24/07/24

For the exam remember to know what the effect of the hyperparameters etc.

Introduction

Machine learning in the literature can only be used in tabular form, but not everything can be represented in this way. This because sometimes you have relation between the elements. For example in text you can see this fenomenon. What is able to operate on this problems is **deep learning**.

Generative AI is able to create new data from previous knowledge

- - - -

Understanding is about **causality**

Example:

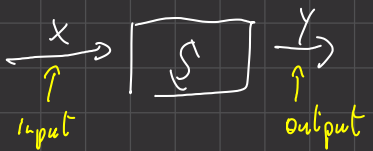
It's hot then people go to the beach and eat ice cream.
Then I hide this fenomen and give this info to a machine it will tell me that there is a correlation between hotness and ice cream consumption.

The data is very important because a machine learns correlation from it.

Machine learning

What we want to do needs to be agnostic. So we must be able to learn independently from the source.

We have a system:



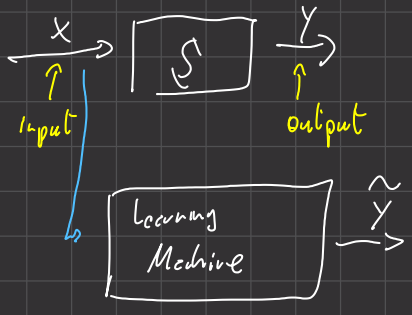
I don't know what's inside S.

Input in machine learning can be easily measured.

The output is not easy to measure or I don't know it now. (For example a plant will breed in one month)

The objective of ML is to predict the output.

Future can be predicted when it's similar to the past.



26/09/24

We want to learn from a completely unknown system S.

Inputs (x) are easy to measure, outputs (y) no.

Usually there is no causality between inputs and outputs. It may happen that y generates x

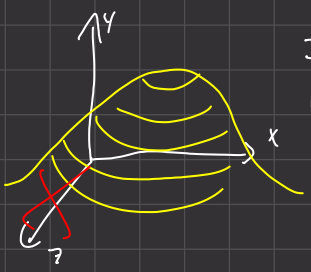
We want to create $\hat{y}$ as close as possible to y.

Even with an oracle there is no possibility knowing x to know y. This because of noise, the measurement will be wrong even with the best tool in the world.

The only way to have a very exact measure is taking more than one measurement. This noise not only takes into account the measurement noise, but also the numerical error due to the fact that we need to save data on digital hardware.

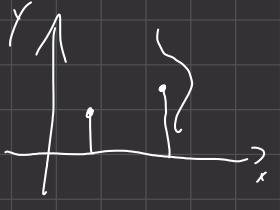
The last "source of noise" could be that we missed a variable and so we are not able to compute correctly the output

Example:



If I omit a dimension then I will lose relevant information. For the same x it seems that there could be two different results.

We cannot distinguish between all these sources of noise:



Two sources of noise:

- Measure
- Partial Knowledge of x

> Input x: it can be $\mathbb{R}$ (number), $\mathbb{R}^d$ (vector), $\{G, Y, B\}$ (category)
> Output y) (graph)

We like real numbers because we have many different ways to operate on them. Other types of functions they could be more challenging.

Note that by doing: $\{Y, G, B\} \to \{1, 2, 3\}$ you are expressing that yellow is closer to green than Blue

the way of transforming them is to transform them in a vector like this:

Y -> [1 | 0 | 0]
G -> [0 | 1 | 0]
B -> [0 | 0 | 1]

this approach is called One Hot coding

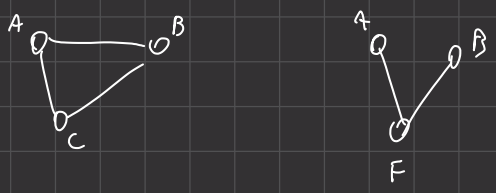
Here the notion of distance is not in play. they are all distant in the same way.

> What if we have ordinal variables?

$x \in \{1, 2, 3, 4\}$ $\quad \hookrightarrow \mathbb{R}$

We can leave it like they are, no need for one hot coding because there is order.

> Graphs:



For them I can use an adjacency matrix

	A	B	C
A	0	1	1
B	1	0	1
C	1	1	0

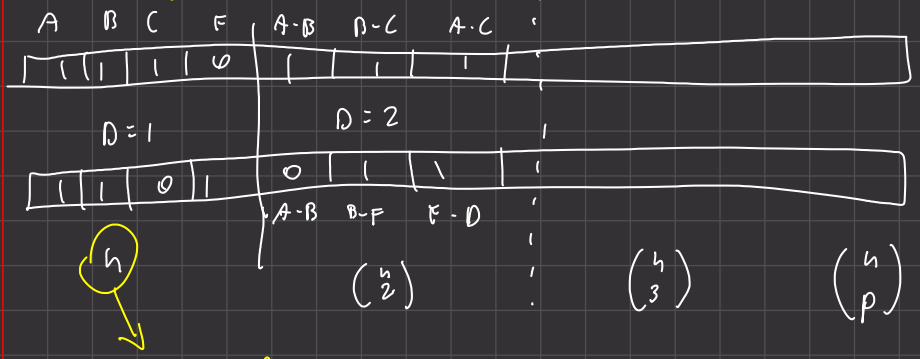
	A	B	F
A	0	0	1
B	0	0	1
F	1	1	0

The problem here is that there is no way of defining distance between matrices easily.

Distance is THE parameter needed for machine learning.

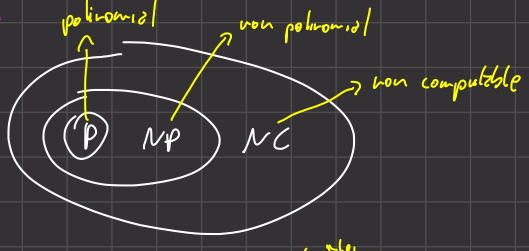
One trick could be to define vectors.

Structures of dimension 1



number of nodes

Reminder:



Note O → This is better.

$O(\ln(n))$ $O(n^7)$

$10^{70} \ln(n)$ $10^{-30} n^7$ ← This is better.

Machine learning is almost polynomial problems. Deep learning is non polynomial } approximation of non computable problem of learning from data.

A sequence is a unidirected graph.

Whatever we have in input we can map in a series of real numbers.

- - - - - - .

y can be the same as what x can be.

It can be transformed in a vector of elements.

L.M => $y = f(x)$ $y_i = f_i(x)$ Element by element.

Learning machine

$y \in \mathbb{R}^p$ $x \in \mathbb{R}^d$

The most general problem is:

$$y = f(x)$$

$x \in \mathbb{R}^d$, $y \in \mathbb{R}$

The good approximation is called loss function that usually returns a real number.

Taylor and Fourier are an approximation for example, but in Machine learning everything is different.

First of all we should define $f(x)$

There is a theorem called no free lunch that tells us which the pecurses of ML if we choose the optimal for one problem will be the worse for another.

$$f(x) = \sum_{i=0}^{p} c_i x^i$$ with enough variables I can fit any function.

loss function — distance — I want to find these.

$$l = (f(x), y) = (y - f(x))^2$$

$D_n = \{(x_1, y_1) \dots (x_n, y_n)\}$ approximated with min square error.

Let's consider p (grade of the polinomial)

Finding it for a computer is hard but finding c is easy

The actual way of computing goodness of c_i is:

$$\min_c \frac{1}{n} \sum_{i=1}^{n} (y_i - f(x_i))^2 = \frac{1}{n} \sum_{i=1}^{n} \left(y_i - \sum_{j=0}^{p} c_j x_i^j\right) = \|Xc - y\|^2$$

$$c = \begin{bmatrix} c_0 \\ \vdots \\ c_p \end{bmatrix} \in \mathbb{R}^{p+1} \qquad y = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \in \mathbb{R}^n \qquad X = \begin{bmatrix} x_1^0 & \dots & x_1^p \\ \vdots & & \\ x_n^0 & \dots & x_n^p \end{bmatrix} \in \mathbb{R}^{n \times (p+1)}$$

$$\left[\begin{array}{|c|} \hline \\ \\ \\ \\ \hline \end{array} \begin{array}{|c|} \hline \\ \\ \\ \\ \hline \end{array} - \begin{array}{|c|} \hline \\ \\ \\ \\ \hline \end{array} \right]^2 = \left\| \quad \right\|^2$$

X c y

$\mathbb{R}^{n \times (p+1)}$ $\mathbb{R}^{(p+1)}$ $\mathbb{R}^n$

$\mathbb{R}^{n \times 1}$

$\min \|Xc - y\|^2$ This I want to do

↳ paraboloid

To find the minimum I will need to find the gradient of the function.

$\min \|Xc - y\|^2 =$

$\nabla_c (c^T X^T X c - 2 c^T X^T y + y^T y) = \emptyset$

$2 X^T X c - 2 X^T y = \emptyset$

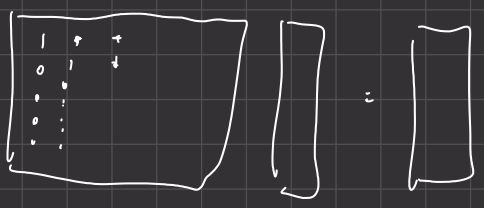
$X^T X c = X^T y \qquad Ax = b$

$c = (X^T X)^{\#} X^T y \longrightarrow$ pseudoinverse.

$O(n^2 \cdot n^3)$ → computational cost.

Remark:

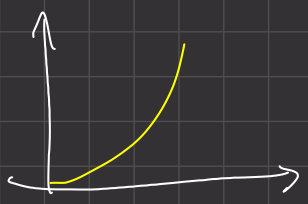
$Ax = b$



To do Gauss Jordan $O(n^2)$

$x = A^{\#} b : O(n^{2.3})$

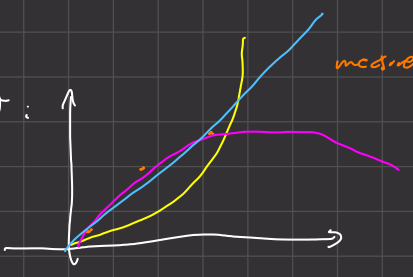
Suppose now that our data has been generated by something like $y = x^2$



On here I fit my function. $\hat{y} = f(x) = \sum_{i=0}^{p} c_i x^i$

Even though it may seem that the best $p = 2$ it can be either $\{0, 1, 2\}$.

It may happen due to noise that:

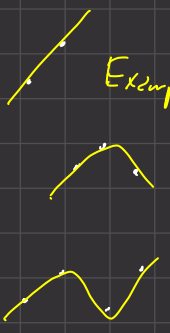


the line fits better the original data than the predicted.

Suppose that $p = n-1$

$$\min_{c} \| X c - y \|^2 = 0 \quad \forall D_n$$

Examples.



this means that minimizing over data is not smart otherwise we would minimize over noise.

remark on statistics.

$x \to x_1, \dots, x_n$ $\qquad \mu = \mathbb{E}_x\{x\} = \int_D x\, p(x)\, dx$

independent and identical distributed.

μ can be estimated with empirical average $= \frac{1}{n} \sum_{i=1}^{n} x_i$

$$\mathbb{E}_{\tilde{x}}\{\tilde{x}\} = \mathbb{E}_{x_1, x_n}\left\{ \frac{1}{n} \sum_{i=1}^{n} x_i \right\} = \frac{1}{n} \sum_{i=1}^{n} \mathbb{E}_{x_i}\{x_i\} = \mu$$

This is not enough to get the average, we need.

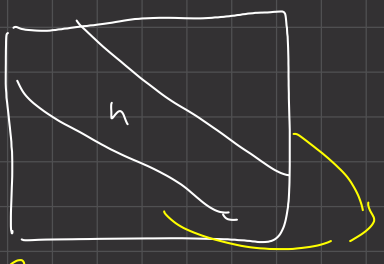
- Estimator
- Accuracy
- Confidence.

We need the variance:

$\sigma^2 = \mathbb{E}_x\{(x-\mu)^2\} =$ the variance is important because we can prove the law of large number so the larger the bolic the more my estimator $\bar{x}$ will be close to μ

$= \mathbb{E}_x\{x^2\} - 2\mathbb{E}_x\{x\mu\} + \mathbb{E}_x[\mu^2] = \mathbb{E}_x\{x^2\} - \mu^2$

$\sigma_{\bar{x}}^2 = \mathbb{E}_{\bar{x}}\{\bar{x}^2\} - \left(\mathbb{E}_{\bar{x}}\{\bar{x}\}\right)^2 = \mathbb{E}_{x_1 \dots x_n}\left\{\frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=i}^{m} x_i x_j\right\} - \mu^2 =$



$i = j$ → x_i^2 ; $i \neq j$ → $x_i x_j$

$= \frac{1}{n}\sum_{x=1}^{n}\sum_{\substack{j=1 \\ i \neq j}}^{n} \mathbb{E}_{x_i x_j}(x_i x_j) + \frac{1}{n^2}\sum_{i=1}^{n}\mathbb{E}_i(x_i^2) - \mu^2$

TODO ???

$\mathbb{E}_{x_i x_j}(x_i x_j) \to \mu^2$; $\mathbb{E}_i(x_i^2) \to \mathbb{E}_x(x^2)$

$= \frac{n^2-n}{n^2}\mu^2 + \frac{n}{n^2}\mathbb{E}_x\{x^2\} - \mu^2 = \frac{n^2-n-n^2}{n^2}\mu^2 + \frac{1}{n}\mathbb{E}_x\{x^2\} =$

$= \frac{1}{n}\left(\mathbb{E}_x\{x^2\} - \mu^2\right) = \frac{1}{n}\sigma_x^2$ Proof of Large number theorem.

The more data like more I will follow the data that I want to predict.

> Confidence

$\sigma_{\bar{x}}^2 = \int_{-\infty}^{+\infty}(\bar{x}-\mu)^2 p(\bar{x})\,d\bar{x} \geq \int_{|\bar{x}-\mu|>\varepsilon}(\bar{x}-\mu)^2 p(\bar{x})\,d\bar{x} \geq$

this is because positive function.

$\geq \varepsilon^2 \int_{|\bar{x}-\mu|\geq\varepsilon} p(\bar{x})\,d\bar{x} = \varepsilon^2 p\{|\bar{x}-\mu| \geq \varepsilon\}$

I have proved that:

$$P\{|\bar{x}-\mu| \geq \varepsilon\} \leq \frac{\sigma_{\bar{x}}^2}{\varepsilon^2} = \frac{\sigma_x^2}{n\varepsilon^2} \leq \frac{1}{n\varepsilon^2} = \delta \qquad \varepsilon = \sqrt{\frac{1}{n\delta}}$$

Variance is less than the size of the domain

$\frac{1}{n\varepsilon^2}$ confidence

TODO what's what?

$$P\{|\bar{x}-\mu| \geq \varepsilon\} \leq \frac{1}{n\varepsilon^2} = \delta \qquad \varepsilon = \sqrt{\frac{1}{n\delta}}$$

$$|x-\mu| \underset{(1-\delta)}{\leq} \sqrt{\frac{1}{n\delta}}$$

$$\mu \underset{(1-\delta)}{\leq} \bar{x} + \sqrt{\frac{1}{n\delta}}$$

$x \in \{0,1\}$

$x_1 \dots x_n \Rightarrow$ i.i.d.

If I want 0.95 accuracy δ is 0.01

$x \in \mathbb{R}$

$y \in \mathbb{R}$ $\qquad D_n = \{(x_1,y_1) \dots (x_n,y_n)\}$

$$f(x) = \sum_{i=0}^{p} c_i x^i$$

$$l = (f(x),y) = (y - f(x))^2$$

$$\min_c \frac{1}{n} \sum_{i=1}^{n} (y_i - f(x_i))^2 \longleftrightarrow \mathbb{E}_{x,y} (y - f(x))^2$$

$\frac{1}{n}\sum$: this is the learning part. $\qquad (y_i - f(x_i))^2$: x_i $\qquad$ $\bar{x}$

$(y - f(x))^2$: x $\qquad$ μ

$l \in [0,1]$

independent and identically distributed

allows the large number theorem $\qquad$ is time independent

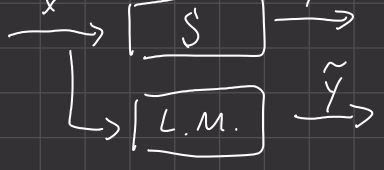
20:32 ??? exist if I remove the learning part.

Once I choose the minimum I lose the independence. (Difference between statistics and ML)

I.I.D must be there in order to do M.L.

7/10/24

RECAP:



$\tilde{y} \approx y$

$Dataset = \{(x_1, y_1) \dots (x_n, y_n)\}$ $x, y \in \mathbb{R}$

Only one input and output.

$\hat{y} = f(x) = \sum_{i=1}^{n} c_i x^i$ → For a polynome of high enough degree we can build any function.

↳ I need a function to map y from x.

There is no optimal choice because of the no-free-launch.

Loss function

$$l(f(x), y) = (f(x) - y)^2$$

We want the coefficient that minimizes the avg error on my data:

$$\min_c \frac{1}{n} \sum_{i=1}^{n} \left(y_i - \sum_{j=0}^{p} c_j x_i^j\right) \rightarrow \min_{\underline{c}} \| X\underline{c} - \underline{y} \|^2$$

(X: $n \times (p+1)$, $\underline{c}$: $(p+1) \times 1$, $\underline{y}$: $n \times 1$)

$\bar{x}$ ↳ empirical avg

Just one l. system $(X'X)\underline{c} = X'\underline{y}$ $O(p^2)$

Solve all l. systems. $c = (X'X)^{*} X'\underline{y}$ $O(p^3)$

Remember the problem with overfitting

Having a distribution. (statistics)

$x_1 \dots x_n$ i.i.d.

$x \in \{0, 1\}$

$\mu = \mathbb{E}_x\{x\}$ ↳ average

$\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i$ ↳ empirical average

$$\mu \leq \bar{x} + \sqrt{\frac{1}{n\delta}}$$

$(1-\delta)$

The more measures that I do the better the avg will be.

upper bound, so I want the worst case

What I really want to minimize is:

$$\mathbb{E}_{(x,y)} \underbrace{l(f(x), y)}_{x}$$

To have iid losses I need iid data. Data iid means that the system is not changing in time.

By doing the mining the errors won't be independent anymore and so we cannot use statistics.

We now need to bring back statistics.

We have selected a function $f \in F$ based on a dataset: $D_n = \{(x_1, y_1) \dots (x_n, y_n)\}$ → independent ← i.i.d.

Since they did not see each other (8.3?) they are independent.

Note, the independence is between the way of fitting and the dataset.

We suppose finite functions even though we have real coefficients so they could be infinite.

The functions are selected independently from the data. N.B.

I can put the independency hyp on the way because one of them will fit the data probably.

$$P\left\{\left|\underbrace{\mathbb{E}_{(x,y)} \ell(f_1(x), y)}_{\text{expected value}} - \underbrace{\frac{1}{n}\sum_{i=1}^{n} \left(f_1(x_i) - y_i\right)}_{\text{Empirical error}}\right| \geq \varepsilon\right\} \leq \frac{1}{n\varepsilon^2}$$

The errors are iid since I apply something independent to iid data

The expected error of each function is close to the empirical error on the data because f's are independent from the data.

I don't know the price for the selection on all the functions, but I know the price for every single function. So I will need to consider all the functions, so I consider the case where everything will go wrong.

To contemplate all the bad cases I will compute:

$$\frac{n_{\text{functions}}}{n\varepsilon^2}$$

$\forall f^* \in F$ based on D_n (any f)

$$P\left\{\left|\mathbb{E}_{(x,y)} \ell(f^*(x), y) - \frac{1}{n}\sum_{i=1}^{n} \ell(f^*(x_i), y_i)\right| > \varepsilon\right\} \leq \frac{n_F}{n\varepsilon^2} = \delta$$

$$\underbrace{\mathbb{E}_{(x,y)} \ell(f^*(x), y)}_{\mu} \leq_{(1-\delta)} \frac{1}{n}\sum_{i=1}^{n} \ell(f^*_{(x_i)}, y_i) + \sqrt{\frac{n_F}{n\varepsilon^2}}$$

$$\leq_{(1-\delta)} \underbrace{\frac{1}{n}\sum_{i=1}^{n} \ell(f^*(x), y_i)}_{\bar{x}} + \sqrt{\frac{n_F - 1}{n\delta}} + \sqrt{\frac{1}{n\delta}}$$

the first error is bounded by a term that is based upon how much data I have.
Not only that but I am also learning and this is from this I learn

This formula tells also that one function could fit the model very good for both. (Chebyshev theorem). STATISTICAL LEARNING THEORY.

$$\min \| X\underline{c} - \underline{y} \|^2$$

$\underline{c}$ Remember that c depends on p

$$\int_0^1 \left(f''(x)\right)^2 dx$$ → This measures the complexity of a function.

↳ Interval of my data.

$$f(x) = \sum_{i=0}^{p} c_i x^i$$

$$f'(x) = \sum_{i=1}^{p} i c_i x^{(i-1)}$$

$$f''(x) = (f'(x))' = \sum_{i=1}^{p} i(i-1) c_i x_i^{(i-2)}$$

$$\int_0^1 \left(f''(x)\right)^2 dx = \int_0^1 \sum_{i=1}^{p}\sum_{j=1}^{p} c_i c_j (i-1)\, i(j-1)_j\, x^{i+j-4} dx$$

$$\int_0^1 x^{i+j-4} dx = \frac{x^{i+j-3}}{i+j-3}\Bigg|_0^1 = \frac{1}{i+j-3}$$

↳ this because before is linear operator.

$$\int_0^1 \left(f''(x)\right)^2 dx = \sum_{i=1}^{p}\sum_{j=1}^{p} c_i c_j \frac{ij(i-1)(j-1)}{i+j-3} = \underline{c}^t M \underline{c}$$

$$M = \begin{bmatrix} 0 & 0 & 0 & 0 & \cdots \\ 0 & 0 & 0 & 0 & \cdots \\ 0 & 0 & & & \\ 0 & 0 & & & \\ \vdots & \vdots & & & \end{bmatrix}$$

so At the end:

$$\min_{\underline{c}} \| X\underline{c} - \underline{y} \|^2 + \lambda\, \underline{c}^t M \underline{c}$$

this regulates OA and underfitting
If ∞ overfit, if ∞ underfit.

RIDGE REGRESSION

Going back M is symmetric and semidefinite positive ($M \geq 0$)

$$\min_{\underline{c}} \underbrace{\|X\underline{c} - \underline{y}\|^2}_{\text{Paraboloid}} + \underbrace{\lambda\, \underline{c}^T M \underline{c}}_{M \geq 0 \text{ and symmetric} \Rightarrow \text{Paraboloid}}$$

=> Paraboloid at the end so one solution so gradient $= \emptyset$

$$\nabla_{\underline{c}} \left(\underline{c}^T X^T X \underline{c} - 2\, \underline{c}^T X^T \underline{y} + \underline{y}^T \underline{y} + \lambda \underline{c}^T M \underline{c} \right) = \emptyset$$

$$2X^T X \underline{c} - 2X^T \underline{y} + \emptyset + 2\lambda M \underline{c} = \emptyset$$

$$O(p^2) \quad (X^T X + \lambda M)\, \underline{c} = X^T \underline{y}$$

$(p+1) \times (p+1)$

$$\underline{c} = (X^T X + \lambda M)^{-1} X^T \underline{y}$$

Depends on the data / does not depend on the data

$$(X^T X + \lambda M)$$

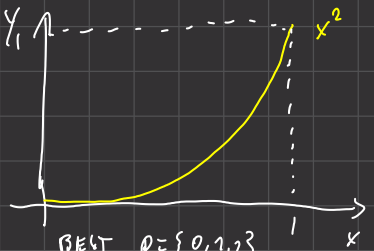
- $\lambda = \emptyset$ => min error
- $\lambda = \infty$ => min complexity

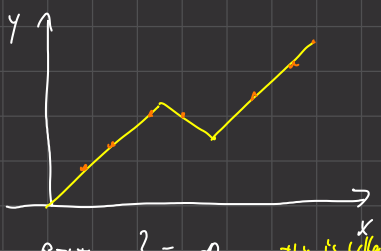
Suppose that M is an identity matrix:

$$X^T X + \lambda I$$

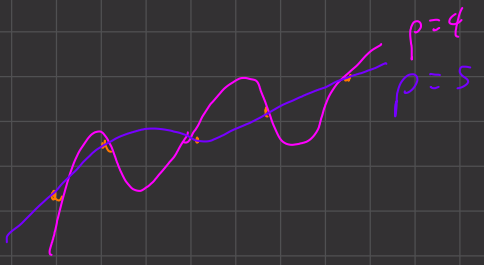
by adding this you are adding to eigenvalues an eigenvector.

By adding λI I force the matrix to react in a strict way not depending on the data but only on my lambda. This is a filter.
I reject small perturbations that "escape" only big ones.



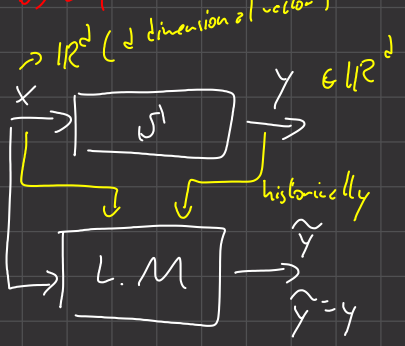


This is called overparametrization.



the higher degree one fits a simpler solution than the one of degree four. this is the idea of overgeneralization.

03/10/24



We considered $\begin{cases} x \in \mathbb{R} \\ y \in \mathbb{R} \end{cases}$

$$D_n = \{(x_1, y_1) \dots (x_n, y_n)\}$$

We need to define the tool to transform x into y.
There is no optimal way of doing that.

We have mathematical constituents but also computer constraints.

$$y = f(x) = \sum_{i=0}^{p} c_i x^i$$ (we used a polynom of degree p.)

We choosed the loss fuction:

$$l(f(x), y) = \underbrace{(y - f(x))^2}_{\hookrightarrow \text{distance}}$$

Once defined all those things we needed to defined the optimal value for this parameter.

Now we still select is the best conf. of param $\underline{c}$ knowing the params.

The best thing is to minimize the error on my data:

$$\min_{\underline{c}} \frac{1}{n} \sum_{i=1}^{n} (y_i - f(x_i))^2$$

The problem here is that it assumes that we know what we are doing. In some cases this approach does not work (degree of polinom equal at number of samples become error always $= 0$ which is impossible)

We want to be good on previously unseen data NOT historical data.
We used statistics and discovered that if the dataset is IID.

The more data the more info I have

↳ the system does not change in time

and choose f from a set of function independently from F.

$f \in F$ independently F

we know that the true error is bounded by the empirical error + something which goes like

(Error of the fuction on the data)

$$R(f) \underset{(1-\delta)}{\leq} \hat{R}(f) + \sqrt{\frac{1}{n\delta}} \qquad R(f) = \left| E_{(x,y)} \, l(f(x), y) \right|$$

↓ Expected value

$$\hat{R}(f) = \frac{1}{n} \sum_{i=1}^{n} l(f(x_i), y_i)$$

↓ empirical average.

empirical risk (error that I measure on my data)

For now we are just doing statistics, and if n (number of samples) goes to infinity then the true error is equal to the empirical error.
Since we are in statistics we have probability of getting something right at this probability is the confidence.

The higher the confidence the higher the risk, so I could be far away from the data

If instead f is independent from the dataset, but f is choosen in F using the dataset:

$$F \text{ ind. } D_n, \quad f \in F \text{ using } D_n$$

Let's say that we have two boys: one of data and one of functions the boy of functions is chosen before the one of data. Then I look at the data and I choose the best function between the ones that I have

(statistical learning theory)

Now we have

$$R(f) \underbrace{\underset{(1-\delta)}{\leq}}_{\text{probability}} \hat{R}(f) + \sqrt{\frac{h_F - 1}{n\delta}} + \sqrt{\frac{1}{n\delta}}$$

When I learn something from the data I don't just risk that my data are not enough or do not represent the population but I also risk that the function that I select works well on those data just by luck. (chance problem)

Ch. P. = If I check enough functions I will found one that works well on that data probabilistically speaking

The problem of just minimizing the error on my data is that I am not taking into account that I cannot choose whatever function I want, but I have to select a function that is simple, because it means that I trade off error on my data with simplicity. The simpler, the smaller the chance of over fitting

$$\min_{\underline{c}} \frac{1}{n} \sum_{i=1}^{n} \left(y_i - f(x_i) \right)^2 + \lambda \underline{c}' M \underline{c}$$

trade off error and simplicity

$\underline{c}' M \underline{c}$ → $C(f)$ (complexity of the function)

In matricial form:

$$\min_{\underline{c}} \| X\underline{c} - y \|^2 + c' M c$$

Solution: $(\underbrace{X'X}_{(p+1)(p+1)} + \lambda M)\, \underline{c} = X'y \quad O(p^2)$

When I measure my system we know that there are some noises that affect my measurement.

noise → measure

partial knowledge on X

In order to predict y we may need many x but we cannot have them for many reasons. (money, many approach)

The combination of not minimizing the error that compromising simplicity and accuracy and noise we discover that even if we know something about the system it is useless because the noise could make everything wrong.

Since we are adding the complexity term the logical rule that the more we increase p the more the model becomes complex is no more true
How the complexity is described by the second derivative.

— — — — — — —

- Function approximation works by defining a pointwise error and below a certain value. If we increase the elements for example in tailor the error decrease.

- In ML first of all we take a sample and we say that this is a data generating distribution which is IID.

The difference between sampling and Taylor is that in Taylor you "look" at all the function while sampling only the sampled part is relevant.

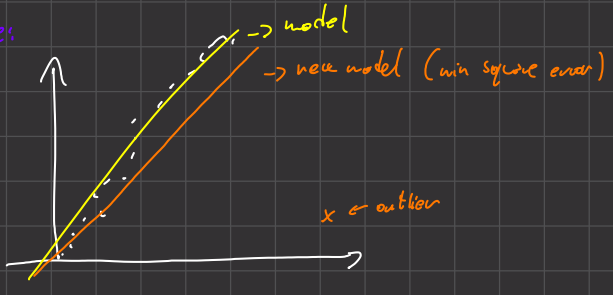
We always talk about expected error or empirical error because if x any

$$\mathbb{E}_{(x,y)} (\ell(f(x), y)) \quad \frac{1}{n} \sum_{i=1}^{n} \ell(f(x), y)$$

does not appear I don't care.

In ML I don't make the model work well point-wise, but I make it work well wrt the data generating distribution

Example:



In order to remove the problem, just remove the outlier. But that's a dog following its tail.

In order to mitigate the problem we could change the measure of the error:

$$(y - f)^2 \to |y - f|$$

What I would want is

Getting an upper bound for the error.

The good of the quadratic is that is differentiable and convex.
The modulus is convex but not diff.
The third one is non convex because there are concave parts

The problem of optimization with concave functions is that we don't have efficient solutions. With concave we have polynomial solutions. Non concave not polinomial!

In order to solve:

$$\min_{c} \|X_c - y\|^2 + \lambda c^t M c \quad \text{we need } O(p^2) \text{ lin. sistem.}$$

Since it's squared we cannot really increase p how much we want.

Remember that I want to find a minimum in a concave function:

GRADIENT DESCENT

$$\min_{x} f(x) \qquad x_0 = \emptyset$$

$$x_{i+1} = x_i - \gamma \nabla f(x)\Big|_{x_i}$$

↳ step size or LEARNING RATE

GRADIENT DESCENT

This approach does not depend on the number of parameters that I need to optimize. It does not depend on p

This algorithms depends on the space that it has to travel, but this depends on the number of iterations and on the learning rate.

There is a theorem that states that:
Given a function and using gradient descent if the gradient of the function has a norm that behaves like: $\|\nabla f(x) - \nabla f(y)\| \leq L \|x-y\|^2$ (the gradient is Lipschitz which means that it does not change too much (constant). I can also state that there are no empty points

If $\gamma \leq \frac{1}{L}$ I am sure that I will never go too far away from the point because if maximum it will be dividing the gradient by L.

The problem is that L is a worse case approach, so even if the function behaves very flat you will go slower if L very large.

$$|f(x^*) - f(x_u)| \leq \frac{\|x_0 - x^*\|^2}{2\,t\,K} \approx O\left(\frac{1}{u}\right)$$

x^*: optimal point; x_u: u^{th} point (iterations)

The accuracy goes down linearly.

Let's take Ridge regression as main space:

$$\min_{c_0 \dots c_p} \underbrace{\frac{1}{n} \sum_{i=1}^{n} \left(y_i - \sum_{j=0}^{p} c_j x_i^j \right)^2 + \lambda c' M c}$$

We start from $c = \emptyset$

$$c_{i+1} = c_i + \gamma \nabla_c (\quad)\Big|_{c_i}$$

$$= c_i + \gamma \left(\frac{1}{n} \sum_{i=1}^{n} \nabla_c \left(y_i - \sum_{j=1}^{p} c_j x_i^j \right)^2 + 2\lambda M_c \right)$$

this can be executed in parallel on different pcs, on different sets of data, then I can just sent a vector of dimension of c

Since these are just matrix operations you can use the GPU.

$$\min_{c_0 \dots c_p} \frac{1}{n} \sum_{i=1}^{n} \left(y_i - \sum_{j=0}^{p} c_j x_i^j \right)^2 + \lambda c' M c$$

This could be substituted by another with different complexity.
The simplest is the identity. $\Rightarrow c' I c = \|c\|^2$

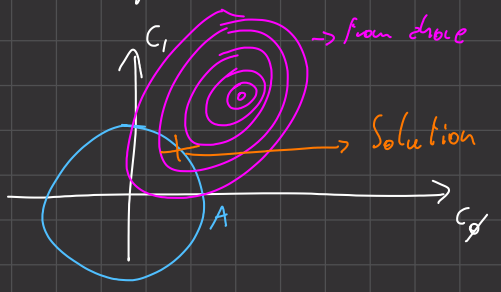
$$\min_{c} \|Xc - y\|^2 + \lambda \|c\|^2 \qquad = \qquad \min_{c} \|Xc - y\|^2 \quad \text{subject to } \|c\|^2 \le A$$

(L) (R)

they are the same

Suppose that the solution to (L) is c^* and its norm is $\|c^*\| = A$
Now I put A in (R) and the solution is c^* because c can go up to A and we know the complexity increases if we diminish the error.
In (R) we have to minimize the error so search for c with maximum complexity so it reaches A.
So if I find $\|c\| = A$ with smaller error than the one previously replaced it means that before I didn't minimize the error. (PROOF BY ABSURD)

The second problem can be represented in 2D:

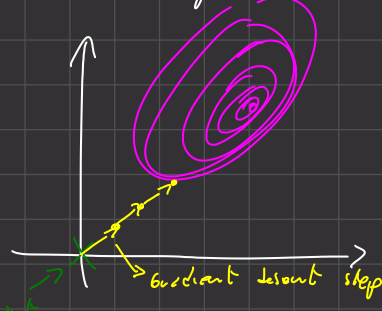


Now I remove completly the complexity, but in order to compute this I use the Gradient Descent

$$\min_{c} \|Xc - y\|^2$$

$$c_0 = 0$$

$$c_{i+1} = c_i - \gamma \nabla_c \|Xc - y\|^2 \Big|_{c_i}$$



Number of step = (K)

K is the same as A or λ. It's just another way of regulating the complexity of the result.

This is called early stopping.

This because the more steps I do the more I look at the data -> more step = more look
This is an indicator of how many times you look at the overall

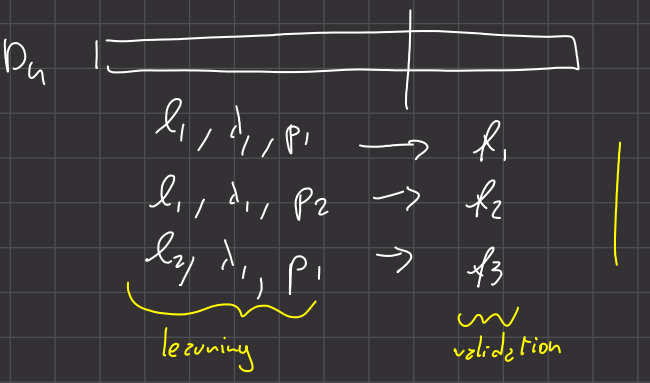
γ is the learning rate or how much time you spend on each page of the book

---

Now the problem is how to set the hyperparameters. Well we don't have a clue. We can for example with the other on $O(p^2)$

$D_n \rightarrow$ RidGlit $\rightarrow$ $\xi, \rho, \lambda \Rightarrow f \rightarrow$ I need to test this function.

So I take my dataset an I split it in two parts, and I can do this because it's IID



I am using always the same validation set because I don't have so much data.

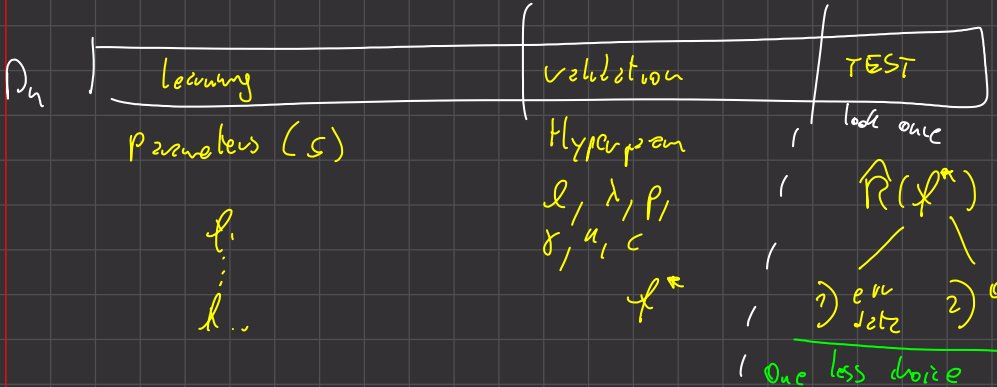
Both on the left and on the right I have a measure or error on the data, limited data, and catches to do.

On the left I will overfitting and on the right overvalidation.

Cross validation is what helps with overvalidation, is repeating validation on different splits.

- 70% of data for learning
- 30% for validation.

To see the performance of the model we do:



$$R(f) \underset{(1-\delta)}{\leq} \hat{R} + \sqrt{\ } + \sqrt{\ } \quad \vdots \quad R(f) \underset{(1-\delta)}{\leq} \hat{R}(f) + \sqrt{\frac{1}{n\delta}}$$

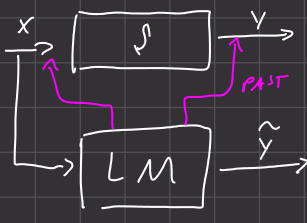
8/10/24

$x \in \mathbb{R}$

$y \in \mathbb{R}$

$D_n = \{(x_1, y_1) \dots (x_n, y_n)\}$ i i d

- the more data the better
- system not changing in time



$\hat{y} \neq y$ because of:
- measurement noise
- partial knowledge of x

$\hat{y} = f(x)$

$\ell(f(x), y)$

$R(f) = \mathbb{E}_{(x,y)}\, \ell(f(x), y)$ real error

$\hat{R}(f) = \frac{1}{n} \sum_{i=1}^{n} \ell(f(x_i), y_i)$ empirical error

$C(f)$ → complexity of the function.

Now we can build the algorithm:

$$\min_f \hat{R}(f) + \lambda\, C(f)$$

↳ this needs to be tuned to have the best trade-off between accuracy and complexity

This approach is theoretically grounded because when f is chosen inside a set where f is independent from a dataset is bounded by the empirical risk + a complexity term + something that depends on the number of samples + probability statement.

$$\min_f \hat{R}(f) + \lambda C(f) \quad \begin{cases} \text{SLT} \\ \text{start} \end{cases} \quad f \in F,\ \text{Find } D_n \quad R(f) \underset{(1-\delta)}{\leq} \hat{R}(f) + C(f) + \delta(n, \delta)$$

$$\text{start} \quad f \text{ and } D_n \quad R(f) \underset{(1-\delta)}{\leq} \hat{R}(f) + \delta(n, \delta)$$

f? min? ℓ? λ? C? ...

D_n

overfitting

overvalidation

LEARNING	VALIDATION	TEST
SLT choice $f \in F$	SLT choice	one statistics

Suppose now

$x \in \mathbb{R}^d$ $\quad D_n = \{(x_1, y_1) \dots (x_n, y_n)\}$

$y \in \mathbb{R}$

First thing to do is to write the function

$$\hat{y} = f(x) = c_{0,0} + \sum_{j=1}^{d} c_{1,j} x_j + \sum_{j=1}^{d} c_{2,j} x_j^2 + \sum_{j=d+1}^{\binom{d}{2}} c_{1,j} x_1 x_2 + \dots + c_{2,\binom{d}{2}} x_{d-1} x_d + \sum_{j=1}^{d} c_{3,j} x_j^3 + \sum_{j=d+1}^{\binom{d}{2}} c_{3,j} x_1 x_2^2 + \dots + c_{3,\binom{d}{2}} x_{d-1} x_d^2 + c_{3,-} x_1 x_2 x_3$$

P

Taylor extended to multi dimension

$$d, \binom{d}{2}, \binom{d}{3}, \dots, \binom{d}{p}$$

the problem is that the number is enormous:

$$d = 10^{12} \quad p = 15 \qquad \binom{d}{p} \simeq d^p \simeq (10^{12})^{15} = 10^{180}$$

↓ Stirling approximation

Classical mathematics cannot be used in this context.

$y = f(x) = \cancel{b} + wx$ → We linearize the model and remove the constant

$O(d)$

$$l(f(x), y) = (y - f(x))^2$$

$$C(f) = w^T w = \|w\|^2 \quad \text{this is done by analogy}$$

$$\min_w \frac{1}{\cancel{n}} \sum_{i=1}^{n} (w x_i - y_i)^2 + \lambda \|w\|^2 \qquad X = \begin{bmatrix} x_1^T \\ \vdots \\ x_n^T \end{bmatrix} \in \mathbb{R}^{n \times d}$$

$$\min_w \|Xw - y\|^2 + \lambda \|w\|^2 \qquad y = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \in \mathbb{R}^n$$

$n \times d$ $d \times 1$ $n \times 1$

$n \times 1$

$$\min_w \underbrace{\|Xw - y\|^2}_{\text{Paraboloid}} + \underbrace{\lambda \|w\|^2}_{\text{Paraboloid}} = \text{Paraboloid}$$

Ridge Regression

Primal Formulation

$$w^T X^T X w - 2 w^T X^T y + y^T y + \lambda w^T \mathbb{I} w$$

$$\nabla_w (*) = \emptyset$$

$$2X^T X w - 2X^T y + \emptyset + 2\lambda \mathbb{I} w = \emptyset$$

$$(X^T X + \lambda \mathbb{I}) w^* = X^T y$$

$$\underbrace{w}_{d \times d} = \underbrace{(X^T X + \lambda \mathbb{I})^{\#}}_{d \times n \;\; n \times d} X^T y$$

$O(d^2)$ $\quad$ $O(d^3)$

the solution is quadratical in the number of dimensions

This is good because this algorithm is quadratical in the number of dimensions but not in the number of samples

---

$$\min_w \|Xw - y\|^2 + \lambda \|w\|^2$$

$$(X^T X + \lambda \mathbb{I}) w^* = X^T y$$

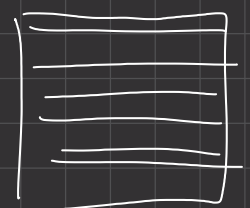
$X^T w \overset{\parallel}{\neq} \emptyset$ → parallel — Span

$X^T w \overset{\perp}{=} \emptyset$ → perpendicular — Kernel

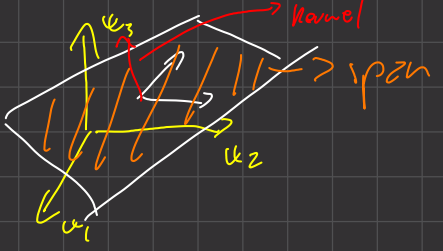
$$w^* = X^T \alpha = \sum_{i=1}^{n} \alpha_i x_i$$ we want to linearize the solution

---

EXAMPLE



The lines represent a subspace and the lines live in the subspace



Proof by absurdum

$$w^* = X^T \alpha = \sum_{i=1}^{n} \alpha_i x_i = w^{\parallel}$$

By absurd I assume : $w^* = w_{\parallel} + w_{\perp}$

$\| X(w_{\|} + w_{\perp}) - y\|^2 + \lambda \|w_{\|} + w_{\perp}\|^2 =$ 

$= \| X w_{\|} + \emptyset - y\|^2 + \lambda (\|w_{\|}\|^2 + \|w_{\perp}\|^2)$

$= \| X w_{\|} - y\|^2 + \lambda \|w_{\|}\| + \lambda \|w_{\perp}\|^2$

> 0 because I assumed $w_{\perp}$ to be present

↳ less than?

So since $w_{\|}$ is smaller than w^* we reached an absurdum since w^* must be the minimum so:

$$w^* = X' \alpha$$

$$\min_{\alpha} \|X X' \alpha - y\|^2 + \lambda \alpha^T X X^T \alpha$$

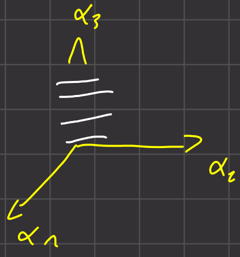
$Q = X X' \stackrel{?}{=}$ n [x_i^T] (d) d [x_j^T] (n) $\rightarrow Q_{ij} = x_i^T x_j$

Properties of Q:

$Q_{ij} = Q_{ji}$ symmetric

$Q \geq \emptyset$ semidefinit positive

$$\min_{\alpha} \|Q \alpha - y\|^2 + \lambda \alpha^T Q \alpha$$



$\nabla_{\alpha} (\alpha^T Q^T Q \alpha - 2\alpha^T Q^T y + y^T y + \lambda \alpha^T Q \alpha) = \emptyset$

$2 Q^T Q \alpha - 2 Q^T y + 2 \lambda Q^T \alpha = \emptyset$

↳ $> \emptyset$

$$(Q + \lambda \mathbb{1}) \alpha = y$$

→ linear system of which we can do the pseudoinverse

Dual

$X X^T$ → n×d, d×n → n×n $O(n^3)$

primal

$$(X' X + \lambda \mathbb{1}) w^* = X y$$

d×n, n×d → d×d $O(d^3)$

We can adapt

$w^* = X' \alpha$ Representer Theorem.

We moved to yet the Ridge regression in multidimension. The limitation is that we assumed the model to be linear.

$$\min_{w} \|Xw - y\|^2 + \lambda \|w\|^2 \qquad f(x) = w^* x$$

$$(X^T X + \lambda n) w^* = X^T y$$

$$w^* = X^T \alpha^*$$

$$(Q + \lambda n)\, \alpha^* = y \qquad f(x) = w^{*T} x = \alpha^{*T} X x = \sum_{i=1}^{n} \alpha_i^* x_i^T x$$

↳ $Q_{ij} = x_i^T x_j$

$w^* = X^T \alpha$

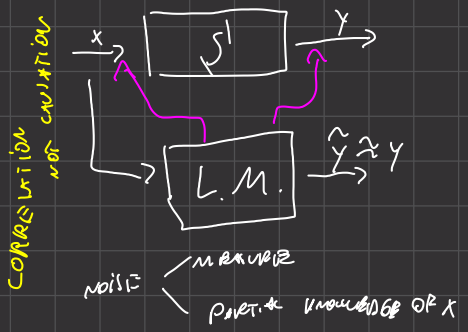
20/10/24

$D_n = \{(x_1, y_1) \dots (x_n, y_n)\}$ i.i.d

$x \in \mathbb{R}^d \quad y \in \mathbb{R}$

$f(x) = w x$

We use a linear model because it's simple and allows us to use all the inputs



$$l(f(x), y) = (f(x) - y)^2$$

$$C(f) = w^T n w = \|w\|^2$$

$$\min_{w} \|Xw - y\|^2 + \lambda \|w\|^2$$

$$X = \begin{bmatrix} x_1^T \\ \vdots \\ x_n^T \end{bmatrix} \in \mathbb{R}^{n \times d} \qquad y = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \in \mathbb{R}^{n \times 1}$$

$(X^T X + \lambda n) w^* = X^T y \quad O(d^2)$

$w^* = X^T \alpha^*$

$(Q + \lambda n)\, \alpha^* = y \quad O(n^2)$

↳ $Q_{ij} = x_i^T x_j$ → Note, I only need to know the inner products.

$f(x) = w x = \sum_{i=1}^{n} \alpha_i^* x_i^T x$

Problems:
- X in not always bidimensional
- Who told me that $\|w\|^2$ is a good complexity measure
- Is linear a good model (Ridge regression)?
- How to choose f, ℓ, C, λ and algorithm for min

Why $C(f) = \|w\|^2$

Can be stated as

$$R(f) \leq \underset{(1-\delta)}{\hat{R}(f)} + \text{“}\alpha_{n_f}\text{”} + \text{“}\delta_{,n}\text{”}$$

This formula is a bit naive because the number of function is infinite

Now we want to bound the distance: $R(f) - \hat{R}(f)$

$\mathbb{E}_{D_n}\{R(f) - \hat{R}(f)\} \leq \ldots$ (D_n dependent)

↳ I want to check what happens if I repeat the measure many times

$\hat{R}(f)$ is random because depends on D_n, but once we chose E_{D_n} now it's deterministic.

$$\mathbb{E}_{D_n}\left\{R(f) - \hat{R}(f)\right\} \leq \mathbb{E}_{(x_1,y_1)\ldots(x_n,y_n)}\left\{\mathbb{E}_{(x,y)}\ell(f(x),y) - \frac{1}{n}\sum_{i=1}^{n}\ell(f(x_i),y_i)\right\} =$$

$$= \mathbb{E}_{(x_1,y_1)\ldots(x_n,y_n)}\left\{\underbrace{\mathbb{E}_{(x_1',y_1')\ldots(x_n',y_n')}}_{\text{GHOST SAMPLE}}\frac{1}{n}\sum_{i=1}^{n}\ell(x_i',y_i') - \frac{1}{n}\sum_{i=1}^{n}\ell(f_{x_i},y_i)\right\}$$

You assume you have it

$$= \mathbb{E}_{\substack{(x_1,y_1)\ldots(x_n,y_n)\\(x_1',y_1')\ldots(x_n',y_n')}}\left\{\frac{1}{n}\sum_{i=1}^{n}\left[\ell(f(\underline{x_i}),y_i) - \ell(f(\underline{x_i'}),\underline{y_i'})\right]\right\}$$

$$\leq \mathbb{E}_{\substack{(x_1,y_1)\ldots(x_n,y_n)\\(x_1',y_1')\ldots(x_n',y_n')}}\left\{\sup_{f\in F}\frac{1}{n}\sum_{i=1}^{n}\left[\ell(f(\underline{x_i}),y_i) - \ell(f(\underline{x_i'}),\underline{y_i'})\right]\right\}$$

$$= \mathbb{E}_{\substack{(x_1,y_1)\ldots(x_n,y_n)\\(x_1',y_1')\ldots(x_n',y_n')\\\sigma_1\ldots\sigma_n}}\left\{\sup_{f\in F}\frac{1}{n}\sum_{i=1}^{n}\sigma_i\left[\ell(f(x_i),y_i) - \ell(f(x_i'),y_i')\right]\right\}$$

Note that adding σ does not change anything because I have all the terms replicated twice because of this

σ_i iid

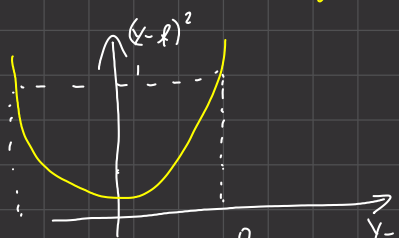
$\sigma_i \in \{-1,+1\}$ $\quad P\{\sigma_i = 1\} = P\{\sigma_i = -1\} = \frac{1}{2}$ Rademacher random variables

$$\leq \mathbb{E}_{\substack{(x_1,y_1)\dots(x_n,y_n)\\(x_1',y_1')\dots(x_n',y_n')\\ \sigma_1 \dots \sigma_n}} \left\{ \sup_{f \in F} \frac{1}{n} \sum_{i=1}^{n} \sigma_i \left[\ell(f(x_i), y_i) + \ell(f(x_i'), y_i') \right] \right\}$$

$$= \mathbb{E}_{\substack{(x_1,y_1)\dots(x_n,y_n)\\ \sigma_1 \dots \sigma_n}} \sup_{f \in F} \frac{2}{n} \sum_{i=1}^{n} \sigma_i \ell(f(x_i), y_i)$$

Rademacher Complexity

Let's now consider the square loss:



It's Lipshitz

Lipshitz constant

$$|g(x) - g(x')| = L\,|x - x'|$$

$$|\ell - \ell'| \leq L\,|f(x) - f(x')|$$

$$\mathbb{E}_{\substack{(x_1,y_1)\dots(x_n,y_n)\\ \sigma_1 \dots \sigma_n}} \sup_{f \in F} \frac{2}{n} \sum_{i=1}^{n} \sigma_i \ell(f(x_i), y_i) \leq$$

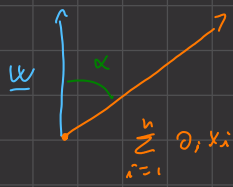
$$\leq \mathbb{E}_{\substack{(x_1,y_1)\dots(x_n,y_n)\\ \sigma_1 \dots \sigma_n}} \sup_{f \in F} \frac{2L}{n} \sum_{i=1}^{n} \sigma_i \underbrace{f(x_i)}_{w x_i} =$$

$\|w\|^2$

$$\left(w^* = (X^T X + \lambda I)\, X^T y\right)$$

$$= \mathbb{E}_{\substack{(x_1,y_1)\dots(x_n,y_n)\\ \sigma_1 \dots \sigma_n}} \sup_{\|w\|^2 \leq \|w^*\|^2} \frac{L^2}{n} \underbrace{\sum_{i=1}^{n} \sigma_i w^T x_i}_{n} =$$

$$w^T \sum_{i=1}^{n} \sigma_i x_i$$



$$\|w\| \cdot \left\| \sum_{i=1}^{n} \sigma_i x_i \right\| \cdot \cos(\alpha)$$

ANGLE BETWEEN THE VECTORS

Note: in order to have the supremum $\cos(\alpha) = 1$ and $w = w^*$ with it the maximum

$$= \mathbb{E}_{\substack{(x_1,y_1)\dots(x_n,y_n)\\ \sigma_1 \dots \sigma_n}} \sup_{\|w\|^2 \leq \|w^*\|^2} \frac{2L}{n} \|w^*\| \left\| \sum_{i=1}^{n} \sigma_i x_i \right\|$$

this is a constant, but I still leave it.

$$\propto \|w^*\|$$

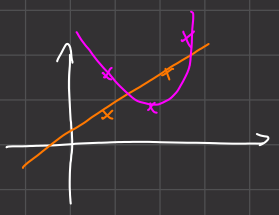
proportional

$R(f)$ risk on data
$R(\beta)$ risk on population

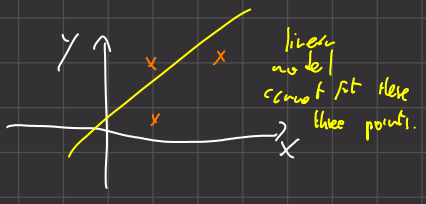
Just now we proved that the distance between $R(\beta)$ and $\hat{R}(\beta)$ is proportional to $\|\cdot\|$ so it does make sense that the complexity is measured with $\|\cdot\|^2$

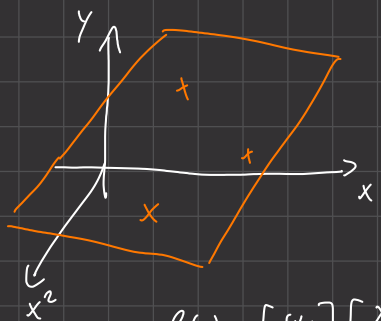
Generalize to non linear model

$x \in \mathbb{R}$ $\quad f(x) = \sum_{i=0}^{p} c_i x^i$



We are using a more complex function in the space





$$f(x) = \begin{bmatrix} w_1 \\ w_1 \end{bmatrix} \begin{bmatrix} x \\ x^1 \end{bmatrix} = w \, \phi(x)$$

One could see the increase of complexity of x as the increase of space.

$$f(x) = \sum_{i=0}^{p} c_i x^i = w' \phi(x) = \begin{bmatrix} w_1 \\ \vdots \\ w_d \end{bmatrix} \begin{bmatrix} \phi_1(x) \\ \vdots \\ \phi_d(x) \end{bmatrix} = \begin{bmatrix} w_0 \\ \vdots \\ w_p \end{bmatrix} \begin{bmatrix} x^0 \\ \vdots \\ x^p \end{bmatrix}$$

Model for the ridge regression

$$f(x) = w' x \longrightarrow w \, \phi(x)$$

Now it's non linear

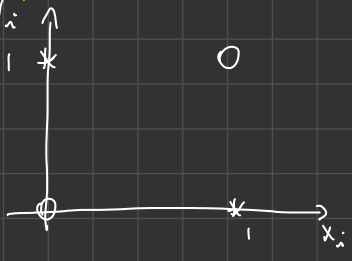
Non linear thanks to ϕ

$$= \sum_{i=1}^{n} x_i^T x_i \, x \rightarrow \sum_{i=1}^{n} \alpha_i \, \phi(x_i)^T \phi(x_i)$$

$$\alpha^* = (Q + \lambda n)^{\#} Y$$

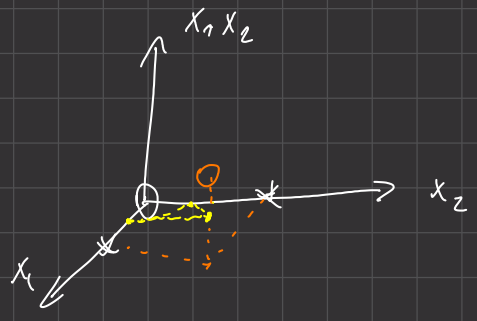
$\hookrightarrow Q_{ij} = x_i^T x_j \rightarrow \phi^T(x_i) \, \phi(x_j)$

Example.



In this configuration there is no linear model to split circles on stars

We apply ϕ to go in higher dimension:



Now we have a place able to split circles and stars.

Using a linear model in ϕ it's equivalent of using a non linear in the first space. This just by calculating $\phi(x)$ to x.

$$\phi(x) = \begin{bmatrix} 1 \\ x_1 \\ \vdots \\ x_d \\ x_1^2 \\ \vdots \\ x_d^2 \\ x_1 x_2 \\ \vdots \\ x_d, x_d \\ \vdots \end{bmatrix} \quad \begin{matrix} 1 \\ d \\ d \\ \binom{d}{2} \\ \\ \binom{d}{p} \simeq d^p \end{matrix}$$

N.P problem if using ϕ too much.

To solve this problem we will use Mercer and Kernels.

Kernels are a class of function that takes two vectors and return a number.

$$K(u, v) \to \mathbb{R} \qquad K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$$

$$K(u, v) = \phi^T(u)\, \phi(v) \quad \text{where } \phi \text{ overspace}$$

A Kernel function is a Kernel when the integral over u and v :

$$\int_{\mathbb{R}^d} \int_{\mathbb{R}^d} K(u, v)\, g(u) g(v)\, du\, dv \geq 0 \quad \forall g : \int_{\mathbb{R}^d} g(x)^2 dx < \infty$$

Example of Kernels:

$$\begin{cases} K(u, v) = e^{-\gamma \|u - v\|^2} \\ K(u, v) = (u^T v)^p \\ K(u, v) = (v^T u + c)^p \\ K(u, v) = e^{-\gamma \|u - v\|_1} \end{cases}$$

the mercer theorem is very important because we can just put the kernel

$$\sum_{i=1}^{n} \alpha_i \; K(x_i, \underline{x}) \qquad K(x_i, x_j)$$

} even if I don't know the ϕ that parented the kernel

$$K(u, v) = e^{-\gamma \|u - v\|^2}$$ GAUSSIAN / RBF Kernel

Let's go one dimensional with is simpler.

$$K(u, v) = e^{-\gamma (u-v)^2} = e^{-\gamma u^2} e^{-\gamma v^2} e^{2uv} = *$$

$\in \mathbb{R}^1$

Using the McLaren series which is the Taylor series called on ϕ:

$$e^x = \sum_{i=0}^{\infty} \frac{d^i e^x}{dx^i}\bigg|_{x=0} \frac{1}{i!} x^i = \sum_{i=0}^{\infty} \frac{x^i}{i!}$$

$$* = e^{-\gamma u^2} e^{-\gamma v^2} \sum_{i=0}^{\infty} \frac{(2\gamma uv)^i}{i!} \rightarrow 1 + 2uv + \frac{2^2}{2!} u^2 v^2 + \frac{2^3}{3!} u^3 v^3 + \dots$$

$$= e^{-\gamma u^2} \begin{bmatrix} 1 \\ \sqrt{2\gamma}\, u \\ \sqrt{\frac{2^2\gamma^2}{2}}\, u^2 \\ \sqrt{\frac{2^3\gamma^3}{3!}}\, u^3 \end{bmatrix}^T e^{-\gamma v^2} \begin{bmatrix} 1 \\ \sqrt{2\gamma}\, v \\ \sqrt{\frac{2^2\gamma^2}{2}}\, v^2 \\ \sqrt{\frac{2^3\gamma^3}{3!}}\, v^3 \end{bmatrix} = \phi^T(u)\, \phi(v)$$

$\phi \in \mathbb{R}^{\infty}$

OVER-PARAMETRIZATION

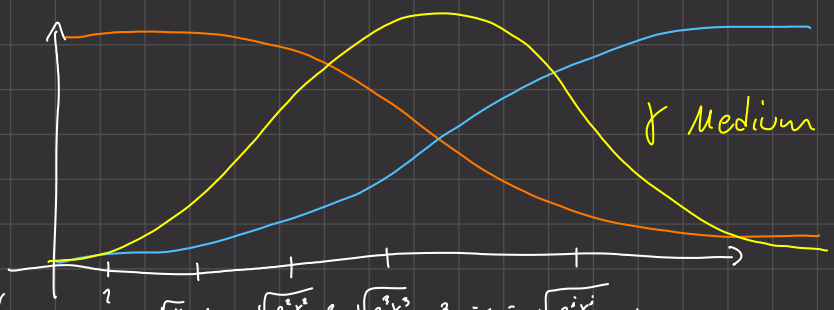
With the Gaussian kernel I projected my data in an infinite dimension space and then I can fit them - an infinite model

the problem here is that I can over fit my data but use a complexity term like w^2 that allows us to regulate and choose 'simple' functions in this infinite space.

Using Gaussian kernel (Kernel Ridge Regression) I introduces two additional hyperparameters, the hyperparameter of the kernel and the kernel itself.

1:39:35

Let's understand the effect of gamma on the solution



ϕ Gaussian $\quad 1 \quad \sqrt{2\gamma}\,u \quad \sqrt{\frac{2^2\gamma^2}{2}}\,u^2 \quad \sqrt{\frac{2^3\gamma^3}{3!}}\,u^3 \quad \cdots \quad \sqrt{\frac{2^i\gamma^i}{i!}}\,u^i$

$w \quad w_1 \quad w_2 \quad w_3 \quad w_4 \quad \cdots \quad w_s$

$\downarrow$

$\|w\|^2 = C(\phi)$

I cannot choose w too big because limits complexity

γ small $\rightarrow$ linearity

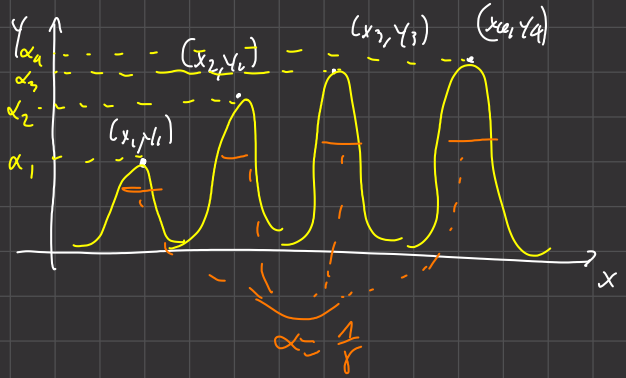
γ large $\rightarrow$ non-linearity

This because w needs to be the same power of the solution to have an effect, but the bigger the more complex $\rightarrow$ non linearity.

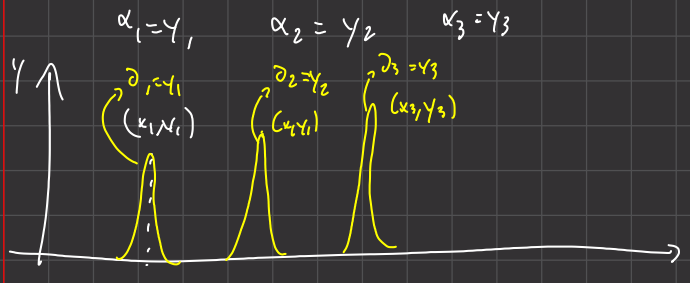
Another perspective

$$f(x) = w^T \phi(x) = \sum_{i=1}^{n} \alpha_i K(x_i, x) = \sum_{i=1}^{n} \alpha_i e^{-\gamma \|x_i - x\|^2}$$

Gaussian centred in x_i with variance $\sigma = \frac{1}{\gamma}$



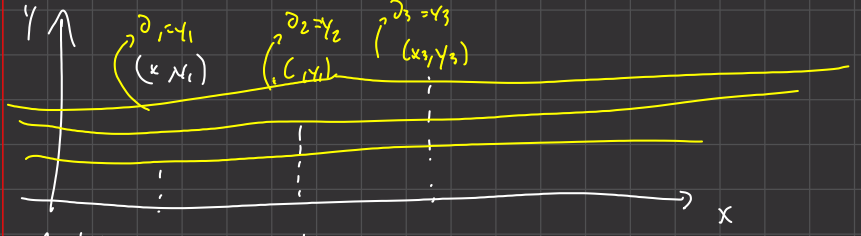
Suppose now that γ is very large, then the sum of these gaussians



$$y_i \triangleq f(x_i) \quad \forall i \in (1 \dots n)$$

This solution is over fitting when γ very large

If γ is very small



$\gamma = \phi$

Model is line each is the simplest model.

$$f(x) = w^* x = \sum_{i=1}^{n} \alpha_i^* k(x_i, x)$$

$Q_{ij} = k(x_i, x_j)$

$k(u, v) = e^{-\gamma \|x - v\|^2}$

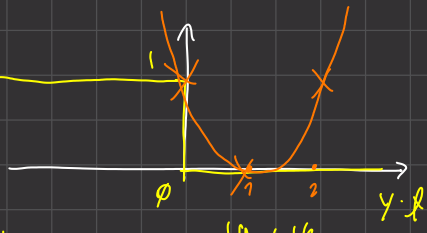
y — $\mathbb{R}$

$\{-1, 1\}$

$\{1, \dots, C\}$

With kernel ridge regression we can deal with all of them

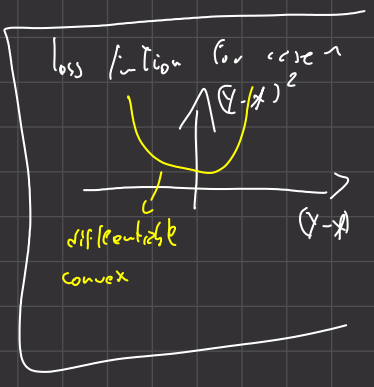
$y \in \{+1, -1\}$ In the binary decision we we check the product between y and f



and y can only be ± 1

non differentiable

non convex



(let's stay with yf then $y = \pm 1$

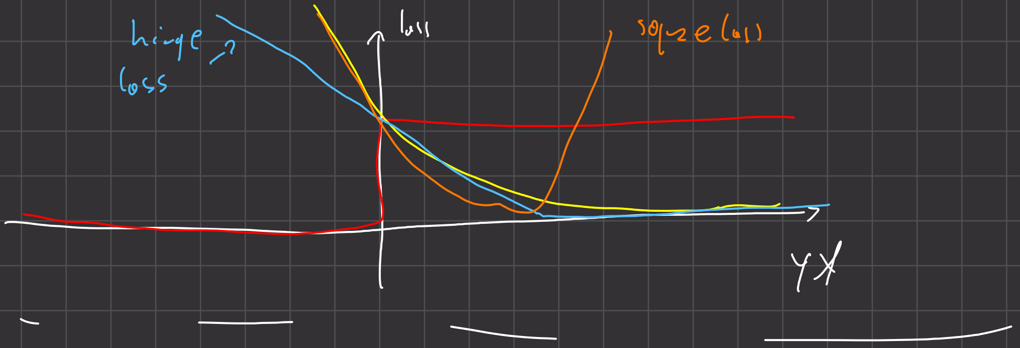
Since y

$yf = \phi \rightarrow f = \phi \rightarrow (y-f)^2 = 1$

$yf = 1 \rightarrow f = y \rightarrow (y-f)^2 = \phi$

$yf = 2 \rightarrow f = 2y \rightarrow (y-f)^2 = 1$

K.R.R. is used in binary classification.



Multiclass classification

$y \in \{1, 2, 3 \dots C\}$

$(f(x) - y)^2$

$f_1 = 6 - 1 \to 5$

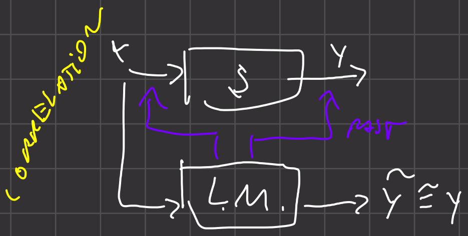
$f_2 = 2 - 1 \to 1$

wrong because there is no weight of distance in class

For binary case ok because you had just two label.

2017/10/24

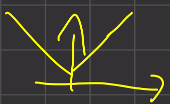
$x \in \mathbb{R}^d$, y — $\mathbb{R}$ / $\{-1,+1\}$ / $\{1 \dots C\}$



$D_n = \{(x_1, y_1) \dots (x_n, y_n)\}$ i.i.d

Noise — measure / x partially

$f(x) =$
- $\sum_{i=0}^{p} c_i x^i$
- wx
- $w\phi(x)$ — ϕ defined implicitly thanks to kernel

$\ell(f(x), y)$
- $(y - f(x))^2$ 
- $|y - f(x)|$ Better if outliers 
- $\max(0, 1 - y f(x))$ 

$C(f)$
- CMC $\int \left| \frac{d^2 f(x)}{dx^2} \right|^2 dx$
- $w^T 1 w = \|w\|^2$
- $\|w\|_1$

$$\min_{f \in \mathcal{F}} \hat{R}(f) + \lambda \Omega(f) \quad \text{s.t.} \quad \hat{R}(f) \triangleq R(f)$$

$$\hat{R}(f) \triangleq \frac{1}{n} \sum_{i=1}^{n} \ell(f(x_i), y_i) \qquad R(f) \triangleq \mathbb{E}_{(x,y)} \ell(f(x), y)$$

$$\hat{f} \in \arg\min$$

$$\omega$$

Ridge Regression | Kernel RIDGE | SVM | Kernel SVM | Lasso regression

If we define:

$$X = \begin{bmatrix} x_1^T \\ \vdots \\ x_n^T \end{bmatrix} \in \mathbb{R}^{n \times d} \qquad y = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \in \mathbb{R}^n$$

We know that: $\underset{\text{(PRIMAL)}}{\text{RIDGE}} = \min_{\omega} \|X\omega - y\|^2 + \lambda \|\omega\|^2$

solution: $(\underbrace{X^T X}_{d \times d} + \lambda \mathbb{I}) \omega^* = X^T y \quad O(d^2)$

With the representer theorem:

$$\omega^* = X^T \alpha = \sum_{i=1}^{n} x_i x$$

$$\downarrow \text{when}$$

$$(Q + \lambda \mathbb{I}) \alpha^* = y \quad O(n^2)$$

$$\hookrightarrow Q_{ij} = x_i^T x_j$$

$$\hookrightarrow \phi^T(x_i)\, \phi(x_j)$$

$$\downarrow \qquad\qquad \downarrow$$

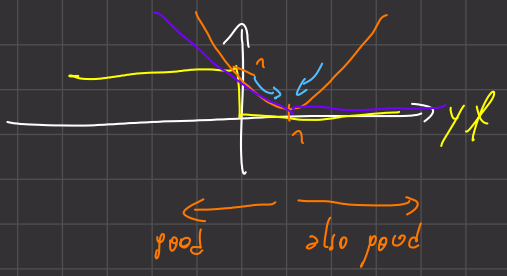
$$f(x) = \omega x = \sum_{i=1}^{n} \alpha_i x_i x \qquad k(x_i, x_j)$$

$$\downarrow$$

$$k(x_i, x)$$

Ridge allows to perform many calculus properties.. and allows me to deal with regression.

If we have to deal with binary classification where it's important to not make mistakes

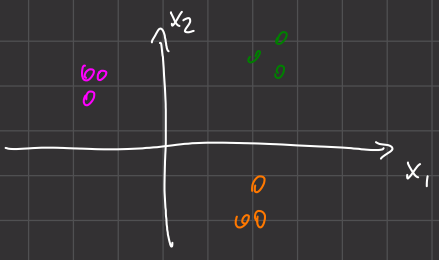


Best convex approx is the hinge loss but it's not differentiable.

the orange is an upper bound of the yellow.

Orange = convex + differentiable
Yellow: no

MULTICLASS CLASSIFICATION



- $y=1$
- $y=2$
- $y=3$

We cannot use $\ell(f(x), y) = (y - f(x))^2$ because of the concept of distance that does not work for y

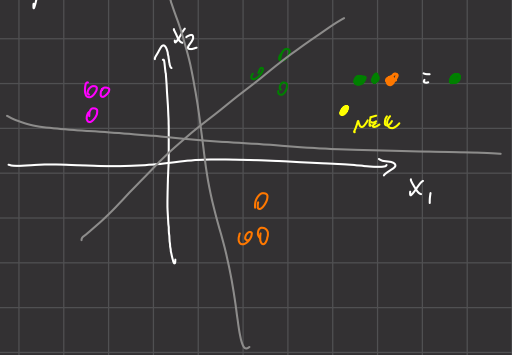
Ridge regression cannot be used since this loss function is not available, or is it?

The idea is to bring this problem to a series of binary classification

{ O.V.O. (AVA) One vs one
{ O.V.A. One vs all

O.V.O.
The idea is to take two problems at a time

Let's start with green and orange, then green vs purple and then purple vs orange



- In this case we have to solve 3 problems but in general is:

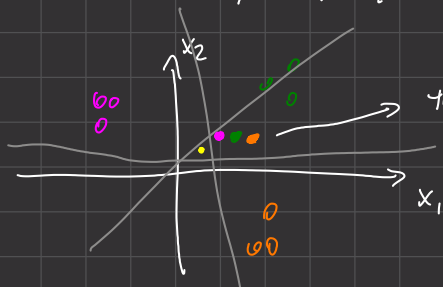
The number of samples $\approx 2\frac{n}{c}$

$\binom{c}{2} = \frac{c(c-1)}{2} \approx O(c^2)$

- If the original is balanced then the problem is balanced.

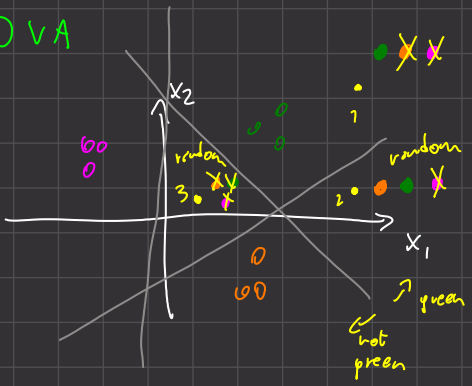
Now let's assume that one letter point moves. I need to label it now, so I check all the problems and then go with the majority voting

Now the problem is why majority voting is always available



But also when here is a point similar to all three so choose randomly

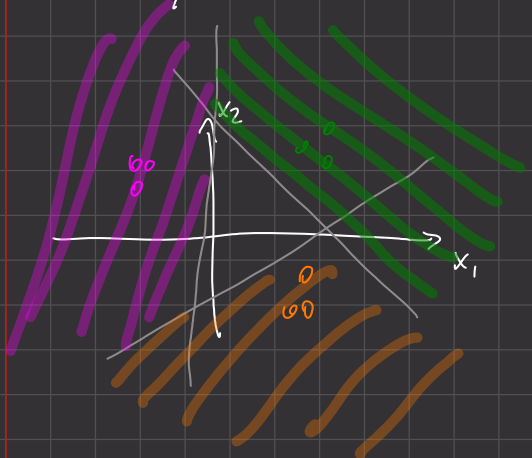
OVA



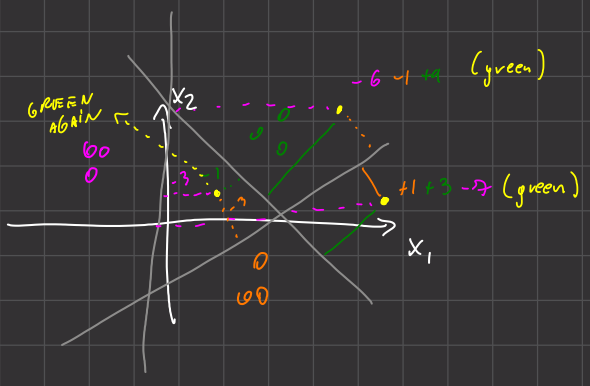
- PROBLEMS: C
- SAMPLES: n
- BALANCED: NO, $\left(\frac{n}{c}, (n-1)\frac{n}{c}\right)$ (+1, -1)
- Most score (not major voting)

while for 1 works, for 2 and 3 not very good. In the 3rd one in particular.

Let's try another idea. Color the space with the classes:



Note also that we can use distances:

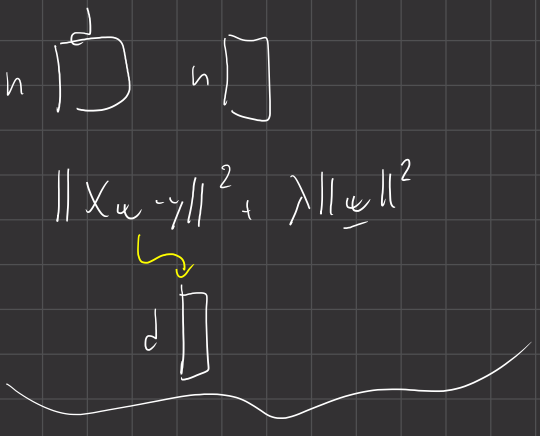


Optimization

$$\min_{f \in F} R(f) + \lambda C(x)$$

$R(f)$ ↓ squareloss — $C(x)$ ↓ norm. weights

$f(x) = wx$ is $\in \mathbb{R}^d$



$w_0 = \phi$

$w_{i+1} = w_i - \gamma \nabla_w (\dots)|_{w_i}$ (γ ↓ learning rate)

$Aw = b$ ← $(X'X + \lambda \mathbb{1})$ ← $X'y$

In general we want to solve a function $f(x)$ constrained:

$$\min_{x \in \mathbb{R}^d} f(x) \quad \text{s.t.} \quad g_i(x) \leq \phi \quad i = 1, \dots, m_i$$
$$h_i(x) = \phi \quad i = 1 \dots m_e$$

Optimization problem

One last case is:

$x \in \mathbb{R}^d$
$\mathbb{N}^d$
mix
Immediately give up because discrete is NP complete.

We cannot solve the opt. problem like this

We have also to add other options, i.e.
- convex function
- convex domain

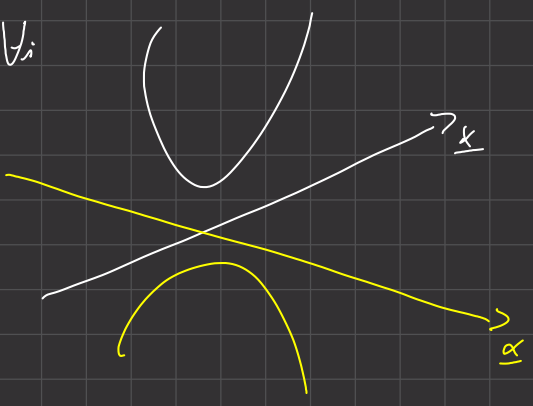
Lagrange Multiplier

- $\mathcal{L}_p(\underline{x}, \underline{\alpha}, \underline{\beta}) = f(\underline{x}) - \sum_{i=1}^{n_i} \alpha_i g_i(\underline{x}) - \sum_{i=1}^{n_e} \beta_i h_i(\underline{x})$

  $\underline{x} \in \mathbb{R}^d$, $\underline{\alpha} \in \mathbb{R}^{n_i}$, $\underline{\beta} \in \mathbb{R}^{n_e}$

  $\underline{\alpha}$: inequality constraints; $\underline{\beta}$: equality constraints

- 1) $g_i(\underline{x}) \leq 0 \ \forall i \quad h_i(\underline{x}) = 0 \ \forall i$
- 2) $\alpha_i \geq 0$
- 3) $\alpha_i g_i(\underline{x}) = 0$
- 4) $\nabla_{\underline{x}} \mathcal{L}_p = \underline{0}$



Note that if you minimise the primal, then it's the same as minimising the original function, so if I had to plot along the x axis we have something to be minimised.

$\mathcal{L}_d(\alpha, \beta) = \mathcal{L}_p(\underline{x}, \alpha, \beta)$ (dual)

$\nabla_{\underline{x}} \mathcal{L}_p = 0$ | d equations

$\underline{x}^* = f(\alpha, \beta)$

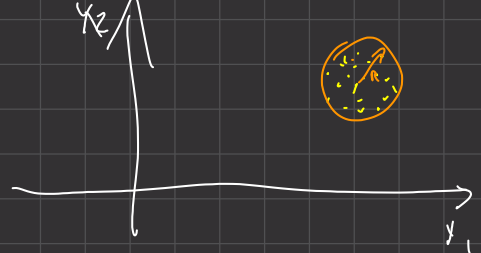
So replace $\underline{x}$ with $f(\alpha, \beta)$ and done.

If everything is convex then:

$$\min_{\underline{x}} \mathcal{L}_p = \max_{\alpha, \beta} \mathcal{L}_d$$

duality gap

Fermat problem



$P_n = \{x_1, \dots x_n\}$

$$\min_{\alpha, R} R^2 \quad \|x_i - \alpha\|^2 \leq R^2 \quad i \in 1 \dots n$$

$$\mathcal{L}_p(z, R^2, \alpha) = R^2 - \sum_{i=1}^{n} \alpha_i (R^2 - \|x_i - z\|^2)$$

↳ $\mathbb{R}^n$ -> one for each constraint

1) $\|x_i - z\|^2 \leq R^2$

2) $\alpha_i \geq 0$

3) $\alpha_i (R^2 - \|x_i - \alpha\|^2)$

4) $\frac{\partial \mathcal{L}_p}{\partial z} = 0 \Rightarrow z = \sum_{i=1}^{n} \alpha_i x_i \qquad \frac{\partial \mathcal{L}_p}{\partial R^2} = 1 - \sum_{i=1}^{n} \alpha_i = 0$

$$\frac{\partial \mathcal{L}_p}{\partial z} = \left(R^2 - \sum_{i=1}^{n} \alpha_i (R^2 - \|x_i - z\|^2)\right) =$$

TOPO CORRECTO

$$= 0 + \sum_{i=1}^{n} \alpha_i \frac{\partial}{\partial z} \|x_i - \alpha\|^2 =$$

$$x_i' x_i - 2x_i' z + \alpha' \alpha \quad (z z \to 0)$$

$$= 2 \sum_{i=1}^{n} \alpha_i (-x_i + z) = 0 \rightarrow z \sum_{i=1}^{n} \alpha_i = \sum_{i=1}^{n} \alpha_i x_i$$

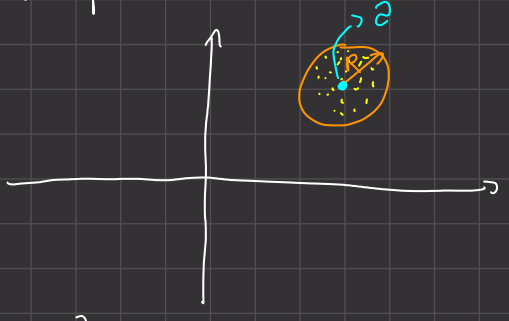
$$\mathcal{L}_D(\alpha) = R^2 \left(1 - \sum_{i=1}^{n} \alpha_i\right) + \sum_{i=1}^{n} \alpha_i \left(x_i' x_i - 2x_i' z + z' z\right)$$

(1 - Σ αᵢ = 0)

31/10/24

Minimizing a general function

Lagrangian multipliers



$S_n = \{x_1 \dots x_n\}$

Primal problem

$$\min_{z, R^2} \; R^2 \qquad \|x_n - z\|^2 \leq R^2$$

convex

$R \geq 0$ → Combination of circles (convex)

$$\mathcal{L}(a, R^2, \lambda) = R^2 - \sum_{i=1}^{n} \lambda_i \underbrace{(R^2 - \|x_i - a\|^2)}_{\neq 0}$$

λ_i vector of one Lagrangian mult for constraint

1) $\|x_i - a\|^2 \leq R^2$

2) $\lambda_i \geq 0$

3) $\lambda_i (R^2 - \|x_i - a\|^2) = 0$

Kuush-Kuhn-Tucker condition

Representer Theorem

4) $\frac{\partial \mathcal{L}}{\partial a} = 0 \rightarrow -\sum_{i=1}^{n} 2\lambda_i x_i + \sum_{i=1}^{n} 2\lambda_i a = 0 \rightarrow a = \sum_{i=1}^{n} \lambda_i x_i$

$\frac{\partial \mathcal{L}_P}{\partial R^2} = 0 \rightarrow 1 - \sum_{i=1}^{n} \lambda_i \rightarrow \sum_{i=1}^{n} \lambda_i = 1$

$$\mathcal{L}_D(\lambda) = R^2\left(1 - \sum_{i=1}^{n} \lambda_i\right) + \sum_{i=1}^{n} \lambda_i \left(x_i^T x - 2 x_i^T a + a^T a\right) =$$

$= 0$

$$= \sum_{i=1}^{n} \lambda_i x_i^T x_i - 2a \sum_{i=1}^{n} \lambda_i x_i + \sum_{i=1}^{n} \lambda_i a^T a =$$

$$= \sum_{i=1}^{n} \lambda_i x_i^T x_i - 2 \sum_{i=1}^{n} \lambda_i x_i^T \sum_{i=1}^{n} \lambda_i x_i + \sum_{i=1}^{n} \lambda_i x_i^T \sum_{j=1}^{n} \lambda_j x_j =$$

$$= -\sum_{i=1}^{n} \sum_{j=1}^{n} \lambda_i \lambda_j x_i^T x_j + \sum_{i=1}^{n} \lambda_i x_i^T x_i$$

$\sum_{i=1}^{n} \lambda_i = 0$ $\lambda_i \geq 0$ Lagrangian dual of lambda

$$\max_{\lambda} \; -\sum_{i=1}^{n} \sum_{j=1}^{n} \lambda_i \lambda_j x_i^T x_j + \sum_{i=1}^{n} \lambda_i x_i^T x_i \quad \Big\| \quad \text{Dual!!!}$$

$$\sum_{i=1}^{n} \lambda_i = 1 \;, \; \lambda_i \geq 0$$

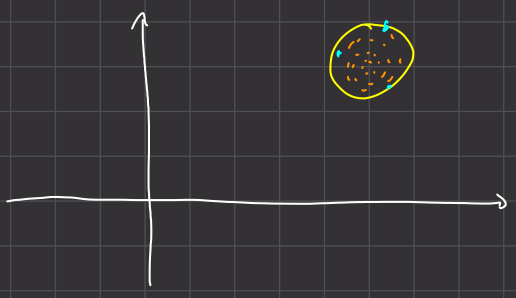
$a = \sum_{i=1}^{n} \lambda_i x_i$, R^2

$\lambda_i \geq 0 \begin{cases} \lambda = 0 \\ \lambda > 0 \rightarrow \lambda_i (\|x_i - a\|^2 + R^2) = 0 \end{cases}$

if $\lambda_i > 0$ $v = 0$

$R^2 = \|x_i - a\|^2$

I can compute the x_i referring to that λ_i that is greater than 0



x_i : BORDER / INSIDE

$\|x_i - a\|^2 < R^2$

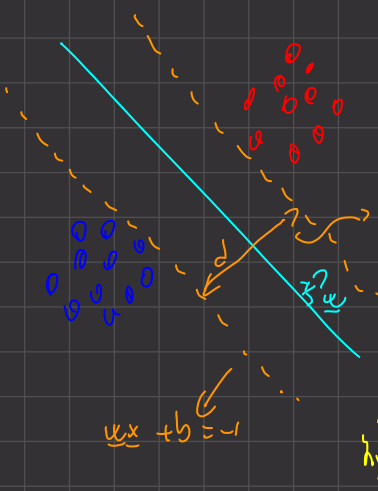
$\lambda_i (R^2 - \|x_i - a\|) = 0$

$\lambda_i = 0$

$\lambda_i = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}$ Mostly 0, only points different from 0 are the ones on the border

Space: $z = \sum_{i=1}^{h} \lambda_i x_i$

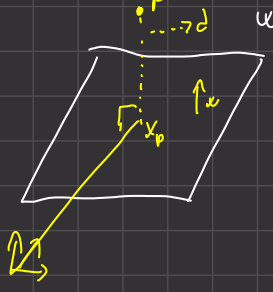
I just did compression which is the same of learning.



Infinite models to split the points.

Now E

Finding the distance between two planes



$p = x_p + d \frac{w}{\|w\|}$

$w^T p = w^T x_p + d \frac{w^T w}{\|w\|}$

$-b$

$d = \frac{w^T p \pm b}{\|w\|}$

$w^T x_g + b = 1$

$$d_\phi = \frac{w^T x_g + b + 1}{\|w\|} = \frac{1 - b + b + 1}{\|w\|} = \frac{2}{\|w\|}$$

we want $\max d \rightarrow \max \frac{2}{\|w\|} \approx \max \frac{2}{\|w\|^2} \rightarrow \min \frac{\|w\|^2}{2}$

this is like the regularizer of the ridge regression.

$$D_n = \{(x_1, y_1) \dots (x_n, y_n)\}$$

$x_i \in \mathbb{R}^d$, $y_i \in \{\pm 1\}$

$$\min_{w,b} \frac{1}{2}\|w\|^2$$

yf

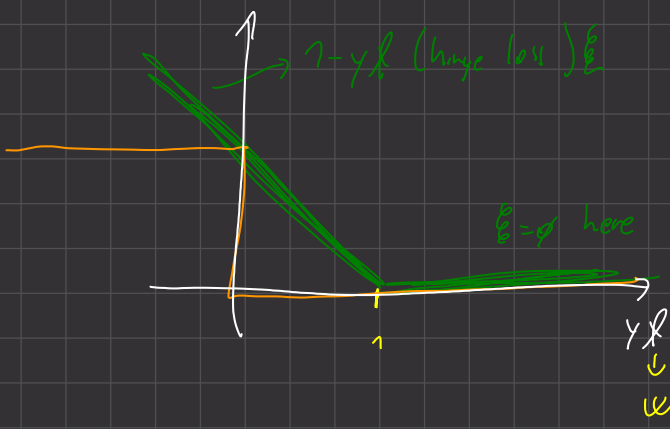
$y_i(w^T x_i + b) \geq 1 - \xi_i$ works in all directions

$\xi_i \geq 0$

I need to relax the constant otherwise I will not be able to do classification when a blue point is between the roads *

I need to force ξ to be small.

this is the SVM.



$$\min_{w,b} \frac{1}{2}\|w\|^2 + C \sum_{i=1}^{n} \xi_i$$

$y_i(w^T w_i + b) \geq 1 - \xi$

$f(x)$ in general is written like: $w \cdot x$ ~~+b~~ ← this can be removed because in kernel I have infinite number of powers.

$$\min_{w,\xi} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i$$

$$y_i\, w x_i \geq 1 - \xi_i$$

$$\xi_i \geq 0$$

Primal SVM no bias (N_0 bias)

Lagrange Primal

$$\mathcal{L}(w, \xi, \alpha, \beta) = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i - \sum_{i=1}^{n}\alpha_i\left(y_i\, w x_i - 1 + \xi_i\right) - \sum_{i=1}^{n}\beta_i\xi_i$$

constraints:

1) $w x_i \geq 1 - \xi_i$, $\xi_i \geq 0$
2) $\alpha_i, \beta_i \geq 0$
3) $\alpha_i\left(y_i\, w x_i - 1 + \xi_i\right) = 0$, $\beta_i\,\xi_i = 0$
4) $\frac{\partial \mathcal{L}_P}{\partial w} = 0 \quad w - \sum_{i=1}^{n}\alpha_i y_i x_i = 0 \rightarrow w = \sum_{i=1}^{n}\alpha_i y_i x_i$ → representer theorem

$$\frac{\partial \mathcal{L}}{\partial \xi_i} = 0 \rightarrow C - \alpha_i - \beta_i = 0 \quad \alpha_i = C - \beta_i \rightarrow \beta_i \geq 0 \rightarrow \alpha_i \leq C$$

$$\mathcal{L}_D(\alpha,\beta) = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j x_i^T x_j + \sum_{i=1}^{n}(C - \alpha_i - \beta_i)\xi_i + \sum_{i=1}^{n}\alpha_i + \sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j x_i^T y_j =$$

$$\mathcal{L}_D(\alpha) = -\frac{1}{2}\sum_{i=1}^{n}\alpha_i\alpha_j x_i x_j x_i^T x_j + \sum_{i=1}^{n}\alpha_i$$

$$0 \leq \alpha_i \leq C$$

$$\min_{w,b} + \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j x_i^T x_j - \sum_{i=1}^{n}\alpha_i$$

$$0 \leq \alpha_i \leq C_i \quad *$$

Thinking about Ridge:

$$\|Xw - y\|^2 + \lambda \|w\|^2$$

$$w^T X^T X w - 2 w^T X^T y + w^T \Gamma w$$

$$w^T (X^T X + \lambda \Gamma) w - 2 y^T X w$$

We can understand why α is convenient because both are trying to minimize a problem

$$\min_{\alpha} \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \underbrace{x_i^T x_j}_{\text{kernel}} - \sum_{i=1}^{n} \alpha_i$$

$$w = \sum_{i=1}^{n} \alpha_i y_i x_i \qquad f(x) = \underline{w}\, \underline{x} = \sum_{i=1}^{n} \alpha_i y_i \underbrace{x_i^T x}_{\text{kernel}}$$

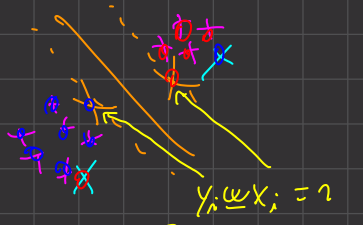
maximum margin

$$\min_{w, \xi} \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{n} \xi_i$$

$$y_i w_i x_i \geq 1 - C \xi_i$$

$$\xi_i \geq 0$$

$\|w\|^2 \to \|w\|_1$ ↓ Sparsity w



To interpret $0 \leq \alpha_i \leq C$

$\alpha_i = 0 \quad y_i \underline{w} x_i > 1 \to \xi_i = 0 \to \alpha_i (y_i \underline{w} x_i - 1 + \xi_i) = 0$ ($= 0$, > 0)

$y_i \underline{w} x_i = 1$

$\alpha_i \in (0, C)$

$\alpha_i (y_i \underline{w} x_i - 1 + \xi_i) = 0 \to y_i \underline{w} x_i = 1 - \xi_i$

$C - \alpha_i - \beta_i = 0 \to \beta_i \neq 0 \to \beta_i \xi_i = 0 \to \xi_i = 0$

$\alpha_i = C \quad \xi_i > 0 \to y_i \underline{w} x_i = 1 - \xi_i$

$\beta_i \xi_i = 0 \to \beta_i = 0 \to C - \alpha_i - \beta_i = 0$

The points not on the border do not contribute to the sum $w = \sum_{i=1}^{n} \alpha_i y_i x_i$

the algorithm learns from and which means that it's learning from the mistakes.

Note that we took Ridge and changed the loss function

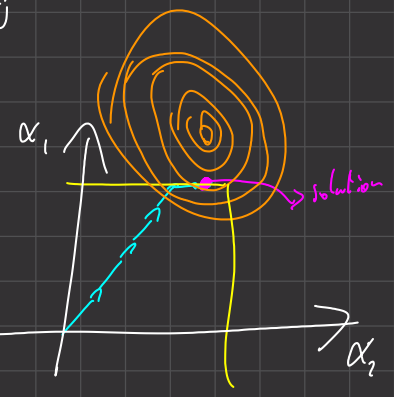
Now I want to solve:

$$\min_{\alpha} \frac{1}{2} \sum_{i,j}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j - \sum_{i=1}^{n} \alpha_i$$

$$0 \le \alpha_i \le C$$

$$\min_{\alpha} \frac{1}{2} \alpha^T Q \alpha - x_i^T \alpha \qquad Q_{ij} = y_i y_j x_i^T x_j$$

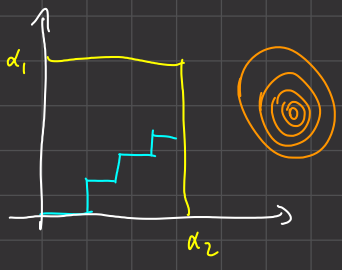
$$0 \le \alpha \le C \quad \text{BOX constraint} \qquad y_i = -1$$



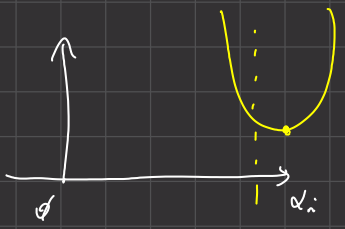
$$\alpha_{i+1} = \alpha_i + \gamma \nabla_\alpha (\dots)\Big|_{\alpha_i}$$

This is costly

Now I allow to move only on α_1 or α_2



Projecting on one axis

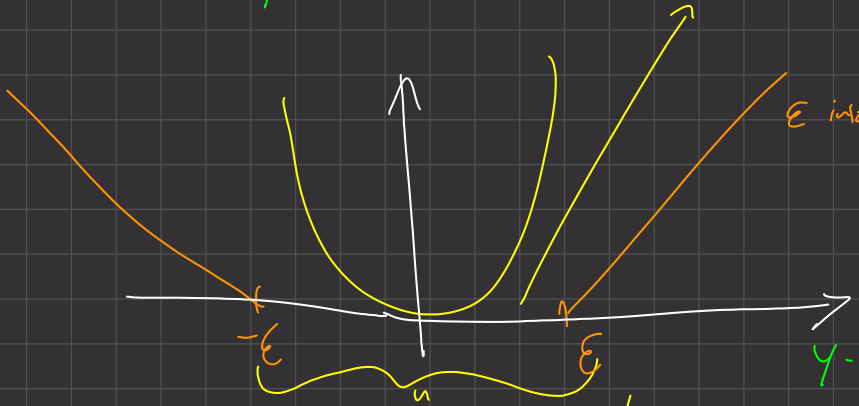


CHUNKING

↳ S.M.O

sequential minimal optimization.

Graph for regression

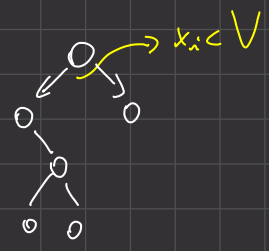


$$w = \sum_{i=1}^{n} \alpha_i x_i \rightarrow \text{sparsity}$$

Decision Rule Methods

$f(x) = w x$

$f(x) = \sum_{i=1}^{n} \alpha_i x_i^T x \simeq \sum_{i=1}^{n} \alpha_i K(x_i, x)$

$f(x) \rightarrow$ 

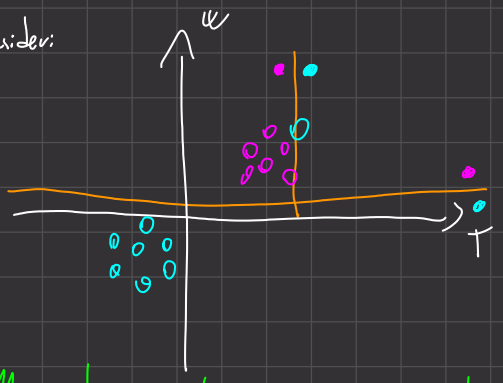
Binary trees
if - else
+
recursion

With recursion or if-else I can do decision trees.

This structure is discrete so the problem of optimizing it is N.P.

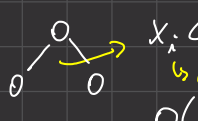
The idea of optimizing the tree is the one of optimizing one process at a time. In this case we do a greedy approach.

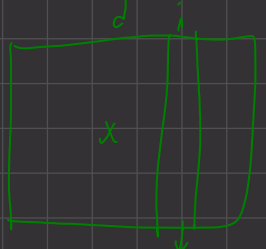
Consider:

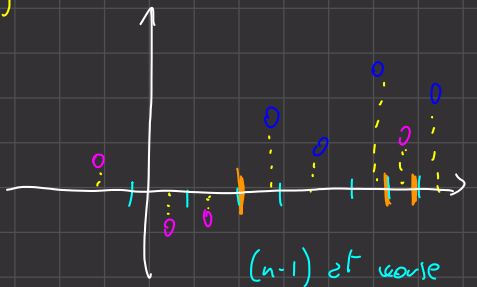


My decision tree is an universal approximator as a Gaussian Kernel.

It can create a square for every point.



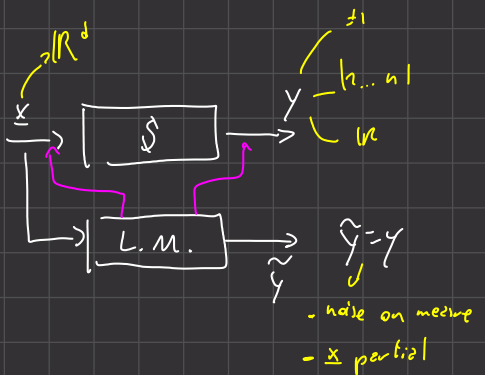




$\min_{\alpha, V} \hat{R}(f) + \lambda C(f)$ ~~+ $\lambda C(f)$~~ Goes away because we prune it

I trust more elements with more data than the ones with few.

05/11/24



I.i.d. because we sample data from the past and we need that the future will be similar to the past and that past info can be enriched with more info.

$D_n = \{(x_1, y_1), (x_2, y_2), \dots (x_n, y_n)\}$ i.i.d.

$x \in \mathbb{R}^d$

$y \in \dots$

$\ell(f(x), y)$ ⟨ $(y - f(x))^2$; $\max[0, 1 - y f(x)]$; ε - insensitive

$f(x)$ ⟨ $w \in X$; $w \in \phi(x)$ 

$C(f)$ = ⟨ $\|w\|_2^2$ norm of the weights ; $\|w\|_1$; depth of the tree

The loss function lets us define the empirical risk of the error that I commit on the data:

$$\hat{R}(f) = \frac{1}{n} \sum_{i=1}^{n} \ell(f(x_i), y_i)$$

$$R(f) = \mathbb{E}_{(x,y)} \ell(f(x), y)$$

$$R(f) \underset{(1-\delta)}{\leq} \hat{R}(f) + C(f) + \Delta(n, \delta)$$

this cannot be well represented

Occam - Razor principle (The simpler, the better)

$$\min_f \hat{R}(f) + \lambda C(f)$$

- closed form → dual (Lagrangian) ↳ appropriate for sparsity data
- gradient descent
- greedy

$w \in X$, $(y - f(x))^2$, $\|w\|_2^2$, closed → RIDGE REGRESSION Universal approximator

$\phi(x)$, " " " " → Kernel ridge

??? ↳ Kernel

2 x '' '' '' -> Lasso Sparsity ⊂

x φ(x) '' '' Sequential multiple optimization (SMO) -> SVC sparsity ...

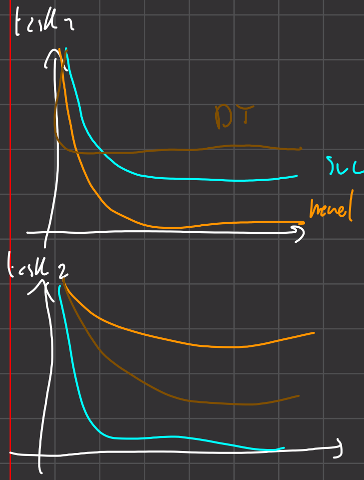
?, ?, DEPT, GREEDY -> D.T. interpretability, universal approximator.

No free lunch

λ?, ℓ?, $C(x)$?, $\phi(x)$?, kernel?

How to set up these parameters?

There is no optimal way of doing this because by setting one way good to others will go bad



with an universal classificator with infinite samples then I will get the best that I can get on that particular task

Suppose now that I have a RNG.

2, 3, 7, 9, 24 ...

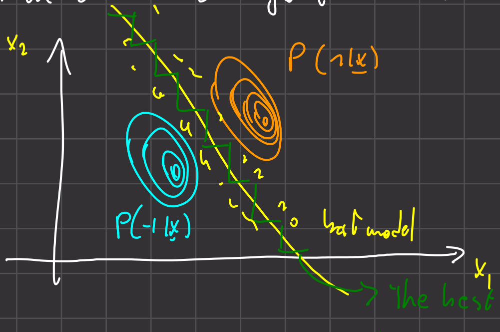
with a sum I will never be able to get the right number

Suppose not to use the exact RNG and the hyper is the seed, then a very predictor is perfect for the task.

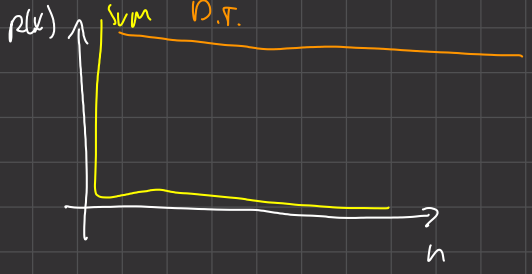
Generally, I need simple things because otherwise I fit the noise and not the signal

D.T. $O(n \ln(n))$ $O(d)$

The problem of this algorithms is:



-> The best that I can do with D.T.
But I would need many lots of samples



Suppose that you are sick and in the world the best doctors are these and the probability of error of them is:

$M_1 \rightarrow 0.1\%$

$M_2 \rightarrow 1.0\%$

$M_3 \rightarrow 0.01\%$

N.B. %

The best thing to do is Hiring all of them and use majority voting

M_1	M_2	M_3	Luce D.
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

0. correct
1. not correct

$0,999 \cdot 0,01 \cdot 0,0001 +$

$0,001 \cdot 0,99 \cdot 0,0001 +$

$0,001 \cdot 0,01 \cdot 0,9999 +$

$0,001 \cdot 0,01 \cdot 0,0001 = 0,00007$ x10 better than others

Note that I made a mistake here because I wrongly assumed: $P(A \cap B) = P(A)\,P(B)$ which is only true if the data is independent.
This is not correct since medics are dependent.

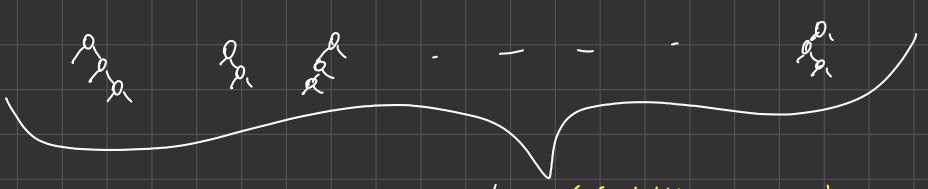
In a perfect world of medics I should get $P(A|B) = P(A)$

How can I make this assumption good enough? I can go ask people from around the world in order to have more independent results.

Democracy is another good example of using independent samples. Internet broke the independence.

Th:

$$\lim_{n.models \to \infty} P(maj.Vot.(fi)) \rightarrow oracle$$



oracle (if slightly independent)

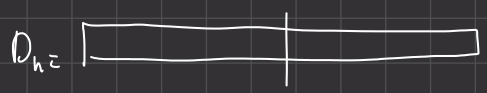
Note that I cannot create the oracle because we have n samples so limited trees.

D_n

I want to construct many decision trees where:

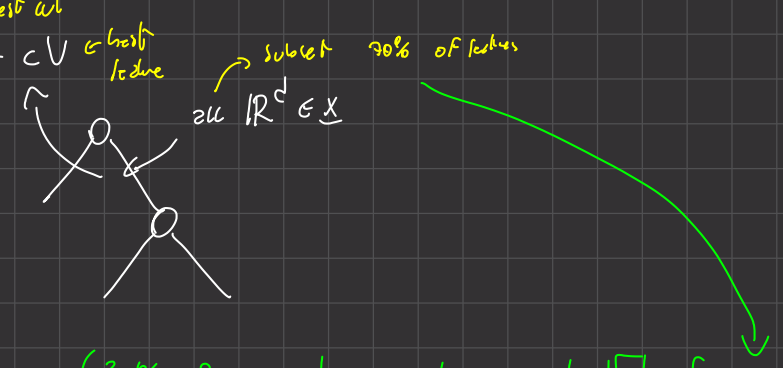
- maximise the accuracy of every D.T.
- minimize the dependency of each D.T.

In order to have independency we will split the data

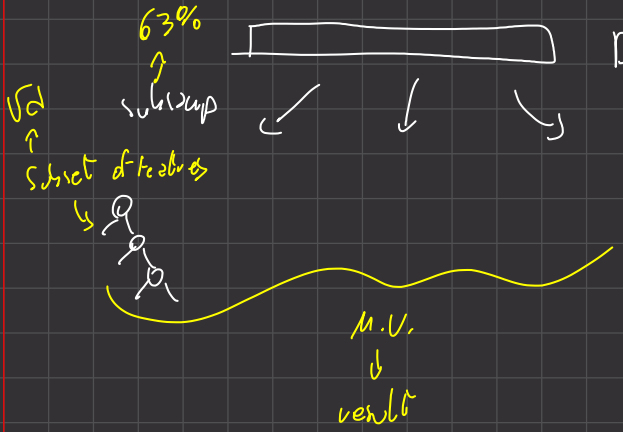


But this is not good for many trees. Better is to subsample 70% of the samples at random.

This is called BOOSTING



The best values are 63% for random sampling and $\sqrt{d}$ for



RANDOM FOREST

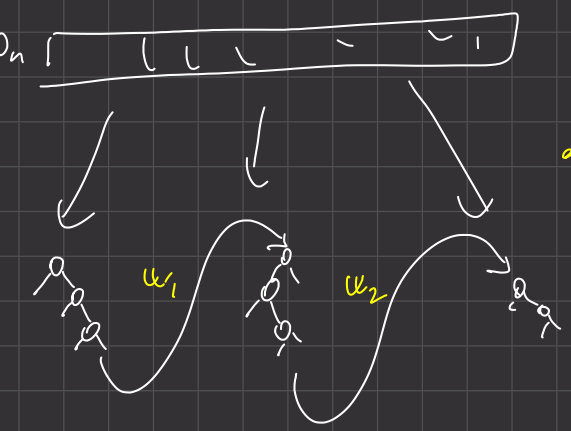
Note that number of trees is not an hyperparameter.

$\sqrt{d}$
63 %
are hyperparameters N.B.

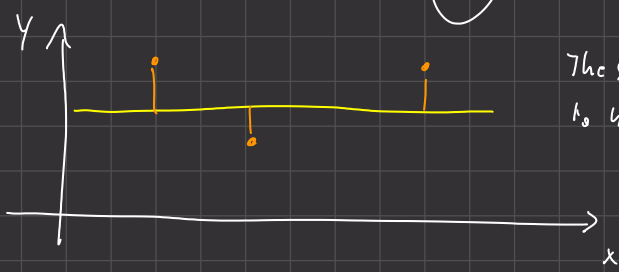
This is one of the ENSEMBLE METHOD algorithms.

The idea is using many independent weak classifiers in order to have a good classifier.

One other option is that the result of a decision tree goes into another decision tree.



another gradient
↑ ↑
x6 Boost
(good on testing data)



The sum of the residual is going to be smaller and smaller.

vector of weights can be optimized with gradient descent.

There are methods called ø learning.

I take a decision tree and construct it fully random.

$x_i < V$ i and V total random

I construct many of them and they are all random. The accuracy is ø because random. But I apply these trees on data and based on the desired a leaf becomes sick or healthy.

You take the N.V and get something that more or less is good as random forest. Nothing to be optimized.

Here eager is the fact that I am using a subset of data to do some of task

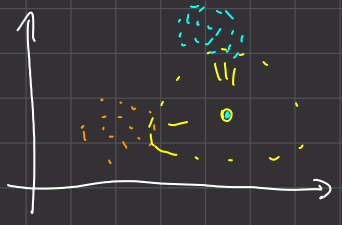
The complexity is the number of samples in order to do my decision

eg: as ???

LAZY Learners

The idea is that until someone asks me something I do nothing.

Example: K-nearest neighbours:



Parameter that regulates complexity: K

K = 1 → ∅ error
K = n. data → 50 % } Because M.V.

Hypotheses:
- distance
- weight of distance → (Gradient K-NN)