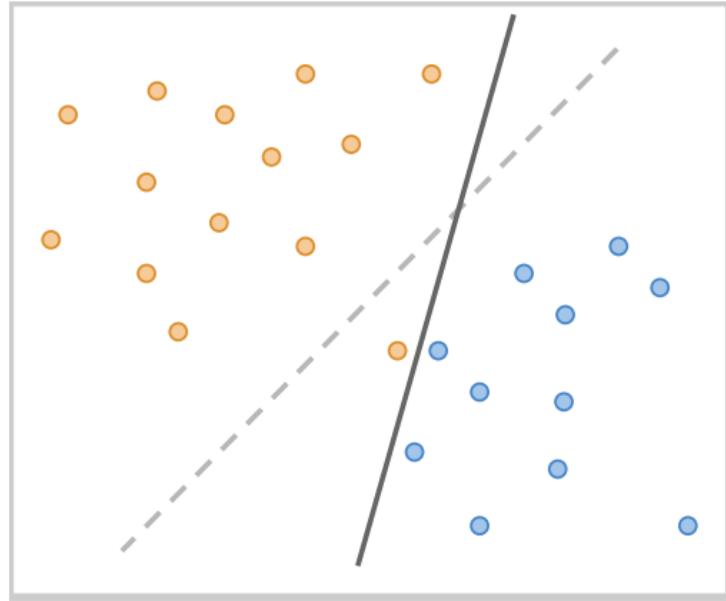
Motivating examples

Even when linearly separable, max margin solution can have a very narrow margin.

We could get a much larger margin if we ignored the constraint on the one outlier.

Trade off between satisfying constraints and maximizing the margin.

How do we allow the classifier to violate constraints? **Turn hard constraints into soft ones.**



Linearly separable

Objective

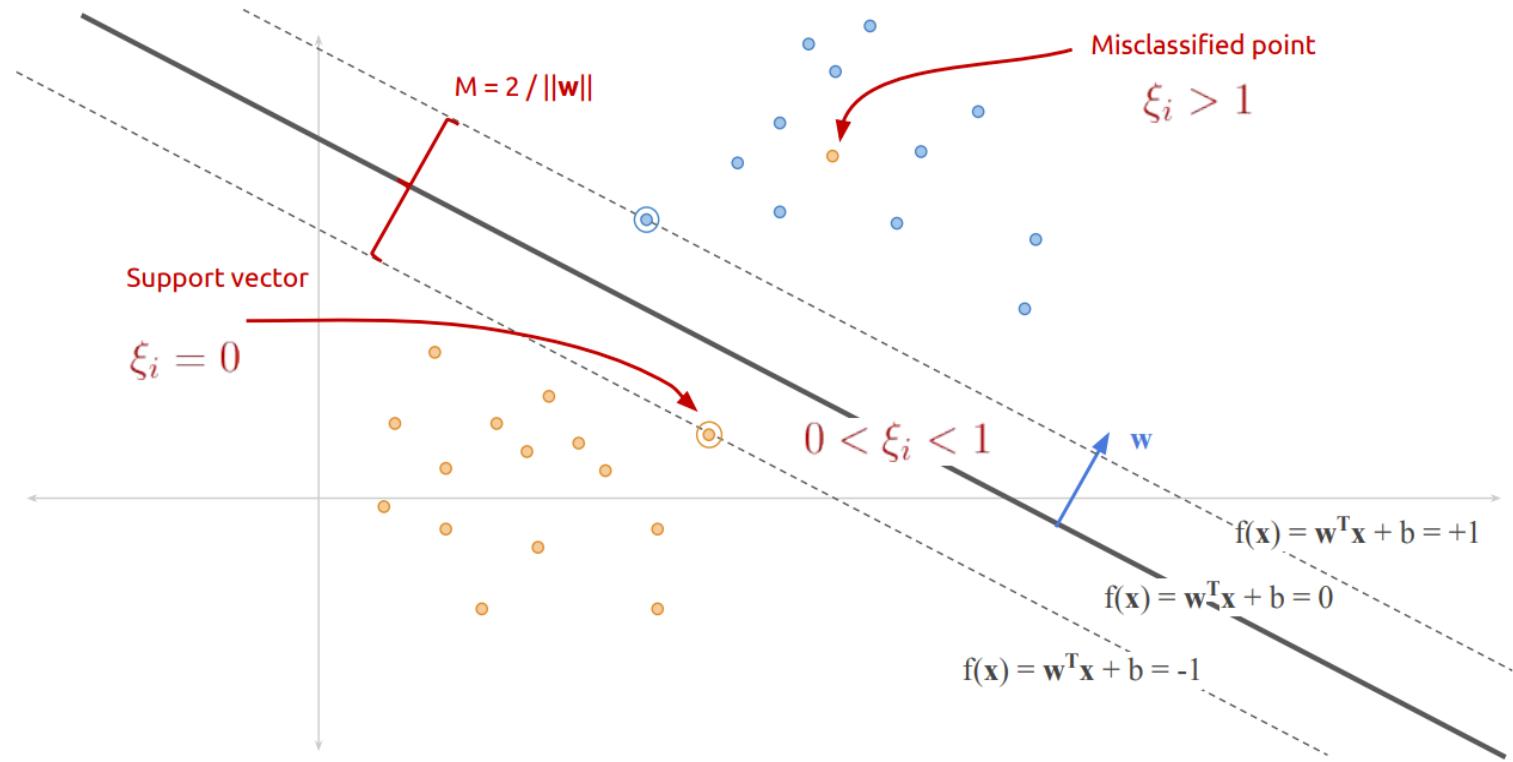
Hard-margin SVM:

$$\begin{aligned} & \underset{\mathbf{w}, b}{\text{minimize}} \quad \frac{1}{2} \|\mathbf{w}\|^2 \\ & \text{subject to} \quad y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1 \end{aligned}$$

Soft-margin SVM:

$$\begin{aligned} & \underset{\mathbf{w}, b, \xi}{\text{minimize}} \quad \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^N \xi_i \\ & \text{subject to} \quad y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1 - \xi_i \\ & \quad \xi_i \geq 0 \end{aligned}$$

Objective



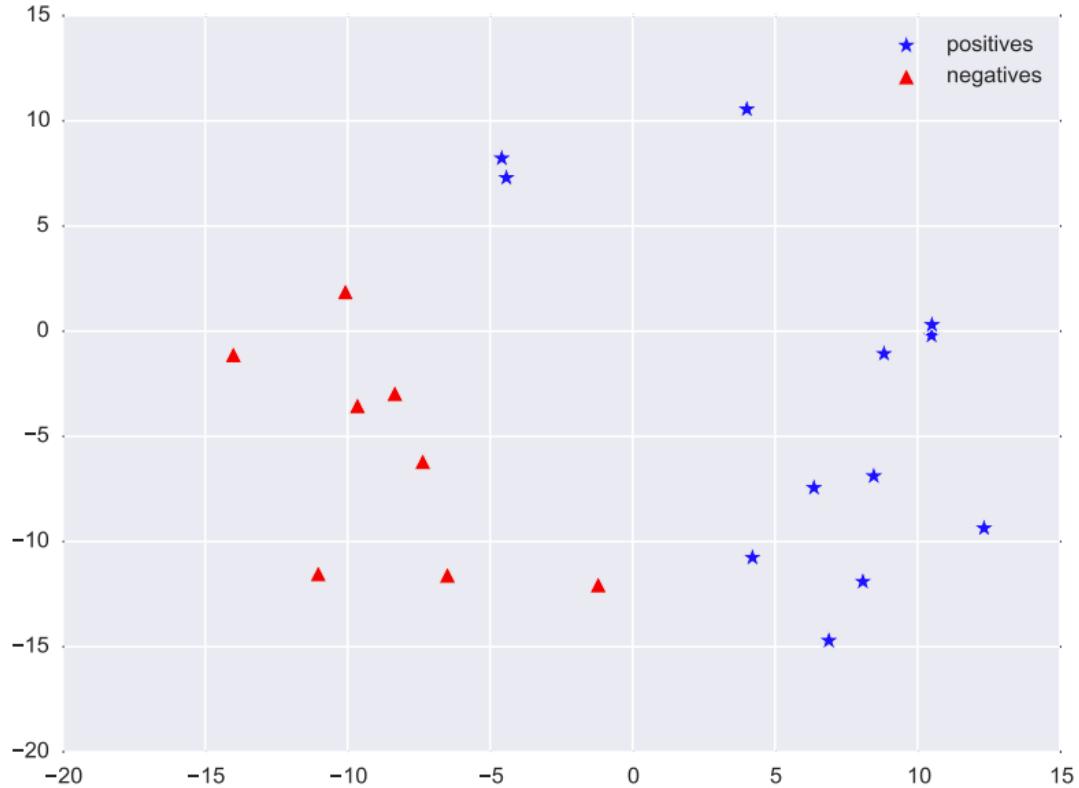
Soft-margin linear SVM

$$\begin{aligned} & \underset{\mathbf{w}, b, \xi}{\text{minimize}} && \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^N \xi_i \\ & \text{subject to} && y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1 - \xi_i \\ & && \xi_i \geq 0 \end{aligned}$$

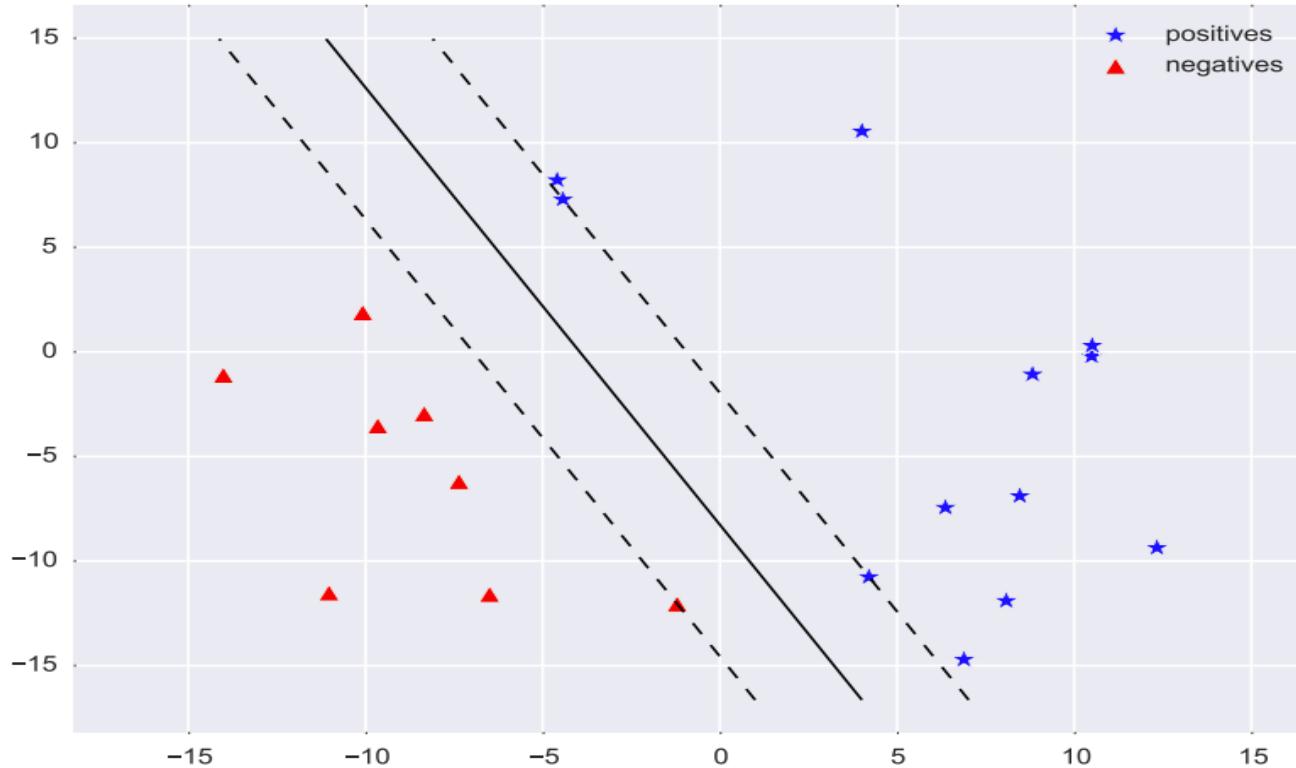
C is a **regularization** hyperparameter:

- Large C : high penalty for violation constraints \rightarrow narrower margin.
- Small C : low penalty for violating constraints \rightarrow larger margin.
- $C \rightarrow \infty$ produces hard-margin SVM.

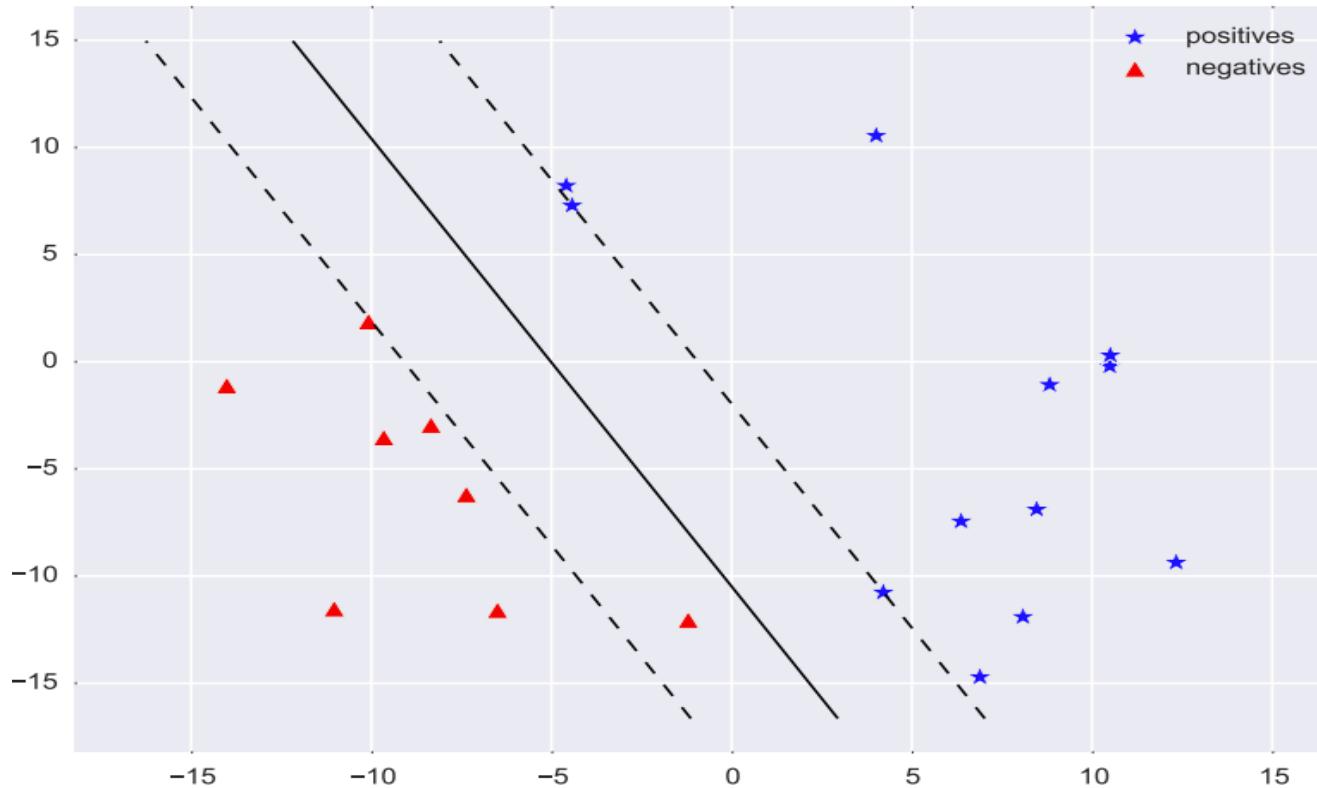
Examples: a separable dataset



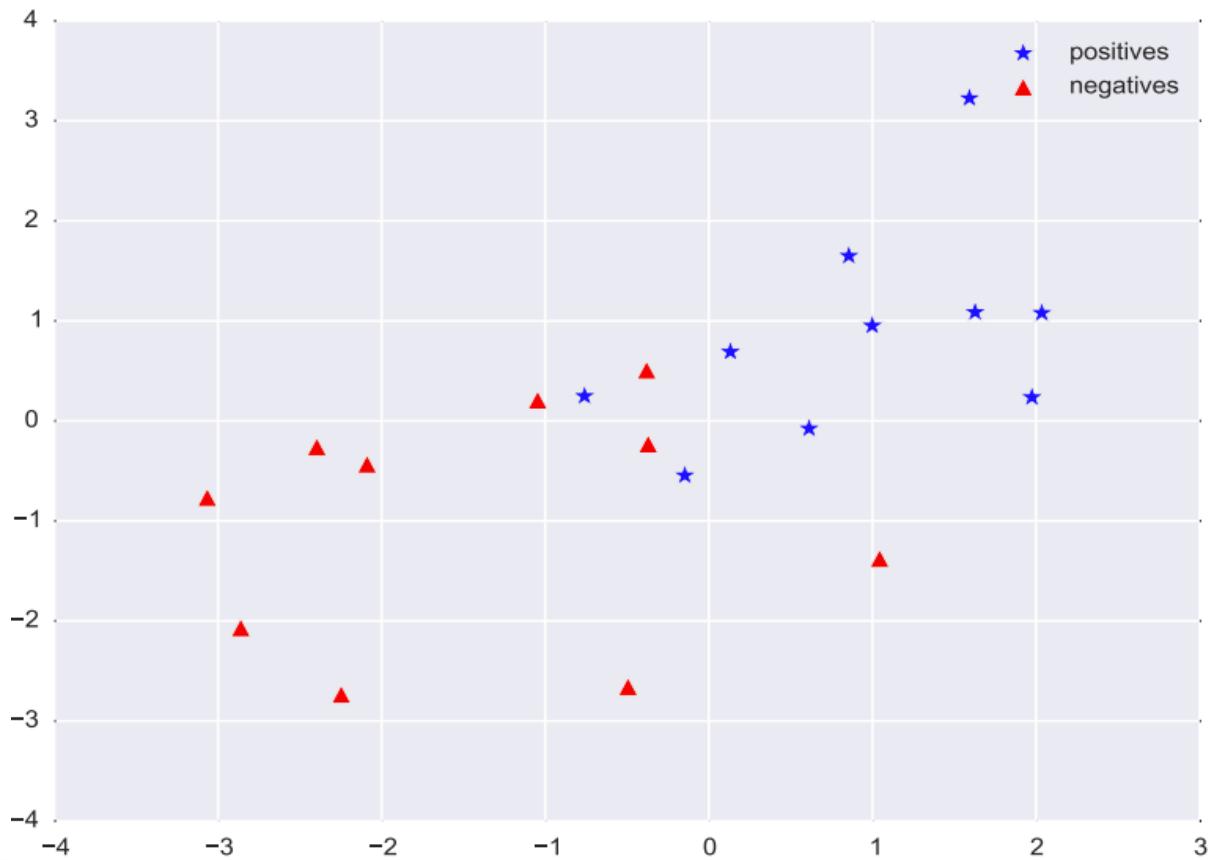
Examples: soft-margin SVM with $c = 1$



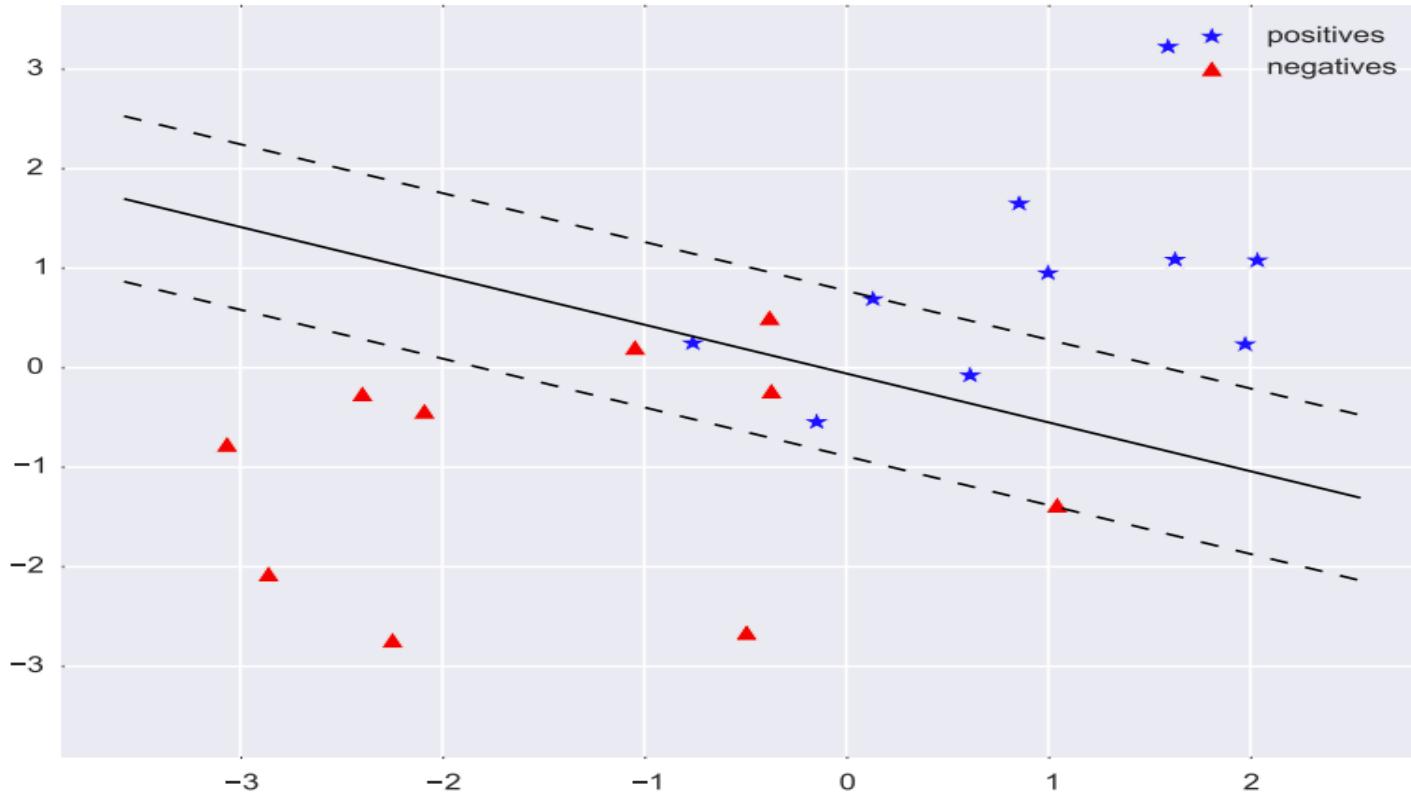
Examples: soft-margin SVM with $c = 0.01$



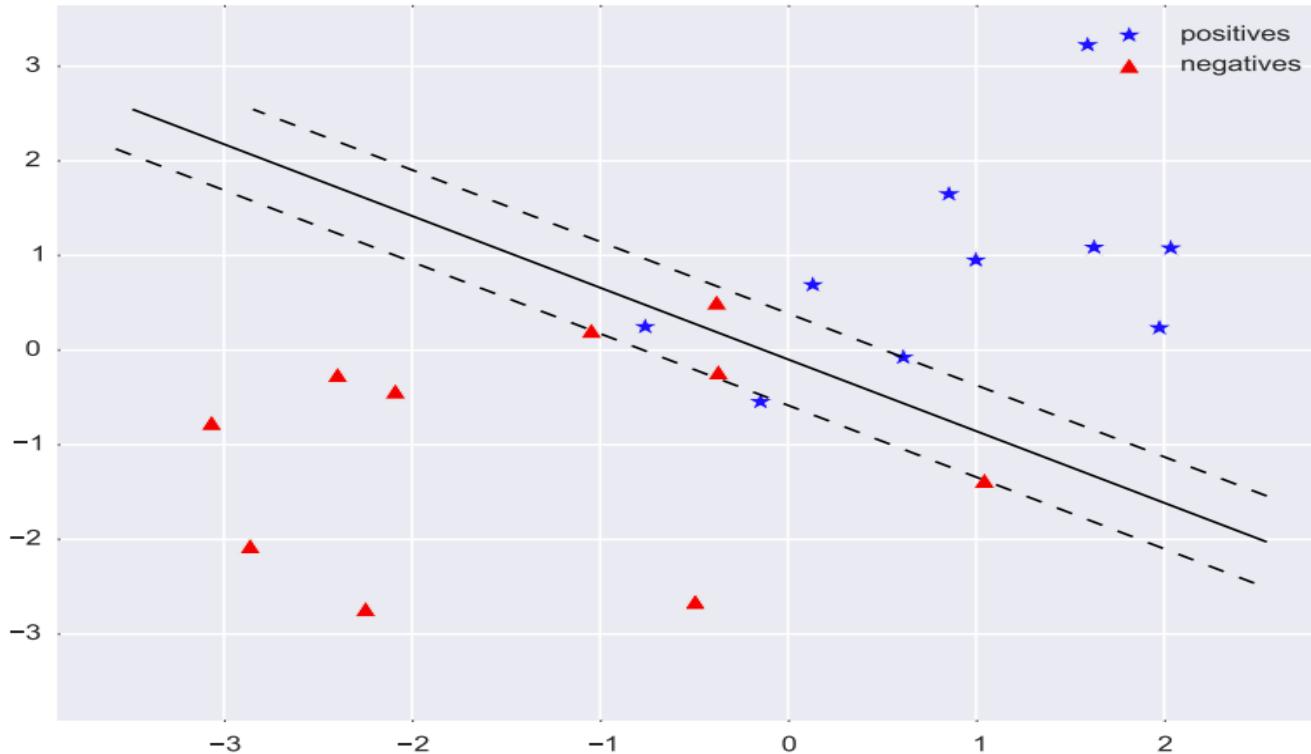
Examples: a non-separable dataset



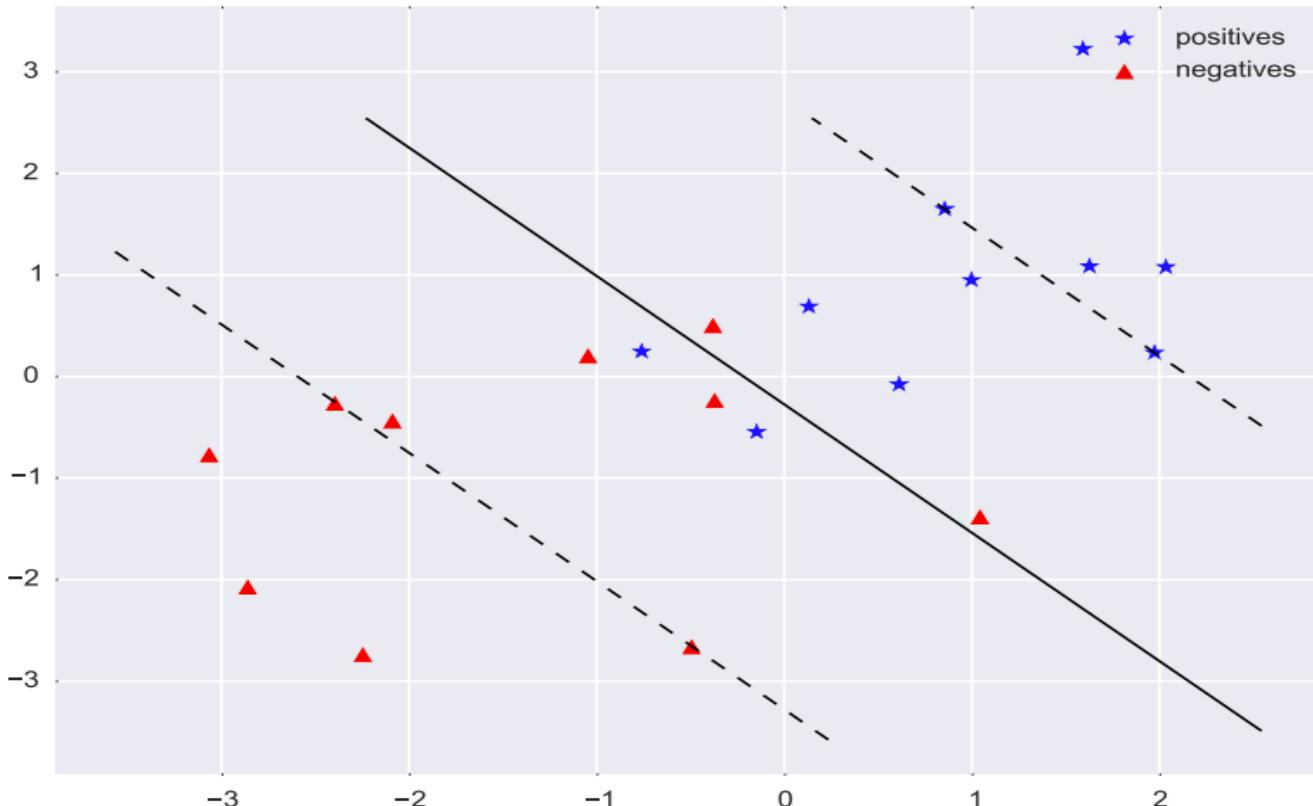
Examples: soft-margin SVM with $c = 1$



Examples: soft-margin SVM with $c = 100$



Examples: soft-margin SVM with $c = 0.1$



Soft-margin linear SVM in Python

- Available in scikit-learn⁴: sklearn.svm.LinearSVC
- Wrapper around LIBLINEAR⁵ (very fast).
- Multi-class support: one-vs-rest (OVR)

```
from sklearn import svm

clf = svm.LinearSVC(C=1.0)
clf.fit(X_train, y_train)

# evaluate accuracy on val set
print(clf.score(X_val, y_val))

# predict on test set
y_pred = clf.predict(X_test)
```

⁴<http://scikit-learn.org>

⁵<http://www.csie.ntu.edu.tw/~cjlin/liblinear/>

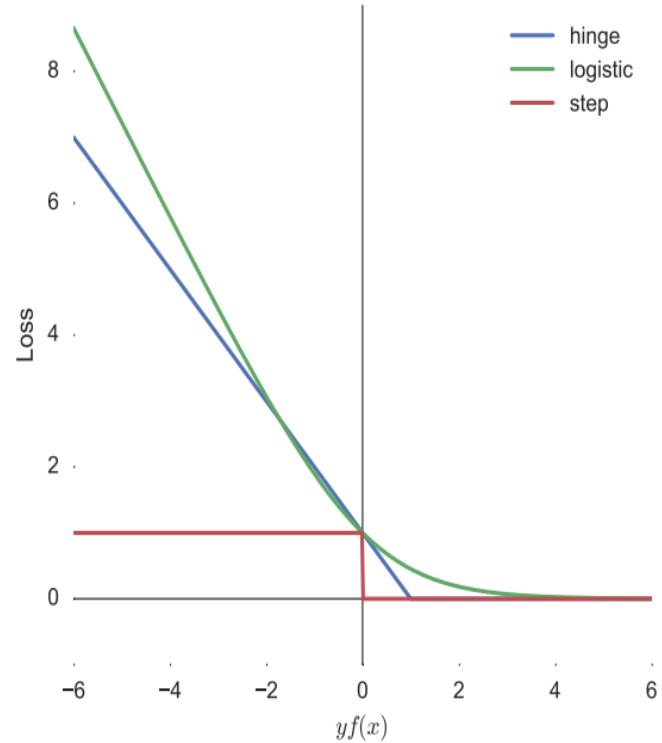
Comparison of linear SVM and logistic regression

Linear SVM uses the hinge loss:

- No penalty for examples on correct side of margin
- Linear penalty for examples that violate margin
- Robust to outliers
- Must use L_2 to maximize margin.

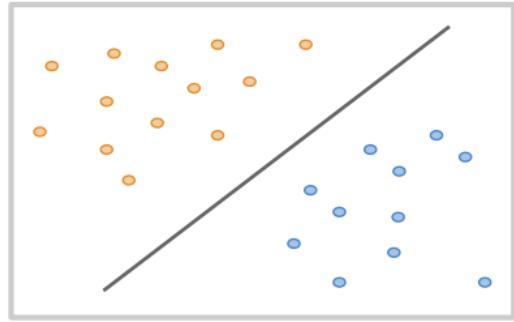
Logistic regression uses logistic loss:

- Penalty for correct examples
- Less robust than hinge to outliers
- Can use L_2 but not necessary
- Naturally produces probabilities

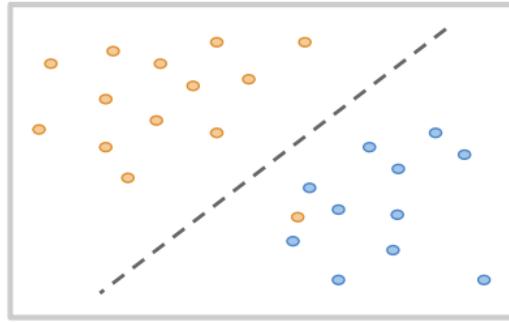


Kernel SVMs

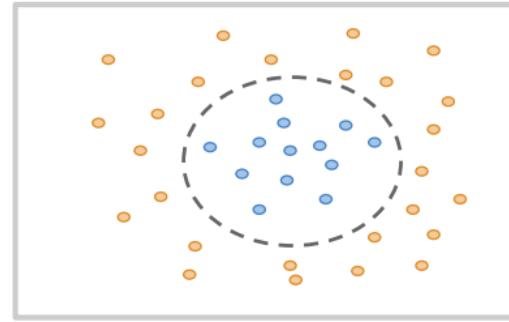
Three problems



Linearly separable



Almost linearly separable



Not linearly separable

Mapping functions

Linear SVM can only produce linear decision boundaries.

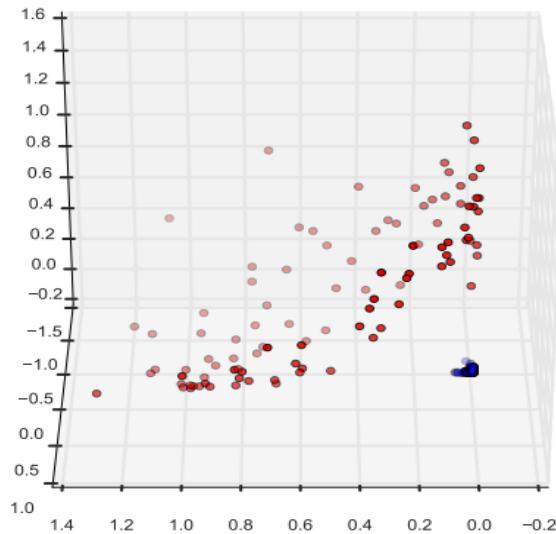
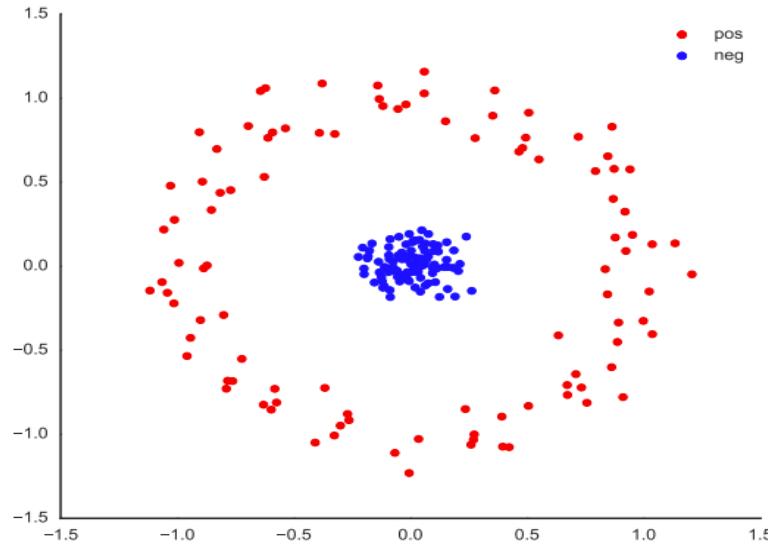
Many interesting classification problems are **not** linearly separable.

Although data may not be linearly separable in the original feature space, it may be separable in another feature space that we can compute from the original feature space via a mapping function $\phi(\mathbf{x})$.

For example, for $\mathbf{x} \in \mathbb{R}^2$, we could define $\phi : \mathbb{R}^2 \rightarrow \mathbb{R}^3$ as:

$$\phi(\mathbf{x}) = [x_1^2 \quad \sqrt{2}x_1x_2 \quad x_2^2]$$

Mapping functions



$$\phi(\mathbf{x}) = [x_1^2 \quad \sqrt{2}x_1x_2 \quad x_2^2]^T$$

Mapping functions

The decision boundary, while linear in the new space, corresponds to a non-linear boundary when back projected to the original space.

These mapping functions are sometimes called *feature extraction algorithms*.

Many non-linear ML problems can be solved using a good feature extraction algorithm.

But...

- Designing good feature extractors is hard!
- The more features we use, the more parameters we need to fit.
- Large number of features means we need a lot of data to avoid overfitting.

Kernel SVMs

Kernel SVMs offer a way to avoid needing to define an explicit feature mapping function.

Instead of the mapping function, we need to define a *kernel function* $K(\mathbf{x}, \mathbf{z})$ that compares **how similar** the examples \mathbf{x} and \mathbf{z} are.

This formulation allows us to implicitly work in very high dimensional feature spaces without ever actually computing the explicit projections $\varphi(\mathbf{x})$.

To do this, we'll need to reformulate the SVM objective function (loss) and decision function so that they only use inner products between training examples $\mathbf{x}_i^T \mathbf{x}_j$. We can then replace these inner products with kernel evaluations.

Kernels

Kernel design intuition:

- $K(\mathbf{x}, \mathbf{z})$ should be large for similar values;
- $K(\mathbf{x}, \mathbf{z})$ should be small for different values.

Common kernels:

Linear	$K(\mathbf{x}, \mathbf{z}) = \mathbf{x}^T \mathbf{z}$	Same as linear SVM
Polynomial	$K(\mathbf{x}, \mathbf{z}) = (1 + \mathbf{x}^T \mathbf{z})^d$	All polynomials up to degree d
Gaussian	$K(\mathbf{x}, \mathbf{z}) = \exp(-\gamma \parallel \mathbf{x} - \mathbf{z} \parallel^2)$	Infinite dimensional feature spaces.

- Gaussian kernel also known as Radial Basis Function (**RBF**)

So, what has the kernel trick gotten us?

- No need to define feature mapping function, only the kernel.
- Number of parameters doesn't depend on dimension of features.
- Reduce overfitting (number of parameters \leq number of data points).
- Can work in very high dim (even infinite) feature spaces.

But...

- We still need to design appropriate kernel functions. This is difficult.
- It is slower to solve the kernel SVM than the linear SVM when we have many data points (big N).
- Sometimes it's more efficient to do an explicit feature mapping and solve the linear SVM.

Kernel SVM in Python

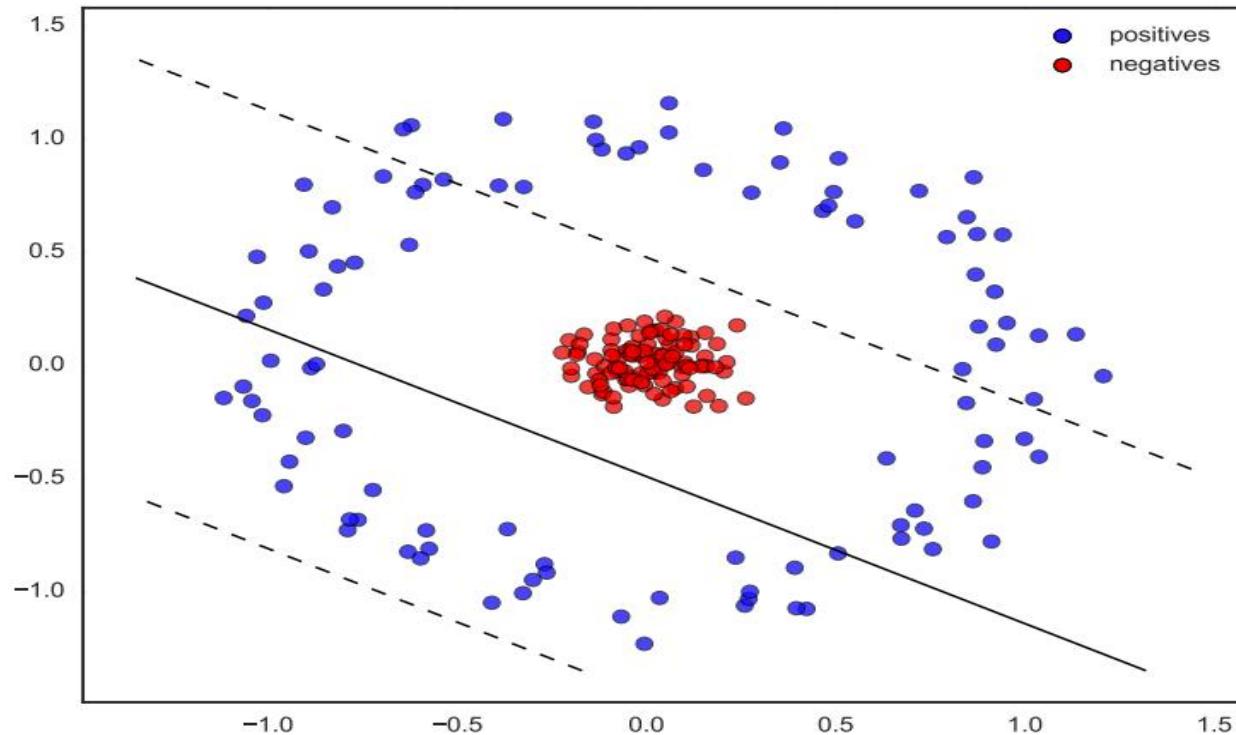
Can use `sklearn.svm.SVC`

```
import numpy as np
from sklearn import svm

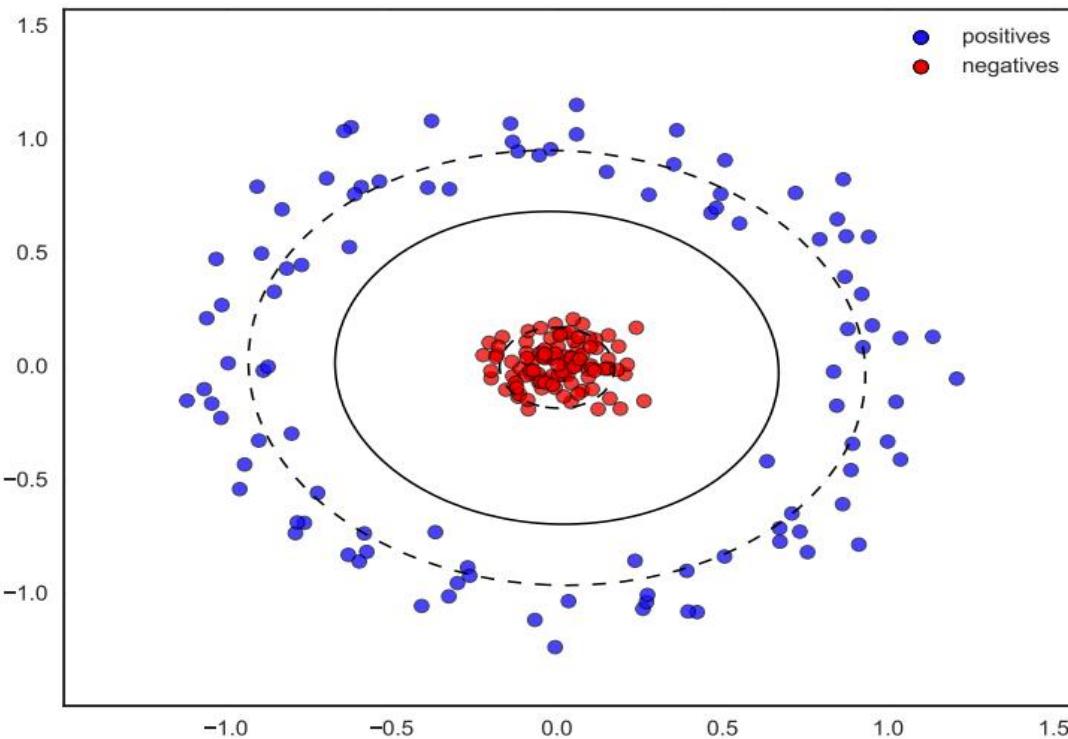
# generate some data
X= np.random.randn(300, 2)
Y= np.logical_xor(X[:, 0] > 0, X[:, 1] > 0)

# fit the model
clf = svm.SVC(kernel='rbf')
clf.fit(X, Y)
```

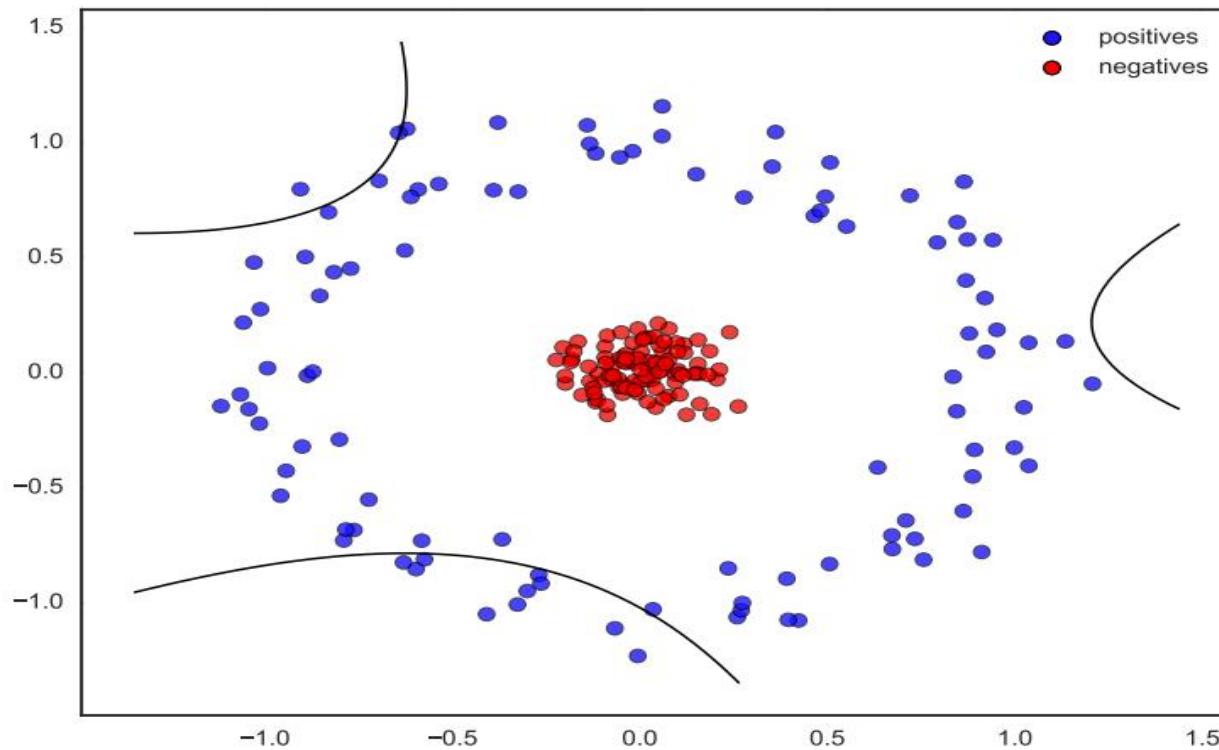
Example:Soft margin linear SVM with C=1.0



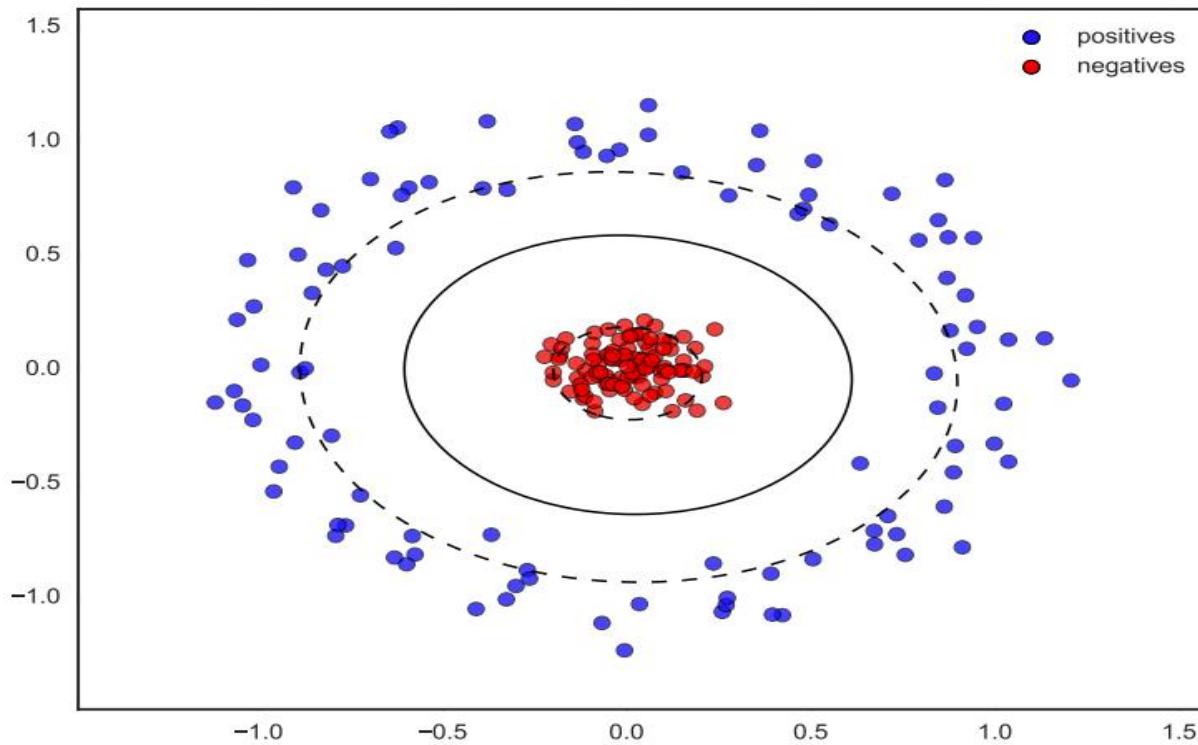
Example: SVM ($C=1$) with degree-2 polynomial kernel



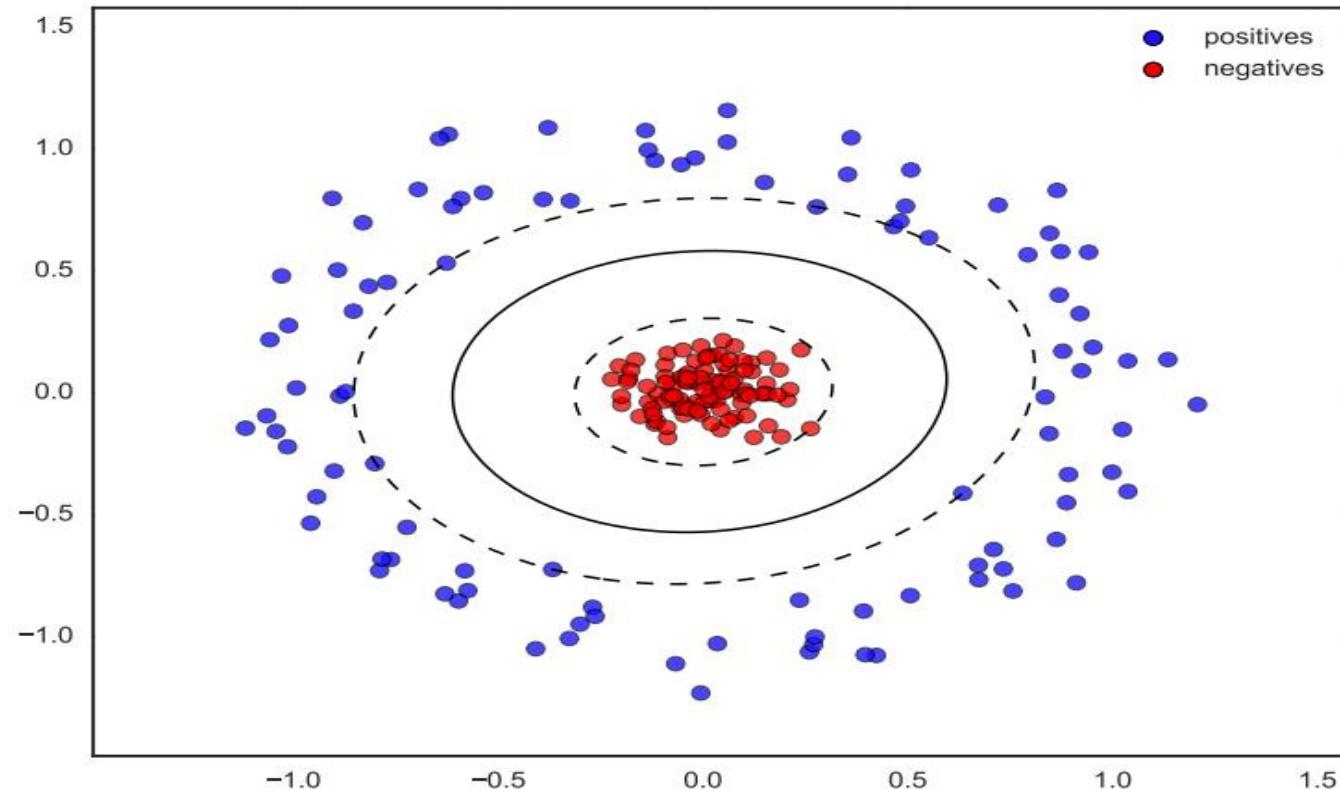
Example: SVM (C=1) with degree-3 polynomial kernel



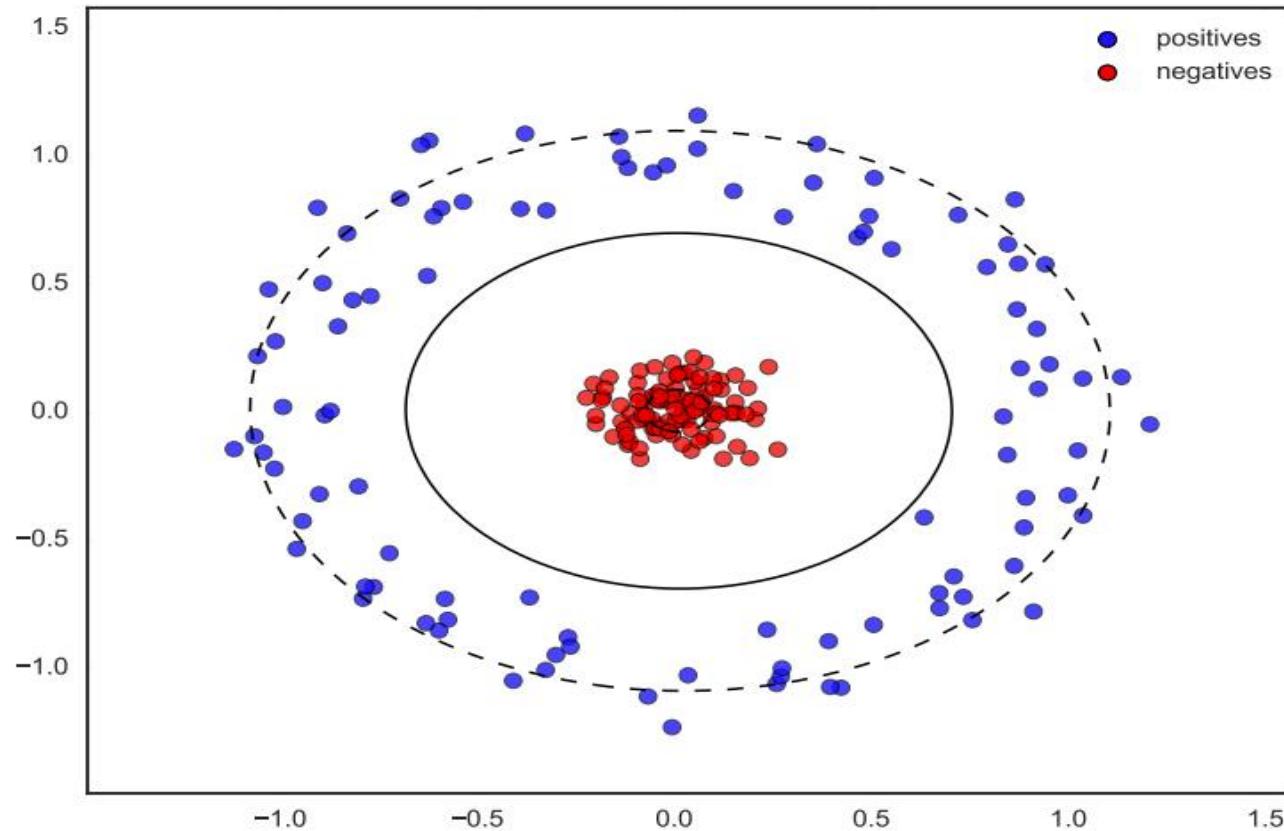
Example: RBF SVM with $C = 1$



Example: RBF SVM with $C = 100$



Example: RBF SVM with $C = 0.1$



Tips for use

Scaling data

The SVM is not scale invariant! Good idea to try **normalizing** your data. Common strategies:

- standardize data to have $\mu = 0$ and $\sigma = 1$.
- min-max scale from to range $[0, 1]$ or $[-1, +1]$

Fit normalization parameters (e.g. min and max or $\mu = 0$ and $\sigma = 1$) **on the training data**. Apply same normalization to validation and test data.

The support vectors

- The support vectors are the training vectors \mathbf{x}_i for which the corresponding α_i is non-zero.
- You do not need to keep the entire training set to predict. Only the support vectors.
- The number of support vectors is typically a small fraction of the training set (only the values that touch/violate the margin).
- This limits the effective number of parameters (α_i), which helps prevent overfitting.

Kernels and hyperparameters

- Kernel SVMs:
 - typically work well when you have $O(10,000)$ or fewer training examples.
 - 1M+ training examples makes training very slow.
 - kernel needs to be evaluated at least once for all pairs of training examples $O(N^2)$.
 - training a kernel SVM has approx $O(N^2D)$ complexity (SMO algorithm).
 - the RBF kernel is a good default for many tasks.
- Hyperparameters (C, γ) :
 - try grid search or *random search* on an exponential scale (log space, e.g. $C = \{0.01, 0.1, 1.0, 10, 100\}$).
 - optimize on a **validation** set, or via **cross-validation**. Never on the test set!

Linear SVMs

- Try first as a baseline!
- Much faster to train, approx $O(ND)$ for LIBLINEAR.
- Can be trained out-of-core using SGD (Pegasos) on very large datasets.
- Very fast at predict time $\mathbf{w}^T \mathbf{x} + b$.
- Often faster to explicitly compute a feature map $\varphi(\mathbf{x})$ and use linear SVM. Also possible to approximate kernels with explicit feature maps.
- Possible to use with deep learning and backprop.

Further reading

- Hastie et al., The Elements of Statistical Learning, Chapter 12
 - http://statweb.stanford.edu/~tibs/ElemStatLearn/printings/ESLII_print10.pdf
- Andrew Ng's Lecture Notes
 - <http://cs229.stanford.edu/notes/cs229-notes3.pdf>
- Bishop, Pattern Recognition and Machine Learning, Chapter 6, 7