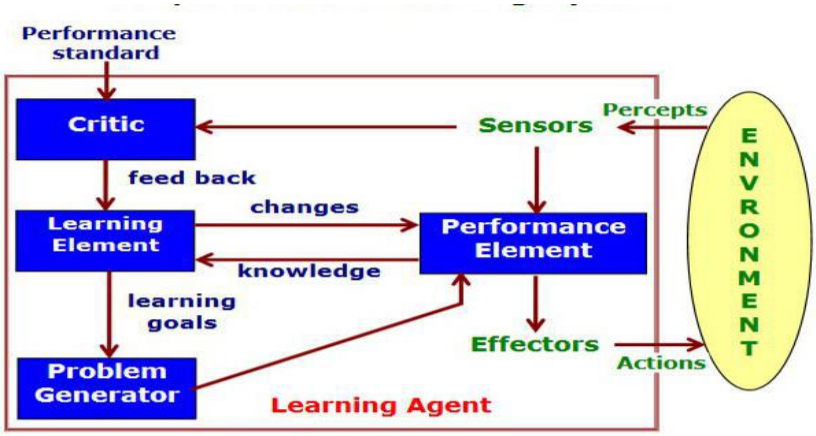
Learning

Learning denotes changes in a system that enables the system to do the same task more efficiently next time.

Definition of Learning System/Agent

A learning agent is an entity that is capable of perceiving and do action. An agent can be viewed as perceiving its environment through sensors and acting upon that environment through actuators. A learning agent has a learning component which improves its action to the perceived inputs as time proceeds.



Performance Element: Agent that takes percepts and decides the actions.

Learning element: Responsible for making improvements in the performance.

Critic: Tells the learning element how it is doing by comparing with standard parameters.

Problem Generator: suggests actions that will generate new examples that will aid in learning.

Aspects of developing a learning system

- Training Data
- Concept Representation
- Function Approximation

To make a system learn, a training data is to be provided. For example if we want to make a system learn to differentiate between animals and birds, then we provide it with some examples of animals and some example of birds (labeled dataset i.e. each example also contains information of whether it is a bird or animal).

From these examples, the system learns some concept, for example, birds have wings, a beak and two legs whereas animals have four legs and have a nose, etc.

After concept representation the system should be able to identify new objects. However, the learning system needs to use function approximation as some of the new example may exactly not match the concepts i.e. an elephant has a trunk (which is neither a nose nor a beak) so the system should be able to approximate whether it is an animal or a bird.

Machine Learning

Definition

Tom M. Mitchell provided a widely quoted, more formal definition: "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*."

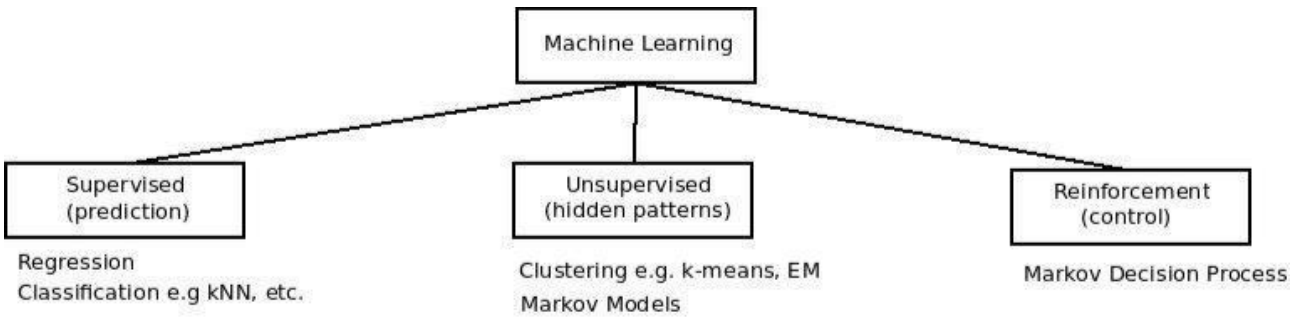
Goals and Applications of ML

The goal of machine learning is to design and develop algorithms that allow computers to learn from historical/training data.

- Stock Market Prediction
- Fraud Detection
- Speech and Handwriting Recognition
- Natural Language Processing
- Computer Vision
- Robotics
- Game Playing
- Recommender Systems
- Sentiment Analysis
- Bioinformatics
- Medical Diagnosis

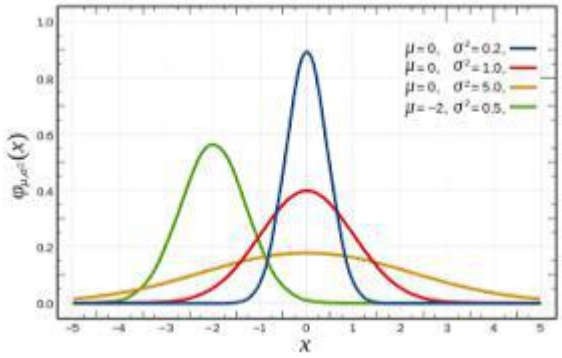
Types of Machine Learning

- **Supervised learning**: The computer is presented with example inputs and their desired outputs, and the goal is to learn a general rule that maps inputs to outputs. Supervised learning is used in predictive analytics i.e. classification and regression. Classification deals with discrete output, and regression deals with continuous output.
- **Unsupervised learning**: No labels are given to the learning algorithm, leaving it on its own to find patterns in its input. Unsupervised learning deals with pattern mining and clustering.
- **Reinforcement learning**: A computer program interacts with a dynamic environment in which it must perform a certain goal (such as driving a vehicle or playing a game against an opponent). The program is provided feedback in terms of rewards and punishments as it navigates its problem space. Reinforcement learning differs from standard supervised learning in that correct input/output pairs are never presented, nor sub-optimal actions explicitly corrected. It finds its main application in control theory, and robotics.



Distributions

Normal/gaussian distribution is the most common distribution in statistics. Normal distribution is characterized by a bell-shaped curve. Normal distribution curve reaches maxima at the mean value of the random variable and is symmetric around the imaginary axis passing through the mean. Lower the variance in data, steeper is the curve of normal distribution.



The probability distribution function of normal distribution is:

$$P(x \mid \mu, \sigma^2) = \frac{1}{\sqrt{2\sigma^2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

The probability of the random variable X in normal distribution depends upon the mean μ and variance σ^2 of X.

Exponential Family

Exponential family is a set of probability distribution of the form described below.

$$f(x|\eta) = h(x) * \exp[T(x)-A(\eta)]$$

h(x) is any function of x.

T(x) is a sufficient statistics of the distribution. It's a function such that for any two datasets x and y, the density function is same if T(x)=T(y).

η is called the natural parameter.

A(η) is called log-partition function.

The Bernoulli distribution may not look like a member of exponential family but it belongs to the exponential family. Bernoulli distribution is given by

$f(k;p)=p^k(1-p)^{(1-k)}$ $k\epsilon\{0,1\}$

We can rewrite the RHS of the equation as

$\exp(\log[p^k(1-p)^{(1-k)}]) = \exp[k*\log(p)+(1-k)\log(1-p)] = \exp[k*\log(p)+\log(1-p)-k*\log(1-p)]$

$= \exp[k*\log(p/(1-p))+\log(1-p)]$

Comparing with the equation of exponential family

h(x)=1, T(x)=x, η=log[p/(1-p)], A(η)=-log(1-p)

Ensemble Learning

Ensemble learning is the process by which multiple models are strategically generated and combined to solve a particular computational intelligence problem.

Bagging

Bagging (bootstrap aggregating) is a technique where N new datasets are created from the original dataset. These datasets are the same size as the original dataset. Each dataset is built by randomly selecting an example from the original “with replacement” i.e. an example can be selected more than once. Then on each of these datasets we apply the learning model. We obtain the results provided by each of the learning model and then merge the result using techniques like voting, etc.

Boosting

Whereas in bagging individual models are built separately, in boosting each new model is influenced by the performance of those built previously.

A popular boosting algorithm is AdaBoost which begins by assigning equal weights to all the instances in the training data. It then calls the learning algorithm to form a classifier for this data and re-weights each instance according to the classifiers output. The weight of correctly classified instance is decreased and that of misclassified is increased. In next iteration, the classifier focuses on correct classification of high weight instances. The amount of weight increase or decrease in AdaBoost is –log(e/1-e).

Model Selection and Feature Selection

Model selection

- Given a set of models, choose the model that is expected to give the best results.
- Chosing among different learning algorithms e.g. choosing kNN over other classifiction algorithms.
- Choosing parameters in same learning model e.g. choosing value of k in kNN.

Feature Selection

Selecting a useful subset from all the features.

Why Feature Selection?

- Some algorithms scale (computationally) poorly with increased dimension
- Irrelevant features can confuse some algorithms
- Redundant features adversely affect regularization
- Removal of features can increase (relative) margin (and generalization)
- Reduces data set and resulting model size

Note: Feature Selection is different from Feature Extraction. The latter transforms original features to get a small set of new features

How?

- Remove a binary feature if nearly all of it values are same.
- Use some criteria to rank features and keep top ranked features.
- Wrapper Methods: requires repeated runs of the learning algorithm with different set of features.

Evaluating Learning Algorithms and Classification Errors

Evaluation of classification algorithms

The performance of a classifier can be measured by keeping some part of training dataset for testing; in k-fold validation data is divided into k-parts, k-1 parts for training, 1 part for testing.

- *Confusion Matrix* -> 2X2 matrix reporting number of True Positives, True Negatives, False Positives, False Negatives.
- *Area Under (ROC) Curve* -> Receiver Operating Characteristic graph has False Positive Rate [FPR=FP/(FP+TN)] and True Positive Rate [TPR=TP/(TP+FN)] as X and Y axes (range 0-1). High value is better value (100% is maximum).

Evaluation of classification algorithms

The performance of a clustering algorithm is evaluated using WCSS (within cluster sum of squared errors). Lower is the value of WCSS, better is the result of clustering.