**Avitan & Nesher** - 204945323, 204502926

**ML HW 5 - Theory + SVM - solution**

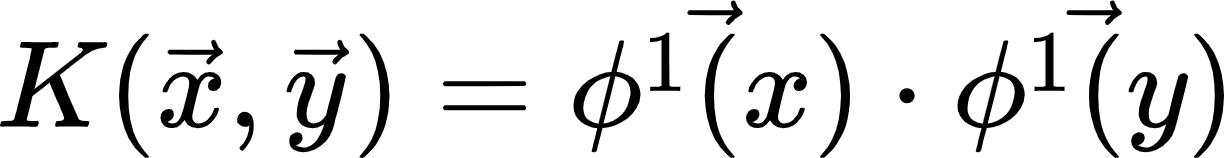
**a)** Let 𝐾, 𝐿 be two kernels (operating on the same space) and let 𝛼, 𝛽 be two positive scalars.

Prove that 𝛼𝐾 + 𝛽𝐿 is a kernel.

**Answer:**

Let be the feature map of K.

We say that is a kernel function iff there is a feature map such that for all x, y,



First, we show that is a valid kernel K’:

Let be the feature map of K’.

Then,

a valid kernel

Now we need to show that **the sum of the 2 kernels is a valid kernel**:

Denote:

as the feature map of K’

as the feature map of L’

Define by concatenating the feature maps (or alternate features if the spaces are infinite):

The mapping clearly satisfies

* <https://web.iitd.ac.in/~sumeet/CLT2008S-lecture18.pdf>

We showed that both characteristics hold and thus 𝛼𝐾 + 𝛽𝐿 is a kernel ⬛

**b)** Provide (two different) examples of non-zero kernels 𝐾, 𝐿 (operating on the same space), so that:

i. 𝐾 − 𝐿 is a kernel

ii. 𝐾 − 𝐿 is not a kernel

Prove your answers.

**Answer:**

Generally, a function is a valid kernel function (in the sense of the kernel trick) if it satisfies two key properties:

* **Symmetry**:
* **Positive semi-definiteness**

**i**.

For any d-dimensional vector , consider the identity mapping, s.t

→ **non-zero kernel**

Similarly for s.t

This is a valid kernel (the zero kernel) since it satisfies both properties above:

* Symmetry:
* Positive semi-definiteness: For any non-zero vector x:
* **A simpler example**:

Let K=2L where L is a valid kernel s.t K-L=L

K=2L is a valid kernel as proved in Q1a

**ii**.

Let be a **non-zero kernel** function:

(each term in the dot-product will be of the power of 2 hence positive).

Now, consider , a valid non-zero kernel (proved above in question 1a: )

And there is no such mapping that fulfills this condition.

Applying a kernel on the same vector x is always the sum of its values squared:

Hence we can never get a negative sum and so -K is not a kernel.

1. Use Lagrange multipliers to find the maximum and minimum values of the function subject to the given constraints:

Function:

Constraint: , where 𝛼 > 𝛽 > 0

**Answer:**

i.

ii.

iii.

iv.

From ii and iii we can refer that **y=z**

Multiplying equation i with y

Multiplying equation ii with x

Now we have

Considering the fact that 𝛼 > 𝛽 > 0, the possible solutions are:

Plugging in the constraint **g**:

**x = 0**:

**y = z = 0**:

Remember that

f reaches a **maximum** at

f reaches a **minimum** at

1. Let .

Let be the set of all origin-centered boxes.

Describe a polynomial sample complexity algorithm 𝐿 that learns 𝐶 using 𝐻.

State the time complexity and the sample complexity of your suggested algorithm.

Prove all your steps.

**Answer:**

The algorithm will produce a hypothesis which is the smallest relevant origin-centered box that contains all the positive points.

This can be done in O(m) (m = num. of instances) as follows:

Let be a set of points in the 3D space, labeled positive and negative.

Our algorithm seeks to return a hypothesis ℎ ∈ 𝐻.

Let be all positively labeled data points.

Find:

(the maximal x-axis distance from the origin in absolute value)

(the maximal y-axis distance from the origin in absolute value)

(the maximal z-axis distance from the origin in absolute value)

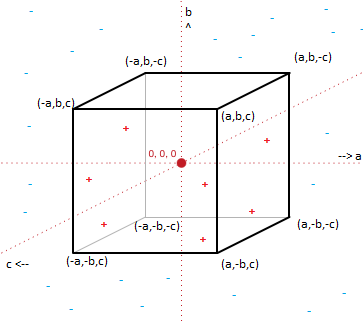
The box for will be .

Simply, this is an origin-centered box where the sizes a,b,c are the maximal distances we’ve seen in the training set.

It spans from to

This box is contained in the ground-truth box (concept c).

h:



* Time complexity is: O(m) for each value a,b,c => Total O(3m) = **O(m)**

Now we consider the area between our h to c (remember that ).

There are 6 such areas: for each of the coordinates and their negative sides .

* The areas and are actually symmetrical hence equal.

Consider training data, .

Assume that D visits each one of the 6 sets 𝐵, defined above.

What can we say about Err(ℎ, 𝑐)?

So the probability of a point in to be in **either** of those areas B\_i is

For a given 𝜀 and 𝛿, the number of samples needed is:

Meaning, when we want a confidence of to get an error of , we will need **at least** training instances.