getting started with
Machine Learning

Part 2 | The well-known algorithms

:: Now let's study some popular algorithms

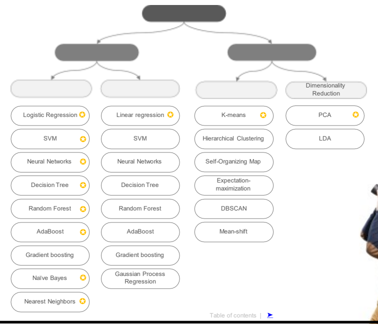


Table of contents | ➤

Naïve Bayes

Table of contents | ➤

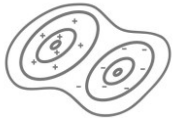
:: Naïve Bayes

Naïve Bayes is a simple but important probabilistic model

Naive Bayes is a simple multiclass classification algorithm based on applying Bayes' theorem with the **"naive" assumption** of independence between the features. It assumes that the conditional probabilities of the independent variables are statistically independent.

It computes the conditional probability distribution of each feature given label, and then it applies Bayes' theorem to compute the conditional probability distribution of label given an observation and use it for prediction[1].

It classifies new data based on the **highest probability** of its belonging to a particular class.



Naive Bayes is a simple classifier model that is [3]:

- Based on the Bayes theorem
- Supervised Learning
- Easy to build
- Faster to train, compared to other models
- Often used as a baseline classifier for benchmarking

1 Source: http://spark.apache.org/docs/2.2.0/mllib-naive-bayes.html
3 Source: https://www.slideshare.net/ananth/an-overview-of-nave-bayes-classifier

:: Naïve Bayes

Naïve Assumption

It assume that each input feature x_i is conditionally independent of every other feature x_j, given the class C.

$$P(X1, X2, \ldots, Xn \mid C) = P(X1 \mid C) \cdot P(X2 \mid C) \cdot \ldots \cdot P(Xn \mid C)$$

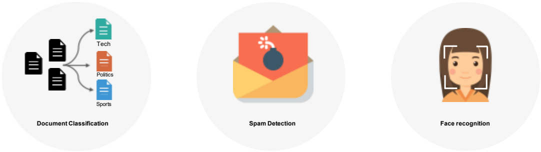
For example, a fruit may be considered to be an apple if it is red, round, and about 10 cm in diameter.

A naive Bayes classifier considers each of these features to contribute independently to the probability that this fruit is an apple, regardless of any possible correlations between the color, roundness, and diameter features[2].

:: Naïve Bayes

Application of Naïve Bayes

Although the assumption that the predictor (independent) variables are independent is not always accurate, it does simplify the classification task dramatically[1]. Naive Bayes classifiers have worked quite well in many real-world situations, famously document classification and spam filtering. They require a small amount of training data to estimate the necessary parameters[2]. Naive Bayes learners and classifiers can be extremely fast compared to more sophisticated methods[3].

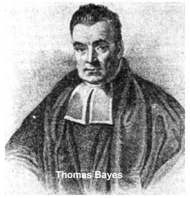


http://scikit-learn.org/stable/modules/naive_bayes.html
Icon made by Freepik] from www.flaticon.com
1 Source: http://documentation.statsoft.com/STATISTICAHelp.aspx?path=MachineLearning/MachineLearning/Overviews/NaiveBayesClassifierIntroductoryOverview

:: Naïve Bayes

About Bayes Theorem (Bayes' rule)

Bayes theorem named after Rev. Thomas Bayes. It works on **conditional probability**. Conditional probability is the probability that something will happen, given that something else has already occurred. Using the conditional probability, we can calculate the probability of an event using its prior knowledge[1].



It tells us how often A happens given that B happens

$$\underbrace{P(A \mid B)} = \frac{P(B \mid A)\,P(A)}{P(B)}$$

- **P (A | B)** is "Probability of A given B", the probability of A given that B happens.
- **P (A)** is Probability of A
- **P (B | A)** is "Probability of B given A", the probability of B given that A happen
- **P (B)** is Probability of B

For example:

- P(Fire | Smoke) means how often there is fire when we see smoke.
- P(Smoke | Fire) means how often we see smoke when there is fire.

Source: http://dataaspirant.com/2017/02/06/naive-bayes-classifier-machine-learning/

:: Naïve Bayes

Naïve Bayes classifier

Classification is to derive the maximum posteriori, i.e., the **maximal P (C_k| X)**

- Given training data X = (X_1, X_2, X_3, ... X_n), representing some *n* features (independent variables)
  - for example, there are 3 features : X = (Pants, long hair, red T-Shirt)
- Suppose C represents the class variable.
  - for example, there are 2 classes. C_1= Man , C_2= Woman
- Using Bayes' theorem, the conditional probability can be decomposed as

$$\underbrace{P(C_k \mid X_1,\ldots,X_n)}_{\text{Posterior Probability}} = \frac{P(X_1,\ldots,X_n \mid C_k)\,P(C_k)}{P(X_1,\ldots,X_n)} = \frac{P(X_1 \mid C_k)\,P(X_2 \mid C_k)\ldots P(X_n \mid C_k)\;P(C_k)}{P(X_1,\ldots,X_n)}$$

For example, there is a new observation :
a person who wears pants and red T-shirt and has long hair

- P (Man | Pants, long hair, red T-Shirt) = 0.7
- P (Woman | Pants, long hair, red T-Shirt) = 0.3

Classify *X* to the class which has bigger posterior *P*

- Predicts X belongs to C_i if $P(C_i \mid X_1,\ldots,X_n)$ is the highest among all the $P(C_k \mid X)$ for all the *k* classes

In this example, this person is predicted as Man because its posteriori probability 0.7 is bigger than 0.3

:: Naïve Bayes

Example : Can we play tennis today ?

Lets say we have a table that decided if we should play tennis under certain circumstances. These could be the **outlook** of the weather; the **temperature**; the **humidity** and the strength of the **wind:**

Day	Outlook	Temperature	Humidity	Wind	Play Tennis ?
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No

there are 5 cases of not being able to play a game, and 9 cases of being able to play a game.

If we were given a new instance :
X = (**Outlook** = Sunny, **Temperature** = Cool, **Humidity** = High, **Wind** = Strong)

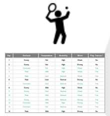
We want to know if we can play a game or not ?

Source: https://computersciencesource.wordpress.com/2010/01/28/year-2-machine-learning-naive-bayes-classifier/

:: Naïve Bayes

Example : Can we play tennis today ?

Here we have 4 attributes. What we need to do is to create "look-up tables" for each of these attributes, and write in the probability that a game of tennis will be played based on this attribute.



Outlook	Play = Yes	Play = No	Total
Sunny	2/9	3/5	5/14
Overcast	4/9	0/5	4/14
Rain	3/9	2/5	5/14

Temperature	Play = Yes	Play = No	Total
Hot	2/9	2/5	4/14
Mild	4/9	2/5	6/14
Cool	3/9	1/5	4/14

Humidity	Play = Yes	Play = No	Total
High	3/9	4/5	7/14
Normal	6/9	1/5	7/14

Wind	Play = Yes	Play = No	Total
Strong	3/9	3/5	6/14
Weak	6/9	2/5	8/14

If we were given a new instance :
X = (**Outlook** = Sunny, **Temperature** = Cool, **Humidity** = High, **Wind** = Strong), can we play the game ?

Firstly we look at the probability that we can play the game

- P (Outlook=Sunny | Play=Yes) = 2/9
- P (Temperature=Cool | Play=Yes) = 3/9
- P (Humidity=High | Play=Yes) = 3/9
- P (Wind=Strong | Play=Yes) = 3/9
- P (Play=Yes) = 9/14

Next we consider the fact that we cannot play a game:

- P (Outlook=Sunny | Play=No) = 3/5
- P (Temperature=Cool | Play=No) = 1/5
- P (Humidity=High | Play=No) = 4/5
- P (Wind=Strong | Play=No) = 3/5
- P (Play=No) = 5/14

Source: https://computersciencesource.wordpress.com/2010/01/28/year-2-machine-learning-naive-bayes-classifier/

:: Naïve Bayes

Example : Can we play tennis today ?

If X = (Outlook = Sunny, Temperature = Cool, Humidity = High, Wind = Strong) · then

P (Play=Yes | X) = P (Play=Yes | Outlook = Sunny, Temperature = Cool, Humidity = High, Wind = Strong)

$$= \frac{P(\text{Outlook = Sunny, Temperature = Cool, Humidity = High, Wind = Strong} \mid \text{Play=Yes}) * P(\text{Play=Yes})}{P(\text{Outlook = Sunny, Temperature = Cool, Humidity = High, Wind = Strong})}$$

$$= \frac{P(\text{Outlook = Sunny} \mid \text{Play=Yes}) * P(\text{Temperature = Cool} \mid \text{Play=Yes}) * P(\text{Humidity = High} \mid \text{Play=Yes}) * P(\text{Wind = Strong} \mid \text{Play=Yes}) * P(\text{Play=Yes})}{P(\text{Outlook=Sunny}) * P(\text{Temperature=Cool}) * P(\text{Humidity=High}) * P(\text{Wind=Strong})}$$

$$= \frac{(2/9) * (3/9) * (3/9) * (3/9) * (9/14)}{(5/14) * (4/14) * (7/14) * (6/14)}$$

$$= \frac{0.0053}{0.02186} = \mathbf{0.2424}$$

P (Play= No | X) = P (Play= NO | Outlook = Sunny, Temperature = Cool, Humidity = High, Wind = Strong)

$$= \frac{(3/5) * (1/5) * (4/5) * (3/5) * (5/14)}{(5/14) * (4/14) * (7/14) * (6/14)} = \frac{0.0206}{0.02186} = \mathbf{0.9421}$$

- P(Play=Yes | X) = 0.2424
- P(Play=No | X) = 0.9421

Since 0.9421 is greater than 0.2424 then the answer is 'no', we cannot play a game of tennis today.

Source: https://computersciencesource.wordpress.com/2010/01/28/year-2-machine-learning-naive-bayes-classifier/

:: Naïve Bayes

More details about Naive Bayes classifier

$$\underbrace{P(C_k \mid X_1,\dots,X_n)}_{\text{Posterior Probability}} = \frac{P(X_1,\dots,X_n \mid C_k)\, P(C_k)}{P(X_1,\dots,X_n)}$$

Likelihood
which is the probability of *predictor* given *class*

Prior Probability
of Class

Posterior Probability
of *class* (*target*) given *predictor* (*attribute*).

Prior Probability
of Evidence

Informally, this can be viewed as

Posteriori = Likelihood ✕ Priori / Evidence

Estimating a-posteriori probabilities

In the Bayesian's interpretation for probability, probability measures a "degree of belief." To evaluate the probability of a hypothesis, the Bayesian probabilistic specifies some prior probability, which is then updated to a posterior probability in the light of new, relevant data (evidence)[1].

- Before observing the data, our prior beliefs can be expressed in a prior probability distribution that represents the knowledge we have about the unknown features[2].
- After observing the data our revised beliefs are captured by a posterior distribution over the unknown features[3].

1 Source: https://en.wikipedia.org/wiki/Bayesian_probability
2&3 Source: Gladys Castillo, Naïve Bayes

:: Naïve Bayes

More details about Naive Bayes classifier

$$P(C_k \mid X_1,\dots,X_n) = \frac{P(X_1,\dots,X_n \mid C_k)\, P(C_k)}{P(X_1,\dots,X_n)}$$

P(x) can be ignored

- $P(X_1,\dots,X_n)$ can be ignored because it is the same for all the classes (a normalization constant).

Naïve Assumption: attribute independence

- It's difficult to learn the probability $P(X_1,\dots,X_n \mid C_k)$. It required initial knowledge of many probabilities, involving significant computational cost
- However, if we assume that the feature probabilities are conditionally independent (i.e., no dependence relationship between all input features), it can be simplified as follow.

$$P(X_1, X_2,\dots, X_n \mid C) = P(X_1 \mid C) \cdot P(X_2 \mid C) \cdot \ldots \cdot P(X_n \mid C)$$

therefore,

$$P(C_k \mid X_1,\dots,X_n) = \frac{P(X_1,\dots,X_n \mid C_k)\, P(C_k)}{P(X_1,\dots,X_n)} = \frac{P(X_1 \mid C_k)\, P(X_2 \mid C_k) \dots P(X_n \mid C_k)\; P(C_k)}{P(X_1,\dots,X_n)}$$

1 Source: https://en.wikipedia.org/wiki/Bayesian_probability

:: Naïve Bayes

If the features are continuous value

- One common technique for handling continuous values is to use binning to discretize the feature values, to obtain a new set of Bernoulli-distributed features
- **Another approach** : $P(X_j \mid C = c_i)$ is estimated assuming it has **Gaussian (normal) distribution**.

$$\hat{P}(X_j \mid C = c_i) = \frac{1}{\sqrt{2\pi}\sigma_{ji}} \exp\left(-\frac{(X_j - \mu_{ji})^2}{2\sigma_{ji}^2}\right)$$

μ_{ji} : mean (average) of feature values X_j of examples for which $C = c_i$

σ_{ji} : standard deviation of feature values X_j of examples for which $C = c_i$

Example

Example: Continuous-valued Features

- Temperature is naturally of continuous value.

Play Golf ?
Yes: 25.2, 19.3, 18.5, 21.7, 20.1, 24.3, 22.8, 23.1, 19.8
No: 27.3, 30.1, 17.4, 29.5, 15.1

Estimate mean and variance for each class

- $\mu_{Yes} = 21.64$, $\sigma_{Yes} = 2.35$
- $\mu_{No} = 23.88$, $\sigma_{No} = 7.09$

- **Learning Phase:** Output two Gaussian models for P(temperature | C)

$$\hat{P}(x \mid Yes) = \frac{1}{2.35\sqrt{2\pi}} \exp\left(-\frac{(x-21.64)^2}{2 \times 2.35^2}\right) = \frac{1}{2.35\sqrt{2\pi}} \exp\left(-\frac{(x-21.64)^2}{11.09}\right)$$

$$\hat{P}(x \mid No) = \frac{1}{7.09\sqrt{2\pi}} \exp\left(-\frac{(x-23.88)^2}{2 \times 7.09^2}\right) = \frac{1}{7.09\sqrt{2\pi}} \exp\left(-\frac{(x-23.88)^2}{50.25}\right)$$

Source of example: Chen Ke, Generative Models and Naïve Bayes

:: Naïve Bayes

Avoid the Zero frequency problem

- If a given class and feature value never occur together in the training data, then the frequency-based probability estimate will be zero. This is problematic because it will wipe out all information in the other probabilities when they are multiplied [1]

What if you've never seen feature before?

For example , if we receive an email that contains a word that has never appeared in the training emails [2]

- P(**x** | y) will be 0 for all y values. This is bad !
- Just because an event has not happened before, does not mean that it won't ever happen [3]
- Naïve Bayesian prediction requires each conditional probability be non-zero. Otherwise, the predicted probability will be zero

Solution: Use Laplacian Correction [4]

For example, suppose a dataset with 1000 tuples, income=low (0), income= medium (990) and income = high (10)

Use **Laplacian correction** (or Laplacian estimator)

- Adding 1 to each case
  - Prob (income = low) = 1/1003
  - Prob (income = medium) = 991/1003
  - Prob(income = high) = 11/1003
- The "corrected" probablity estimates are close to their "uncorrected" counterparts

1 Source: https://en.wikipedia.org/wiki/Naive_Bayes_classifier
2 Source: http://classes.engr.oregonstate.edu/eecs/winter2011/cs434/notes/bayes-6.pdf
3 Source: https://classes.soe.ucsc.edu/cmps140/Winter17/slides/3.pdf
4 Source: Jiawei Han, Micheline Kamber, and Jian Pei, Data Mining: Concepts and Techniques

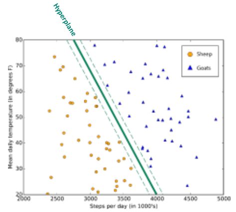
Support Vector Machine

Table of contents

:: Support Vector Machine (SVM)

SVMs are a set of supervised learning methods used for classification, regression and outliers detection

Classifies data by finding the linear decision boundary (hyperplane) that separates all data points of one class from those of the other class. The best hyperplane for an SVM is the one with the largest margin between the two classes, when the data is linearly separable. [1]



Support vector machines (SVMs) find the boundary that separates classes by as wide a margin as possible. SVMs are capable of both classification and regression.

Best Used...[2]

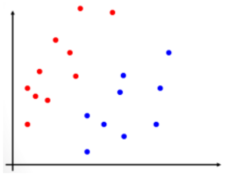
- For data that has exactly two classes (you can also use it for multiclass classification with a technique called error- correcting output codes)
- For high-dimensional, nonlinearly separable data
- When you need a classifier that's simple, easy to interpret, and accurate

1 &2 Source: Mathworks, Applying Supervised Learning

:: Support Vector Machine (SVM)

How to classify the data set in the following example ?

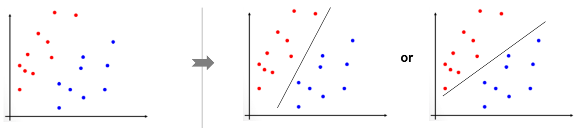
There is a two dimensional data set with two classes. How to classify it into 2 classes correctly?



:: Support Vector Machine (SVM)

How to classify the data set in the following example ?

There is a two dimensional data set with two classes. How to classify it into 2 classes correctly?



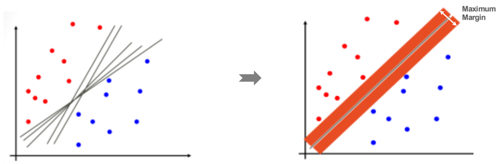
A two dimensional data set with two classes

Both lines can be linear classifiers. Actually there are more lines that can be linear classifiers.

:: Support Vector Machine (SVM)

Which line is the best classifier ?

There is a two dimensional data set with two classes. How to classify it into 2 classes correctly?



There are many lines that can be linear classifiers. Which one is the optimal classifier ?

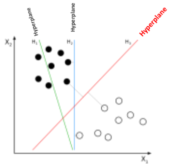
- Define the margin of a linear classifier as the width that the boundary could be increase by before hitting a data data point.[1]
- The maximum margin linear classifier is the simplest kind of SVG(called Linear SVM) [2]

1,2 source: https://www.cs.cmu.edu/~cga/ai-course/svm.pdf

:: Support Vector Machine (SVM)

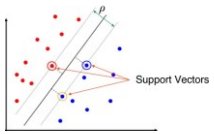
Select the hyper-plane which segregates the two classes better

a good separation is achieved by the hyper-plane that has the largest distance to the nearest training data points of any class (so-called functional margin)



- H_1 does not separate the classes.
- H_2 does, but only with a small margin.
- **H_3 separates them with the maximum margin.**

Source: https://en.wikipedia.org/wiki/Support_vector_machine

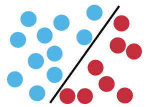


- Examples closet to the hyper-plane are ***support vectors***
- **Margin** μ of the separator is the distance between support vectors.

:: Support Vector Machine (SVM)

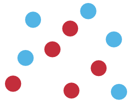
There is a new challenge !

How to classify the nonlinearly separable data by the hyperplane?



Use Linear SVM algorithm

The linear SVM can work well on classifying this data set.



How to classify this data set ?

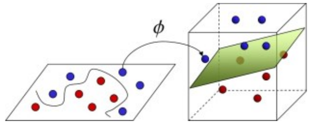
It seems that it's hard to classify this data set via the linear classifier.

Source: http://bytesizebio.net/2014/02/05/support-vector-machines-explained-well/

:: Support Vector Machine (SVM)

Create nonlinear classifiers by applying the kernel trick

SVMs sometimes use a kernel transform to transform nonlinearly separable data into higher dimensions where a linear decision boundary can be found.



Input Space **Feature Space**

Kernel trick allows the algorithm to fit the maximum-margin hyperplane in a transformed feature space. Although the classifier is a hyperplane in the transformed feature space, it may be nonlinear in the original input space.

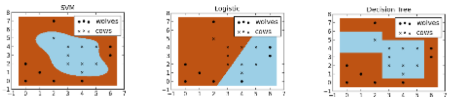
https://www.youtube.com/watch?v=9NrALgHFwTo

Image Source: http://bytesizebio.net/2014/02/05/support-vector-machines-explained-well/

:: Support Vector Machine (SVM)

Example: SVM using a non-linear kernel

If you are farmer and you need to set up a fence to protect your cows from packs of wolves, where do you build your fence?



Where do you build your fence ?

Well if you're a really data driven farmer one way you could do it would be to build a classifier based on the position of the cows and wolves in your pasture. Trying a few different types of classifiers, we see that SVM does a great job at separating your cows from the packs of wolves. I thought these plots also do a nice job of illustrating the benefits of using a non-linear classifiers.

You can see the the logistic and decision tree models both only make use of straight lines. [1]

1 & Image source:https://www.kdnuggets.com/2017/02/yhat-support-vector-machine.html

:: Support Vector Machine (SVM)

Strengths of SVM

Advantages

- The major strengths of SVM are the training is relatively easy. No local optimal, unlike in neural networks.[1]
- Non-linear SVMs use a non-linear kernel. Non-linear SVM means that the boundary that the algorithm calculates doesn't have to be a straight line. The benefit is that you can capture much more complex relationships between your data points without having to perform difficult transformations. The downside is that the training time is much longer, because it is much more computationally intensive. [2]
- SVMs have a regularization parameter, which can help avoid over-fitting. [3]
- Effective in high dimensional spaces.[4]
- Still effective in cases where number of dimensions is greater than the number of samples.[5]
- Uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.[6]

http://scikit-learn.org/stable/modules/svm.html

:: Support Vector Machine (SVM)

Weakness of SVM

Disadvantage

- For classification, the SVM is only directly applicable for two-class tasks. Therefore, algorithms that reduce the multi-class task to several binary problems have to be applied. Well, many SVM packages already have built-in multi-class classification functionality. Otherwise, use one-versus-the-rest approach to build a multiclass classifier
- Unlike Logistic Regression classifiers, SVMs do not directly provide probability estimates. In many classification problems you actually want the probability of class membership
- Parameters of a solved model are difficult to interpret.
- Long training time on large data sets
- Choosing a "good" kernel function can be tricky.

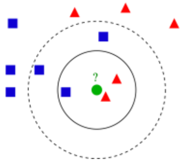
1 Sourece: https://open.sap.com/courses/ds...
2 source:https://...02/yhat-support-vector-machine.html

k Nearest Neighbor

:: k Nearest Neighbor (kNN)

kNN categorizes objects based on the classes of their nearest neighbors in the dataset

kNN predictions assume that objects near each other are similar. Distance metrics, such as Euclidean, city block, cosine, and Chebychev, are used to find the nearest neighbor.



Example of *k*-NN classification

The test sample (green circle) should be classified either to the first class of blue squares or to the second class of red triangles.

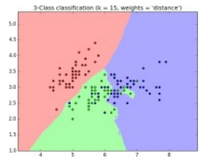
- If *k = 3* (solid line circle) it is assigned to the second class because there are 2 triangles and only 1 square inside the inner circle.
- If *k = 5* (dashed line circle) it is assigned to the first class (3 squares vs. 2 triangles inside the outer circle).

1 Source : Mathworks, Applying Supervised Learning
Source of diagram: wikipedia

:: k Nearest Neighbor (kNN)

An object is classified by a majority vote of its neighbors

- **KNN algorithm is one of the simplest classification algorithm**
  Despite its simplicity, nearest neighbors has been successful in a large number of classification and regression problems, including handwritten digits or satellite image scenes. [1]
- **Often used in classification**
  Being a non-parametric method, it is often successful in classification situations where the decision boundary is very irregular.[2]
- **Classification is computed from a simple majority vote of the nearest neighbors of each point**
  In k-NN classification, the output is a class membership. An object is classified by a majority vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors.
- **K is constant specified by user**
  In the classification phase, k is a user-defined constant. The best choice of k depends upon the data; generally, larger values of k reduce the effect of noise on the classification, but make boundaries between classes less distinct.
- **KNN is computationally expensive**

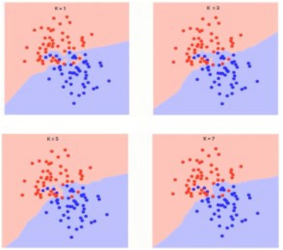


Source: https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm
1,2 source: http://scikit-learn.org/stable/modules/neighbors.html
Source of diagram: http://scikit-learn.org/stable/auto_examples/neighbors/plot_classification.html#sphx-glr-auto-examples-neighbors-plot-classification-py

:: k Nearest Neighbor (kNN)

How do we choose the factor K?

let's try to see the effect of value "K" on the class boundaries. Following are the different boundaries separating the two classes with different values of K.[1]



If you watch carefully, you can see that the boundary becomes smoother with increasing value of K. With K increasing to infinity it finally becomes all blue or all red depending on the total majority.

At times, choosing K turns out to be a challenge while performing KNN modeling.[2]

1,2 & image Source: https://www.analyticsvidhya.com/blog/2014/10/introduction-k-neighbours-algorithm-clustering/

:: k Nearest Neighbor (kNN)

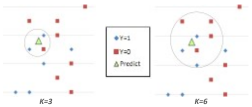
Strengths and Weakness of kNN

Strengths [1]

- It is beautifully simple and logical

Weaknesses [2]

- It may be driven by the choice of k, which may be a bad choice.
- Generally, larger values of k reduce the effect of noise on the classification, but make boundaries between classes less distinct.
- The accuracy of the algorithm can be severely degraded by the presence of noisy or irrelevant features, or if the feature scales are not consistent with their importance.
- In binary (two class) classification problems, it is helpful to choose k to be an odd number, as this avoids tied votes.
- It is important to review the sensitivity of the solution to different values of k.



Best Used...[3]

- When you need a simple algorithm to establish benchmark learning rules
- When memory usage of the trained model is a lesser concern
- When prediction speed of the trained model is a lesser concern

1,2 Souece: https://open.sap.com/courses/ds1/items/4lcgezLoUcuOfkxLM1y8#0
3 Source : Mathworks, Applying Supervised Learning

K-mean

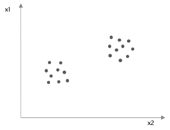
Table of contents | ➤

Unsupervised Learning

Discover the structure within the un-labeled data

Unsupervised machine learning is the machine learning task of inferring a function to describe hidden structure from **"unlabeled"** data. Examples of unsupervised learning tasks include

- **clustering** (where we try to discover underlying groupings of examples)
- **anomaly detection** (where we try to infer if some examples in a dataset do not conform to some expected pattern)[1].



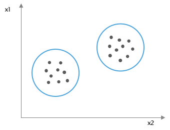
Training set : $\{x^{(1)}, x^{(2)}, x^{(3)}, \ldots, x^{(m)}\}$

Example: **Clustering**

➡

A data set without labels is grouped into 2 clusters using K-means algorithm.

Note: There is no target variable to be predicted



Training set : $\{x^{(1)}, x^{(2)}, x^{(3)}, \ldots, x^{(m)}\}$

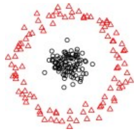
1Source: https://www.coursera.org/learn/machine-learning/lecture/czmip/unsupervised-learning-introduction

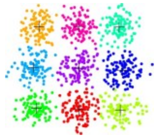
Cluster Analysis

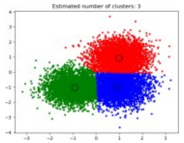
A cluster is a subset of data which are similar

Clustering is a technique for finding similarity groups in a data, called clusters. It attempts to group individuals in a population together by similarity, but not driven by a specific purpose. Clustering is often called an unsupervised learning, as you don't have prescribed labels in the data and no class values denoting a priori grouping of the data instances are given[1].

A cluster is a collection of data items which are "similar" between them, and "dissimilar" to data items in other clusters[2]







1 Source: http://bigdata-madesimple.com/possibly-the-simplest-way-to-explain-k-means-algorithm/
2 Source: http://www.mit.edu/~9.54/fall14/slides/Class13.pdf
picture: http://scikit-learn.org/stable/auto_examples/cluster/plot_mean_shift.html
Picture : http://www.stat.cmu.edu/~ryantibs/statml/lectures/clustering.pdf

Application of Clustering

Clustering is widely used in many applications

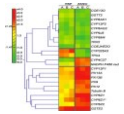
Clustering is hard to evaluate, but very useful in practice. Here are some examples of clustering applications

Market segmentation

Discover customer groups and use them for targeted marketing programs

Social network Analysis

Help you find a group of coherent friends on the social network



Genomics

Finding groups of gene with similar expressions

Source of picture: https://www.freepik.com/free-vector/social-media-network_716594.htm#term=social_network&page=1&position=3
Source of picture: https://www.researchgate.net/figure/7981540_fig2_Fig-2-Clustering-analysis-of-gene-expression-patterns-between-the-permethrin-resistant

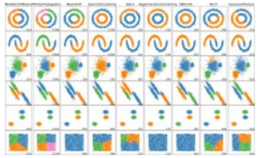
Clustering Methods

More than 100 Clustering algorithms

Clustering has a long history and still is in active research. There are possibly over 100 published clustering algorithms. Some of the methods of clustering include:

- **Partitioning methods (Centroid-based clustering)**
  - Find mutually exclusive clusters of spherical shape[1]
  - Distance-based [2]
  - k-means is most popular algorithm due to its simplicity and efficiency
- **Connectivity-based methods (Hierarchical clustering)**
  - Agglomerative hierarchical clustering ("bottom-up")
  - Divisive hierarchical clustering ("top-down")
- **Density-based methods**
  Popular examples of density models are DBSCAN and its variations (e.g. OPTICS)
- **Grid-based methods**
  Use a multi-resolution grid data structure[3]
- **Distribution-based clustering**
  A popular example of these models is Expectation-Maximization algorithm

Comparing different clustering algorithms on toy dataset

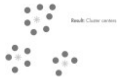


http://scikit-learn.org/stable/auto_examples/cluster/plot_cluster_comparison.html

1, 2,3 Source: http://users.math.yale.edu/users/gw289/CpSc-445_2016/Slides/CPSC445%20-%20Topic%2008%20-%20Clustering.pdf

:: k-Means

Let's start to learn k-means



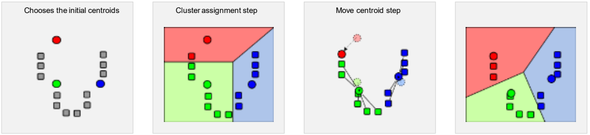
k-Means

Partitions data into k number of mutually exclusive clusters. How well a point fits into a cluster is determined by the distance from that point to the cluster's center[1].

1 Source : Mathworks, Applying Supervised Learning

:: k-Means

A graphical view of K-means algorithm



1. *k* initial "means" (in this case *k*=3) are randomly generated within the data domain (shown in color).
2. *k* clusters are created by associating every observation with the nearest mean.
   In Cluster assignment step, the algorithm goes through each of the data points and depending on which cluster is closer, whether the red cluster centroid or the blue cluster centroid or the green; It assigns the data points to one of the three cluster centroids.
3. The centroid of each of the *k* clusters becomes the new mean.
   In move centroid step, K-means moves the centroids to the average of the points in a cluster. In other words, the algorithm calculates the average of all the points in a cluster and moves the centroid to that average location.
4. Steps 2 and 3 are repeated until convergence has been reached
   In other words, it repeats until the centroids do not move significantly.

Source of diagram: https://en.wikipedia.org/wiki/K-means_clustering

:: k-Means

K-means (Llyod's) clustering algorithm

K-Means clustering intends to partition n objects into k clusters in which each object belongs to the cluster with the nearest mean

1. Randomly initialize *K* **cluster centroids** $\mu_1, \mu_2, \ldots, \mu_k \in R^n$
2. Repeat until convergence : {
   - For every example i, set

     $$c^{(i)} := j \quad \text{that minimizes} \quad \underbrace{\|x^{(i)} - \mu_j\|^2}_{\text{Euclidean distance}}$$

     Centroid for cluster : μ_j = mean of the cluster

     Cluster assignment step
     Assign each data point to the nearest cluster whose mean has the least squared Euclidean distance.

   - For every k, set
     μ_k := average (mean) of points assigned to the cluster k

     $$\mu_k := \frac{1}{|C_k|}\sum_{i \in C_k} x^{(i)}$$

     Update centroid step (⇒ Move Centroid)
     Calculate the new means to be the centroids of the observations in the new clusters.

}

Recommended video:
https://www.youtube.com/watch?v=LvgcfMOyREE&t=159s

Source of formular: http://stanford.edu/~cpiech/cs221/handouts/kmeans.html

:: k-Means

Weakness of K-means

K-Means is simple and can be used for a wide variety of data type It's also quite computaDonally efficient , even though multiple runs are often performed[1].. However, similar to other algorithm, K-mean clustering has many weaknesses. For example,

- **The number of cluster "*k*" must be specified in advance.**
  The number of clusters k is an input parameter: an inappropriate choice of k may yield poor results[2]. Unfortunately there is no global theoretical method to find the optimal number of clusters. A practical approach is to compare the outcomes of multiple runs with different *k* and choose the best one based on a predefined criterion. In general, a large *k* probably decreases the error but increases the risk of overfitting[3]. For details, see next slide.

- **Sensitive to initial centroids selection, which leads to unwanted solution**
  One of the major drawback of K-means clustering is the random choice of initial cluster centers .For every different run of the algorithm on the same dataset, the initial centers might be different. This may lead to different clustering results. Therefore, it is very hard to repeat the clustering results.

  There are some methods for addressing this issue.
  - k-means is usually run many times, starting with different random centroids each time. The results can be compared by a numeric measure such as the clusters' distortion, which is the sum of the squared differences between each data point and its corresponding centroid [4]. Then, pick clustering that gave lowest cost function (distortion). *For more details, ⇒ Video (7min) by Andrew Ng,* https://www.youtube.com/watch?v=PpH_hv55GNQ
  - One of the other methods is K-means ++ algorithm.

- **k-means can only handle numerical data**[5]
  Categorical data (i.e., category labels such as gender, country, browser type) needs to be encoded or separated in a way that can still work with the algorithm[6].

1 Source: https://www-users.cs.umn.edu/~kumar/dmbook/ch8.pdf 2 source: https://en.wikipedia.org/wiki/K-means_clustering
3Source: http://www.saedsayad.com/clustering_kmeans.htm 4 Source: Data Science for business
5 Source: https://www.inovex.de/blog/disadvantages-of-k-means-clustering 6Source: https://www.datascience.com/blog/k-means-clustering

:: k-Means

Weakness of K-means *(continued)*

- **The algorithm may get stuck in the local optimum**
  Given enough time, K-means will always converge, however this may be to a local minimum (namely, a locally best clustering). This is highly dependent on the initialization of the centroids[1].

(a) Optimal Clustering

The figure shows a clustering solution that is the **global minimum**

(b) Suboptimal Clustering

The figure shows a suboptimal clustering that is only a **local minimum.**

For more details, check it out on this short video (7minutes) : https://www.youtube.com/watch?v=PpH_hv55GNQ

1 Source: http://scikit-learn.org/stable/modules/clustering.html

:: k-Means

Weakness of K-means *(continued)*

- **Sensitive to outliers and noise, which results in an inaccurate partition**
  When the squared error criterion is used, outliers can unduly influence the clusters that are found[2]. Weakness of arithmetic mean is not robust to outliers. Very far data from the centroid may pull the centroid away from the real one[3].
  - Because of this, it is often useful to discover outliers and eliminate them beforehand[4].
  - It is important, however, to appreciate that there are certain clustering applications for which outliers should not be eliminated[5].

2 · 4 & 5 Source: Introduction to Data Mining · https://www-users.cs.umn.edu/~kumar/dmbook/ch8.pdf
3 Source: http://people.revoledu.com/kardi/tutorial/kMean/Weakness.htm

:: k-Means

Weakness of K-means *(continued)*

- **K-means cannot handle non-globular clusters or clusters of different sizes and densities**
  K-means has difficulty detecting the "natural" clusters, when clusters have non-spherical shapes or widely different sizes or densities.

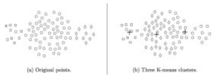


Figure 1: K-means with clusters of <u>different size</u>

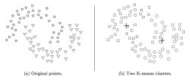


Figure 3: K-means with <u>non-globular clusters</u>

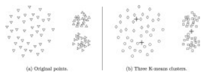


Figure 2: K-means with clusters of <u>different density</u>

:: k-Means

Weakness of K-means *(continued)*

- **K-means cannot handle clusters of different sizes**

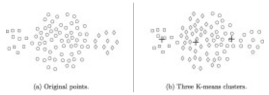


Figure 1: K-means with clusters of <u>different size</u>

In Figure 1, K-means cannot find the three natural clusters because one of the clusters is much larger than the other two, and hence, the larger cluster is broken, while one of the smaller clusters is combined with a portion of the larger cluster.

:: k-Means

Weakness of K-means *(continued)*

- **K-means cannot handle clusters of densities**

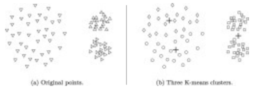


Figure 2: K-means with clusters of <u>different density</u>

In Figure 2, K-means fails to find the three natural clusters because the two smaller clusters are much denser than the larger cluster.

:: k-Means

Weakness of K-means *(continued)*

- **K-means cannot handle non-globular clusters**

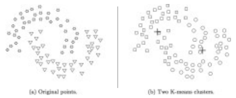


Figure 3: K-means with <u>non-globular clusters</u>

In Figure 3 , K-means finds two clusters that mix portions of the two natural clusters because the shape of the natural clusters is not globular.

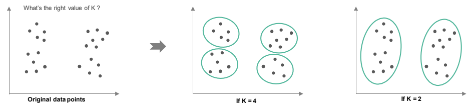
:: k-Means

How to choose the value of *k* ?

In k-means clustering we must choose a value for k. This is still an active area of research and there are no definitive answers[1]. By far the most common way of choosing the number of clusters, is still choosing it **manually** by looking at visualizations or by looking at the output of the clustering algorithm or something else[2]

Why is it not easy to determine the right value of K?

Because it is often generally ambiguous about how many clusters there are in the data. For instance, the data points in the following example can be viewed as 4 clusters or 2 clusters. There are no absolutely right answer for that



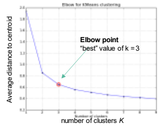
Source: CMU http://www.stat.cmu.edu/~ryantibs/statml/lectures/clustering.pdf
2 Source: https://www.youtube.com/watch?v=lbR5brSyvrY

:: k-Means

How to choose the value of *k* ? *(continued)*

In general, there is no method for determining exact value of *K*, but an accurate estimate can be obtained using the following techniques[1].

- **Elbow Method**
  - Try different k, looking at the change in the average distance to the centroid, as k increases[2].
  - Average falls rapidly until right k, then falls much more slowly[3]
  - It might be hard to determine where the elbow of the curve actually is



Unfortunately, the elbow method often does not work well in practice because there may not be a well-defined elbow[4].

Video : Choosing The Number Of Clusters

Youtube Video (8 min)

https://www.youtube.com/watch?v=lbR5br5yvrY&t=24s

- **Silhouette method**

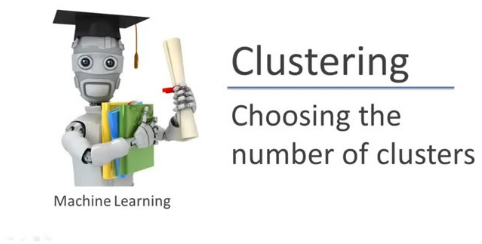
  The silhouette value is a measure of how similar an object is to its own cluster (cohesion) compared to other clusters (separation).
  Silhouette values lies in the range of [-1, 1]. Silhouette coefficients near +1 indicate that the sample is far away from the neighboring clusters. More details ⇒ Video (3 min)

1 Source: h^ps://www.datascience.com/blog/k-means-clustering
2&3 Source: h^ps://www.youtube.com/watch?v=RD0nNK51Fp8&t=415s
4Source: h^ps://www.coursera.org/learn/machine-learning/lecture/Ks0E9/choosing-the-number-of-clusters

:: k-Means

How to choose the value of *k* ? *(continued)*

In general, there is no method for determining exact value of *K*, but an accurate estimate can be obtained using the following techniques[1].



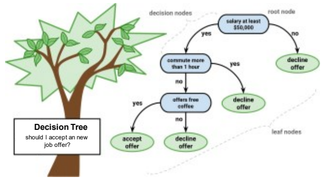
Decision Tree

:: Decision Tree

How it Works

Decision tree builds **classification** or **regression** models in the form of a tree structure. A decision tree lets you predict the value of a target variable by following the decisions in the tree from the root (beginning) down to a leaf node. [1]

A tree consists of branching conditions where the value of a predictor is compared to a trained weight. The number of branches and the values of weights are determined in the training process. Decision trees are prone to overfitting, additional modification, or pruning, may be used to simplify the model. [2]



Decision trees are a non-parametric supervised learning method used for

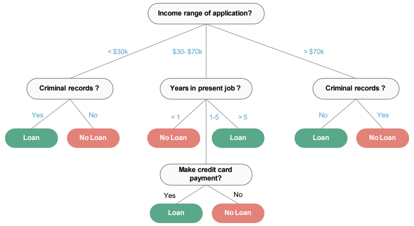
- **Classification**
- **Regression**

1 & 2 Source : Mathworks, Applying Supervised Learning
image: h>ps://cdn-images-1.medium.com/max/2000/1*S10T4oh3_JadQ-eY6HauOQ.png

:: Decision Tree

Example 1 : using decision tree to decide whether or not to offer someone a loan

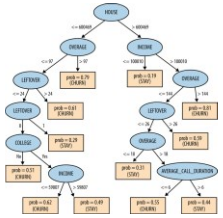
Decision trees can be applied to both classification and regression problems. We might have a decision tree to help a financial institution decide whether a person should be offered a loan



:: Decision Tree

Example 2: Classification tree learned from the cellular phone churn data

A Decision Tree can also estimate the probability that an instance belongs to a particular class k. [1]



Probabilities at the leaves are the estimated probabilities of churning for the corresponding segment;

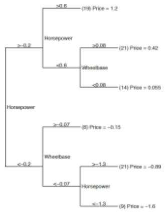
In parentheses are shown the classifications resulting from applying a decision threshold of 0.5 to the probabilities (i.e., are the individuals in the segment more likely to CHURN or to STAY?). [2]

Variable	Explanation
COLLEGE	Is the customer college educated?
INCOME	Annual income
OVERAGE	Average overcharges per month
LEFTOVER	Average number of leftover minutes per month
HOUSE	Estimated value of dwelling (from census tract)
HANDSET_PRICE	Cost of phone
LONG_CALLS_PER_MONTH	Average number of long calls (15 mins or over) per month
AVERAGE_CALL_DURATION	Average duration of a call
REPORTED_SATISFACTION	Reported level of satisfaction
REPORTED_USAGE_LEVEL	Self-reported usage level
LEAVE (Target variable)	Did the customer stay or leave (churn)?

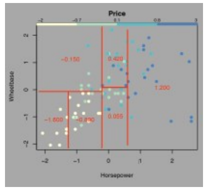
1 Hands-On Machine Learning with Scikit-Learn and TensorFlow
2 & diagram : Data Science for Business

:: Decision Tree

Example 3 : using decision tree to predict the price of 1993-model cars



Note that the order in which variables are examined depends on the answers to previous questions. The numbers in parentheses at the leaves indicate how many cases (data points) belong to each leaf[1]



The partition of the data implied by the regression tree from left figure. . Notice that all the dividing lines are parallel to the axes, because each internal node checks whether a single variable is above or below a given value[2]

1,2 & images source: http://www.stat.cmu.edu/~cshalizi/350-2006/lecture-10.pdf

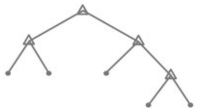
:: Decision Tree

Tree construction algorithms

There are various algorithms used in decision tree

- **ID3 :** It's for classification. It was developed in 1986 by Ross Quinlan.
- **C4.5 :** It's for classification. It is the successor to ID3
- **C5.0:** C5.0 is Quinlan's latest version release under a proprietary license. It uses less memory and builds smaller rulesets than C4.5 while being more accurate.
- **CART** (Classification and Regression Trees): It can used for both classification and regression problem.

  Scikit-learn , a famous machine learning library for Python, uses an optimized version of the CART algorithm.

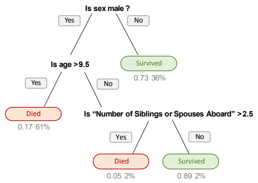


A Decision Tree model contains rules to predict the target variable.

:: Decision Tree

Decision Tree generates the output as a tree-like structure

A decision tree is built top-down from a root node. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with **decision nodes** and **leaf nodes**. A decision node (e.g., Outlook) has two or more branches (e.g., Sunny, Overcast and Rainy). Leaf node (e.g., Play) represents a classification or decision. The topmost decision node in a tree which corresponds to the best predictor called root node. [1]



A tree showing survival of passengers on the Titanic. The figures under the leaves show the probability of survival and the percentage of observations in the leaf.

A decision tree is drawn upside down with its root node at the top.

In the image left, the bold text in black represents a **condition**/internal node, based on which the tree splits into branches/ edges.

The end of the branch that doesn't split anymore is the decision/leaf, in this case, whether the passenger died or survived, represented as red and green text respectively.[2]

1 http://www.saedsayad.com/decision_tree.htm

2 Source: https://medium.com/towards-data-science/decision-trees-in-machine-learning-641b9c4e8052

:: Decision Tree

A must-watch video which explains the high level concept of Decision tree

Example

Example: Predict if John will play tennis

Training example: 9 yes / 5 No

Day	Outlook	Humidity	Wind	Play
D1	Sunny	High	Weak	No
D2	Sunny	High	Strong	No
D3	Overcast	High	Weak	Yes
D4	Rain	High	Weak	Yes
D5	Rain	Normal	Weak	Yes
D6	Rain	Normal	Strong	No
D7	Overcast	Normal	Strong	Yes
D8	Sunny	High	Weak	No
D9	Sunny	Normal	Weak	Yes
D10	Rain	Normal	Weak	Yes
D11	Sunny	Normal	Strong	Yes
D12	Overcast	High	Strong	Yes
D13	Overcast	Normal	Weak	Yes
D14	Rain	High	Strong	No

Decision Tree 1: how it works

https://www.youtube.com/watch?v=eKD5gxPPeY0&t=48s

:: Decision Tree

Constructing decision tree

A decision tree is built top-down from a root node.

- Strategy: top- down Recursive divide-and-conquer fashion[1]

  1. **First: select attribute for root node**
     Create branch for each possible attribute value
  2. **Then: split instances into subsets**
     One for each branch extending from the node
  3. **Finally: repeat recursively for each branch, using only instances that reach the branch**

- Stop if all instances have the same class[2]

⇒ Pure subsets

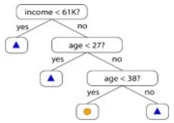
Recursively partitioning the data into subsets that contain instances with similar values (homogenous)

Recursively partitions the training set until each partition consists entirely or dominantly of examples from one class. Each non-leaf node of the tree contains a split point that is a test on one or more attributes and determines how the data is partitioned. The tree is built by recursively partitioning the data. Partitioning continues until each partition is either '**pure**' (all members belong to the same class) or sufficiently small (a parameter set by the user).[3]

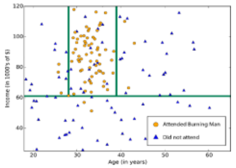
1,2source: Marina Santini Decision Trees 3 Source: https://www.ibm.com/support/knowledgecenter/en/SSEPGG_9.5.0/com.ibm.im.model.doc/c_decision_tree_calssification.html

:: Decision Tree

Want subsets to be as <u>homogeneous</u> as possible



A decision tree subdivides a feature space into regions of roughly uniform values



Growing a tree involves deciding on **which features to choose** and **what conditions to use** for splitting, along with knowing **when to stop**.

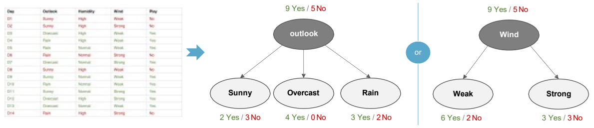
In this procedure all the features are considered and different split points are tried and tested using a cost function. The split with the best cost (or lowest cost) is selected[1].

1 Source: https://medium.com/towards-data-science/decision-trees-in-machine-learning-641b9c4e8052
2 Source: https://www.ibm.com/support/knowledgecenter/en/SSEPGG_9.5.0/com.ibm.im.model.doc/c_decision_tree_calssification.html
Source of diagram: https://docs.microsoft.com/en-us/azure/machine-learning/machine-learning-algorithm-choice

:: Decision Tree

The key is to decide which attribute to split on

You need to pick the best attribute for splitting the training examples.



- **To compare the different ways to split data in a node**
- **Want to measure the "purity" of the split[2].**
  - Pure set (4yes/ 0 No) ⇒ completely certain(100%)
  - Impure (3 yes/ 3 No) ⇒ completely uncertain(50%)

- **Splits on all attributes are tested**
  - Constructing a decision tree is all about finding attribute that returns the highest information gain or lowest Gini Index (i.e., the most homogeneous branches)[3].

1 Source: https://medium.com/towards-data-science/decision-trees-in-machine-learning-641b9c4e8052
2 source: https://www.youtube.com/watch?v=AmCV4g7_-QM&t=11s
3 Source: http://www.saedsayad.com/decision_tree.htm

:: Decision Tree

Measures that can be used to capture the purity of split

Information Gain, Gain Ratio and Gini Index are the most common methods of *attribute selection*

Information Gain | Information Gain Ratio | Gini Index

:: Decision Tree

We need to understand Entropy first before we move on

ID3 uses *Entropy* and *Information Gain* to construct a decision tree. It's necessary to understand what Entropy means.

What's Entropy ?

http://www.saedsayad.com/decision_tree.htm

:: Decision Tree

Generally entropy is a measure of <u>disorder</u> or <u>uncertainty</u>

Entropy is a concept used in Physics, mathematics, computer science (information theory) and other fields of science.[1] The concept of entropy originated in thermodynamics as a measure of molecular disorder: entropy approaches zero when molecules are still and well ordered. [2]

Entropy in Physics:

Entropy, so far, had been a concept in physics. If the particles inside a system have many possible positions to move around, then the system has **high entropy**, and if they have to stay rigid, then the system has **low entropy.** [2]

For example, water in its three states, solid, liquid, and gas, has different entropies. The molecules in ice have to stay in a lattice, as it is a rigid system, so ice has low entropy. The molecules in water have more positions to move around, so water in liquid state has medium entropy. The molecules inside water vapor can pretty much go anywhere they want, so water vapor has high entropy.[3]

1 source: https://bricaud.github.io/personal-blog/entropy-in-decision-trees/
2 Source: Hands-On Machine Learning with Scikit-Learn and TensorFlow
3,4 & image source: https://medium.com/udacity/shannon-entropy-information-gain-and-picking-balls-from-buckets-5810d35d54b4

:: Decision Tree

Entropy in Information theory

Information entropy is defined as the average amount of information produced by a stochastic source of data… Generally, information entropy is the average amount of information conveyed by an event

- **when a low-probability event occurs** (i.e. when the data source has a lower-probability value,)
  the event carries **more "information" ("surprisal")** than when the source data has a higher-probability value. Rarer events provide more information when observed.[1]

- **when a high-probability event occurs ,**
  the event with high-probability value is less informative than less common event. Entropy is zero when one outcome is certain to occur.

Entropy is a measure of *unpredictability* of the state, or equivalently, of its *average information content*. To get an intuitive understanding of these terms, consider the example of a political poll. [2]

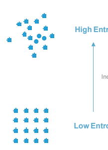
- Usually, such polls happen because the outcome of the poll is not already known. In other words, the outcome of the poll is relatively *unpredictable*, and actually performing the poll and learning the results gives some new *information*; these are just different ways of saying that the *a priori* entropy of the poll results is large. [3]
- Now, consider the case that the same poll is performed a second time shortly after the first poll. Since the result of the first poll is already known, the outcome of the second poll can be predicted well and the results should not contain much new information; in this case the *a priori* entropy of the second poll result is small relative to that of the first.[4]

1 source: https://en.wikipedia.org/wiki/Entropy_(information_theory)

:: Decision Tree

More uncertainty, more entropy!

entropy is a measure of
disorder or **uncertainty**



- High Disorder
- Messy
- High Randomness;
- High uncertainty
- Low certainty

- Well- ordered
- Neat
- Low Randomness
- Low uncertainty
- High certainty

:: Decision Tree

Mathematical definition of entropy

The measure of information entropy associated with each possible data value is the negative logarithm of the *probability mass function* for the value.[1] If we have a set with n different values in it, we can calculate the entropy as follows:

$$Entropy = -\sum_{i=1}^{n} p_i \log_2 p_i$$

- Where p_i is the **probability** of getting the i^{th} value when randomly selecting one from the set.

In other words, Where there are n classes, and p_i is the probability an object from the i^{th} class appearing.

Example

Entire population (30 instances)

16/30 are blue circles: $\log_2\left(\frac{16}{30}\right) = -0.9$;

14/30 are orange crosses: $\log_2\left(\frac{14}{30}\right) = -1.1$;

$$Entropy = -\sum_{i=1}^{n} p_i \log_2 p_i = -\left(\frac{16}{30}\right)(-0.9) - \left(\frac{14}{30}\right)(-1.1) = 0.99$$

Video: Shannon Entropy and Information Gain (21mins)
https://www.youtube.com/watch?time_continue=1247&v=9r7FIXEAGvs

1 source: https://en.wikipedia.org/wiki/Entropy_(information_theory)
Source of example: Linda Shapiro, Introduction to Artificial Intelligence

:: Decision Tree

Another example of calculating entropy

Use entropy to measure the "purity" of the split

$$Entropy = -\sum_{i=1}^{n} p_i \log_2 p_i$$

Pure set (4 yes / 0 No) ⇒ completely certain(100%)

The entropy is 0 if the sample is completely homogeneous

$$Entropy = -\frac{4}{4}\log_2\frac{4}{4} - \frac{0}{4}\log_2\frac{0}{4} = 0$$

Impure set (3 yes / 3 No) ⇒ completely uncertain(50%)

The entropy is 1 if the sample is an equally divided

$$Entropy = -\frac{3}{6}\log_2\frac{3}{6} - \frac{3}{6}\log_2\frac{3}{6} = 1$$

Video : Decision tree entropy (5mins)
https://www.youtube.com/watch?v=tJmhT3oLXCU

:: Decision Tree

- "信息熵"是度量样本集合纯度最常用的一种指标，假定当前样本集合D中第k类样本所占的比例为$p_k(K=1,2,...,|\mathcal{Y}|)$，则$D$的信息熵定义为

$$\mathrm{Ent}(D) = -\sum_{k=1}^{|\mathcal{Y}|} p_k \log_2 p_k$$

$\mathrm{Ent}(D)$的值越小，则D的纯度越高

- 计算信息熵时约定：若 $p=0$，则$p\log_2 p = 0$
- $\mathrm{Ent}(D)$的最小值为0，最大值为$\log_2|\mathcal{Y}|$

:: Decision Tree

Measures that can be used to capture the purity of split

Information Gain, Gain Ratio and Gini Index are the most common methods of *attribute selection*

Information Gain　Information Gain Ratio　Gini Index

:: Decision Tree

Measures that can be used to capture the purity of split – Information Gain

Information gain increases with the average purity of the subsets. Strategy: choose attribute that gives greatest information gain.

Information Gain

i.e. entropy reduction

- 离散属性a有V个可能的取值$\{a^1,a^2,...,a^V\}$，用a来进行划分，则会产生V个分支结点，其中第v个分支结点包含了D中所有在属性a上取值为a^v的样本，记为D^v。则可计算出用属性a对样本集D进行划分所获得的"信息增益"：

$$\mathrm{Gain}(D,a) = \mathrm{Ent}(D) - \sum_{v=1}^{V} \frac{|D^v|}{|D|}\mathrm{Ent}(D^v)$$

为分支结点权重，样本数越多的分支结点的影响越大

- 一般而言，信息增益越大，则意味着使用属性a来进行划分所获得的"纯度提升"越大
- ID3决策树学习算法[Quinlan, 1986]以信息增益为准则来选择划分属性

3 Source: http://www.saedsayad.com/decision_tree.htm

:: Decision Tree

Measures that can be used to capture the purity of split – Information Gain (continued)

The goal is to decrease in entropy (uncertainty) after splitting. And also to want many items in pure sets

Information Gain

i.e. entropy reduction

Example 1

Entropy before splitting
(parent)

$$Information\ Gain(S,A) = H(S) - \underbrace{\sum_{v \in Value(A)} \frac{|s_v|}{|s|} H(s_v)}_{\text{Entropy after splitting}}$$

(Weighted) Average Entropy of Children

Where
v : possible value of A
S: set of example $\{x\}$
s_v: subset where $x_A = v$

Video: Information Gain (9mins)
https://www.youtube.com/watch?v=2k-ZlWwxoCM

$$H(S_{weak}) = -\frac{6}{8}\log_2\frac{6}{8} - \frac{2}{8}\log_2\frac{2}{8} = 0.81$$

$$H(S_{strong}) = -\frac{3}{6}\log_2\frac{3}{6} - \frac{3}{6}\log_2\frac{3}{6} = 1$$

$$Gain(S,Wind) = H(S) - \frac{8}{14}H(S_{weak}) - \frac{6}{14}H(S_{strong}) = 0.94 - \frac{8}{14}*0.81 - \frac{6}{14}*1 = 0.049$$

:: Decision Tree

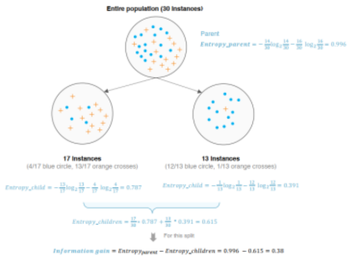
Measures that can be used to capture the purity of split – Information Gain (continued)

The goal is to decrease in entropy (uncertainty) after splitting. And also to want many items in pure sets

Information Gain

i.e. entropy reduction

Example 2



:: Decision Tree

Measures that can be used to capture the purity of split – Information Gain Ratio

Information gain approach has a problem that it bias towards attributes that have a large number of values over attributes that have a smaller number of values. These 'Super Attributes' will easily be selected as the root, resulted in a broad tree that classifies perfectly but performs poorly on unseen instances. We can penalize attributes with large numbers of values by using an alternative method for attribute selection, referred to as Gain Ratio.[1]

Information Gain Ratio

- 增益率定义：

$$\text{Gain\_ratio}(D, a) = \frac{\text{Gain}(D, a)}{\text{IV}(a)}$$

其中

$$\text{IV}(a) = -\sum_{v=1}^{V} \frac{|D^v|}{|D|}\log_2\frac{|D^v|}{|D|}$$

称为属性a的“固有值” [Quinlan, 1993]，属性a的可能取值数目越多（即V越大），则$\text{IV}(a)$的值通常就越大

- 存在的问题

增益率准则对可取值数目较少的属性有所偏好

- C4.5 [Quinlan, 1993]使用了一个启发式：先从候选划分属性中找出信息增益高于平均水平的属性，再从中选取增益率最高的

1 Source: http://www.saedsayad.com/decision_tree_super.htm

:: Decision Tree

Measures that can be used to capture the purity of split – Gini Index

The impurity measure used in building decision tree in CART algorithm is Gini Index. The attribute that maximizes the reduction in impurity is chosen as the splitting attribute

Gini Index

- 数据集D的纯度可用“基尼值”来度量

$$\text{Gini}(D) = \sum_{k=1}^{|\mathcal{Y}|}\sum_{k' \neq k} p_k p_{k'} = 1 - \sum_{k=1}^{|\mathcal{Y}|} p_k^2$$

反映了从D中随机抽取两个样本，其类别标记不一致的概率

$\text{Gini}(D)$越小，数据集D的纯度越高

- 属性a的基尼指数定义为：

$$\text{Gini\_index}(D, a) = \sum_{v=1}^{V}\frac{|D^v|}{|D|}\text{Gini}(D^v)$$

- 应选择那个使划分后基尼指数最小的属性作为最优划分属性，即

$$a_* = \underset{a \in A}{\arg\min}\ \text{Gini\_index}(D, a)$$

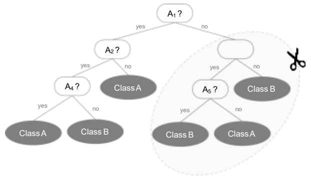
- CART [Breiman et al., 1984]采用“基尼指数”来选择划分属性

1 source: Hands-On Machine Learning with Scikit-Learn and TensorFlow

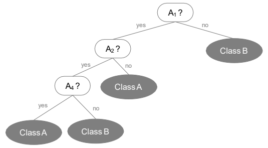
:: Decision Tree

Address overfitting in Decision Trees by Pruning

Pruning means to change the model by deleting the child nodes of a branch node. Pruning usually results in reducing size of tree, avoids unnecessary complexity, and to avoid overfitting.



The main problem with tree induction is that it will keep growing the tree to fit the training data until it creates pure leaf nodes. This will likely result in large, overly complex trees that overfit the data 1



The pruned node is regarded as a leaf node. Leaf nodes cannot be pruned.

1 Source: Data Science for Business
Source of diagram: http://www.inf.unibz.it/dis/teaching/DWDM/slides2011/lesson5-Classification-2.pdf

:: Decision Tree

Approaches for Pruning

pre-pruning

One strategy to prevent overfitting is to prevent the tree from becoming really detailed and complex by stopping its growth early.

- Halt tree construction early—do not split a node if this would result in the goodness measure falling below a threshold[1]
- Upon halting the node becomes a leaf Upon halting, the node becomes a leaf[2]
- The leaf may hold the most frequent class among the subset tuples[3]

Problem: difficult to choose an appropriate threshold[4]

post-pruning

Another strategy is to build a complete tree with pure leaves but then to prune back the tree into a simpler form

- Remove non-significant branches from a "fully grown" tree[5]
- After trimming, replace subtree by a leaf node
- The leaf is labeled with the most frequent class[6]

Practically, post-pruning strategy is more successful because it is not easy to precisely estimate when to stop growing the tree[7].

1,2,3,4,5,6 Source: http://www.inf.unibz.it/dis/teaching/DWDM/slides2011/lesson5-Classification-2.pdf
7 Source: http://www.saedsayad.com/decision_tree_overfitting.htm

:: Decision Tree

Advantages of Decision Tree

- **Simple to understand and to interpret**
  Decision tree is a "white box" model. Decision Trees are able to produce 'understandable' rules. The trees can be also visualized. In contrast, in a black box model (e.g. in neural network, random forest), it is usually hard to explain in simple terms why the predictions were made.
- **To build decision tree requires little data preparation**
  Decision trees do not need feature scaling. Decision trees can deal with a reasonable amount of missing values; Decision trees are also not sensitive to outliers. In contrast, other algorithms often require data normalization, dummy variables need to be created and blank values to be removed[1].
- **Decision trees are able to handle both continuous and categorical variables.**
  Other techniques are usually specialized in analyzing datasets that have only one type of variable[2]. Able to handle multi-output problems[3].
- **Implicitly perform feature selection**
  Decision trees such as CART, for instance, have a built-in mechanism to perform variable selection. Decision trees provide a clear indication of which fields are most important for prediction or classification

1 source: http://scikit-learn.org/stable/modules/tree.html
Image: Designed by Asierromero / Freepik

:: Decision Tree

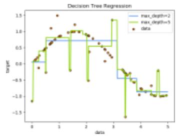
Disadvantages of Decision Tree

- **They are prone to over-fitting.**
  Decision-tree learners can create over-complex trees that do not generalize the data well[1].
- **Decision Tree learner can create biased trees in case of unbalanced data**
  Decision tree learners create biased trees if some classes dominate. It is therefore recommended to balance the dataset prior to fitting with the decision tree.[2]
- **Instability**
  Decision trees can be unstable because small variations in the data might result in a completely different tree being generated. This problem is mitigated by using decision trees within an ensemble[3]. Random Forests can limit this instability by averaging predictions over many trees.
- **Greedy approach used by Decision tree doesn't guarantee best solution**
  Greedy algorithm where locally optimal decisions are made at each node cannot guarantee to return the globally optimal decision tree. This can be mitigated by training multiple trees, where the features and samples are randomly sampled with replacement[4].

:: Decision Tree

Disadvantages of Decision Tree - continued

- **Standard decision trees are restricted by hard, axis-aligned splits of the input space [1]**
  The splits are aligned with the axes of the feature space and this lead to rectilinear decision boundaries.



The deeper the tree, the more complex the decision rules and the fitter the model.

We can see that if the maximum depth of the tree is set too high, the decision trees learn too fine details of the training data and learn from the noise, i.e. they overfit.[3]

Another issue is particularly problematic in regression where we are typically aiming to model smooth functions, and yet the tree model produces piecewise-constant predictions with discontinuities at the split boundaries.[2]

:: Decision Tree Example

编号	色泽	根蒂	敲声	纹理	脐部	触感	好瓜
1	青绿	蜷缩	浊响	清晰	凹陷	硬滑	是
2	乌黑	蜷缩	沉闷	清晰	凹陷	硬滑	是
3	乌黑	蜷缩	浊响	清晰	凹陷	硬滑	是
4	青绿	蜷缩	沉闷	清晰	凹陷	硬滑	是
5	浅白	蜷缩	浊响	清晰	凹陷	硬滑	是
6	青绿	稍蜷	浊响	清晰	稍凹	软粘	是
7	乌黑	稍蜷	浊响	稍糊	稍凹	软粘	是
8	乌黑	稍蜷	浊响	清晰	稍凹	硬滑	是
9	乌黑	稍蜷	沉闷	稍糊	稍凹	硬滑	否
10	青绿	硬挺	清脆	清晰	平坦	软粘	否
11	浅白	硬挺	清脆	模糊	平坦	硬滑	否
12	浅白	蜷缩	浊响	模糊	平坦	软粘	否
13	青绿	稍蜷	浊响	稍糊	凹陷	硬滑	否
14	浅白	稍蜷	沉闷	稍糊	凹陷	硬滑	否
15	乌黑	稍蜷	浊响	清晰	稍凹	软粘	否
16	浅白	蜷缩	浊响	模糊	平坦	硬滑	否
17	青绿	蜷缩	沉闷	稍糊	稍凹	硬滑	否

该数据集包含17个训练样本，$|\mathcal{Y}|=2$，其中正例占 $p_1=\frac{8}{17}$，反例占 $p_2=\frac{9}{17}$，计算得到根结点的信息熵为

$$\mathrm{Ent}(D) = -\sum_{k=1}^{2} p_k \log_2 p_k = -(\frac{8}{17}\log_2\frac{8}{17} + \frac{9}{17}\log_2\frac{9}{17}) = 0.998$$

:: Decision Tree Example

- 以属性“色泽”为例，其对应的3个数据子集分别为 D^1(色泽=青绿)，D^2(色泽=乌黑)，D^3 (色泽=浅白)
- 子集 D^1 包含编号为$\{1,4,6,10,13,17\}$的6个样例，其中正例占 $p_1=\frac{3}{6}$，反例占 $p_2=\frac{3}{6}$，D^2、D^3 同理，3个结点的信息熵为：

$$\mathrm{Ent}(D^1) = -(\tfrac{3}{6}\log_2\tfrac{3}{6} + \tfrac{3}{6}\log_2\tfrac{3}{6}) = 1.000$$
$$\mathrm{Ent}(D^2) = -(\tfrac{4}{6}\log_2\tfrac{4}{6} + \tfrac{2}{6}\log_2\tfrac{2}{6}) = 0.918$$
$$\mathrm{Ent}(D^3) = -(\tfrac{1}{5}\log_2\tfrac{1}{5} + \tfrac{4}{5}\log_2\tfrac{4}{5}) = 0.722$$

- 属性“色泽”的信息增益为

$$\begin{aligned}\mathrm{Gain}(D,\text{色泽}) &= \mathrm{Ent}(D) - \sum_{v=1}^{3}\frac{|D^v|}{|D|}\mathrm{Ent}(D^v)\\ &= 0.998 - (\tfrac{6}{17}\times 1.000 + \tfrac{6}{17}\times 0.918 + \tfrac{5}{17}\times 0.722)\\ &= 0.109\end{aligned}$$

:: Decision Tree Example

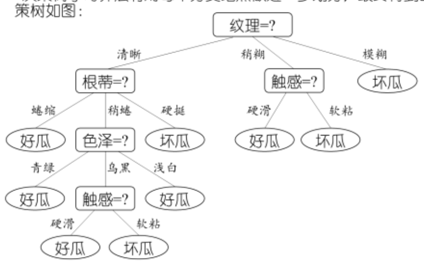
- 类似的，其他属性的信息增益为

$$\mathrm{Gain}(D,\text{根蒂}) = 0.143 \qquad \mathrm{Gain}(D,\text{敲声}) = 0.141$$
$$\mathrm{Gain}(D,\text{纹理}) = 0.381 \qquad \mathrm{Gain}(D,\text{脐部}) = 0.289$$
$$\mathrm{Gain}(D,\text{触感}) = 0.006$$

- 显然，属性“纹理”的信息增益最大，其被选为划分属性

:: Decision Tree Example

- 决策树学习算法将对每个分支结点做进一步划分，最终得到的决策树如图：



:: Decision Tree 练习

ID	年龄	有工作	有自己的房子	信贷情况	类别
1	青年	否	否	一般	否
2	青年	否	否	好	否
3	青年	是	否	好	是
4	青年	是	是	一般	是
5	青年	否	否	一般	否
6	中年	否	否	一般	否
7	中年	否	否	好	否
8	中年	是	是	好	是
9	中年	否	是	非常好	是
10	中年	否	是	非常好	是
11	老年	否	是	非常好	是
12	老年	否	是	好	是
13	老年	是	否	好	是
14	老年	是	否	非常好	是
15	老年	否	否	一般	否

通过所给的训练数据学习一个贷款申请的决策树，用以对未来的贷款申请进行分类，即当新的客户提出贷款申请时，根据申请人的特征利用决策树决定是否批准贷款申请。（自由选择方法）