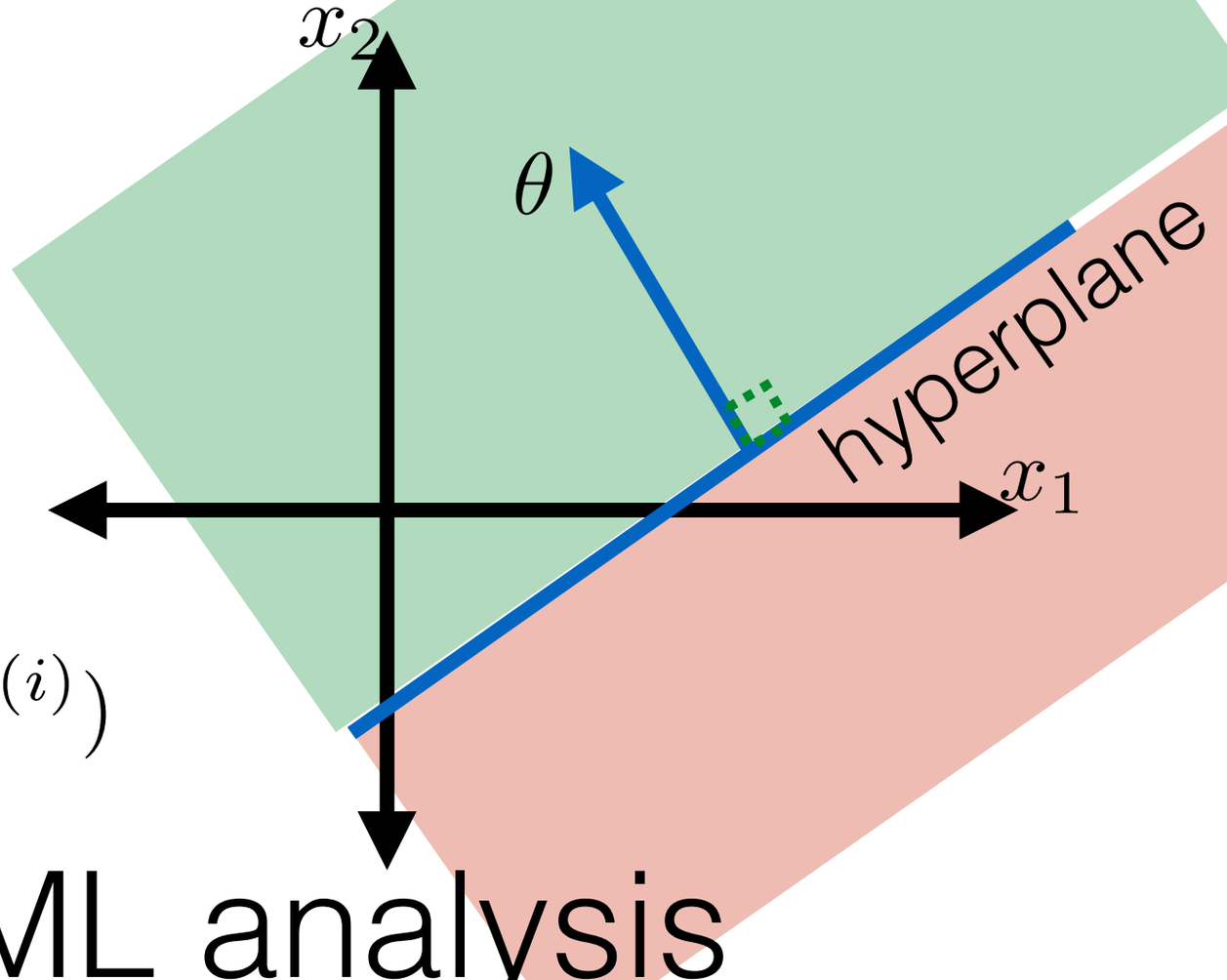
Recall

- Linear classifier h
- 0-1 Loss

$$L(g, a) = \begin{cases} 0 \text{ if } g = a \\ 1 \text{ else} \end{cases}$$

- Training error

$$\mathcal{E}_n(h) = \frac{1}{n} \sum_{i=1}^{n} L(h(x^{(i)}), y^{(i)})$$



A more-complete ML analysis

1. Establish a goal & find data
   - Example goal: diagnose whether people have heart disease based on their available information
2. Encode data in useful form for the ML algorithm
3. Run the ML algorithm & return a classifier
   - Example algorithms: (A) choose best classifier from a finite list; (B) perceptron; (C) averaged perceptron
4. Interpretation & evaluation

A machine learning (ML) analysis

- First, need goal & data. E.g. diagnose whether people have heart disease based on their available information
- Next, put data in useful form for learning algorithm

	has heart disease?	resting heart rate (bpm)	pain?	job	medicines	age	family income (USD)
1	no	55	no	nurse	pain	40s	133000
2	no	71	no	admin	beta blockers, pain	20s	34000
3	yes	89	yes	nurse	beta blockers	50s	40000
4	no	67	no	doctor	none	50s	120000

Encode data in usable form

- Identify the labels and encode as real numbers

	has heart disease?
1	no
2	no
3	yes
4	no

$$\{\text{‘yes’,‘no’}\} \leftrightarrow \{+1, -1\}$$

	$y^{(1)}$
1	-1
2	-1
3	+1
4	-1

- Depending on your algorithm, might instead use $\{0, 1\}$
- Save mapping to recover predictions of new points

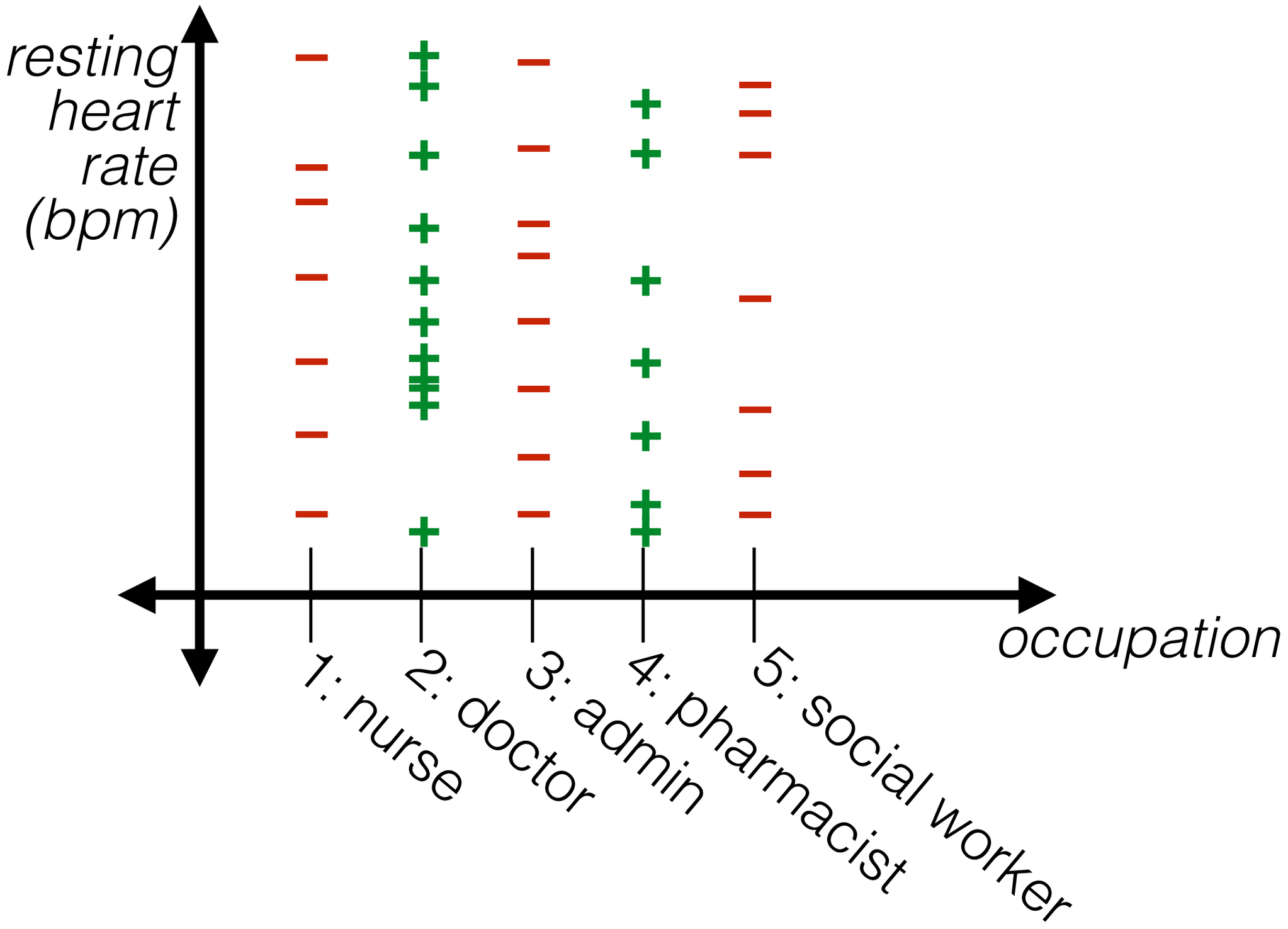
Encode data in usable form

- Identify the features and encode as real numbers
- Feature: any function of the data (except labels)
- Today, old features: x; new features: $\phi(x)$

	resting heart rate (bpm)	pain?	job	medicines	age	family income (USD)
1	55	0	nurse	pain	40s	133000
2	71	0	admin	beta blockers, pain	20s	34000
3	89	1	nurse	beta blockers	50s	40000
4	67	0	doctor	none	50s	120000

Encode categorical data

- Idea: turn each category into a unique natural number



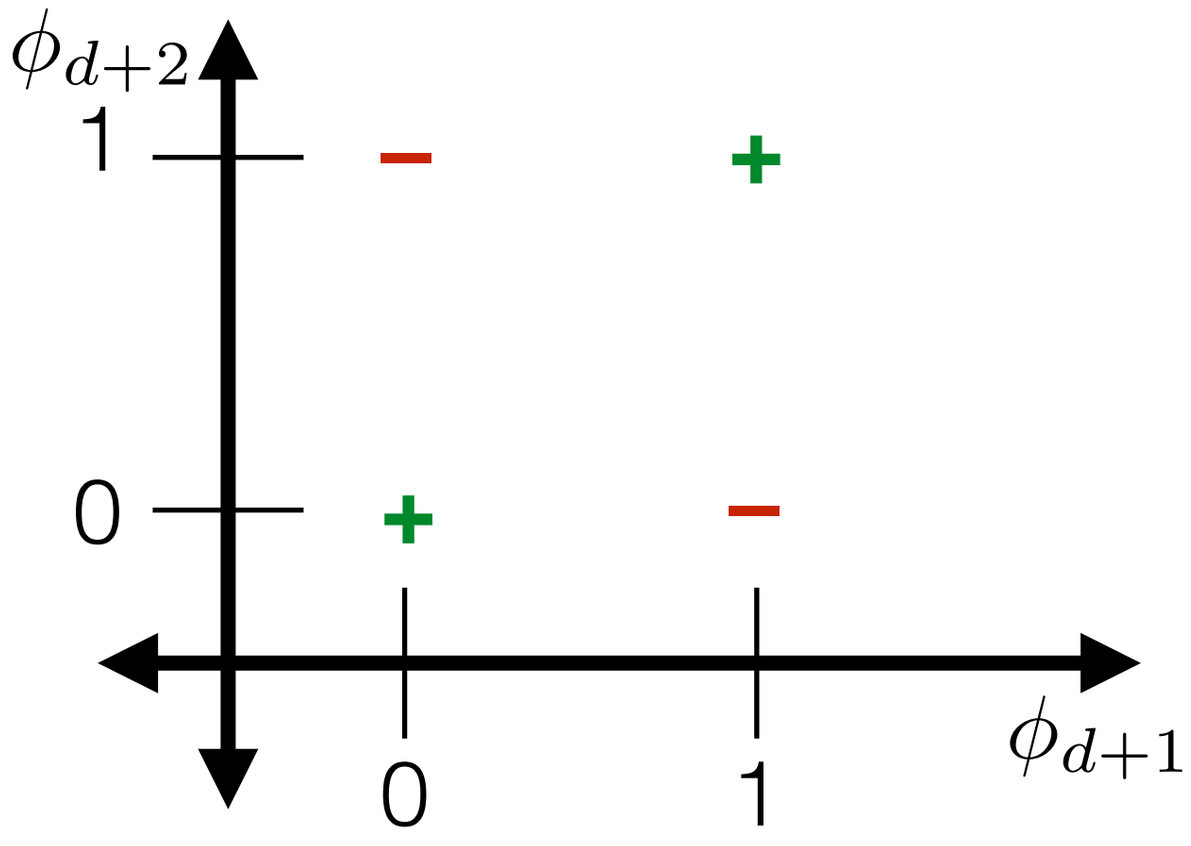
★ Encoding Categorical data (Slide - 7 -)

- If you ~~go~~ map categorical data to unique natural numbers, then even if you don't want, you have implicitly ordered the data or assigned rank to it.
- And you can imagine, if you have 20 features (categoric) mapped to natural numbers & you take dot products, you have unintentionally weighed some features more than others due to their high value of natural number.

Encode categorical data

- Idea: turn each category into a unique binary number

	ϕ_d	ϕ_{d+1}	ϕ_{d+2}
nurse	0	0	0
admin	0	0	1
pharmacist	0	1	0
doctor	0	1	1
social worker	1	0	0



★ Encoding categorical data (slide - 9 -)

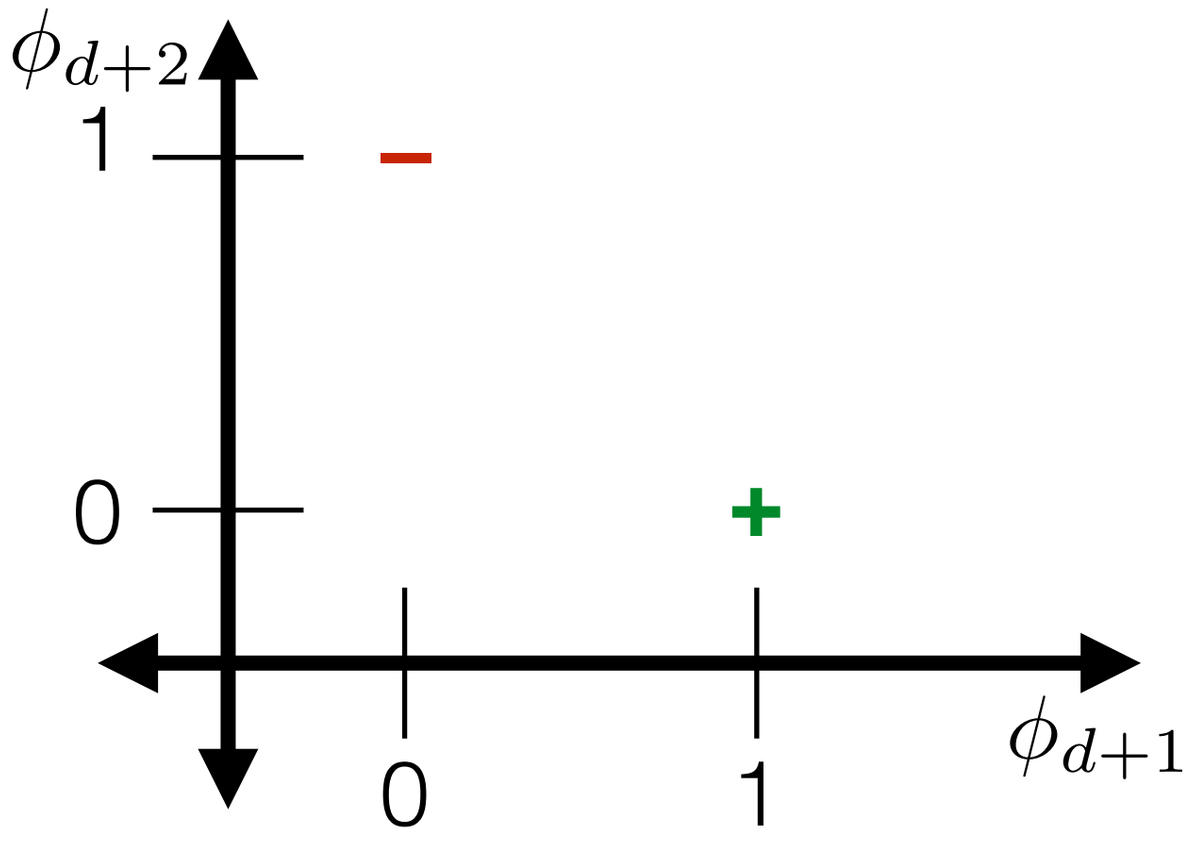
- Imagine you have 'n' features & you decide that each feature must get a unique binary representation so that the mapping is distributed & you don't assign higher values to some & lower values to others like in case of natural number mapping.
- For 'n' features, we would need ceiling($\log_2 n$) no. of binary bits. & assign 'ceiling($\log_2 n$) - 1' extra features for their representation.
- But then, like given in the slide, you would again group some categories together for a feature. Like you would group admin & doctor together for ϕ_{d+2} (see the slide) without it meaning anything. Same for pharmacist & doctor for ϕ_{d+1}.
- So, you need to find an even better way to encode categorical data.

Encode categorical data

- Idea: turn each category into own unique 0-1 feature

	ϕ_d	ϕ_{d+1}	ϕ_{d+2}	ϕ_{d+3}	ϕ_{d+4}
nurse	1	0	0	0	0
admin	0	1	0	0	0
pharmacist	0	0	1	0	0
doctor	0	0	0	1	0
social worker	0	0	0	0	1

- "one-hot encoding"



Encode data in usable form

- Identify the features and encode as real numbers

	resting heart rate (bpm)	pain?	j1,j2,j3,j4,j5	medicines	age	family income (USD)
1	55	0	1,0,0,0,0	pain	40s	133000
2	71	0	0,1,0,0,0	beta blockers, pain	20s	34000
3	89	1	1,0,0,0,0	beta blockers	50s	40000
4	67	0	0,0,0,1,0	none	50s	120000

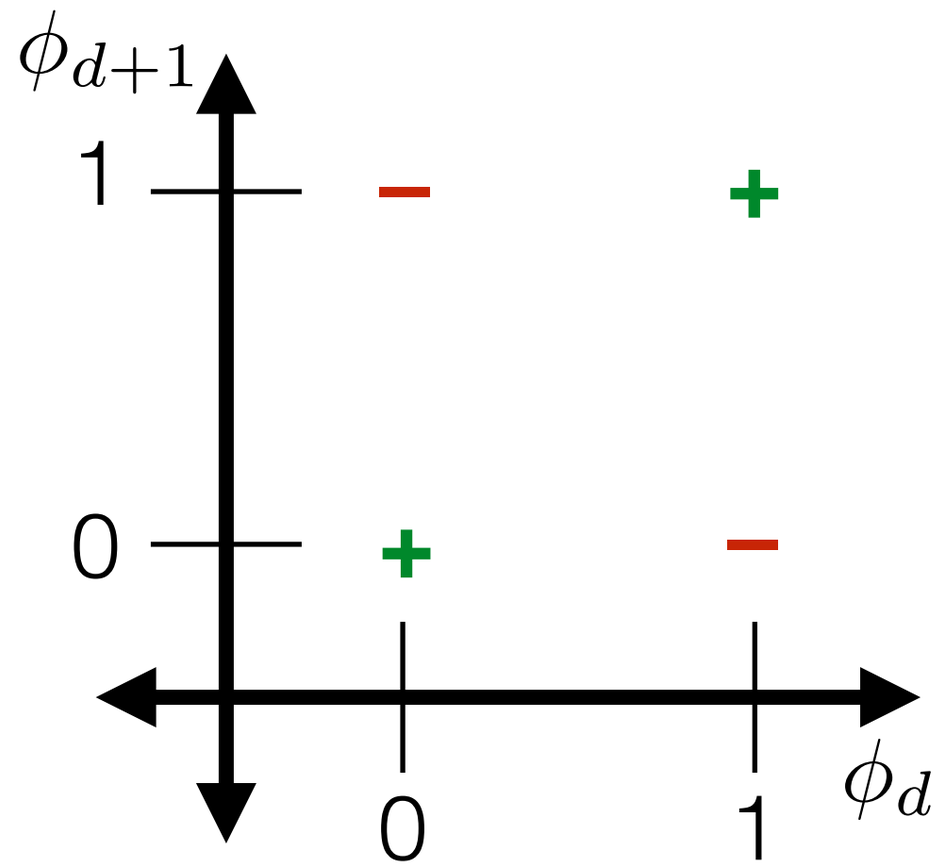
Encode categorical data

- Should we use one-hot encoding?

	ϕ_d	ϕ_{d+1}	ϕ_{d+2}	ϕ_{d+3}
pain	1	0	0	0
pain & beta blockers	0	1	0	0
beta blockers	0	0	1	0
no medications	0	0	0	1

- Idea: factored encoding

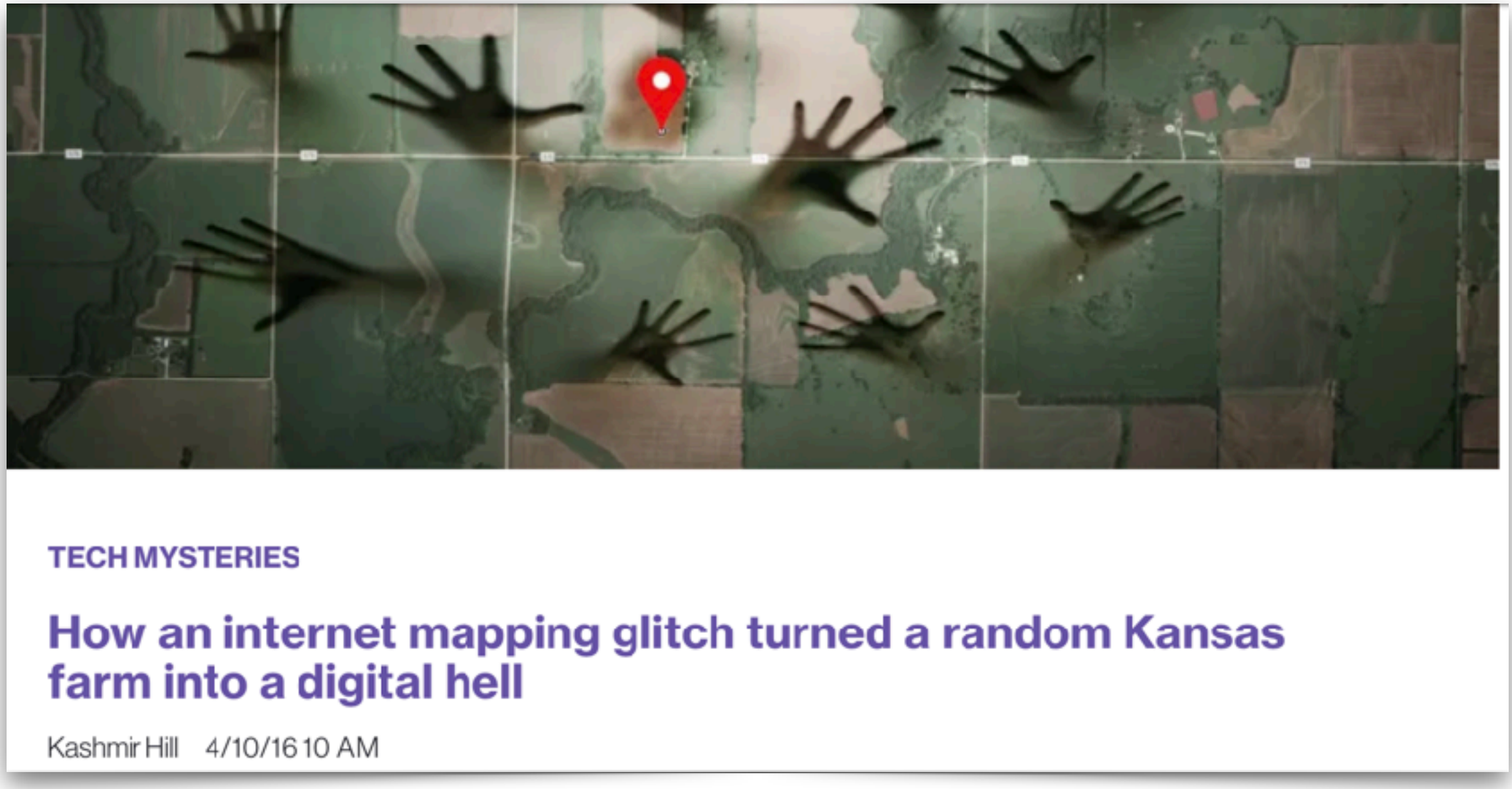
	ϕ_d	ϕ_{d+1}
pain	1	0
pain & beta blockers	1	1
beta blockers	0	1
no medications	0	0



Using a representative # for a range

- Potential pitfall: level of detail might be treated as meaningful (by you or others using the data)
- A way to diagnose many problems: plot your data!

age
45
25
55
55



TECH MYSTERIES

How an internet mapping glitch turned a random Kansas farm into a digital hell

Kashmir Hill 4/10/16 10 AM

[https://fusion.tv/story/287592/internet-mapping-glitch-kansas-farm/?utm_medium=social]

Encode data in usable form

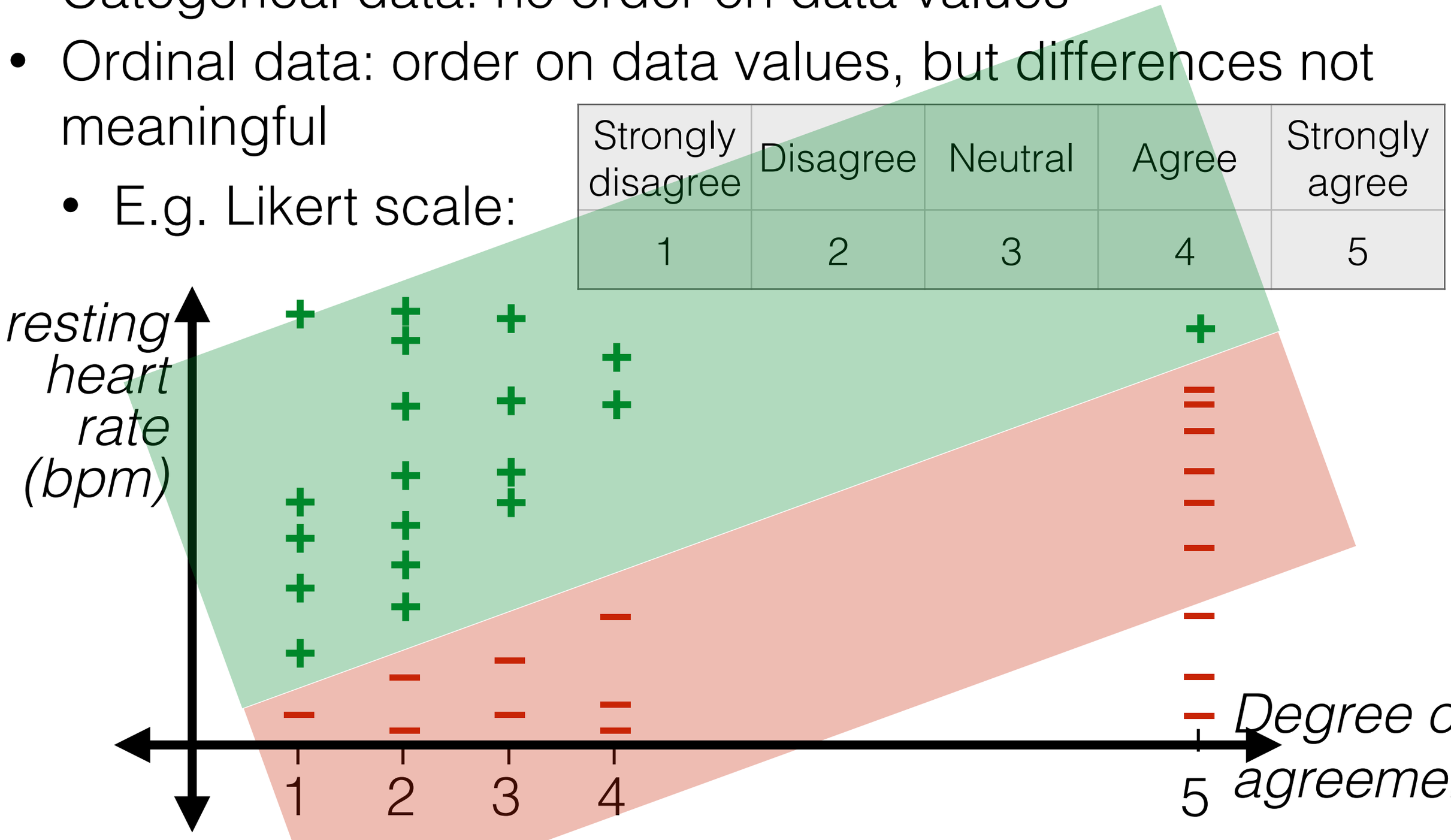
- Identify the features and encode as real numbers

	resting heart rate (bpm)	pain?	j1,j2,j3,j4,j5	m1, m2	decade	family income (USD)
1	55	0	1,0,0,0,0	1,0	4	133000
2	71	0	0,1,0,0,0	1,1	2	34000
3	89	1	1,0,0,0,0	0,1	5	40000
4	67	0	0,0,0,1,0	0,0	5	120000

Encode ordinal data

- Numerical data: order on data values, and differences in value are meaningful
- Categorical data: no order on data values
- Ordinal data: order on data values, but differences not meaningful
  - E.g. Likert scale:

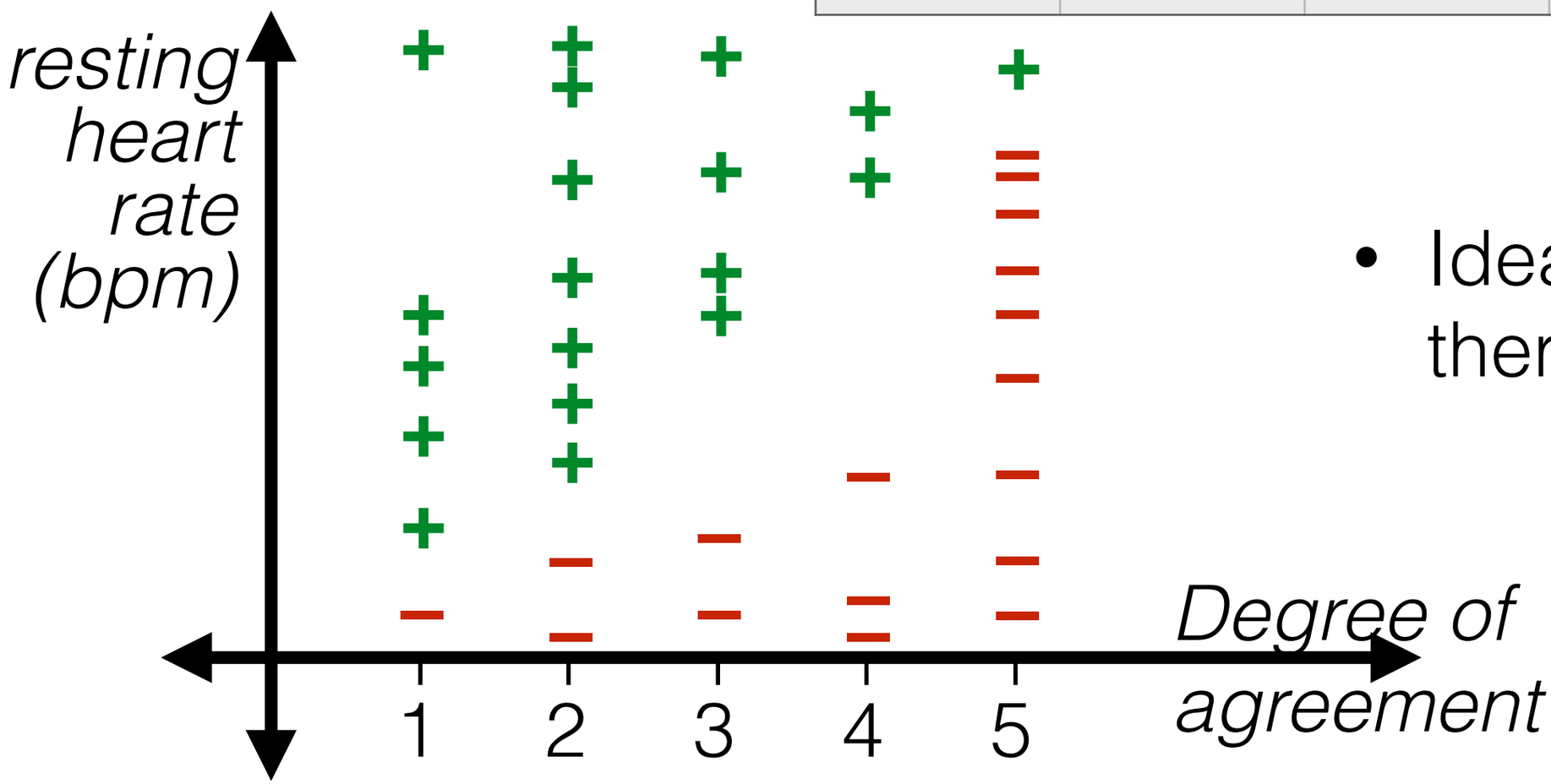
Strongly disagree	Disagree	Neutral	Agree	Strongly agree
1	2	3	4	5



Encode ordinal data

- Numerical data: order on data values, and differences in value are meaningful
- Categorical data: no order on data values
- Ordinal data: order on data values, but differences not meaningful
  - E.g. Likert scale:

Strongly disagree	Disagree	Neutral	Agree	Strongly agree
1,0,0,0,0	1,1,0,0,0	1,1,1,0,0	1,1,1,1,0	1,1,1,1,1



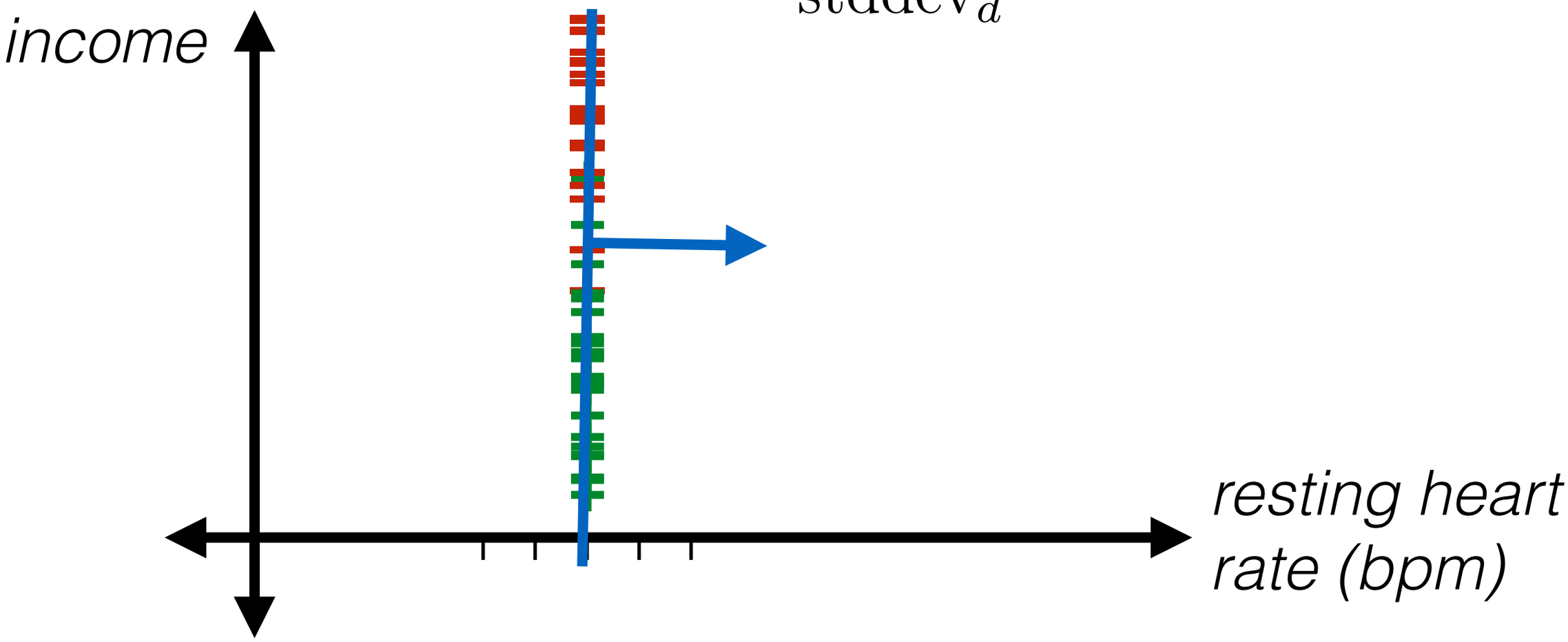
- Idea: Unary/ thermometer code

★ Encoding Ordinal Data (slide - 16 - 96, 100)

- Ordinal data lies sort of between numeric data & categoric data. i.e. it has numeric data associated with it which is ordered. But the scale may not match with the order. i.e. the differences may not be meaningful.
- An example may be the 'pain scale' that the doctor uses to ask patients for a level or measure of pain. It goes from 0 to 10. But the difference between level 3 pain & level 4 pain may not be same as difference between level 5 & level 6. Because that pain or this scale is not quantified properly. It doesn't have any unit of measurement or scientifically defined formula for measurement.
- So, you can't treat ordinal data like numeric or categoric. You wanna get a new way of encoding it.
- One way could be ~~the~~ unary / thermometer code. Like for 'n' different levels of ordinal data, you take 'n' binary bits i.e. 'n' features with binary values. And you fill it up with 1s as you go up a level (like shown in the slide)

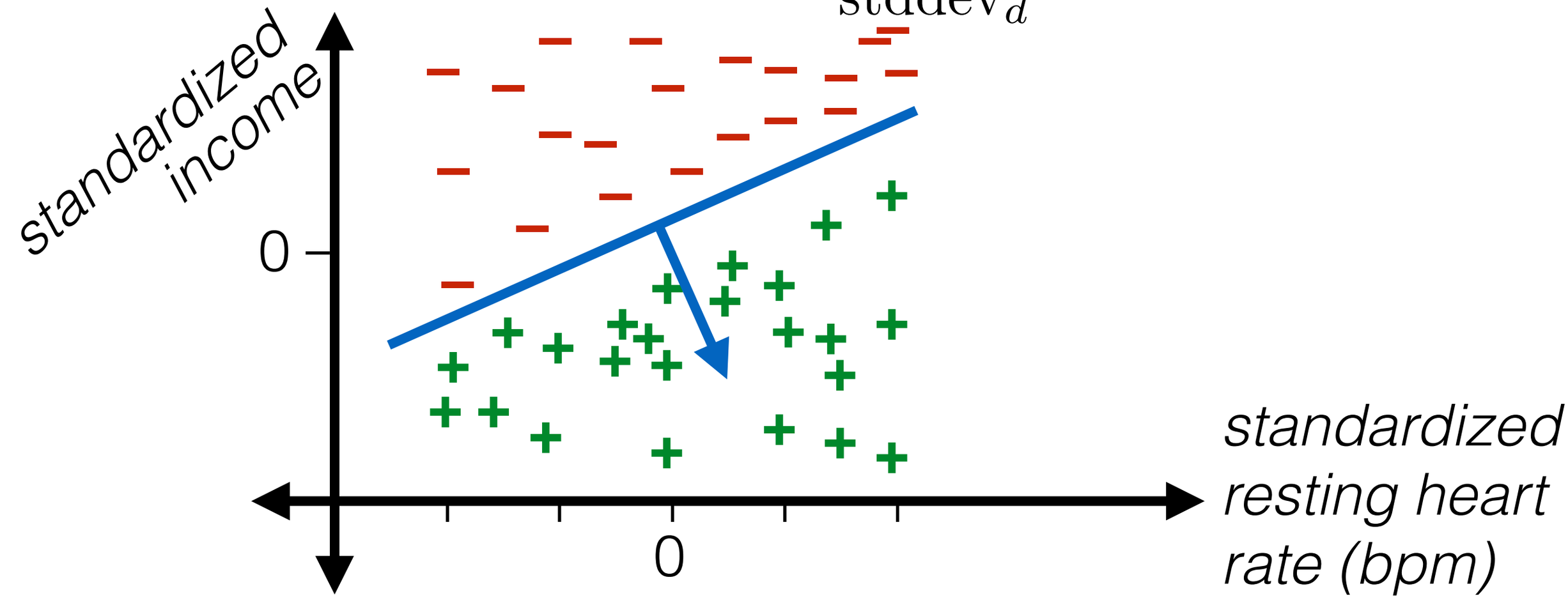
Encode numerical data

- A closer look at the output of a linear classifier
- Idea: standardize numerical data
  - For *d*th feature: $\phi_d^{(k)} = \frac{x_d^{(k)} - \text{mean}_d}{\text{stddev}_d}$



Encode numerical data

- A closer look at the output of a linear classifier
- Idea: standardize numerical data
  - For *d*th feature: $\phi_d^{(k)} = \frac{x_d^{(k)} - \text{mean}_d}{\text{stddev}_d}$



Encode Numerical Data (Standardize Numerical Data) (slide - 21, 22 - 113, 114)

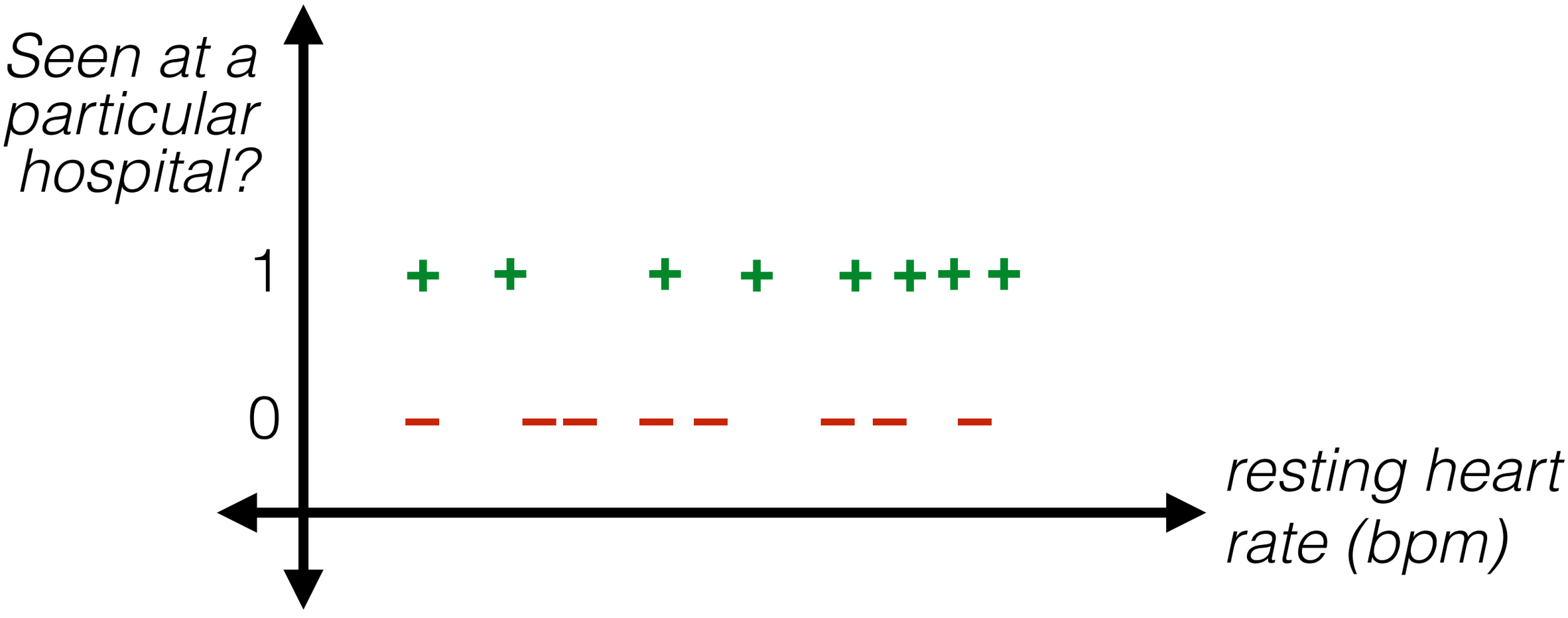
- If you have two features with different scales like one with values ranging in 10,000s & one with values ranging in 100s, then you will have

issues vizualizing data & interpreting it from vizualisation.

- Standardization distributes the data around zero & changes its scale. This gives much better results.
- In standardization, you take a data point (feature), you subtract its mean from it & you divide ~~the data~~ that by the standard deviation.

More benefits of plotting your data

- And talking to experts



★ More benefits of plotting your data (slide - 23 - 118) (& talking to experts)

- It's always beneficial to vizualize your data. It may give you an idea about how ~~to~~ the data is, how is one feature related to other, etc.
- You will be able to improve your model from good interpretations of your visualization.

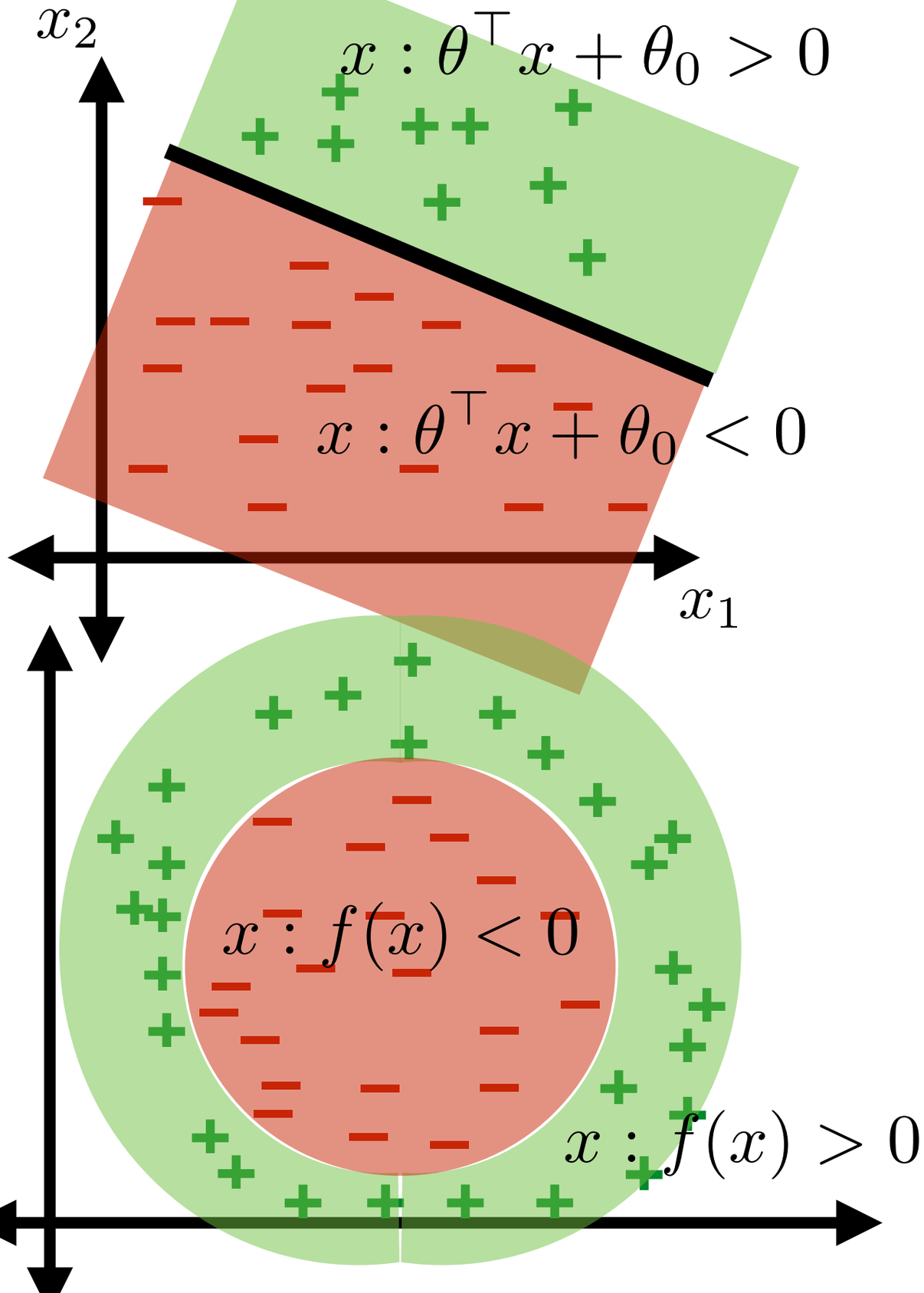
NOTE:- What are quantiles? (Question)

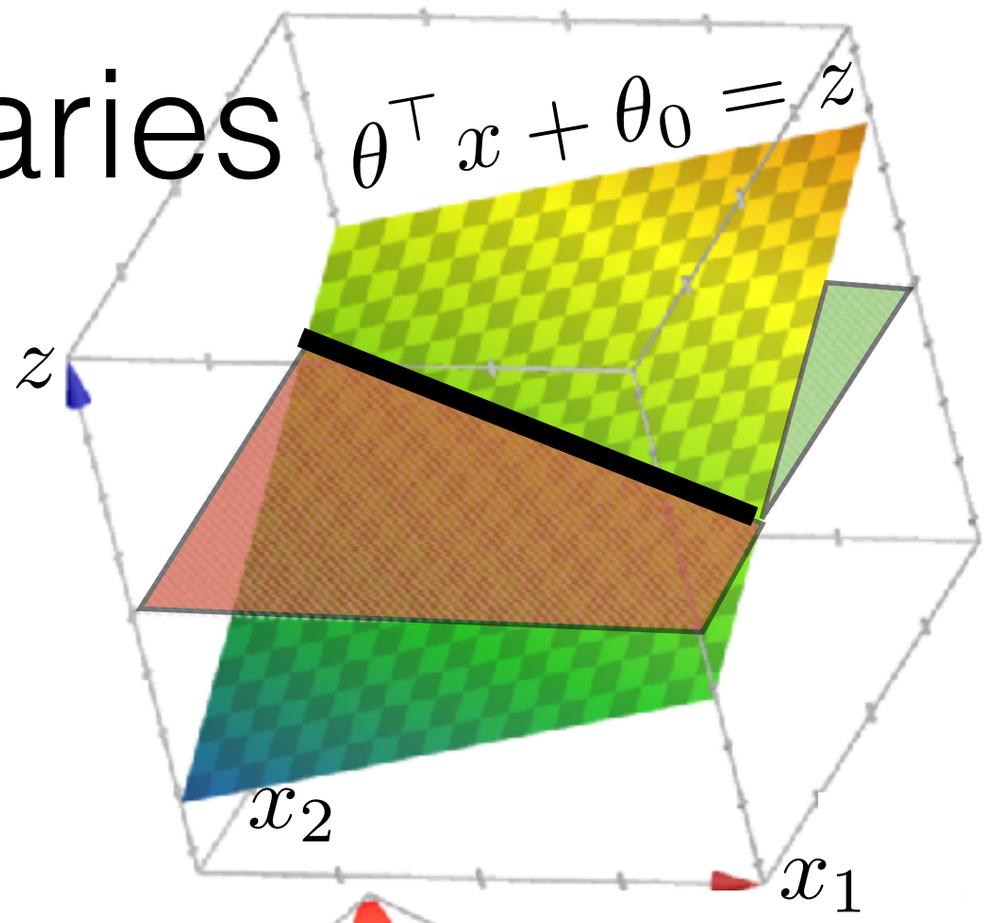
Encode data in usable form

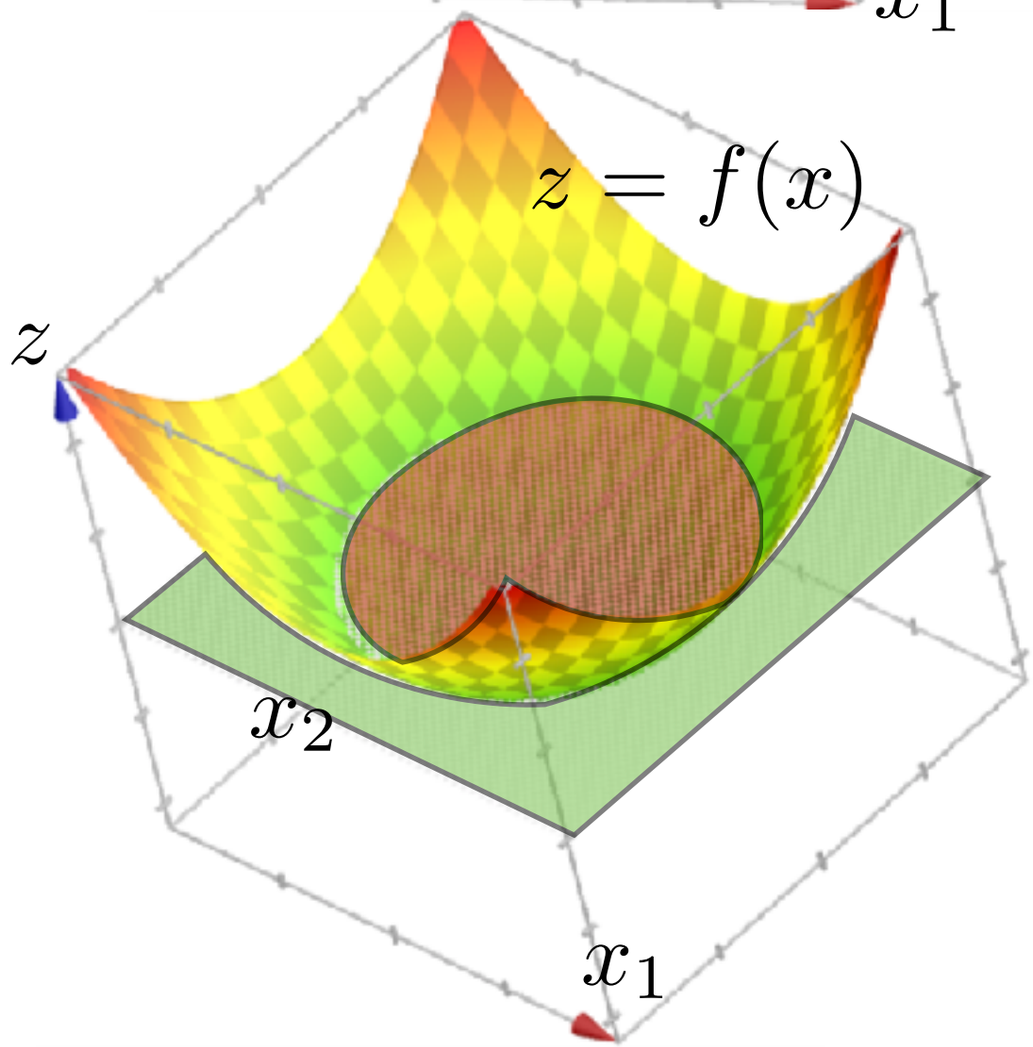
- Identify the features and encode as real numbers
- Standardize numerical features

	resting heart rate (bpm)	pain?	j1,j2,j3,j4,j5	m1, m2	decade	family income (USD)
1	-1.5	0	1,0,0,0,0	1,0	1	2.075
2	0.1	0	0,1,0,0,0	1,1	-1	-0.4
3	1.9	1	1,0,0,0,0	0,1	2	-0.25
4	-0.3	0	0,0,0,1,0	0,0	2	1.75

Classification boundaries







Classification Boundaries (Slide - 27 - 147)
(3-D functions)

- It's better to visualize these classifiers in 3-D (for 2 feature vectors), where the 3rd dimension 'z' is defined by hypothesis function.
- For example:- Like given in the slide, for linear classifiers, imagine $\theta^T x + \theta_0 = z$, where when $z=0$, we have the classifying hyperplane.
  In case of non-linear concentric boundaries, you can define $z = f(x)$. In both cases, labels are assigned for $z>0$ & $z<0$. (+1 for $z>0$ & -1 for $z<0$)

Nonlinear boundaries

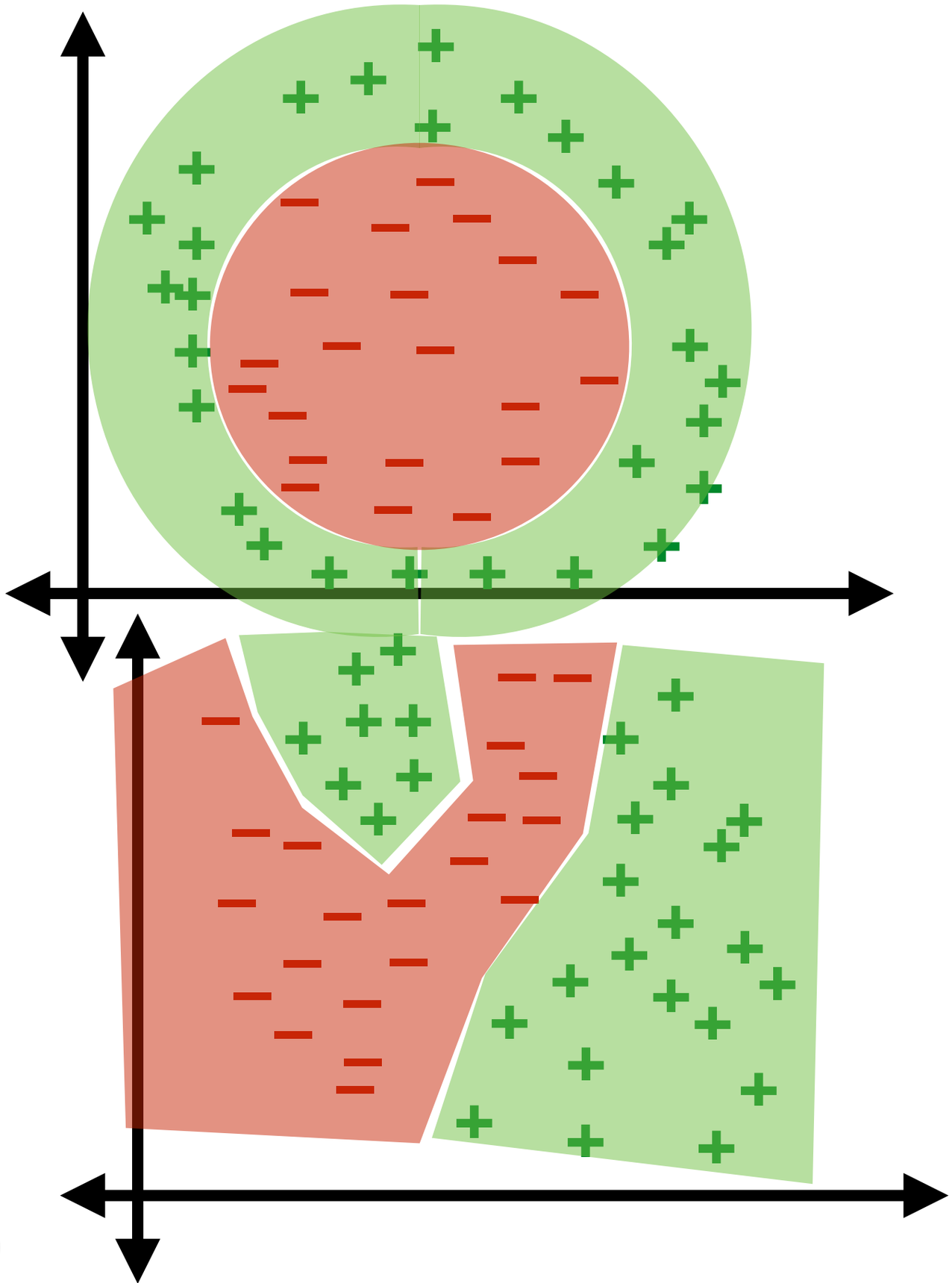
- Idea: can approximate a smooth function with a *k*th order Taylor polynomial (e.g. around 0)

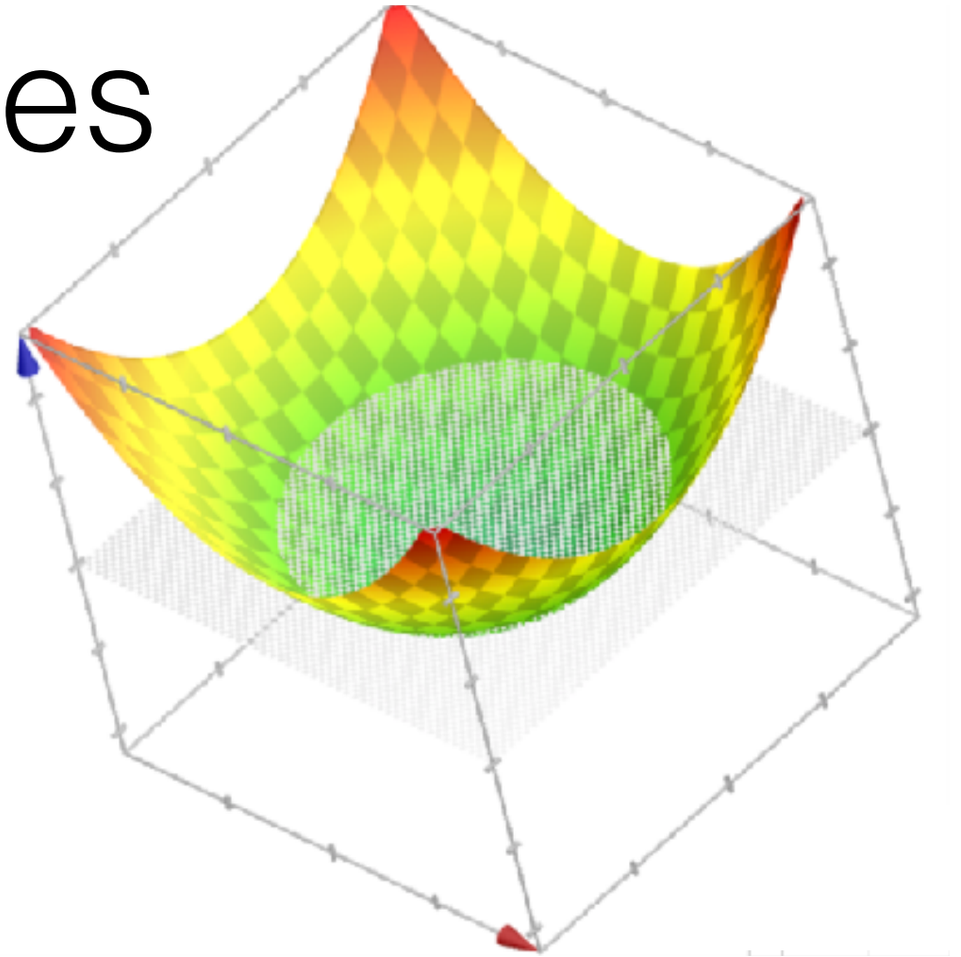
order (*k*)	terms when *d*=1	terms for general d
0	$[1]$	$[1]$
1	$[1, x_1]$	$[1, x_1, \ldots, x_d]$
2	$[1, x_1, x_1^2]$	$[1, x_1, \ldots, x_d,$ $x_1^2, x_1x_2, \ldots, x_{d-1}x_d, x_d^2]$
3	$[1, x_1, x_1^2, x_1^3]$	$[1, x_1, \ldots, x_d,$ $x_1^2, x_1x_2, \ldots, x_{d-1}x_d, x_d^2,$ $x_1^3, x_1^2x_2, x_1x_2x_3, \ldots, x_d^3]$

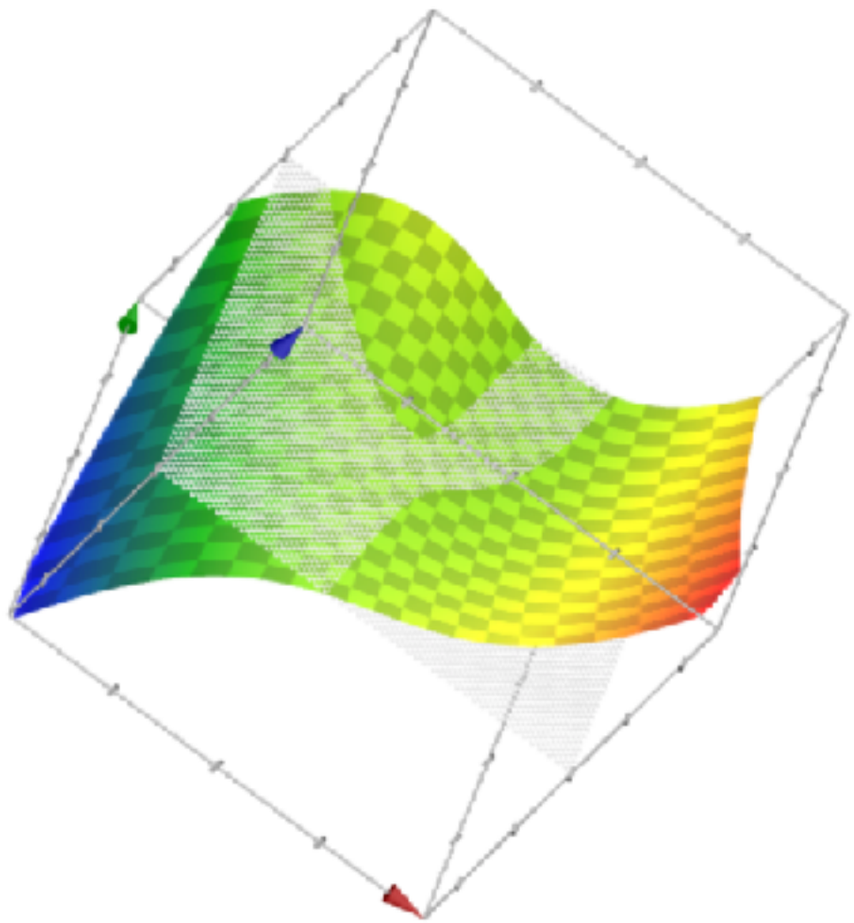
* Nonlinear Boundaries :- (slide - 28 -)
(Taylor expansion example)

- You really have to decide yourself on how to choose 'z'. It could be a linear classifier (two planes intersecting each other) with linear boundaries, you can set it to polynomial functions like $z = f(x)$ where $f(x)$ is a polynomial in x_1, x_2.
- To determine what & how to choose, you must really vizualize the data.
- In this case (slide) You expand the feature space by using taylor expansion on both x_1 & x_2

Nonlinear boundaries



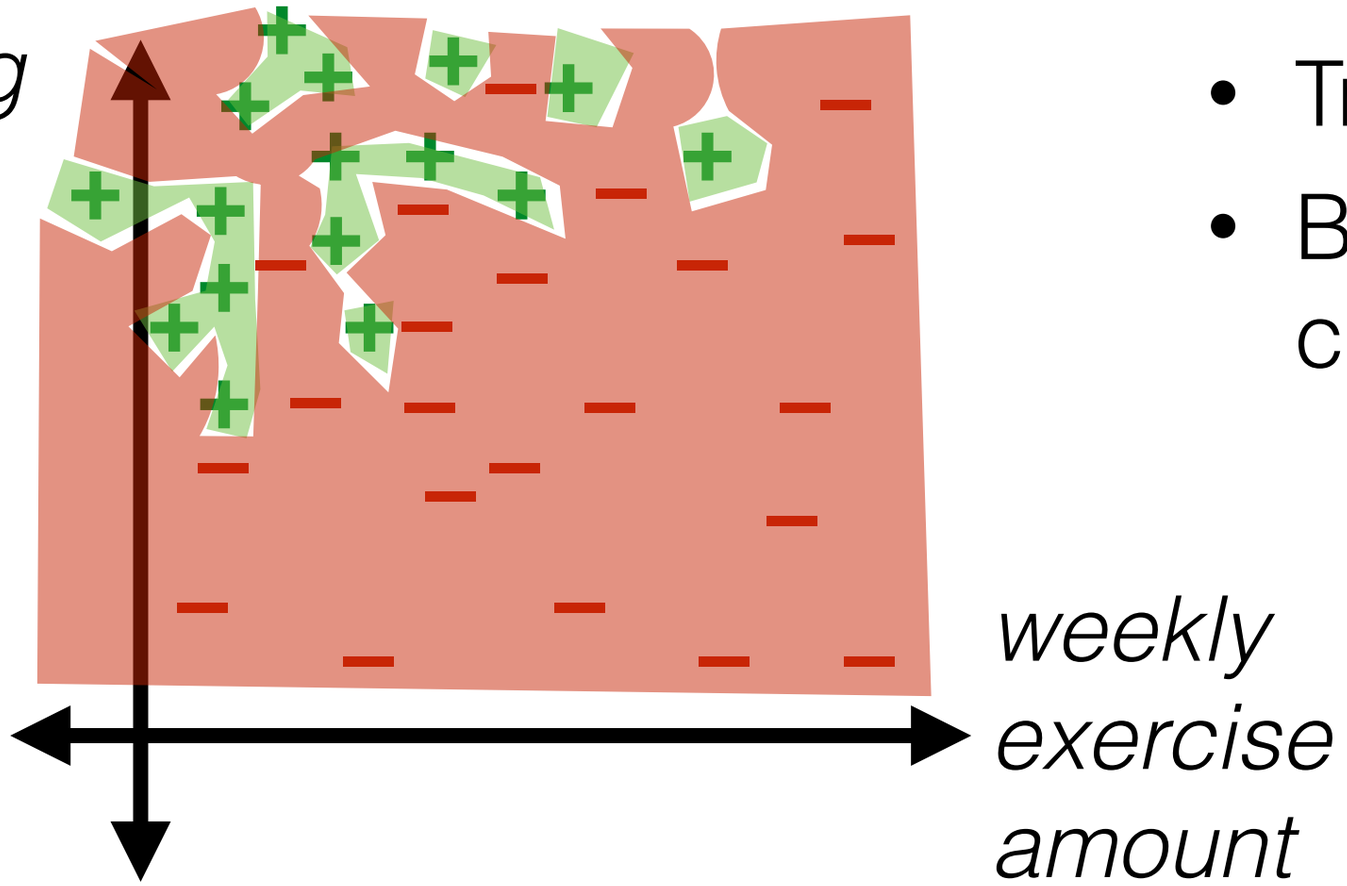




★ Non linear Boundaries (slide - 29 - 167)

- You can see in the slide, that for a dataset that doesn't obviously seem classifyable, we can classify it just by creating a smooth function in 3-D (for 2 feature-vectors)
- Thus, you can linearly classify data even for non linear boundaries.
- But, this also means that with enough computational & mathematic power you can linearly classify extremely cluttered & complex non-linear boundary data.

Nonlinear boundaries



- Training error is 0!
- But seems like our classifier is overfitting

- How can we detect overfitting?
- How can we avoid overfitting?

* Nonlinear Boundaries (Slide - 30 - 174)

(Heart rate vs. Exercise Example)

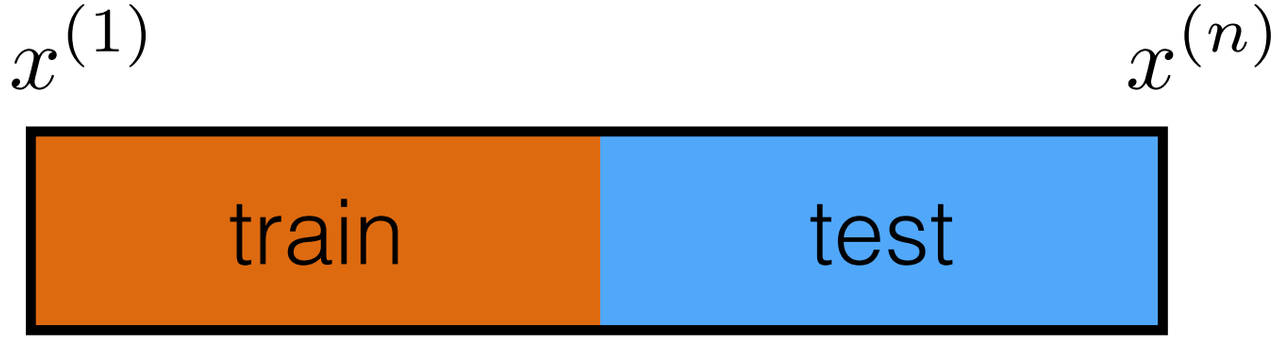
- (Continued from previous slide) So, although you get a classifier for overlapping cluttered data with training error

as zero, it may be overfitting. You may have perfected the classifier for the training set, but it may fail on new sets for prediction.

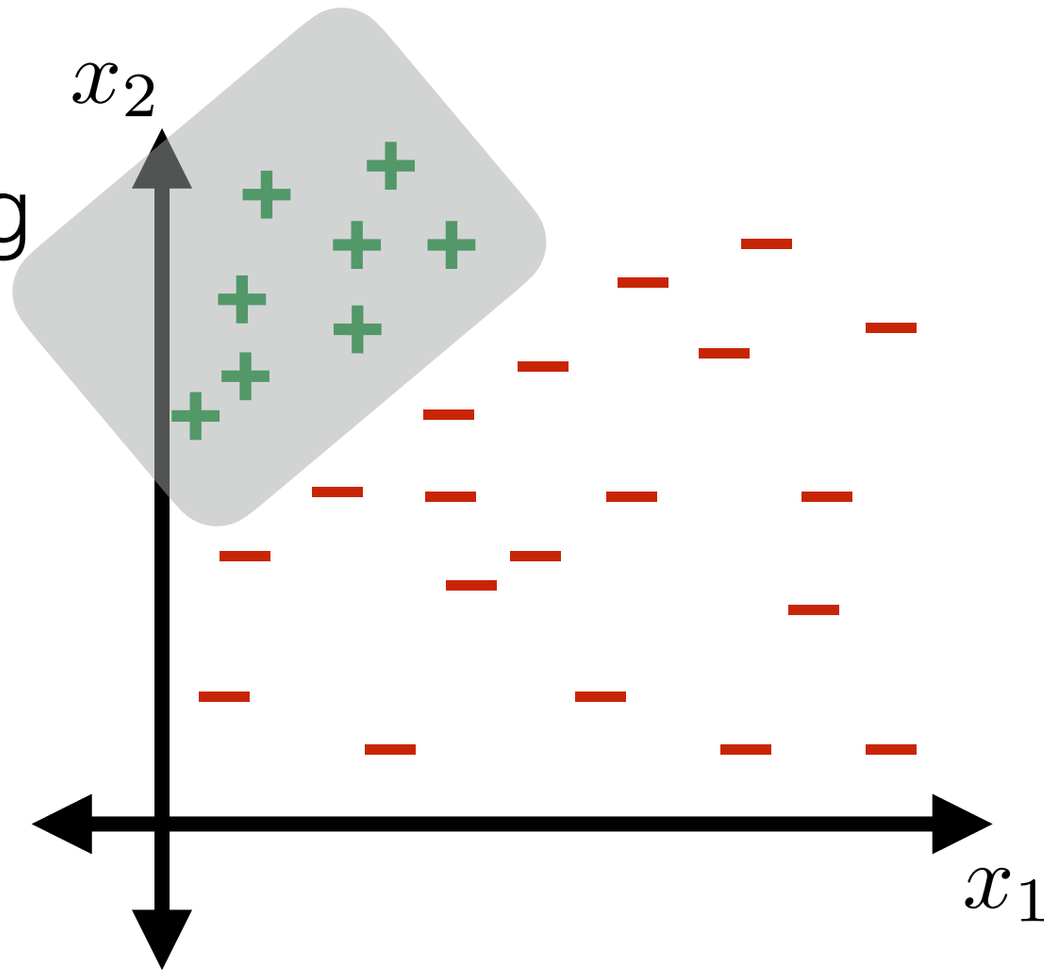
- For example, you may so overfit data for resting heart rate vs. exercise time that even 0.01 change in any parameter changes the label. Like you may get heart disease for 78 bpm & no disease for 79 bpm & again disease for 80 bpm. That seems illogical. It means you have just overfitted your data.

Evaluation of a learning algorithm

- How good is our learning algorithm on data like ours?



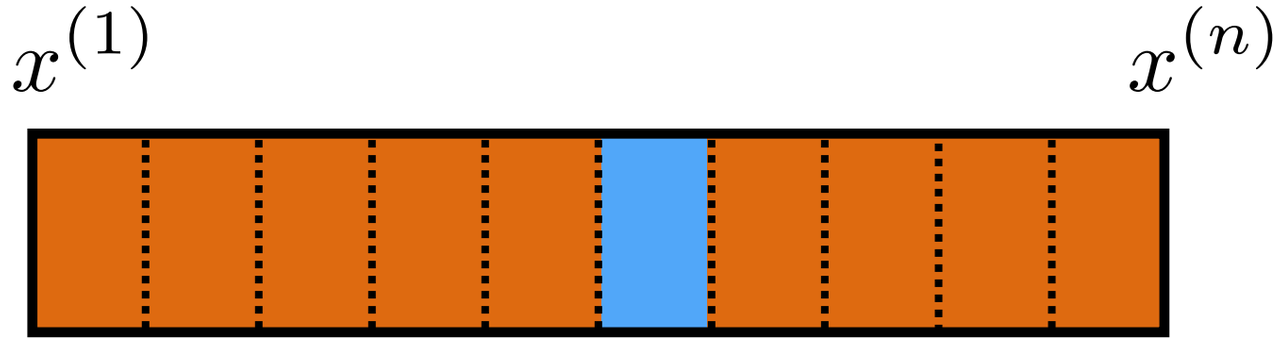
- Idea: use full data for training and then report training error
- Idea: reserve some data for testing
  - More training data: closer to training on full data
  - More testing data: less noisy estimate of performance
  - Only one classifier might not be representative
  - Good idea to shuffle order of data



★ Evaluation of Learning Algorithm (Slide - 31 -)

- One good solution is to train your classifier only on one portion of your data. So that you can test it on the remaining data.
- How much portion to choose depends on you. More training portion means high chances of overfitting & low amount of data to test on. Less training portion means your classifier will obviously be weak. It's a compromise.
- Now, since you are dividing your data into portions, make sure you shuffle it first. You don't want your portions to be biased. (For example: You are collecting data from hospitals & your training portion ~~comprises~~ has majority data from only one hospital. There will be bias)

Evaluation of a learning algorithm



Cross-validate($\mathcal{D}_n$, k)

  Divide $\mathcal{D}_n$ into k chunks $\mathcal{D}_{n,1}, \ldots, \mathcal{D}_{n,k}$ (of roughly equal size)

  for i = 1 to k

    train h_i on $\mathcal{D}_n \backslash \mathcal{D}_{n,i}$ (i.e. except chunk i)

    compute "test" error $\mathcal{E}(h_i, \mathcal{D}_{n,i})$ of h_i on $\mathcal{D}_{n,i}$

Return $\frac{1}{k} \sum_{i=1}^{k} \mathcal{E}(h_i, \mathcal{D}_{n,i})$

- Again, good idea to shuffle order of data first