Learning gLMPM - generalized LMPM

Peter Vojtas

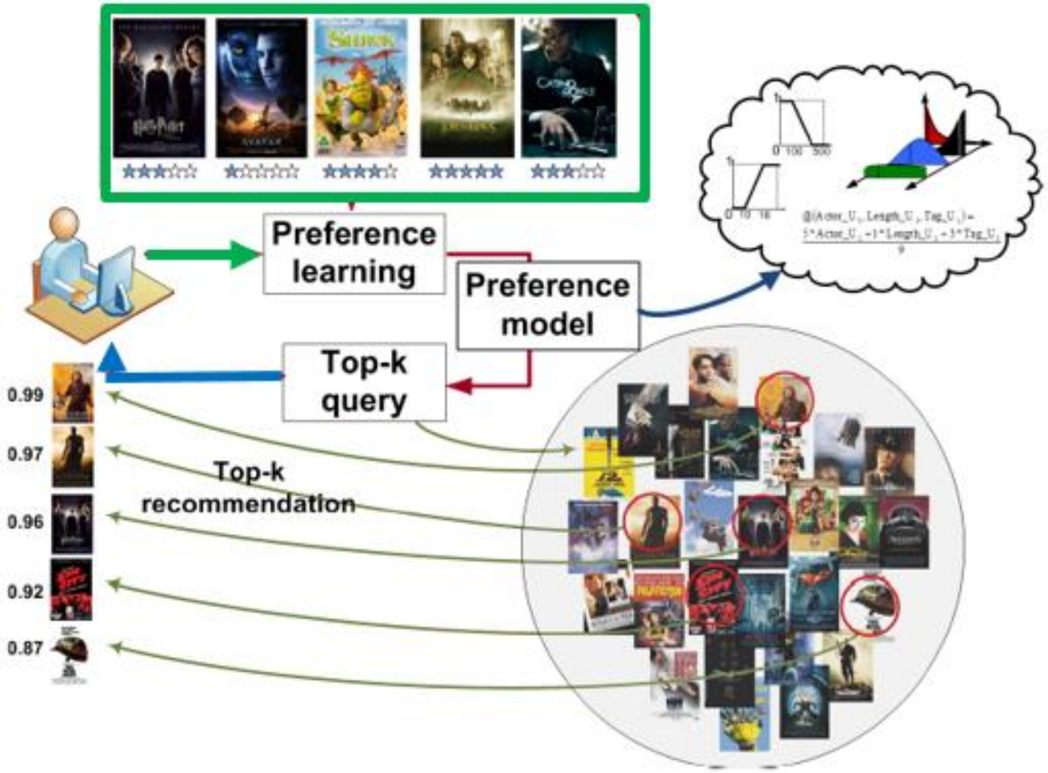
Content

- We have extended the portfolio of models in hope some of them will fit to user's behavior. Let's try to find methods to learn from data generated by respective models
- There was some learning of LMPM preferences in NSWI166 – specific methods under specific circumstances of data distribution, here we learn gLMPM models, still based on a method finding ideal points and methods finding aggregation or at least PC contour lines
- Our experiments involve:
  - Experiment 1. LMPM learning, more general data distribution, hill/valley attribute preferences
  - Experiment 2. Data generated by product disjunction on respective PC contour lines
  - Experiment 3. Data generated by product conjunction on respective PC contour lines
  - Experiment 4. Arbitrary Pareto compliant contour lines

User's preference learning

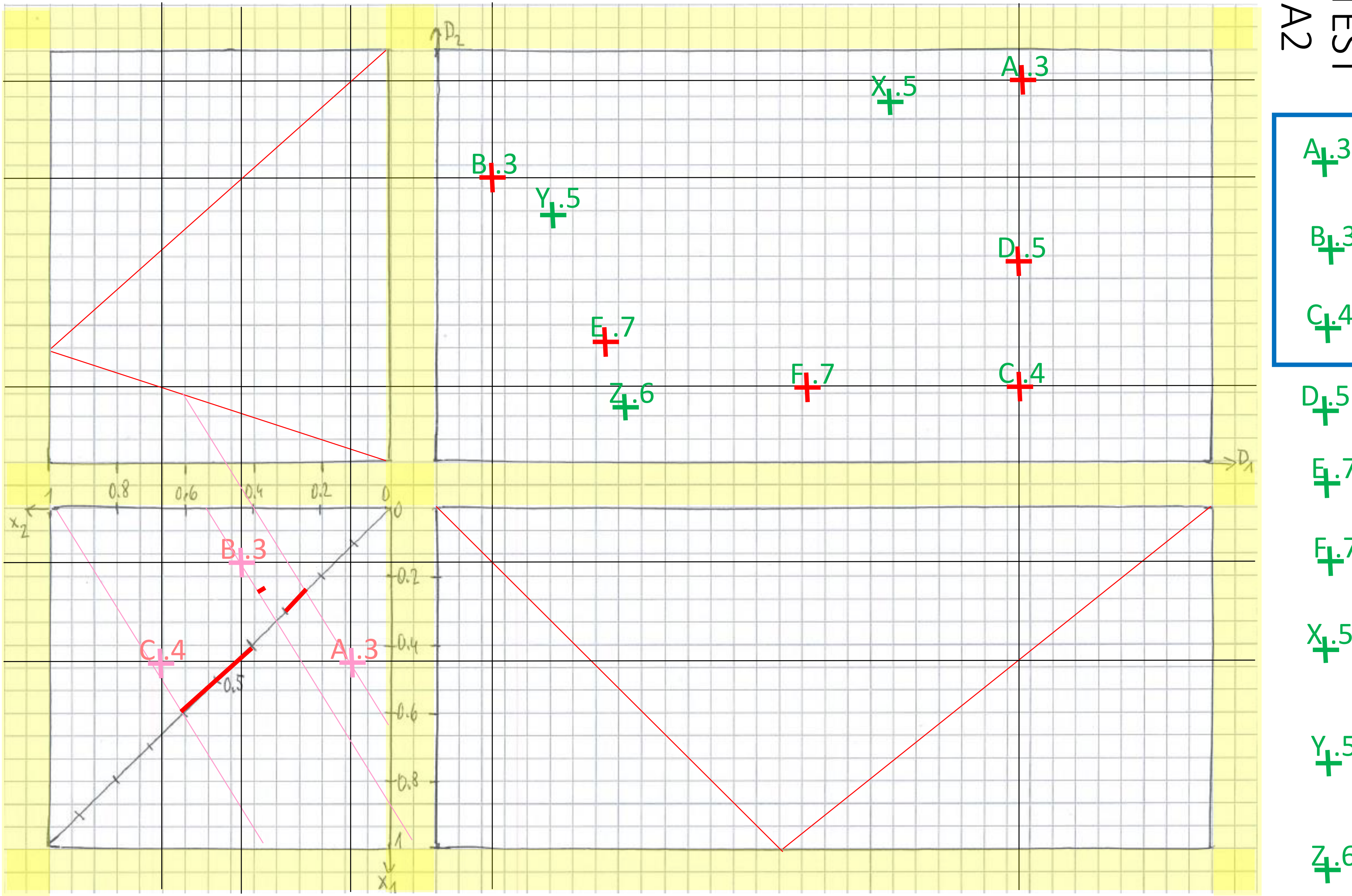
- We have
  - User behavior

- Would like to have an
  - gLMPM user model to compute top-k for recommendation
- Can we make this recommendation visual? Human intuitive?



Learning from NSWI166, go to slides 3-40

C1 TEST
M2 A2



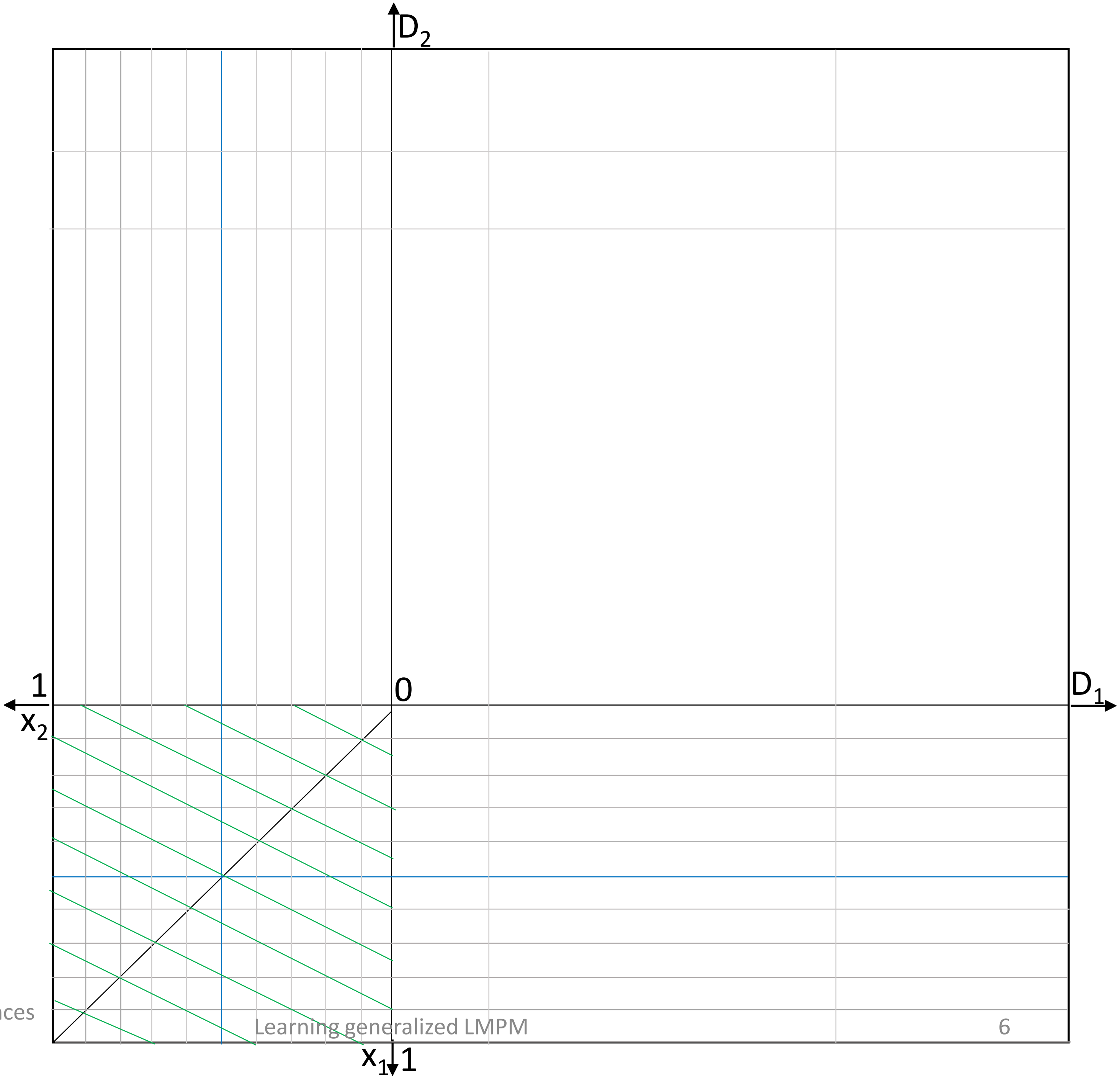
Experiment 1.

LMPM learning, more general data distribution, hill/valley attribute preferences

Ex.1 can be hill – valley
We would like to generate data in such a way that learning aggr. needs a new method

There will be no data with same preference

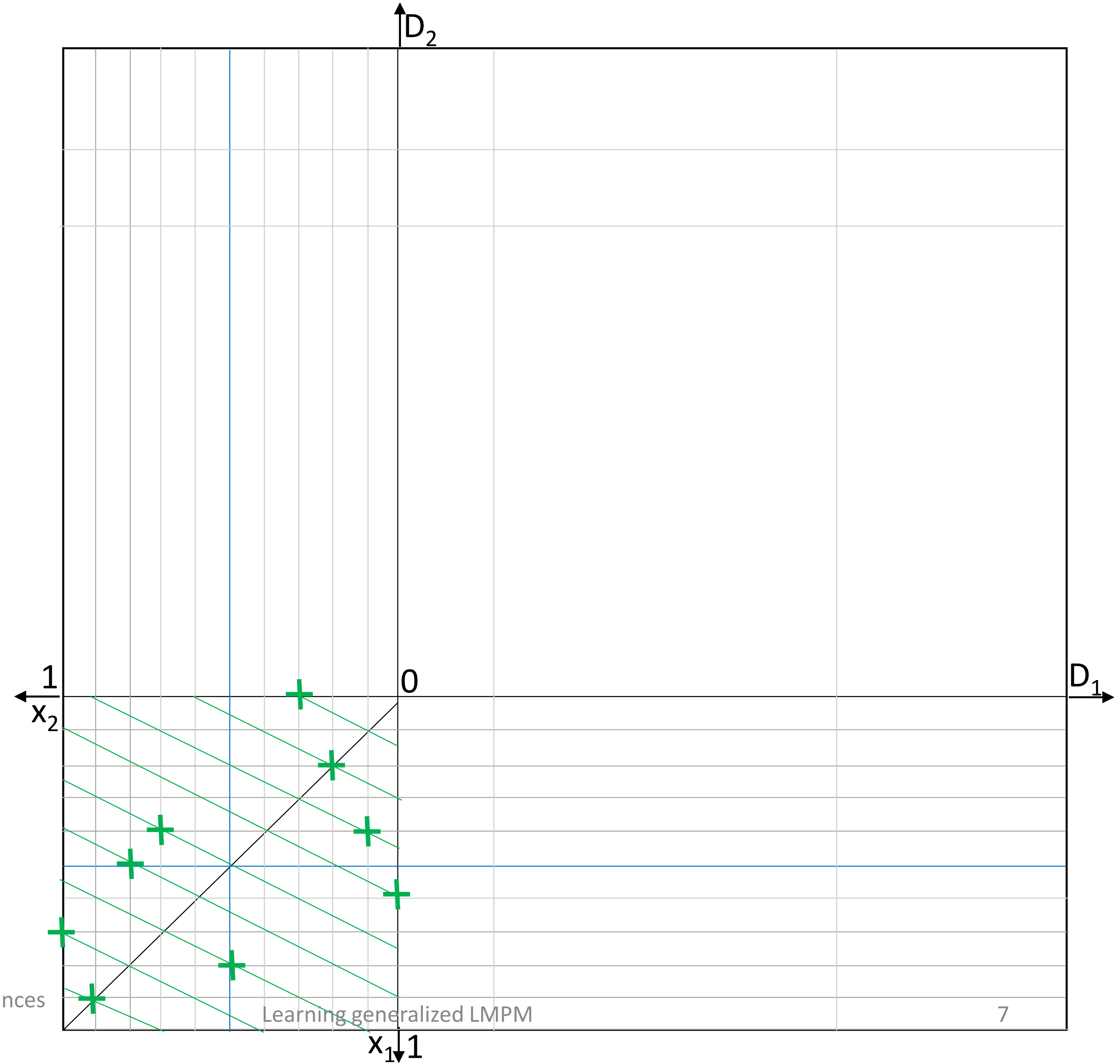
Choose aggr.
$(2x_1 + x_2)/3$



Ex.1 can be hill – valley , new @ methods needed

Draw all decimal valued contour lines of $(2x_1 + x_2)/3$ And chose one point on each.

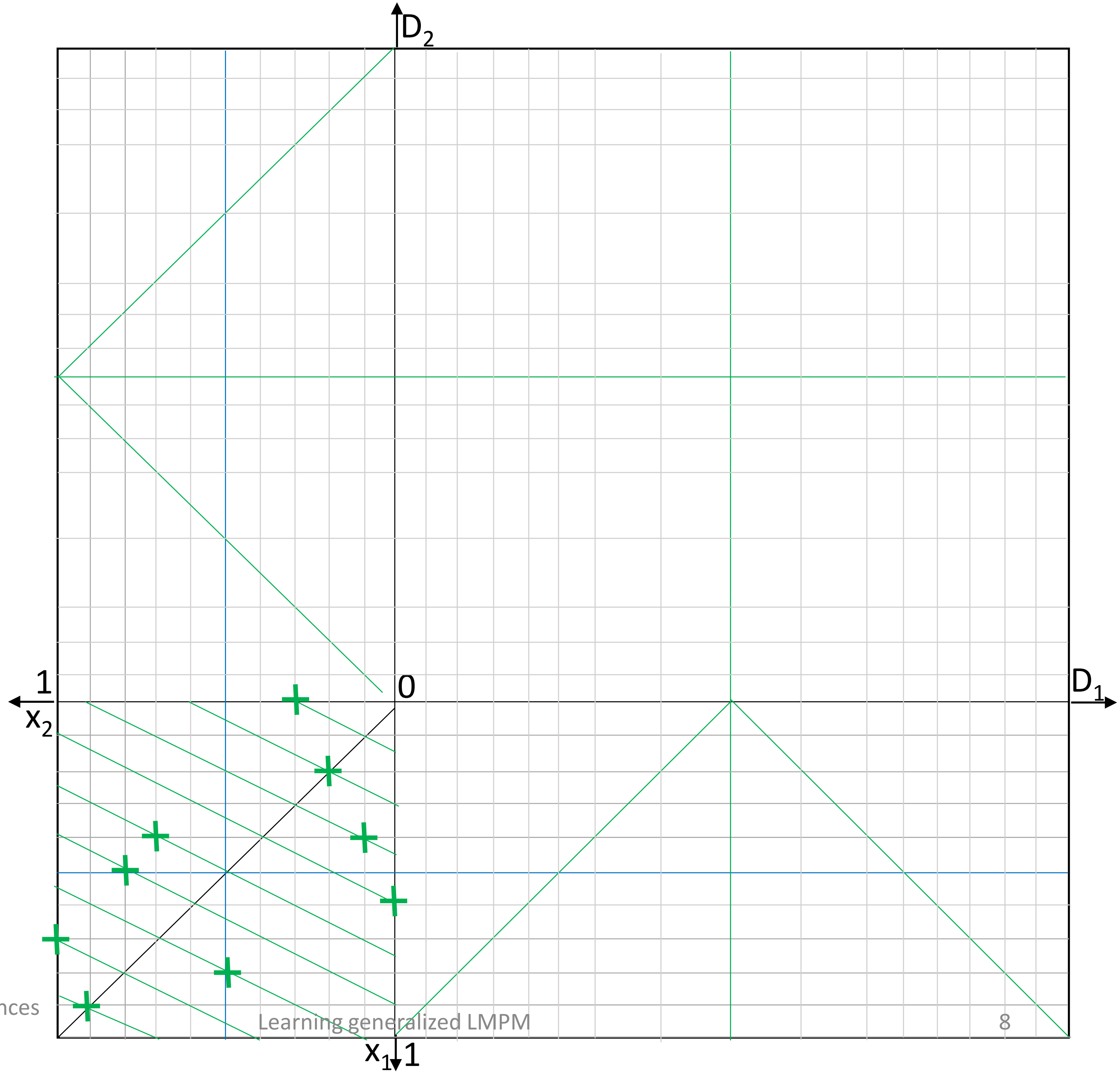
Only for copying



Ex.1 can be hill – valley , new @ methods needed

We have decimal valued contour lines of $(2x_1 + x_2)/3$ points on each.

Chose attr. prefs. hill/valley



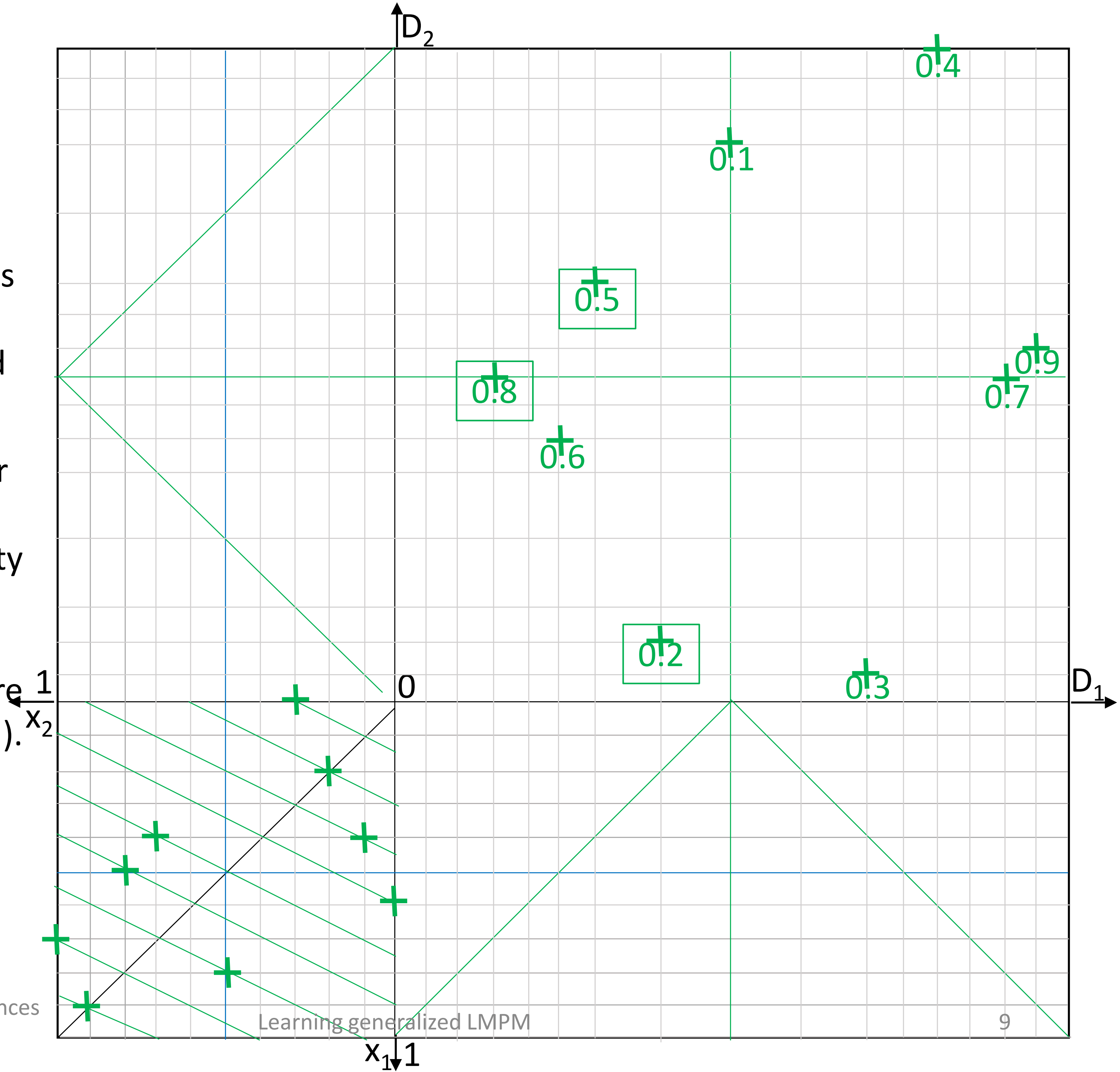
Ex.1 can be hill – valley , new @ methods needed

We have decimal valued contour lines of $(2x_1 + x_2)/3$ points on each. and attr. prefs.

Each point has four coimages in DC – chose one possibility rotating choice in quadrants (this can be changed in future data set generation).

Pref.score is also ID

Fix also train/ test set.

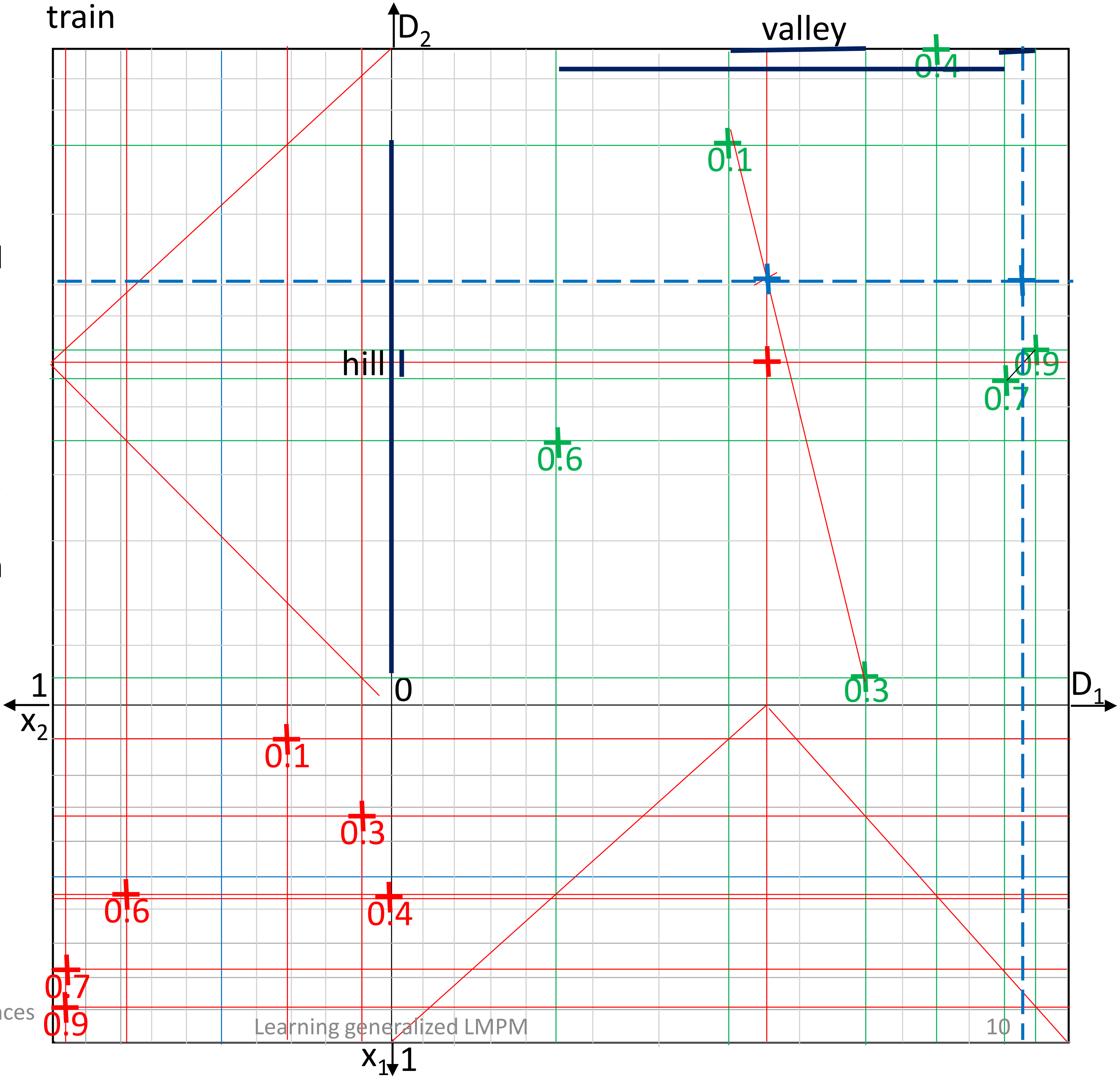


Ex.1 method i1 First decide hill / valley – projection smaller / bigger.

To learn an LMPM model, first a method i1 for atr.prefs.

We find four candidates for peak / valley points by center of mass (min and max pairs). Chose a red one, more regularly dividing train set

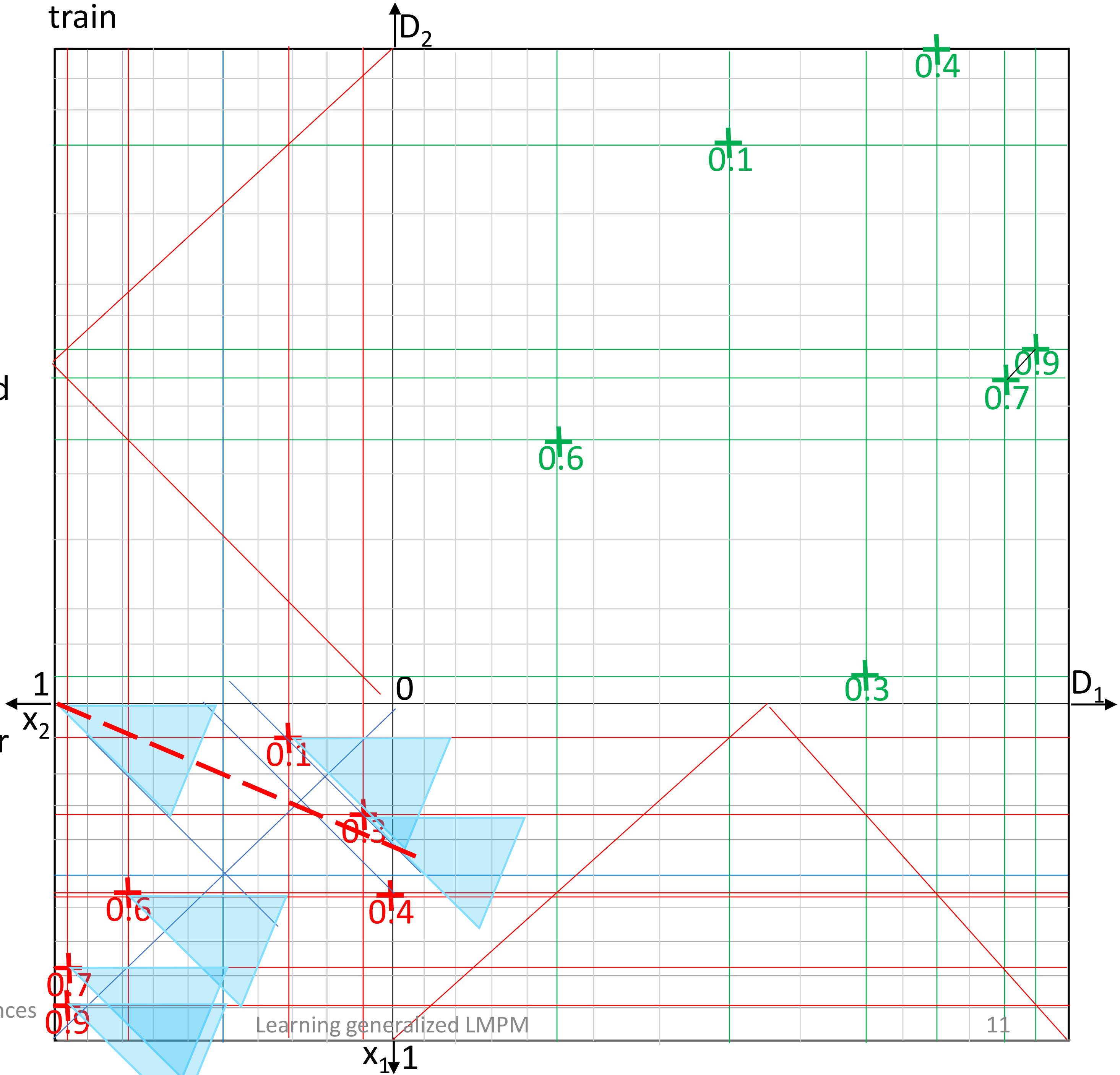
Find candidates in PC



Ex.1.1 **method i1. a1**. Learning gives attr. prefs and training set images in PC. Points are Pareto compliant

Let's first try to find a max NW-SE **angle for all train points preserves Pareto (0.1 to 0.3)**

Angle method **a1**: Let's chose angle axis as our guess for aggregation



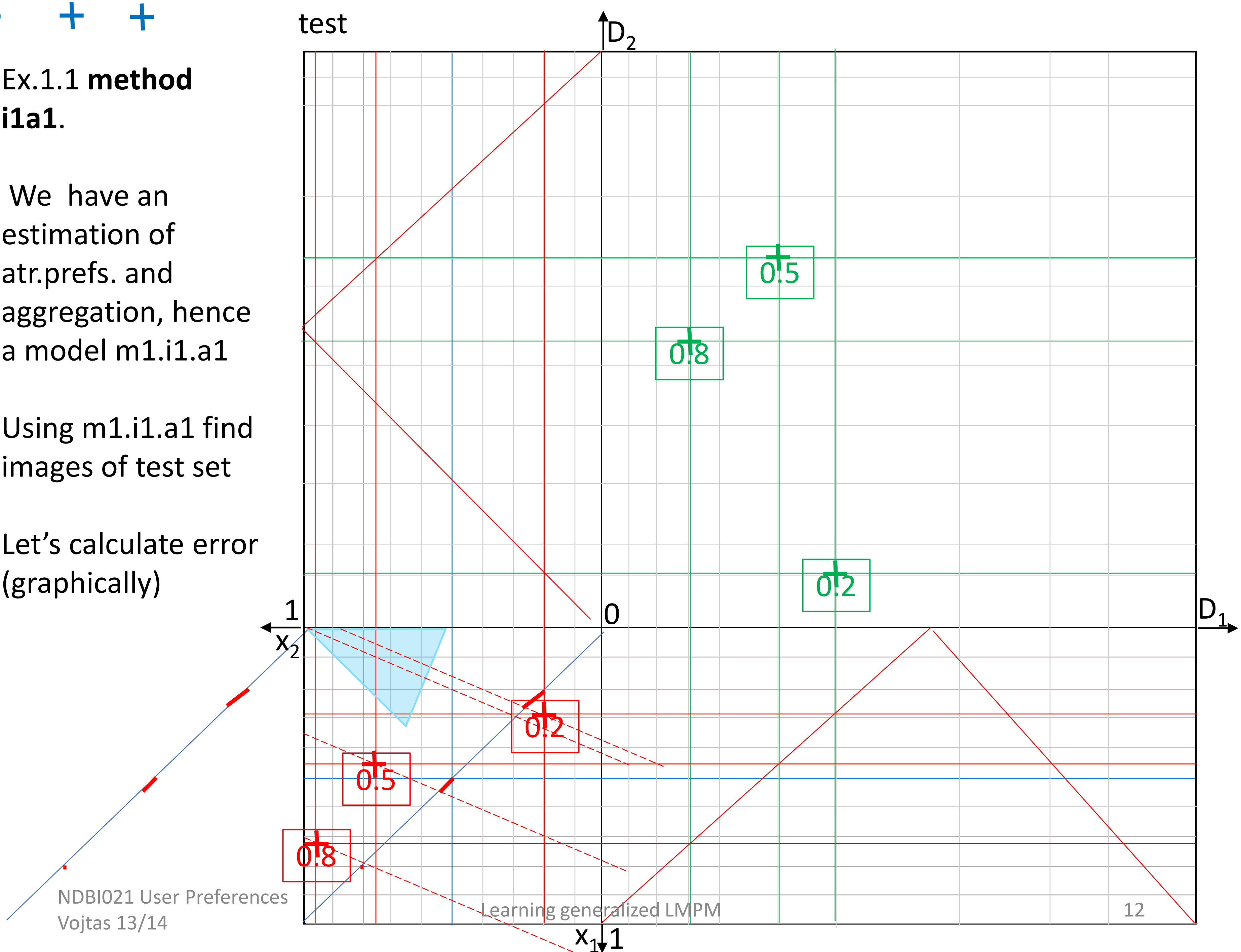
+ + +

Ex.1.1 **method i1a1**.

We have an estimation of atr.prefs. and aggregation, hence a model m1.i1.a1

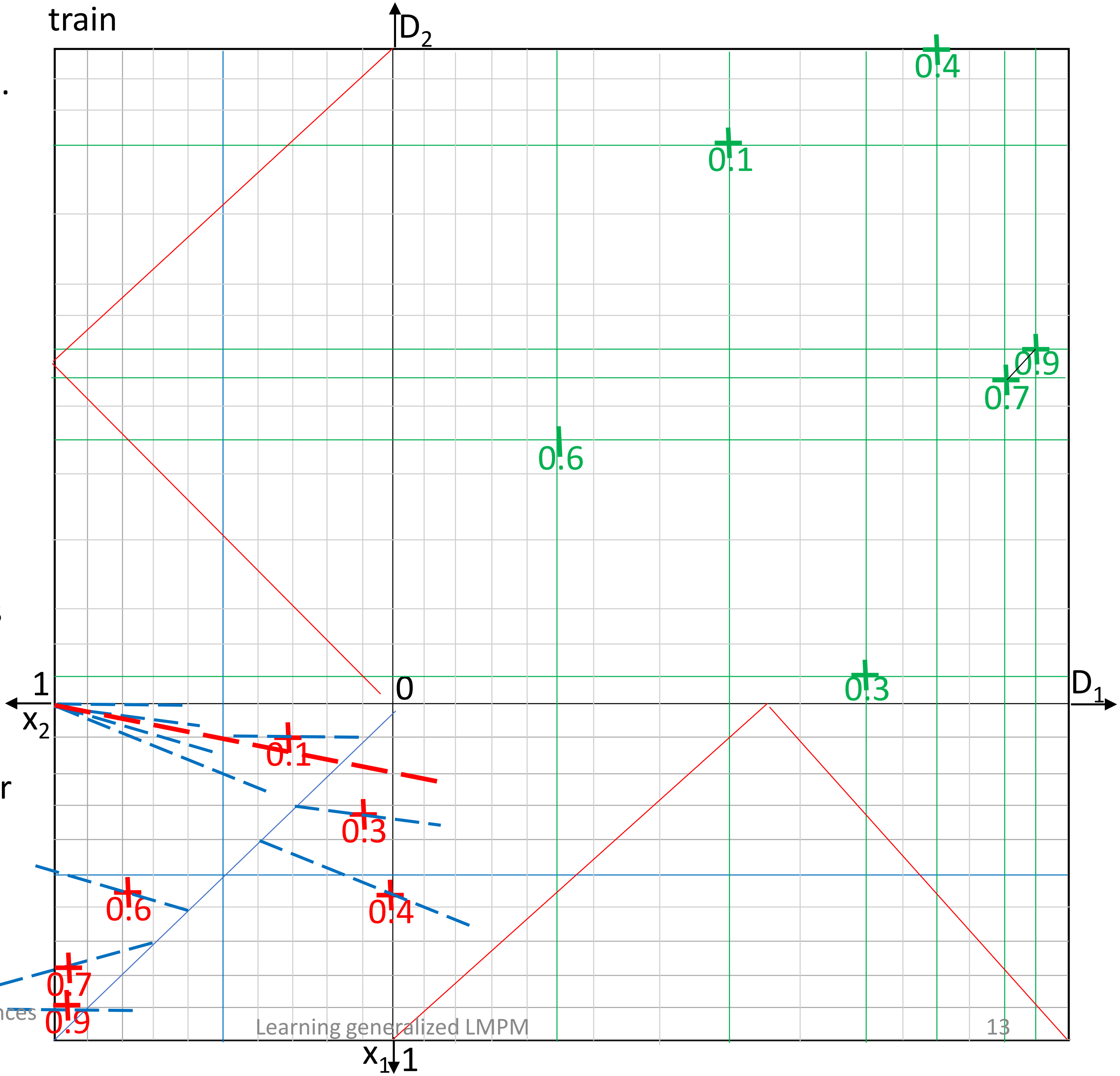
Using m1.i1.a1 find images of test set

Let’s calculate error (graphically)



Ex.1.2 **method i1a2**. Learning gives attr. prefs and training set images in PC by i1. **a2 method:** construct an **angle of connections point with “it’s” value on the diagonal** – and get the angle axis.

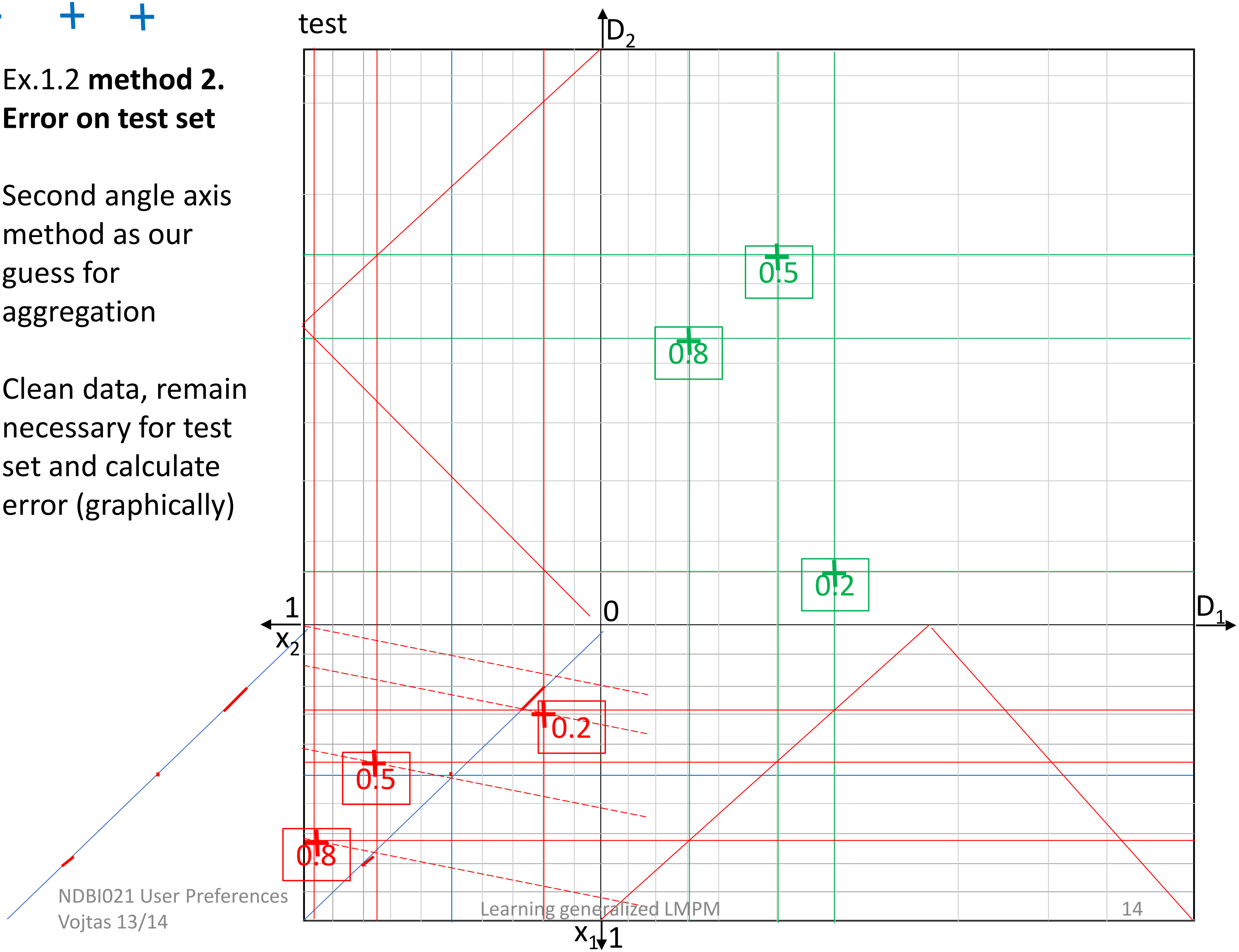
Expected directions NW-SE, violated by 0.7 – confidence 5/6. Lines create an angle (much smaller than previous one), chose again an axis



Ex.1.2 **method 2. Error on test set**

Second angle axis method as our guess for aggregation

Clean data, remain necessary for test set and calculate error (graphically)

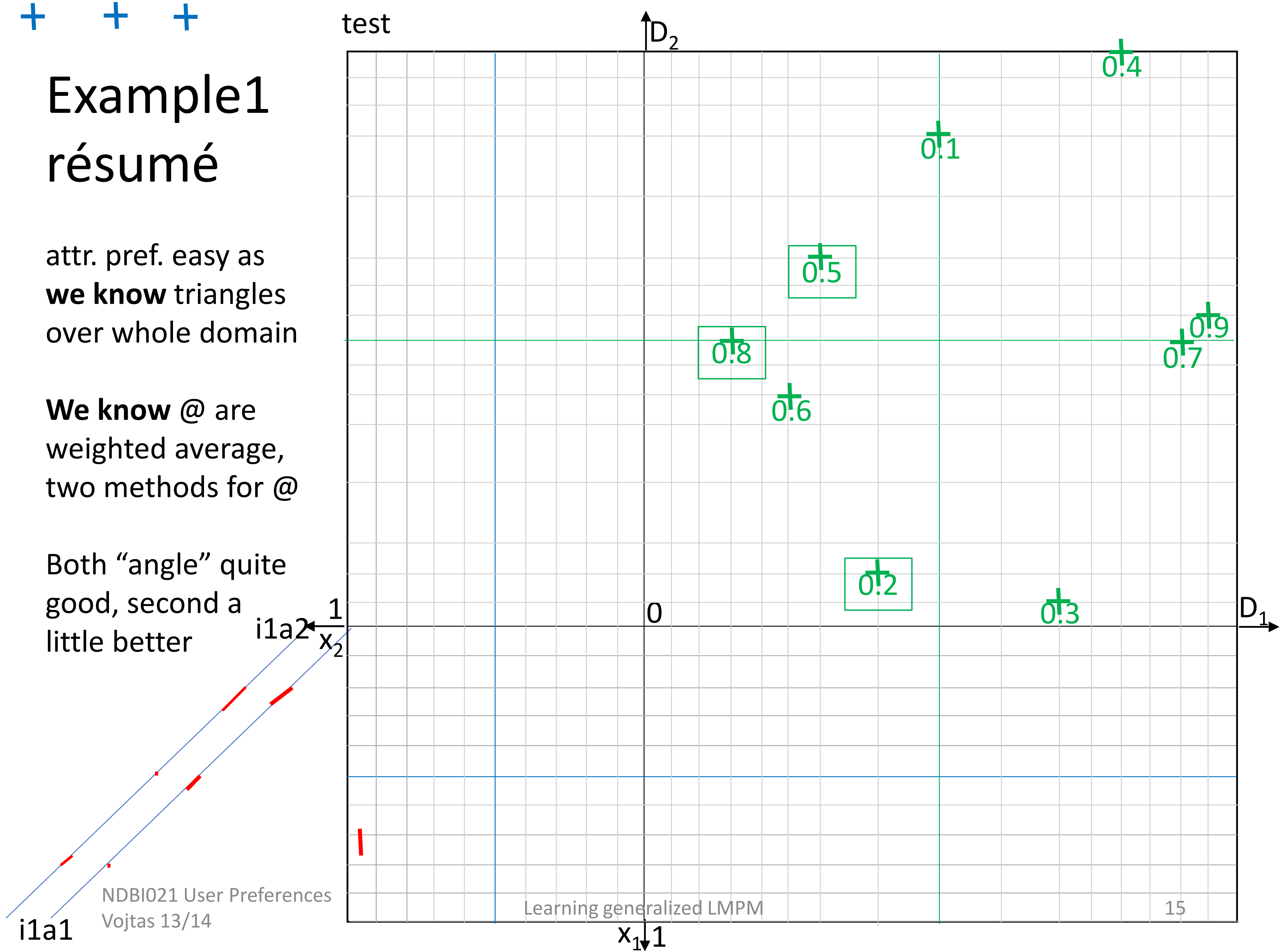


Example1 résumé

attr. pref. easy as **we know** triangles over whole domain

We know @ are weighted average, two methods for @

Both “angle” quite good, second a little better



Experiment 2.

Data generated by product disjunction on respective PC contour lines

We know that attribute preferences are triangular (hill, valley) on whole domain

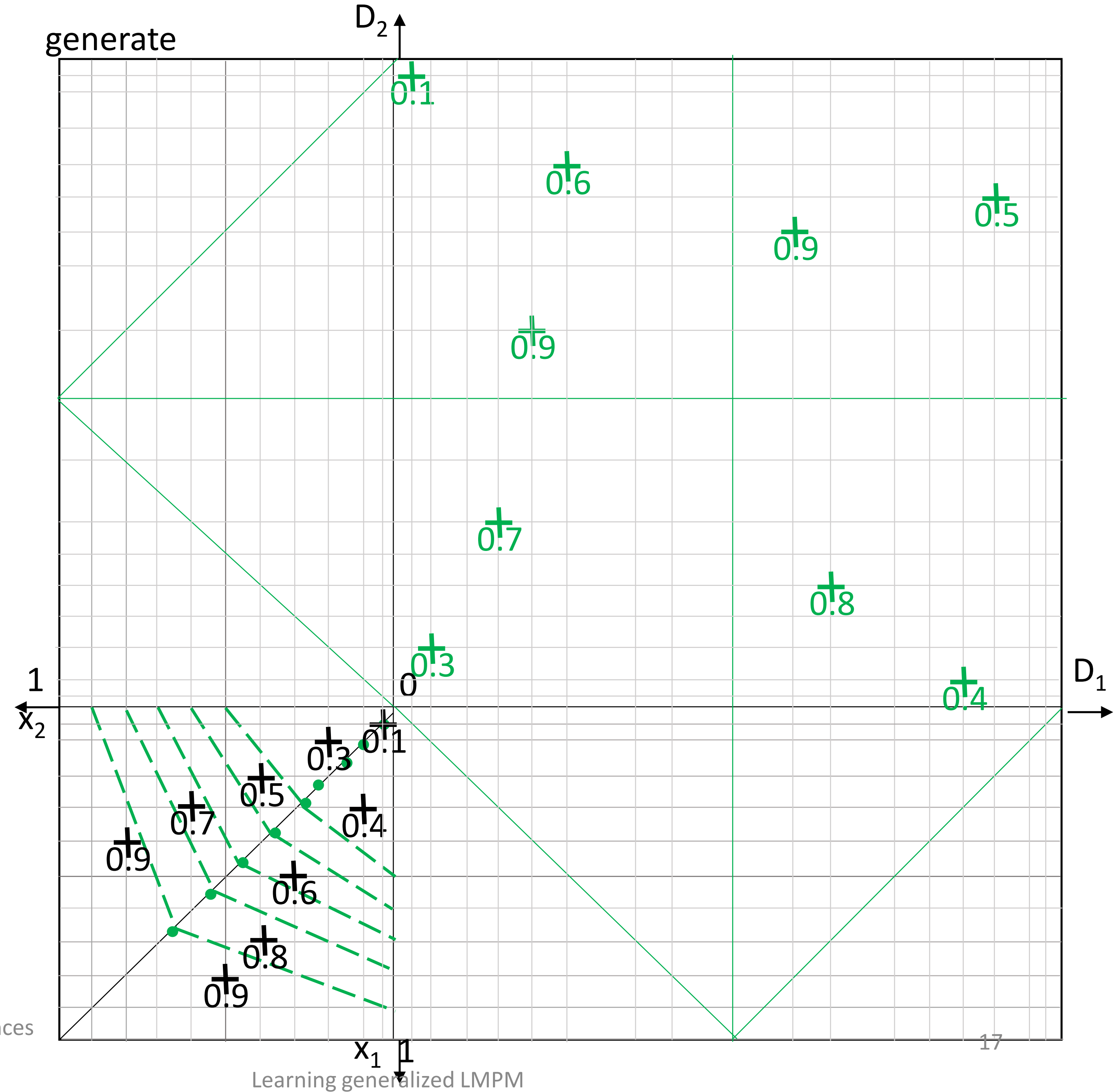
Let's try to learn also weighted average model, as if did not know how data were generated

+ + +

Ex.2 data generated by product logic **disjunction** and attr. Prefs.

This is a base for generating examples

Only for copying

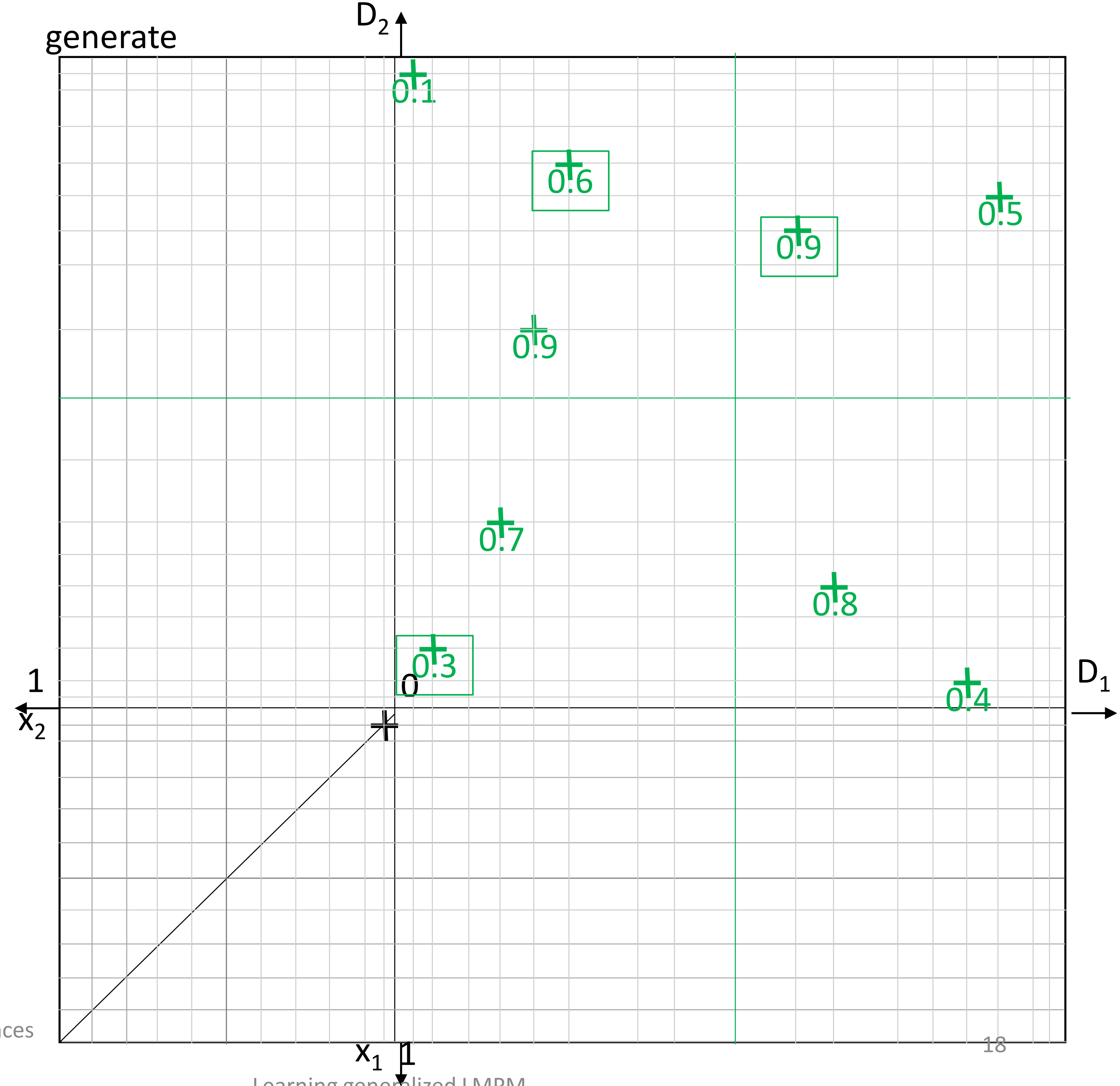


0.1

Ex.2 data generated by product logic disjunction and attr. Prefs.

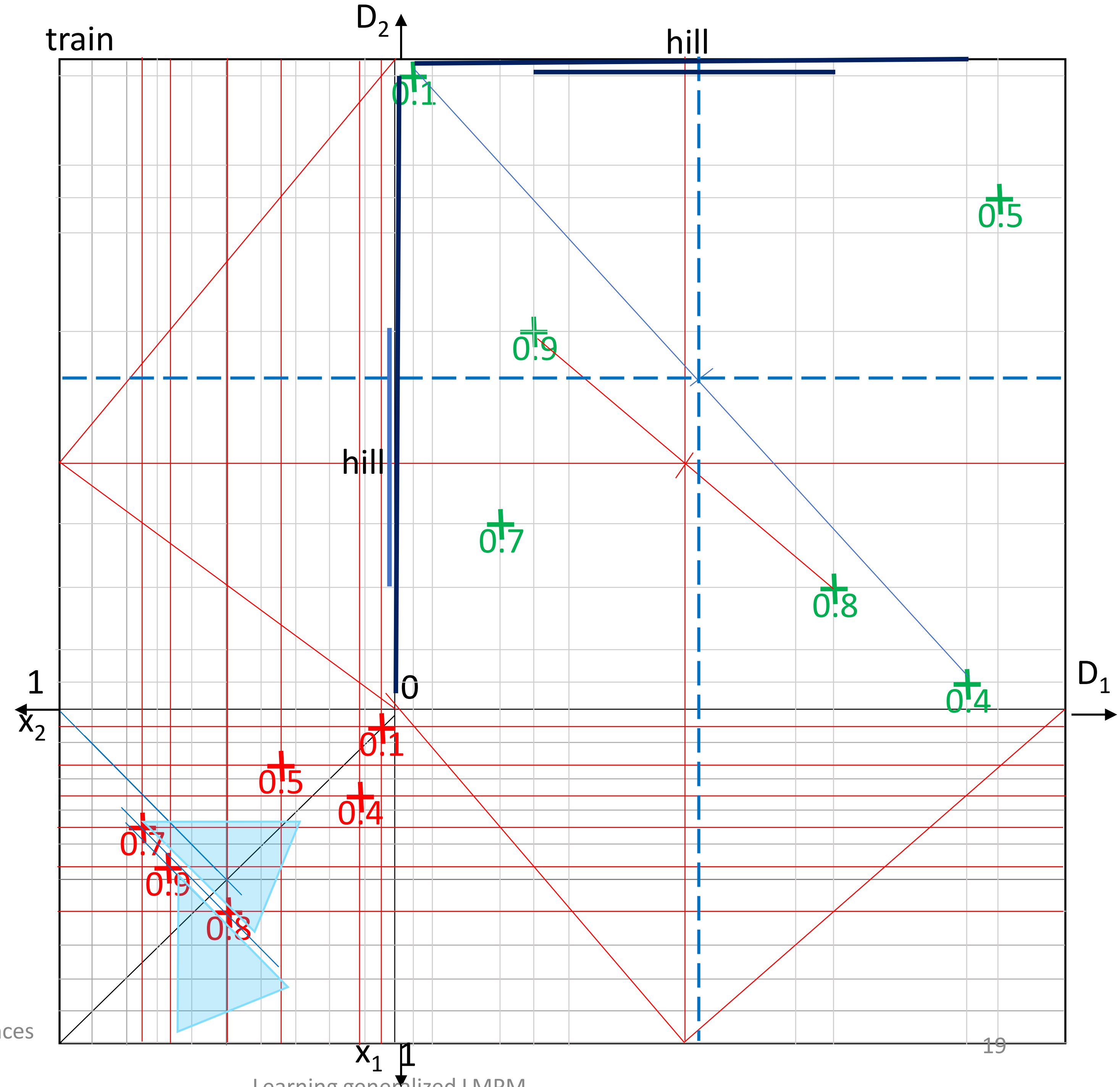
Start to work
Decide which DC points are training and which are testing

Chose train/test

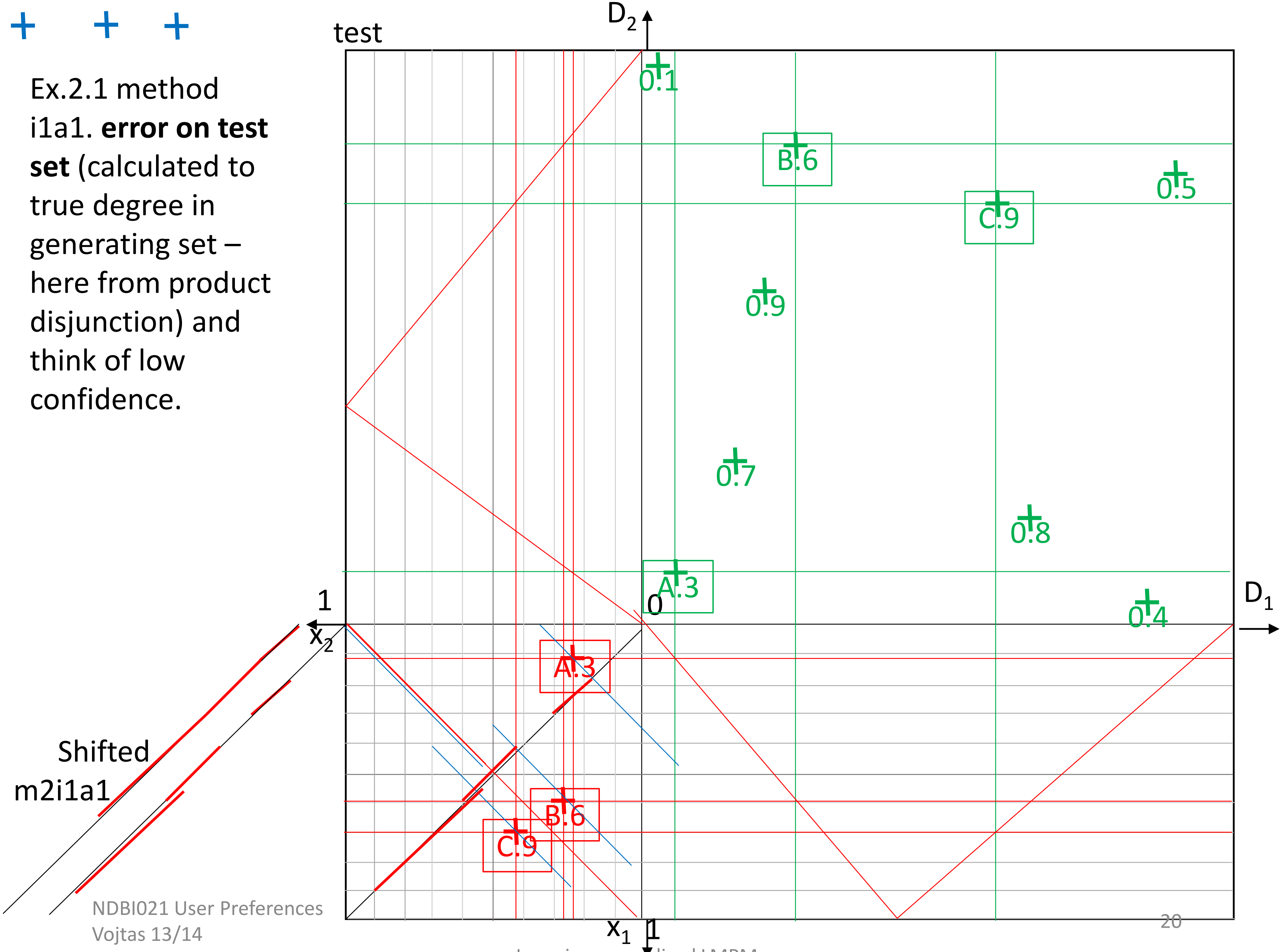


Ex.2.1 **method m2i1a1**. Data generated by product logic disjunction, as in m1i1 chose first red ideal points, blue later.
Attr. prefs and training set images in PC. Again, points Pareto compliant.
m2a1 b**y m1a1 as weighted average (angle almost 0º)**.

angle very small – lowers confidence substantially

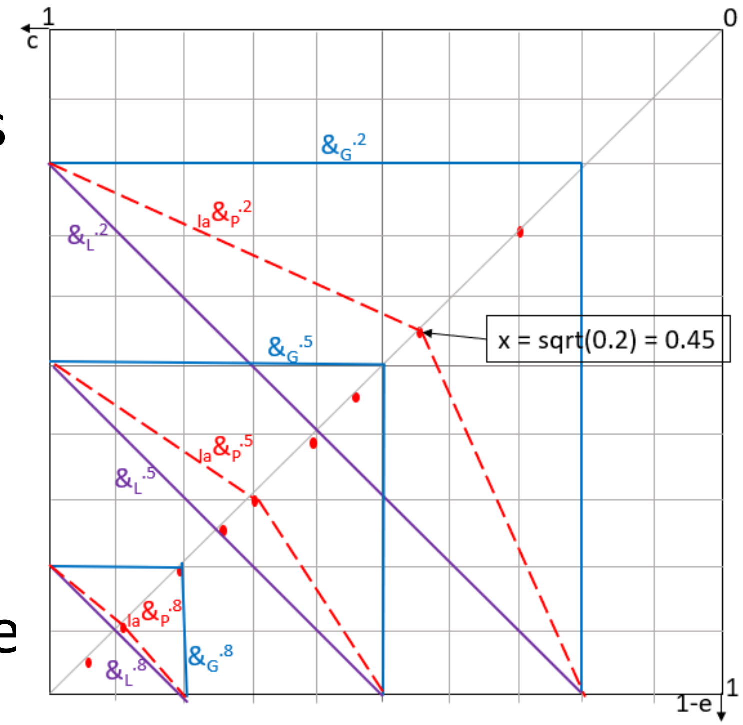


Ex.2.1 method i1a1. **error on test set** (calculated to true degree in generating set – here from product disjunction) and think of low confidence.



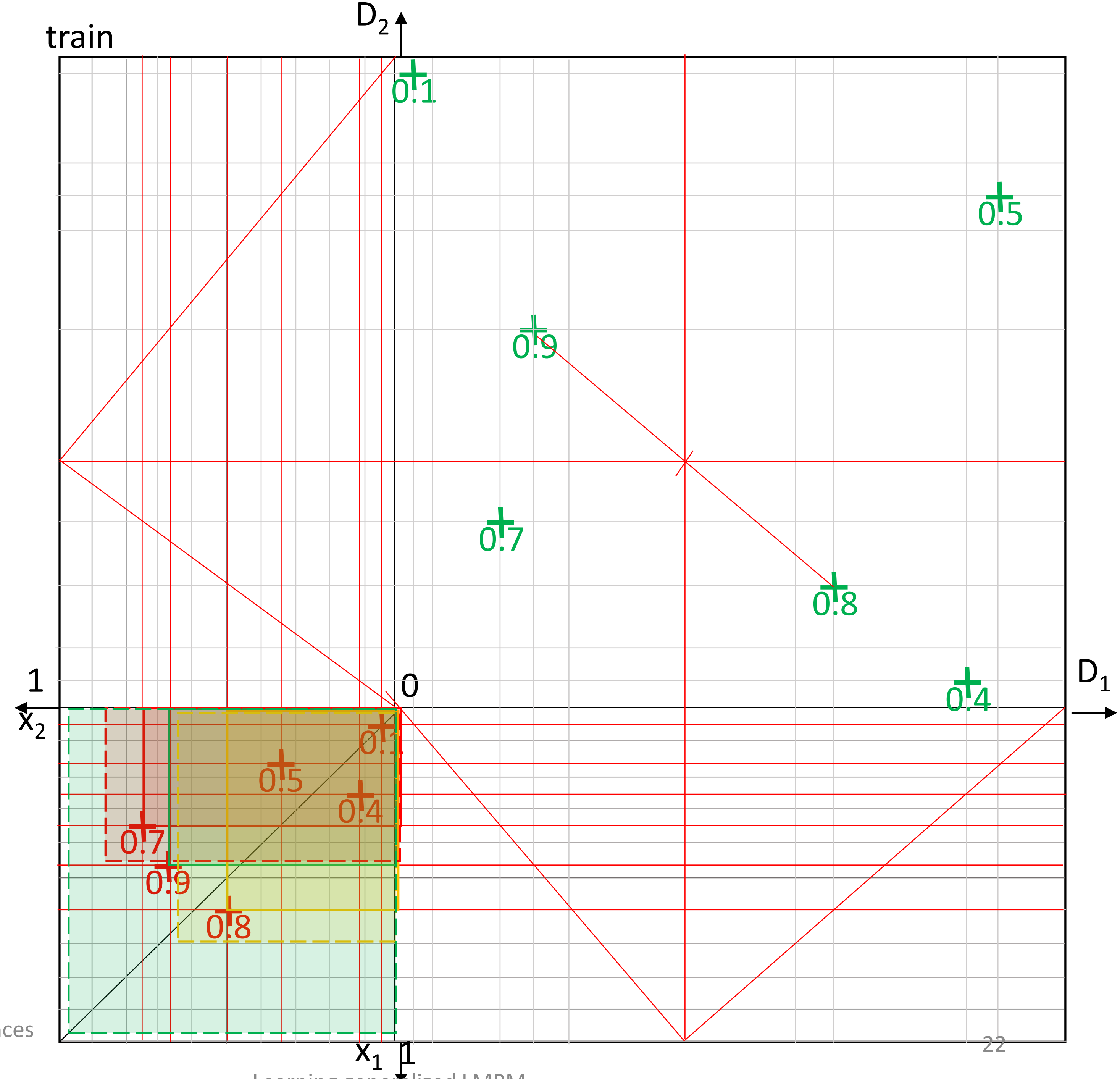
Error versus confidence

- c – confidence – proportion of compliance with model assumptions
- e - error – proportion of maximal error
- Ideal c=1, e=0 (depict c, 1-e)
- How to aggregate c and 1-e to usability of learning results?
- Guess: c-e-aggregation is conjunctive
  - Minimal confidence c=0 and error e=1 make results of learning not usable. So, on 0-axis we have usability 0
    - Are c, 1-e independent? Product logic conjunction?
    - Do c↑1, 1-e↓0 behave opposite, i.e., c↑1, e↑1, rather not Lukasiewicz
    - Dependent? Goedel logic? Probably yes.
    - **How to test these hypotheses?**



+ + +

Ex.2.2 **method 2a and 2b** Data, attr. Prefs, training set in PC as before. Let’s try to find agg. In form of Eckhardt geometry / Pareto min (solid outline) /max (outline dashed) method. There are no violations of Pareto order, so confidence is high.



0.9

Ex.2.2 method 2a – Pareto **minimal** region.

error on test set

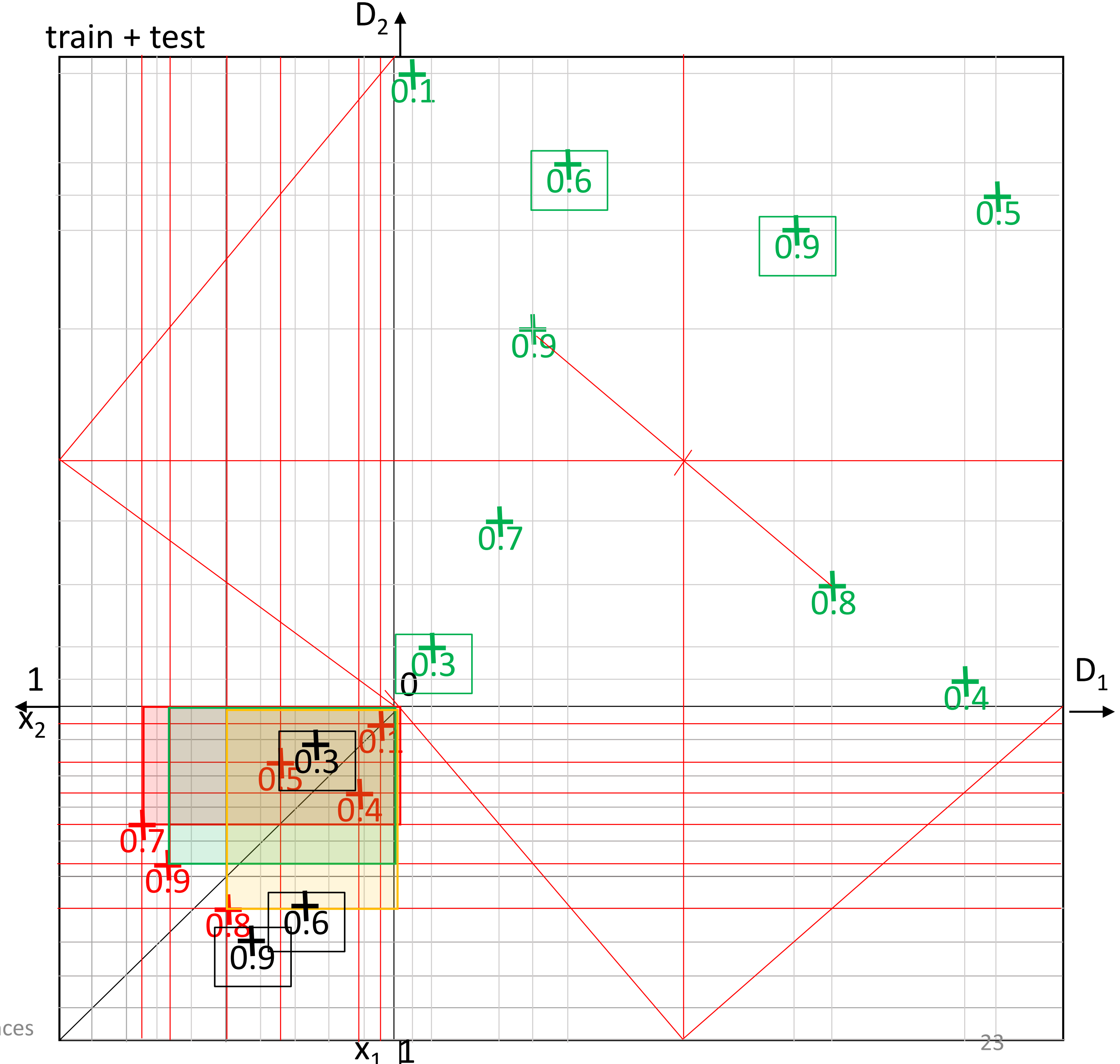
Image of test set in PC is always same (point identifiers are pref.score)

0.3 lies in Eckhardt geometry minimal region of 0.5 (learned from training example) – gives to error 0.2

0.6 lies in 0.8 e=0.2

0.9 lies in 1, e=0.1

In **total error = 0.5**



0.9

Ex.2.2 method 2b – Pareto **maximal** region.

error on test set

Image of test set in PC is always same

Image of test set in PC is always same

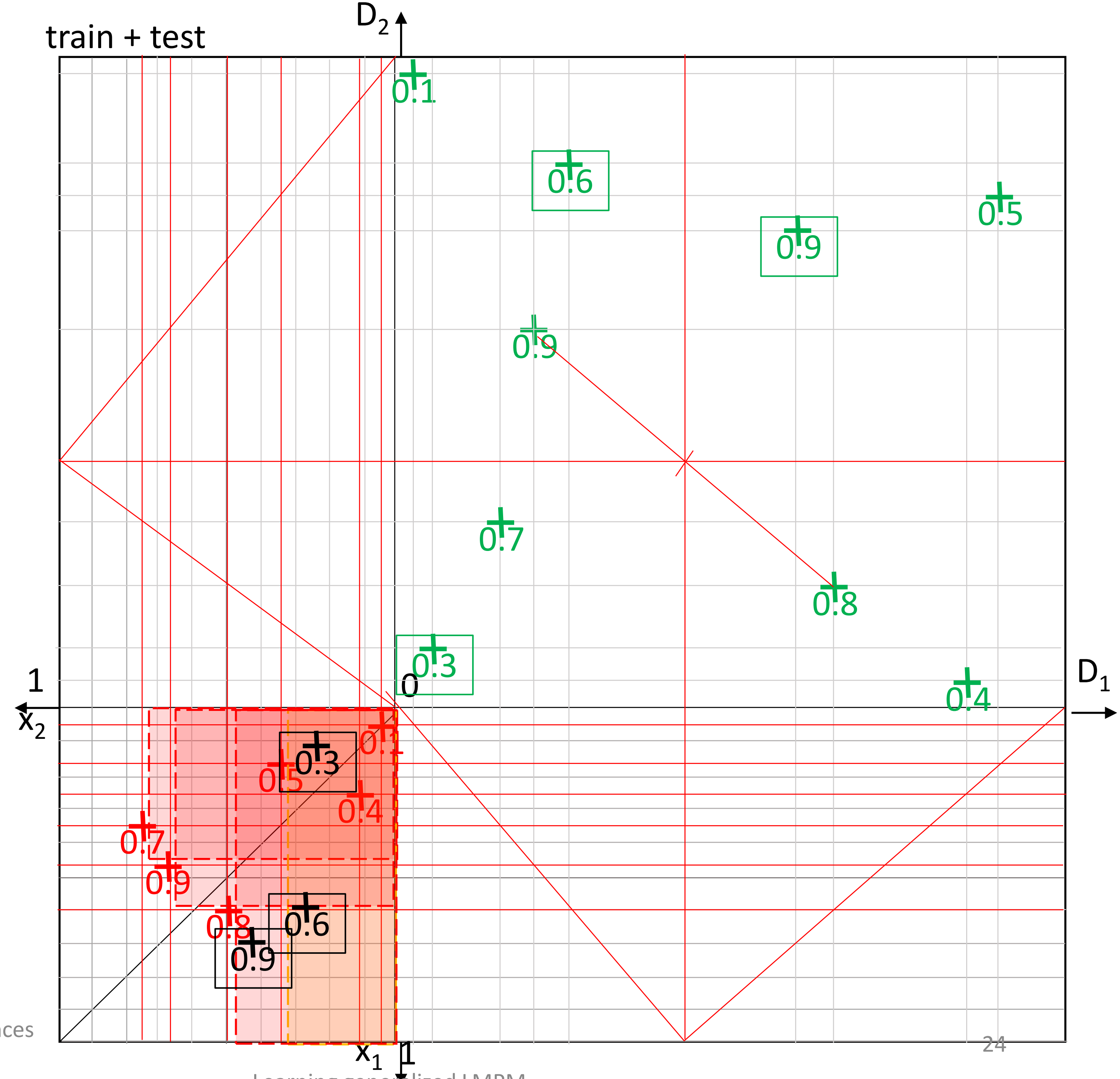
(point identifiers are pref.score)

0.3 lies in Eckhardt geometry maximal region of 0.4 – gives to error 0.1

0.6 lies in 0.4, e+0.1

0.9 lies in 0.5, e+0.4

Total **error = 0.6**



Ex.2.2 comparing methods 1, 2a, 2b

error on test set

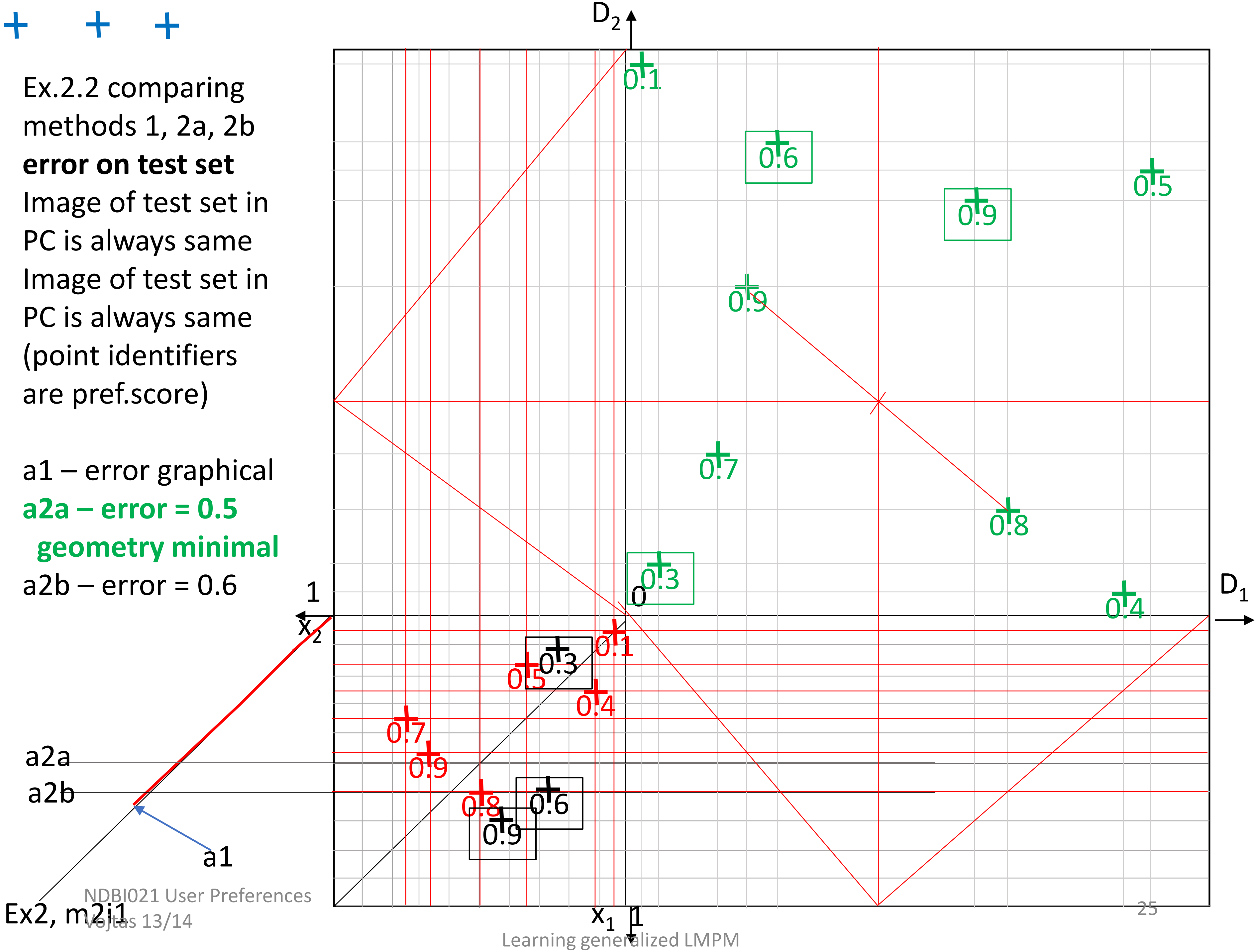
Image of test set in PC is always same

Image of test set in PC is always same (point identifiers are pref.score)

a1 – error graphical

a2a – error = 0.5
geometry minimal

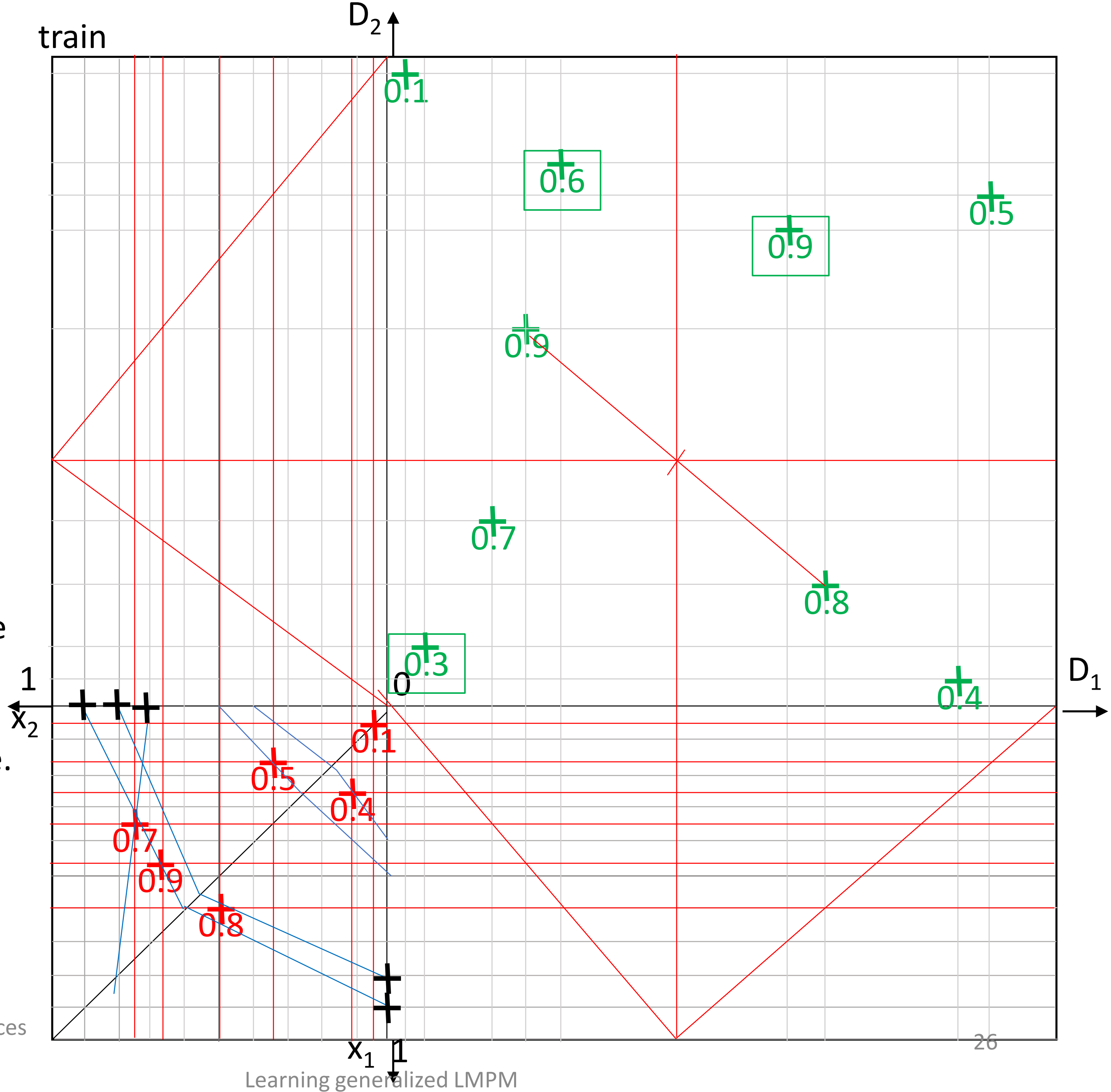
a2b – error = 0.6



Ex.2.3 **method 3.** Data, attr. Prefs, training set in PC as before.

Let's try to **find agg. in form of many valued disjunction** – connecting points to known behavior on axis and corresponding intersection with diagonal. Confidence is 5/6 high – there is only one violation with 0.7 contour line. For test set we must approximate corresponding contour lines

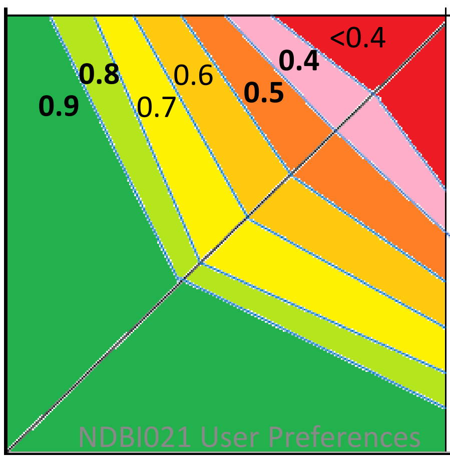
0.9

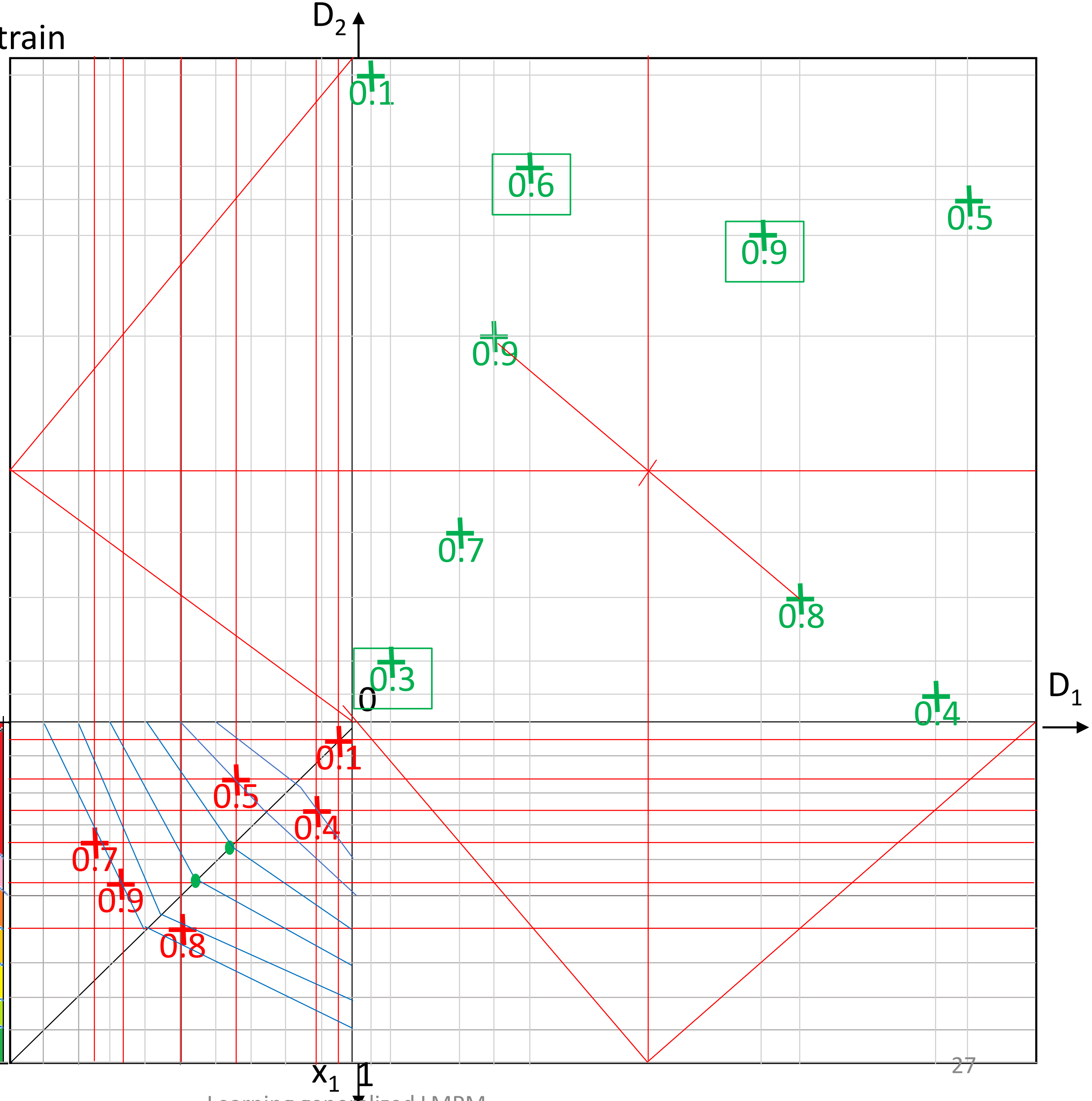


+ + +

Ex.2.3 **method 3**. Approximating missing contour lines

Data, attr. Prefs, training set in PC as before. We found agg. in form of many valued disjunction. Missing 0.6 and 0.7 are heuristically constructed by equidistant points on diagonal.





Ex.2.3 method 3.

Error on test set

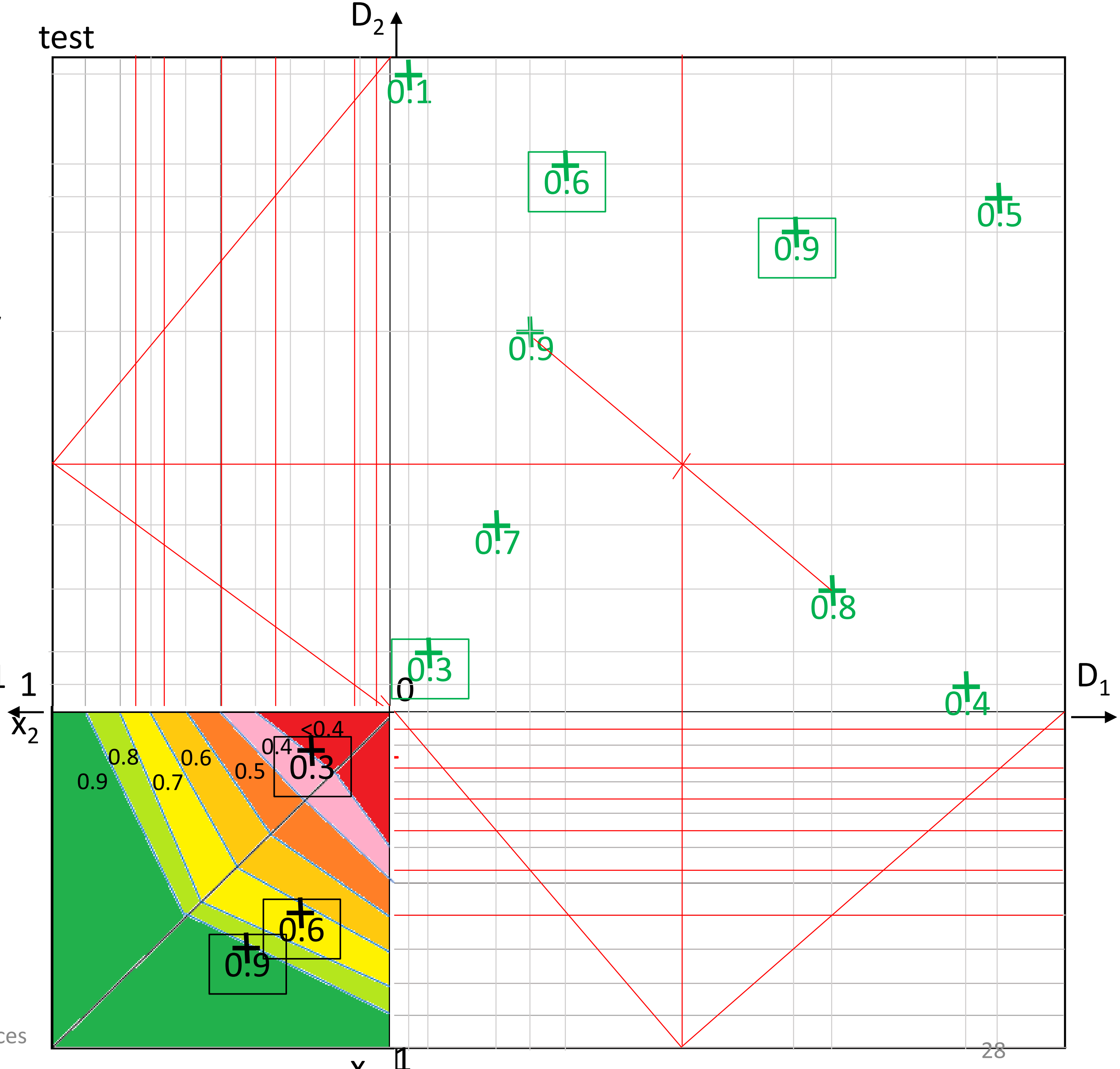
Data, attr. Prefs, training set in PC as before. We found agg. in form of many valued disjunction. Missing 0.6 and 0.7 are added. As f_1, f_2 did not change, images in PC are same.

0.3 lies in region of 0.4 – gives error +0.1

0.6 gives to error +.1

0.9 lies on border of 0.9, so error is 0

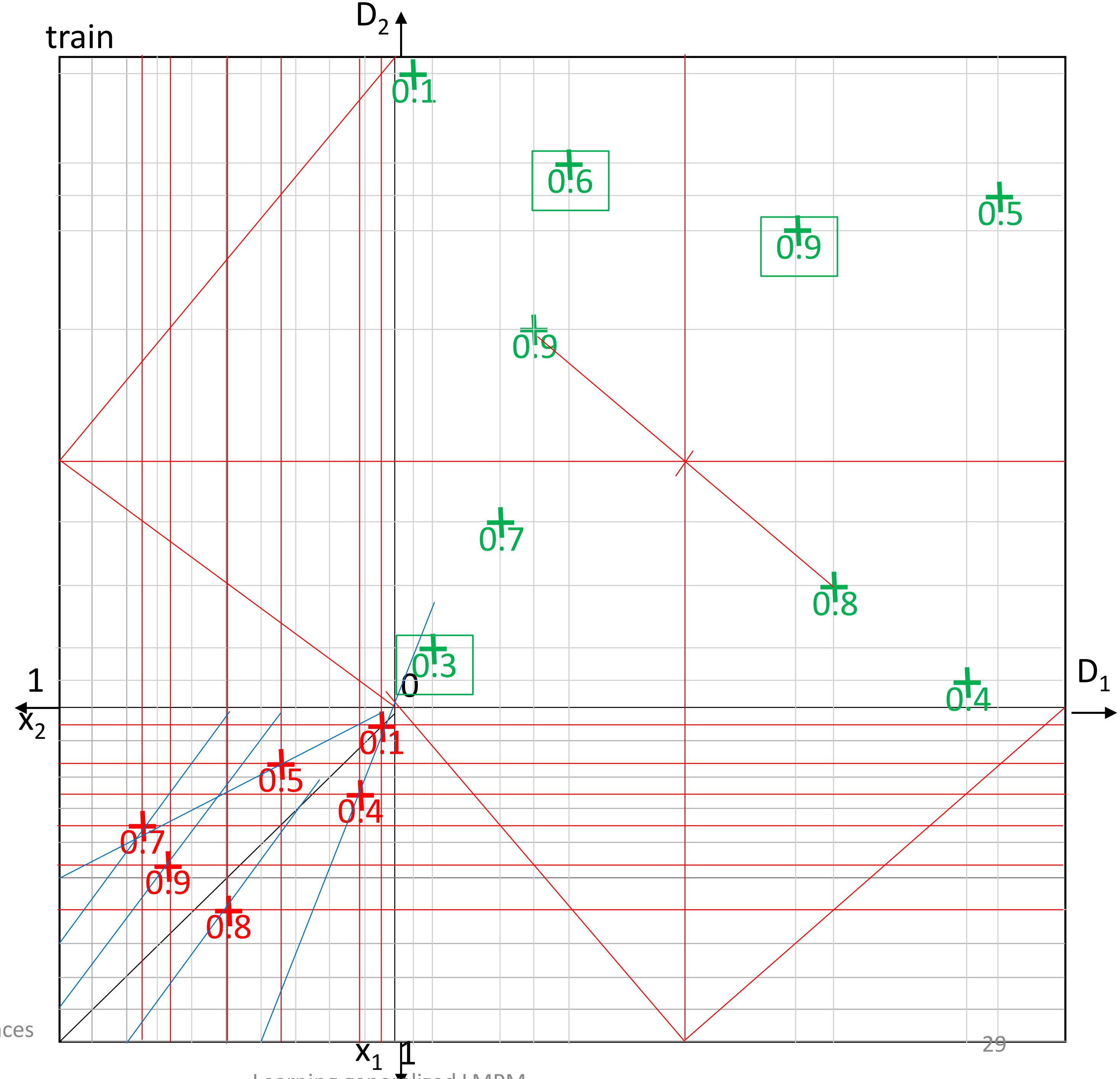
Overall error is 0.2



+ + +

Ex.2.4 **method 4.** Data, attr. Prefs, training set in PC as before.
Let's try to **find agg. in form of many valued conjunction** – connecting points to known behavior on axis and corresponding intersection with diagonal.
Confidence is very low, zero – there is no intersection of estimated contour lines with diagonal – **learning failed**

0.9



Example2 résumé

attr. pref. easy as
we know triangles over whole domain

We do not know @ models and have to try many

Methods

a1 – e>0.6

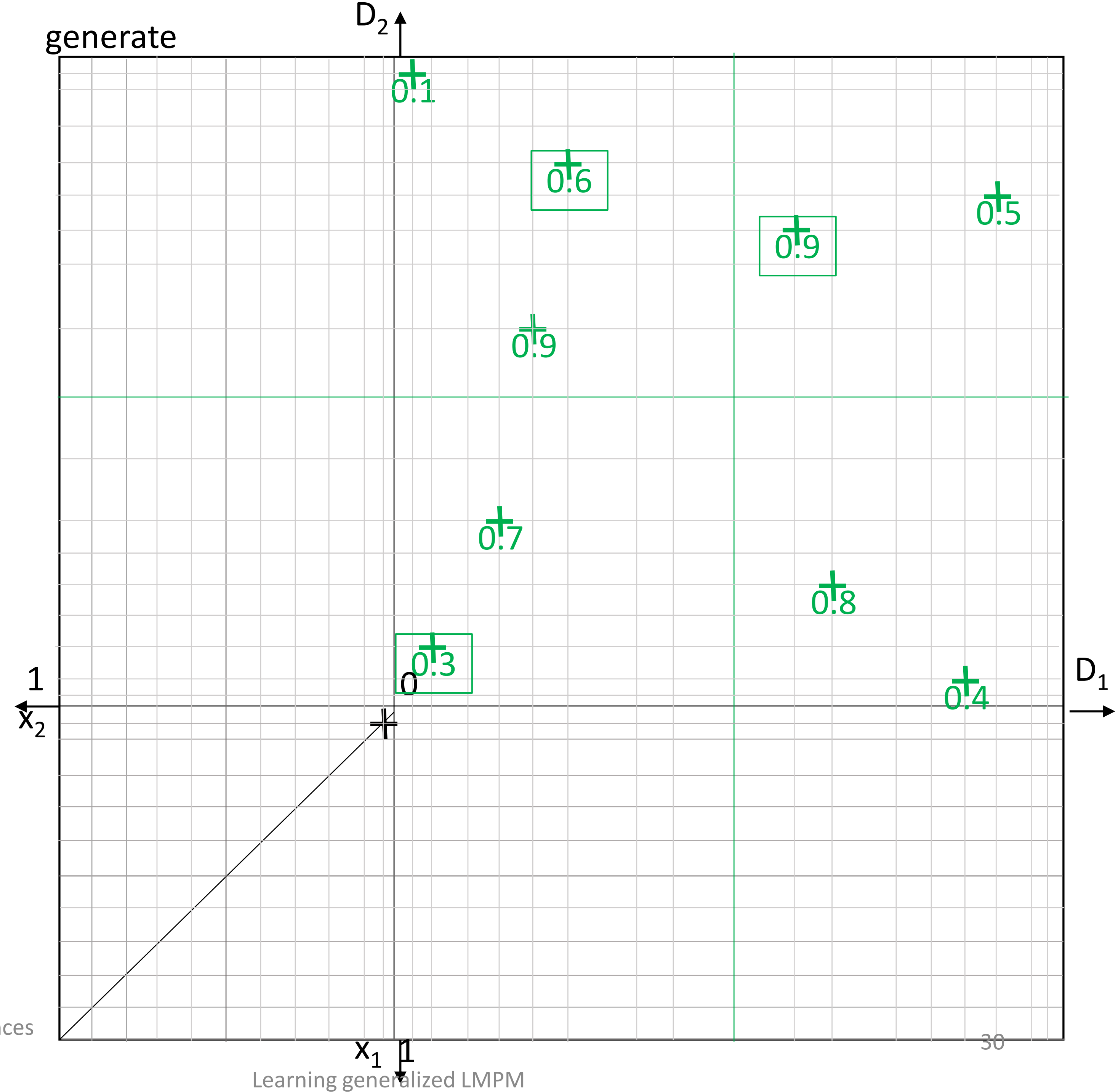
a2a – e=0.5

a2b – e=0.6

a3 – e=0.2

a4 – failed c=0

generate

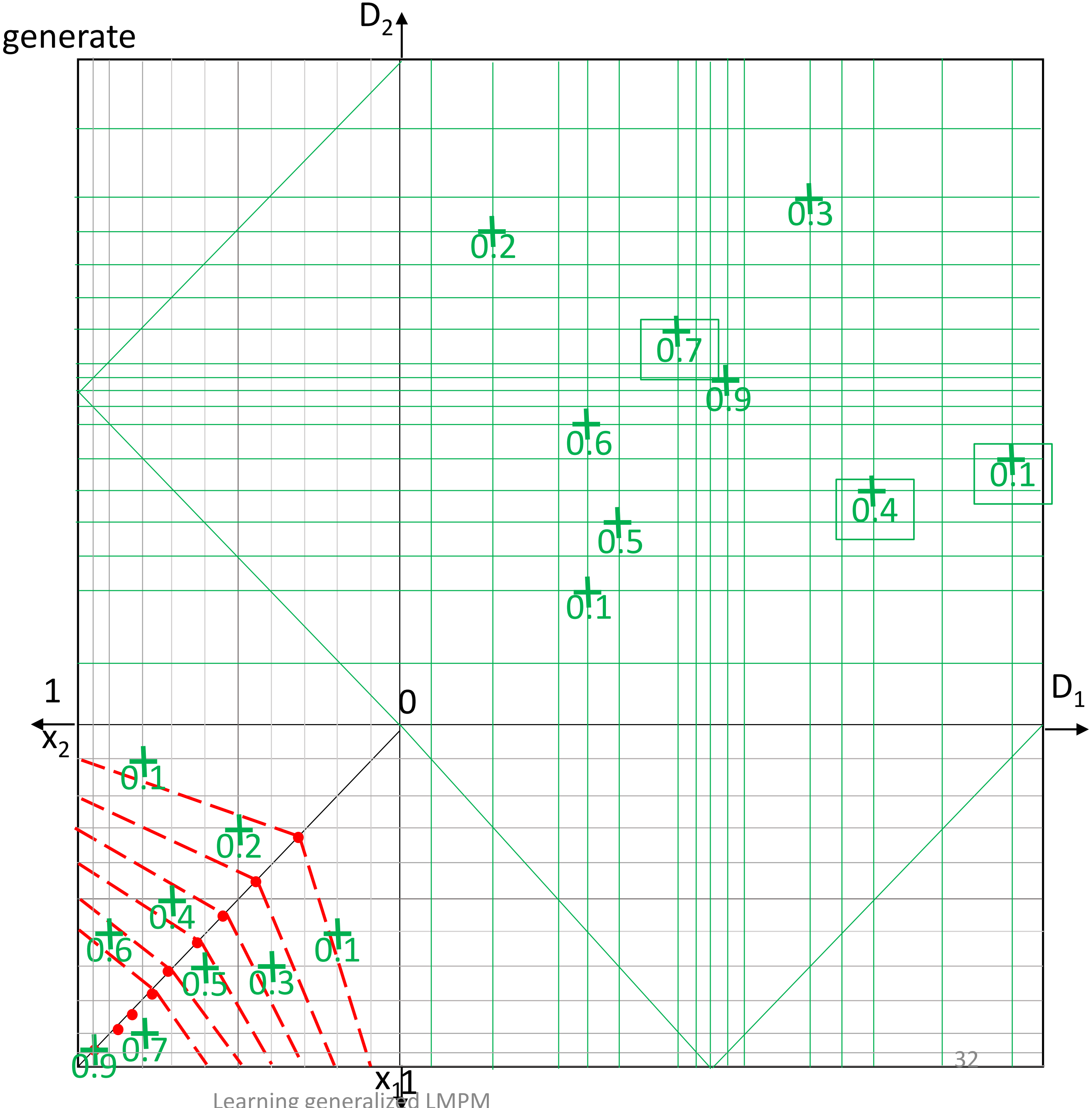


Experiment 3.

Data generated by product **conjunction** on respective PC contour lines

Ex.3generated by product logic **Conjunction**, again Nine different preference degrees.

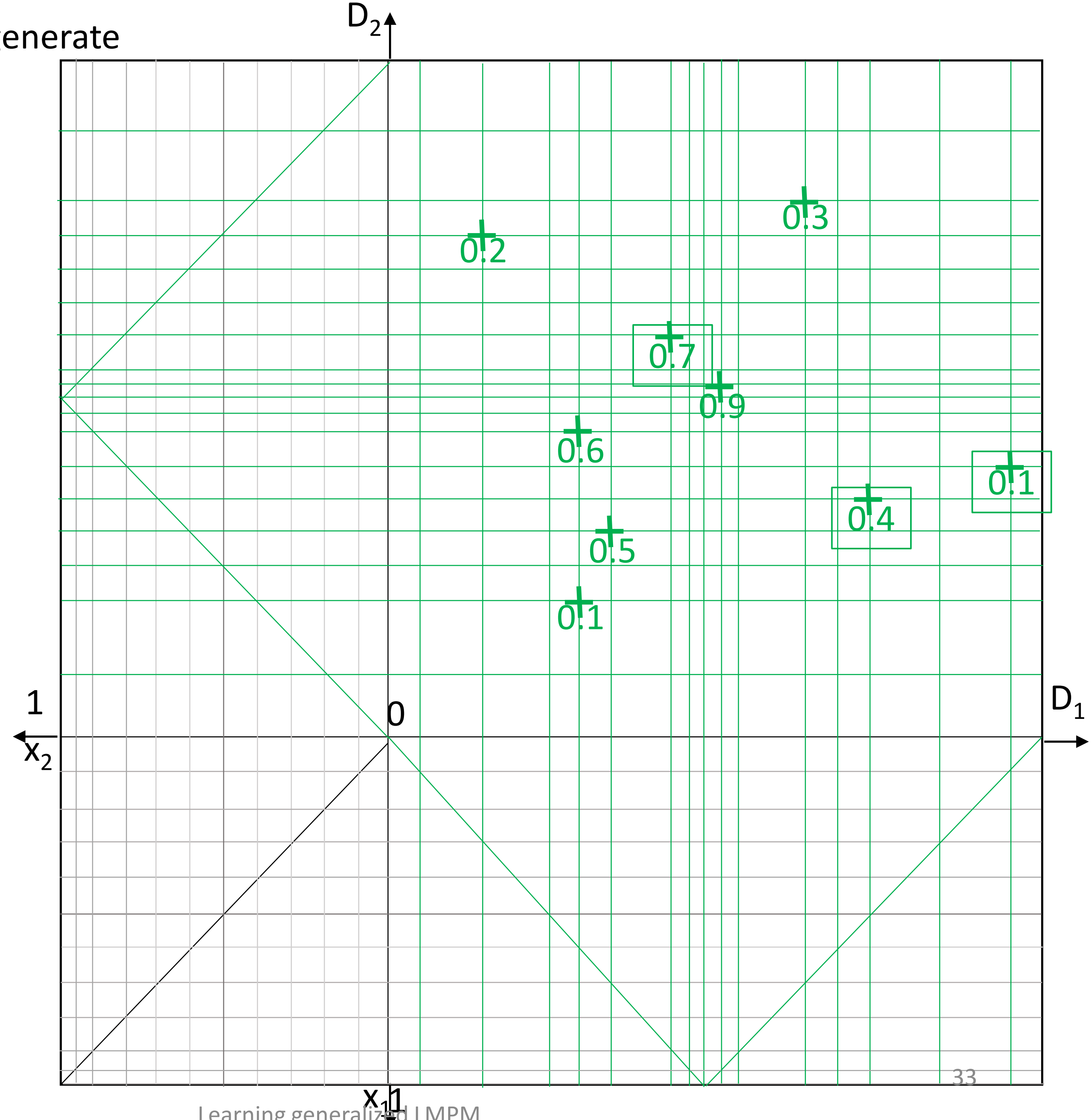
Basis for generating data points
Only for copying



Ex.3generated by product logic Conjunction, again Nine different preference degrees.

Basis for generating data points

Only for copying

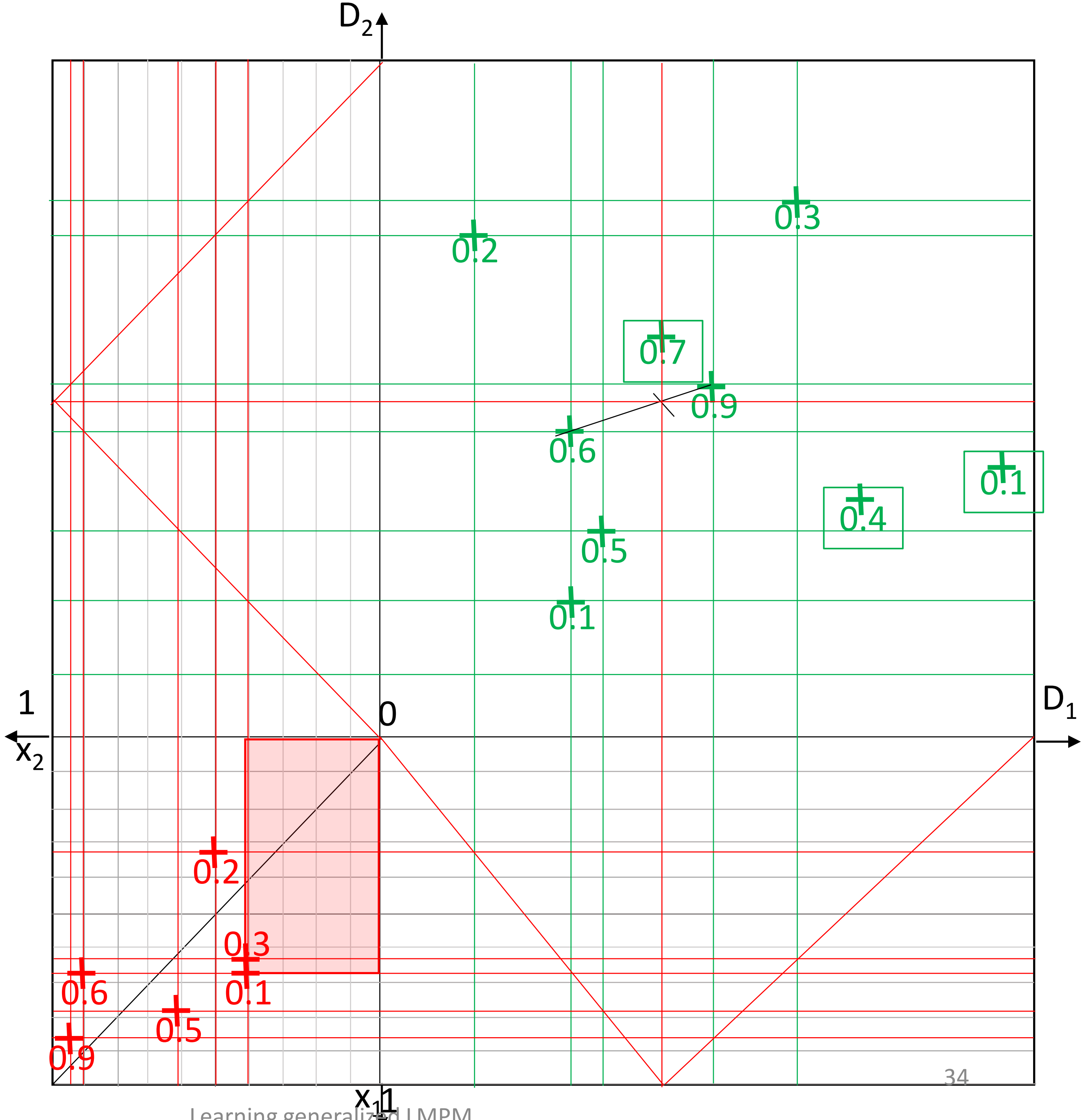


Ex.3data generated by product logic conjunction

i1 as before

One Pareto violation 0.3 is dominated by 0.1

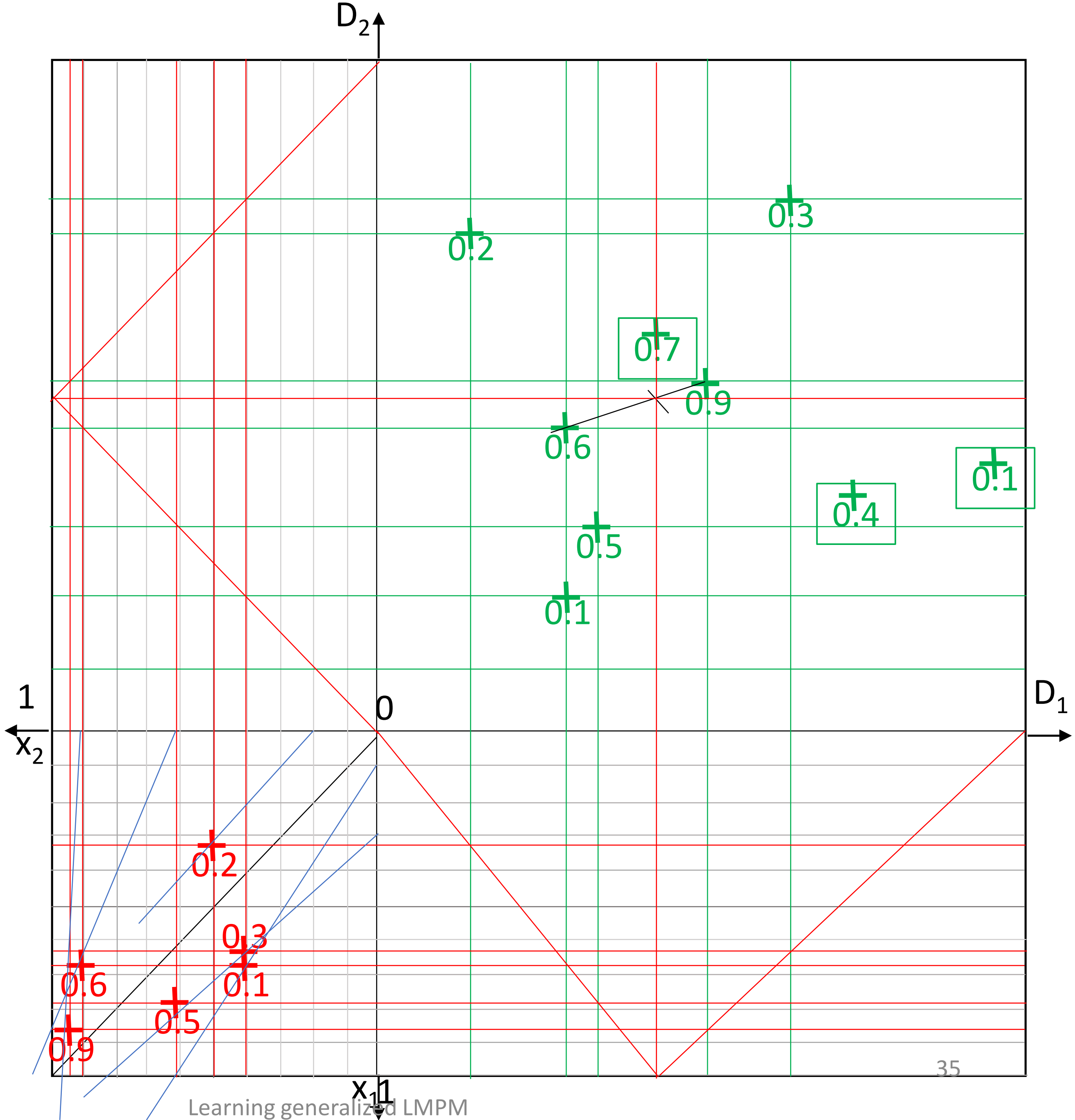
Does it influence confidence?



Ex.3data generated by product logic conjunction

a1 - **Or-like aprox.** never intersects diagonal

m3i1a1 Confidence =0



0.1

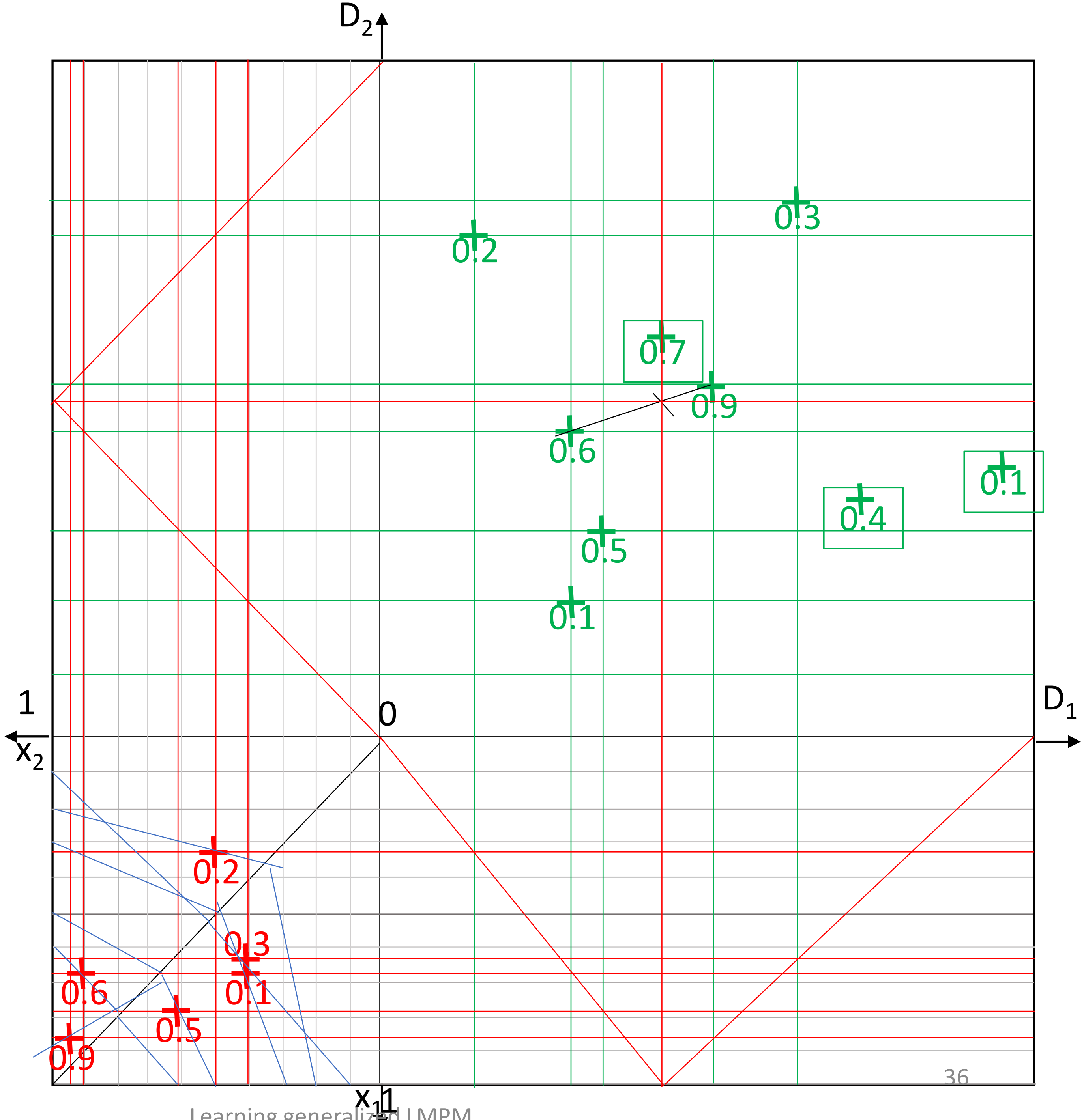
+ + +

Ex.3data generated by product logic conjunction

a2-And-like aprox. intersects diagonal

0.1 causes problem, decreases confidence
Also 0.9 violates contour lines 0.6 and 0.5

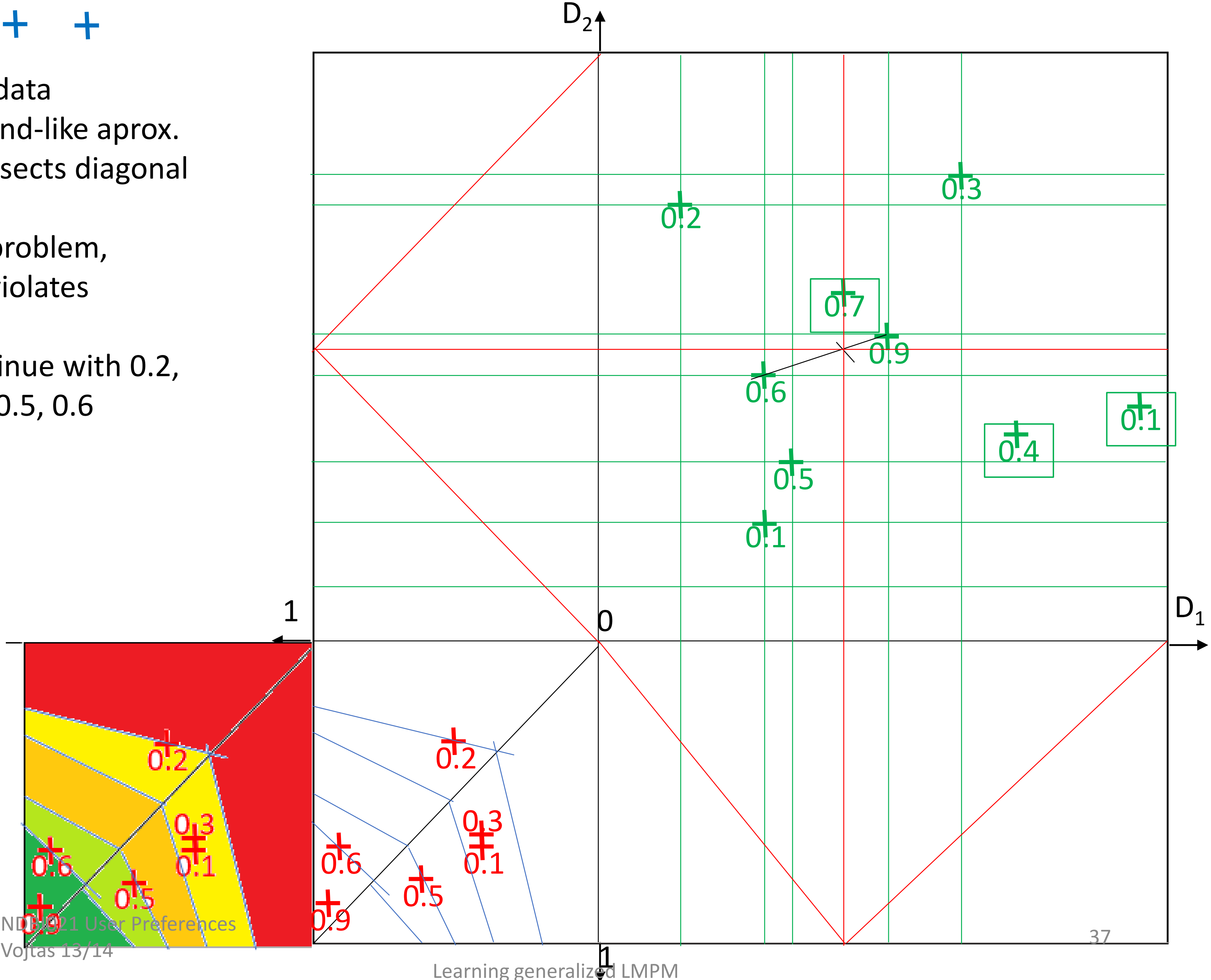
Let us continue with 0.2, 0.3, 0.5, 0.6 which are order compliant and do not intersect smaller / larger lines



Ex.3data
a2-And-like aprox.
intersects diagonal

0.1 problem,
0.9 violates

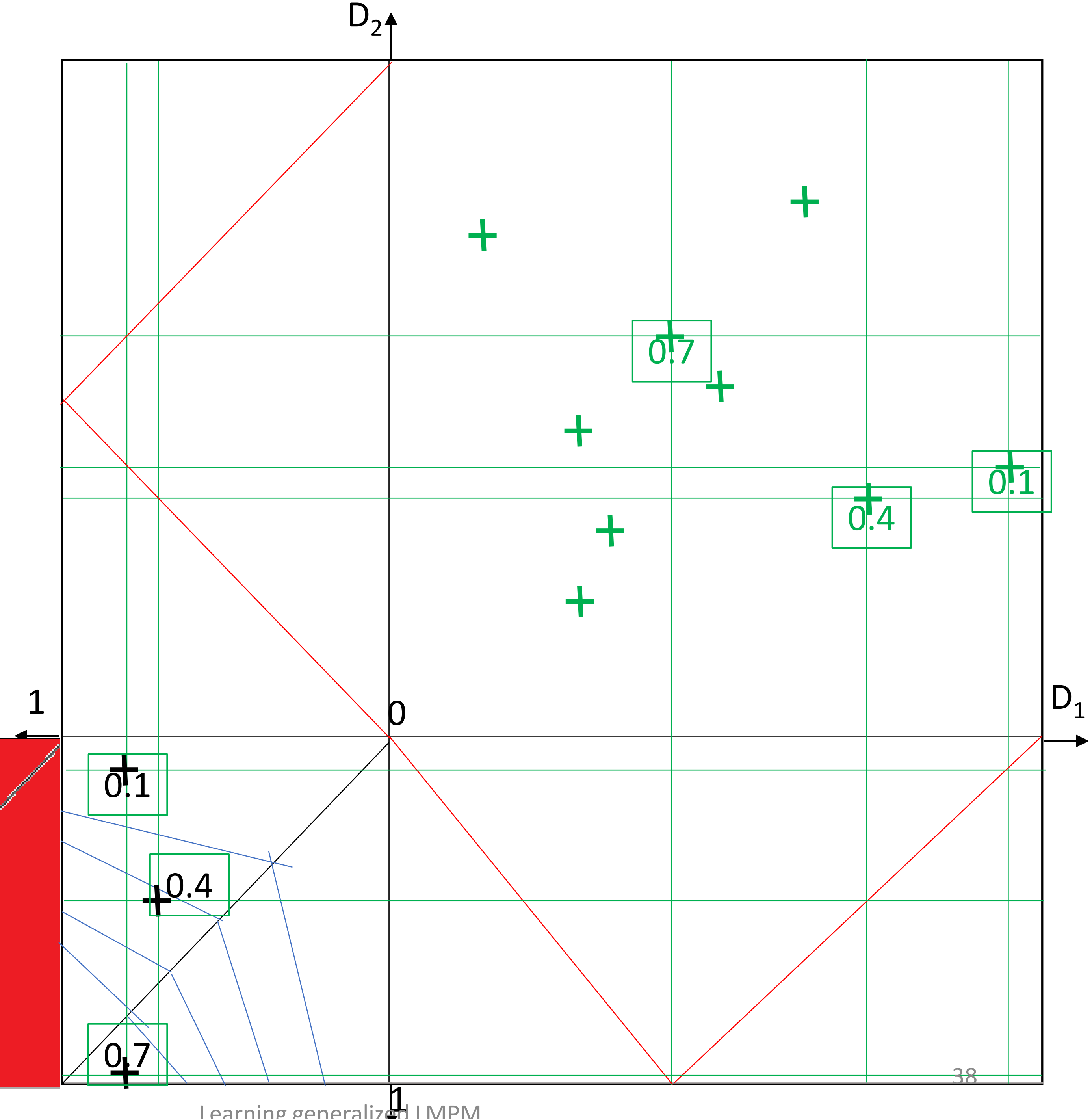
continue with 0.2,
0.3, 0.5, 0.6

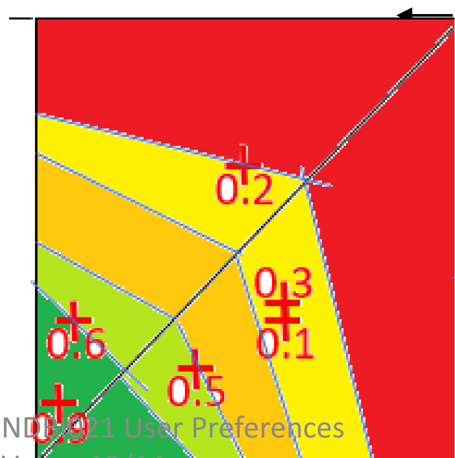


Ex.3data
a2-And-like aprox.
intersects diagonal

0.1 problem,
0.9 violates

continue with 0.2,
0.3, 0.5, 0.6





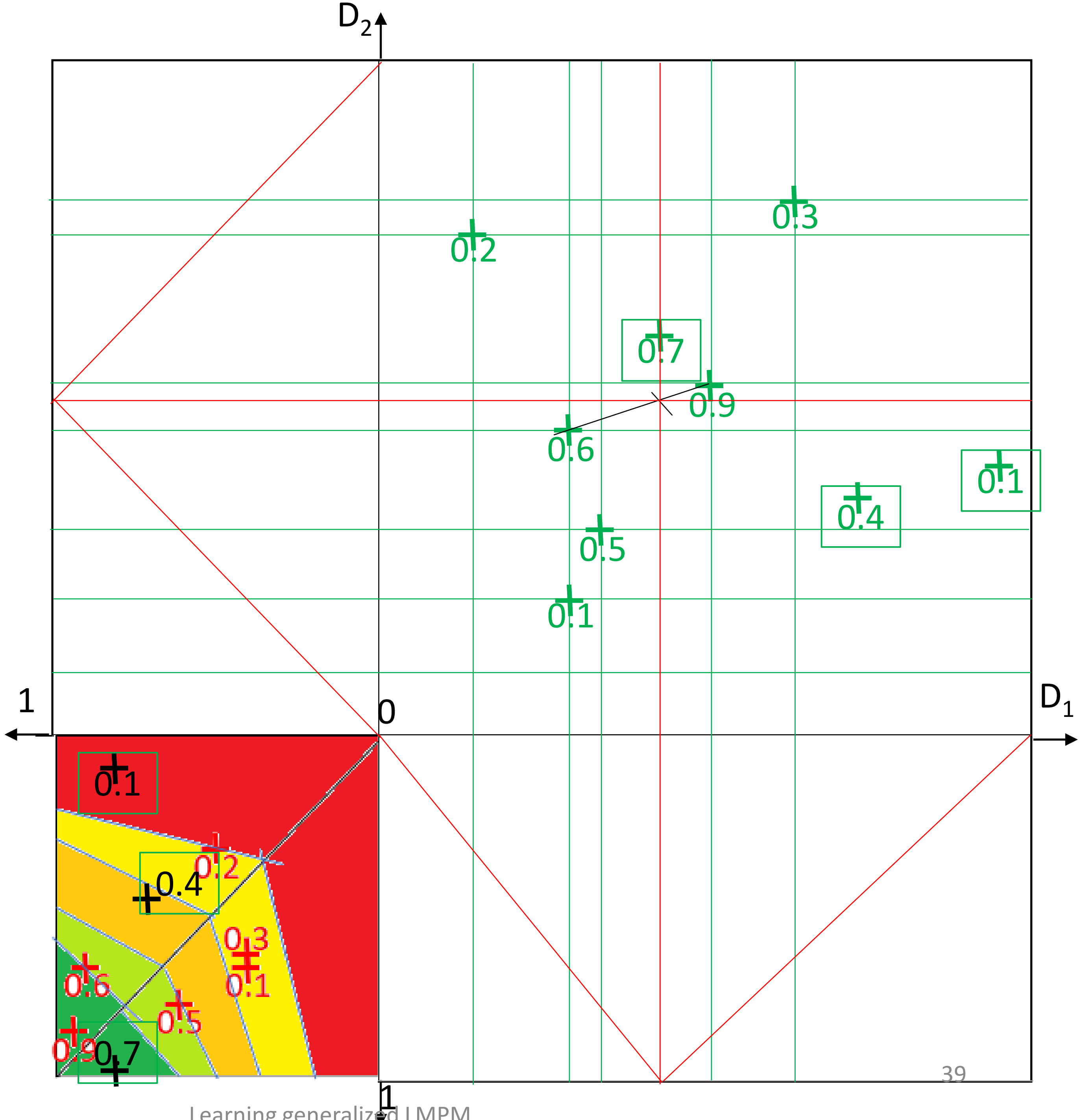
+ + +

Ex.3data generated by product logic conjunction

a2-And-like aprox. intersects diagonal

0.1 causes problem, decreases confidence
Also 0.9 violates contour lines 0.6 and 0.5

Let us continue with 0.2, 0.3, 0.5, 0.6 which are order compliant and do not intersect smaller / larger lines



Example3 résumé

3data generated by product logic conjunction

i1 makes candidate data in PC with one violation of Pareto order

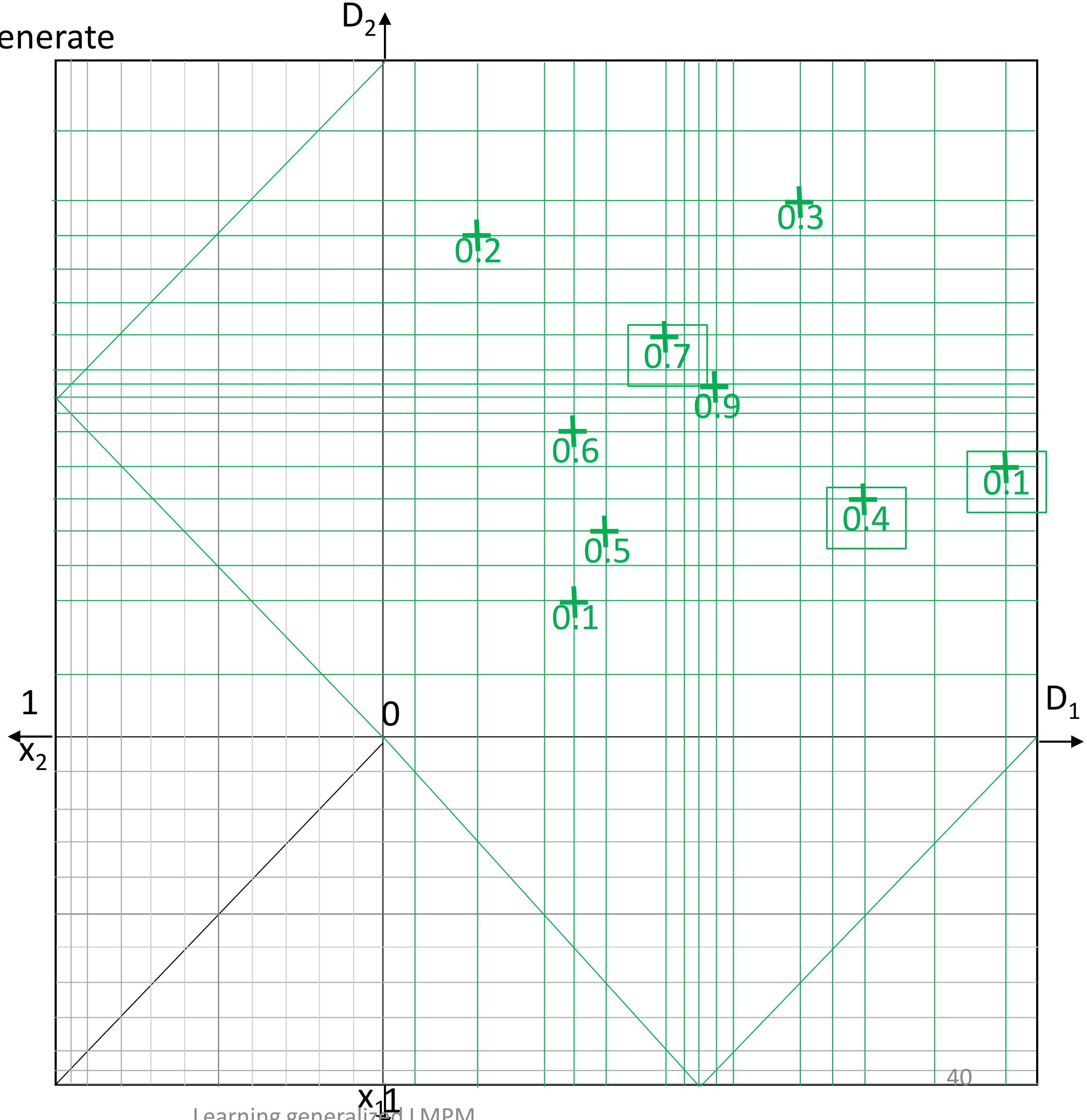
We do not know @ models and try conjunction and disjunction

Methods

a1 – or like fails

a2 – fit well

Rather low confidence

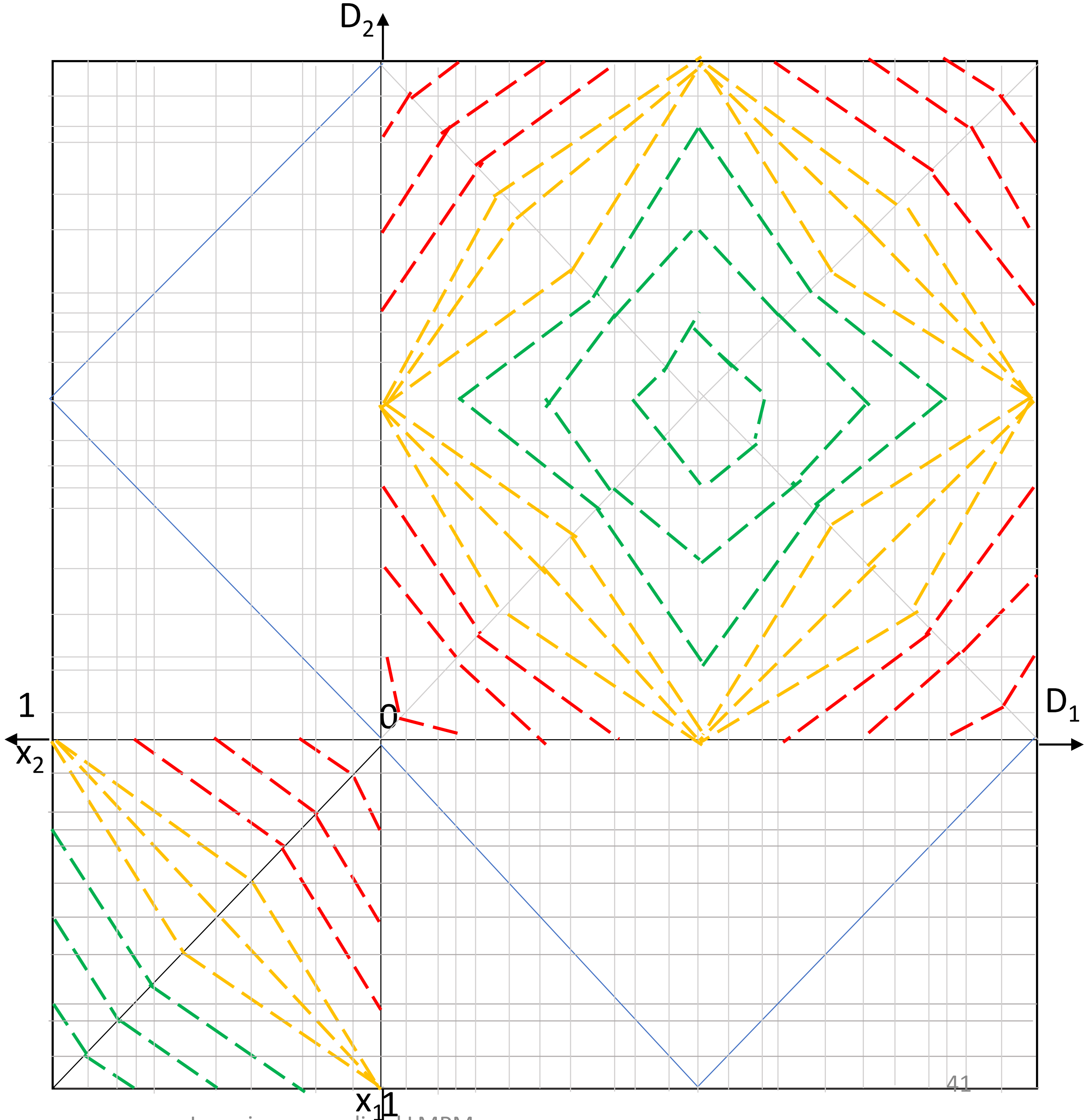


+ + +

Ex.3 **generate** regular contour lines in the PC,

Fix attribute preferences, construct corresponding images in DC and chose points in layers between contour lines in the DC

Now arbitrary choice in DC layers gives a learning task

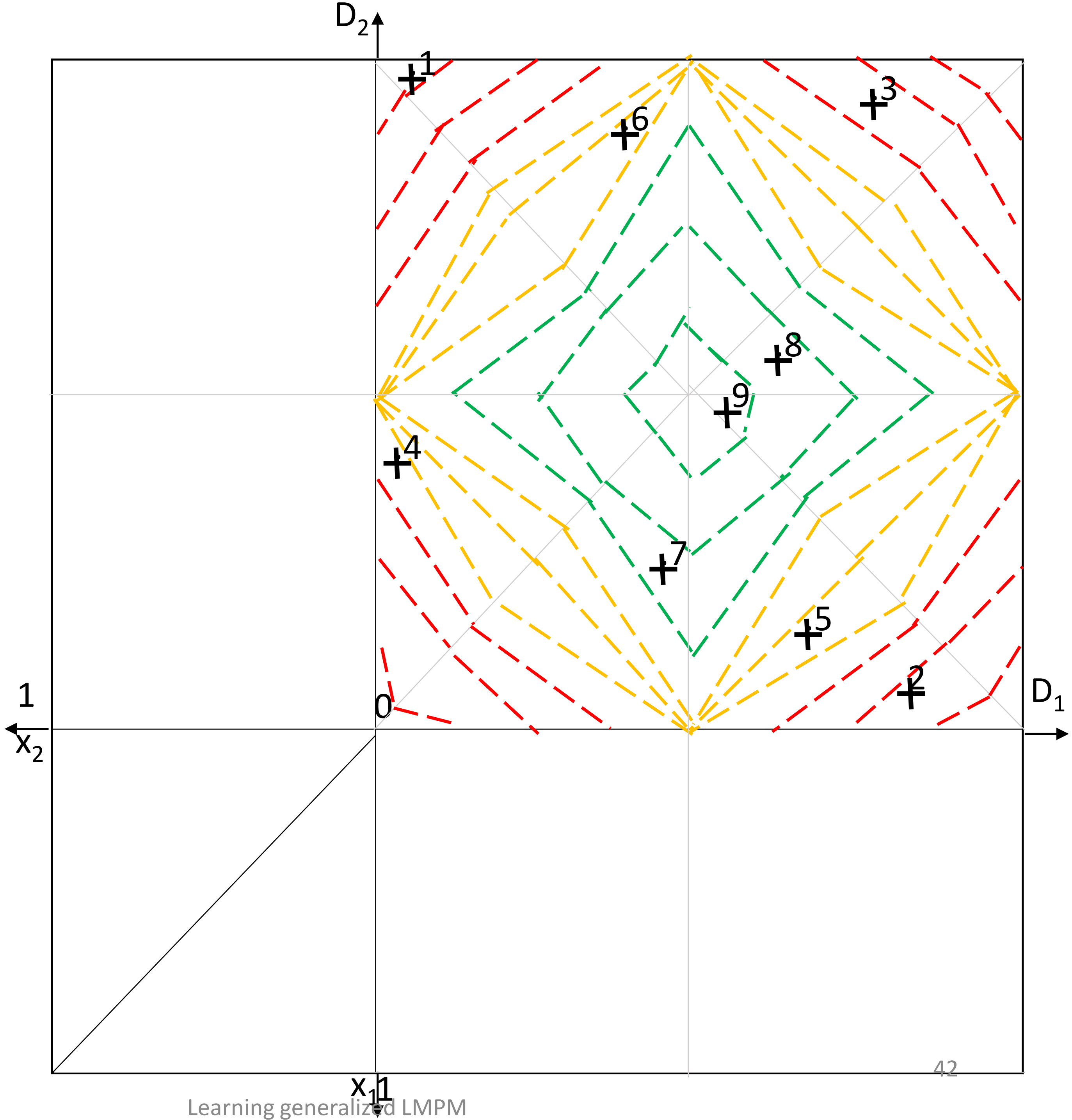


Ex.3 **generate** regular contour lines in the PC,

Fix attribute preferences, construct corresponding images in DC and chose points in layers between contour lines in the DC

Now arbitrary choice of points in DC layers gives a learning task

We can try to cheat some methods

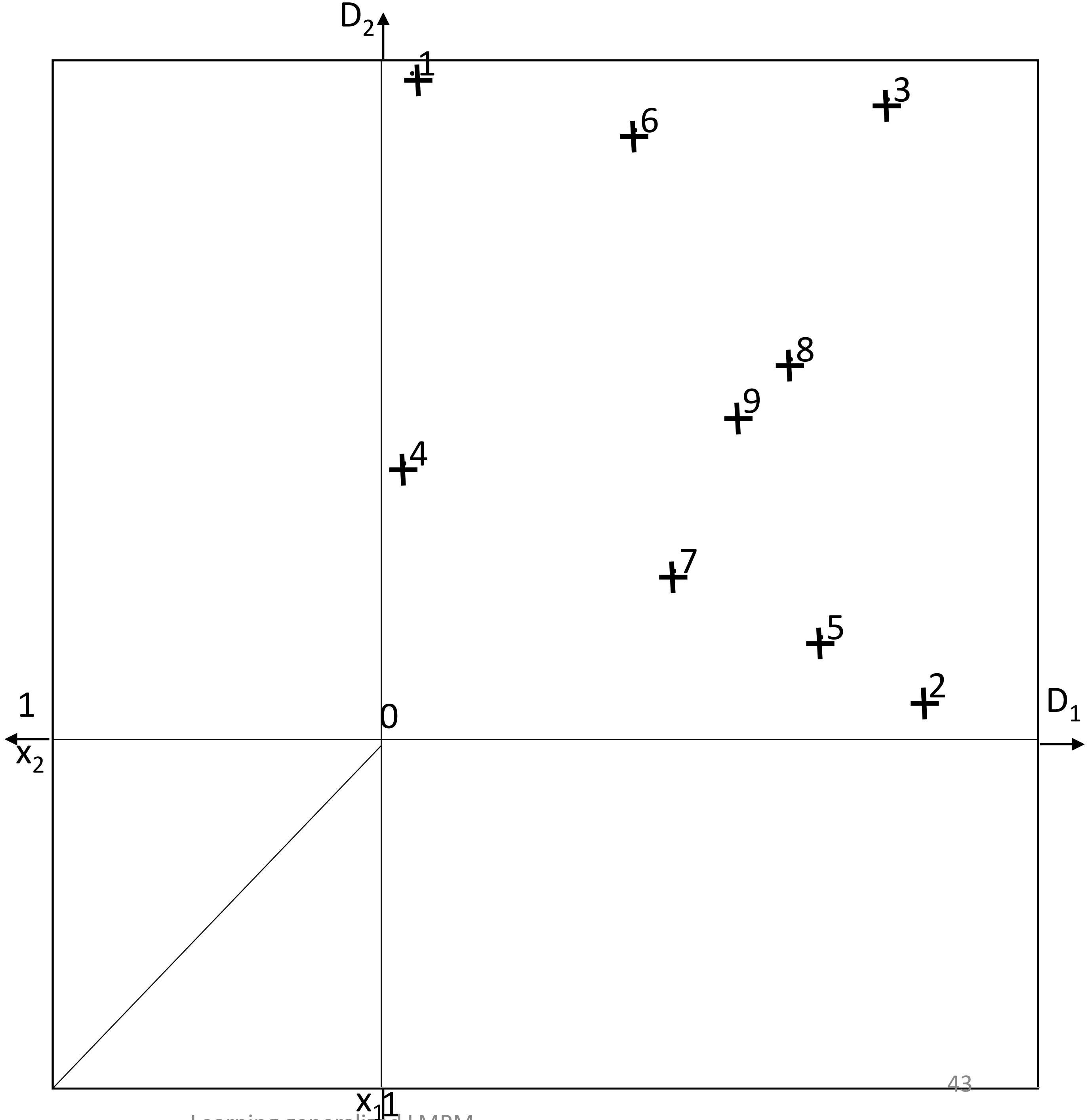


Ex.3 **learn and test** regular contour lines in the PC,

Fix attribute preferences, construct corresponding images in DC and chose points in layers between contour lines in the DC

Now arbitrary choice of points in DC layers gives a learning task

We can try to cheat some methods



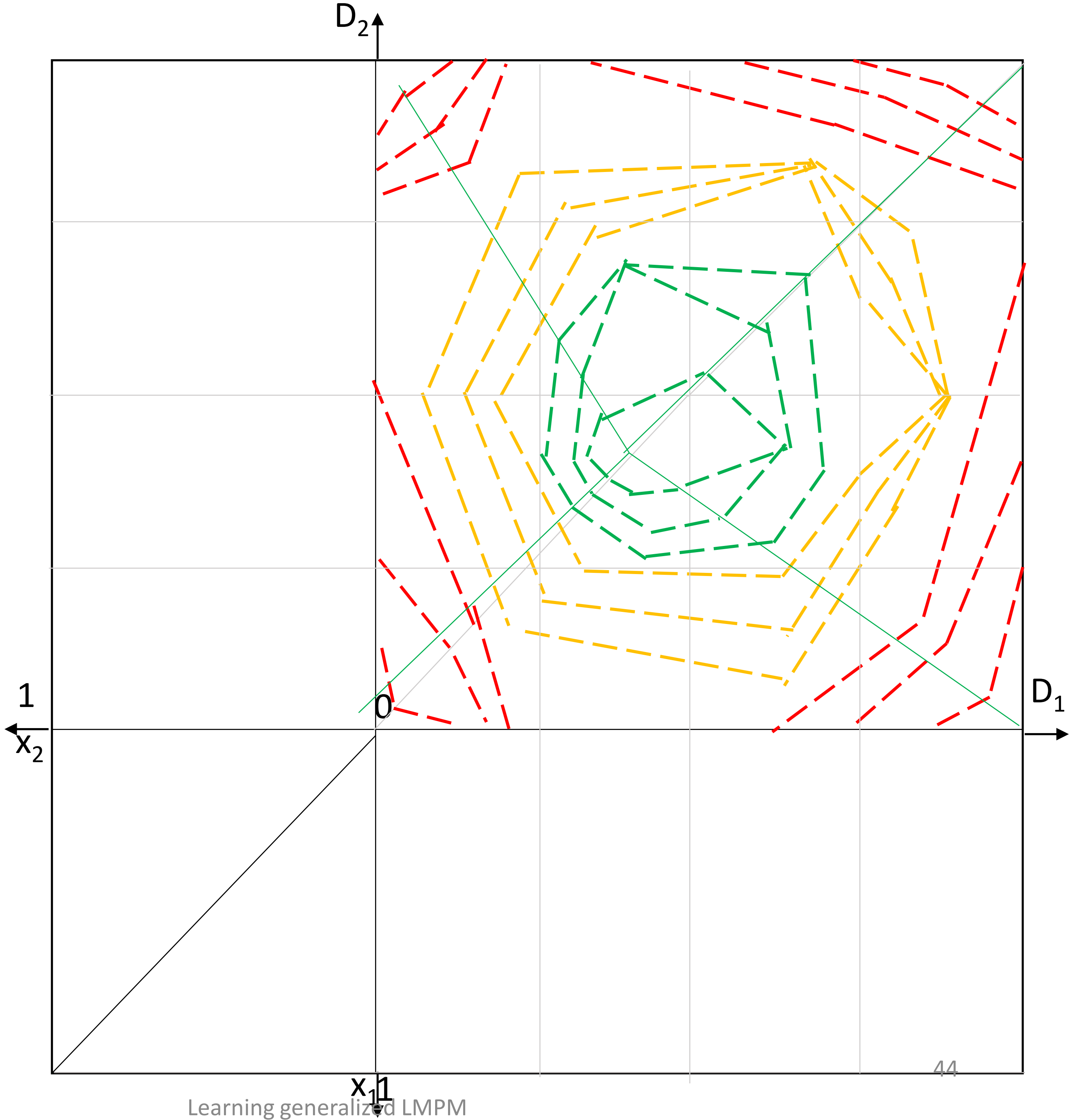
+ + +

Ex.3 design contour lines in DC that are not concentric

We can choose arbitrarily points in layers in DC

Here it seems we do not have any Pareto compliance

We must preserve only monotony



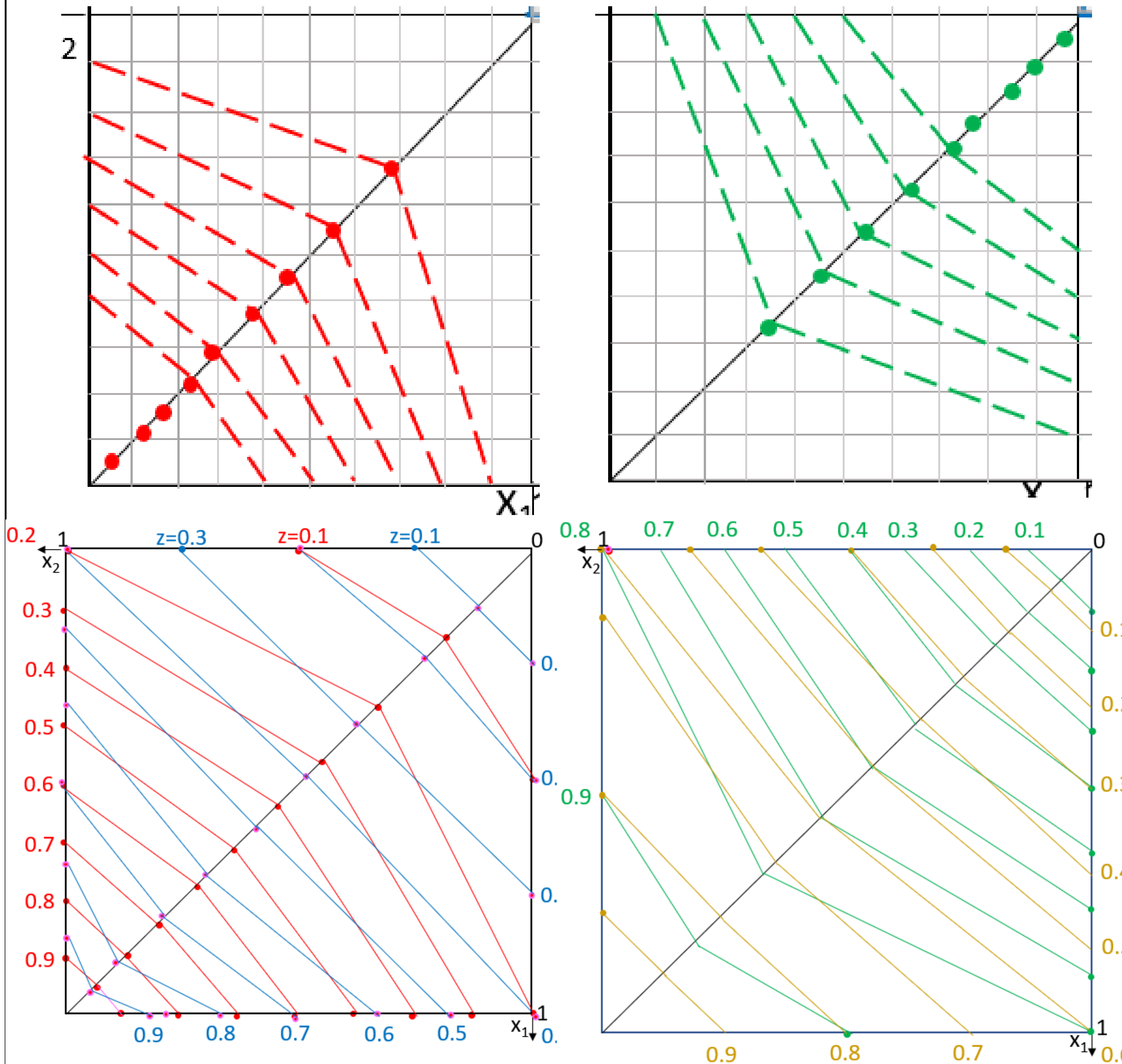
learning users so far

We did not try to learn convex combinations

Learning from data generated by product conjunction and disjunction showed success (partial)

Nevertheless, we have to give up exact behavior on the diagonal

This leads us to a class of connectives where point is on contour line connecting (some) point on the diagonal and respective point on the 0 or 1 axis (of course not violating order and without any intersections of contour lines)



Experiment 4.

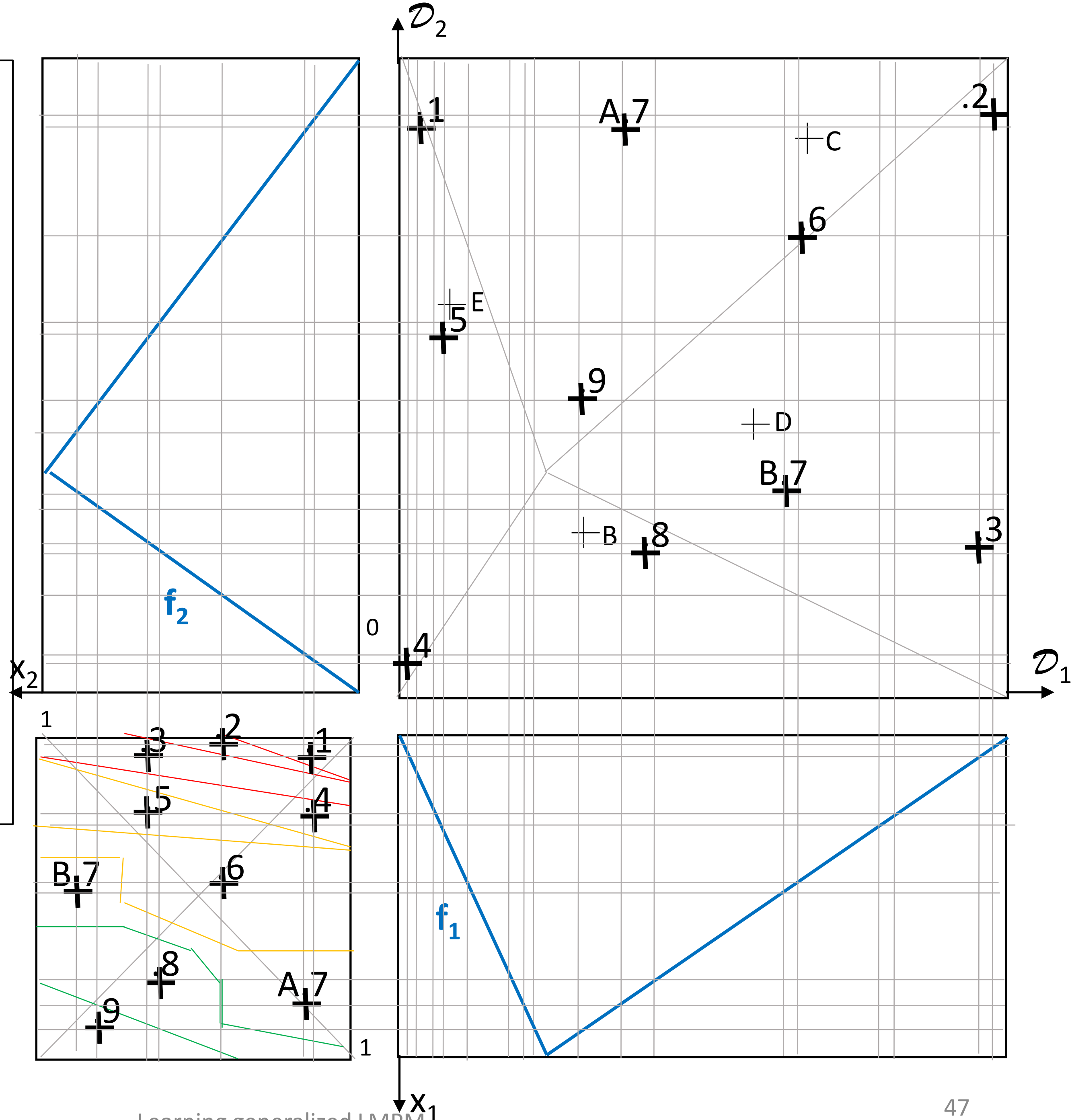
Arbitrary Pareto compliant contour lines

Ex4. Generate

Ex.4 Yet again a new idea – let's call it "**Wild guess**".

It assumes almost arbitrary borders between different preference degrees (z does not correlate with intersection of the diagonal)

Let's chose PC points in respective layers from (0,0) to (1,1) point

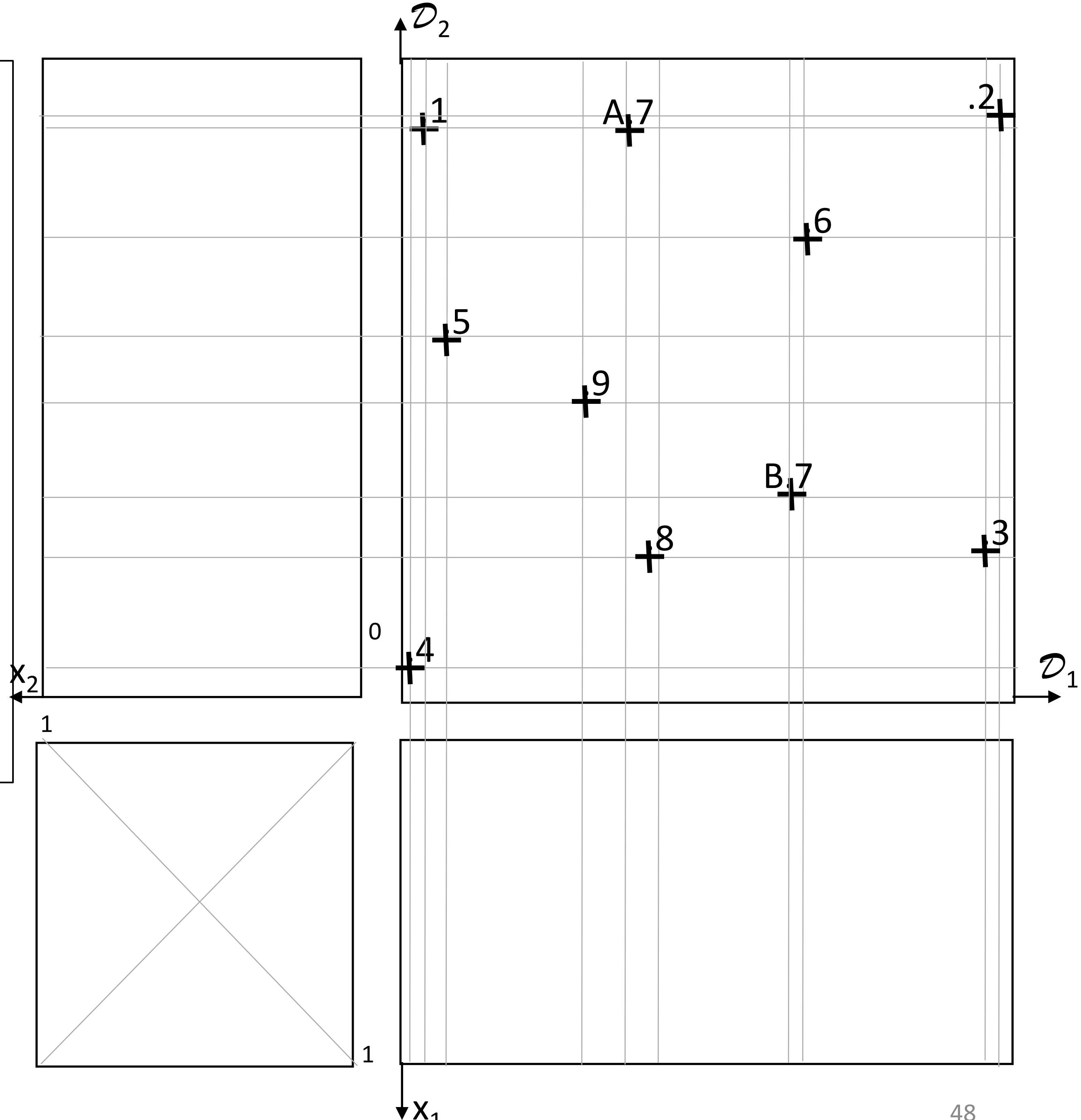


Ex4. **Train and test**

Ex.4 Yet again a new idea – let's call it "Wild guess".

It assumes almost arbitrary borders between different preference degrees (z does not correlate with intersection of the diagonal)

Let's chose PC points in respective layers from (0,0) to (1,1) point



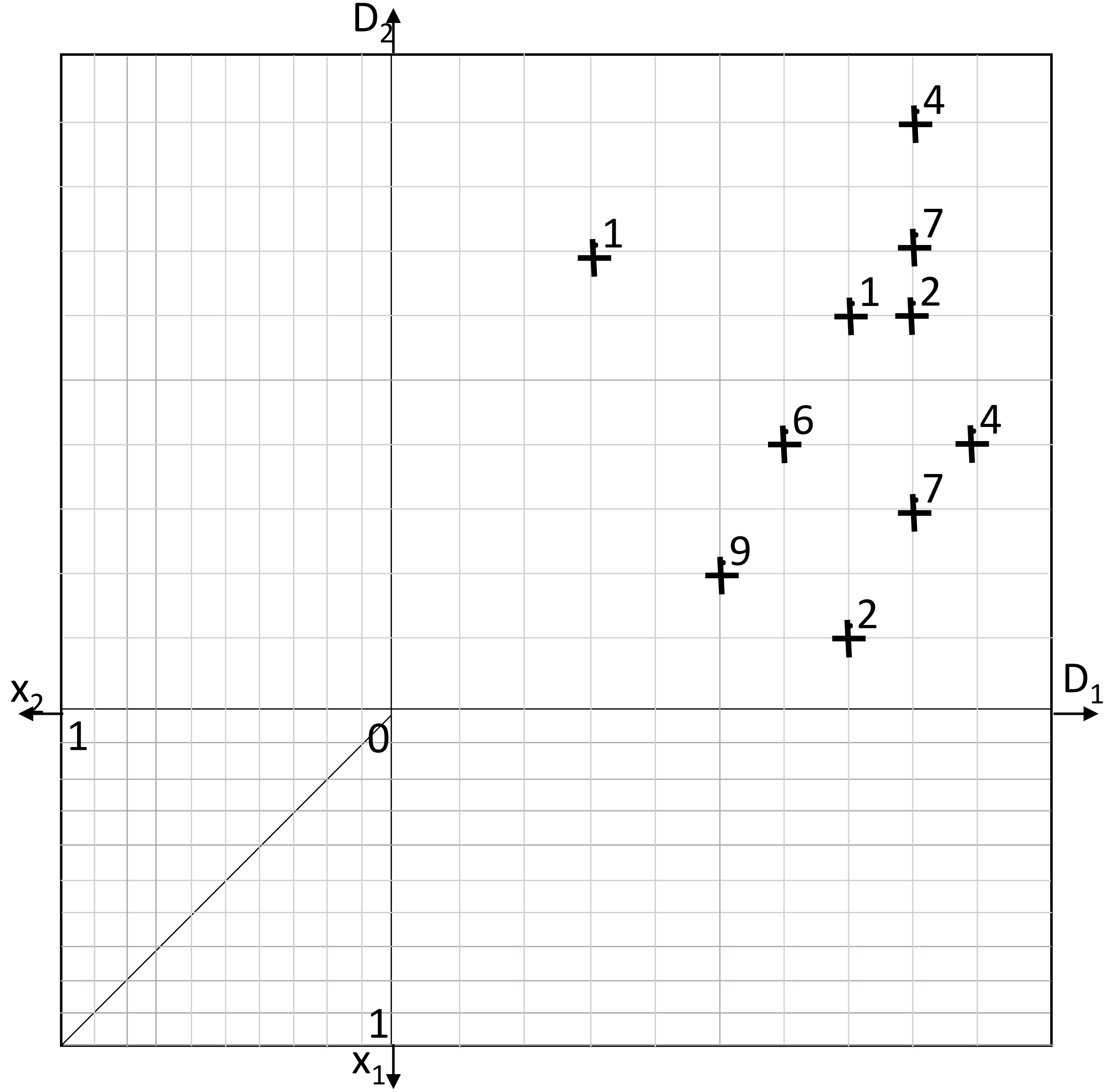
Randomly generated data

Ex4. **Train and test**

Ex.4 Yet again a new idea – let's generate randomly data in DC.

For each data point we need 3 numbers between 1 and 9 normalized domains are divided equidistant

(x_1,	x_2	score 0.z)
8	7	7
8	9	4
6	4	6
8	3	7
9	4	4
4	7	1
7	6	1
5	2	9
8	6	2
7	1	2



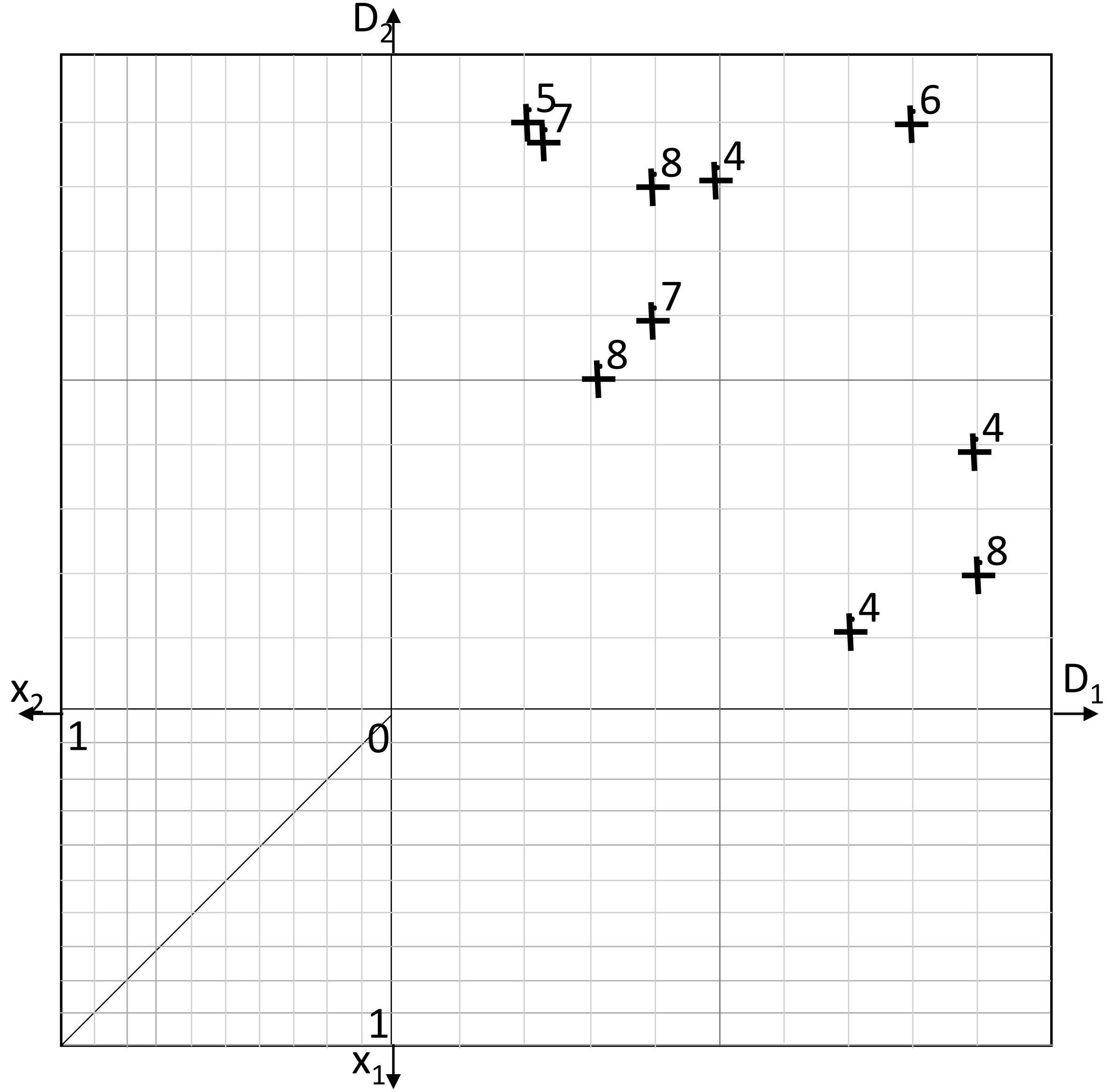
Randomly generated data

Ex4. Train and test

Ex.4 Yet again a new idea – let's generate randomly data in DC.

For each data point we need 3 numbers between 1 and 9 normalized domains are divided equidistant

(x_1,	x_2	score 0.z)
9	4	4
3	5	8
2	9	5
5	8	4
7	1	4
2	9	7
4	8	8
8	9	6
4	6	7
9	2	8

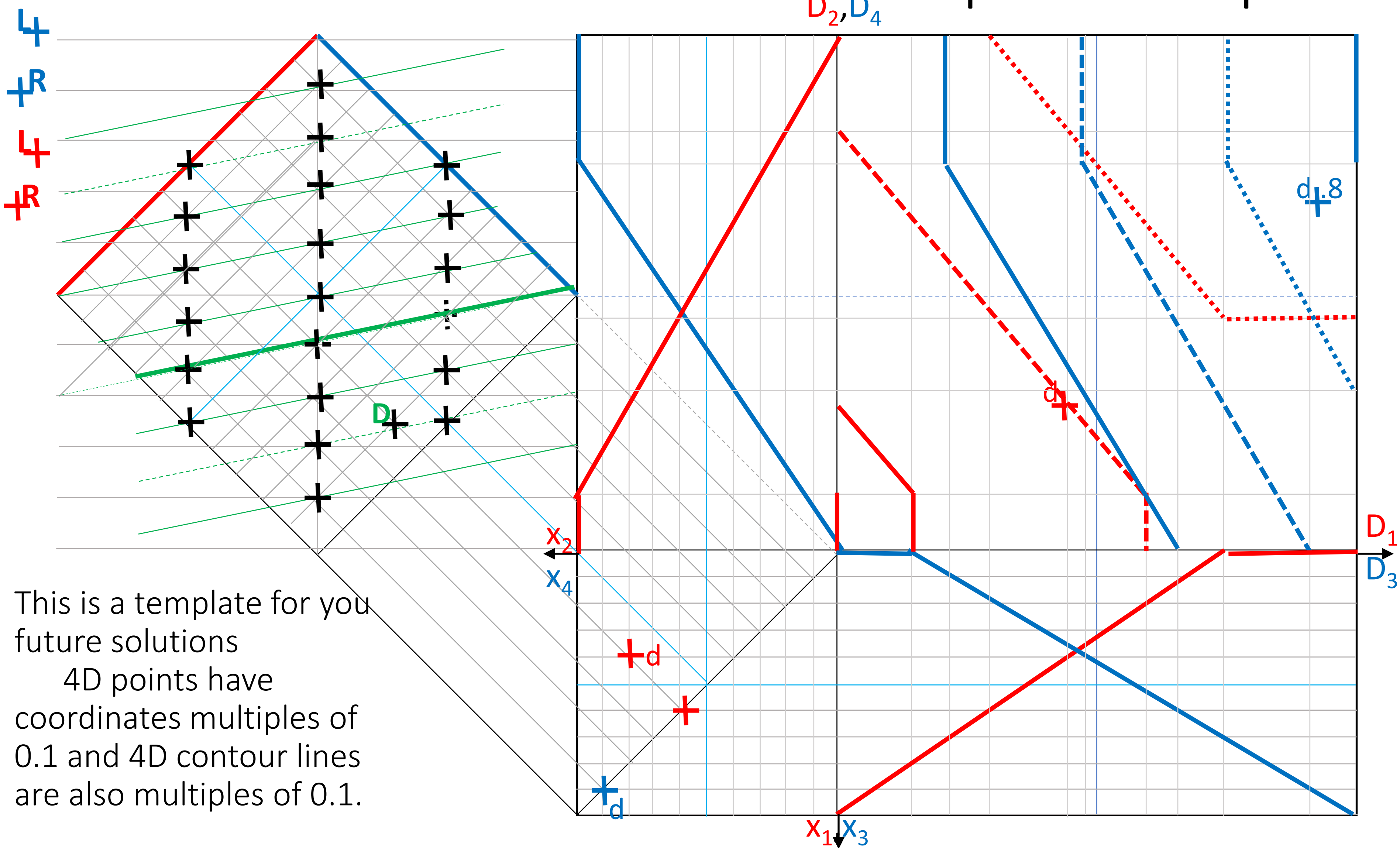


4-D learning?

advantage of visual insight disappeared

4d points easy to depict&compute

$$\frac{2 * \frac{x_1 + x_2}{2} + 3 * \frac{2 * x_3 + x_4}{3}}{5} = \frac{2 * y_{1,2} + 3 * y_{3,4}}{5} = z$$



This is a template for you future solutions

4D points have coordinates multiples of 0.1 and 4D contour lines are also multiples of 0.1.

Conclusions

- No libraries, paper work, one should be able to verify results given by a software
- Graphical motivation, data visualization
- Aggregation of partial results (attribute score, several recommenders, …)
- ChRF challenge response framework as in NSWI166 applies too, we reduce reality to models

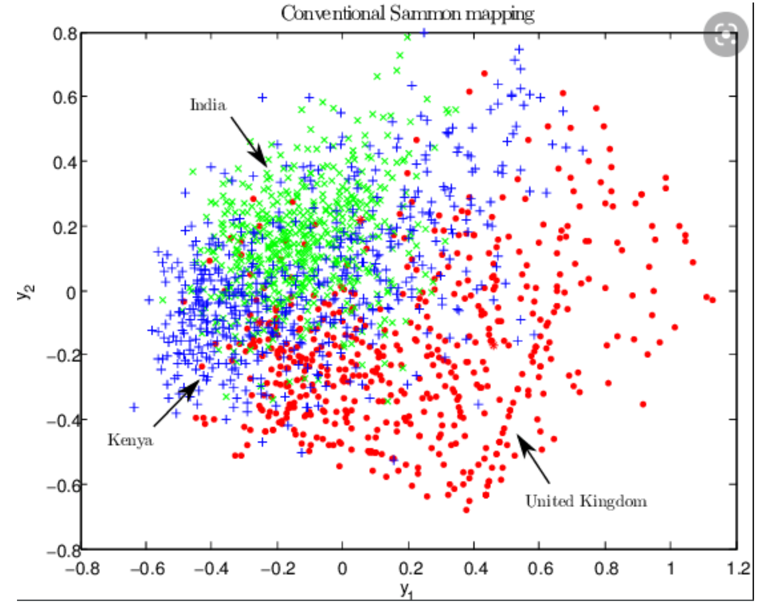
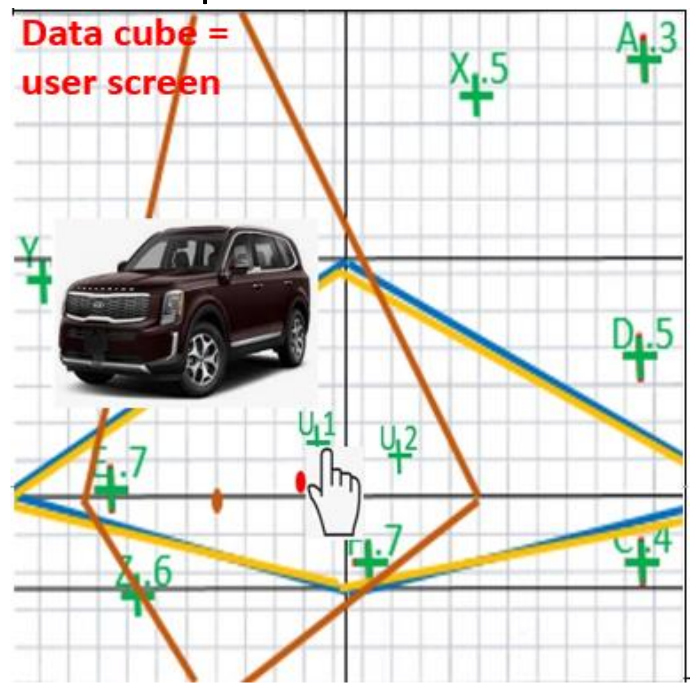
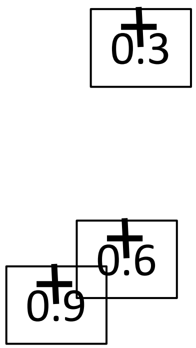


Image is only illustrative
Can we make visualization compliant with preferences?



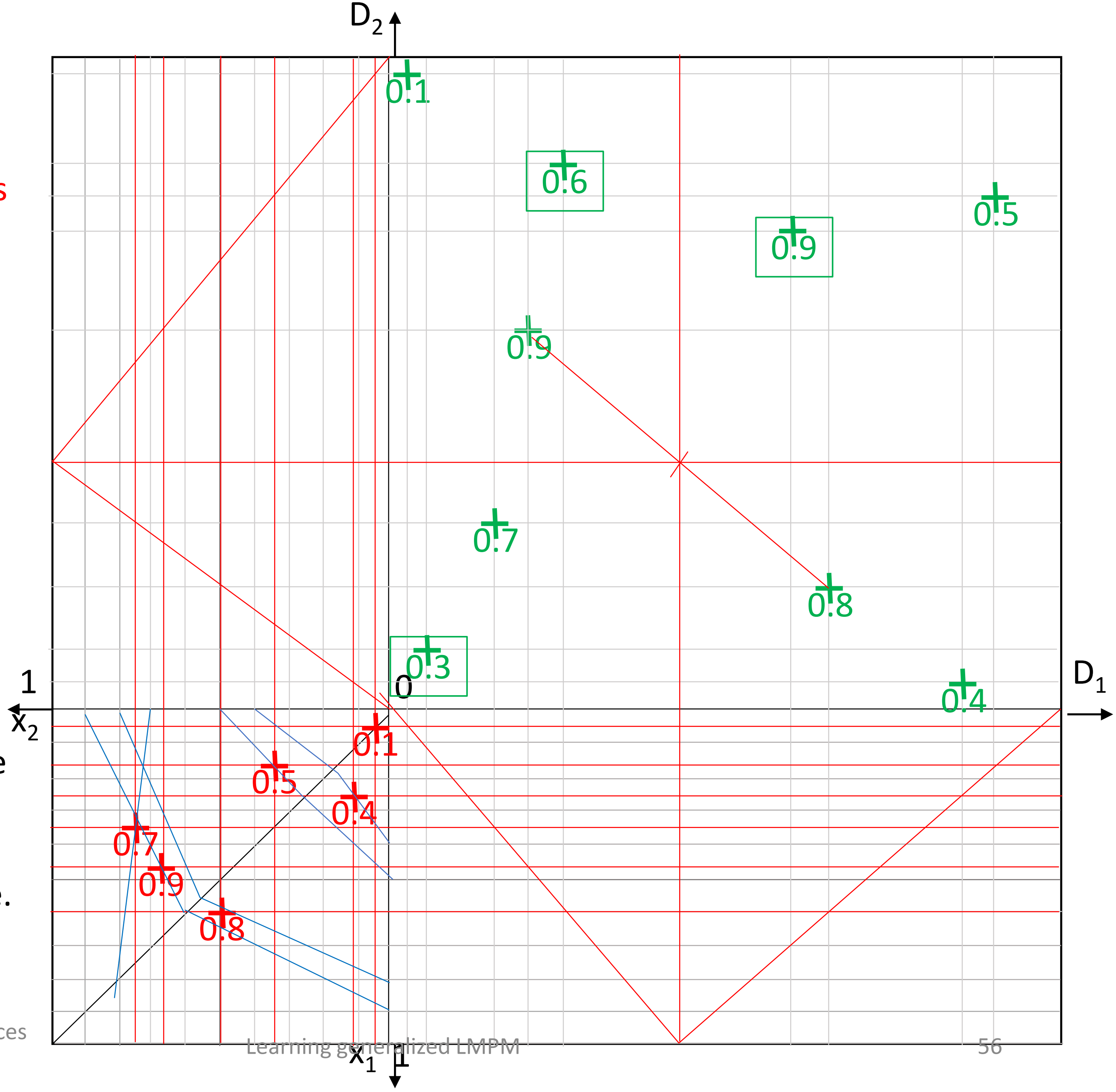
Thanks

Questions?


0.3
0.6
0.9

Ex.2.3 **method 3**.
Approximating missing contour lines

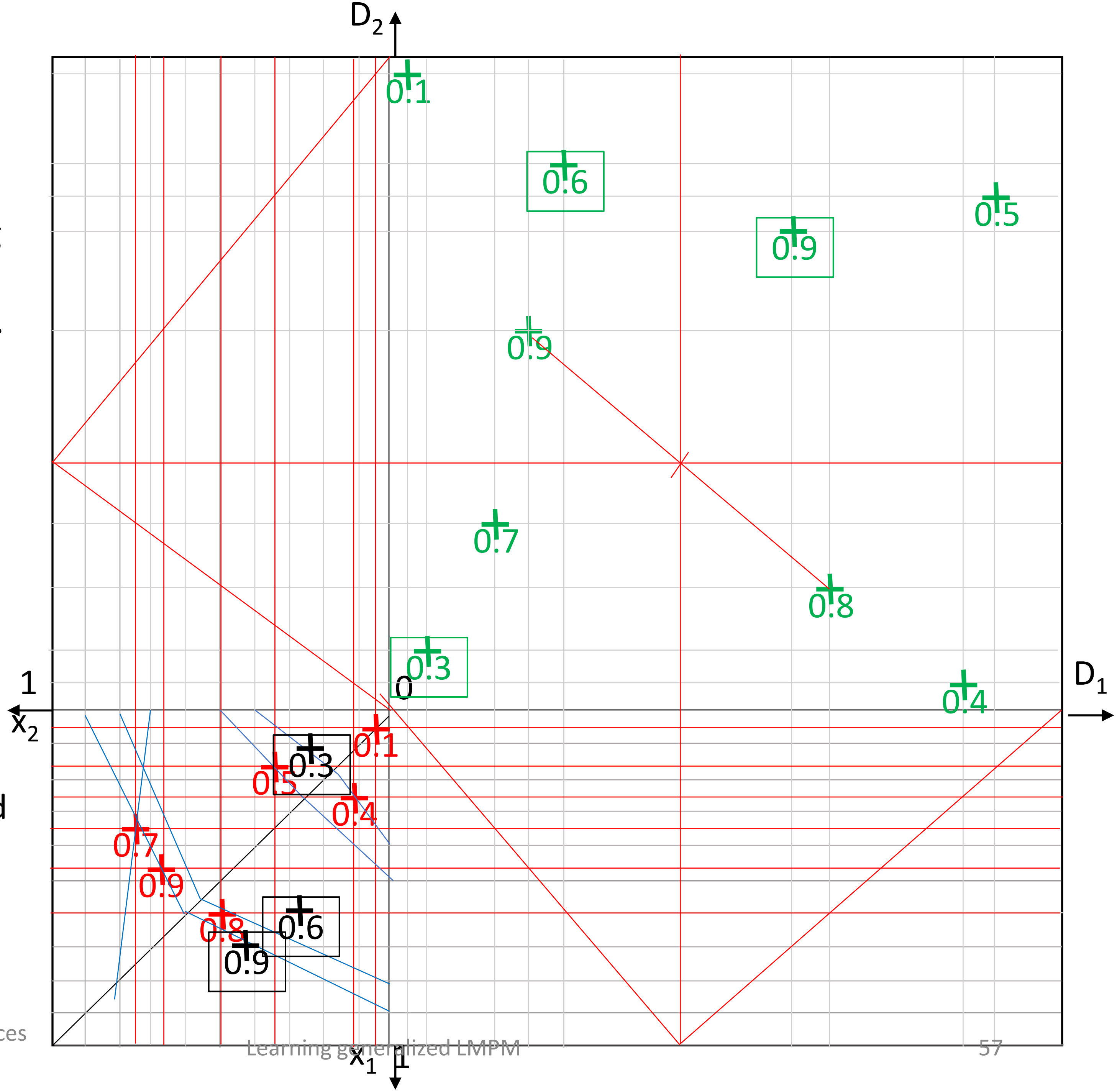
Data, attr. Prefs, training set in PC as before.
Let's try to **find agg. in form of many valued disjunction** – connecting points to known behavior on axis and corresponding intersection with diagonal. Confidence is 5/6 high – there is only one violation with 0.7 contour line. For test set we must approximate corresponding contour lines



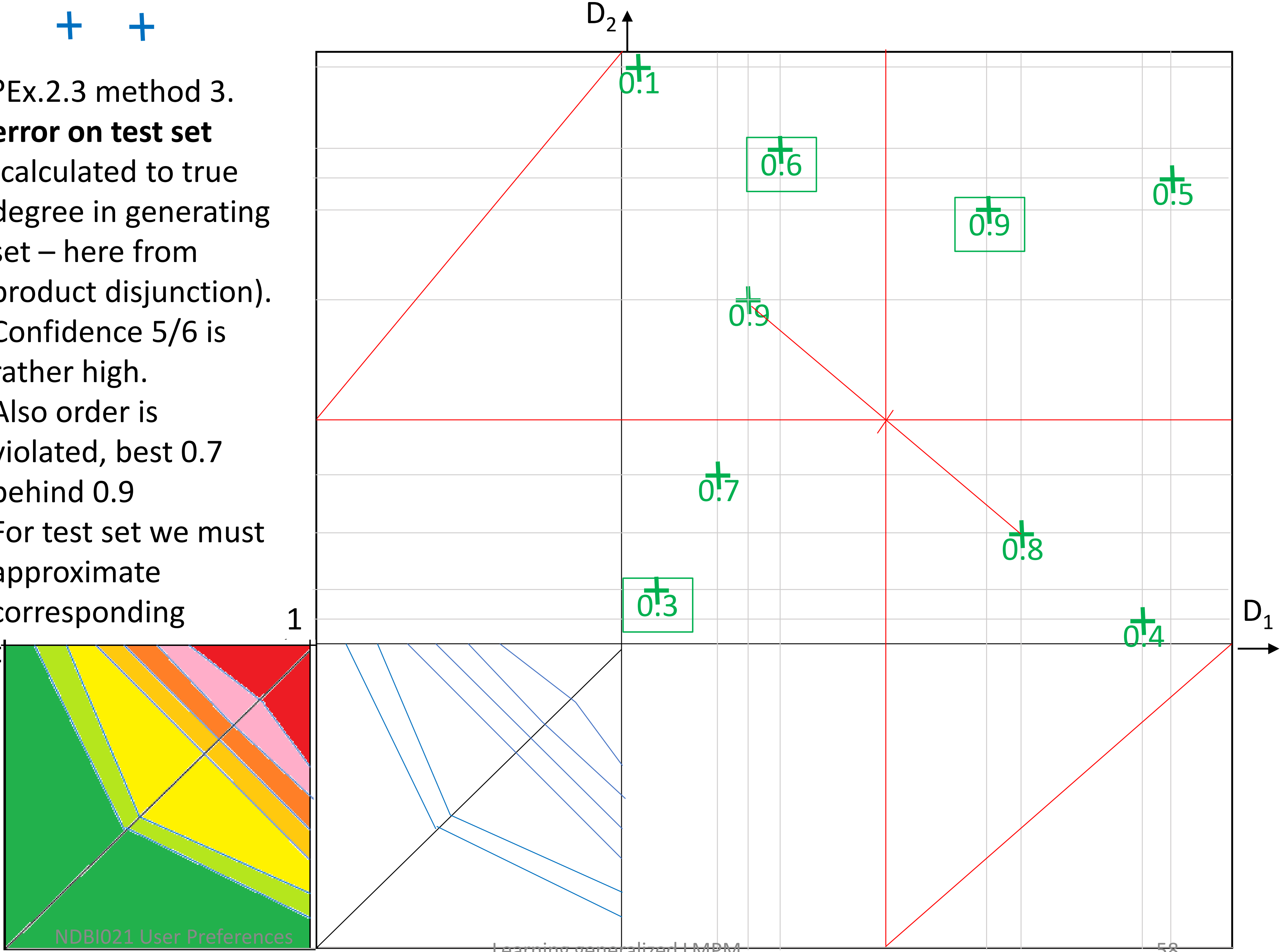
?Ex.2.3 method 3.
error on test set (calculated to true degree in generating set – here from product disjunction). Confidence 5/6 is rather high.

Also order is violated, best 0.7 behind 0.9

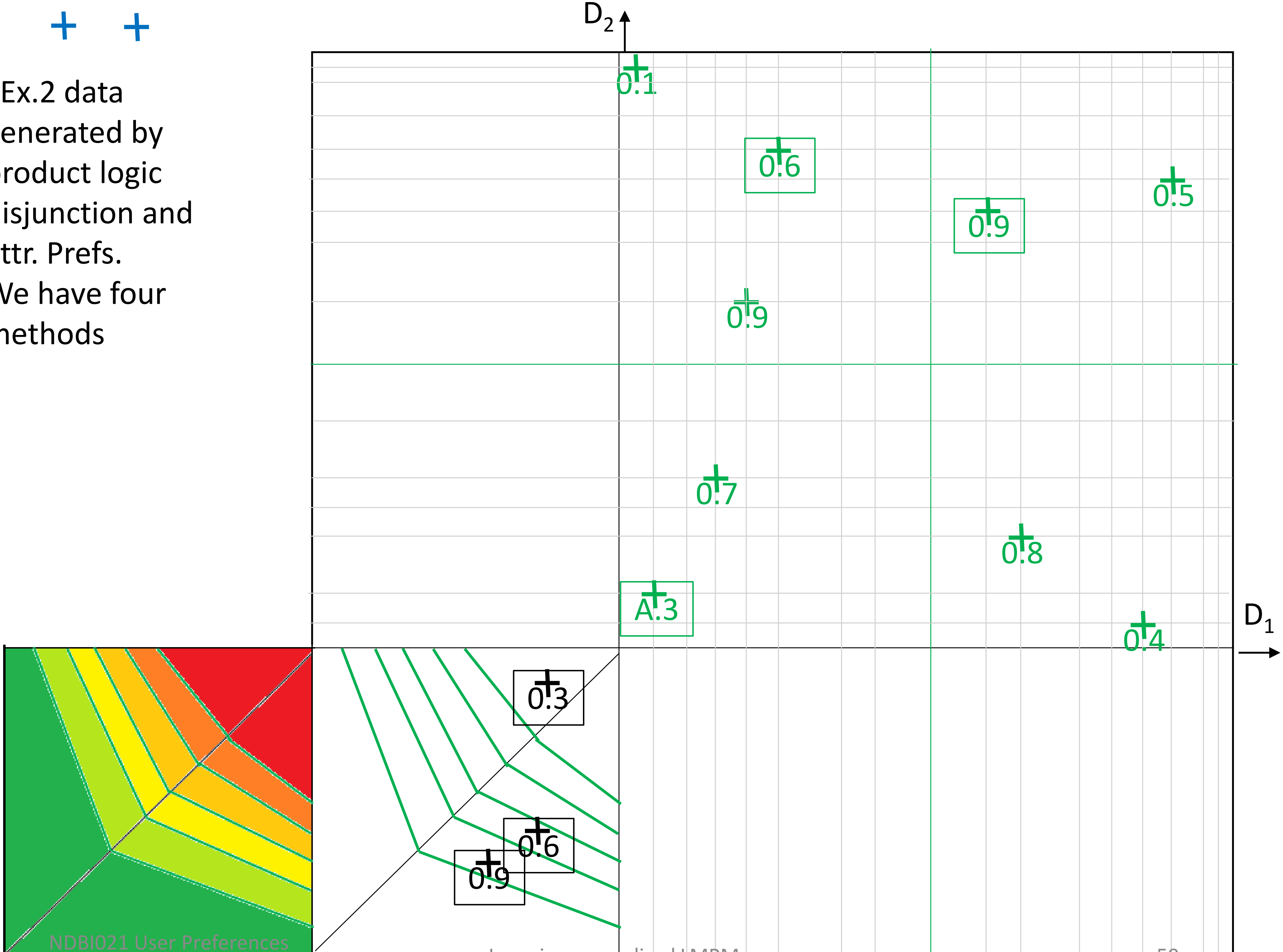
For test set we must approximate corresponding contour lines 0.6 and 0.7 in between contour lines 0.5 and 0.8



?Ex.2.3 method 3.
error on test set (calculated to true degree in generating set – here from product disjunction). Confidence 5/6 is rather high.
Also order is violated, best 0.7 behind 0.9
For test set we must approximate corresponding c

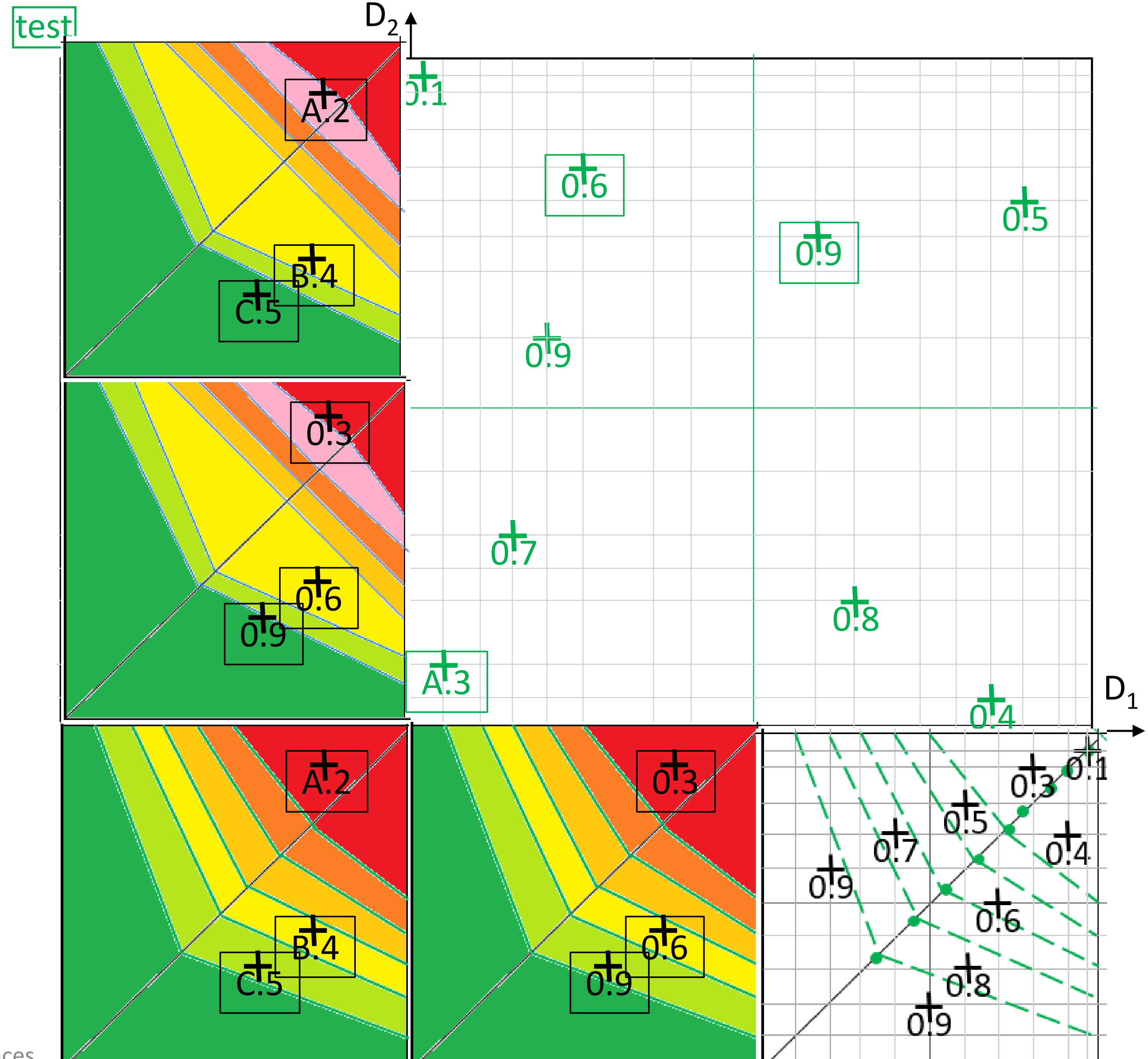


?Ex.2 data generated by product logic disjunction and attr. Prefs.
We have four methods



?Ex.2 data generated by product logic disjunction and attr. Prefs.

We have four methods



Randomly generated data