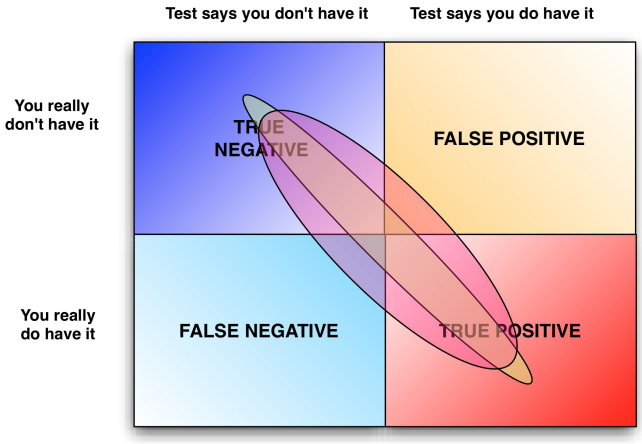
- Categorical data: ordinal, nominal and binary.
- Generative classifiers. Steps in modeling: model selection; density estimation of the data; classification of new data.
- **Bayesian Theorem**. $p(c|x) = \frac{p(c,x)}{p(x)} = \frac{p(x|c)p(c)}{p(x)}$. c: the model to be inferred. x: the observations. $p(x|c)$: the likelihood. $p(c)$ the prior. $p(c|x)$ the posterior. $p(x)$ the evidence.
- Something more about Bayes. $p(c|x)$ means the probability of c given x. Take a concrete example. $p(L|W) = 0.75$ (from Wikipedia)—the probability of a woman with long hair is 75%. Or rather, the probability of event L given event W is 0.75.
- Even something more. $P(A|B) = \frac{P(B|A)P(A)}{P(B)}$. $P(A)$ the prior, the initial degree of belief in A. $P(A|B)$ the posterior, the degree of belief having accounted for B.
- **Bayes decision rule**. $p(x)$ is independent of the class and $p(c)$ is frequently assumed to be the *same* for all classes.
- Standard *k-NN*. When we use big k, then we could have a better noise resistance. At the same time we have worse resolution.
- Kernel density estimation. When we increase the number of kernels, better smoothness is obtained. But we will have a much higher dimension. (You know what this implies.)
- Calculate the joint probability distribution is normally very difficult, but Naive Bayesian method solves this by assuming that class conditional distributions are all independent. $p(c_i|x) = \frac{p(x_1,x_2,\ldots,x_n|c_i)p(c_i)}{p(x)} = \frac{p(x_1|c_i)p(x_2|c_i)\ldots p(x_n|c_i)p(c_i)}{p(x)}$.
- Kind note about Naive Bayesian classifier. The evidence $p(x)$ is class-independent, which means it has nothing to do with, the variable. Aka, the stuff that we *should* really look into.
- **Multivariate Bernoulli** model. For this model, one weakness is that whether a word (in Anti-Spamming case) appears 1 or 100 times, the final probability representation is the same. (Let's do it in Chinese. It's NOT scientific.)
- **Multinomial** event model. For this model, each "word" (in spamming example) is an event. One word appears, then it's probability is calculated. Appears many times, then times as many.
- **Naive Bayesian Classifier for Continuous Data**. When we try to extend NBC to classify data with continuous data, there is one "best practice" that we should think about. The basic idea is to divide each feature into *two or more* bins.
- Equal Frequency Discretization. In this method, each bin contains the same amount of points. (If you do want to "imagine" what point is like, in hyper space.) It's analogous to *k-NN* approach.
- **Fuzzy Discretization**. Form k equally spaced bins; the corresponding likelihood includes contributions from *every training instance*.
- **Lazy Discretization**. The determination of $p(x|c)$ is postponed. When query x is presented, place it at *center* of the bin. The bin scale thus equals to $[x-\delta, x+\delta]$. $p(x|c)$ is then proportional to the number of instances within the specific scale.
- Non-disjoint discretization. Bins are set in advance, but are overlapping. The actual bin for a query point is $i-1$, i and $i+1$.
- To avoid overfitting, as top scientists and engineers, we always prefer simpler models than complex ones.
- Type 1 and Type 2 errors. A type 1 error is the incorrect rejection of a true null hypothesis. It is a false positive. Usually a type 1 error leads one to conclude that a supposed effect or relationship exists *when in fact it doesn't*.



- $(1,1)$ True positive. $(1,0)$ False negative. $(0,1)$ False positive. $(0,0)$ True negative. The first element 1 means *you do have it*.
- False positive. A result indicating that a given condition is present when it actually it not.
- False negative. It's failing to assert what is present. It could also be regarded where a test result indicates that a condition failed, *while actually it was successful*.
- One word to describe False Negative? Miss Jessie.
- Precision: $\frac{TP}{TP+FP}$. Recall: $\frac{TP}{TP+FN}$.
- When we say that we are encouraging *overfitting*, we are saying the "generalization" of the models we build, sucks.
- Another explanation. Precision: $\frac{\sum True\ Positive}{\sum Test\ Outcome\ Positive}$ Recall: $\frac{\sum True\ Positive}{\sum Condition\ Positive}$
- Data Mining methods are always intrinsic probabilistic, why? One explanation given by Dr. Woon is that real world data *is always noisy*.
- **Top down inductive tree**. Select a feature to split on; Sort examples into subsets based on the values of feature, one for each *value*; Branch the tree by creating new nodes (aka, new subtrees) foe each subset; *recurse* until a complete tree is obtained.

- **Entropy**. In Information Theory, we define entropy as follows: $H(x) = -\sum_{i=1}^{n} p(x_i) \ln p(x_i)$
- Information Gain. In general terms, the expected information gain is the *change* in information entropy from **a prior state** to a state that takes some information as given: $IG(T, a) = H(T) - H(T|a)$.
- Gain Ratio. Normalize information gain with respect to *the number of values* that a feature can take.
- Gini impurity index. It's the probability that a randomly selected instance is wrongly classified based on a label randomly sampled from that subset. $I_G(f) = \sum_{1}^{m} f_i(1 - f_i)$.
- Post pruning. Each node to which *only leaves are attached* is considered for pruning. The idea is to evaluate combined (*weighted*) error rate and compare that of the father node. Core idea is that a compact tree with *good* prediction.
- Decision trees with continuous variables. Treat N values in the training set as N separate features, and choose the split point with highest *Information Gain* or any other criterion that's reasonable. Split between points w. different labels.
- In decision tree classification, there are usually two ways to avoid overfitting: limit tree depth; reduced error pruning.
- Decision Tree. Disadvantages: learning process is heuristic thus often results in overfitting; closeness to boundary, confidence intervals. . . are ignored.
- **K-means Clustering**. Generate k initial cluster centroids; Assign each point to the closest centroid; recompute the location of each centroid using existing class memberships; iterate untill convergence criterion is met.
- Dunn Index. $DI(c) = \min_{i,j \in c: i \neq j} \{ \frac{\delta(A_i, A_j)}{\max_{k \in c}(\triangle(A_k))} \}$. We want to see a larger DI when build clusters.
- Cluster Quality metrics. As with clustering algorithms, we want to see **compact** clusters and long **distances** between each cluster. For groups of the same interest, get as close as possible. And vice versa. Eg: $C = \frac{S - S_{\min}}{S_{\max} - S_{\min}}$. Smaller!