CentraleSupélec

ESSEC BUSINESS SCHOOL

Machine Learning

M.Sc. in Data Sciences and Business Analytics

Lecture 1

Introduction

Fragkiskos Malliaros

Tuesday, October 3, 2017

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 yesterday!)

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

Office Hours

Instructor: Fragkiskos Malliaros

Office: CentraleSupélec, Gif Sur Yvette campus, CVN Lab, Room SC.217

Office hours: I will be available right after the lecture

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

Email: fragkiskos.me@gmail.com

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)

Thank you!

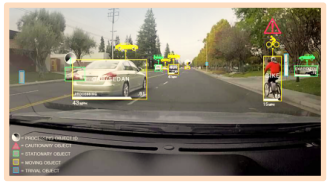
Slides of Today's Lecture

http://fragkiskos.me/introduction.pdf

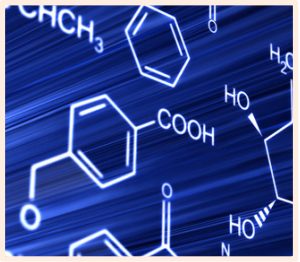
http://fragkiskos.me/python.pdf

Why
Machine Learning?

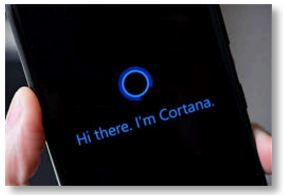
Automation and Robotics



Drug Discovery and Healthcare



Intelligent Personal Assistants

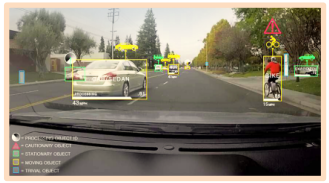


Recommender Systems

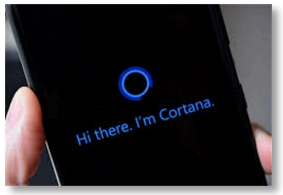
amazon

NETFLIX

Automation and Robotics



Real and important problems

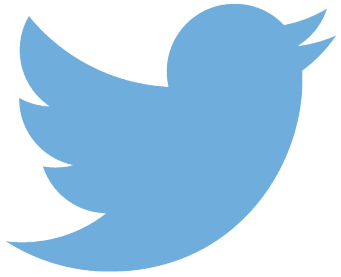


Recommender Systems

amazon

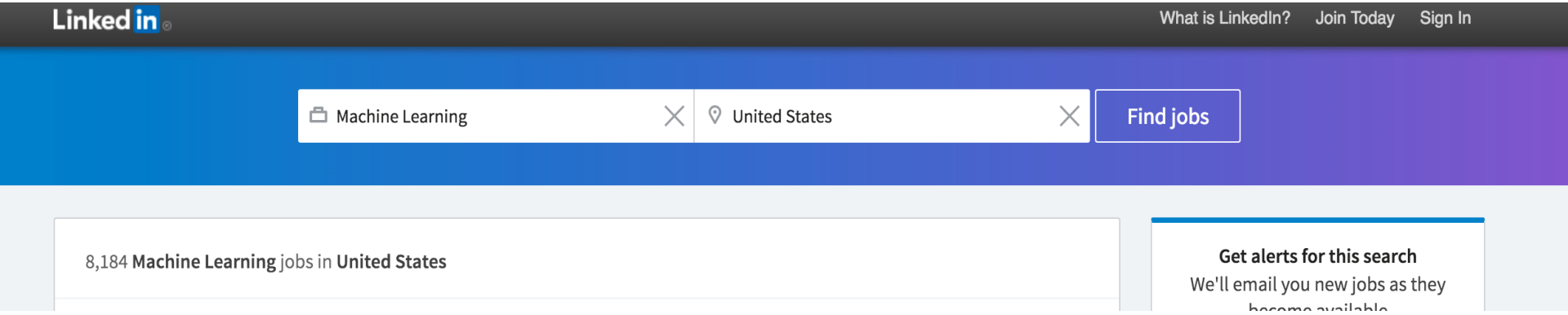
NETFLIX

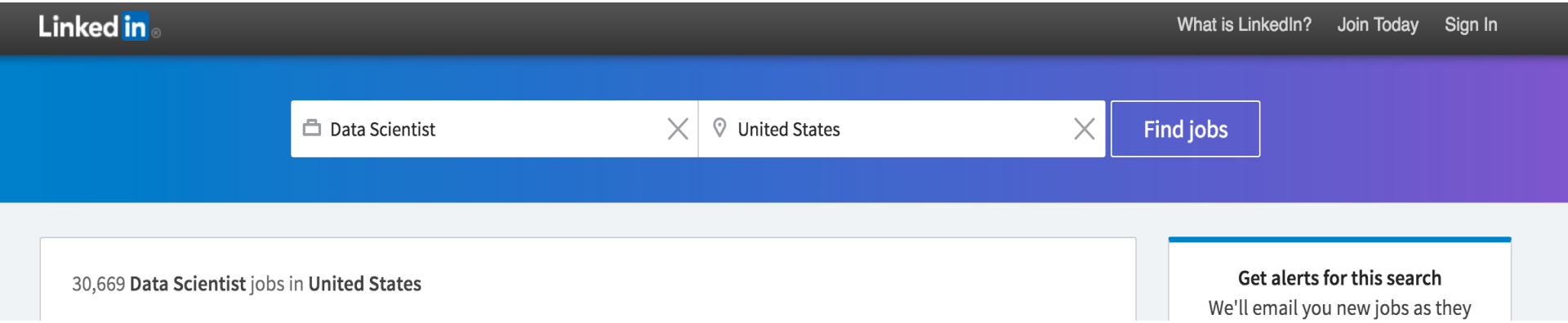

facebook

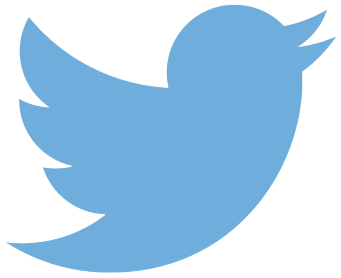



amazon


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

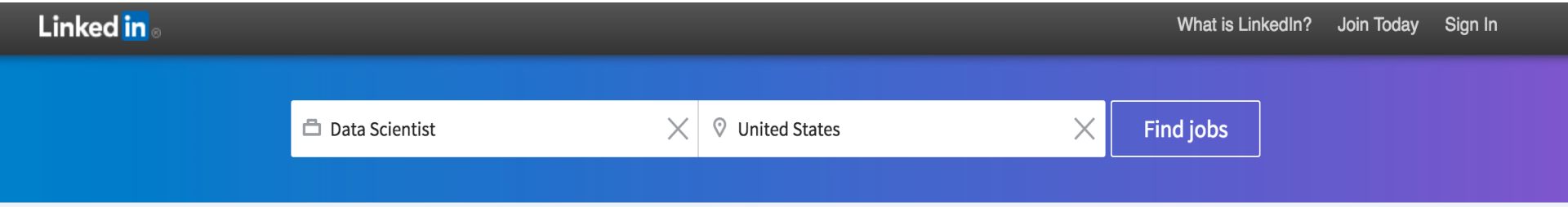

Linked in
What is LinkedIn?
Join Today
Sign In
Data Scientist
United States
Find jobs
30,669 Data Scientist jobs in United States
Get alerts for this search
We'll email you new jobs as they


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



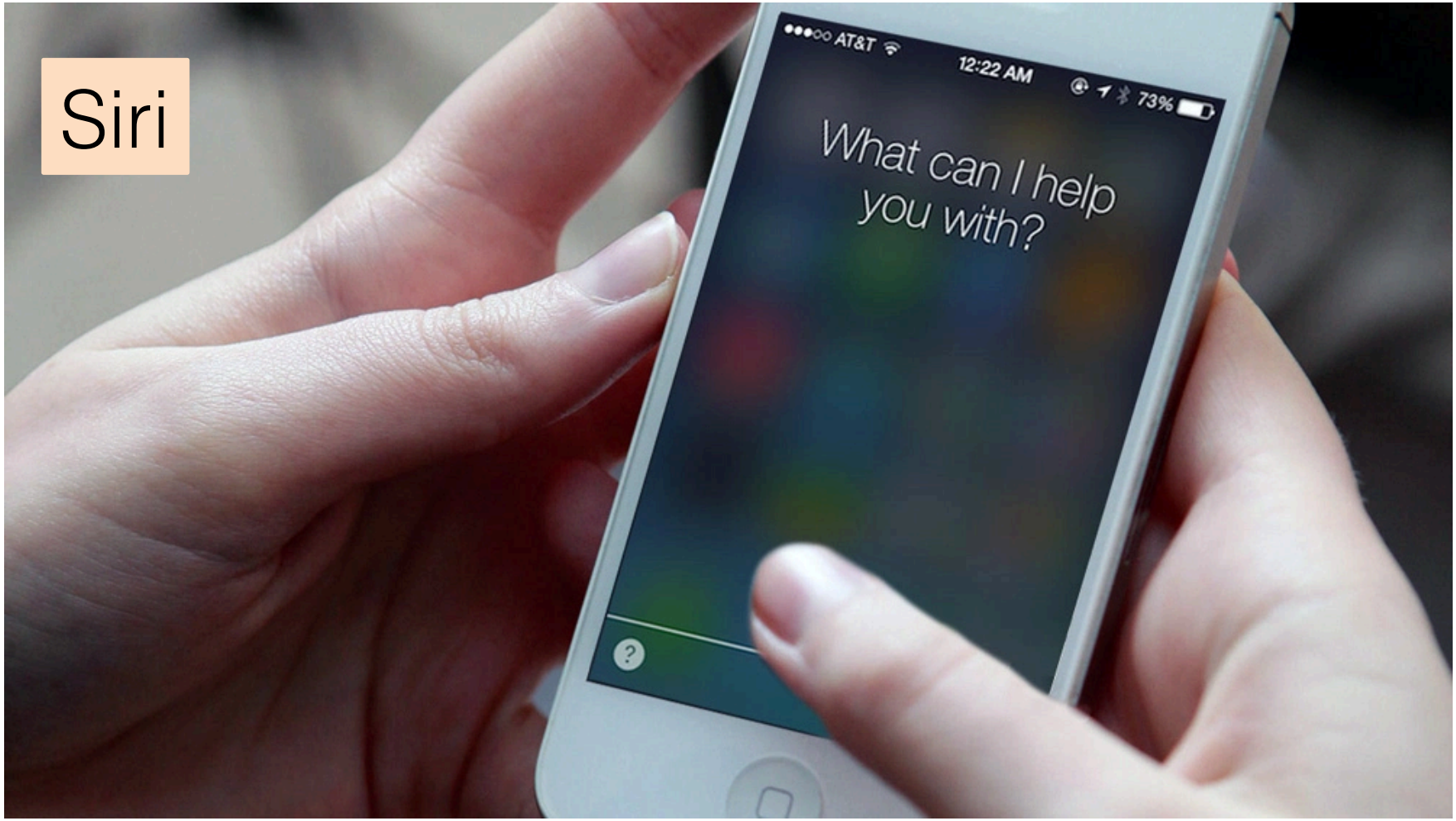
30,669 **Data Scientist** jobs in **United States**

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)

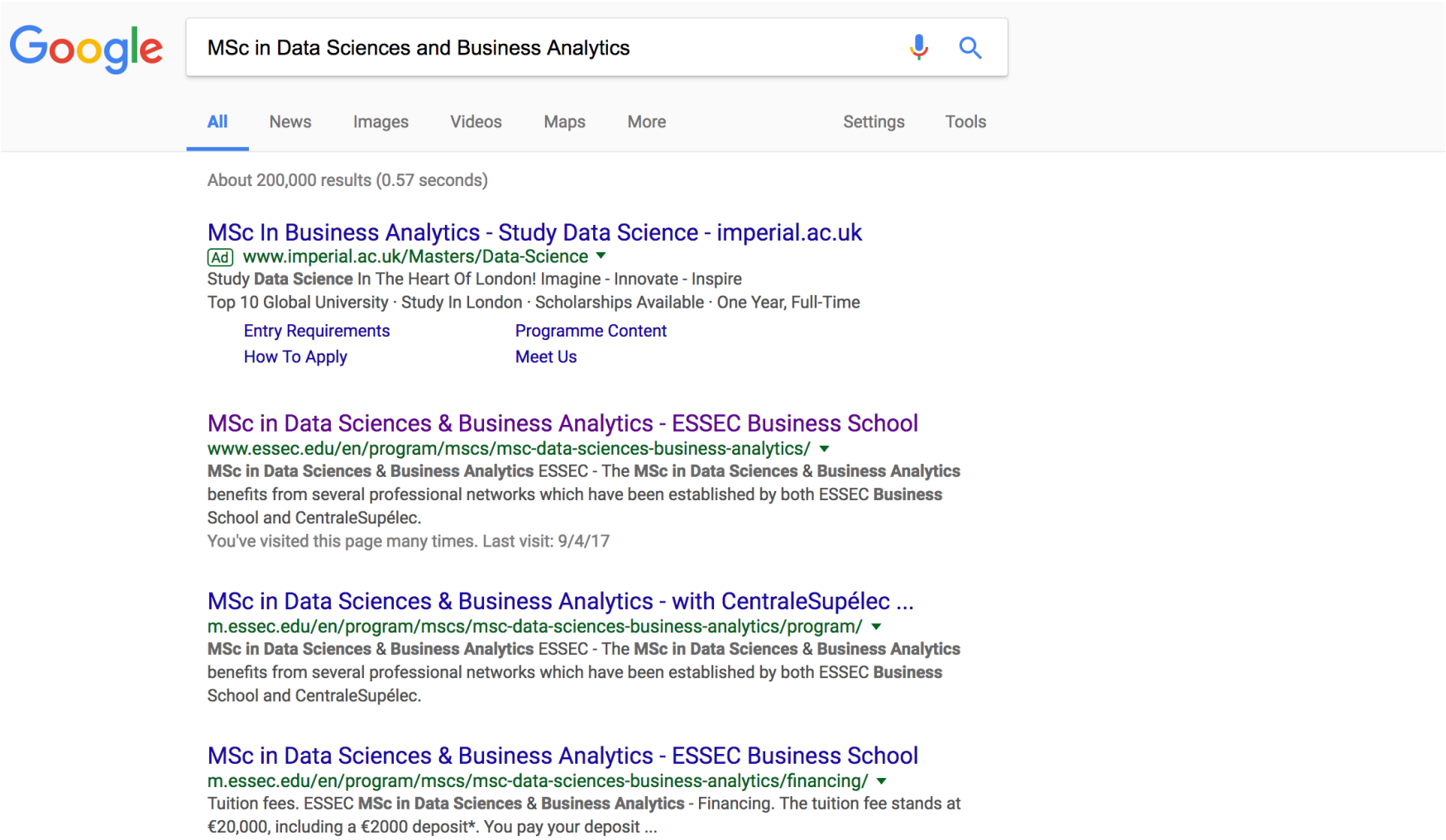
Example of a Recommendation System

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

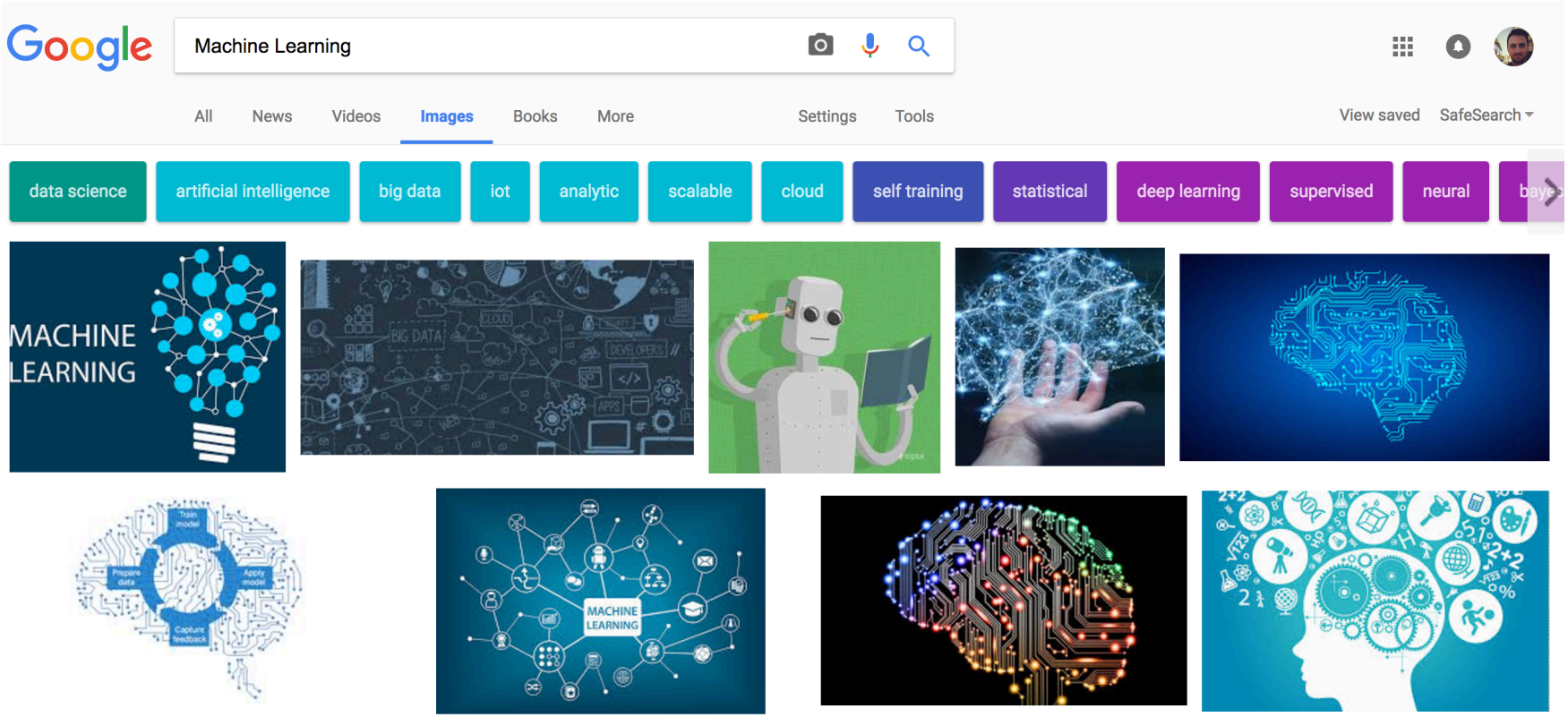
Machine Learning is Almost Everywhere

- 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)



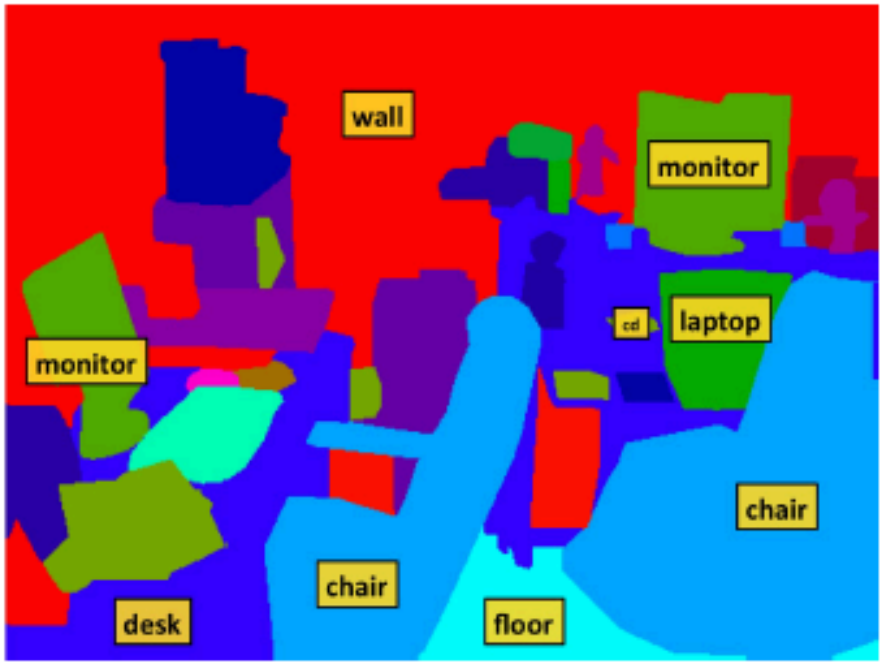
Information Retrieval (2/2)



Machine Learning is Almost Everywhere

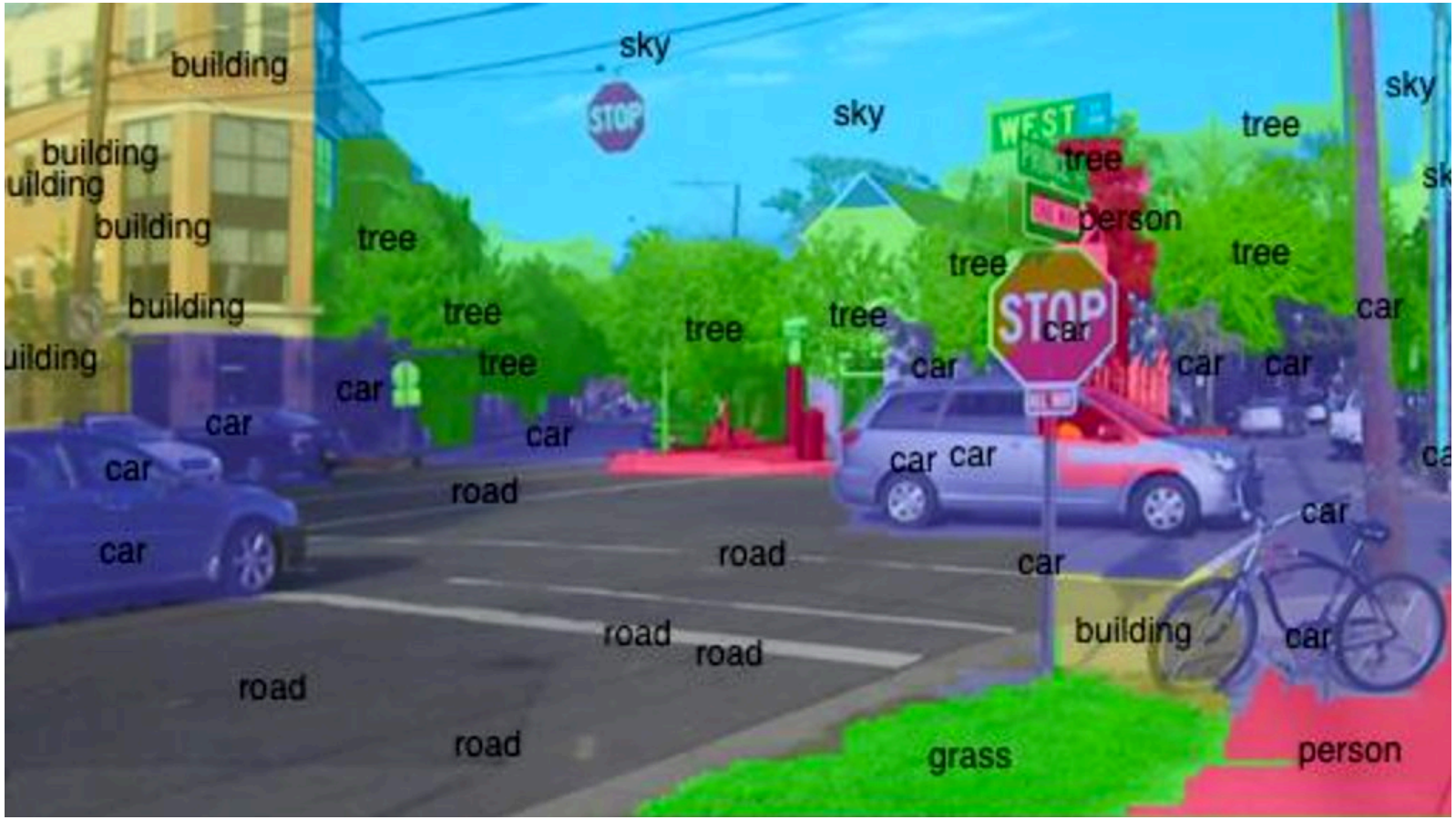
- Recognizing patterns: Speech Recognition, facial identity, etc
- Recommender Systems: Noisy data, commercial pay-off (e.g., Amazon, Netflix)
- Information retrieval: Find documents or images with similar content
- Computer vision: detection, segmentation, depth estimation, optical flow, etc.

Computer Vision (1/4)



Source: R. Urtusan, U. of Toronto

Computer Vision (2/4)



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

Computer Vision (3/4)

Source: https://www.amazon.com/Kinect-Sports-Xbox-360/dp/B002I0JBVY

Computer Vision (4/4)

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

Source: R. Urtusan, U. of Toronto

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

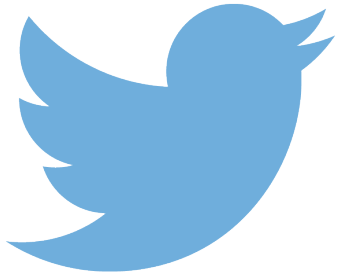
Source: https://www.newscientist.com

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

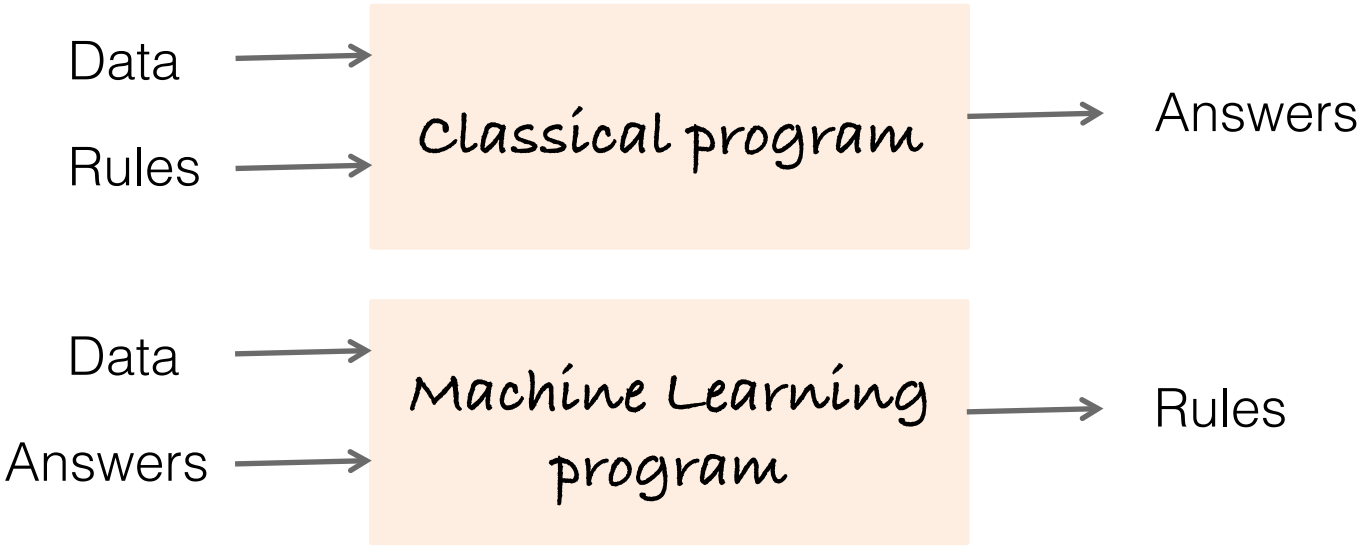
We have to handle large data sets

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 payroll
- 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?

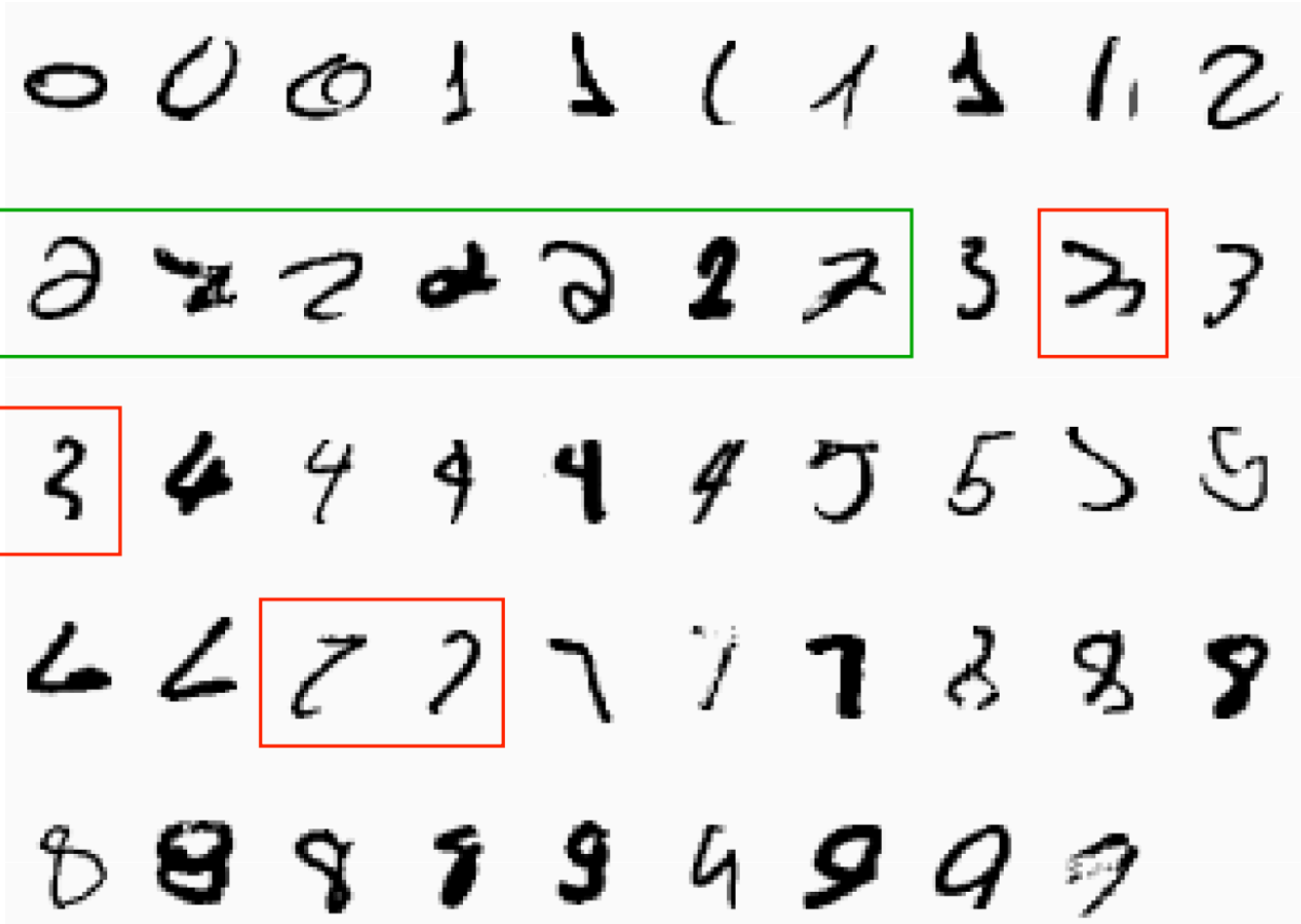
- Learning systems are not directly programmed to solve a problem, instead develop their own program based on:
  - Examples of how they should behave
  - From trial-and-error experience trying to solve the problem

- 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’?



Why Use Learning?

- It is very hard to write programs that solve problems like recognizing a handwritten digit
  - What distinguishes a ‘2’ from a ‘7’?
  - How does our brain do it?
- Instead of writing a program by hand, we collect examples that specify the correct output for a given input
- A machine learning algorithm then takes these examples and produces a program that does the job
  - The program produced by the learning algorithm may look very different from a typical hand-written program
  - If we do it right, the program works for new cases as well as for the ones we trained it on

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, dimension reduction, supervised – classification, regression?)
  - Describe it formally in terms of design matrix (observations x features), features, samples
- Define a loss function
- Define generalization

More on that will follow soon

Example of ML Pipeline

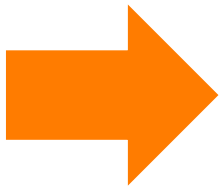
- Running example: image classification
- Goal: “train a computer to recognize a cat from a dog”
- How?

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



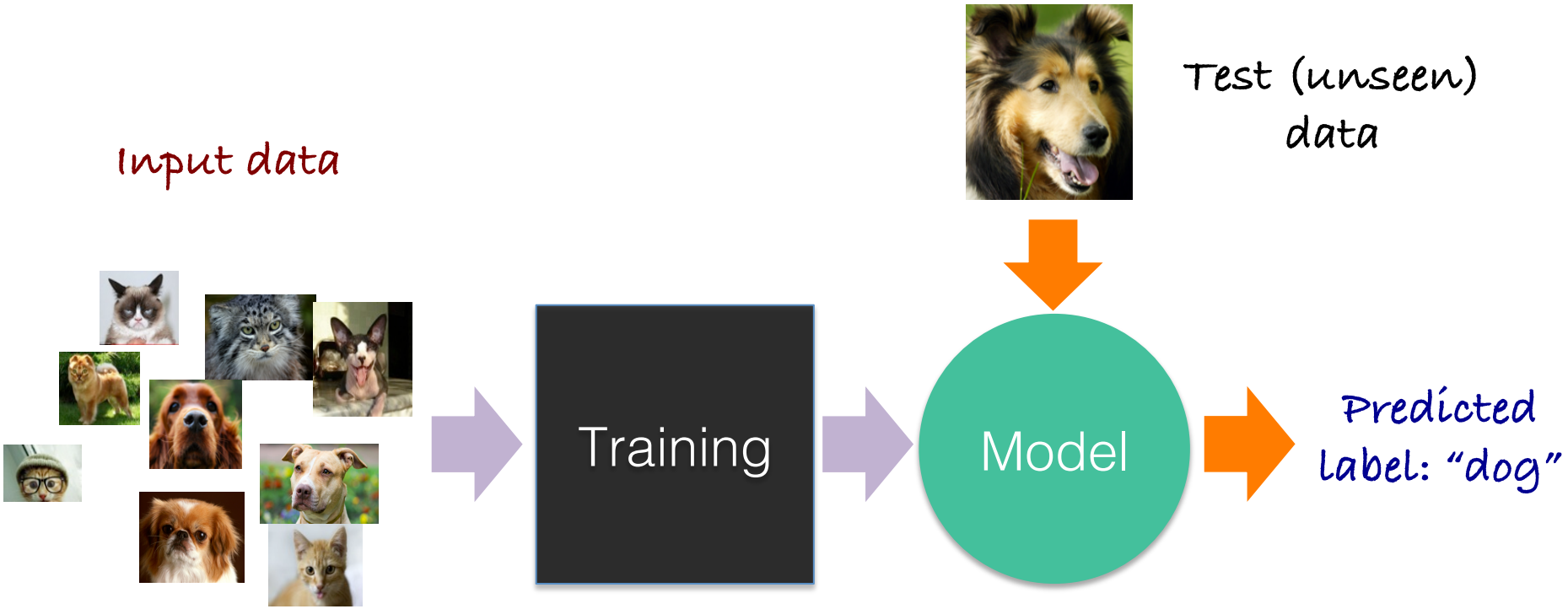
“predicts” the right label on unseen data

From This

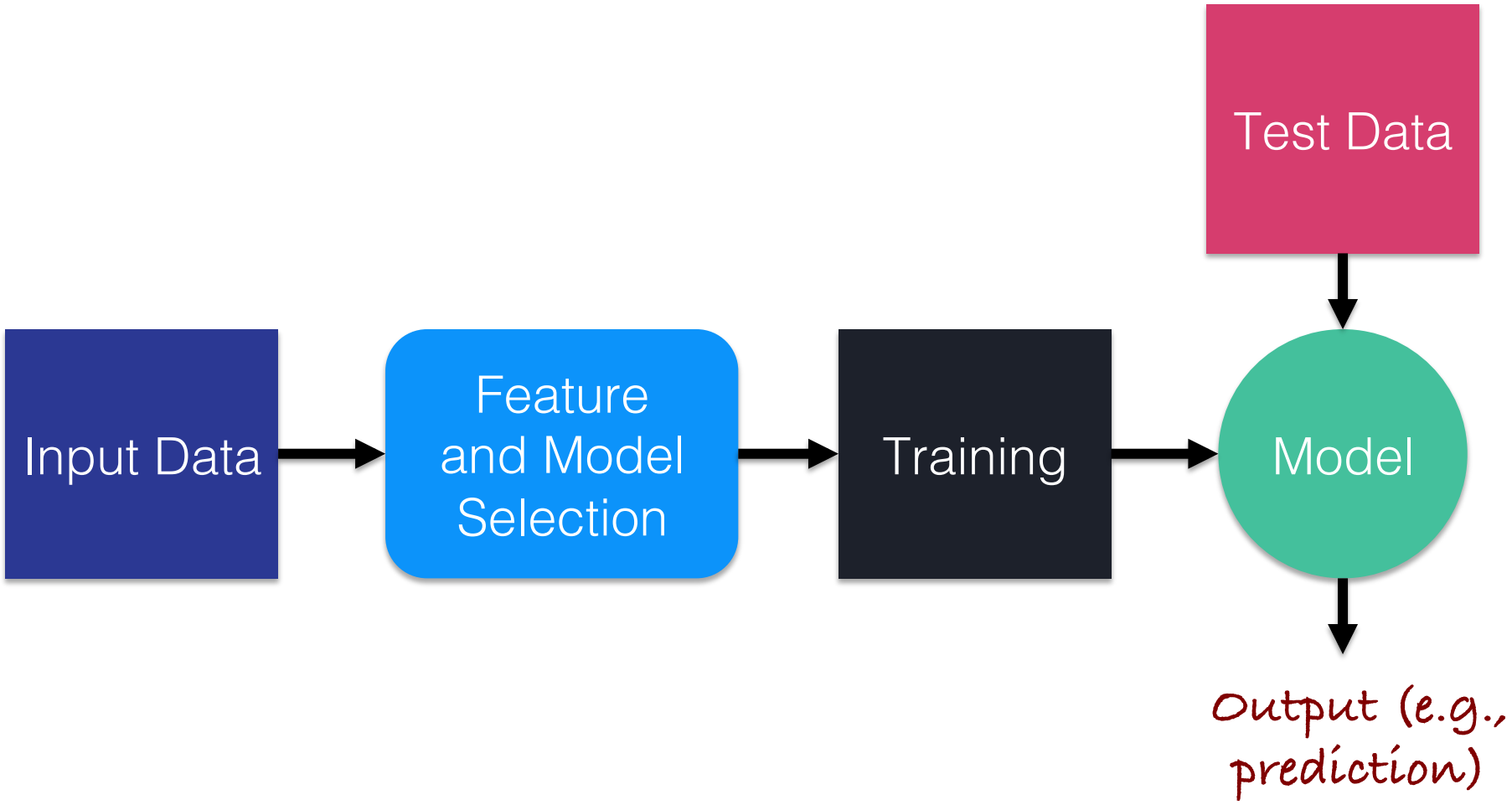
To this

Example of ML Pipeline

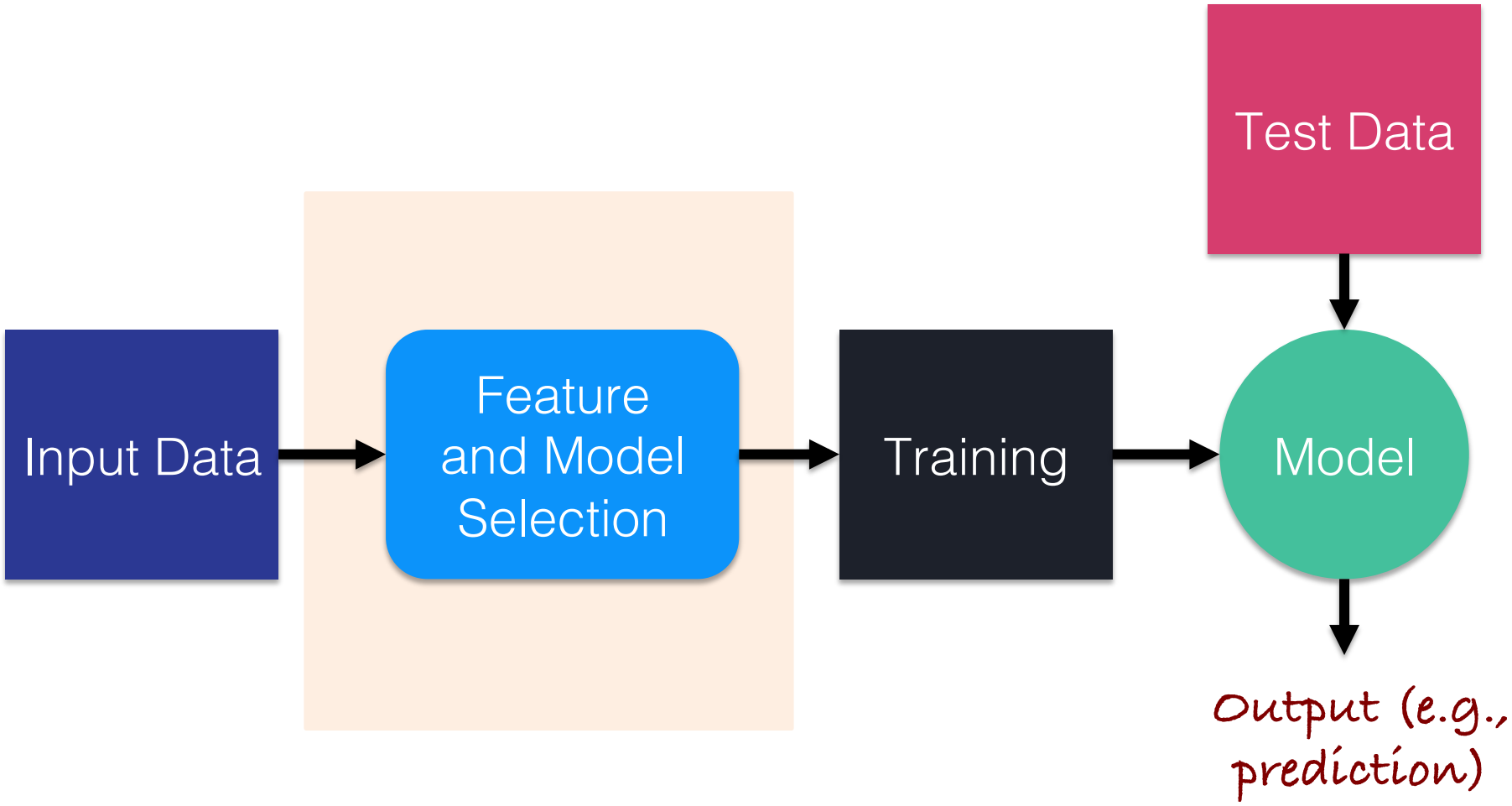
- Running example: image classification
- Goal: “train a computer to recognize a cat from a dog”
- How?



ML Pipeline


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

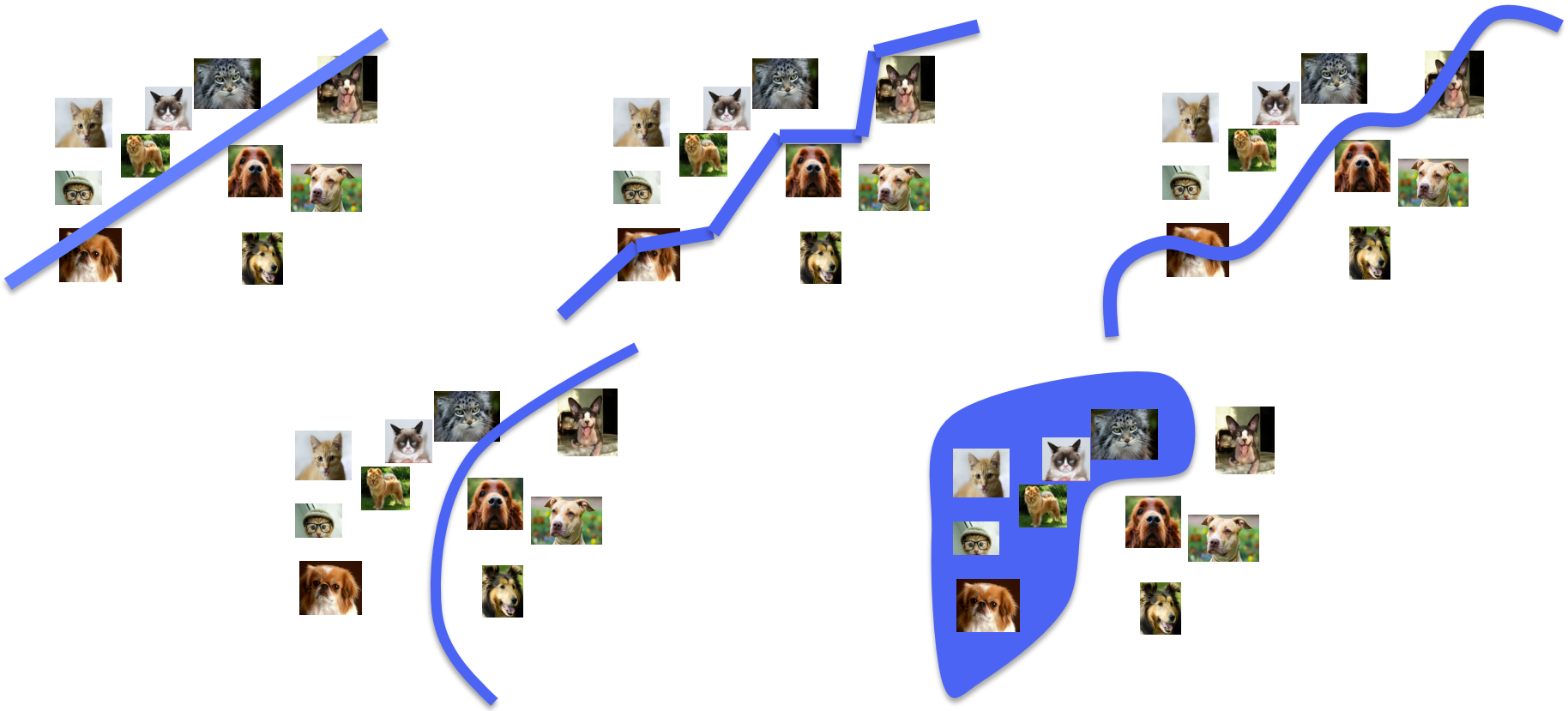
ML Pipeline


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

Feature Selection

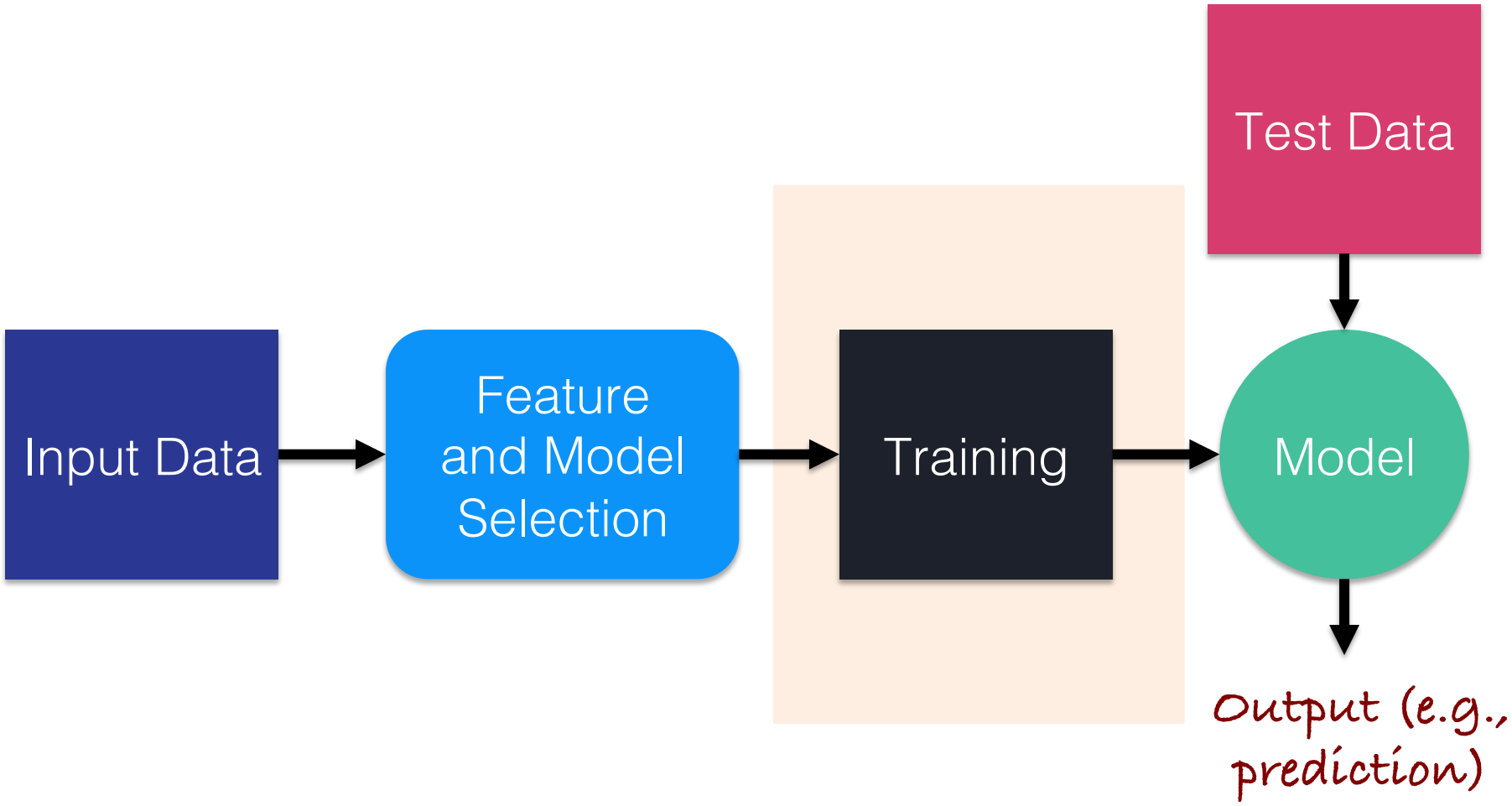
Goal: Use “informative” features

Choose your Predictor (Hypothesis Class)

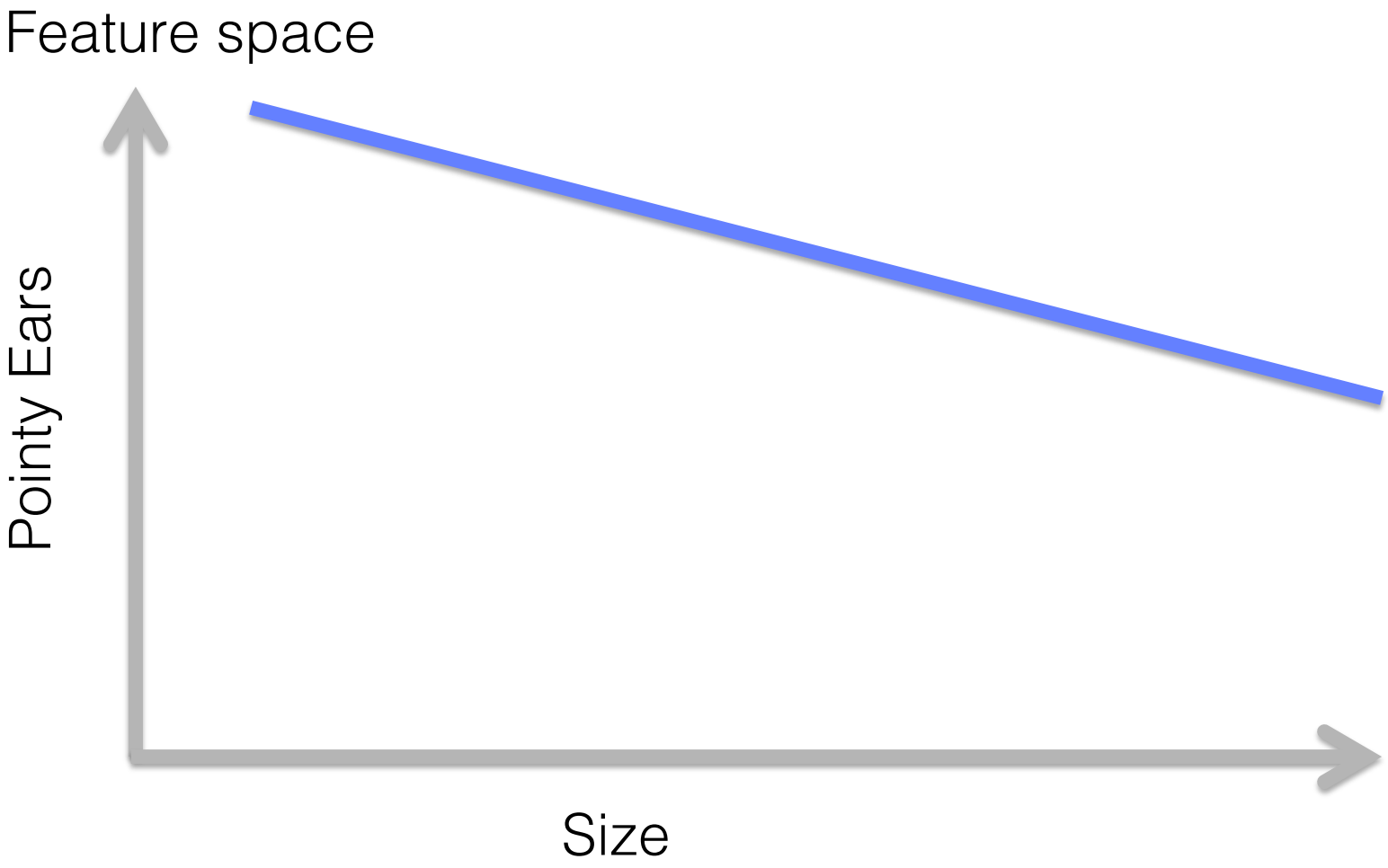


Goal: Pick a predictor that is
1) expressive 2) easy to train 3) doesn't overfit

ML Pipeline

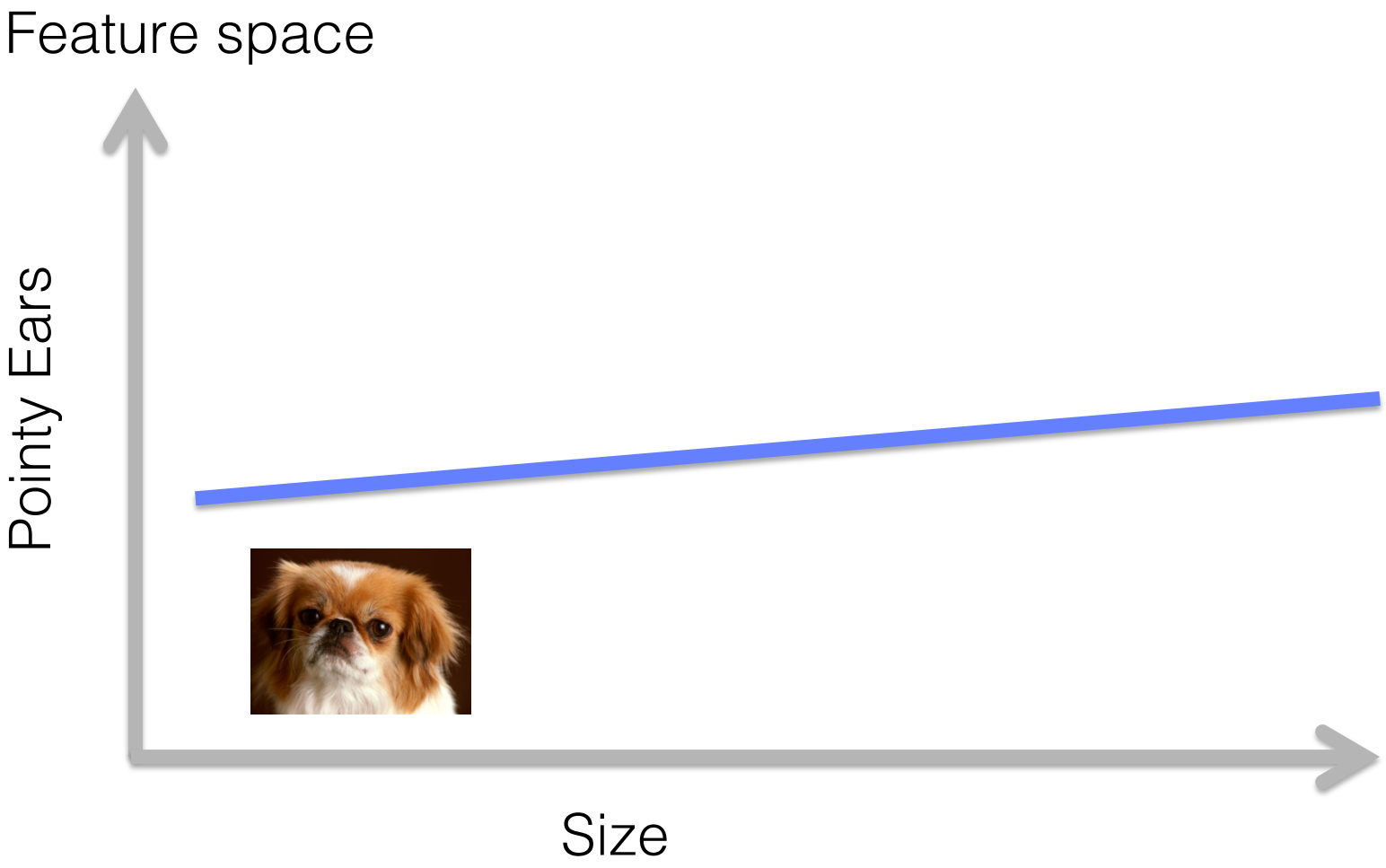

Test Data
Input Data
Feature and Model Selection
Training
Model
Output (e.g., prediction)

Then, Train the Model



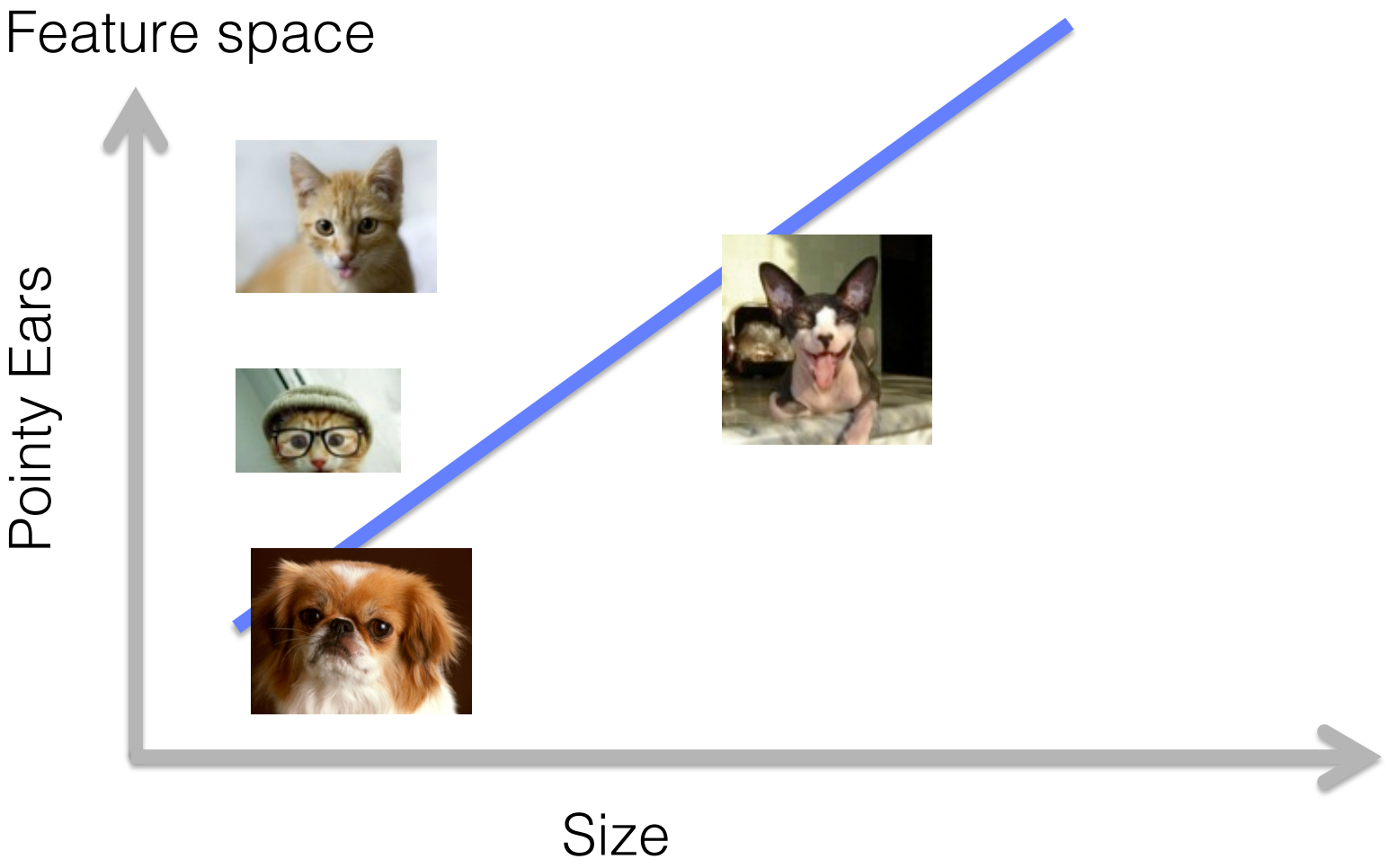
Goal: Train a model to minimizes training error

Then, Train the Model



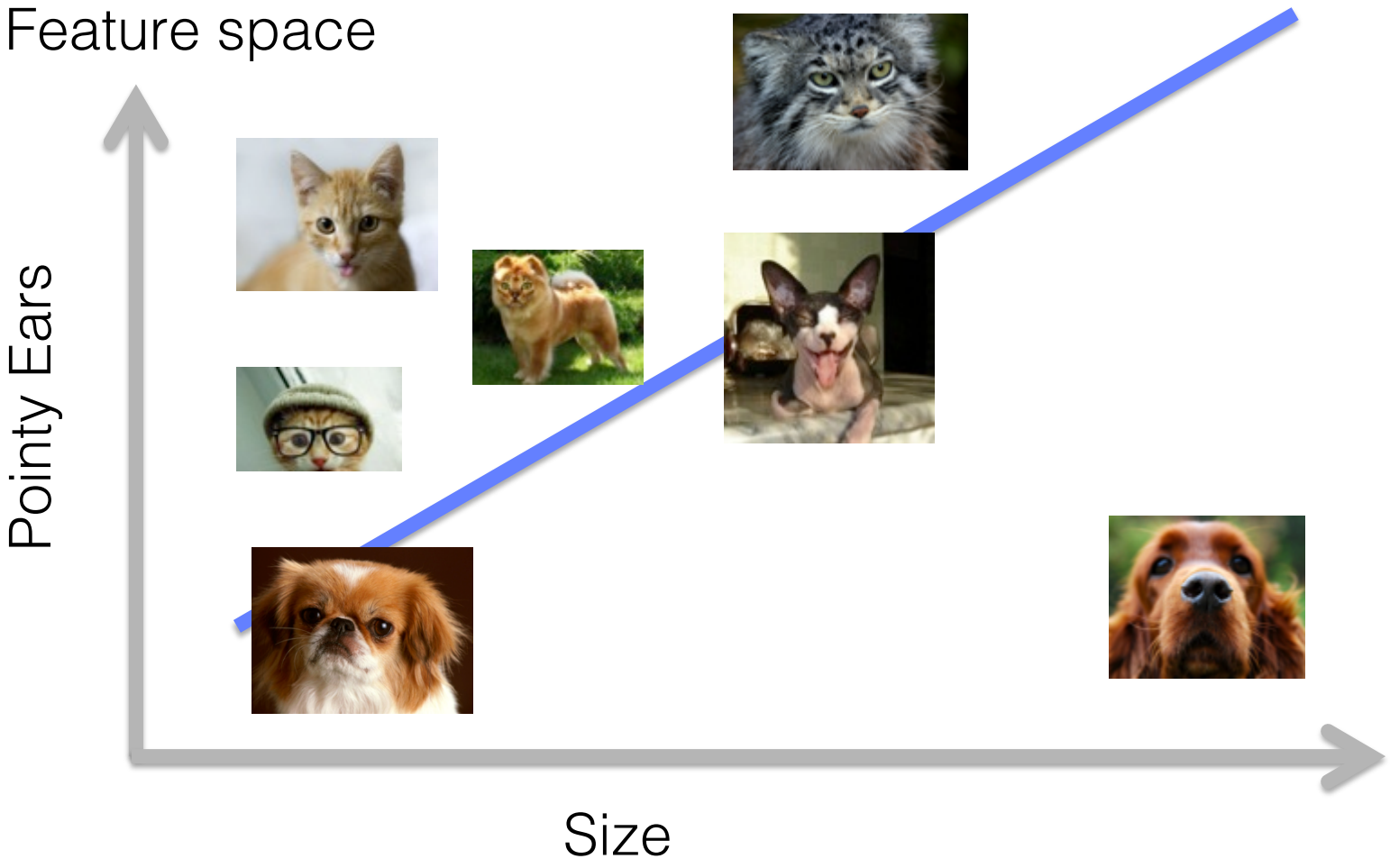
Goal: Train a model to minimizes training error

Then, Train the Model



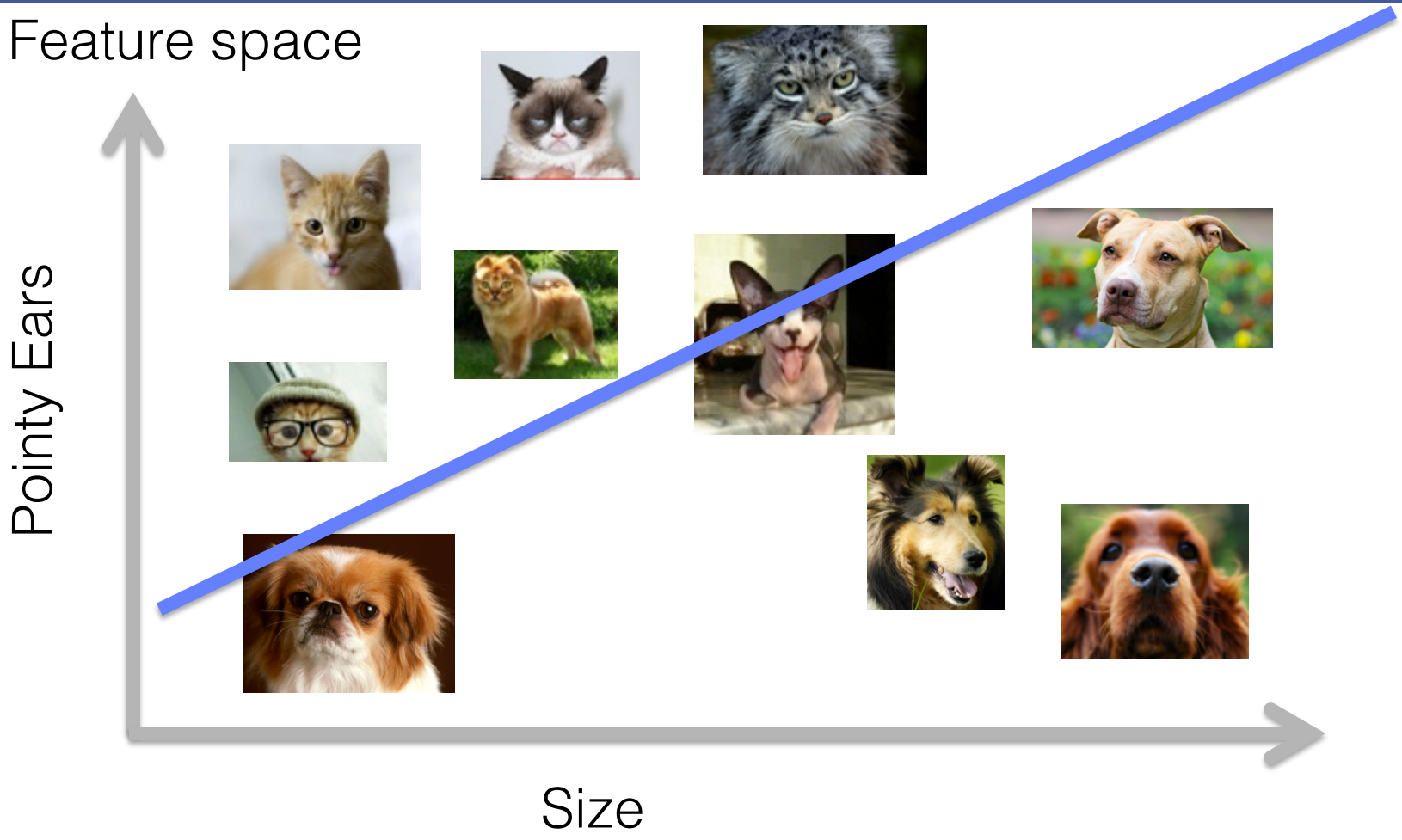
Goal: Train a model to minimizes training error

Then, Train the Model



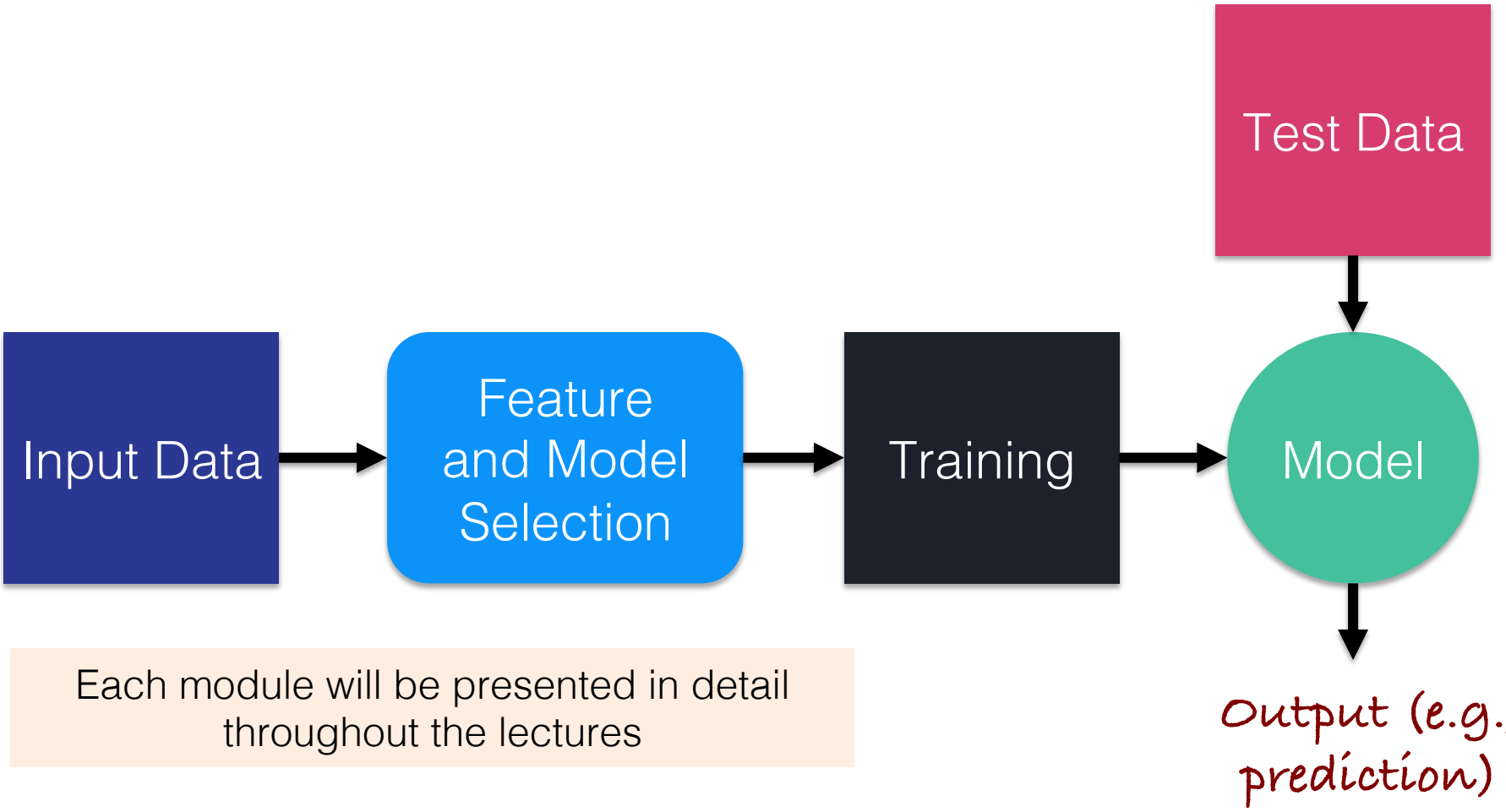
Goal: Train a model to minimizes training error

Then, Train the Model



Goal: Train a model to minimizes training error

ML Pipeline



Each module will be presented in detail throughout the lectures

Questions?

Machine Learning vs. Data Mining

- Data-mining: typically using very simple machine learning techniques on very large databases
  - Often, exploratory analysis of data
  - Data mining also covers the task of association rule mining (e.g., market basket analysis)

- Now lines are blurred: many ML problems involve tons of data

- Problems with AI flavor (e.g., recognition, robot navigation) traditionally belong to the domain of ML

Machine Learning vs. Statistics

- ML uses statistical theory to build models
- A lot of ML is rediscovery of things statisticians already knew; often disguised by differences in terminology
- But the emphasis is very different
  - Good piece of statistics: Clever proof that a relatively simple estimation procedure is asymptotically unbiased
  - Good piece of ML: Demo that a complicated algorithm produces impressive results on a specific task
  - Nevertheless, nowadays many ML algorithms come with theoretical guarantees
- We can view ML as applying computational techniques to statistical problems
  - But go beyond typical statistics problems, with different aims (speed vs. accuracy)

Cultural Gap

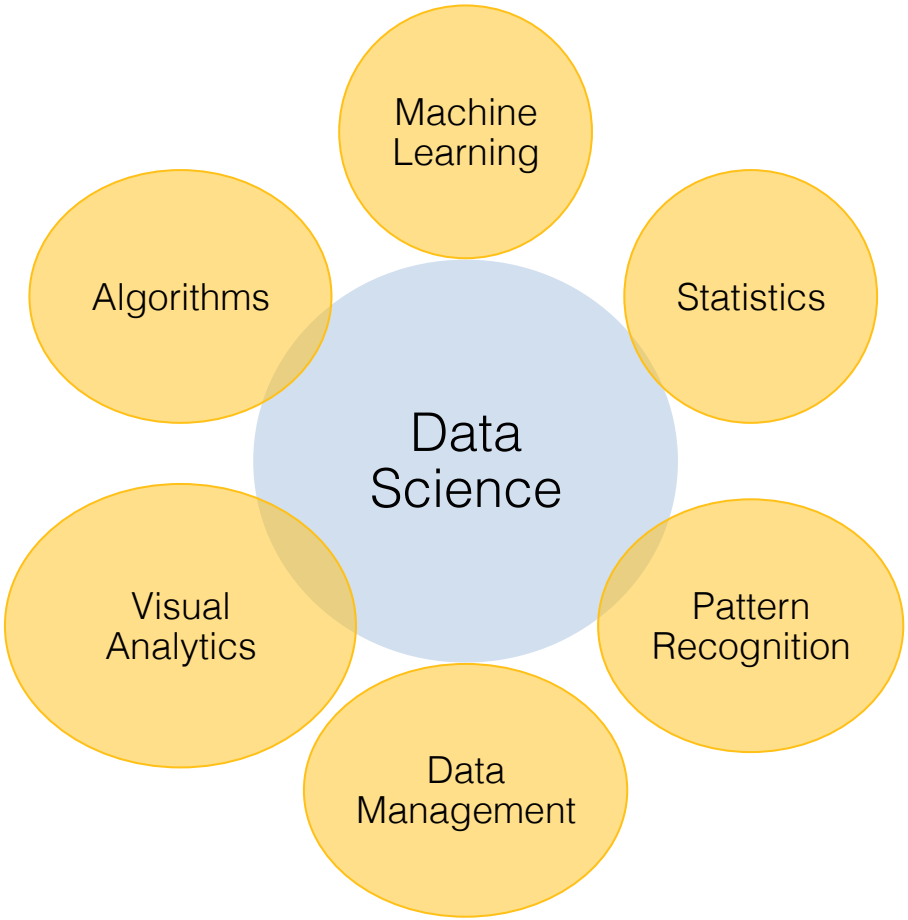
- Machine Learning
  - Weights
  - Learning
  - Generalization
  - Supervised learning
  - Unsupervised learning
  - Large grant: $1,000,000
  - Conference location: Snowbird, French Alps
- Statistics
  - Parameters
  - Fitting
  - Test set performance
  - Regression/classification
  - Density estimation, clustering
  - Large grant: $50,000
  - Conference location: Las Vegas in August

- http://www.kdnuggets.com/2016/11/machine-learning-vs-statistics.html
- https://normaldeviate.wordpress.com/2012/06/12/statistics-versus-machine-learning-5-2/

ML and Artificial Intelligence

- ML is subfield of Artificial Intelligence
  - A system that lives in a changing environment must have the ability to learn in order to adapt
  - ML algorithms are building blocks that make computers behave in a more intelligent way by generalizing, rather than merely storing and retrieving data (like a database system would do)

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
- 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)

[We may slightly deviate in some lectures]

2. Lab sessions
   - 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
   - Hands-on experience on ML algorithms
     - Some of the algorithms will be implemented from scratch
3. Assignments and project

Coursework and Grading

	Weight	Details
Assignment 1 (individually)	20%	• Theoretical questions • Some of them may also require some programming in order to perform some tasks • *Week 3 (out) – Week 5 (due)*
Assignment 2 (teams of 3-4 students)	35%	• Deal with a real machine learning task • Kaggle competition • Deliver short report and code • *Week 4 (out) – ~Dec 10 (due)*
Project (teams of 3-4 students)	45%	• Project proposal (5%) • Final report + poster presentation (40%) • *Proposal due: ~Oct 31; Final report due: Dec 19*

- Small adjustments may be done in the weights of the coursework
- A detailed description of the project will be provided soon

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 [TBA]

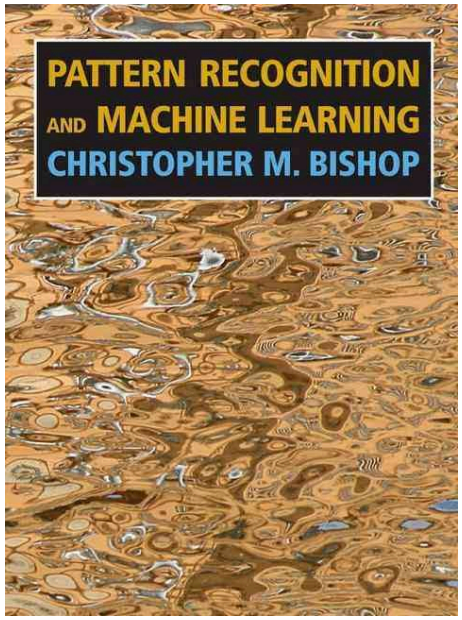
Course Logistics

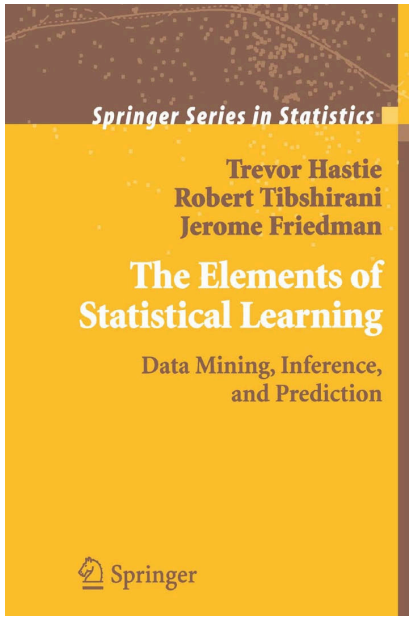
- Website
  - http://fragkiskos.me/teaching/ML-F17/
  - Information about the course, schedule, reading material
  - Resources (helpful for the assignment and project)
  - [Will go live today]

- Piazza for Q&A and material
  - http://piazza.com/centralesupelec/fall2017/mldsba/
  - Please, participate and help each other!
  - All announcements will be posted there
  - Also, lecture slides and assignments
  - Use key to enroll: mldsba17

We might switch to ESSEC's platform; until then, all material and communication will be done through piazza

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
- Some small overlap with other courses may exist
  - Take advantage of it. Some courses focus more on the theoretical aspects, while others have also an application component
- First time that I teach the course
  - Give us your feedback!

Topics that will be covered

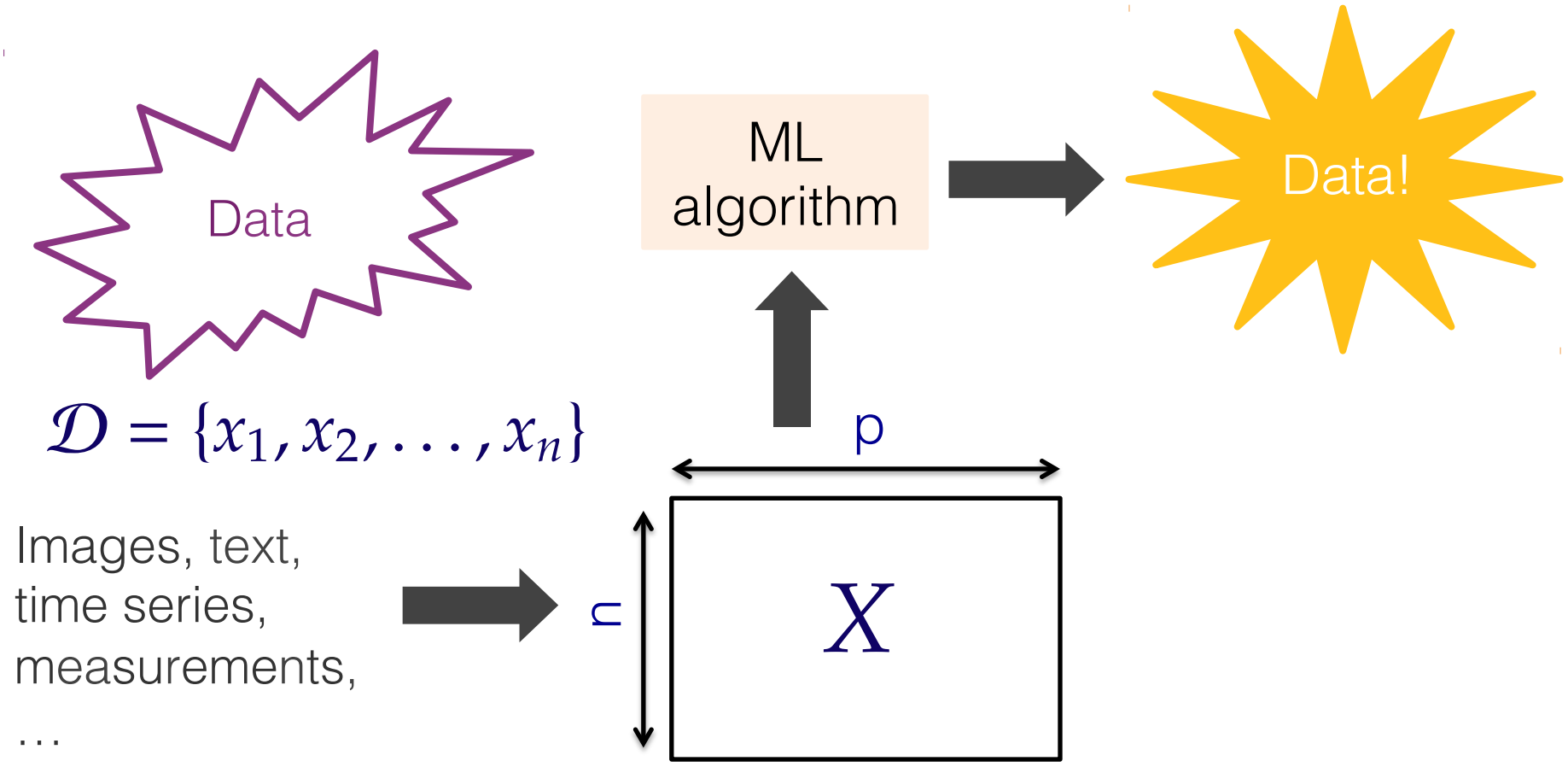
Schedule (Subject to Change)

1. Introduction. Overview of ML problems. Basic optimization concepts
2. Dimensionality reduction. Feature selection. Principal Component Analysis (PCA). Linear Discriminant Analysis (LDA)
3. Supervised learning: classification. Hypothesis testing. Model evaluation and selection. Overfitting and regularization
4. Probabilistic classifiers. Bayesian decision theory
5. Linear and logistic regression
6. Non-parametric learning. K-Nearest Neighbors
7. Kernel Methods. Support Vector Machines
8. Tree-based methods. Ensemble methods. Boosting. Random forests
9. Neural Networks
10. Unsupervised learning. Mixture Models and EM. Data Clustering. K-Means Clustering. Spectral Clustering

Over 10 weeks

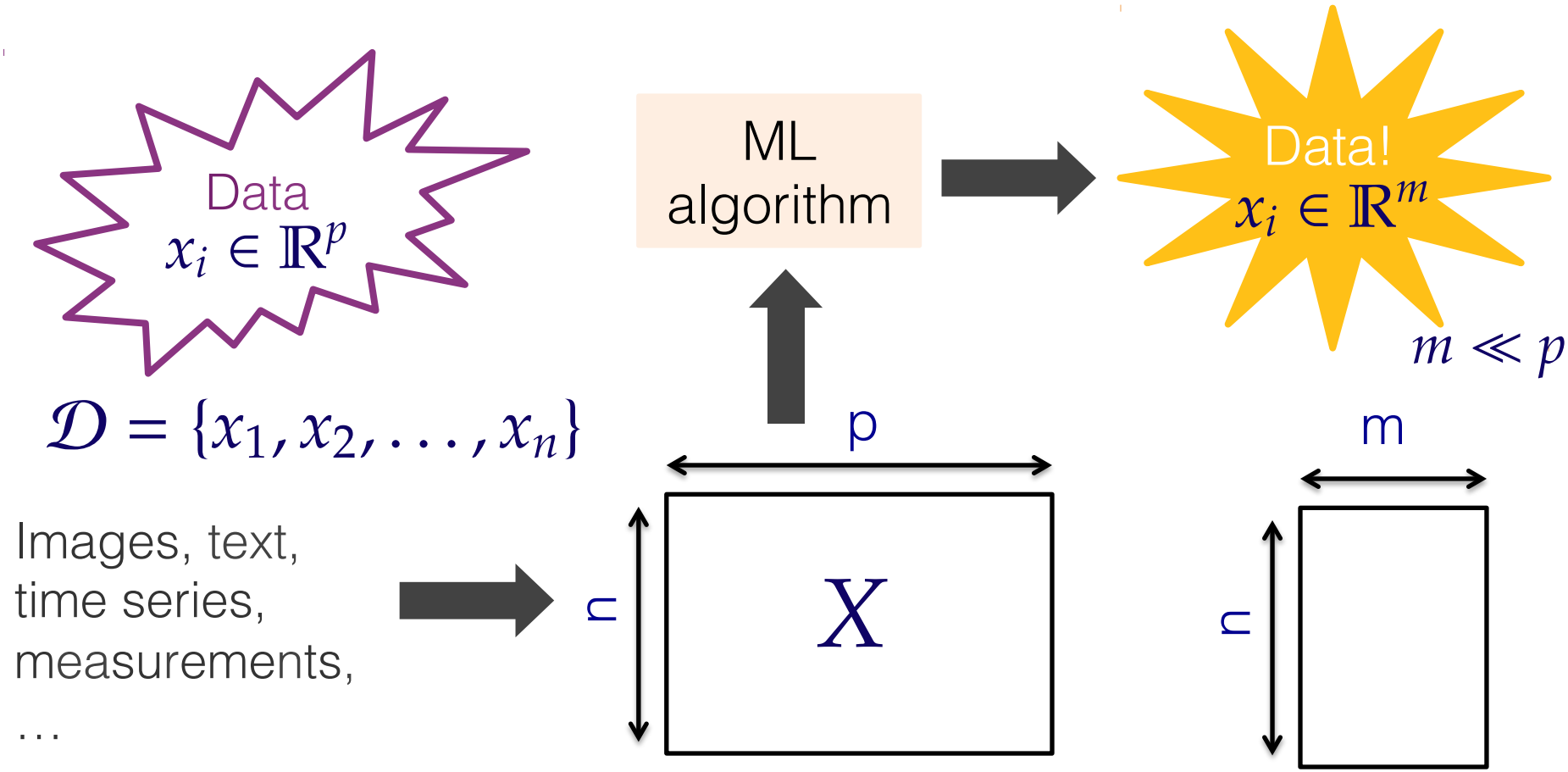
Unsupervised Learning

Learn a new representation of the data



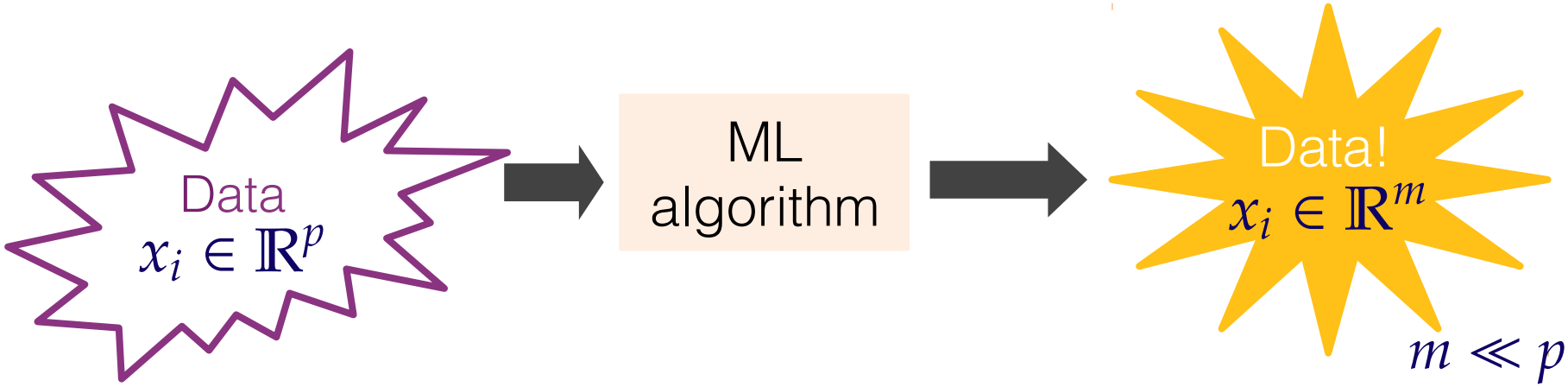
Dimensionality Reduction (1/2)

Find a lower-dimensional representation



Dimensionality Reduction (2/2)

Find a lower-dimensional representation



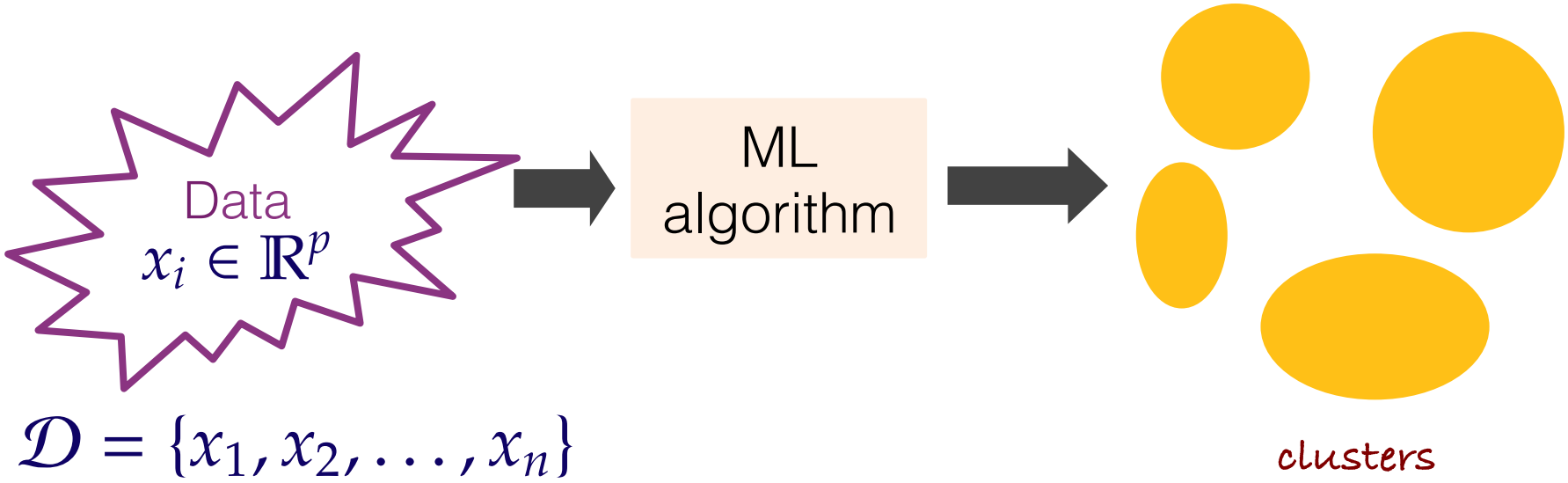
$$\mathcal{D} = \{x_1, x_2, \ldots, x_n\}$$

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

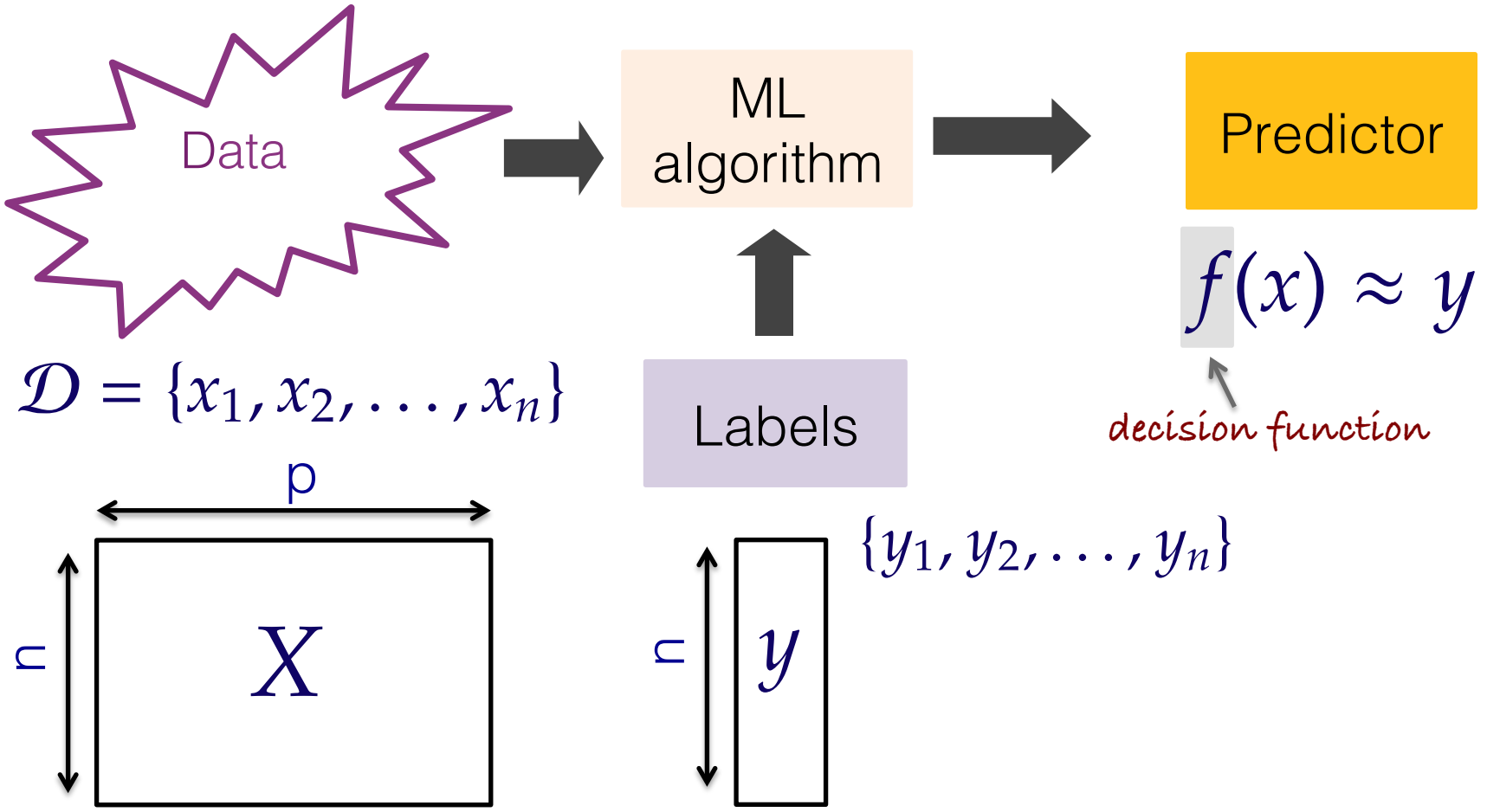
Data Clustering – Applications

- 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

$f(x) \approx y$

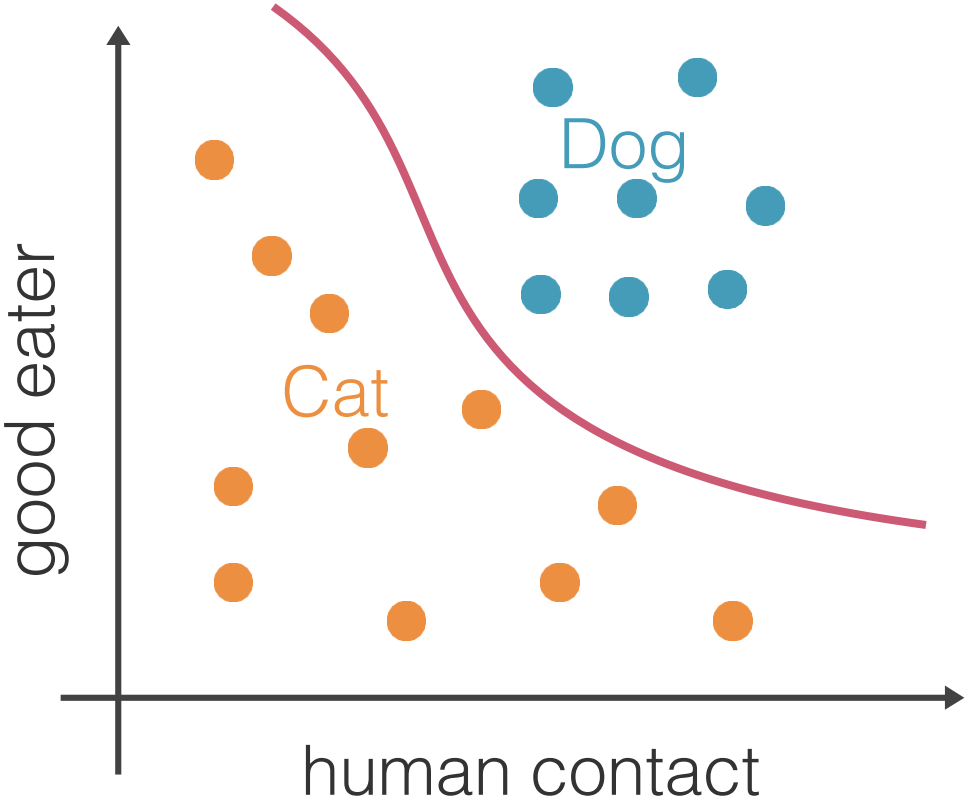
Binary classification

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

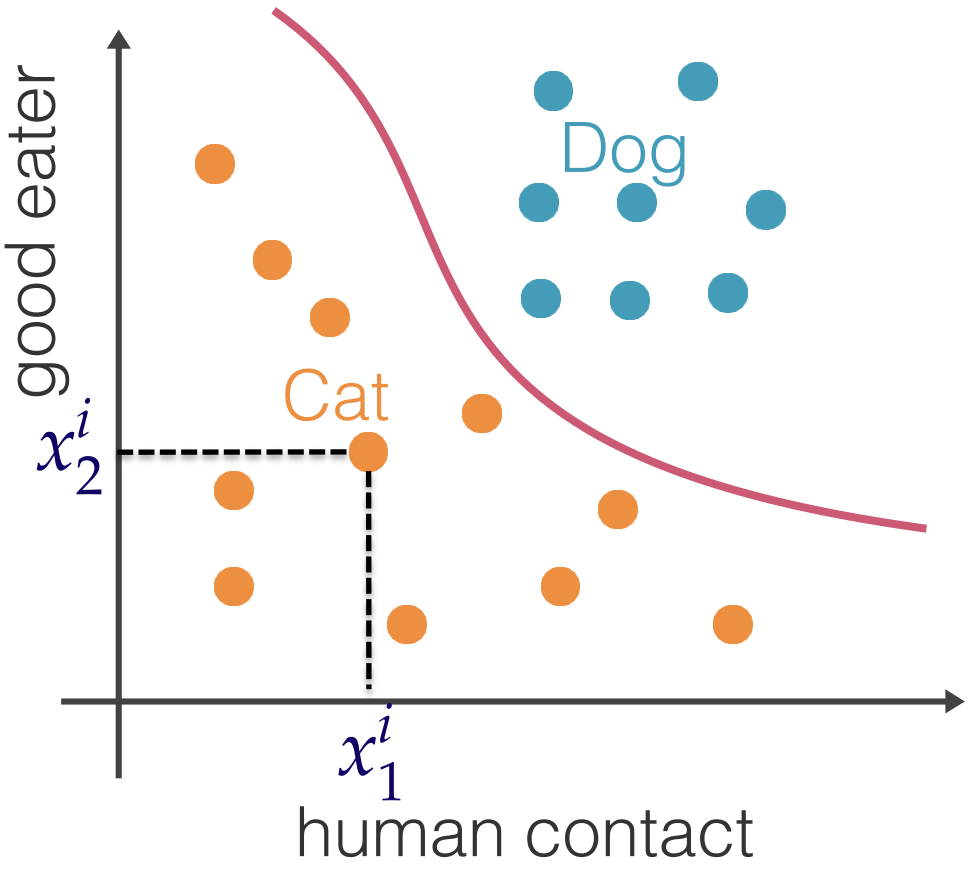
Multiclass classification

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

Classification (2/2)



Training Set $\mathcal{D}$



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

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

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

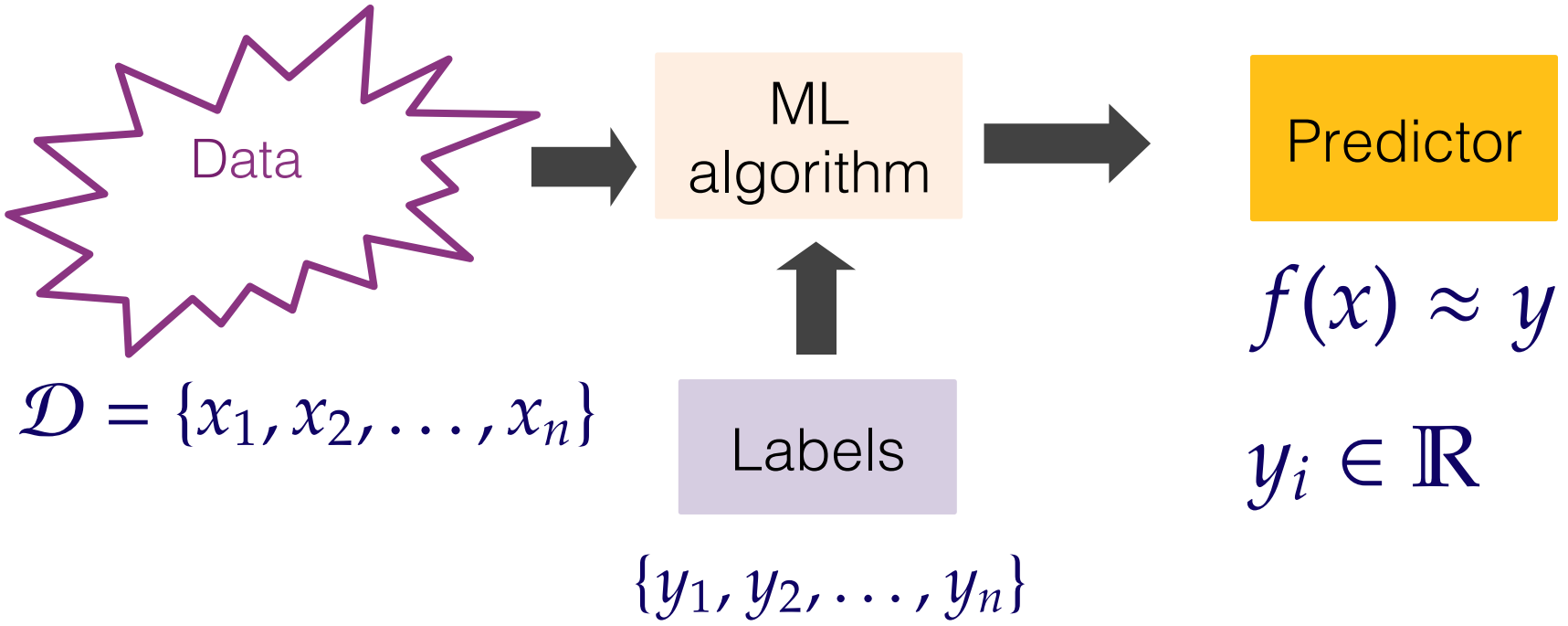
Given $\mathcal{D} = \{x^i, y^i\}_{i=1,\dots,n}$ find **f** such that $f(x) \approx y$

Classification – Applications

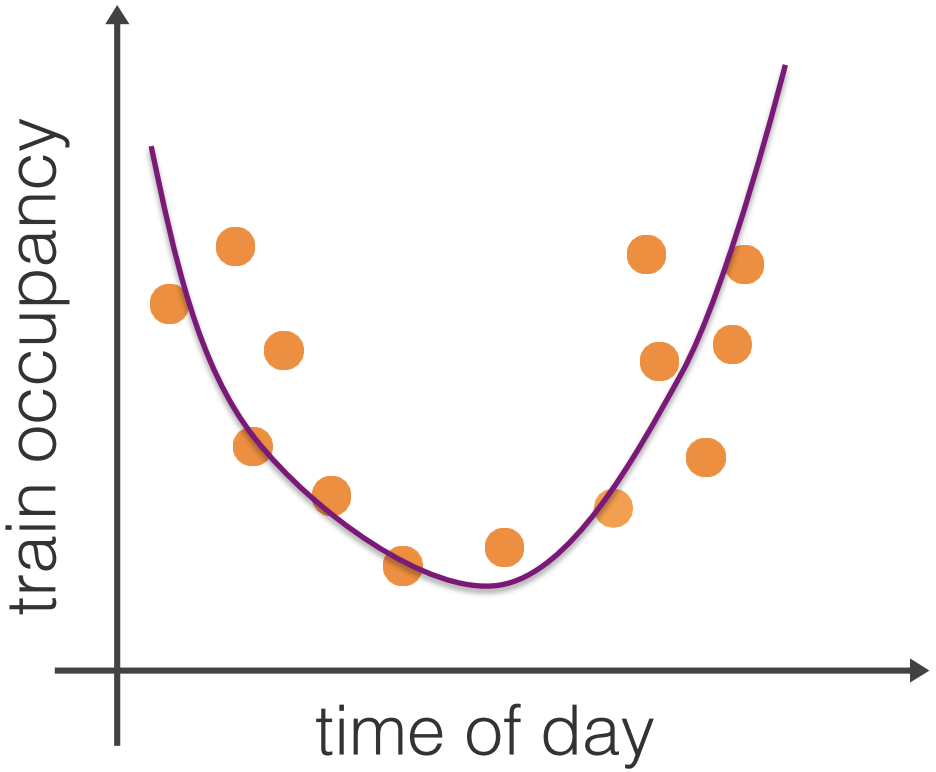
- Face recognition
  - Identify faces independently of pose, lighting, occlusion (glasses, beard), make-up, hair style
- Self-driving cars. How?
- Character recognition
  - Read letters or digits independently of different handwriting styles
- Sound recognition
  - Which language is spoken? Who wrote 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)

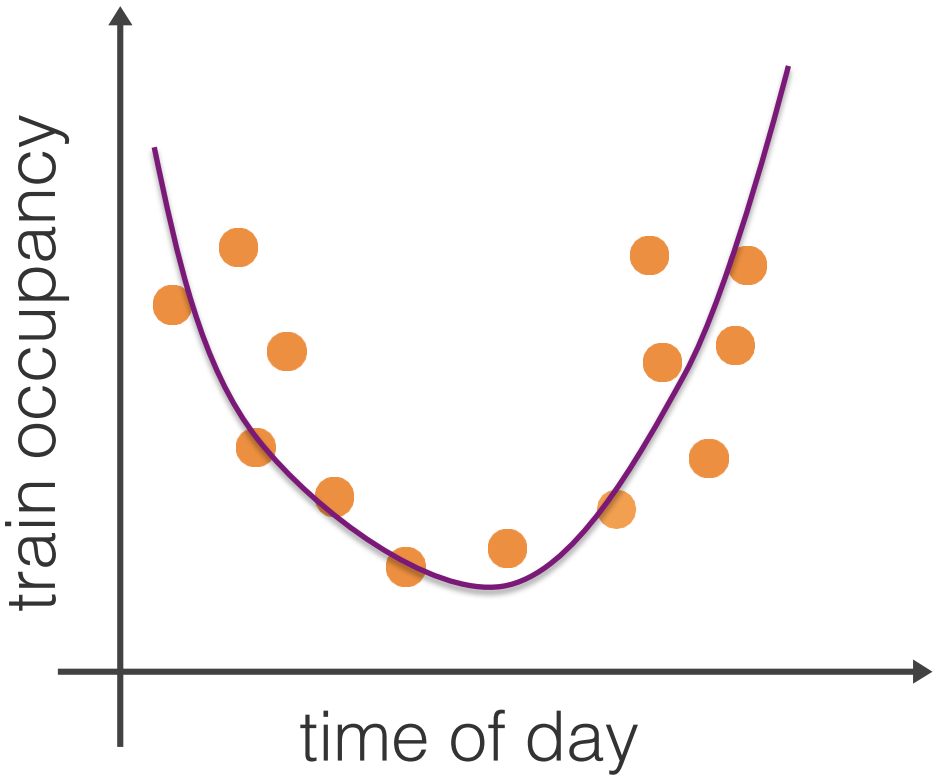
Make continues predictions



Regression (2/3)



Regression (3/3)



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

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

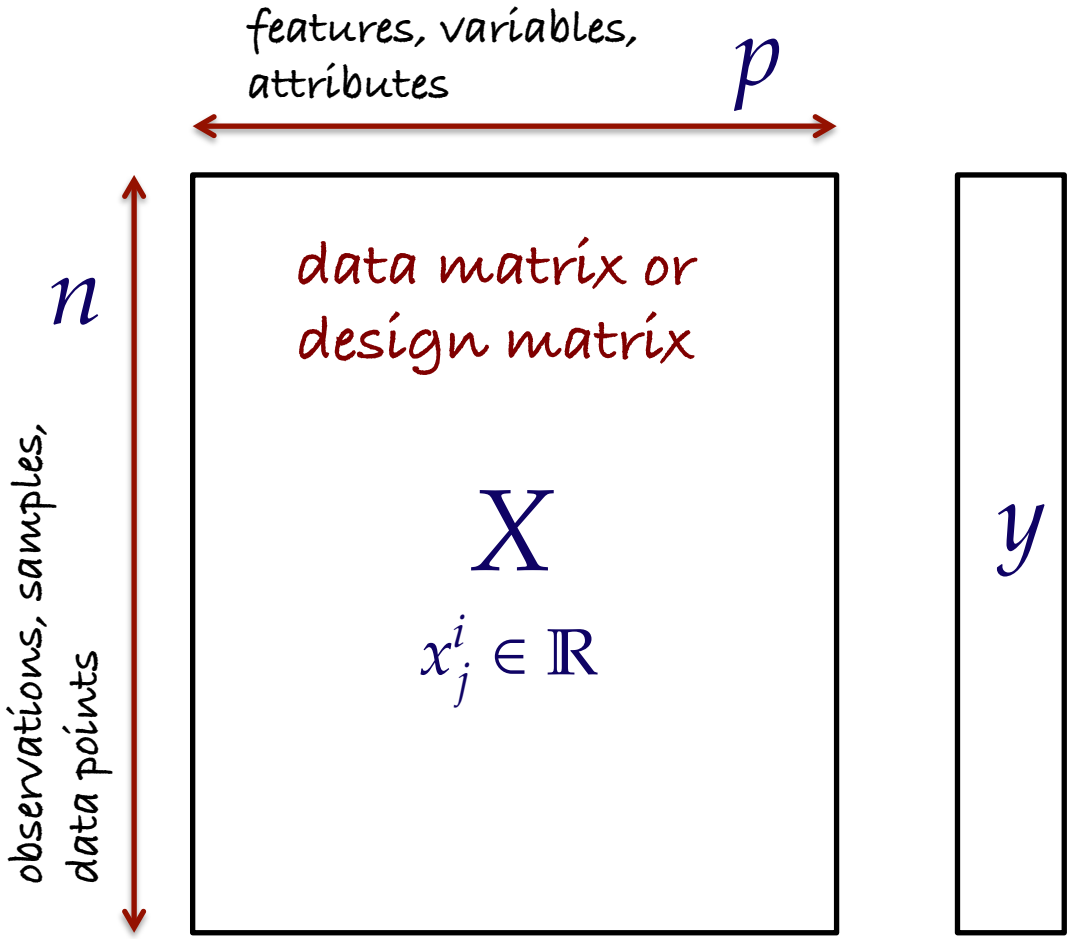
Given $\mathcal{D} = \{x^i, y^i\}_{i=1,\ldots,n}$ find **f** such that $f(x) \approx y$

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

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

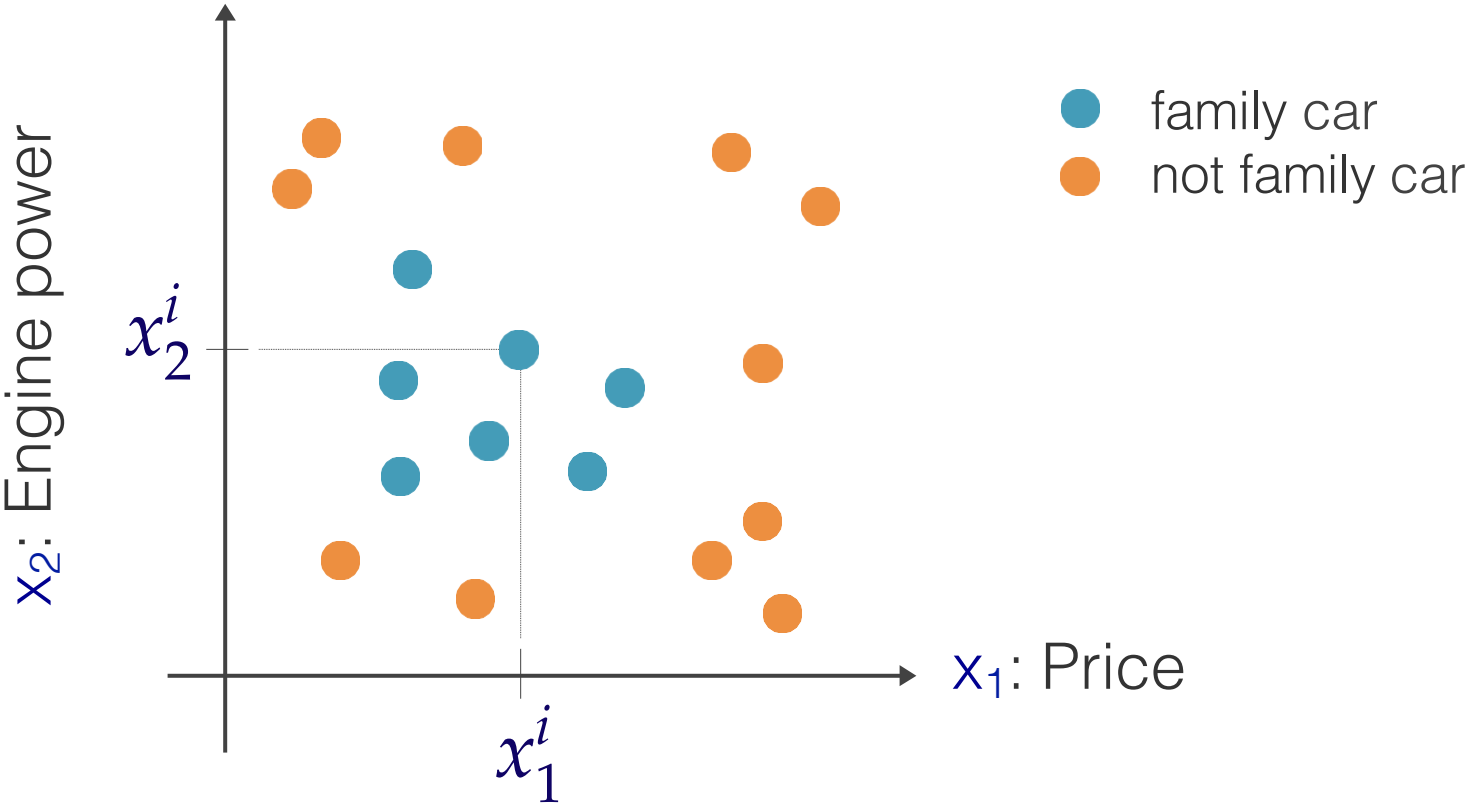
Regression

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

Hypothesis class, loss function, and risk minimization

Hypothesis Class

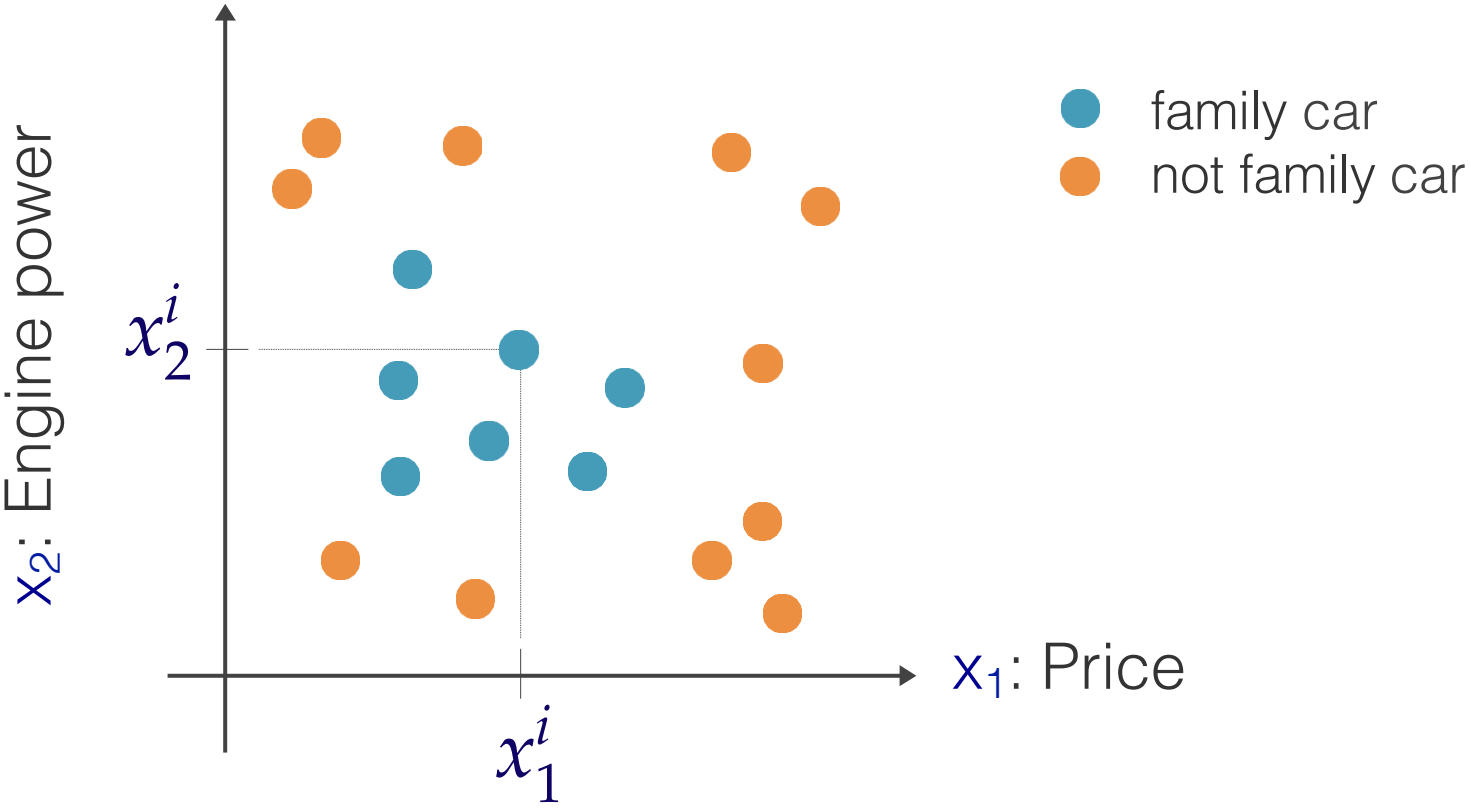
- Hypothesis class $\mathcal{F}$
  - The space of possible decision functions we are considering
  - Chosen based on our beliefs about the problem



Hypothesis Class

- Hypothesis class $\mathcal{F}$

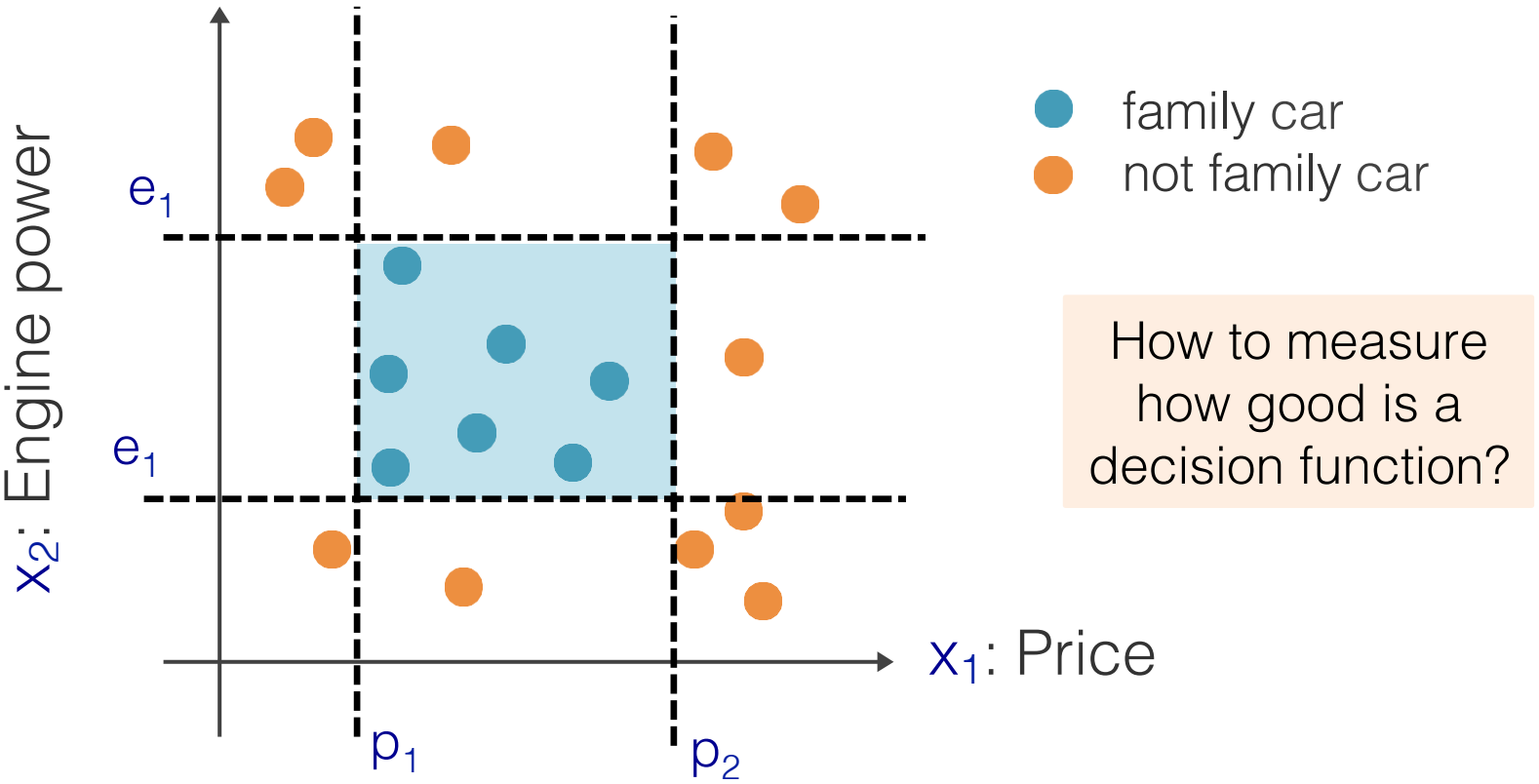
What shape do you think the discriminant should take?



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

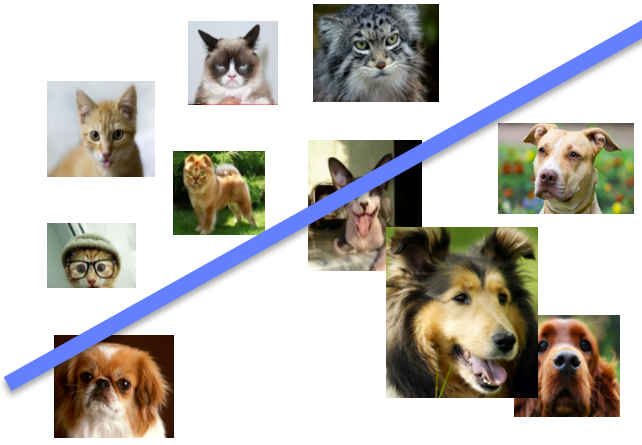
$$(\mathbf{x}^i, y^i), \ldots, (\mathbf{x}^n, y^n)$$

- 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

The Goal of Training (4/4)

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

By averaging the loss function on the training set

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

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

Generalization

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

- "Predictions are hard, especially about the future" — Niels Bohr

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

More details in next lecture

Optimization

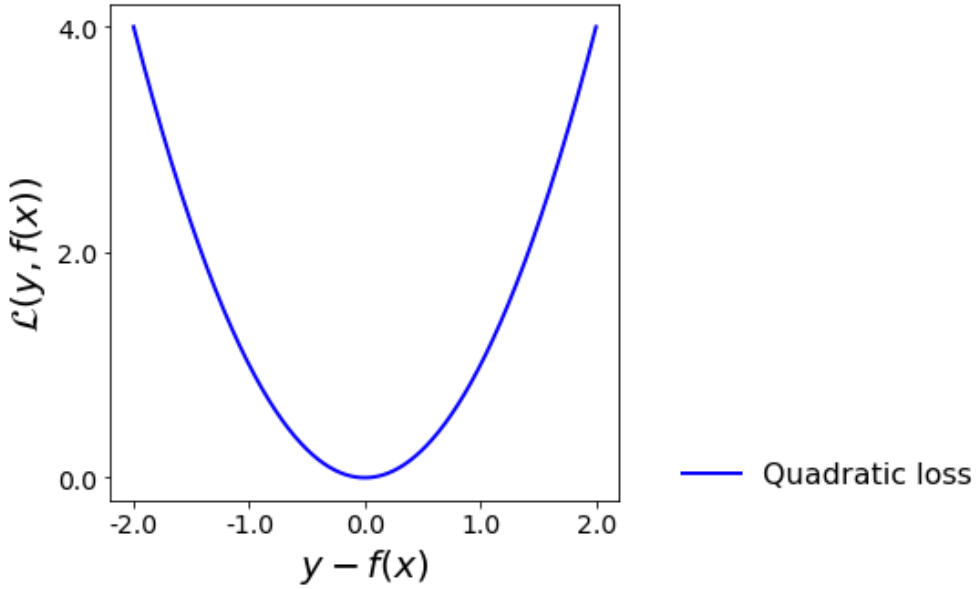
Why Optimization? (1/4)

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



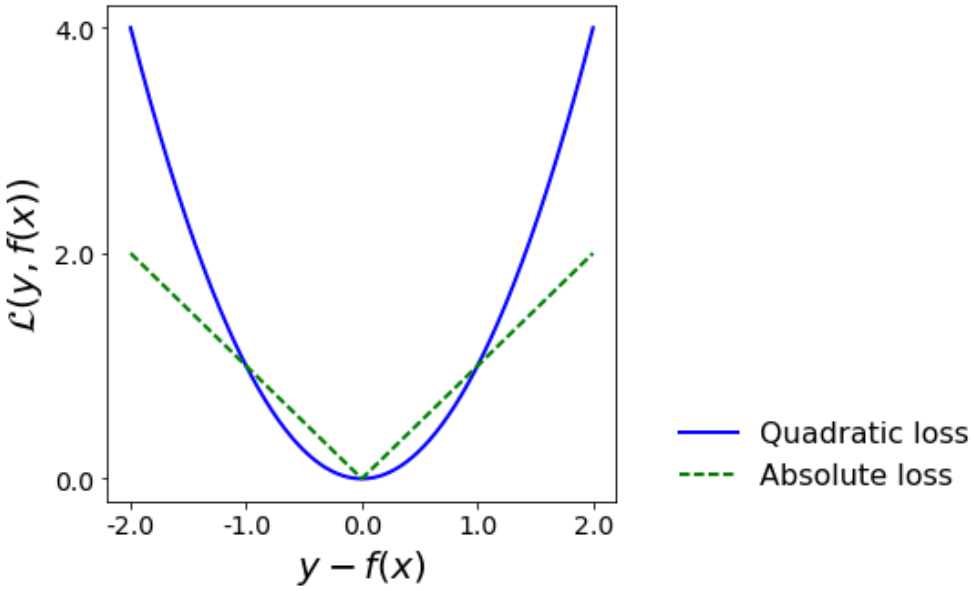
Why Optimization? (2/4)

- 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 m features, $m<p$, such that the data projected on these m 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 lecture

Convex Set

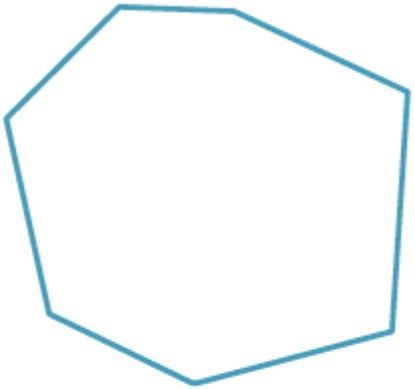
- $\mathcal{S} \subseteq \mathbb{R}^n$ is a convex set iff:

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

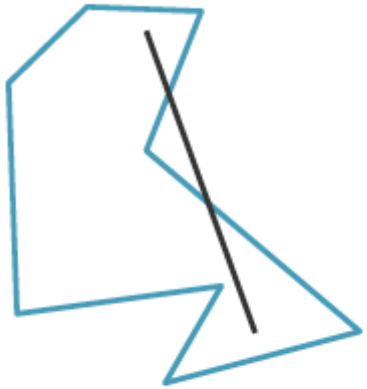
convex combination

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



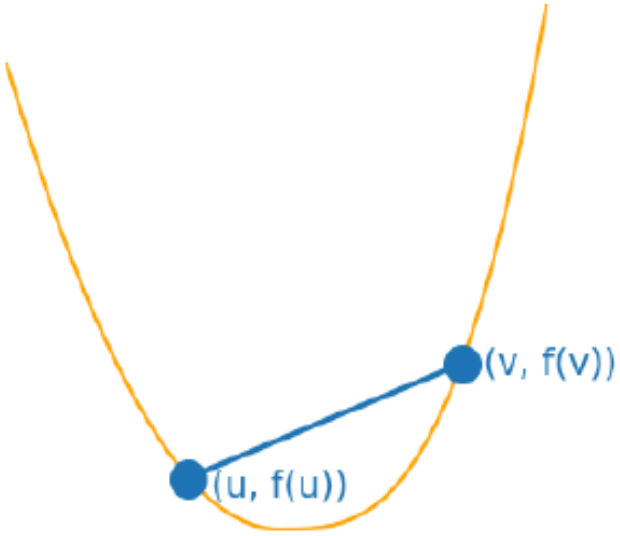
Non-convex set

Convex Function

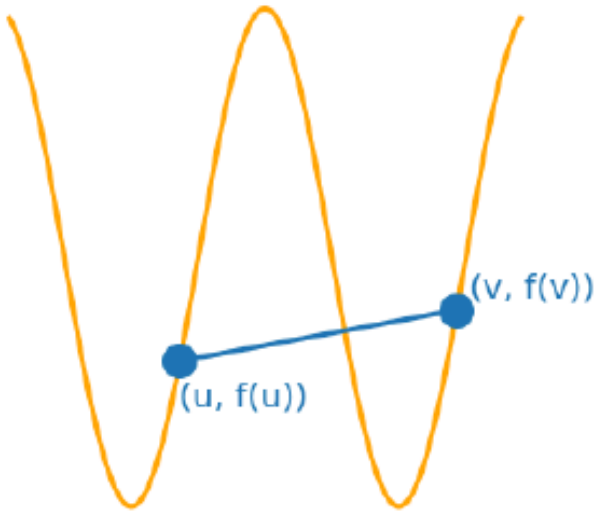
- $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is convex iff:

  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

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

1. Its domain is a convex set
2. $f(t\mathbf{u} + (1-t)\mathbf{v}) \geq tf(\mathbf{u}) + (1-t)f(\mathbf{v})$ 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 $\mathbf{u}, \mathbf{v} \in \mathrm{dom}(f)$

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

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

Gradient of f

What does it mean if:

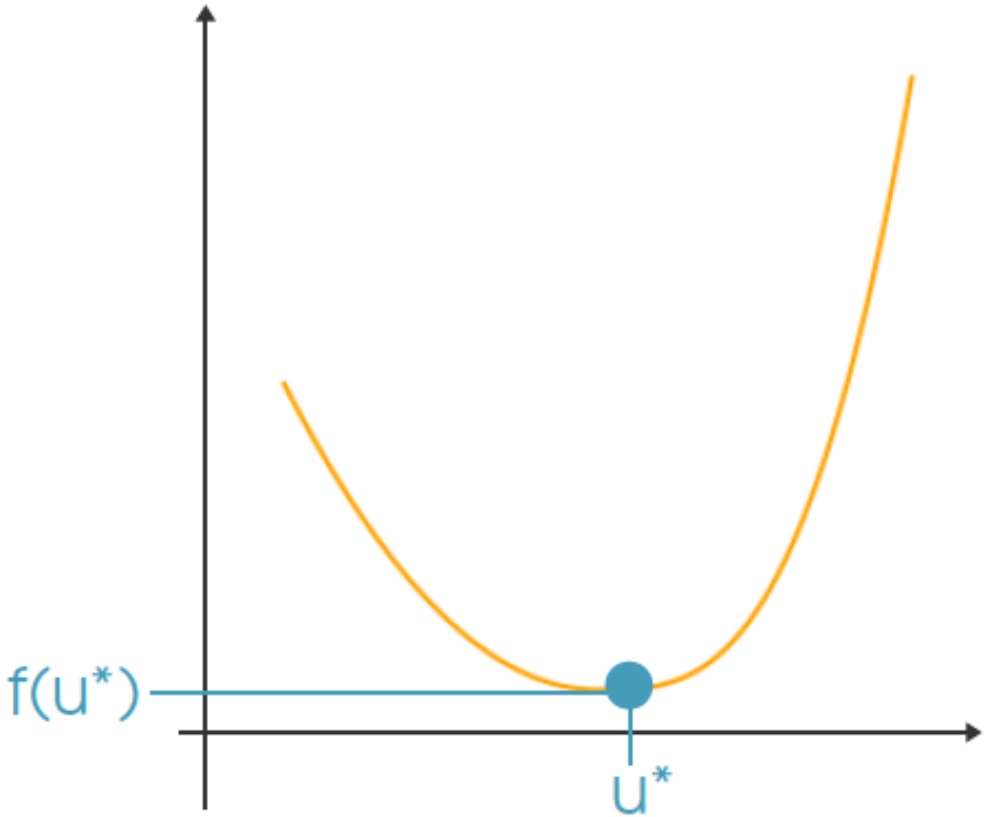
$$\nabla f(\mathbf{u}) = 0$$

$\mathbf{u}$ minimizes f

Unconstraint Convex Optimization

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

where f is convex



Constraint Convex Optimization (1/2)

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

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

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

- f is convex
- g_i, i =1,…,m are convex
- h_j, j =1,…,r are affine: $h_j : \mathbf{u} \rightarrow \mathbf{a}_j^T \mathbf{u} + b_j$
- D is the common domain of all functions

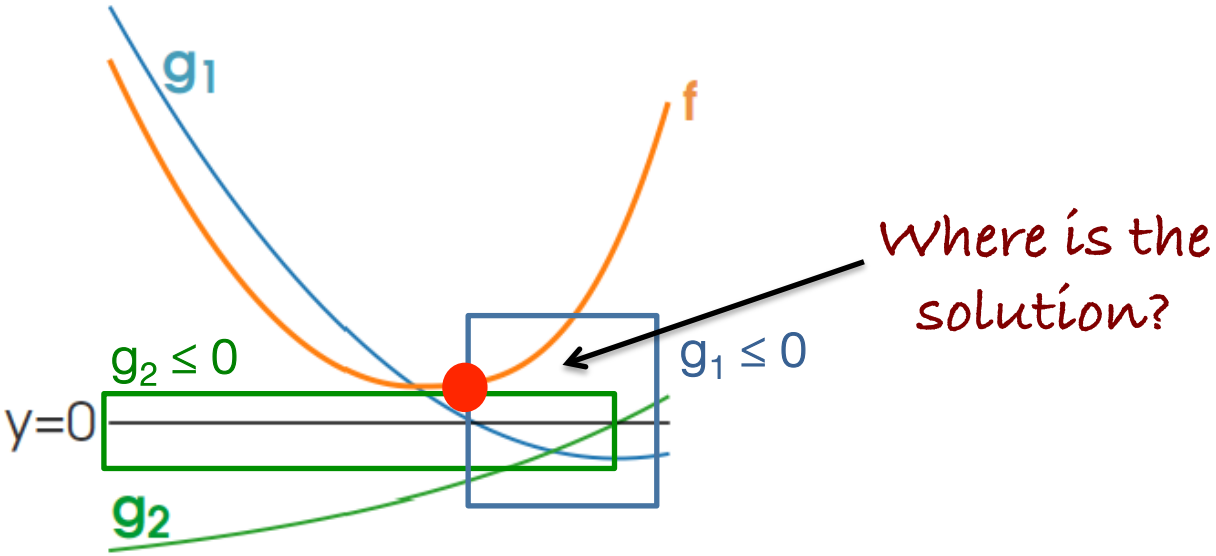
$$D = \text{dom}(f) \cap \bigcap_{i=1}^{m} \text{dom}(g_i) \cap \bigcap_{i=1}^{r} \text{dom}(h_j)$$

Constraint Convex Optimization (2/2)

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

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

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



Local and Global Minima

For convex optimization problems

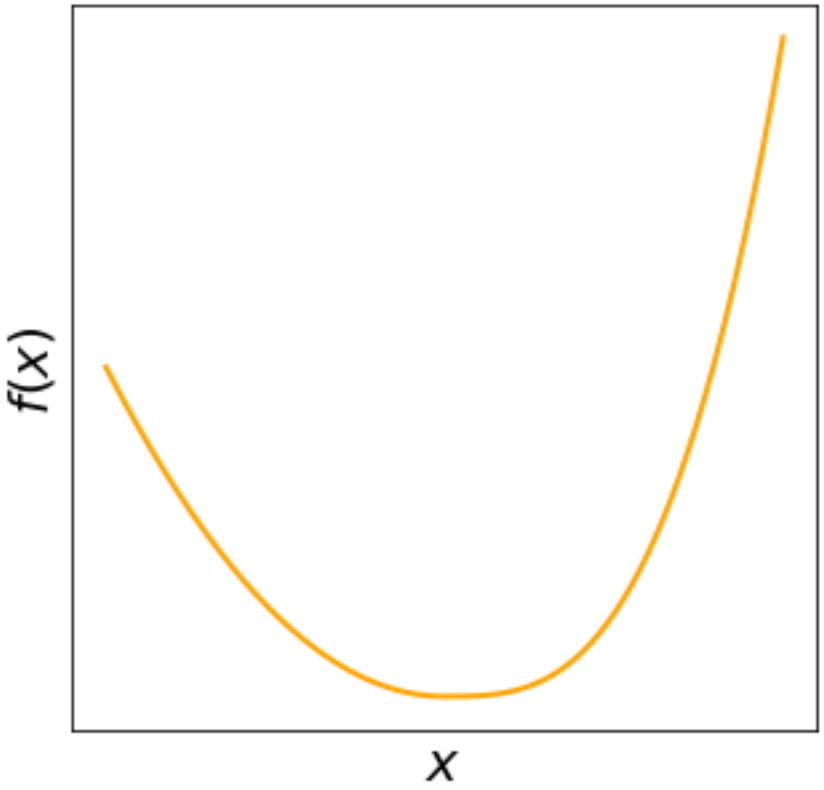
Local minima are global minima

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

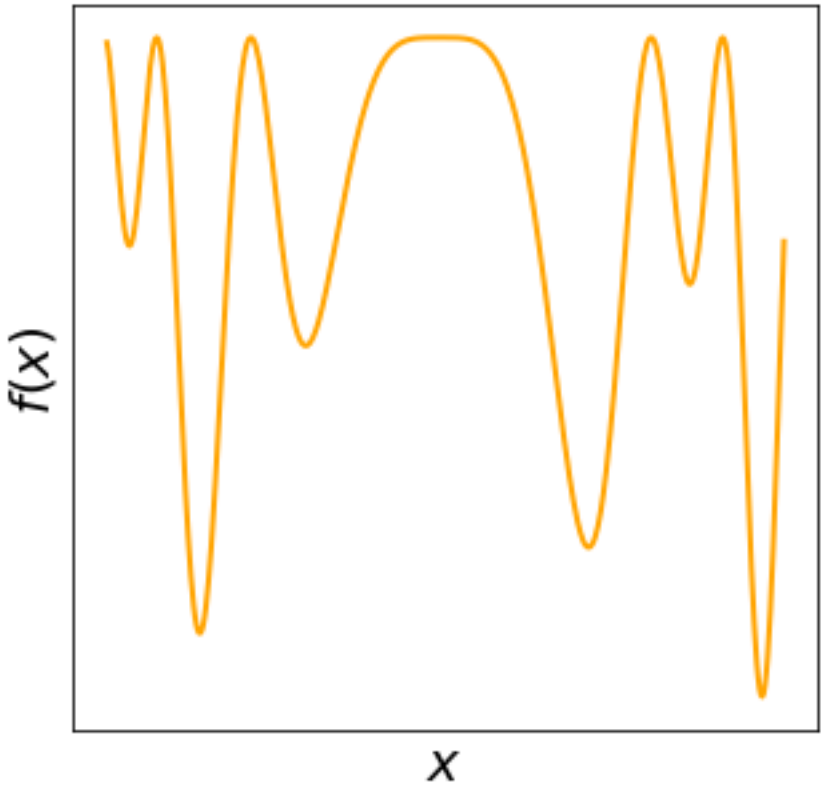
f(u) <= f(v) for all feasible v,

then u maximizes f globally

Why Talk about Convex Optimization?



Convex function



Non-convex function

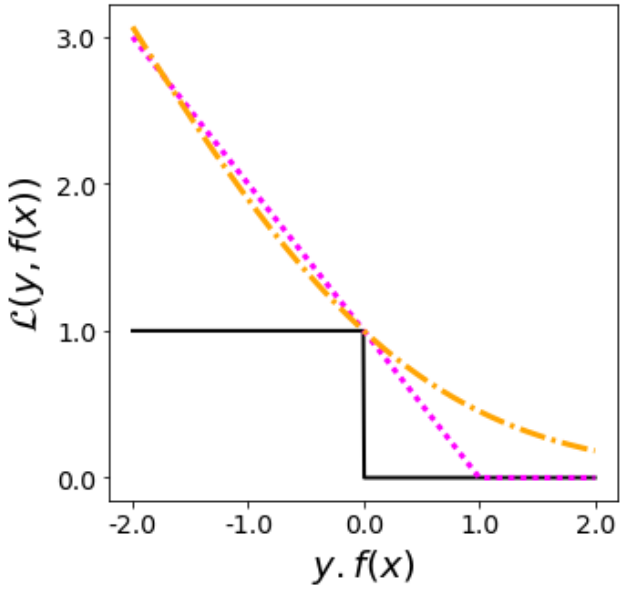
- Convex optimization is easy
- We will often try to formulate ML problems as convex optimization problems

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:

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

$$\downarrow$$

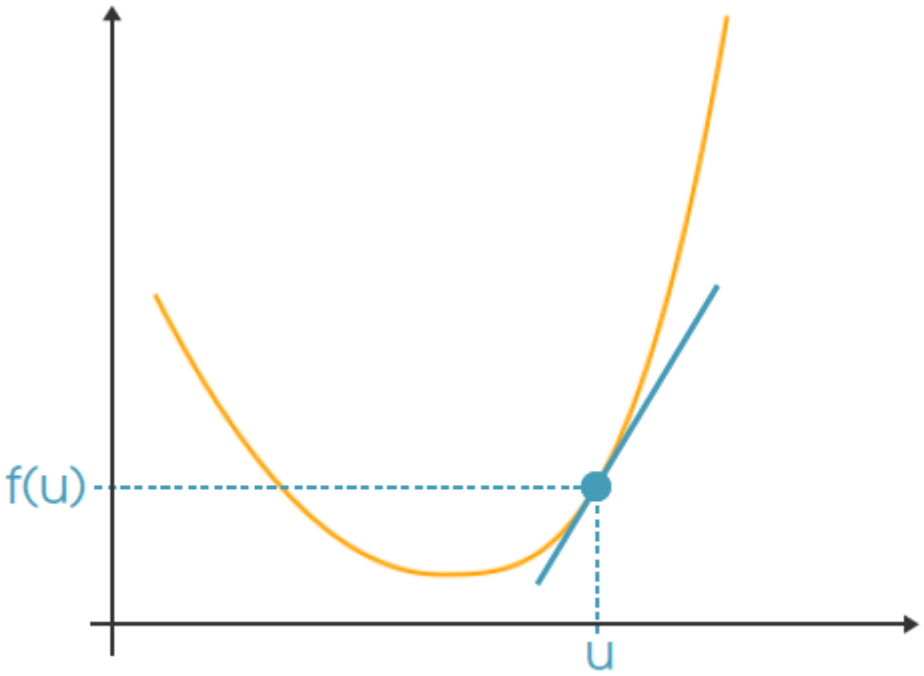
$$\mathbf{u} \text{ minimizes } f$$

→ Set the gradient of **f** to 0

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

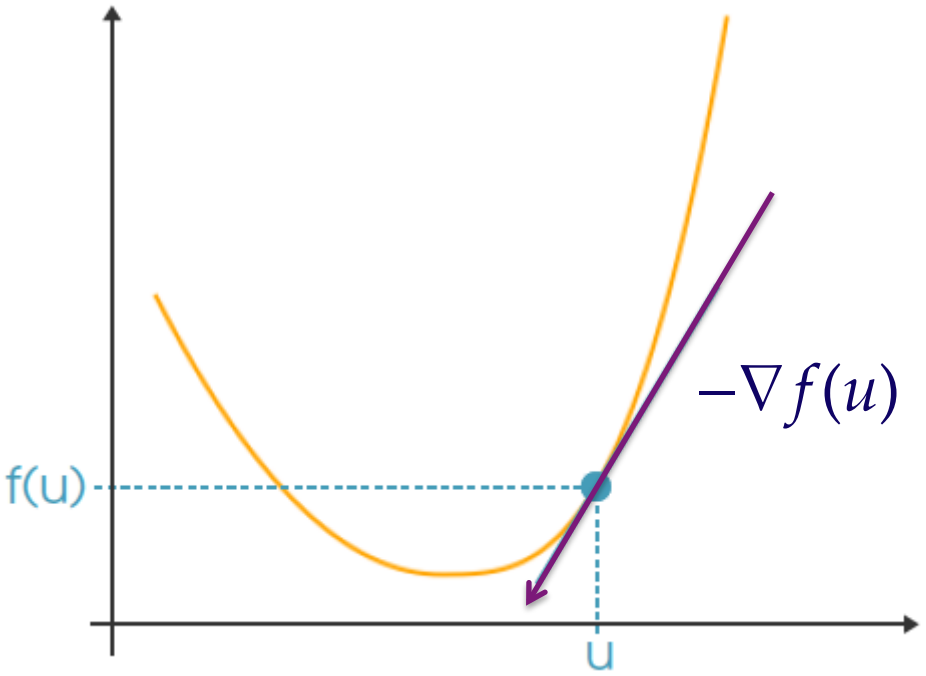
Gradient Descent – The Idea (1/3)

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



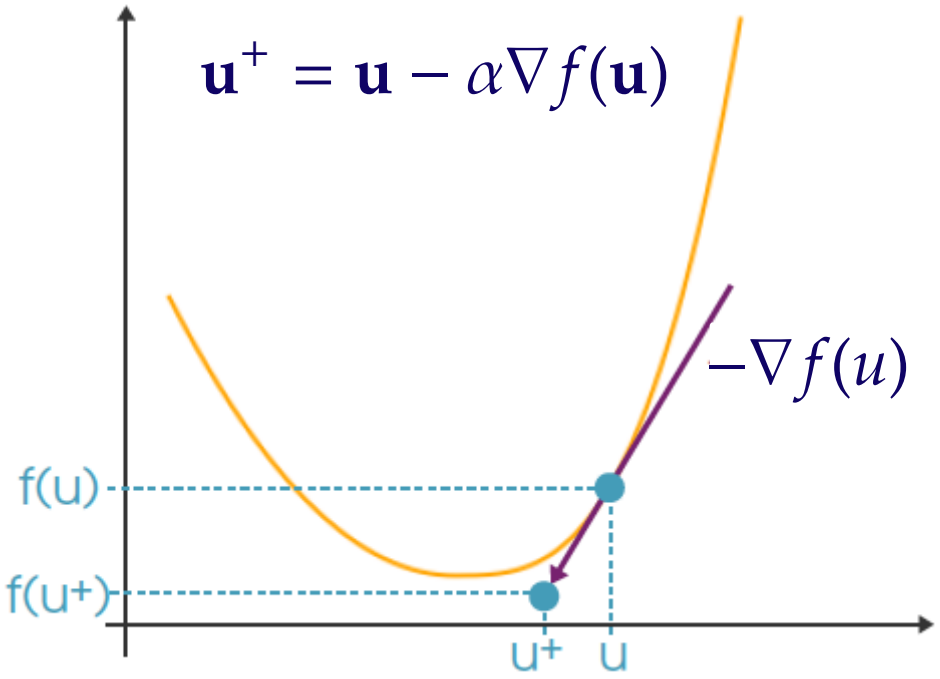
Gradient Descent – The Idea (2/3)

- 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, …

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

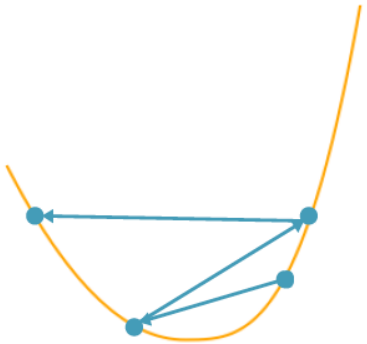
step size

- 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:

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



Next Lecture

- Lecture: dimensionality reduction

- Lab: implementation of dimensionality reduction techniques in Python

Thank You!