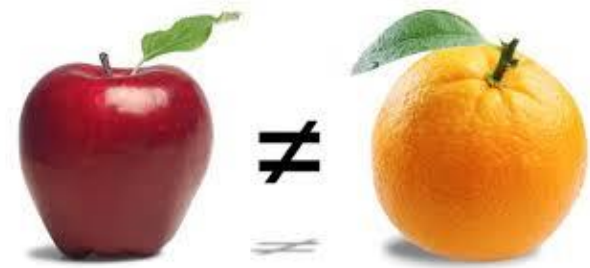
Introduction to Machine Learning

Lecture 3

Instructor:

Dr. Tom Arodz

Recap: Simple threshold-based predictive model for single feature



- Data: Two classes (y_i is -1 or +1), one feature (x_i is a real number)

  Predictive model: $f(x_i)=\text{sign}(x_i + w_0\, b)$

  with one trainable parameter b

  w_0 can be set to -1 (some textbooks) or +1 (other textbooks)

- Set initial value of b (0, or random, or a guess)
- Loop:
  - Present a sample x_i and predict $f(x_i)=\text{sign}(x_i + w_0\, b)$
  - Compare true class y_i with predicted class $f(x_i)$
  - If prediction is right, go to next sample (i=i+1)
  - If prediction is wrong, update b
    - $b_{new}=b_{old} + c[y_i - f(x_i)]w_0$

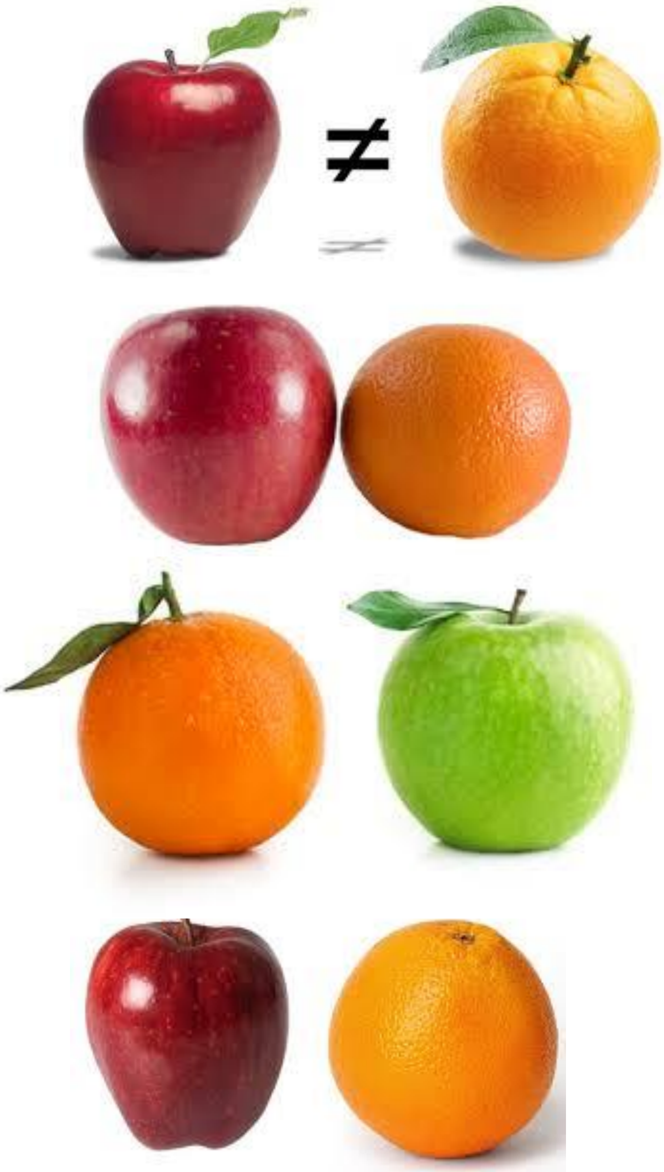
      Learning rate (small positive number chosen by user, e.g., 0.001)

    e.g., if we want $f(x_i)$ to increase from -1 to +1 because y_i is +1:
    - increase b if $w_0 = +1$
    - decrease b if $w_0 = -1$

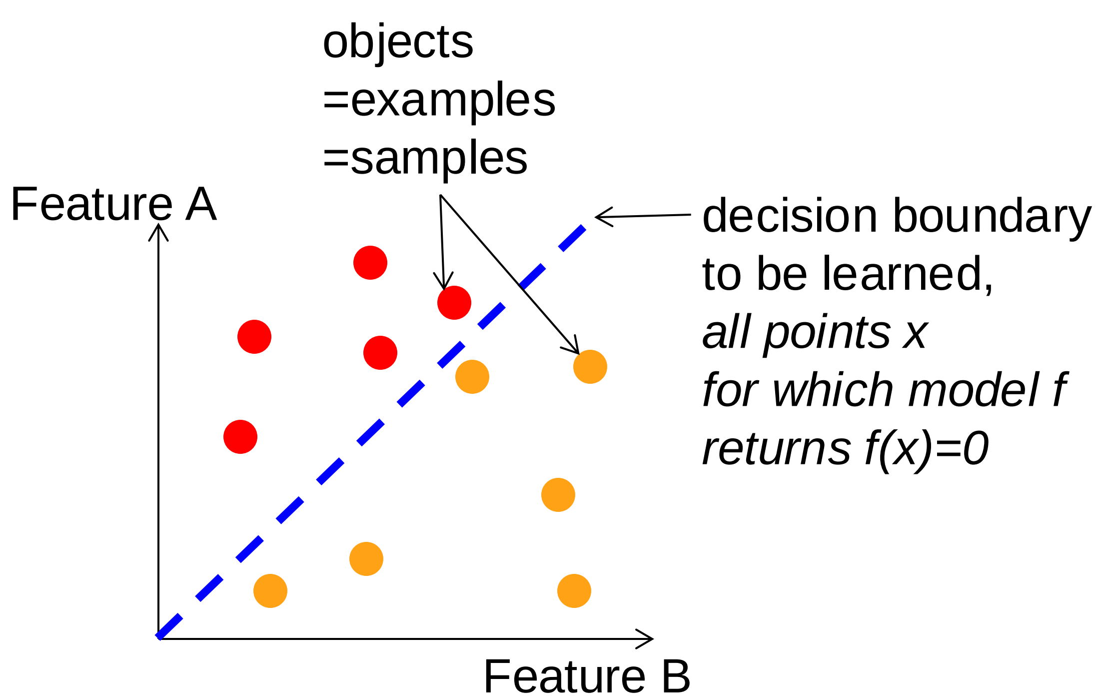
Recap: Supervised ML

- **Typically one feature is not sufficient**
- **We have many features**
- **Possible features**
  - color
  - color variability
  - texture
  - reflectance
  - shape

Recap: Supervised ML



- Building a ***classifier (a predictive model)***
  - F-dimensional feature space: R^F
    - Two features => F=2
  - Training a classifier can be seen as:
    - finding a ***decision boundary*** between class 1 and class -1



Recap: Simple predictive model for two features

- **Data:** **Two classes (y_i is -1 or +1),**
  two features (real numbers x_i^1 and x_i^2)

  Predictive model: $f(\mathbf{x}_i) = \text{sign}(w^T x) = \text{sign}(w_1 x_i^1 + w_2 x_i^2)$
  with two trainable parameters w_1 and w_2

  - Set initial value of w_1 and w_2 (random, or a guess)
  - Loop:
    - Present a sample x_i and predict $f(x_i)=\text{sign}(w_1 x_i^1 + w_2 x_i^2)$
    - Compare true class y_i with predicted class $f(x_i)$
    - If prediction is right, go to next sample (i=i+1)
    - If prediction is wrong, update w
      - $w_{new}=w_{old} + c[y_i - f(x_i)]x_i$
        - that is:
          - $w_{1,new}=w_{1,old} + c[y_i - f(x_i)]x_i^1$
          - $w_{2,new}=w_{2,old} + c[y_i - f(x_i)]x_i^2$

e.g., $y_i=+1$, $x_i=(0.75, -0.25)$, $w_1=w_2= -1$: we want $f(\mathbf{x}_i)$ to increase from -1 to +1:
increase w_1 (from negative -1 towards >0, so that w_1x^1 becomes positive)
but decrease w_2 (make it more negative, so that w_2x^2 is larger positive)

Recap: Simple predictive model for two features

- **Data:** Two classes (y_i is -1 or +1), two features (real numbers x_i^1 and x_i^2)

  Predictive model: $f(\mathbf{x}_i) = \text{sign}(w^T x) = \text{sign}(w_1 x_i^1 + w_2 x_i^2)$ with two trainable parameters w_1 and w_2

  - Set initial value of w_1 and w_2 (random, or a guess)
  - Loop:
    - Present a sample x_i and predict $f(\mathbf{x}_i)=\text{sign}(w_1 x_i^1 + w_2 x_i^2)$
    - Compare true class y_i with predicted class $f(\mathbf{x}_i)$
    - If prediction is right, go to next sample (i=i+1)
    - If prediction is wrong, update w
      - $w_{new}=w_{old} + c[y_i - f(\mathbf{x}_i)]x_i$
        - that is:
          - $w_{1,new}=w_{1,old} + c[y_i - f(\mathbf{x}_i)]x_i^1$
          - $w_{2,new}=w_{2,old} + c[y_i - f(\mathbf{x}_i)]x_i^2$

e.g., y_i=-1, x_i=(0.75, 0.25), w_1=w_2=1: we want $f(\mathbf{x}_i)$ to decrease from +1 to -1:

decrease w_1 and w_2,

but decrease w_1 faster (by c*2*.75)

and w_2 slower (by c*2*0.25)

Why? w_1 gets multiplied by larger number (x^1=0.75) so we want to decrease it more than w_2

Recap: Perceptron using vectors

- One way of making predictions with samples that have 2 features, $x=(x^1, x^2)$
  - Define a 2D vector $w=(w_1,w_2)$ of 2 trainable parameters

Vector (2x1)

^{T} is transpose op., makes it (1x2)

  - Prediction is: $\mathbf{f(x_i)=sign(w_1x_i^1 + w_2x_i^2) = sign(w^T x)}$

Matrix multiplication: (1x2) times (2x1)

Vector (2x1)

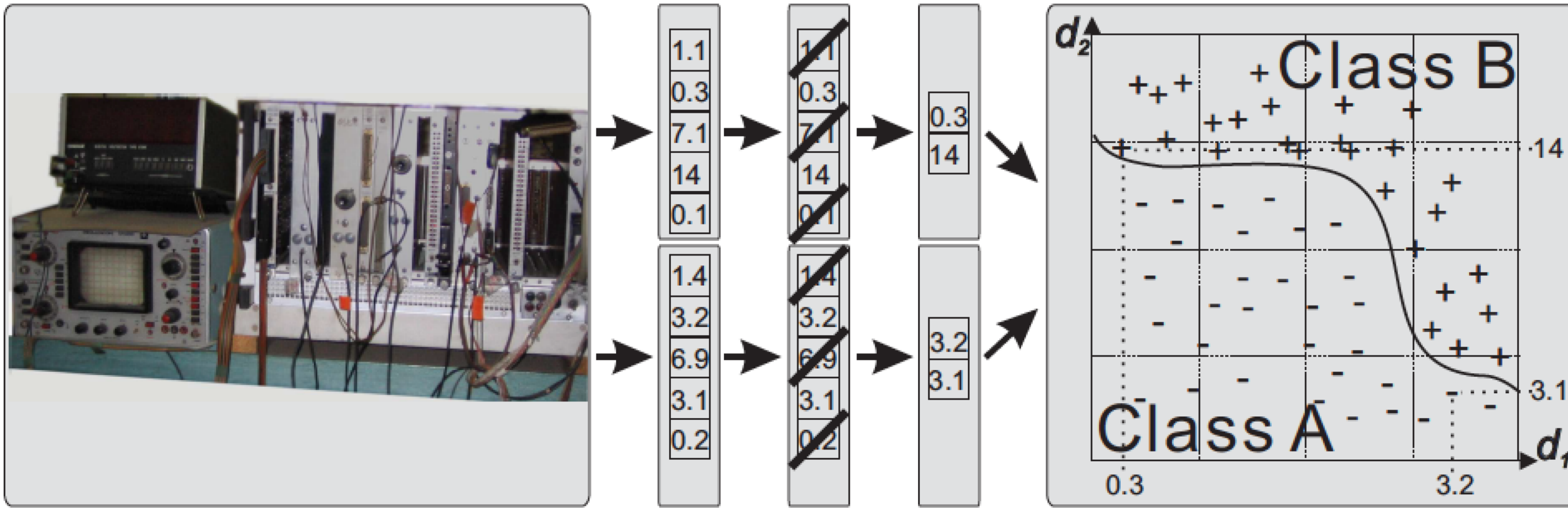
- Perceptron Algorithm
  - **Set initial values of vector w(random guess)**
  - **Loop:**
  - **Present a sample x_i and predict $f(x_i)=sign(w^Tx)$**
    - **Compare true class y_i with predicted class $f(x_i)$**
    - **If prediction is right, go to next sample (i=i+1)**
    - **If prediction is wrong, update w**
      - $w_{new}=w_{old} + c[y_i - f(x_i)]x_i$

vector vector vector

HW1

Supervised ML / Classification

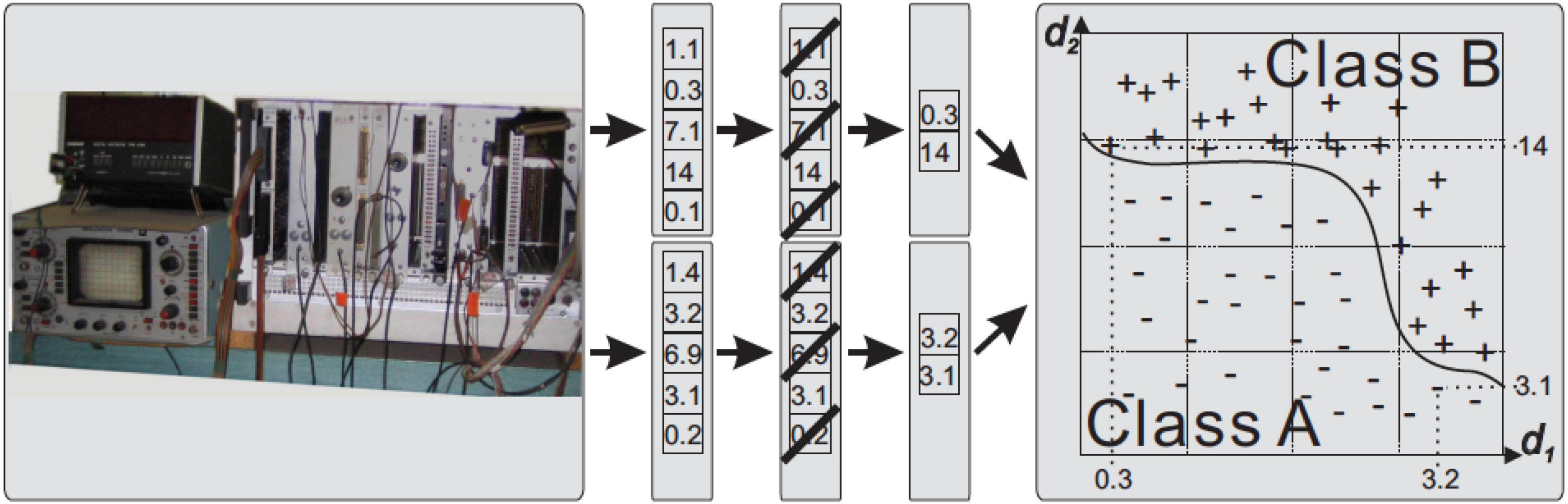
- Building a ***Classifier***



  - Feature generation: measure attributes, transform them if needed
  - Feature selection (optional): select most relevant features
  - Learning: train a machine learning model
  - Validation: test the model, redo the above if not satisfied

Supervised ML / Classification

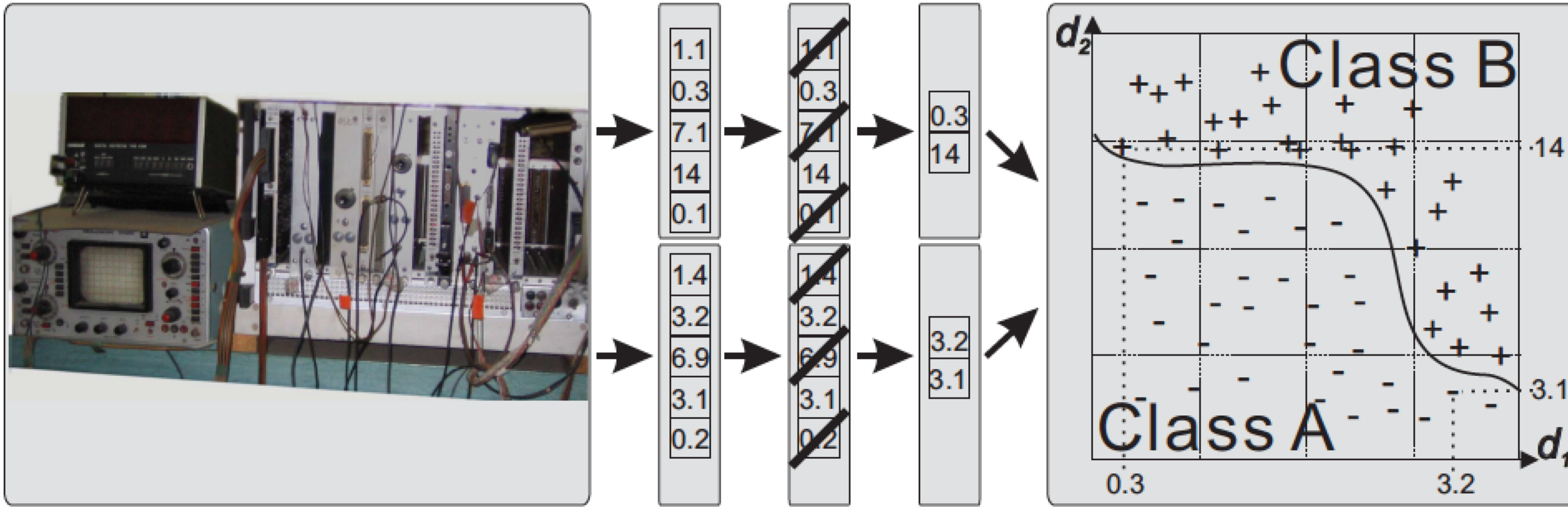
- Building a ***Classifier***



- Features are typically numerical (real numbers, or binary)
    - If not, some numerical encoding is typically used
- Feature space:
    - if we have 2 features for each object
    - then each object is a point in a 2D space
    - 1000 features => 1000 dimensions

Supervised ML / Classification

- Building a ***Classifier***



  - F-dimensional feature space: R^F
  - If we have just two classes (A/B, apples/oranges,…)
  - Then, building a classifier translates to:
    - assigning regions of feature space to class A and the rest to class B
    - finding a ***decision boundary*** in feature space between class A and class B

Summary

- What kind of learning are we talking about?
  - Learning from experience/examples to predict to which of two classes an object belongs
- What are features?
  - The descriptors/attributes (typically numerical) that describe each object/sample
  - Each sample is a vector in the feature space
- What is a decision boundary?
  - A line/surface/hypersurface that has most of samples from one class on one side, and most samples from the other class on the other side
- What is a classifier? Why classification is “supervised machine learning”?
  - A model (mathematical formula) that predicts a class of an example based on its features
  - It’s supervised because the learning algorithm bases its actions on the availability of information about true classes of some samples
- How to build a classifier?
  - One simple way is to use a perceptron

Another ML problem

- If you got a mixed breed dog and you don't know what the breeds are…

- … you could consult an expert, who will look at:
  - size of the dog
  - fur
  - shape of tail, etc.,

  and compare these with what he/she learned about dog breeds

- It's learning to predict classes from features!
  - except, more than one class is correct

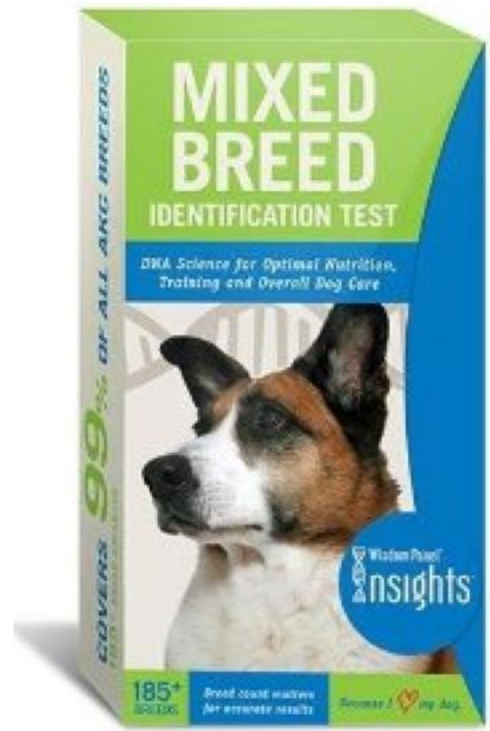
Supervised ML / Classification

- If you got a mixed breed dog
  and you don't know what the breeds are

- You could consult an expert, who will look at:
  - size of the dog
  - fur
  - shape of tail, etc.,

  and compare these with
  what he/she learned about dog breeds

- Or you could order a genetic test!
  - If you know the attribute that truly differentiates the classes
  - And you can measure it
  - Then there's nothing to learn!
  - Classification methods are useful if the differences between classes are not known or not easily quantified



A fully accurate feature eliminates the need for machine learning!

ML: Typical assumptions

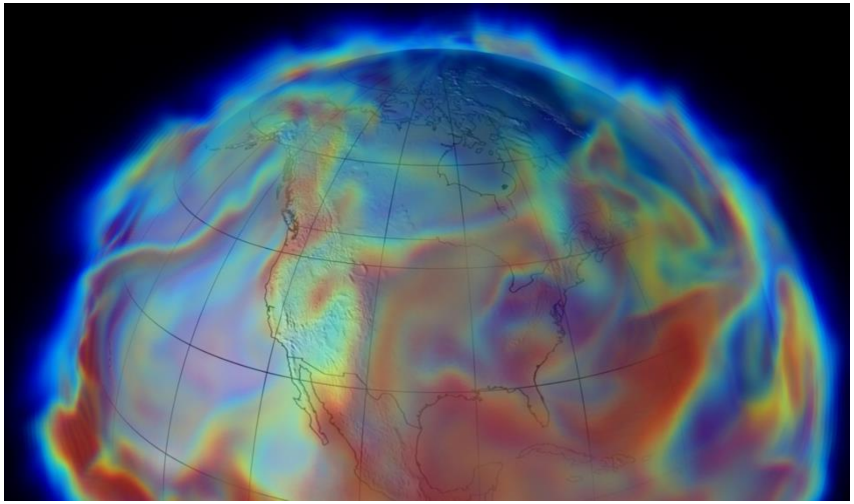
- The features are somewhat informative but not perfectly correlated with the class
  - If features were perfect, no need for machine learning

Supervised ML / Classification

- Rain or sun
  tomorrow afternoon
  - Learning from experience:
    - *Ring around the moon?*
      Rain real soon!
    - *Red sky at night, sailors delight.*
      Red sky in morning, sailors take warning!

Supervised ML / Classification

- Rain or sun
  tomorrow afternoon
  - Learning from experience:
    - *Ring around the moon?*
      Rain real soon!
    - *Red sky at night, sailors delight.*
      Red sky in morning, sailors take warning!
  - Predictions from supercomputer simulation model
  - There are other, better ways of predicting than machine learning
  - Accurate simulation beats learning from data unless the problem is very complex



ML: Typical assumptions

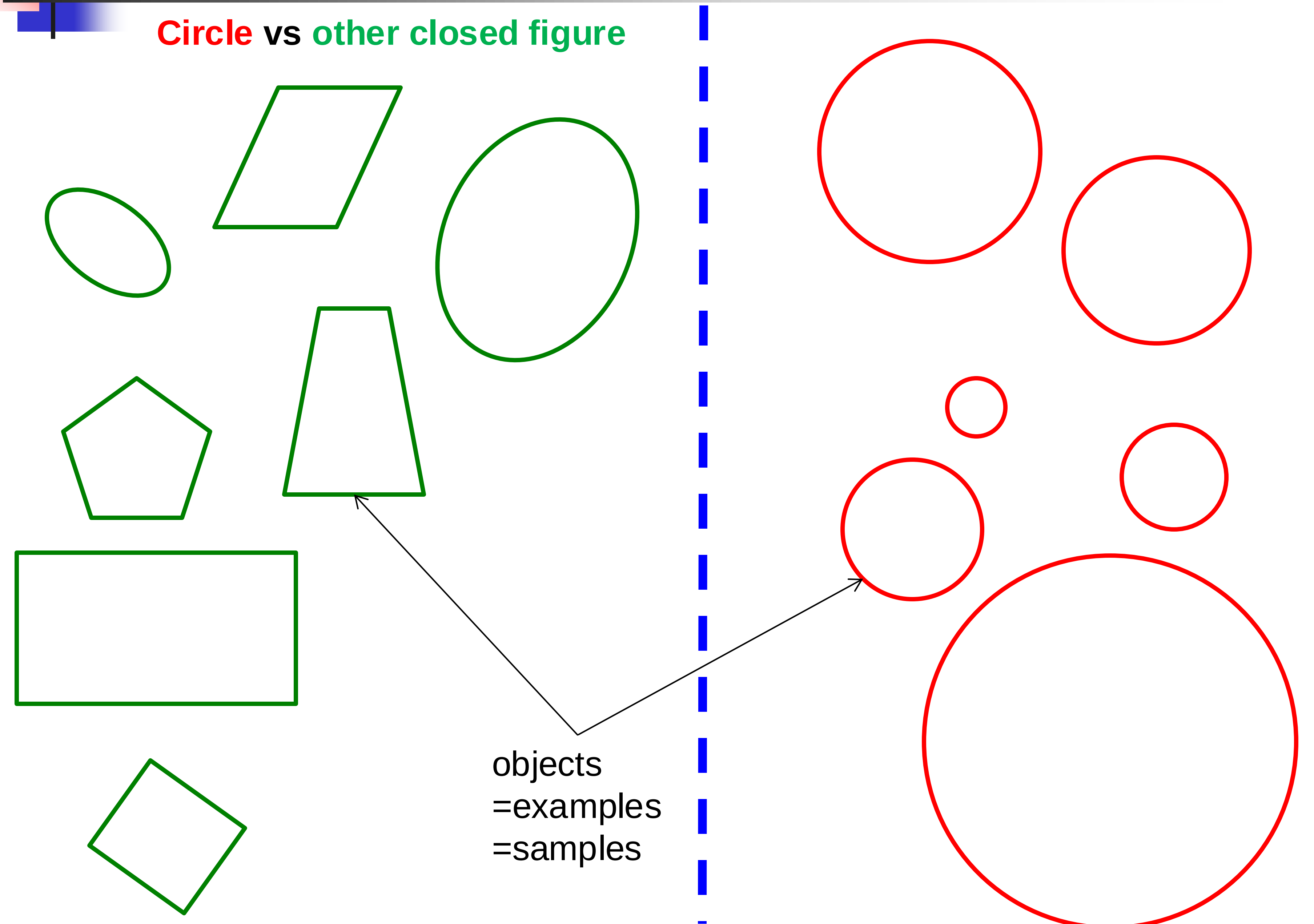
- We do not have a detailed physical/mechanical understanding of the modeled object/process, or, if we do, the process is too complex to simulate
  - Otherwise we would apply known scientific formula or run a computer simulation to get our predictions

This further reinforces the assumption that:

- The features are somewhat informative but very strongly correlated with the class
  - If they were perfectly informative, no need for machine learning

Yet another ML example

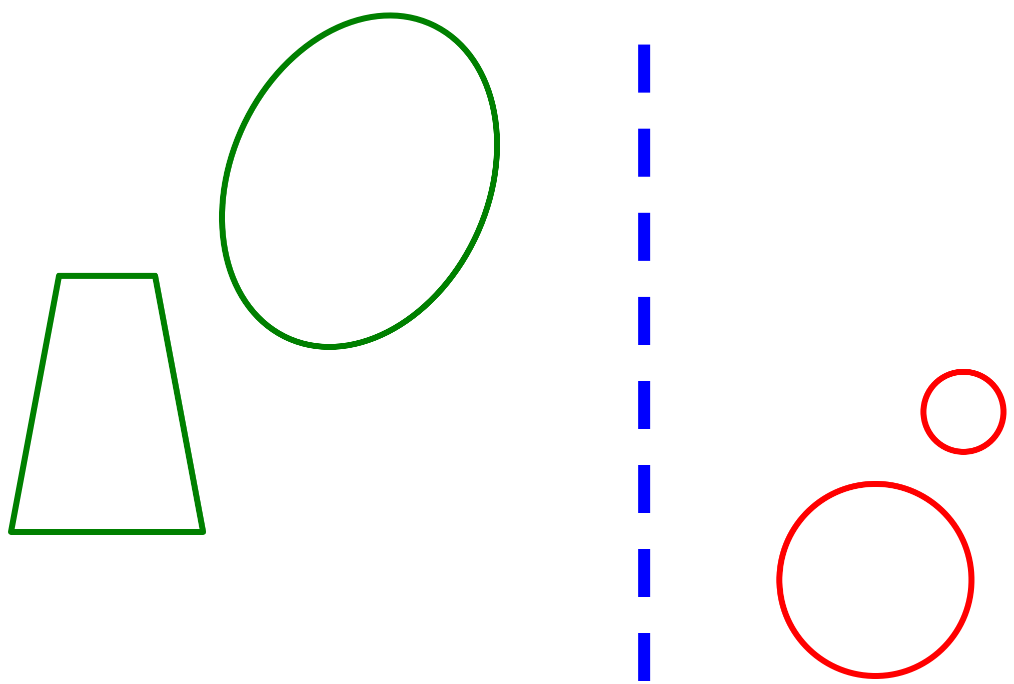
Circle vs other closed figure



Yet another ML example

- Circle vs other closed figure
  - Feature A: figure circumference

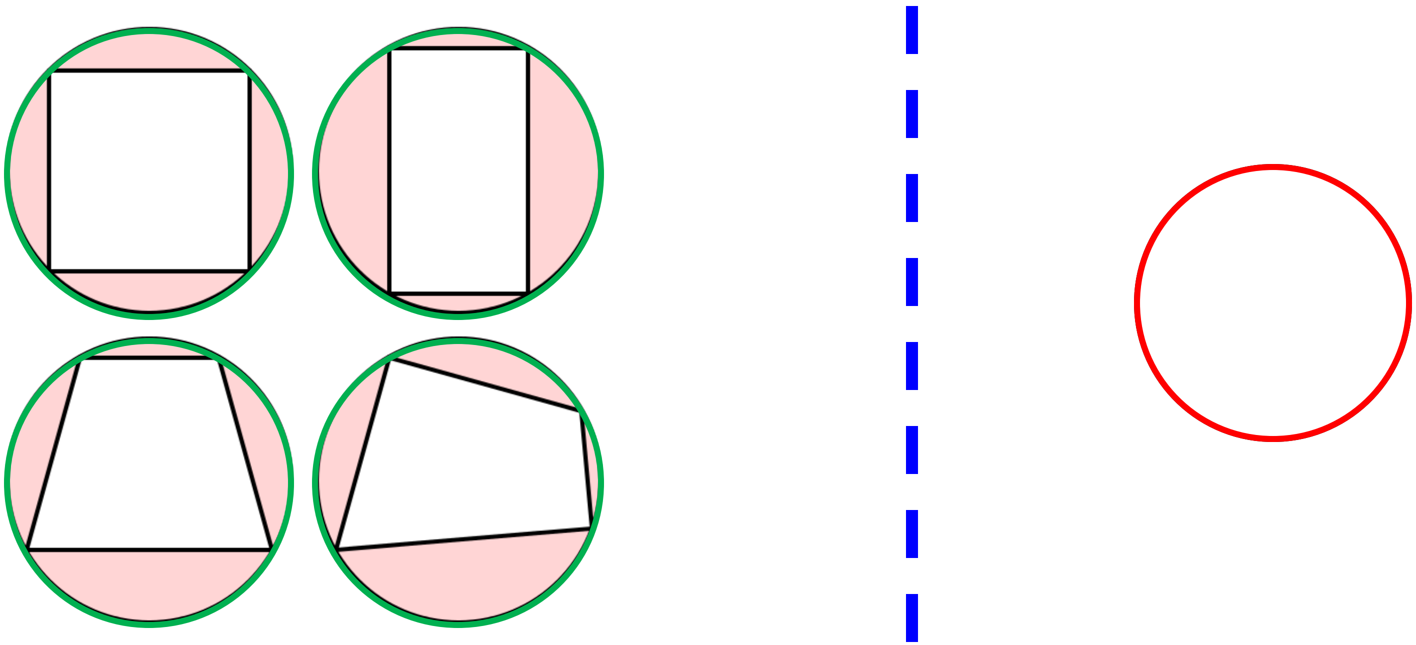
  - Is this a useful/informative feature for this classification task?



Yet another ML example

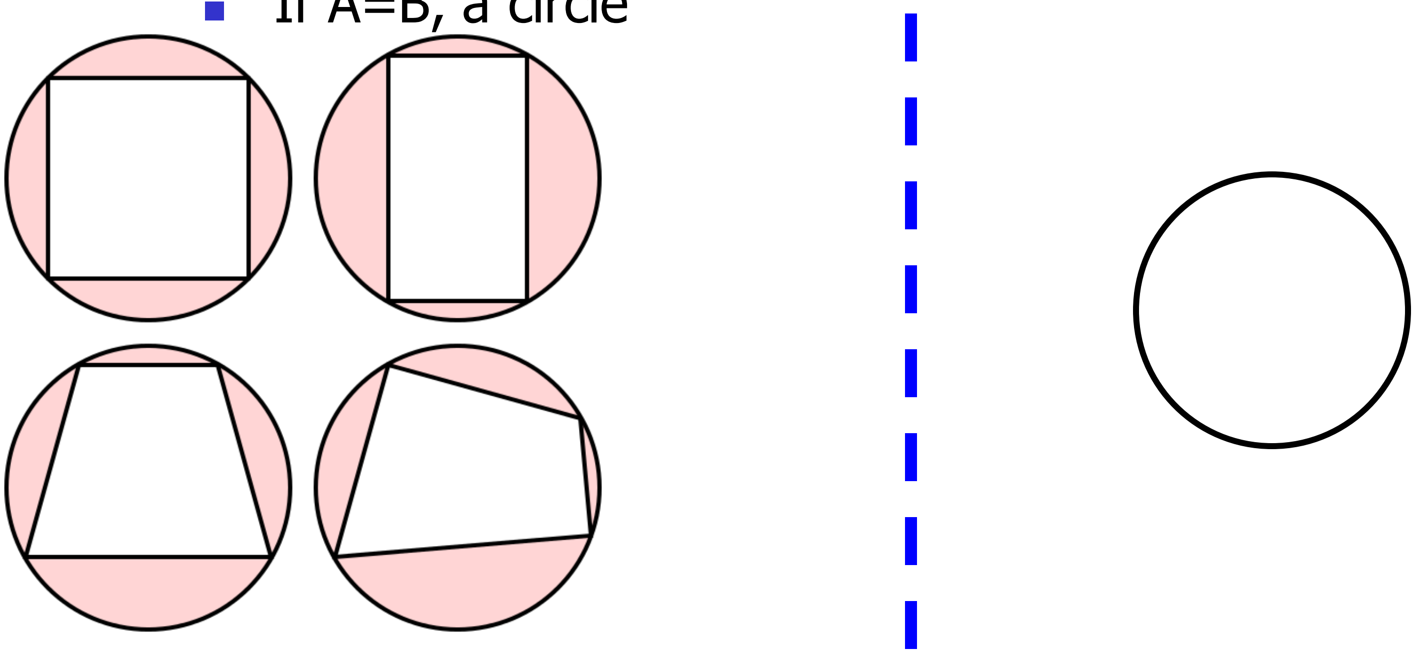
- Circle vs other closed figure
  - Feature B: circumference of circumscribed circle

  - Is this a useful/informative feature for this classification task?



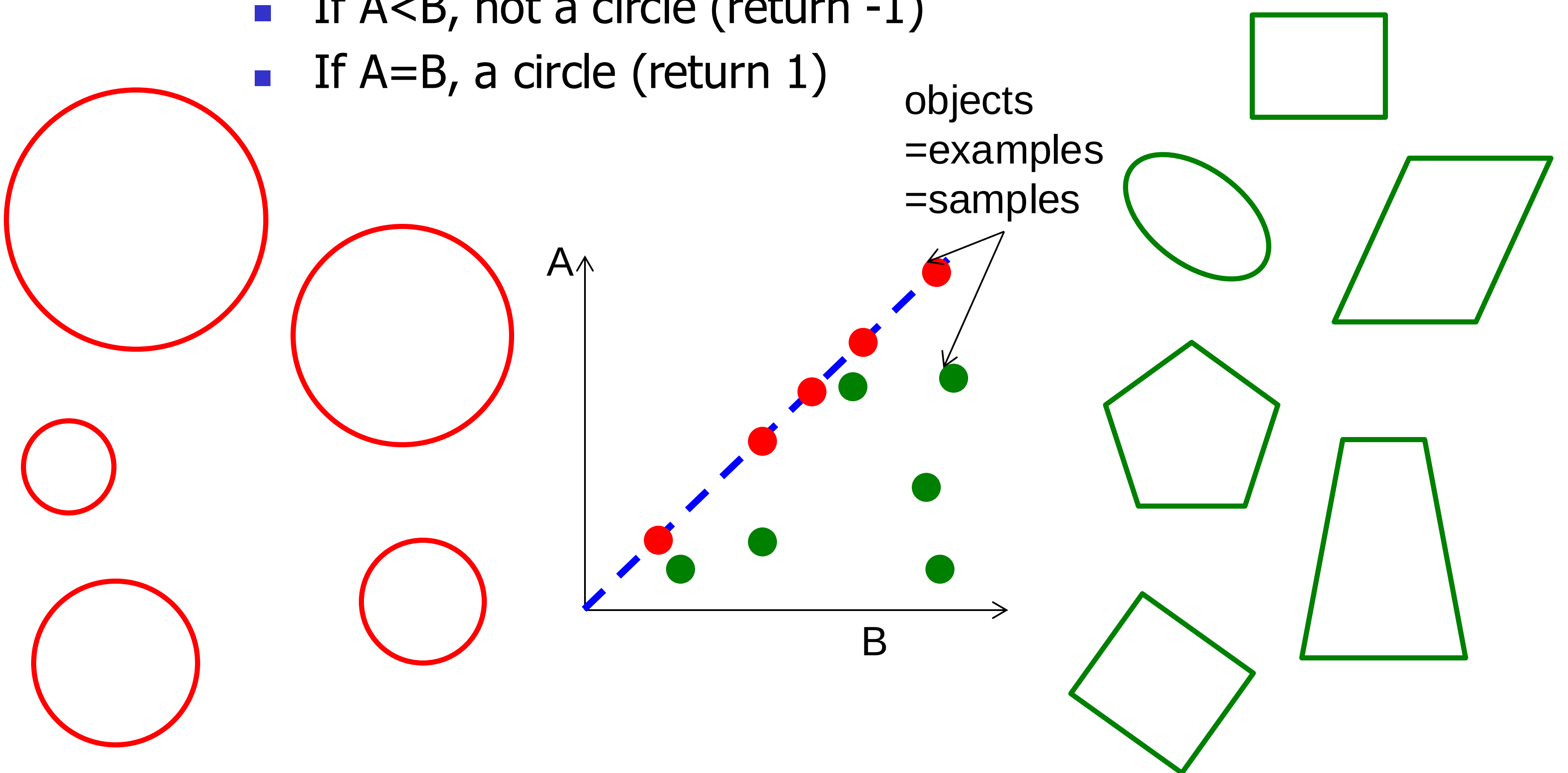
Yet another ML example

- Circle vs other closed figure
  - Feature A: figure circumference
  - Feature B: circumference of circumscribed circle
  - A,B uninformative individually, but informative together!
    - If A<B, not a circle
    - If A=B, a circle



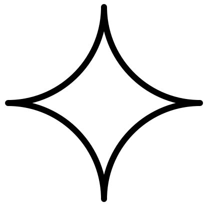
Supervised ML / Classification

- Circle vs other closed figure
  - Feature A: figure circumference
  - Feature B: circumference of circumscribed circle
    - If A<B, not a circle (return -1)
    - If A=B, a circle (return 1)



Supervised ML / Classification

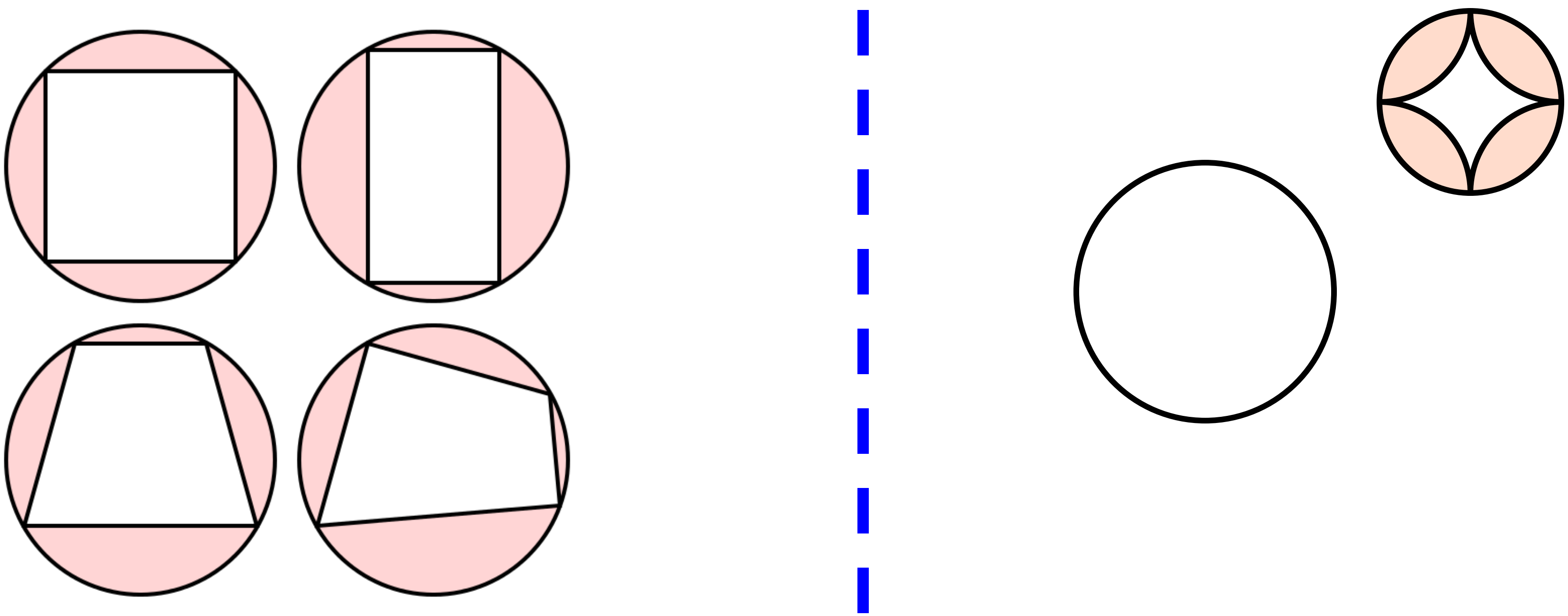
- Circle vs other closed figure
  - Feature A: figure circumference
  - Feature B: circumference of circumscribed circle
    - If A<B, not a circle
    - If A=B, a circle



?

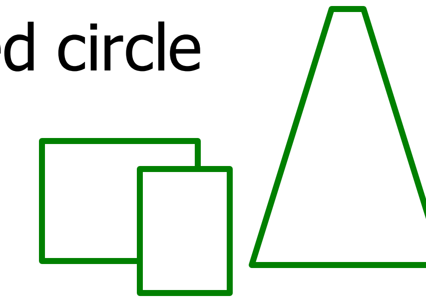
Supervised ML / Classification

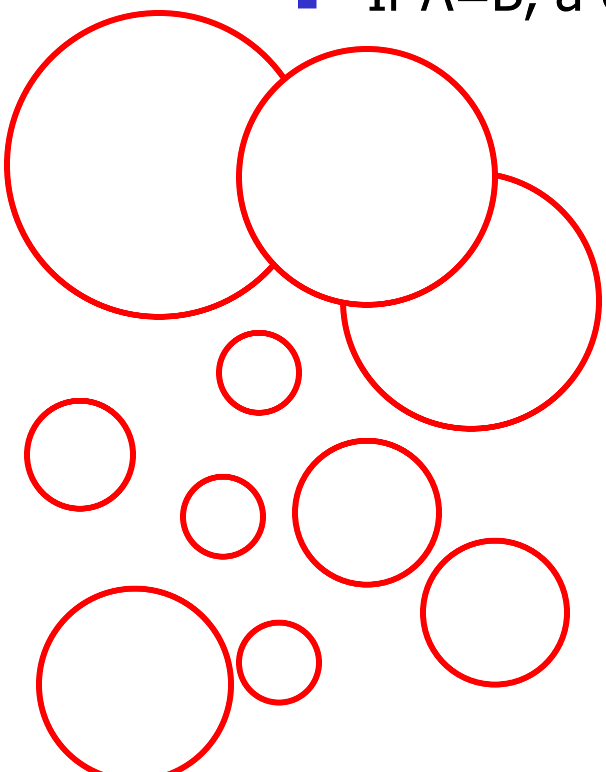
- Circle vs other closed figure
  - Feature A: figure circumference
  - Feature B: circumference of circumscribed circle
    - If A<B, not a circle
    - If A=B, a circle **FALSE if we allow non-convex shapes**

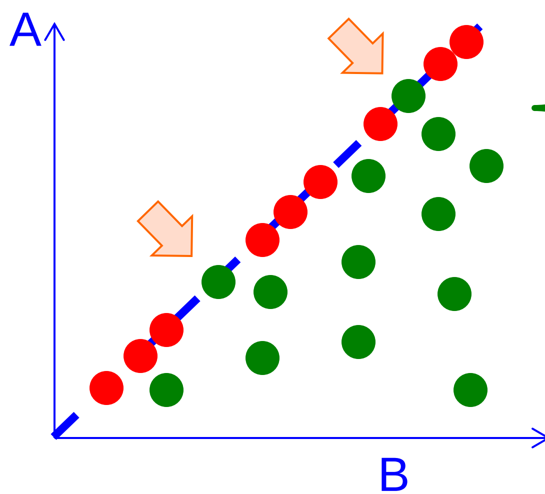


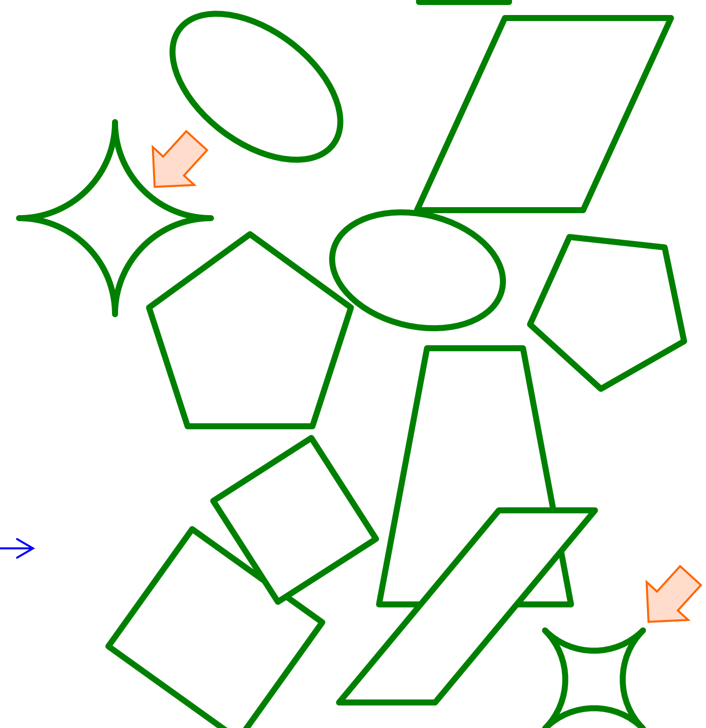
Supervised ML / Classification

- Circle vs other closed figure
  - Feature A: figure circumference
  - Feature B: circumference of circumscribed circle
    - If A<B, not a circle
    - If A=B, a circle **FALSE if non-convex**









ML: Typical assumptions

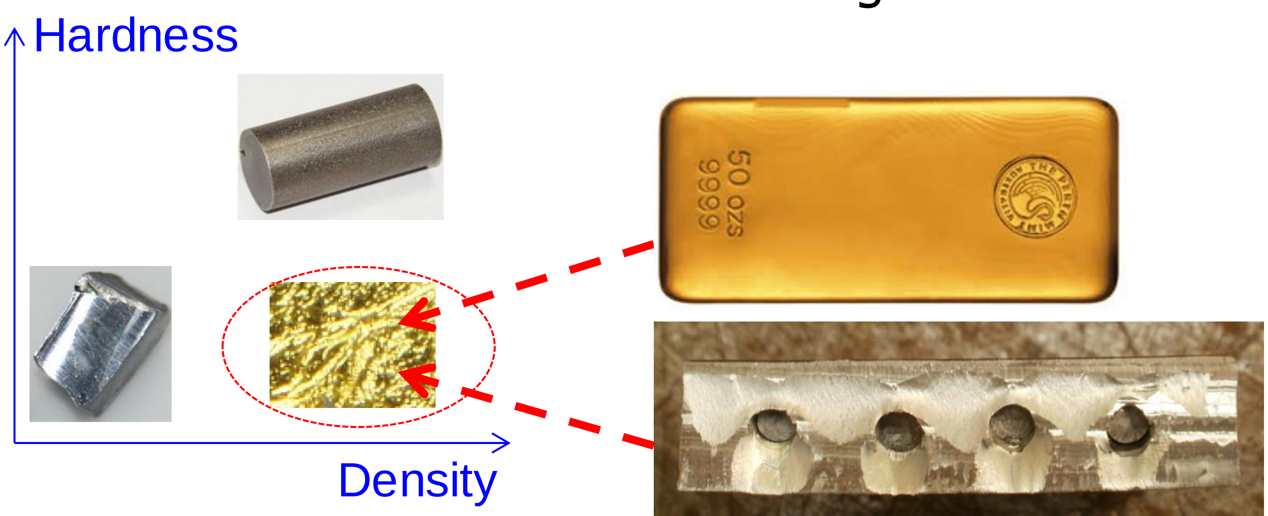
- We do not have a detailed physical/mechanical understanding of the modeled object/process, or the process is too complex
- The features are somewhat informative but not very strongly correlated with the class
- The association between features and class we can learn is likely to be accurate only for objects similar to our training set
  - We do not have much information about objects fundamentally different from we have seen during training

Supervised ML / Classification

- Gold vs fake
  - Magnetism
    - Not magnetic –
      but so is lead, etc.
  - Hardness
    - Soft – but so is lead, or lead covered with gold
  - Density = mass/vol.
    - Much denser than lead

Supervised ML / Classification

- Gold vs fake
  - Hardness
    - Soft – but so is anything covered with gold
  - Density = mass/vol.
    - Much denser than lead
    - But tungsten has the same density as gold
    - And is not magnetic



Gold bar filled with cheap tungsten (wolfram) rods inside!

Hardness/density classifier will fail!

Supervised ML / Classification

- Gold vs fake
  (also, is this cheap gold deal a spam?)

The classification problem changed over time

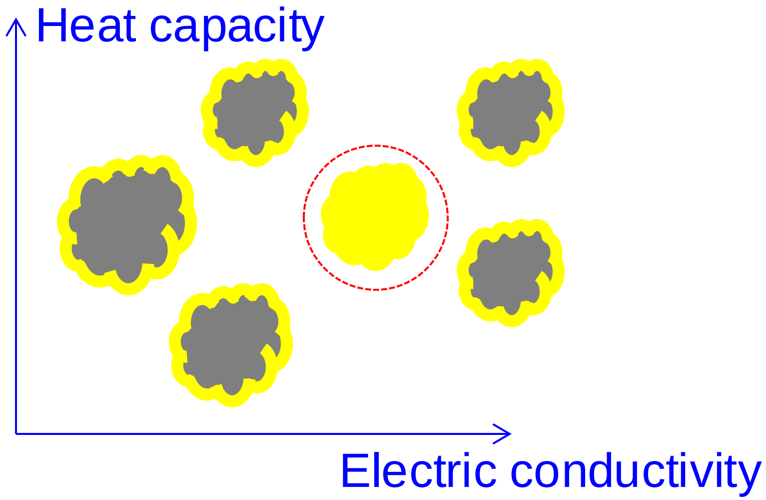
Classifier trained previously *may no longer work*!

Typically, we assume that the problem is stable:

- ***past and future sample are similar***

Supervised ML / Classification

- Gold vs fake
  - Heat capacity
  - Electric conductivity
- These can tell gold easily (especially both features together)



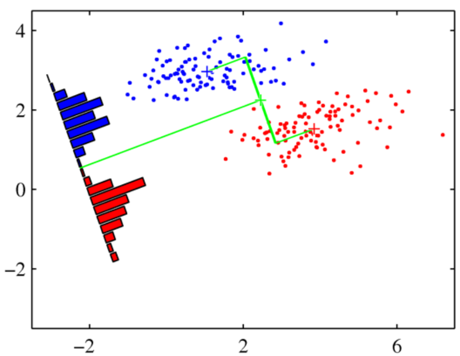
ML: Typical assumptions

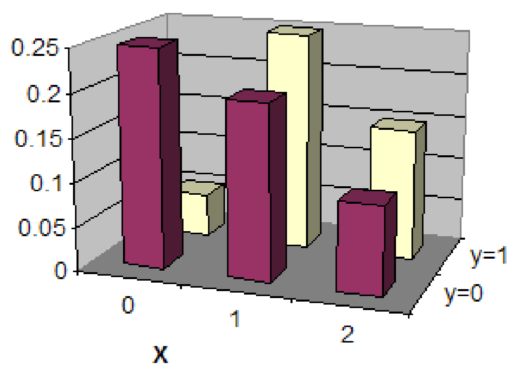
- We do not have a detailed physical/mechanical understanding of the modeled object/process, or the process is too complex
- The features are somewhat informative but not perfectly correlated with the class
- The association between features and class we can learn is likely to be accurate only for objects similar to our training set

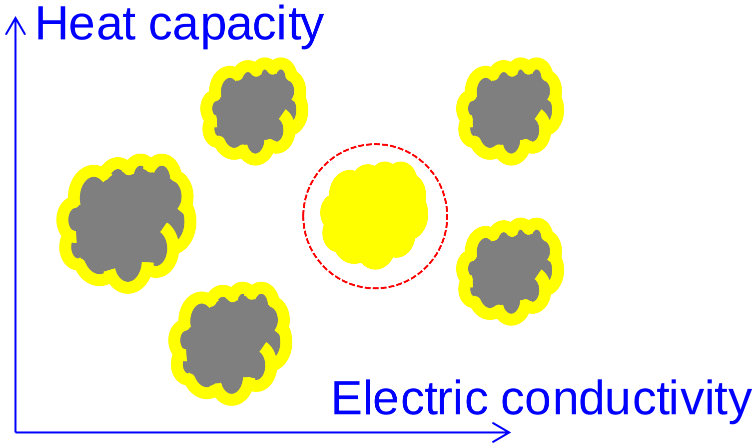
- The association between *regions of feature space* and the *class variable* is fixed
  - If it was changing quickly over time, classifier would soon stop being accurate

ML: Typical assumptions

- What is the mathematical framework in which we can talk about imperfect, typically weak *associations* between *features/ regions of feature space* and the *class variable*?
  - Probability!







- Modern way of thinking about Machine Learning starts with framing learning in a probabilistic way
  - For next class, please revisit what you know about probability from previous courses

News & updates

- Homework 1 is out in Canvas
  - It is due on Thursday, 9/12 (in two weeks), at 5pm

- Next Monday (9/2) is Labor Day holiday, VCU will be closed, **no class**

- Next Wednesday (9/4) there will be **no class** (I have an appointment that I wasn't able to reschedule)
  - There WILL be office hours, starting at 1pm

News & updates

- Tentative schedule of topics, tests, homeworks is also in Canvas

			Lectures	Homework assignments (+/- day or two)			
8/21/24	Wednesday	1	Course Intro (today)	8/21/24	Wednesday		
8/26/24	Monday	2	ML Example: Apples vs Oranges	8/26/24	Monday		
8/28/24	Wednesday	3	ML Assumptions / When to use ML	8/28/24	Wednesday	HW1 out	
9/2/24	Monday		(Labor day)	9/2/24	Monday		
9/4/24	Wednesday		(cancelled)	9/4/24	Wednesday		
9/9/24	Monday	4	Probabilistic View of ML	9/9/24	Monday		
9/11/24	Wednesday	5	Maximum Likelihood Estimation	9/11/24	Wednesday		HW1 due (9/12)
9/16/24	Monday	6	Bayesian Learning	9/16/24	Monday	HW2 out	
9/18/24	Wednesday	7	Trainable Probabilistic Models	9/18/24	Wednesday		
9/23/24	Monday	8	Error Metrics	9/23/24	Monday		
9/25/24	Wednesday	9	Loss Functions	9/25/24	Wednesday		
9/30/24	Monday	10	Cross-Entropy	9/30/24	Monday		HW2 due (10/1)
10/2/24	Wednesday	11	Intro to PyTorch, Test 1 Study Problems	10/2/24	Wednesday		
10/7/24	**Monday**	**12**	**Test 1**	**10/7/24**	**Monday**		
10/9/24	Wednesday	13	Nonlinear Kernel Methods	10/9/24	Wednesday	HW3 out	
10/14/24	Monday	14	Nonlinear Features and Neural Network	10/14/24	Monday		
10/16/24	Wednesday	15	Details of an MLP Layer	10/16/24	Wednesday		
10/21/24	Monday	16	MLP: Universal Approximation	10/21/24	Monday		
10/23/24	Wednesday	17	MLP: Why Depth	10/23/24	Wednesday		
10/28/24	Monday	18	Training Deep MLPs	10/28/24	Monday		HW3 due
10/30/24	Wednesday	19	Convolutional Neural Networks	10/30/24	Wednesday	HW4 out	
11/4/24	Monday	20	Weight Init, and Dropout	11/4/24	Monday		
11/6/24	Wednesday	21	Normalizations and Skip Connections	11/6/24	Wednesday		
11/11/24	Monday	22	LLM Intro	11/11/24	Monday		
11/13/24	Wednesday	23	Self-Attention and Transformers	11/13/24	Wednesday	HW5 out	HW4 due
11/18/24	Monday	24	Large Language Models	11/18/24	Monday		
11/20/24	Wednesday	25	Modern Models for Vision Task	11/20/24	Wednesday		
11/25/24	Monday		(Thanksgiving break)	11/25/24	Monday		
11/27/24	Wednesday		(Thanksgiving break)	11/27/24	Wednesday		
12/2/24	Monday	26	Diffusion-based Generative Models	12/2/24	Monday		
12/4/24	Wednesday	27	Summary, Test 2 Study Problems	12/4/24	Wednesday		HW5 due
12/9/24	**Monday**	**28**	**Test 2**	12/9/24	Monday		

- Now, we move to HW1 slides (they are in Canvas)