Part 1: Introduction to Machine Learning in the Nuclear Medicine Context

Carlos F. Uribe[1*], Sulantha Mathotaarachchi[2*], Vincent Gaudet[3], Kenneth C. Smith[4], Pedro Rosa-Neto[2], François Bénard[1,5], Sandra E. Black[6], Katherine Zukotynski[7,8]

[1]Department of Molecular Oncology, BC Cancer, Vancouver, BC, Canada
[2]Translational Neuroimaging Lab, McGill University, Montreal, QC, Canada
[3]Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada
[4]Department of Electrical and Computer Engineering, University of Toronto, Toronto, ON, Canada
[5]Department of Radiology, University of British Columbia, Vancouver, BC, Canada
[6]Department of Medicine (Neurology), Sunnybrook Health Sciences Centre, University of Toronto, Toronto, ON, Canada
[7]Institute of Biomaterials and Biomedical Engineering, University of Toronto, Toronto, ON, Canada
[8]Departments of Medicine and Radiology, McMaster University, Hamilton, ON, Canada

*Contributed equally

Corresponding author:
Katherine Zukotynski
Departments of Medicine and Radiology
McMaster University, Hamilton, ON
Telephone: 905-521-2100 x76556
Fax: 905-546-1125
zukotynk@mcmaster.ca

First author:
Carlos Uribe
BC Cancer Agency, Vancouver, BC
curibe@bccrc.ca

Word Count: 5981

Disclosure: There is no conflict of interest that will directly or indirectly influence the content of the manuscript submitted.

Running Title: Introduction to machine learning

Abstract
This article, the first in a two-part series, provides an introduction to machine learning in a nuclear medicine context. Part I addresses the history of machine learning and describes common algorithms with illustrations of when these algorithms can be helpful in nuclear medicine. Part II focuses on current contributions of machine learning to our field and addresses future expectations, limitations and provides a critical appraisal of what machine learning can and cannot do.

Keywords: Machine learning, Artificial intelligence, Nuclear Medicine, Algorithms

Learning Objectives

1. Provide an introduction to machine learning (ML), neural networks and deep learning.
2. Discuss common ML algorithms with illustrative examples/figures.
3. Compare ML algorithms and provide guidance on selection for a given application.

Knowledge Acquisition

This paper should familiarize the reader with ML algorithms and differences between them.

INTRODUCTION

It seems that currently every conference hosts sessions on artificial intelligence (AI) and machine learning (ML) and that these terms are increasingly pervading our literature. However, AI and ML are not new. First described in 1956, *Artificial Intelligence* (AI) refers to computers that perform tasks normally requiring human intelligence (e.g. visual perception, decision making). *Machine learning* (ML), a branch of AI, dates to the electronic computing of the 1950s (*1*) and its application to medicine in the 1960s (*2*).

Broadly speaking, ML refers to algorithms that are programmed to learn from observations and can then make statistical inferences based on what has been learned, rather than being based on the explicit knowledge of the programmer. Traditionally, a programmer specifies all aspects of an algorithm and nothing further is learned beyond what has been written, like writing a detailed recipe for baking a cake. ML algorithms include parameters with unknown values at the time of programming that are learned by presenting the program with observations, like a novice chef using an incomplete recipe to bake a cake. The chef must decide what ingredients to use and in what proportion, based on observations.

Why the rise in ML algorithms for medical imaging now? Reasons include: advances in theory (better algorithms) (*3*), microelectronics (better hardware) (*4*), and availability of massive amounts of training data (*big data*). Also, fields such as *radiomics* and *radiogenomics* have shows great promise (*5*). Radiomics refers to the extraction and quantification of mathematical descriptors (“features”) of image texture, shape, etc., which can be correlated with a variety of data, such as histophathologic findings and clinical outcomes (survival). Radiogenomics more specifically correlates radiomic features with genomic data. These new fields use ML to investigate scientific phenomena by identifying often previously unknown relationships.

The aim of this article, the first in a two-part series on ML in nuclear medicine, is to introduce ML and discuss common algorithms highlighting differences between them. The second article will focus on ML in nuclear medicine, addresses future expectations, limitations and provides a critical appraisal of what ML can and cannot do.

A BIRD'S EYE VIEW OF MACHINE LEARNING ALGORITHMS

ML algorithms include core elements: a) a mathematical model, b) a cost function, and c) data. The art is identifying the mathematical model and cost function suitable to the task and finding access to sufficient data.

Mathematical Model: A mathematical model is a way to explain relationships between data, and to translate input observations into outputs. For example, a mathematical model could be a function that translates input *features* from an image (e.g. the standardized uptake value ratio (SUVR) for an amyloid brain PET in a region of interest (ROI)) into an output decision (e.g. scan classification as positive or negative for amyloid deposition). The decision of which model to use is often based on tradeoffs between accuracy, suitability for the task, and practical implementation considerations. Problems where the model's output observation is continuous-valued (e.g. amount of amyloid deposited) are known as *regression* problems; those with discrete-valued output observations or categories (e.g. amyloid positive or negative) are *classification* problems. Mathematical models vary and have different complexity and/or predictive capability. Many models have parameters called *hyperparameters* that are chosen prior to training but may be adjusted if performance is insufficient. For example, it might be initially stipulated that any given amyloid brain PET is classified as positive or negative for amyloid deposition (e.g. 2 outcomes allowed); however training could suggest some scans are equivocal (e.g. 3 outcomes allowed).

Cost Function: A cost function (or loss function) measures how close the model performance is to its intended result. For example, the cost function could be a misclassification rate or mean square error.

Data: Just like physicians learn by reviewing cases, a ML algorithm is trained using data. Data may be image measurements (e.g. SUVRs for an amyloid PET), survival data, or a combination of these and other observations. Data are usually divided into training, validation and testing subsets. Training data is used to teach the model, validation data is used to fine-tune hyperparameters, and testing data is used to confirm performance of the trained model. Often the data includes known outputs. For example, a physician might have classified all available amyloid PET scans as amyloid positive or negative. Sometimes, outputs are unknown. For example, one might have all of the SUVRs for the ROIs but not know if the scan is amyloid positive or negative. Whether the data includes known outputs or not distinguishes two broad ML algorithm categories: supervised and unsupervised.

SUPERVISED AND UNSUPERVISED LEARNING

Supervised learning refers to ML algorithms that are trained using known outputs. For example, a ML algorithm could determine which SUVR/ROI amyloid PET combination best predicts neurological findings using knowledge of prior SUVR/ROI combinations and neurological findings. In a sense, supervised algorithms determine unknown values of a mathematical model that allow the model to approximate the training data in view of predicting outcomes for future test data. Examples of supervised ML algorithms, described below are: linear regression, support vector machines (SVMs), random forests, and artificial neural networks (ANNs).

Unsupervised learning refers to ML algorithms that use training data where outputs are unknown. These algorithms often solve tasks such as identifying commonalities between data points (e.g. clustering amyloid PET into amyloid positive or negative with no *a priori* knowledge of which category the PET scan belongs to) or finding ways to compress data (e.g. reducing the number of SUVR/ROIs needed). Examples of unsupervised ML algorithms, described below are: k-means clustering, principal component analysis and singular value decomposition.

There are also ML algorithms that are not strictly supervised or unsupervised (e.g. reinforcement learning, competitive learning, and semi-supervised learning). These are beyond the scope of this paper.

Figure 1 illustrates these concepts theoretically. In Figures 1A and 1B, the data points are classified into circles and triangles. In Figure 1C, the data points are not classified. If the purpose is to train a ML algorithm to appropriately classify data into circles and triangles, and the classification scheme is known, a supervised approach is preferred. For a scenario where the classification scheme is unknown, assigning data to categories requires an unsupervised approach. An observer might say there are two clusters of data points (bottom left and top right). However, another observer might group data points into more clusters. Ultimately, an initial educated guess on the number of categories might be necessary. Figure 1D-F shows potential models applied to each scenario. In Figure 1D, a straight line creates a delineation between circles and triangles. The mathematical model in this case is linear, namely $y = ax + b$ and the ML algorithm must learn the values of a and b that minimizes the number of false classifications (according to the cost function). In Figure 1D, line '2' has a lower cost than line '1,' and an effective ML algorithm could adapt a and b towards the better solution. In Figure 1E, a linear model would inevitably lead to a high number of false classifications. Here, we might suggest a more complex model. In Figure 1F, the algorithm categorized data in 2 clusters. Several common ML algorithms, are discussed below and summarized in Table 1.

COMMON MACHINE LEARNING ALGORITHMS

There are many ML algorithms, typically programmed in languages such as R, Matlab and Python (*6*) that include built-in support (e.g. as libraries, toolkits or packages). Many aspects differentiate these algorithms, and the best algorithm to use often depends on the: a) task, b) data, and c) mathematical model complexity (e.g. number of operations such as additions and multiplications to be performed).

Naïve Bayes Classification: A supervised algorithm for classification where features follow a simple probability distribution (e.g. Gaussian or multinomial), and assumptions of independence are made (allowing mathematical simplifications). The hyperparameters are the distribution parameters (e.g. mean and variance of feature values for each class). Classification is done by looking at features and computing the class that maximizes a probability function. The computational cost is low. For example, a retrospective study by Mehta et al. used a multinomial naïve Bayes classifier to predict response to ^{90}Y radioembolization therapy from ^{18}F-FDG PET/CT features (7). Using 22 training cases and 8 testing cases, the sensitivity for predicting response was approximately 80%.

Linear regression: A supervised algorithm for regression problems where datasets contain n-dimensional continuous features, and the model is the n-dimensional hyperplane (hence the linearity) that best fits the dataset. The cost function is defined as the mean square error between the true outcomes and the model's predicted outcome. The computational cost is low.

Support Vector Machines (Figure 2): A supervised algorithm for binary classification, or less commonly regression. The algorithm identifies a curve (or "hypersurface," if there are many dimensions) that best separates two classes in such a way that the curve is at maximum distance to the closest points in each class. The closest data points to the line are called the *support vectors*, hence the algorithm name. Once trained, SVMs tend to be computationally efficient, at least for smaller datasets (e.g. once trained, the algorithm need only determine which side of the curve a new data point is on). SVMs can also handle data where no clear linear separation (hyperplane) can be drawn between classes (Figure 1B), by moving to a higher-dimensional space, using a nonlinear data transformation, or minimizing the number of misclassifications. For example, Van Weehaeghe et al. used an SVM to classify subjects as Amyotrophic Lateral Sclerosis (ALS) versus healthy controls based on differences in brain metabolism on ^{18}F-FDG PET (8). All PET data was spatially normalized and analyzed quantitatively using: 1. volume-of-interest (VOI) and 2. voxel-based analysis with statistical parametric mapping protocols. A cohort of 70 ALS and 20 healthy subjects trained an SVM (linear model) using a voxel-based comparison and a brain mask defined by those voxels exceeding 50% of the mean of the ^{18}F-FDG PET images. Subsequently 105 ALS subjects served as test data showing the SVM was 100% accurate for classification. Since ALS subjects had similar patterns of brain metabolism in both the training and testing data, the SVM was able to correctly classify subjects with ALS from healthy controls. There would have been little need to explore a more computationally intensive ML algorithm to solve this task. Of interest, 10 subjects with primary lateral sclerosis (PLS) were also recruited. Since brain metabolism in ALS and PLS was similar, these subjects could not be discriminated and further evaluation would be needed to classify these cases.

Random Forest (Figure 3): A supervised algorithm that creates a collection of decision trees (e.g. forest) for classification or regression. For example, one might want to predict neurological findings by learning from a series of SUVR/ROIs on amyloid PET scans in patients with known neurological findings. To build each tree in the forest, the algorithm picks a random subset of cases, and a random subset of features (the sizes of these subsets are hyperparameters). In our example, the subset of cases is randomly chosen from the amyloid PET scans with clinical reads, and the subset of features is randomly chosen from the available ROIs. Each tree is recursively constructed starting at the root by identifying the feature (e.g. ROI) and feature value (e.g. SUVR) that best (according to misclassification rate) splits the subset's cases into a left pile and a right pile; then, two branches are created in similar fashion: a left branch for the left pile and a right branch for the right pile. The end points of the trees represent the outputs, and are reached when a pile only has cases from one class (e.g. all negative or all positive); the end points can be classes (for classification problems) but may also be numerical (for regression problems). Each tree is different since it deals with different subsets of data. Once the forest has been created, new data is presented to each tree for categorization; the final result is the most common decision among all trees' decisions. In our example, the final result could be the clinical finding (class) that is most common: positive or negative for amyloid deposition. As a recent study showed, a random forest with 2000 trees could be trained to predict subject mortality post ^{90}Y radioembolization in a group of 366 subjects with intrahepatic tumors based on factors such as age at radioembolization, liver function and tumor burden (9). Here, the authors used a parameter 'minimal depth' (e.g. shortest distance in a tree from trunk to branch of first split in the variable) to identify predictive measurements: baseline cholinesterase and bilirubin. In particular, post procedural mortality was found to increase significantly once bilirubin levels exceeded 1.5mg/dl.

Artificial Neural Networks (Figure 4): ANNs are typically (but not always) supervised and can be used for classification or regression problems. ANNs process data in steps (called *layers*). At each layer, several simple computation units called *neurons* (inspired by biology) process input data and convey results to neurons in the next layer (Figure 4A). A neuron's computation typically involves a weighted summation of inputs and a bias, followed by a nonlinear transformation of the result (e.g. thresholding). An algorithm such as backpropagation (*10*) is used for training (e.g. learning values for weights). Among ML algorithms, ANNs tend to be the most powerful, but also the most computationally challenging. The computational cost comes from the fact that normally, every neuron in a layer is connected to every neuron in the subsequent layer. To illustrate how this can become problematic, consider an ANN that processes a 100x100-pixel image, e.g. with 10,000 inputs. If each layer has as many neurons as there are inputs, each layer requires 10,000x10,000 = 10^8 connections, each of which requires at least one multiplication and addition, hence the high computational cost *per layer*. In the earlier days of ML, ANNs were limited by this high computational cost (which could not be met by the microelectronic technologies of the time), and the fact that some problems remained unsolvable (although it could be argued they would have been solvable had the proper technology been available). Recent developments have allowed ANNs to overcome these barriers, and ANNs are now very popular.

Deep learning, first described in 1986 (*11*), was applied to ANNs in 2006 (*12*) to form deep neural networks (DNNs). DNNs are essentially ANNs with many layers (10-150, depending on the task). By having a large number of layers, DNNs are more computationally taxing, but are able to tackle complex problems. Also, each DNN layer is thought to treat data at a different level of abstraction. For instance, a DNN might devote a first layer to image segmentation and a subsequent layer to lesion identification within a segmented ROI.

The high computational cost per layer can be reduced by using structures whereby each neuron connects to a subset of neurons in the subsequent layer. A method for image processing applications is convolutional neural network (CNN). CNNs are a form of DNNs where the main difference is the use of convolutional layers. Specifically, in a CNN, the outputs of neurons at one layer are considered to be an image; 2-dimensional convolution is applied to the image, followed by a nonlinear operation (called ReLU, for *rectified linear unit*), and pooling of pixel data (combining several pixels into one, which also reduces cost). Say a convolutional mask of size 10x10 is used in the example above, only 10,000x100 =

10^6 multiplications and additions are required – a 100x savings. Figure 4B depicts a 2-layer CNN where each performs a convolution, ReLU, and pooling. After these layers, another layer of neurons (where each neuron is connected to every neuron in the previous layer) computes the output classification. Also, CNNs can be used to directly process images; but can also be used with pre-defined ROIs/VOIs. Thus, there is no need to decide which radiomic features to extract, as the CNN can make its own choice of features. In 2012, a breakthrough occurred with the introduction of a CNN called AlexNet5 (*13*) that outperformed prior ML algorithms in an open challenge called ImageNet.

Together, DNNs and CNNs (indeed, there is little distinction between the two in the literature, and most ANNs today are both deep and convolutional, to the point where all three acronyms are nearly used interchangeably) paved the way for the current boom in ML. In general, deep learning requires less human input for training purposes than more traditional ML algorithms. Unfortunately, deep learning is also more complex and often requires a large amount of training data to be effective. For example, in the ImageNet challenge (*13*), more than one million annotated images were used. In medical imaging, it is often difficult to acquire such large numbers of training samples. To circumnavigate this limitation, researchers have tried to reduce requirements for training data by using data-augmentation techniques (*14-16*), reducing image dimensionality (*16-19*) and fine-tuning existing pre-trained DNNs (*19-21*). For example, in some cases the available data can be slightly varied, such as rotating images, flipping images or adding noise, to *augment the data*. Also, if layers of a network can be trained on a set of images with similar features to the images being studied, this learning might be transferred to the new task, known as *transfer learning*, decreasing the data needed to learn the new task.

CNNs have been used in several medical imaging settings. A nice example, includes the use of CNNs in conjunction with image preprocessing and random forests in an automated system to detect malignancy on mammograms (*22*). A sequence of image processing steps optimized by using genetic algorithms was used (Figure 4C). Mammograms were pre-processed using a technique called “contrast limited adaptive histogram equalization.” This was followed by two deep CNNs (each with at least 20 layers) that processed images at two different scales, and then a random forest made a final prediction. A dataset with over 7,000 images divided into training, validation, and test sets reportedly resulted in a ML algorithm with comparable sensitivity and specificity to that of a radiologist.

Principal Component Analysis (Figure 5): Principal component analysis (PCA) is an unsupervised algorithm for data simplification. The idea is to find linear combinations of building blocks (e.g. features) that can be used to reconstruct the initial data, where the combination of building blocks is simpler than the initial data and any initial data loss is minimized. PCA is often used for data compression. An example of PCA in nuclear medicine, is to reduce respiration artifact in PET scans (*23*). In this case, the task requires identifying the data that is most suggestive of breathing and using this information during the reconstruction process to reduce breathing artifact. The authors showed very high correlation between PCA and results obtained using costly external gating equipment. Linear discriminant analysis (LDA) is closely related to PCA. The fundamental difference is that LDA is a supervised algorithm that requires linear combinations of building blocks.

K-means Clustering (Figure 6): An unsupervised algorithm for *clustering* data, where a "cluster" refers to a collection of data points that are close to each other, e.g., in terms of their measurements/feature values. K refers to the number of clusters and is specified by the programmer. The goal is to create K data clusters where every data point is assigned to the cluster with closest resemblance. The computer does this through iteration: 1. K cluster centers are chosen (i.e., random values chosen for each feature), 2. Each data point is assigned to the cluster whose center is closest, 3. The cluster centers are adjusted to be at the center of those data points that were assigned to it in step 2, 4. The process is repeated starting at step 2, and ends when no data points change clusters. Given the random nature of initially choosing the center of the K clusters, it is possible to produce different solutions each time the program is run. To illustrate, amyloid brain PET scans could be clustered into K=2 groups (e.g. amyloid positive versus negative) using known SUVR/ROI combinations for each scan. This, in turn, may suggest which SUVR/ROI combinations are most useful for clustering the scans. As another example, Blanc-Durand et al. used K-means clustering in 37 patients with ^{18}F-fluoro-ethyl-tyrosine (FET) PET for newly diagnosed gliomas to suggest imaging features associated with progression and survival (*24*).

Hierarchical Clustering (Figure 7): An unsupervised algorithm for clustering. Hierarchical clustering looks for data similarities, by grouping nearest neighboring data-points into pairs and clusters, until no free data points remain. The resulting structure is referred to as a *dendrogram*. In *(25)*, Tsujikawa et al. used a dendrogram to suggest related features on ^{18}F-FDG PET in subjects with cervical cancer.

ISSUES TO CONSIDER

One of the questions that can be challenging is which ML algorithm to choose, or why a certain algorithm has been used in a publication. To start off with, each ML algorithm targets a specific task (classification, regression, dimensionality reduction, clustering, …). For instance, someone who wants to use an existing database of images and benign/malignant classifications to train an ML algorithm to classify future images might look to a supervised algorithm for classification, such as: naïve Bayes classification, random forests, SVMs, or CNNs. As it would be unlikely that the user would have a sense of the underlying probability distribution of the input data, a naïve classifier might be ruled out quickly. Next, the ML algorithm complexity, computational power and predictive capability must be considered. For example, is it desirable to perform the work on a desktop workstation or will more computing power be needed? Indeed, to move forward, the user may try out each of the options to see which works best (using an available pre-programmed toolkit), or possibly reach out to someone with more of a theoretical grounding in ML (e.g. statistician, mathematician, computer scientist or engineer) about next steps (these people will often welcome your data with open arms!).

Another important consideration to take into account prior to selecting a complex ML algorithm, in addition to the potentially high computational cost, is *overfitting*, which is a situation where the mathematical model too closely reflects minor variations in training data that are not predictive of validation and test data (e.g. where there is a lack of generalizability, for instance when the number of samples is too small). Consider for instance the data in Figure 5A; instead of unsupervised learning, had we seen this as a regression problem with *y* as the dependent variable, and had we tried to represent the data using an nth-order polynomial for an exact match, we would not necessarily have discovered the underlying (simpler and more representative) linear trend. A good model should be predictive, and might not necessarily exactly reproduce the training data. Although a detailed analysis of the signs of overfitting and strategies to reduce it is beyond the scope of this paper, it is useful to know that some ML algorithms, e.g. random forests, are more tolerant to overfitting.

This leads us naturally to the last issue in this article, namely the amount of data required to adequately train an ML algorithm, and the potentially high cost of acquiring sufficient data. As seen above, there are indeed ML algorithms that require very large datasets (especially when dividing them into training, validation, and testing subsets). However, many techniques have emerged to deal with this problem, including data augmentation. One pitfall is potential biases in data sets. In the early days of ML, there were ANNs unintentionally fooled by such innocuous issues as lighting conditions! The bottom line is that even the best augmentation techniques and the most tolerant ML algorithms will fail when subgroups are too small (which is, unfortunately, common in current radiology and nuclear medicine).

FUTURE PROSPECTS

Although ML has long existed, medical imaging applications are in their infancy (*26*). The advent of hybrid imaging (e.g. PET/MR), has brought enthusiasm to the field since this is ideally suited for the application of ML (due to different types of information that can be obtained simultaneously). However, several limitations remain. Typically, ML algorithms require large medical databases of sufficient quality to give reliable results. In March 2018, Thrall et al. suggested AI medical imaging applications could benefit from: "(1) national and international image sharing networks, (2) reference datasets of proven cases against which AI programs can be tested and compared, (3) criteria for standardization and optimization of imaging protocols for use in AI applications, and (4) a common lexicon for describing and reporting AI applications". (*27*) Standardization of imaging data acquisition is important since, the scanner model and protocol used to acquire imaging data can influence radiomic features. Further, although initial results in ML are exciting, standardization and methodologic transparency is needed to deliver reproducible results and speed clinical translation (28). Also, several topics require clarification as ML becomes ubiquitous in nuclear medicine, including the role of personal responsibility for patient care, fiduciary compact, data confidentiality, the need for bias free algorithms, the process for validating ML algorithms and the generalizability of results beyond a specific patient population where data exists, to name a few. Further discussion on this is beyond the scope (and word limit) of this paper.

The second article in this series will address several key issues, with a focus on ML in nuclear medicine, and how this might impact our work flows.

References

(1) Rosenblatt F. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychol Rev.* 1958;65:386–408.

(2) Silink K. The possibility of designing machines which learn diagnostic. The zero-systems of types and pathotypes in endocrinology. *Act Nerv Super (Praha)*. 1961;3:148-153.

(3) Goodfellow I, Bengio Y, Courville A. *Deep learning*. The MIT Press; 2016:1-800.

(4) Patterson D. 50 years of computer architecture: from mainframe CPUs to neural-network TPUs. *Proc IEEE Int Solid-State Circuits Conf.* 2018:27-31.

(5) Kelchtermans P, Bittremieux W, De Grave K, et al. Machine learning applications in proteomics research: how the past can boost the future. *Proteomics*. 2014;14:353-366.

(6) VanderPlas J. *Python data science handbook: essential tools for working with data*. 1st ed. O'Reilly Media, Inc.; 2016:1-548.

(7) Mehta R, Cai K, Kumar N, et al. A lesion-based response prediction model using pretherapy PET/CT image features for Y90 radioembolization to hepatic malignancies. *Technol Cancer Res Treat*. 2017;16:620-629.

(8) Van Weehaeghe D, Ceccarini J, Delva A, Robberecht W, Van Damme P, and Van Laere K. Prospective validation of ^{18}F-FDG brain PET discriminant analysis methods in the diagnosis of amyotrophic lateral sclerosis. *J Nucl Med*. 2016;57:1238–1243.

(9) Ingrisch M, Schöppe F, Paprottka K, et al. Prediction of ^{90}Y radioembolization outcome from pretherapeutic factors with random survival forests. *J Nucl Med*. 2018;59:769-773.

(10) Rumelhart DE, Hinton GE, Williams RJ. Learning representations by back-propagating errors. *Nature*. 1986;32:533–536.

(11) Dechter R. Learning while searching in constraint-satisfaction problems. *Proc 5th Nat Conf on Artificial Intelligence*. 1986;1:178-183.

(12) Hinton GE, Osindero S, Teh YW. A fast learning algorithm for deep belief nets. *Neural Comput.* 2006;18:1527–1554.

(13) Krizhevsky A, Sutskever I, Hinton GE. ImageNet classification with deep convolutional neural networks. *Proc 25th Int Conf on Neural Information Processing Systems*. 2012;1:1097-1105.

(14) Setio AAA, Ciompi F, Litjens G, et al. Pulmonary nodule detection in CT images: false positive reduction using multi-view convolutional networks. *IEEE Trans Med. Imaging.* 2016;35:1160–1169.

(15) Cheng JZ, Ni D, Chou YH, et al. Computer-aided diagnosis with deep learning architecture: applications to breast lesions in US images and pulmonary nodules in CT scans. *Sci Rep.* 2016;6:24454.

(16) Shen W, Zhou M, Yang F, Yang C, Tian J. Multi-scale convolutional neural networks for lung nodule classification. *Inf Process Med Imaging.* 2015;24:588–599.

(17) Suk HI, Lee SW, Shen D. Alzheimer's Disease Neuroimaging Initiative. Hierarchical feature representation and multimodal fusion with deep learning for AD/MCI diagnosis. *Neuroimage.* 2014;101:569–582.

(18) Li R, Zhang W, Suk HI, et al. Deep learning based imaging data completion for improved brain disease diagnosis. *Med Image Comput Comput Assist Interv.* 2014;17:305–312.

(19) Ciompi F, de Hoop B, van Riel SJ, et al. Automatic classification of pulmonary peri-fissural nodules in computed tomography using an ensemble of 2D views and a convolutional neural network out-of-the-box. *Med Image Anal.* 2015;26:195–202.

(20) Shin HC, Roth HR, Gao M, et al. Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning. *IEEE Trans Med Imaging*. 2016;35:1285-1298.

(21) Brosch T, Tam R, Initiative for the Alzheimers Disease Neuroimaging. Manifold learning of brain MRIs by deep learning. *Me Image Comput Comput Assist Interv.* 2013;16:633–640.

(22) Teare P, Fishman M, Benzaquen O, Toledano E, Elnekave E. Malignancy detection on mammography using dual deep convolutional neural networks and genetically discovered false color input enhancement. *J Digit Imaging*. 2017;30:499-505.

(23) Walker MD, Bradley KM, McGowan DR. Evaluation of principal component analysis-based data-driven respiratory gating for positron emission tomography. *Br J Radiol*. 2018;91:20170793.

(24) Blanc-Durand P, Van Der Gucht A, Verger A, et al. Voxel-based 18F-FET PET segmentation and automatic clustering of tumor voxels: A significant association with IDH1 mutation status and survival in patients with gliomas. *PLoS One*. 2018;13:e0199379.

(25) Tsujikawa T, Rahman T, Yamamoto M, et al. ^{18}F-FDG PET radiomics approaches: comparing and clustering features in cervical cancer. *Ann Nucl Med*. 2017;31:678-685.

(26) Kohli M, Prevedello LM, Filice RW, Geis JR. Implementing machine learning in radiology practice and research. *AJR Am J Rontgenol*. 2017;208:754-760.

(27) Thrall JH, Li X, Li Q, et al. Artificial intelligence and machine learning in radiology: opportunities, challenges, pitfalls, and criteria for success. *J Am Coll Radiol*. 2018;15:504-508.

(28) Vallières M, Zwanenburg, Badic B, Cheze Le Rest C, Visvikis D, Hatt M. Responsible radiomics research for faster clinical translation. *J Nucl Med*. 2018;59:189-193.

Table 1. Key aspects of ML algorithms

Algorithm	**Task**	**Supervision?**	**Model**	**Typical cost function**	**Computational burden**	**Assumptions / comments**
Naïve Bayes classification	Classification	Supervised	Several e.g. Gaussian…	Probabilistic	Low	Naïve probability distribution
Linear regression	Regression	Supervised	hyperplane	MSE	Low	
Support vector machine	Classification	Supervised	hyperplane	Classification rate	Moderate	Handle complex problems
	Regression			MSE		
Random forest	Classification	Supervised	Tree	Classification rate	Low-moderate	Tolerant to overfitting
	Regression			MSE		
Artificial neural network	Classification or regression	Supervised (typical); unsupervised / reinforcement learning (less common)	Neurons connected in layers	Classification rate, MSE	High	Used for complex problems; may be convolutional and/or deep
k-means clustering	Clustering	Unsupervised	Cluster centroid	Distance to cluster center	Moderate (depends on problem)	Identifies centroids and assigns data to the nearest centroid
Hierarchical clustering	Clustering	Unsupervised	Dendogram	Distance between data points	Moderate (depends on problem)	Clusters data by identifying data-points that are similar
Principal component analysis	Dimensionality reduction	Unsupervised	Principal components		Moderate (depends on problem)	

MSE: Mean Square Error

Figure 1. Machine learning tasks. A,B: Input measurements classified as circles or triangles for supervised learning (e.g. triangles could be positive cases, and circles be negative cases). C: Input measurements not classified (e.g. clinical information unknown). D. Classification using line ‘2’ is better fit than line ‘1.’ E. Classification requires nonlinear approach. F. Potential clustering.

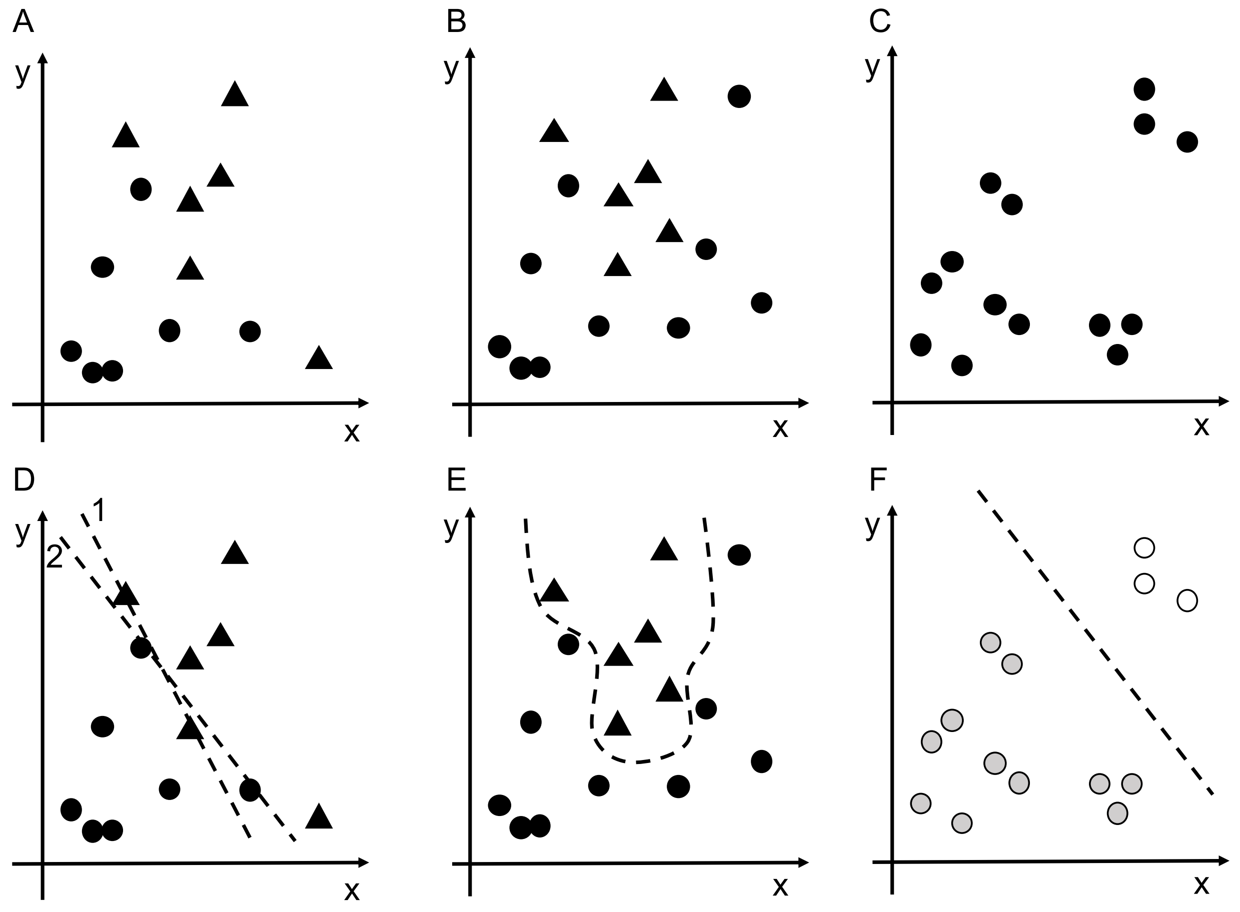


Figure 2. Illustration of a support vector machine. A. Training dataset shown as a scatter plot. B. Solid line with maximum distance to closest data-points in each category; solid points indicate the closest points, known as *support vectors*. Note there are no data points in the region between the dashed lines.

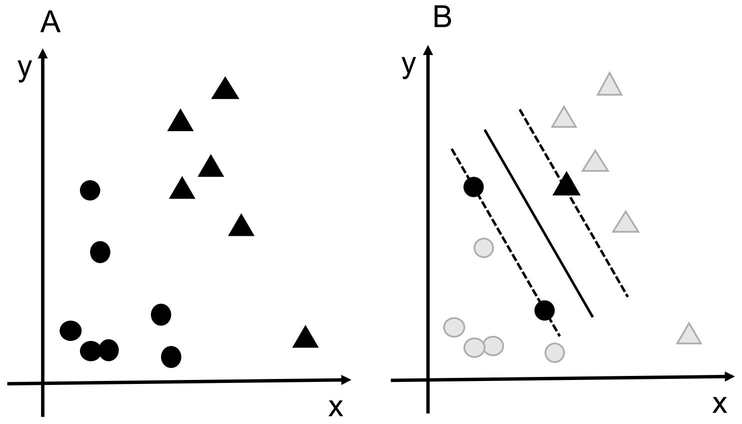
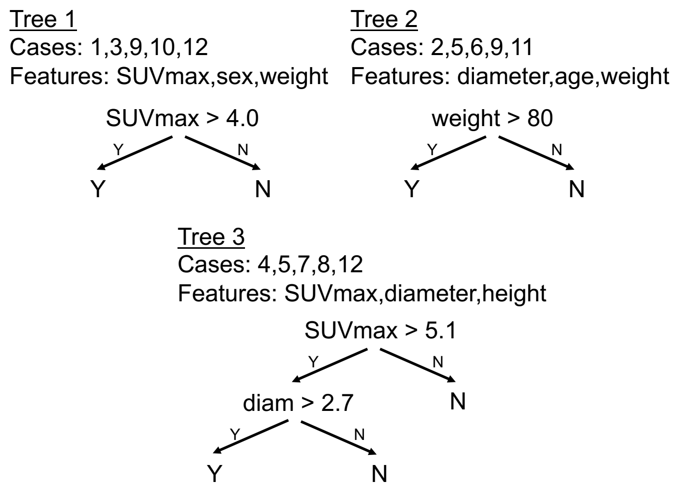


Figure 3. Illustration of a random forest. A. Training dataset with 12 cases, each with 6 features, and an outcome for supervised training. B. 3 trees, each generated from 5 randomly selected cases and 3 randomly selected features; for trees 1 and 2, one feature is sufficient to fully separate the 5 cases, and for tree 3, two features are needed. C. New case where trees 1 and 3 return Y, and tree 2 returns N, so the majority decision is Y.

A

	SUVmax	diameter	sex	age	height	weight	disease?
1	7.2	3.5	M	62	171	75	Y
2	2.1	2.0	M	78	165	76	N
3	6.0	2.0	F	71	162	70	Y
4	1.5	4.0	M	83	173	81	N
5	5.9	1.5	F	66	148	68	N
6	5.5	7.0	F	68	177	85	Y
7	4.4	7.5	F	59	166	64	N
8	8.9	6.5	F	88	146	61	Y
9	1.7	3.5	F	89	155	63	N
10	2.3	4.0	M	71	185	83	N
11	2.9	3.5	M	62	180	91	Y
12	5.8	4.0	F	66	170	71	Y

B



C New case: SUVmax = 5.3, diameter = 3.5, sex = M, age = 62, height = 171, weight = 72
Tree 1: Y, Tree 2: N, Tree 3: Y, so decision is YES

Figure 4. A. ANN with 4 layers, and neurons (circles). Arrows indicate values passed from one layer to another. B. Sample CNN for brain PET processing. Two layers each perform a 2D convolution followed by a rectified linear unit (ReLU) and pooling (taking the maximum value of every 4 pixels to reduce the total number of pixels). The resulting matrix (feature map) is "flattened" into a 1D array of neurons that are fully connected to another layer. A nonlinear operation ("soft max") is performed to generate a classification. For simplicity we show a single input slice; however, CNNs can process higher-dimensional or multi-modality images. Parameters including: number of layers, number of pixels in a convolutional filter, convolution stride, and number of pixels pooled called *hyperparameters*. C. Image processing used in (23), with two paths processing images at different scales, each with contrast limited adaptive histogram equalization (CLAHE), a deep CNN, and a subsequent random forest.

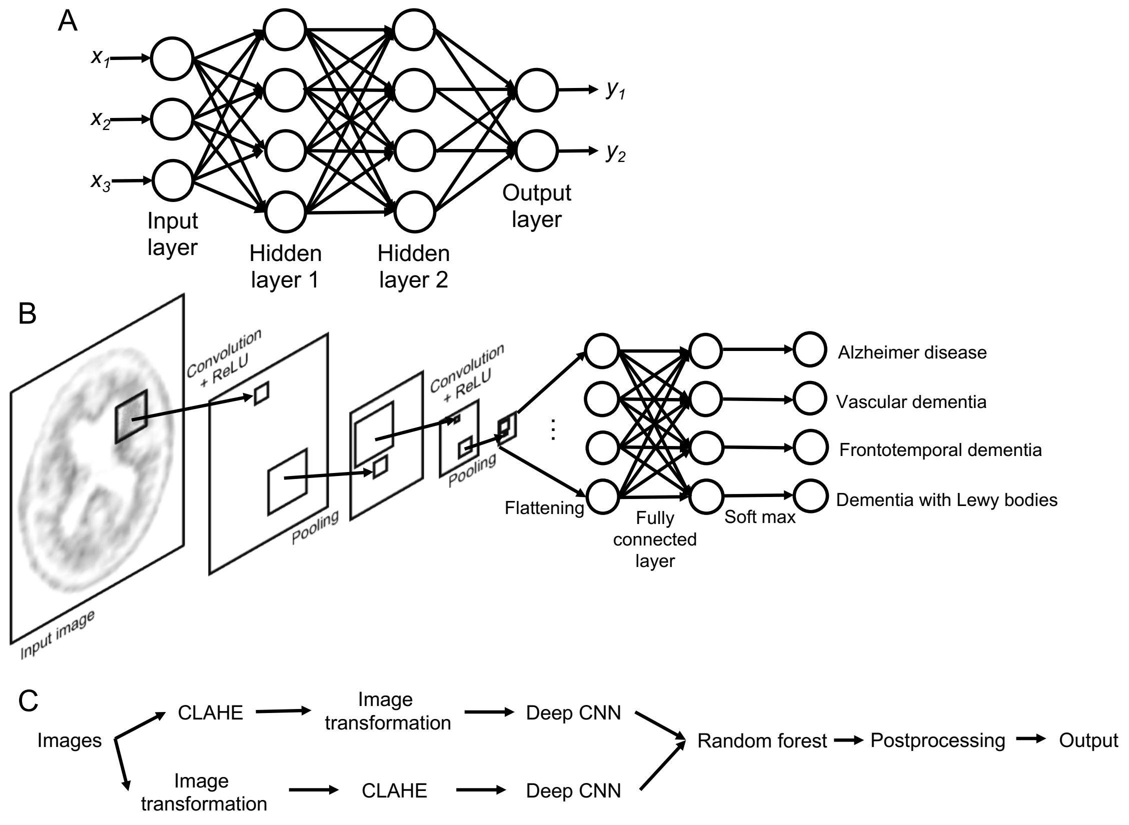


Figure 5. Illustration of PCA. A. Training dataset shown as scatter plot. B. Spread of data is greater along rotated axis x' than y'. C. Component y' is removed since it is less important than x'. D. Data points are moved to closest point along x' axis; compressed (lossy) data is represented by measurement along x', rather than using x and y.

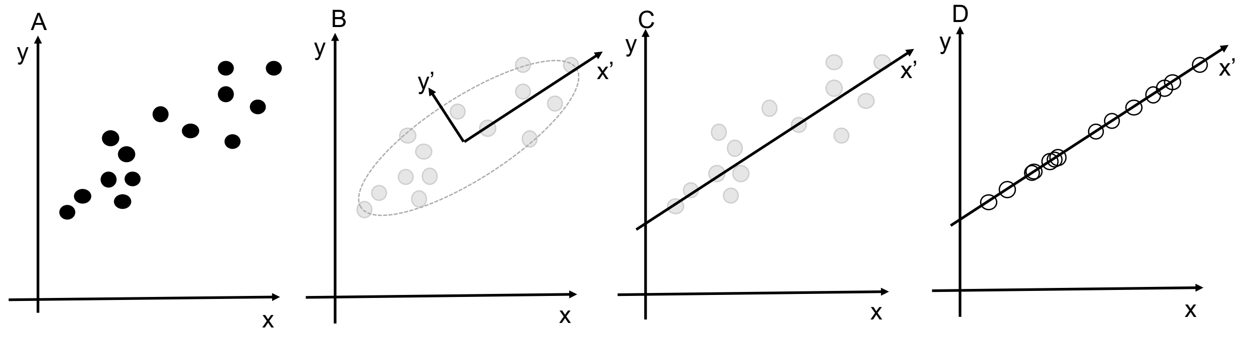


Figure 6: Illustration of K-means clustering. A. Training dataset shown as scatter plot (circles); stars indicate initial guesses for centers of two clusters. B. Solid line at midway point between stars separates data into two clusters (gray and white). C. Stars are moved to the centers of each cluster. D. Updated solid line at midway point between stars; one data point changes cluster. E. Stars are moved to updated centers; no data point changes cluster, so the algorithm converges.

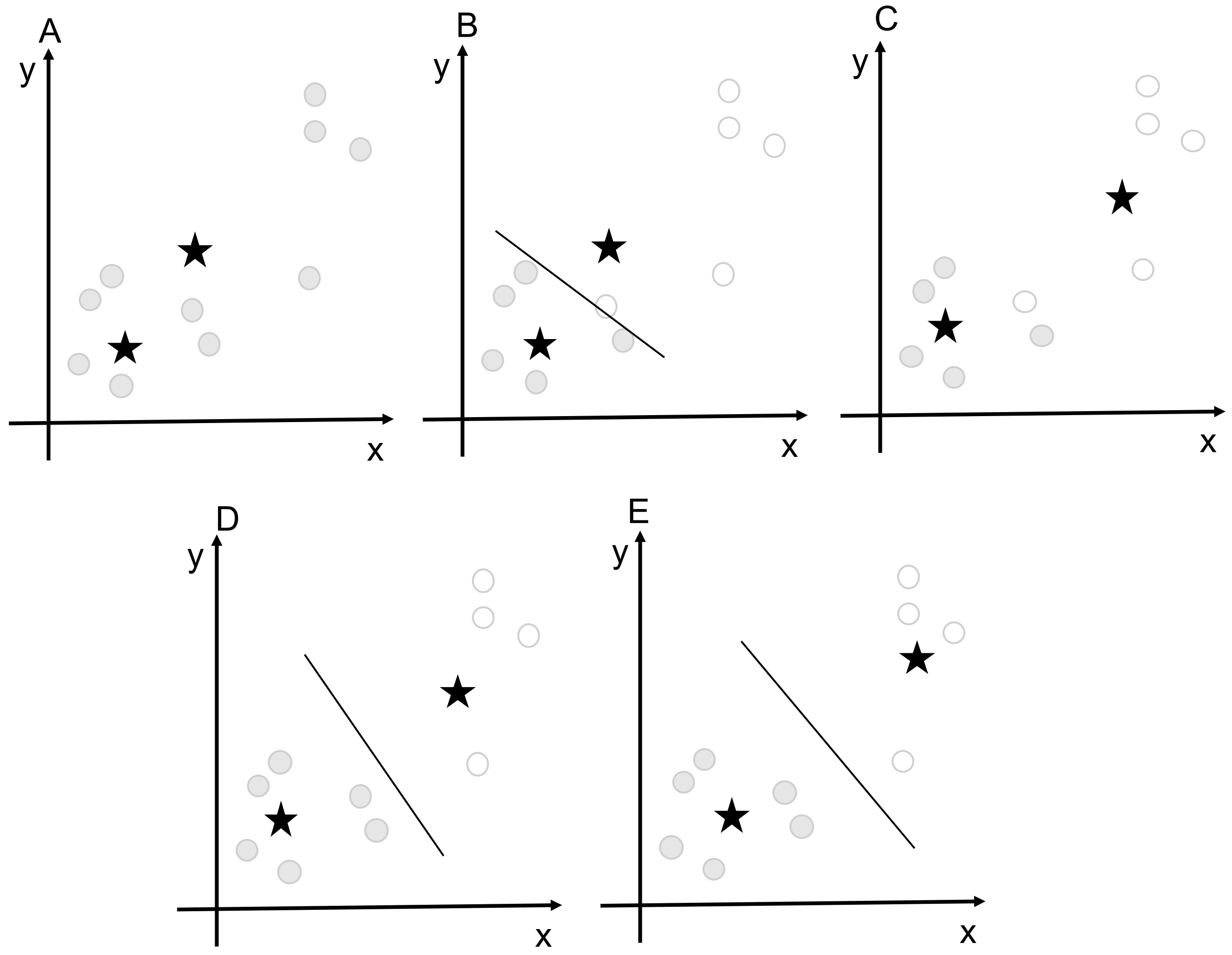
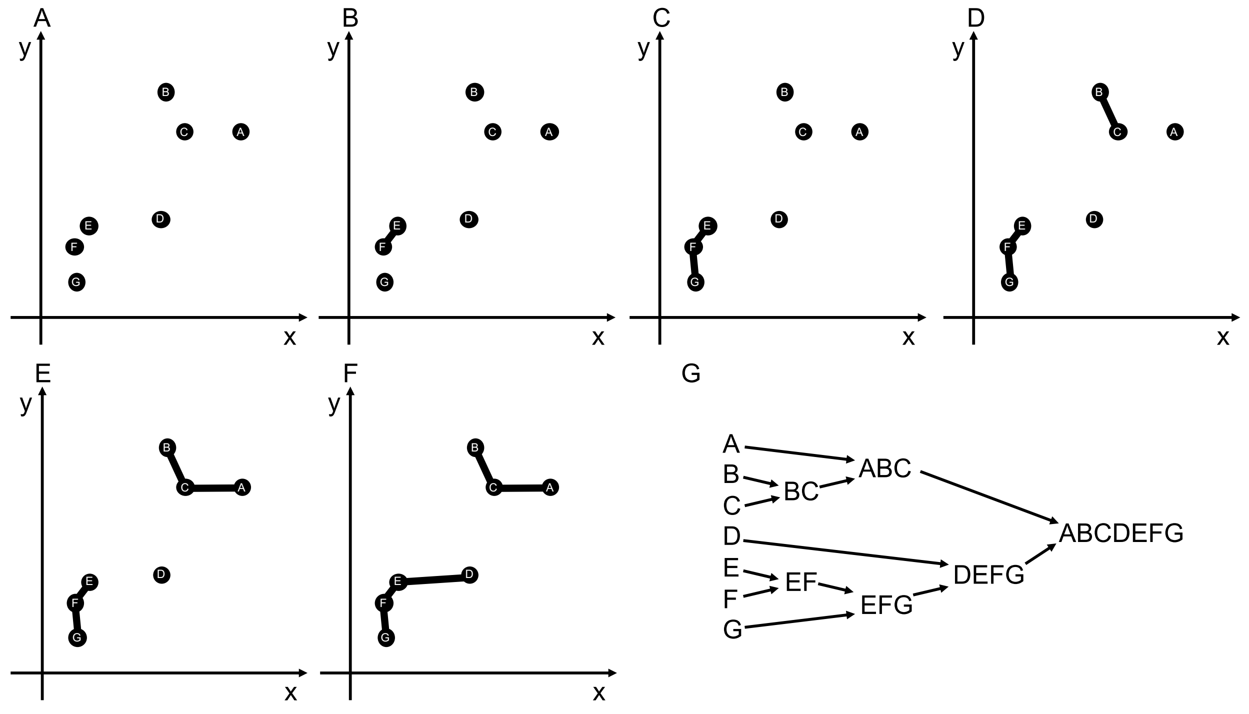


Figure 7: Illustration of hierarchical clustering. A. Training dataset shown as scatter plot. B. Data-points *E* and *F* are similar, and are assigned to new cluster *EF*. C. Nearest neighbors are *G* and *EF* (*G* is closest to point *F* within *EF*), and are grouped into *EFG*. D. Data-points *B* and *C* are similar, and are assigned to new cluster *BC*. E. Nearest neighbors are *A* and *BC* (*A* is closest to point *C* within *BC*), and are grouped into *ABC*. F. Nearest neighbors are *D* and *EFG*, so they are grouped into *DEFG*. G. Dendrogram indicating how clusters are hierarchically formed.


A
B
C
D
E
F
G
x
y
ABC
BC
DEFG
EF
EFG
ABCDEFG