Lecture 2

Introducing Machine Learning

Why Machine Learning?

We are entering the era of **big data when** vast amounts of data are automatically created and analyzed**.** For example, there are more than 1 trillion web pages; the genomes of thousands of people, each of which has a length of 3.8×10^9 base pairs, have been sequenced by various labs; Walmart handles more than 1M transactions per hour and has databases containing more than 2.5 petabytes (2.5×10^{15}) of information; and so on. This deluge of data calls for automated methods of data analysis, which is what **machine learning (ML)** provides.

Learning, like intelligence, covers a broad range of processes that it is difficult to define precisely. A dictionary definition includes phrases such as "*to gain knowledge, or understanding of, or skill in, by study, instruction, or experience,"* and *"modification of a behavioral tendency by experience*." Zoologists and psychologists study learning in animals and humans. In this course, we focus on learning in machines. There are several parallels between animal, human and machine learning. Certainly, many techniques in machine learning derive from the efforts of psychologists to make more precise their theories of animal and human learning through computational models. It seems likely also that the concepts and techniques being explored by researchers in machine learning may illuminate certain aspects of biological learning.

Why should machines have to learn?

One might ask: “Why should machines have to learn?” There are several reasons why we need ML:

- Some tasks cannot be defined well except by example; that is, we might be able to specify input/output pairs but not a concise relationship between inputs and desired outputs. We would like machines to be able to adjust their internal structure to produce correct outputs for a large number of inputs.
- It is possible that important relationships and correlations are hidden among large piles of data. ML methods can often be used to extract these relationships.
- Designers often produce machines that do not work as well as desired in the environments in which they are used. In fact, certain characteristics of the working environment might not be completely known at design time. ML methods an be used for on-the-job improvement of existing machine designs.

Why should machines have to learn? *(cont.)*

- The amount of knowledge available about certain tasks might be too large for explicit encoding by humans. Machines that learn this knowledge gradually might be able to capture more of it than humans would want to write down.
- Environments change over time. Machines that can adapt to a changing environment would reduce the need for constant redesign.
- New knowledge about tasks is constantly being discovered by humans. There is a constant stream of new events in the world. Continuing redesign of AI systems to conform to new knowledge is impractical, but machine learning methods might be able to track much of it.

Some examples of ML

Examples:

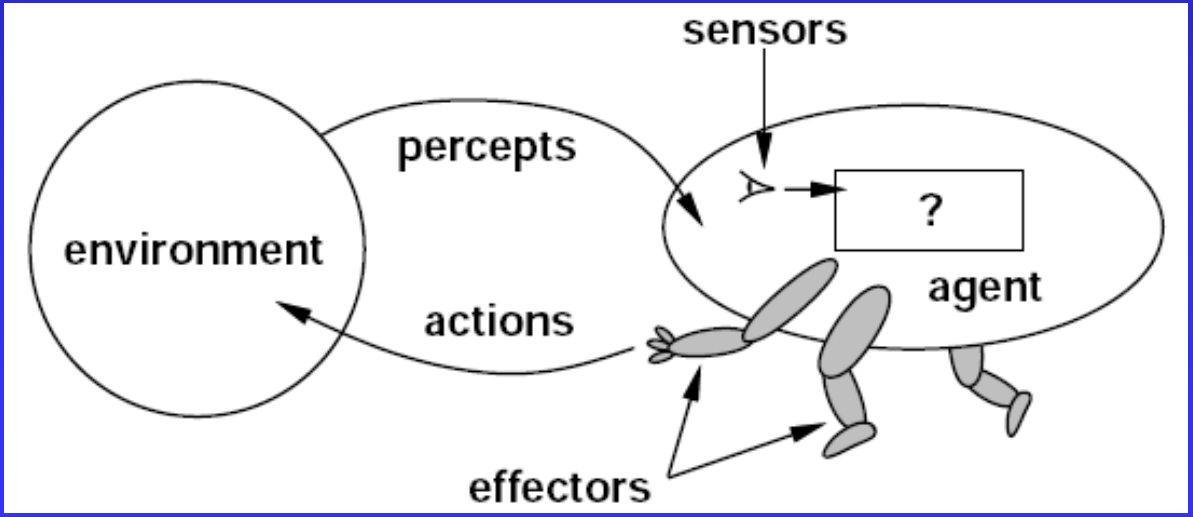
- Learning to play chess
- Learning to recognize spoken words
- Learning to drive an autonomous vehicle. The early systems for driving autonomous vehicles date back to 1980s. For example, the **ALVINN** system has used its learned strategies to drive unassisted at 70 miles per hour for 90 miles on public highways among other cars.
- Learning to classify new astronomical structures
- Identification of spam messages in e-mail
- Segmentation of customer behavior for targeted advertising
- Weather forecasts
- Detecting fraudulent credit card transactions
- Detecting computer and network intrusions
- Stock market analysis
- Genetic sequencing
- Exploration of human brain
- Aircraft and rocket design
- Solving complex optimization problems, and so on and so on…

Tom Mitchell's definition of ML

According to Tom Mitchel, one of the pioneers in ML: *"A computer program is said to learn from experience E with respect to some class of tasks **T** and performance measure **P**, if its performance at tasks in **T**, as measured by **P**, improves with experience **E**."* The definition emphasizes three important elements: learning is goal directed, i.e., it happens with respect to some task or a class of tasks, **T**. That is, machines learn in order to improve their performance at specific tasks. To measure how well machines learn, we need a formal criteria of efficiency, the performance measure **P**. The performance measure helps us differentiate between good learners and bad learners. Finally, learning happens through experience, **E**. Even, when you read a book, you are still involved in some experience of direct knowledge acquisition. Learning through experience could be **offline**: there is a special learning phase and/or environment, which is different from the real-world production phase. For example, a surgeon may be involved in offline learning by reading and watching how other surgeons operate. **Online** learning involves a real-world setting, where mistakes are costly. For example, a surgeon can still continue to learn after he starts operating on real patients.

Nils Nilsson's definition of ML

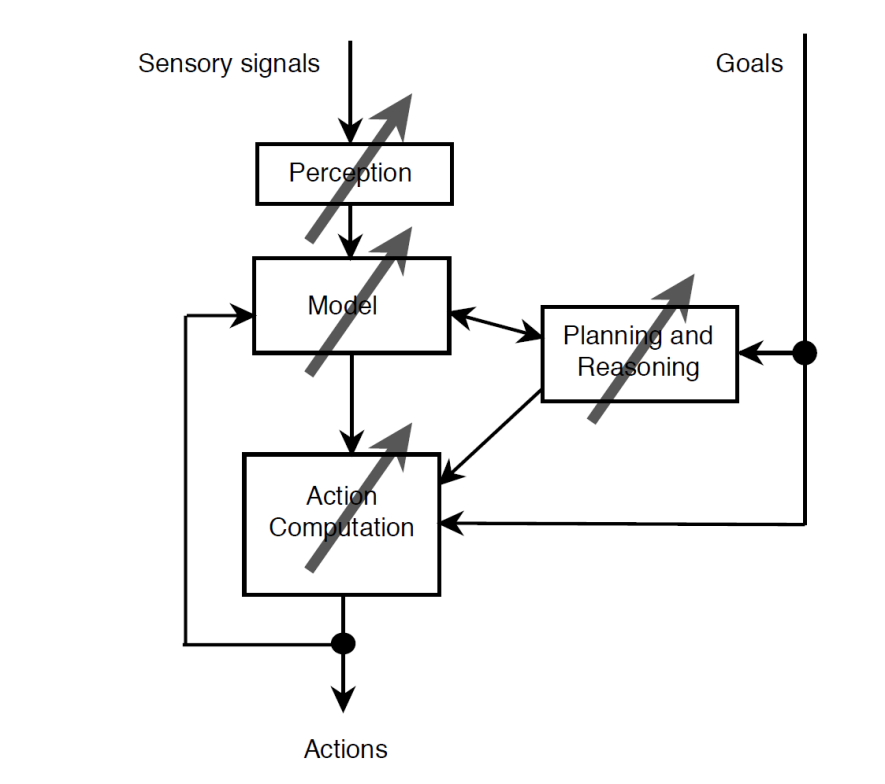
According to Nils Nilsson, *"a machine learns whenever it changes its structure, program, or data (based on its inputs or in response to external information) in such a manner that its expected future performance improves."* Although this definition does not explicitly mention experience and a performance measure, it is very similar to Tom Mitchell's definition. One important aspect of this definition is that ML is based on inputs or happens in response to external information. The definition is aligned with classic notions in Artificial Intelligence, such as perception, interaction with the environment, planning, reasoning and action. According to the standard paradigm in Artificial Intelligence, a rational agent perceives its environment and computes appropriate actions, perhaps by anticipating their effects.



Source: Russel and Norvig

Nils Nilsson's definition of ML *(cont.)*

Changes made to any of the components of a rational agent shown in the next figure might count as learning. Different learning mechanisms might be employed depending on which subsystem is being changed. For example, learning can happen at sensors level or at reasoning level.



Source: Nils Nilsson

Kevin Murphy's definition of ML

Kevin Murphy defines machine learning "*as a set of methods that can automatically detect patterns in data, and then use the uncovered patterns to predict future data, or to perform other kinds of decision making under uncertainty (such as planning how to collect more data!).*"

Murphy's definition emphasizes other important aspect of ML: predicting future data and decision making under uncertainty. Decision making is at the core of intelligent behavior and it inherently involves dealing with uncertainty. We all leave in an uncertain world and things change as we speak.

Many ML researchers adopt the view that the best way to solve ML problems is to use the tools of probability theory and statistics. Probability theory can be applied to any problem involving uncertainty. In machine learning, uncertainty comes in many forms: what is the best prediction about the future given some past data? what is the best model to explain some data? what measurement should I perform next? etc. The probabilistic approach to machine learning is closely related to the field of statistics, but differs slightly in terms of its emphasis and terminology.

Sources of ML

Work in machine learning is coming from several disciplines, each discipline bringing in different methods which are now being assimilated into a more unified discipline. Main sources of ML:

- **Statistics:** A long-standing problem in statistics is how best to use samples drawn from unknown probability distributions to help decide from which distribution some new sample is drawn. A related problem is how to estimate the value of an unknown function at a new point given the values of this function at a set of sample points.
- **Neuroscience**: Non-linear elements with weighted inputs have been suggested as simple models of biological neurons. Several important machine learning techniques are based on networks of nonlinear elements, often called neural networks. Work inspired by this school is sometimes called connectionism, brain-style computation, or sub-symbolic processing.
- **Adaptive Control Theory:** Control theorists study the problem of controlling a process having unknown parameters which must be estimated during operation. Often, the parameters change during operation, and the control process must track these changes. Some aspects of controlling a robot based on sensory inputs represent instances of this sort of problem.

Sources of ML *(cont.)*

- **Psychological Models:** Psychologists have studied the performance of humans in various learning tasks. For example, some of the work in reinforcement learning can be traced to efforts to model how reward stimuli influence the learning of goal-seeking behavior in humans and animals.
- **Artificial Intelligence:** From the beginning, AI research has been concerned with machine learning.
- **Evolutionary models:** In nature, not only do individual animals learn to perform better, but species evolve to be better in their individual niches. Since the distinction between evolving and learning can be blurred in computer systems, techniques that model certain aspects of biological evolution have been proposed as learning methods to improve the performance of computer programs. Genetic algorithms and genetic programming are the most prominent computational techniques for evolution.

Types of machine learning: supervised learning

Machine learning is usually divided into two main types. In the **predictive** or **supervised learning** approach, the goal is to learn a mapping from inputs **x** to outputs y, given a labeled set of input-output pairs
$D = \{(\mathbf{x}_1, \mathbf{y}_1), (\mathbf{x}_2, \mathbf{y}_2), \ldots, (\mathbf{x}_N, \mathbf{y}_N)\}$
Here D is called the **training set**, and N is the number of training examples. In the simplest setting, each training input $\mathbf{x}_i$ is a multidimensional vector of numbers, representing, say, the height and weight of a person. These are called **features** or **attributes**. In general, $\mathbf{x}_i$ could be a complex structured object, such as an image, a sentence, an email message, a time series, a molecular shape, a graph, etc.

The form of the output or **response variable** (also known as dependent variable) can in principle be anything, but some methods assume that y_i is a **categorical** or **nominal** variable from some finite set, $y_i \in \{1, \ldots, C\}$ (such as male or female), or that y_i is a real-valued scalar (such as income level). When y_i is categorical, the problem is known as **classification**, and when y_i is real-valued, the problem is known as **regression**.

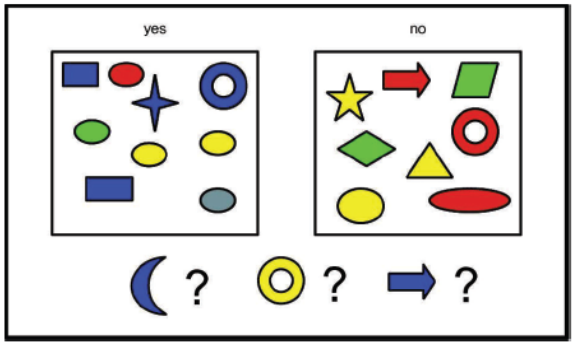
Types of machine learning: unsupervised and reinforcement learning

The second main type of machine learning is the **descriptive** or **unsupervised learning** approach. Here we are only given inputs, $D = \{\mathbf{x}_1, \mathbf{x}_{2, \ldots,} \mathbf{x}_N\}$ and the goal is to find "interesting patterns" in the data. This is sometimes called **knowledge discovery**. This is a much less well-defined problem, since we are not told what kinds of patterns to look for, and there is no obvious error metric to use (unlike supervised learning, where we can compare our prediction of ***y*** for a given **x** to the observed value).

There is a third type of machine learning, known as **reinforcement learning**. It is useful for learning how to act or behave when given reward or punishment signals. Reinforcement learning is inspired by behaviorist psychology and it is concerned with how a rational agent can maximize his expected utility. Utility is understood as a generalized form of reward, including pleasure, profits, etc. In other words, rational agents learn how to get more rewards and how to avoid punishments in their interaction with the environment and with the other agents. Examples include learning how to trade on a stock market, how a baby learns to walk, etc.

Classification

In classification, the goal is to learn a mapping from inputs **x** to outputs y, where $y \in \{1, \ldots, C\}$, with C being the number of classes. As a simple toy example of classification, consider the problem illustrated in the next figure. We have two classes of objects which correspond to labels 0 and 1. The inputs are colored shapes. These have been described by a set of D features or attributes, which are stored in an $N \times D$ matrix **X**. The input features **x** can be discrete, continuous or a combination of the two. In addition to the inputs, we have a vector of training labels **y**. The test cases are a blue crescent, a yellow circle and a blue arrow. None of these have been seen before. Thus we are required to **generalize** beyond the training set.



(a)

D features (attributes)

N cases

Color	Shape	Size (cm)	Label
Blue	Square	10	1
Red	Ellipse	2.4	1
Red	Ellipse	20.7	0

(b)

Source: Kevin Murphy

Classification *(cont.)*

One reasonable guess is that blue crescent should be $y = 1$, since all blue shapes are labeled 1 in the training set. The yellow circle is harder to classify, since some yellow things are labeled $y = 1$ and some are labeled $y = 0$, and some circles are labeled $y = 1$ and some $y = 0$ ("yes" stands for 1 and "no" stands for 0 in the figure).

To handle ambiguous cases, such as the yellow circle, it is desirable to use probability theory. We can compute the conditional probability that the class of the given input **x** is $\mathbf{c}_i$ based on the training set **D**:

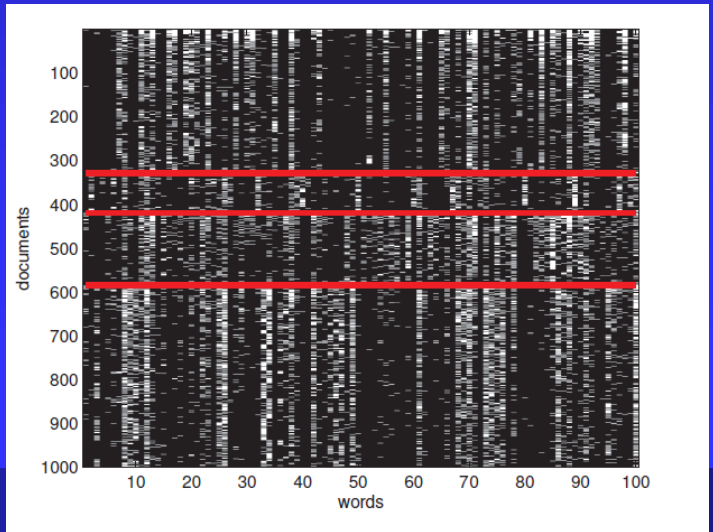
$P(y = c_i|\mathbf{x}, D)$

We can compute the conditional probabilities for all possible classes $\mathbf{c}_i$ and choose as an answer the class with the maximum probability. This is also known as a **MAP estimate** (MAP stands for **maximum a posteriori)**. In other words, we compute our "best guess" as to the "true label" of **x**. In 2011, IBM unveiled a computer system called Watson which beat the top human Jeopardy champion. Watson uses a variety of interesting techniques, including one that estimates how confident it is of its answer. The system only chooses to "buzz in" its answer if it is sufficiently confident that the answer is correct. Similarly, Google has a system known as SmartASS (ad selection system) that predicts the probability you will click on an ad based on your search history and other user and ad-specific features. This probability is known as the **click-through rate** or **CTR**, and can be used to maximize expected profit.

Classification *(cont.)*

Another real-world application of classification are **document classification and email spam filtering.** In **document classification**, the goal is to classify a document, such as a web page or email message, into one of C classes, that is, to compute $p(y = c|\mathbf{x},D)$, where $\mathbf{x}$ is some representation of the text. A special case of this is **email spam filtering**, where the classes are spam $y = 1$ or ham $y = 0$.

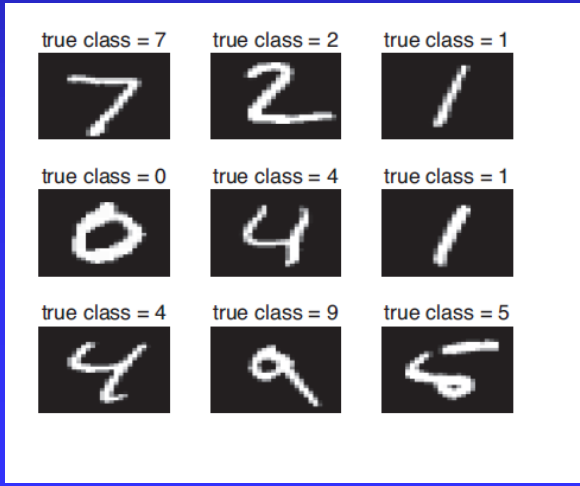
A common way to represent variable-length documents in feature-vector format is to use a **bag of words** representation. The basic idea is to define $x_{ij} = 1$ iff word j occurs in document i. If we apply this transformation to every document in our data set, we get a binary (document × word co-occurrence) matrix as the one presented on the figure below. Essentially the document classification problem has been reduced to one that looks for subtle changes in the pattern of bits. For example, we may notice that most spam messages have a high probability of containing the words “buy”, “cheap”, “viagra”, etc.



Source: Kevin Murphy

Classification *(cont.)*

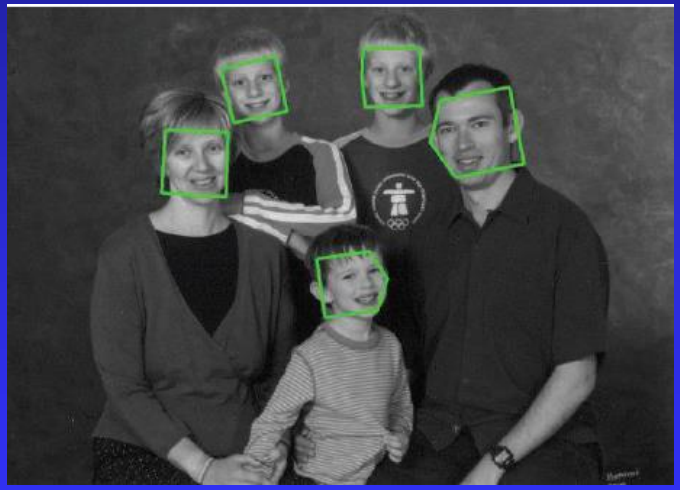
Another example of classification is **handwriting recognition**. A standard dataset used in handwriting is known as **MNIST**, which stands for “Modified National Institute of Standards”. This dataset contains 60,000 training images and 10,000 test images of the digits 0 to 9, as written by various people. The images are size 28×28 and have grayscale values in the range 0 : 255. The next figure shows examples of such images.



Source: Kevin Murphy

Classification *(cont.)*

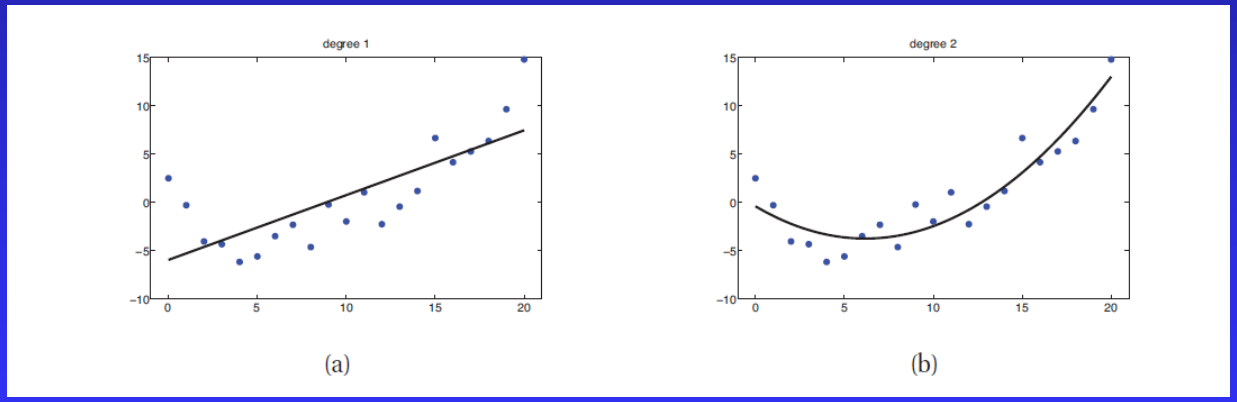
Another example of classification is **face detection and recognition.** One approach to face detection is to divide the image into many small overlapping patches at different locations, scales and orientations, and to classify each such patch based on whether it contains face-like texture or not. This is called a **sliding window detector**. The system then returns those locations where the probability of face is sufficiently high.



After we have detected faces, we can perform **face recognition**, which means estimating the identity of the person In this case, the number of class labels might be very large.

Regression

Regression is just like classification except the response (dependent) variable we want to label is continuous. The next figure shows a simple example: we have a single real-valued input $x_i \in \mathrm{R}$, and a single real-valued response $y_i \in \mathrm{R}$. We consider fitting two models to the data: a straight line and a quadratic function. Then, we can use the straight line or the quadratic function to predict the value y_i for unknown x_i.



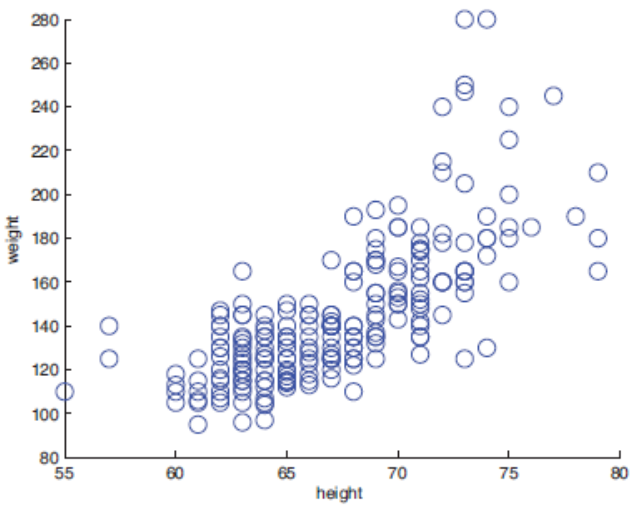
Regression *(cont.)*

Examples of real-world regression problems include:

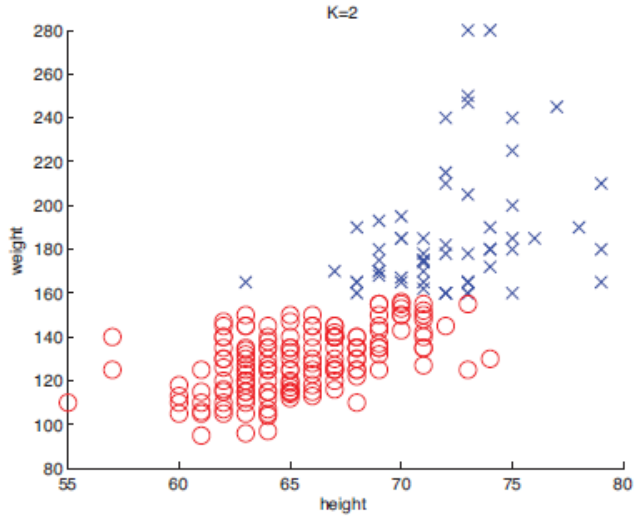
- Predict tomorrow's stock market price given current market conditions and other possible side information.
- Predict the age of a viewer watching a given video on YouTube.
- Predict the location in 3d space of a robot arm end effector, given control signals (torques) sent to its various motors.
- Predict the amount of prostate specific antigen (PSA) in the body as a function of a number
- of different clinical measurements.
- Predict the temperature at any location inside a building using weather data, time, door sensors, etc.

Clustering

As a canonical example of unsupervised learning, consider the problem of **clustering** data into groups. For example, the figure below plots some 2D data, representing the height and weight of a group of 210 people. The first task is to determine the number of clusters. The second task is to estimate which cluster each point belongs to. You can see two clusters identified on the figure to the right and their corresponding points.



(a)



(b)

Source: Kevin Murphy

Market basket analysis

Market basket analysis is another example of unsupervised learning. The data consists of a (typically very large but sparse) binary matrix, where each column represents an item or product, and each row represents a transaction. We set the element on the i^{th} row and the j^{th} column $x_{ij} = 1$ if item j was purchased on the i^{th} transaction. Many items are purchased together (e.g., bread and butter), so there will be correlations amongst the bits. Given a new partially observed bit vector, representing a subset of items that the consumer has bought, the goal is to predict which other bits are likely to turn on, representing other items the consumer might be likely to buy.

It is common to solve such tasks using **frequent itemset mining**, which create association rules. An association rule might look as follows:

{peanut butter, jelly} → {bread}

The association rule states that if peanut butter and jelly are purchased together, then bread is also likely to be purchased.

ML and data mining

Data mining, a field closely related to ML, is concerned with the generation of novel insights from large datasets. The main differences between ML and data mining are:

- ML focuses on prediction and learning from known data.

- Data mining focuses on data exploration and the discovery of new properties of data.

Almost all data mining involves the use of machine learning, but not all machine learning involves data mining. Data mining borrows several unsupervised methods from ML, such as cluster analysis.

The ML components

In general, ML includes the following interrelated components:

- **Data:** utilizes observation, memory, and recall to provide a factual basis for further reasoning and learning.
- **Abstraction** involves the translation of stored data into conceptual model that represents the main concepts represented in data and the relationships between them.
- **Generalization** uses the model to derive new knowledge to be used in decision making and action.
- **Evaluation:** evaluates the overall learning quality, including accuracy and provides recommendations for potential improvements.

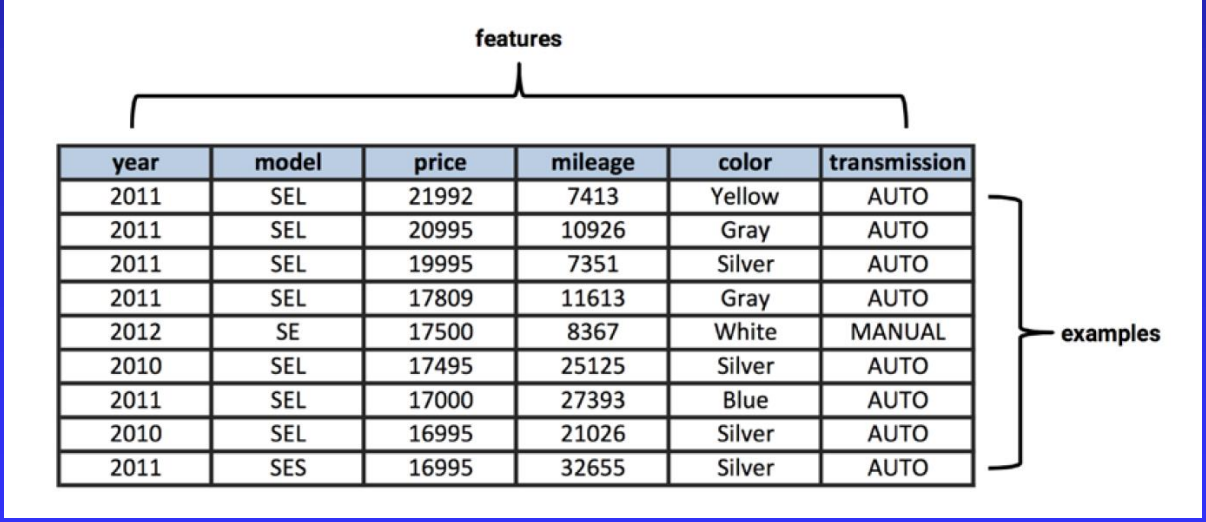
The ML process

The ML process usually involves the following main steps:

- Data collection
- Data preparation and exploration
- Choosing a model
- Choosing an algorithm
- Training the model
- Generalization: using the model to solve a given problem
- Model evaluation
- Model improvement

Data

Data sets are made up of data objects. A **data object** represents an entity. For example, in a car sales scenario, the objects may be customers, cars, and sales. Data objects are typically described by *attributes.* Data objects can also be referred to as *samples, examples, instances, data points*, or *objects*. An **attribute** is a data field, representing a characteristic or feature of a data object. The terms *attribute*, *dimension*, *feature*, and *variable* are often used interchangeably in the literature. The following figure shows a dataset in a matrix format:



year	model	price	mileage	color	transmission
2011	SEL	21992	7413	Yellow	AUTO
2011	SEL	20995	10926	Gray	AUTO
2011	SEL	19995	7351	Silver	AUTO
2011	SEL	17809	11613	Gray	AUTO
2012	SE	17500	8367	White	MANUAL
2010	SEL	17495	25125	Silver	AUTO
2011	SEL	17000	27393	Blue	AUTO
2010	SEL	16995	21026	Silver	AUTO
2011	SES	16995	32655	Silver	AUTO

From features to models

Many researchers believe that **ML is all about using the right features to build the right models that achieve the right tasks.**

In essence, **features** define a ‘ language’ in which we describe the relevant objects in our domain. We should not normally have to go back to the domain objects themselves once we have a suitable feature representation, which is why features play such an important role in machine learning. Therefore, it is important to identify the right set of features to solve a particular task

A **task** is an abstract representation of a problem we want to solve regarding those domain objects: the most common form of these is classifying them into two or more classes. Many of these tasks can be represented as a mapping from data points to outputs. This mapping or the model is itself produced as the output of a machine learning algorithm applied to training data; there is a wide variety of models to choose from.

Nominal attributes

The **type** of an attribute is determined by the set of possible values, nominal, binary, ordinal, or numeric, the attribute can have.

Nominal means “relating to names.” The values of a **nominal attribute** are symbols or *names*. Each value represents some kind of category or code, and so nominal attributes are also referred to as **categorical**. The values do not have any meaningful order. In computer science, the values are also known as *enumerations*.

Suppose that *hair color* and *marital status* are two attributes describing *person* objects. In our application, possible values for *hair color* are *black*, *brown*, *blond*, *red*, *auburn*, *gray*, and *white*. The attribute *marital status* can take on the values *single, married, divorced*, and *widowed*. Both *hair color* and *marital status* are nominal attributes. Another example of a nominal attribute is *occupation*, with the values *teacher, dentist, programmer, farmer*, and so on.

Because nominal attribute values do not have any meaningful order about them and are not quantitative, it makes no sense to find the mean (average) value or median (middle) value for such an attribute.

Binary Attributes

A **binary attribute** is a nominal attribute with only two categories or states: 0 or 1, where 0 typically means that the attribute is absent, and 1 means that it is present. Binary attributes are referred to as **Boolean** if the two states correspond to *true* and *false*.

Given the attribute *smoker* describing a *patient* object, 1 indicates that the patient smokes, while 0 indicates that the patient does not.

Ordinal Attributes

An **ordinal attribute** is an attribute with possible values that have a meaningful order or *ranking* among them, but the magnitude between successive values is not known.

As an example, suppose that the *drink size attribute* corresponds to the size of drinks available at a fast-food restaurant. This nominal attribute has three possible values: *small, medium*, and *large*. The values have a meaningful sequence (which corresponds to increasing drink size); however, we cannot tell from the values *how much* bigger, say, a medium is than a large.

Note that nominal, binary, and ordinal attributes are *qualitative*. That is, they *describe* a feature of an object without giving an actual size or quantity. The values of such qualitative attributes are typically words representing categories. If integers are used, they represent computer codes for the categories, as opposed to measurable quantities (e.g., 0 for *small* drink size, 1 for *medium*, and 2 for *large*). There are numeric attributes, which provide *quantitative* measurements of an object.

Interval-Scaled Attributes

Interval-scaled attributes are measured on a scale of equal-size units. The values of interval-scaled attributes have order. In addition to providing a ranking of values, such attributes allow us to compare and quantify the *difference* between values.

For example, a *temperature* attribute is interval-scaled. Suppose that we have the outdoor *temperature* value for a number of different days, where each day is an object. By ordering the values, we obtain a ranking of the objects with respect to *temperature*. In addition, we can quantify the difference between values. For example, a temperature of 20C is five degrees higher than a temperature of 15C. Calendar dates are another example.

Temperatures in Celsius and Fahrenheit do not have a true zero-point, that is, neither 0C nor 0F indicates “no temperature.” Although we can compute the *difference* between temperature values, we cannot talk of one temperature value as being a *multiple* of another. Without a true zero, we cannot say, for instance, that 10C is twice as warm as 5C. That is, we cannot speak of the values in terms of ratios. This brings us to ratio-scaled attributes, for which a true zero-point exits.

Ratio-Scaled Attributes

A **ratio-scaled attribute** is a numeric attribute with an inherent zero-point. That is, if a measurement is ratio-scaled, we can speak of a value as being a multiple (or ratio) of another value. In addition, the values are ordered, and we can also compute the difference between values, as well as the mean.

Examples of ratio-scaled attributes include *count* attributes such as *years of experience* (e.g., the objects are employees) and *number of words* (e.g., the objects are documents). Additional examples include attributes to measure weight, height, latitude and longitude coordinates.

Discrete versus Continuous Attributes

Classification algorithms often talk of attributes as being either *discrete* or *continuous*. Each type may be processed differently.

A **discrete attribute** has a finite or countably infinite set of values, which may or may not be represented as integers. The attributes *hair color*, *smoker*, and *drink size* each have a finite number of values, and so are discrete. Note that discrete attributes may have numeric values, such as 0 and 1 for binary attributes or, the values 0 to 110 for the attribute *age*. An attribute is *countably infinite* if the set of possible values is infinite but the values can be put in a one-to-one correspondence with natural numbers. For example, the attribute *customer ID* is countably infinite. In theory, the number of customers can grow without limits.

If an attribute is not discrete, it is **continuous**. Continuous attributes are typically represented as floating-point variables.

ML Models

Data remains raw data until its is used to build a model. A model is an “generalized framework” which is suitable for describing and analyzing of a certain domain. Every model provides the following means:

- Knowledge representation
- Reasoning about knowledge

For a long time, the prevailing two main classes of models in AI have been logical models and statistical models. ML makes extensive use of statistical models. For example, linear regression uses a straight line as a model of the relationship between two variables.

The choice of a model depends on the data and the task you want to solve. Once you have chosen a model, you can choose a ML algorithm that works with the model.

Every model requires data and it is used to give structure and meaning to data. The process of fitting a model to a dataset is known as **training**. If the model has some parameters, their estimates are learned during the training.

Generalization

During the generalization phase, the model is applied to unknown data to solve the problem at hand, using the knowledge learned during the training phase. In other words, the training phase is similar to a “lab environment”, in which the model is trained to solve problems with known solutions. The generalization phase corresponds to the “real-word environment” in which the model tries to generalize the knowledge it learned before. Whether the model will succeed depends on how well it is trained (i.e., on the quality of the training data) and on the model itself (i.e., whether the model is based on valid assumptions). For example, if we are using a straight line to model the relationship between two variables, the success of the model depends on the data quality (how well we can draw the straight line for the given training data) and on the model’s assumptions (the model assumes a linear relationship and it will fail if the relationship between the variables is not linear).

Many ML algorithms use **heuristics**. A heuristic is any technique that is practical for solving the problem at hand but is not always guaranteed to find a solution or is not guaranteed to find an optimal solution. Heuristics are used when exact solutions do not exist or are impractical. In other words, we use heuristics for solving difficult problems for which there are no better ways to solve them.

Generalization *(cont.)*

Long before machine learning came into existence, philosophers knew that generalizing from particular cases to general rules is not a well-defined problem with well-defined solutions. Such logical inference by generalization is called ***induction*** and is to be contrasted with ***deduction***, which is the kind of reasoning that goes from the more general to the more specific. There are many versions of this so called problem of induction. One version is due to the eighteenth century Scottish philosopher David Hume, who claimed that the only justification for induction is itself inductive: since it appears to work for certain inductive problems, it is expected to work for all inductive problems. This doesn't just say that induction cannot be deductively justified but that its justification is circular, which is much worse.

Learning Requires Bias

Many ML algorithms use models which are fitted to training data. For example, we might try to learn a function that maps input to output. In classification, for example, the output is a class. In regression the output is a real number. The problem, however, is that there are an uncountable number of different functions that might be chosen. For example, in regression, we can choose a straight line (a polynomial of degree 1), a quadratic function (a polynomial of degree 2), a cubic function (a polynomial of degree 3), etc. to fit the training data. In general, there are many models that can be selected? Why would a learning algorithm happen to select one particular model or one particular class of models? Every model acts as a framework that imposes some limitations and comes with some assumptions. This kind of a priori information embedded in the model is called **bias**, and useful learning without bias is impossible. Not all ML algorithms, however, are biased in the same way. Some algorithms are based on too many or too rigid assumptions. As a result of too much bias, such algorithms produce conclusions, which are systematically erroneous. A ML algorithm is said to have a bias if the conclusions are systematically erroneous, or wrong in a predictable manner.

Performance Evaluation

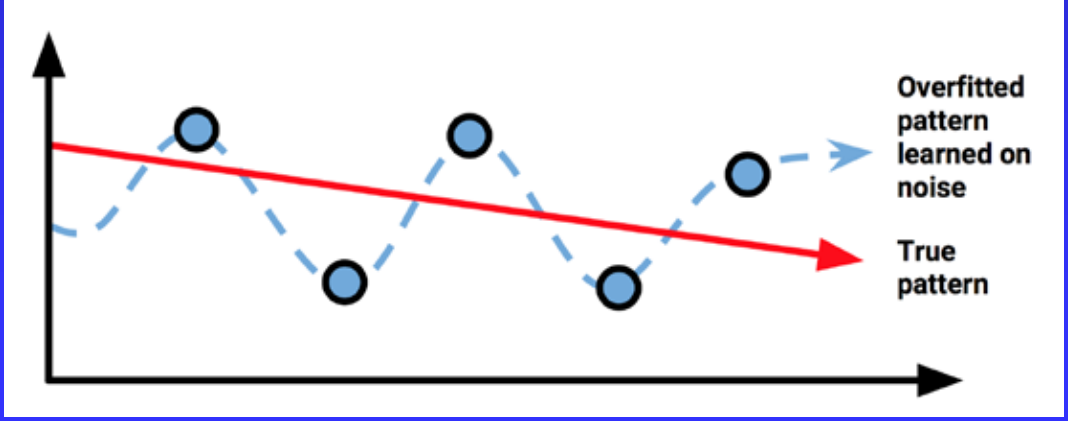
Even though there is no correct answer in inductive learning, it is important to have methods to evaluate the result of learning. We will discuss this matter in more detail later, but, briefly, in supervised learning the induced function is usually evaluated on a separate set of inputs and function values for them called the testing set. A model is said to generalize when it guesses well on the testing set.

A model may fail to perfectly generalize due to the problem of **noise**. Noisy data is caused by:

- Measurement error due to imprecise sensors that sometimes add or subtract a bit from the readings
- Issues with human subjects, such as survey respondents reporting random answers to survey questions, in order to finish more quickly
- Data quality problems, including missing, null, truncated, incorrectly coded, or corrupted values
- Phenomena that are so complex or so little understood that they impact the data in ways that appear to be unsystematic

Overfitting

When we fit highly flexible models, we need to be careful that we do not **overfit** the data, that is, we should avoid trying to model every minor variation in the input, since this is more likely to be noise than true signal. Because most noisy data is unexplainable, attempting to explain the noise will result in erroneous conclusions that do not generalize well to new cases. Efforts to explain the noise will also typically result in more complex models that will miss the true pattern that the learner tries to identify. A model that seems to perform well during training, but does poorly during evaluation, is said to be overfitted to the training dataset, as it does not generalize well to the test dataset. Using an overfitted model usually results in inaccurate predictions of future outputs. The following figure shows an overfitted regression model that uses a complex curve to fit all variations in data, whereas the actual pattern is a straight line:



Legal and ethical issues of ML

There are several legal and ethical issues related to the use of data in ML:

- Legal issues related to the collection, processing and the use of data. For example, according to the U.S. federal anti-discrimination law, a protected class is a characteristic of a person which cannot be used for discrimination, such as race, color, national origin, sex, religion, age, etc.
- Privacy issues. For example, HIPAA privacy laws protect the privacy of individually identifiable health information. FERPA laws protect the privacy of student educational records.
- Robo-ethics: the potential impact of robots and intelligent machines who can become self-sufficient and able to make their own decisions.

No free lunch theorem in ML

The ***no free lunch theorem*** states that no learning algorithm can outperform another when evaluated over all possible learning problems, and thus the performance of any learning algorithm, over the set of all possible learning problems, is no better than random guessing. Consider, for example, the ‘ guess the next number’ questions popular in psychological tests: what comes after 1, 2, 4, 8, ...? If all number sequences are equally likely, then there is no hope that we can improve, on average, on random guessing.

Some typical ML algorithms and models

Model	Learning task	Chapter
Supervised Learning Algorithms		
Nearest Neighbor	Classification	3
Naive Bayes	Classification	4
Decision Trees	Classification	5
Classification Rule Learners	Classification	5
Linear Regression	Numeric prediction	6
Regression Trees	Numeric prediction	6
Model Trees	Numeric prediction	6
Neural Networks	Dual use	7
Support Vector Machines	Dual use	7
Unsupervised Learning Algorithms		
Association Rules	Pattern detection	8
k-means clustering	Clustering	9
Meta-Learning Algorithms		
Bagging	Dual use	11
Boosting	Dual use	11
Random Forests	Dual use	11

R

This course is based on R. R is a language and environment for statistical computing and graphics, similar to the S language originally developed at Bell Labs. It's an open source solution to data analysis that's supported by a large and active worldwide research community. Why should we use R if there are many popular statistical and graphing packages available (such as SAS, SPSS, Stata, etc.):

- R is a comprehensive statistical platform, offering a wide range of data analytic techniques. Just about any type of data analysis can be done in R.
- R has state-of-the-art graphics capabilities. If you want to visualize complex data, R has the most comprehensive and powerful feature set available.
- R is a powerful platform for interactive data analysis and exploration.
- R can easily import data from a wide variety of sources, including text files, database management systems, statistical packages, and specialized data repositories. It can write data out to these systems as well.

R

- R provides a powerful platform for programming new statistical methods in an easy and straightforward manner.
- R contains advanced statistical routines not yet available in other packages. New methods become available for download on a weekly basis.
- If you don’t want to learn a new language, a variety of graphic user interfaces (GUIs) are available, offering the power of R through menus and dialogs.
- R runs on a wide array of platforms, including Windows, Unix, and Mac OS X.
- R is free.