Introduction to Machine Learning

INTERSECT

Introductions

Technical Details

- http://www.cs.waikato.ac.nz/ml/weka/
- http://prdownloads.sourceforge.net/weka/datasets-UCI.jar

INTERSECT

Exercise One - A toe in the water

- At the end of the instructions you will see a matrix of scatter plots.
  - look at “class vs class” and experiment with “jitter”

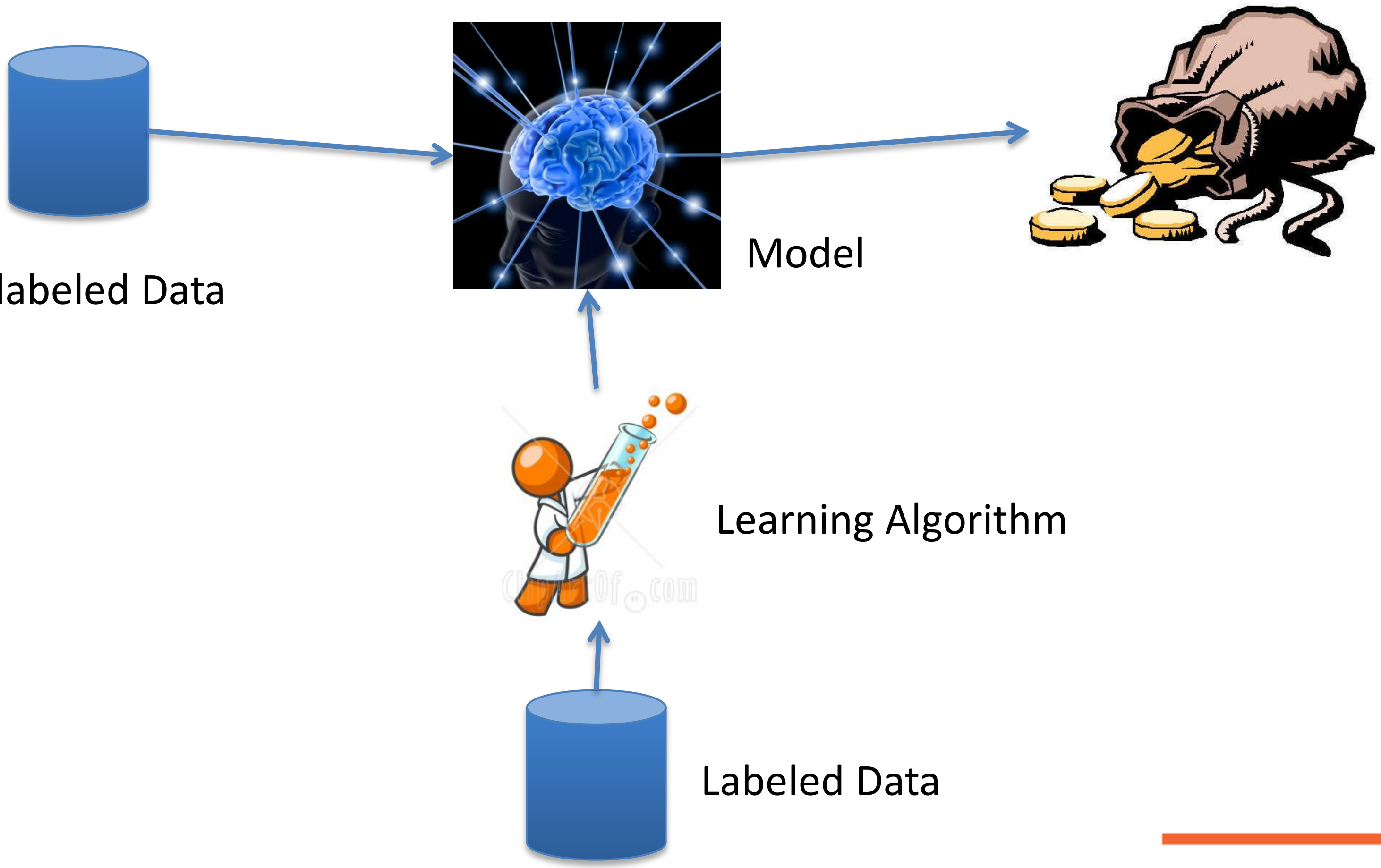
Iris – we're going to get familiar with them

Versicolour

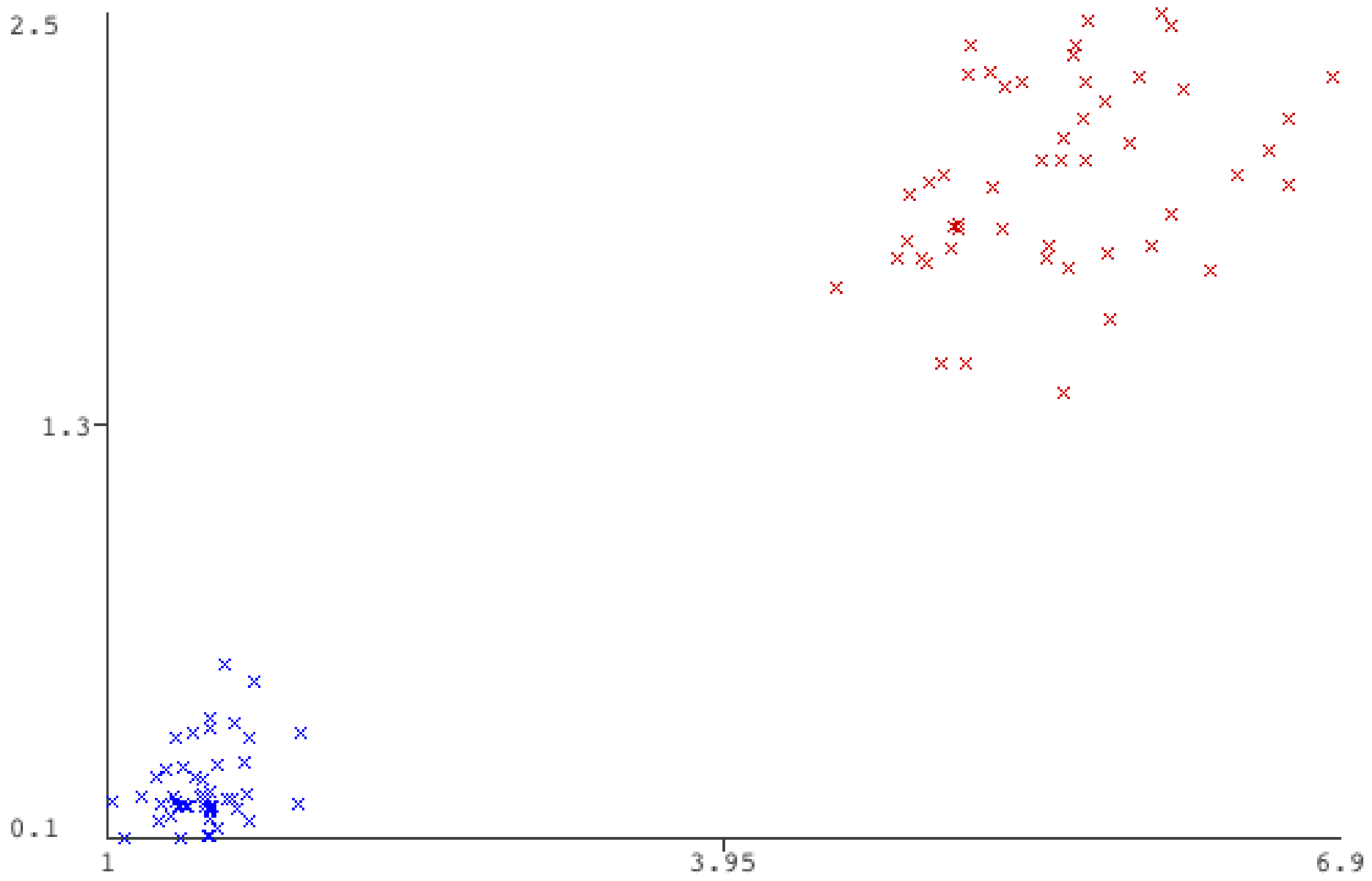
Virginica

Setosa

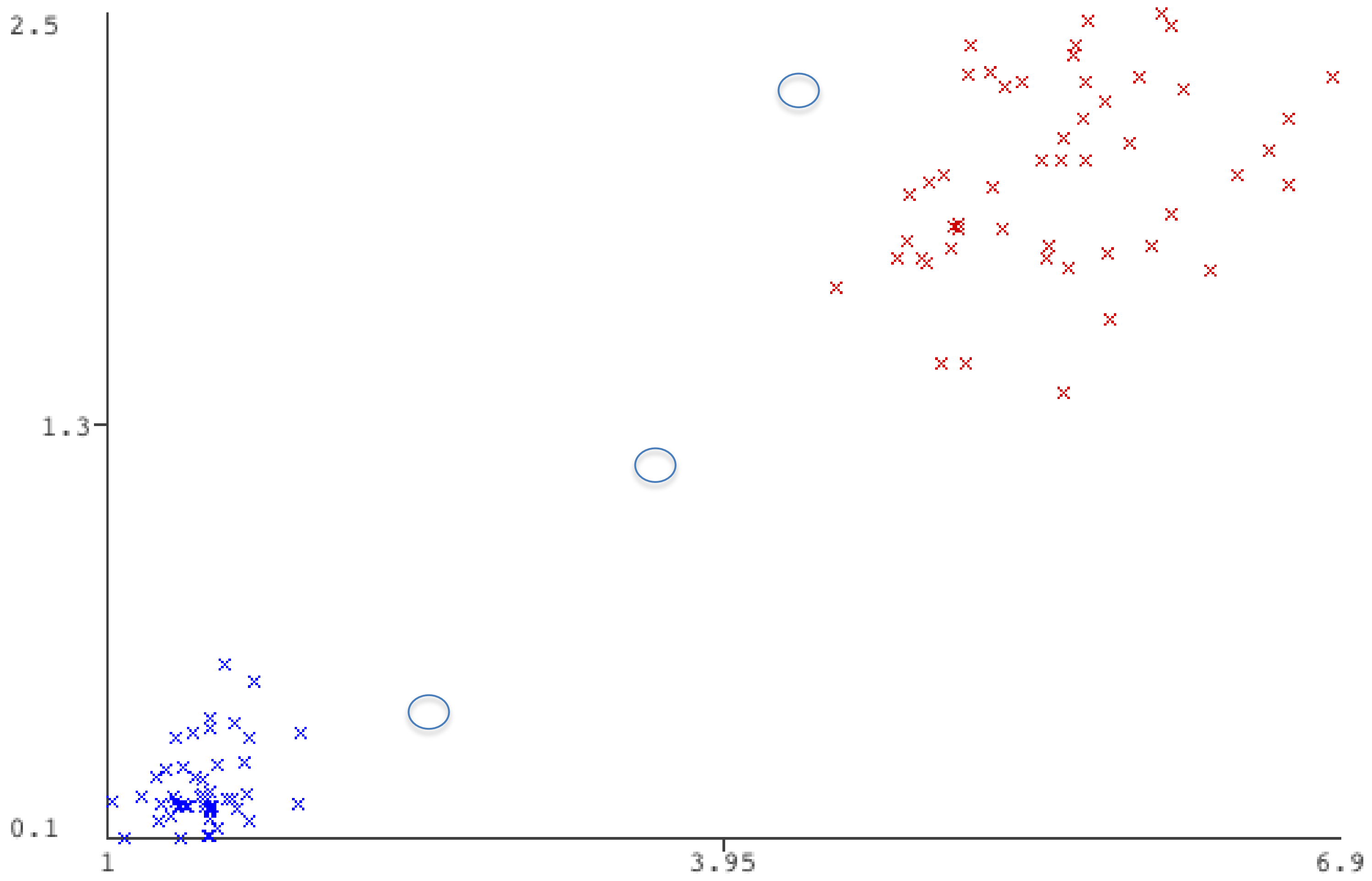
Some terminology



What is machine learning – motivating example

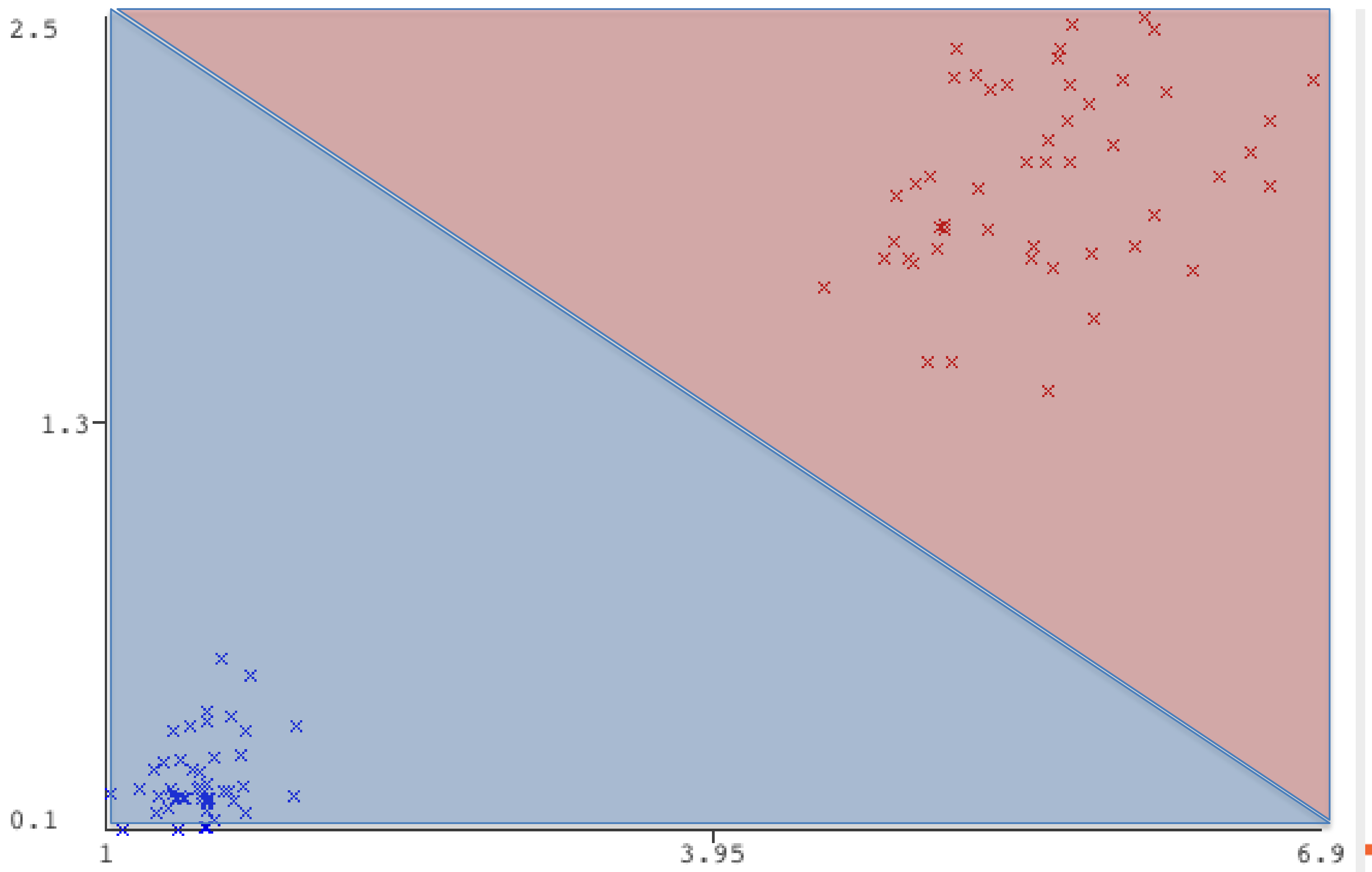


What is machine learning – motivating example

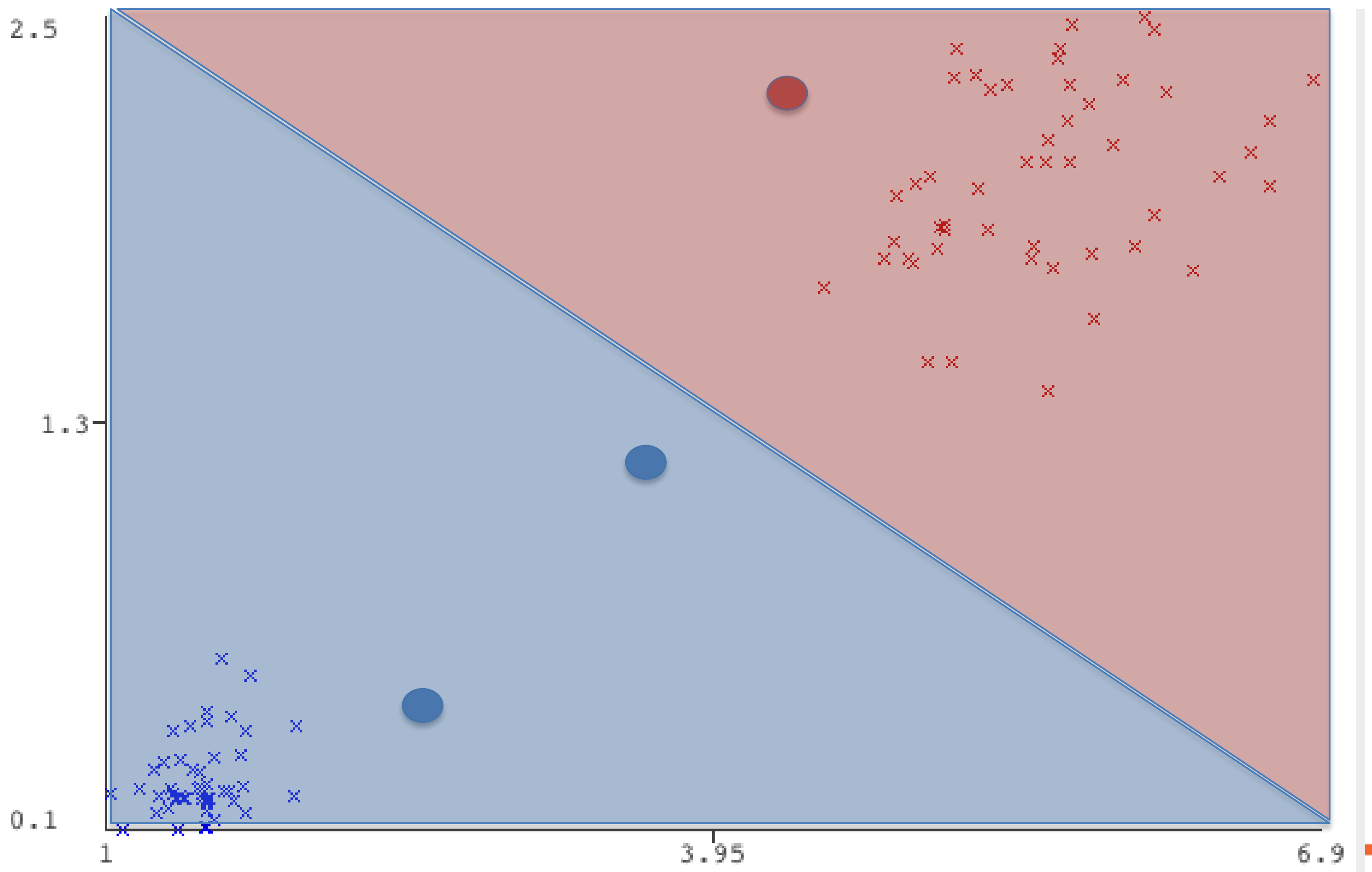


INTERSECT

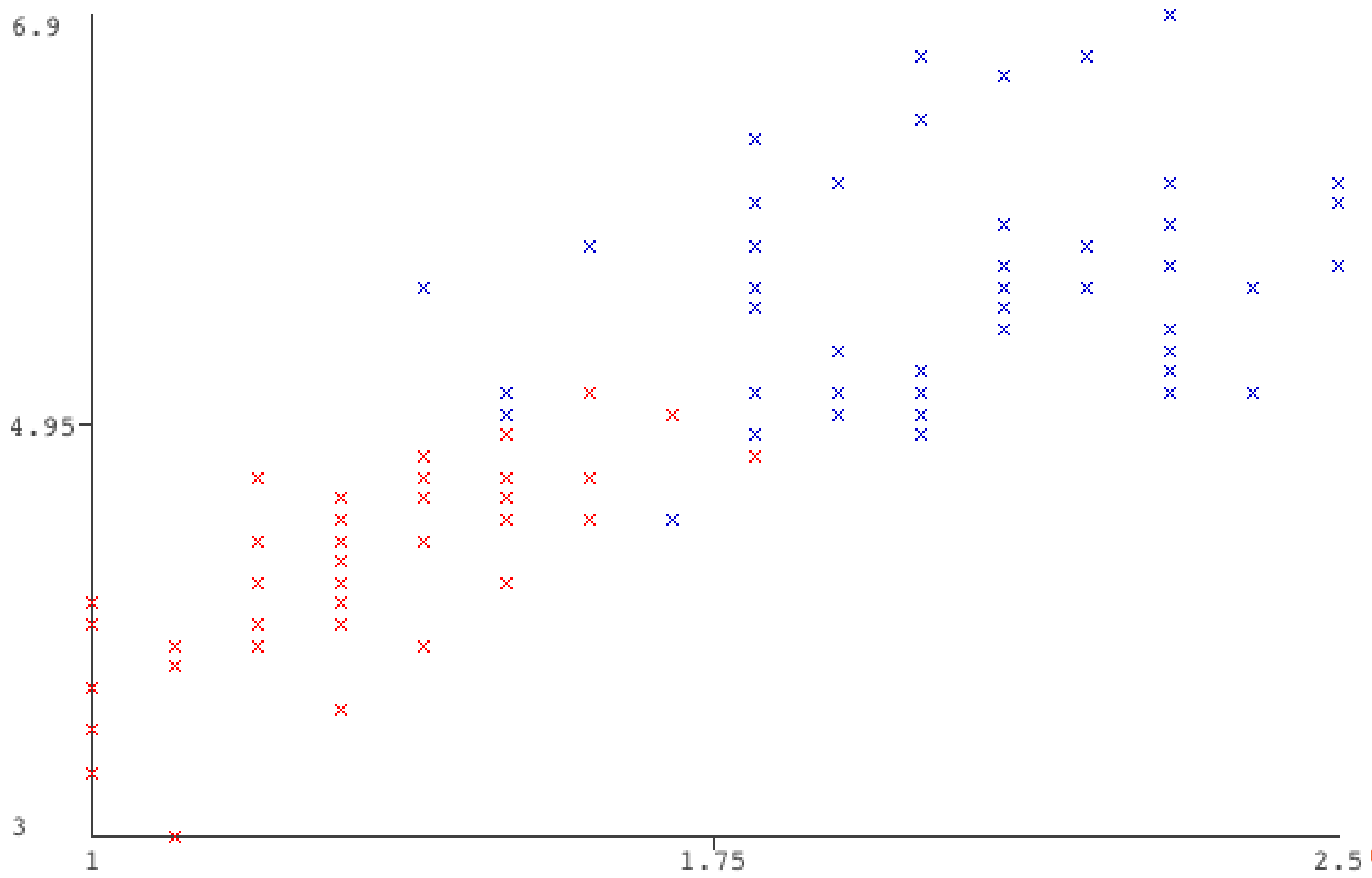
What is machine learning – motivating example



What is machine learning – motivating example

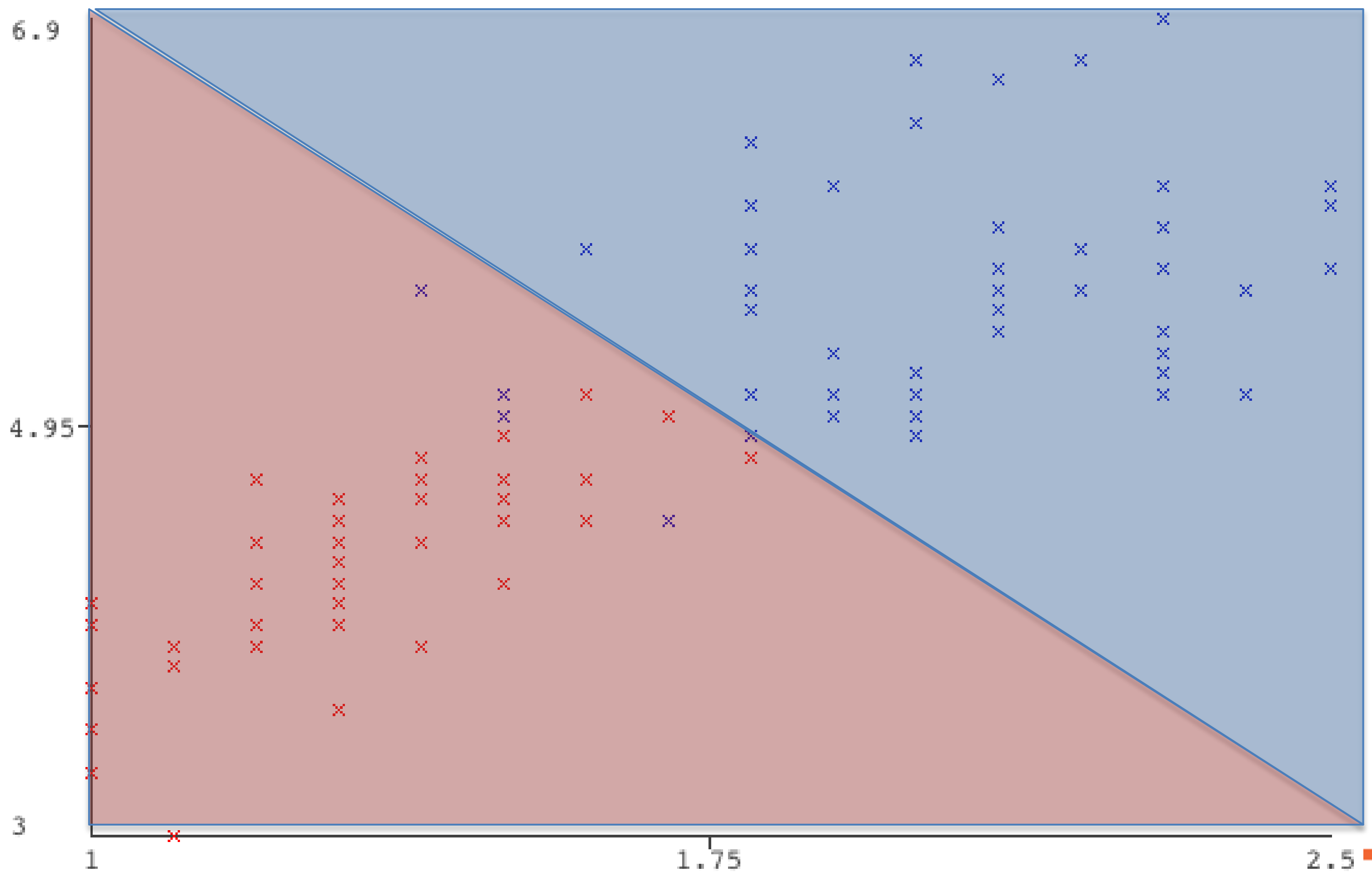


Some complications - separability



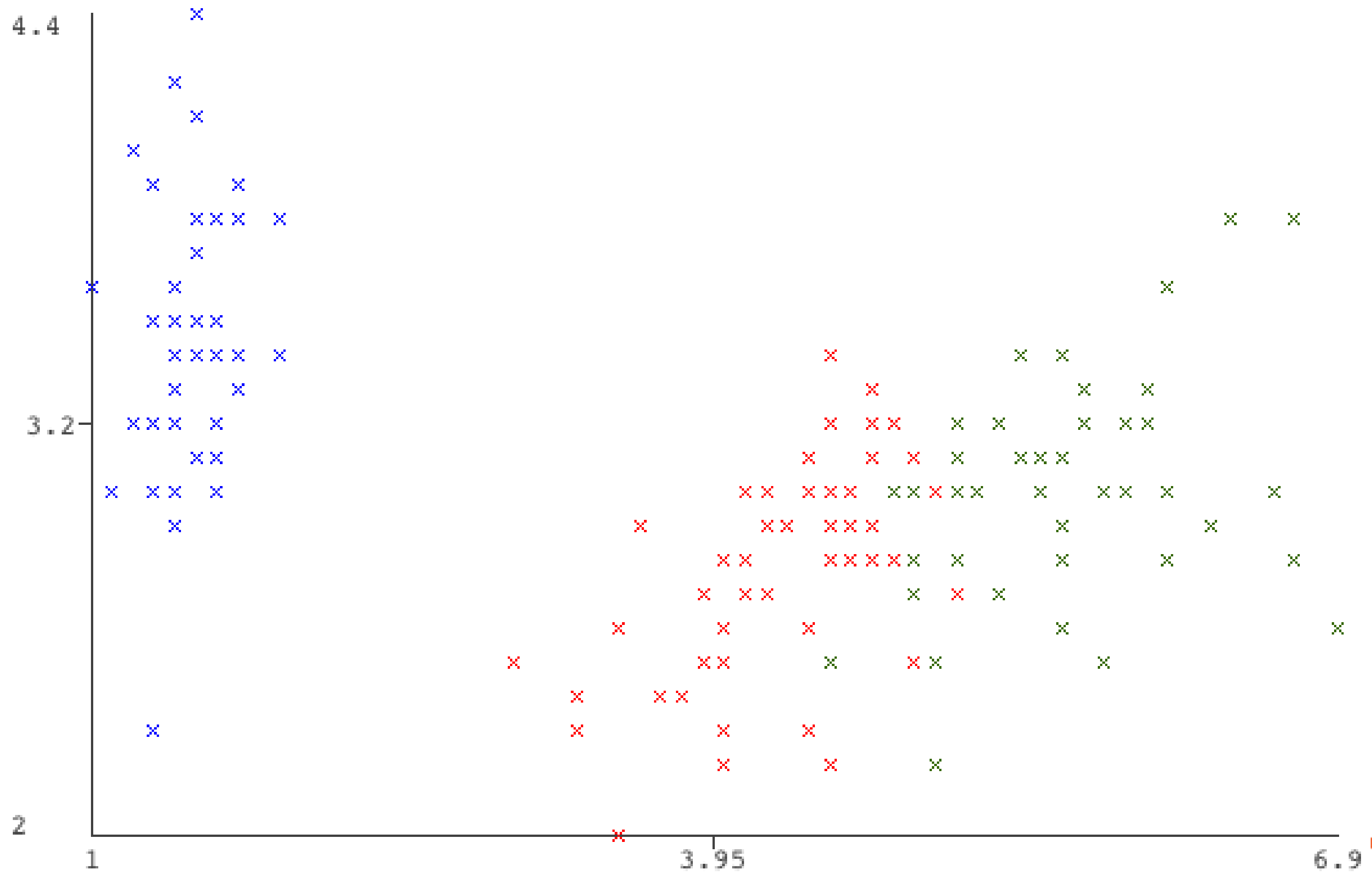
INTERSECT

Some complications - separability



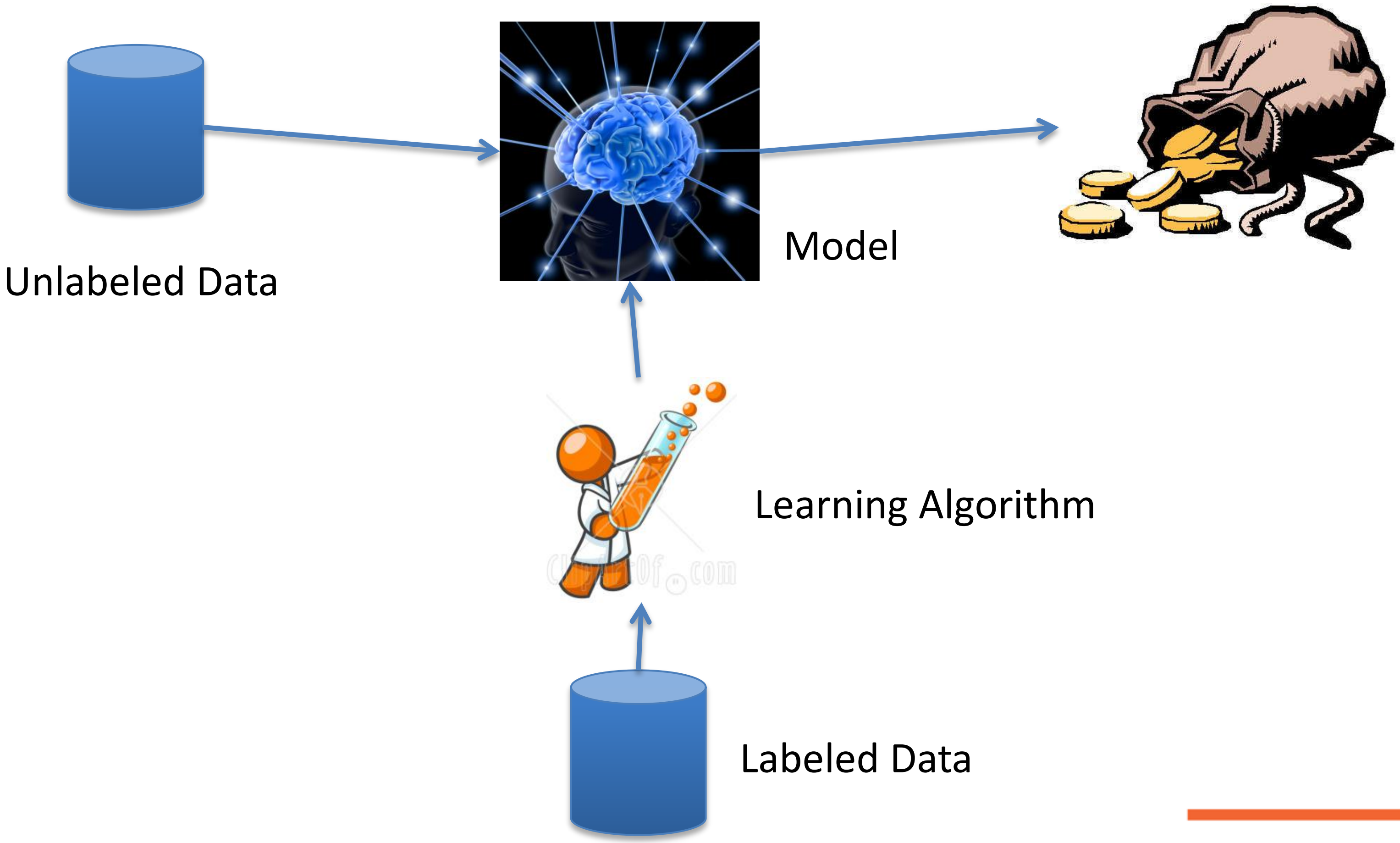
INTERSECT

Some complications - > 2 classes

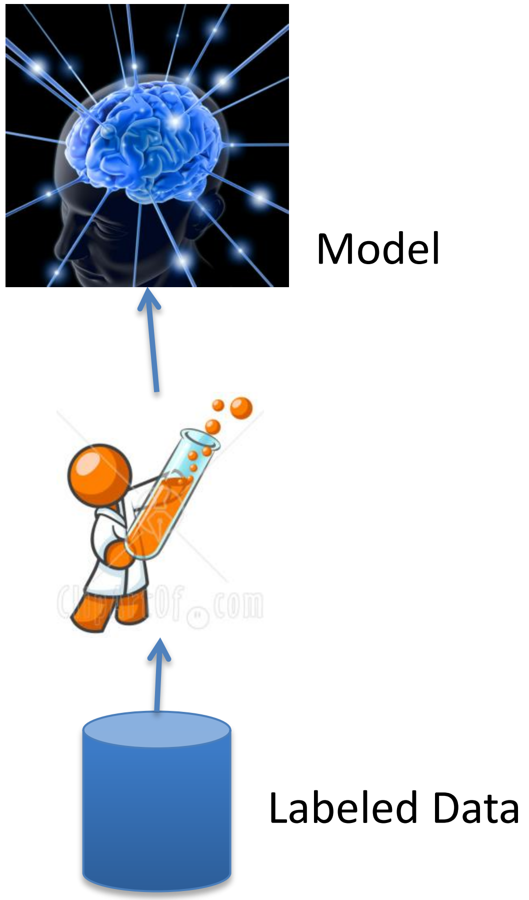


INTERSECT

What is machine learning?

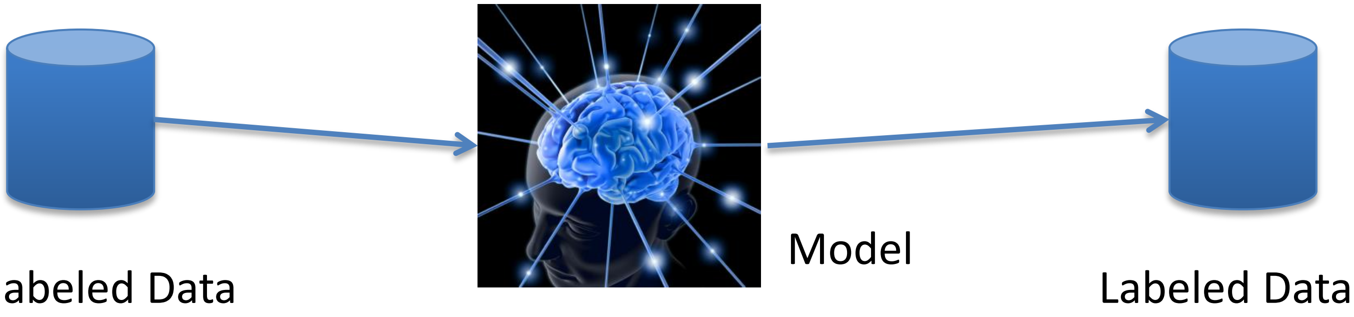


Training / Model Building



INTERSECT

Recall / Testing



Why do we care?

- Machines can
  - sometimes do things faster/better/longer/more repeatably than humans
  - be situated and replicated in difficult, dangerous, boring, on uninhabitable locations

- Machine learning vs hand crafted
  - machines can discover some types of patterns better than us
  - can learn once situated

- Lots of real-world problems can be cast as machine learning problems (how to do it is the take-away from this workshop)

What does the data look like?

Attribute

Instance

Sepal Length	Sepal Width	Petal Length	Petal Width	Variety
5.1	3.5	1.4	0.2	Setosa
7.0	3.2	4.7	1.4	Versicolor
7.1	3.0	5.9	2.1	Virginica
7.6	3.0	6.6	2.1	Virginica
…	…	…	…	…

Dataset

INTERSECT

Some kinds of machine learning

Attribute

Target

Instance

Sepal Length	Sepal Width	Petal Length	Petal Width	Variety
5.1	3.5	1.4	0.2	Setosa
7.0	3.2	4.7	1.4	Versicolor
7.1	3.0	5.9	2.1	Virginica
7.6	3.0	6.6	2.1	Virginica
…	…	…	…	…

Dataset

Some (other) kinds of machine learning

Attribute

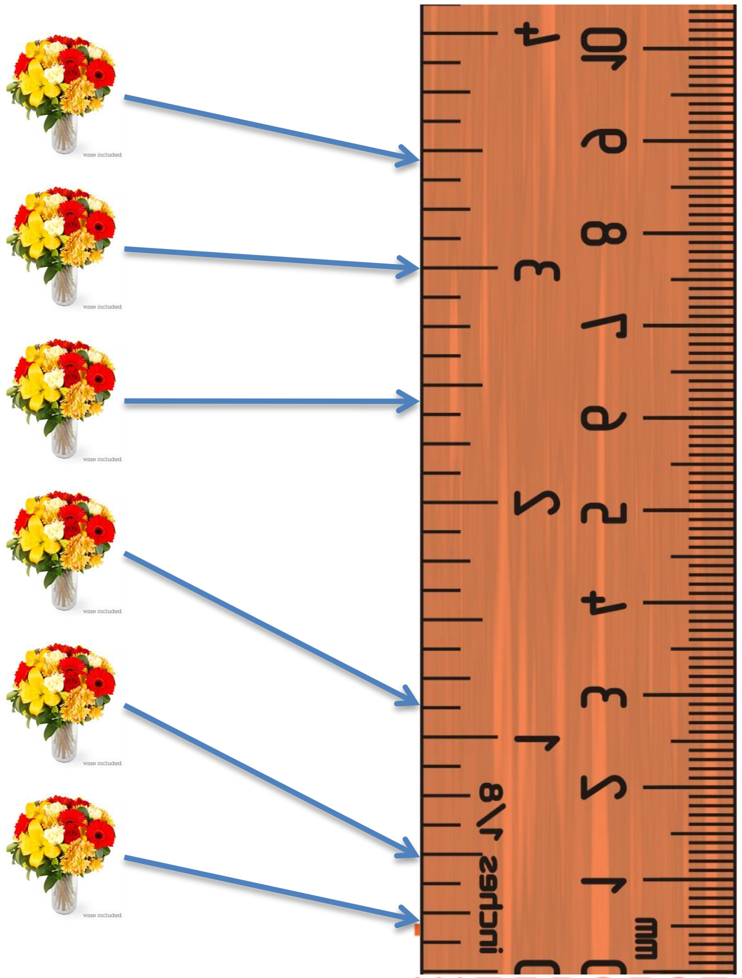
Target

Instance

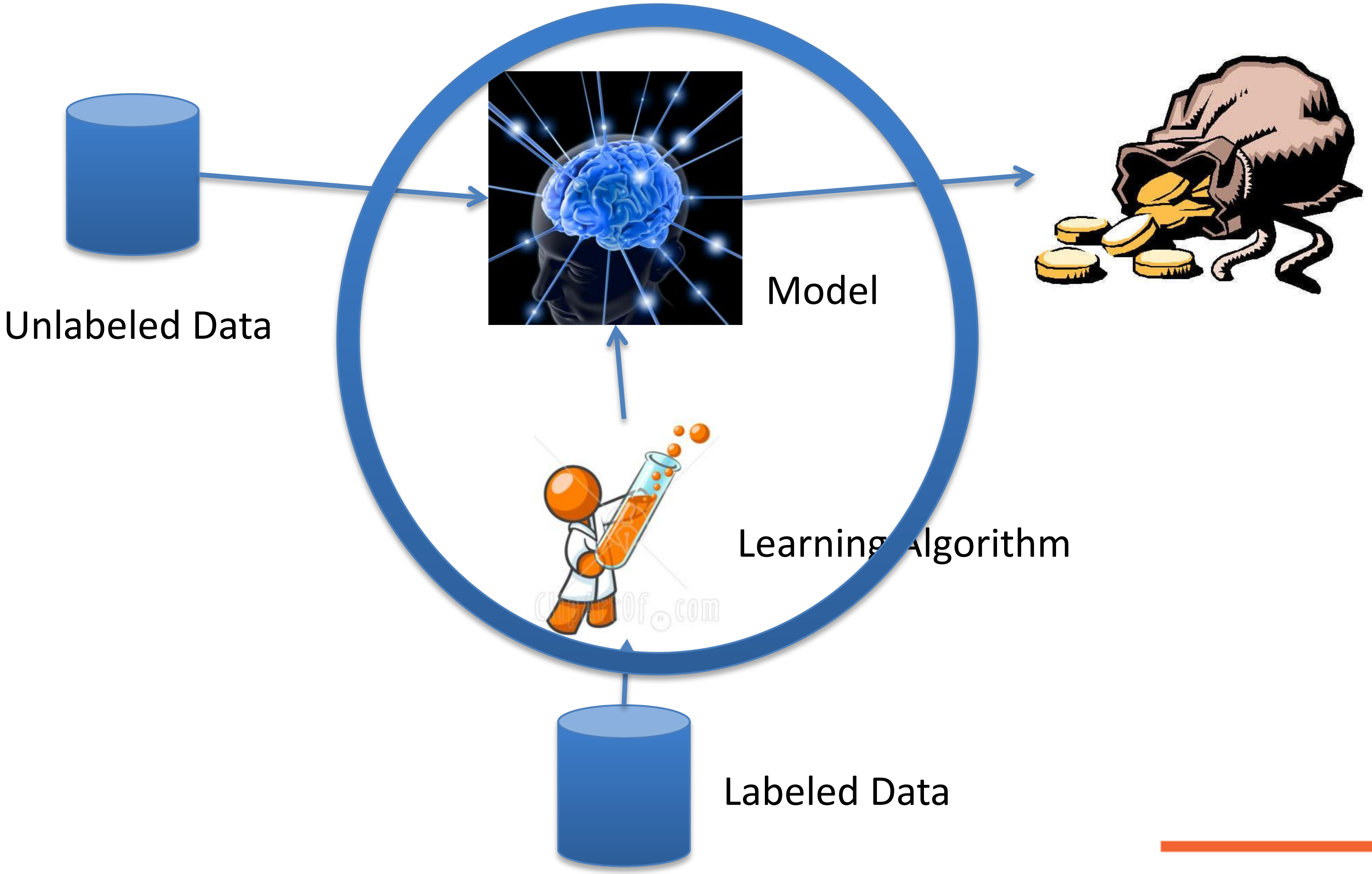
Sepal Length	Sepal Width	Petal Length	Petal Width	Variety
5.1	3.5	1.4	0.2	Setosa
7.0	3.2	4.7	1.4	Versicolor
7.1	3.0	5.9	2.1	Virginica
7.6	3.0	6.6	2.1	Virginica
…	…	…	…	…

Dataset

Some (other) kinds of machine learning



What do the models and algorithms look like?



INTERSECT

What do the models look like?

- In general, they can be any computable function from the "instance type" to the "target type"

- In practice, each machine learning algorithm falls into a family where
  - a family has the same "kind of model"
  - the algorithm used to generate the model varies
  - different families have very different "kinds of models"

- Some families' models have a very intuitive interpretation (e.g. IBK, Decision Tree) and some do not (e.g. Neural Net)

Exercise – Let’s train and test a classifier

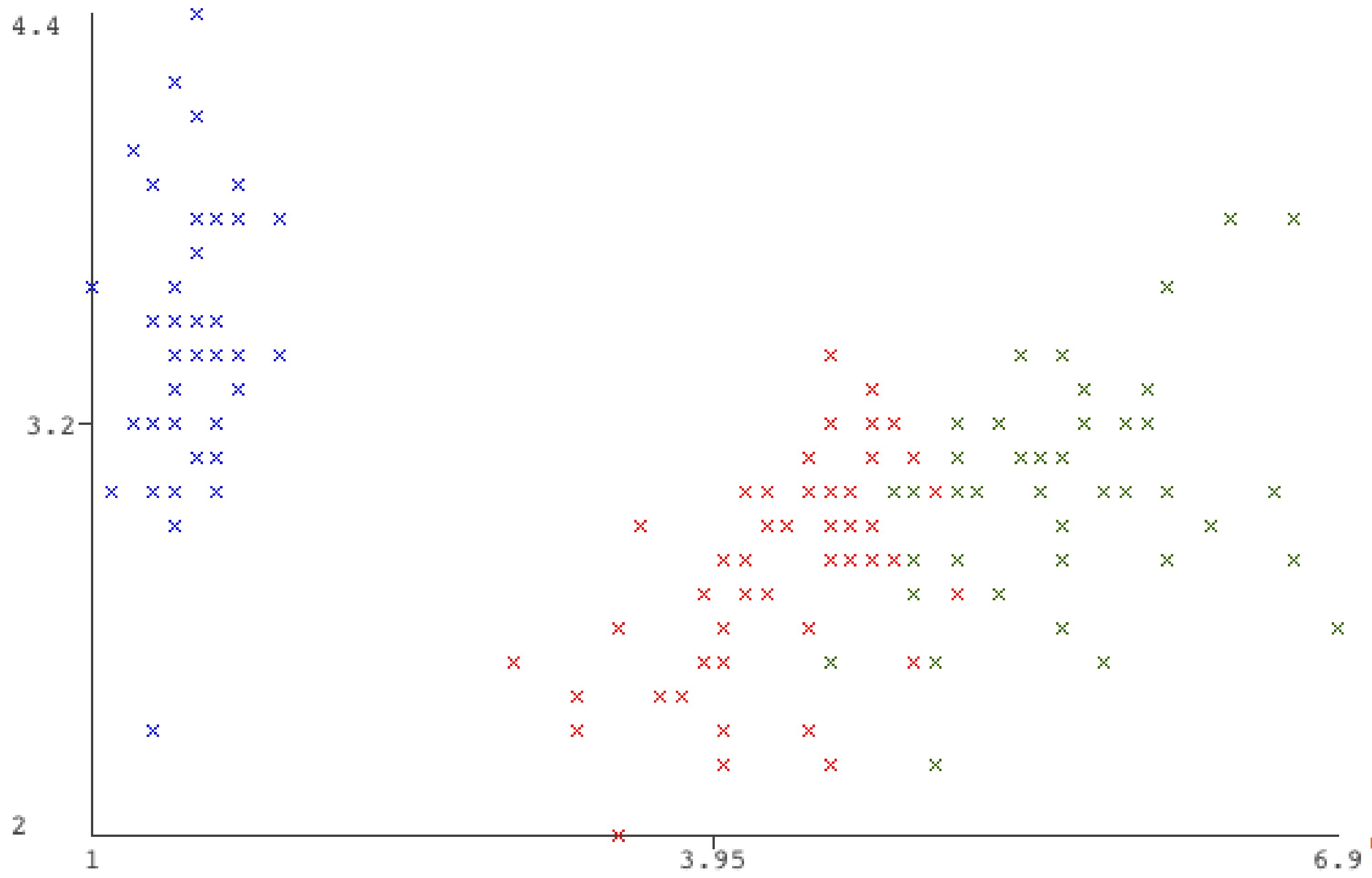
- If you don’t still have Iris loaded, load it again
- Click the “Classifier” tab
- Click “Choose” and select Lazy -> IB1
- In ‘Test Options’ choose “use training set”
- Click “Start”

- What just happened?
  - 1. The IB1 learning algorithm was trained on the test data, producing a model
  - 2. The model was tested against the training data, and its performance analysed

What algorithms will we examine today?

- IBK (IB1 is the version of this with K = 1)
- Decision Tree
- Support Vector Machines

IBK – what does a model look like?



IBK – what does the model building look like?

- Store all the instances

- Perhaps do some indexing (only works for low-dimensional spaces)

- Remember the distance metric that you're going to use
- Remember K

PARAMETERS

INTERSECT

A note about parameters

- Most algorithms have a bunch of parameters for you to fine-tune the algorithm to the problem.

INTERSECT

IBK – how does it recall from a model?

- Given an unlabelled instance
- Find the K nearest neighbours in the training set, according to the distance metric
- Return the most common label amongst those nearest neighbours

Exercise - Let’s Try IBK

- Run IBK and experiment with different values for K

- You can find it under “Lazy”
- Note that all the results from your session are kept in the Explorer
- You can click on the parameters of the learner to twiddle the parameters

Decision Tree – what does the model look like?

```
petalwidth <= 0.6: Iris-setosa (50.0)
petalwidth > 0.6
|   petalwidth <= 1.7
|   |   petallength <= 4.9: Iris-versicolor (48.0/1.0)
|   |   petallength > 4.9
|   |   |   petalwidth <= 1.5: Iris-virginica (3.0)
|   |   |   petalwidth > 1.5: Iris-versicolor (3.0/1.0)
|   petalwidth > 1.7: Iris-virginica (46.0/1.0)
```

INTERSECT

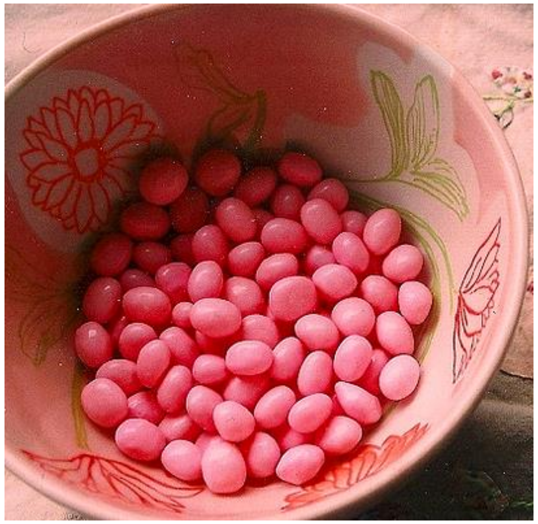
Jumbledness

INTERSECT

Jumbledness

P(Pink) =	0.125
P(Yellow) =	0.125
P(Orange) =	0.125
P(Green) =	0.125
P(Red) =	0.125
P(Black) =	0.125
P(Green) =	0.125
P(Blue) =	0.125

P(Orange)	= 0.5
P(Purple)	= 0.35
P(Pink)	= 0.15

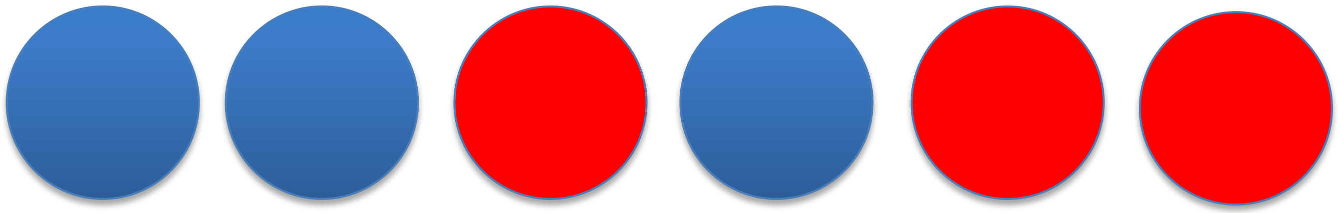


P(Pink) =	1

INTERSECT

Decision Trees – what does model building look like

- How jumbled is a set of instances?

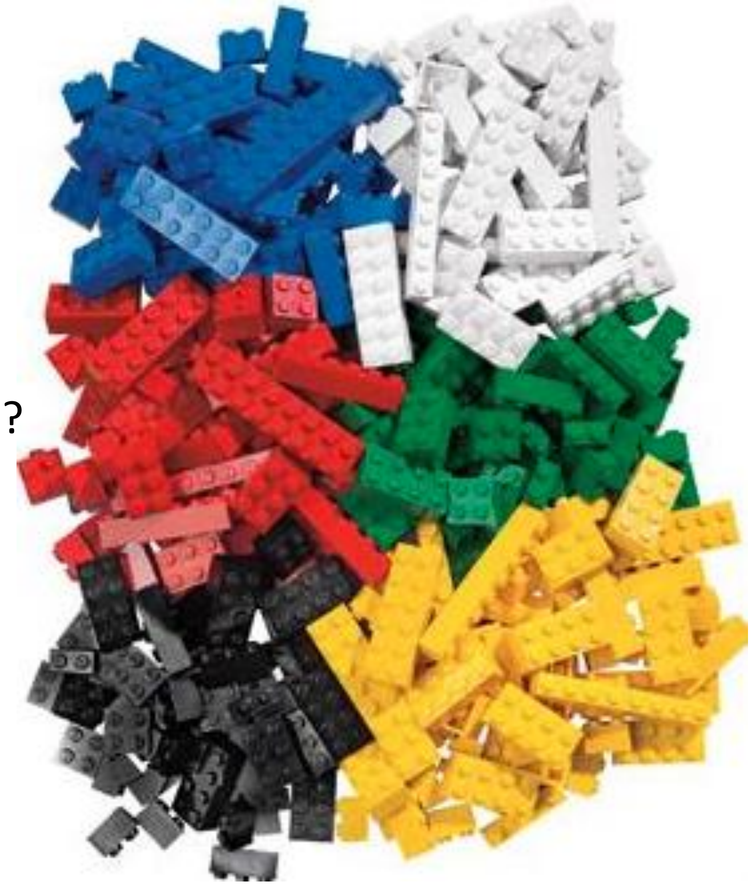


- P(Red) = 0.5 P(Blue) = 0.5

- Entropy = -0.5 * log(0.5) - 0.5 * log(0.5)
  = -(0.5 * - 1) – (0.5*-1)
  = 1

What helps resolve jumbledness?

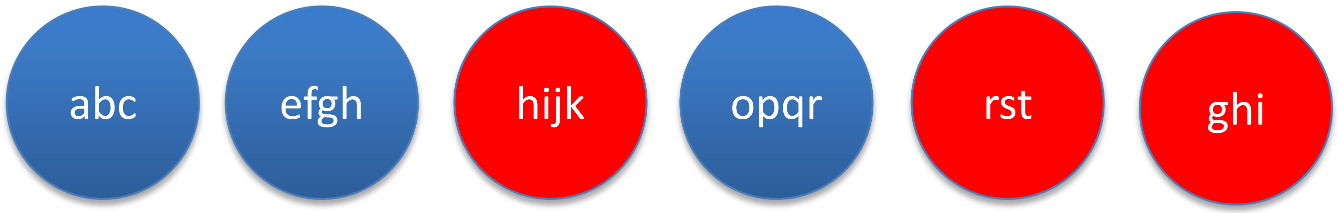
Width?
Length?
Number of dots?



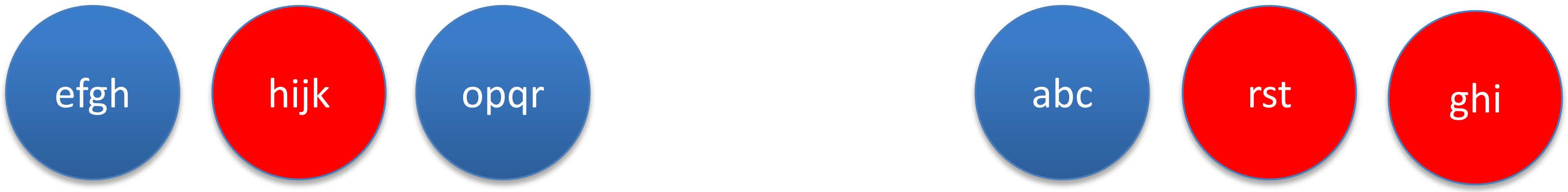
INTERSECT

Decision Tree – what does model building look like

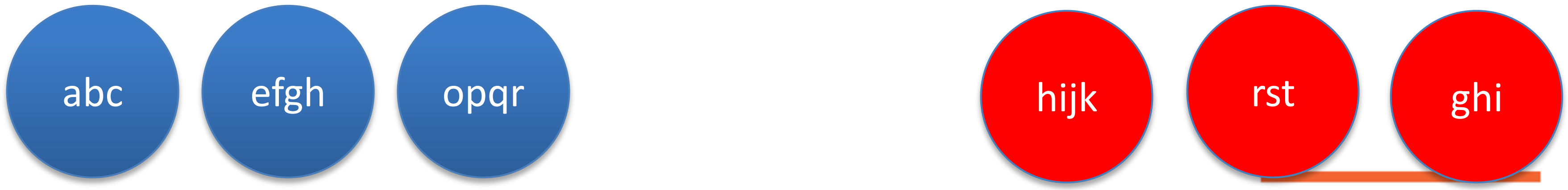
Given the attribute written in the middle of the circle……



There's ways to split them up, like length of the string



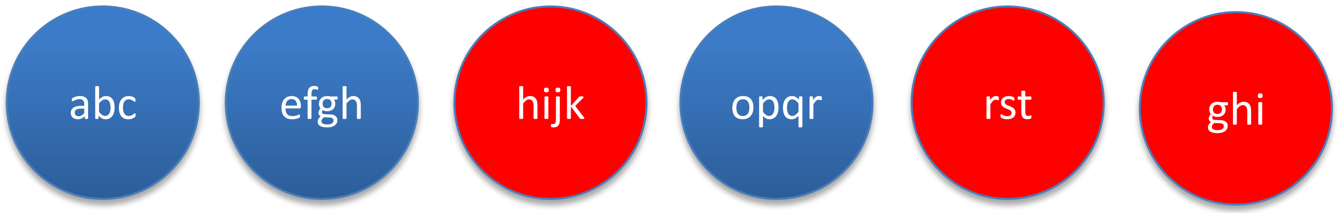
Or the nature of the first letter (consonant / vowel)



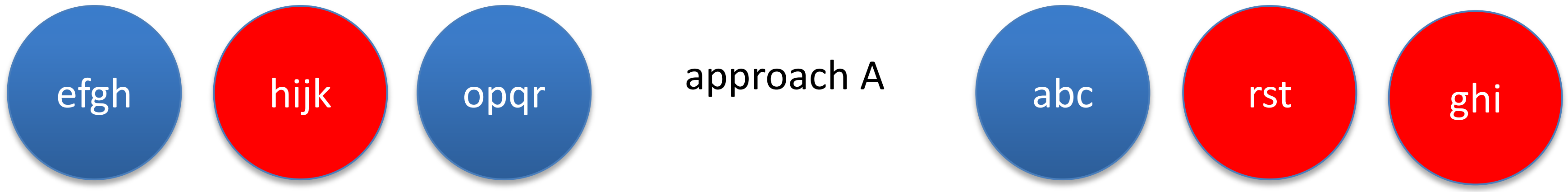
INTERSECT

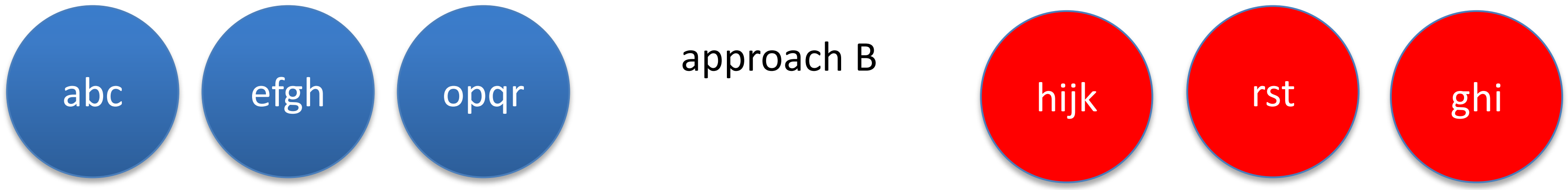
Decision Tree – what does model building look like

Start with all the instances



Consider all the ways of splitting the instance into a number of groups





Keep the one that reduces entropy the most, then do that recursively for each node.
This is called maximising the information gain at each step

Decision Tree – what does model-building look like

Stop when further splitting doesn't improve the entropy enough (or at all)

```
petalwidth <= 0.6: Iris-setosa (50.0)
petalwidth > 0.6
|   petalwidth <= 1.7
|   |   petallength <= 4.9: Iris-versicolor (48.0/1.0)
|   |   petallength > 4.9
|   |   |   petalwidth <= 1.5: Iris-virginica (3.0)
|   |   |   petalwidth > 1.5: Iris-versicolor (3.0/1.0)
|   petalwidth > 1.7: Iris-virginica (46.0/1.0)
```

http://en.wikipedia.org/wiki/List_of_important_publications_in_computer_science

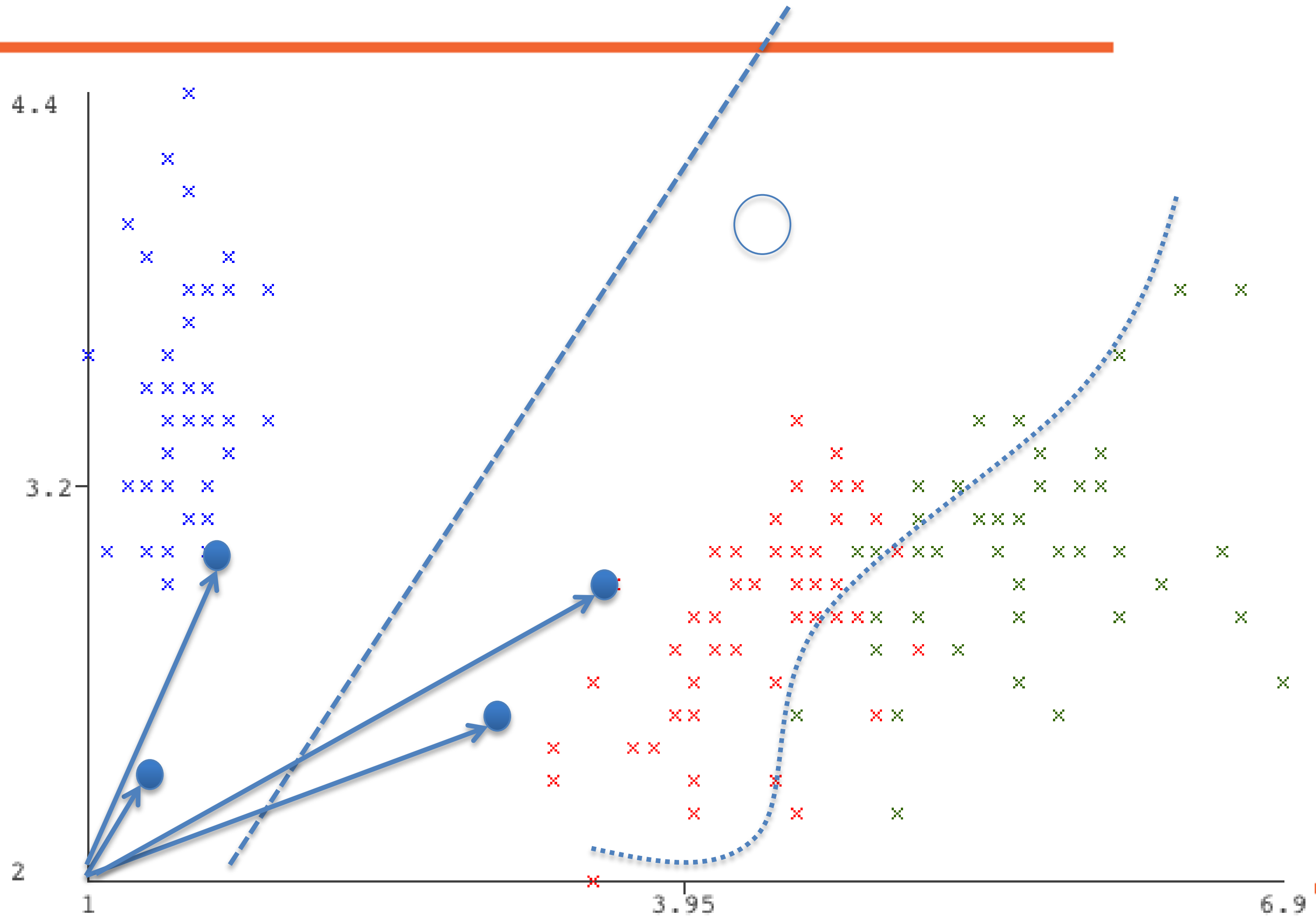
Exercise - Let’s try it J48

- Run the J-48 Decision Tree algorithm

- This time, use n-fold cross validation
  - I’ll explain it later.

http://en.wikipedia.org/wiki/List_of_important_publications_in_computer_science

Support Vector Machines



INTERSECT

Section Two. Evaluation

INTERSECT

Evaluation

- How do I know which learning algorithm to choose?
- You need to know what you need it to be good at
  - what are the characteristics of the situation where the learner and model will be situated?
  - what kinds of mistakes can we tolerate?
  - what kinds of correctness must we guarantee?
  - does training time matter?
  - does recall time matter?
  - does model size matter?
- How do you come up with these numbers?

Evaluation – Statistics are tricky, a cautionary tale

- Let’s imagine that HIV runs at 1% in the community
- Let’s imagine that we have a test for HIV that is 95% accurate
- Let’s imagine that I return a positive test to HIV.
- What is the probability that I have HIV?

INTERSECT

Let's build a confusion matrix

	True Positive	True Negative	
Test Positive	190000	990000	**1180000**
Test Negative	10000	18810000	**18820000**
	200000	**19800000**	**20000000**

Accuracy	0.95
Prevalence	0.01

What are my probabilities?	0.161016949	**0.838983051**

INTERSECT

Evaluation – Statistics are tricky, a cautionary tale

- James and John are trivia buffs from rival towns, and very competitive, they want to find out who's the best, so they'll add up all the points they get this year.
- They mostly play in their own town, but sometimes play at each-others' pub
- All pubs play games that have a maximum value of 100 points
- At the end of the competition, James had a better average score at both towns' competitions

- Who had the best overall average score?

We don’t know, but it could have been John

TOWN A	James	John
Average Score	98	90
Total Games	5	100

TOWN B	James	John
Average Score	68	60
Total Games	100	5

Combined	James	John
Town A Points	490	9000
Town B Points	6800	300
Total Games	105	105
Average	69	89

INTERSECT

Fundamental Performance Measures

- Confusion Matrix
  - how often did the algorithm get confused between a pair of classes classes?
  - many other measurements can be derived
    - Accuracy (!)
    - Sensitivity (a.k.a True Positive Rate for binary problems)
    - Specificity (a.k.a 1 – False Positive Rate for binary problems)
    - ROC (receiver operating curve)

- Sensitivity and Specificity are very important in medical diagnosis, and correspond to our intuitions of how accuracy "should behave".

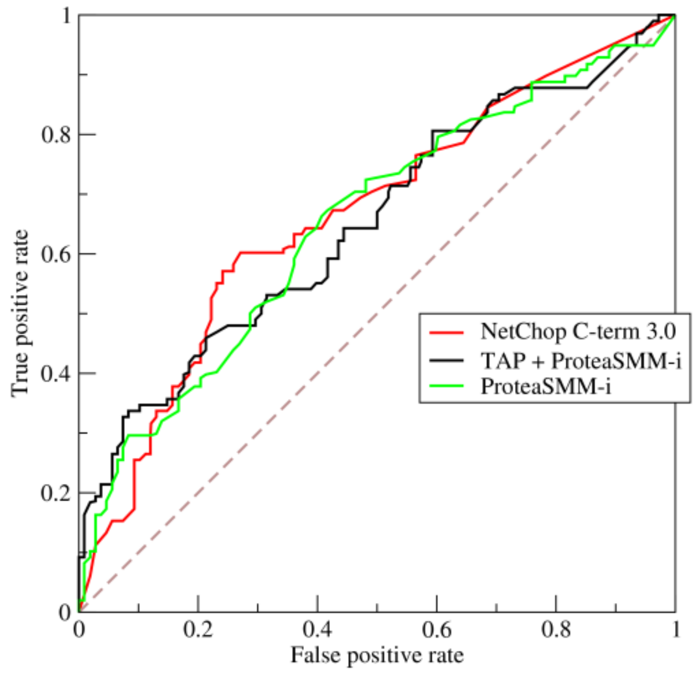
Confusion Matrix

```
  a  b  c   <-- classified as
 50  0  0 |  a = Iris-setosa
  0 46  4 |  b = Iris-versicolor
  0  7 43 |  c = Iris-virginica
```

INTERSECT

ROC Curves

- Intuitively, you can imagine that True positive rate and False positive rate trade off against each other
- That is, you can 'fudge' a high TPR by always returning TRUE
- That is, you can 'fudge' a low FPR by never returning TRUE



INTERSECT

Exercise – Run a learner now, and we can go through the output

- Select a decision tree, and 10-fold evaulation
- I’ll step you through what is happening
- Look at the confusion matrix (what’s it based on? that comes later)
- Look at the ROC curve (right-click on the result entry in the list, select “Threshold Curve”
  - I’m not going to go into great detail

INTERSECT

Some other performance measures

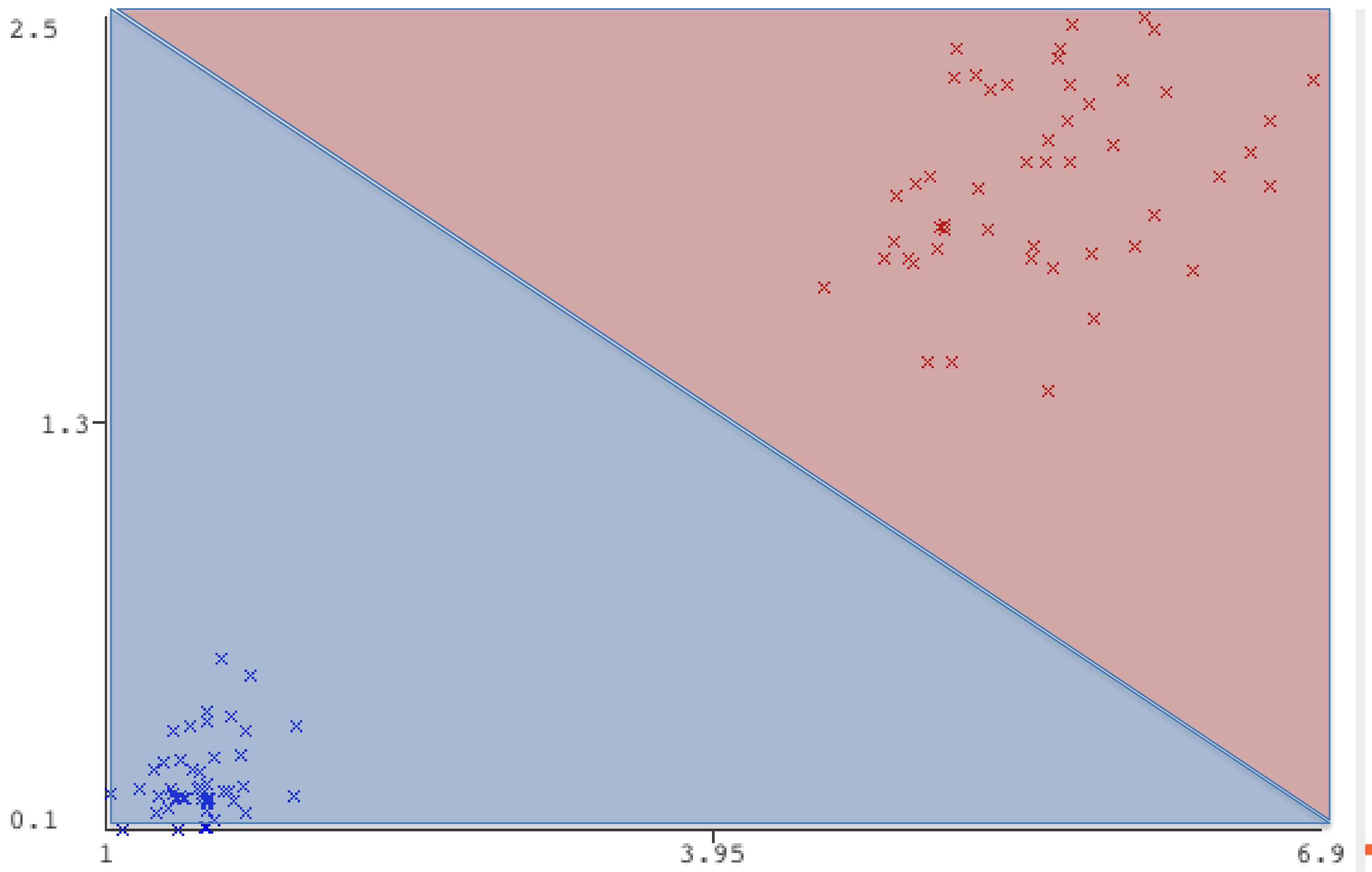
- How scrutable is your model?
- How long does training take?
- How long does recall take? (Average, Best, Worst)
- How big is the model (IB-K)?
- Can the model be incrementally updated?
- Can the model be compiled or must it be interpreted?
- Can the model degrade gracefully?
- …

INTERSECT

Evaluation – that was measurement, what about experiment design?

INTERSECT

Evaluation – before you even think about evaluation, think about sampling



INTERSECT

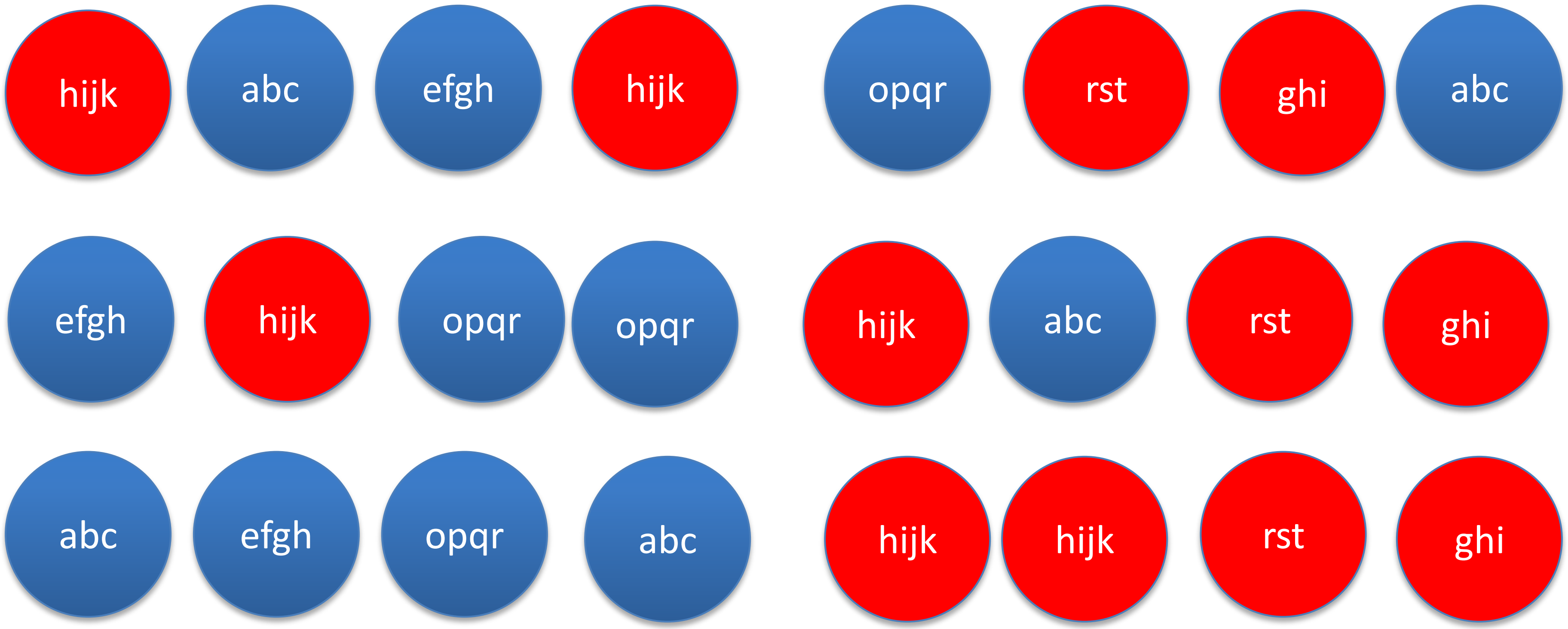
A few constraints

- Data is often scarce
- Data is often expensive
- The amount of data required grows quickly as dimensionality increases

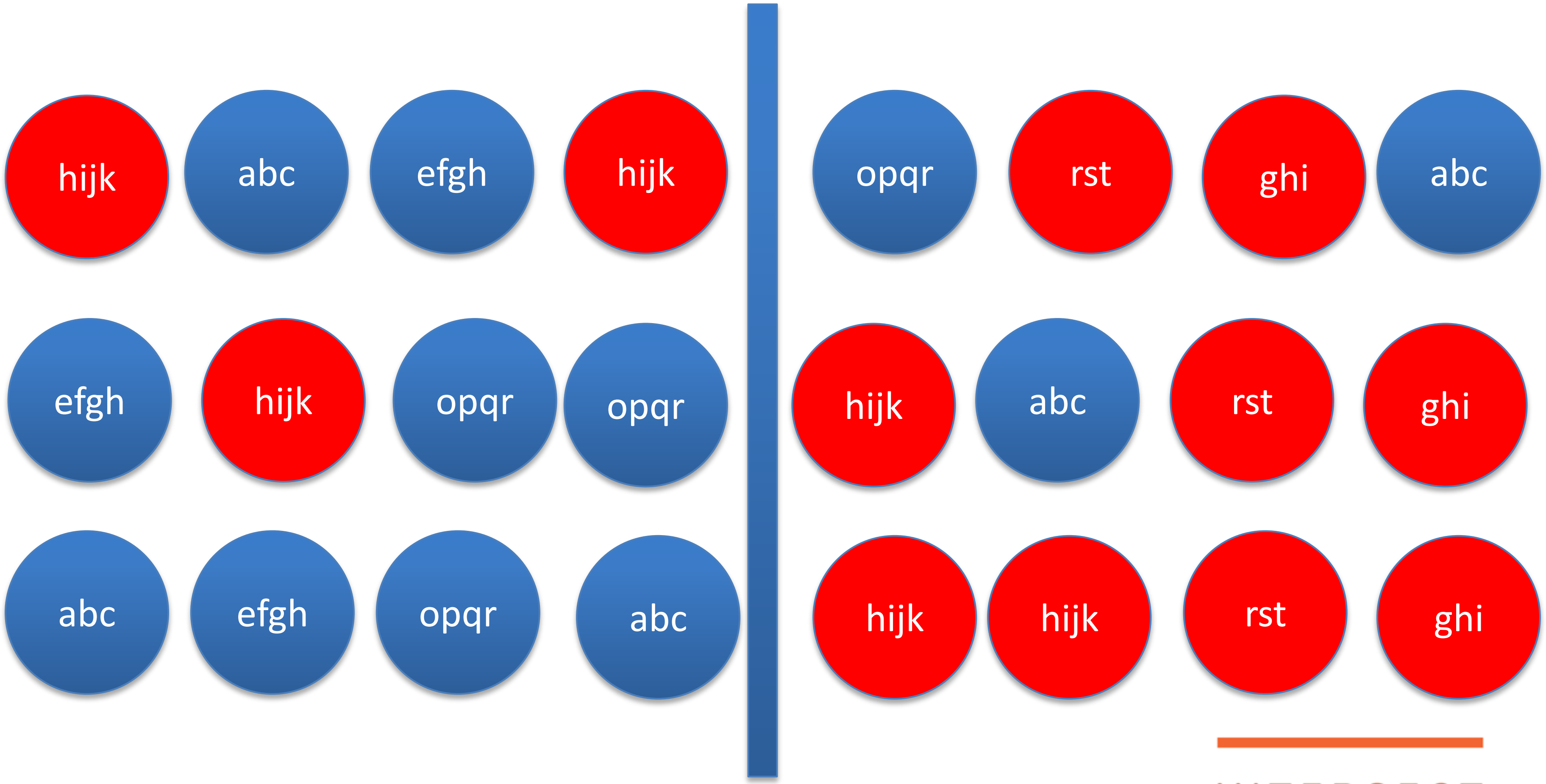
- How do we
  - make the best possible model from the data available
  - measure how good it is

A basic approach – train and test

Use the whole dataset as the training set, and then test on that set, too.

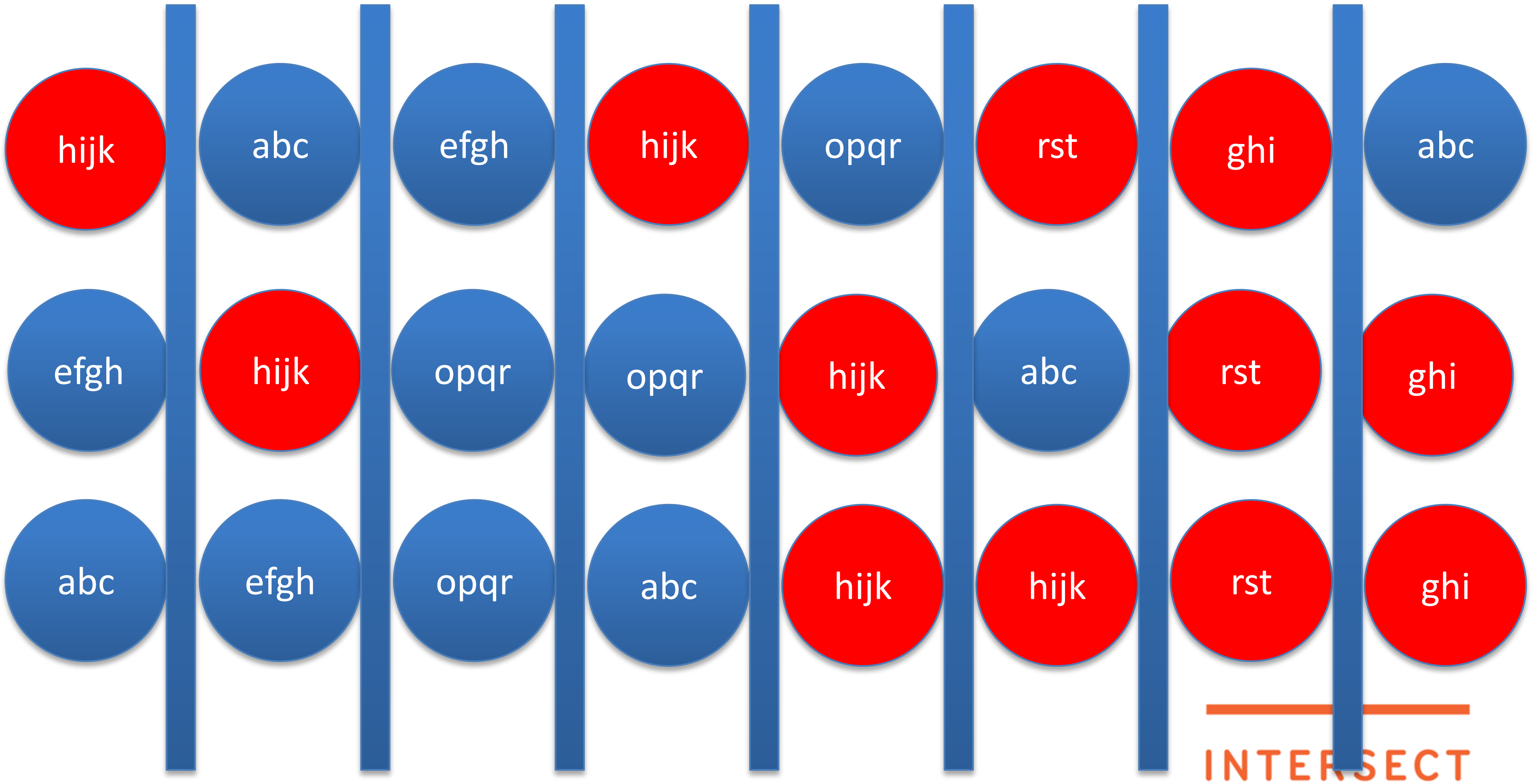


A slightly better approach – train and test



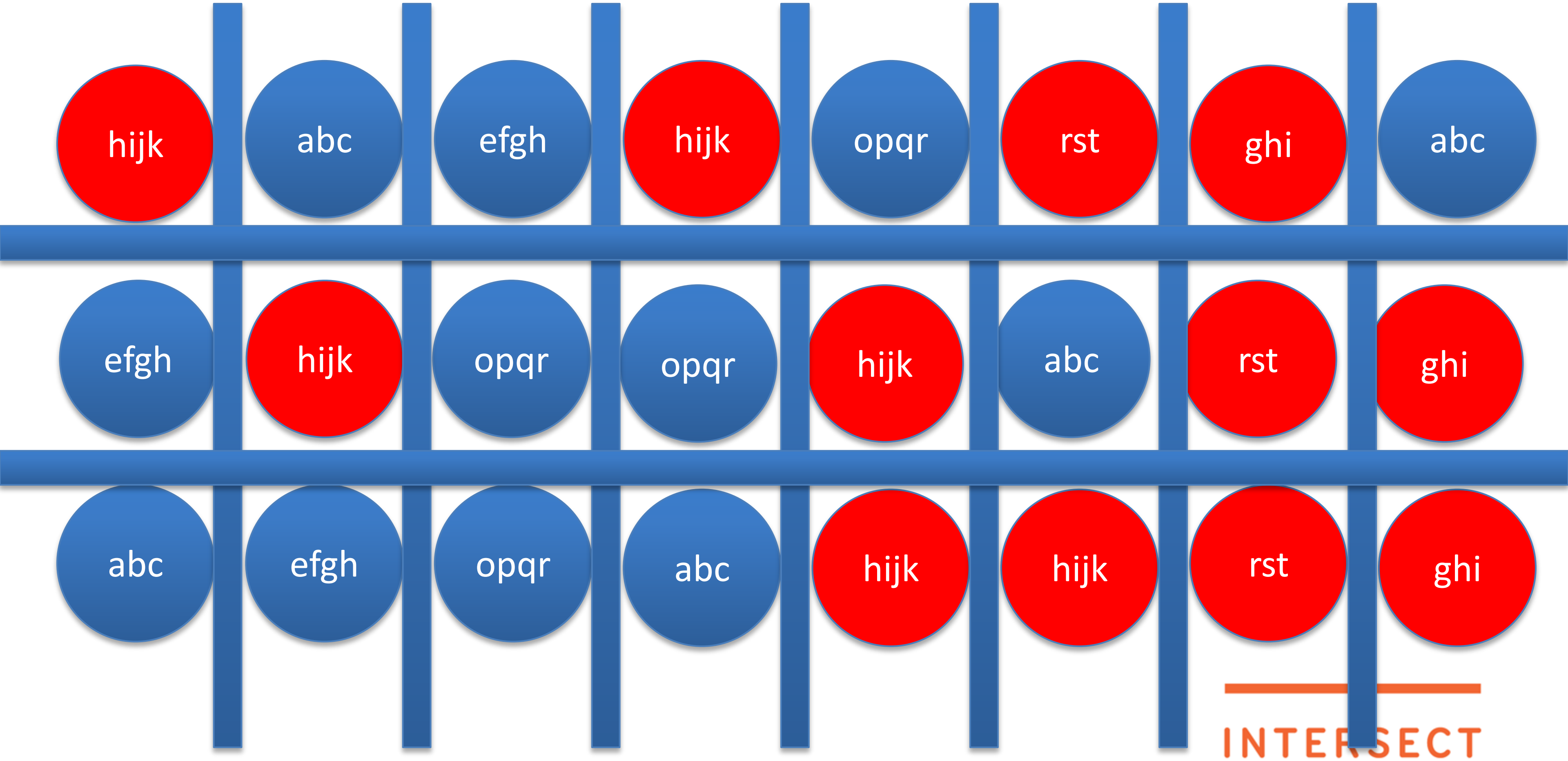
INTERSECT

The ‘standard’ approach – n-fold cross validation



INTERSECT

The best, impossible, approach – leaving one out



INTERSECT

So, in summary

- If you want to know how well your model will work in the "real world" you have to
  - define what "well" means
  - work out a way to capture sufficient data to feed your algorithm
  - work out a way to ensure that the data represents the "real world"
  - work out an experiment design that preserves that representation and accurately evaluates your learning algorithm
  - be aware of the nature of your algorithms and what impact that has on your
    - sampling,
    - experimental design
    - metric choice

INTERSECT

Final case study (and the end of the main body)

- Let’s reconsider IB1 and why it got such good numbers?

Advanced Topics

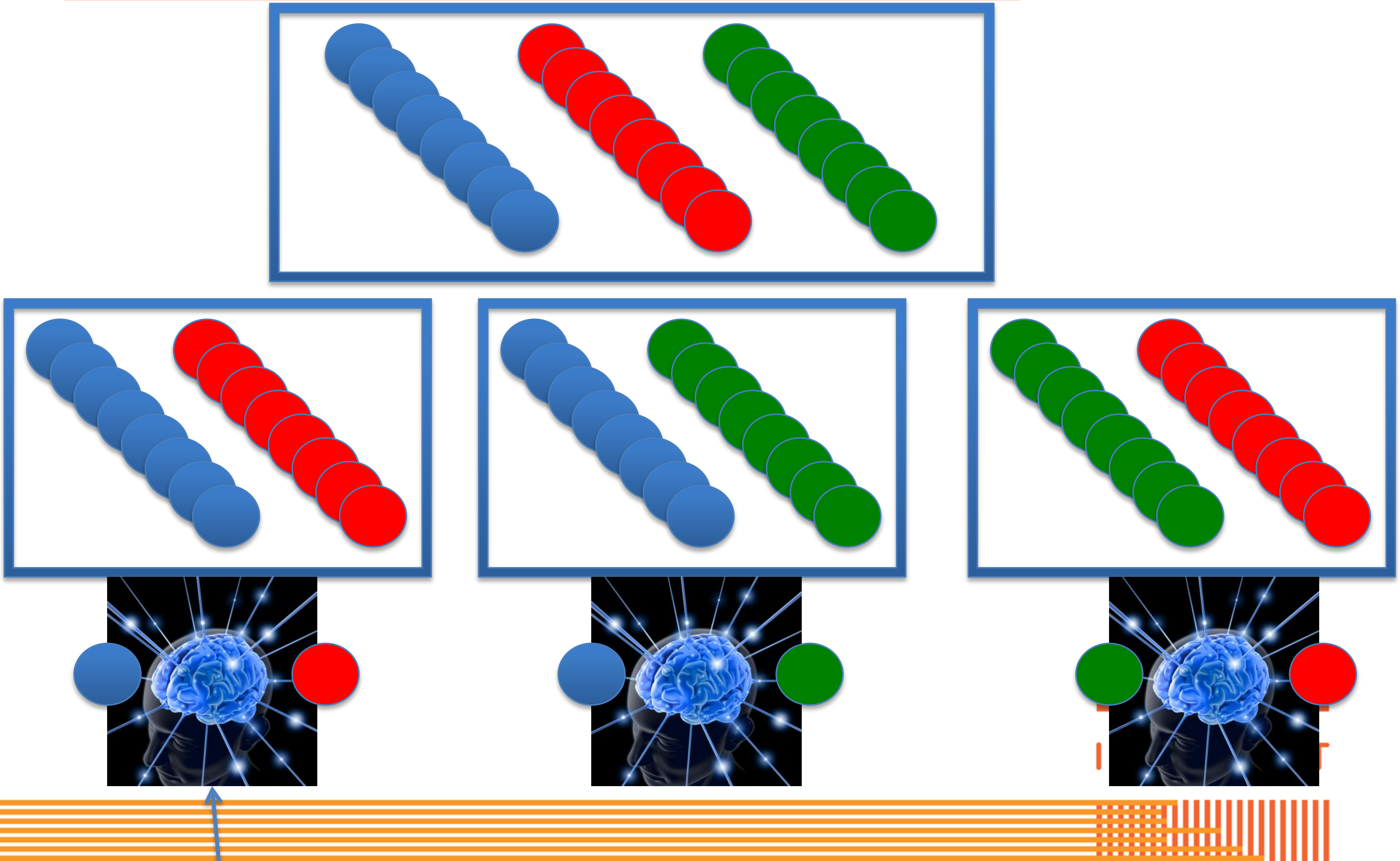
- Extending 2-class algorithms to multiclass algorithms
- The kernel trick
- Data representation and the curse of dimensionality

INTERSECT

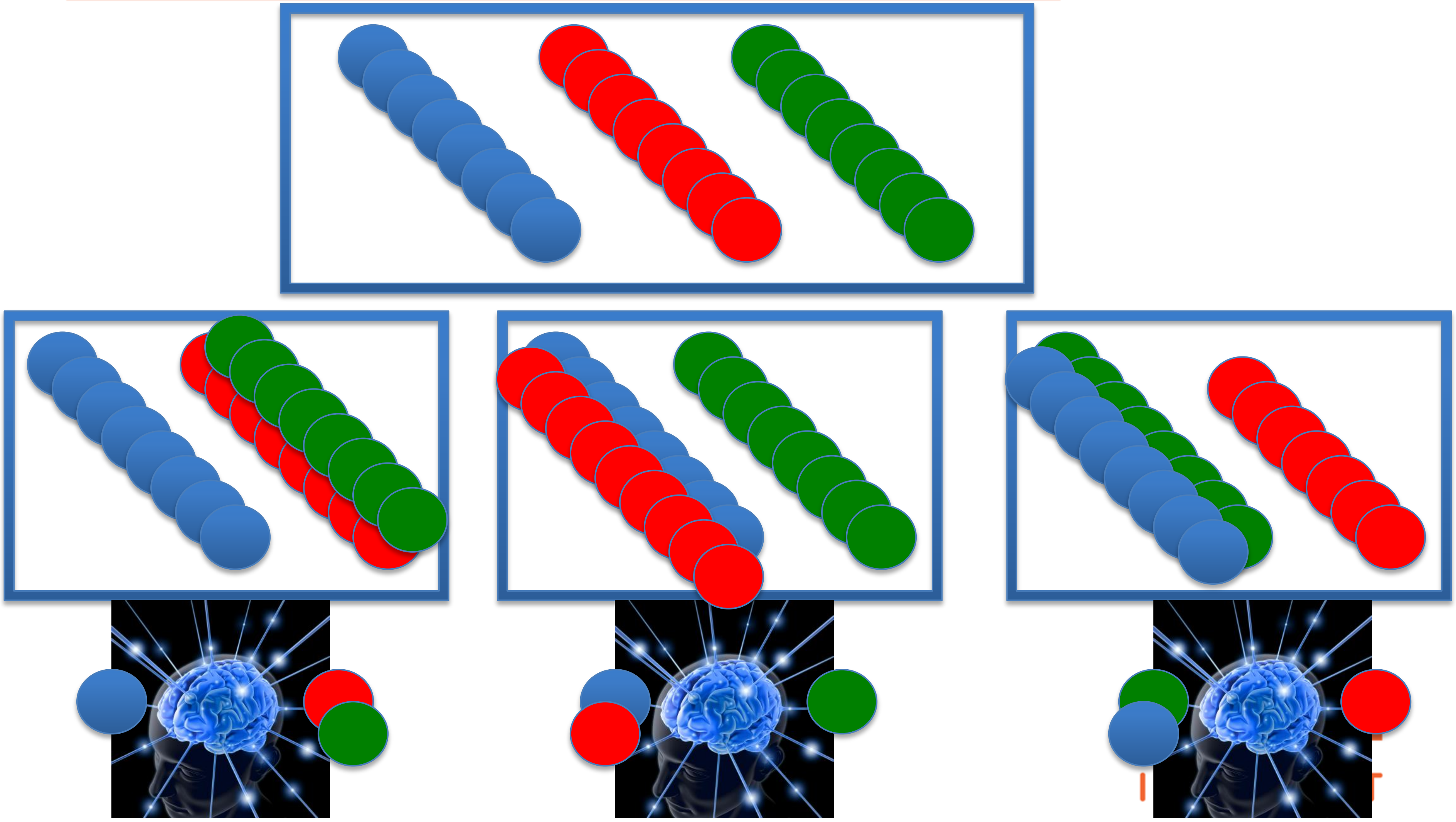
Multiclass Algorithms

- Imagine that I had a learning algorithm that could only generate 2-class decisions
  - e.g. SVM
  - not, e.g. Decision Tree
- There are two approaches that are ‘standard’ to boost these to multiclass learners
  - 1 vs 1
  - 1 vs rest
- The algorithms have two parts: binarisation and debinarisation

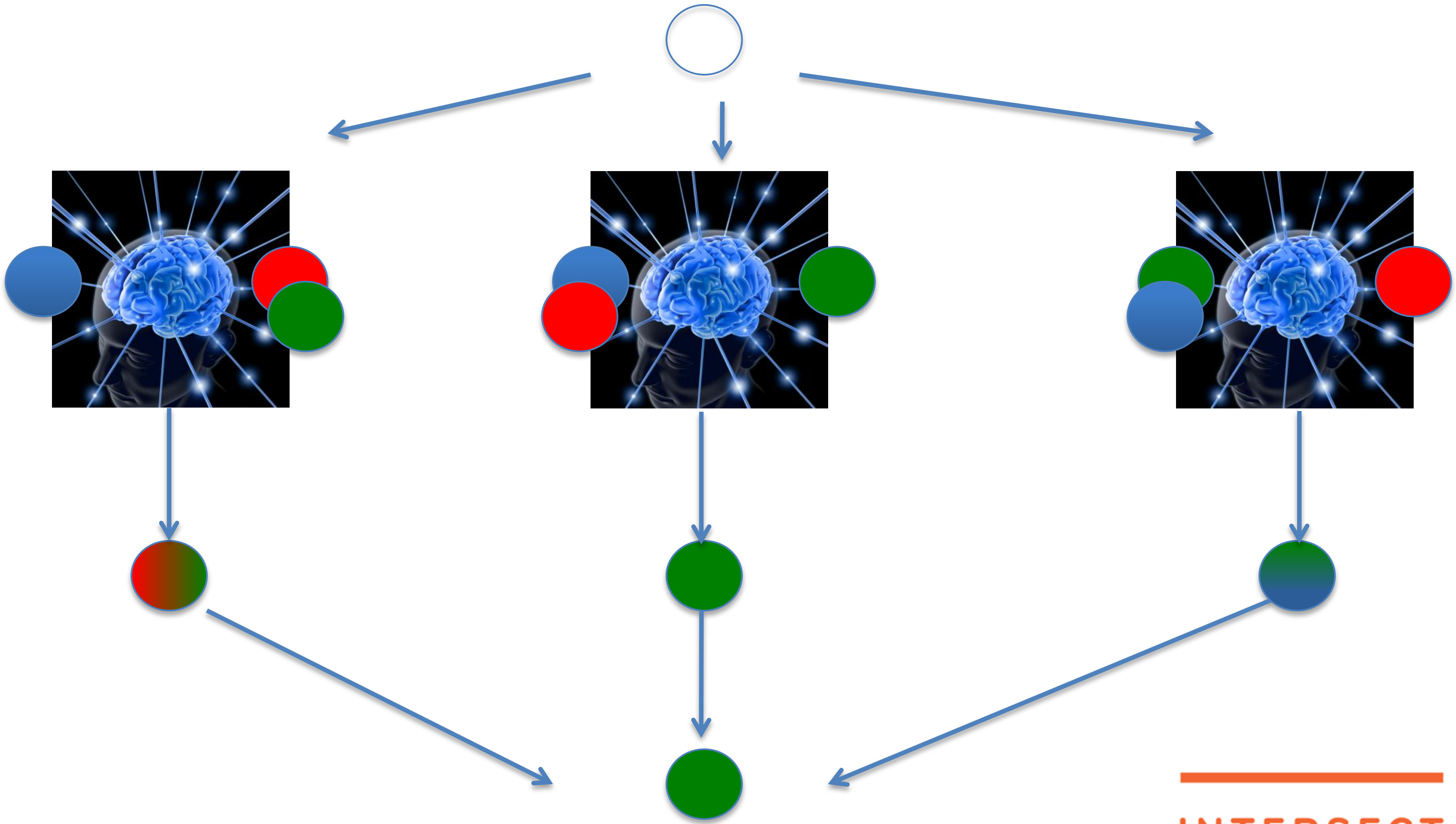
Binarisation – happens at training time



Binarisation – another approach – one-vs-rest



Debinarisation – happens at recall time



INTERSECT

Distribution Learning

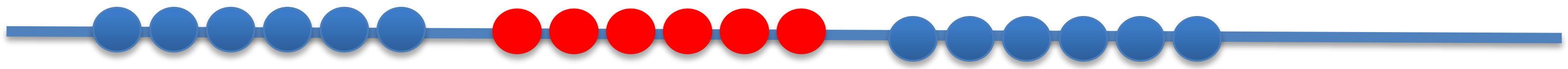
- Some models can give more than a vote

```
petalwidth <= 0.6: Iris-setosa (50.0)
petalwidth > 0.6
|   petalwidth <= 1.7
|   |   petallength <= 4.9: Iris-versicolor (48.0/1.0)
|   |   petallength > 4.9
|   |   |   petalwidth <= 1.5: Iris-virginica (3.0)
|   |   |   petalwidth > 1.5: Iris-versicolor (3.0/1.0)
|   petalwidth > 1.7: Iris-virginica (46.0/1.0)
```

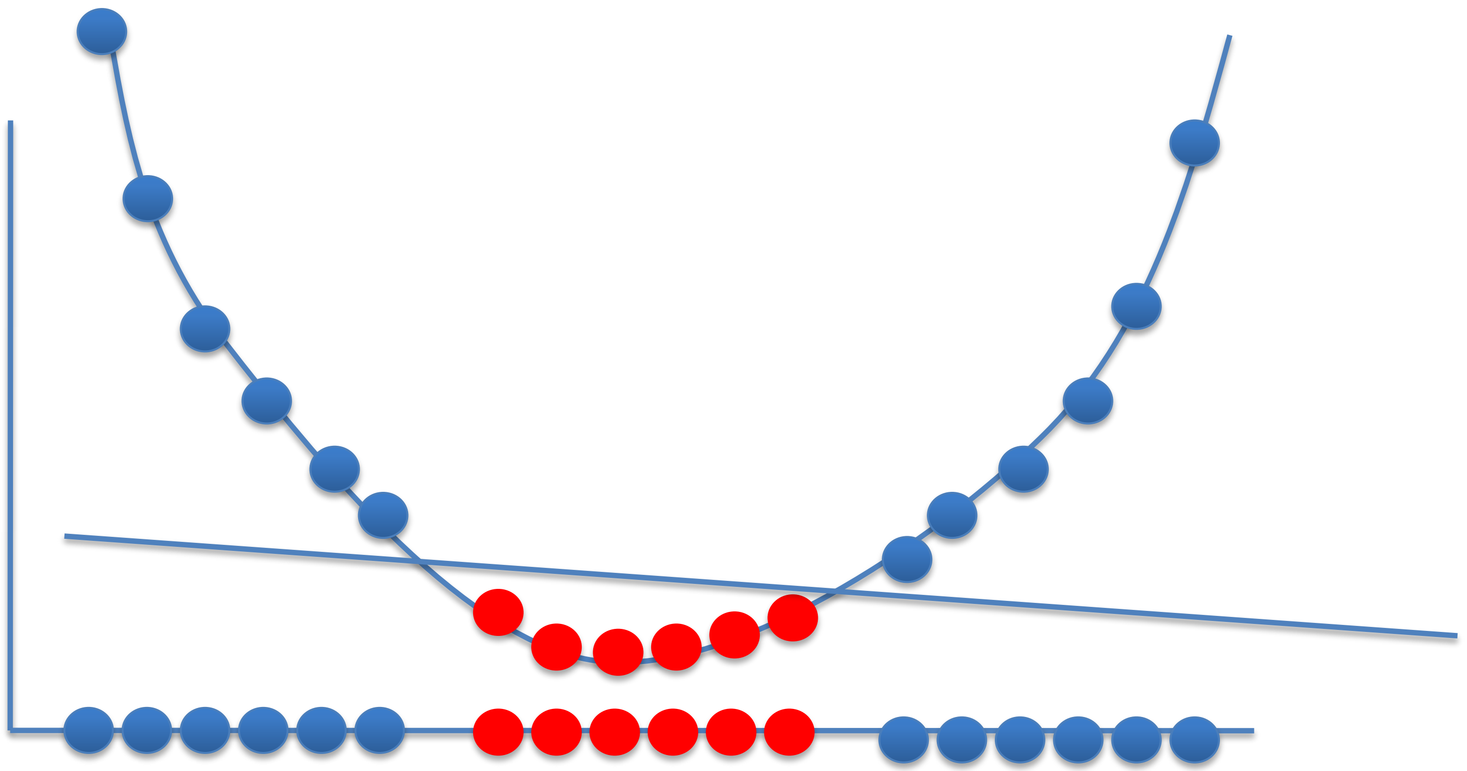
This makes it easier to combine them when debinarising

The kernel trick

Imagine you may draw only one line to separate these two classes. Where would you put it.



Adding a dimension helps a lot



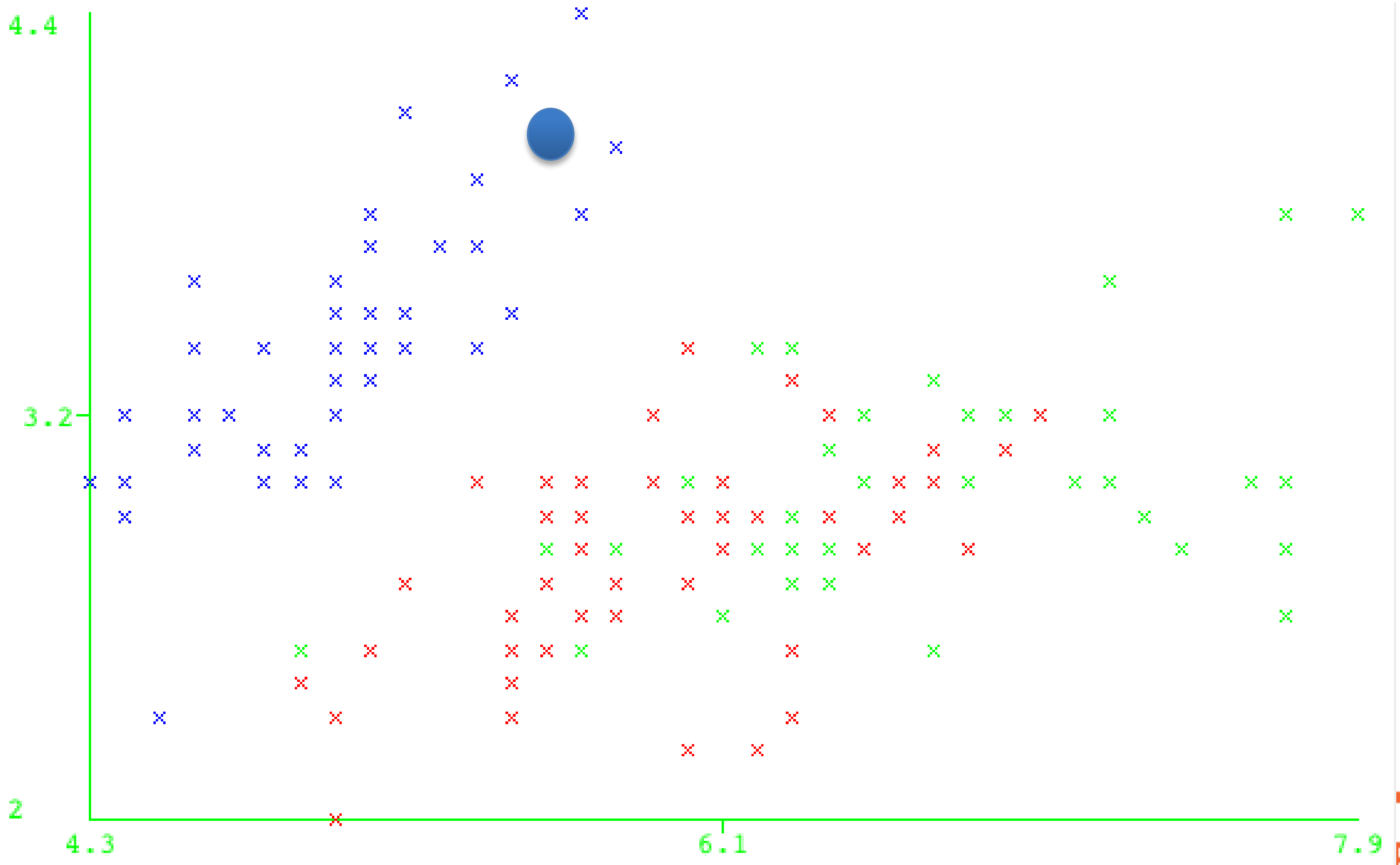
INTERSECT

The kernel trick

- The SVM algorithm works by comparing instances ONLY by computing their inner product (dot product)

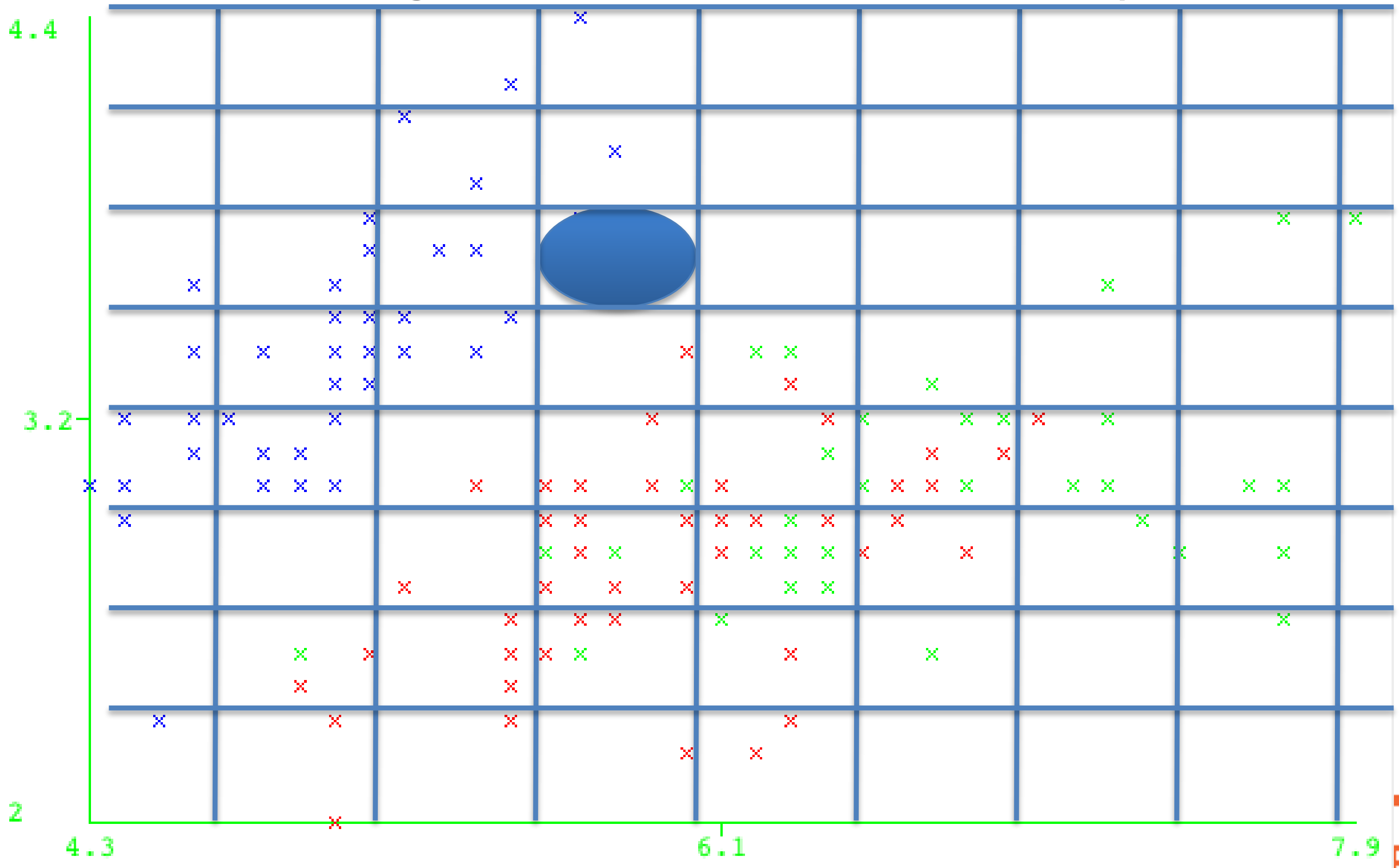
Dimensionality and Indexing

- Consider indexing in the abstract – and some analogy



Dimensionality and Indexing

- Consider indexing in the abstract – and it's a fairly loose analogy,



Thanks for attending!

- Please complete our **course survey** at:
  - http://svy.mk/18c8dHa
- Any **further questions**, contact us at
  - training@intersect.org.au
- Find out about **upcoming courses** by signing up to our mailing list
  - http://bit.ly/1aZvRqw

INTERSECT