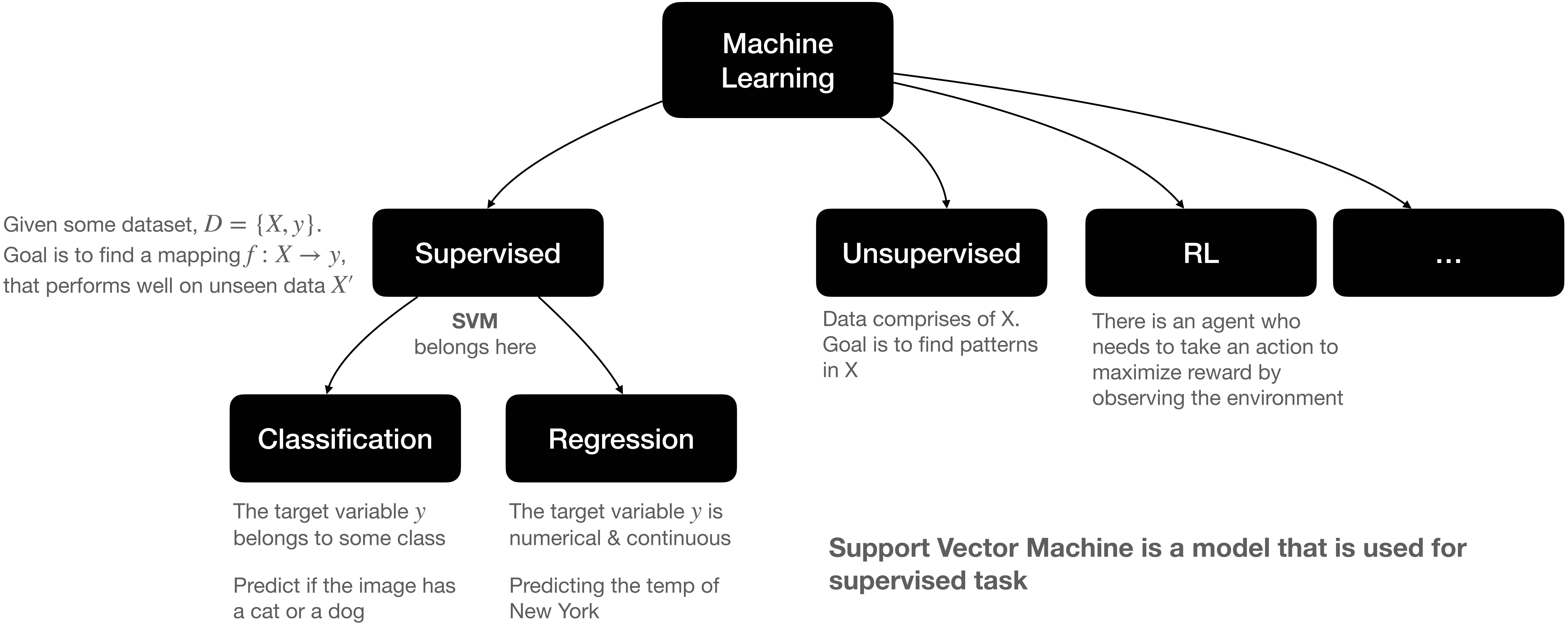
Support Vector Machine

Soham Dandapath

Where does SVM fit in Machine Learning landscape

Supervised tasks

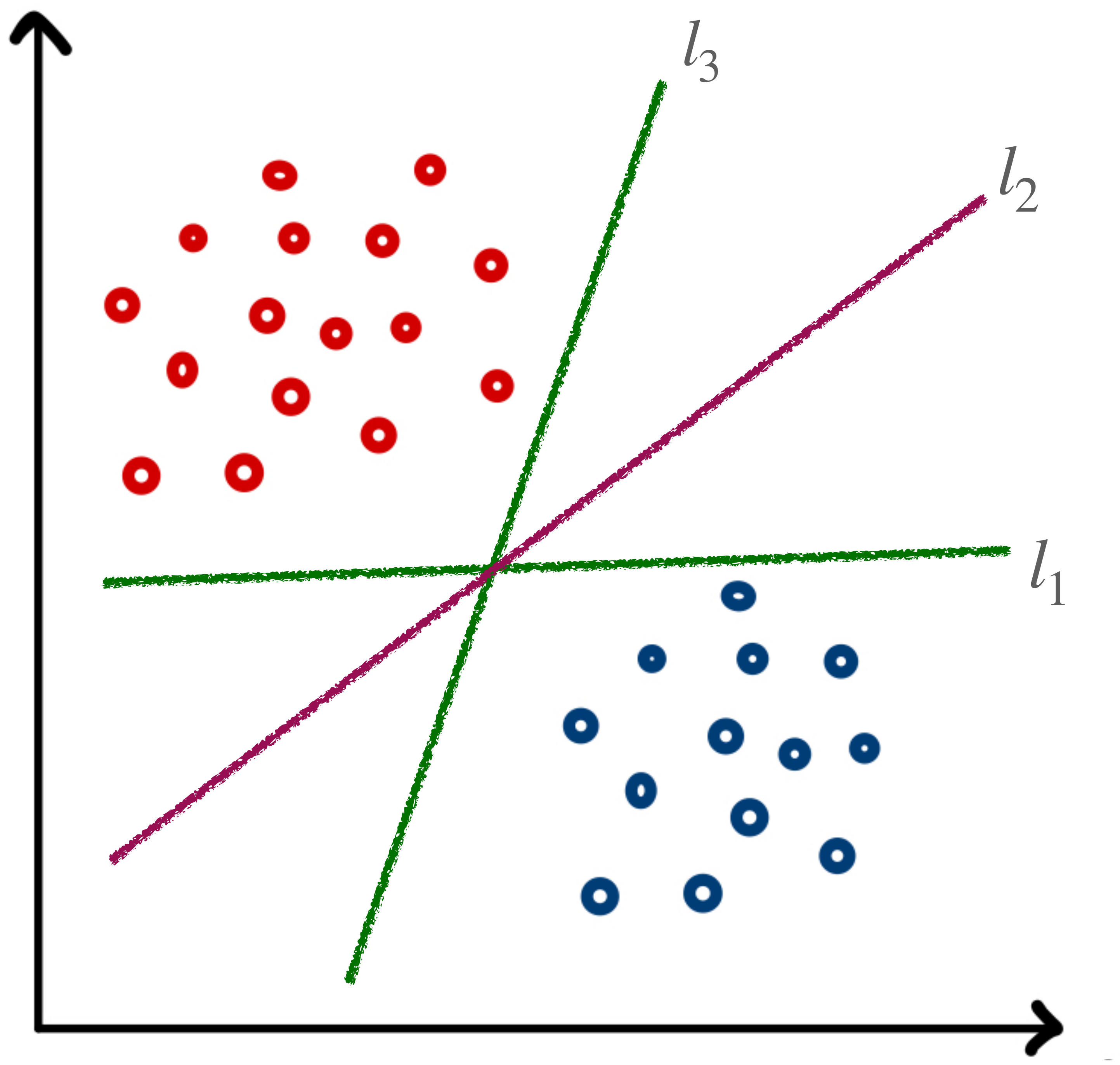


Support Vector Machine is a model that is used for supervised task

The **2 most important things** that I learnt through SVM was **problem formulation of an optimization problem** and **solving the optimization problem**

Motivation (Part of Problem Formulation)

What was wrong with previous Linear Classification models



What are the previous methods?

- Perceptron
- Logistic Regression

The above methods can yield any of the three l_1, l_2, l_3 classifiers.

For l_1 & l_3, any small perturbations to the point that lies near the decision boundary can make incorrect predictions.

Hence in some sense, **these are not stable**

So among l_1, l_2 & l_3, the preferred classifier was l_2 because it was the furthest away from the nearest point of any of the classes.

Or in other words, l_2 had the largest margin.

Margin: Distance between the hyperplane of the classifier to the nearest point

Problem Formulation

What properties of SVM is needed

So we want a classifier that has a decision boundary with the the largest margin.

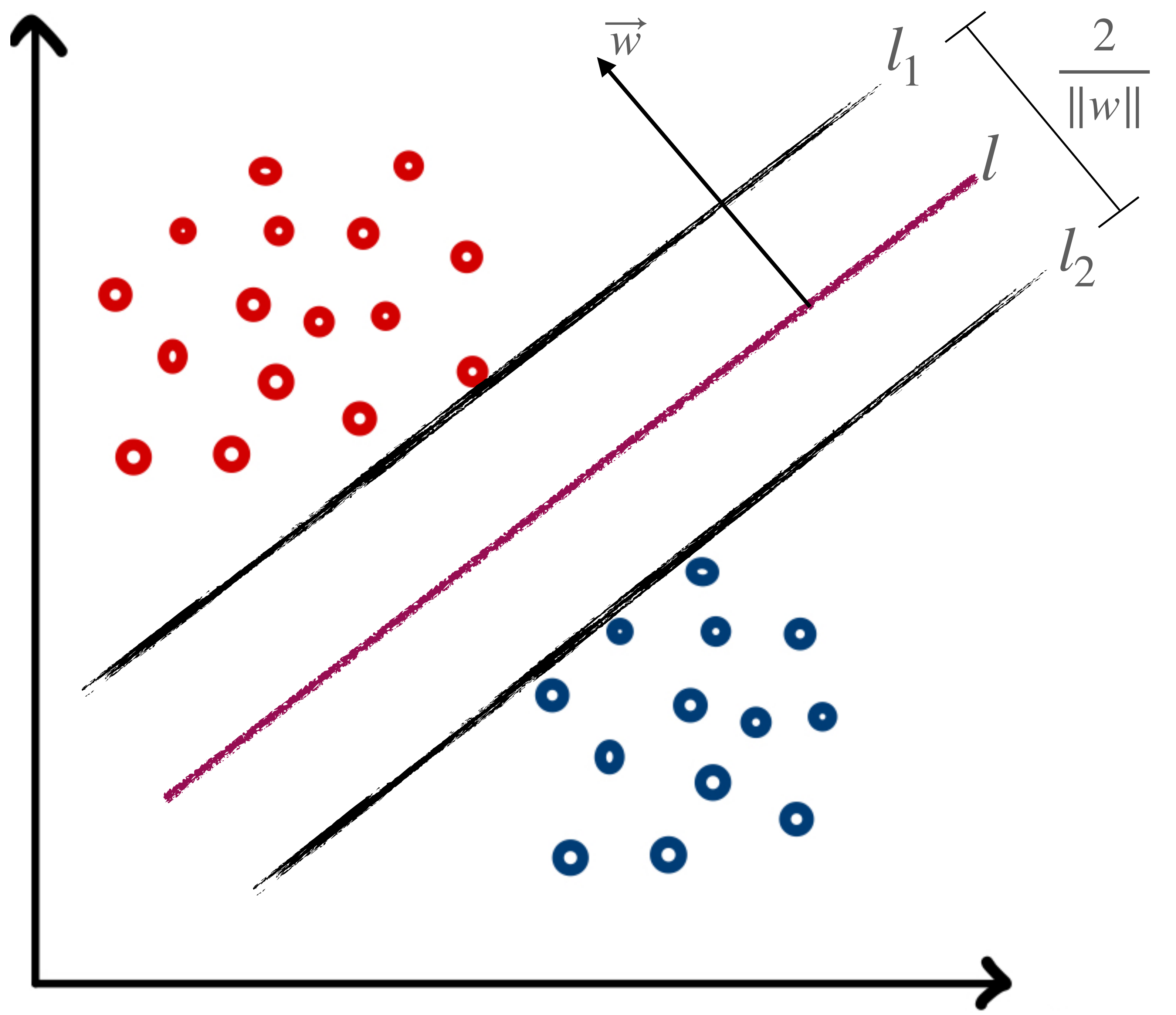
By how do we find the largest margin? We can instead formulate the problem in a different way.

<u>Find two parallel hyperplanes l_1, l_2 that correctly classifies all the points and maximize the distance between them.</u>

$$l_1 : w^T x - b = 1$$
$$l_2 : w^T x - b = -1$$

Note that l, l_1 & l_2 are all parallel to each other (w, the normal vector to the plane is the same for all)

The distance between l_1 & l_2, is given by $\frac{2}{\|w\|}$ which is exactly, half the margin of l.



$$l : w^T x - b = 0$$

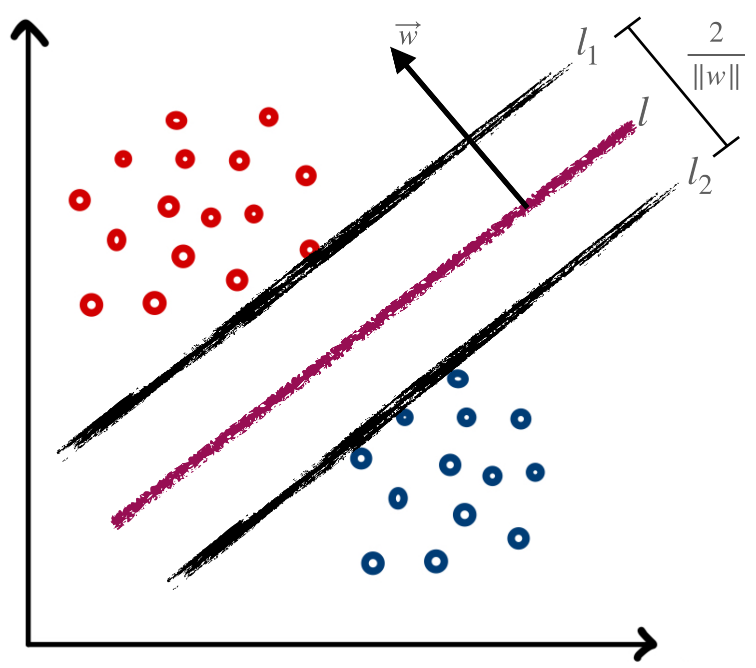
Recall that a classifier is $f : X \rightarrow y$

$$f(x) : sgn(w^T x - b)$$

Note: l_1 & l_2 barely touches the points that are lying on either side of l in order to correctly classify the points and maximize the distance between the two planes

Problem Formulation

Converting from statement to mathematical formulation



Find two parallel hyperplanes l_1, l_2 that correctly classifies all the points and maximize the distance between them.

Objective Function : $\max\limits_{w} \dfrac{2}{\|w\|_2}$ such that $\begin{array}{ll} w^T x_i - b \geq 1, & \text{if} \quad y_i = 1 \\ w^T x_i - b \leq -1, & \text{if} \quad y_i = -1 \end{array}$ All the points are correctly classified

Maximize the distance between the planes

$\sim y_i(w^T x_i - b) \geq 1$ where $i \in [N]$ **There are N constraints**

Converting to standard optimization problem

Primal SVM :

$$\boxed{\begin{array}{c} \min\limits_{w} \dfrac{\|w\|_2^2}{2} \\ \text{s.t} \quad y_i(w^T x_i - b) \geq 1 \end{array}}$$

Maximize the margin while correctly classifying all the points

Let’s talk about

Convex Optimization

Solution

How to solve constrained based optimization problem?

How do we usually solve a maxima-minima problem ?

We take the derivative of the objective function $\mathscr{L}$ w.r.t x and set it to 0. Here x is our parameter space.

However, the minima point we found through the above method may not satisfy our constraints or x^* may not be in the feasible region. **Feasible region is the space of parameter that satisfies the constraints.**

So there are two methods :

- **Projection Method :** Same as gradient descent, except after every update, you project the parameter back to the feasible space.
- **Lagrange Penalty Method :** Use of penalty term in the loss function

Standard Form :

$$\min_{x \in R^d} f(x)$$

$$\text{s.t} \quad g_i(x) \leq 0 \quad \forall i \in [N]$$

Lagrange Penalty Method

Incorporating a penalty term

Standard Form : $\min_{x \in R^d} f(x)$

s.t $\quad g_i(x) \leq 0 \quad \forall i \in [N]$

x^* **is the value of x that is a solution**

Consider a new function $L(\vec{x}, \vec{\lambda}) = f(x) + \sum_{i \in [N]} \lambda_i \cdot g_i(x)$ **and a new optimization problem** $\min_x \max_{\lambda_i \geq 0} L(\vec{x}, \vec{\lambda})$

Lets look at some characteristic of this new function

In the **feasible region** i.e. when x satisfies the constraint $L(x, \lambda) \leq f(x)$ $\quad \min_x \max_{\lambda_i \geq 0} L(\vec{x}, \vec{\lambda}) = p^*$

In the **infeasible region, $\exists$ a set of λ s.t** $L(x, \lambda) \geq f(x)$ $\quad \min_x \max_{\lambda_i \geq 0} L(\vec{x}, \vec{\lambda}) = \infty$

$$p^* = \min_x \max_{\lambda_i \geq 0} L(x, \lambda)$$

This is unconstrained in x with simpler constraints in λ.
So projected method can be easily applied.

Lagrange Penalty Method

The dual's introduction

$$L(\vec{x}, \vec{\lambda}) = f(x) + \sum_{i \in [N]} \lambda_i \cdot g_i(x)$$

Standard Form : $\min_{x \in R^d} f(x)$

s.t $\quad g_i(x) \leq 0 \quad \forall i \in [N]$

For all $\lambda_i \geq 0$, x^* be parameter solution to feasible region

$$\min_x L(x, \lambda) \leq L(x^*, \lambda) \leq f(x^*) = p^* \quad d^* \leq p^*$$

Duality gap : $\quad p^* - d^*$

Primal $\quad p^* = \min_x \max_{\lambda_i \geq 0} L(x, \lambda)$

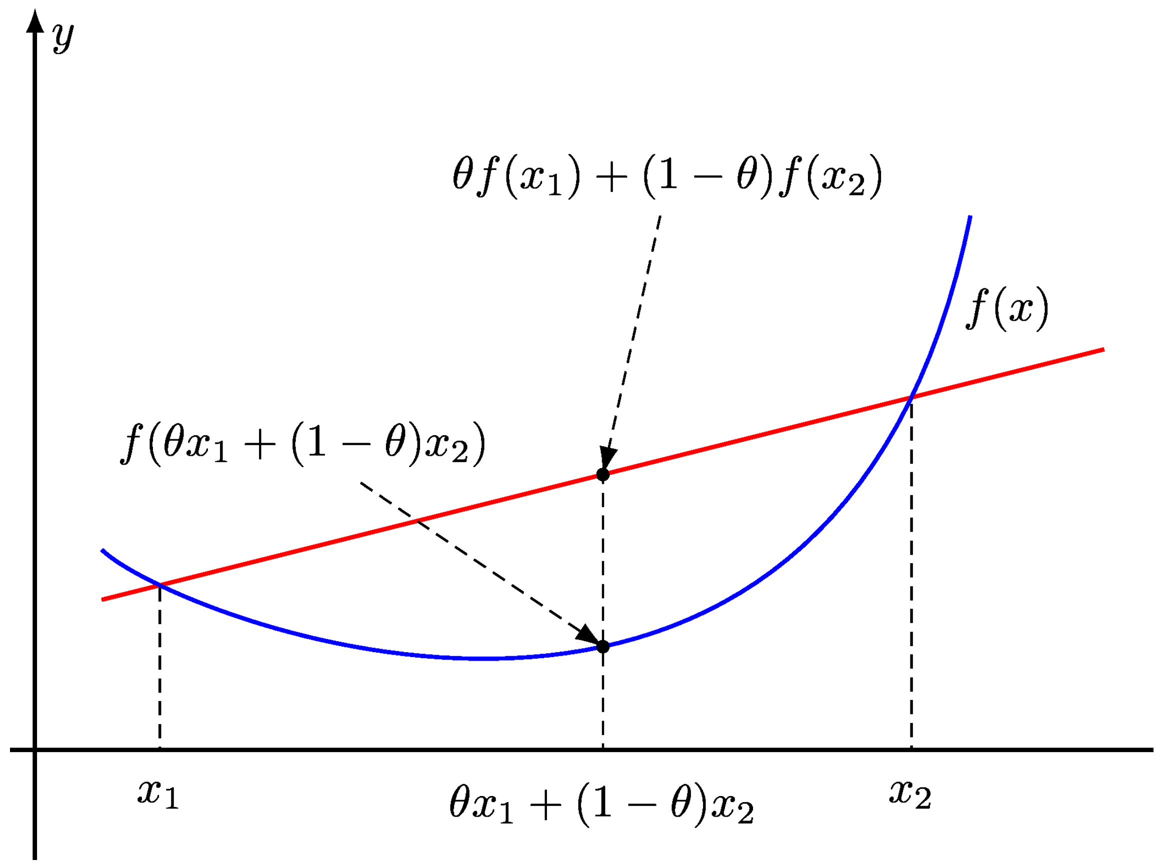
Dual $\quad d^* = \max_{\lambda_i \geq 0} \min_x L(x, \lambda)$

If Slater's condition is satisfied, the Strong Lagrangian Duality holds $\Longrightarrow$ $d^* = p^*$

1. The objective function, f(x) must be convex
2. $\exists x$ in feasible space such that :

   $g_i(x) \leq 0 \forall i$, or

   $g_i(x) \leq 0$, whenever $g_i(x)$ is affine



Coming back to SVM

SVMs with penalty

Constructing the Lagrange Function

$$\min_{w} \frac{\|w\|_2^2}{2}$$

s.t $\quad y_i(w^T x_i - b) \geq 1$

$$\mathcal{L}(w, b, \lambda) = \frac{\|w\|^2}{2} + \sum_{i \in [N]} \lambda_i (1 - y_i (w^T \cdot x - b))$$

$$p^* = \min_{w,b} \max_{\lambda \geq 0} \frac{\|w\|^2}{2} + \sum_{i \in [N]} \lambda_i (1 - y_i (w^T \cdot x - b))$$

$$d^* = \max_{\lambda \geq 0} \min_{w,b} \frac{\|w\|^2}{2} + \sum_{i \in [N]} \lambda_i (1 - y_i (w^T \cdot x - b))$$

With the dual, we can find the minima using the gradient method as if there is no constrains in the function

$$\frac{\partial L}{\partial w} = w - \sum_i \lambda_i y_i x_i = 0$$

$$w = \sum_i \lambda_i y_i x_i$$

New Constraint

$$\frac{\partial L}{\partial b} = -\sum_i \lambda_i y_i = 0$$

$$\sum_i \lambda_i y_i = 0$$

Whenever $\lambda \geq 0$, the corresponding x_i forms the support vector essentially determining the parameters of the model

Solving the Lagrangian problem

Putting the w, b in the Lagrange function

$$\mathscr{L}(w,b,\lambda) = \frac{\|w\|^2}{2} + \sum_{i\in[N]} \lambda_i(1 - y_i(w^T \cdot x - b)) \qquad w = \sum_i \lambda_i y_i x_i \qquad \sum_i \lambda_i y_i = 0$$

$$= \frac{1}{2}w^T w + \sum_i \lambda_i - \sum_i \lambda_i y_i (w^T x_i)$$

$$= \frac{1}{2}w^T w + \sum_i \lambda_i - \sum_i \lambda_i y_i (w^T x_i)$$

$$= \frac{1}{2}(\sum_i \lambda_i y_i x_i)^T (\sum_j \lambda_j y_j x_j) + \sum_i \lambda_i - \sum_i y_i \lambda_i (\sum_j \lambda_j y_j x_j)^T x_i$$

$$= \frac{1}{2}\sum_{i,j} \lambda_i \lambda_j y_i y_j (x_i \cdot x_j) + \sum_i \lambda_i - \sum_{i,j} \lambda_i \lambda_j y_i y_j (x_i \cdot x_j)$$

$$= \sum_i \lambda_i - \frac{1}{2}\sum_{i,j} \lambda_i \lambda_j y_i y_j (x_i \cdot x_j)$$

Dual SVM :

$$\boxed{\begin{array}{c} \max_{\lambda} \sum_i \lambda_i - \frac{1}{2}\sum_{i,j} \lambda_i \lambda_j y_i y_j (x_i \cdot x_j) \\ \text{s.t} \sum_i \lambda_i y_i = 0 \quad \text{s.t} \quad \lambda \geq 0 \end{array}}$$

Why the dual is important

Kernelization allows to incorporate non-linear boundary

Dual SVM :

$$\max_{\lambda} \sum_{i} \lambda_i - \frac{1}{2} \sum_{i,j} \lambda_i \lambda_j y_i y_j (x_i \cdot x_j)$$

$$\text{s.t} \sum_{i} \lambda_i y_i = 0 \quad \text{s.t} \quad \lambda \geq 0$$

We want to transform x_i to a different higher dimensional space using the function $\phi(x)$.

The special characteristic of $\phi(x)$ is $\phi(x_i) \cdot \phi(x_j)$ does not need explicit computation of the higher dimensional vector. We can instead use Kernel function that does it efficiently.

$$K(x_i, x_j) = \phi(x_i) \cdot \phi(x_j)$$

Kernelized Dual SVM :

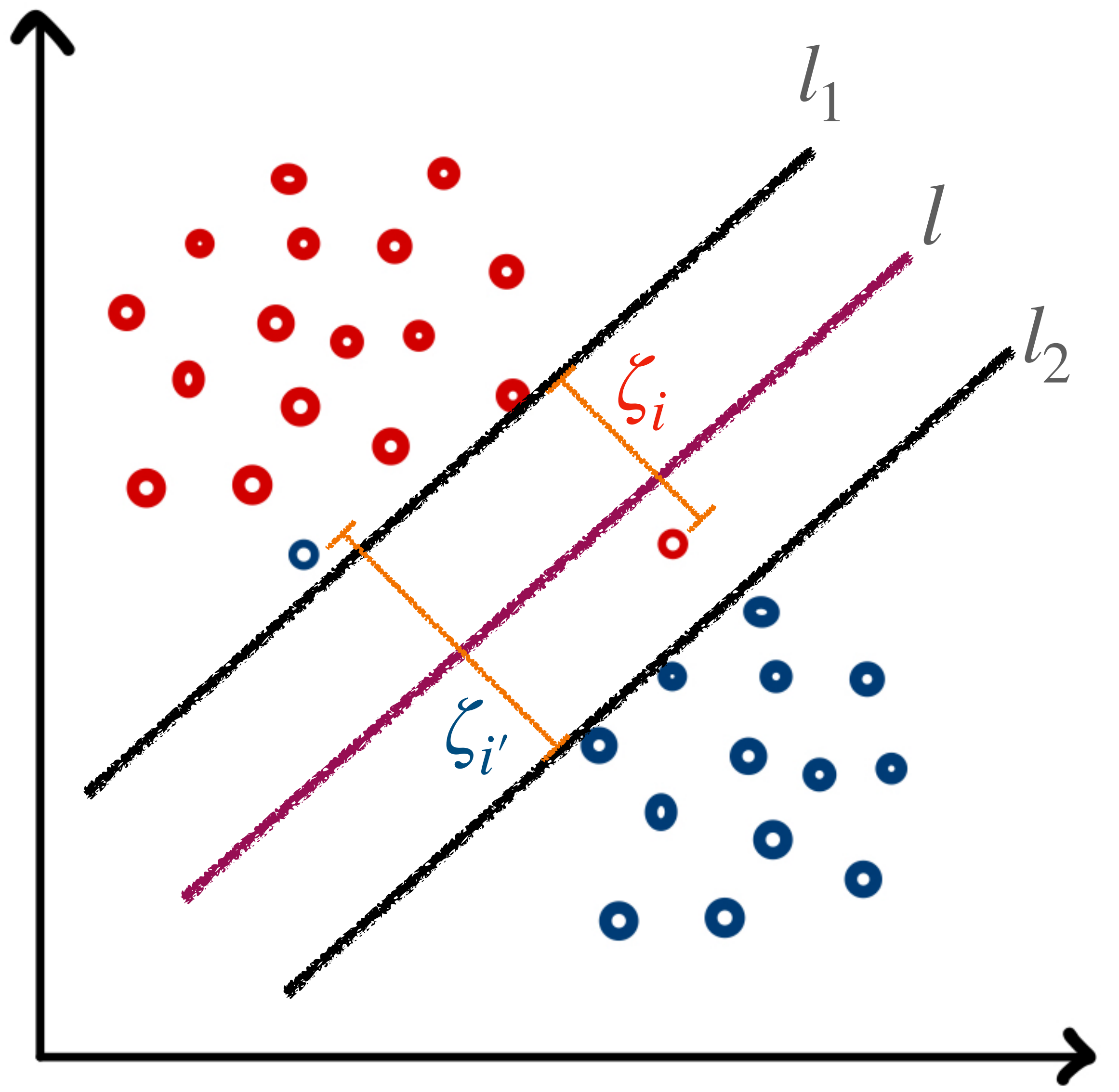
$$\max_{\lambda} \sum_{i} \lambda_i - \frac{1}{2} \sum_{i,j} \lambda_i \lambda_j y_i y_j K(x_i, x_j)$$

$$\text{s.t} \sum_{i} \lambda_i y_i = 0 \quad \text{s.t} \quad \lambda \geq 0$$

This allows for non-linear decision boundary

SVM with non-separable cases

Adding the slack variable



Primal SVM with Slack variable

$$\min_w \frac{\|w\|^2}{2} + C\sum_i \zeta_i$$

C is a penalty weight defined by user

$$\text{s.t} \quad y_i(w \cdot x_i - b) \geq 1 - \zeta_i$$

$$\zeta_i \geq 0$$

The maths is a bit involved with book keeping to derive the dual form but the solution comes out to be pretty simple

Kernelized Dual SVM with slack :

$$\boxed{\begin{array}{l} \max\limits_\lambda \sum\limits_i \lambda_i - \frac{1}{2}\sum\limits_{i,j} \lambda_i \lambda_j y_i y_j K(x_i, x_j) \\ \text{s.t} \sum\limits_i \lambda_i y_i = 0 \quad \text{s.t} \quad 0 \leq \lambda \leq C \end{array}}$$

Without an upper bound on λ, misclassification penalty reaches ∞ however, with an upper bound, now it reaches to C

Hence misclassification is allowed.

Connecting it to the Hinge loss

How is the code written?

Primal SVM with Slack variable

$$\min_{w,\zeta} \frac{\|w\|^2}{2} + C \sum_i \zeta_i$$

s.t $\quad y_i(w \cdot x_i - b) \geq 1 - \zeta_i$

$$\zeta_i \geq 0$$

Let's take a closer look at the constraints

$$y_i(w^T x_i - b) \geq 1 - \zeta_i$$

$$\zeta_i \geq 1 - y_i(w^T x_i - b)$$

$$\zeta_i \geq 0$$

$$\zeta_i = \max\left(0, 1 - y_i(w^T x_i - b)\right)$$

Hinge Loss

$$\min_{w,b} \frac{\|w\|^2}{2} + C \sum_i \max(0, 1 - y_i(w^T x_i - b))$$

Support Vector Regression

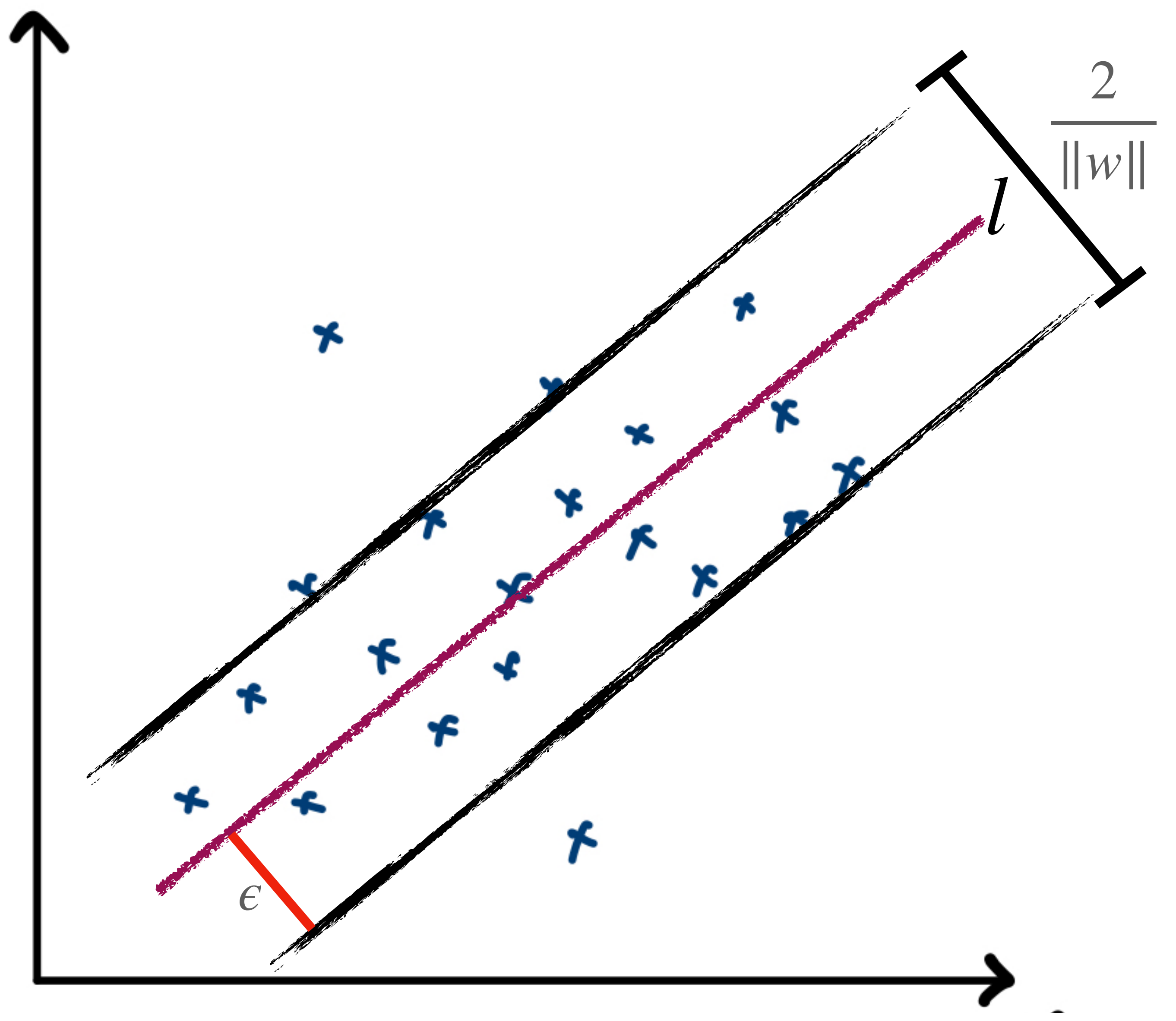
How does support vector work with regression

Primal SVR

$$\min_w \frac{\|w\|^2}{2}$$

s.t $\quad |y_n - (w^T x - b)| \leq \epsilon$

ϵ is defined by user



Intuition: The residual error must be within a range ϵ

2 slack variables, $\zeta, \zeta*$ are introduce in the regression case for each variable

$$\min_w \frac{\|w\|^2}{2} + c\sum_i (\zeta_i + \zeta_i^*)$$

s.t $\quad (y - (w^T x - b)) \leq \epsilon + \zeta$

$$((w^T x - b) - y) \leq \epsilon + \zeta*$$

$$\zeta, \zeta* \geq 0$$