Univerza *v Ljubljani*
Fakulteta *za računalništvo*
in informatiko

Delovni zapiski izobraževanja

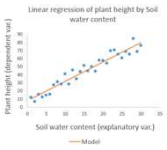
BAYESOVA STATISTIKA S PROGRAMSKIM JEZIKOM STAN

Izvajalca: izr. prof. dr Erik Štrumbelj
dr. Jure Demšar

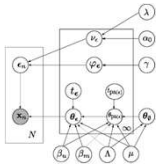
Akademija FRI

Zakaj naj mi bo mar za probabilistično programiranje?

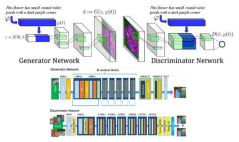
- Temelj statističnega modeliranja in probabilističnega strojnega učenja.
- Prihodnost "podatkovnega inženirstva".
- Obvezno orodje za vsakega, ki se želi resno ukvarjati s kvantitativno analizo podatkov!



uporabna statistika



probabilistični grafični modeli



(generativno) globoko učenje

2

Oris vsebine

1 Negotovost in probabilistično razmišljanje,

2 statistično modeliranje,

3 probabilistično programiranje,

4 programski jezik Stan,

5 praktični del.

Predpostavljamo znanje programiranja in osnovno razumevanje verjetnosti.

3

Interaktivni test opreme za delavnico

4

1 del

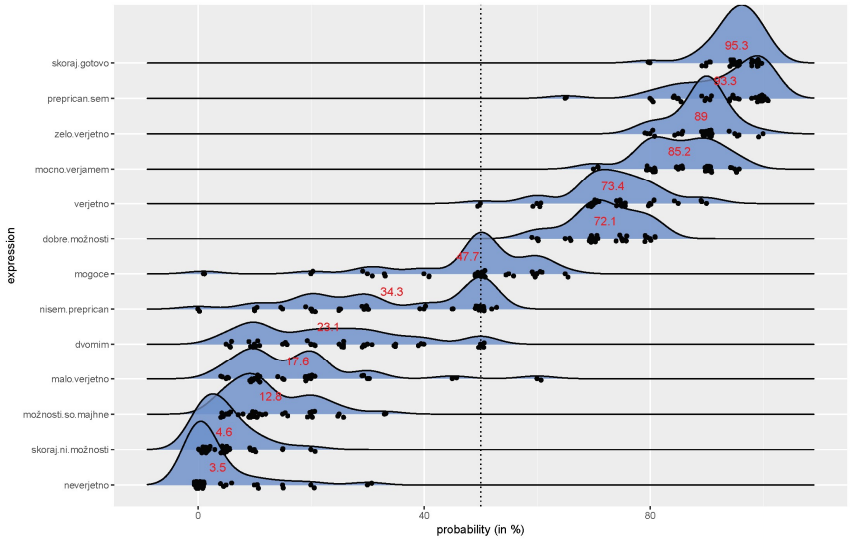
Negotovost in probabilistično razmišljanje

5

Q: Ali bo naslednji teden v Ljubljani deževalo?

6

Probabilistični izrazi v naravnem jeziku



7

Q: Kako toplo (°C) bo jutri opoldne v Ljubljani?

8

Naravni jezik je nekonsistenten, nenatačnen in premalo ekspresiven za resno kvantitativno delo!

- **Dobra novica** Primeren jezik so že razvili!
- **Slaba novica** Gre za teorijo verjetnosti – matematiki se ne moremo izogniti.
- **Dobra novica** Ni se nam potrebno naučiti niti vse dodiplomske verjetnosti[1] – potrebujemo le verjetnost kot jezik, računal pa bo računalnik.

[1] Kar pa ne pomeni, da nam ne bo koristilo! Verjetnost je osnova kvantitativne analize podatkov.

9

Gramatika verjetnosti

Verjetnost P (pogosto Pr) je funkcija, ki dogodkom prireja numerične vrednosti in zadošča tem aksiomom:

A1 $P(A) \geq 0$.

A2 $P(\Omega) = 1$.

A3 $P(A_1 \cup A_2 \cup A_3 \cup ...) = \sum_{i=1}^{\infty} P(A_i)$, za poljubno sekvenco disjunktnih dogodkov.

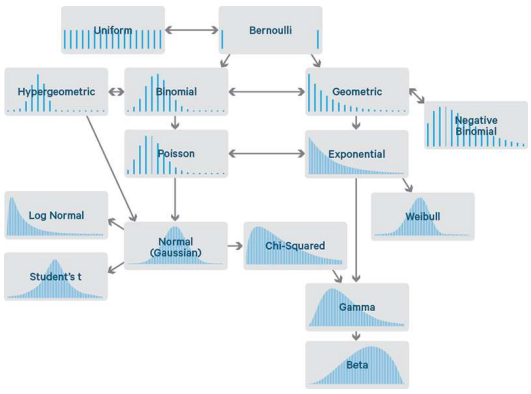
Definicija **pogojne verjetnosti**:

$$P(A|B) = \frac{P(A \cap B)}{P(B)}$$

10

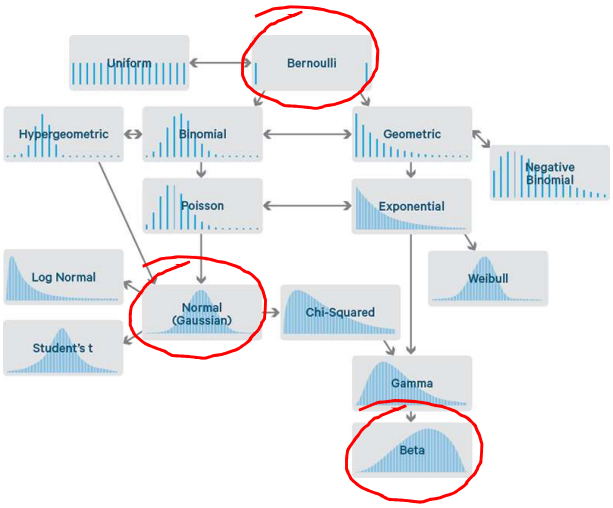
Porazdelitve

- Porazdelitve so elementarni izrazi probabilističnega razmišljanja in
- osnovni gradniki statističnih modelov.
- Porazdelitve so v skladu s pravili teorije verjetnosti, zato so konsistentne in natančne probabilistične izjave.
- Več kot vemo o porazdelitvah, bolj bogato se lahko izražamo.



11

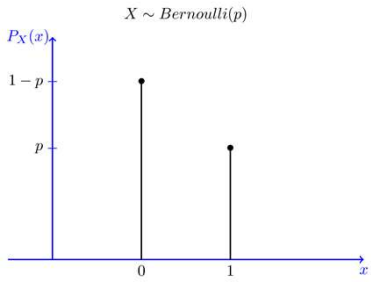
Beseda na dan ...



12

Bernoullijeva porazdelitev

distribution	pmf	mean	variance
Bernoulli(p)	$p^x(1-p)^{1-x};\ x=0,1;\ p\in(0,1)$	p	$p(1-p)$

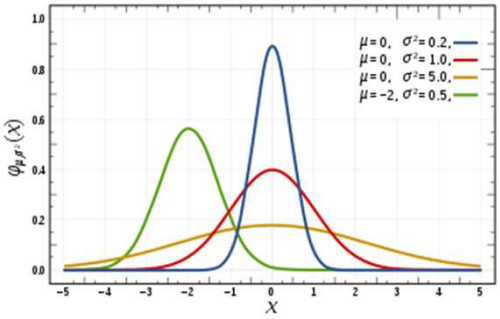


Q: Ali bo naslednji teden v Ljubljani deževalo?

13

Normalna (Gaussova) porazdelitev

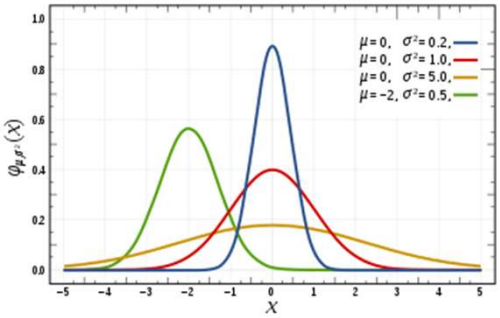
distribution	pdf	mean	variance
Normal(μ, σ^2)	$\frac{1}{\sqrt{2\pi}\sigma} e^{\frac{(x-\mu)^2}{2\sigma^2}};\ \sigma > 0$	μ	σ^2



14

Normalna (Gaussova) porazdelitev

distribution	pdf	mean	variance
Normal(μ, σ^2)	$\frac{1}{\sqrt{2\pi}\sigma} e^{\frac{(x-\mu)^2}{2\sigma^2}};\ \sigma > 0$	μ	σ^2

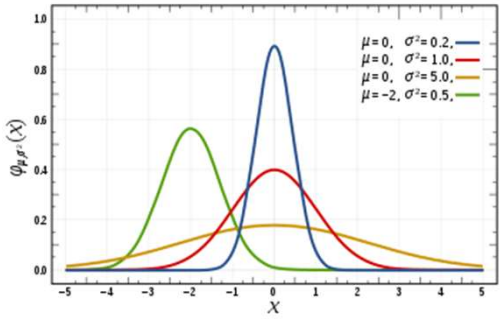


Q: Kako toplo (°C) bo jutri opoldne v Ljubljani?

15

Normalna (Gaussova) porazdelitev

distribution	pdf	mean	variance
Normal(μ, σ^2)	$\frac{1}{\sqrt{2\pi\sigma}} e^{\frac{(x-\mu)^2}{2\sigma^2}};\ \sigma > 0$	μ	σ^2

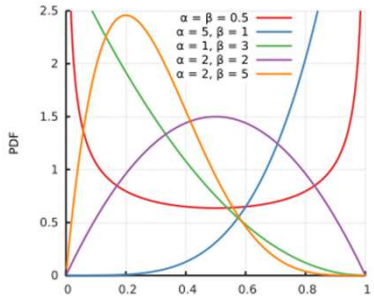


Q: Kako toplo (°C) je bilo na današnji dan pred 50 leti?

16

Porazdelitev Beta

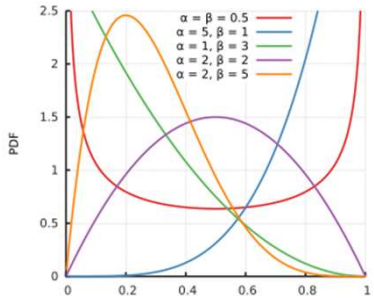
distribution	pdf	mean	variance
Beta(α, β)	$\frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} x^{\alpha-1}(1-x)^{\beta-1};\ x \in (0,1),\ \alpha, \beta > 0$	$\frac{\alpha}{\alpha+\beta}$	$\frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}$



17

Porazdelitev Beta

distribution	pdf	mean	variance
$\mathrm{Beta}(\alpha, \beta)$	$\frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} x^{\alpha\ 1}(1-x)^{\beta\ 1};\ x \in (0,1),\ \alpha, \beta > 0$	$\frac{\alpha}{\alpha+\beta}$	$\frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}$



Q: Kolikšna je verjetnost, da naslednji teden v LJ dežuje?

18

Preizkus probabilističnega razmišljanja

To so izidi 10 metov (morda nepoštenega) kovanca:

c c g c c g c c c g (?)

Q1: Je enajsti met **c**ifra ali **g**rb?

Q2: Kolikšna je verjetnost *p*, da na tem kovancu pade grb?

Q3: Je kovanec pošten? Poštenost je npr., da je *p* med 48% and 52%.

19

V razmislek ...

Verjetnost je koherenten in natančen jezik za izražanje negotovosti:

- Če ne sledimo zakonom verjetnosti, nas nihče ne bo razumel!
- Sicer pa so probabilistične izjave lahko subjektivne ali navidez popolnoma nesmiselne.
- Precej naravno nam je, da imamo verjetnostno mnenje o stvareh, ki niso naključne. Naključje je samo eden izmed virov negotovosti (in ne preveč pogost).

Uporaba verjetnosti za izražanje negotovosti je bistvo bayesovskega pogleda na statistično sklepanje!

20

2 del

Statistično modeliranje

21

Model =

Hipoteza, kako so nastali naši podatki.

22

Ni modeliranja brez modela.

23

Ne, resno, ni.

24

Q: Zapišite 1 metodo iz statistike ali strojnega učenja, ki se uporablja za napovedovanje, razpoznavanje vzorcev, gručenje, testiranje hipotez, ipd.

25

Zaporedje enic in ničel (= podatki**):**

1001001010110110011111111111101

Statistični model (= poskus statistične interpretacije**):**

Zaporedje je nastalo s 30 neodvisnimi meti kovanca z neznano verjetnostjo enice θ.

Predhodno mnenje o parametrih modela:

Nimam pojma, koliko je θ, zato ne bom izrazil preference do nobene vrednosti θ.

Statistično sklepanje (= učenje**):**

Pri vseh teh predpostavkah in upoštevajoč zakone verjetnosti, kakšno mora biti moje mnenje o θ, ko vidim podatke?

26

Zaporedje enic in ničel (= podatki**):**

1001001010110110011111111111101

$y_1, \ldots, y_n \qquad y_i \in \{0, 1\}$

Statistični model (= poskus statistične interpretacije**):**

Zaporedje je nastalo s 30 neodvisnimi meti kovanca z neznano verjetnostjo enice θ.

$y_1, y_2, \ldots, y_n | \theta \sim_{\text{iid}} Bernoulli(\theta)$

Predhodno mnenje o parametrih modela:

Nimam pojma, koliko je θ, zato ne bom izrazil preference do nobene vrednosti θ.

$\theta \sim Beta(1, 1)$

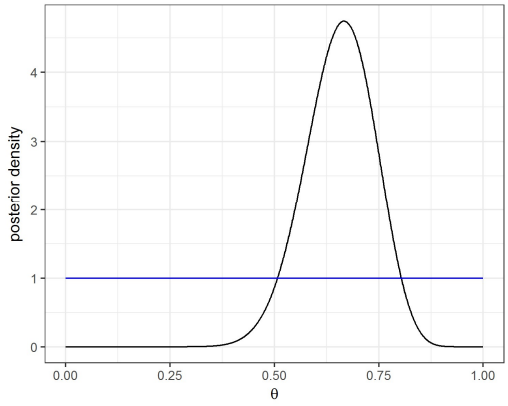
Statistično sklepanje (= učenje**):**

Pri vseh teh predpostavkah in upoštevajoč zakone verjetnosti, kakšno mora biti moje mnenje o θ, ko vidim podatke?

$$p(\theta|y) = \frac{p(\theta, y)}{p(y)} = \frac{p(y|\theta)p(\theta)}{\int p(y|\theta)p(\theta)d\theta}$$

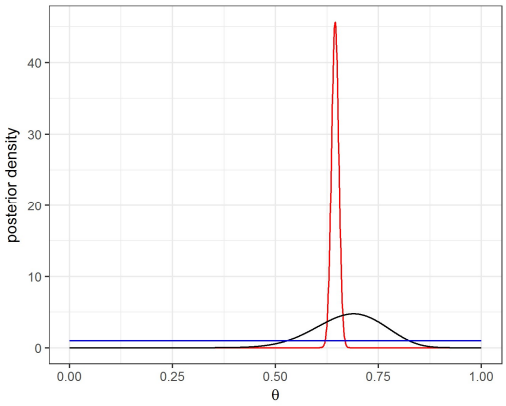
$$\theta|y_1, \ldots, y_n \sim Beta(\sum y_i + 1, n - \sum y_i + 1)$$

27



Naše mnenje o θ **prej** in **potem**, ko smo videli zaporedje, ki vsebuje 20 enic in 10 ničel.

28



Naše mnenje o θ **potem**, ko smo videli zaporedje, ki vsebuje 20 enic in 10 ničel.

29

3 del
Probabilistično programiranje

30

Probabilistični programski jezik (PPL) je programski jezik, ki je zasnovan za opisovanje probabilističnih modelov in računsko sklepanje iz teh modelov.

Vir: Wikipedia

31

Probabilistični programski jezik nam omogoča, da se osredotočimo na modeliranje in preskočimo matematične in računske probleme pri sklepanju.

32

Dva primera imperativnega programiranja

```
# Bubble Sort
sort <- function(x) {
  n <- length(x)

  for (k in n:2) {
    i <- 1
    while (i < k) {
      if (x[i] > x[i+1]) {
        temp   <- x[i+1]
        x[i+1] <- x[i]
        x[i]   <- temp
      }
      i <- i + 1
    }
  }
  x
}
```

```
# Generate 30 Bernoulli variables
bernoulli <- function(p) {
  x <- c()

  for (i in 1:30) {
    if (runif(1) > p) {
      x <- c(x, 0)
    } else {
      x <- c(x, 1)
    }
  }
  x
}
```

33

Imperativno programiranje in Statistično modeliranje

```
Source on Save
1  # Generate 30 Bernoulli variables
2  bernoulli <- function(p) {
3    x <- c()
4
5    for (i in 1:30) {
6      if (runif(1) > p) {
7        x <- c(x, 0)
8      } else {
9        x <- c(x, 1)
10     }
11   }
12   x
13 }
14
```

- Podane imamo vhodne podatke in parametre,
- sprogramiramo algoritem, ki generira zahtevane izhodne podatke.

```
#Sklepanje o relativni frekvenci tega zaporedja
10010010101101100111111111111101
```

$$y_1, \ldots, y_n \qquad y_i \in \{0, 1\}$$
$$y_1, y_2, \ldots, y_n | \theta \sim_{\text{iid}} Bernoulli(\theta)$$
$$\theta \sim Beta(1, 1)$$

- Podane imamo vhodne in izhodne podatke,
- opišemo generator, ki naj bi generiral podatke,
- sklepamo o najbolj verjetnih vrednostih parametrov.

34

Odmor

koda za drugi del:

https://github.com/bstatcomp/Stan-Intro-Workshop

35

4 del
Programski jezik Stan

36

Kaj je Stan?

- Orodje za učinkovito Bayesovo statistično modeliranje.
- Najlažje ga uporabljamo preko vmesnikov (na primer RStan, PyStan, ...).
- Stan je "compiled" jezik, to pomeni, da se statistični model preslika v c++ kodo, ki se nato pred uporabo prevede (zato je potrebno pred uporabo modela malo počakati).

37

Obvezni bloki vsakega Stan programa

- **data** – blok, v katerem s pomočjo spremenljivk deklariramo vhodne podatke. Vrednosti vhodnih podatkov pripravi uporabnik/razvijalec, običajno v programskem jeziku, ki ga uporabljamo kot vmesnik.
- **parameters** – blok, v katerem deklariramo parametre, ki jih želimo oceniti (kateri parametri našega statističnega modela nas zanimajo). Stan preko vmesnika (na primer RStan) vrne vrednosti parametrov nazaj v izhodiščni programski jezik.
- **model** – opis statističnega modela.

```
/*
Primer komentarja, ki obsega
več vrstic.
*/

data {
  // tukaj definiramo vhodne podatke
}

parameters {
  // parametri modela, ki jih želimo oceniti
}

model {
  // sem spada statistično modeliranje
}
```

38

Osnovni tipi spremenljivk

- **int** – celo število

```
int a;
```

- **real** – realno število

```
real b;
```

- **seznam** (array) – seznam celih ali realnih števil

```
int a[10]; real b[n];
```

- **matrika** (matrix) – 2D seznam [vrstice, stolpci]

```
int A[10, 10];
```

- **vector** – vektor realnih števil (optimiziran seznam)

```
vector[n] v;
```

- **simplex** – vektor pozitivnih realnih števil, ki se seštejejo v 1

```
simplex[n] s;
```

- (skoraj) vsem spremenljivkam lahko določimo zgornjo in spodnjo mejo

```
real<lower=0> sigma;
real<lower=0,upper=1> success_rate;
```

39

Porazdelitve

- **Bernoulli**
  - **y** je vektor "uspehov" (1) in "neuspehov" (0)
  - **θ** (theta) predstavlja verjetnost uspeha

```
y ~ bernoulli(theta);
```

- **beta**
  - **y** je vektor realnih števil med 0 in 1
  - **α, β** parametra porazdelitve

```
y ~ beta(alpha, beta);
```

- **normal**
  - **y** je vektor realnih števil
  - **μ, σ** sta upanje oziroma varianca

```
y ~ normal(mu, sigma);
```

- porazdelitve uporabimo tudi za vnašanje predznanja o določenih parametrih modela

```
theta ~ beta(1,1);
```

40

5 del
Praktični primeri

41

Izjemno kratek uvod v R

42

43

Izjemno kratek uvod v R

Jure Demšar

30/5/2019

R in RStudio

R (https://www.r-project.org/) je odprto kodni programski jezik namenjen primarno statisičnim izračunom. Osnovni vmesnik za delo z Rjem je precej okoren, zato si delo lahko precej olajšamo z namestitvijo drugega razvojnega okolja. Najbolj razširjeno ter razmeroma kvalitetno razvojno okolje se imenuje RStudio (https://www.rstudio.com).

Paketi

Ker je R odprto kodni programski jezik, lahko vsak prispeva k njegovem razvoju s pomočjo dodatnih paketov. Uporaba paketov nam močno olajša delo, večina uporabnih paketov se nahaja v repozitoriju CRAN. V RStudiu lahko pakete iz repozitorija CRAN naložimo preko menuja Tools in izbire Install Packages. Če poznamo ime paketa, ki ga želimo namestiti, lahko to naredimo tudi neposredno preko ukazne vrstice, na primer ukaz `install.packages("dplyr")` namesti paket dplyr. Ko paket namestimo, ga moramo pred uporabo še uvoziti v našo skripto, to storimo s pomočjo ukaza `library()`. Spodnja koda namesti ter uvozi vse pakete, ki jih bomo potrebovali pri reševanju današnjih nalog.

```
# install packages
install.packages("dplyr")
install.packages("ggplot2")
install.packages("mcmcse")
install.packages("reshape2")
install.packages("rstan")

# load packages into workspace
library(dplyr)
library(ggplot2)
library(mcmcse)
library(rstan)
library(reshape2)
```

Nalaganje podatkov

Podatki so običajno shranjeni v obliki .csv (angl. *comma separated value*) datotek. Vsaka vrstica v datoteki predstavlja eno meritev, v vsaki vrstici lahko najdemo več atributov te meritve, atributi so ločeni s posebnim znakom, ki v vrednostih atributov ne nastopa. V R podatke iz datoteke csv naložimo s pomočjo ukaza `read.csv()`. Privzeti znak za ločevanje atributov je vejica, zaradi praktičnih razlogov (zmeda z decimalno vejico/piko, vejice v tekstovnih podatkih) se vejica običajno ne uporablja. Delilni znak (angl. *separator*) lahko nastavimo z uporabo parametra `par`. Spodnja koda v spremenljivko `basket_data` naloži podatke iz datoteke `basketball_shots.csv`, ki se nahaja v direktoriju `data`.

```
# load data
basket_data <- read.csv("../data/basketball_shots.csv", sep=";")
```

Podatki se naložijo v podatkovno strukturo imenovano podatkovni okvir (angl. *data frame*). Ta podatkovni tip si lahko predstavljate kot tabelo, kjer stolpci predstavljajo atribute, vrstice pa kot primerke/meritve. V posameznih celicah se tako nahajajo vrednosti atributa za določen primerek.

S pomočjo konstruktorja `data.frame()` lahko naredimo tudi nov podatkovni okvir, operacija `c()` v spodnjem primeru predstavlja *combine* in združi več posameznih elementov v vektor.

```
# we can create new data frames like this, c() combines elements into a vector
new_df <- data.frame(first_column=c(1, 2), second_column=c(10, 11))
```

```
##   first_column second_column
## 1            1            10
## 2            2            11
```

Do posameznih stolpcev podatkovnega okvirja dostopamo z znakom $, do števila vrstic v okvirju pa pridemo z ukazom `nrow()`.

```
# to access columns of a data frame use the $ sign
new_df$first_column
```

```
## [1] 1 2
```

```
# to get number of rows in a data frame use nrow
nrow(new_df)
```

```
## [1] 2
```

Z ukazom `rbind()` lahko združimo več podatkovnih okvirjev v enega, pogoj je, da se okvirji ujemajo v vseh stolpcih.

```
# create another data frame
new_df2 <- data.frame(first_column=c(3, 4), second_column=c(12, 13))

# we can merge data frames together (if they have same columns) using rbind
new_df3 <- rbind(new_df, new_df2)
```

Pri pripravi podatkov za Stan moramo običajno iz podatkovnega okvirja izluščiti samo določen stolpec. Ko stolpec shranimo v spremenljivko, dobimo spremenljivko tipa vektor. Za izračun dolžine vektorja ne uporabljamo ukaza `nrow()`, saj vektor nima vrstic, ampak uporabimo ukaz `length()`.

```
# we can extract a column and save it into a vector
first_column <- new_df$first_column

# to get length of a vector use the length function
length(first_column)
```

```
## [1] 2
```

Ko pripravljamo podatke za Stan, jih shranimo v podatkovno struktoro `list`. Gre za seznam spremenljivk različnega tipa, v spodnjem primeru tako naredimo seznam, ki vsebuje celo število (`n`) ter vektor realnih števil (`y`).

```
# we pass data to stan through lists
n <- nrow(new_df)
y <- new_df$first_column
stan_data <- list(n = n, y = y)
```

Osnovne operacije nad podatki

Za potrebe delavnice bomo spoznali zgolj najbolj osnovne operacije za delo s podatki. Urejanje podatkov si bomo olajšali z uporabo knjižnice `dplyr`. V naših podatkih o metih na koš nastopa več igralcev, vsak izmed njih je 60 krat vrgel na običajen koš ter 60 krat na koš z manjšim obsegom obroča. Spodnji primer

prikazuje, kako iz vseh podatkov izluščimo samo podatke o metih prvega igralca (atribut `PlayerID` je enak 1) na običajen obroč (atribut `SpecialRim` je enak 0).

```
player1_data <- filter(basket_data, PlayerID == 1 & SpecialRim == 0)
```

Za vsak met imamo 5 podatkov: * ID igralca (*PlayerID*), * ID meta (*ThrowNum*), * kot izmeta (*Angle*), * uspešnost (*Made*), * tip obroča (*SpecialRim*).

Recimo, da za nadaljno analizo potrebujemo zgolj stolpca s podatki o zaporedni številki meta ter o uspešnosti meta, ostalih stolpcev se lahko znebimo z uporabo operacije `select`. Reduciranje števila stolpcev je še posebej koristno, če operiramo z ogromnimi količinami podatkov.

```
player1_data <- select(player1_data, ThrowNum, Made)
```

Operacijo `melt` iz paketa `reshape2` bomo uporabljali predsem za pripravo podatkov pred vizualizacijami. Spodnji primer prikaže uporabo melt, ko želimo primerjati zneske, ki so jih zagonska podjetja vložila v raziskave in razvoj, marketing in administracijo.

```
##   research marketing administration
## 1 165349.2  471784.1       136897.8
## 2 162597.7  443898.5       151377.6
## 3 153441.5  407934.5       101145.6
```

Osnovna uporaba ukaza `melt` nam podatke zreducira na 2 stolpca, poimenovana `variable` in `value`, kjer vsaka vrstica predstavlja eno celico (par `variable`/`value`) iz izhodiščnega podatkovnega okvirja. Tako pripravljene podatke lahko nato v vizualizacijah grupiramo po vrednosti atributa `variable`, medtem ko vrednosti (`value`) preslikamo na želeno os grafa.

```
melt_investments <- melt(investments)
```

```
##          variable    value
## 1        research 165349.2
## 2        research 162597.7
## 3        research 153441.5
## 4       marketing 471784.1
## 5       marketing 443898.5
## 6       marketing 407934.5
## 7  administration 136897.8
## 8  administration 151377.6
## 9  administration 101145.6
```

Cev (angl. *pipe*)

V kodi boste srečali tudi cev (`pipe` operator, `%>%`), s pomočjo tega operatorja lahko kodo precej skrajšamo. Operator nam omogoča, da podatke, ki so rezultat ene operacije, prenesemo neposredno v naslednjo operacijo (kot bi naredili cev med dvema operacijama, po cevi se rezultat prve operacije prenese na vhod druge operacije). Spodnji primer prikazuje, kako lahko združimo filtriranje in selektiranje podatkov iz zgornjega primera.

```
player1_data <- basket_data %>%
  filter(PlayerID == 1 & SpecialRim == 0) %>%
  select(player1_data, ThrowNum, Made)
```

Problem #1

V datoteki **basketball_shots.csv** so podatki o metih na koš. Vsak košarkaš je 60 krat vrgel na koš običajne velikosti, nato pa še 60 krat na koš z manjšim obsegom.

Zanima nas:

1) Primerjava med uspešnostjo košarkaša #1 ter košarkaša #2:
   - *Kdo je boljši?*
   - *Kako prepričani smo v to ugotovitev?*
   - *Kolikšna je razlika v uspešnosti?*

2) Primerjava med metanjem na običajni ter manjši obroč za košarkaša #1.

Namigi:

Za modeliranje uspešnosti uporabi Bernoullijevo porazdelitev.
Predznanje v model lahko vnesemo s pomočjo Beta porazdelitve.

44

Naš prvi model – meti na koš

45

Naš prvi model – meti na koš

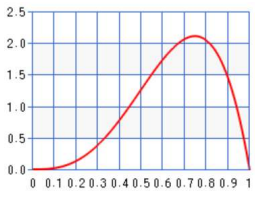
- **data**
  - n – število metov
  - y – rezultat meta (0 – neuspešen met, 1 – uspešen)
- **parameters**
  - theta – parameter Bernoulli distribucije, ki ocenjuje uspešnost
- **model**
  - predznanje (prior)
  - opis modela

```
data {
  int<lower=1> n;
  int y[n];
}

parameters {
  real<lower=0,upper=1> theta;
}

model {
  // prior
  theta ~ beta(1,1);

  for (i in 1:n) {
    y[i] ~ bernoulli(theta);
  }
}
```

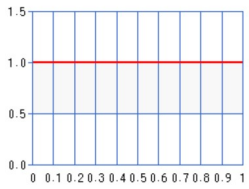
46

Prior za parameter theta

- **brez**
  - "flat" porazdelitev na intervalu [-∞, ∞]
- **beta(4, 2)**



- **beta(1, 1)**



47

Problem #2

V datoteki **temperature.csv** so podatki o temperaturi, za Slovenijo imamo podatke o povprečni temperaturi za vsak mesec med 1901 in 2015.

Zanima nas, ali je bila temperatura julija (najbolj vroč mesec) med 1970 in 1985 nižja kot med 2000 in 2015?

- *Kako prepričani smo v to trditev?*
- *Za koliko je bila temperatura nižja?*

Namig:

Za modeliranje uspešnosti uporabi normalno porazdelitev – N(μ, σ).

48

Problem #3

V datoteki **temperature.csv** so podatki o temperaturi. Za Slovenijo imamo podatke o povprečni temperaturi za vsak mesec med 1901 in 2015, za Finsko pa med 2000 in 2015.

Zanima nas:

1) Ali julijska temperatura skozi čas na Finskem narašča?
   - *Kakšno je naše zaupanje v napovedi modela, zakaj?*
   - *Kako lahko ta problem rešimo?*

2) Ali julijska temperatura skozi čas v Sloveniji narašča?

3) Kakšna bo temperatura leta 2019, kakšna 2070? Kakšna je verjetnost, da bo leta 2019, oziroma 2070, pričakovana temepratura višja od 25°C?

Namig:

Za modeliranje uspešnosti uporabi normalno linearno regresijo – normalni linearni model, kjer je μ porazdelitve linearno odvisno (a + bx) od leta (x).

49

Problem #4

Novo nastalo zagonsko podjetje nas je najelo, da jim pomagamo pri dveh pomembnih odločitvah:
(ob predpostavki, da želijo maksimizirati svoj dobiček)

1) Kam vlagati sredstva (razvoj, marketing ali administracija) ?

2) Kje naj imajo svoje prostore (na voljo imajo Florido, Kalifornijo in New York)?

S pomočjo analiziranja podatkov iz datoteke **50_startups.csv** jim pomagaj pri odločitvi. V datoteki se nahajajo podatki o 50 zagonskih podjetjih (kako so vlagali denar, kje imajo pisarno ter koliko so zaslužili).

Namig:
Nadgradi linearno normalno regresijo iz prejšnjega problema.
Odvisna spremenljivka (profit) je odvisna od več atributov (research, marketing, ...) – vhod X je torej matrika.
Za vsak atribut želimo svojo beta (b) vrednost – parameter b je torej vektor.
Iz kategorične spremenljivke (state) naredimo več binarnih spremenljivk.

50