CentraleSupélec

Machine Learning

2EL1730

Lecture 1

Part I: Introduction; Part II: Model selection and evaluation

Fragkiskos Malliaros and Maria Vakalopoulou

Tuesday, November 26, 2019

About Me

- Undergrad at the University of Patras, Greece
- Ph.D. in CS at Ecole Polytechnique, Paris
- Postdoc researcher at UC San Diego
- Assistant Professor at CentraleSupélec (since Oct. 2017)

Research interests: Data science, ML, graph mining, text mining and NLP

The Team

Fragkiskos Malliaros

Office: Bâtiment Bouygues, CVN Lab, Room SC.217

Email: fragkiskos.malliaros@centralesupelec.fr

Maria Vakalopoulou

Office: Bâtiment Bouygues, MICS Lab, Room SB.132

Email: maria.vakalopoulou@centralesupelec.fr

Office hours: we will be available right after the lecture

Or, send us an email and we will find a good time to meet

The Team (2/2)

- Yunshi Huang (PhD student, CVN lab)
- Yoann Pradat (PhD student, MICS lab)
- Mohamed El Amine Seddik (PhD student, MICS lab)
- Jun Zhu (PhD student, MICS lab)

Acknowledgements

- The lecture is partially based on material by
  - Richard Zemel, Raquel Urtasun and Sanja Fidler (University of Toronto)
  - Chloé-Agathe Azencott (Mines ParisTech)
  - Julian McAuley (UC San Diego)
  - Dimitris Papailiopoulos (UW-Madison)
  - Jure Leskovec, Anand Rajaraman, Jeff Ullman (Stanford Univ.)
    - http://www.mmds.org
  - Panagiotis Tsaparas (UOI)
  - Evimaria Terzi (Boston University)

Thank you!

Slides of Today's Lecture

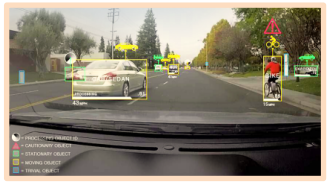
http://fragkiskos.me/introduction.pdf

Roadmap

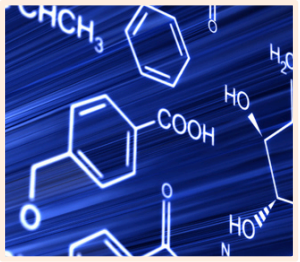
- Motivation
- Basic pipeline in Machine Learning
- *(Course logistics)*
- Overview of basic tasks in Machine Learning
- Empirical Risk Minimization
  - Loss function
  - Basic concepts in optimization
- Model selection
  - Overfitting and generalization
- Bias-variance tradeoff
- Training, validation and test sets
  - Cross-validation
- Evaluation of supervised learning algorithms

Why
Machine Learning?

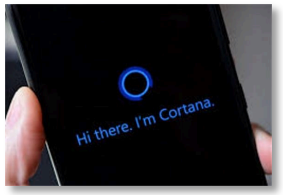
Automation and Robotics



Drug Discovery and Healthcare



Intelligent Personal Assistants

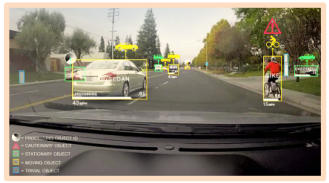


Recommender Systems

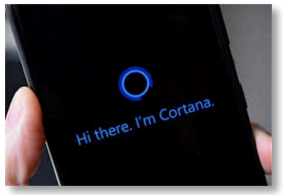
amazon

NETFLIX

Automation and Robotics



Real and important problems

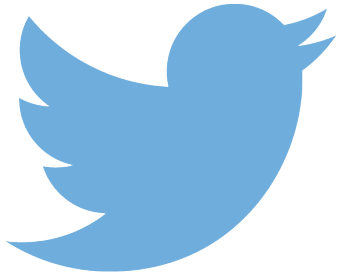


Recommender Systems

amazon

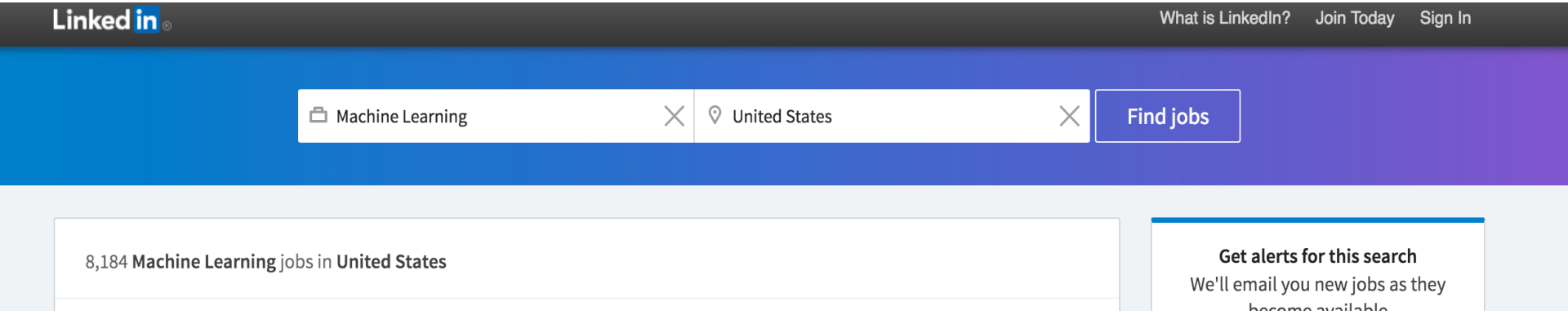
NETFLIX

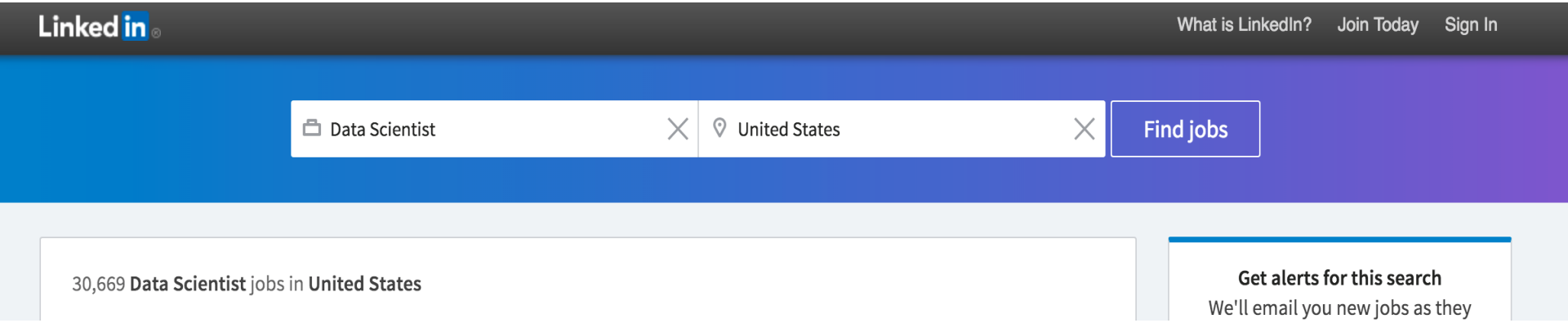

facebook

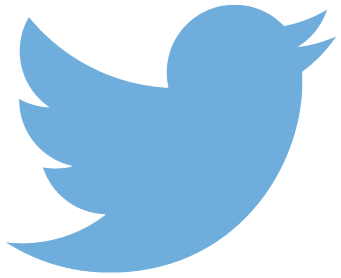



amazon


IBM


Linked in
What is LinkedIn?
Join Today
Sign In
Machine Learning
United States
Find jobs
8,184 Machine Learning jobs in United States
Get alerts for this search
We'll email you new jobs as they

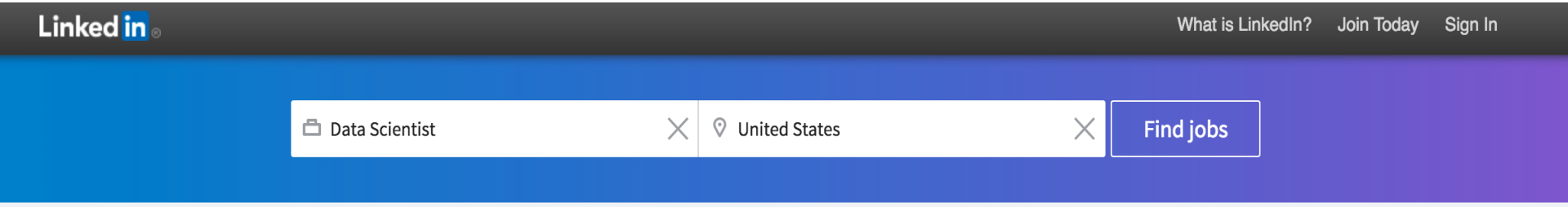

Linked in
What is LinkedIn?
Join Today
Sign In
Data Scientist
United States
Find jobs
30,669 Data Scientist jobs in United States
Get alerts for this search
We'll email you new jobs as they


Job market

8,184 **Machine Learning** jobs in **United States**

Get alerts for this search
We'll email you new jobs as they become available



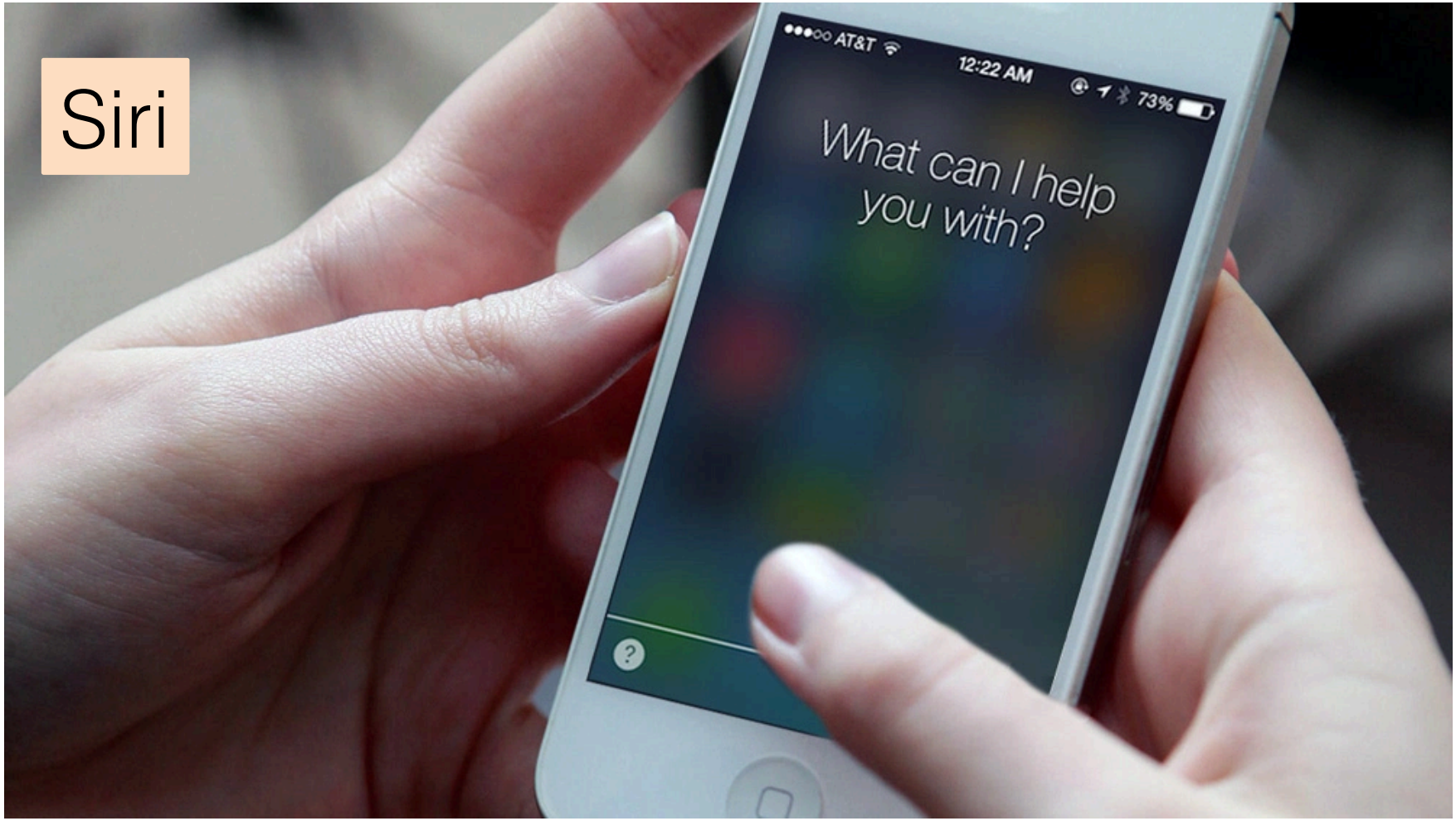
30,669 **Data Scientist** jobs in **United States**

Get alerts for this search
We'll email you new jobs as they

Machine Learning is Almost Everywhere

- Recognizing patterns: Speech Recognition, facial identity, etc

Speech Recognition

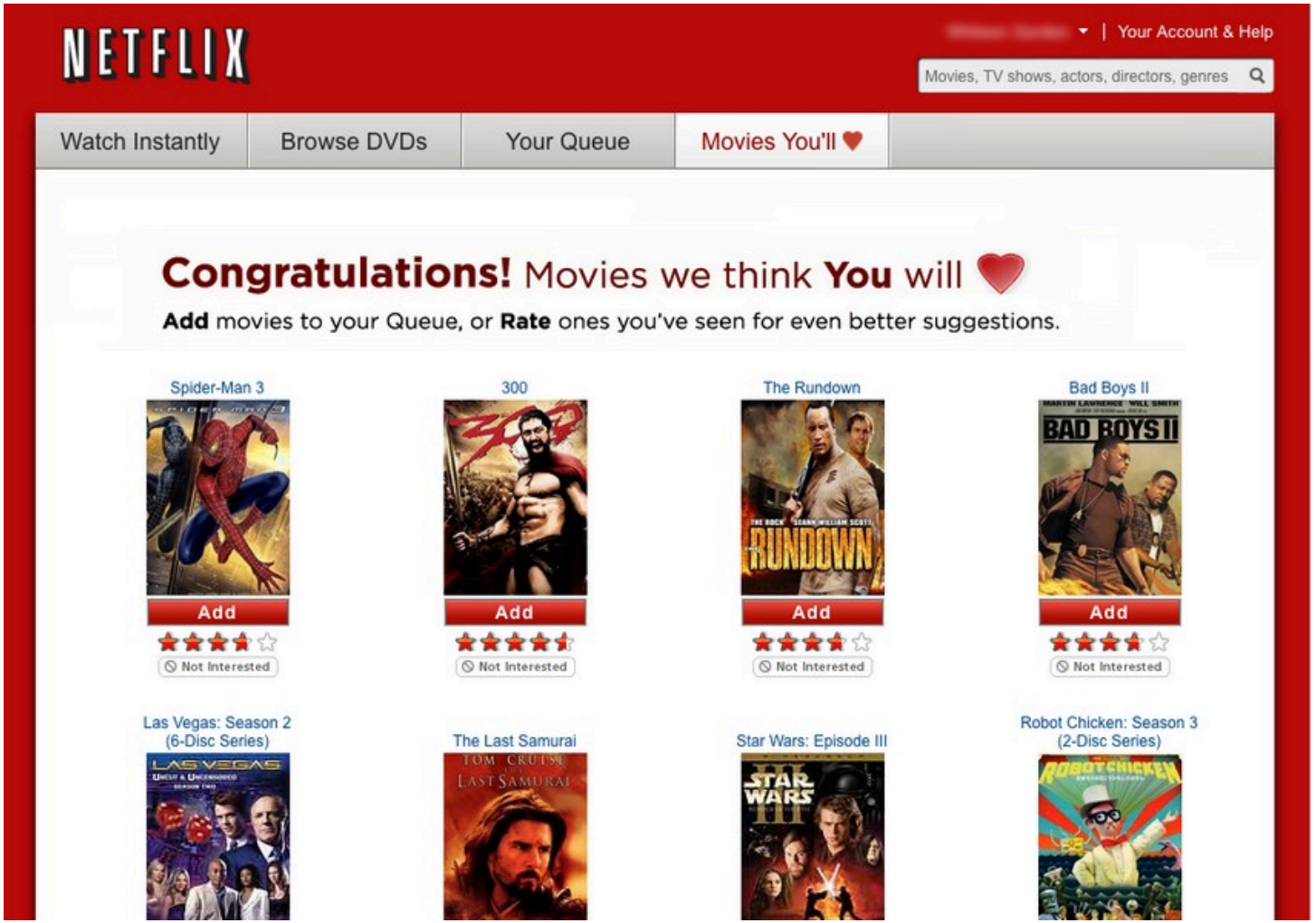


Source: http://bgr.com/2017/01/24/iphone-8-enhanced-siri-upgrade/

Machine Learning is Almost Everywhere

- Recognizing patterns: Speech Recognition, facial identity, etc
- Recommender Systems: Noisy data, commercial pay-off (e.g., Amazon, Netflix)

Example of a Recommendation System



Source: http://www.aboutdm.com/2012/12/how-netflix-makes-recommendation-for.html

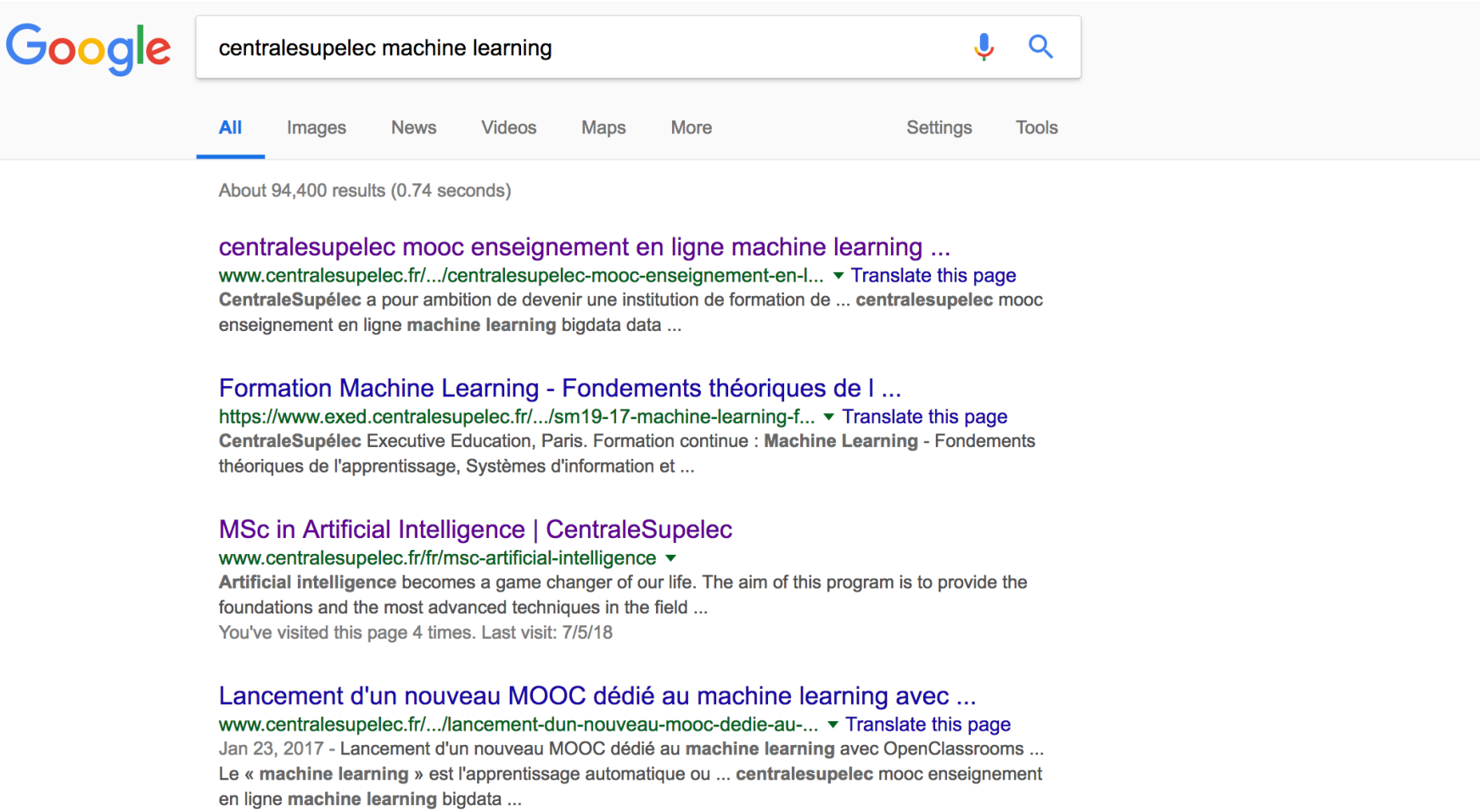
Collaborative Filtering

Source: Wikipedia

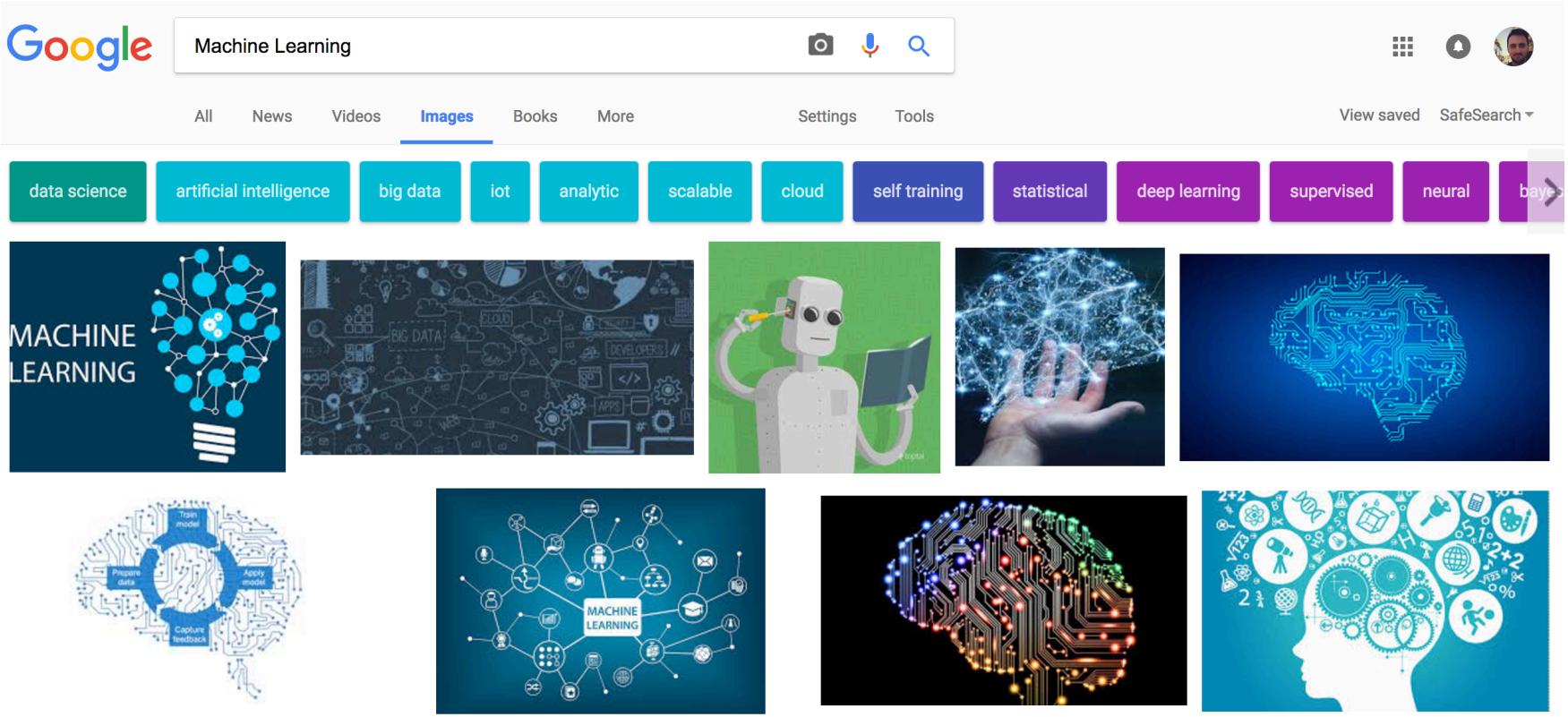
Machine Learning is Almost Everywhere

- Recognizing patterns: Speech Recognition, facial identity, etc
- Recommender Systems: Noisy data, commercial pay-off (e.g., Amazon, Netflix)
- Information retrieval: Find documents or images with similar content

Information Retrieval (1/2)



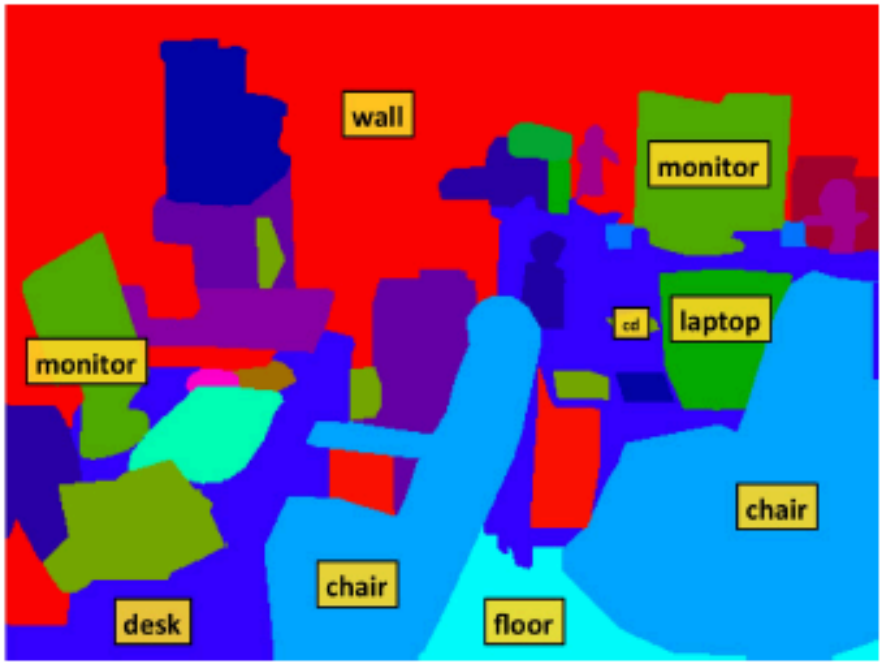
Information Retrieval (2/2)



Machine Learning is Almost Everywhere

- Recognizing patterns: Speech Recognition, facial identity, etc
- Recommender Systems: Noisy data, commercial pay-off (e.g., Amazon, Netflix)
- Information retrieval: Find documents or images with similar content
- Computer vision: detection, segmentation, depth estimation, optical flow, etc.

Computer Vision (1/3)



Source: R. Urtusan, U. of Toronto

Computer Vision (2/3)

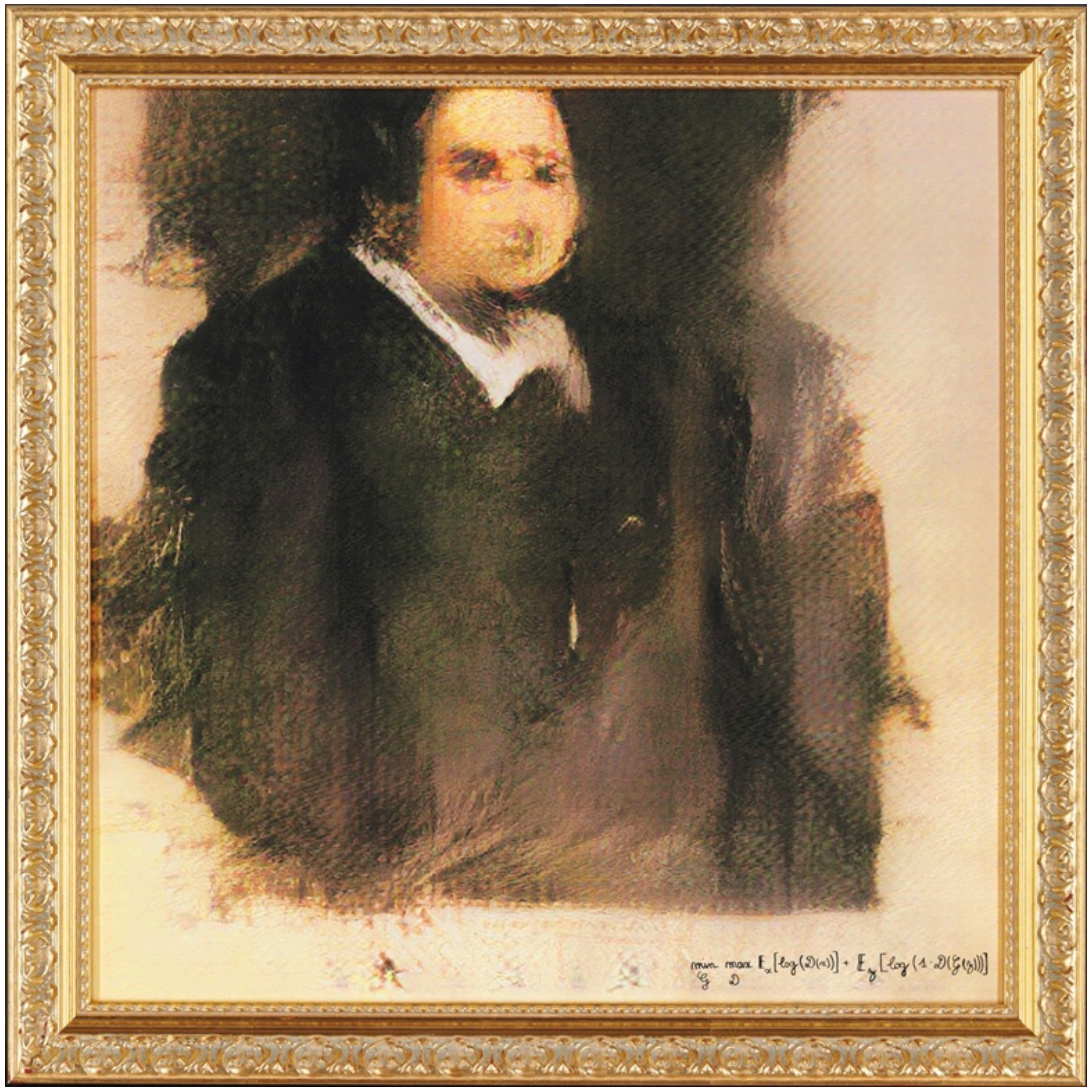
Source: https://techcrunch.com/2016/11/13/wtf-is-computer-vision/

Computer Vision (3/3)

[Gatys, Ecker, Bethge. A Neural Algorithm of Artistic Style. Arxiv'15.]

Source: R. Urtusan, U. of Toronto

AI Artwork Sells for $432,500 (Christie's)



Source: https://www.christies.com/features/A-collaboration-between-two-artists-one-human-one-a-machine-9332-1.aspx

Machine Learning is Almost Everywhere

- Recognizing patterns: Speech Recognition, facial identity, etc
- Recommender Systems: Noisy data, commercial pay-off (e.g., Amazon, Netflix)
- Information retrieval: Find documents or images with similar content
- Computer vision: detection, segmentation, depth estimation, optical flow, etc.
- Robotics: perception, planning, autonomous driving, etc.

Autonomous Driving

Source: https://www.tesla.com/videos/autopilot-self-driving-hardware-neighborhood-long
https://www.youtube.com/watch?v=hLaEV72elj0

Machine Learning is Almost Everywhere

- Recognizing patterns: Speech Recognition, facial identity, etc
- Recommender Systems: Noisy data, commercial pay-off (e.g., Amazon, Netflix)
- Information retrieval: Find documents or images with similar content
- Computer vision: detection, segmentation, depth estimation, optical flow, etc.
- Robotics: perception, planning, autonomous driving etc
- Learning to play games

AlphaGo

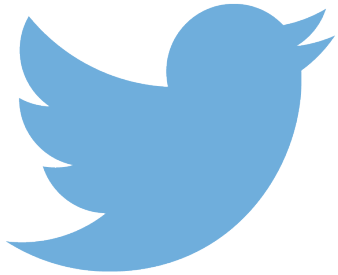
Source: https://www.newscientist.com

Also, large-scale. Why?

Content Generation

More than 700 photos uploaded / second

More than 7k tweets / second



More than 55K google searches / second

More than 2.5 million emails sent / second

Source: http://www.internetlivestats.com/one-second/

Content Generation

More than 700 photos uploaded / second

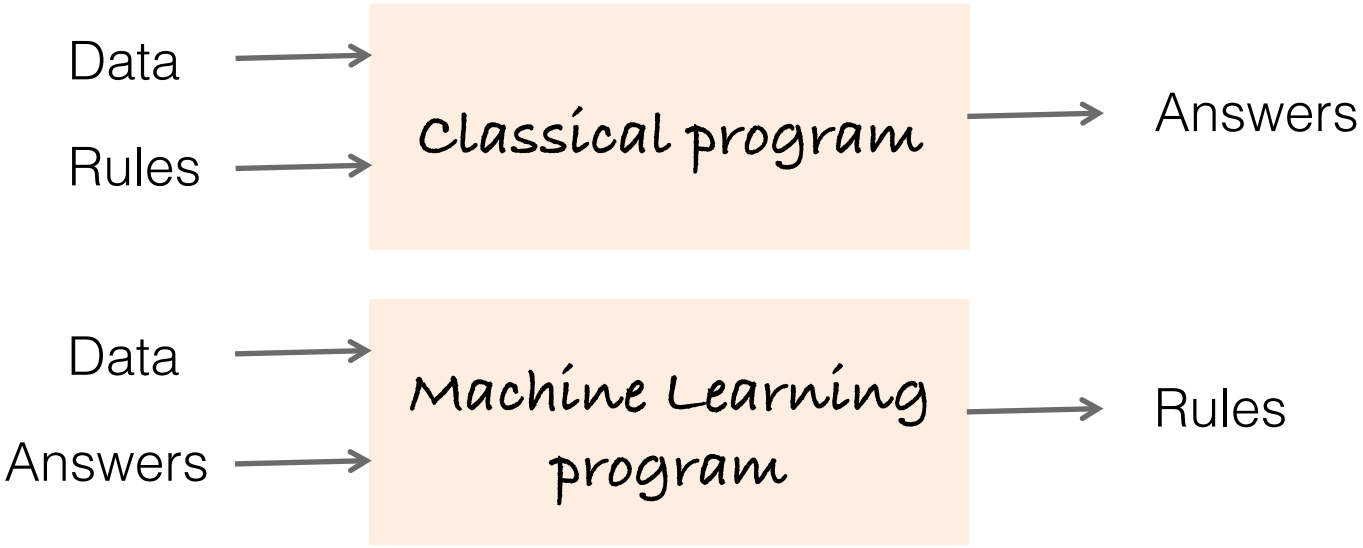
We have to handle large data sets

More than 2.5 million emails sent / second

Source: http://www.internetlivestats.com/one-second/

Why Learning?

- There is no need to “learn” to calculate our taxes
- Learning is used when
  - Human expertise does not exist (e.g., bioinformatics)
  - Humans are unable to explain their expertise (speech recognition, computer vision)
  - Complex solutions change in time (routing computer networks)



What is Machine Learning?

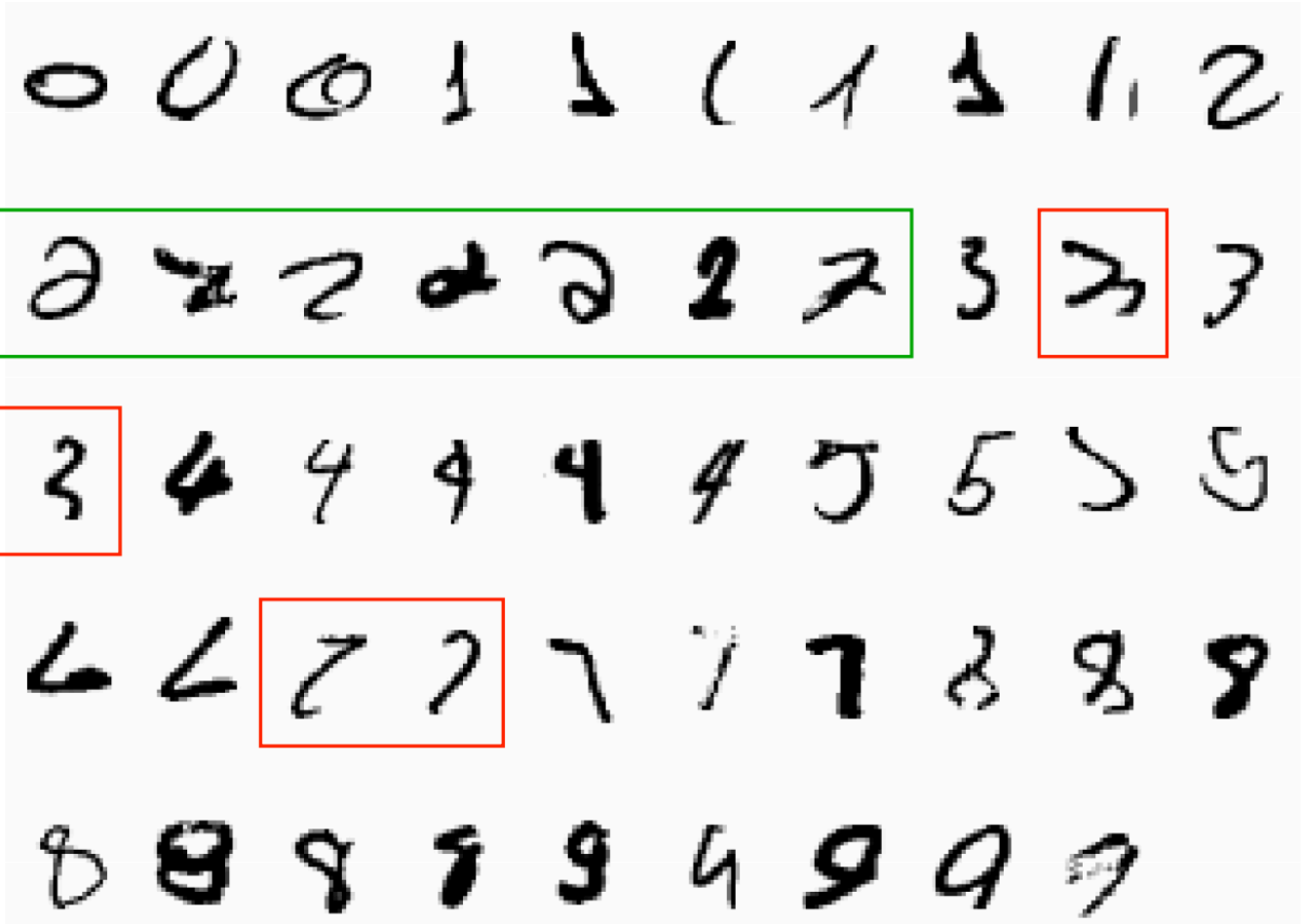
- Learning systems are not directly programmed to solve a problem, instead develop their own program based on:
  - Examples of how they should behave
  - From trial-and-error experience trying to solve the problem

- Different than standard CS programs
  - Want to implement unknown function, only having access to sample input-output pairs (training examples)

- Learning simply means incorporating information from the training examples into the system

Task that Requires ML

What makes a ‘2’?



Learning Objectives of Today's Class

- Given a problem
  - Decide weather it can be solved with machine learning
  - Decide as what type of machine learning problem you can formalize it (unsupervised – clustering, dimensionality reduction, supervised – classification, regression?)
  - Describe it formally in terms of design matrix (observations x features), features, samples
- Define a loss function
- Model selection and evaluation in ML

More on that will follow soon

Example of ML Pipeline

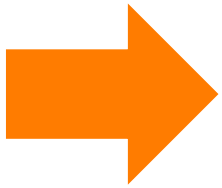
- Running example: image classification
- Goal: “train a computer to recognize a cat from a dog”
- How?

Example of ML Pipeline

- Running example: image classification
- Goal: “train a computer to recognize a cat from a dog”
- How?

Simple idea, inspired by inductive human learning

Show it a lot of labeled examples



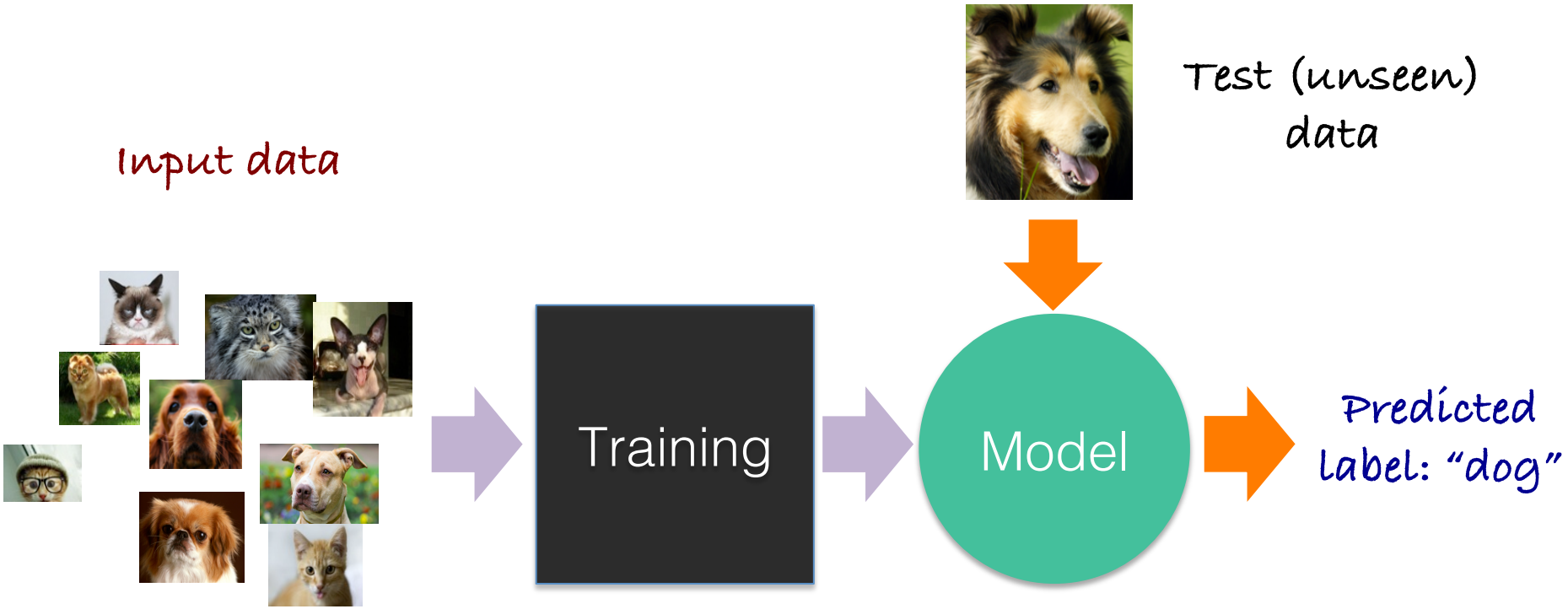
“predicts” the right label on unseen data

From This

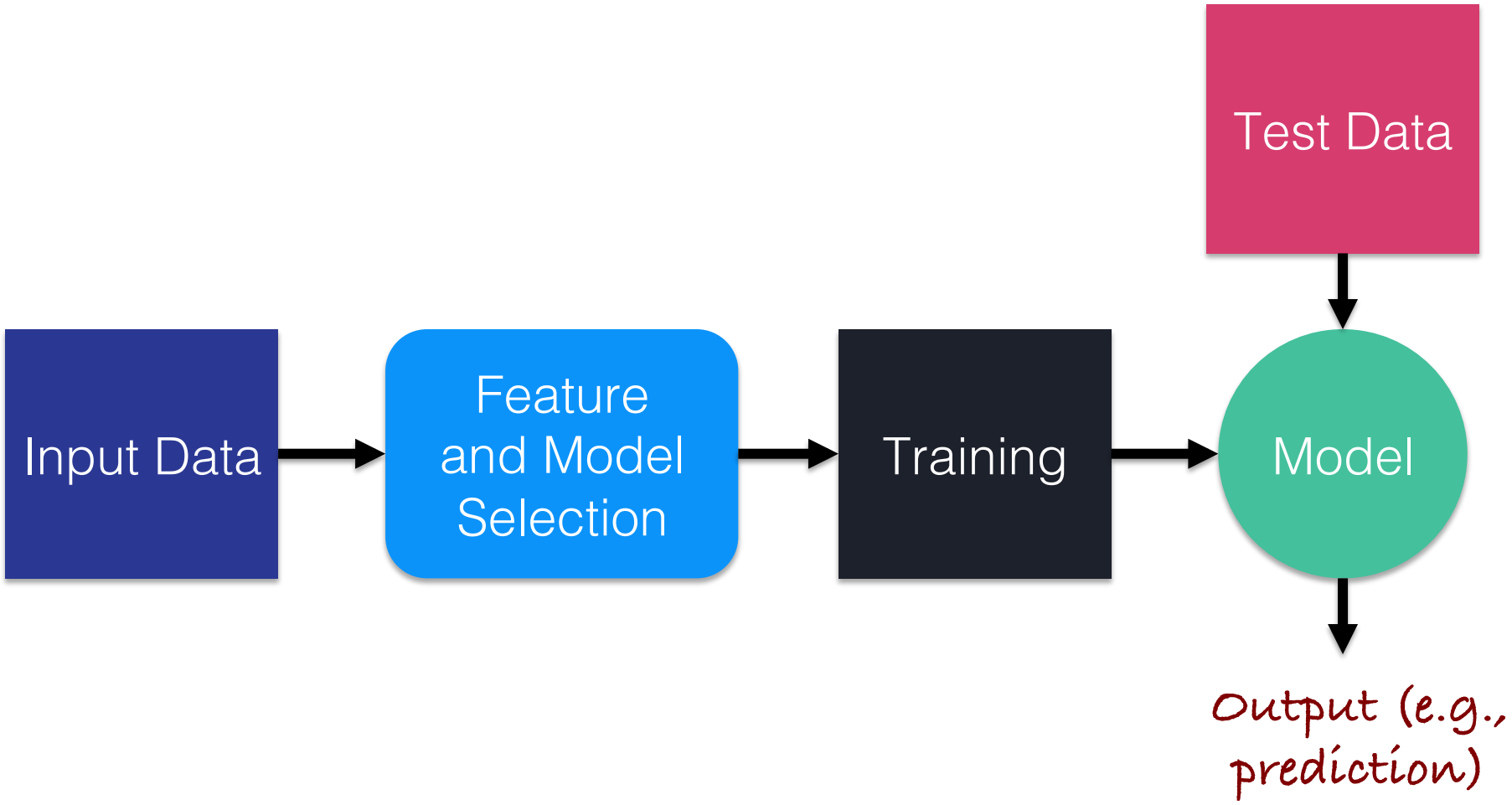
To this

Example of ML Pipeline

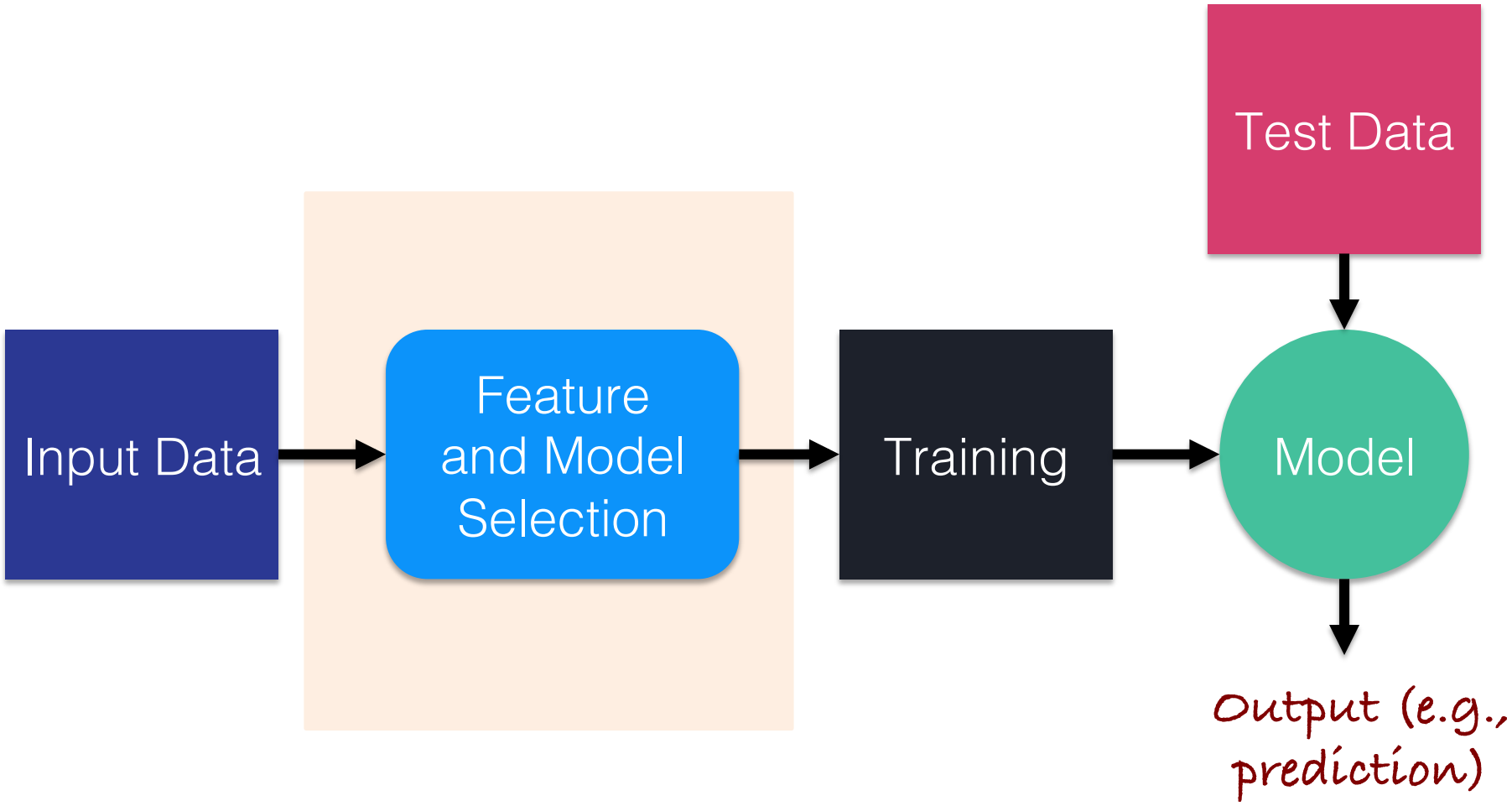
- Running example: image classification
- Goal: “train a computer to recognize a cat from a dog”
- How?



ML Pipeline


Test Data
Input Data
Feature and Model Selection
Training
Model
Output (e.g., prediction)

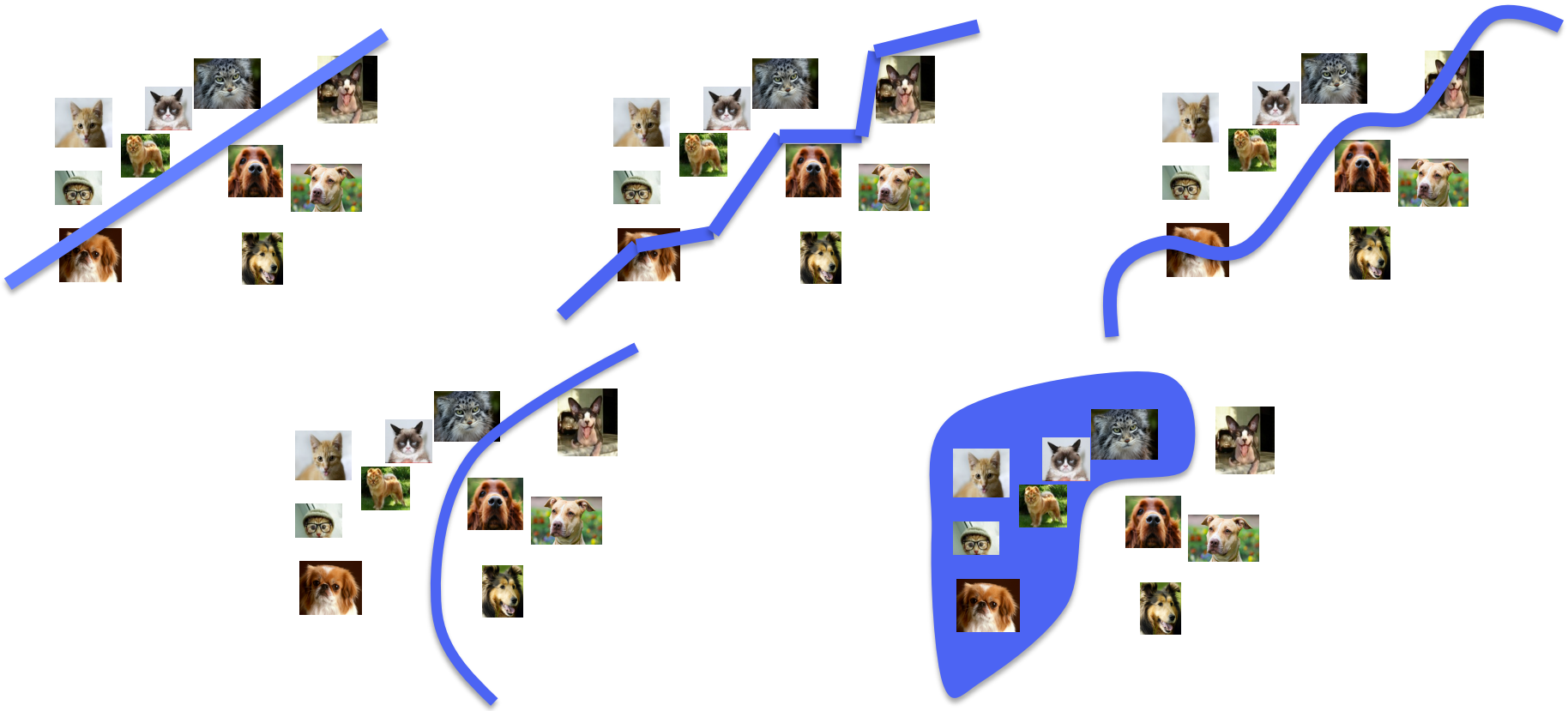
ML Pipeline


Test Data
Input Data
Feature and Model Selection
Training
Model
Output (e.g., prediction)

Feature Selection

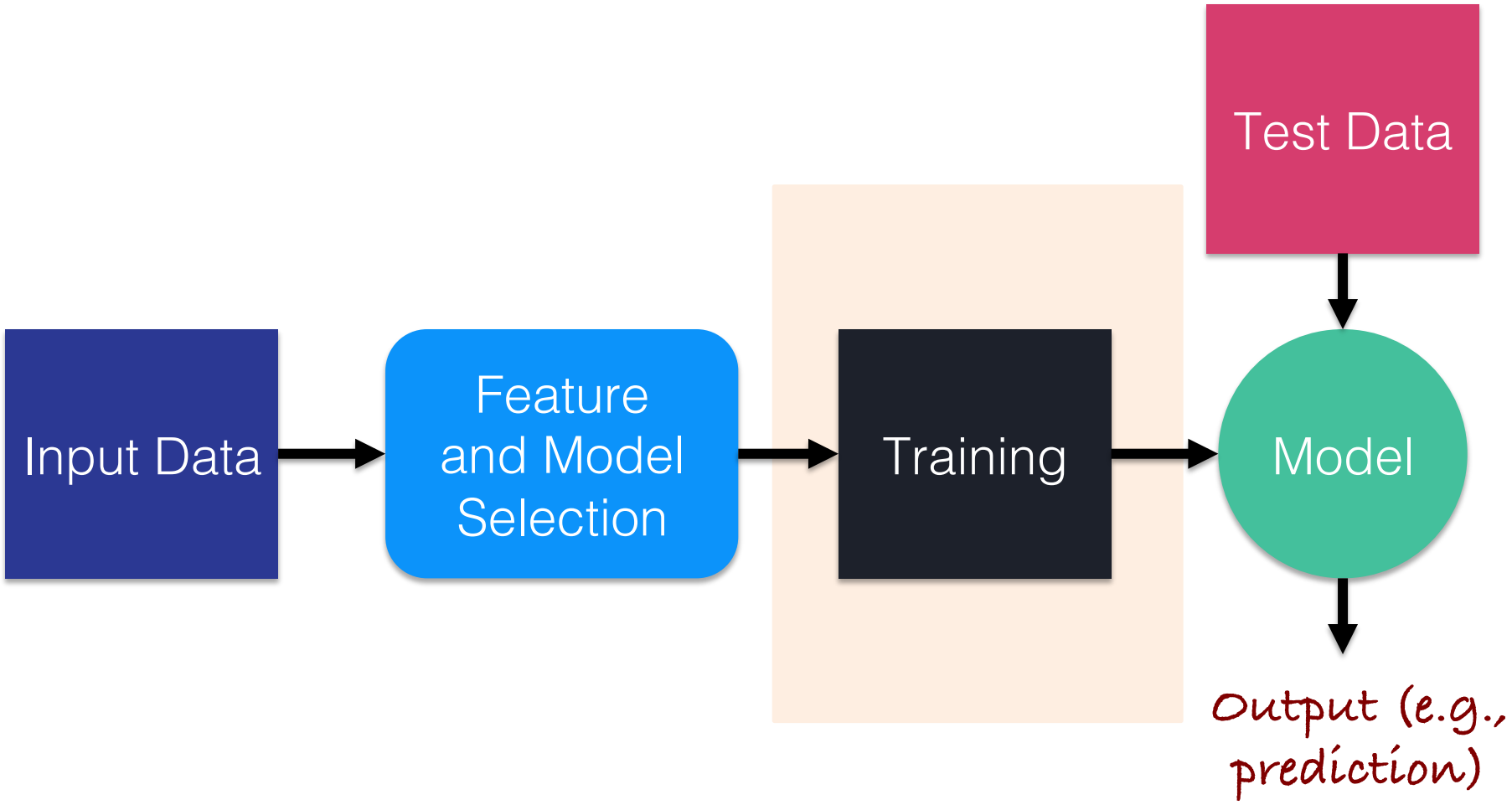
Goal: Use “informative” features

Choose your Predictor (Hypothesis Class)

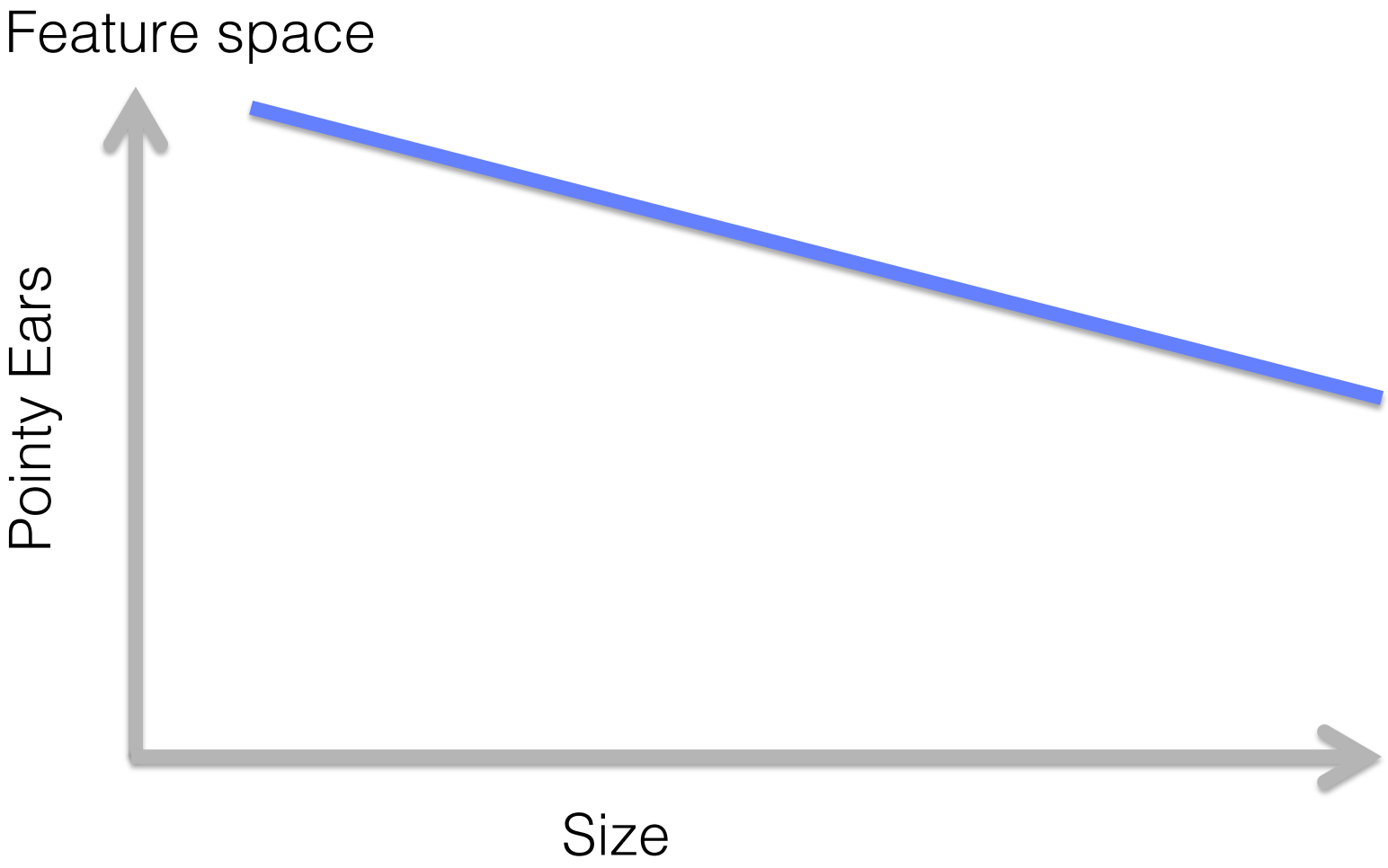


Goal: Pick a predictor that is
1) expressive 2) easy to train 3) doesn't overfit

ML Pipeline

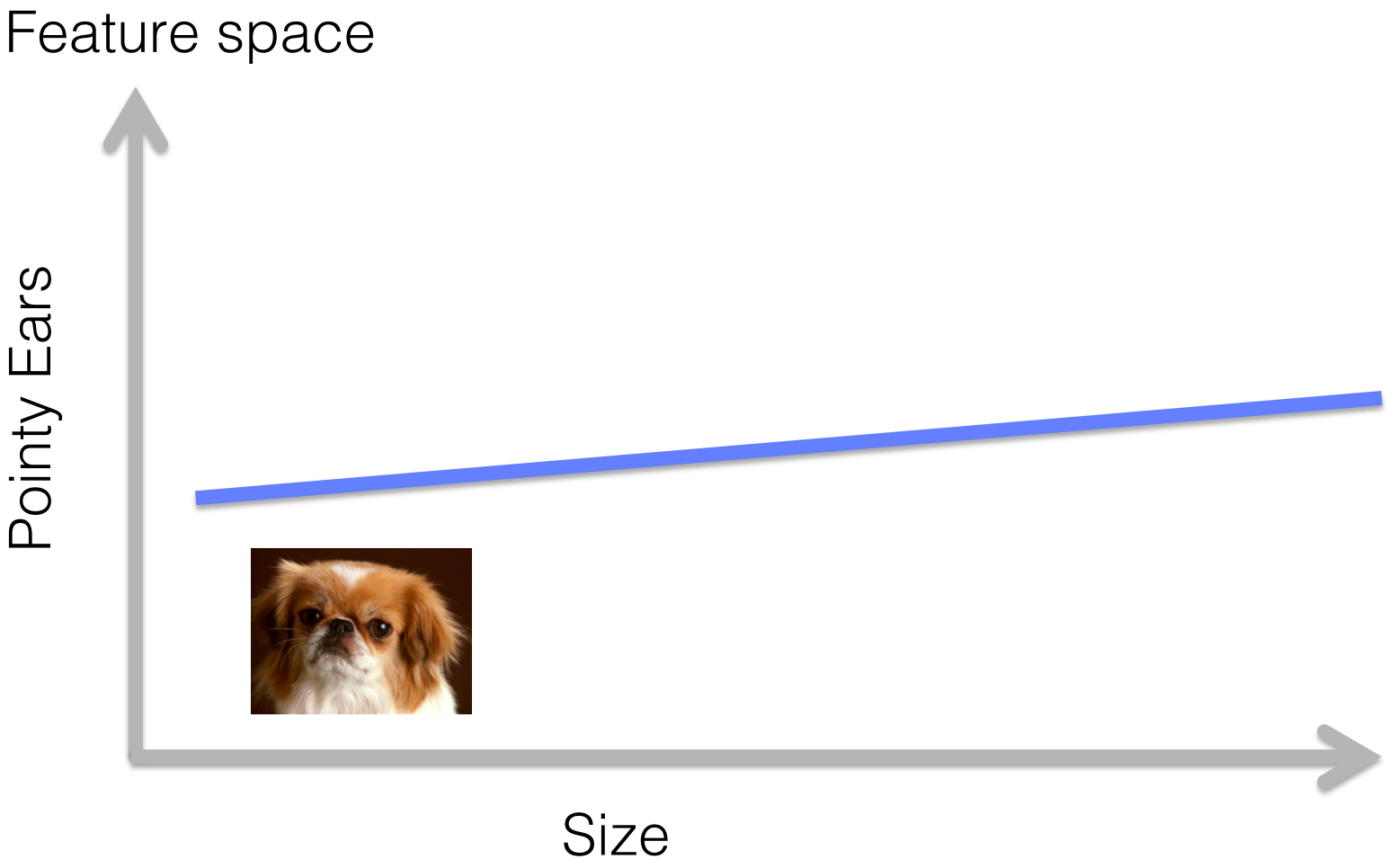

Test Data
Input Data
Feature and Model Selection
Training
Model
Output (e.g., prediction)

Then, Train the Model



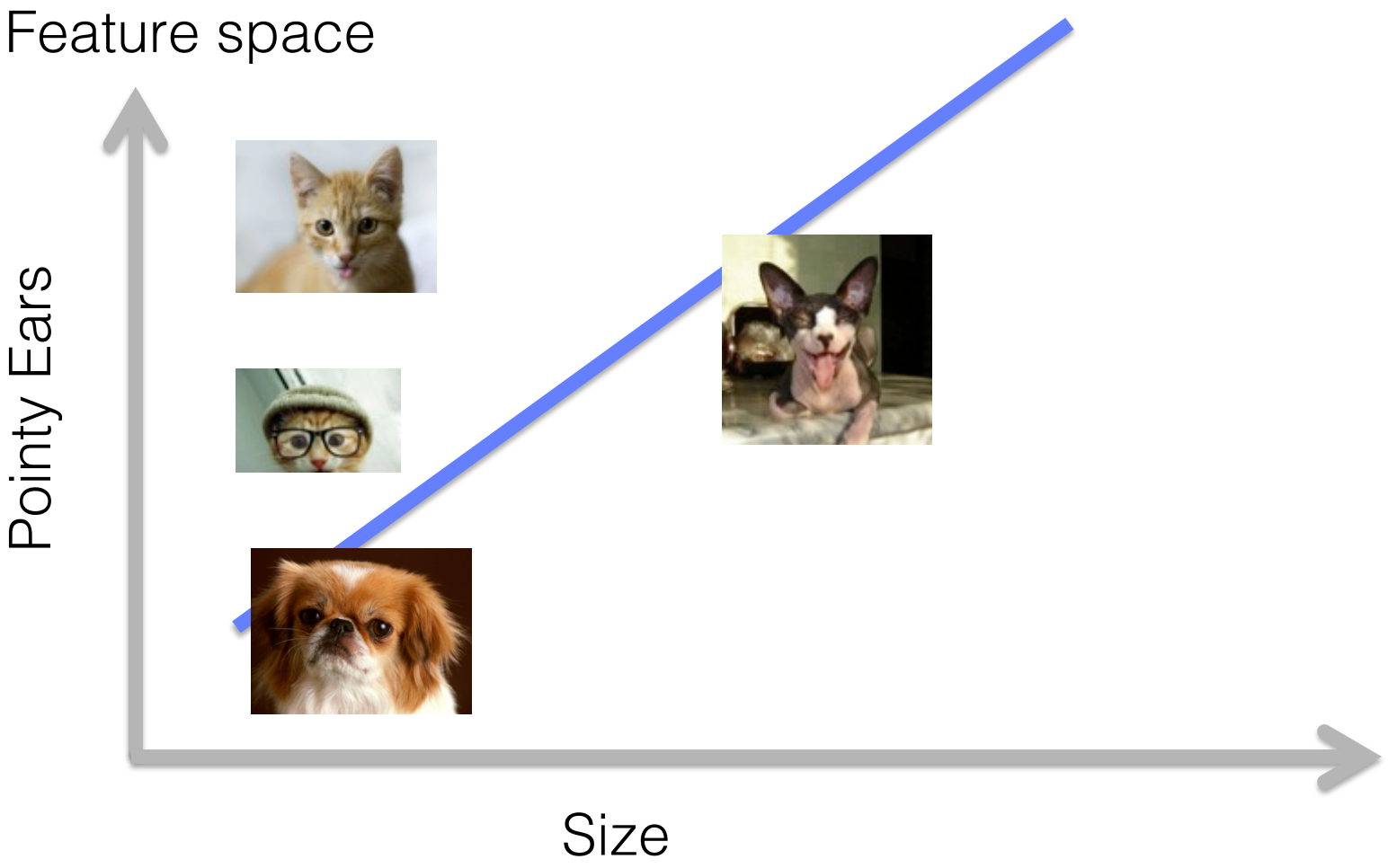
Goal: Train a model to minimize training error

Then, Train the Model



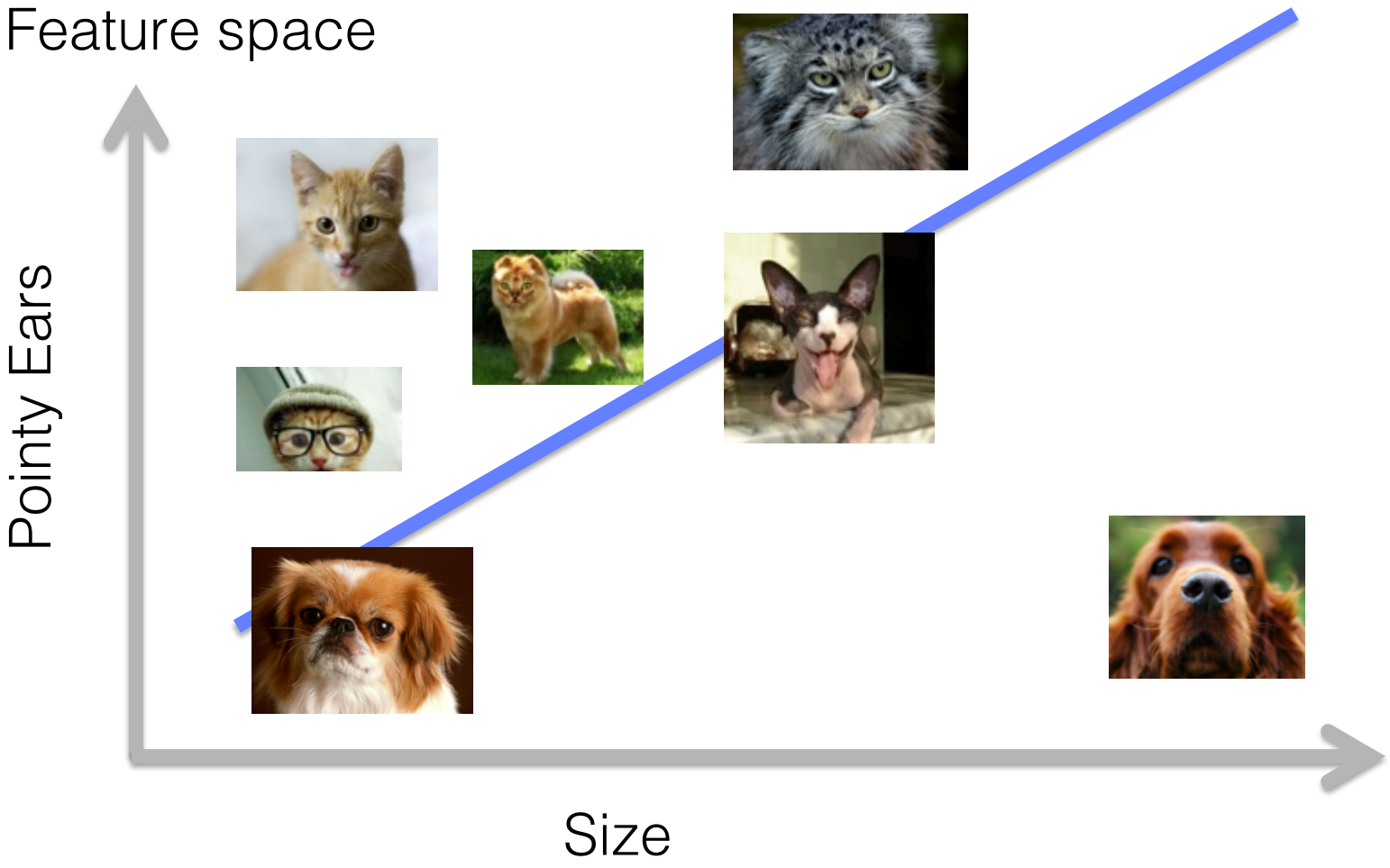
Goal: Train a model to minimize training error

Then, Train the Model



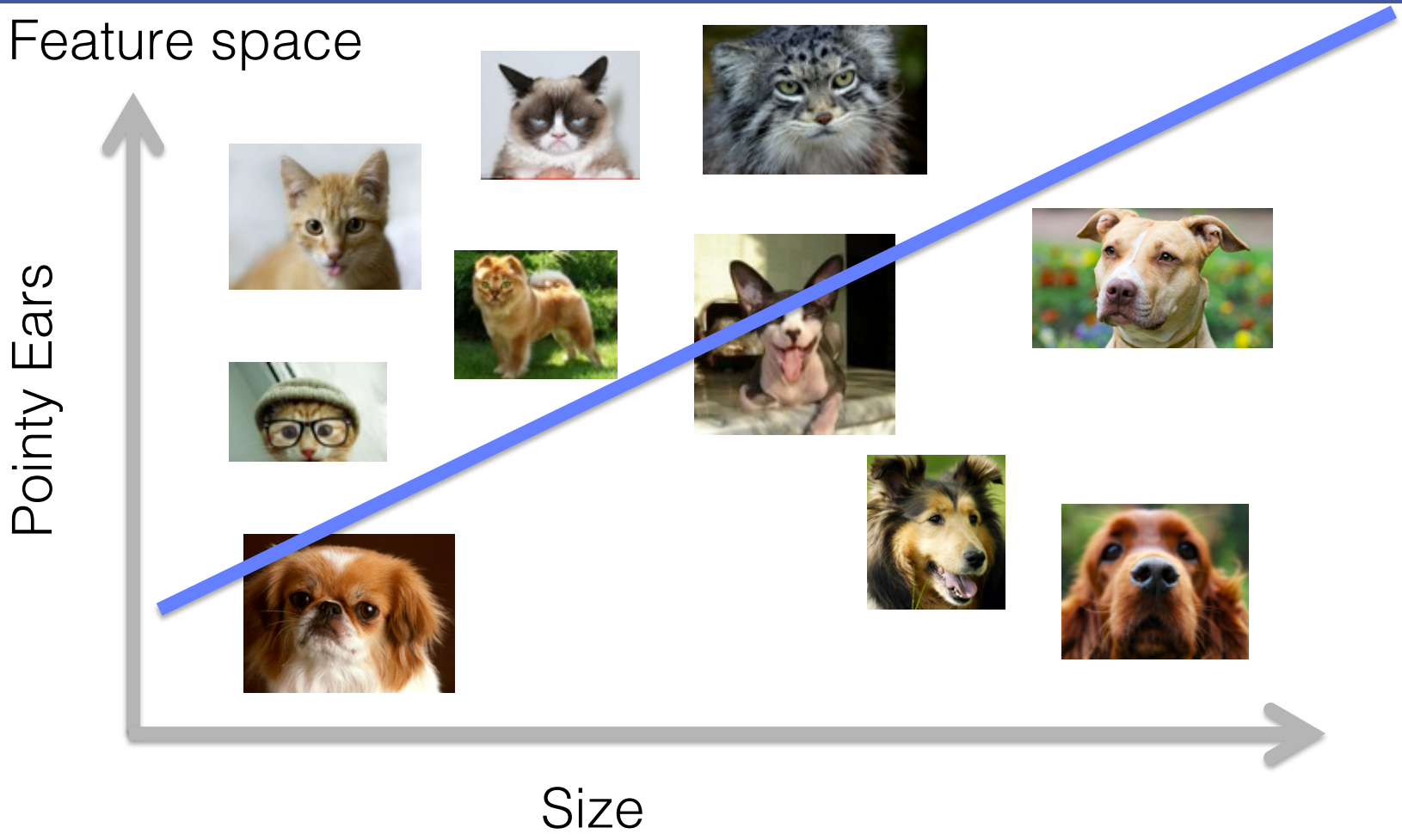
Goal: Train a model to minimize training error

Then, Train the Model



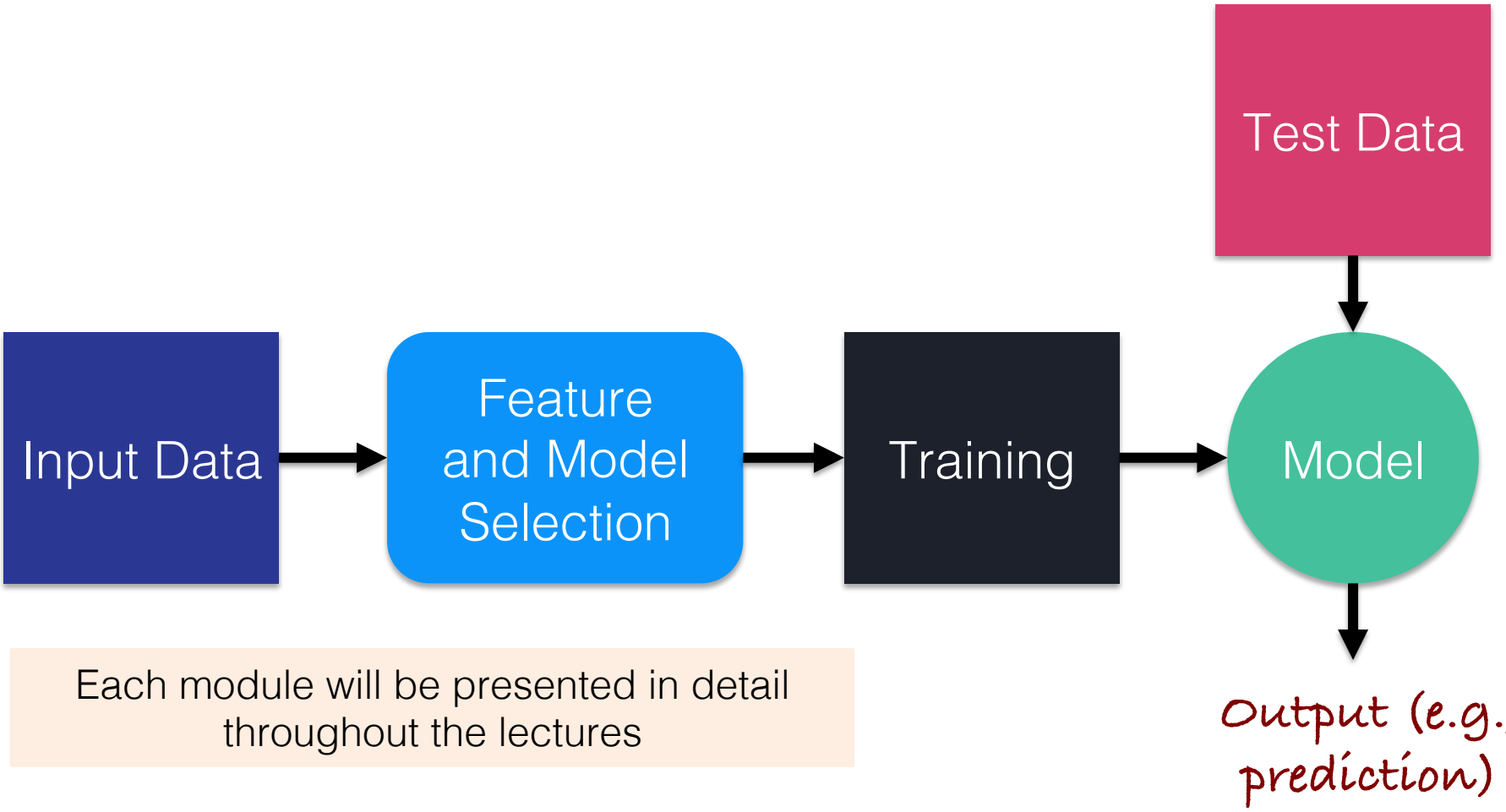
Goal: Train a model to minimize training error

Then, Train the Model



Goal: Train a model to minimize training error

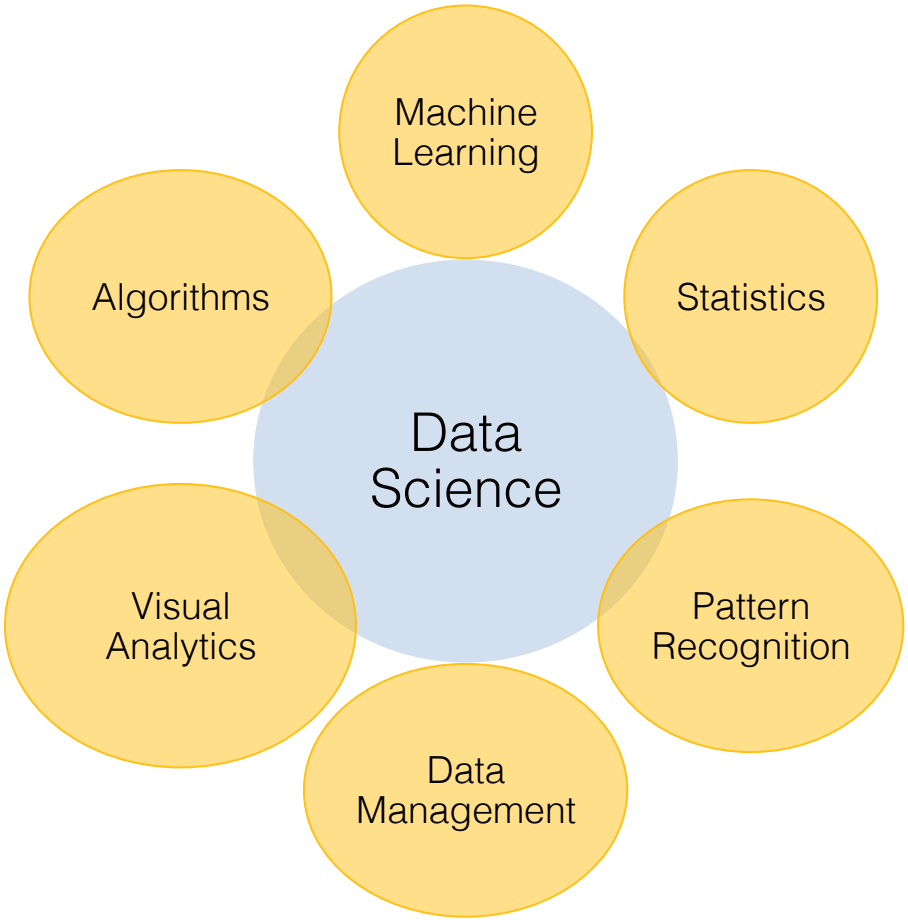
ML Pipeline



Each module will be presented in detail throughout the lectures

Questions?

Data Science: The Big Picture



About this course

Learning Objectives

- By the end of the course, you will be able to
  - Identify problems that can be solved by machine learning
  - Formulate your problem in machine learning terms
  - Given such a problem, identify and apply the most appropriate classical algorithm(s)
  - Implement some of those algorithms by yourself
  - Evaluate and compare machine learning algorithms for a particular task
  - Deal with real-world data challenges

Prerequisites

- Basic knowledge of
  - Probability theory and statistics
  - Linear algebra
  - Algorithms
- The course extends the “Statistics and Statistical Learning” course (1CC5000)
- We will review the main background concepts
- Programming is necessary
  - Python (or any other language of your preference)
  - We will deal with real-world ML tasks

Structure of the Course

Three components:

1. Lectures
   - First half of each session (1 ½ hours)
2. Lab sessions
   - Hands-on experience on ML algorithms
     - Some of the algorithms will be implemented from scratch
   - Need to install software and to experiment in class
   - Labs will not be graded, but will help you to further understand the material presented in the lectures
3. Assignments and project

Coursework and Grading

	Weight	Details
Assignment 1 (individual)	10%	• Theoretical questions • Some of them may also require some programming in order to perform some tasks • *Week 2 (out) – Week 3 (due)*
Assignment 2 (teams of 3-4 students)	20%	• Deal with a real machine learning task • Kaggle competition • Deliver short report and code • *Week 3 (out) – ~Week 5 (due)*
Project (teams of 3-4 students)	30%	• Project proposal (2-pages, mandatory) • Final report and code • *Proposal due: ~Week 3; Final report due: end of course*
Exam	40%	• Written exam in class

- Small adjustments may be done in the weights of the coursework
- A detailed description of the project will be provided soon
- The exact dates will be posted on the website of the course (subject to change)
- Written exam is mandatory

Goals of the Different Course Components

- Understand the basics behind ML algorithms and get comfortable working with data and tools [Lab sessions]
- Comprehend the foundational material and the motivation behind different techniques [Assignment 1]
- Build something that actually works [Assignment 2]
- Apply your knowledge creatively [Project]

Software Tools

- We strongly advise to use Python
  - numpy
  - scipy
  - scikit-learn
  - pandas
  - …
  - anaconda includes almost all packages that will be needed

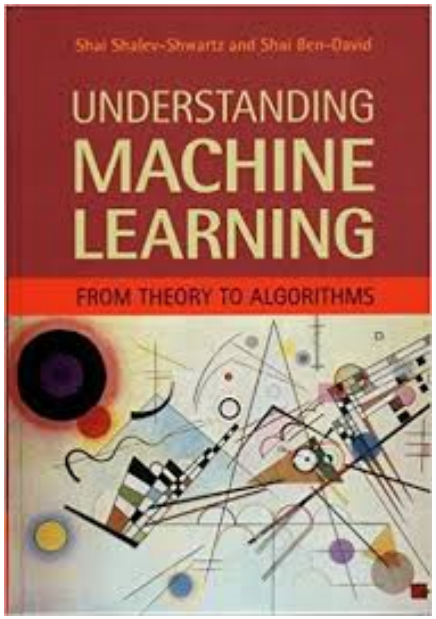
- Python will be used in the lab sessions
- See the Resources section of the website

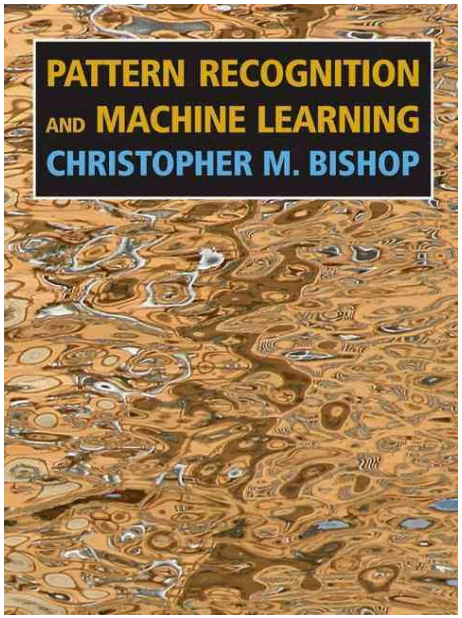
Course Logistics

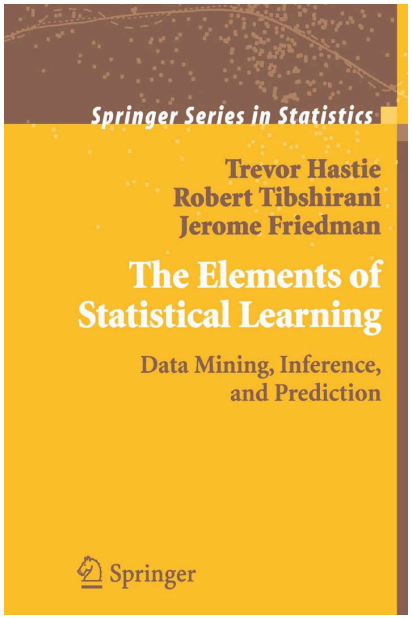
- Website
  - http://fragkiskos.me/teaching/2EL1730-F19/
  - Information about the course, schedule, reading material
  - Resources (helpful for the assignment and project)

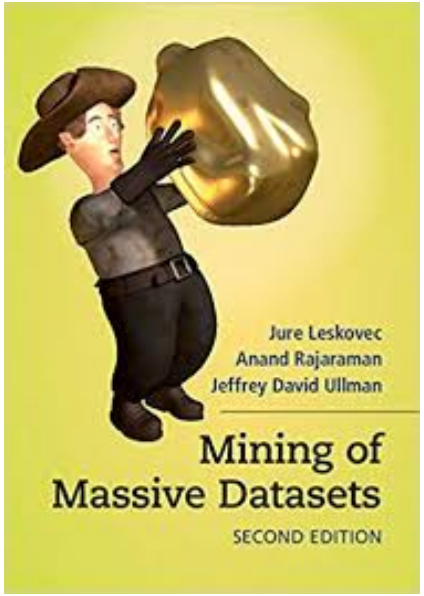
- Piazza for Q&A and material
  - https://piazza.com/centralesupelec/winter2020/2el1730
  - Please, participate and help each other!
  - All announcements will be posted there
  - Also, lecture slides and assignments
  - Use key to enroll: 2el1730-2019

Reading Material









- The books are publicly available in electronic form
  - Pointers to chapters for every lecture (see the website)

- Additional resources for every lecture will be given in the website of the course

Some Personal Notes ☺

- Please ask questions, participate in discussions on piazza
- Check out the additional suggested material on the website
  - Search the web, google is your friend!
  - For every topic covered in the class, you can find material in textbooks or even in the web
  - Typically, the suggested reading material is overlapping – read selectively
- Play with software tools. Apply what you've learnt in theory
  - This is the actual goal of the lab sessions, assignments and the project
- Give us your feedback!

Topics that will be covered

Schedule (Subject to Change)

http://fragkiskos.me/teaching/2EL1730-F19/

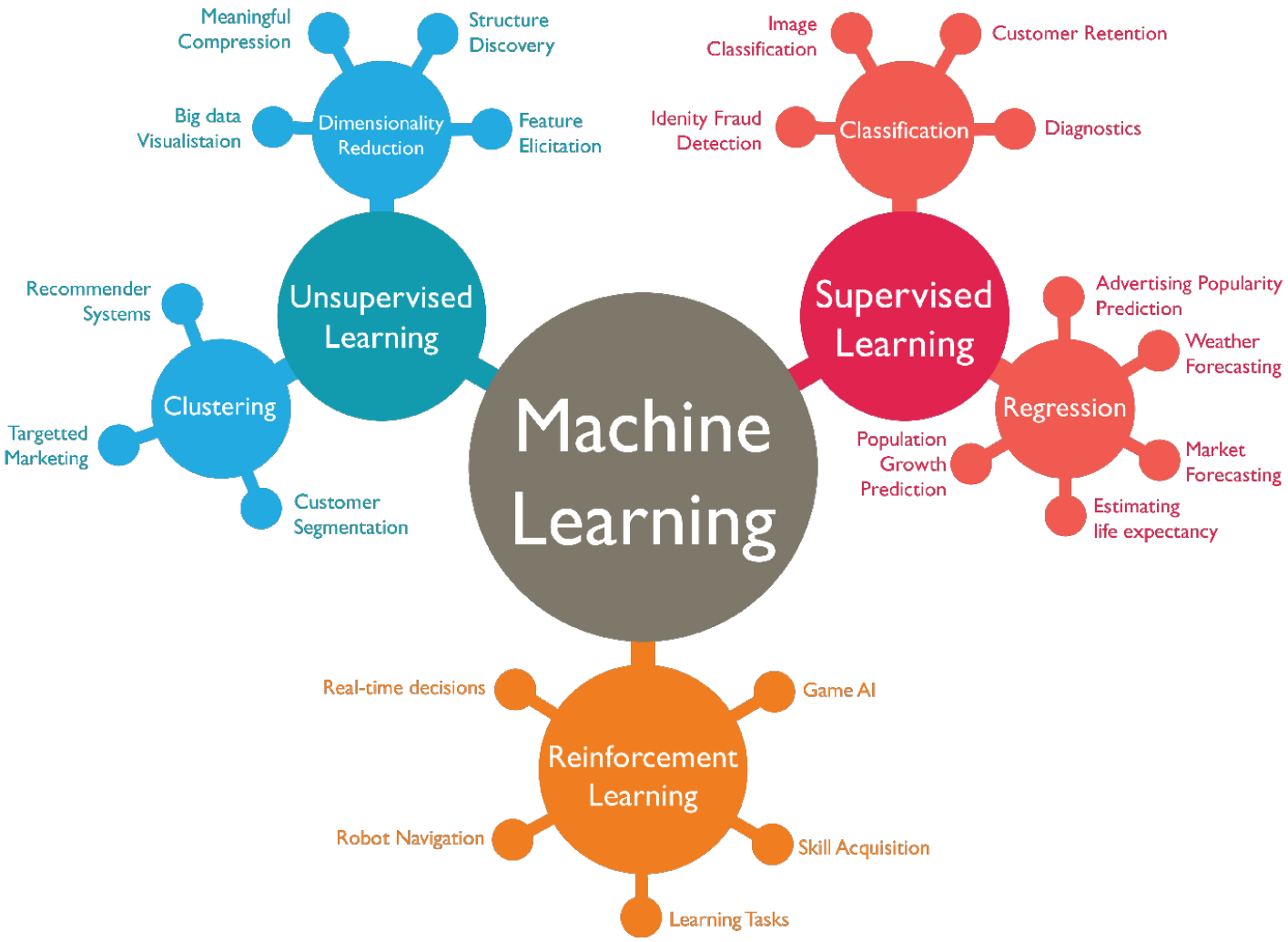
Lecture	Date	Topic	Material	Assignments/Project
1	November 26	Introduction; Model selection and evaluation	Lecture 1	
2	November 29	Dimensionality reduction		
3	December 3	Linear and logistic regression		**Assignment 1 out**
4	December 6	Probabilistic classifiers and linear discriminant analysis		
5	December 13	Non-parametric learning and nearest neighbor methods		**Project proposal due on December 13** **Assignment 2 out**
6	December 17	Support Vector Machines		**Assignment 1 due on December 17**
7	December 20	Tree-based methods and ensemble learning		
8	January 7	Neural Networks		
9	January 10	Introduction to deep learning Guest lecture		
10	January 14	Introduction to reinforcement learning Guest lecture		**Assignment 2 due on January 14**
11	January 17	Unsupervised learning: clustering		
12	January 22	Exams		**Project final report due on January 20**

Schedule (Subject to Change)

1. Introduction. Overview of ML problems. Model evaluation and selection. Overfitting and regularization. Concepts in optimization
2. Dimensionality reduction. Feature selection. Principal Component Analysis (PCA)
3. Linear and logistic regression
4. Probabilistic classifiers. Linear Discriminant Analysis (LDA)
5. Non-parametric learning. K-Nearest Neighbors
6. Support Vector Machines (SVMs)
7. Tree-based methods. Ensemble methods. Boosting. Random forests
8. Neural Networks
9. Deep Learning
10. Reinforcement Learning
11. Unsupervised learning. Data Clustering

Over 11 lectures

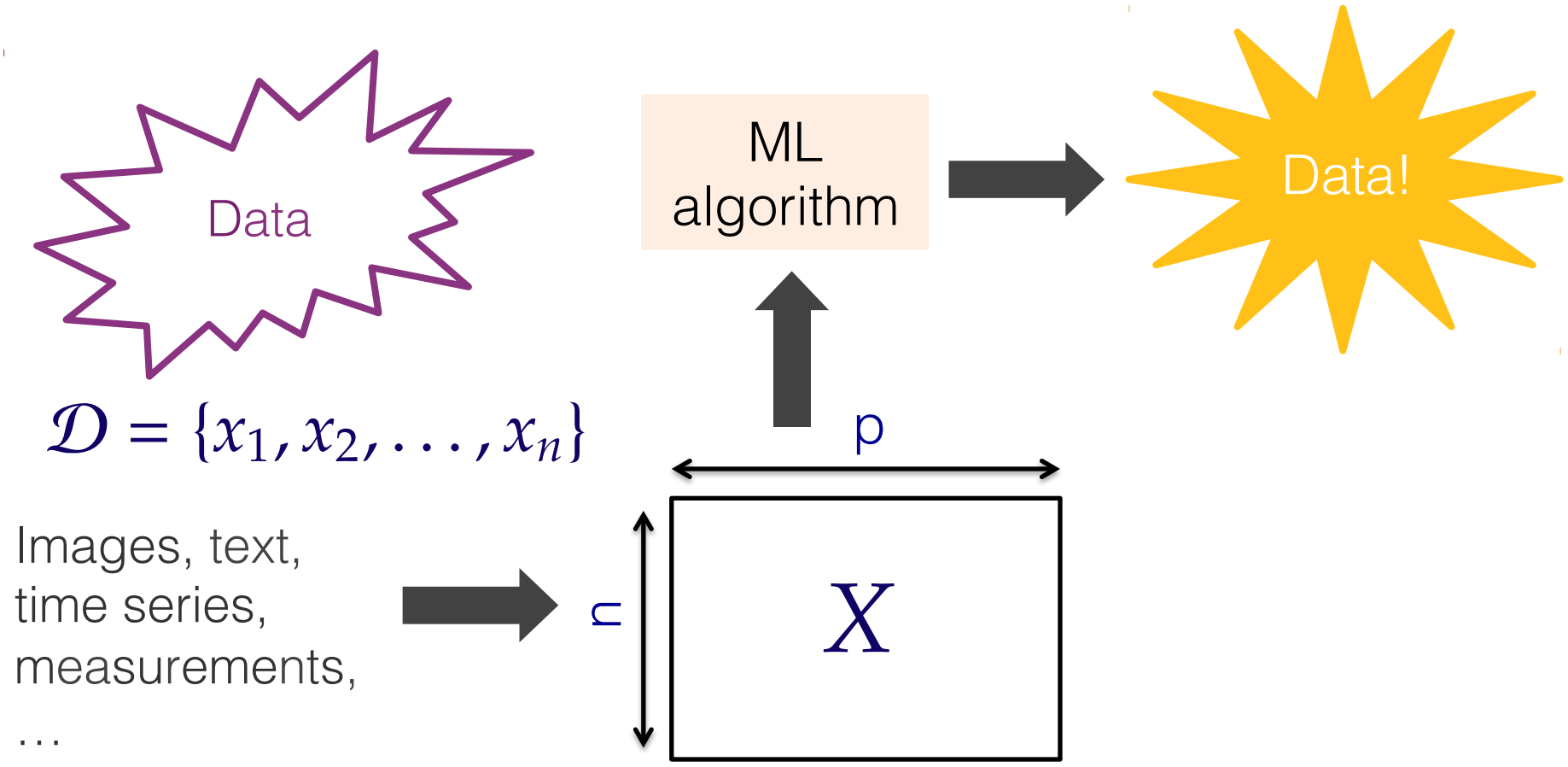
ML Models and Applications



Source: http://towardsdatascience.com

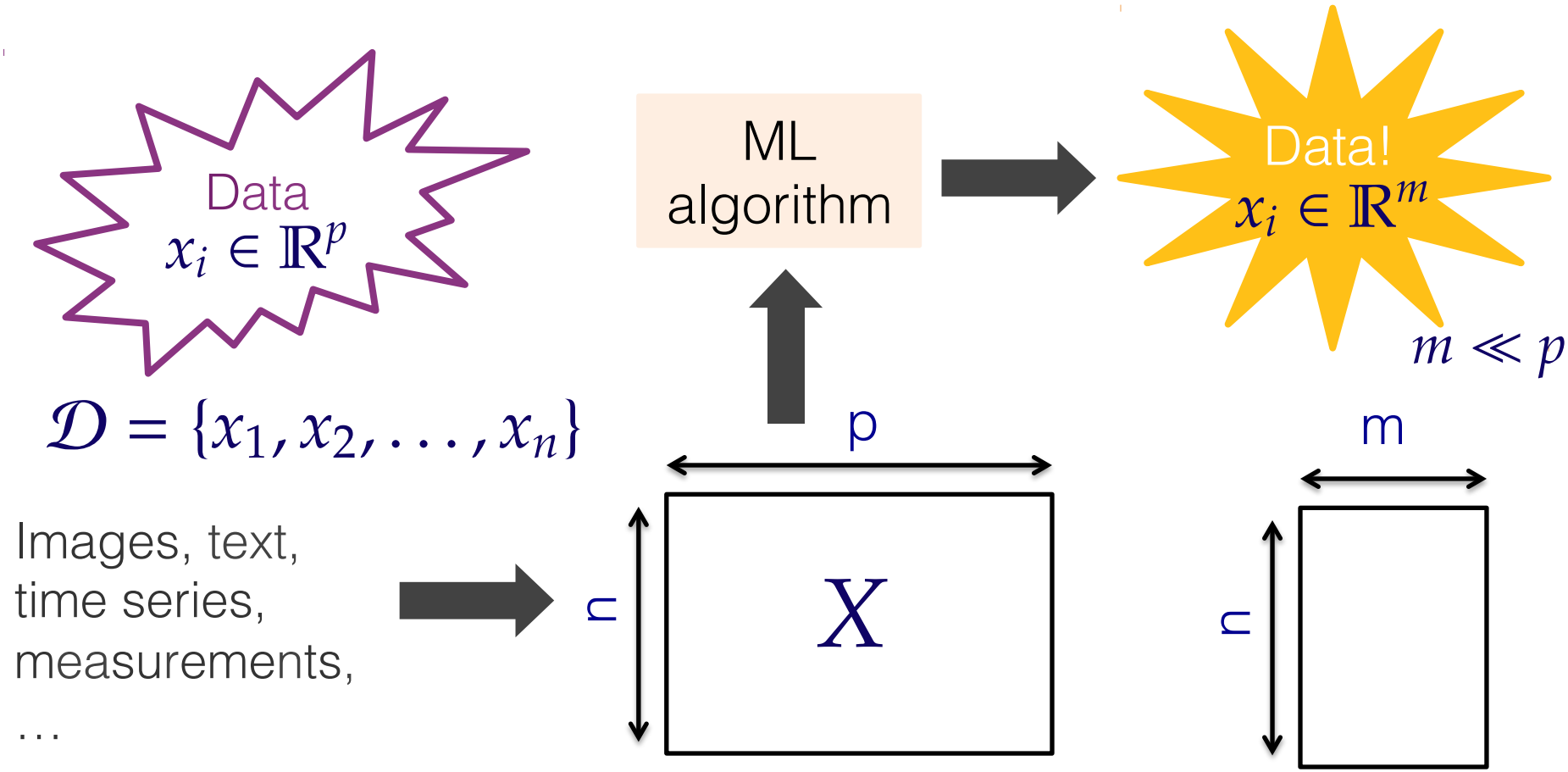
Unsupervised Learning

Learn a new representation of the data



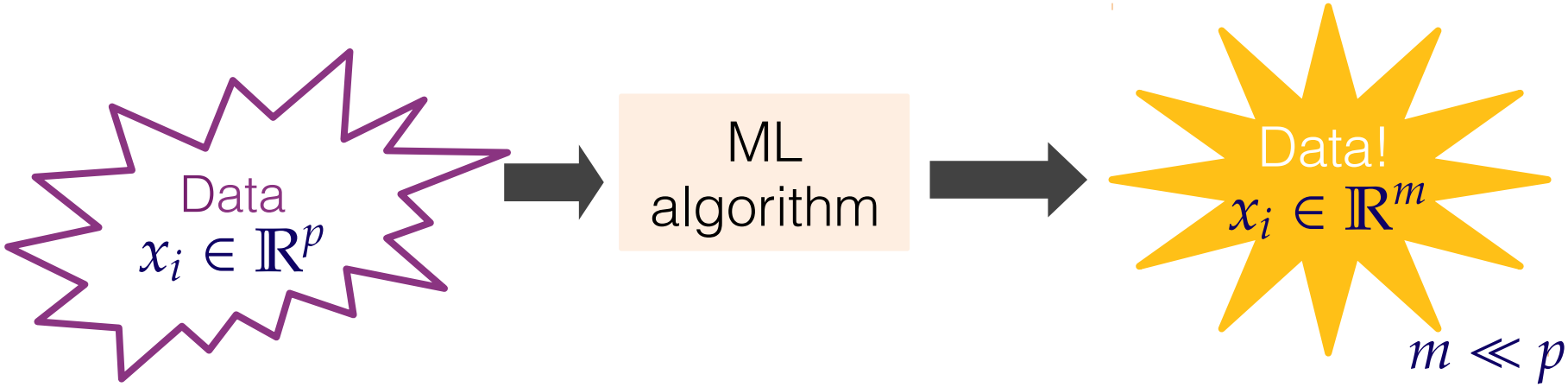
Dimensionality Reduction (1/2)

Find a lower-dimensional representation



Dimensionality Reduction (2/2)

Find a lower-dimensional representation



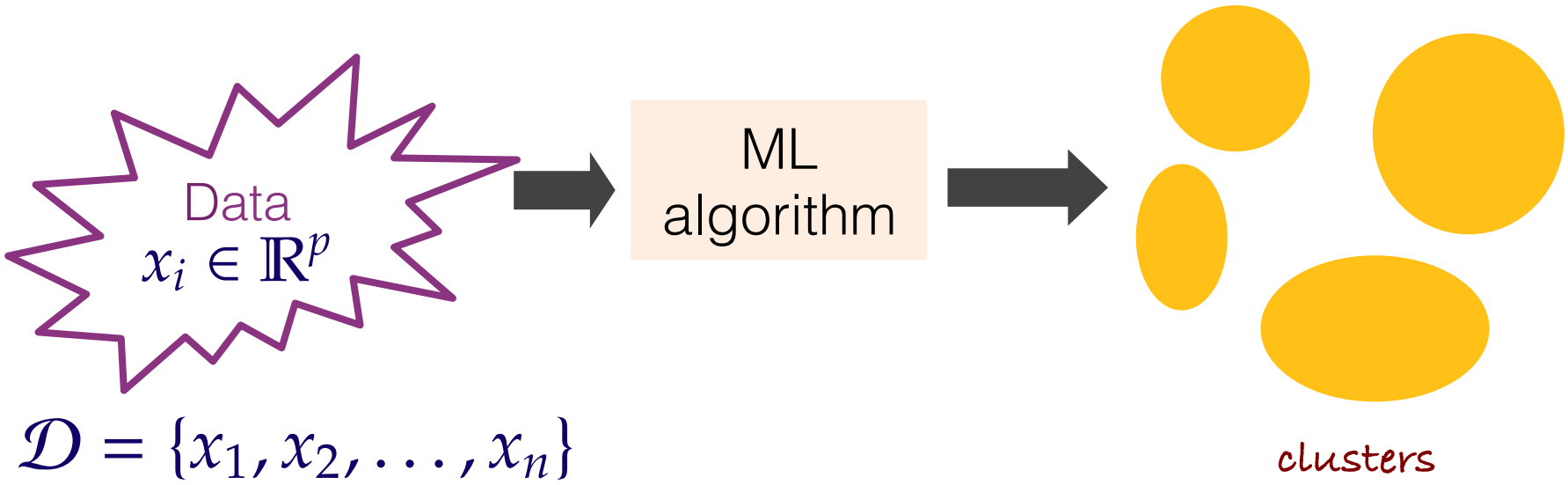
$$\mathcal{D} = \{x_1, x_2, \ldots, x_n\}$$

Why dimensionality reduction is useful?

- Reduce storage space and computational time
- Remove redundancies
- Curse of dimensionality
- Visualization (in 2 or 3 dimensions) and interpretability

Data Clustering

Group similar data points together



$$\mathcal{D} = \{x_1, x_2, \ldots, x_n\}$$

- Understand general characteristics of the data
- Infer some properties of an object based on how it relates to other objects
- Problems that can be solved by clustering?

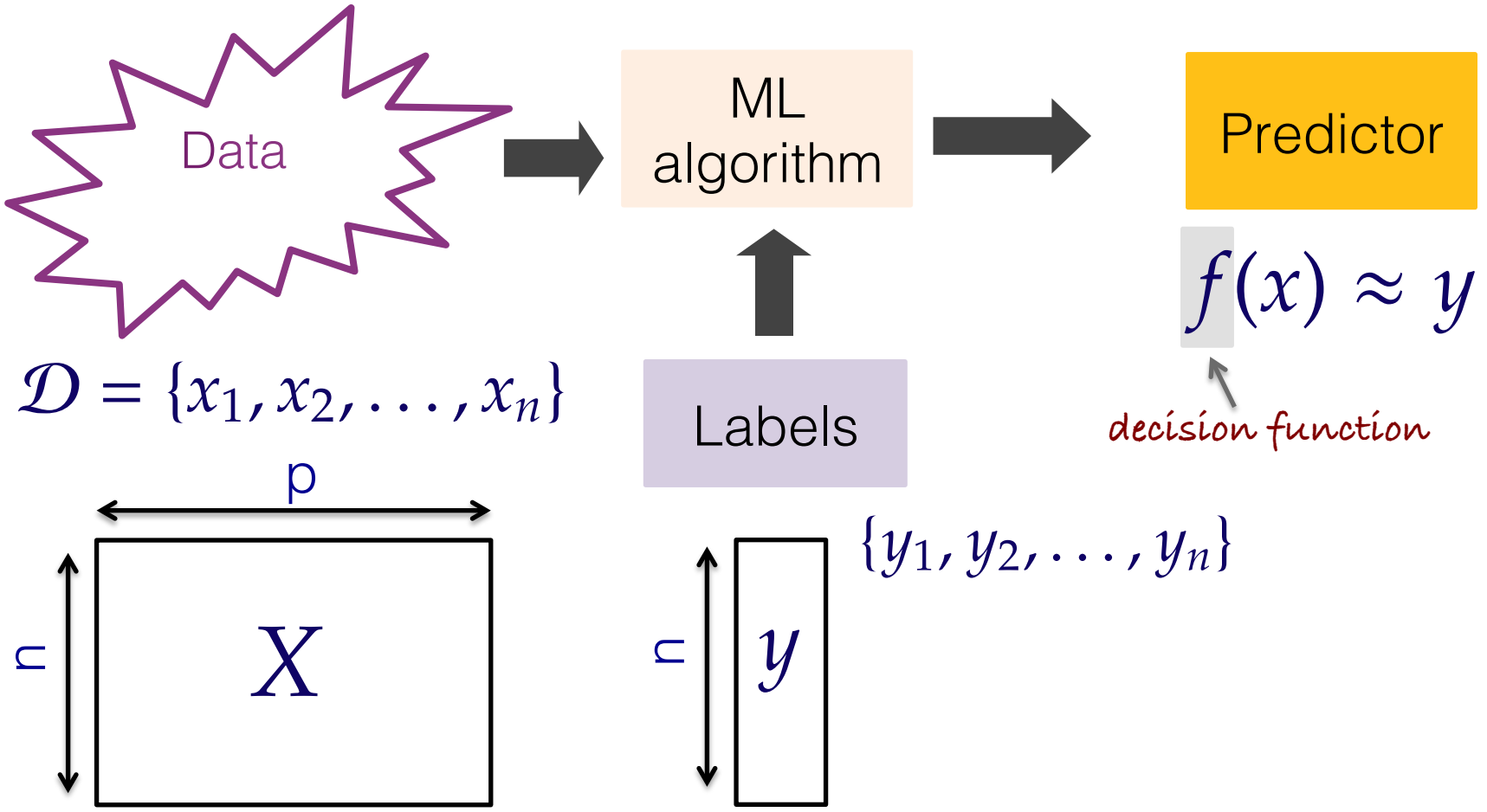
Data Clustering – Applications

- Customer segmentation
  - Find groups of customers with similar buying behavior
- Topic modeling
  - Group documents based on the words they contain to identify common topics
- Image compression and segmentation
  - Find groups of similar pixels that can easily be summarized
- Disease subtyping (e.g., cancer, mental health)
  - Find groups of patients with similar pathologies (at the molecular or symptoms level)
- Community detection in networks
  - Communities of similar users in social networks
- …

Can you think any inherent difficulty in the clustering task?

Supervised Learning

Make predictions



Classification (1/2)

Make discrete predictions

$\mathcal{D} = \{x_1, x_2, \ldots, x_n\}$

ML algorithm

Labels

$\{y_1, y_2, \ldots, y_n\}$

Predictor

$f(x) \approx y$

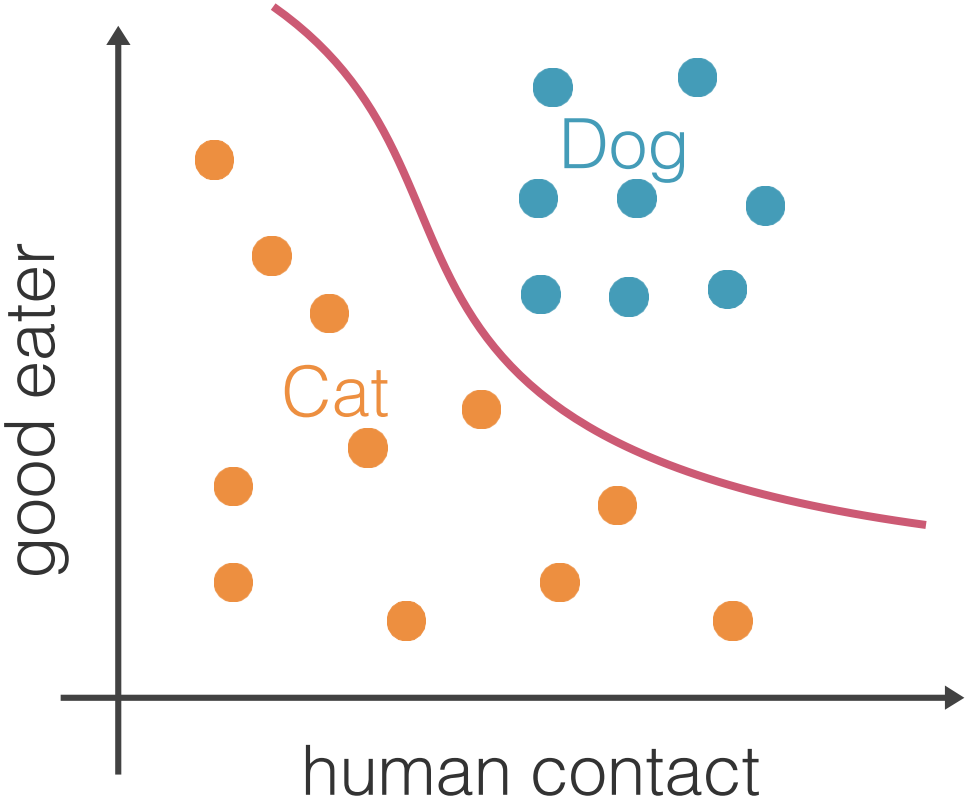
Binary classification

$y_i \in \{0, 1\}$

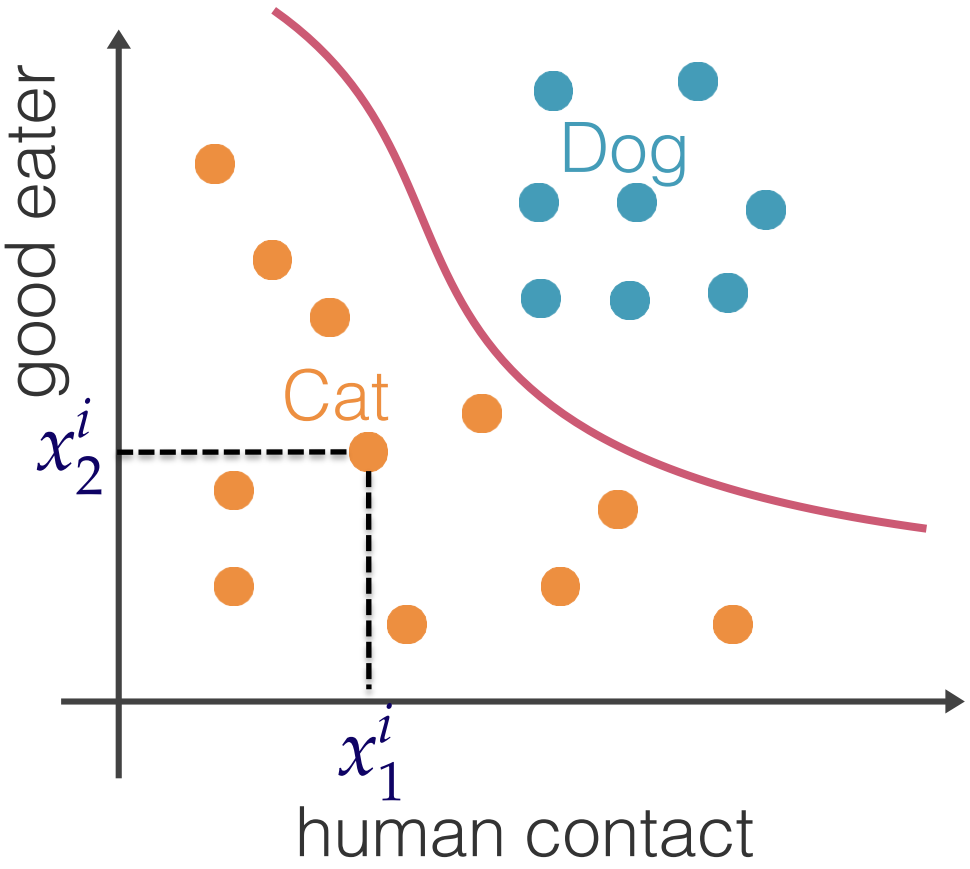
Multiclass classification

$y_i \in \{0, 1, \ldots, k\}$

Classification (2/2)



Training Set $\mathcal{D}$



$$\mathcal{D} = \{x^i, y^i\}_{i=1,\dots,n}$$

$$y_i = \begin{cases} 1 & \text{if } x^i \in \mathcal{P} \\ -1 & \text{if } x^i \in \mathcal{N} \end{cases}$$

$$x^i = \begin{pmatrix} x_1^i \\ x_2^i \end{pmatrix}$$

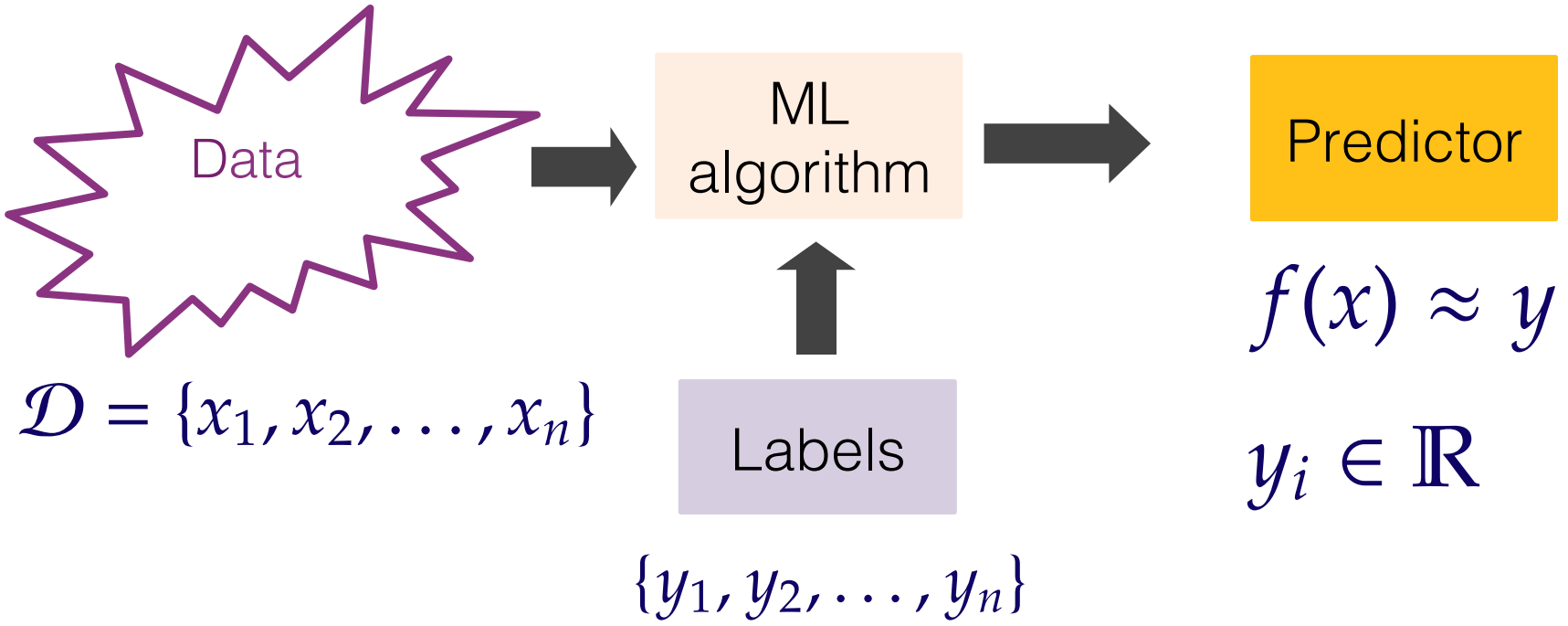
Given $\mathcal{D} = \{x^i, y^i\}_{i=1,\dots,n}$ find **f** such that $f(x) \approx y$

Classification – Applications

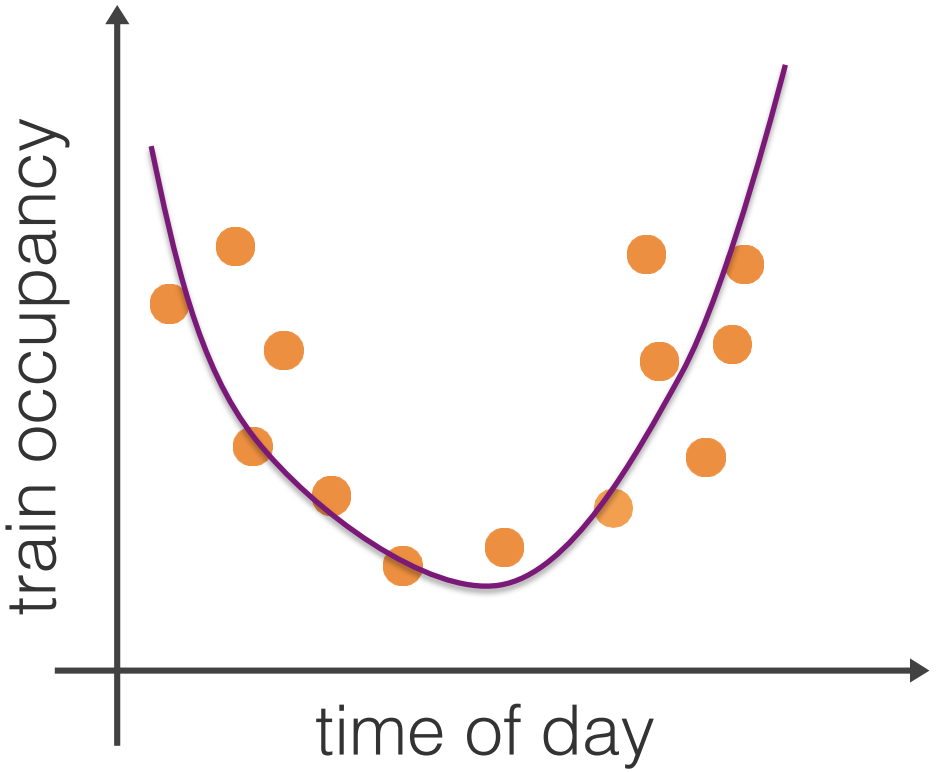
- Face recognition
  - Identify faces independently of pose, lighting, occlusion (glasses, beard), make-up, hair style
- Self-driving cars. How?
- Character recognition
  - Read letters or digits independently of different handwriting styles
- Sound recognition
  - Which language is spoken? Who wrote this music? What type of bird is this?
- Spam detection. Any spam application that you may know?
- Precision medicine
  - Does this sample come from a sick or healthy person? Will this drug work on this patient?

Regression (1/3)

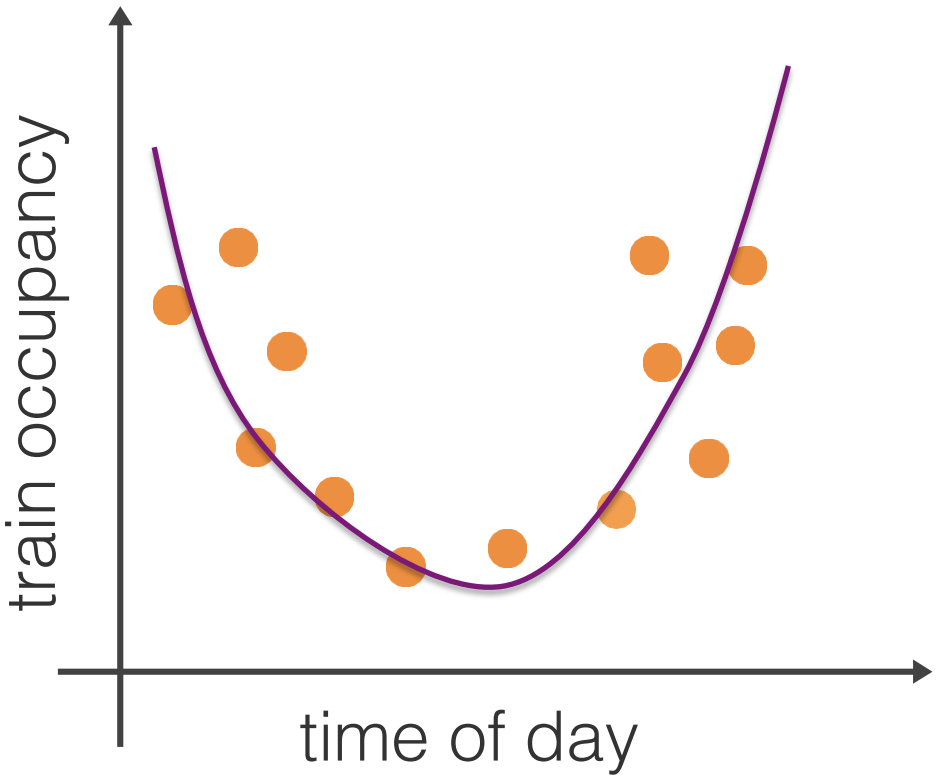
Make continues predictions



Regression (2/3)



Regression (3/3)



$$\mathcal{D} = \{x^i, y^i\}_{i=1,\ldots,n}$$

$$y^i \in \mathbb{R}$$

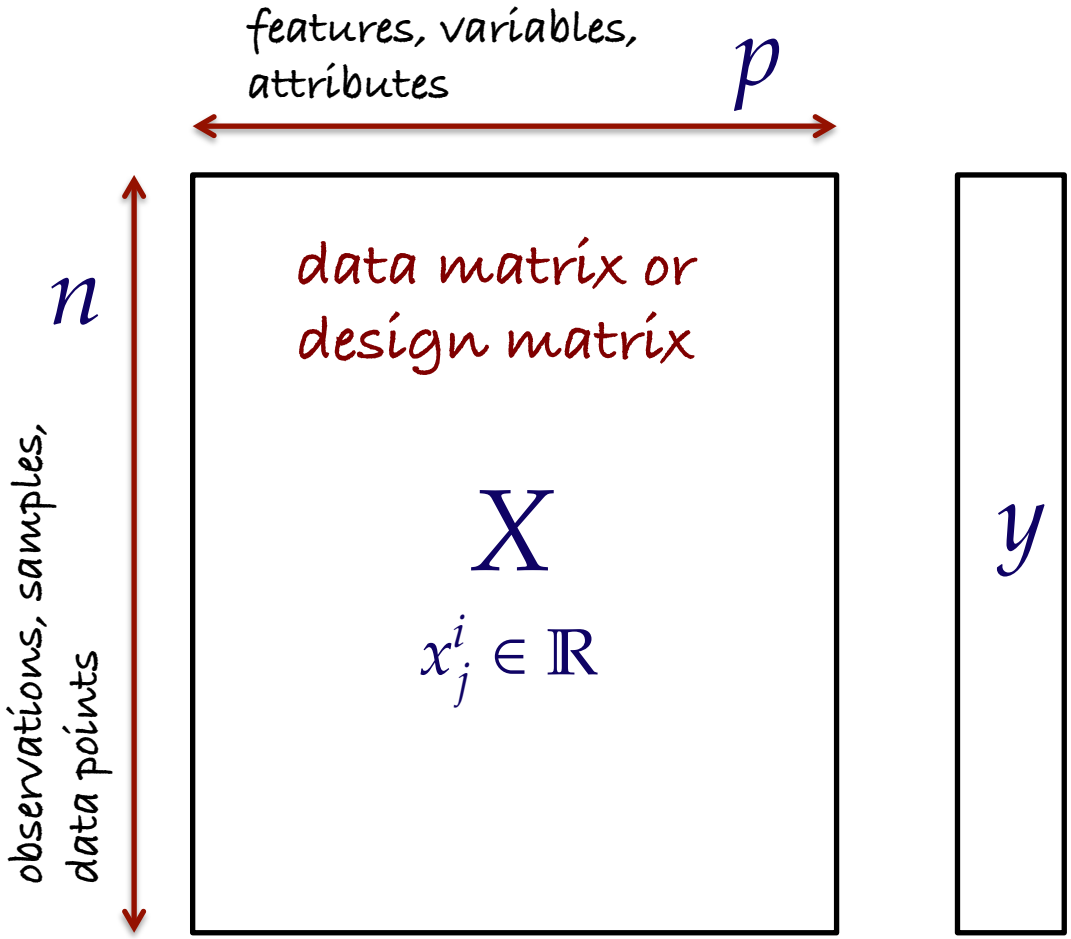
Given $\mathcal{D} = \{x^i, y^i\}_{i=1,\ldots,n}$ find **f** such that $f(x) \approx y$

Regression – Applications

- Click prediction
  - How many people will click on this ad? … comment on this post? … share this article on social media?
- Load prediction
  - How many users will my service have at a given time?
- Algorithmic trading
  - What will the price of this share be?

Supervised Learning Setting – Summary

Given $\mathcal{D} = \{x^i, y^i\}_{i=1,\ldots,n}$ find f such that $f(x) \approx y$



Binary classification

$y^i \in \{0, 1\}$

Multiclass classification

$y^i \in \{0, 1, \ldots, k\}$

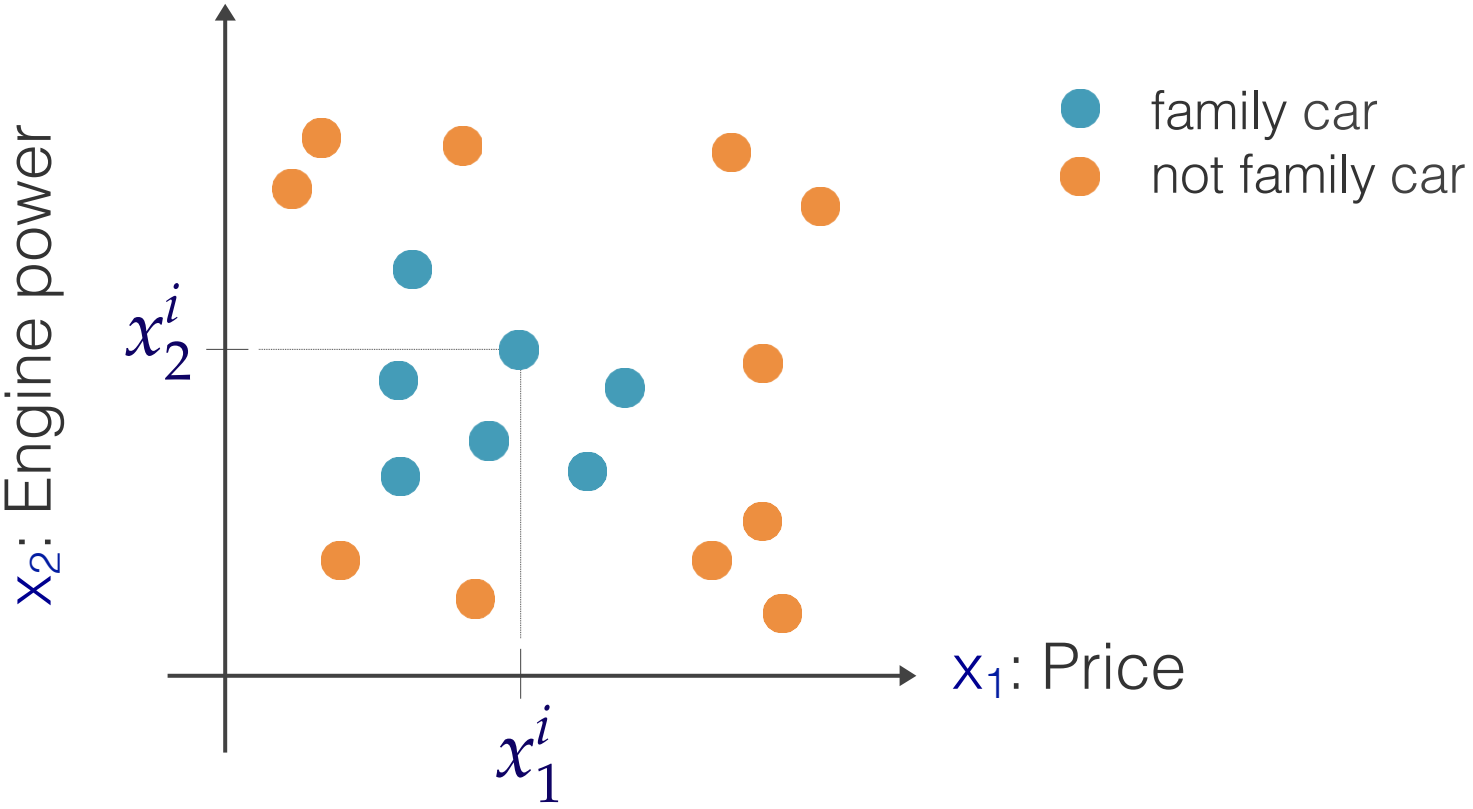
Regression

$y^i \in \mathbb{R}$

Hypothesis class, loss function, and risk minimization

Hypothesis Class

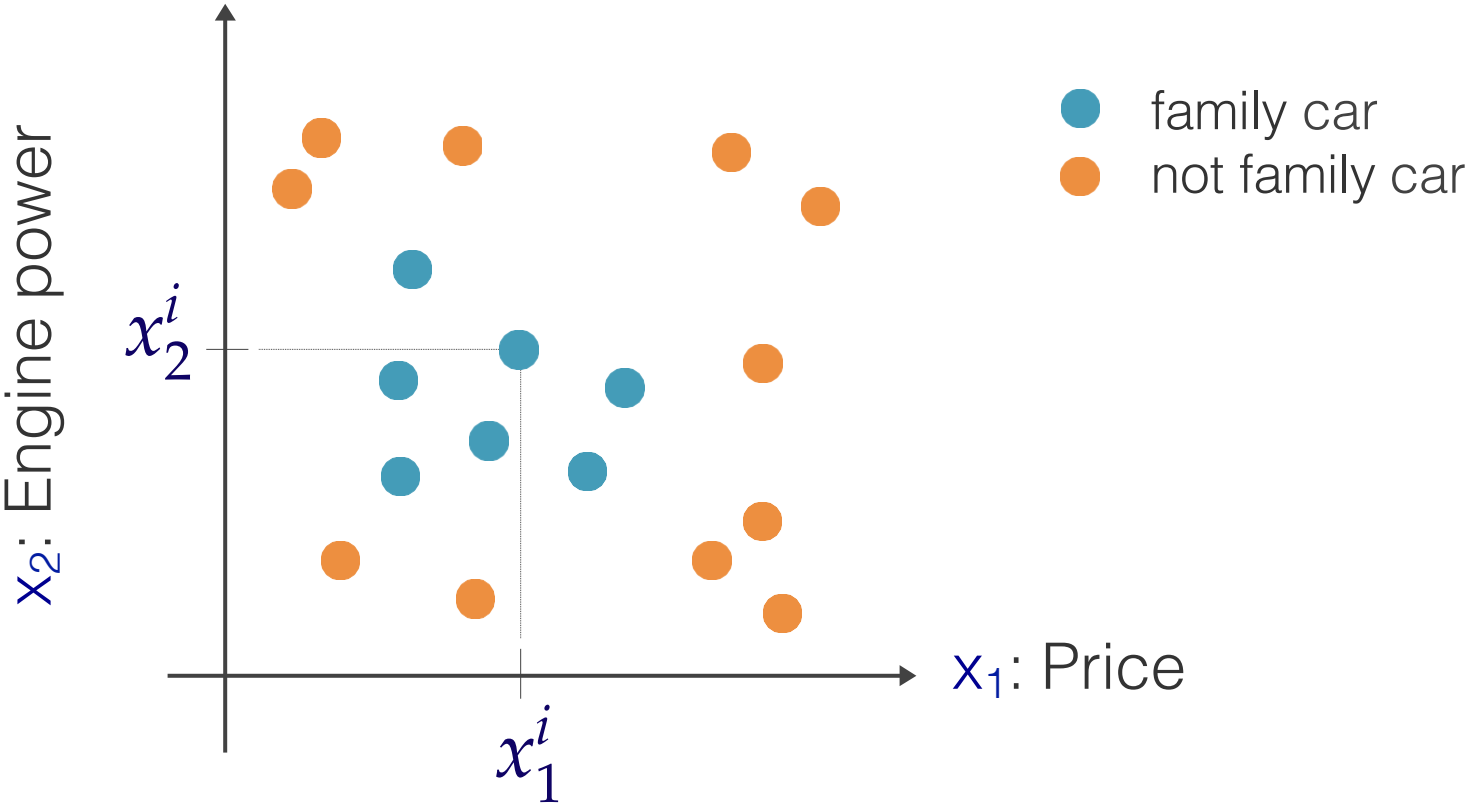
- Hypothesis class $\mathcal{F}$
  - The space of possible decision functions we are considering
  - Chosen based on our beliefs about the problem



Hypothesis Class

- Hypothesis class $\mathcal{F}$

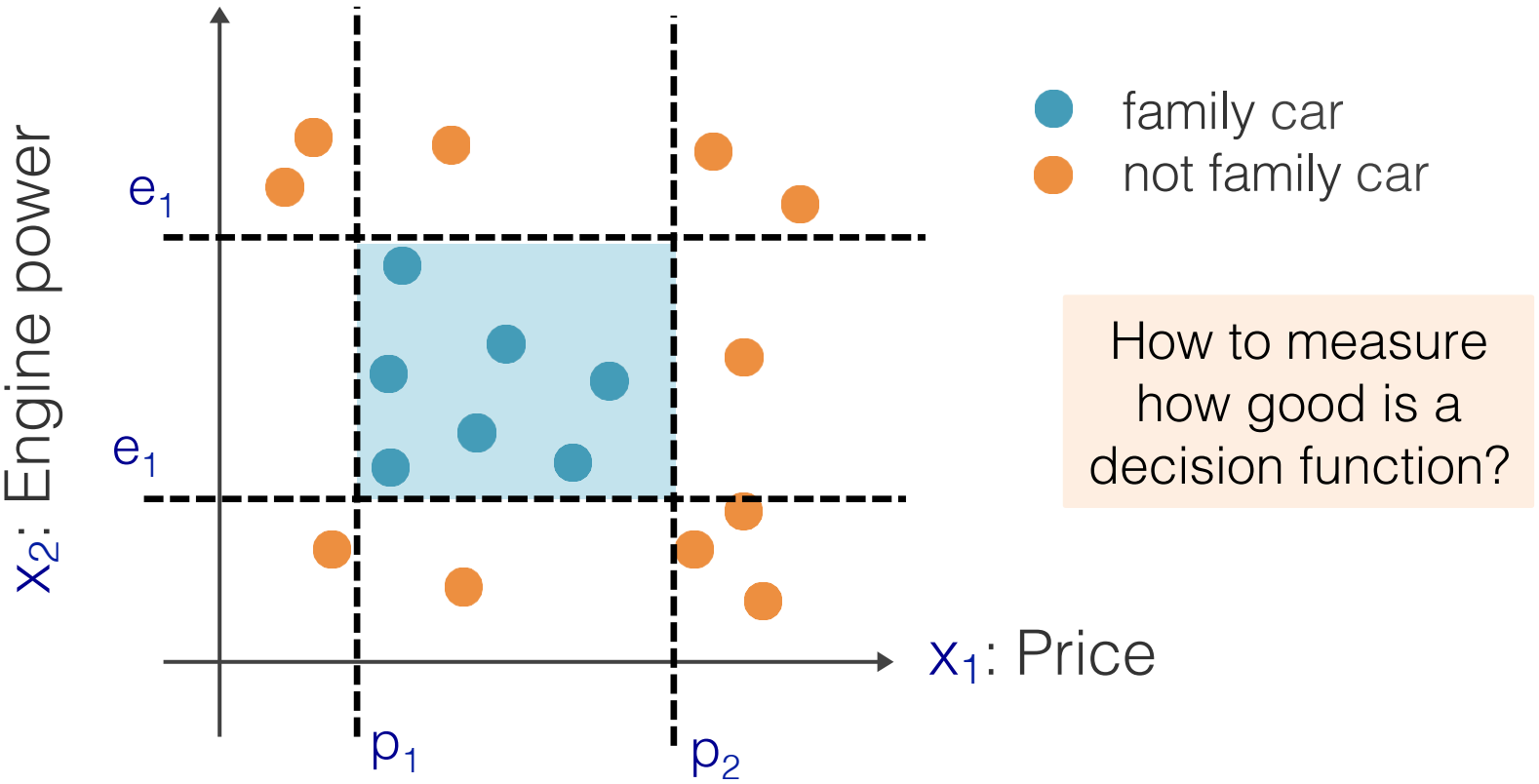
What shape do you think the discriminant should take?



Hypothesis Class

- Hypothesis class $\mathcal{F}$
  - Belief: the decision function is a rectangle

$$(p_1 \leq x_1 \leq p_2) \text{ AND } (e_1 \leq x_2 \leq e_2)$$



Loss Function

- Loss function (or cost function, or risk):

$$\mathcal{L} : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}$$

$$y, f(x) \rightarrow \mathcal{L}(y, f(x))$$

Quantifies how far the decision function is from the truth

Example of loss functions:

$$\mathcal{Y} = \{0, 1\} \quad \mathcal{L}(y, f(x)) = \begin{cases} 0 & \text{if } y = f(x) \\ 1 & \text{otherwise} \end{cases}$$

$$\mathcal{Y} = \mathbb{R} \quad \mathcal{L}(y, f(x)) = \|y - f(x)\|^2$$

Training via optimization: find that **f** among the hypothesis class $\mathcal{F}$ that minimizes the total loss

The Goal of Training (1/4)

- What we have: labeled examples presented as (observations, label)

$$\mathbf{d}_i = (\mathbf{x}^i, y^i)$$

- E.g., observation = image, label = “cat” (-1), or “dog” (+1)

$$\left(\text{  } \;,\; +1 \right)$$

- Assumption: all labeled (train/test) examples come from unknown distribution, say D

The Goal of Training (2/4)

- What we have: n labeled examples drawn *i.i.d.* from D

$$(\mathbf{x}^i, y^i), \ldots, (\mathbf{x}^n, y^n)$$

- What we want: train a model (a predictor function f)

$$f : \text{feature vector} \rightarrow \text{label}$$

That performs well on **unseen data**

- For example:

$$f(\ \ ) = +1$$

The Goal of Training (3/4)

- What we have: n labeled examples drawn *i.i.d.* from D

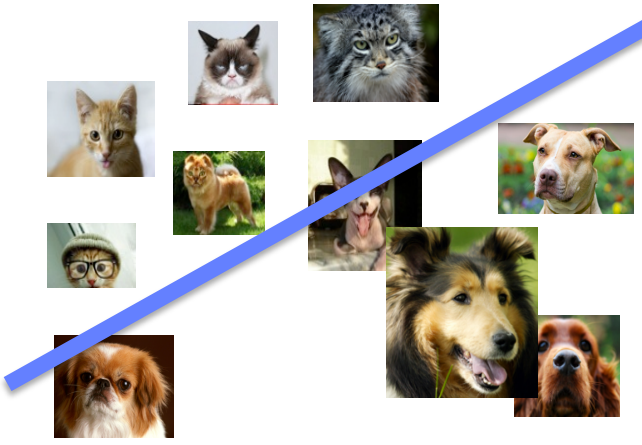
$$(\mathbf{x}^i, y^i), \ldots, (\mathbf{x}^n, y^n)$$

- What we want: train a model (a predictor function f)

$$f : \text{feature vector} \rightarrow \text{label}$$

That performs well on **unseen data**

- For example:



Our trained predictor f

The Goal of Training (4/4)

- How to measure performance? As we said before, we need to define the loss:

$$\mathcal{L}(y, f(\mathbf{x}))$$

Measures disagreement between predicted and true label

- Goal: we want a predictor f with small loss on unseen data (e.g., on a test set)

$$\sum_{(\mathbf{x},y) \text{ is an unseen example}} \mathcal{L}(y, f(\mathbf{x}))$$

But, we haven't seen unseen examples
(the test set is not known to the learning algorithm)

Empirical Risk

- The loss on the "unseen" examples converges to the expected loss

$$\sum_{(\mathbf{x},y) \text{ is an unseen example}} \mathcal{L}(y, f(\mathbf{x})) \rightarrow \mathbb{E}_{\mathbf{x},y}\{\mathcal{L}(y, f(x))\}$$

Expected/True risk

- What else converges to the expected loss?

$$\frac{1}{n}\sum_{i=1}^{n} \mathcal{L}(y^i, f(\mathbf{x}^i))$$

Empirical risk (ER)

By averaging the loss function on the training set

Theoretical aspects

- When is the ER a good estimator for true risk?
  - Does ER concentrate?
- Choice of the model and concentration

Empirical Risk Minimization (ERM)

- Training via optimization: we want to solve:

$$\arg\min_{f \in \mathcal{F}} \mathbb{E}\{\mathcal{L}(y, f(x))\}$$

f can be:
- separating hyperplanes
- NN's of depth-t
- …

- We instead solve the ERM:

$$\arg\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(y^i, f(\mathbf{x}^i))$$

Supervised Learning: 3 Ingredients

Given $\mathcal{D} = \{x^i, y^i\}_{i=1,\ldots,n}$ find f such that $f(x) \approx y$

- Chose a hypothesis class $\mathcal{F}$
  - Parametric methods – e.g., $f(x) = \sum_{j=1}^{p} \beta_j x_j$
  - Non-parametric methods – e.g., f(x) is the label of the point closest to x (Nearest Neighbors is such a method)
- Chose a loss function $\mathcal{L}$
  - Empirical error: $\frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(y^i, f(\mathbf{x}^i))$
- Chose an optimization procedure

$$f^* = \arg\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(y^i, f(\mathbf{x}^i))$$

Basic Concepts in Optimization

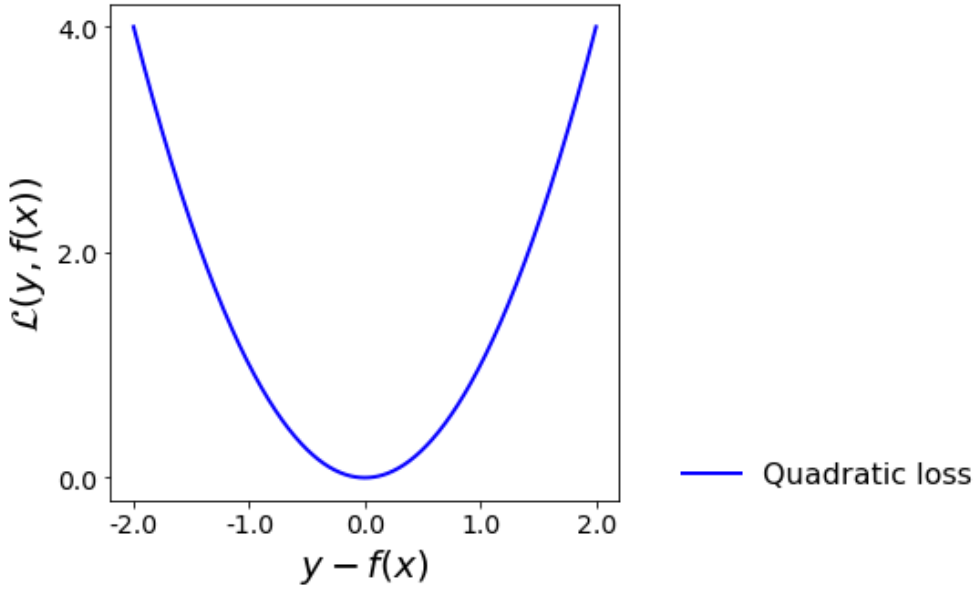
Why Optimization? (1/4)

- Empirical risk minimization:

$$f^* = \arg\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(y^i, f(\mathbf{x}^i))$$

Which loss functions are mainly used?

- Quadratic loss: $\mathcal{L}(y, f(\mathbf{x})) = (y - f(\mathbf{x}))^2$



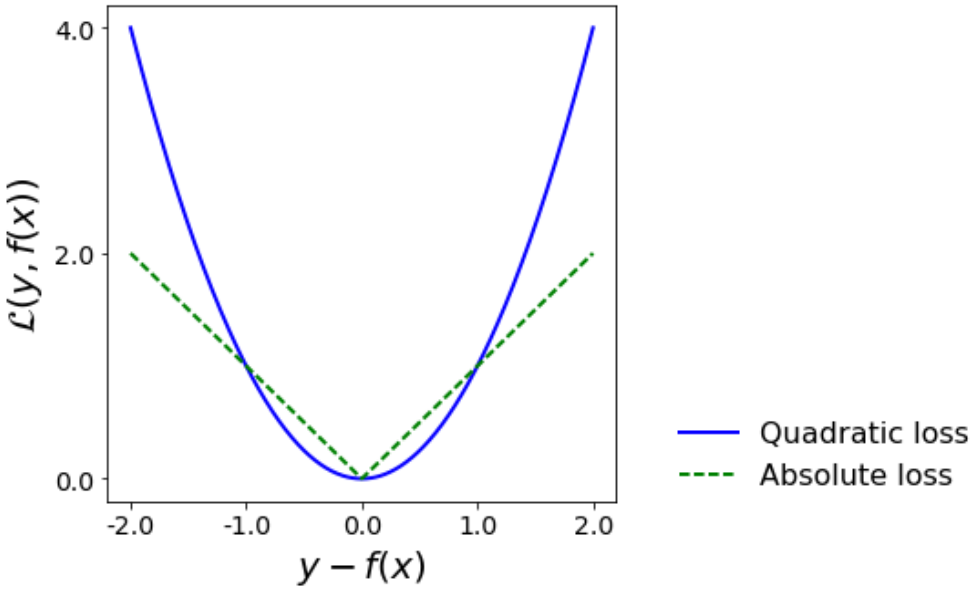
Why Optimization? (2/4)

- Empirical risk minimization:

$$f^* = \arg\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(y^i, f(\mathbf{x}^i))$$

Which loss functions are mainly used?

- Absolute loss: $\mathcal{L}(y, f(\mathbf{x})) = |y - f(\mathbf{x})|$



Why Optimization? (3/4)

- Empirical risk minimization:

$$f^* = \arg\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(y^i, f(\mathbf{x}^i))$$

Which loss functions are mainly used?

- 0/1 loss:

$$\mathcal{L}(y, f(\mathbf{x})) = \begin{cases} 0 & \text{if } y = f(\mathbf{x}) \\ 1 & \text{otherwise} \end{cases}$$

Why Optimization? (4/4)

- Unsupervised machine learning also involves optimization algorithms
- Some examples
  - Dimensionality reduction: find a set of n features, n<p, such that the data projected on these n features retains maximal information (e.g., PCA maximizes the total variance of the data)
  - Clustering: find k groups of samples such that the between-groups variance is high and the within-group variance is small

Both topics will be covered in next lectures

Convex Set

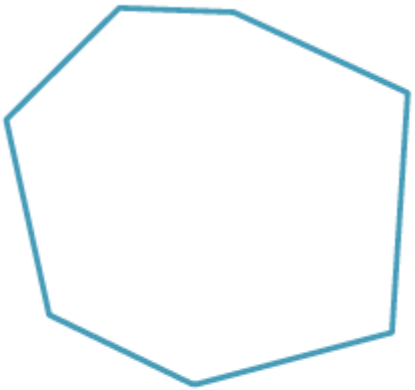
- $\mathcal{S} \subseteq \mathbb{R}^n$ is a convex set iff:

$$t\mathbf{u} + (1-t)\mathbf{v} \in S$$

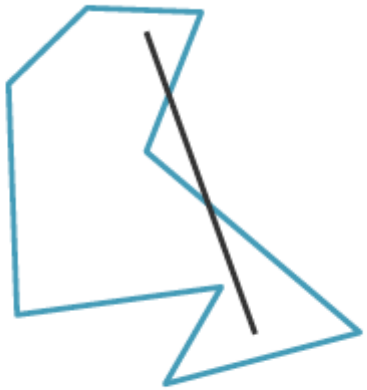
convex combination

for all u,v in S and $0 \leq t \leq 1$

- Line segments between two points of S lie entirely in S



Convex set



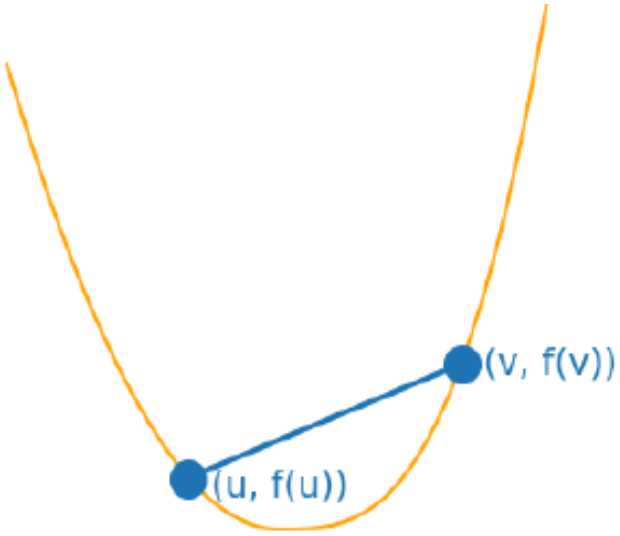
Non-convex set

Convex Function

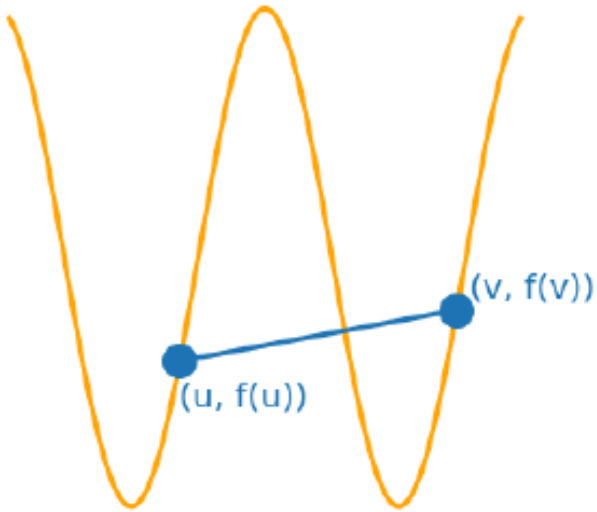
- $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is convex iff:

  1. Its domain is a convex set
  2. $f(t\mathbf{u} + (1-t)\mathbf{v}) \leq tf(\mathbf{u}) + (1-t)f(\mathbf{v})$ for all u,v in dom(f) and $0 \leq t \leq 1$

- f lies below the line segment joining f(u) and f(v)



Convex function



Non-convex function

Concave Function

- $f : \mathbb{R}^n \to \mathbb{R}$ is concave iff:

1. Its domain is a convex set
2. $f(t\mathbf{u} + (1-t)\mathbf{v}) \geq t f(\mathbf{u}) + (1-t) f(\mathbf{v})$ for all u,v in dom(f) and $0 \leq t \leq 1$

f concave ⇔ -f convex

First-order Characterization of Convex Functions

- If f is differentiable, then f is convex if and only if:
  - Its domain is a convex set
  - For all $\mathbf{u}, \mathbf{v} \in \mathrm{dom}(f)$

First-order Taylor expansion

$$f(\mathbf{v}) \geq f(\mathbf{u}) + \nabla f(\mathbf{u})^T(\mathbf{v} - \mathbf{u})$$

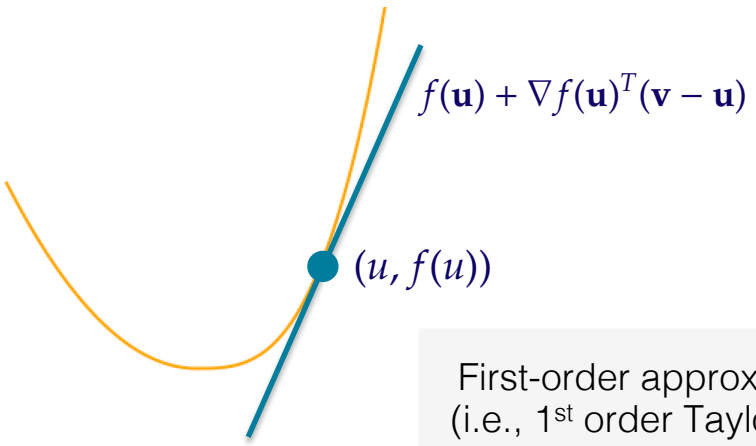
$$\nabla f(\mathbf{u}) = \begin{pmatrix} \frac{\partial f}{\partial u_1} \\ \frac{\partial f}{\partial u_2} \\ \cdots \\ \frac{\partial f}{\partial u_n} \end{pmatrix}$$

Gradient of f

What does it mean if:

$$\nabla f(\mathbf{u}) = 0$$

$\mathbf{u}$ minimizes f



First-order approximation of f (i.e., 1st order Taylor exp.) is a global under estimator

Second-order Characterization

- If f is twice-differentiable, then f is convex if and only if:
  - Its domain is a convex set
  - For all $\mathbf{u}, \mathbf{v} \in \mathbf{dom}(f)$

$$\nabla^2 f(\mathbf{u}) \geq 0$$

Hessian of f, H_f

$$\nabla^2 f = \begin{pmatrix} \frac{\partial^2 f}{\partial u_1^2} & \frac{\partial^2 f}{\partial u_1 u_2} & \cdots & \frac{\partial^2 f}{\partial u_1 u_n} \\ \frac{\partial^2 f}{\partial u_2 u_1} & \frac{\partial^2 f}{\partial u_2^2} & \cdots & \frac{\partial^2 f}{\partial u_2 u_n} \\ \cdots & & & \\ \frac{\partial^2 f}{\partial u_n u_1} & \frac{\partial^2 f}{\partial u_n u_2} & \cdots & \frac{\partial^2 f}{\partial u_n^2} \end{pmatrix}$$

- Let f be a twice-differentiable function at u
  - If $\nabla^2 f(\mathbf{u}) = H_f(\mathbf{u})$ is positive definite, then u is a local minimum
  - If $\nabla^2 f(\mathbf{u}) = H_f(\mathbf{u})$ is negative definite, then u is a local maximum

- Let $S \subseteq \mathbb{R}^n$ be convex, and f be twice-differentiable on S
  - If $H_f(\mathbf{u})$ is positive semi-definite for any $\mathbf{u} \in S$, then f is convex on S

Operations Preserving Convexity

- Non-negative linear combination

  If $f_1, f_2, \ldots, f_m$ convex and $\alpha_1, \alpha_2, \ldots, \alpha_m \geq 0$

  then $\alpha_1 f_1 + \alpha_2 f_2 + \cdots \alpha_m f_m$ is also convex

- Pointwise maximization

  If $f_1, f_2, \ldots, f_m$ are convex

  then $f(\mathbf{u}) = \max\{f_1(\mathbf{u}), \ldots, f_m(\mathbf{u})\}$ is also convex

- Composition with scalar functions

  Composition of $g : \mathbb{R}^n \rightarrow \mathbb{R}$ and $h : \mathbb{R} \rightarrow \mathbb{R}$

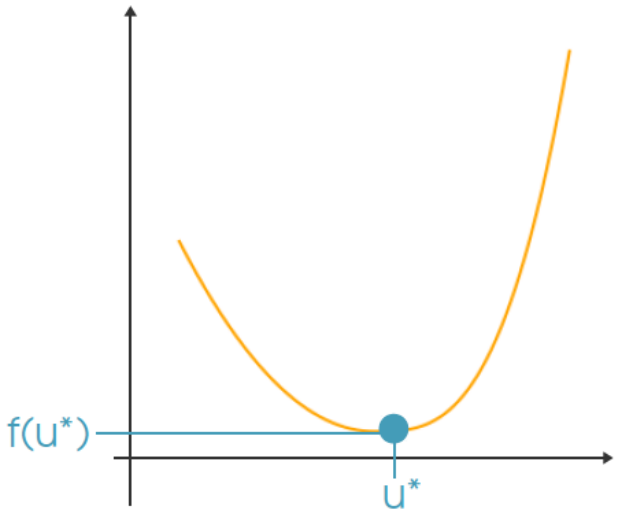
$$f(\mathbf{u}) = h(g(\mathbf{u}))$$

  f is convex if g is convex and h is convex

Unconstrained Convex Optimization

$$\min_{\mathbf{u} \in \mathrm{dom}(f)} f(\mathbf{u})$$

where f is convex



Any point u* that satisfies $\nabla f(\mathbf{u}^*) = 0$ is a global minimum

- From the first-order characterization:

$$f(\mathbf{v}) \geq f(\mathbf{u}^*) + \nabla f^T(\mathbf{u}^*)(\mathbf{v} - \mathbf{u}^*), \forall \mathbf{v}$$

- Since $\nabla f(\mathbf{u}^*) = 0$, we get:

$$f(\mathbf{v}) \geq f(\mathbf{u}^*), \forall \mathbf{v}$$

Constrained Convex Optimization (1/4)

$$\min_{\mathbf{u} \in D} f(\mathbf{u})$$

$$\text{subject to} \quad g_i(\mathbf{u}) \leq 0, i = 1, \ldots, m$$

$$h_i(\mathbf{u}) = 0, i = 1, \ldots, r$$

- f is convex
- $g_i, i = 1 \ldots, m$ are convex [inequality constraints]
- $h_i, i = 1 \ldots, r$ are affine: $h_j : \mathbf{u} \rightarrow \mathbf{a}_j^T \mathbf{u} + b_j$ [equality constraints]
- D is the common domain of all functions

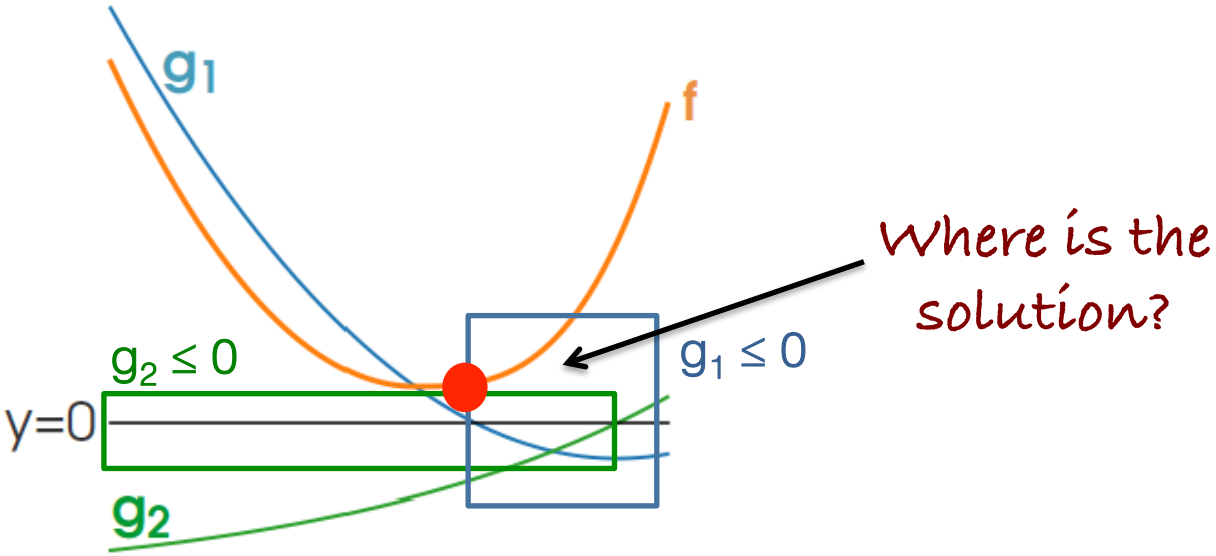
$$D = \text{dom}(f) \cap \bigcap_{i=1}^{m} \text{dom}(g_i) \cap \bigcap_{i=1}^{r} \text{dom}(h_j)$$

Constrained Convex Optimization (2/4)

$$\min_{\mathbf{u} \in D} f(\mathbf{u})$$

$$\text{subject to} \quad g_i(\mathbf{u}) \leq 0, i = 1, \ldots, m$$

$$h_i(\mathbf{u}) = 0, i = 1, \ldots, r$$



Constrained Convex Optimization (3/4)

$$\min_{\mathbf{u} \in D} f(\mathbf{u})$$

$$\text{subject to} \quad g_i(\mathbf{u}) \leq 0, i = 1, \ldots, m$$

$$h_i(\mathbf{u}) = 0, i = 1, \ldots, r$$

- f is the objective function
- $g_i, i = 1 \ldots, m$ are the inequality constraints
- $h_i, i = 1 \ldots, r$ are the equality constraints
- $\mathbf{u} \in D$ that verifies all constraints is a feasible point

$$g_i(\mathbf{u}) \leq 0, i = 1, \ldots, m \text{ and } h_j(\mathbf{u}) = 0, j = 1, \ldots, r$$

- The set of all feasible points is the feasible region

$$\{\mathbf{u} : \mathbf{u} \in D; g_i(\mathbf{u}) \leq 0, i = 1, \ldots, m; \; h_j(\mathbf{u}) = 0, j = 1, \ldots, r\}$$

Constrained Convex Optimization (4/4)

$$\min_{\mathbf{u} \in D} f(\mathbf{u})$$

$$\text{subject to} \quad g_i(\mathbf{u}) \leq 0, i = 1, \ldots, m$$

$$h_i(\mathbf{u}) = 0, i = 1, \ldots, r$$

- Assuming it exists, the solution f*, that is to say the minimum value of f over all feasible points, is the optimal value (optimum)
- A u feasible such that f(u) = f* is called optimal or a minimizer (it doesn't need to be unique)

Local and Global Minima

For convex optimization problems

$$\min_{\mathbf{u} \in D} f(\mathbf{u})$$
$$\text{subject to} \quad g_i(\mathbf{u}) \leq 0, i = 1, \ldots, m$$
$$h_i(\mathbf{u}) = 0, i = 1, \ldots, r$$

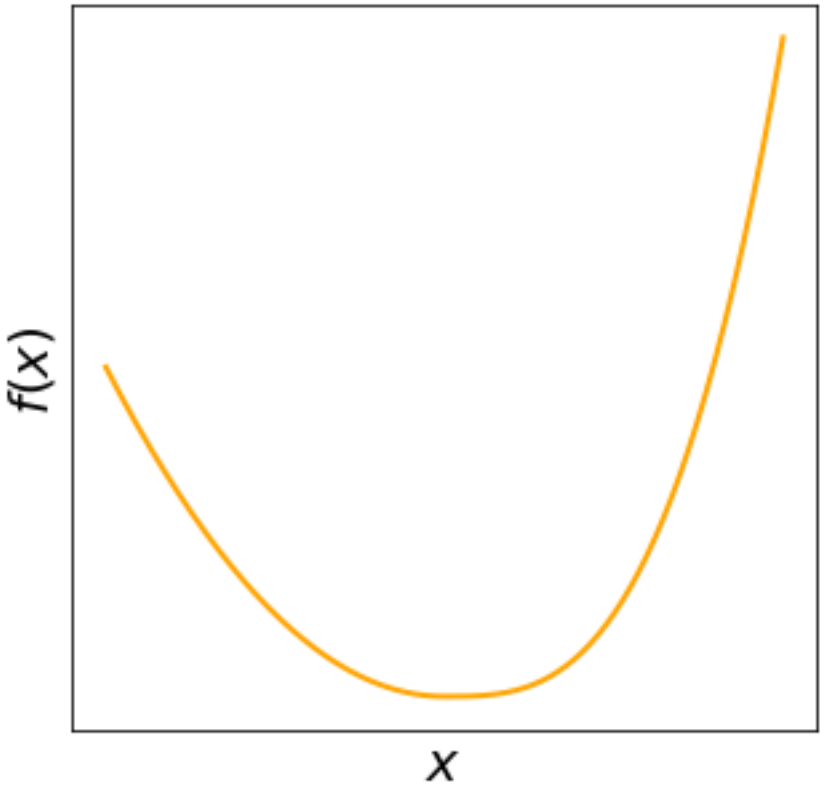
Local minima are global minima

If u is a feasible solution and minimizes f in a local neighborhood:

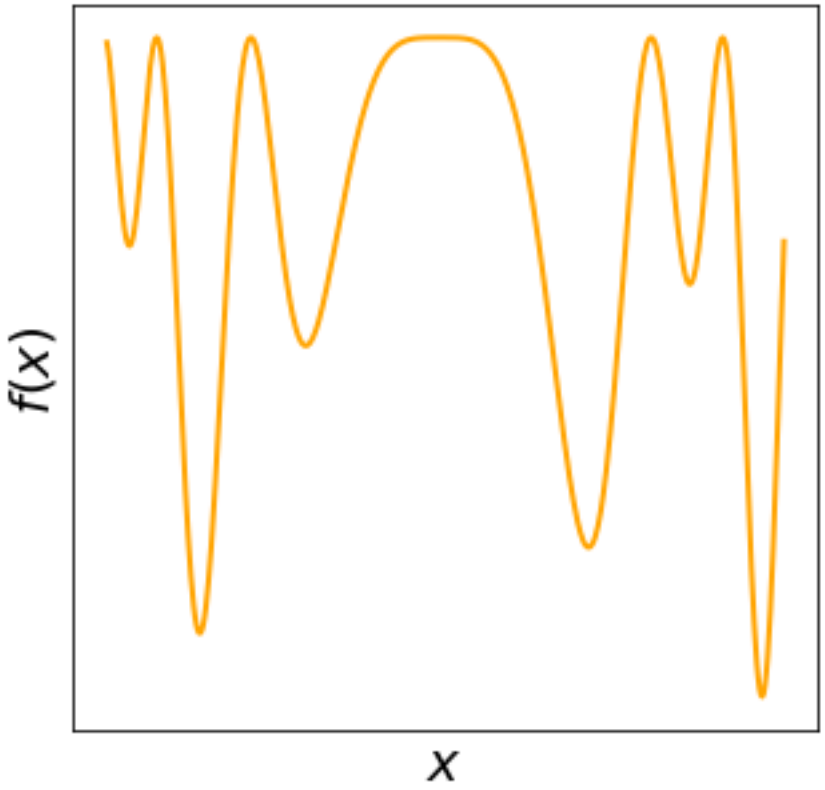
f(u) <= f(v) for all feasible v,

then u minimizes f globally

Why Talk about Convex Optimization?



Convex function



Non-convex function

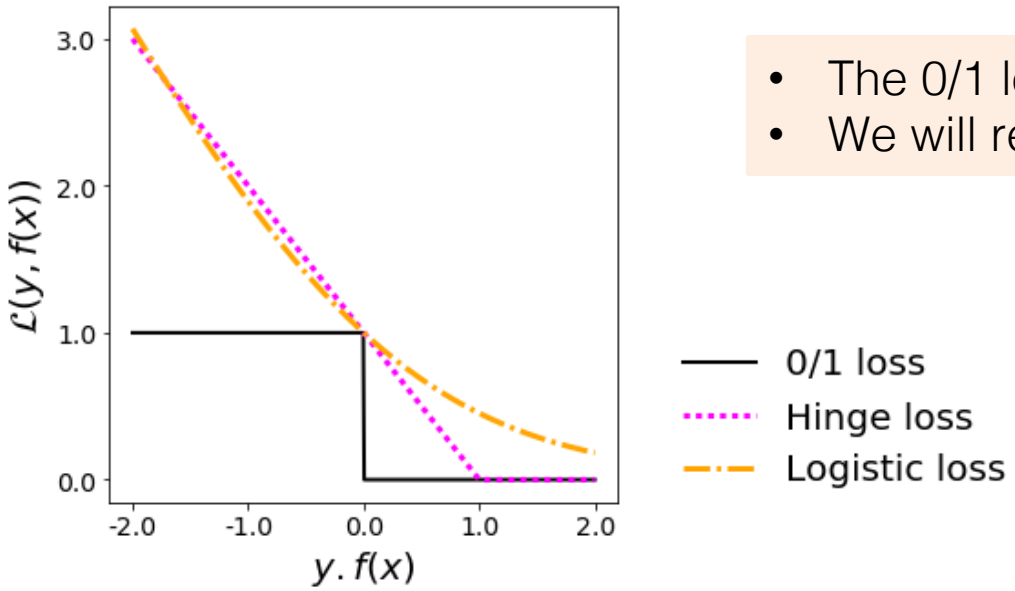
- Convex optimization is easy
- We will often try to formulate ML problems as convex optimization problems

Back to the ERM Problem

- Empirical risk minimization:

$$f^* = \arg\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(y^i, f(\mathbf{x}^i))$$

- Loss functions for classification



- The 0/1 loss function is non-convex
- We will replace it with other losses

Unconstrained Convex Optimization

- Suppose that **f** is differentiable
- Given the first-order characterization of convex functions, how can we solve $\min_{\mathbf{u} \in \text{dom}(f)} f(\mathbf{u})$?

Recall that:

$$\nabla f(\mathbf{u}) = 0$$

$$\downarrow$$

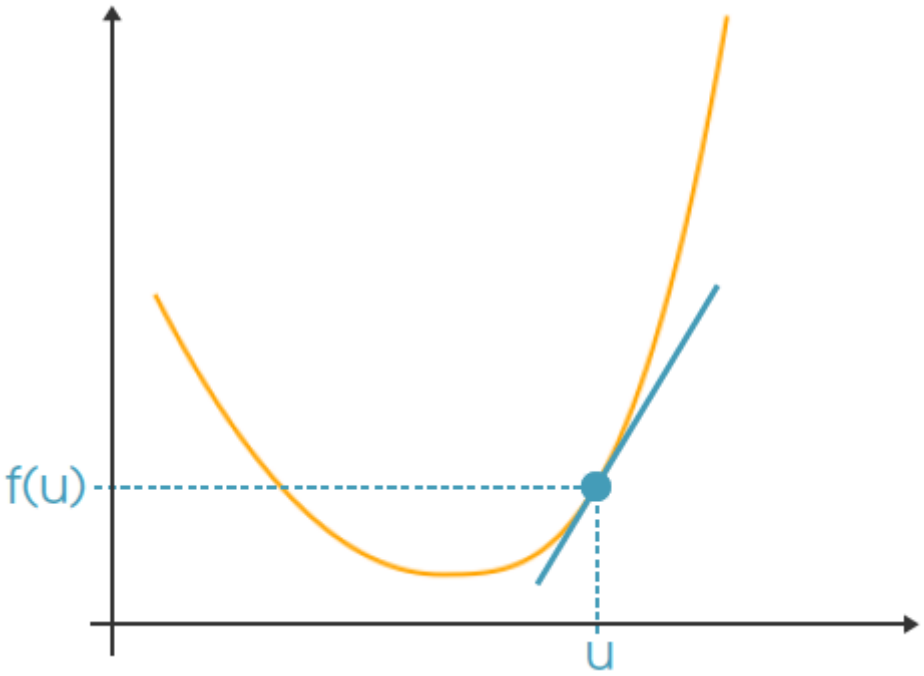
$$\mathbf{u} \text{ minimizes } f$$

→ Set the gradient of **f** to 0

But, what if $\nabla f(\mathbf{u}) = 0$ cannot be solved?

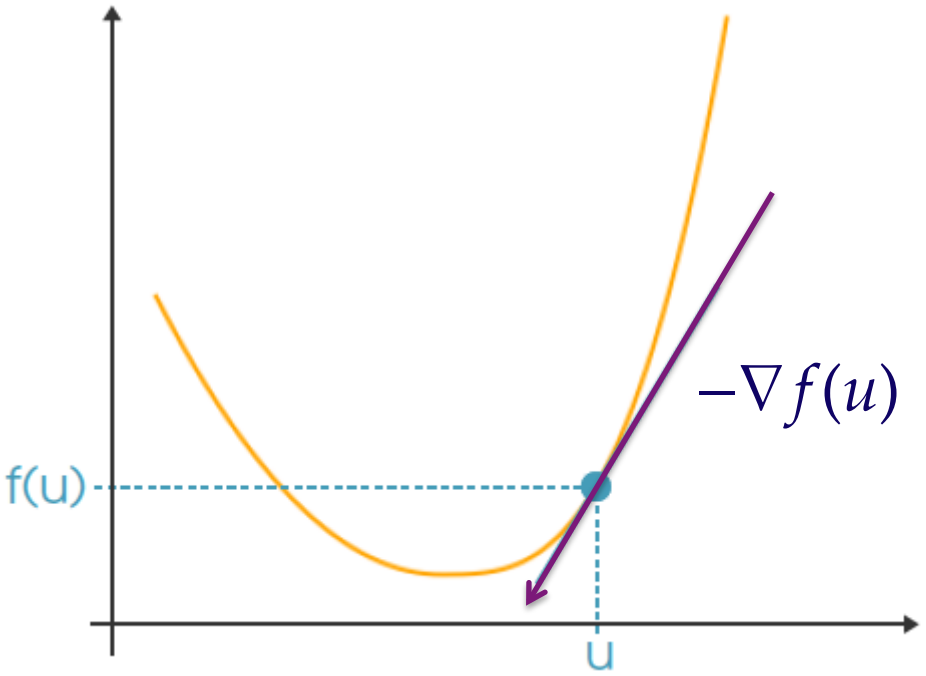
Gradient Descent – The Idea (1/3)

- Start from a random point u
- How do I get closer to the solution?



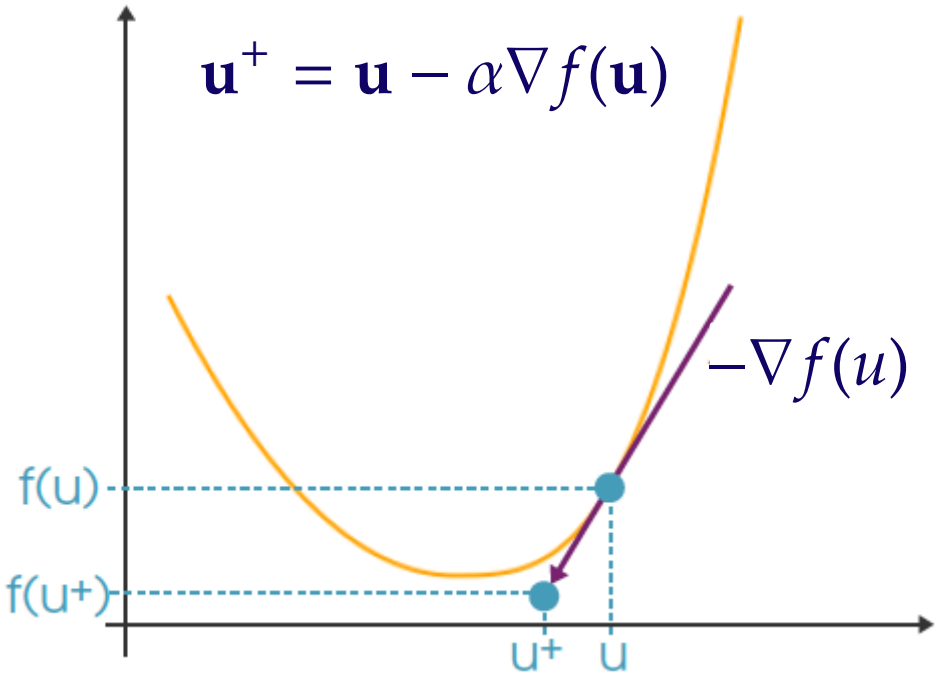
Gradient Descent – The Idea (2/3)

- Start from a random point u
- How do I get closer to the solution?
- Follow the opposite of the gradient
  - The gradient indicates the direction of the steepest descent



Gradient Descent – The Idea (3/3)

- Start from a random point u
- How do I get closer to the solution?
- Follow the opposite of the gradient
  - The gradient indicates the direction of the steepest descent



Gradient Descent Algorithm

- Choose an initial point $\mathbf{u}^{(0)} \in \mathbb{R}^n$
- Repeat for k=1, 2, 3, …

$$\mathbf{u}^{(k)} = \mathbf{u}^{(k-1)} - \alpha_k \nabla f(\mathbf{u}^{(k-1)})$$

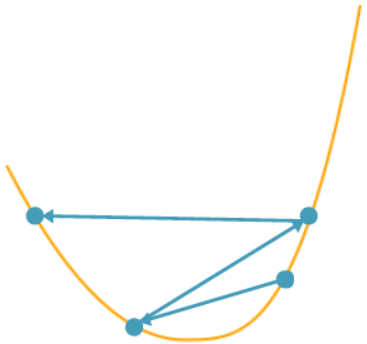
step size

- Stop at some point

Stopping criterion: Usually when $\|\nabla f(\mathbf{u}^{(k)}\|_2 < \epsilon, \quad \epsilon = \text{small constant}$

If the step is too big, the search might diverge

If the step is too small, the search might take very long time



Supervised Learning: 3 Ingredients

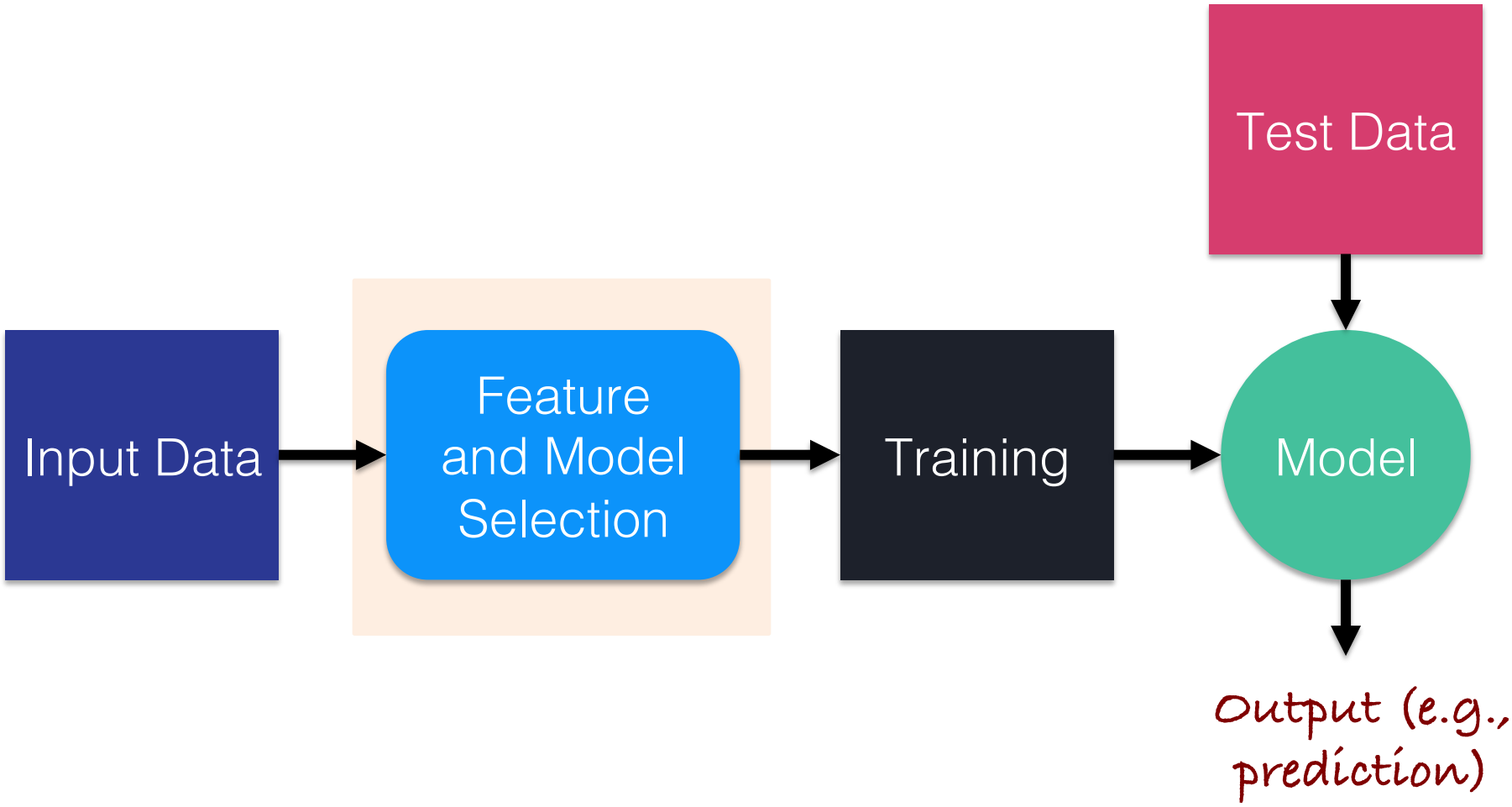
Given $\mathcal{D} = \{x^i, y^i\}_{i=1,\ldots,n}$ find f such that $f(x) \approx y$

- Chose a hypothesis class $\mathcal{F}$
  - Parametric methods – e.g., $f(x) = \sum_{j=1}^{p} \beta_j x_j$
  - Non-parametric methods – e.g., f(x) is the label of the point closest to x (Nearest Neighbors is such a method)
- Chose a loss function $\mathcal{L}$
  - Empirical error: $\frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(y^i, f(\mathbf{x}^i))$
- Chose an optimization procedure

$$f^* = \arg\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(y^i, f(\mathbf{x}^i))$$

Model selection and evaluation

ML Pipeline


Test Data
Input Data
Feature and Model Selection
Training
Model
Output (e.g., prediction)

Generalization

- It's easy to build a model that performs well on the training data
- But how well will it perform on new (unseen) data?

- Learn models that generalize well
- Evaluate whether models generalized well

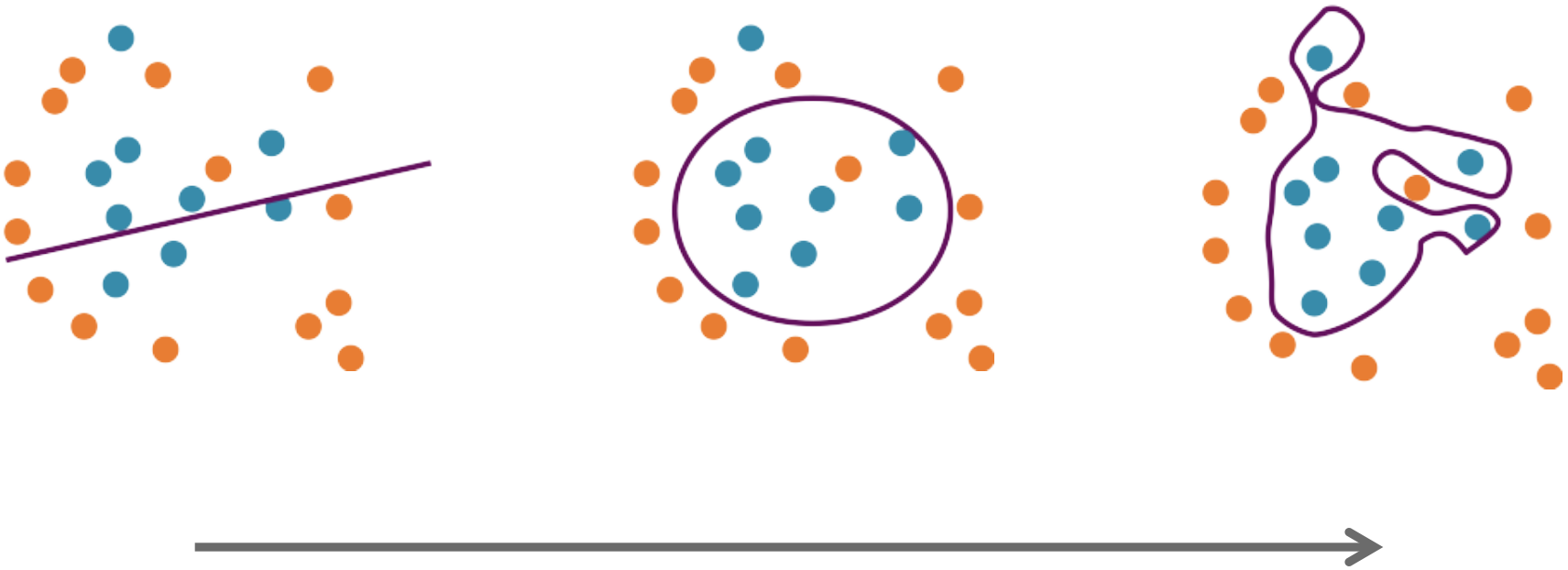
Given $\mathcal{D} = \{x^i, y^i\}_{i=1,\ldots,n}$ find **f** such that $f(x) \approx y$

Additional Factor: Noise in the Data

- Imprecision in recording the features
- Errors in labeling the data points
- Missing features (hidden or latent)

Making no errors on the training set might not be possible

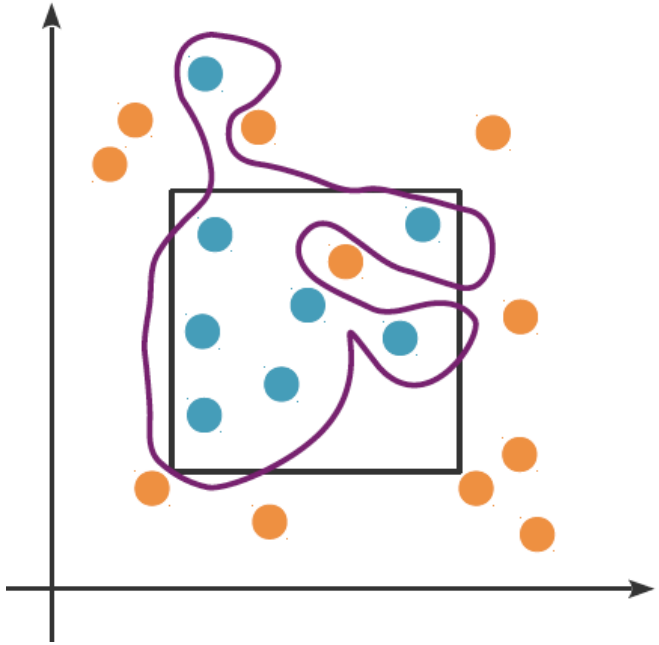
Models of Increasing Complexity



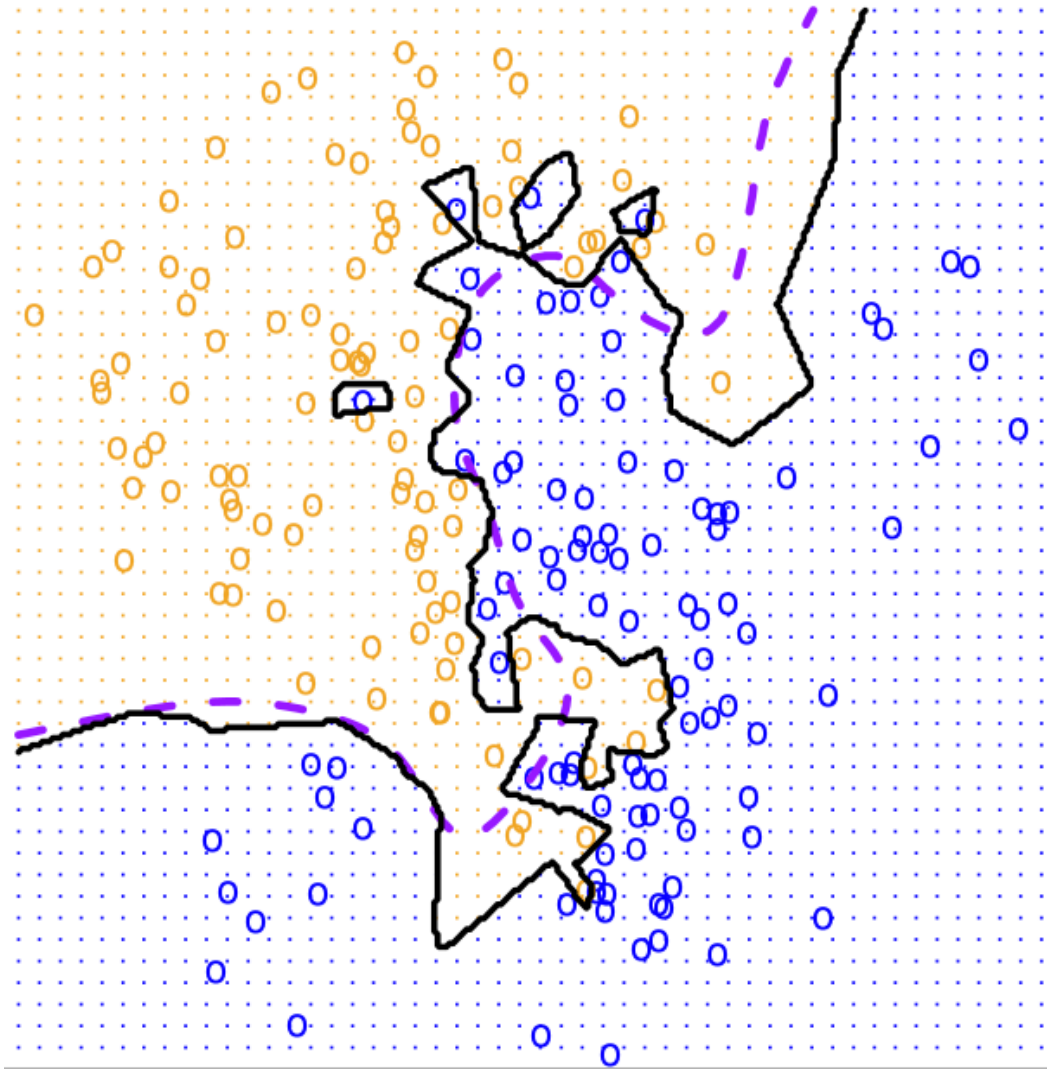
Model complexity

Noise and Model Complexity

- Use simple models!
  - Easier to use
    - Lower computational complexity
  - Easier to train
    - Lower space complexity
  - Easier to explain
    - More interpretable
  - Generalize better
    - Occam's razor: simpler explanations are more plausible



Overfitting



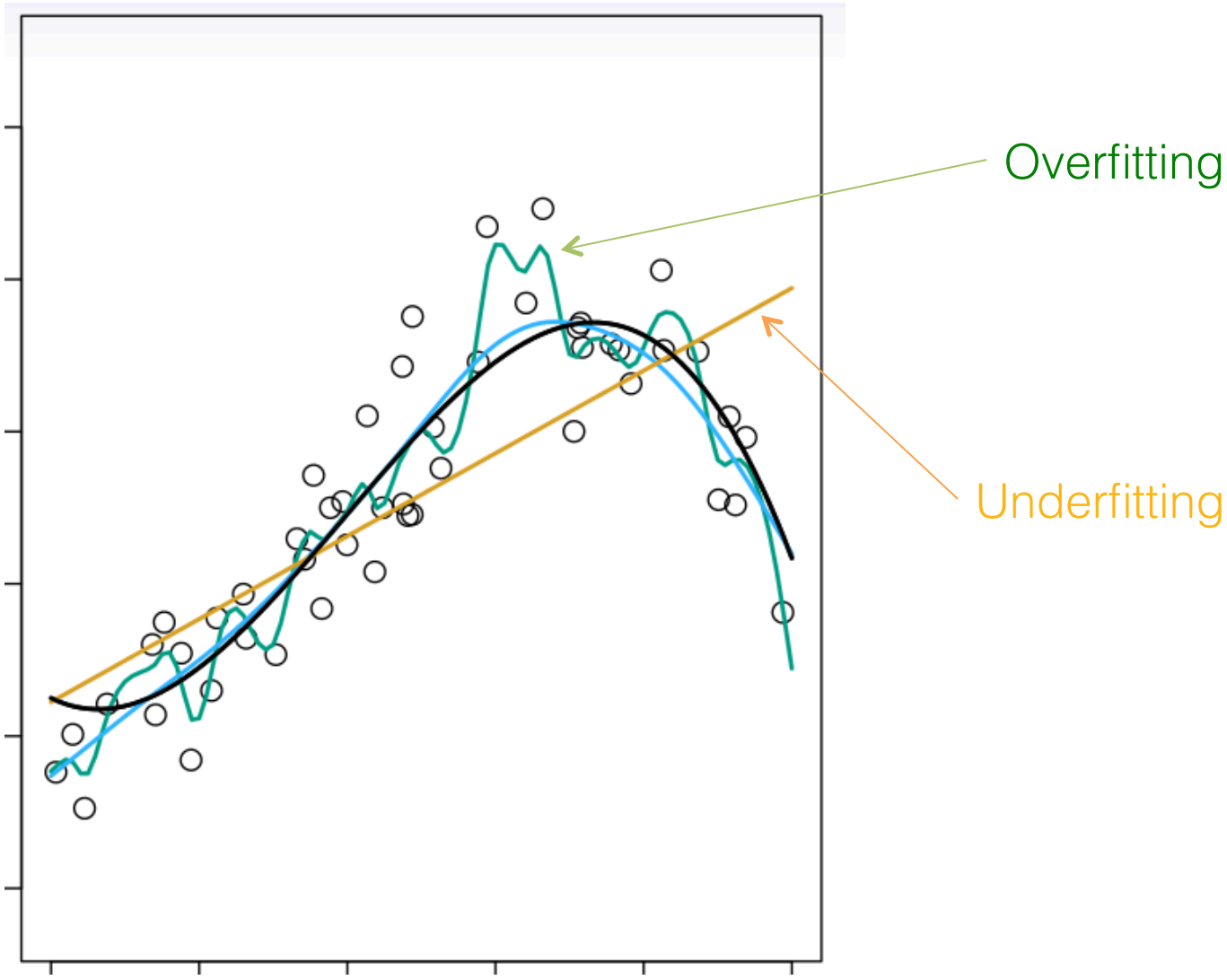
- What is the empirical error of the **black** and **purple** classifiers?

$$\frac{1}{n}\sum_{i=1}^{n}\mathcal{L}(y^i, f(\mathbf{x}^i))$$

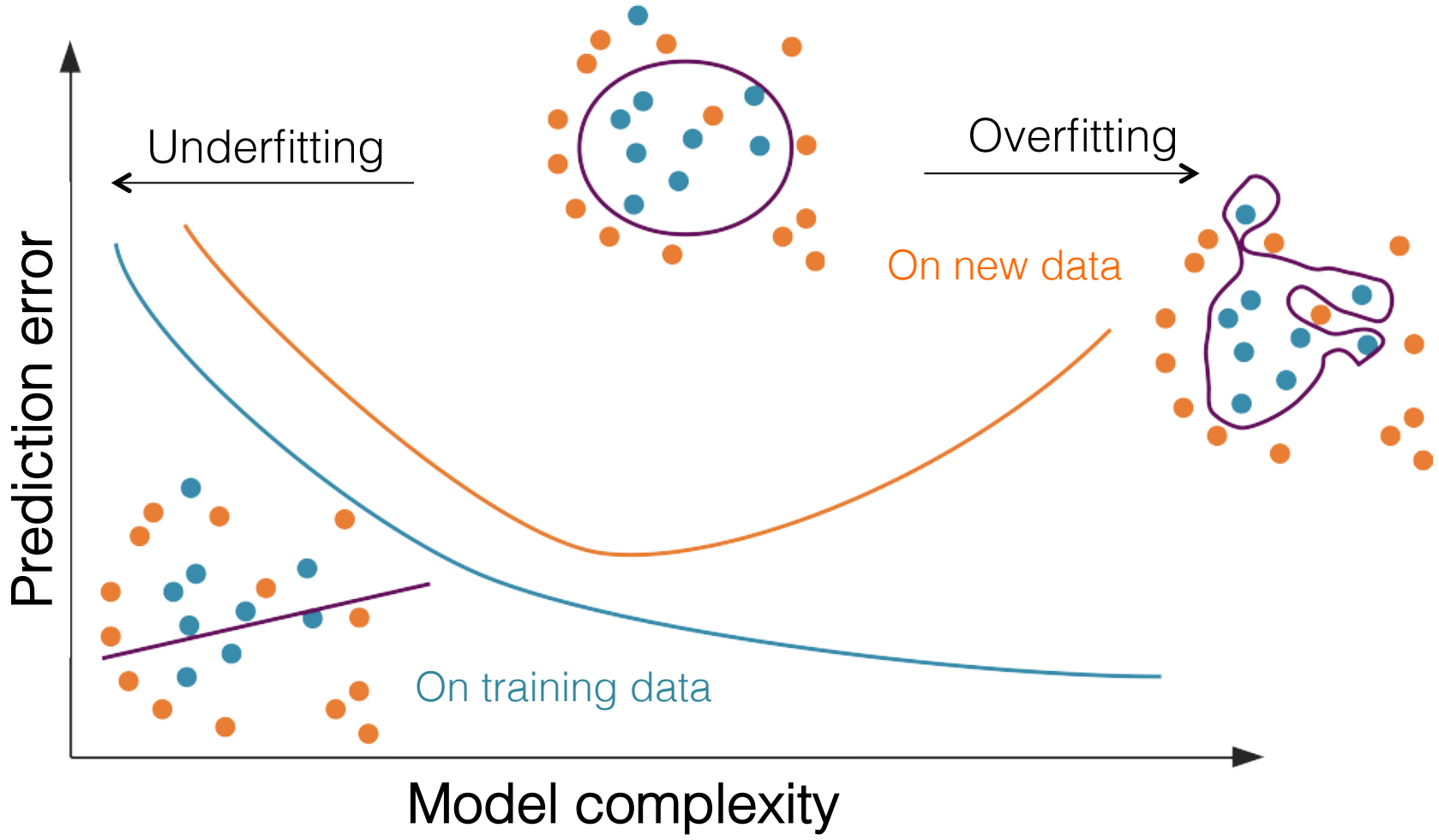
For some loss function $\mathcal{L}$

- Which model seems more likely to be correct?

Overfitting and Underfitting in Regression



Generalization Error vs. Model Complexity



Fixed dataset size; varying model complexity

Bias-Variance Tradeoff (1/5)

- Bias: difference between the expected value of the estimator (model) and the true value being estimated

$$\text{Bias}(f(\mathbf{x})) = \mathbb{E}[f(\mathbf{x}) - y]$$

  - A simpler model has a higher bias (naturally a simple model will do some errors)
  - High bias can cause underfitting

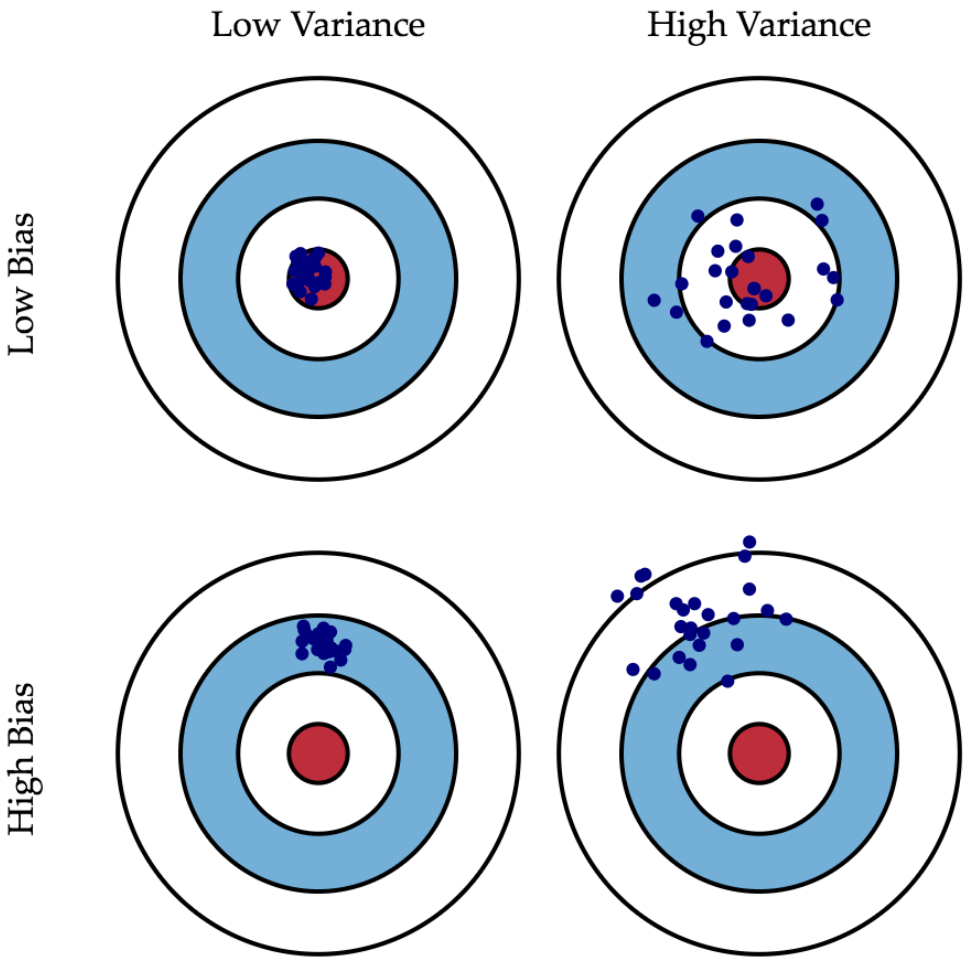
- Variance: deviation from the expected value of the estimates

$$\text{Var}(f(\mathbf{x})) = \mathbb{E}[(f(\mathbf{x}) - \mathbb{E}(f(\mathbf{x}))^2]$$

  - A more complex model has a higher variance
  - High variance can cause overfitting

Ideally, we want to optimize both

Bias-Variance Tradeoff (2/5)



- The center of the target is a model that perfectly predicts the correct values
- We can repeat our entire model building process to get a number of separate hits on the target
  - Each hit represents an individual realization of the model
- Bias measures how far are in general these models' predictions are from the correct value
- Variance is how much the predictions for a given point vary between different realizations of the model

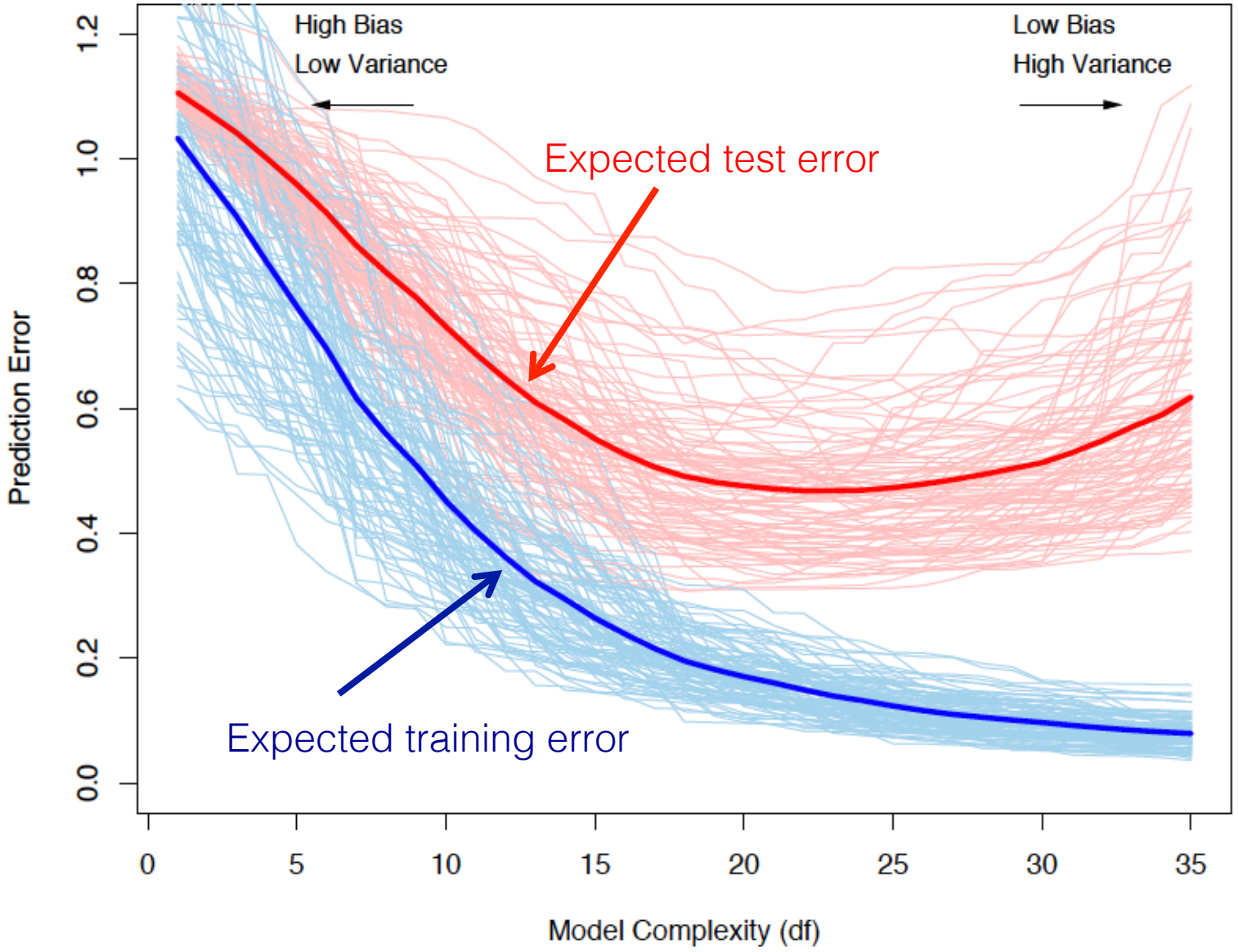
Source: http://scott.fortmann-roe.com/docs/BiasVariance.html

Bias-Variance Tradeoff (3/5)

- When do we have high bias?
  - We have high bias when the model (function) cannot model the true data distribution well
  - This doesn't depend on the training data size
  - Underfitting

- When do we have high variance?
  - We have high variance when there is a small amount of training data and a very complex model
  - Overfitting
  - Variance decreases with larger training data, and increases with more complicated classifiers

Bias-Variance Tradeoff (4/5)



Bias-Variance Tradeoff (5/5)

- High bias → high training and test errors
- High variance → low training error, high test errors

Bias-variance tradeoff

If we make the model more complicated, then, for the same training set size, the bias decreases but the variance increases

Bias-Variance Decomposition

- $\text{Bias}(f(\mathbf{x})) = \mathbb{E}[f(\mathbf{x}) - y]$
- $\text{Var}(f(\mathbf{x})) = \mathbb{E}[(f(\mathbf{x}) - \mathbb{E}(f(\mathbf{x}))^2]$

We want to find a function f(x) that approximates y as well as possible

- Use the mean squared error: we want $(f(x) - y)^2$ to be minimized, both for $x_1, \ldots, x_m$ and for points outside the sample
  - f should generalize to samples outside the training set
- We can decompose the expected error on an unseen example x as follows:

$$\begin{aligned}\text{MSE}(f(\mathbf{x})) &= \mathbb{E}[(f(\mathbf{x}) - y)^2] \\ &= \text{Var}(f(\mathbf{x})) + \text{Bias}^2(f(\mathbf{x}))\end{aligned}$$

How much your model changes if you train on a different training set

What is the inherent error that you obtain from your model even with infinite training data

Model Selection and Evaluation

- Model selection: estimating the performance of different models in order to choose the best

- Model evaluation (assessment): having chosen a final model, estimating its prediction error (generalization error) on new data

How to estimate if a model is good in practice

or

How do we estimate the prediction error

Learning Objectives

- After this part of the lecture you will be able to design experiments to select and evaluate supervised machine learning models

- Concepts:
  - Training and testing sets
  - Cross-validation
  - Measures of performance for classification and regression
  - Measures of model complexity

Reminder – Supervised Learning Setting

Given $\mathcal{D} = \{x^i, y^i\}_{i=1,\ldots,n}$ find f such that $f(x) \approx y$

- Empirical error of f on the training set, given a loss function

$$\frac{1}{n}\sum_{i=1}^{n} \mathcal{L}(y^i, f(\mathbf{x}^i))$$

Binary classification

$y^i \in \{0, 1\}$

Multiclass classification

$y^i \in \{0, 1, \ldots, k\}$

Regression

$y^i \in \mathbb{R}$

Generalization Error

- The empirical error on the training set is a poor estimate of the generalization error (expected error on new data)
  - If the model is overfitting, the generalization error can be arbitrarily large

- We would like to estimate the generalization error on new data, which we do not have

Any (simple) idea?

Validation Sets

- Choose the model that performs best on a validation set separate from the training set

- Because we have not used the validation data at any point during training, the validation set can be considered "new data" and the error on the validation set is an estimation of the generalization error

Model Selection

- What if we want to choose among k models?
  - Train each model on the train set
  - Compute the prediction error of each model on the validation set
  - Pick the model with the smallest prediction error on the validation set

- What is the generalization error?
  - We don't know!
  - Validation data was used to select the model
  - We have "cheated" and looked at the validation data: it is not a good proxy for new, unseen data any more

Validation Sets (1/2)

- Hence we need to set aside part of the data, the test set, that remains untouched during the entire procedure and on which we'll estimate the generalization error

- Model selection: pick the best model
- Model assessment: estimate its prediction error on new data

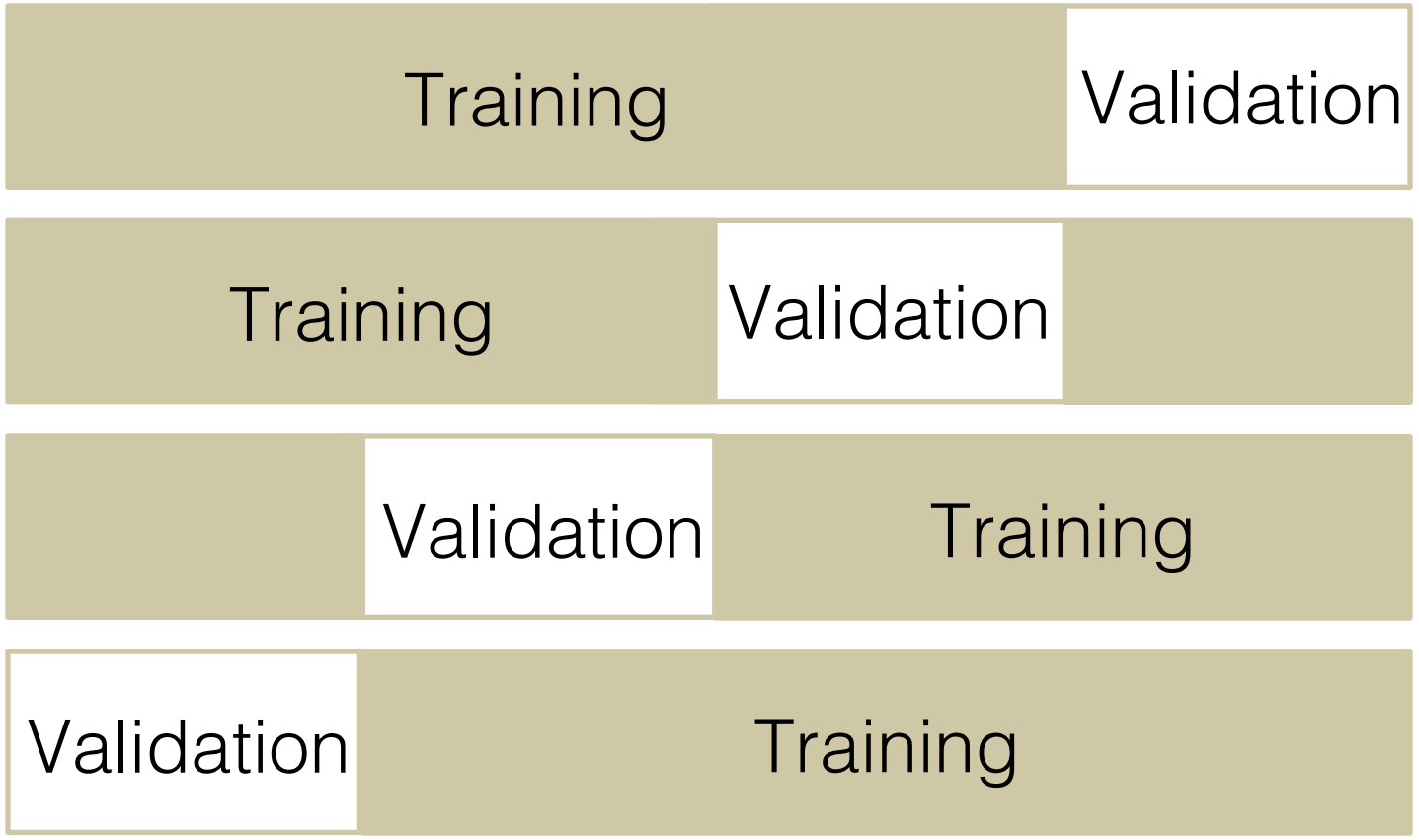
Validation Sets (2/2)

- How much data should go in each of the training, validation and test sets?
- How do we know that we have enough data to evaluate the prediction and generalization errors?

- Empirical evaluation with sample re-use
  - Cross-validation
  - Bootstrap (random sampling with replacement)

Cross-validation

- Cut the training set in k separate folds
- For each fold, train on the (k-1) remaining folds



Cross-validated Performance

- Cross-validation estimate of the prediction error

$$CV(f) = \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}(y^i, f_{k(i)}(\mathbf{x}^i))$$

Computed with the k(i)-th part of the data removed.
k(i) = fold in which i belongs to

Issues with Cross-validation

- The training set can become very small (depending on k)
  - Small training set → biased estimator of the error
- Leave-one-out cross-validation: k = n
  - Approximately unbiased estimator of the expected prediction error
  - Potential high variance (the training sets are very similar to each other)
  - Computationally-intense approach (n repeats)
- In practice: set k=5 or k=10

Cross-validation in scikit-learn

http://scikit-learn.org/stable/modules/cross_validation.html

Evaluating model performance

Classification Model Evaluation

- Confusion matrix

		True class	
		-1	+1
Predicted class	-1	True Negatives	False Negatives
	+1	False Positives	True Positives

- False positives (false alarms) are also called type I errors
- False negatives (misses) are also called type II errors

Evaluation Measures (1/2)

- Sensitivity = Recall = True positive rate (TPR)

$$TPR = \frac{TP}{TP + FN}$$

# of positives

- Specificity = True negative rate (TNR)

$$TNR = \frac{TN}{FP + TN}$$

- Precision = Positive predictive value (PPV)

$$PPV = \frac{TP}{TP + FP}$$

# of predicted positives

- False discovery rate (FDR)

$$FDR = \frac{FP}{FP + TP}$$

Evaluation Measures (2/2)

- Accuracy

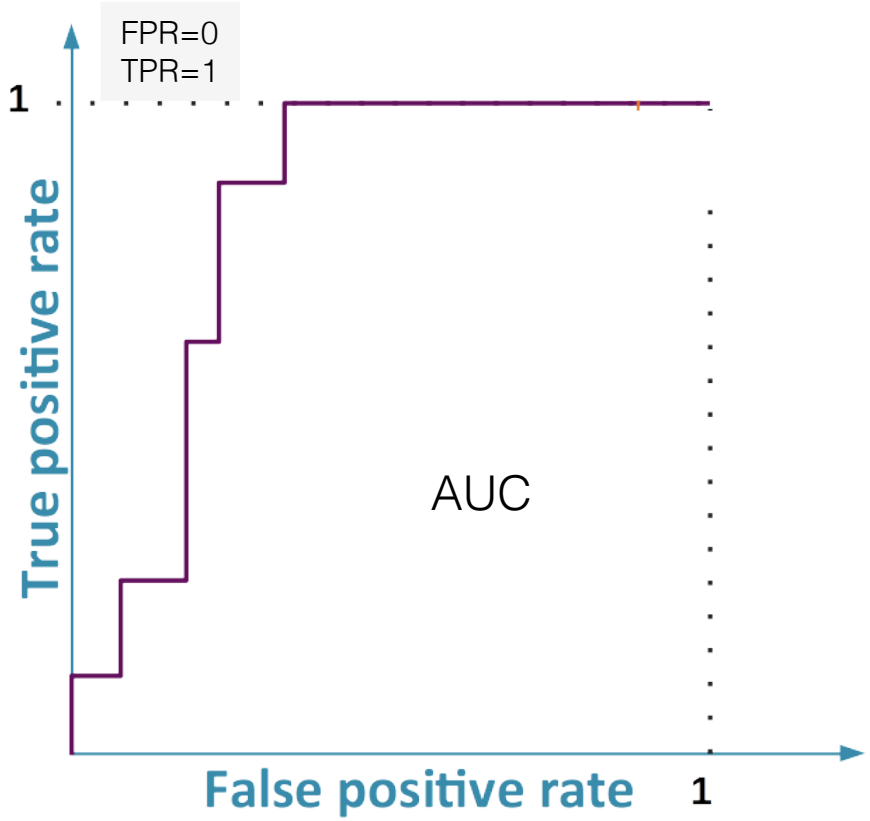
$$Acc = \frac{TP + TN}{TP + FN + FP + TN}$$

- F1-score: harmonic mean of precision and recall (sensitivity)

$$F1 = \frac{2TP}{2TP + FP + FN}$$

ROC Curve (1/2)

- ROC = Receiver-Operator Characteristic
- Summarized by the area under the curve (AUROC or AUC)



- Plot TPR vs. FPR for all possible thresholds (typically generated by the classifier)

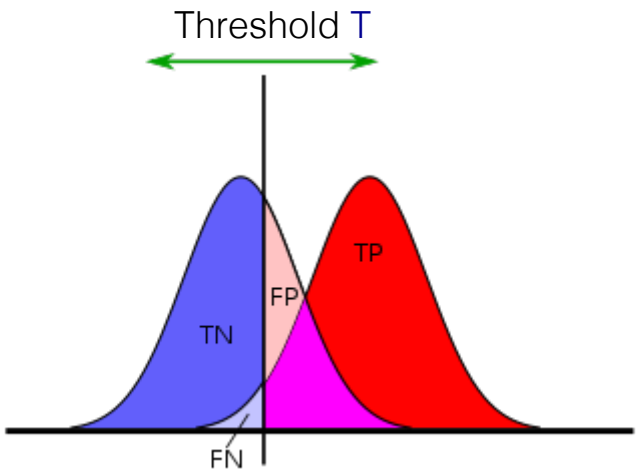
$$TPR = \frac{TP}{TP + FN} \qquad FPR = \frac{FP}{FP + TN}$$

- Shows the trade off in sensitivity (TPR) and (1 – specificity) (FPR) for all possible thresholds (cutoff values between positive and negative class)

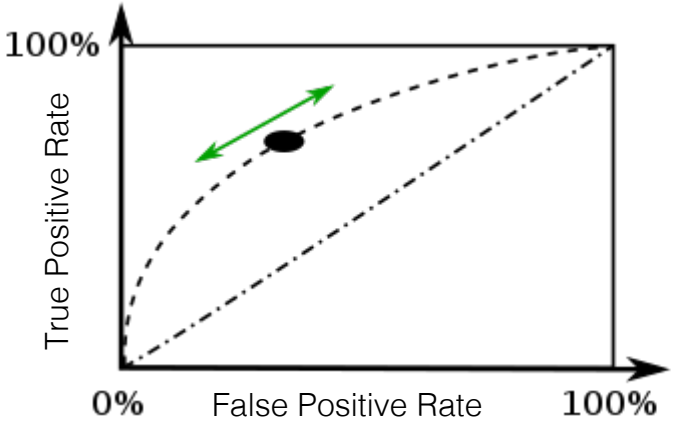
- The larger the AUC, the better is overall performance

Source: https://en.wikipedia.org/wiki/Receiver_operating_characteristic
Visual description of ROC curve: https://www.youtube.com/watch?v=OAl6eAyP-yo

ROC Curve (2/2)



TP	FP
FN	TN



- In binary classification, the class prediction is often made based on a cont. random variable X
  - Given by the classifier
- Given a threshold parameter T, the instance is classified as positive if X>T
- X follows a PDF
  - $f_1(x)$, if it actually belongs to the positive class
  - $f_0(x)$, for the negative class

$$TPR = \frac{TP}{TP + FN} \qquad FPR = \frac{FP}{FP + TN}$$

Demo: https://kennis-research.shinyapps.io/ROC-Curves/

Evaluation Measures in scikit-learn

http://scikit-learn.org/stable/modules/model_evaluation.html

About the project

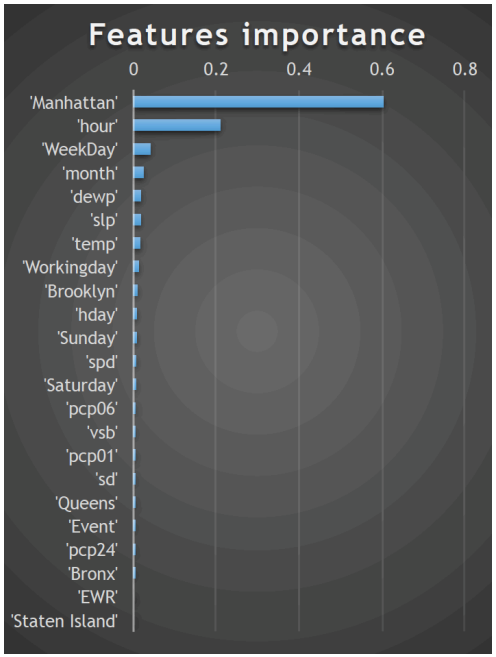
Examples of Previous Projects (1/5)

How can Uber predict the demand for pickups?

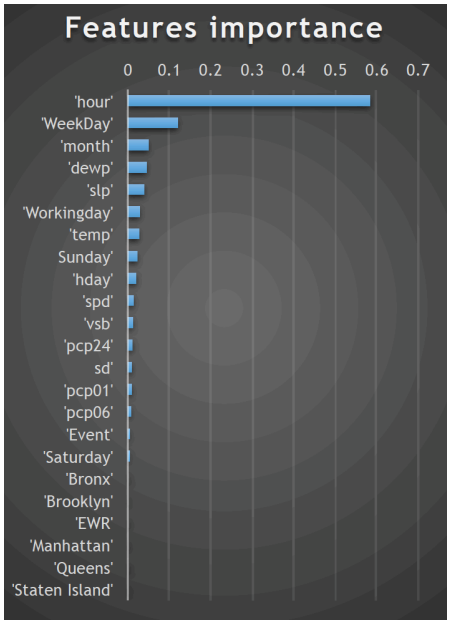
- Dataset: 34,000 rides per day in NYC
- Prediction problem

Weather	Events	Days	Boroughs
Temperature	Concerts	Time	Bronx
Wind Speed	Music Festivals	Day / Month	Brooklyn
Visibility		Holidays	EWR
Precipitation			Manhattan
Sea Level			Queens
Dew Point			Staten Island
Snow Depth			

Dataset description



For the whole dataset

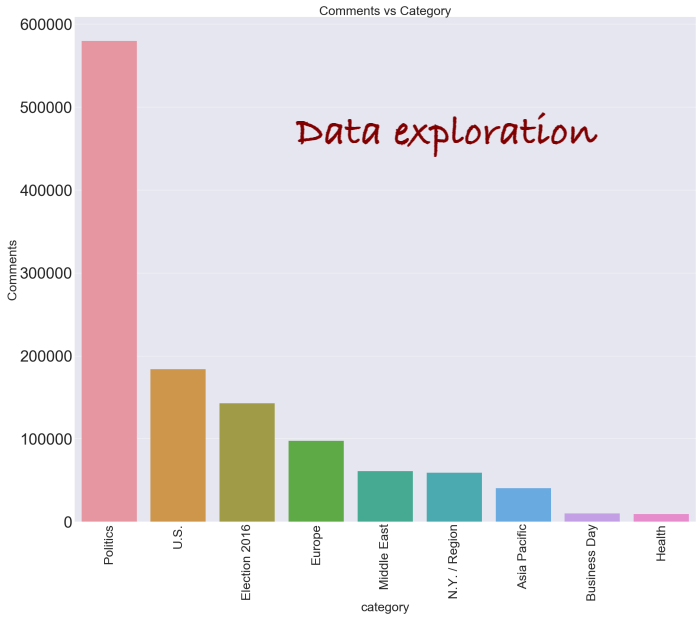


For a particular district in NYC (Borough)

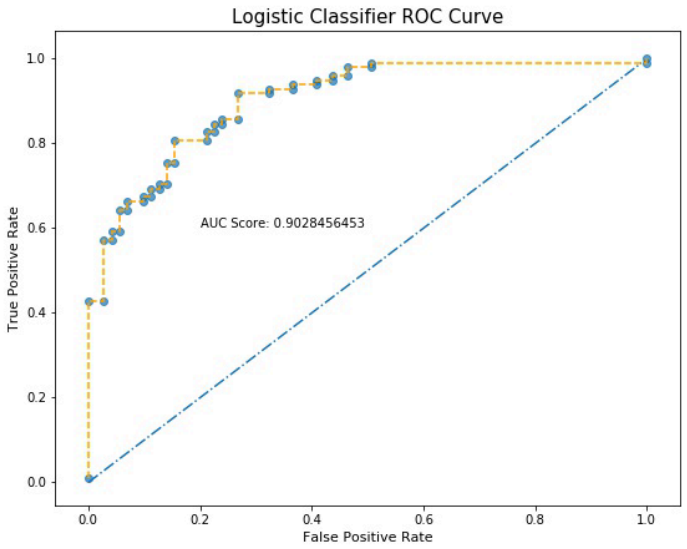
Examples of Previous Projects (2/5)

Analyzing New York Times' readers engagement

- Problem: predict if an article will be commented or not
- Data: 1,280 articles and 280 authors' data
- Features: article length, images, videos, content, authors' information, etc.



Top-10 most commented articles

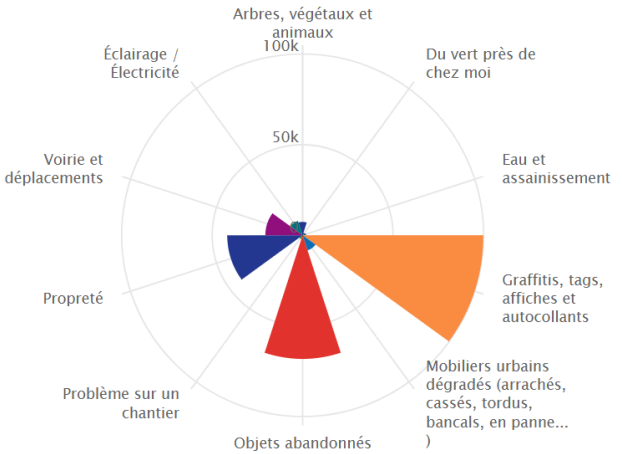


Prediction results (logistic regression)

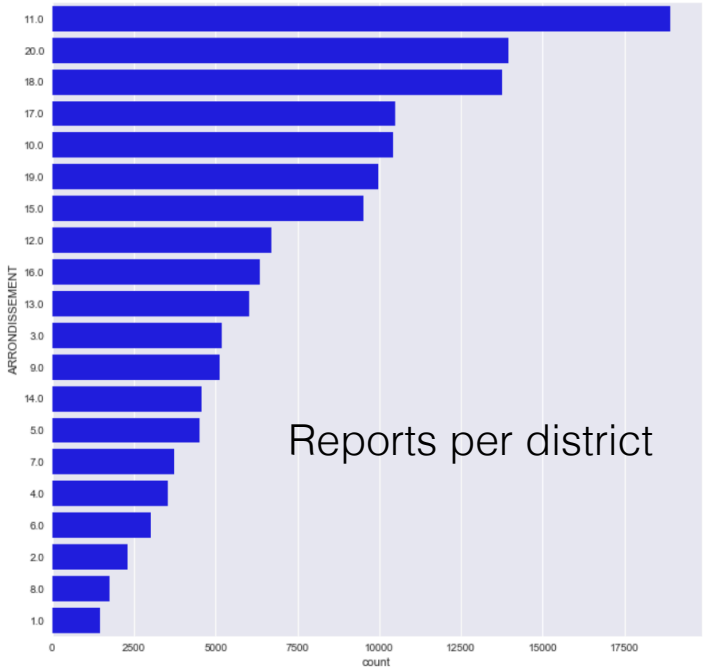
Examples of Previous Projects (3/5)

Predict reports made by citizens in Paris

- Data: open Paris data



Distribution of reports by type



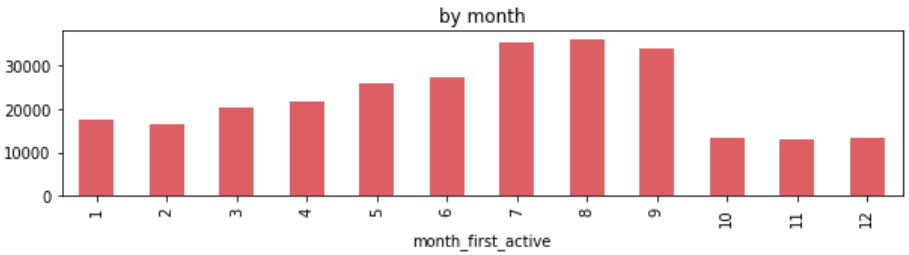
	precision	recall	f1-score	support
Arbres, végétaux et animaux	0.48	0.02	0.03	634
Du vert près de chez moi	0.47	0.15	0.23	277
Eau et assainissement	1.00	0.01	0.02	124
Graffitis, tags, affiches et autocollants	0.54	0.86	0.66	11314
Mobiliers urbains dégradés	0.52	0.05	0.10	1067
Objets abandonnés	0.57	0.58	0.58	7286
Problème sur un chantier	0.00	0.00	0.00	229
Propreté	0.42	0.20	0.27	4120
Voirie et déplacements	0.40	0.08	0.13	2301
Éclairage / Électricité	0.42	0.04	0.07	909
avg / total	0.51	0.54	0.47	28261

Prediction results
(random forest)

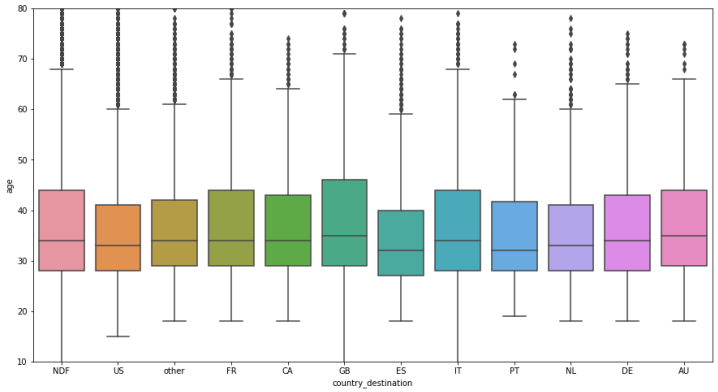
Examples of Previous Projects (4/5)

Airbnb destination prediction

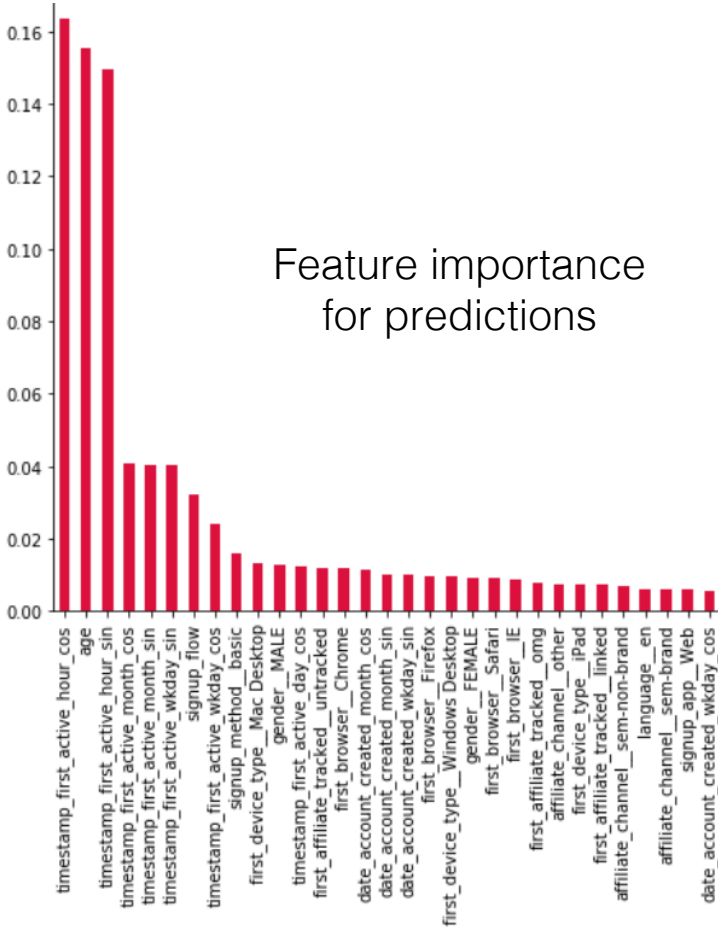
- Data: 214,000 users (USA), 11 countries
- Features: age, gender, language, important dates



Activity per month



Destination per age



Feature importance for predictions

Data exploration

Examples of Previous Projects (5/5)

Pokemon battle prediction

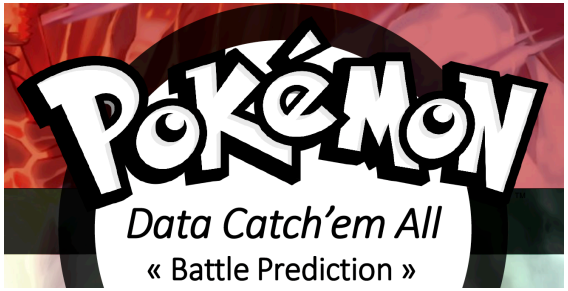
- Problem: predict the outcome of a pokemon fight
- Data:

	First_pokemon	Second_pokemon	Winner
0	266	298	298
1	702	701	701
2	191	668	668
3	237	683	683
4	151	231	151

#	Name	Type 1	Type 2	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation	Legendary
1	Bulbasaur	Grass	Poison	45	49	49	65	65	45	1	False
2	Ivysaur	Grass	Poison	60	62	63	80	80	60	1	False
3	Venusaur	Grass	Poison	80	82	83	100	100	80	1	False
4	Mega Venusaur	Grass	Poison	80	100	123	122	120	80	1	False
5	Charmander	Fire	NaN	39	52	43	60	50	65	1	False

	Normal	Fire	Water	Electric	Grass	Ice	Fighting	Poison	Ground	Flying	Psychic	Bug	Rock	Ghost	Dragon	Dark	Steel	Fairy
Normal	1	1	1	1	1	1	2	1	1	1	1	1	1	0	1	1	1	1
Fire	1	0.5	2	1	0.5	0.5	1	1	2	1	1	0.5	2	1	1	1	0.5	0.5
Water	1	0.5	0.5	2	2	0.5	1	1	1	1	1	1	1	1	1	1	0.5	1
Electric	1	1	1	0.5	1	1	1	1	2	0.5	1	1	1	1	1	1	0.5	1
Grass	1	2	0.5	0.5	0.5	2	1	2	0.5	2	1	2	1	1	1	1	1	1
Ice	1	2	1	1	1	0.5	2	1	1	1	1	1	2	1	1	1	2	1
Fighting	1	1	1	1	1	1	1	1	1	2	2	0.5	0.5	1	1	0.5	1	2
Poison	1	1	1	1	0.5	1	0.5	0.5	2	1	2	0.5	1	1	1	1	1	0.5
Ground	1	1	2	0	2	2	1	0.5	1	1	1	1	0.5	1	1	1	1	1
Flying	1	1	1	2	0.5	2	0.5	1	0	1	1	0.5	2	1	1	1	1	1
Psychic	1	1	1	1	1	1	0.5	1	1	1	0.5	2	1	2	1	2	1	1
Bug	1	2	1	1	0.5	1	0.5	1	0.5	2	1	1	2	1	1	1	1	1
Rock	0.5	0.5	2	1	2	1	2	0.5	2	0.5	1	1	1	1	1	1	2	1
Ghost	0	1	1	1	1	1	0	0.5	1	1	1	0.5	1	2	1	2	1	1
Dragon	1	0.5	0.5	0.5	0.5	2	1	1	1	1	1	1	1	1	2	1	1	2
Dark	1	1	1	1	1	1	2	1	1	1	0	2	1	0.5	1	0.5	1	2
Steel	0.5	2	1	1	0.5	0.5	2	0	2	0.5	0.5	0.5	0.5	1	0.5	1	0.5	0.5
Fairy	1	1	1	1	1	1	0.5	2	1	1	1	0.5	1	1	0	0.5	2	1

Feature engineering
(generate features for 2 opponents)



Summary and next lecture

Summary of the Lecture

- Tasks in Machine Learning
- Overfitting and generalization
- Bias-variance tradeoff
- Training, validation and test sets
  - Cross-validation
- Evaluation of supervised learning algorithms
- Basic concepts in optimization

Next Lecture

- Lecture: dimensionality reduction

- Lab: implementation of dimensionality reduction techniques in Python
  - The description/data will be posted on piazza

- About the lab:
  - Bring your laptops
  - Please install Python 3
  - Main libraries: numpy, scipy, matplotlib, scikit-learn
  - Strongly recommended to use anaconda
    - https://www.anaconda.com/download/

Thank You!