PSM ULTIMATE v3

Obsah

1. Mnohorozměrná statistika

1.1 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)
  - testuje rozdíl mezi dvěma skupinami ve více proměnných současně

- ***MANOVA*** (mnohoroz. ANOVA)
  - porovnává průměry více skupin v několika závislých proměnných zároveň

- ***Kanonické korelace*** (mnohoroz. korelační koeficient)
  - hledá vztahy mezi dvěma množinami proměnných

- ***Mnohorozměrná regrese***
  - modeluje vliv jedné či více nezávislých proměnných na několik závislých proměnných současně

– cíle:

- redukovat data (hlavní komponenty)
- seskupit pozorování (shluky)
- najít rozdíly mezi skupinami (diskriminace).

1.2 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)

1.2.1 MANOVA

– rozšíření (zobecnění) ANOVY na více závislých proměnných.

ANOVA

– ANalysis Of Variance
– 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

– 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: ***T = W + B*** (T = celková variabilita; W = vnitroskupinová variabilita – šum uvnitř skupin; B = meziskupinová variabilita – rozdíly mezi skupinami)

$$W = \sum_{g=1}^{G} \sum_{i \in g} (y_i - \bar{y}_g)(y_i - \bar{y}_g)^\top$$

- G → počet skupin (tříd, kategorií...)
- $i \in g$ → jednotlivé osoby v dané skupině
- y_i → vektor všech proměnných pro jednotlivce i
- $\bar{y}_g$ → průměr vektoru proměnných pro skupinu g
- $(y_i - \bar{y}_g)(y_i - \bar{y}_g)^\top$ → maticový součin rozdílu → říká, jak moc se každý člověk liší od průměru své skupiny

$$B = \sum_{g=1}^{G} n_g (\bar{y}_g - \bar{y})(\bar{y}_g - \bar{y})^\top$$

- n_g → počet lidí ve skupině g
- $\bar{y}_g$ → průměr vektoru proměnných pro skupinu g
- $\bar{y}$ → celkový průměr přes všechny skupiny
- $(\bar{y}_g - \bar{y})(\bar{y}_g - \bar{y})^\top$ → maticový součin → ukazuje, jak moc se průměr skupiny liší od celkového průměru

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

– A to pak používají jednotlivé testové statistiky, které se vybereme (Dáváme jako parametr do MANOVi). Např Wilks Lambda:

$$\Lambda = \frac{|W|}{|W + B|}$$

- $|\cdot|$ → determinant matice (číslo, které shrnuje „velikost variability")
- Poměr říká: **kolik variability je uvnitř skupin vs. celkem**
  - malé Λ → skupiny jsou od sebe hodně rozdílné
  - velké Λ → skupiny jsou podobné

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)

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

1.2.2 Hotellingův test

– T-test pro více závislých proměnných (zobecnění t-testu)
– jako MANOVA pro 2 skupiny
– předpoklady:
- Stejné jako pro MANOVU
- Vzorec:

$$T^2 = \frac{n_1 n_2}{n_1 + n_2}(\bar{X} - \bar{Y})^T \mathbf{S}^{-1}(\bar{X} - \bar{Y})$$

$$\mathbf{S} = \frac{(n_1 - 1)\mathbf{S}_1 + (n_2 - 1)\mathbf{S}_2}{n_1 + n_2 - 2}$$

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
```

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

1.3 Redukce dimenzionality

1.3.1 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:

- mějme mnohorozměrná data v prostoru
- daty proložíme vektor ve smeru s největší variabilitou
- tak získáme první hlavní komponentu
- hledáme vektor, který by byl k prvnímu kolmý a opět byl ve směru s největší variabilitou
- získáme druhou hlavní komponentu
- hledáme vektor, který by byl kolmý k prvním dvěma a byl
- ve směru s největší variabilitou
- získáme třetí hlavní komponentu
- 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.

– vzorec: $$PC_1 = a_{11} \cdot X_1 + a_{12} \cdot X_2 + \cdots + a_{1p} \cdot X_p$$

- Máš hodně proměnných (X_1, X_2, …, X_p) – třeba výška, váha, délka nohy.
- PCA si řekne: „Já je všechny smíchám do jedné nové proměnné (PC1), která zachytí co nejvíc informací najednou.“
- Ty koeficienty (a_{11}, a_{12}, …) jsou jen „míra příměsi“, kolik si PCA vezme z každé původní proměnné.
- Výsledek je nová osa, na které se data roztahují nejvíc.

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)
```

– k***orelace***

```
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
```

- 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:

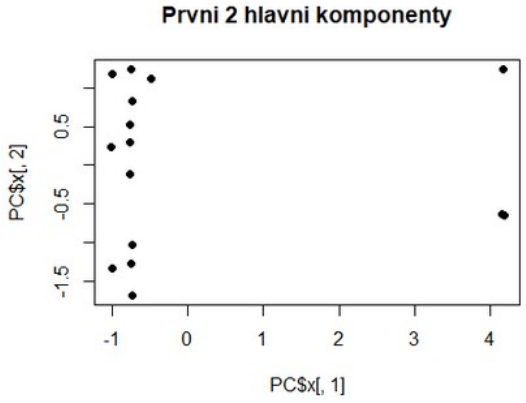
```
screeplot(princomp(m1, cor = T), type="l")
 abline(h=1, col="green")

  # dostatecne velke procento vyuzite variability (80%)

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:

```
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

1.3.2 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.

– vzorec: $$X_i = \lambda_{i1}F_1 + \lambda_{i2}F_2 + \cdots + \lambda_{im}F_m + \varepsilon_i$$

- Xi → jedna konkrétní otázka (např. „umíš násobit?“).
- F1,F2,…,Fm → skryté faktory (např. matematické schopnosti, jazykové schopnosti).
- λij → „váha“, jak moc faktor Fj ovlivňuje otázku Xi.
- εi → zbytek, co faktor nevysvětlil (náhoda, jiné schopnosti).
- Představ si, že máš test s 10 otázkami:
  - Některé otázky měří „matematické schopnosti“.
  - Jiné měří „jazykové schopnosti“.
  - Faktorová analýza ti pomůže zjistit, že za 10 čísly (odpověďmi) stojí třeba jen 2 skryté faktory (matematika a jazyky).

– předpoklady:

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

Využití v R

factanal(m1, factors = 3, rotation = "varimax")

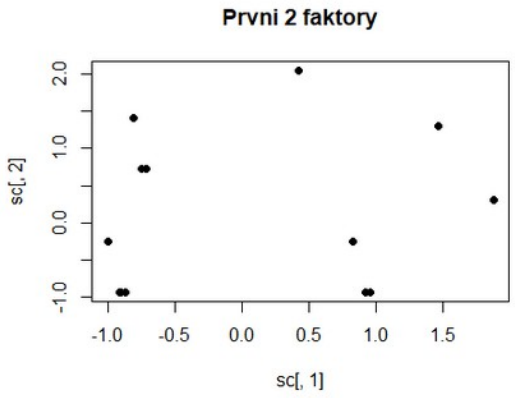
- Počet faktorů mi znázorňuje screeplot z eigen v PCA.. kolik jich je větších než 1
- rotation: “vyčistí” nám proměnné, aby se lépe interpretovali (velké hodnoty zvýší, malé sníží)

```
## Call:
 ## factanal(x = m1, factors = 3)
 ##
 ## Uniquenesses:
 ##    v1   v2    v3    v4    v5    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
```

- Uniquenesses: jaké proměnné jsou zajímavé, obsaženy v analýze.. nízká čísla good (0.1). Pokud velká, může se vynechat (Nekoreluje s ostatními)
- Factor: Jako u PCA.. ale tady už ty factory jdou snáze pojmenovat dle proměných (Factor1: kdyby V1 byl alkohol a v2 byla kyselost.. tak ho pojmenuji alkohol a kyselost)

```
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")
```



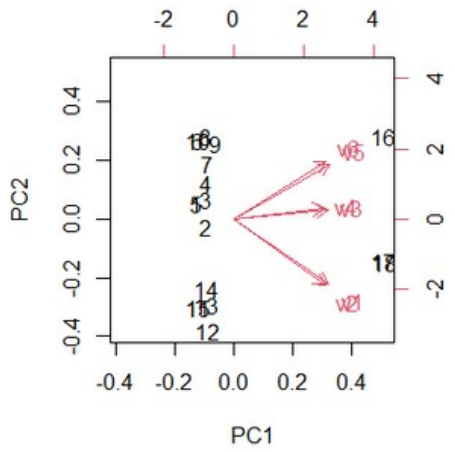
- Graf zobrazuje rozložení pozorování podle prvních dvou faktorů.
- Blízká pozorování = pozorování, která jsou si podobná ve faktorech. Pokud jsou body u sebe, znamená to, že mají podobné hodnoty faktorových skóre → tedy jsou si podobná ve významných rysech, které faktory vystihují.
- Vzdálená pozorování = 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:
  - Standard deviation: Odchylka každé hlavní komponenty (PC1 až PCn) – čím větší, tím víc informací komponenta obsahuje.
  - Proportion of Variance Podíl vysvětlené variability – kolik % informací o původních datech tato komponenta zachycuje.
  - Cumulative Proportion Kumulovaný podíl – součet všech předchozích Proportion of Variance → ukazuje, kolik % informace získáme, když si vezmeme např. první 2 komponenty.

```
biplot(PC)
```



- Čísla v grafu označují pozorování
- 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.

1.4 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

– vzorec: $$D = w_1X_1 + w_2X_2 + \cdots + w_pX_p$$

- X1,X2,…,Xp → pozorované proměnné (např. krevní tlak, hladina cukru).
- w1,w2,....,wp → váhy, které určí, jak moc daná proměnná přispívá k rozdělení skupin.
- D → diskriminační skóre (jedno číslo, které říká, „kam“ daný člověk spadá).

– předpoklady:
- ***Multivariátní normalita*** – prediktory v každé skupině (třídě) mají vícerozměrné normální rozdělení.
- ***Shodné kovarianční matice napříč skupinami*** – předpokládá se, že všechny třídy mají stejnou kovarianční strukturu (homogenita rozptylů a kovariancí).
- ***Nezávislá pozorování*** – jednotlivé řádky dat nejsou vzájemně závislé.
- ***Absence silné multikolinearity*** – prediktory by neměly být příliš silně korelované, protože to ovlivňuje odhad diskriminačních funkcí.
- ***Žádné výrazné odlehlé hodnoty*** – outliery mohou silně zkreslit kovarianční matice i diskriminační funkce.

– 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.

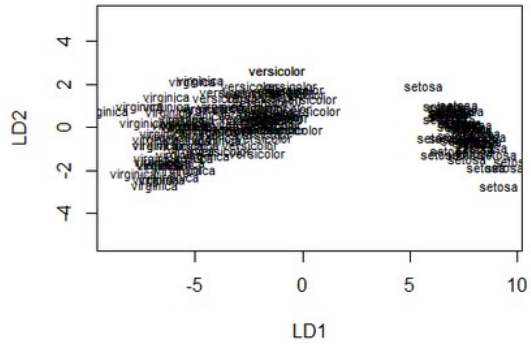
– model

```
## 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
```

- Prior (apriorní) probabilities: máme rovnoměrné zastoupení druhů -> ovlivňují, jaký druh bude preferován. Tady LDA spočítal, že v datech jsou stejně zastoupeny, takže žádný druh nebude vybírán přednostně
- Group means: průměrné hodnoty jednotlivých měření pro každý druh. Např. setosa má nejmenší okvětní lístek (Petal.Length).

- LD1, LD2 Diskriminační osy (směry) pro oddělení skupin
- Coefficients of linear discriminants: Jak silně daná proměnná přispívá k oddělování skupin podél daného směru.
- 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.

– předpověď:

```
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 %

1.5 Shluková analýza (Clustering)

– shluková analýza je metoda, která automaticky hledá skupiny (shluky) podobných objektů
– na rozdíl od diskriminační analýzy, tady nevíme, kolik skupin existuje
– obecný princip:

- Vezmeš data (např. lidi popsané výškou, váhou, věkem).
- Změříš vzdálenosti mezi nimi (typicky euklidovská).
- Algoritmus je postupně seskupuje podle podobnost

– vzorec $$d(i,j) = \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + \cdots + (x_{ip} - x_{jp})^2}$$

- Klasická Euklidovská (Pythagoras).
- x = proměnná, u které se měří vzdálenost (výška)
- i,j, = Hodnota dané proměnné pro skupinu/objekt (naměřená výška prvního jedince)

1.5.1 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***

- 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
```

1.5.2 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 <- iris[, 1:4] (bez labelu Species)
```

– 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)

```
 # Porovnání s původními druhy květin
 table(iris$Species, seg.km$cluster)
##
 ##              1  2  3
 ##   setosa     0 50  0
 ##   versicolor 2 0 48
 ##   virginica  36  0 14
```

- Všechny setosa byly zařazeny do shluku 2 → perfektní rozpoznání.
- Versicolor: Většina versicolor (48) byla zařazena do shluku 3, jen 2se dostalo do shluku 1 → K-means si spletl versicolor a virginica (ale zde jen u 2 případů).
- virginica: 36 z nich šlo do shluku 1 (většina), ale 14 skončilo ve shluku 3 → 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")
```

- 2 skupiny (zelená a černá) se míchají.
- U K-means můžeme ještě standartizovat data (scale(idis[, 1,4]))... ale zde je to zbytečné, páč proměnné jsou ve stejných jednotkách.

- Jak snadno zjistit, jestli ***standratizovat***?
  - Rozsahy
    `apply(iris[,1:4], 2, range)`
  - Směrodatné odchylky
    `apply(iris[,1:4], 2, sd)`

- Výsledek ti ukáže, jestli jsou proměnné ve stejném měřítku:
- Pokud se směrodatné odchylky hodně liší (např. jedna je 0.5 a jiná 200), pak je standardizace nutná.
- Pokud jsou hodnoty v podobném rozsahu (jako u iris: kolem 0.4–1.7), pak standardizace spíš není potřeba.

1.6 Kanonické korelace (CCA)

– Canonical Correlation Analysis (CCA) metoda, 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 korelaci

– vzorec:

$$U = a_1X_1 + a_2X_2 + \cdots + a_pX_p$$
$$V = b_1Y_1 + b_2Y_2 + \cdots + b_qY_q$$

- X1,…,Xp → první skupina proměnných
- Y1,…,Yq → druhá skupina proměnných
- ai, bj → váhy, které hledáme
- U,,V → tzv. kanonické proměnné (lineární kombinace obou skupin)
- Pak spočítáme korelaci mezi Ua V – To je kanonický korelační koeficient.
- V reálu máme sadu kanonických párů (U1,V1) (U2,V2)... takže se řeší přes matice.. ale v podstatě se řeší toto.

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] 0.66 0.27 znamená, že první kanonická korelace je 0.66 (poměrně silná) -> existuje poměrně silná lineární závislost mezi určitou lineární kombinací proměnných v X a určitou lin. kombinací proměnných v Y, druhá je 0.27 (slabá) -> slabá závislost mezi X a Y
  - Vyšší korelace ukazuje na silnější vztah mezi lineární kombinacemi proměnných.
  - (a1X1 + a2X2 + a3X3….) a (a1Y1 + a2Y2…) -> jak silně jsou propojeny tyto nejlepší lineární kombinace

- 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.
    - Jak velkou váhu má každá proměnná pro vztah s Y

- 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.

1.6.1 Asymptotické p-hodnoty

– každá hodnota je p-hodnota testu, že příslušná kanonická korelace je nulová.
– fungují dobře pro velká n, ale při menším vzorku můžou být nepřesné.
– příklad interpretace:
- Pro první korelaci 0.0001 (velmi malá) → statisticky velmi významná.
- Pro druhou korelaci 0.023 → už nevýznamná

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

```
      Wilks' Lambda, using F-approximation (Rao's F):
                    stat   approx df1 df2      p.value
      1 to 2:  0.5118282 9.148872   4  92 2.875983e-06
      2 to 2:  0.9251131 3.804601   1  47 5.708922e-02
```

- Řádek 1 to 2: testuje se hypotéza, že oba kanonické korelační koeficienty jsou nulové.
    - Wilks’ Lambda = 0.512
    - p-hodnota ≈ 2.9e-06 → velmi významné.
    - Znamená to, že alespoň první kanonická korelace je statisticky významná.

- Řádek 2 to 2: testuje se hypotéza, že druhá kanonická korelace je nulová (poté, co už jsme vzali v úvahu první).
    - p-hodnota ≈ 0.057 → na 5% hladině už to nevychází jako významné.
    - Znamená to, že druhou kanonickou korelaci 0.274 interpretovat nemusíš, protože se statisticky neliší od nuly.

1.6.2 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.
– často se doporučují, pokud není vzorek velký nebo když jsou pochybnosti o splnění předpokladů

```
# Permutační test významnosti (výpočetně náročnější)
set.seed(123)
p_perm <- p.perm(X, Y, nperms = 1000)  # 1000 permutací
print("Permutační p-hodnoty:")
print(p_perm)
```

2. Regresní modely

– jakou metodu vybrat:

- **Závislá proměnná spojitá (číselná)**
  - ***Lineární regrese*** – pokud vztahy mezi prediktory a Y jsou lineární, residualy ~ $N(0,\sigma^2)$.
  - ***Transformace*** / ***nelineární regrese*** – pokud vztah není lineární.
  - ***Regularizace*** (***Ridge***, ***Lasso***) – pokud máš hodně prediktorů nebo multikolinearitu.

- **Závislá proměnná binární (0/1, ano/ne)**
  - ***Logistická regrese*** – nejběžnější volba.
  - ***Penalizovaná logistická regrese*** – pokud hodně prediktorů.

- **Závislá proměnná kategoriální (více než 2 tříd)**
  - ***Multinomická logistická regrese***.
  - ***LDA*** / ***QDA*** – pokud jsou splněny jejich předpoklady.

- **Počítaná data (počty výskytů)**
  - ***Poissonova regrese*** – když je střední hodnota ≈ rozptyl.
    - `(mean(road$deaths)`
    - `var(road$deaths)`
  - ***Negativně binomická regrese*** – když je nadměrná disperze (rozptyl > střední hodnota).

2.1 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

$$\mathrm{MAE} = \frac{1}{n}\sum |y_i - \hat{y}_i|$$

- Mean Absolute Error
- Průměrná velikost chyby v jednotkách výstupu
- vyčíslení rozdílu mezi skutečnými hodnotami a predikcemi modelem
- Vezme se absolutní hodnota rozdílu mezi skutečnou a predikovanou hodnotou (ignoruje znaménko).
- Sečtou se tyto rozdíly a zprůměrují.
- Míň citlivý na velký chyby

$$\mathrm{MSE} = \frac{1}{n}\sum (y_i - \hat{y}_i)^2$$

– MSE

- Mean Squared Error
- vyčíslení rozdílu mezi skutečnými hodnotami a predikcemi modelem
- Vyzdvihuje větší chyby – čím větší chyba, tím víc se penalizuje.

– RMSE

$$\mathrm{RMSE} = \sqrt{\mathrm{MSE}}$$

- Root Mean Squared Error
- vyčíslení rozdílu mezi skutečnými hodnotami a predikcemi modelem
- MSE v původních jednotkách

– Akaikeho informační kritérium (AIC)

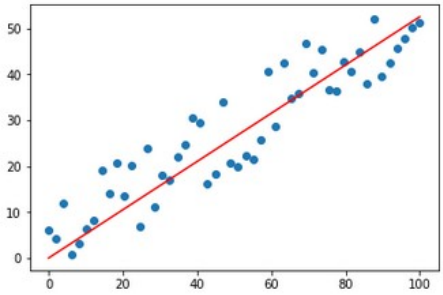
- Penalizuje složité modely (míň než BIC ale)
- Nižší hodnota = lepší model
- GPT vysvětlení (dávám to sem protože z tohohle jsem to náhle pochopil i já xd)
  - Máš dvě pizzy – jedna chutná skvěle, ale má 12 ingrediencí (zbytečně složitá), druhá skoro stejně chutná a má jen 5 surovin. AIC vybere tu druhou.

– Bayesovské informační kritérium (BIC)

- Jako AIC
- Trestá složitost modelu víc než AIC
- Nižší hodnota = lepší model

2.2 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)

2.2.1 Jednoduchá lineární regrese

– jedna závislá a jedna nezávislá proměnná
– jednoduchá lineární regrese se používá, pokud mám přírodní proměnnou (hodnoty tlaku krve, výška…)
– model má tvar: $y = \beta 0 + \beta 1 Xi + ei$

2.2.2 Mnohonásobná lineární regrese

– jedna závislá a více nezávislých proměnných
– model má tvar: $y = \beta 0 + \beta 1 X1i + \beta 2 X2i + \cdot \cdot \cdot + \beta k Xki + ei$

– předpoklady:

- Lineární vztah mezi závislými a nezávislými prom.
- Nezávislost, homoskedasticita a normální rozdělení reziduí
- Žádná nebo minimální multikolinearita mezi nezávislými prom.

2.2.3 Zobecněné lineární regresní modely

– když nemám splněné předpoklady lineární regrese, ale data pořád vypadají že budou mít lineární vztah
– obecně matematicky fungují tak, že levá strana rovnice lineární regrese se obalí nějakou funkcí

- Příklad: $log(Y) = \beta0 + \beta1Xi + ei$

– Poissonova regrese

- Pokud mám četnosti nečeho
- $log(Yi) = \beta0 + \beta1X1i + \beta2X2i + \cdot\cdot\cdot + \beta kXki + ei$

– Negativní binomická regrese

- Pokud mám četnosti nečeho ale není splněn předpoklad disperze poissonovi regrese
- $log(Yi) = \beta0 + \beta1X1i + \beta2X2i + \cdot\cdot\cdot + \beta kXki$
- Rovnice neobsahuje náhodu, ta je vyřešena přímo v

– Logistická regrese

- Pokud mě zajímá binární výsledek (ano/ne)
- $log(p/1-p) = \beta0 + \beta1X1i + \beta2X2i + \cdot\cdot\cdot + \beta kXki$
- Modeluje PRAVDĚPODOBNOST VÝSLEDNÉ HODNOTY

2.2.4 Interpretace koeficientů

– výsledné koeficienty modelů (hlavně v R - summary()) jsou logaritmem opravdového koeficientu (kvůli tvaru rovnice)
– pokud chceme interpretovat koeficienty musíme brát exponent koeficientu exp(puvodni_log_koeficient)

– pokud mi vyjde výsledný exponent např. 1.91 tak to znamená:

- Pro Poisson, negativní binomická:
  - Že se závislá proměnná zvedne o 91% (*1.91)

- Pro Logistickou:
  - Nárust nebo klesání pravděpodobnosti výsledku
  - Šance na pozitivní výsledek (pokud máme no/yes) je o 91% větší

Jak číst v summary v R a složit rovnici regresního modelu

```
Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  5.6586927  0.3810901  14.849  < 2e-16 ***
drivers      0.0021422  0.0006268   3.418 0.000632 ***
fuel         0.0041162  0.0014580   2.823 0.004755 **
popden      -0.0017765  0.0008718  -2.038 0.041568 *
temp         0.0071314  0.0077198   0.924 0.355599
```

- Řekněme že zde byla využita poissonova regrese:
  - hodnoty ve sloupci estimate jsou moje regresní koeficienty $\beta1,2,3,...,$ intercept je to $\beta0$
  - zapisuji i hodnoty, které nemají statistický význam (hvězdičky, p-hodnota pod 0.05)
  - $log(Yi) = 5.6586927 + 0.0021422*DRIVERS(i) + 0.0.0041162*FUEL(i) - 0.0017765*POPDEN(i) + 0.0071314*TEMP(i)$

– Příklad: závisí spotřeba (mpg) na hmotnosti auta (wt) a výkonu (hp)?

```
model <- lm(mpg ~ wt + hp, data = mtcars)
par(mfrow = c(2,2))
plot(model)
```

- Residuals vs Fitted (linearitu a homoskedasticitu) :
  - Nesmí být oblouk / U - nelinearita.
  - Zmenšující/Zvětšující rozptyl - heteroskedasticita.

- Q-Q residuals (normalita rezidui):
  - Body leží na přímce. Odchylky na krajích - špatná normalita

- Scale-location (homoskedasticita, konstantní rozptyl)
  - červená čára musí být přibližně rovná. Stoupá/klesá - problém

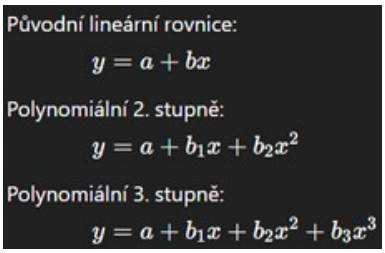
- Residuals vs leverage (vlivné body, Cookova vzdálenost)
  - Body mimo čárkované linie jsou problém

- Pokud selže -> transformace (log(závislá proměnná))
  Nebo zvolit nelineární metodu

2.3 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:



- 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)

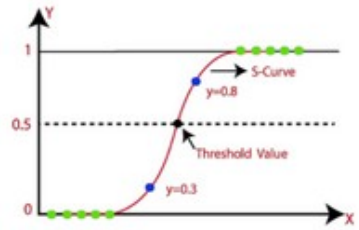
– jak zvolit jaký řád?

```
m1 <- lm(mpg ~ poly(wt, 1), data = mtcars)
m2 <- lm(mpg ~ poly(wt, 2), data = mtcars)
m3 <- lm(mpg ~ poly(wt, 3), data = mtcars)

anova(m1, m2, m3)
AIC(m1,m2,m3)
# čím nižší tím lepší
```

2.4 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.

– rovnice logistické regrese modeluje pravděpodobnost výsledku:

$$\log\left(\frac{P(\text{decision}=1)}{1-P(\text{decision}=1)}\right) = \text{lineární kombinace prediktorů}$$

– 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
  - Yi říká, kterou stranu kontrolujeme:
  - Yj (stříška) je třeba 70% (0,7) -> na 70% jsme si jistý, že je to spam
  - Yi= 1 (reálně to je spam) -> 1 * log(0.7) + 0 * log(1-0.7) -> 1 * log (0.7)
  - Yi= 0 (reálně to je nespam) -> 0 * log(Yi) + 1 * log(1-Yi) -> 1 * log(1-0.7)

- logaritmus má vlastnost:
  - Když je číslo moc blízko 1 -> log je blízko 0 (malá chyba)
  - Když je číslo moc blízko 0 -> log jde do mínus nekonečna (Obrovská chyba)
  - V našem případě, kdyby Yi bylo 1.. tak máme malou chybu log(0.7)
  - Kdyby Yi byla 0, tak máme log (0.3) což se bude blížit -nekonečnu - velká chyba

Hodnocení modelu

– Akaikeho informační kritérium (AIC)

- Penalizuje složité modely
- Nižší hodnota = lepší model

– R^2

- Jak moc model vysvětluje variabilitu výstupu

– Matice záměn (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.

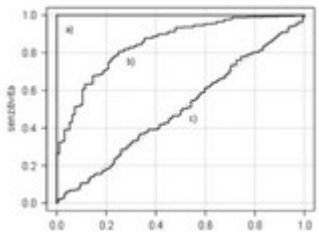
$$RC = \frac{TP}{TP + FN}$$

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

```
# Načtení dat (příklad s vestavěnými daty 'mtcars')
data <- mtcars

# Logistická regrese: predikce převodovky podle hmotnosti a výkonu
# am je převodovka 1 = automat, 0 = manuál
model <- glm(am ~ wt + hp, data = data, family = binomial)

# Shrnutí modelu
summary(model)

# Exponent koeficientů (OR)
exp(coef(model))

# Predikce pravděpodobností
predict(model, type = "response")
# predikce pro každý typ auta, zda am = 1 (druhá hodnota v levels = c(0,1))
```

2.4.1 Ordinální logistická regrese

– závislá proměnná má pořadové kategorie (např. 1 = nesouhlasím, 2 = spíše nesouhlasím, …, 5 = zcela souhlasím).
– místo předpovědi konkrétní hodnoty model odhaduje pravděpodobnost, že výsledek spadne do určité kategorie nebo nižší.
– např.: $P(Y \leq 2) P(Y \leq 2), P(Y \leq 3) P(Y \leq 3)$, atd.
– tyto kumulativní pravděpodobnosti se modelují pomocí logitové funkce.

– ***modelová rovnice*** (pro kategorii jjj):
- $logit(P(Y \leq j)) = \theta j - \beta 1 X 1 - \beta 2 X 2 - \cdots - \beta k X k$
- θj = „cutpoints“ (hranice mezi kategoriemi).
- β = vliv prediktorů.
- znaménko koeficientu říká, zda prediktor zvyšuje pravděpodobnost být v vyšších nebo nižších kategoriích.

– interpretace:
- pokud je $\beta > 0$, vyšší hodnota prediktoru → vyšší šance, že odpověď bude ve vyšší kategorii (např. „víc spokojený“).
- pokud $\beta < 0$, prediktor snižuje pravděpodobnost vyšší kategorie.

– předpoklady:
- závislá proměnná má pořadové kategorie (ne čistě nominální)
- proportional odds assumption: vliv prediktorů je stejný napříč všemi hranicemi (β je stejné pro všechna θj)
- pozorování jsou nezávislá.

– funguje podobně jako logistická regrese, ale pro více než 2 kategorie s pořadím

Příklad v R

```
# Příklad: vliv věku a pohlaví na spokojenost (1–5)
library(MASS)

# simulovaná data
set.seed(123)
data <- data.frame(
  spokojenost = factor(sample(1:5, 100, replace = TRUE), ordered = TRUE),
  vek = rnorm(100, 40, 10),
  pohlavi = factor(sample(c("M", "F"), 100, replace = TRUE))
)

# Ordinální logistická regrese
model <- polr(spokojenost ~ vek + pohlavi, data = data, Hess = TRUE)
summary(model)

# P-hodnoty
coef_table <- coef(summary(model))
p_values <- pnorm(abs(coef_table[, "t value"]), lower.tail = FALSE) * 2
cbind(coef_table, "p value" = p_values)

> cbind(coef_table, "p value" = p_values)

              Value Std. Error    t value     p value
vek       0.01431166  0.0176152  0.8124605 0.416527420
pohlaviM  0.38486890  0.3557840  1.0817487 0.279364222
1|2      -0.55792557  0.7770925 -0.7179654 0.472778605
2|3       0.41520887  0.7784195  0.5333999 0.593756794
3|4       1.36161509  0.7859474  1.7324508 0.083193336
4|5       2.24307182  0.8012655  2.7994114 0.005119586

# Pravděpodobnosti pro konkrétní hodnoty prediktorů
new_data <- data.frame(
  vek = c(30, 30, 50, 50),
  pohlavi = factor(c("M", "F", "M", "F"))
)

# Predikce pravděpodobností pro každou kategorii
predict(model, newdata = new_data, type = "probs")
```

- Závěr: v tomto datasetu věk ani pohlaví významně nevysvětlují rozdíly ve spokojenosti (Velké p-hodnoty).

Cut-points (1|2, 2|3, 3|4, 4|5)

– tyto hodnoty (θj\theta_jθj) jsou hranice mezi kategoriemi spokojenosti.
– interpretují se takto:

- ***1|2 = –0.558*** → logitová hranice mezi kategorií ≤1 a ≥2.
- ***2|3 = 0.415*** → hranice mezi kategorií ≤2 a ≥3.
- ***3|4 = 1.362 (p = 0.083)*** → hraniční hodnota mezi kategorií ≤3 a ≥4 (blízko významnosti, ale hranice samy o sobě nejsou obvykle hlavním zájmem).
- ***4|5 = 2.243 (p = 0.005)*** → hranice mezi kategorií ≤4 a =5.

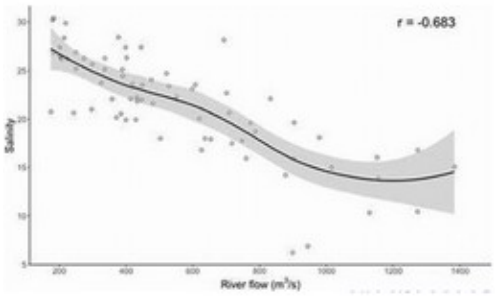
```
> predict(model, newdata = new_data, type = "probs")
          1         2         3         4         5
1 0.2022736 0.1992766 0.2319811 0.1731912 0.1932774
2 0.2714506 0.2250143 0.2210698 0.1422765 0.1401889
3 0.1599800 0.1751122 0.2298301 0.1932381 0.2418396
4 0.2186573 0.2068088 0.2306497 0.1655228 0.1783614
```

- Řádek 1 (30 let, muž)
  - Nejvyšší pravděpodobnost je u kategorie 3 (23,2 %) → muž 30 let má největší šanci spadnout do „střední“ spokojenosti.

- Řádek 2 (30 let, žena)
  - Největší pravděpodobnost je taky kategorie 2 (22,5 %) a 1 (27,1 %) → ženy mají při stejném věku o něco vyšší pravděpodobnost nižších kategorií než muži.
  - (Ale je to jen pro ilustraci, věk ani pohlaví nebyli významné)

2.5 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.

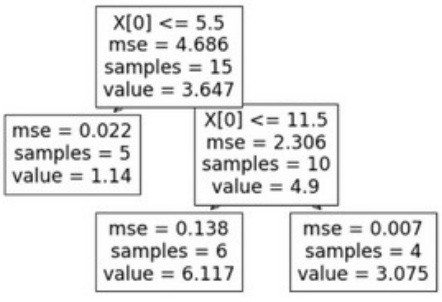
2.6 Decision tree regrese

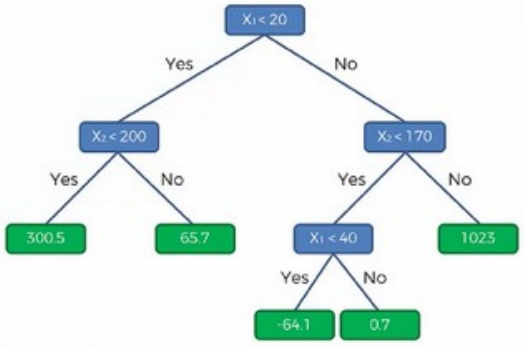
– model, který rozděluje vstupní prostor do oblastí podle hodnot vstupních proměnných
– každé rozdělení (uzel) klade otázku typu „je hodnota menší než...?“ a na základě odpovědi tě pošle dolů buď doleva, nebo doprava.
– cílem je předpovědět číselnou hodnotu, ne kategorii.

– jak funguje:
- 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:
- 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)





2.7 Random forest regrese

– kolekce několika decision trees
– řeší problém Decision tree - náchylnost na malou změnu dat

– jak 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)
- 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.

2.8 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:

$$\min\left(\frac{1}{2}\|w\|^2 + C\sum \text{chyby mimo } \varepsilon\right)$$

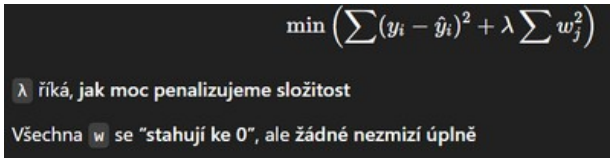
- C = důraz na chybu mimo ε
- w = vektor koeficientů modelu

– 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.

2.9 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: 

– 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

3. Náhradní otázky

3.1 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
  - hodně 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)

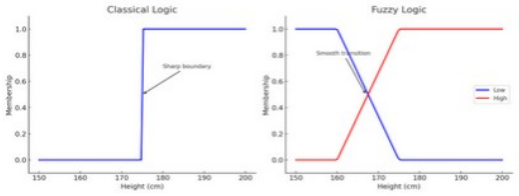
Pokud máme jednoduchou lineární regresi, tak:

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

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

3.2 Fuzzy modely

3.2.1 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.
- 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)

3.3 Bayesovské sítě

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

3.3.1 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

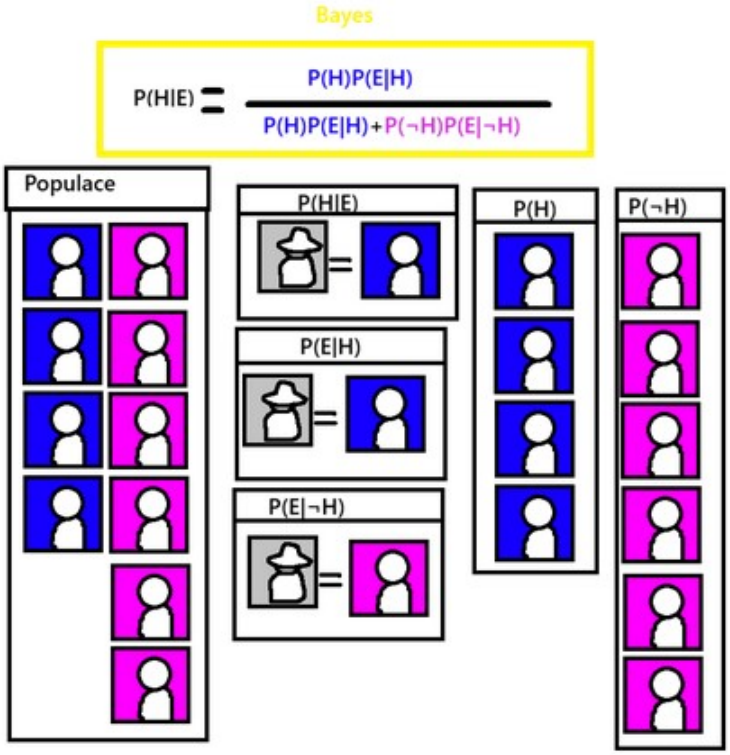
Příklad

– V naší skupině je 1 špion, je to muž? Ve skupině je 10 lidí, 4 jsou muži.

– 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 (pohlaví)
- ***E*** = Evidence - důkaz (špion)

- ***P(H|E)*** = Pravděpodobnost, že platí hypotéza(je to muž), pokud platí důkazy(je špion) (pravděpodobnost, že když je špion, tak je muž)
- ***P(E|H)*** = Pravděpodobnost, že platí důkazy(je špion), pokud platí hypotéza(je to muž) (pravděpodobnost, že když je muž, tak je špion)
- ***P(E|¬H)*** = Pravděpodobnost, že platí důkazy(je špion), pokud neplatí hypotéza (není muž) (pravděpodobnost, že když je žena, tak je špion)
- ***P(H)*** = Pravděpodobnost, že platí hypotéza (vybraný je muž)
- ***P(¬H)*** = Pravděpodobnost, že neplatí hypotéza (vybraný není muž)

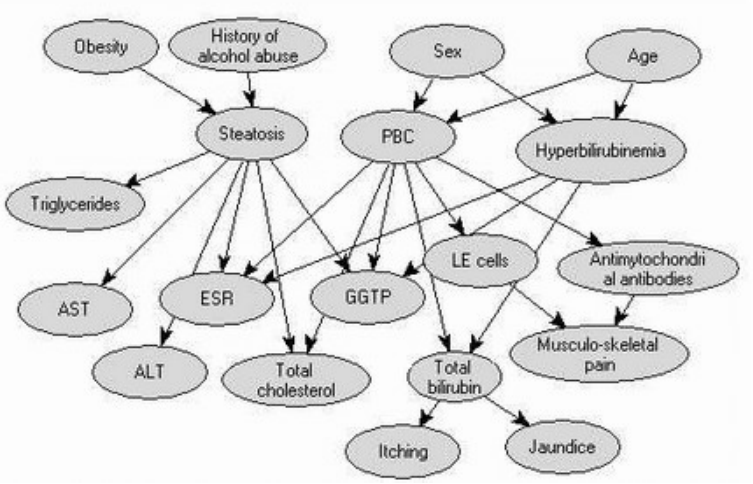


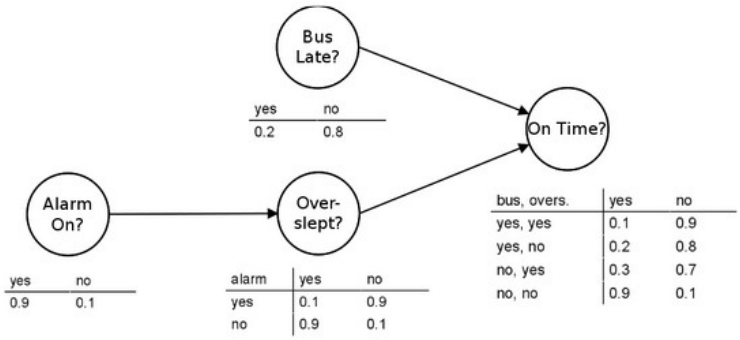
3.3.2 Bayesovské sítě

– 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





4. Last minute doplnění

1. MANOVA (Multivariate Analysis of Variance)

– co to je:

- Rozšíření ANOVA pro vícero závislých proměnných.
- Testuje, zda se vektor průměrů závislých proměnných liší mezi skupinami.

– vzorec:

- Testuje se hypotéza:
  H0:μ1=μ2=⋯=μkH_0: \mu_1 = \mu_2 = \cdots = \mu_kH0:μ1=μ2=⋯=μk
  kde μi\mu_iμi jsou vektorové střední hodnoty skupin.
- Hlavní statistiky: Wilks' Lambda, Pillai’s Trace, Hotelling-Lawley trace, Roy’s greatest root.

– matematika:

- Rozkládá se celková kovarianční matice na meziskupinovou (B) a vnitroskupinovou (W):
  T=B+WT = B + WT=B+W
- Poměr mezi těmito dvěma maticemi určuje statistiku testu.

– analogie:

- Představ si, že porovnáváš studenty ze 3 škol podle výkonu z matiky, fyziky a chemie. Nechceš porovnat předměty odděleně, ale najednou. MANOVA ti umožní zjistit, zda školy ovlivňují kombinovaný výkon studentů.

2. Hotellingův T² test

– co to je:

- Multivariantní obdoba t-testu. Testuje rozdíl mezi dvěma vektory průměrů.
- Používá se u dvou skupin, kde máme více závislých proměnných.

– vzorec:

- T2=n(x⁻−μ)TS−1(x⁻−μ)T^2 = n (\bar{x} - \mu)^T S^{-1} (\bar{x} - \mu)T2=n(x⁻−μ)TS−1(x⁻−μ)
  - x⁻\bar{x}x⁻ = průměr vektoru,
  - μ\muμ = hypotetická hodnota (např. nulový vektor),
  - SSS = kovarianční matice,
  - nnn = počet pozorování.

– matematika:

- Měří vzdálenost mezi průměry pomocí Mahalanobisovy vzdálenosti (bere v úvahu korelace mezi proměnnými).

– analogie:

- Představ si, že máš průměrné skóre z více testů (matika, čtení, psaní) pro dvě školy. Chceš vědět, jestli se celkový profil liší – nejen v jednotlivých testech, ale v celkové kombinaci. Hotelling T² ti řekne, zda jsou školy rozdílné ve všech testech dohromady.

3. PCA (Principal Component Analysis – metoda hlavních komponent)

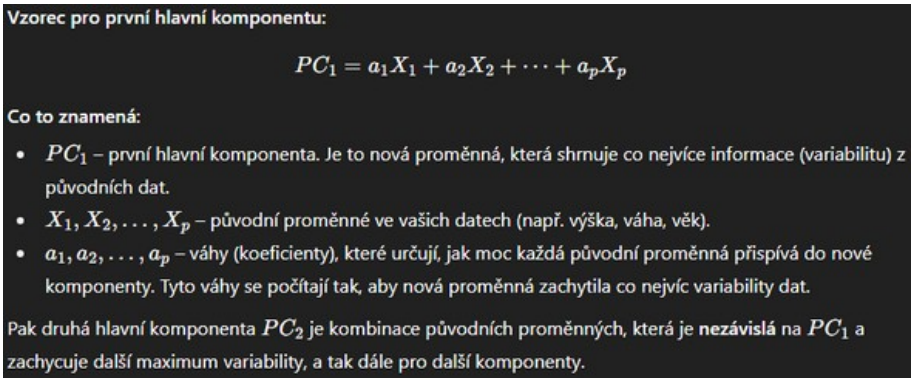
– co to je:

- Metoda pro redukci dimenze. Převádí korelované proměnné na menší sadu nepřímo souvisejících (ortogonálních) proměnných – hlavní komponenty.

– vzorce:

- Výpočet vlastních čísel a vlastních vektorů kovarianční nebo korelační matice:

- Sv=λvS v = \lambda vSv=λv
  - vvv = vlastní vektor (směr hlavní komponenty),
  - λ\lambdaλ = vlastní číslo (variance podél dané komponenty).



– matematika:

- První komponenta maximalizuje rozptyl.
- Druhá maximalizuje rozptyl ortogonálně k první, atd.
- Výsledkem jsou lineární kombinace původních proměnných.

– analogie:

- Představ si, že máš fotku s vysokým rozlišením (1000x1000 pixelů). PCA je jako komprese – zachová podstatnou informaci v menším rozměru (např. jen 100 hlavních prvků), aniž bys výrazně ztratil kvalitu.

4. Faktorová analýza

– co to je:

- Podobné jako PCA, ale cílem je najít latentní proměnné (faktory), které způsobují závislosti mezi pozorovanými proměnnými.

– model:

- X=ΛF+ϵX = \Lambda F + \epsilonX=ΛF+ϵ
  - XXX: vektor pozorovaných proměnných
  - Λ\LambdaΛ: matice faktorových zátěží
  - FFF: vektor latentních faktorů
  - ϵ\epsilonϵ: specifická chyba

$$X_j = \lambda_{j1}F_1 + \lambda_{j2}F_2 + \cdots + \lambda_{jm}F_m + \epsilon_j$$

Co znamená:

- X_j – původní proměnná.
- $F_1, F_2, \ldots, F_m$ – **skryté faktory**, které vysvětlují společnou variabilitu mezi proměnnými.
- λ_{jk} – **faktorové zátěže** (váhy), určují, jak moc každý faktor přispívá k dané proměnné.
- ϵ_j – unikátní variabilita (šum) proměnné, kterou faktory nevysvětlují.

Lajcky: FA říká "Existují pár skrytých faktorů, které ovlivňují všechny proměnné, a každá proměnná je kombinací těchto faktorů + něco, co je jen její vlastní."

– matematika:

- Vysvětluje kovariance mezi proměnnými pomocí menšího počtu faktorů.
- Hledají se matice tak, aby rekonstruovaly původní kovarianční strukturu.

– analogie:

- Jako když z několika symptomů (kašel, horečka, únava) odhaduješ skrytou nemoc. Symptomy jsou pozorované proměnné, nemoc je latentní faktor.

5. Diskriminační analýza (LDA – Linear Discriminant Analysis)

– co to je:
- Supervizovaná metoda pro klasifikaci – hledá lineární kombinaci prediktorů, která co nejlépe odděluje skupiny.

– vzorec:
- Hledáme www, aby:
  w=SW−1(μ1−μ2)w = S_W^{-1}(\mu_1 - \mu_2)w=SW−1(μ1−μ2)
  - SWS_WSW: vnitroskupinová kovarianční matice
  - μ1,μ2\mu_1, \mu_2μ1,μ2: průměry tříd

– matematika:
- Maximalizuje poměr meziskupinového rozptylu k vnitroskupinovému – Fisher criterion.

$$Y = w_1X_1 + w_2X_2 + \cdots + w_pX_p$$

Co znamená:

- Y – nová proměnná (diskriminantní funkce), podle které se snažíme rozlišit třídy.
- $X_1, X_2, \ldots, X_p$ – původní proměnné.
- $w_1, w_2, \ldots, w_p$ – váhy, které maximalizují **poměr mezi-třídní variability k uvnitř-třídní variabilitě.**

Lajcky: LDA vezme původní proměnné a udělá z nich novou osu (nebo několik os), která **co nejlépe odděluje třídy.**

– analogie:
- Jako když se snažíš rozdělit studenty na "prospěl" vs. "neprospěl" na základě více známek. LDA ti najde nejlepší hranici (čáru nebo rovinu), která skupiny co nejlépe rozdělí.

6. Hierarchické shlukování (Hierarchical Clustering)

– co to je:
- Ne-supervizovaná metoda, která spojuje objekty do hierarchicky uspořádaných shluků (clusterů) podle jejich podobnosti.

– postup:
- Začni s každým objektem jako samostatným clusterem.
- Postupně slučuj nejbližší clustery podle zvolené metriky (např. průměr, nejbližší soused).
- Výsledek je dendrogram.

– metriky vzdálenosti:
- Eukleidovská vzdálenost
- Manhattanova vzdálenost
- Mahalanobisova vzdálenost

– metody spojování:
- Single linkage (nejbližší bod)
- Complete linkage (nejvzdálenější bod)
- Average linkage (průměr)

– analogie:
- Představ si, že máš skupinu lidí a chceš je rozdělit podle podobnosti zájmů. Začneš tak, že každý je sám. Pak postupně spojuješ dvojice, které mají nejvíc společného. Nakonec dostaneš strom, který ti ukazuje, jak jsou si lidé podobní.