What is Machine Learning

> 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.
>
> T. Mitchell

Successful applications of machine learning

- Speech recognition, Optical character recognition, Computer Vision
- Learning to drive an autonomous vehicle (DARPA Grand Challenges, Google Self-Driving Car)
- Game playing (IBM's Deep Blue, Watson, AlphaGO)
- Recommender Systems

Hot topic

Big players are heavily investing in machine learning:
Google, Facebook, Amazon, IBM, Uber ...

Components of a well-posed learning problem

task to be addressed by the system (e.g. recognizing handwritten characters)

performance measure to evalute the learned system (e.g. number of misclassified characters)

training experience to train the learning system (e.g. labelled handwritten characters)

Designing a machine learning system

1. Formalize the learning task
2. Collect data
3. Extract features
4. Choose class of learning models
5. Train model
6. Evaluate model

Formalize the learning task

- Define the task that should be addressed by the learning system (e.g. recognizing handwritten characters from images)
- A learning problem is often composed of a number of related tasks. E.g.:
  - Segment the image into words and each word into characters.
  - Identify the language of the writing.
  - Classify each character into the language alphabet.
- Choose an appropriate performance measure for evaluating the learned system (e.g. number of misclassified characters)

Collect data

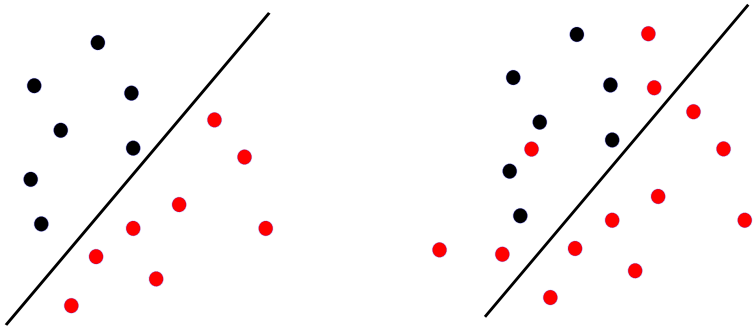
- A set of training examples need to be collected in machine readable format.
- Data collection is often the most cumbersome part of the process, implying manual intervention especially in labelling examples for supervised learning.
- Recent approaches to the problem of data labeling try to make use of the much cheaper availability of unlabeled data (semi-supervised learning)

Extract features

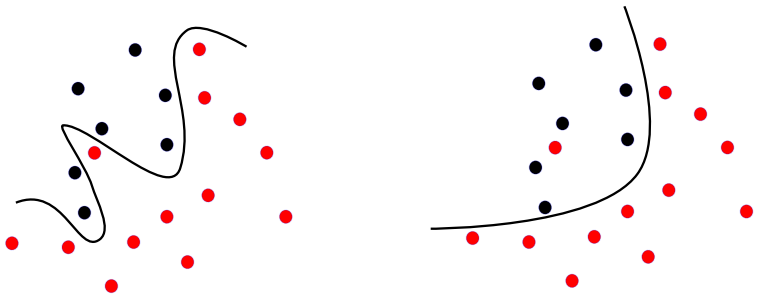
- A relevant set of features need to be extracted from the data in order to provide inputs to the learning system.
- Prior knowledge is usually necessary in order to choose the appropriate features for the task at hand.
- Too few features can miss relevant information preventing the system from learning the task with reasonable performance.
- Including noisy features can make the learning problem harder.
- Too many features can require a number of examples greater than those available for training.

Choose learning model class

- A simple model like a linear classifier is easy to train but insufficient for non linearly separable data.



- A too complex model can memorize noise in training data failing to generalize to new examples.



Train model

- Training a model implies searching through the space of possible models (aka hypotheses) given the chosen model class.
- Such search typically aims at fitting the available training examples well according to the chosen performance measure.
- However, the learned model should perform well on unseen data (generalization), and not simply memorize training examples (overfitting).
- Different techniques can be used to improve generalization, usually by trading off model complexity with training set fitting

> Entia non sunt multiplicanda praeter necessitatem (Entities are not to be multiplied without necessity)
>
> William of Occam (Occam's razor)

Evaluate model

- The learned model is evaluated according to its ability to *generalize* to *unseen* examples.
- Evaluation can provide insights into the model weaknesses and suggest directions for refining/modifying it.
- Evaluation can imply comparing different models/learners in order to decide the best performing one
- Statistical significance of observed differences between performance of different models should be assessed with appropriate statistical tests.

Learning settings

Supervised learning

- The learner is provided with a set of input/output pairs $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$
- The learned model $f : \mathcal{X} \to \mathcal{Y}$ should map input examples into their outputs (e.g. classify character images into the character alphabet)
- A domain expert is typically involved in labeling input examples with the corresponding outputs.

Learning settings

Unsupervised learning

- The learner is provided with a set of input examples $x_i \in \mathcal{X}$, with no labeling information
- The learner models training examples, e.g. by grouping them into clusters according to their similarity

Learning settings

Semi-supervised learning

- The learner is provided with a set of input/output pairs $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$
- A (typically much bigger) additional set of unlabeled examples $x_i \in \mathcal{X}$ is also provided.
- Like in supervised learning, the learned model $f : \mathcal{X} \to \mathcal{Y}$ should map input examples into their outputs
- Unlabelled data can be exploited to improve performance, e.g. by forcing the model to produce similar outputs for similar inputs, or by allowing to learn a better internal representation of examples.

Learning settings

Reinforcement learning

- The learner is provided a set of possible states $\mathcal{S}$, and for each state, a set of possible actions, $\mathcal{A}$ moving it to a next state.
- In performing action a from state s, the learner is provided an immediate reward $r(s, a)$.
- The task is to learn a *policy* allowing to choose for each state s the action a maximizing the overall reward (including future moves).
- The learner has to deal with problems of *delayed reward* coming from future moves, and trade-off between *exploitation* and *exploration*.
- Typical applications include moving policies for robots and sequential scheduling problems in general.

Supervised learning tasks

Classification

binary Assign an example to one of two possible classes (often a positive and a negative one). E.g. digit vs non-digit character.

multiclass Assign an example to one of $n > 2$ possible classes. E.g. assign a digit character among $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$

multilabel Assign an example to a subset $m \leq n$ of the possible classes. E.g. predict the topics of a text.

Supervised learning tasks

Regression
Assign a *real* value to an example. E.g. predict biodegradation rate of a molecular compound under aerobic conditions.

Ordinal regression or ranking
Order a set of examples according to their relative importance/quality wrt the task. E.g. order emails according to their urgency.

Unsupervised learning tasks

Dimensionality reduction
Reduce dimensionality of the data maintaining as much information as possible. E.g. principal component analysis (PCA), random projections.

Clustering
Cluster data into homogeneous groups according to their similarity. E.g. cluster genes according to their expression levels

Novelty detection
Detect novel examples which differ from the distribution of a certain set of data. E.g. recognize anomalous network traffic indicating a possible attack.

Probabilistic Reasoning

- Reasoning in presence of uncertainty
- Evaluating the effect of a certain piece of evidence on other related variables
- Estimate probabilities and relations between variables from a set of observations

Choice of Learning Algorithms

Information available

- Full knowledge of probability distributions of data:
  - Bayesian decision theory
- Form of probabilities known, parameters unknown:
  - Parameter estimation from training data
- Form of probabilities unknown, training examples available:
  - *discriminative* methods: do not model input data (*generative* methods), learn a function predicting the desired output given the input
- Form of probabilities unknown, training examples unavailable (only inputs):
  - *unsupervised* methods: cluster examples by similarity