



DE VINCI
INNOVATION
CENTER

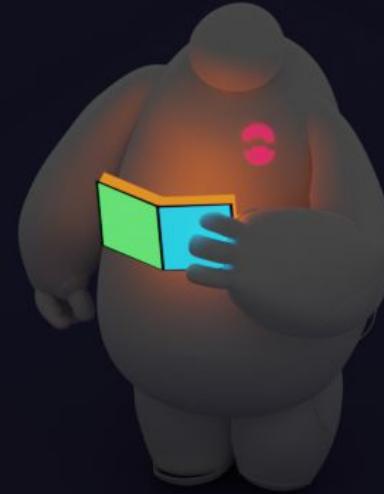
MACHINE LEARNING

Learning from Data

Yliess HATI

PHD Student - Computer Science

yliess.hati@devinci.fr



|00 INTRODUCTION



Pattern Recognition
Study **Correlations** Between **Data Domains**
Tune a Parametric Model with **Data**



|00 INTRODUCTION



SUPERVISED

Learn parameters from
Labeled Data

Regression

Classification

UNSUPERVISED

Learn parameters from
Unlabeled Data

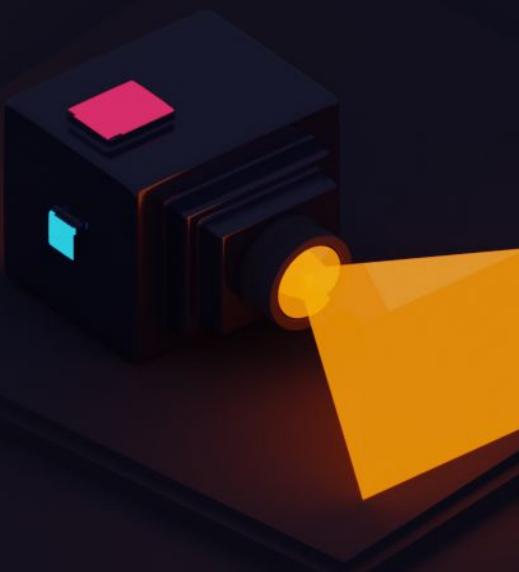
Clustering

Latent Space

REINFORCEMENT

Autonomous Agent Learning
from **Experience**

Q-Learning



01|

LINEAR ALGEBRA

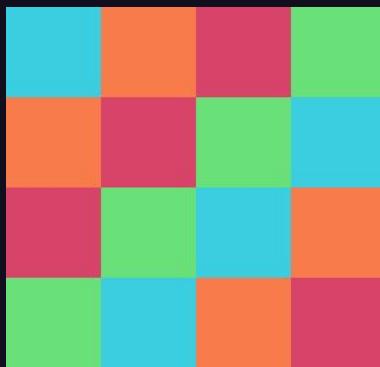
The Last Space Bender

|01 LINEAR ALGEBRA

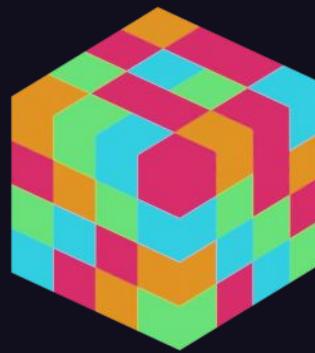
VECTOR



MATRIX



TENSOR



|01 LINEAR ALGEBRA

VECTOR



$$\vec{v} = \begin{pmatrix} v_x \\ v_y \end{pmatrix}$$

LENGTH

$$\|\vec{v}\| = \sqrt{v_x^2 + v_y^2}$$

|01 LINEAR ALGEBRA

VECTOR



ADDITION / SUBTRACTION

$$\vec{v} = \begin{pmatrix} v_x \\ v_y \end{pmatrix}$$

$$\vec{u} = \begin{pmatrix} u_x \\ u_y \end{pmatrix}$$

$$\vec{v} + \vec{u} = \begin{pmatrix} v_x + u_x \\ v_y + u_y \end{pmatrix}$$

|01 LINEAR ALGEBRA

VECTOR



SCALAR (DOT) PRODUCT

$$\vec{v} = \begin{pmatrix} v_x \\ v_y \end{pmatrix}$$

$$\vec{u} = \begin{pmatrix} u_x \\ u_y \end{pmatrix}$$

$$\vec{v} \cdot \vec{u} = v_x u_x + v_y u_y = |v| |u| \cos\theta$$

|01 LINEAR ALGEBRA

VECTOR



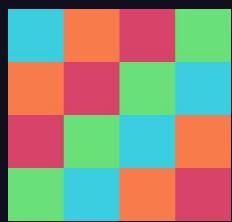
$$\vec{v} = \begin{pmatrix} v_x \\ v_y \\ v_z \end{pmatrix}$$
$$\vec{u} = \begin{pmatrix} u_x \\ u_y \\ u_z \end{pmatrix}$$

CROSS PRODUCT

$$\vec{v} \times \vec{u} = \begin{pmatrix} v_y u_z - v_z u_y \\ v_z u_x - v_x u_z \\ v_x u_y - v_y u_x \end{pmatrix}$$

|01 LINEAR ALGEBRA

MATRIX



ADDITION / SUBTRACTION

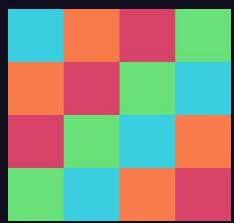
$$A = \begin{pmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{pmatrix}$$

$$B = \begin{pmatrix} b_{1,1} & b_{1,2} \\ b_{2,1} & b_{2,2} \end{pmatrix}$$

$$A + B = \begin{pmatrix} a_{1,1} + b_{1,1} & a_{1,2} + b_{1,2} \\ a_{2,1} + b_{2,1} & a_{2,2} + b_{2,2} \end{pmatrix}$$

|01 LINEAR ALGEBRA

MATRIX



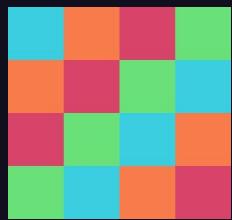
HADAMARD PRODUCT

$$A = \begin{pmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{pmatrix}$$
$$B = \begin{pmatrix} b_{1,1} & b_{1,2} \\ b_{2,1} & b_{2,2} \end{pmatrix}$$

$$A \odot B = \begin{pmatrix} a_{1,1}b_{1,1} & a_{1,2}b_{1,2} \\ a_{2,1}b_{2,1} & a_{2,2}b_{2,2} \end{pmatrix}$$

|01 LINEAR ALGEBRA

MATRIX



MATRIX MULTIPLICATION

$$A = \begin{pmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{pmatrix}$$

$$B = \begin{pmatrix} b_{1,1} & b_{1,2} \\ b_{2,1} & b_{2,2} \end{pmatrix}$$

$$A \cdot B = \begin{pmatrix} a_{1,1}b_{1,1} + a_{1,2}b_{2,1} & a_{1,1}b_{1,2} + a_{1,2}b_{2,2} \\ a_{2,1}b_{1,1} + a_{2,2}b_{2,1} & a_{2,1}b_{1,2} + a_{2,2}b_{2,2} \end{pmatrix}$$

|01 LINEAR ALGEBRA

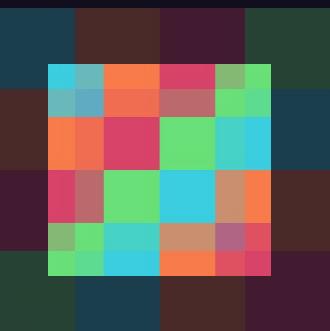
MATRIX



EYE

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

|01 LINEAR ALGEBRA



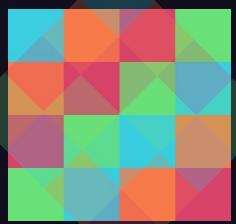
MATRIX

SCALE

$$\begin{pmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

|01 LINEAR ALGEBRA

MATRIX



ROTATION

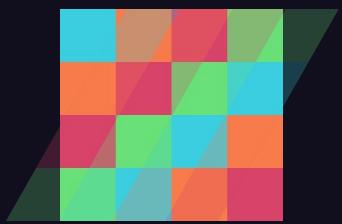
$$R_x(\theta) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ \cos(\theta) & -\sin(\theta) & 0 & 0 \\ \sin(\theta) & \cos(\theta) & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$R_y(\theta) = \begin{pmatrix} \cos(\theta) & 0 & \sin(\theta) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$R_z(\theta) = \begin{pmatrix} \cos(\theta) & -\sin(\theta) & 0 & 0 \\ \sin(\theta) & \cos(\theta) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

|01 LINEAR ALGEBRA

MATRIX

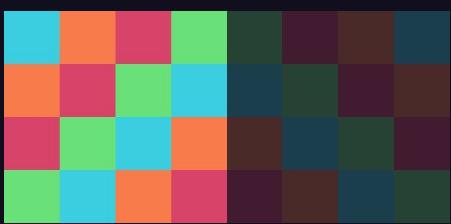


SHEAR

$$\begin{pmatrix} 1 & 0 & \lambda & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

|01 LINEAR ALGEBRA

MATRIX

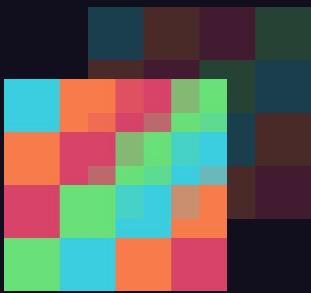


REFLECTION

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

|01 LINEAR ALGEBRA

MATRIX



TRANSLATION

$$\begin{pmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

102

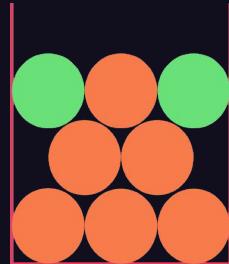
PROBABILITIES

Roll the Dice

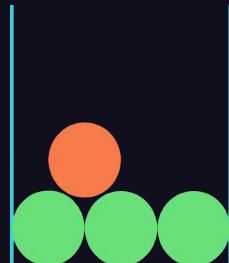


|02 PROBABILITIES

RANDOM VARIABLES



$$B = \begin{cases} r & \text{if Box is red} \\ b & \text{if Box is blue} \end{cases}$$

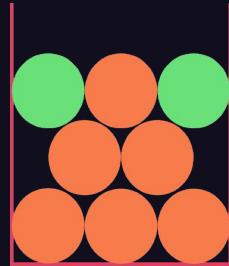


$$F = \begin{cases} a & \text{if Fruit is an apple} \\ o & \text{if Fruit is an orange} \end{cases}$$

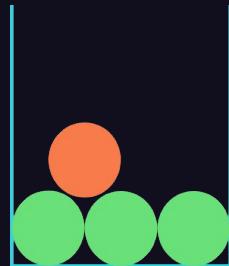
|02 PROBABILITIES



PROBABILITIES



$$p(B = r) = \frac{\# \text{ red boxes picked}}{\# \text{ total boxes picked}} \in [0; 1]$$



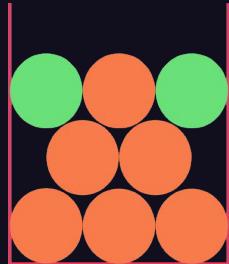
$$p(B = r) + p(B = b) = 1$$



|02 PROBABILITIES



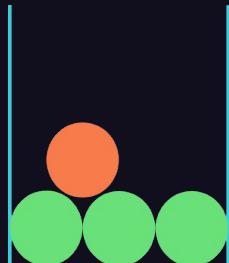
JOINT & CONDITIONAL



Sum Rule

Joint probabilities $p(B = r, F = a)$

Marginal probability $p(B = r) = p(B = r, F = a) + p(B = r, F = o)$



Product Rule

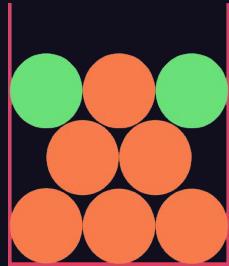
Conditional probabilities $p(B = r | F = o)$

$p(B = r, F = o) = p(F = o | B = r) p(B = r)$

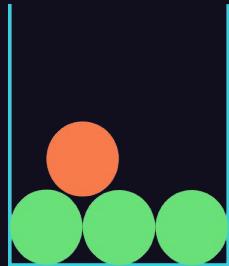
|02 PROBABILITIES



BAYES THEOREM



$$p(B = r|F = o) = \frac{p(F=o|B=r)p(B=r)}{p(F=o)}$$



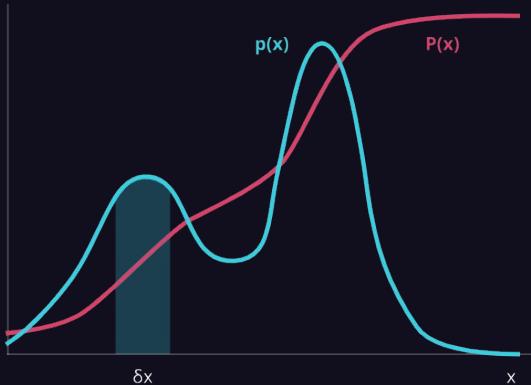
$$p(B = r|F = o) = \frac{p(F=o|B=r)p(B=r)}{p(F=o|B=r)p(B=r) + p(F=o|B=b)p(B=b)}$$



|02 PROBABILITIES



DENSITY



$$p(x \in (a, b)) = \int_a^b p(x)dx$$

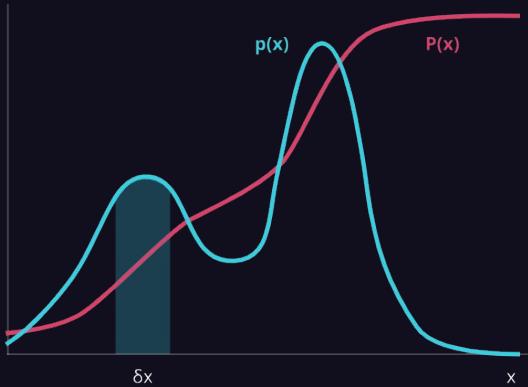
$$\int_{-\infty}^{+\infty} p(x)dx = 1$$

$$p(x) \geq 0$$

|02 PROBABILITIES



EXPECTATION & COVARIANCE



$$\mathbb{E}[f] = \int p(x)f(x)dx$$

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

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

$$cov[x, y] = \mathbb{E}_{x,y}[xy] - \mathbb{E}[x]\mathbb{E}[y]$$



A dark, abstract background featuring a dark blue-grey surface with several glowing spheres in green, pink, yellow, and light blue. The spheres appear to be scattered across the surface, with some having faint trails or reflections.

03|

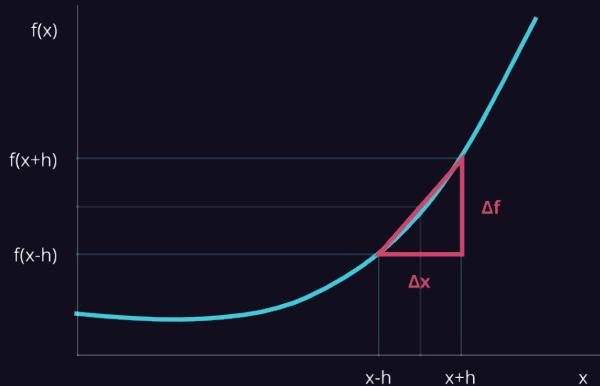
OPTIMIZATION

One Method to Rule them All

|03 OPTIMIZATION



DERIVATIVES



$$f'(x) = \lim_{h \rightarrow 0} \frac{\Delta f}{\Delta x} = \lim_{h \rightarrow 0} \frac{f(x+h)-f(x-h)}{2h}$$

First order Derivative

Direction of the Slope

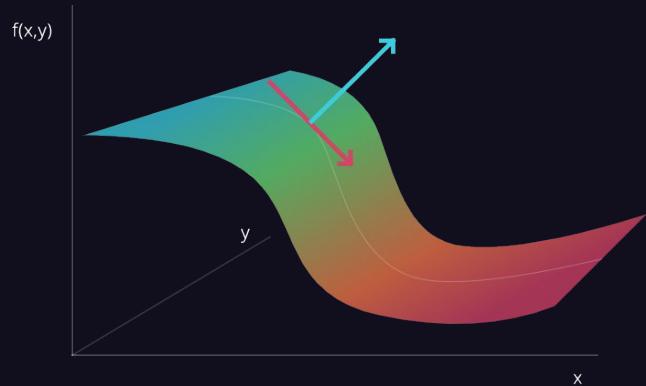
Second order Derivative

Rate of Changes in the Slope

|03 OPTIMIZATION

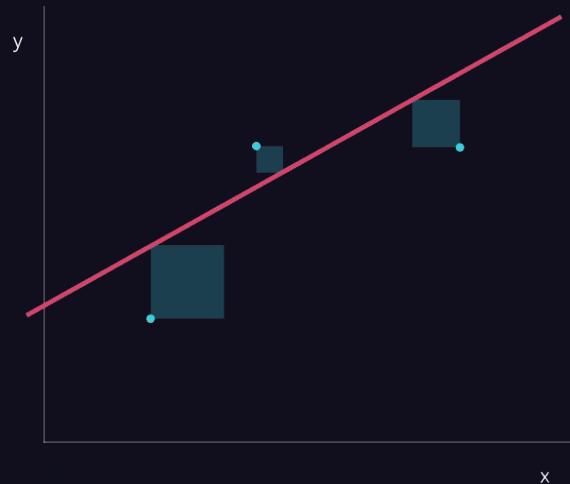


GRADIENTS



$$\nabla f = \begin{pmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{pmatrix}$$

|03 OPTIMIZATION



LEAST SQUARES

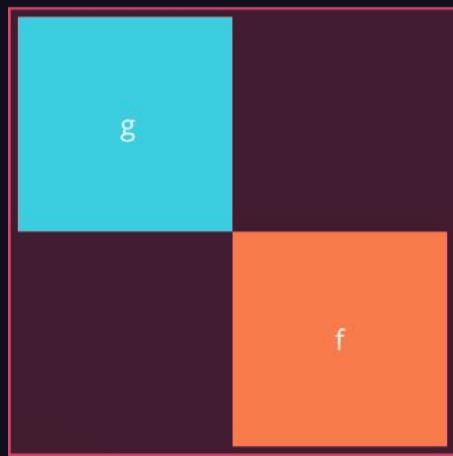
Objective Function

Minimize Squared Distances
from Expected Value

$$\hat{y} = ax + b$$

$$\sum_i (y_i - \hat{y}_i)^2 = 0$$

|03 OPTIMIZATION

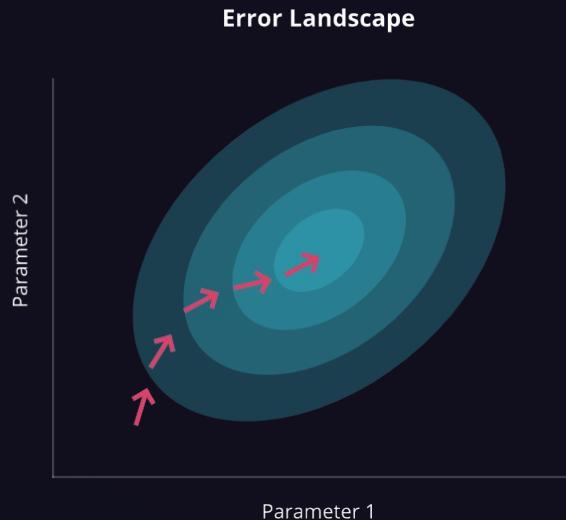


CHAIN RULE

$$h = g \circ f$$

$$h'(x) = g'(f(x))f'(x)$$

|03 OPTIMIZATION



GRADIENT DESCENT

Steps

- 1) **Forward Propagate** Through the Chain
- 2) Compute Output **Error**
- 3) **Backpropagate** Error Through the Chain
- 4) **Update** Weights w/ Learning Rate
- 5) **Repeat** Until Convergence Threshold

|03 OPTIMIZATION



DATASET SPLIT

Training Set

Samples used to Fit/Train the Model

Validation Set

Samples used to provide an Unbiased Evaluation of the Model
Becomes Biased during Training

Testing Set

Samples used to provide an Unbiased Evaluation of the Model
after Training

|03 OPTIMIZATION



CROSS-VALIDATION



Method

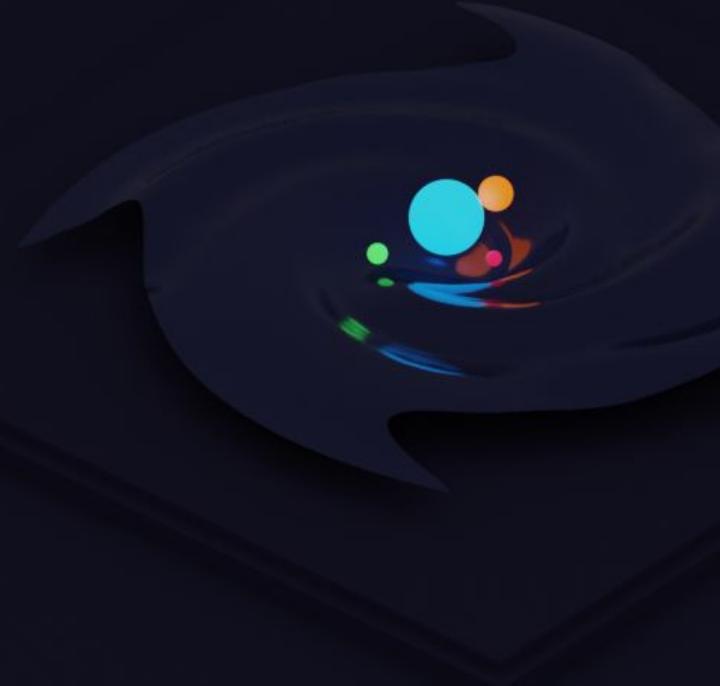
- 1) Split Dataset into **k-Folds**
- 2) **Train on k-1 Folds** and Validate w/ Last
- 3) **Repeat** k-times
- 4) Use **Ensemble** Method for **Inference** or **Retrain** on all Dataset

|04

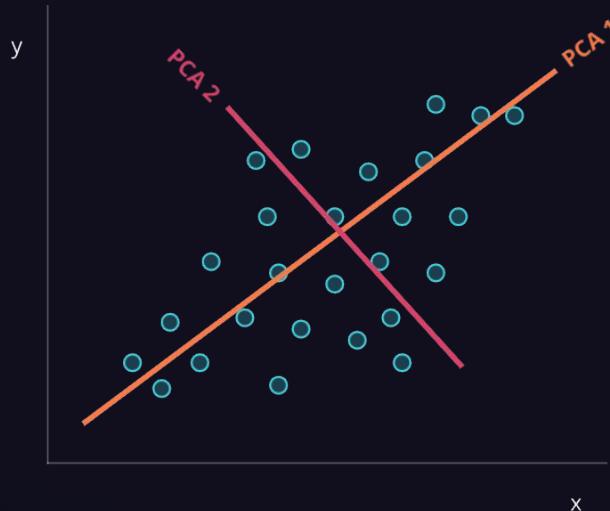
DIMENSIONALITY REDUCTION

Small Worlds are Filled with
Things to See

Paul Safranek



|04 DIMENSIONALITY REDUCTION

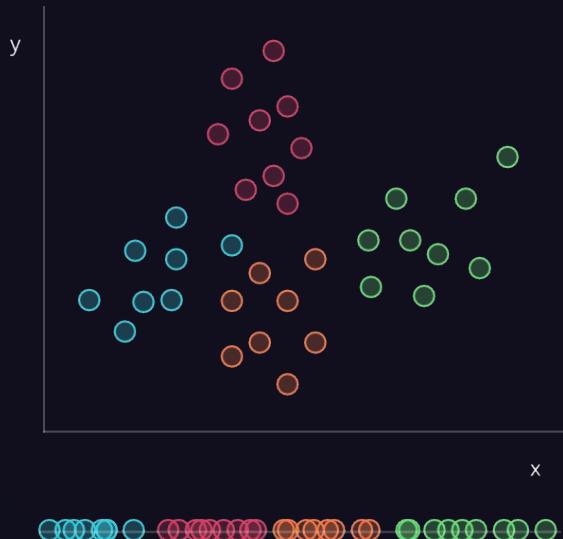


PCA

$$C = \begin{pmatrix} Var(x) & Cov(x, y) & Cov(x, z) \\ Cov(y, x) & Var(y) & Cov(y, z) \\ Cov(z, x) & Cov(z, y) & Var(z) \end{pmatrix}$$

v_1	λ_1	↑ Big
v_2	λ_2	
v_3	λ_3	Small

|04 DIMENSIONALITY REDUCTION



t-SNE

Steps

- 1) Use **Normal Distribution** to Estimate **Similarity** b/ Data Points
- 2) Create another Distribution (**t-distribution**) Capturing the **Same Similarity** Between Data Points using **Gradient Descent** on **KL-divergence**

05|

SUPERVISED LEARNING

Keep Calm & Label them All



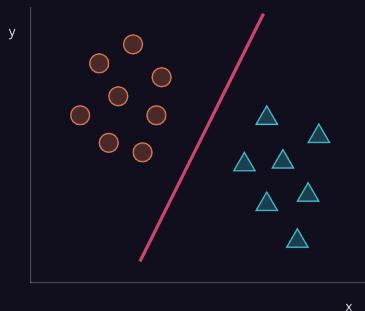
|05 SUPERVISED LEARNING



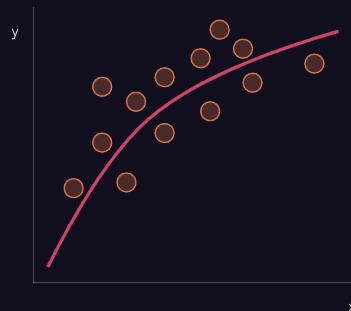
LEARN FROM LABELED DATA

$$Y = f(X)$$

Classification

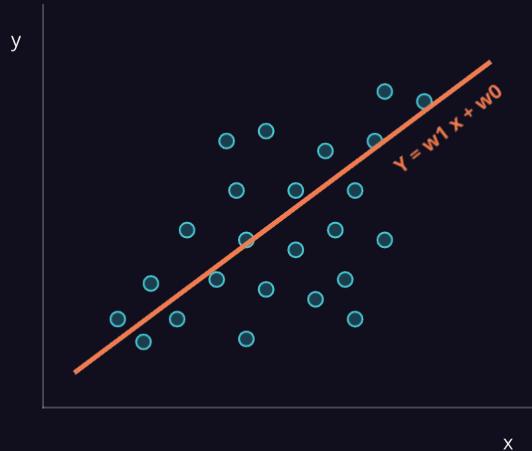


Regression



|05 SUPERVISED LEARNING

LINEAR REGRESSION



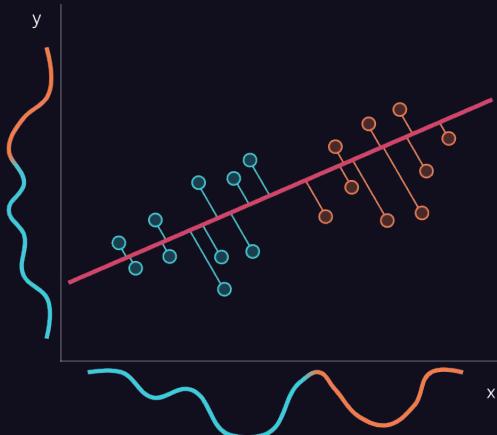
$$Y = f(W, X) = XW^t + \epsilon$$

$$\mathcal{L} = MSE(\hat{Y}, Y) = \frac{1}{N} \sum_i (w_i x_i - y_i)^2$$

$$\frac{\partial \mathcal{L}}{\partial W} = \frac{2}{N} (X^t X W - X^t Y) = 0$$

|05 SUPERVISED LEARNING

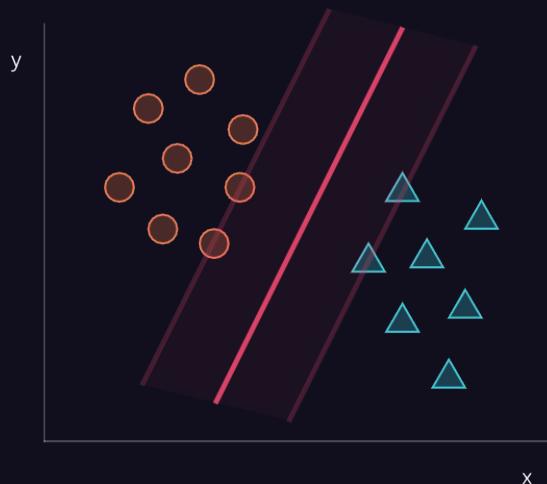
LINEAR DISCRIMINANT ANALYSIS



$$Y = f(W, X) = XW^t + \epsilon$$

$$\text{Objective}(W) = \frac{|\tilde{\mu}_1 - \tilde{\mu}_2|}{\tilde{s}_1^2 + \tilde{s}_2^2}$$

|05 SUPERVISED LEARNING



SUPPORT VECTOR MACHINE

$$Y_i(WX + b) - 1 \geq 0$$

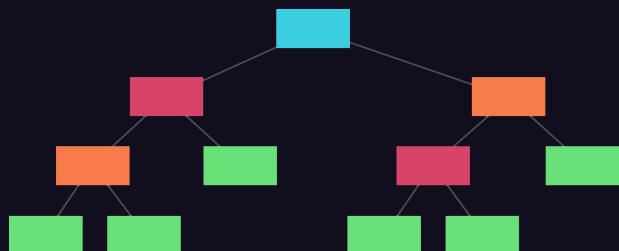
$$\text{margin} = (X_+ - X_-) \frac{W}{\|W\|}$$

$$J(W) = \frac{1}{2} \|W\|^2 + C(\frac{1}{N} \sum_i \max(0, 1 - y_i(wx_i + b)))$$

|05 SUPERVISED LEARNING



DECISION TREE

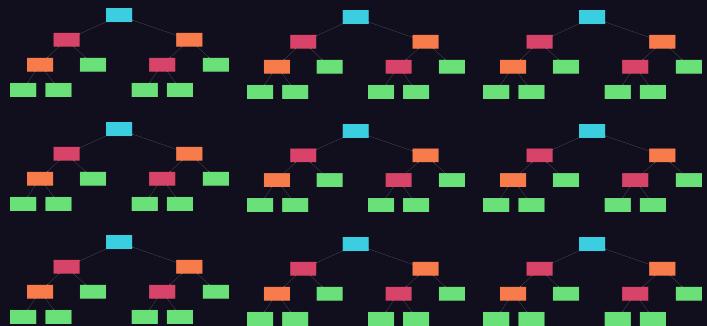


Steps

- 1) Compute all **Gini Impurity Scores**
- 2) **Lowest Impurity Score Becomes a Leaf**
- 3) Else **if Improvement** pick the Separation with the **Lowest Score**

$$Gini(K) = \sum_i P_{i,K}(1 - P_{i,K}) = 1 - \sum_i P_{i,K}^2$$

|05 SUPERVISED LEARNING



RANDOM FOREST

Intuition

Forest of **Dense Decision Trees**

Like **Cross Validation**

Ensemble Methods give Better Results

Use Voting, Max or Average

|05 SUPERVISED LEARNING

GRADIENT BOOSTING



Steps

- 1) Start from a Single Leaf of the **Average Prediction**
- 2) Build a **New Tree** to Predict the **Residual Error** from the Previous Tree and **Weight** its Contribution by a **Learning Rate**
- 3) Do **Until Convergence** Condition

|06

UNSUPERVISED LEARNING

In my Defense I was Left
Unsupervised



|06 UNSUPERVISED LEARNING



K-MEANS

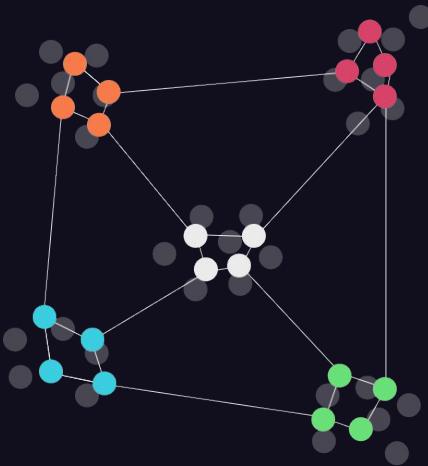
Steps

- 1) **Randomly Place K Centroids**
- 2) **Assign** each Point to the **Closest Centroid**
- 3) **Move** Centroids to its **Cluster's Mean**
- 4) Do **Until Convergence** Condition

|06 UNSUPERVISED LEARNING



SELF-ORGANIZING MAP

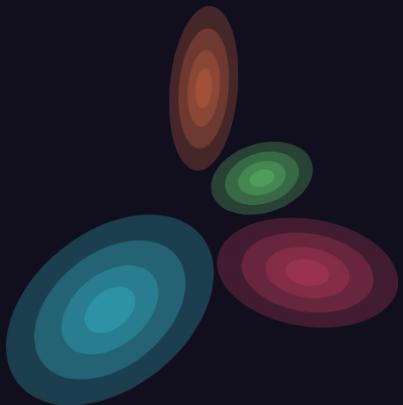


Steps

- 1) Randomly Initialize the **Neurons**
- 2) Compute **Distance** from **Training Sample** and **Each Lattice**
- 3) The **Closest** is Declared as **Winner**
- 4) **Update** Winner and Neighbours **Weights**

$$W_i(s+1) = W_i(s) + \eta \cdot \text{Neighbour}(i, p) \cdot \text{Distance}(X - W_i(s))$$

|06 UNSUPERVISED LEARNING



MIXTURE OF GAUSSIANS

$$p(x) = \sum_i \alpha_i \mathcal{N}(x|\mu_i, \Sigma_i)$$

Expectation Maximization

- 1) **Expectation:** Compute the **Likelihood** of each Datum **being part of Class k** with Initial Parameters

$$\gamma_{i,k} = \frac{\alpha_k \mathcal{N}(x_i|\mu_k, \Sigma_k)}{\sum_{j=1}^K \alpha_j \mathcal{N}(x_i|\mu_j, \Sigma_j)}$$

- 2) **Maximization:** Update each **Parameter** to **Maximize the Expectation**

$$\alpha_k = \sum_{i=1}^N \frac{\gamma_{i,k}}{N} \quad \mu_k = \frac{\sum_{i=1}^N \gamma_{i,k} x_i}{\sum_{i=1}^N \gamma_{i,k}} \quad \sigma_k = \frac{\sum_{i=1}^N \gamma_{i,k} (x_i - \mu_k)^2}{\sum_{i=1}^N \gamma_{i,k}}$$



07|

REFERENCE

Not Sponsored

Pattern Recognition and Machine Learning
Christopher Bishop