Classification Trees

- **Impurity:** at max. when observations are distributed evenly over all classes; at min. when all observations belong to a single class; Symmetric function of $p_1, \dots, p_n$
- ***π*** is the proportion of cases sent to left (l) and right (r) wrt to the total; ***i*** is a function of the relative frequencies of the different classes in the node wrt to the split
- **Gini-index** $i(t) = p(0|t)p(1|t) = p(0|t)\big(1 - p(0|t)\big)$; for multiclass $\sum_{j=1}^{C} p(j|t)\big(1 - p(j|t)\big) = 1 - \sum_{j=1}^{C} p(j|t)^2$
  $\Rightarrow$Gini index is stricly concave when 2nd der. is negative. Impurity with gini index:
- **Quality for a split** is the reduction of impurity it achieves: $\Delta i(s,t) = i(t) - \{\pi(l)i(l) + \pi(l)i(r)\}$
- **Entropy:** $i(t) = -p(0|t)\log p(0|t) - p(1|t)\log p(1|t)$; average amount of info. generated by drawing an example at random from this node and observing its class
- **Importance measure:** of Breiman et al. may overestimate the importance of variables with low similarity to the best split but with high impurity reduction
- **Splits:** split on border of segments ($\leq$), For a categorical variable with L distinct splits, there are $2^{L-1} - 1$ possible splits to consider
  $\Rightarrow$E.g: For a cat. var. (marital status) with 3 possible values there are: $2^{3-1} - 1 = 3$ splits **to consider**: {1}, {2}, {3} → because {2,3} = {1}
- **Pruning:** Stopping rules: don't expand a node if the impurity reduction is below some threshold
  $\Rightarrow$To prune a tree T in a node t means that t becomes a leaf node and all descendant are removed
  $\Rightarrow$Definitions: $T' \leq T$ means that T' is a pruned subtree of T; $|\tilde{T}|$ is the number of leaf nodes of the binary tree T; T_{t_n} is the branch of tree with root note in t_2

Total cost of a tree $C_\alpha(T) = R(T) + \alpha|\tilde{T}|$

- $R(T)$ is the **Resubstitution error** → $\frac{\text{\# of wrong classifications made by T}}{\text{number of examples in the training sample}}$
- $\alpha|\tilde{T}|$ is the penalty for the complexity of the tree
- Depending on α, different pruned subtrees will have the lowest total cost
- For $\alpha = 0$ the tree with smalles **Resubstitution wins,** for higher α a less complex tree that makes a few more errors could win
- The total cost of T and $T - T_t$ becomes equal when $C_\alpha(\{t\}) = C_\alpha(T_t)$ → solve for $\alpha = \frac{R(t) - R(T_t)}{|\tilde{T}_t| - 1} \rightarrow g(t) = \frac{R(t) - R(T_t)}{|\tilde{T}_t| - 1}$
- $g_i(t_k) = \frac{\text{increase in error due to pruning in t}}{\text{decrease in \# leaf nodes due to pruning in t}}$ used to rank Trees $T_1 > T_2 > \cdots > T_n$: to rank iterate on nodes and prune nodes with minimum g value
- E.g $g_1(t_3) = \frac{R(t) - R(T_1, t_3)}{|\widetilde{T_1, t_3}| - 1} = \frac{\left(Errors\ in\ t_3{:}\frac{10}{200}\right) - \left(errors\ in\ leaf\ nodes, rest\ assigned\ maj\ class{:}\frac{0}{200}\right)}{(\#leaf\ nodes\ under\ t_3{:}2) - 1} = \frac{1/20}{1} = \frac{1}{20}$
- When the costs are equal, the smaller tree is preferred
- Simple models tend to have high bias and low variance, Complex models the opposite
- As sample size increase, var. goes down but bias remains the same → we can fit more complex models if the data set is large → we have to find a trade-off between bias and variance in order to get small prediction error
- **Reducing variance:** By averaging, bootstrapping; Draw a sample with replacement; Grow a tree on this bootstrap sample; Repeat M times to grow M trees; Predict a test sample; Take majority voting

Graph

- If there is a superfluous variable in an independence, but the independence still holds without it, the independence still holds
- **Curse of dimensionality**: fewer data point than probabilities to estimate. For example the saturated model (dependent var) estimates cell probabilities by dividing cell count by total number of observations: estimation of $m^k - 1$ probabilities
- **Independence model** $\hat{P}(x,y) = \hat{P}(x)\hat{P}(y) = \frac{n(x)}{n}\frac{n(y)}{n} = \frac{n(x)n(y)}{n^2}$; Fitted count: $n\hat{P}(x,y) = \frac{n(x)n(y)}{n}$; estimation of $k(m-1)$ probabilities
- **Independence Rules**: conditional independence $x \perp y|z \rightarrow iff \rightarrow P(x,y|z) = P(x|z)P(y|z)$ or $P(X_a, X_b) = P(X_a)P(X_b|X_a)$
- **Rules of probability**: $P(X) = \sum_Y P(X,Y)$ summing all values of Y; $P(X,Y) = P(X)P(Y|X)$ marg. P of X * cond. P of Y; if X,Y independent then $P(X,Y) = P(X)P(Y)$ → $P(x|y) = P(x)\ and\ P(y|x) = p(y)$
- Relax constraints: X and Y independent iff $P(x|y) = g(x)h(y) = logP(x,y) = g^*(x)h^*(y) = logg(x) + logh(y)$
- **Factorisation criterion for independence**: $P(x,y) = g(x)h(y) = \frac{1}{c_1}P(x)\frac{1}{c_2}P(y) \rightarrow P(x) = g(x)\sum_y h(y)$
- **Markov properties**: **pairwise**: for all non-adjacent vertices *i,j* = $X_i \perp X_j|\ rest$; **global**, *a* separates *b* from *c*: $X_b \perp X_c|\ X_a$; **local** $X_i \perp rest\ |\ boundary$ → boundary is the set of adjacent vertices
  $\Rightarrow\hat{P}(care, survival|clinic) = \hat{P}(care|clinic)\hat{P}(survival|clinic)$; product law, multiply by $\hat{P}(clinic)$
  $\Rightarrow\hat{P}(care, survival, clinic) = \hat{P}(care, clinic)\hat{P}(survival|clinic)$→ divide by $\hat{P}(clinic)$
  $\Rightarrow\hat{P}(care, survival, clinic) = \frac{\hat{P}(care,clinic)\hat{P}(survival,clinic)}{\hat{P}(clinic)}$→ in numerator we have distribution over cliques, in denominator over a subset of a clique; get fitted count by writing $\hat{n}\ for\ n\hat{P}$ + jump
  $\Rightarrow\hat{n}(care, survival, clinic) = \frac{\hat{n}(care,clinic)\hat{n}(survival,clinic)}{\hat{n}(clinic)}$ = **fitted count** as ML satisfies margin constraints $\rightarrow \hat{n} = n$
  $\Rightarrow e.g.\ \hat{n}(clinic1, more, yes) = \frac{n(clinic1,more)n(clinic1,yes)}{n(total\ clinic\ \mathbf{1})}$; create table in function of **Z**, create 2 tables collapsing the independent variables; calculate fitted values
- **CPR** (degree of association) > 1 pos. association, < 1 neg. association, =1 independent! $cpr(care, survival) = \frac{n(less,no)n(more,yes)}{n(less,yes)n(more,no)}$
- **Confounder term:** has to be controlled for – distorts effect of other variables → $X \perp Y\ does\ not\ imply\ X \perp Y\ |Z$
- **Bernoulli random variable** has probability p: $P(x) = p^x(1-p)^{1-x}$ → log linear expansion by writing the 2X2 table in log and then reparametrizing $logP(x_1, x_2) = u_0 + u_1x_1 + u_2x_2 + u_{12}x_1x_2 \rightarrow u_{12} = \log\left(\frac{p(1,1)p(0,0)}{p(0,1)p(1,1)}\right) = \log(cpr(X_1, X_2)$
- For 2 variable (classification problems) $X_1 \perp X_2 \Leftrightarrow u_{12} = 0$ because if CPR=1 then logCPR=0! If $u_{12} = 0$ then $logP(x_1, x_2) = u_0 + u_1x_1 + u_2x_2\ g(x) = u_0 + u_1x_1; h(x_2) = u_2x_2$
- For 3 variables: $X_2 \perp X_3|X_1 \Leftrightarrow u_{23} = 0\ and\ u_{123} = 0$ because there cannot be terms with $X_2\ and\ X_3$ in common
- When we have an arbitrary number of variables then the *u* terms become functions of x rather than just constant: logP(x) $u_0 + u(x)$
- $X_b \perp X_c|X_a$ iff all u-terms in the log linear expansion with coordinates in both b and c are zero. E.g. $X_4 \perp (X_2, X_5)|(X_1, X_3)$ holds iff there are functions g and h such that: $logP(x_1, \dots, x_5) = g(x_1, x_3, x_4) + h(x_1, x_2, x_3, x_5)$ → iff u-terms with (4) and (2,5) = 0
- **A hierarchical model** is identified by listing its highest order interaction terms
  Graphical log-linear model: all its constraints can be read from the independence graph; is a hierarchical model in which the highest order interaction terms correspond the **cliques:** subset of vertices of graph such that every vertex is connected - in the graph → it gives a full characterization of the model [1, 13-16]
- **Margin constraints:** $\hat{n}(X_1, X_n\ n = nodes\ in\ clique) = n(X_1, X_n\ n = nodes\ in\ clique)$ per each clique
- **ML estimator**: returns estimates of the cell probabilities that maximize the probability of the observed data, subject to constraint that the conditional independencies expressed in the graph are satisfied by the estimates: $\widehat{n_a}^M = N\widehat{P_a^M} = n_a$
- **Maximum likelihood estimation of hierarchical and graphical models**: Example with 2 binary variables {0,1}
  Usually use log-likelihood $argmax_\theta \sum_{i=1}^{n} logP(x_i;\theta) \rightarrow$ independence model has a log-linear representation as $\log P(x_1, x_2) = u_0 + u_1x_1 + u_2x_2$ –> yield log-likelihood function applying to data available and solve it putting as constraint that cell probabilities must sum to one. Take derivative wrt Lagrange multiplier and equate to zero → solve the set of equation and find $\hat{p}_1, \hat{p}_2$, complete the table with the fitted probabilities and obtain the fitted *u terms* from the fitted probabilities
- **Decomposable graphical Models**: have no chordless cycle – chordless only if successive pairs of vertices are connected by an edge - of length greater than three (no shortcut).
- **RIP**: E.g. {AC, BC, CDE, DEF} have {0,C,C,DE} as separators → ordering is RIP → $\widehat{\mathrm{n}(ABCDEF)} = \frac{n(AC)n(BC)n(CDE)n(DEF)}{n(C^2)n(DE)}$
- **Likelihood score of model** M is: $\prod_x \hat{P}^M(x)^{n(x)}$ (*product over all cells in the table – P of observed data using the fitted cell P according to model M*). Log-likelihood score is $\sum_x n(x)\quad log\hat{P}^M(x) \rightarrow for\ saturated{:}\ \sum_x n(x)\quad log\frac{n(x)}{N}$
- **Log linear expansion**: for 2X2 table = $logP(x_1, x_2) = u_0 + u_1x_1 + u_2x_2 + u_{12}x_1x_2$, independence model excludes all u-terms with interactions between x_1x_2
- **Model deviance** compares the log-likelihood of the fitted model with the log-likelihood of the saturated model: $2\sum_{cells} observed\ count * \log\frac{observed}{fitted}$; reject null hypothesis that the independence model is the true model when the observed *deviance* > χ_v^2, v = df = the **difference** of the # of restrictions (**count terms**: r=rows, c=col → $(r-1)*(c-1) \rightarrow L0 - L1$; how many u-terms are put to zero in case of independence) of the two models (M_0vs. M_1)

- $\boldsymbol{AIC(M)} = dev(M) + 2\dim(M)$ where dim(M) is the number of parameters of the model
- **IPF**: sufficient statistics are row totals $n_1(x_1)$ and column totals $n_2(x_2)$. 1°: Begin with table with uniform counts
  2°: fit to row margins; e.g. $\hat{n}_1(0,0)^1 = \hat{n}_1(x_1,x_2)^1 = n_1(x_1)\frac{(\hat{n}(x_1,x_2)^0)}{\hat{n}_1(x_1)^0}$ → *(tot of row* x_1*)** $\frac{count\ of\ cell\ unif.count\ = 0{,}0}{total\ row\ unif.count\ x_1=0}$
  3°: fit to column margins; e.g. $\hat{n}_1(0,0)^2 = \hat{n}_2(x_1,x_2)^2 = n_2(x_2)\frac{(\hat{n}(x_1,x_2)^1)}{\hat{n}_2(x_2)^1}$ → *(tot of col* x_2*)* * $\frac{count\ cell\ updated\ unif.count\ = 0{,}0}{total\ col\ updated\ \ unif.count\ x_2=0}$
- With *k* labelled nodes there are $\frac{k}{2}$ undirected graphs, edge can be excluded or included →$2^{\frac{k}{2}}$
- **Hill climbing search**: how many **neighbouring graphical models**: count edges, count new edges you can create on top of old ones, sum. **Decomposable:** count edges that can be taken away and that can be added without creating cordless cycles.

Sets/ Pattern mining
- **Confidence**: Conditional probability [P(Y|X)]; **Support** = relative # of X buying all items occurring in the rule [P(X)]
- **Association rule:** for association $(X \to Y)$ support is: $s(X \cup Y)$; confidence is:$\frac{s(X\cup Y)}{s(X)}$
- **Apriori property:** count patter only once per item
- **GSP algorithm:** level-wise search – no double count; don' extend infrequent sequences; candidate generated iff all its subsequences are frequent [GA+GA can be GAA]
- **Lift:** $lift(A \to C) = \frac{P(C|A)}{P(C)} = \frac{P(A,C)}{P(A)P(C)} = \frac{s(XY)*|db|}{s(X)*s(Y)}$; if *lift > 1* then rule is "interesting".
- **Frequent set mining**: count frequency, eliminate elements which do not have support, count on all possible candidate → until no more support
- **I is Maximal frequent** iff *I* is frequent and no proper superset (set that contains it – no matter the position within the sequence) of I is frequent
- $\sigma(I)$set of tuples that contain all items in I
- $f(T)$ set of items included in all transaction (which are "common elements")in T
- $c(I) = f(\sigma(I))$. Itemset is closed iff $c(I) = I \to c(\{A,B\}) = \{A,B\}$
- If $X \subset Y, s(X) = s(Y) : c(X) = c(Y)\ because \to \sigma(Y) \subseteq \sigma(X) \to s(X) = s(Y) : \sigma(Y) = \sigma(X) \to c(X) = f(\sigma(X)) = f(\sigma(Y)) = c(Y)$
- If $c(X) = Z : s(X) = s(Z) because \to z\ is\ closed\ and\ \sigma(X) = \sigma(Z)$
- Itemset I is a generator of a closed itemset J if there is a minimal itemset with c(I)=J

Closures
- **Compute closed frequent item set:** determine generators (→ A-priori pruning + prune if itemset has subset with same support); select all generators not pruned and determine their closure (→ a superset of the generator that has same support as the generator's – if there is no closure take generator itself)
- **One-to-one:** if 1-to-1, same label and order of mapping is preserved
- **Matching functions:** denoting nodes of d by v and nodes of T by w $\phi(w_1) = v_1, \dots \phi(w_n) = v_n$. Requirements: $Labeling\ preserved = L(w_i) = L(v_i)$; Ancestor-descendant: $w_i \epsilon \pi^*(w_{i+1})\ and\ v_i \epsilon \pi^*(v_{i+1})$; ordering $w_i < w_{i+1}\ and\ v_i < v_{i+1}$; e.g. AI in MAIN = $\phi(1) = 2;\ \phi(2) = 3$
- **Induced subtree =** if it is one-to-one, label preserved, left-to-right and parent-child
- **Embedded subtree=**if it has label preserved, left-to-right, ancestor-descendant (stricter than what you think)
- **Anti-monotonicity property:** D=db of trees; $T_1 \preccurlyeq T_2$; therefore support$(T_1, D) \geq$ support(T_2, D) because $\forall d \epsilon D : T_2 \preccurlyeq d \to T_1 \preccurlyeq d$; if subsequence relation is transitive then the relation is anti-monotone wrt to support (anti-monotonicity property holds).
- **Subsequence:** $T_1 is\ subsequece\ of\ T_2$ if exists a mapping $\phi: [1,k] \to [1,m]$ such that$X_i \subseteq Y_{\phi(i)}\ and\ i < j \to \phi(i) < \phi(j)$ now assume support $S^1 \preccurlyeq S^2$ and $S^2 \preccurlyeq S^3$ $S_1 \subseteq S^3_{\phi(i)}\ and\ i < j \to \phi_{13}(i) < \phi_{13}(j)$
- **Right-most occurrence (RMO)** list: list of nodes in the data tree to which the nodes in pattern tree can be mapped. XY graph: Y=candidate; X=# nodes

Bayesian Networks
- **Definitions:** $i \to j$ then $X_i$is parent of $X_j$and coordinates of parents of $X_j = pa(j)$, *i* is ancestor of *j.* Ancestors = all nodes above *(i);* descendents = all nodes below *(i);* $G \perp R$ are **marginally independent** not conditionally given the response
- **Construction of DAG**: all variables must be labelled and ordered (temporal or causal); then draw arrow for all the nodes that do not appear in the factorized joint distribution of that note. Joint distribution of $X_1, \dots, X_k$is $P(X) = P(X_1)P(X_2|X_1) \dots P(X_k|X_{k-1}, X_{k-2}, \dots, X_1)$ [also factorisation of graph]. Draw arrow unless $j \perp i|rest$
- **Joint density:** $P(X_1, \dots, X_k) = \Pi^{k}_{i=1} P(X_i|X_{pa(i)})$
- **Independence properties** - **D-separation:** path p is blocked by set Z iff: ***1.)*** P contains a chain of nodes $A \to B \to C\ or\ a\ fork\ A \leftarrow B \to C$ such that B is in Z; ***2)*** p contains a collider $A \to B \leftarrow C$ such that neither B nor any of its descendants are NOT in Z; ***3)*** if Z blocks every path between two nodes, they are independent given Z
- **Independence properties -** you get a **Moral graph/moralization** when marry parents and delete directions of the graph edges. Factorization of moral graphsP(X) $= \Pi^{k}_{i=1} g_i(X_i|X_{pa(i)})$
- **Independence properties:** to determine if $X_1, X_2$are independent we have to look at the "smallest marginal distribution that includes both"
- **Check conditional independencies:** to verify $X \perp Y|Z$ always take directed independence graph on ancestors of $X \cup Y \cup Z$ and moralize the graph
- **Maximum likelihood estimation:** collection of independent multinomial estimation problems: "find value of unknown parameters that maximize the probability of the observed data". Take log and derivative wrt to p(1) of $L = p(1)^{n(1)}(-p(1))^{n-n(1)}$ → $p_i\binom{0}{1} = \frac{n\binom{0}{1}}{\boldsymbol{n}}$, where $i = 1 for\ X_1, i = n\ for\ X_n$
- **BN-DAG ML estimation**: with joint distribution factorization of $P(X) = \Pi^{k}_{i=1} p(X_i|X_{pa(i)})$ → $\hat{p}(x_i|x_{pa(i)} = \frac{n(x_i, x_{pa(i)})}{n(x_{pa(i)}}$ which is $\frac{\#of\ records\ in\ data\ with\ X_i=x_i\ and\ X_{pa}=x_{pa}}{\#\ of\ records\ in\ data\ with\ X_{pa}=x_{pa}}$
  Pay attention in the calculation to when variables are dependent! Directed acyclic graphs
  ML estimation can be smoothed adding "prior counts" $\hat{p}(x_i|x_{pa(i)}) = \frac{n(x_{pa(i)})\hat{p}(x_i|x_{pa}(i)) + m(x_{pa(i)})p^0(x_i|x_{pa(i)})}{n(x_{pa(i)}) + m(x_{pa(i)})}$, where *m* is prior precision,$p^0$is prior estimate of *p*
- **Log-likelihood score:** the higher the better the fit on data, saturated model has highest; $L = \sum^k \sum_{x_{i,x_{pa(i)}}} n(x_i, x_{pa(i)})\ log \frac{n(x_i, x_{pa(i)})}{n(x_{pa(i)})}$; recompute only changes in model
- **Scoring functions:** $\boldsymbol{AIC}(M) = L^M - \dim(M)\,; BIC(M) = L^M - \frac{logn}{2}\dim(M)$; BIC highest penalty → less complex models preferred
  Count parameters: node has *k* different parents ***configurations*** (0,1;0,0;1,0…) and take *m* values, # param. = $k(m-1)$ if k=0 then # param. = $m-1$
  If asked to add a parameter, add only if variables are not independent, only in that case recompute L-score and then BIC-score
- **To score model**: Start node 1,..n,; when dependency: (1,1)-(1,2)-(2,1) …and build using $\#\ of\ records\ in\ current\ node\ *\ ln\ \frac{\#\ of\ records\ in\ current\ node\ equal\ to\ (1,\mathbf{1})[or\ (1,\mathbf{2})\dots]}{total\ \#\ of\ records\ that\ match}$
- **# of parameters of Bayesian network:** $\sum^k\{(d_i - 1)\Pi_{j\epsilon pa(i)} d_j\}$; where *k* # var in network; d_i#of possible values of X_i; $\Pi_{j\epsilon pa(i)} d_j$ # parent configuration for X_i (1 if null)
- **Markov equivalence**: two DAGs are Markov equivalent (same score) iff have the same undirected graph when you drop the direction of all edges and same v-structures
- **Essential graph**: edge becomes bi-directional in the essential graph if there is equivalent DAG in which direction of edges is reversed
- **V-structure:** encodes the independence between two nodes
- **Bayesian network:** we pick $add(A \to B), old\ score\ is: n_{ab}(0,0)\frac{n_{ab(0,0)}}{n_a(0)} + n_{ab}(0,1)\frac{n_{ab(0,1)}}{n_a(0)} + n_{ab}(1,0)\frac{n_{ab(1,0)}}{n_a(1)} + n_{ab}(1,1)\frac{n_{ab(1,1)}}{n_a(1)}$
  add interaction term with B (i.e. $C \to B$): $new\ score: n_{abc}(0,0,0)\frac{n_{abc(0,0,0)}}{n_{ac}(0,0)} + n_{abc}(1,0,0)\frac{n_{abc(1,0,0)}}{n_{ac}(1,0)} \dots$
- **Logistic regression** $\hat{P}(Y = 1|X) = \frac{e^{\beta Tx}}{1+e^{\beta Tx}}$; LN $\{\frac{\hat{P}(Y = 1|X)}{\hat{P}(Y = 0|X)}\}$ the odds, if > 1 class 1; same if $\beta^T x > 0$ class 1
- **Missing terms**: if we have three binary variables $X = (X_1, X_2, X_3)$ and we observe (1,0,?); $P(1,0,?) = P(1,0,0) + P(1,0,1)$ in BN $= p_1(1)p_2(0)p_{3|12}(0|1,0) + p_1(0)p_2(1)p_{3|12}(0|0,1) = p_1(1)p_2(0)\ \ as\ p_{3|12}(0|0,1) + p_{3|12}(0|1,0) =$
- **Naïve Bayes:** compute class priors**:** $\hat{P}(c) = \frac{N_c}{N_{doc}}$:# doc in each class; word probability $\hat{P}(w_i|c) = \frac{count(w_i,c)+1}{\sum w_i \epsilon V\ (count(w_j,c)+|V|}$ →$\hat{P}("no"|+) = \frac{count("no"|+)+1}{count(words\ in\ +)+tot.\#individ\ words}$