Introduction to Machine Learning

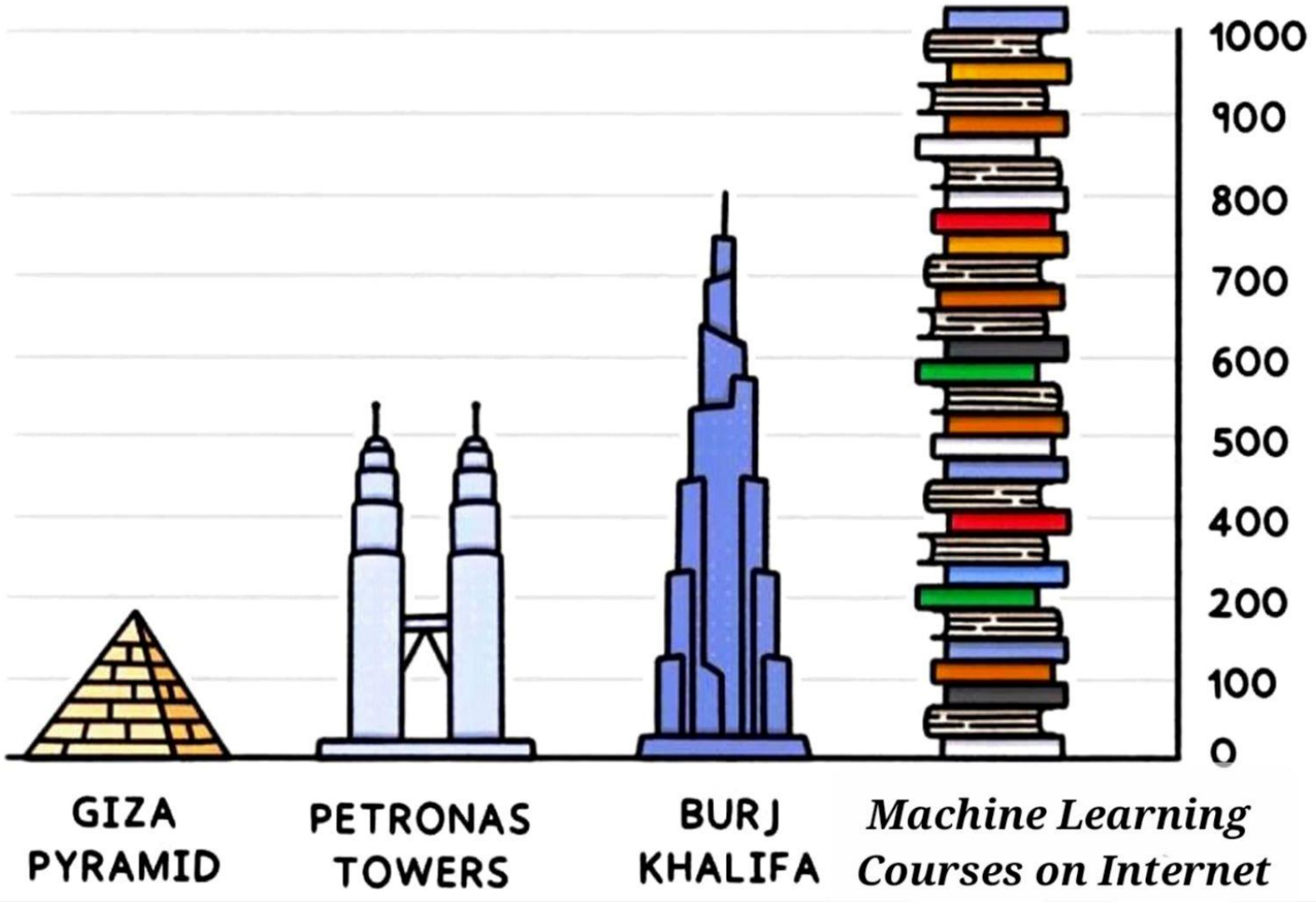
MATH 370: Machine Learning

Tanujit Chakraborty

@ Sorbonne

Course Website: https://www.ctanujit.org/ml.html

Yet Another ML Course



Course Logistics

- Course Name: MATH 370 – Data Science and Machine Learning (L3)
  - An introductory course on ML – supposed to be your first intro to the subject.

- Divided into two parts:
  - Machine Learning part (mostly Theory) will be taught by Dr. Tanujit Chakraborty.
  - Data Science part (Coding and Project) will be taught by Madhurima Panja (TA).

- All material will be posted on the Sorbonne Learn.

- Office hours will be given on demand.

- Course Syllabus, Booklist and other details are available at https://www.ctanujit.org/ml.html

Workload and Grading Policy

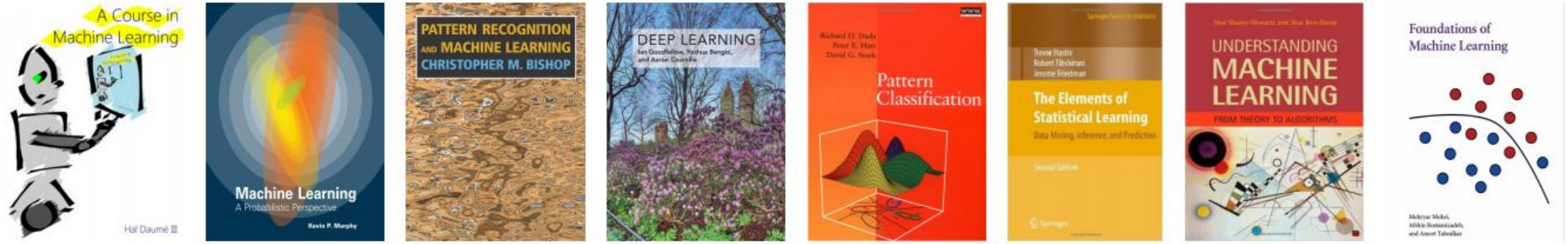
- 5 homework assignments (programming) worth 25%
  - Theory part: Derivations/analysis
  - Programming part: Implement/use ML algorithms, analysis of results. Must be done in Python (learn if not already familiar)
  - Must be typeset in LaTeX (learn if not already familiar)
  - To be submitted via email (before deadline).

- Individual Project Presentation worth 25%
  - Will be given with a live project.
  - Presentation will be done at the end of the semester.

- Quizzes and exams (mid-term and end-term) worth 50%
  - Will be held at Sorbonne.
  - Mid-term will have 20% weightage and final term will have 30% weightage.

Textbook and References

- Many excellent texts are there. Some include:



- Different books might vary in terms of
  - Set of topics covered
  - Flavour (e.g., classical statistics, deep learning, probabilistic/Bayesian, theory)
  - Terminology and notation (beware of this especially)

- We will provide you the reading material from the relevant sources. Also see this latest student-friendly books: Math for ML and Dive into Deep Learning

Course goals..

- Introduction to the foundations of machine learning models and algorithms.

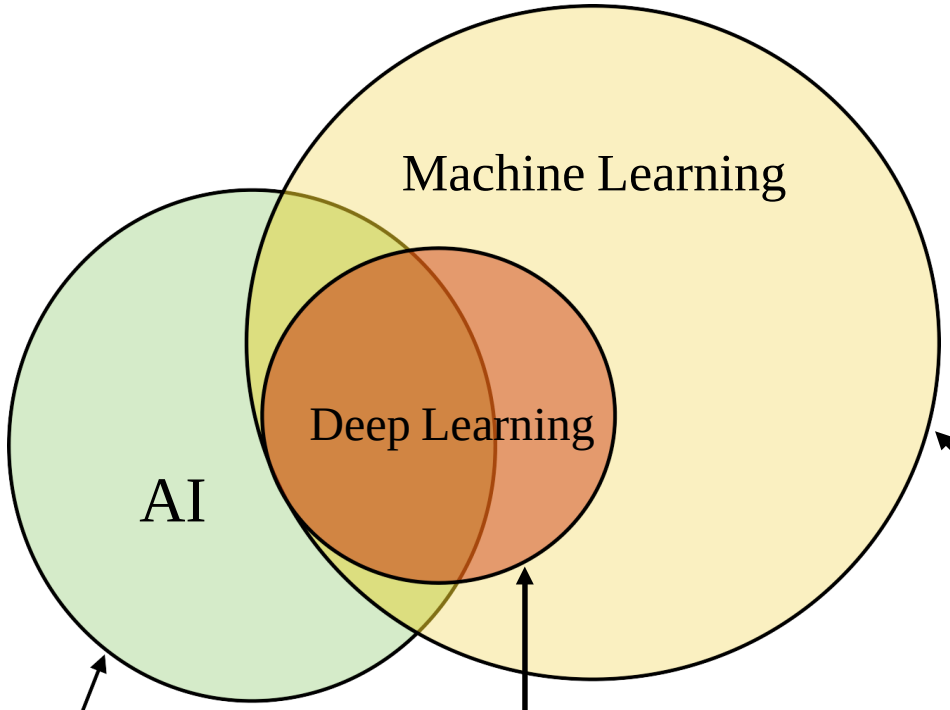
- Focus on developing the ability to
  - Understand the underlying principles behind ML models and algorithms.
  - Understand how to implement and evaluate them.
  - Understand/develop intuition on choosing the right ML model/algorithms for your problem.

- (Hopefully) inspire you to work on and learn more about ML in future.

- An introduction to popular software frameworks and libraries, such as scikit-learn, Keras, Tensorflow, etc. will be given in Practical classes.
  - Detailed coding of various ML techniques from scratch will be done as well.

Introduction to Machine Learning

Machine Learning, AI, and Deep Learning



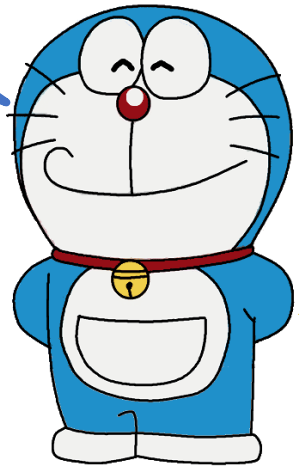
A non-biological system that is intelligent through rules

Algorithms that parameterize multilayer neural networks that then learn representations of data with multiple layers of abstraction

Algorithms that learn models/representations / rules automatically from data/examples

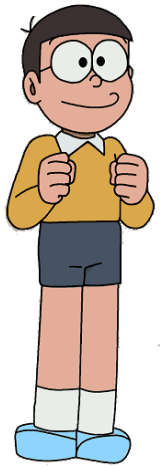
Why is it needed?

Hi, I am Doraemon and I will accompany you as you learn about ML



What is Machine Learning?

How is it done?



What is it?

Machine Learning

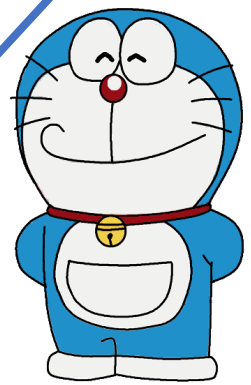
"The art and science of designing adaptive algorithms"

- A Non-adaptive Algorithm
- **Sorting**: given n numbers, sort them in decreasing order of their value

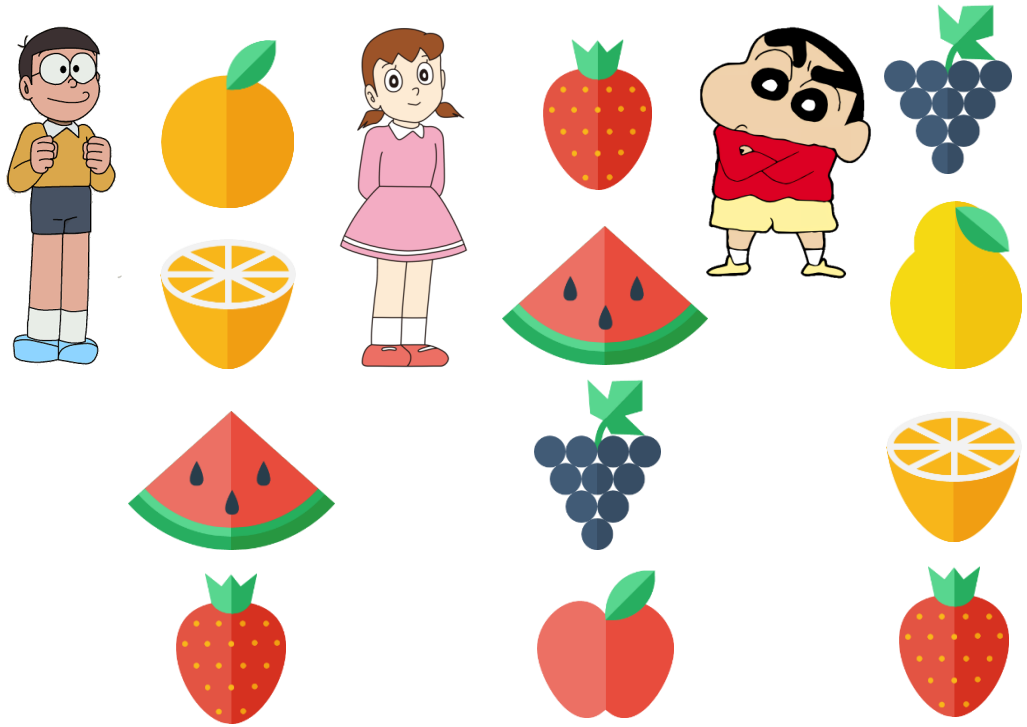
4	9
1	7
5	5
9	4
3	3
7	2
2	1

5	5
-6	4
4	1
-3	0
-2	-2
1	-3
0	-6

I can help you learn patterns that allow you to sort the same set of items differently for each person according to their taste

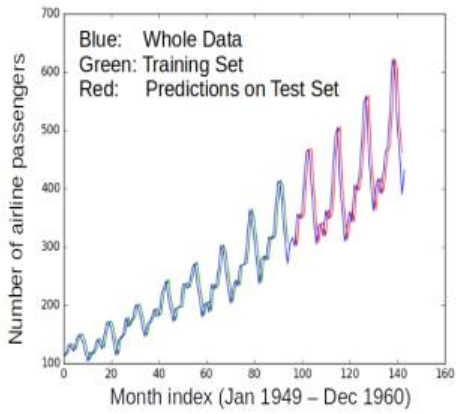


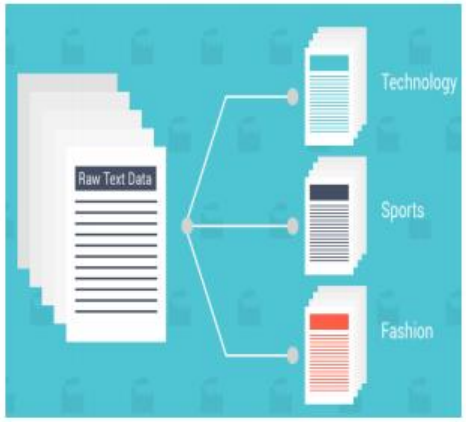
- An Adaptive Algorithm
- **Recommendation**: given a person Nobita and n items, sort items in decreasing order of how much Nobita likes them

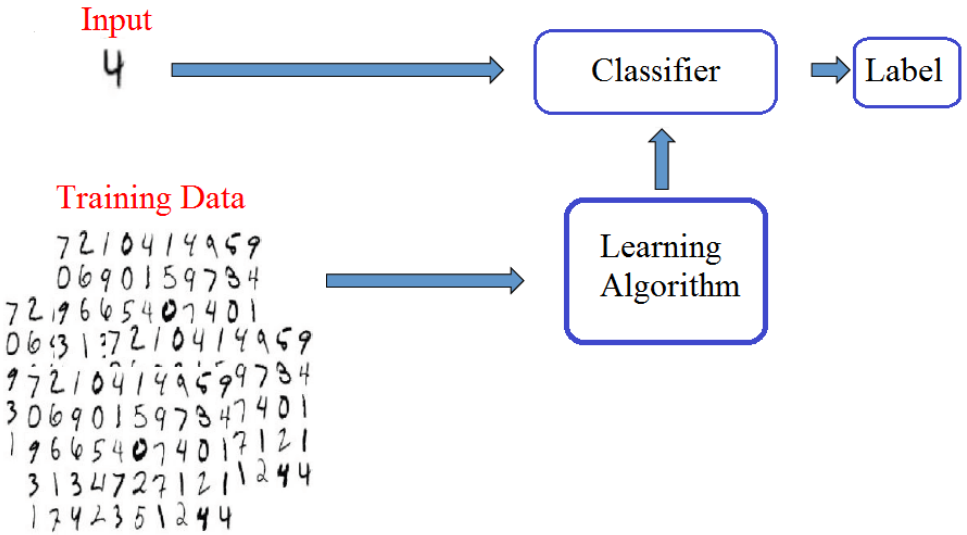


Machine Learning (ML)

- Designing algorithms that ingest data and learn a model of the data
- The learned model can be used to
  - Detect patterns/structures/themes/trends etc. in the data
  - Make predictions about future data and make decisions





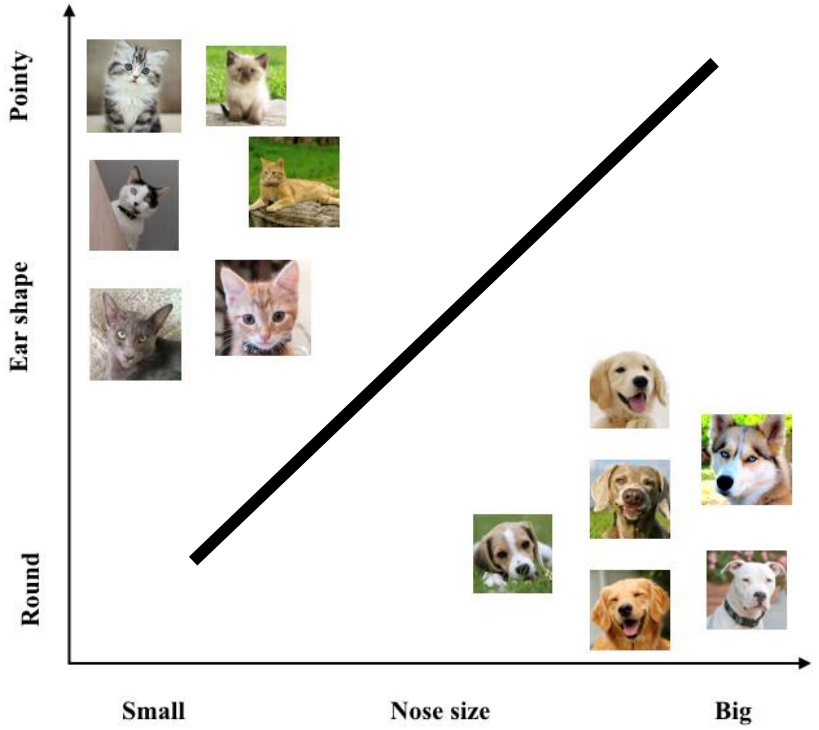


- Modern ML algorithms are heavily "data-driven"
  - No need to pre-define and hard-code all the rules (infeasible/impossible anyway)
  - The rules are not "static"; can adapt as the ML algorithm ingests more and more data

ML: From What It Does to How It Does It?

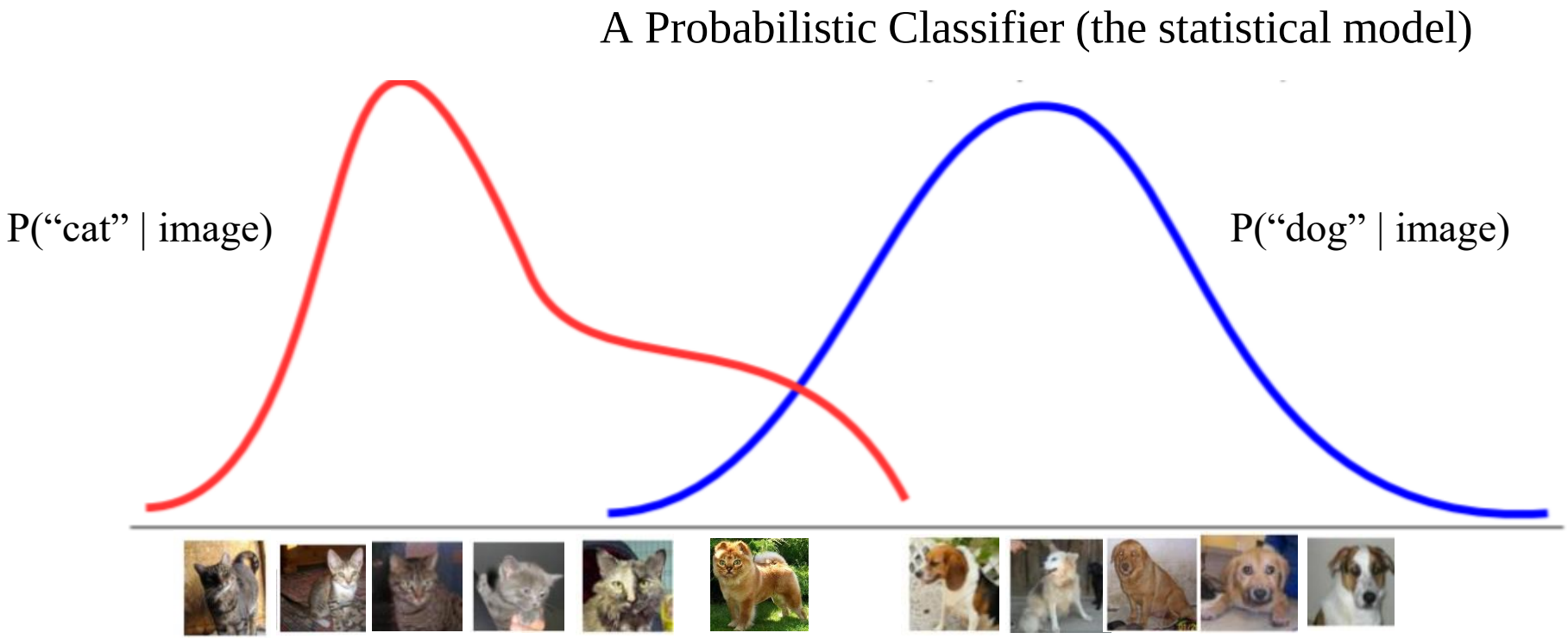
- ML enables intelligent systems to be data-driven rather than rule-driven
- How: By supplying training data and building statistical models of data
- Pictorial illustration of an ML model for binary classification:

A Linear Classifier (the statistical model)



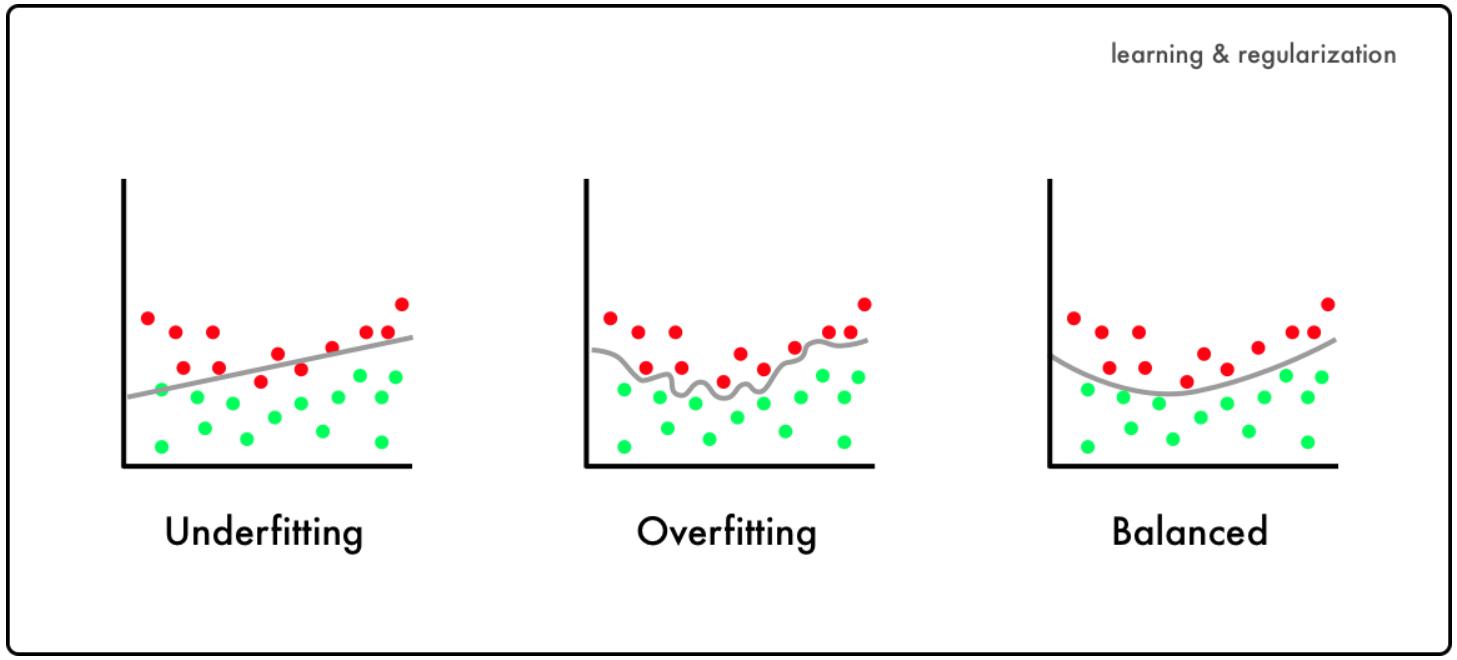
ML: From What It Does to How It Does It?

- ML enables intelligent systems to be data-driven rather than rule-driven
- How: By supplying training data and building statistical models of data
- Pictorial illustration of an ML model for binary classification:



Overfitting = Bad ML

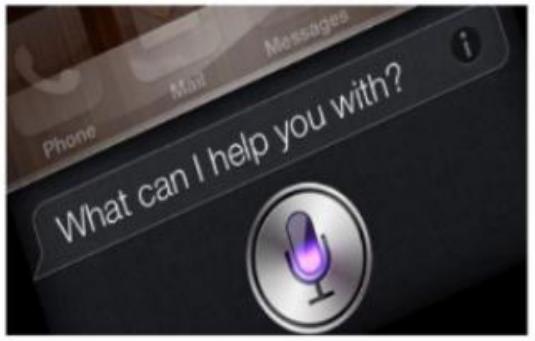
- Doing perfectly on training data is not good enough



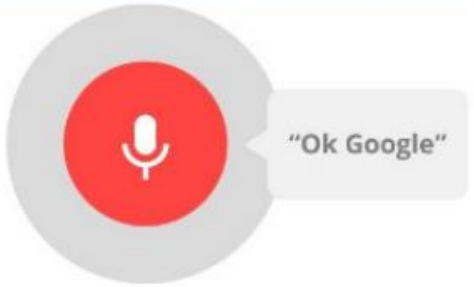
- A good ML model must generalize well on unseen (test data)

- Simpler models should be preferred over more complex ones!

ML Applications Around…



Recommendation systems



Predictive Policing

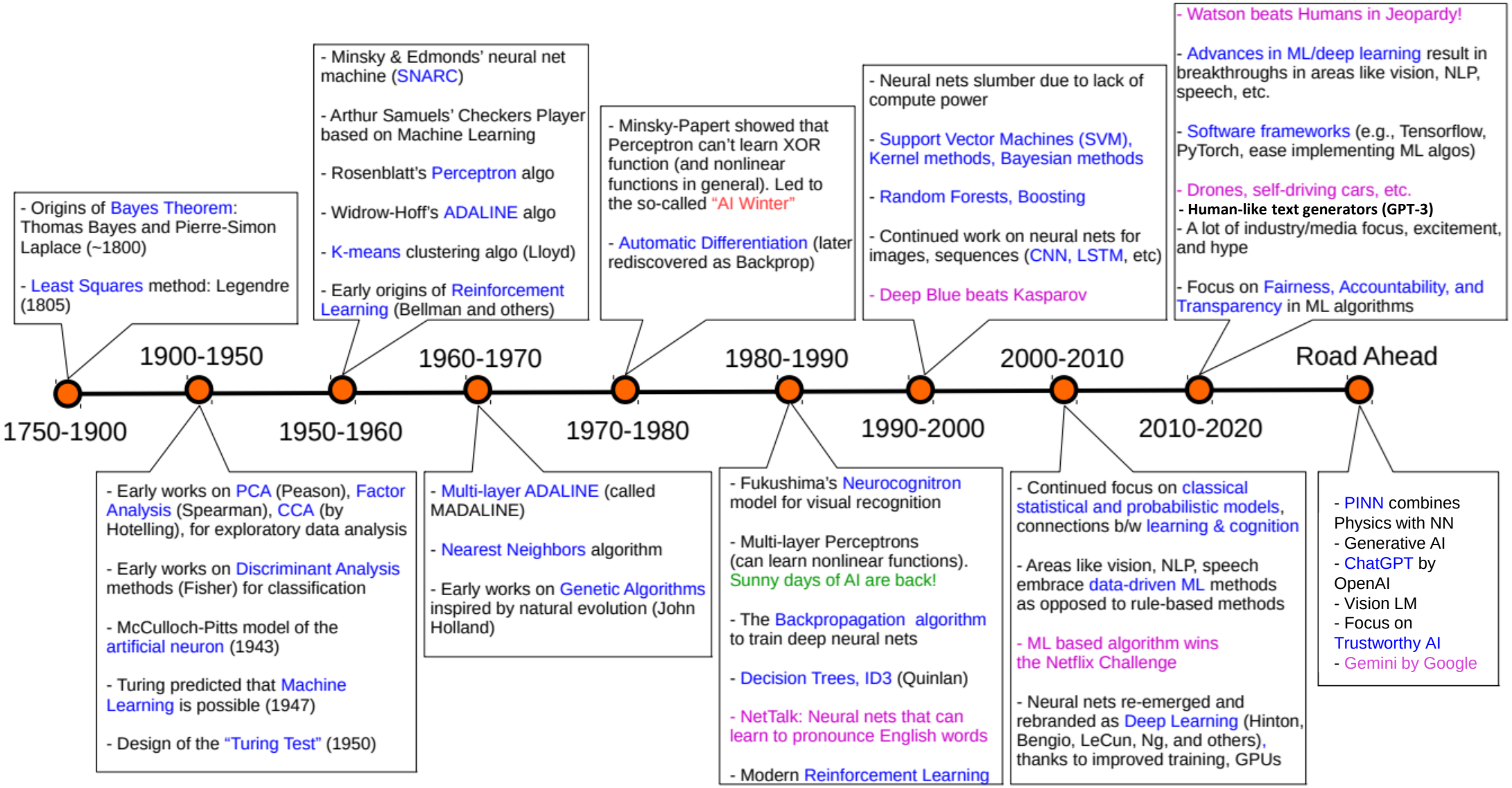
Online Fraud Detection

Key Enablers for Modern ML

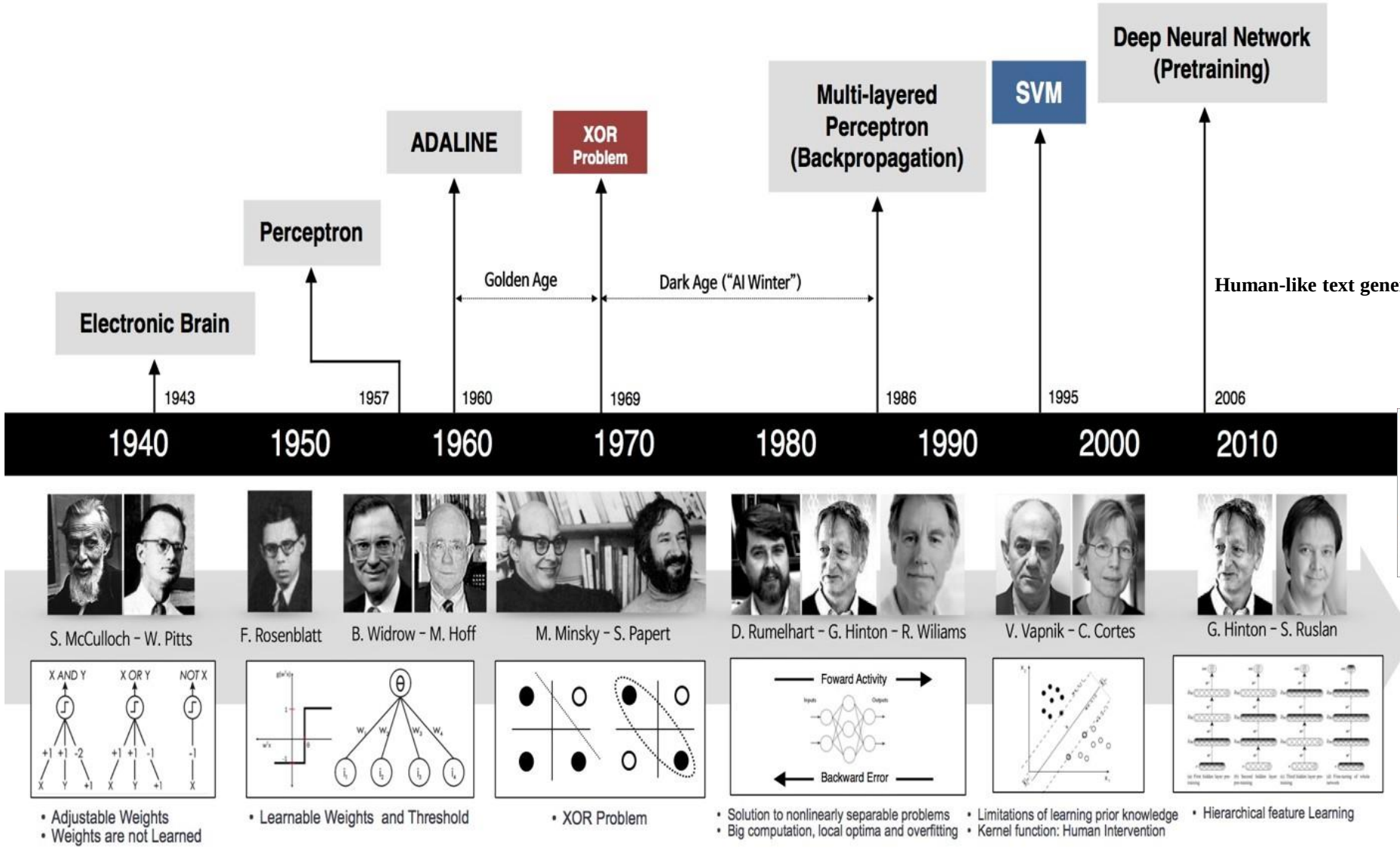
- Availability of large amounts of data to train ML models

- Increased computing power (e.g., GPUs)

History of ML

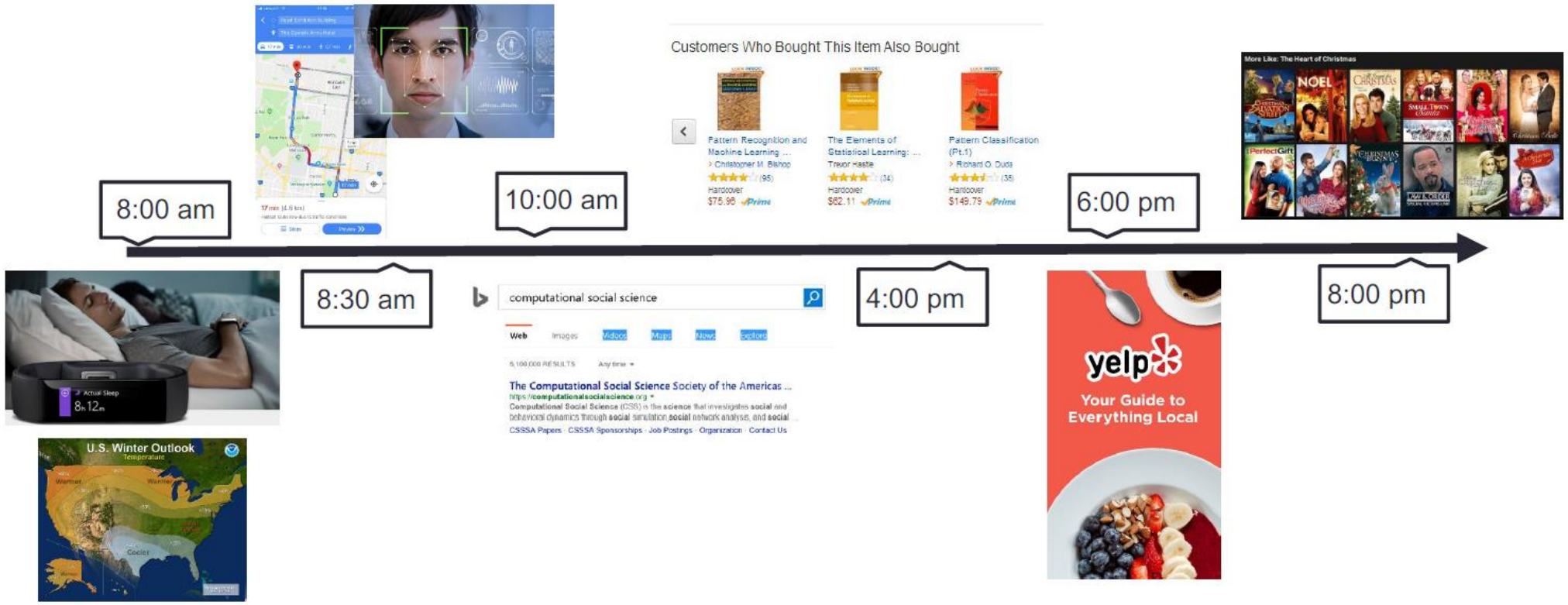


Development of DL



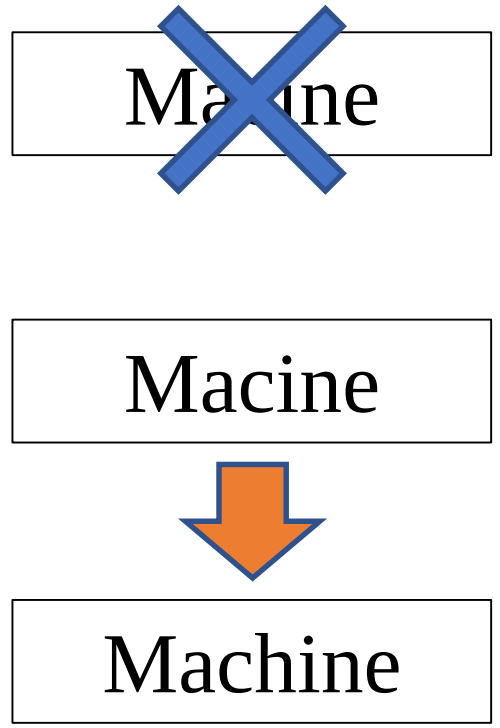
Human-like text generators

ML: Impacting our daily life



When to apply ML?

- Complexity: no “closed form” solutions
  - Humans cannot specify simple rules to get solution
  - Detecting spelling mistakes not a good ML problem
    - A simple dictionary lookup (binary search) is enough
- Presence of immense variety
  - Too many variants to be solved independently
  - Correcting spelling mistakes a very good ML problem
- Need for automation
  - Scalability and speed are main criterion
  - Do we need to automate medicine, driving?

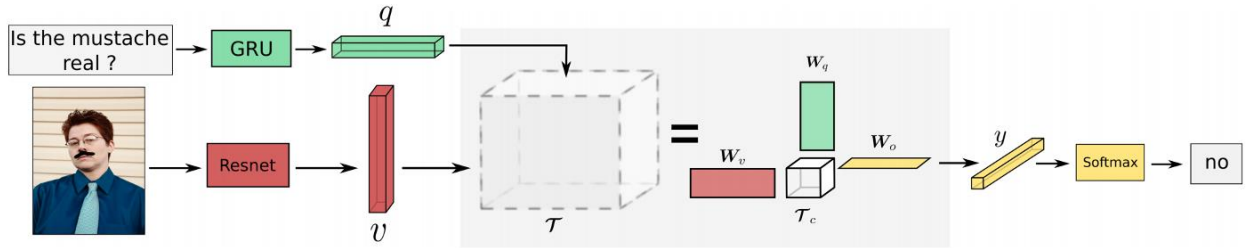


When to apply ML?

- ML + Programming Languages
- ML + Logic/Cognition
- ML + Image Processing
- ML + Video Processing
- ML + Earth Sciences
- …
- ?

ML: Some Success Stories

ML algorithms can learn to translate speech in **real time**



[VQA - Mutan 2017]

"man in black shirt is playing guitar."

"construction worker in orange safety vest is working on road."

"two young girls are playing with lego toy."

"boy is doing backflip on wakeboard."

[Karpathy 2015]

ML: Some Success Stories (Image Restoration)

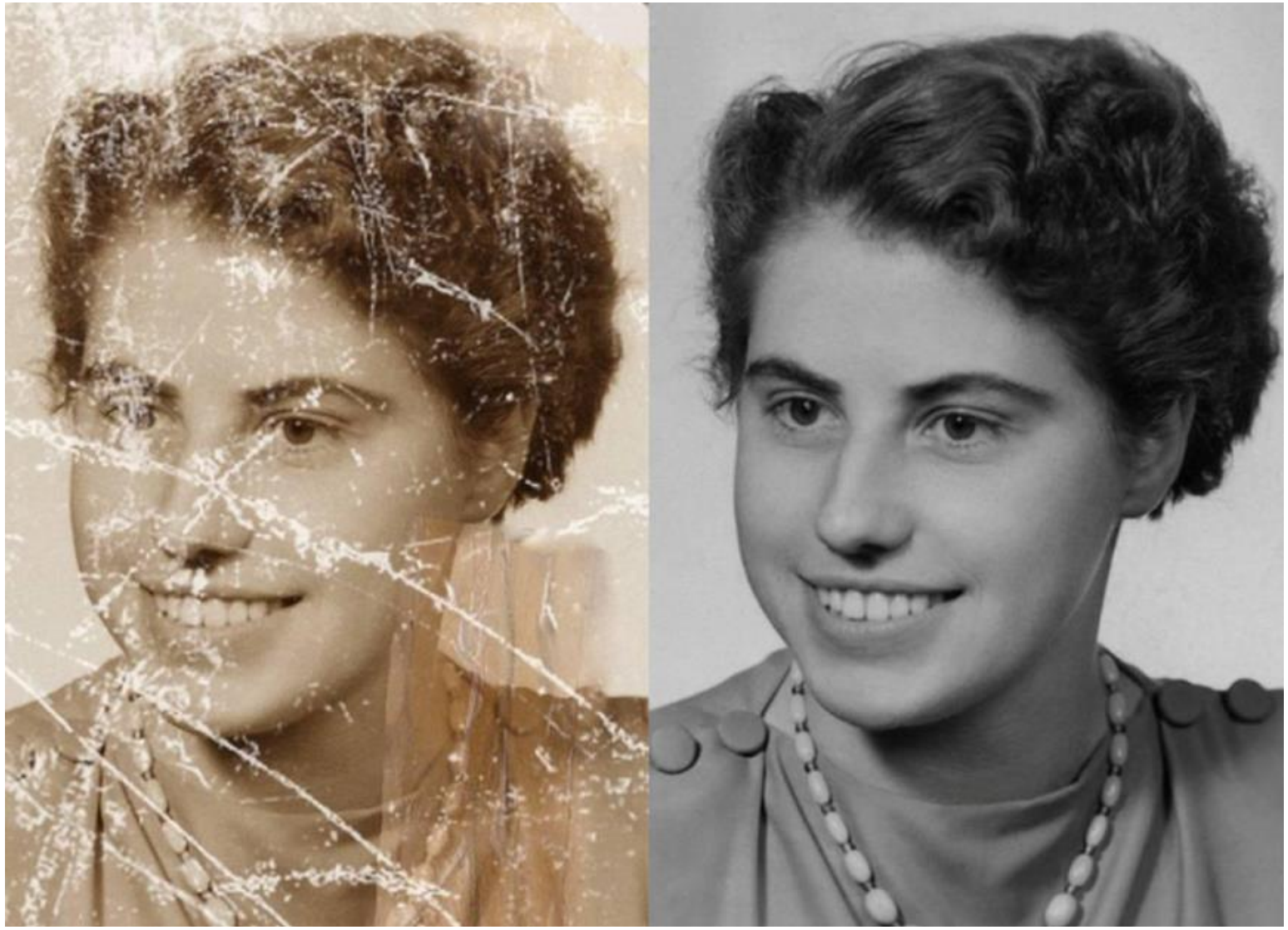
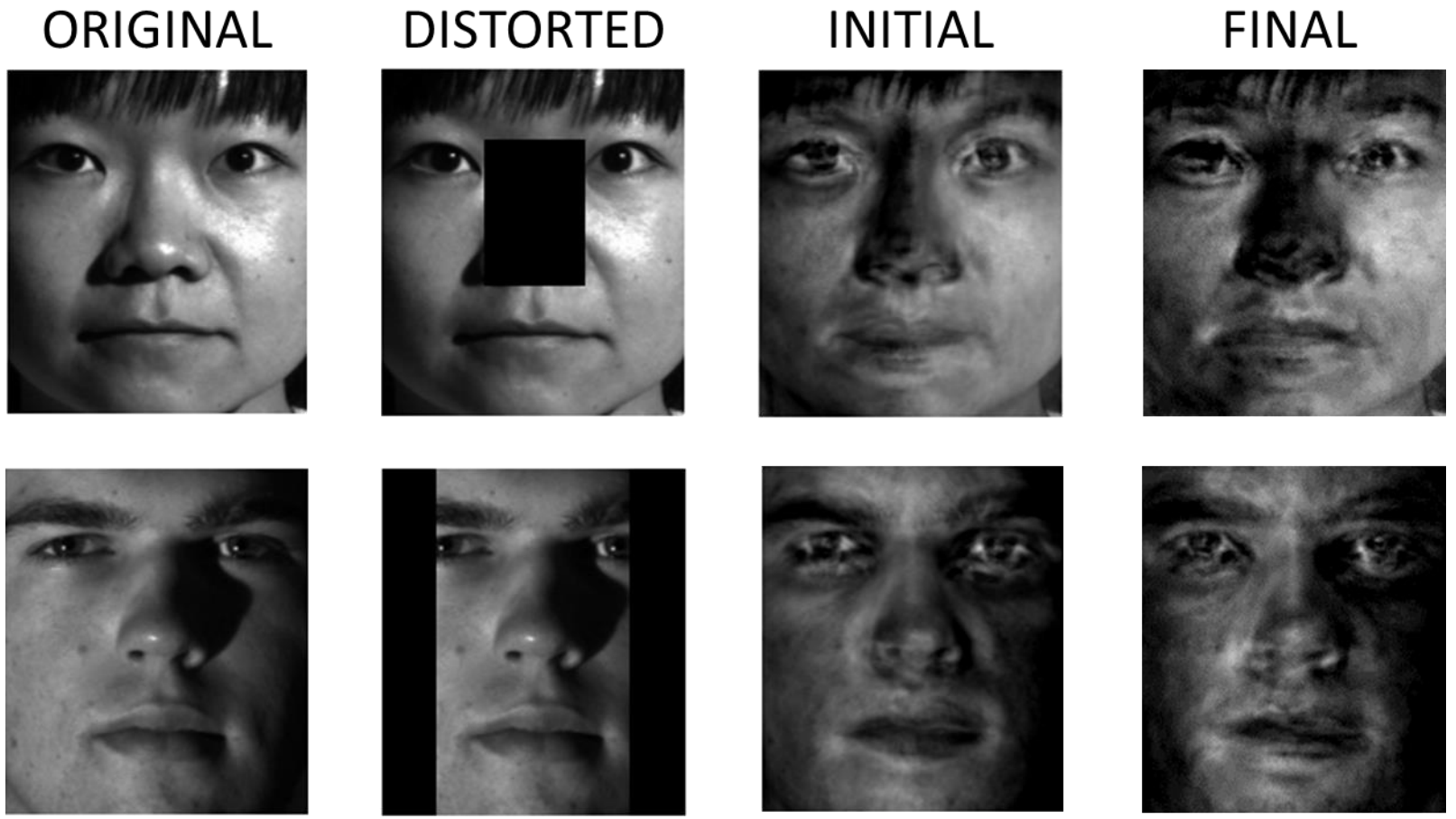


Image Reconstruction with ML



Bhatia et al. Robust Regression via Hard Thresholding, NIPS 2015

ML: Some Success Stories

- Automatic Program Correction

```
1  #include<stdio.h>
2  int main(){
3   int a;
4   scanf("%d", a );
5   printf("ans=%d",
6    a+10);
7   return 0;
8  }
```

```
1  #include<stdio.h>
2  int main(){
3   int a;
4   scanf("%d", &a );
5   printf("ans=%d",
6    a+10);
7   return 0;
8  }
```

Figure 1: Left: erroneous program, Right: fix by TRACER. The compiler message read: *Line-4, Column-9: warning: format '%d' expects argument of type 'int *', but argument 2 has type 'int'.*

```
1  #include<stdio.h>
2  int main(){
3     int x,x1,d;
4     // ...
5     d=(x-x1)(x-x1);
6     return d;
7  }
```

```
1  #include<stdio.h>
2  int main(){
3     int x,x1,d;
4     // ...
5     d=(x-x1)*(x-x1);
6     return d;
7  }
```

Figure 2: Left: erroneous program, Right: fix by TRACER. The compiler message read: *Line-5, Column-11: error: called object type 'int' is not a function or function pointer.*

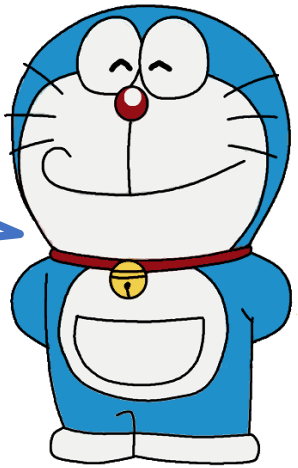
Example from "Compilation error repair: for the student programs, from the student programs", Ahmed et al (2018)

Deep Learning: Generative Models

ML in risk-sensitive problems

Human

ML is the art and science of designing adaptive algorithms. I am learning to be Fair and Unbiased over time



How can it be done?

What to do?

Good ML Systems Should be Fair and Unbiased

- Good ML should not just be about getting high accuracies

- Should also ensure that the ML models are fair and unbiased

An image captioning system should not always assume a specific gender in examples like the above

Don't want a self-driving car that is more likely to hit black people than white people

Criminals?

Not Criminals?

Don't want a predictive policing system that predicts criminality using facial features

- A lot of recent focus on Fairness and Transparency of ML systems

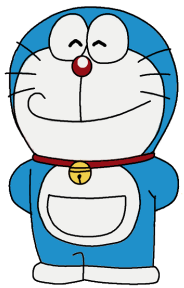
Picture courtesy: Bhargava and Forsyth (2019), https://www.thestranger.com/, Xiaolin Wu and Xi Zhang, "Automated Inference on Criminality Using Face Images"

Keep in mind: ML is like an exam

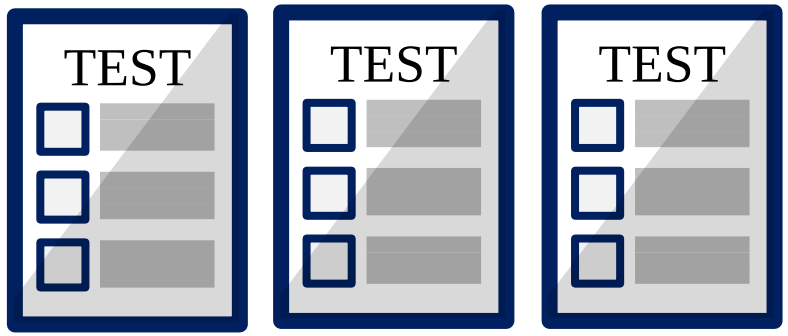
- It's the performance on the D-day which matters
- In an exam, our success is measured based on how well we did on the questions in the test (not on the questions we practiced on)
- Likewise, in ML, success of the learned model is measured based on how well it predicts/fits the future test data (not the training data)

In Machine Learning, generalization performance on the test data matters

Plus, of course, issues such as fairness, biasness



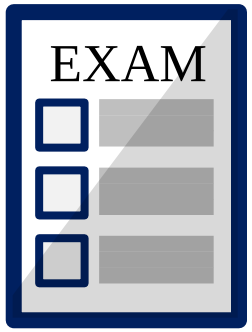
A typical study cycle (e.g. in a course)



Practice Tests/Study Material

Preparation

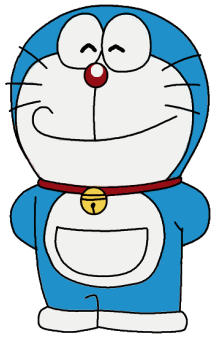
Actual Exam



Subject Knowledge

Pass / **Fail**

ML algo. as a Student like you



- Our brain stores subject matter
- Use subject matter to solve exam
- Critical to do well on exam-day
- Mock test results indicative
- No out-of-syllabus questions
- Should not leak exam paper before exam

- The model stores data patterns
- Use model to predict on test data
- Critical to do well on test data
- Training accuracies indicative
- Training/test data are similar
- Should not look at test data while training

A Loose Taxonomy of ML

Learning using **labeled** data

Learning using **unlabeled** data

Some examples of supervised learning problems

- Classification
- Regression
- Ranking

Supervised Learning

Machine Learning

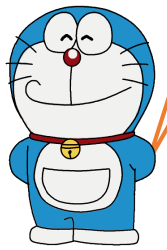
Unsupervised Learning

Some examples of unsupervised learning problems

- Clustering
- Dimensionality Reduction
- Unsupervised Probability Density Estimation

Reinforcement Learning

"Labeled" means, during training, for each input, the corresponding output is available (i.e., the machine learner is explicitly told that a cat image is of a cat)

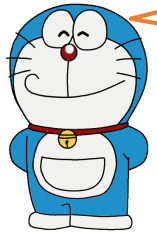


RL doesn't use "labeled" or "unlabeled" data in the traditional sense! In RL, an agent learns via its interactions with an environment

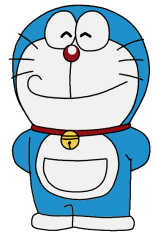
Many other specialized flavors of ML also exist, some of which include (not included in this course):

- Semi-supervised Learning
- Active Learning
- Transfer Learning
- Multitask Learning
- Imitation Learning (somewhat related to RL)
- Zero-Shot Learning
- Few-Shot Learning
- Continual learning

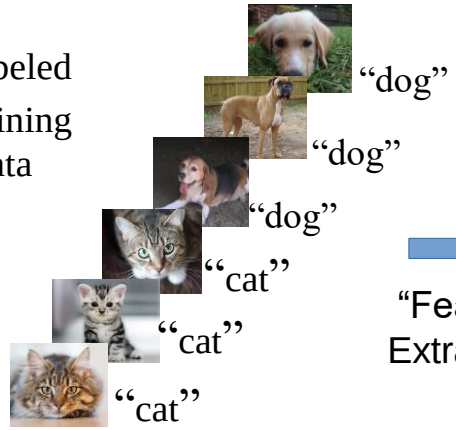
A Typical Supervised Learning Workflow



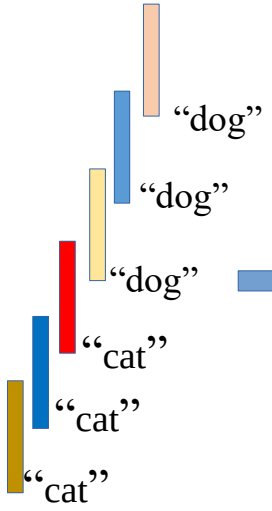
Note: This example is for the problem of binary classification, a supervised learning problem

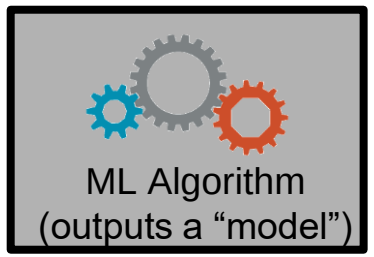


Labeled Training Data

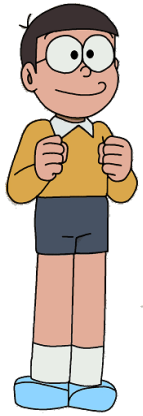


"Feature" Extraction





Is feature extraction done "**manually**" as a pre-processing step before the ML algo starts working? Can't we "automate" this part? Can't we "learn" good features directly from raw inputs?

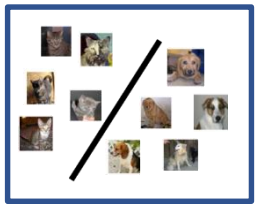


Feature extraction converts raw inputs to a numeric representation that the ML algo can understand and work with. More on feature extraction later.

Indeed. Deep Learning algos do precisely that! (feature + model learning). More on Deep Learning later.

Test Image

"Feature" Extraction



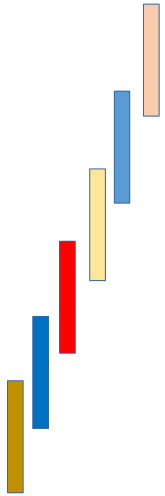
Cat vs Dog Prediction model

Predicted Label (cat/dog)

A Typical Unsupervised Learning Workflow

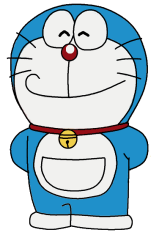
Unlabeled Data

"Feature" Extraction

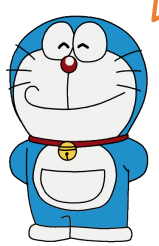


ML Algorithm (outputs a clustering)

Note: This example is for the problem of data clustering, an unsupervised learning problem

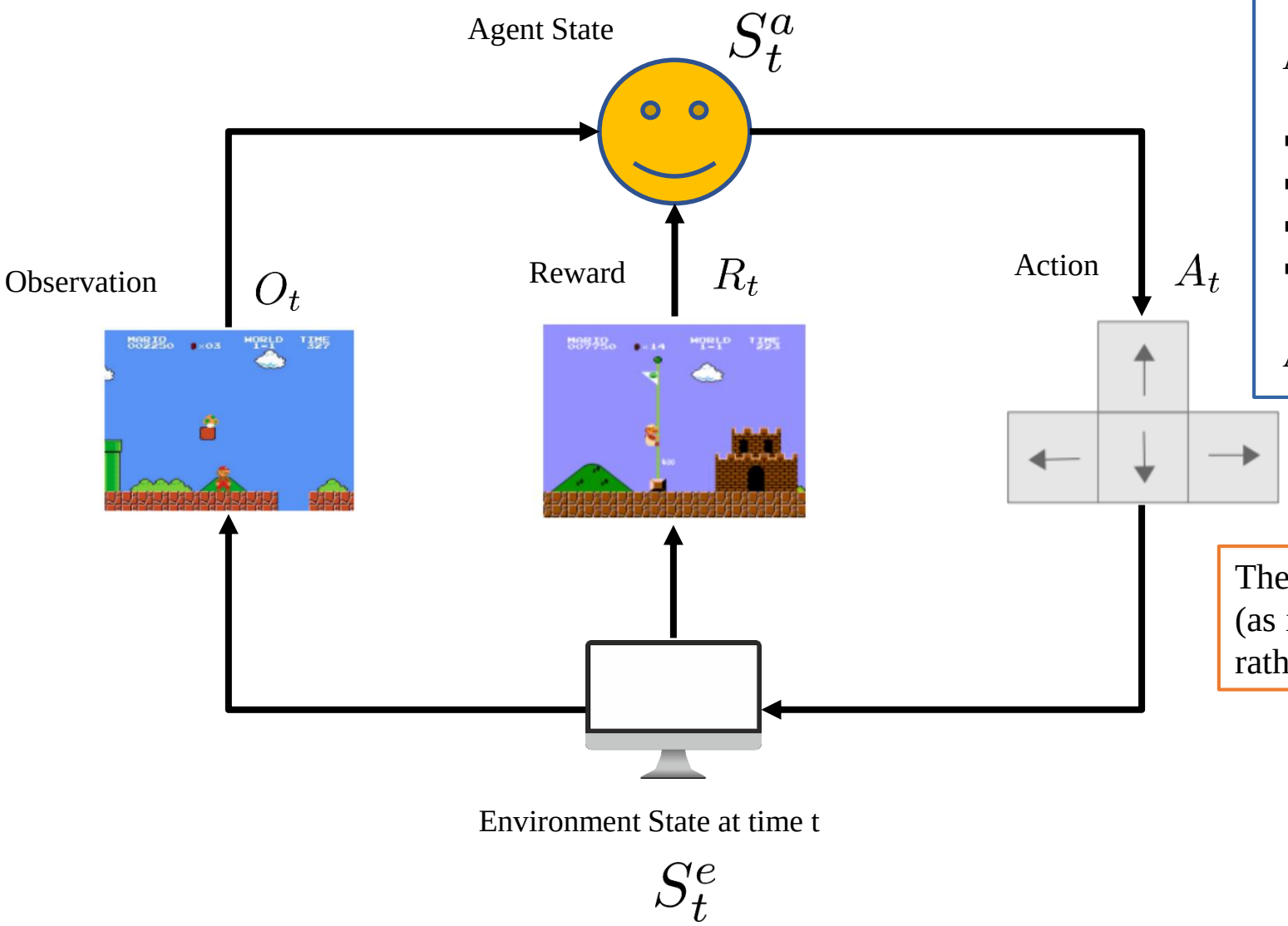


Yes. In this example, given a new "test" cat/dog image, we can assign it to the cluster with closer centroid



Does unsupervised learning also have a test phase? That is, can we also predict the cluster of a new test input?

A Typical Reinforcement Learning Workflow



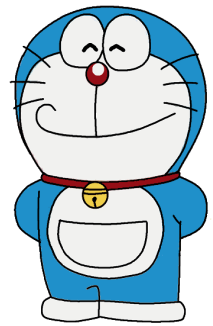
Wish to teach an agent optimal policy for some task

Agent does the following repeatedly

- Senses/observes the environment
- Takes an action based on its current policy
- Receives a reward for that action
- Updates its policy

Agent's goal is to maximize its overall reward

There is supervision, not explicit (as in Supervised Learning) but rather implicit (feedback based)

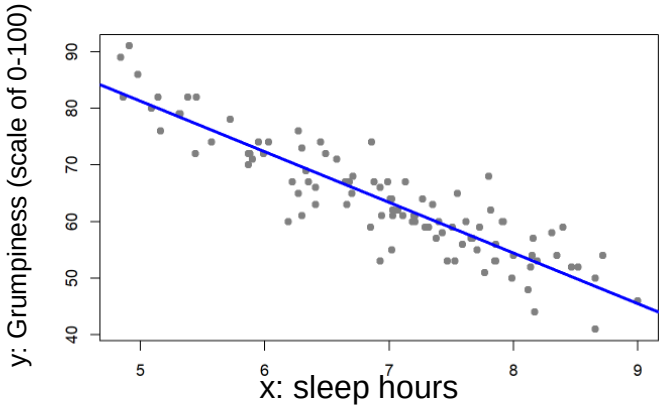


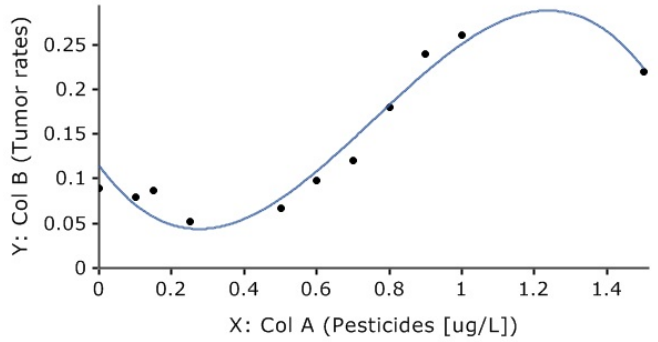
Geometric Perspective of ML

- Basic fact: Inputs in ML problems can often be represented as points or vectors in some vector space
- Doing ML on such data can thus be seen from a geometric view

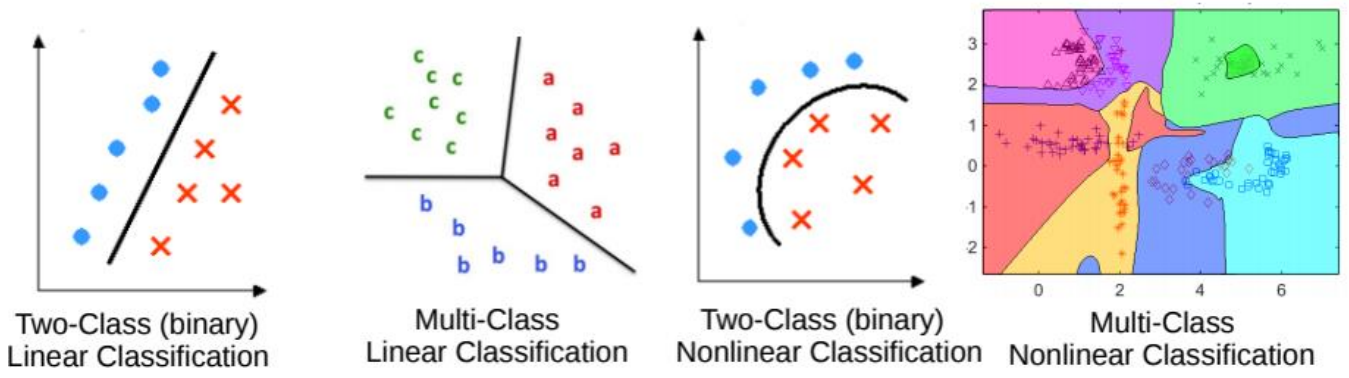
Recall that feature extraction converts inputs into a numeric representation

Regression: A supervised learning problem. Goal is to model the relationship between input (x) and real-valued output (y). This is akin to a line or curve fitting problem.





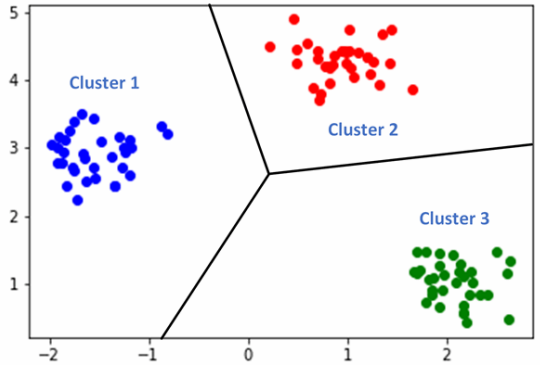
Classification: A supervised learning problem. Goal is to learn a to predict which of the two or more classes an input belongs to. Akin to learning linear/nonlinear separator for the inputs.



Pic from: https://learningstatisticswithr.com/book/regression.html, https://maxstat.de/

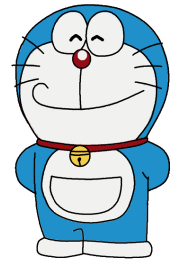
Geometric Perspective of ML

Clustering: An unsupervised learning problem. Goal is to group inputs in a few clusters based on their similarities with each other

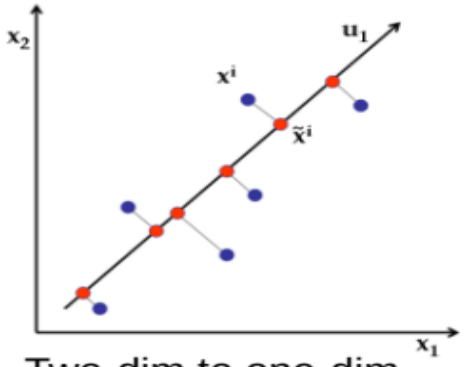


Clustering looks like classification to me. Is there any difference?

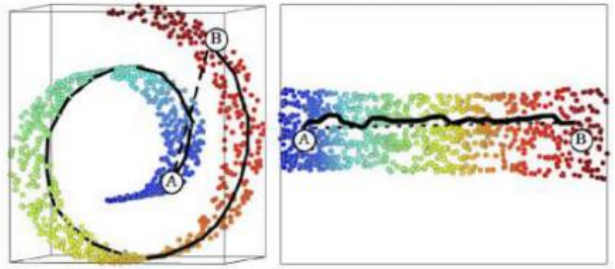
Yes. In clustering, we don't know the labels. Goal is to separate them without any labeled **"supervision"**



Dimensionality Reduction: An unsupervised learning problem. Goal is to compress the size of each input without losing much information present in the data



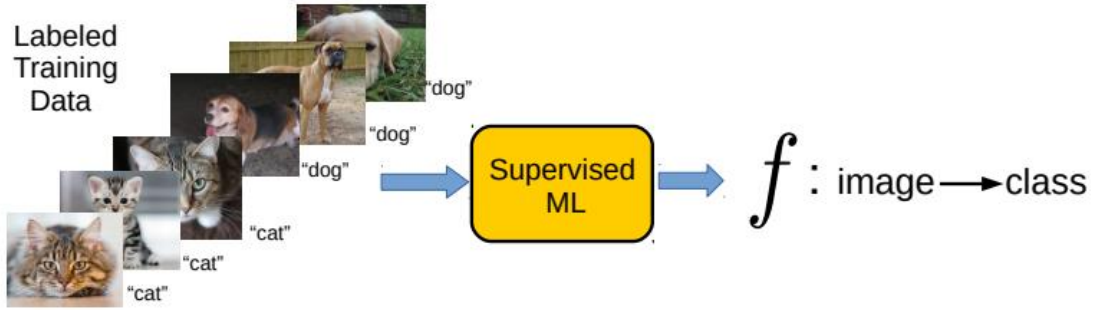
Two-dim to one-dim linear projection



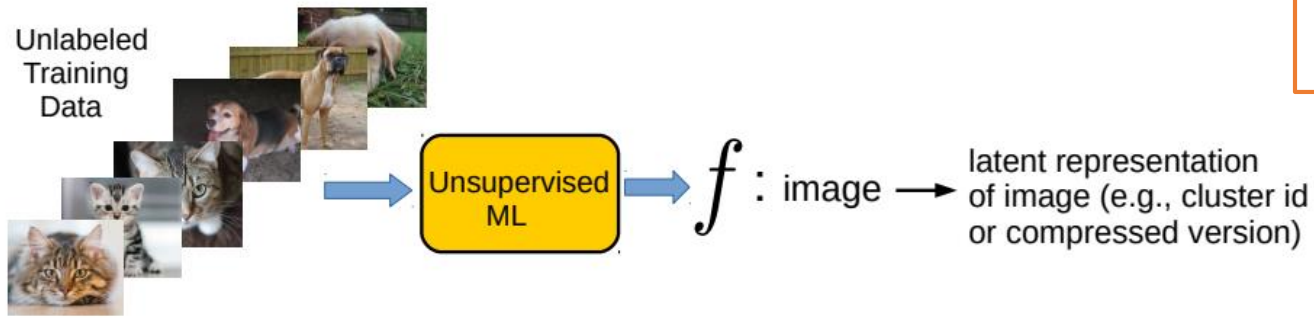
Three-dim to two-dim nonlinear projection (a.k.a. manifold learning)

Perspective as function approximation

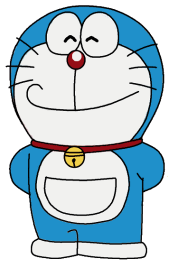
- Supervised Learning ("predict output given input") can be usually thought of as learning a function *f* that maps each input to the corresponding output



- Unsupervised Learning ("model/compress inputs") can also be usually thought of as learning a function *f* that maps each input to a compact representation



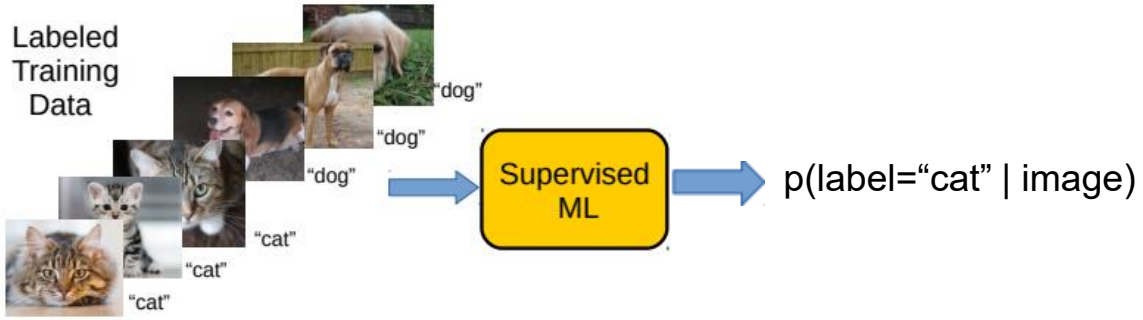
Unsupervised is harder since we don't know the labels in this case



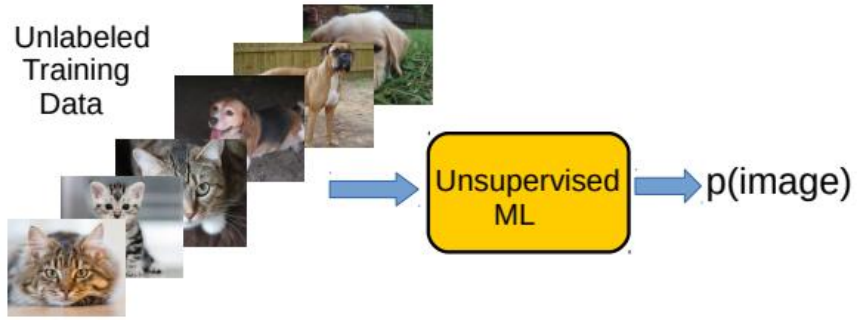
- Reinforcement Learning can also be seen as doing function approximation

Perspective as probability estimation

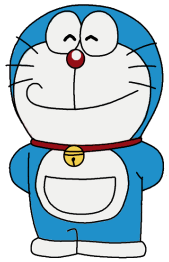
- Supervised Learning ("predict output given input") can be thought of as estimating the conditional probability of each possible output given an input



- Unsupervised Learning ("model/compress inputs") can be thought of as estimating the probability density of the inputs



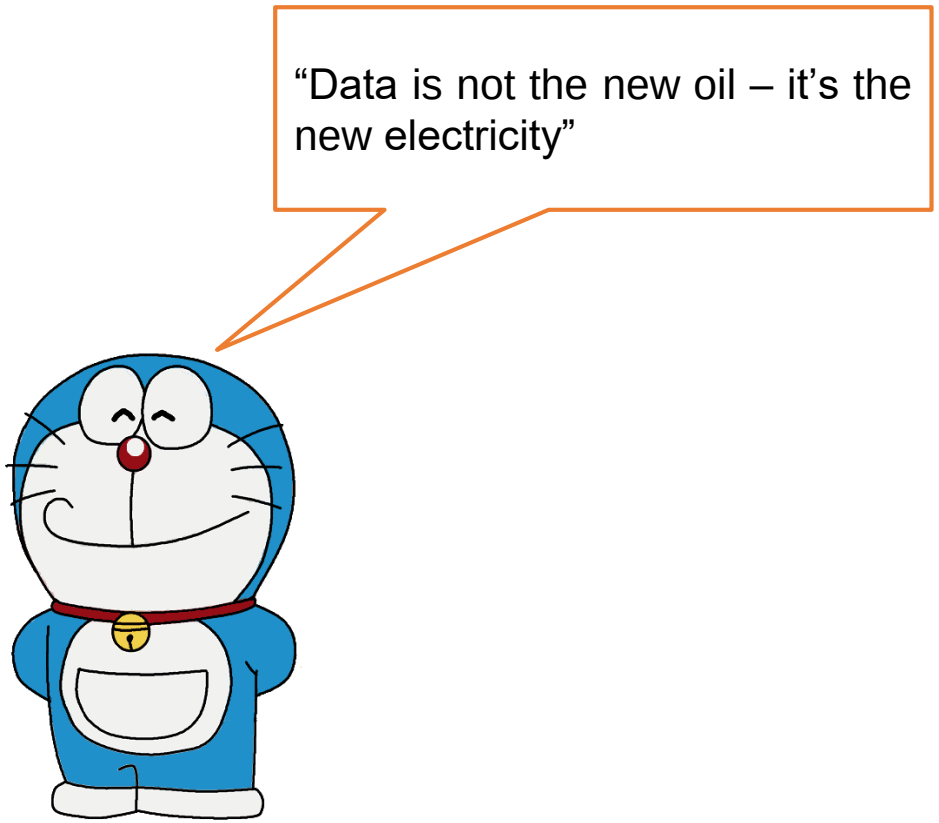
The basic idea is to learn the underlying data distribution using the unlabeled inputs; many ways to do this as we will see later

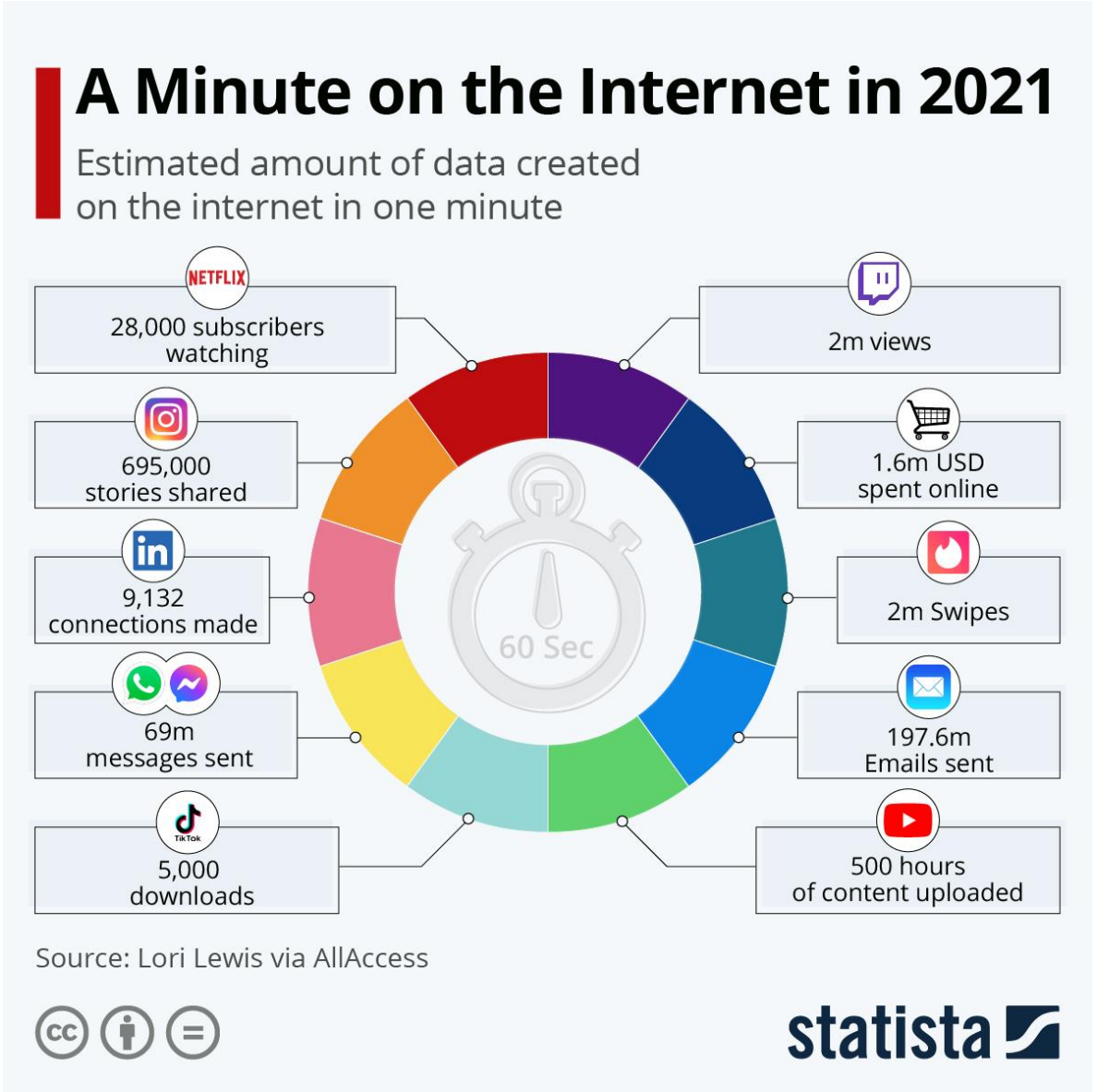


- Reinforcement Learning can also be seen as estimating probability densities

Data and Fantastic Features

Data and Features

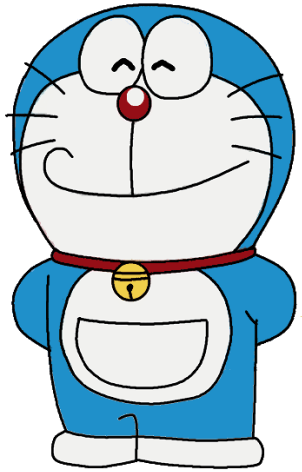




Data and Features

- ML algos require a numeric feature representation of the inputs
- Features can be obtained using one of the two approaches
  - Approach 1: Extracting/constructing features manually from raw inputs
  - Approach 2: Learning the features from raw inputs
- Approach 1 is what we will focus on primarily for now
- Approach 2 is what is followed in Deep Learning algorithms (will see later)
- Approach 1 is not as powerful as Approach 2 but still used widely

Features represent semantics of the inputs. Being able to extract good features is key to the success of ML algos



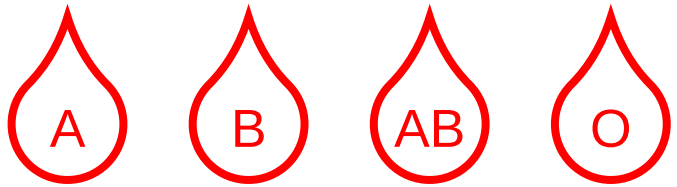
Types of Features

- Numerical Features

Titanic ★★★★★

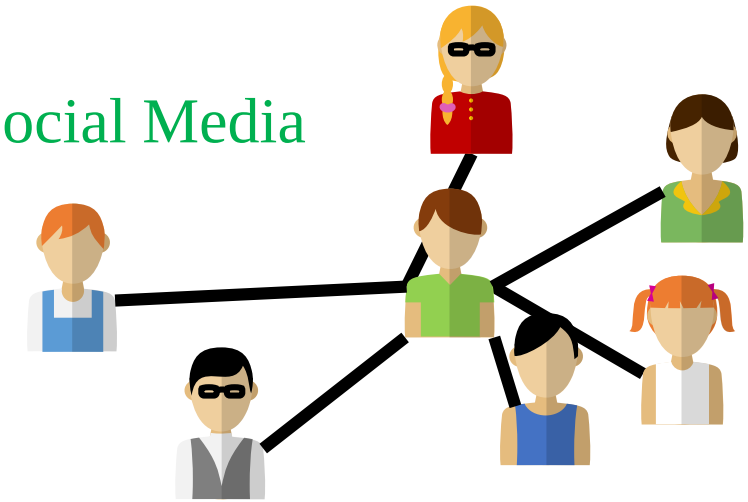
The Terminator ★★★

- Categorical Features

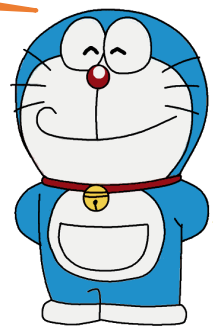


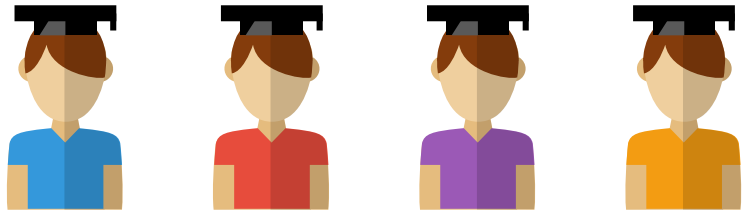
- Relational Features

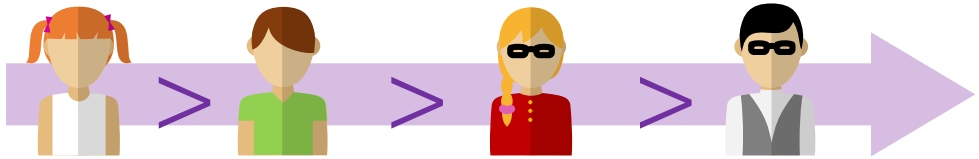
Social Media



Discrete numerical features
often called *ordinal* features







Ranking in Class

Types of Features and Types of Outputs

- Features as well as outputs can be real-valued, binary, categorical, ordinal, etc.
- Real-valued: Pixel intensity, house area, house price, rainfall amount, temperature, etc
- Binary: Male/female, adult/non-adult, or any yes/no or present/absent type value
- Categorical/Discrete: Zipcode, blood-group, or any “one from a finite many choices“ value
- Ordinal: Grade (A/B/C etc.) in a course, or any other type where relative values matter
- Often, the features can be of mixed types (some real, some categorical, some ordinal, etc.)

What are features?

- Features are a way for us to give input to ML algorithms.
- Most ML algorithms use numerical vectors to represent features.
- For example, in spam classification, every old email (training data) as well as every new email (test data) must be converted into a vector.

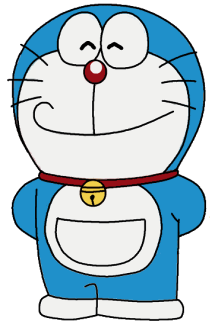
Do you want to go for dinner?

Do you want to win a million dollars?

I have a million things to do today!

Do	You	Want	Go	Million	Dollars	Dinner	Today
1	1	1	1	0	0	1	0
1	1	1	0	1	1	0	0
1	0	0	0	1	0	0	1

Since I am basically a computer program, I need you to convert your data into a nice set of numbers

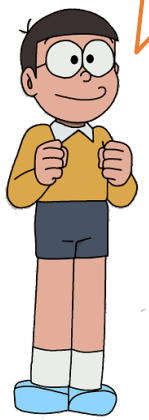


What are features?

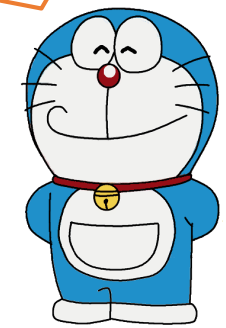
Why did we not keep the word “to” as a feature?

We could have – but it does not carry much information about spam/non-spam since it is such a common word!

Guys, something is wrong with our feature. The word “do” means different things in two emails



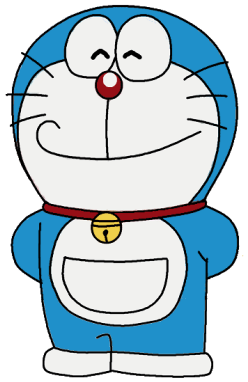
Good catch – this may not be the best feature representation!



Example: Feature Extraction for another Text Data

- Consider some text data consisting of the following sentences:
  - John likes to watch movies
  - Mary likes movies too
  - John also likes football
- Want to construct a feature representation for these sentences
- Here is a "bag-of-words" (BoW) feature representation of these sentences

BoW is just one of the many ways of doing feature extraction for text data. Not the most optimal one, and has various flaws (can you think of some?), but often works reasonably well

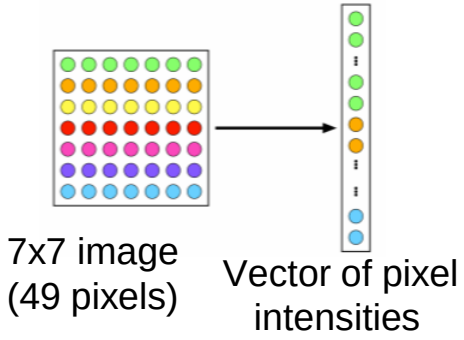


	John	likes	to	watch	movies	Mary	too	also	football
Sentence 1	1	1	1	1	1	0	0	0	0
Sentence 2	0	1	0	0	1	1	1	0	0
Sentence 3	1	1	0	0	0	0	0	1	1

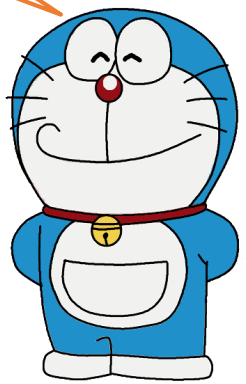
- Each sentence is now represented as a binary vector (each feature is a binary value, denoting presence or absence of a word). BoW is also called "unigram" rep.

Example: Feature Extraction for Image Data

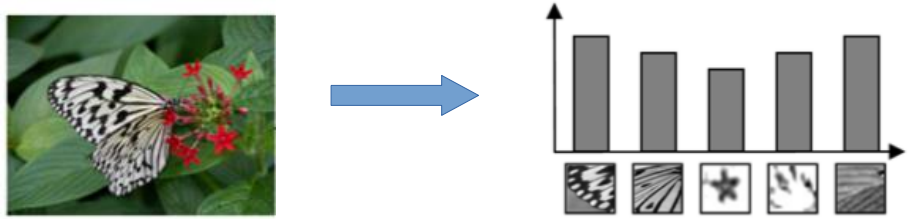
- A very simple feature extraction approach for image data is flattening



Flattening and histogram based methods destroy the spatial information in the image but often still work reasonably well



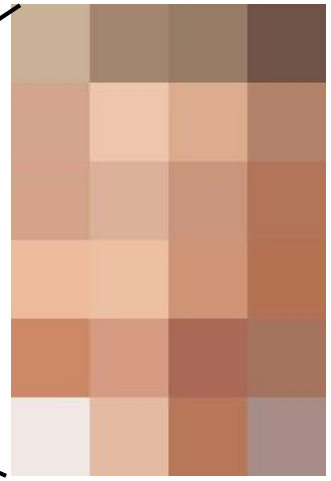
- Histogram of visual patterns is another popular feature extr. method for images



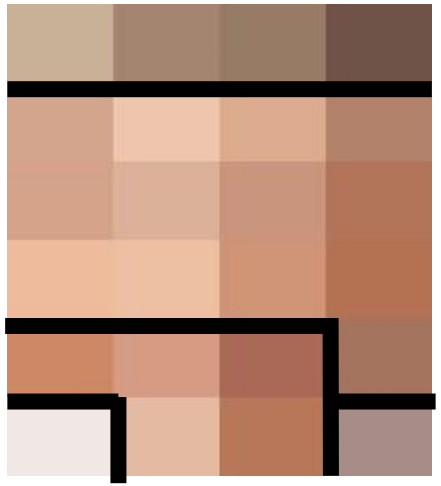
- Many other manual feature extraction techniques developed in computer vision and image processing communities (SIFT, HoG, and others)

Pic credit: cat.uab.cat/Research/object-recognition

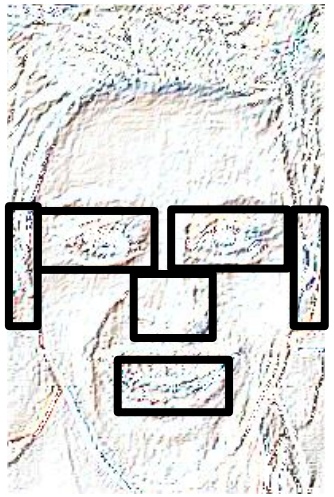
What are features in Ellen's face?



May represent images as a vector/matrix of pixel RGB values



May instead encode images using edges present in them



May instead encode images using presence of eyes/noses/ears

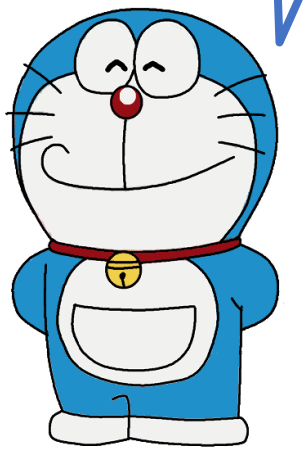
Makes sense. Features are also a way for us to store what we know about data. If we throw away data, naturally doing anything becomes hard!

Raw / Low-level features

⇩

Derived / High- level features

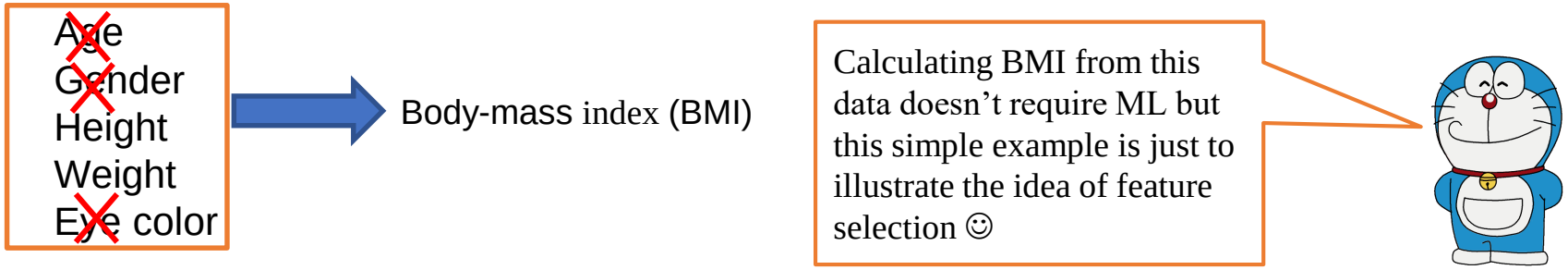
If your features are good, my job becomes twice as easy. If not, then doing any ML may become impossible!



Feature Selection

- Not all the extracted features may be relevant for learning the model (some may even confuse the learner)

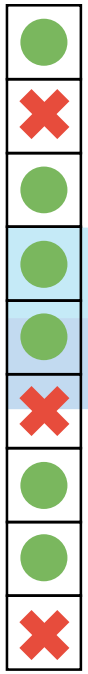
- Feature selection (a step after feature extraction) can be used to identify the features that matter, and discard the others, for more effective learning



- Many techniques exist – some based on intuition, some based on algorithmic principles (will visit feature selection later)

- More common in supervised learning but can also be done for unsupervised learning

Example: Feature Selection



DIET (VEG/NON-VEG)

EYE COLOR

TOTAL INCOME

EDUCATION LEVEL

FAMILY SIZE

HOUSE NO (ODD/EVEN)

NO OF CHILDREN

RENTED/SELF OWNED

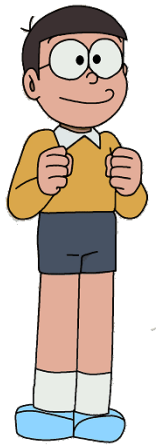
SURNAME LENGTH

Useful for predicting expenditure

Not useful for predicting expenditure

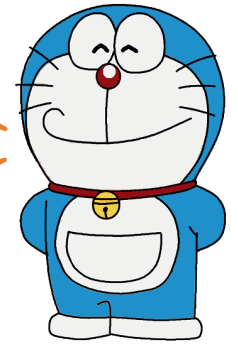
(EXPENDITURE)

In fact, one reason for the success of deep learning is that it learns good features themselves!



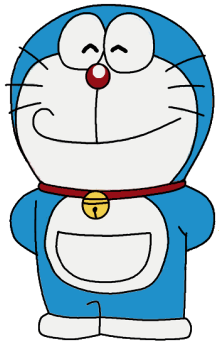
Can we perform automatic feature selection?

Yes indeed, but those methods are a bit advanced for now!



True, but more on deep learning later! For now, back to basics.

A bit of caution with features

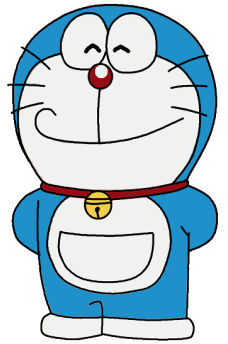


- Tricks, mnemonics lessen cognitive load, increase speed
- Easy questions can be solved in one step with a mnemonic!
- Too many mnemonics can confuse you at time of exam

- Derived features make learning easier, faster at test
- What you are trying to predict is just another feature of the data!
- Too many useless features can confuse classifier

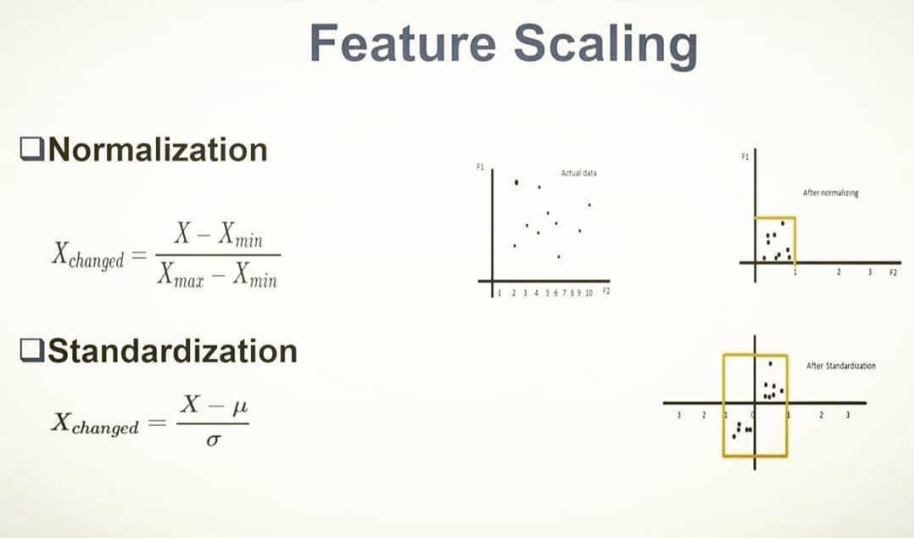
However, not to worry. For most of this course, we will give you pre-made feature vectors ☺

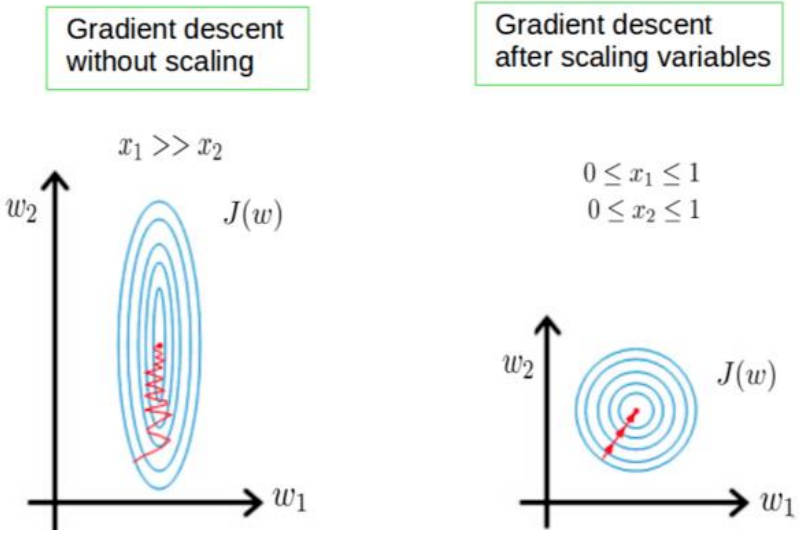
In fact, one of the main challenges in deep learning is that it learns way too many features



Some More Postprocessing: Feature Scaling

- Even after feature selection, the features may not be on the same scale
- This can be problematic when comparing two inputs – features that have larger scales may dominate the result of such comparisons
- Therefore helpful to standardize the features (e.g., by bringing all of them on the same scale such as between 0 to 1)



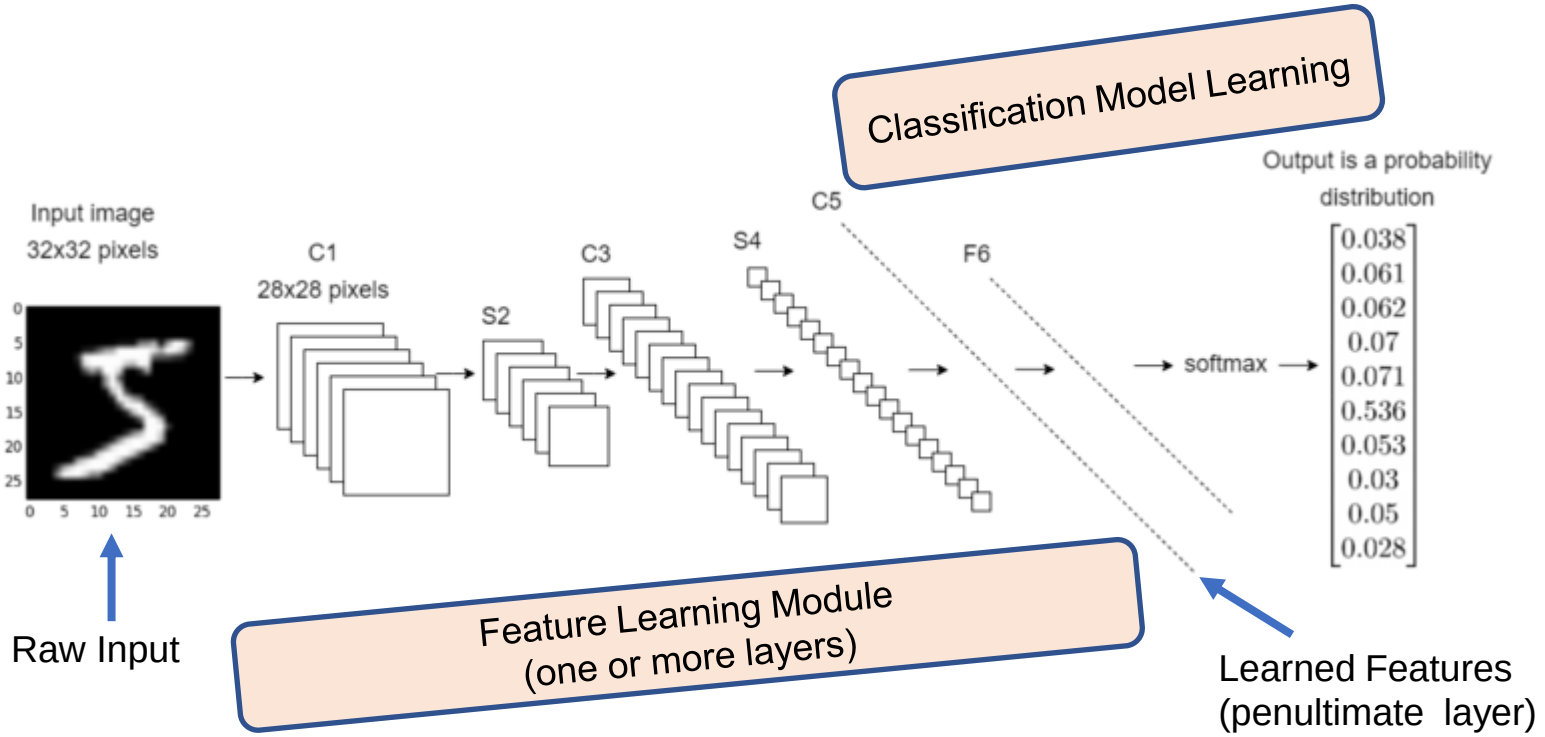


- Also helpful for stabilizing the optimization techniques used in ML algos

Deep Learning: An End-to-End Approach to ML

Deep Learning = ML with automated feature learning from the raw inputs

Feature extraction part is automated via the feature learning module

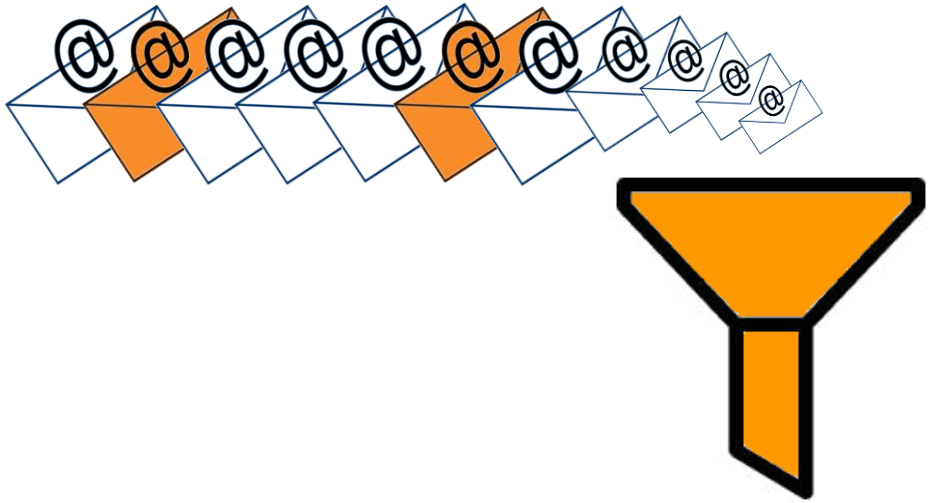


Pic an adaptation of the original from: https://deepai.org/

Spam Filtering: A simple case

- Suppose Mary has already tagged several old emails as spam/non-spam, can we tag her new emails too?

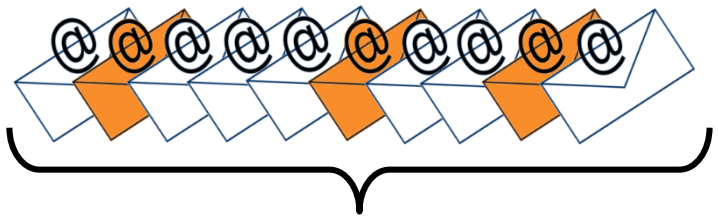
- **Trick**: use the old tagged emails to try and understand what sort of emails does Mary think of as spam and which as non-spam!



- E.g. may find that emails about shopping always tagged as spam
- E.g. may find that emails from Jill are never tagged as spam
- These insights/patterns are what are stored in the spam filter
- Our spam filter helps us make predictions on new emails

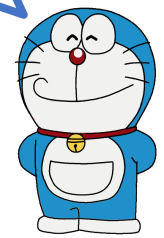
A typical ML workflow

Ah! That is the fun (and artistic) bit about ML. We will learn tons of ways on how ML algos. store patterns



Old Emails (Training data)

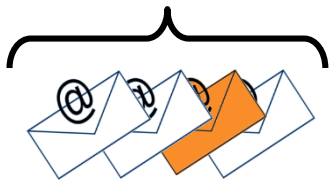
ML Algorithm



The spam filter stores information about your personal preferences about what looks spam to you

How is this information stored?

New Emails (Test data)



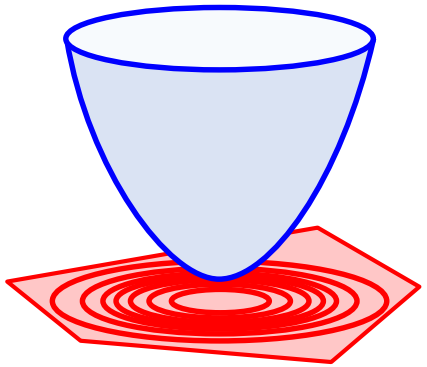
Spam Filter

Spam/Non-spam (Output)

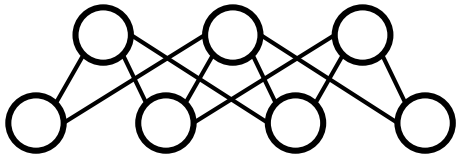
ML can store info in lots of innovative ways

ML Models and Algorithms

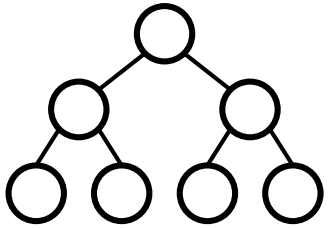
Linear/Optimization



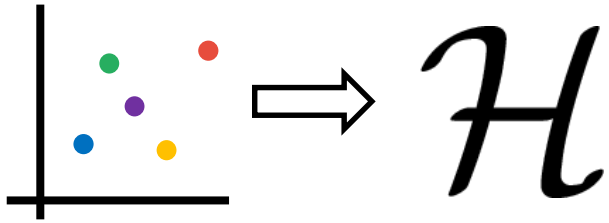
Neural/Deep



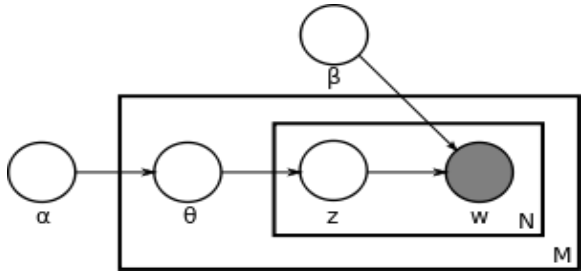
Local



Kernel

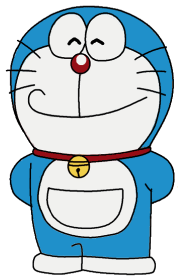


Probabilistic/Bayesian



We can mix-n-match these methods too e.g. Bayesian Deep Learning or Kernel Nearest Neighbours (Local)

Correct! But we will not be able to cover such advanced methods either

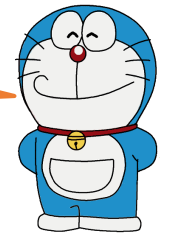


Some Notation/Nomenclature/Convention

- Sup. learning requires training data as N input-output pairs $\{(\mathbf{x_n}, y_n)\}_{n=1}^{N}$

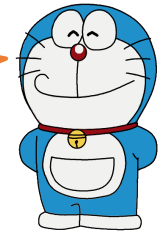
- Unsupervised learning requires training data as N inputs $\{\mathbf{x_n}\}_{n=1}^{N}$

RL and other flavors of ML problems also use similar notation



- Each input $\mathbf{x_n}$ is (usually) a vector containing the values of the features or attributes or covariates that encode properties of the it represents, e.g.,
  - For a 7 × 7 image: $\mathbf{x_n}$ can be a 49 × 1 vector of pixel intensities

Size or length of the input $\mathbf{x_n}$ is commonly known as data/input dimensionality or feature dimensionality



- (In sup. Learning) Each y_n is the output or response or label associated with input $\mathbf{x_n}$ (and its value is known for the training inputs)
  - Output can be a scalar, a vector of numbers, or even an structured object (more on this later)

Some Basic Operations of Inputs

- Assume each input feature vector $\mathbf{x}_n \in R^D$ to of size D

- Given N inputs $\{\mathbf{x}_n\}_{n=1,}^{N}$ their average or mean can be computed as

$$\boldsymbol{\mu} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n$$

- Can compute the Euclidean distance between any pair of inputs $\mathbf{x}_n$ and $\mathbf{x}_m$

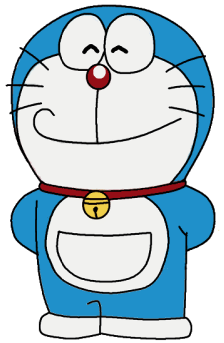
$$d(\boldsymbol{x}_n, \boldsymbol{x}_m) = ||\boldsymbol{x}_n - \boldsymbol{x}_m|| = \sqrt{(\boldsymbol{x}_n - \boldsymbol{x}_m)^\top(\boldsymbol{x}_n - \boldsymbol{x}_m)} = \sqrt{\sum_{d=1}^{D}(x_{nd} - x_{md})^2}$$

- .. or Euclidean distance between an input $\mathbf{x}_n$ and the mean $\boldsymbol{\mu}$ of all inputs

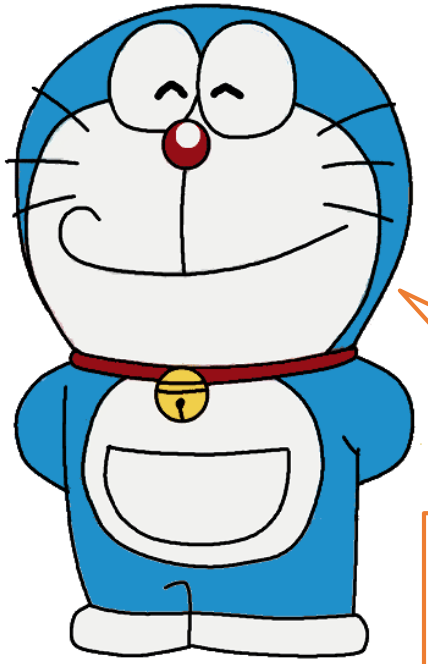
- .. and various other operations that we will look at later..

What does such a "mean" represent?

If inputs are all cat images, mean vector would represents what an "average" cat looks like



Any question?



Readings for you:

- Machine learning Overview: Science Paper (2015)
- Deep Learning Overview: Nature Paper (2015)
- A short history of ML (2016) and a short history of Neural Networks and Deep Learning (2020)