Příklad: Random Forest - Mořští krabi

Datový soubor obsahuje 200 měření a 8 prediktorů popisujících 5 morfologických měření 200 krabů (*Leptograpsus variegatus*) rozdělených do dvou barevných variant. Každá varianta je rovnoměrně zastoupena oběma pohlavími. Zajímá nás, zda je možné kraby pomocí morfologických měření rozlišit. Měření proběhlo na krabech nasbíraných v přístavním městě Fremantle v západní Austrálii.

Popis prediktorů:
sp - druh s kategoriemi *B* nebo *O* pro modrou a oranžovou variantu
*se*x - pohlaví kódované jako *M* a *F*
index - index 1:50 - id pozorování uvnitř čtyřech skupin *sp* x *sex*
FL - frontal lobe size (mm) - velikost čelního laloku
RW- rear width (mm) - šířka zadní části (zadečku)
CL - carapace length (mm) - délka krunýře
CW - carapace width (mm) - šířka krunýře
BD - body depth (mm) - výška trupu

Nejdříve načteme knihovnu MASS, která obsahuje výše popsaný datový soubor se jménem *crabs*:

```
> library(MASS)
> data(crabs)
> crabs
> summary(crabs)
```

Načteme knihovnu pro tvorbu náhodného lesa *randomForest*:

```
> library(randomForest)
```

V náhodném lese je možno nastavit hodnoty parametrů, které specifikují výsledný les a jeho výstupy. Defaultní nastavení hodnot je následující:

```
randomForest(x, y=NULL,
xtest=NULL, ytest=NULL, / testovací soubor není předem zadán;
ntree=500, / počet stromů v lese;
mtry=if (!is.null(y) && !is.factor(y))max(floor(ncol(x)/3), 1) else
floor(sqrt(ncol(x))), /počet náhodně vybraných proměnných; defaultní hodnoty jsou
```

$\sqrt{p}$ pro klasifikaci a $p/3$ pro regresi, kde p je počet prediktorů;

```
replace=TRUE /pozorování může být vybráno vícekrát (při procesu rozdělení na oob
vzorky);
classwt=NULL /váha jednotlivých kategorií závisle proměnné, defaultně mají všechny
stejnou váhu;
strata, sampsize = if (replace) nrow(x) else ceiling(.632*nrow(x)),
/parametry pro stratifikovaný výběr;
nodesize = if (!is.null(y) && !is.factor(y)) 5 else 1, /minimální počet
vzorků v terminálním uzlu, defaultně 1 pro klasifikaci a 5 pro regresi;
maxnodes = NULL, /maximální počet terminálních uzlů stromu;
importance=FALSE, /výpočet významnosti proměnných;
localImp=FALSE, /významnost každého pozorování;
```

`nPerm=1, /`počet iterací, kdy jsou *oob* pozorování permutovány pro výpočet *importance* proměnných, zatím pouze pro regresi;

`proximity, /`výpočet matice těsnosti;

`oob.prox=proximity, /`matice těsnosti pouze pro *oob* pozorování;

`norm.votes=TRUE, /`výsledné hlasování je vyjádřeno jako podíl, jinak přímo počet;

`do.trace=FALSE, /`zobrazí výstupy procesu hledání;

`keep.forest=!is.null(y) && is.null(xtest), corr.bias=FALSE, keep.inbag=FALSE, ...) /`FALSE – výsledek lesa nebude uložen ve finálním výstupu.

Nyní vytvoříme vlastní náhodný les. Začneme rozlišením dvou barevných variant krabů bez ohledu na pohlaví:

```
> set.seed(12)
> les_krabi <- randomForest(sp ~ FL+RW+CL+CW+BD, data=crabs,
importance=TRUE, proximity=TRUE)
```

Před každým spuštěním lesa je vhodné zadat jiné náhodné číslo *set.seed*, aby byla zajištěna větší „náhodnost“ algoritmu. V lese potřebujeme vybrat náhodně *oob* vzorky i počet prediktorů a algoritmus musí odněkud začít.

Do proměnné *les_krabi* uložíme výsledek náhodného lesa, který spustíme pomocí funkce *randomForest*, specifikujeme závisle proměnnou *sp* a prediktory z datového souboru *crabs*, zadáme výpočet významnosti (*importance*) proměnných a matici těsnosti (*proximity*). Zobrazíme výsledek výpočtu:

```
> print(les_krabi)
```

Zobrazíme výsledek pro prvních 150 stromů:

```
> set.seed(52)
> les_krabi <- randomForest(sp ~ FL+RW+CL+CW+BD, data=crabs, ntree=150)
> plot(les_krabi, cex.axis=1.5, cex.lab=1.5, lwd=2, col='black')
```

Další zajímavou informací je velikost výsledných stromů, kterou můžeme orientačně zjistit z histogramu počtu terminálních uzlů.

```
> hist(treesize(les_krabi))
```

Pomocí funkce *getTree* si můžeme zobrazit i vybraný strom. V tomto případě vybereme dvacátý. Výstup bude v textové podobě. Vzhledem k velké velikosti stromů však bývá často nepřehledný.

```
> getTree(randomForest(sp ~ FL+RW+CL+CW+BD, data=crabs, ntree=100), 20,
labelVar=TRUE)
```

Zbývá nastavit nejdůležitější proměnnou pro vytvoření lesa - počet náhodně vybraných proměnných (prediktorů), na základě kterých se budou jednotlivé stromy dělit. Zde je tento parametr označen jako *mtry*. Defaultní hodnoty parametru *mtry* jsou $\sqrt{p}$ pro klasifikaci a $p/3$ pro regresi, kde p je počet prediktorů. Nastavitelným parametrem je *improve.* Aby probíhalo další vyhledávání optimálního počtu prediktorů, musí být zlepšení *oob* chyby větší než jeho stanovená hodnota. Parametr *trace* zobrazí výstupy procesu hledání. Poněkud záludný je parametr *stepFactor*, který určuje, o kolik se bude parametr *mtry* zvyšovat (nebo snižovat) při testování jeho vlivu na výslednou klasifikaci.

```
> set.seed(126)
```

```
> mtryles <- tuneRF(crabs[,4:8], crabs[,1], stepFactor=1.5,
improve=0.05, ntree=100, mtry=4, trace=TRUE, plot=TRUE)
```

Pokud začneme vždy od různého náhodného čísla, výsledky se mohou lišit i při stejném nastavení parametrů funkce.

Nyní již můžeme spustit les s optimálním nastavením hodnot parametrů:

```
> set.seed(10)
> randomForest(formula = sp ~ FL + RW + CL + CW + BD, data = crabs,
importance = TRUE, proximity = TRUE, ntree = 100, mtry =3)
```

Poté, co jsme nastavili optimální hodnoty pro tvorbu lesa, můžeme použít další funkce, které jsou důležité zejména pro interpretaci našich výsledků.

Hodnoty významnosti proměnných zjistíme pomocí funkce *importance* a v grafické podobě je zobrazíme funkcí *varImpPlot*. U funkce *importance* můžeme zvolit významnost založenou na Gini indexu (*type* = 2) nebo na poklesu klasifikační chyby při randomizaci proměnné (*type* = 1).

```
> importance(les_krabi, type=2)
> varImpPlot(les_krabi)
```

Zobrazíme, kolikrát byly jednotlivé proměnné použité ve stromech při tvorbě lesa:

```
> set.seed(2)
> varUsed(randomForest(sp ~ FL + RW + CL + CW + BD, data=crabs,
ntree=100, mtry =3, by.tree=TRUE, count=TRUE))
```

Funkce *partialPlot* zobrazí graf efektu proměnné na pravděpodobnost kategorie. Do funkce zadáme výsledný les, proměnnou a kategorii (pomocí parametru `which.class`), pro kterou chceme znázornit průběh proměnné.

```
> set.seed(123)
> par(mfrow=c(3,2))
> partialPlot(les_krabi, crabs, FL, which.class= "B", cex.axis=1.2,
cex.lab=1.2, lwd=2)
> partialPlot(les_krabi, crabs, FL, which.class= "O", cex.axis=1.2,
cex.lab=1.2, lwd=2)
```

…atd.

Podobnou informaci můžeme získat z prototypů kategorií založených na matici těsnosti pomocí funkce *classCenter*. Získáme tak odhad hodnot proměnných, které jsou charakteristické pro danou barevnou variantu. Prototyp je medoid[1] z jeho nejbližších sousedů (z „reprezentativních“ pozorování) z příslušné kategorie. Můžeme porovnat hodnoty prototypů pro obě varianty.

```
> set.seed(22)
> les_krabi <- randomForest(sp ~ FL+RW+CL+CW+BD, data=crabs, mtry=3,
ntree=150, importance=TRUE, prox=TRUE)
> krabi_prototyp <- classCenter(crabs[,4:8], crabs[,1], les_krabi$prox)
> krabi_prototyp
```

[1] Obecněji medoid charakterizuje střed shluku objektů, přičemž jeho průměrná vzdálenost k ostatním objektům v tomto shluku je minimální.

Zobrazíme ještě vztah dvou nejvýznamnějších proměnných a jejich prototypů:

```
> plot(crabs[,4], crabs[,7], pch=21, xlab=names(crabs)[4],
ylab=names(crabs)[7],cex.axis=1.5, cex=1.5, cex.lab=1.5, bg=c("red",
"blue")[as.numeric(factor(crabs$sp))], main="Prototypy dvou variant
krabů")
> points(krabi_prototyp[,1], krabi_prototyp[,4], pch=21, cex=3,
bg=c("red", "blue"))
```

Kromě výpočtu prototypů použijeme matici těsnosti ve spojení s vícerozměrnými metodami, konkrétně ve Vícerozměrném škálování (MDS). Nastavíme funkci pro grafické znázornění první a druhé faktorové osy *MDSplot* s argumenty našeho optimálního lesa a závisle proměnné pro definici kategorií.

```
> MDSplot(les_krabi, crabs$sp, palette=rep(1,2),
pch=as.numeric(crabs$sp), cex.axis=1.5, cex=1.5, cex.lab=1.5)
```

Vzorky z obou kategorií se částečně překrývají (zejména v pravém horním rohu). Bylo by zajímavé zjistit, zda by rozdělení barevné varianty podle pohlaví mohlo odlišit tyto vzorky.

Podíváme se, jak se liší samičky a samečci nezávisle na barevné variantě.

```
> set.seed(51)
> les_krabi4 <- randomForest(sex ~ FL+RW+CL+CW+BD, data=crabs, mtry=3,
ntree=100, importance=TRUE, prox=TRUE)
> les_krabi4
```

rozlišení krabů podle pohlaví je stejně úspěšné jako podle barevné varianty.

```
> importance(les_krabi4, type=2) /typ 1 a 2
```

```
> krabi_prototyp1 <- classCenter(crabs[,4:8], crabs[,2],
les_krabi4$prox)
> krabi_prototyp1
```

rozdělení souboru na čtyři kategorie: kombinace pohlaví a barevné varianty.

```
> set.seed(321)
> les_krabi1 <- randomForest(TX$spsex ~ FL+RW+CL+CW+BD, data=crabs,
mtry=3, ntree=150, importance=TRUE, prox=TRUE)
> les_krabi1
> krabi_prototyp1
> varImpPlot(les_krabi1, cex =1.4)
```

Opět bychom mohli zkoumat efekt proměnných na klasifikaci a prototypy jednotlivých kategorií. Pro ostatní klasifikace je rovněž nutné znovu otestovat parametry *ntree* a *mtry*.

Nyní se podíváme, jak lze doplnit chybějící hodnoty pomocí náhodného lesa. K tomu je určena funkce *rfImpute*. Protože náš datový soubor žádné prázdné hodnoty neobsahuje, musíme nejdříve nějaké vytvořit. Dvacet vzorků z celkového souboru obsahujícího 200 pozorování nahradíme hodnotou *NA*.

```
> krabi_chybi <- crabs
> set.seed(111)
> for (i in 4:8) krabi_chybi[sample(200, sample(20)), i] <- NA
> krabi_chybi
```

Defaultní nastavení funkce *rfImpute* je pro pět iterací a 300 stromů. Nyní doplníme chybějící hodnoty na základě měření proximity:

```
> krabi_dopln <- rfImpute(sp ~ FL+RW+CL+CW+BD, data=krabi_chybi, iter=5,
ntree=300)
```

Výsledkem jsou hodnoty klasifikační chyby *oob* vzorků pro pět iterací.

Podíváme se, jak se změní procento chyby klasifikace, po doplnění 10% chybějících hodnot, od klasifikace bez chybějících hodnot.

```
> set.seed(15)
> randomForest(sp ~ FL + RW + CL + CW + BD, data = krabi_dopln,
importance = TRUE, proximity = TRUE, ntree = 150, mtry =3)
```

Poslední velmi užitečná funkce, kterou si vyzkoušíme, je pro predikci nových vzorků a jmenuje se příhodně *predict*. Opět můžeme nastavit parametry pro zobrazení výsledku. Parametr *type* nabízí tři možnosti: *response* – na výstupu budou uloženy predikované hodnoty, *prob* – výstupem bude matice pravděpodobností jednotlivých kategorií a *vote* – matici hlasování jednotlivých stromů. Dalším parametrem je *norm.votes*, pokud je nastaven na TRUE, počty hlasů budou vyjádřeny jako podíl kategorií (normované), jinak bude výstupem přímo počet hlasů. Nastavením parametru *predict.all* na *TRUE* bude uložena predikce všemi stromy, jinak pouze výsledné hlasování nebo průměrování.

Nejdříve si soubor rozdělíme na dva podsoubory, jeden použijeme pro tvorbu lesa a druhý soubor (s odstraněním závisle proměnné) použijeme pro predikci. Vytvoříme proměnnou s hodnotami 1 a 2 pro rozdělení souboru, poměr rozdělení udává parametr *prob*. V tomto případě rozdělíme soubor na 80% a 20% původního souboru.

```
> set.seed(112)
> index <- sample(2, nrow(crabs), replace = TRUE, prob=c(0.8, 0.2))
> krabi_lesP<- randomForest(sp ~ FL + RW + CL + CW + BD,
data=crabs[index == 1,], mtry=3, ntree = 150)
> krabi_pred <- predict(krabi_lesP, crabs[index == 2,], type="response",
norm.votes=TRUE, predict.all=FALSE, proximity=FALSE)
> krabi_pred
```

Výsledek predikce můžeme zobrazit v tabulce:

```
> table(observed = crabs[index==2, "sp"], predicted = krabi_pred)
```

Dále si můžeme zobrazit další varianty výsledků, predikce jednotlivými stromy a matici těsnosti (*proximity*) nebo pravděpodobnost zařazení všech pozorování do kategorie B a O:

```
> predict(krabi_lesP, crabs[index == 2,], predict.all=TRUE,
proximity=TRUE)
> krabi_pred <- predict(krabi_lesP, crabs[index == 2,], norm.votes =
FALSE)
> krabi_pred <- predict(krabi_lesP, crabs[index == 2,], type="prob")
```

Z pravděpodobností jednotlivých vzorků můžeme vybrat vzorky jednoznačně zařazené lesem (např. 18, 25, 55) a vzorky, u kterých zařazení do dané kategorie není tak snadné (44).