Warming-up to Machine Learning, Data and Features

CS771: Introduction to Machine Learning

Piyush Rai

Plan for today

- Types of ML problems
- Typical workflow of ML problems
- Various perspectives of ML problems
- Data and Features
- Some basic operations of data and features

Keep in mind: ML is like an exam

Keep in mind: ML is like an exam

- It’s the performance on the D-day which matters

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)

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)

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)

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

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)

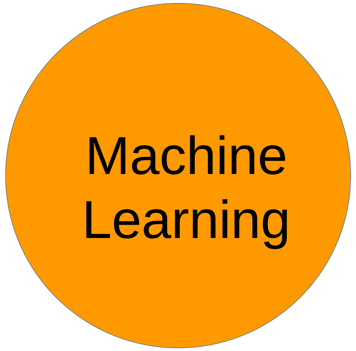
- Likewise, in ML, success of the learned model is measured based on how well it predicts/fits the future test data (not the training data)

Plus, of course, issues such as fairness

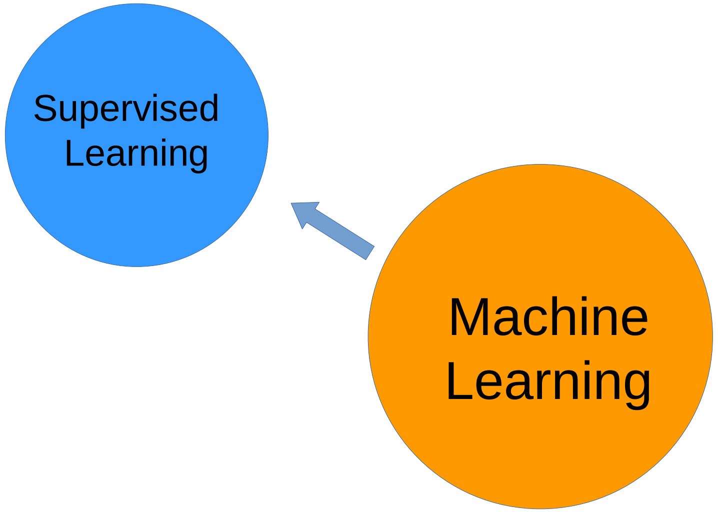
In Machine Learning, generalization performance on the test data matters

A Loose Taxonomy of ML

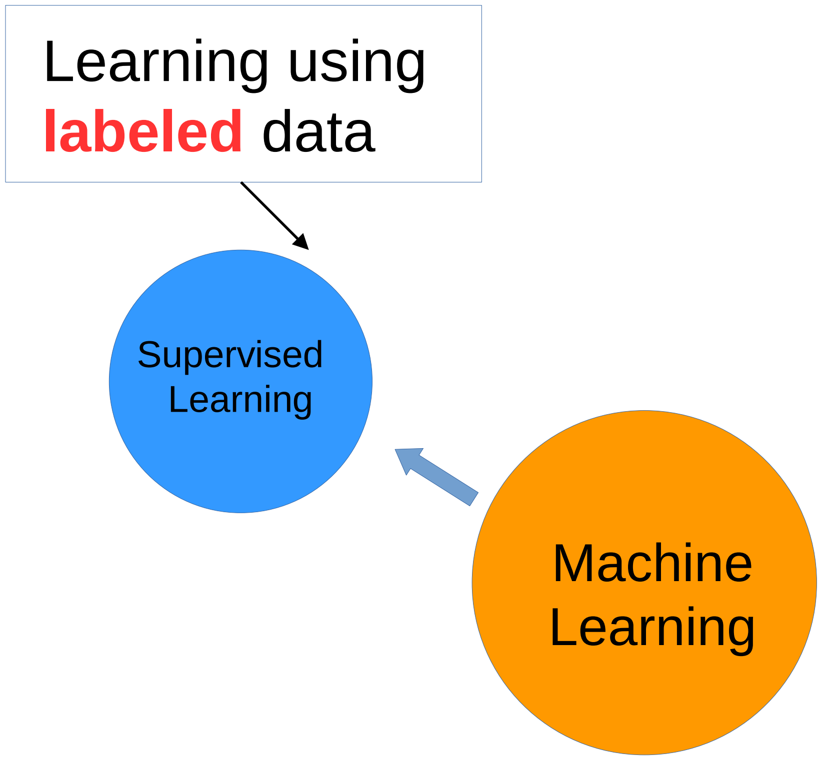
A Loose Taxonomy of ML



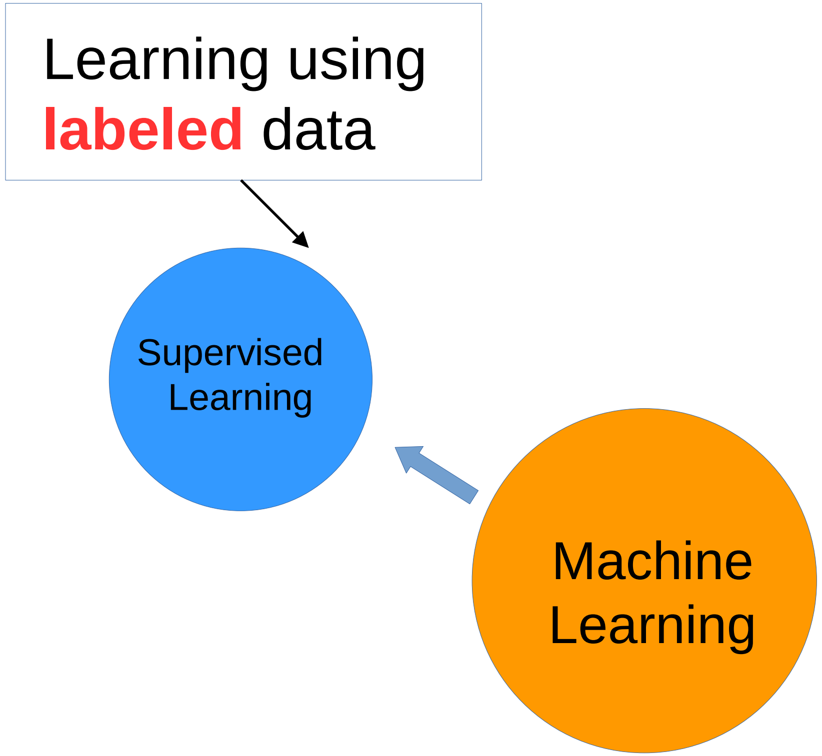
A Loose Taxonomy of ML



A Loose Taxonomy of ML



A Loose Taxonomy of ML



"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)

A Loose Taxonomy of ML

Learning using **labeled** data

Some examples of supervised learning problems

- Classification
- Regression
- Ranking

Supervised Learning

Machine 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)

A Loose Taxonomy of ML

Learning using **labeled** data

Some examples of supervised learning problems

- Classification
- Regression
- Ranking

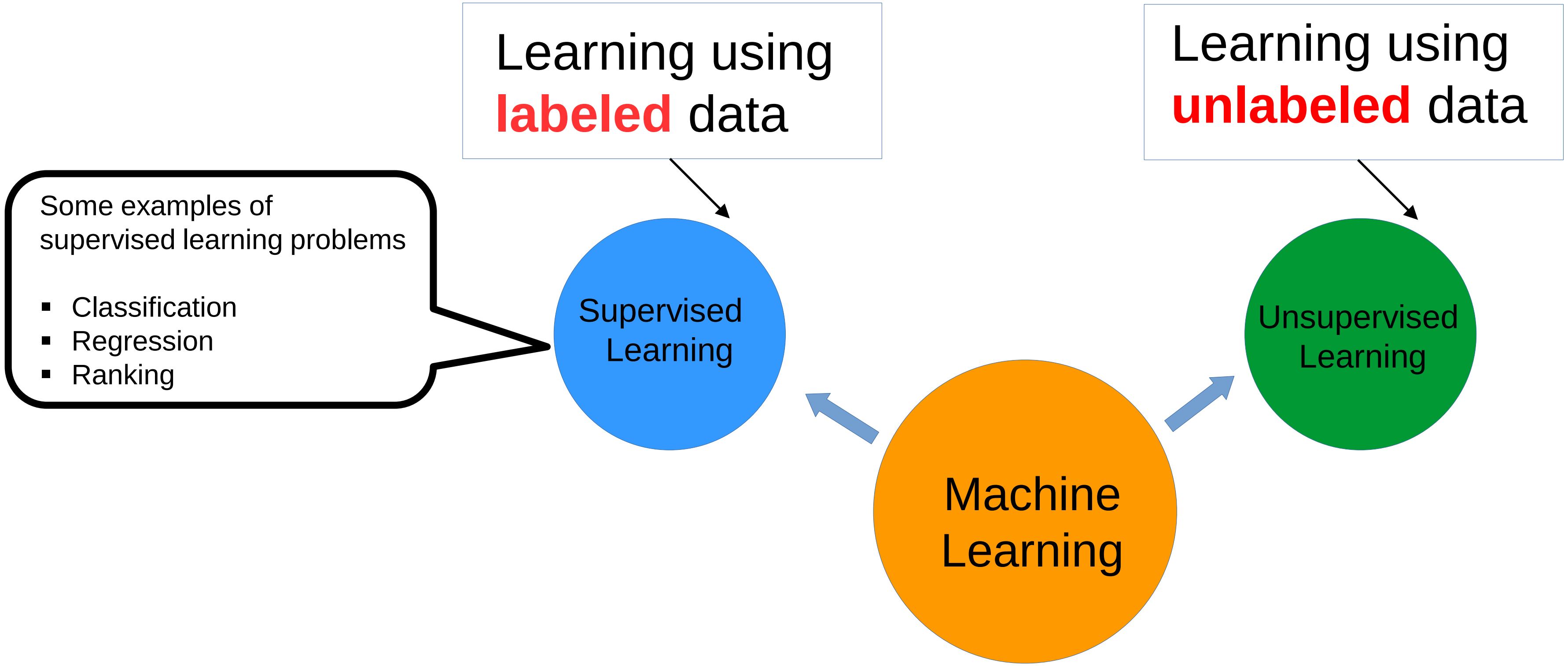
Supervised Learning

Machine Learning

Unsupervised 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)

A Loose Taxonomy of ML



"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)

A Loose Taxonomy of ML

Learning using **labeled** data

Learning using **unlabeled** data

“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)

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

A Loose Taxonomy of ML

Learning using **labeled** data

Learning using **unlabeled** data

"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)

Some examples of supervised learning problems

- Classification
- Regression
- Ranking

Supervised Learning

Machine Learning

Unsupervised Learning

Reinforcement Learning

Some examples of unsupervised learning problems

- Clustering
- Dimensionality Reduction
- Unsupervised Probability Density Estimation

A Loose Taxonomy of ML

Learning using **labeled** data

Learning using **unlabeled** data

“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)

Some examples of supervised learning problems

- Classification
- Regression
- Ranking

Supervised Learning

Unsupervised Learning

Some examples of unsupervised learning problems

- Clustering
- Dimensionality Reduction
- Unsupervised Probability Density Estimation

Machine Learning

Reinforcement Learning

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

A Loose Taxonomy of ML

Learning using **labeled** data

Some examples of supervised learning problems

- Classification
- Regression
- Ranking

Supervised Learning

Machine Learning

Learning using **unlabeled** data

“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)

Unsupervised Learning

Some examples of unsupervised learning problems

- Clustering
- Dimensionality Reduction
- Unsupervised Probability Density Estimation

Reinforcement Learning

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

- 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

A Typical Supervised Learning Workflow

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

A Typical Supervised Learning Workflow

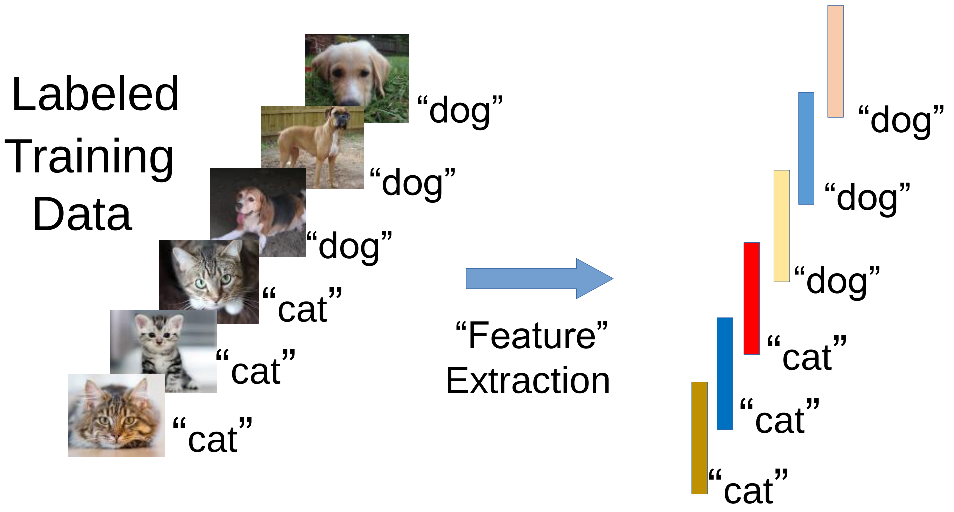
Note: This example is for the problem of binary classification, a supervised learning problem

A Typical Supervised Learning Workflow

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

Labeled Training Data

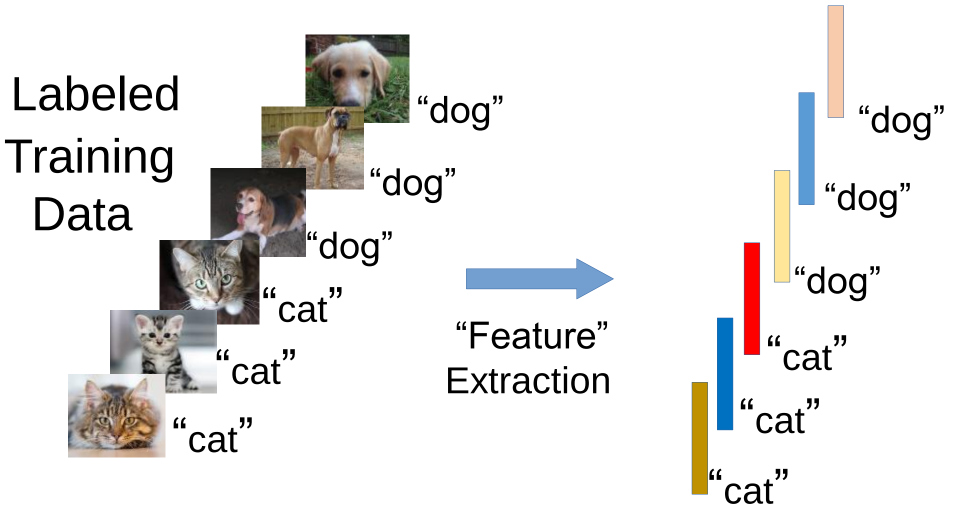
A Typical Supervised Learning Workflow



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

A Typical Supervised Learning Workflow

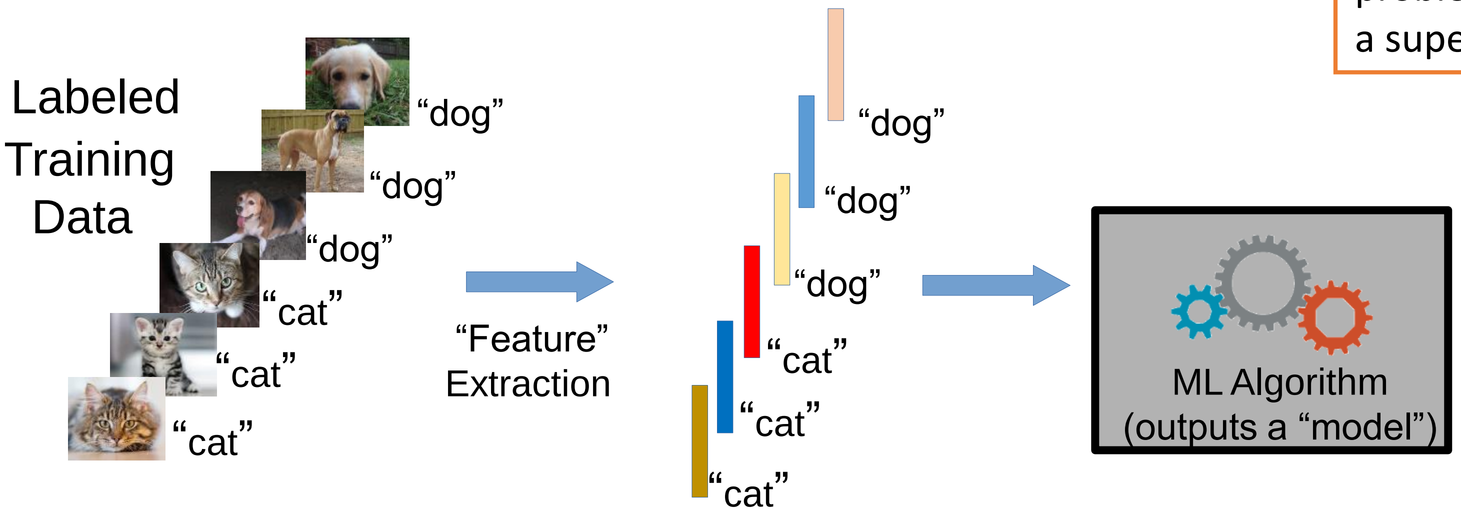
Note: This example is for the problem of binary classification, a supervised learning problem



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

A Typical Supervised Learning Workflow

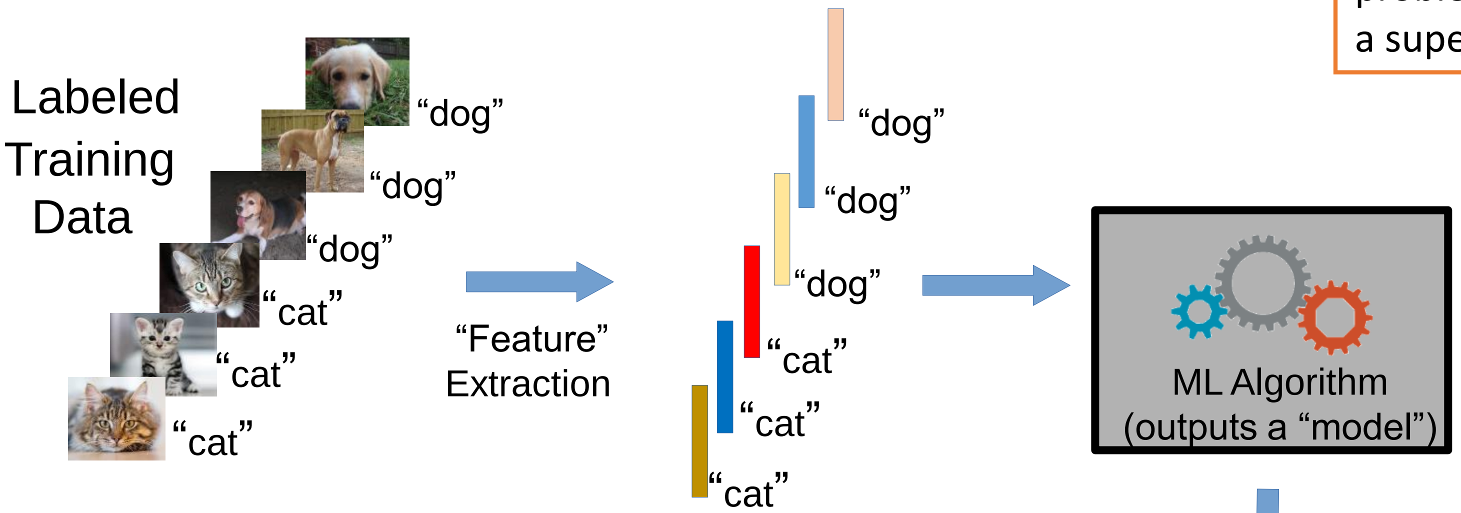
Note: This example is for the problem of binary classification, a supervised learning problem

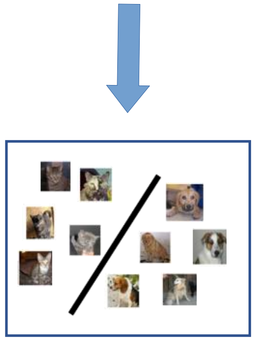


Feature extraction converts raw inputs to a numeric representation that the ML algo can understand and work with. More on feature extraction later.

A Typical Supervised Learning Workflow

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





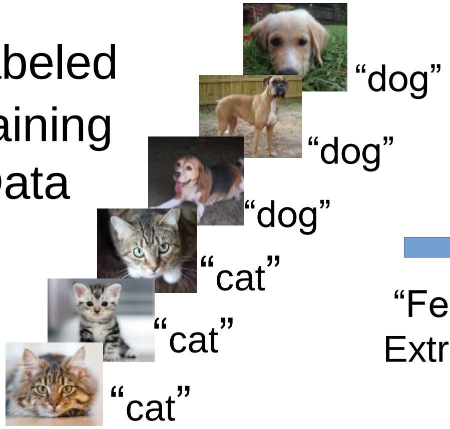
Cat vs Dog Prediction model

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

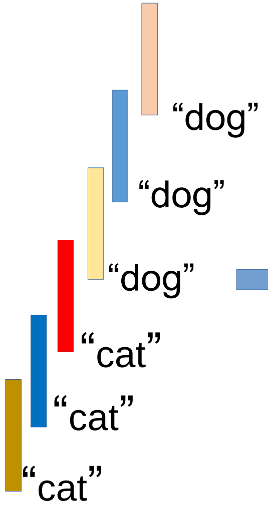
A Typical Supervised Learning Workflow

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

Labeled Training Data



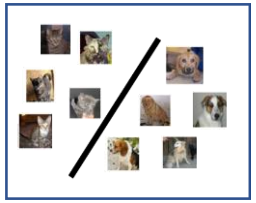
"Feature" Extraction



ML Algorithm (outputs a "model")

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

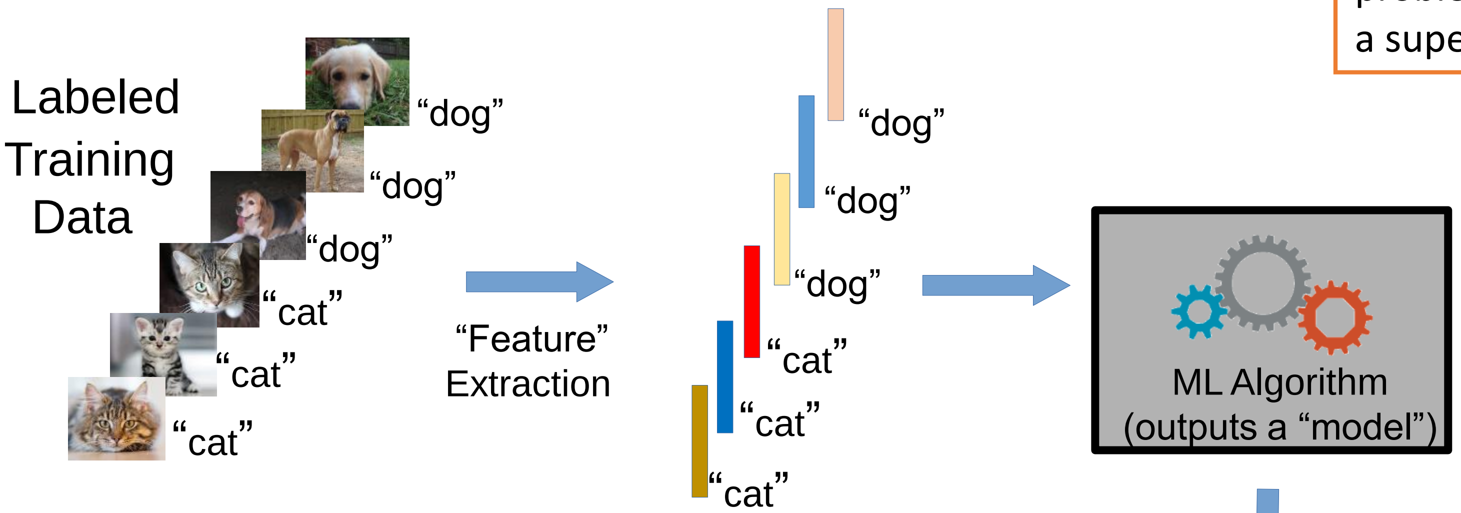
Test Image



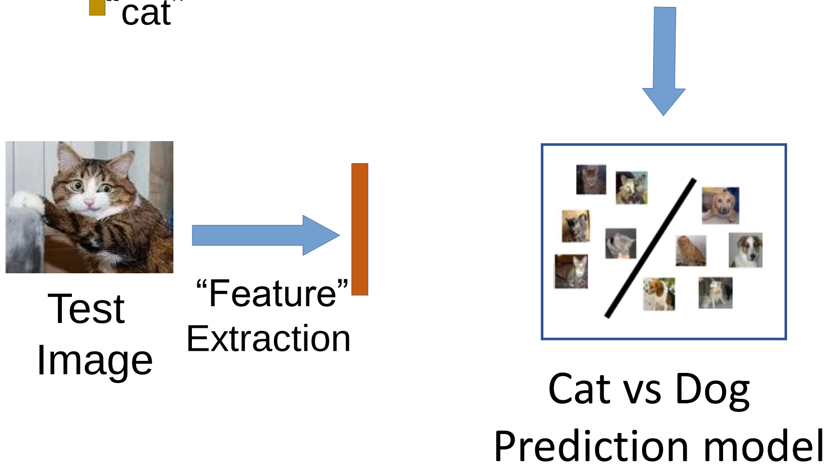
Cat vs Dog Prediction model

A Typical Supervised Learning Workflow

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

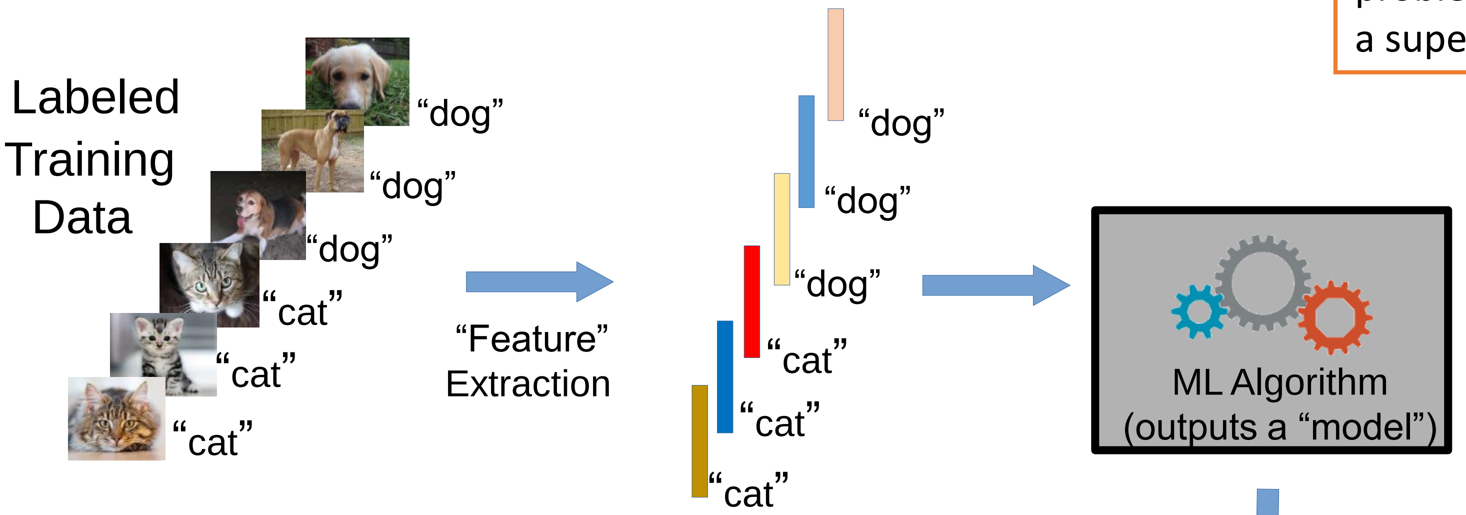


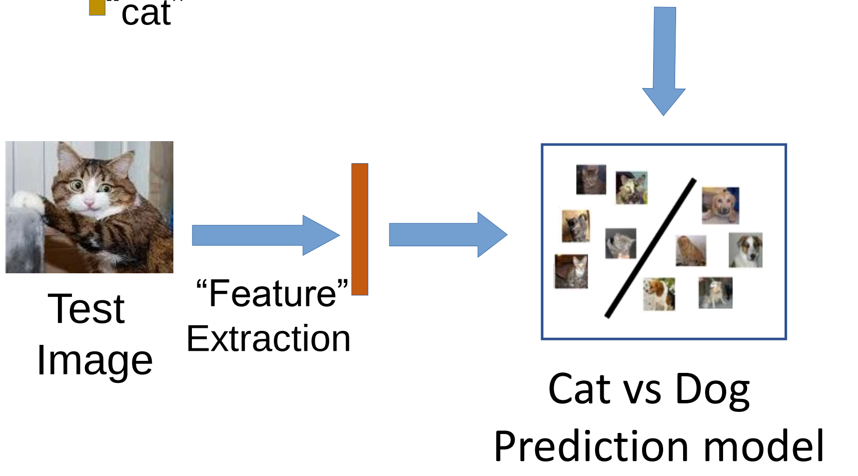
Feature extraction converts raw inputs to a numeric representation that the ML algo can understand and work with. More on feature extraction later.



A Typical Supervised Learning Workflow

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

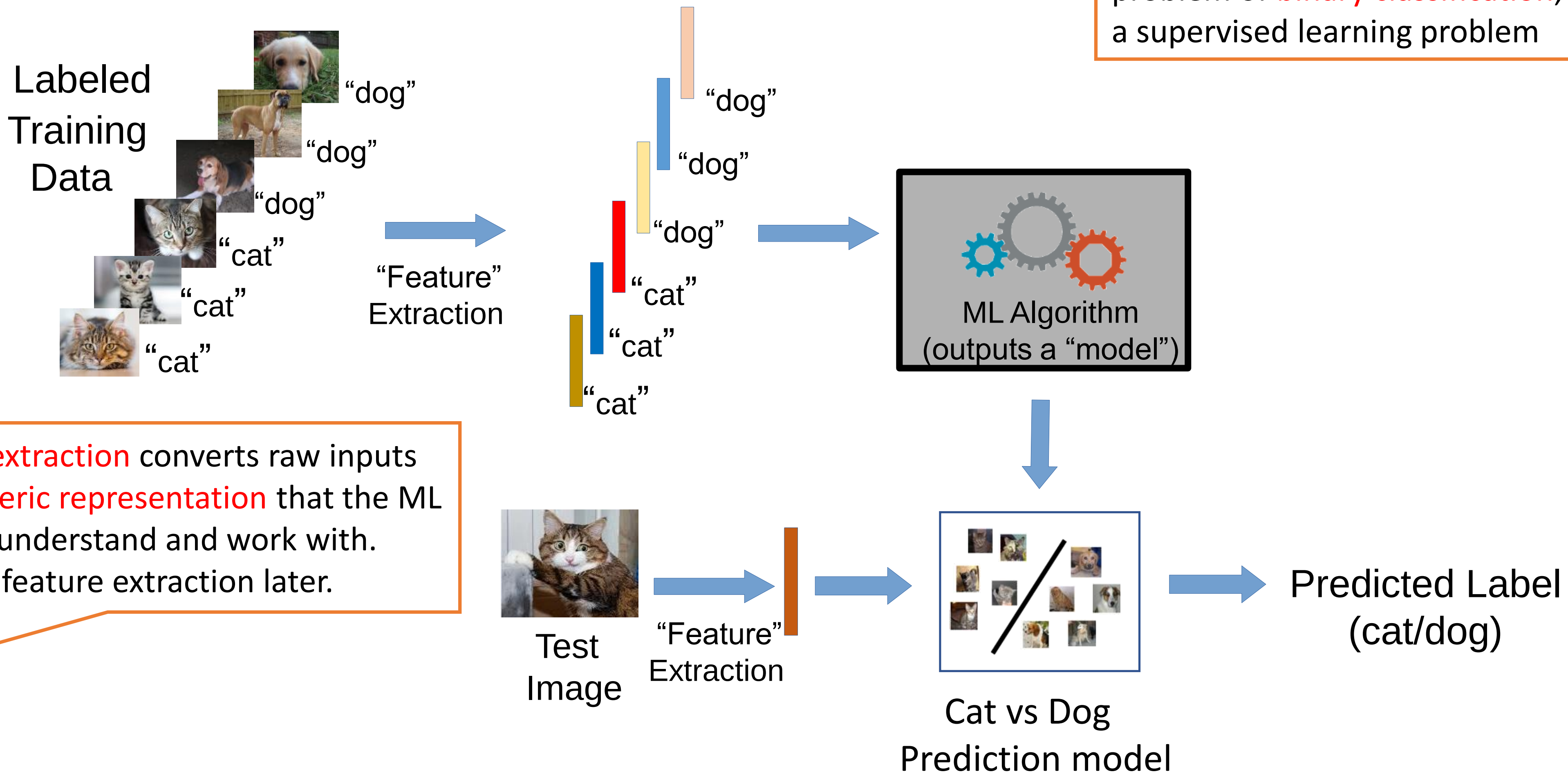




Feature extraction converts raw inputs to a numeric representation that the ML algo can understand and work with. More on feature extraction later.

A Typical Supervised Learning Workflow

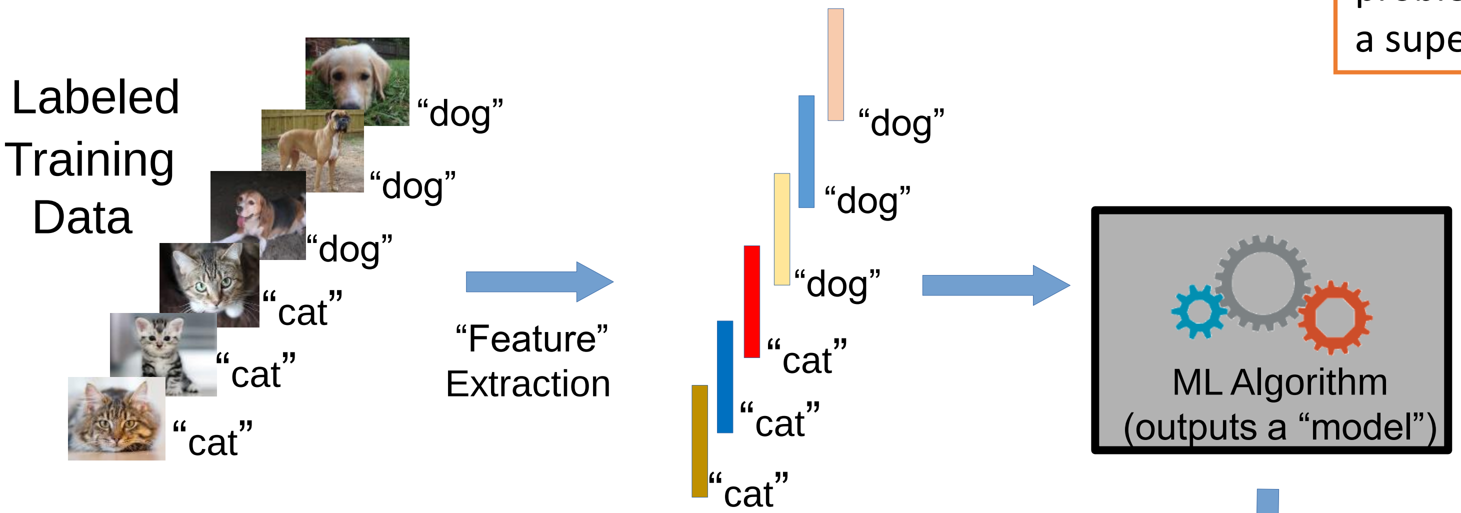
Note: This example is for the problem of binary classification, a supervised learning problem

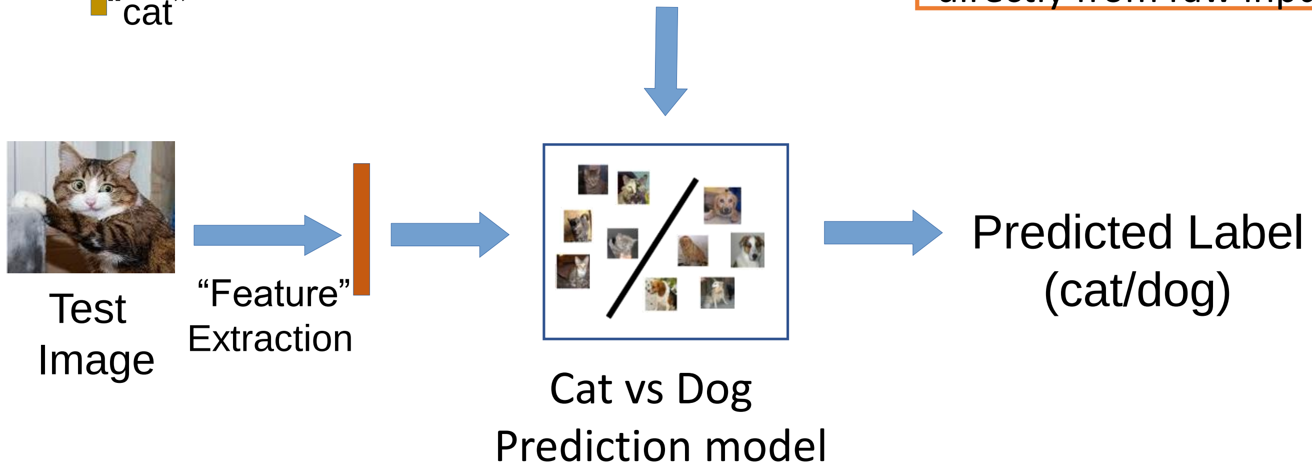


Feature extraction converts raw inputs to a numeric representation that the ML algo can understand and work with. More on feature extraction later.

A Typical Supervised Learning Workflow

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



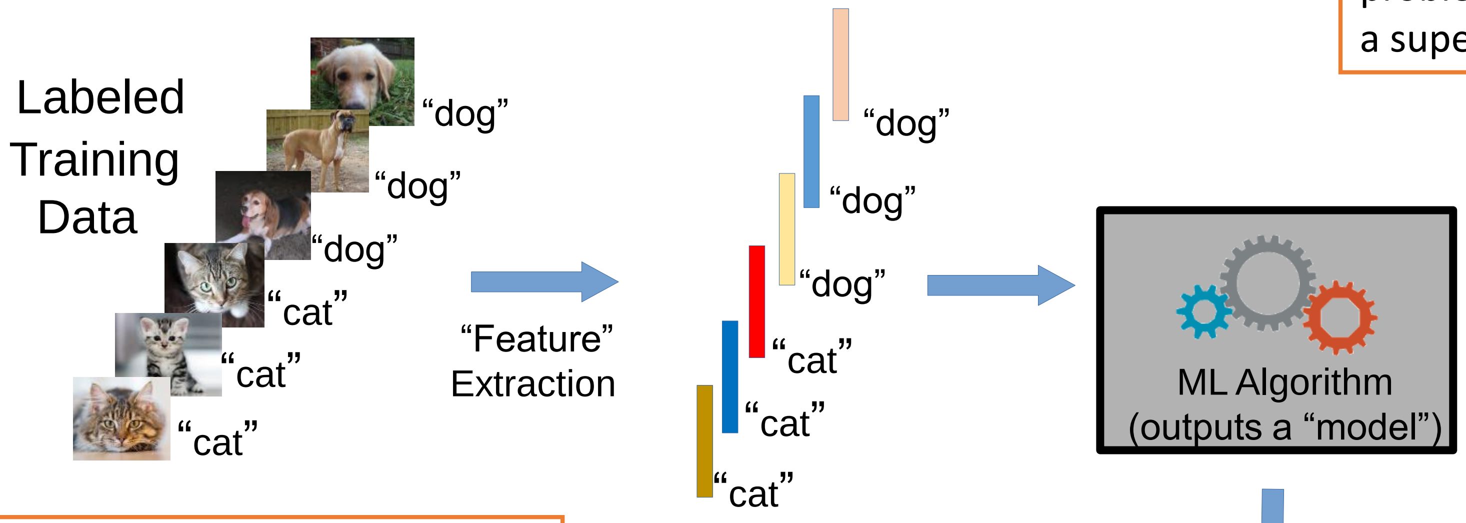


Is feature extraction done "manually" as a pre-processing step before the ML algo starts working? Can't we "automate" this part? Can't we "learn" good features directly from raw inputs?

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

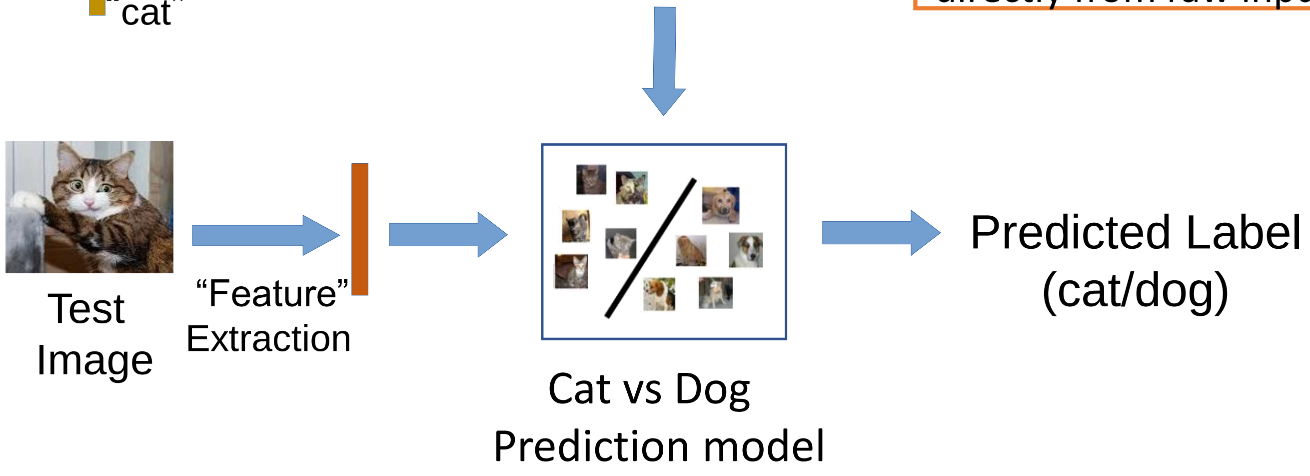
A Typical Supervised Learning Workflow

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



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?

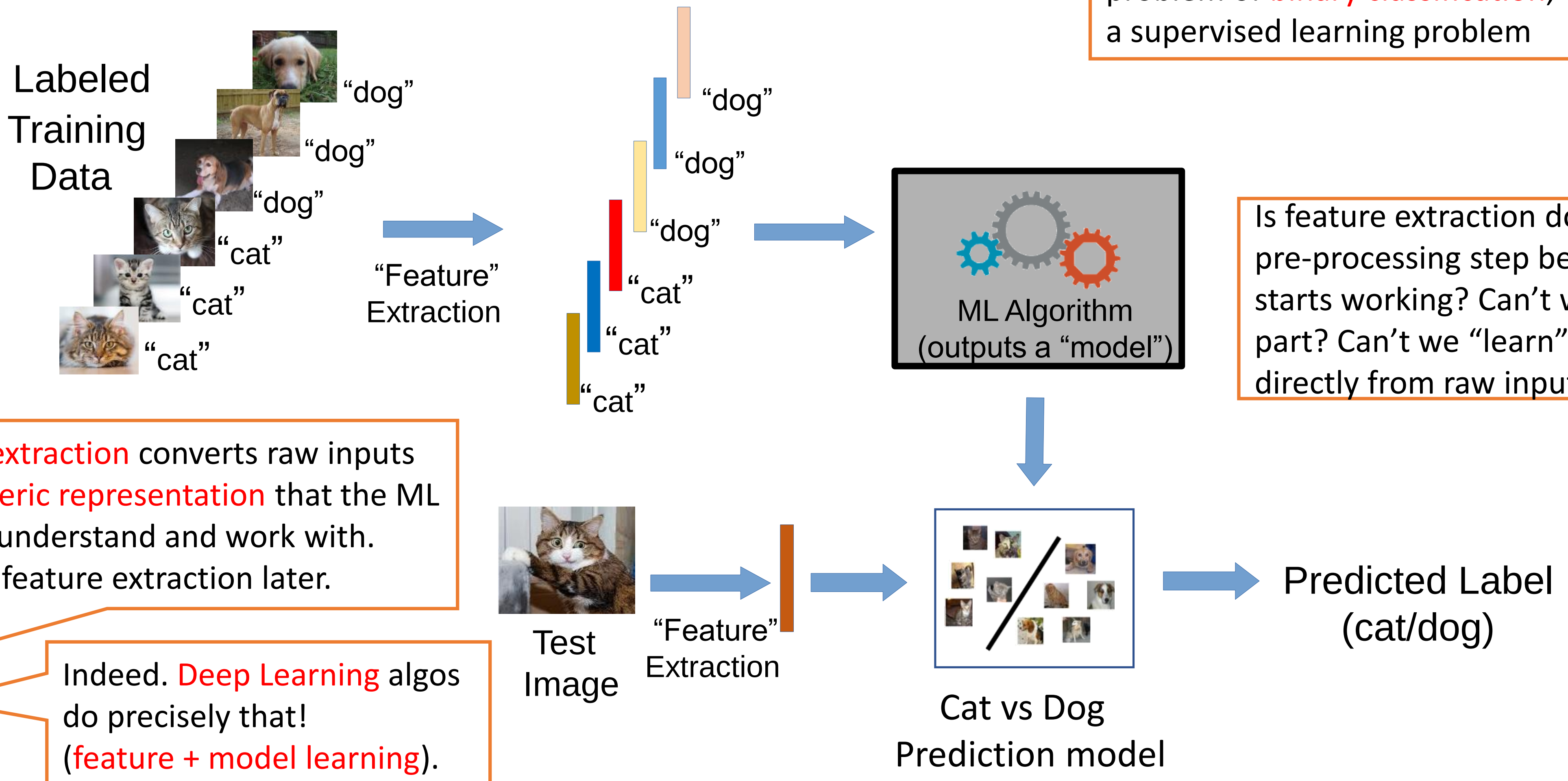
Feature extraction converts raw inputs to a numeric representation that the ML algo can understand and work with. More on feature extraction later.



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

A Typical Supervised Learning Workflow

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



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.

https://www.pinclipart.com/, http://www.pngtree.com

A Typical Unsupervised Learning Workflow

A Typical Unsupervised Learning Workflow

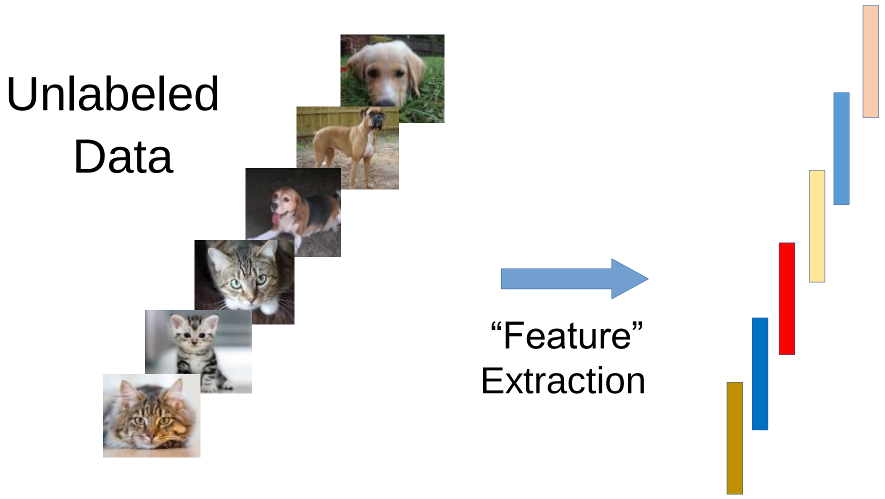
Note: This example is for the problem of data clustering, an unsupervised learning problem

A Typical Unsupervised Learning Workflow

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

Unlabeled Data

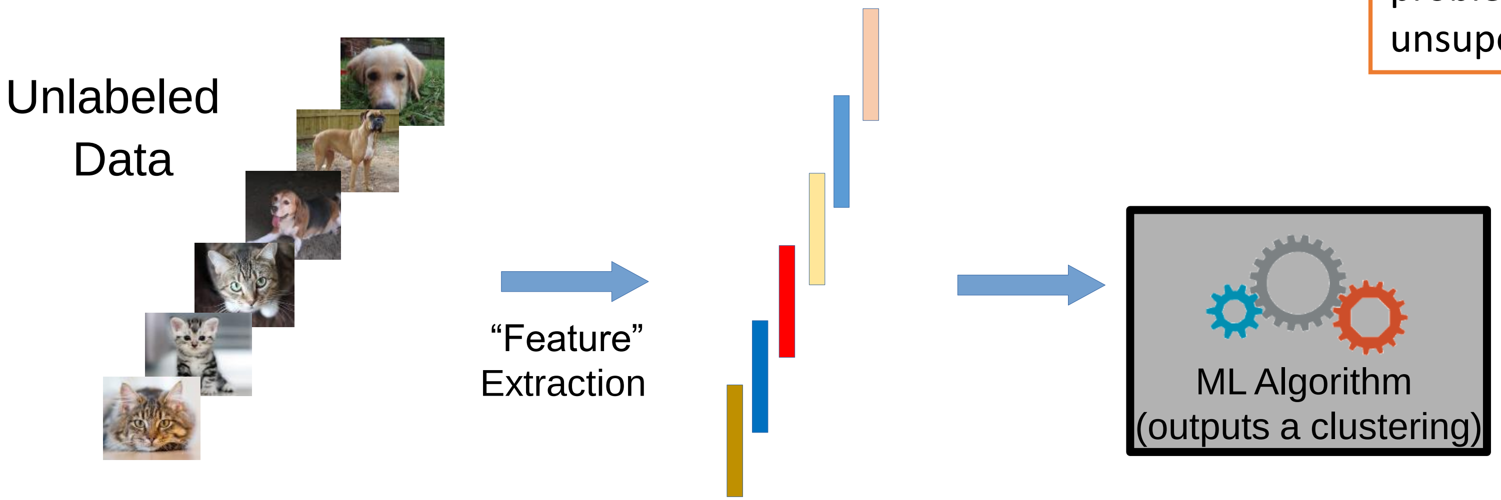
A Typical Unsupervised Learning Workflow



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

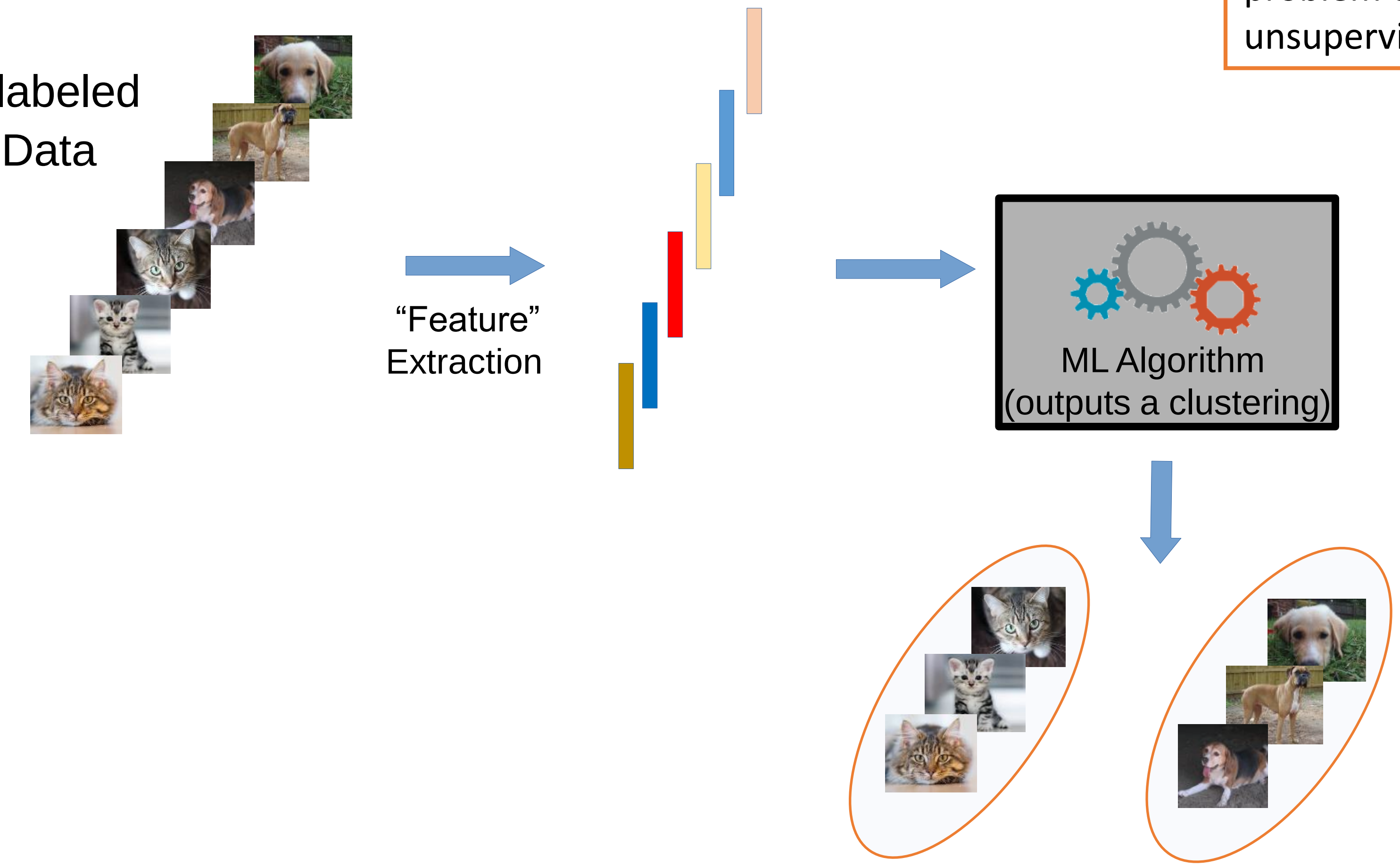
A Typical Unsupervised Learning Workflow

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



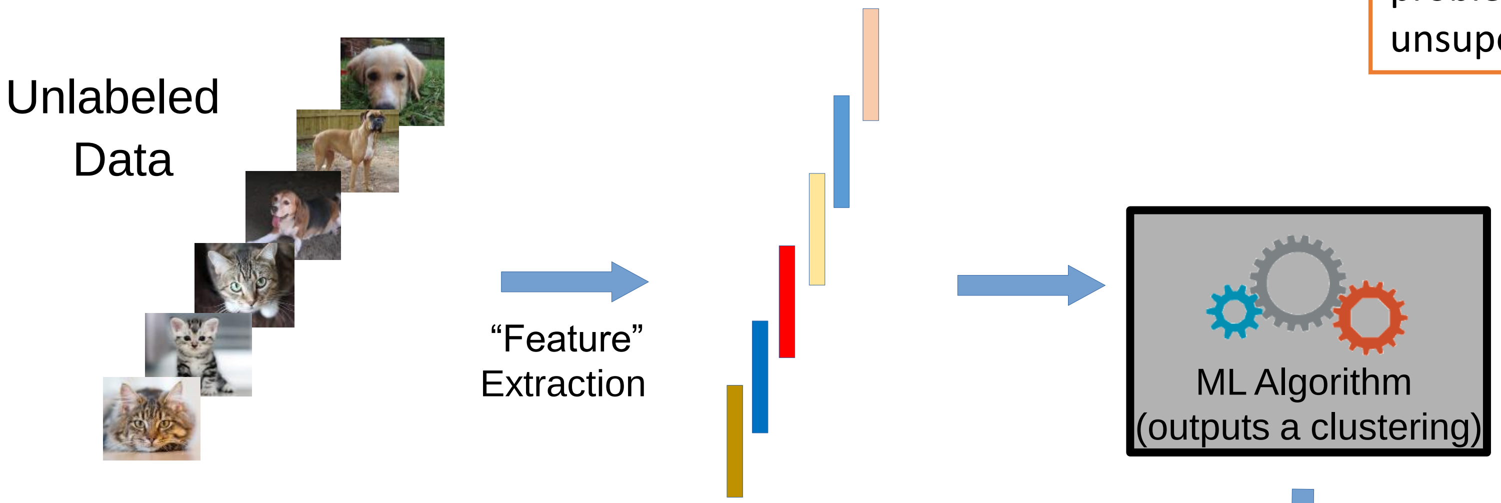
A Typical Unsupervised Learning Workflow

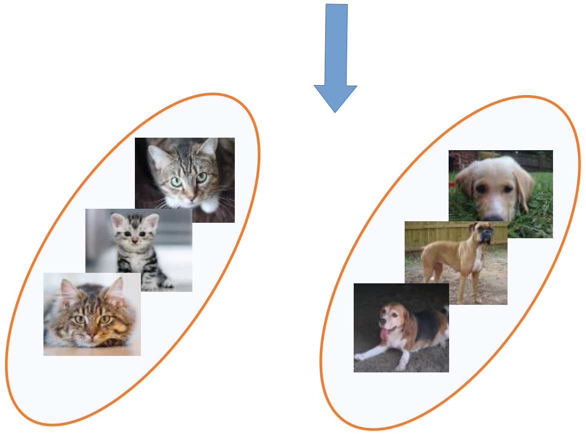
Note: This example is for the problem of data clustering, an unsupervised learning problem



A Typical Unsupervised Learning Workflow

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

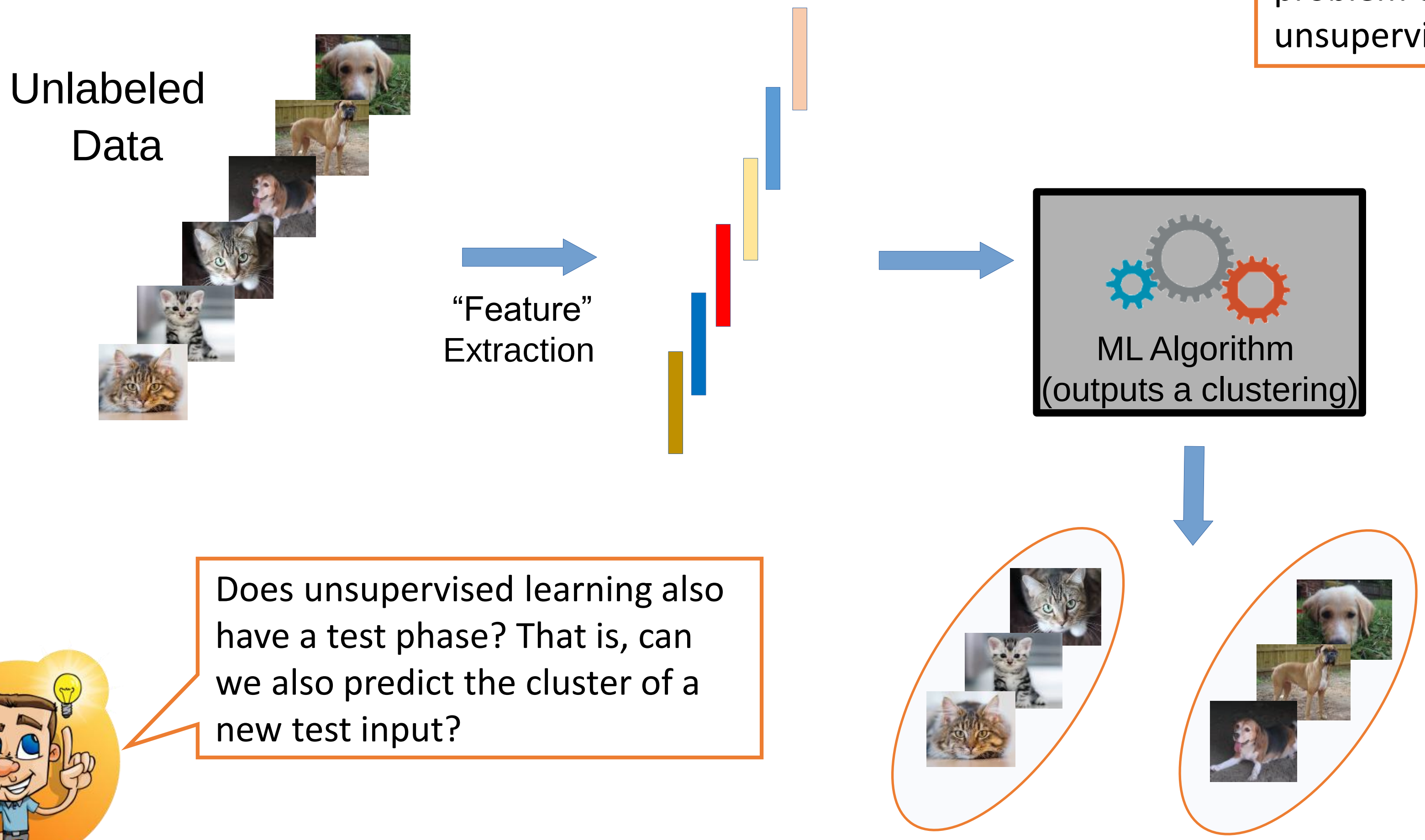




Does unsupervised learning also have a test phase? That is, can we also predict the cluster of a new test input?

A Typical Unsupervised Learning Workflow

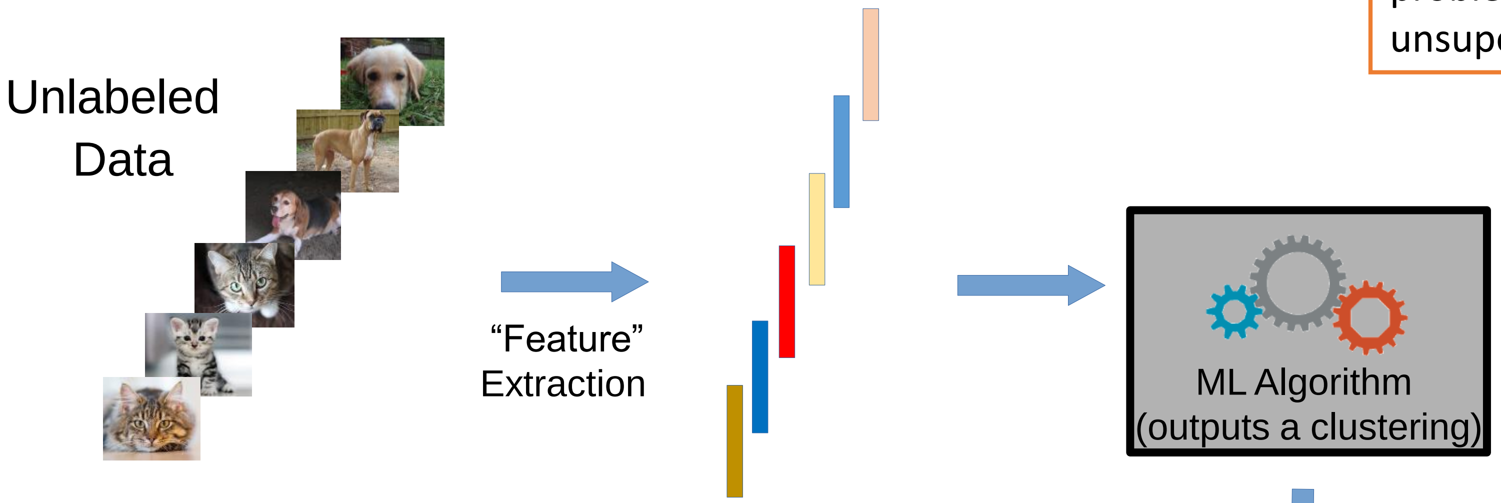
Note: This example is for the problem of data clustering, an unsupervised learning problem

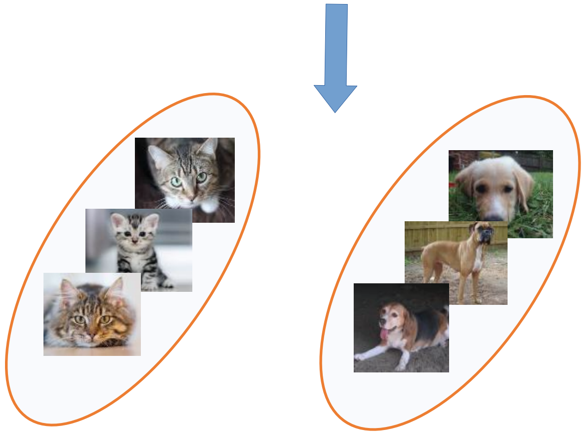


Yes. In this example, given a new “test” cat/dog image, we can assign it to the cluster with closer centroid

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

A Typical Unsupervised Learning Workflow





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?

https://www.pinclipart.com/, http://www.pngtree.com

A Typical Reinforcement Learning Workflow

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

A Typical Reinforcement Learning Workflow

Agent State S_t^a

Environment State at time t

S_t^e

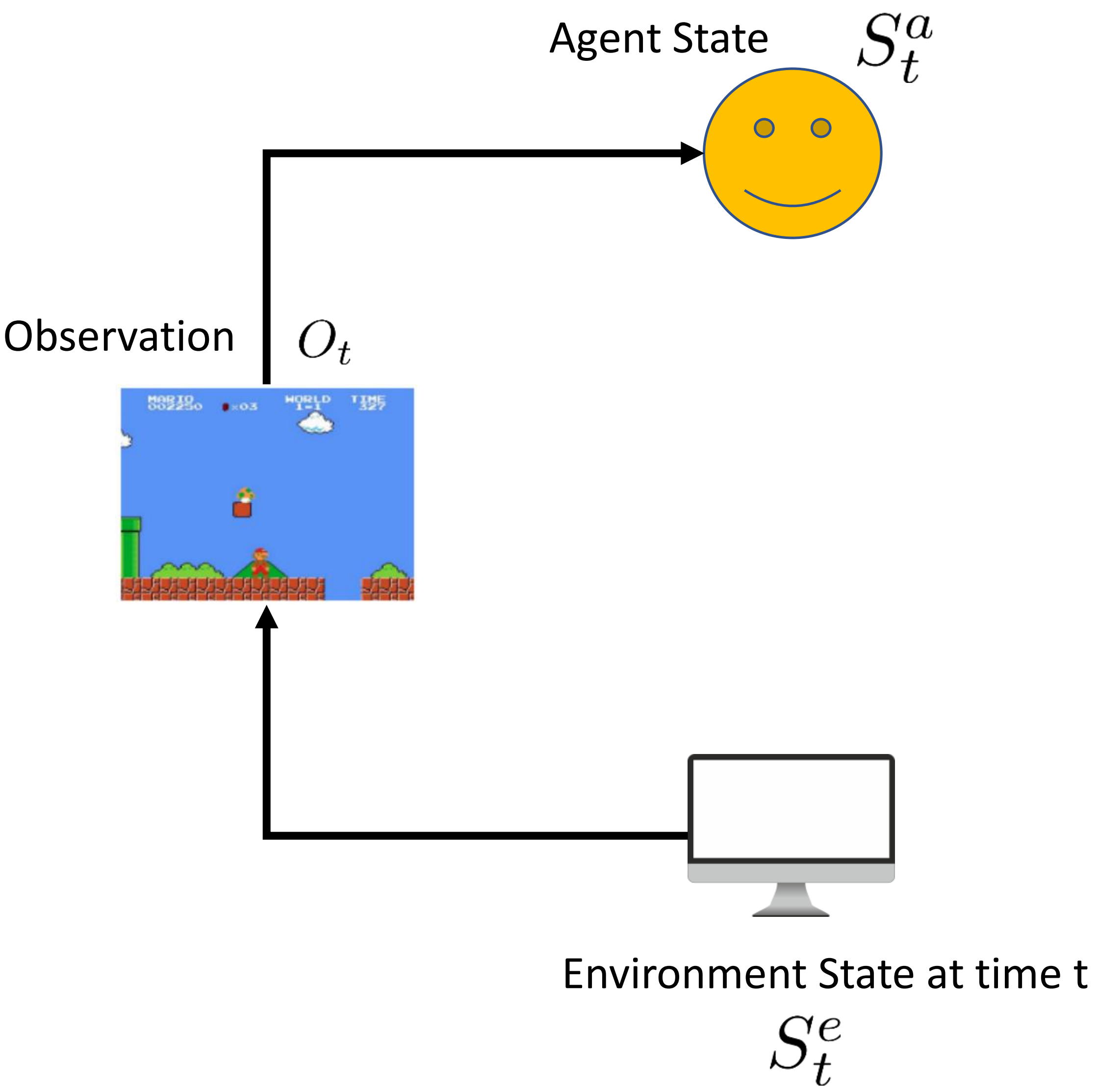
Wish to teach an agent optimal policy for some task

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

A Typical Reinforcement Learning Workflow



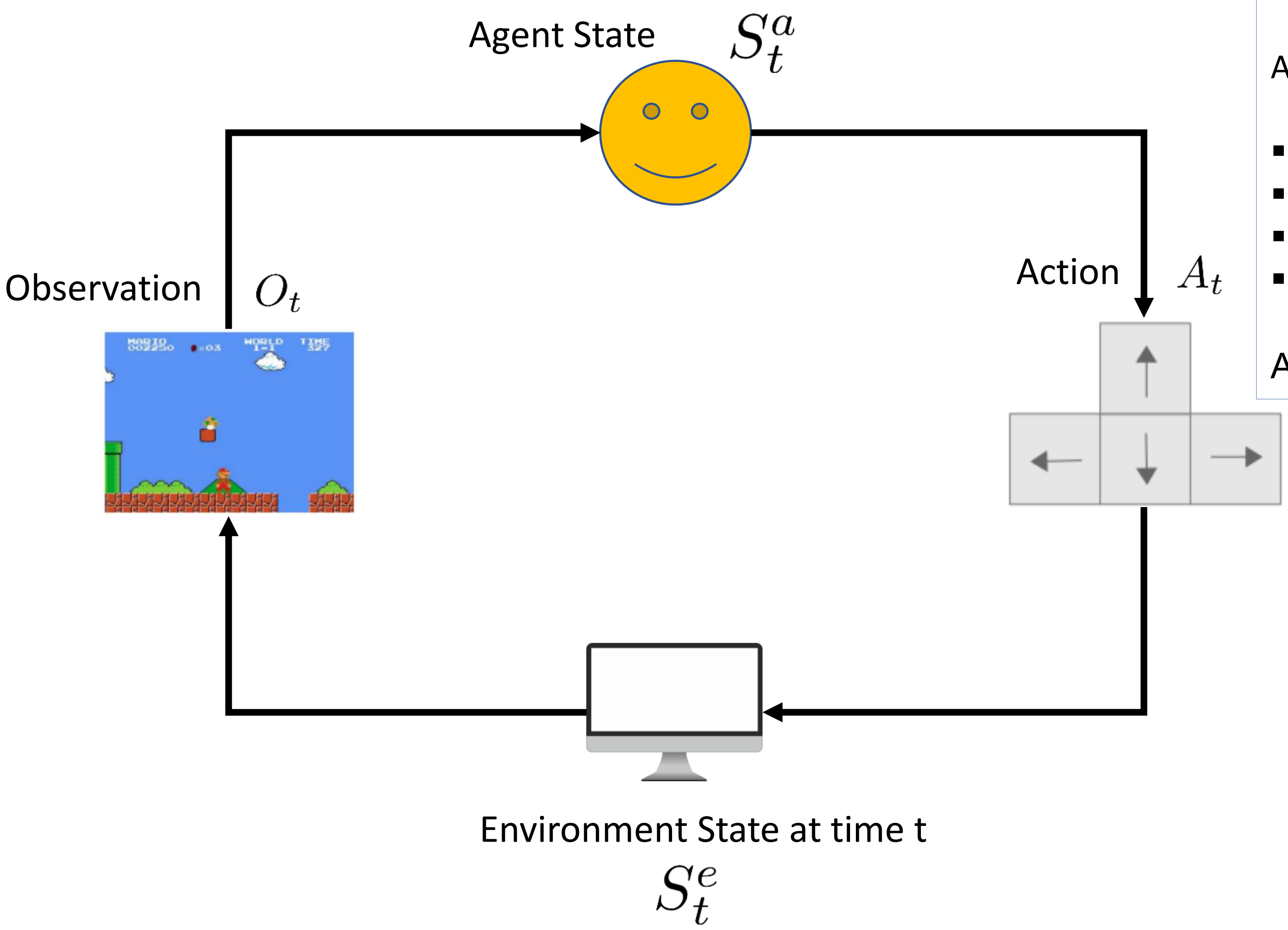
Wish to teach an agent optimal policy for some task

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

A Typical Reinforcement Learning Workflow



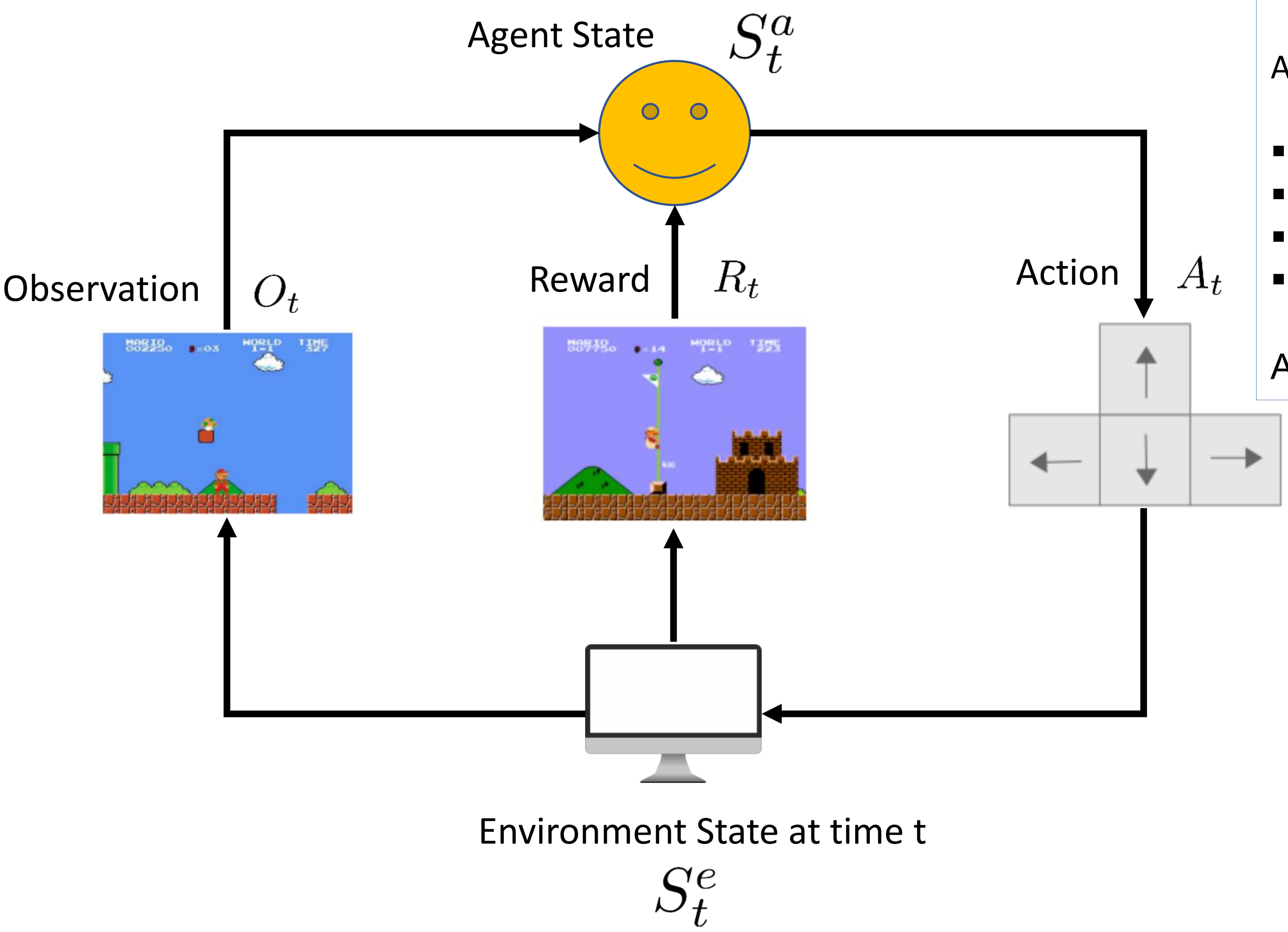
Wish to teach an agent optimal policy for some task

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

A Typical Reinforcement Learning Workflow



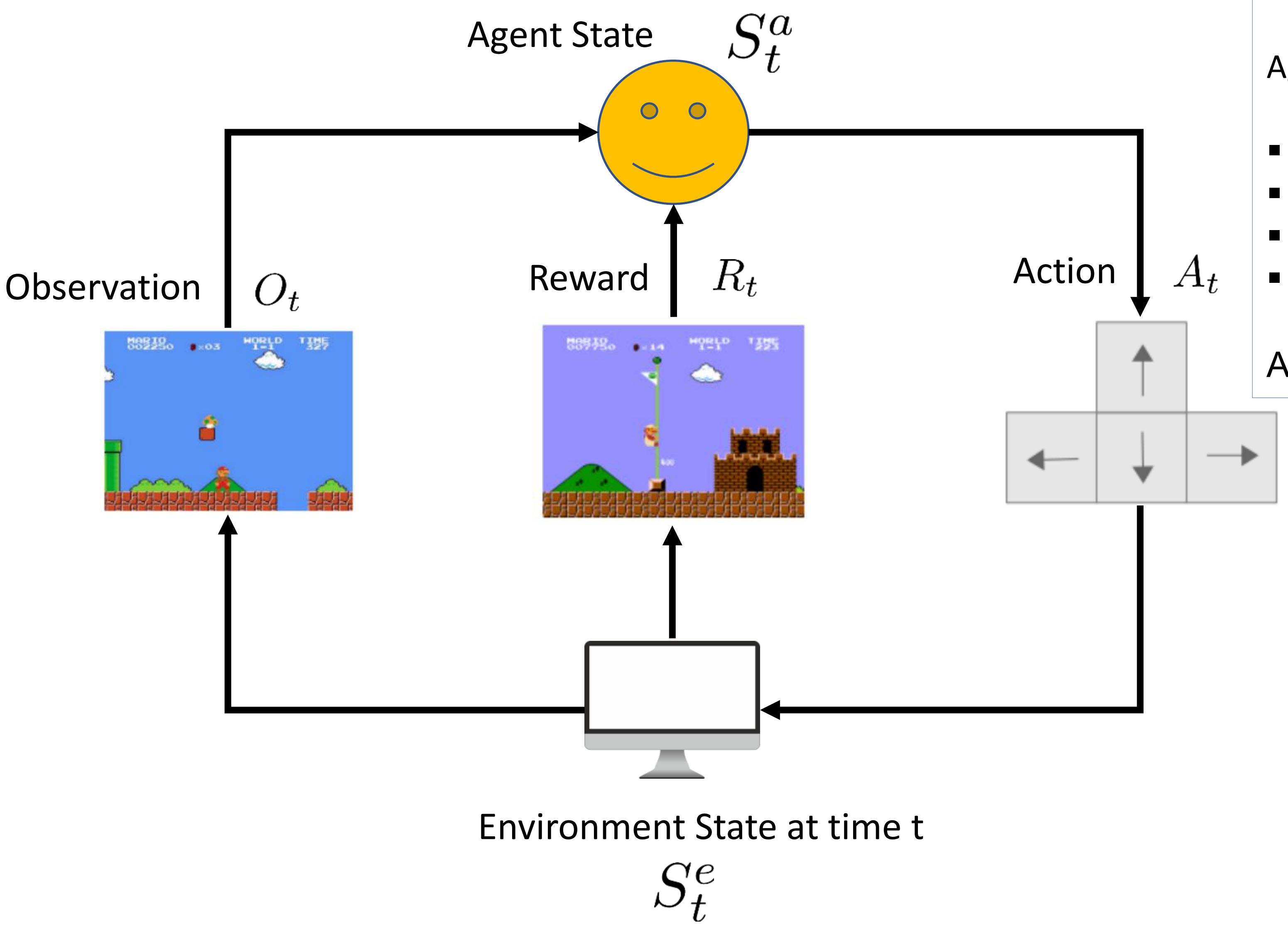
Wish to teach an agent optimal policy for some task

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

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)

ML: Some Perspectives

Geometric Perspective

Geometric Perspective

- Basic fact: Inputs in ML problems can often be represented as points or vectors in some vector space

Geometric Perspective

Recall that feature extraction converts inputs into a numeric representation

- Basic fact: Inputs in ML problems can often be represented as points or vectors in some vector space

Geometric Perspective

Recall that feature extraction converts inputs into a numeric representation

- 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

Geometric Perspective

Recall that feature extraction converts inputs into a numeric representation

- 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

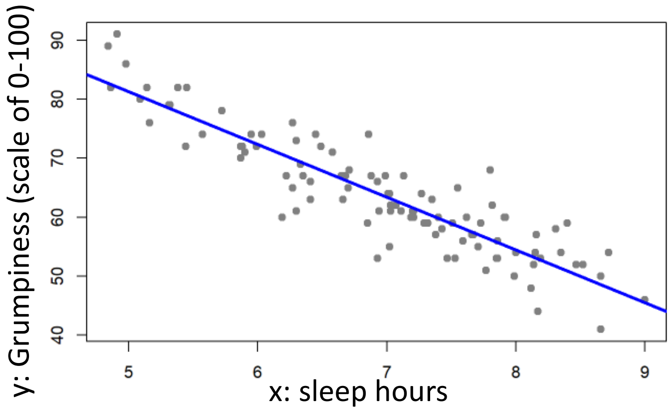
Regression: A supervised learning problem. Goal is to model the relationship between input (x) and real-valued output (y). This is akin to a line or curve fitting problem

Geometric Perspective

Recall that feature extraction converts inputs into a numeric representation

- 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

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



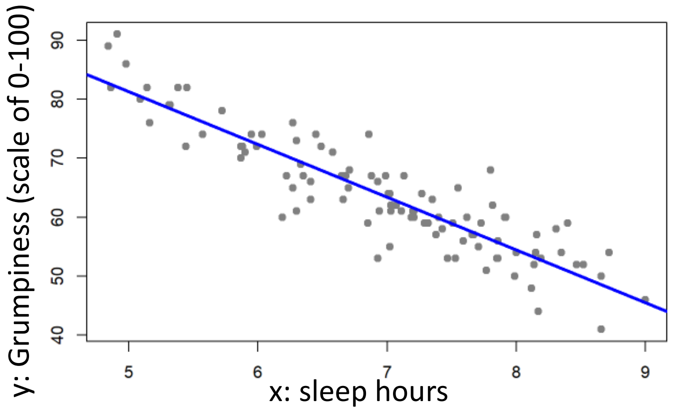
Geometric Perspective

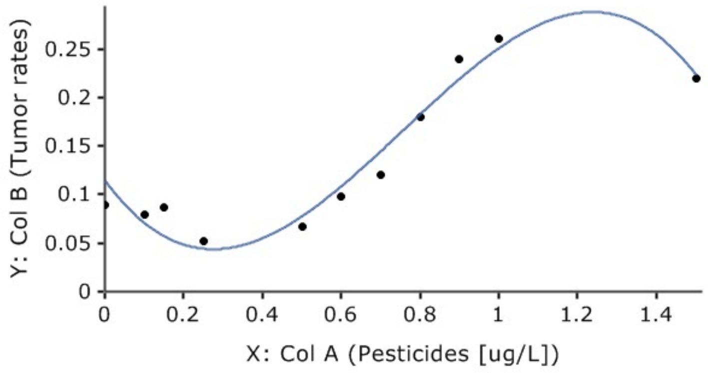
Recall that feature extraction converts inputs into a numeric representation

- 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

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



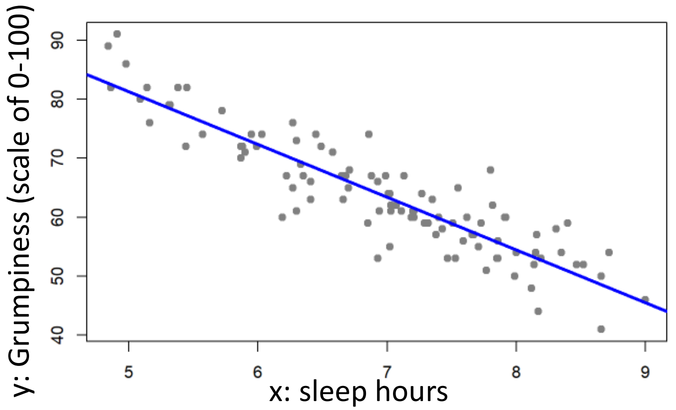


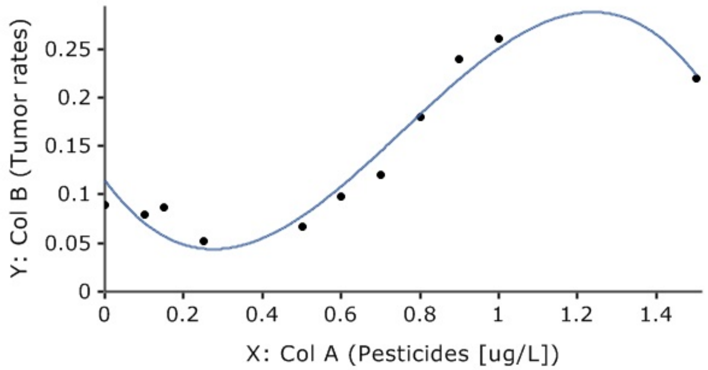
Geometric Perspective

Recall that feature extraction converts inputs into a numeric representation

- 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

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

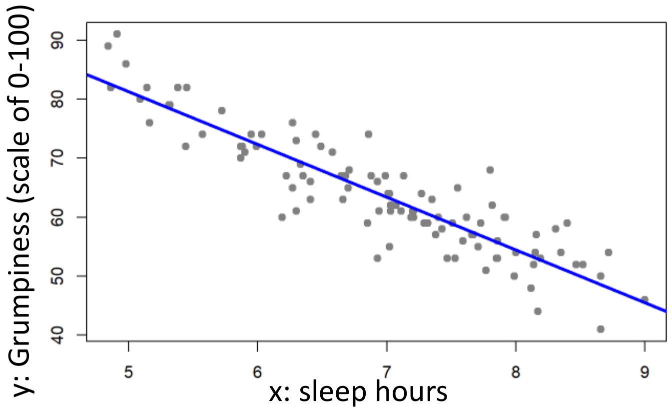
Geometric Perspective

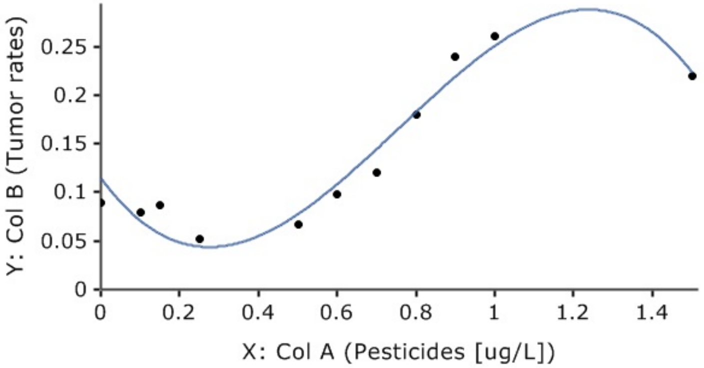
Recall that feature extraction converts inputs into a numeric representation

- Basic fact: Inputs in ML problems can often be represented as points or vectors in some vector space

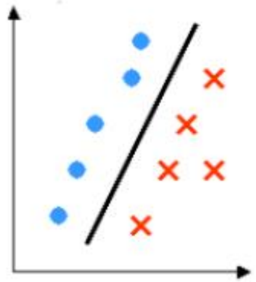
- Doing ML on such data can thus be seen from a geometric view

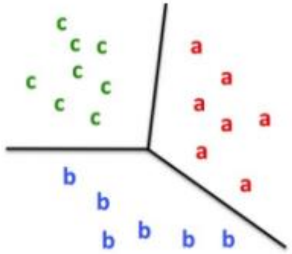
Regression: A supervised learning problem. Goal is to model the relationship between input (x) and real-valued output (y). This is akin to a line or curve fitting problem

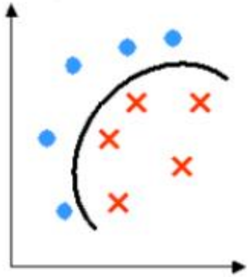


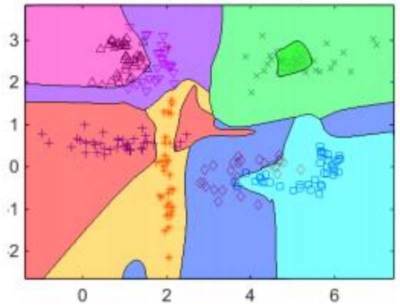


Classification: A supervised learning problem. Goal is to learn a to predict which of the two or more classes an input belongs to. Akin to learning linear/nonlinear separator for the inputs


Two-Class (binary) Linear Classification


Multi-Class Linear Classification


Two-Class (binary) Nonlinear Classification


Multi-Class Nonlinear Classification

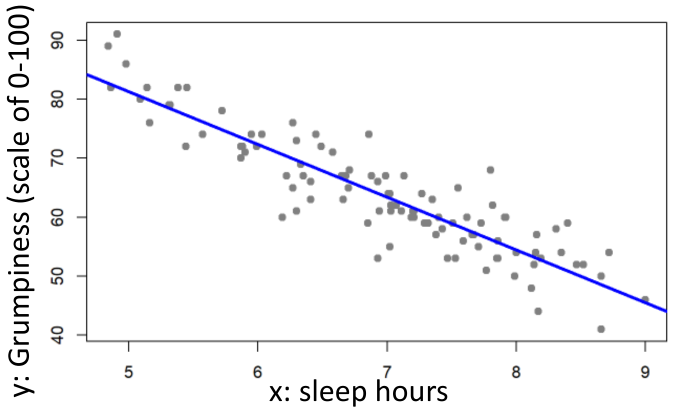
Geometric Perspective

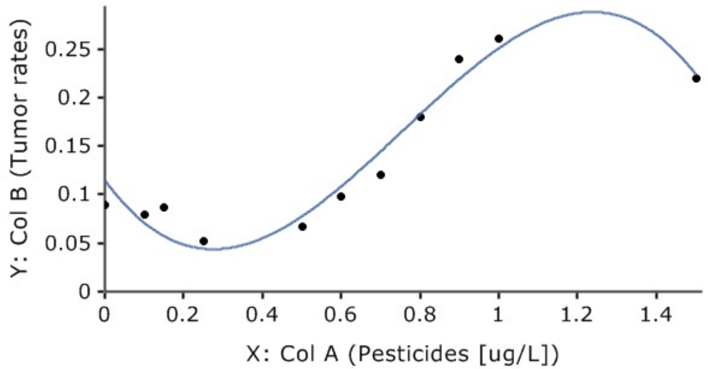
Recall that feature extraction converts inputs into a numeric representation

- Basic fact: Inputs in ML problems can often be represented as points or vectors in some vector space

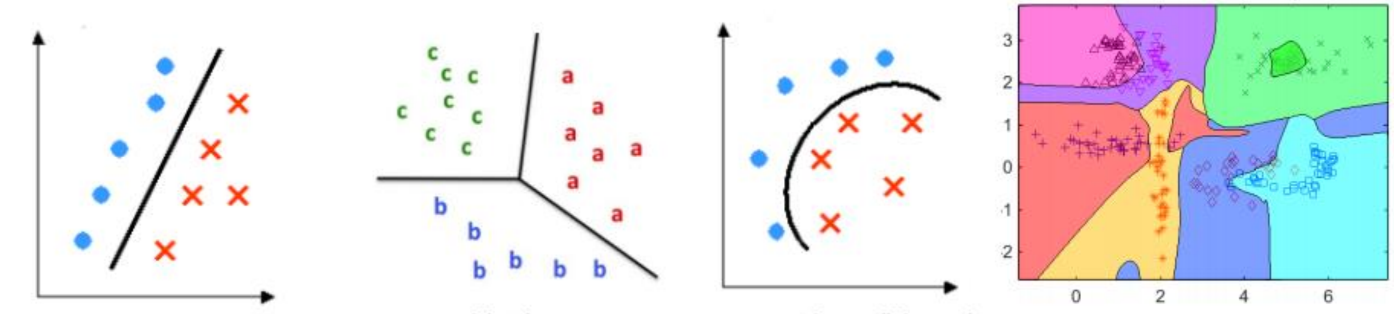
- Doing ML on such data can thus be seen from a geometric view

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



Two-Class (binary) Linear Classification

Multi-Class Linear Classification

Two-Class (binary) Nonlinear Classification

Multi-Class Nonlinear Classification

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

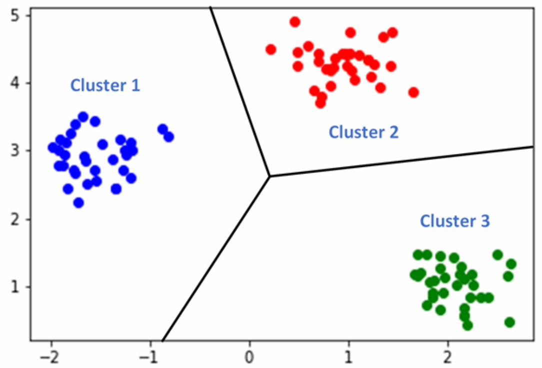
Geometric Perspective

Geometric Perspective

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

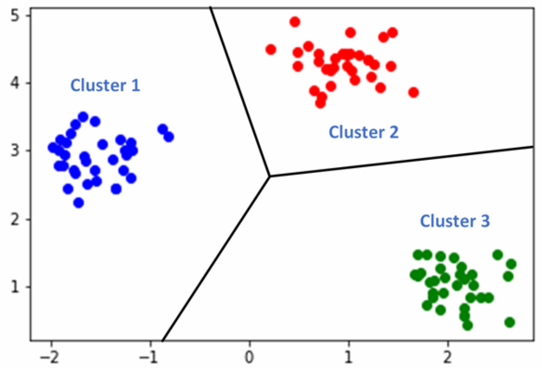
Geometric Perspective

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



Geometric Perspective

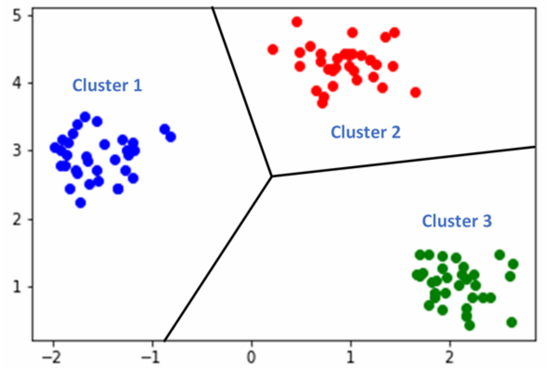
Clustering: An unsupervised learning problem. Goal is to group inputs in a few clusters based on their similarities with each other



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

Geometric Perspective

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

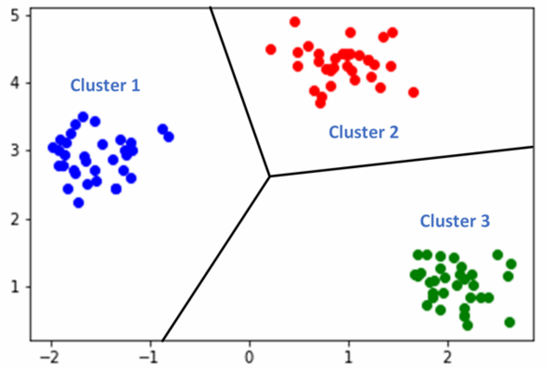


Clustering looks like classification to me. Is there any difference?

Yes. In clustering, we don’t know the labels. Goal is to separate them without any labeled “supervision”

Geometric Perspective

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



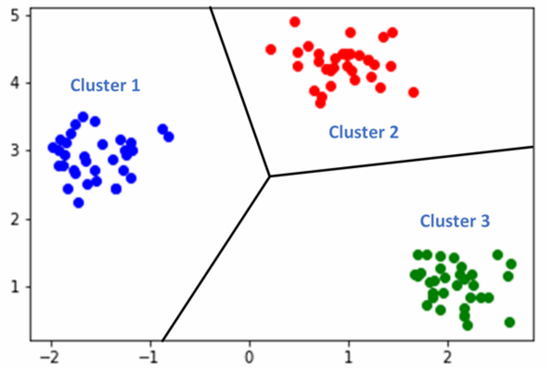
Clustering looks like classification to me. Is there any difference?

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

Geometric Perspective

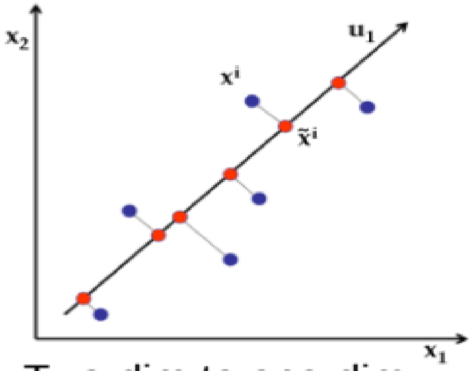
Clustering: An unsupervised learning problem. Goal is to group inputs in a few clusters based on their similarities with each other



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"

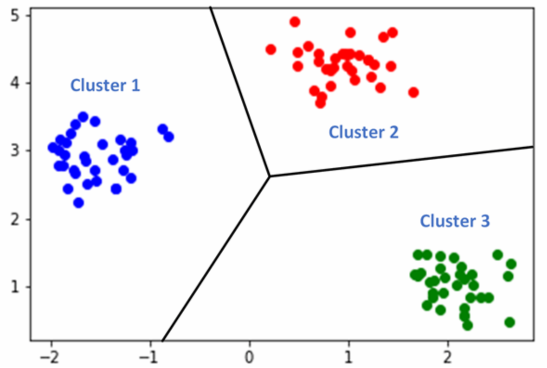
Dimensionality Reduction: An unsupervised learning problem. Goal is to compress the size of each input without losing much information present in the data



Two-dim to one-dim linear projection

Geometric Perspective

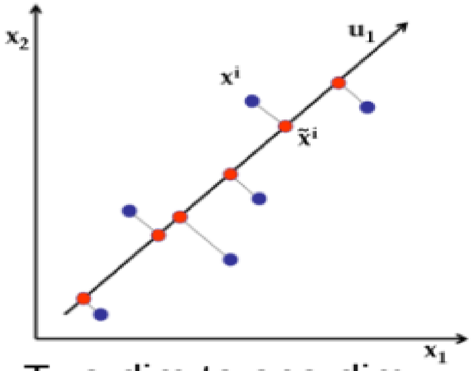
Clustering: An unsupervised learning problem. Goal is to group inputs in a few clusters based on their similarities with each other



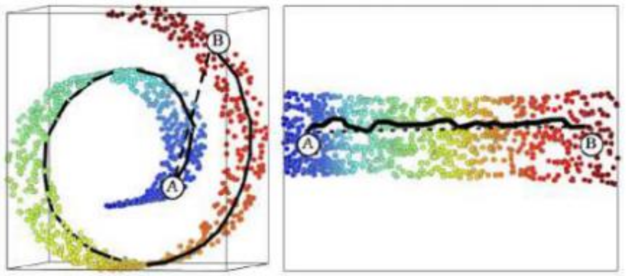
Clustering looks like classification to me. Is there any difference?

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)

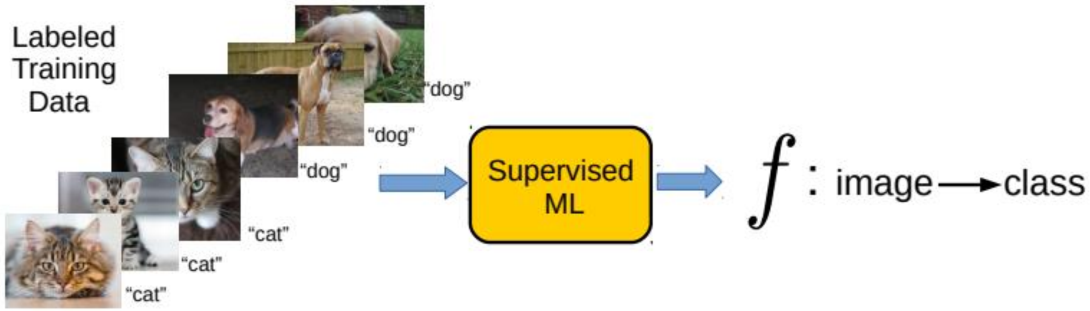
Perspective as function approximation

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

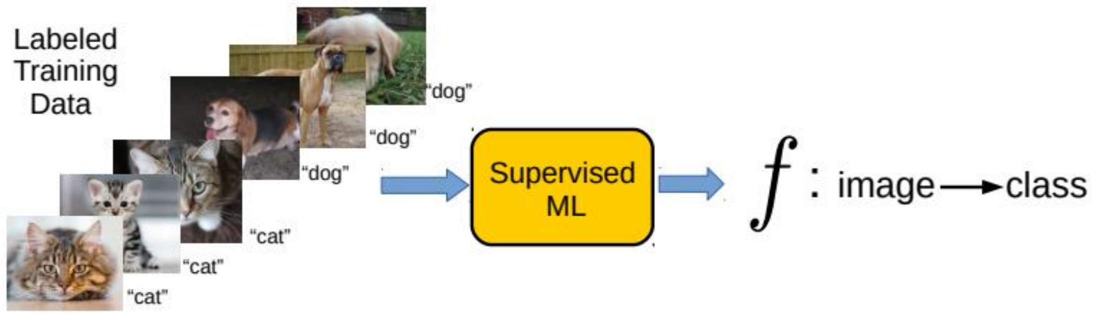
Perspective as function approximation

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



Perspective as function approximation

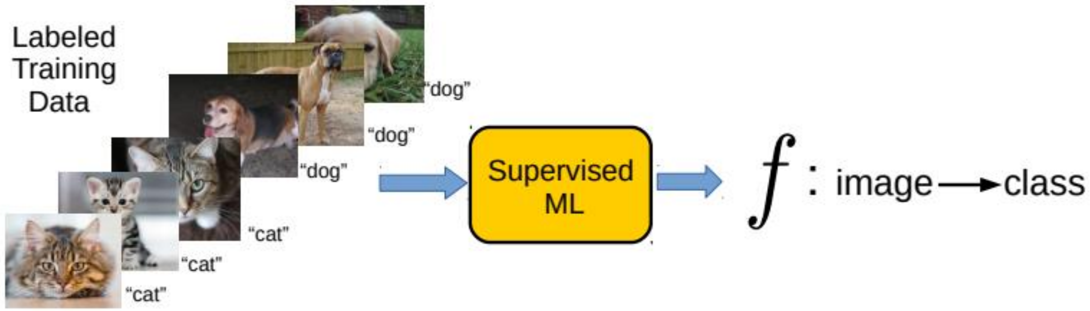
- Supervised Learning ("predict output given input") can be usually thought of as learning a function f that maps each input to the corresponding output



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

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

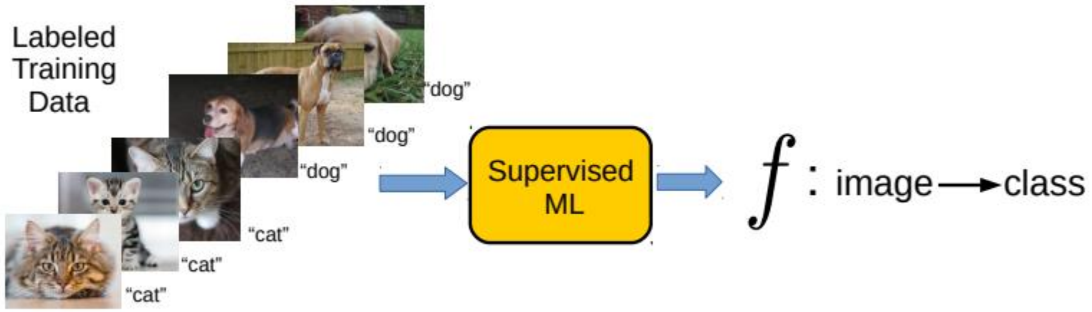


- Unsupervised Learning ("model/compress inputs") can also be usually thought of as learning a function *f* that maps each input to a compact representation

Harder since we don't know the labels in this case

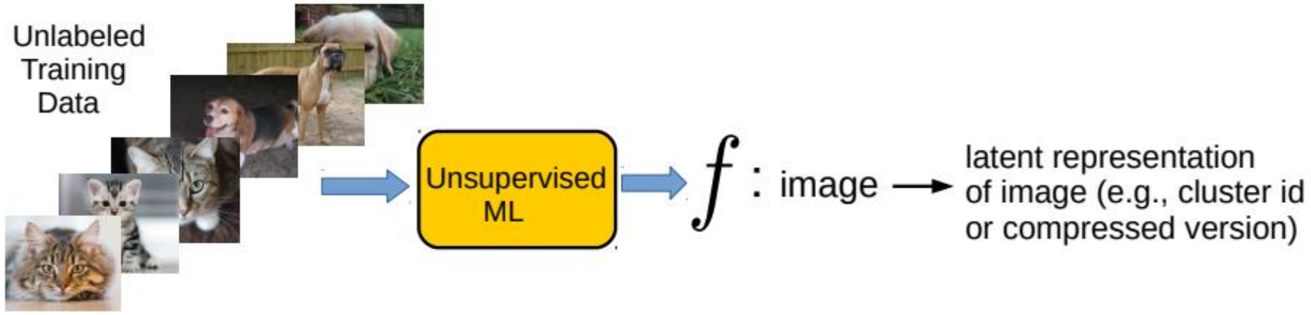
Perspective as function approximation

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



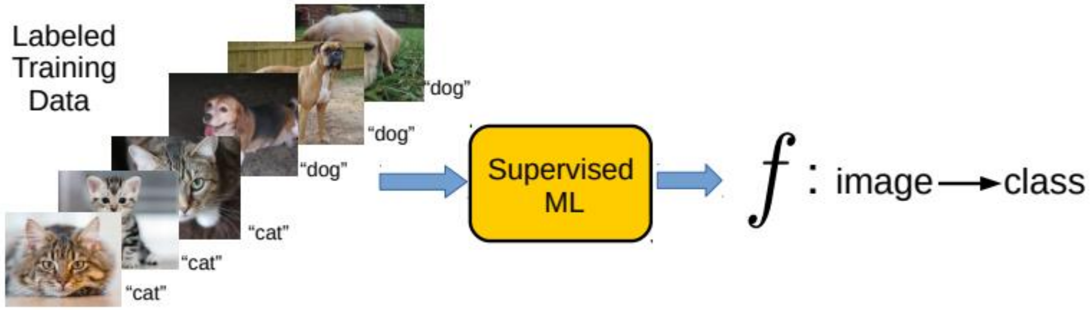
- Unsupervised Learning ("model/compress inputs") can also be usually thought of as learning a function f that maps each input to a compact representation

Harder since we don't know the labels in this case



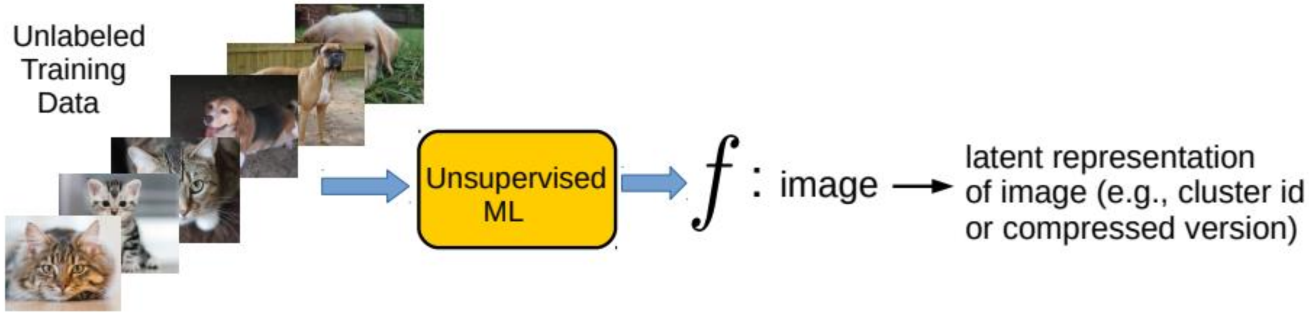
Perspective as function approximation

- 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

Harder since we don't know the labels in this case



- Reinforcement Learning can also be seen as doing function approximation

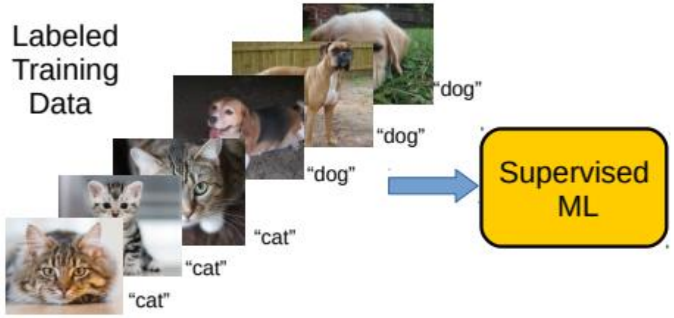
Perspective as probability estimation

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

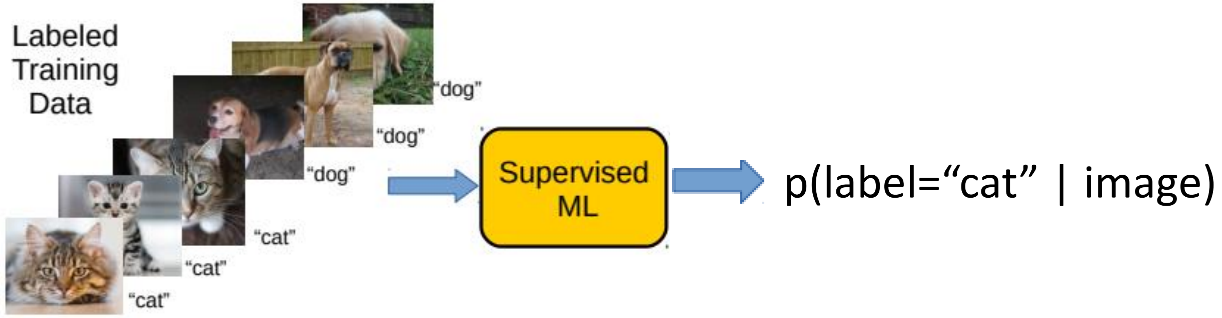
Perspective as probability estimation

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



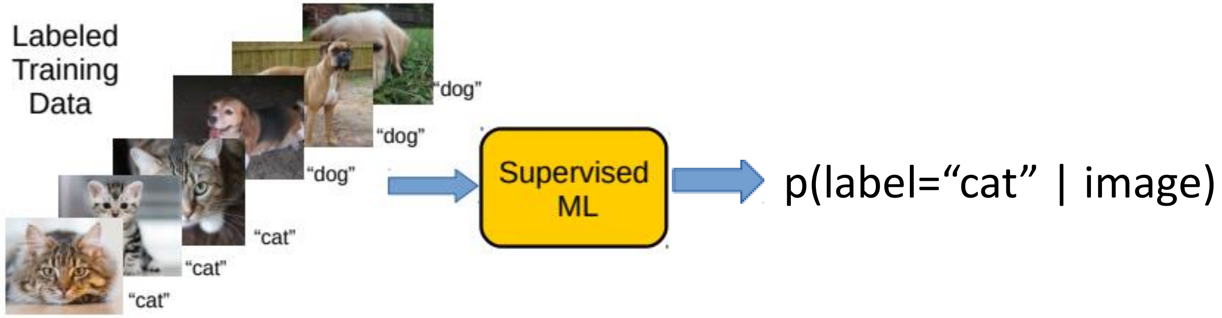
Perspective as probability estimation

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



Perspective as probability estimation

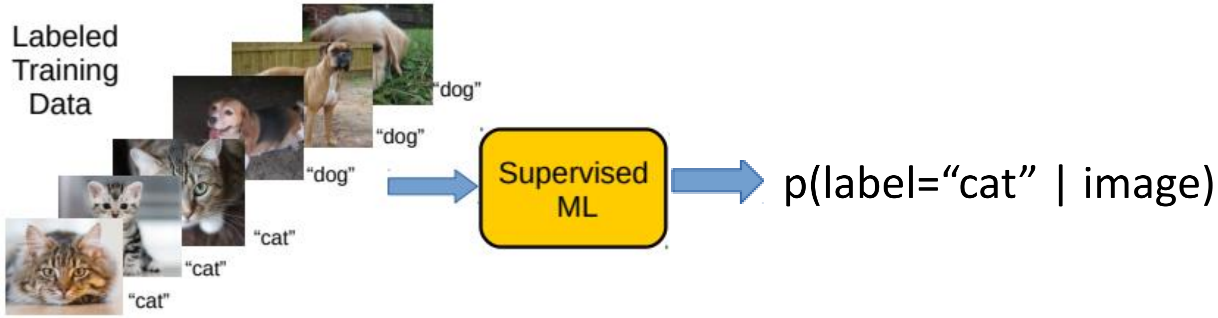
- Supervised Learning (“predict output given input”) can be thought of as estimating the conditional probability of each possible output given an input



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

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

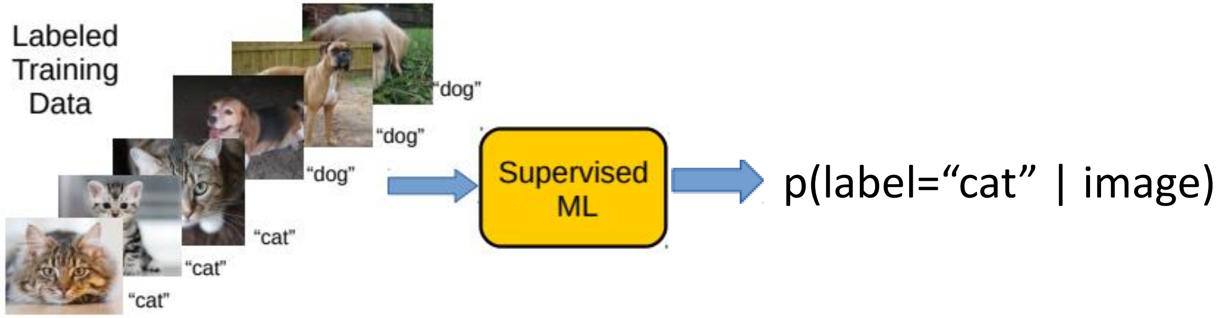


- Unsupervised Learning ("model/compress inputs") can be thought of as estimating the probability density of the inputs

Harder since we don't know the labels in this case

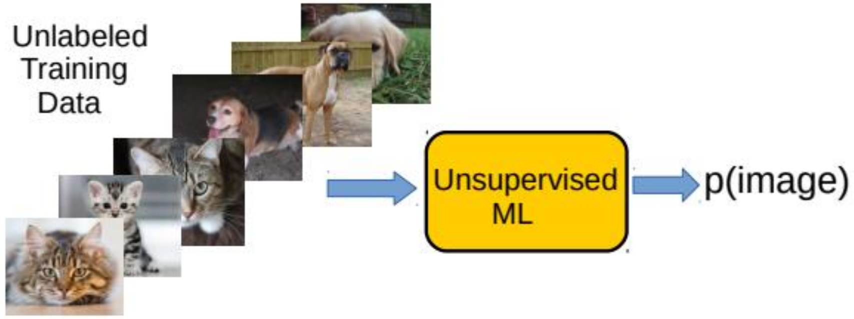
Perspective as probability estimation

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



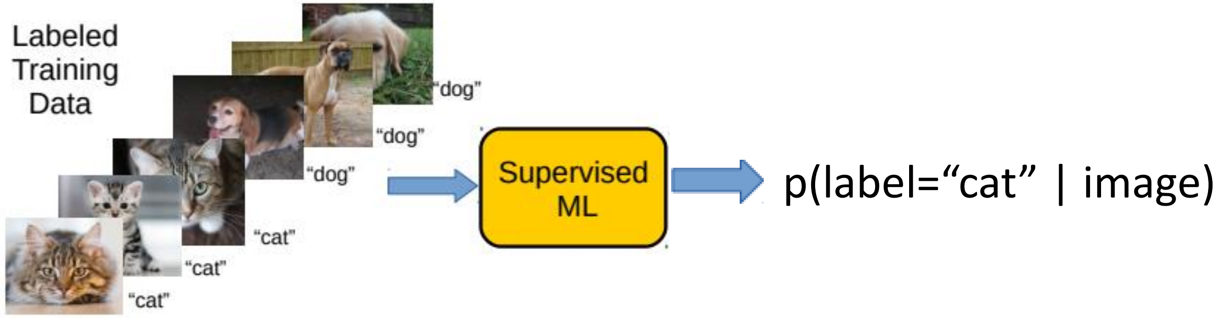
- Unsupervised Learning ("model/compress inputs") can be thought of as estimating the probability density of the inputs

Harder since we don't know the labels in this case



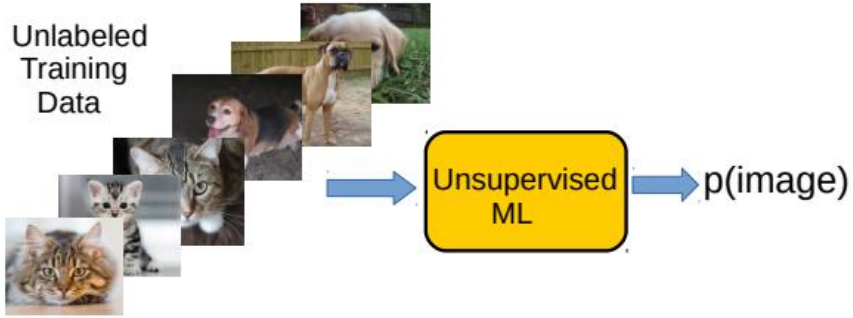
Perspective as probability estimation

- 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

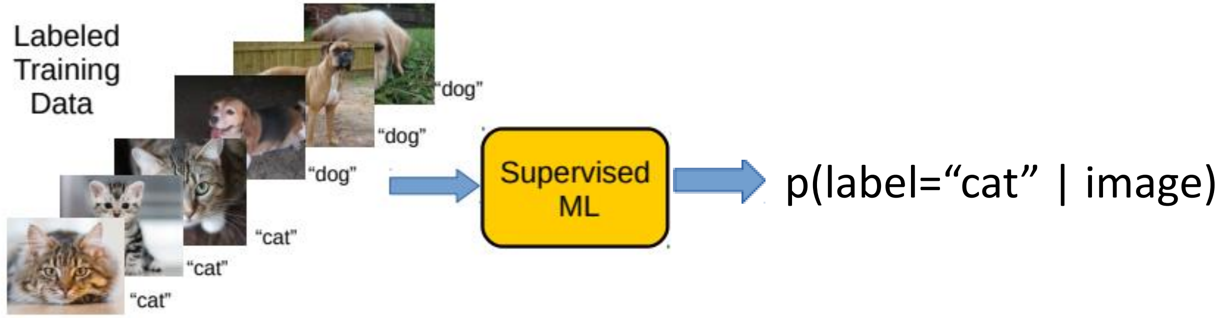
Harder since we don't know the labels in this case



Don't worry if this doesn't make much sense as of now ☺ But the basic idea is to learn the underlying data distribution using the unlabeled inputs; many ways to do this as we will see later

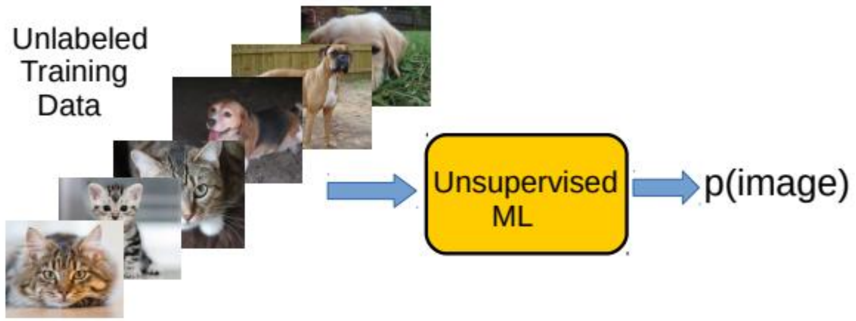
Perspective as probability estimation

- 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

Harder since we don't know the labels in this case



Don't worry if this doesn't make much sense as of now ☺ But 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 Features

Data and Features

Data and Features

- ML algos require a numeric feature representation of the inputs

Data and Features

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

- ML algos require a numeric feature representation of the inputs

Data and Features

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

- ML algos require a numeric feature representation of the inputs

- Features can be obtained using one of the two approaches

Data and Features

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

- 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

Data and Features

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

- 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 <u>manually</u> from raw inputs
  - Approach 2: <u>Learning</u> the features from raw inputs

Data and Features

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

- 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

Data and Features

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

- 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 <u>manually</u> from raw inputs
  - Approach 2: <u>Learning</u> 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)

Data and Features

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

- 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

Example: Feature Extraction for Text Data

Example: Feature Extraction for Text Data

- Consider some text data consisting of the following sentences:

Example: Feature Extraction for Text Data

- Consider some text data consisting of the following sentences:
  - John likes to watch movies

Example: Feature Extraction for Text Data

- Consider some text data consisting of the following sentences:
  - John likes to watch movies
  - Mary likes movies too

Example: Feature Extraction for Text Data

- Consider some text data consisting of the following sentences:
  - John likes to watch movies
  - Mary likes movies too
  - John also likes football

Example: Feature Extraction for 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

Example: Feature Extraction for 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

Example: Feature Extraction for 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

$$
\begin{array}{l}
\\
\text{Sentence 1} \\
\text{Sentence 2} \\
\text{Sentence 3}
\end{array}
\begin{pmatrix}
\text{John} & \text{likes} & \text{to} & \text{watch} & \text{movies} & \text{Mary} & \text{too} & \text{also} & \text{football} \\
1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\
1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1
\end{pmatrix}
$$

Example: Feature Extraction for Text Data

- Consider some text data consisting of the following sentences:
  - John likes to watch movies
  - Mary likes movies too
  - John also likes football

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 or some?), but often works reasonably well

- Want to construct a feature representation for these sentences
- Here is a "bag-of-words" (BoW) feature representation of these sentences

$$
\begin{array}{l}
\\
\text{Sentence 1} \\
\text{Sentence 2} \\
\text{Sentence 3}
\end{array}
\begin{pmatrix}
\text{John} & \text{likes} & \text{to} & \text{watch} & \text{movies} & \text{Mary} & \text{too} & \text{also} & \text{football} \\
1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\
1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1
\end{pmatrix}
$$

Example: Feature Extraction for Text Data

- Consider some text data consisting of the following sentences:
  - John likes to watch movies
  - Mary likes movies too
  - John also likes football

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 or some?), but often works reasonably well

- Want to construct a feature representation for these sentences
- Here is a "bag-of-words" (BoW) feature representation of these sentences

	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

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

Example: Feature Extraction for Image Data

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

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

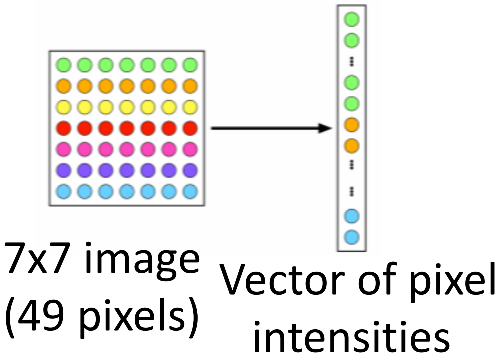
Example: Feature Extraction for Image Data

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

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

Example: Feature Extraction for Image Data

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

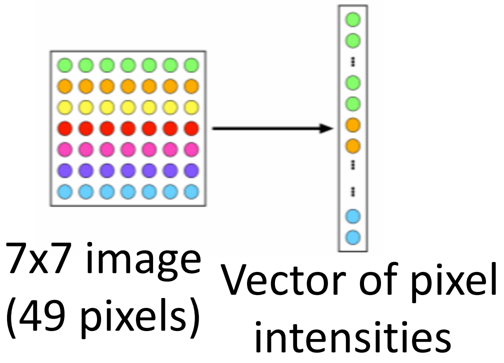


- Histogram of visual patterns is another popular feature extr. method for images

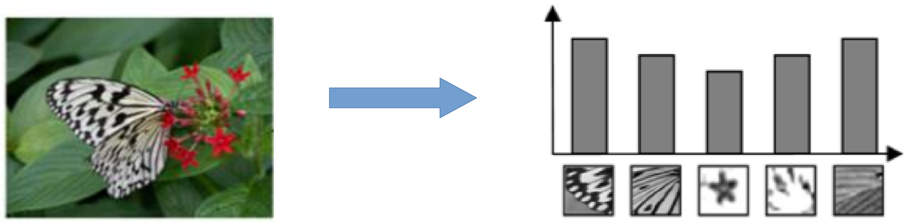
Pic credit: cat.uab.cat/Research/object-recognition

Example: Feature Extraction for Image Data

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



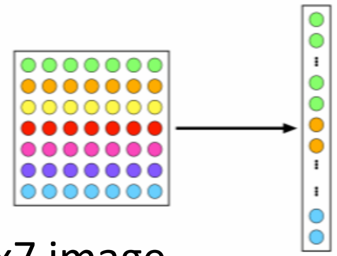
- Histogram of visual patterns is another popular feature extr. method for images



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

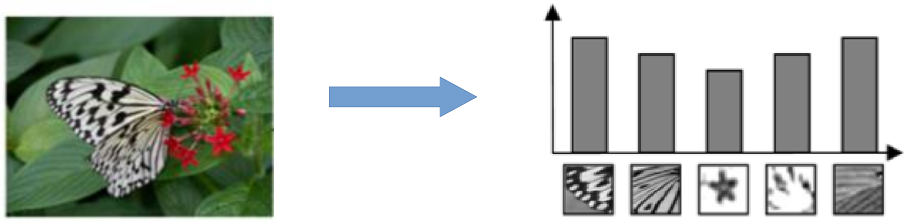
Example: Feature Extraction for Image Data

- 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

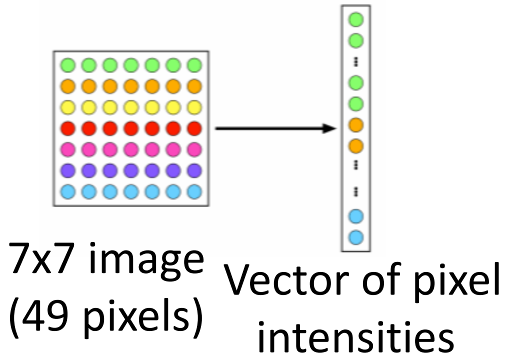
- Histogram of visual patterns is another popular feature extr. method for images



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

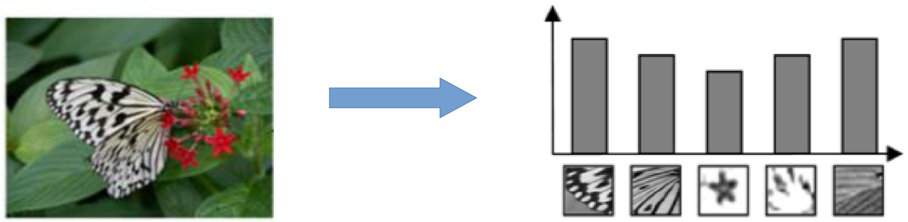
Example: Feature Extraction for Image Data

- 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

Feature Selection

Feature Selection

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

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

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

Age
Gender
Height
Weight
Eye color

$\rightarrow$ Body-mass index (BMI)

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

~~Age~~
~~Gender~~
Height
Weight
~~Eye color~~

→ Body-mass index (BMI)

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

~~Age~~
~~Gender~~
Height
Weight
~~Eye color~~

→ Body-mass index (BMI)

Calculating BMI from this data doesn't require ML but this simple example is just to illustrate the idea of feature selection ☺

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

~~Age~~
~~Gender~~
Height
Weight
~~Eye color~~

→ Body-mass index (BMI)

Calculating BMI from this data doesn't require ML but this simple example is just to illustrate the idea of feature selection ☺

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

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

~~Age~~
~~Gender~~
Height
Weight
~~Eye color~~

→ Body-mass index (BMI)

Calculating BMI from this data doesn't require ML but this simple example is just to illustrate the idea of feature selection ☺

- 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 unsup. learning

Some More Postprocessing: Feature Scaling

Some More Postprocessing: Feature Scaling

- Even after feature selection, the features may not be on the same scale

Pic credit: https://becominghuman.ai/demystifying-feature-scaling-baff53e9b3fd, https://stackoverflow.com/

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

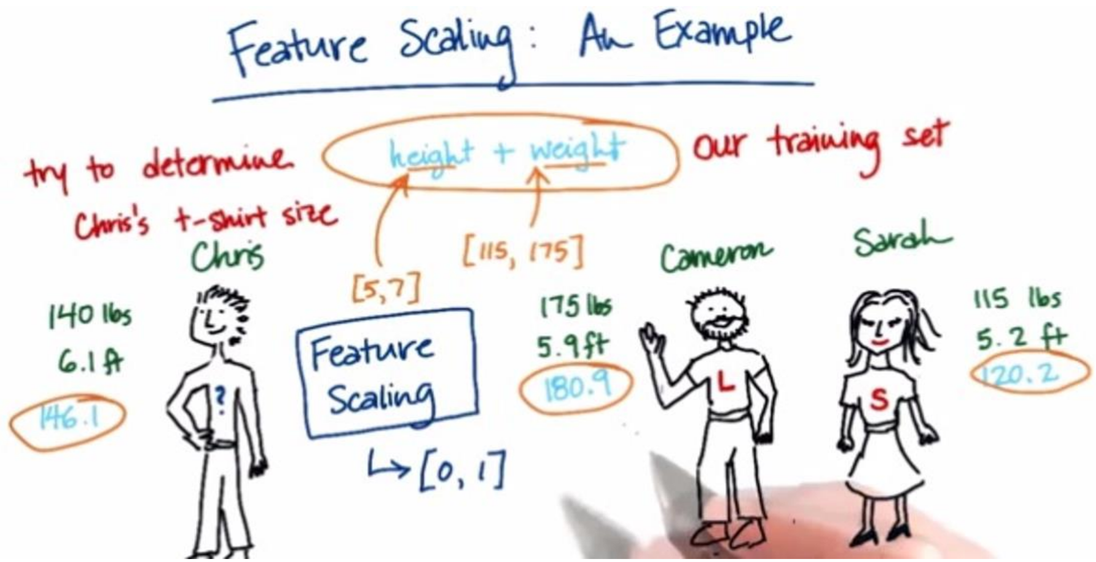
Some More Postprocessing: Feature Scaling

- 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)

Pic credit: https://becominghuman.ai/demystifying-feature-scaling-baff53e9b3fd, https://stackoverflow.com/

Some More Postprocessing: Feature Scaling

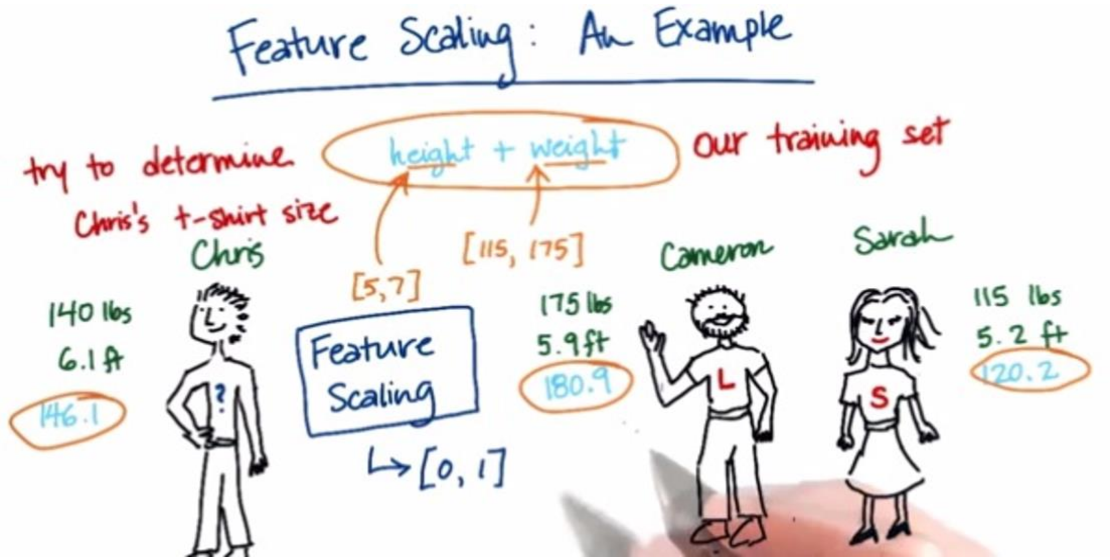
- Even after feature selection, the features may not be on the same scale
- This can be problematic when comparing two inputs – features that have larger scales may dominate the result of such comparisons
- Therefore helpful to standardize the features (e.g., by bringing all of them on the same scale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



Pic credit: https://becominghuman.ai/demystifying-feature-scaling-baff53e9b3fd, https://stackoverflow.com/

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)

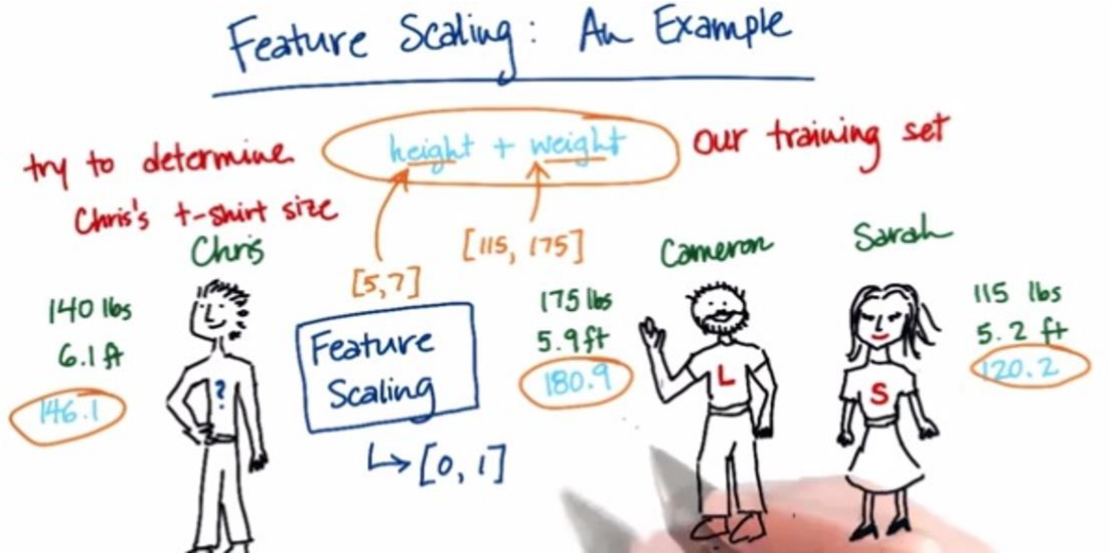


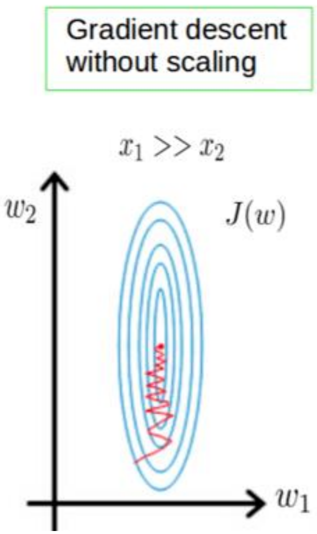
- Also helpful for stabilizing the optimization techniques used in ML algos

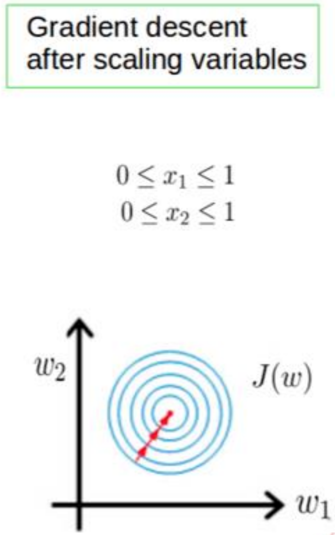
Pic credit: https://becominghuman.ai/demystifying-feature-scaling-baff53e9b3fd, https://stackoverflow.com/

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

Pic credit: https://becominghuman.ai/demystifying-feature-scaling-baff53e9b3fd, https://stackoverflow.com/

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

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

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

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

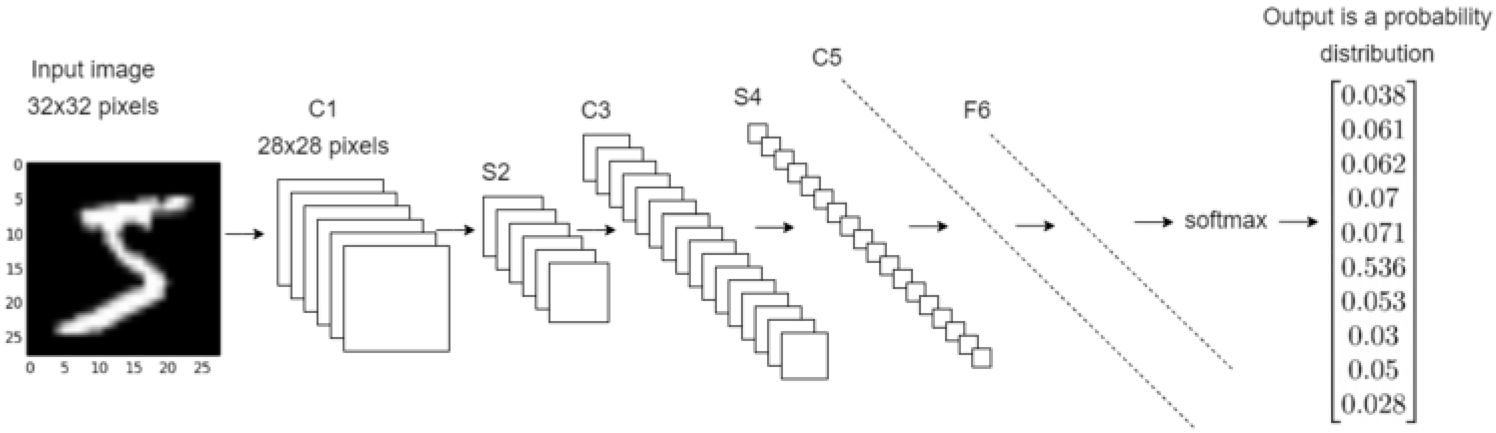
Deep Learning = ML with automated feature learning from the raw inputs

Feature extraction part is automated via the feature learning module

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

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

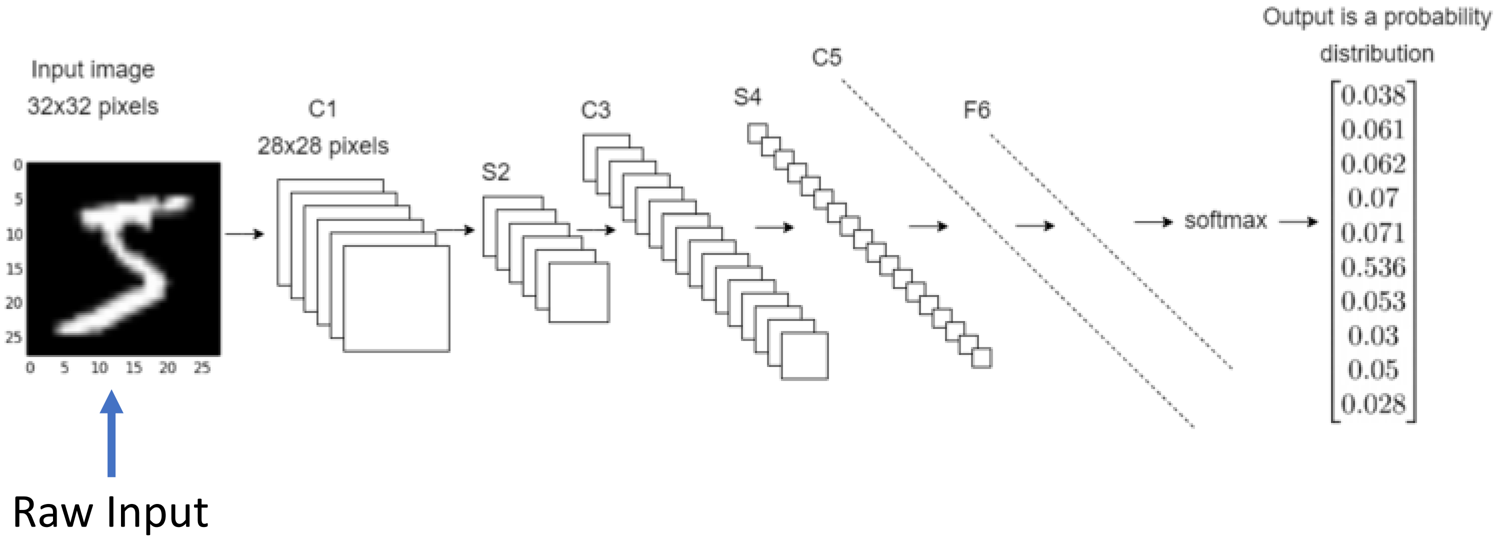
Feature extraction part is automated via the feature learning module



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

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

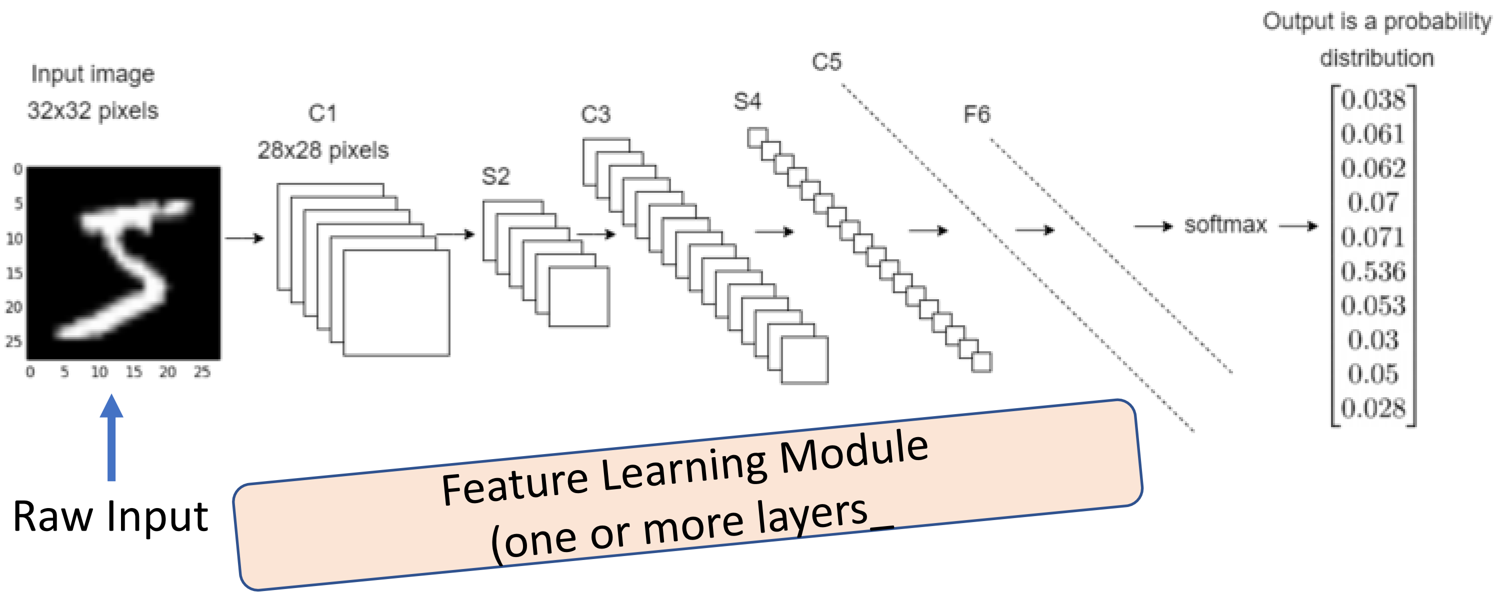
Feature extraction part is automated via the feature learning module



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

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

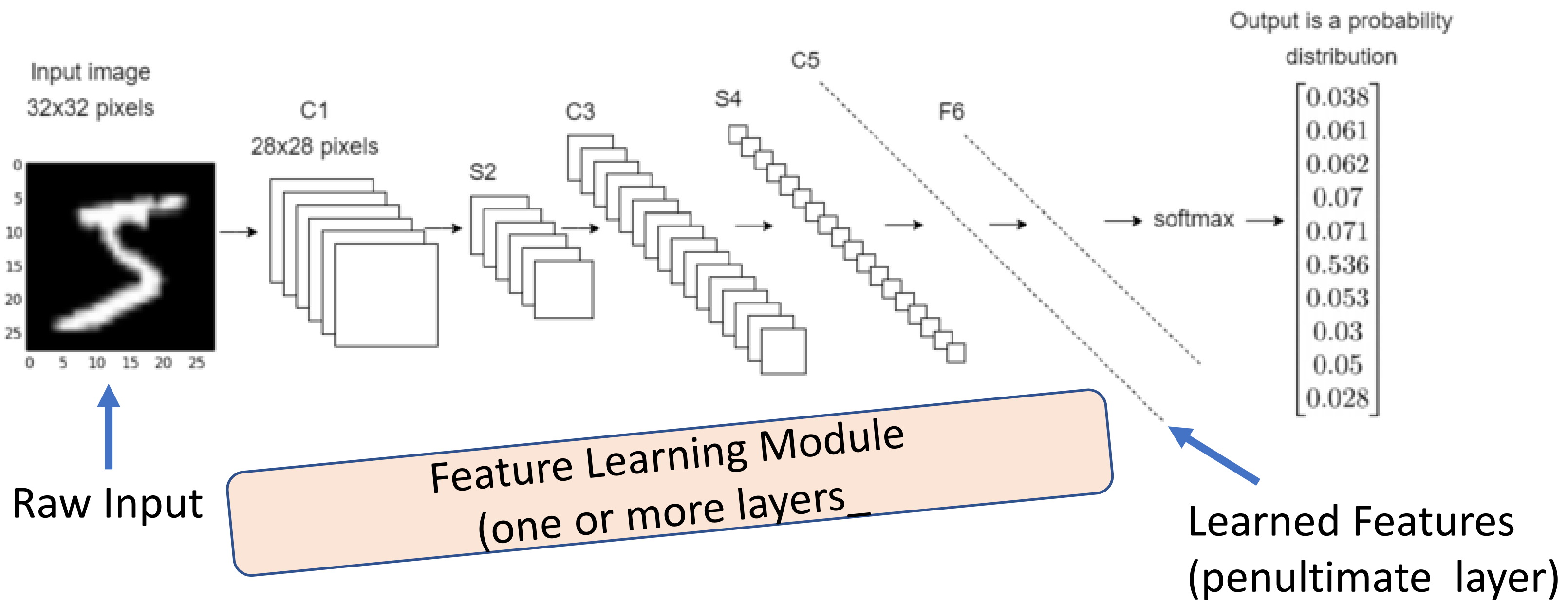
Feature extraction part is automated via the feature learning module



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

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

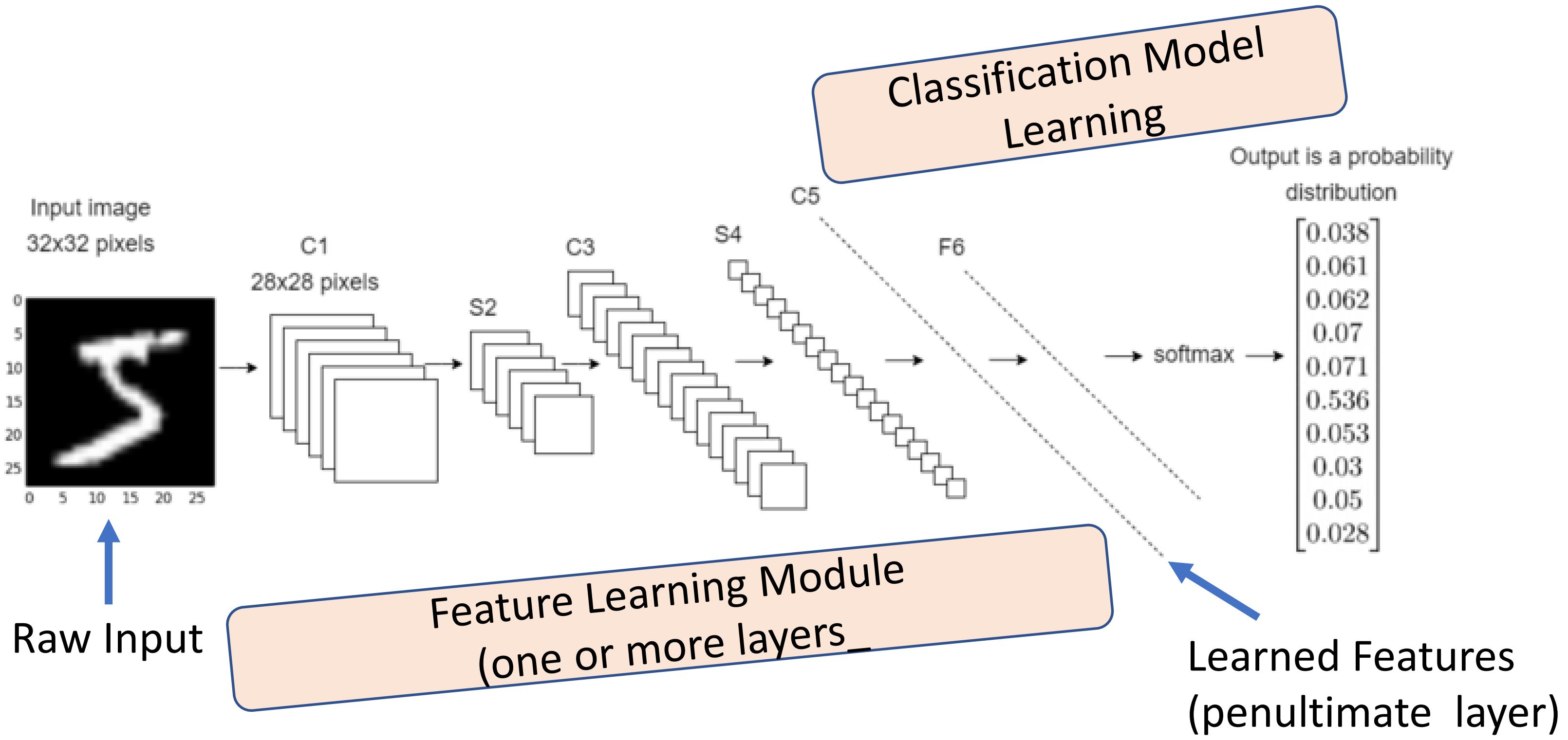
Feature extraction part is automated via the feature learning module



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

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

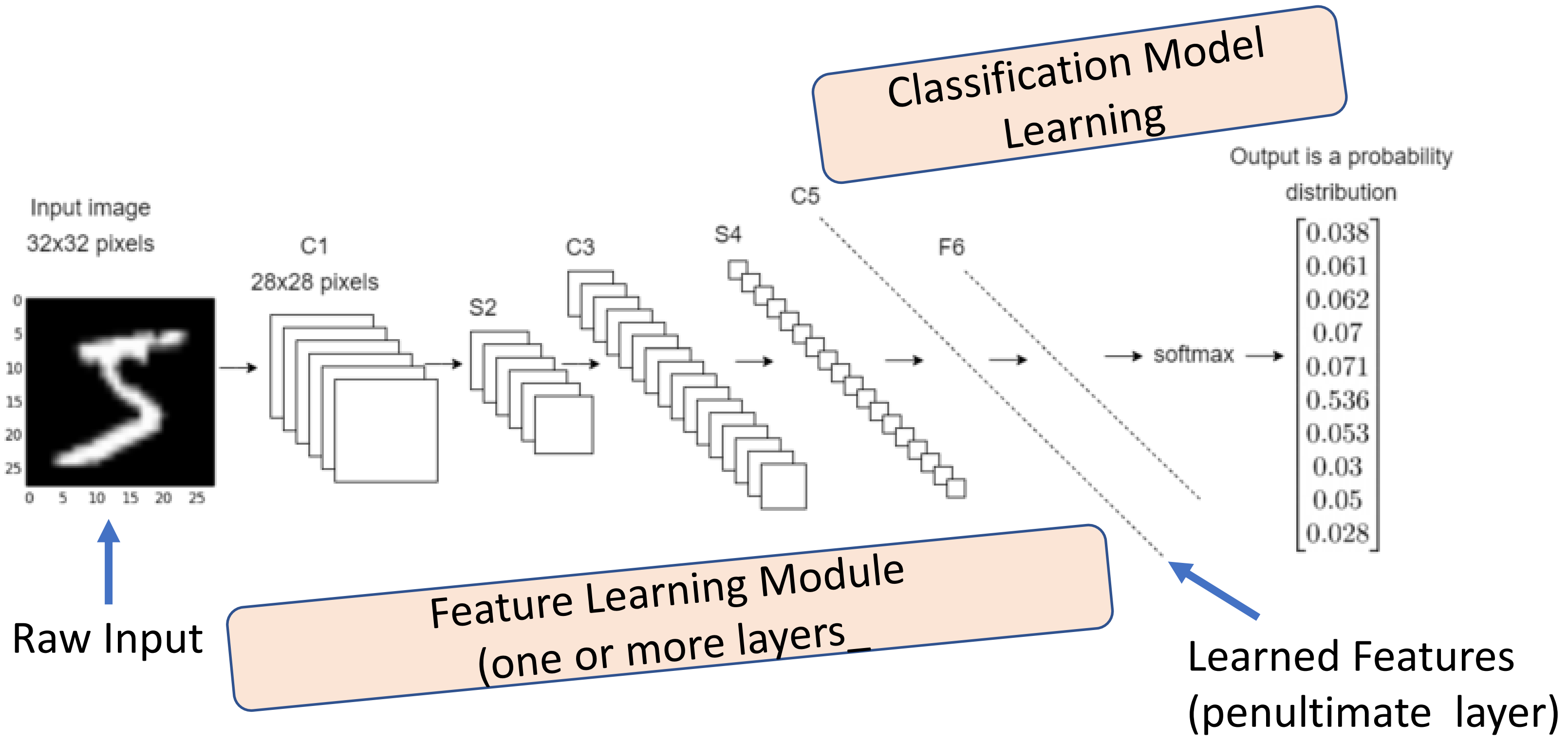
Feature extraction part is automated via the feature learning module



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/

Some Notation/Nomenclature/Convention

input-output pairs $\{(\mathbf{x_n}, y_n)\}_{n=1}^{N}$

- inputs $\{\mathbf{x_n}\}_{n=1}^{N}$
- is (usually) a vector containing the values of the features or attributes or covariates that encode properties of the it represents, e.g.,
  - can be a 49×1 vector of pixel intensities
- is the output or response or label associated with input $\mathbf{x_n}$ (and its value is known for the training inputs)

Some Notation/Nomenclature/Convention

- $\mathbf{x}$ $\mathbf{n}$, y n $\mathbf{x}$ $\mathbf{n}$ $\mathbf{xx}$ $\mathbf{x}$ $\mathbf{n}$ $\mathbf{nn}$ $\mathbf{x}$ $\mathbf{n}$, y n yy y n nn y n $\mathbf{x}$ $\mathbf{n}$, y n }{ $\mathbf{x}$ $\mathbf{n}$, y n } n=1 N *nn=1 { **x n** , y n } n=1 N NN { **x n** , y n } n=1 N*
- *Sup*. learning requires training data as N input-output pairs { $\mathbf{x}$ $\mathbf{n}$, y n } n = 1 N
- inputs $\{\mathbf{x_n}\}_{n=1}^{N}$
- is (usually) a vector containing the values of the features or attributes or covariates that encode properties of the it represents, e.g.,
  - can be a 49 × 1 vector of pixel intensities
- is the output or response or label associated with input $\mathbf{x_n}$ (and its value is known for the training inputs)

Some Notation/Nomenclature/Convention

- **x n xx x n nn x n** } { **x n** } *n*=1 *N* *nn*=1 { **x n** } *n*=1 *N* *NN* { **x n** } *n*=1 *N*
- **x n** , *y n* **x n xx x n nn x n** , *y n yy y n nn y n* **x n** , *y n* }{ **x n** , *y n* } *n*=1 *N* *nn=1 {* ***x n*** *, y n } n=1 N NN {* ***x n*** *, y n } n=1 N*
- *Sup*. learning requires training data as N input-output pairs $\{\mathbf{x}\,\mathbf{n}, y\,n\}\, n = 1\, N$
- Unsupervised learning requires training data as N inputs $\{\mathbf{x}\,\mathbf{n}\}\, n = 1\, N$
- inputs $\{\mathbf{x_n}\}_{n=1}^{N}$
- is (usually) a vector containing the values of the features or attributes or covariates that encode properties of the it represents, e.g.,
  - can be a 49 × 1 vector of pixel intensities
- is the output or response or label associated with input $\mathbf{x_n}$ (and its value is known for the training inputs)

Some Notation/Nomenclature/Convention

- $\mathbf{x}\ \mathbf{n}\ \mathbf{xx}\ \mathbf{x}\ \mathbf{n}\ \mathbf{nn}\ \mathbf{x}\ \mathbf{n}\ \}\{\ \mathbf{x}\ \mathbf{n}\ \}\ n{=}1\ N\ nn{=}1\ \{\ \mathbf{x}\ \mathbf{n}\ \}\ n{=}1\ N\ NN\ \{\ \mathbf{x}\ \mathbf{n}\ \}\ n{=}1\ N$
- $\mathbf{x}\ \mathbf{n}\ ,y\ n\ \ \mathbf{x}\ \mathbf{n}\ \mathbf{xx}\ \mathbf{x}\ \mathbf{n}\ \mathbf{nn}\ \mathbf{x}\ \mathbf{n}\ ,y\ n\ yy\ y\ n\ nn\ y\ n\ \ \mathbf{x}\ \mathbf{n}\ ,y\ n\ \ \}\{\ \mathbf{x}\ \mathbf{n}$ $nn{=}1\ \{\ \mathbf{x}\ \mathbf{n}\ ,y\ n\ \}\ n{=}1\ N\ NN\ \{\ \mathbf{x}\ \mathbf{n}\ ,y\ n\ \}\ n{=}1\ N$

RL and other flavors of ML problems also use similar notation

- *Sup*. learning requires training data as N input-output pairs $\{\ \mathbf{x}\ \mathbf{n}\ ,y\ n\ \}\ n = 1\ N$
- Unsupervised learning requires training data as N inputs $\{\ \mathbf{x}\ \mathbf{n}\ \}\ n = 1\ N$
- inputs $\{\mathbf{x}_\mathbf{n}\}_{n=1}^{N}$
- is (usually) a vector containing the values of the features or attributes or covariates that encode properties of the it represents, e.g.,
  - can be a 49 × 1 vector of pixel intensities
- is the output or response or label associated with input $\mathbf{x}_\mathbf{n}$ (and its value is known for the training inputs)

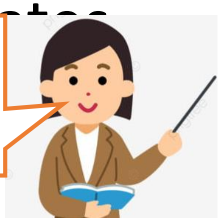
Some Notation/Nomenclature/Convention

- is (usually) a vector containing the values of the features or attributes or covariates that encode properties of the it represents, e.g.,
- $\mathbf{x}\ \mathbf{n}\ \mathbf{xx}\ \mathbf{x}\ \mathbf{n}\ \mathbf{nn}\ \mathbf{x}\ \mathbf{n}\ \}\{\ \mathbf{x}\ \mathbf{n}\ \}\ n{=}1\ N\ nn{=}1\ \{\ \mathbf{x}\ \mathbf{n}\ \}\ n{=}1\ N\ NN\ \{\ \mathbf{x}\ \mathbf{n}\ \}\ n{=}$
- $\mathbf{x}\ \mathbf{n}\ ,y\ n\ \ \mathbf{x}\ \mathbf{n}\ \mathbf{xx}\ \mathbf{x}\ \mathbf{n}\ \mathbf{nn}\ \mathbf{x}\ \mathbf{n}\ ,y\ n\ yy\ y\ n\ nn\ y\ n\ \ \mathbf{x}\ \mathbf{n}\ ,y\ n\ \}\{\ \mathbf{x}\ \mathbf{n}\ ,y\ n\ \}\ n{=}1\ N$ $nn{=}1\ \{\ \mathbf{x}\ \mathbf{n}\ ,y\ n\ \}\ n{=}1\ N\ NN\ \{\ \mathbf{x}\ \mathbf{n}\ ,y\ n\ \}\ n{=}1\ N$
- Sup. learning requires training data as N input-output pairs $\{\ \mathbf{x}\ \mathbf{n}\ ,y\ n\ \}\ n = 1\ N$
- Unsupervised learning requires training data as N inputs $\{\ \mathbf{x}\ \mathbf{n}\ \}\ n = 1\ N$
- Each input $x\ n\ \ \mathbf{n}\ \mathbf{n}\ \mathbf{n}$ is (usually) a vector containing the values of the features or attributes or covariates that encode properties of the it represents, e.g.,
- is (usually) a vector containing the values of the features or attributes or covariates that encode properties of the it represents, e.g.,
  - can be a 49 × 1 vector of pixel intensities

RL and other flavors of ML problems also use similar notation

Some Notation/Nomenclature/Convention

- can be a 49×1 vector of pixel intensities
- is (usually) a vector containing the values of the features or attributes or covariates that encode properties of the it represents, e.g.,

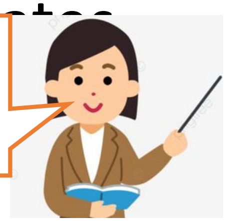


- $\mathbf{x}\ \mathbf{n}\ \mathbf{xx}\ \mathbf{x}\ \mathbf{n}\ \mathbf{nn}\ \mathbf{x}\ \mathbf{n}\ \}\{\ \mathbf{x}\ \mathbf{n}\ \}\, n{=}1\ N\ nn{=}1\ \{\ \mathbf{x}\ \mathbf{n}\ \}\, n{=}1\ N\ NN\ \{\ \mathbf{x}\ \mathbf{n}\ \}\, n{=}1\ N$
- $\mathbf{x}\ \mathbf{n}\ , y\ n\ \ \mathbf{x}\ \mathbf{n}\ \mathbf{xx}\ \mathbf{x}\ \mathbf{n}\ \mathbf{nn}\ \mathbf{x}\ \mathbf{n}\ , y\ n\ yy\ y\ n\ nn\ y\ n\ \ \mathbf{x}\ \mathbf{n}\ , y\ n\ \}\{\ \mathbf{x}\ \mathbf{n}\ , y\ n\ \}\, n{=}1\ N$ $nn{=}1\ \{\ \mathbf{x}\ \mathbf{n}\ , y\ n\ \}\, n{=}1\ N\ NN\ \{\ \mathbf{x}\ \mathbf{n}\ , y\ n\ \}\, n{=}1\ N$
- *Sup*. learning requires training data as N input-output pairs $\{\ \mathbf{x}\ \mathbf{n}\ , y\ n\ \}\, n = 1\ N$
- Unsupervised learning requires training data as N inputs $\{\ \mathbf{x}\ \mathbf{n}\ \}\, n = 1\ N$
- Each input $x\ n\ \ \mathbf{n}\ \mathbf{n}\ \mathbf{n}$ 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: $x\ n\ \ \mathbf{n}\ \mathbf{n}\ \mathbf{n}$ can be a 49×1 vector of pixel intensities
- is (usually) a vector containing the values of the features or attributes or covariates that encode properties of the it represents, e.g.,

Some Notation/Nomenclature/Convention

- can be a 49 × 1 vector of pixel intensities
- is (usually) a vector containing the values of the features or attributes or covariates that encode properties of the it represents, e.g.,

RL and other flavors of ML problems also use similar notation



- $\mathbf{x}\ \mathbf{n}\ \mathbf{xx}\ \mathbf{x}\ \mathbf{n}\ \mathbf{nn}\ \mathbf{x}\ \mathbf{n}\ \}\{\ \mathbf{x}\ \mathbf{n}\ \}\ n{=}1\ N\ nn{=}1\ \{\ \mathbf{x}\ \mathbf{n}\ \}\ n{=}1\ N\ NN\ \{\ \mathbf{x}\ \mathbf{n}\ \}\ n{=}1\ N$
- $\mathbf{x}\ \mathbf{n}\ ,y\ n\ \ \mathbf{x}\ \mathbf{n}\ \mathbf{xx}\ \mathbf{x}\ \mathbf{n}\ \mathbf{nn}\ \mathbf{x}\ \mathbf{n}\ ,y\ n\ yy\ y\ n\ nn\ y\ n\ \ \mathbf{x}\ \mathbf{n}\ ,y\ n\ \}\{\ \mathbf{x}\ \mathbf{n}\ ,y\ n\ \}\ n{=}1\ N$ $nn{=}1\ \{\ \mathbf{x}\ \mathbf{n}\ ,y\ n\ \}\ n{=}1\ N\ NN\ \{\ \mathbf{x}\ \mathbf{n}\ ,y\ n\ \}\ n{=}1\ N$
- *Sup*. learning requires training data as N input-output pairs $\{\ \mathbf{x}\ \mathbf{n}\ ,y\ n\ \}\ n =$
- Unsupervised learning requires training data as N inp

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

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

Some Notation/Nomenclature/Convention

- is the output or response or label associated with input $\mathbf{x}\,\mathbf{n}\;\mathbf{xx}\;\mathbf{x}\,\mathbf{n}\;\mathbf{nn}\;\mathbf{x}\,\mathbf{n}$ (and its value is known for the training inputs)
- can be a 49×1 vector of pixel intensities
- is (usually) a vector containing the values of the features or attributes or covariates that encode properties of the it represents, e.g.,
- $\mathbf{x}\,\mathbf{n}\;\mathbf{xx}\;\mathbf{x}\,\mathbf{n}\;\mathbf{nn}\;\mathbf{x}\,\mathbf{n}\;\}\{\,\mathbf{x}\,\mathbf{n}\,\}\,n{=}1\;N\;nn{=}1\,\{\,\mathbf{x}\,\mathbf{n}\,\}\,n{=}1\;N\;NN\,\{\,\mathbf{x}\,\mathbf{n}\,\}\,n{=}1\;N$
- $\mathbf{x}\,\mathbf{n}\,,y\,n\quad \mathbf{x}\,\mathbf{n}\;\mathbf{xx}\;\mathbf{x}\,\mathbf{n}\;\mathbf{nn}\;\mathbf{x}\,\mathbf{n}\,,y\,n\;yy\;y\,n\;nn\;y\,n\quad \mathbf{x}\,\mathbf{n}\,,y\,n\,\}\{\,\mathbf{x}\,\mathbf{n}\,,y\,n\,\}\,n{=}1$ $nn{=}1\,\{\,\mathbf{x}\,\mathbf{n}\,,y\,n\,\}\,n{=}1\;N\;NN\,\{\,\mathbf{x}\,\mathbf{n}\,,y\,n\,\}\,n{=}1\;N$
- *Sup.* learning requires training data as N input-output pairs $\{\,\mathbf{x}\,\mathbf{n}\,,y\,n\,\}\,n = 1\;N$
- Unsupervised learning requires training data as N inputs $\{\,\mathbf{x}\,\mathbf{n}\,\}\,n = 1\;N$
- Each input $x\,n\quad \mathbf{n}\,\mathbf{n}\,\mathbf{n}$ 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: $x\,n\quad \mathbf{n}\,\mathbf{n}\,\mathbf{n}$ can be a 49×1 vector of pixel intensities

RL and other flavors of ML problems also use similar notation

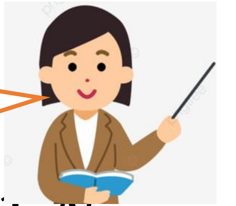
Size or length of the input $\mathbf{x}_\mathbf{n}$ is commonly known as data/input dimensionality or feature dimensionality

Some Notation/Nomenclature/Convention

- is the output or response or label associated with input $\mathbf{x\,n\,xx\,x\,n\,nn\,x\,n}$ (and its value is known for the training inputs)
- can be a 49 × 1 vector of pixel intensities
- is (usually) a vector containing the values of the features or attributes or covariates that encode properties of the it represents, e.g.,
- $\mathbf{x\,n\,xx\,x\,n\,nn\,x\,n}\,\}\,\{\,\mathbf{x\,n}\,\}\,n{=}1\;N\;nn{=}1\,\{\,\mathbf{x\,n}\,\}\,n{=}1\;N\;NN\,\{\,\mathbf{x\,n}\,\}\,n{=}1\;N$
- $\mathbf{x\,n}\,,y\,n\;\;\mathbf{x\,n\,xx\,x\,n\,nn\,x\,n}\,,y\,n\,yy\,y\,n\,nn\,y\,n\;\;\mathbf{x\,n}\,,y\,n\;\}\{\;\mathbf{x\,n}\,,y\,n\;\}\,n{=}1$ $nn{=}1\,\{\;\mathbf{x\,n}\,,y\,n\;\}\,n{=}1\;N\;NN\,\{\;\mathbf{x\,n}\,,y\,n\;\}\,n{=}1\;N$
- *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 $\{\,\mathbf{x\,n}\,\}\,n=1\;N$
- Each input $x\,n\;\;\mathbf{n\,n\,n}$ 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: $x\,n\;\;\mathbf{n\,n\,n}$ can be a 49 × 1 vector of pixel intensities

RL and other flavors of ML problems also use similar notation

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



Types of Features and Types of Outputs

Types of Features and Types of Outputs

- Features as well as outputs can be real-valued, binary, categorical, ordinal, etc.

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

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

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

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

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.)

Some Basic Operations of Inputs

$\in R^D$ to of size D

- inputs $\{\mathbf{x}_n\}_{n=1}^N$, their average or mean can be computed as
  $= \frac{1}{N}\sum_{n=1}^N \mathbf{x}_n$
- and $\mathbf{x}_m$
- and the mean $\boldsymbol{\mu}$ of all inputs

Some Basic Operations of Inputs

- $R\ D\ RR\ R\ D\ DD\ R\ D$ to of size *D*
- *Assume* each input feature vector $x\ n \in\ n\ n\ n\ \in R^D$ to of size *D*

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

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

- and $\mathbf{x}_m$

- and the mean $\boldsymbol{\mu}$ of all inputs

Some Basic Operations of Inputs

- $\mathbf{x}\,n\,\mathbf{xx}\,\mathbf{x}\,n\,nn\,\mathbf{x}\,n\,\}\{\,\mathbf{x}\,n\,\}\,n$=1, $N\,nn$=1, $\{\,\mathbf{x}\,n\,\}\,n$=1, $N\,NN\,\{\,\mathbf{x}\,n\,\}\,n$=1, N their average or mean can be computed as
- $R\,D\,RR\,R\,D\,DD\,R\,D$ to of size *D*
- *Assume* each input feature vector $x\,n \in\ n\,n\,n\ \in R^D$ to of size *D*
- Given N inputs $\{\,\mathbf{x}\,n\,\}\,n = 1, N$ their average or mean can be computed as
- inputs $\{\mathbf{x}_n\}_{n=1,}^{N}$ their average or mean can be computed as

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

- and $\mathbf{x}_m$
- and the mean μ of all inputs

Some Basic Operations of Inputs

- $1\ N\ 1\ 1\ N\ NN\ 1\ N\ \ n{=}1\ N\ \ \mathbf{x}\ n\ \ nn{=}1\ n{=}1\ N\ \ \mathbf{x}\ n\ \ NN\ n{=}1\ N\ \ \mathbf{x}\ n\ \ \ \mathbf{x}\ n\ \mathbf{xx}\ \mathbf{x}\ n\ nn\ \mathbf{x}\ n\ \ n{=}1\ N$ $\mathbf{x}\ n$
- $\mathbf{x}\ n\ \mathbf{xx}\ \mathbf{x}\ n\ nn\ \mathbf{x}\ n\ \}\ \{\ \mathbf{x}\ n\ \}\ n{=}1,\ N\ nn{=}1,\ \{\ \mathbf{x}\ n\ \}\ n{=}1,\ N\ NN\ \{\ \mathbf{x}\ n\ \}\ n{=}1,\ N$ their average or mean can be computed as
- $R\ D\ RR\ R\ D\ DD\ R\ D$ to of size *D*
- *Assume* each input feature vector $x\ n \in\ \ n\ n\ n\ \ \in R^D$ to of size *D*

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

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

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

- and $\mathbf{x}_m$

Some Basic Operations of Inputs

- $1 N 1 1 N NN 1 N \;\; n{=}1 N \;\; \mathbf{x} n \;\; nn{=}1 n{=}1 N \;\; \mathbf{x} n \;\; NN n{=}1 N \;\; \mathbf{x} n \;\; \mathbf{x} n \mathbf{xx} \mathbf{x} n nn \mathbf{x} n \;\; n{=}1 N$ $\mathbf{x} n$
- $\mathbf{x} n \mathbf{xx} \mathbf{x} n nn \mathbf{x} n \} \{ \mathbf{x} n \} n{=}1, N nn{=}1, \{ \mathbf{x} n \} n{=}1, N NN \{ \mathbf{x} n \} n{=}1$ mean can be computed as

What does such a
“mean” represent?

- $R D RR R D DD R D$ to of size *D*
- *Assume* each input feature vector $x n \in \;\; n n n \;\; \in R^D$ to of size *D*

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

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

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

- and $\mathbf{x}_m$

Some Basic Operations of Inputs

- 1 N 1 1 N NN 1 N n=1 N $\mathbf{x}$ n nn=1 n=1 N $\mathbf{x}$ n NN n=1 N $\mathbf{x}$ n $\mathbf{x}$ n $\mathbf{xx}$ $\mathbf{x}$ n nn $\mathbf{x}$ n n=1 N $\mathbf{x}$ n
- $\mathbf{x}$ n $\mathbf{xx}$ $\mathbf{x}$ n nn $\mathbf{x}$ n } { $\mathbf{x}$ n } n=1, N nn=1, { $\mathbf{x}$ n } n=1, N NN { $\mathbf{x}$ n } n=1 mean can be computed as
- R D RR R D DD R D to of size *D*
- *Assume* each input feature vector x n ∈ n n n ∈ R^D to of si

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

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

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

- and $\mathbf{x}_m$

What does such a "mean" represent?

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

Some Basic Operations of Inputs

- and $\mathbf{x}\, m\, \mathbf{xx}\, \mathbf{x}\, m\, mm\, \mathbf{x}\, m$
- $1\, N\, 1\, 1\, N\, NN\, 1\, N\;\; n{=}1\, N\;\; \mathbf{x}\, n\;\; nn{=}1\, n{=}1\, N\;\; \mathbf{x}\, n\;\; NN\, n{=}1\, N\;\; \mathbf{x}\, n\;\; \mathbf{x}\, n$ $n{=}1\, N$ $\mathbf{x}\, n$
- $\mathbf{x}\, n\, \mathbf{xx}\, \mathbf{x}\, n\, nn\, \mathbf{x}\, n\, \}\, \{\, \mathbf{x}\, n\, \}\, n{=}1, N\, nn{=}1, \{\, \mathbf{x}\, n\, \}\, n{=}1, N\, NN\, \{\, \mathbf{x}\, n\, \}\, n{=}1, N$ their avera mean can be computed as
- $R\, D\, RR\, R\, D\, DD\, R\, D$ to of size *D*
- *Assume* each input feature vector $x\, n \in\;\; n\, n\, n\;\; \in R^D$ to of size *D*

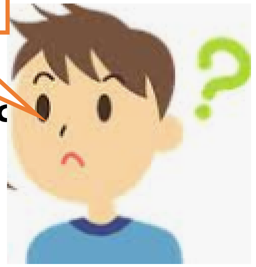
$$\mu = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n$$

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

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

- and $\mathbf{x}_m$

What does such a "mean" represent?



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

Some Basic Operations of Inputs

- and $\mathbf{x}\, m\, \mathbf{xx}\, \mathbf{x}\, m\, mm\, \mathbf{x}\, m$
- $1\, N\, 1\, 1\, N\, NN\, 1\, N\;\; n{=}1\, N\;\; \mathbf{x}\, n\;\; nn{=}1\, n{=}1\, N\;\; \mathbf{x}\, n\;\; NN\, n{=}1\, N\;\; \mathbf{x}\, n\;\; \mathbf{x}\, n$ $n{=}1\, N$ $\mathbf{x}\, n$
- $\mathbf{x}\, n\, \mathbf{xx}\, \mathbf{x}\, n\, nn\, \mathbf{x}\, n\, \}\, \{\, \mathbf{x}\, n\, \}\, n{=}1, N\, nn{=}1, \{\, \mathbf{x}\, n\, \}\, n{=}1, N\, NN\, \{\, \mathbf{x}\, n\, \}\, n{=}1, N$ their avera mean can be computed as
- $R\, D\, RR\, R\, D\, DD\, R\, D$ to of size D
- *Assume* each input feature vector $x\, n \in\;\; n\, n\, n\;\; \in R^D$ to of size D
- Can cc$d(\boldsymbol{x}_n, \boldsymbol{x}_m) = ||\boldsymbol{x}_n - \boldsymbol{x}_m|| = \sqrt{(\boldsymbol{x}_n - \boldsymbol{x}_m)^\top (\boldsymbol{x}_n - \boldsymbol{x}_m)} = \sqrt{\sum_{d=1}^{D} (x_{nd} - x_{md})^2}$ n and $\mathbf{x}_m$

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

- and $\mathbf{x}_m$

Some Basic Operations of Inputs

- and the mean $\boldsymbol{\mu}\boldsymbol{\mu}$ of all inputs
- and $\mathbf{x}\, m\, \mathbf{xx}\, \mathbf{x}\, m\, mm\, \mathbf{x}\, m$
- $1\, N\, 1\, 1\, N\, NN\, 1\, N\;\; n{=}1\, N\;\; \mathbf{x}\, n\;\; nn{=}1\, n{=}1\, N\;\; \mathbf{x}\, n\;\; NN\, n{=}1\, N\;\; \mathbf{x}\, n\;\;\; \mathbf{x}\, n$ xx x n n n
  $\mathbf{x}\, n$
- $\mathbf{x}\, n\, \mathbf{xx}\, \mathbf{x}\, n\, nn\, \mathbf{x}\, n\, \}\, \{\, \mathbf{x}\, n\, \}\, n{=}1,\, N\, nn{=}1,\, \{\, \mathbf{x}\, n\, \}\, n{=}1,\, N\, NN\, \{\, \mathbf{x}$ ra
  mean can be computed as
- $R\, D\, RR\, R\, D\, DD\, R\, D$ to of size D
- *Assur*

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

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

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

- or Euclidean distance between an input $x\, n\;\; n\, n\, n$ and the mean $\boldsymbol{\mu}$ of all inputs

What does such a "mean" represent?

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

Some Basic Operations of Inputs

- and the mean $\boldsymbol{\mu\mu}$ of all inputs
- and $\mathbf{x}\,m\,\mathbf{xx}\,\mathbf{x}\,m\,mm\,\mathbf{x}\,m$
- $1\,N\,1\,1\,N\,NN\,1\,N\;\;n{=}1\,N\;\;\mathbf{x}\,n\;\;nn{=}1\,n{=}1\,N\;\;\mathbf{x}\,n\;\;NN\,n{=}1\,N\;\;\mathbf{x}\,n\;\;\;\mathbf{x}\,n\,\mathbf{xx}\,\mathbf{x}\,n\,n$ … n $\mathbf{x}\,n$
- $\mathbf{x}\,n\,\mathbf{xx}\,\mathbf{x}\,n\,nn\,\mathbf{x}\,n\,\}\,\{\,\mathbf{x}\,n\,\}\,n{=}1,N\,nn{=}1,\{\,\mathbf{x}\,n\,\}\,n{=}1,N\,NN\,\{\,\mathbf{x}$ … ra mean can be computed as
- $R\,D\,RR\,R\,D\,DD\,R\,D$ to of size D
- *Assur*

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

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

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

- or Euclidean distance between an input $x\,n\;\;n\,n\,n$ and the mean $\boldsymbol{\mu}$ of all inputs

Next Class

- Introduction to Supervised Learning
- A simple Supervised Learning algorithm based on computing distances