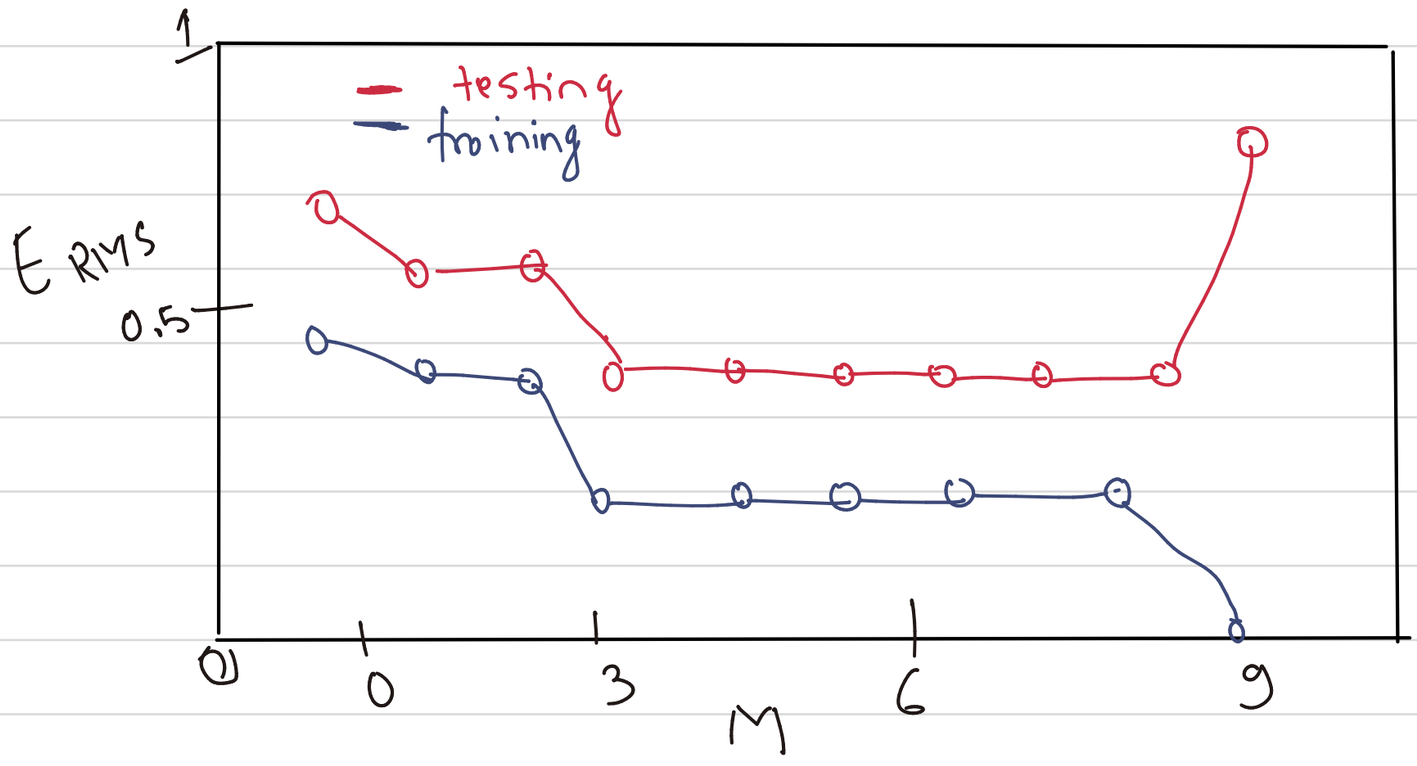
Day-9, Oct-11, 2024

Bias- Variance trade-off

- test data: a different sample from the same true function



Root Mean Square (RMS) Error

- training error goes to zero, but test error increases with M

RMSE =>

$$\sqrt{\frac{(Y_i \hat{Y}_i)^2}{n}}$$

Decrease Variance ↓	↑ Increase Bias (Noise)	Increase variance ↑	↓ Decrease Bias

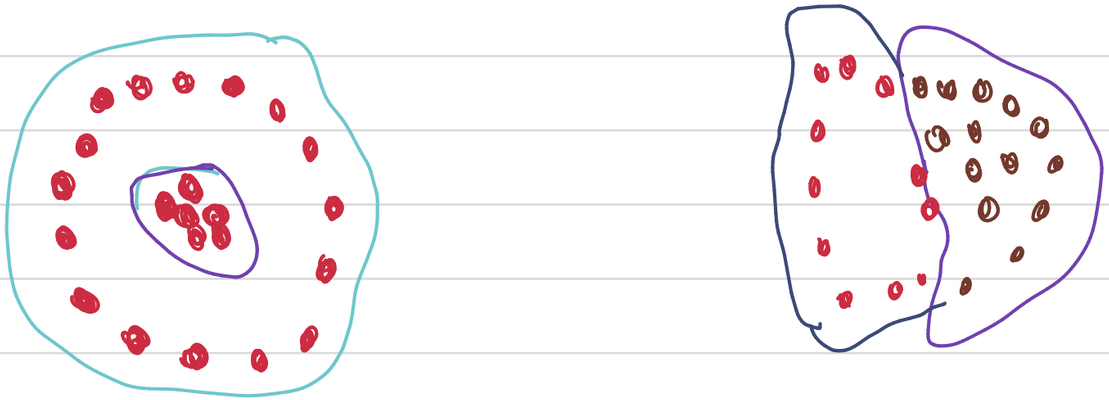
Unsupervised Learning

What is Data Clustering?

→ the organization of unlabeled data into Similarity group

called Clusters.

→ A cluster is a collection of data items which are similar between them, and dissimilar to data items in other clusters.



→ Clusters the Similar group of data using mathematical techniques like centroid, distance from the center, Characteristics of data-points and so on.

What do we need for clustering?

(1) Proximity Measure:

> Similarity measure $S(X_i, X_k)$: large if X_i, X_k are similar

> dissimilarity (or distance) measure $D(X_i, X_k)$: small if X_i, X_k are similar

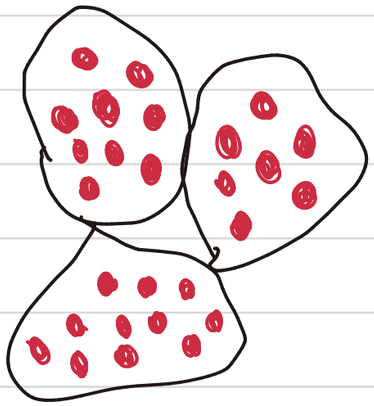
large d, small s

large s, small d

that means A large distance (d) between points implies small similarity (s)

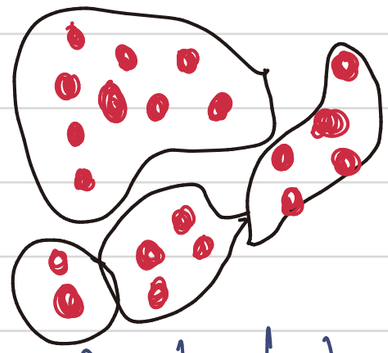
→ A small distance means large similarity between the points.

② Criterion function to evaluate a clustering.



→ Good clustering

↑ when points that are more similar to each other are grouped together.



Bad clustering

→ When similar points are not well-grouped, indicating poor clustering quality.

③ Algorithm to Compute Clustering

- for example, by optimizing the criterion function.

K-Means Clustering (Mathematical Approach)

- K-means (MacQueen, 1967) is a partitional clustering algorithm
- Let the set of data points D be $x_1, x_2, \ldots\ldots x_n$
  - Where $x_i = (x_{i1}, x_{i2}, x_{i3} \ldots\ldots x_{ir})$ is a vector in $X \subseteq R^n$ and n is the number of dimensions.
- The K-means algorithm partitions the given data into K clusters.
  - Each cluster has a cluster center C_i called Centroid
    - K is specified by the user.

- Convergence (Stopping) Criterion
  - minimum decrease in the sum of squared error (SSE)

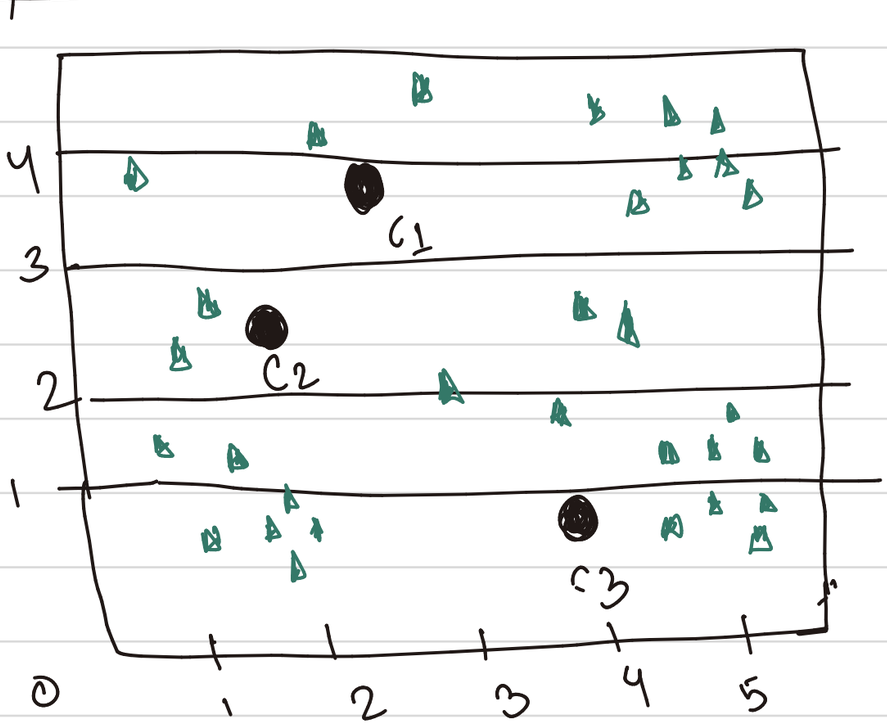
$$SSE = \sum_{j=1}^{k} \sum_{x \in C_j} d(x, m_j)^2$$

K-Means clustering: Working Example

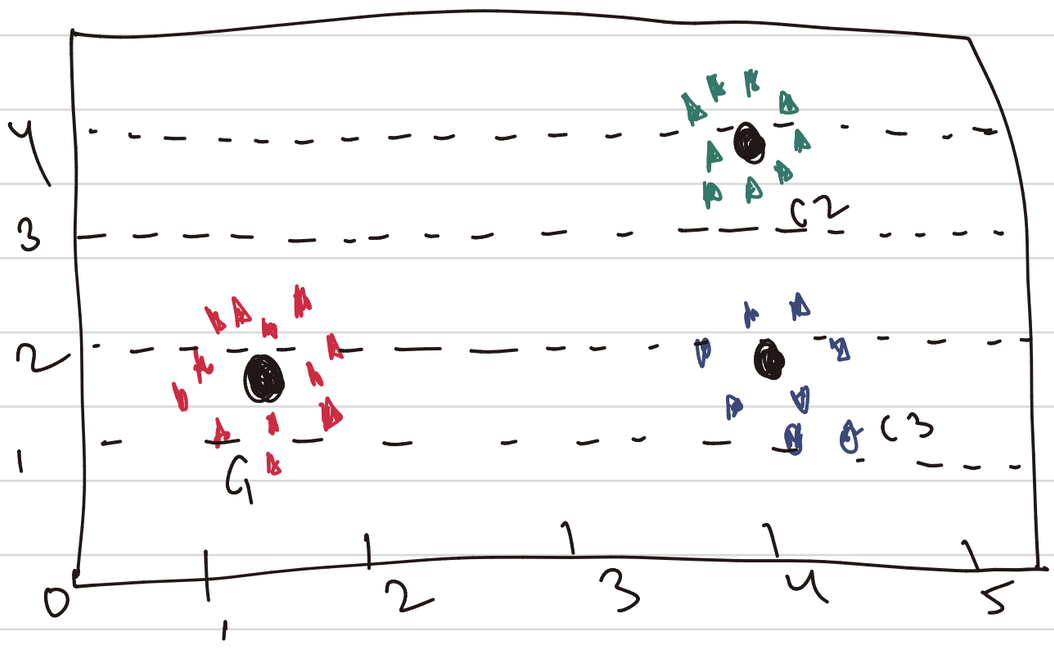
Where C_1, C_2, C_3 are the centroids

* greens are dobb points

we use distance metics.



then after apply K-means clustering finally we cluster them properly:



Why to use K-Means?

- Strengths:
  - Simple: easy to understand and to implement
  - Efficient: Time Complexity: O(tKn)

where n is the number of data points

K is the number of clusters and t is the number of iterations

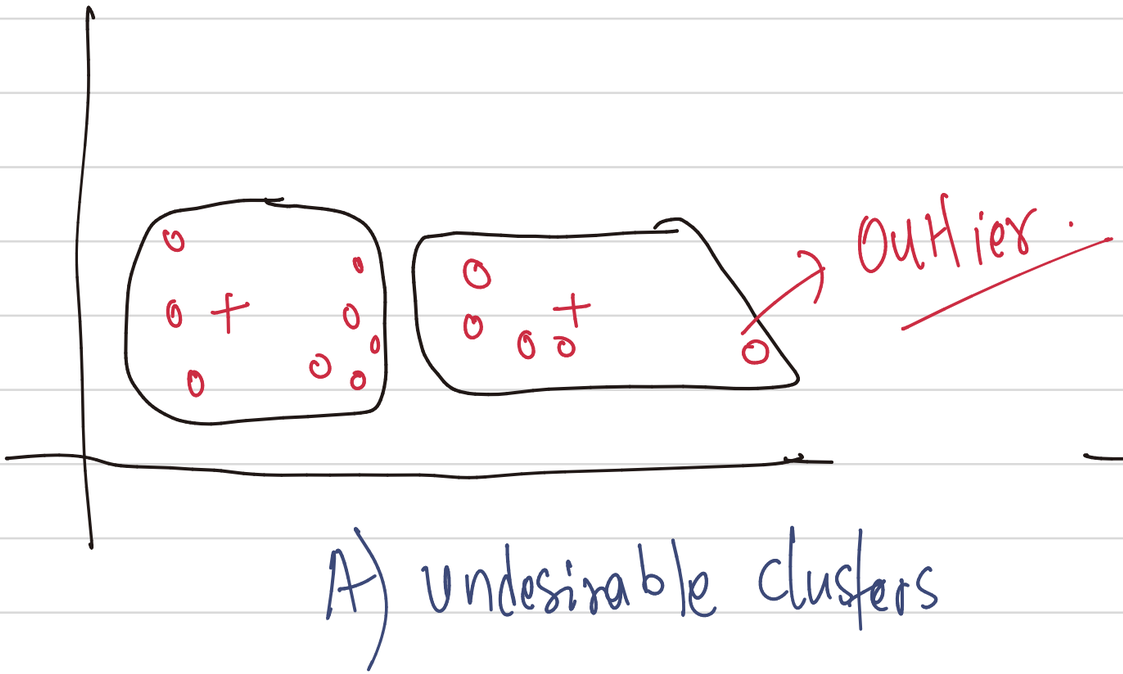
* Since both K and t are small. K-means is considered a linear Algorithm.

- K-means is the most popular clustering Algorithm

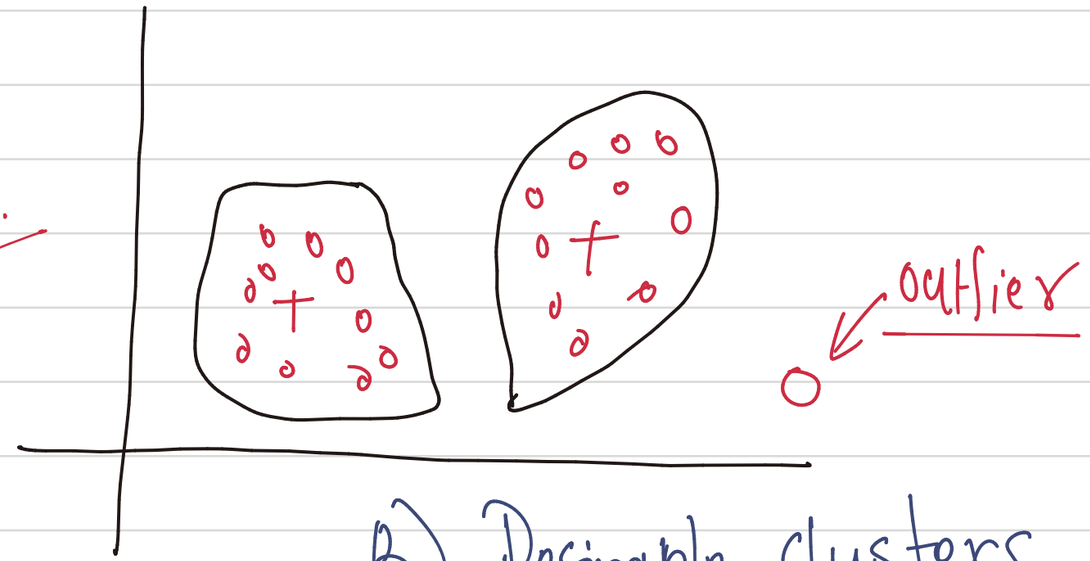
- Note that: If terminates at a local optium SSE is used. the global optium is hard to find due to Complexity.

- SSE is the Stopping Criteria.
- Need to define the 'K' clusters and mean, Sensitive to outliers because of distance metics.

Outliers



A) Undesirable clusters



B) Desirable clusters

K- Nearest Neighbors

- K-nearest neighbours uses the local neighbourhood to obtain a prediction.
- The k memorized examples more similar to the one that is being

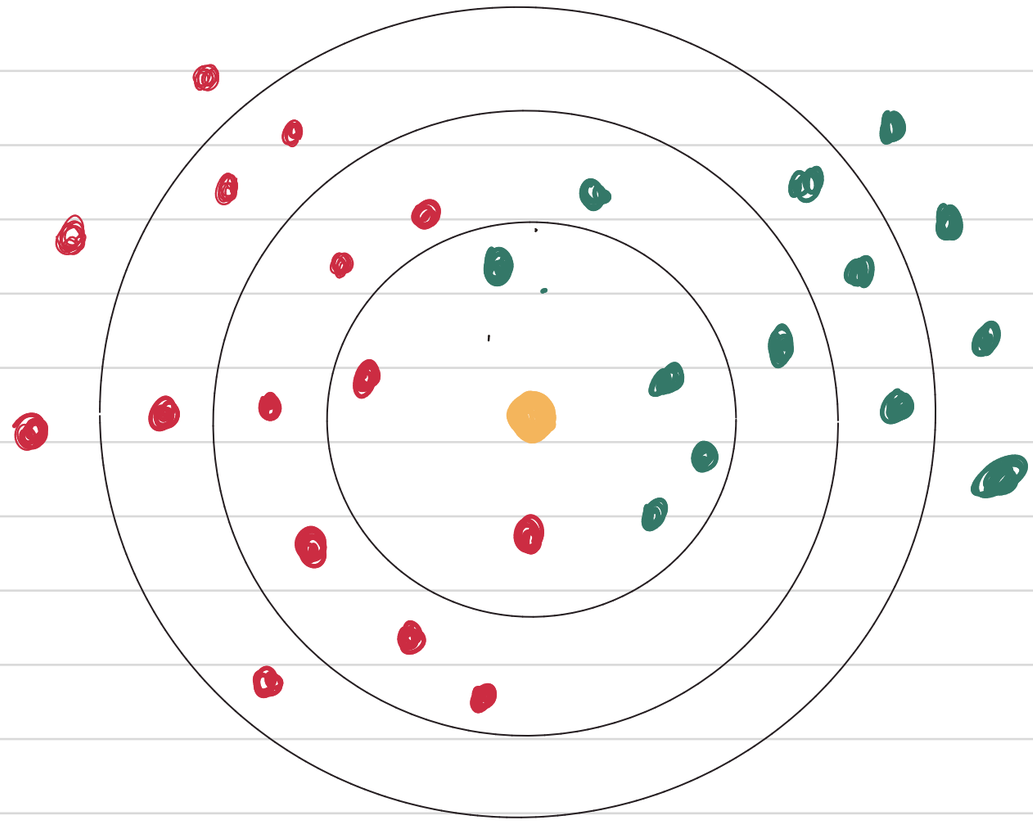
Classified are retrieved.

- A distance function is needed to compare the examples similarity
  - Euclidean Distance $d(X_j, X_K) \Rightarrow \sqrt{\Sigma_i (X_{j,i} - X_{K,i})^2}$
  - Manhattan Distance $d(X_j, X_K) \Rightarrow \sqrt{\Sigma_i (X_{j,i} - X_{K,i})^2}$

Which shows that if we change the distance function, we change how examples are classified.

Based on the nearest neighbour giving the data points as label.

→ does the voting to the nearest neighbours.



→ very sensitive to distance formula used

→ In the above figure for a yellow dot the neorest neighbours are 4 green and 2 red dots..

Support Vector Machine (SVM)

→ Linear Data represented by linear Equation

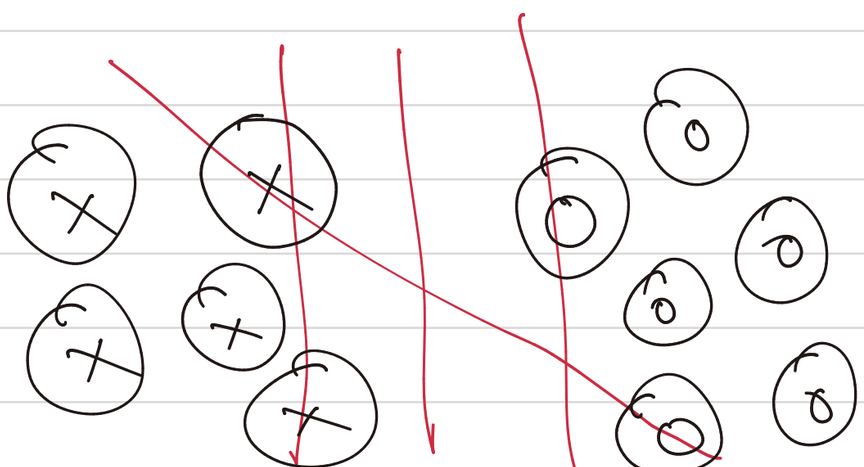
→ Complex Data is backed by SVM

→ Wants to separate two data points.

A 'Good' Separator tends to reduce noise in observations

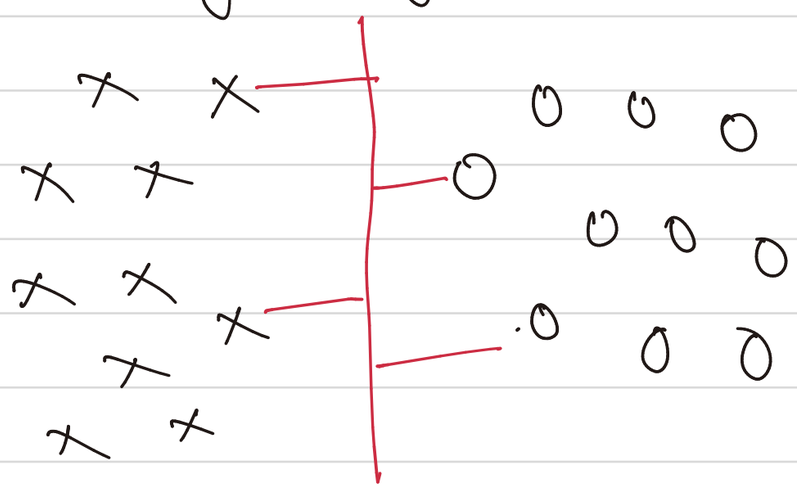
Intuitions Figures for SVM

1)

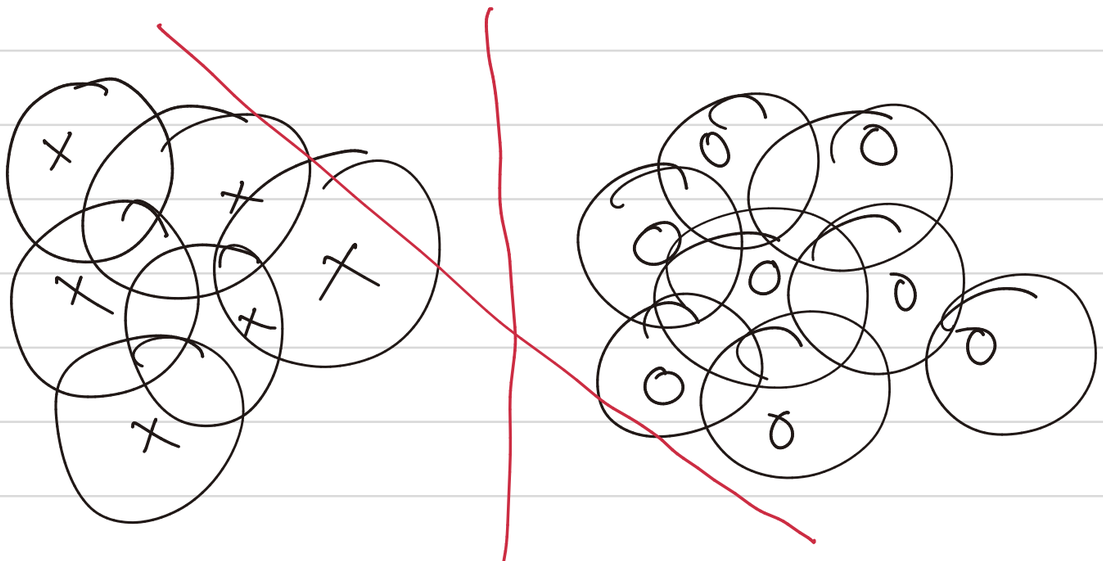


→ Best Solution will have the high marginal value.

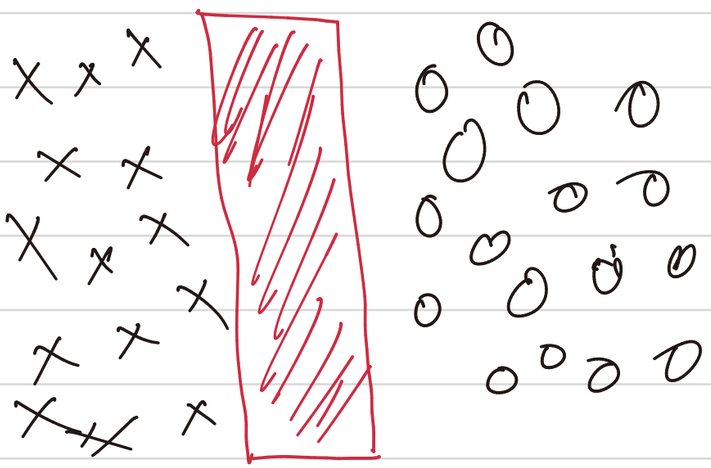
3)



2)



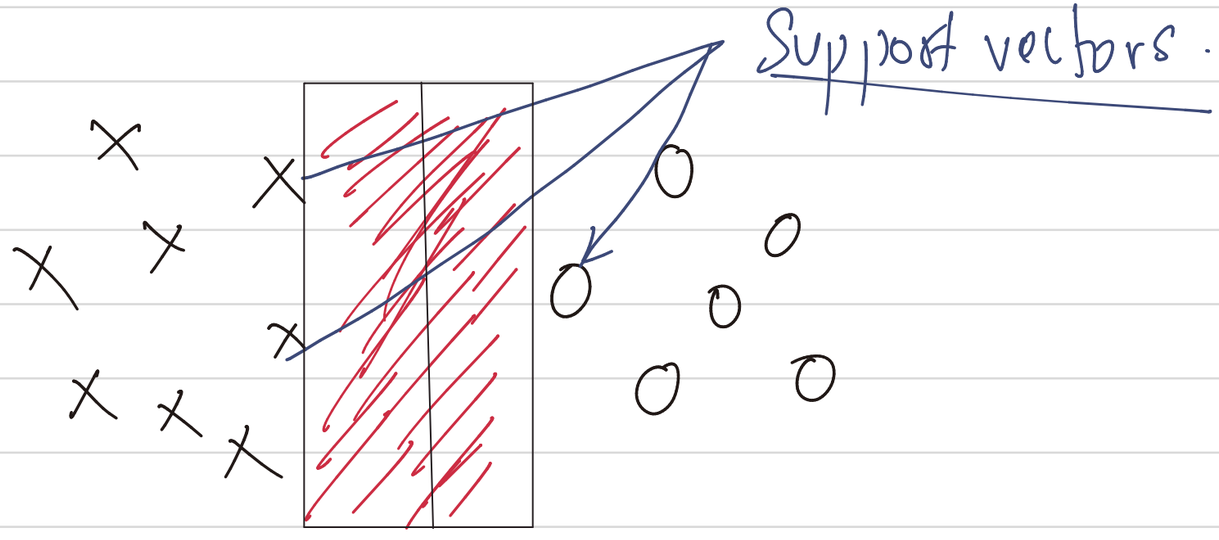
4)



Best → Much More Margin / Distance

→ Maximizes the margin

- SVMs maximize the margin around the Separating Hyperplane
  - A.ka large margin classifier.



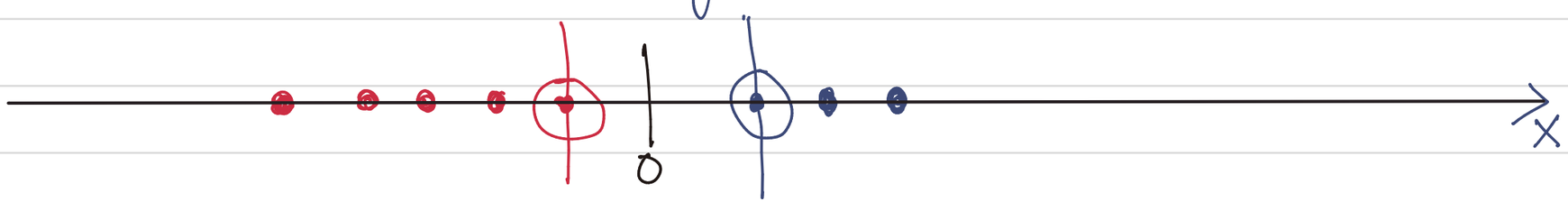
- The decision function is fully specified by a subset of training samples, the support vectors
- Solving SVMs is a quadratic programming problem
- Optimize by increasing the margin

Why might predictions be wrong?

- True non-determinism
- partial observability
  - hard, soft
- Representational basis
- Algorithmic bias
- Bounded Resources

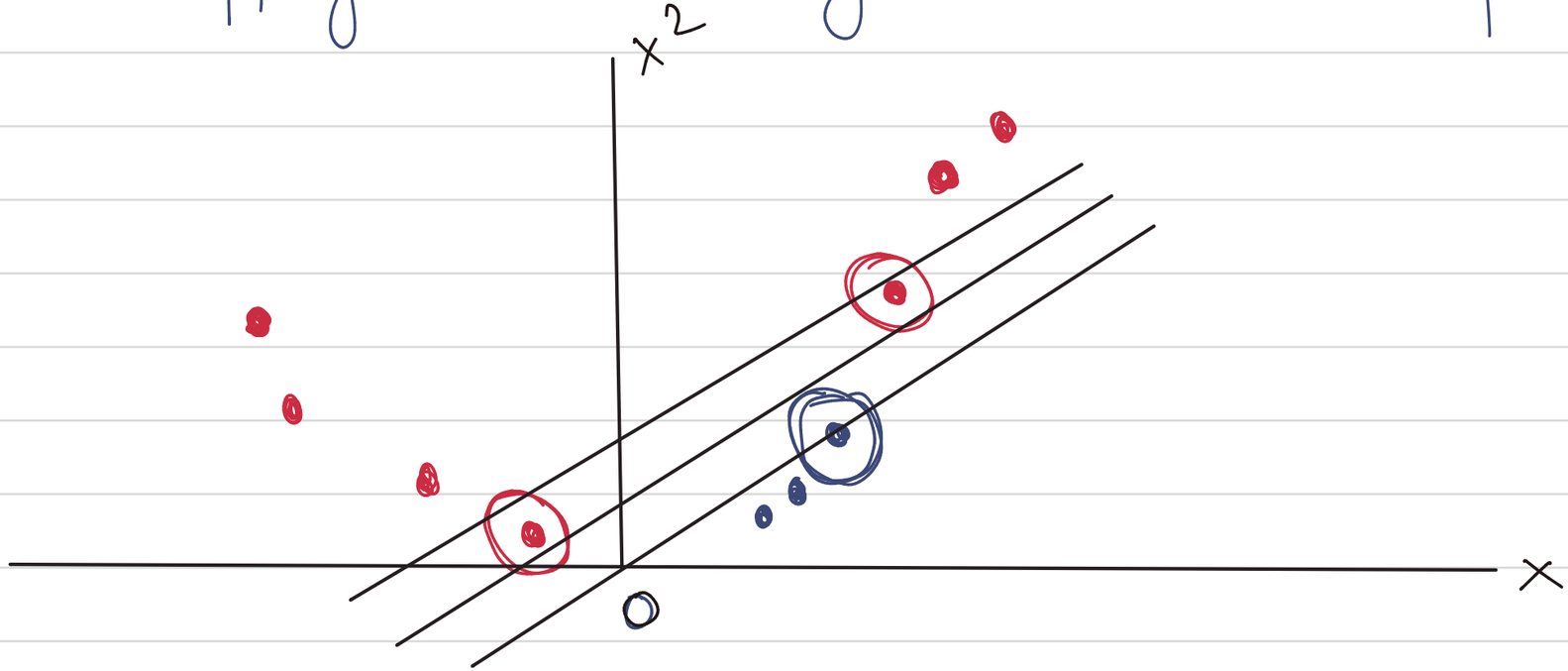
Support Vector Machine (Reasonable ways to use)

- Datasets that are Linearly Separable works great



- But what are we going to do if the dataset is just too hard?

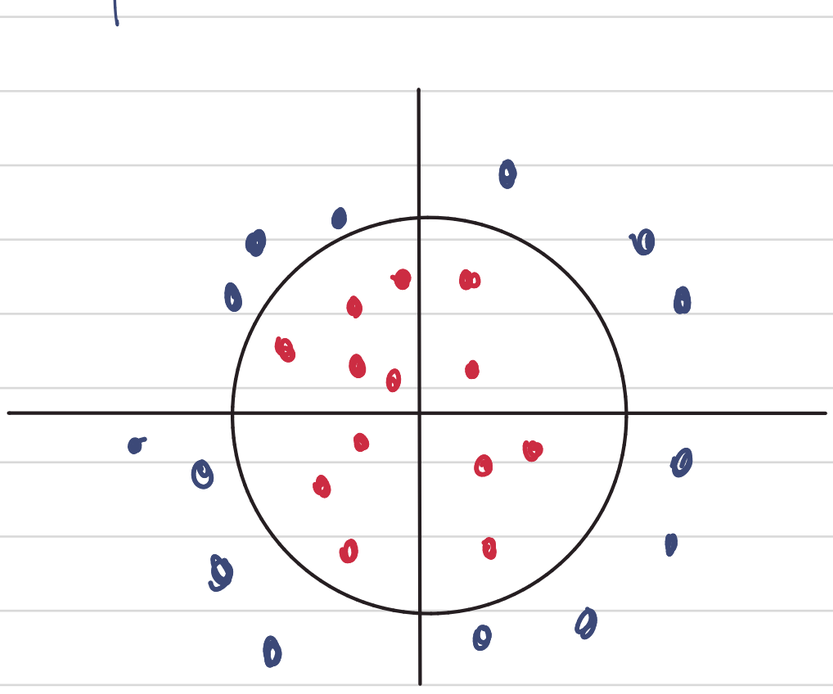
- How about mapping data to a higher-dimensional space?



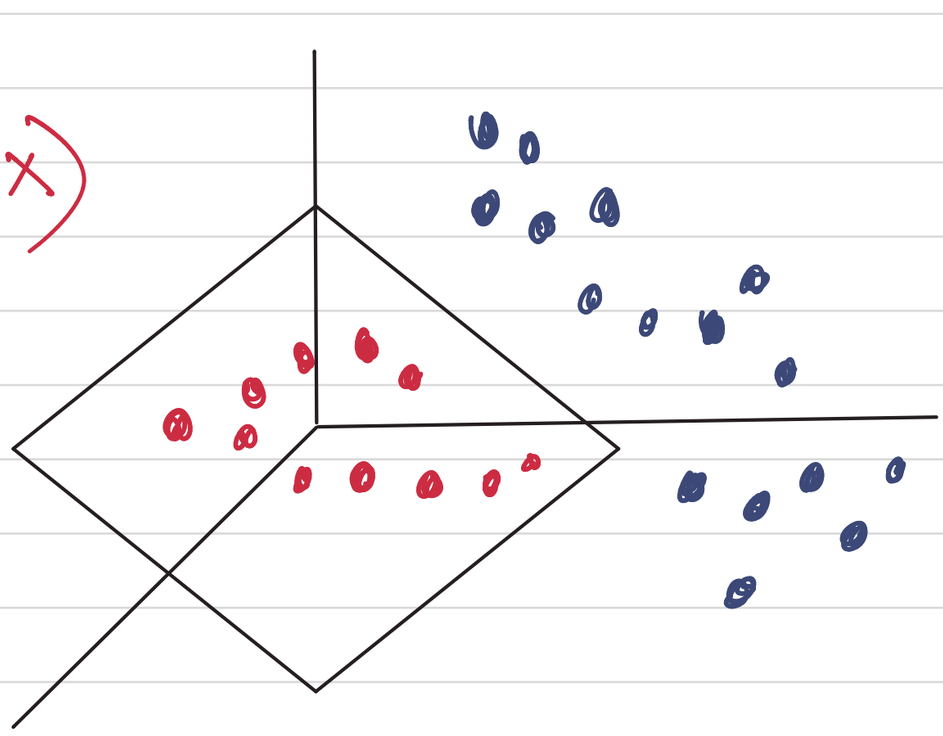
Yes, SVMs can work well with complex-data which are non-linear.

Non-linear SVM: Feature Spaces

- General Idea: The original feature space can always be mapped to some higher-dimensional feature space where the training set is separable.



$\phi : x \rightarrow \psi(x)$



→ Non-linear data into Separable in a higher dimensional Space.

The "Kernel Trick"

- The linear classifier relies on an inner product between vectors

$$K(x_i, x_j) = (x_i)^T (x_j)$$

- if every datapoint is mapped into high-dimensional space via some transformation $\phi : x \rightarrow \varphi(x)$, the inner product becomes

$$K(x_i, x_j) = \varphi(x_i)^T \varphi(x_j)$$

- A Kernel function is some function that corresponds to an inner product in some expanded function feature space.

- Make non-Separable problem separable
  - Map data into better representational Space.

- Common Kernels
  - linear
  - polynomials $K(x, z) = (1 + x^T z)^d$
  - Radial basis function (infinite dimensional Space)

$$K(x_i, x_j) = e(\|x_i - x_j\|^2) / 2\sigma^2$$

Regularization and Evaluation

We need regularization so that low-bias in the model doesnot get overfit.

We need trade-bias figure to explain this Since our model must not learn noise. So, Regularization helps us to get good and balanced surface for bias.

$$MSE = bias^2 + Variance.$$

Not only finding the predicted loss, we add penalty (β) in terms of regularization which helps model not to learn noise and handle add or look the uncertainty.

In Regression, when $p >> n$, non-convex optimization. Regularization is introduced as a convex function of the regression model.

argmin_β loss + Penalty (β)

Regularization- L_1 and L_2 norms

1) Lasso penalty

$$\text{argmin}_\beta \text{loss} + \lambda |\beta|$$

(L_1 β)

→ β are parameters

2) Ridge penalty

$$\text{argmin}_\beta \text{loss} + \lambda |\beta|_2 \rightarrow$$

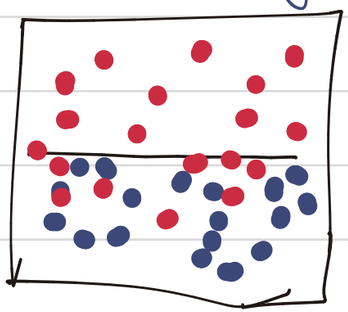
How to choose λ ?

With Regularization, the model will focus on the important things like size of the house instead of random details.

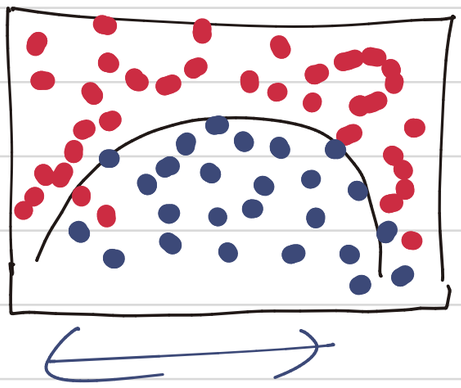
Overfitting happens when model learns data, working poorly on unseen data.

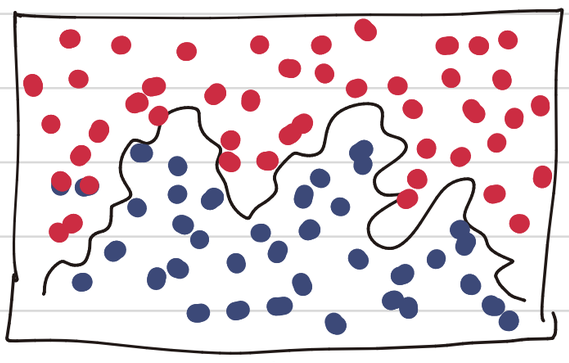
Generalization Problem in Regression / Classification

- Best-fit will be somewhere between under-fitting and Over-fitting.



Underfitting





Overfitting

Cross-Validation

→ technique used in ML to evaluate how well a model performs on unseen data by spilting data into subsets (folds)


Total no. of examples
Experiment 1
Experiment 2
Experiment 3
Experiment N
Single test Example
Test Data
Training Data
Iteration 1
Iteration 2
Iteration N
All data

AI agents: An AI agent can be anything that perceive, think and act. It takes input from sensory, performs Rational thinking and take suitable action.

Types of AI Agent

1) Human-Agent
2) Robotic Agent
3) Software Agent (the one (2024) currently Hyped in the ML model)

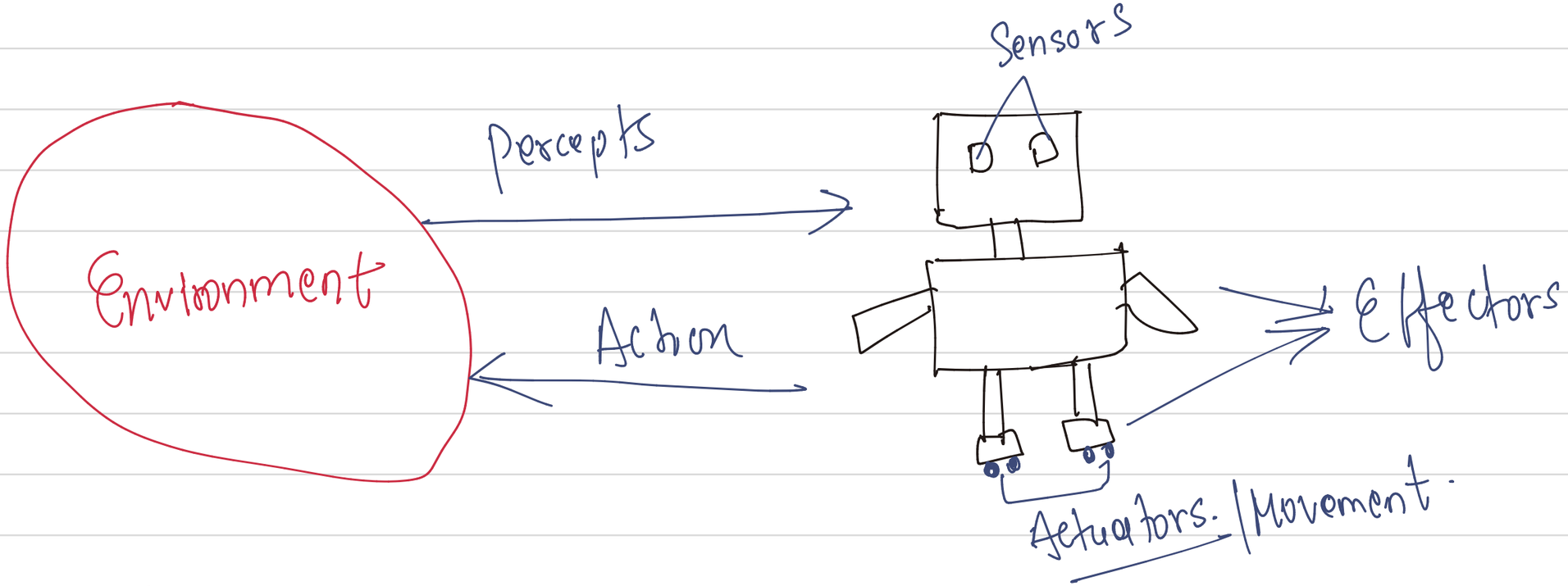
Components of Agents:

Main parts of Agents are:

(1) Sensors: Agent observe environment via Sensors, takes input, Camera, Sensors, detect the changes, sends/receive information

(2) Actuators: Motor Gears, Rails that converts energy into motion and movements, responsible for controlling system

(3) Effectors: parts of the AI Agents, legs, arms, wheels, fingers, fins, display screen, that can affect the environment and helps to take action.



Rules for AI agents:

① Must Have Ability to perceive the environment

② Observation must be used to make decision

③ Decision must result in an action | Rational Action | Decision

④ Must Have optimal / best solution and clear ethical goal

PEAs (Model Representation for AI Agent)

1. P → Performance Measure
2. E → Environment
3. A → Actuators
4. S → Sensors

→ AI agents with these factors determine the performance, utility and success, it's behaviour.

Self-Driving Car

P → Safety, legal drive

E → road, road signs, vehicles

A → brake, signal, horn, steering

S → Camera, GPS, Sonar, Speedometer, etc.

Summary!

① Lower-Bias, Higher Variance and vice-versa which is called trade-off. In generalization lower-Bias and Higher variance is the Key trade-off visulalization.

② Unsupervised ML →

→ K-means do clustering based on the centroids, doesnot require labelled data, Cluster data into K

→ K-NN perform classification (for Regression), Classify new points based on nearest neighbours, find K nearest data points,

→ Classify Images, requires labelled data, local optimal

⑧ L1 vs L2:

L1 (Lasso):

→ L1 adds values to the absolute values of the weights to the loss function

→ can shrink weights to exactly zero, performs feature selection

→ best for the models that selects only the most important features from a large dataset

#L2 (Ridge):

Adds the square of the weights to the loss function

→ Not exactly zero, for smoothy- shrink weights, best for all features contribute to prediction

→ used when all features might have influnie on o/p, don't eliminate all features.