# CS559 Machine Learning
## Introduction and Background

Tian Han

Department of Computer Science
Stevens Institute of Technology

Week 1

# Outline

- Course Logistics
- Machine Learning Basics
- Basic linear algebra and probability reviews

# Course Logistics

# Instructor and TA

Instructor: Tian Han

- Office Hours: 10AM - 11AM Monday via Canvas Zoom, 2:00PM - 2:45PM Friday GS 248
- Email: than6@stevens.edu

TA: Hanao Li

- Office Hours: TBD
- Email: hli136@stevens.edu

TA: Nan Cui

- Office Hours: TBD
- Email: ncui@stevens.edu

# Prerequisites

In general, you do not need to take exactly these courses as long as you know the materials very well.

- MA222 Probability theory
- Calculus and Linear Algebra
- Some optimization
- Programming skills in Python or Matlab

**Suggestion**: take **CS556**, *mathematical foundations of ML*, BEFORE you take this course. Take **CS515** for Python programming.

# Textbooks

Textbook is NOT mandatory if you could understand the lecture notes.

- R. Duda et al., *Pattern Classification*, John Wiley & Sons Inc, 2001
- C. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2006
- I. Goodfellow et al., *Deep Learning*, MIT, 2016

# Coursework

Grading:

- Homeworks (15%, 15%, 20%)
- Midterm Exam (20%)
- Final Exam (30%)

Policy:

- Submit the file through Canvas.
  - **Do Not** submit your assignments through email.
- You have a **total** of 4 late days (not including weekends) for entire class (3 homeworks), but after using 4 late days, NO credit will be given for late submission.
- You are encouraged to work and discuss in a group, but each person have to write down his/her OWN solutions. Don't cheat on the homeworks and exams!

# What you will learn in this course...

- Primary machine learning algorithms, including:
  - MLE, EM algorithm, linear models, non-parametric model, SVMs, boosting, latent variable model, neural network and more.
- Identify and use them for real data.
- Prepared for further upper-level courses (e.g., statistical machine learning, deep learning), understand research papers and involved in ML-related research.

# The course is NOT suitable for...

- Students who ONLY care about the real Python codes, but not the underlying reasoning.

# The course is NOT suitable for...

- Students who ONLY care about the real Python codes, but not the underlying reasoning.
- Students who hates mathematics, especially *Gaussian* and *Bayesian*.

# Machine Learning Basics

# What is Machine Learning

Definition by Tom Mitchell (1998):

Machine learning is the study of the algorithm that:

- improve their performance $P$
- on some task $T$
- with experience $E$

The learning task can be defined as $< P, T, E >$.

Example [Ray Mooney]:

- T: recognize hand-written digits
- P: percentage of digits being correctly classified
- E: Database of human-labeled images of hand-written digits.

# Why do we need Machine Learning

- Human expertise does not exist. (navigating on Mars)
- Humans cannot explain their expertise. (speech recognition)
- Models must be customized. (personalized medicine)
- Models are based on huge amounts of data. (genomics)

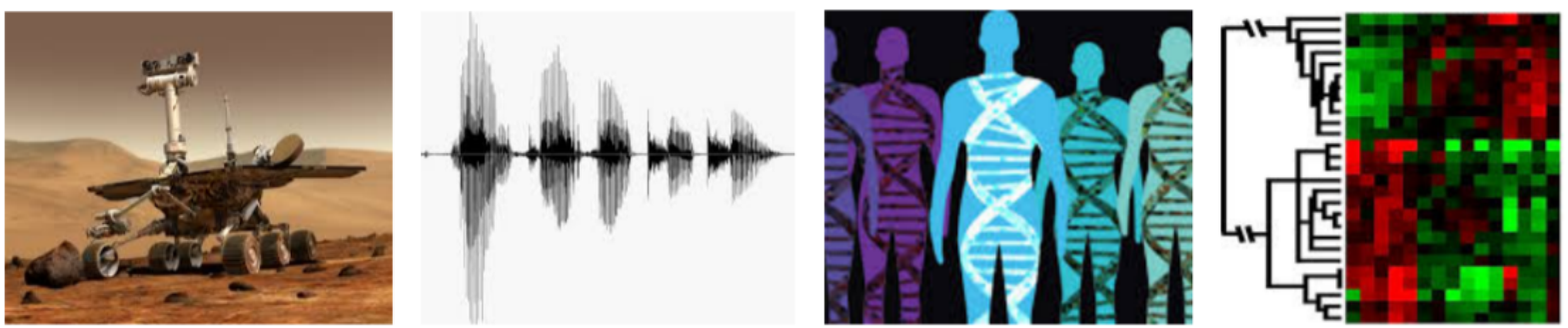

Figure: From E. Alpaydin, E. Eaton

# Some Examples

Some examples of tasks that are best solved by machine learning [G. Hinton]:

- Recognizing patterns:
  - Facial identities or facial expressions
  - Handwritten digits
  - Medical images
- Generating patterns:
  - Generating images or motion sequences
- Recognizing anomalies:
  - Unusual credit card transactions
  - Unusual patterns of sensor readings in a nuclear power plant
- Prediction:
  - Future stock prices or currency exchange rates.
- Completion:
  - Corrupted/blurred images.

# Applications: Scene Labelling

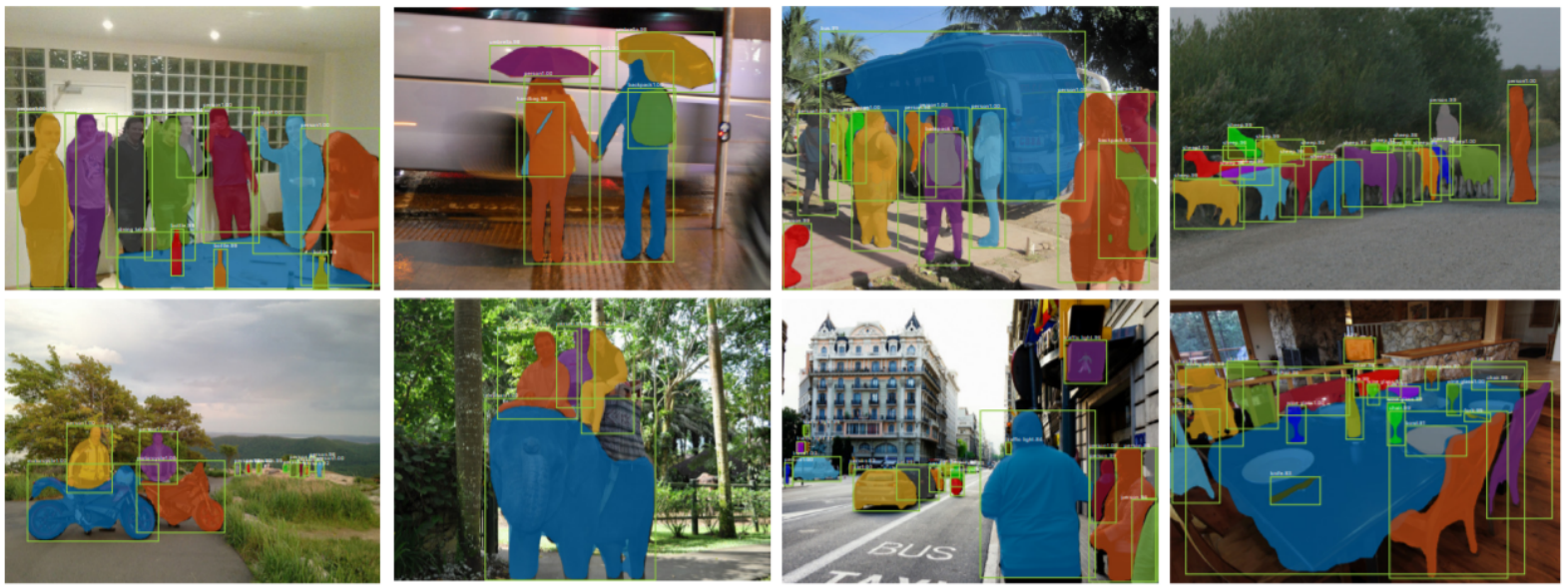


Figure: Mask R-CNN on COCO test set. [He et al., 2018]

# Applications: Image Completion

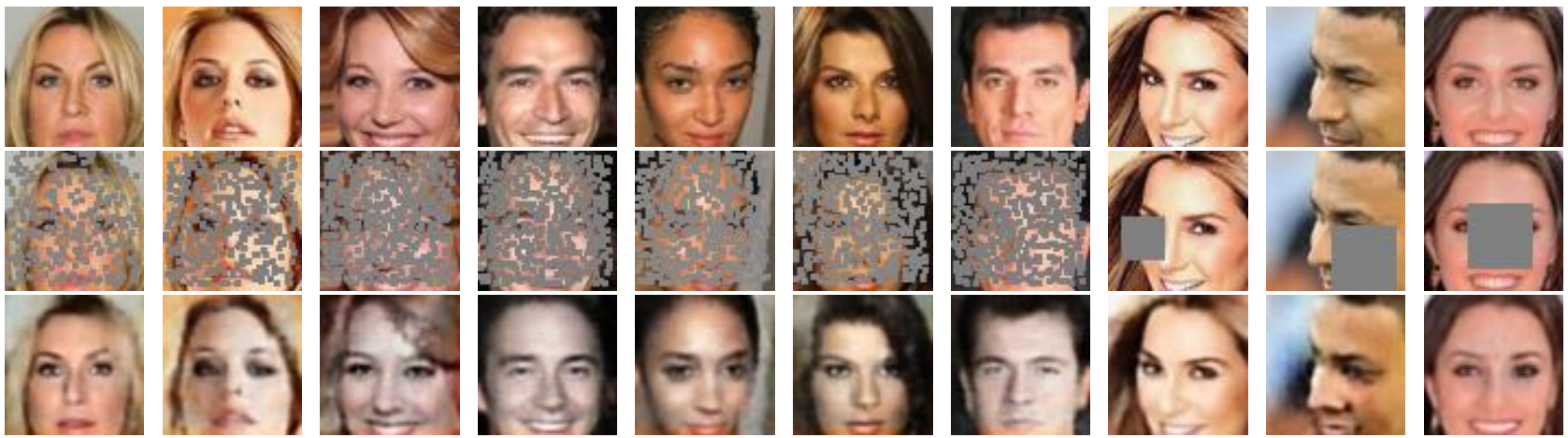

Figure: Learning from incomplete images [Han et al., 2017]. First: ground truth. Second: corrupted images for training. Third: recovered images.

# Applications: Generating Patterns

Generate Textures:

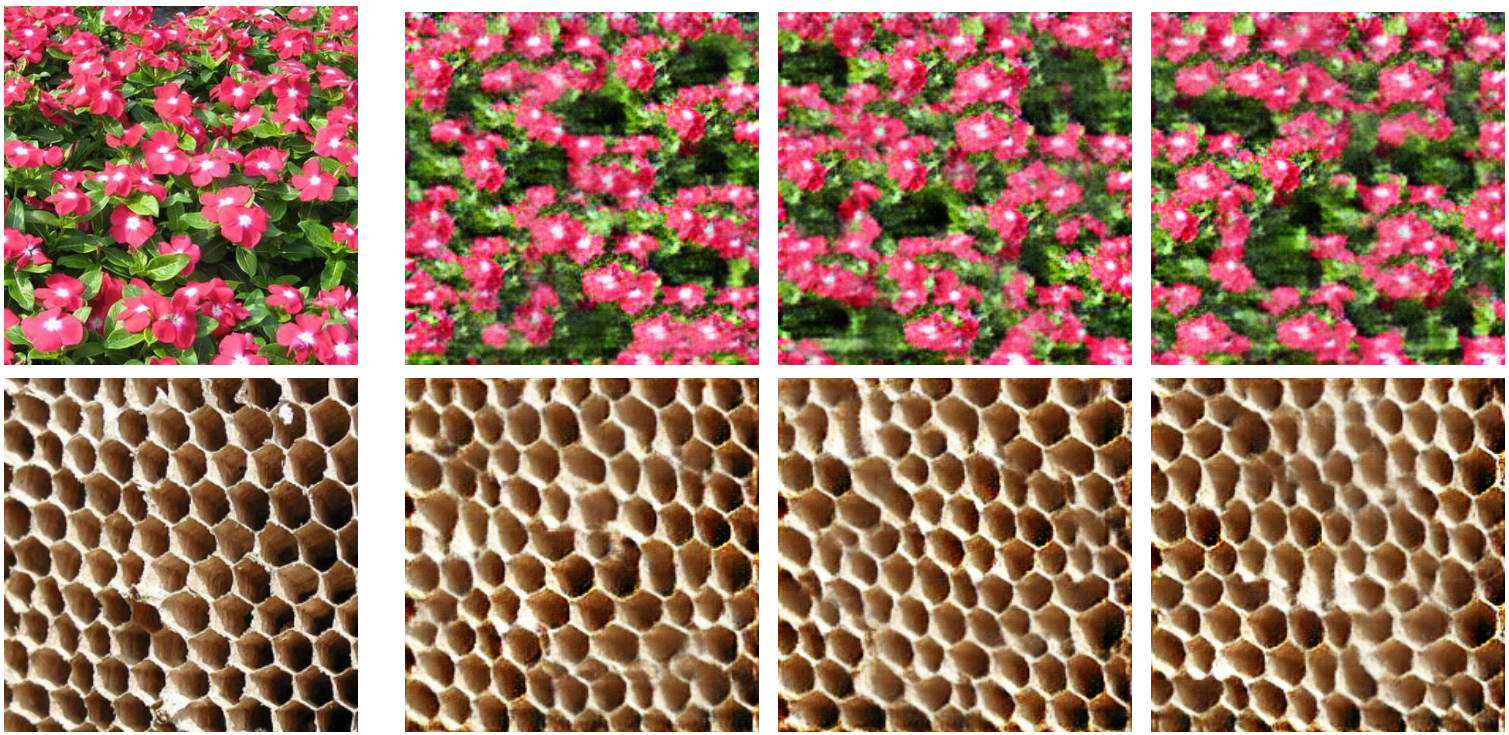

Figure: [Han et al. 2019]. Left: training texture. Right three: generated textures.

# Applications: Generating Patterns

High-resolution Face:

Figure: [Han et al., 2019].

Molecule:

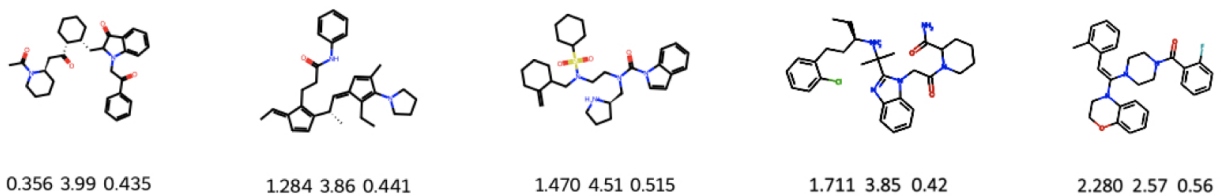


Figure: [Kong et al., 2023].

# Applications: Domain Transfer

Domain transformation

Figure: [Zhu et al. 2017]

# Applications: Domain Transfer

Video to Video synthesis

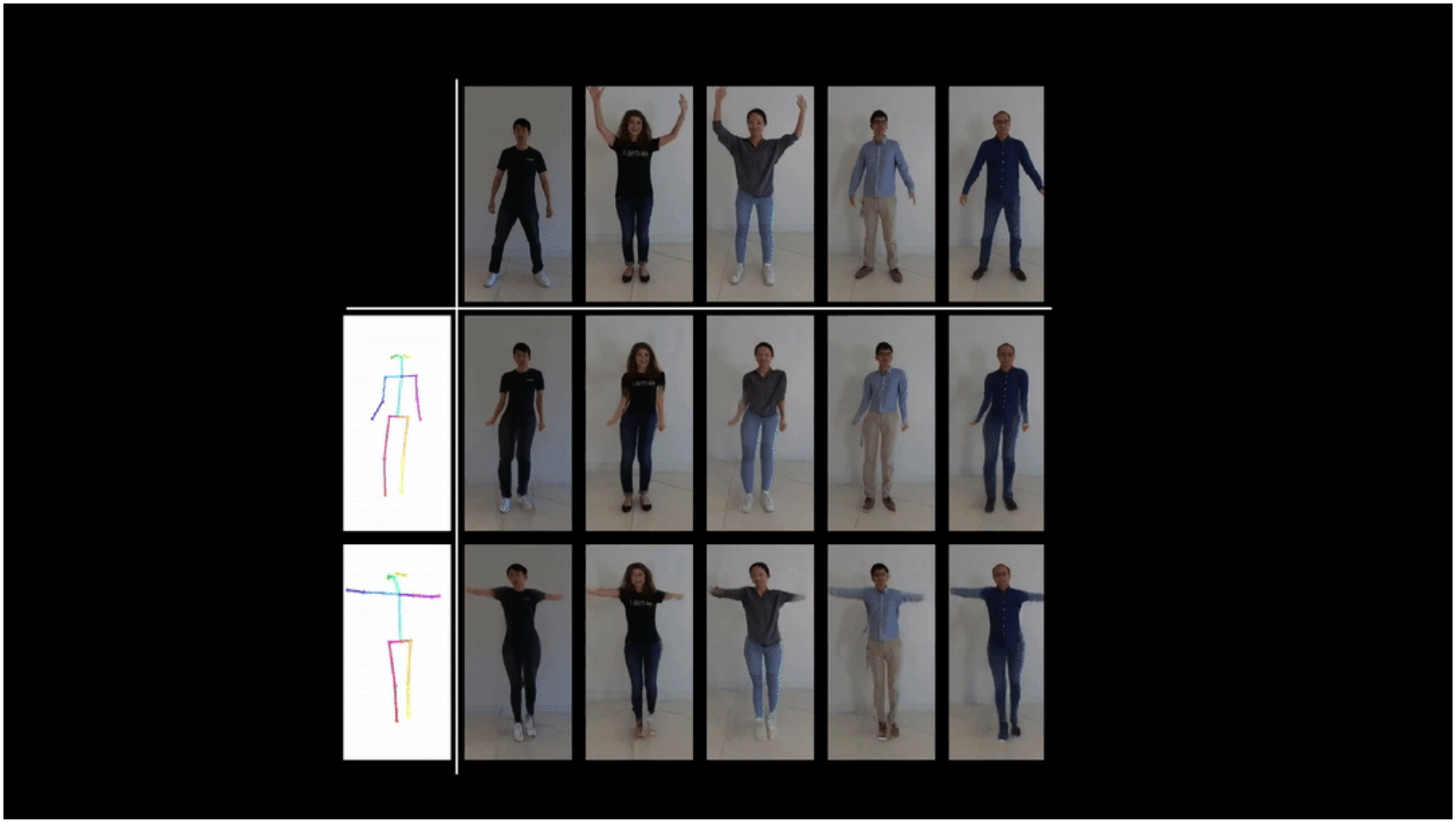

Figure: [Wang et al. 2019]

# Applications: Domain Transfer

## Video to Video synthesis

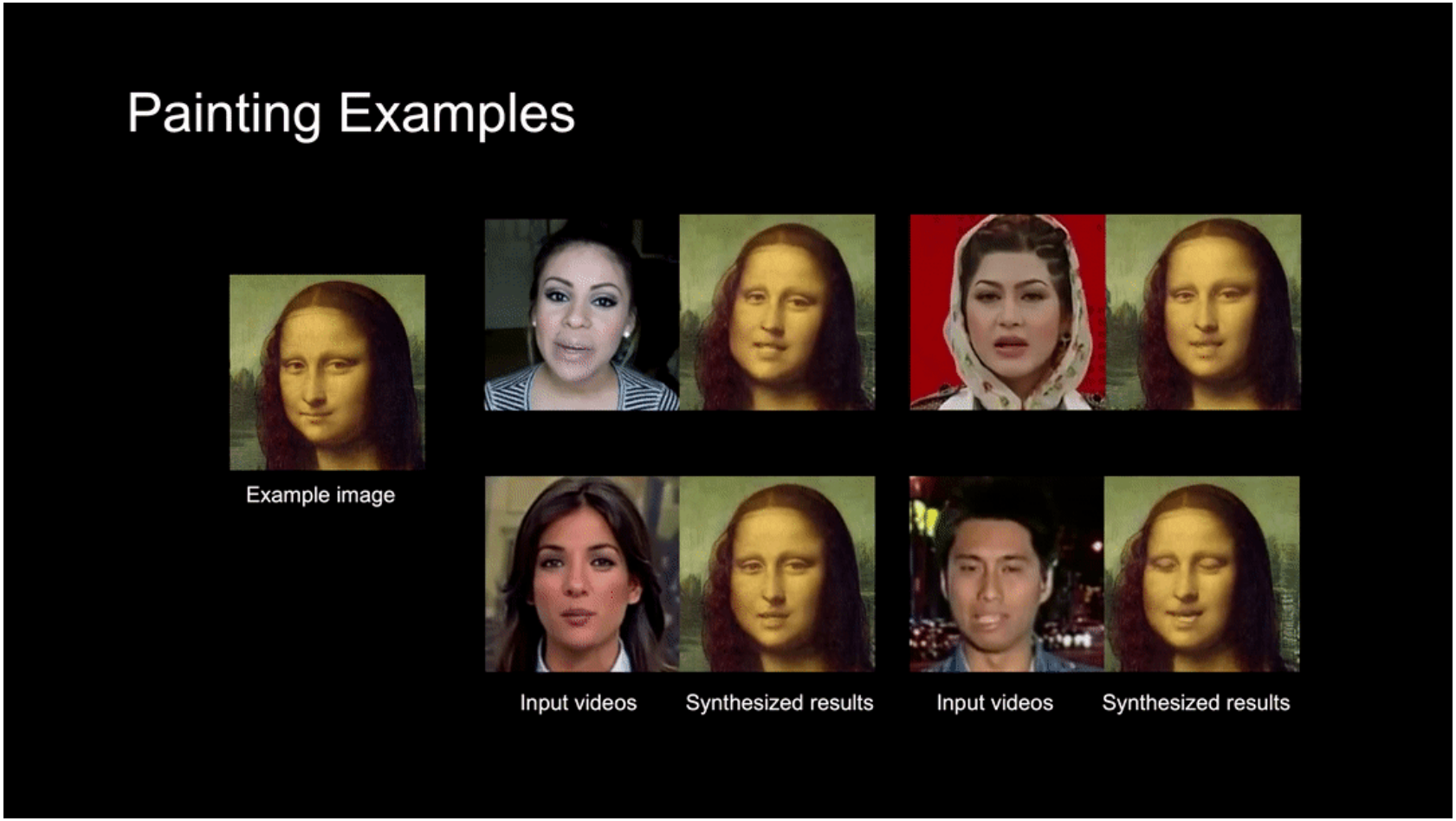


Figure: [Wang et al. 2019]

# Applications: Game

Figure: Clockwise: self-driving car [slide: E. Eaton], starcraft [CNN.com], poker [The Economic Times], Go [MIT review]

# Type of Learning

- Supervised Learning $(X, Y)$:
  - training data $X$ and their desired output (label) $Y$.
- Semi-supervised Learning $(X_1, Y_1) + (X_2)$
  - small set of training data $X_1$ with their desired label $Y_1$, with rest of unlabelled training data $X_2$
- Unsupervised Learning $(X)$
  - training data $X$, NO label $Y$ given.
- Reinforcement Learning $(X, R)$
  - training data $X$, reward $R$ for sequence of actions.

# Supervised Learning: Regression

- Given $(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(n)}, y^{(n)})$
- $y$ continuous, real value
- Learn function $f : x \rightarrow y$, predict $y$ given $x$.

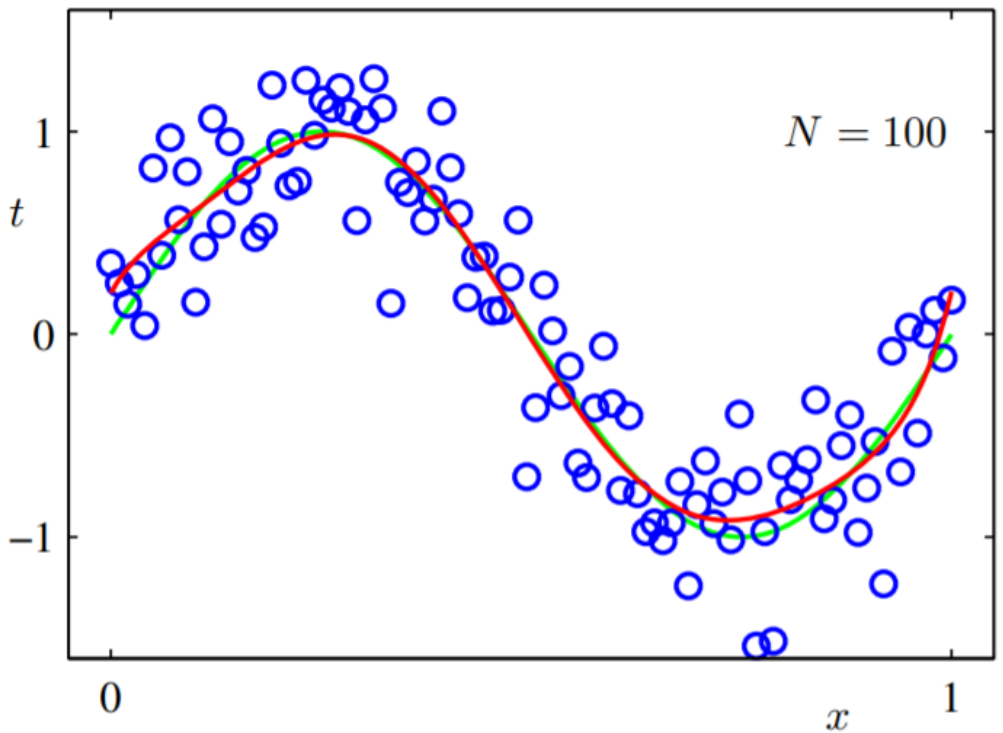


Figure: Curve fitting [C. Bishop 2006]

# Supervised Learning: Classification

- Given $(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(n)}, y^{(n)})$
- $y$ categorical
- Learn function $f : x \rightarrow y$, predict $y$ given $x$.

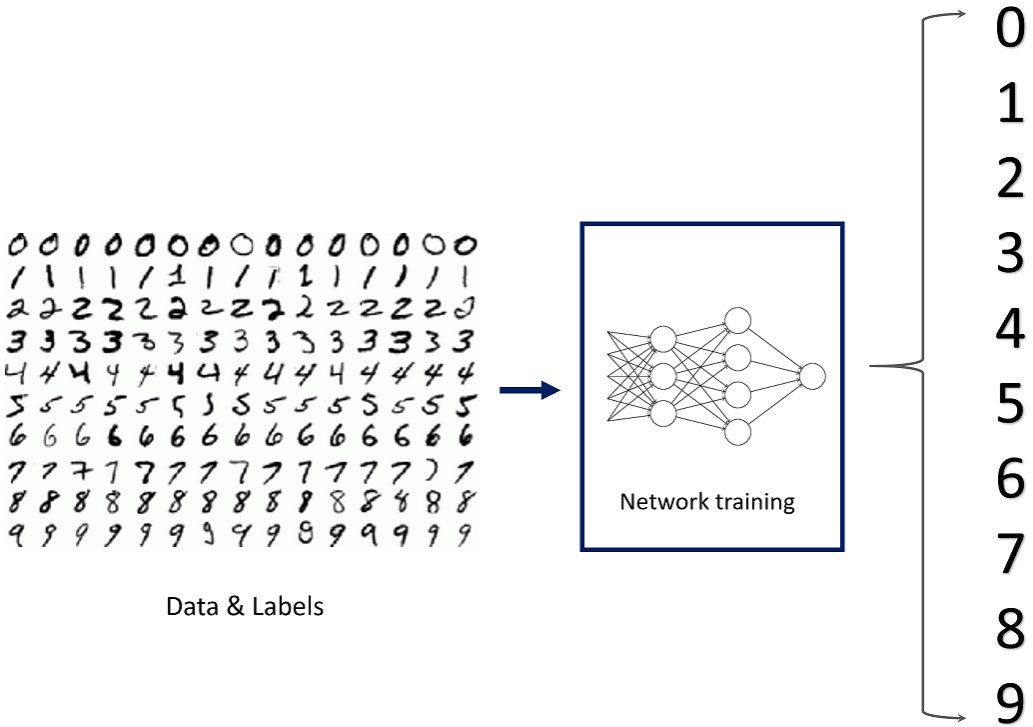


Figure: Classify the hand-written digit [figure: Orhan Gazi Yalcin]

# Unsupervised Learning

- Given only training data $x^{(1)}, x^{(2)}, \ldots, x^{(n)}$
- Learn the hidden structure behind training data.

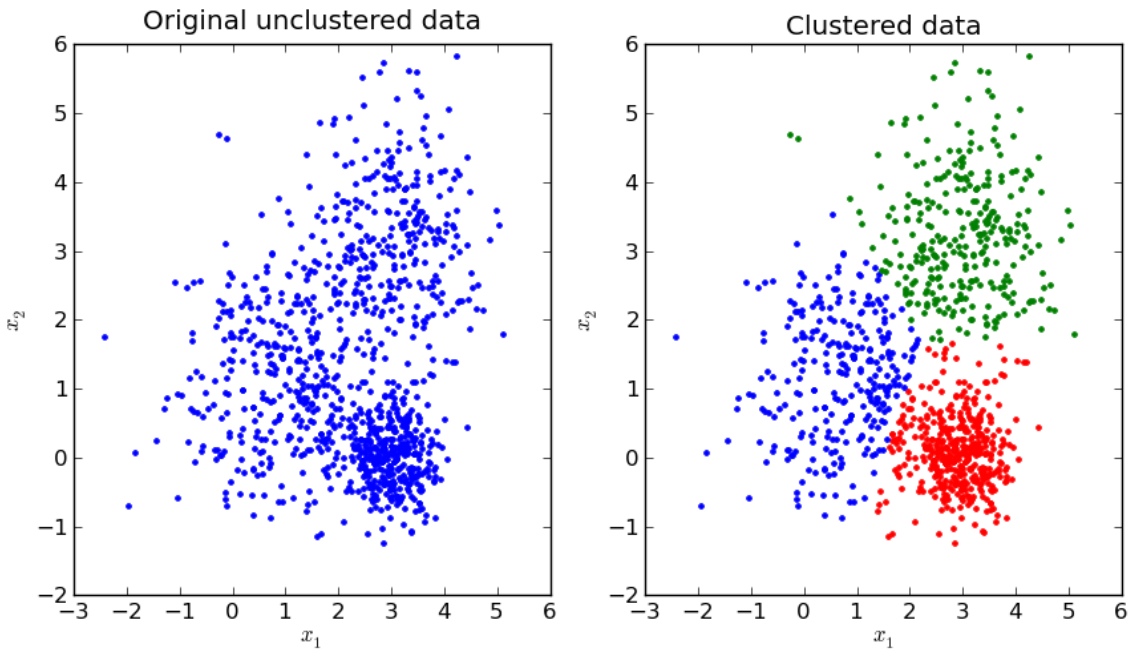


Figure: Unsupervised clustering. [figure: https://mubaris.com]

# Reinforcement Learning

- Given sequence of states $s$ and actions $a$ with their final rewards $R$, learn the optimal policy.
- policy: $f : s \rightarrow a$

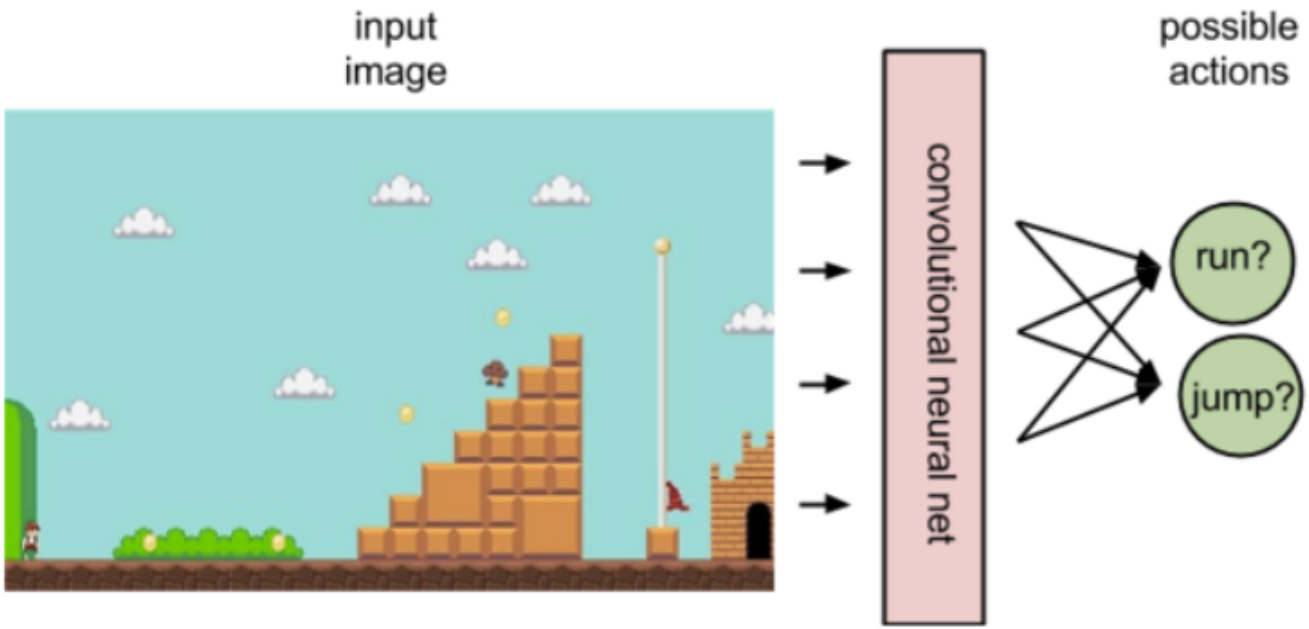


Figure: Learning the policy. [figure: skymind.ai]

# Key Components of ML

Suppose we have training data $x$ comes from the experience/environment $E$.

- Construct the representation of $x$, i.e., $h(x)$
- Build the model $f$ based on representation with parameters $\theta$ to be learned, i.e., $f(h(x); \theta)$.
- Learning the model $f$ on given task $T$ using specified performance measure $P$. (Learning from error)

# ML in Practice

Iterates the following [slides: Pedro Domingos] :

- Understand domain, prior knowledge, and goals
- Data integration, selection, cleaning, pre-processing, etc
- Learn models
- Interpret results
- Consolidate and deploy discovered knowledge

# Basic Reviews:
## Linear Algebra

# Notation

- $A \in \mathbb{R}^{m \times n}$: denote a matrix with $m$ rows and $n$ columns.
- $x \in \mathbb{R}^n$: denote a vector with $n$ entries, $x = [x_1, x_2, ..., x_n]^T$
- $a_{ij}$: denote the entry of $A$ in the $i$-th row and $j$-th column.

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix}$$

- $a_j$: $j$-th column of $A$. $a_i^T$: $i$-th row of $A$.

$$A = \begin{bmatrix} | & | & & | \\ a_1 & a_2 & \dots & a_n \\ | & | & & | \end{bmatrix} \quad A = \begin{bmatrix} — & a_1^T & — \\ & \vdots & \\ — & a_m^T & — \end{bmatrix}$$

# Matrix Multiplication

Vector-Vector product:

- inner product (dot product): $x \in \mathbb{R}^n$, $y \in \mathbb{R}^n$

$$< x, y >= x^T y \in \mathbb{R} = [x_1, x_2, \ldots, x_n] \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \sum_{i=1}^{n} x_i y_i$$

- outer product: $x \in \mathbb{R}^m$, $y \in \mathbb{R}^n$

$$xy^T \in \mathbb{R}^{m \times n} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix} [y_1, y_2, \ldots, y_n] = \begin{bmatrix} x_1 y_1 & \ldots & x_1 y_n \\ x_2 y_1 & \ldots & x_2 y_n \\ \vdots & \ddots & \vdots \\ x_m y_1 & \ldots & x_m y_n \end{bmatrix}$$

# Matrix Multiplication

Matrix-Vector product: $A \in \mathbb{R}^{m \times n}$, $x \in \mathbb{R}^n$, $y = Ax \in \mathbb{R}^m$

- Loading Matrix view:

$$y = Ax = \begin{bmatrix} — & a_1^T & — \\ & \vdots & \\ — & a_m^T & — \end{bmatrix} x = \begin{bmatrix} a_1^T x \\ \vdots \\ a_m^T x \end{bmatrix}$$

  i.e., $y_i = a_i^T x$.

- Basis vector view:

$$y = Ax = \begin{bmatrix} | & & | \\ a_1 & \dots & a_n \\ | & & | \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} | \\ a_1 \\ | \end{bmatrix} x_1 + \dots + \begin{bmatrix} | \\ a_n \\ | \end{bmatrix} x_n$$

  $y$ is the *linear combination* of the column vectors of $A$, $a_i$ is the basis vector.

# Matrix Multiplication

Matrix-Matrix product: $A \in \mathbb{R}^{m \times n}$, $B \in \mathbb{R}^{n \times p}$, $C = AB \in \mathbb{R}^{m \times p}$

$$C = AB = \begin{bmatrix} — & a_1^T & — \\ & \vdots & \\ — & a_m^T & — \end{bmatrix} \begin{bmatrix} | & & | \\ b_1 & \dots & b_p \\ | & & | \end{bmatrix}$$

$$= \begin{bmatrix} a_1^T b_1 & \dots & a_1^T b_p \\ a_2^T b_1 & \dots & a_2^T b_p \\ \vdots & \ddots & \vdots \\ a_m^T b_1 & \dots & a_m^T b_p \end{bmatrix}$$

- $a_i \in \mathbb{R}^n$, $b_j \in \mathbb{R}^n$, so $a_i^T b_j$ makes sense.
- Matrix Multiplication is *not* commutative, i.e., $AB \neq BA$.

# Operations

- Transpose: flipping rows and columns, i.e., $(A^T)_{ij} = A_{ji}$
  - $(A^T)^T = A$
  - $(AB)^T = B^T A^T$
  - $(A + B)^T = A^T + B^T$
- Identity Matrix $\mathbf{I}$:

$$\mathbf{I} = \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{bmatrix}$$

for every $A \in \mathbb{R}^{n \times n}$, $A\mathbf{I} = A = \mathbf{I}A$

# Operations

- Inverse of the square matrix $A \in \mathbb{R}^{n \times n}$, denote as $A^{-1}$:

$$AA^{-1} = \mathbf{I} = A^{-1}A$$

  - Not all matrices have inverses. Non-square matrix may need pseudo-inverse.
  - $(A^{-1})^{-1} = A$
  - $(AB)^{-1} = B^{-1}A^{-1}$
  - $(A^{-1})^T = (A^T)^{-1}$
- Orthogonal matrix:
  - $x \in \mathbb{R}^n$, $y \in \mathbb{R}^n$, $< x, y >= x^T y = 0$
  - square $U \in \mathbb{R}^{n \times n}$, $U^T U = \mathbf{I} = UU^T$.
  - $||Ux|| = ||x||$

# Operations

- Trace tr(.): sum of diagonal elements of square $A \in \mathbb{R}^{n \times n}$:

$$trA = \sum_{i=1}^{n} A_{ii}$$

  - $tr(A) = tr(A^T)$
  - $tr(A + B) = tr(A) + tr(B)$
  - if $AB$ square, then $tr(AB) = tr(BA)$

- Norms: measures the length of the vector $x \in \mathbb{R}^n$
  - Euclidean or $l_2$ norm: $||x||_2 = \sqrt{\sum_{i=1}^{n} x_i^2} = \sqrt{x^T x}$
  - $l_1$ norm: $||x||_1 = \sum_{i=1}^{n} |x_i|$
  - Frobenius Norm: for matrix $A \in \mathbb{R}^{m \times n}$:

$$||A||_F = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} A_{ij}^2} = \sqrt{tr(A^T A)}$$

# Eigendecomposition

For given square matrix $A \in \mathbb{R}^{n \times n}$, $\lambda$ is the *eigenvalue* of $A$, and $x$ is the corresponding *eigenvector*, if:

$$Ax = \lambda x, x \neq 0$$

Note that for constant $c$, if $x$ is eigenvector, then $cx$ is also the eigenvector of $A$. Write all equations together, we have:

$$AX = X\Lambda$$

where

$$X \in \mathbb{R}^{n \times n} = \begin{bmatrix} | & & | \\ x_1 & \dots & x_n \\ | & & | \end{bmatrix}, \Lambda = \begin{bmatrix} \lambda_1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \lambda_n \end{bmatrix}$$

If $x_1, x_2, \dots, x_n$ are linearly independent, then $X$ is invertible, then we have eigendecomposition: $A = X\Lambda X^{-1}$

# SVD

**Singular Value Decomposition** provides another way to factorize a matrix into *singular vectors* and *singular values*:

$$A = UDV^T$$

- $A \in \mathbb{R}^{m \times n}$, $U \in \mathbb{R}^{m \times m}$, $D \in \mathbb{R}^{m \times n}$, $V \in \mathbb{R}^{n \times n}$.
- $U$, $V$ orthogonal matrices, $D$ diagonal matrix.
- column of $U$: left-singular vectors.
  column of $V$: right-singular vectors.
  elements along the diagonal of $D$: singular value.
- More general than eigendecomposition. Every real matrix has SVD, but not the eigendecomposition.

# Matrix Calculus

Suppose $y \in \mathbb{R}^m$, $x \in \mathbb{R}^n$, $y = h(x)$, then

$$\frac{\partial y}{\partial x} = \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \frac{\partial y_1}{\partial x_2} & \cdots & \frac{\partial y_1}{\partial x_n} \\ \frac{\partial y_2}{\partial x_1} & \frac{\partial y_2}{\partial x_2} & \cdots & \frac{\partial y_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial y_m}{\partial x_1} & \frac{\partial y_m}{\partial x_2} & \cdots & \frac{\partial y_m}{\partial x_n} \end{bmatrix}$$

- if $y = Ax$, where $A \in \mathbb{R}^{m \times n}$, then $\frac{\partial y}{\partial x} = A$
- if $y = x^T A x$, where $A \in \mathbb{R}^{n \times n}$ and $y$ is scalar (i.e., $m = 1$), then $\frac{\partial y}{\partial x} = x^T (A + A^T)$. If $A$ is symmetric, then $\frac{\partial y}{\partial x} = 2x^T A$

# Basic Reviews:

## Probability and Distribution

Univariate Distribution
Joint Distribution
Conditional Distribution

# Probability is important!

- *"The true logic of this world is in the calculus of probabilities"*
  —— Maxwell
- Machine learning is about *learning* from training examples $x_{train}$, and *generalizing* to the testing examples $x_{test}$.
- Both $x_{train}$ and $x_{test}$ can be considered random samples from a population or probability distribution $P_{data}$.
- Learning is about estimating properties of the probability distribution based on a finite number of $x_{train}$.

# Basic Language and Notation

- Experiment: the phenomenon under study, generates **random** outcomes $\omega$.
- Sample space $\Omega$: The set of all possible outcomes.
- $X(\omega)$: numerical descriptions of the outcome.
  - E.g., randomly sample a person from a population.
  - The probability of getting a person taller than 6 feet.
  - $\omega$ is a random person, $X(\omega)$ is the height of the person.
  - $X$: random variable.
- Event: a subset of the sample space. E.g., Let $A$ be the event that the person is taller than 6 feet, then $A = \{\omega : X(\omega) > 6\}$. $\rightarrow$ connects event to random variable.
- Probability: defined on event. E.g., $P(A) = P(\{\omega : X(\omega) > 6\}) = P(X > 6)$.

# Axioms

- For any event $A$, $P(A) \geq 0$.
- For the sample space $\Omega$, $P(\Omega) = 1$.
- If $A \cap B = \phi$, then $P(A \cup B) = P(A) + P(B)$.

# Univariate Distribution

Discrete Random Variable:

- Example: rolling a fair die, the prob. of getting any number.
- $X$ has a uniform distribution over $\{1, 2, ..., 6\}$.
  $P(X = x) = 1/6$ for $x \in \{1, 2, ..., 6\}$.
- Probability Mass Function: $p(x) = P(X = x)$.
- Expectation: $\mathbb{E}(X) = \sum_x xp(x)$.

Continuous Random Variable:

- Example: the height of a random person.
- $X \in (x, x + \Delta x)$ as the basic event.
- Probability Density Function $f(x)$:
  $P(X \in (x, x + \Delta x)) = f(x)\Delta x$
- Expectation: $\mathbb{E}(X) = \int xf(x)dx$
- Cumulative Density Function: $F(x) = P(X \leq x)$,
  $F'(x) = f(x)$.

# Expectation and Variance

- Expectation (average, mean) $\mathbb{E}(X) = \sum_x xp(x)$
- Expectation for function $h(.)$: $\mathbb{E}[h(X)] = \sum_x h(x)p(x)$.
- $\mathbb{E}(aX + b) = a\mathbb{E}(X) + b$.
- Variance: averaged squared deviation from the center.

$$\begin{aligned} \mathrm{Var}(X) &= \mathbb{E}[X - \mathbb{E}(X)]^2 \\ &= \mathbb{E}(X^2) - \mathbb{E}(X)^2. \end{aligned}$$

- $\mathrm{Var}(aX + b) = a^2\mathrm{Var}(X)$

# Transformation

Suppose $X \sim f_X(x)$, and let $Y = h(X)$. In order to derive the density of $Y$, i.e., $f_Y(y)$. One could use the cumulative density.

$$\begin{aligned} F_Y(y) &= P(Y \leq y) \\ &= P(h(X) \leq y) \\ &= P(X \in \{x : h(x) \leq y\}). \end{aligned}$$

Then we calculate $f_Y(y) = F'_Y(y)$.
For a monotone increasing $h(x)$, let $x = g(y)$ be the inverse of $h$, then

$$\{x : h(x) \leq y\} = \{x : x \leq g(y)\}.$$

So $F_Y(y) = F_X(g(y))$, and $f_Y(y) = f_X(g(y))g'(y)$.

In general, for a monotone $h(x)$, $f_Y(y) = f_X(g(y))|g'(y)|$.

# Some common distributions

Bernoulli distribution: $Z \sim \text{Bernoulli}(p)$, $Z \in \{0, 1\}$, $P(Z = 1) = p$ and $P(Z = 0) = 1 - p$.

Gaussian (Normal) distribution:

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

# Joint Distribution

Discrete $X$ and $Y$:

- Joint probability mass function (joint distribution): $p(x, y) = P(X = x, Y = y)$.
- Marginal distribution: $p_X(x) = \sum_y p(x, y)$, $p_Y(y) = \sum_x p(x, y)$.

Continuous $X$ and $Y$:

- Joint basic event: $\{X \in (x, x + \Delta x), Y \in (y, y + \Delta y)\}$
- Joint distribution $f(x, y)$: $P(X \in (x, x + \Delta x), Y \in (y, y + \Delta y)) = f(x, y)\Delta x \Delta y$.
- Marginal Distribution: $f_X(x) = \int f(x, y)dy$, $f_Y(y) = \int f(x, y)dx$

# Expectation, Variance, Covariance

Expectation:

- If $(X, Y) \sim p(x, y)$, then $\mathbb{E}(h(X, Y)) = \sum_x \sum_y h(x, y)p(x, y)$.
- $\mathbb{E}(X + Y) = \mathbb{E}(X) + \mathbb{E}(Y)$

Covariance: let $\mu_X = \mathbb{E}(X)$, $\mu_Y = \mathbb{E}(Y)$

- $\mathrm{Cov}(X, Y) = \mathbb{E}[(X - \mu_X)(Y - \mu_Y)]$
- $\mathrm{Cov}(X, Y) = \mathbb{E}(XY) - \mathbb{E}(X)\mathbb{E}(Y)$

Variance:

- $\mathrm{Var}(h(X, Y)) = \mathbb{E}[(h(X, Y) - \mathbb{E}(h(X, Y)))^2]$
- $\mathrm{Var}(X + Y) = \mathrm{Var}(X) + \mathrm{Var}(Y) + 2\mathrm{Cov}(X, Y)$
- $\mathrm{Var}(X) = \mathrm{Cov}(X, X)$, $\mathrm{Var}(Y) = \mathrm{Cov}(Y, Y)$

# Covariance Intuition

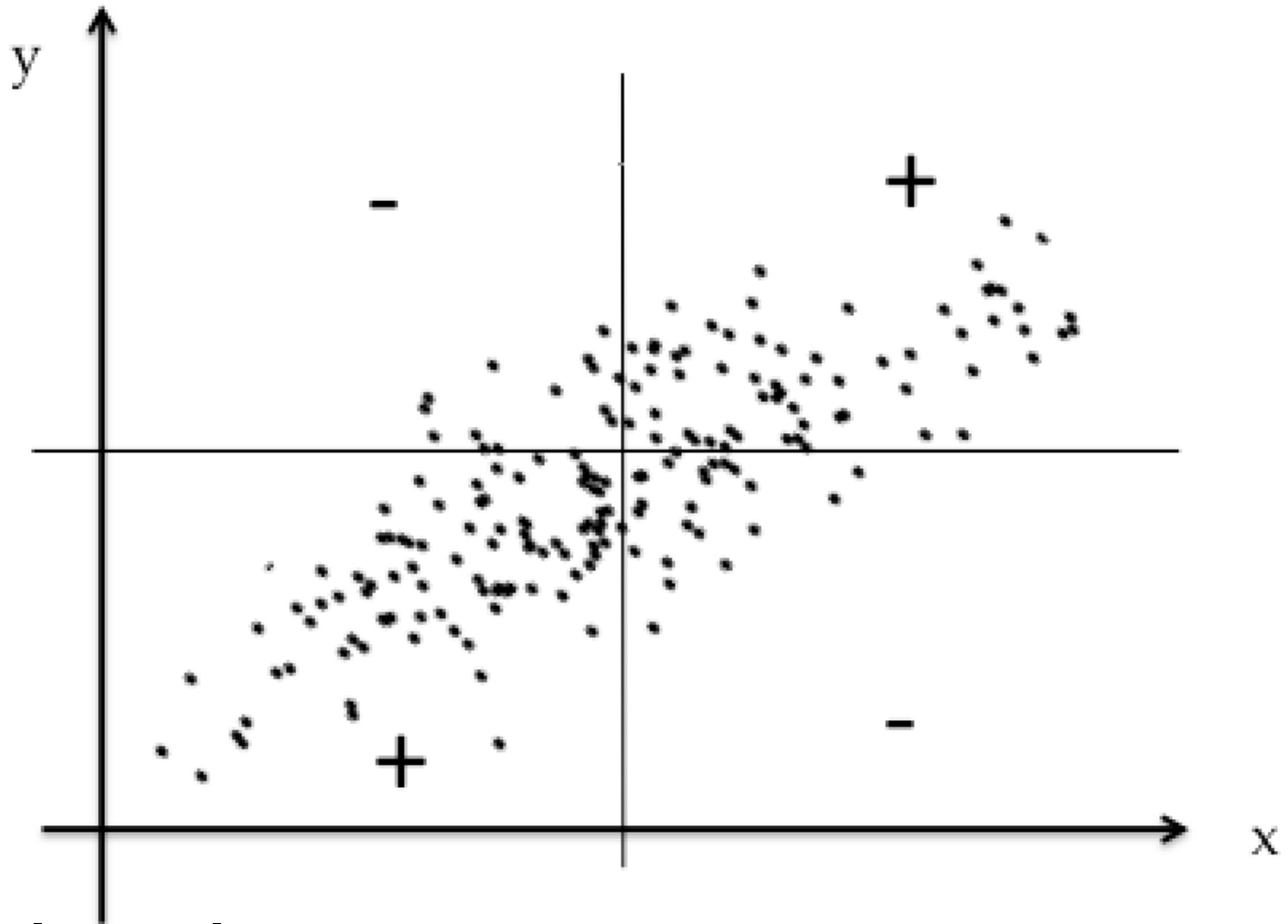


Figure: [Y.N.Wu] The sign of the covariance determines the trend of the linear relationship.

# Independence and Uncorrelated

Independence:

- Discrete: $p(x, y) = p_X(x)p_Y(y)$ for all $(x, y)$
- Continuous: $f(x, y) = f_X(x)f_Y(y)$ for all $(x, y)$

Uncorrelated: $\mathrm{Cov}(X, Y) = 0$.

Relation:

- $X$, $Y$ independent, then they are uncorrelated, i.e., $\mathrm{Cov}(X, Y) = 0$.
- $X$, $Y$ uncorrelated, then they MAY NOT be independent.

# Conditional Distribution

- Conditional Probability:
  - Discrete: $P(Y = y|X = x) = \frac{P(X=x,Y=y)}{P(X=x)}$
  - Continuous: $f_{Y|X}(y|x) = \frac{f(x,y)}{f_X(x)}$
- Conditional probability behaves like regular probability
- In general, $P(A|B)$ and $P(B|A)$ are not the same.

# Chain Rule of Conditional Probability

Any joint probability distribution over many random variables can be decomposed into conditional distribution over only one variable.

$$p(X_1, X_2, ..., X_n) = p(X_1)p(X_2|X_1)\dots p(X_n|X_1, X_2, ..., X_{n-1})$$

# Conditional Independence

$X$ and $Y$ are conditional independence given $Z$, $X \perp Y|Z$, if:

- $p(X|Y,Z) = p(X|Z)$ (Markov Property)
- $p(X,Y|Z) = p(X|Z)p(Y|Z)$ (Shared Cause Property)

# Bayes' Rule

Bayes rule: Given the prior probability $p(X)$, and conditional probability $p(Y|X)$, we can get the posterior probability $p(X|Y)$ by

$$\begin{aligned} p(X|Y) &= \frac{p(X,Y)}{p(Y)} \\ &= \frac{p(X,Y)}{\sum_x p(X,Y)} \\ &= \frac{p(X)p(Y|X)}{\sum_x p(X)p(Y|X)} \end{aligned}$$

For continuous random variable $X$ and $Y$, we have

$$f_{X|Y}(x|y) = \frac{f_{Y|X}(y|x)f_X(x)}{f_Y(y)} = \frac{f_{Y|X}(y|x)f_X(x)}{\int f_{Y|X}(y|x)f_X(x)dx}$$