Medvěd a Pták představují…

PSM ULTIMATE v3

Zápočet:

- Dva až tři domácí úkoly – procvičíte si příklady ze cvičení.
- Seminární práce – vypracování komplexního statistického úkolu, kde výstupem je souvislý text.

Zkouška:

- Ústní zkouška u počítače.
- Vylosujete si metodu, kterou předvedete na příkladu a vysvětlíte.
- Jedna otázka na **mnohorozměrnou statistiku.**
- Jedna otázka na **regresní modely.**
- V případě nerozhodné známky doplňující otázka.

Doplňující otázky:

- Fuzzy modely
- Bayesovské sítě
- Věcná významnost

Co máme umět:

Zkouška:

- Jedna otázka na **mnohorozměrnou statistiku.**
    - Testy nemusíme
- Jedna otázka na **regresní modely.**
    - Lineární regrese
    - Mnohonásobná regrese
    - Logistická regrese
    - Loess regrese
    - Decision tree regrese
    - Random forest regrese
    - Support vector regrese
    - Ridge regrese
    - (Nelineární NEMUSÍME)

Doplňující otázky:

- Fuzzy modely
    - nemusí to být hluboký, hlavně že víme co to je k čemu to je a jak to funguje
    - ve stručnosti během jedné minuty vysvětlit princip a k čemu to je
- Bayesovské sítě
    - nemusí to být hluboký, hlavně že víme co to je k čemu to je a jak to funguje
    - ve stručnosti během jedné minuty vysvětlit princip a k čemu to je
- Věcná významnost
    - nemusí to být hluboký, hlavně že víme co to je k čemu to je a jak to funguje
    - ve stručnosti během jedné minuty vysvětlit princip a k čemu to je

Mnohorozměrná statistika

Praktické využití všech postupů mnohorozměrné statistiky na jednom místě i s kódy tady:
PSM Úkol

Legenda

- `factanal(prom)` -> kod
- #Bhahaha.. I -> Výstup kodu / dodatek ke kodu

Mnohorozměrná statistika obecně

- Nepracuje se s jednou proměnnou X, ale s vektorem proměnných X(X1, X2, …, Xn)
- Například: několik fyzických parametrů jedince (výška, váha, BMI, tlak, …)
- Nástroje:
  - Hotellingův test (mnohoroz. Dvouvýběrový test)
  - MANOVA (mnohoroz. ANOVA)
  - Kanonické korelace (mnohoroz. korelační koeficient)
  - Mnohorozměrná regrese
- Cíle:
  - redukovat data (hlavní komponenty)
  - seskupit pozorování (shluky)
  - najít rozdíly mezi skupinami (diskriminace).

Porovnání výběrů

- Zkoumáme, zda se **skupiny (např. muži vs. ženy)** liší v několika **závislých proměnných současně** (např. výška, váha, tlak).
- Nástroje:
  - Hotellingův test - porovnání 2 skupin
  - MANOVA - porovnání 3 a více skupin
  - Oboje snižuje šanci na chybu I. typu (false positive)

ANOVA

- **OPAKOVÁNÍ pro kontext**
- **AN**alysis **O**f **V**ariance
- Zjišťuje, jestli se průměry mezi více než dvěma skupinami významně liší
  - Jestli je rozdíl mezi skupinam
  - Jeden dataset aut - rozdělím na značky - kontroluji jestli je rozdíl ve výsledcích 1 proměnné ve skupinách
  - Funguje pouze pro 1 závislou proměnnou (to podle čeho rozdíl hodnotím)
- Pokud testuju pouze dvě skupiny -> t-test

MANOVA

- Rozšíření (zobecnění) ANOVY na více závislých proměnných.
- Testuje **globální hypotézu**, že **vektory středních hodnot jsou stejné mezi skupinami**.
    - V řeči lidí: nulová hypotéza = všechny skupiny jsou stejné
    - p > 0.05 = Skupiny jsou rozdílné
- Pokud testuju pro dvě skupiny -> Hotellingův test
- **Interakce:**
    - Zkoumá, zda kombinace dvou (nebo více) faktorů má společný vliv na více závislých proměnných jiný, než by měly tyto faktory samostatně.
- **Předpoklady:**
    - Multivariátová normalita
        - každá skupina má normální rozdělení ve všech závislých proměnných společně.
    - Homogenita matic kovariancí
        - rovnost kovariančních matic
        - všechny skupiny mají podobnou strukturu rozptylů a kovariancí.
    - Nezávislost pozorování
        - každé pozorování je nezávislé na ostatních.
    - Lineární vztahy mezi proměnnými
        - mezi závislými proměnnými by měl být lineární vztah.
    - Absence multikolinearity
        - závislé proměnné by neměly být silně korelované (jinak může být výsledek nestabilní).
- Vzorec z prezentace:
    -
    - **W (within):**

$$W = \sum_{i=1}^{p} \sum_{j=1}^{n_i} (Y_{ij} - \bar{Y}_i)^T (Y_{ij} - \bar{Y}_i)$$

    - **B (between):**

$$B = \sum_{i=1}^{p} n_i (\bar{Y}_i - \bar{Y})^T (\bar{Y}_i - \bar{Y})$$

    - **Popis:**
        - Y_{ij}: j-té pozorování v i-té skupině
        - $\bar{Y}_i$: průměr i-té skupiny
        - $\bar{Y}$: celkový průměr přes všechny skupiny
        - W: variabilita **uvnitř skupin** – jak moc se liší lidé ve stejné skupině
        - B: variabilita **mezi skupinami** – jak moc se liší průměry skupin

    - Pokud jsou rozdíly mezi skupinami (B) velké a rozdíly uvnitř skupin (W) malé -> skupiny se liší -> zamítáš H0 (takže skupiny jsou různé)

Využití v R

- Jak výška a kvalita masa závisí na rase prasete

```
manova_result <- manova(cbind(vyska, kvalita_masa) ~ rasa, data = data)
```

- Interakce druhu krmiva a rasy

```
manova(cbind(vyska, kvalita_masa) ~ rasa * krmivo, data = data)
```

- Výběr testů
    - Ve výchozím nastavení R použije Wilksovo lambda
    - Testy se liší matematickým výpočtem a citlivostí na porušení předpokladů (např. normalitu, homogenitu kovariance).
    - **Wilks**
        - Musí splňovat:
            - Multivariátová normalita
            - Homogenita kovariančních matic
            - Nezávislost pozorování
        - Nemusí splňovat:
            - Rovnoměrný počet pozorování ve skupinách (ale při nerovnosti může být ovlivněn)
            - Rovnoměrné rozdělení závislých proměnných (není klíčové, ale doporučeno)
    - **Pillai**
        - Musí splňovat:
            - Nezávislost pozorování
        - Nemusí splňovat:
            - Přesná multivariátová normalita (je robustní vůči mírnému porušení)
            - Homogenita kovariančních matic (je tolerantní vůči porušení)
            - Rovnoměrnost skupin (funguje dobře i při různých velikostech)
    - **Hotelling-Lawley**
        - Musí splňovat:
            - Multivariátová normalita
            - Nezávislost pozorování
        - Nemusí splňovat:
            - Homogenita kovariančních matic (je méně citlivý než Wilks, ale stále záleží na míře porušení)
            - Rovnoměrný počet pozorování (částečně tolerantní)
    - **Roy**
        - Musí splňovat:
            - Multivariátová normalita
            - Nezávislost pozorování
        - Nemusí splňovat:
            - Homogenita kovariančních matic (velmi citlivý na porušení)

- Vícero významných dimenzí rozdílu mezi skupinami (stačí silný rozdíl v jedné)

```
summary(manova_result, test = "Wilks")
```

- Testování předpokladů:

```
zavisla1 <- "Nonflavanoid.phenols"
zavisla2 <- "Proanthocyanin"
zavisla3 <- "Flavanoids"

# Multivariátová normalita
library(MVN)
mvn(data[, c(zavisla1 , zavisla2 , zavisla3 )], mvnTest = "royston", group = "Cultivar")

# Homogenita matic kovariancí
library(biotools)
boxM(data[, c(zavisla1 , zavisla2 , zavisla3 )], grouping = data$Cultivar)
#  Cultivar = odrůda

# Lineární vztahy mezi proměnnými
pairs(data[,c(zavisla1 , zavisla2 , zavisla3 )])
#  Hledám lineární vzory v grafech.

# Absence multikolinearity
# Hodnota blízká nule značí silnou multikolinearitu.
det(cov(data[, c(zavisla1 , zavisla2 , zavisla3 )]))
# Korelační matice (hodnota nad 0.3 už je nějak zajímavá)
cor(data[, c(zavisla1 , zavisla2 , zavisla3 )])
```

- Summary

```
# Máme data o výšce, váze a krevním tlaku mužů a žen:
data <- data.frame(
  pohlavi = rep(c("muž", "žena"), each = 5),
  vyska = c(180, 175, 178, 185, 182, 165, 160, 158, 162, 166),
  vaha = c(80, 82, 85, 90, 78, 60, 55, 58, 62, 65),
  tlak = c(120, 125, 130, 135, 122, 110, 112, 108, 114, 116)
)

manova_result <- manova(cbind(vyska, vaha, tlak) ~ pohlavi, data = data)
summary(manova_result)

##          Df  Pillai approx F num Df den Df    Pr(>F)
## pohlavi    1 0.93326   27.968      3      6 0.0006337 ***
## Residuals  8
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- Df Stupně volnosti: kolik “nezávislých informací” máme. Pro pohlaví je 1 (dvě skupiny - 1), pro Residuals (chyby) 8.
- Pillai: Pillaiho testové kritérium – hodnotí rozdíl mezi skupinami. Hodnoty blízké 1 značí velký rozdíl.
- approx F - Přibližná F-statistika – říká, jak velké rozdíly jsou mezi skupinami vůči rozptylu uvnitř skupin.
- num Df - Počet stupňů volnosti v čitateli (odpovídá počtu závislých proměnných).
- den Df - Počet stupňů volnosti ve jmenovateli – spojený s počtem pozorování.
- Pr(>F) p-hodnota – pravděpodobnost, že takový výsledek vznikl náhodně. Hodnota < 0.05 značí statisticky významný rozdíl.
    - Značka významnosti: * pro p < 0.05, ** pro p < 0.01, *** pro p < 0.001

Konkrétně:

1. pohlaví Df = 1 → Porovnáváš 2 skupiny: např. muži vs. ženy.
2. Pillai = 0.95 → Vysoká hodnota, znamená velký rozdíl mezi skupinami v kombinaci závislých proměnných.
3. approx F = 18.6 → Velká hodnota F → skupiny se výrazně liší.
4. num Df = 3 → Porovnáváš 3 závislé proměnné (např. výška, váha, tlak).
5. den Df = 6 → Odpovídá velikosti vzorku – máš málo dat (asi 10 pozorování).
6. Pr(>F) = 0.0023 → Velmi nízká p-hodnota → rozdíl je statisticky významný, nejedná se o náhodu.
7. → Závěr: Kombinace proměnných (např. výška, váha, tlak) se významně liší mezi pohlavími.

Hotellingův test

- T-test pro více závislých proměnných (zobecnění t-testu)
- Jako MANOVA pro 2 skupiny
- Předpoklady:
  - Stejné jako pro MANOVU
- Vzorec z prezentace:
  - $$T^2 = \frac{n_1 n_2}{n_1 + n_2}(\bar{X} - \bar{Y})^T \mathbf{S}^{-1}(\bar{X} - \bar{Y})$$
    $$\mathbf{S} = \frac{(n_1 - 1)\mathbf{S}_1 + (n_2 - 1)\mathbf{S}_2}{n_1 + n_2 - 2}$$
  - **Popis:**
    - $\bar{X}, \bar{Y}$: vektor průměrů obou skupin (např. průměrná délka, šířka, váha...)
    - S_1, S_2: kovarianční matice v každé skupině (popisuje rozptyl a vzájemné vztahy mezi proměnnými)
    - $\mathbf{S}$: tzv. **spojená kovarianční matice** – vážený průměr obou
    - $(\bar{X} - \bar{Y})^T \mathbf{S}^{-1}(\bar{X} - \bar{Y})$: kvadratická forma – měří vzdálenost mezi průměry, s **přihlédnutím k rozptylu**
  - Chceš vědět, jak moc se skupiny liší průměrně ve vícero proměnných najednou.
  - Porovnáváš průměry, ale zároveň zohledňuješ variabilitu uvnitř skupin (rozptyl).
  - Větší T^2 = větší rozdíl mezi skupinami → pokud je „příliš velký“, zamítáš H0 (takže nejsou stejné).

Využití v R

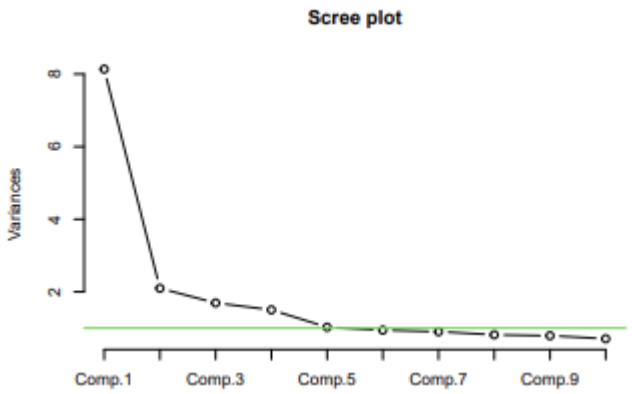
```
library(Hotelling)
# Pokud má Cultivar 2 hodnoty
hotelling.test(data[, c(zavisla1, zavisla2, zavisla3)] ~ data$Cultivar)
```

```
## Test stat:  121.79
 ## Numerator df:  2
 ## Denominator df:  3
 ## P-value:  0.005671
```

1. Test stat: Hotellingova testová statistika (čím větší, tím větší rozdíl)
2. Numerator df: Počet proměnných, které byly zahrnuty do testu (muži, ženy
3. Denominator df: Odhad stupňů volnosti ve zbytku rozptylu (zohledňuje velikosti vzorku obou skupin).
4. p-value Pravděpodobnost, že rozdíl vznikl náhodou (p < 0.05 = významné)

Redukce dimenzionality

Metoda hlavních komponent (PCA)

- Zredukovat rozměrnost vícerozměrných dat.
- Najít nové proměnné (**hlavní komponenty**), které vysvětlují co nejvíce variability v datech.
- Vstupní proměnné musí být nezávislé
- Hlavní využití při grafickém zobrazení výstupů
- Vytváří nové proměnné lineární kombinací původních (práce s maticemi)
- Výsledná matice hlavních komponent Y má tyto vlastnosti:
    - vektory jsou vzájemně nezávislé
    - součet koeficientů lineární transformace u každé komponenty je 1
    - řadí se podle velikosti variability (od největší k nejmenší)
    - **obsahuje veškeré informace, které obsahovala původní data**
- Postup PCA:
    1. mějme mnohorozměrná data v prostoru
    2. daty proložíme vektor ve smeru s největší variabilitou
    3. tak získáme první hlavní komponentu
    4. hledáme vektor, který by byl k prvnímu kolmý a opět byl ve směru s největší variabilitou
    5. získáme druhou hlavní komponentu
    6. hledáme vektor, který by byl kolmý k prvním dvěma a byl
    7. ve směru s největší variabilitou
    8. získáme třetí hlavní komponentu
    9. poslední dva kroky opakujeme, dokud máme body ve volném prostoru
- Optimální počet hlavních komponent
    - počet hlavních komponent k reprezentaci informace původních dat = počet vlastních čísel korelační matice větších než 1
    - Screeplot - tady jde vidět že je počet komponent 2, 3 nebo 4
        - Čára = kaiserovo kritérium = hledám komponenty které mají eigen value větší než 1
        - Vybírám buď Kaiserovým kritériem nebo tam kde se láme loket (klesání není tak a výrazné a začíná být stále
        - Vzhledem k loktu bych rozhodl pro comp 2 nebo 3
    - 

- Nevýhoda:
    - **Proměnné nemají přirozenou interpretaci.**
    - Pokud chceme menší počet proměnných, které jsou interpretovatelné -> **faktorová analýza**

Využívá:

- **Korelační (nebo kovarianční) matici** původních proměnných.
- **Vlastní čísla (eigenvalues)** – určují význam (variabilitu) jednotlivých komponent.
    - Vyjadřují **variabilitu vysvětlenou každou hlavní komponentou (PC)**. Čím větší je vlastní číslo, tím více informace (rozptylu) daná komponenta vysvětluje.
    - Seřazeny sestupně – první PC vysvětluje nejvíce variability.
- **Vlastní vektory (eigenvectors)** – určují směr hlavních komponent (tzv. **loadingy**).
    - Obsahují **koeficienty (loadingy)** pro každou původní proměnnou.
    - Každý **sloupec** v matici vlastních vektorů odpovídá jedné **hlavní komponentě**.
    - Každý **řádek** odpovídá **původní proměnné**.

Využití v R

Příprava dat

```
v1 <- c(1,1,1,1,1,1,1,1,1,1,3,3,3,3,3,4,5,6)
v2 <- c(1,2,1,1,1,1,2,1,2,1,3,4,3,3,3,4,6,5)
v3 <- c(3,3,3,3,3,1,1,1,1,1,1,1,1,1,1,5,4,6)
v4 <- c(3,3,4,3,3,1,1,2,1,1,1,1,2,1,1,5,6,4)
v5 <- c(1,1,1,1,1,3,3,3,3,3,1,1,1,1,1,6,4,5)
v6 <- c(1,1,1,2,1,3,3,3,4,3,1,1,1,2,1,6,5,4)
vmat <- data.frame(v1,v2,v3,v4,v5,v6)
m1 <- cbind(v1,v2,v3,v4,v5,v6)
```

Korelace

```
cor(m1)
```

```
##          v1        v2        v3        v4        v5        v6
## v1 1.0000000 0.9393083 0.5128866 0.4320310 0.4664948 0.4086076
## v2 0.9393083 1.0000000 0.4124441 0.4084281 0.4363925 0.4326113
## v3 0.5128866 0.4124441 1.0000000 0.8770750 0.5128866 0.4320310
## v4 0.4320310 0.4084281 0.8770750 1.0000000 0.4320310 0.4323259
## v5 0.4664948 0.4363925 0.5128866 0.4320310 1.0000000 0.9473451
## v6 0.4086076 0.4326113 0.4320310 0.4323259 0.9473451 1.0000000
```

#Tato matice nám ukáže, jak jsou jednotlivé proměnné vzájemně korelovány. Vysoké hodnoty (např. 0.9) naznačují, že proměnné se silně ovlivňují.

Vlastní vektory a čísla

```
eigen(cor(m1))
```

```
## eigen() decomposition
## $values
## [1] 3.69603077 1.07311448 1.00077409 0.16100348 0.04096116 0.02811601
##
## $vectors
##          PC1         PC2         PC3         PC4         PC5         PC6
## [v1,] 0.4154985  0.53088297 -0.1760717  0.2791358 -0.5317514  0.39223298
## [v2,] 0.4007058  0.54223870 -0.2485226 -0.3048547  0.5042931 -0.36932463
## [v3,] 0.4133938 -0.07418871  0.5496063  0.5693303  0.4344463  0.09302655
## [v4,] 0.3940548 -0.08433475  0.5976225 -0.5877130 -0.3543977 -0.09721936
## [v5,] 0.4206885 -0.44028459 -0.3342420  0.2798686 -0.2920358 -0.59484588
## [v6,] 0.4045287 -0.46655507 -0.3691854 -0.2850910  0.2516003  0.58121033
```

PC -> hlavní komponenta *v1,2,3..-> původní proměnné*

číslo -> vlastní číslo *sloupec -> vlastní vektor*

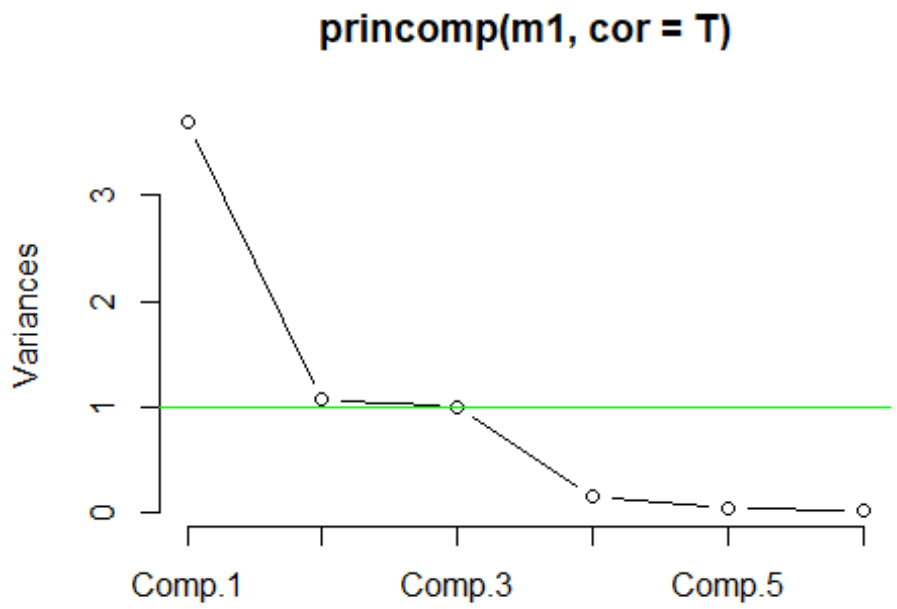
```
          PC1
## [v1,] 0.4154985
## [v2,] 0.4007058
## [v3,] 0.4133938
## [v4,] 0.3940548
## [v5,] 0.4206885
## [v6,] 0.4045287
```

Všechny tyto hodnoty jsou pozitivní, což znamená, že první hlavní komponenta je pozitivně korelována s každou z těchto proměnných (v1, v2, v3, v4, v5, v6).

Jinými slovy, pokud se hodnota v jedné z těchto proměnných zvyšuje, hodnota první hlavní komponenty se také zvyšuje.

Ukázka variability na grafu:

```
screeplot(princomp(m1, cor = T), type="l")
abline(h=1, col="green")
```



 # dostatecne velke procento vyuzite variability (80%)
 # první hlavní komponenta má velké množství informace (Koukám se nad zelenou čáru, vidím, že využiji 2 - 3 hlavní komponenty

```
cumsum(eigen(cor(m1))$values / sum(eigen(cor(m1))$values))
```

```
## [1] 0.6160051 0.7948575 0.9616532 0.9884871 0.9953140 1.0000000
```

první komponenta 61%..

první a druhá 79%

prvni 3 komponenty vysvetli pres 90% variability

-> obvykle se potřebuje min 80%.. takže volíme 3 komponenty:

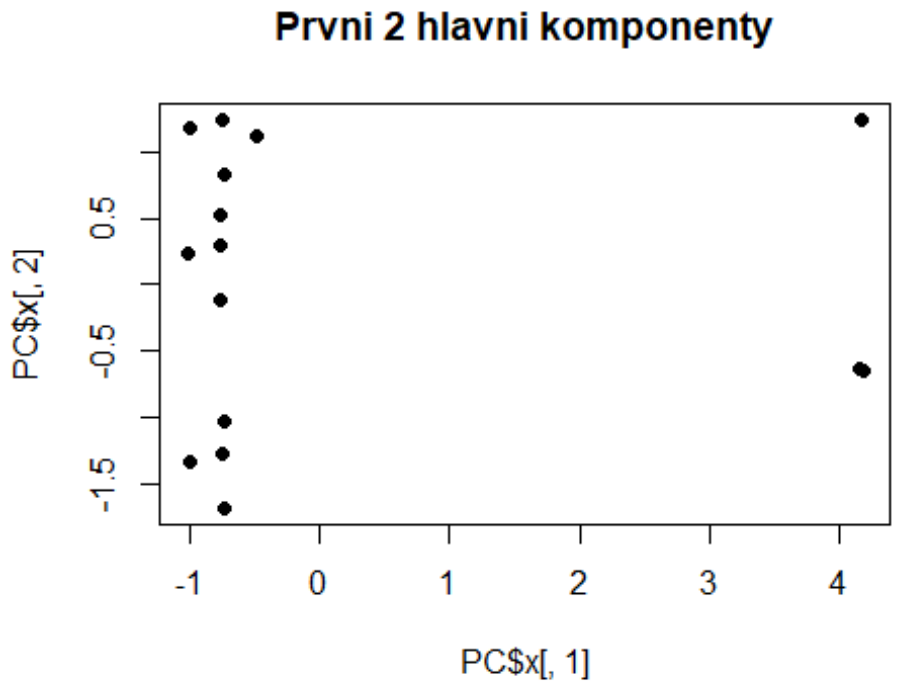
Vytvoření hlavních komponent: (Dělá to “samé” jako to nad tím)

```
prcomp(m1)
```

```
## Standard deviations (1, .., p=6):
 ## [1] 3.0368683 1.6313757 1.5818857 0.6344131 0.3190765 0.2649086
 ##
 ## Rotation (n x k) = (6 x 6):
 ##         PC1         PC2         PC3         PC4         PC5         PC6
 ## v1 0.4168038 -0.52292304  0.2354298 -0.2686501 -0.5157193  0.39907358
 ## v2 0.3885610 -0.50887673  0.2985906  0.3060519  0.5061522 -0.38865228
 ## v3 0.4182779  0.01521834 -0.5555132 -0.5686880  0.4308467  0.08474731
```

```
## v4 0.3943646  0.02184360 -0.5986150  0.5922259 -0.3558110 -0.09124977
## v5 0.4254013  0.47017231  0.2923345 -0.2789775 -0.3060409 -0.58397162
## v6 0.4047824  0.49580764  0.3209708  0.2866938  0.2682391  0.57719858
```

```
PC <- prcomp(vmat, scale = T)
  # hlavni komponenty
  # vrati variabilitu hlavnich komponent spolu s koeficienty jednotlivych komponent
plot(PC$x[,1], PC$x[,2], pch = 19, main = "Prvni 2 hlavni komponenty")
```



vykresleni prvnich dvou hlavnich komponent, ukazuji v datech skupiny

maji hlavni komponenty prirozenou interpretaci?
mnohdy ne, pak je potreba pouzit **faktorovou analyzu**

Faktorová analýza

- **Statistická metoda** používaná ke zjednodušení dat – místo mnoha proměnných (sloupců) hledá několik **skrytých faktorů** (tzv. latentních).
- Pomáhá zjistit, **co mají proměnné společného** – najít „neviditelné“ příčiny, které ovlivňují více proměnných najednou.
- Řeší problém PCA s interpretací.
- Snaží se vysvětlit pouze společnou variabilitu (common variance) mezi původními proměnnými.
- Hlavní myšlenka faktorové analýzy pochází z psychologie.
    - Spočívá v předpokladu, že na každého člověka působí "k" neměřitelných faktorů.
    - Na základě toho, jak tyto faktory na nás působí, reagujeme.
    - Cílem je pak podle našich reakcí na "p" podnětů (nebo proměnných) identifikovat původní, skryté faktory.

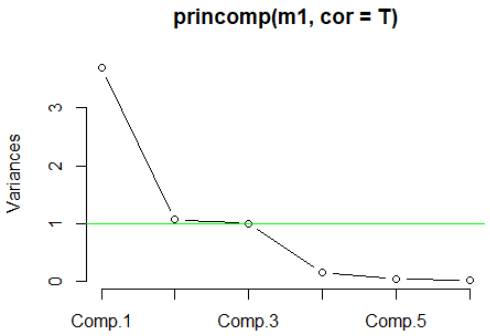
Předpoklady:

1. Spojitá, intervalová nebo poměrová data
2. Chyby jsou náhodné, nezávislé a mají konstantní rozptyl
3. Vhodnost dat: Proměnné spolu korelují.. řešilo se nahoře, jinak to nemá smysl
4. Počet faktorů je správně zvolen: Musíš zvolit takový počet faktorů, aby vysvětlili většinu variability (stejným způsobem jako u PCA)
5. Normalita (Pro metody factanal)

Využití v R

```
factanal(m1, factors = 3)
```

Počet faktorů mi znázorňuje screeplot z eigen v PCA.. kolik jich je větších než 1

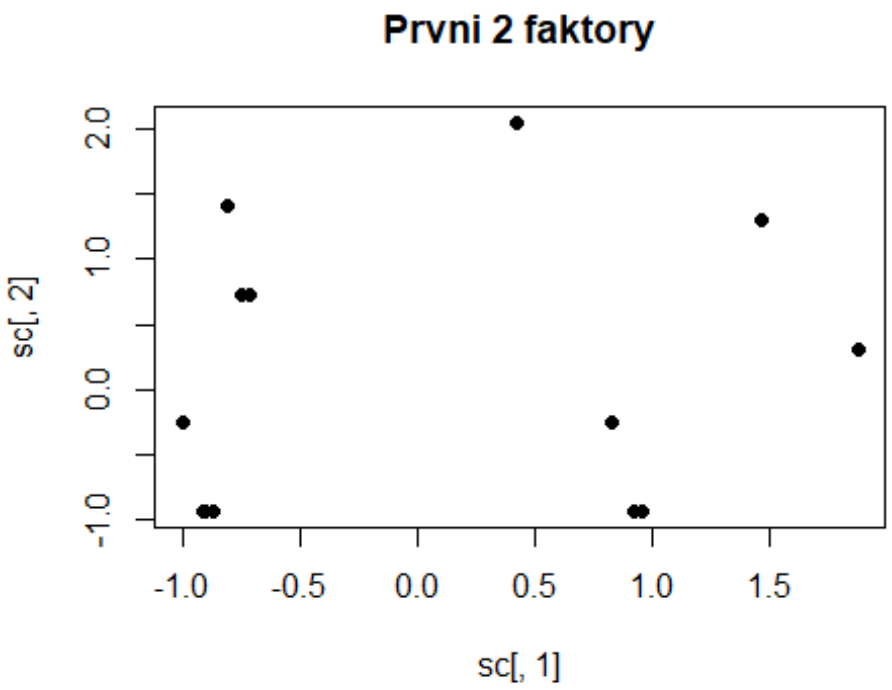



```
## Call:
 ## factanal(x = m1, factors = 3)
 ##
 ## Uniquenesses:
 ##    v1          v2        v3    v4v5        v6
 ## 0.005 0.101 0.005 0.224 0.084 0.005
 ##
 ## Loadings:
 ##    Factor1 Factor2 Factor3
 ## v1 0.944   0.182   0.267
 ## v2 0.905   0.235   0.159
 ## v3 0.236   0.210   0.946
 ## v4 0.180   0.242   0.828
 ## v5 0.242   0.881   0.286
 ## v6 0.193   0.959   0.196
```

```
##
##                Factor1 Factor2 Factor3
## SS loadings      1.893   1.886   1.797
## Proportion Var   0.316   0.314   0.300
## Cumulative Var   0.316   0.630   0.929
##
## The degrees of freedom for the model is 0 and the fit was 0.4755
```

1. Uniquenesses: jaké proměnné jsou zajímavé, obsaženy v analýze.. nízká čísla good (0.1). Pokud velká, může se vynechat (Nekoreluje s ostatními)

```
sc <- factanal(~v1+v2+v3+v4+v5+v6, factors = 3, scores = "Bartlett")$scores
 faktorove skory pro jednotliva pozorovani
 plot(sc[,1], sc[,2], pch = 19, main = "Prvni 2 faktory")
```



 # vykresleni prvnich dvou faktoru

Graf zobrazuje rozložení pozorování podle prvních dvou faktorů.

Blízká pozorování = pozorování, která jsou si podobná ve faktorech. Pokud jsou body u sebe, znamená to, že mají podobné hodnoty faktorových skóre → tedy jsou si podobná ve významných rysech, které faktory vystihují.

Vzdálená pozorování = Hodně rozptýlené body znamenají, že faktory dobře rozlišují pozorování. Např. některé body mají vysokou hodnotu na prvním faktoru (sc[,1] blízko 1.5), jiné zápornou (např. -1.0). To značí, že první faktor silně diferencuje pozorování.

Lze hledat shluky - seskupení bodů. To indikuje latentní skupiny (např. typy respondentů, podobné odpovědi na otázky, podobné vzorce chování atd.)

Příklad interpretace: Pozorování vpravo nahoře (např. [1.5, 1.5]) má vysoké skóre v obou faktorech → typický profil "silně ovlivněn oběma faktory". Pozorování vlevo dole ([-1, -1]) má nízké skóre v obou faktorech → opačný profil. Pozorování kolem [0,0] jsou průměrná, nemají extrémní hodnoty ani v jednom faktoru.

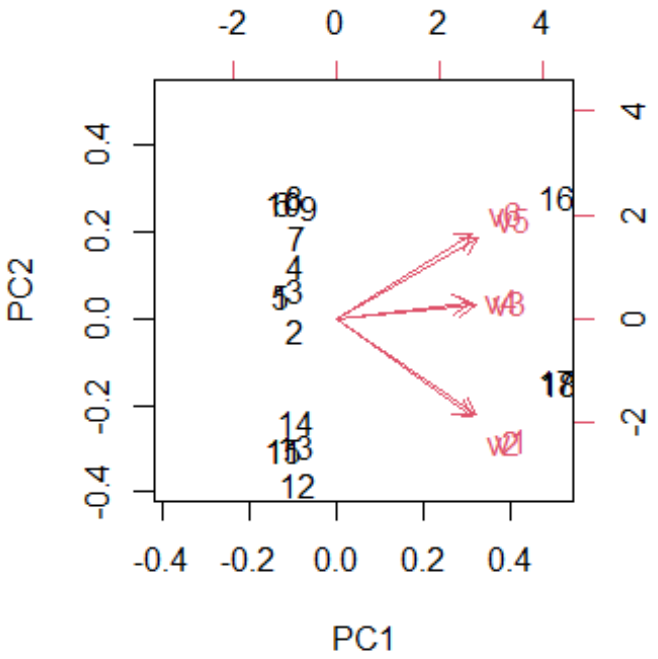
```
summary(PC)
```

```
## Importance of components:
##                           PC1    PC2    PC3     PC4     PC5     PC6
## Standard deviation     1.923 1.0359 1.0004 0.40125 0.20239 0.16768
## Proportion of Variance 0.616 0.1789 0.1668 0.02683 0.00683 0.00469
## Cumulative Proportion  0.616 0.7949 0.9617 0.98849 0.99531 1.00000
```

Interpretace:

1. Standard deviation: Odchylka každé hlavní komponenty (PC1 až PCn) – čím větší, tím víc informací komponenta obsahuje.
2. Proportion of Variance Podíl vysvětlené variability – kolik % informací o původních datech tato komponenta zachycuje.
3. Cumulative Proportion Kumulovaný podíl – součet všech předchozích Proportion of Variance → ukazuje, kolik % informace získáme, když si vezmeme např. první 2 komponenty.

```
biplot(PC)
```



1. Čísla v grafu označují pozorování
2. Šipky znázorňují původní proměnné - Směr šipky -> směr růstu proměnné - Délka šipky -> síla vlivu této proměnné

Silně přispívají do komponenty PC1. Pozorování, která leží vpravo, mají pravděpodobně větší hodnoty těchto atributů.

Diskriminační analýza

- Diskriminační analýza je metoda, která se používá k rozeznávání skupin (kategorií) na základě několika číselných proměnných.
- Představ si třeba, že máš vzorky vína ze tří různých odrůd.
    - Každý vzorek je popsán několika číselnými hodnotami (např. kyselost, obsah cukru, barva).
    - Ty ale nevíš, jaká odrůda to je – a právě diskriminační analýza se snaží na základě těchto čísel poznat, do které skupiny (odrůdy) víno patří.
- **Lineární diskriminační analýza:**
    - Vysvětlení budeme pokračovat s příkladem s víny (zadání úkolu PSM)
    - Hledá rozhodovací pravidla, která co nejlépe rozliší odrůdy vína (kategorie) podle chemických vlastností (čísla).
    - LDA hledá lineární kombinace původních proměnných, které co nejlépe oddělí skupiny.
    - Kombinace se jmenují LD1, LD2 (Linear Discriminant)
    - Jsou to osy rozhodování – zjednodušeně: čím dál je vzorek na LD ose od jiných skupin, tím lépe ho odliší.
    - Např. LD1 může rozdělovat třídy 1 vs 2+3, LD2 třeba 2 vs 3.
    - Jak funguje:
        - Změří průměry každé skupiny v každé proměnné.
        - Vypočítá rozptyl uvnitř skupin a mezi skupinami.
        - Hledá směry, které maximalizují rozdíl mezi skupinami a minimalizují rozdíl uvnitř skupin.
    - Je to model, který z venku působí jako regresní model - musí se natrénovat na části dat a pak předpovídá pro zbytek
    - Předpoklady:
        - awa
    - Počet LD os (funkcí) = Max. počet skupin - 1.
        - Např. máš 3 odrůdy → max. 2 LD funkce (LD1 a LD2).
    -

Příklad v R

```
library(MASS)
 head(iris)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa
```

Chceme zjistit, jestli lze předpovědět druh květiny (Species) na základě těchto měření.

Naučení modelu:

```
model <- MASS::lda(Species ~ ., data = iris)
 summary(model)
```

```
##         Length Class  Mode
 ## prior    3    -none- numeric
 ## counts   3  -none- numeric
 ## means    12          -none- numeric
 ## scaling  8   -none- numeric
 ## lev      3     -none- character
 ## svd      2    -none- numeric
 ## N        1     -none- numeric
 ## call     3     -none- call
 ## terms    3   terms  call
 ## xlevels  0   -none- list
```

Tím říkáme: použij všechny proměnné kromě Species k předpovědi Species.

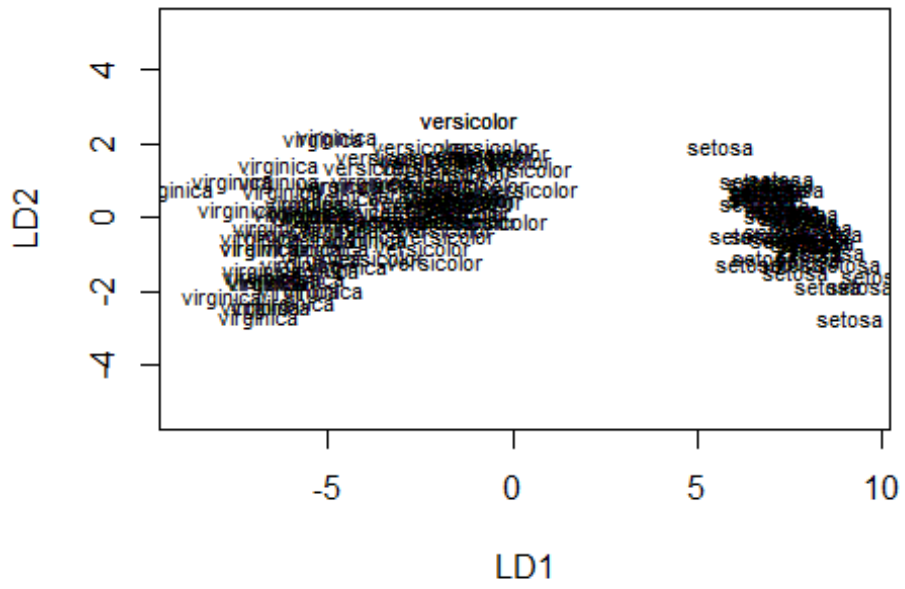
```
model
```

```
## Call:
 ## lda(Species ~ ., data = iris)
 ##
 ## Prior probabilities of groups:
 ##     setosa versicolor  virginica
 ##  0.3333333  0.3333333  0.3333333
 ##
 ## Group means:
 ##            Sepal.Length Sepal.Width Petal.Length Petal.Width
 ## setosa            5.006       3.428        1.462       0.246
 ## versicolor        5.936       2.770        4.260       1.326
 ## virginica         6.588       2.974        5.552       2.026
 ##
 ## Coefficients of linear discriminants:
 ##                 LD1     LD2
 ## Sepal.Length  0.8293776 -0.02410215
 ## Sepal.Width   1.5344731 -2.16452123
 ## Petal.Length -2.2012117  0.93192121
 ## Petal.Width  -2.8104603 -2.83918785
 ##
 ## Proportion of trace:
 ##    LD1       LD2
 ## 0.9912 0.0088
```

1. Prior probabilities: máme rovnoměrné zastoupení druhů (každý tvoří třetinu dat).
2. Group means: průměrné hodnoty jednotlivých měření pro každý druh. Např. setosa má nejmenší okvětní lístek (Petal.Length).
3. LD1, LD2 Diskriminační osy (směry) pro oddělení skupin

4. Coefficients of linear discriminants: Jak silně daná proměnná přispívá k oddělování skupin podél daného směru.
5. Proportion of trace: Kolik % rozptylu mezi skupinami každá osa vysvětluje

```
plot(model)
```



LD1 pěkně rozděluje 3 skupiny (hlavně odděluje Setosu od ostatních),

LD2 je víceméně jen do šířky a nepřináší tolik rozlišení.

Předpoveď:

```
pred <- predict(model)
 head(pred$class)
```

```
## [1] setosa setosa setosa setosa setosa setosa
 ## Levels: setosa versicolor virginica
```

```
table(Předpověď = pred$class, Skutečnost = iris$Species)
```

```
##                Skutečnost
 ## Předpověď setosa versicolor virginica
 ##   setosa         50         0         0
 ##   versicolor      0         48        1
 ##   virginica       0         2         49
```

Model perfektně pozná Setosu (50/50) U Versicolor a Virginica udělá pár chyb (2 kusy si spletl) Celková úspěšnost: (50 + 48 + 49) / 150 = 98 %

Shluková analýza (Clustering)

- Shluková analýza je metoda, která automaticky hledá skupiny (shluky) podobných objektů
- Narozdíl od diskriminační analýzy, tady nevíme, kolik skupin existuje.

Hierarchické shlukování

Zaměřuje se na to, jakým způsobem lze spojovat objekty do skupin. Existují dvě hlavní varianty:

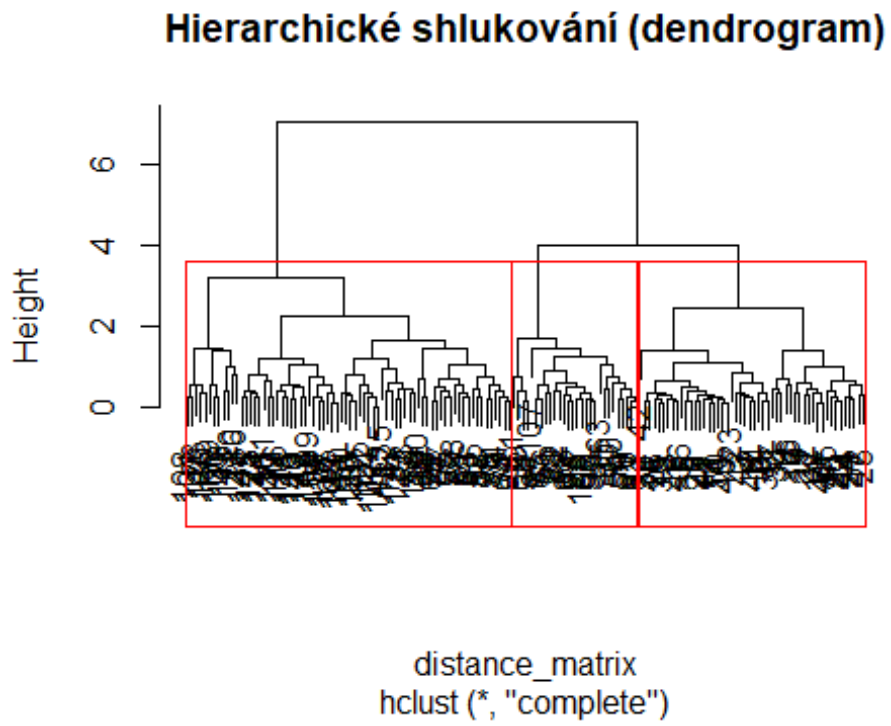
Agregativní (bottom-up): Začínáme se všemi objekty jako samostatnými shluky a postupně je spojujeme do větších shluků.

Dezregativní (top-down): Začínáme s jedním velkým shlukem, který postupně dělíme na menší shluky.

Obvykle se používá agregativní přístup, protože je intuitivnější.

Příklad v R

```
# Načteme data a odstraníme sloupec "Species"
data(iris)
iris_data <- iris[, -5]

 # Vypočteme vzdálenosti mezi objekty
distance_matrix <- dist(iris_data)
  # - dist(iris_data) spočítá vzdálenosti mezi každými dvěma objekty (na základě rozdílů v hodnotách jejich proměnných).

 # Použijeme hierarchické shlukování (agregativní metoda: "complete")
hc <- hclust(distance_matrix, method = "complete")
   #- hclust() provede samotné shlukování, v tomto případě s metodou "complete", což znamená, že dva shluky se spojí, když nejvíce vzdálené objekty mezi nimi mají co nejmenší vzdálenost.

 # Vykreslíme dendrogram
plot(hc, main = "Hierarchické shlukování (dendrogram)")
seg <- cutree(hc, k = 3)
   # rozdeli data do 3 skupin
rect.hclust(hc, k=3, border="red")
```

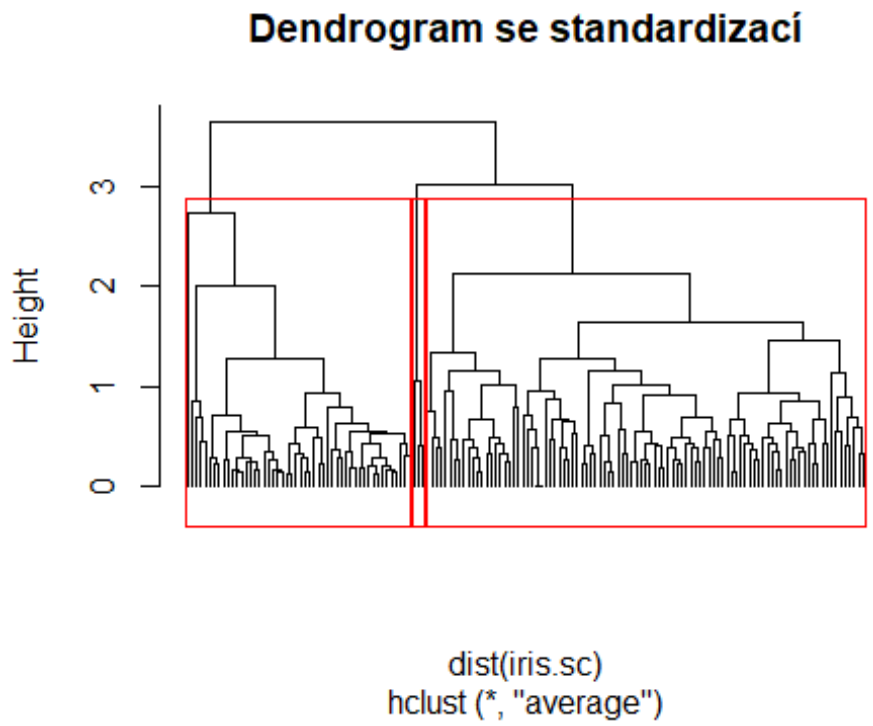


Dendrogram: - Graf, který ukazuje, jak se shluky spojovaly. Když se shluky spojují, je to zobrazeno jako spojení na určité úrovni na ose y (vzdálenost).

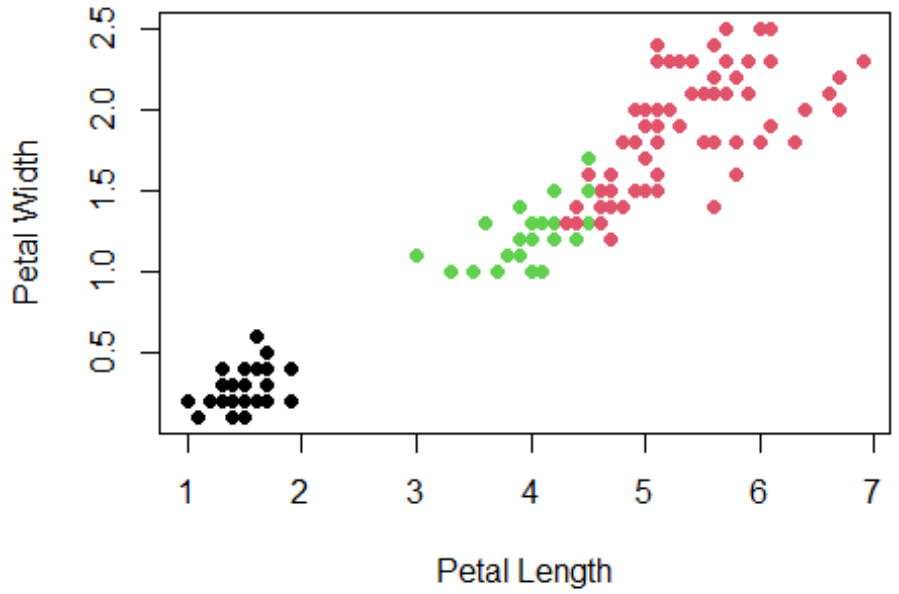
Standardizace

Standardizace zajistí, že všechny proměnné mají stejnou váhu. Výsledný dendrogram může mít odlišnou strukturu.

```
iris.sc <- scale(iris[,1:4])
 hc.iris.sc <- hclust(dist(iris.sc), method = "average")

 plot(hc.iris.sc, hang = -1, labels = FALSE, main = "Dendrogram se standardizací")
 rect.hclust(hc.iris.sc, k = 3, border = "red")
```



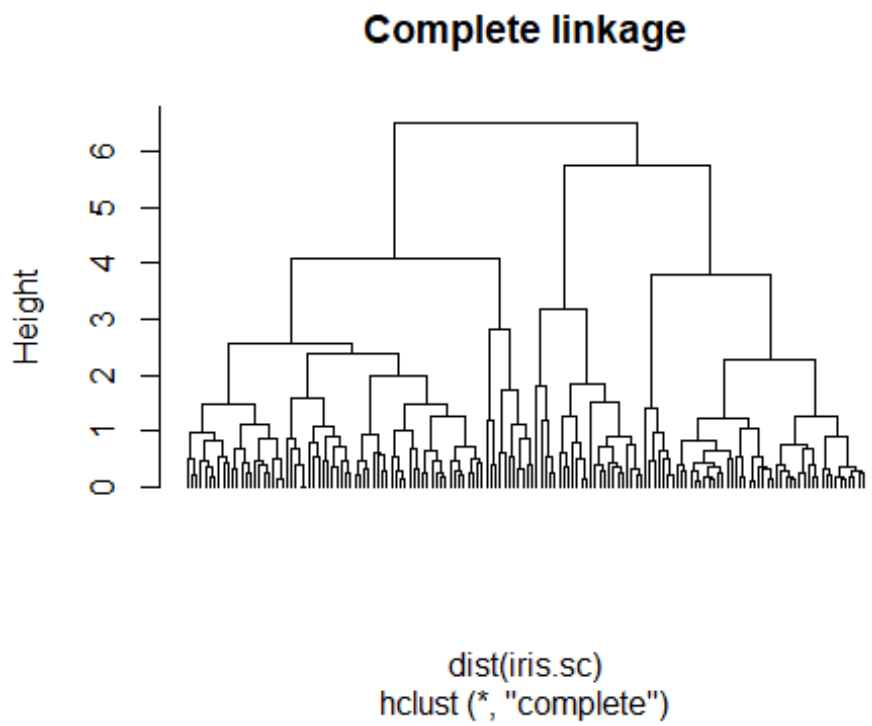

```
plot(iris$Petal.Length, iris$Petal.Width, col = seg, pch = 19,
          xlab = "Petal Length", ylab = "Petal Width")
```



Interpretace: první shluk odpovídá přesně Setosa, druhý a třetí rozlišují zbytek

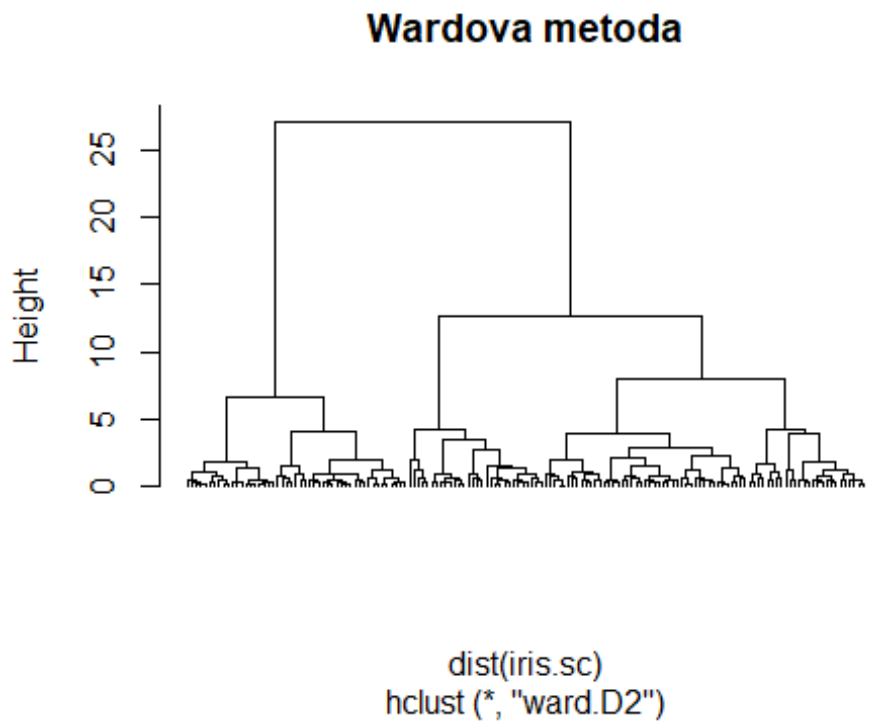
Srovnání různých metod

```
hc.complete <- hclust(dist(iris.sc), method = "complete")
 hc.ward <- hclust(dist(iris.sc), method = "ward.D2")

 plot(hc.complete, hang = -1, main = "Complete linkage", labels = FALSE)
```



#Complete = maximální vzdálenost mezi členy shluků.

```
plot(hc.ward, hang = -1, main = "Wardova metoda", labels = FALSE)
```



#Ward.D2 = minimalizace vnitroshlukové variability – často dává kompaktní, vyvážené shluky.

```
seg.comp <- cutree(hc.complete, k = 3)
 seg.ward <- cutree(hc.ward, k = 3)

 table(seg.comp, iris$Species)
```

```
##
 ## seg.comp setosa versicolor virginica
 ##        1     49         0         0
 ##        2      1        21         2
 ##        3      0        29        48
```

```
table(seg.ward, iris$Species)
```

```
##
 ## seg.ward setosa versicolor virginica
 ##        1     49         0         0
 ##        2      1        27         2
 ##        3      0        23        48
```

K-means shluková analýza

Nejčastější metoda je K-means clustering – zadáš počet skupin (např. 3) a algoritmus přiřadí každý řádek do nějakého shluku (1, 2 nebo 3):

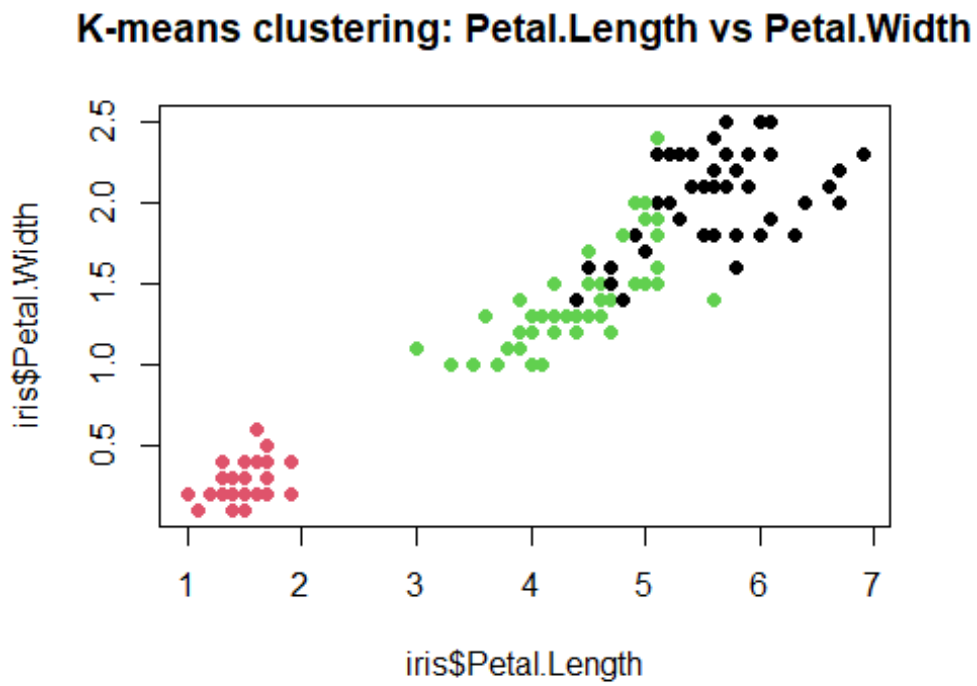
```
iris.sc <- scale(iris[, 1:4])  # Standardizace čtyř číselných proměnných

 # K-means shlukování na základě předchozí hierarchie (víme, že 3 skupiny dávají smysl)
 seg.km <- kmeans(iris.sc, 3)  # rozdělení do 3 skupin
 #Náhodně vybere 3 počáteční středy (centroidy).
 #Každý bod přiřadí do nejbližšího středu (v eukleidovském smyslu).
 #Přepočítá nové středy = průměry bodů ve shluku.
 #Opakuje kroky 2–3, dokud se přiřazení nezmění (nebo po max. počtu iterací).
 #(K-nejbližších sousedů - USU)

 # Porovnání s původními druhy květin
 table(iris$Species, seg.km$cluster)

##
 ##              1  2  3
 ##   setosa     0 50  0
 ##   versicolor 11  0 39
 ##   virginica  36  0 14
```

Všechny setosa byly zařazeny do shluku 2 → perfektní rozpoznání.

Versicolor: Většina versicolor (39) byla chybně zařazena do shluku 1 (stejně jako špatné virginica), jen 11 se dostalo do shluku 3 → K-means si spletl versicolor a virginica.

virginica: 36 z nich šlo do shluku 3 (správně), ale 14 skončilo ve shluku 1 → některé špatně zařazené.

```
# Vizualizace výsledků shlukování ve 2 proměnných
 plot(iris$Petal.Length, iris$Petal.Width, col = seg.km$cluster, pch = 19,
           main = "K-means clustering: Petal.Length vs Petal.Width")
```


K-means clustering: Petal.Length vs Petal.Width
iris$Petal.Width
0.5
1.0
1.5
2.0
2.5
1
2
3
4
5
6
7
iris$Petal.Length

Kanonické korelace (CCA)

- **Kanonická korelace (Canonical Correlation Analysis, CCA)** je metoda, která zkoumá **vztah mezi dvěma soubory proměnných**.
- Například:
    - Soubor 1: měření IQ, paměť, pozornost
    - Soubor 2: známky z matematiky, češtiny a angličtiny

CCA hledá **lineární kombinace proměnných v každém souboru**, které spolu mají **# Načti data**

Příklad v R:

```
data("USArrests")

# Vyber dva soubory proměnných:

# X: První dvě proměnné (Murder, Assault)

X <- USArrests[, c("Murder", "Assault")]


# Y: Další dvě proměnné (UrbanPop, Rape)

Y <- USArrests[, c("UrbanPop", "Rape")]

# Proveď kanonickou korelaci

cca <- cancor(X, Y)

# Výsledky

print("Kanonické korelace:")

print(cca$cor)

print("Kanonické váhy pro X:")

print(cca$xcoef)

print("Kanonické váhy pro Y:")

print(cca$ycoef)
```

Interpretace výsledků

1. Kanonické korelace (`cca$cor`)

- Vrátí vektor korelací mezi kanonickými vektory (lineárními kombinacemi z X a Y).
- Například: `[1] 0.89 0.58` znamená, že první kanonická korelace je 0.89 (silná), druhá je 0.58 (střední).
- Vyšší korelace ukazuje na silnější vztah mezi kombinacemi proměnných.

2. Kanonické váhy pro X (`cca$xcoef`)

- Udávají, jak silně se každá původní proměnná z X podílí na tvorbě kanonického vektoru.
- Například:
  - Murder: 0.8
  - Assault: 0.6
- Znamená, že Murder je v první kanonické kombinaci důležitější.

3. Kanonické váhy pro Y (`cca$ycoef`)

- Totéž platí pro druhý soubor Y.
- Váhy říkají, jak přispívají UrbanPop a Rape k odpovídajícím kanonickým vektorům.

Co z toho plyne?

- První kanonická korelace blízká 1 znamená, že existuje silný vztah mezi určitými kombinacemi proměnných z X a Y.
- Podíváme-li se na váhy, zjistíme, které proměnné nejvíce přispívají k tomuto vztahu.
- Například pokud Murder a Assault mají vysoké váhy, a UrbanPop a Rape také, můžeme říct, že kombinace násilných trestných činů koreluje s kombinací urbanizace a znásilnění.

Asymptotické p-hodnoty

Každá hodnota je p-hodnota testu, že příslušná kanonická korelace je nulová.

Příklad interpretace:

- Pro první korelaci 0.0001 (velmi malá) → statisticky velmi významná.
- Pro druhou korelaci 0.023 → stále významná, ale méně.

```
library(CCP)

p_asym <- p.asym(cca$cor, nrow(X), ncol(X), ncol(Y), tstat = "Wilks")

print("Asymptotické p-hodnoty (Wilks lambda):")
```

```
print(p_asym)
```

Permutační p-hodnoty

- Tyto hodnoty ukazují pravděpodobnost, že podobnou korelaci bys získal náhodou, pokud by mezi proměnnými žádný vztah nebyl.
- Permutační test je méně založený na předpokladech (např. normalita).
- Pokud jsou hodnoty < 0.05, korelace považujeme za statisticky významné.

```
# Permutační test významnosti (výpočetně náročnější)

# Doporučuji spustit jen na menším vzorku pro rychlost

set.seed(123)

p_perm <- p.perm(X, Y, nperms = 1000)  # 1000 permutací

print("Permutační p-hodnoty:")

print(p_perm)
```

Regresní modely

Kódy: Babichev kódy

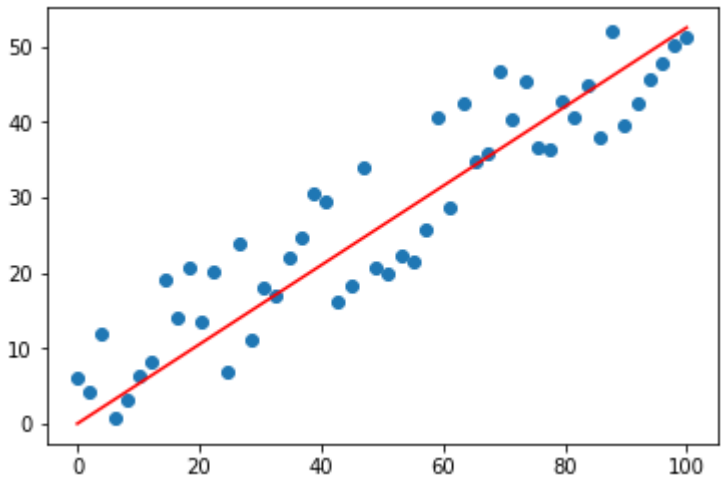
Regresní analýza

- Statistická metoda, která zkoumá vztah mezi závislou proměnnou a jednou nebo více nezávislými proměnnými.
- Závislá proměnná = její hodnota ZÁVISÍ na hodnotě nezávislé proměnné
- **Cíle:**
  - Zjištění vztahů mezi jednotlivými proměnnými
  - Předpovídání hodnot závislé proměnné
  - Odhalení klíčových faktorů, které ovlivňují výsledky
  - Zlepšení rozhodovacích procesů
- Aplikace:
  - Ekonomie: předpověď cen, analýza trhu
  - Biostatistika: modelování vztahů mezi zdravotními faktory
  - Strojové učení: regresní modely pro predikci
  - Fyzika a chemie: analýza dat z experimentů
- **Funkční závislost**
  - Hodnoty nezávislé proměnné určují přímo závislou
  - Jako matematická funkce.
- **Stochastická (volná) závislost**
  - Systematický pohyb proměnné podle pohybu druhé proměnné
  - (pohyb = klesání/růst)
- **Korelační analýza vs Regresní analýza:**
  - Korelační analýza:
    - Zjišťuje, zda mezi dvěma proměnnými existuje vztah a jak silný ten vztah je.
    - Korelace je symetrická - proměnné se navzájem popisují
    - Výstup = informace jaký je vztah (Pearsonův koeficient)
  - Regresní analýza:
    - Zjišťuje, jak konkrétně jedna nebo více proměnných ovlivňuje jinou proměnnou a vytváří matematický model (funkci) pro predikci.
    - Regrese je asymetrická - nezávislé prom. vysvětlují závislou
    - Výstup = matematický model - rovnice popisující vztah a predikující budoucí hodnoty
  - ChatGPT vysvětlení (za mě docela mňamózní):
    - Korelace je jako říct:
      - „Když lidé jedí více zmrzliny, častěji chodí k vodě.“ – vidíš vztah, ale nevíš, co co ovlivňuje.
    - Regrese je jako říct:
      - „Každé zvýšení teploty o 1 °C zvyšuje spotřebu zmrzliny o 3%.“ – víš, co ovlivňuje co a o kolik.

- **Hodnocení modelů:**
  - **R^2**
    - Udává, jak velkou část rozptylu v datech model vysvětluje v procentech
    - Hodnoty:
      - $R^2 = 1 \rightarrow$ model dokonale vysvětluje data
      - $R^2 = 0 \rightarrow$ model nic nevysvětluje
  - **MAE**
    - Mean Absolute Error
    - Průměrná velikost chyby v jednotkách výstupu
    - vyčíslení rozdílu mezi skutečnými hodnotami a predikcemi modelem
    - $$\mathrm{MAE} = \frac{1}{n}\sum |y_i - \hat{y}_i|$$
    - Vezme se absolutní hodnota rozdílu mezi skutečnou a predikovanou hodnotou (ignoruje znaménko).
    - Sečtou se tyto rozdíly a zprůměrují.
    - Míň citlivý na velký chyby
  - **MSE**
    - Mean Squared Error
    - vyčíslení rozdílu mezi skutečnými hodnotami a predikcemi modelem
    - $$\mathrm{MSE} = \frac{1}{n}\sum (y_i - \hat{y}_i)^2$$
    - Vyzdvihuje **větší chyby** – čím větší chyba, tím **víc se penalizuje**.
  - **RMSE**
    - Root Mean Squared Error
    - vyčíslení rozdílu mezi skutečnými hodnotami a predikcemi modelem
    - $$\mathrm{RMSE} = \sqrt{\mathrm{MSE}}$$
    - MSE v původních jednotkách
  - **Akaikeho informační kritérium (AIC)**
    - Penalizuje složité modely (míň než BIC ale)
    - Nižší hodnota = lepší model
    - GPT vysvětlení (dávám to sem protože z tohohle jsem to náhle pochopil i já xd)
      - Máš dvě pizzy – jedna chutná skvěle, ale má 12 ingrediencí (zbytečně složitá), druhá skoro stejně chutná a má jen 5 surovin. AIC vybere tu druhou.
  - **Bayesovské informační kritérium (BIC)**
    - Jako AIC
    - Trestá složitost modelu víc než AIC
    - Nižší hodnota = lepší model

Lineární regrese

- Modeluje lineární vztah mezi závislou proměnnou a jednou nebo více nezávislými proměnnými.
- Výstup je přímka (od slova lineární…)
- Obvykle pomocí metody nejmenších čtverců
- Předpoklady:
    - Náhodné chyby **ei**
        - jsou nezávislé
        - stejně rozdělené
        - náhodné
        - normální rozdělení
        - nulová střední hodnota
        - konstantní rozptyl
        - Pokud splňují tyto předpoklady značí se takto: ei ~ iid N(0, σ2)
    - Hodnoty nezávislé proměnné **Xk**
        - jsou vzájemně nezávislé
        - mají se závislou proměnnou Y lineární vztah
    - v datech nejsou vlivná pozorování
        - **vlivné pozorování** = outlier který ovlivní zbytek pozorování (může na obou osách)
            - NEŘÍKAT ŽE JE TO OUTLIER (nazval jsem to outlier pro lehčí pochopení)
        - pozná se cookovou vzáleností nebo vlivem (leverage)
- Omezení:
    - Výsledný model má tvar lineární funkce, i když může mít víc proměnných
    - Nezachytí nelineární vztahy (např. křivky, exponenciální průběh)
    - Citlivá na odlehlé hodnoty (outliery)



Jednoduchá lineární regrese

- jedna závislá a jedna nezávislá proměnná
- Model má tvar:
    - **$y = \beta 0 + \beta 1 Xi + ei$**

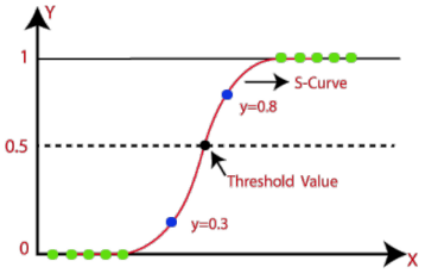
Mnohonásobná lineární regrese

- jedna závislá a více nezávislých proměnných
- Model má tvar:
    - **$y = \beta 0 + \beta 1 X1i + \beta 2 X2i + \cdots + \beta k Xki + ei$**
- Předpoklady:
    - Lineární vztah mezi závislými a nezávislými prom.
    - Nezávislost, homoskedasticita a normální rozdělení reziduí
    - Žádná nebo minimální multikolinearita mezi nezávislými prom.

Polynomiální regrese

- Rozšíření lineární regrese
- **Modeluje křivky**
- Umožňuje **modelovat nelineární vztahy** mezi závislou a nezávislou proměnnou pomocí **vyšších mocnin nezávislé proměnné.**
- Přestože výsledný tvar rovnice je zakřivený, model je stále **lineární vůči parametrům**.
- Funguje pro jednoduchou i mnohonásobnou regresi
- **Jak to funguje:**
  - Původní lineární rovnice:
    $$y = a + bx$$
    Polynomiální 2. stupně:
    $$y = a + b_1x + b_2x^2$$
    Polynomiální 3. stupně:
    $$y = a + b_1x + b_2x^2 + b_3x^3$$
  - Přidáním vyšších mocnin **nevzniká nelineární regrese**, protože koeficienty b1, b2, b3 se stále počítají lineárními metodami (např. maticově nebo metodou nejmenších čtverců).
  - Každá mocnina reprezentuje další „ohnutí“ **= stupeň volnosti** modelu.
- **Kdy to použít:**
  - Model není přímka (lineární

Logistická regrese

- Odhaduje dvě hodnoty (ano/ne, muž/žena, …) => BINÁRNÍ VÝSTUP
- Model vrací pravděpodobnost (0 - 1), ta se pak převede na kategorii (0 - 0.5 = NE, 0.51 - 1 = ANO)
- 

- **Předpoklady:**
  - Lineární vztah mezi nezávislými proměnými a log-odds (logaritmy poměru pravděpodobnosti, že nastane událost vůči tomu, že nenastane)
  - Lineární vztah mezi nezávislými a závislou proměnnou
- $$P(Y=1) = \frac{1}{1+e^{-(b_0+b_1x_1+b_2x_2+\cdots+b_nx_n)}}$$
  - $P(Y=1)$ = pravděpodobnost výsledku 1
  - e = Eulerovo číslo (~2.718)
  - $b_0, b_1, \ldots$ = koeficienty (model je naučí)
  - $x_1, x_2, \ldots$ = vstupy

  LIN. REG.
- b0,b1, …
  - „Udávají, jak moc každý vstup (nezávislá proměnná) zvyšuje nebo snižuje výslednou pravděpodobnost.“ (DUH když to ty vstupy násobí :D)
  - Pozitivní - šance roste
  - Negativní - šance klesá
- Odhad koeficientů (b0,b1, ...)
  - Získávají se pomocí MLE (Maximální věrohodnostní odhad)
  - MLE hledá takové hodnoty koeficientů, které dělají predikce modelu co nejblíže skutečným výsledkům.
  - MLE iterativně upravuje koeficienty
  - **Jak funguje MLE:**
    - $$\log L(b) = \sum_{i=1}^{n} [y_i \log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i)]$$
    - Spočítáme pravděpodobnost, že by celý dataset dopadl přesně tak, jak dopadl, pro dané koeficienty.

        - Cílem je najít ty koeficienty, pro které je ta pravděpodobnost největší = maximum likelihood.
        - Vypočítá se pravděpodobnost pro každý pozorování (řádek) a vynásobí se mezi sebou
        - Funkce je sigmoida (fun fact)
- **Hodnocení modelu:**
    - **Akaikeho informační kritérium (AIC)**
        - Penalizuje složité modely
        - Nižší hodnota = lepší model
        - GPT vysvětlení (dávám to sem protože z tohohle jsem to náhle pochopil i já xd)
            - Máš dvě pizzy – jedna chutná skvěle, ale má 12 ingrediencí (zbytečně složitá), druhá skoro stejně chutná a má jen 5 surovin. AIC vybere tu druhou.
    - **R^2**
        - Jak moc model vysvětluje variabilitu výstupu
    - **Matice záměny = CONFUSION MATRIX**
        - Tabulka, která ukazuje, kolikrát model:
            - uhodl správně ANO (True Positive = TP)
            - uhodl správně NE (True Negative = TN)
            - řekl ANO, ale bylo to NE (False Positive = FP)
            - řekl NE, ale bylo to ANO (False Negative = FN)
        - Na základě tabulky se počítá:

          Přesnost - podíl správně klasifikovaných případů.

          $$ACC = \frac{TP + TN}{TP + TN + FP + FN}$$

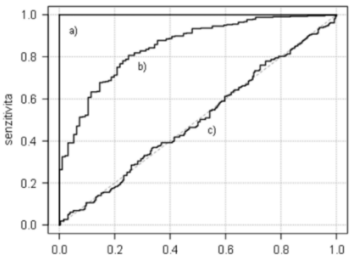
          Sensitivita (Recall) - schopnost modelu správně identifikovat pozitivní případy.

          $$RC = \frac{TP}{TP + FN}$$

          Specificita - schopnost modelu správně identifikovat negativní případy.

          $$SP = \frac{TN}{TN + FP}$$

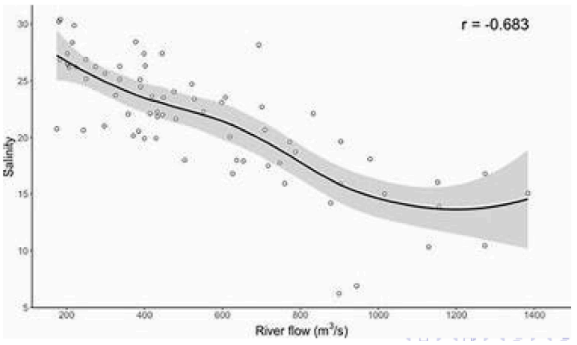
        -

- **ROC křivka**
  - Receiver Operating Characteristic
  - Graf, který ukazuje, jak dobře model odděluje 1 a 0 při různých prahových hodnotách
  - 

  - Osa Y: True Positive Rate (citlivost)
  - Osa X: False Positive Rate
  - Ideální model: křivka stoupá rychle k levému hornímu rohu.
- **AUC hodnota**
  - Area Under Curve
  - Schopnost modelu správně klasifikovat případy nezávisle na prahové hodnotě
  - Hodnoty:
    - AUC = 1: perfektní model
    - AUC = 0.5: hádání náhodně
    - AUC > 0.7: už celkem dobrý
    - AUC > 0.9: výborný model
    - AUC < 0.5: horší jak náhoda

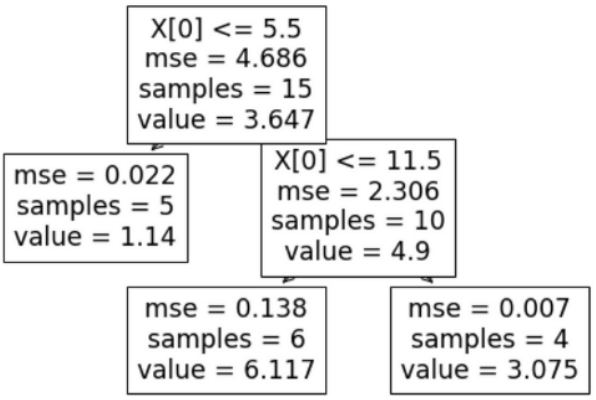
Loess regrese

- Jinak nazývaný LOWESS (”Locally Weighted Scatterplot Smoothing”)
- Neparametrická nelineární technika
- Kombinuje několik regresních modelů do meta-modelu (založený na metodě k-means)
- Aplikuje několik regresí v lokálních podmnožinách
- Může modelovat složitější nelineární vztahy
- Výstup není jedna rovnice pro všechna data
- Výstup je hladká křivka
- LOESS se používá hlavně pro zobrazení trendu v grafech a vizualizacích.

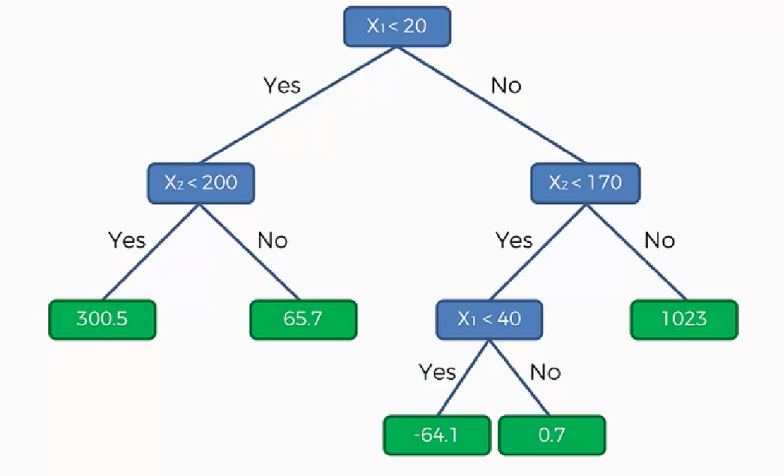


- **Charakteristiky:**
  - LOESS nevytváří jeden globální model, ale staví malé jednoduché modely v okolí každého bodu v datech.
  - Každý malý model je vytvořen pomocí **vážené regresní analýzy**, kde mají nejbližší body **větší vliv** než ty vzdálenější. (nejmenší čtverce)
  - Vyhlazovací parametr (šířka pásma)
    - Říká, kolik sousedních bodů se použije při tvorbě lokálního modelu. Větší šířka = víc bodů = hladší křivka.
  - Dobře hodí tam, kde nemáš tušení, jak data vypadají. (nepředpokládá nějaký tvar jako parabolu)
  - Náročný na výpočet = POMALÝ
  - Není dobrý pro předpovídání hodnot mimo křivku
  - S každou nezávislou proměnnou roste náročnost výpočtu
- **Jak funguje:**
  - Zaměříš se na jeden konkrétní bod (např. X = 5).
  - Vybereš jeho okolí (např. 30 % nejbližších bodů podle X).
  - Těmto bodům přiřadíš váhy – nejbližší bod dostane nejvyšší váhu, vzdálenější menší.
  - Fitneš jednoduchou regresi (např. přímku) jen na tyto body a váhy.
  - Výsledek (predikce) pro tento bod je hodnota na této regresní křivce.
  - To celé zopakuješ pro každý bod – tím vznikne hladká zakřivená křivka přes celá data.

Decision tree regrese



- [Kvalitní vysvětlení part 1]
- [Kvalitní vysvětlení part 2]
- (^^^ Doporučuji pustit si ty dvě videa)
- Model, který rozděluje vstupní prostor do oblastí podle hodnot vstupních proměnných
- Každé rozdělení (uzel) klade otázku typu „je hodnota menší než...?“ a na základě odpovědi tě pošle dolů buď doleva, nebo doprava.
- Cílem je předpovědět číselnou hodnotu, ne kategorii.
-
- Jak funguje decision tree:
    - Strom dělí data (hodnoty závislé proměnné) podle vstupních (nezávislých) proměnných, jako jsou věk, plocha, cena atd.
    - Decision node => x <= 5
    - Leaf node => hodnota 5.2654
    - Každá node se vybere tak, aby minimalizovala chybu ve 2 vzniklých nodes
        - Chyba se může reprezentovat různě, v případě regrese například pomocí hodnoty MSE



    - Výsledné nodes obsahují průměrnou hodnotu závislé proměnné v té oblasti
    - Strom se rozšiřuje dokud:
        - Nedosáhne limitu hloubky
        - V listu není už málo dat (když mi zůstane už jen jedna hodnota tak to nebudu dělit)
        - Dělení už nikam nevede - všechny možnosti mi vrátí stejnou hodnotu entropy
- Jak využít decision tree:
    - Strom je prakticky série IF ELSE
    - Do stromu vložím moji vstupní (nezávislou) hodnotu
    - Strom mi vrátí hodnotu na základě podmínek (ve výsledné větvi je průměrná hodnota v dané oblasti)
- Sám o sobě není úplně dobrý, je lepší používat random forest
    - Je vhodný pro složité a nelineární vztahy, ale může se snadno přetrénovat.
    - Hodí se jako základ pro silnější modely (RANDOM FOREST)
    - Náchylný k přetrénování (overfittingu) – strom může být příliš složitý
    - **Výsledky jsou často nestabilní – malá změna dat → jiný strom**
    - Horší predikční výkon než složitější modely (jako RANDOM FOREST)

Random forest regrese

- Opět kvalitní vysvětlení
- Kolekce několika decision trees
- Řeší problém Decision tree - náchylnost na malou změnu dat
- Jak to funguje:
    - Udělám bootstrap z původních dat
    - Z každého bootstrap vzorku udělám decision tree
    - Dám můj vstup do všech vzniklých stromů
    - Pro klasifikaci platí nejčastěji se vyskytující hodnota mezi stromy (Agregace)
    - **Pro regresi platí průměr hodnot všech stromů** (taky agregace)
    - Fun fact - Agregace z bootstrapu se nazývá **bagging**
- Proč to funguje:
    - Snižuje se rozptyl modelu, aniž by se výrazně zvýšila bias.
    - Každý strom je "hloupý" trochu jinak, ale když je jich hodně, dohromady dají rozumný výsledek.
    - Agregace (průměrování) snižuje šum a chyby jednotlivých stromů.

Support vector regrese

- Babichev neprobíral - tady je pouze obecně ať máme přehled co to je
- Regresní verze klasifikátoru SVM
- Místo hledání hranice mezi třídami hledá funkci, která se "trefí do trubice" o šířce ε kolem reálných hodnot.
- SVR najde "tunel", kterým chce projet co nejvíc bodů bez trestu, a jen ty venku model penalizuje.
- Jak funguje:
    - Model netrestá chyby, které spadnou do ε trubice (epsilon-insensitive zone) – malá odchylka nevadí.
    - Penalizuje se jen chyba mimo ε.
    - Minimalizuje se složitost modelu + penalizuje body mimo ε.
    - Může být nelineární díky kernelům (např. RBF, polynomiální).
- Vzorec:
    - $$\min\left(\frac{1}{2}\|w\|^2 + C\sum \text{chyby mimo } \varepsilon\right)$$
        - `C` = důraz na chybu mimo ε
        - `w` = vektor koeficientů modelu
- Předpoklady:
    - Nevyžaduje normalitu, lineární vztah nebo homoskedasticitu.
    - Je citlivý na výběr kernelu, ε a C (parametr ladění).
- Kdy použít:
    - Když chceš odolnost proti outlierům.
    - Když chceš nelineární vztah, ale nevíš přesný tvar.
    - Když potřebuješ model s pevně danou tolerancí chyby.

Ridge regrese

- Babichev neprobíral - tady je pouze obecně ať máme přehled co to je
- Klasická lineární regrese + penalizace velkých koeficientů
- Zabraňuje přeučení modelu (overfittingu)
- Minimalizuje MSE + penalizuje velikost koeficientů
- Vzorec:
  - $$\min\left(\sum(y_i - \hat{y}_i)^2 + \lambda \sum w_j^2\right)$$

    `λ` říká, **jak moc penalizujeme složitost**

    Všechna `w` se **"stahují ke 0"**, ale **žádné nezmizí úplně**
- Předpoklady:
  - Stejné jako klasická lineární regrese
    - Lineární vztah, nezávislost, homoskedasticita, normalita reziduí
  - Plus:
    - Nezávislé proměnné nemusí být zcela nekorelované → Ridge pomáhá i při multikolinearitě
- Kdy použít:
  - Když máš hodně prediktorů
  - Když máš silně korelované vstupy
  - Když chceš zabránit přetrénování

Náhradní otázky

Pro tyto otázky platí, že **nemusíme jít moc do hloubky.** Stačí říct o co jde, k čemu to je a jak to funguje. Ve stručnosti okolo jedné minutky ukázat, že tomu rozumíme

Věcná významnost

- **Statistická významnost**
  - Je pravděpodobný, že nalezený efekt není způsobený náhodou
  - Hodnotí se pomocí p-hodnoty
  - Statisticky významný výsledek = co jsem zjistil není náhoda
  - Závisí na počtu pozorování (protože p-hodnota závisí na počtu pozorování)
    - málo pozorování dává "velkou" p-hodnotu
    - hodne pozorování dává "malou" p-hodnotu ˇ
    - Statistické testy dobře fungují pro počet pozorování kolem 100 hodnot
- **Věcná významnost**
  - Je nalezený efekt dost velký, aby měl smysl v reálném životě
  - Hodnotí se pomocí velikosti efektu (effect size)
  - Nezávisí na počtu pozorování
  - Věcně významný výsledek = co jsem zjistil ovlivní populaci pstruhů v ČR
    - Rozdíl mezi skupinami je 10cm - 10cm je hodně.
    - “Jde to použít, ta informace něco znamená.”
- Výsledek může být statisticky významný ale věcně nevýznamný
  - Rozdíl výšky 0.5cm mezi dvěma skupinami
- Řeší to jen nějaký odvětví, např. psychologie

Odhad počtu pozorování

- “Existuje vztah mezi počtem pozorování, hladinou významnosti a sílou testu.”
- Každý test má svůj vzorec pro výpočet ideálního počtu pozorování (sample size)
- Hlavní faktory vzorců jsou:
  - **hladina významnosti**
  - **síla testu**
  - Pro obě platí - čím větší to chci, tím víc pozorování potřebuji

Tabulka analýzy rozptylu

- Pro porovnání variability vysvětlené a nevysvětlené
- Nejčastěji v ANOVĚ - porovnání střední hodnoty v několika nezávis. výběrech
- Důležité pro věcnou významnost, protože ukáže data potřebná pro výpočet velikosti efektu - eta-squared a koeficient determinace (R^2, poměr vysvětlené variability k celkové)
- **Variabilita vysvětlená**
  - Část variability dat, kterou náš model dokáže vysvětlit
- **Variabilita nevysvětlená**
  - Část variability dat, kterou náš model nedokáže vysvětlit

- Model to chápe jako náhodný šum

$$SST = \sum_{i=1}^{k}\sum_{j=1}^{n_i}(X_{ij} - \overline{X}_{..})^2$$

$$SSA = \sum_{i=1}^{k} n_i(\overline{X}_{i.} - \overline{X}_{..})^2$$

$$SSe = \sum_{i=1}^{k}\sum_{j=1}^{n_i}(X_{ij} - \overline{X}_{i.})^2$$

	Součty čtverců	Stupně volnosti	Průměrné čtverce	Testová statistika	*p*-hodnota
Faktor *A*	SSA	$dfA = k - 1$	$MSA = \frac{SSA}{dfA}$	$F = MSA/MSe$	p
Chyba *e*	SSe	$dfe = n - k$	$MSe = \frac{SSe}{dfe}$		
Celkem	SST	$dft = n - 1$			

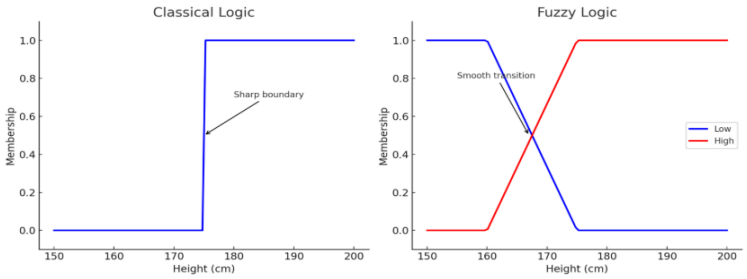
- **Ukazatele věcné významnosti**
  - Převážně pro velká data - získané metaanalýzou = kombinace několika výzkumů na stejné téma
  - Hodnoty ukazatelů:
    - do 0,2–0,3 Malý efekt (téměř nevýznamné)
    - 0,5 Střední efekt (něco to znamená)
    - 0,8 a víc Velký efekt (významný dopad)
  - Ukazatele:
    - Cohenovo D
      - Používá se k měření velikosti efektu mezi dvěma skupinami (například experimentální vs. kontrolní).
      - Počítá se jako rozdíl mezi průměry dvou skupin, vydělený společným směrodatným odchylkou
    - Hedgesovo G
      - Podobné jako Cohenovo d, ale používá úpravu pro malé vzorky => menší než 20–30 jedinců
    - Glassovo OMEGA
      - Podobné jako Cohenovo d
      - Místo společné směrodatné odchylky používá směrodatnou odchylku kontrolní skupiny.
      - Kdy použít?
        - Když jsou směrodatné odchylky mezi skupinami výrazně rozdílné.
    - Eta squared (η^2)
      - Podíl vysvětlené variability
    - Omega squared (ω^2)
      - Lepší a méně zkreslený ukazatel podobný η^2

- Existuje vztah mezi korelací(r) a regresí(R)
  - Pokud máme jednoduchou lineární regresi, tak:

    $$R^2 = (r)^2$$

    R^2 (koeficient determinace) ukazuje, **kolik procent variability** v závislé proměnné umí vysvětlit nezávislá proměnná.

Fuzzy modely



Fuzzy logika

- Klasická logika
  - Hodnoty 0 a 1
  - Ostré hranice - něco platí, nebo neplatí
  - Příklad:
    - "Člověk vyšší než 180 cm je vysoký"
    - => 179 = není vysoký, 181 = vysoký.
- Fuzzy logika
  - Hodnoty mezi 0 a 1
  - Plynulý přechod mezi 2 stavy
  - Člověk může být 70% vysoký (0.7 hodnota vysokosti)
  - Založena na fuzzy množinách
- Fuzzy množina
  - Množina do které každý prvek patří na několik procent (má stupeň příslušnosti)
  - Funkce příslušnosti
    - Určuje jak moc daný prvek do množiny patří
  - Příklad:
    - Množina vysokých lidí
      - Osoba 170 cm → příslušnost 0.2
      - Osoba 180 cm → příslušnost 0.7
      - Osoba 190 cm → příslušnost 0.95
- Fuzzy Model
  - Model využívající fuzzy logiku k rozhodování
  - Složení:
    - Fuzzyfikace vstupů
      - Převod hodnot na fuzzy hodnoty
      - 175cm -> 0.2 vysoký
    - Fuzzy inference system (FIS)
      - Pravidla jak klasifikovat
      - IF výška je vysoký AND váha je nízká THEN sportovec.
    - Defuzzyfikace výstupu
      - převede fuzzy výstup zpět na konkrétní hodnotu
      - Příklad:
        - hodnota 0.8 → ANO, s důvěrou 80 %
- K čemu se to používá:
  - Řízení systémů s neurčitostí - nedá se přesně definovat hranice
  - Výhoda:
    - Nevyžadují přesný matematický model, stačí expertní pravidla.
  - Irl příklady:
    - Regulace klimatizací
    - Ovládání praček, mikrovln, aut
    - Hodnocení rizik

- Dříve to bylo populární, nyní se od toho upustilo
    - Problém škálovat - na vše musí být přesná IF THEN pravidla
    - Neučí se automaticky - vše musím psát ručně
    - Nefunguje dobře s velkými daty
    - Nahrazeno Bayesovskými sítěmi a Neuronovými sítěmi
    - Přesto v jednoduchých systémech jsou stále využitelné (např. zabudovaná elektronika).
- Příklad fuzzy modelu v akci:
    - Automatizovaná klimatizace
        - IF teplota je vysoká (0.8) AND vlhkost je vysoká (0.6)
        - THEN aktivuj chlazení silně (výsledek = max(0.8, 0.6) → 0.8)

Bayesovské sítě

Bayesova věta v kostce - mám pravděpodobnost něčeho, s každou novou informací týkající se dané věci aktualizuji pravděpodobnost,

Bayesova věta (formule, vzorec)

- Vzorec ukázaný v příkladu
- Pravděpodobnost
- Umožňuje aktualizovat naše přesvědčení (pravděpodobnost hypotézy) na základě nových důkazů nebo informací.
- Spojuje předchozí znalosti s novými daty
    - Kombinuje to, co jsme věděli dřív (prior), s tím, co jsme nově pozorovali (evidence).
- Základ pro uvažování v nejistotě
    - Umožňuje formálně pracovat s nejistotou v rozhodování, diagnostice, predikci nebo inferenci.
    - Když nevíš s jistotou, co je pravda (např. zda pacient má nemoc), ale musíš rozhodnout, co udělat.
    - Bayes ti poskytne pravděpodobnosti každé možnosti na základě dostupných dat → rozhoduješ racionálně, ne intuitivně.
- Bayesovská statistika:
    - zachází s neznámými veličinami jako s náhodnými proměnnými
        - například parametry modelu, o kterých nic nevíme, modelujeme pomocí rozdělení pravděpodobnosti.
        - nazývá se PRIOR
    - Pracuje s Bayesovou větou jako jádrem výpočtu
    - Výsledkem není bodový odhad, ale rozdělení pravděpodobnosti.
        - nazývá se postprior
- Využití:
    - Detekce spam mailu
        - Pravděpodobnost se aktualizuje podle počtu specifických slov

Příklad

V naší skupině je 1 špion, je to muž?

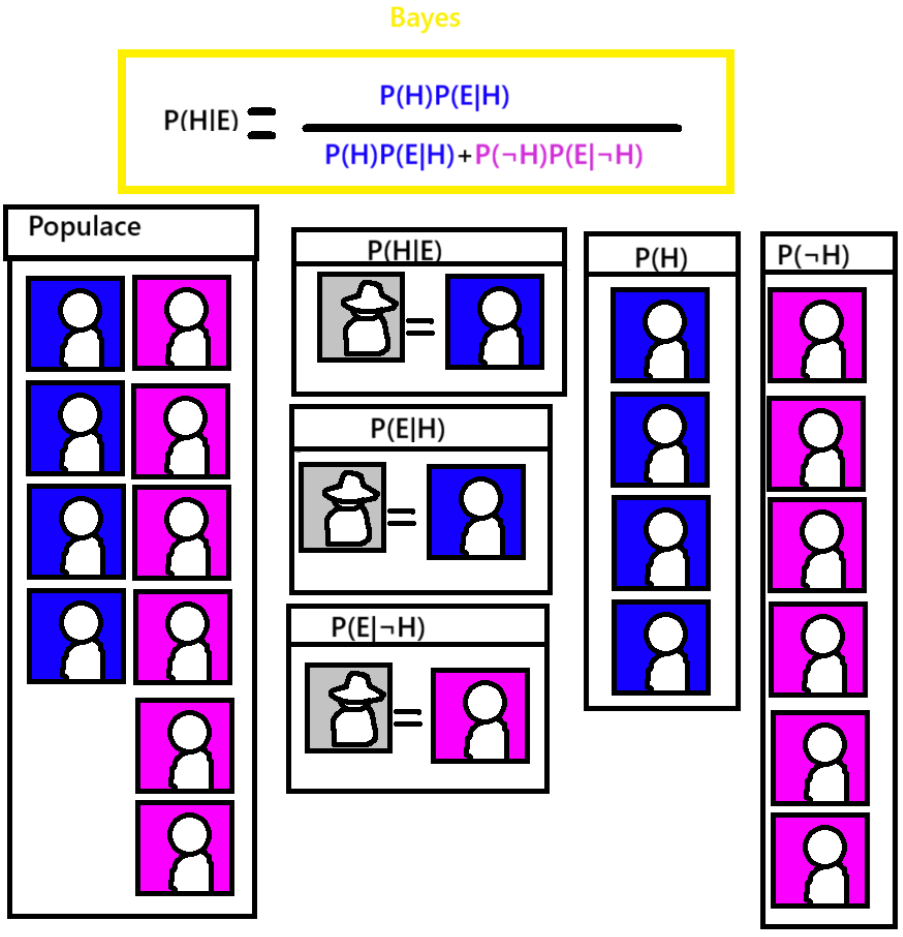
- Ve skupině je 10 lidí, 4 jsou muži.
- Vím důvěryhodně, že šance na to, že člověk je špion je 20%

Bayes vzorec

$$P(H|E) = \frac{P(H)P(E|H)}{P(H)P(E|H)+P(\neg H)P(E|\neg H)}$$

H = Hypotéza (špion je muž)
E = Evidence - důkaz (prozatím je to jen fakt, že ve skupině je špion)

P(H|E) = Pravděpodobnost, že platí hypotéza(je to muž), pokud platí důkazy(je špion)
P(E|H) = Pravděpodobnost, že platí důkazy(je špion), pokud platí hypotéza(je to muž)
P(E|¬H) = Pravděpodobnost, že platí důkazy(je špion), pokud neplatí hypotéza (není muž)
P(H) = Pravděpodobnost, že platí hypotéza (špion je muž)
P(¬H) = Pravděpodobnost, že neplatí hypotéza (špion není muž)



P(E|H) = Ve skupině mužů je stejná šance jako u žen že je někdo špion = 0.2
P(E|¬H) = Ve skupině mužů je stejná šance jako u žen že je někdo špion = 0.2
P(H) = 0.4 (4/10 lidí jsou muži)
P(¬H) = 0.6 (6/10 lidí jsou ženy)

Dosazení do Bayesova vzorce:
P(H|E) = (0,2*0,4) / (0,2*0,4 + 0,2*0,6) = 0,4 = 40%
Na tuhle pravděpodobnost zatím ani nebyl potřeba Bayes :-)

Později se dozvím, další důkazy (špion nosí černou, má brýle , má dlouhé vlasy)

2 muži a 3 ženy jsou v černé
1 muž a 2 ženy nosí brýle
1 muž a 4 ženy mají dlouhé vlasy

=> Nové důkazy (a všechny budoucí) upraví P(E|H) a P(E|¬H)
P(E|H) = šance že je špion * šance že muž nosí černou * šance že muž nosí brýle * šance že muž má dlouhé vlasy
P(E|H) = 0.2 * 0.5 * 0.25 * 0.25 = 0.0063

P(E|¬H) = šance že je špion * šance že žena nosí černou * šance že žena nosí brýle * šance že žena má dlouhé vlasy
P(E|¬H) = 0.2 * 0.5 * 0.333 * 0.667 = 0.0222

P(H|E) = (0.0063*0.4) / (0.0063*0.4 + 0.0222*0.6) = 0,16 = 16%

Bayesovské sítě

- Kombinují Bayesovskou statistiku s teorií grafů
- Rozdíl mezi Bayesovou větou a sítěmi:
    - Bayesova věta pracuje typicky jen s dvěma hypotézami a evidencí (např. je špion muž?).
    - Bayesovské sítě umožňují pracovat s desítkami až stovkami závislých proměnných.
    - Reprezentují komplexní závislostní struktury.
    - Efektivně umožňují výpočet libovolné podmíněné pravděpodobnosti.
- Nejčastěji jsem našel využití v medicíně