Exercise 1 (Nearest Neighbour Classification) 4 points

We consider the k-nearest neighbour classification algorithm in 3-dimensions. Given is a training set consisting of 20 examples together with their classification (1 or -1), and a list of 5 query points. These can be found in the file `nn.py` that has been uploaded along with this sheet.

Classify the 5 query points using the k-nearest neighbour algorithm, for each of the following four configurations:

a) $k = 2$ with Manhattan distance,

b) $k = 3$ with Manhattan distance.

c) $k = 2$ with Euclidean distance,

d) $k = 3$ with Euclidean distance,

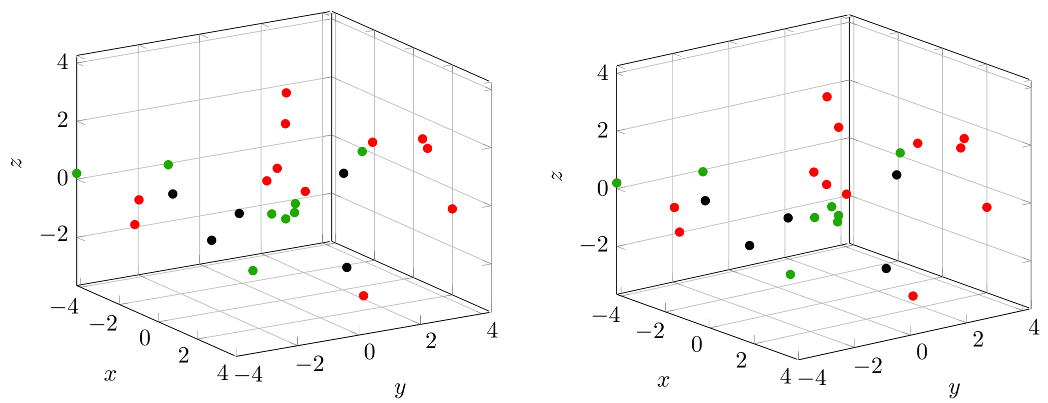
When ties occur, indicate them with class label „0“.

Hint: Solve this by writing a program that does the job for you.

- Give the results of your classifications in form of a table.
- You do not need to worry about the precision of representations of real numbers.
- You do not need to turn in your code (code submissions will be ignored, only the answers count).

Solution:

The data set looks as follows (class 1 in green, class -1 in red, query points in black). (Cross-view.)



The algorithms return the following classifications. (For ease of representation, we treat all points as row vectors.)

		$(1,-2,0)$	$(4,-0.5,2)$	$(1,1.5,-2.5)$	$(-2,-1,-2)$	$(-4,-1,-1)$
$k=2$	Euclidean	1	-1	1	0	0
	Manhattan	1	-1	0	0	0
$k=3$	Euclidean	1	-1	1	1	-1
	Manhattan	1	-1	1	-1	1

The distances are as follows:

Point: (1, -2, 0)

Example	Euclidean	Manhattan
(-4, -2.1, -1, -1)	5.1000000000000005	6.1
(-3.6, -1.4, 0.2, 1)	4.643274706497559	5.3999999999999995
(1, -0.2, -0.3, 1)	1.8248287590894658	2.1
(0.3, -0.5, -0.5, 1)	1.7291616465790582	2.7
(-2, -3.5, -1, -1)	3.5	5.5
(-4.2, -4, 0.2, 1)	5.574943945906542	7.4
(-1.3, -0.1, -3, 1)	4.230839160261236	7.199999999999999
(-0.7, 0.9, -0.7, 1)	3.4336569426778794	5.3
(1, 2, 1.4, 1)	4.237924020083418	5.4
(2.6, -1.5, 0.2, 1)	1.6881943016134133	2.3000000000000003
(2, 4.3, -0.7, -1)	6.417164482853778	8.0
(0.6, 0.4, 0.2, -1)	2.4413111231467406	3.0
(2.9, -1.7, 3.6, -1)	4.08166632639171	5.800000000000001
(3.6, 0.4, -2.5, -1)	4.332435804486894	7.5
(1.2, 4, 1.2, -1)	6.1220911460055865	7.4
(-1, 0.5, 0.5, -1)	3.24037034920393	5.0
(3, 2.7, 2.3, -1)	5.601785429664368	9.0
(4, -3, 2.2, -1)	3.8522720568516444	6.2
(0.1, 0.1, 3.5, -1)	4.179712908801273	6.5
(2.8, 1.2, 2.4, -1)	4.386342439892262	7.4

Point: (4, -0.5, 2)

Example	Euclidean	Manhattan
(-4, -2.1, -1, -1)	8.692525524840292	12.6
(-3.6, -1.4, 0.2, 1)	7.861933604400384	10.3
(1, -0.2, -0.3, 1)	3.792097045171708	5.6
(0.3, -0.5, -0.5, 1)	4.465422712353221	6.2
(-2, -3.5, -1, -1)	7.3484692283495345	12.0
(-4.2, -4, 0.2, 1)	9.095603333479312	13.5
(-1.3, -0.1, -3, 1)	7.297259759663212	10.7
(-0.7, 0.9, -0.7, 1)	5.598214000911362	8.8
(1, 2, 1.4, 1)	3.950949253027682	6.1
(2.6, -1.5, 0.2, 1)	2.4899799195977463	4.2
(2, 4.3, -0.7, -1)	5.859180830116101	9.5
(0.6, 0.4, 0.2, -1)	3.950949253027682	6.1
(2.9, -1.7, 3.6, -1)	2.282542442102666	3.9
(3.6, 0.4, -2.5, -1)	4.606517122512408	5.8
(1.2, 4, 1.2, -1)	5.360037313302959	8.1
(-1, 0.5, 0.5, -1)	5.315072906367325	7.5
(3, 2.7, 2.3, -1)	3.366006535941367	4.5
(4, -3, 2.2, -1)	2.5079872407968904	2.7
(0.1, 0.1, 3.5, -1)	4.221374183841086	6.0
(2.8, 1.2, 2.4, -1)	2.118962010041709	3.3000000000000003

Point: (1, 1.5, -2.5)

Example	Euclidean	Manhattan
(-4, -2.1, -1, -1)	6.341135544995076	10.1
(-3.6, -1.4, 0.2, 1)	6.071243694664216	10.2
(1, -0.2, -0.3, 1)	2.7802877548915688	3.9000000000000004
(0.3, -0.5, -0.5, 1)	2.9137604568666933	4.7
(-2, -3.5, -1, -1)	6.020797289396148	9.5
(-4.2, -4, 0.2, 1)	8.036168241145777	13.399999999999999
(-1.3, -0.1, -3, 1)	2.8460498941515415	4.4
(-0.7, 0.9, -0.7, 1)	2.5475478405713994	4.1
(1, 2, 1.4, 1)	3.9319206502674997	4.4
(2.6, -1.5, 0.2, 1)	4.341658669218482	7.3
(2, 4.3, -0.7, -1)	3.4756294393965534	5.6
(0.6, 0.4, 0.2, -1)	2.9427877939124323	4.2
(2.9, -1.7, 3.6, -1)	7.145628033979938	11.2
(3.6, 0.4, -2.5, -1)	2.823118842698621	3.7
(1.2, 4, 1.2, -1)	4.469899327725402	6.4
(-1, 0.5, 0.5, -1)	3.7416573867739413	6.0
(3, 2.7, 2.3, -1)	5.336665625650533	8.0
(4, -3, 2.2, -1)	7.165193647069143	12.2
(0.1, 0.1, 3.5, -1)	6.226556030423239	8.3
(2.8, 1.2, 2.4, -1)	5.228766584960549	7.0

Point: (-2, -1, -2)

Example	Euclidean	Manhattan
(-4, -2.1, -1, -1)	2.4919871588754225	4.1
(-3.6, -1.4, 0.2, 1)	2.749545416973504	4.2
(1, -0.2, -0.3, 1)	3.539774004085572	5.5
(0.3, -0.5, -0.5, 1)	2.7910571473905725	4.3
(-2, -3.5, -1, -1)	2.692582403567252	3.5
(-4.2, -4, 0.2, 1)	4.3220365569948616	7.4
(-1.3, -0.1, -3, 1)	1.51657508881031	2.6
(-0.7, 0.9, -0.7, 1)	2.6438608132804573	4.5
(1, 2, 1.4, 1)	5.436910887627275	9.4
(2.6, -1.5, 0.2, 1)	5.123475382979799	7.3
(2, 4.3, -0.7, -1)	6.766091929614909	10.600000000000001
(0.6, 0.4, 0.2, -1)	3.6823905279043943	6.2
(2.9, -1.7, 3.6, -1)	7.473954776421918	11.2
(3.6, 0.4, -2.5, -1)	5.793962374748389	7.5
(1.2, 4, 1.2, -1)	6.743886120034946	11.399999999999999
(-1, 0.5, 0.5, -1)	3.082207001484488	5.0
(3, 2.7, 2.3, -1)	7.561745830163825	13.0
(4, -3, 2.2, -1)	7.592101158440923	12.2
(0.1, 0.1, 3.5, -1)	5.989156868875618	8.7
(2.8, 1.2, 2.4, -1)	6.8731361109758335	11.4

Point: (-4, -1, -1)

Example	Euclidean	Manhattan
(-4, -2.1, -1, -1)	1.1	1.1
(-3.6, -1.4, 0.2, 1)	1.3266499161421599	1.9999999999999998
(1, -0.2, -0.3, 1)	5.111751167652823	6.5
(0.3, -0.5, -0.5, 1)	4.357751713900185	5.3
(-2, -3.5, -1, -1)	3.2015621187164243	4.5
(-4.2, -4, 0.2, 1)	3.237282811247729	4.4
(-1.3, -0.1, -3, 1)	3.4785054261852175	5.6
(-0.7, 0.9, -0.7, 1)	3.8196858509568554	5.499999999999999
(1, 2, 1.4, 1)	6.305553108173778	10.4
(2.6, -1.5, 0.2, 1)	6.726812023536855	8.299999999999999
(2, 4.3, -0.7, -1)	8.011242100947893	11.600000000000001
(0.6, 0.4, 0.2, -1)	4.955804677345546	7.2
(2.9, -1.7, 3.6, -1)	8.322259308625274	12.2
(3.6, 0.4, -2.5, -1)	7.87210264160726	10.5
(1.2, 4, 1.2, -1)	7.541883053985922	12.399999999999999
(-1, 0.5, 0.5, -1)	3.6742346141747673	6.0
(3, 2.7, 2.3, -1)	8.577878525602936	14.0
(4, -3, 2.2, -1)	8.845337754998393	13.2
(0.1, 0.1, 3.5, -1)	6.186275131288617	9.7
(2.8, 1.2, 2.4, -1)	7.914543574963751	12.4

Exercise 2 (Decision Trees) **2 + 2 + 2 = 6 points**

Recall that a propositional formula is built from Boolean variables $X_1, \dots, X_n$ using $\neg$, $\wedge$ and $\vee$. For all variables X, the formulas X and $\neg X$ are called *(positive / negative) literals*. Every propositional formula with n variables represents a Boolean function from $\{0,1\}^n$ to $\{0,1\}$, via the usual notion of (satisfying or non-satisfying) assignments.

A propositional formula is in *k-CNF*, if it is a conjunction of disjunctions of at most k literals, that is, of the shape $\bigwedge_{i=1}^{m} \bigvee_{j=1}^{m_i} L_{i,j}$ with $m_i \leq k$ for all $i = 1, \dots, m$.

A propositional formula is in *k-DNF*, if it is a disjunction of conjunctions of at most k literals, that is, of the shape $\bigvee_{i=1}^{m} \bigwedge_{j=1}^{m_i} L_{i,j}$ with $m_i \leq k$ for all $i = 1, \dots, m$.

Answer the following tasks.

a) Give an example of a 2-CNF over 3 variables X_1, X_2, X_3 that has no 2-DNF representation. Explain why it has no 2-DNF representation.

b) Now give an example of a 2-DNF over X_1, X_2, X_3 that has no 2-CNF representation. Explain why it has no 2-CNF representation.

c) Prove the following statement: If f is a Boolean function that can be represented by a decision tree of height $k \in \mathbb{N}$, then f can be represented by both a k-CNF and a k-DNF.

Solution:

a) We consider the function f represented by $X_1 \wedge X_2 \wedge X_3$. This is a 2-CNF (even a 1-CNF). Then $f(x_1, x_2, x_3) = 1$ if and only if $x_1 = x_2 = x_3 = 1$. A (2-)DNF is satisfied if at least one of its disjuncts is satisfied. Any disjunct over 2 literals (out of 3 variables) has at least 2 satisfying assignments. Thus, every 2-DNF has at least 2 satisfying assignments. Therefore, f has no 2-DNF representation.

b) We consider the function f represented by $\neg X_1 \vee \neg X_2 \vee \neg X_3$. This is a 2-DNF (even a 1-DNF). Then $f(x_1, x_2, x_3) = 0$ if and only if $x_1 = x_2 = x_3 = 1$. A (2-)CNF is not satisfied if at least one of its conjuncts is not satisfied. Any disjunct over 2 literals (out of 3 variables) has at least 2 non-satisfying assignments. Thus, every 2-CNF has at least 2 non-satisfying assignments. Therefore, f has no 2-CNF representation.

c) Let f be a Boolean function over n variables that is represented by a decision tree of height k. Every path in the decision tree corresponds to a conjunction of at most k literals (node = variable; outgoing 0-edge = negative literal; outgoing 1-edge = positive literal). Our k-DNF is the disjunction over all such paths that lead to leaves labelled with 1.

If in the decision tree for f, we swap all 1- and 0-leaves, we obtain a decision tree representation of the function f' where $f'(x_1, x_2, x_3) = 1 - f(x_1, x_2, x_3)$ for all

$x_1, x_2, x_3 \in \{0,1\}^n$. By the previous result, f' is representable by a k-DNF. The negation of a k-DNF is a k-CNF (using De Morgans laws and the idempotence of $\neg$). Thus, f has a k-CNF representation.

Exercise 3 (Perceptron) **1+1+1+2=5 points**

We want to use the Perceptron algorithm for linear classification on the instance space $\{-1, 1\}^n$ where $n \in \mathbb{N}$ is an odd number. The target function (unknown to the algorithm) is the *majority function* $\mathrm{maj}\colon \{-1,1\}^n \to \{-1,1\}$ with

$$\mathrm{maj}(\mathbf{x}) = \begin{cases} 1 & \text{if more than } n/2 \text{ of } \mathbf{x}\text{'s entries are positive} \\ -1 & \text{otherwise.} \end{cases} \tag{$*$}$$

We consider a training sequence $S = \big((\mathbf{x}_1, y_1), \dots, (\mathbf{x}_k, y_k)\big)$ of $k \in \mathbb{N}$ data items $\mathbf{x}_1, \dots, \mathbf{x}_k \in \{-1,1\}^n$ together with their class labels $y_1, \dots, y_k \in \{-1, 1\}$.

a) Show that $\mathrm{maj}\colon \{-1,1\}^n \to \{-1,1\}$ is realisable by a homogenous linear separator by specifying a suitable weight vector $\widehat{\mathbf{w}}$ satisfying $\mathrm{maj}(\mathbf{x}) = \mathrm{sgn}\big(\langle \widehat{\mathbf{w}}, \mathbf{x}\rangle\big)$ for all $\mathbf{x} \in \{-1,1\}^n$.

b) Using the weight vector from part a), derive an upper bound on the number of weight vector updates $\mathbf{w} \leftarrow \mathbf{w} + y\mathbf{x}$ the Perceptron algorithm performs when run on S.

c) Find the smallest possible number of updates $\mathbf{w} \leftarrow \mathbf{w} + y\mathbf{x}$ after which the Perceptron algorithm terminates. For all $k \in \mathbb{N}$, $k \geq 1$, describe a training sequence S (with k examples) that for which the algorithm achieves this lower bound, and argue that it does.

d) If the domain of the target function is extended to $\mathbb{R}^n$ (leaving the definition $(*)$ unchanged), can we still find a consistent linear separator for any given training sequence S?

Solution:

a) Let $\widehat{\mathbf{w}} = \mathbf{1}$ be the all ones vector. It holds that

$$\langle \widehat{\mathbf{w}}, \mathbf{x}\rangle = x_1 + \cdots + x_n \begin{cases} > 0 & \text{if } \mathrm{maj}(\mathbf{x}) = 1 \\ < 0 & \text{if } \mathrm{maj}(\mathbf{x}) = -1. \end{cases}$$

Thus, the function $\mathbf{x} \mapsto \mathrm{sgn}\big(\langle \widehat{\mathbf{w}}, \mathbf{x}\rangle\big)$ is equal to maj.

b) Let $\widehat{h}\colon \mathbf{x} \mapsto \mathrm{sgn}\big(\langle \widehat{\mathbf{w}}, \mathbf{x}\rangle\big)$ be the hypothesis defined by $\widehat{\mathbf{w}}$. The margin γ of $\widehat{h}$ with respect to S is

$$\min_{(\mathbf{x},y)\in S} \frac{|\langle \widehat{\mathbf{w}}, \mathbf{x}\rangle|}{\|\widehat{\mathbf{w}}\|} = \min_{(\mathbf{x},y)\in S} \frac{|x_1 + \cdots + x_n|}{\sqrt{n}} \geq \frac{1}{\sqrt{n}}$$

As $\lambda \coloneqq \max_{(\mathbf{x},y)\in S} \|x\| = \sqrt{n}$, Theorem 1.13 of the lecture yields an upper bound of $\left(\frac{\lambda}{\gamma}\right)^2 \leq \frac{\sqrt{n}}{1/\sqrt{n}} = n^2$ rounds.

c) The smallest possible number of updates is 1. First note that the initial weight vector $\mathbf{0}$ never yields a consistent hypothesis (cf. slide 1.49).

Now for the weight vector $\widehat{w} = \mathbf{1}$ from a), it holds that $\mathrm{maj}(\mathbf{1}) = 1$. If the training sequence contains $(\mathbf{1}, 1)$ as an example, and the algorithm happens to check this example first, the first update of the weight vector is $\mathbf{w} \leftarrow \mathbf{0} + \mathbf{1} = \mathbf{1}$. The resulting hypothesis is then equal to maj, so, in particular, consistent with S.

d) The answer is no. Suppose that $\mathbf{w}$ and b satisfy $\mathrm{sgn}\big(\langle \mathbf{w}, \mathbf{x}\rangle - b\big) = \mathrm{maj}(\mathbf{x})$ for all $\mathbf{x} \in \mathbb{R}^n$. Then it directly follows that $\mathbf{w} \neq \mathbf{0}$. Without loss of generality, we assume that $w_1 \neq 0$.

Case $n = 1$. Let $n = 1$. The function value of $\mathrm{maj}(\mathbf{x})$ changes at $\mathbf{x} = 0$ with $\mathrm{maj}(0) = -1$. For every linear function $wx - b$, the sign changes at $x_0 = b/w$. If $b = 0$, then $\mathrm{sgn}(w \cdot 0 - b) = 0$ but $\mathrm{maj}(0) = -1$. If $b \neq 0$, consider the point $x_0 = b/w$. As x_0 lies between $\frac{1}{2}x_0$ and $2x_0$, it holds that $\mathrm{sgn}(w \cdot \frac{1}{2}x_0 - b) \neq \mathrm{sgn}(w \cdot 2x_0 - b)$ (and both sides are different from 0). This however, contradicts $\mathrm{maj}\left(\frac{1}{2}x_0\right) = \mathrm{maj}(x_0) = \mathrm{maj}(2x_0)$.

Case $n \geq 3$. We set

$$Z \coloneqq w_2 + w_3 + \cdots + w_n$$

and consider the data item

$$\mathbf{x} = \left(-\frac{|Z - b| + \varepsilon}{w_1}, 1, \ldots, 1 \right) \in \mathbb{R}^n$$

for some $\varepsilon > 0$. Since $n \geq 3$ it holds that $\mathrm{maj}(\mathbf{x}) = 1$. However, it also holds that

$$\begin{aligned}
\langle \mathbf{w}, \mathbf{x} \rangle - b &= w_1 x_1 + \sum_{i=2}^{n} w_i x_i - b \\
&= w_1 \left(-\frac{|Z - b| + \varepsilon}{w_1} \right) + Z - b \\
&= -|Z - b| + Z - b - \varepsilon \leq -\varepsilon < 0,
\end{aligned}$$

so $\mathrm{sgn}\big(\langle \mathbf{w}, \mathbf{x} \rangle - b\big) = -1 \neq \mathrm{maj}(\mathbf{x})$, a contradiction.

Exercise 4 (k-Means Algorithm) **2+1+2=5 points**

Consider the following set of points in $\mathbb{R}^2$ and the execution of the 3-Means Algorithm on it.

$$S = \{A = (2, 12), B = (3, 11), C = (3, 8), D = (5, 4),\\ E = (7, 5), F = (7, 3), G = (10, 8), H = (13, 8)\}$$

a) Give all intermediate clusters and their centers during the execution of the 3-means algorithm on S with the initial cluster means $z^1 = A$, $z^2 = B$, $z^3 = C$.

b) Draw a coordinate system and, in it, indicate (1) the points of S; (2) the three final cluster means; and (3) the three final cluster regions (i. e. the set of points which are closer to the corresponding centroid than to any of the other two).

Describe the lines that separate the regions algebraically as linear equations. Justify the correctness of your expressions.

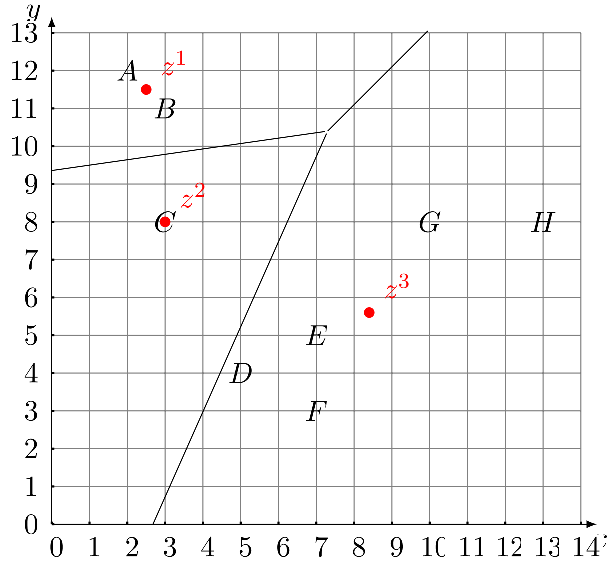
c) Find a different initialisation of the centroids (as opposed to part (a)) for which the execution of the 3-means algorithm yields a different set of final clusters. Justify your solution.

Solution:

a) After three rounds the algorithm terminates.

initial centroids:	$z^1 = A$, $z^2 = B$, $z^3 = C$
first round:	$C_1 = \{A\}$,$C_2 = \{B\}$, $C_3 = \{C, \dots, H\}$
	$z^1 = A$, $z^2 = B$, $z^3 = (7.5, 6)$
second round:	$C_1 = \{A\}$,$C_2 = \{B, C\}$, $C_3 = \{D, \dots, H\}$
	$z^1 = A$, $z^2 = (3, 9.5)$, $z^3 = (8.4, 5.6)$
third round:	$C_1 = \{A, B\}$,$C_2 = \{C\}$, $C_3 = \{D, \dots, H\}$
	$z^1 = (2.5, 11.5)$, $z^2 = C$, $z^3 = (8.4, 5.6)$

b) The separation between C_1 and C_2 is $f(x) = 1/7x + 131/14$,
the separation between C_1 and C_3 is $f(x) = x + 31/10$, and
the separation between C_2 and C_3 is $f(x) = 9/4x - 241/40$



Formula for the separation (= perpendicular bisector) between two centroids (x_1, y_1) and (x_2, y_2): $f(x) = -\frac{x_1-x_2}{y_1-y_2}x + \frac{x_1^2-x_2^2+y_1^2-y_2^2}{2(y_1-y_2)}$ (provided that $y_1 \neq y_2$; otherwise, it's $x = \frac{x_1+x_2}{2}$)

c) E.g. initial cluster means B, F, H result in clusters $C_1 = \{A, B, C\}$, $C_2 = \{D, E, F\}$, $C_3 = \{G, H\}$. It is easy to check that this clusting is obtained after one round and that afterwards, no more updates happen.