# Solutions to Sheet 1

Friedrich May, 355487; Markus Moll, 406263; Mariem Mounir, 415862

April 29, 2020

## Exercise 1

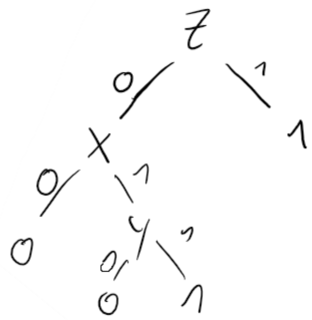
To calculate the class for every point we first need to find the $k$ closest points (depending on the norm). To archive this, a script like the following can be used:

```
function [ nbs ] = knn( spc,pnt,k, norm_p )
%KNN Summary of this function goes here
%   Detailed explanation goes here
  nbs = [];
  nonb = 0;
  while nonb < k
    if nonb == 0
      candidates = spc;
    else
      candidates = transpose(setdiff(transpose(spc),transpose(nbs),'rows'));
    end
    closestp = [0,0,0];
    closestn = inf;
    for i = 1:length(candidates)
      d =transpose(pnt-candidates(:,i));
      n=norm(d,norm_p);
      if n < closestn
        closestn = n;
        closestp = candidates(:,i);
      end
    end
    nonb = nonb+1;
    nbs(:,nonb) = closestp;
  end
end
```

Using the results the following table can be created. Note that the ties can be arbirarily chosen.

| Norm | Euclidian | | Manhattan | |
|---|---|---|---|---|
| k | 2 | 3 | 2 | 3 |
| 4,3,3 | 1 | 1 | 1 | 1 |
| 4,-1,1 | tie | -1 | tie | 1 |
| -2,4,5 | -1 | -1 | tie | 1 |
| -2,-6,1 | tie | 1 | tie | 1 |
| 6,0,2 | tie | 1 | -1 | -1 |

## Exercise2




The first split is done using $z$, because this is the only variable for wich the result stays the same for a value of the variable. The other two splits could be swapped because the influence on the result is the same for both variables.

## Exercise 3

### a)

We have

$$S = ((x_1, y_1), ..., (x_k, y_k))$$

with

$$x_i \in \{-1, 1\}^n, y_i \in \{-1, 1\} \, \forall i \in [k]$$

Let p be the number of updates needed for the Perceptron Algorithm.
So we have (1) :

$$\langle w_p, x_i y_i \rangle \geq 0 \; \forall i \in [k]$$

And we have also :

$$w_p - w_{p-1} = x_{p-1} y_{p-1}$$

So :

$$\langle w_p - w_{p-1}, x_{p-1} y_{p-1} \rangle = \langle x_{p-1} y_{p-1}, x_{p-1} y_{p-1} \rangle$$

$$\langle w_p - w_{p-1}, x_{p-1} y_{p-1} \rangle = n$$

$$\langle w_p, x_{p-1}y_{p-1}\rangle - \langle w_{p-1}, x_{p-1}y_{p-1}\rangle = n$$

Using (1) we have :

$$\langle w_{p-1}, x_{p-1}y_{p-1}\rangle + n \geq 0$$

$$\|w_{p-1}\| \, \|x_{p-1}y_{p-1}\| + n \geq \langle w_{p-1}, x_{p-1}y_{p-1}\rangle + n \geq 0$$

And we saw in the lecture that for each $i \leq p-1$ :

$$\|w_i\| \leq \sqrt{i}$$

Thus :

$$n + n^2(p-1) \geq n + n^2\sqrt{p-1} \geq 0$$

Finally:

$$1 - \frac{1}{n} \leq p$$

## b)

We have that the function:

$$maj(x_i) = \begin{cases} 1 & \text{if } \sum_{j=1}^{n} x_{ij} > 0 \\ -1 & \text{else} \end{cases}$$

If we have $\sum_{j=1}^{n} x_{ij} > 0$, we need to find $w$ such as

$$\langle w, x_i\rangle \geq 0 \ \forall i$$

$$\langle w, x_i\rangle = \sum_{j=1}^{n} w_j x_{ij}$$

for w=(1,..,1) we have

$$\langle w, x_i\rangle = \sum_{j=1}^{n} x_{ij} > 0$$

We need to normalize this w, so finally our normalized vector w is :

$$w = (\frac{1}{n}, ..., \frac{1}{n})$$

To find an upper bound on the number of updates of w the Perceptron Algorithm performs for any training set for the function maj, we need to find the margin:

$$min_{(x,y)\in S} \, |\langle w, x\rangle|$$

And S is a normalised set We have for each p:

$$|\langle w, x_p\rangle| = \left| \sum_{j=1}^{n} \frac{x_{pj}}{n^2} \right|$$

And because n is an odd number so will must have :

$$\frac{1}{n^2} \leq \left| \sum_{j=1}^{n^2} \frac{x_{pj}}{n^2} \right|$$

So we found a lower bound, we just need to find a vector that verify it, and a vector that have the majority positive or negative by 1 elements verify it.
So the margin is:

$$\gamma = \frac{1}{n^2}$$

Thus ( using theorem 1.10) we find that the upper bound is :

$$n^4 = \frac{1}{\gamma^2}$$

### c)

Yes, we will still find a linear separator that realizes maj. Because the Perceptron Algorithm has no restriction on the set.

## Exercise 4

### a)

The Certer is underlined.

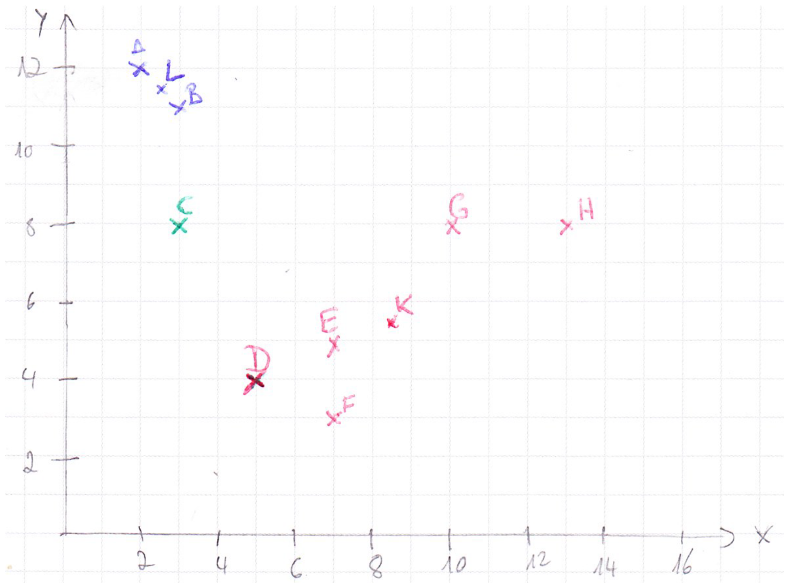
**Round 1**
Cluster 1: $\underline{A}$
Cluster 2: $\underline{B}$
Cluster 3: $\underline{C}, D, E, F, G, H$

**Round 2**
Cluster 1: $\underline{A}$
Cluster 2: $\underline{B}, C$
Cluster 3: $\underline{(7.5, 6)}, D, E, F, G, H$

**Round 3**
Cluster 1: $\underline{A}, B$
Cluster 2: $\underline{(3, 9.5)}, C$
Cluster 3: $\underline{(8.5, 5.6)}, D, E, F, G, H$

**Round 4**
Cluster 1: $\underline{(3.5, 11.5)}, A, B$
Cluster 2: $\underline{C}$
Cluster 3: $\underline{(8.5, 5.6)}, D, E, F, G, H$
The clustering of the Points stays the same as the roud before so the algorithm halts.

**b)**




$L$ is mean of cluster 1: $(2.5, 11.5)$, $C$ is mean of cluster 2: $(3, 8)$, K is mean of cluster 3: $(8.5, 5.6)$