COMP 204: Python programming for life sciences

Introduction to machine learning

Yue Li

Outline

What is machine learning?

Machine learning (ML) is a subset of artificial intelligence (AI).
However, the line between ML and AI is becoming blurred.
What AI are:

- More generic on building automated intelligent agents that can both learn from external environment and also "think internally" (not well defined)

What ML are:

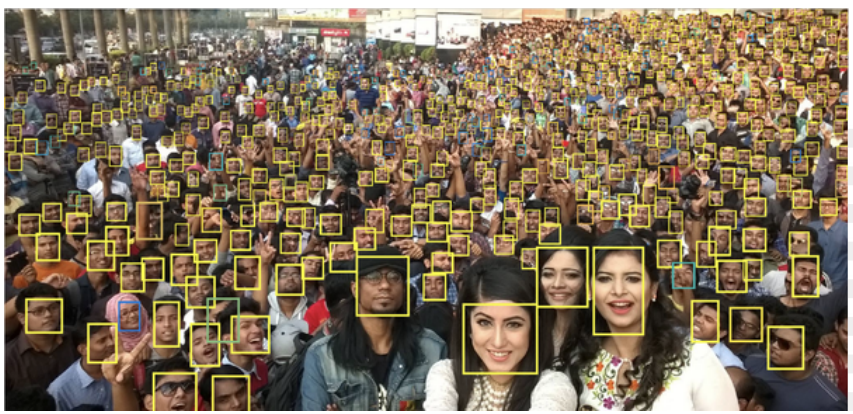
- *tools* that allow us to perform tasks that are hard to solve by traditional programming
- ways to represent complex structures in the massive amount data by simple interpretable patterns
- data-driven (as opposed to rule-based), leading to novel scientific discoveries

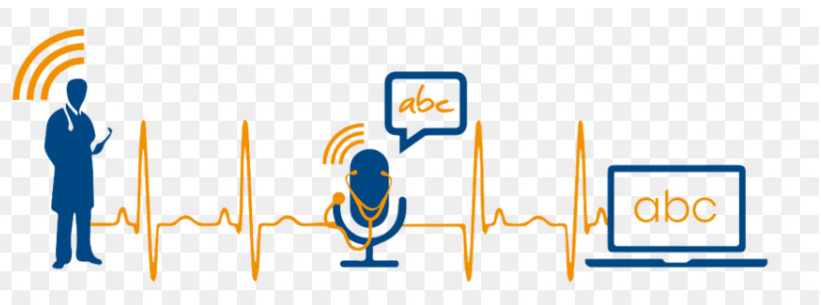
What ML are *not*:

- Computational neuroscience (use math to study brains)
- Rule-based system involving human-crafted rules
- Terminator (neither is AI)

Problems that cannot be solved by typical programs 1

- Typically, programs are written by human in exact sequential order (what we have seen so far)
- However, it is hard to write programs that solve tasks like recognizing faces or speech:
  - We don't know how to program it because we don't know how our brains work
  - Even if we had a good idea how our brain do it, the program will be horrendously complicated

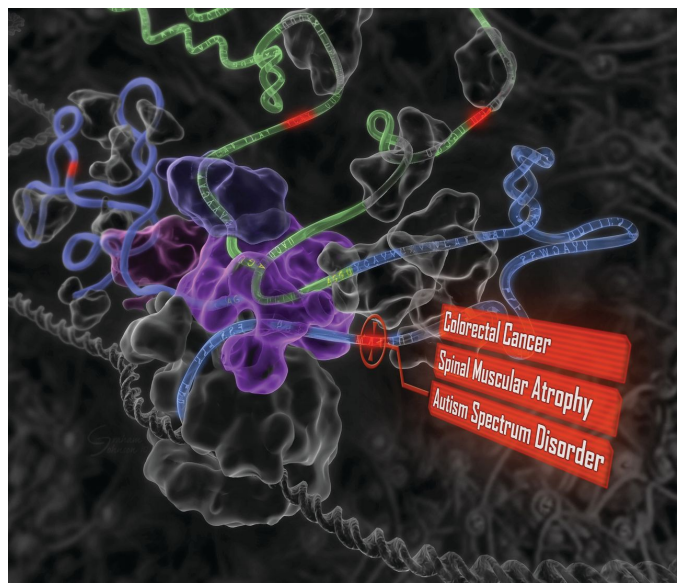




Problems that cannot be solved by typical programs 2

Detecting disease-causing mutations

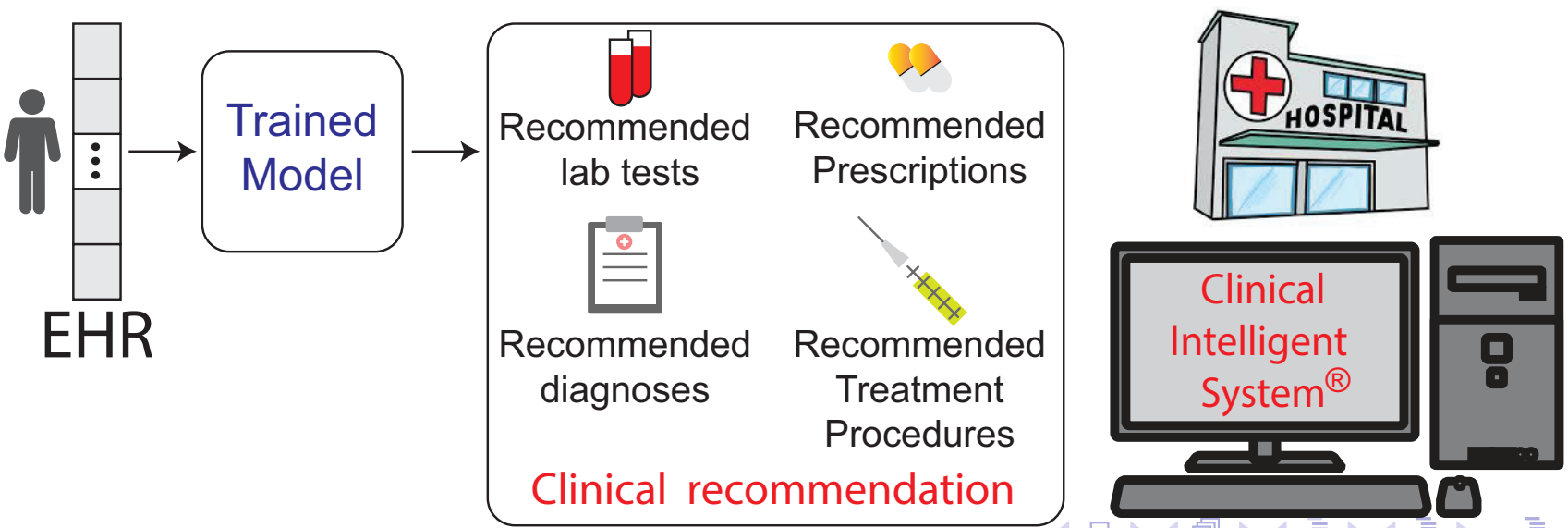
- We don't know how to program it because we don't fully understand the functions of our genome
- We have very limited understanding of the physiology underlying most of the complex phenotypes (e.g. Alzheimer's disease, cancers) and how they interact with the environments (e.g., nutrition, exposed to radiation, neighbourhoods)
- There are unknown causal factors that we may not even observe or not yet have a way to measure them (e.g., uncharacterized pathways)



Problems that cannot be solved by typical programs 3

Building a clinical recommender system:

- We have limited knowledge about how various diseases and symptoms are related to each other (despite ICD taxonomy and other efforts)
- How do we predict future disease onset based on current limited amount of health record for most people?
- How do we handle missing data that are common in healthcare (e.g., unordered lab tests)?



Introduction to machine learning: a data-driven approach

What we do in machine learning:

- We collect lots of data (e.g., genotype of over 1 million genetic variants and phenotypes for large population cohort)
- We develop a machine learning algorithm that take these individual data as "examples" or "training data" and *automatically* produces a program that does the job
- The program produced by the learning algorithm may look very different from a typical hand-written program. It may contain millions of numbers (e.g., one decimal number per mutation indicating their impact on the phenotype)
- If we succeed, the program will work on new data ("testing data") that are not seen before (e.g., predicting risk of Alzheimer's disease using genotype of a new individual)
- In practice, we need **two essential components**:
  1. An objective function that quantitatively characterizes the problem (e.g., sum of errors in predicting diseases)
  2. An algorithm that optimizes the objective function (e.g., minimizing the errors disease prediction)

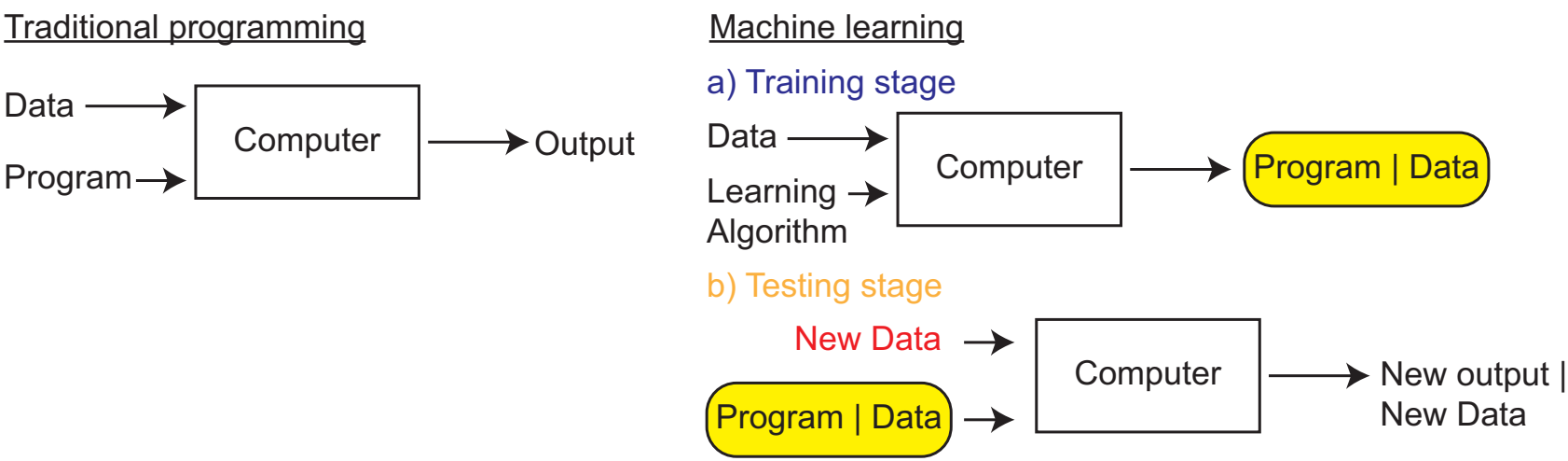
'Traditional' programming vs. machine learning

Traditional programming

- Program is written first *independent* of the data
- Program is applied to data to produce an output
- The program does not adapt to the data: it remains the same throughout its execution

Machine learning

- Program (or parameters of the program) adjusts itself automatically to **fit** the data
- End result is a program that is trained to achieve a given task

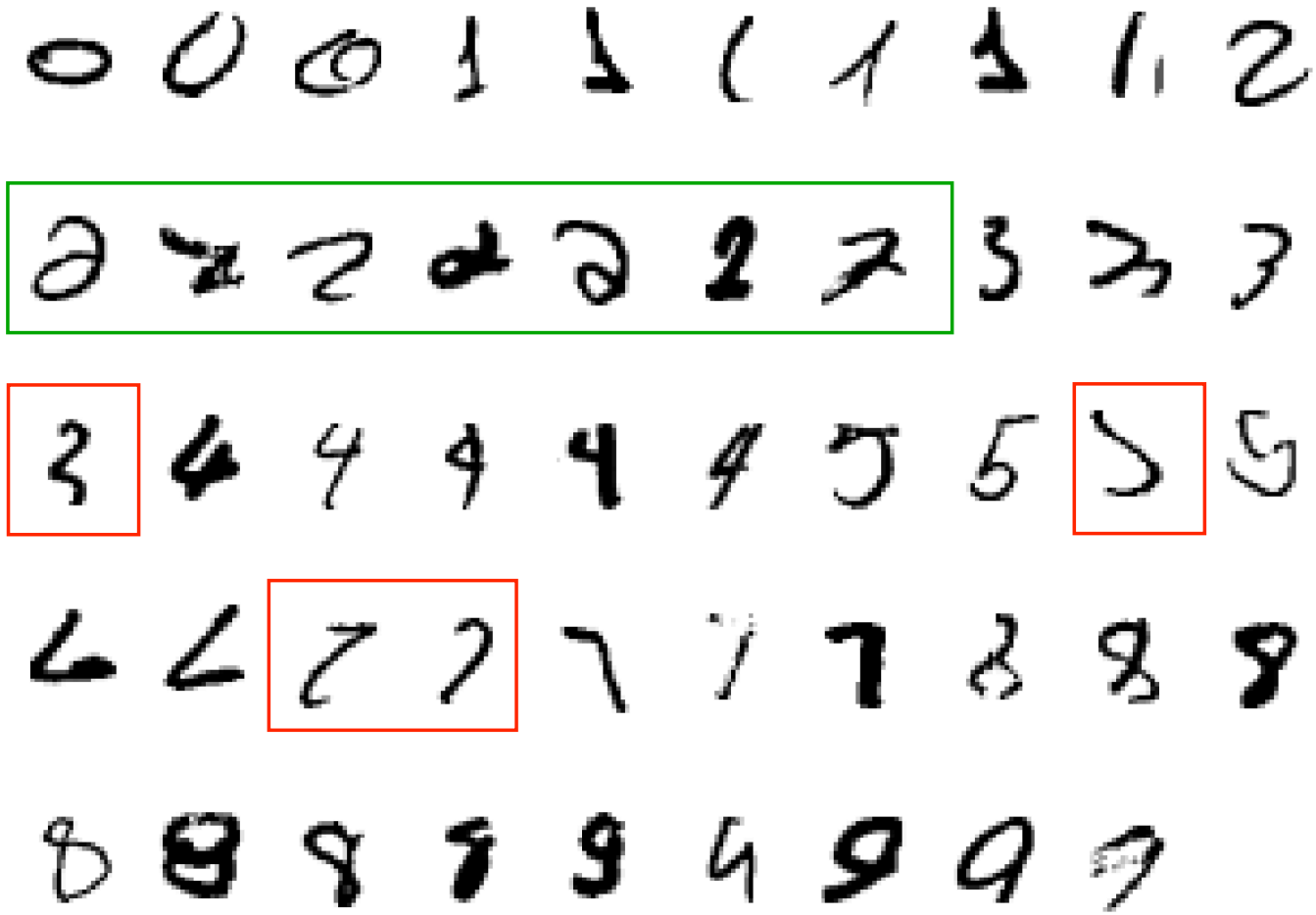


Types of learning tasks

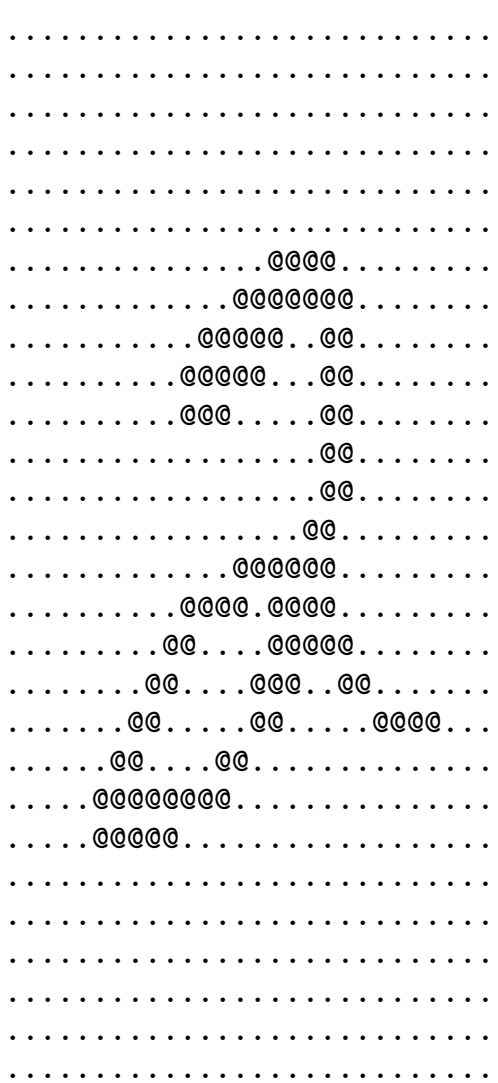
- Supervised learning:
  - Given examples of inputs (e.g., genotype) and corresponding desired outputs (e.g., disease), predict outputs on future unseen inputs, e.g., classification, regression, time series prediction
  - Often the connotation of machine learning (people often ask how accurate is your model?)
- Unsupervised learning
  - Create a new representation of the input, e.g., form clusters, extract latent continuous features, compression
  - This is the new frontier of machine learning because most big datasets do not come with labels
- Reinforcement learning
  - Learn action to maximize payoff (e.g., robotics, self-driving vehicle)
  - An important research area but not the focus of this class

A classic example in machine learning: recognizing hand-written digits

What makes a "2"?



First, how do we represent the data such as images: matrix



We represent each 28x28 image as a matrix or a two-dimensional list (see Lecture28.ipynb) Each entry in the 2D list is either 0 ('.') or number that is greater than 0 ('@')

```
1  from mnist import MNIST
2  import random
3  mndata = MNIST('mnist')
4  images, labels =
   ↪  mndata.load_training()
5  print(type(images)) # <class 'list'>
6  # display the first `2' in the data
7  for index,digit in enumerate(labels):
8      if digit == 2:
9          break
10 print(mndata.display(images[index]))
```

Outline

Classification

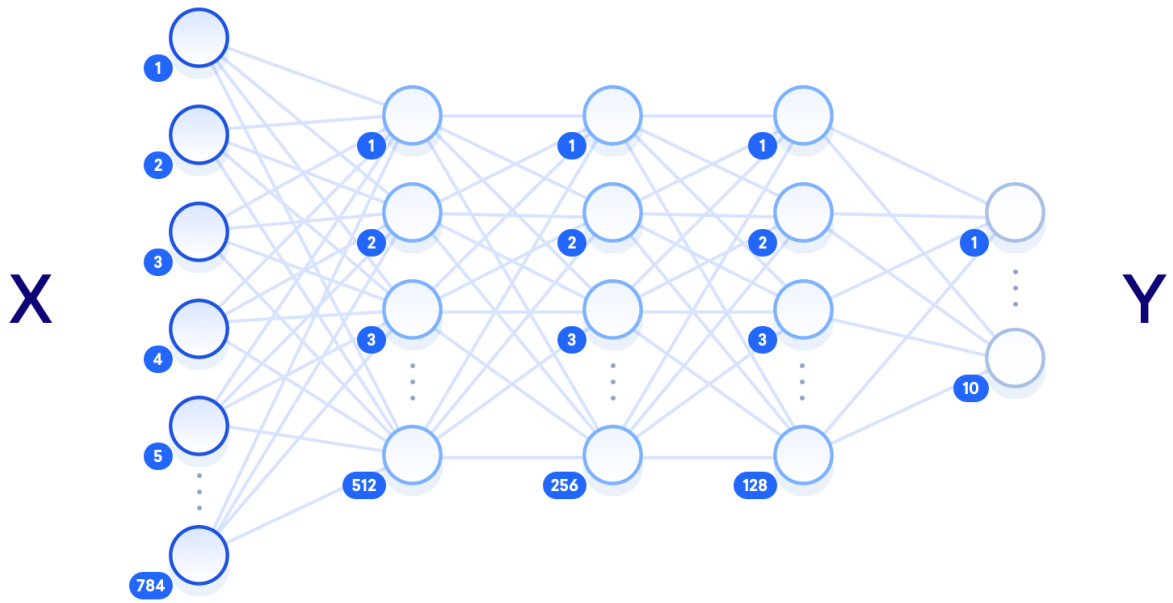
- Outputs or labels are categorical (1-of-N) (e.g., 10 digits)
- Inputs: feature (e.g., flattened 28×28 image $= 784$ input features)
- Goal: select the correct class for the new inputs (e.g., correctly classify the new image into one of the 10 digits)

How to represent labels? Use "one-hot encoding": create a vector as long as the number of categories we have, and set exactly one of the positions in the vector to 1 and the rest to 0.

```
# This is the one-hot version of: [5, 0, 4, 1, 9]
"""
[[0, 0, 0, 0, 0, 1, 0, 0, 0, 0]
 [1, 0, 0, 0, 0, 0, 0, 0, 0, 0]
 [0, 0, 0, 0, 1, 0, 0, 0, 0, 0]
 [0, 1, 0, 0, 0, 0, 0, 0, 0, 0]
 [0, 0, 0, 0, 0, 0, 0, 0, 0, 1]]
"""
```

Classifying MNIST hand-written image with neural network

Classification of flattened 28×28 input image (784 features) X into one of the 10 categories (digits) Y:



Simple math behind neural network (not required to understand in this class):

$$h_j^{(1)} = \sum_{i=1}^{784} w_i^{(1)} x_i; \qquad h_j^{(m)} = \sum_{i=1}^{H_m} w_i^{(m)} h_i^{(m-1)}$$

$$\hat{y}_k = \frac{\exp(-\sum_{i=1}^{M=128} w_k a_i)}{\sum_{k'=1}^{10} \exp(-\sum_{i=1}^{M=128} w_{k'} a_i)}; \quad CE = -y_k \log \hat{y}_k - (1 - y_k) \log(1 - \hat{y}_k)$$

Modern neural network computing infrainstructure

- Neural network also known as "Deep learning" models are typically implemented in a number of different libraries, including: Theano, Tensorflow, or Torch.
- These libraries typically have the following features:
  - Automatic or symbolic differentiation on computational graphs for backpropagation
  - Compilation for CUDA (GPU), enabling speedups due to highly optimized and parallel implementations of core NN functions
- High-level neural network libraries wraps on top of Theano and Tensorflow simplifies neural net creation (Keras, Lasagene)
- More powerful GPUs + Easily accessible cloud GPU computing (e.g., Amazon Web Services (AWS)) $\rightarrow$ More sophisticated and powerful models!

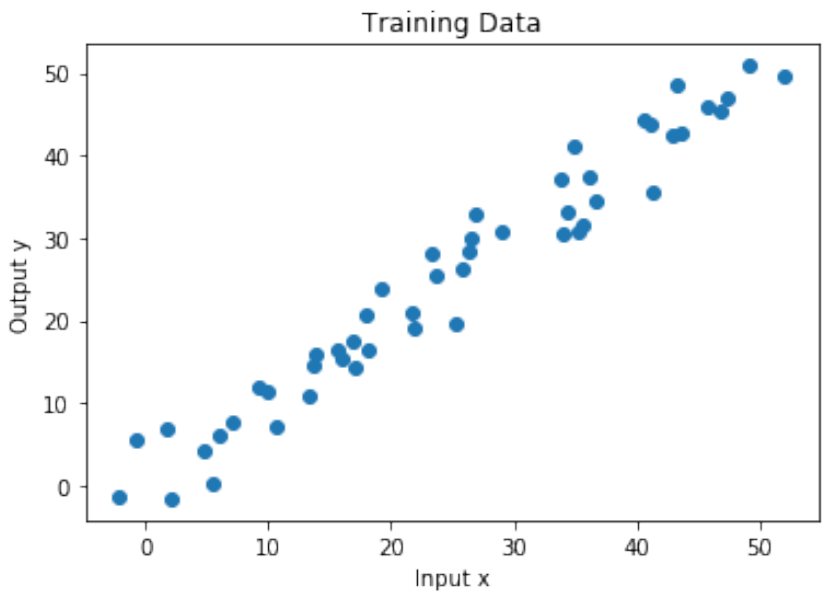
Deep learning on classifying MNIST digits (8 lines of code)

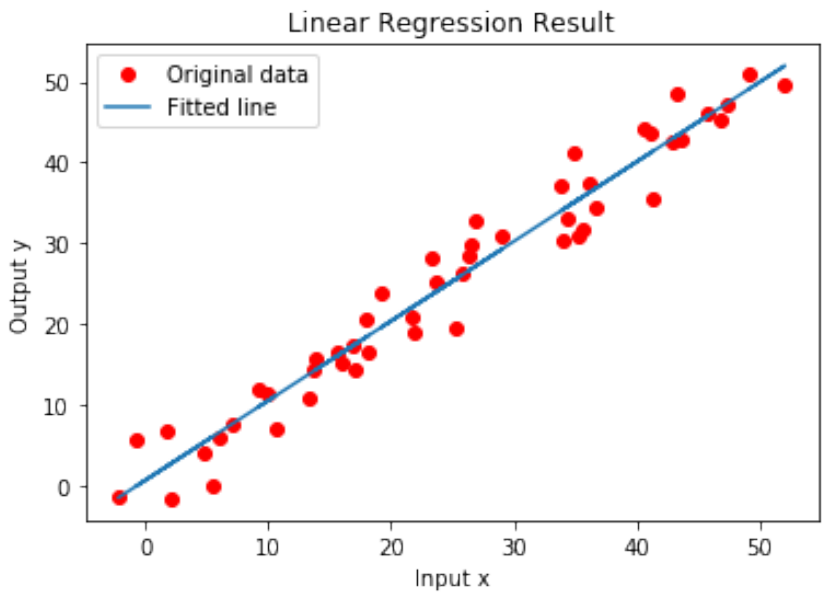
```
1  import tensorflow as tf
2  mnist = tf.keras.datasets.mnist
3
4  (x_train, y_train),(x_test, y_test) = mnist.load_data()
5  x_train, x_test = x_train / 255.0, x_test / 255.0
6
7  model = tf.keras.models.Sequential([
8    tf.keras.layers.Flatten(input_shape=(28, 28)),
9    tf.keras.layers.Dense(128, activation='relu'),
10   tf.keras.layers.Dropout(0.2),
11   tf.keras.layers.Dense(10, activation='softmax')
12 ])
13
14 model.compile(optimizer='adam',
15               loss='sparse_categorical_crossentropy',
16               metrics=['accuracy'])
17
18 model.fit(x_train, y_train, epochs=5)
19 model.evaluate(x_test, y_test)
```

Outline

Regression

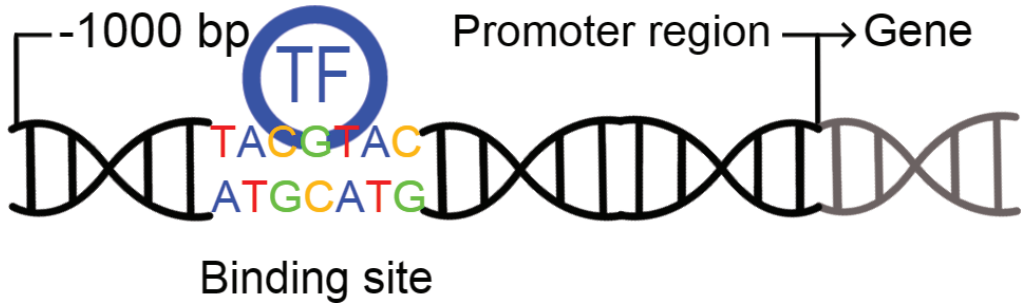
- Outputs are continuous
- Inputs: feature (continuous or discrete)
- Goal: predict outputs accurately for new inputs
- A toy example (see Lecture28.ipynb):
  - Output (y): a continuously measured variable
  - Input (x): another continuously measured variable
  - Task: fitting a line as $y = wx + b$ by estimating w and b





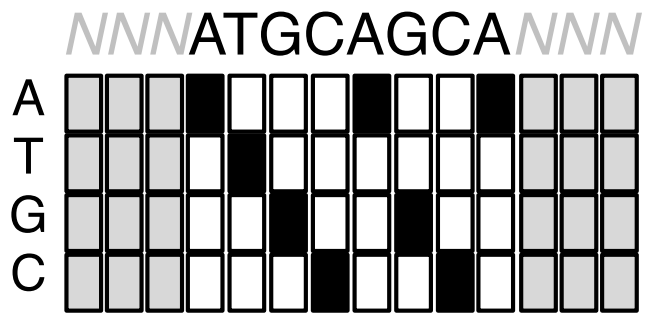
Example: predict transcription factor binding affinity

Transcription factors (TF) are proteins that bind to specific regions of the genome to regulate nearby gene expression



Goal: predict the TF binding affinity based on the DNA sequence

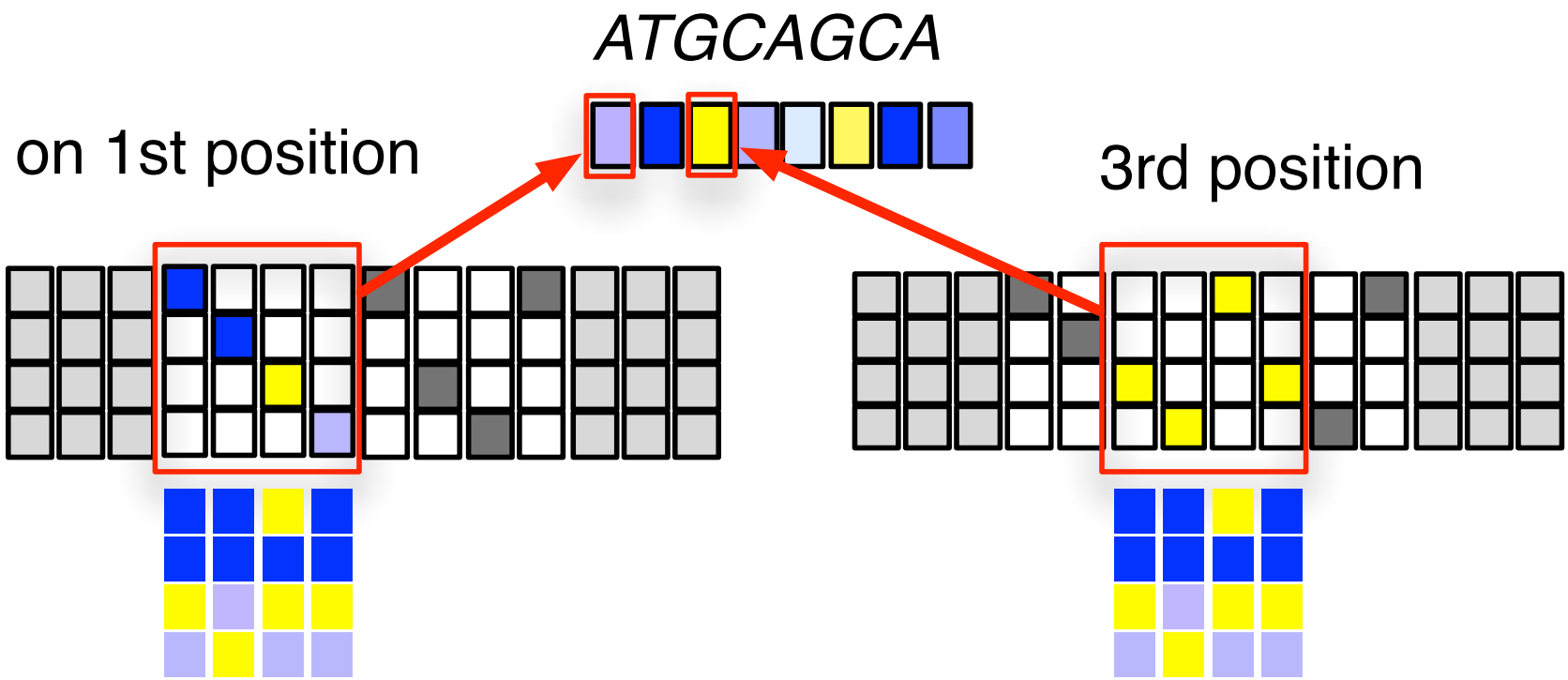
Input: representing DNA sequence as 2D matrix:



Matrix representation of DNA sequence (darker = stronger)

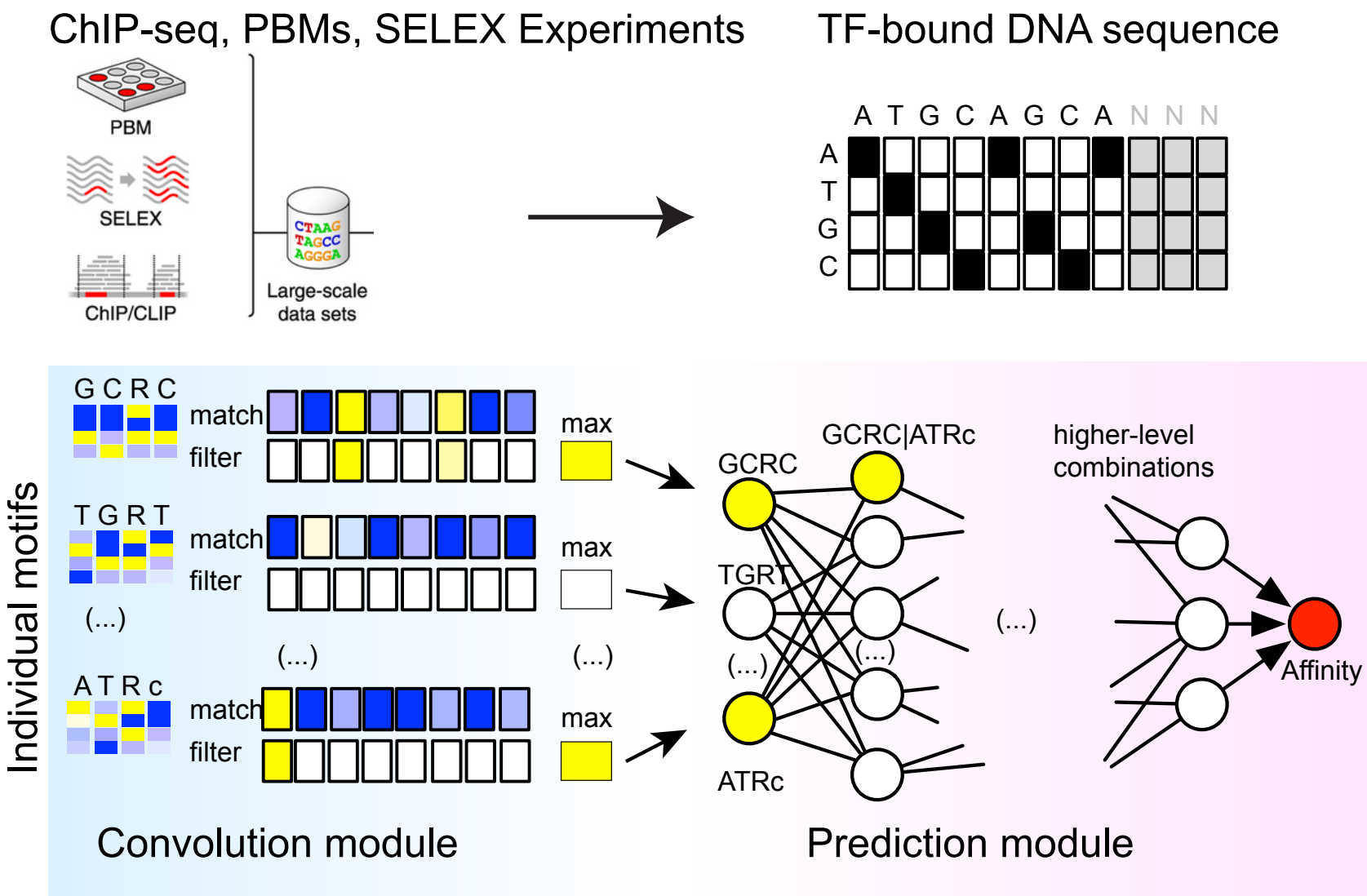
Convolutional neural network (CNN)

Applying 4 bp sequence filter along the DNA matrix:

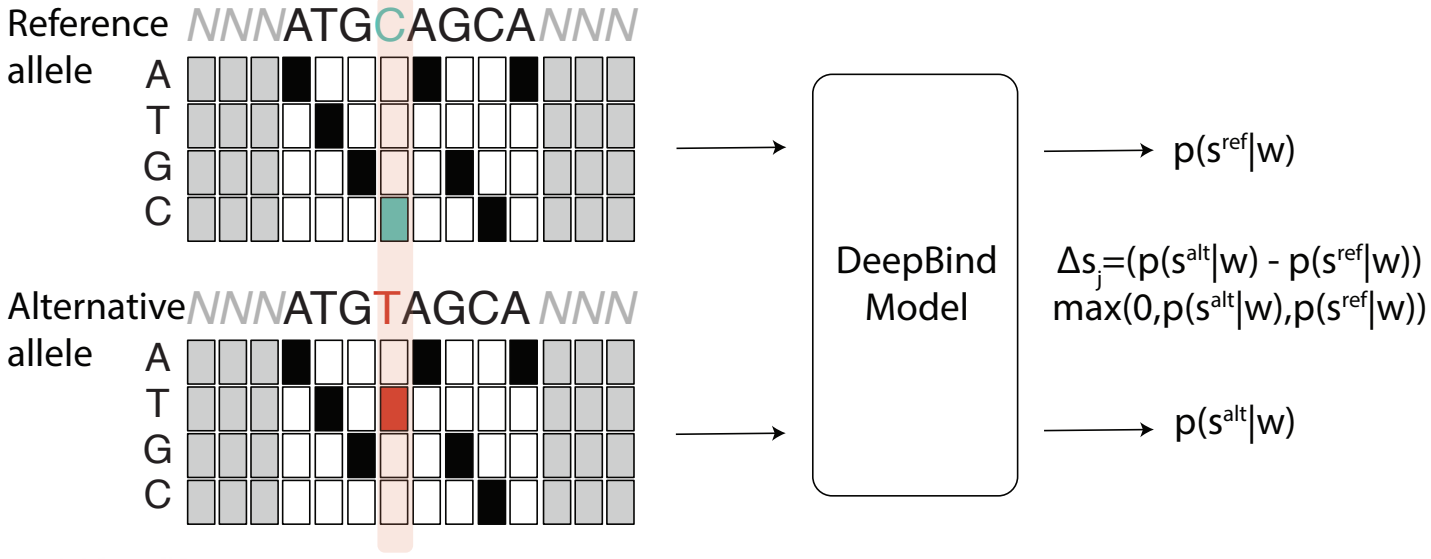


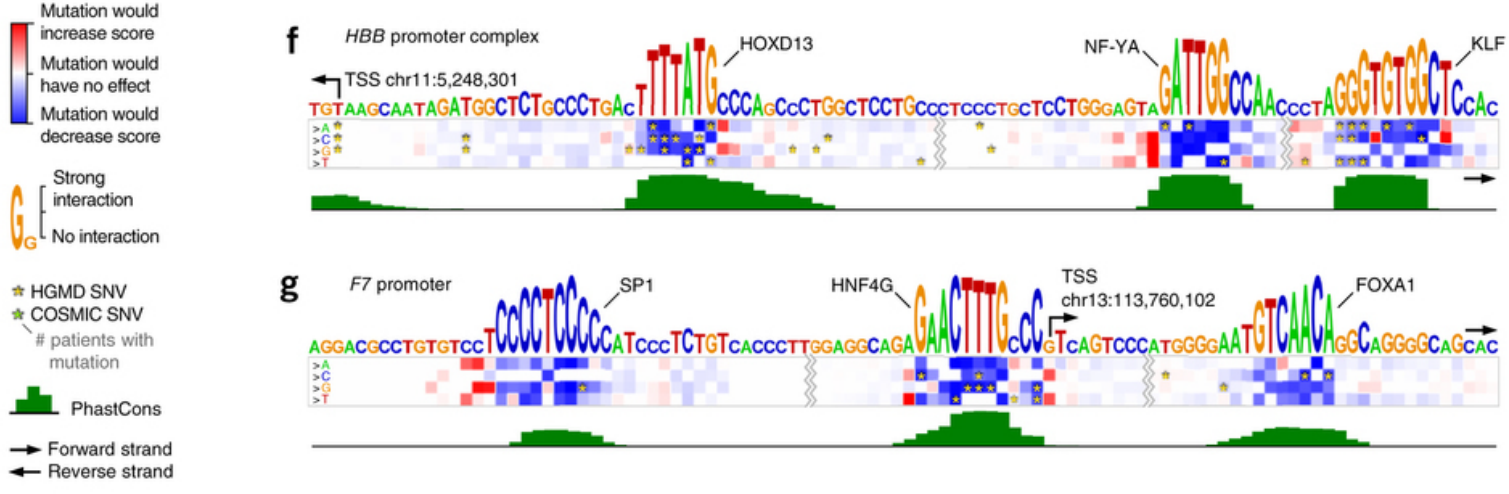
Yellow = high activity; blue = low activity

Predicting transcription factor binding affinity



in-silico prediction of mutation impact [Alipanahi 2015]





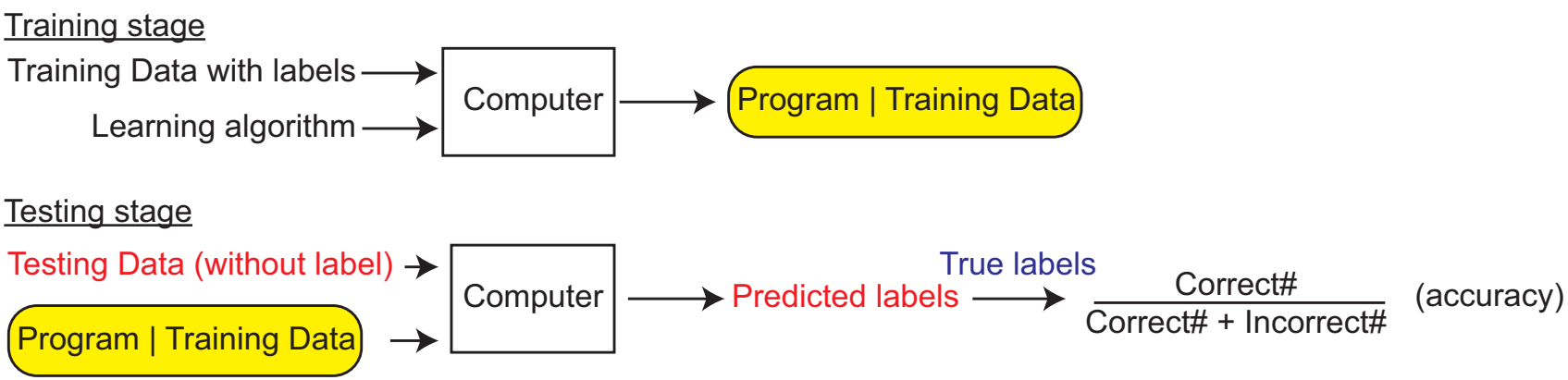
Outline

Evaluating machine learning algorithms

- How can we get an *unbiased* estimate of the accuracy for a learned model?
- Goal: Estimate accuracy of predictor on examples it has not seen as part of its training.

Training data vs Testing data

- split available data into **training** and **testing** datasets
- create a learned model from the training data
- measure accuracy of trained model by applying it to the testing data



Outline

Clustering

- Inputs: continuous or categorical data
- Goal: group data cases into a finite number of clusters so that within each cluster all cases have very similar
- One of the simplest algorithm is called: k-means clustering
- following slides are based on Bishop 2006 textbook: Pattern recognition and machine learning

- Setting:
  - data: $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$
  - goal: partition the data into K clusters
  - objective function in K-means:

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} ||\mathbf{x}_n - \boldsymbol{\mu}_k||^2 \tag{1}$$

- Algorithm:
  1. initialize K cluster centers $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K$
  2. assign each point $\mathbf{x}_n$ to the closest center k:

$$r_{nk} = \begin{cases} 1 & \text{if } k = \arg\min_j ||\mathbf{x}_n - \boldsymbol{\mu}_j||^2 \\ 0 & \text{if otherwise} \end{cases}$$

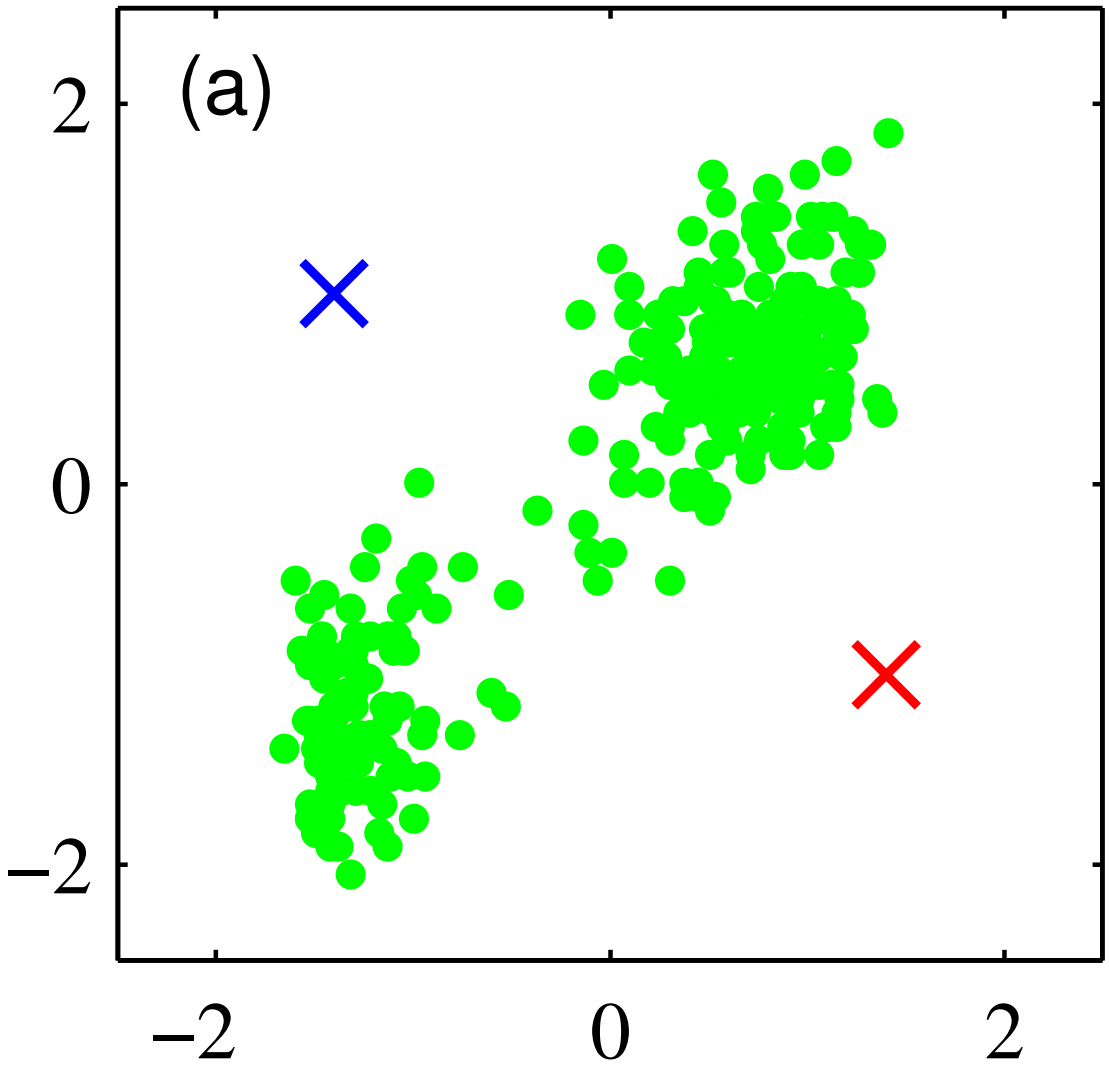
  3. update cluster centers:

$$\boldsymbol{\mu}_k = \frac{\sum_n r_{nk} \mathbf{x}_k}{\sum_n r_{nk}}$$

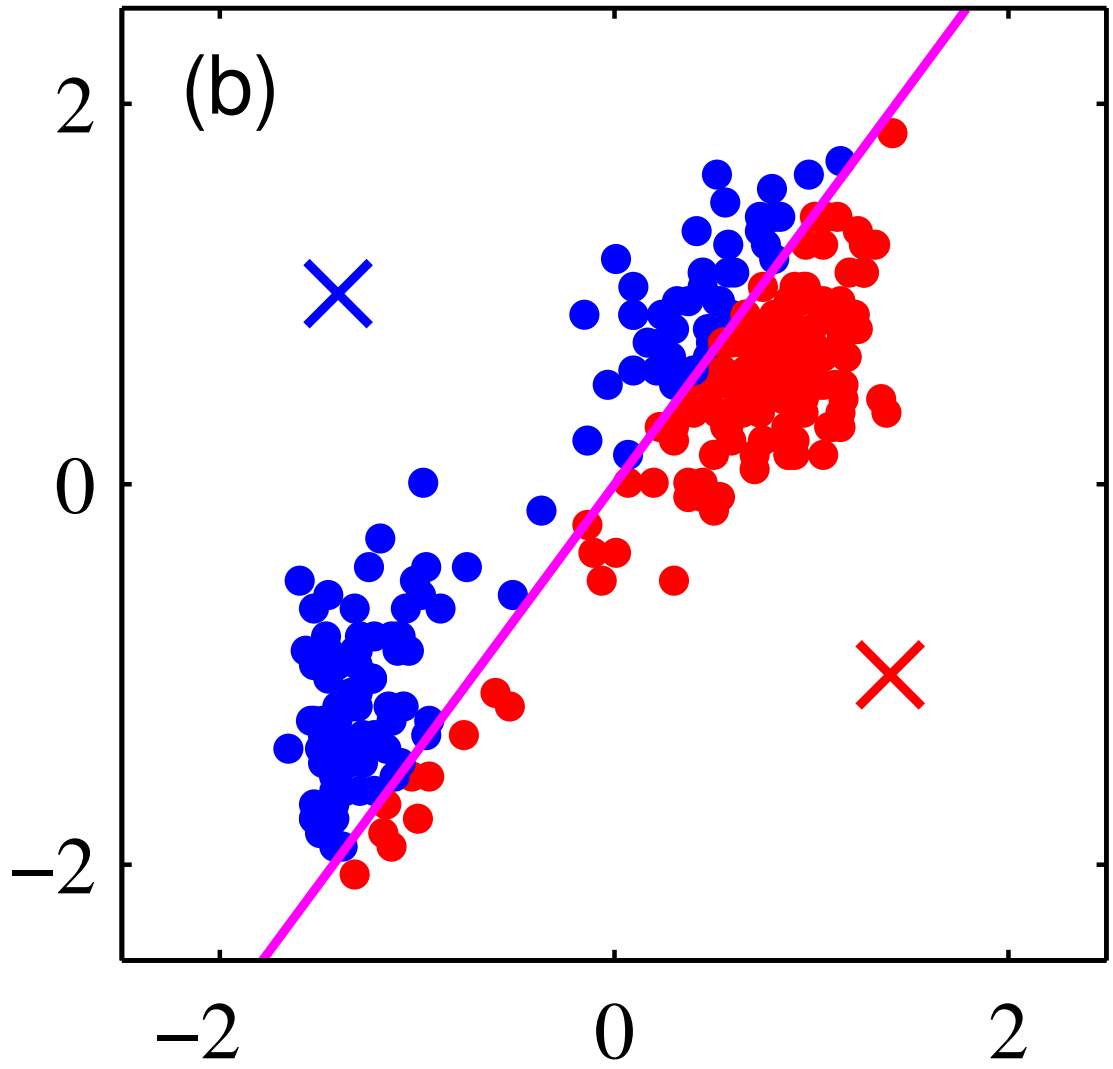
  4. repeat 2 & 3 until convergence (i.e. little change from J (1))

Pattern recognition and machine learning (Bishop, 2006)

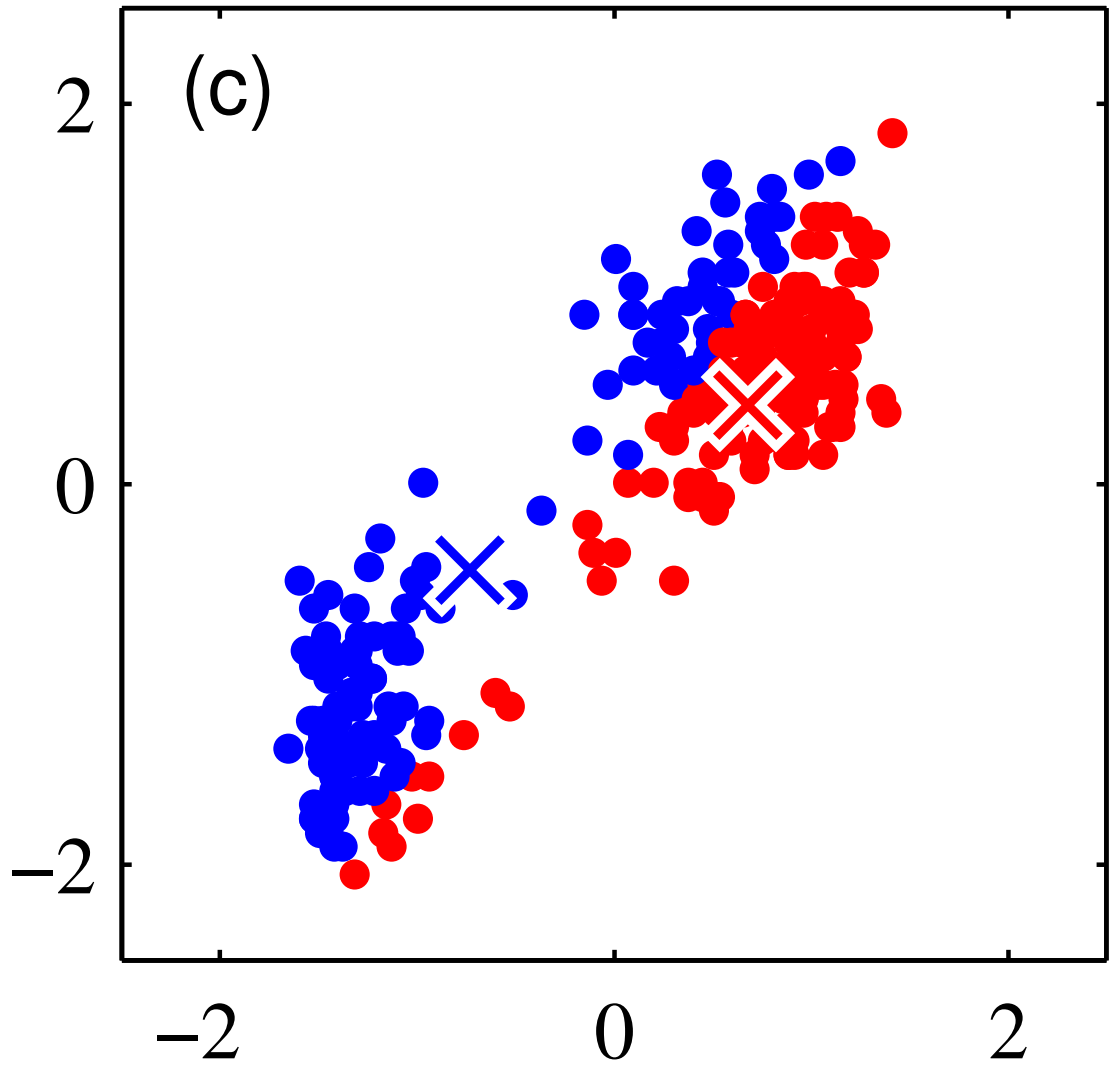
K-means clustering (K=2)



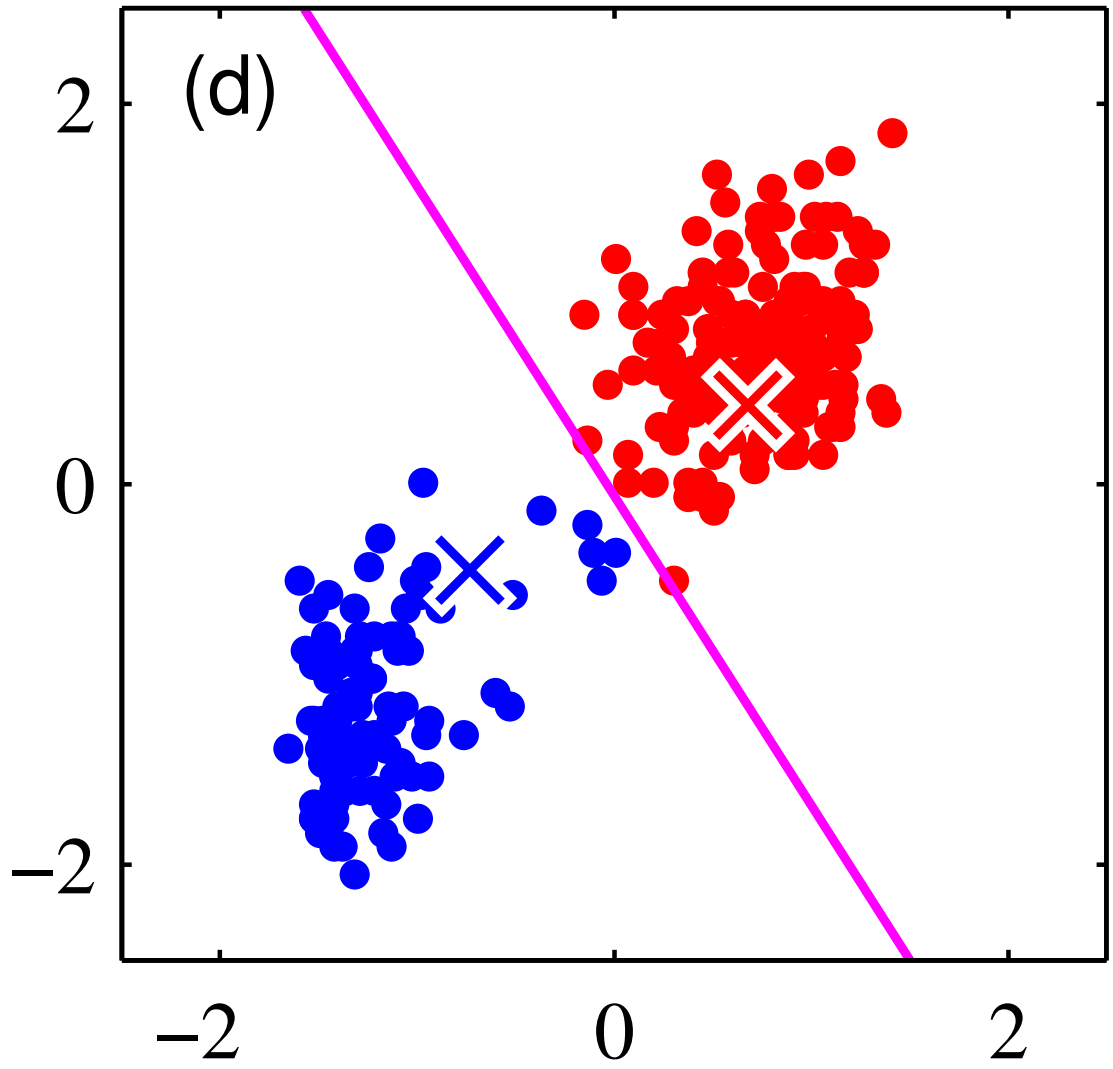
K-means clustering (K=2)



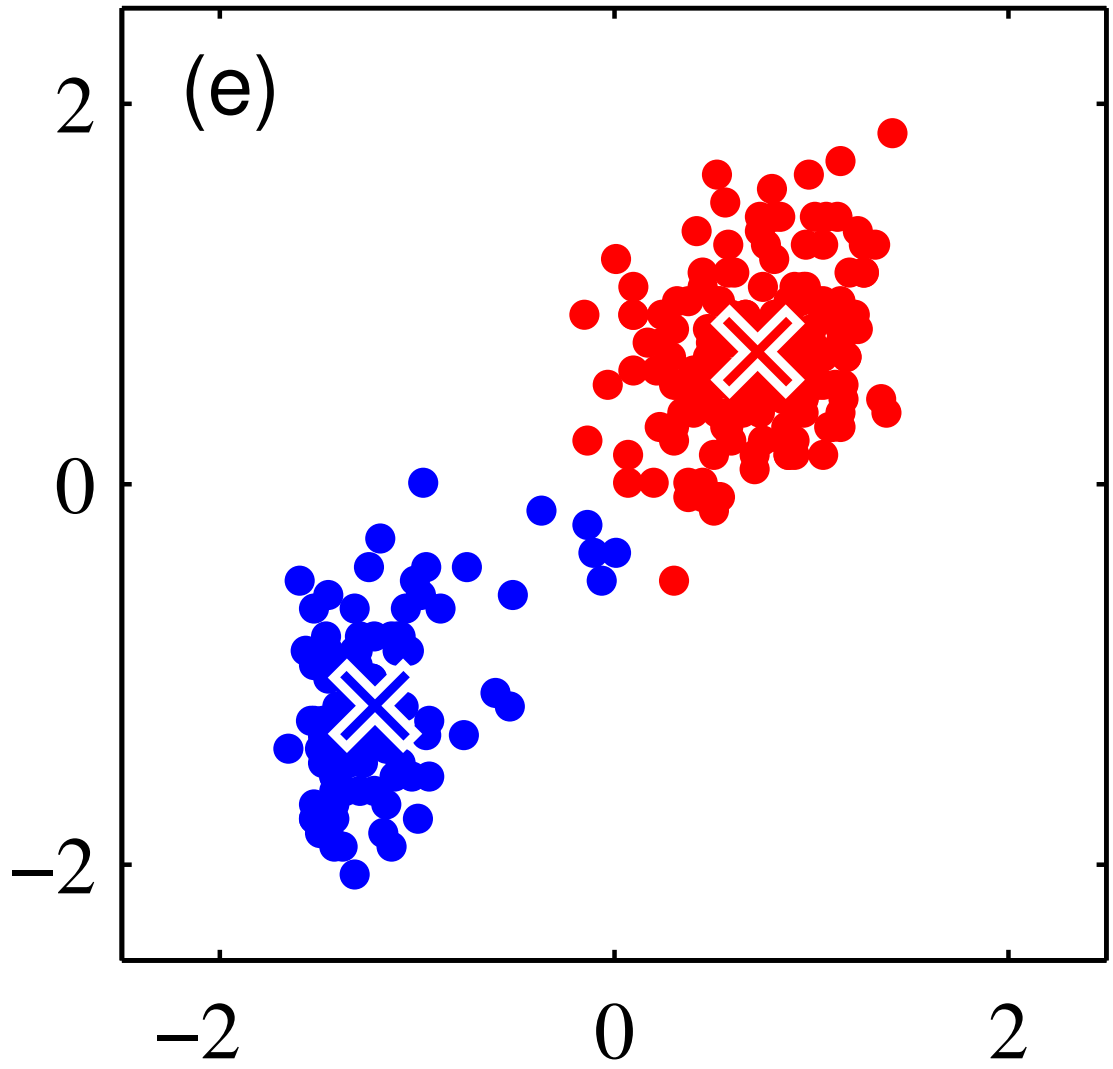
K-means clustering (K=2)



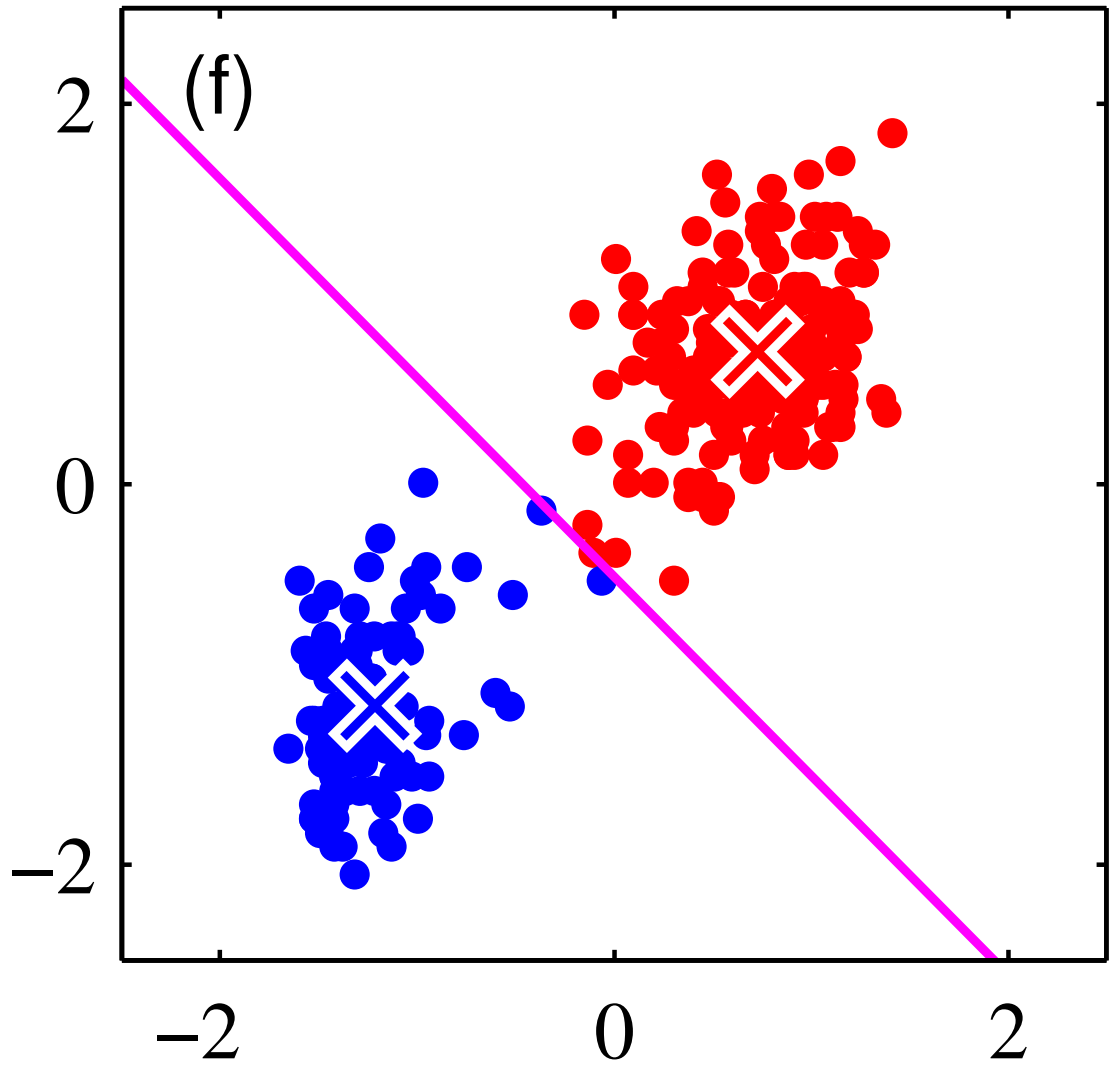
K-means clustering (K=2)



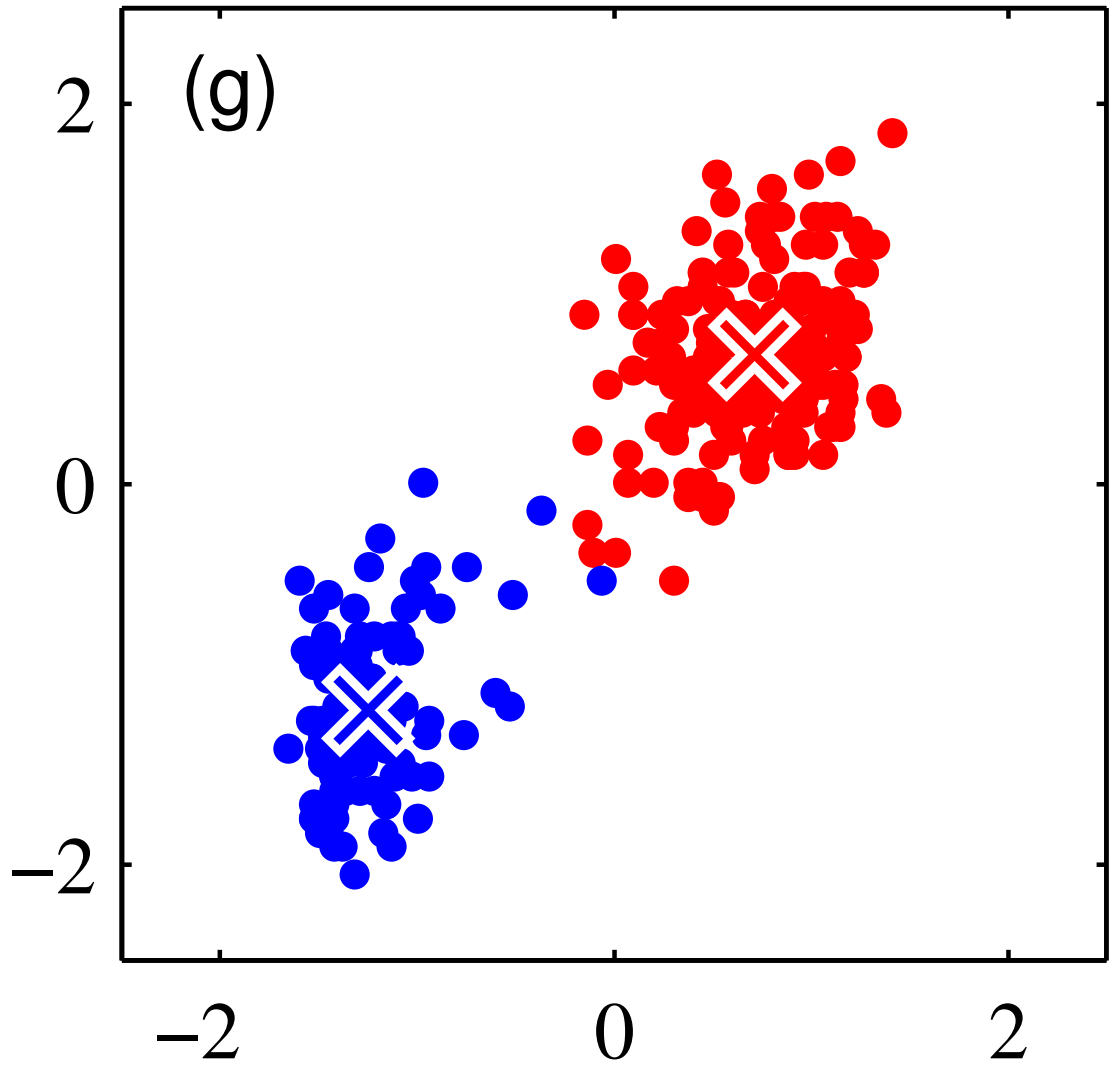
K-means clustering (K=2)



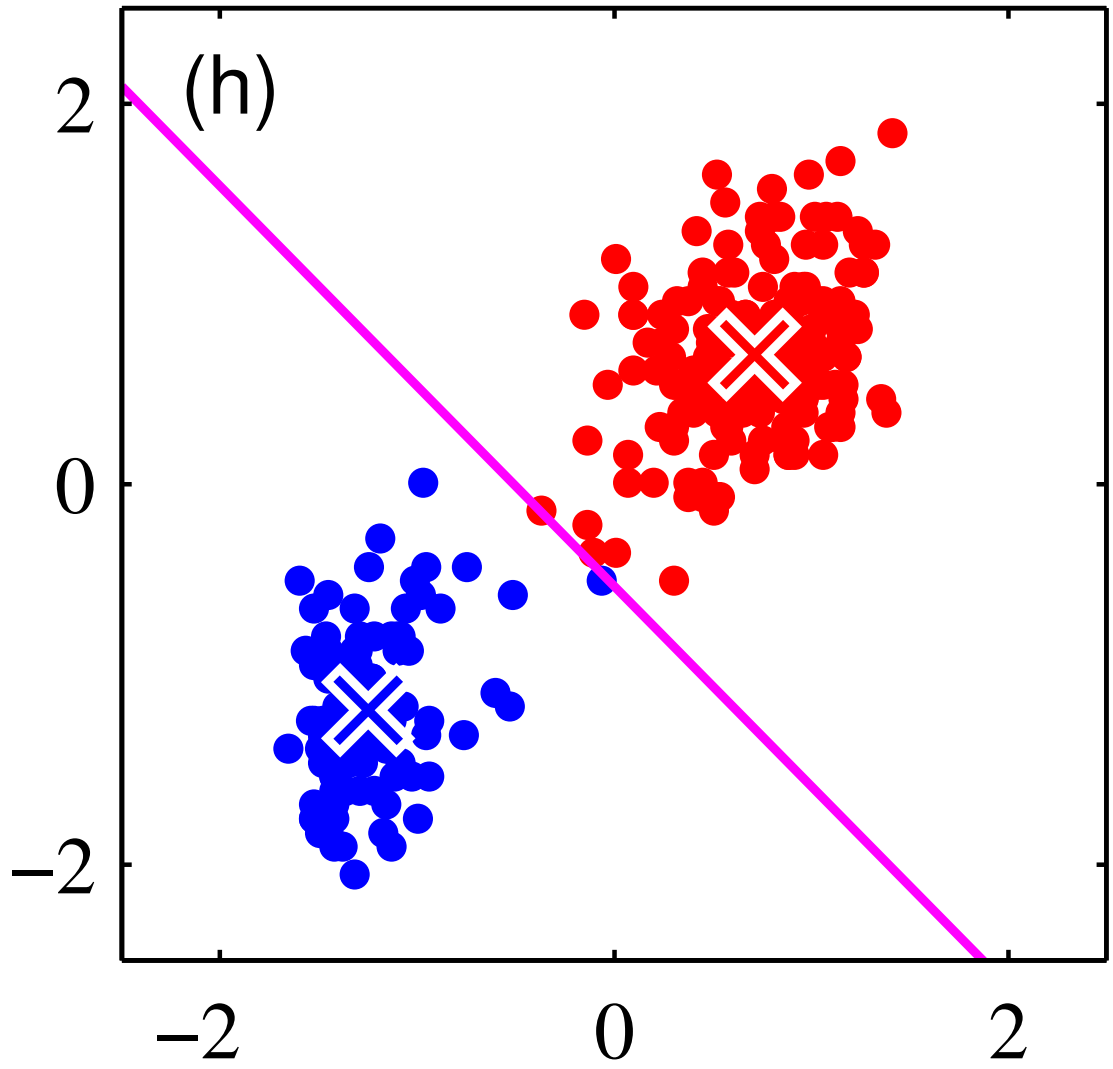
K-means clustering (K=2)



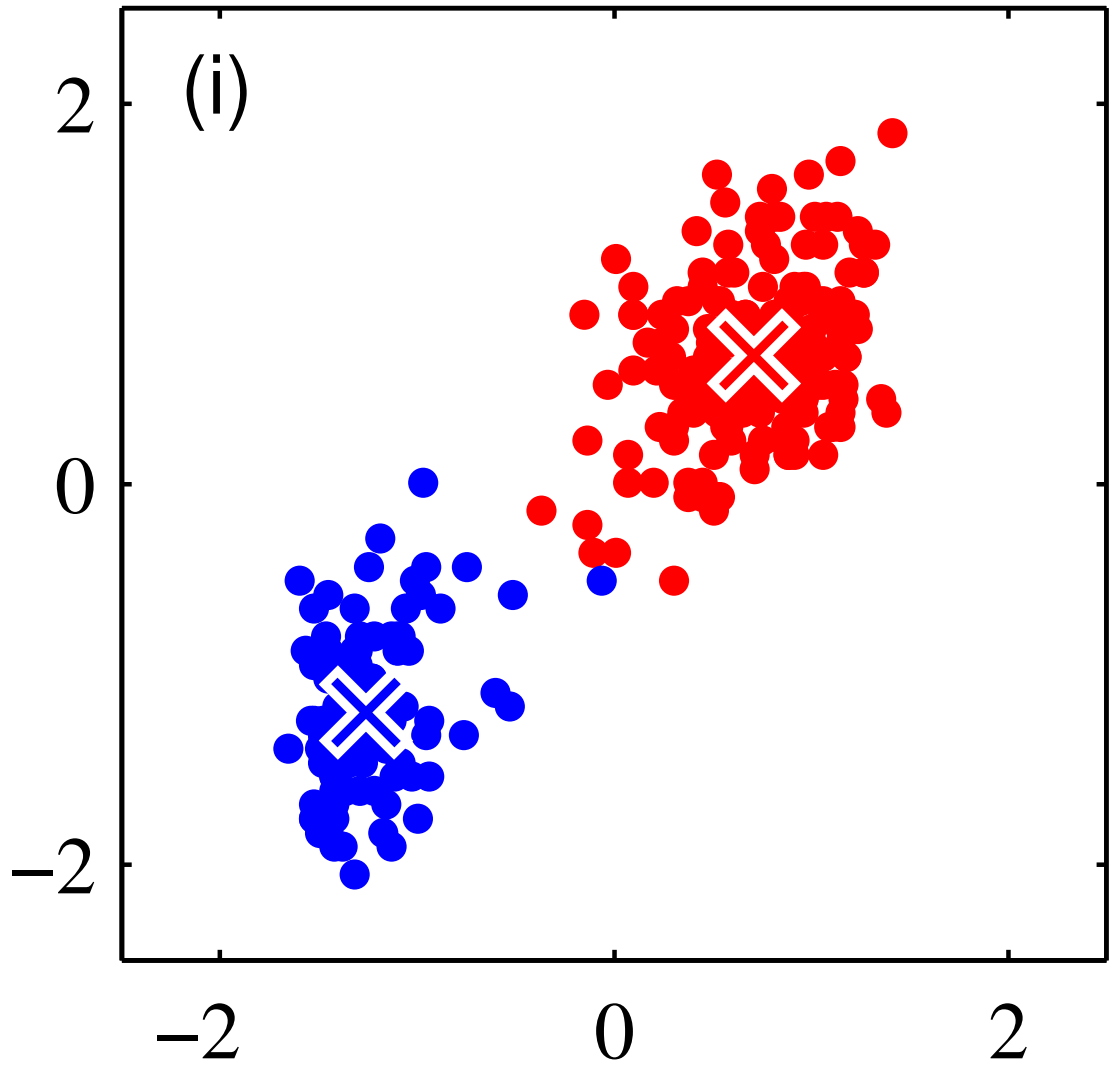
K-means clustering (K=2)



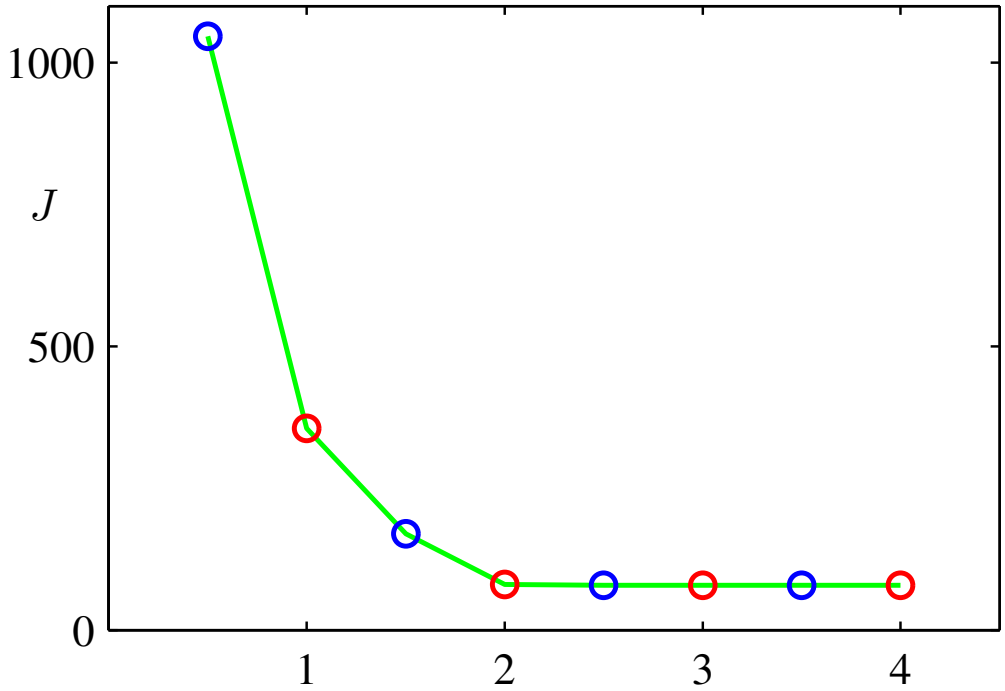
K-means clustering (K=2)



K-means clustering (K=2)



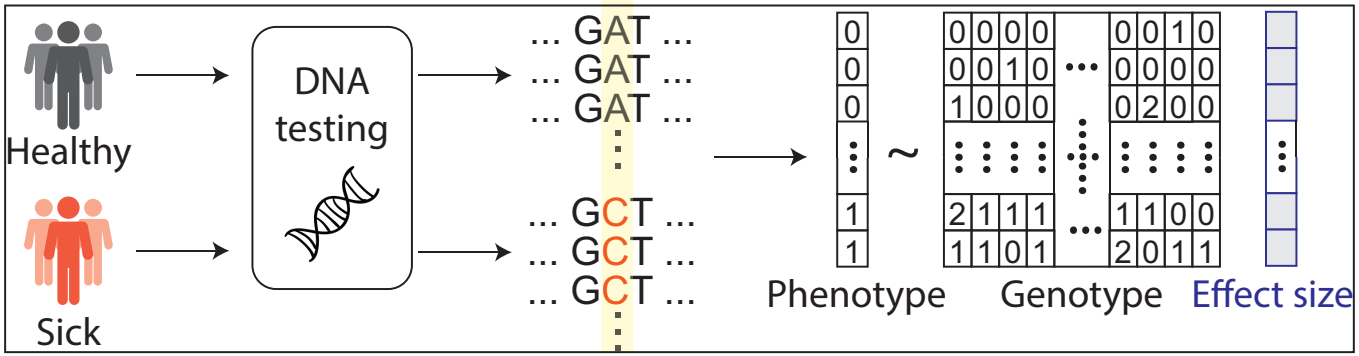
K-means clustering (K=2) convergence

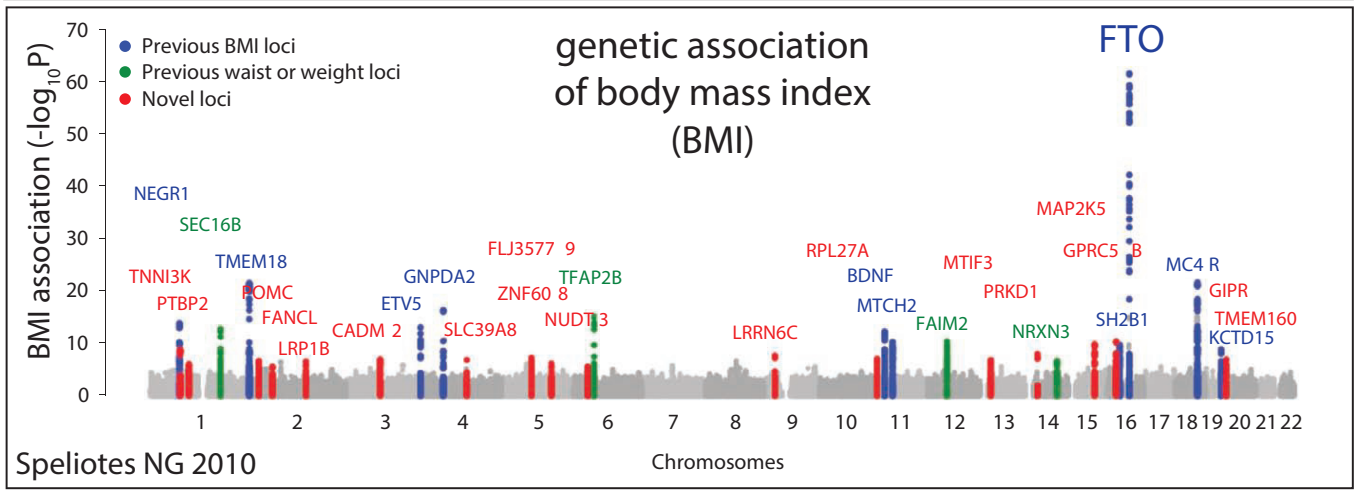


- data: $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$
- $J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} ||\mathbf{x}_n - \boldsymbol{\mu}_k||^2$

$$r_{nk} = \begin{cases} 1 & \text{if } k = \arg\min_j ||\mathbf{x}_n - \boldsymbol{\mu}_j||^2 \\ 0 & \text{if otherwise} \end{cases}$$

Clustering mutations based on epigenomic annotations



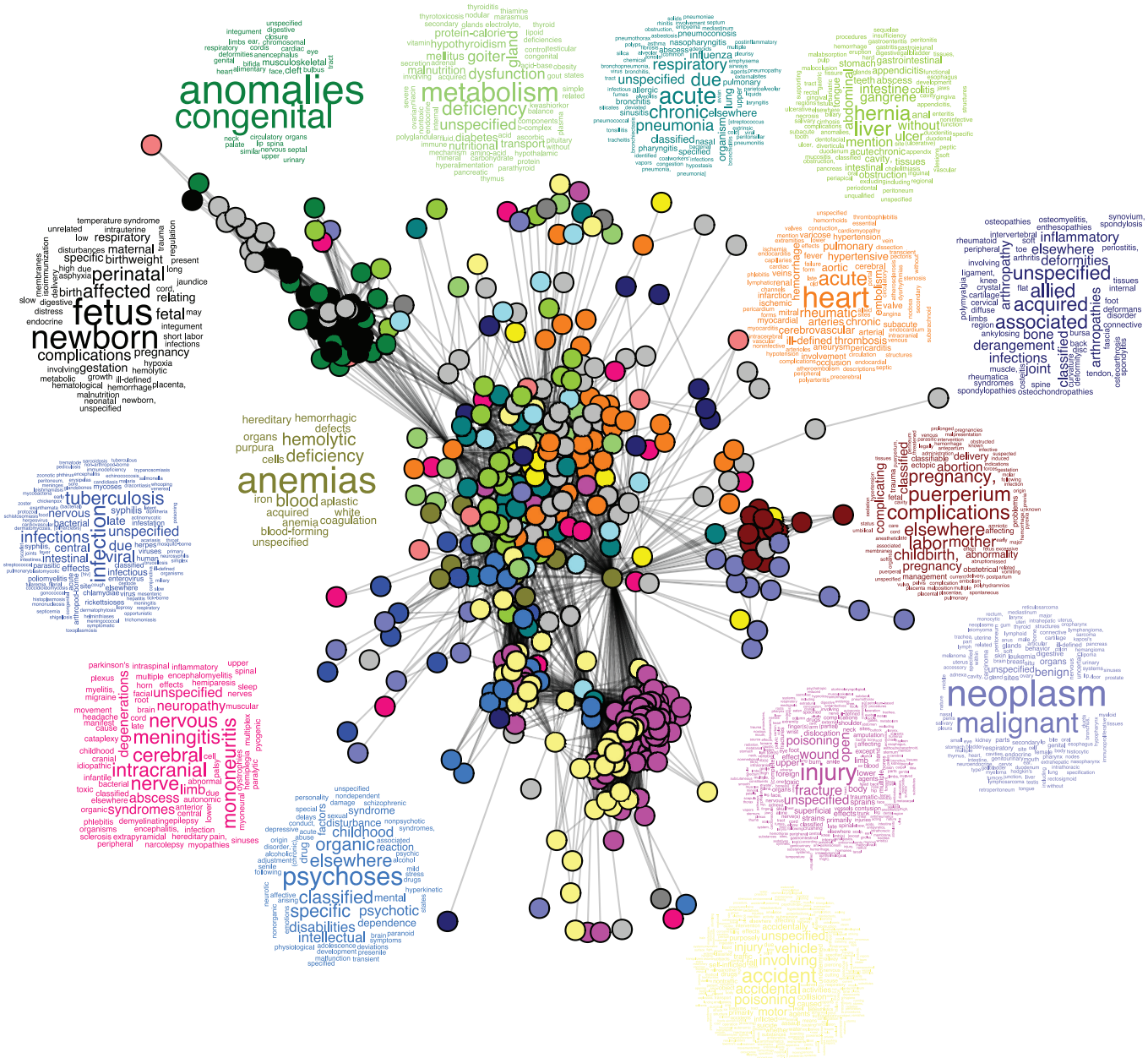


Promises of genomic medicine:

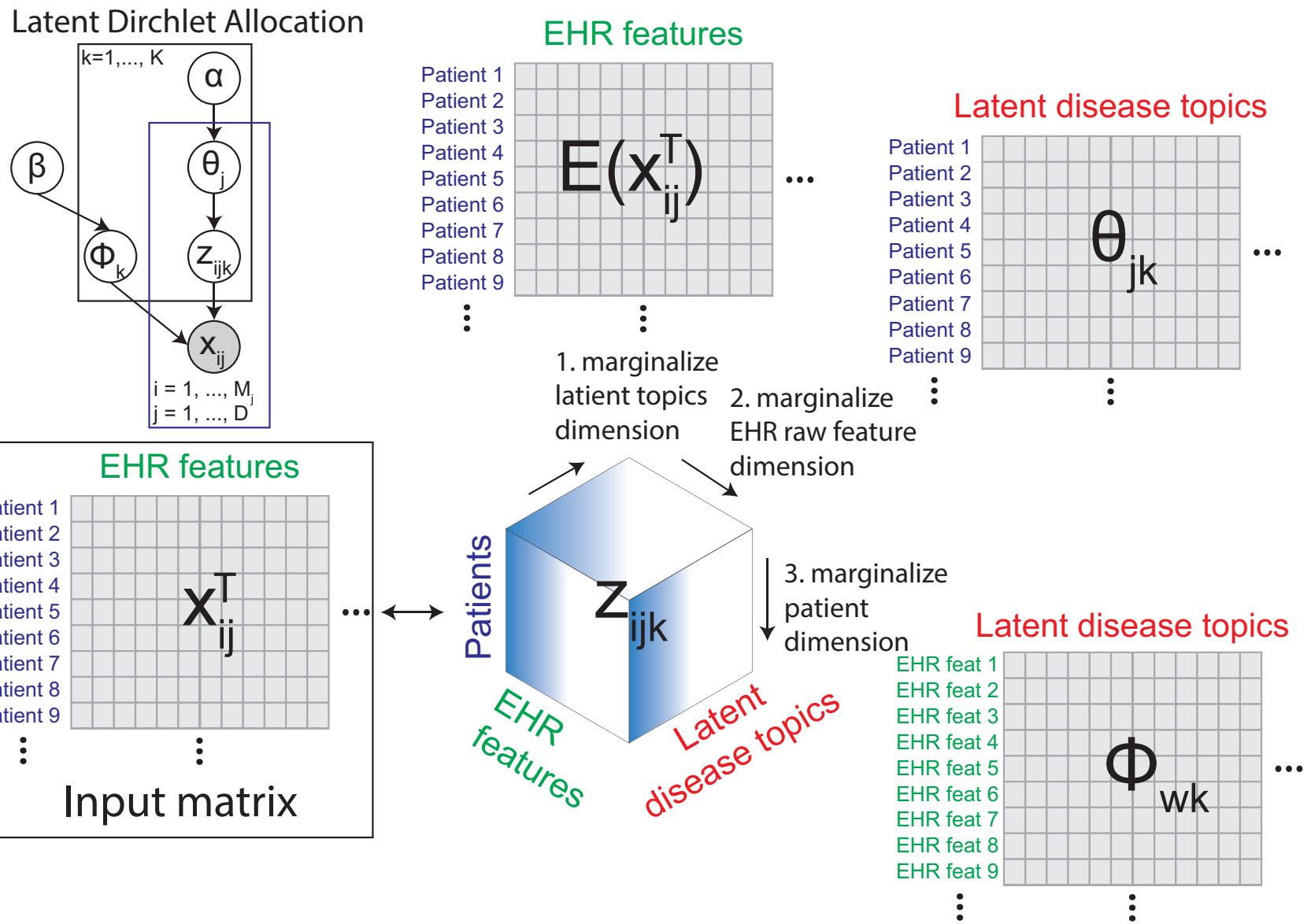
1. Disease mechanisms
2. Novel target genes
3. Novel therapeutics
4. Personalized medicine

Outline

Latent topic models (from healthcare data)



Latent Dirichlet Allocation (LDA) (Blei et al, JMLR 2003)



Grouping words by their topics (Blei et al, JMLR 2003)

"Arts"	"Budgets"	"Children"	"Education"
NEW	MILLION	CHILDREN	SCHOOL
FILM	TAX	WOMEN	STUDENTS
SHOW	PROGRAM	PEOPLE	SCHOOLS
MUSIC	BUDGET	CHILD	EDUCATION
MOVIE	BILLION	YEARS	TEACHERS
PLAY	FEDERAL	FAMILIES	HIGH
MUSICAL	YEAR	WORK	PUBLIC
BEST	SPENDING	PARENTS	TEACHER
ACTOR	NEW	SAYS	BENNETT
FIRST	STATE	FAMILY	MANIGAT
YORK	PLAN	WELFARE	NAMPHY
OPERA	MONEY	MEN	STATE
THEATER	PROGRAMS	PERCENT	PRESIDENT
ACTRESS	GOVERNMENT	CARE	ELEMENTARY
LOVE	CONGRESS	LIFE	HAITI

The William Randolph Hearst Foundation will give $1.25 million to Lincoln Center, Metropolitan Opera Co., New York Philharmonic and Juilliard School. "Our board felt that we had a real opportunity to make a mark on the future of the performing arts with these grants an act every bit as important as our traditional areas of support in health, medical research, education and the social services," Hearst Foundation President Randolph A. Hearst said Monday in announcing the grants. Lincoln Center's share will be $200,000 for its new building, which will house young artists and provide new public facilities. The Metropolitan Opera Co. and New York Philharmonic will receive $400,000 each. The Juilliard School, where music and the performing arts are taught, will get $250,000. The Hearst Foundation, a leading supporter of the Lincoln Center Consolidated Corporate Fund, will make its usual annual $100,000 donation, too.

Outline

Reinforcement learning

Reinforcement learning (RL) learn action to maximize payoff (e.g., robotics, self-driving vehicle)
Approach:

- We start with a robot that doesn't know how to walk, but moves its "muscles" randomly.
- The goal of the robot it to reach a certain destination (e.g. its "mother" at the other end of the room).
- When it reaches its mother, it gets a reward (satisfaction).
- Over time, it realizes that certain actions seem to lead to better rewards (reaching destination faster).
- It slowly learns to adjust its behavior to maximize its reward

A fun demo using RL:
https://www.youtube.com/watch?v=gn4nRCC9TwQ

Outline

Python's scikit-learn module

Over the next 3 lectures

- we're going to perform some basic machine learning
- using Python's scikit-learn module

scikit-learn API:
http://scikit-learn.org/stable/modules/classes.html

scikit-learn tutorials:
http://scikit-learn.org/stable/