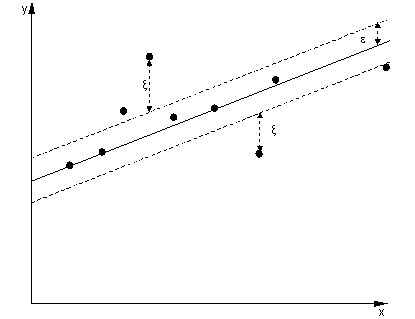
Prova per capire se sto capendo (EP guarda anche te le cose così da capire se hai capito e se stiamo capendo entrambi o il capire di qualcuno non è davvero capire o meglio è un capire sbagliato – probabilmente il mio)

Spero in ogni caso di darti una mano a dare un filo logico a quello che guarderai in giro.

Ocio che SVR è ben differente da SVM, a partire dall’obiettivo.

DEFINIZIONE DEL PROBLEMA

Per affrontare questo problema di regressione ci vogliamo affidare ad un modello di apprendimento supervisionato che è il Support Vector Regression. SVR ha come obiettivo, dati un certo parametro C (per capire il livello di regolarizzazione che desideriamo) ed un valore ε (per esprimere l’errore che accettiamo) trovare una funzione tale per cui ogni valore del nostro input non devii da essa più di ε (per questo ogni valore all’interno del cosiddetto ε-tube non viene considerato come errore nella fase di ottimizzazione, rendendo la loss del modello ε-insensitive). Parte fondante del modello, oltre a ciò sopra descritto, è dare allo stesso tempo importanza al mantenere la funzione *as flat as possible*, per evitare overfitting ed avere dunque un modello generalizzato.

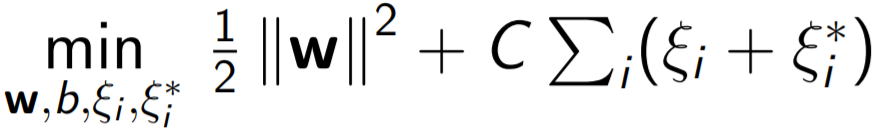


La funzione risultante dall’ottimizzazione del modello è descritta come f(x) = w \* x + b

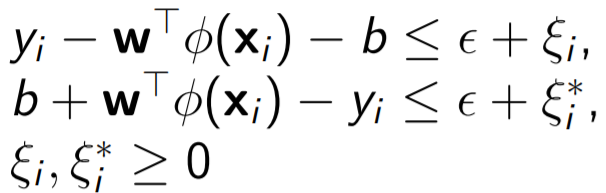
Obiettivo dell’ottimizzazione è fare in modo che la curva sia, di nuovo, *as flat as possible,* ma questo è equivalente ad un problema di ottimizzazione dove vogliamo avere ||w|| minimo.

Per comodità di formulazione del problema possiamo minimizzare ||w||^2 senza cambiare il significato. Questo ci permette di trasformarci in un problema di ottimizzazione quadratico, grazie al quale potremo approfittare della ***STRONG DUALITY*** più tardi. **(is this bullshit true? Have to check this!!!)**

Introduciamo inoltre delle variabili dette *slack* per formulare la nostra *objective function,* la quale rappresenta il nostro **PRIMAL PROBLEM**:

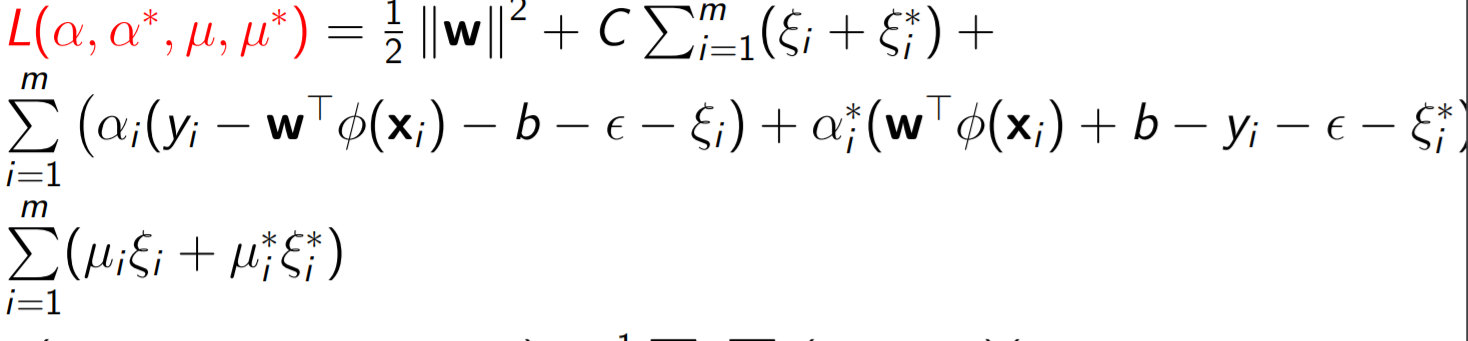


Ciò che viene sommato alla norma di w è un elemento che ci permette di regolare l’errore, e di conseguenza la penalità, dovuti alla possibile presenza di elementi che non rimangono all’interno dell’ ε-tube. Vediamo dunque come C funga da regolarizzatore in una casistica simile a L1. I vari ξ vengono detti *slack variables* e ci permettono a questo punto di poter definire i constraints che vincolano il problema per qualsiasi i-esimo dato del nostro problema:



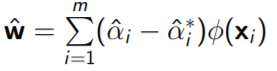
(x i-esimo rappresenta l’i-esimo input mentre y i-esimo rappresenta l’output che ci si aspetta 🡪 **supervised**)

Dato che il problema è quadratico ci troviamo nelle condizioni di STRONG DUALITY, dunque la soluzione al problema duale risulta equivalente a quella del problema primale. Costruiamo dunque il problema duale definendo la lagrangiana (equivalente del problema primale al quale aggiungo i vincoli sommandoli o sottraendoli, non sembra rilevante il + o il – ma sarebbe meglio guardare):

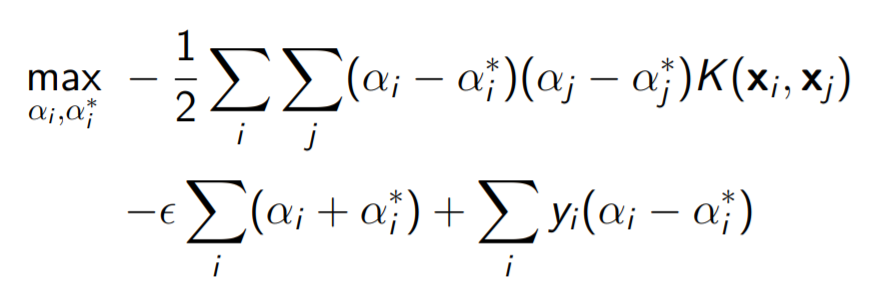


A causa del concetto di WEAK DUALITY qualsiasi valore del PRIMAL PROBLEM risulta >= del DUAL PROBLEM, diciamo dunque che (e qui è inventiva mia eh, però ha senso, meglio riguardare l’indiano un attimo) il punto in cui sono più vicini è dove il PRIMAL PROBLEM ha il minimo e il DUAL PROBLEM ha il massimo, che nella casistica di STRONG DUALITY ci daranno lo stesso valore!

Cerchiamo di lavorare con meno variabili possibili, togliamoci dai calcoli la variabile w, b e le 2 ξ. Come? Stiamo cercando un valore ottimo, dunque la derivata parziale relativa ad ognuna di queste variabili va posta a 0. Questo ci porta a poter ridefinire w:



E tramite sostituzione nella lagrangiana di qui sopra arriviamo ad una funzione dipendente solamente dai 2 α (se ci sono dubbi basta guardare la derivata completa che ho scritto su carta)



Dove quel K(xi,xj) rappresenta il KERNEL sostituendo ciò che sarebbe il dot product nello spazio necessitato. Infatti certi problemi di regressione non possono essere adeguatamente descritti da modelli lineari. Viene dunque comodo cambiare spazio di visualizzazione, per portarci in un nuovo spazio (con ogni probabilità a più dimensioni di prima) dove il problema diventa linearmente affrontabile. Il cambio di base risulta essere incredibilmente dispendioso ed è in questo caso che il kernel diventa il fulcro di tutto il modello. Ci permette infatti di eseguire il dot-product nello spazio attuale ma avere come risultato il dot-product nello spazio richiesto. Ci permette di risparmiare molte computazioni, infatti possiamo salvarci tutti i valori in una kernel matrix da riutilizzare ad ogni ciclo di ottimizzazione (credo, anche qui supposizione mia ma non credo di essere distante dalla realtà) . Ovviamente non c’è la certezza di aver preso il kernel giusto, bisognerà svilupparne vari, magari certi funzioneranno meglio di altri (incredibilmente facili da implementare, davvero).

Definito il problema possiamo a questo punto cercare il max del PROBLEMA DUALE, in che modo??? ***DEFLECTED SUBGRADIENT METHODS*** (su questo purtroppo non ho dato molto un’occhiata).

Trovato l’ottimo è fatta, abbiamo infatti visto sopra come w sia definito tramite i 2 insiemi di α.

E per trovare b? Nessun problema, ci basta prendere un punto che sia di errore o che sia comunque al margine dell’ ε-tube e possiamo calcolarcelo direttamente. Perché? Perché in quel punto uno dei 2 α è attivo essendo il valore al margine dell’area di constraint e dunque possiamo calcolarcelo:



Una volta che abbiamo w e b abbiamo finito.

ECCO IL MOMENTO DELLE MIE DOMANDE:

Una volta trainato che famo? La prediction io me la immagino così: si carica tutti gli elementi di test e facciamo una sorta di media tipo MEE o RMSE basata su ciò che ci dice il modello rispetto a ciò che ci dice l’expected value (molto nebuloso come aspetto).

Ste slack variables come le definiamo? Non le sappiamo a priori perché se le sapessimo avremmo già il modello in mano, non ci servono per l’ottimizzazione (vedi formula sopra). A mio parere l’unica possibilità è riuscire a definirli dopo l’ottimizzazione, a quel punto capiremo chi è dentro e chi è fuori la ε-tube. Uno potrebbe farsi la stessa domanda per C, però C effettivamente serve perché impone un vincolo su α (vincolo sempre per la derivazione, vedi foglio scritto a mano).

MANCA DA DISCUTERE (E NEL CASO MIO RIPASSARE) IL DEFLECTED SUBGRADIENT METHODS

MANCA RIUSCIRE A SCRIVERE UNO PSEUDO CODICE

PORCA EVA SE FRANGIONI HA RAGIONE: “LA MATEMATICA SOTTO FORMA DI IMPLEMENTAZIONE SCOMPARE”, INFATTI DAI CODICI IN GIRO NON SI CAPISCE UN CAZZO