ual: creative computing institute

Classification.
Introduction to Machine Learning

Marysia Tańska

Based on materials by prof. Rebecca Fiebrink

Peckham DAZ, 12th September 2024

What is Machine Learning?

Learning from data

There are many Machine Learning algorithms, methods and applications

Today, we will be working with an area of Machine Learning called

“supervised learning”. It means that data is labelled.

Examples of labelled datasets:
https://www.kaggle.com/datasets/kazanova/sentiment140

https://www.kaggle.com/c/fake-news/data

There are many Machine Learning algorithms, methods and applications

- Classification – for assigning objects into categories. We will explore this topic further today.
- Regression – for predicting future data based on historical data.

Classification or regression?

- Weather forecast

Classification or regression?

- Weather forecast - regression

Classification or regression?

- Flagging fake news

Classification or regression?

- Flagging fake news - classification

Classification or regression?

- Recognising emotions in social media posts

Classification or regression?

- Recognising emotions in social media posts - classification

Classification or regression?

- Predicting the GBP/EUR rates for the next week

Classification or regression?

- Predicting the GBP/EUR rates for the next week - regression

What is classification?

Example Questions

- Is this tweet toxic or not?
- Is this email spam or not?
- Does this product review have positive, negative, or neutral sentiment?
- Is this book written for young children, older children, or adults?

Every document belongs to 1 (and only 1) pre-defined category (also called a "class")

A classifier makes a decision

You could build your own decision function.

```
def labelSentiment(numHappy, numSad):
    if (numHappy > numSad):
        return 2
    elif (numHappy < numSad):
        return 1
    else:
        return 0
```

Note that a “class” is usually represented as an integer, e.g., 0, 1, 2, etc.

Designing this by hand will often leave you with something that doesn’t work well.

A classifier makes a decision

Or, a machine learning algorithm can look at examples of different classes and provide a “good” function for you.

```
def labelSentiment(numHappy, numSad):
    ?????
```

How can we “learn” a good function?

Training data:
Represented as a table/DataFrame

Each “training example” is a document represented as a vector of numbers (e.g., BoW) plus a label of the correct class

	# “happy”	# “sad”	Label
Doc 1	25	2	2
Doc 2	30	34	1
Doc 3	14	13	0

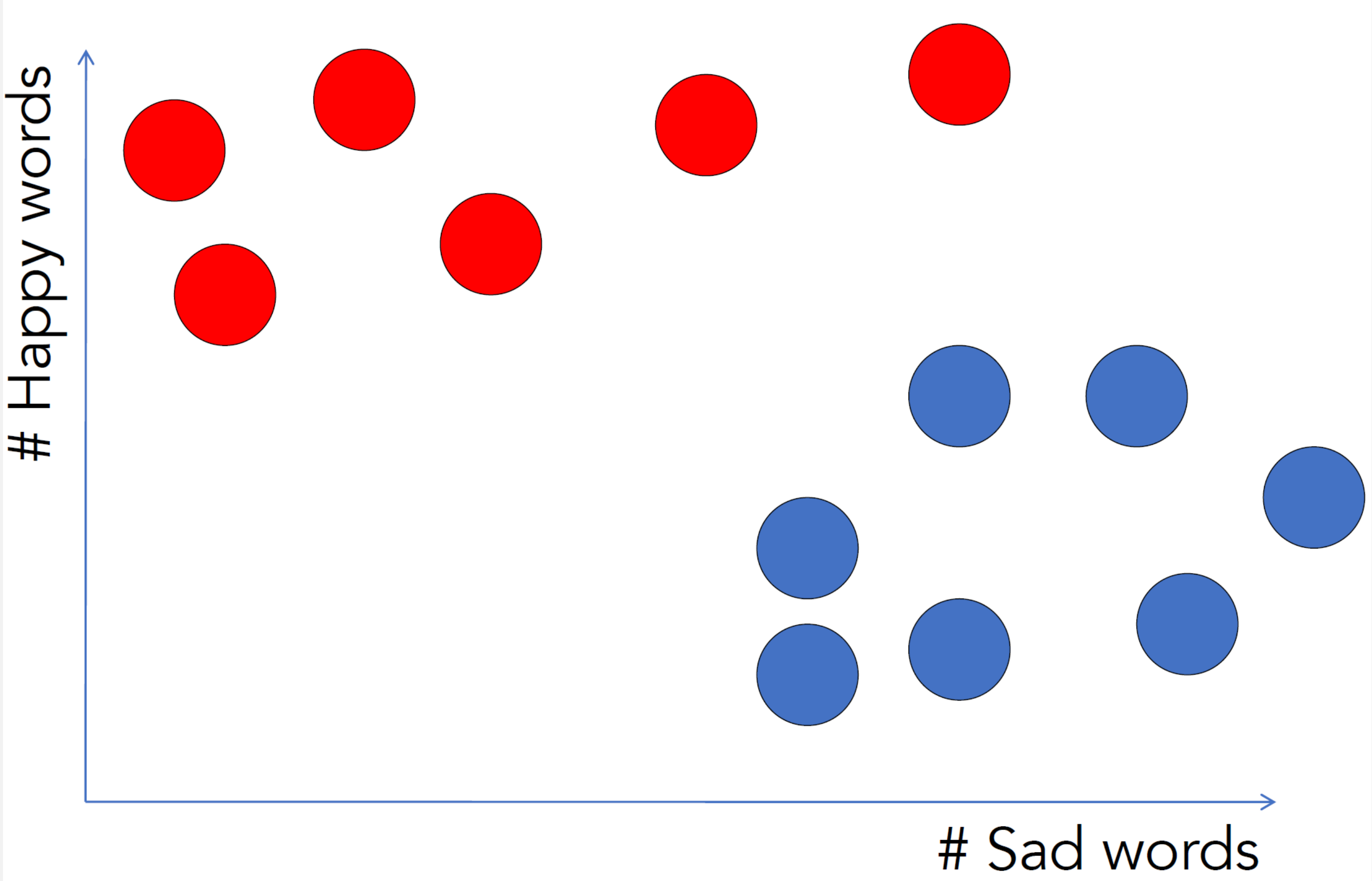
Training data: Represented as a table/DataFrame

Each “training example” is a document represented as a vector of numbers (e.g., BoW) plus a label of the correct class

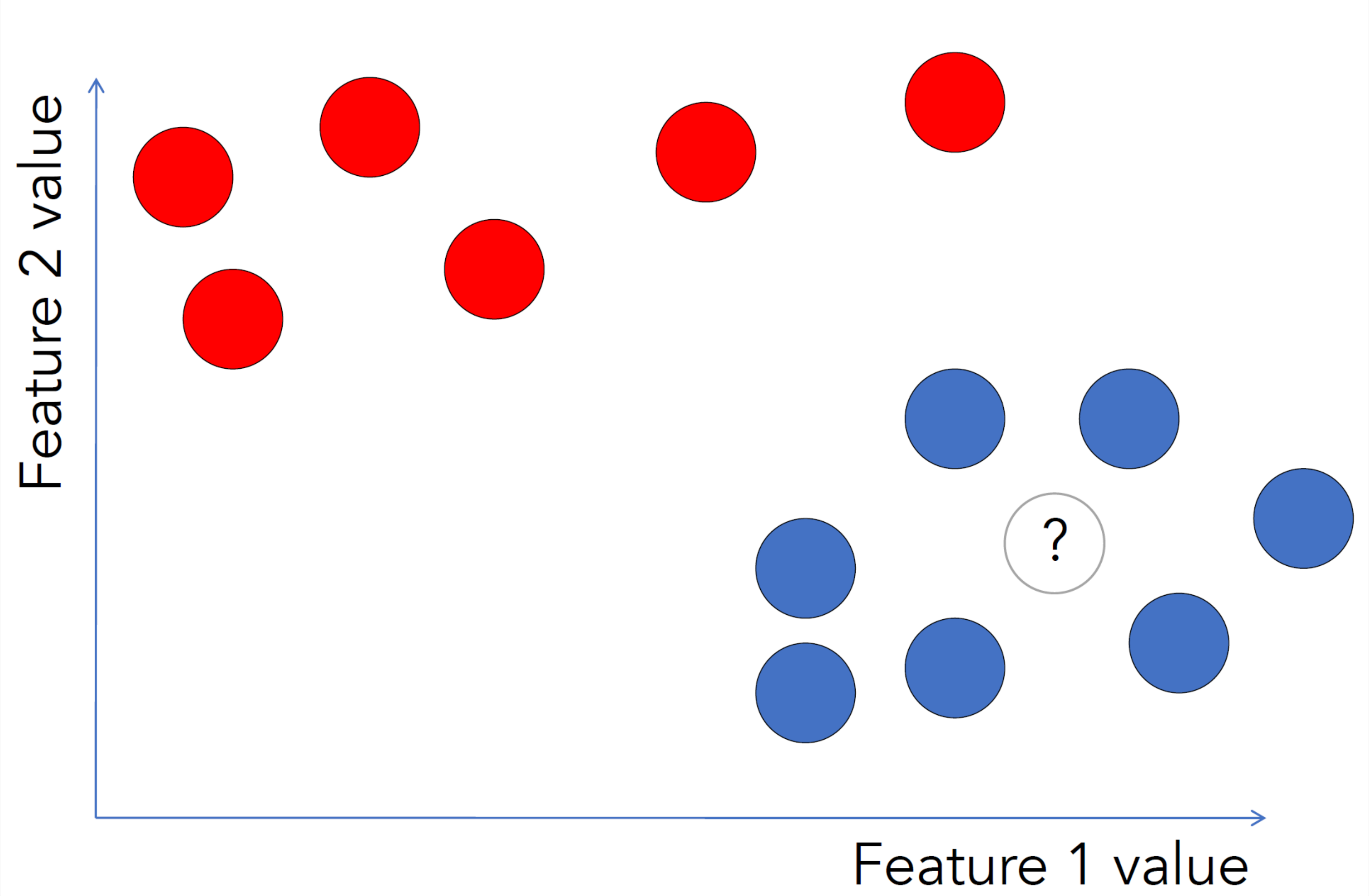
	# “happy”	# “sad”	Label
Doc 1	25	2	2
Doc 2	30	34	1
Doc 3	14	13	0

“Supervised learning”: This label column tells the computer how we want to label the data. More info: https://www.ibm.com/think/topics/supervised-vs-unsupervised-learning

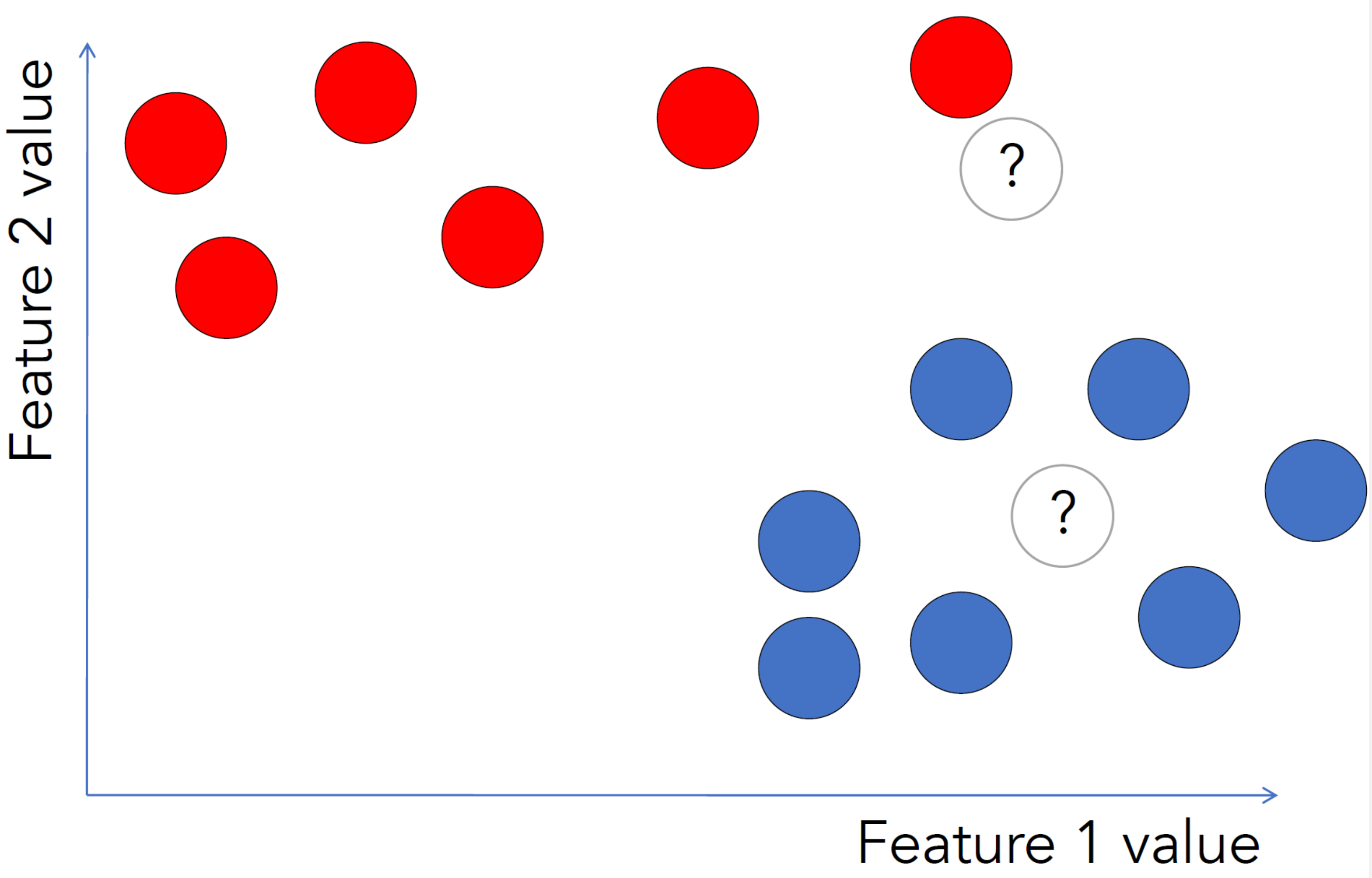
You can visualise each training example as a point in the "feature space":



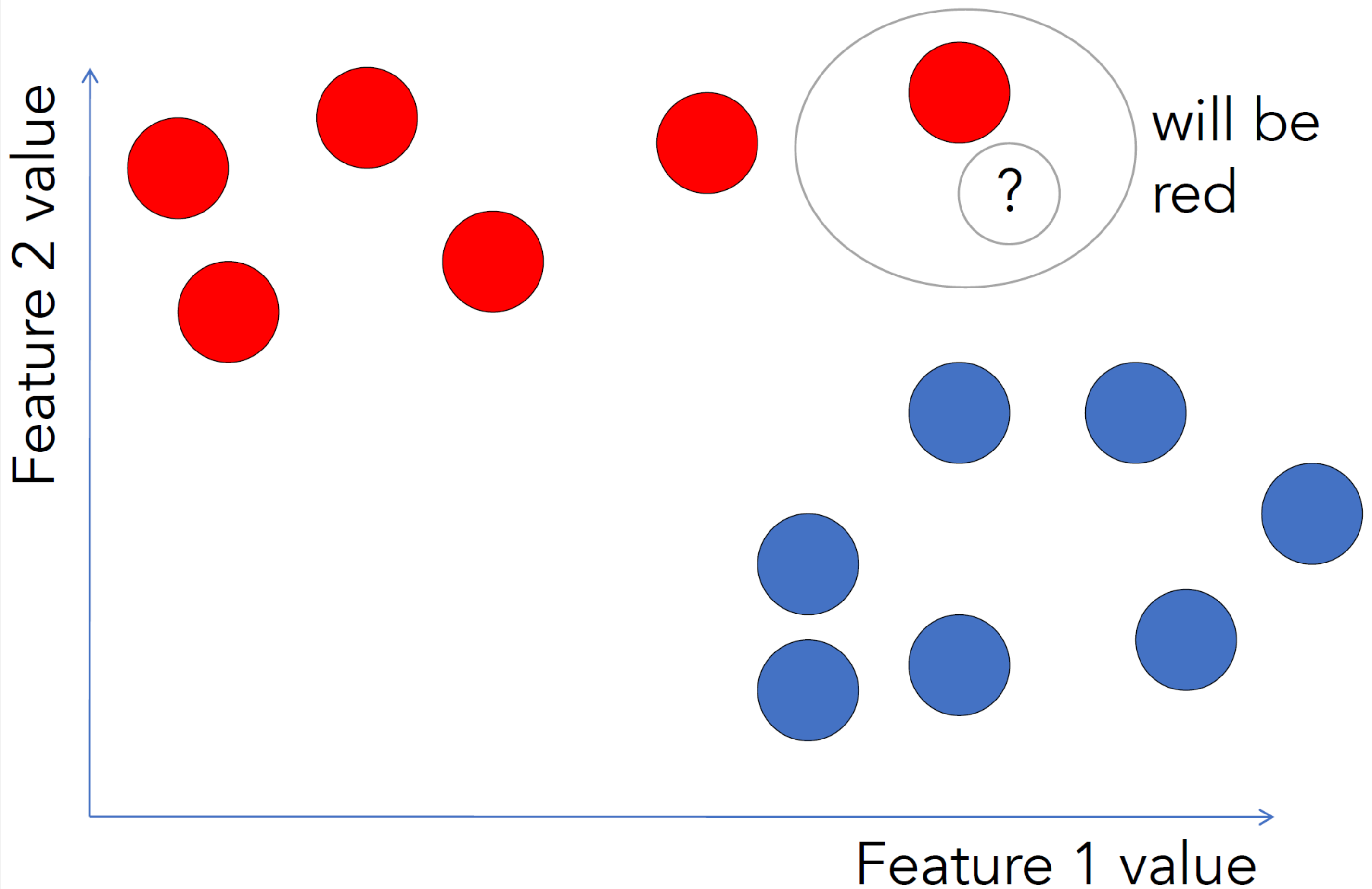
The job of a classifier is to assign the right label to a new point:



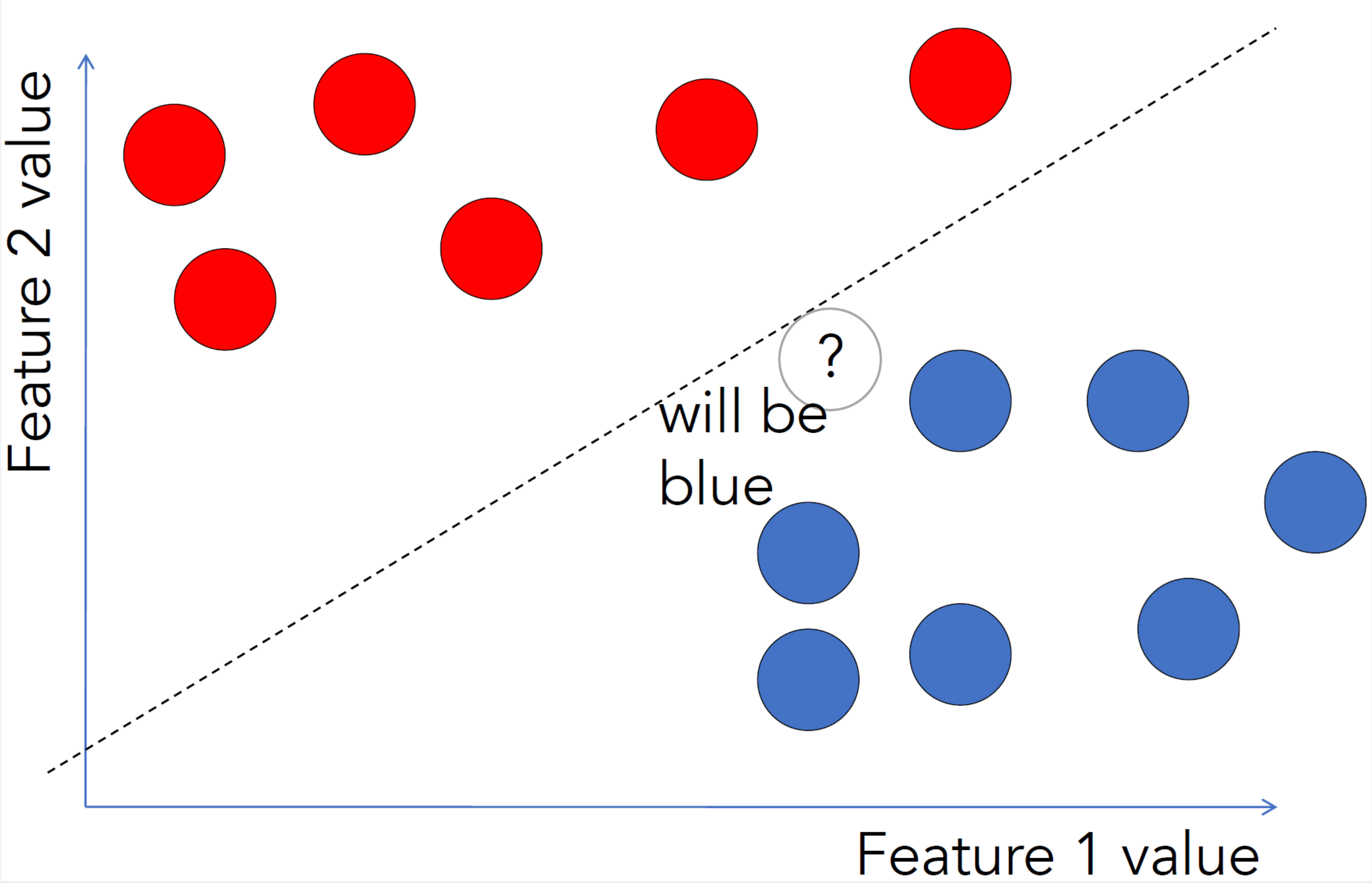
How would you decide?



1-nearest-neighbour classifier:
Classify new point to be same as closest point



Another strategy: Draw a line separating the classes (“Decision boundary”)



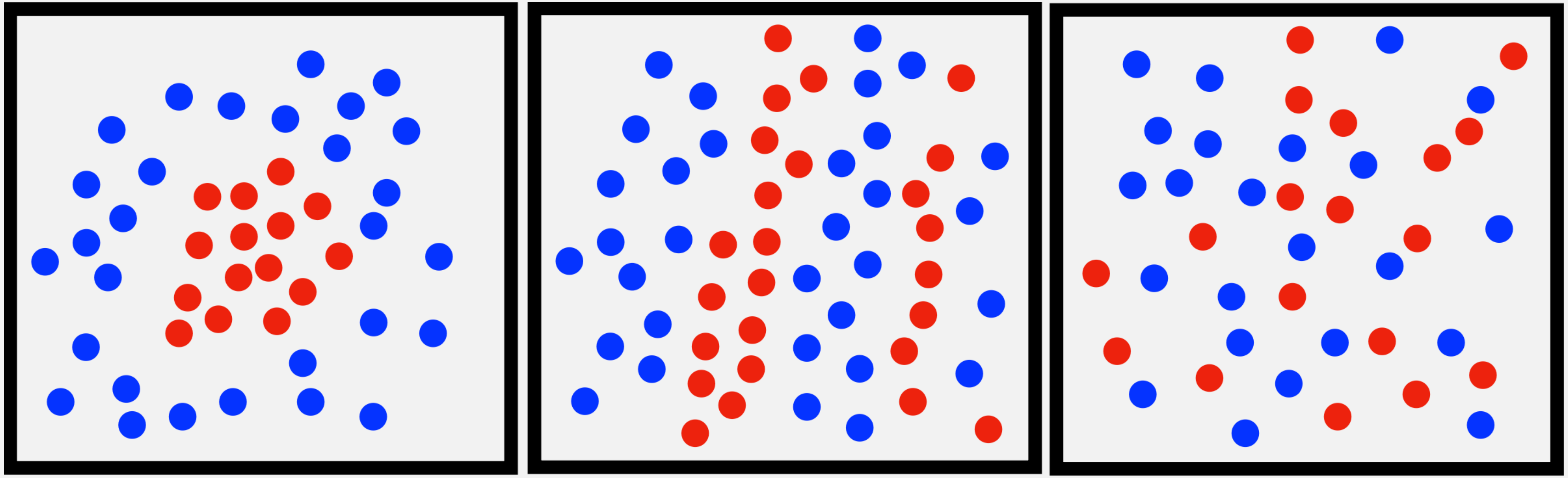
Any alternative ideas?

Recap: How to perform classification

1. Gather a training dataset.

- Represent each document/item as a vector of numbers - for example: [5, 0, 1]
    - Ideally these provide enough information to “discriminate” (tell the difference amongst) different classes.

Different feature choices might make our classification problem easier or harder!



Recap: How to perform classification

1. Gather a training dataset.

- Represent each document/item as a vector of numbers - for example: [5, 0, 1]
- Ideally, these provide enough information to “discriminate” (tell the difference amongst) different classes.
- Also store the “class” label for each document, typically as an integer (0, 1, 2, etc.) This template has bespoke bullet points, the bullet levels can be toggled between using the list level buttons, or using the tab key at the start of the paragraph.

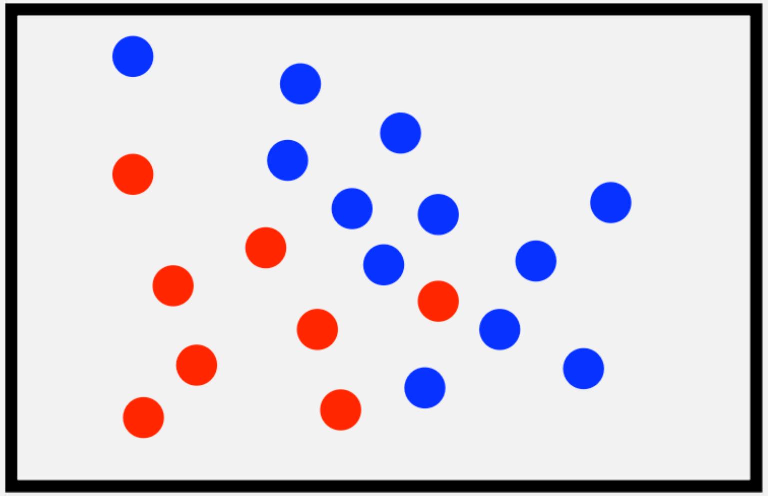
2. Give this dataset to a supervised learning algorithm

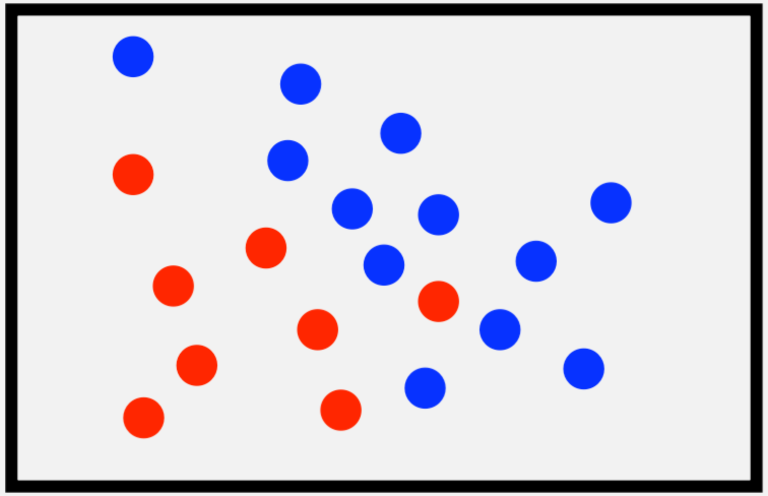
- “Train” this algorithm on the dataset to build a classifier (also called a “model”)
- This is your decision function (but you’ll rarely look at the function itself)

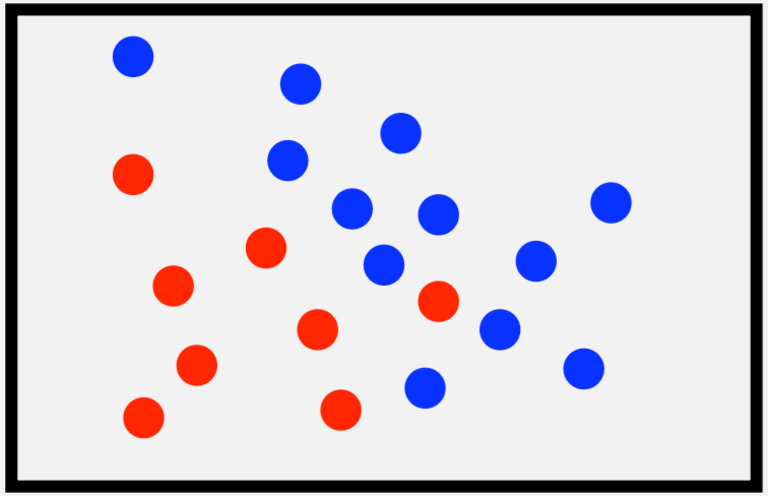
3. Use the classifier (decision function) to classify new data:

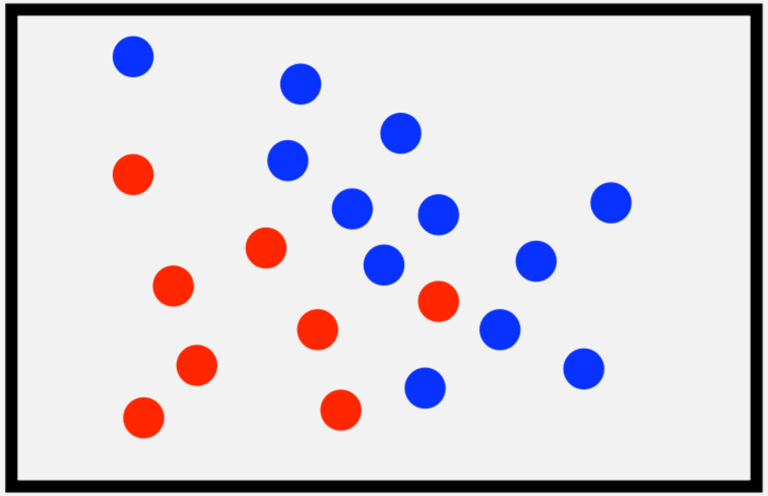
- Represent a new document as a vector
- Pass this vector to the classifier to get a label back

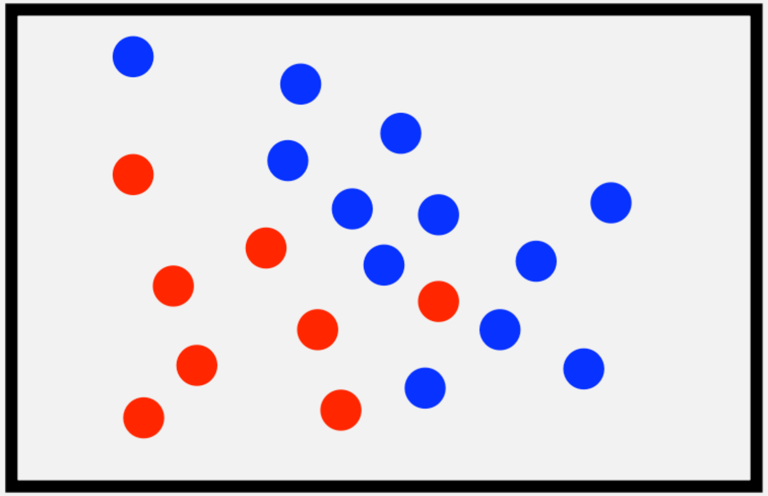
What makes a good classifier?

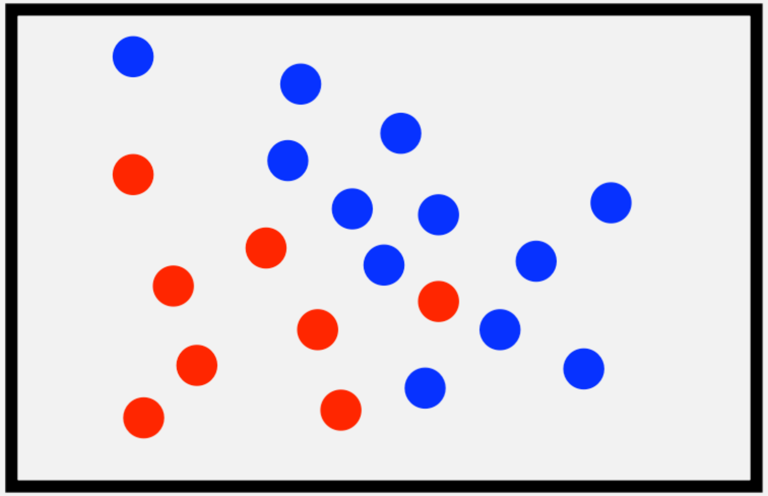


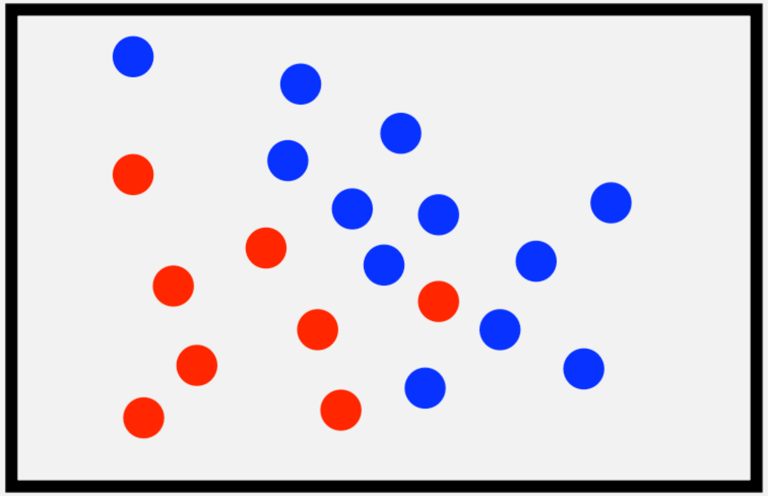


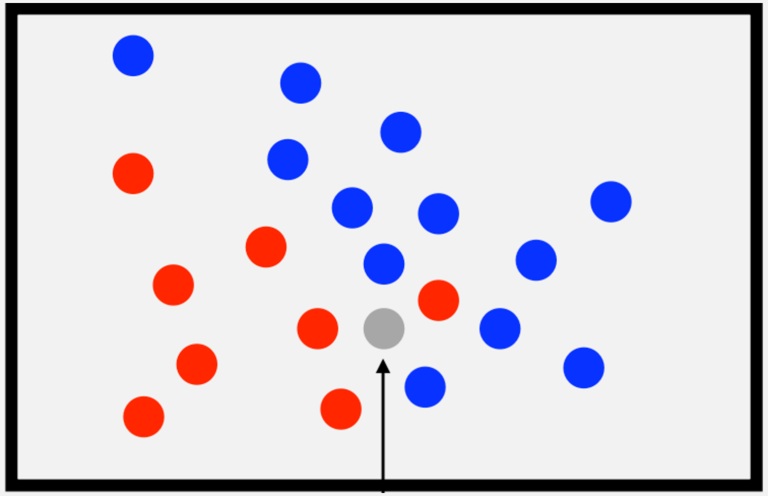




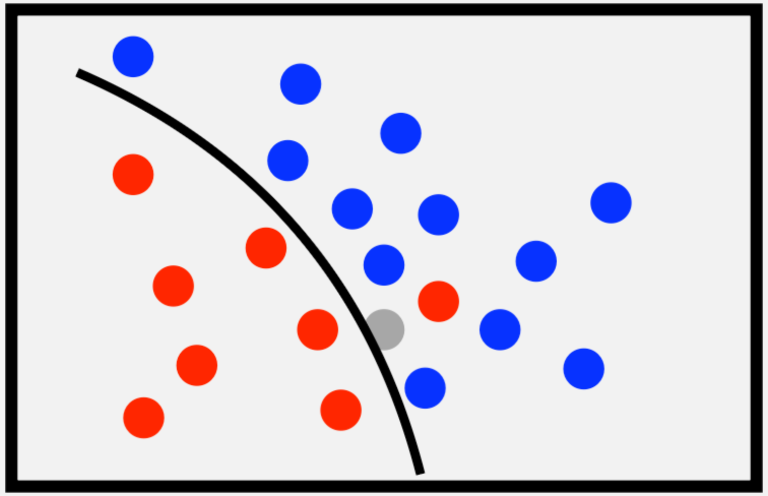


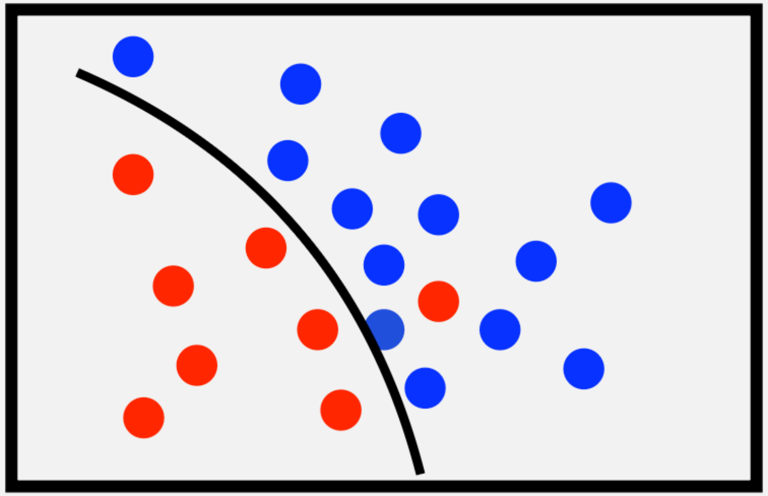


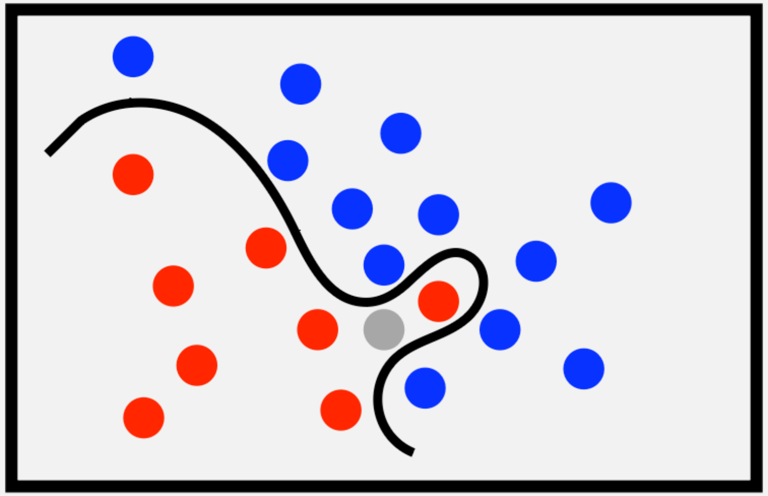


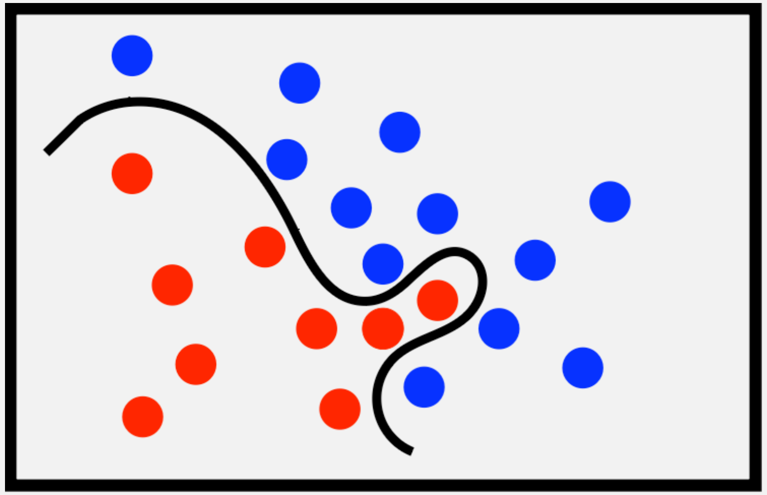


A new document that we want to classify

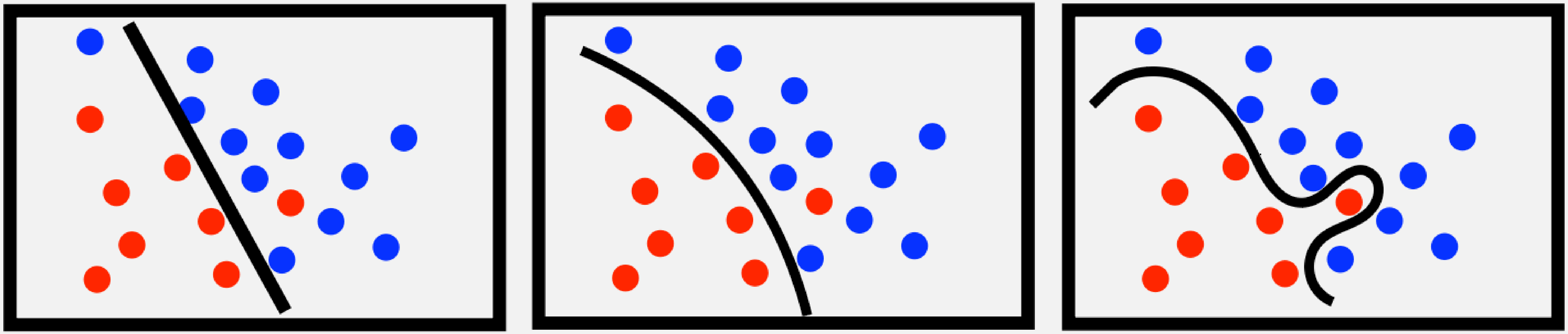








Recap: How to perform classification



“Underfit” “Good fit” “Overfit”


THE BEST WAY TO
EXPLAIN OVERFITTING

Why is testing models important?

Label: umbrella
Confidence: 0.07

Dangers of classification gone wrong

- The self-driving car fatal accident from 2018
  https://www.bbc.co.uk/news/technology-54175359
- Activists posting about violence being banned/restricted on social media because their content gets labelled as promoting violence (for example, spreading information about charities acting against domestic violence)

How to evaluate and compare classifiers

Is this model any good?

Can I stop training yet?

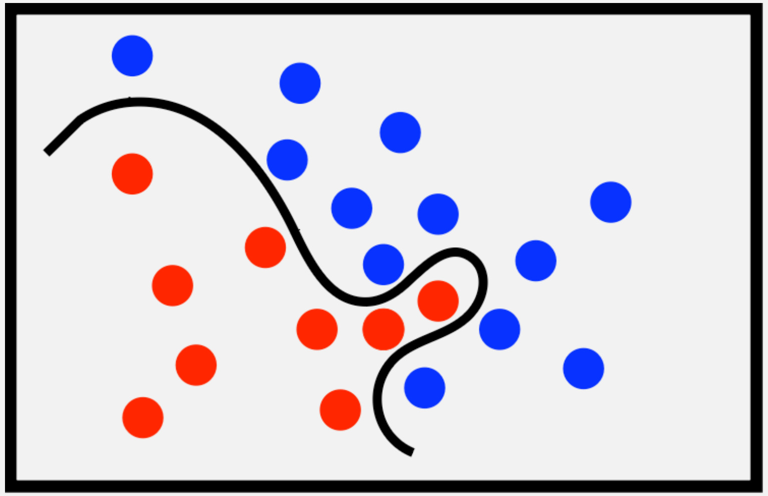
How should it be made better?

Is model A better than model B?

Should we compute accuracy on the training set?

- **No!**

- Problem: A model can achieve perfect accuracy by just memorising the training data!
  - Neural networks can be very good at this ☹



- We usually want to build models that *generalise:* performing well on data that is similar but not identical to the training data

Fonts – styles

- The text on each slide is set to 24pt, this is the minimum accessible font size.
- This template has bespoke bullet points, the bullet levels can be toggled between using the list level buttons, or using the tab key at the start of the paragraph.
  - Level 2
    - Level 3
- Keep text clear and concise.

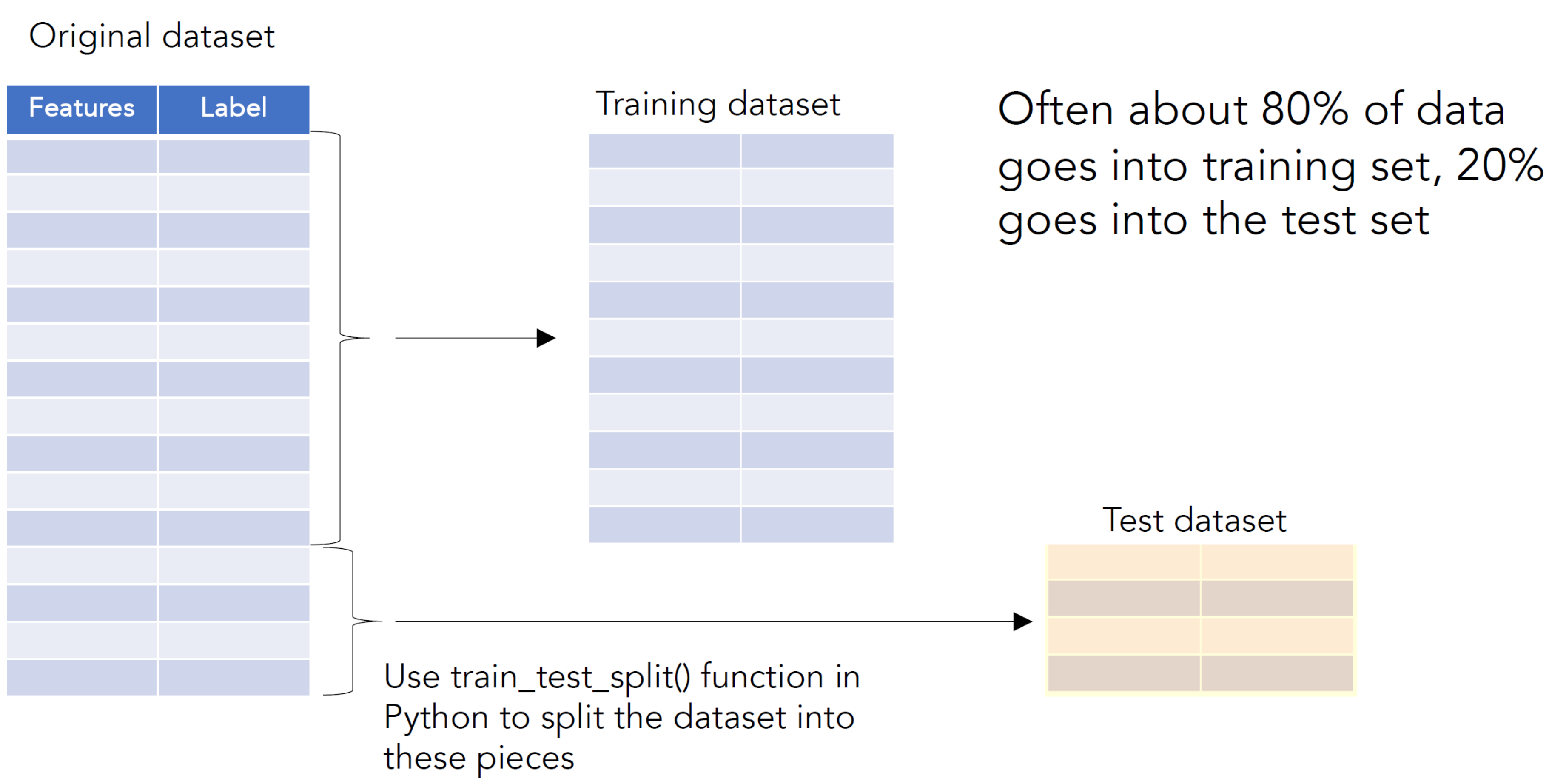
Option #1: Compute accuracy on a known test dataset

- Start with a dataset where both feature vectors and desired labels (“targets”) are known.
- ”Accuracy” is proportion of examples in this dataset where the exact desired label is output.
- Where should this dataset come from?

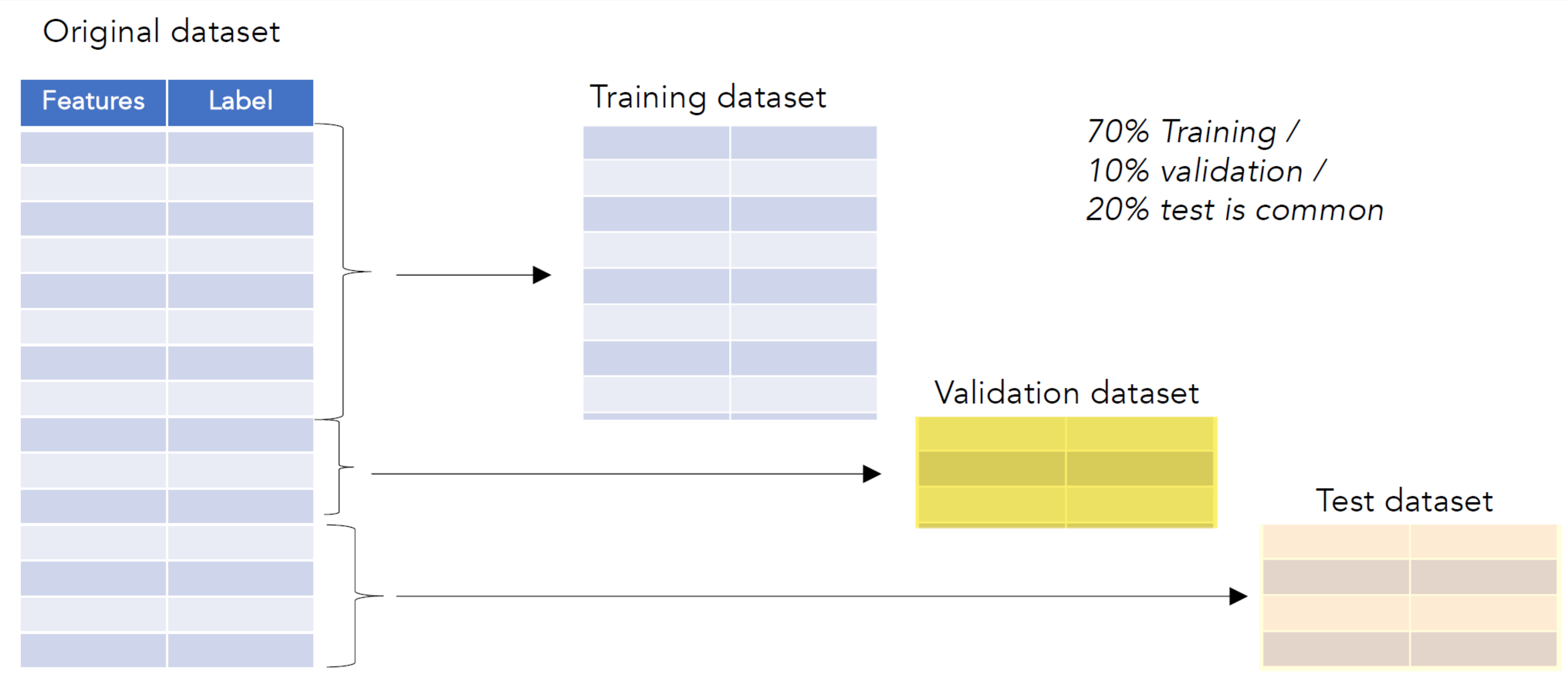
Test dataset: Features	Test dataset: Label	Model output: Label	
<.....>	1	1	Correct
<.....>	0	1	Incorrect
<.....>	0	0	Correct
<.....>	1	0	Incorrect

2 of 4 examples are correct → 2/4 = 50% accuracy

Option #2: Compute accuracy on a "hold-out test set": Data that has not been used for training

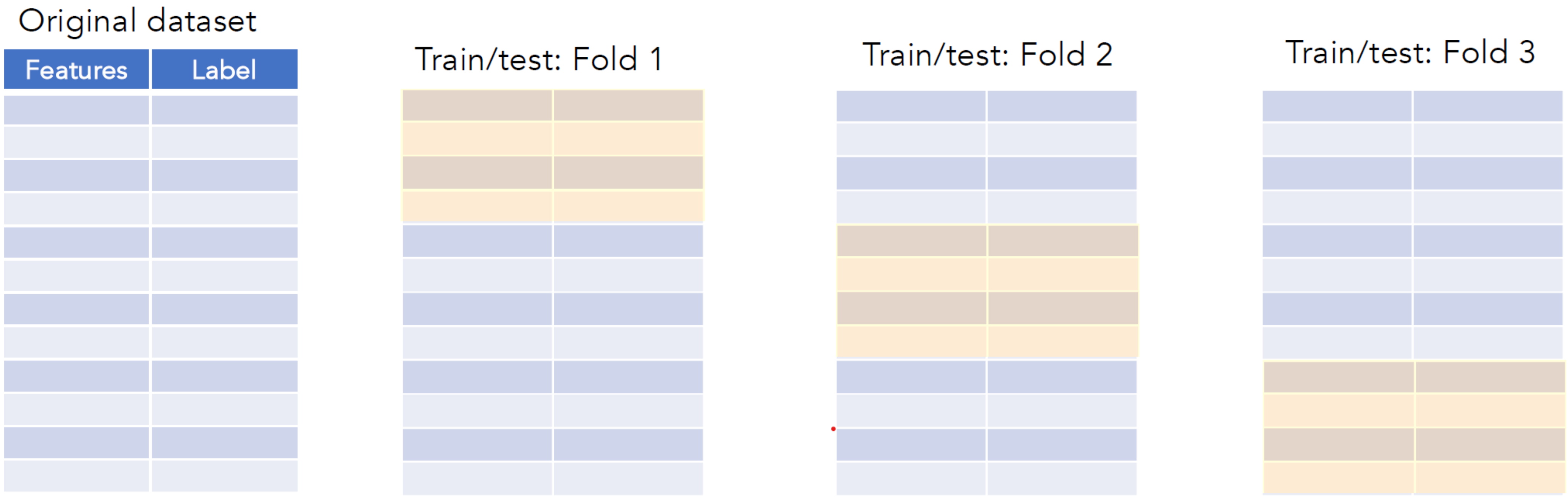


Option #2b: Split into a training set, a "validation set" (to use for tweaking/tuning), and a test set (to use for getting an idea of ultimate accuracy on new data)



Option #3: Compute cross-validation accuracy

Example: 3-fold cross-validation accuracy uses 3 splits:



If test accuracies are 95%, 90%, 85% then overall cross-validation is the average of these (90%)

What do we do with this test set (or cross-validation) accuracy?

- It’s an estimate of how well this classification method is going to perform on data like that in the test set in the future
- Is my classifier good enough to use?
- Is using BOW or other techniques values a better representation to use as features for my classifier?
- Is my nearest-neighbor classifier or my neural network better?
- Which type of neural network is better?
- Slightly different techniques can help us answer more nuanced Questions:

What types of mistakes is it likely to make? Do I need more training data?

Summary: Key points

- A classifier assigns a label / category / class to a document / example
- Documents/examples are represented as vectors of numbers (”features”)
- Labels/classes/categories are mutually exclusive and known in advance
- A classification algorithm is trained on a training dataset (feature vectors + labels for each document), producing a classifier/model
- You can think of a classifier as a function whose input is a feature vector, which outputs/returns a label
- Or you can think of it as a line (hyperplane) separating items of one class from items of other classes in “feature space”
- To evaluate and compare classifiers, we have to examine what they do on “held-out” test data

Thank you