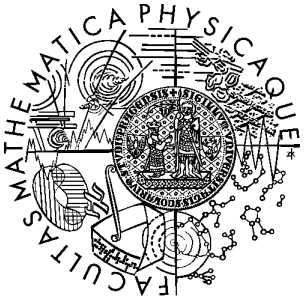
Univerzita Karlova v Praze
Matematicko-fyzikální fakulta

BAKALÁŘSKÁ PRÁCE



Martin Švantner

Klasifikační a regresní stromy

Katedra pravděpodobnosti a matematické statistiky

Vedoucí bakalářské práce: Mgr. Jaroslav Ševčík
Studijní program: Obecná matematika

2008

Na tomto mieste by som rád poďakoval môjmu vedúcemu bakalárskej práce Mgr. Jaroslavovi Ševčíkovi za pomoc a cenné rady, ktorými mi pomohol pri vypracovaní práce.

Prehlasujem, že som svoju bakalársku prácu napísal samostatne a výhradne s použitím citovaných prameňov. Súhlasím so zapožičaním práce a jej zverejňovaním.

V Prahe dňa 28.5.2008 Martin Švantner

Obsah

Název práce: Klasifikační a regresní stromy
Autor: Martin Švantner
Katedra (ústav): Katedra pravděpodobnosti a matematické statistiky
Vedoucí bakalářské práce: Mgr. Jaroslav Ševčík
e-mail vedoucího: sevcik@karlin.mff.cuni.cz

Abstrakt: V predloženej práci študujeme klasifikačné a regresné stromy, ktoré predstavujú alternatívny prístup k riešeniu klasifikačných a regresných úloh. Táto neparametrická štatistická metóda produkuje modely v tvare stromu, ktoré sú jednoducho pochopiteľné a interpretovateľné. Po uvedení do problematiky klasifikácie sa venujeme klasifikačným stromom a spôsobu ich konštrukcie. Proces konštrukcie následne prispôsobíme za účelom konštrukcie regresných stromov. Práca obsahuje aj detailný popis algoritmu *CART*, ktorý vytvára binárne klasifikačné a regresné stromy. V závere ilustrujeme uvedenú teóriu na niekoľkých príkladoch riešených v štatistickom prostredí *R*.

Kľúčové slová: klasifikácia, klasifikačné a regresné stromy, algoritmus *CART*.

Title: Classification and regression trees
Author: Martin Švantner
Department: Department of probability and mathematical statistics
Supervisor: Mgr. Jaroslav Ševčík
Supervisor's e-mail address: sevcik@karlin.mff.cuni.cz

Abstract: In the present work we study classification and regression trees that represent alternative approach to classification and regression problems. This nonparametric statistical method produces tree-structured models that are simple to understand and interpret. After an introduction to the classification we aim our focus on classification trees and their construction process. Consecutively we transform this process to the process of regression trees construction. The work also contains detailed description of CART algorithm that constructs binary classification and regression trees. In the last part we illustrate theory on some examples solved in statistical environment *R*.

Keywords: classification, classification and regression trees, *CART*.

Úvod

Rozšírenie výpočtovej techniky spôsobilo, že štatistici sa v súčasnosti často stretávajú s dátami, pre ktoré je okrem iného charakteristická vysoká dimenzionalita, zmes dátových typov a nehomogénnosť. Vzhľadom k tomu, že mnoho štatistických metód nebolo navrhnutých pre takéto súbory, je ich použitie obtiažne a často nevyhovujúce. Preto sa začali objavovať nové metódy, priamo navrhnuté pre dáta tohto typu.

V posledných rokoch sa stali populárne klasifikačné a regresné stromy, ktoré predstavujú metódu na riešenie klasifikačných a regresných úloh. Popularitu si získali hlavne kvôli svojej jednoduchosti a zrozumiteľnosti. Tento prístup navyše kladie len minimálne požiadavky na dáta a okrem samotnej klasifikácie resp. predikcie poskytuje aj náhľad do predpovednej štruktúry.

Cieľom tejto práce je predstaviť klasifikačné a regresné stromy a podrobnejšie popísať jeden konkrétny algoritmus na ich konštrukciu. Kapitola 1 predstavuje uvedenie do problematiky klasifikácie a zahrňuje aj stručný prehľad klasifikačných metód. V kapitole 2 sa zoznámime s pojmom klasifikačný strom a s metodológiou konštrukcie klasifikačných stromov. Regresné stromy a spôsob ich konštrukcie sú súčasťou kapitoly 3. V kapitole 4 popíšeme algoritmus *CART*, ktorý konštruuje binárne klasifikačné a regresné stromy. Náplňou kapitoly 5 budú potom príklady riešené pomocou štatistického programu *R*.

Kapitola 1

Klasifikácia

V praxi sa často stretávame s nasledujúcim typom úloh. K dispozícii máme dáta, ktoré obsahujú výsledky meraní na danom objekte. Na základe týchto dát potom chceme priradiť objektu príslušnosť k jednej skupine. Dané úlohy nazývame klasifikačné úlohy. Na riešenie tohto druhu úloh bolo navrhnutých množstvo postupov a metód. Väčšina z nich produkuje určité pravidlo, podľa ktorého sa klasifikácia prevádza. Navyše je v praxi často žiadúce, aby zvolená metóda okrem samotnej klasifikácie dávala aj predstavu o tom, ktoré vlastnosti objektu najviac vplývajú na jeho zaradenie do skupiny a bola čo najviac zrozumiteľná.

1.1 Klasifikačné metódy

Predtým ako sa budeme venovať hlavnému tématu tejto práce, stručne popíšeme najčastejšie používané klasifikačné metódy.

- *Zhluková analýza:* slúži na zoskupovanie objektov do skupín na základe ich podobnosti a rozdielnosti. K tomu je potrebné definovať miery podobnosti a nepodobnosti. Metódy zhlukovej analýzy delíme podľa spôsobu zhlukovania na hierarchistické (napr. metóda najbližších susedov, mediánová metóda, ...) a nehierarchistické (metóda k-means, fuzzy clustering, ...).

- *Regresné metódy* zamerené na kategoriálnu vysvetľovanú veličinu: patrí sem logistická regresia, v prípade binárnej veličiny, multinomická logistická regresia pre nebinárnu veličinu. Klasifikácia prebieha na základe získaného regresného modelu.

- *Neurónové siete*: predstavujú modely, zostavené na základe abstrakcie vlastností biologických nervových systémov. Neurónová sieť sa skladá z neurónov, ktoré sú poprepájané prepojeniami. Pritom na prepojeniach sú váhy, ktoré uchovávajú určitý parameter, podľa veľkosti ktorého potom modifikujú prechádzajúci signál. Váhy priradené jednotlivým prepojeniam sa získavajú počas fázy učenia. Neuróny predstavujú výpočtové jednotky, ktoré zlučujú informácie z prepojení, ktoré do nich ústia, a tak vzniknutú informáciu posúvajú ďalej. Problémom práce s neurónovými sieťami je v porozumení a interpretácii modelu.

- *Klasifikačné stromy:* metóda produkuje klasifikátor vo forme stromu. Dáta, ktoré chceme klasifikovať, na základe použitých kritérií postupne prechádzajú stromom, až sa dostanú do listu stromu, ktorý priradí klasifikovanému objektu triedu. Podrobnejšie ich popíšeme neskôr v tejto práci.

- *Diskriminačná analýza:* metóda, ktorá klasifikuje prípady do predom definovaných tried. Na základe prípadov so známymi triedami sa vytvorí sústava diskriminačných rovníc, v ktorých sú vysvetľujúcim premenným priradené určité váhy. Na základe týchto rovníc potom prebieha samotná klasifikácia.

- *Naivný bayes:* pre každý dvojicu, premenná a jej hodnota, sa počíta pravdepodobnosť príslušnosti ku každej skupine a to tak, že sa delí počet výskytov dvojice v danej skupine, počtom výskytov v celom súbore. Pri klasifikácii nového objektu potom počítame pravdepodobnosti príslušnosti ku každej kategórii. Tie získame vynásobením pravdepodobností príslušnosti pre dvojice, ktoré objekt obsahuje.

- a ďalšie.

1.2 Klasifikátor

Pre precíznejšiu formuláciu úlohy z úvody kapitoly predpokladajme nasledujúce. Na jednotlivých objektoch sledujeme M znakov, a to vždy v rovnakom poradí. Nech náhodná veličina X_i, $i = 1, \ldots, M$, ktorá popisuje znak i, môže byť spojitá alebo kategoriálna veličina. Ďalej predpokladajme, že X_i môže nadobúdať hodnoty z priestoru $\mathcal{X}_i \subseteq \mathbb{R}$. Položme $\boldsymbol{X} = (X_1, \ldots, X_M)'$, potom všetky možné hodnoty $\boldsymbol{X}$ padnú do priestoru $\mathcal{X} = \mathcal{X}_1 \times \ldots \times \mathcal{X}_M \subseteq \mathbb{R}^M$. Ďalej

uvažujme náhodnú veličinu Y, ktorej hodnoty budú reprezentovať príslušnosť objektu k skupine. Predpokladáme, že každý objekt môže byť zaradený práve do jednej z J tried. Veličina Y bude teda kategoriálna a nadobúda hodnoty iba z množiny tried $\mathcal{C} = \{1, \ldots, J\}$. Vektor veličín $\boldsymbol{X}' = (X_1, \ldots, X_M)$ budeme nazývať *vektor vysvetľujúcich veličín*, alebo vektor prediktorov a veličinu Y *vysvetľovaná veličina*.

Naším cieľom bude nájsť *klasifikátor* (klasifikačné pravidlo) pomocou, ktorého bude možné systematicky predpovedať príslušnosť objektu k jednej z J tried.

Definícia 1.1. *Klasifikátor je funkcia* $d(\boldsymbol{x}) : \mathcal{X} \mapsto \mathcal{C}$ *taká, že pre každý vektor* $\boldsymbol{x}$ *je* $d(\boldsymbol{x})$ *rovná jednému z čísel* $1, \ldots, J$.

Je zrejmé, že zostrojenie klasifikátora $d(\boldsymbol{x})$ je ekvivalentné nájdeniu takého delenia priestoru $\mathcal{X}$ na J disjunktných podmnožín $\mathcal{A}_1, \ldots, \mathcal{A}_j$, $\mathcal{X} = \cup_{j \in \mathcal{C}} \mathcal{A}_j$, že pre každé $\boldsymbol{x} \in \mathcal{A}_j$ je predpovedaná trieda j.

Pri konštrukcii klasifikátoru sledujeme dva ciele:

1. Nájsť čo najpresnejší klasifikátor.
2. Odkryť predpovednú štruktúru úlohy.

Aby sme mohli klasifikačnú úlohu riešiť, je nutné mať k dispozícii dáta, na základe ktorých zostrojíme klasifikátor. Tieto dáta tvoria takzvaný *učebný súbor*. Ten obsahuje merania na N objektoch spolu s hodnotou vysvetľovanej premennej. Učebný súbor dát značíme $\mathcal{L}$ a možno ho zapísať ako množinu dvojíc $\mathcal{L} = \{(\boldsymbol{x}'_1, y_1)', \ldots, (\boldsymbol{x}'_n, y_n)'\}$, kde $\boldsymbol{x}_i \in \mathcal{X}$ a $y_i \in Y$, $i = 1, \ldots, N$. Klasifikátor teda vytvárame na základe učebného súboru dát. To so sebou ale prináša riziko, že nastane jav, ktorý nazývame *preučenie*. Jedná sa o stav keď zostrojené pravidlo nezachytáva skutočné závislosti medzi premennými, ale v podstate sa len naučí použitý súbor dát naspamäť. Pravidlo potom síce dáva dobré výsledky pre dáta použité pri konštrukcii, ale zlyháva pri aplikácii na iné dáta.

1.3 Chyba klasifikácie

Kvalitu (presnosť) klasifikátoru určuje *chyba klasifikácie*, ktorú budeme značiť $R^*(d)$. Aby sme ju mohli definovať budeme potrebovať zaviesť pravdepodobnostný model. Nech $P(A, j)$ je pravdepodobnosť na $\mathcal{X} \times \mathcal{C}$, $A \subset \mathcal{X}$,

$j \in \mathcal{C}$. $P(A, j)$ možno interpretovať tak, že náhodne vytiahnutý prvok má pravdepodobnosť $P(A, j)$, že pre jeho vektor $\boldsymbol{x}$ platí $\boldsymbol{x} \in A$ a jeho trieda je j. Ďalej predpokladáme, že súbor $\mathcal{L}$ tvorí N prvkov $(\boldsymbol{x}'_1, y_1)', \ldots, (\boldsymbol{x}'_N, y_N)'$ náhodne vybraných z rozdelenia $P(A, j)$.

Definícia 1.2. *Nech $(\boldsymbol{X}', Y)'$, kde $\boldsymbol{X} \in \mathcal{X}$ a $Y \in \mathcal{C}$, je náhodný vektor z pravdepodobnostného rozdelenia $P(A, j)$, t.j.*

1. $P(\boldsymbol{X} \in A, Y = j) = P(A, j)$
2. *$(\boldsymbol{X}, Y)$ je nezávislé na $\mathcal{L}$*

Potom definujeme chybu klasifikácie ako

$$R^*(d) = P\big(d(\boldsymbol{X}) \neq Y\big).$$

Optimálne by bolo zostrojiť klasifikátor na základe učebného súboru $\mathcal{L}$ a jeho presnosť určiť pomocou iného súboru prípadov rovnakého druhu ako obsahuje $\mathcal{L}$. V reálnych situáciách ale máme k dispozícii len dáta z učebného súboru. Pomocou nich potom musíme aj zostrojiť klasifikátor aj odhadnúť jeho chybu klasifikácie. Takéto odhady nazývame *vnútorné odhady* chyby klasifikácie. Pri ich definovaní budeme používať indikátorovú funkciu I. Poznáme tri základné typy vnútorných odhadov:

Resubstitučný odhad

Najmenej presný je takzvaný *resubstitučný odhad* $R(d)$, ktorý určíme nasledovne. Pomocou dát z $\mathcal{L}$ zostrojíme klasifikátor $d(\boldsymbol{x})$. Potom postupne dosadíme všetky dáta do klasifikátoru. Podiel všetkých prípadov a nesprávne klasifikovaných prípadov predstavuje požadovaný odhad. Platí teda

$$R(d) = \frac{1}{N} \sum_{n=1}^{N} I\big(d(\boldsymbol{x}_n) \neq j_n\big).$$

Nevýhoda tohto odhadu spočíva v tom, že používa rovnaké dáta pri konštrukcii $d(\boldsymbol{x})$ ako aj pri odhadovaní jeho presnosti, čo vedie k už spomínanému problému preučenia klasifikátoru. Odhad nám síce ukáže len malú chybu klasifikácie, ale pri aplikácii na iný súbor je táto chyba často výrazne vyššia.

Testový odhad

Druhý spôsob odhadu kvality klasifikátoru je *testový odhad*. V tomto prípade je učebný súbor rozdelený na dve časti $\mathcal{L}_1$ a $\mathcal{L}_2$. Pomocou $\mathcal{L}_1$ zostrojíme klasifikátor a dáta z $\mathcal{L}_2$ použijeme na určenie jeho presnosti. Odhad značíme $R^{ts}(d)$ a zapíšeme ho v tvare

$$R^{ts}(d) = \frac{1}{N} \sum_{(\boldsymbol{x}'_n, y_n)' \in \mathcal{L}_2} I\big(d(\boldsymbol{x}_n) \neq j_n\big).$$

Nevýhoda daného postupu je, že redukuje počet dát, ktoré môžeme použiť na zostrojenie klasifikátoru. Pre veľký učebný súbor je tento nedostatok zanedbateľný.

Odhad V-násobným krížovým overovaním

Nedostatky oboch predchádzajúcich postupov odstraňuje *odhad V-násobným krížovým overovaním*. Učebný súbor je rozdelený na V podmnožín $\mathcal{L}_1, \ldots, \mathcal{L}_V$, ak možno rovnakej veľkosti. Pre $v = 1, \ldots, V$ zostrojme pomocou dát z množín $\mathcal{L} - \mathcal{L}_v$ klasifikátory $d^v(\boldsymbol{x})$. Presnosť týchto klasifikátorov určíme pomocou testového odhadu. Teda

$$R^{ts}(d^v) = \frac{1}{N_v} \sum_{(\boldsymbol{x}'_n, y_n)' \in \mathcal{L}_v} I\big(d^v(\boldsymbol{x}_n) \neq j_n\big),$$

kde N_v sú veľkosti množín $\mathcal{L}_v$. Nakoniec použitím celého učebného súboru zostrojíme klasifikátor $d(\boldsymbol{x})$. Odhad V-násobným krížovým overovaním $R^{cv}(d)$ klasifikátoru $d(\boldsymbol{x})$ definujeme ako

$$R^{cv}(d) = \frac{1}{V} \sum_{v=1}^{V} R^{ts}(d^v).$$

Kapitola 2

Klasifikačné stromy

2.1 Základné pojmy a definície

Ako napovedá samotný názov kapitoly, v nasledujúcej časti sa budeme zaoberať klasifikátormi, ktoré budú mať štruktúru stromu. Preto je úvodná časť tejto kapitoly venovaná terminológii týkajúcej sa stromov, ktorú sme čerpali hlavne z [4].

Definícia 2.1. *Strom je neprázdna konečná množina T prirodzených čísel a funkcia sons, ktorá každému $t \in T$ priradzuje podmnožinu T. Funkcia sons musí spĺňať nasledujúce požiadavky:*

1. $\forall t \in T$ *platí buď* $sons(t) = \emptyset$, *alebo* $|sons(t)| \geq 2$,
2. $\forall t \in T$, $s \in sons(t) \Rightarrow s > t$,
3. $\forall s \in T$, $s \neq min(T)$ *existuje práve jedno* $t \in T$ *také, že* $s \in sons(t)$.

V súvislosti so stromami budeme používať nasledujúcu terminológiu:

- Každé $t \in T$ nazývame *uzol* stromu.
- Najmenší prvok T sa nazýva *koreň* stromu a značíme ho $root(T)$.
- *Koncovým uzlom* alebo *listom* nazveme také $t \in T$, pre ktoré $sons(t) = \emptyset$. Množinu uzlov stromu T budeme značiť $\widetilde{T}$.
- Ak $s \in sons(t)$, potom s nazývame *synom* uzlu t a uzol t *otcom* uzlu s.

- Vzhľadom k bodu 3 z definície 2.1 možno zaviesť funkciu *father(t)*, ktorá každému $t \in T - \{root(s)\}$ priradí jeho otca.
- Ak možno $t \in T$ zapísať ako $t = father(s)$, alebo $t = father(father(s))$, alebo ..., nazýva sa t predchodcom s a s následovníkom t.

Definícia 2.2. *Nech $T_1 \subset T$ je neprázdna. Definujme funkciu $sons_1(t) = sons(t) \cap T_1$, $\forall t \in T_1$. Ak T_1 a $sons_1(t)$ vyhovujú definícii stromu, povieme, že T_1 je podstrom stromu T.*

- T_1 budeme nazývať *prerezaným podstromom* T (značíme $T_1 \preccurlyeq T$, prípadne $T_1 \prec T$ ak platí $T_1 \neq T$), ak je T_1 podstromom T a navyše platí $root(T) = root(T_1)$.
- *Vetvou stromu* T vychádzajúcou z $t \in T$ (značíme T_t) rozumieme podstrom stromu T, ktorý obsahuje t ako koreň a obsahuje všetkých nasledovníkov t.
- *Prerezaním stromu* T rozumieme vytvorenie stromu $T_1 \preccurlyeq T$ pomocou odrezania niektorých vetví.
- *Odrezanie vetvy* T_t predstavuje odstránenie všetkých nasledovníkov uzlu t, ktorý sa potom stáva koncovým uzlom stromu $T - T_t$

Definícia 2.3. *Klasifikačný strom je strom T, v ktorom každému uzlu $t \in T$ priradíme podmnožinu $\mathcal{A}(t) \subset \mathcal{X}$ a triedu $j(t) \in \mathcal{C}$. Navyše požadujeme:*

1. *$\{\mathcal{A}(t); t \in \widetilde{T}\}$ je disjunktný rozklad $\mathcal{X}$,*
2. *$\forall t \in T - \widetilde{T}$ je $\{\mathcal{A}(s); s \in sons(t)\}$ disjunktný rozklad $\mathcal{A}(t)$.*

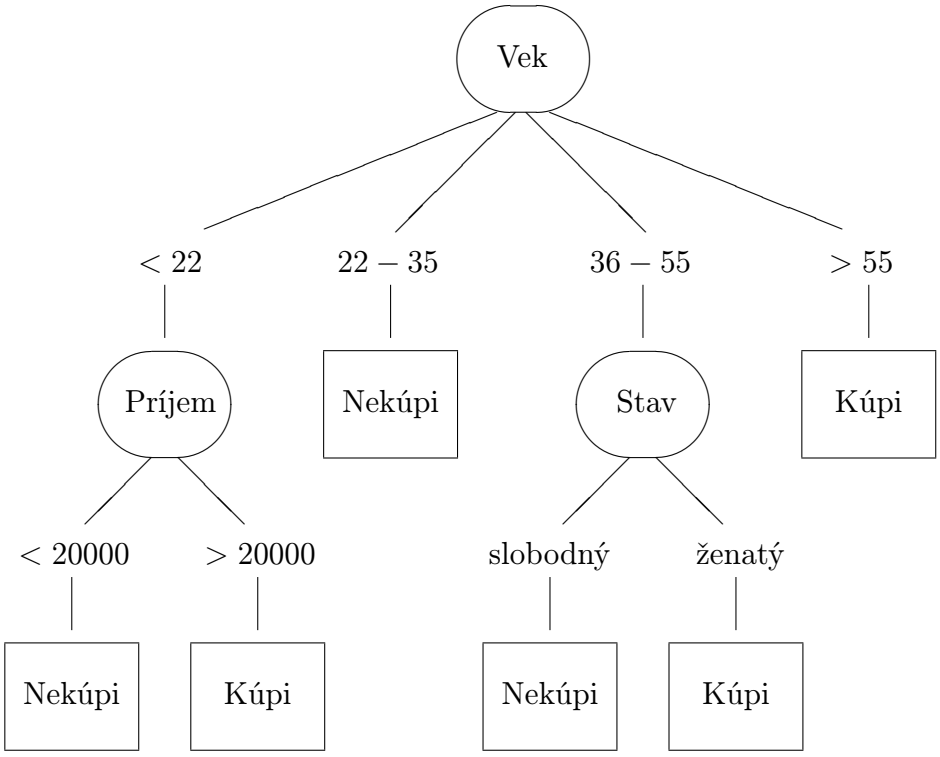
Klasifikačné pravidlo odpovedajúce takémuto klasifikačnému stromu potom znie

$$\forall \boldsymbol{x} \in \mathcal{X} \quad \forall t \in \widetilde{T} \quad d(\boldsymbol{x}) = j(t) \Longleftrightarrow \boldsymbol{x} \in \mathcal{A}(t).$$

Klasifikačný strom teda možno popísať ako klasifikátor založený na rekurzívnom delení podmnožín priestoru $\mathcal{X}$ na disjunktné časti, ktoré odpovedajú uzlom stromu. Rozklad začíname s $\mathcal{X}$, ktorý predstavuje koreň stromu. Postupne delíme vzniknuté podmnožiny. Delenie ukončíme prehlásením podmnožiny za koncovú. Tá potom predstavuje list stromu. Každej koncovej podmnožine je priradené číslo triedy, do ktorej zaradíme prvky danej podmnožiny. Rôzne koncové podmnožiny môžu mať pridelené rovnaké triedy.

2.2 Ilustračný príklad

Pre lepšiu predstavu o pojme klasifikičný strom a jeho využití pri riešení klasifikačných úloh si v tejto časti uvedieme jeden ilustračný príklad. Za úlohu máme predpovedať či si zákazník kúpi alebo nekúpi daný tovar. Jedná sa teda o klasifikačnú úlohu, kde chceme zákazníkov rozdeliť do dvoch skupín. To prevedieme na základe informácii, ktoré o zákazníkoch máme. V našom prípade sa jedná o 4 údaje:
- vek: spojitá veličina
- príjem: spojitá veličina
- rodinný stav: kategoriálna veličina (slobodný/á, ženatý/vydatá)
- vzdelanie: kategoriálna veličina (základné, stredoškolské, vysokoškolské)



Obr. 2.1: Príklad jednoduchého klasifikačného stromu

Výsledný klasifikačný strom môže vyzerať ako je znázornené na ob-

rázku 2.1. Už pri zbežnom pohľade je vidieť ako sa bude pri klasifikácii postupovať. Ako najvhodnejšie sa ukázalo rozdeliť najprv zákazníkov podľa veku. Následne pre určité vekové skupiny už bez ďalších kritérií priamo predpokladáme, že si produkt kúpia alebo nekúpia. Pri klasifikácii zatiaľ nezaradených zákazníkov rozhoduje príjem a rodinný stav. Vidíme, že stupeň vzdelania sme pri klasifikácii nevyužili. To ale ešte nemusí znamenať, že nemá vplyv na rozhodnutie zákazníkov. Mohla len nastať situácia, že vzdelanie úzko súvisí s príjmom zákazníka a tým, že sme použili pri klasifikácii veľkosť príjmu, sme vyčerpali aj informáciu, ktorú prináša dosiahnuté vzdelanie zákazníka.

2.3 Konštrukcia klasifikačného stromu

Postupy konštrukcie klasifikačných stromov možno podľa [4] rozdeliť do 4 skupín:

1. Konštrukcia „zdola nahor“, kde sú opakovane počítané vzdialenosti medzi dvojicami definovaných tried. Najbližšie triedy sú následne spojené.

2. Konštrukcia „zhora nahor“, kde sa stretávame s tromi problémami: ako vybrať vhodné deliace kritérium, ako rozhodnúť kedy delenie ukončiť, ako priradiť triedu listu.

3. Hybridná konštrukcia, ktorá kombinuje dva predošlé postupy.

4. Konštrukcia rast-prerezávanie, ktorá vychádza z konštrukcie „zhora nadol“, ale namiesto hľadania vhodného pravidla na zastavenie sa konštruuje rozsiahly strom, z ktorého následne odstraňujeme vetvy.

Podrobnejšie si popíšeme len (2) a (4) typ konštrukcie. Najprv sa budeme venovať konštrukcii stromu „zhora nadol“, ktorú s určitými zmenami využíva aj konštrukciu metódou rast-prerezávanie, ktorá bude podrobne popísaná v kapitole 4 ako súčasť algoritmu *CART*.

Aby sme mohli konštrukciu klasifikačného stromu popísať podrobnejšie, zavedieme nasledujúce značenia. Predpokladajme, že na konštrukciu stromu T použijeme učebný súbor $\mathcal{L}$, ktorý obsahuje N dát. Označme $p(t) = \frac{N(t)}{N}$, kde $N(t)$ je počet dát $(\boldsymbol{x}', y)' \in \mathcal{L}$, pre ktoré $\boldsymbol{x} \in \mathcal{A}_t$. Ďalej $p(j \mid t) = \frac{N_j(t)}{N(t)}$,

$j = 1, \ldots, J$, kde $N_j(t)$ je počet dát z $(\boldsymbol{x}', y)' \in \mathcal{L}$, pre ktoré $\boldsymbol{x} \in \mathcal{A}_t$ a zároveň $y = j$. Teda $p(t)$ a $p(j \mid t)$ sú odhady $P(\boldsymbol{X} \in \mathcal{A}_t)$ a $P(Y = j \mid \boldsymbol{X} \in \mathcal{A}_t)$.

Pri konštrukcii klasifikačného stromu „zhora nadol“ sa musíme zaoberať riešením 3 problémov:

1. Tvorba uzlov, t.j. nájsť vhodné pravidlo na delenie uzlov.
2. Rozhodnutie, kedy označiť uzol za koncový, t.j. nájsť pravidlo na zastavenia delenia.
3. Priradenie triedy koncovému uzlu.

2.3.1 Tvorba uzlov

Zaoberajme sa najprv problémom delenia priestoru $\mathcal{X}$ na podmnožiny, t.j. tvorbou uzlov stromu. Základná myšlienka je deliť podmnožiny tak, aby dáta v nasledujúcich uzloch boli „čistejšie“, tzn. homogénnejšie z hľadiska klasifikácie ako v predchádzajúcom uzle. K tomu bude potrebné definovať takzvanú funkciu nečistoty a pomocou nej mieru nečistoty.

Definícia 2.4. *Funkciou nečistoty nazveme funkciu ϕ definovanú pre J-tice čísel $(p_1, \ldots, p_J)$ spĺňajúce $p_j \geq 0$, $j = 1, \ldots, J$, $\sum_{j=1}^{J} p_j = 1$ takú, že platí:*

1. *$\phi \geq 0$,*
2. *ϕ nadobúda maxima jedine v bode $(\frac{1}{J}, \ldots, \frac{1}{J})$,*
3. *ϕ nadobúda minima jedine v bodoch $(1, 0, \ldots, 0)$, $(0, 1, \ldots, 0)$, $\ldots$, $(0, 0, \ldots, 1)$,*
4. *ϕ je symetrická funkcia premenných $p_1, \ldots, p_J$,*
5. *ϕ je rýdzo konkávna.*

Medzi najpoužívanejšie funkcie nečistoty patrí Giniho index diverzity definovaný ako

$$i(t) = 1 - \sum_{j=1}^{J} p^2(j \mid t)$$

a entropia daná vzťahom

$$i(t) = -\sum_{j=1}^{J} p(j \mid t) \log p(j \mid t).$$

Definícia 2.5. *Mieru nečistoty $i(t)$ pre uzol t definujeme ako*

$$i(t) = \phi\big(p(1 \mid t), \ldots, p(J \mid t)\big).$$

Aby sme sa mohli pomocou miery nečistoty rozhodovať medzi možnými deleniami uzlov, musíme zaviesť množinu kandidátov $\mathcal{S}$, ktorá bude obsahovať možné delenia uzlov. Pre každý uzol t potom máme kandidátov na delenie, ktorí rozdelia uzol t na časti, ktoré odpovedajú uzlom z množiny $sons(t)$.

Pre ilustráciu uvedieme možné delenie uzlu t na K uzlov, teda $sons(t) = \{t_1, \ldots, t_K\}$. Ak budeme uzol deliť podľa spojitého prediktoru X_1, ktorý nadobúda v uzle t hodnoty z intervalu (a, b), môže delenie vyzerať nasledovne:

$$\mathcal{A}(t_k) = \mathcal{A}(t) \cap \{\boldsymbol{x}; h_{k-1} < x_1 \leq h_k\}, \quad a = h_0 < h_1 < \ldots < h_K = b.$$

Pri delení podľa kategariálneho prediktoru X_2 môže mať delenie tvar

$$\mathcal{A}(t_k) = \mathcal{A}(t) \cap \{\boldsymbol{x}; x_2 \in A_k\},$$

kde A_k, $k = 1, \ldots, K$, je disjunktný rozklad množiny tried $\mathcal{C}_2(t)$, ktoré nadobúda veličina X_2 v uzle t. Poznamenajme ešte, že vzhľadom ku konečnému počtu prípadov v učebnom súbore existuje len konečný počet rôznych delení.

Pred každým delením uzlu teda prejdeme množinu $\mathcal{S}$ a získame tak všetky možné delenia daného uzlu. Aby sme vedeli rozhodnúť, ktoré delenie $s \in \mathcal{S}$ je najvhodnejšie pre uzol t, definujme *pokles nečistoty* ako

$$\Delta i(s, t) = i(t) - \sum_{s \in sons(t)} \frac{p(s)}{p(t)} i(s).$$

Tvrdenie 2.1. *Pre každý uzol t a jeho delenie s platí $\Delta i(s, t) \geq 0$. Navyše rovnosť nastáva práve vtedy, keď $p(j \mid s) = p(j \mid t),\ \forall s \in sons(t)\ \ j = 1, \ldots, J$.*

Dôkaz. Z rýdzej konkávnosti funkcie ϕ dostávame nasledujúce:

$$\begin{aligned}
\sum_{s \in sons(t)} i(s) \frac{p(s)}{p(t)} &= \phi\Big(\sum_{s \in sons(t)} p(1 \mid t_L), \ldots, \sum_{s \in sons(t)} p(J \mid t_L) \Big) \frac{p(s)}{p(t)} \\
&\leq \phi\Big(\sum_{s \in sons(t)} \frac{p(s)}{p(t)} p(1 \mid s), \ldots, \sum_{s \in sons(t)} \frac{p(s)}{p(t)} p(J \mid s) \Big).
\end{aligned}$$

Pritom rovnosť nastáva len ak je $p(j \mid s)$ rovnaké pre všetky $s \in sons(t)$, $j = 1, \ldots, J$. Vzťah

$$\begin{aligned}\sum_{s \in sons(t)} \frac{p(s)}{p(t)} p(j \mid s) &= \sum_{s \in sons(t)} \frac{N(s)}{N} \frac{N}{N(t)} \frac{N_j(s)}{N(s)} \\ &= \frac{1}{N(t)} \sum_{s \in sons(t)} N_j(s) = \frac{N_j(t)}{N(t)} = p(j \mid t)\end{aligned}$$

nám dáva

$$\begin{aligned}\sum_{s \in sons(t)} i(s) \frac{p(s)}{p(t)} &\leq \phi\big(\sum_{s \in sons(t)} p(1 \mid t_L), \ldots, \sum_{s \in sons(t)} p(J \mid t_L)\big) \frac{p(s)}{p(t)} \\ &= \phi\big(p(1 \mid t), \ldots, p(J \mid t)\big) = i(t).\end{aligned}$$

Pritom rovnosť nastáva opäť len ak je $p(j \mid s)$ rovnaké pre všetky $s \in sons(t)$, $j = 1, \ldots, J$. □

Čiže $\Delta i(s,t)$ je vždy nezáporné a teda pri delení nemôže dôjsť k nárastu nečistoty.

Pre rozdelenie uzlu t nájdeme medzi všetkými možnými deleniami také delenie $s^* \in \mathcal{S}$ uzlu t, pre ktoré platí

$$\Delta i(s^*, t) = \max_{s \in \mathcal{S}} \Delta i(s, t)$$

a podľa neho rozdelíme uzol t. Tento istý postup potom aplikujeme na všetkých vzniknutých synov $s \in sons(t)$.

2.3.2 Pravidlo na ukončenie delenia

Postup na delenie uzlov sme si už popísali, môžeme sa teda venovať ďalšej časti konštrukcie stromu a to je rozhodovanie kedy ukončiť delenie uzlov. Predpokladajme, že sme už previedli určité delenie. Množina použitých delení, čiže prvkov množiny $\mathcal{S}$, spolu s ich poradím v akom boli použité, určujú strom T. Uzly, ktoré sme ešte ďalej nedelili, tvoria množinu koncových uzlov $\widetilde{T}$. Položme $I(t) = p(t)i(t)$ a definujme *nečistotu stromu* $I(T)$ ako

$$I(T) = \sum_{t \in \widetilde{T}} I(t) = \sum_{t \in \widetilde{T}} p(t) i(t).$$

Zvoľme jeden koncový uzol $t \in \widetilde{T}$ a rozdeľme ho pomocou delenia s. Vzniknutý strom T' má nečistotu

$$I(T') = \sum_{\widetilde{T}-\{t\}} I(t) + \sum_{s \in sons(t)} I(s).$$

*P*okles nečistoty stromu potom budeme potom rovný

$$\Delta I(s,t) = I(T) - I(T') = I(t) - \sum_{s \in sons(t)} I(s).$$

Z toho, že $\Delta i(s,t) \geq 0$ dostávame, že aj pokles nečistoty stromu je vždy nezáporný.

$$\begin{aligned} \Delta I(s,t) &= I(t) - \sum_{s \in sons(t)} I(s) = p(t)i(t) - \sum_{s \in sons(t)} p(s)i(s) \\ &= p(t)i(t) - p(t) \sum_{s \in sons(t)} \frac{p(s)}{p(t)} i(s) = p(t)\Delta i(s,t) \geq 0. \end{aligned}$$

Metóda používaná k nájdeniu pravidla pre ukončenie delenia je jednoduchá, ale ako sa neskôr ukázalo, toto pravidlo má určité nedostatky. Zvoľme hodnotu $\beta > 0$ a označme t za koncový, ak

$$\max_{s \in \mathcal{S}} \Delta I(s,t) < \beta.$$

Iné pravidlo na zastavenie delenia pri konštrukcii stromu „zhora nadol“ využíva algoritmus *FIRM*, ktorý ale uvažuje len kategoriálne prediktory, pričom vysvetľovaná premenná môže byť aj spojitá aj kategoriálna. *FIRM* využíva pri konštrukcii stromu kontingenčné tabuľky a pri rohodovaní o zastavení delenia využíva testy významnosti a volenú minimálnu hladinu α.

Oba prístupy majú spoločný problém. Tým je voľba konštanty β (resp. α). Keď je konštanta nastavená ako príliš malá tak to vedie ku konštrukcii veľkých stromov, čo je často nežiadúce. Pri jej zvyšení môže naopak nastať situácia, že zamietneme delenie, ktoré síce priamo neprinesie veľký pokles nečistoty (resp. nie je významné), ale pri následnom delení novovzniknutých uzlov by bol pokles nečistoty výrazný (resp. delenie významné).

2.3.3 Priradenie triedy koncovému uzlu

Najjednoduchšia časť konštrukcie klasifikačného stromu je priradenie triedy koncovému uzlu. Pravidlo pre priradenie triedy koncovému uzlu môžeme

chápať ako funkciu $j(t)$, ktorá každému koncovému uzlu $t \in \widetilde{T}$ priradí jednu triedu z množiny $\mathcal{C}$. Pri priradení triedy $j(t)$ koncovému uzlu t dostaneme odhad chyby klasifikácie rovný

$$\sum_{j \neq j(t)} p(j \mid t) = 1 - p\big(j(t) \mid t\big).$$

Snaha minimalizovať chybu klasifikácie nás vedie k tomu, aby sme priradili koncovému uzlu triedu $j(t)$ tak, aby platilo

$$p\big(j(t) \mid t\big) = \max_{j \in \mathcal{C}} p(j \mid t).$$

Kapitola 3

Regresné stromy

3.1 Regresná úloha a prediktor

Regresné úlohy sa svojím charakterom značne podobajú úlohám klasifikačným. Sledujeme určitý objekt a jeho vlastnosti, popísané vysvetľujúcimi veličinami, a pomocou nich potom chceme určiť (predikovať), akú hodnotu nadobúda veličina, o ktorú sa zaujímame. Tú predstavuje vysvetľovaná veličina a na rozdiel od klasifikačnej úlohy sa nejedná o diskrétnu veličinu ale o veličinu spojitú.

Uvažujeme náhodný vektor $(\boldsymbol{X}', Y)'$. Vektor $\boldsymbol{X}' = (X_1, \ldots, X_m)$ predstavuje vektor vysvetľujúcich veličín. Jeho zložky, veličiny X_i, môžu byť spojité aj kategoriálne. Veličina Y predstavuje vysvetľovanú veličinu. Jedná sa o spojitú veličinu, ktorej hodnoty sú reálne čísla. Pravidlo na predpoveď hodnoty vysvetľovanej premennej budeme nazývať *predikčné pravidlo* alebo len prediktor. Keďže predikčné pravidlo v regresii zastáva rovnakú úlohu ako klasifikačné pravidlo v klasifikácii, budeme aj prediktor (tak ako klasifikátor) značiť $d(\boldsymbol{x})$. Predikčné pravidlo teda definujeme ako reálnu funkciu $d(\boldsymbol{x})$ definovanú na $\mathcal{X}$, ktorá každému $\boldsymbol{x}$ priradí $y \in \mathbb{R}$. Konštrukcia prediktoru by mala sledovať dva ciele:

1. Čo najpresnejšie predpovedať hodnotu veličiny Y na základe vektoru $\boldsymbol{X}$.

2. Vysvetliť vzťahy medzi vysvetľovanou a vysvetľujúcimi veličinami.

Aby sme mohli predikčné pravidlo skonštruovať, budeme potrebovať súbor dát $\mathcal{L} = \{(\boldsymbol{x}'_1, y_1)', \ldots, (\boldsymbol{x}'_N, y_N)'\}$, kde $\boldsymbol{x}_i \in \mathcal{X}$ a $y_i \in \mathbb{R}$, ktorý je zís-

kaný náhodným výberom z rovnakého rozdelenia ako aj spomínaný vektor $(\boldsymbol{X}', Y)'$.

3.2 Odhad chyby

Pri riešení regresnej úlohy máme k dispozícii dve základné varianty, a to variantu najmenších štvorcov a variantu najmenších absolútnych odchýliek. Preto máme aj dve možnosti ako určiť chybu zostrojeného prediktoru a tým aj jeho presnosť. V tejto práci budeme používať variantu najmenších štvorcov. Chybu predikcie bude teda určovat stredná štvorcová chyba, ktorá podobne ako chyba klasifikácie, určuje presnosť zostrojeného pravidla. Preto ju budeme tiež značiť $R^*(d)$.

Definícia 3.1. *Stredná štvorcová chyba $R^*(d)$ predikčného pravidla d je definovaná ako*

$$R^*(d) = E\big(Y - d(\boldsymbol{X})\big)^2.$$

Pri odhadovaní veľkosti strednej štvorcovej chyby máme k dispozícii viacero druhov odhadov. Tie sú podobné ako odhady chyby klasifikácie a preto sa im už nebudeme tak podrobne venovať. Najčastejší je *resubstitučný odhad*, ktorý dostávame ako

$$R(d) = \frac{1}{N} \sum_{(\boldsymbol{x}_n', y_n)' \in \mathcal{L}} \big(y_n - d(\boldsymbol{x}_n)\big)^2.$$

Pomocou rozdelenia súboru $\mathcal{L}$ na $\mathcal{L}_1$, ktorý použijeme pri konštrukcii $d(\boldsymbol{x})$ a $\mathcal{L}_2$, získame *testový odhad*

$$R^{ts}(d) = \frac{1}{N_2} \sum_{(\boldsymbol{x}_n', y_n)' \in \mathcal{L}_2} \big(y_n - d(\boldsymbol{x}_n)\big)^2,$$

kde N_2 je počet dát v $\mathcal{L}_2$.
Ak učebný súbor rozdelíme na V častí (označme $\mathcal{L}_1, \ldots, \mathcal{L}_V$) a pre každú z nich zostrojíme $d^v(\boldsymbol{x})$, získame odhad V-násobným krížovým overovaním

$$R^{cv}(d) = \frac{1}{N} \sum_{v=1}^{V} \sum_{(\boldsymbol{x}_n', y_n)' \in \mathcal{L}_v} \big(y_n - d^v(\boldsymbol{x}_n)\big)^2.$$

Je zrejmé, že veľkosť strednej štvorcovej chyby závisí na veľkosti hodnôt, ktoré sledovaná veličina Y nadobúda. Preto zavedieme ešte jednu mieru

presnosti, ktorá túto závislosť odstraňuje. Označme $\mu = E(Y)$ a uvažujme $R^*(\mu) = E(Y - \mu)^2$ strednú štvorcovú chybu pri použití hodnoty μ ako prediktoru.

Definícia 3.2. *Relatívna stredná štvorcová chyba $RE^*(d)$ predikčného pravidla d je definovaná ako*

$$RE^*(d) = \frac{R^*(d)}{R^*(\mu)}.$$

Pri značení $\overline{y} = \frac{1}{N}\sum_n y_n$ a $R(\overline{y}) = \frac{1}{N}\sum_n (y_n - \overline{y})^2$ dostávame odhady relatívnej štvorcovej chyby nasledovne. Resubstitučný odhad $RE(d) = R(d)/R(\overline{y})$, testový odhad $RE^{ts}(d) = R^{ts}(d)/R^{ts}(\overline{y})$ a odhad získaný pomocou krížového overovania $RE^{cv}(d) = R^{cv}(d)/R^{cv}(\overline{y})$.

3.3 Konštrukcia regresného stromu

Regresný strom má podobnú štruktúru ako klasifikačný strom, preto aj postup rastu stromu je podobný ako v prípade stromu klasifikačného. Navyše pri konštrukcii regresných stromov sa nestretávame s odhadmi apriórnych pravdepodobností, t.j. pravdepodobností výskytu jednotlivých tried.

Konštrukciu regresného stromu možno zjednodušene popísať podobne ako v prípade klasifikácie. Priestor $\mathcal{X}$ rekurzívne delíme na disjunktné podmnožiny, ktoré predstavujú uzly stromu. Delenie končíme prehlásením, že vzniknutá podmnožina predstavuje list stromu. Keďže na rozdiel od klasifikácie nemáme triedy, ktoré by sme mohli vzniknutému listu prideliť, nahrádza ich úlohu vhodná číselná hodnota.

Konštrukciu regresného stromu možno rozdeliť do 3 krokov:

1. Tvorba uzlov, t.j. vhodné pravidlo na delenie uzlov.
2. Rozhodnutie kedy označiť uzol za koncový, t.j. pravidlo na ukončenie delenia.
3. Priradenie hodnoty $y(t)$ koncovému uzlu.

Priradenie hodnoty $y(t)$ koncovému uzlu

Vychádzať budeme z resubstitučného odhadu strednej štvorcovej chyby zostrojeného stromu

$$R(d) = \frac{1}{N} \sum_n \big(y_n - d(\boldsymbol{x}_n)\big)^2.$$

Je pochopiteľné, že predikčné pravidlo chceme voliť tak, aby vzniknutá chyba bola čo najmenšia. Preto hodnotu $y(t)$ chceme voliť tak, aby

$$R(d) = \frac{1}{N} \sum_n \big(y_n - y(t)\big)^2$$

bolo čo najmenšie.

Tvrdenie 3.1. *Hodnotu $y(t)$, ktorá minimalizuje odhad chyby $R(d)$, získame ako priemer hodnôt y_n z prípadov $(\boldsymbol{x}'_n, y_n)'$, pre ktoré platí $\boldsymbol{x}_n \in t$ a budeme ju značiť $\overline{y}(t)$, t.j.*

$$\overline{y}(t) = \frac{1}{N(t)} \sum_{\boldsymbol{x}_n \in t} y_n,$$

kde N_t je počet prípadov v uzle t.

Tvrdenie dostávame pri minimalizácii vzťahu $\sum_n (y_n - a)^2$.

Hodnotu, ktorú priradíme uzlu t, získame ako

$$\overline{y}(t) = \frac{1}{N(t)} \sum_{\boldsymbol{x}_n \in t} y_n.$$

Delenie uzlov

Pri výbere vhodného delenia postupujeme podobne ako pri klasifikačnom strome. Aj tu volíme ako najlepšie to delenie, ktoré spôsobí najväčší pokles nečistoty stromu. Nečistotu budeme v prípade regresie chápať ako variabilitu a je preto celkom logické, že úlohu miery nečistoty stromu berie na seba sledovaná stredná štvorcová chyba. Na nasledujúcich pár riadkoch popíšeme ako bude výber najlepšieho delenia prebiehať.

Vzhľadom na predchádzajúcu časť budeme každému uzlu t priraďovať hodnotu $\overline{y}(t)$. Po zohľadnení faktu, že prediktor má tvar stromu, môžeme

testový odhad strednej štvorcovej chyby písať v tvare

$$R(d) = R(T) = \frac{1}{N} \sum_{t \in \widetilde{T}} \sum_{\boldsymbol{x}_n \in t} \big(y_n - y(t)\big)^2.$$

Ďalej položme pre každý uzol t stromu T

$$R(t) = \frac{1}{N} \sum_{\boldsymbol{x}_n \in t} \big(y_n - y(t)\big)^2$$

a teda platí

$$R(T) = \sum_{t \in \widetilde{T}} R(t).$$

Uvažujme teraz množinu možných delení $\mathcal{S}$. Nech delenie s rozdelí uzol t na K synov $t_1, \ldots, t_K$. Označme

$$\Delta R(s,t) = R(t) - \sum_{k=1}^{K} R(t_k).$$

Je zrejmé, že ak budeme uzol t deliť pomocou delenia s, ktoré $\Delta R(s,t)$ maximalizuje, dosiahneme tak aj maximálny pokles v $R(T)$. Ako najlepšie delenie s^* pre uzol t preto vyberieme také delenie, ktoré spĺňa

$$\Delta R(s^*,t) = \max_{s \in \mathcal{S}} \Delta R(s,t).$$

Pravidlo na ukončenie delenia

Aj pri regresných stromoch nastáva problém s určením vhodného pravidla na ukončenie delenia. Podobne ako pri klasifikácii aj tu je možnosť ukončiť delenie uzlu t ak už dochádza len k malému poklesu sledovanej štatistiky, v tomto prípade $\Delta R(s,t)$. Pravidlo by teda mohlo znieť, že delenie končíme ak nastane

$$\max_{s \in \mathcal{S}} \Delta R(s,t) < \beta,$$

kde β je pevne volená konštanta. Zase ale narážame na problém ako voliť konštantu β a na nedostatky tohto pravidla popísané v časti 2.3.2. Preto je vhodnejšie riešiť problém koncového pravidla použitím konštrukcie rast-prerezávanie, ktorá bude podrobne popísaná v kapitole 4.

3.4 Prehľad konštrukčných algoritmov

Na konštrukciu klasifikačných a regresných stromov bolo navrhnutých viacero algoritmov. V tejto časti uvedieme a stručne popíšeme niektoré z nich.

- *CART*- podrobnejšie ho popíšeme, v kapitole 4. Algoritmus produkuje binárne klasifikačné a regresné stromy. Na to používa konštrukciu typu rast-prerezávanie. Kritérium na delenie uzlov je maximálny pokles nečistoty, kde za funkciu nečistoty často volíme Giniho index diverzity.

- *FIRM*- produkuje nebinárne klasifikačné a regresné stromy pričom uvažuje len kategorické vysvetľujúce premenné. Algoritmus konštruuje strom, pomocou konštrukcie typu „zhora nadol“ a delenie zastaví v prípade, že nie je možné nájsť významné delenie. Pri konštrukcii používa kontingenčné tabuľky a štatistické testy. V prípade klasifikácie χ^2-test, v prípade predikcie dvojvýberový t-test. Podrobný popis algoritmu možno nájsť v [3].

- *QUEST*- produkuje binárne klasifikačné stromy. Konštruuje strom pomocou metódy rast-prerezávanie. Na určenie premennej, ktorá riadi delenie v danom uzle využíva štatistické testy, následne pri určení vhodného delenia využíva okrem iného diskriminačnú analýzu. Podrobný popis možno nájsť v [5].

- *ID 3*- jednoduchý algoritmus, ktorý produkuje nebinárne klasifikačné stromy. Uvažuje len kategorické veličiny. Pri delení uzlu vytvorí taký počet synov, koľko kategórií má premenná, ktorou sa delenie riadi. Tú vyberá pomocou maximalizácie informačného zisku. Podrobný popis možno nájsť v [7].

- *C4.5*- predstavuje rozšírenie algoritmu *ID3*, pričom odstraňuje jeho nedostatky. Dokáže si poradiť so spojitými veličinami a chýbajúcimi pozorovaniami. Na rozdiel od *ID3* používa na zostrojenie konečného stromu aj prerezávanie. V súčasnosti je to jeden z najpoužívanejších algoritmov.

- a mnoho ďalších, napr.:*CHAID, CRUISE, GUIDE, ...*

3.5 Výhody použitia klasifikačných a regresných stromov

Medzi hlavné výhody použitia klasifikačných a regresných stromov pri klasifikácii resp. predikcii patrí:

1. Jednoduchá a názorná prezentácia výsledkov.
2. Schopnosť pracovať s kategoriálnymi aj spojitými veličinami.
3. Minimálne predpoklady na dáta.
4. Metóda je odolná voči odľahlým pozorovaniam.
5. Schopnosť vysporiadať sa s chýbajúcimi pozorovaniami.
6. Metóda dokáže stanoviť dôležitosť vysvetľujúcich premenných.

Kapitola 4

Algoritmus *CART*

V tejto kapitole podrobne popíšeme algorimus *CART* (Classification And Regression Trees). Čerpať budeme hlavne z [2]. Algoritmus vytvára binárny klasifikačný alebo regresný strom a využíva pri tom konštrukciu typu rast-prerezávanie. Preto sa budeme v tejto kapitole najprv venovať rastu počiatočného stromu T_{max}, popísanom v časti 4.1, následne v časti 4.2 popíšeme metódu postupného prerezávania stromu T_{max}. Problému výberu najlepšieho stromu, ktorý súvisí s odhadom chyby klasifikácie, sa budeme venovať v časti 4.3. Časť 4.4 bude venovaná aplikácii algoritmu na konštrukciu regresného stromu a na záver popíšeme ako algoritmus rieši problém chýbajúcich pozorovaní a určuje významnosť vysvetľujúcich veličín.

4.1 Rast stromu

Ako bolo spomenuté *CART* vytvára binárny strom metódou rast-prerezávanie. Vytvorí preto najprv strom T_{max}, na ktorý potom aplikuje postup prerezávania. Pri konštrukcii stromu T_{max} vychádzame z konštrukcie typu „zhora nadol“. Preto sa na tomto mieste budeme venovať odlišnostiam voči postupu popísanom v časti 2.3.

Tvorba uzlov

Kedže chceme zostrojiť binárny strom, budeme uzol vždy deliť len na dvoch synov, teda vždy bude platiť $|sons(t)| = 2$ alebo $|sons(t)| = 0$ v prípade, že uzol je listom. Synov uzlu t budeme značiť t_L a t_R. Pre binárne stromy sa nám teda zjednoduší množina možných delení $\mathcal{S}$. Množina $\mathcal{S}$ je reprezento-

vaná množinou dichotomických otázok $\mathcal{Q}$ tvaru {Je $\boldsymbol{x} \in \mathcal{A}$?}, kde $\mathcal{A} \subset \mathcal{X}$. Pri odpovedi „áno“ zaradíme $\boldsymbol{x}$ do t_L inak do t_R. Pre štandardné súbory dát, t.j. ak $\mathcal{X}$ má pevnú dimenzionalitu a prediktory môžu byť spojité aj kategoriálne, môžeme množinu otázok $\mathcal{Q}$ štandardizovať. *Štandardizovanú množinu otázok* $\mathcal{Q}$ definujeme nasledovne:

- Každé delenie závisí iba na hodnote práve jednej premenej.
- Pre každú merateľnú veličinu X_m obsahuje $\mathcal{Q}$ všetky otázky tvaru {Platí $x_m \leq c$?}, kde $c \in (-\infty, \infty)$.
- Ak je X_m kategoriálna s hodnotami z množiny $\mathcal{C}_m$, potom $\mathcal{Q}$ obsahuje všetky otázky tvaru {Platí $x_m \in S$?}, kde $S \subseteq \mathcal{C}_m$.

Počet rôznych delení je vždy konečný a závisí na druhu dát a ich počte. Uvažujme spojitú veličinu X_m, ktorá v súbore dát nadobúda N rôznych hodnôt $x_{m,1}, \ldots, x_{m,N}$ zoradených podľa veľkosti. Potom máme maximálne $N-1$ odlišných delení a tie sú reprezentované otázkami {Je $x_m \leq c_n$?}, $n = 1, ...N-1$ kde $c_i \in (x_{m,i}, x_{m,i+1})$. Pre kategoriálnu veličinu X_m, ktorá nadobúda N rôznych hodnôt, získame počet rôznych delení keď od počtu všetkých podmnožín odčítame prázdnu množinu a množinu obsahujúcu všetky kategórie a výsledný počet vydelíme dvoma, keďže otázky {Platí $x_m \in S$?} a {Platí $x_m \notin S$?} dávajú rovnaké delenie. Máme teda $\frac{2^N-2}{2}$ rôznych delení.

Na výber vhodného delenia sa najčastejšie uvádzajú dve kritéria:

- Giniho kritérium, ktoré ako funkciu nečistoty používa

$$i(t) = \sum_{\substack{i=1 \\ i \neq j}}^{J} p(j|t)p(i|t) = 1 - \sum_{j=1}^{J} p^2(j \mid t)$$

  a pokles nečistoty je daný

$$\Delta i(s,t) = i(t) - p_L i(t_L) - p_R i(t_R),$$

  kde p_R resp. p_L je podiel prípadov z t, ktoré sú zaradené do t_R resp. t_L.

- Twoing kritérium, pri ktorom je pokles nečistoty daný vzťahom

$$\Delta i(s,t) = p_L p_R \Big[\sum_{j=1}^{J} |p(j|t_L) - p(j|t_R)| \Big].$$

Pravidlo na zastavenie delenia

Pre ukončenie delenia uzlov máme k dispozícii dve pravidlá. Prvým je, že delenie ukončíme, až keď bude uzol „čistý“, t.j. bude obsahovať len prípady z jednej triedy. Iný spôsob je, že delenie zastavíme ak počet prípadov v uzle bude menší alebo rovný číslu N_{min}, ktoré sa bežne volí ako 1 alebo 5.

4.2 Prerezávanie

V predchádzajúcej časti sme zostrojili strom T_{max}, z ktorého postupným prerezávaním vetiev získame postupnosť jeho podstromov. Z nej potom budeme vyberať „najlepší strom“. Je zrejmé, že strom T_{max} bude vo väčšine prípadoch rozsiahly, takže bude k dispozícii množstvo odlišných spôsobov prerezávania. Preto bude potrebný systematický prístup k prerezávaniu, ktorý popíšeme. Predtým ale ešte zavedieme pojmy, ktoré budeme potrebovať.

Definícia 4.1. *Pre každý podstrom $T \preccurlyeq T_{max}$ definujme jeho zložitosť ako $|\widetilde{T}|$. Nech $\alpha \geq 0$ je reálne číslo, tzv. parameter zložitosti. Definujme penalizovanú mieru $R_\alpha(T)$ ako*

$$R_\alpha(T) = R(T) + \alpha|\widetilde{T}|.$$

Definícia 4.2. *Najmenší optimálne prerezaný podstrom $T(\alpha)$ pre parameter zložitosti α definujeme podľa podmienok:*

1. $R_\alpha(T(\alpha)) = \min_{T \preccurlyeq T_{max}} R_\alpha(T)$.
2. *Ak $R_\alpha(T) = R_\alpha(T(\alpha))$, potom $T(\alpha) \preccurlyeq T$.*

Dá sa dokázať, že $T(\alpha)$ vyhovujúce definícii 4.2 vždy existuje. Dôkaz možno nájsť napríklad v [2].

Samotné prerezávanie začíname so stromom $T_1 = T(0)$, ten je potom najmenší podstrom stromu T_{max}, pre ktorý platí $R(T_1) = R(T_{max})$. Strom T_1 získame nasledovne. Nech t_L a t_R sú listy stromu T_{max} a zároveň synmi uzlu t. Ak platí $R(t) = R(t_L) + R(t_R)$, potom odrežeme uzly t_L a t_R. Tento postup opakujeme. Ak ďalšie odstránenie uzlov už nie je možné, tak sme získali strom T_1.

Pre každú vetvu T_t stromu T_1 definujme $R(T_t)$ vzťahom

$$R(T_t) = \sum_{t' \in \widetilde{T}_t} R(t'),$$

kde $\widetilde{T}_t$ je množina listov vetvy T_t. Z konštrukcie stromu T_1 je zrejmé, že pre každý uzol t stromu T_1, ktorý nie je jeho listom, platí $R(t) > R(T_t)$. Položme

$$R_\alpha(\{t\}) = R(t) + \alpha \quad \text{a} \quad R_\alpha(T_t) = R(T_t) + \alpha|\widetilde{T}_t|.$$

Pokým platí $R_\alpha(T_t) < R_\alpha(\{t\})$ tak odstránením vetvy T_t zo stromu by sme penalizovanú mieru zvýšili. Pre určitú hodnotu α (kritickú hodnotu) ale nastane rovnosť a následne pri jej zvyšovaný sa nerovnosť obráti. Riešením nerovnosti $R_\alpha(T_t) < R_\alpha(\{t\})$ dostávame

$$\alpha < \frac{R(t) - R(t_t)}{|\widetilde{T}_t| - 1}.$$

Definujme funkciu $g_1(t)$, $t \in T_1$ ako

$$g_1(t) = \begin{cases} \frac{R(t)-R(T_t)}{|\widetilde{T}_t|-1} & \text{pre } t \notin \widetilde{T}_1 \\ \infty & \text{pre } t \in \widetilde{T}_1. \end{cases}$$

Teraz nájdeme „najslabší článok“ $\bar{t}_1 \in T_1$ ako uzol, pre ktorý platí

$$g_1(\bar{t}_1) = \min_{t \in T_1} g_1(t)$$

a položíme $\alpha_2 = g_1(\bar{t}_1)$. Uzol $\bar{t}_1 \in T_1$ je najslabší v tom zmysle, že so zväčšujúcim sa α dosiahne práve on ako prvý rovnosť medzi $R_\alpha(\{t\})$ a $R_\alpha(T_t)$. To nastane práve pre hodnotu α_2. Teda pre $\alpha \geq \alpha_2$ preferujeme $\bar{t}_t$ pred $T_{\bar{t}_1}$.

V ďalšom kroku vytvoríme odrezaním vetvy $T_{\bar{t}_1}$ zo stromu T_1 strom T_2, teda

$$T_2 = T_1 - T_{\bar{t}_1}.$$

Na strome T_2 teraz aplikujeme postup popísaný pre T_1. Zostrojíme funkciu $g_2(t)$ definovanú pre $t \in T_2$ ako

$$g_2(t) = \begin{cases} \frac{R(t)-R(T_t)}{|\widetilde{T}_t|-1} & \text{pre } t \notin \widetilde{T}_2 \\ \infty & \text{pre } t \in \widetilde{T}_2. \end{cases}$$

Následne nájdeme $\bar{t}_2$ a α_3. Zostrojíme strom T_3 a opakujeme postup.

Ak sa v určitej fáze prerezávania dostaneme do situácie, že nastane

$$g_k(\bar{t}_k) = g_k(\bar{t}'_k),$$

potom strom T_{k+1} získame odrezaním oboch vetiev $T_{\bar{t}_k}$ a $T_{\bar{t}'_k}$ teda

$$T_{k+1} = T_k - T_{\bar{t}_k} - T_{\bar{t}'_k}.$$

Popísaným spôsobom teda dostaneme vnorenú postupnosť stromov

$$T_1 \succ T_2 \succ T_3 \succ \ldots \succ \{t_1\}.$$

4.3 Výber optimálneho stromu

Proces samotného prerezávania máme už popísaný, teraz je potrebné zo získanej postupnosti stromov $T_1 \succ T_2 \succ T_3 \succ \ldots \succ \{t_1\}$ vybrať najlepší. Výber medzi stromami prevedieme pomocou odhadov chyby klasifikácie. Pre odhad $\widehat{R}(T_k)$ budeme voliť za najlepší ten strom T_{k_0}, ktorý spĺňa

$$\widehat{R}(T_{k_0}) = \min_k \widehat{R}(T_k).$$

Je zrejmé, že pri použití resubstitučného odhadu $R(T_k)$ bude za najlepší strom považovaný maximálny strom, teda T_1. To je ale nežiadúce a preto popíšeme použitie ďalších odhadov chyby klasifikácie.

Testový odhad

Ako bolo už popísané v časti 1.3, pri použití testového odhadu chyby klasifikácie musíme učebný súbor $\mathcal{L}$ náhodne rozdeliť na podsúbory $\mathcal{L}_1$ a $\mathcal{L}_2$. Súbor $\mathcal{L}_1$ budeme používať na zostrojenie klasifikátora a $\mathcal{L}_2$, tzv. *testový súbor* použijeme na samotné testovanie. V praxi sa používa delenie aby pre počet prípadov $N^{(2)}$, ktoré budú v $\mathcal{L}_2$, platilo $N^{(2)} = N/3$, kde N je celkový počet prípadov v učebnom súbore. Pomocou $\mathcal{L}_1$ zostrojíme najprv strom T_{max} a následne pomocou prerezávania získame postupnosť jeho podstromov $T_1 \succ T_2 \succ T_3 \succ \ldots \succ \{t_1\}$. Teraz použijeme súbor dát $\mathcal{L}_2$ na testovanie. Označme $N_j^{(2)}$ počet prípadov v $\mathcal{L}_2$, ktoré patria do triedy j. Pre každý strom $T \in \{T_1, T_2, \ldots, \{t_1\}\}$ označme $N_{ij}^{(2)}$ počet prípadov triedy j, ktoré strom T klasifikoval ako prípady z triedy i. Odhad pravdepodobnosti, že prípad triedy j strom nesprávne klasifikuje ako prípad triedy i, je daný vzťahom

$$Q^{ts}(i|j) = N_{ij}^{(2)}/N_j^{(2)}$$

a pre prípad, že $N_j^{(2)} = 0$ definujeme $Q^{ts}(i|j) = 0$. Na základe toho dostaneme odhad pravdepodobnosti nesprávnej klasifikácie prípadu j-tej triedy

$$R^{ts}(j) = \sum_{\substack{i=1 \\ i \neq j}}^{J} Q^{ts}(i|j).$$

Po označení apriórnej pravdepodobnosti $P(Y = j)$ symbolom π_j, dostávame pre testový odhad chyby klasifikácie

$$R^{ts}(T) = \sum_{j=1}^{J} R^{ts}(j)\pi_j$$

a po dosadení odhadu $\pi_j = \frac{N_j^{(2)}}{N^{(2)}}$

$$R^{ts}(T) = \sum_{j=1}^{J} R^{ts}(j)\frac{N_j^{(2)}}{N^{(2)}}.$$

Za optimálny strom získaný pomocou testového odhadu berieme ten strom $T_{k_0} \in \{T_1, T_2, \ldots, \{t_1\}\}$, pre ktorý platí

$$R^{ts}(T_{k_0}) = \min_k R^{ts}(T_k).$$

Odhad V-násobným krížovým overovaním

Ak máme k dispozícii rozsiahly učebný súbor tak pri odhadovaní chyby klasifikácie uprednostňujeme odhad V-násobným krížovým overovaním pred testovým odhadom. Učebný súbor náhodne rozdelíme na V približne rovnako veľkých podsúborov $\mathcal{L}_v$, $v = 1, \ldots, V$. Ďalej zavedieme značenie

$$\mathcal{L}^{(v)} = \mathcal{L} - \mathcal{L}_v, \;\; v = 1, \ldots, V.$$

Na základe súborov dát $\mathcal{L}$ a $\mathcal{L}^{(v)}$, $v = 1, \ldots, V$ zostrojíme najprv stromy T_{max} a $T_{max}^{(v)}$, $v = 1, \ldots, V$. Pre každý parameter zložitosti α označíme $T(\alpha)$ a $T^{(v)}(\alpha)$ najmenšie minimálne prerezané stromy získané z T_{max} a $T_{max}^{(v)}$, $v = 1, \ldots, V$. Kedže stromy $T_{max}^{(v)}$ sú konštruované na základe $\mathcal{L}^{(v)}$ budeme pomocou nich klasifikovať dáta $\mathcal{L}_v$, ktoré neboli použité pri ich konštrukcii. Pre pevné hodnoty parametru α a všetky v, i, j definujeme $N_{ij}^{(v)}$ ako počet prípadov triedy j z $\mathcal{L}_v$, ktorým strom $T^{(v)}(\alpha)$ priradil triedu i a položíme $N_{ij} = \sum_{v=1}^{V} N_{ij}^{v}$. Ďalej postupujeme na základe idey, že pre veľké V, majú stromy $T^{(v)}(\alpha)$ a $T(\alpha)$ skoro rovnaké chyby klasifikácie, keďže sú zostrojené z takmer rovnakého súboru dát. Preto budeme odhadovať pravdepodobnosť, že strom $T(\alpha)$ priradil prípadu z j-tej triedy triedu i ako

$$Q^{cv}(i|j) = \frac{N_{ij}}{N_j}.$$

Pre pravdepodobnosť nesprávnej klasifikácie prípadu z j-tej triedy dostávame

$$R^{cv}(j) = \sum_{\substack{i=1 \\ i \neq j}}^{J} Q^{cv}(i|j)$$

a ak položíme $\pi_j = N_j/N$ máme

$$R^{cv}\big(T(\alpha)\big) = \sum_{j=1}^{J} R^{cv}(j)\pi_j = \frac{1}{N} \sum_{\substack{i,j=1 \\ i \neq j}}^{J} N_{ij}.$$

Teraz položme $\alpha'_k = \sqrt{\alpha_k \alpha_{k+1}}$. Odhad chyby klasifikácie získaný pomocou V-násobného krížového overovania pre strom T_k definujeme potom ako

$$R^{cv}(T_k) = R^{cv}\big(T(\alpha'_k)\big)$$

a za optimálny strom volíme strom T_{k_0}, ktorý spĺňa

$$R^{cv}(T_{k_0}) = \min_k R^{cv}(T_k).$$

Pravidlo 1 SE

Kvôli presnejšiemu odhadu chyby klasifikácie používame pri konštrukcii stromu a pri odhadovaní jeho chyby klasifikácie rozdielne súbory dát. Tie získavame náhodným delením učebného súboru dát $\mathcal{L}$. Toto náhodné delenie vnáša do procesu výberu optimálneho stromu určité problémy.

Pozrime sa na odhad chyby klasifikácie $\widehat{R}(T_k)$ $\big(R^{ts}(T_k)$ alebo $R^{cv}(T_k)$ $\big)$ ako funkciu závislú na počte uzlov $|\widetilde{T}_k|$. Podľa [2] má funkcia $\widehat{R}(T_k)$ vo väčšine prípadov nasledujúce charakteristiky. Z maxima, ktoré dosahuje pre $|\widetilde{T}_k| = 1$, s rastúcim počtom listov spočiatku rýchlo klesá. Od určitého $|\widetilde{T}_k|$ sa začne správať takmer ako konštanta. Následne pre veľké $|\widetilde{T}_k|$ začne $\widehat{R}(T_k)$ mierne stúpať. Práve počet listov $|\widetilde{T}_k|$, pre ktorý dosahuje funkcia $\widehat{R}(T_k)$ minima, býva citlivý na zmeny pri delení $\mathcal{L}$. Preto sa za optimálny strom neberie strom, ktorý dosahuje minimálnu chybu klasifikácie, ale najmenší strom, ktorého chyba je dostatočne blízka minimu. Na rozhodnutie, ktorá chyba to spĺňa a ktorá už nie, použijeme odhad smerodajnej odchýlky odhadu $\widehat{R}(T_k)$, ktorý budeme značiť $SE\big(\widehat{R}(T_k)\big)$.

Nech T_{k_0} je strom, pre ktorý platí

$$\widehat{R}(T_{k_0}) = \min_k \widehat{R}(T_k).$$

1 SE pravidlo vyberie za optimálny strom T_{k_1}, ktorý je najmenším stromom spĺňajúcim

$$\widehat{R}(T_{k_1}) \leq \widehat{R}(T_{k_0}) + SE\big(\widehat{R}(T_{k_0})\big).$$

4.4 Použitie algoritmu pre regresné stromy

4.4.1 Rast a prerezávanie

Pri konštrukcii regresného stromu v algoritme *CART* postupujeme podobne ako pri konštrukcii klasifikačného stromu. Najprv zostrojíme binárny strom T_{max}, na to použijeme postup uvedený v časti 3.3. Uzol stromu ďalej nedelíme, ak počet prípadov v uzle bude menší ako N_{min} (bežne sa volí $N_{min} = 5$). Aj tu sa ponúka možnosť zastaviť delenie ak všetky prípady v uzle majú rovnakú hodnotu y, ale keďže Y je spojitá veličina, táto situácia takmer vôbec nenastáva. To spôsobuje, že strom T_{max} má vo väčšine prípadov, keď sa jedná o regresný strom, viac uzlov ako strom klasifikačný.

Ďalšia časť algoritmu, prerezávanie, sa nelíši od postupu popísanom v časti 4.2, preto ju už nebudeme uvádzať. Výsledkom prerezávania je postupnosť stromov $T_1 \succ T_2 \succ T_3 \succ \ldots \succ \{t_1\}$.

4.4.2 Výber optimálneho stromu

Posledným krokom je z postupnosti stromov $T_1 \succ T_2 \succ T_3 \succ \ldots \succ \{t_1\}$ vybrať optimálny strom. Ten určíme pomocou odhadov strednej štvorcovej chyby $R(T)$. Za optimálny volíme strom T_{k_0}, ktorý tento odhad minimalizuje, teda spĺňa $\widehat{R}(T_{k_0}) = \min_k \widehat{R}(T_k)$.

Testový odhad

Keďže na rozdiel od odhadu chyby klasifikácie nenarážame pri odhade strednej štvorcovej chyby na objekty z rôznych tried, celý postup sa nám zjednoduší. Učebný súbor rozdelíme na súbor $\mathcal{L}_1$, ktorý sa použije na konštrukciu stromov T_k a súbor $\mathcal{L}_2$, pomocou ktorého odhadujeme chybu. Ak prediktor odpovedajúci stromu T_k označíme $d_k(\boldsymbol{x})$, dostávame testový odhad chyby

$$R^{ts}(T_k) = \frac{1}{N_2} \sum_{(\boldsymbol{x}'_n, y_n)' \in \mathcal{L}_2} (y_n - d_k(\boldsymbol{x}_n))^2,$$

kde N_2 je počet prípadov zo súboru $\mathcal{L}_2$.

Odhad V-násobným krížovým overovaním

Postupujeme podobne ako pri klasifikácii. Súbor $\mathcal{L}$ rozdelíme na V súborov $\mathcal{L}_1, \ldots, \mathcal{L}_V$. Ďalej pre $v = 1, \ldots, V$ označíme $\mathcal{L}^{(v)} = \mathcal{L} - \mathcal{L}_v$. Na každý súbor $\mathcal{L}^{(v)}$ aplikujeme proces rastu a prerezávania a označíme $T^{(v)}(\alpha)$ minimálne prerezané stromy pre parameter zložitosti α. Proces rastu a prerezávania teraz zopakujeme pre celý súbor $\mathcal{L}$, tak získame postuponosť stromov T_k a príslušnú postupnosť parametrov α_k. Definujme teraz $\alpha'_k = \sqrt{\alpha_k \alpha_{k+1}}$. Na záver označme $d_k^{(v)}(\boldsymbol{x})$ prediktor zodpovedajúci stromu $T^{(v)}(\alpha'_k)$. Potom odhad chyby získaný pomocou V-násobného krížového overovanie dostávame ako

$$R^{cv}(T_k) = \frac{1}{N} \sum_{v=1}^{V} \sum_{(\boldsymbol{x}'_n, y_n)' \in \mathcal{L}_v} (y_n - d_k^{(v)}(\boldsymbol{x}_n))^2.$$

1 SE pravidlo

Z podobných dôvodov ako v prípade klasifikačných stromov sa na výber optimálneho stromu v používa *1 SE* pravidlo. Nech strom pre strom T_{k_0} platí

$$\widehat{R}(T_{k_0}) = \min_k \widehat{R}(T_k).$$

Potom za optimálny považujeme strom T_{k_1}, ktorý je najmenším stromom splňujúcim

$$\widehat{R}(T_{k_1}) \leq \widehat{R}(T_{k_0}) + SE\big(\widehat{R}(T_{k_0})\big),$$

kde $SE\big(\widehat{R}(T_{k_0})\big)$ je odhad smerodajnej odchýlky odhadu $\widehat{R}(T_{k_0})$.
V praxi sa ako odhad $\widehat{R}$ používa odhad relatívnej chyby získaný pomocou krížového overovania, teda RE^{cv}. *1 SE* pravidlo má potom tvar

$$RE^{cv}(T_{k_1}) \leq RE^{cv}(T_{k_0}) + SE\big(RE^{cv}(T_{k_0})\big).$$

4.5 Chýbajúce pozorovania

Algoritmus *CART* sa v snahe maximálne využiť informácie z učebného súboru dát dokáže vysporiadať aj s chýbajúcimi hodnotami vysvetľujúcich veličín. To docieli tak, že okrem najlepšieho delenia pre daný uzol určuje aj delenia, ktoré sú použitému deleniu najviac podobné, z hľadiska výsledného rozdelenia uzlu, ale riadia sa na základe rozdielnej veličiny.

Uvažujme situáciu, že pri konštrukcii klasifikačného stromu sme vo fáze, keď chceme rozdeliť uzol t. Nájdeme preto najlepšie delenie s^*, ktoré uzol t rozdelí na t_L a t_R. Ďalej pre $m = 1, \ldots, M$ označme $\mathcal{S}_m$ množinu delení uzlu t podľa veličiny X_m a $\bar{\mathcal{S}}_m$ množinu delení k nim komplementárnych. Uvažujme delenie $s_m \in \mathcal{S}_m \cup \bar{\mathcal{S}}_m$. Nech s_m rozdelí uzol t na t'_L a t'_R. Označme $N_j(LL)$ počet prípadov zo skupiny j, ktoré obidve delenia pošlú naľavo, t.j. budú patriť do $t_L \cap t'_L$. Pravdepodobnosť, že prípad padne do $t_L \cap t'_L$ možno odhadnúť ako

$$p(t_L \cap t'_L) = \sum_j \pi(j) N_j(LL)/N_j.$$

Ďalej zaveďme odhad pravdepodobnosti $p_{LL}(s^*, s_m)$, že prípad z uzlu t zaradia obidve delenia s^*, s_m naľavo ako

$$p_{LL}(s^*, s_m) = p(t_L \cap t'_L)/p(t).$$

Podobné úvahy možno previesť aj v prípade, že obidve delenia zaradia prípad napravo a získame tak odhad

$$p_{RR}(s^*, s_m) = p(t_R \cap t'_R)/p(t).$$

Pravdepodobnosť, že delenia s^*, s_m zaradia prípad rovnako, možno teda odhadnúť ako

$$p(s^*, s_m) = p_{LL}(s^*, s_m) + p_{RR}(s^*, s_m).$$

Definícia 4.3. *Delenie $\tilde{s}_m \in \mathcal{S}_m \cup \bar{\mathcal{S}}_m$ nazývame náhradným delením pre X_m k deleniu s^* ak platí*

$$p(s^*, \tilde{s}_m) = \max_{s_m \in \mathcal{S}_m \cup \bar{\mathcal{S}}_m} p(s^*, s_m).$$

Náhradné delenie $\tilde{s}_m$ predstavuje teda delenie na X_m, ktoré najviac predikuje výsledok delenia s^*. Na posúdenie kvality tejto predikcie definujeme prediktívnu mieru zhodnosti.

Definícia 4.4. *Prediktívnu mieru zhodnosti $\lambda(s^*|\tilde{s}_m)$ medzi s^* a $\tilde{s}_m$ definujeme ako*

$$\lambda(s^*|\tilde{s}_m) = \frac{min(p_L, p_R) - (1 - p(s^*, \tilde{s}_m))}{\min(p_L, p_R)}.$$

V prípade, že nastane $\lambda(s^*|\tilde{s}_m) \leq 0$, tak delenie s_m nemá dostatočnú predikčnú schopnosť pre delenie s^*, lebo platí $1 - p(s^*, s_m) \geq \min(p_L, p_R)$. Hodnota $\min(p_L, p_R)$ predstavuje odhad chyby nesprávneho zaradenia prípadu ak použijeme na predikciu delenia s^* delenia tvaru: Zaraď prípad do t_L ak $p_L = \max(p_L, p_R)$, inak do t_R.

Poznamenajme, že v prípade regresných stromov postupujeme takmer rovnako, jediný rozdiel je v odhadovaní pravdepodobnosti $p(t_L \cap t'_L)$, ktorú určime ako podiel počtu prípadov, ktoré zaradia obe delenia s^* a s_m naľavo a celkovému počtu N.

Na záver teda zhrnieme ako postupujeme v prípade chýbajúcich hodnôt vysvetľujúcich veličín. Predstavme si situáciu, keď pri určitom prípade chýba hodnota vysvetľujúcej veličiny X_{m_1}, podľa ktorej najlepšie nájdené delenie s^* rozhoduje kam prípad ďalej zaradiť. V tom prípade použijeme náhradné delenie $\tilde{s}_{m_2}$, $m_2 \in \{1, \ldots M\} - \{m_1\}$, pre ktoré $\lambda(s^*|\tilde{s}_{m_2})$ nadobúda maximálnu hodnotu. V prípade, že chýba aj hodnota veličiny X_{m_2} volíme druhé najlepšie. Ak žiadne z náhradných delení nedosahuje $\lambda(s^*|\tilde{s}_m) \geq 0$ tak prípad zaradíme do t_L ak $p_L = \max(p_L, p_R)$, v inom prípade do t_R.

4.6 Významnosť premenných

V prípade klasifikačných a regresných úloh si často kladieme otázku, ktoré veličiny sú najdôležitejšie. Nie vždy totiž platí, že premenná, ktorá sa vo výslednom strome vôbec nenachádza, teda žiadne použité deliace kritérium ju neobsahuje, je pre riešenie úlohy nedôležitá. Môže nastať situácia, keď sa veličina X_1 v strome nemusí nachádzať, ale pri odstránení veličiny X_2 a skonštruovaní nového stromu sa veličina X_1 v tomto strome objaví a nový strom bude takmer rovnako presný. Vtedy hovoríme, že nastal jav, ktorý nazývame maskovanie veličin. Maskovanie nastáva hlavne v prípade veličín, ktoré sú vzájomne vysoko korelované. Práve určovanie významnosti premenných predstavuje ďalšie využitie náhradných delení.

Nech T je výsledný skonštruovaný klasifikačný strom. Príslušné náhradné delenia

Definícia 4.5. *Miera významnosti vysvetľujúcej premennej X_m je definovaná ako*

$$M(x_m) = \sum_{t \in T} \Delta I(\tilde{s}_m, t),$$

kde $\Delta I(\tilde{s}_m, t)$ *predstavuje pokles nečistoty stromu pri použití delenia* $\tilde{s}_m$.

V prípade regresného stromu v definícii miery významnosti použijeme ΔR namiesto ΔI.

Základná myšlienka zisťovania významnosti veličín je nasledujúca. Nech delenie v uzle t prebieha podľa veličiny X_{m_1} a X_{m_2} je maskovaná X_{m_1}. Potom pre náhradné delenie $\tilde{s}_{m_2}$ k deleniu $s^*_{m_1}$ spôsobí takmer rovnaký pokles nečistoty ako najlepšie delenie $s^*_{m_1}$.

Kapitola 5

Riešené príklady

V tejto kapitole sa zameriame na praktické použitie algoritmu *CART*, ktorý sme popísali v predchádzajúcej kapitole. Pomocou dvoch klasifikačných a jednej regresnej úlohy sa zoznámime s knižnicou *rpart*, ktorá implementuje algoritmus *CART* v štatistickom programe *R*. Pri prvej úlohe, klasifikácii kvetov iris, popíšeme podrobnejšie aj niektoré funkcie knižnice *rpart*, preto bude trochu obsiahlejšia. V druhej úlohe budeme rozdeľovať pacientov do skupín podľa toho či nám pozorovania poukazujú alebo nepoukazujú na prítomnosť srdcovej choroby. V poslednej úlohe sa pokúsime predpovedať cenu automobilov. Táto časť práce je spracovaná pomocou [6].

5.1 Kvety iris

Ako prvú uvedieme úlohu klasifikácie kvetov iris. Učebný súbor dát obsahuje 150 záznamov o kvetoch troch rôznych druhov [1]. Klasifikáciu budeme prevádzať na základe rozmerov kališných a okvetných lístkov.
Spojité vysvetľujúce premenné:
- *Sepal.Length*... dĺžka kališného lístku
- *Sepal.Width*... šírka kališného lístku
- *Pedal.Length*... dĺžka okvetného lístku
- *Pedal.Width*... šírka okvetného lístku

Vysvetľovaná premenná:

[1] Použité dáta sa nachádzajú v archíve dát University of California, Irvine. http://archive.ics.uci.edu/ml/datasets.html, súbor dát Iris

- *Species*... jednotlivé druhy kvetov (*setosa, versicolor* a *virginica*)

Knižnica *rpart* implementuje algoritmus $CART$ a preto postup konštrukcie výsledného (optimálneho) stromu možno rozdeliť na dve časti. V prvej časti zostrojíme najprv rozsiahly strom T_{max}, ktorého postupným orezávaním získame postupnosť jeho podstromv. V druhej časti vyberáme z tejto postupnosti optimálny strom. Ako deliace kritérium je predvolené Giniho kritérium.

Po načítaní dát a aktivovaní knižnice sa pustíme do riešenia danej úlohy. Rozsiahly strom (nie nutne T_{max} ale len $T(\alpha)$) a postupnosť jeho podstromov získame pomocou funkcie *rpart*.

```
con1<-rpart.control(minsplit=2,minbucket=1,cp=0,xval=10,maxcompete=2,maxsurrogate=3)
T_0<-rpart(Species~Sepal.Length+Sepal.Width+Petal.Length+Petal.Width,data=iris,
 method="class", control=con1)
```

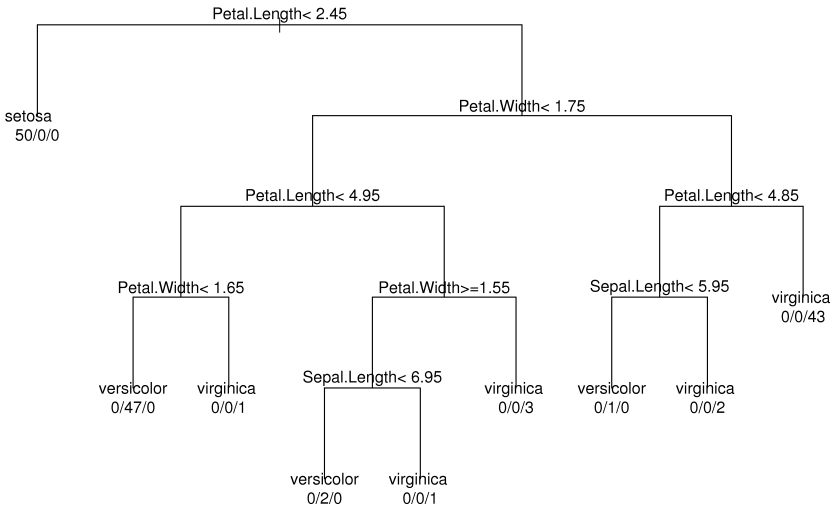
Funkcia *rpart()* má viacero parametrov. Pri jej volaní musíme minimálne doplniť formulu, t.j. určiť vysvetľovanú premennú a premenné, ktoré použijeme ako vysvetľujúce. Ak nie je z dát jasné o akú vysvetľovanú veličinu sa jedná, musíme to pomocou parametru *method* špecifikovať (v prípade klasifikácie uvádzame *"class"* v prípade regresie *"anova"*). Pomocou parametru *control* a funkcie *rpart.control()* možno okrem iného nastaviť pravidlo na zastavenie delenia (*minsplit,minbucket*), štartovaciu hodnotu parametru zložitosti (*cp*), určiť stupeň krížového overovania (*xval*). Pomocou parametru (*maxsurrogate*) určíme koľko náhradných delení chceme pre každý uzol počítať.

Vo výstupe programu R vidíme získaný strom. Pri každom uzle máme naznačené použité deliace kritérium, celkový počet a počet nesprávne klasifikovaných objektov v uzle a priradenú triedu. Hviezdička označuje koncový uzol stromu.

```
>T_0
n= 150

node), split, n, loss, yval, (yprob)
      * denotes terminal node

 1) root 150 100 setosa (0.33333333 0.33333333 0.33333333)
   2) Petal.Length< 2.45 50   0 setosa (1.00000000 0.00000000 0.00000000) *
   3) Petal.Length>=2.45 100  50 versicolor (0.00000000 0.50000000 0.50000000)
     6) Petal.Width< 1.75 54   5 versicolor (0.00000000 0.90740741 0.09259259)
      12) Petal.Length< 4.95 48   1 versicolor (0.00000000 0.97916667 0.02083333)
        24) Petal.Width< 1.65 47   0 versicolor (0.00000000 1.00000000 0.00000000) *
        25) Petal.Width>=1.65 1   0 virginica (0.00000000 0.00000000 1.00000000) *
      13) Petal.Length>=4.95 6   2 virginica (0.00000000 0.33333333 0.66666667)
```

```
   26) Petal.Width>=1.55 3   1 versicolor (0.00000000 0.66666667 0.33333333)
     52) Sepal.Length< 6.95 2   0 versicolor (0.00000000 1.00000000 0.00000000) *
     53) Sepal.Length>=6.95 1   0 virginica (0.00000000 0.00000000 1.00000000) *
   27) Petal.Width< 1.55 3   0 virginica (0.00000000 0.00000000 1.00000000) *
 7) Petal.Width>=1.75 46   1 virginica (0.00000000 0.02173913 0.97826087)
  14) Petal.Length< 4.85 3   1 virginica (0.00000000 0.33333333 0.66666667)
    28) Sepal.Length< 5.95 1   0 versicolor (0.00000000 1.00000000 0.00000000) *
    29) Sepal.Length>=5.95 2   0 virginica (0.00000000 0.00000000 1.00000000) *
  15) Petal.Length>=4.85 43   0 virginica (0.00000000 0.00000000 1.00000000) *
```

Pomocou príkazov *plot(),text()* možno strom $T(0)$ zobraziť graficky (obr. 5.1)

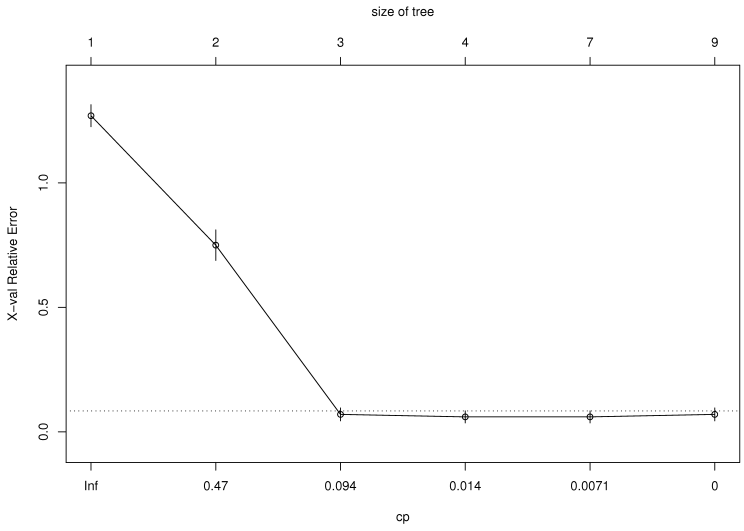


Obr. 5.1: Strom $T(0)$ pre úlohu Kvety iris

Vidíme, že výsledný strom má 9 listov a vzhľadom na počet premenných je dosť rozsiahly a obsahuje aj listy s jedným prípadom. Príkazom *printcp()* zobrazíme údaje o postupnosti podstromov. Nás budú hlavne zaujímať hodnoty *xerror* a *xstd*, ktoré predstavujú relatívnu odhadnutú chybu klasifikácie (vzhľadom k chybe klasifikácie koreňa) a odhad smerodajnej odchýlky. Vidíme, že odhadnutá chyba klasifikácie je minimálna pre strom so šiestimi deleniami. Podľa *1 SE* pravidla (mierne modifikovaného, keďže používame relatívnu chybu) ale vyberáme za optimálny strom s dvoma deleniami. Ten získame pomocou príkazu *prune(T_0,cp=...)* kde *cp* volíme z intervalu $(0.020, 0.440)$.

```
>printcp(T_0)
Classification tree:
rpart(formula = Species ~ Sepal.Length + Sepal.Width + Petal.Length +
    Petal.Width, data = iris, control = con1)

Variables actually used in tree construction:
[1] Petal.Length Petal.Width  Sepal.Length

Root node error: 100/150 = 0.66667

n= 150

     CP nsplit rel error xerror     xstd
1 0.500      0      1.00   1.27 0.044129
2 0.440      1      0.50   0.75 0.061237
3 0.020      2      0.06   0.07 0.025833
4 0.010      3      0.04   0.06 0.024000
5 0.005      6      0.01   0.06 0.024000
6 0.000      8      0.00   0.07 0.025833
```

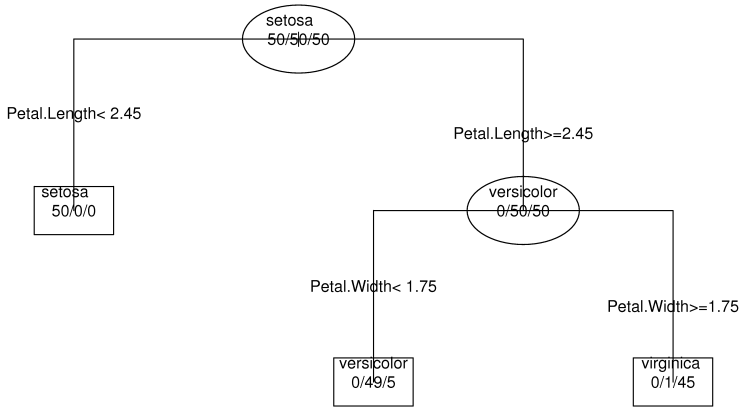
Zobrazenie veľkosti odhadnutej chyby v závislosti na počte listov stromu získame pomocou príkazu *plotcp()*. V obrázku 5.2, ktorý tak získame, je pomocou horizontálnej bodkovanej čiary naznačené aj *1 SE* pravidlo. Teda optimálny strom možno vybrať aj na základe získaného grafu.



Obr. 5.2: Grafické znázornenie odhadu chyby pre úlohu Kvety iris

Výsledný optimálny strom je znázornený na obrázku 5.3. Napriek jednoduchému tvaru správne klasifikuje 96% prípadov z učebného súboru dát. Testový odhad chyby klasifikácie je teda rovný 0.04. Relatívny odhad chyby

pomocou 10-násobného krížového overovania je rovný 0.07. Odhad chyby pomocou krížového overovania, ktorý získame vynásobením relatívneho odhadu chybou klasifikácie uzlu, je rovný $0.07 \times 0.66667 = 0.04667$. Jedná sa teda o kvalitný klasifikátor.



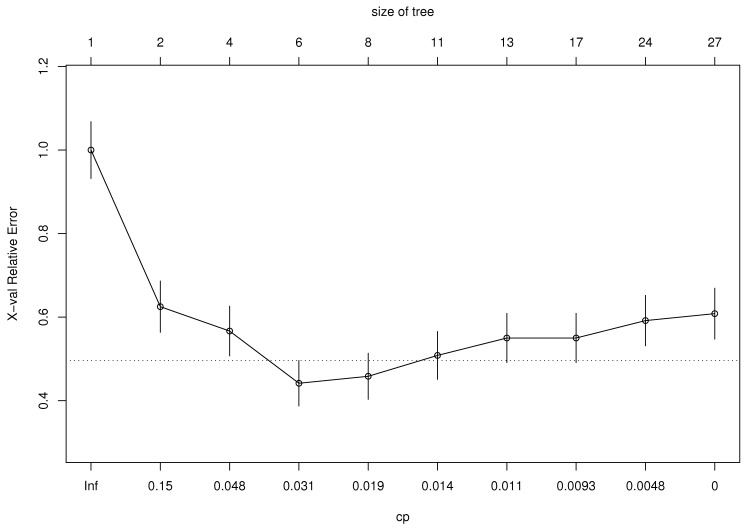
Obr. 5.3: Optimálny klasifikačný strom pre úlohu Kvety iris

5.2 Choroby srdca

V druhom príklade máme za úlohu zostrojiť klasifikačný strom, na základe ktorého bude možné rozhodovať, či pacient má chorobu srdca. K dispozícii máme 14 údajov o 270 pacientoch [2]. Model teda zostrojíme na základe súboru dát o veľkosti 270, ktorý obsahuje 13 vysvetľujúcih veličín *(x1,. . . ,x13)*. Na rozdiel od predchádzajúceho príkladu budú medzi vysvetľujúcimi veličiny aj veličiny kategoriálne. Navyše sme niekoľko hodnôt zo súboru vynechali aby sme mohli ukázať ako si s tým funkcia *rpart()* poradí.

Podobne ako v prvom prípade vytvoríme najprv rozsiahly strom $T(0)$ a pomocou *1 SE* pravidla vyberieme optimálny strom. Na základe závislosti odhadnutej chyby od počtu listov (obr. 5.4) za optimálny volíme strom so šiestimi listami.

[2]Použité dáta sa nachádzajú v archíve dát University of California, Irvine. http://archive.ics.uci.edu/ml/datasets.html, súbor dát Statlog (Heart)



Obr. 5.4: Grafické znázornenie odhadu chyby pre úlohu Choroby srdca

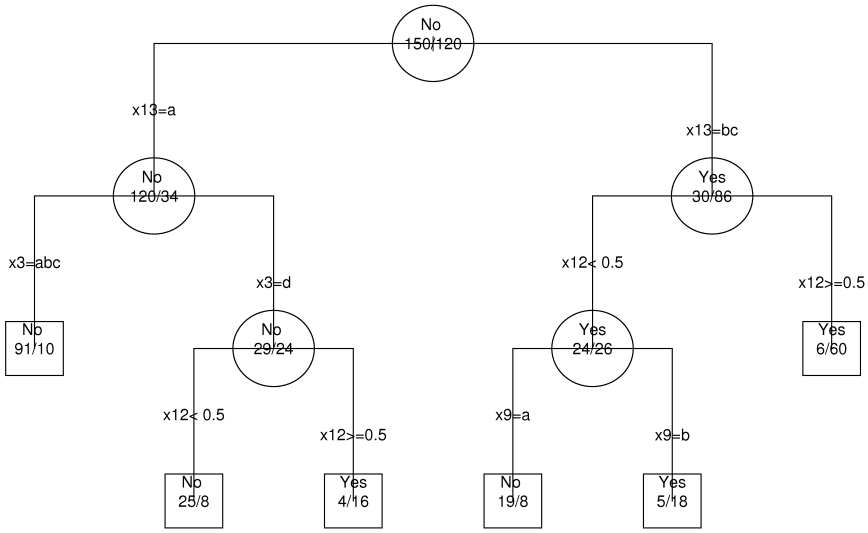
```
 1) root 270 120 No (0.5555556 0.4444444)
   2) x13=3 154  34 No (0.7792208 0.2207792)
     4) x3=1,2,3 101  10 No (0.9009901 0.0990099) *
     5) x3=4 53  24 No (0.5471698 0.4528302)
      10) x12< 0.5 33   8 No (0.7575758 0.2424242) *
      11) x12>=0.5 20   4 Yes (0.2000000 0.8000000) *
   3) x13=6,7 116  30 Yes (0.2586207 0.7413793)
     6) x12< 0.5 50  24 Yes (0.4800000 0.5200000)
      12) x9=0 27   8 No (0.7037037 0.2962963) *
      13) x9=1 23   5 Yes (0.2173913 0.7826087) *
     7) x12>=0.5 66   6 Yes (0.0909091 0.9090909) *
```

Zvolený optimálny strom je v zobrazený v textovej forme vo výpise. Pozrime sa teraz na daný strom podrobnejšie. Pomocou príkazu *summary()*, dostaneme detailný popis jednotlivých uzlov.

```
> summary(T_opt)
...
Node number 1: 270 observations,    complexity param=0.4666667
  predicted class=No   expected loss=0.4444444
    class counts:   150   120
   probabilities: 0.556 0.444
  left son=2 (154 obs) right son=3 (116 obs)
  Primary splits:
      x13 splits as  LRR, improve=36.11846, (7 missing)
  Surrogate splits:
      x8  < 150.5 to the right, agree=0.681, adj=0.263, (7 split)
      x9  splits as  LR,        agree=0.677, adj=0.254, (0 split)
      x10 < 1.55  to the left,  agree=0.669, adj=0.237, (0 split)
...
```

Vo výpise uvádzame informácie o uzle, ktorý je koreňom stromu. K dispozícii sú informácie o počte prípadov jednotlivých skupín v uzle a počte prípadov zaradených do jednotlivých synov. Ďalej máme informáciu o použitom (najlepšom) delení (*primary splits*) spolu s poklesom nečistoty aký spôsobí(*improve*). Ten je ale uvádzaný po vynásobení počtom prípadov. Pri náhradných delenich (*primary splits*) máme okrem iného uvádzané ako sa zhodujú s najlepším delením (*agree*). Pri každom delení navyše máme v zátvorke uvedený počet prípadov, pri ktorých sme použili dané delenie. Môžeme vidieť, že pri siedmich prípadoch chýba informácia o veličine $x13$, podľa ktorej koreň delíme. Pre tieto prípady preto používame náhradné deliace kritéria. Vidíme, že vo všetkých siedmich prípadoch bolo možné využiť najlepšie náhradné delenie, kedže boli prítomné informácie o veličine $x8$. Toto delenie sa zhoduje s najlepším delením v 68% prípadov.

Výsledný strom ešte znázorníme graficky (obr. 5.5)



Obr. 5.5: Optimálny klasifikačný strom pre úlohu Choroby srdca

Testový odhad chyby klasifikácie pre výsledný strom je 0.1519 a odhad chyby získaný pomocou krížového overovania je rovný 0.2037. Môžeme teda povedať, že pomocou daného stromu možno na 80% správne klasifikovať pacientov.

5.3 Cena áut

V poslednej úlohe sa pokúsime vysvetliť cenu automobilov na základe sledovaných veličín. Súbor dát sa obsahuje údaje o 205 automobiloch, pritom pri každom sledujeme nasledujúce veličiny[3]:

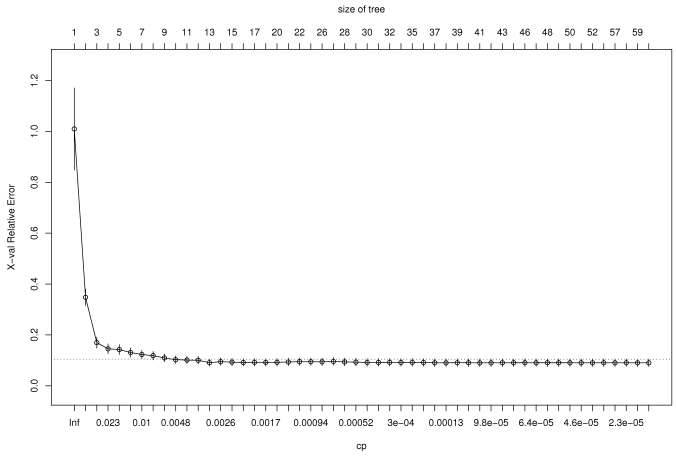
make ... výrobca automobilu (kategoriálna veličina, 20 kategórií)
f_type ... typ paliva (kategoriálna veličina, 2 kategórie)
doors ... počet dverí (kategoriálna veličina, 2 kategórie)
e_loc ... umiestnenie motoru (kategoriálna veličina, 2 kategórie)
w_base ... vzdialenosť kolies
length ... dĺžka automobilu
width ... šírka automobilu
height ... výška automobilu
c_weight ... váha automobilu
cylinders ... počet valcov (kategoriálna veličina, 7 kategórií)
e_size ... objem motoru
horsepower ... sila motoru
city_mpg ... spotreba v meste
higway_mpg ... spotreba mimo mesta

price ... cena automobilu v dolároch (vysvetľovaná veličina)

Postup riešenia je takmer totožný s postupom, ktorý sme použili pri riešení predchádzajúcich úloh. Najprv zostrojíme rozsiahly strom $T(0)$, ktorý má až 60 uzlov. Na základe tabuľky získanej príkazom *printcp()* a *1 SE* pravidla zvolíme podľa veľkosti odhadu relatívnej chyby (*xerror*) a príslušnej smerodajnej odchýlky (*xstd*) strom s 10 deleniami. Závislosť veľkosti relatívnej chyby stromu na počte delení je znázornené na obrazku 5.6.

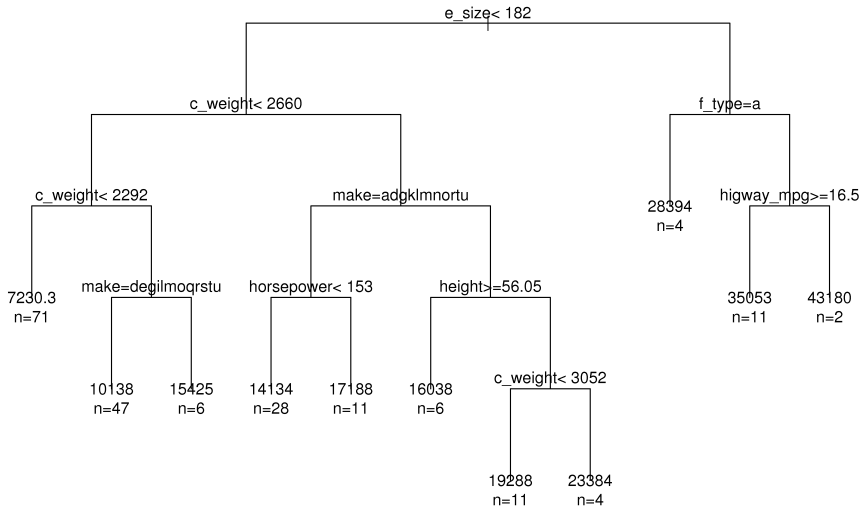
```
           CP nsplit rel error   xerror     xstd
1  6.6296e-01      0  1.000000 1.008459 0.160794
2  1.9064e-01      1  0.337043 0.345601 0.032155
3  2.9529e-02      2  0.146405 0.168262 0.021574
4  1.8561e-02      3  0.116876 0.142189 0.019269
....
10 3.8950e-03      9  0.050311 0.105109 0.014633
11 3.5082e-03     10  0.046416 0.099893 0.013980
12 3.3266e-03     11  0.042907 0.098351 0.013926
....
```

[3]Použité dáta sa nachádzajú v archíve dát University of California, Irvine. http://archive.ics.uci.edu/ml/datasets.html, súbor dát Automobile



Obr. 5.6: Grafické znázornenie odhadu chyby pre úlohu Cena áut

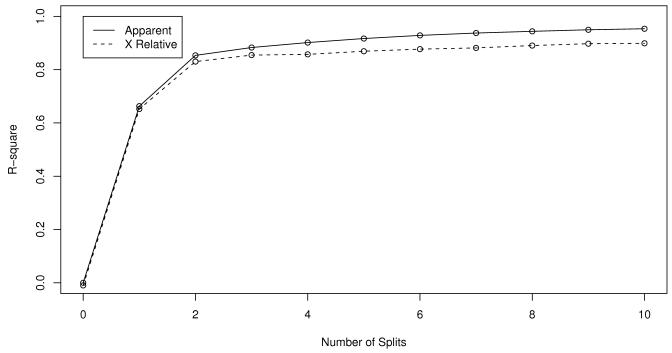
Výsledný optimálny strom (obr. 5.7) má teda už iba 11 uzlov, čiže oproti pôvodnému stromu je o dosť jednoduchší.



Obr. 5.7: Optimálny strom pre úlohu Cena áut

V prípade regresie o presnosti výsledného modelu najlepšie vypovedá koeficient determinácie. Ten získame jednoducho, odčítaním príslušných odhadov relatívnej štvorcovej chyby od čisla 1. Jeho hodnota pri použití testového odhadu chyby je $0,9536$ v prípade odhadu pomocou 10-násobného krížového

overovania 0.9000. Grafické znázornenie veľkosti oboch koeficientov a ich závislosti na počte listov získame pomocou príkazu *rsq.rpart()* (obr. 5.8).



Obr. 5.8: Závislosť koeficientu determinácie na počte delení

Záver

Cieľom tejto práce bolo predstaviť klasifikačné a regresné stromy, ktoré si v posledných rokoch získali veľkú popularitu. Hlavne vďaka jednoduchej interpretácii a nenáročnej práci s výsledným modelom si našli uplatnenie v mnohých odborných oblastiach.

V práci sme po zavedení základných pojmov detailne popísali jeden možný postup konštrukcie klasifikačných stromov, ktorý sme pomocou drobných úprav prispôsobili na konštrukciu regresných stromov. Ďalej sme predstavili najznámejšie algoritmy používané na konštrukciu stromov a jednému z nich, algoritmu *CART*, sme sa podrobne venovali. V závere sme potom popísanú teóriu využili pri riešení dvoch klasifikačných a jednej regresnej úlohy. Úlohy sme riešili v prostredí voľne šíriteľného štatistického programu *R* pomocou knižnice *rpart*.

Literatúra

[1] Antoch J. : *Klasifikace a regresní stromy.* Sborník ROBUST'88, 0-6, 1988.

[2] Breiman L., Friedman J.H., Olshen R.A., Stone C.J. : *Classification and regresion trees*, The Wadsworth Statistics/Probability Series, Belmont, CA : Wadsworth, 1984.

[3] Hawkins, D. M., *FIRM (Formal Inferencebased Recursive Modeling).* Technical Report Number 546, University of Minnesota, School of Statistics, 1999.

[4] Keprta S. : *Klasifikátory konstruované pomocí rekurzívních postupů.* MFF UK, Praha, Diplomová práca, 1993.

[5] Loh, W. Y. and Shih, Y. S. :. *Split selection methods for classification trees.* Statistica Sinica **7**, 815 - 840, 1997.

[6] Therneau T.M., Atkinson E.J. : *An introduction to recursive partitioning using the RPART routines.* Technical Report Series No. 61, Department of Health Science Research, Mayo Clinic, Rochester, Minnesota, 1997.

[7] Quinlan J. R. : *Induction of decision trees.* Machine Learning **1**, 81-106, 1986.