Machine Learning

$$y_i = f(x_i)$$

$$y_i = \beta x_i$$

$$y_i = \beta x_i$$

"explanation"

Explanation

- Goal is to understand the complex process by which one variable “affects” another
- Technical in estimation, moral in implication
- Generalizable across all contexts within scope conditions (otherwise you don’t really understand the cause)
- Output is a story that must “makes sense” to someone

$$y_i = \beta x_i$$

"prediction"

Prediction

- Goal is to forecast what will happen
- Technical in estimation, practical in implication
- Only relevant to context of prediction
- Output it a state of affairs

Prediction and Explanation

Prediction

- Goal is to forecast what will happen
- Technical in estimation, practical in implication
- Only relevant to context of prediction
- Output it a state of affairs
- **Machine learning**

Explanation

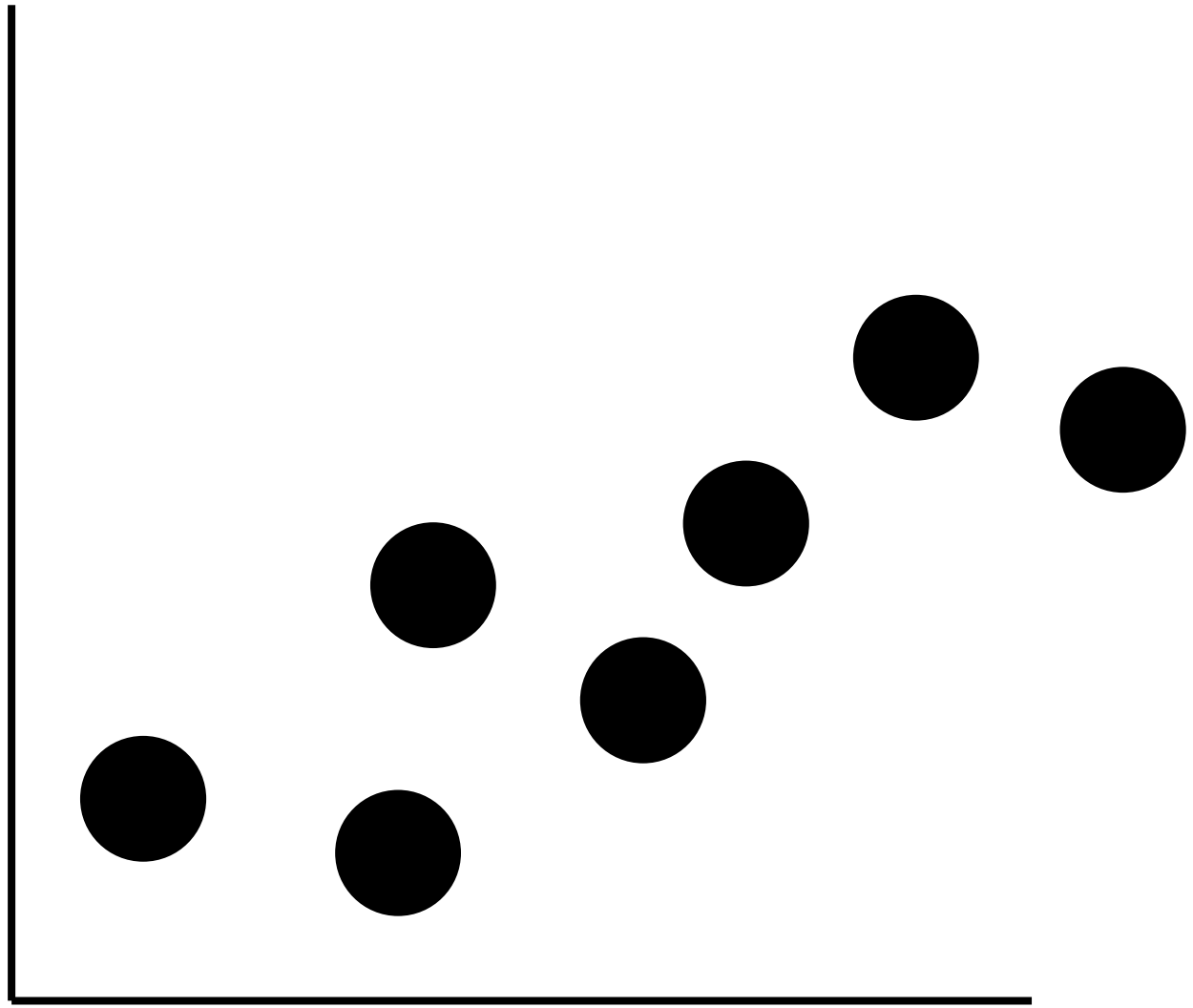
- Goal is to understand the complex process by which one variable "affects" another
- Technical in estimation, moral in implication
- Generalizable across all contexts within scope conditions (otherwise you don't really understand the cause)
- Output is a story that must "makes sense" to someone
- **Causal inference, process tracing**

Prediction **vs.** Explanation?

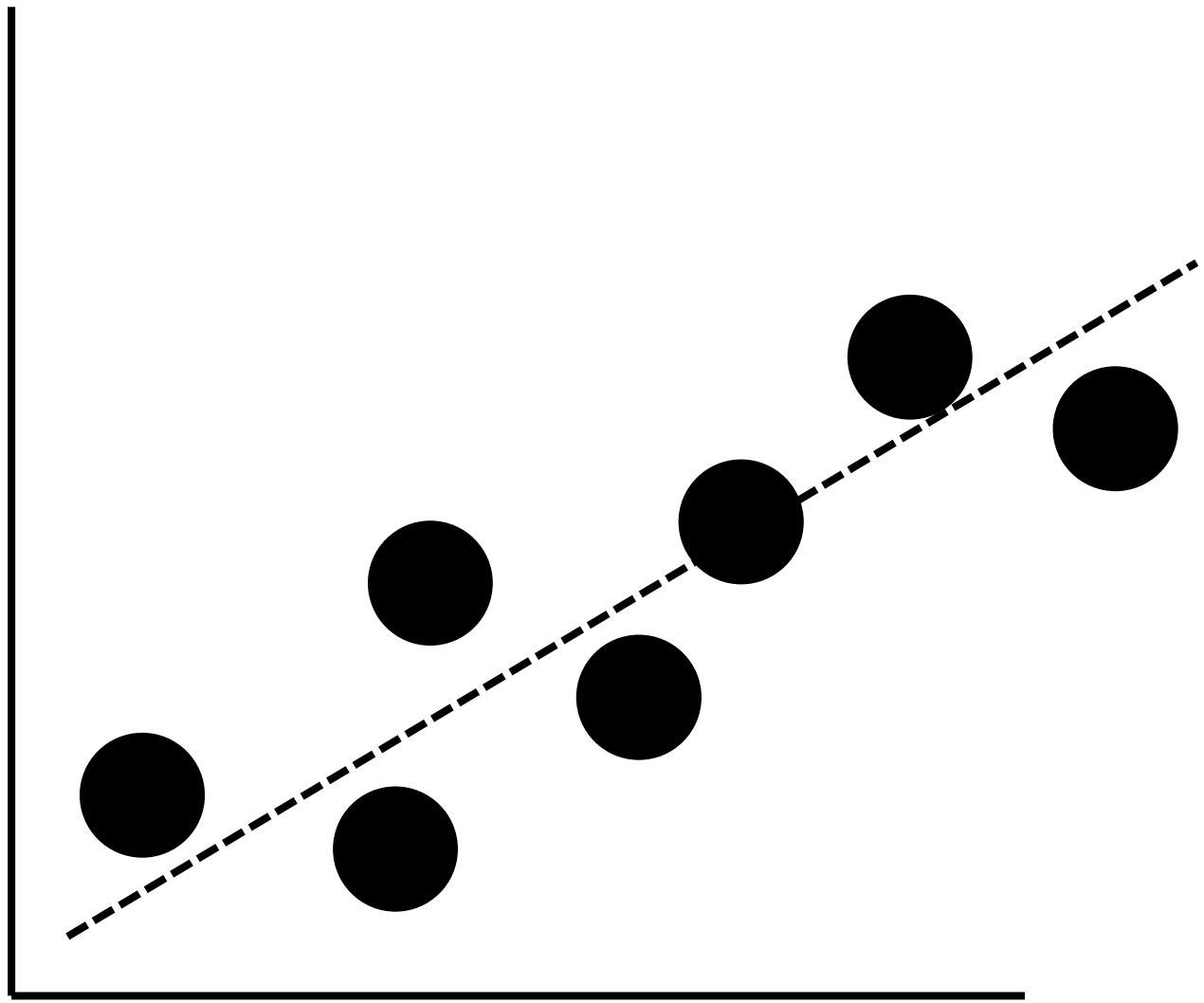
- If you can perfectly explain, you can perfectly predict
- If you can perfectly predict, doesn't necessarily mean you can perfectly explain ("black box" prediction)
- However, if you can't predict at all than your explanation is probably wrong
- Some philosophers (see Carl Hempel) have said that a good theory is one that predicts (though I disagree with this to some degree)

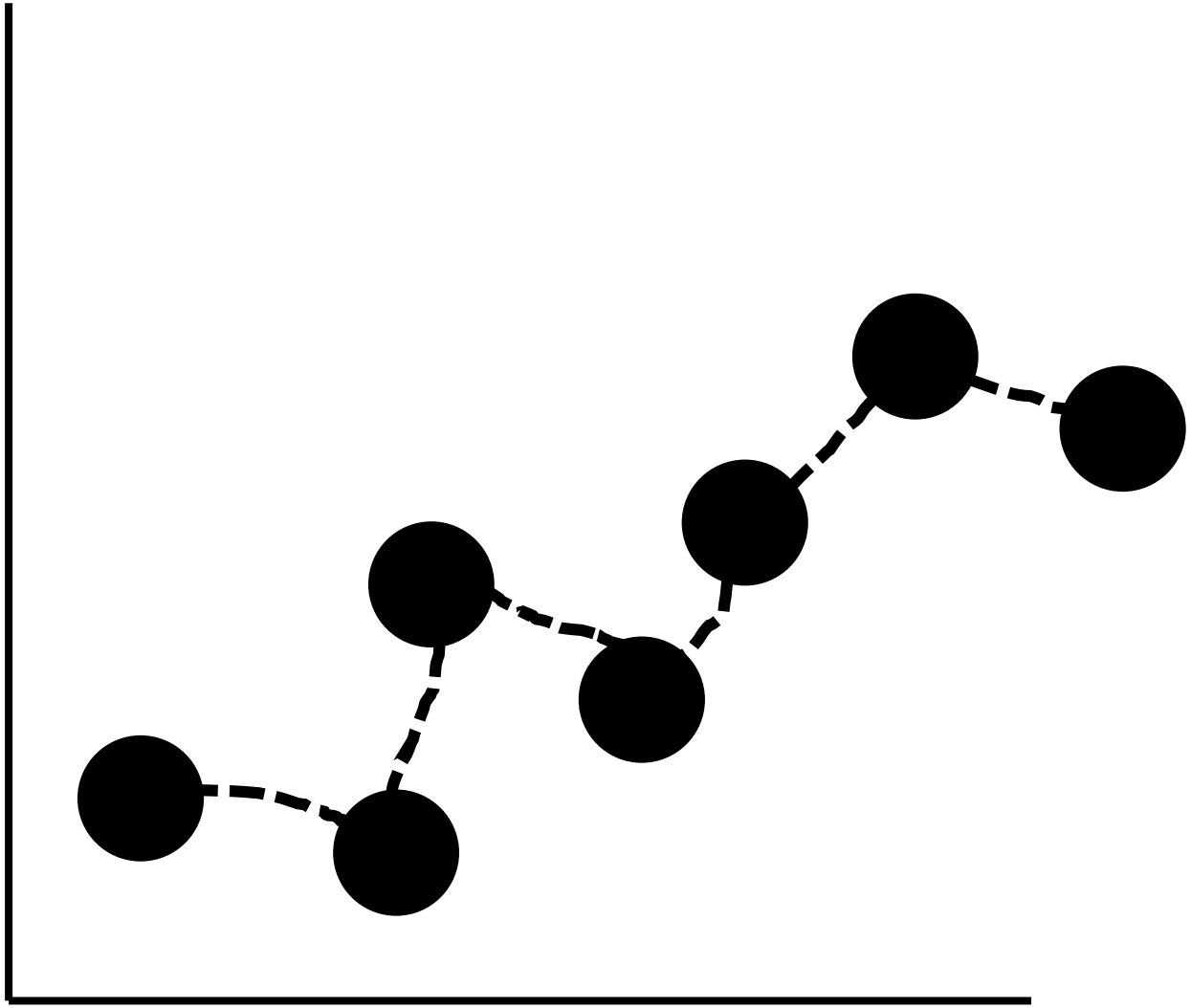
A difference in games

- For prediction, **LITERALLY ANYTHING** that increases your performance is “fair-game”
- The one caveat is that it has to be information that came about before the outcome was decided
- For models intended for explanation, you should only include variables that are either of theoretical interest or may confound your variables of interest
- Given that you don’t have perfect explanation, **different tools work better for different goals** (in the words of Susan Athey, “you don’t get anything for free”)



Predict y from x





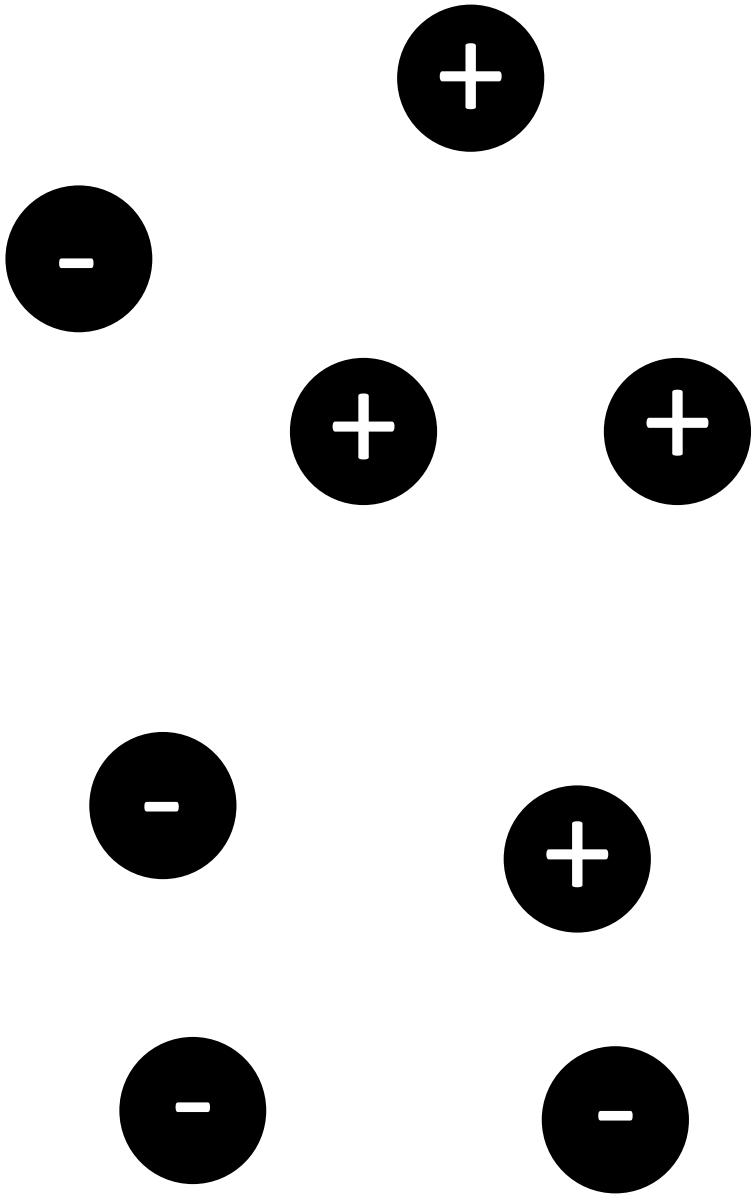
Higher
R-squared!

“Over-fitting”

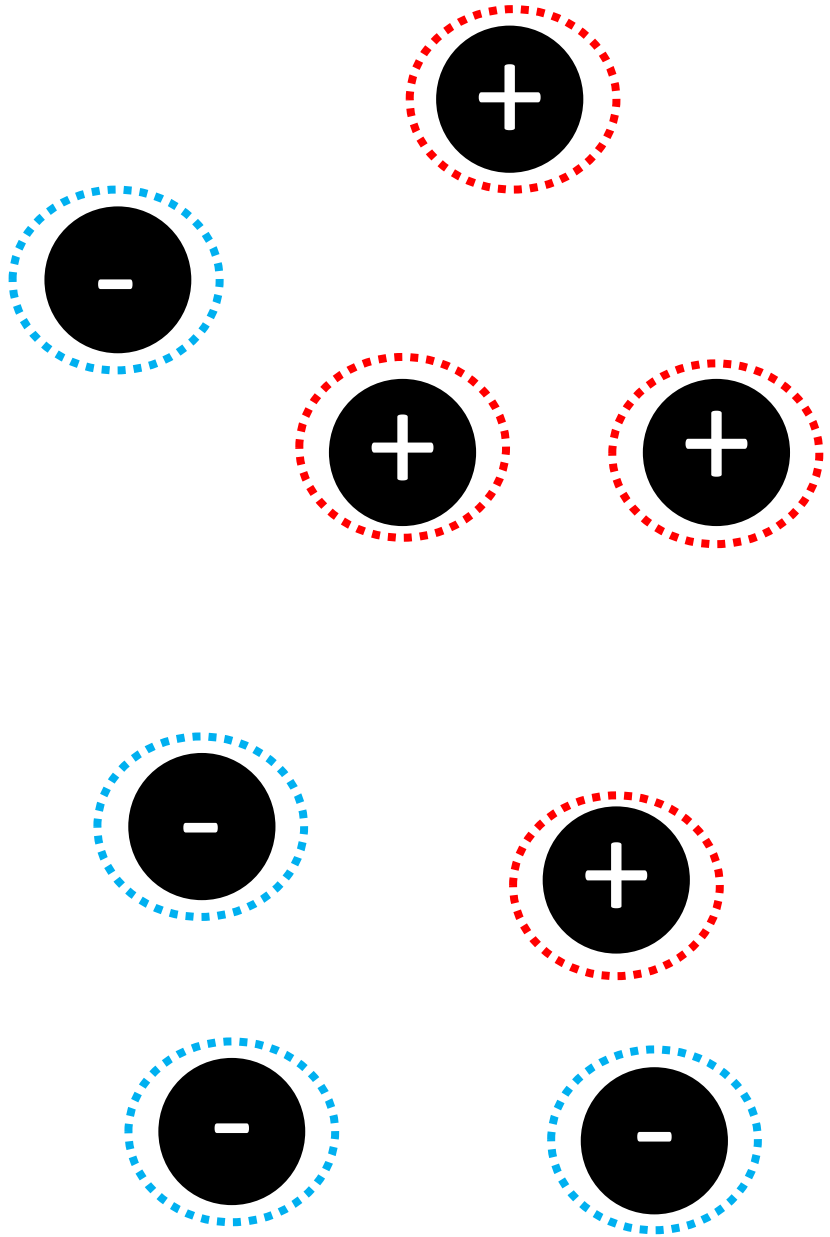
- More complex models (same model with added terms) always fit the data better
- Standard regression framework fits the “signal”, but also fits the “noise”

“Over-fitting” (it gets worse…)

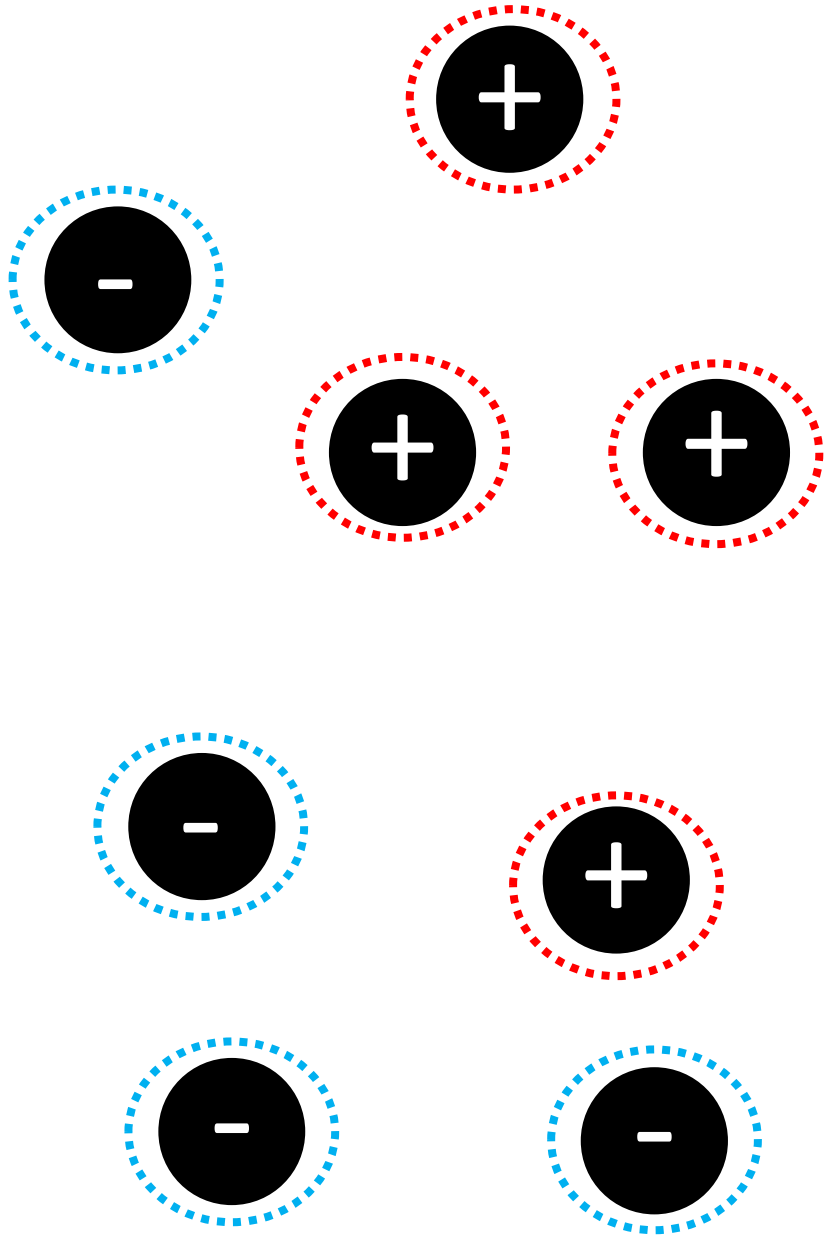
- We can “explain” our data arbitrarily well with a binary variable that is equal to 1 for each observation and 0 for all others
- So how do we properly evaluate prediction models?



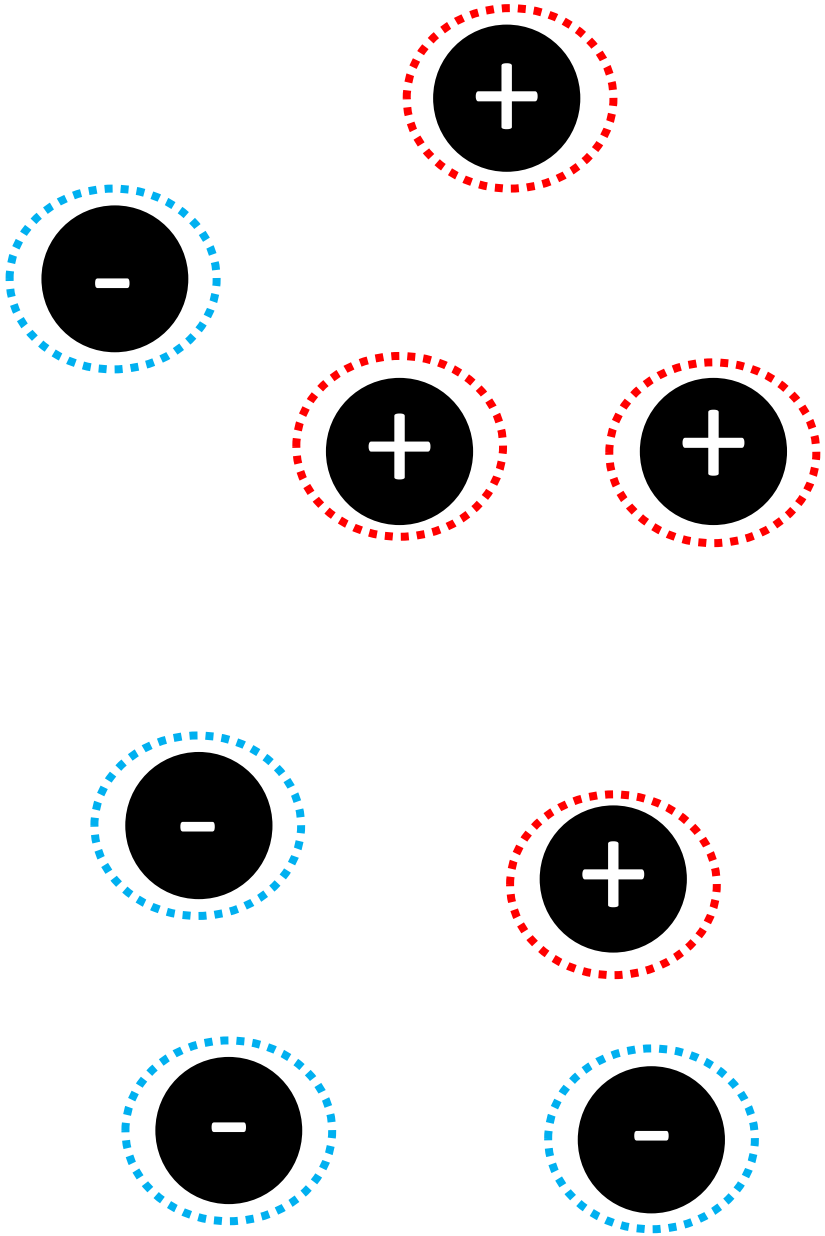
Predict whether
The circle will have a
“+” or a “-”

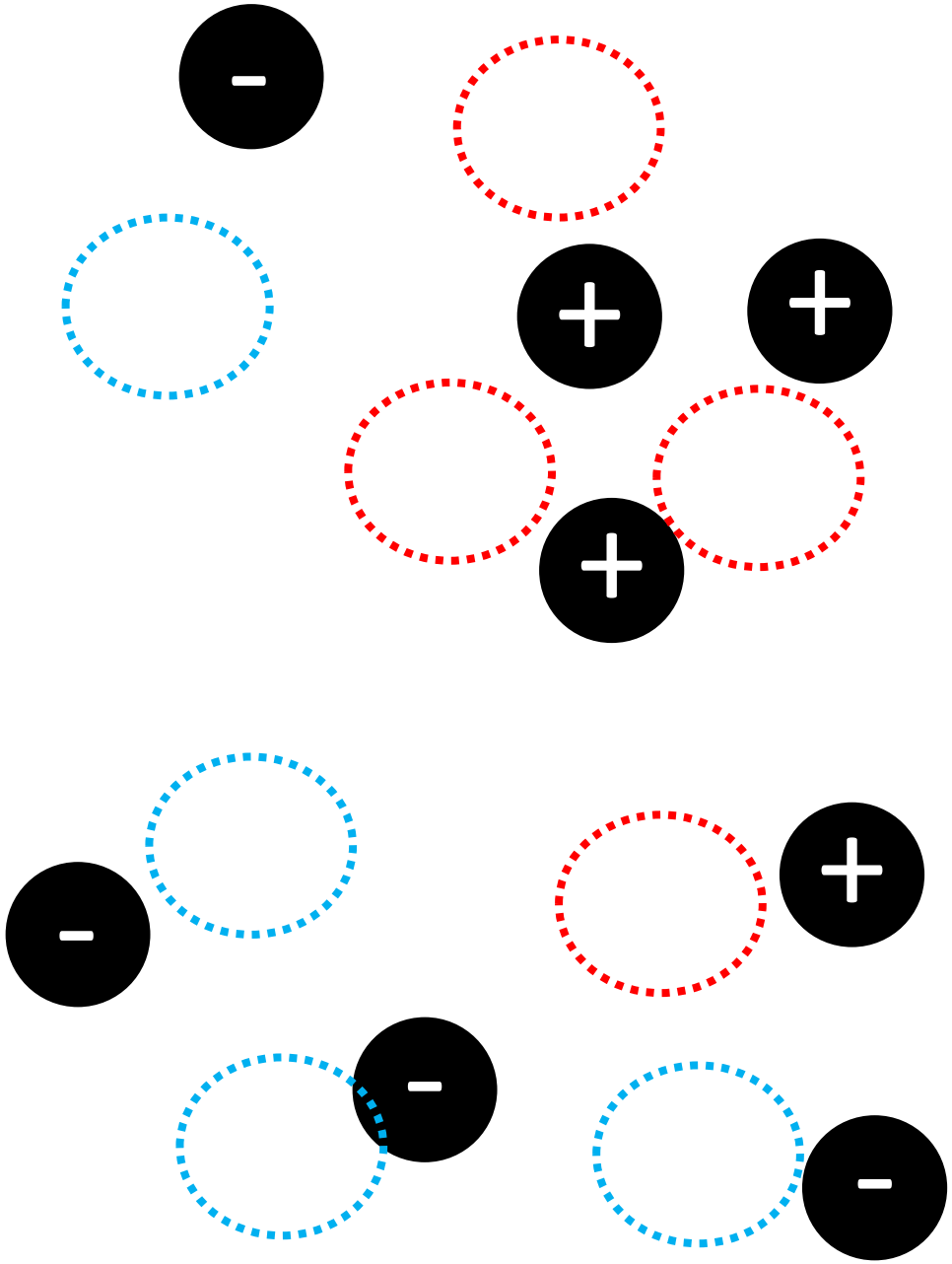


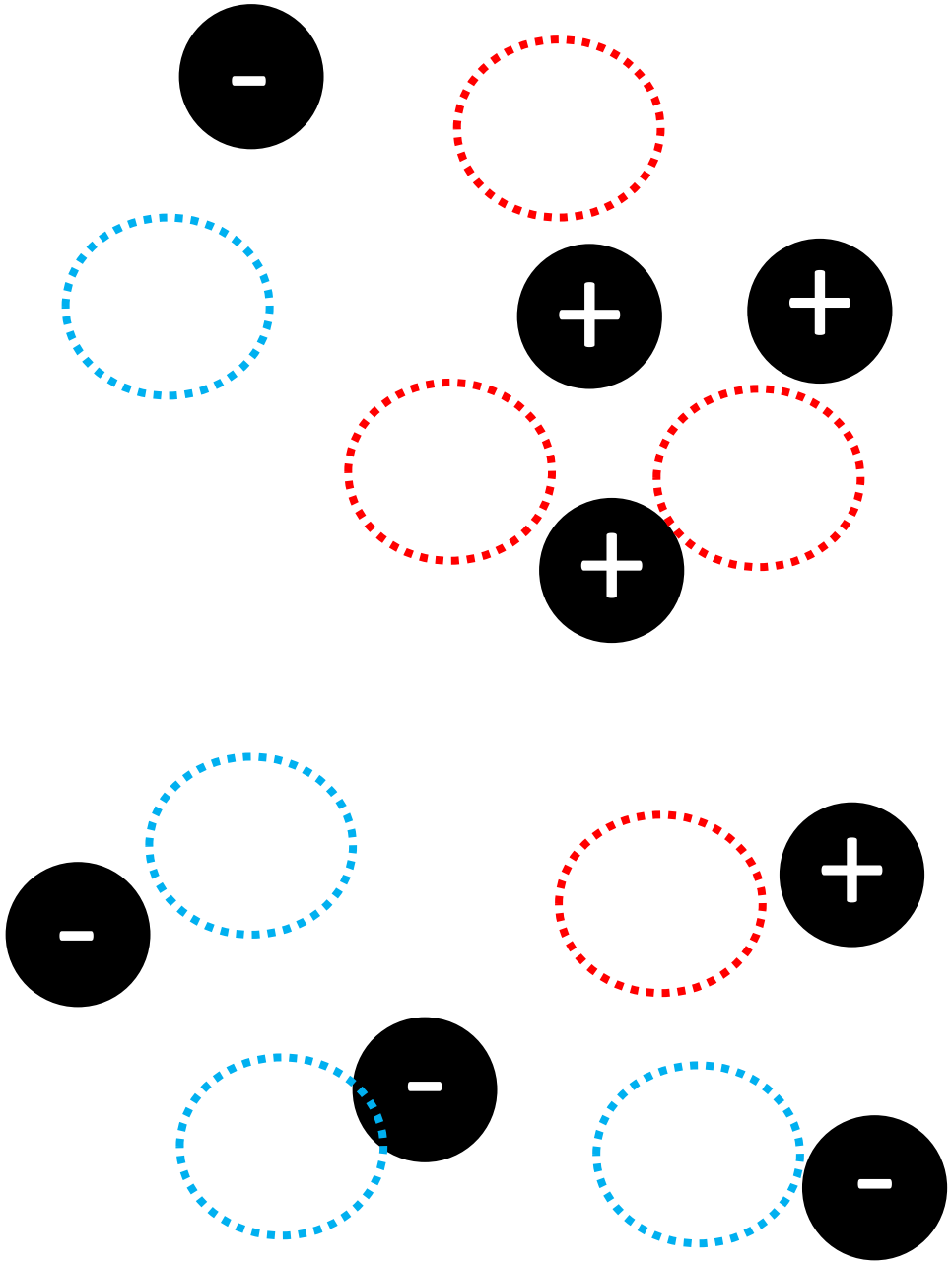
If the observation appears in a red circle, predict “+”; if in blue circle, predict “-”



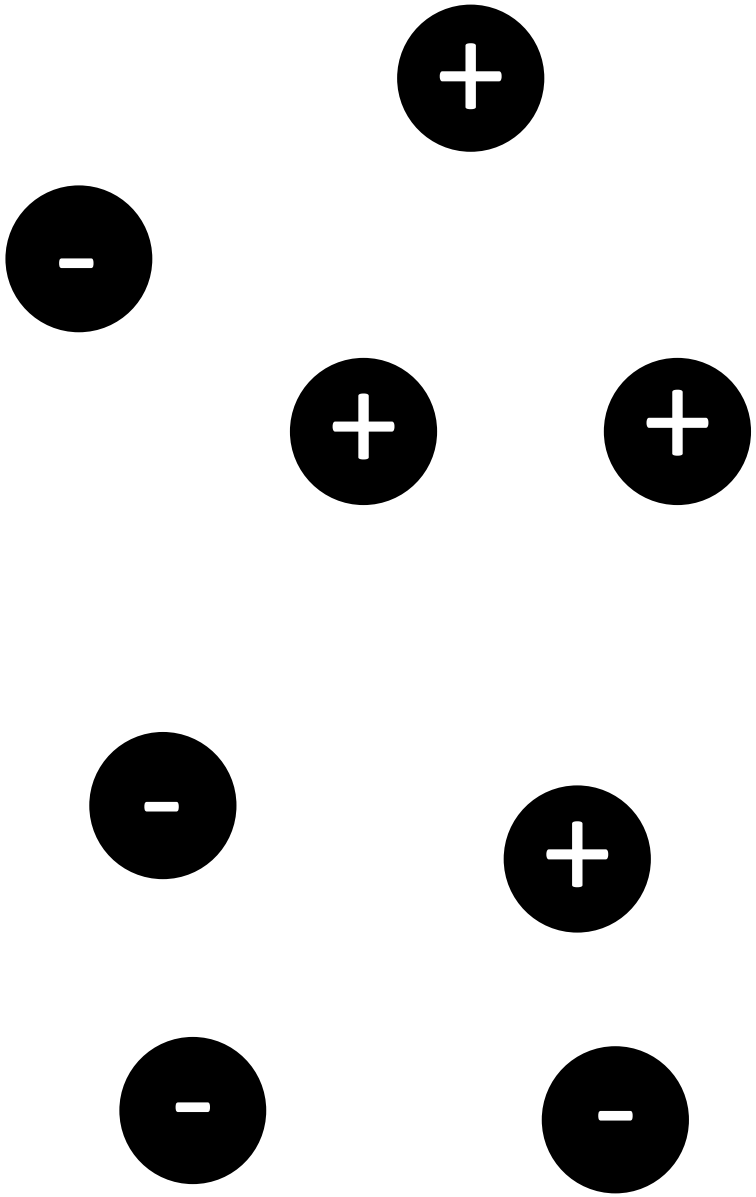
Why is this not ideal?



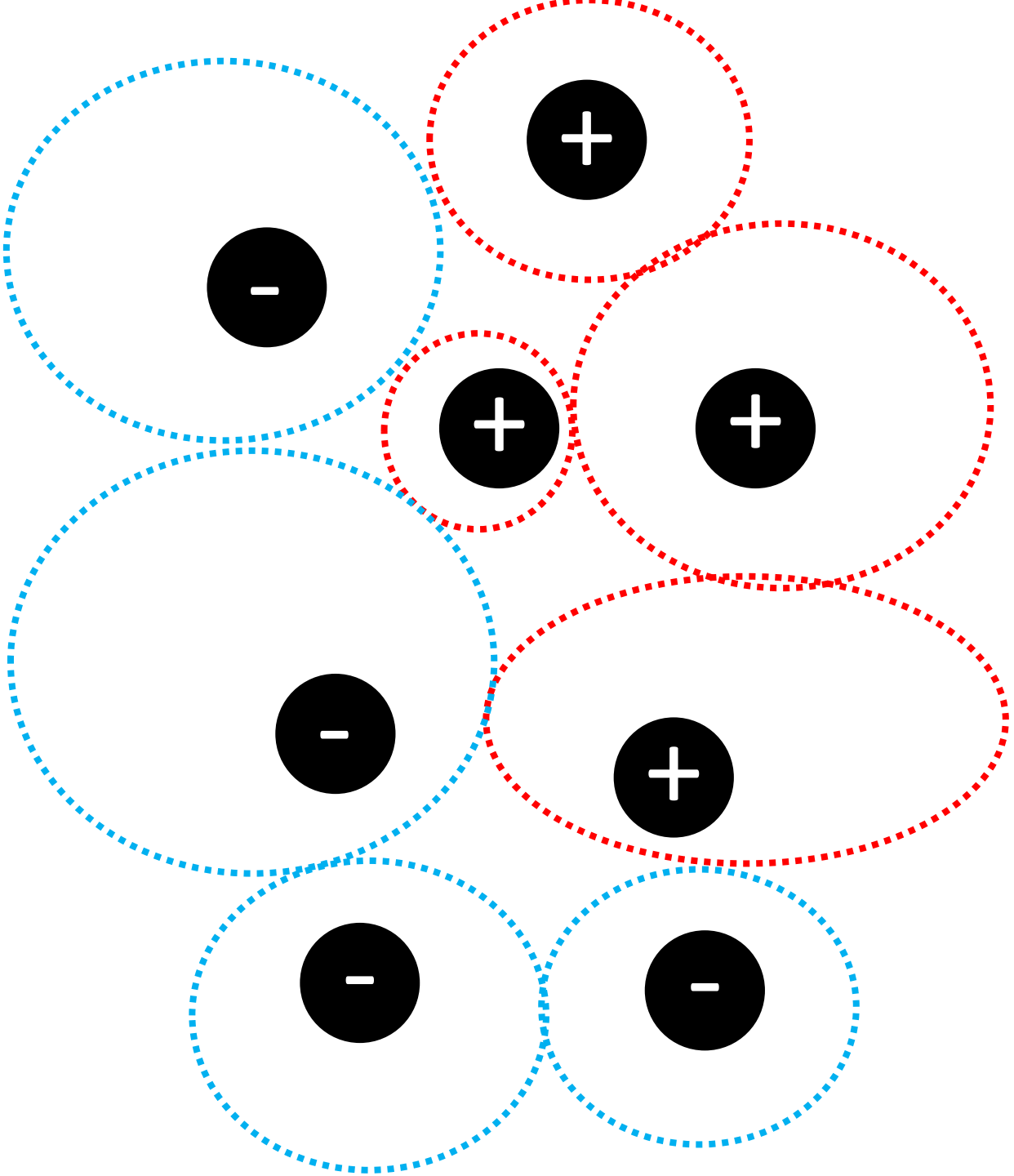


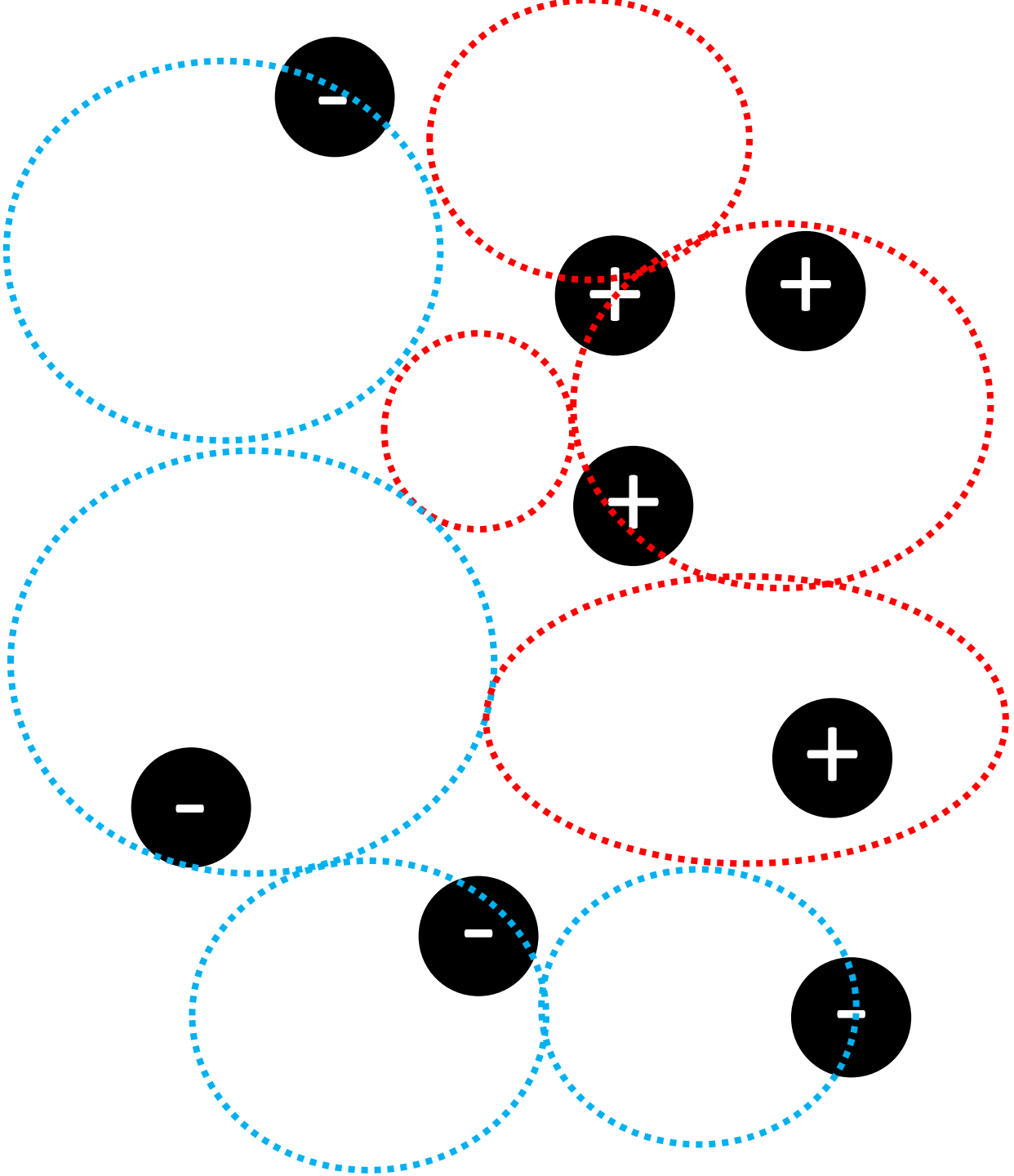


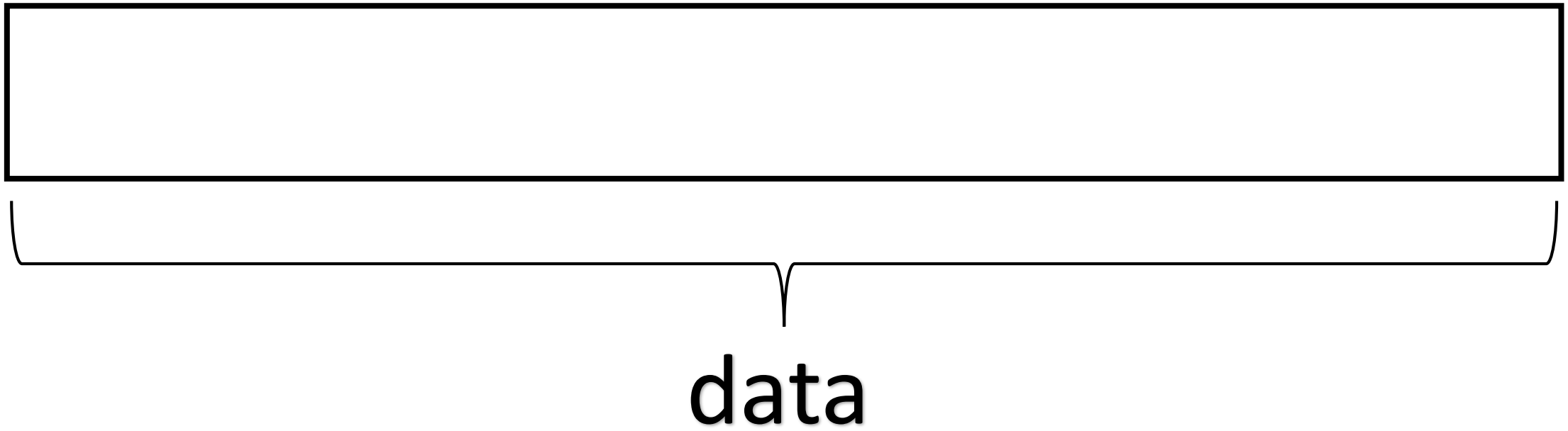
We were “overfitting” to the data, so can’t predict on new data!

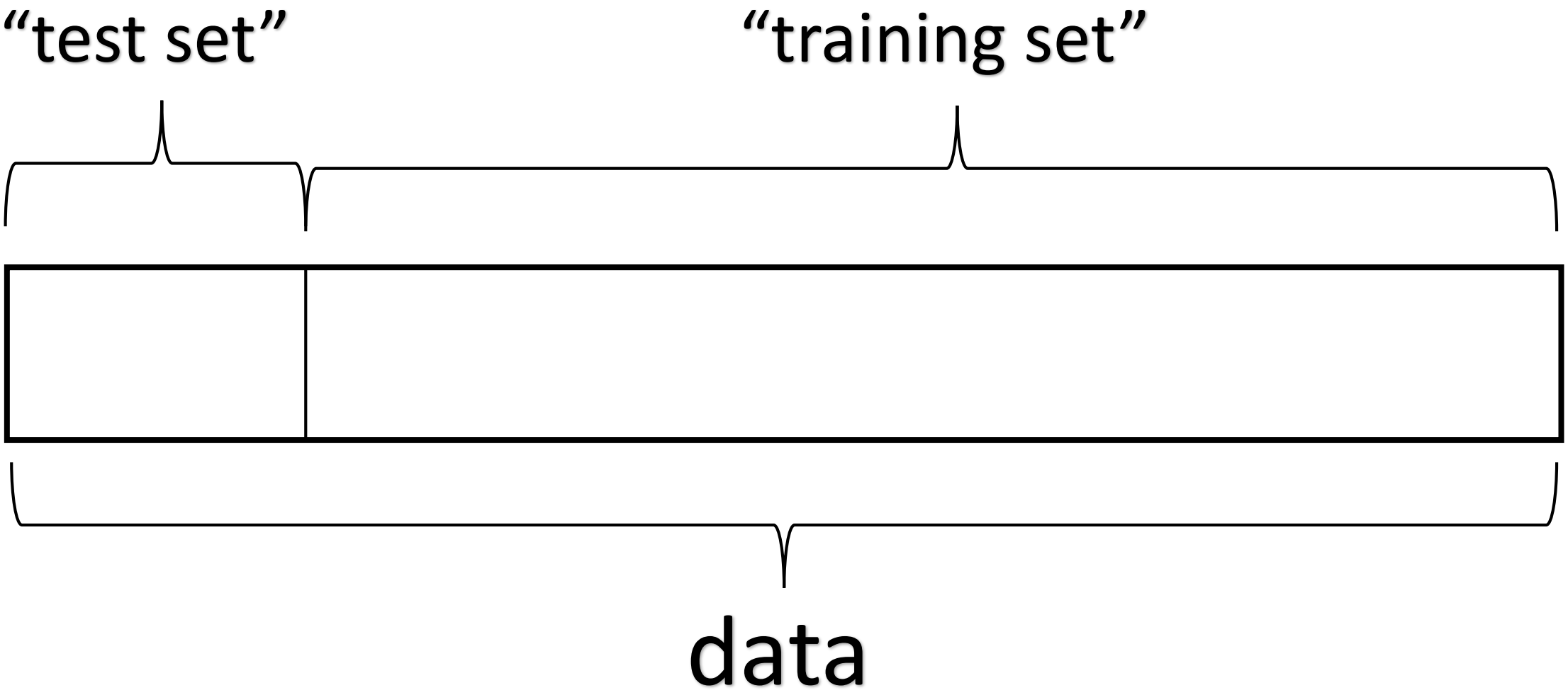


Let’s try “fitting less” to the data…






data


“test set”
“training set”
data

Regularized regression

OLS

$$\lambda = \sum_{i=1}^{n} (\hat{y}_i - y_i)^2$$

OLS

$$\lambda = \sum_{i=1}^{n} (\hat{y}_i - y_i)^2$$

"cost" that we'll minimize

OLS

$$\lambda = \sum_{i=1}^{n} \left(\sum_{j=1}^{k} \beta_j x_{ij} - y_i \right)^2$$

LASSO

$$\lambda = \sum_{i=1}^{n} \left(\sum_{j=1}^{k} \beta_j x_{ij} - y_i \right)^2 + \alpha \sum_{j=1}^{k} |\beta_j|$$

LASSO

$$\lambda = \sum_{i=1}^{n} \left(\sum_{j=1}^{k} \beta_j x_{ij} - y_i \right)^2 + \alpha \sum_{j=1}^{k} |\beta_j|$$

Standard OLS

Regularization

LASSO

$$\lambda = \sum_{i=1}^{n} \left(\sum_{j=1}^{k} \beta_j x_{ij} - y_i \right)^2 + \alpha \sum_{j=1}^{k} |\beta_j|$$

Regularization parameter

Ridge

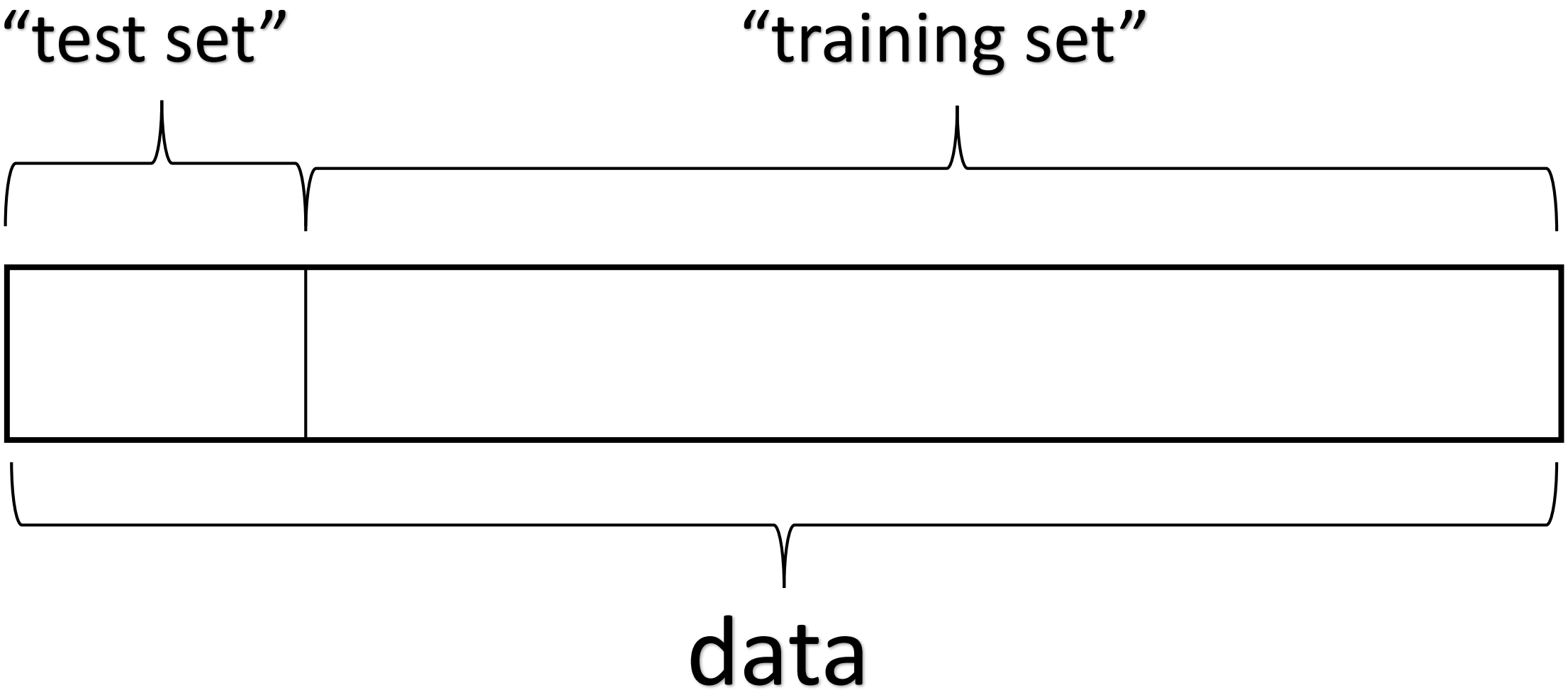
$$\lambda = \sum_{i=1}^{n} \left(\sum_{j=1}^{k} \beta_j x_{ij} - y_i \right)^2 + \alpha \sum_{j=1}^{k} (\beta_j)^2$$

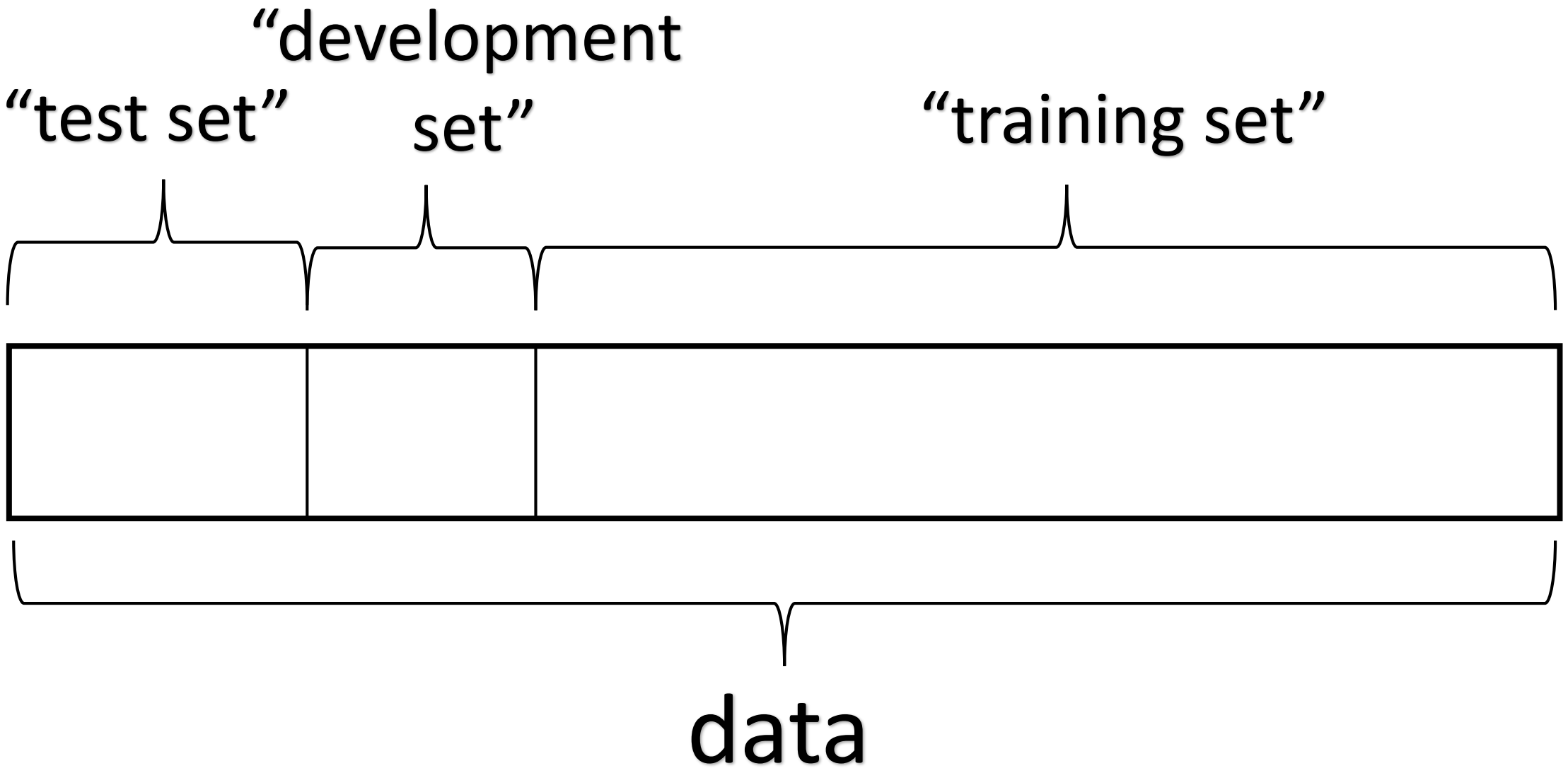
Regularized regression

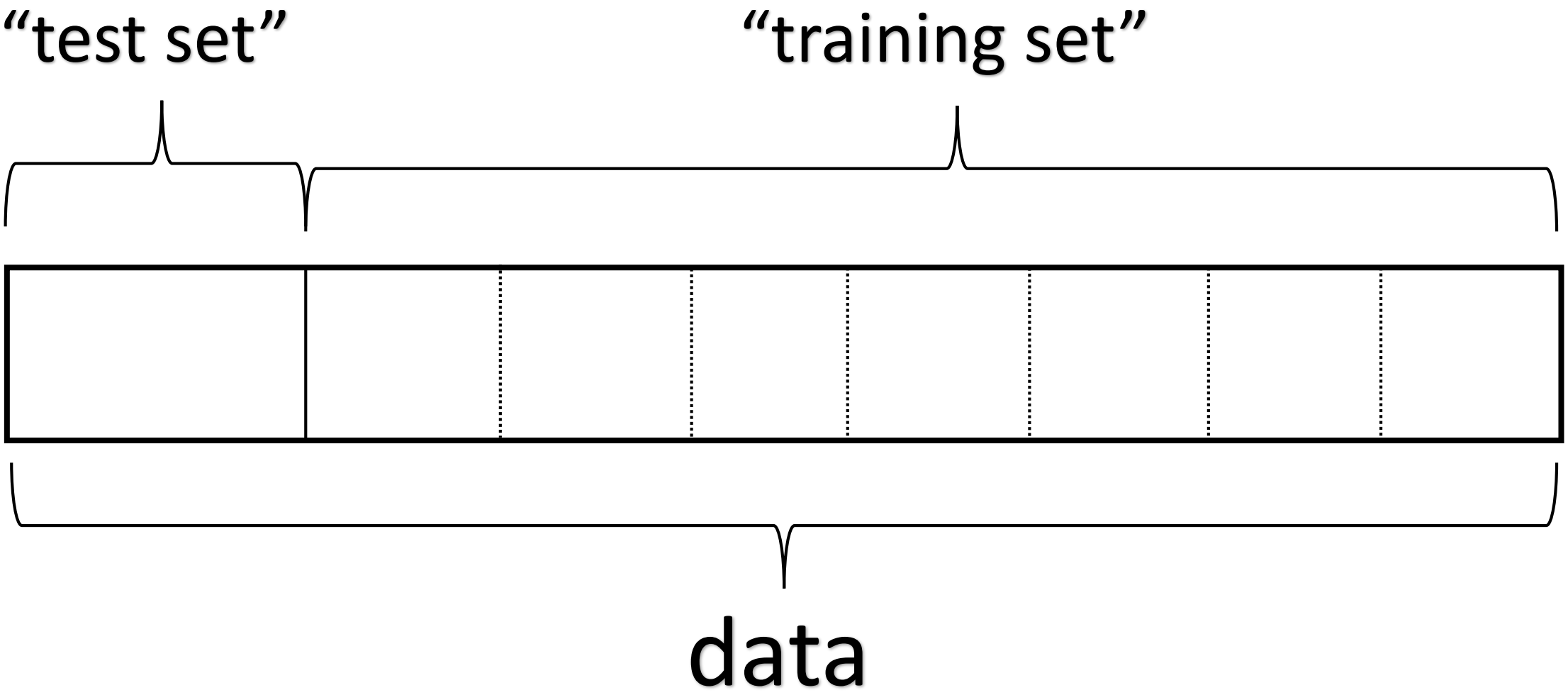
- By penalizing the model for the size of the coefficients, we…
  - Force the model to only emphasize important coefficients
  - De-select variables that have strong correlates
  - Allow the model to predict relatively well even when we throw k >> n variables at it
  - Bias the coefficient estimates associated with any particular variable
- We’re basically purposefully introducing omitted variable bias
- We better our performance on prediction by introducing bias to OLS!

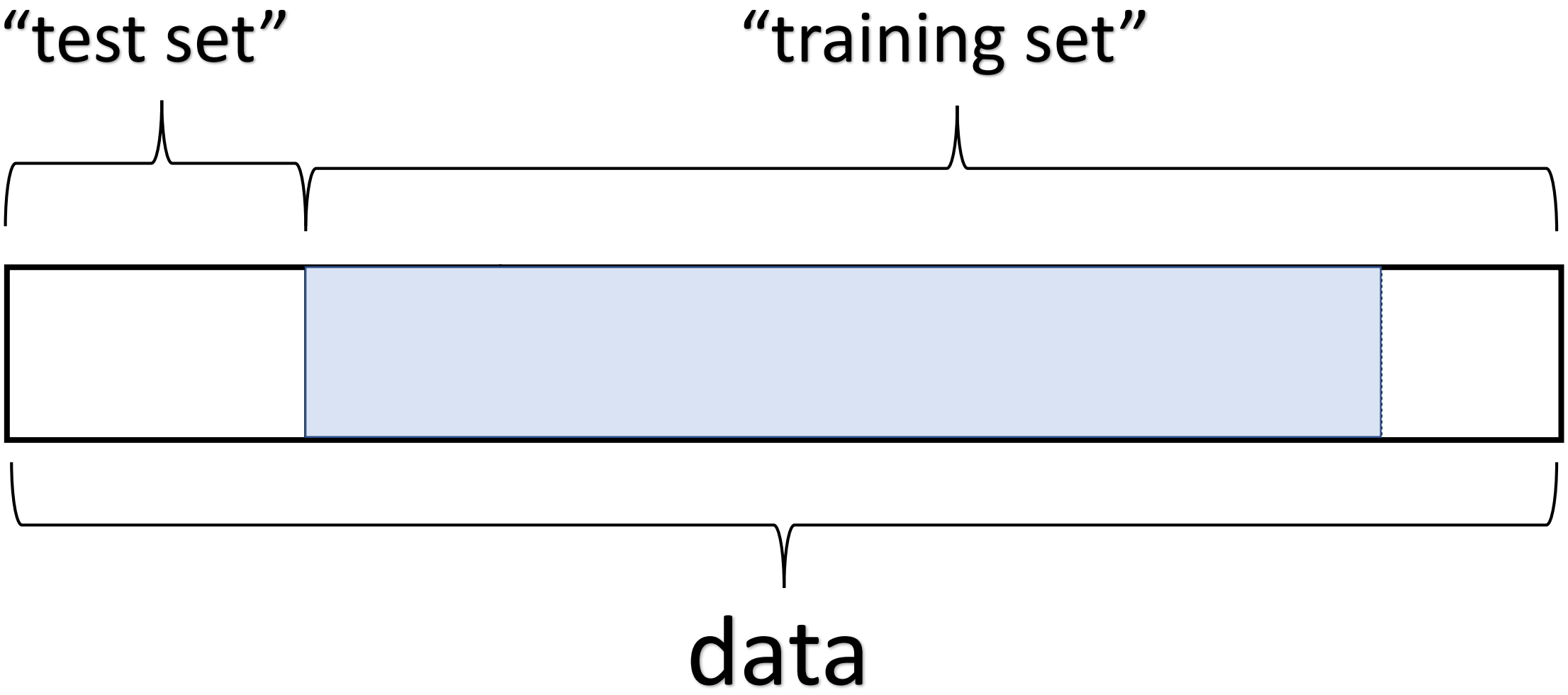
Regularized regression (cont.)

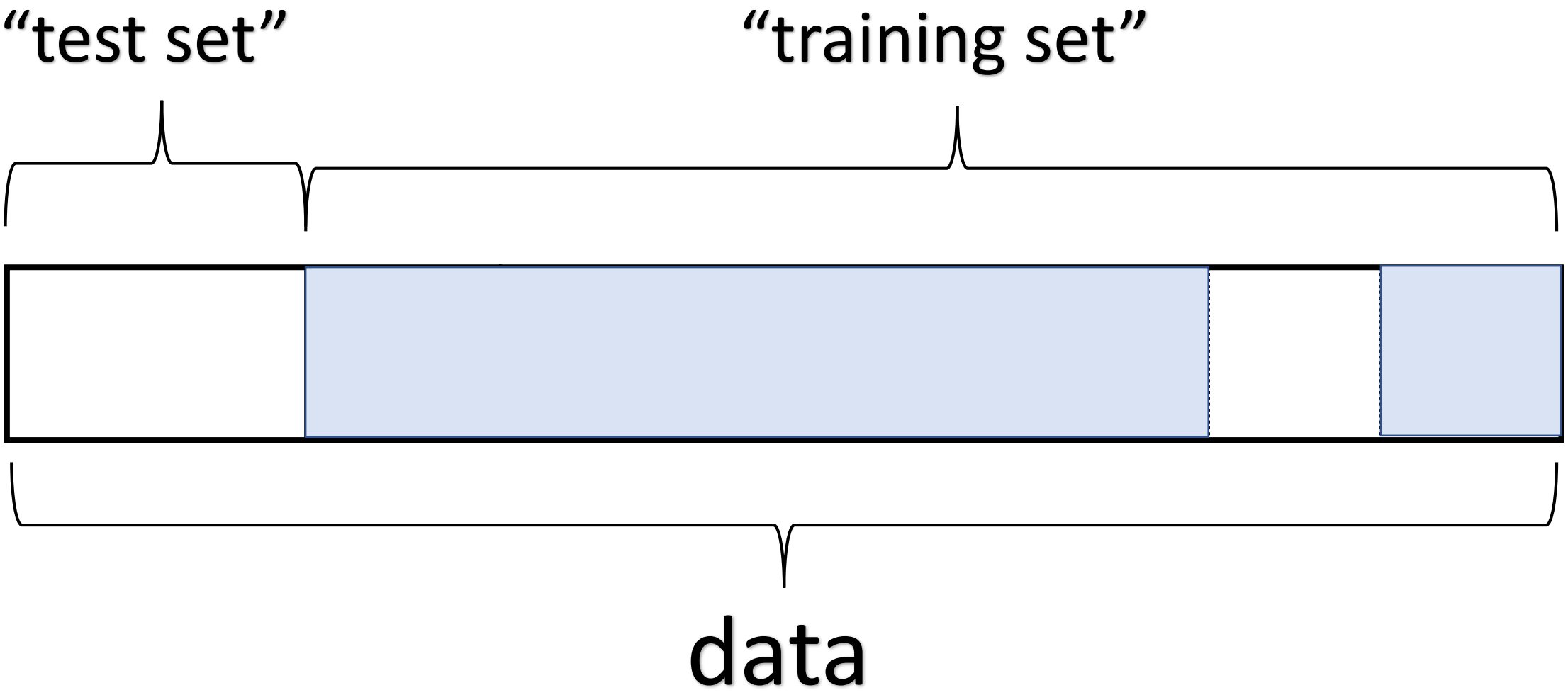
- The slight different in cost function leads to very different outcomes…
  - LASSO will select important variables and then send all others to zero
  - Ridge will select important variables and send all others near zero
- So some people use LASSO for model selection (even though they shouldn't!)
- Which one "should" you use? Whichever performs better!
- But we can only test on the test data once…?
- The lower α, the more you "fit" the data (the closer we get to regression); as $\alpha \rightarrow \infty$, model becomes the constant only model
- How do you select α?

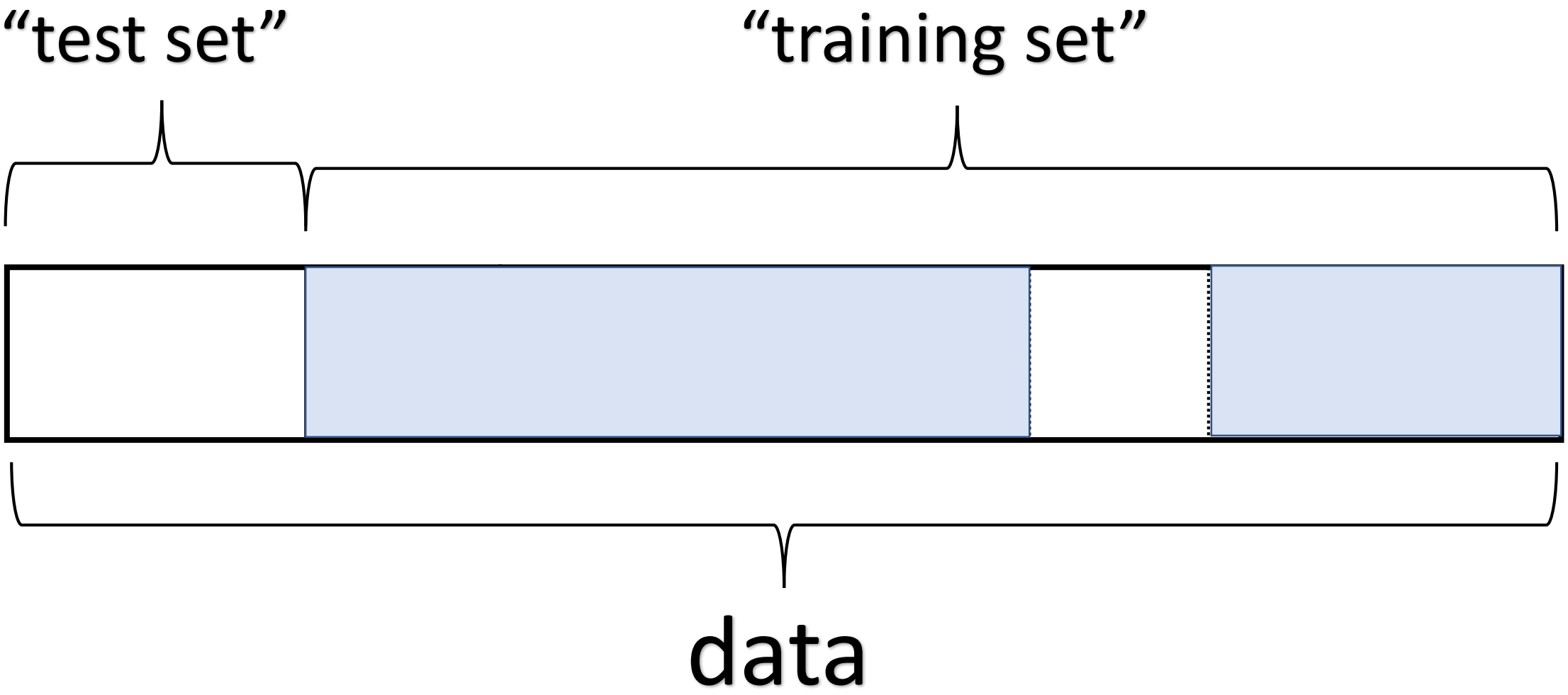

“test set”
“training set”
data

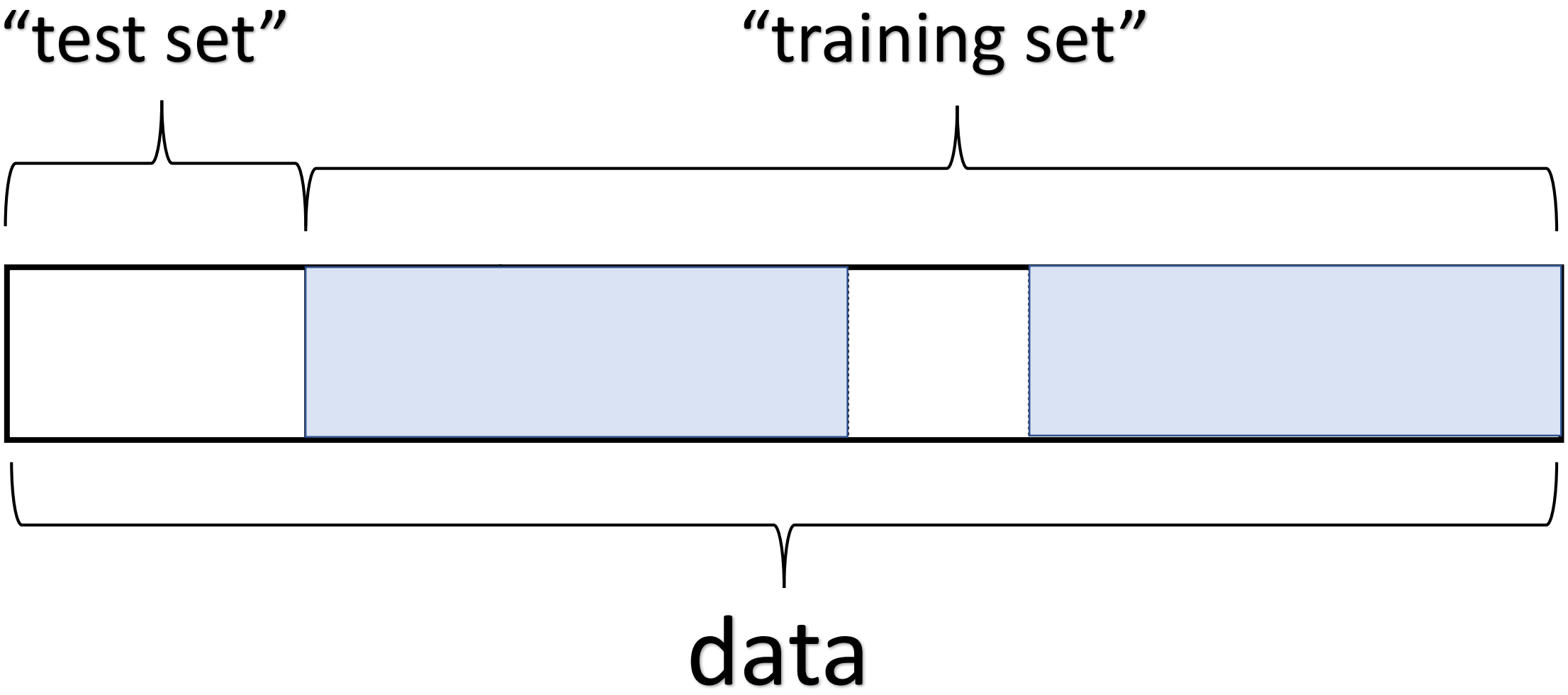

“development
“test set”
set”
“training set”
data


“test set”
“training set”
data

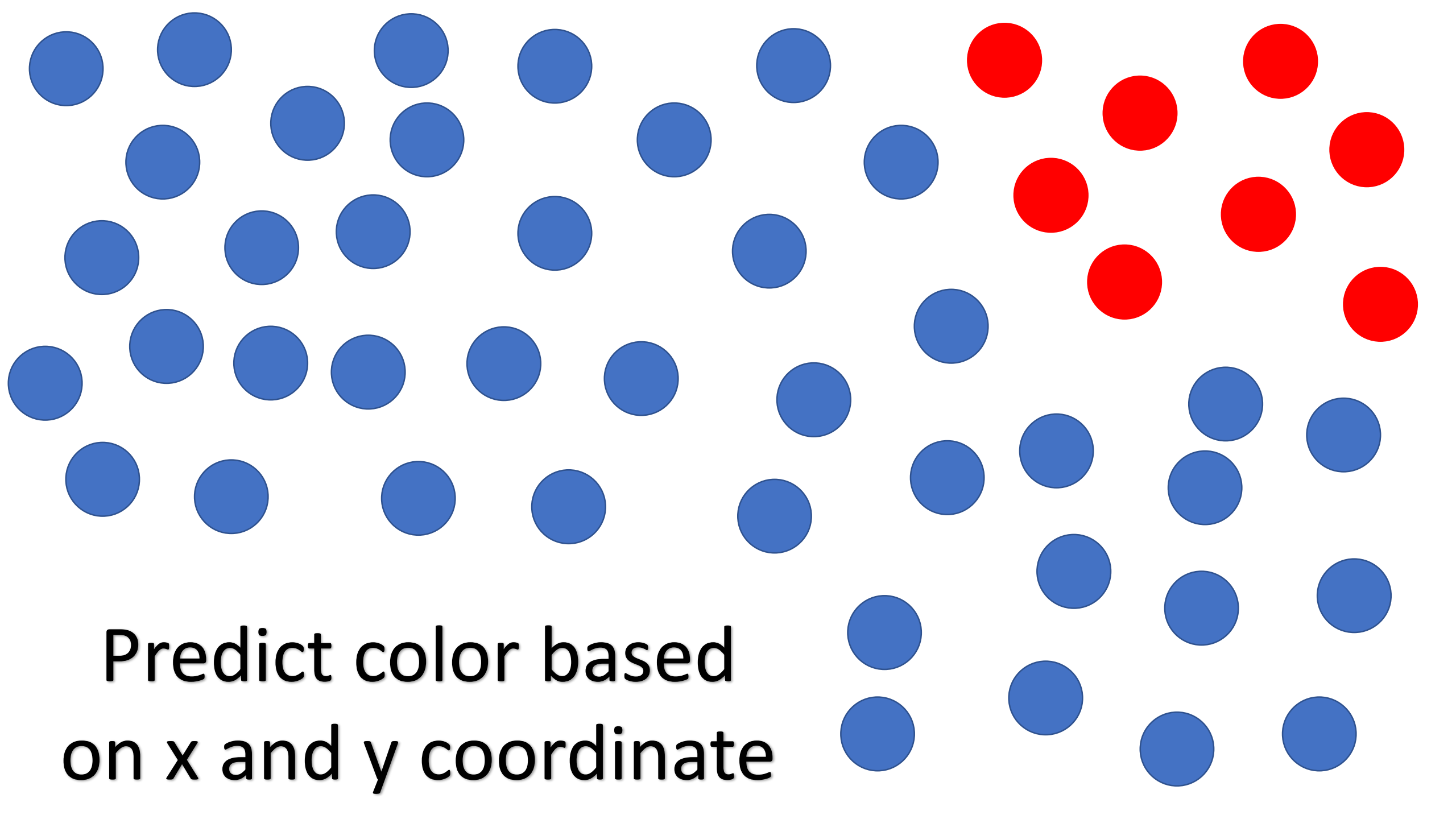

“test set”
“training set”
data

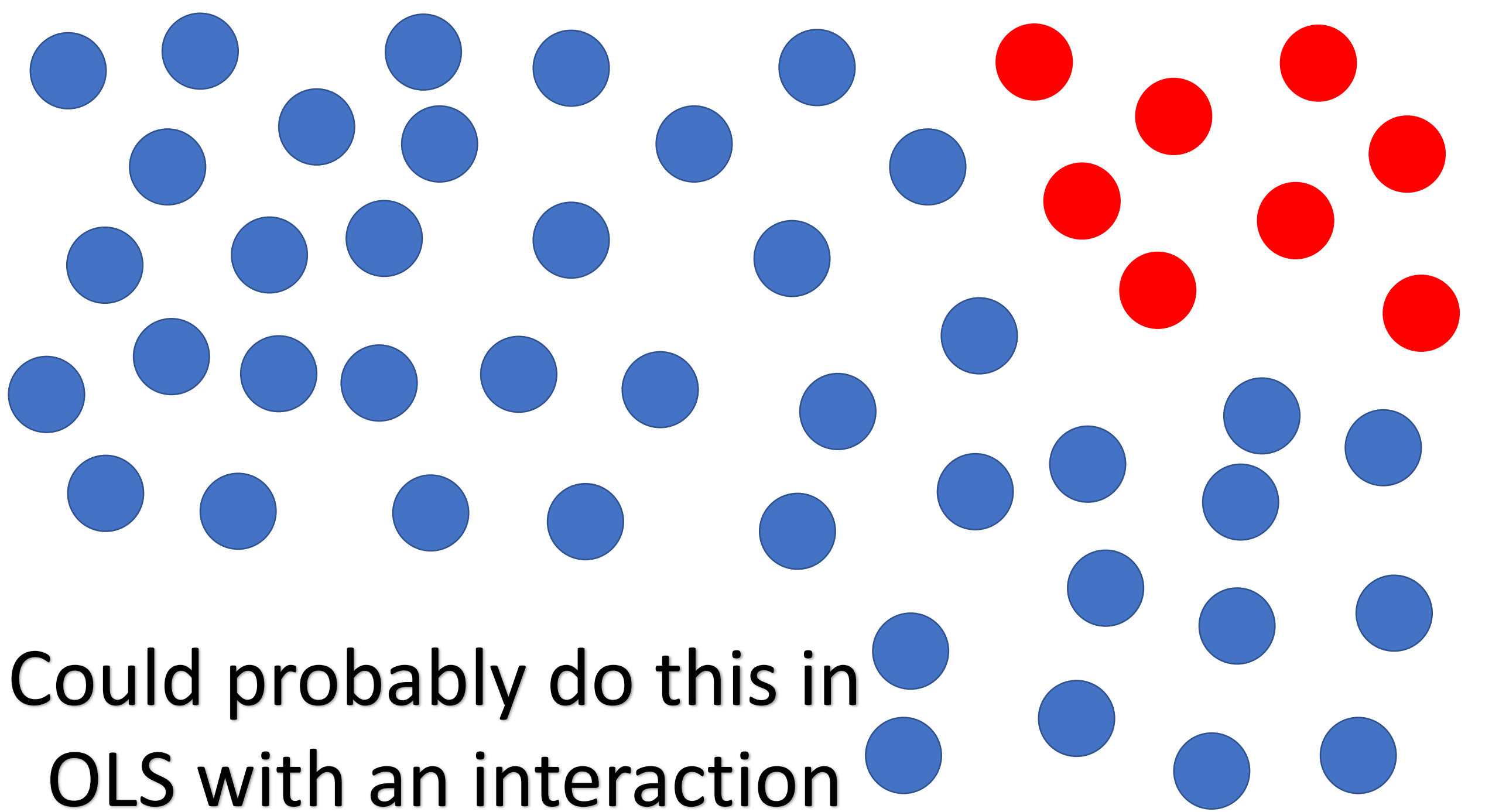

“test set”
“training set”
data

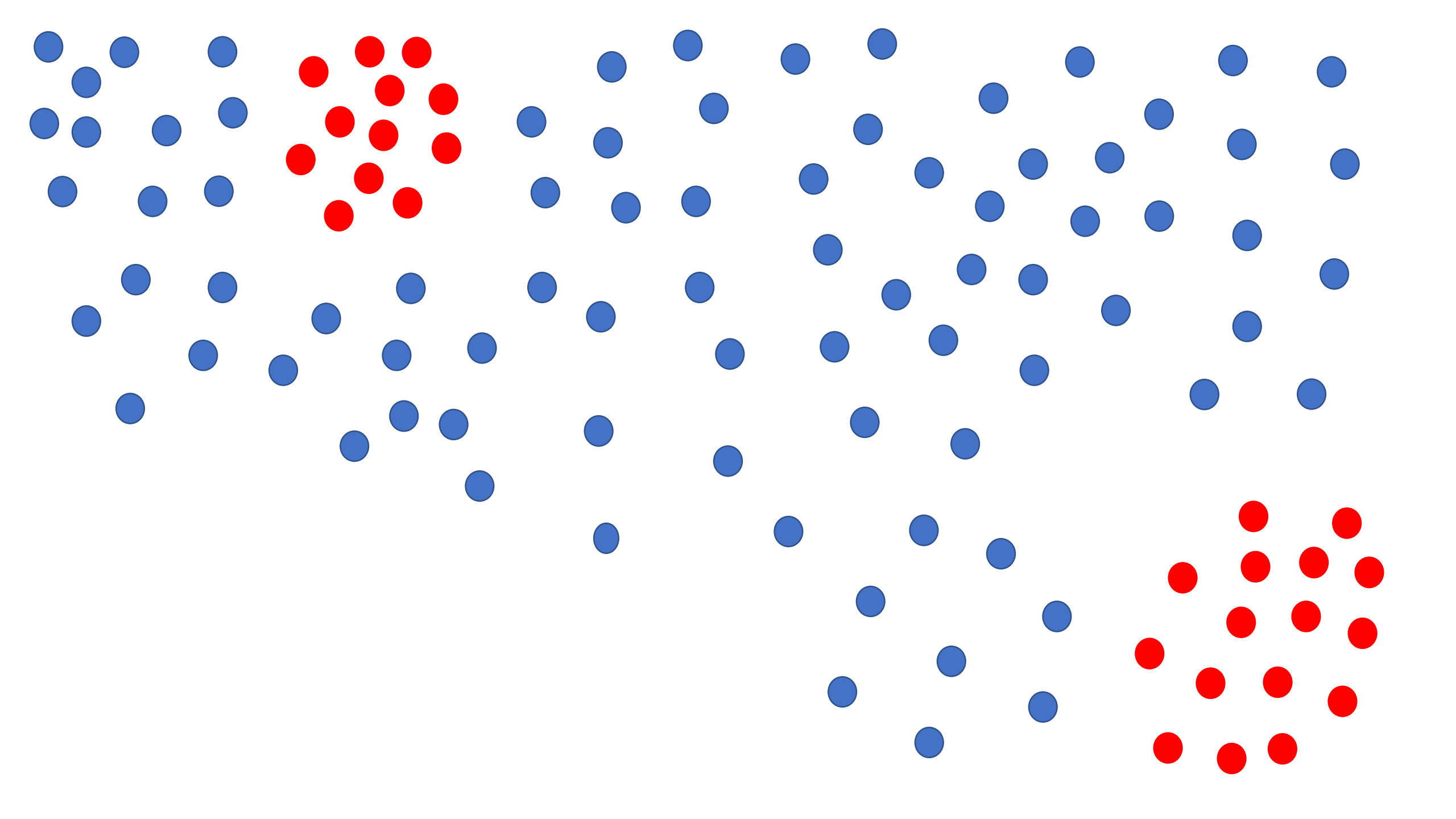

“test set”
“training set”
data

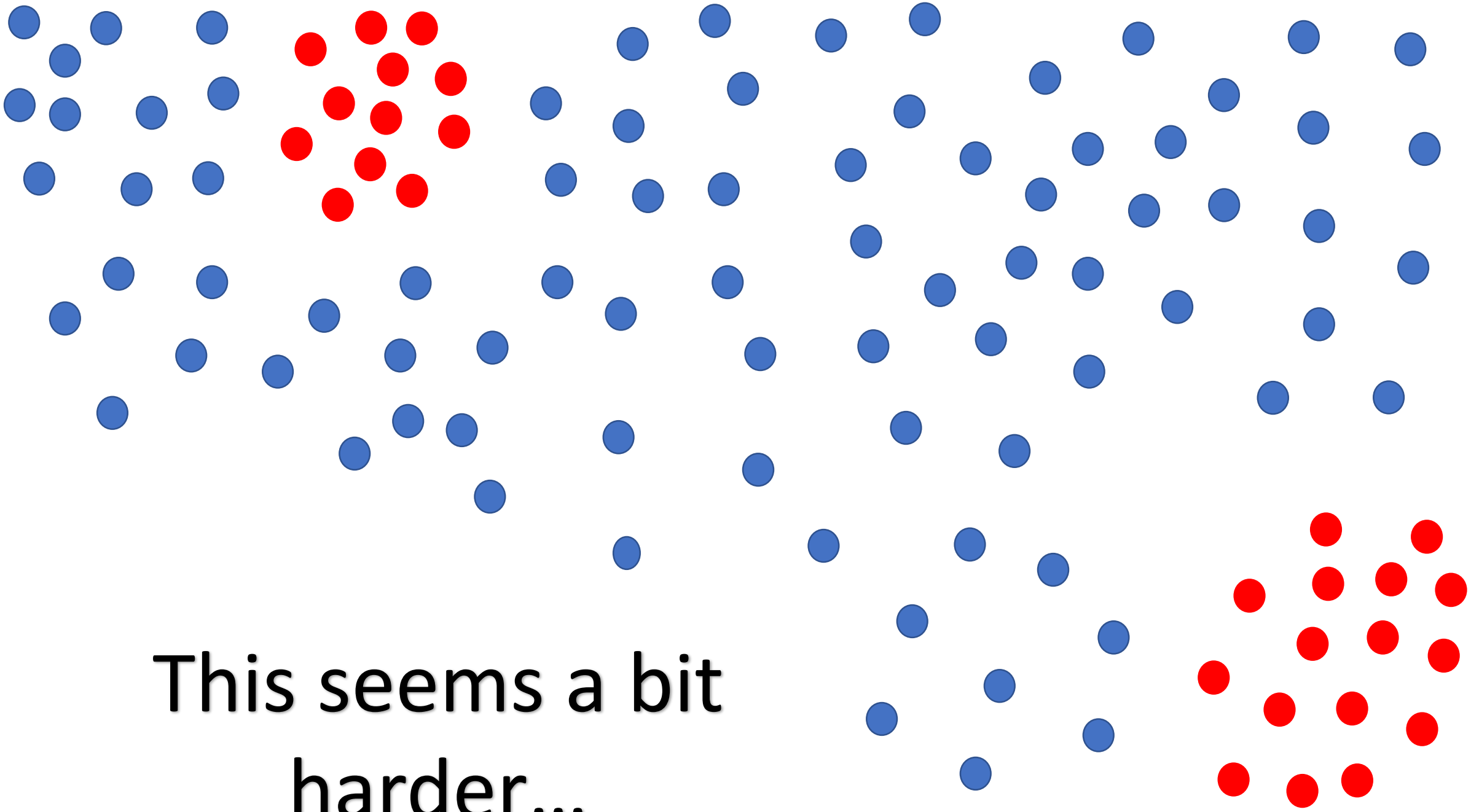

“test set”
“training set”
data

Decision trees

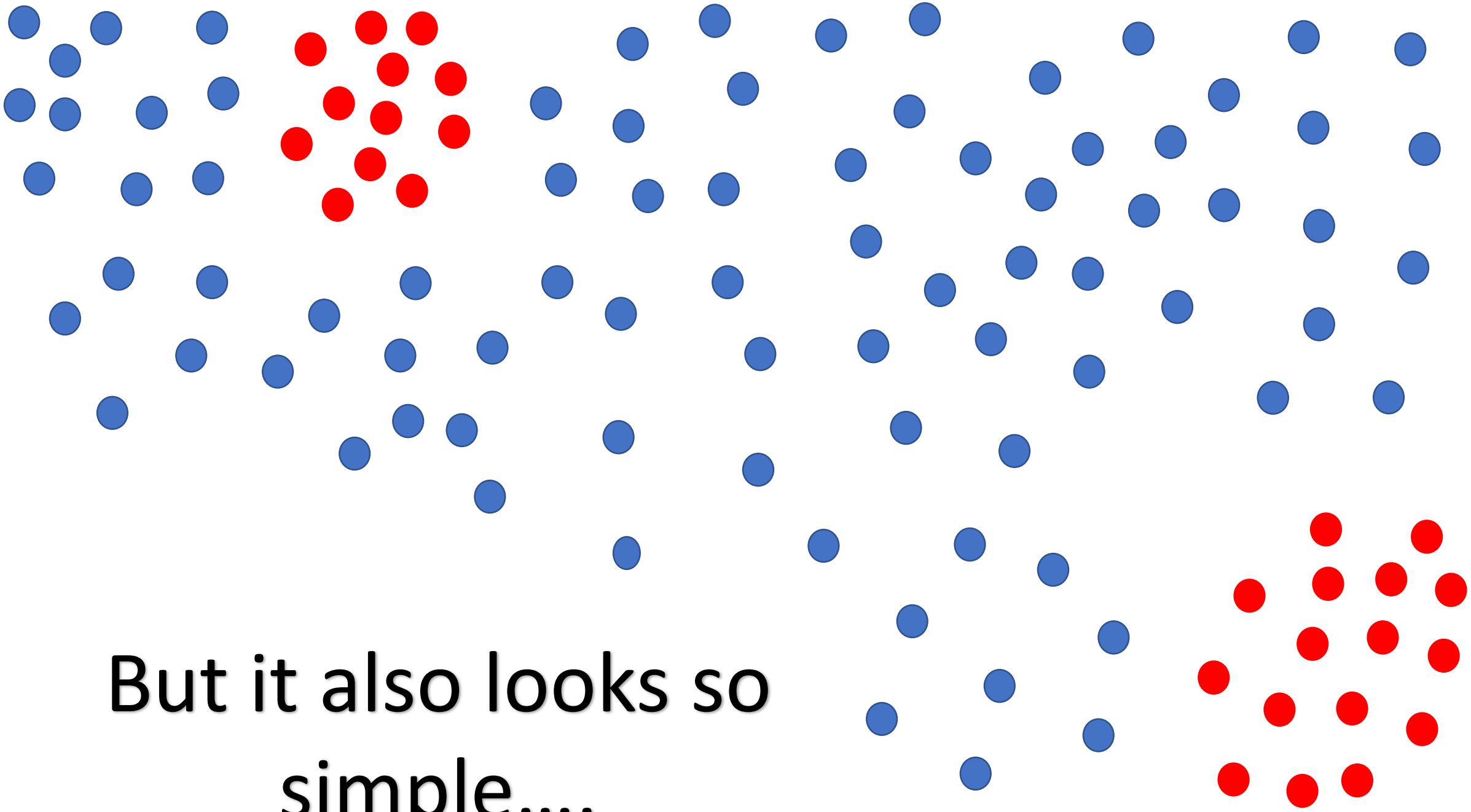

Predict color based
on x and y coordinate


Could probably do this in
OLS with an interaction

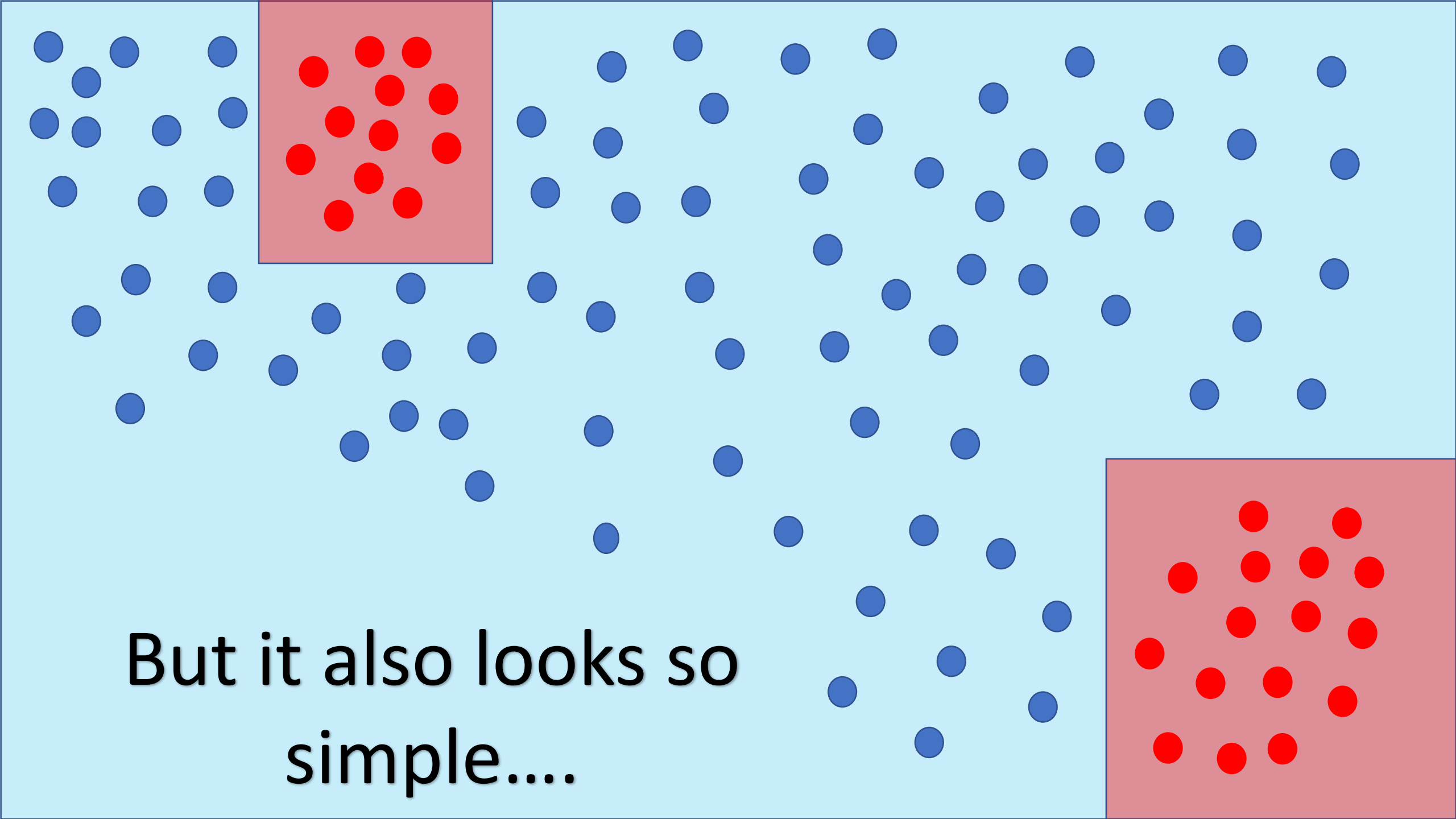


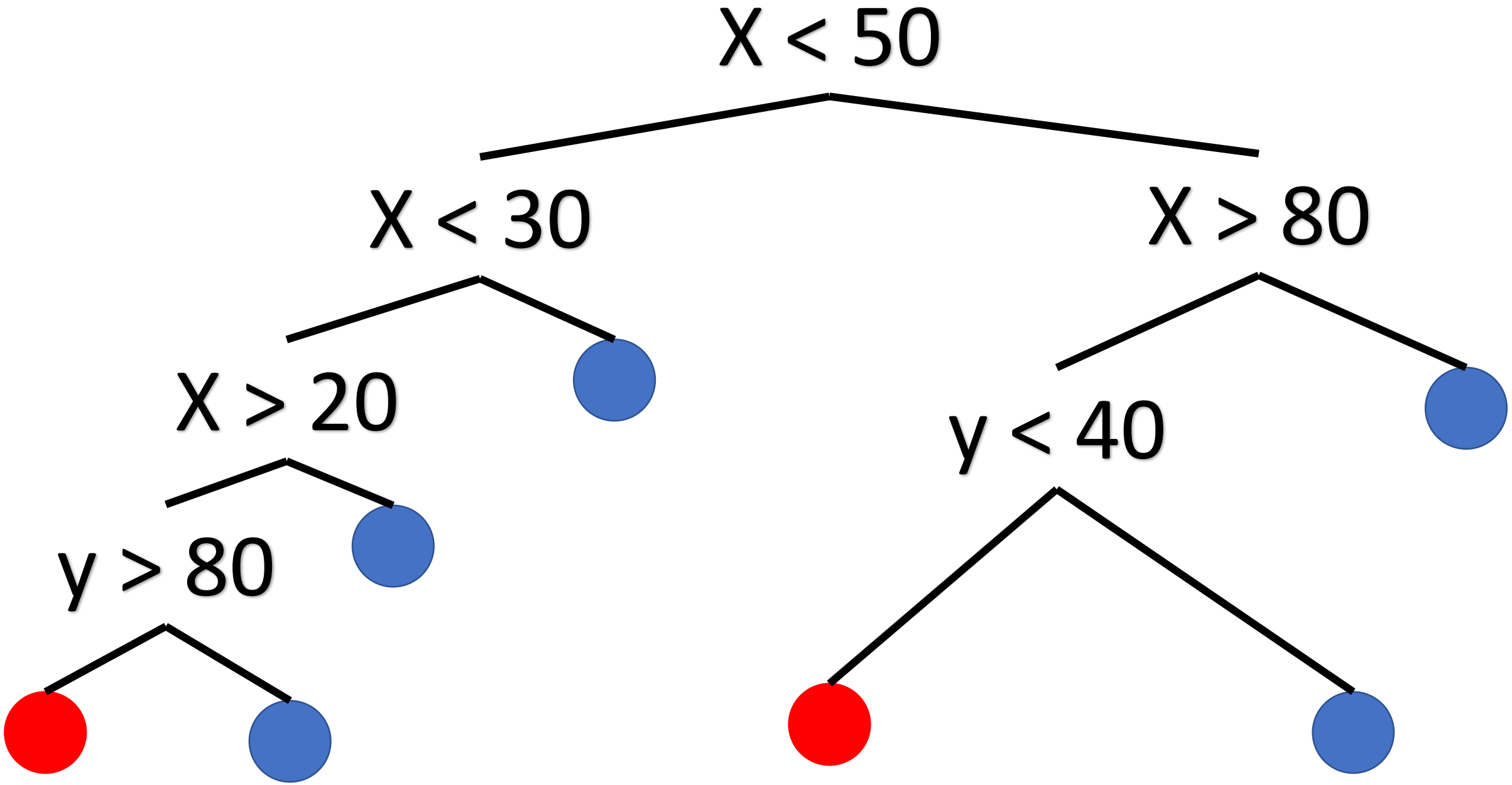


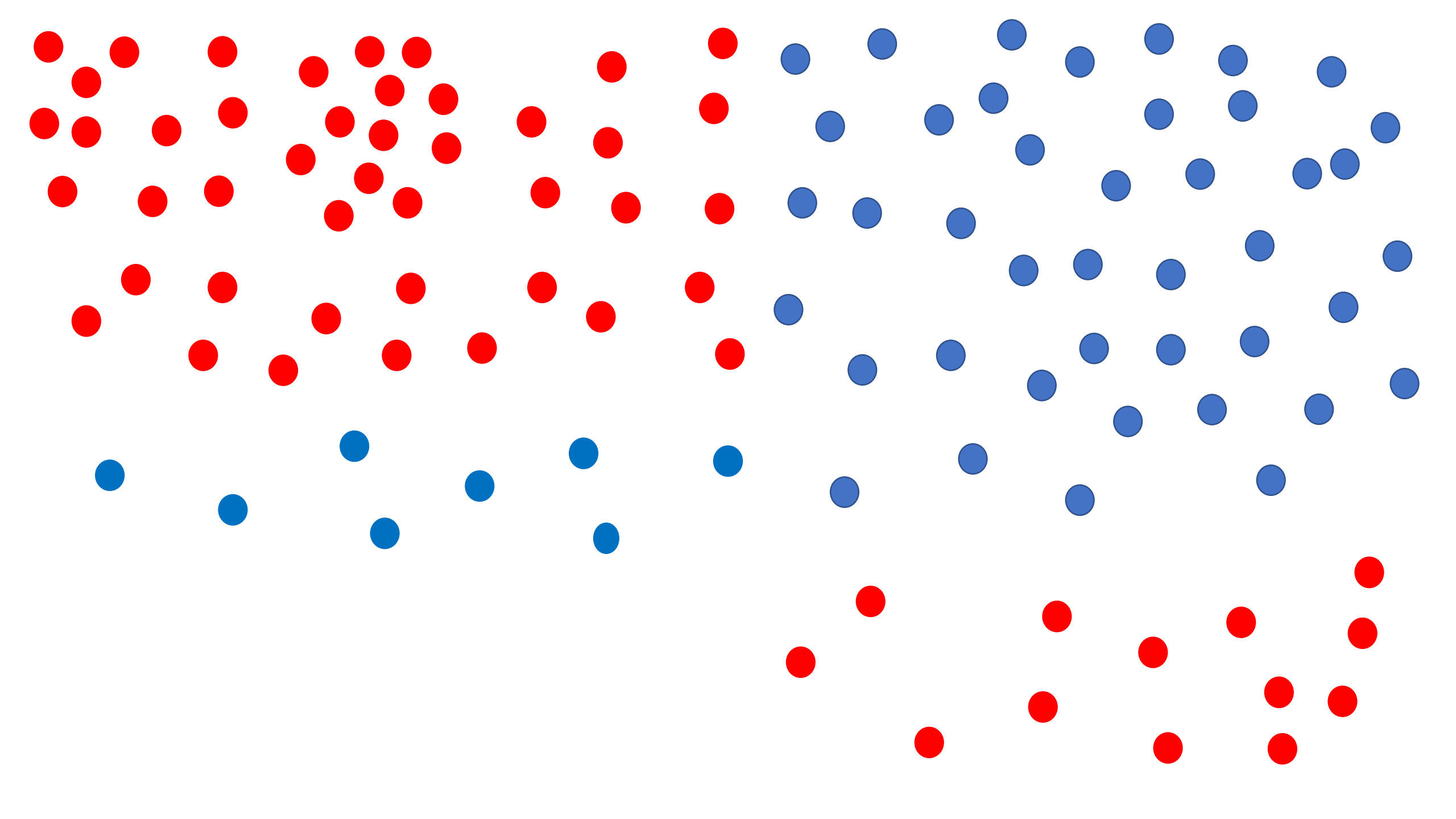
This seems a bit harder…

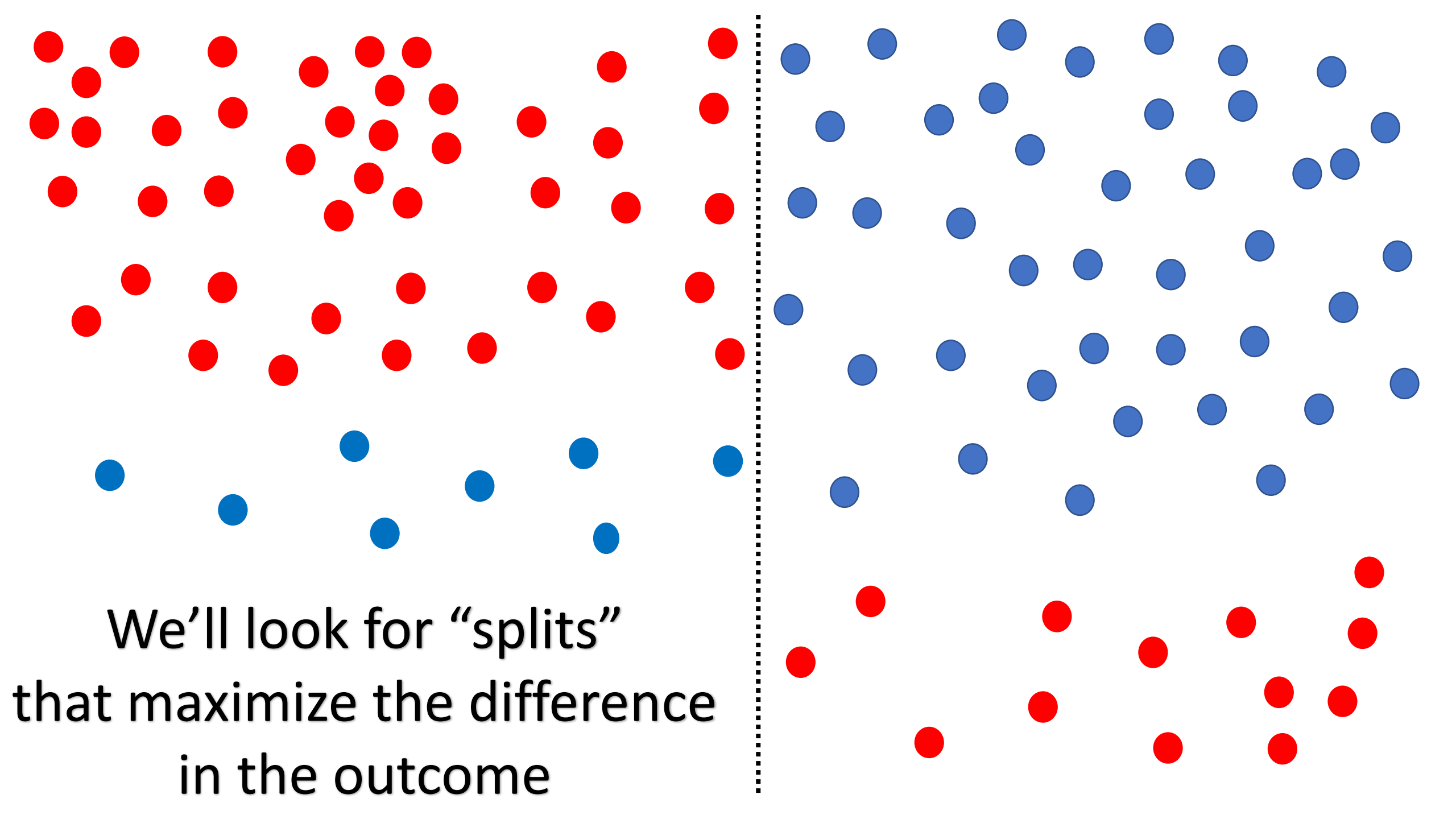


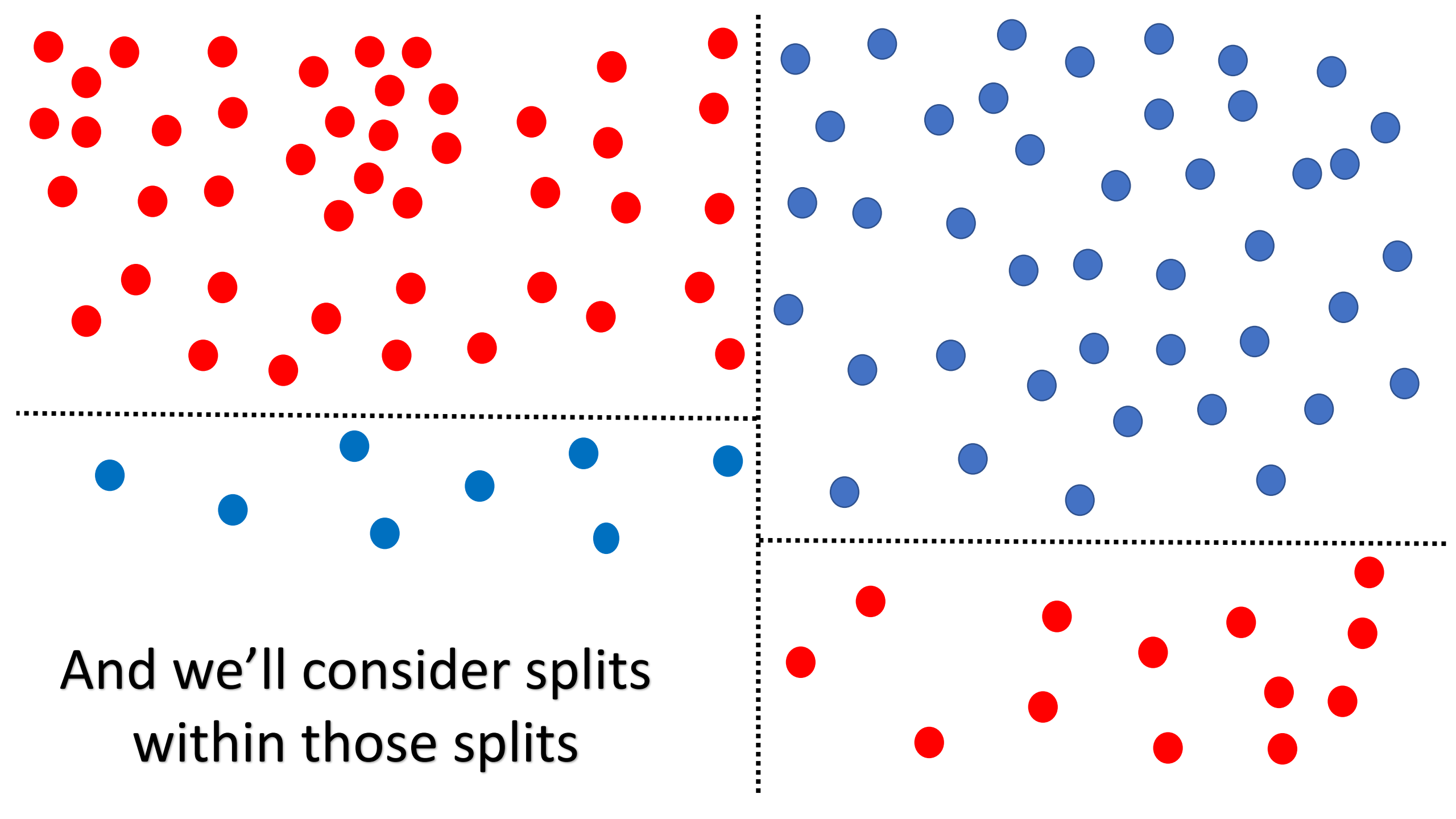
But it also looks so simple….


But it also looks so simple….


X < 50
X < 30
X > 80
X > 20
y < 40
y > 80




We'll look for "splits"
that maximize the difference
in the outcome


And we'll consider splits
within those splits

Decision Trees

- Some parameters that can be set…
  - How “deep” is the tree allowed to be (how many splits can the tree make)
  - How many nodes must exist in a “leaf” (the endpoint of a tree)
  - The “complexity factor” (how much the tree prefers few splits)
  - How do we select these? Whatever performs the best on the development set or on our k-folds!
- In general, this is a decent “out of the box” classifier (i.e. using the standard parameter values is pretty good)
- Also can do “regression trees” (on continuously valued variables)
- Generally good if variables are somewhat meaningful on their own
- Very fast to train

Random forests

One in the hand isn’t worth 10,000 in the forest

- Decision trees, in many contexts, tend to over-fit to the data (and don’t generalize well)
- Let’s take a random sample of the training set (say 10% with replacement) and “grow” a decision tree that overfits on that data
- Let’s do this 10,000 times so we have a “forest” of 10,000 trees which, on their own, perform poorly
- Now, when we predict, take the average prediction over the 10,000 trees and make that prediction
- This is a general meta-algorithm that can be used on many different classifiers (for instance LASSO), but is really popular to use with decision trees
- Can also randomly sample variables AND observations

So, as social scientists, why do we care about this?

- Some points in the research process can benefit from using machine learning
  - Classify tweets as “civil” or “uncivil”; emails as being “formal” or “informal”
  - Classify images of neighborhoods to get a measure of gentrification
- It turns out there are ways to incorporate these techniques into causal inference procedures, or ways to slightly modify them in order to do explanation
- To really understand the difference between prediction and explanation you need to at least kind of understand both